G3078E (D) € 14,90 | CHF 27,00 | (A, B, L) € 16,00

elektor MAG

design › share › sell

578
SEIT 1961
MÄRZ/APRIL 2021
ELEKTORMAGAZINE.DE

elektor
Nr. 578 – März/April 2021

Eine komplette Ausgabe unser speziellen Gastredaktion

sparkfun

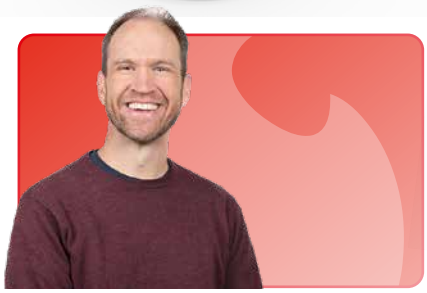
MicroMod stellt sich vor

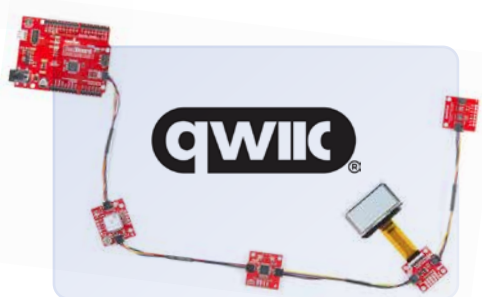
S. 10

In dieser Ausgabe

- „Hello World" vom Raspberry Pi Pico und RP2040
- Programmierung eines FPGAs
- Perfektes Einparken mit LiDAR
- Vierbeiniger Roboter selbstgebaut
- RISC-V-IoT-Entwicklung in AWS
- Vom Entwurf zum Verkauf: der SparkFun RTK Surveyor
- Entwerfen mit dem Artemis

und vieles mehr





4 198630 314904 10004

UNSER SORTIMENT VON TECHNIKERN FÜR TECHNIKER

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

Nur das Beste für Sie – von über 900 Markenherstellern.

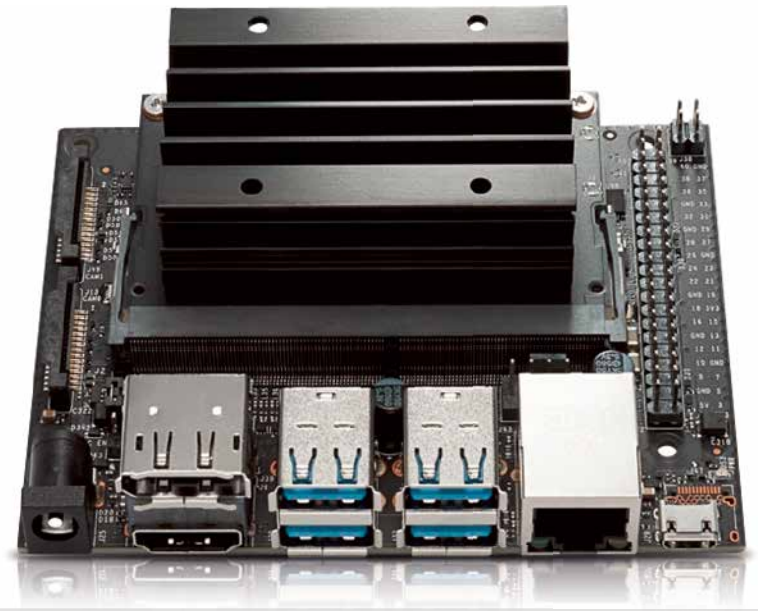
Unsere Produktmanager sind seit vielen Jahren bei reichelt tätig und kennen die Anforderungen unserer Kunden. Sie stellen ein breites Spektrum an Qualitätsprodukten zusammen, optimal auf den Bedarf in Forschung & Entwicklung, Instandhaltung, IT-Infrastruktur und Kleinserienproduktion sowie auf Maker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

MODERNE KI FÜR OBJEKTERKENNUNG & SPRACHVERARBEITUNG

JETSON NANO™ – ENTWICKLERKIT

NVIDIA

High Performance Computing
mit 472 GFLOPs bei nur 5 bis 10 Watt.



GPU
NVIDIA Maxwell™

Arbeitsspeicher
4 GB 64-bit LPDDR4

CPU
ARM Cortex-A57

Kamera
2x MIPI CSI-2 DPHY

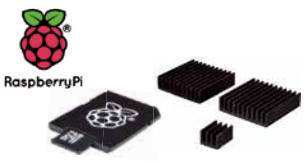
Bestell-Nr.: **JETSON NANO KIT**

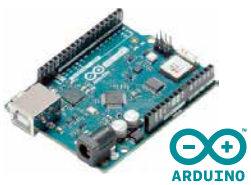
109,95

Mehr unter reichelt.de/ki

Über 1.000 Zubehör-Produkte von A wie Adapter bis Z wie ZIGBEE.

Gleich entdecken ▶
www.reichelt.de/entwicklerboards





- Top Preis-Leistungs-Verhältnis
- über 110.000 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: 3. 2. 2021

IMPRESSUM

Sonderausgabe, Gastredaktion von sparkfun (USA)

52. Jahrgang, Nr. 578
März/April 2021
ISSN 0932-5468

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

Verlag
Elektor Verlag GmbH
Kackertstraße 10, 52072 Aachen
Tel. 0241 95509190

Leitender Redakteur: Jan Buiting
Redaktion deutsche Ausgabe:
Rolf Gerstendorf, Jens Nickel (V.i.S.d.P.)

Technische Fragen: redaktion@elektor.de

Mitwirkende: C.J. Abate, Mathias Claußen, Megan Hemmings, Luc Lemmens, Chris McCarty, Justin Rajewski, Rob Reynolds, Derek Runberg, Glenn Samala, Avra Saslow, Nathan Seidle, Marcus Stevenson, Alex Wende

Übersetzung:
Sophia und Rolf Gerstendorf, Kurt Diedrich

Grafikdesign und Druckvorstufe:
Harmen Heida, Patrick Wielders

Herausgeber: Don Akkermans

Hauptsitz des Verlags
Elektor International Media
Postbus 11, 6114 ZG Susteren, Niederlande

Anzeigen:
Raoul Morreau (Leitung), 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.

© 2021 elektor international media b.v.

EDITORIAL

Mit Ingenieursgeist aus der EU & USA

Sie haben sicher sofort etwas Besonderes an dieser Elektor-Ausgabe bemerkt. Richtig: Unsere US-amerikanischen Freunde von SparkFun Electronics aus Boulder, Colorado, haben die Ausgabe als Gastredakteure zusammen mit uns erstellt. Wie kam es dazu? Die Entstehungsgeschichte dieser Ausgabe begann schon vor einigen Jahren. Während eines Besuchs der Maker Faire im Mai 2019 in San Mateo, Kalifornien, trafen sich Don Akkermans (Elektor-Geschäftsführer) und C. J. Abate (Elektor-Content-Director) mit Jahnell Pereira (Leiter der strategischen Entwicklung von SparkFun), um über eine mögliche Kooperation bei Inhalten und Produkten zu sprechen. Während ihres Treffens bei einem Espresso stellten sie schnell fest, dass die beiden Unternehmen viele Gemeinsamkeiten haben: talentierte Ingenieurteams, weit verbreitete Online-Elektronik-Shops, eine ernsthafte Leidenschaft für DIY-Elektronikprojekte und schnell wachsende Communities von neugierigen Ingenieuren und Makern. Die Gespräche in den Monaten nach diesem ersten Treffen führten dazu, dass neue SparkFun-Produkte im Elektor-Store angeboten werden. Und schließlich auch zu einem detaillierten Plan zur Zusammenarbeit bei einer vollständigen Elektor-Ausgabe. Und jetzt, nach Monaten der Entwicklung, Bearbeitung und Übersetzung, halten Sie dieses Heft in Ihren Händen! Wie Sie sehen werden, hat das Elektor-Team eng mit SparkFun zusammengearbeitet, um Projekte, Tutorials, Interviews und weitere interessante Artikel dieser besonderen Ausgabe auszuwählen, vorzubereiten und zu bearbeiten. Dies ist nur eine der vielen Überraschungen in Elektors 60. Jahr. Unser E-Commerce-Team hat zudem hart daran gearbeitet, Ihnen einen einfachen und erschwinglichen Zugang zu vielen Baugruppen und Tools zu geben, die Sie für Ihre nächsten innovativen Projekte benötigen - egal ob Sie ein Maker in Amsterdam, ein Ingenieurstudent in Paris, ein Hardware-Hacker in Cambridge oder ein Profi-Ingenieur in München sind. Viel Spaß!

C.J Abate (Content Dirctor, Elektor) und Jan Buiting (Managing Editor, Elektor)

Zunächst einmal vielen Dank an Elektor, dass wir eine Ausgabe dieses unglaublichen Magazins mitgestalten durften! Wir freuen uns sehr, in Interviews, Projekten und Artikeln der Elektor-Community mehr über SparkFun zu berichten. Wir haben es wirklich genossen, mit dem Elektor-Team zusammenzuarbeiten, um diese Elektor-Ausgabe zu produzieren - die Leute sind wirklich professionell und talentiert.
Vor kurzem hat SparkFun Electronics sein 18-jähriges Bestehen gefeiert. Es war eine lange Reise, über die wir in dieser Ausgabe der Elektor-Community erzählen möchten. Vor 18 Jahren gründete Nathan die Firma in seinem Zimmer im Studentenwohnheim. Im Laufe der Jahre ist SparkFun von einer Person auf mehr als 100 Personen und von einer „Bude" auf ein großes Geschäftsgebäude angewachsen. In diesen 18 Jahren hat sich SparkFun darauf konzentriert, das Spielen, Prototyping und Experimentieren mit fortschrittlicher Elektronik einfacher zu machen - sei es durch die Entwicklung von einfach handhabbaren Breakout-Boards, die Entwicklung von Schritt-für-Schritt-Tutorials oder das Bauen und Teilen von inspirierenden Projekten.
Diese Elektor-Ausgabe spiegelt wider, was SparkFun war, ist, und wohin die Firma gehen möchte. Tauchen Sie ein in das, was SparkFun ausmacht, durch Interviews mit uns beiden, und hören Sie, was unsere Ingenieure über ihre Arbeitsbereiche und Must-Have-Tools zu sagen haben. Unser Team teilt einige ihrer Lieblingsprojekte aus den vergangenen Jahren sowie die neuesten Projekte wie den Vierpfötigen Hund von Rob. Entdecken Sie unsere Artemis-, Qwiic- und MicroMod-Produkte und erfahren Sie, wie wir weiterhin daran arbeiten, schnelles Prototyping, Produktentwicklung und Hobbyelektronik zu unterstützen.
SparkFun Electronics freut sich sehr über die Zusammenarbeit mit Elektor und wir hoffen, dass Ihnen das Endergebnis gefällt. Viel Spaß beim Hacken!

Nathan Seidle (Gründer) und Glenn Sanala (CEO) von SparkFun

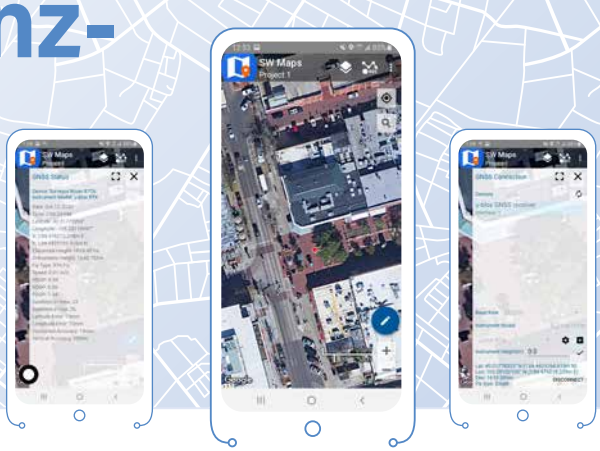
INHALT

Gastausgabe von SparkFun (USA)

52. Jahrgang
Elektor Nr. 578 - März/April 2021

Wie man eine **GNSS-Referenz-station** baut

28



Rubriken

Hintergrund

Projekte

Extra!

Vorschau

Elektor Mai/Juni 2021
Das nächste Heft ist wie immer randvoll gefüllt mit Schaltungsprojekten, Grundlagen sowie Tipps und Tricks für Elektroniker.

Aus dem Inhalt:

- 60 Jahre Elektor!
- Java auf dem Raspberry Pi
- Wärme-Suchgerät
- Lötstation 2021
- Objekt-Orientiertes Programmieren
- Raspberry Pi Pico und RP2040
- Stromversorgung mit wiederaufladbarem LiPo-Akku
- Tracking-Zeit mit IoT-Button erfassen
- WLAN-Schalter

Und vieles mehr!

Elektor Mai/Juni 2021 erscheint 6. Mai 2021.
Änderungen vorbehalten.

Q&A mit Nathan Seidle

(Gründer von SparkFun)

Die Vision & Technik hinter SparkFun

Im Jahr 2003, nur wenige Monate nach dem Platzen der Dotcom-Blase, tat Nathan Seidle - damals ein unternehmerisch denkender Student der Elektrotechnik in Boulder, Colorado - etwas Überraschendes: Er startete eine Website und begann mit dem Verkauf elektronischer Bauteile und Komponenten. Heute wächst Seidles Unternehmen weiter und hat spannende neue Produkte in Arbeit.

Von C. J. Abate (Elektor)

C. J. Abate: In den letzten Monaten habe ich mehrere bekannte Ingenieure wie Eben Upton von Raspberry Pi oder Ryan Cousins von krtkl dazu befragt, wie sich COVID-19 auf ihre Unternehmen sowie auf ihre Design- und Innovationsprozesse ausgewirkt hat. Wie hat es Ihren kreativen Designprozess beeinflusst?

Nathan Seidle: Ich denke, Reibung ist die Würze des Lebens. COVID hat sich auf viele Menschen schrecklich ausgewirkt, und es war persönlich schwierig, aber ich fand die Veränderungen auch interessant. Ich habe gelernt, dass der „Cocooning"-Effekt, der durch das individuelle Arbeiten von zu Hause aus entsteht, sehr nützlich sein kann, um sich auf schwierige technische Herausforderungen zu konzentrieren. Außerdem hat mir die Tatsache, dass ich zu reinen Videomeetings gezwungen bin, gezeigt, wie global ich arbeiten kann. Mein Team hat sich auf drei Kontinente ausgedehnt. Vor der Pandemie 2020 hätte ich das nicht geglaubt. Ich bin auch fasziniert davon, die kreativen Prozesse anderer zu beobachten. Die lokalen Unternehmen in meiner Heimatstadt Boulder waren unglaublich innovativ, weil sie dazu gezwungen waren. So schmerzhaft ein massiver Waldbrand auch ist (und COVID macht wirklich viele Dinge kaputt), bin ich gespannt, welche neuen Unternehmen jetzt eine Chance bekommen, zu gedeihen.

Abate: Kommen wir zu Ihrem Hintergrund und Ihren technischen Interessen. Wann haben Sie angefangen, sich für Elektronik und Technik zu interessieren? Wurden Sie durch einen Freund, einen Verwandten oder einen Lehrer inspiriert? Oder haben Sie Ihre Leidenschaft selbst entdeckt?

Seidle: Viele Jahre lang habe ich jeden Elektroschrott auseinandergenommen, den ich in die Finger bekam. Wie *Rosie Revere, Engineer* habe ich ständig Müll aus Mülleimern geholt und meine Eltern damit in den Wahnsinn getrieben. Ich glaube, ich unterschied mich aber ein wenig von anderen Müllsammlern, da ich dann versuchte, diese Dinge an meine Freunde in der Schule zu verhökern. Ich habe Programmierkabel für TI-Taschenrechner zusammengelötet, damit man Spiele herunterladen konnte. Als ein Kabel von TI 100 Dollar kostete, habe ich meinen Hack für 25 Dollar an Freunde verkauft. 1994 startete ich eine Mailbox und versuchte, eine Zugangsgebühr zu erheben, damit ich meinen Eltern die zusätzlichen Telefongebühren zurückzahlen konnte. Ich sparte 500 $ und kaufte 1995 einen CD-Brenner (als CD-Rs 8 $ pro Stück kosteten!), damit ich Kopien (OK, seien wir ehrlich, Raubkopien) von Software und Musik an meine Freunde verkaufen konnte. Ich wurde nie wirklich von einer einzelnen Person inspiriert, sondern ich denke, dass ich immer gelernt und darüber nachgedacht habe, wie ich das, was ich liebe, auch zu meinem Job machen kann.

Abate: Können Sie uns von Ihrem ersten Mikrocontroller-basierten Projekt erzählen?

Seidle: Im Jahr 2002 machte ich gerade meinen Abschluss in Elektrotechnik, aber alle meine Kurse waren reine Theorie. Ich konnte ein Bode-Diagramm erstellen und eine RC-Kombination für ein Notch-Filter berechnen, aber tatsächlich etwas bauen? Das lag außerhalb des Rahmens aller meiner Kurse.

Mein Problem war, dass ich ein Gerät für den Rudersport bauen wollte, das ein paar Sensoren auslesen und Sprache verstärken konnte. Dazu begab ich mich erstmalig in der Welt der Mikrocontroller, um genau zu sein, das *Board of Education* von Parallax. Diese BASIC-Briefmarke war bahnbrechend für die Mikrocontroller- und Bildungswelt. Ich erinnere mich an das erste Mal, dass ich eine LED mit BASIC zum Blinken brachte. Ich hatte Kontrolle über die physikalische Welt! Ich konnte alles tun! Dann versuchte ich, einen Messwert von meinem Sensor zu nehmen und ihn in eine Fließkommazahl umzuwandeln (Fließkommaoperationen benötigen eine Menge Code). Damit war meine Briefmarke am Limit und ich konnte keine der anderen Funktionen ausführen, die ich für mein Rudergerät brauchte.

Abate: Was hat Sie dazu bewogen, die Firma im Jahr 2003 zu gründen?

Seidle: Im Herbst 2002 half ich einem Freund, eine Fernsteuerung für einen Rohrkriechroboter zu bauen, der das Innere von vertikalen Stahlrohren inspizieren sollte. Ich musste ein paar Joysticks für die Richtung und einen Schieberegler für die Geschwindigkeit in die Software einbauen und verwendete einen PIC, um alle erforderlichen Analog-Digital-Wandlungen durchzuführen und die Werte in digitale Signale umzuwandeln, die der Schrittmotor-Controller verstehen konnte.

An einem besonders aufregenden Arbeitstag hatte ich den PIC dann so weit, dass er die Joysticks auslas und den Motor drehte! Als ich ein paar Dinge auf meinem Schreibtisch neu anordnete, kam der unter Spannung stehende Programmierer in Kontakt mit einer Schraube oder einem abgeknipsten LED-Beinchen oder einem Stück Draht, ich bin mir nicht sicher, aber im Bruchteil einer Sekunde gab es einen Funken, ein bisschen Rauch, und mein 150 $ teurer Programmierer war tot.

> *Ich entwerfe und entwickle leidenschaftlich gerne neue Produkte, die Elektronik zugänglich machen.*
>
> Nathan Seidle

Ich brauchte einen neuen Programmierer, also ging ich ins Internet und suchte nach dem billigsten PIC-Programmierer, der zu finden war. Es gab viele zur Auswahl, aber eine Firma namens Olimex tauchte immer wieder in den Suchergebnissen auf. Sie hatten gute Hardware und sehr niedrige Preise, aber ihre Website war mir unheimlich. Es gab keine Möglichkeit, online zu bestellen, und nach ein paar E-Mails, um meine Bestellung aufzugeben, stellte ich fest, dass die Firma ihren Sitz in Bulgarien hatte. Ich lernte etwas über die große weite Welt, aber hatte keine Ahnung von Bulgarien (wo ist das?), geschweige denn, wie man Geld dorthin schickt. Der ganze Bestellvorgang war umständlich und schwierig und brachte mich dazu, etwas auszuprobieren: Anstatt nur einen Ersatzprogrammierer für mich zu bestellen, würde ich gleich ein paar Programmierer mehr und andre Produkte dieser Firma Olimex ordern, um sie dann an Freunde an der Uni weiter zu verkaufen. Vielleicht könnte ich ja durch den Verkauf dieser Teile meine eigene Sucht nach neuen Projekten und Entwicklungen finanzieren!

In den Weihnachtsferien 2002 erstellte ich mit einer E-Commerce-Software von der Stange eine Website, fing an, alle bei mir eintreffenden Programmierer, Kabel und Teile zu fotografieren und erledigte den Papierkram beim Staat Colorado, um ein Geschäft betreiben zu dürfen. Der Tag, an dem mein Programmierer in Rauch aufging, war noch frisch in meiner Erinnerung. Dieser Moment, wenn die Funken fliegen und man dennoch Spaß hat, blieb bei mir hängen. Der Domainname SparkFun.com war verfügbar und SparkFun Electronics war geboren. Am 3. Januar 2003 hatte ich mein Einzelhandelssteuerzertifikat vom Staat Colorado in der Tasche und war bereit, Geschäfte zu machen. Innerhalb weniger Stunden hatte ich meine erste Bestellung, und mir wurde schnell klar, dass ich einen Karton brauchen würde, um die Bestellung zu versenden. Es dauerte nicht allzu lange, bis ich auch die erste internationale Bestellung erhielt und lernen musste, wie man international versendet.

Abate: Am Anfang, als Sie noch ein Twen waren und SparkFun leiteten, wie haben Sie da die Technik, die Entwicklung neuer Produkte und die tägliche Arbeit eines wachsenden Unternehmens unter einen Hut gebracht?

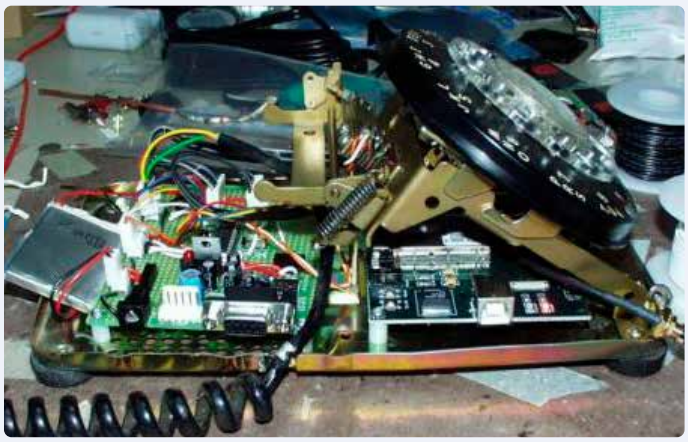
Seidle: Anfangs bestand SparkFun nur aus mir. Ich kümmerte mich um die Website, die Produktbeschaffung, die Fotos, den Versand von Bestellungen, die Beantwortung von Telefonanrufen, das Design und die Herstellung von Produkten, um alles. Mit der Zeit merkte ich, dass SparkFun schnell wuchs und ich Hilfe brauchen würde, um die Bestellungen auszuliefern, also stellte ich meine ersten Mitarbeiter ein. Im Jahr 2006 zogen wir in 2300 m² große Geschäftsräume um und im Jahr 2008 hatte SparkFun 50 Mitarbeiter. SparkFun wuchs weiterhin schnell, so dass ich mich (leider) von der Produktentwicklung wegbewegte zum Leiter eines Unternehmens. Bevor ich die Zügel an Glenn übergab, habe ich zwischen all den Verpflichtungen als Geschäftsführer wöchentlich höchstens vier oder fünf Stunden Zeit für das technische Entwickeln gefunden.

Abate: Bevor Sie „Founder & Engineer“ wurden, haben Sie etliche Blogbeiträge als „Nate the Engineer“ geschrieben. Was war am meisten herausfordernd bei diesem Übergang?

Seidle: Während der 13 Jahre, in denen ich SparkFun leitete, entfernte ich mich immer weiter von der Entwicklung neuer Produkte und dem Bloggen, den beiden Dingen, mit denen SparkFun begann. Also gaben wir im Jahr 2015 bekannt, dass wir einen neuen Geschäftsführer einstellen möchten. Der sechs Monate währende Weg, unseren Geschäftsführer Glenn zu finden, war voller Herausforderungen, aber letztendlich war es der richtige Schritt. Ich wurde wieder zu „Nate the Engineer“ und konnte unser „Experimentallabor“ SparkX gründen. Meine wahre Leidenschaft ist es, neue Produkte zu entwerfen und Elektronik zugänglich zu machen. SparkX und Glenn an der Spitze von SparkFun erlaubten es mir, zu diesen Wurzeln zurückzukehren.

Während der Suche nach einem neuen Geschäftsführer waren die größten und wichtigsten Erkenntnisse, dass:

- die Einstellung eines neuen Geschäftsführers eine Herztrans-



Nathans Drehscheibentelefon-Projekt war ein Hit in der SparkFun-Community. Lesen Sie dazu „Retronik" an anderer Stelle dieser Ausgabe.

plantation ist. Es ist unglaublich riskant und stresst die Leute.

› man klugen Rat bei klugen Leuten suchen, aber auch den eigenen Weg gehen soll. Irgendwann hat mir ein Berater gesagt, dass die Einstellung eines Geschäftsführers extrem riskant und töricht sei. Das war niederschmetternd, da wir kurz vor dem Abschluss der Bewerbungsgespräche standen und ich extrem hart daran arbeitete, SparkFun an jemanden zu übergeben, der das Unternehmen erfolgreich führen konnte. Ich konnte nicht mehr lange weitermachen, aber mir zu sagen, dass ich völlig falsch lag, war schrecklich. Letztendlich entschied ich mich, diesem Rat nicht zu folgen.

So wurde Glenn der nächste Geschäftsführer bei SparkFun! Glenn hat mehr als 20 Jahre Erfahrung in der Technologie- und Fertigungsbranche und führte Organisationen und Firmen durch die Herausforderungen, die mit großem organischem Wachstum einhergehen, und die kulturellen Herausforderungen, die mit strategischen Übernahmen einhergehen. Er liebt es, an Unternehmen zu arbeiten, so wie ich es liebe, Bauteile auf einer Platine gut aussehen zu lassen.

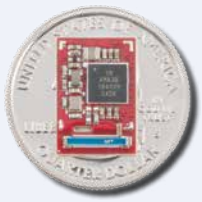
Abate: Ich bin sicher, dass viele Ihrer Produkte eine interessante Geschichte haben. Warum haben Sie Artemis entwickelt?
Seidle: Im Oktober 2018 wurden wir den Leuten von TensorFlow vorgeschlagen, um ein Low-Power-Board mit dem neuen Apollo3 von Ambiq zu entwickeln. Nach einem kurzen Blick auf das Datenblatt war klar, dass dieses IC ebenso herausfordernd wie spannend ist! 1 MB Flash, fast 400 k RAM und ein außergewöhnlich niedriger Strombedarf ließen uns von den Möglichkeiten weit über denen des alten Arduino Uno träumen. Gleichzeitig bedeutete ein 81-Ball-BGA mit 0,5-mm-Raster, dass wir unsere Platine völlig neu gestalten mussten. Das Projekt wurde SparkFun-Edge genannt (nach dem Begriff „Edge Computing") und wir machten uns daran, dieses neue IC kennenzulernen.
Wir waren in der Lage, den Edge pünktlich und (fast) innerhalb des Budgets für die TensorFlow-Konferenz von Google zu liefern. Was mich schmerzte, war, dass wir die „sehr engen" Spezifikationen nur für ein paar Quadratmillimeter um den Apollo3 herum brauchten, der Rest der Platine aber trivial war und mit normalen 8-mil-Leiterbahnen geroutet werden konnte. Natürlich mussten wir die teuren Spezifikationen für die gesamte Platine bezahlen. Die Edge-Platine ist relativ groß, so dass die Kosten dafür fast dem Preis für das Haupt-IC entsprachen. Da die Herstellung einer solchen komplizierten Leiterplatte Zeit in Anspruch nahm, bedeutete dies auch eine lange Vorlaufzeit für das Edge-Board. Alle Änderungen, die wir an der Leiterplatte vornehmen wollten und mussten, würden viele Monate und Hunderte von Dollars kosten, um sie zu testen.
Aber was wäre, wenn wir ein SMD-Modul mit dem Apollo3 entwickeln? Dank der vielen „Was-wäre-wenn"-Vorschläge beschloss SparkFun dann, mit dem Apollo3 weiter zu basteln. Aus dieser Tüftelei entstanden dann der Artemis-Chip und die auf ihn folgenden Breakout-Boards.

Abate: Das Artemis-Modul wurde im August 2019 von der FCC zertifiziert. Wie verlief der Zertifizierungsprozess?
Seidle: SparkFun hat sich jahrelang davor gescheut, ein Produkt vollständig FCC-zertifizieren zu lassen. Der Prozess war unübersichtlich und schien unerschwinglich teuer zu sein. Wir haben in der Vergangenheit über die FCC-Anforderungen für Produkte und Hobby-Projekte geschrieben, aber mit der Entwicklung von SparkFun-Edge und Artemis war klar, dass wir uns auf dieses unbekannte Terrain wagen wollten und müssten.
Der Zertifizierungsprozess war ein Lernprozess für SparkFun: von der Suche nach einer Testfirma über das Einholen von Angeboten bis hin zu den von der Testeinrichtung benötigten Formularen und Komponenten. Der Zertifizierungsprozess selbst dauerte, nachdem wir eine Testeinrichtung gefunden hatten, etwa zweieinhalb Monate. Die Testfirma war sehr hilfreich bei der Rückmeldung und hat uns durch die notwendigen Änderungen in den Formularen geführt. Ich kann nicht mit Sicherheit sagen, ob SparkFun noch einmal einen FCC-Zertifizierungsprozess durchlaufen wird, aber wenn wir das tun, wissen wir jetzt, wie es geht.

Abate: Können Sie ein paar Details über ein oder zwei Projekte verraten, an denen Sie gerade arbeiten?
Seidle: Ich unterstütze und entwickle MicroMod und ALC weiter. Ich habe auch am SparkFun RTK-Surveyor gearbeitet. Ich kann noch nicht verraten, woran ich arbeite, aber halten Sie die Augen offen nach einigen Ergänzungen für bestimmte Produktlinien.

Abate: Haben Sie ein Lieblingsdesign oder -projekt?
Seidle: Das ist, als würde man ein Lieblingskind auswählen. Ich glaube nicht, dass ich einen Lieblingsentwurf habe, aber ich lache sehr gerne. Das Rotary-Cell-Phone hat die Leute umgehauen und zum Lachen gebracht. In ähnlicher Weise arbeite ich an einem Geschenk für einen Freund, der kürzlich in der Marine zum Commander befördert wurde. Aufgrund seines Ranges müssen sie in alter Tradition der Marine nun jedes Mal eine Glocke läuten, wenn er ein Schiff betritt oder verlässt. Also habe ich seiner Frau zum Spaß eine WLAN-fähige Essensglocke gebastelt, damit sie sie – vom Handy gesteuert - läuten lassen kann und ihn damit jedes Mal, wenn er einen Raum in ihrem Haus betritt, zu Tode ärgert.

210018-02



Das Artemis-Modul in wahrer Größe, mit entfernter Metallabschirmung.

Was Sie auch noch über Nathan wissen sollten...

Nathan stellt SparkFun-Produkte auf dem YouTube-Kanal des Unternehmens vor.

Erste Liebe - Hardware oder Software?
Löten. Programmieren war interessant, aber die Idee, Metall zu schmelzen und es elektrisch und physikalisch nach meinen Wünschen und Bedürfnissen einzusetzen, war wirklich ein Game Changer.

Ein Laborgerät, ohne das Sie nicht leben könnten?
Ganz einfach, die Klebepistole. Es ist die dritte Hand, die sich nie bewegt, super billig und in der Lage, drei oder vier Dinge festzuhalten, wenn man sie braucht.

Lieblingsingenieur?
Inspector Gadget. Wir alle wollen „Go-go-Gadget-Arme".

Was lesen Sie gerade?
Ready Player Two von Ernest Cline. Davor *Network Effect* von Martha Wells, bei dem ich laut lachen musste und damit meine beiden Kinder verschreckte. „Warum lachst du über Roboter, Daddy?"

Welches Thema im Bereich Elektronik begeistert Sie derzeit am meisten?
Triangulation basierend auf der Schallgeschwindigkeit, mit GNSS als Zeitreferenz.

Wie viel Schlaf bekommen Sie pro Nacht?
Idealerweise 8,5 Stunden, aber meine Kinder haben mich auf etwa 7 Stunden reduziert.

Hinweis: Sie können Nathans Blogbeiträge und Tutorials unter *https://www.sparkfun.com/users/7185* lesen.

Ganz einfach Preis und Verfügbarkeit aller Bauteile prüfen, die Sie benötigen

Price Availability Assistant

STOCK • PRICE • BUY

MOUSER ELECTRONICS

mouser.de/price-availability-assistant

Erste Schritte mit MicroMod

Schauen Sie sich das Poster auf Seite 104 an!

Von Nathan Seidle (SparkFun)

MicroMod ist eine kompakte Schnittstelle zum Anschluss eines Mikrocontrollers an verschiedene Peripheriebausteine. Generell kann man sich das MicroMod-System wie ein Gehirn vorstellen, das auf eine Trägerplatine gesteckt wird.

Eine MicroMod-Prozessorplatine ist ungefähr 22 × 22 mm² groß und kann in jede MicroMod-Trägerplatine eingesetzt werden. Eine kleine Schraube hält die Prozessorplatine an ihrem Platz. Während der M.2-Standard [1] ursprünglich für den Anschluss von Computer-**Peripherie** konzipiert war (der User kann verschiedene NVMe-SSD-Festplatten auf das Motherboard stecken), ist der MicroMod-Standard für den problemlosen Tausch von **Controllern** konzipiert. Der User kann mit einem leistungsstarken Controller beginnen und dann zu stromsparenden Varianten übergehen, um die Akkulaufzeit zu verlängern.

Empfohlene Lektüre

Wenn Sie mit dem MicroMod-Ökosystem nicht vertraut sind, empfehlen wir Ihnen die Lektüre von [2], um einen Überblick zu erhalten. Die darin enthaltenen Konzepte werden in den folgenden Tutorials erläutert.

- *Serial Peripheral Interface (SPI).* SPI wird üblicherweise verwendet, um Mikrocontroller mit Peripheriebausteinen wie Sensoren, Schieberegistern und SD-Karten zu verbinden [3].
- *Pulse Width Modulation*. Eine Einführung in das Konzept der Pulsweitenmodulation in [4].
- *Logic Levels.* Lernen Sie den Unterschied zwischen 3,3-V- und 5-V-Bausteinen und Logikpegeln kennen [5].
- *I²C.* Eine Einführung in I²C, eines der aktuell wichtigsten Kommunikationsprotokolle im Embedded-Bereich [6].

Wie funktioniert es?

MicroMod nutzt den M.2-Steckverbinder und die M.2-Spezifikation [7], um viele verschiedene Baugruppen anschließen zu können und die Kosten für Steckverbinder zu senken. Alle MicroMod-„Gehirne" besitzen die gleiche Anschlussbelegung. So befinden sich zum Beispiel die I²C-Pins des MicroMod-ESP32 an der gleichen Position wie die I²C-Pins beim MicroMod-Artemis.

So ermöglicht jede Variante der MicroMod-Trägerplatinen dem Anwender den Zugriff auf unterschiedliche Technologien. Da der MicroMod-Steckverbinder standardisiert ist, kann der Controller einfach und schnell in Bezug auf Leistungsfähigkeit, Stromverbrauch und drahtlose Konnektivität ausgetauscht werden. Der User kann beispielsweise mit dem MicroMod-Artemis und einem RFID-Trägerboard beginnen. Dann könnte er entscheiden, dass er WLAN für sein Projekt benötigt und zum MicroMod-ESP32 wechseln möchte. Er kann sofort WLAN-Funktionalität hinzufügen, ohne die zugrunde liegende Hardware zu verändern. Die MicroMod-Schnittstelle ist wie folgt definiert:

- SparkFun MicroMod Schnittstelle v1.0 - Pinbelegung [8]
- SparkFun MicroMod Interface v1.0 - Pin-Beschreibungen [9]

Hardware-Übersicht

Q. Welchen Stecker und welche Schlüssel verwendet das MicroMod?

A. MicroMod verwendet den in **Bild 1** dargestellten **M.2-Verbinder**. Dies ist derselbe Anschluss, der auf modernen Motherboards und in Laptops zu finden ist. Wir empfehlen den Verbinder mit **4,2 mm** Höhe. Die Firma TE Connectivity stellt den 2199230-4 her, der weit verbreitet und zu einem vernünftigen Preis erhältlich ist [10]. Sie können auch das MicroMod-DIY-Carrier-Kit bestellen, das fünf Stecker, Schrauben und reflowfähige Abstandshalter enthält [11].

Es gibt verschiedene Positionen für den „Schlüssel" am M.2-Verbinder, der verhindert, dass eine inkompatible Platine einsteckt wird. Der MicroMod-Standard verwendet den E-Schlüssel, weicht aber vom M.2-Standard ab, indem er die Montageschraube 4 mm zur Seite verschiebt. Der E-Schlüssel ist relativ weit verbreitet, so dass man

ein M.2-kompatibles WLAN-Modul einsetzen könnte, aber da die Befestigungsschraube nicht passt, ist der User nicht in der Lage, ein inkompatibles Modul (außer mit Gewalt) in die MicroMod-Trägerkarte zu stecken.

Q. Was ist ein Prozessor-Board?

A. Wie in **Bild 2** dargestellt, ist jede Prozessorplatine etwa 22 × 22 mm² groß und verfügt über einen Mikrocontroller oder Prozessor. Die Anschlüsse des Controllers sind gemäß der MicroMod-Pinout-Spezifikation an den Platinenrand geführt, der in den M.2-Kantenverbinder passt.

Jede Prozessorplatine benötigt zur Programmierung nur USB-D+ und USB-D-. Bei einem Controller ohne USB-Unterstützung muss die Schnittstelle nachgerüstet werden, wie dies bei der Artemis-Prozessorplatine mit dem USB-nach-Seriell-Chip CH340E der Fall ist. Auf jeder Prozessorplatine gibt es auch eine On-Board-Status-LED, die nicht an den Platinenrand geroutet ist.

Hinweis: Die MicroMod-Spezifikation verschiebt die Schraubenposition von der Mittellinie der Platine um 4 mm nach rechts. Dies soll verhindern, dass man eine der immer populärer werden Platinen mit M.2-Anschluss (WLAN-Karten, SSDs, Mobilfunkmodems und so weiter) mit einer MicroMod-Platine verwechselt. Spätestens, wenn die Schraube nicht passt, dürfte der Irrtum augenscheinlich werden! Die MicroMod-Spezifikation kann in Zukunft andere Größen umfassen und Benutzer können auch gerne eigene Prozessorplatinen erstellen, aber beachten Sie, dass die Abstandsbohrung auf den meisten Trägerplatinen so angeordnet ist, dass sie in den 2222-MicroMod-Schlüssel passt.

Q. Wie ist die MicroMod-Schnittstelle belegt?

A. Die MicroMod-Schnittstelle ist wie folgt definiert:

- SparkFun MicroMod Interface v1.0 - Pinbelegung [8]
- SparkFun MicroMod Interface v1.0 - Pin-Beschreibungen [9]

Beim MicroMod-Formfaktor sind nicht alle Pins zwingend angeschlossen. Bitte lesen Sie dazu die Dokumentation Ihres Controllerboards. Die Referenzdaten finden Sie online:

- MicroMod Allgemeine Pinbelegungstabelle [12]
- MicroMod Allgemeine Pin-Beschreibungen [13]

Jeder Pin am M.2-Verbinder besitzt eine bestimmte Funktion. Es gibt zusätzliche Regeln in der MicroMod-Spezifikation, um plattformübergreifende Kompatibilität zu gewährleisten. Im Extremfall werden maximal 49 GPIOs unterstützt, aber normalerweise konzentriert sich MicroMod auf Schnittstellentypen und -positionen. Wenn zum Beispiel eine Trägerplatine PWM-Fähigkeit benötigt, sollte die Trägerplatine Pins 32 (PWM0) und Pin 47 (PWM1) nutzen, da diese am besten PWM unterstützen.

Unterstützte Schnittstellen:

- USB für Programmierung und serielles Debugging
- 2× Analog fest zugeordnet
- 2× PWM fest zugeordnet
- 2× Digital I/O fest zugeordnet
- 12× GPIO
- 2× I²C
- 2× SPI
- 2× UART
- SDIO
- USB-HOST
- CAN
- SWD
- PDM / PCM / I2S
- ADC mit Differenzeingängen

Zwölf GPIOs, das klingt vielleicht nicht nach viel, aber wenn alle anderen Schnittstellen angeschlossen sind (UART, SPI, I2C, PWM, ADC), dann sollten zwölt GPIOs für die meisten verbleibenden Anwendungen völlig ausreichen.

Bild 1. M.2-Kantensteckverbinder Vorderansicht (oben) und Rückansicht (unten).

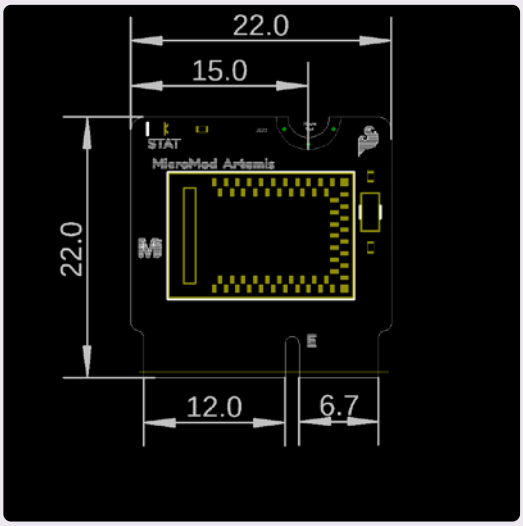


Bild 2. Jede Prozessorplatine folgt dem M.2-Standard „2222" (22 × 22 mm² Gesamtgröße).

Anschließen der Hardware

Um mit MicroMod loszulegen, benötigen Sie sowohl ein Prozessorboard als auch ein Trägerboard. Hier verwenden wir das Artemis MicroMod Processor Board mit dem Machine Learning Carrier Board. Stecken Sie, wie es in **Bild 3** zu sehen ist, das Prozessor-Board richtig herum und in einem Winkel von etwa 25 ° schräg in den M.2-Anschluss des Trägerboards, drücken Sie es sanft herunter und fixieren Sie es vorsichtig mit der Kreuzschlitzschraube in der Abstandshalter-Mutter auf der Trägerplatine (**Bild 4**). Verwenden Sie dazu den SparkFun-Mini-Schraubendreher [14] oder das schickere Taschenschraubendreher-Set [15] oder aber irgendeinen anderen Kreuzschlitzschraubendreher der Größe Ph0 oder PH1. Sobald die Platine fest sitzt, sollte Ihr zusammengebautes MicroMod-System so wie in **Bild 5** aussehen!

Hinweis: Es gibt wegen des Schlüsseleinsatzes ohnehin keine Möglichkeit, die Prozessorplatine falsch herum einzusetzen, da der Schlüssel des M.2-Verbinders dies verhindert, aber als zusätzliche Sicherheitsmaßnahme ist die Montageschraube versetzt, so dass Sie nicht in der Lage sind, irgend ein falsches Board auf der Trägerplatine zu fixieren. Noch ein **Hinweis**: Wenn Sie noch nie einen CH340 an Ihren Computer angeschlossen hatten, müssen Sie möglicherweise Treiber für diesen USB-Seriell-Wandler installieren. Hilfe bei der Installation finden Sie in *How to Install CH340 Drivers* [16].

Entwerfen mit MicroMod

Kann ich meine eigene MicroMod-Prozessorplatine herstellen?

A. Auf jeden Fall. SparkFun ist ein Open-Source-Hardware-Unternehmen und patentiert diese Schnittstelle nicht. Alles, worum wir Sie bitten, ist, dass Sie die Spezifikation nicht verfälschen, die Regeln befolgen und versuchen, die Community nicht durch die Einführung konkurrierender oder ähnlicher, nur teilweise kompatibler Schnittstellen zu verwirren.

Wir empfehlen Ihnen, von einem unserer Open-Source-Prozessorboard-Designs auszugehen, die derzeit im Eagle-PCB-Format vorliegen. Wenn Sie ein andere Software zur Platinenproduktion verwenden und Ihren Entwurf als Referenzdesign einreichen möchten, lassen Sie es uns bitte unter [17] wissen! Die folgenden Dateien finden Sie auf Github.

- MicroMod ESP32 Processor Board [18]
- MicroMod SAMD51 Processor Board [19]
- MicroMod Artemis Processor Board [20]

Wie bereits erwähnt, beschreibt der Artikel *Designing with MicroMod* [21] detailliert, wie man ein gutes Prozessor- und Trägerboard erstellt.

Kann ich meine eigene MicroMod-Trägerplatine herstellen?

A. Auf jeden Fall! An dieser Stelle wird es nämlich richtig spannend. Wir haben eine Reihe von Ressourcen zur Verfügung gestellt, darunter einen Verbinder-Footprint und ein Symbol für Eagle PCB (**Bild 6**). Wir haben mehrere Trägerplatinen bereits entworfen und als Open-Source zur Verfügung gestellt, so dass Sie diese als Referenzdesigns und Ausgangspunkte verwenden können. Wir können es kaum erwarten zu sehen, was Sie daraus machen!

Wir empfehlen, mit einer unserer Open-Source-Trägerplatinen zu beginnen. Derzeit liegen all diese Dateien im Eagle-PCB-Format vor. Wenn Sie ein andere Software zur Platinenproduktion verwenden und Ihren Entwurf als Referenzdesign einreichen möchten, lassen Sie es uns bitte unter [17] wissen! Die folgenden Dateien finden Sie auf Github.

- MicroMod All The Pins (ATP) Carrier Board [22]
- MicroMod Data Logging Carrier Board [23]
- MicroMod Machine Learning Carrier Board [24]
- MicroMod Input and Display Carrier Board [25]

Bild 3. Das MicroMod-Prozessorboard wird in einem Winkel von etwa 25 Grad eingeschoben.

Bild 5. Fertig zum Einsatz, es kann losgehen!

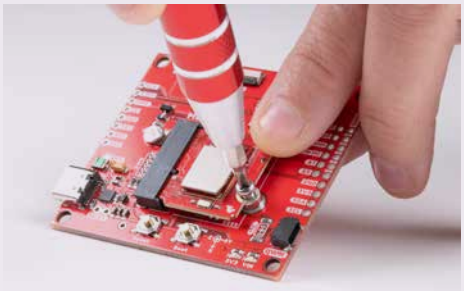


Bild 4. Halten Sie die MicroMod Prozessorplatine fest und ziehen Sie die Kreuzschlitzschraube vorsichtig an.

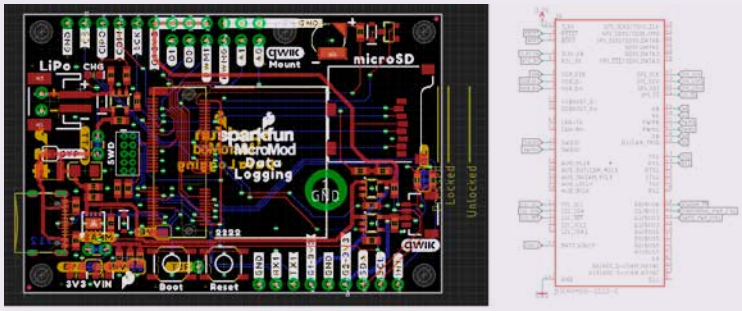
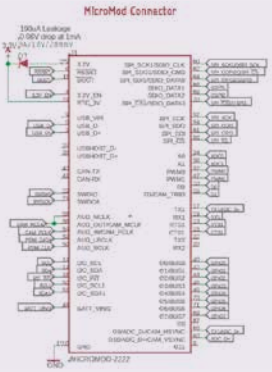


Bild 6. Fertiges Platinenlayout mit Schaltplan für eine Trägerplatine im Eagle-PCB-Format.

Tutorial

Entwerfen mit MicroMod

In diesem Tutorial lernen Sie die Spezifikationen des MicroMod-Prozessors und der Trägerplatine kennen und erfahren, wie Sie den MicroMod-Formfaktor in Ihre eigenen Leiterplattendesigns einbinden können! Weiter zu:
https://learn.sparkfun.com/tutorials/designing-with-micromod



Anschluss der Prozessor-Platine MicroMod SAMD51

Dieses Tutorial behandelt die Grundfunktionen der Platine MicroMod SAMD51 und zeigt die Besonderheiten des ARM-Cortex-M4F Entwicklungsboards. Weiter zu:
https://learn.sparkfun.com/tutorials/micromod-samd51-processor-board-hookup-guide

Anschluss der MicroMod-Datenlogger-Trägerplatine

Beginnen Sie mit der anpassbaren MicroMod-Datenerfassung mit dem Data Logging Carrier Board. Besuchen Sie die Seite:
https://learn.sparkfun.com/tutorials/micromod-data-logging-carrier-board-hookup-guide

Anschluss des MicroMod Machine Learning Carrier Boards

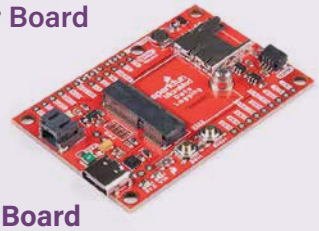
Mit diesem Tutorial zu unserem Machine Learning Carrier Board werden Sie zum Hacker! Gehen Sie auf:
https://learn.sparkfun.com/tutorials/micromod-machine-learning-carrier-board-hookup-guide

Passende Produkte

Suchen Sie nach den in diesem Artikel erwähnten Produkten? SparkFun und Elektor haben sie!
www.elektormagazine.de/esfe-en-micromod6

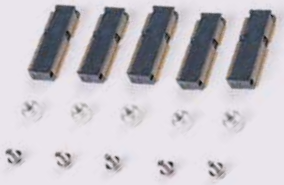
Micromod Artemis-Prozessorplatine

Machine Learning Carrier Board



MicroMod Data Logging Carrier Board

Input and Display Carrier Board



MicroMod DIY Carrier Kit (5er-Pack)

Zusätzlich haben wir den Artikel *Designing with MicroMod* [21] geschrieben, der sich ausführlich mit der Erstellung guter Prozessor- und Carrier-Boards beschäftigt.
Beachten Sie beim Entwurf Ihrer eigenen Trägerplatine die folgenden Faustregeln:

- Alle Trägerplatinen müssen eine geregelte Versorgung von 3,3 V/1 A zur Verfügung stellen.
- Alle Trägerplatinen müssen die Anschlüsse USB-D+ und USB-D- für die Programmierung bereitstellen.
- Nicht alle Prozessorboards haben Anschlüsse für jedem Pin.
- Die Signale A0/1, PWM0/1 und D0/1 sollten von jedem Prozessorboard unterstützt werden, so dass Sie darauf vertrauen können, dass diese Pins verfügbar sind.

UART1-, SPI- und I²C-Anschlüsse sind sehr verbreitet und auf fast jedem Prozessor-Board vorhanden, aber andere Signale nicht. Wenn Ihr entworfenes Trägerboard einen zweiten I²C-Port benötigt, sollten Sie darauf achten, dass das eingesetzte Prozessorboards dies auch unterstützt.
Um Ihren Einstieg in den Entwurf einer eigenen Trägerplatine zu erleichtern, haben wir das *MicroMod DIY Carrier Kit* [26] (**Bild 7**) zusammengestellt, das je fünf Steckverbinder, Schrauben und Abstandshalter-Muttern enthält, so dass Sie alle „Spezialteile" für Ihre eigene Trägerplatine in den Händen halten.
Der M.2-Steckverbinder weist ein Rastermaß von 0,5 mm auf und besitzt Zapfen zur leichteren Ausrichtung. Handlöten oder Reflow-at-home ist zwar prinzipiell möglich, aber wir empfehlen die Verwendung eines Edelstahl-Stencils (kein Polyester!) und eines hochwertigen Reflow-Ofens, um Lötbrücken zu zu vermeiden.

Q. Was ist mit der Wärmeabfuhr?

A. Einer der Vorteile des M.2-Standards ist die Möglichkeit, Bauteile unter dem Modul zu platzieren. Damit können wir auch Kühlkörper für unsere Mikrocontroller hinzufügen! Aus diesem Grund empfehlen wir auch einen Steckverbinder mit 4,2 mm Höhe. TE Connectivity stellt den 2199230-4 [27] her, der weit verbreitet und zu einem vernünftigen Preis erhältlich ist.

Q. Was ist, wenn ich viele GPIOs brauche?

A. Es gibt Anwendungen, bei denen ein User mehr als zwölf GPIO benötigt. Die MicroMod-Spezifikation ist flexibel. Wenn Sie ein MicroMod entwerfen möchten, an dem nur wenig Peripherie angeschlossen ist (zum Beispiel nur UART und I²C) und der Rest für GPIO übrig bleiben soll, geht das in Ordnung. Ihre Trägerplatine würde die UART- und I²C-Pins an den normalen Positionen und die GPIOs an nicht standardmäßigen Positionen verwenden. Dies würde zwar verhindern, dass andere MicroMods vollständig kompatibel sind und vielleicht ein oder zwei der MicroMod-Artemis Ihre Trägerplatine nicht ansteuern können, aber es ist grundsätzlich erlaubt. Sie, der Entwickler, müssen nur über die Kompromisse nachdenken, die Sie eingehen.
Wir haben eine ausführliche Anleitung für die Erstellung eines MicroMod-Prozessorboards geschrieben und wollen hier nur die Grundprinzipien skizzieren:

- Verbinden Sie die dedizierte Hardware des Mikrocontrollers mit den verfügbaren I²C-, SPI-, UART-, USB-, USB_HOST-, CAN-, SDIO- und JTAG-Pins, die auf den MicroMod-Platinenkantenverbinder geführt sind.
- Als nächstes sollten A0/A1 an dem MicroMod-Kantenverbinder den Pins des Mikrocontrollers zugewiesen werden, die ausschließlich ADC sind (also keine PWM-Fähigkeit).
- PWM0/PWM1 sollten Pins zugewiesen werden, die ausschließlich PWM sind (also keine ADC-Fähigkeit).
- D0/D1 sollten Pins zugewiesen werden, die ausschließlich GPIO sind (keine ADC- oder PWM-Fähigkeit).
- Die restlichen Pins sollten Gx zugewiesen werden, wobei ADC- und PWM-fähige Pins eine Priorität (0, 1, 2, et cetera) erhalten.
- Die Absicht dabei ist, PWM-, ADC- und Digital-Pin-Funktionalität an diesen spezifischen Pins zu garantieren. Gx-Pins hingegen besitzen keine ADC/PWM-Funktion.
- Wenn dem Mikrocontroller eine bestimmte Pin-Funktion fehlt und nur GPIO-Pins übrig bleiben, können diese mit GPIO „überschrieben" werden. Zum Beispiel kann CTS/RTS in GPIO umgewandelt werden, wenn der Mikrocontroller kein RS232 benötigt.

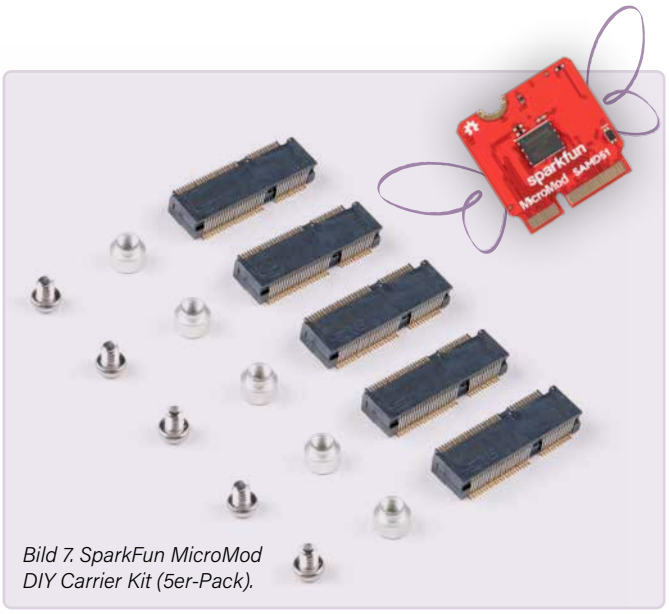


Bild 7. SparkFun MicroMod DIY Carrier Kit (5er-Pack).

Weitere Informationen finden Sie im Tutorial *Designing with MicroMod* [21]. Es führt Sie durch die Spezifikationen der MicroMod-Prozessor- und der Trägerplatinen sowie durch die Grundlagen der Einbindung des MicroMod-Formfaktors in Ihr eigenes Platinendesign!

Ressourcen und weiterführende Informationen

Zu den wertvollen MicroMod-Dokumenten gehören:

- SparkFun MicroMod Interface v1.0 - Anschlussbelegung [8]
- SparkFun MicroMod Interface v1.0 - Anschlussbeschreibungen [9]
- SparkFun Eagle Libraries mit Beispiel-Footprints für den M.2-Verbinder und die SMD-Abstandsmutter [28]
- Datenblatt M.2 MicroMod Connector [29]
- Datenblatt M2.5 Reflowable Standoff [30]
- MicroMod Info-Seite [31]
- MicroMod Foren [32]

Nachdem Sie nun mit den Grundlagen des MicroMods vertraut sind, finden Sie in den Textkästen verwandte Tutorials zum MicroMod.

200684-03

Elektor-Online-Artikel:
www.elektormagazine.de/esfe-en-micromod

Zukunftsaussichten: SparkFun MicroMod Prozessor- und Trägerplatinen

MicroMod

Hier finden Sie eine Liste der Sparkfun-MicroMod-Prozessor- und Trägerplatinen, die zum Zeitpunkt der Drucklegung entwickelt und verfügbar waren. Die aktuelle Übersicht finden Sie unter **www.sparkfun.com/micromod.**

Trägerplatinen

ATP: Greifen Sie auf alle Anschlüsse des Prozessorboards zu.

Input und Display: Eine großartige Möglichkeit, Daten und Eingaben sichtbar zu machen.

Datenerfassung: Eine anpassbare, stromsparende Plattform zur Datenprotokollierung.

Neu #1: Wird noch bekanntgegeben

Neu #2: Wird noch bekanntgegeben

Prozessor-Boards

Artemis: Wählen Sie den ultra-effizienten ARM-Cortex-M4F-Controller mit BLE 5.0 mit bis zu 96 MHz und einem Stromverbrauch von nur 6 µA/MHz (weniger als 5 mW).

ESP32: Stecken Sie das leistungsstarke Prozessorboard von Espressif mit dem Dual-Core-Tensilica- LX6 in den M.2-Verbinder, mit integriertem Bluetooth und WLAN ins Trägerboard.

SAMD51: Wählen Sie den SAMD51 für eine kostengünstige, leistungsstarke und einfach zu bedienende Entwicklungsplattform mit dem Komfort eines UF2-Bootloaders.

Neu #1: Wird noch bekanntgegeben

Neu #2: Wird noch bekanntgegeben

WEBLINKS

[1] Originaler M.2-Standard: https://de.wikipedia.org/wiki/M.2
[2] Das Ökosystem MicroMod im Überblick: https://www.sparkfun.com/micromod
[3] Serial Peripheral Interface (SPI): http://www.elektormagazine.de/esfe-en-micromod1
[4] Pulse Width Modulation: https://learn.sparkfun.com/tutorials/pulse-width-modulation
[5] Bausteine mit 3,3 V- oder 5-V-Logikpegel: https://learn.sparkfun.com/tutorials/logic-levels
[6] Eine Einführung in I2C: https://learn.sparkfun.com/tutorials/i2c
[7] M.2-Verbinder: https://de.wikipedia.org/wiki/M.2
[8] SparkFun MicroMod Interface v1.0 — Anschlussbelegung: https://bit.ly/3p96pJe
[9] SparkFun MicroMod Interface v1.0 — Anschlussbeschreibung: https://bit.ly/3asRCF1
[10] M.2-Kantensteckverbinder TE 2199230-4: https://de.farnell.com/te-m2-ngff-steckverbinder
[11] MicroMod DIY Carrier Kit: https://www.sparkfun.com/products/16549
[12] MicroMod-Anschlussbelegung: https://bit.ly/3rhRvlS
[13] MicroMod-Anschlussbeschreibung: https://bit.ly/34wu4eW
[14] Mini-Schraubendreher von SparkFun: https://www.sparkfun.com/products/9146
[15] Schraubendreher-Taschenset: https://www.sparkfun.com/products/12891
[16] Installation von CH340-Treibern: https://bit.ly/3p4V50C
[17] Eigener Entwurf als Referenzdesign: https://www.sparkfun.com/static/contact
[18] MicroMod ESP32 Processor Board : https://www.elektormagazine.com/esfe-en-micromod2
[19] MicroMod SAMD51 Processor Board : https://www.elektormagazine.com/esfe-en-micromod3
[20] MicroMod Artemis Processor Board: https://github.com/sparkfun/MicroMod_Artemis_Processor
[21] Entwerfen mit MicroMod: https://learn.sparkfun.com/tutorials/designing-with-micromod
[22] MicroMod All The Pins (ATP) Carrier Board: https://github.com/sparkfun/MicroMod_ATP_Carrier_Board
[23] MicroMod Data Logging Carrier Board: https://github.com/sparkfun/MicroMod_Data_Logging_Carrier
[24] MicroMod Machine Learning Carrier Board: https://github.com/sparkfun/MicroMod_Machine_Learning_Carrier
[25] MicroMod Input and Display Carrier Board : https://github.com/sparkfun/MicroMod_Input_and_Display_Carrier
[26] MicroMod DIY Carrier Kit: https://www.sparkfun.com/products/16549
[27] TE 2199230-4: https://de.farnell.com/te-m2-ngff-steckverbinder
[28] Eagle-Bibliotheken, Footprints für M.2 und SMD-Abstandsmutter: https://github.com/sparkfun/SparkFun-Eagle-Libraries
[29] Datenblatt M.2 MicroMod Connector: https://bit.ly/3nCZM1B
[30] Reflowgeeignete M2,5-Abstandsmutter: https://www.elektormagazine.de/esfe-en-micromod4
[31] MicroMod Infoseite: https://www.sparkfun.com/micromod
[32] MicroMod Foren: https://forum.sparkfun.com/viewforum.php?f=180

Mein getunter SparkFun-JetBot

Wie ich meinen von einem Jetson Nano von NVIDIA gesteuerten JetBot aufgebohrt habe

Von Derek Runberg (USA)

Bei Elektronik-Bausätzen geht es darum, Komplikationen zu vermeiden und und Dinge zu vereinfachen. In Bezug auf die Robotik bedeutet das: ein vorgefertigtes Chassis, Beispielcode und sogar eine Anleitung, die etwas zum Laufen bringt, so dass sich schnell ein erster und befriedigender (Lern-) Erfolg einstellt.

Das *JetBot AI Kit* von SparkFun ist ein Beispiel für diese Art von Bausatz. Unser JetBot basiert auf dem Open-Source-Projekt von NVIDIA und entstand auch in Zusammenarbeit mit NVIDIA, um Kunden ohne 3D-Drucker die Möglichkeit zu geben, schnell einen entsprechenden Roboter zu bauen. Der Bausatz enthält alle Teile einschließlich des Chassis und aller notwendigen Mechanik. Mit im Kit sind auch die erforderlichen Komponenten, um dem Jetson Nano die Grundlagen der Computer Vision (CV) und des Maschinellen Lernens (ML) auf dem NVIDIA-Board zu implementieren. Mit dem SparkFun JetBot AI-Kit erübrigt sich das Warten auf einen 3D-Druck der Chassisteile und die Plackerei des früheren frühen Prototypings.

Eine Frage, die mir oft gestellt wird, ist die nach der Skalierbarkeit des JetBot: „Könnte jemand mit dem Kit beginnen und es zu einem echten Robotersystem ausbauen“? Meine Antwort lautet „ja“! Der Jetbot ist als „Sandkastenübung“ für relativ kleine und minimale Sensoren ausgelegt, abgesehen von der Kamera. Aber er ist leicht anpassbar und skalierbar auf verschiedene Chassis, Motoren und

Kameras. Damit meine Antwort nicht bloße Hypothese bleiben sollte, habe ich die Ärmel hochgekrempelt, um herauszufinden, welche Teile für die Aufrüstung meines JetBot nützlich sein könnten. Dieser Artikel dokumentiert diesen Prozess und zeigt die Fallstricke bei der Verwendung des JetBot-Bausatzes als Grundlage für etwas, das ein wenig mehr … fortgeschrittenen Spaß macht.

Planung

Nachdem ich meinen JetBot zusammengesetzt hatte, war der nächste Schritt, ihn mit mir verfügbaren Komponenten zu modifizieren. Wenn ich zusammen mit dem Jetson Nano irgendwelche Boards aus unserem Qwiic-Ökosystem verwenden wollte, mussten diese in Python unterstützt werden, zumindest für eine einfache Integration (im Moment möchte ich kein rohes I²C in Python verwenden).

Inhdem ich das Github-Repository von SparkFun für die *Qwiic_Py*-Bibliothek aufrief, konnte ich schnell alle verfügbaren Komponenten herausfiltern. Eine kurze Aufsummierung der verschiedenen Treiberverzeichnisse ergab 20 verschiedene Boards, die ich integrieren konnte. Von diesen 20 sind bereits zwei im JetBot-Kit enthalten, nämlich der *Qwiic Motor Driver* und das *Qwiic OLED Display*. Als ich mir dann die Liste genauer ansah, fragte ich mich, welche Boards denn *sinnvoll* wären, um meinen Roboter auf eine höhere Leistungsstufe zu heben. Ich entschied für drei verschiedene Boards, die ein Maximum an Potenzial freisetzen würden (**Bild 1**).

1. GPS-RTK-SMA ZED-F9P Breakout

Für mich braucht ein High-End-Roboter GPS, vor allem wenn man Robotik nicht nur im Labor, sondern im Freien betreiben möchte. Das GPS-Modul ZED-F9P von u-blox ist das neuste und beste, was der Markt hergibt, mit allem Drum und Dran, einschließlich der Fähigkeit, Real Time Kinematics (RTK) zu verwenden, um die Position millimetergenau zu bestimmen (was natürlich eine Basisstation voraussetzt). Ich könnte mir vorstellen, dass das GPS sowohl für die Navigation als auch für die Fernvermessung eingesetzt wird. Mit der Kombination von Computer Vision und GPS wäre es sogar möglich, verschiedene Objekte wie Felsen, Pflanzen oder sogar Menschen zu kartieren.

2. SparkFun Auto Phat

Der SparkFun Auto Phat ist ein Raspberry-Pi-Phat (partial HAT), der auch mit dem Jetson Nano kompatibel ist. Er besitzt eigentlich den gleichen Motortreiber-Chip wie der Qwiic Motor Driver, der Teil des originalen Bausatzes ist, aber er verleiht diesem mehr Funktionalität als ein ganzes Bündel von Qwiic-Boards. Der Auto Phat enthält den Motor-Controller, Eingänge für Motor-Encoder, eine IMU und Ausgänge für Servo-Steuerungen. Wenn ich den ursprünglichen Motortreiber durch diesen Phat ersetze, kann ich nicht nur die Motoren genauer steuern, sondern auch Servos verwenden und erhalte eine IMU als Zugabe.

3. VL53L1X Time of Flight Distance Sensor

Im JetBot-Bausatz ist ein Sensor enthalten, eine Kamera. Computer Vision ist sehr leistungsfähig und es ist erstaunlich, was der JetBot nur mit Computer Vision und etwas ML machen kann. Wenn man jedoch ein paar weitere Sensoren hinzufügt, kann der Roboter Objekte identifizieren und die Entfernung zu diesen Objekten bestimmen. Oder besser noch, er kann interessante Objekte außerhalb des Sichtfeldes der Kamera erkennen.

Für die Entfernungsmessung gibt es eine Menge von Optionen. Ich habe mich letztendlich für einen Time-of-Flight-Sensor (ToF) entschieden, der kostengünstig und stromsparend ist sowie eine gute Reichweite für einen Roboter von der Größenordnung des JetBot hat. Der VL53L1X-Entfernungssensor wurde wegen seiner Reichweite von vier Metern und dem relativ engen Sichtfeld ausgewählt. Meine Hoffnung war, dass ich ihn auf das Sichtzentrum des Roboters ausrichten und eine ungefähre Entfernung zu einem Objekt im Sichtfeld der Kamera erhalten könnte – was leichter gesagt als getan ist, wenn man es mit einer Fischaugen-Optik zu tun hat.

Echtzeit-Kinematik (RTK) und Dead Reckoning

RTK-fähige GPS-Empfänger nehmen die normalen Signale des globalen Navigationssatellitensystems (GNSS) zusammen mit einem RTCM-Korrekturstrom auf, um in Echtzeit eine Positionsgenauigkeit von 1 cm zu erreichen. Die Frequenz der Positionsausgabe variiert von Empfänger zu Empfänger, liegt aber bei den meisten bei wenigstens einmal pro Sekunde. Einige Empfänger können diese hochpräzise Position bis zu 20 Mal pro Sekunde ausgeben! Bei der Navigation in einer dicht bebauten Stadt, in einem kurzen Tunnel oder einem Parkhaus kann es zu einer schlechten Signalqualität oder einem vollständigen Signalverlust kommen. Diese Probleme werden mit der Koppelnavigation überwunden, bei der die aktuelle Position durch die Kombination von zuvor ermittelten Positionsdaten, Geschwindigkeit und Fahrtrichtung abgeschätzt wird. 3D-Inertialmesseinheiten (inertial measurement units, IMU) und Fahrzeugdaten wie Raddrehzahlen und Kilometerzähler können verwendet werden, um die aktuelle Position eines Fahrzeugs kontinuierlich zu berechnen, wenn GNSS-Daten kurzzeitig ausfallen.

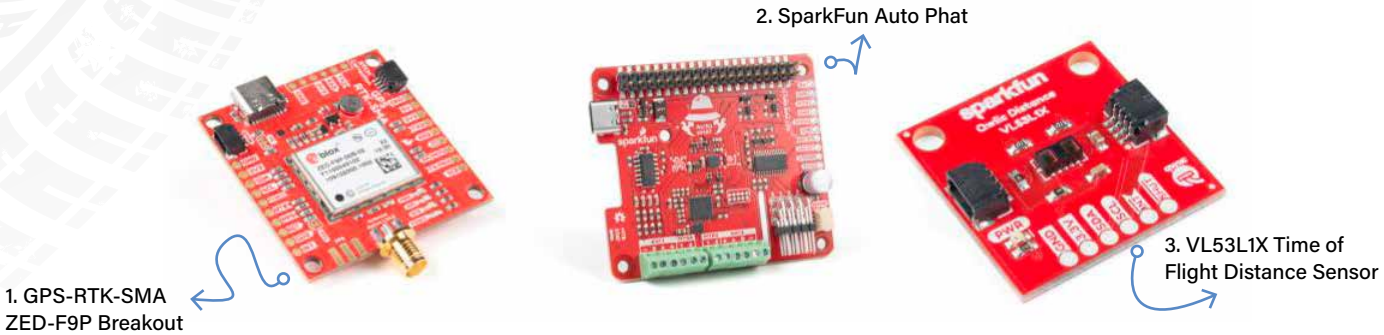


Bild 1. Die drei SparkFun-Boards, die ich für die Erweiterung des JetBot ausgewählt habe.

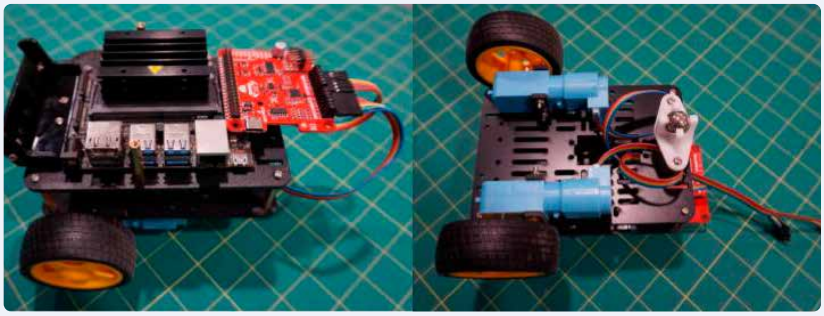


Bild 2. Bei dem modifizierten JetBot ersetzt der Auto-Phat den einfachen Qwiic-Phat und den Qwiic-Motortreiber.

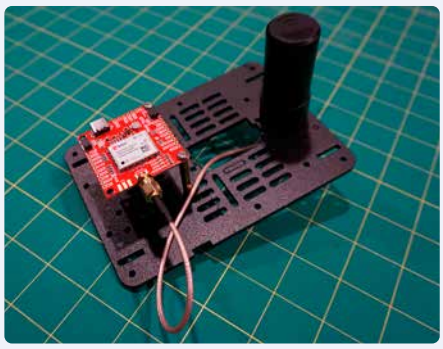


Bild 3. Die GPS-Platine wurde auf ein zusätzliches Chassisteil montiert, damit es über dem Jetson Nano und der Kamerahalterung schwebt.

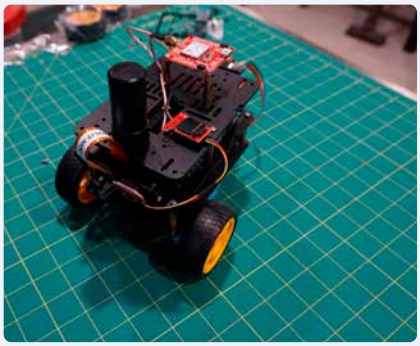


Bild 4. Der modifizierte JetBot (von hinten gesehen) mit der OLED-Platine und der zusätzlichen Montageplatte zur Befestigung der GPS-Platine.

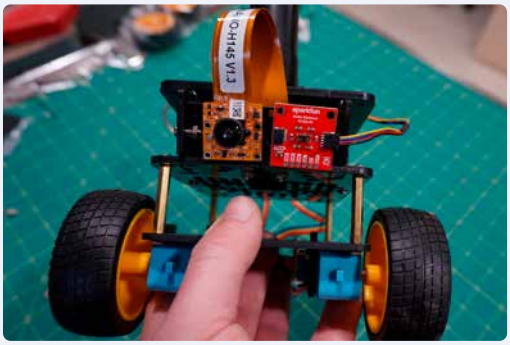


Bild 5. Vorderansicht mit dem Qwiic-kompatiblen ToF-Abstandssensor und dem Kameramodul auf der Halterung.

Bild 6. So sieht der „aufgebohrte" JetBot nach dem Zusammenbau aus.

Hardware-Integration

Ich begann mit der Integration des Moduls, das sich am meisten auf die bestehenden Systeme und den Aufbau meines Standard-JetBot auswirkte - dem Auto-Phat. Er ersetzt den einfachen Qwiic-Phat und den Qwiic-Motortreiber, so dass diese aus dem JetBot entfernt werden konnten. Da der Auto-Phat außerdem über Eingänge für Motor-Encoder verfügt, habe ich die serienmäßigen Getriebemotoren gegen Motoren mit Encodern ausgetauscht (**Bild 2**).

Der Austausch verlief relativ einfach, am schwierigsten erwies es sich, einen effizienten Verlauf für die Motorkabel zum Phat zu finden. Die Anschlussleitungen sind glücklicherweise recht lang, so dass ich sie durch das Chassis bis zu den Platinenklemmen am Phat führen konnte. Den Anschluss habe ich gemäß der Anleitung für den Phat vorgenommen.

Nachdem der Phat ausgetauscht war, musste ich nur noch das Qwiic-Kabel für das Qwiic-OLED-Display und das USB-Kabel des Akkupacks in den USB-Anschluss des Phat einstecken, und alles war schon perfekt zusammengebaut.

Als Nächstes war die GPS-Einheit an der Reihe. Das Board selbst ist etwas größer als ein Standard-Qwiic-Board mit Abmessungen von etwa 2,5 × 2,5 cm², so dass eine Montage neben dem Auto-Phat nicht in Frage kam. Und ich musste noch eine Stelle auf dem bereits überfüllten JetBot finden, auf der ich die Antenne montieren könnte. Ich habe deshalb ein Chassisteil aus meinem Ersatzteillager genommen und es über dem Jetson Nano und der Kamerahalterung (**Bild 3**) montiert. So hatte ich mehr Platz, um später sowohl das GPS als auch den Abstandssensor zu montieren (**Bild 4**). Das GPS ließ sich gut mit Abstandsbolzen über den Schlitzen montieren; zusätzlich bohrte ich ein Loch in der für die SMA-Verlängerung passenden Größe in das Gehäuse.

Das GPS-Board arbeitet am besten über einen UART-Port mit Einplatinencomputern zusammen. Obwohl ich einen der UARTs an der Stiftleiste hätte anschließen können, entschied ich mich, den USB-Port als UART-Port zu verwenden, da er bereits entsprechend konfiguriert ist. Dies ersparte mir zusätzliche Fummel- und Löterei.

Die letzte Ergänzung des JetBot, die ich vornehmen wollte, war der ToF-Abstandssensor. Glücklicherweise weist dieser Sensor unseren „zölligen" Qwiic-Formfaktor auf. Ich habe ihn direkt neben der Kamera an der einfachen Halterung befestigt, und zwar mit einem Quadrat aus Klettband (**Bild 5**). Ich verwende Klettband gerne, da es mir eine gewisse Flexibilität bei der Montage des Sensors ermöglicht, ohne eine Menge Löcher bohren zu müssen. Der letzte Schritt nach der Montage des Sensors bestand darin, ein Qwiic-Kabel in der passenden Länge zu finden und es an das nächstgelegene Qwiic-Board anzuschließen. Bei meinem Aufbau war dies die OLED-Display-Platine auf dem Zusatzchassis. Das Ergebnis ist in **Bild 6** zu sehen.

Software-Integration

Es gibt mehrere Möglichkeiten, die angepasste Software für die neuen Komponenten in den JetBot zu integrieren. Die einfachste ist, das spezielle Softwarepaket für den SparkFun JetBot zu verwenden, mit dem Qwiic-Motortreiber sowie die von NVIDIA für den

JetBot verwendeten Beispiele, die Jupyter-Notebooks nutzen, um auf das Dateisystem zuzugreifen und Python-Skripte ferngesteuert auszuführen.
Darauf können Sie aufbauen, indem Sie das *Qwiic_py*-Python-Package auf Ihrem JetBot einfach über die Terminal-Kommandozeile und `pip` installieren und den Beispielcode für jedes Board ausführen. Öffnen Sie ein Terminal-Fenster auf Ihrem NVIDIA Jetson Nano, entweder vom Desktop oder von Jupyter-Notebooks aus, und geben Sie den folgenden Befehl ein:

```
sudo pip install sparkfun-qwicc
```

Mit diesem Befehl werden die Treiber für alle derzeit unterstützten Qwiic-Karten des Ökosystems heruntergeladen und installiert. Die letzte Installation, die mit `pip` durchgeführt werden muss, ist die des PySerial-Pakets, der die gewünschte USB-Kommunikation zwischen dem Jetson Nano und dem F9P-GPS-Board ermöglicht. Um das PySerial-Paket zu installieren, geben Sie folgenden Befehl ein:

```
sudo pip install py_serial
```

Wenn beide Pakete installiert sind, können wir nun Beispielcode für jeden der zusätzlichen Sensoren ausführen. Die Beispiel-Python-Codes für alle der von uns unterstützten Qwiic-Karten im Qwiic_py-Github-Repository finden Sie unter [1]. Laden Sie den Beispielcode entweder herunter oder copy-pasten Sie ihn in eine Python-Datei und führen Sie ihn von der Kommandozeile aus, indem Sie den folgenden Befehl eingeben:

```
python3 example.py
```

Wenn Sie ein Skript ausführen, das mit dem GPS-Modul zu tun hat, das ja über USB mit dem Jetson Nano verbunden ist, müssen Sie `sudo` mit dem Befehl verwenden, da Sie die Berechtigungen zum Lesen und Schreiben auf der UART-Verbindung über USB benötigen:

```
sudo python3 exampleGPS.py
```

Hinweis: Wenn Sie von Jupyter Notebooks aus arbeiten und das GPS- oder ein anderes Skript verwenden möchten, das Zugriff auf eine UART-Verbindung benötigt, müssen Sie die Berechtigungen für Ihren UART-Port ändern. Das können Sie mit dem folgenden Befehl tun:

```
sudo chmod 660 /dev/ttyAMC0
```

Wenn Sie neugierig auf ein paar Python-Beispielprogramme sind, die ich für den modifizierten JetBot in Jupyter Notebook erstellt habe, können Sie diese unter [2] auf github finden und auf Ihren eigenen JetBot herunterladen. Jupyter Notebook erklärt die Codeschnipsel und ermöglicht es Ihnen, den Code von dort aus auszuführen, ohne dass Sie eine Kommandozeile benötigen. Wenn Jupyter Notebooks neu für Sie ist, schauen Sie sich das kurzweilige How-to-Tutorial unter [3] an.

SparkFun und NVIDIA

Maschinelles Lernen (ML) ist für uns alle neu! Um es für unsere Kunden sinnvoll zu nutzen und gleichzeitig innovativ zu sein, brauchen wir eine Community. Seit einigen Jahren arbeitet SparkFun mit Nvidia zusammen, um unseren Kunden maschinelles Lernen in Form von Bausätzen mit dem Jetson-Nano-Board und kostenlosen Online-Lerninhalten näher zu bringen. NVIDIA treibt die Grenzen des maschinellen Lernens ständig voran, SparkFun hilft, diese Technologie zu entmystifizieren und zu vereinfachen.

Abschließend

Ich habe nur an der Oberfläche der Möglichkeiten gekratzt, wie Sie Ihren JetBot aufwerten könnten. Es gibt so viele verschiedene Sensoren und Aktoren, die einfach zu integrieren und mit Einplatinencomputern zu verwenden sind. Mit den ML- und KI-Fähigkeiten des Jetson Nano von NVIDIA können Sie spielend leicht High-End-Leistung selbst in die einfachsten Roboterplattformen bringen.
Ich habe den JetBot mit ein paar Zubehörteilen selbst aufgepeppt, aber das ist nicht das Ende der Fahnenstange! Wenn Sie den Jetson Nano als Steuerungssystem für eine beliebige Roboterplattform verwenden, steht Ihnen die Welt offen, und Sie können ihn skalieren oder in andere Plattformen einbauen.
Tatsächlich habe ich letztes Jahr die JetBot-Plattform auf die Sphero RVR-Roboterplattform portiert, mit ein paar Teilen aus dem SparkFun-Katalog, etwas Schneiden, Schleifen und Bohren in Hard- und Software, um den RVR intelligenter zu machen und die ML-Fähigkeiten des Jetson Nano zu nutzen. Sie können meine Schritt-für-Schritt-Anleitung unter [4] finden.
Der Jetson Nano von NVIDIA und die JetBot-Plattform gehören zu den spaßigsten Robotik-Bausätzen und -Produkten, die ich seit langem verwendet habe. Sie verbinden Spitzentechnologie, Zugänglichkeit und Flexibilität. Wenn Sie hacken wollen, sind Sie hier richtig. Ich bin ständig auf der Suche nach Möglichkeiten, den Roboter zu verbessern und gleichzeitig mein Wissen und Verständnis von ML und KI zu erweitern.

200689-03

WEBLINKS

[1] Beispiel-Python-Code für Qwiic-Boards: https://bit.ly/2KNPf5k
[2] Beispiel-Python-Programme in Jupyter Notebook: https://bit.ly/3a9RbO7
[3] Anleitung für Jupyter Notebook: https://bit.ly/36eanJA
[4] Tutorial für Sphero RVR Mash-up: https://bit.ly/3c7Non3

Passende Produkte

- SparkFun JetBot AI Kit v2.1 Powered by Jetson Nano
  www.elektormagazine.de/esfe-en-jetbot1
- SparkFun GPS-RTK-SMA Breakout - ZED-F9P (Qwiic)
  www.elektormagazine.de/esfe-en-jetbot2
- SparkFun Auto pHAT für Raspberry Pi
  www.elektormagazine.de/esfe-en-jetbot3
- SparkFun Distance Sensor Breakout - 4 Meter, VL53L1X (Qwiic)
  www.elektormagazine.de/esfe-en-jetbot4

Programmierung eines FPGAs

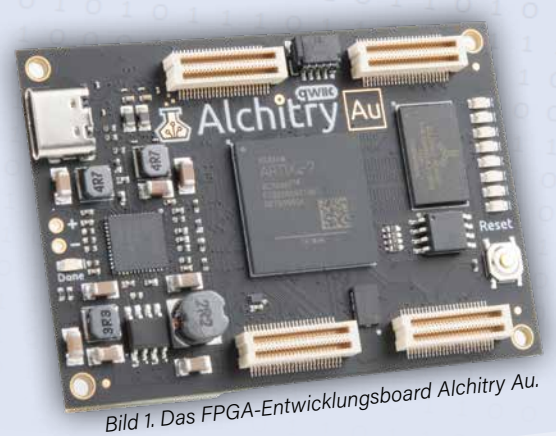


Bild 1. Das FPGA-Entwicklungsboard Alchitry Au.

Passende Produkte

Sie suchen die in diesem Artikel genannten Produkte? Elektor und SparkFun haben sie!

PGA-Entwicklungsboard Alchitry Au

www.elektormagazine.de/esfe-en-fpga1

Von Justin Rajewski (Alchitry)

Dieses Tutorial deckt die Grundlagen ab, die man zum Erstellen eines Designs für ein FPGA benötigt und erläutert die fundamentalen Bausteine, die Sie dabei verwenden können. Der erste Kontakt mit einem FPGA mag abschrecken, aber eine gute Kombination aus zugänglicher Software, vielseitiger Hardware und einigen interessanten Beispielen wird Ihnen den Einstieg sicher erleichtern.

Lassen Sie uns kurz etwas klarstellen, bevor wir in die Materie eintauchen. FPGAs werden, wie der Titel suggeriert, nicht programmiert [1]. Wenn Sie etwas Text schreiben, den Text in eine Binärdatei verwandeln und die Binärdatei auf den FPGA laden, fühlt sich das wie Programmieren an, aber in Wahrheit schreiben Sie kein Programm., sondern erstellen eine Schaltung. Sie verwenden dazu keine Programmiersprache, sondern Hardware-Beschreibungssprachen (HDLs).[2]
Große, komplexe Designs lassen sich nicht mit einem Schaltplan entwerfen. Stattdessen beschreiben wir das gewünschte Verhalten der gewünschten Schaltung und die Tools finden heraus, wie sie das Verhalten tatsächlich implementieren. Es ist wichtig, bei der Erstellung eines Designs für ein FPGA im Hinterkopf zu behalten, dass Sie Hardware beschreiben und dass alles, was Sie schreiben, letztendlich als physikalische Schaltung auftauchen wird. Es ist möglich, Schaltungen zu beschreiben, die unmöglich zu implementieren sind, oder etwas zu beschreiben, das einfach erscheint, aber eine riesige Menge an Ressourcen für die Implementierung benötigt. Aus diesem Grund ist es wichtig, eine gute Vorstellung davon zu haben, wie die Schaltung, die Sie zu beschreiben versuchen, implementiert werden könnte. Um diesem Tutorial zu folgen, benötigen Sie das FPGA-Entwicklungsboard Alchitry A F (**Bild 1**) und ein reversibles USB-A-zu-C-Kabel.

Aufbau eines Designs

HDLs basieren üblicherweise auf Modulen. Ein Modul ist ein Schaltungsblock, der eine gewisse Anzahl von Eingängen und Ausgängen besitzt und eine Logik enthält, um diese miteinander zu verbinden. Ein Modul kann Untermodule enthalten oder es kann eigenständig sein, ähnlich einem Programm, das in Funktionen gegliedert ist.
Es wäre zwar möglich, ein ganzes Design in ein einziges Modul zu packen, aber es ist besser, kompaktere Module zu verwenden, um Ihr Projekt auszuführen. So vereinfachen Sie die Komplexität einer einzelnen Funktion, an der Sie zu einem bestimmten Zeitpunkt arbeiten müssen. Einige Module können so gestaltet werden, dass sie allgemeine Aufgaben erfüllen und immer wieder verwendet werden können.

Wenn Sie mit einem Entwurf beginnen, ist es oft hilfreich, ein Blockdiagramm zu zeichnen, das die verschiedenen Module und ihre Verbindungen zueinander zeigt. Dies hilft, den Umfang Ihres Entwurfs zu definieren und ihn logisch aufzuschlüsseln.

Lucid

Für den Rest des Tutorials werden wir Lucid [3] verwenden., eine HDL, die speziell für FPGAs entwickelt wurde und viele der Fallstricke beseitigt, die bei anderen HDLs wie Verilog und VHDL auftreten (und glauben Sie mir, es gibt viele davon).
Lucid ist fantastisch, um in den Umgang mit FPGAs einzusteigen. Ich werde oft von Leuten angesprochen, die sich Sorgen machen, dass sie mit Lucid nicht weiterkommen können oder aus anderen Gründen einfach in Verilog oder VHDL einsteigen wollen. Wenn Sie gerade erst anfangen, sollten Sie aber mit Lucid beginnen. Es wird Ihnen die richtigen Grundlagen des Hardware-Designs vermitteln, bevor Sie versuchen, sich durch die schwerfälligen anderen HDLs zu kämpfen. Wenn Sie später auf etwas anderes umsteigen wollen, ist das dann nicht sehr schwer. Lucid basiert nämlich auf Verilog und Alchitry Labs konvertiert Lucid nach Verilog, wenn Sie Ihre superschicken Module woanders verwenden wollen.

Anatomie eines Moduls

Wenn Sie ein beliebiges Design erstellen, gehen Sie vom Top-Level-Modul aus. Dies ist das Modul, dessen Eingänge und Ausgänge die tatsächlichen Eingänge und Ausgänge an den Pins des FPGAs sind. Für jedes Alchitry-Projekt sind dies entweder *cu_top.luc* oder a*u_top.luc*, je nach verwendetem Board (Cu oder Au). Die Top-Level-Module für beide Boards sehen im Wesentlichen identisch aus (**Listing 1**).
Der erste Abschnitt des Moduls betrifft die Deklaration der Ein- und Ausgänge des Moduls. In diesem Fall sind, da es sich um das Top-Level-Modul handelt, dies die Signale auf dem Board selbst (**Listing 2**). Sie haben vielleicht bemerkt, dass hinter dem Eingang *led* eine Zahl in eckigen Klammern steht. Das macht den Eingang nicht zu einem, sondern zu acht einzelnen Eingängen, die als Array zusammengefasst sind. Wir werden später mehr über die Array-Syntax erfahren.
Ein Modul kann auch eine Liste von Parametern besitzen, die zum Anpassen des Moduls verwendet werden kann. Diese Liste ist hier weggelassen und wäre bei einem Top-Level-Modul auch sinnlos, da die Parameter von dem übergeordneten Modul übergeben werden, das es instanziiert. Der Begriff *Instanziierung* wird verwendet, wenn ein Modul oder eine andere Ressource zu Ihrem Entwurf hinzugefügt wird. Er bedeutet, dass eine Instanz dieses Moduls oder einer anderen Ressource erstellt wird.
Wenn Sie Code schreiben, werden jedes Mal, wenn Sie eine Funktion aufrufen, die darin enthaltenen Anweisungen immer wieder gleich verwendet, egal wie oft die Funktion aufgerufen wird. Wenn Sie jedoch ein Modul instanziieren, wird jedes Mal der gesamte Schaltkreis, den das Modul repräsentiert, dupliziert. Wenn Sie dieselben Ressourcen für mehrere Aufgaben wiederverwenden wollen, ist es Ihre Aufgabe als Entwickler, zu bestimmen, wie Sie damit umgehen wollen. Üblicherweise instanziiert man alles, was das Modul braucht, direkt nach der Port-Deklaration.
Die erste Zeile ist die Deklaration eines Signals mit dem Schlüsselwort *sig*:

```
  sig rst;                    // reset signal
```

Signale sind kein Speicher, die irgendwelche Werte speichern. Sie sollten sie sich als Drähte vorstellen. Ein Draht kann einen Wert haben,

Listing 1.

```
module au_top (
    input clk,      // 100MHz clock
    input rst_n,    // reset button (active low)
    output led [8], // 8 user controllable LEDs
    input usb_rx,   // USB->Serial input
    output usb_tx   // USB->Serial output
  ) {

  sig rst;          // reset signal

 .clk(clk) {
   // The reset conditioner is used to
   // synchronize the reset signal to
   // the FPGA clock. This ensures the
   // entire FPGA comes out of reset at
   // the same time.
   reset_conditioner reset_cond;
  }

  always {
   reset_cond.in = ~rst_n; // input raw inverted
                           // reset signal
   rst = reset_cond.out;   // conditioned reset
    led = 8h00;            // turn LEDs off
    usb_tx = usb_rx;       // echo the serial data
  }
}
```

Listing 2.

```
    input clk,          // 100MHz clock
    input rst_n,        // reset button (active low)
    output led [8],     // 8 user controllable LEDs
    input usb_rx,       // USB->Serial input
    output usb_tx       // USB->Serial output
```

aber er ist eigentlich nur eine Verbindung von einem Punkt zu einem anderen.
In diesem Entwurf verwenden wir das Signal *rst* als Platzhalter für den Ausgang des Moduls *reset_conditioner*. Die Instanziierung dieses Moduls erfolgt in den nächsten paar Zeilen:

```
 .clk(clk) {
// The reset conditioner is used to synchronize
// the reset signal to the FPGA clock. This
// ensures the entire FPGA comes out of reset
// at the same time.
    reset_conditioner reset_cond;
  }
```

Um etwas zu instanziieren, verwenden wir einfach den Namen der Ressource, gefolgt von dem Namen dieser speziellen Instanz. Die Zeile `reset_conditioner reset_cond;` erzeugt also eine Instanz des Moduls *reset_conditioner* mit dem Namen *reset_cond*. Der Block, in dem diese Instanziierung verpackt ist, wird als Verbindungsblock (connection block) bezeichnet. Er ermöglicht es, einen Eingang oder Parameter mit einem bestimmten Namen vieler Module mit demselben Signal zu verbinden.
In unserem Fall verbinden wir den Eingang *clk* mit dem Signal *clk* (das ein Eingang zu unserem Modul ist). Die Syntax lautet *.port(signal)*, wobei *port* der Name des Eingangs am zu instanziierenden Modul und *signal* das Signal ist, das damit verbunden werden soll.

Das Modul *reset_conditioner* besitzt einen Eingang namens *clk*, so dass dieser Eingang direkt mit dem Signal *clk* in unserem Modul verbunden ist. Sie könnten dies in der Instanziierungszeile auch direkt mit dem Modul verbinden:

```
reset_conditioner reset_cond(.clk(clk));
```

Das wurde hier aber nicht gemacht, weil der Eingang *clk* Teil fast jedes Moduls ist und es praktisch ist, einen solchen Block einzurichten, damit Sie einfach Ihre anderen Instanziierungen hinzufügen können, die mit *clk* verbunden werden müssen.

Fast jedes Modul dürfte einen *clk*-Eingang haben und oft auch einen *rst*-Eingang für einen Reset. Wofür Takt und Reset genau verwendet werden, wird später behandelt. Sie können mehrere Verbindungen zu demselben Verbindungsblock hinzufügen:

```
.clk(clk), .rst(rst) { … }
```

Sie können die Blöcke auch verschachteln, und da nicht jeder Block mit *clk* auch *rst* verwendet, werden Sie am Anfang eines Moduls normalerweise etwas in der folgenden Form sehen:

```
.clk(clk) {
    // clk only instantiations
    .rst(rst) {
    // rst and clk instantiations
    }
}
```

Über die always-Blöcke

Der nächste Abschnitt betrifft den *always*-Block, das Herzstück des Moduls. In *always*-Blöcken beschreiben Sie die gesamte Logik, die in Ihrem Modul abläuft. Er enthält die kombinatorische Logik des Moduls. Kombinatorische Logik ist jede digitale Schaltung, deren Ausgang eine Funktion ausschließlich der aktuellen Eingänge ist. Sie besitzt keinen internen Zustand oder Speicher. Ein gutes Beispiel hierfür ist eine Additionsschaltung. Der Ausgang wird ausschließlich durch die beiden aktuellen Eingangszahlen bestimmt. Es spielt keine Rolle, was die zuletzt eingegebenen Zahlen waren oder wie oft Sie die Zahlen geändert haben.

Innerhalb des *always*-Blocks schreiben wir Anweisungen, die aus vier Haupttypen zusammengesetzt sind:

- Zuweisungen (Assignments)
- if-Anweisungen
- case-Anweisungen
- for-Schleifen

Mein erstes FPGA-Projekt: Vertraut werden mit PWM

Wenn Sie zum ersten Mal ein Alchitry-Au- [5] oder Alchitry-Cu-Board [6] in Betrieb nehmen, erzeugt die Standard-FPGA-Konfiguration einen ausgefallenen Welleneffekt auf der LED-Reihe. In diesem Tutorial lernen wir, wie man so etwas erzeugen kann. Es ist ein großartiger Überblick über die Herangehensweise an ein Design und betrachtet verschiedene Dinge, die bei der Arbeit mit Hardware zu beachten sind.

Möchten Sie mehr erfahren?
Dann schauen Sie sich das Tutorial an!
https://learn.sparkfun.com/tutorials/first-fpga-project---getting-fancy-with-pwm

Zuweisungen (Assignments)

Zuweisungen sind die bei weitem häufigsten Anweisungen. Sie haben ein Signal auf der linken Seite, gefolgt von einem Gleichheitszeichen und einem Ausdruck.

```
signal = expression;
```

Die Leistungsfähigkeit einer Zuweisung liegt in ihrem Ausdruck (expression) begründet. Sie können eine Handvoll verschiedener Operatoren verwenden, um Bits zu manipulieren. Dazu gehören einige mathematische Operatoren wie „+", „-" und „*". Das Zeichen „/" für Divisionen darf nicht verwendet werden, wenn der Ausdruck dynamisch ist. Das liegt daran, dass eine solche Division zu kompliziert wäre. Natürlich ist eine Division möglich, aber sie erfordert etwas mehr Aufwand und Planung.

***if*-Anweisungen**

If-Anweisungen folgen der typischen Notation:

```
if (expr) { … } else { … }
```

Wenn der Ausdruck, der dem *if* folgt, wahr (ungleich null) ist, dann ist der erste „Satz" von Zeilen gültig. Wenn er falsch (null) ist, sind die Zeilen im (optionalen) *else*-Block gültig. Beachten Sie, dass ich von „gültig" und nicht von „ausgeführt" spreche. Es kann leicht passieren, dass man bei *if*-Anweisungen und *for*-Schleifen in die Falle tappt, wenn man in der Kategorie „Programmierung" denkt. *If*-Anweisungen werden in der Hardware am häufigsten mit einem Multiplexer realisiert. Sie wählen, basierend auf einem Ausdruck, einfach einen von zwei Eingängen aus.

Wenn Sie einem Signal in einem *always*-Block einen Wert zuweisen, muss diesem unabhängig von den Bedingungen ein Wert zugewiesen werden. Wenn Sie also in einer if-Anweisung einen Wert zuweisen, sollten Sie eine passende Zuweisung im else-Teil der Anweisung haben. Die einzige Ausnahme von dieser Regel ist der d-Eingang eines DFF- oder FSM-Typs. Dazu werden wir später kommen.

Eine weitere Möglichkeit, um sicherzustellen, dass Sie immer einen Wert zuweisen, besteht darin, Ihren *always*-Block mit einigen sinnvollen Standardwerten zu beginnen. Schauen Sie sich den folgenden Pseudocode an:

```
led = 0;
if (button_pressed)
    led = 1;
```

Wenn die Taste nicht gedrückt ist, hat *led* den Wert 0. Was passiert aber, wenn die Taste gedrückt wird? Wird *led* ein Wert von 0 zugewiesen und dann mit dem Wert 1 aktualisiert? Nein. Wenn die Taste gedrückt wird, hat *led* immer den Wert 1. In einem *always*-Block haben folgende Zuweisungen Vorrang vor führenden Zuweisungen. Sie können sich es so vorstellen, dass der Block immer und augenblicklich ausgewertet wird.

Stellen Sie sich nun vor, wir hätten diesen Standardwert nicht vor der if-Anweisung zugewiesen. Welchen Wert würde led haben, wenn die Taste nicht gedrückt ist? Es ist verlockend, anzunehmen, dass sie einfach ihren vorherigen Wert behalten würde, aber denken Sie daran, dass Signale keine Werte speichern können. Sie sind einfach wie Drähte, die zwei Dinge miteinander verbinden.

Mit dem Standardwert von 0 könnten wir dies in Hardware mit einem Multiplexer realisieren. In diesem trivialen Fall könnte man es vereinfachen, indem man einfach die Signale *led* und *button_pressed* miteinander verbindet, wie in **Bild 2** gezeigt.

Case-Anweisungen

Case-Anweisungen folgen dieser Syntax:

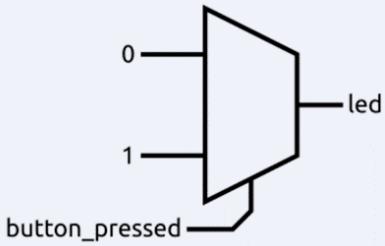


Bild 2. À la FPGA: Ein-/ Ausschalttaste für eine LED.

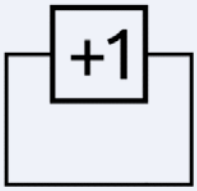


Bild 3. Addierer-Funktion.

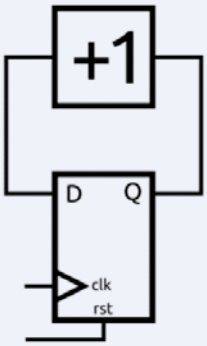


Bild 6. DFF und Zähler kombiniert.

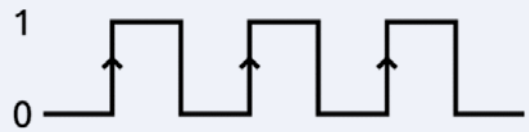


Bild 4. Taktsignal mit steigenden Flanken (Pfeile).

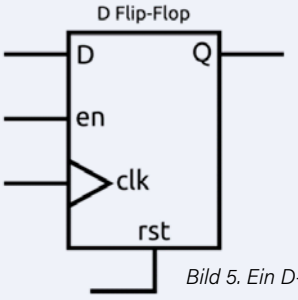


Bild 5. Ein D-Flip-Flop (DFF).

```
case(expr) {
    value: statement;
    value: statement;
    default: statement;
}
```

Eine *case*-Anweisung funktioniert genau wie eine *if*-Anweisung, kann aber auf einfache Weise viele Verzweigungen von einem einzigen Ausdruck ermöglichen. Der Werte-Teil der *case*-Anweisung muss eine Konstante sein. Der optionale *default*-Anweisung ist eine Art Notausgang.

Anders als bei Programmcode bieten *case*-Anweisungen keine Leistungsverbesserungen gegenüber vielen *if*-Anweisungen, sondern sorgen nur lediglich für Übersichtlichkeit im „Code".

for-Schleifen

For-Schleifen in Lucid haben die gleiche Syntax wie C- oder Java-for-Schleifen, weisen aber einige Einschränkungen auf:

```
for (init; eval; increment) { … }
```

For-Schleifen werden normalerweise mit einem *var*-Typ verwendet, der Werte speichert, die nicht direkt in der Schaltung auftauchen, aber in der Beschreibung verwendet werden. Die große Einschränkung bei *for*-Schleifen in Hardware ist, dass sie eine konstante Anzahl von Iterationen haben müssen, weil die Tools in der Lage sein müssen, die Schleife zu durchlaufen. Eine *for*-Schleife ist nichts anderes als das wiederholte Einfügen eines Codeabschnitts, außer dass sie viel einfacher zu lesen ist. Sie sollten *for-Schleifen* normalerweise vermeiden, es sei denn, Sie haben einen guten Grund, eine zu verwenden. Es ist schnell erreicht, dass mit *for*-Schleifen eine sehr große und langsame Schaltung zu erstellen.

Zahlen

In Lucid gibt es eine Handvoll Möglichkeiten, eine numerische Konstante zu definieren. Der einfachste Weg ist es, die Zahl einfach einzugeben, zum Beispiel 14. Wenn Sie eine einsame Zahl sehen, ist sie dezimal (Radix von 10) und die Anzahl der Bits, die zur Darstellung verwendet werden, ist das Minimum, das in einem vorzeichenlosen Format erforderlich ist.

Wenn Sie mehr Kontrolle über die Anzahl der verwendeten Bits haben möchten, können Sie der Zahl ein *xd* voranstellen, wobei *x* die Anzahl der zu verwendenden Bits ist. Zum Beispiel ist *8d14* der Dezimalwert 14, der mit 8 Bits dargestellt wird. Sie können das *d* gegen *h* tauschen, um hexadezimale (Radix von 16) Zahlen zu verwenden oder *b* für Binär (Radix von 2). Bei beiden Formaten können Sie die Anzahl der zu verwendenden Bits vor dem Buchstaben angeben. Wenn Sie die Anzahl der Bits weglassen, verwenden hexadezimale Zahlen standardmäßig 4 Bits pro Stelle, bei *h08* sind es 8 Bits, da hinter dem *h* zwei Ziffern stehen, obwohl der Wert 8 auch mit nur 4 Bits dargestellt werden könnte.

Wenn nicht explizit angegeben, ist bei Binärzahlen die Anzahl der Bits einfach die Anzahl der Ziffern. Also ist *b101001* 6 Bits. Wenn eine Dezimalzahl mit dem *d* geschrieben wird, aber die Anzahl der Bits weggelassen wird, verhält sie sich genauso, als ob das *d* ebenfalls weggelassen würde.

Arrays

Viele Signale, auf die Sie stoßen werden, sind Signale mit mehreren Bits, zum Beispiel der *led*-Eingang in unserem Top-Level-Modul. Die Bits in einem Array können mit der Syntax *signal[bit]* einzeln indiziert werden, wobei *bit* ein Ausdruck ist. Sie müssen sicherzustellen, dass der Wert immer innerhalb der Grenzen des Arrays liegt, wenn es sich um einen dynamischen Wert handelt.

Sie können auch auf Teilmengen der Bits zugreifen, indem Sie die Array-Syntax *[max:min]* verwenden. Hier wird der Bereich der Bits ausgewählt, der bei *min* beginnt und bis einschließlich *max* reicht. Dabei müssen beide Werte Konstanten sein.

Wenn Sie eine Teilmenge von Bits dynamisch auswählen möchten, können Sie die Syntax *[start+:width]* verwenden. Hier ist *start* das niedrigste auszuwählende Bit und *width* die Anzahl der auszuwählenden Bits (inklusive Startbit). Bei dieser Syntax muss nur die Breite konstant sein. Sie können auch die Variante *[start-:width]* verwenden, bei der der Start das höchste Bit in der Auswahl ist, nicht das niedrigste.

Arrays in Lucid können mehrdimensional sein. Wenn Sie sie deklarieren, fügen Sie einfach ein paar zusätzliche Dimensionen an:

```
sig my_array[dim1][dim2][dim3];
```

Alle Dimensionen eines Arrays müssen mit konstanten Werten deklariert werden. Sie können dann das Array mit den Selektoren wie zuvor indizieren. Beachten Sie jedoch, dass Sie die Sub-Array-Selektoren nur als letzten Selektor verwenden können.

Sequentielle Logik und DFFs

Jetzt wird es wirklich interessant. Kombinatorische Logik ist zwar sehr wichtig, aber ein System ohne jeden Zustand oder Speicher wäre doch ziemlich eingeschränkt.

Wie kann man also Speicher erzeugen? Im Grunde genommen brauchen Sie nur eine Art von Rückkopplungsschleife. Wenn Sie einen Zähler erzeugen wollen, addieren Sie einfach eine Eins zum Ergebnis der letzten Addition. Das Problem ist, wie steuern Sie diese Schleife? Auf den ersten Blick scheint dies kein Problem zu sein, aber bei näherer Betrachtung öffnet sich die Büchse der Pandora.

Schauen wir uns das Zählerbeispiel an. Wir könnten ihn mit einem Addierer erstellen, bei dem einer der Eingangswerte auf 1 festgelegt ist. Der Einfachheit halber packe ich das in einen einzigen Block (**Bild 3**). Wenn wir seinen Eingang mit dem Ausgang verbinden, erzeugen wir einen inkrementierenden Zähler, richtig?

Nun, das wirft einige Fragen auf. Bei welchem Wert beginnt dieser Zähler und wie schnell zählt er? Der Anfangswert würde davon abhängen, wie die Schaltung versorgt und wie sie aufgebaut ist. Er würde wahrscheinlich auch von der Temperatur und anderen Umgebungsfaktoren abhängen. Wollen Sie wirklich, dass sich Ihre Schaltung je nach Wetterlage unterschiedlich verhält?

Was noch schlimmer ist: Diese Schaltung würde nicht einmal funktionieren. Das liegt daran, dass eine Additionsschaltung, wie die meisten kombinatorischen Logiken mit Multi-Bit-Ausgängen, falsche Zwischenergebnisse produziert. Im Falle eines Addierers wird das niederstwertige Bit zuerst berechnet, wobei jedes folgende Bit das Ergebnis des vorherigen Bits in die Berechnungen einfließt. Da nichts das Ergebnis validiert, werden die falschen Zwischenwerte zurück in den Addierer gespeist, der weitere falsche Werte propagiert, bis er nur noch Datenmüll erzeugt. Wie können wir das beheben? Wir brauchen einfach eine Möglichkeit, das Timing der Rückkopplungsschleife zu kontrollieren. An dieser Stelle sind D-Flip-Flops (DFF) nützlich.

Bevor wir uns damit beschäftigen, was genau ein DFF ist, lassen Sie mich erklären, was ein Taktgeber ist. Ein Taktgeber ist einfach ein Signal, das mit einer bestimmten Frequenz immer wieder von 0 auf 1 schaltet, wie in **Bild 4** dargestellt. Der Takt auf den Alchitry-Boards schaltet mit 100 MHz, also 100 Millionen Mal pro Sekunde. Dieses regelmäßige Signal kann verwendet werden, um unseren Schaltungen ein Zeitgefühl zu verleihen. Der Übergang von 0 auf 1 wird als steigende Flanke bezeichnet und ist normalerweise der wichtige Teil des Signals. Diese Flanken sind im Bild mit Pfeilen markiert.

DFFs sind eine Art von Speicher. Sie haben einen Eingang D und einen Ausgang Q. Wenn sich ihr Takteingang von 0 auf 1 ändert, wird der Wert von D gespeichert und zur nächsten steigenden Flanke des Taktes an Q ausgegeben. Es spielt keine Rolle, ob sich D zwischen den steigenden Flanken des Taktes ändert, der Wert an Q bleibt derselbe. In **Bild 5** ist das DFF mit den optionalen Enable- und Reset-Signalen dargestellt. Mit dem Enable-Signal kann man den DFF daran hindern, bei einer steigenden Flanke einen neuen Wert einzulesen. Das Reset-Signal wird verwendet, um den Q-Wert auf einen vorgegebenen Wert (bei FPGAs 0 oder 1) zu zwingen.

Wir können den DFF in unserem Zähler verwenden, um die Schleife zu steuern (**Bild 6**). Mit dem Reset-Signal setzen wir den Anfangswert auf beispielsweise 0 fest. Wir wissen also, dass Q gleich 0 und damit D gleich 1 ist. Bei der steigenden Flanke des Taktgebers erhält Q den Wert von D; Q wird 1 und D wird 2. Bei jeder steigenden Flanke wird Q um 1 erhöht. Das ist genau das, was wir wollen! Die Frequenz des Taktgebers bestimmt, wie schnell unser Zähler inkrementiert. Dabei gibt es natürlich einige Einschränkungen. Der Takt muss langsam genug sein, damit der Wert an D Zeit hat, sich nach der Änderung von Q zu aktualisieren. Wir wollen ja nicht, dass unser DFF einen der ungültigen Zwischenwerte speichert, die der Addierer erzeugt. Die Zeitspanne, die benötigt wird, damit der Ausgangswert des Addierers gültig ist, wird als Laufzeitverzögerung (*propagation delay*) bezeichnet. Das ist die Zeitspanne, die von einer Änderung der Eingänge bis zum gültigen und stabilen Ausgang vergeht. Je mehr Logik Sie addieren, desto länger ist diese Verzögerung. Die Verzögerung ist auch eine Funktion der Technologie, die für die Herstellung der Schaltung verwendet wird. Die Tools kennen Modelle für jedes FPGA und wenn Sie ihnen Ihre Taktfrequenz mitteilen, versuchen sie, Ihr Design so zu layouten, dass die Timing-Anforderungen erfüllt werden. In diesem Beispiel haben wir einen Satz von DFFs, die durch einen Block mit kombinatorischer Logik geschleift werden. Es ist üblich, den Ausgang eines Satzes von DFFs durch einen Block mit kombinatorischer Logik in einen anderen Satz von DFFs zu leiten, wodurch eine Pipeline entsteht.

In jedem Design legt die längste Laufzeitverzögerung die maximale Taktfrequenz fest. Indem Sie Ihr Design in ungefähr gleichmäßig getaktete Blöcke mit kombinatorischer Logik aufteilen, können Sie die Taktfrequenz optimieren. Die Feinheiten des Timings können ziemlich kompliziert werden, aber meistens dürften Sie mit einem gemeinsamen Takt für das gesamte Design auskommen. Die Tools kümmern sich darum, solange Sie keine unmöglich langen Pfade vorgeben. Bei 100 MHz kann man zwischen den DFFs eigentlich eine ganze Menge anstellen. Es wird nur dann problematisch, wenn Sie zu viele Dinge miteinander zu verketten versuchen oder eine Menge Multiplikationen einbauen. Die Einhaltung des Timings kann auch schwierig werden, wenn man sich dem Ressourcenlimit im FPGA nähert. Wenn die Tools immer mehr in die gleiche FPGA-Größe packen müssen, haben sie weniger Möglichkeiten, die Dinge anzuordnen, was dazu führen kann, dass sie die Timing-Anforderungen nicht erfüllen können..

Listing 3.

```
module blinker (
   input clk,  // clock
   input rst,  // reset
   output out
 ) {

 always {
   out = 0;
 }
}
```

Blinken einer LED

Lassen Sie uns nun all dies in einem Demoprojekt zusammenfassen, das eine LED blinken lässt. Legen Sie in Alchitry Labs [4] ein neues Projekt an und wählen Sie im Dropdown-Menü *From Example* den Eintrag *Base Project* (**Bild 7**). Klicken Sie auf das Icon *New File* in der Symbolleiste links und erstellen Sie eine neue Lucid-Source-Datei

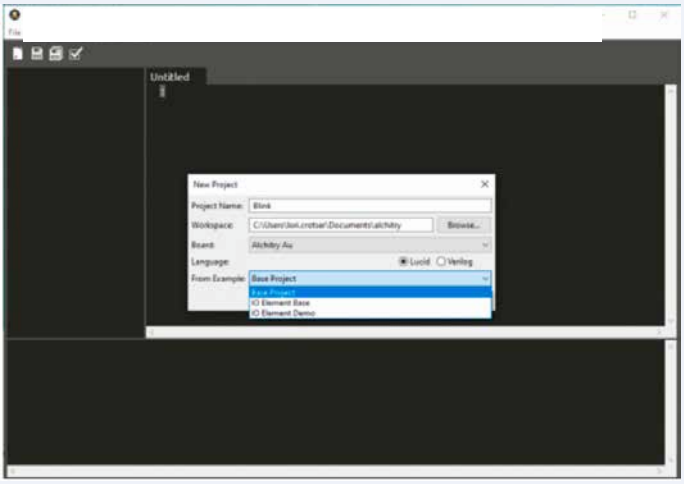


Bild 7. Auswahl des Basisprojekts in der Alchitry-Labs-Software.

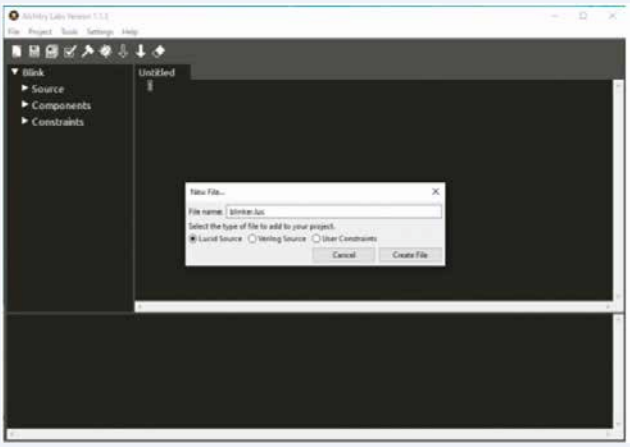


Bild 8. Erstellen einer neuen Lucid-Source-Datei.

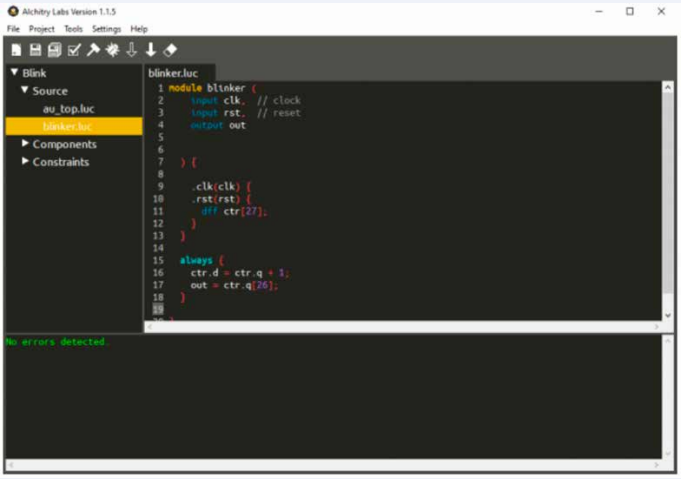


Bild 9. Das mit Listing 3 erstellte „Modul".

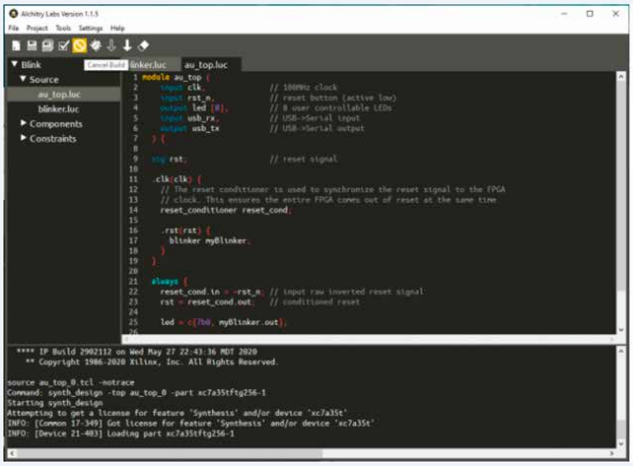


Bild 10. Hochladen des Projekts auf das Alchitry-Board.

namens *blinker.luc* (**Bild 8**). Dadurch wird ein Basismodul erstellt, das wie **Listing 3** und auf dem Bildschirm wie **Bild 9** aussieht.

Die Standard-Modulvorlage fügt die Takt- und Reset-Eingänge sowie einen Ausgang hinzu, der noch inaktiv ist. Um eine LED blinken zu lassen, müssen wir einen Zähler erstellen, der zum Umschalten der LED verwendet wird. Wenn wir die LED einfach nur in jedem Taktzyklus umschalten, würde sie viel zu schnell blinken, als dass wir das sehen könnten.

Um es einfacher zu machen, können wir die Verbindungsblöcke für Takt und Reset einrichten und dann darin das DFF deklarieren:

```
.clk(clk) {
  .rst(rst) {
    dff ctr[27];
  }
}
```

Dies erzeugt ein Array von 27 DFFs mit dem Namen *ctr*. Wir können dann das DFF in den *always*-Block einhängen.

```
always {
  ctr.d = ctr.q + 1;
  out = ctr.q[26];
}
```

Um auf ein Modul oder die Signale des DFFs zuzugreifen, verwenden wir die Punktschreibweise. Die erste Zeile des *always*-Blocks verbindet den D-Eingang des DFFs mit dem Q-Ausgang plus 1. Dadurch wird der Wert von *ctr.q* einmal pro Taktzyklus inkrementiert. Beachten Sie, dass das *d*-Signal nur schreibt und das *q*-Signal nur liest. Die zweite Zeile nimmt das höchstwertige Bit 26 und verbindet es mit dem Ausgang. Da *ctr* 27 Bits breit ist, besitzt es die Indizes 0...26. Da *ctr* einmal pro Taktzyklus inkrementiert wird, eine 27-Bit-Zahl 2^{27} = 134.217.728 verschiedene Werte enthalten kann und unsere Taktfrequenz 100 MHz beträgt, dürfte *ctr* einmal pro 1,34 Sekunden überlaufen. In der ersten Hälfte dieses Zyklus ist das höchstwertige Bit gleich 0, in der zweiten Hälfte ist es 1. Indem wir dieses Bit mit dem Ausgang verbinden, schalten wir den Ausgang alle 0,67 Sekunden um.

Wir können nun zum Top-Level-Modul zurückkehren und unser neues Modul instanziieren. Ich verwende ein Au-Board, aber das Modul würde für das Cu-Board genauso aussehen, nur dass der Name *cu_top* statt *au_top* lauten würde (**Listing 4**). Ich habe einen Verbindungsblock für das Reset-Signal hinzugefügt und eine Instanz des Blinker-Moduls mit dem Namen *myBlinker* erstellt. Dann habe ich es im *always*-Block mit Hilfe der Verkettungssyntax (*concatenation)* mit dem *led*-Ausgang verbunden. Die Elemente innerhalb von c{...} werden zu einem einzigen Array zusammengefügt. In unserem Fall verketten wir also sieben Nullen mit dem Bit aus *myBlinker.out*. Dadurch werden die sieben höchstwertigen LEDs ausgeschaltet und unser Blinkersignal mit der ersten LED verbunden. Sie können nun das Projekt erstellen, indem Sie auf das Hammersymbol klicken und es dann auf Ihre Platine laden, indem Sie auf das „Ausgefüllter-Pfeil-nach-unten"-Symbol drücken (**Bild 10**). Die oberste LED sollte nun etwas langsamer als einmal pro Sekunde blinken (**Bild 11**).

Listing 4.

```
module au_top (
    input clk,          // 100MHz clock
    input rst_n,        // reset button (active low)
    output led [8],     // 8 user controllable LEDs
    input usb_rx,       // USB->Serial input
    output usb_tx       // USB->Serial output
  ) {

  sig rst;              // reset signal

  .clk(clk) {
    // The reset conditioner is used to
    // synchronize the reset signal to the FPGA
    // clock. This ensures the entire FPGA comes
    // out of reset at the same time.
    reset_conditioner reset_cond;

    .rst(rst) {
      blinker myBlinker;
    }
  }

  always {
    reset_cond.in = ~rst_n; // input raw inverted
    //reset signal
    rst = reset_cond.out;   // conditioned reset

    led = c;

    usb_tx = usb_rx;        // echo the serial data
  }
}
```

Listing 5.

```
module blinker #(
    MAX_VALUE = 50000000 : MAX_VALUE > 0
  )(
    input clk,  // clock
    input rst,  // reset
    output out
  ) {

  .clk(clk) {
    .rst(rst) {
      dff ctr[$clog2(MAX_VALUE)];
      dff led;
    }
  }

  always {
    ctr.d = ctr.q + 1;
    if (ctr.q == MAX_VALUE-1) {
      ctr.d = 0;
      led.d = ~led.q;
    }

    out = led.q;
  }
}
```

Bild 11. Wenn Sie mir bis hierher gefolgt sind, liegt die Blinkrate der obersten LED unter einer Sekunde.

Verbesserungen am Modul

Wir können unser Blinker-Modul noch ein wenig verbessern. Erstens blinkt die LED in einem etwas ungünstigen Timing. Wir können dies auf eine Sekunde genau machen, indem wir bis 50.000.000 zählen und die LED dann umschalten. Dazu benötigen wir einen weiteren DFF, um den Zustand der LED zu speichern.

```
  .clk(clk) {
    .rst(rst) {
      dff ctr[28];
      dff led;
    }
  }
```

Im *always*-Block können wir nun prüfen, ob *ctr.q* 49.999.999 ist (da 0 bis 49.999.999 50.000.000 Inkremente erfordert) und wenn ja, können wir es auf 0 zurücksetzen und die LED umschalten.

```
  always {
    ctr.d = ctr.q + 1;
    if (ctr.q == 49999999) {
      ctr.d = 0;
      led.d = ~led.q;
    }

    out = led.q;
  }
```

Hier habe ich den bitweisen Invertierungsoperator ~, die Tilde, verwendet. Damit wird jedes Bit eines Signals invertiert. Da *led.q* nur ein Bit breit ist, wird dieses Bit einfach gespiegelt.

Ich habe vorhin gesagt, dass einem Signal unter allen Umständen ein Wert zugewiesen werden muss. Mit Ausnahme von DFFs! Da DFFs ihren Wert tatsächlich speichern können, ändert sich der Wert des DFFs nämlich nicht, wenn Sie den d-Eingang während eines Taktzyklus nicht zuweisen.

Unser Modul speichert jetzt nur die Werte von 0…49.999.999 in *ctr*, aber es ist immer noch ein 28-Bit-Array. Das ist verschwenderisch, da wir eigentlich nur 26 Bit benötigen. Wir könnten einfach die Arraygröße auf 26 reduzieren, aber wenn wir den Maximalwert ändern wollen, müssten wir diesen Wert jedes Mal neu berechnen. Stattdessen überlassen wir die Berechnungen den Lucid-Funktionen. Wir können *$clog2(50000000)* verwenden, was den Logarithmus zur Basis 2 des angegebenen Wertes berechnet. Das ist gleichbedeutend mit „wie viele Bits brauche ich, um so viele Kombinationen zu speichern?“. In unserem Fall lautet das Ergebnis 26.

```
dff ctr[$clog2(50000000)];
```

Apropos: Wir können unser Modul so bearbeiten, dass es den geänderten Maximalwert als Parameter akzeptiert, sodass er bei der Instanziierung des Moduls angegeben werden kann. Dazu fügen wir vor der Port-Liste eine Parameterliste zu unserem Modul hinzu.

```
module blinker #(
    MAX_VALUE = 50000000 : MAX_VALUE > 0
  )(
    input clk,  // clock
    input rst,  // reset
    output out
  ) {
```

Die Portliste wird mit der Syntax `#(param, param, param)` angegeben. Jeder Parameter kann so etwas einfaches wie ein Name (in Großbuchstaben) oder so komplex wie in unserem Beispiel sein, in dem wir einen Standardwert von 50.000.000 festlegen. Standardwerte sind in der Regel eine gute Idee, es sei denn, Sie möchten erzwingen, dass bei der Instanziierung ein Wert angegeben wird.

Nach der Standardzuweisung können Sie eine Bedingung angeben, die der Parameter erfüllen muss. Diese Bedingung kann (und sollte) sowohl den Parameter selbst als auch alle vor ihm deklarierten Parameter verwenden. Diese Bedingung muss als „wahr" ausgewertet werden, sonst wird bei der Instanziierung ein Fehler gemeldet. So können Sie Ihr Modul mit einigen Prämissen über den Parameterwert schreiben und Sie wissen, dass diese befolgt werden. Wir können dann das Auftauchen der 50.000.000 in unserem Modul durch `MAX_VALUE` ersetzen, wie in **Listing 5** gezeigt.

Wenn Sie das Projekt jetzt bauen und laden, sollte die LED mit einer Frequenz von einmal pro Sekunde blinken. Wenn Sie jedoch zurück zum Modul der obersten Ebene gehen, können Sie die Instanziierung leicht so ändern, dass die LED auch zweimal pro Sekunde blinkt.

```
blinker myBlinker(#MAX_VALUE(25000000));
```

Zusammengefasst

Wir wissen nun: FPGA-Designs bestehen aus Blöcken mit kombinatorischer Logik, die die gesamte Verarbeitung durchführen, und DFFs, die Werte speichern und den Datenfluss steuern. Die Designs selbst sind in Module unterteilt. Module können von anderen Modulen verwendet werden und können sogar Parameter haben, um jede ihrer Instanziierungen anzupassen. Jedes Mal, wenn ein Modul instanziiert wird, wird die Schaltung für dieses Modul im FPGA dupliziert.

Beim Design von Hardware ist es wichtig, darüber nachzudenken, wie dieses Design implementiert werden könnte, um effiziente Schaltungen zu erstellen. Im Fall unseres Blinkers würde die erste Version des Moduls viel weniger Ressourcen für die Implementierung benötigen, wenn wir uns nicht um die Blinkrate kümmern müssten. Das liegt daran, dass kein Komparator nötig ist, um den Wert des Zählers zu überprüfen. Es inkrementiert einfach weiter und nutzt den natürlichen Überlauf der binären Addition, um den Zähler zurückzusetzen.

200646-02

Surfen Sie auf der FPGA-Welle

Im Internet können Sie viel mehr über FPGAs erfahren. Hier ein paar Links, die Ihnen den Einstieg erleichtern oder Ihr Interesse wecken können.

- SparkFun and Alchitry, „Using FPGAs": www.sparkfun.com/fpga.
- J. Rajewski, „How Does an FPGA Work?", SparkFun: https://learn.sparkfun.com/tutorials/how-does-an-fpga-work
- J. Rajewski, „First FPGA Project: Getting Fancy with PWM", SparkFun: www.elektormagazine.de/esfe-en-fpga2
- J. Rajewski, „External IO and Metastability", SparkFun: https://learn.sparkfun.com/tutorials/external-io-and-metastability
- S. Cass, „Painless FPGA Programming", IEEE Spectrum, November 2020 https://spectrum.ieee.org/geek-life/hands-on/painless-fpga-programming.

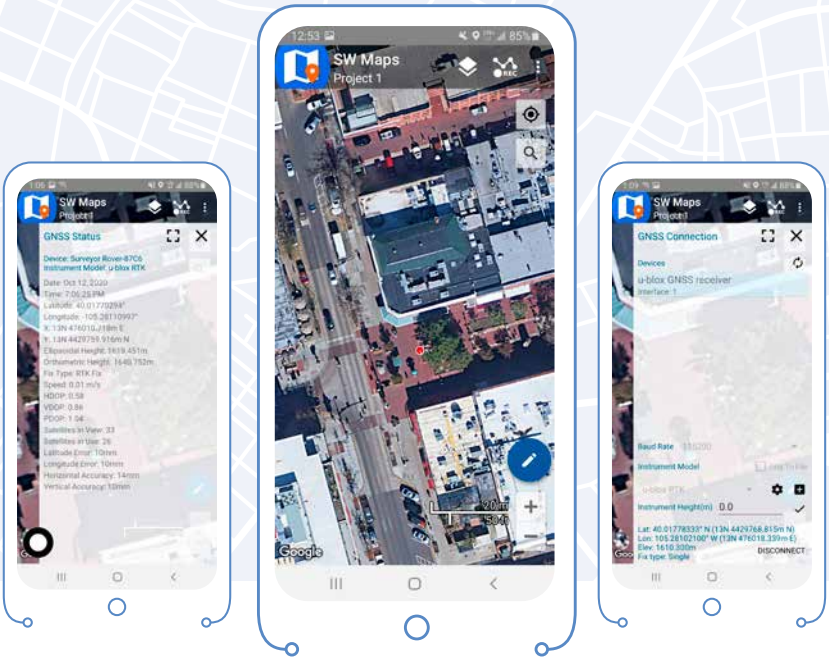
Alchitry und Justin Rajewski

Alchitry wird von KickStarter unterstützt und ist eine der kompetentesten und umfassendsten Ressourcen für die FPGA-Technologie. Das SparkFun-Team war von den Alchitry-Au- und Alchitry-Cu-Boards begeistert und nahm Kontakt auf zu Justin Rajewski, dem Gründer und Mädchen-für-alles-Ingenieur. Heute konzentriert sich Justin auf die Entwicklung neuer Produkte, den Bau von Projekten und die Generierung von Inhalten. Dabei nutzt er die Produktions- und Logistikerfahrung von SparkFun für die Herstellung seiner Boards. Für SparkFun bringt Justin eine unübertroffene Expertise in Sachen FPGAs mit.

WEBLINKS

[1] SparkFun, „Using FPGAs": https://www.sparkfun.com/fpga
[2] Wikipedia, Hardwarebeschreibungssprache: https://de.wikipedia.org/wiki/Hardwarebeschreibungssprache
[3] Lucid: https://alchitry.com/pages/lucid-fpga-tutorials
[4] Alchitry Labs: https://alchitry.com/blogs/tutorials/getting-started-with-the-au
[5] Alchitry Au board: https://www.elektormagazine.de/esfe-en-fpga1
[6] Alchitry Cu board: https://www.sparkfun.com/products/16526

Wie man eine **GNSS-Referenzstation** baut



Von Nathan Seidle (SparkFun)

Echtzeitkinematik oder Real Time Kinematics (RTK), die genaue Positionsbestimmung mobiler Rover mittels GNSS-Satellitensignalen ist eine faszinierende Sache, aber eines der größten Probleme dabei ist der Zugang zu Korrekturdaten. Dieser Artikel konzentriert sich auf den Aufbau einer eigenen festen Antenne auf dem Dach oder einer ähnlichen Struktur und die Konfiguration eines Minicomputers, um genau diese Daten über das Internet bereitzustellen, wo sie per WLAN oder, was häufiger der Fall ist, über ein Mobiltelefon oder Modem vom Rover abgerufen werden können.

In früheren Artikeln auf der SparkFun-Webseite habe ich beschrieben, wie man öffentlich zugängliche RTCM-Korrekturdaten erhält, die aber nur sporadisch verfügbar sind [1]. Ich habe auch beschrieben, wie Sie Ihre eigene temporäre Basis einrichten können, um RTCM-Korrekturdaten über eine Telemetrie-Funkverbindung zu senden [2], aber wenn Sie einen Kilometer oder mehr von Ihrer Basis entfernt sind, treten Probleme auf. Bevor Sie weiterlesen, stellen Sie sicher, dass Sie mit „Getting Started with U-Center“ [3] und dem Tutorial „What Is GPS RTK?“ von Sparkfun [4] vertraut sind. Betrachten Sie diesen Artikel als die Fortsetzung von „Setting up a Rover Base RTK System“ [5].

