#makers
DO IT.

Behandelt auch openHAB, FHEM, Home Assistant und ioBroker

Peter & Stephan Hüwe

IoT @ Home

Smart Gadgets mit Arduino, Raspberry Pi, ESP8266 und Calliope entwickeln

HANSER

Hüwe/Hüwe

IoT at Home

BLEIBEN SIE AUF DEM LAUFENDEN!

Hanser Newsletter informieren Sie regelmäßig über neue Bücher und Termine aus den verschiedenen Bereichen der Technik. Profitieren Sie auch von Gewinnspielen und exklusiven Leseproben. Gleich anmelden unter

www.hanser-fachbuch.de/newsletter

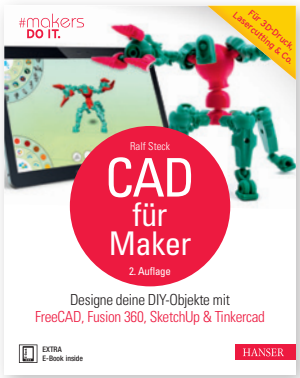
HANSER

Alles für Maker

Horsch
3D-Druck für alle
ISBN 978-3-446-44261-0

Regele
Mach was mit 3D-Druck!
ISBN 978-3-446-44781-3

Rother
3D-Drucken ... und dann?
ISBN 978-3-446-45062-2



Steck
CAD für Maker
ISBN 978-3-446-45681-5

Pomaska
3D-Fotos und -Videos
ISBN 978-3-446-45630-3

Kehrer, Philipp, Rens
Lasercutting
ISBN 978-3-446-45039-4

Steck
CNC-Fräsen für Maker
ISBN 978-3-446-45491-0

Bartmann, Donges
Open Robots für Maker
978-3-446-45489-7

Jänisch, Donges
Mach was mit Arduino!
ISBN 978-3-446-45128-5

Bertko, Weber
Home, Smart Home
ISBN 978-3-446-45061-5

Schmidt
Raspberry Pi programmieren mit C/C++ und Bash
ISBN 978-3-446-45342-5

Hüwe
Raspberry Pi für Windows 10 IoT Core
ISBN 978-3-446-44719-6

Weitere Infos unter **www.hanser-fachbuch.de/maker**

Peter Hüwe
Stephan Hüwe

IoT at Home

Smart Gadgets mit Arduino, Raspberry Pi, ESP8266 und Calliope entwickeln

HANSER

Die Autoren:
Peter Hüwe, Gersthofen
Stephan Hüwe, Augsburg

Alle in diesem Buch enthaltenen Informationen wurden nach bestem Wissen zusammengestellt und mit Sorgfalt geprüft und getestet. Dennoch sind Fehler nicht ganz auszuschließen. Aus diesem Grund sind die im vorliegenden Buch enthaltenen Informationen mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autor(en, Herausgeber) und Verlag übernehmen infolgedessen keine Verantwortung und werden keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Weise aus der Benutzung dieser Informationen – oder Teilen davon – entsteht.
Ebenso wenig übernehmen Autor(en, Herausgeber) und Verlag die Gewähr dafür, dass die beschriebenen Verfahren usw. frei von Schutzrechten Dritter sind. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Bibliografische Information der Deutschen Nationalbibliothek:
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Dieses Werk ist urheberrechtlich geschützt.
Alle Rechte, auch die der Übersetzung, des Nachdruckes und der Vervielfältigung des Buches, oder Teilen daraus, vorbehalten. Kein Teil des Werkes darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren) – auch nicht für Zwecke der Unterrichtsgestaltung – reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

© 2019 Carl Hanser Verlag München
Internet: www.hanser-fachbuch.de
Lektorat: Julia Stepp
Herstellung: Björn Gallinge
Einbandrealisierung: Max Kastopoulos
Satz: Kösel Media GmbH, Krugzell
Druck und Bindung: NEOGRAFIA, a.s., Martin-Priekopa (Slowakei)
Printed in Slovakia

Print-ISBN: 978-3-446-45661-7
E-Book-ISBN: 978-3-446-45980-9
ePub-ISBN: 978-3-446-46160-4

Inhaltsverzeichnis

1 Einführung 1

1.1 Worum geht es in diesem Buch? 3
1.2 Material zum Buch 4
1.3 Für wen ist dieses Buch geeignet und für wen nicht? 4
1.4 Wichtige Hinweise 5
1.5 Wer sind wir? 6

2 Grundlagen 7

2.1 Prototyping und Testaufbauten 7
2.1.1 Breadboarding 8
2.1.2 Software zur Schaltplanerstellung 9
2.1.2.1 Fritzing 9
2.1.2.2 Virtual Breadboard 11
2.2 Elektrotechnische Grundlagen 12
2.2.1 Begriffserklärungen und Definitionen 12
2.2.2 Vorsichtsmaßnahmen im Umgang mit Spannungen 13
2.2.3 Statische Aufladung vermeiden 14
2.2.4 Ohmsches Gesetz 14
2.3 Netzwerktechnik 15
2.3.1 WLAN 15
2.3.2 GSM/3G/LTE 17
2.3.3 ZigBee 17
2.3.4 Z-Wave 18
2.3.5 Bluetooth Low Energy 19
2.4 Programmiersprachen 21
2.4.1 C/C++/Arduino C 21
2.4.2 Python 3 22

2.4.3 JavaScript/Node.js ... 24
2.4.4 Grafische Programmiersprachen ... 25

3 Sicherheitsaspekte ... 27

3.1 Sicherheit: Security vs. Safety ... 28
3.2 Security-Analyse am Beispiel des Raspberry Pi ... 28
3.3 Security Best Practices ... 31
3.3.1 Zugriffsbeschränkungen ... 31
3.3.2 Verschlüsselung ... 32
3.3.3 Sichere Programmierung ... 34
3.4 Sicherer Zugriff über das Internet ... 35
3.4.1 Dynamic DNS statt statischer IP ... 36
3.4.2 Port Forwarding ... 36
3.4.3 VPN ... 37
3.4.4 Indirekter Zugriff über Server von Dritten ... 39

4 Plattformen, Schnittstellen und Komponenten ... 41

4.1 Plattformen ... 41
4.1.1 Historie ... 42
4.1.2 Arduino ... 43
4.1.3 Raspberry Pi ... 47
4.1.3.1 Varianten ... 47
4.1.3.2 Hardwareaufbau ... 50
4.1.3.3 Schnittstellen ... 50
4.1.3.4 Installation und Inbetriebnahme ... 51
4.1.3.5 Auf einen Blick ... 62
4.1.4 ESP8266 ... 63
4.1.4.1 Flasher-Schaltungen ... 64
4.1.4.2 Nutzung der Arduino-IDE ... 65
4.1.5 ESP32 ... 67
4.1.6 Calliope mini ... 69
4.2 Schnittstellen ... 71
4.2.1 SPI ... 71
4.2.2 I2C ... 72
4.2.3 UART ... 74
4.3 Komponenten ... 75
4.3.1 LEDs ... 76
4.3.2 Smarte LEDs: NeoPixel & Co. ... 78
4.3.3 Widerstände ... 79

4.3.4 Schalter/Buttons ... 81
4.3.5 A/D-Wandler ... 81
4.3.6 Temperatur- und Feuchtigkeitssensoren ... 82
4.3.7 Motoren ... 83
4.3.7.1 Gleichstrommotoren ... 84
4.3.7.2 Schrittmotoren ... 86
4.3.7.3 Servomotoren ... 88
4.3.8 Kamera, Mikrofon, Lautsprecher & Co. ... 90

5 Projekte ... 93

5.1 Digitale Spardose ... 93
5.1.1 Einführung ... 93
5.1.2 Exkurs: IOTA ... 94
5.1.2.1 IOTA-Einrichtung ... 95
5.1.2.2 Einsatzzwecke ... 99
5.1.3 Benötigte Komponenten ... 100
5.1.4 Hardwareaufbau ... 100
5.1.5 Software ... 101
5.1.5.1 IOTA-API ... 101
5.1.5.2 IOTA-Kontostand abfragen ... 101
5.1.5.3 Spardosenanwendung auf dem ESP8266 ... 101
5.1.6 Fertiges Programm der Spardose ... 102
5.1.7 Fertige Umsetzung der Spardose ... 107
5.1.8 Offene Punkte ... 108
5.1.9 Ausblick und Alternativen ... 108
5.2 Mobile Temperaturmessung ... 110
5.2.1 Einführung ... 110
5.2.2 Exkurs: Hologram.io ... 111
5.2.3 Hardwareaufbau ... 112
5.2.4 Software ... 114
5.2.4.1 Vorbereiten und Einrichten des Surfsticks unter Linux ... 114
5.2.4.2 Konfiguration der Hologram-Plattform/Routing ... 114
5.2.4.3 Python-Skript zur Temperaturmessung und Datenversand ... 116
5.2.5 Temperaturmessung im Einsatz ... 118
5.3 Fitnesstrainer ... 118
5.3.1 Einführung ... 118
5.3.2 Benötigte Komponenten ... 119

5.3.3 Software 119
5.3.3.1 Don't move 121
5.3.3.2 Keep your balance 126
5.3.4 Ausblick 131
5.4 Word Clock 132
5.4.1 Einführung 132
5.4.2 Hardwareaufbau 133
5.4.2.1 Das Gehäuse 133
5.4.2.2 Das Ziffernblatt 133
5.4.2.3 Lichttrenner/Lichtgitter und Zwischenplatte 135
5.4.2.4 LEDs 136
5.4.2.5 Stromversorgung und Verkabelung 139
5.4.2.6 Arduino 140
5.4.2.7 DS3231 Real Time Clock 140
5.4.3 Software 141
5.4.4 Alternative: Raspberry Pi, Display und HTML 147
5.4.5 Alternative: LED-Punktuhr mit dem Raspberry Pi Zero 150
5.5 Smartes Türschloss 152
5.5.1 Einführung 152
5.5.2 Hardware 152
5.5.3 Software 157
5.5.4 Ausblick und Erweiterungen 159
5.6 Smart Mirror 159
5.6.1 Einführung 159
5.6.2 Hardwareaufbau 160
5.6.2.1 Gehäuse und Spiegel 160
5.6.2.2 Raspberry Pi und Display 161
5.6.3 Software 162
5.7 Smarter Adventskalender 165
5.7.1 Einführung 166
5.7.2 Hardwareaufbau 166
5.7.3 Umsetzung und benötigte Komponenten 168
5.7.4 Software 171
5.8 Smarter Kühlschrank 173
5.8.1 Einführung 173
5.8.2 Benötigte Komponenten 173
5.8.3 Hardwareaufbau 174

5.8.4 Software ... 176
5.8.4.1 Kalibrierung ... 176
5.8.4.2 Telegram-API und Bot-Erstellung ... 180
5.8.4.3 Milch-Tracker ... 182
5.8.5 Ausblick ... 188

6 Smart Home-Plattformen ... 189

6.1 Einführung und Übersicht ... 189
6.1.1 MQTT – das IoT-Protokoll ... 191
6.1.1.1 MQTT Broker Mosquitto: Installation und Konfiguration ... 197
6.1.1.2 TLS-Verschlüsselung ... 198
6.1.1.3 Let's Encrypt ... 200
6.1.2 MQTT-Sensor als Grundlage ... 202
6.2 Home Assistant ... 205
6.2.1 Installation ... 206
6.2.2 Einrichtung des MQTT-Sensors ... 209
6.2.3 Weitere Features ... 212
6.2.4 Auf einen Blick ... 213
6.3 FHEM ... 213
6.3.1 Installation ... 213
6.3.2 Einrichtung des MQTT-Sensors ... 216
6.3.3 Weitere Features ... 219
6.3.4 Auf einen Blick ... 220
6.4 openHAB ... 221
6.4.1 Installation ... 221
6.4.2 Einrichtung des MQTT-Sensors ... 226
6.4.3 Weitere Features ... 232
6.4.4 Auf einen Blick ... 233
6.5 ioBroker ... 233
6.5.1 Installation ... 234
6.5.2 Einrichtung des MQTT-Sensors ... 236
6.5.3 Weitere Features ... 241
6.5.4 Auf einen Blick ... 244

Stichwortverzeichnis ... 245

1 Einführung

Die Cloud und das Internet der Dinge (Internet of Things, IoT) sind seit einigen Jahren die beherrschenden Themen in der IT-Branche und versprechen ein immenses wirtschaftliches Potential. Es handelt sich dabei jedoch um keine neue Erfindung. Bereits vor circa 20 Jahren wurde die Idee geboren, „Dinge“ miteinander zu vernetzen und so den Menschen bei seinen Tätigkeiten zu unterstützen. Diese Dinge können Geräte oder Sensoren sein und Daten untereinander austauschen. Ziel von IoT ist es, die reale und virtuelle Welt zu verbinden. Namen gab und gibt es dafür viele, z. B. Ubiquitous Computing, Ambient Intelligence und Physical Computing, um nur ein paar zu nennen. Auch der aktuelle Trend der Wearables ist nichts wirklich Neues[1].

Durch die Verfügbarkeit von Internet mit entsprechender Bandbreite (via WLAN und LTE) sowie durch deutlich geringere Kosten für passende Hardware können heutzutage zahlreiche Geräte mit Internet aus- bzw. nachgerüstet werden. Somit ist es möglich, eine große Anzahl von „Dingen“ über das Internet in vernetzte Prozesse und Entscheidungen einzubeziehen. In ähnlicher Form wird diese Art der Informationsgewinnung bereits täglich genutzt. Wetterdaten werden ebenfalls durch Sensoren bereitgestellt und stehen für die Weiterverarbeitung zur Verfügung (Unwetterwarnungen, Veranstaltungstipps, gezielte Werbung an heißen Tagen etc.).

Das Internet der Dinge geht jedoch noch einen Schritt weiter und bietet die Möglichkeit, auch Daten enger definierter räumlicher Bereiche zu beziehen. Diese sogenannten Domänen können sich auch auf Personen beziehen. Somit kann die Datengrundlage für Prozesse besser lokal fokussiert und enger gefasst werden. So kann zum Beispiel die gemessene Temperatur an einer Maschine in einer Werkshalle zur gemessenen Außentemperatur am Standort in Relation gesetzt werden. Das Ergebnis liefert deutlich spezifischere Ergebnisse, als wenn z. B. die Temperatur in der nächstgelegenen Stadt als Bezugsgröße verwendet worden wäre. Dasselbe gilt natürlich auch für unser Zuhause. Wurde früher ein zent-

[1] Die Jacke mit integriertem 128 MB-MP3-Player aus dem Jahr 2004: *https://www.infineon.com/cms/de/about-infineon/press/market-news/2004/132017.html*

raler Sensor für die Heizungssteuerung verwendet, besitzt nun jeder Raum seinen eigenen und regelt die Temperatur individuell.

Bei automatisierten Prozessen ist zum Beispiel folgendes Szenario denkbar: Ein Sensor zur Erkennung eines Füllstands registriert, dass ein Mindestwert unterschritten ist. Damit ist klar, dass der Vorrat nicht mehr lange reichen wird. Daraufhin wird automatisch eine Bestellung beim Lieferanten generiert und elektronisch verschickt. Das beliebteste Beispiel ist der intelligente Kühlschrank, der erkennt, dass die Milch zu Neige geht und diese automatisch bestellt oder zumindest als Posten auf dem Einkaufszettel einträgt (siehe Abschnitt 5.8).

Im Bereich Sensoren ist auch denkbar, dass sich diese „absprechen“ oder selbst Recherchen einleiten können. Ein mögliches Beispiel: Mehrere Sensoren werden in einem Raum, zum Beispiel einem Rechenzentrum, verteilt. Diese Geräte messen periodisch die Temperatur. Der Sensor auf der Südseite registriert zur Mittagszeit einen deutlichen Temperaturanstieg, der der direkten Sonneneinstrahlung geschuldet ist. Durch gezielte Kommunikation mit den anderen Sensoren wird die Durchschnittstemperatur ermittelt. Des Weiteren werden die aktuelle amtliche Außentemperatur und die Wetterdaten über das Internet bezogen. Mithilfe dieser Werte kann geschlussfolgert werden, dass nur ein Temperatursensor eine Abweichung festgestellt hat, da draußen gerade die Sonne scheint. Dadurch kann ein Fehlalarm verhindert werden.

Anstatt Daten lokal am Sensor auszuwerten, werden die Daten von vielen Sensoren zentral gesammelt, in der Cloud kombiniert und ausgewertet. Sämtliche großen Cloud-Provider wie AWS, Google oder Microsoft Azure haben mindestens ein IoT Board im Programm oder ermöglichen eine Integration von IoT-Lösungen. Für den Privatanwender werden die Daten hingegen oft in einer Smart Home-Plattform gesammelt und ausgewertet, z. B. zur Steuerung der Heizung.

Die Umsetzung dieser Szenarien ist technisch schon länger möglich, jedoch ermöglichen der rapide Preisverfall der Bauteile und der technologische Fortschritt einen kostengünstigen und zuverlässigen Einsatz. Selbst komplexe Bauteile, wie z. B. ein GPS-Modul, können für wenige Euro bezogen und in einem Projekt verbaut werden.

Genau diesen Umstand machen wir uns in diesem Buch zunutze. Wir werden dir zeigen, wie du deine Geräte mit preiswerter Hardware wie Arduino, Raspberry Pi, ESP8266, Calliope & Co. smart machst.

1.1 Worum geht es in diesem Buch?

Wenn man es genau nimmt, könnte man sagen: Man kann gut ohne die in diesem Buch vorgestellten IoT Gadgets leben. Doch von einer Sache sind wir fest überzeugt: Sie bringen jede Menge Spaß – und mit Spaß lernt es sich am besten. Anhand von spannenden Beispielprojekten möchten wir dir fundiertes, aber stets praxisorientiertes Wissen zum Thema Internet der Dinge bzw. Internet of Things (IoT) vermitteln. Natürlich funktioniert das nicht ganz ohne Theorie.

In Kapitel 2 erlernst du deshalb die wichtigsten Grundlagen der Elektrotechnik, der Vernetzung und der Schaltplanerstellung. Außerdem enthält das Kapitel einen Überblick über die zur Verfügung stehenden Netzwerktechnologien.

In Kapitel 3 erfährst du, wie du deine Projekte absicherst, um nicht Opfer eines Angriffs zu werden. Dazu wirst du auch kurz die Brille eines Angreifers aufsetzen.

In Kapitel 4 geben dir einen Überblick, welche Mikrocontroller, Einplatinencomputer, Sensoren und Smart Home-Devices du für deine IoT-Projekte verwenden kannst, wie du die jeweilige Hardware einsetzt und wie du deine Smart Gadgets mit der dazugehörigen Software programmierst.

In Kapitel 5 stellen wir acht smarte Beispielanwendungen vor, die dir Anregungen für eigene Ideen liefern sollen. Diese Projekte haben wir alle schon selbst umgesetzt. Teile davon sind tatsächlich täglich im Einsatz und bringen jede Menge Spaß. Für die Umsetzung der Projekte liefern wir eine Teile- und Einkaufsliste mit. Den Programmcode findest du als kommentiertes Listing im Buch und auch zum Download unter *https://github.com/stephanhuewe/iotgadgets*.

Um unsere Projekte miteinander zu vernetzen zu können, schauen wir uns in Kapitel 6 die vier gängigsten Smart-Home-Plattformen im Detail an und installieren jede davon als mögliche Schaltzentrale für unser „IoT@Home“.

Was die Programmiersprachen angeht, haben wir uns sehr breit aufgestellt. Python, Arduino C, Shell-Skripte und auch eine grafische Programmiersprache kommen zum Einsatz. Wir sind keine Profis in allen Programmiersprachen, sondern haben immer die Sprache gewählt, die uns schnell ans Ziel bringt. Mit einem Grundverständnis in einer Hochsprache ist es aus unserer Sicht möglich, die Programmbeispiele zu verstehen und gegebenenfalls anpassen zu können. Die Listings und die dazugehörigen Erklärungen helfen dabei.

Dieses Buch ist jedoch nicht nur hinsichtlich der Programmiersprachen breit aufgestellt. Auch in anderen Bereichen versuchen wir einen Überblick zu geben, was alles möglich bzw. am Markt vorhanden ist. Wir schauen dabei immer über den Tellerrand hinaus und geben Hinweise auf weitere Ideen, die man noch umsetzen könnte. Dabei gehen wir eher in die Breite anstatt uns in Details zu verstricken, auch wenn wir vermutlich über viele der behandelten Themen ein eigenes Buch hätten schreiben können.

An erster Stelle wollen wir dich inspirieren, eigene Ideen und Lösungen umzusetzen. Den passenden Baukasten hältst du mit diesem Buch in den Händen.

1.2 Material zum Buch

Sämtliche Codebeispiele und Projekte aus diesem Buch sind bei GitHub hinterlegt und können dort kostenlos heruntergeladen werden. Die Projektseite findest du unter *https://github.com/stephanhuewe/iotgadgets*. Sofern notwendig, werden diese Projekte kontinuierlich verbessert und überarbeitet. Auf GitHub findest du auch Erweiterungen, die es nicht mehr ins Buch geschafft haben, sowie Errata, falls sich bei einem Projekt der Fehlerteufel eingeschlichen hat. Wir freuen uns auch über Forks (Abspaltungen) deiner eigenen Projekte.

1.3 Für wen ist dieses Buch geeignet und für wen nicht?

Dieses Buch eignet sich für alle, die sich für das Internet der Dinge interessieren, experimentierfreudig sind und ein wenig Programmiererfahrung mitbringen. Technisches Verständnis und ein wenig Geschick schaden sicher auch nicht. Für alles Weitere vermitteln wir die notwendigen Grundlagen. Uns ist aber auch klar, dass ein Buch niemals alle Themenbereiche vollständig abdecken kann. An Stellen, an denen eine umfassende Erläuterung zu weit führen würde, möchten wir dich trotzdem nicht im Regen stehen lassen. Wir öffnen stets Türen für Lösungsansätze und weiterführende Informationen und Ideen. So findest du hoffentlich immer eine geeignete Lösung für die Realisierung deiner Projekte. Nicht zuletzt entwickelt sich dieses Themenfeld natürlich stetig weiter, weshalb wir keine Zeitlosigkeit der Inhalte gewährleisten können.

Dieses Buch ist ein Rundumschlag zu den Themen IoT, Smart Gadgets und Smart Home. Solltest du also auf der Suche nach einem Buch sein, das in die Arduino-Programmierung einführt oder dir zeigt, wie du das Letzte aus deiner Smart Home-Installation herausholst, raten wir zu spezieller Literatur über das jeweilige Thema. Dieses Buch hingegen soll ein Ratgeber durch den IoT-Dschungel sein, der dir dabei hilft, die für dich passende Lösung für die Realisierung deiner Projekte zu finden.

Dieses Buch ist auch keine Einführung in die Programmierung, obwohl wir natürlich versucht haben, die Einstiegshürden so gering wie möglich zu halten, und jedes Listing genauestens kommentieren.

Da wir sehr viel Spaß bei der Erstellung unserer Projekte hatten und im täglichen Betrieb immer noch haben, gehen wir die Themen sehr locker an. Den ein oder anderen Witz konnten wir uns also nicht verkneifen. Wir haben zumindest versucht, ihn in den Fußnoten zu verstecken. Solltest du also auf der Suche nach trockener und sachlicher Literatur sein, dann ist dieses Buch vermutlich nichts für dich.

1.4 Wichtige Hinweise

Wir gehen davon aus, dass du ein neugieriger Charakter bist, der gerne experimentiert. Neben der Programmierung hat dieses Buch logischerweise auch einen deutlichen Hardwarebezug. Von daher gibt es auch viele Projekte, die mit elektrischen Schaltungen und damit einer Stromquelle zu tun haben. Das könnte vielleicht in ein oder anderer Hinsicht Neuland für Dich sein.

Deshalb gilt der Grundsatz: Der leichtsinnige Umgang mit Strom ist immer gefährlich. Die Netzspannung mit 220 V ist generell tabu, doch auch niedrigere Spannungen können gefährlich sein. Von daher der dringende Rat: Sollte dir etwas unklar sein, dann mache nicht weiter, sondern suche den Rat eines Fachkundigen. Überprüfe außerdem vor jedem Testlauf gewissenhaft den Aufbau auf mögliche Gefahren.

Wir haben uns bemüht, unsere Vorgehensweisen und Bezugsquellen möglichst genau zu beschreiben. Die Welt da draußen dreht sich allerdings schnell weiter. Von daher kann es gut sein, dass sich einige der hier ausgeführten Sachverhalte in der Zwischenzeit geändert haben. Auch verschwinden Links und Webseiten aus dem Netz. Hersteller kaufen oder werden aufgekauft. Sollte ein Inhalt nicht mehr erreichbar sein, empfehlen wir dir, gezielt nach der URL zu suchen. Darüber hinaus ist *www.archive.org* ein guter Anlaufpunkt, sofern die Website archiviert wurde. Sollte ein Bauteil im Baumarkt deines Vertrauens nicht mehr erhältlich sein, empfehlen wir bei den gängigen Onlinehändlern (Amazon, eBay, Conrad, Voelkner, Watterott) zu suchen oder anzufragen.

Wir haben uns bemüht, alle wichtigen Details sehr genau zu beschreiben. Gleichzeitig möchten wir nicht deine Kreativität beschneiden. Vielmehr möchten dich ermutigen, unsere Projekte nach deinen Wünschen abzuwandeln und zu erweitern. Sollte dir etwas partout nicht gelingen, darfst du uns jederzeit gerne kontaktieren. Wir versuchen dir dann im Rahmen unserer Möglichkeiten weiterzuhelfen.

1.5 Wer sind wir?

Bevor wir uns ins Thema stürzen, möchten wir dir noch ein bisschen etwas über uns erzählen. Die Namensgleichheit lässt es erahnen: Wir sind Brüder. Auch darüber hinaus gleicht sich unser Lebenslauf in einigen Punkten.

Dipl.-Inf. (FH) Stephan Hüwe

Ich bin selbstständiger Softwareentwickler, Coach und Consultant sowie Lehrbeauftragter für Elektrotechnik/Informatik an der Hochschule Augsburg. Ich habe bei Hanser bereits *Raspberry Pi für Windows 10 IoT Core* (ISBN 978-3-446-44719-6) veröffentlicht.

Dipl.-Inf. (FH) Peter Hüwe

Ich bin Senior Staff Engineer bei der Infineon Technologies AG und kümmere mich dort um die Sicherheit von Embedded Systems, elektronischen Reisepässen und Trusted Platform Modules. Außerdem bin ich Lehrbeauftragter für Python, C/C++ und Embedded Linux an der Hochschule Augsburg.

Du kannst dir sicher vorstellen, dass es in einer Techie-Familie computertechnisch hoch her geht. Wir sind ständig am Experimentieren, Bauen und Basteln – manchmal zum Leidwesen des einen oder anderen Familienmitglieds. An dieser Stelle möchten wir uns noch einmal recht herzlich bei unseren Familien bedanken, die uns während der Erstellung dieses Buches unterstützt haben, insbesondere bei Bine, Sonja und natürlich Oma. