Robotik für Maker

Tauchen Sie ein in die Welt der Robotertechnik und verwirklichen Sie, wovon Isaac Asimov noch geträumt hat

FRANZIS

Bibliografische Information der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte Daten sind im Internet über http://dnb.ddb.de abrufbar.

Autoren: E. F. Engelhardt, Mattias Schlenker, Heinz Schmid, Ulli Sommer, Dr. Günter Spanner, Ulrich Stempel

ISBN 978-3-645-39078-1

INHALTSÜBERSICHT

Ulrich Stempel

Der kleine Hacker: Roboter konstruieren und programmieren

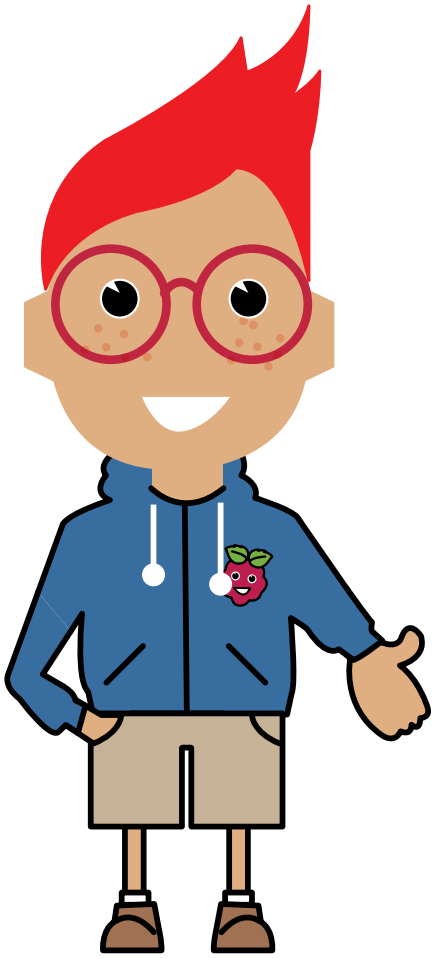
Vom Zahnbürstenbot zum autonomen Roboter:
Baue eigene Roboter und lerne spannendes
Hintergrundwissen!

Inhalt

Das DVD-Zusatzmaterial für den kleinen Hacker kannst du dir einfach hier herunterladen: Klick auf diesen Text und starte den Download!

Vorwort

Wenn du dich mit Robotern beschäftigst, dann wirst du Spezialist für das Thema „Roboter“. Man nennt dieses Gebiet auch Robotik. Dabei geht es um alle Arten von Robotern, z. B. um Industrieroboter, die Autos herstellen, Haushaltsroboter, Forschungsroboter auf dem Mars und vieles mehr. Das Thema ist spannend, brandaktuell und zukunftsweisend. Der kleine Hacker erforscht das komplexe Zusammenspiel von Mechanik und Elektronik. Mit diesem Buch wirst du zusammen mit dem kleinen Hacker zum Roboterexperten und entwickelst deine eigenen Roboterkreationen.



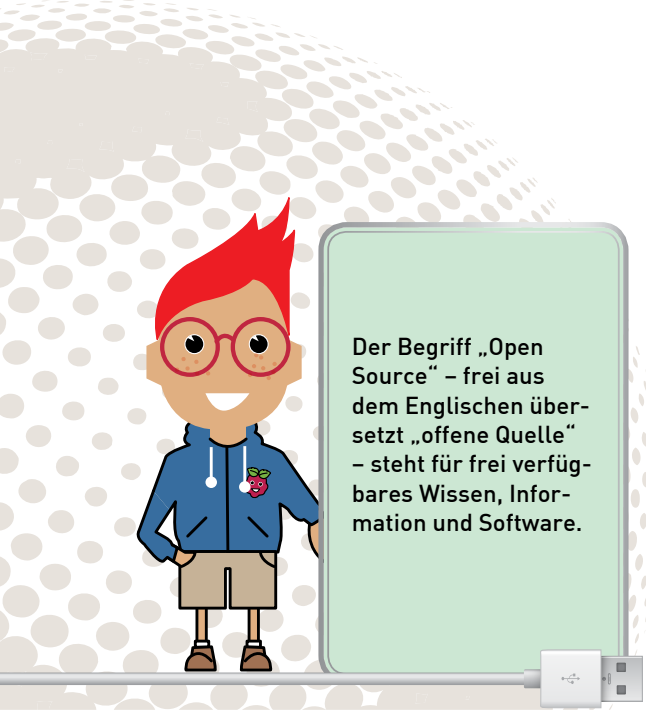
3 Roboter mit Mikrocontroller

Wer gerne einen Roboter frei programmieren möchte, für den gibt es eine große Vielfalt an unterschiedlichen Anwendungen. Du kannst mit den Reaktionen des Roboters so lange spielerische Experimente und Erfahrungen machen, bis er das macht, was du dir vorgestellt hast.

WAS IST EIN MIKROCONTROLLER UND WOZU IST ER GUT?

Ein Mikrocontroller ist ein kleiner Computer, aufgebaut auf einer kleinen Platine, die den Prozessorkern, einen Speicher und programmierbare Ein-/Ausgangsanschlüsse enthält. Im Handel gibt es einige Möglichkeiten der praktischen Umsetzung. Für eigene Forschungen sind kleine, komplexe und preiswerte Einplatinen-Computer (Singleboard-Mikrocontroller) mit den erforderlichen Verbindungsanschlüssen, wie z. B. USB sowie Ein- und Ausgängen, geeignet. Um einige auf dem Markt angebotene Modelle zu nennen: Raspberry Pi, Banana Pi, Cubieboard, Arduino und andere mehr. Wenn du die Software für den kleinen Computer frei und ohne erforderliche Lizenzen aus dem Internet herunterladen kannst, so spricht man von „Open Source".

Für diese Singleboard-Mikrocontroller gibt es umfangreiches Zubehör, sodass du das Robotergehirn mit den Sensoren und dem Bewegungsapparat des Roboters verbinden kannst.



Der Begriff „Open Source" – frei aus dem Englischen übersetzt „offene Quelle" – steht für frei verfügbares Wissen, Information und Software.

WIE PROGRAMMIERE ICH DEN MIKROCONTROLLER DES ROBOTERS?

Die Programme kannst du entweder von einer dem Roboter-Kit beiliegenden CD oder aus dem Internet herunterladen. Du kannst sie auch einfach am Computer in einzelnen Teilen umprogrammieren oder komplett neu programmieren. Das von dir erstellte Programm wird mit dem USB-Verbindungskabel vom Computer auf die Platine des Roboters übertragen und dort gespeichert.

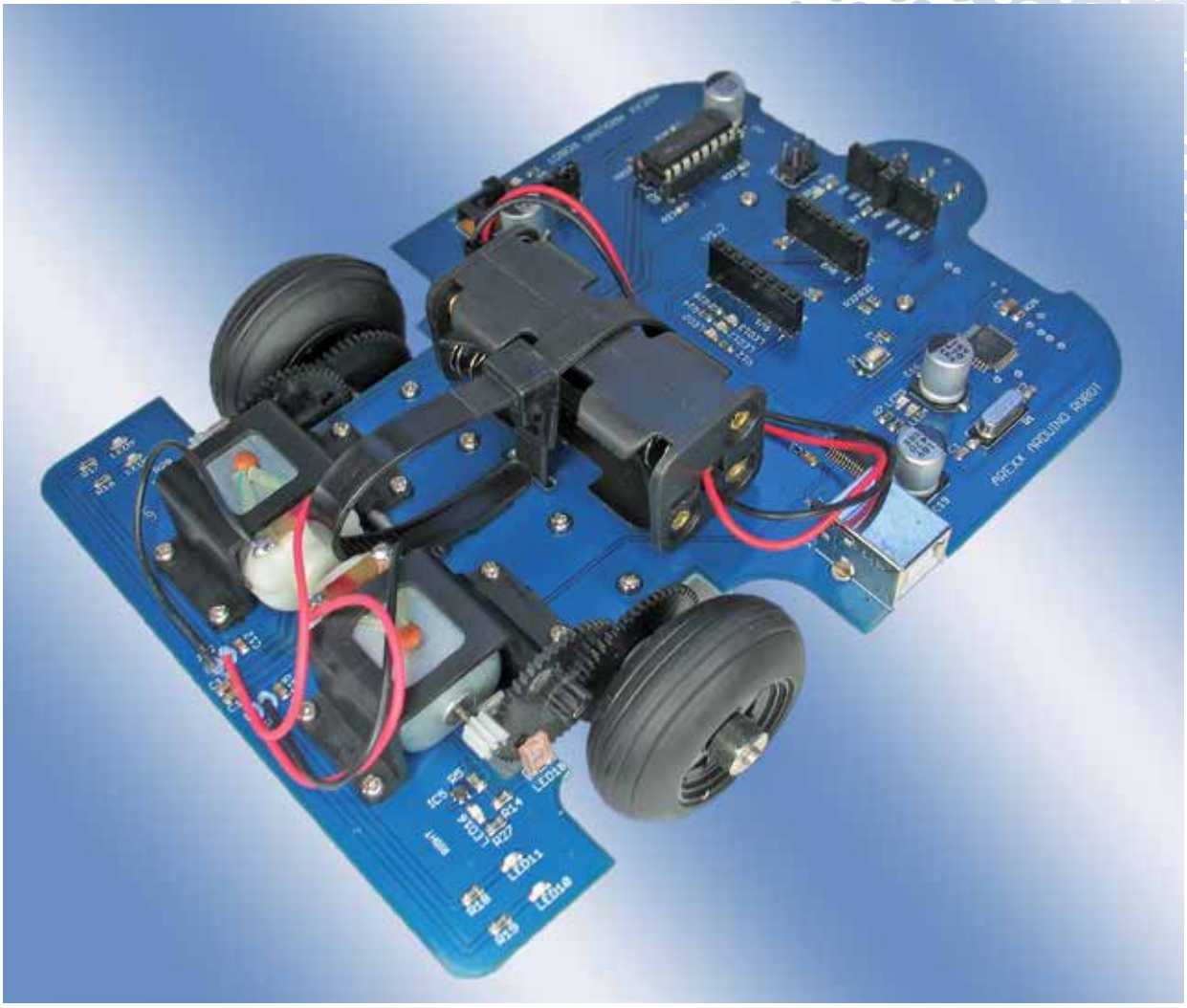
ROBOTER IN AKTION

Neben dem Robotergehirn brauchst du einen Bewegungsapparat und die Sensoren. Praktisch ist es, wenn du eine universale, fahrbare Basis hast und für die unterschiedlichen Anwendungen zusätzliche Teile und Sensoren anbauen kannst. Die Basis kannst du natürlich komplett selbst aufbauen, z. B. mit einem Arduino-Board. Einfacher und meist auch preiswerter ist es, ein Grundmo-

dell mit dem kleinen Computer (on board) und Motoren komplett zu kaufen. Dieses Grundmodell besteht in der Regel aus dem Mikrocontroller, den Schnittstellen, den Motoren mit Getriebe, einem Steckplatz für eine Speicherplatte, einfachen Sensoren, einem Batteriehalter und, was ganz wichtig ist, einer USB-Schnittstelle.

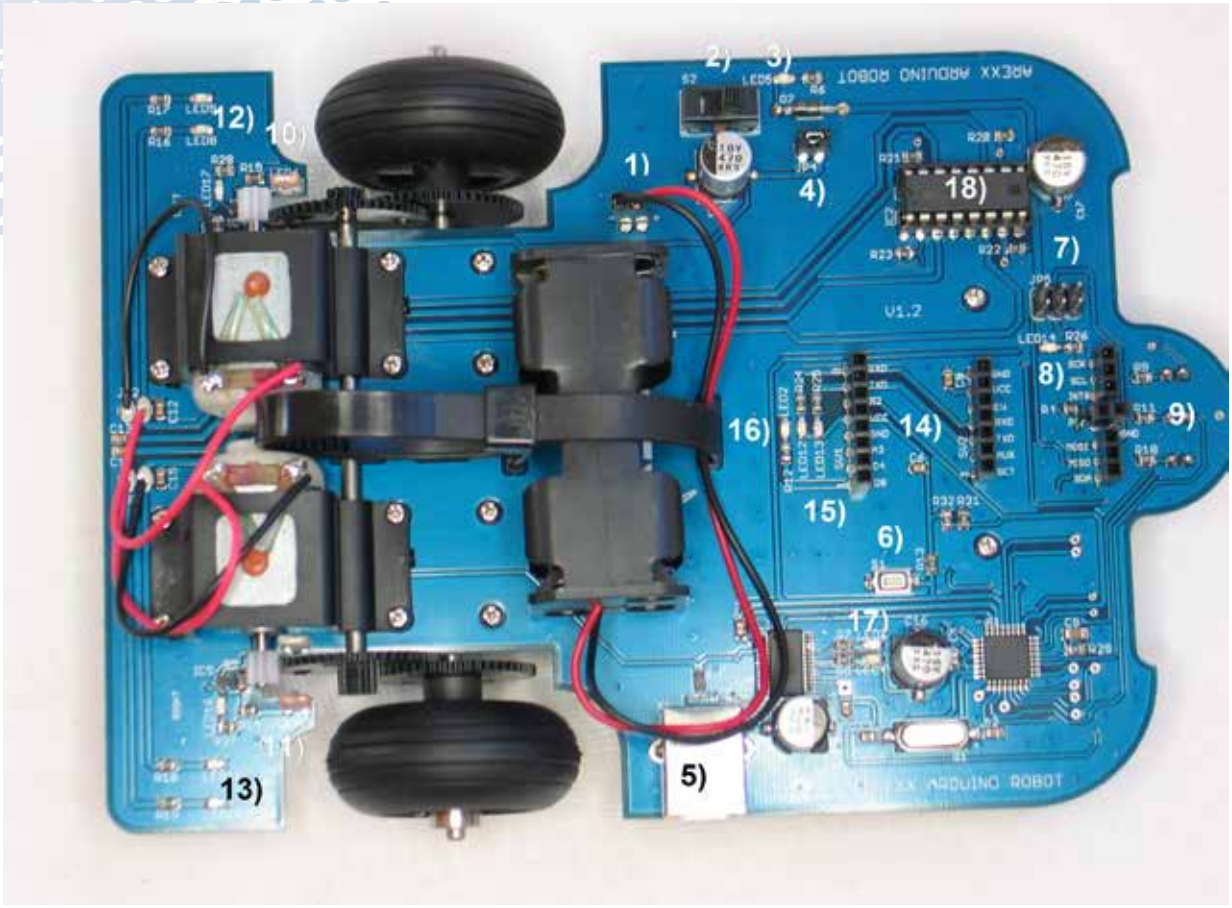
Bei dem in der Abbildung auf der nächsten Seite von oben gezeigten Modell kannst du folgende Komponenten erkennen:

0 Anschluss für den Batteriehalter (beachte bitte die Polarität!).



Beispiel: Roboter AREXX 04 mit Arduino on board

3 Roboter mit Mikrocontroller



Der Roboter von oben

2 Ein-/Ausschalter für den Roboter.

3 Status-LED (LED 5): Anzeige, dass der Roboter von der Stromversorgung gespeist wird.

4 Pin-Paar: Falls du aufladbare Akkuzellen verwendest, kannst du dieses Pin-Paar überbrücken.
(Achtung: Die Polarität-Überprüfung mittels Diode wird dabei abgeschaltet.)

5 USB-Anschluss zum Programmieren des Roboters mittels Arduino-Software.

6 Reset-Taste für den Reset des Roboters.

7 ISP-Steckverbinder: Damit kannst du einen anderen Bootloader einlesen.

8 LED 14: Diese LED ist frei programmierbar und blinkt, falls der Bootloader neu gestartet wird.

9. Linienfolger: Dieses Modul ist frei programmierbar, zum Beispiel so, dass der Roboter einer Linie folgen kann.

10. Radsensor links: Er liefert eine Impulsfolge beim Drehen des linken Rads.

11. Radsensor rechts: Er liefert eine Impulsfolge beim Drehen des rechten Rads.

12. Status-LEDs für den Motor auf der linken Seite: Diese LEDs zeigen an, ob dieser Motor vorwärts- oder rückwärtsläuft.

13. Status-LEDs für den Motor auf der rechten Seite: Diese LEDs zeigen an, ob dieser Motor vorwärts- oder rückwärtsläuft.

14. Steckverbinder für Erweiterungsplatinen.

15. Status-LEDs für die RS232-Data-Kommunikation (LED 12 und 13).

16. Status-LED 2: frei programmierbare LED.

17. Status-LEDs für die USB-Data-Kommunikation.

18. Motorcontroller.

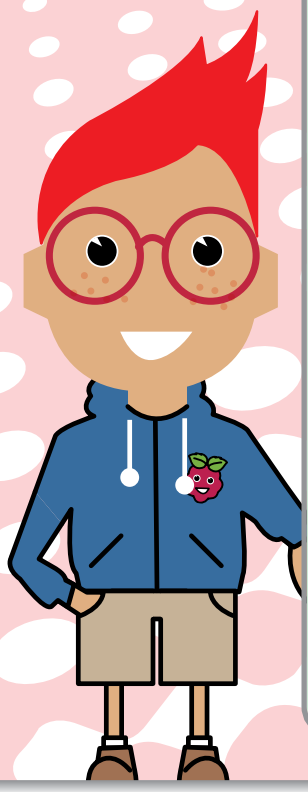
ROBOTERPROJEKT MIT ARDUINO

Für dein konkretes programmierbares Roboterprojekt kannst du entweder eine Arduino-Platine mit entsprechenden Shields oder eine Roboterbasis mit einem Arduino-Mikrochip verwenden. Die hier gezeigten Beispiele wurden mit dem AAR-04 der holländischen Firma Arexx durchgeführt. In ähnlicher Art eignen sich auch andere Basisaufbauten.

Der AAR-04 ist ein autonomer, multisensorieller Roboter. Der Vorteil ist, dass die Basisplatine fertig aufgebaut und bereits mit einigen Sensoren ausgestattet ist, so z. B. mit zwei Odometer-Sensoren und einer optischen Einheit zur Verfolgung einer Linie. Nachteilig ist die oft unzureichende Verarbeitungsqualität. So gibt es immer wieder Probleme mit dem FTDI-USB-Treiber und auch verarbeitungstechnische Mängel (mehr dazu weiter unten beim kleinen Hacker-Roboter-Doktor).

Auf der Platine des AAR gibt es einige LEDs, die die Betriebszustände des Roboters anzeigen, und es gibt frei programmierbare Eingänge und Ausgänge. Über die USB-Verbindung kannst du den Roboter über den PC oder den Laptop programmieren und das Programm zum Robotergehirn übertragen.

Programmiert wird in der „Arduino-Sprache“, die ein vereinfachter C-Dialekt ist. Das Tolle daran ist, dass es sich um Open-Source-Programme und -Software handelt. So kannst du viele Anregungen im Internet finden. Dieser Arduino-Roboter ist sehr gut für einfache Experimente, Robotererfahrung und auch für die Schule und die weitere Ausbildung im Fachbereich Robotic geeignet.

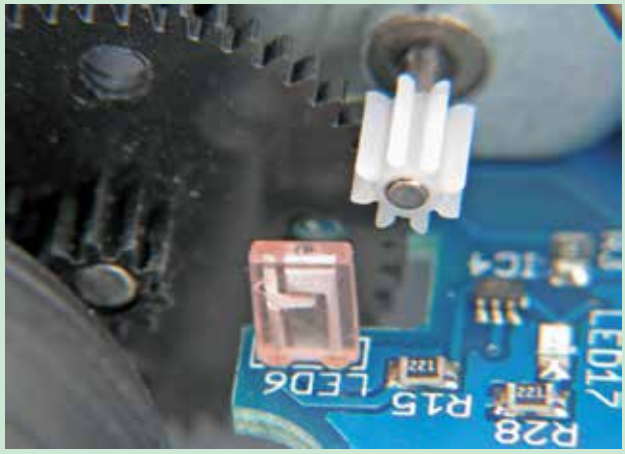


WAS IST ODOMETRIE BZW. WAS MACHEN ODOMETER-SENSOREN?

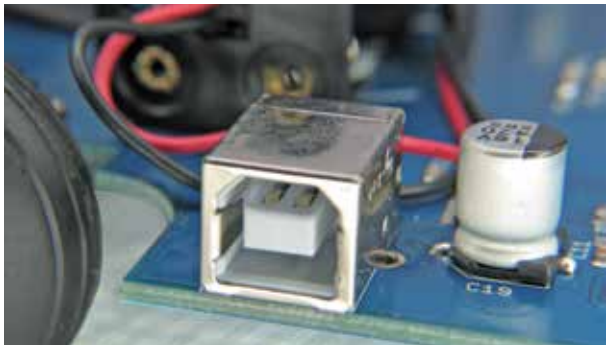
Erst einmal zur Odometrie: Der Begriff kommt aus dem Griechischen, und es geht um die Positionsbestimmung eines Fahrzeugs. Durch eine kleine Lichtschranke an der Radachse erhält das Robotergehirn bei jeder Umdrehung Signale. Durch die Anzahl der Signale kann das Robotergehirn z. B. den zurückgelegten Weg des Roboters errechnen und bestimmen.

Konkret sieht das so aus: An jedem Rad befindet sich entweder eine Scheibe, auf der sich weiße und schwarze Segmente abwechseln, oder eine Scheibe mit Löchern.

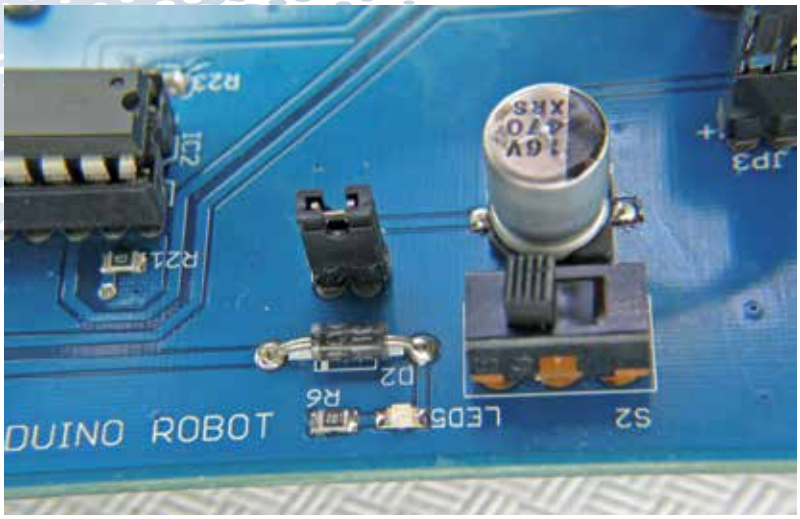
Ein Lichtschranke, bestehend aus einem Fototransistor, und eine Infrarot-LED messen die reflektierten Strahlen. Beim AAR-04 sind Löcher in eines der Zahnräder des Getriebes gebohrt. Wenn die Lichtstrahlen der Infrarot-LED durch das Loch fallen, bekommt der Fototransistor das Signal und schaltet durch.



Die Fotodiode und das schwarze Zahnrad mit vier Bohrungen



USB-Verbindung



Ein-/Ausschalter und Jumper für Akkubetrieb

sofort loslegen. Natürlich müssen die Batterien noch beschafft und eingelegt und die erforderlichen Programme auf dem Basiscomputer installiert werden. Bei Akkubetrieb, der auch möglich ist, wird die Diode mit einem kleinen Jumper überbrückt.

UNTERSCHIEDE AAR-04 – ASURO

Wenn du dich mit einfachen programmierbaren Robotermodellen beschäftigen möchtest, so kannst du neben anderen Produkten auch beide Modelle, den Asuro und den AAR-04, erforschen. Um die Unterschiede zwischen den beiden Arduino-Robotern zu zeigen, hier ein kleiner Überblick. Die gute Nachricht: Viele Erweiterungs-Kits, die für den Asuro angeboten werden, können auch für den AAR-04 verwendet werden.

Die Ähnlichkeiten des AAR-04 zum Vorgängermodell, dem Asuro, der ebenfalls von Arexx produziert wird, sind sehr groß, abgesehen davon, dass der Asuro ein Bausatz ist, der zusammengelötet werden muss. Der AAR-04 ist ein fertig aufgebauter Basisroboter. Nach dem Auspacken kannst du

	AAR-04	Asuro
Prozessor:	ATmega328P @16 MHz	ATmega8 @8 MHz
Speicher:	32 kB Flash, 2 kB RAM, 1 kB EEPROM	8 kB Flash, 1 kB RAM; 512 B EEPROM
Programmierung:	C/C++ Arduino IDE, AVR-GCC oder AVR-Studio	C/C++ AVR-GCC oder AVR-Studio
Sensoren:	Liniensensor 2 x analog, Rad-Encoder 2 x digital	Liniensensor 2 x analog, Rad-Encoder 2 x analog 6 x Taster
Aktive LEDs:	L293D Dual H-Bridge 4 x Status-LED 1 x Front-LED rot 6 x Back-LED	Transistor Dual H-Bridge Status-LED 2 1 x Front-LED rot 2 x Back-LED

3 Roboter mit Mikrocontroller

DAS ROBOTERSYSTEM

Der Arduino AAR-04-Roboter ist mit einer Leiterplatte ausgestattet, die mit dem Arduino Duemilanove-Modell weitgehend identisch ist. Als Mikrocontroller wird der ATmega328P verwendet, der über 14 digitale Ein- und Ausgänge verfügt. Sechs der Kanäle können als pulsbreitenmodulierte Ausgänge (PBM-Ausgänge) verwendet werden. Außerdem verfügt der Roboter über sechs Analogeingänge, die über einen DC/AC-Wandler gehen. Das System arbeitet mit etwa 5 V Gleichspannung, also auch mit einer USB-Spannungsversorgung. Bei vier Akkuzellen reicht die Versorgungsspannung nur knapp, wenn die Akkus voll geladen sind.

PRINZIP DER MOTORANSTEUERUNG

Damit die Signale des Robotergehirns in mechanische Bewegung umgesetzt werden können, braucht es eine Ansteuerung für die Motoren. Diese kann mit einer H-Brücke aus einzelnen Transistoren oder mit einem integrierten Schaltkreis aufgebaut sein. Bei den meisten programmierbaren Robotern wird dafür der integrierte Schaltkreis L293D verwendet. Mit diesem ist es möglich, zwei Motoren mit einem Ausgangsstrom von 600 mA (maximal 1,2 A) mit doppelter H-Brückenschaltung anzusteuern. Die Motoren können damit sowohl vorwärtsdrehend als auch rückwärtsdrehend „fahren".

DEN ROBOTER IN BEWEGUNG SETZEN

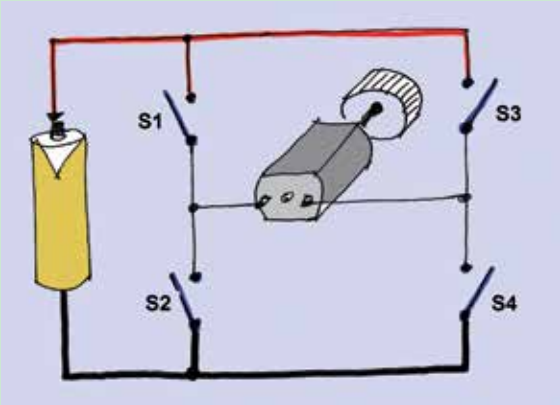
Unabhängig davon, welchen programmierbaren Roboter du dein Eigen nennst, gibt es nachfolgend einige Hinweise (für Arduino), welche Schritte erforderlich sind, um den Roboter zum Leben zu erwecken. Als konkretes Beispiel dient hier wieder der AAR-04.

L293D-Chip mit integrierter H-Brücke

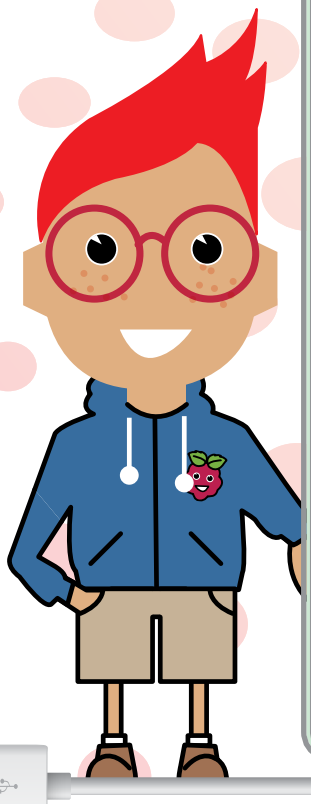
INSTALLATION DER ARDUINO-SOFTWARE

Der erste Schritt ist, die erforderliche Grundsoftware von der dem Roboter beiliegenden CD zu installieren. Wenn erforderlich, kannst du später auch die Arduino-Webseite besuchen und von dort aktuellere Versionen abrufen.

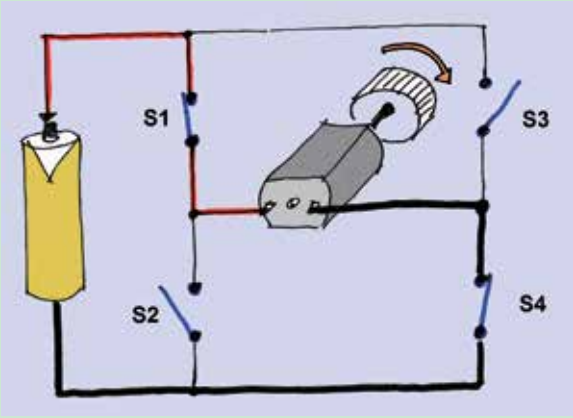
Wenn du einen Motor in beide Richtungen (vorwärts und rückwärts) betreiben möchtest, so wäre es möglich, eine Steuerung mit vier Schaltern dafür aufzubauen. Dadurch kannst du die Stromrichtung des Motors ändern.



Motorsteuerung mit Schaltern



Soll sich die Motorachse nach rechts drehen, werden die Schalter S1 und S4 geschlossen. Der Strom fließt vom Pluspol über S1 zum Motor. Aus dem Motor heraus fließt der Strom über S4 wieder zurück zum Minuspol der Spannungsquelle.



Motor rechtsdrehend

Wenn die Motorachse linksherum drehen soll, werden die Schalter S3 und S2 geschlossen. Jetzt fließt der Strom über S3 zum Motor und dann über S2 zurück zur Spannungsquelle.

E. F. Engelhardt

Roboter mit Raspberry Pi

Mit Motoren, Sensoren, LEGO® und Elektronik
eigene Roboter mit dem Pi bauen, die Spaß machen
und Ihnen lästige Aufgaben abnehmen

Vorwort

Wer sich mit Mechanik und Robotik in Kombination mit dem Raspberry Pi beschäftigen möchte, der hat sich ein komplexes Themenfeld ausgesucht. Grundsätzlich ist neben einem Grundverständnis für die Elektronik und die GPIO-Anschlüsse auf dem Raspberry Pi der Wille notwendig, sich mit der Programmierung des Raspberry Pi auseinanderzusetzen.

Gerade wer sich noch nicht lange mit dem Thema Raspberry Pi beschäftigt, für den stellen die GPIO-Anschlüsse noch eine Wissenschaft dar, auch Änderungen im grundsätzlichen Setup des Raspberry Pi sind anfangs eine spannende Angelegenheit. Doch wenn Sie in Sachen Linux und Shell-Umgang schon fit sind, dann sollten Sie sich mit der Raspberry-Pi-Sprache Python beschäftigen: Hier bieten zig Codebeispiele umgehend passende Lösungen zu verschiedenen Problemstellungen an. Egal ob einfach und banal oder technisch kompliziert und verschachtelt mit mehreren Python-Bibliotheken: Mit diesem Buch werden Sie Schritt für Schritt zum Raspberry-Pi-Python-Experten, und Ihnen raubt anschließend keine nervige Fehlermeldung mehr den Schlaf.

Um Mechanik- und Robotik-Modelle zu bauen, bieten sich einige Möglichkeiten – die einfachste Lösung ist die Legokiste im Kinderzimmer. Angereichert mit LEGO®-Technic, lässt sich schon einiges bewegen und umsetzen – wer in Sachen Elektronik und Robotik hier Gas geben möchte, holt sich noch LEGO®-Mindstorms ins Haus. Mit und ohne Kinder: Hier sind einige Stunden Beschäftigung garantiert. Wer jenseits der traditionellen Legowelt mit Mindstorms eigene Lösungen bauen möchte, muss sich spätestens jetzt ernsthaft mit der Elektronik und der Programmierung auseinandersetzen. Hier lernen Sie die Gesetze der Physik und der Mechanik spielerisch kennen: Wer damit später spielen – oder besser, sich damit beschäftigen – will, sollte sich auch mit der Programmierung des Raspberry Pi auseinandersetzen. Hier bringt der Raspberry Pi eine Menge an Programmierschnittstellen mit.

Diese Schnittstellen werden auch benötigt, wenn es darum geht, den Raspberry Pi mit Robotern im Haushalt zu koppeln. In diesem Buch wird das an einem Vorwerk/Neato-Staubsaugerroboter demonstriert – hier wird das vorhandene Betriebssystem angezapft und mit dem Raspberry Pi gesteuert. Auch wenn Sie keinen Haushaltsroboter im Einsatz haben, die in diesem Projekt beschriebenen Kniffe wie das Anzapfen des USB-Anschlusses, die Steuerung eines Fahrzeugs über eine Webserversteuerung und vieles mehr lassen sich für weitere Raspberry-Pi-Projekte auch dank des vorliegenden Quellcodes schnell und bequem verwenden. Sie benötigen dafür ebenfalls Programmierkenntnisse – denn Mechanik und Robotik machen mehrere komplexe Denkvorgänge notwendig, damit beispielsweise klar ist, welche Motoren sich wie schnell bewegen müssen, um einen Arm zu bewegen und was mit

diesem erfasst werden kann. Außerdem müssen Sie sich überlegen, wie eine passende Steuerung in Form einer Software zu funktionieren hat, damit die Motoren im richtigen Moment das Gewünschte in der richtigen Reihenfolge tun.

Das Buch ist kein Einsteigerbuch zum Thema Raspberry Pi und Programmierung, doch nach dem Lesen werden Sie feststellen, dass die Mechanik und Robotik mit dem Raspberry Pi kein Hexenwerk ist. Sie brauchen aber etwas Zeit und Geduld sowie den Willen, auftretende Probleme selbst zu lösen. Wir wünschen Ihnen viel Spaß mit den Projekten!

Autor und Verlag

Sie haben Anregungen, Fragen, Lob oder Kritik zu diesem Buch? Sie erreichen den Autor per E-Mail unter *ef.engelhardt@gmx.de*.

Inhaltsverzeichnis

Lenken und Steuern mit der GPIO-Schnittstelle

Die zentrale Schnittstelle für das Messen, Steuern und Regeln von angeschlossenen Schaltern, Sensoren und Aktoren ist die GPIO-Schnittstelle (*General Purpose Input/Output*) des Raspberry Pi. Mit dieser Schnittstelle sind Sie für sämtliche Dinge in diesem Buch gerüstet – hier erweitern Sie den Raspberry Pi mit Schaltungen und Funktionen auf dem Steckboard, die später per Lötkolben in ein festes Platinendesign überführt werden können. Neben den Schaltungslösungen der Marke »Eigenbau« können Sie auch auf die hilfreiche Unterstützung in Form von zusätzlich zu erwerbenden Steck- und Erweiterungsboards für die Steuerung von Motoren aller Art zählen – beispielsweise gewöhnliche Gleichstrommotoren, Schritt- oder Steppermotoren, die gerade in der Robotik und bei Mechanikbasteleien mit dem Raspberry Pi eine wesentliche Rolle spielen.

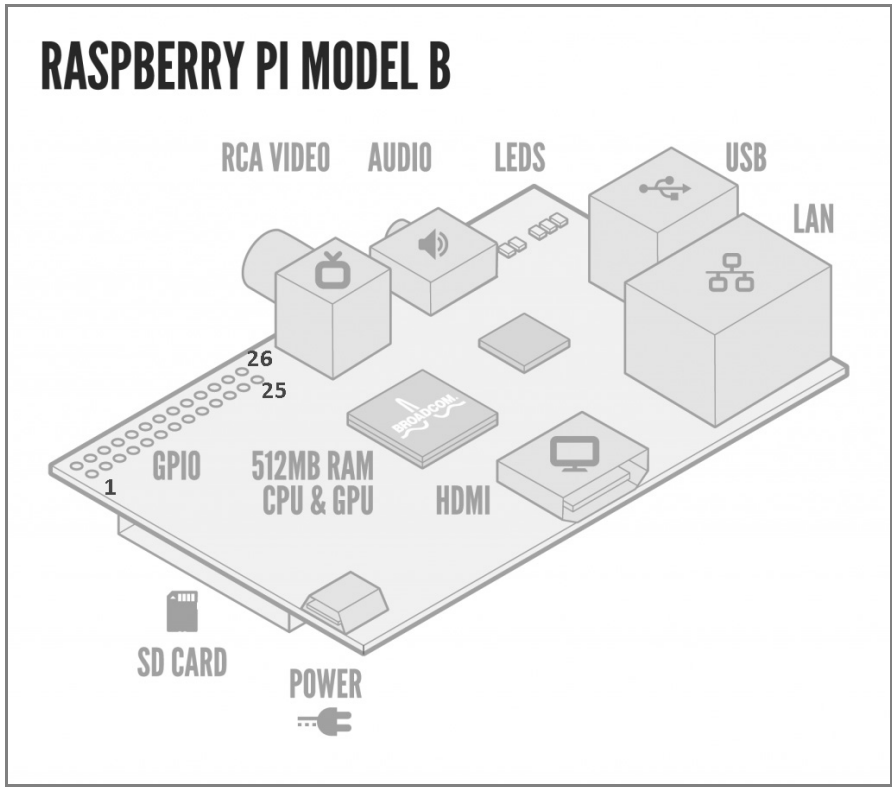


Bild 1.1: Der grundsätzliche Aufbau des Raspberry Pi und der GPIO-Pinleiste Modell B, Revision 2. (Grafik: *raspberrypi.org*)

Neben dem verbesserten Original werden zunehmend Klonplatinen im Internet angeboten, die auf die grundsätzliche Idee und Raspberry-Pi-Technik setzen. Erwähnenswert ist hier die Banana-Pi-Lösung, die nicht nur die gleichen Abmessungen wie das Original besitzt, sondern ebenfalls mit einer Pin-kompatiblen GPIO-Leiste ausgestattet ist, mit der bestehende Projekte eins zu eins auf eine leistungsfähigere Plattform gehievt werden können.

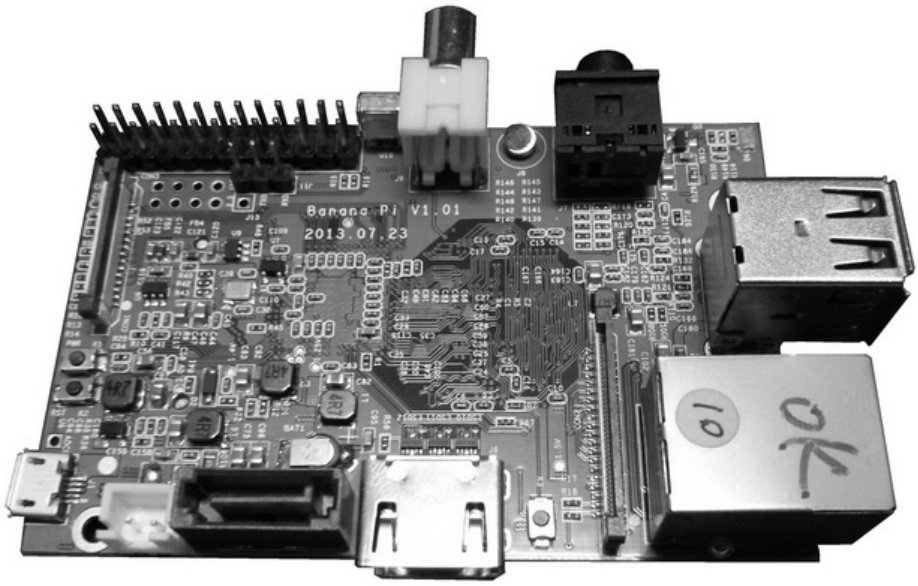


Bild 1.2: Stark verbesserter Raspberry-Pi-Klon, der nahezu die gleichen Abmessungen wie das Original besitzt und ebenfalls eine Pin-kompatible GPIO-Leiste bietet. (Abbildung: *Bananapi.org*)

Der im Vergleich zum Original mit 1 GB verdoppelte RAM-Speicher sowie der ARM Cortex A7-Dualkernprozessor mit 1 GHz Takt und die schnellere Grafikeinheit (Mali 400 GPU) beschleunigen die Platine enorm. Zwar stellt der bei Allnet (*www.allnet.de*) erhältliche Banana Pi mit einem Verkaufspreis von 69 Euro im Vergleich zum Original keine Low-Cost-Lösung mehr dar, er bringt jedoch zusätzlich ähnlich wie das Cubieboard einen SATA-Port mit, mit dem sich eine Festplatte mit bis zu 2 TB Kapazität anschließen lässt.

Wie der Raspberry Pi bietet der Banana Pi die CSI/DSI-Schnittstellen, zwei USB-2.0-Steckplätze sowie je einen HDMI- und einen AV-Videoausgang, außerdem sind noch ein Mikrofon und als zusätzliche Schnittstelle ein Onboard-IR-Empfänger verbaut. Der Klon bringt eine Gigabit-Netzwerkschnittstelle mit, während beim Original mit Ethernet-Anschluss ein 100/10-MBit-Modul verbaut ist.

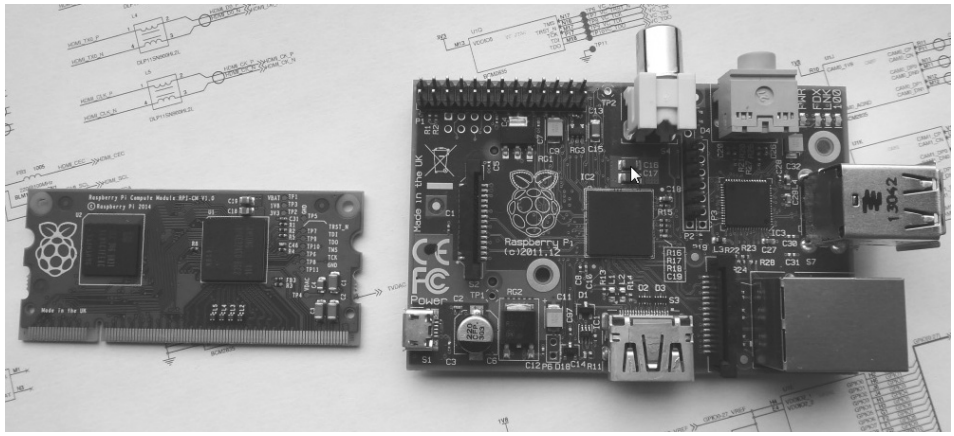


Bild 1.3: Die Raspberry Pi Foundation stellte im April 2014 das deutlich geschrumpfte Raspberry-Pi-Modul vor, das sich nun besser für kleinere, kompakte Anwendungen eignen soll. (Abbildung: Raspberry Pi Foundation)

Die Raspberry Pi Foundation setzt hier weiter auf Miniaturisierung: Das im April 2014 vorgestellte Compute Module ist mit dem BCM2835-Prozessor und 512 MB RAM wie ein »alter« Raspberry Pi ausgestattet. Statt des Speicherkartenslots sind hier 4 GB eMMc-Flash-Speicher direkt verlötet. Das Compute Modul sieht mit den Abmessungen von 67,7 × 30 mm nicht nur aus wie ein Speichermodul, sondern nutzt auch dieselbe Anschlussleiste wie ein DDR2 SODIMM. Dennoch ist es selbstverständlich, den Raspberry Pi nicht in einem RAM-Steckplatz des Computers einzustecken, sondern dafür das sogenannte »Breakout-Board« zu verwenden, das die bisher bekannten Anschlüsse zur Verfügung stellt.

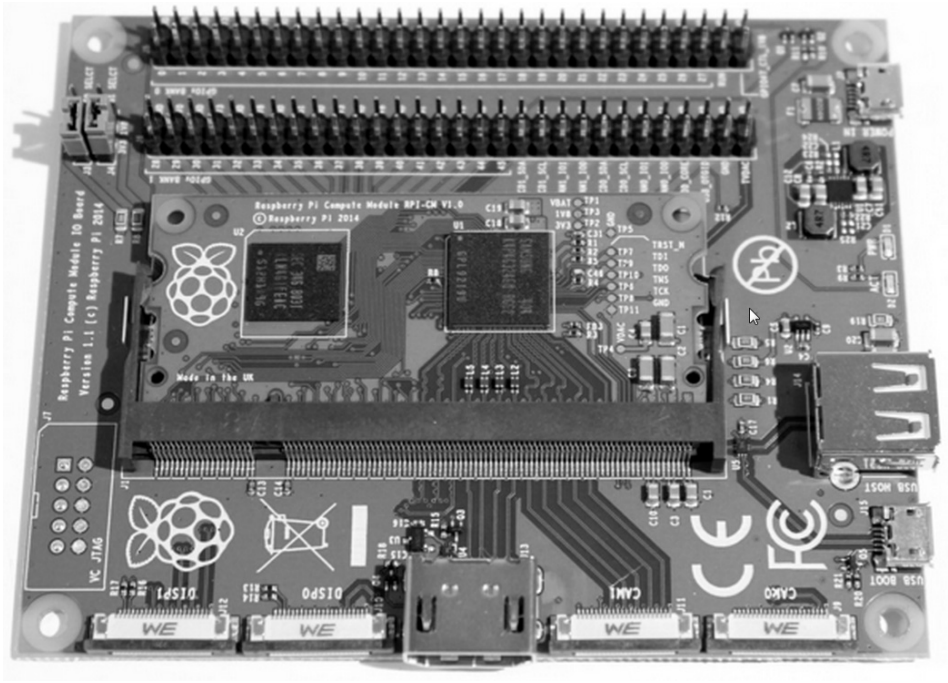


Bild 1.4: Das Raspberry-Pi-Modul wird in das flachere Breakout-Board eingesetzt, was bei mobileren Robotik-Anwendungen manchmal sinnvoller sein kann, wenn die Bauhöhe begrenzt ist. (Abbildung: Raspberry Pi Foundation)

Ab Juni 2014 soll der neue Raspberry Pi samt Breakout-Board im Handel erhältlich sein, im Mai 2014 wird bei einer Abnahme von 100 Stück für das Compute Modul ein Preis um die 30 Dollar fällig – für das Breakout-Board werden schätzungsweise weitere 10 bis 20 Dollar veranschlagt werden.

1.1 Betriebssystem und Treiber aktualisieren

Egal ob Sie selbst eine passende Schaltung im Eigenbau entwickeln oder die eine oder andere Fertiglösung zum Anschluss an den Raspberry Pi nutzen, das A und O ist der Zugriff per Software auf die Schnittstelle bzw. die Funktionen der einzelnen GPIO-Pins. Dafür stehen zahlreiche Möglichkeiten, Zubehör und Erweiterungsboards zur Verfügung, die in den nachfolgenden Kapiteln Schritt für Schritt erklärt und eingesetzt werden. Doch diese technischen Hilfsmittel sind oft wertlos, wenn das Grundsystem nicht stimmt.

Aus diesem Grund stellen Sie mit Kommandos wie `sudo apt-get update` und `sudo apt-get upgrade` sicher, dass der Raspberry Pi mit einem aktuellen Betriebssystem samt Treiber bestückt ist. Nach einem etwaigen Neustart per `sudo reboot` sollten Sie

hin und wieder auch das Kommando `sudo apt-get dist-upgrade` ausführen. Während sich das `apt-get upgrade`-Kommando primär um Anwendungen und Treiber kümmert, sorgt `apt-get dist-upgrade` für den aktuellen Kernel und installiert dessen Aktualisierungen.

Hat das Installationsprogramm hier Änderungen durchgeführt, sollten Sie den Raspberry Pi mit `sudo reboot` neu starten und den dazugehörigen Neustart live mitverfolgen. Nur dann haben Sie die Gewissheit, dass bei der Kernel-Aktualisierung alles gut gegangen ist und alle Dienste wieder starten.

1.2 Analog-digital-Wandler MCP3008 nachrüsten

Anders als bei anderen Mikrocontroller-Boards, beispielsweise aus der Arduino-Ecke, fehlen dem Raspberry Pi trotz der Menge an GPIO-Anschlüssen die analogen Eingänge. Ist der Einsatz von kostengünstigen analogen Sensoren gewünscht, ist entweder der direkte Umweg über einen Arduino oder ein Analog-digital-Wandler-Board notwendig, falls auf dem Raspberry Pi bzw. im zu steuernden Programmcode genauere Messwerte verarbeitet werden sollen. Im Raspberry-Pi-Umfeld ist der Analog-digital-Wandler MCP3008-IC (Datenblatt: *http://bit.ly/OaQwQh* bzw. *http://bit.ly/1fkK3gv*) sehr verbreitet, der für kleines Geld im Elektronikhandel erhältlich ist.

Datenblatt prüfen, Funktionen verstehen

Um die Funktionsweise und das Zusammenspiel der Anschlüsse zu durchschauen, damit Sie den Analog-digital-Wandler MCP3008-IC mit dem Raspberry Pi einsetzen können, benötigen Sie Informationen aus dem Datenblatt dazu, wie die Kommunikation von Mikrocontroller und MCP3008-Chip vonstatten geht.

Ulli Sommer

Roboter selbst bauen

Das große Praxisbuch für Einsteiger und Fortgeschrittene

Vorwort

Roboterbauen wird, wie man in den verschiedensten Foren und Fachzeitschriften beobachten kann, immer populärer. Das liegt daran, dass Mikrocontroller und zusätzliche Peripheriebausteine immer günstiger angeboten werden und auch an den Schulen zunehmend in Mikrocontroller, Computer und Robotik unterrichtet wird.

Dieses Buch bietet zahlreiche detaillierte Bauanleitungen mit Schaltplänen und Programmquellcodes. Für den Einsteiger wie auch für den fortgeschrittenen Roboterentwickler dient es als Leitfaden und nützliches Nachschlagewerk. Die Beispielprogramme und Schaltpläne können für eigene Entwicklungen übernommen werden und dienen als Grundbausteine für eigene Ideen und Projekte. Besonderer Wert wurde bei den Anleitungen auf Nachbausicherheit gelegt. Selbstverständlich wurden alle Projekte aufgebaut und ausführlich getestet.

Der Weg zum eigenen Roboter ist für Anfänger mitunter schwer zu bewältigen. Da die Robotik aus einer Kombination von Elektronik, Mechanik und Software besteht, muss der Erbauer handwerkliches Geschick und Grundkenntnisse in den genannten Bereichen für den erfolgreichen Aufbau eines Roboters mitbringen. Man sollte sich jedoch von anfänglichen Misserfolgen nicht abschrecken lassen, denn es macht enorm viel Spaß, seinen selbst gebauten Roboter zum ersten Mal in Aktion zu sehen. Dieses Buch legt den Grundstein für eigene Konstruktionen.

Die Mikrocontroller-Programme dieses Buchs wurden, bis auf den Tischroboter TR-1, mit den Basic-Compiler *Bascom* geschrieben, der auf der Buch-CD als Demoversion enthalten ist. Die einzige Einschränkung der Demoversion ist, dass Programme nur bis 4 KB RAM kompiliert werden können. Alle Funktionen, die in der Vollversion vorhanden sind, werden auch bei der Demoversion bereitgestellt.

Inhaltsverzeichnis

5 Die Sensoren: Sinne für die Maschinen

Zu den wichtigsten Bestandteilen eines Roboters zählen die Sensoren. Sie dienen dazu, Hindernisse und Umgebungseinflüsse zu messen und den Roboter darauf reagieren zu lassen. Die Roboter müssen sehen und fühlen können, sonst können sie nicht auf ihre Umgebung reagieren.

Als Hindernissensoren kommen Infrarot, Ultraschall und einfache Schalter (Bumper) zum Einsatz.

Zur Messung von Umgebungseinflüssen gibt es Temperatur-, Luftfeuchte-, Licht- und Erschütterungssensoren.

Die bekanntesten und am leichtesten zu beschaffenden Sensoren werden in diesem Kapitel vorgestellt. Die Beispielprogramme zur Auswertung sind für den Bascom-Compiler geschrieben. Er befindet sich als Demoversion auf der mitgelieferten CD-ROM.

5.1 Ultraschallsensoren

Bei Ultraschallsensoren wird ein Ultraschallimpuls von 8-16 Perioden Dauer ausgesendet und die Zeit gemessen, bis das Echo am Empfänger eintrifft. Mit diesem Verfahren wird die Zeit bis zum ersten eingehenden Echo gemessen und über die Schallgeschwindigkeit, die bekannt ist (etwa 300 m/s), der Abstand zum Objekt bestimmt. Die Genauigkeit der Sensoren liegt im Zentimeterbereich. Es gibt auch Systeme mit anderen Verfahren. Kurz nach dem ersten Echo wird der Empfänger wieder empfindlich gemacht, dann werden weitere (maximal sieben) Echos (Bursts) registriert. Auf diese Weise kann man erkennen, ob hinter einem kleinen Hindernis (z. B. ein Ball) noch ein anderes (z. B. eine Wand) vorhanden ist. Bei ungünstiger Montage der Sensoren kann es auch vorkommen, dass der Sendepuls am eigenen Roboter reflektiert wird. Ein „normales“ Echolot liefert dann keine brauchbaren Ergebnisse mehr. Denkbar wäre, dass man einfach alle Ergebnisse weglässt, die kleiner als 10 cm sind.

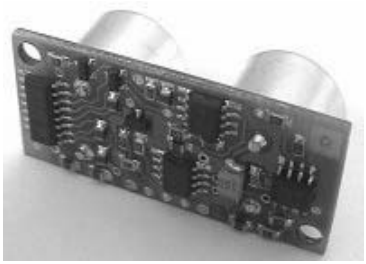


Abb. 5.1: SRF04 von hinten

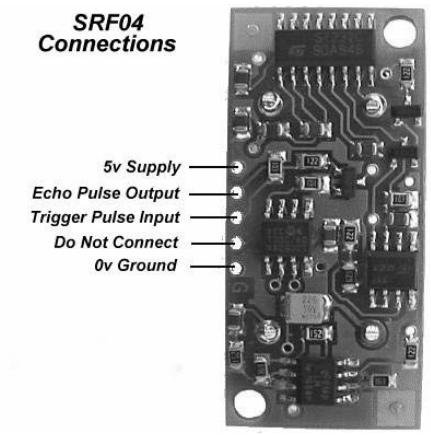


Abb. 5.2: Anschlüsse des SRF04

Ultraschallsensor SRF04

Die Firma Devantech hat eine Serie preiswerter und kleiner Ultraschallsensoren entwickelt, die sich als „autonome Robotersysteme" durchgesetzt haben. Sie sind sehr verbreitet, da sie auch für „Hobbyrobotiker" erschwinglich sind. Es würde sich nicht lohnen, vergleichbare Module selbst zu bauen, da diese meistens ungenauer und teurer wären.

Bekannt wurde die Serie durch den SRF04. Mit ihren kleinen Abmessungen, der niedrigen Stromaufnahme und der hohen Genauigkeit ist sie für kleine Messaufgaben im Entfernungsbereich von 3 cm bis 3 m gut geeignet. Der SRF04 kann einen 3 cm dicken Besenstiel in 2 m Entfernung erkennen und gibt seine gemessene Entfernung als PWM aus.

Ultraschallsensor SRF08

Die Alternative zum SRF04 sind der beliebte SRF08 und dessen Nachfolger SRF10, die nun über den I²C-Bus ausgewertet werden können und über eine Reichweite von 3 cm bis 6 m verfügen. Sie haben eine noch kleinere Stromaufnahme und zusätzlich befindet sich auf ihrer Platinenfront ein Fotowiderstand (LDR), dessen Lichtmesswerte sich ebenfalls über den I²C-Bus auswerten lassen. Durch den SRF08 wird es möglich, bis zu 16 Mehrfachechos von weiter hinten gelegenen Gegenständen auszuwerten, die bei dem SRF04 ignoriert wurden. Über den I²C-Bus kann man die Messwerte in cm, Zoll und in der Laufzeit µs auslesen und spart sich somit die externe Auswertung der Laufzeit, wie sie beim SRF04 erforderlich ist. Weiterhin können insgesamt 16 SRF08-Module an einen I²C-Bus angeschlossen werden.

Abb. 5.3: SRF08

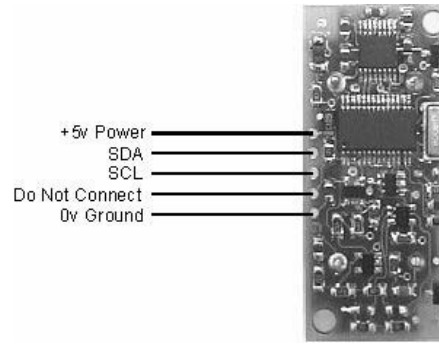


Abb. 5.4: Anschlüsse SRF08

SRF08-Programmbeispiel

```
Const Sf08_adr_0 = &HE0    'I2C-Adresse
Const Sf08_c_range = 100   'Reichweite
Const Sf08_c_gain = 1     'Empfindlichkeit
```

Die Adresse ist der Default-Wert für den Sensor und kann eingestellt werden. Range und Gain sind anzupassen, die angegeben Werte sind aber grundsätzlich verwendbar.

Setup
Nach dem Reset sind einmal Range und Gain zu setzen.

Die Reichweite „Range" einstellen:

```
I2cstart
I2cwbyte Sf08_adr_0     'Device I2C-Adresse
I2cwbyte 2              'Register „range"
I2cwbyte Sf08_c_range
I2cstop
```

Gain einstellen:

```
I2cstart
I2cwbyte Sf08_adr_0     'Device I2C-Adresse
I2cwbyte 1              'Register „gain"
I2cwbyte Sf08_c_gain
I2cstop
```

Abfrage Trigger:
Die Abfrage soll laut Beschreibung in zwei Schritten erfolgen, zwischen denen ca. 70 mS gewartet werden muss. Diese Zeit braucht das Gerät zum Messen.

```
I2cstart
I2cwbyte Sf08_adr_0     'Device I2C-Adresse
I2cwbyte 0              'Register "Trigger"
```

```
I2cwbyte 81                'Messwert in Zentimetern
Waitms 70
```

Messergebnis abholen:

```
DIM Lsb as Byte
DIM Msb as Byte
DIM IVal as word

 I2cstart                  ' Repeated Start
 I2cwbyte Sf08_adr_0       ' Device I2C-Adresse
 I2cwbyte 2                ' Messwert US

 I2cstart                  ' repeated Start
 I2cwbyte Sf08_adr_0 + 1   ' Device I2C-Adresse read!
 I2crbyte Msb , Ack        ' Bit 8-15
 I2crbyte Lsb , Nack       ' Bit 0-7
 I2cstop

 Ival = Makeint(lsb , Msb) ' umwandeln in Word (16 Bit)
```

Ultraschallsensor SRF02

Der Ultraschallsensor SRF02 ist der erste Sensor aus der SRF-Reihe, der mit nur einem Ultraschallwandler auskommt. Dennoch können sich die Leistungen sehen lassen. Vor allem die Tatsache, dass eine RS232- und eine I²C-Bus-Schnittstelle vorhanden sind, dürfte viele Bastler erfreuen. Er ist der aktuellste und vor allem preiswerteste Sensor der SRF-Reihe.

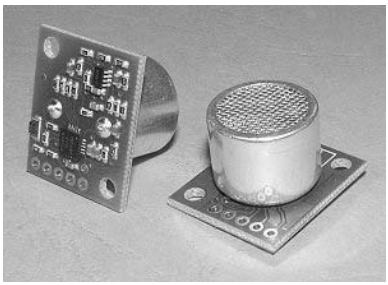


Abb. 5.5: SRF02

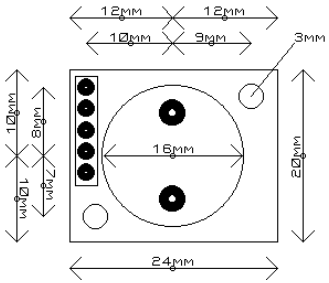


Abb. 5.6: Masse des SRF02

- Betriebsspannung 5 V (stabilisiert)
- Stromaufnahme nur 4 mA (typisch)
- Ultraschallfrequenz 40 kHz
- Reichweite 15 cm bis 6 m
- Schnittstellen RS232 (TTL) und I²C-Bus

- Ausgabeeinheit wahlweise in mm, Inch oder µSek
- einfachste Verwendung, keine Kalibration/Justierung notwendig
- Größe 24 x 20 x 17 mm

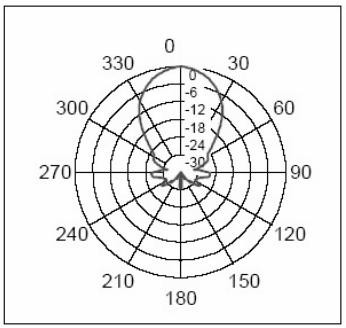


Abb. 5.7: Öffnungswinkel des SRF02.

Der Sensor lässt sich wahlweise per RS232 oder I²C anschließen. Jeweils bis zu 16 Stück können an einer Schnittstelle betrieben werden – auch an der RS232-TTL, was durchaus ungewöhnlich ist.

SRF02 im RS232-Mode

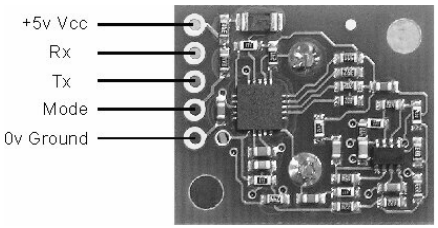


Abb. 5.8: Anschlüsse des SRF02 im RS232 Mode

Beispiel-Code:

```
,#################################################################
, SRF02_RS232.bas
, Gibt die Firmware-Version vom SRF02 aus und
, anschließend in einer Endlosschleife die
, Entfernung von Objekten in Zentimetern
, Die Entfernung wird 1-mal pro Sekunde über RS232
, ausgegeben.
, Dieses Beispiel nutzt den RS232-Mode vom SRF02
, Um die Ergebnisse auch gleichzeitig an den PC
, übermitteln zu können, wird in dem Beispiel eine
, zweite RS232-Schnittstelle per Software eingerichtet.
, Dadurch kann der Sensor (oder auch mehrere) bequem
, an der Steckklemme angeschlossen werden
,#################################################################
```

```
Declare Function Srf02_firmware(byval Slaveid As Byte) As Byte
Declare Function Srf02_entfernung(byval Slaveid As Byte) As Integer

$regfile = "m32def.dat"
$crystal = 16000000
$baud = 9600

Open "COMA.0:9600,8,N,2" For Output As #1
Open "COMA.1:9600,8,N,2" For Input As #2

Const Srf02_slaveid = 0
Dim Entfernung As Integer
Wait 1
Print "SRF02 RS232 Testprogramm"
Print "SRF02 US-Firmware Version:" ;Srf02_firmware(srf02_slaveid)
Print "PA0 wird TX und PA1 wird als RX genutzt

Do

 Entfernung = Srf02_entfernung(srf02_slaveid)
 Print "Entfernung:" ; Entfernung ; "cm"
 Wait 1

Loop
End


'------------- Sub-Routine ----------------------------------

Function Srf02_firmware(byval Slaveid As Byte) As Byte
 Print #1 , Chr(slaveid) ; Chr(93);
 Srf02_firmware = Waitkey(#2)
End Function

Function Srf02_entfernung(byval Slaveid As Byte) As Integer
 Local Lob As Byte
 Local Hib As Byte

 Print #1 , Chr(slaveid) ; Chr(84); ,Messvorgang in cm starten
 Inputbin #2 , Hib , Lob    ,Warte auf Ergebnis
 Srf02_entfernung = Makeint(lob , Hib)
End Function
```

SRF02 im I²C-Mode

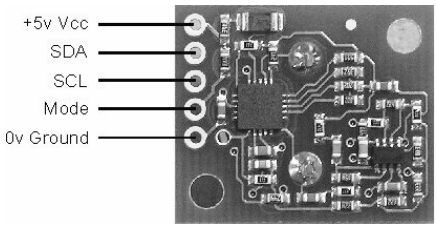


Abb. 5.9: Anschlüsse SRF02 im I²C-Mode

```
Beispiel-Code:
'############################################################
'SRF02_i2c.bas
'Aufgabe:
'Gibt die Firmware-Version vom SRF02 aus und
'anschließend in einer Endlosschleife die
'Entfernung von Objekten in Zentimetern.
'Die Entfernung wird 1 mal pro Sekunde über RS232
'ausgegeben.
'Hier wird noch genau gescheckt, ob ein Sensor die
'Messung schon beendet hat. Dadurch sind noch
'schnellere Messungen als nur 15 pro Sekunde möglich
'############################################################

$regfile = "m32def.dat"
$framesize = 42
$swstack = 42
$hwstack = 42
$crystal = 16000000
$baud = 9600

Config Scl = Portc.0
Config Sda = Portc.1

Declare Function Srf02_firmware(byval Slaveid As Byte) As Byte
Declare Function Srf02_entfernung(byval Slaveid As Byte) As Integer

'Default I2C- Adresse von SRF02
Const Srf02_slaveid = &HE0
Dim Entfernung As Integer

Wait 1 'Warte 1 Sekunde
I2cinit

Print "SRF02 Testprogramm"
Print "SRF02 Ultraschall-Firmware Version:" ;
                Srf02_firmware(srf02_slaveid)
```

```
,------------ Hauptprogramm ----------------------

Do
     Entfernung = Srf02_entfernung(srf02_slaveid)
     Print "Entfernung:" ; Entfernung ; "cm"
     Wait 1
Loop
End

'------------ Hilfsfunktionen für SRF02 -----------

Function Srf02_firmware(byval Slaveid As Byte) As Byte

      Local Firmware As Byte
      Local Slaveid_read As Byte
      slaveid_read = Slaveid + 1

      I2cstart
      I2cwbyte Slaveid
      I2cwbyte 0 'Leseregister festlegen
      I2cstop

      I2cstart
      I2cwbyte Slaveid_read
      I2crbyte Firmware , Nack
      I2cstop

      Srf02_firmware = Firmware

End Function

Function Srf02_entfernung(byval Slaveid As Byte) As Integer

      Local Lob As Byte
      Local Hib As Byte
      Local Firmware As Byte
      Local Temp As Byte
      Local Slaveid_read As Byte
      slaveid_read = Slaveid + 1

      'Messvorgang in starten
      I2cstart
      I2cwbyte Slaveid
      I2cwbyte 0
      I2cwbyte 81            'in Zentimetern messen
      I2cstop

      Warteaufmessung:
      Waitms 1

      Firmware = Srf02_firmware(slaveid)
```

```
      If Firmware = 255 Then Goto Warteaufmessung

      I2cstart
      I2cwbyte Slaveid
      I2cwbyte 2            'Leseregister festlegen
      I2cstop

      I2cstart
      I2cwbyte Slaveid_read
      I2crbyte Hib , Ack
      I2crbyte Lob , Nack
      I2cstop

      Srf02_entfernung = Makeint(lob , Hib)

End Function
```

Die I²C-Adresse des SRF02 ändern:

Will man mehrere SRF02-Sensoren an einem I²C-Bus betreiben, muss man die Adresse der Sensoren ändern. Wie das geht, sieht man am folgenden Beispiel-Code.

Beispiel-Code:

```
'SRF02_Adr_Change.bas

'Mikrocontroller
'AVR-File, Quarz, Baudrate, Hardwarestack, Softwarestack,
'             Framesize
$regfile = "m32def.dat"
$crystal = 16000000
$baud = 19200

'I²C-Pins
'I²C-Bus Pinbelegung (Hardware I²C-Bus)
Config Scl = Portc.0
Config Sda = Portc.1
I2cinit

'Command # 1
I2cstart
I2cwbyte &HE2
I2cwbyte 0             'Register 0 Command Register
I2cwbyte &HA0          '1. Byte Sequenz
I2cstop

'Command # 2
I2cstart
I2cwbyte &HE2
I2cwbyte 0             'Register 0 Command Register
I2cwbyte &HAA          '2. Byte Sequenz
```

```
I2cstop

'Command # 3
I2cstart
I2cwbyte &HE2
I2cwbyte 0              'Register 0 Command Register
I2cwbyte &HA5           '3. Byte Sequenz
I2cstop

'Command # 4
I2cstart
I2cwbyte &HE2
I2cwbyte 0              'Register 0 Command Register
I2cwbyte &HE4           'neue Adresse ist E2
I2cstop

End
```

5.2 Infrarotsensoren

1. Optischer Reflexkoppler

Ein integriertes Sensor-IC gibt Stromimpulse für eine IR-LED aus und detektiert die Echos, die diesem Impulsmuster entsprechen. Man erreicht damit eine gute Unterdrückung der Umgebungshelligkeit. Eine genaue Abstandsmessung ist jedoch mit dieser Methode nicht möglich.

Der Distanzsensor mit IS471F

Das hier vorgestellte *IC IS471F* erlaubt eine einfache und sogar preisgünstige Hinderniserkennung per Infrarot. Dazu muss im Wesentlichen nur noch eine Infrarotdiode

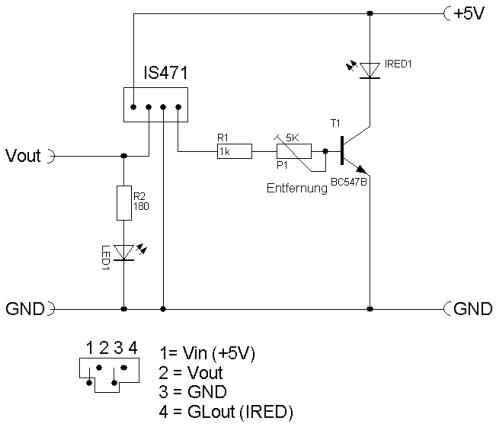


Abb. 5.10: IR-Kollisionsschalter mit IS471F

an das IC angeschlossen werden. Das modulierte Licht wird von einem Gegenstand direkt auf das IC zurückgeworfen und somit das Hindernis erkannt. Tageslicht/ Fremdlicht stört den IS471 nicht, da das Licht mit einer bestimmten Frequenz moduliert wird. Die Reichweite liegt bei der unten aufgeführten Schaltung zwischen 5 und 50 cm. Die Reichweite hängt jedoch stark vom reflektierenden Objekt ab. Bei hellen Gegenständen erreicht man mühelos eine Entfernung von bis zu 50 cm. Dunkle oder sogar ganz schwarze Objekte lassen die Reichweite jedoch bis unter 5 cm sinken.

Die Einbaurichtung des IC erkennt man an den unterschiedlich langen Beinchen. Pin1 ist das kurze Bein außen.

Die IR-empfindliche Fläche ist, gegenüber der beschrifteten Seite, annähernd quadratisch.

2. Optische Triangulation

Sehr viel genauer arbeiten Sensoren, die den Abstand bis zu einem Hindernis trigonometrisch vermessen. Sie bestehen aus einer IR-Diode und einem Empfänger, die zusammen mit einer Steuerung in einem Modul integriert sind. Die Impulse werden wieder nach einem Schema zur Unterdrückung der Umgebungshelligkeit ausgesendet und detektiert. Hier aber handelt es sich um einen gebündelten Lichtstrahl, der auf das Hindernis trifft. Der seitlich versetzte Empfänger – eine *Position Sensitive Device* (PSD) – misst, aus welchem Winkel das Licht reflektiert wird, und setzt den Winkel in einen Spannungswert um. Leider ist der Zusammenhang Entfernung/Spannung nichtlinear. Über entsprechende Formeln oder Tabellen ist eine einfache Umrechnung möglich. Für die Module mit einer Abstandsmessung von 80 und 150 cm kann der folgende Bascom-Code benutzt werden. Er verwendet eine Formel zur Umrechnung auf cm.

Beispiel-Code:

```
Dim Sharp_dist as word

Sharp_dist = Getadc(X)                  'X = ADC Port am AVR
Sharp_dist = Shap_dist - 13             'Umrechnung auf cm
Sharp_dist = 5915  Sharp_dist
Sharp_dist = Sharp_dist - 2

'Begrenzung
If Sharp_dist > 80 then Sharp_dist = 80 'bzw. 150 bei Sensor bis
                                        '150 cm
```

Heinz Schmid

Roboter mit Mikrocontrollern selbst bauen

Robotertechnik richtig verstehen und anwenden

Vorwort

Dieses Buch ist für den Einsteiger geschrieben und gibt einen Überblick über die Robotertechnik. Es befasst sich mit der geschichtlichen Entwicklung und den damit befassten Firmen. Auch auf die Mathematik wird kurz hingewiesen. Sie ist die Basis für diese Maschinen mit ihrer teilweise recht komplexen Steuerungs-Software. Auch auf die Wissenschaftler, die die Grundsteine für diese Konzepte legten, wird eingegangen.

Vor allem aber werden die unterschiedlichen Wege aufgezeigt, Zugang zu diesem Bereich zu bekommen – unterschiedliche Wissensniveaus sowie handwerkliche, finanzielle und zeitliche Möglichkeiten werden berücksichtigt.

Das Buch bietet jenen Neues, die sich mit dem Themenkreis *Roboter* bereits auseinandersetzen, und ist dennoch für den Einsteiger eine Hilfe.

Roboter sind Bestandteil unserer Zukunft. Aus dem Industrie-, Spielzeug- und Ausbildungsbereich sind sie längst nicht mehr wegzudenken, im Alltag steht die Entwicklung aber erst am Anfang. In welcher Art und Weise und für welche Verwendungszwecke sie Einzug halten werden, wird von uns auch als Konsumenten mitbestimmt werden.

Faszinierend, lehrreich und interessant zugleich ist dieses Gebiet allemal – und zwar für Alt und Jung!

Ich freue mich über konstruktive Rückmeldungen, Verbesserungsvorschläge und neue Ideen via E-Mail an: *robots@gmx.net*

Heinz Schmid

Eferding, August 2010

Inhaltsverzeichnis

Anhang

1 Robotik heute

Bekanntlich geht in der modernen Technik nichts mehr ohne Elektronik und Software. Diese Entwicklung ist aus technischer Sicht eine logische Weiterentwicklung des Bestehenden. Mechanischen Apparaten und Vorrichtungen wachsen quasi Nervensysteme, die im Lauf der technischen Evolution immer komplexer werden. Die Robotik erlebt eine faszinierende Entwicklung mit ungeahnten Möglichkeiten. Von den Japanern wird sie mit aller Kraft vorangetrieben. Von den Amerikanern wird sie dank ihrer Anwendungsmöglichkeiten im Weltraum, in der Militärtechnik und im Sicherheitsbereich auf breiter Basis unterstützt. In Europa wird das Thema an den Universitäten und Fachhochschulen gelehrt. Aber oft wird die Robotik auch kritisch und misstrauisch beäugt.

1.1 Industrie

Die Hauptanwendung der Robotik ist derzeit natürlich nach wie vor die Industrie – respektive die Autoindustrie mit ihren Schweiß- und Lackierrobotern. In der Autoindustrie kommen im Schnitt nur mehr 10 Arbeiter auf einen Roboter.

Abb. 1.1 und 1.2: ©KUKA AG – Industrieroboter

Aber auch in kleinen und mittelständischen Betrieben werden aufgrund fallender Preise und der Möglichkeit einer Programmierung auch ohne Spezialkenntnisse vermehrt kleine Industrieroboter eingesetzt. Im Verpackungs- und Handhabungsbereich kommen teilweise auch pneumatische Roboter zum Einsatz.

Die Industrieroboter sind die Stärke der europäischen Roboterindustrie, obwohl ihre Wurzeln in den USA liegen. Der erste kommerzielle Anbieter von »Handhabungsautomaten« war eine amerikanische Firma.

Es gibt sechs Freiheitsgrade im Raum: drei der Translation (Verschiebung) um Δx, Δy oder Δz (bei kartesischem Koordinatensystem) und drei der Rotation (Verdrehung um diese Achsen). Wenn wir einen Roboterarm bauen, der einen mit einem Greifer erfassten Gegenstand innerhalb seines Arbeitsbereichs beliebig verschieben und verdrehen kann, benötigen wir für jeden Freiheitsgrad einen Servomotor plus einen zusätzlichen Motor für das Öffnen und Schließen des Greifers.

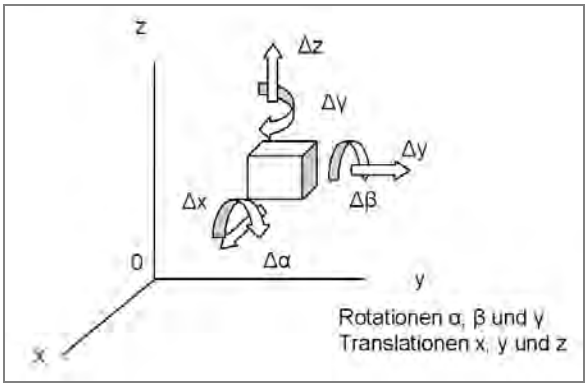


Abb. 1.3
Freiheitsgrade im Raum – degree of freedom

Diese sieben Servomotoren (für eine allgemeine Beweglichkeit im Arbeitsraum plus Greifer) können jetzt konstruktiv sehr unterschiedlich angeordnet werden. Dadurch ergeben sich die verschiedenen Bauformen der Industrieroboter. Jeder von ihnen ist für einen bestimmten Einsatzzweck besonders gut geeignet. Alternativ zu den Servomotoren können auch hydraulische und pneumatische Antriebe oder andere Elektromotortypen verwendet werden. Für kleine Industrieroboter ist die Pneumatik auch aufgrund ihrer Elastizität (Luft ist kompressibel) von Vorteil. Sie kommen dann zum Einsatz, wenn wenig Kraft benötigt wird und keine hohen Positionierungsgenauigkeiten gefordert werden (z. B. beim Palettieren).

Bei sehr großen Kräften und ebenfalls geringer Genauigkeit ist die Hydraulik die beste Wahl. Sollen jedoch komplizierte Bahnen mit einem Werkzeug gefahren werden oder ist eine präzise Positionierung von Kleinteilen notwendig (z. B. bei der Leiterplattenbestückung), sind Schritt- oder Servomotoren die richtige Wahl.

Für die Konstruktion der Automaten werden verwendet:

- Rotationsgelenke
- Torsionsgelenke
- Revolvergelenke

Lineargelenke

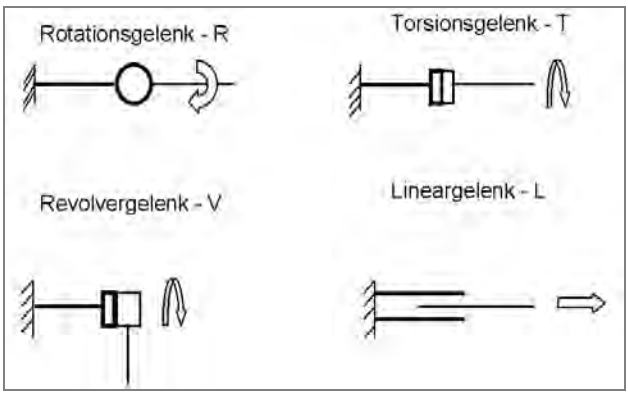


Abb. 1.4: Gelenke

1.1.1 Handhabungsautomaten

Die erste automatisierte Maschine für Produktionszwecke war der Webstuhl (von Joseph M. Jacquard 1806), der mit Lochkarten gesteuert wurde. Es handelt sich hierbei um einen rein mechanischen frei programmierbaren Webroboter.

1945 begann die Entwicklung von Master-Slave-Manipulatoren für die Fernhantierung gefährlicher Stoffe. Für die Atomlabore wurden dann 1951 ferngesteuerte Handhabungsgeräte, sogenannte *Teleoperatoren*, entwickelt. Der Prototyp einer CNC-Maschine wurde 1952 am MIT entwickelt.

Die Firma Planet Corp. stellte dann 1959 einen der ersten kommerziellen Roboter vor, der allerdings noch mechanisch durch Kurvenscheiben und Begrenzungsschalter gesteuert wurde.

Bei vielen Manipulationen von Objekten in der Industrie und im Gewerbe ist es nicht notwendig, dass der Automat über sehr viele Freiheitsgrade verfügt. Oft genügt es, ein kleines Objekt z. B. durch Ansaugen mit Unterdruck aufzunehmen und mit ihm eine einfache Schwenkbewegung zu vollführen, um es über einem Förderband wieder abzusetzen. Dazu ist kein vollwertiger Industrieroboter nötig. Ein einfacher Handhabungsautomat reicht dafür aus.

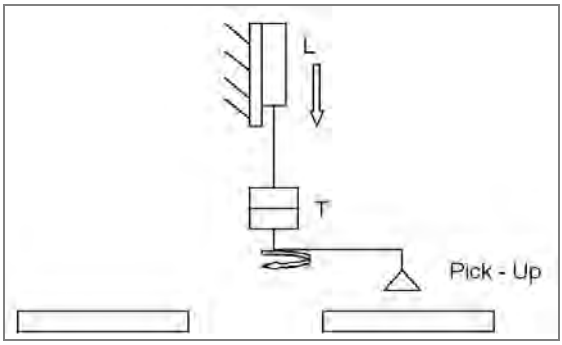


Abb. 1.5:
Handhabungsautomat

1.1.2 Industrieroboter

Der erste Industrieroboter wurde 1958 von George C. Devol (der 1946 die Firma Unimation – Universal Automation gründete) und Joseph E. Engelberger gebaut. Zu Beginn der 60er-Jahre wurde er bereits bei General Motors und der Ford Company eingesetzt. Einer der ersten Industrie-Einsätze war das Heben und Einsortieren von heißen, rund 10 kg schweren Aluminiumdruckgussteilen.

Abb. 1.6:
©UNIMATION INC. – der erste Industrieroboter

Die Unimation war ein hydraulischer Industrieroboter. Er wurde über eine magnetische Trommel programmiert, da damals noch keine kleinen Computer verfügbar waren. Aber erst Mitte der 70er-Jahre war die Firma auch kommerziell erfolgreich. Die vollelektrische *PUMA*-Serie löste dann 1978 die Unimation ab. Der Industrieroboter *PUMA* basierte aber auf Entwürfen von General Motors. »PUMA« steht für *Programmable Universal Machine for Assembly*.

Abb. 1.7:
©UNIMATE – PUMA 700

Die Firma existiert heute nicht mehr. Sie wurde zuerst innerhalb der USA und gegen Ende der 80er-Jahre an die Schweizer Firma Stäubli verkauft.

Der erste Lackierroboter wurde 1966 von der norwegischen Firma Trallfa gebaut. Der erste Einsatz erfolgte aber nicht in der Autoindustrie, sondern beim Lackieren von Schubkarren.

Die Firma Kawasaki Heavy Industries brachte 1969 den ersten japanischen Industrieroboter auf den Markt. Aufgrund der guten Qualität und des günstigen Preises erlangten die Japaner (die Firma Hitachi war der zweite Anbieter) schnell eine starke Stellung am Markt.

Bereits 1975 wurde dann der erste Roboter-Scheinman-Arm mit einem handelsüblichen PC angesteuert.

Derzeit dominieren die europäischen und japanischen Industrieroboterhersteller den Weltmarkt. Sie versuchen, sich auch von der Umklammerung durch die zwischenzeitlich in die Krise gekommene Automobilindustrie zu befreien und auch in anderen Anwendungsbereichen Fuß zu fassen.

Bauformen von Industrierobotern

In Abhängigkeit von der Art der Tätigkeit, die ein Industrieroboter erledigen soll (Heben großer oder kleiner Lasten, präzise Positionierung, Größe und Form des Arbeitsraums, Geschwindigkeit der Bewegung), wird aus der Gruppe der Haupttypen der optimale Industrieroboter ausgewählt. Laut der deutschen Industrie-Norm (DIN) können frei programmierbare Handhabungsautomaten als *Roboter* bezeichnet werden.

Die Haupttypen sind:

Flächenportalroboter – Cartesian Robot

Der Flächenportalroboter hat eine einfache Bauform mit prismatischem Arbeitsraum, ähnlich wie ein Brückenkran mit Laufkatze oder eine Autowaschstraße.

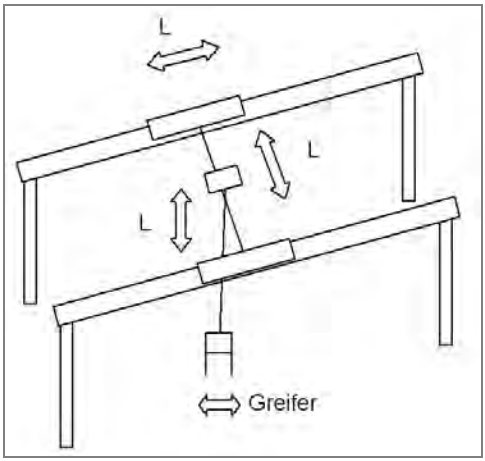


Abb. 1.8: Cartesian Robot

Standsäulenroboter – Cylindrical Robot

Der Standsäulenroboter verfügt über eine einfache Bauform mit zylindrischem Arbeitsraum. Er ist gut geeignet für das Archivieren hunderter gleicher Objekte mit automatischem Zugriff über den Standsäulenroboter

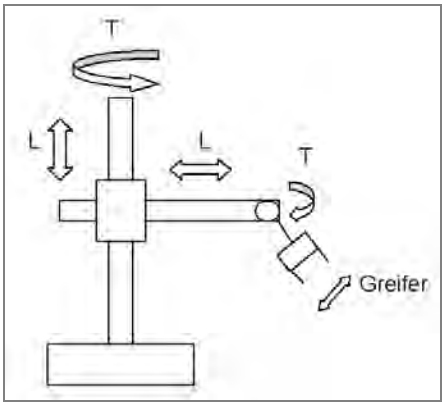


Abb. 1.9: Cylindrical Robot

Polarroboter – Spherical Robot

Der Polarroboter hat einen fassförmigen Arbeitsraum, ähnlich wie den erste Industrieroboter Unimate.

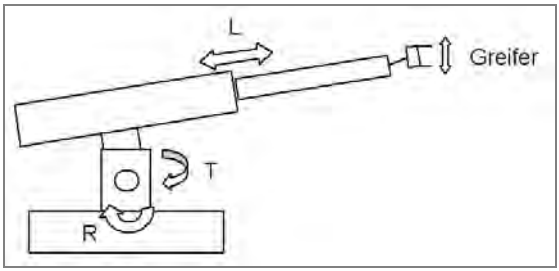


Abb. 1.10: Spherical Robot

SCARA-Roboter – Scara Robot

Der Arbeitsraum des SCARA-Roboters ist ähnlich dem des Standsäulenroboters. Bevorzugt wird der Roboter in der Montage eingesetzt. Anfang der 80er-Jahre wurde er in Japan entwickelt.

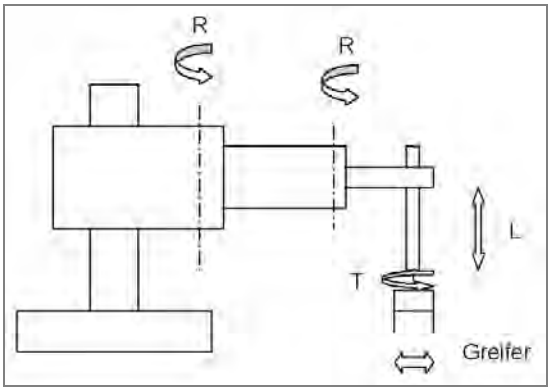


Abb. 1.11: Scara Robot

Gelenkarmroboter – Articulates Robot

Der Gelenkarmroboter ist die Bauform der meisten Industrieroboter in der Automobilindustrie zum Punktschweißen von Autokarosserien, Heben und Einbauen der Windschutzscheibe und des Armaturenbretts. Er kann auch kooperativ arbeiten und z. B. Autokarosserien von der Fertigungslinie heben.

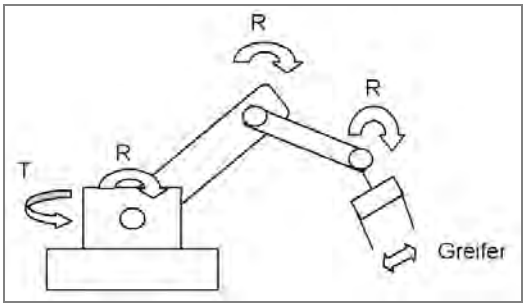


Abb. 1.12: Articulates Robot

Parallel-Kinematikroboter – Parallel Robot

Parallel-Kinematikroboter bestehen, je nach Bauform, aus drei oder sechs längenverstellbaren Stäben, die mit einer Platte verbunden sind. Auf dieser befindet sich meist ein Greifarm. Vom Platzbedarf her ist eine hängende Installation über diversen Produktionslinien der Elektronik- oder Lebensmittelindustrie optimal. Die Systeme zeichnen sich durch große Starrheit und minimalen Drehmomentbedarf aus. Sie ermöglichen auch bei empfindlichen Objekten hohe Bewegungsgeschwindigkeiten.

Tripod – drei Stabkinematik

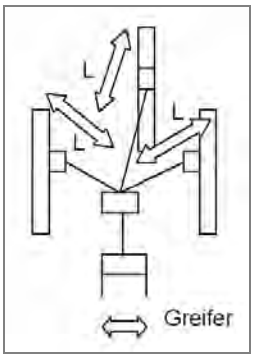


Abb. 1.13: Tripod

Hexapod – 6-Stab-Kinematik

Da die Platte des Hexapod in allen sechs Freiheitsgraden bewegbar ist, wird diese Konstruktion auch als Plattform für Flugzeugsimulatoren verwendet.

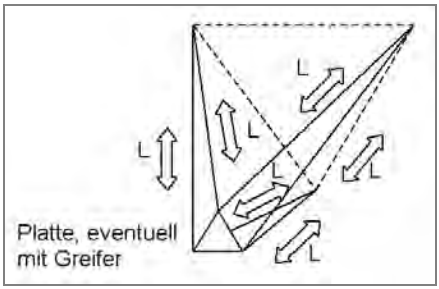


Abb. 1.14: Hexapod

Koordinatensysteme und Programmierung

Aus der Mathematik kennen wir drei Koordinatensysteme:

- rechtwinkeliges kartesisches Koordinatensystem
- Zylinderkoordinatensystem
- Kugelkoordinatensystem

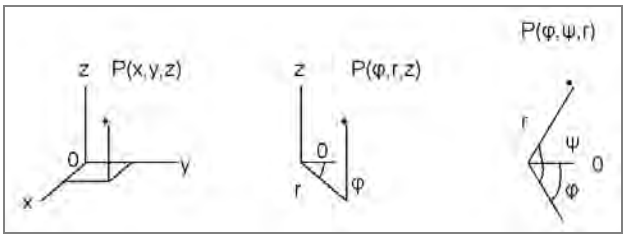


Abb. 1.15: Koordinatensysteme

Bei den Industrierobotern werden meist kartesische Koordinatensysteme verwendet, und zwar mit folgender Einteilung:

Das Weltkoordinatensystem liegt unverändert im Raum und ist meist mit dem Basiskoordinatensystem des Roboters identisch. Die x-y-Ebene liegt in der Aufstellfläche des Roboters. Die z-Achse weist von der Aufstellfläche weg und führt bei einem Drehgelenkroboter durch das Zentrum seiner ersten Drehachse. Damit ist jeder Punkt im Arbeitsraum des Roboters eindeutig definiert.

Beim Gelenkskoordinatensystem werden die Winkel zwischen den einzelnen Armsegmenten vom Basispunkt bis zum Handgelenkflansch angegeben. Damit ist über die Gelenkskette die Stellung im Raum eindeutig definiert.

Ferner muss noch ein Greifer- oder Werkzeugkoordinatensystem im Handgelenkflansch festgelegt werden.

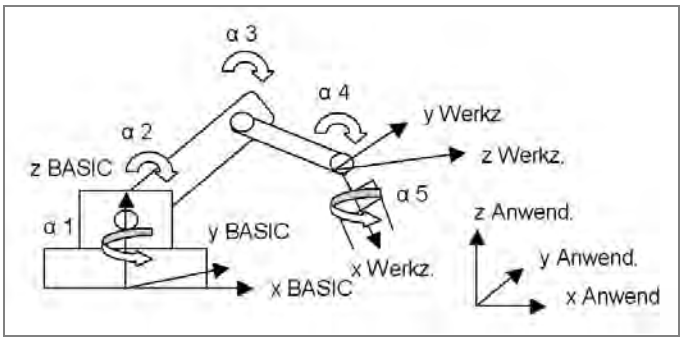


Abb. 1.16: Industrieroboter-Koordinatensysteme

Der Programmierer kann dann innerhalb des Arbeitsbereichs noch ein für Lage und Richtung passendes Anwenderkoordinatensystem definieren.

Ansteuerung des Roboters – Kinematikmodul

Teachen

Teachen ist die einfachste Form der Programmierung. Es werden vom Benutzer einzelne Zielpunkte manuell angefahren und die einzelnen Gelenkspositionen für diese Stellung abgespeichert. Diese einzelnen gespeicherten Stellungen werden dann im Betriebsmodus beliebig oft als Punkt-zu-Punkt-Bewegungen wiederholt. Meist wird dabei nicht der kürzeste, sondern der für den Roboter schnellste Weg verwendet.

Rückwärtsrechnung

Für die Steuerung des Roboters muss also jeder Punkt im Arbeitsraum in die dazugehörenden Gelenkskoordinaten zur Ansteuerung der Motoren umgerechnet werden. Dies ist ein inverses kinematisches Problem, das als *Rücktransformation* bezeichnet wird. Für diese mathematisch komplexe Aufgabe, Vektor- und Matrizenrechnungen mit Transformationsmatrizen in Echtzeit durchführen zu können, ist eine entsprechende Rechenleistung erforderlich.

Wird dem Roboter eine Bewegungsbahn vorgegeben, die er mit einer konstanten Geschwindigkeit abfahren soll, muss eine Bahnsteuerung implementiert werden. Diese ist z. B. für einen Schweißroboter notwendig.

Genügt es hingegen, wie bei einem Punktschweißroboter einzelne Punkte anzufahren, wird eine viel einfacher zu realisierende Punktsteuerung verwendet. Der Bewegungsablauf zwischen den Punkten kann dabei zumindest beeinflusst werden, um Kollisionen zu vermeiden. Die über 1.000 Schweißpunkte einer Autokarosserie werden dabei von mehreren Roboterarmen in weniger als 5 min angebracht.

Dr. Günter Spanner

Coole Projekte mit dem Arduino™ Micro

Physical Computing im Projekteinsatz
Alles für den Einstieg: Platine, ATmega32U4, USB-Zugriff und Programmierung
Projekte für die Praxis: Lichteffekte, Designeruhr, Motorensteuerung, Messen, Steuern und Regeln

Vorwort

Der aus dem Leonardo entstandene Arduino Micro zeichnet sich besonders durch seine kompakte Bauform aus. Kaum größer als ein klassisches IC kann der Baustein direkt in ein Breadboard gesteckt werden. Das ist ein erheblicher Vorteil gegenüber den älteren Varianten der Arduino-Familie wie etwa dem UNO oder dem Mega.

Der Micro kann so unmittelbar mit allen gängigen elektronischen Bauelementen verbunden werden. Spezielle Protoboards oder aufsteckbare »Shields« sind nicht erforderlich. Im Rahmen des Buchs kann diese Arduino-Variante daher als nahezu professionelles Modul betrachtet werden, da der Micro das den anderen Arduinos anhaftende Image eines speziellen »Lernsystems« abgelegt hat.

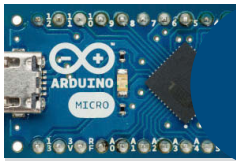
Aufgrund seiner Pinanordnung ist der Micro auch direkt in eine Lochrasterplatine oder sogar in eine IC-Fassung einsteckbar. Ein Schwerpunkt des Buchs liegt daher in der Beschreibung, wie fertig ausgetestete Laboraufbauten schnell und effizient in professionelle und dauerhaft ausgelegte Prototypen oder sogar fertige Kleingeräte integriert werden können.

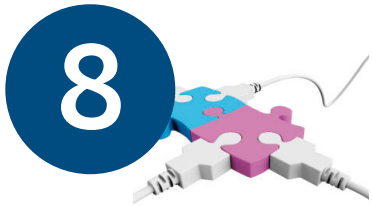
Trotz seiner kompakten Bauform bringt der Baustein alle erforderlichen Komponenten mit. Über eine integrierte Micro-USB-Buchse kann der Controller mit den gewohnten, vielfach bewährten Arduino-Programmiertools mit Software versorgt werden. Darüber hinaus wird im Buch aber auch die Programmierung in C über die professionelle Tool-Chain des Controllerherstellers Atmel erläutert.

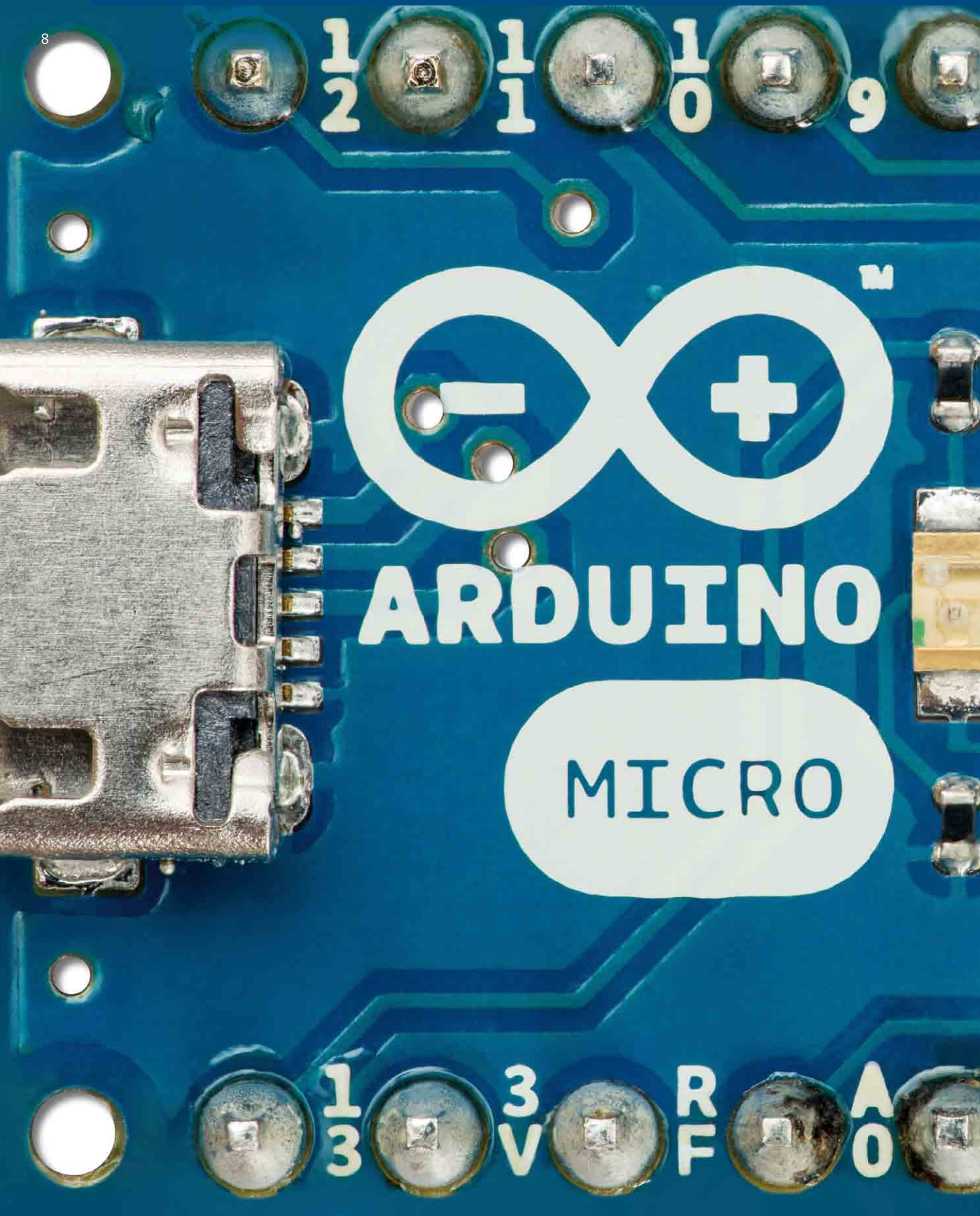
Durch die Integration der USB-Schnittstelle in den Hauptprozessor konnte ein separater USB-zu-RS-232-Wandler eingespart werden. Durch den Wegfall dieses meist recht hochpreisigen Bausteins ist der Micro deutlich günstiger erhältlich als seine Vorgänger. Das trägt dazu bei, dass der Micro direkt, sozusagen als »komplettes elektronisches Bauteil«, in den im Buch beschriebenen Projekten eingesetzt werden kann.

Ein weiterer Vorteil des im Prozessor integrierten USB-Anschlusses ist, dass der Baustein auch als sogenanntes USB-Device/HID (Human Interface Device) konfiguriert werden kann. Damit ist es möglich, den Micro so zu programmieren, dass er sich wie eine USB-Maus oder -Tastatur verhält. Einem direkten Schreiben von Daten in die bekannten Anwenderprogramme wie etwa Word oder Excel steht somit nichts mehr im Weg. Im Buch wird dieser Technik und der Fülle der damit möglichen Anwendungen ein eigenes Kapitel gewidmet.

Korrekturen, Ergänzungen oder Anregungen werden gern unter der Adresse *elo@franzis.de* angenommen.



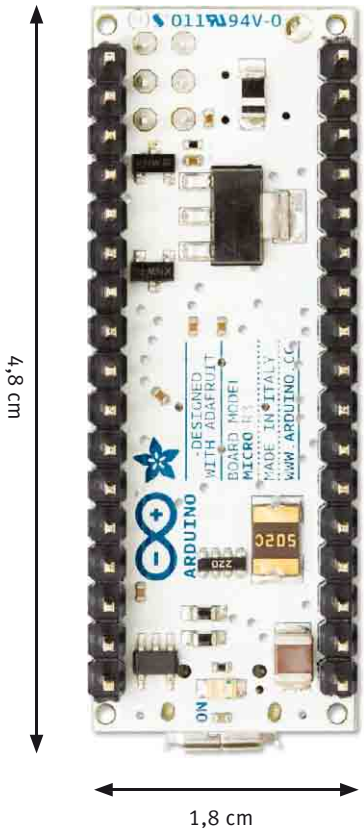



12
11
10
9
ARDUINO
MICRO
13
3V
RF
A0

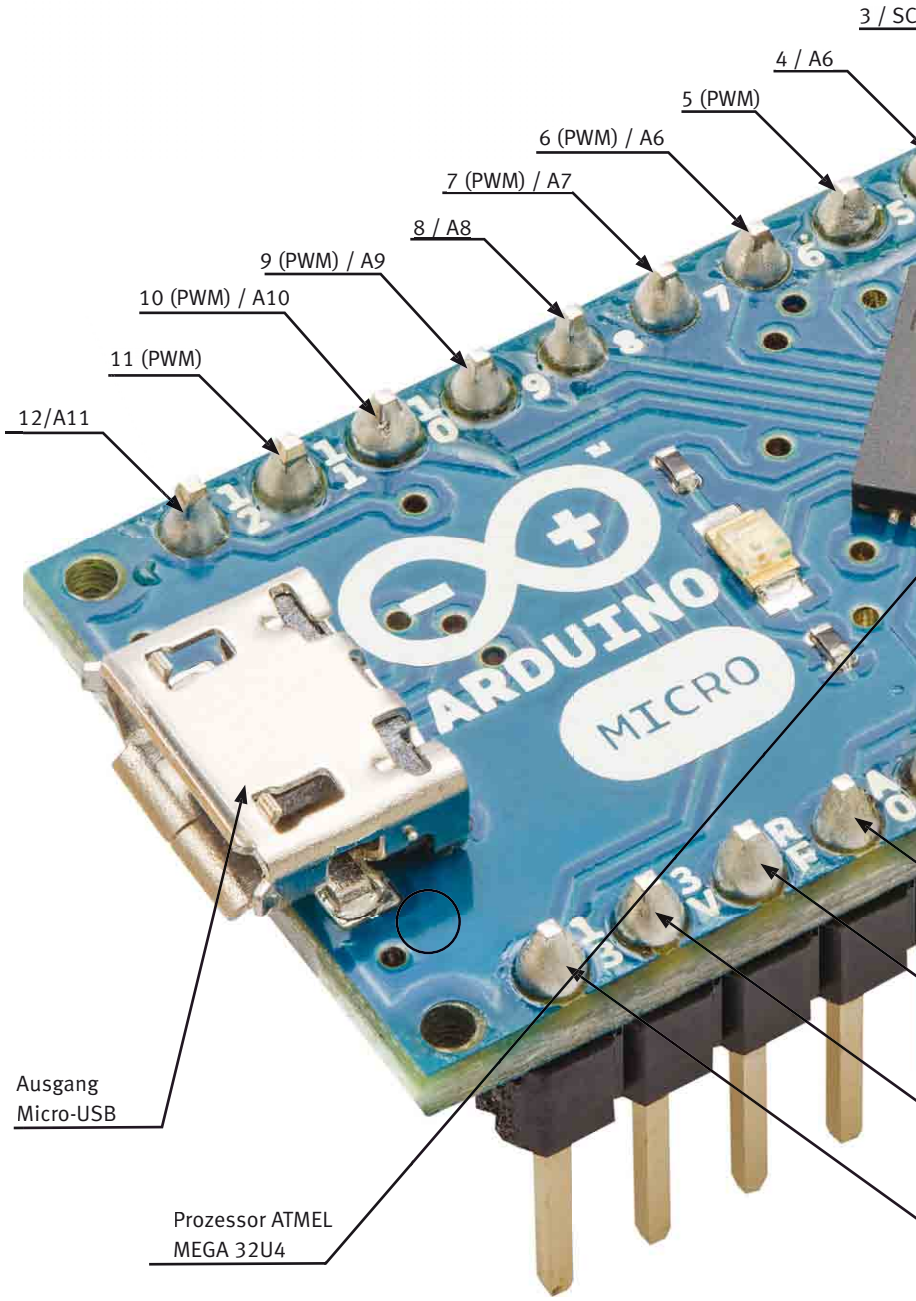
KAPITEL 1 bis 5

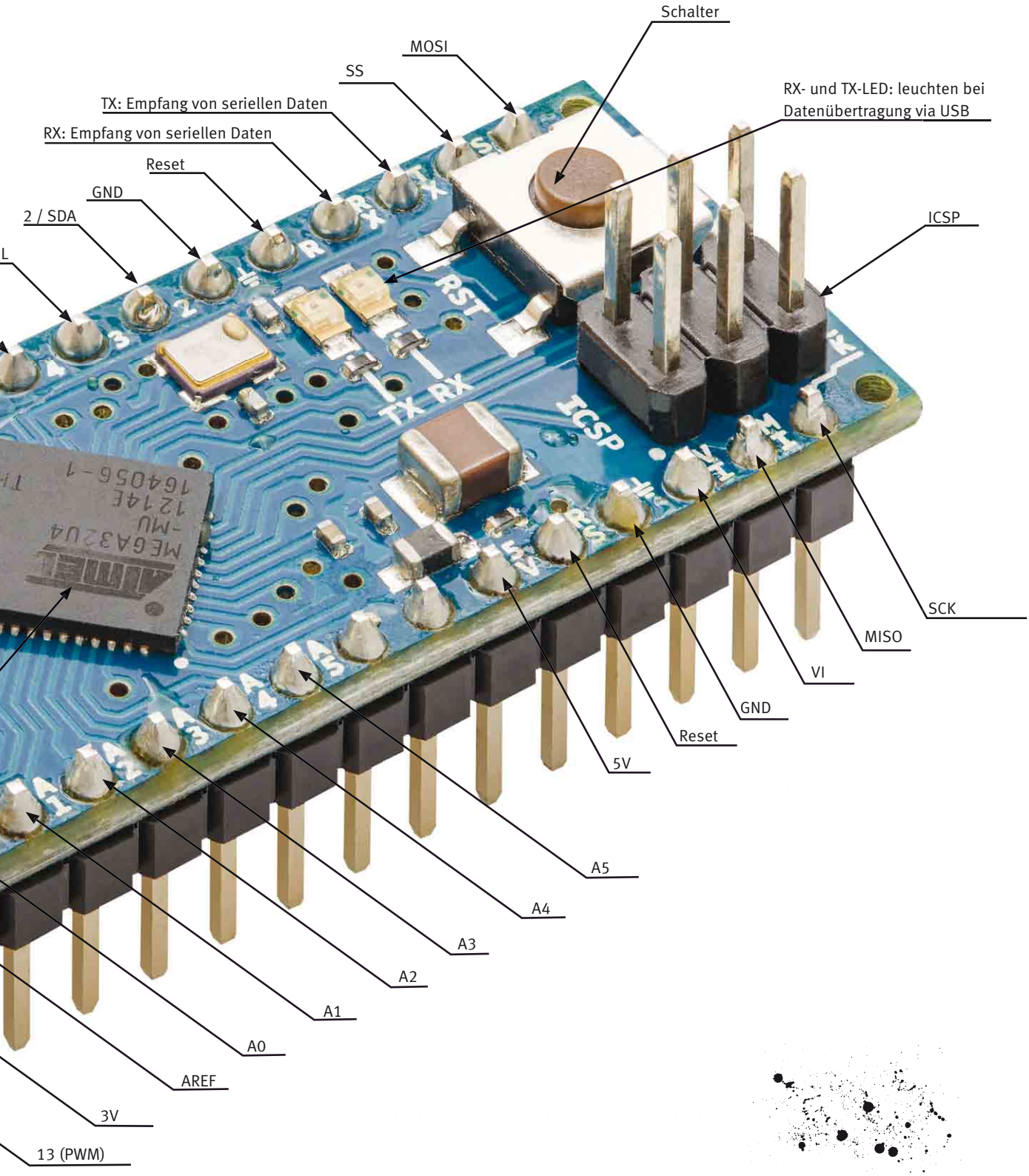
KLEINE PLATINE GANZ GROSS

Experten erfassen den Aufbau des Arduino Micro schnell. In seinem Aufbau unterscheidet er sich aber doch sehr von den anderen Arduino-Modellen. Deshalb lohnt sich der Blick auf die Details.



Gewicht:
6,6 Gramm




Schalter
MOSI
SS
TX: Empfang von seriellen Daten
RX: Empfang von seriellen Daten
Reset
GND
2 / SDA
RX- und TX-LED: leuchten bei Datenübertragung via USB
ICSP
SCK
MISO
VI
GND
Reset
5V
A5
A4
A3
A2
A1
A0
AREF
3V
13 (PWM)

1 Einführung

Der Erfolg des Arduino-Konzepts ist ungebrochen. Mehr und mehr Anwender erkennen das außergewöhnliche Potenzial dieser Mikrocontrollerplattform.

Neben der einfachen und kostengünstigen Hardware ist es vor allem auch die problemlos zu bedienende Programmieroberfläche, die den Arduino so erfolgreich werden ließ. Damit lassen sich auch außerhalb einer eng umrissenen Technikgemeinde eindrucksvolle Projekte realisieren.

Bereits Jugendliche kommen mit dem Konzept bestens zurecht. Sogar Kinder können mit Unterstützung eines Erwachsenen die ersten Schritte in das Reich der Elektronik und Mikrocontrollertechnik wagen. Genau hier setzt das vorliegende Buch an. Es werden einfache aber dennoch äußerst interessante Projekte vorgestellt, die ohne detailliertes Fachwissen nachgebaut werden können. Anstelle von Schaltbildern werden realitätsnahe Aufbaubilder verwendet.

Aber natürlich bleibt es nicht beim bloßen Zusammenbauen und Aufspielen von Programmen. In jedem Kapitel wird die Funktion des aufgebauten Geräts genau erläutert.

Zusätzlich wird ab Kapitel 5 auch auf die direkte Programmierung in C eingegangen. Nachdem das professionelle Programmiertool der Firma Atmel vorgestellt wurde, können auch schnelle und effiziente C-Programme für den Arduino Micro erstellt werden. Dadurch wird der Anwendungsbereich dieses kompakten Controllerboards auf die gesamte Welt des *Physical Computing* erweitert.

1.1 Arduino Micro – ein Controllersystem in IC-Größe

Der weltweite Siegeszug des Systems Arduino ist bislang einmalig. Neben seiner raschen Verbreitung in ganz Europa konnte das Board bald auch große Erfolge in den USA vermelden. Dort werden vor allem auch Shields in allen Variationen entwickelt und vertrieben. So haben beispielsweise US-Firmen wie SparkFun oder Seeedstudio die verschiedensten Hardwareerweiterungen, sogenannte »Shields«, im Programm.

Das Wachstum der Arduino-Familie ist weiterhin ungebrochen. Die Entstehung immer neuer Boardvarianten scheint sich sogar noch zu beschleunigen.

Der Arduino-Stammbaum hat seinen Anfang im Jahr 2005 im Ur-Arduino mit einem ATmega8-Controller und einer seriellen Schnittstelle. Wurde zunächst nur diese eine Version immer weiter entwickelt und mit dem Arduino NG und dem Duemilanove mit immer leistungsfähigeren Prozessoren und neuen Schnittstellen versehen, kamen im Jahr 2010 die ersten eigenständigen Äste hinzu.

Arduino Nano, Mega und das Lilypad wurden am Markt eingeführt. Schließlich kamen 2012 noch der Leonardo und der Arduino Due hinzu.

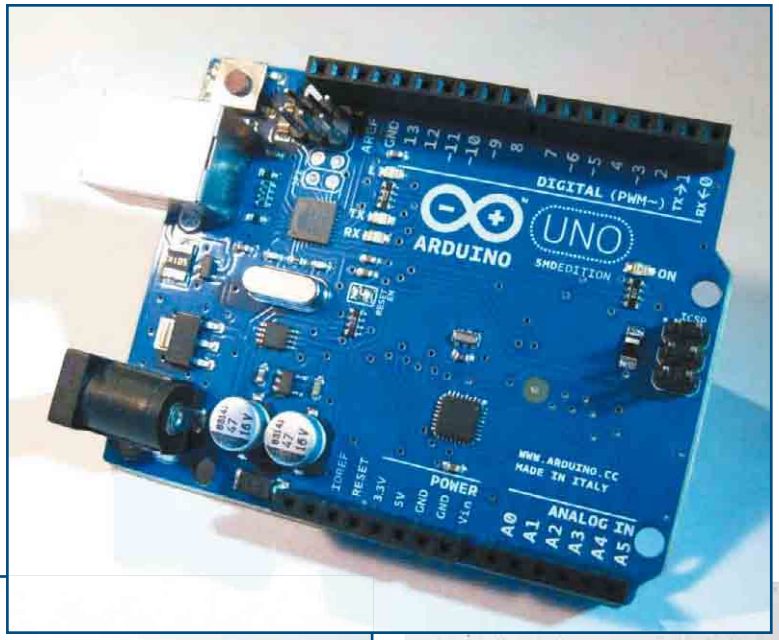


Abb. 1.1: Arduino UNO in der klassischen Bauform

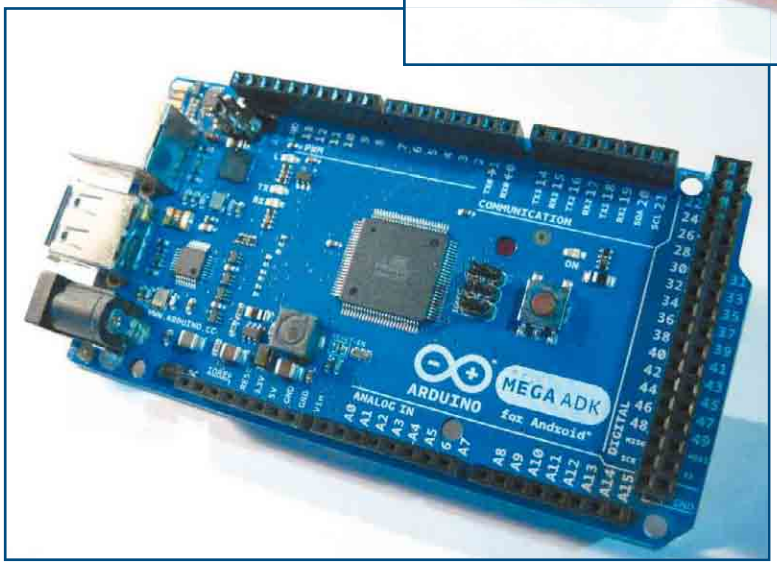


Abb. 1.2: Der Arduino Mega mit dem leistungsfähigen 2560-Prozessor

Im selben Jahr wurde auch der Arduino Micro vorgestellt. In vielerlei Hinsicht hat dieser Baustein das Spielzeugimage der anderen Boards abgelöst. Er hat einen modernen Controller an Board, der direkt über USB kommunizieren kann. Die relativ teuren RS-232-zu-USB-Umsetzer können damit entfallen, was sich auch im Preis des Micro positiv niederschlägt.

Darüber hinaus ist dieser Baustein für den Einsatz auf lötfreien Aufbausystemen bestens geeignet. Durch zwei parallele Pinreihen kann der Micro

Abb. 1.3: Arduino Micro im Vergleich mit einem klassischen IC

direkt in ein Breadboard eingesteckt werden. Spezielle und meist auch entsprechend hochpreisige Prototyping-Shields sind somit nicht erforderlich.

Abbildung 1.3 zeigt den Micro im Vergleich zu einem klassischen IC und verdeutlicht so seine äußerst kompakte Bauform.

1.2 Der Arduino Micro auf einen Blick

Die technischen Daten des Micro sind in Tab. 1.1 zusammengefasst.

Mikrocontrollertyp	ATmega32U4
Versorgungsspannung	5 V
Eingangsspannungsbereich (empfohlen)	7–12 V
Maximaler Eingangsspannungsbereich	6–20 V
Digitale I/O-Pins	20
PWM-Kanäle	7
Analogeingänge	12
Max. Strombelastung pro Pin	40 mA
Max. Strombelastung des 3,3-V-Ausgangs	50 mA
Flashspeicher	32 KB, davon 4 kB für den Bootloader
SRAM	2,5 KB
EEPROM	1 KB
Clock Speed	16 MHz

Tab. 1.1: Technische Daten des Arduino Micro

Weitere Einzelheiten dazu, etwa die Bedeutung der einzelnen Speicherarten etc., werden in späteren Kapiteln näher erläutert.

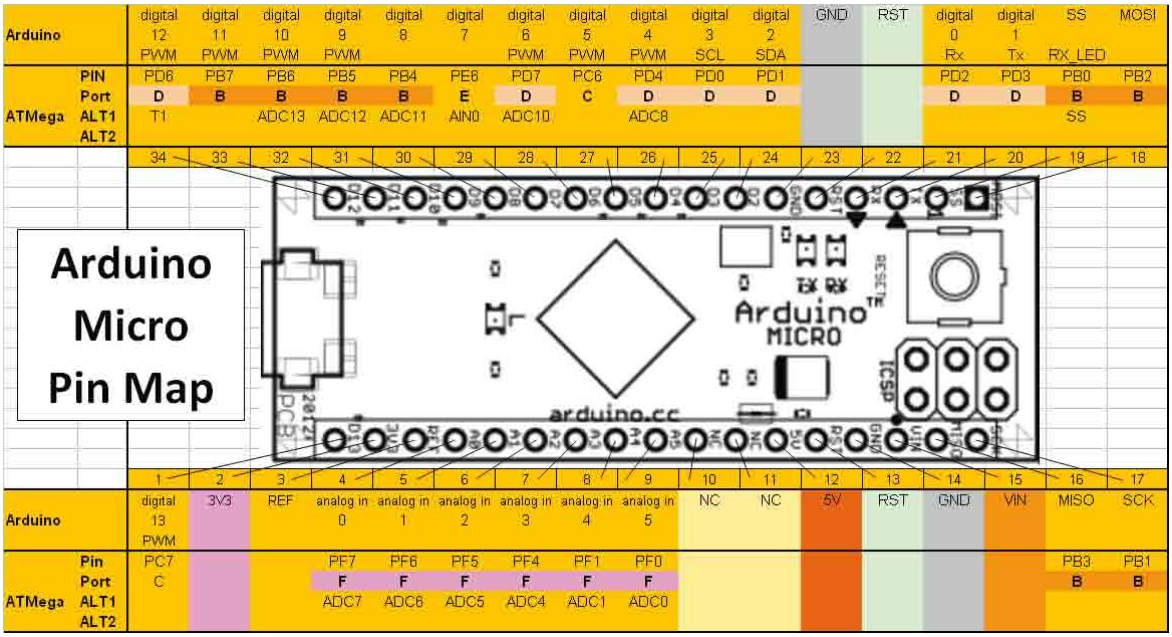


Abb. 1.4: Pinzuordung beim Arduino Micro

1.3 Elektronische Aufbausysteme

Will man eigene Schaltungen und Projekte entwickeln und testen, ist das Anfertigen von geätzten Leiterplatten mit erheblichem Aufwand verbunden. Nach dem Entwurf und dem Layout der Platine muss diese chemisch geätzt werden. Dann erst kann das Einlöten der Bauteile erfolgen. Soll die Schaltung nachträglich geändert oder verbessert werden, ist das nur mit erheblichem Aufwand und meist auch nur mit großen Einschränkungen möglich.

Wesentlich einfacher ist der Einsatz von sogenannten lötfreien Steckplatinen. Sie werden oft auch unter dem Namen *Breadboards* vertrieben. Der Name rührt daher, dass in den Anfangszeiten tatsächlich elektronische Prototypen auf hölzernen Brotbrettern aufgebaut wurden. Sie waren mit Lötnägeln bestückt. Die in früheren Zeiten oftmals noch recht voluminösen Komponenten wurden dann über diese Lötnägel verbunden.

Breadboards waren auch in den Entwicklungsabteilungen der Elektronikindustrie weitverbreitet. Es war durchaus üblich, auch kommerzielle Schaltungen oder Schaltungsteile zunächst auf einem Breadboard auszutesten. Leistungsfähige Simulationswerkzeuge sorgten aber dafür, dass der Einsatz der Breadboards deutlich zurückging. Neue Schaltungsideen werden

nun hauptsächlich virtuell am Rechner überprüft, reale Aufbauten, egal ob als Breadboardversion oder als Leiterplattenprototyp, bilden eher eine seltene Ausnahme. In der Elektronikausbildung und an Hochschulen sind Breadboard-Schaltungen aber immer noch häufig anzutreffen.

Im Hobbybereich werden sie sogar wieder öfter verwendet. Durch die Massenproduktion wurden die Breadboards sehr preiswert, sodass hier in den letzten Jahren der Einsatz wieder zugenommen hat. Im Umfeld der Arduino-Gemeinde haben sich lötfreie Steckbretter sogar hervorragend bewährt. Waren sie bei den klassischen Arduino-Versionen noch recht umständlich über Drähte mit den Buchsenleisten der Boards zu verbinden, kann man nun den Arduino Micro in geradezu idealer Weise in ein Breadboard einsetzen. Wackelkontakte, wie sie früher bei den umständlichen Drahtbrückenverbindungen häufig auftraten, gehören nun der Vergangenheit an.

Manchmal hört man das Argument, dass eine Breadboard-Schaltung über längere Zeit hinweg nicht zuverlässig funktioniert. Die Praxis aber beweist das Gegenteil. Wenn man beim Aufbau sorgfältig vorgeht, kann man auch eine Breadboard-Schaltung problemlos im Dauereinsatz betreiben. In Kapitel 1.4 werden einige Tipps und Ratschläge für erfolgreiche Breadboard-Aufbauten gegeben.

Natürlich gibt es Situationen, in denen eine Breadboard-Schaltung nicht zu empfehlen ist. In Fahrzeugen beispielsweise oder in anderen Umgebungen, bei denen mit stärkeren Vibrationen zu rechnen ist. Bei hohen Luftfeuchtigkeiten oder großen Temperaturschwankungen, wird ein Breadboard-Aufbau ebenfalls keine hohe Zuverlässigkeit erreichen, da die Kontaktfedern in diesem Fall leicht korrodieren.

In normalen Wohnräumen dagegen spricht nichts gegen den Dauereinsatz. Wenn die Schaltung in ein Gehäuse eingebaut wird und so vor Staub und mechanischen Einflüssen geschützt ist, kann eine Schaltung über Jahre hinweg zuverlässig arbeiten.

In Kapitel 12 des Buchs wird darauf eingegangen, wie man eine fertig entwickelte Schaltung dauerhaft auf einer Lochrasterplatine aufbauen kann. Der letzte Schritt wäre dann die Entwicklung einer geätzten Leiterplatte. Dieser Schritt lohnt sich allerdings meist nur, wenn bereits eine Kleinserie gefertigt werden soll. Bei Einzelstückzahlen ist es so gut wie immer günstiger, beim Lochrasteraufbau zu bleiben.

Experimentierboards gibt es in den unterschiedlichsten Ausführungen, Formen und Farben. Meist weisen sie ein Hauptsteckfeld auf, das aus zwei Reihen mit Metallfedern besteht. Die Metallfedern haben jeweils 5 Öffnungen, die zur Aufnahme von Bauteildrähten dienen. Bauteilanschlüsse, die

in eines dieser 5 Aufnahmelöcher gesteckt werden, sind leitend miteinander verbunden.

Daneben weisen viele Boards noch zwei sogenannte Busschienen auf. Hier gibt es mehrere Varianten. Entweder sind die Schienen über die gesamte Länge des Steckboards miteinander verbunden oder sie sind in zwei elektrisch getrennte Teilschienen aufgeteilt.

Kleinere Experimentierboards verfügen oft nur über jeweils eine einzelne Schiene pro Boardseite. Bei größeren Steckboards ist häufig eine Doppelschiene an jeder Seite vorhanden. Oft können die Schienen auch vom Board abgetrennt werden. Meist dienen die Busschienen der Stromversorgung. Im Bedarfsfall können sie aber auch zur Signalführung verwendet werden.

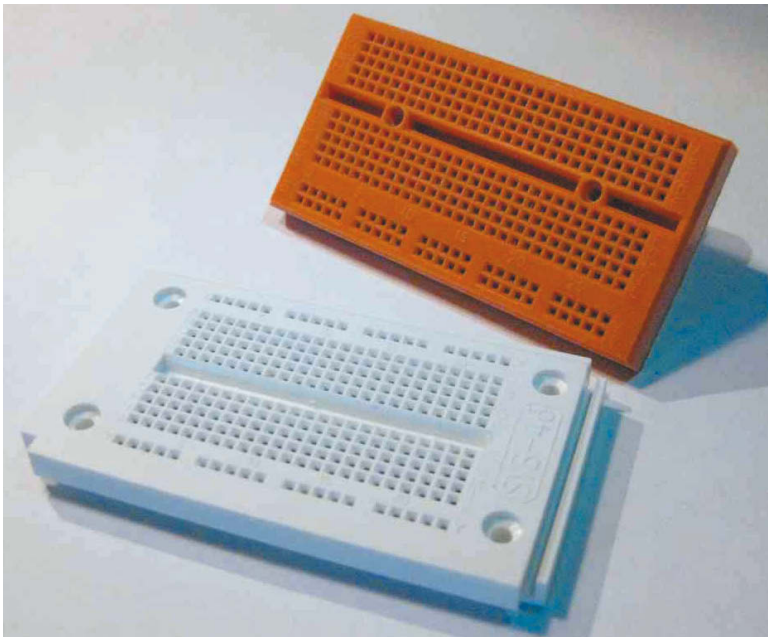


Abb. 1.5: Breadboards in verschiedenen Ausführungen

1.4 Breadboard-Betrieb des Arduino Micro

Der Arduino Micro ist ideal für den Betrieb in einem Breadboard geeignet. Allerdings sollte man einige Vorsichtsmaßnahmen berücksichtigen, wenn man dauerhaft erfolgreich mit dem Micro arbeiten will.

Insbesondere wenn die Steckplatinen noch neu sind, kann das Einstecken der Anschlusspins einigen Kraftaufwand erfordern. In diesem Fall kann man die Buchsen des Breadboards mit einer Stecknadel etwas aufweiten. Allerdings sollte man das nicht übertreiben, da es sonst zu Wackelkontakten kommen kann.

Beim Einsetzen eines Arduino Micro in ein neues Breadboard sollte man behutsam vorgehen, da es bei Anwendung zu großer Kräfte leicht zu Beschädigungen am Board kommen kann. So entstehen unter Umständen Haarrisse, die im Lauf der Zeit zu Kontaktproblemen an den Pins führen können.

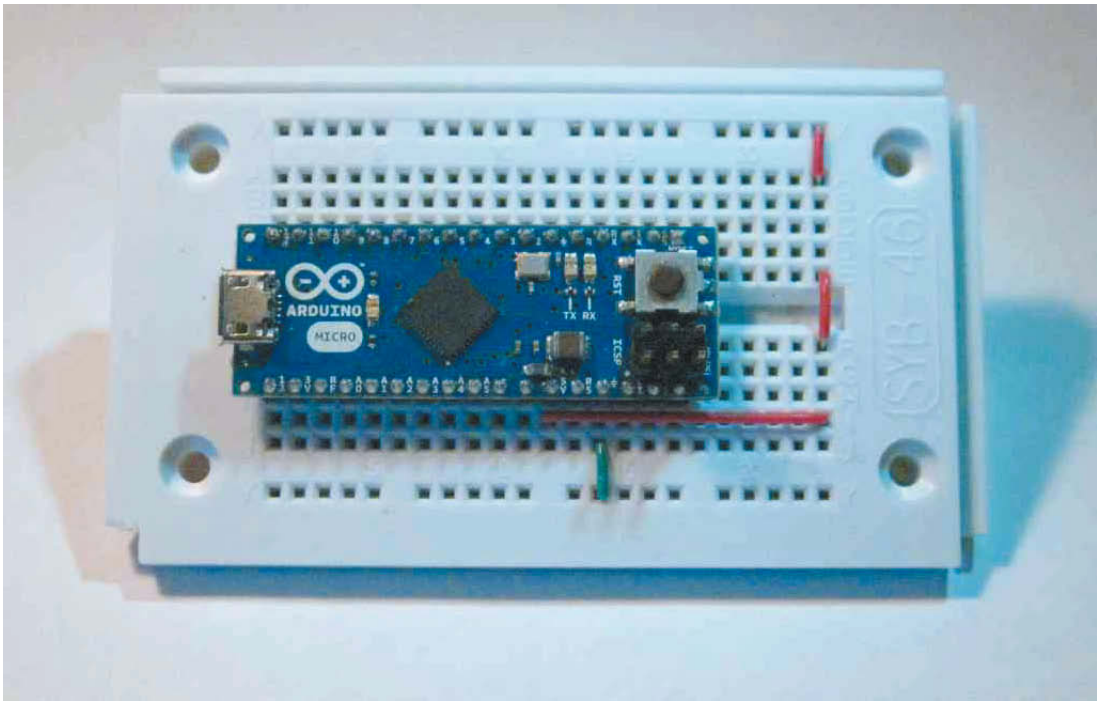


Abb. 1.6: Arduino Micro in einem Breadboard

Auch beim Entfernen der Arduino-Platine aus einem Breadboard ist Vorsicht geboten. Unsachgemäßes Vorgehen kann auch hier wieder zu Schäden am Arduino führen. Am besten geht man so vor wie in der folgenden Abbildung gezeigt. Mir einem kleinen Schraubenzieher hebt man den Arduino zunächst an einer Seite leicht an. Dann zieht man auf der anderen Seite nach. Wenn die Anschlusspins weit genug aus den Öffnungen des Breadboards herausgezogen sind, kann man den Micro ohne großen Kraftaufwand vollständig aus dem Breadboard ziehen.

Mattias Schlenker

Sensoren am Arduino™

Hören, Sehen, Fühlen, Riechen: Zeigen Sie dem Arduino in über 20 Projekten mit analogen und digitalen Sensoren die Welt

Vorwort

Nirgendwo in der EDV wurde in den letzten zehn Jahren das Prinzip *Eingabe-Verarbeitung-Ausgabe* (EVA) so deutlich wie bei der Mikrocontroller-Plattform Arduino. Kaum eine Woche verging, in der nicht irgendein Bastler Musikinstrumente baute, die auf Bewegung reagierten, seine Zimmerpflanzen Twitter-Meldungen verschicken ließ oder neue Kunstwerke zur Interaktion einluden. Besonders offensichtlich ist die Eingabeseite: Das gern zitierte »Physical Computing« bezieht Sensoren verschiedenster Art mit ein: Schalter, Temperatursensoren, Annäherungssensoren, GPS-Empfänger.

Möglich macht es die Vielzahl an Schnittstellen: Digitale Eingänge reagieren auf Schalter, die beiden Bus-Systeme SPI und I^2C erlauben die Anbindung von (bezahlbaren) Sensoren in Industriequalität — doch der Clou ist eine Reihe analoger Eingänge, die es erlauben, den Arduino selbst zum Sensor zu machen. Oder hätten Sie gedacht, dass es möglich ist, aus Alufolie, Kupferdraht, Widerständen, Kondensatoren und ganz viel Gehirnschmalz einen berührungslosen Sensor für die Füllmenge von Flüssigkeiten zu zaubern?

Arduino lebt nicht nur von genial einfacher Hardware für kleines Geld, sondern vor allem von einer großen Community aus Entwicklern, die Bibliotheken für viele Einsatzzwecke bereitstellen. Das alles zusammengenommen, führt zu schnellen Ergebnissen, für die nicht einmal viel Geld investiert werden muss — hätten Sie geglaubt, dass es möglich sein würde, mit Hardware für rund 10 Euro in einer guten Stunde einen kleinen Webserver zusammenzulöten, der Temperatur und Luftfeuchte anzeigt und dessen Programmierung keine 50 Zeilen Code braucht? Ich ehrlich gesagt auch nicht, bis ich die entsprechenden Optimierungen für dieses Buch vorgenommen habe.

Wer sind Sie? Vielleicht sind Sie ein Student oder wissenschaftlicher Mitarbeiter, der im Rahmen eines Forschungsprojekts Dutzende oder Hunderte kleine Sensoren lange Zeit in freier Wildbahn aussetzen muss — die müssen dann billig und zuverlässig sein. Oder Sie sind Eigenheimbesitzer, der seine Heizungs- und Lüftungssteuerung optimieren möchte — das soll einfach sein und flexibel in der Anwendung. Vielleicht möchten Sie das schrägste Musikinstrument der letzten 100 Jahre erfinden und suchen dafür nach Inspirationen? Möglicherweise wollen Sie auch einfach nur einen tollen Mausersatz für den heimischen PC bauen. Ich weiß es nicht.

Ich weiß, dass Sie neugierig sind (sonst hätten Sie nicht bis hierher gelesen), und ich mutmaße, dass Sie leicht umsetzbare Rezepte für einfache Sensoren suchen. Ich weiß jedoch nicht, wie viel Erfahrung mit Arduino Sie haben. Ich nehme an, dass Sie bereits Kontakt hatten und wissen, wie man die aufgelötete LED auf dem Arduino Uno zum Blinken bringt. Mehr müssen Sie nicht können. Wenn Sie noch gar keinen Kontakt hatten, ist das auch kein Problem — legen Sie los, Sie können nicht viel kaputt machen.

Ich kann in diesem Buch zwar keinen kompletten Arduino-Einstieg bieten, aber ich verweise an den entsprechenden Stellen auf die passenden Tutorials.

In diesem Sinne: Lassen Sie uns gemeinsam loslegen: Keine Elektronikplattform wäre in Summe besser für die kreative Eingabeverarbeitung geeignet als der Arduino: Schlafmodi helfen beim Energiesparen, und die Vielzahl der Pins erleichtert den Bau kombinierter Sensoren. Und auch wenn die Betonung auf der Sensorseite liegt, dürfen Sie einige komplette Lösungen erwarten, die Verarbeitung und Ausgabe einbeziehen. So lernen Sie Sensoren kennen, die direkt auf SD-Karte schreiben, ihre Werte übers Netzwerk übertragen oder in den Äther funken. Wir lassen also auch die Verarbeitungs- und Ausgabeseite nicht zu kurz kommen und wünschen Ihnen viel Spaß — und noch viel mehr Anregungen für eigene Ideen — mit diesem Buch.

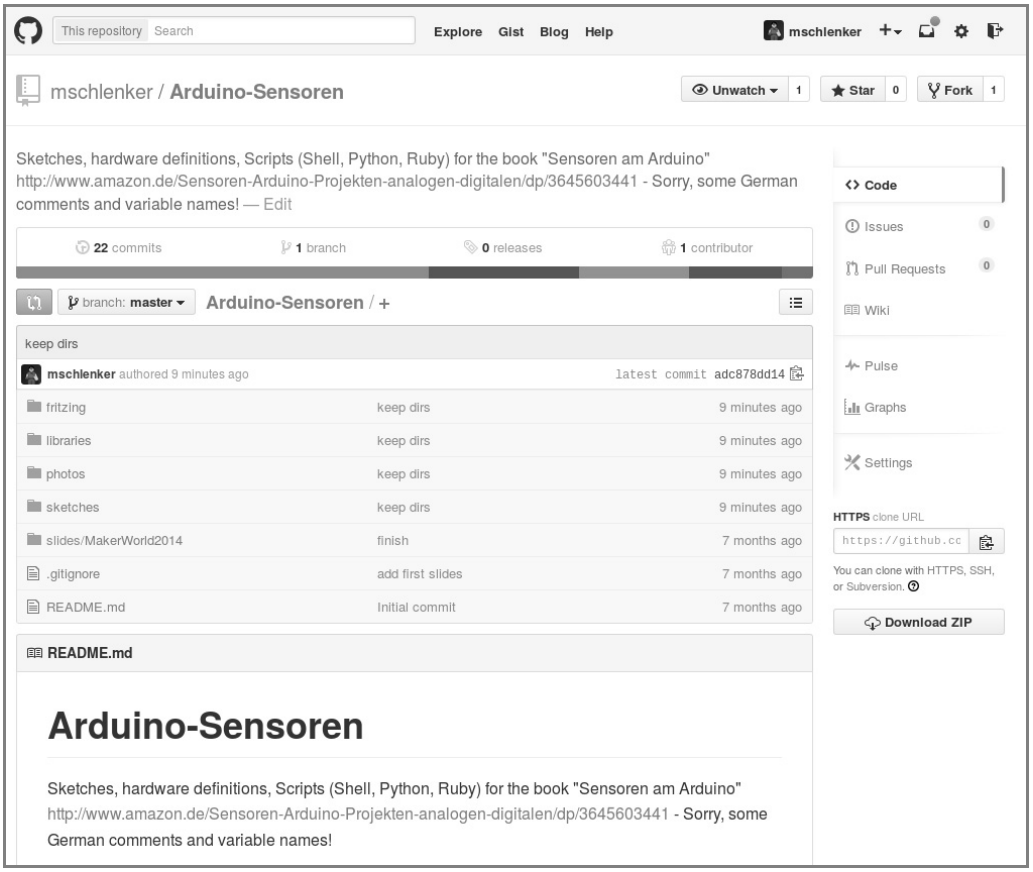
In diesem Sinne: Lassen Sie uns losmessen!

Konventionen im Buch

Jeder Autor macht seine eigenen Erfahrungen, und daraus sind Vorlieben für seine Arbeit an Projekten entstanden. Diese sind mit der Zeit einem Wandel unterworfen, was es umso wichtiger macht, die in diesem Buch verwendeten Konventionen etwas zu erläutern. Ich möchte mir keinesfalls anmaßen, dass meine Arbeitsweise die einzig wahre ist. Wenn Sie bereits Arduino-Erfahrung haben, werden Sie für viele Vorgehensweisen eigene »Best Practices« entwickelt haben, die genauso gut funktionieren (wenn Ihre Vorgehensweisen besser funktionieren, bin ich für einen Hinweis per E-Mail dankbar). Sollten Sie wenig Routine haben, betrachten Sie meine Erfahrung einfach als ganz bewährten Startpunkt.

Downloads, Updates und Errata

Die Codebeispiele in diesem Buch versuche ich, so kurz zu halten, dass Sie sie leicht abtippen können. Das gelingt nicht immer, und gelegentlich kommt es vor, dass ein Sketch nach längerer Laufzeit Probleme offenbart, die nachträglich korrigiert werden müssen. Um Ihnen den Einstieg zu erleichtern, finden Sie alle Bilder, Codebeispiele und Verdrahtungspläne auf meiner GitHub-Seite *https://github.com/mschlenker/Arduino-Sensoren*. Dort können Sie Einzeldateien herunterladen oder rechts mit dem Button *Download ZIP* ein Archiv aller Dateien herunterladen. Einen vollständigen Download finden Sie auch auf den Seiten des Franzis Verlags unter *www.buch.cd*.



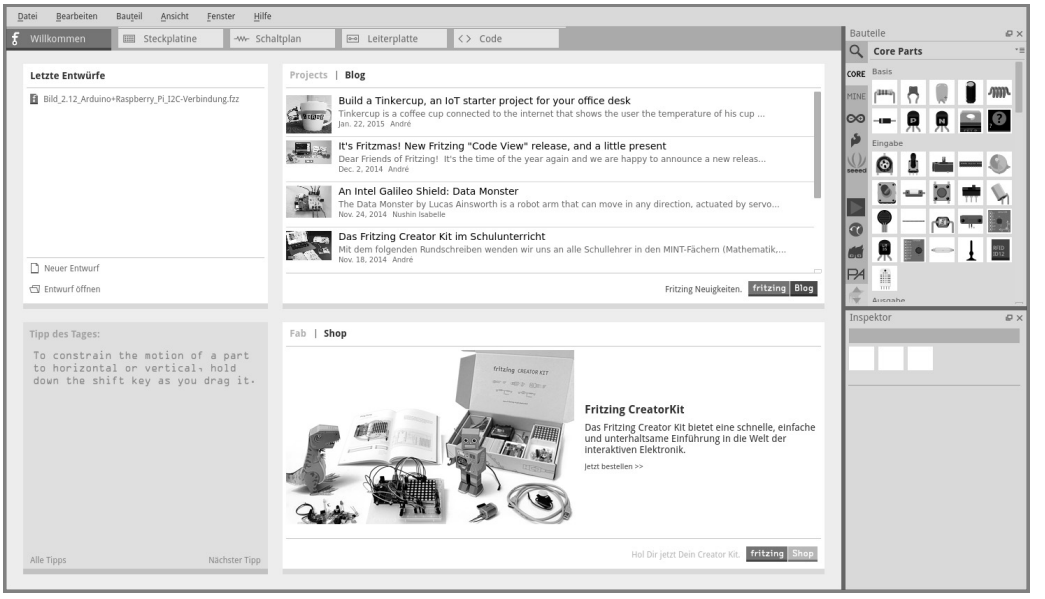
Ein Blick ins GitHub-Repository lohnt sich, hier können Sie die aktuellen Quellcodes der Sketches im Buch herunterladen und finden hochauflösende, farbige Versionen aller Fotos.

Prinzipiell rate ich jedem, der Projekte in Angriff nimmt, die Software (beispielsweise in Form von Arduino-Sketches) oder andere Dokumentationen (Verdrahtungs- und Schaltpläne) involviert, sich mit Versions- und Revisionskontrollsystemen vertraut zu machen. Das oft gehörte »Gestern hat es doch noch funktioniert!« verliert nämlich schnell seinen Schrecken, wenn man alle Änderungen der letzten Tage zeilenweise nachvollziehen kann.

Der schnellste Einstieg in Git für Windows-Anwender dürfte der Client des Repository-Anbieters *github.com* sein. Selbstverständlich müssen Sie sich nicht von diesem Anbieter abhängig machen: Installieren Sie einen Git-Client lokal, »pullen« Sie meine Repositories und legen Sie eigene Repositories für eigene Zwecke an — aber vergessen Sie bitte das Backup nicht!

Bilder und Verdrahtungspläne

Wie andere Bücher dieser Reihe ist auch dieses Buch in Graustufen gedruckt. Ich versuche immer, die Verdrahtungsskizzen so zu zeichnen, dass sie auch ohne Farbe leicht nachvollziehbar sind. In vielen Fällen werden Sie aber die Schaltungen mehrerer Beispiele kombinieren wollen, was es erforderlich macht, die Bauelemente anders anzuordnen. Die für dieses Buch anfertigten Breadboard-Ansichten und Schaltpläne wurden mit der freien (und kostenlosen) Software Fritzing erstellt, Sie können sie für Linux, Windows und Mac OS X unter *www.fritzing.org* herunterladen.



Die freie Software Fritzing hilft beim Erstellen von Schaltplänen, besonders hilfreich ist die Steckplatinenansicht.

Die von mir erstellten FZZ-Dateien erhalten Sie unter den oben genannten Adressen (*https://github.com/mschlenker*, *www.buch.cd*). Rekombinieren Sie Bauelemente nach Belieben, fügen Sie Bauelemente hinzu oder kombinieren Sie Schaltungen. Außerdem können Sie Fritzing dafür verwenden, eigene Shields für Arduino Uno, Leonardo oder Pro Mini zu entwerfen. Diese lässt Fritzing ab Losgröße eins von einem Berliner PCB-Dienstleister wöchentlich belichten.

Inklusive Zersägen der großen Platine und Versand vergehen so im Regelfall maximal 14 Tage von der Aufgabe der Bestellung bis zum Eingang der Platine. Für die Weitergabe modifizierter Fritzing-Dateien auf Basis unserer Entwürfe gilt die Lizenz CC BY-SA 3.0. Auf Deutsch: Toben Sie sich aus, spinnen Sie unsere Ideen weiter, aber bitte nennen Sie die Quelle und weisen Sie auf Weitergabe unter gleichen Bedingungen hin.

Projekte, die im Zuge der Umsetzung dieses Buchs nicht fertiggestellt werden konnten, die Dauerbaustellen des Autors sind oder die im Nachhinein von Lesern vorgeschlagen wurden, finden Sie in meinem Arduino-Blog *www.arduino-hausautomation.de*. Wenn Sie selbst etwas vorschlagen wollen, veröffentliche ich dies gern, solange Fritzing-Dateien und Sketches unter CC BY-SA 3.0 oder GPL v2 vorliegen. Um Kontakt mit mir aufzunehmen, verwenden Sie bitte die im Impressum angegebene E-Mail-Adresse.

Projekte oder Ideen, die es nicht ins Buch geschafft haben oder die von Lesern vorgeschlagen wurden, finden Sie im Blog *www.arduino-hausautomation.de*.

Arduino-IDE

Zur Entwicklung von Arduino-Sketches beziehen wir uns in diesem Buch auf die Arduino-Entwicklungsumgebung, die unter *http://arduino.cc/en/Main/Software* erhältlich ist. Andere Methoden der Softwareentwicklung sprechen wir nicht an. Falls Sie bislang keine Erfahrung mit Arduino haben: Besorgen Sie sich einen Arduino Uno und investieren Sie zwei Stunden in das Tutorial *http://arduino.cc/de/Guide/HomePage* — es zeigt, wie Sie den Arduino anschließen, Programme erstellen und hochladen und über den seriellen Port eine Kommunikation zwischen Arduino und PC stattfinden lassen.

Weiterführende Details wie das Auswerten analoger und digitaler Pins und den Sinn von Pull-up- und Pull-down-Widerständen greift dieses Buch auf. Außen vor bleibt zwar nicht die Ausgabeseite (ich behandle Speicherung, Datenübertragung per Funk und Ausgabe auf LC-Displays), aber die Ansteuerung von Aktoren. Falls Sie die gemessene Drehzahl eines Elektromotors über ein Servo auf einer Zeigerskala ausgeben wollen, müssen Sie die entsprechenden Tutorials in anderen Büchern oder im Internet suchen.

Bei Redaktionsschluss war Arduino 1.6 soeben freigegeben worden. Da bereits 1.5 Beta nach meiner Erfahrung für die klassischen Arduinos (Uno, Leonardo etc.) keine Stabilitätsprobleme aufwies und Änderungen bei den APIs die eine oder andere Anpassung notwendig machen, empfehle ich Version 1.6 und beziehe mich auch im weiteren Verlauf des Buchs auf diese. So gehe ich stillschweigend davon aus, dass Libraries per Archivauswahl installiert werden können und keinen Neustart der Entwicklungsumgebung benötigen — dies ist bei Version 1.0 noch der Fall.

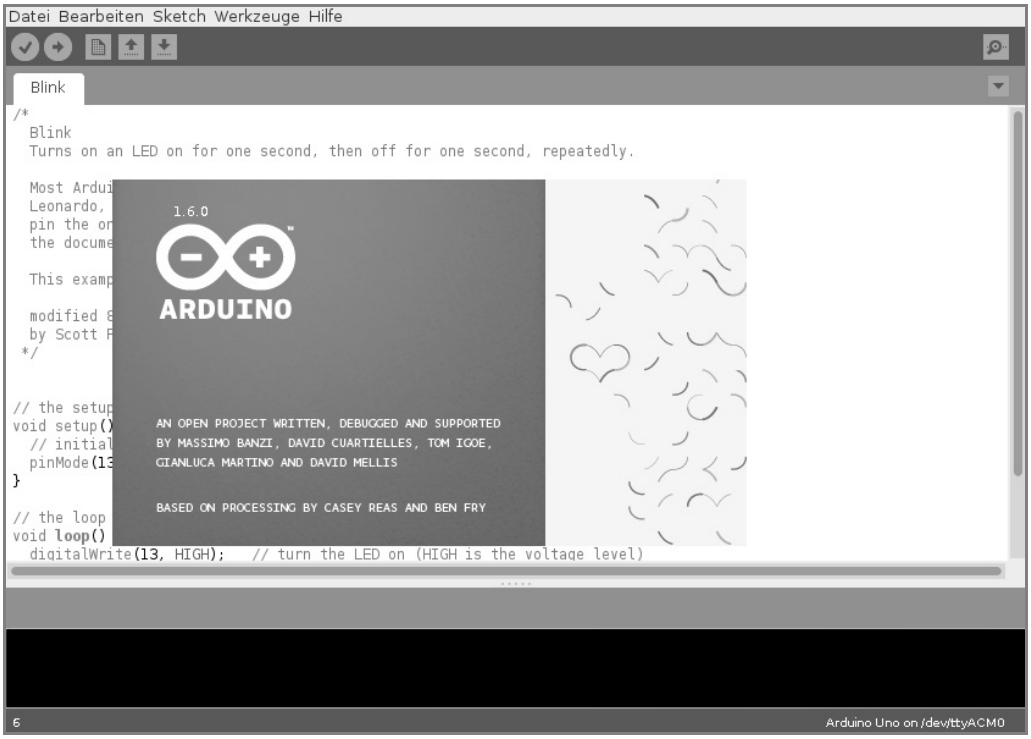
Auf allen Betriebssystemen ist eine Parallelinstallation möglich. Unter Linux-Distributionen wie Ubuntu 14.04 LTS, die Version 1.0 als Default mitbringen, sollten Sie zunächst die vom Distributor mitgelieferte Version installieren und diese einmal starten. Testen Sie mit dem Blink-Beispiel, ob Zugriff auf per USB angeschlossene Arduinos möglich ist. Möglicherweise bietet die IDE an, eine neue Gruppe für den Gerätezugriff anzulegen und den aktuell angemeldeten Benutzer dieser Gruppe hinzuzufügen, was Abmelden und erneutes Anmelden erfordert, damit die neue Zuordnung aktiv wird. Entpacken Sie dann das TAR-Archiv mit Root-Rechten in `/opt` oder `/usr/local`.

```
001 sudo tar -C /opt -xvzf arduino-1.6.x-linux64.tgz
```

Um die Arduino-IDE nicht mit vollem Pfad aufrufen zu müssen, erstellen Sie anschließend einen Softlink:

```
001 sudo ln -sf /opt/arduino-1.6.x/arduino /usr/bin/arduino16
```

Die IDE steht nun über den Befehl `arduino16` zur Verfügung. Ist ein Starter erwünscht, kopieren Sie `/usr/share/applications/arduino.desktop` ins Heimatverzeichnis oder auf den Desktop und ändern die Datei mit dem Starter entsprechend ab.



Optisch kaum ein Unterschied zu Arduino 1.0, technisch weit besser auf neue Arduinos und Erweiterbarkeit zugeschnitten: Im Buch beziehe ich mich immer auf die Entwicklungsumgebung Arduino 1.6.

Sketches und Programme

Im Buch abgedruckte Sketches halte ich so weit wie möglich einfach und lesbar. Dabei bleibt momentan die eine oder andere Optimierung auf der Strecke — so nutze ich aus Gründen der Lesbarkeit nicht immer bitweise Verschiebungen, wenn eine Division durch eine Zweierpotenz erfolgt. Die Sketches, die Sie unter den oben genannten Adressen herunterladen können, enthalten in zusätzlichen Kommentaren Hinweise zu Optimierungsmöglichkeiten.

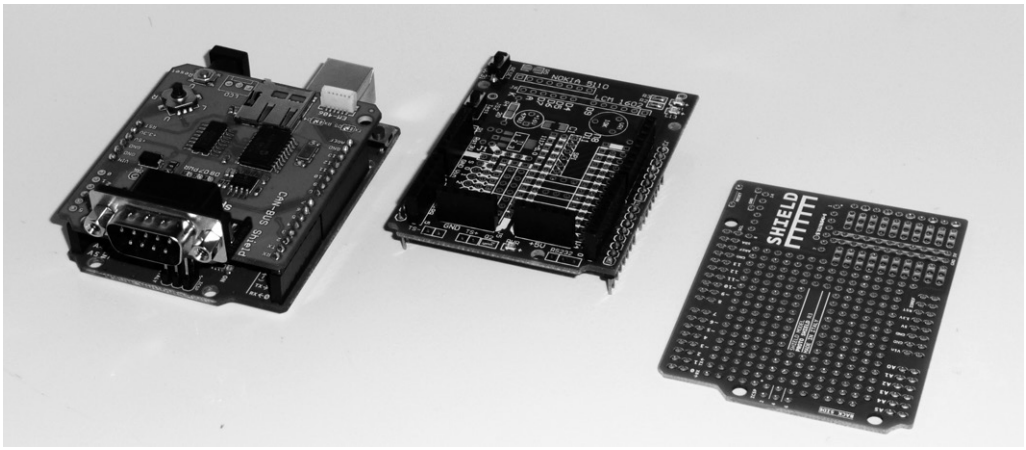
Ein häufiges Problem bei der Arduino-Programmierung ist die Verwendung von Bibliotheken: Arduino bringt eine Reihe von auf den gängigsten Arduinos getesteten Bibliotheken für einige auf offiziellen Shields und auf solchen von Unternehmen aus dem »engen Kreis« (Sparkfun, Adafruit) verwendeten Chips mit. Doch oft handelt es sich dabei um Hardware, die ihren Zenit bereits vor fünf Jahren überschritten hat. So sind mittlerweile günstige RTCs (*Real Time Clocks* — Echtzeituhren) erhältlich, die einen Temperatursensor haben, der zur Korrektur verwendet wird. Die ideale Bibliothek für diese Uhren liest nicht nur die Uhrzeit, sondern auch die Temperatur aus.

Ethernet-Module mit Empfangspuffer kosten mittlerweile weniger als ein Fünftel der »offiziellen« Ethernet-Shields. So liegt es in vielen Fällen nahe, sie zu verwenden. Dafür müssen häufig Bibliotheken genutzt werden, die nicht Teil der offiziellen Entwicklungsumgebung und in einigen Fällen einem rapiden Wandel unterworfen sind oder von verschiedenen Entwicklern den unterschiedlichen Schwerpunkten folgend in eigenen GitHub-Repositories betreut werden. Damit Sie die im Buch beschriebenen Beispiele umsetzen können, stellen wir die verwendeten Versionen als direkt installierbares Zip-Paket bereit.

Sie sollten dennoch die GitHub-Seiten der erwähnten Entwickler besuchen: Die Chancen stehen gut, dass neuere Versionen der Bibliotheken moderne Arduinos besser unterstützen, lang ersehnte Features hinzufügen oder einfach Optimierungen hinsichtlich Speicherverwaltung oder Last vornehmen. Zudem bietet GitHub die Möglichkeit, Fehlermeldungen und Optimierungsvorschläge direkt an die jeweiligen Entwickler zu richten. An dieser Stelle sei nochmals darauf hingewiesen, dass Versions- und Revisionskontrolle eine sehr praktische Erfindung ist und es sich insbesondere dann lohnt, sich mit Git und den verwendeten Tools vertraut zu machen, wenn gemeinsam an Projekten gearbeitet wird.

Shields bauen

Die einfache Erweiterbarkeit von Arduino-Platinen durch sogenannte Shields ist ein großer Pluspunkt der Plattform. Mit dem 2007 finalisierten Design entstand schnell eine Reihe fertiger »Shields« zum Aufstecken auf den Arduino. Der Kauf fertiger Shields für Uno & Co. ist bei sehr kleinen Stückzahlen interessant, wenn Platz und Energieaufnahme eine untergeordnete Rolle spielen und die verwendeten Bauteile nur als SMD-Versionen erhältlich sind. So habe ich einige fertige CAN-Bus-Shields von Sparkfun, die als Logger für den OBD-II-Port dienen und Daten direkt auf die dort einsteckbare µSD-Karte schreiben.

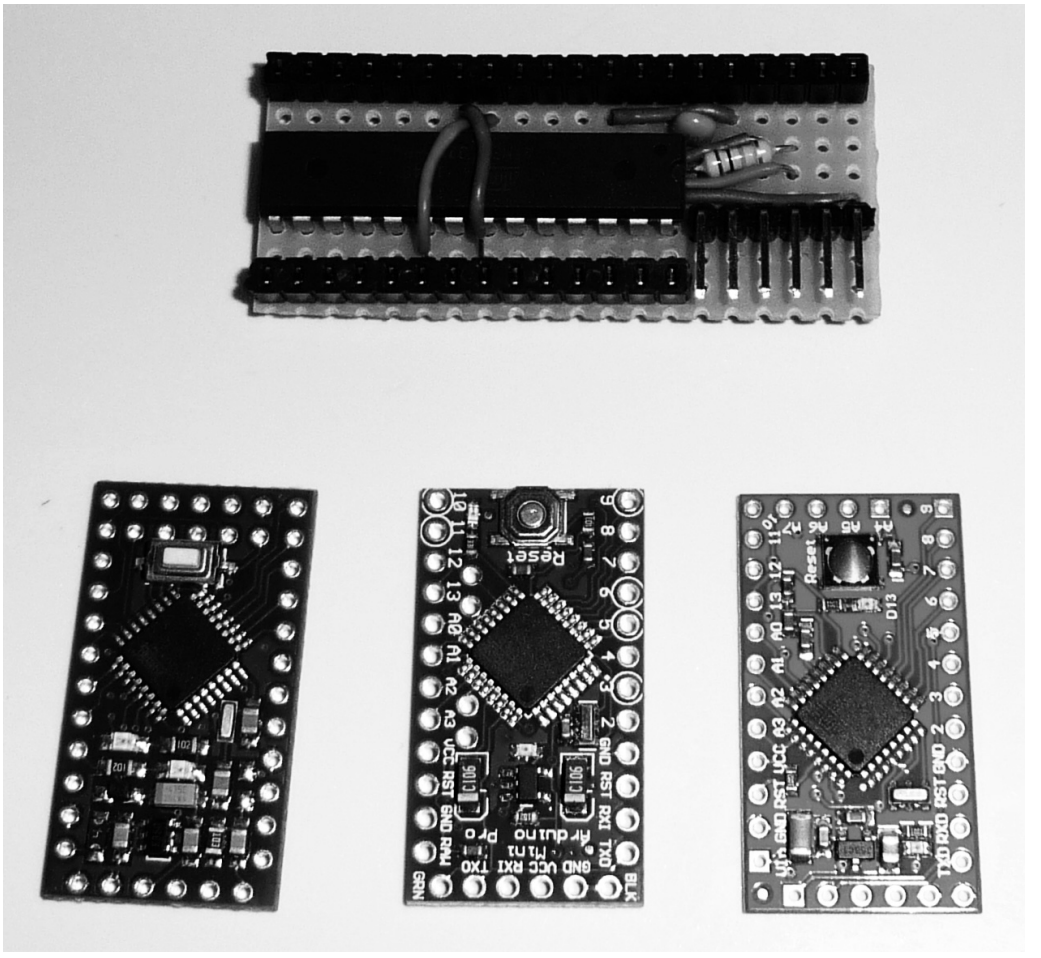


Drei Shields für Arduino Uno & Co.: Links ein fertiges CAN-Bus-Shield von Sparkfun, in der Mitte ein günstiges chinesisches Prototypen-Shield (das auch ein Mini-Breadboard aufnehmen kann) und rechts ein Original-Arduino-Proto-Shield.

Beim Selbstbau von Shields ist darauf zu achten, dass die Buchsenleiste für die digitalen Pins 8 bis 13 aus dem Rastermaß von 2,54 mm ausbricht — sie ist genau um 1,27 mm versetzt. Berücksichtigt man das bereits beim Aufbau einer Schaltung, verschmäht die versetzten Pins und zweckentfremdet stattdessen analoge Input-Pins als digitale Input- oder Output-Pins, können eigene Shields auf einer billigen Hartpapier-Loch- oder -Streifenrasterplatine angefertigt werden.

Werden alle Ein- und Ausgänge oder die SPI-Pins 11 bis 13 benötigt, können Sie zu einem der originalen »Proto-Shields« greifen, die zu Preisen um die 8 Euro erhältlich sind und den Komfort durchgeöster Lötaugen und Stapelbarkeit bieten. Alternativ zu den originalen Proto-Shields sind zu Preisen ab 3 Euro bereits pfiffige chinesische Kopien auf dem Markt, die das Konzept weiterspinnen und eigene Schienen für Masse und Versorgungsspannung oder Felder für oft benutzte Komponenten mitbringen.

Noch einfacher wird es, wenn Sie einen Arduino Pro Mini oder den später beschriebenen selbst gebauten »Rasterduino« verwenden. Der Pro Mini bleibt komplett — mit allen Pins — im Rastermaß von 2,54 mm, beim Rasterduino haben Sie zudem die Wahl, eine etwas größere Platine zu verwenden und die benötigten Bauteile gleich mit auf den Selbstbau-Arduino zu löten.



Vier »Arduino-Derivate«, die im Rastermaß von 2,54 mm bleiben: vorne drei Pro-Mini-Derivate aus China und Deutschland (3 bis 10 Euro), hinten ein selbst gelöteter Rasterduino (3 bis 5 Euro für die Komponenten).

Spannungsängste vermeiden

Ein auf den ersten Blick großes Ärgernis sind Betriebsspannungen: Sie werden den Großteil der Bauteile auf Spannungsbereiche von 3 bis 7 Volt spezifiziert vorfinden, aber es gibt Bauteile, die auf 5 bis 12 Volt ausgelegt sind, und andere, die 3,3 Volt verlangen. Arduino Pro Mini und Rasterduino werden immer mit entweder 3,3 Volt oder 5 Volt betrieben — beziehungsweise dem nächstliegenden Wert, den die Spannungsquelle (Batteriepack, Akku, Solarzelle …) bereitstellt. Das bedeutet im Idealfall, dass die Spannung des Arduino auf die am wenigsten flexible Peripherie ausgelegt werden kann.

Können nicht alle Komponenten in einem gemeinsamen Spannungsbereich betrieben werden, müssen Sie eine Spannungsteilerschaltung aufbauen, um die Versorgungsspannung bereitzustellen, und für IO-Pins einen Level-Shifter oder ebenfalls einen Spannungsteiler verwenden. In der Praxis gelten viele 3,3-Volt-Peripheriegeräte jedoch als »5-Volt-sicher«, was bedeutet, dass zumindest der Testbetrieb über einige Stunden hinweg am PC mit 5 Volt möglich ist. So haben wir — ohne dass Schäden aufgetreten wären — mehrere µSD-Karten für einige Minuten mit einem am PC angeschlossenen Pro Mini getestet, bevor die fertigen Sensorknoten auf Batteriebetrieb (2x Alkaline, also 3,0 Volt) in den »Regelbetrieb« übernommen wurden.

Gleiches gilt für viele andere Komponenten wie Ethernet-Module, die sowieso hohe Pegel zu verarbeiten haben. Verlassen sollten Sie sich aber nicht auf die Nutzbarkeit von 3,3-Volt-Komponenten mit 4,5 oder 5 Volt. Konsultieren Sie im Zweifel das Datenblatt des Bauteils oder suchen Sie in Foren nach Nutzern, die das betreffende Bauteil bereits seit längerer Zeit unter der erwähnten höheren Spannung einsetzen.

Keine Angst sollten Sie beim Betrieb von LEDs mit deutlich höheren Spannungen als den üblichen 1,8 bis 2,0 Volt haben. Verwenden Sie einen Pin, der Pulsweitenmodulation bietet — oder lassen Sie die LED einfach selbst blitzen: Drei Millisekunden auf vollen 5 Volt, gefolgt von einer halben Sekunde Pause ist für das menschliche Auge als deutliches Aufblitzen wahrnehmbar und gibt der LED genügend Zeit zur Abkühlung:

```
001 void loop() {
002   digitalWrite(led, HIGH);
003   delay(3);
004   digitalWrite(led, LOW);
005   delay(500);
006 }
```

Soll eine LED dauernd leuchten und steht kein Pin mit Pulsweitenmodulation zur Verfügung, arbeiten Sie mit Vorwiderständen. 560 Ohm oder 1 kOhm (Kiloohm) sind gute Ausgangswerte. Messen Sie in solch einem Aufbau mit dem Multimeter die an der LED anliegende Spannung. Liegt diese unter 2 Volt, ist alles okay, liegt sie darüber, hilft die Tatsache, dass Spannungsabfälle proportional zu den jeweiligen Widerständen sind, den wirksamen Widerstand der Diode zu berechnen und den Vorwiderstand entsprechend zu dimensionieren.

Inhaltsverzeichnis

Arduinos vorbereiten

In diesem Kapitel möchte ich einen kurzen Überblick über die Arduino-Familie geben — den einen richtigen Arduino gibt es nicht, stattdessen muss je nach Anwendungsfall neu entschieden werden, ob ein billiger Selbstbau-Arduino die Platine der Wahl ist oder vielleicht doch einer der klassischen Arduinos schneller zum Ziel führt. Am Ende des Kapitels erfahren Sie, wie Sie Ihren Arduino auf minimale Leistungsaufnahme trimmen und welche verschiedenen Stromversorgungen die Diät ermöglicht.

Arduino Pro und Pro Mini

Bei den abgedruckten Fritzing-Schaltplänen werden Sie bemerken, dass wir bis auf wenige Ausnahmen Arduino Uno und Pro Mini verwenden. Das hat auch mit der Allgegenwärtigkeit der ATmega328P zu tun. Meist sind Uno und Pro Mini beliebig austauschbar, und die Entscheidung für den einen oder anderen basiert auf der Positionierung der Pins oder auf der Möglichkeit des Pro Mini, ihn direkt aufs Breadboard zu stecken. In einigen Fällen verwenden wir den Pro Mini, weil es ihn in einer (mit 8 MHz getakteten) 3,3-Volt-Variante gibt – das spart Logic Level Converter oder Spannungsteiler und hält so den Aufbau kompakt.

Prototypen mit Arduino Uno, Zero und Leonardo
Hört man »Arduino«, hat man meist unwillkürlich die blaue Platine mit den beiden charakteristischen Buchsenleisten, dem USB-Port und der Buchse für die separate Stromversorgung vor Augen. Das liegt daran, dass viele Projekte, die den Weg in die Presse fanden, schnell und pfiffig auf Basis eines Arduino Uno oder des Vorgängers umgesetzt wurden. Tatsächlich sollten die klassischen Arduinos fürs »Schnell-mal-Ausprobieren« in keiner Bastelkiste fehlen.

1.1 Arduino Uno — der Klassiker

Der mit Abstand beliebteste Arduino ist der Arduino Uno. Er verwendet einen ATmega328P-Mikrocontroller, ein relativ altes Design, das auf Atmels 8-Bit-AVR-Architektur basiert. Dem Controller selbst fehlen einige moderne Schnittstellen wie USB oder ein Debugger, zudem ist er mit Preisen zwischen 1 Euro (Tausenderstückzahlen) und 5 Euro (einzeln beim Elektronikladen um die Ecke) relativ teuer.

Sein großer Vorteil ist, dass dieser Mikrocontroller (beziehungsweise seine beiden Vorgänger) seit über zehn Jahren das Herz der Arduino-Familie sind. Entsprechend viele fertige Bibliotheken verwenden die Hardwareeigenheiten des ATmega328P — manche unterstützen gar keine anderen Mikrocontroller als den ATmega328.

Der Arduino Uno kostet als fertige Platine mit Spannungswandler und USB-Anschluss 25 bis 30 Euro. Er nimmt Shields im gängigen verdrehsicheren Arduino-Format auf und verträgt Eingangsspannungen von 5 bis 12 Volt. Ein aufgelöteter Spannungswandler stellt daraus den Pins des Boards 3,3 und 5 Volt bereit — das erleichtert auch den Betrieb von mit 3,3 Volt spezifizierter Hardware. Unserer Ansicht nach ist wenigstens ein einziger Arduino Uno fürs Prototyping Pflicht, mit keinem anderen Board gelingt der Aufbau von Testschaltungen so schnell, bei keinem anderen Board ist die softwareseitige Schwelle so gering: Praktisch jede Bibliothek und jeder Sketch läuft sofort.

Das spätere Deployment, wenn kompakte Größe oder ein geringer Stückpreis wichtig ist, kann mit dem Arduino Pro Mini oder selbst gelöteten Rasterduinos zu Stückpreisen von weniger als 4 Euro erfolgen. Die Gefahr, dass der ATmega328P irgendwann einmal nicht mehr verfügbar sein wird, besteht kaum: Dieser Mikrocontroller wird so zahlreich in der Industrie verbaut, dass allein aus Ersatzteilgründen die Lieferbarkeit bis 2025 sichergestellt ist — auch wenn in einigen Jahren mit Lieferfristen von bis zu einem halben Jahr für große Stückzahlen gerechnet werden muss.

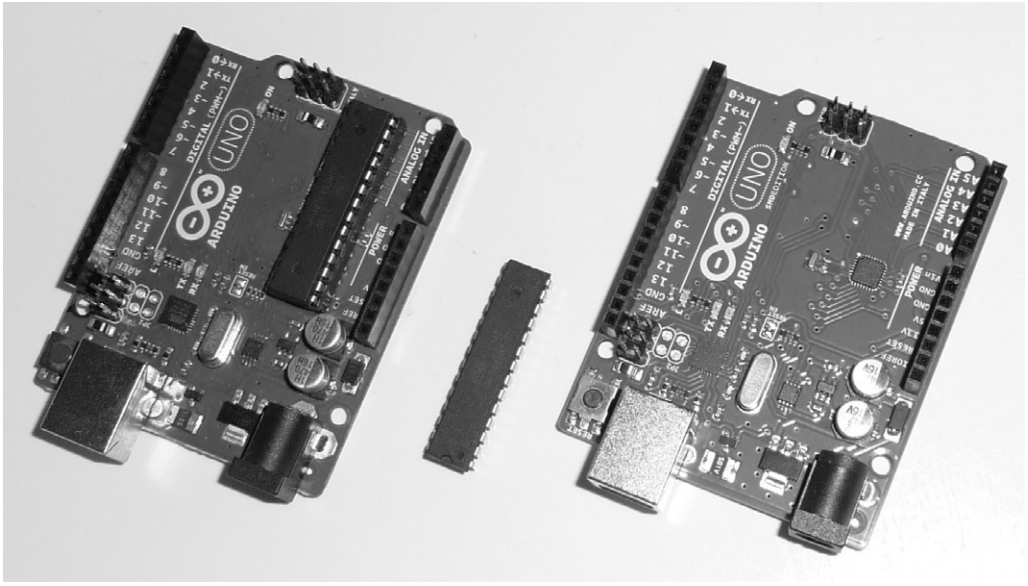


Bild 1.1: Zwei Arduino Uno mit ATmega328P — die Version links mit dem gesockelten Controller ist etwas teurer, ein beschädigter Mikrocontroller kann jedoch leichter ausgetauscht werden.

Der Arduino Uno ist in zwei Versionen erhältlich: als SMD-Variante mit fest verlötetem Controller und als gesockelte Version mit Controller im DIL-Gehäuse (Dual-Inline-Gehäuse zum Durchstecken in Lochrasterplatinen oder Einstecken in Sockel). Letztere ist 2 bis 3 Euro teurer, bietet aber im Fall eines beschädigten Controllers eine leichtere Austauschbarkeit, zudem kann solch ein Arduino dazu genutzt werden, fast ohne Umstände Programme auf »standalone« betriebene ATmega328P zu flashen.

1.2 Zwischenlösung Leonardo

Aufgrund der alten Architektur und den kleinen Margen versucht Chiphersteller Atmel, seine Kunden auf andere Mikrocontroller-Familien zu »stupsen«. Eine Art Zwischenlösung ist der ebenfalls auf dem AVR-Kern aufbauende ATmega32u4, der ähnliche Eckdaten wie der ATmega328P aufweist, aber ein paar Eingänge mehr und einen integrierten USB-Port hat. Verbaut wird der ATmega32u4 unter anderem auf dem Arduino Leonardo. 32u4-basierte Arduino-Boards fristen mittlerweile ein Nischendasein, sie haben aber durchaus ihre Berechtigung, wenn feinere Auflösungen am Analog-Port notwendig sind oder der Arduino als Eingabegerät an einem PC agieren soll.

Ein Arduino Leonardo kostet ebenfalls 25 bis 30 Euro. Der ATmega32u4 ist leider nicht im bastelfreundlichen DIL-Package erhältlich, sondern nur als SMD-Komponente. Für den Hobbybereich und die Arbeit auf dem Breadboard kommen daher am

ehesten die in Minimalkonfiguration bereits ab 6 Euro erhältlichen Klone und Derivate infrage.

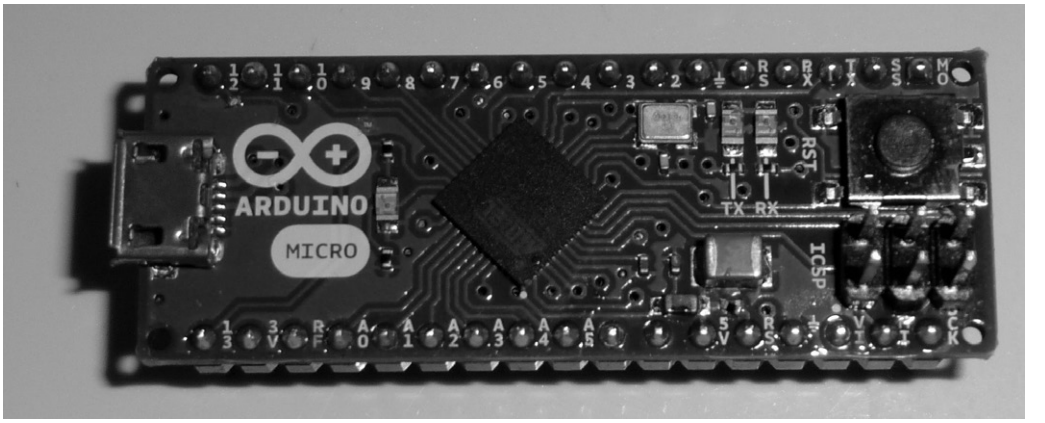


Bild 1.2: Arduino Micro, die Breadboard-freundliche Version des Leonardo, kann sich gegenüber einem PC als Tastatur oder Maus ausgeben.

1.3 Arduino-Zukunft Zero?

Im Sommer 2014 stellte das Arduino-Projekt gemeinsam mit Atmel eine neue Platine als designierten Nachfolger des Uno vor: Der Arduino Zero verwendet den Controller SAMD21, der auf einem 32-Bit-ARM-Cortex-Kern basiert — dieser verspricht bei vielen Datentypen deutlich schnellere Berechnungen, dazu kommt ein erheblich größerer Programm- und Variablenspeicher. Zudem ist der ARM-Kern mit 48 MHz dreimal so schnell getaktet wie der ATmega328P.

Mit an Bord sind zwei unabhängig voneinander ansprechbare USB-Schnittstellen und ein JTAG-Port. Bei Redaktionsschluss waren vom Arduino Zero nur einzelne Vorserienexemplare im Umlauf und die Anpassungen an der Arduino-Standard-Bibliothek noch nicht zu 100 % abgeschlossen.

Der erwartete Preis wird mit 23 bis 28 Euro knapp unter dem von Uno und Leonardo liegen. Kolportierte Tausenderpreise des verwendeten SAMD21 lassen im Laufe des Jahres 2015 günstige Klone zu Preisen ab ca. 6 Euro erwarten. Wer also einen modernen Mikrocontroller mit hoher analoger Auflösung und viel Rechenleistung sucht, sollte sich zusätzlich zum Uno einen Zero besorgen — allerdings müssen Sie mit vielen kleinen zu umschiffenden Klippen rechnen, bis Code, der für Uno/328P entwickelt wurde, reibungslos mit dem ARM-Kern des SAMD21 harmoniert.

Im Gegenzug arbeiten Sie auf einer modernen Plattform, die im Gegensatz zu Uno/328P ihr gesamtes Potenzial noch gar nicht ausschöpft: So sind in der Cortex-Familie auch Controller vertreten, die integriertes Ethernet oder einen CAN-Bus (für Automobilanwendungen) mitbringen.

1.4 »Starke« Mitglieder der Arduino-Familie

Die Arduino-Familie ist mittlerweile fast unüberschaubar groß. Einige ihrer Mitglieder weisen praktisch keine Kompatibilität zu »klassischen« Arduino-Konzepten mehr auf und sind dementsprechend schwierig in eine gewachsene Arduino-Infrastruktur einzufügen. Andere kombinieren pfiffig Mikrocontroller und Mikroprozessor und empfehlen sich daher als oft teure, aber meist sehr effiziente Nischenlösungen. Gemein ist ihnen der relativ hohe Preis, der sich aber rechtfertigt, wenn kleine Stückzahlen benötigt werden, aber eine schnelle Entwicklung erforderlich ist.

1.5 Intel Galileo

Intels Galileo ist eher ein Versuchsballon des großen Chipherstellers, um herauszufinden, wie stark ein Mikroprozessor mit Pentium-Kern zum Mikrocontroller konvergieren kann. Hardwareseitig ist der Galileo ein interessanter Zwitter, der es schafft, die meisten Funktionen für die digitale Ein- und Ausgabe des Prozessors zu integrieren. Leider ist die Gesamtlösung softwareseitig relativ fett: Arduino-Code wird in Linux-Objektdateien kopiert, deren Abarbeitung auch mal unterbrochen wird, wenn der Linux-Kernel mit etwas anderem beschäftigt ist. Die Folge ist ein relativ träges IO, eine höhere Leistungsaufnahme und eine geringere Stabilität als bei reinen Mikrocontroller-Anwendungen.

Da Intel derzeit starke wirtschaftliche Interessen hat, von oben in einen Markt vorzudringen, den ARM gerade mit den Cortex-basierten Mikrocontrollern bedient, bestehen zwischen Intel und vielen Universitäten Programme, aus denen die Platinen für Studienzwecke kostenlos bezogen werden können. Sollten Sie wissenschaftlicher Mitarbeiter, Dozent oder Student an einer Hochschule sein und Bedarf nach Basisknoten für vernetzte Sensoren haben oder einen Kurssatz für ein Seminar benötigen, prüfen Sie, ob zwischen Ihrer Hochschule und Intel ein Rahmenvertrag besteht, über den die Platinen kostenlos oder sehr billig bezogen werden können.

Ohne Bezug zu ermäßigten Preisen sinkt die Attraktivität den Boards: Bei Redaktionsschluss waren für ein Galileo-Board rund 65 Euro zu zahlen. Das macht diese Plattform nur dann interessant, wenn beispielsweise für die effizientere Auswertung von Rohdaten ein x86-Assembler vorhanden ist, der weiterverwendet werden sollte, oder generell x86-Code-Kompatibilität neben analogen Eingabemöglichkeiten erforderlich ist, gleichzeitig aber nicht allerhöchste Ansprüche an Reaktionszeiten gestellt werden.

1.6 Arduino Yún

Das chinesische Wort Yún bedeutet »Wolke«, was bereits andeutet, dass dieser Arduino als Cloud-Variante ins Internet der Dinge strebt. Der Arduino Yún vereinigt auf einer Platine im Standardformat einen Arduino Leonardo mit ATmega32u4-Mikrocontroller und einen MIPS-basierten Mikroprozessor, auf dem ein Linux läuft.

Konkret handelt es sich um eine OpenWRT-basierte Linux-Distribution — und auch die restliche Hardware entspricht mit WLAN-Schnittstelle und Ethernet-Port einem typischen WLAN-Access-Point. Dank des OpenWRT-Frontends für die Netzwerkkonfiguration ist der Yún sehr leicht einzurichten, er kann als drahtloser Client in ein bestehendes WLAN integriert oder per Ethernet verbunden werden. Wird er per Ethernet eingebunden, kann die drahtlose Schnittstelle einen weiteren Access-Point aufspannen.

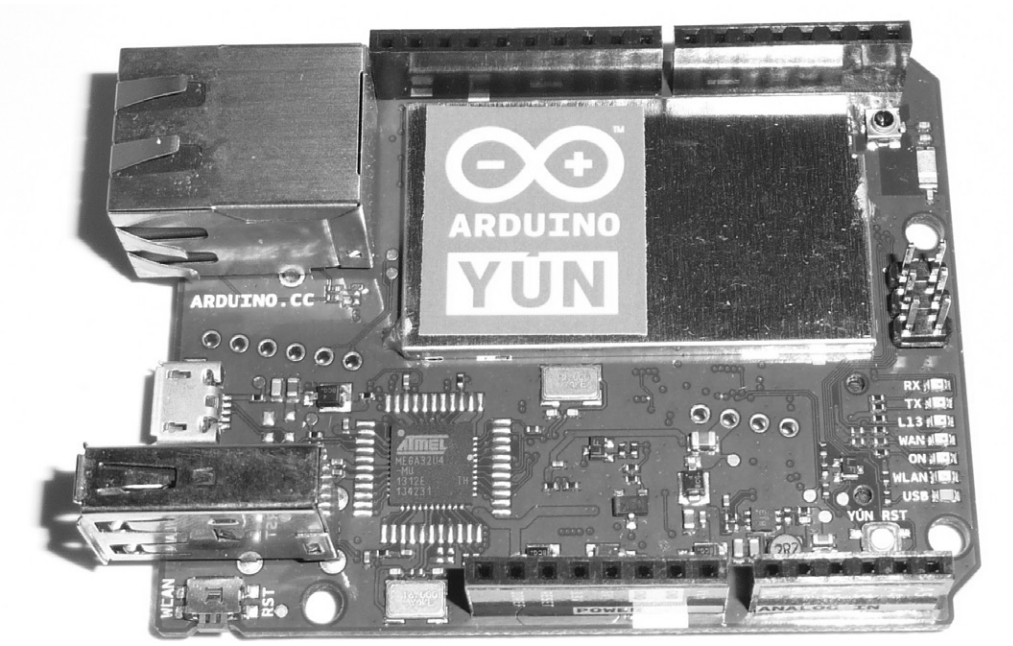


Bild 1.3: Auf den ersten Blick ein normaler Arduino mit USB-Host und Ethernet-Port, tatsächlich der Chipsatz eines DSL-Routers, der um einen Arduino-kompatiblen Mikrocontroller ergänzt wurde.

Der Clou des Arduino Yún ist jedoch die Kommunikation zwischen Mikrocontroller-Seite und Mikroprozessor-Seite: Technisch läuft sie über eine serielle Schnittstelle, softwareseitig stellt Arduino eine Bridge-Bibliothek bereit, die die Kommunikation beider Seiten stark vereinfacht. Zur Bridge gehört ein gemeinsamer Variablenspeicher (Mailbox genannt), die Möglichkeit, direkt aus der Mikrocontroller-Seite Linux-Befehle auszuführen, eine netzwerktransparente serielle Konsole und eine API, mit der verschiedene Webdienste aus Arduino-Sketches angesprochen werden können, ohne dass auf Linux-Seite programmiert werden muss.

Das Auslesen und Setzen von Variablen in der Mailbox ist zudem über eine simple REST-API möglich. Wer bereits ein wenig Arduino-Erfahrung hat und vielleicht über erste REST-Kenntnisse verfügt, kann mit dem Yún in kürzester Zeit effektive Ergeb-

nisse erzielen. Wenn man dazu noch bereit ist, sich in die Linux-Seite der Bridge einzuarbeiten, erhält man den wahrscheinlich flexibelsten Arduino der letzten zehn Jahre.

Bei Drucklegung betrugen die Preise für den Arduino Yún 65 bis 75 Euro.

1.7 Arduino Tre

Einem ähnlichen Konzept wie der Arduino Yún folgt der Arduino Tre, dessen Entwicklung maßgeblich von Texas Instruments finanziert wurde. Er nutzt jedoch auf Linux-Seite keine kompakte Hardware auf MIPS-Architektur, sondern einen leistungsfähigeren ARM-Prozessor. Konkret entspricht die Linux-Seite fast vollständig dem Beagle Bone Black, während die Mikrocontroller-Seite ebenfalls auf den ATmega32u4 setzt.

Die Kombination beider Boards resultiert in einer etwas größeren Platine, was teilweise den zusätzlichen Ports geschuldet ist: Der Tre führt neben den charakteristischen Arduino-Leisten auch die typischen zweireihigen Erweiterungsleisten des Beagle Bone nach draußen und enthält darüber hinaus einen Sockel für XBee-Funkmodule. Neben vier USB-Host-Ports und Ethernet verfügt der Tre über einen HDMI-Port und Audioausgänge.

Softwareseitig soll die finale Version auch die vom Yún bekannte Bridge-Bibliothek unterstützen. Ansonsten spricht der Tre eher Linux-erfahrene Kundschaft an, da große Teile der Konfiguration der Linux-Seite auf der Kommandozeile erledigt werden müssen. Ein erstes Los Vorserienplatinen war im Juni 2014 zum Preis von 150 Euro erhältlich, eine finale Version soll zum Preis von 75 bis 100 Euro erhältlich sein, wenn dieses Buch ausgeliefert wird.

1.8 Klein, billig und schnell einsatzbereit

Platinenpreise ab 25 Euro aufwärts und die Größe der Arduino-Standardplatine, die etwa der Größe einer Scheckkarte entspricht, sowie die (In-) Effizienz des aufgelöteten Spannungswandlers lassen Arduino Uno & Co. nicht gerade als glücklichste Wahl erscheinen, wenn Dutzende Sensoren herausgebracht werden sollen, die vielleicht ein Jahr oder länger auf einem Batteriepack Daten auf SD-Karte loggen oder per Funkmodul zu einer Zentrale schicken sollen.

1.9 Arduino Pro Mini

Unter der Bezeichnung Arduino Pro Mini entwickelte Sparkfun eine Platine, die den ATmega328P in SMD-Version ohne USB-nach-seriell-Wandler und ohne Spannungswandler auf eine Platine im Rastermaß 13 × 7 Punkte bringt. Die Pins sind dabei so angeordnet, dass mit zwei Stiftsockelleisten ein Aufstecken aufs Breadboard möglich ist. Zum Aufspielen von Sketches ist ein separater USB-nach-seriell-Wandler

erforderlich, beispielsweise Arduinos USB2SERIAL (ca. 10 Euro) oder ein FTDI-Breakout-Board.

Darüber kann beim Test auch die Stromversorgung erfolgen. Sparkfuns Design muss sich die Kritik gefallen lassen, dass die Pins A4 bis A7 (A6 und A7 fehlen beim Uno!) an einer ungünstigen Stelle herausgeführt werden. Besser macht dies Watterotts Wattuino, der mit rund 10 Euro nicht teurer als das Original ist. In China haben wir zudem Klone eingekauft, von denen einer das Sparkfun-Design eins zu eins übernahm und der andere die SPI- und I^2C-Pins ähnlich intelligent wie Wattuino platzierte.

Der Preis dieser Platinen erschien mit ca. 3 Euro unschlagbar günstig, zu bedenken ist jedoch, dass ab 22 Euro Zollwert 19 % Einfuhrumsatzsteuer bezahlt werden müssen — immerhin ist der Eigenimport bis 150 Euro zollfrei und wird vereinfacht bearbeitet. Ein stärkerer Wermutstropfen waren Charge-Platinen, die ohne Bootloader ausgeliefert wurden und von uns von Hand mit einem Bootloader versehen werden mussten. Gegen den Import größerer Mengen spricht zudem die fehlende ROHS-Kennzeichnung vieler Platinen, die verbietet, diese in den Verkehr zu bringen: Selbst importierte Platinen müssen in der eigenen Organisation verarbeitet werden.

Pro Mini und Klone gibt es in den Versionen für 5-Volt-Spannungsversorgung (16 MHz) und 3,3-Volt-Spannungsversorgung (8 MHz), beide unterscheiden sich nur durch den aufgelöteten Quarz und die Voreinstellungen des Mikrocontrollers. Prinzipiell können Sie die 3,3-Volt-Variante mit 5 Volt betreiben, wenn angeschlossene Peripherie diese Spannung erfordert — allerdings steht dann nur die halbe maximale Rechenleistung zur Verfügung, sollten tatsächlich Daten verarbeitet werden.

Umgekehrt kann ein 5-Volt-Pro-Mini per Änderung des Bootloaders (konkret werden sogenannte Fuses gesetzt) auf die Verwendung des internen Taktgebers (8 MHz) umgeschaltet werden. Dann ist ein stabiler Betrieb mit 3,3 Volt möglich — einziger Nachteil ist, dass der interne Oszillator weit weniger präzise läuft als ein externer Quarz, und das nehmen Ihnen manche Peripheriegeräte und Funkprotokolle übel.

Nicht möglich ist ein Betrieb der 5-Volt-Variante mit 3,3 Volt: Der Brownout, also die Spannung, bei der der Mikrocontroller aus Sicherheitsgründen in eine Reset-Schleife schaltet, liegt bei 4,3 Volt. Und selbst mit deaktiviertem Brownout wäre mit 3,3 Volt kein stabiler Betrieb mit 16 MHz möglich.

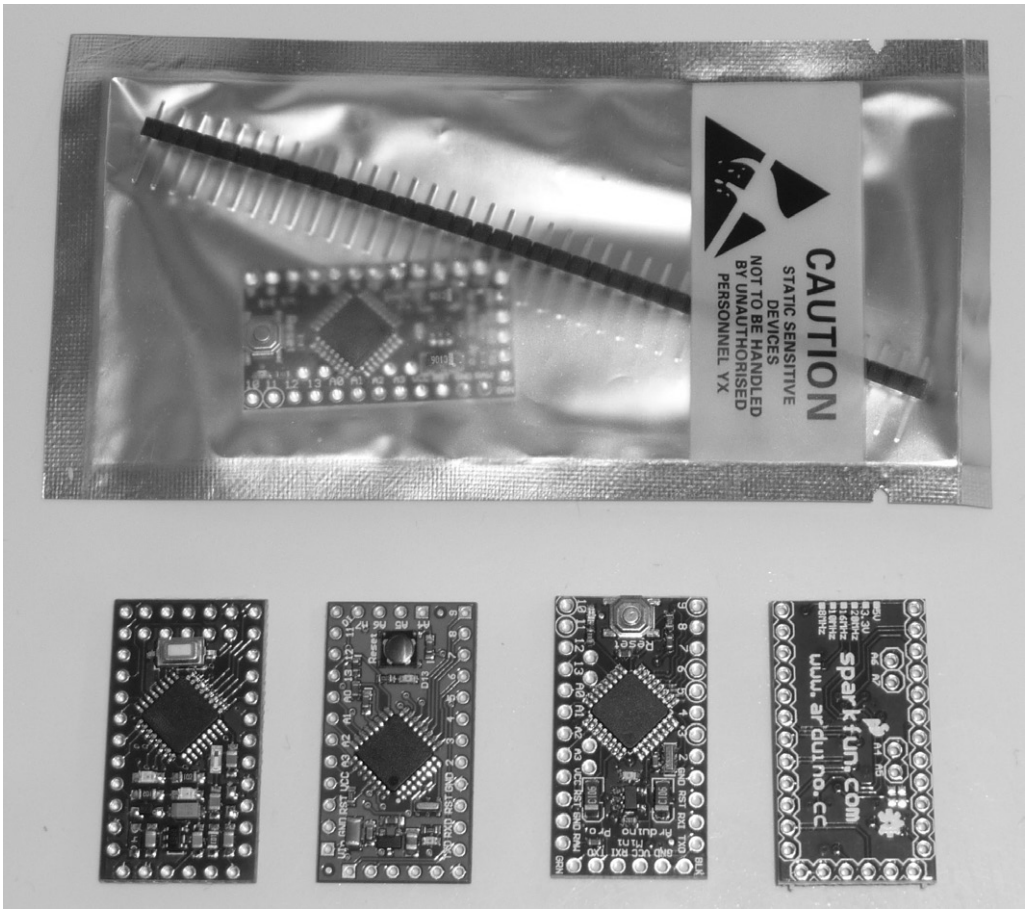


Bild 1.4: Klone und Derivate: Viele »Pro Minis« basieren auf dem Design von Sparkfun (rechts) — unsere Favoriten sind die chinesische Weiterentwicklung mit doppelter Leiste am Kopfende (links, ca. 3 Euro bei *www.banggood.com*) und die in Deutschland hergestellte Version von Watterott (zweite von links, ca. 10 Euro, *www.watterott.com*).

Ausgeliefert werden »Pro Minis« und Klone meist mit unverlötetem Stiftsockel. Üblicherweise verlötet man diese nach unten, sodass der Mikrocontroller oben ist und der längere Teil der Stifte nach unten zeigt. Die sechs abgewinkelten Pins verlöten Sie in der Regel so, dass sie nach vorne zeigen. Mit den zwei parallel nach unten zeigenden Stiftleisten können Sie auf die mittlerweile als zweiten Standard entstandenen Shields verbinden — oder natürlich selbst aus Loch- oder Streifenrasterplatine Shields erstellen.

Beachten Sie, dass für die Position von A4 bis A7 und die serielle Schnittstelle keinerlei eindeutige Standards existieren, der USB-seriell-Wandler muss mal so, mal andersherum aufgesteckt werden. Sollte der Platz knapp sein, verzichten Sie auf die abgewinkelten Pins für den Wandler: Da die Lötaugen durchgeöst sind, können Sie zum

Programmieren die Stifte einfach einstecken und mit der Hand oder einer Wäscheklammer leicht verkanten.

1.10 Der Selbstbau-Minimal-Arduino

Nackte ATmega328P-Mikrocontroller gibt es im DIL-Package beim Elektronikfachhandel um die Ecke meist vorrätig für ca. 5 Euro. Wer zwei Tage warten kann, erhält die einzelnen Controller ab rund 3 Euro. Viel mehr als eine Loch- oder Streifenrasterplatine ist nicht nötig, um einen eigenen Arduino zu bauen, lediglich ein Kondensator und ein Widerstand sollten vorhanden sein. Da einige Versandhändler die ATmega328P-Mikrocontroller in Arduino-Einstellungen ausliefern (16 MHz mit externem Taktgeber), sollten ein 16-MHz-Quarz und zwei Kondensatoren bereitliegen.

Die Einkaufsliste zum Aufbau auf dem Breadboard:

- Arduino Uno oder Pro Mini als ISP
- USB-seriell-Wandler (5 bis 12 Euro)
- ATmega328P im DIL-Package (3 bis 5 Euro)
- Quarz 16 MHz (<1 Euro)
- 2 Kondensatoren 22 pF (0,20 Euro)
- 1 Kondensator 10 µF (0,10 Euro)
- 1 Kondensator 0,1 µF (0,20 Euro)
- 1 Widerstand 10 kOhm oder 22 kOhm

Wenn sichergestellt ist, dass der frisch erworbene ATmega328P auf einen internen Taktgeber (1 MHz oder 8 MHz) eingestellt ist, können Sie auf Quarz und die beiden 22pF-Kondensatoren verzichten.

Zur Vorbereitung müssen Sie die Hardwaredefinitionen für die nackten ATmegas in Ihrem Arduino-Ordner ablegen. Den finden Sie im Heimatverzeichnis, also dem Ordner, den Sie unter Linux und OS X unter der Pfadvariablen `$HOME` und unter Windows unter `%APPDATA%` erreichen. Entpacken Sie dort das zur Arduino-IDE passende Zip-Paket mit den Hardwaredefinitionen — eines für die im Buch verwendete Version Arduino 1.6 finden Sie auf der Webseite des Autors (*www.arduino-hausautomation.de*) oder des Verlags (*www.buch.cd*). Starten Sie anschließend die Arduino-IDE neu — in der Hardwareliste sollten nun ganz oben die Barebone-ATmegas und -ATtinys auftauchen.

Ulli Sommer

Mikrocontroller programmieren mit Bascom Basic

Messen, Steuern, Regeln
und Robotertechnik
mit den AVR-Controllern

Vorwort

Mikrocontroller zu programmieren wird, wie man in den verschiedensten Foren und Fachzeitschriften beobachten kann, immer populärer. Das liegt daran, dass Mikrocontroller und zusätzliche Peripheriebausteine immer günstiger angeboten werden und an Schulen zunehmend in Mikrocontroller und Computertechnik unterrichtet wird. Schaltungen, die man früher mit mehreren ICs aufbauen musste, können nun in einem einzigen Mikrocontroller untergebracht werden. Das spart Zeit, Geld und Platz auf der Leiterplatte. Dieses Buch baut auf dem beliebten Basic-Compiler BASCOM-AVR mit integrierter Entwicklungsumgebung auf. Er ist für fast alle 8-Bit-AVR- und XMega-Mikrocontroller der Firma Atmel geeignet. BASCOM erfreut sich nicht nur bei Einsteigern, sondern auch bei Entwicklungsprofis, immer größerer Beliebtheit und stellt inzwischen schon fast einen Basic-Standard bei AVRs dar.

Viele Problemstellungen, die früher zeitaufwendig in Assembler oder C gelöst werden mussten, können durch diesen modernen Compiler blitzschnell mit wenigen Befehlen erledigt werden. Beispielsweise genügt ein einziger Befehl, um aus einem I/O-Port eine RS232-Schnittstelle, einen I²C-Bus oder einen Servoanschluss zu machen. Solche Dinge erfordern in anderen Programmiersprachen oft einen enormen Aufwand.

BASCOM erzeugt optimierten Maschinen-Code. Es werden alle AVR-RISC-Controller mit internem RAM der Serien AT90S, ATmega und ATTiny unterstützt. Mit einigen Einschränkungen sind jetzt auch ATTiny-Controller ohne SRAM mit BASCOM-AVR programmierbar. Dazu steht die $TINY-Funktion zur Verfügung.

Aus diesen Gründen ist der BASCOM Basic-Compiler ideal für den Einstieg in die Mikrocontroller-Programmierung geeignet. Er ist trotzdem sehr leistungsfähig und ermöglicht auch optimierte komplexe Software-Entwicklungen mit Profianforderungen. Ein weiterer großer Vorteil ist, dass diese Entwicklungsumgebung in hohem Tempo weiterentwickelt wird und die Updates kostenlos sind. So war BASCOM auch eine der ersten AVR-Entwicklungsumgebungen, die unter Vista und Windows 7 liefen.

Ich wünsche Ihnen viel Spaß beim Lesen und Experimentieren mit diesem Buch!

Waidhaus, Juli 2011

Ulli Sommer

Inhaltsverzeichnis

6 Programmiergeräte

Wenn Sie hauptsächlich mit BASCOM und 8-Bit-Atmel-Controllern arbeiten möchten, empfehle ich, den BASCOM USB-ISP-Programmer zu verwenden. Er kann direkt mit BASCOM verwendet werden und man kann zudem auch die Fuse- und Lock-Bits unter BASCOM einstellen. Mit der USB-Schnittstelle ist er nicht nur sehr schnell bei der Programmierung, sondern erlaubt auch den Betrieb an neueren Rechnern, die meist nur noch über USB-Schnittstellen verfügen.

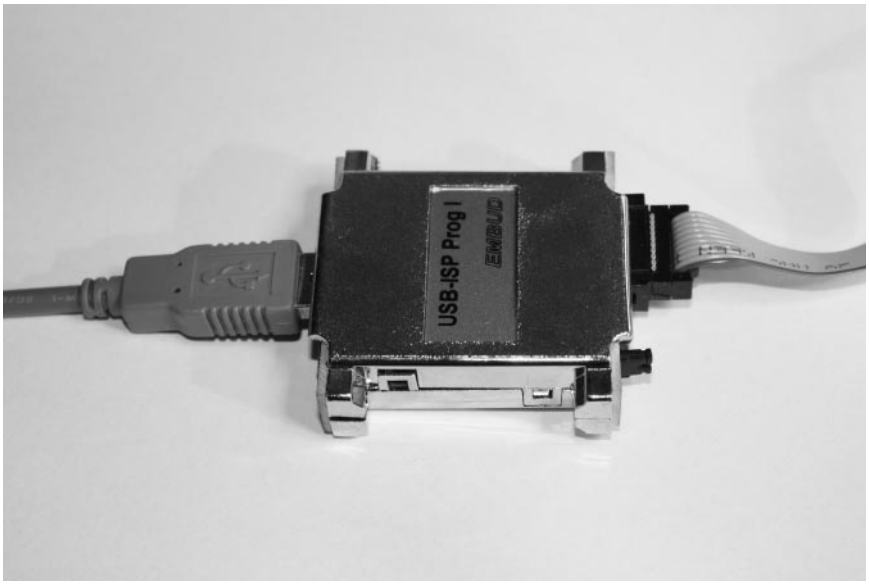


Bild 6.1: BASCOM USB-ISP-Programmer.

Beziehen lässt sich der Programmer direkt vom BASCOM-Hersteller oder in Deutschland über *www.Robotikhardware.de*.

Weit verbreitet ist auch der Atmel USB-ISP-Programmer MKII. Er wird nicht direkt unter BASCOM unterstützt, lässt sich aber auch dafür einrichten – jedoch nicht so komfortabel wie der BASCOM-eigene Programmer. Eine Aufstellung der unter BASCOM verwendbaren Programmiergeräte finden Sie unter *http://avrhelp.mcselec.com/index.html?supported_programmers.htm*.

In diesem Buch wird der BASCOM USB-ISP-Programmer verwendet. Die Art, wie bei der Programmierung vorgegangen wird, ist bei anderen Programmiergeräten ähnlich.

Eine kostengünstige Alternative zu käuflich erwerbbaren Programmiergeräten ist der Eigenbau. Es werden nicht allzu viele Bauteile benötigt, womöglich finden sich die Teile sogar in Ihrer Bastelkiste. Der zu verwendende Computer muss jedoch eine parallele Schnittstelle besitzen. Der folgende Schaltplan zeigt, wie es geht.

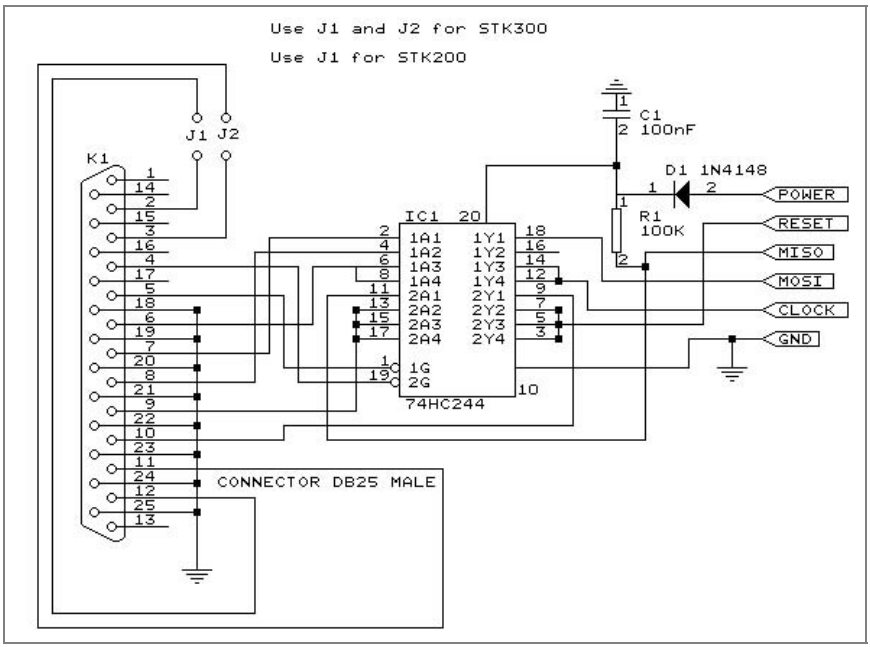


Bild 6.2: Parallel-Port ISP-Programmer. (Quelle: BASCOM-Hilfe)

Diese Version ist auch bei verschiedenen Elektronikversendern für meist unter 20 € käuflich zu erwerben.

Es gibt noch eine Sparversion eines Parallel-Port-ISP-Programmers. Sie ist aber mit Vorsicht zu genießen, da ein Kurzschluss der I/Os des Parallelports zu einem Defekt der Schnittstelle führen kann. Die 330-Ohm-Widerstände schützen den Port nur geringfügig. Es ist also immer ratsamer, die Version mit dem Puffer-IC 74HC244 aufzubauen.

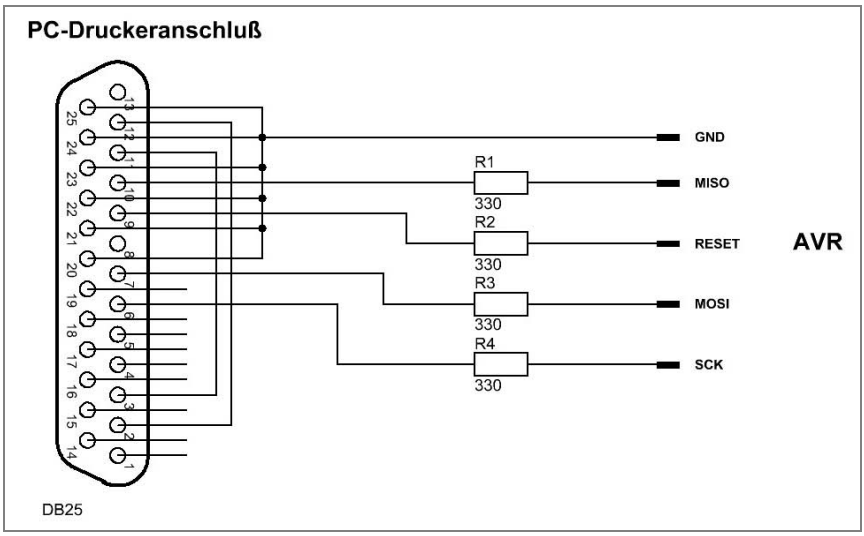


Bild 6.3: Der »Lowcost«-ISP-Programmer für die parallele Schnittstelle.

7 Interessante AVR-Boards für den Einstieg

Es gibt natürlich auch einige kostengünstige AVR-Boards zu kaufen, die den Einstieg in die AVR-Welt vereinfachen. Eine breite Palette diverser AVR-Platinen bietet die Firma Robotikhardware an. Die Module besitzen meist schon diverse Zusatzhardware wie DC-Motortreiber, Schrittmotortreiber, Funkmodule, Schaltausgänge mit Transistoren oder Relais. Programmiert werden die Boards wie die Eigenbauschaltung. Es folgt ein kleiner Überblick verschiedenster Mikrocontroller-Platinen aus dem Angebot der Firmen Robotikhardware und Atmel.

7.1 RN-CONTROL

Für den Einstieg, erste Mikrocontroller-Experimente aber auch für konkrete Projekte wie autonome Roboter, Steuerungen und vieles mehr gibt es das Board RN-CONTROL. Bei der Entwicklung wurde besonders auf ein gutes Preis-Leistungs-Verhältnis geachtet. Trotz günstigen Preises ist ein sehr flexibles Board für unzählige Anwendungsmöglichkeiten entstanden. Über den I²C-Bus stehen zahlreiche Erweiterungs-Boards zur Verfügung. So können beispielsweise die gleichen I²C-Erweiterungen kombiniert werden wie beim großen RNBFRA-Board (Relaiskarte, Sprachausgabe usw.).

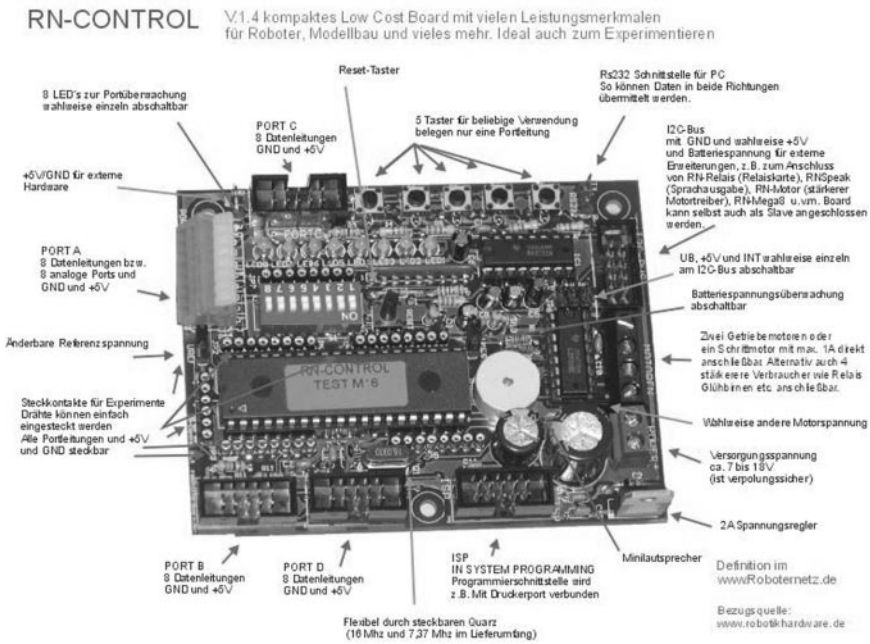


Bild 7.1: RN-Control 1.4. (Quelle: Robotikhardware)

Besonders viel Wert wurde auch auf den einfachen Aufbau und viele Experimentier- und Einsatzmöglichkeiten gelegt. Mit diesem Board lässt sich schon ein recht ausgereifter Roboter konstruieren. Ultraschallsensoren, Infrarot-Entfernungssensoren, Motoren u. v. m. können direkt angeschlossen werden. Da das Board auch in der Community Roboternetz recht beliebt ist, findet man dort viele Tipps und Programme.

7.2 RN-Mega8PLUS

Der Nachfolger des RN-Mega8-Boards, jetzt mit Funkmodulsteckplatz und weiteren Optimierungen. Dieses Nachfolge-Board wurde speziell zum Experimentieren mit den Mikrocontrollern Mega8 und Mega168 entworfen.

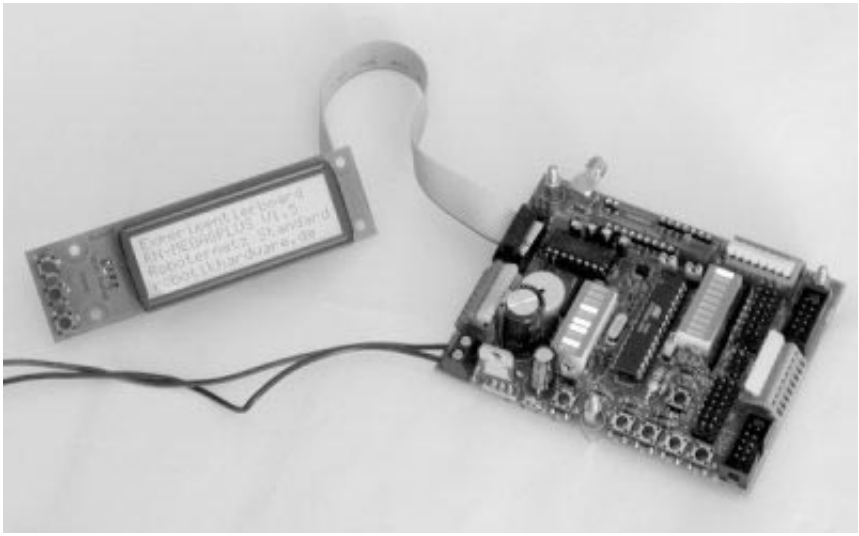


Bild 7.2: RN-Mega8PLUS. (Quelle: Robotikhardware)

Die Controller Mega8 und Mega168 eignen sich wegen ihres günstigen Preises und ihrer geringen Baugröße für zahlreiche Aufgaben, bei denen ein Mega16 oder ein Mega32 überdimensioniert wäre. Im Bereich Robotik kann dieser Controller ideal auch als Co-Controller für Servosteuerung, Motorsteuerung, Display-Ausgabe, Sensorüberwachung und vieles mehr eingesetzt werden. Oft sind kaum externe Bauteile notwendig. Um den Controller jedoch für eine spezielle Aufgabe programmieren zu können, bedarf es einer Entwicklungsumgebung die quasi alle Ports steckbar zugänglich macht, also auch die visuelle Überwachung der Port-Zustände erlaubt.

Für diese Aufgabe ist RN-Mega8PLUS ideal. Ganze 20 Ports können gleichzeitig visuell über Leuchtbalken überwacht werden. Nahezu alle Ports sind über einfache Steckklemmen erreichbar. Zudem verfügt das Board über jeweils einen genormten LCD-Display-, I²C-Bus-, RS-232-, Servo- und ISP-Anschluss.

Eine Besonderheit von RN-Mega8PLUS ist der Steckplatz für ein EasyRadio-Funkmodul. Dadurch wird das Board funkkompatibel zu RN-Mega128Funk, RN-Steuerung und RN-Funk. So können Daten mit anderen Boards oder PCs per Funk ausgetauscht werden. Das Funkmodul ist optional über *www.robotikhardware.de* beziehbar – es wird nur eingesteckt und man kann loslegen.

7.3 RN-MINICONTROL

Dieses Controllerboard zeichnet sich durch seine kompakte Größe (nur 5 cm x 7,8 cm) und sehr geringen Strombedarf aus. Besonders viel Wert wurde auch auf die vielseitigen Anschlüsse gelegt. Nahezu alle Ports stehen dem Anwender somit zur Verfügung. Besonders günstig sind die wichtigen AD-, Interrupt-, Timer- und PWM-Ports auf die Stecker verteilt worden. So lassen sich Servos, Drehgeber und RC-Empfänger, aber auch Motortreiber (Doppel-H-Brücken wie RN-VN2Dualmotor) oder LCDs direkt anschließen. Natürlich steht auch der I²C-Bus zur Verfügung. Alle Stecker sind zudem kompatibel zu den Roboternetz-Definitionen.

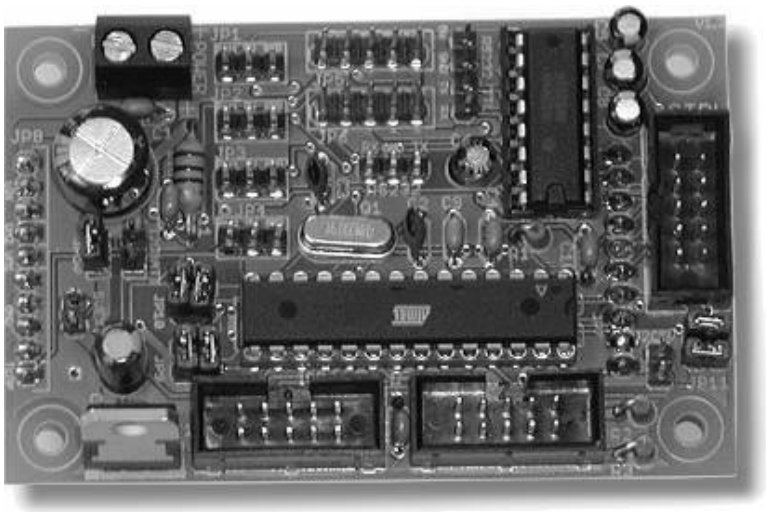


Bild 7.3: RN-MINICONTROL. (Quelle: Robotikhardware)

Obwohl das Board, wie auch der »große Bruder« RN-Control, zum Experimentieren als Haupt-Board verwendet werden kann, ist es in erster Linie als praktisch einsetzbares Zusatz-Board für echte Projekte gedacht.

RN-MINICONTROL ist als kostengünstiges Co-Controllerboard ideal als Zweit- oder Dritt-Board in Projekten (Roboter etc.). Ein besonderes Feature sind zwei Stiftleisten auf der Unterseite. Dadurch kann das Board auf ein anderes Board wie RN-VNH2Dualmotor oder Lochrasterplatinen aufgesteckt werden. Das Board ist mit dem leistungsstarken Controller ATmega168 ausgestattet. Er ist weitgehend kompatibel zum ATmega8, daher kann wahlweise auch ein ATmega8 bestückt werden. Der Mega168 bietet jedoch weit mehr Features: 6x PWM, drei Timer, 16 KB Speicher und Interrupt an jedem Pin.