Wir werden viel über **NTRIP** sprechen (Networked Transport of RTCM via Internet Protocol), einem Verfahren zur Übertragung von RTCM-Korrekturdaten an einen Rover über das Internet. Stellen Sie es sich wie einen Musik-Stream für Ihren Rover vor. Sie müssen den Rover mit einer konstanten Musikquelle versorgen. Es gibt eine Vielzahl von Musikanbietern (YouTube, Spotify, Pandora), genau wie mögliche Quellen für RTCM-Daten (Trimble, Leica, Telit, und so weiter). Alle diese RTCM-Dienste verlangen unterschiedlich viel Geld und verhalten sich zudem verwirrend unterschiedlich.

Dieses Tutorial zeigt Ihnen, wie Sie Ihre eigenen GNSS-Korrekturdaten (Global Navigation Satellite System) generieren und ins Internet stellen können - und das alles kostenlos (oder für den Preis eines dedizierten Mini-PCs, wenn Sie so wollen)! Sie werden Ihr eigener Musik-

Streaming-Dienst sein! Ihr Rover wird die Korrekturdaten über eine Mobiltelefonverbindung abhören. Wir werden über NTRIP-Clients und -Server und Einhängepunkte sprechen, aber keine Sorge; es dreht sich nur um die Weitergabe von Bytes über das Internet, von einem Computer zum anderen.

Eine statische Basis mit Laserpräzision!

Im Tutorial [2] haben wir beschrieben, wie man eine temporäre Basisstation mit der 1- bis 10-minütigen Survey-In-Methode erstellt. Die Methode der temporären Basisstation ist flexibel, aber sie ist nicht so genau und kann in der benötigten Zeit stark variieren. Der ZED-F9P bietet eine viel schnellere Methode zur Bereitstellung von Basiskorrekturen: Wenn Sie den genauen Standort Ihrer Antenne kennen, können Sie die Koordinaten des Empfängers einstellen und er beginnt sofort mit der Bereitstellung von RTCM-Korrekturen. Das Problem ist: „Wo ist der Standort der Antenne?“ Das ist so, als ob Sie einen Lötkolben bräuchten, um Ihren Lötkolbenbausatz zusammenzubauen. Wo fangen wir an?

Q. Warum kann ich nicht einfach meine feste Antenne einmessen, um ihren Standort zu ermitteln?

A. Ein Survey-In ist zwar einfach einzurichten und eignet sich gut, um den Standort einer Basisstation im Feld zu bestimmen, wird aber nicht empfohlen, um den festen Standort einer statischen Basisstation zu ermitteln, da es wenig genau ist. Stattdessen wird die viel genauere Precise Point Positioning (PPP) zur Ermittlung der Antennenposition empfohlen. Es ist ein ähnlicher Prozess, bei dem jedoch Laser auf die armen Satelliten gerichtet werden!

Ein großes Problem ist, dass die vorhergesagten Bahnen oft um einen Meter oder mehr abweichen. Die Bodenstationen beschießen deshalb die einzelnen Satelliten beim Überflug mit Lasern und verwenden diese neuen Daten, um die tatsächlichen Bahnen der Satelliten zu berechnen. Mit diesen neuen Ephemeridendaten können in Verbindung mit den Rohdaten des Empfängers die Standorte besser berechnet werden. Dies ist die Grundlage von PPP, und der Prozess funktioniert wie folgt:

- Installieren Sie eine Antenne an einem festen Standort
- Sammeln Sie 24 Stunden GNSS-Rohdaten von dieser Antenne
- Geben Sie die Rohdaten weiter an ein Verarbeitungszentrum für PPP
- Erhalten Sie die hochgenaue Position der Antenne, die wir verwenden, um einen „Fixed Mode“ am Empfänger einzustellen.

Es gibt einige gute Artikel über PPP. Wir werden nur an der Oberfläche kratzen, aber für weitere Informationen lesen Sie bitte:

- Gary Miller: „PPP HOWTO“ [6];
- Emlid: „PPP“ [7];
- Suelynn Choy: „GNSS PPP“ [8].

Befestigen Sie Ihre Antenne

Sie möchten nicht, dass sich Ihre Antenne bewegt, nachdem Sie ihre Position festgelegt haben. Sie können viel in eine hochwertige Antenne investieren, aber wir haben die klassische L1/L2-Antenne von u-blox [9] mit gutem Erfolg verwendet. Montieren Sie die Antenne auf einer geeigneten Grundplatte auf einer festen Fläche, die eine sehr freie Sicht auf den Himmel hat, ohne Abschattung duurch Gebäude oder Bäume.

Wir haben die u-blox-Antenne an der eisenhaltigen Verkleidung auf dem Dach des SparkFun-Gebäudes befestigt (**Bild 1**). Der Magnetfuß der u-blox-Antenne hält zwar nicht allen Belastungen stand, doch – wie getestet – Windgeschwindigkeiten von über 160 km/h, was einem schweren Orkan entspricht. Die Antenne ANN-MB-00 von u-blox ist mit einem 5 m langen Kabel ausgestattet, was in der Regel nicht ausreicht, um vom (SparkFun-) Dach zum Empfänger zu gelangen, daher haben wir eine 10 m lange SMA-Verlängerung angebracht. Zwar haben die meisten L1/L2-Antennen einen eingebauten Verstärker, aber jeder Meter Verlängerung und jeder Stecker verschlechtert das GNSS-Signal leicht. Setzen Sie deshalb Verlängerungen und Adapter nur ein, wenn es notwendig ist, und halten Sie die Verbindung so kurz wie möglich.

Sie können auch eine höherwertige Vermessungsantenne verwenden, die keinen Magnetfuß besitzt, sondern ein zölliges Innengewinde (5/8 Zoll 11 UNC) an ihrer Unterseite. Um einen stabilen Fixpunkt zu schaffen, ohne dass Sie Löcher in Ihr Dach bohren müssen, haben die US-amerikanischen „Heimwerker“ von SparkFun eine Lösung gefunden, nämlich einen massiven Betonstein. Es sei Ihnen an dieser Stelle das handwerkliche Geschick nicht vorenthalten (**Bild 2** bis **Bild 6**), einen unpassenden (nur für Betonfunda-

Bild 1. Die u-blox-Antenne an der Dachbrüstung von SparkFun.

Bild 2. Die alte Wetterstation auf dem Stativ ist mit zwei schweren Betonsteinen gesichert.

Bild 3. Ja, das ist ein Betonstein. Lachen Sie nicht, er funktioniert!

Bild 4. Arrghh, die Hälfte des Betonstein-Bestandes ist vernichtet.

Bild 5. Der Bolzenanker wird nur leicht angezogen - und der Betonstein bleibt intakt!

Bild 6. Der Antennenfuß wird auf den Bolzenanker gedreht und mit der Schraube gekontert.

ment geeigneten) Bolzenanker in einem Betonblock zu befestigen, der natürlich beim Anziehen zerspringt. Es funktioniert nur, wenn man den „Dübel" mit der Hand anzieht (hält dann aber nicht besonders). Wir in Europa geraten aber gar nicht erst in Versuchung, denn solche Bolzenanker sind mit Zollgewinden hierzulande nicht zu bekommen. Statt dessen greift man zu einem Magnethalter aus dem Vermessungsartikel-Fachhandel, nicht billig, aber problemlos, und verwendet als „Masse" keinen Betonklotz, sondern eine dicke Stahlplatte.

Aufmerksame Leser werden meinen TNC-zu-SMA-Adapter im Bild bemerken. Er hat das falsche Geschlecht. Ursprünglich habe ich eine SMA-Verlängerung verwendet, um meine GPS-RTK-SMA mit meiner u-blox-L1/L2-Antenne auf dem Dach zu verbinden. Der GPS-RTK-SMA erwartet aber einen normalen SMA-Anschluss. Glücklicherweise habe ich eine Reihe von Adaptern und fand den richtigen TNC-zu-SMA-Adapter. Also, bevor Sie sich an die Außenmontage machen, schließen Sie alles testweise an!

Die SMA-Verlängerung habe ich einmal um den Sockel gewickelt. Falls etwas am SMA-Kabel zieht, wird die Kraft auf den Bolzen übertragen und nicht auf die TNC-Verbindung zur Antenne.

Warnung vor Blitzschlag: Mein Antennenprofil ist niedriger als die Brüstung (**Bild 7**), so dass Blitzeinschläge unwahrscheinlich sind. Ihre Antenne könnte aber der höchste Punkt in der Umgebung sein, daher sollten Sie einen Blitzschutz in Betracht ziehen.

Sammeln von GNSS-Rohdaten

Sobald Sie die Antenne an einem Ort platziert haben, an dem sie sich nicht bewegen kann, müssen wir ihren genauen Standort bestimmen. Öffnen Sie das Programm u-center und stellen Sie eine Verbindung mit Ihrem ZED-F9P her, so dass Sie mehr als 25 Satelliten sehen können. Angenommen, Sie haben einen guten Empfang, müssen Sie nun den Empfänger so einstellen, dass er Rohdaten von den Satelliten ausgibt (**Bild 8**). Sobald die *RXM-RAWX*-Meldung für USB aktiviert ist, überprüfen Sie den Empfang im Packet-Viewer (**Bild 9**). RAWX-Nachrichten sind binär, so dass Sie sie im Text-Viewer nicht sehen können.

Drücken Sie die den *Record*-Knopf (**Bild 10**), so dass alle Daten (NMEA, UBX und RAWX) vom Empfänger in einer *.ubx-Datei aufgezeichnet werden. Die Aufzeichnung von sechs Stunden ist gut, 24 sind etwas besser, wie der Positionsfehler in **Bild 11** zeigt (beachten Sie die logarithmische Skalierung!). Lassen Sie dies mindestens 24 Stunden lang laufen. Ein Tag erzeugt etwa 300 MB, also sollten Sie nicht einen ganzen Monat aufnehmen. Die meisten PPP-Analysedienste akzeptieren zwar mehr als 24 Stunden an Daten, kürzen sie aber möglicherweise auf 24 Stunden. Wenn Sie 30 Stunden RAWX-Daten erfassen, ist das in Ordnung, wir zeigen Ihnen, wie Sie eine zu lange Datei abschneiden können.

Die 300-MB-UBX-Datei muss in RINEX (Receiver Independent Exchange Format) konvertiert werden. Hier hilft das beliebte Programm *RTKLIB* [10]. Wir empfehlen die modifizierte Version der RTKLIB von rtklibexplorer, die Sie von [11] herunterladen können. Öffnen Sie *RTKCONV*. Wählen Sie Ihre .UBX-Datei und klicken Sie auf „Convert". Bei unserer 300-MB-Datei dauerte die Konvertierung etwa 30 Sekunden. Die Konvertierung sollte eine *.obs-Datei erzeugen.

Wenn Ihre Datendatei 25 Stunden oder etwas mehr umfasst, ist das in Ordnung. Wenn Sie Ihre RINEX-Datei verkleinern müssen, weil sie zu groß (sagen wir, 40 Stunden lang) ist, können Sie das Zeitfenster kürzen (**Bild 12**). Konvertieren Sie die gesamte Datei und klicken Sie dann auf das Notepad-Symbol, um die OBS-Datei zu öffnen. Sie sehen die GPS-Startzeit und die Stoppzeit für diese Aufnahme, wie in **Bild 13** dargestellt. Anhand dieser Zeiten können Sie das Zeitfenster auf die von Ihnen gewünschten Werte begrenzen und die Datei erneut konvertieren.

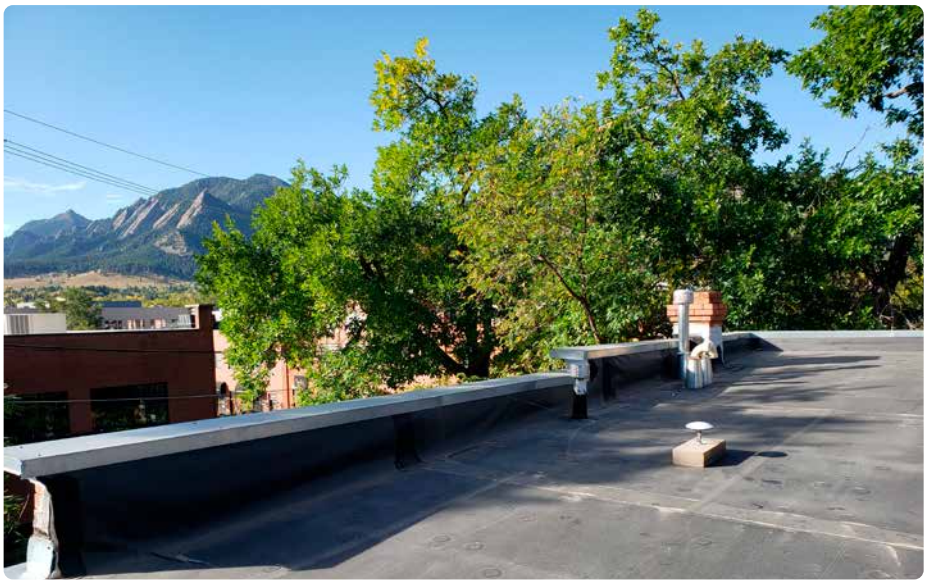


Bild 7. Die L1/L2-Antenne auf dem Flachdach widersteht einem Orkan! Das Hinaufschleppen des schweren Steins ist eine schweißtreibende Arbeit, aber dafür ist die Aussicht ziemlich spektakulär!

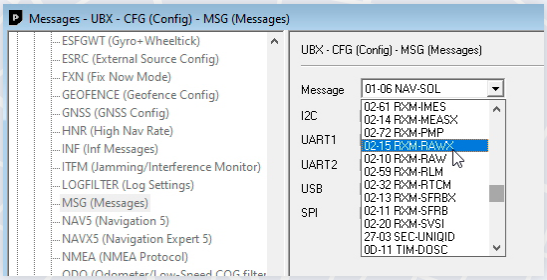


Bild 8. Auswahl des Rohdatenformats für die Nachrichten (MSG).

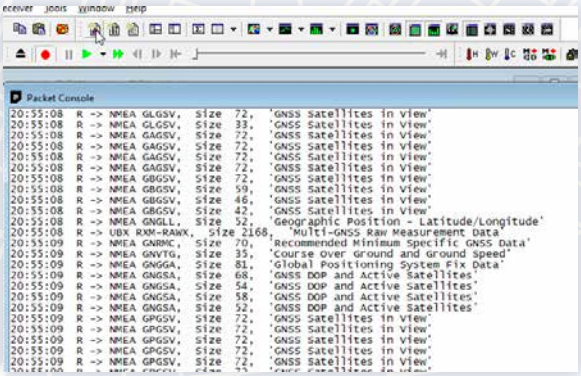


Bild 9. Ein RAWX-Paket im Packet Viewer.

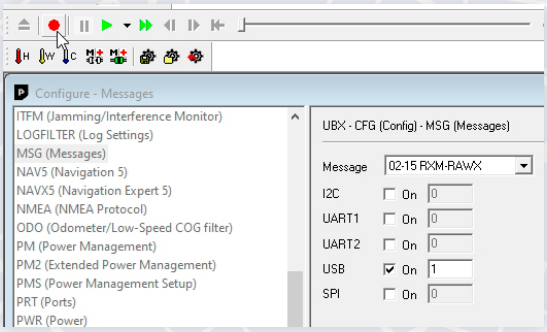


Bild 10. Wie beim Cassettenrekorder: Aufnahme läuft!

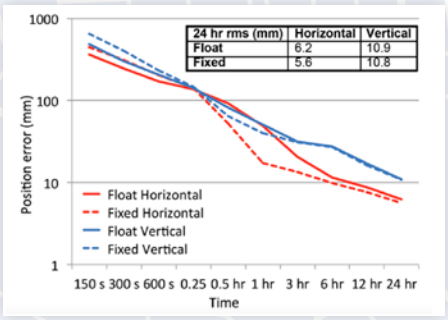


Bild 11. Der GNSS-Positionsfehler verringert sich mit der Aufnahmezeit (Suelynn Choy, 2018 [8]).

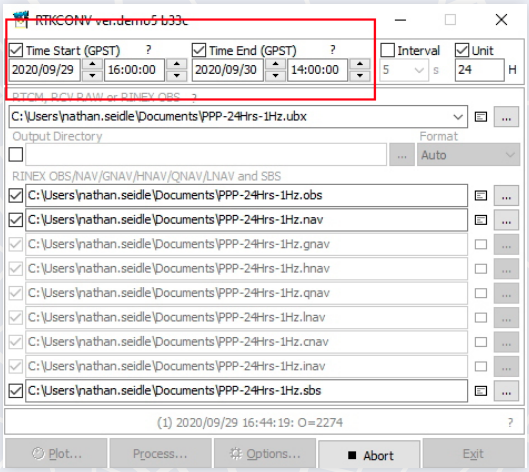


Bild 12. Durch Verkürzen des Zeitfensters hält man die Datengröße überschaubar.

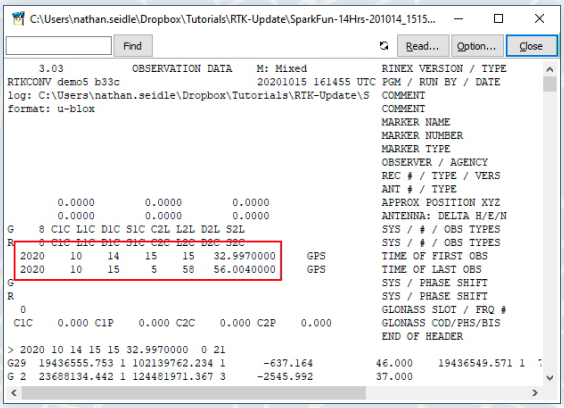


Bild 13. Blick in eine OBS-Datei, die auf 14 Datenstunden gekürzt wurde.

Bild 14. Alles bereit zum Senden der Daten an den kanadischen CSRS-PPP-Service.

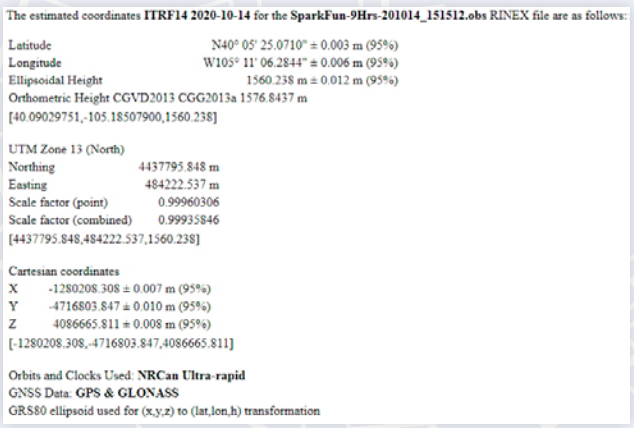


Bild 15. Die SparkFun-Antennenposition mit einem Fehler von nur ±7 mm!

Q. Warum drehen wir die Standortrate nicht höher? Mehr ist doch besser!

A. Der ZED-F9P kann bis zu 30 Hz gehen. Warum werden RAWX-Daten dann mit nicht mehr als 1 Hz aufgenommen? Weil sich die Natur nicht so schnell bewegt. Die meisten PPP-Analysedienste ignorieren alles, was größer als 1 Hz ist. OPUS geht sogar so weit, „decimate all recording rates to 30 seconds“. Und Ihre OBS-Dateien würden monströs groß werden. Wenn 24 Stunden bei 1 Hz 300 MB erzeugt, bedeutete dies, dass 24 Stunden bei 30 Hz etwa 9 Gigabyte wären. Also nein, belassen Sie es bei 1 Hz.

Wir müssen nun die rohen GNSS-Satellitendaten im RINEX-Format (*.obs) zu einem Post-Processing-Center schicken, um zu versuchen, den tatsächlichen Standort der Antenne zu ermitteln. Es gibt eine Handvoll von Diensten, aber wir hatten großes Glück mit dem kanadischen CSRS-PPP-Dienst [12]. Der US National Geodetic Service bietet den genannten Dienst namens OPUS [13] an, aber wir haben festgestellt, dass er durch Dateigrößen- und Formatprobleme frustrierend eingeschränkt ist. „Your mileage may vary“ …

Packen Sie Ihre .obs-Datei in ein Archiv und erstellen Sie ein Konto bei CSRS. Wählen Sie ITRF und laden Sie Ihre Datei hoch (**Bild 14**). Nach ein paar Stunden Däumchen drehen sollten Sie eine E-Mail mit einem schicken PDF-Bericht über den Standort Ihrer Antenne erhalten, wie (in Auszügen) **Bild 15** zeigt.

Wenn alles gut geht, sollte der Standort der Antenne sehr genau auf wenige Millimeter bestimmt sein. Für den u-blox-Empfänger sind besonders die ECEF-Koordinaten [14] interessant. Anstelle von Längen- und Breitengrad ist ECEF die Anzahl der Meter vom international vereinbarten Referenzrahmen des Erdmittelpunkts. Im Grunde genommen sind Ihre ECEF-Koordinaten die Entfernung, die Sie vom Erdmittelpunkt haben. Toll.

Nun, da Sie die ECEF-Position Ihrer Antenne haben, können Sie dem ZED-F9P mit einer Genauigkeit von wenigen Millimetern mitteilen, wo sich seine Antenne befindet.

Kehren Sie zur TMODE3-Meldung zurück und geben Sie die ECEF-Koordinaten aus dem Bericht ein (**Bild 16**). Angenommen, der Empfänger besitzt eine feste Antenne, empfehlen wir, diese Einstellungen im BBR/Flash zu speichern, so dass dieser Empfänger jedes Mal, wenn er eingeschaltet wird, sofort in den TIME-Modus geht und mit der Ausgabe von RTCM-Daten beginnt.

Fast unmittelbar nach der ECEF-Eingabe sollte Ihr Modul mit der Ausgabe von RTCM-Meldungen beginnen. Verwenden Sie den Paketbetrachter zur Bestätigung (**Bild 17**). Wenn Sie sie nicht sehen, schauen Sie im SparkFun-Tutorial nach, in dem beschrieben wird, wie Sie eine Basisstation einrichten [2]. Höchstwahrscheinlich haben Sie die erforderliche RTCM-Meldung nicht aktiviert. Auch hier sollten Sie die Einstellung in BBR/Flash speichern, damit der Empfänger bei jedem Einschalten die Korrekturdaten ohne Benutzereingriff sendet.

Mini-Computer-Einrichtung

Sie haben Ihre Antenne aufgebaut und ausgemessen und der ZED-F9P gibt RTCM aus. Gute Arbeit! Wie bekommen wir nun diese Daten in die Welt hinaus und zu den Rovern überall?

Sie benötigen einen mit dem Internet verbundenen, dedizierten Computer, der eine Verbindung zum ZED-F9P herstellt, der die seriellen Daten empfängt und diese Daten dann in das Internet weiterleitet. Wir empfehlen die Verwendung eines PCs für dieses Casting. Ja, Windows ist mühsam und nicht so stabil wie Linux, aber, da u-center nur unter Windows läuft, ist es wirklich die einzige Option. Es ist extrem praktisch, sich per Remote-Desktop in den dedizierten Rechner einwählen zu können, mit u-center an ein paar Empfängereinstellungen zu drehen und dann weiter zu senden. Ich bin mir nicht sicher, wie man die gleiche Flexibilität unter Linux erreichen kann. Außerdem bin ich kein Systemadministrator. Die folgende Anleitung kann hilfreich sein, um einen dedizierten Computer zu konfigurieren, aber eigentlich ist es nur meine Aufzeichnung meiner Notizen, damit ich das System bei Bedarf wiederherstellen kann.

- Beschaffen Sie einen ausrangierten Rechner mit Windows. Jeder alte PC sollte funktionieren, da Sie keine große Rechenleistung benötigen. Wir haben

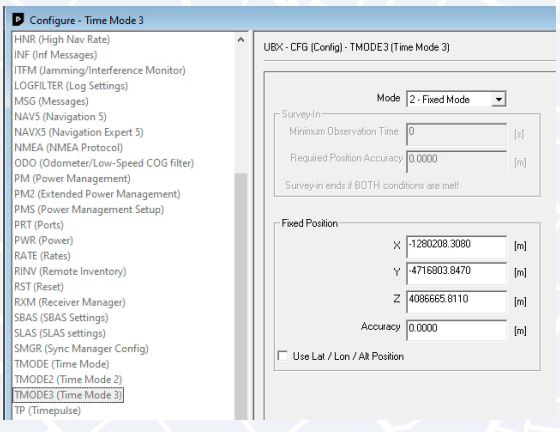


Bild 16. Kopieren der empfangenen ECEF-Daten in das Fenster Time Mode 3.

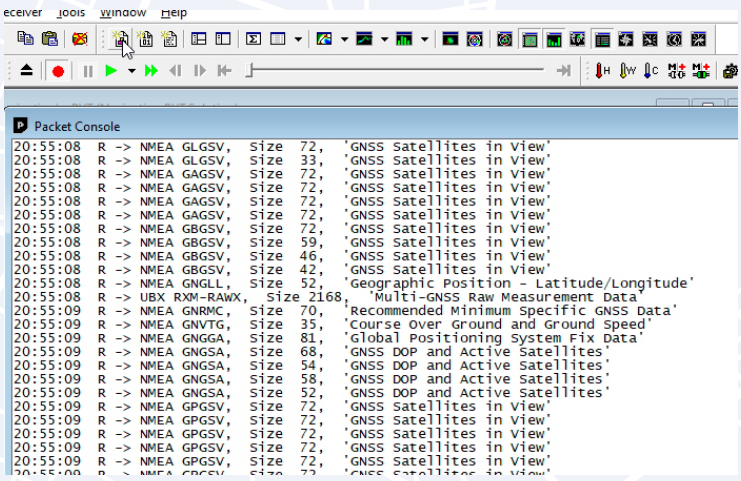


Bild 17. Packet Console mit den vom Modul gesendeten RTCM-Nachrichten.

Eigene GNSS-Korrekturdaten erstellen und kostenlos ins Internet stellen

Bild 18. Wetterfestes Orbit-Gehäuse mit eingebauter Steckdose (mit FI-Schutzschalter).

100 € für einen alten Mini-PC ausgegeben, der Befestigungsmaterial enthielt, so dass es leicht war, ihn an einer Wand oder in einem Außengehäuse zu befestigen. Achten Sie auf den Strombedarf des PCs, mit (bei unserem Modell gemessenen) 4,4 W kommt man auf nicht unerhebliche 30 €/Jahr!

- Aktivieren Sie im BIOS des Mini-PCs das automatische Wiedereinschalten nach einem Stromausfall. Aktivieren Sie Remote Desktop. Es gibt viele Tutorials, die zeigen, wie man das macht. Ziel ist es, den Mini-PC so weit zu bringen, dass Sie ihn bequem vom Schreibtisch aus konfigurieren und steuern können und dazu nicht auf den Speicher krabbeln müssen.
- Vorteilhaft ist eine statische IP-Adresse des Mini-PCs, entweder am Mini-PC selbst oder vom Router aus über die MAC-Adresse des Mini-PCs. Dadurch wird der Zugriff auf den kleinen Rechner über das RDP (Remote-Desktop-Protokoll) etwas einfacher.
- Wenn Sie von einem externen Netzwerk auf den Mini-PC zugreifen wollen, müssen Sie die Portweiterleitung auf Port 3389 (für RDP) auf der statischen IP-Adresse des Mini-PCs aktivieren.
- Wenn Sie das u-center als Caster nutzen wollen, müssen Sie die Portweiterleitung auf Port 2101 (für NTRIP) auf der statischen IP-Adresse des Mini-PCs aktivieren (mehr dazu später).
- Deaktivieren Sie die Energiesparfunktion von Windows. Der Mini-PC sollte sich nie ausschalten.

Vergewissern Sie sich jetzt, dass Sie per Remote-Desktop auf den PC zugreifen können. Wenn das gelingt, installieren Sie den Mini-PC „im Feld“. Es sollte nun möglich sein, alles per Fernzugriff zu konfigurieren.

- Wenn der Computer im Freien stehen wird, sollten Sie ihm ein wasserdichtes Gehäuse (IP65) gönnen. Das Orbit-Gehäuse [15] ist sehr schön (**Bild 18**)
- Schalten Sie alle Windows-Benachrichtigungen aus (verwenden Sie das Untermenü „Benachrichtigungen und Aktionen“)
- Deaktivieren Sie aus Sicherheitsgründen Bluetooth, und wenn Sie ein kabelgebundenes Ethernet verwenden, sollten Sie auch WLAN deaktivieren
- Installieren Sie u-center
- Installieren Sie eine geeignete rtkexplorer-Version von RTKLIB
- Installieren Sie ein serielles Terminal Ihrer Wahl
- Verbinden Sie das ZED-F9P über USB-C
- Schließen Sie die Antenne an
- Eventuell ein ISM-Funkadapter [16] über microUSB-B anschließen, nicht mit dem ZED-F9P-Breakout verlöten. Mehr dazu später.

Die Hardware-Installation ist individuell für jeden Standort. Bei mir zu Hause habe ich den Mini-PC im Haus, wobei das Antennenkabel durch einen Spalt im Fensterrahmen geführt wird. Bei SparkFun ist der Mini-PC in einem externen Schaltschrank mit Stromversorgung untergebracht (wirklich praktisch!) (**Bild 19**). Mit Hilfe von Klettband lässt sich die Elektronik leicht in dem Gehäuse einbauen, die Geräte sind an Ort und Stelle fixiert und der Benutzer kann das System dennoch bei Bedarf leicht auseinandernehmen. Sobald alles installiert ist, schalten Sie die Remote Desktop Connection ein und bestätigen, dass Sie den GNSS-Empfänger mit u-center konfigurieren können.

Wenn Sie dies noch nicht getan haben oder wenn sich Ihre Antenne inzwischen bewegt hat, sollten Sie die PPP-Vermessung Ihrer Antenne wie im vorherigen Abschnitt beschrieben erneut durchführen.

Caster-Einrichtung

Nun, da Sie den Windows-Mini-PC eingerichtet haben, lassen Sie uns über das Casting sprechen. Wie bereits erwähnt, ist NTRIP der Industriestandard für die Übertragung von RTCM-Korrekturen über das Internet. Für unsere Zwecke müssen

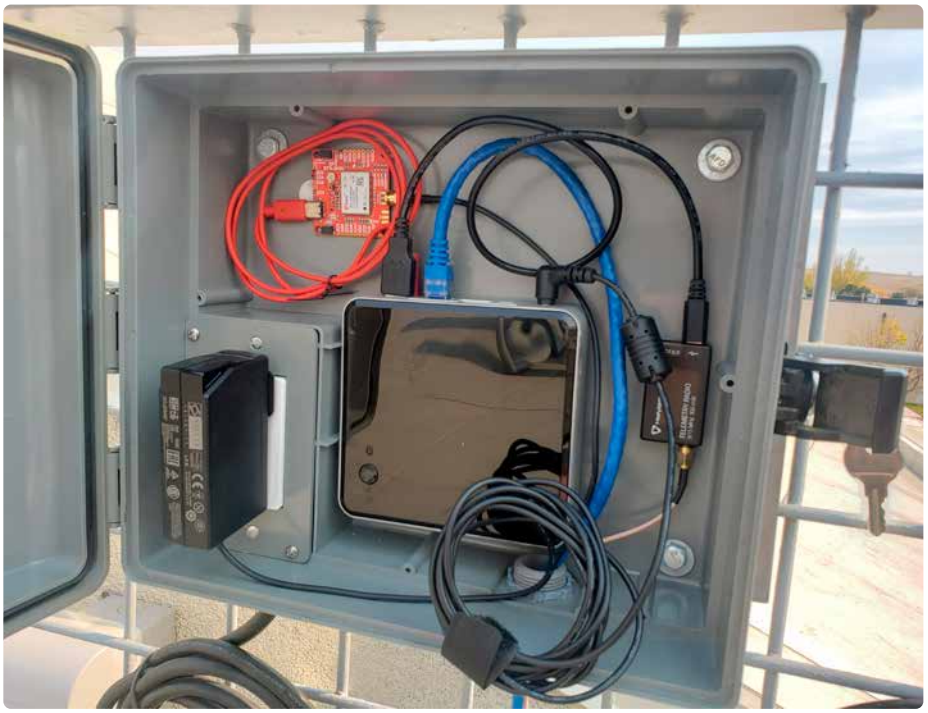


Bild 19. Mini-PC, verbunden mit einem ZED-F9P und einem über USB angeschlossenen ISM-Funkmodul.

wir von der Basisstation aus „casten“ und einen „Client“ am Rover verwenden, um Zugriff auf den Caster zu erhalten. Es gibt eine Reihe von Optionen:

- Verwendung des in u-center eingebauten Casters/Clients
- den in STRSVR eingebauten Caster verwenden
- Verwendung von STRSVR als Server und RTK2GO als Caster.

Und es gibt mehrere Möglichkeiten, Daten vom ZED-F9P ins Internet zu leiten. Wir beginnen mit der einfachsten.

U-center als Caster

U-center besitzt einen sehr einfach zu bedienenden *NTRIP*-Caster (**Bild 20**) mit der besten Benutzerfreundlichkeit. Geben Sie einfach einen Benutzernamen, ein Kennwort und Informationen zum Mount Point ein und klicken Sie auf OK. U-center konfiguriert den Empfänger **automatisch** so, dass er die RTCM-Sätze sendet und beginnt, Korrekturdaten über Port 2101 an jeden zu übertragen, der die IP-Adresse dieses PCs mit den richtigen Anmeldeinformationen erreicht (normalerweise mit einem NTRIP-Client).

Vorteile:

- Verblüffend einfach einzurichten.

Nachteile:

- Sie müssen im Router den Port 2101 freigeben, was nicht die sicherste Lösung ist.
- Wir empfehlen auf jeden Fall, mit dem u-center NTRIP „auszuprobieren“. Es ist sehr befriedigend und eine tolle Lernerfahrung, Korrekturdaten zu erhalten, aber langfristig betrachtet ist u-center nicht unsere erste Wahl.

STRSVR als Caster

Wirklich schnell, aber ich habe einen Tag gebraucht, um herauszufinden, dass die RTKLIB kein NTRIP-Casting unterstützt, obwohl das Manual es angibt (**Bild 21**).

Vorteile:

- STRSVR kann automatisch starten.

Nachteile:

- Der Caster macht nichts.

Bild 25. Die ISM-Antenne von SparkFun.

STRSVR und RTK2GO

Und der Gewinner ist: die Verwendung von STRSVR als NTRIP-Server und RTK2GO als NTRIP-Caster. Zugegeben, es ist verwirrend, aber es bedeutet, Daten auf einen Server im Internet hochzuladen.

Es gibt eine Vielzahl von Windows-Anwendungen, die behaupten, ein NTRIP-Caster zu sein. Wir haben festgestellt, dass sie im Allgemeinen durchweg nur **schrecklich** sind. Die einfachste Lösung ist die Verwendung von RTK2GO, scheinbar ein Lieblingsprojekt von SNIP. Wir empfehlen als beste Lösung, einen Mount Point und ein Passwort über RTK2GO.com zu erstellen.

Hinweis: Nach der Registrierung haben wir STRSVR gestartet und mit dem temporären Passwort von rtk2go.com verbunden. Sehr schnell, nach ein bis zwei Minuten, waren unser Account und unser Passwort aktiv. Das bedeutete aber, dass unser Broadcasting bei RTK2GO mit dem temporären Passwort ungültig wurde. Nach etwa 60 Sekunden ungültiger Verbindung durch STRSVR (weil das Password von temporär auf permanent geändert wurde) wurde unsere IP für ein paar Minuten, dann für

Stunden gesperrt. Unsere Empfehlung ist es deshalb, einfach ein paar Minuten zu warten. Verbinden Sie STRSVR nicht mit RTK2GO mit dem temporären Passwort. Warten Sie einfach auf die Bestätigungs-E-Mail von RTK2GO, antworten Sie mit der „Yes I'm not a robot"-E-Mail, und warten Sie auf die nächste E-Mail von RTK2GO, die besagt, dass Ihr Mount Point und Ihr Passwort gültig sind. Und erst dann starten Sie STRSVR mit Ihren Anmeldedaten.

Richten Sie STRSVR mit dem Mount Point und Passwort auf RTK2GO (**Bild 22**) ein. Da wir die Daten an einen NTRIP-Server übertragen, müssen wir den Port 2101 in unserem lokalen Netzwerk nicht öffnen.

Drücken Sie dann auf „Start". Innerhalb weniger Sekunden sollten die Kästchen auf der linken Seite Lichter grün aufleuchten und damit anzeigen, dass die Daten vom ZED-F9P korrekt an RTK2GO übertragen werden (**Bild 23**). Gut. Warten Sie eine Minute ab, öffnen Sie dann einen Browser und geben „rtk2go.com:2101" in der Adresszeile ein. Es sollte die Liste der aktuellen Einhängepunkte mit Ihrem Mount Point erscheinen. Wenn nicht, überprüfen Sie, ob Sie das richtige Passwort eingegeben haben, ob die Basis korrekt eingerichtet ist, ob RTCM-Meldungen eingeschaltet sind und ob der TIME-Modus aktiviert ist.

Hinweis: Wenn Sie STRSVR schließen, speichert es diese Einstellungen. Wenn Sie *strsvr.exe -auto* ausführen, wird STRSVR gestartet und das Casting automatisch gestartet (**Bild 24**).

Herzlichen Glückwunsch, Sie nähern sich der Ziellinie! Sie können sich jetzt eine Vermessungseinrichtung schnappen, ins Feld gehen und mit SW Maps eine Verbindung zu Ihren Korrekturdaten über den eingebauten NTRIP-Client herstellen. Sie können auch u-center als Client verwenden.

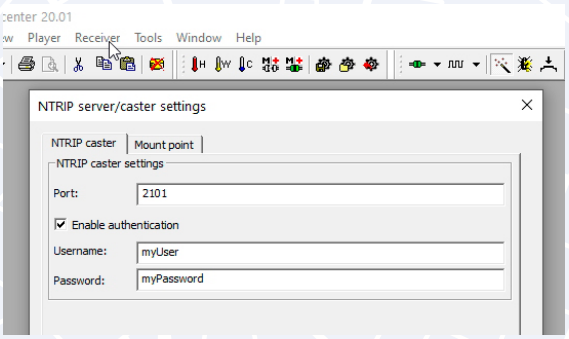


Bild 20. In u-center wählen Sie Receiver -> NTRIP server/caster.

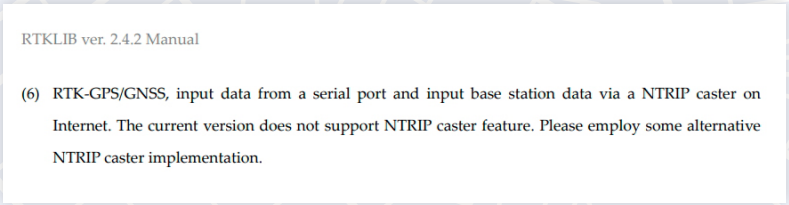


Bild 21. Da steht Caster! Lassen Sie sich nicht täuschen, es ist nicht so!

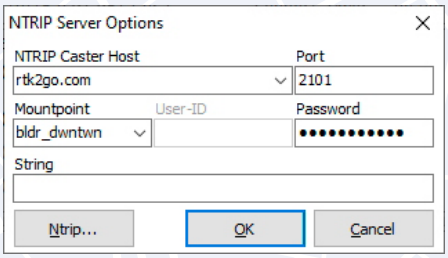


Bild 22. Auswahl von RTK2GO als Caster-Dienst.

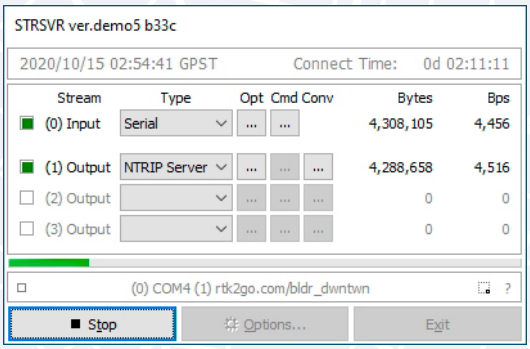


Bild 23. STRSVR sendet Daten an RTK2GO.

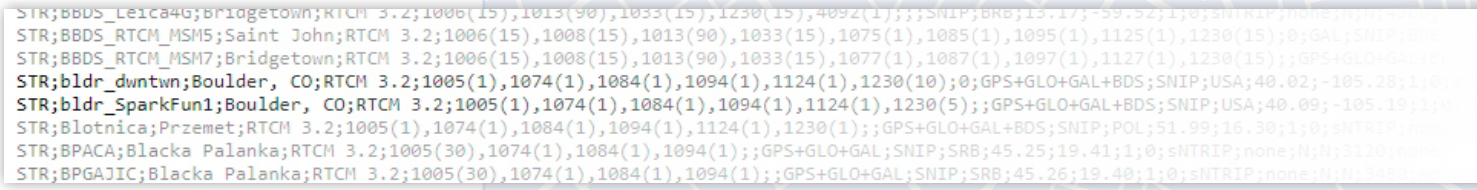


Bild 24. Zwei Mount Points, die sich zum Dienst melden!

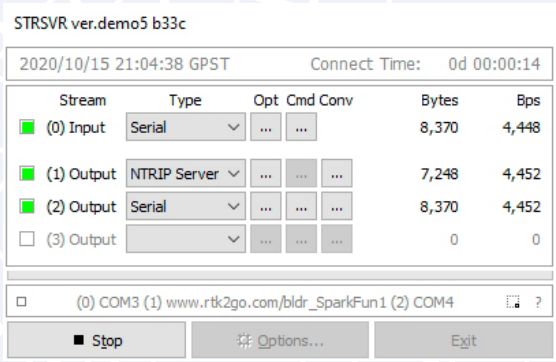


Bild 26. STRSVR ist so eingerichtet, dass zwei COM-Ports bedient werden.

Bild 27. Ändern der Einstellungen des Funkgeräts über AT-Befehle.

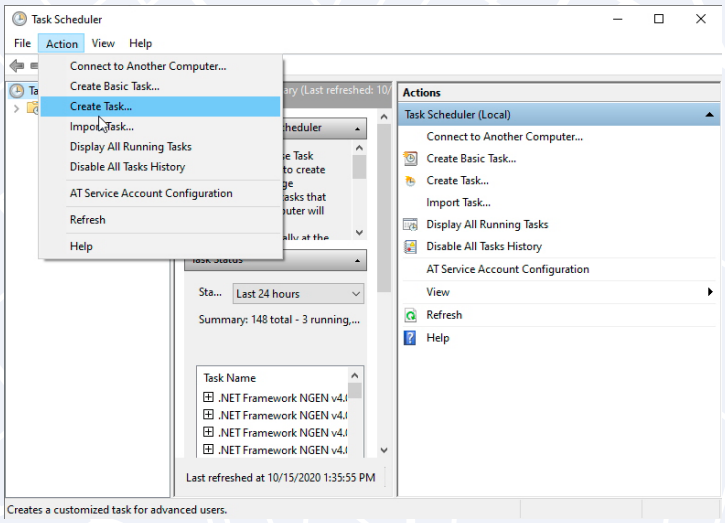


Bild 28. Anlegen einer neuen Aufgabe im Windows-Aufgabenplaner.

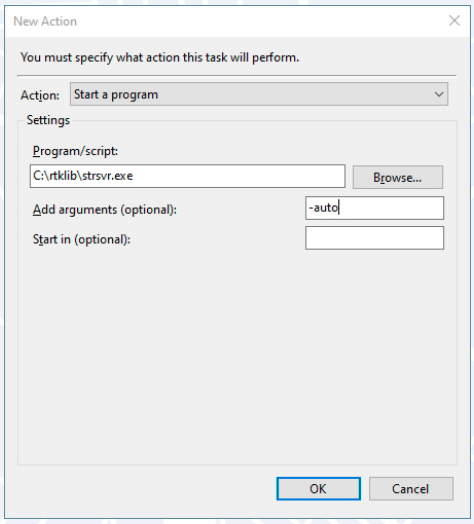


Bild 29. STRSVR muss -auto(matisch) starten!

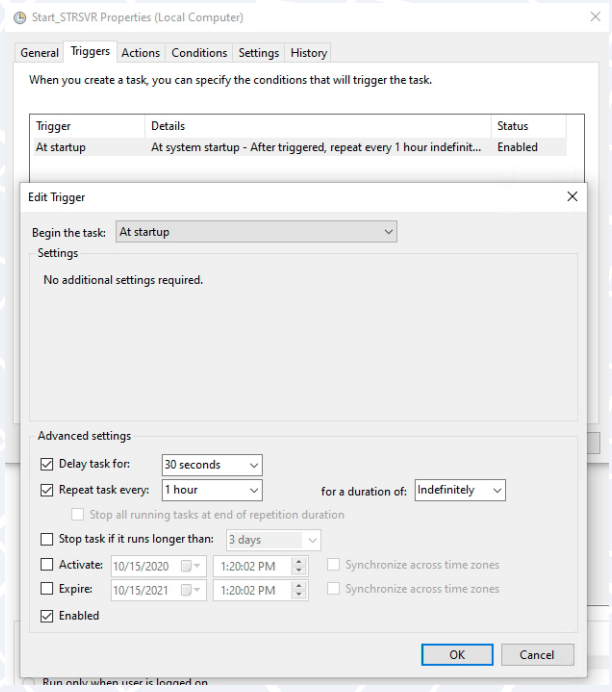


Bild 30. Einstellungen von delay task und repeat task für den automatischen Start von STRSVR.

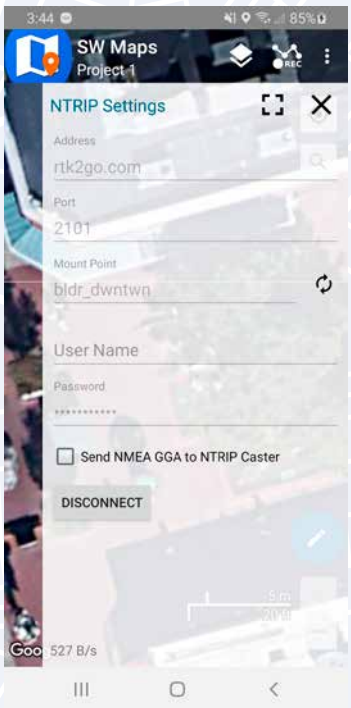


Bild 31. Mit SW-Maps erhält man die Korrekturdaten über NTRIP.

Für Eingeweihte - ein Funkgerät hinzufügen

Natürlich haben wir eine ISM-Antenne auf dem Dach von SparkFun (**Bild 25**). Warum sie also nicht anschließen? Im Gegensatz zu der kleinen GNSS-Antenne im vorherigen Abschnitt hat diese Antenne einen **wirksamen** Blitzschutz.

STRSVR ist sehr leistungsfähig. Es kann Ihre RTCM-Daten an mehrere Stellen weiterleiten, nicht nur an einen NTRIP-Server. Wir haben ein 100-mW-ISM-Funkgerät [16] an USB angeschlossen, das als COM-Port enumeriert wurde. Wir können dann diesen COM-Port (denken Sie an die korrekte Baudrate!) zu STRSVR hinzufügen, damit die RTCM-Daten **sowohl** an den NTRIP-Server als auch an das Funkgerät gehen (**Bild 26**). Dies ermöglicht es uns, eine Funkverbindung für RTK im lokalen Bereich zu verwenden und auf eine Mobilfunkverbindung umzuschalten, wenn wir uns außerhalb der Reichweite des Funkgeräts befinden.

Weiterführende Lektüre

Möchten Sie mehr über RTK wissen? Dann sehen Sie sich diese SparkFun-Tutorials an:

1. Getting Started with U-Center for u-blox. Tipps und Tricks zur Verwendung der u-blox-Software, um Ihren GPS-Empfänger zu konfigurieren.
https://learn.sparkfun.com/tutorials/getting-started-with-u-center-for-u-blox

2. GPS-RTK2 Hookup Guide. Präzision auf den Millimeter mit dem neuen ZED-F9P von u-blox.
https://learn.sparkfun.com/tutorials/gps-rtk2-hookup-guide

Q. Warum wird das Funkgerät nicht an UART2 angelötet?

A. Durch den Anschluss des Funkgeräts an USB können wir das Funkgerät [17] über ein Terminalfenster konfigurieren (**Bild 27**). Wäre es direkt mit dem UART2 am ZED-F9P verdrahtet, würden zwar die Korrekturdaten übertragen, aber wir könnten nicht die AIR_SPEED-Parameter [18] des Funkgeräts und andere Einstellungen ändern, um eine größere Reichweite zu erzielen.

Aufgabenplanung

Wir brauchen STRSVR, um automatisch zu starten. STRSVR merkt sich seine Einstellungen und startet automatisch mit den zuletzt verwendeten Werten, indem Sie *strsvr.exe -auto* ausführen. Lassen Sie uns nun eine Aufgabe in Windows erstellen, um sicherzustellen, dass STRSVR bei jedem Einschalten startet.

Öffnen Sie den Windows-Aufgabenplaner und und erstellen Sie eine neue Aufgabe (**Bild 28**). Weisen Sie dann die Aufgabe an, STRSVR stets mit „-auto“ zu starten (**Bild 29**). Die wichtigsten Dinge, die der optionale Befehl „-auto“ ausführen soll, sind das Starten des Tasks beim Hochfahren, ohne dass eine Anmeldung erforderlich ist,

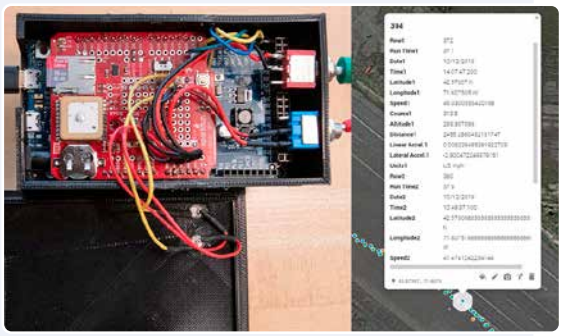
Anwendungen der Community

GPS-RTK lässt Träume vom Orientierungslauf wahr werden

Der passionierte Orientierungsläufer Don Bayly hatte mehrere Jahre lang einen GPS-Glonass-Empfänger mit einer Frequenz verwendet, der in der Regel auf weniger als fünf Meter genau war. Allerdings waren oft Anpassungen erforderlich, da die Positionen zu verschiedenen Zeiten abwichen. Als er begann, nach genaueren GPS-Empfängern zu suchen, stellte er fest, dass ein zentimetergenaues RTK-Gerät mehrere zehntausend Dollar und selbst ein GIS-Gerät mit einer Genauigkeit von einem Meter rund 2.000 Dollar kostete.

Dann stieß er auf das GPS-RTK2-Board ZED-F9P (Qwiic) von SparkFun zu einem Preis von 200 Dollar und einem Fehler von 10 mm in allen drei Dimensionen im Rover- und Basisstationsbetrieb. Zunächst dachte er, dass der Mangel an UNAVCO-Basisstationen in Kanada den Einsatz des Boards unpraktisch machen würde, als er jedoch weiter forschte, erfuhr er vom BKG NTRIP-Service und dessen Basisstationen auf der ganzen Welt - auch in Calgary, wo er lebt. Daraufhin entschied er sich, das GPS-RTK von SparkFun auszuprobieren.

u-blox-GPS für mehr Präzision



Anker Berg-Sonne bemerkte, dass geringe Geschwindigkeit, mit der die Daten auf die microSD-Karte geschrieben werden können, ein Problem darstellte. Damit der Arduino Mega 2560 mit dem 10-Hertz-Datenstrom vom GPS mithalten konnte, durfte er die Daten nur kurz analysieren und dann mit so wenig Verarbeitung wie möglich an die microSD-Karte senden.

Die Aufzeichnung von Motocross-Renndaten stellt eine besondere Herausforderung dar, da sich der Kurs bei jedem Rennen ändert. Daher ließ Berg-Sonne den Logger in zwei Modi arbeiten. In einem Modus markierte er die GPS-Koordinaten des mit Leitkegeln abgesteckten Kurses, indem er an jedem Kegel den Knopf am Logger drückte. Im anderen Modus wurden die Daten jedes Mal aufgezeichnet, wenn die Geschwindigkeit des Fahrzeugs 10 mph überschritt.

„Es stellte sich heraus, dass der Parameter ‚Fahrzeuggeschwindigkeit‘ zum Auslösen der Aufzeichnung wirklich gut funktionierte“, sagte er. „Bei den meisten kommerziellen Datenloggern muss man die Aufzeichnung manuell ein- und ausschalten oder dazu sehr genau Start und Ziel markieren. Motocross ist sehr anstrengend und man kann leicht vergessen, den Logger zu Beginn eines Laufs zu starten. Mit meinem Logger muss man sich darüber keine Gedanken mehr machen.“

Passende Produkte

Suchen Sie nach den in diesem Artikel erwähnten Produkten?
SparkFun und Elektor haben die passenden Produkte für Sie!

GNSS Multi-Band Magnetic Mount Antenna - 5m (SMA)
www.elektormagazine.de/esfe-en-diygnss1

SparkFun 915 MHz radio module
www.elektormagazine.de/esfe-en-diygnss2

Elektor-Hinweis:
SparkFun 915-MHz-Funkmodul [16] ggf. durch funktional gleichwertigen 868-MHz-Typ ersetzen.

und die Verzögerung von 30 Sekunden (bis 60 Sekunden). Wir haben festgestellt, dass dieser Task (STRSVR) beim Windows-Start nicht gestartet wurde, möglicherweise weil das ZED-F9P und/oder die Funk-COM-Ports noch nicht enumeriert waren. Eine Verzögerung von 30 Sekunden oder mehr behebt das Problem (**Bild 30**). Und wenn das Programm jemals aus irgendeinem Grund herunterfährt oder abstürzt, sollte der Aufgabenplaner jede Stunde STRSVR neu starten, wenn das Programm nicht bereits läuft.
Sobald die Aufgabe eingerichtet ist, setzen Sie Ihren Mini-PC zurück und überprüfen Sie, ob STRSVR automatisch startet und mit dem Senden beginnt.

Im Einsatz!

Das war's! Sie sollten nun in der Lage sein, den NTRIP-Client Ihrer Wahl auf RTK2GO.com, Port 2101, mit Ihrem Mount-Punkt und Passwort zu richten und die Korrekturdaten von Ihrer Basis an eine beliebige Anzahl von Rovern im Umkreis von etwa 10 km von Ihrer Basis zu empfangen. Zur Darstellung empfehlen wir SW Maps [19] (**Bild 31**), da es unglaublich einfach ist, die Korrekturdaten von RTK2GO über das Mobilfunknetz abzurufen und diese Daten automatisch an einen Rover-konfigurierten ZED-F9P weiterzuleiten. Es kann ein wenig Arbeit sein, eine dedizierte Korrektur-Basisstation einzurichten, aber mit der einmaligen Einrichtung sollte die Basis viele Monate oder Jahre lang ohne Aufsicht laufen.
Es hat mir sehr viel Spaß gemacht, etwas über RTK und Vermessungstechnik zu erfahren. Aber ich habe auch gelernt, dass man sich nicht zu sehr darauf versteifen sollte, zu wissen, wo genau auf der Welt dieser Punkt ist. Der Raum ist relativ und verändert sich genauso wie die Zeit. Ich will Ihre Weltsicht nicht erschüttern, aber die nordamerikanische tektonische Platte bewegt sich mit 2 cm pro Jahr [20] auf Europa zu. Ob das gut ist oder schlecht, genießen Sie einfach das Hier und Jetzt, wo immer es auch sei. ▶|

200660-02

Elektor-Projektseite für diesen Artikel:
www.elektormagazine.de/esfe-en-diygnss

WEBLINKS

[1] Öffentlich zugängliche RTCM-Korrekturdaten: https://bit.ly/SF-RTCM-correction
[2] Setting up a Rover Base RTK System: https://bit.ly/SF-rover-base
[3] Getting Started with U-Center: https://bit.ly/SF-U-Center
[4] Tutorial: What is GPS RTK?: https://bit.ly/SF-GPS-RTK
[5] Setting up a Rover Base RTK System: https://bit.ly/SF-rover-base
[6] Garry Miller: Precise Point Positioning (PPP) HOWTO: https://gpsd.gitlab.io/gpsd/ppp-howto.html
[7] Emlid: Precise Point Positioning (PPP): https://docs.emlid.com/reachrs/common/tutorials/ppp-introduction/
[8] Suelynn Choy: GNSS Precise Point Positioning (PPP): https://bit.ly/GNSS-PPP
[9] u-blox-Antenne: https://www.elektormagazine.de/esfe-en-diygnss1
[10] RTKLIB: http://www.rtklib.com
[11] rtklibexplorer: http://rtkexplorer.com/downloads/rtklib-code/
[12] CSRS-PPP-Service: https://bit.ly/NRCAN-PPP
[13] OPUS: https://www.ngs.noaa.gov/OPUS/
[14] Geozentrisches Koordinatensystem (ECEF): https://de.wikipedia.org/wiki/Geozentrisches_Koordinatensystem
[15] Outdoor-Schaltschrank „Orbit": https://bit.ly/orbit-enclosure
[16] Serielles Telemetrie-Funksystem (915 MHz, nicht für Europa!): https://www.elektormagazine.de/esfe-en-diygnss2
[17] SiK Radio, Konfiguration über Terminal: https://bit.ly/ardupilot-radio
[18] AIR_SPEED-Einstellung: https://bit.ly/2LLzg8e
[19] SW Maps: https://bit.ly/google-sw-maps
[20] Kontinentaldrift: https://de.wikipedia.org/wiki/Plattentektonik

u-blox und SparkFun

Technik ist komplex. Um die nützlichsten Produkte auf den Markt zu bringen, müssen Unternehmen zusammenarbeiten. SparkFun arbeitet seit mehreren Jahren mit u-blox zusammen, um innovative Positionierungs-, Kommunikations- und Timing-Boards zu entwickeln. u-blox hat in der Vergangenheit immer wieder neue innovative Produkte auf den Markt gebracht, und SparkFun macht diese einfacher anwendbar und beschleunigt Prototyping, Forschung und Entwicklung.

Grrr. Stecken geblieben?

Sie finden unsere Elektor Projekte richtig klasse, kommen aber aus irgendeinem Grund nicht weiter und benötigen Hilfe? Oder haben Sie gar eine Idee oder einen Kommentar zu einem bestimmten Artikel?

Kein Problem. Unsere Elektor-Ingenieure, -Redakteure und Community-Mitglieder sind auch in den **sozialen Medien aktiv**.

Sie können uns hier finden:

www.elektor.de/FB

www.elektor.de/TW

www.elektor.de/YT

www.elektor.de/insta

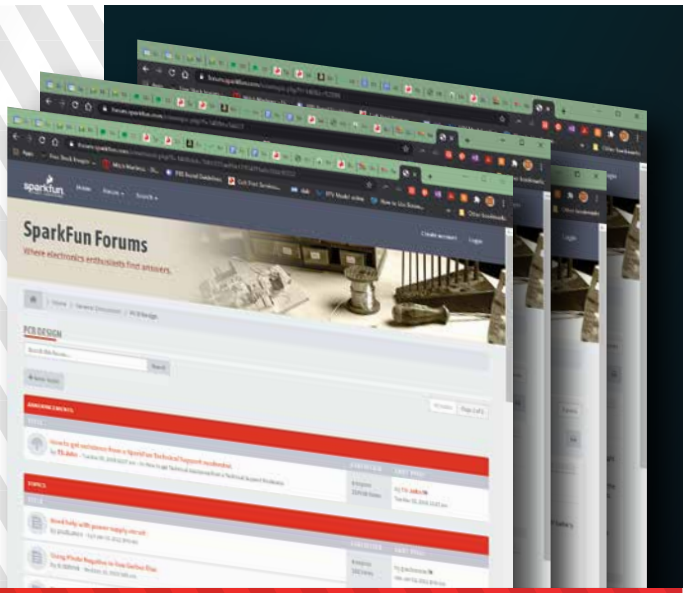
www.elektor.de/LI

elektor
design > share > sell

SPARKFUN-FOREN

Hier können Sie diskutieren, Probleme teilen und lösen.

Sie brauchen technische Unterstützung oder Hilfe bei einem Projekt? Besuchen Sie die SparkFun-Foren! Die Foren sind eine großartige Möglichkeit, um sich mit Leuten auszutauschen, die an Projekten arbeiten, oder um dem SparkFun-Team Fragen zu stellen über ein Produkt, das Sie gekauft haben, oder über ein Problem, das Sie haben.



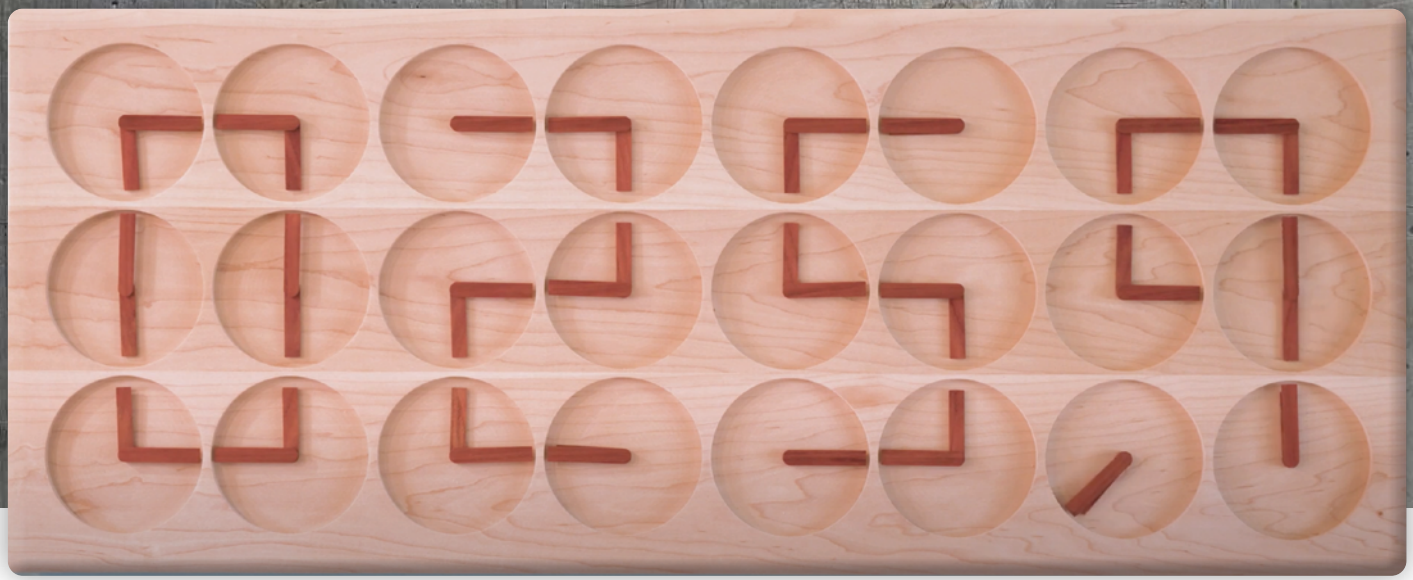
SCHAUEN SIE SICH DIE FOREN AN UNTER

www.forum.sparkfun.com

sparkfun®
START SOMETHING

CLOCKCLOCK

Zeit-Anzeige der besonderen Art



Von Justin Rajewski (Alchitry) (USA)

Zeit-Anzeige einmal anders: In diesem ausgefallenen Projekt bauen wir mit einem FPGA-Entwicklungsboard von Alchitry eine Digital-Uhr, deren vier Ziffern aus den Zeigern von 24 einzelnen Analog-Uhren gebildet werden.

Die Idee zu diesem Projekt stammt eigentlich gar nicht von mir: Ich bin schon vor ein paar Jahren auf dieses Konzept gestoßen, da es sich aufgrund der vielen Steuersignale als ein tolles FPGA-Demo-Projekt anbietet. Die ursprüngliche Uhr von „Humans“ aus dem Jahre 1982 finden Sie unter [1].

Es gibt mehrere Gründe, warum sich diese Uhr so gut als FPGA-Demoprojekt eignet. Erstens: Sie benötigt 48 Schrittmotoren, die zusammen 24 einzelne „Uhren“ bilden. Jede davon hat, wie es sich gehört, zwei unabhängig voneinander arbeitende Zeiger. Die Verwendung eines bidirektionalen Schrittmotortreibers bedeutet, dass Sie insgesamt 96 Steuersignale benötigen (zwei pro Motor). Da ich, um Strom zu sparen, in der Lage sein wollte, die Treiber beim Stillstand der Uhr zu deaktivieren, wurden vier weitere Ausgänge hinzugefügt (einer für jede „Ziffer“). Zum Erzeugen der Bewegungen habe ich einen Arduino eingesetzt, da es viel einfacher ist, einen Code anstelle von Hardware zu verwenden. Die Kommunikation mit dem Arduino erfolgt mittels I²C über den Qwiic-Anschluss der Alchitry Au: Zwei weitere von insgesamt 102 I/O-Pins. Praktischerweise besitzt das Alchitry Au Board genau diese Anzahl von I/O-Pins [2].

Dieses Projekt zeigt, dass FPGAs eine große Zahl an I/O-Schnittstellen steuern können. Außerdem nutzt es den Qwiic-Anschluss des FPGAs auf eine ziemlich unkonventionelle Weise. Das FPGA fungiert in diesem Projekt nämlich als Peripherie und nicht als

Controller. Dessen Rolle wird hier vom Arduino übernommen, der alle Befehle an die FPGA-Einheit weitergibt. Vielleicht ein nützliches Vorbild für weitere Projekte.
Oft sind bestimmte Aufgaben per Software sehr einfach realisierbar und mittels Hardware recht kompliziert, und manchmal ist es auch umgekehrt. Indem man einen Mikrocontroller und ein FPGA miteinander kombiniert, erhält man das Beste aus beiden Welten, was durch den Qwiic-Anschluss auf beiden Boards sogar noch vereinfacht wird.

Werkzeug, Hardware und sonstige Materialien

Zur Bewältigung des vorliegenden Projektes benötigen Sie die unter *Passende Produkte* aufgeführten Materialien, sofern Sie die diese noch nicht besitzen. Legen Sie die Produkte in Ihren Warenkorb, lesen Sie sich die Anleitung durch und passen Sie den Warenkorb bei Bedarf an. Hier die verwendeten Werkzeuge:

- 3D-Drucker
- CNC-Fräse Shapeoko XXL
- Bandsäge, Hobel und Exzenterschleifer für die Holzbearbeitung

Außerdem:

- 48× Getriebe-Schrittmotoren [3]
- 48× StepStick-Schrittmotor-Treibermodul mit Kühlkörper [4]
- 1× UBEC einstellbares BEC 2-6S für Quadcopter RC-Drohne [5]
- 1× AC-DC-Schaltnetzteil im Gehäuse [6]

Empfohlene Literatur

Wenn Sie mit dem Qwiic-System nicht vertraut sind, empfehlen wir Ihnen, die Dokumente unter *www.sparkfun.com/qwiic* zu lesen. Wir empfehlen außerdem, sich die folgenden Tutorials anzusehen, bevor Sie fortfahren:

- *Programming an FPGA* [7]. Schauen Sie sich die Grundlagen der Arbeit mit Field Programmable Gate Arrays, auch bekannt als FPGAs, an.
- *How Does an FPGA Work* [8]. Das Was, Wie, Warum und Wann von FPGAs.
- *First FPGA Project - Getting Fancy with PWM* [9]. Ein erstes Projekt, das Alchitrys Onboard-FPGA zur PWM-Steuerung verwendet.

Mechanischer Aufbau

In diesem Abschnitt werde ich mich recht kurz fassen, da der Schwerpunkt dieses Tutorials auf dem Thema „FPGA“ und nicht auf Holzarbeiten liegt. Folgen Sie den unten stehenden Links, um auf die Projektdateien zuzugreifen und sie herunterzuladen.

- CAD-Datei (Fusion 360) [10]
- Alchitry (FPGA) [11]
- Arduino Code (ZIP) [12]

Zunächst wollte ich herausfinden, wie man ein einziges der benötigten Uhrwerke herstellt. Ich brauchte dazu zwei Schrittmotoren und eine Möglichkeit, diese mit zwei konzentrischen Ausgangswellen zu verbinden. Ursprünglich verwendete ich die extrem kleinen, nur 8x9,2 mm² großen Schrittmotoren, die bei Amazon angeboten wurden. Ich entwarf eine Einheit mit zwei großen Zahnrädern und zwei kleinen Zahnrädern, die auf die Motoren gepresst werden sollten (**Bild 1**). Leider waren diese kleinen Motoren nicht imstande, die Zahnräder zu drehen. Sie hatten nur ein sehr geringes Drehmoment, und beim Versuch, die Leistung zu erhöhen, erhitzten sie sich so stark, dass sie die 3D-gedruckten Plastikteile zum Schmelzen brachten.
Nach diesem Misserfolg bestellte ich ein paar Schrittmotoren vom Typ28BYJ-48. Diese sind etwas größer, aber immer noch klein genug für die Uhr. Sie besitzen ein inneres Getriebe und ein höheres Drehmoment. Die Getriebe verleihen ihnen genügend Innenwiderstand, um ihre Position auch nach dem Ausschalten zu halten. Ich entwarf ein Werk, bei dem der Minutenzeiger direkt von der Motorwelle und der Stundenzeiger über ein Getriebe angetrieben wird, so dass der Motor zur Seite hin versetzt werden kann. Der Rest der Geschichte ist am besten mit Bildern und einigen kurzen Kommentaren erzählt.

FPGA

Bei jeder FPGA-Entwicklung ist es wichtig, dessen Aufgabe vor dem Start zu umreißen. In dieser bestand das Ziel darin, Befehle über I²C annehmen und die Motoren entsprechend zu schalten. Ich entschied mich für Befehle, die aus einer Anzahl von Schritten und einem Wert bestehen, der der Periode zwischen den Schritten entspricht. Ich wollte außerdem auch eine Reihe von Befehlen in einer Warteschlange unterbringen. Dann wäre das Qwiic-Timing weniger wichtig, da alle Befehle einfach nacheinander ausgeführt werden würden. Schließlich musste ich noch herausfinden, zu welchen genauen Zeitpunkten irgendwelche Schritte ausgegeben werden müssen, um die Motoren dann entsprechend zu aktivieren beziehungsweise zu deaktivieren, um Strom zu sparen.

Der Animator

Für den Anfang hatte ich ein Modul erstellt, das einen einzelnen Schrittmotor steuert. Dieses Modul würde den Befehl erhalten, eine bestimmte Anzahl von Schritten (mit einer bestimmten Verzögerung zwischen jedem Schritt) zu machen. Es würde somit die entsprechenden Richtungs- und Schrittsignale für den Schrittmotortreiber erzeugen. Der Code für das Modul ist in **Listing 1** dargestellt.
Der von mir verwendete Stepper-Controller verlangte, dass der Richtungseingang für mindestens 200 ns vor und nach der steigenden Flanke des Schrittsignals stabil ist. Außerdem musste der Schrittimpuls eine Mindestzeit von 1 µs aufweisen (high oder low). Die Schrittimpulsbreite lässt sich leicht mit dem `pulse_extender`-Baustein aus der Komponentenbibliothek realisieren. Dieser Baustein empfängt Einzelimpulse und verlängert sie auf die angegebene Länge. In diesem Fall habe ich die Länge zur Sicherheit auf 2 µs eingestellt.

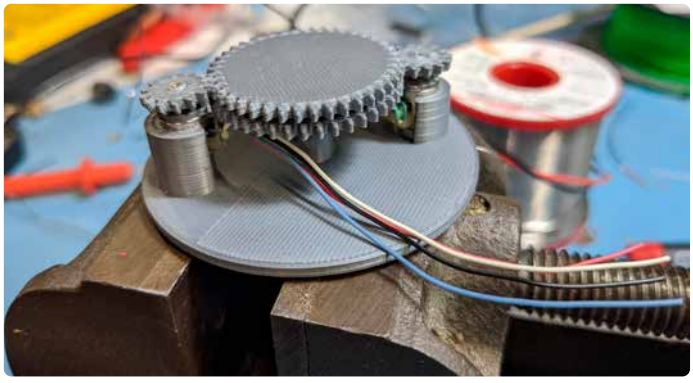


Bild 1. Von Amazon bezogener Miniatur-Schrittmotor mit großen Zahnrädern und zwei kleinen, damit gekoppelten Zahnrädern.

Sobald ein neuer Bewegungs-Befehl empfangen wird, wird der Richtungsausgang gesetzt und das Modul wartet auf einen Überlauf von `dirCt`. Dieser Zähler kann 256 Werte speichern. Bei einem 100-MHz-Takt bedeutet das eine Wartezeit von 2,56 µs. Das ist deutlich länger als die geforderten 200 ns, stellt aber sicher, dass es keine Timing-Probleme mit den langen Kabeln gibt. Eine Straffung des Timings würde hier auch keinen Leistungsunterschied ergeben. `Stepping State` inkrementiert einen Zähler und geht bei jedem Überlauf einen Schritt weiter. Dieser Zähler hat einen Pre-Scaler von 8, so dass jedes Inkrement von `delayCycles` im Animationsbefehl 256 zusätzliche Verzögerungszyklen zur Folge hat. Durch diesen „Vor-Teiler" bleiben `delayCycles` mit 16 Bit relativ klein, er ermöglicht aber trotzdem einen sehr großen Geschwindigkeitsbereich. Selbst mit diesem Pre-Scaler habe ich festgestellt, dass der niedrigste, sichere Wert für `delayCycles` bei etwa 760 liegt. Das entspricht acht Sekunden für eine volle Umdrehung.

Enable Gate

Das nächste vorzustellende Modul ist *Enable Gate*. Es ist dafür verantwortlich, das neue Animationsflag für die Animatoren zu sperren, während die Motoren aktiviert werden. Es sorgt auch dafür, dass die Motoren nach einer Animation lange genug aktiviert bleiben, um ihren letzten Schritt abzuschließen.

Das Modul nimmt die neuen anstehenden Animations-Flags für zwölf verschiedene Motoren auf. Dann aktiviert es die Motoren und wartet eine Weile, 42 ms, bis die Treiber die Motoren wieder mit Strom versorgen und die Motoren zur Ruhe kommen. Nach dieser Zeitspanne wird das Zeichen für anstehende Animationen an die Animatoren weitergegeben.

Während ein beliebiger Animator in der Gruppe läuft, bleiben die Motoren aktiviert. Sobald der letzte Animator fertig ist, bleiben die Motoren für weitere 42 ms aktiviert, bevor sie deaktiviert werden. Der Code für das Modul ist in **Listing 2** dargestellt.

Wenn sich das Modul im Leerlauf befindet, ist `onCtr` gleich null und `offCtr` ist maximal. Wenn eine anstehende Animation erkannt wird (der Fifo-Speicher ist nicht leer), wird der Enable-Ausgang gesetzt und `onCtr` in jedem Zyklus inkrementiert. Sobald `onCtr` gefüllt ist, werden die mit `new_animation` bezeichneten Flags durchgereicht. Wenn alle Animationen ausgeführt wurden, wird `offCtr` inkrementiert. Sobald sein Maximalwert erreicht ist, wird `onCtr` zurückgesetzt, wodurch die Motoren deaktiviert werden.

Interessant ist auch die erste Zeile im always-Block:

```
running = |(animator_busy | ~fifo_empty);
```

Diese Zeile kann ein wenig kryptisch erscheinen, wenn man nicht mit bitweisen Reduktionsoperatoren vertraut ist. Das Ziel dieser Zeile ist es, die zwölf `animator_busy`-Signale und die zwölf `fifo_empty`-Signale in ein einzelnes Bit zu verwandeln.

Denken wir uns zunächst folgenden Einzelfall: Ein beliebiger Motor läuft, wenn der Fifo nicht leer ist oder er gerade beschäftigt ist. Dies kann durch den Ausdruck `animator_busy | ~fifo_empty` bewerkstelligt werden. Eine einzelner senkrechter Strich (|), auch Pipe oder Verkettungszeichen genannt, entspricht einem bit-weisen ODER. Damit werden die Bits der beiden Operanden miteinander mit der ODER-Bedingung verknüpft, wobei die Bitbreite gleich bleibt. Die Tilde (~) entspricht einer bit-weisen Invertierung. Dadurch wird jedes der Bits innerhalb von `fifo_empty` umgedreht.

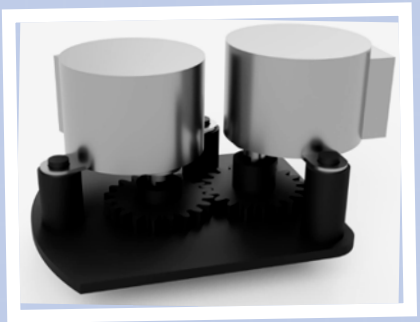
Nach diesen Operationen haben wir nun ein 12 Bit breites Signal, das angibt, wann jeder Animator aktiv ist. Wir müssen diese Information jedoch auf ein einziges Bit verdichten. Hier kommt der OR-Verknüpfungsoperator zum Einsatz (Pipe). Er wird, wenn er vor einem Wert ohne vorangehenden Wert steht, alle Bits im Signal mit der ODER-Bedingung-verknüpfen und ein einzelnes Bit ausgeben. In diesem Fall bedeutet das: Wenn einer der Motoren läuft, wird „running" den Wert 1 besitzen.

Später im Modul verwende ich den AND-Verknüpfungsoperator, um zu prüfen, ob alle Bits in einem Signal 1 sind (auch bekannt als Maximalwert). Dies funktioniert genauso wie beim OR-Operator, mit dem Unterschied, dass alle Bits hier mit der AND-Bedingung (UND) verknüpft werden. Das Ergebnis ist 1, wenn alle Bits den Wert 1 besitzen. Ansonsten ist das Ergebnis 0. Um eine XOR-Ver-

Aufbau der Hardware

Dieses Bild zeigt die beiden Ausgangswellen. Die mittlere, längere Welle ist direkt mit dem Motor verbunden. Die äußere Welle ist mit dem Getriebe gekoppelt, das durch den zweiten, versetzten Motor angetrieben wird. Die Stunden- und Minutenzeiger sind auf diesen beiden Wellen aufgesteckt, aber nicht fest verbunden.

Durch die Reibungskopplung können die Uhrzeiger in eine neutrale Position gebracht werden, bevor die Uhr eingeschaltet wird. Dies ist wichtig, denn auch wenn Schrittmotoren gut für die Steuerung präziser Positionen geeignet sind, gibt es keine Möglichkeit zur Erfassung der Startposition.



Die Motoren wurden einfach auf die Montagezapfen geklebt. Alle Teile wurden auf meinem Prusa MK3S in schwarzem PLA gedruckt.

Der Qwiic-Anschluss am Au ist ein wertvolles und wichtiges Tool zur Kommunikation mit einem Mikroprozessor.

knüpfung (Exklusiv-Oder) durchzuführen, die 1 ergibt, wenn es eine ungerade Anzahl von Einsen gibt, kann das Caret-Zeichen (^) verwendet werden.

Qwiic

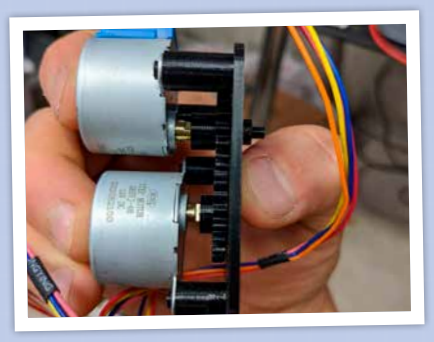
Wir werden uns nun mit dem *Top-Level*-Modul beschäftigen, das sich um die Qwiic-Schnittstelle kümmert und alles verbindet. Betrachten wir dazu **Listing 3**.
Die Qwiic-Schnittstelle wird von dem Modul `i2c_peripheral` behandelt. Dieses Modul ist etwas kompliziert, da Sie bei den Start-Stopp-Signalen vorgeben müssen, wann es Daten annehmen oder senden soll.
Für unseren Fall können wir das Ganze stark vereinfachen, indem wir nur Daten einlesen. Die wichtigen Flags sind dabei `rx_valid`, eine Flag, die uns sagt, dass ein neues Byte eingelesen wurde, und *stop*, die uns mitteilt, dass die I²C-Transaktion gestoppt wurde und wir zurücksetzen sollten. Der Ausgang `rx_data` hat den Wert des eingelesenen Bytes, wenn `rx_valid` den Zustand *high* annimmt.
Soll auf eine Aktion reagiert werden, so müssen die Start-, Next- und Write-Flags überwacht werden. Im nächsten Taktzyklus können Sie *tx_enable* auf 1 setzen und Daten zum Senden an *tx_data* bereitstellen. Dadurch wird das Modul veranlasst, ein Byte zu schreiben, anstatt auf ein Byte zu warten.
Das Startflag signalisiert, dass Ihre ID auf dem Bus erkannt wurde. Gleichzeitig mit dem Setzen dieses Flags zeigt *write* an, ob das letzte Bit im ID-Byte Lesen (0) oder Schreiben (1) anzeigt. Auch dies können wir für diese Version ignorieren.

Das von mir verwendete Transaktions-Protokoll sieht wie folgt aus: Das erste Byte ist die Adresse, gefolgt von den Daten des Befehls. Für die Adressen 0 bis 47 werden vier Bytes erwartet. Die ersten beiden repräsentieren den Schrittzähler und das zweite und dritte Byte stellen den Verzögerungszähler dar. Die Adresse 8hFF ist insofern eine Besonderheit, als sie nur ein Byte erwartet und zum Setzen der LEDs auf dem *Au* verwendet wird. Dies kann nützlich sein, um den Qwiic-Bus zu testen.

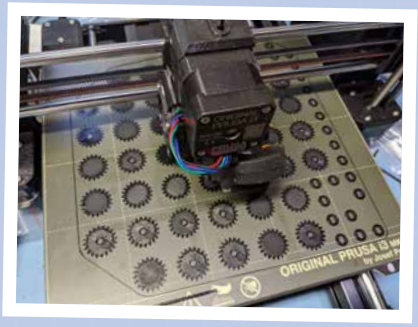
Ich habe dafür gesorgt, dass man nicht für jede Animation die I²C-Transaktion starten oder stoppen muss. Jedes 5-Byte-Paket stellt eine gültige Animation dar, die nach dem Empfang des letzten Bytes eine Schleife bildet. So können Sie allen 48 Motoren eine neue Animation innerhalb einer einzigen Transaktion senden.

FIFOs

Das hier besprochene Projekt enthält 48 FIFO-Buffer, um zusätzliche Animationen zu ermöglichen, während die Animatoren beschäftigt sind. Diese Buffer werden aus der Komponente `fifo` in der Komponentenbibliothek erstellt. Jeder FIFO ist 32 Bit breit und 128 Einträge tief. Die 32 Bits werden in jeweils 16 Abschnitte für die Verzögerung und für die Schrittanzahl aufgeteilt. 128 Einträge sind für die aktuelle Verwendung zwar überdimensioniert, würden aber erlauben, viele kurze Animationen zu stapeln, wenn man andererseits ein Hochlaufen (ramping) und schnellere Bewegungen implementieren wollte. Der Au verfügt ohnehin über ein ausreichendes, eigenes Block-RAM, um all dies unterzubringen.
Der FIFO folgt dem „first-word-fallthrough"-Prinzip, wenn *empty* gleich 0 ist und zeigt damit an, dass Daten vorhanden sind. Der Wert ist bereits auf `dout` verfügbar. Wenn `rget` auf 1 gesetzt wird, wird der Eintrag entfernt und der nächste Eintrag im folgenden Taktzyklus angezeigt. Um den FIFO mit Daten zu versorgen, legen Sie die Daten einfach auf `din` und setzen Sie `wput` auf 1. Sie sollten auch sicherstellen, dass `full` nicht 1 ist, sonst werden Ihre Daten möglicherweise ignoriert.



Dies ist das erste fertige Uhrwerk, das ich gedruckt und montiert habe.



Als das erste Exemplar funktionierte, konnte ich die restlichen Uhrwerke in Serie herstellen.

Alle Teile wurden in einem Karton gesammelt und sind nun bereit für die Montage.

Bit-Zuweisungen

Am Anfang des *always*-Blocks gibt es eine ganze Reihe von Array- beziehunghsweise Bit-Manipulationen. In *Lucid* können Sie Module ganz bequem in Arrays umwandeln und ihre Ports in Arrays packen lassen. In einigen Fällen, wie den Step-Ausgängen, können wir diese Arrays direkt zuweisen, da die Bits perfekt aneinandergereiht sind. In anderen Fällen müssen wir sie in Unterabschnitte aufteilen. Dabei ist es praktisch, for-Schleifen zu verwenden. Denken Sie daran, dass for-Schleifen nicht in Form einer Hardware realisiert werden können und eine feste Anzahl von Iterationen haben müssen, damit sie während der Synthese abgewickelt werden können. Sie stellen einfach nur eine Möglichkeit dar, Dinge kompakter zu schreiben. Zum Beispiel durchläuft die erste for-Schleife vier Iterationen, wobei i Werte von 0 bis 3 annehmen kann. Die erste Zeile wird für die erste Iteration wie folgt ausgewertet:

```
gates.fifo_empty[0] = ani_fifos.empty[0+:12];
```

Der Bitselektor `[0+:12]` bedeutet, bei Bit 0 zu beginnen und 12 Bits darüber auszuwählen. Es werden also die Bits 0-11 ausgewählt. In der nächsten Iteration wird er wie folgt ausgewertet:

```
gates.fifo_empty[1] = ani_fifos.empty[12+:12];
```

Hier bekommt das zweite Freigabegatter die Bits 12-23. Alle vier Iterationen könnten aufgezählt werden:

```
gates.fifo_empty[0] = ani_fifos.empty[0+:12];
gates.fifo_empty[1] = ani_fifos.empty[12+:12];
gates.fifo_empty[2] = ani_fifos.empty[24+:12];
gates.fifo_empty[3] = ani_fifos.empty[36+:12];
```

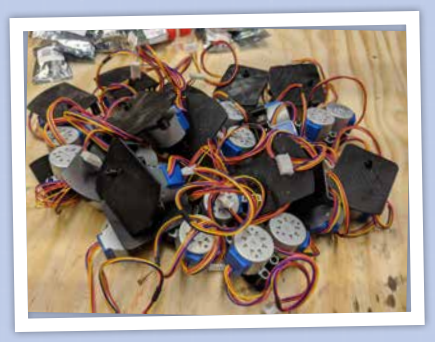
Dies würde eine identische Schaltung im FPGA erzeugen, was jedoch umständlich in der Eingabe und Pflege ist.
Es ist üblich, die Start/Breiten-Bitselektoren in for-Schleifen anstelle der Start/ Stop-Bitselektoren zu verwenden. Das liegt daran, dass Sie die Start/ Stopp-Selektoren nicht mit nicht-konstanten Werten verwenden können.
Der oben verwendete Start/Breiten-Selektor sorgt dafür, dass die Auswahl immer 12 Bit breit ist. Ein Signal, das die Breite ändert, können Sie nicht in Form von Hardware realisieren, da Sie nicht spontan irgendwelche Verbindungen herstellen oder entfernen können. In diesem Fall habe ich die obere Variante des Selektors verwendet, indem ich das „`+:`“ genommen habe. Sie können auch ein „`-:`“ einsetzen, um die Abwärtsvariante des Selektors zu verwenden. Diese wählt das Startbit und die Bits darunter aus. Zum Beispiel ist `[11-:12]` das gleiche wie `[0+:12]`. Sie wählen beide die Bits 0-11 aus.

Pin-Zuweisungen

Doch wie werden die `step`- und `dir`-Signale auf die IO-Pins beim *Au* abgebildet? Dieses Mapping wird in einer Constraint-Datei definiert. In diesem Fall befinden sie sich in der Datei *clockclock.acf*. Die Erweiterung „acf“ steht für *Alchitry Constraint File*. Dieses Format ist sehr einfach und erlaubt es Ihnen, die Pin-Namen als Pins anstelle des FPGAs auf den Alchitry-Boards anzugeben. Zum Beispiel entspricht A2 dem zweiten Pin der oberen linken Stiftleiste (Bank A) auf dem *Au*-Board. Wenn Sie diese Datei öffnen, erscheint eine ganze Reihe von Zeilen, die den beiden hier gezeigten ähneln:

```
pin step[0] A2;
pin dir[0] A3;
```

Jeder IO-Port muss auf einen physikalischen Pin abgebildet werden. Das Format ist das Pin-Schlüsselwort, gefolgt von dem Signalnamen und schließlich der physikalischen Pin-Position. Sie können auch das Schlüsselwort `pullup` oder `pulldown` hinzufügen, um dem Pin einen internen Pullup- oder Pulldown-Widerstand hinzuzufügen. Bei der Control-Unit wird `pulldown` jedoch ignoriert, da das *Lattice*-FPGA keine internen Pulldown-Widerstände besitzt. Die meisten Pins auf einem FPGA sind vollständig austauschbar und die Pinbelegung, die ich für die Uhr verwendet habe, war mit Ausnahme der Qwiic-Signale willkürlich.



24 vormontierte Laufwerke warten auf ihren Einbau.

Der nächste Schritt war die Erstellung des Rahmens. Ich habe ihn aus zwei Brettern aus Ahorn gefertigt. Das erste Brett habe ich in drei dünnere Stücke zersägt…

… die ich zusammenkleben konnte, um die Front zu machen.

Arduino + FPGA = ein nützliches Duo für viele Projekte

Software-Setup und Programmierung

Hinweis: Dieses Beispiel setzt voraus, dass Sie die neueste Version der Arduino-IDE auf Ihrem Desktop verwenden. Wenn Sie die Arduino-IDE zum ersten Mal benutzen, lesen Sie bitte unser Tutorial zur Installation der Arduino IDE [14]. Wenn Sie noch keine Arduino-Bibliothek installiert haben, lesen Sie bitte unsere Installationsanleitung [15].

Ich habe den RedBoard Turbo als Mikrocontroller verwendet, aber wie ich bereits sagte, könnte wahrscheinlich jedes Board mit einem Qwiic-Anschluss genutzt werden. Zuerst musste ich die Bibliotheken für das Board, die Qwiic RV8803 RTC und die Taster installieren. Der Weblinks-Kasten führt Sie zu den jeweiligen Setup-Tutorials für beide.

Der Code selbst ist nicht allzu kompliziert, und die Struktur nutze ich bei vielen Projekten. Ich habe grundsätzlich so lange Abstraktions-Schichten aufgebaut, bis ich eine einfach zu bedienende Schicht erhielt, die die Verwaltung der Zahlen und das Ausführen von Animationen einfach machte.

Die erste Schicht ist natürlich das FPGA. Das FPGA gibt uns eine Schnittstelle, um einen Motor eine bestimmte Anzahl von Schritten mit einer festen Verzögerung zwischen jedem einzelnen Schritt ausführen zu lassen. Um den I²C-Bus zu verwalten, habe ich die im Arduino eingebaute Wire-Bibliothek verwendet. Dies erlaubte mir, eine einfache Funktion zu erstellen, die eine einzelne Animation sendet - hier ist sie:

```
void sendAnimation(uint8_t id, int16_t steps, uint16_t delay_cycles) {
  int32_t p = currentPosition[id] + steps;
  while (p < 0) p += FULL_CIRCLE;
  currentPosition[id] = p % FULL_CIRCLE;
  Wire.write(id);
  Wire.write((uint8_t)(steps >> 8));
  Wire.write((uint8_t)(steps & 0xFF));
  Wire.write((uint8_t)(delay_cycles >> 8));
  Wire.write((uint8_t)(delay_cycles & 0xFF));
}
```

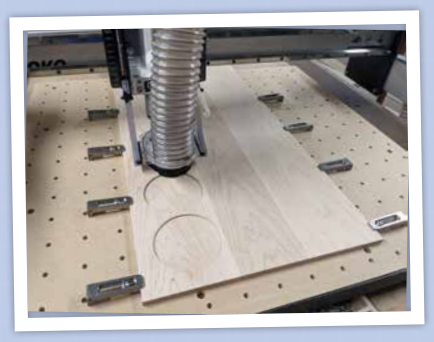
Eine Besonderheit ist hier, dass ich eine Konstante namens `FULL_CIRCLE` deklariert habe, die die Anzahl der Schritte in einer vollen Umdrehung angibt. Diese ist in meinem Fall die Zahl 4096, die dafür sorgt, ein globales Array mit den Positionswerten der Motoren zu aktualisieren. Dadurch, dass man die Animationen immer mit dieser Funktion sendet, ist die Position der Motoren bekannt.

Es ist nicht sehr bequem, sich die Bewegung in Form von Schritten und Verzögerungen vorzustellen. Viel einfacher ist es, sie in Form von Grad und Dauer zu visualisieren. Anstatt also zu sagen: „Mache 2048 Schritte mit 760 Verzögerungszyklen zwischen jedem Schritt" ist es viel sinnvoller, sich vorzustellen: „Drehe dich innerhalb von 4 Sekunden um 180 Grad." Also habe ich eine Funktion geschrieben, die sich um diese Übersetzung kümmert:

```
void animate(uint8_t id, float deg, float duration) {
  float steps = deg * FULL_CIRCLE / 360.0f;
  if (steps > 32767 || steps < -32768) {
    animate(id, deg / 2, duration / 2);
    animate(id, deg / 2, duration / 2);
    return;
  }

  float cycles = constrain(duration * 390625 / abs(steps), 760, 65535);

  sendAnimation(id, (int16_t)steps, (uint16_t)cycles);
}
```

Nun wurde meine CNC-Fräse eingesetzt, um das Profil zu glätten und auszuschneiden.

Da die Uhr ziemlich groß ist, musste ich alle Arbeiten in zwei Schritten ausführen. Sobald ich die erste Seite fertig hatte, konnte ich sie umdrehen und die andere Seite glätten, um das gesamte Zifferblatt perfekt flach zu bekommen.



Die Platte war nun bereit, um die Vertiefungen für die Uhren auszufräsen.

Zunächst werden die Gradzahlen mit `FULL_CIRCLE` in Schritte verwandelt. Dann prüft die Funktion, ob es zu viele Schritte für einen einzelnen Animationsbefehl gibt und ruft sich selbst rekursiv mit jeweils der Hälfte der Animation auf.

Anschließend werden die Verzögerungszyklen berechnet. Der Wert 390625 gibt die Anzahl der Zyklen in einer Sekunde (100.000.000/256 = 390.625) an. Die Zyklen müssen auf den Bereich 760…65535 begrenzt werden. Das Minimum von 760 wurde empirisch ermittelt, indem getestet wurde, wie schnell die Motoren sich zuverlässig drehen können, ohne einen Schritt zu überspringen. Der ermittelte Wert beträgt 8 Sekunden pro Umdrehung.
Die obere Grenze ist für die Praxis recht untauglich: Es würde 687 Sekunden dauern, um eine volle Umdrehung durchzuführen. Das vorliegende Projekt kann nicht langsamer sein als diese Geschwindigkeit. Bei Bedarf müssten Sie daher den Pre-Scaler des FPGAs oder die Größe des Verzögerungszykluswertes ändern.
Die letzte Abstraktion, die ich brauchte, war eine Funktion, die dem Motor sagt, dass er eine bestimmte Position anfahren soll. Diese verwendet den Wert `currentPosition`, um die minimale Drehung zum Erreichen dieses Punktes zu berechnen. Dies ist hilfreich, wenn die aktuelle Zeit angezeigt wird: Ich kann die Funktion einfach mit allen Positionen aufrufen, in denen die Zeiger sein müssen, und muss nicht anpassen, wo sie sich gerade befinden.

```
void moveTo(uint8_t id, float pos, float duration) {
  float curDeg = (float)currentPosition[id] * 360.0f /
FULL_CIRCLE;
  float angle = pos - curDeg ;

  if (angle > 180)
    angle = angle - 360;
  if (angle < -180)
    angle = 360 + angle;

  animate(id, angle, duration);
}
```

Das Ganze beginnt mit der Berechnung des Winkels, in dem sich der Zeiger gerade befindet. Der Winkel der auszuführenden Bewegung ergibt sich, indem der gewünschte Winkel vom aktuellen Winkel subtrahiert wird. Die beiden if-Anweisungen prüfen das Ergebnis und schalten auf den kleineren der beiden möglichen Wege um. Wenn die Differenz der Winkel zum Beispiel +270° beträgt, ist es besser, den Zeiger stattdessen um -90° zu bewegen.
Um die aktuelle Zeit anzuzeigen, brauchte ich eine Tabelle für alle Ziffern. Dazu zeichnete ich einfach auf, wie jede Zahl aussehen sollte und fügte die betreffenden Winkel hinzu. Das alles habe ich in ein 2D-Array (**Listing 4**) eingetragen, das zum Nachschlagen der Ziffern dient. Zum Ermitteln der Zeit und zum Darstellen der Ziffern wird die RTC verwendet.

```
void showTime() {
  uint8_t digits[4];
  digits[0] = rtc.getMinutes() % 10;
  digits[1] = rtc.getMinutes() / 10;
  digits[2] = rtc.getHours() % 10;
  digits[3] = rtc.getHours() / 10;
  for (uint8_t d = 0; d < 4; d++) {
    Wire.beginTransmission(0x50);
    for (uint8_t m = 0; m < 12; m++) {
      moveTo(m + 12 * d, digitAngles[digits[d]][m], 4.0f);
    }
    Wire.endTransmission();
  }
}
```

Der Code geht davon aus, dass 0-Grad-Marke für alle Zeigerpositionen bei 12:00 (senkrecht nach oben stehend) beginnt. In der Funktion `loop()` des Arduino habe ich einen Code eingefügt, der die RTC alle 100 ms überprüft und die Zeit aktualisiert, wenn sie sich ändert. Wenn sich der Wert für die Stunde änderte, führt dieser Code ebenfalls eine Animation aus. Ich habe drei dieser einfachen Animationen geschrieben, die vor einer erneuten Anzeige der Zeit zufällig ausgewählt werden.

Die Frontplatte geriet leider etwas dünner, als ich ursprünglich geplant hatte. Das führte dazu, dass die Böden der runden Vertiefungen nach dem Fräsen nur 1,5 mm dick wurden. Zum Glück trotzdem noch ausreichend stark.

Dann baute ich einen Rahmen und klebte ihn zusammen…

… um auf diese Weise einen stabilen Holzkasten zu erhalten.

Innerhalb von `loop()` prüfe ich auch den Zustand der vier Tasten. Mein ursprünglicher Plan war es, die FIFO-Schnittstelle des Qwiic-Buttons zu verwenden, um jeden Tastendruck zu überwachen, aber ich bin auf einen Fehler gestoßen und habe stattdessen die Funktion `isPressed()` verwendet, um den aktuellen Status zu überprüfen.

Wenn eine einzelne Taste gedrückt wird, aktualisiere ich die Zeit. Die vier Knöpfe erlauben es mir, die Minuten und Stunden unabhängig voneinander einzustellen. Ich habe auch eine Funktion hinzugefügt, bei der sich alle Zeiger auf die 12:00-Position bewegen, wenn Sie gleichzeitig die Tasten *Hour Up* und *Hour Down* drücken. Dies ist sehr hilfreich, wenn Sie die Uhr ausschalten oder den Arduino neu programmieren müssen, da Sie dann nicht jeden Zeiger manuell neu einstellen müssen. Soviel zu meiner Entwicklung: Man nehme eine einfache Schnittstelle und erweitere sie so lange, bis etwas Nützliches dabei heraus kommt.

Fazit

Dieses Projekt machte wesentlich mehr Arbeit, als ich ursprünglich dachte. Die meiste Zeit habe ich mit dem mechanischen Aufbau und der Verdrahtung verbracht. Es dauerte auch eine Weile, bis ich eine funktionierende Lösung für die beweglichen Teile fand. Die FPGA- und Arduino-Entwicklungen vollzogen sich dagegen etwas einfacher.

Dem Qwiic-Anschluss am Au, der zur Kommunikation mit einem Mikrocontroller dient, kommt offensichtlich eine besondere Bedeutung zu: Ich war angenehm überrascht, wie einfach die Einrichtung auf der Arduino-Seite war, da ich zuvor noch nie Qwiic (oder I²C) an einem Arduino verwendet hatte. Es gibt allerdings noch ein paar Dinge, die für zukünftige, in diese Richtung weisende Entwicklungen noch verbessert werden könnten:

- Die Uhr ist ziemlich laut. Die einzelnen Motoren sind zwar ziemlich leise, aber das Holz, auf dem sie montiert sind, verstärkt ihre Geräusche, so dass irgendeine Art von Schalldämmung im Hinblick auf deren Befestigung vonnöten wäre, vielleicht eine Art weicher, gummiartiger Kleber anstelle von Sekundenkleber. Das Holz ist auf dem größten Teil der Fläche auch nur 1,5 mm dick, so dass sehr leicht Vibrationen entstehen können. Ich vermute, hier könnte ein Bekleben der Rückseite mit Dämmmaterial Abhilfe schaffen.
- Das andere große Problem ist, dass die Schrittmotoren interne Getriebe besitzen und diese ein gewisses Spiel aufweisen. Das bedeutet, dass die Zeiger sich je nach Bewegungsrichtung nicht genau dort befinden, wo sie sein sollen. Es scheint, dass sie beim Umschalten der Richtung aufgrund des Spiels ein paar Grad daneben liegen. Kein Weltuntergang, aber die Abweichung ist gerade so groß, dass sie unangenehm auffällt. Vielleicht kann ich einen Code schreiben, der das kompensiert, aber der Fehler ist bei jedem Motor ein wenig anders und die Korrektur müsste fein abgestimmt werden. Das Problem könnte am einfachsten mit getriebelosen Motoren behoben werden. Ich hatte allerdings Schwierigkeiten, kleine und kostengünstige Schrittmotoren dafür zu finden.

Ich hoffe, dieses Demo-Projekt ist ein gutes Beispiel für eine praktische FPGA-Anwendung. Vielleicht wird es Sie sogar zu neuen eigenen Projekten beflügeln!

200676-02

Elektor-Projektseite:
www.elektormagazine.de/esfe-en-clockclock

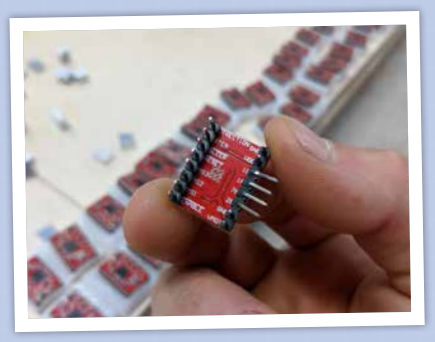
Als nächstes musste ich die Zeiger anfertigen. Ich entschied mich für Padouk, ein schönes rotes Holz, das mit der Zeit zu einem tiefen Rotbraun wird. Dies würde gut zu den Padouk-Schubladengriffen in meiner Küche passen, wo die Uhr ihren Platz finden soll. Die Zeiger stammen aus einer gemeinsamen Platte, wurden auf eine Dicke von 2 mm herunter gefräst ...

... und können sich sehen lassen.

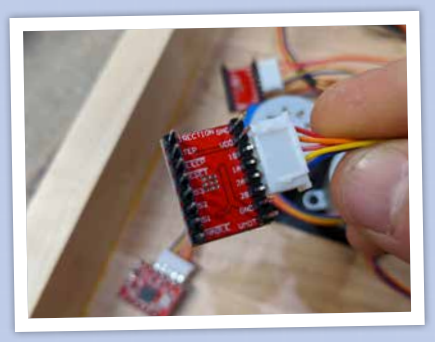
Die einzelnen Uhrwerke wurden von hinten an die Frontplatte geklebt. Vier von ihnen sind in abweichenden Winkeln angeordnet, um Platz für die Stromversorgung zu schaffen, ein 12-V-6-A-Netzteil, das für alle Motoren ausreicht.

Listing 1.

```
module animator (
    input clk,  // clock
    input rst,  // reset
    signed input stepCount[16], // it can be negative to indicate direction
    input delayCycles[16],      // cycles between each step
    input newAnimation,         // flag for new animation
    output busy,                // flag the animator is busy and won't accept animations
    output step,                // step signal for the driver
    output direction            // direction signal for the driver
  ) {

  .clk(clk) {
    // The driver requires each "step" pulse to be at least 1us so we make them 2us
    pulse_extender stepExt(#MIN_PULSE_TIME(2000));

    dff dirCt[8];      // counter for waiting after changing direction. 200ns delay required
    dff counter[16+8]; // counter for delaying between steps. The +8 is the pre-divider

    .rst(rst) {
      fsm state = ;
      dff dir;         // saved direction of the motor
      dff delayCt[16]; // saved delay count
      dff steps[16];   // saved number of steps (absolute value)
    }
  }

  always {
    busy = state.q != state.IDLE; // busy when not idle
    step = stepExt.out;           // step output is the extended pulse

    stepExt.in = 0;               // default to no new step
    direction = dir.q;            // output the saved direction

    case (state.q) {
      state.IDLE:
        if (newAnimation && stepCount != 0) {   // if new animation with steps (skip 0 step animations)
          state.d = state.DIR_WAIT;             // move to next state
          dir.d = stepCount[stepCount.WIDTH-1]; // direction is the sign of the step input (0 = positive, 1
= negative)
```



Ich habe generische A4988-Schrittmachertreiber (www.amazon.com/gp/product/B01FFGAKK8) zur Steuerung der Motoren verwendet, deren Motorstifte sich leicht verbiegen ließen ...



... um die Motoren direkt dort einstecken zu können, nachdem zwei der Drähte am Stecker des Motors vertauscht wurden.

Nach dem alles auf die geschilderte Weise verdrahtet war, konnte ich die Stromversorgung anschließen. Auf diesem Bild fehlen noch zwei bis zu diesem Zeitpunkt nicht gelieferte Motoren.

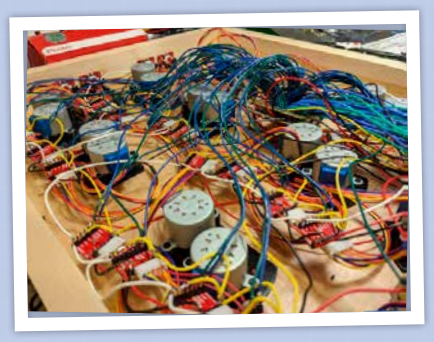
```
          steps.d = stepCount[stepCount.WIDTH-1] ? -stepCount : stepCount; // save the absolute value of
stepCount
          delayCt.d = delayCycles;              // save the number of delay cycles
        }
      state.DIR_WAIT:
        dirCt.d = dirCt.q + 1;          // wait for the direction output after it changed
        if (&dirCt.q) {                 // if done waiting
          state.d = state.STEP;         // move to stepping state
        }
      state.STEP:
        counter.d = counter.q + 1;      // increment step delay counter
        if (counter.q[counter.WIDTH-1-:16] == delayCt.q) { // if counter has reached the delay count
          counter.d = 0;                // reset counter
          stepExt.in = 1;               // send a pulse
          steps.d = steps.q - 1;        // decrement the number of steps remaining
          if (steps.q == 1) {           // if no more are left
            state.d = state.IDLE;       // return to idle
          }
        }
    }
  }
}
```

Passende Produkte

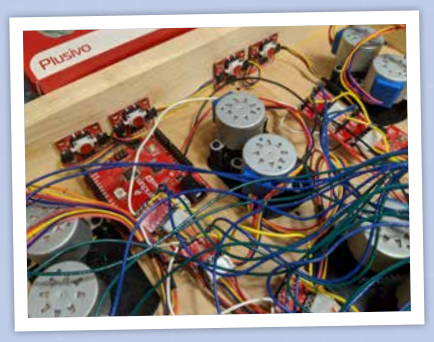
Sie suchen die in diesem Artikel erwähnten Produkte? SparkFun und Elektor halten sie für Sie bereit!

- (2) Anschlusskabelsortiment (starr, 22 AWG); PRT-11367
- (2) Qwiic-Kable - 100 mm; PRT-14427
- (3) Qwiic-Kable - 50 mm; PRT-14426
- SparkFun Real-Time-Clock-Modul - RV-8803 (Qwiic); BOB-16281
- (4) SparkFun Qwiic-Button - Red LED; BOB-15932
- Alchitry Br Prototyp-Element-Board; DEV-16524
- Alchitry Au FPGA-Entwicklungsboard (Xilinx Artix 7); DEV-16527
- SparkFun RedBoard-Turbo - SAMD21-Entwicklungsboard; DEV-14812

www.elektormagazine.de/esfe-en-ClockClock1



Jeder Treiber musste mit der 12-V-Versorgung für die Motoren und mit 3,3 V für die Steuerlogik verdrahtet werden. Bei jeder Sechser-Gruppe, die eine Ziffer bildet, sind auch die Enable-Pins zusammen verdrahtet und mit der Alchitry Au verbunden. Jeder Treiber erzeugt ein Richtungs- und Schrittsignal, das ebenfalls mit der Alchitry Au verdrahtet werden musste. Wie Sie auf dem Bild sehen können, wurde die Verdrahtung schnell zu einem riesigen Kabel-Gewirr.



Um die Zeit einzustellen, habe ich vier einfach zu verkabelnde Qwiic-Tasten hinzugefügt. Das obere Paar ist für die Stundeneinstellung und das untere Paar für die Minuteneinstellung.
Hinter den Tastern befindet sich eine RV-8803 Echtzeituhr, die ebenfalls an den Qwiic-Bus angeschlossen ist. Da die RTC batteriegepuffert ist, musste ich die Zeit nicht jedesmal neu einstellen, wenn ich das Board umprogrammiert oder ausgesteckt habe.
Letztendlich wird die Alchitry Au angeschlossen. Sie muss sich am Ende der Kette befinden, da sie nur einen Qwiic-Stecker hat. Ich habe auch die

Listing 2.

```
module enable_gate (
    input clk,  // clock
    input rst,  // reset
    output new_animation[12], // output to the animators
    input fifo_empty[12],     // input from fifos (pending animations)
    input animator_busy[12],  // input from animators
    output enable             // output to the stepper drivers
  ) {

  .clk(clk) {
    .rst(rst) {
      dff onCtr[22];   // counter to ensure motors are fully on (22bits ~ 42ms)
      dff offCtr[22];  // counter to keep motors on after animators finish (22bits ~ 42ms)
    }
  }

  sig running; // value used to know when the motors should be on

  always {
    // run when we have pending animations or are actively running
    running = |(animator_busy | ~fifo_empty);

    // enable flag is set when running or onCtr isn't 0 (it is reset after offCtr overflows)
    enable = running || (onCtr.q != 0);

    // pass on new_animation flag only when onCtr is full
    new_animation = ~fifo_empty & 12x{&onCtr.q};

    if (running) {
      offCtr.d = 0;            // reset off counter
      if (!&onCtr.q) {         // if not full
        onCtr.d = onCtr.q + 1; // increment onCtr
      }
    } else {                     // not running
      if (!&offCtr.q) {          // if offCtr not full
        offCtr.d = offCtr.q + 1; // increment offCtr
      } else {                   // if offCtr is full
        onCtr.d = 0;             // reset the onCtr
      }
    }
  }
}
```

Stromversorgungsleitung vom Qwiic-Kabel entfernt, damit der 3,3-V-Regler auf der Alchitry Au nicht mit dem 3,3-V-Regler auf dem RedBoard Turbo in Konflikt gerät. Beim Mikrocontroller handelt es sich um den RedBoard Turbo. Ich habe ihn einfach deshalb verwendet, weil er einen Qwiic-Anschluss hat und ich ihn gerade zur Hand hatte. Prinzipiell kann jeder Mikrocontroller mit einem Qwiic-Anschluss verwendet werden. Die benötigte Rechenleistung ist minimal. Die Alchitry Au und das RedBoard Turbo benötigen beide 5 V. Ich habe einen kleinen Regler für ferngesteuerte Fahrzeuge verwendet, der die 12 V auf 5 V bei bis zu 2 A herunter setzt. Ich habe die Schrittmotor-Treiber so verdrahtet, dass sie im Half-Stepping-Betrieb arbeiten. Ursprünglich hatte ich sie auf 1/16-Mikroschritt eingestellt, aber die Motoren erzeugten ein störendes Heulgeräusch, wenn sie nicht im Halb- oder Vollschritt liefen. Außerdem brauchte ich diese Auflösung auch nicht wirklich.

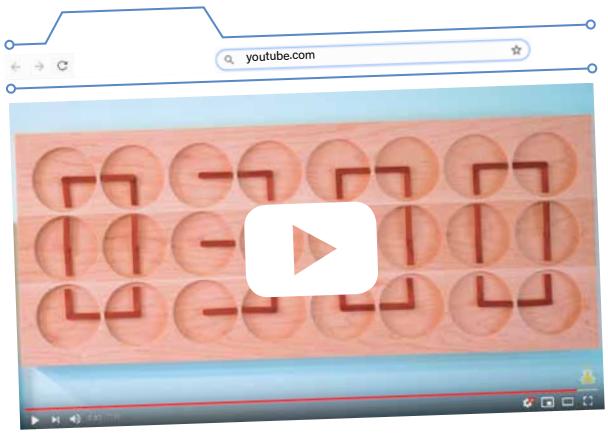
Bei den 102 Drähten, die in zum FPGA führen, ist es egal, an welche Pins sie angeschlossen werden. Es ist nur wichtig, dass Sie den Überblick behalten, denn die tatsächliche Pinbelegung wird in der Constraint-Datei des FPGA-Designs definiert

Nach Abschluss der Verkabelung konnte ich das Ganze noch ein wenig ordnen und übersichtlicher gestalten.

YouTube Time!

Würden Sie gerne Justins ClockClock-Prototyp in Aktion zu sehen? Besuchen Sie:

https://youtu.be/rNJlQ4Fa9mQ



Anzeige

... und die Zeiger auf die Achsen der Motoren stecken.

Das war's im Grunde mit dem mechanischen Aufbau. Die Verdrahtung war zwar nicht kompliziert, aber dafür sehr mühsam.

Listing 3.

```
module au_top (
    input clk,          // 100MHz clock
    input rst_n,        // reset button (active low)
    output led [8],     // 8 user controllable LEDs
    input usb_rx,       // USB->Serial input
    output usb_tx,      // USB->Serial output
    output step[48],    // step output to motors
    output dir[48],     // direction output to motors
    output enable[4],   // enable output to motors (one per digit)
    inout sda,          // Qwiic SDA
    input scl           // Qwiic SCL
  ) {

  sig rst;              // reset signal

  .clk(clk) {
    // The reset conditioner is used to synchronize the reset signal to the FPGA
    // clock. This ensures the entire FPGA comes out of reset at the same time.
    reset_conditioner reset_cond;

    dff ani_id[6];            // saved ID for the motor
    signed dff ani_steps[16]; // saved number of steps
    dff ani_delay[16];        // saved delay counts
    dff byteCt;               // byte flag for 16bit numbers

    .rst(rst) {
      i2c_peripheral qwiic (.sda(sda), .scl(scl)); // i2c peripheral module for qwiic interface

      dff ledReg[8]; // reg to hold the LED values (useful for qwiic testing)

      fsm state = ;

      animator animators[48]; // need 48 individual animators (one per motor)

      // need one fifo per animator, 32 bits wide for 16 bit steps and 16 bit delay
      // the 128 depth is definitely overkill and 16 would probably be plenty for the
      // current usage.
      fifo ani_fifos[48] (#SIZE(32), #DEPTH(128));

      enable_gate gates[4]; // modules to control the enable signals (one per digit)
    }
  }

  var i;

  always {
    reset_cond.in = ~rst_n; // input raw inverted reset signal
    rst = reset_cond.out;   // conditioned reset

    led = ledReg.q;         // output ledReg to the leds

    usb_tx = usb_rx;        // echo the serial data

    // the ~ here flips every motor direction so positive steps would go clock-wise
    // the 48hAAAAAAAAAAAA constant has every other bit flipped so the geared motors
    // and direct drive motors will turn the hands the same way
    dir = animators.direction ^ ~48hAAAAAAAAAAAA;
    step = animators.step;
    enable = ~gates.enable; // enable of the controllers is active low so invert the bits

    qwiic.tx_data = 8bx; // this design is "write only" and never sends data to the microcontroller
    qwiic.tx_enable = 0; // never send data
```

```
// combined groups of 12 motors for the four enable gates
for (i = 0; i < 4; i++) {
  gates.fifo_empty[i] = ani_fifos.empty[i*12+:12];
  gates.animator_busy[i] = animators.busy[i*12+:12];
  animators.newAnimation[i*12+:12] = gates.new_animation[i];
  // only remove a value from the fifo when the gate passes the new_animation flag
  // and the animator isn't busy
  ani_fifos.rget[i*12+:12] = gates.new_animation[i] & ~animators.busy[i*12+:12];
}

// for each motor split the fifo output to the animator signals
for (i = 0; i < 48; i++) {
  animators.stepCount[i] = ani_fifos.dout[i][15:0];
  animators.delayCycles[i] = ani_fifos.dout[i][31:16];
}

// default to no new animations
ani_fifos.wput = 48b0;

// always input the saved delay and steps
// this line takes the two values, joins them, packs them into a 1x32 array,
// and finally duplicates it 48 times into a 48x32 array
// essentially, it just feeds the same 32 bits to each of the 48 fifos
ani_fifos.din = 48x{}};

case (state.q) {
  state.IDLE:
    byteCt.d = 0;
    if (qwiic.rx_valid) {          // new data
      case (qwiic.rx_data) {       // case on the value
        8hFF: state.d = state.LED; // make "address" FF the LEDs for testing
        default:
          ani_id.d = qwiic.rx_data[5:0];  // default to "address" as the motor id
          state.d = state.ANIMATION_STEPS;
      }
    }
  state.LED:
    if (qwiic.rx_valid) {        // if new data
      state.d = state.IDLE;      // return to idle
      ledReg.d = qwiic.rx_data;  // show value on the LEDs
    }
  state.ANIMATION_STEPS:
    if (qwiic.rx_valid) {                // if new data
      ani_steps.d = c;  // save byte and shift old byte
      byteCt.d = ~byteCt.q;              // flip byte counter
      if (byteCt.q == 1) {               // if second byte
        state.d = state.ANIMATION_DELAY;  // go to delay capture state
      }
    }
  state.ANIMATION_DELAY:
    if (qwiic.rx_valid) {                // if new data
      ani_delay.d = c;  // save byte and shift old byte
      byteCt.d = ~byteCt.q;              // flip byte counter
      if (byteCt.q == 1) {               // if second byte
        state.d = state.ANIMATION_PUT;    // go to put state
      }
    }
  state.ANIMATION_PUT:
    state.d = state.IDLE;                // return to idle
    ani_fifos.wput[ani_id.q] = 1;        // put the new animation into the correct fifo
}

if (qwiic.stop) {          // if I2C stop condition is detected
  state.d = state.IDLE;    // reset to IDLE
} } }
```

Listing 4.

```
const float digitAngles[10][12] = {{270, 180, 0, 180, 270, 0, 90, 180, 0, 180, 90, 0}, // 0
  {180, 180, 0, 180, 0, 0, 225, 225, 225, 225, 225, 225}, // 1
  {270, 180, 270, 0, 270, 270, 90, 90, 90, 180, 90, 0}, // 2
  {270, 180, 0, 180, 270, 0, 90, 90, 90, 90, 90, 90}, // 3
  {180, 180, 0, 180, 0, 0, 180, 180, 90, 0, 225, 225}, // 4
  {270, 270, 270, 180, 270, 0, 90, 180, 90, 0, 90, 90}, // 5
  {270, 270, 270, 180, 270, 0, 90, 180, 0, 180, 90, 0}, // 6
  {270, 180, 0, 180, 0, 0, 90, 90, 225, 225, 225, 225}, // 7
  {270, 180, 270, 0, 0, 270, 90, 180, 90, 0, 0, 90}, // 8
  {270, 180, 0, 180, 0, 0, 90, 180, 90, 0, 225, 225} // 9
};
```

WEBLINKS

[1] ClockClock von Humans seit 1982: https://clockclock.com
[2] Alchitry Au IO-Pins: https://www.elektormagazine.de/esfe-en-ClockClock1
[3] Getriebe-Schrittmotor : https://www.amazon.com/gp/product/B01J3KV3B2
[4] StepStick Schrittmotor-Treibermodulmit Kühlkörper: https://www.amazon.com/gp/product/B01FFGAKK8
[5] Einstellbare BEC UBEC 2-6S für Quadcopter-RC-Drohnen: https://www.amazon.com/gp/product/B07PLSYX9G
[6] Gerschlossenes AC-DC-Schaltnetzteil : https://www.amazon.com/gp/product/B07Z55FCQQ
[7] Programming an FPGA: https://bit.ly/3nplgif
[8] How Does an FPGA Work? : https://bit.ly/3r42FdG
[9] Getting Fancy with PWM: https://bit.ly/2IUDmcZ
[10] CAD-Datei (Fusion 360) : https://bit.ly/3mpXSjo
[11] ClockClock-Alchitry-Datei (FPGA) : https://bit.ly/2K90HrW
[12] ClockClock-Arduino-Code (zip) : https://bit.ly/3moeXdv
[13] 28BYJ-48 Schrittmotoren: https://www.amazon.com/gp/product/B01J3KV3B2
[14] Arduino-IDE-Installation: https://bit.ly/2Lz6OWM
[15] Installation einer Arduino-Bibliothek: https://bit.ly/37owopV
[16] Learning FPGAs: Digital Design for Beginners with Mojo and Lucid HDL: https://amzn.to/38aexlN

Fehlersuche

Alchitry

Brauchen Sie Hilfe? Wenn Ihr Produkt nicht wie erwartet funktioniert oder Sie technische Unterstützung oder Informationen benötigen, besuchen Sie die Alchitry-Foren unter: https://forum.alchitry.com.
Hier können Sie eine erste Fehlersuche durchführen und Hilfe erhalten.

Ressourcen und weiterführende Informationen

1. Produktionsdateien:
   - CAD File (Fusion 360) [10]
   - Alchitry (FPGA) [11]
   - Arduino Code (ZIP) [12]

2. Auf der Website von Alchitry (https://alchitry.com) finden Sie weitere interessante Informationen wie zum Beispiel Tutorials, Projekte, das Alchitry-Forum, den Alchitry-Au-Schaltplan (PDF), den Alchitry-Cu-Schaltplan (PDF) und das *Xilinx Artix 7 User Guide*.

3. Wenn Sie tiefer in die Welt der FPGAs und in Lucid eintauchen möchten, lesen Sie *Learning FPGAs: Digital Design for Beginners with Mojo and Lucid HDL* von Justin Rajewski [16]. Es ist bei Amazon erhältlich und eine großartige Informationsquelle, um FPGAs zu verstehen und schließlich selbst zu programmieren.

Unter der Lupe: SparkFun Inventor's Kit

Von Luc Lemmens (Elektor)

Und noch ein Arduino-Starterkit? Oder erweist sich das SparkFun Inventor's Kit (SIK) als genau das Richtige, wenn man in die Elektronik und Mikrocontroller-Programmierung einsteigen möchte?

Ein Anfänger, der in die Elektronik und Programmierung einsteigen will, sieht bei der Suche nach geeignetem Material den Wald vor lauter Bäumen nicht. Natürlich benötigt er ein Prozessor-Board, und für einen Einsteiger ist der Arduino Uno sicherlich eine der nächstliegenden und günstigsten Optionen: Einfach, billig, und im Internet finden sich Unmengen an Arduino-Software und Tutorials.

Abgesehen von der Controllerplatine benötigen Sie einige zusätzliche elektronische Bauteile wie Schalter, LEDs und Sensoren, damit die Platine mit der Außenwelt interagieren kann. Und wenn Sie etwas im Bereich der Robotik machen wollen, macht die Auswahl der benötigten Motoren und der anderen (elektro-) mechanischen Teile die Sache nicht einfacher. Die Suche kann zwar eingegrenzt werden, indem man nach einem kompletten Kit sucht, das „alles Wesentliche" enthält, aber bei der großen Menge Arduino-Starter-Kits im Angebot fragt man sich, welches das beste Preis-Leistungs-Verhältnis bietet? Lassen Sie uns sehen, was das Inventor's Kit von SparkFun zu bieten hat und warum ich letztendlich dieses Kit empfehle.

Was ist das SIK?

SparkFun behauptet auf seiner Website [1]: „Das SparkFun Inventor's Kit (SIK) ist eine großartige Möglichkeit, um mit der Programmierung und Hardware-Interaktion mit der Arduino-Programmiersprache zu beginnen". Die Aussage ist in jeder Hinsicht richtig, denn alles, was Sie brauchen, ist bis hin zu einem Schraubendreher in diesem Kit enthalten. Nur die Software müssen Sie selbst herunterladen [2] und auch vier Mignon-Batterien besorgen. Das SIK ist schön verpackt in einem stabilen Kunststoffkoffer, so dass der Inhalt samt Projekten/Schaltungen sicher aufbewahrt werden kann.

Und es enthält...

Der Inhalt des Koffers (**Bild 1**) ist auf der Website von SparkFun und in der beiliegenden (auf Papier!) gedruckten Anleitung zusammengefasst - was auch gleich den einzigen kleinen Kritikpunkt offenbart: Eine wirklich vollständige Stückliste des Kits fehlt, so dass man weder den Inhalt nach dem Kauf überprüfen noch als Vorlage verwenden kann, wenn man Ersatzteile bestellen will, zum Beispiel, wie viele Widerstände man für eine Schaltung im Bausatz benötigt. Ansonsten gibt es an der mitgelieferten Dokumentation nichts zu bemängeln, im Gegenteil! Aber zu dieser Anleitung gleich mehr.

Als Prozessorboard liegt dem SIK das RedBoard von SparkFun bei (**Bild 2**), das - abgesehen von einigen kleinen Details - voll kompatibel zum bekannten klassischen Arduino Uno ist. Der auffälligste Unterschied ist SparkFuns sogenannter qwiic-Verbinder, eine 4-Draht-Verbindung, die den I²C-Bus mit 3,3-V-Versorgungsspannung für externe Hardware bereitstellt. SparkFun hat eine Reihe von Erweiterungen mit diesem Anschluss ausgestattet, die aber in den SIK-Projekten weder verwendet noch diskutiert werden. Der Anschluss bietet

aber natürlich Möglichkeiten für neue Herausforderungen, wenn alle Tutorials zum SIK abgeschlossen sind.
Obwohl das RedBoard-Prozessorboard voll kompatibel mit dem Arduino Uno ist und als solches auch in der Arduino-IDE verwendet wird, müssen Sie einen anderen USB-Treiber installieren, damit es mit Ihrem Computer interagieren kann. Aber keine Sorge, die komplette Installation der Arduino-Programmierumgebung für Windows, MAC OS und Linux wird ebenfalls in der Anleitung erklärt.

Was müssen Sie können?

Für den Aufbau der Projekte in der Anleitung sind keine Lötwerkzeuge oder -kenntnisse erforderlich. Alle zusätzlichen elektronischen Komponenten können auf dem mitgelieferten Breadboard (**Bild 3**) angeschlossen werden. Es werden lediglich Jumper-Kabel gesteckt, so dass sich selbst der unerfahrenste angehende Maker mit zwei linken Händen nicht die Finger verbrennen kann. Und wie versprochen, es sind keine Erfahrungen in Elektronik oder Programmierung nötig, wenn Sie mit den Projekten im SIK beginnen.

SIK-Handbuch

Das SparkFun Inventor's Kit beschreibt fünf sogenannte Projekte (Light, Sound, Motion, Display und Robot) und jedes Projekt umfasst mehrere Schaltungen, insgesamt 16. Vielleicht wären die Bezeichnungen Projektfelder beziehungsweise Tutorials passender gewesen, aber das ist Geschmackssache. Es beginnt - wie fast jedes Mikrocontroller-Programmiertutorial - mit einer blinkenden LED und endet mit einem sich bewegenden Roboter, der mit Hilfe eines Ultraschallsensors Hindernissen ausweichen kann.
Die Anleitung für jede Schaltung beginnt mit einer Erklärung der neu eingeführten elektronischen oder elektromechanischen Bauteile und der eingesetzten Konzepte. Danach folgt der Verdrahtungsplan, der erklärt, wie jede Schaltung mit Hilfe des Breadboards und der Jumper-Drähte aufgebaut wird, gefolgt von Anweisungen zum Hochladen des Sketches auf das RedBoard. Als Nächstes gibt es einen Abschnitt, der erläutert, wie der Arduino-Sketch für diese Schaltung funktioniert und was die wichtigsten Anweisungen im Programm bedeuten und bewirken. Dann gibt es Vorschläge für weitere Experimente mit der Software (Coding Challenges), und die Schaltungsbeschreibung endet mit Tipps zur Fehlersuche, was zu prüfen und zu tun ist, wenn die Schaltung nicht wie beschrieben funktionieren sollte. Die Anleitungen werden mit zunehmender Komplexität der Schaltungen und Sketches natürlich immer umfangreicher.
Es ist nicht nur diese konsequente Struktur aller Schaltungen, die das Handbuch so gut macht. Der Text wurde so kurz und einfach wie möglich gehalten, ohne wichtige Informationen auszulassen, und ist daher auch für nicht-englischsprachige Leser leicht zu verstehen. Das übersichtliche Layout und die Verwendung von Farben zur klaren Trennung der verschiedenen Abschnitte helfen ebenfalls enorm, aber es ist wohl vor allem den klaren Bilden zu verdanken, dass sich das Arbeiten mit dem SIK so angenehm gestaltet (**Bild 4**). Gute Zeichnungen sagen mehr als tausend Worte und so kann ein relativ dünnes Büchlein eine Menge nützlicher Informationen enthalten. Es ist eine wahre Freude zu sehen, dass im Zeitalter der Online-Informationen eine solide, in diesem Fall sogar richtig auf Papier gedruckte Dokumentation bereitgestellt wird! Hinweis: Im gedruckten Handbuch, das dem SIK beiliegt, sind einige Fehler noch nicht korrigiert, die Online-Version des Handbuchs [3] ist aber auf dem neuesten Stand.

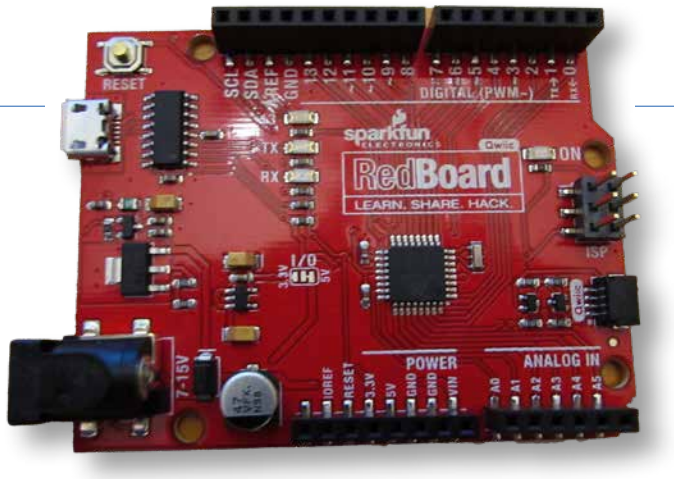


Bild 2. Das RedBoard im Arduino-Uno-Look.

Bild 1. Jede Menge schöner Sachen: das Innere der SIK-Box.

WEBLINKS

[1] Das SIK auf der Website von SparkFun: https://www.sparkfun.com/products/15267
[2] SIK Installation: https://bit.ly/2XwWYrh
[3] SIK Handbuch: https://www.sparkfun.com/SIKerrata

Passende Produkte

Sie suchen die erwähnten Bauteile? SparkFun und Elektor haben die passenden Produkte für Sie!

SparkFun Inventor's Kit - v4.1
www.elektormagazine.de/esfe-en-sik1

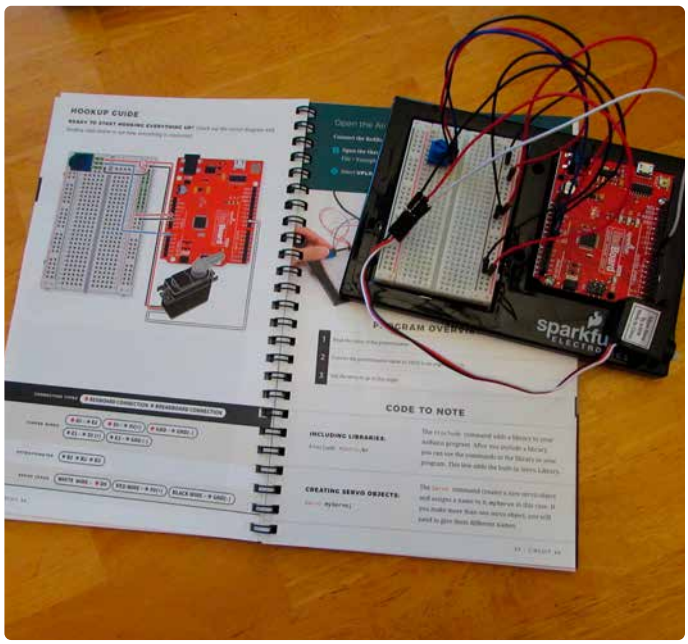


Bild 3. Die auf und mit dem SIK-Breadboard aufgebaute Servo-Schaltung.

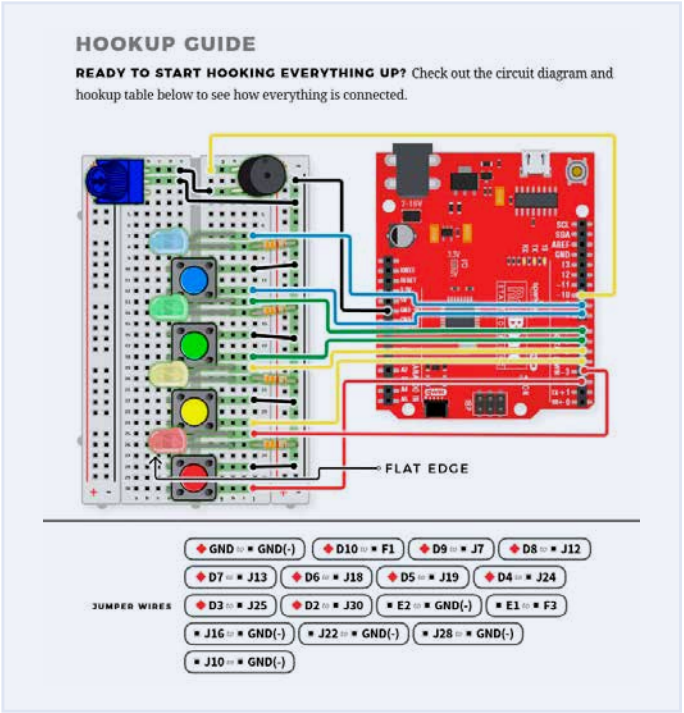


Bild 4. Illustrierte Anleitung zum Aufbau einer Schaltung.

Alles in allem sind die im SparkFun Inventor's Kit vorgestellten Schaltungen und Sketches zum Einstieg in die Elektronik und Programmierung in der Arduino-Programmierumgebung nicht besonders komplex. Aber es ist die Kombination aus der Hardware-Bestandteile Elektronik und Mechanik mit der Software und, vielleicht noch wichtiger, einer gründlichen, klaren Dokumentation, die das SIK so empfehlenswert macht. Es bietet hervorragendes Material für den Technikunterricht, sowohl an der höheren Schule als auch für das Selbststudium, und ist sogar für den erfahreneren Maker sehr unterhaltsam, vor allem, wenn man das Material mit einem Newbie „teilen" kann, der es mit der modernen Technik aufnehmen möchte!

200648-02

TDK·Lambda
Wo Qualität zählt ...
Zuverlässige Stromversorgung für die Industrie
T M E ®
Electronic Components
TRANSFER MULTISORT ELEKTRONIK
HUMBOLDTSTRAßE 2, 04105 LEIPZIG, DEUTSCHLAND
TEL. +49 341 212 03 40, TME@TME-GERMANY.DE
tme.eu
facebook.com/TME.eu
youtube.com/TMElectroniComponent
instagram.com/tme.eu

Glenn Samala von SparkFun **über Produktentwicklung und neue Projekte**

Von C.J. Abate (Elektor)

SparkFun hat im herausfordernden Jahr 2020 erstaunliche 70 neue Produkte auf den Markt gebracht. Geschäftsführer Glenn Samala berichtet über seine Gedanken zum Produktentwicklungsprozess und was die DIY-Elektronik-Community in den kommenden Wochen und Monaten vom Unternehmen erwarten kann.

Abate: Sie haben neun Jahre bei Arrow Electronics gearbeitet, bevor Sie zu SparkFun kamen. Was hat Sie zu SparkFun geführt?

Samala: Wie bei den meisten CEO-Positionen begann es damit, dass mich ein Personalvermittler kontaktierte. Was mich aber letztendlich zu SparkFun führte, war die Geschichte des Unternehmens und seines Gründers. Im Jahr 2003, noch während seiner College-Ausbildung, gründete er diese Tech-Firma aus seinem Zimmer im Studentenwohnheim der Universität von Colorado heraus, ohne jegliche finanzielle Unterstützung. Mit einigen jungen und talentierten Mitarbeitern hat er gegen alle Widrigkeiten der großen Finanzkrise in den USA ein Unternehmen aufgebaut, das seine Ideen von Technologie mit dieser neuen Welt des Digitalen und Sozialen auf eine sinnvolle und wirkungsvolle Weise verbindet. Die Gelegenheit, dieses Unternehmen auf die nächste Stufe zu führen, konnte ich mir einfach nicht entgehen lassen.
Ich sah auch diese tolle Gelegenheit, Nathan Seidles offensichtliches unternehmerisches Talent mit meiner völlig unterschiedlichen Erfahrung zu verbinden, um SparkFun auf einen einzigartigen Weg zu führen. Nathan ist ein Unternehmer, der erfolgreich ein Unternehmen ohne Finanzierungsrunden gegründet hat, ich dagegen habe meine Karriere bei einem großen börsennotierten Unterneh-

men begonnen, das bei der Integration von übernommenen Unternehmen geholfen hat – Umsatz 800 Millionen, als ich anfing, und 9 Milliarden, als ich ging. Der Hintergrund des Gründers von SparkFun und des neuen Geschäftsführers könnte also nicht unterschiedlicher sein. Aber das ist es, was für mich wirklich interessant war.
Der Übergang der CEO-Position vom Gründer zu einem „externen" Geschäftsführer ist immer knifflig, besonders wenn es sich um den ersten Geschäftsführer handelt, der den Gründer ersetzt. Es war eine der größten Herausforderungen, die ich in meiner Karriere erlebt habe, aber auch eine der schönsten Erfahrungen.

Abate: Wie hat sich COVID-19 auf SparkFun ausgewirkt? Können Sie uns von ein oder zwei größeren Maßnahmen erzählen, die Sie vornehmen mussten?
Samala: COVID-19 hat uns auf ähnliche Weise wie so viele Unternehmen auf der ganzen Welt beeinflusst. Angefangen bei der globalen Lieferkette bis hin zu der Art und Weise, wie wir miteinander umgehen und die tägliche Interaktion erleben. Unser Hauptaugenmerk liegt auf dem Schutz der Gesundheit unserer Mitarbeiter. Daher haben die meisten, wenn nicht sogar alle unsere größeren Anstrengungen den Schutz unserer Mitarbeiter zum Ziel, wobei uns Datenlage und Wissenschaft helfen, die schwierigen Entscheidungen zu treffen.
Jedes funktionale Team ist in unserer Hauptniederlassung in Boulder (IT, Produktservice, HR, Entwicklung, Marketing, Lieferkette, Verkauf, Geschäftsentwicklung, Finanzen, Ops Admin, Herstellung und Lagerhaltung) untergebracht, was es uns ermöglicht hat, eine hochgradig kollaborative Kultur im Büro zu schaffen. Am 13. März 2020 rief die US-Regierung dann den Ausnahmezustand aus, und bis zum 20. März verlegten wir 70% unserer Mitarbeiter ins „Home-Office". Die einzigen Mitarbeiter, die sich derzeit in unserer Unternehmenszentrale aufhalten dürfen, sind mit der Lagerhaltung und dem Fertigungsprozess beschäftigt. Der Umzug von 70% unserer Mitarbeiter ins Home-Office war schwierig, aber notwendig, um alle zu schützen. Und für das Team vor Ort haben wir einen gestaffelten Zeitplan erstellt, um mehr Raum zu schaffen und eine soziale Distanzierung zu ermöglichen. Snacks, Essen und Getränke werden bereitgestellt, um so viel Verkehr in und aus dem Gebäude zu vermeiden. Täglich werden Körpertemperaturkontrollen durchgeführt und die erforderlichen persönlichen Schutzutensilien (Masken, Handschuhe, Desinfektionstücher und Handdesinfektionsmittel) vor Ort bereitgestellt.

Lokal ist, wo man lebt; Talent und großartige Ideen sind global.

Glenn Samala

Ich weiß, dass jeder gerne wieder ins Büro gehen möchte - die Anregungen und Ideen, die sich aus informellen Gesprächen auf dem Flur ergeben, haben bei SparkFun wirklich einen echten Wert. Wir alle vermissen diese informellen Gelegenheiten, aber wir werden es zur rechten Zeit nachholen.

Abate: Was waren Ihre beliebtesten Produkte im letzten Jahr? Gab es Überraschungen?
Samala: Ein Großteil unseres Fortschritts im Jahr 2020 lässt sich auf die Nachfrage nach Rapid-Prototyping-Tools zurückführen. Jeder versucht, mit den neuen Technologien zu experimentieren und schneller zum Proof-of-Concept zu kommen - hier kommen wir ins Spiel.
Alle unserer Qwiic-Produkte finden weiterhin Anklang - von den Qwiic-Entwicklungsboards bis hin zu Qwiic-Sensoren und allem, was dazwischen liegt (Qwiic-Accessory Boards, HATs und so weiter). Der Wert des Rapid Prototyping, den das Qwiic-Ökosystem genau wie viele andere unserer Produkte und Dienstleistungen bietet, steigt weiter.
SparkFun erhält auch eine Menge Aufmerksamkeit durch die Markteinführung von Artemis, ALC, MicroMod und durch unsere Services. Dies sind neue und aufregende Dinge. Aber das Team war genauso damit beschäftigt, neue SparkFun-Originale und Partnerprodukte wie Alchitry, NVIDIA, micro:bit und Raspberry Pi herauszubringen. Um Ihnen eine Vorstellung von der Größenordnung zu geben: SparkFun hat im Jahr 2020 zusätzlich zu all diesen Wachstumsinitiativen über 70 Originalprodukte vorgestellt.
Eine angenehme Überraschung sind die mehr als sieben Jahre alten SparkFun-Originalprodukte, die sich weiterhin in hohen Stückzahlen verkaufen. So wurde zum Beispiel wurde der Logic-Level-Konverter 2013 auf den Markt gebracht und gehört immer noch zu unseren Bestsellern. Das zeigt, dass die Nachfrage nach Basiskomponenten, die gut dokumentiert sind, weiterhin einen Wert auf dem Markt darstellt.

Abate: Wo sehen Sie in den nächsten sechs Monaten Wachstumschancen?
Samala: Ich denke, es hängt davon ab, über welchen Kundenmarkt wir sprechen. SparkFun hat eine großartige und vielfältige Kundenbasis, von den Hochschulen (Colleges und Universitäten) über den Hacker in der Garage, den Unternehmensgründer, Startups, Forschung und Entwicklung bis hin zu großen Unternehmen.
Aus der Produktperspektive denke ich, dass alles, was mit aufstrebenden Technologien wie Low-Power-Machine-Learning und RISC-V zu tun hat, das Potenzial hat, ungefähr in den nächsten sechs Monaten richtig durchzustarten. Ich denke auch, dass ausgereifte Technologien, die bisher dem professionellen Bereich vorbehalten waren, ihren Weg „nach unten" auf die Ebene der Verbrauchertechnik finden werden - man denke nur an Technologien der Geomatik und globalen Verfolgung von Wirtschaftsgütern.
Die Ungewissheit und Unvorhersehbarkeit des Jahres 2020 wird sich im Jahr 2021 fortsetzen. Wer weiß wirklich, wo das Wachstumspotenzial in sechs Monaten liegt? Aber was ich weiß, ist, dass COVID das Innovationstempo im Jahr 2020 nicht verlangsamt hat. Solange SparkFun sich darauf konzentriert, neue und interessante Technologien für die breite Masse zugänglich zu machen, werden wir weiter wachsen.

SparkFun befindet sich in Niwot, Colorado, USA.

Spaß ist Teil der Unternehmenskultur.

Abate: SparkFun scheint viel Wert auf die Erstellung von Inhalten zu legen: Projektartikel, Blogbeiträge und Videos. Können Sie uns etwas über Ihren Ansatz bei diesem Content Marketing erzählen?

Samala: Digitale Inhalte sind ein wichtiger Grund, warum SparkFun heute existiert. Vor achtzehn Jahren war es für Nathan schwer genug, ein Einplatinen-Entwicklungsboard zu kaufen (was der andere wichtige Grund ist, warum SparkFun heute existiert), aber die andere Herausforderung war, dass es keine richtige Dokumentation gab, die ihm sagte, was er mit dem Produkt tun sollte. Als Nathan mehr über ein neue Produkt lernte, dokumentierte und teile er es. Dieser Schritt mag heute nicht revolutionär erscheinen, aber vor 18 Jahren war er es. Er schuf damit einen enormen Wert für das Produkt.

Was aber vor mehr als 10 Jahren einen zusätzlichen, besonderen Wert darstellte, wird heute als selbstverständlich erachtet. Daher sind wir immer bestrebt, unsere digitalen Inhalte so zu erweitern, dass sie für unsere Kunden weiterhin einen echten Nutzwert und ein Erlebnis darstellen. Das ist der Grund, warum es so viele Inhalte rund um unsere Produkte gibt. Aber denken Sie auch daran, dass dies nicht über Nacht passiert ist, sondern sich über 18 Jahre hinweg entwickelt hat.

Was unser Content Marketing betrifft, so wollen wir eine digitale Plattform schaffen, auf der die Menschen berichten können, was unser Produkt leistet, wofür sie es verwenden und wie und warum sie ein Projekt erstellt haben. Es geht also nicht nur um das Produkt, sondern auch darum, was die Leute damit machen.

Abate: Ich nehme an, dass Ihre Ingenieure und Community-Mitglieder jedes Jahr Dutzende von großartigen Produktideen einreichen. Wie entscheidet Ihr Team, welche Produkte weiterentwickelt werden sollen?

Samala: Wir erhalten jedes Jahr Dutzende von großartigen internen und externen Produktideen, und manchmal hat man wirklich die Qual der Wahl. Bei Produktideen von außerhalb trifft Kirk Benell, unser Technischer Direktor, die endgültigen Go/NoGo-Entscheidungen. Kirk wägt ab, ob und wie eine externe Produktidee in unser Produktangebot passt und bestimmt, ob das Produkt uns helfen kann, in bestimmte für uns interessante Märkte vorzudringen.

Aus interner Sicht kommen viele der Produktideen von unserem SparkX-Labor, von dem Team, das (wieder) von Nathan dem Gründer und jetzt Nathan dem Ingenieur geleitet wird und das mit einem hohen Maß an Autonomie und Freiheit arbeitet, unabhängig von Markt- oder Geschäftsüberlegungen. Wenn ihnen ein neues Produkt oder eine neue Technologie zusagt, entwickeln sie das Produkt und stellen es unter der Marke SparkX der Öffentlichkeit vor. Das Interesse der Community stellt für uns einen schnellen Markttest dar und hilft uns zu entscheiden, ob das Produkt von SparkX „Black" zu SparkFun „Red" werden sollte.

Wie bei den meisten Produktentwicklungsprozessen ist die Entscheidung nicht immer so einfach und klar. Es gibt Fälle, bei denen Kirk in einer externen Idee keine Anwendungsmöglichkeit sieht, aber dann Nathan fragt, ob er an dem neuen Produkt/der neuen Technologie interessiert wäre. Und auf der anderen Seite erhält Nathan manchmal eine Produkt-/Technologieidee, an der er nicht sonderlich interessiert ist, die aber in unser Produktportfolio passen könnte. Auf jedem Fall gibt es ein hohes Maß an Zusammenarbeit zwischen Nathan und Kirk, und sie ziehen mich bei Bedarf hinzu.

Abate: SparkFun ist dafür bekannt, den Markt der Hobbyelektroniker und Maker anzusprechen. Aber mit der Entwicklung und Einführung eines Produkts wie Artemis und eines Dienstes wie À La Carte scheinen Sie auch über diesen Markt hinaus zu schauen?

Samala: Ich würde nicht sagen, dass ich über den Markt für Maker hinausschaue. Es ist eher

so, dass SparkFun mit diesem Markt wächst. Wir haben einen großen Kundenstamm von Enthusiasten, die uns seit 18 Jahren folgen. In dieser Zeit haben viele von ihnen ihre eigenen Unternehmen gegründet. Viele haben auch angefangen, in der Forschung und Entwicklung zu arbeiten und sind mit Rapid Prototyping beschäftigt. Der Punkt ist: Der lebenslange Prozess des Lernens und Wachsens als Unternehmen oder Einzelperson hört nie auf, und wir freuen uns, dass wir mit unserer Community wachsen und uns weiterentwickeln.
Egal, ob jemand gerade erst mit der Elektronik anfängt oder ob er schon so weit fortgeschritten ist, dass er seine eigene Produktidee auf den Markt bringen möchte, wir wollen diese Plattform für unsere alten und neuen Kunden sein, indem wir ihnen Zeit und Ressourcen sparen helfen.

Abate: Haben Sie viele Kunden außerhalb der Vereinigten Staaten? Was ist mit Europa?
Samala: Ja. Im Jahr 2020 sind etwa 30% unseres aktuellen Geschäfts international, im internationalen B2B-Segment unseres Geschäfts aufgrund von COVID etwas weniger.

Abate: Warum hat SparkFun À la Carte eingeführt?
Samala: Bei SparkFun ging es schon immer darum, die Bausteine der Technologie bereitzustellen, mit denen die Menschen in die Wunderwelt der Elektronik einsteigen und fortschreiten können. Ein Tool wie À la Carte (ALC) geht noch einen Schritt weiter und ermöglicht es jedem, diese Bausteine in ein eigenes, individuelles Board zu integrieren.
À la Carte ist für diejenigen gedacht, die in der Prototyping-Phase ihres Produkts Zeit und Ressourcen sparen wollen. Viele Unternehmen haben einfach nicht das Kapital, die Ressourcen oder die Hardware-Expertise, um schnell die Phase des Prototypings zu durchlaufen, und ALC soll dies gewährleisten.
Wir wissen, dass Professoren an den Hochschulen gerne kundenspezifische Kits einsetzen. Wir arbeiten seit Jahren erfolgreich daran, solche Kits zu erstellen, die auf bestimmte Kursziele abgestimmt sind. ALC geht noch einen Schritt weiter und gibt Dozenten die Möglichkeit, selber individuelle Boards für ihren Lehrplan oder ihre Abschlussprojekte zu erstellen. ALC bietet also die Möglichkeit, einen individuellen Ansatz zu verfolgen - egal ob im Bildungswesen oder im privaten oder geschäftlichen Bereich.

Es gibt so viele tolle Hardware- und Software-Ideen da draußen. Unsere Absicht ist es, diesen Ideen zum Erfolg zu verhelfen.

Glenn Samala

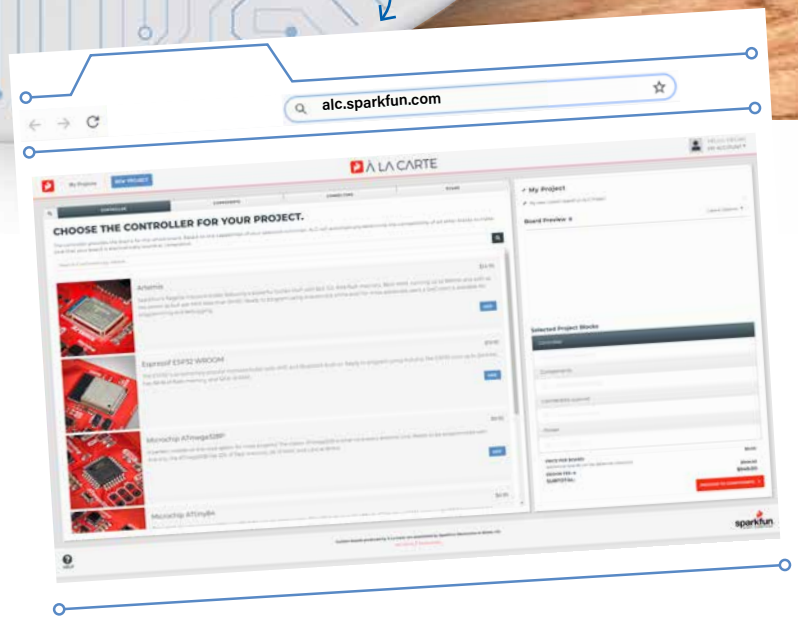
Abate: À la Carte dürfte Ihr Team vor eine einzigartige geschäftliche Herausforderung stellen. Welche Erfahrungen haben Sie dabei gemacht? Und was macht SparkFun anders als andere Auftragsfertiger?
Samala: Jedes Mal, wenn das Wort „kundenspezifisch" in einer geschäftlichen Diskussion fällt, verkompliziert es die Dinge und schafft Herausforderungen. In Bezug auf ALC entfällt ein Großteil der Arbeit auf unsere Operation-Teams Produktion/Fertigung und Versorgungskette.
Wir stellen alle Boards tatsächlich in unserem Hauptsitz in Colorado her. Die kundenspezifischen ALC-Boards durchlaufen einen ähnlichen Prozess wie unsere SparkFun-Originale. Während uns aber der Produktionsprozess von originalen SparkFun-Boards in Fleisch und Blut übergegangen ist, muss unser Team bei ALC-Boards die besonderen individuellen Anforderungen, das Tempo der Produktion und den Umfang des Auftrags berücksichtigen, und dies immer mehr, da das ALC-Tool immer beliebter wird.
Was uns von anderen Auftragsfertigern unterscheidet? Ich würde einfach sagen, dass wir kein Auftragsfertiger sind, denn hohe Stückzahlen für eine ausgereifte Produktpalette sind nicht unser Ding. Solche Dinge überlasse ich Auftragsfertigern wie Flextronic, FoxConn oder Jabil - sie machen das gut, besser, als SparkFun dies könnte.
Bei À la Carte geht es darum, neue Ideen zu unterstützen und Kunden zu helfen, diese Ideen so schnell und problemlos wie möglich auf den Markt zu bringen. Unsere Initiativen wie ALC zeigen unser Engagement, anderen zu helfen, im Elektroniksektor zu wachsen - speziell in der Nische der Entwicklung und des Rapid Prototypings. Dieser Vision fühlt sich unser Team verpflichtet.

Abate: Erzählen Sie uns ein wenig über Boulder in Colorado und die Umgebung. Gibt es dort eine Tech-Szene? Ist es ein guter Standort, um technische Talente zu finden?
Samala: Es gibt eine große Tech-Szene in der Stadt Boulder und dem Umland von Denver. Hier herrscht ein hohes Maß an Zusammenarbeit, Unterstützung und Freundschaft zwischen den Unternehmen - es ist eine echte Gemeinschaft. Boulder selbst ist in den letzten mehr als 15 Jahren rasant gewachsen und hat Dutzende von kleineren bis mittelgroßen Tech-Unternehmen angezogen - viele davon aus Kalifornien. Auch der Zustrom von Talenten und Unternehmen, die nach Colorado kommen, ist sehr vorteilhaft für Colorado. Aber ich würde auch sagen, dass sich die Welt grundlegend verändert hat, wenn so etwas wie COVID uns lange beeinträchtigt und auf eine Art und Weise, über die wir noch gar nicht richtig nachgedacht haben.
Speziell zu Ihrer Frage nach den Talenten: Ich denke, dass Talente nicht mehr nur lokal oder regional gesucht werden. Das Modell, in irgendeiner regionalen Tech-Hochburg eine Firmenflagge aufzustellen und damit Tech-Talente anzuziehen, hat sich schon seit einiger Zeit verändert, und ich denke, COVID hat diesen Wandel beschleunigt. Wir haben schon vor COVID Talente und Partner auf globaler Ebene gewonnen, und ich kann Ihnen sagen, dass sich dies im Jahr 2020 nur noch beschleunigt hat. Lokal ist, wo man lebt; Talent und großartige Ideen sind global.

Abate: Wenn Sie in die Zukunft blicken: Wie wird sich SparkFun entwickeln müssen, um die Bedürfnisse der Community zu erfüllen?
Samala: In der Tech-Branche, insbesondere in einer Open-Source-Umgebung, entwickelt man sich weiter, indem man neue Werte in seinem Modell schafft. Manchmal ist dieser neue Wert offensichtlich, und manchmal wird dieser Wert zufällig geschaffen.
Bei SparkFun entwickeln und testen wir viele Bereiche, die einen Wert darstellen könnten - unsere Produkt-Roadmap, SparkX, Partnerschaften, Dienste und das Feedback der Community. Wir tun unser Bestes, um sowohl selbstverständliches als auch zufälliges Wachstum zu ermöglichen. Wir werden uns zunehmend auf diese Art und Weise verlassen, um sicherzustellen, dass wir die sich ständig ändernden Bedürfnisse der aktuellen und zukünftigen Kunden erfüllen.

(200687-02)

Eigene Boards entwickeln mit SparkFuns À La Carte



Von Megan Hemmings (SparkFun)

Bei SparkFun entwickeln und produzieren wir seit mehr als einem Jahrzehnt Platinen. Seither sehen wir, dass unsere Kunden von solchen, die nur einen einfacheren Zugang zu elektronischen Bauteilen suchen, zu solchen werden, die nach der neusten und besten Technologie verlangen, und schließlich zu Kunden, die Produkte mit ihrem eigenen Design erstellen und verkaufen wollen. So wie sich die Motivation unserer Kunden entwickelt hat, ist auch das Angebot von SparkFun gewachsen.

SparkFuns *À La Carte* (ALC) ist ein kundenspezifischer Design- und Fertigungsservice für Elektronikplatinen, der die Lücke zwischen Prototyp und Produktion überbrückt. ALC wurde geschaffen, um unseren Kunden Hilfestellung bei der Produktion von Platinen zu geben, auch und besonders wenn sie keine Erfahrung im Platinen-Design haben.

„Nicht jeder, der eine Idee hat, ist ein Hardware-Ingenieur oder kennt einen, der ihm hilft, seine Idee zu verwirklichen", sagt Nathan Seidle, Gründer und Ingenieur bei SparkFun. „Das Schöne an ALC ist, dass es die ganze komplizierte Arbeit für Sie erledigt. Sie müssen nur wissen, welche Teile Sie wie verwenden möchten, und unser System verknüpft sie mit Leiterbahnen und platziert die Funktionsblöcke, so dass jeder ein maßgeschneidertes Board erstellen kann."

Seidle meint, dass es einen besseren Weg geben muss, um kundenspezifische Boards zu erstellen, als Drähte zu schneiden und abzuisolieren, Breakout-Boards zu löten, alles zu testen und dann dies bei der folgenden Iteration zu wiederholen, bis alles wirklich passt. Stattdessen bietet ALC (**Bild 1**) eine einfache Schnittstelle zum Entwerfen und Herstellen von Boards, die den spezifischen Anforderungen eines jeden Kunden entsprechen.

Mit ALC wird, wenn Sie Ihre Idee mit SparkFun-Breakout-Boards realisieren, ein vollständig bestücktes, kundenspezifisches Board in wenigen Wochen bis an Ihre Tür geliefert. Keine nächtlichen Lötsitzungen mehr vor dem großen Tag der Markteinführung - nutzen Sie die Designs, das Schaltungs-Know-how und die Maschinen von SparkFun, um ein professionelles Board nur für Sie anfertigen zu lassen. **Bild 2** zeigt die zur Verfügung stehenden ALC-Blöcke.

Die À-La-Carte-Schnittstelle

Die Schnittstelle von *À La Carte* basiert auf Blöcken. Ein ALC-Block enthält das Modul, das der Karte hinzugefügt werden soll, zusammen mit Informationen über Netze, Klassen, Verbindungseinschränkungen und andere elektr(on)ische Anforderungen und Einschränkungen. Durch die Einbeziehung all dieser Informationen durch ALC muss der Benutzer keine intimen Kenntnissen über eine Technologie besitzen, die normalerweise nur durch viele Versuche und viele Irrtümer zustande kommen. Um eine Platine mit ALC zu erstellen, wählt der Anwender nur die verschiedenen Blöcke aus, die er für sein Design benötigt. Das System erledigt den Rest und sorgt für eine elektrisch einwandfreie Platine (**Bild 3**).

Gerber und Schaltpläne

Alle ALC-Bestellungen enthalten eine herunterladbare Version der Gerber-Dateien und eine PDF-Datei des Schaltplans. Sobald ein Board gebaut wurde, hat der Anwender die Möglichkeit, vollständig geroutete Eagle-Quelldateien (.BRD und .SCH) zu erwerben. Wenn diese Dateien freigeschaltet sind, kann das Design – wenn erforderlich - für die traditionelle Fertigung durch SparkFun oder anderswo optimiert und angepasst werden. Wenn keine Anpassungen erforderlich sind, können die Boards natürlich einfach jederzeit mit ALC bestellt werden.

„Mit *À La Carte* haben wir SparkFuns in vielen Jahren erworbenes Wissen über Board-Entwurf und -Fertigung in eine einfach zu bedienende Plattform gepackt, die jeder nutzen kann, um eigene Boards zu erstellen", sagte SparkFun-Geschäftsführer Glenn Samala. „Mit diesem Angebot hoffen wir, unser Serviceangebot abzurunden und unseren Kunden zu erleichtern, den Proof-of-Concept zu erreichen und die Produkte auf den Markt zu bringen."

Um mit dem Bau eines eigenen Boards zu beginnen, besuchen Sie ALC unter https://alc.sparkfun.com/.

Vier Schritte zum eigenen Board

Der ALC-Designer macht den Entwurf einer eigenen Platine so einfach wie möglich. ALC führt Sie durch die Auswahl des Controllers, der Bauteile und Komponenten, der Steckverbinder und der Stromversorgung. Sie müssen lediglich die Blöcke auswählen, die Sie zu Ihrem Board hinzufügen möchten. Wenn nicht genügend Ressourcen für eine Komponente vorhanden sind, lässt ALC Sie diesen Block nicht auswählen. Derzeit fordert ALC, dass alle Boards mindestens einen Controller, eine Komponente und eine Stromversorgung enthalten.

Um zu zeigen, wie einfach dies alles abläuft, lassen Sie uns ein automatisches Lüftungssystem entwerfen, das einen Abluftventilator einschaltet und das aus der Ferne überwacht werden kann.

Schritt 1: Wählen Sie einen Controller. In diesem Fall wählen wir SparkFuns Flaggschiff, den Artemis, da er die Fähigkeit hat, das Projekt in der Zukunft mit Bluetooth 5.0 und mit einem großen Flash-RAM von 384 K für den gesamten Code zu erweitern (**Bild 4**).

Schritt 2: Komponenten hinzufügen. Für diesen Entwurf benötigen wir:

- Das **16×2-LCD** zur Anzeige der Messwerte
- Den **Luftqualitätssensor BME680** zur Erfassung von Temperatur, Luftfeuchtigkeit und Luftqualität (VOC)
- Einen **RGB-Encoder** zum Umschalten zwischen den Werten für Temperatur, Luftfeuchtigkeit und Luftqualität
- Dreifacher **Pixel-LED-Streifen APA102**

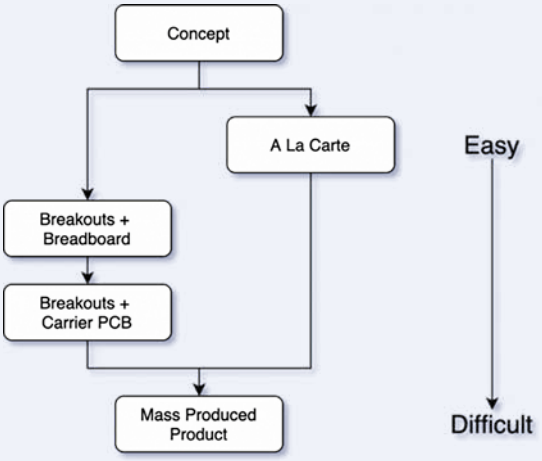


Bild 1. Die von SparkFuns À la Carte angebotenen Prototyping-Blöcke.

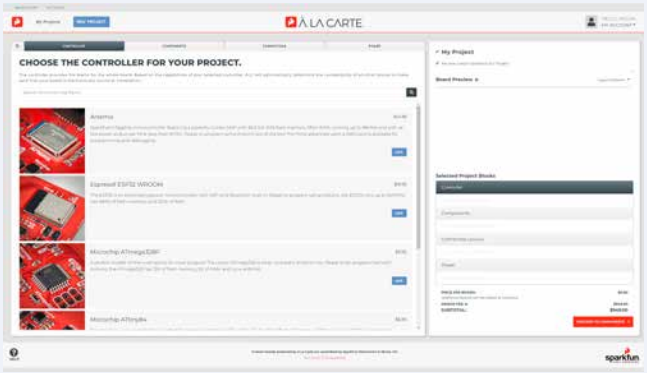


Bild 2. Übersicht der ALC-Blöcke, die dem Benutzer zur Verfügung stehen.

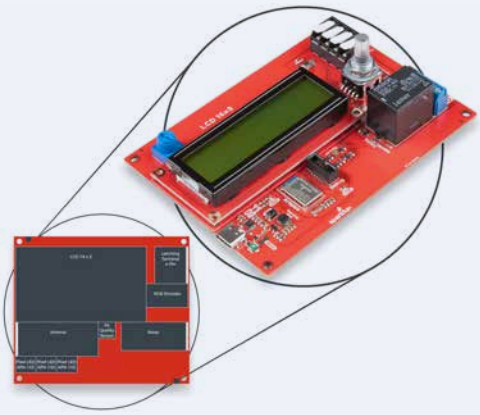


Bild 3. Eine Beispielplatine im ALC-Design.

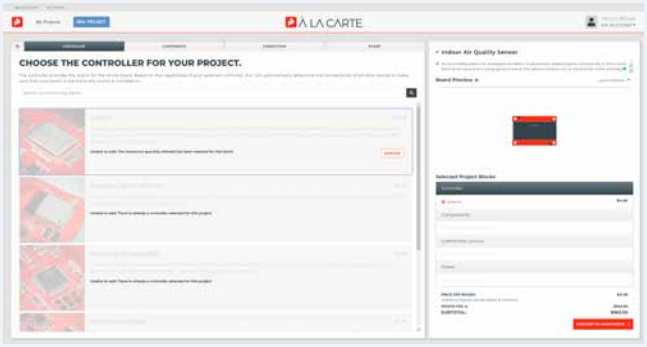


Bild 4. Wählen Sie Ihren Controller aus!

zur Anzeige des VOC-Status (grün gleich akzeptabel, gelb grenzwertig und rot gleich gesundheitsschädlich)
- Ein **Qwiic-Anschluss**, um in Zukunft weitere Sensoren und Funktionen hinzufügen zu können
- Eine **LED**, die anzeigt, ob das Board eingeschaltet ist
- Und ein **Relais**, um bei zu hohen VOC-werten einen Luftreiniger oder Abluftventilator einzuschalten

Diese Komponenten sind rechts in **Bild 5** eingetragen.

Schritt 3: Hinzufügen von Verbindern (optional). Jedes Signal auf der Platine kann über eine Vielzahl von verschiedenen Steckertypen zugänglich gemacht werden. In diesem Projekt fügen wir einen 4-poligen Verriegelungsklemmblock hinzu (**Bild 6**), damit wir die Möglichkeit haben, zukünftig zusätzliche Bauteile anzuschließen, zum Beispiel einen Feinstaubsensor.

Schritt 4: Fügen Sie einen Power-Block hinzu. Für dieses Indoor-Projekt im Büro verwenden wir ein Steckernetzteil (**Bild 7**).

Das war es schon, das Board ist einsatzbereit. Wie Sie sehen können, hat ALC währenddessen eine Vorschau der Platine generiert (**Bild 8**). Richtig aufregend wird es natürlich erst, wenn Sie die „echte" Platine in Ihren Händen halten (**Bild 9**)!
À La Carte ermöglicht es Ihnen also, jedes Problem oder jedes Projekt anzugehen, egal, wie „kundenspezifisch" es auch sein mag. Starten Sie Ihr eigenes Design unter http://alc.sparkfun.com.

Boards gebaut mit À La Carte

Die Designmöglichkeiten sind mit *À La Carte* nahezu endlos. Um dies zu veranschaulichen, zeigen wir Ihnen einige Boards, die wir mit ALC erstellt haben.

Steuerung für einen Reflow-Ofen
Ziel: Erstellung einer Steuerplatine, die eine Temperaturregelung bei einem selbstgebauten Reflow-Ofen ermöglicht (**Bild 10**)
Verwendete Blöcke:
- Controller: Artemis
- Komponenten:
  - RGB-Encoder - um die Navigation in den Menüs zu ermöglichen und eine Auswahl zu treffen
  - 16x2-Zeichen-LCD - zur Anzeige notwendiger Informationen
  - (2) Thermoelement-Verstärker - zur Messung der Ofentemperatur
- (2) Relais - zur Steuerung des Ofens
- Stromversorgung: USB Power

Cuisine
Ziel: Ein IoT-Gerät, das so ziemlich jedes Kochprojekt überwachen kann (**Bild 11**).
Verwendete Blöcke:
- Controller: ESP32 WROOM
- Komponenten:
  - 16x2-LCD - zur Anzeige von Informationen wie Timer, Temperatur, Gewichte und mehr

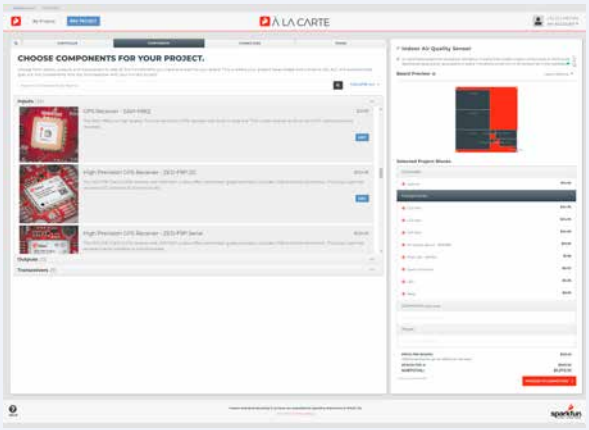


Bild 5. Wählen Sie Ihre Komponenten!

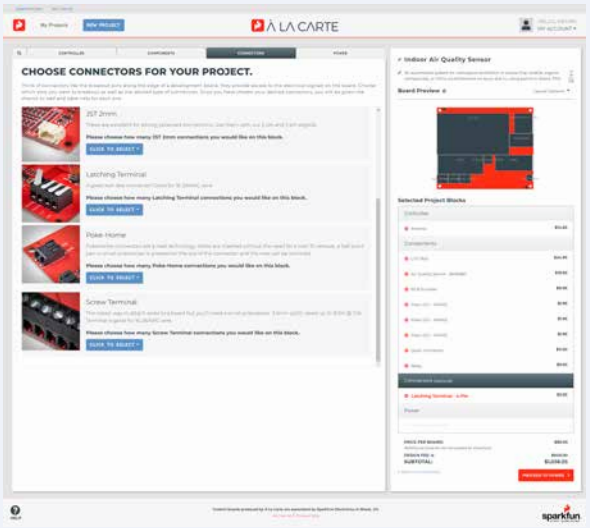


Bild 6. Ein vierpoliger verriegelbarer Klemmanschluss.

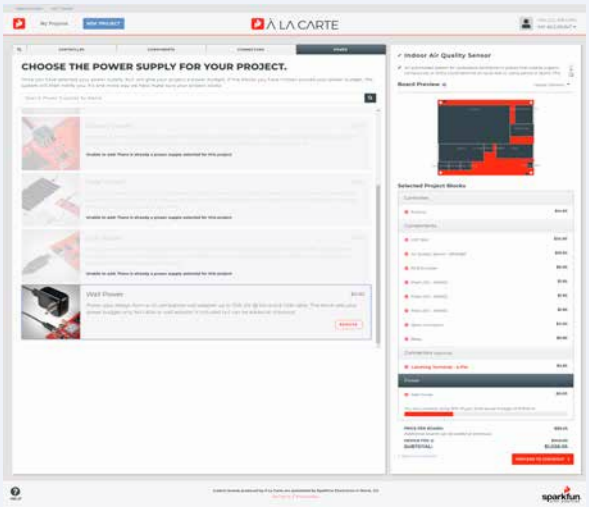


Bild 7. Auswahl der Spannungsversorgung - ein Steckernetzteil.

Bild 8. Vorschau auf ein ALC-Board.

Bild 9. Eine „echte" ALC-Platine.

Bild 10. Refow-Ofen-Platine.

Bild 11. Cuisine-Platine - los geht es mit dem Kochen!

Bild 12. Platine für eine Wetterstation.

Bild 13. Hühnerstall-Fernüberwachung.

Bild 14. Zugangscontrolle basierend auf RFID.

  - Thermoelement - um eine sichere und genaue Temperaturmessung zu ermöglichen
  - Digitalwaage - weil Waagen in der Küche immer praktisch sind, um Gewichtsveränderungen festzustellen, zum Beispiel das Gewicht eines Regals im Kühlschrank oder das Gewicht eines Bratens, während er gegart wird
- Qwiic-Anschluss - für zukünftige Erweiterungen und Ergänzungen
- Stromversorgung: Batteriebetrieb

Wetterstation

Ziel: Schaffung eines Umweltsensorknotens, der Lüfter steuern kann und eine Erweiterbarkeit auf andere Sensoren über Bluetooth ermöglicht (**Bild 12**).

Verwendete Blöcke:

- Controller: Artemis
- Komponenten:
  - (2) Taste - weil alles Knöpfe haben sollte
  - Wettermessgerät – um Regensensor, Anemometer und Windrichtungssensor anschließen zu können
  - (2) LED - als Hilfe bei der Fehlersuche
  - Hochstromschalter – mit Verriegelungsklemmen, um zusätzliches Zubehör wie einen Lüfter anzuschließen und über Bluetooth steuern zu können
- Luftqualitätssensor BME680 - zur Erfassung von Luftdruck, Temperatur, Feuchtigkeit und VOC
- Stromversorgung: 12 V Autobatterie

ChickenNet

Ziel: Erstellung eines IoT-Controller-Boards zur Überwachung eines abgelegenen und außer Sichtweite befindlichen Hühnerstalls (**Bild 13**).

Verwendete Blöcke:

- Controller: ESP32 WROOM
- Komponenten:
  - Adressierbarer RGB-LED-Streifen – um für „Tageslicht" auch während der Wintermonate zu sorgen, damit die Hühner weiterhin viele Eier legen
  - Waage - um das Gewicht einer Legebox zu überwachen und zu erkennen, ob sie belegt ist und/oder ein Ei hineingelegt wurde
  - Bewegungssensor - um zu erkennen, ob die Hühner zu Hause sind
  - BME280 / CCS811 - zur Überwachung von Temperatur, Feuchte, Luftdruck und Luftqualität
- RFM95 LoRa Radio - für erweiterte drahtlose Anwendungen, wenn der Stall außerhalb der WLAN-Reichweite sind.
- Stromversorgung: USB Power

RFID-Zugangssteuerung

Ziel: Eine einfache programmierbare Zutrittskontrolle (**Bild 14**).

Verwendete Bausteine:

- Controller: ATmega328p
- Komponenten:
  - Basic RFID – 125 kHz - zum Erkennen und Lesen von RFID-Schlüsseln
  - Hochstromschalter - zur Ansteuerung eines elektronischen Türöffners
  - Türkontaktschalter - damit die Steuerung weiß, wann die Tür offen ist
  - (2) Taster - nur für den Fall der Fälle, es ist immer gut, Taster in Reserve zu haben
- Pixel-LED - APA102 - als Statusanzeige
- Stromversorgung: Netzspannung.

200690-03

Entwerfen mit dem SparkFun-Artemis

Von Nathan Seidle (SparkFun)

Das Artemis-Modul ist das erste Open-Source-Hardware-Funkmodul der Welt, das sowohl Spracherkennung als auch BLE ermöglicht. Auf 10 × 15 mm² lässt sich eine erstaunliche Menge an Leistungsfähigkeit unterbringen. Dieses Tutorial führt Sie durch die Funktionen des Artemis-Moduls sowie die Grundlagen, die Sie kennen müssen, um Artemis in Ihr eigenes Projekt einzubinden!

Jahrelang hat sich SparkFun davor gedrückt, ein Produkt vollständig FCC-zertifizieren zu lassen (vergleichbar mit CE). Die Zertifizierung erschien uns immer zu kompliziert und unerschwinglich teuer. Wir haben in der Vergangenheit über die FCC-Anforderungen für Produkte und Hobby-Projekte geschrieben, aber mit der Entwicklung von SparkFun-Edge und Artemis schien es an der Zeit, sich auf unbekanntes Terrain zu wagen.

Im Oktober 2018 wollten wir mit den Leuten von TensorFlow ein Low-Power-Board mit dem neuen Apollo3 von Ambiq zu entwickeln. Nach einem kurzen Blick auf das Datenblatt war klar, dass dieses IC ebenso herausfordernd wie spannend ist! 1 MB Flash, fast 400 K RAM und ein außergewöhnlich niedriger Stromverbrauch ließen uns von Möglichkeiten träumen, die unser alter Uno nicht hatte. Gleichzeitig bedeutete ein 81-ball-BGA mit 0,5-mm-Raster, dass wir unsere Platine auf Vordermann bringen mussten. Die Herausforderungen, denen wir uns stellten und die wir meisterten, werden in einem Blog aus dem Jahr 2019 beschrieben, der unter www.sparkfun.com/news/3122 zu finden ist.

Bevor wir uns mit dem Artemis beschäftigen, sollten Sie sich folgende Tutorials ansehen.

PCB Basics. Was genau ist eine Platine? In diesem Tutorial erfahren Sie, was eine Platine ausmacht und welche Begriffe in der Welt der Leiterplattenherstellung verwendet werden. [1]

Using EAGLE: Board Layout. Teil 2 der „Using Eagle Tutorials“, das das Layout einer Platine nach dem Entwurf eines Schaltplans behandelt. [2]

Bluetooth Basics. Ein Überblick über die drahtlose Bluetooth-Technologie. [3]

ARM Programming. Wie man SAMD21- oder SAMD51-Boards (und andere ARM-Prozessoren) programmiert. [4]

Hardware-Übersicht

Dieser Abschnitt behandelt die technischen Details des Artemis einschließlich des Footprints und der elektrischen Eigenschaften. Wenn Sie schon einmal ein Entwicklungsboard mit dem Artemis-Modul bestückt haben, können Sie diesen Abschnitt eigentlich überspringen und zu den **Besonderheiten** übergehen. Aber: Wann haben Sie das letzte Mal einen Blick in das Innere eines dieser Module geworfen? Werfen Sie besser doch einen Blick auf **Bild 1**!

Apollo3

Der Kern des Artemis-Moduls ist der Apollo3-Controller von Ambiq [5]. Dabei handelt es sich um einen ARM-Cortex-M4F (F steht für Hardware-Gleitkommaoperationen) mit 1 MB Flash und 384 KB RAM. Das Datenblatt ist unter [6] verfügbar.

BLE-Antenne

Der Apollo3 besitzt eingebaute Bluetooth-5.0-Funktionalität. Das Artemis-Modul stellt dazu eine 2,4-GHz-Antenne mit 2 dBi Gewinn zur Verfügung.

Integrierter DC-Abwärtsregler

Der Apollo3 kann mit 1,8...3,6 V betrieben werden. Um ein so breites Spannungsfenster zu ermöglichen, sind zwei DC-Abwärtsregler integriert, die die Eingangsgleichspannung mit einem Wirkungsgrad >80%

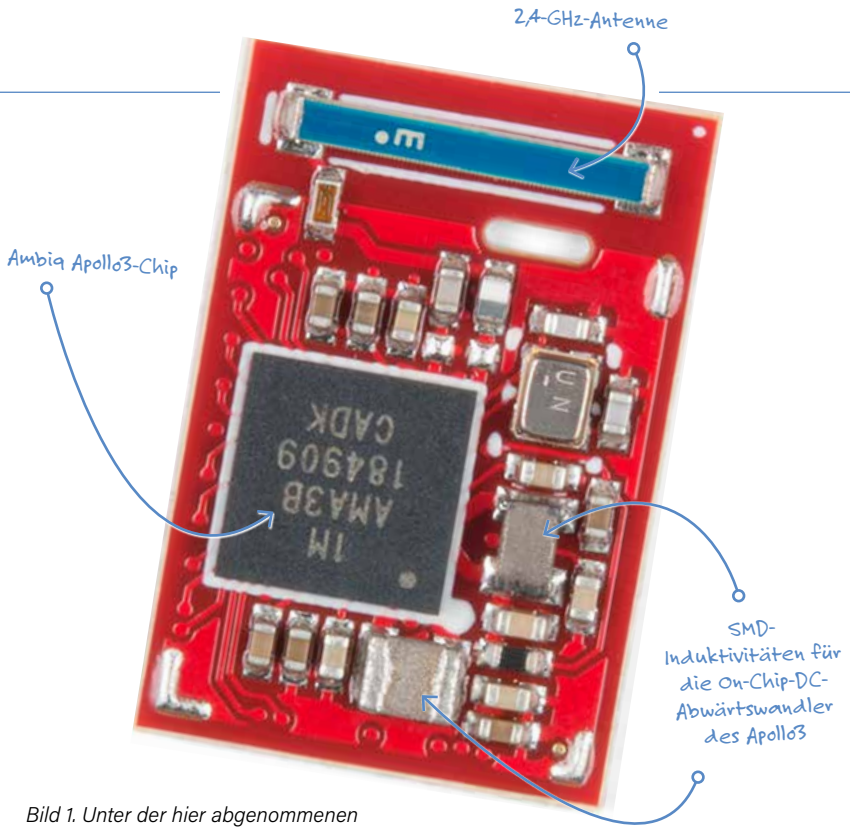


Bild 1. Unter der hier abgenommenen HF-Abschirmung verbergen sich einige bemerkenswerte Interna des Artemis-Moduls.

Bild 2. Klein und leicht: das Artemis-Modul

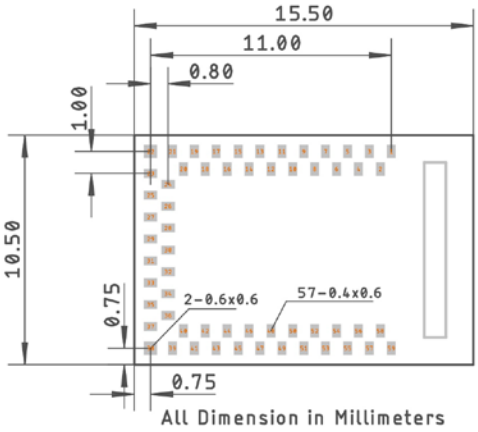


Bild 3. Empfohlener SMD-Footprint für das Artemis-Modul. Ansicht von oben.

auf die Core-Spannung herunterregeln. Das Artemis-Modul enthält zwei Induktivitäten, um eine minimale Leistungsaufnahme zu garantieren.

Abmessungen

Das Artemis-Modul misst 15,5 × 10,5 × 2,3 mm³ und wiegt 0,6 g (**Bild 2**).

Empfohlener Footprint

Das empfohlene Platinenlayout für das Modul ist in **Bild 3** zu sehen. Beachten Sie unbedingt den *Artemis Integration Guide* [7] für die genauen Abmessungen und sonstige Überlegungen. Wenn Sie mit Eagle-PCB arbeiten, klonen Sie einfach eines unserer Open-Source-Hardware-Designs mit Artemis (RedBoard, RedBoard Nano oder RedBoard ATP) und beginnen Sie mit dem Layout Ihrer Platine! Zu Ihrer Information ist in **Bild 4** die Unterseite des Artemis-Moduls abgebildet.

Apollo3 ist ein leistungsfähiger Controller, aber sein 0,5-mm-BGA-Gehäuse erfordert eine vierlagige Platine mit vergrabenen Durchkontaktierungen (buried vias), die mit Epoxidharz verfüllt werden müssen. Dadurch sind die Platinen in „Heimarbeit" schwierig zu produzieren und entsprechend teuer. Wir haben das Artemis-Board entwickelt, um Sie von dieser Last zu befreien.

Das Layout einer Platine für das und mit dem Artemis-Modul wie in **Bild 5** kann auf zwei Lagen mit 8 mil-Leiterbahnen/Abstand erstellt werden. Ein Routen unter dem Modul ist erlaubt, halten Sie aber alle Masseverbindungen vom Antennenbereich fern. Wenn die mechanische Belastung es zulässt, kann die Antenne über den Rand der Leiterplatte hinaus ragen, um den Empfang zu verbessern.

Besonderheiten

Es gibt eine Vielzahl von Funktionen, die in das Artemis-Modul gepackt wurden, und über die wir Ihnen hier einen Überblick vermitteln wollen. Schauen Sie sich aber unbedingt auch die Beispiele an, die in der Arduino-IDE sowie im Ambiq-SDK enthalten sind, um mehr darüber zu erfahren.

Pin-Flexibilität

Das Artemis-Modul verfügt über 48 Interrupt-fähige GPIO-Anschlüsse und eine Reihe weiterer peripherer Funktionen. Die beiden Hardware-UARTs können auf einige andere verschiedene Pads verlegt werden, zudem gibt es 16 völlig unabhängige PWM-Ausgänge. Ein fortschrittlicher 14-Bit-Hochgeschwindigkeits-ADC ist mit zehn Pads verbunden; an sechs Anschlüssen sind entweder SPI- oder I²C-Master verfügbar.

Das ist noch lange nicht alles, denn PDM, SSC und DMA sind ebenfalls über die Hardwareabstraktionsschicht verfügbar. Weitere Informationen finden Sie in der Pin-Funktionstabelle [11], im Apollo3-Datenblatt [12] und im Ambiq-SDK/HAL [13]. Aber lassen Sie sich nicht verwirren, wir zeigen in vielen Beispiele den Umgang mit den verschiedenen Ports und Pins.

Cortex-M4F

„Kopf" des Artemis-Moduls ist natürlich der Apollo3 von Ambiq. Dieses IC nutzt

Bild 4. Rückansicht des Artemis-Moduls.

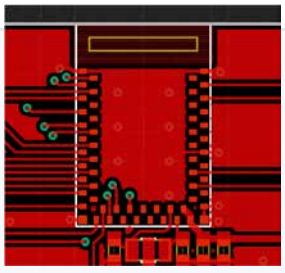


Bild 5. Beispiel für das Artemis-Modul in einem Platinenlayout.

Auf der Artemis-Welle

Schauen Sie sich diese Webseiten an, um einen Überblick über die Entwicklung des und mit dem Artemis zu erhalten:

Artemis-Entwicklung:
www.sparkfun.com/news/3122
Entwicklung eines HF-Shields:
www.sparkfun.com/news/3123
Ein Modul FCC-zertifizieren lassen:
www.sparkfun.com/news/3124
Pressemitteilung:
https://bit.ly/3b2Zpd7
Blog zur Entwicklung eines HF-Shields:
https://www.sparkfun.com/news/3123
Webseite des Ökosystems:
https://www.sparkfun.com/artemis
Artikel von Hackster:
https://bit.ly/38XRX0i

den ARM-Cortex-M4F, der mit 48 MHz und einem optionalen 96-MHz-Burst-Modus läuft. Der leistungsfähige Kern kann sowohl mit GCC als auch mit Keil und IAR programmiert und mit einer Vielzahl von modernen JTAG-Tools fehlerbefreit werden. [14]

BLE

Das Artemis-Modul verfügt über Bluetooth-5.0-Hardware, die bis zu 4 dBm sendet, was eine Übertragungsreichweite von etwa 70 m ermöglichen sollte. Wir haben auch erfolgreiche RSSI-Checks bei weit über 70 m festgestellt.

PWM

Mit 31 PWM-fähigen Pins (von 48 Pins) sollten Sie für alle Aufgaben gerüstet sein! Auf dem „grafischen" Datenblatt [15] und in der Pin-Tabelle ist zu sehen, welche Pins PWM-fähig sind.

Interrupts

Jeder Pin kann als Interrupt konfiguriert werden und den Prozessor aus seinem Tiefschlaf erwecken. Zusätzlich besitzen alle Pins (außer Pin 20) einen internen Pull-Up-Widerstand, der per Software aktiviert werden kann.

14-Bit-ADC

Während der ursprüngliche Uno einen 10-Bit-Wandler hatte, verfügt der Artemis über einen 14-Bit-ADC. Das bedeutet eine Auflösung der Messwerte von bis zu 0…16.383 (statt von 0…1023). Dies ermöglicht eine präzisere Erfassung der Daten von (zum Beispiel) analogen Flex- oder Licht- und Akustik-Sensoren. Der ADC-Bereich reicht von **0…2 V**. Sie können zwar einen Sensor einsetzen, der beispielsweise 0…3,3 V ausgibt, aber er wird nur bis 2 V ausgelesen. Verwenden Sie die Funktion `.setResolution`, um die Messwertauflösung von der Standardeinstellung 10 Bit auf 14 Bit oder irgend etwas dazwischen zu ändern. Außerdem ist der ADC **viel** schneller (bis zu 1,2 MS/s), wodurch mehr Daten in einem Intervall erfasst werden können.

Low Power

Ambiq, der Hersteller des Apollo3, hat jahrelang an etwas geforscht, das sich Sub-threshold Power Optimized Technology (SPOT) nennt. Dies ist eine zugegeben ausgefallene Beschreibung einer stromsparenden Technik, die die zur Anzeige einer Eins oder einer Null notwendigen Logikpegel-Spannungen reduziert. Durch diese Maßnahmen auf Siliziumebene ist es Ambiq gelungen, einen 48-MHz-Prozessor mit weniger als einem halben Milliampere laufen zu lassen! Die permanente Überwachung von Sprachbefehlen, ohne dass BLE oder eine Verbindung zum Internet erforderlich wären, benötigt so nur etwa 6 µA/MHz.

Burst-Modus

Manchmal sind auch 48 MHz nicht genug. Der Prozessor hat die Möglichkeit, in einen 96-MHz-Burst-Modus zu wechseln, in dem interne Berechnungen und Überwachungen in der Hälfte der Zeit durchgeführt werden können.

Externe Pin-Operationen sind auf 48 MHz begrenzt.

JTAG

Moderne Leistungsfähigkeit verlangt nach modernen Debugging-Tools. Der Artemis basiert auf einem Cortex-M4F, der über einen eigenen JTAG-Port zum Debuggen verfügt. Mit einem JTAG-Debugger können Sie Haltepunkte setzen, Register untersuchen und sehen, welche Assembler- und C-Anweisungen ausgeführt werden. Es ist ein Werkzeug, das Sie nicht oft brauchen werden, aber wenn Sie es brauchen, ist es ein Lebensretter.

PDM

Eine der glänzendsten Anwendungen des Artemis ist die schon erwähnte permanente Spracherkennung. Digitale MEMS-Mikrofone sind empfindlicher und einfacher zu bedienen als ihre analogen Großeltern. Das Artemis-Board besitzt einen PDM-Port, der es ermöglicht, bis zu zwei MEMS-Mikrofone entweder als Zweikanal oder in Beamforming-Anwendungen zu verwenden.

Bisher war der Übergang vom Prototyp zum Endprodukt – milde ausgedrückt – umständlich. Artemis ändert das, da das gleiche Modul vom Prototyp bis zur Produktion verwendet werden kann. Wenn man die Leistungsfähigkeit der TensorFlow-Maschinenlernplattform und des Apollo3-Mikrocontrollers von Ambiq hinzunimmt, hat man ein Werkzeug, mit dem die meisten Ingenieure experimentieren wollen.

Nathan Seidle, SparkFun Gründer und Ingenieur

Interne Pull-Ups

Jeder Pin verfügt über einen internen (schwachen) Pull-up, der per Software aktiviert werden kann. Zusätzlich verfügen die als I²C-Ports konfigurierten Pins über per Software wählbare Pull-ups (1,5 kΩ, 6 kΩ, 12 kΩ und 24 kΩ), so dass keine externen SDA- und SCL-Pull-ups erforderlich sind.

Pin-Treiber

Eines der besonderen Merkmale des Artemis-Boards ist die Möglichkeit, die

Premiere für SparkFun

FCC/IC/CE-Zulassung! Das Artemis-Modul von SparkFun hat die Zulassung der Federal Communications Commission (FCC), von Industry Canada (IC) und Conformité Européenne (CE) erhalten. Damit ist es das erste in den USA hergestellte FCC/IC/CE-zertifizierte Open-Source-BLE-Modul auf dem Markt. Mit dieser Zertifizierung ermöglicht das Artemis-Modul Entwicklern die Verwendung desselben Moduls vom Prototyp bis zur Produktion und macht Low-Power-Maschinenlernen in jedem Entwurf zugänglich.

Stromstärke aller GPIOs auszuwählen. Dabei kann als maximaler Strom 2 mA, 4 mA, 8 mA oder 12 mA bestimmt werden. Zusätzlich haben Pin 3 und Pin 36 wählbare High-Side-Schalttransistoren, die mit etwa 1 Ω an die positive Versorgung schalten. Pad 37 und Pad 41 haben wählbare Low-Side-Leistungsschalttransistoren, die ebenfalls mit etwa 1 Ω gegen VSS schalten.

Sicherheit

Der Cortex-M4F im Artemis enthält mehrere Sicherheitsebenen, einschließlich Secure Boot, OTA, Keystorage sowie Inline-Verschlüsselung/Entschlüsselung von externem Flash (wie einer SD-Karte).

Programmierung

Artemis kann über die Standard-JTAG-Schnittstelle oder mit einem seriellen Bootloader programmiert werden. Sie finden einen mit dem CH340 realisierten USB-Anschluss für serielles Bootloading [16] (**Bild 6**) oder einen JTAG-Footprint für fortschrittlichere Programmierung und Debugging wie auf einer Vielzahl von Entwicklungsboards von SparkFun. Für weitere Informationen zur ARM-Programmierung, einschließlich JTAG-Schnittstellen, lesen Sie bitte in unserem ARM-Programmiertutorial. [17]

SparkFun Bootloader

Wir haben einen Bootloader mit flexibler Baudrate entwickelt, der bei jedem Power-On-Reset ausgeführt wird. Was flexible Baudrate genau bedeutet? Nun, der Computer und der Bootloader beginnen die Kommunikation mit 9.600 bps und vereinbaren dann, auf eine schnellere Baudrate zu wechseln, um den Großteil der Binärdaten zu übertragen. Dieses Vorgehen ermöglicht Upload-Geschwindigkeiten von bis zu 921.600 bps, was die Upload-Zeiten erheblich reduziert. Eine flexible Baudrate ermöglicht es aber Computersystemen, die möglicherweise Probleme mit sehr hohen Übertragungsgeschwindigkeiten haben, die Rate zu wählen, die am besten funktioniert. Dieser Bootloader ist die bevorzugte Methode für das schnelle und zuverlässige Hochladen von Sketches und Anwendercode auf den Artemis.

Sobald Sie das Artemis-Zielboard ausgewählt haben, werden beim nächsten Öffnen des Menüs „Werkzeuge“ zusätzliche Menüoptionen angezeigt. Mit den SVL-Baudrate-Optionen (**Bild 7**) können Sie die Upload-Geschwindigkeit ändern. 921.600 bps ist die empfohlene Geschwindigkeit, die neue Sketche extrem schnell hochlädt. Es gibt jedoch einige Plattformen (Linux-Varianten), bei denen der Standard-CH340-USB-zu-Seriell-Treiber bei Geschwindigkeiten über 115.200 bps nicht gut funktioniert. Wenn Sie also Upload-Probleme haben, sollten Sie die Baudrate reduzieren. Weitere Hinweise zu Upload-Problemen finden Sie im Forum unter [18], und ziehen Sie gegebenenfalls ein Upgrade mit Treibern für Mac OSX [19] oder Linux [20] in Betracht.

Wie beim klassischen Arduino Uno, Arduino Mega oder ähnlichen Boards wird der Bootloader durch einen Reset des Boards aktiviert. Ein kleiner 0,1-µF-Kondensator zwischen DTR und Reset genügt, um das Artemis-Board zurückzusetzen und in den Bootloader-Modus zu bringen. Wird innerhalb einer kurzen Zeitspanne (50 ms) keine neue Firmware erkannt, wird der Anwendercode ausgeführt. Wenn Sie sich für elektrotechnische Nischendiskussionen über Dinge wie Bootloader interessieren, können Sie mehr über den Artemis-Bootloader unter [21] lesen.

Bootloader ab Werk

Zusätzlich zum SparkFun-Artemis-Bootloader programmieren wir jeden Artemis mit dem originalen Secure Bootloader (SBL) von Ambiq. Dieser Bootloader eignet sich am besten für Low-Level-Updates mit sicherer, vertrauenswürdiger Herkunft. Der Bootloader wird beim Reset aktiviert, wenn Pin 47 high ist und kommuniziert mit 115.200 Baud. Der Bootloader wartet dann eine unbestimmte Zeit auf neue Binärdaten. Wir stellen sowohl ein Python-Tool als auch eine ausführbare Datei zur Verfügung,

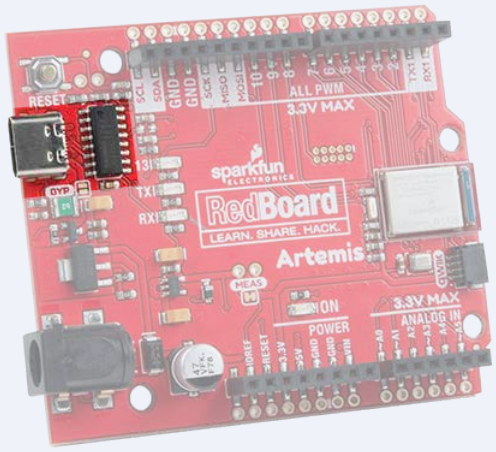


Bild 6. USB-Anschluss am RedBoard-Artemis, für CH340-basiertes serielles Bootloading.

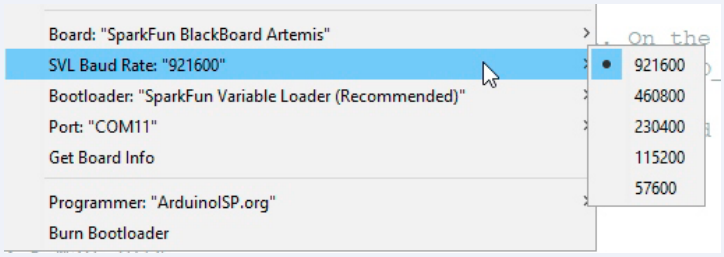


Bild 7. Schnelles Hochladen des Sketches mit eingestellter SVL-Baudrate von 921600.

um mit diesem Bootloader kommunizieren zu können.
Diese Art des Bootloadens unterscheidet sich etwas von den Bootloadern, die Sie vielleicht gewohnt sind. Der STK500-Bootloader, der auf den meisten ATmega328-basierten Arduinos läuft, wird beim Zurücksetzen automatisch ausgeführt, dann wird ein Timeout durchgeführt und dann der Code des Benutzers ausgeführt. Der Artemis-Bootloader ist ähnlich, benötigt aber einen zusätzlichen Pin (den Bootload-Pin), der dabei auf high gehalten werden muss. Um Kosten zu sparen und Komplexität zu vermeiden, haben wir eine einfache RC-Schaltung entworfen, die in Ihrem Entwurf beim USB-zu-Seriell-IC mit dem absoluten Minimum an Steuerpins (der CH340E hat nur RTS) implementiert werden kann und trotzdem die Aktivierung des Werk-Bootloaders ermöglicht. Wenn Sie den SparkFun-Artemis-Bootloader modifizieren oder wenn Sie die Secure-Bootload-Toolchain verwenden wollen oder müssen, kann die Schaltung in **Bild 8** verwendet werden, um mit einem einzigen Pin zu booten (DTR oder RTS wird unterstützt).

Achtung! Wenn Sie die Ambiq-Secure Bootloader-Tools einsetzen, wird der SparkFun-Bootloader überschrieben. Das Artemis-Modul wird dabei zwar nicht beschädigt, aber die schnelleren Upload-Fähigkeiten stehen dann nicht mehr zur Verfügung. Wir raten deshalb davon ab, den Ambiq-Secure-Bootloader für die allgemeine Arduino-Programmierung zu verwenden (**Bild 9**).

Um mit der Ambiq-Bootloader-Toolchain neuen Code auf Ihr Artemis-Modul zu laden, wählen Sie die Option *Ambiq Secure Bootloader* im Menü Arduino Werkzeuge -> Bootloader. Diese Tools modifizieren Ihren Code und statten ihn mit verschiedenen Sicherheits-Headern aus. Der Code wird mit 115.200 bps geladen. Sollte dies fehlschlagen, drücken Sie einfach erneut auf Upload.
Wird DTR (oder RTS) auf low gezogen, wird das Modul zurückgesetzt. Nach 10 ms wird DTR von der Software wieder auf high gestellt. So ist der Bootload-Pin für 100 ms high und der Bootloader kann anlaufen. Das Öffnen einer seriellen Schnittstelle führt dazu, dass DTR auf low geht und das Modul zurückgesetzt wird. Da DTR jedoch während des normalen seriellen Betriebs auf low bleibt, tritt das Modul nicht in den SBL ein, sondern fährt stattdessen mit der Ausführung des SparkFun-Artemis-Bootloaders fort. Wenn Sie es wollen, kann der Bootload-Pin mit einem Taster ausgestattet werden.
Wir haben das Ambiq-Python-Bootload-Tool so modifiziert, dass sowohl DTR als auch RTS zur gleichen Zeit und auf die gleiche Weise angesteuert werden, so dass Sie entweder RTS oder DTR verwenden können, um den Artemis zu booten. Unsere Ambiq-SBL-Tools [22] setzen dann DTR/RTS auf high, um den Werks-Bootloader aufzurufen.

Probleme?

Brauchen Sie Hilfe? Wenn Ihr Artemis nicht wie erwartet funktioniert oder Sie technische Unterstützung oder Informationen benötigen, besuchen Sie bitte die SparkFun-Foren [24] - sie sind der richtige Ort, um Hilfe zu finden und zu erbitten. Wenn es Ihr erster Besuch im Forum ist, müssen Sie ein Konto [24] erstellen, um die Produktforen zu durchsuchen und Fragen zu stellen. Der Weblink [25] führt Sie direkt zu den SparkFun-Artemis-Foren.

Q. Ich habe versehentlich den Ambiq-Bootloader verwendet. Jetzt funktioniert der variable SparkFun-Bootloader nicht mehr. Was kann ich tun?
A. Sie konnten es einfach nicht lassen und haben Code mit dem Ambiq-Bootloader geladen, wie wir in **Bild 9** gezeigt haben. Das ist in Ordnung! Um Ihr Modul wieder mit dem SparkFun-SVL-Bootloader laden zu können, folgen Sie diesen Schritten:

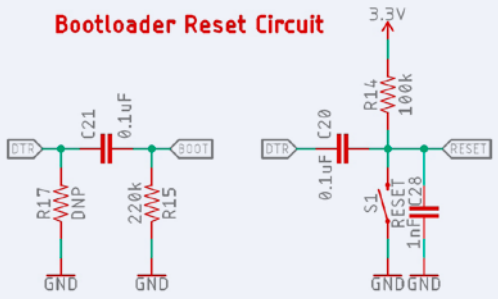


Bild 8. Diese Ein-Pin-Reset-und-Bootload-Lösung für Artemis ist ideal für jeden USB-zu-seriell-Wandler mit freien Steuerpins (CH340, CP210x, FT232, …).

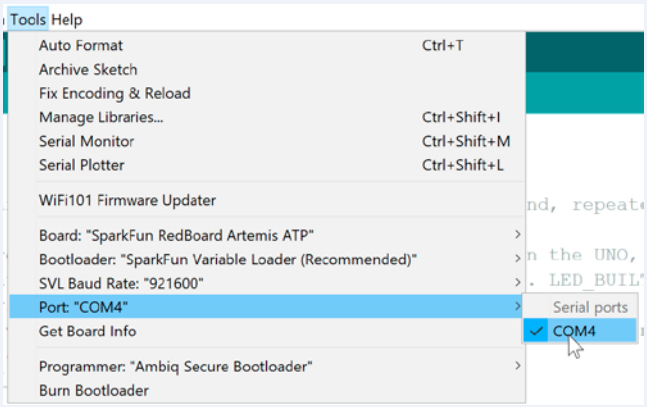


Bild 10. Auswahl des COM-Ports.

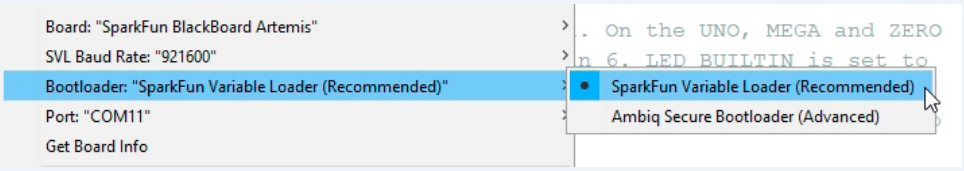


Bild 9. Wählen Sie den Ambiq Secure Bootloader nur, wenn Sie wissen, was Sie tun.

Passende Produkte

Interessiert an den in diesem Artikel erwähnten Produkten?
Elektor und SparkFun haben die passende Lösung für Sie!

SparkFun-Artemis-Modul - Low Power Machine Learning BLE Cortex-M4F

parkFun RedBoard-Artemis

SparkFun RedBoard-Artemis-Nano

SparkFun RedBoard-Artemis-ATP

SparkFun OpenLog-Artemis

SparkFun Edge-Entwicklungsboard - Apollo3 blue

www.elektormagazine.de/esfe-en-artemis

Artemis-Entwicklung mit Arduino

Bringen Sie die leistungsstarken Artemis-basierten Boards (RedBoard Artemis, RedBoard Artemis Nano und RedBoard Artemis ATP) mit dem SparkFun Artemis Arduino Core in weniger als fünf Minuten zum Blinken!

https://learn.sparkfun.com/tutorials/artemis-development-with-arduino

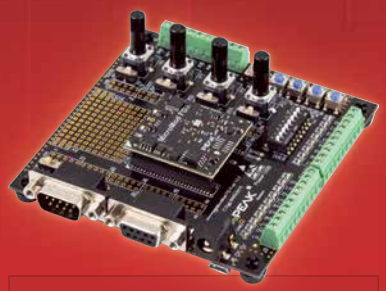
Anleitung für das RedBoard-Artemis-Board

Starten Sie mit dem RedBoard Artemis - alle Funktionen des SparkFun Artemis-Moduls in der gewohnten Uno-Bauform.

https://learn.sparkfun.com/tutorials/hookup-guide-for-the-sparkfun-redboard-artemis

You CAN get it...

Hardware und Software
für CAN-Bus-Anwendungen...



PCAN-MicroMod FD

Universelles Einsteckmodul mit I/O-Funktionalität und CAN-FD-Interface. Erhältlich mit Evaluation-Board für die Entwicklung eigener Anwendungen.

ab 110 €

PCAN-miniPCIe FD

CAN-FD-Interface für PCI Express Mini. Erhältlich als Ein-, Zwei- und Vierkanalkarte inkl. Treiber für Windows und Linux.

ab 240 €

PCAN-GPS

Programmierbares Sensor-modul mit CAN-Anbindung zur Positions- und Lagebestimmung. Entwicklungspaket inkl. Library und Programmierbeispielen.

240 €

Alle Preise verstehen sich zzgl. MwSt., Porto und Verpackung. Irrtümer und technische Änderungen vorbehalten.

www.peak-system.com

PEAK System

Otto-Röhm-Str. 69
64293 Darmstadt
Germany
Tel.: +49 6151 8173-20
Fax: +49 6151 8173-29
info@peak-system.com

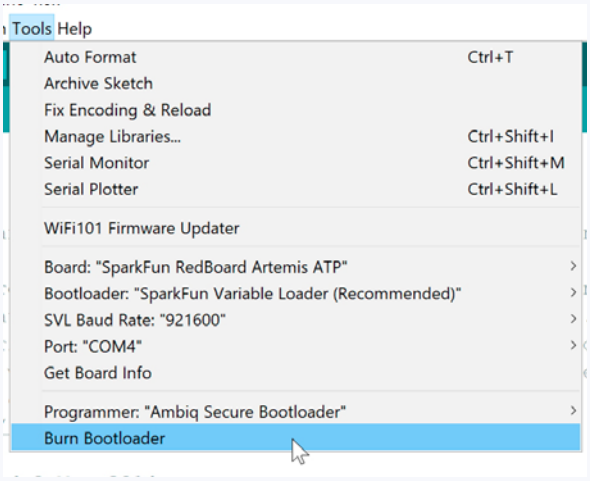


Bild 11. Wählen Sie aus dem Werkzeug-Menü: Bootloader brennen.

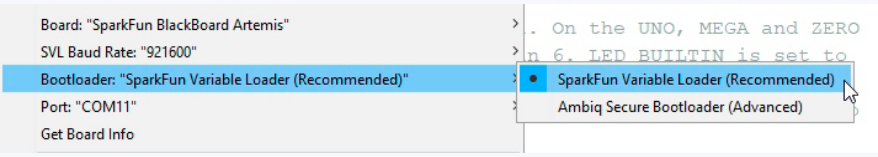


Bild 12. Ja, das ist wieder Bild 9. Verwenden Sie von nun an unbedingt den SparkFun Variable Loader SVL.

Schritt 1: Wählen Sie das richtige Board und den richtigen COM-Port in der Arduino-IDE aus.
Vergewissern Sie sich, dass Sie das richtige Board und den richtigen COM-Port im Werkzeug-Menü ausgewählt haben. In **Bild 10** wird COM 4 gezeigt, aber es kann bei Ihnen auch ein anderer COM-Port sein.

Schritt 2: Wählen Sie im Werkzeuge-Menü „Bootloader brennen“ aus.
Die Auswahl in **Bild 11** veranlasst die Arduino-IDE, den Ambiq-Werks-Bootloader zu verwenden, um den SVL über die serielle Schnittstelle neu zu laden.

Schritt 3: Ändern Sie Ihr Bootload-Werkzeug auf SVL.
Ändern Sie den Bootloader wieder auf SVL (**Bild 12**). Jetzt werden alle Ihre Sketches viel schneller hochgeladen.

Weiterführendes

Die folgenden Links am Ende dieses Artikels sind besonders nützlich für den schnellen Zugriff auf die Ressourcen des Artemis-Boards: [5], [7], [11], [12], [13], [14], [17], [21], [22], [27] und [28].

200685-03

Elektor-Projektseite:
www.elektormagazine.de/esfe-en-artemis

WEBLINKS

[1] PCB Basics: https://learn.sparkfun.com/tutorials/pcb-basics
[2] EAGLE Board Layout: https://bit.ly/3rLBDYW
[3] Bluetooth Basics: https://learn.sparkfun.com/tutorials/bluetooth-basics
[4] ARM Programming: https://learn.sparkfun.com/tutorials/arm-programming
[5] Ambiq: https://ambiq.com
[6] Ambiq Apollo3 datasheet: https://bit.ly/3b8uxb8
[7] Artemis Integration Guide: https://bit.ly/3rLALn8
[8] RedBoard Artemis: https://www.elektormagazine.de/esfe-en-artemis2
[9] RedBoard Artemis Nano: https://www.elektormagazine.de/esfe-en-artemis3
[10] RedBoard Artemis ATP: https://www.elektormagazine.de/esfe-en-artemis4
[11] Apollo3 Pinfunktionen: https://bit.ly/3hHtY9F
[12] Apollo3 Datenblatt: https://bit.ly/2LkjIIh
[13] Ambiq SDK/HAL: https://ambiq.com/apollo3-blue/
[14] JTAG-Tools: https://www.sparkfun.com/categories/tags/jtag
[15] Artemis grafisches Datenblatt: https://bit.ly/39eBG7p
[16] Serial bootloading with the CH340: https://bit.ly/3rNBCnz
[17] ARM Programming Tutorial: https://learn.sparkfun.com/tutorials/arm-programming
[18] Forum-Post: https://bit.ly/3rUDJ8O
[19] Treiber für Mac OSX: https://bit.ly/354YgOb
[20] Treiber für Linux: https://github.com/juliagoda/CH341SER
[21] Der Artemis-Bootloader: https://bit.ly/392ZEIA
[22] Ambiq-SBL-Tools: https://bit.ly/38a1qlM
[23] SparkFun-Foren: https://forum.sparkfun.com/index.php
[24] Einen Forumsaccount anlegen: https://forum.sparkfun.com/ucp.php?mode=register
[25] SparkFun Artemis-Foren: https://forum.sparkfun.com/viewforum.php?f=163
[26] Tutorial Artemis Development with Arduino: https://bit.ly/2XbG8hk
[27] Setting up the Ambiq Apollo3 SDK: https://bit.ly/3ncdbfX
[28] SparkFun Qwiic-Boards: https://www.sparkfun.com/qwiic

TensorFlow ambiq micro

Mit vereinten Kräften:
Artemis war eine großartige Zusammenarbeit mit den Teams von TensorFlow und Ambiq Micro.

Erste Schritte mit dem Qwiic-Ökosystem für schnelles Prototyping

Von Chris McCarty (SparkFun)

Im April 2017 wurden die ersten Qwiic-Boards durch SparkX, der Entwicklungsabteilung von SparkFun, entwickelt und veröffentlicht. Die Ursprünge von Qwiic sind erstaunlich einfach: SparkFun-Gründer Nathan Seidle war es leid, immer die gleichen vierpoligen Stiftleisten einzulöten und dann die vier Drähte zu einem Entwicklungsboard zu führen. Er wollte kleine gepolte Stecker mit farbcodierten Kabeln, die eine Verkettung der Boards und eine einfache I²C-Kommunikation ermöglichen.

Mit nur sieben Produkten hatte das Qwiic-Connect-System bescheidene Anfänge. Seitdem ist die Qwiic-Produktlinie von SparkFun auf über 150 einzigartige Boards, Kits und Steckverbinder angewachsen. Zahlreiche Partner von SparkFun sind diesem Beispiel gefolgt und haben entweder eigene Qwiic-Boards entwickelt oder bieten ihren Anwendern die Möglichkeit, vorhandene Boards an das Ökosystem anzuschließen. Entwicklungsteams, Prototypenbauer und Maker haben Qwiic als ein System angenommen, das die Verdrahtung vereinfacht und viel Lötarbeit überflüssig macht.

Das Qwiic-Connect-System von SparkFun ist ein Ökosystem von I²C-Sensoren, Aktoren, Shields für Mikrocontroller-Entwicklungsboards und Kabeln, die das Prototyping schneller und weniger fehleranfällig machen. Es ist nicht schwer, ein Qwiic-fähiges Board anzuwenden: Der gepolte JST-Stecker mit vier Leitungen verhindert, dass Benutzer versehentlich die SDA- und SCL-Leitungen auf einem Breadboard vertauschen.

Da die meisten unserer Qwiic-Boards mindestens zwei Anschlüsse haben, ist die Verkettung über den I²C-Bus einfach. So können Sie (nach unseren Berechnungen) Milliarden von Kombinationen von Qwiic-Sensoren, Zubehörschaltungen und Entwicklungsboards miteinander verbinden. Müssen Sie Daten eines Umweltsensors mit Standortbestimmung auf einem LCD darstellen? Müssen Sie Ihr eigenes Virtual-Reality-Hand-und-Head-Set bauen? Dies sind nur einige Dinge, die Sie mit Qwiic herstellen können, ohne auch nur ein einziges Teil löten zu müssen.

Wir werden oft nach dem Lieferumfang von Qwiic gefragt. Das Qwiic-Ökosystem ist in sechs übergreifende Kategorien aufgeteilt: Entwicklungsplatinen, Peripheriebausteine, Sensoren, Zubehör, Kits und Kabel. Innerhalb jeder Kategorie gibt es mehrere Unter-

abschnitte, die genauer definieren, was Sie für ein bestimmtes Projekt oder einen bestimmten Aufbau benötigen könnten. Lassen Sie uns einen intensiveren Blick darauf werfen, was jede Qwiic-Kategorie zu bieten hat!

Entwicklungsboards

Qwiic-Entwicklungsboards sind der Einstieg in das gesamte Ökosystem. Die Boards in dieser Kategorie reichen vom bekannten Arduino-R3-Formfaktor mit einem ATmega328 (**Bild 1**) bis hin zu SAMD21/51, RISC-V, SparkFun-Artemis und mehr. Sie können auch die Feather-kompatiblen Thing-Plus-Boards verwenden, um sich in der wunderbaren Welt des IoT umzusehen und die Chipsets ESP32 WROOM und nRF52840 sowie FPGA- und Machine-Learning-Entwicklungstools zu verwenden.

Peripherie

Qwiic-Peripherie, das klingt kompliziert, aber keine Sorge, sie umfasst einfach alles, was an eine Entwicklungsplattform angeschlossen werden kann, zum Beispiel Shields für Arduino (**Bild 2**), HATs oder pHATs für Raspberry Pi (**Bild 3**), Trägerplatinen für micro:mod und mehr. Diese Boards sind perfekt geeignet, um ein Entwicklungsboard zu erweitern, das keine Qwiic-Fähigkeiten hat, so dass Sie nicht jedes Mal einen komplett neuen Arduino oder Thing Plus kaufen müssen, wenn Sie Qwiic verwenden möchten.

Sensoren

Die Qwiic-Sensoren erfassen GPS, Umwelt- und Bewegungsdaten und viele Parameter mehr (**Bild 4**). Mit Qwiic-Sensoren können Sie einen kompletten Satz von Bildgebungsplatinen verketten, um mit IMUs ein breiteres Lichtspektrum als das eines Menschen zu sehen - perfekt für Robotik- und Drohnenprojekte.

Zubehör

Das Qwiic-Zubehör ist eine Art Sammelbegriff: Anzeige von Informationen, die Stromversorgung und das Hinzufügen von Eingabe-/Ausgabeoptionen zur Steuerung eines Projekts (**Bild 5**) fallen in diese Kategorie. Es gibt dabei so viele Optionen, dass man sie hier gar nicht aufzählen kann: einen MP3-Trigger, Solid-State-Relais-Kits mit hoher Strombelastbarkeit, transparente (oder undurchsichtige) OLED-Bildschirme und und und.

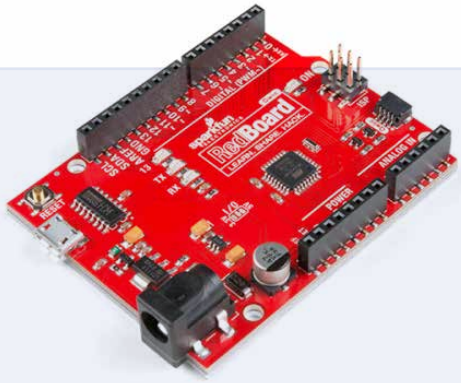


Bild 1. RedBeard ist Qwiic-kompatibel und besitzt den bekannten Arduino-R3-Footprint.

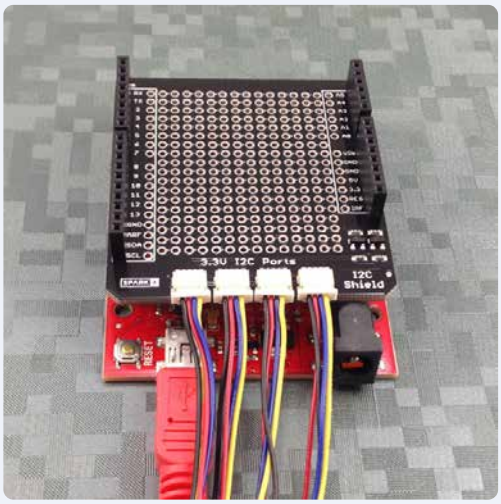


Bild 2. Uno-Style-Shield mit Qwiic

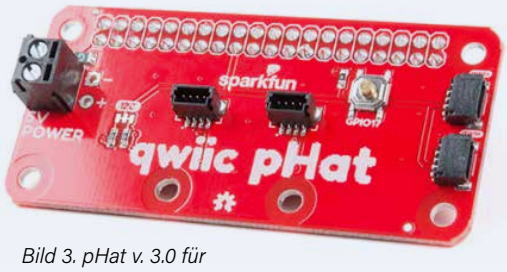


Bild 3. pHat v. 3.0 für den Raspberry Pi.

Bild 4. Qwiic-Sensorik mit dem GPS-RTK-Dead-Reckoning-Breakout ZED-F9.

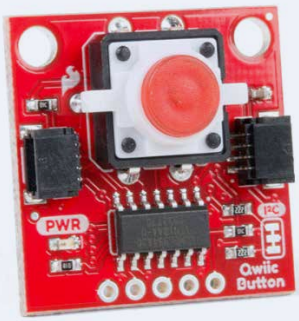


Bild 5. I/O-Funktionalität inklusive: Qwiicc-kompatible Tasterplatine.

Bausätze

Bei all den vielen Möglichkeiten kann es sein, dass man den Wald vor lauter Bäumen nicht sieht: Manche Leute brauchen halt ein Sprungbrett, um mit etwas anzufangen, und da sind die Qwiic-Kits hilfreich. Von der Entwicklung über die Robotik bis hin zum Raspberry Pi - egal, was Sie brauchen, es gibt ein Qwiic-Kit, mit dem Sie beginnen können, Ihre aktuellen Möglichkeiten zu erweitern oder ein geplantes Projekt voranzubringen (**Bild 6**).

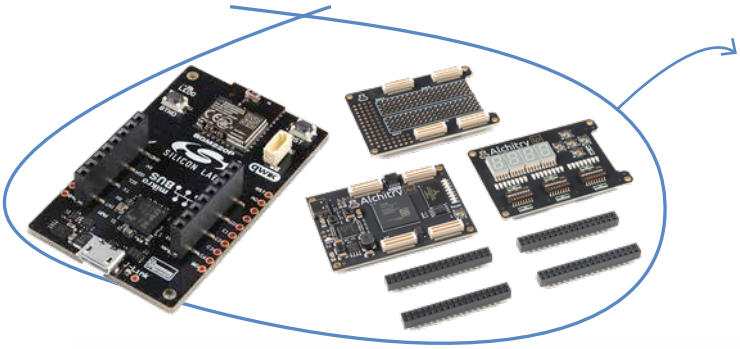
Kabel

Diese Kategorie ist ebenso einfach wie umfassend. In dieser Kategorie finden Sie eine Vielzahl von Kabeln in verschiedenen Längen (**Bild 7**) sowie Adapter, um Qwiic mit anderen Verbindertypen zu koppeln. Zusätzlich enthält diese Kategorie die Anschlüsse, die Qwiic zu dem machen, was es ist. Wenn Sie also Ihr eigenes Qwiic-fähiges Board erstellen wollen, ist dies die richtige Kategorie für Sie!

Ausblick

Wir hier bei SparkFun sind sehr stolz auf dieses Ökosystem, das wir geschaffen haben. Es ist ein einfacher und schneller Weg, um I²C-Optionen einer Schaltung hinzuzufügen, ohne dass es zu Problemen kommt. Das Qwiic-Ökosystem von SparkFun ist gerade einmal drei Jahre alt und gedeiht weiter und beeinflusst unsere Freunde und Partner in der globalen Maker-Community. Wir können es kaum erwarten zu sehen, wie es weiter wächst und sich entwickelt.

200691-02



Passende Produkte

Suchen Sie die in diesem Artikel erwähnten Artikel? SparkFun und Elektor haben die passenden Produkte für Sie!

www.elektormagazine.de/esfe-en-qwiic1
Qwiic Starter Kit für Raspberry Pi (Bild 6)

www.elektormagazine.de/esfe-en-qwiic2
Qwiic Cable Kit (Bild 7)

Partner, die Qwiic-Verbinder verwenden

Die Unterstützung, die wir seit der Einführung des Qwiic-Ökosystems von anderen Firmen erhalten haben, ist erstaunlich. Das BGM220-Explorer-Kit von Silicon Labs, das deren MikroBUS-Peripherie nutzt, die FPGA-Produktreihe von Alchitry, die komplette Sensorik von Zio aus Hongkong oder sogar das neue RA6M4-Entwicklungskit von Renesas Electronics: Jeder dieser Hersteller produziert Boards, die standardmäßig mit Qwiic-Anschlüssen ausgestattet sind. Wir sehen immer mehr Partner in der Industrie, die diesen Standard des einfachen I²C-Anschlusses übernehmen, einschließlich unserer Freunde bei QuickLogic und Circuit Dojo, die ebenfalls bald Produkte in Zusammenarbeit mit uns herausbringen.

Bild 6. Qwiic-Starterkit.

Bild 7. Qwiic-Kabel-Kit.

sparkfun
START SOMETHING

Qwiic®

Prototyping mit I²C war nie einfacher

Das Qwiic-Connect-**Ökosystem** von SparkFun verwendet **4-polige JST**-Verbinder zum schnellen Anschluss von Sensoren, LCDs, Relais und mehr an Entwicklungsboards.

Prototyping leichtgemacht

Schneller Anschluss von Sensoren, Boards und mehr in einer immer weiter wachsenden Produktpalette. Wechseln oder fügen Sie ihrem Projekt Komponenten hinzu – ganz nach Ihren Wünschen. Einfach einstecken und weitermachen!

Mehr Informationen, Tutorials und kaufen bei

www.elektormagazine.de/qwiic

elektormagazine.de/posters

elektor
design > share > sell

Unter der Lupe: Das DIY-LiPo-Supercharger-Kit von GreatScott! und Elektor

Von Alex Wende (USA)

Löten gehört zu den essentiellen Fähigkeiten, die jeder Elektronik-Liebhaber früh erlernt. Bausätze, bei denen man löten muss, überbrücken die Kluft zwischen dem Aufbau einer Schaltung auf einem Breadboard und dem Entwurf einer eigenen Platine. Da das Lernziel eines Bausatzes darin besteht, das Löten zu erlernen, sind diese Bausätze oft recht einfach gehalten. In der Schaltung blinken ein paar LEDs oder sie macht irgendwelche Geräusche. Gibt es aber auch Bausätze, die etwas wirklich Nützliches hervorbringen? Werfen wir einen Blick auf das DIY-LiPo-Supercharger-Kit, das der Elektroniker und YouTuber GreatScott! in Zusammenarbeit mit Elektor entwickelt hat und erfahren Sie, warum ich den Bausatz empfehle.

Im Gegensatz zu den meisten Elektronik-Bausätzen lehrt das DIY-Li-Po-Supercharger-Kit den Umgang mit oberflächenmontierten Bauteilen, kurz SMDs. Nach dem Zusammenbau erhält man eine Schaltung, die die typische Klemmenspannung von etwa 3,7 V eines einzelnen Lithium-Ionen- oder Polymer-Akkus umwandelt, entweder auf eine Spannung von 5 V bei einem Strom von bis zu 1,52 A oder aber 12 V bei einem Strom von maximal 0,76 A.

Aber das ist nicht alles! Da ein Akku irgendwann wieder aufgeladen werden muss, stellt das Kit eine spezielle Ladeschaltung mit einem maximalen Ausgangsstrom von 1 A zur Verfügung. Die Energie bezieht das Supercharger-Kit entweder über ein USB-C-Anschlussadapter oder über ein Labornetzgerät, das an den +5 V/Masse-Platinenklemmen angeschlossen wird. Da sich dieses Kit an Einsteiger richtet, verfügt es über einige Schutzmaßnahmen für den Akku: Verpolungsschutz, Unterspannungs- und Überspannungsschutz sowie Kurzschlussschutz am Ausgang der Platine.

Erste Eindrücke

Abgesehen von den benötigten Werkzeugen, das heißt Lötzinn, Lötkolben und Pinzette, liegt dem Bausatz alles bei, was man braucht. Das Handbuch ist entweder über eine URL oder einen QR-Code zugänglich, zusammen mit dem Platinenlayout und einer Übersicht der zu verlötenden Teile. Auf der Platine sind bereits einige Teile angelötet, namentlich alle ICs und sogar der USB-Verbinder auf einer kleinen Breakout-Platine (**Bild 1**), die man dann leicht an die viel größeren Pads am Rand auf die Hauptplatine löten kann.

Jeder der Widerstände ist mit Zahlen bedruckt, die den Wert symbolisieren, aber da es keine solchermaßen bezeichneten Kondensatoren und LEDs gibt, sind sie in individuellen Tütchen mit Wert-Aufdruck untergebracht. So hält man leicht Ordnung beim Bestücken!

SMD-Teile können ziemlich klein sein und auf einen Anfänger einschüchternd wirken. Die kleinsten Teile im Bausatz sind die Widerstände und Kondensatoren, die jedoch in etwa die gleiche Größe haben wie bedrahtete 0,25-W-Widerstände, wie man sie üblicherweise für Breadboard-Schaltungen und in Bausätzen für Einsteiger verwendet: die perfekte Größe, um das SMD-Löten zum ersten Mal auszuprobieren (**Bild 2**).

Die Anleitung beginnt wirklich gut, indem sie erklärt, wie die Schaltung funktioniert und wofür die einzelnen Bauteile zuständig sind. Da es wahrscheinlich das erste Mal ist, dass man SMD-Bauteile lötet, gibt es eine kurze Anleitung, wie man die Nummerncodes auf den Widerständen liest, damit man weiß, wohin jedes Bauteil gehört. Nachdem die benötigten Werkzeuge besprochen wurden, gibt es einige Bilder und Hinweise zur Reihenfolge, in der die Bauteile verlötet werden sollen. Für die „visuell orientierten" Elektronik-Einsteiger hat GreatScott! ein Video [1] produziert, das nicht nur einen detaillierten Überblick über die Platine, sondern auch eine Anleitung zum SMD-Löten enthält (**Bild 3**).

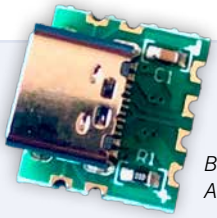


Bild 1. Das leicht zu verlötende USB-C-Adapterboard aus dem Bausatz.

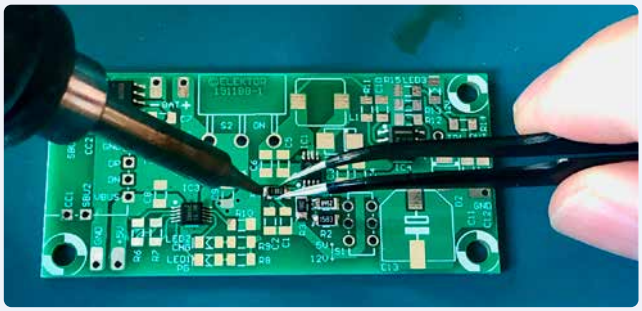


Bild 2. Anlöten der Widerstände auf der Platine.

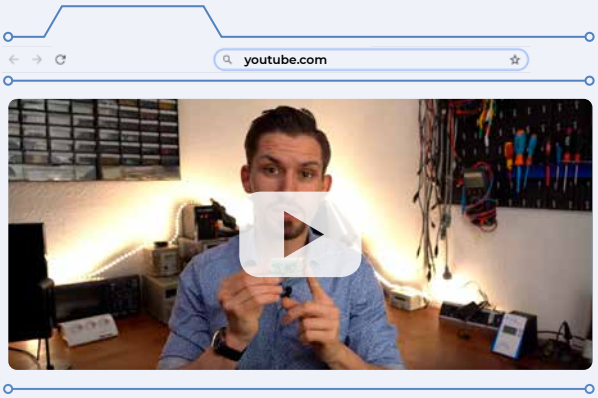


Bild 3. Sehen Sie GreatScott! bei der Präsentation des DIY-LiPo-Supercharger-Kits auf YouTube.

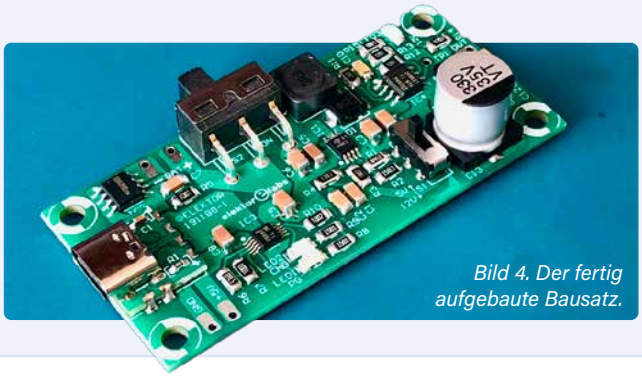


Bild 4. Der fertig aufgebaute Bausatz.

Testen des Endprodukts

Nachdem ich den Bausatz zusammengebaut hatte (**Bild 4**), schloss ich zuerst das Ladegerät an, um den Akku vollständig zu laden. Als ich alles angeschlossen hatte, leuchteten sowohl die Betriebs- als auch die Lade-LED auf. Sobald der Akku aufgeladen war, konnte der Test des Boards beginnen.

Der erste Schritt war, zu überprüfen, ob ich richtig gelötet hatte. Zum Glück zeigte mir mein Messgerät direkt ziemlich genau 5 V und 12 V an. Schaltnetzteile sind praktisch, weil sie nicht nur klein, sondern auch effizient sind, aber wenn sie nicht richtig konstruiert sind, können sie am Ausgang auch ziemlich rauschen. Doch das war beim Kit nicht der Fall; eine Reihe von Tests zeigte, dass das erzeugte Rauschen selbst unter Volllast weniger als 40 mV_{pp} betrug, bei normaler Beanspruchung war es viel weniger.

Bei jeder Art von Spannungsregler ist der größte - und leider unvermeidliche - Feind die Wärme. Ich war überrascht, wie effizient die Schaltung war, vor allem, weil sie die Batteriespannung anhebt, anstatt sie zu senken. Nur bei starker Belastung ist etwas Abwärme zu spüren. Bei kontinuierlicher Belastung würde ich deshalb bei 5 V einen maximalen Strom von etwa 1,3 A und bei 12 V von 0,4 A entnehmen. Aber bei kurzfristigen Belastungen ist das Board durchaus in der Lage, die versprochenen 1,52 A beziehungsweise 0,76 A zu liefern.

Fazit

Die Entwickler der Platine haben sich sehr auf den typischen Einsteiger konzentriert. Die SMD-Bauteile, die beim ersten Mal (zu) schwierig zu löten sein könnten, waren bereits auf der Platine verlötet. Die Teile, die gelötet werden mussten, waren groß genug, selbst für das „erste Mal". Der Bausatz ist so aufgebaut, dass es kein Rätselraten über die Bauteilwerte gab. Es mussten paar Dutzend Bauteile gelötet werden, aber dank der interaktiven „Click-and-locate"-Stückliste auf der Elektor-Website war es nicht nötig, die Platine umständlich nach „R7" oder „C3" abzusuchen. [2]

Was mir persönlich an diesem Bausatz am besten gefallen hat, ist die Tatsache, dass er nach dem Zusammenlöten zu etwas geworden ist, das ich tatsächlich sinnvoll benutzen kann. Ein batteriebetriebenes Projekt braucht nämlich oft eine geregelte Stromquelle. Für ein LED-Streifen-Projekt oder etwas mit ein paar kleinen Motoren ist die Möglichkeit, die Batteriespannung auf 5 V oder gar 12 V zu erhöhen, wirklich praktisch. Außerdem würde ich mir wegen der gut durchdachten Schutzschaltung, der Ladeschaltung und des Netzschalters eher mehr Sorgen um die Schaltung machen, die ich an den Supercharger angeschlossen habe, als um das Board selbst.

200693-02

Passende Produkte

Sie suchen das in diesem Artikel erwähnte Produkt? Elektor hat es!

- **DIY-LiPo-Supercharger-Kit (von GreatScott!)**
  www.elektor.de/diy-lipo-supercharger-kit-by-greatscott

WEBLINKS

[1] DIY LiPo SuperCharger Kit Video: https://youtu.be/6LxRnf6sQNQ
[2] DIY LiPo SuperCharger Kit Produkt und Stückliste: https://www.elektormagazine.de/articles/diy-lipo-supercharger-kit

Unvergessliche Elektronik

aus der Geschichte von SparkFun

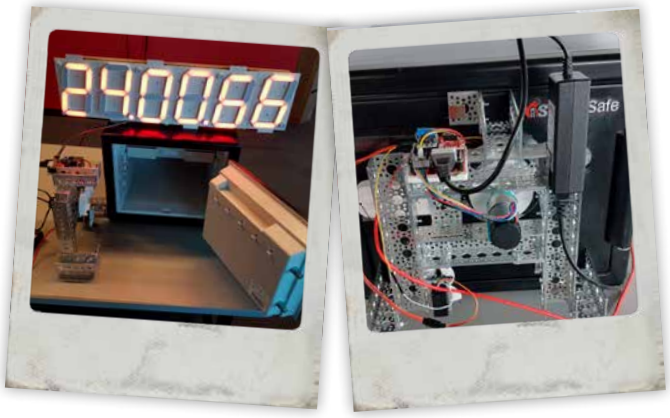
Von Jahnell Pereira

Nostalgie! Seit der Gründung SparkFun im Jahr 2003 sind verrückte Produkte, ausgefallene Projekte und eine schrullige Kultur unsere Markenzeichen. Wir haben für diese Ausgabe von Retronics einige SparkFun-Veteranen danach befragt, was ihnen in den Sinn kommt, wenn sie an bemerkenswerte Elektronik aus der SparkFun-Vergangenheit denken.

2005: Das berüchtigte Port-O-Rotary-Telefon

Anmerkung von Nathan Seidle (SparkFun-Gründer), 17. Januar 2005: „Ja, Sie haben richtig gelesen, Port-O-Rotary. Wir haben uns in ein Wählscheibentelefon gehackt. Wäre es nicht cool, ein Telefon mit Drehscheibe zu sehen, das klingelt, ohne dass Kabel angeschlossen sind? Es könnte die Leute mehr als verwirren!"

Das Port-O-Rotary ist gleich doppelt „retro". Wir haben dazu nicht nur ein fast 50 Jahre altes Wählscheibentelefon verwendet, sondern es auch Technologie aus dem Jahr 2005 verwendet, um es in ein Mobiltelefon zu verwandeln, zum Beispiel mit dem Modul *GM862 Quad Band Cellular* von Telit, einem kundenspezifischen 18-Pin-PIC-Entwicklungsboard mit einem 16F88, einer Drei-Band-Mobilfunkantenne, einer 3,7-V-Polymer-Lithium-Ionen-Batterie und einfachen AT-Befehlen. Dieses verrückte Projekt hat so viel Aufmerksamkeit erregt, dass wir es tatsächlich in ein Produkt verwandelt haben. Und - ja - es wurde verkauft!



2017: Safeknacker-Roboter

Als Weihnachtsgeschenk schenkte die Partnerin von Nate ihm einen alten Safe, den sie in einer Kleinanzeige gefunden hatte - er war besonders billig, weil niemand die Kombination kannte. Klassische Tresortechnik trifft SparkX: Unser hochexperimentelles Team hat neue Technik verwendet, um den alten Safe ohne menschliches Zutun in 40 Minuten und 42 Sekunden zu knacken. Diese neue Technik bestand aus einem autonomen Selbstbauroboter, der mit dem RedBoard, einigen Servos, einem Motor-Controller, einem Stromsensor, einem Summer, einem Rahmen aus Actobotics-Teilen und einem roten „Go"-Knopf ausgestattet war. Ein vollständiges Tutorial finden Sie im Internet unter [1]. Der Tresor war leer – leider, aber eigentlich war das egal.

Ein paar Monate nach diesem grandiosen Erfolg ging das Team zur DEF CON 25, um das System live auf der Bühne vor rund 800 Hackern zu demonstrieren. Wir besorgten vor Ort einen neuen Safe (mit dem alten konnten wir schlecht reisen), stellten ihn auf die Bühne und hatten keine Ahnung, ob unser Roboter funktionieren würde. Er funktionierte, in weniger als 25 Minuten war der Safe offen! Können Sie sich einen Raum voller Hacker vorstellen, die fast 25 Minuten lang schweigend ausharrten, während der Roboter seine Arbeit erledigte?

2009-2018: Das Jahrzehnt der autonomen Fahrzeuge

Der Wettbewerb AVC begann mit einer Wette - es ging darum, ein Fahrzeug zu bauen, das autonom um das SparkFun-Gebäude navigieren kann, was im Jahr 2009 keine einfache Aufgabe war. Doch die SparkFun-Community stellte sich der Herausforderung! Im ersten Jahr nahmen 16 Fahrzeuge am Wettbewerb teil. *DIY Drones*, angeführt von Chris Anderson, belegte den 1. Platz und absolvierte den Parcours in 36 Sekunden. Mindestens eine Drohne landete in einem sehr hohen Baum (ja - es gab Luftfahrzeuge) und jeder machte entzückende Erfahrungen. Da es der erste AVC überhaupt war, war diese Veranstaltung sehr experimentell, und niemand konnte vorher abschätzen, was zu erwarten war. Sowohl die Mitarbeiter als auch die Teilnehmer haben viel gelernt.

Im Jahr 2018, beim letzten AVC, nahmen fast 200 Teams mit fast 500 Teilnehmern teil. Es gab sowohl den Parcours für autonome Fahrzeuge als auch eine Arena für „Kampfroboter". Im Laufe der Jahre nahm das Fachwissen der Wettbewerber schnell zu und neue Technologie, insbesondere LiDAR und andere Sensoren, wurde leichter verfügbar. Dies veranlasste SparkFun, den Parcours schwieriger zu gestalten, mit beweglichen Hindernissen, Rampen, verrückten Kurven und allem, was sich unsere Ingenieure sonst noch ausdenken konnten. In diesen Jahren entwickelten sich ähnliche Wettbewerbe an vielen Orten und der SparkFun-Contest war nur noch einer unter vielen – zumindest fühlte es sich so an. So ging das zehnjährige Abenteuer in die SparkFun-Geschichte ein.

Legendäre Projekte, Streiche und andere seltsame Dinge

2006: Roomba Remote Tilt Controller. Ursprünglich öffnete iRobot die Roomba-Plattform für Hacker, was alles war, was SparkFun brauchte, um einige Roomba-Entwicklungstools und einen Tilt-Controller zu entwickeln. Der Tilt-Controller RooTilt ist nur eine Verschmelzung von SparkFun-Spielzeugen. Wir haben einen drahtlosen Bluetooth-Beschleunigungssensor genommen, die Firmware ein wenig optimiert und ihn mit der drahtlosen RoboDynamics-RooTooth-Verbindung dem Roomba-Staubsauger hinzugefügt.

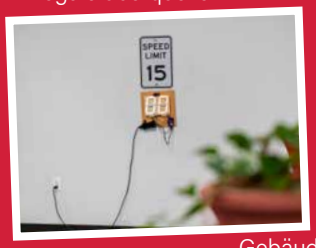
2006: 12 ft GPS Wall Clock. Das Projekt wurde größer und größer. Wir haben einige LED-Lichtleisten mit einem einfachen Controller und einem GPS-Empfänger kombiniert, um eine sehr große Uhr zu bauen, die sich selbst stellt, Stunden, Minuten und Sekunden anzeigt und auf 100 ns genau ist.

2007: Giant NES Controller. Ja! Wir tauchten auf der Maker Faire mit einem funktionstüchtigen, 40 kg schweren und 1,5 m breiten Nintendo-Controller auf.

2010: Antimov-Wettbewerb. Basierend auf den drei Robotergesetzen von Isaac Asimov forderte der Antimov-Wettbewerb von den Teilnehmern, einen Roboter zu entwerfen, der die Gesetze der Robotik bricht (außer dem Verletzen von Lebewesen). Der Roboter sollte eine triviale Aufgabe auf die ineffizienteste und mühsamste Weise erledigen und sich dabei selbst vernichten. Erstaunlicherweise fanden sich Teilnehmer an dem Wettbewerb.

2012-2013: National Tour. Die SparkFun National Tour hatte nur ein Ziel - unsere Leidenschaft für Elektronik mit Schülern und Lehrern im ganzen Land zu teilen. Das Team stieg in das SparkFun-Wohnmobil und reiste ein Jahr lang durch das Land - mit Zwischenstopps überall dort, wo wir Elektronik-Workshops abhalten konnten.

2013 : Einführung des Dumpster Dive. Prototypen, Entwicklungsmuster, Kundenrücksendungen und einfach altes Zeug, das schon viel zu lange in den Regalen liegt: Um unsere elektronischen Überschüsse loszuwerden, hatte jemand im Jahr 2013 eine großartige Idee. Da das meiste davon noch brauchbar war, wollten wir es nicht wegwerfen, sondern in die Hände der Menschen geben. So war der Dumpster Dive Sale war geboren. Die Leute kaufen geheimnisvolle Kisten, die mit elektronischen Dingen gefüllt sind. Erst beim Auspacken erfährt der glückliche Käufer, was drin ist. Wir machen das immer noch von Zeit zu Zeit, wenn unsere Regale überquellen.



2015: SparkFun Speed Trap. Diese Radarfalle, ausgestattet mit dem Large Digit Driver, dem RedBoard und LiDAR, haben wir tatsächlich in unserer Zentrale aufgestellt. Die Gäste, meist Leute, die das Gebäude besichtigen, laufen so schnell sie können darauf zu, um zu sehen, wie schnell sie sind. Dabei rennen sie oft gegen die Wand, eine kleine Gefahr, aber ein großer Spaß!

2017: Skimmer Scammer App. Das SparkX-Team arbeitete mit einer lokalen Strafverfolgungsbehörde in Colorado zusammen, um betrügerische Hardware, die in/an Zapfsäulen zum Lesen von Kreditkartendaten verwendet wird, zurückzuentwickeln. Es stellte sich heraus, dass diese Hardware von Betrügern landesweit verwendet wird und über Bluetooth erkannt werden kann. Das SparkX-Team schrieb eine kostenlose App, um Leute zu warnen, wenn ein Kreditkarten-Skimmer an der Zapfsäule entdeckt wurde. Die App wurde weit über 200 K Mal heruntergeladen. Leider machte ein Android-Update die App im Jahr 2019 kaputt und wir beschlossen, sie wieder zu entfernen.

2018: Giant Joystick. Als Microsoft im Jahr 2018 den Xbox Adaptive Controller veröffentlichte, waren wir erstaunt und begeistert. Gibt es einen besseren Weg als ein technisches Review und ein groß(artig)es Projekt, unsere Community darüber zu informieren? Da Microsoft es ermöglichte, dass der Controller mit 3rd-Party-Zubehör funktioniert, baute das SparkFun-Team einen gigantischen Joystick, der in etwa auf den alten Atari-Joysticks basiert. Durch die Kombination einiger kreativer Konstruktionen mit einem Riesenknopf und einigen Mikroschaltern kam dieses Projekt zustande.

200695-02

WEBLINK

[1] Safe Cracking Robot tutorial: https://learn.sparkfun.com/tutorials/building-a-safe-cracking-robot

Perfektes Einparken mit LiDAR

Von Rob Reynolds (USA)

Abstandssensoren sind meist das erste, mit dem sich ein Maker eines autonomen Roboters beschäftigt. Aber gibt es nicht auch andere Verwendungsmöglichkeiten dafür? Es gibt sie! Von Halloween-Requisiten über Musikinstrumente bis hin zu Parkhaus-Leitsystemen, all dies kann mit Abstandssensoren gebaut werden.

Die Ampel im Haus erspart den Beulendoktor

Ich hatte noch nie einen Tennisball oder irgendetwas anderes in meiner Garage befestigt, das mir genau anzeigt, wann ich mein Auto beim Einparken in die Garage stoppen soll. Aber jetzt beginnt mein Kind mit dem Autofahren und aus Sorge um mein heilig's Blechle denke ich verstärkt über eine solche Sicherheitsmaßnahme nach. Nun, ich spiele kein Tennis, aber ich spiele mit der Elektronik, also werde ich das nutzen, was ich zur Verfügung habe. Und nein, ich werde keinen Raspberry Pi an meine Garagendecke hängen. Mein Plan sieht einen Abstandssensor und eine kleine Verkehrsampel vor, die bei genügend Platz grün leuchtet, die mit gelbem Licht dem Fahrer signalisiert, dass der „Freiraum" zunehmend begrenzt ist und mit rot zeigt, dass das Auto die idealen Parkposition erreicht hat. Ich werde eine einfache Ampel als Anzeige entwerfen und sie 3D-drucken, damit sie auch ästhetisch ansprechend ist!

Welcher Sensor ist der richtige?

Bei der großen Auswahl an Abstands- und Näherungssensoren (**Bild 1**) stellt sich die Frage, welcher der geeignete Typ für das Projekt ist. Es gibt eine Reihe von Parametern wie Reichweite, Auflösung, Schnittstellentyp, Aktualisierungsrate, Strombedarf und Kosten. Sie müssen entscheiden, welche Eigenschaften für Sie und Ihr Projekt wichtig sind (einfacher Anschluss durch I²C) und welche weniger (Aktualisierung mit 635 Hz). Eine Einzelgarage weist üblicherweise eine innere Länge von 5...8 m auf. Obwohl man eine solche Entfernung gut mit einem unser XL-MaxSonar-Ultraschallmesser ausloten könnte, habe ich mich für das LiDAR-Modul aus dem TFMini-Micro-Kit [1] entschieden.
Der LiDAR-Sensor läuft mit einer Betriebsspannung von 5 V, kommuniziert aber mit 3,3 V. Glücklicherweise enthält das Kit eine kleine, nützliche Boost-Platine: Möchte man den Sensor an eine 5-V-Controllerplatine anschließen, so erübrigt sich durch den Booster eine Logikpegel-Wandlung, soll dagegen eine 3,3-V-Controllerplatine verwendet werden, die Notwendigkeit einer doppelten Stromversorgung. Mit dem Sensor-Kit und einem Sparkfun-RedBoard mit Qwiic-Anschluss hat man also ein unkompliziertes Plug-and-Play-Design in den Händen (**Bild 2**). Fügen Sie ein paar superhelle LEDs mit Vorwiderständen und eine Stromversorgung hinzu, dann kann es sofort losgehen!

Qwiic-System = schnell und einfach mit I²C

Das Qwiic-System macht einen solchen Aufbau wirklich sehr einfach. Im Sensor-Kit sind zwei Kabel enthalten. Ein Kabel führt vom Modul zum Boost-Board, das andere vom Boost-Board zum Qwiic-Anschluss am RedBoard. Die grünen, gelben und roten LEDs sind an den Pins 8...10 angeschlossen. Für den Code habe ich einfach den Sketch *LidarTest.ino*, den Sie auf der „Hookup"-Seite des Modul-Kits [2] finden, ein wenig angepasst. **Listing 1** zeigt das angepasste Programm. Ich habe auch eine kleine Verkehrsampel entworfen, deren 3D-Druckdateien Sie genau wie den Arduino-Sketch von meinem Github-Repository herunterladen können. [3]

WARNUNG! Aufgrund des hohen Strombedarfs des TFMini und des Power-Management-Systems des SparkFun-RedBoards wird Ihr Projekt abstürzen, wenn Sie versuchen, es über die Klinkenbuchse mit Strom zu versorgen. Die grüne LED wird dauerhaft leuchten und der Fahrer gegen die Wand fahren – buchstäblich! Die Stromversorgung über den microUSB-Anschluss funktioniert hingegen problemlos. Also, auch wenn die große, schöne Klinkenbuchse Sie anlächelt und lockt - geben Sie der Versuchung nicht nach!

Machen Sie ihr eigenes Ding!

Der Aufbau der Schaltung ist so schnell und so einfach wie nur möglich gehalten. Aber, muss das wirklich sein? Sie haben den Sketch, Sie haben die STL-Dateien [3], also modifizieren Sie sie:

- Fügen Sie einige Tasten hinzu, um die Einstellung des optimalen „Bremswegs" zu erleichtern
- Entwickeln und bauen Sie ein schöneres Gehäuse
- Nehmen Sie mehr LEDs für mehr Sichtbarkeit und Sicherheit

Wenn Sie wirklich ehrgeizig sind, könnten Sie sogar das RedBoard gegen einen Raspberry Pi tauschen, dieses Projekt in Python programmieren und einen Monitor für mehr visuelles Feedback hinzufügen. Viel Spaß beim Hacken, Freunde!

200698-03

-weiter auf der nächsten Seite-

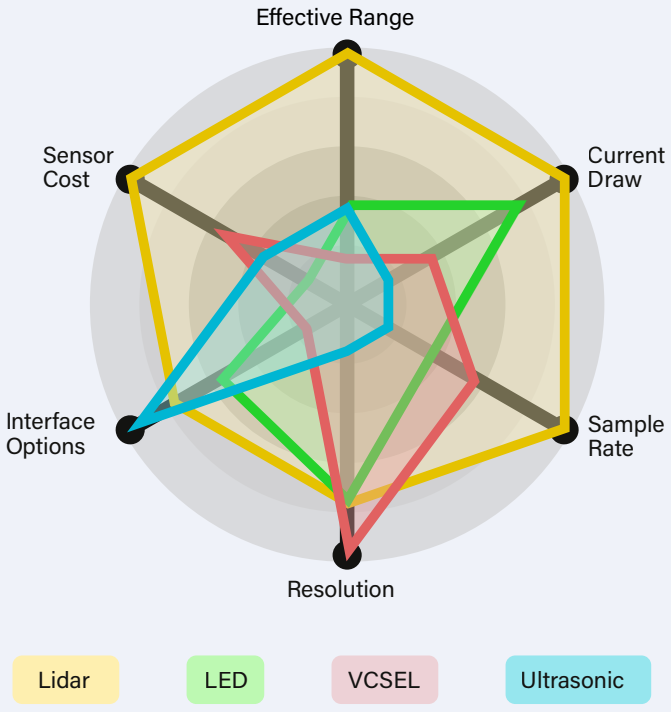


Bild 1. Um den richtigen Abstandssensor für Ihre Anwendung auszuwählen, sollten Sie sich überlegen, wie die verfügbaren Technologien im Hinblick auf Ihr(e) Entwicklungsziel(e) abschneiden.

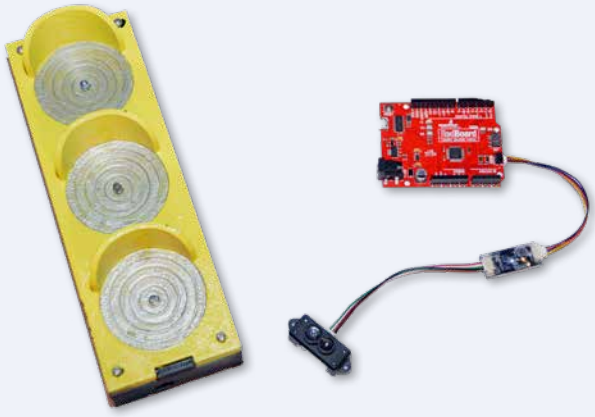


Bild 2. Die Elektronik und das 3D-gedruckte Gehäuse des Projekts „Perfektes Einparken mit LiDAR".

Tauschen Sie das RedBoard gegen einen Raspi und programmieren Sie das Projekt in Python!

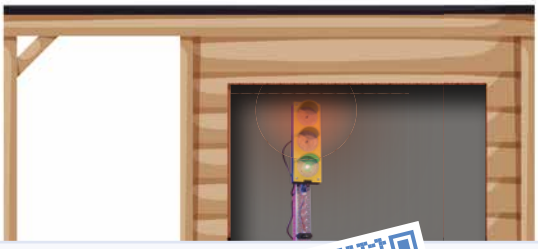
Rob Reynolds

Listing 1. Sketch zum Perfekten Einparken mit LiDAR

```
/*
  TFMiniStopLight.ino
  Rob Reynolds, November 19, 2018

  This code is a small practical demonstration
application of the TFMini Lidar Module. The 3D
files can be found in the Github repository, here
  [ https://github.com/ThingsRobMade/TFMini_Stop_
Light ]
  Based heavily on the previous collaborative work
done by Nate Seidle and Benewake. The original
example sketch for the Qwiic Enabled TFMini can be
found here:
  (https://www.sparkfun.com/products/14786)

  This code is free, but if you find it useful,
and we meet someday, you can buy me a beer
(Beerware license).
*/

#include <Wire.h>

uint16_t distance = 0; //distance
uint16_t strength = 0; //signal strength
uint8_t rangeType = 0; //range scale
/*Value range:
  00 (short distance)
  03 (intermediate distance)
  07 (long distance) */

boolean valid_data = false;
//ignore invalid ranging data

const byte sensor1 = 0x10;
//TFMini I2C Address

// Define pins for LEDs
const int greenLED = 8;
const int yellowLED = 9;
const int redLED = 10;
int stopLimit = 160; //Change this number of cm to
                     // adjust stop distance

void setup()
{
  Wire.begin();

  Serial.begin(115200);
  Serial.println("TFMini I2C Test");
  //For testing using Serial Monitor

  // Set LED pins as outputs
  pinMode(redLED, OUTPUT);
  pinMode(yellowLED, OUTPUT);
  pinMode(greenLED, OUTPUT);
}

void loop()
{
  if (readDistance(sensor1) == true)
  {
    if (valid_data == true) {
      Serial.print("\stopLimit[");
/* These Serial.print lines remain for testing and
adjustment purposes */
      Serial.print(stopLimit);
      Serial.print("\tDist[");
      Serial.print(distance);
      Serial.println();

      if (distance <= stopLimit) {
        digitalWrite(redLED, HIGH);
        digitalWrite(yellowLED, LOW);
        digitalWrite(greenLED, LOW);
      }
      else if (distance > stopLimit && distance
< (stopLimit + 200) ) { //change this number to
   // increase distance yellow stays lit
        digitalWrite(redLED, LOW);
        digitalWrite(yellowLED, HIGH);
        digitalWrite(greenLED, LOW);
      }
      else if (distance > (stopLimit + 199) ) {
   //change this number to adjust when yellow LED
   // illuminates
        digitalWrite(redLED, LOW);
        digitalWrite(yellowLED, LOW);
        digitalWrite(greenLED, HIGH);
      }
    }

    //  else {
    //   Serial.println("Read fail");
    //  }

    delay(50);
    //Delay small amount between readings
  }
}


//Write two bytes to a spot
boolean readDistance(uint8_t deviceAddress)
```

```
{
  Wire.beginTransmission(deviceAddress);
  Wire.write(0x01); //MSB
  Wire.write(0x02); //LSB
  Wire.write(7);
  //Data length: 7 bytes for distance data
  if (Wire.endTransmission(false) != 0) {
    return (false); //Sensor did not ACK
  }
  Wire.requestFrom(deviceAddress, (uint8_t)7);
  //Ask for 7 bytes

  if (Wire.available())
  {
    for (uint8_t x = 0 ; x < 7 ; x++)
    {
      uint8_t incoming = Wire.read();

      if (x == 0)
      {
        //Trigger done
        if (incoming == 0x00)
        {
          //Serial.print("Data not valid: ");
          //for debugging
          valid_data = false;
          //return(false);
        }
        else if (incoming == 0x01)
        {
          Serial.print("Data valid:     ");
          valid_data = true;
        }
      }
      else if (x == 2)
        distance = incoming;
```

```
      //LSB of the distance value "Dist_L"
      else if (x == 3)
        distance |= incoming << 8;
      //MSB of the distance value "Dist_H"
      else if (x == 4)
        strength = incoming;
      //LSB of signal strength value
      else if (x == 5)
        strength |= incoming << 8;
      //MSB of signal strength value
      else if (x == 6)
        rangeType = incoming; //range scale
    }
  }
  else
  {
    Serial.println("No wire data avail");
    return (false);
  }

  return (true);
}
```



Passende Produkte

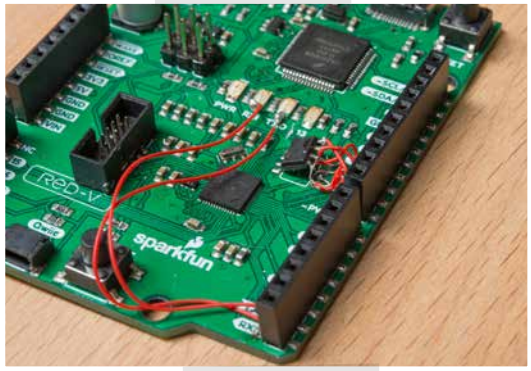
Suchen Sie die in diesem Artikel erwähnten Produkte? Elektor und SparkFun haben sie!

- **SparkFun RedBoard - Programmed with Arduino**
  www.elektormagazine.de/esfe-en-perfectparking1
- **TFMini - Mikro-LiDAR-Modul**
  www.elektormagazine.de/esfe-en-perfectparking2

WEBLINKS

[1] TFMini - Mikro-LiDAR-Modul: https://www.sparkfun.com/products/14588
[2] TFMini - Qwiic-Anleitung: https://bit.ly/2Xzun4r
[3] Sketch Perfect Parking, STL-Dateien für Gehäuse: https://github.com/ThingsRobMade/TFMini_Stop_Light

Ein handgemachtes Buried Pad



Reparaturmaßnahme: SOIC mit vertauschten Anschlüssen.

Von Nathan Seidle (SparkFun)

Viele Firmen leugnen, dass sie jemals Fehler machen. Wir von SparkFund dagegen stehen zu unseren Fehlern. Wir nageln sie an die Wand und erzählen jedem davon, damit wir nicht noch einmal den gleichen Fehler machen.

Ich habe schon oft gesagt: „Jeder kann eine Platine machen, aber es erfordert echtes Können, eine schlechte Platine gut zu machen." Das ist eigentlich nur eine Ausrede, wenn ich einen Entwurf vermasselt habe. Wir haben eine Wand im SparkX-Labor, an die wir – wie eingangs versprochen – all unsere schlechten Platinen, die wir in den letzten Jahren in den Sand gesetzt haben, für jedermann sichtbar an die Wand genagelt haben (**Bild 1**).

Funktioniert die Schaltung?

Ich entwerfe (und verpatze) seit fast zwei Jahrzehnten Platinen. Ich habe viele Tricks gelernt, aber die wichtigste Lektion ist, dass man unabhängig von der Anzahl der durchtrennten Leiterbahnen oder der benötigten Korrektur-Drahtverbindungen sicherstellen muss, dass die Schaltung zu 100% funktioniert, bevor man weitere Platinen bestellt. Wenn Sie nur einen Fehler finden und behoben haben und dann neue Platinen bestellen, werden Sie schnell feststellen, dass Sie nur den ersten Fehler behoben haben. Bei einem kürzlich entwickelten Projekt musste ich jedoch ein ganz besonderes und für mich neues Manöver durchführen, das ich mit Ihnen teile möchte (**Bild 2**).
Dieser Prototyp ist eine Kombination aus dem Artemis und unserem LTE-Mobilfunkmodul. Nichts allzu Wildes. Wie bei den meisten Prototypen habe ich mit einem RedBoard Artemis und dem LTE-Shield begonnen, um sicherstellen, dass alles funktionieren würde. Dann habe ich ein Platine entworfen und ein paar Prototypen gebaut. Zunächst funktionierte auch alles, außer dass sich das Mobilfunkmodul nicht

Bild 1a. Ungeniert an die SparkX-Wand genagelt.

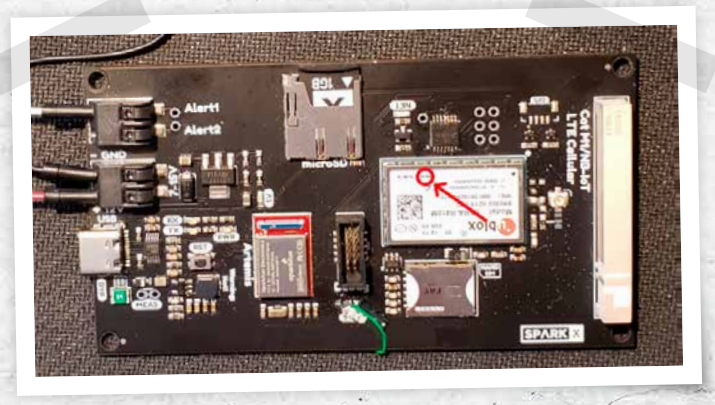


Bild 2. Wie komme ich zu diesem Pad?

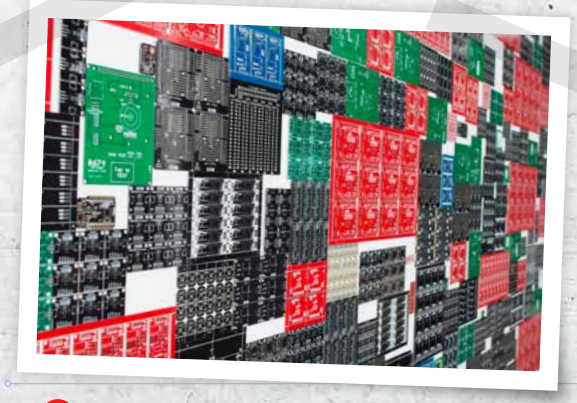


Bild 1b. Schauen Sie sich diese wunderbaren Fehlschläge von SparkX an.

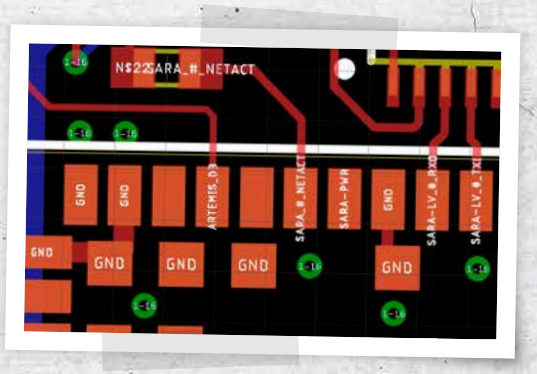


Bild 3. Es führt ein Pad ins Nirgendwo.

einschalten ließ. Ich habe mir alle möglichen Ausreden einfallen lassen (ich hatte zu viele Projekte zur gleichen Zeit, ich hab zu schnell gearbeitet, blablabla), aber ich hatte völlig vergessen, den Power-Pin vom SARA-Modul zum Artemis zu routen (**Bild 3**).

Das Mobilfunkmodul SARA LTE NB-IoT (**Bild 4**) besitzt einen unglaublichen Funktionsumfang und verfügt, wie viele Mobilfunkmodule, über einen Power-Pin, der für zwei oder drei Sekunden aktiviert werden muss, damit das Modul anläuft. Bei einem normalen Entwurf mit einem SO-IC oder TSSOP liegen die IC-Beinchen frei und es ist einfach, daran etwas anzulöten. Mit ein wenig Klebemasse, um einen Wire-wrap-Draht an Ort und Stelle zu halten, etwas Flussmittel, etwas Lötzinn und etwas Entlötlitze können Sie sogar QFN-Bauteile löten. Das SARA-Modul ist aber insofern etwas Besonderes, als dass die Pads direkt unter dem Modul liegen. Es gibt kein Kupfer oder Pad auf der Unterseite des Moduls, das von außen erreichbar wäre. Und ebenso besonders ist, dass die Pads auf der Unterseite des Moduls ziemlich groß sind (im Vergleich zu einem QFN). Ich hatte die Wahl: Ich konnte entweder eine neue Platine anfertigen lassen oder versuchen, durch die Unterseite der Platine zu bohren und damit aufzuhören, bevor ich das Pad auf der Unterseite des SARA-Moduls zerstören und irgendwie heißes Lot und einen Draht auf das SARA-PWR-Pad bekommen würde. Versuch macht klug!

Bohren, Löten, Kleben

Der erste Schritt bestand darin, die genaue Stelle zum Bohren auf der Rückseite der Platine ausfindig zu machen. Ich öffnete das Eagle-Board-Design, drehte es um und fügte einen 1,5-mm-Punkt an der Stelle hinzu, an der ich den Draht befestigen wollte. Ich ließ den Punkt die Lücke zwischen dem benötigten Pad und der nahegelegenen Massefläche überbrücken. Dies war beabsichtigt - sobald das Loch an der Stelle angebracht wäre, könnte ich leicht die genaue Position des SARA-PWR-Pads finden (**Bild 5**).

Ich druckte das Layout im Maßstab 1:1 aus, schnitt es aus und richtete es an der Rückseite des Prototyps aus. Mit einem Körner machte ich eine kleine Vertiefung an der Stelle, an der das Loch angebracht werden sollte (**Bild 6**). Die kleine Vertiefung war wichtig, um den Bohrer in der richtigen Position zu halten.

Mit zwei handelsüblichen Bohrern von 1,5 mm und 2,3 mm (**Bild 7**) suchte ich die Standbohrmaschine im BTU-Labor (Blow-Things-Up-Lab) der CU (University of Colorado Boulder) auf, um die ersten Probebohrungen durchzuführen (**Bild 8**). Ich kam durch die unteren Lagen

Bild 4. Vorder- und Rückseite des Ublox-SARA-Moduls.

Bild 5. Lage des Pads von der Unterseite der Platine aus gesehen.

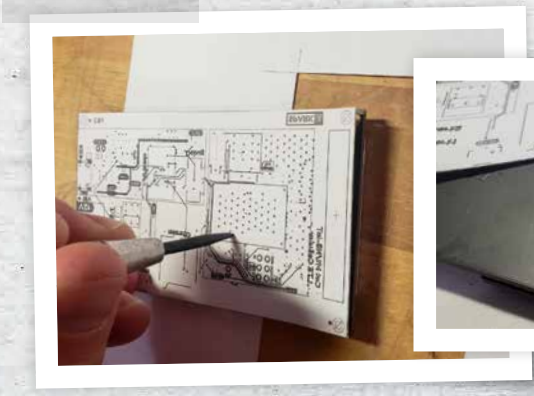


Bild 6a. Der gefederte Körner markiert die Stelle.

Bild 6b. Die Einschlagstelle zentriert den Bohrer.

Bild 7. Standardbohrer aus dem örtlichen Baumarkt.

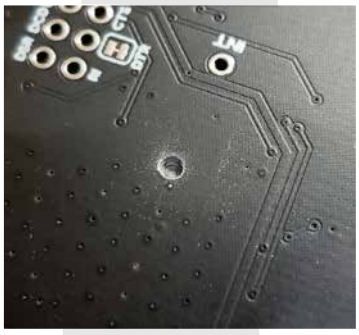


Bild 8. Erste Bohrung mit 1,5-mm-Bohrer.

von Lötstoppmaske, Kupfer und FR4. Da es sich um eine zweilagige Platine handelte, hörte ich auf zu bohren, als ich dachte, dass ich kurz davor war, die obere Kupferlage zu treffen. Ich beendete das Bohren dann von Hand (**Bild 9**): Ein neuer, scharfer Bohrer ist erstaunlich effektiv beim Entfernen von FR4! Als der Bohrer begann, sich zu verhaken, war die oberste Kupferschicht wohl erreicht, also hörte ich auf.
Ich wechselte dann zum 2,3-mm-Bohrer, um das Loch so weit zu vergrößern (**Bild 10**), dass ich eine Multimeter-Messspitze einführen konnte. Ich kratzte an der Seite der Bohrung und prüfte den elektrischen Durchgang zwischen dem unteren Rand des Lochs (wo ich hoffte, einen Kontakt mit der Kupfer-Massefläche auf der Bestückungsseite herzustellen) und einer nahe gelegenen Massefläche auf der Platinenunterseite. Zufrieden damit, dass ich es bis zur oberen Kupferschicht geschafft hatte (zumindest bis zur großen Massefläche), bohrte ich mit dem kleineren 1,5-mm-Bohrer leicht schräg weiter.
Irgendwann entschied ich mich, mit etwas Isopropylalkohol auf einem Wattestäbchen die Glasfaserreste aus dem Loch zu entfernen. Ich wurde mit einer großen Überraschung belohnt, ich konnte die Pads sehen (**Bild 11**)! Es war zwar klar, wo sich die Massefläche befand, aber ich konnte nicht sicher sein, welches der beiden das gewünschte Power-Pad war. Ein schneller Abgleich mit meinem gedruckten Layout ergab, dass es sich um das Pad auf der linken Seite handeln musste (**Bild 12**). Der Alkohol machte die Glasfasern zwar für einen Moment transparent, aber er verdampfte schnell und ließ das undurchsichtige FR4 zurück (**Bild 13**). Es war offensichtlich, dass sich über dem Pad noch mehr Glasfaser befand.
Nach weiterem Bohren von Hand begann etwas zu glänzen. Das Loch (**Bild 14a**) war zwar groß genug für meine Lötkolbenspitze, aber es war doch ein bisschen heikel. Das Hinzufügen von ein wenig Flussmittel bewirkte zwei Dinge, darunter eines, das ich nicht erwartet hatte: Flussmittel lässt nicht nur das Lötzinn dorthin fließen, wohin es fließen soll, sondern es enthält, wenn es sich um Clean-Flussmittel handelt, auch eine milde Säure, um oxidierte Kontakte zu reinigen. Diese Säure war genau das, was ich brauchte, um genug FR4 zu entfernen und eine lötbare Stelle auf dem Pad zu schaffen.
Nach ein paar Lötversuchen mit einem Stück sehr dünner (0,05 mm^2) verzinnter Litze habe ich schließlich ausreichend Hitze auf das Kabel übertragen können, dass das Lot an der Drahtspitze auf das freiliegende Pad schmolz. Es war eine ziemlich friemelige Arbeit (**Bild 14b**), aber die Lötverbindung hielt schließlich sogar einem sehr leichten Ruck am Kabel stand. Ich machte noch einige Durchgangsprüfungen mit meinem Multimeter, um einen versehentlichen Kurzschluss mit der Massefläche auszuschließen.
Der Bereich um die Oberseite des Lochs gehört komplett zur Massefläche. Der Draht liegt frei, also habe ich darauf geachtet, dass er nicht mit der Massefläche in Kontakt kommen kann. Etwas Isolierband hielt das Kabel an seinem Platz (**Bild 14c**), während ich sein anderes Ende an den Mikrocontroller (den Artemis) lötete. Ein kurzer Test der Platine bewies, dass das Mobilfunkmodul eingeschaltet werden konntee, Es funktionierte!
Ich war natürlich besorgt, die Lötverbindung und/oder das Pad zu zerreißen. Nachdem alles elektrisch getestet war, füllte ich deshalb das Loch und umgab den Draht mit Heißkleber (**Bild 14d**). Dies bot ausreichend Schutz und Zugentlastung für das Kabel und die Lötverbindung.

Bild 9. Ich habe von Hand weitergebohrt.

Bild 11. Da sind die Pads!

Bild 10. Nichts vom Pad zu sehen.

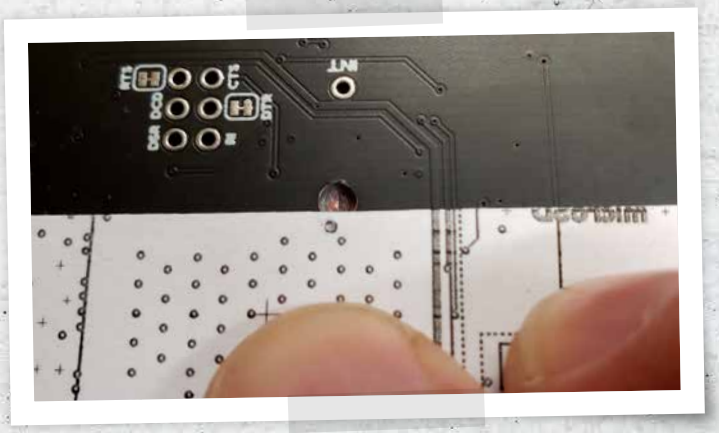


Bild 12. Das Power-Pad befindet sich auf der linken Seite.

Ein bisschen Glück gehört dazu

Offensichtlich funktionierte diese Methode nur (**Bild 15**), weil ich in einigen Punkten pures Glück hatte: Das Modul besaß große Pads, die das Löten einfacher gestalteten, und es gab keine Leiterbahnen direkt unter den Pads, so dass ich von der Rückseite bohren konnte. Nachdem ich das Problem mit dem Power-Pin gefunden und behoben hatte, setzte ich die Entwicklung fort und fand andere Probleme mit meinem Entwurf. Hätte ich sofort nach der Entdeckung des Power-Pin-Problems aufgehört und eine neue Version dieser Platine bestellt, hätte ich die vier anderen Probleme mit dem Board erst später gefunden. Wenn Sie also das nächste Mal eine fehlerhafte Platine haben, denken Sie zweimal darüber nach, ob Sie sie nicht doch zum Laufen bringen können.

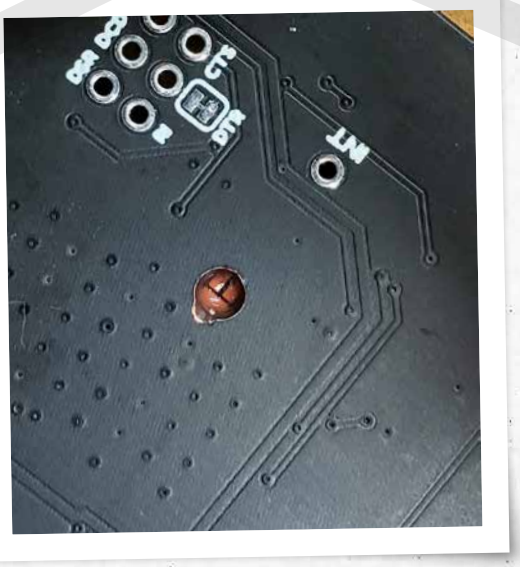


Bild 13. Alkohol macht FR4 transparent.

Bild 14a. Das Loch war groß genug für meine Lötspitze.

Bild 14b. Nicht schön.

200700-02

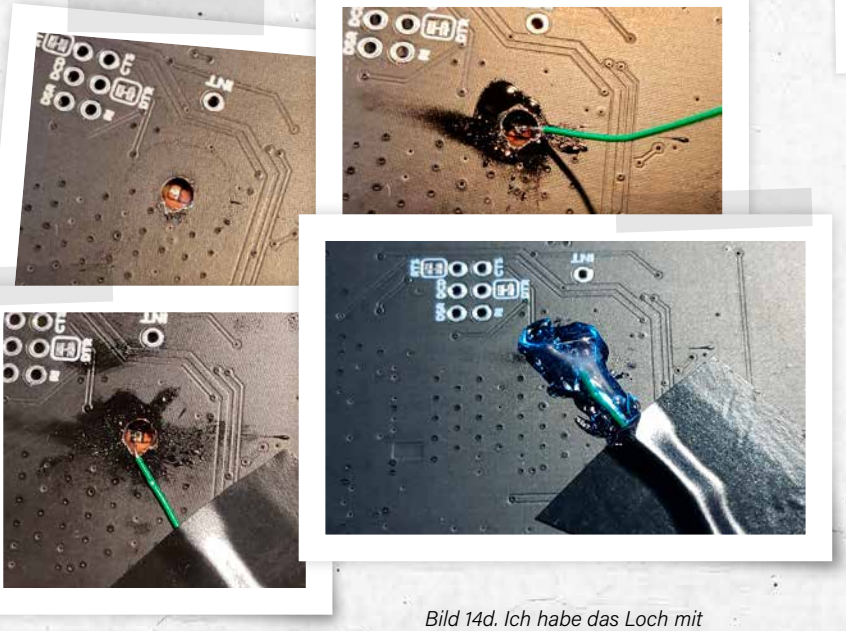


Bild 14c. Die Litze ist erfolgreich durch die Leiterplatte auf das Pad gelötet.

Bild 14d. Ich habe das Loch mit Heißkleber gefüllt und das Kabel fixiert.

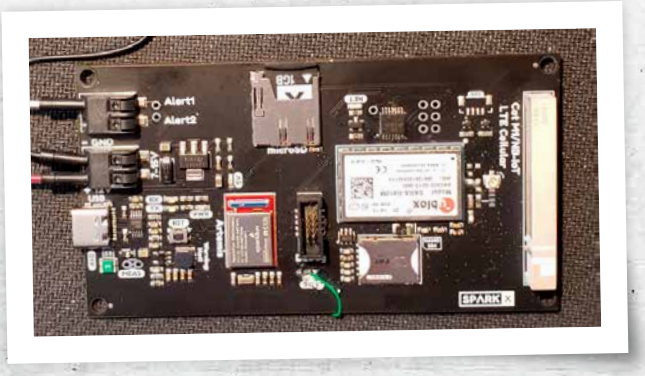


Bild 15. Hackjob erledigt, Schaltung funktioniert.

Passende Produkte

Suchen Sie nach den Produkten, die in diesem Artikel erwähnt werden? SparkFun und Elektor haben die passenden Produkte für Sie!

- **SparkFun Artemis Modul - Low Power ML BLE Cortex-M4F**
  www.elektormagazine.de/esfe-en-erroranalysis1
- **SparkFun RedBoard Artemis**
  www.elektormagazine.de/esfe-en-erroranalysis2
- **SparkFun LTE CAT M1/NB-IoT Schild - SARA-R4**
  www.elektormagazine.de/esfe-en-erroranalysis3

Elektor-Kommentar

Vor einigen Monaten haben wir die Elektor-Artikelserie „Fehleranalyse" (https://www.elektormagazine.de/search?query=Fehleranalyse) gestartet, um Ingenieuren und Makern unserer Community zu helfen, aus den Fehlern ihrer Kollegen zu lernen. Und so ist unser Team stets auf der Suche nach technischen Fehlern und hilfreichen Lektionen, die wir teilen können. Als wir die Fotos der „SparkX Wall of Unabashed Failure" sahen, wussten wir sofort, dass wir mehr darüber erfahren müssten.

Vom Entwurf zum Verkauf: der SparkFun-RTK-Surveyor

Der RTK-Surveyor ist ein GNSS-Empfänger, den Sie zur präzisen Positionsbestimmung verwenden können. Neugierig auf den Entwicklungsprozess bis zur Markteinführung? Dann lassen Sie uns einen Blick darauf werfen!

Von Christopher McCarty (USA)

RTK steht für Echtzeit-Kinematik. Der *SparkFun RTK Surveyor* ist ein einfach zu bedienender GNSS-Empfänger (Global Navigation Satellite System) zur Positionsbestimmung im Zentimeterbereich. Dieses vorprogrammierte Gerät ist perfekt für Vermessungen geeignet, kann aber auch für autonomes Fahren, Navigation, Nachverfolgung von Gütern und jede andere Anwendung im Freien verwendet werden. Der RTK-Surveyor kann auch als Basisstation verwendet werden. Mit zwei dieser Geräte lässt sich ein RTK-System mit einer horizontalen Positionsgenauigkeit von 14 mm erstellen. Die eingebaute Bluetooth-Verbindung über ein ESP32-WROOM-Controllerboard ermöglicht es dem Anwender, das ZED-F9P-Board des RTK Surveyors mit einer GIS-Anwendung (Geografisches Informationssystem) seiner Wahl auf einem Handy oder Tablet zu nutzen. Der eingebaute Akku ermöglicht einen Feldeinsatz von bis zu vier Stunden und ist mit gängigen

31 Zeitleiste

15. August 2020
SparkFun-Gründer Nathan Seidle lötet den ersten Prototyp des RTK-Surveyors zusammen. Zu diesem Zeitpunkt ist es nicht viel mehr als ein Thing-Plus-Controllerboard ESP32 WROOM, das mit ein paar Kabeln an ein GPS-RTK-SMA Breakout gelötet ist. Egal, wie es aussieht, es funktioniert - und es funktioniert gut!

20. August 2020
Die ersten Platinen für den RTK-Surveyor sind bestellt. V0.1 der Platine ist nur wenig besser als der ursprüngliche zusammengelötete Entwurf von Nate. Glücklicherweise konnten wir bei diesem Einplatinenentwurf die Knackpunkte auf der Platine entdecken, zum Beispiel Probleme mit der Wärmeableitung des u-blox-ZED-F9P, Rückkopplungsrauschen, die den GNSS-Empfänger störten, und einige schlechte Leiterbahnen auf der Platine selbst. Aber seien wir ehrlich: Wer hat solche Probleme nicht bei einer ersten Produktionsserie?

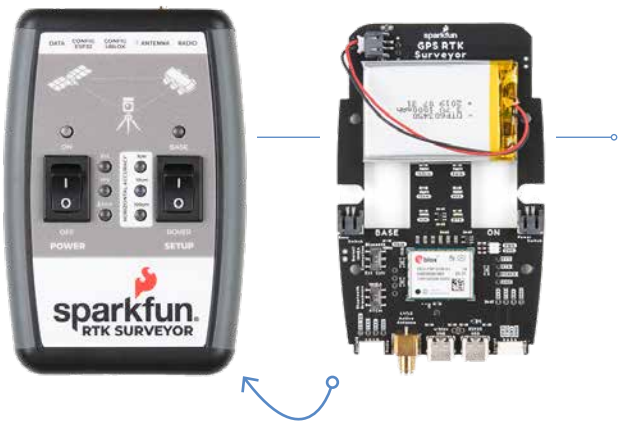
23. August 2020
Die ersten Platinen für den RTK-Surveyor treffen ein. Dies ist das erste Mal, dass ein speziell angefertigtes Vermessungsgerät im SparkFun-Headquarter ist, oder um genau zu sein, das erste Mal seit dem Vermessungsgerät, das beim Bau des Gebäudes verwendet wurde.

16. September 2020
V0.2 des Prototyps der RTK-Surveyor-Platine wird bestellt. Nach den behobenen Problemen wie Rauschen und Wärmeableitung der vorherigen Version haben wir schon einen beeindruckend genauen Prototyp des Vermessungsgeräts geschaffen. Es waren nur noch ein paar Kleinigkeiten zu verbessern. Die nächste Bestellung wäre dann schon die Platine, die im Surveyor eingesetzt werden sollte!

14. und 15. Oktober 2020
Die Tutorials „Setting up a Rover Base RTK System“ und „How to Build a DIY GNSS Reference Station“ gehen im Learn-Bereich der SparkFun-Seite online. Dies sind die ersten guten Beispiele auf SparkFun.com, wie Anwender, die an GNSS-Vermessung interessiert sind, diese erlernen können. Auch darüber können Sie in dieser Ausgabe etwas lesen!

19. Oktober 2020
Ein Gehäuse für den RTK-Surveyor ist gefunden und gekauft. Wir hatten eine Vorstellung von der Form des Gehäuses, und da wir uns für ein generisches Design entschieden hatten, brauchten wir nur einen Lieferanten zu finden.

20. Oktober 2020
Der Frontplattenaufkleber für den RTK-Surveyor ist entwor-



USB-Powerbanks kompatibel.
Der RTK-Surveyor senkt die Einstiegshürde für Vermessungstools sowohl in finanzieller Hinsicht als auch in Bezug auf die Benutzerfreundlichkeit. Im Gegensatz zu anderen Vermessungsgeräten auf dem Markt mit exorbitanten Preisen und/oder proprietärer Software ist der RTK-Surveyor von SparkFun mit Open-Source-Software zu einem Bruchteil des Preises erhältlich. Aber was war der Beweggrund, ein solches Vermessungsgerät zu entwickeln? Und wie sieht der Zeitplan für ein neues, und offen gesagt, experimentelles Produkt dieser Art aus?

Der Anstoß

Die Idee, den RTK-Surveyor bei SparkFun zu entwickeln, begann mit dem Wunsch von Nathan Seidle, einen ESP32 mit einem leistungsstarken GNSS-RTK-Board zu kombinieren. Leider fand er keine fertige Lösung, die sowohl finanziell akzeptabel als auch in einem schützenden und nicht leitenden Gehäuse untergebracht war. Auf diesen beiden Bedingungen lag zwar der Hauptaugenmerk, aber es sollten auch weitere Funktionen hinzugefügt werden. Nate wusste, was er von einem Vermessungsgerät verlangte, und er brachte all das mit, was SparkFun zu bieten hatte. Keine Programmierung, kostengünstig, Open-Source, alles schien sich ohne wirkliche Komplikationen zusammenzufügen. Nach nur ein paar Monaten war der RTK-Surveyor fertig für die Anwendung durch jedermann, und dies, ohne irgendwelche Abstriche an seine Funktionalität machen zu müssen.

Wie Sie sehen, steckt eine Menge Arbeit in der Entwicklung und der Produktion eines SparkFun-Produkts, insbesondere eines, das eine Genauigkeit verlangt, die es bisher nur für einen Nischenmarkt gab. Am Montag nach der ersten Auslieferung des SparkFun-RTK-Surveyors haben wir mit der Arbeit an seiner nächsten Iteration begonnen. Natürlich wollen wir nicht gleich nach der Markteinführung eines Produkts eine neue Version herausbringen, aber wir können auch nicht aufhören, das, was wir erschaffen, zu erneuern. Dem SparkFun-RTK-Surveyor geht es da nicht anders.

Ein Blick in die Zukunft

Was soll es Neues geben beim RTK-Surveyor? Wir arbeiten bereits an neuen Iterationen des GNSS-Tools, indem wir alle neuen Builds mit V1.1 des internen Boards ausstatten. Die FCC- und CE-Zertifizierung sollte kurz darauf folgen, zusammen mit einigen zusätzlichen Funktionsänderungen, die damit einhergehen. Sobald dies geschehen ist, wird der RTK-Surveyor von SparkX „graduiert" und erhält volle Produktionsreife, technischen Support und Markenkennzeichnung. Wir erwarten, wenn alles gut läuft, V2.0 des RTK-Surveyor Ende 2021 oder Anfang 2022. Halten Sie die Augen offen, denn wir arbeiten daran, das Gerät mit einigen beeindruckenden Upgrades auszustatten. Der Preis des Geräts soll allerdings kein Upgrade erhalten, sondern in etwa so bleiben wie er ist.

200701-02

Passende Produkte

Suchen Sie nach den in diesem Artikel erwähnten Produkten? Elektor und SparkFun haben Sie!

› **SparkFun RTK Surveyor**
www.elektormagazine.de/esfe-en-rtk1

fen und gedruckt. Nachdem wir wussten, welches Gehäuse wir wollten, konnten wir uns direkt an die Arbeit machen, den Aufkleber dafür zu gestalten! Dank des untermüdlichen Einsatzes unseres hauseignen Grafikers konnten wir den Aufkleber für den RTK-Surveyor in kürzester Zeit entwerfen und in Auftrag geben.

23. Oktober 2020
Die erste Produktionsserie von SparkX-RTK-Surveyor-Platinen wird bestellt. Sobald die V1.0-Platinen, alle Bauteile, das Gehäuse und die Aufkleber bei uns eintreffen, können wir mit der Produktion einer ersten Charge des RTK-Surveyor loslegen. Die Entwicklung eines solch speziellen Vermessungsgeräts war für SparkX eine echte Herausforderung, insbesondere während einer Pandemie.

27. November 2020
Die erste Charge von RTK-Surveyors wird gebaut und für die Freigabe getestet. Alles ist bereit - noch sieben Tage bis zur Freigabe. Es ist an der Zeit, sich zu freuen!

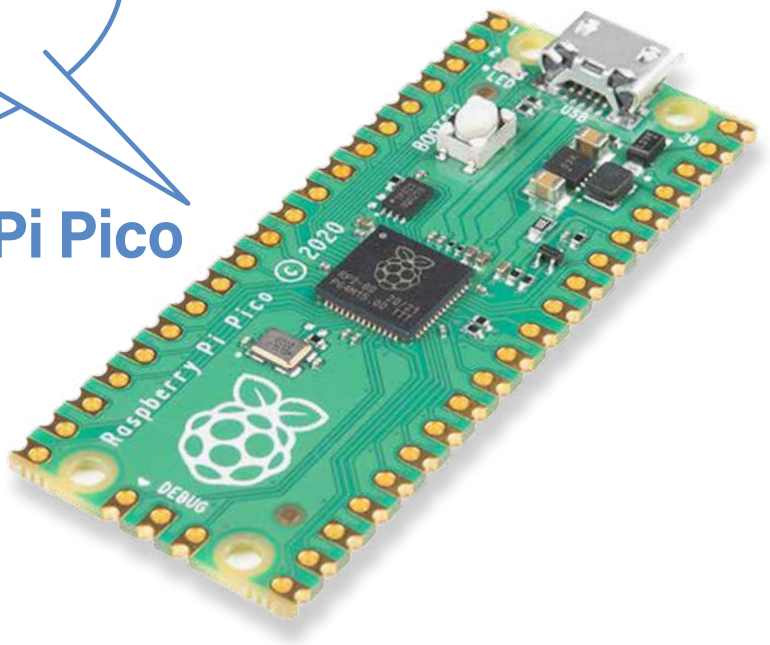
30. November 2020
Der RTK-Surveyor wird zur FCC- und CE-Zertifizierung eingereicht. Es ist nicht das erste Mal, dass wir ein „SparkFun-Original" für alle Zertifizierungen einreichen, um es weltweit verkaufen zu können, und wir haben ein paar Dinge aus unserem ersten Versuch mit Artemis über den Zertifizierungsprozess gelernt, insbesonders darüber, wie lange es dauern kann, eine Zulassung zu erhalten. Es stellte sich heraus, dass die Entwicklung eines GNSS-Vermessungswerkzeugs etwas mehr Zeit in Anspruch nehmen kann!

3. Dezember 2020
V1.1 der RTK Surveyor-Platinen sind bestellt, jetzt in SparkFun-Rot!

4. Dezember 2020
Der *SparkFun RTK Surveyor* geht mit V1.0-Platinen an die Öffentlichkeit. Der Tag ist gekommen, der RTK-Surveyor wird für jedermann verfügbar und nutzbar. Noch vor dem Wochenende ist er ausverkauft.

7. Dezember 2020
Die Arbeit an der V2.0 des RTK-Surveyors beginnt. Wenn Sie Nathan Seidle jemals getroffen haben, werden Sie wissen, dass er nie aufhört zu arbeiten, und der RTK-Surveyor macht da keine Ausnahme.

Hello World vom Raspberry Pi Pico und RP2040



Ein Blick auf den ersten Mikrocontroller der Raspberry Pi Foundation

Von Avra Saslow (USA)

Mit Blick auf ihre zukünftige Rolle im Physical Computing hat die Raspberry Pi Foundation ein revolutionäres Produkt entwickelt, einen Raspberry Pi mit einem Mikrocontroller, der sowohl MicroPython als auch C/C++ vollständig unterstützt. Willkommen in der neuen Welt des Raspberry Pi Pico mit dem RP2040-Mikrocontroller-Chip!

Der Raspberry Pi Pico (**Bild 1**) ist ideal, wenn Sie ein wenig komplexes Projekt realisieren möchten, bei dem nur ein Programm ausgeführt werden soll. In diesem Fall brauchen Sie nicht die Größe und Leistungskraft eines vollwertigen Raspberry Pi, aber Sie möchten auch nicht auf einer Arduino-Plattform entwickeln. Der Pico wird von einer umfassenden und gründlichen Dokumentation begleitet, die sowohl das *MicroPython Software Development Kit* (SDK) als auch das *C/C++ SDK* behandelt. Wenn Sie mit Python-Programmierung vertraut sind oder Ihre Erfahrungen mit dem Arduino-C++ einbringen möchten, könnte sich der Pico als Ihr neuer Lieblings-Mikrocontroller entpuppen.

Eine Einführung in den Raspberry Pi Pico

Lassen Sie uns einen genaueren Blick auf die technischen Daten werfen! Der Mikrocontroller-Chip ist der brandneue RP2040, der von der Raspberry Pi Foundation entwickelt wurde. Die Bezeichnung RP2040 besteht, wie **Bild 2** zeigt, aus den Initialen der Stiftung, gefolgt von der Anzahl der Prozessorkerne (2), dem Typ des Prozessorkerns (M0+), RAM – floor(log2(ram/16k)) - und schließlich ROM – floor(log2(nonvolatile/16k)).

- Der Chip verfügt sowohl über einen UF2-Boot, um den Mikrocontroller mit Firmware über USB flashen zu können (Drag-and-Drop-Programmierung), als auch über Fließkommaroutinen. Er ist mit Dual-Cortex-M0+-Prozessorkernen ausgestattet, die flexibel mit bis zu 133 MHz getaktet werden können. Dank einer leistungsfähigen Busmatrix kann er die volle Leistung auf beiden Kernen gleichzeitig abrufen. Damit könnte der Controller letztlich Machine-Learning-Modelle mit TensorFlow für MicroPython ermöglichen. Er verfügt über ein großes internes RAM (264 KB SRAM), verwendet aber auch externen Flash-Speicher „on Board“, so dass der Benutzer selbst entscheiden kann, welchen Speicher er benutzt.
- Energiesparender Sleep- und Ruhemodus.
- Eingebauter USB, der sowohl als Device als auch als Host fungieren kann.
- Verschiedene digitale periphere Hardware wie 2x UART, 2x I²C, 2x SPI, bis zu 16 PWM-Kanäle, ein Timer mit vier Alarmen und ein Echtzeitzähler.
- 30 Multifunktions-GPIO (vier können als ADC verwendet werden) (**Bild 3**).
- Ein Temperatursensor.
- 8 programmierbare IO (PIO)-Zustandsautomaten für die Unterstützung kundenspezifischer Peripherie. Bei den meisten Mikrocontrollern müssen Sie „bit-bangen“, also den Hauptprozessorkern verwenden, um Pins direkt ein- und auszuschalten. Das funktioniert zwar, kann aber zu Timing-Problemen führen, besonders bei Verwendung von Interrupts, und kann viel Verarbeitungsressourcen beanspruchen, die Sie für andere Dinge benötigen. Mit den programmierbaren Ein- und

Ausgängen oder 2 PIO-Blöcken, die vier Zustandsmaschinen haben, können Sie eintreffende und ausgehende Daten verarbeiten und damit die Implementierung von Kommunikationsprotokollen auslagern, die ansonsten den Prozessor belasten würden.

- Und eine ganze Menge mehr!

Was die Software betrifft, so wird der RP2040 sowohl von den plattformübergreifenden Entwicklungsumgebungen C/C++ als auch MicroPython unterstützt, einschließlich des einfachen Zugangs zum Laufzeit-Debugging. Zwar stellt der Raspberry Pi Pico nicht unbedingt neue Rekorde in Puncto Leistungsfähigkeit auf, aber die technische Dokumentation sowohl für den RP2040 als auch für den Pico ist, wie man es von der Raspberry Pi Foundation kennt, beispiellos grundlegend.
All diese Spezifikationen machen deutlich, dass der leistungsfähige und unglaublich erweiterbare Pico ein typisches Produkt der Raspberry Pi Foundation ist, sowohl was seine Größe, seine Zugänglichkeit, seine technischen Fähigkeiten und nicht zuletzt seinen niedrigen Preis betrifft: Klein, aber oho! Die Raspberry Pi Foundation hat mit einem Paukenschlag die Mikrocontroller-Welt betreten!

Listing 1. Terminalbefehle für die Installation von MicroPython.

```
echo „Terminal commands for installing MicroPython“;
echo „Obtaining MicroPython“;
 cd ~/;
 mkdir pico;
 cd pico;
 git clone -b pico git@github.com:raspberrypi/micropython.git;

 echo „Obtain additional tools“;
 sudo apt update;
 sudo apt install cmake gcc-arm-non-eabi;
echo „Building MicroPython“;
 cd micropython;
 git submodule update --init --recursive;
 make -C mpy-cross;
 cd ports/rp2;
 make
```

Listing 2. Ein Timer lässt die LED blinken.

```
##credit of the Raspberry Pi Foundation
 from machine import Pin, Timer
led = Pin(25, Pin.OUT)
 tim = Timer()
 def tick(timer):
 global led
 led.toggle()

 tim.init(freq=2.5, mode=Timer.PERIODIC,
callback=tick)
```

MicroPython auf dem Pico zum Laufen gebracht

Vielleicht haben Sie schon bemerkt, dass die „hauseigene“ Programmiersprache gegenüber dem bisherigen Raspberry Pi von Python zu MicroPython beim Pico gewechselt hat. Python beansprucht zu viele Ressourcen, um auf kleinen Mikrocontrollern zu laufen. Daher wird üblicherweise für Mikrocontroller-Boards wie den Pico eine Portierung von Python namens MicroPython verwendet. Beide Sprachen sind nahezu identisch, nur verzichtet MicroPython auf viele Standard-Bibliotheksmodule, so dass nur wenig RAM benötigt wird (nach MicroPython-Angaben rund 16 K).

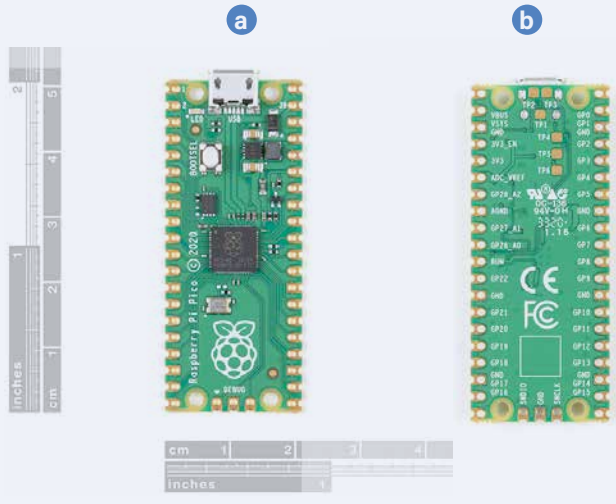


Bild 1a. Der Raspberry Pi Pico ist eine gute Lösung, wenn Sie nicht die Größe und Leistung eines vollwertigen Raspberry Pi benötigen. 1b. Die übersichtliche Rückseite des neuen Pico.

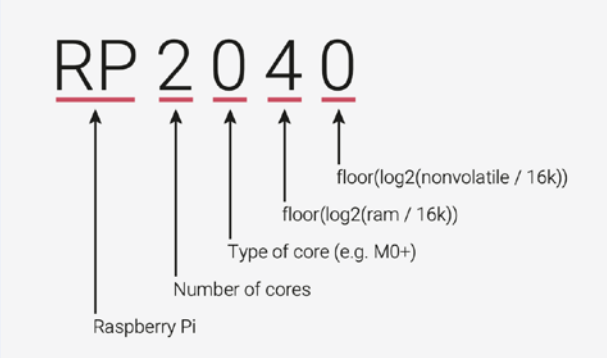


Bild 2. Der Schlüssel für die Bezeichnung des neuen Controllers.

Damit MicroPython auf dem Pico läuft, muss zunächst MicroPython-Github-Repository geklont und *Cmake* und *GNU Embedded Toolchain for Arm* installiert werden. Dazu können Sie einfach die Terminal-Befehle aus dem Shell-Skript (**Listing 1**) kopieren und ausführen. Das Skript erstellt aus dem MicroPython-Quellcode die Executables und die Bibliotheken auf Ihrem Pico.
Ab jetzt erfolgt die Installation der MicroPython-Software per Drag-and-Drop, indem Sie die *firmware.uf2* auf das Board ziehen. Sie müssen dabei die BOOTSEL-Taste gedrückt halten, damit das Board in den USB-Massenspeichermodus wechselt. Sobald die Firmware auf das Board geladen ist, können Sie sich mit der *MicroPython REPL* (Read Evaluate Print Loop) verbinden, was eine einfache Möglichkeit zum Testen von Code und zum Ausführen von Befehlen darstellt. Ich konnte auf diese einfache Weise den Beispielcode der Raspberry Pi Foundation Code implementieren, der einen Timer die On-Board-LED blinken lässt (**Listing 2**).

Ein wenig Morsecode gefällig?

Da ich mit dem Pico schon vor seiner Veröffentlichung gearbeitet habe (top secret!), wollte ich mit ihm etwas kryptischer kommunizieren. Also habe ich, aufbauend auf dem „Blink an LED“-Code, ein MicroPython-Skript erstellt, das „Hello World“ im Morsecode ausgibt (**Listing 3**). Und da eine blinkende LED nur schwer lesbar ist, habe ich das Programm auch dazu gebracht, die Morsecode-Übersetzung im Terminal auszugeben (**Bild 4**).

Listing 3. Ein MicroPython-Skript, das „Hello World“ als Morsecode ausgibt.

```
#-----------------------------------------------------------------------------
# Import necessary libraries, connect to on-board LED, set blink rate for LED
#-----------------------------------------------------------------------------

import time
from machine import Pin
led=Pin(25,Pin.OUT)#the LED on the Pico is pin 25
BlinkRate=0.25

#-----------------------------------------------------------------------------
# Functions for the morse code signal durations and the code itself
#-----------------------------------------------------------------------------

def dash():
    led.value(1)
    time.sleep(4*BlinkRate)
    led.value(0)
    time.sleep(BlinkRate)

def dot():
    led.value(1)
    time.sleep(BlinkRate)
    led.value(0)
    time.sleep(BlinkRate)

def pause():
    time.sleep(BlinkRate)


code = {'A':'.-','B':'-...','C':'-.-.','D':'-..','E':'.','F':'..-.','G':'--.',
    'H':'....','I':'..','J':'.---','K':'-.-','L':'.-..','M':'--','N':'-.',
    'O':'---','P':'.--.','Q':'--.-','R':'.-.','S':'...','T':'-','U':'..-',
    'V':'...-','W':'.--','X':'-..-','Y':'-.--','Z':'--..',
    '0':'-----','1':'.----','2':'..---','3':'...--','4':'....-',
    '5':'.....','6':'-....','7':'--...','8':'---..','9':'----.',
    '.':'.-.-.-',
    ',':'--..--',
    '?':'..--..',
    '/':'--..-.',
    '@':'.--.-.',
    ' ':' / ',

}

#-----------------------------------------------------------------------------
# Function that simply returns morse code sentence from uppercase English sentence
#-----------------------------------------------------------------------------

def convertToMorseCode(sentence):
    sentence = sentence.upper() #make it all caps so that the dictionary understands
the character
    secretSentence = „“ #empty sentence to add to
    for i in sentence: #for each character in the sentence, reference code
dictionary and change to morse code character
        secretSentence += code[i] + „ „
    return secretSentence

#-----------------------------------------------------------------------------
# main function that blinks LED based on morse code sentence
#-----------------------------------------------------------------------------

while True:
    sentence = „Hello World“
    secretSentence = convertToMorseCode(sentence)
    for i in secretSentence:
        if i == „.“:
            dot()
        elif i == „-“:
            dash()
        else:
            pause()
```

Bild 4. „Hello World“ im Morsecode.

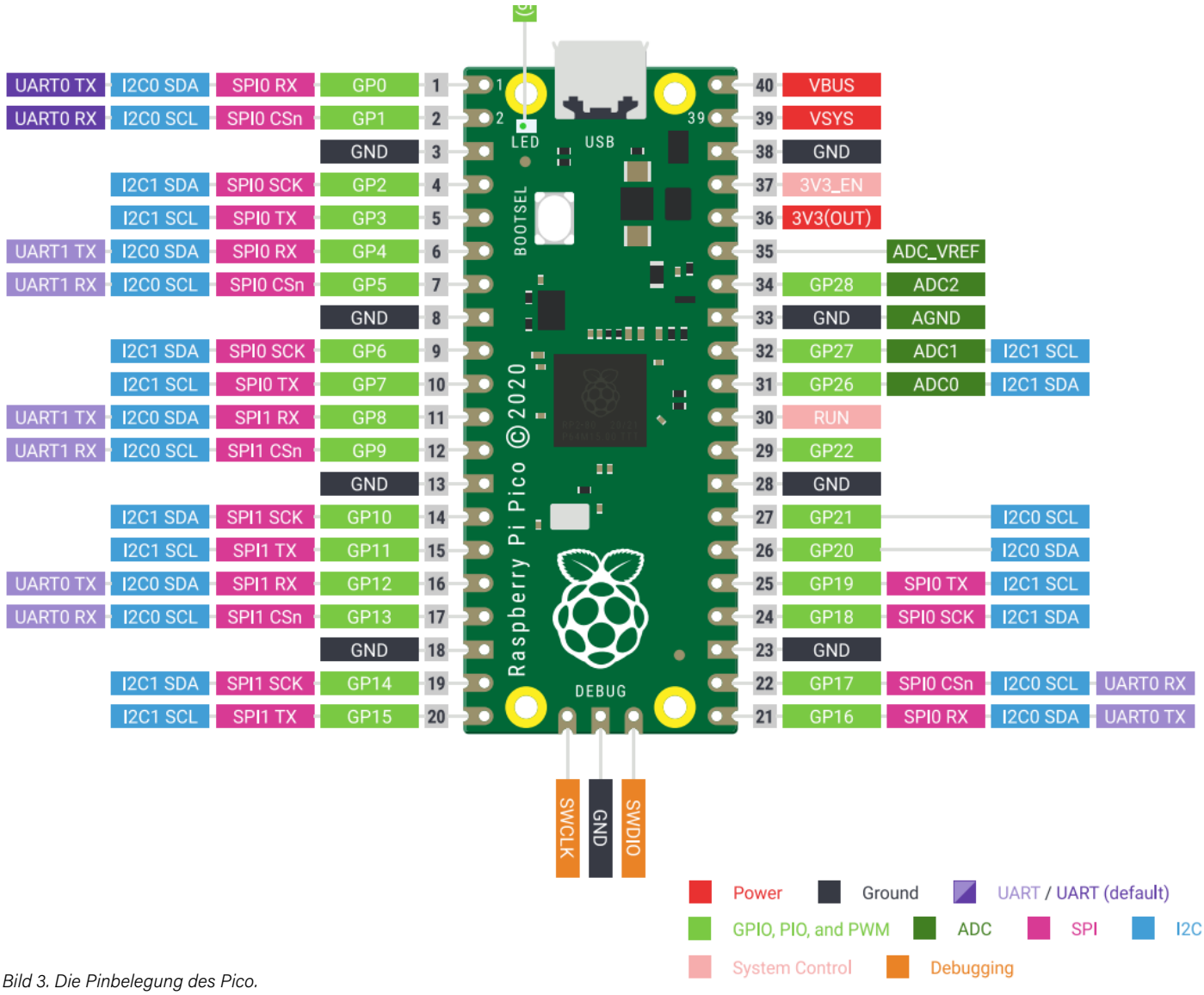


Bild 3. Die Pinbelegung des Pico.

2012: Eine neue Ära mit dem Raspberry Pi

Als der Raspberry Pi im Jahre 2012 als erster Einplatinencomputer auf den Markt kam, eröffnete er eine neue Welt des Physical Computing. Noch nie zuvor war ein Allzweck-PC auf so kleinem Raum und zu so geringen Kosten gebaut worden, aber die Raspberry Pi Foundation bewies, dass dies möglich ist und dass ein Computer nicht nur für das Surfen im Internet genutzt werden kann, sondern auch über umfangreiche interaktive Funktionen verfügt. Mit dem (später) 40-poligen GPIO-Verbinder und vorinstalliertem Python wurde der Raspberry Pi zum idealen Einstieg in den Aufbau automatisierter, IoT-basierter Projekte. Dank des geringen Preises und des leichten Einstiegs konnte jeder mit passablen technischen Fähigkeiten nun Sensoren und Displays an den Pi anschließen und ihn mit Python programmieren, um Zahlen zu berechnen, komplizierte I/O-Aufgaben durchzuführen oder Daten auf einem Webserver anzuzeigen.

Ein ideales Paar: Raspberry Pi und Python

Python ist über das letzte Jahrzehnt hinweg eine tragende Säule des Raspberry Pi gewesen. Während die physikalischen Eigenschaften wie Rechenleistung, Grafik und I/O von Modell zu Modell aufgerüstet wurden, hat sich die Programmiersprache nicht verändert. Ich denke, das liegt zum Teil daran, dass Python so gut mit Prototyping und der Open-Source-Natur des Pi zusammenpasst. Es gibt Dutzende von Python-Bibliotheken, die schnelles Data-Scrubbing, Analysen und maschinelles Lernen ermöglichen. Ähnlich wie die Erweiterbarkeit, die der GPIO bietet, ist Python erweiterbar, modular und flexibel. Webanwendungen wie Jupyter Notebooks ermöglichen eine breite Palette von Arbeitsabläufen in den Bereichen Data Science, wissenschaftliches Rechnen und maschinelles Lernen sowie Plugins, die neue Komponenten hinzufügen und in bestehende integrieren. Es ist also keine Überraschung, dass Python so gut mit dem Raspberry Pi harmoniert.

Vorstellung: **die RP2040-Produkte** von SparkFun

Der Raspberry Pi Pico ist mit dem leistungsfähigen RP2040 aufgebaut, und viele der überzeugenden Eigenschaften eines Boards sind auf die technischen Eigenschaften des Chips zurückzuführen. Bei SparkFun haben wir drei neue Boards mit dem RP2040 gebaut, die Physical-Computing-Eigenschaften des Chips sowie die plattformübergreifenden Entwicklungsumgebungen C/C++ und MicroPython nutzen.

Darüber hinaus haben wir unseren Boards mit dem RP2040 eigene kleine Prise SparkFun-Innovation hinzugefügt. Die Boards, die wir selbst gebaut haben, sind:

- SparkFun RP2040 Thing Plus (**Bild 5**)
- Pro Micro RP2040 (**Bild 6**)
- MicroMod RP2040 Processor Board (**Bild 7**)

All diese Boards verfügen über achtmal so viel Flash wie der Pico. Diese 16 MB an Flash-Speicher sind sicherlich eine Menge, mit der man arbeiten kann, um viele Daten offline zu speichern. Auf jedem dieser RP2040-Boards sollten Machine-Learning-Modelle mit TensorFlow absolut machbar sein. Die Boards besitzen auch alle technischen Eigenschaften des RP2040, als da wären:

- Zwei ARM Cortex-M0+ Prozessoren (bis zu 133 MHz)
- 264 kB embedded-SRAM in sechs Bänken
- Sechs dedizierte IO für SPI-Flash (unterstützt XIP)
- 30 Multifunktions-GPIO: dedizierte Hardware für häufig verwendete Peripherie, programmierbare IO für erweiterte Peripherieunterstützung und vier 12-Bit-ADC-Kanäle mit einem internen Temperatursensor
- Plattformübergreifende C/C++- und MicroPython-Unterstützung mit einfachem Zugriff auf Laufzeit-Debugging und UF2-Boot

Jedes dieser von SparkFun entwickelten RP2040-Mikrocontrollerboards hat jedoch auch seine eigenen Vorteile. Also lassen Sie uns einen genaueren Blick darauf werfen!

SparkFun RP2040 Thing Plus

SparkFun RP2040 Thing Plus ist ein Feather-kompatibles Board mit 18 GPIO-Pins, das sich hervorragend für mobile Projekte eignet, die einen kompletten Satz Peripheriehardware integrieren wollen. Es ist mit einem JST-Halter für einen LiPo-Akku ausgestattet, verfügt über eine Ladefunktion und über eine Batterie-Füllstandsanzeige. Außerdem ist das Board mit einem microSD-Kartenslot zur Datenprotokollierung oder als zusätzliche Speichermöglichkeit ausgestattet. Der RP2040 Thing Plus verfügt über eine adressierbare WS2812-RGB-LED, JTAG-PTH-Pins, vier Montagelöcher und – wie könnte es anders sein – über einen Qwiic-Stecker, der das Prototyping mit Sensoren und Displays erheblich erleichtert.

Pro Micro RP2040

Das Pro Micro RP2040 basiert auf dem Pro-Micro-Footprint und bietet vollständige USB-C-Kompatibilität. Wie das RP2040 Thing Plus verfügt es über eine adressierbare WS2812-RGB-LED, einen Qwiic-Anschluss, eine Boot- und Reset-Taste sowie „castellated" Pads. Diese halben Durchkontaktierungen am Platinenrand prädestinieren den Pico Micro für den Bau eines Controllerboards für Computerprogramme/Spiele oder eines Tastenfeldes. Sie können eine komplett anpassbare Tastatur bauen, die ganz nach ihren Wünschen modifiziert ist. Wenn Sie etwas Inspiration für den Bau einer Tastatur benötigen, die bestimmte Aktionen ermöglicht, schauen Sie sich doch einmal Jason Rudolphs Tastaturprojekt an: https://github.com/jasonrudolph/keyboard.
Wirklich jede Peripheriehardware mit dedizierten Ein- und Ausgängen lässt sich gut mit dem Pro Micro RP2040 realisieren, und das alles in der komfortablen Umgebung von MicroPython.

MicroMod RP2040 Processor Board

Zu guter Letzt bringt SparkFun ein MicroMod-Prozessorboard für den RP2040 heraus. Falls Sie mit dem MicroMod-Ökosystem nicht vertraut sind, lesen Sie einfach den entsprechenden Artikel im Heft oder informieren Sie sich unter https://www.sparkfun.com/micromod. Es handelt sich um ein modulares System, bei dem Träger- und Prozessorplatinen ausgetauscht werden können. Anstatt also Boards mit einem „festen" Controller und bestimmten Arten von Ein- und Ausgängen zu kaufen, können Sie mit MicroMod im vollen Galopp die Pferde wechseln, also Prozessor- oder Trägerboard mitten in der Entwickungsphase tauschen.
SparkFun stellt eine Vielzahl von Trägerplatinen für Ihr Projekt zur Verfügung, die zudem Qwiic-Anschlüsse besitzen, so dass Sie jederzeit weitere Sensoren und Displays zu Ihrem Projekt hinzufügen können. Ab sofort bieten die Trägerplatinen mehrere Optionen, wie Sie den RP2040 nutzen können.

- ATP (alle Pins)
- Eingabe und Anzeige
- Datenerfassung
- Maschinelles Lernen
- Wetter

Neben den anderen Prozessor-Boards (ESP32, Artemis, nRF52840 und SAMD51) können Sie jetzt also auch den RP2040 auf dem MicroMod RP2040 Processor Board nutzen. Damit schöpfen Sie die Vorteile des RP2040 voll aus und können die zusätzliche Peripherie hinzuzufügen, die Sie in Ihrem Projekt benötigen. Wenn Sie schnell und gezielt mit dem RP2040-Chip arbeiten und Resultate sehen wollen, ist das MicroMod RP2040 Processor Board die einfachste Lösung zum Experimentieren und zum Prototyping.

Abschließende Überlegungen

Die Raspberry Pi Foundation hat ein Produkt entwickelt, das leistungsstark und zugänglich ist und kreative Open-Source-Projekte fördert. Der Raspberry Pi Pico, der RP2040 und alle zugehörigen SparkFun-Produkte verfügen nicht nur über eine immense und umfassende Dokumentation, sondern ermöglichen auch, auf innovative Weise mit MicroPython zu entwickeln. Ob es nun der günstige Preis des Pico, das modulare, prototypische Konzept von MicroMod, die kleine Grundfläche des Pro Micro oder die mobilen Fähigkeiten des SparkFun RP2040 Thing Plus ist, was auch immer, Sie sollten ein solches Board in die Hand nehmen und anfangen, Projekte zu bauen, die die vollen Möglichkeiten von MicroPython und Open-Source-Software nutzen. Sagen Sie „Hello“ zu Ihrem neuen Lieblings-Mikrocontroller auf Raspberry-Pi-Basis!

200711-03

Zugelassene Raspberry Pi-Händler

Sind Sie am Raspberry Pi und RPi-Zubehör interessiert? Sowohl SparkFun als auch Elektor sind „Approved Raspberry Pi Reseller“.

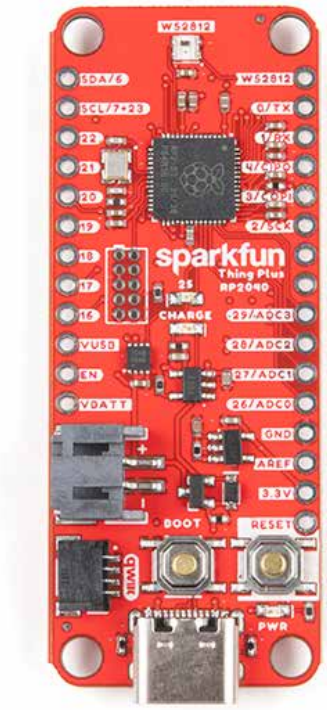


Bild 5. SparkFun Thing Plus

Bild 6. SparkFun Pro Micro

Bild 7. MicroMod RP2040 Processor Board

Passende Produkte

Suchen Sie die in diesem Artikel erwähnten Produkte? SparkFun und Elektor haben die sie!

- **MicroMod RP2040 Processor Board**
  www.elektormagazine.de/esfe-en-rpipico1
- **Pro Micro RP2040**
  www.elektormagazine.de/esfe-en-rpipico2
- **Raspberry Pi Pico Microcontroller Board**
  www.elektormagazine.de/esfe-en-rpipico3
- **SparkFun RP2040 Thing Plus**
  www.elektormagazine.de/esfe-en-rpipico4

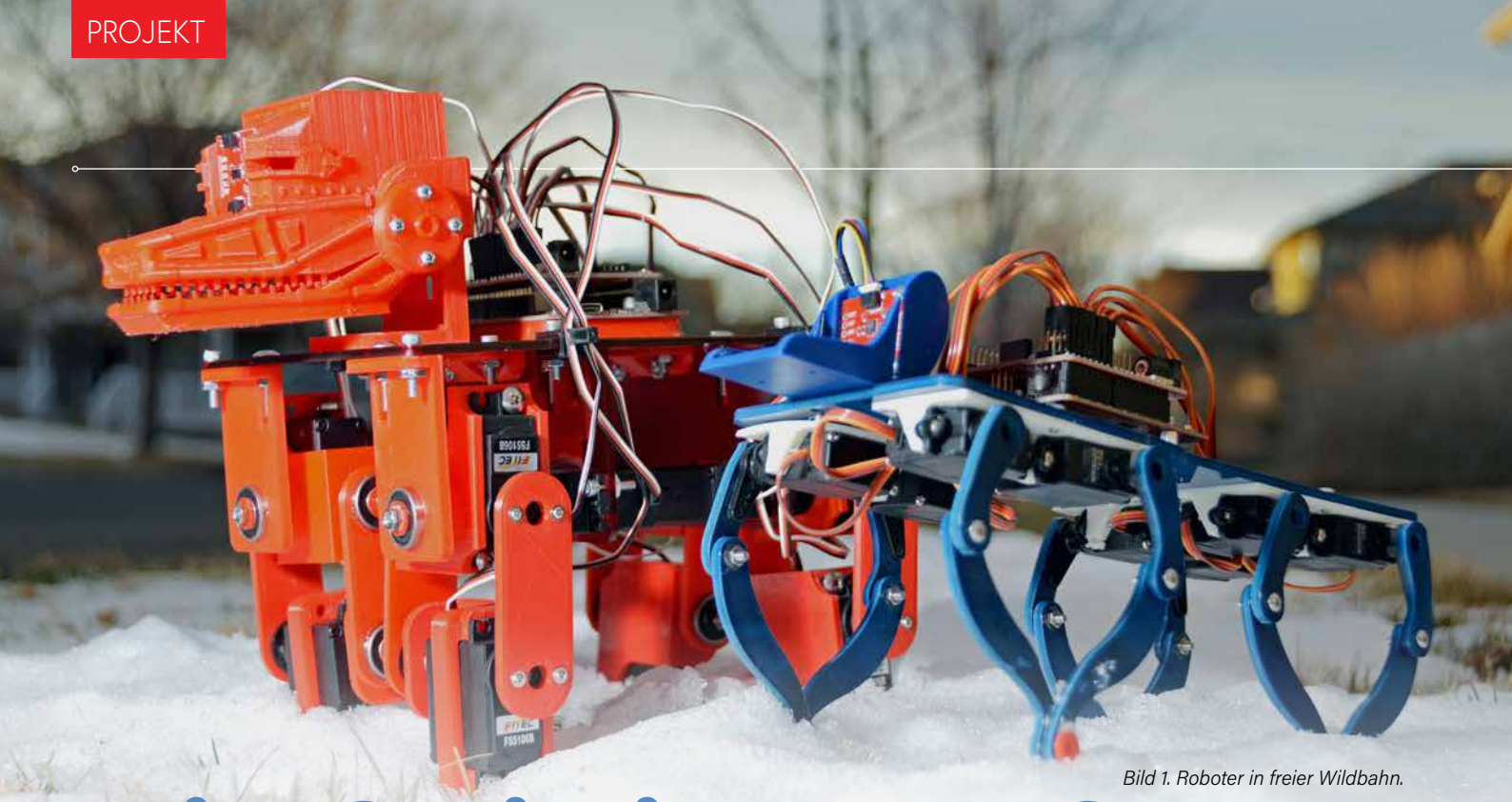


Bild 1. Roboter in freier Wildbahn.

Vierbeiniger Roboter selbstgebaut

Von Rob Reynolds (USA)

Wollen Sie einen vierbeinigen Roboter bauen? Natürlich wollen Sie das! Lesen Sie weiter, um zu erfahren, wie man zwei erweiterbare Plattformen baut, von denen jede mehrere Löcher für die Befestigung von Servos, das Hinzufügen von Sensoren und das Anbringen von Mikrocontrollern hat.

Ich weiß nicht, ob ich für alle Maker sprechen kann, aber ich denke, dass die meisten von uns sich in der Kindheit „Roboter" als zweibeinige, sich etwas staksig bewegende Humanoide mit interaktiven Fähigkeiten vorstellten, die wir heute als KI bezeichnen würden. Aber das waren Träume oder Kreationen in Filmen oder in unseren Köpfen, Kreaturen, die von Menschen in metallischen Kostümen, von unsichtbaren Puppenspielern zum Leben erweckt wurden oder von Genies wie Ray Harryhausen mit seiner Robotereule Bubo aus *Kampf der Titanen*. Nun bin ich erwachsen, und stellen Sie sich meine Überraschung und Enttäuschung vor, als ich herausfand, dass meine Haushaltsroboter ein Staubsauger und mein „Robot Pâtissier" sind, obwohl wir ihn in den Staaten einfach Küchenmaschine nennen und er eigentlich gar nichts Roboterhaftes tut. Es sieht so aus, als hätte ich nur eine Möglichkeit, an einen richtigen Roboter zu kommen: Ich muss ihn selbst entwerfen (**Bild 1**)!

Der Entwurf im Kopf

Bei der Entwicklung eines Roboters - oder, ehrlich gesagt, bei jedem Projekt - ist mir der Anfang am liebsten, der Augenblick, in dem noch alles möglich ist. In Stephen Sondheims Bühnenmusical „Sunday in the Park with George" sagt dieser George, als er sich an seinen Großvater, den Maler George Seurat, erinnert: „Weiß. Eine leere Seite oder Leinwand. Sein Favorit. So viele Möglichkeiten." Ja, am Anfang gibt es so viele Möglichkeiten. Das mag einfach daran liegen, dass ich am Anfang nicht genug weiß (und hoffentlich auch nie wissen werde), was nicht möglich ist. In meinem Kopf ist alles möglich.

Wie soll mein Roboter also aussehen? Für den „Wow"-Faktor könnte es ein glatt beschichteter, zweibeiniger humanoider Roboter sein, wie im Buch/Film *I, Robot*. Wenn ich eher praktisch denke, sollte ein Roboter sich leicht in meinem Haus bewegen und mit mir oder seiner Umgebung interagieren. Dann wäre ein Roboter auf vier Rädern wahrscheinlich der einfachste Weg. Doch damit wären wir im Grunde wieder beim „Staubsaugerroboter" angelangt. Ich müsste einen Mittelweg finden, etwas, das ein wenig praktisch ist, aber immer noch beeindruckend genug, um die Leute zweimal hinschauen zu lassen. Außerdem wollte ich ihn im Laufe der Zeit verändern und verbessern können. So entschied ich mich schließlich für einen Vierbeiner.

Ich besuchte die Webseite des Robotik-Unternehmens *Boston Dynamics* und ließ mich von deren Arbeit mit Vierbeinern inspirieren.

Vom großen und etwas unbeholfenen *Big Dog* aus dem Jahr 2005, der über ein offenes Feld rumpelt, bis zum neuen *Spot*, der geschmeidig und agil ist, Türen öffnet und zu *Uptown Funk* tanzt. Dies ist ein schönes Beispiel für Fortschritt und Iteration im Laufe der Zeit. Für meinen Vierbeiner wollte ich eine Basisplattform bauen, die sich erweitern oder vielleicht auch modular gestalten ließ. Ich spielte mit verschiedenen Entwürfen und Konfigurationen und entschied mich schließlich für zwei verschiedene, einer von Big Dog und der andere von Spot inspiriert. Beide würden unterschiedliche Fähigkeiten besitzen, beide würden erweiterbar sein.

Zuerst wollte ich den kleineren, schlankeren Spot-ähnlichen Roboter bauen. Ich hatte eine Vorstellung von der Mechanik, die ich verwenden wollte, und entwarf ein schönes gerades, glattgebügeltes, aber auch irgendwie uninteressantes Chassis. Aber dann dachte ich ein wenig mehr über Spot von Boston Dynamics nach und erkannte, dass ich durchaus die Form der Funktion folgen lassen könnte. Ein paar Ecken und Kanten mehr als nötig würden den Roboter ein wenig besser aussehen lassen, vielleicht nicht mehr ganz so stabil, aber auch nicht ganz so steril.

Dann begann ich mit der Arbeit am zweiten Roboter, dem eher industriell aussehenden, von Big Dog inspirierten Roboter. Nach einigen Stunden am Zeichenbrett kam mir der Gedanke, dass ich das Rad wahrscheinlich nicht neu erfinden müsse. Sicherlich gäbe es einige Entwürfe, die ich verwenden könnte, zumindest als Ausgangspunkt. Und so fand ich ein tolles Design von Technovation auf Instructables [1].

Beide Plattformen sind mit vielen Löcher versehen, an denen Servos, Sensoren und Mikrocontroller-Boards angebracht werden können. Ich hatte vor, das SparkFun Redboard Artemis zu verwenden, das so groß ist wie der Arduino Uno, aber viel mehr Speicher und Geschwindigkeit und außerdem BLE bietet. Durch den einheitlichen Formaktor kann ich leicht auf das SparkFun Artemis ATP erweitern, wenn ich mehr I/O-Pins benötige. Außerdem wollte ich das SparkFun Wireless Motor Driver Shield verwenden.

Die Beine des Vierbeiners

Für die Beine habe ich unterschiedliche Wege eingeschlagen. Beim kleineren Roboter - nennen wir ihn mal Bluesette - habe ich eine Fünffach-Gestängekopplung verwendet. Dies mag vielleicht nicht ideal für einen vierbeinigen Roboter sein, besonders wenn es mit Servomotoren angetrieben wird, aber das Experiment lohnt sich, um ein paar Erkenntnisse zu Bewegung und Inverser Kinematik zu gewinnen. Das Fünffach-Gelenk ist ein Mechanismus mit zwei Freiheitsgraden, bei dem alle fünf Glieder in einer Schleife verbunden sind (**Bild 2**).

Dieser Mechanismus steuert die x- und y-Koordinaten des Gelenks D, das als Endpunkt oder Endeffektor bezeichnet wird (oder einfach gesagt, der Roboterfuß), durch gleichzeitiges Einstellen der Eingangswinkel theta 1 und theta 2, wodurch die Winkel der Stangen B2 und B5 gesteuert werden. Indem wir den Endeffektor jedes Beins entlang einer elliptischen Bahn bewegen, können wir den Roboter vorwärts und rückwärts laufen lassen, seine Höhe ändern und ihn sich sogar im Kreis drehen lassen. Wie ich schon sagte, ist dies vielleicht nicht die beste Methode der Fortbewegung mit Servomotoren für einen Vierbeiner, aber wenn man acht bürstenlose Hochleistungsmotoren verwendet, kann dieser Aufbau sehr effektiv sein. Ein großartiges Beispiel dafür ist das Stanford Doggo Project [2]. Mit der Geschwindigkeit, Leistung und Steuerung der bürstenlosen Motoren kann dieser kleine Vierbeiner über einen Meter hoch springen und sogar Rückwärtssaltos machen (**Bild 3**).

Mein größerer, klobigerer Roboter, den ich Big Red nenne, verwendet den bei vierbeinigen Robotern üblichen Mechanismus, den sogenannten Seriellen Manipulator. Dabei handelt es sich um einen Mechanismus, bei dem jedes Gelenk seinen eigenen Motor hat, von der Basis bis zum Endaktuator. Diese Gelenke werden oft als Schulter-, Ellbogen-

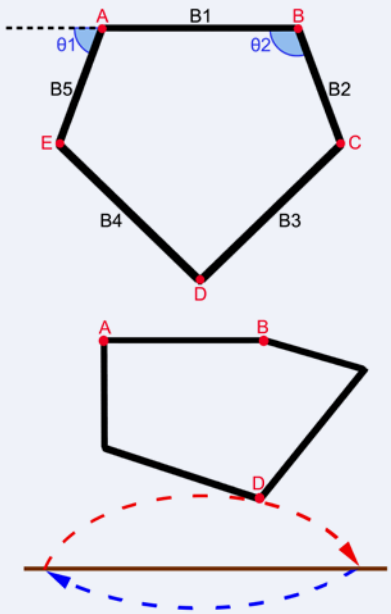


Bild 2. Der fünfgliedrige Mechanismus.

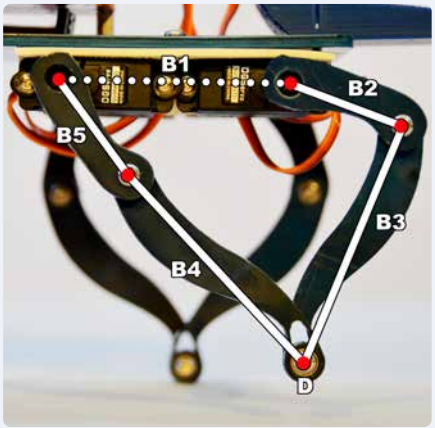


Bild 3. Beide Systeme verwenden Servos zur Steuerung der Beinposition, jedoch auf sehr unterschiedliche Weise.

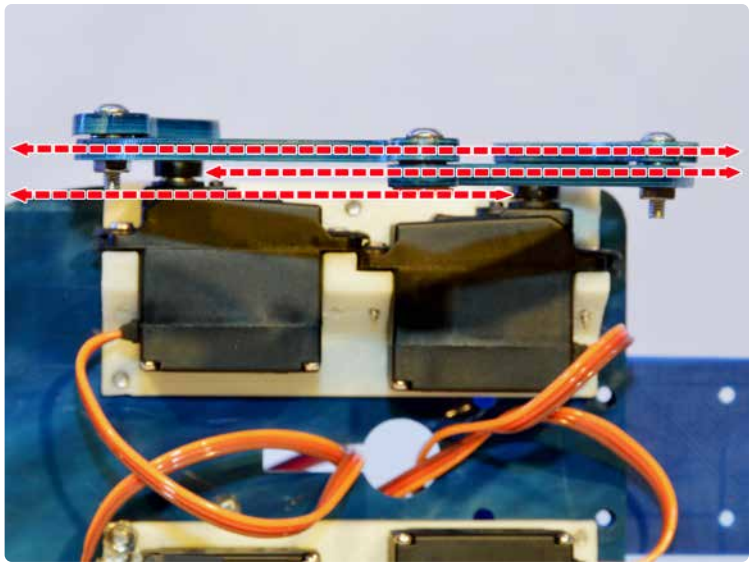


Bild 4. Durch den Versatz der Servo-Halterungen um die Breite des Materials bleibt alles im Lot.

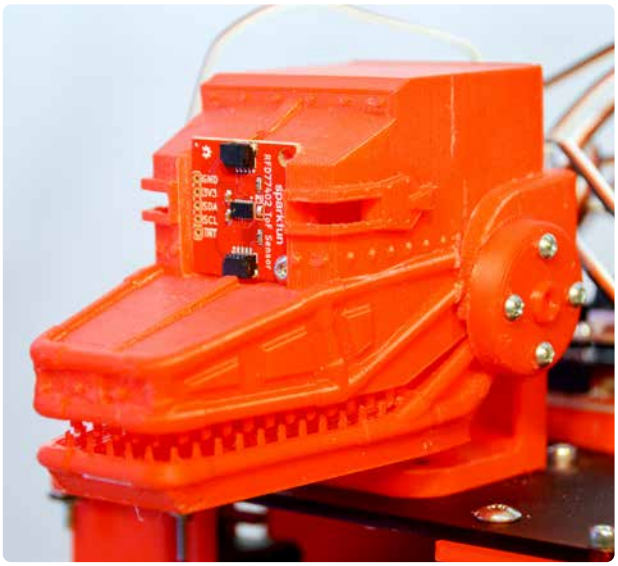


Bild 5. Der Kopf des Hundes mit einem Entfernungssensor.

und Handgelenke bezeichnet, insbesondere bei anthropomorphen Armstrukturen, etwa einem einarmigen Bestückungsautomaten in der Automobilproduktion. Da ich die Arme für die erste Version von Big Red kurz gehalten habe, konnte ich ihm Kraft und Stabilität verleihen, die die fragilen Beine von Bluesette nicht bieten können.

Was die Konstruktion und die Materialien angeht, habe ich eine Kombination aus lasergeschnittenem 3-mm-Acryl- und 3D-gedruckten PLA-Teilen verwendet. Viele lokale Bibliotheken und Makerspaces haben 3D-Drucker und einige auch Laserschneider. Und selbst wenn Sie keinen Zugang zu einem Laserschneider haben, können Sie einfach 3-mm-Sperrholz verwenden, das mit Handwerkzeugen geschnitten, gesägt und gebohrt werden kann. Die 3D-gedruckten Teile, insbesondere die Servohalterungen, sind aber wirklich wichtig. Aber auch ohne Zugang zu einem 3D-Drucker können Sie vielleicht Kabelbinder oder Heißkleber verwenden, um Ihre Motoren zu fixieren, zumindest bei dem Roboter mit den fünfgliedrigen Beinen. Werden Sie kreativ! Schließlich sind wir Ingenieure und Maker!

Auch wenn sie nicht unbedingt notwendig sind, verwende ich bei beiden Robotern Kugellager an allen Gelenken. Das sorgt für viel sanftere Bewegungen, reduziert die Reibung und verringert somit den Strombedarf der Servomotoren. Ein weiteres wichtiges Konstruktionsmerkmal für Bluesette ist der Versatz der Schrittmotoren um 3 mm (die Dicke des für die Beine verwendeten Materials). Wenn die Schrittmotoren auf der gleichen Ebene lägen (**Bild 4**), würde dies zu einer seitlichen Torsion an den Gelenken und dies wiederum zu einem höheren Strombedarf der Servomotoren führen.

> *Für meinen Vierbeiner wollte ich eine Basisplattform, auf der ich erweitern oder Module austauschen kann.*
>
> Rob Reynolds

Ich wusste, dass ich eine Art vordere Halterung für die Sensoren machen musste. Ich hätte schnell eine einfache Halterung entwerfen können, aber hier hatte ich wieder eine Chance, kreativ zu werden. Für Bluesette habe ich einfach ein paar primitive Formen zusammengestellt, die harten Kanten gefast, ein paar Befestigungslöcher angebracht und das Ganze ausgedruckt. Big Red erhielt etwas mehr meiner Zeit und Aufmerksamkeit. Ich erinnerte mich an den Film *Isle of Dogs*, in dem es einige wirklich gut aussehende Roboterhunde gab. Ich nutzte einige Standbilder aus dem Film, dehnte meine 3D-Design-Muskeln und bildete die Köpfe der Roboterhunde nach, wobei ich ein paar Anpassungen für die meisten der Qwiic-Sensorplatinen vornahm (**Bild 5**). Ich befürchtete zwar, dass der Roboter dadurch ein wenig frontlastig werden könnte, aber ich könnte das ja mit einem großen Akkupack am Heck ausgleichen.

Die Metallteile, die ich für die Montage beider Roboter verwendet habe, waren M3-Schrauben in verschiedenen Längen (mit Ausnahme der 4-40er Abstandshalter). Für die Lager von Big Red und die Handgelenke von Bluesette habe ich einige Lagerschalen 3D-gedruckt.

Der Programmcode

Eine kurze Voranmerkung zum Codieren und Testen. Wenn man etwas entwirft und baut, das sich bewegen kann, dann wird es sich bewegen, und zwar meist zu den unerwartetsten und unpassendsten Zeitpunkten. Diese Lektion lernte ich zum ersten Mal bei der Arbeit an einem kleinen Roboterauto. Ich hatte meinem Sketch etwas Code hinzugefügt und wollte ihn auf das Fahrzeug hochladen. In dem Moment, in dem das Hochladen abgeschlossen war, startete mein kleines Auto mit voller Geschwindigkeit, flog von meiner Werkbank und krachte gegen die gegenüberliegende Wand. Nathan, der Gründer von SparkFun, hatte das gleiche Problem mit einer seiner Konstruktionen, aber diese Konstruktion war ein autonomes Fahrzeug, das groß genug war, um darin zu fahren. Als es losjagte, riss es seinen Laptop vom Schreibtisch.

All das soll nur andeuten, dass man immer dafür sorgen sollte, dass die Teile, die Ihre Kreation in Bewegung setzen, keine Bodenberührung haben sein sollten. Wenn Sie an einem Auto arbeiten, ist eine Kiste geeeignet, die unter den Fahrzeugboden passt und höher als seine Bodenfreiheit ist. Für meine Roboter habe ich einige kleine T-Nutschienen aus Aluminium mit einem schnell hergestellten 3D-Teil verbunden, um die Konstruktion in die Luft zu heben (**Bild 6**).

Genug der warnenden Vorrede, es ist an der Zeit, mit dem Schreiben des Codes zu beginnen (**Listing 1**). Unsere vorrangige Ressource ist die Servo-Bibliothek, obwohl, wenn wir vorankommen und zusätzliche Fähigkeiten implementieren, sicherlich auch andere Biblio-

theken ins Spiel kommen werden. Zuerst soll unser Sketch die Servo-Objekte erstellen und sie alle aneinander hängen. Das geht schnell und einfach, indem wir einfach ein paar kleine Anpassungen an so etwas wie dem Servo-Sweep-Sketch vornehmen. Um alles, was wir jetzt und in Zukunft tun, einfacher zu halten, können wir die Servos in einer `for`-Schleife aneinander reihen. Wenn Sie vorhaben, mit diesen oder anderen Robotern ähnlicher Bauart weiterzumachen, ist dies ein guter Ansatz, da Sie die Anzahl der Servo-Objekte, die Sie erstellen und aneinander hängen, einfach durch Ändern des Schleifendurchläufe anpassen können. Dann definieren wir eine Startposition für jeden Servo und machen einen kurzen Sweep-Test, um sicherzustellen, dass sie alle kommunizieren und richtig mit Strom versorgt werden. Wenn ich einen Code beginne, der wachsen und wachsen und vielleicht mehrere Iterationen haben wird, teile ich den Code in Funktionen auf, anstatt alles in der `void:loop()` unterzubringen. Dies macht es einfacher, Dinge zu finden und zu ändern, und erleichtert auch das Kopieren und Einfügen in andere Versionen des Sketches oder gar völlig andere Sketches, die vielleicht nur ein kleines Stück eines bestehenden Codes benötigen. Vielleicht liegt das daran, dass ich alt genug bin, um mich an BASIC und die Verwendung von Unterprogrammen zu erinnern.

Passende Produkte

Suchen Sie die in diesem Artikel erwähnten Produkte? SparkFun und Elektor haben sie!

- **Elektor MIT App Inventor-Paket**
  www.elektor.de/elektor-mit-app-inventor-bundle
- **SparkFun RedBoard Artemis**
  www.elektormagazine.de/esfe-en-qrobot1

Sobald das erledigt ist, haben Sie einen Ausgangspunkt. Die nächsten Schritte erfolgen, zumindest bei mir, in aufsteigender Reihenfolge der Schwierigkeit. Die Höhe des Roboters auf vielleicht die Hälfte seiner vollen Höhe zu verstellen, ist recht einfach. Von dort aus wäre der nächste logische Schritt, ein paar Gehfunktionen zu implementeieren, sowohl vorwärts als auch rückwärts, gefolgt vom Drehen in beide Richtungen. Dann können wir zur BLE-Steuerung oder zur autonomen Bewegung übergehen, je nachdem, was Ihr Ziel für den Roboter ist. Eine Verbindung zu Ihrem Laptop oder Tablet über Bluetooth und das Senden von seriellen Befehlen mit einzelnen Zeichen wäre ein schneller und einfacher Weg, um Ihren Roboter fernzusteuern. Sie könnten ihn auch mit einem micro:bit koppeln, um eine portable Fernbedienung zu erhalten. Wenn Sie Ihren Roboter von einem Android-Handy aus steuern möchten, könnten Sie die App Inventor vom MIT (https://appinventor.mit.edu/) verwenden, um ein Interface auf einem Steuergerät zu erstellen, das Sie immer bei sich tragen. Und wenn Sie sich noch nicht sicher sind, was Sie wollen, können Sie immer eine Funktion für jede Möglichkeit schreiben und dann jene auskommentieren, die Sie nicht verwenden möchten. Näherungssensoren sind eine beliebte Erweiterung für jeden Roboter, und sie scheinen als nächster Schritt am sinnvollsten zu sein. Vielleicht möchten Sie einen Klimasensor für Temperatur oder Luftfeuchtigkeit hinzufügen? Der letzte und „höchste" Schritt wäre das Hinzufügen von grundlegendem maschinellem Lernen oder KI. Mit TensorFlow dürfte es relativ einfach, Ihren Roboter so zu trainieren, dass er auf einfache Sprachbefehle reagiert - stopp, los, sitz, fass, was immer Sie wollen!

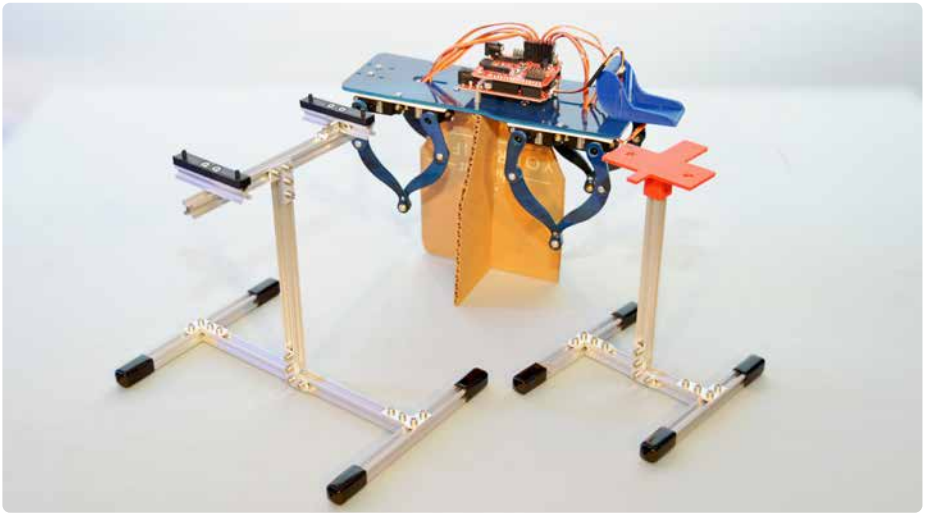


Bild 6: Ich habe zwar beide Ständer aus Aluminiumteilen gebaut, aber man könnte auch etwas so Einfaches wie Wellpappe verwenden.

Bereit, Ihren eigenen Roboter zu bauen?

Wie die meisten meiner Projekte sind auch diese beiden „in Arbeit" und werden, wenn Sie diese Zeilen lesen, schon viel weiter sein als jetzt. Ich veröffentliche alle alten und neuen Funktionen stets brandaktuell in meinem GitHub-Repository (https://github.com/ThingsRobMade/QuadrupedRobots). Wie auch immer, wenn Sie erwägen, Ihren eigenen Roboter zu bauen, warten Sie bitte nicht darauf, zu sehen, was ich so mache. Legen Sie einfach mit Ihren eigenen Ideen los! Eines der großartigen Dinge an der Open-Source-Community ist die Tatsache, dass wir alle zusammen schlauer sind als einer von uns allein, und wo ich vielleicht stolpere, wird sich einer von Ihnen zweifellos etwas so Brillantes einfallen lassen, dass es mich auf neue Art und Weise oder in neue Richtungen inspirieren wird. Bis dahin: Träumen Sie weiter, entwerfen Sie weiter, programmieren Sie weiter!

200712-03

WEBLINKS

[1] Technovation, „3D Printed Arduino Powered Quadruped Robot," Instructables, 2020: https://www.instructables.com/3D-Printed-Arduino-Powered-Quadruped-Robot/

[2] Nate711, „StanfordDoggoProject," GitHub, 2019: https://github.com/Nate711/StanfordDoggoProject

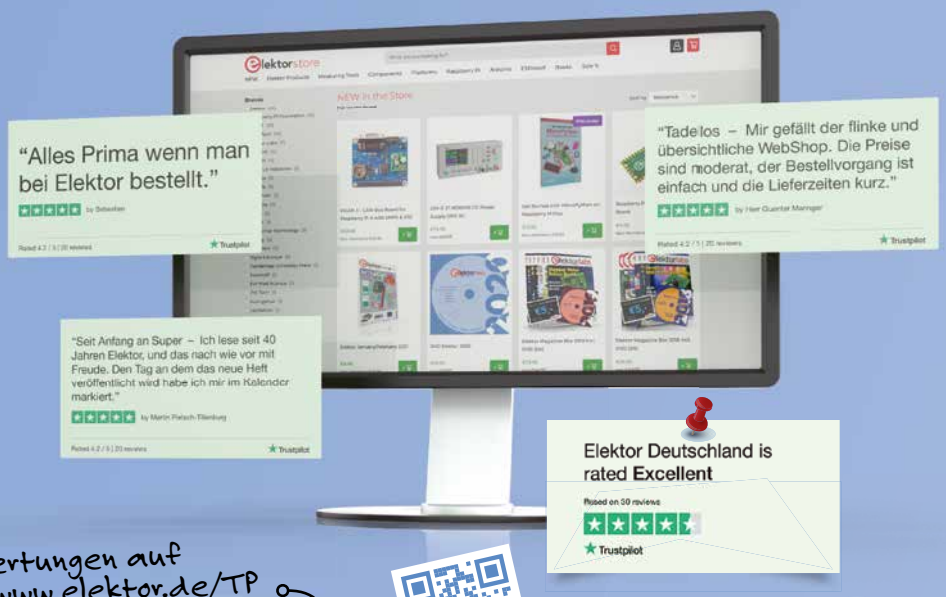
Listing 1. Die Gelenke des Roboters werden in eine neutrale Position gebracht und dann leicht hin und her geschwenkt.

```
/*
* Quadrupedal Robot Setup Sketch
* Rob Reynolds
* SparkFun Electronics
*
* This simple sketch sets the joints of a
* quadrupedal robot to a neutral position,
* then swings them slighty back and forth
* simply to test control and movement.
* Currently setup for 8 DoF, but can easily
* be adjusted by changing to integer in
* Servo leg[*] to correspond to the
* number of servos.
*/

#include <Servo.h>
 Servo leg[8];   // create servo objects to control a servo
int pos = 90;    // variable to store the servo position

void setup() {
  delay(1000);
  for (int l = 0; l < 8; l++){
    leg[l].attach(l + 2); // attach servos sequencially starting with pin 2
    delay(100);
  }
  delay(1000);
  for (int l = 0; l < 8; l++){
    leg[l].write(pos); // set all servos to 90°
    delay(100);
  }
}

void loop() {
  allServoTest(); // Initial test or all servo movement
  while(1){
  }
}

void allServoTest(){
  // Initial movement test for all servos. All servos are set to 90°.
  // then swept front and back before returning to 90° once again.
   for (int i = 0; i < 8; i++){
    for (pos = 90; pos <= 135; pos += 1) {
      leg[i].write(pos);                  // tell servo to go to position in variable 'pos'
      delay(10);                          // wait 10ms for the servo to reach the position
    }
    for (pos = 135; pos >= 45; pos -= 1) {
      leg[i].write(pos);                  // tell servo to go to position in variable 'pos'
      delay(10);                          // waits 10ms for the servo to reach the position
    }
    for (pos = 45; pos <= 90; pos += 1) {
      leg[i].write(pos);                  // tell servo to go to position in variable 'pos'
      delay(10);                          // waits 10ms for the servo to reach the position
    }
  }
}
```

Jede Bewertung spiegelt ein persönliches Erlebnis wider

Wir lieben Elektronik und Projekte, und wir setzen alles daran, die Bedürfnisse unserer Kunden zu erfüllen

Der Elektor-Store: '**Never expensive, always surprising**'



Sehen Sie sich weitere Bewertungen auf unserer Trustpilot-Seite an: www.elektor.de/TP

Oder bilden Sie sich selbst eine Meinung und besuchen Sie unseren Elektor Store, www.elektor.de

elektor
design › share › sell

SPARKFUN SERVICES

Entwicklung, Logistik und Schulungen

Die Dienstleistungen von SparkFun bringen Mehrwert. Nutzen Sie unsere Expertise und Fähigkeiten, die Ihnen helfen, mehr zu erreichen.
Vom Platinendesign und kunden-spezifischer Beschaffung bis hin zu Versand und Training, wir sind bereit, Ihnen zu helfen.

BESUCHEN SIE FÜR MEHR INFORMATIONEN

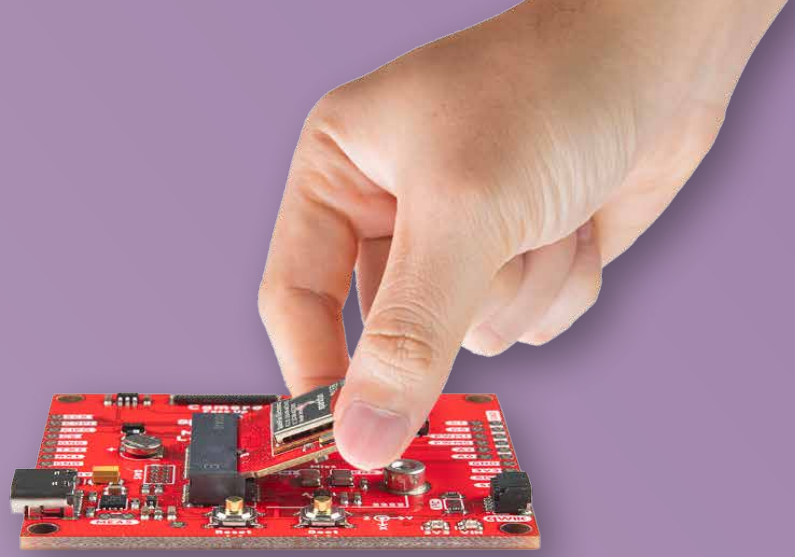
www.sparkfun.com/services

sparkfun
START SOMETHING

MicroMod

Ein **modulares Ökosystem** aus tauschbaren Controller- und Trägerplatinen, das schnelles Prototyping und Entwicklung ermöglicht.

Welche Kombination wählen Sie?



Daten protokollieren

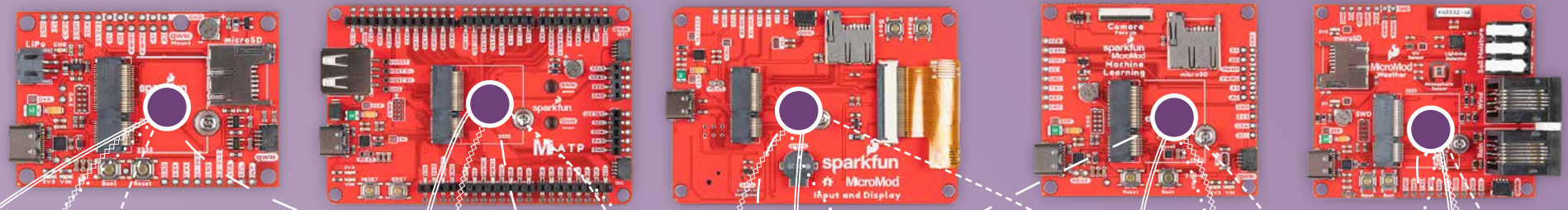
ATP

Sichtbarkeit von Daten und Eingaben hinzufügen

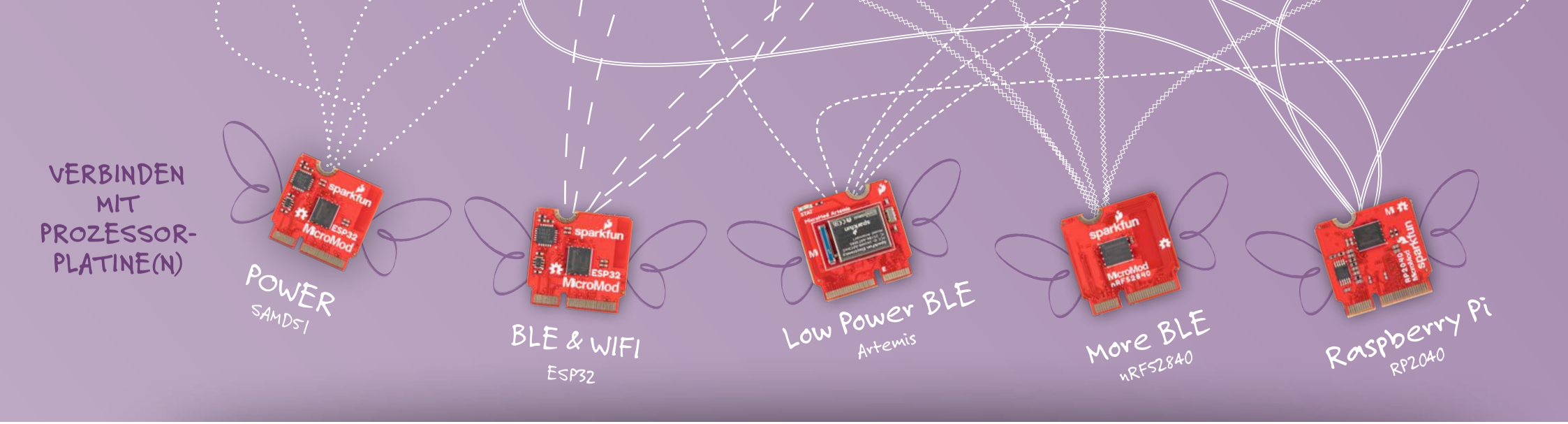
Maschinelles Lernen

Wetter

WÄHLEN SIE EINE TRÄGER-PLATINE



Trägerplatine + Prozessorboard(s) = Rapid Prototyping



Änderung des Projekts? Kein Problem!

MicroMod ist ein modulares Ökosystem, das ein Mikrocontroller- oder „Processor-" Board" mit verschiedenen Peripheriebausteinen auf „Carrier-Boards" verbindet. Der M.2-Standard ermöglicht eine schnelle Entwicklung, schnelles Prototyping und schnelle Modifikationen, wobei der Code nur wenig oder gar nicht angepasst werden muss. Erweitern oder ändern Sie die Fähigkeiten Ihres Projekts einfach durch Austausch des Prozessors, der Trägerplatine oder durch Verwendung des *Qwiic Connect Systems*. Sie finden keine passende Kombination aus Träger- und Controllerplatine? MicroMod ist eine Open-Source-Produktlinie, was bedeutet, dass Sie Ihre eigene Platine erstellen können! SparkFun stellt Eagle-Bibliotheken und Tutorials zur Verfügung, wie Sie Ihr eigenes Board entwerfen können.

Erfahren Sie mehr und bestellen Sie Ihre eigene MicroMod-Kombi!

www.elektormagazine.de/poster-micromod

elektormagazine.de/posters

elektor
design > share > sell

RISC-V-IoT-Entwicklung mit FreeRTOS-Bibliotheken für AWS

Von Avra Saslow (USA)

Die Philosophie von Open-Source-Software hat sich in den letzten Jahrzehnten als immens wichtig für Forschung und Wissenschaft erwiesen. Die Idee ist, dass jeder dazu beitragen kann, ein besseres Projekt/Produkt zu erstellen; das macht Technologie letztendlich zugänglicher und zuverlässiger. Open-Source ermutigt Experten, gemeinsam an einem Projekt zu arbeiten, was Qualität und Sicherheit garantiert und letztendlich ein hervorragendes Produkt ohne proprietäre Einschränkungen hervorbringt. Dieser Artikel ist ein Leitfaden für die Umsetzung dieser Konzepte in der realen Welt.

RISC-V und FreeRTOS

Nach jahrelangem Erfolg im Software-Bereich hat auch die Hardware-Industrie endlich Zugang zu einer Open-Source-Technologie gewonnen: RISC-V könnte die Zukunft der Mikrocontroller komplett verändern. Traditionell beschreibt die ISA (Instruction Set Architecture) auf einem Board, wie Software und Hardware miteinander kommunizieren, und dieses Zusammenwirken ist normalerweise durch Lizenzen, Lizenzgebühren und NDAs (Non-Disclosure Agreements) verschlossen.

Im Gegensatz dazu wird das RISC-V-ISA unter Open-Source-Lizenzen zur Verfügung gestellt. Folglich scheint die RISC-V-Architektur das Geschäftsmodell der Technologiebranche komplett zu verändern. Anstatt dass sich ein Unternehmen für eine ISA entscheiden muss und damit an die Bibliothek des Anbieters gebunden ist, ermöglicht RISC-V den Unternehmen, den Core an seine speziellen Bedürfnisse anzupassen und in beide Richtungen zu skalieren.

Das hier beschriebene Projekt macht sich die Open-Source-Hardware zunutze, indem es den Echtzeit-Betriebssystem-Kernel für Embedded-Controller namens FreeRTOS implementiert, der flexible Multi-Threading-Lösungen ermöglicht.

Stellen Sie sich das so vor: Bei einem gewöhnlichen Mikrocontroller wie dem Arduino ist der Programmablauf traditionell innerhalb einer Endlosschleife angesiedelt, in der alle Aufgaben des Systems enthalten sind. Wenn das Programm startet, führt es (nach der Setup-Funktion) die Endlosschleife mit einer Reihe von Tasks aus, bis das Programm einen Interrupt auslöst.

Diese Tasks können von Sensoren lesen, auf Displays schreiben oder Zahlen berechnen; egal, was die Tasks tun, sie werden sequentiell ausgeführt. Diese Art von Programmstruktur ist gut geeignet, wenn keine Hochleistungaufgaben zu erfüllen sind.. Sie arbeitet zwar schnell, aber sie ist nicht besonders gut für die parallele Abarbeitung mehrerer Aufgaben geeignet. Hier kommt ein Echtzeit-Betriebssystem wie FreeRTOS ins Spiel, das eine minimale Interrupt-Latenz und eine minimale Latenz für die Thread-Umschaltung aufweist.

Minimale Interrupt-Latenz bedeutet, dass das Betriebssystem auf eine

Bild 1. Single-Thread vs. Multithread.

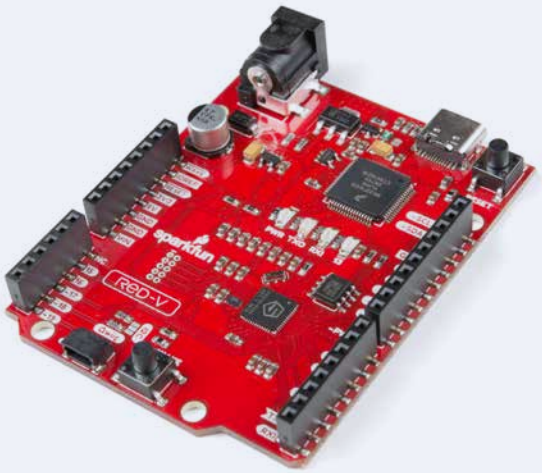


Bild 2. Das RED-V-Board von SparkFun basiert auf einem leistungsfähigen RISC-V-Controller.

minimale Zeitspanne getrimmt ist, die zwischen dem Ausführen von Tasks oder Unterprogrammen vergeht. Die minimale Thread-Umschaltlatenz ist die minimale Zeit, die das Betriebssystem benötigt, um die CPU (mit einzelnem Kern) von einem zu einem anderen Unterprogramm umzuschalten. Diese minimalen Latenzen sind im Wesentlichen das, was ein RTOS zu einem Multitasking-Betriebssystem macht, das die Tasks gleichzeitig und nicht sequentiell ausführt, wie in **Bild 1** dargestellt.

Während Sie für Programme, die von Sensoren lesen, Berechnungen mit den Daten durchführen und auf ein LCD-Display schreiben, vielleicht kein RTOS benötigen, kann Multithreading für Anwendungen wie die Interaktion mit Wireless-Stacks, die eine schnelle Reaktion auf Ereignisse erfordern, sehr nützlich werden. FreeRTOS wurde speziell für Embedded-CPUs entwickelt, so dass es perfekt geeignet ist, um den Betriebssystemkern auf ein RISC-V-Board zu laden und über Wireless-Bibliotheken mit AWS (Amazon Web Services) zu kommunizieren.

RED-V mit FreeRTOS-Kernel/Bibliotheken und AWS-Cloud-Anbindung

Heutzutage ist es für die Entwicklung einer Anwendung unerlässlich, von einer Art Modell-Design auszugehen, bei dem ein IoT-Gerät zusammen mit einer sicheren Cloud-Service-Plattform betrieben wird. Dieses Projekt erweitert den FreeRTOS-Kernel und nutzt Applikations-Protokoll-Bibliotheken, um der mikrocontroller-basierten IoT-Anwendung die nötige Konnektivität bereitzustellen, so zum Beispiel beim SparkFun RED-V-Board (**Bild 2**), das auf RISC-V aufgebaut ist. Die Struktur, an die wir uns für unsere Anwendung halten müssen, ist in **Bild 3** dargestellt.

Glücklicherweise kann FreeRTOS einfach über die AWS-Plattform implementiert werden. Um eine HTTP-Server-Verbindung über AWS IoT zu konfigurieren, müssen Sie einen Benutzer im AWS-Abschnitt *Identity and Access Management (IAM)* erstellen. Dieser Benutzer erhält die Berechtigung, auf die AWS-Policies *AmazonFreeRTOSFullAccess* und *AWSIoTFullAccess* zuzugreifen. Das RED-V-Board muss außerdem bei AWS IoT registriert sein, was bedeutet, dass Sie

- eine *AWS IoT Policy*, die dem Gerät die Berechtigung zum Zugriff auf AWS-IoT-Ressourcen gewährt,
- ein *AWS IoT thing*, das es erlaubt, Geräte in AWS IoT zu verwalten und
- einen privaten Schlüssel und ein Zertifikat, mit denen sich das Gerät bei AWS IoT authentifizieren kann

haben müssen. Ich weiß, das mag kompliziert klingen, aber diese Präliminarien können schnell erledigt werden, wenn Sie sich an den Quick-Connect-Workflow (**Bild 4**) in der FreeRTOS-Konsole halten. Alternativ können Sie jeden der Schritte auch manuell über die Kommandozeile durchführen. Was für die Konfiguration des RED-V-Boards als IoT-Device in AWS noch nötig ist, ist das Herunterladen von FreeRTOS als Betriebssystem für die RISC-V-Architektur.

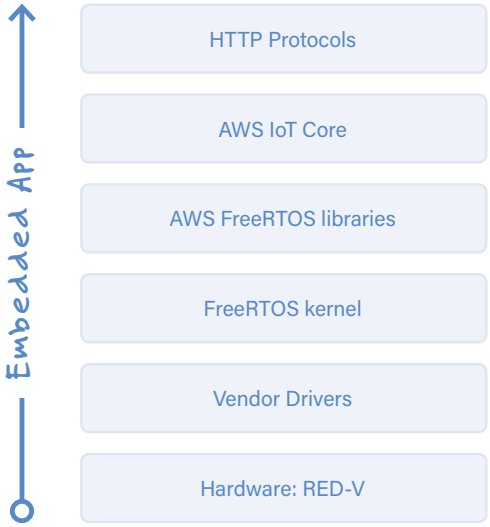


Bild 3. Embedded-App-Layer.

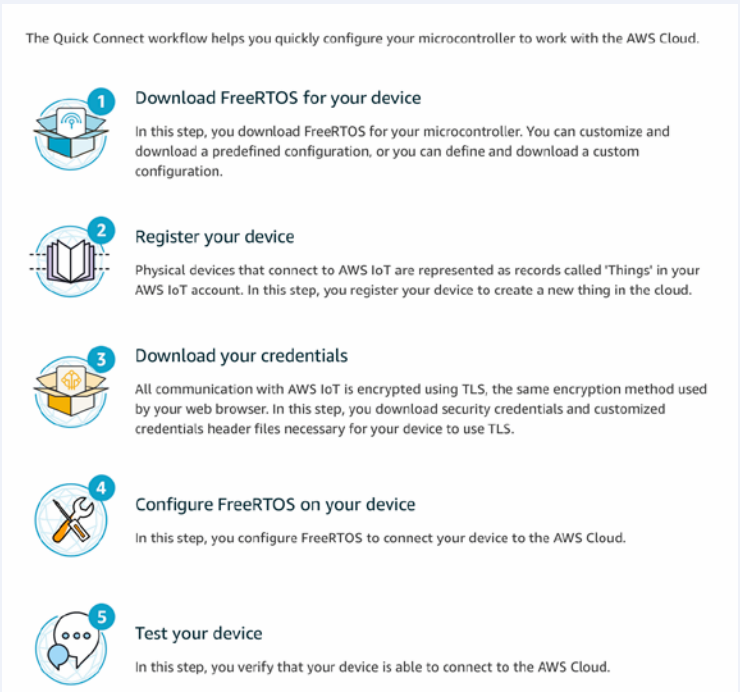


Bild 4. Workflow von Quick-Connect (Quelle: AWS).

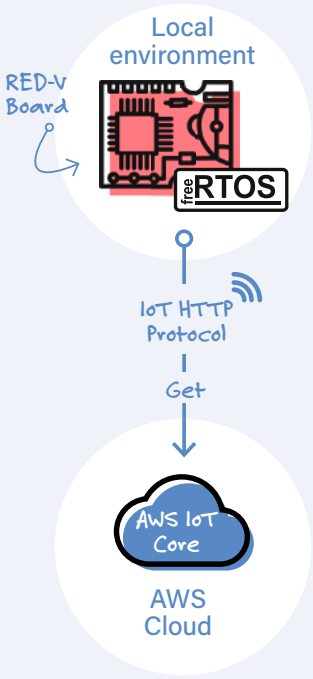


Bild 5. IoT-on-AWS-Anwendung.

Verbinden von RED-V als IoT-Gerät mit dem AWS-IoT-HTTP-Server

Nach der Einrichtung kann man eine HTTP-Demo auf dem RED-V-Board laufen lassen, hierfür genügt ein einzelner Application-Task. HTTP steht für Hypertext Transfer Protocol, und HTTPS ist mit *Transport Layer Security* (TLS) verschlüsselt. HTTP und HTTPS sind Protokolle, die für die Übertragung von Daten über das Web verwendet werden, und sie nutzen bestimmte Request-Methoden, um Aufgaben auszuführen. Zwei allgemein bekannte Methoden sind GET, das Daten von einer bestimmten Ressource anfordert, und POST, das Daten an einen Server sendet, um eine Ressource zu erstellen oder zu aktualisieren. Diese Requests sind die grundlegenden Bausteine hinter jeder IoT-Anwendung. Die Demo zeigt, wie leicht sie mit Open-Source-Hardware und -Software über AWS bereitgestellt werden können.

Wie in **Bild 5** dargestellt, verbindet sich die Demo mit dem AWS-IoT-HTTP-Server, erstellt ein HTTP-Request, sendet ein HTTP-Request, empfängt ein HTTP-Response und trennt dann die Verbindung zum Server. Dank der FreeRTOS-Dokumentation können wir uns die Struktur der Anwendung durch den - recht komplexen - AWS-Code [1] ansehen. Durch die HTTP-Client-Bibliothek von FreeRTOS ist der RED-V nun ein IoT-Gerät, das in der Lage ist, HTTP-Requests zu senden und HTTP-Responses von den meisten HTTP-Servern zu empfangen - nicht nur von AWS-Servern. Das alles funktioniert sowohl synchron als auch asynchron, auch parallel ablaufende Kommunikation ist möglich.

Unendliche Möglichkeiten

Es ist einfach, eine IoT-Anwendung auf der RISC-V-Architektur aufzubauen, gepaart mit dem FreeRTOS-Kernel und den Bibliotheken. Dadurch können Embedded-Geräte für jede Anwendung angepasst und skaliert werden. Das Einrichten von HTTP-Requests über die Client-Bibliothek ist nur die Spitze des Eisbergs im Meer der IoT-Anwendungen. Die Möglichkeiten für diese Open-Source-Technologien sind schier endlos!

200699-03

Elektor Webseite für diesen Artikel:
www.elektormagazine.de/esfe-en-redv

Passende Produkte

Sie suchen die in diesem Artikel erwähnten Artikel? SparkFun und Elektor haben sie!

› **RED-V RedBoard - SiFive RISC-V FE310 SoC**
www.elektormagazine.de/esfe-en-redv1

› **RED-V Thing Plus - SiFive RISC-V FE310 SoC**
www.elektormagazine.de/esfe-en-redv2

WEBLINKS

[1] AWS-Code: https://www.freertos.org/http/http-demo-with-tls-mutual-authentication.html

ELEKTRONIK = SPASS!

Ein Gespräch unter Elektronik-Enthusiasten

Von CJ Abate (USA)

Die globale Elektor-Community besteht aus Elektronikingenieuren, kreativen Makern und Innovatoren, die sich für eine Vielzahl von Themen wie MCU-basierte IoT-Projekte, autonome Fahrzeuge und Embedded-Programmierung interessieren. Wir haben unsere Freunde bei SparkFun gebeten, uns mehr über ihre Produkte, ihre Unternehmenskultur und ihre Community zu erzählen.

Wie lernen Ihre Ingenieure die Produkte und Entwicklungen kennen, aus denen neue SparkFun-Produkte entstehen können? — Mathias Claussen (Deutschland)

Gute Frage! Unsere Ingenieure und Produktentwickler verbringen viel Zeit damit, Technologietrends zu recherchieren und ein Auge darauf zu haben, an welchen Arten von Projekten „Heimelektroniker", Forscher, Ingenieure und Unternehmen arbeiten. Wir arbeiten auch eng mit Zulieferern und anderen Technologieunternehmen zusammen, um deren neue Produkte zu verstehen und um herauszufinden, ob sie für unsere Kunden interessant sein könnten. Außerdem erfahren wir viel von unseren Vertriebspartnern über die Wünsche derer Kunden. Schließlich basteln viele unserer Ingenieure und Produktentwickler an kleinen privaten Projekten und benötigen dafür etwas, was nicht verfügbar ist. Also entwerfen sie eine Lösung für sich selbst und das führt oft zu einem neuen SparkFun-Produkt. Ein aktuelles Beispiel hierfür ist ein Halbleiterrelais zur Steuerung eines selbstgebauten Reflow-Ofens, was zu zwei neuen Produkten führte (SparkFun Qwiic Dual Solid State Relay, www.sparkfun.com/products/16810, und SparkFun Qwiic Quad Solid State Relay Kit, www.sparkfun.com/products/16833).
— Bryan Hoff (Technischer Leiter)

Die Produkte von SparkFun sind überwiegend digital und auf dem neuesten Stand der Technik, komplett mit passenden Tutorials, Software und Online-Tools. Wie steht es mit Entwürfen und Konstruktionen für analoge Elektronik? Gibt es Spezialisten bei SparkFun, an die man sich wenden kann, um neue und alte Erkenntnisse und Publikationen zum Thema „Analog" zu erhalten? — Jan Buiting (Niederlande)

Der Fokus von SparkFun auf digitale, hochmoderne Technik ist ein Selbstläufer, denn die meisten modernen Hightech-Sensoren und ICs sind digital. Viele dieser Komponenten sind an Techniktrends wie Cloud Computing, maschinelles Lernen, mobile Geräte und Wearables ausgerichtet. Den allgemeine Trend „Low Power, Low Cost" unterstützen wir gerne, weil er die Einstiegshürde für Technologien senkt, zu denen wir unseren Kunden Zugang verschaffen wollen. Doch wir sind Elektronikingenieure, wir lieben analog, und die meisten von uns haben mit analoger Elektronik angefangen. Im Moment ist die analoge Elektronik in zwei Bereiche aufgeteilt - klassisches Schaltungsdesign und fortschrittliche Industrie- und Konstruktionslösungen. Die Entwicklung im Bereich der hoch entwickelten (und teuren) Industrie- und Konstruktionsmesstechnik ist interessant, und Trends wie Industrie 4.0 sind sicher ein Markt, den wir weiter beobachten werden. Für analoge Inhalte empfehle ich immer einen Blick auf Hackaday, da dort ständig interessante Artikel und Projekte rund um analoge Themen erscheinen. Auch das IEEE beschäftigt sich viel mit Themen wie fortschrittliche analoge Messtechnik und -geräte.
— Pearce Melcher (Produkt-Service - Technik-Forscher)

GPS-Breakout

Jedes Jahr werden Unmengen von ICs auf den Markt geworfen. Wie entscheiden Sie, welchem IC SparkFun ein Breakout-Board spendiert? — Clemens Valens (Frankreich)

Zunächst einmal sind wir ständig erstaunt über all die coole neue Technologie. Es ist erstaunlich, wenn man sich so etwas wie GNSS-Module anschaut, die noch vor ein paar Jahren nur eine Genauigkeit im Meterbereich hatten und jetzt mit Korrekturdaten eine Genauigkeit im Zentimeterbereich erreichen! Wie entscheiden wir bei so vielen innovativen Produkten, die auf den Markt kommen, welche Technologie oder welches IC wir weiter verfolgen? Es ist eine Mischung aus dem, was wir Ingenieure am raffiniertesten finden, was unsere Partner mit uns teilen, was wir für unsere persönlichen Projekte brauchen und einiges, was durch strategische Planung ausgewählt wird. Letzteres ist oft die bewusste Entscheidung, das Gute, Bessere, Beste einer Technologie anzubieten. Mit einer guten GNSS-Lösung (https://www.sparkfun.com/products/15733) oder einem voll funktionsfähiges GNSS mit Dead Reckoning (https://www.sparkfun.com/products/16344) können unsere Kunden Lösungen finden, die ihren Bedürfnissen und ihrem Budget entspricht. — Bryan Hoff (Technischer Leiter)

Erzählen Sie uns etwas über wichtigsten großen und kleinen Mitbewerber von SparkFun. — Don Akkermans (Niederlande)

Um ehrlich zu sein, wir beschäftigen uns weniger mit der Konkurrenz als mit unserer Community und neuen Technologien. Unsere Ingenieure sind neugierig auf neue und inspirierende Technologien, wie sie genutzt werden können und wie man sie für jeden einfach nutzbar machen kann. Daran halten wir uns. Darüber hinaus konzentrieren wir uns auf Zusammenarbeit - wir haben festgestellt, dass wir selbst mit Unternehmen, die man als Konkurrenz ansehen könnte, die ähnliche Tools, Fachkenntnisse, Produkte oder Kunden haben, gut und vorteilhaft zusammenzuarbeiten können. Obwohl wir konkurrieren, gewinnen wir doch letztendlich alle durch die Zusammenarbeit. Von Anfang an haben wir uns der Open-Source-Technologie verschrieben (und arbeiten eng mit der Open Source Hardware Association zusammen). Das treibt uns zu schneller Innovation und lässt niemals Stagnation zu, und es bedeutet auch, dass wir damit rechnen, dass unsere Produkte innerhalb weniger Wochen nach ihrer Markteinführung kopiert werden (http://bit.ly/TEDx-Seidle). Dies stellt definitiv eine Herausforderung für den Wettbewerb dar. Schließlich achten wir natürlich auf den Markt - es gibt überall Konkurrenz, ob groß oder klein. Ob es sich um Marktplätze von Drittanbietern handelt, um größere Unternehmen, die versuchen, den gleichen Markt wie wir zu erreichen, oder um andere Ingenieurteams, die Boards und Prototyping-Tools entwickeln … wir wissen, dass Kunden viele Optionen haben. Wir tun unser Bestes, um einen Mehrwert zu bieten (Dokumentationen, Tutorials, Videos, Projekte, Beispielcode …), um neue Leute für unsere loyale Community zu gewinnen. — Jahnell Pereira (Chief Business Development Officer)

Wir sehen eine Menge KI- und Machine-Learning-Produkte in Ihrem Shop. Können Sie eines für Einsteiger empfehlen? — Jens Nickel (Deutschland)

DLI-Bausatz für Jetson Nano

NVIDIA legt großen Wert auf die Entwicklung von Einstiegspunkten und Tools für diejenigen, die gerade erst mit dem Thema Maschinelles Lernen beginnen. Um dies zu unterstützen, haben sie einen kostenlosen Online-Kurs des Deep Learning Institute (DLI) zum Selbststudium entwickelt und mit SparkFun zusammengearbeitet, um ein Bündel von Kursmaterialien für Einsteiger zu schnüren - das SparkFun DLI Kit für Jetson Nano: www.elektormagazine.de/esfe-en-jetson1.. — Derek Runberg (Strategische Partnerschaften und Bildungswesen)

Welche SparkFun-Produkte sind derzeit bei den europäischen Ingenieuren und Makern beliebt? Und haben Sie eine Ahnung, warum sie in Europa so beliebt sind? Muhammed Söküt (Deutschland)

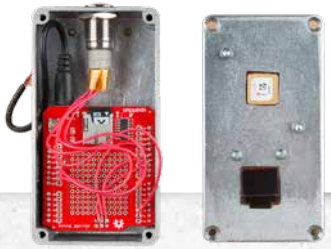
Im Jahr 2020 betraf etwa 30 % unseres Geschäfts andere Länder als die Vereinigten Staaten. Der Prozentsatz war leicht rückläufig, wir vermuten aufgrund von COVID-19 und dem Zollstreit. Für Europa besonders interessant waren einige unserer beliebten Produktfamilien GPS/GNSS (insbesondere Entwicklungsboards mit Modulen von u-blox aus Zürich), RFID, IR, LiDAR und Daten-Tools wie OpenLog. Diese Produkttypen sind nicht nur für viele Hobbyelektroniker interessant, sondern auch Maker und Forscher brauchen diese Tools. Und natürlich schaffen es auch die Produkte und Kits, die mit dem Raspberry Pi zusammenhängen, an die Spitze der Verkaufsliste in Europa. — Jahnell Pereira (Chief Business Development Officer)

Haben Sie in der Zeit vor Corona an vielen Veranstaltungen der Elektronikindustrie in den USA teilgenommen? Wollen Sie in den kommenden Monaten an weiteren Veranstaltungen teilnehmen? — Margriet Debeij (Niederlande)

Man wird sehen, wie sich die Pandemie langfristig auf Veranstaltungen auswirkt. Vor 2020 lag unser Hauptaugenmerk bei Veranstaltungen auf der Unterstützung unserer Partner, indem wir Demonstrationen aufbauten oder Stände betreuten, Präsentationen/Workshops abhielten oder einfach als Besucher teilnahmen, um zu lernen und Partner zu treffen. Wir waren unter anderem bei dem jährlichen RISC-V-Summit, CES, Mobile World Congress, HackaDay's SuperCon, dem ARM-DevSummit, DEF CON, Open Source Hardware Summit, Googles TensorFlow Konferenz und Bildungskonferenzen. Von Ende des vergangenen Jahres und bis heute haben wir einige Konferenzen digital besucht. Wir werden sehen, wie sich das im Laufe des Jahres weiterentwickelt. — Hailey Blessing (Public Relations Spezialist)

Was sind die besten drei Projekte, die mit SparkFun-Produkten durchgeführt wurden und auf die Sie besonders stolz sind?
- Udo Bormann (Deutschland)

Für den Retronics-Artikel habe ich mehrere Leute nach ihren Lieblingsprojekten aus der Vergangenheit von SparkFun gefragt. Natürlich kamen das Port-O-Rotary-Telefon, der Tresor-Knack-Roboter und der Skimmer-Scanner zur Sprache - aber darüber können Sie an anderer Stelle in diesem Elektor-Heft lesen. Ich persönlich bin sehr stolz auf die Arbeit, die wir im Bereich GPS geleistet haben - als Rob Reynolds den RTK-Surveyor benutzte, um mit SW Maps die SparkFun-Flamme auf den Parkplatz zu zeichnen, hat mich das umgehauen. Ich finde auch das Tutorial „GPS Geo-Mapping at the Push of a Button" von Brandon Williams toll, das das RedBoard Turbo SAMD21 Development Board, das SparkFun microSD Shield, die SparkFun GPS Breakout Chip Antenna SAM-M8Q und das Mirco OLED Breakout miteinander verband, um Koordinaten auf der ganzen Welt in Google Earth einzupflegen.
— Jahnell Pereira (Vorstand für Geschäftsentwicklung)



Eine GPS-Geo-Mapping-Lösung

Ich habe 2013 an der SparkFun Autonomous Vehicle Competition (AVC) in Boulder, Colorado, teilgenommen. Es war eine tolle Veranstaltung mit einer Reihe von erstaunlichen Drohnen. Haben Sie immer noch Kunden, die sich für Drohnen und autonome Fahrzeuge interessieren? Planen Sie irgendwelche neuen Wettbewerbe, wenn COVID-19 hinter uns liegt? — C. J. Abate (USA)

Wir haben viele Kunden, die autonome Drohnen und Fahrzeuge sowohl für private als auch für berufliche Zwecke bauen. Die AVC brachte Menschen aus der ganzen Welt und jeden Alters bei einem Wettbewerb zusammen. Als es aber immer mehr solcher Wettbewerbe gab, haben wir die Veranstaltung nach einem Jahrzehnt eingestellt. Wir haben viel über weitere Wettbewerbs nachgedacht, aber noch nicht die richtige Inspiration gefunden. Wir würden gerne wieder einen Wettbewerb veranstalten, aber es muss etwas Einzigartiges sein, das eine neue Herausforderung für unsere Community darstellt. Wenn Ihnen oder Ihren Lesern etwas einfällt, lassen Sie es uns wissen.. **— Hailey Blessing (Referentin für Öffentlichkeitsarbeit)**

Fahrzeug von einem früheren AVC

Vor ein paar Jahren schrieb SparkFun in einem Blog über seine „nicht so traditionelle Arbeitskultur", zum Beispiel Hunde im Büro! Wie haben Sie vor COVID-19 Hunde in Ihren Arbeitsbereich integriert? — Denise Bodrone (Niederlande)

SparkFun legt großen Wert auf eine in vielerlei Hinsicht außergewöhnliche Arbeitskultur. Dazu gehört auch, dass jedem Mitarbeiter erlaubt ist, einen Hund mit zur Arbeit zu bringen, wenn er sich an unsere Hunde-Richtlinie hält. Diese Richtlinie enthält Regeln, an deren Erstellung, Aktualisierung und Einhaltung alle Hundebesitzer mitgewirkt haben, und legt allen Hundebesitzern die Verantwortung auf, dafür zu sorgen, dass wir gut mit unseren befellten Freunden umgehen. Die Grundbedingung, um Hunde auf dem Gelände zu haben, ist, dass alle Hundebesitzer an einem „Hundetribunal" teilzunehmen. Das ist eine Gruppe von fünf zufällig ausgewählten Hundebesitzern, die hundebezogene Dinge und Beschwerden überprüfen und Lob und Tadel erteilen. Sie können einen Hundebesitzer möglicherweise zur Verantwortung ziehen, basierend auf unseren etablierten Regeln und einer vorhergehenden Diskussion, und mit den betreffenden Hundebesitzern über die nächsten Schritte sprechen. Es ist eine interessante Lösung, die im Laufe der Jahre immer wieder überarbeitet wurde. Sie ist bei weitem nicht perfekt, aber funktioniert gut und ermöglicht, dass täglich bis zu 30 Hunde SparkFun bevölkern! **— Kristen Moorefield (Betriebsleiterin)**

Wir alle sollen versuchen, den Planeten nicht umzubringen. Wie steht SparkFun zu einer nachhaltigen und umweltfreundlichen Produktentwicklung? Vermeiden Sie Plastik, wo es möglich ist? — Mathias Claussen (Deutschland)

Wir denken ständig über unseren Einfluss auf die Umwelt nach und streben nach „Null Abfall". Wir haben unseren Hauptsitz 2012 gebaut - es war das erste Gebäude im Kreis Boulder, das nach dem International Green Construction Code gebaut wurde. Unser Nachhaltigkeitsteam trifft sich einmal pro Woche, um unseren schwer recycelbaren Abfall zu sortieren und so viel Material wie möglich vor der Deponie zu bewahren. Wir verwenden Smart-Energy-Steckdosen, nehmen am E-Müll-Recycling und an kommunalen Fahrradprogrammen teil, und unsere Photovoltaik-Anlage auf dem Dach liefert rund 30 % unseres Strombedarfs. In allen Abteilungen tun wir unser Bestes, um unseren ökologischen Fußabdruck zu verkleinern. **— Nick Beni (Facilities Manager)**

Nick mit der Auszeichnung zum Recycler des Jahres

Wie entscheiden Sie, ob ein Produkt für den SparkFun-Shop geeignet ist? Was ist Ihr bisher meistverkauftes Produkt? – Luc Lemmens (Niederlande)

Wenn man die letzten 18 Jahre betrachtet, ist das meistverkaufte Produkt das SparkFun Inventors Kit oder Derivate davon. Was für den SparkFun-Shop geeignet ist? Ja, es ist eine unserer Stärken, dass wir bereit sind zu experimentieren, unseren Interessen und Inspirationen zu folgen und auszuwählen, was wir für nützlich halten. **— Jordan Colby (Datenanalyse-Manager)**

200688-01

Hand-verlesene Produkte

sparkfun ELECTRONICS

Seit 2003 hilft SparkFun dabei, Ihre Ideen in die Realität umzusetzen - egal, ob Sie eine intelligente Wetterstation entwickeln, die Grenzen des maschinellen Lernens erforschen, einen Roboter für die Schule bauen oder Ihren ersten (oder zehnten) Prototyp für ein Produkt entwickeln wollen. Unabhängig davon, welche Ziele Sie haben oder wie Ihr Kenntnisstand ist, die Open-Source-Komponenten, Ressourcen und Online-Tutorials sind darauf ausgelegt, den Zugang zu innovativer Technologie zu erweitern und den Weg zum fertigen Projekt zu verkürzen. Hier einige handverlesene Produkte für die Elektor-Community:

SparkFun Inventor's Kit - V4.1

Das SparkFun Inventor's Kit (SIK) ist ein toller Begleiter auf den ersten Schritten in die Welt der Programmierung in der Arduino-Sprache und der Hardware-Interaktion. Das SIK enthält alles, was Sie brauchen, um fünf umfassende Projekte zu realisieren, die aus 16 miteinander verbundenen Schaltungen bestehen und alles vom Blinken einer LED bis zum Auslesen von Sensoren lehren. Das abschließende Projekt ist Ihr ganz eigener autonomer Roboter! Für dieses Kit sind keine vorherigen Programmier- oder Elektronikkenntnisse erforderlich.

www.sparkfun.com/products/15267

SparkFun MicroMod DIY Carrier Kit (5er-Pack)

www.sparkfun.com/products/16549

SparkFun Qwiic pHAT v2.0 for Raspberry Pi

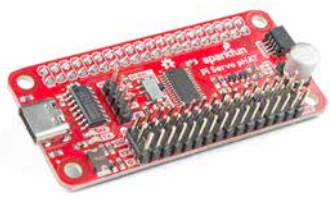
www.sparkfun.com/products/15945

SparkFun Sensor Kit

Entwickeln Sie gerne mit verschiedenartigen Sensoren oder benötigen Sie eine Vielzahl davon für ein Projekt? Das SparkFun-Sensor-Kit enthält Sensoren für so ziemlich jede Aufgabe. Mit dem Inhalt des Kits werden Sie in der Lage sein, Gesten, Feuchtigkeit, Temperatur, Bewegung, Berührung, Ton, Höhe, Beschleunigung und mehr zu erfassen!

www.sparkfun.com/products/16156

SparkFun Servo pHAT for Raspberry Pi



www.sparkfun.com/products/15316

SparkFun RED-V RedBoard - SiFive RISC-V FE310 SoC

Das SparkFun RED-V-RedBoard (V steht für „five") ist ein kostengünstiges Entwicklungsboard mit dem Freedom-SoC E310 mit RISC-V-Befehlssatzarchitektur (ISA). Was das RED-V-RedBoard von anderen unterscheidet, ist der komplett quelloffene Ansatz von der Hardware bis zur ISA.

www.sparkfun.com/products/15594

SparkFun RED-V Thing Plus - SiFive RISC-V FE310 SoC

www.sparkfun.com/products/15799

SparkFun Qwiic Cable Kit

Das SparkFun-Qwiic-Connect-System erfreut sich bei Makern einer immer größeren Beliebtheit. Um den Einstieg noch einfacher zu machen, haben wir diesen Qwiic-Kabelsatz für Sie zusammengestellt.
www.sparkfun.com/products/15081

SparkFun Qwiic GPIO

Was ist, wenn Sie nur einen einzigen I/O-Pin mehr benötigen? Oder wünschen Sie sich gleich ein paar zusätzliche GPIO-Pins? Sie könnten einen anderen, größeren Mikrocontroller zu Ihrem Projekt hinzufügen, aber das würde die Kosten und die Komplexität erhöhen. Stattdessen können Sie mit dem Qwiic-GPIO einfach acht neue Pins zu Ihrem Qwiic-Projekt hinzufügen!
www.sparkfun.com/products/17047

SparkFun Qwiic Adapter

Der SparkFun Qwiic Adapter bietet die perfekte Möglichkeit, jedes alte I²C-Board in ein Qwiic-fähiges Board zu verwandeln.
www.sparkfun.com/products/14495

SparkFun GPS Breakout - Chip Antenna, SAM-M8Q (Qwiic)

Das SparkFun GPS-Breakout SAM-M8Q ist ein hochwertiges GPS-fähiges Board mit beeindruckenden Konfigurationsmöglichkeiten.

www.sparkfun.com/products/15210

SparkFun Auto pHAT for Raspberry Pi

www.sparkfun.com/products/16328

ESP32 WROOM MCU Module - 16MB (Chip-Antenne)

www.sparkfun.com/products/16281

SparkFun Power Delivery Board - USB-C (Qwiic)

Es ist Zeit für eine effektive Stromversorgung in Ihrem Projekt – das SparkFun-Power-Delivery-Board macht es möglich! Herkömmliche Netzteile gibt es zwar für viele Strombereiche, aber die Spannung bleibt fest bei 5 V.

www.sparkfun.com/products/15801

SparkFun Top pHAT for Raspberry Pi

www.sparkfun.com/products/16653

SparkFun Raspberry Pi 4 Basic Kit - 4GB

www.sparkfun.com/products/16384

Qwiic Button - Red LED

Taster sind ein einfacher Weg, um mit einem Projekt zu interagieren. Mit dem Qwiic-Connect-System ist die Verwendung eines Tasters so einfach wie das Anschließen eines Kabels oder das Laden von vorprogrammiertem Code!
www.sparkfun.com/products/15932

Qwiic Cable - 100mm

Dies ist ein 100 mm langes 4-adriges Kabel mit 1-mm-JST-Anschluss. Es wurde entwickelt, um Qwiic-fähige Komponenten miteinander zu verbinden, kann aber auch für andere Anwendungen verwendet werden.
www.sparkfun.com/products/14427

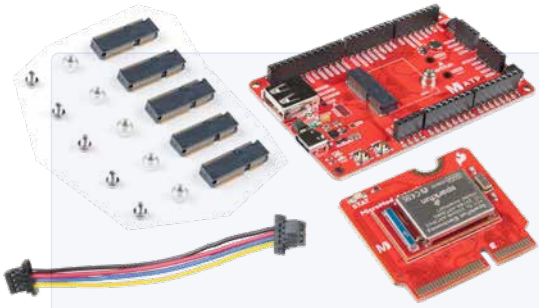
Qwiic Cable - 50mm

Dies ist ein 50 mm langes 4-adriges Kabel mit 1-mm-JST-Anschluss. Es wurde entwickelt, um Qwiic-fähige Komponenten miteinander zu verbinden, kann aber auch für andere Anwendungen verwendet werden.
www.sparkfun.com/products/14426

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 wir verlosen fünf MicroMod-Bundles von SparkFun!

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. Unter allen richtigen Lösungen verlosen wir **fünf SparkFun-MicroMod-Bundles,** mit einer ATP-Trägerplatine, einem Qwiic-Kabel 50 mm, einem Qwiic-Kabel 100 mm und einem SparkFun MicroMod DIY Carrier Kit!



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 10. April 2021.

Die Gewinner des Hexadokus aus der Ausgabe Januar/Februar 2021 stehen fest!

Die richtige Lösung ist: **A314C**

Einen Elektor-Wertgutschein über je 50 € haben gewonnen:

Sigurd Künzel, Antonio Conati, Fred Gross, Petros Tsirvoulis und Johan J. van Dorp.

Herzlichen Glückwunsch!

		C			8	0		7	E				B	5	
B		1				2				4	6				F
2						6	7					A	D		0
	5					C	D			0	1	E	2		
D	9	F			E	3				5	8			6	C
		E	5	6				9			A	F			
8				C				B	D	3					
C				8	D			2		E	F			9	
	C			7	3		A			6	D				E
					0	F	8				3				5
			1	2			6				5	8	A		
A	7			D	5				8	1			6	C	4
		9	8	A	4			1	B					3	
7		A	0					D	C						1
F				0	1				7				8		D
	3	B				D	2		5	8			0		

F	1	6	B	4	5	7	C	9	0	8	D	2	3	A	E
E	9	7	2	F	0	A	B	C	1	6	3	5	4	D	8
3	C	0	4	1	6	8	D	A	E	2	5	F	9	7	B
5	D	8	A	E	9	2	3	7	4	B	F	6	0	C	1
A	E	9	6	D	F	0	8	2	5	1	B	4	C	3	7
7	B	D	0	9	A	3	1	4	C	E	6	8	5	F	2
C	2	F	1	6	4	5	E	8	D	3	7	9	A	B	0
8	3	4	5	2	B	C	7	F	9	A	0	D	1	E	6
4	A	B	C	5	1	9	F	3	6	7	8	E	2	0	D
D	5	1	9	7	8	4	0	E	F	C	2	A	B	6	3
0	F	2	E	3	C	B	6	5	A	D	1	7	8	9	4
6	8	3	7	A	D	E	2	0	B	4	9	1	F	5	C
1	4	5	D	0	E	6	9	B	2	F	C	3	7	8	A
9	0	C	3	8	2	1	A	D	7	5	E	B	6	4	F
2	7	A	F	B	3	D	5	6	8	0	4	C	E	1	9
B	6	E	8	C	7	F	4	1	3	9	A	0	D	2	5

Der Rechtsweg ist ausgeschlossen. Mitarbeiter der in der Unternehmensgruppe Elektor International Media B.V. zusammengeschlossenen Verlage und deren Angehörige sind von der Teilnahme ausgeschlossen.

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

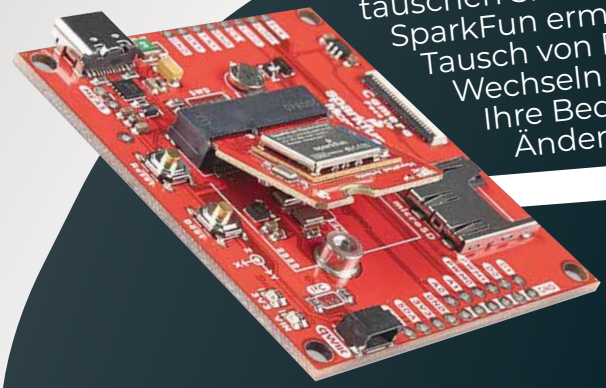
elektor design > share > sell

RAPID PROTOTYPING LEICHT GEMACHT

Vom Bauteilstapel zum fertigen Produkt, SparkFun hilft Ihnen auf Ihrem Weg zum Rapid Prototyping.

QWIIC MACHT DAS PROTOTYPING SCHNELLER UND EINFACHER
DAS SYSTEM NUTZT 4-POLIGE JST-STECKVERBINDER ZUR VERBINDUNG
VON ENTWICKLUNGSBOARDS ZU SENSOREN, LCDS, RELAIS UND MEHR.

Fügen Sie Ihrem Projekt einfach neue Komponenten hinzu oder tauschen Sie diese aus! Die **MICROMOD**-Produktreihe von SparkFun ermöglicht einfache Projekt-Upgrades und den Tausch von Funktionen mitten in der Projektentwicklung. Wechseln Sie Träger- oder Prozessorplatinen, wie es Ihre Bedürfnisse erfordern, ohne dass Änderungen am Code notwendig wären.



SparkFun **À LA CARTE** (ALC) hilft Ihnen bei der Erstellung kundenspezifischer Boards und überbrückt die Lücke zwischen Prototyp und Produktion! ALC erstellt Ihre Platine mit den Komponenten, die Sie auswählen, und macht es Ihnen einfach, genau das zu entwickeln, was Sie wollen.

FÜR WEITERE INFORMATIONEN BESUCHEN SIE

www.sparkfun.com/qwiic

www.sparkfun.com/micromod

www.alc.sparkfun.com

sparkfun®
START SOMETHING