Arduino – dein Einstieg

Die Open-Source Plattform für Elektronik-Prototypen

1 Einleitung

Arduino ist eine Open-Source-*Physical-Computing*-Plattform zur Verwirklichung interaktiver Projekte, die unabhängig oder mit dem Internet verbunden sind. Arduino war ursprünglich für Künstler, Designer usw. konzipiert, die Physical Computing in ihre Entwürfe integrieren wollten, ohne dafür erst Elektroingenieure werden zu müssen. Später wurde daraus die bevorzugte Plattform für buchstäblich Millionen von Menschen, die mithilfe digitaler Technik Innovationen vorantreiben wollten.

Die Arduino-Hardware und -Software sind quelloffen, d.h. Open Source. Die Open-Source-Philosophie fördert eine Community, die ihr Wissen großzügig teilt. Für Anfänger ist das toll, da Hilfe häufig schon in geografischer Nähe und online immer zur Verfügung steht, auf vielen verschiedenen Kompetenzniveaus und zu einer verblüffenden Bandbreite an Themen. Beispielprojekte werden nicht nur mit den Bildern des fertigen Projekts präsentiert, sondern umfassen Anleitungen für den Eigenbau oder als Startpunkt für die Integration in deine Weiterentwicklung oder verwandte Projekte.

Die auf den Namen »Integrated Development Environment« (Integrierte Entwicklungsumgebung; IDE) hörende Arduino-Software ist kostenlos. Du kannst sie auf *www.arduino.cc* herunterladen. Die Arduino-IDE basiert auf der Programmiersprache Processing, die entwickelt wurde, um Künstlern bei der Kreation von Computerkunst zu helfen, ohne dass sie erst Softwareentwickler werden müssen (*http://www.processing.org*). Die Arduino-IDE kann auf Windows, macOS und Linux laufen.

Das Arduino-Uno-Board ist kostengünstig (rund 23 €) und gewöhnlichen Anfängerfehlern gegenüber recht tolerant. Wenn du es irgendwie schaffst, die Hauptkomponente des Arduino Uno zu beschädigen, lässt sie sich für gerade mal 4 € austauschen.

Das Arduino-Projekt wurde in einer Bildungsumgebung entwickelt und ist ein sehr beliebtes Lehrmittel. Dieselbe Open-Source-Philosophie, die die Community schuf, die großzügig Informationen, Antworten und Projekte teilt, führt dazu, dass auch Lehrmethoden, Lehrpläne und andere Informationen geteilt werden.

Da die Arduino-Hard- und -Software quelloffen sind, kannst du das Arduino-Hardwaredesign herunterladen und deinen eigenen bauen oder ihn als Startpunkt für dein eigenes Projekt verwenden, das vom Design her auf Arduino basiert (oder diesen integriert), oder du benutzt es nur, um zu verstehen, wie Arduino funktioniert. Mit der Software kannst du ebenso vorgehen.

Arduino ist für den einfachen Gebrauch vorgesehen, und dieses Buch soll Anfängern ohne Vorerfahrung helfen, direkt mit Arduino loszulegen.

Zielpublikum

Dieses Buch wurde für Anfänger geschrieben – Leute, die lernen wollen, etwas mit Elektronik und Programmieren zu erschaffen, ohne einen technischen Hintergrund zu haben. Deshalb versucht es, Dinge auf eine Weise zu erklären, die manchen Ingenieur sicher in den Wahnsinn treiben würde. Einer von ihnen bezeichnete die einführenden Kapitel der ersten Ausgabe übrigens als »alberne Flausen«. Das ist genau der Punkt. Seien wir ehrlich: Etwas gut zu können und etwas gut erklären zu können, sind zwei Paar Stiefel. Wenn es Flausen sind, die Millionen von Menschen etwas verständlich machen und sie dadurch befähigen, dann haben wir hoffentlich massenhaft Flausen im Kopf.

Dieses Buch ist nicht als Lehrbuch über Elektronik oder Programmieren gedacht, dennoch wirst du beim Lesen etwas zu Elektronik und über Programmieren lernen.

> *Als Arduino allmählich populär wurde, fiel mir auf, wie alle möglichen Experimentatoren, Bastler und Hacker begannen, ihn zum Schaffen schöner und verrückter Objekte einzusetzen. Mir wurde bewusst, dass ihr alle auf eure eigene Art und Weise Künstler und Designer seid, und daher ist dieses Buch auch für euch.*
>
> *– Massimo*

Arduino baut auf der Dissertation von Hernando Barragán über die Wiring-Plattform auf, die er während des Studiums bei Casey Reas und Massimo am Interaction Design Institute Ivrea (IDII) schrieb.

Was ist Interaktionsdesign?

Arduino wurde aus der Taufe gehoben, um Interaktionsdesign zu lehren, eine Design-Disziplin, die das Prototyping in den Mittelpunkt ihrer Methodologie stellt. Es gibt viele Definitionen für Interaktionsdesign, aber eine, die wir bevorzugen, lautet:

Interaktionsdesign ist das Design jedes beliebigen interaktiven Erlebnisses.

Heutzutage beschäftigt sich Interaktionsdesign mit der Gestaltung bedeutungsvoller Erlebnisse zwischen uns (Menschen) und Objekten. Es ist eine gute Möglichkeit, um die Entwicklung schöner – und vielleicht sogar kontroverser – Erfahrungen zwischen uns und Technik zu erforschen. Interaktionsdesign fördert Design durch einen schrittweisen Prozess, basierend auf Prototypen zunehmender Realitätsnähe. Diese Vorgehensweise – auch Teil mancher Arten konventionellen Designs – lässt sich mit dem Einschluss des Prototypings in Technologie erweitern, insbesondere Prototyping mit Elektronik.

Das spezifische Feld des mit Arduino verbundenen Interaktionsdesigns wird häufig *Physical Computing* (oder *Physikalisches Interaktionsdesign*) genannt.

Was ist Physical Computing?

Physical Computing nutzt Elektronik zum Aufbau neuer und innovativer Geräte. Es schließt das Design interaktiver Objekte ein, die mittels Sensoren und Aktoren, deren Verhalten über eine in einem *Mikrocontroller* (ein kleiner Computer auf einem einzelnen Chip) laufende implementierte Software gesteuert wird, mit Menschen kommunizieren können.

In der Vergangenheit bedeutete der Einsatz von Elektronik, dass man sich ständig mit Ingenieuren herumschlagen musste und Schaltungen nur mit einer kleinen Komponente nach der anderen baute; diese Probleme hielten kreative Leute davon ab, direkt mit dem Medium herumzuspielen. Die meisten Tools waren für Ingenieure gedacht und erforderten umfassende Kenntnisse.

In den vergangenen Jahren sind Mikrocontroller billiger und einfacher in der Anwendung geworden. Gleichzeitig sind Computer schneller und leistungsstärker geworden, was das Herstellen besserer (und einfacherer) Entwicklungstools ermöglichte.

Mit Arduino haben wir es geschafft, Neulingen diese Tools ein Stück näherzubringen, da die Leute bereits nach nur einem oder zwei Workshop-Tagen oder mithilfe dieses Buchs damit beginnen können, etwas zusammenzubauen. Mit Arduino kann ein Anfänger die Grundlagen der Elektronik und Sensoren sehr schnell kennenlernen und kann ohne großen Aufwand mit dem Bau von Prototypen beginnen.

8 Automatisches Gartenbewässerungssystem

In Kapitel 6 hast du das, was du über Arduino gelernt hast, mit einem Projekt, der Arduino-Netzwerk-Lampe, kombiniert. Teil des Vergnügens war, einige der einfachen Übungen mit einem praktischen Projekt zu verknüpfen. Du hast auch die Programmiersprache Processing kennengelernt und wie man sie zur Einrichtung eines Proxys auf deinem Computer verwenden kann, um Dinge zu tun, die mit deinem Arduino schwierig oder gar unmöglich wären.

In diesem Kapitel wirst du erneut einfache Beispiele mit einigen neuen Ideen verknüpfen, um ein praktisches Projekt durchzuführen. Unterwegs lernst du mehr zu Elektronik, Kommunikation und Programmieren, und wir werden einen Blick auf Konstruktionstechniken werfen.

Das Ziel dieses Projekts ist, automatisch jeden Tag das Wasser zur richtigen Zeit ein- und auszuschalten, außer wenn es regnet.

Auch wenn du keinen Garten besitzt, kannst du dennoch mit diesem Projekt Spaß haben. Wenn du nur eine kleine Hauspflanze hast, die du gießen möchtest, versuche dies mit nur einem Ventil zu bauen. Wenn du jeden Tag um 17 Uhr ein leckeres Getränk deiner Wahl zapfen möchtest, ziehe den Einsatz einer für Lebensmittel geeigneten Pumpe statt des Wasserventils in Erwägung. Adafruit verkauft zum Beispiel eine peristaltische Flüssigkeitspumpe mit Silikonschlauch (*https://www.adafruit.com/product/1150*).

Als Professor lehre ich viele Studenten, etwas zu bauen. Mit der Zeit wurde mir bewusst, dass Studenten manchmal denken, ich würde von Anfang an genau wissen, wie ein Projekt zu bauen sei. Tatsächlich ist die Entwicklung eines Projekts ein Vorgang, der extrem iterativ, also nur Schritt für Schritt vorangeht.

– Michael

Um ein Projekt neu anzulegen, beginne mit einer Idee und skizziere kleine Teile davon; dabei erfordert dies manchmal Änderungen an der ursprünglichen Idee. Wir müssen häufig einen Umweg machen, um zu lernen, wie ein neues Elektronikteil funktioniert, oder um ein Programmierkonzept zu verstehen, dem wir zuvor nie begegnet sind, oder um uns selbst daran zu erinnern, wie ein Merkmal des Arduino zu nutzen ist, das wir lange nicht genutzt haben oder das uns neu ist. Manchmal müssen wir in Fachbüchern nachschlagen, im Internet schauen oder jemanden um Rat bitten. Wir prüfen viele Beispiele, Anleitungen und Projekte, die etwas von dem enthalten, womit wir es zu tun haben. Wir entnehmen Teile von hier und da und kombinieren sie – vielleicht zunächst nur sehr grob, wie bei Frankensteins Monster, um zu sehen, wie Dinge zusammen funktionieren.

Während das Projekt vom Konzept über grobes Design bis zum Testen von Teilen der Hardware und Software immer weiter voranschreitet, müssen wir immer wieder zurückschauen und Änderungen an etwas vornehmen, das wir vorher getan haben, damit alles richtig zusammenarbeitet. Wir kennen keinen einzigen Ingenieur, der mit einem leeren Blatt Papier anfängt, ein ganzes Projekt von Anfang bis Ende entwirft, das dann genau nach Plan funktioniert, ohne jemals zurückzugehen und etwas ändern zu müssen.

Alles zuvor Genannte gilt sowohl für Hardware als auch für Software.

Was wir damit ausdrücken wollen: Auch du als Anfänger bist bereit, Projekte zu entwerfen. Beginne mit dem, was du weißt, und füge langsam Funktionen hinzu, eine neue Idee oder ein neues Teil nach dem anderen. Hab keine Angst, spannenden Ideen nachzugehen, die keinen unmittelbaren Nutzen haben.

> *Wann immer ich von einem Elektronikteil, einem Programmierkonzept oder einem interessant klingenden Trick höre, probiere ich es aus, auch wenn ich dafür keinen sofortigen Nutzen habe. Dieses Wissen wird dann ein weiteres Werkzeug in meinem Werkzeugkasten. Wenn du nicht mehr vorankommst oder etwas nicht weißt, denk dran, dass selbst professionelle Ingenieure ständig Neues lernen müssen.*
>
> *– Michael*

Dank der breiten und großzügigen Arduino-Community hast du über das Internet viele Ressourcen zur Verfügung, und solange du kein Eremit auf einer Bergspitze bist, kannst du wahrscheinlich in deiner Gegend ein Arduino Meetup, einen Club, Makerspace, Hackerspace oder sogar einen Menschen finden, der dir helfen kann.

Zu einigen Hinweisen, wie du das Beste aus Online-Ressourcen machst, schau unter »Online Hilfe bekommen« auf Seite 226 nach.

Ergänzend zum weiteren Lehren zu Elektronik, Programmieren und Konstruktion werde ich dir einen Einblick in den Entwurfsvorgang bieten. Du wirst sehen, dass manche der einfachsten Schaltungen oder Sketche immer wieder modifiziert werden, bis wir endlich beim angezielten Projekt ankommen. Dennoch habe ich einige Schritte ausgelassen, um zu vermeiden, dass dieses Kapitel ein ganzes Buch für sich wird. Iterationen brauchen Zeit!

Planung

Beginne wie in Kapitel 6 darüber nachzudenken, was du erreichen willst und welche Teile du benötigst.

Dieses Projekt verwendet gewöhnliche elektrische Wasserventile für den Garten, die es in jedem Baumarkt gibt. Wenn du schon im Laden bist, kannst du noch eine Stromquelle oder einen Transformator besorgen, der sich für diese Ventile eignet. In Kapitel 5 hast du den Einsatz eines MOSFET zum Steuern eines Motors erlernt. Das könnte auch für die Wasserventile funktionieren, allerdings könnten manche Wasserventile *Wechselstrom* (AC) benötigen, und MOSFETs können nur *Gleichstrom* (DC) steuern. Um Wechselstrom steuern zu können, benötigst du ein Relais, das sowohl Wechsel- als auch Gleichstrom steuern kann.

In »Antrieb größerer Lasten (Motoren, Lampen und dergleichen)« auf Seite 78 in Kapitel 5 hast du gelernt, dass ein MOSFET eine Art Transistor ist, in dem der *Gate*-Pin steuern kann, ob Strom zwischen den *Drain*- und *Source*-Pins fließt. In diesem Sinne ist ein MOSFET ein Schalter. Ein Relais ist auch ein Schalter. Im Relais befindet sich ein winziger mechanischer Schalter, der von einem Elektromagneten gesteuert wird: Durch Ein- und Ausschalten des Elektromagneten kannst du steuern, ob Strom durch den mechanischen Schalter fließt.

Um zu wissen, wann das Wasser ein- und ausgeschaltet werden soll, benötigen wir eine Art Uhr. Du könntest das in unserem Programm mit der im Arduino integrierten Zeitschaltuhr versuchen, aber das wäre kompliziert. Und noch schlimmer: Sie ist nicht besonders genau. Tatsächlich existiert ein solches Gerät, ist recht kostengünstig und lässt sich leicht mit Arduino einsetzen. Die RTC (Real Time Clock, zu Deutsch Echtzeituhr) ähnelt dem Gerät in deinem Computer, das Datum und Zeit im Auge behält, wenn du ihn längere Zeit ausgeschaltet lässt.

Wir benötigen außerdem einen Sensor, der uns sagt, ob es regnet. Wir verwenden einen Temperatur- und Feuchtigkeitssensor, da diese preisgünstig und leicht einsetzbar sind. Wir müssen die Temperatur nicht unbedingt wissen, aber du erhältst die Zusatzfunktion »frei Haus«, und sie könnte sich als nützlich erweisen.

Abschließend benötigen wir eine Möglichkeit, die Ein- und Ausschaltzeiten einzustellen, d.h. die *Benutzerschnittstelle*. Damit dieses Projekt nicht aus dem Ruder läuft, verwenden wir den seriellen Monitor als Benutzerschnittstelle. Wenn du mit der Zeit Arduino besser beherrschst, könntest du diesen durch ein LCD-Display und einen Drucktaster ersetzen.

Bevor du mit dem Programmieren beginnst, musst du darüber nachdenken, wie die Hardware angeschlossen werden soll. Ich benutze gern ein grobes Blockdiagramm, das mir hilft, alle benötigten Teile zu sehen und mir zu überlegen, wie sie verbunden werden sollten. Am Schluss musst du genau wissen, wie die Dinge anzuschließen sind, aber im Blockdiagramm (Abb. 8–1) verwenden wir nur einen Strich, um eine Art von Verbindung zu symbolisieren.

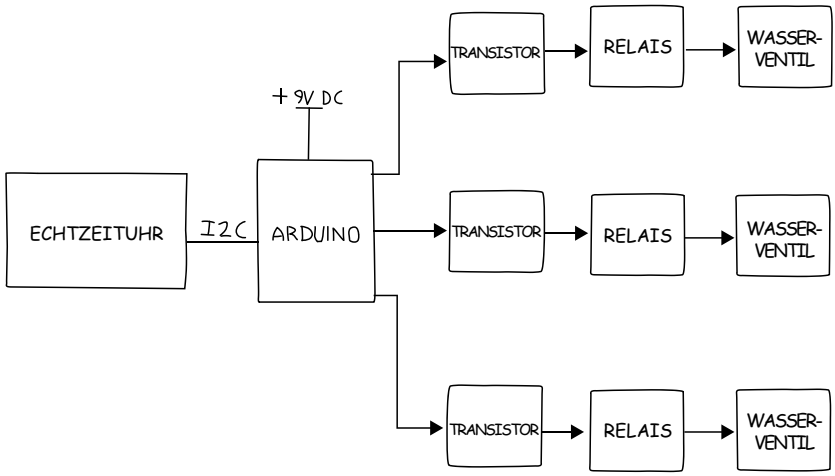

Abb. 8–1 Blockdiagramm mit Echtzeituhr, Arduino, MOSFETs, Relais, Ventilen und Stromzufuhr

In diesem Diagramm gehen wir von drei separaten Wasserventilen aus, aber du siehst, wie davon abgeleitet werden kann, je nachdem, wie dein Bedarf aussieht.

Da dies ein fortgeschritteneres Projekt ist, werde ich Konstruktionstechniken vorstellen. Dieses Projekt soll monate- oder sogar jahrelang zuverlässig arbeiten – du hast hier also ein anderes Ziel als bei einem simplen Beispiel, das dir nur zeigen soll, wie etwas funktioniert. Die lötfreie Steckplatine, die du vorher genutzt hast, eignet sich hervorragend fürs Prototyping oder Experimentieren, aber aus Gründen der Zuverlässigkeit löten wir bei diesem Projekt die Komponenten auf ein *Proto-Shield*. Wir werden uns sowohl Gedanken über die Stromzufuhr machen als auch über alle Verbindungen zu den diversen externen Teilen wie die Wasserventile. Wir schauen uns sogar an, wie dieses Projekt mit einer Art von Gehäuse geschützt werden kann.

Bei *Shields* handelt es sich um Boards, die genau in die Pins des Arduino passen und zusätzliche Funktionen bereitstellen. Das Arduino Proto-Shield ist ein spezielles Shield, das dazu vorgesehen ist, dass du darauf deine eigene Schaltung aufbauen kannst.

Eine weitere Funktion, die für deine an Komplexität zunehmenden Projekte nützlich ist, ist eine Zustandsanzeige. Das ist beim Debugging hilfreich und insbesondere dann, wenn Teile deines Systems weit entfernt sind, wie die Wasserventile. Wir fügen LEDs hinzu, um anzuzeigen, dass die Wasserventile aktiviert sind. Vergiss nicht die Widerstände für die LEDs.

Nun, da wir einige weitere Details haben, lege ich gern eine vorläufige Einkaufsliste an. Bei komplexen Systemen gehe ich davon aus, dass ich Änderungen vornehmen muss: Wenn ich zum Beispiel am Sketch arbeite, stelle ich vielleicht fest, dass ich ein weiteres Teil benötige. (Die abschließende, vollständige Einkaufsliste findet sich als »Einkaufsliste für das Bewässerungsprojekt« auf Seite 188.)

Keine Sorge, wenn du nicht alle diese Teile kennst. Wir gehen im Weiteren detailliert auf sie ein:

- eine Echtzeituhr (RTC)
- ein Temperatur- und Feuchtigkeitssensor
- ein Proto-Shield
- drei elektrische Wasserventile
- ein Transformator oder Netzgerät für die Wasserventile
- drei Relais zum Steuern der Wasserventile
- drei Relaissockel
- drei LEDs als Anzeigen der Ventilaktivierung
- drei Widerstände für LEDs
- ein Netzgerät für Arduino (damit er läuft, auch wenn kein Computer angeschlossen ist)

Jetzt, wo du eine vorläufige Liste hast, lass uns jedes Teil ansehen und die Details ausarbeiten. Lass uns mit der RTC beginnen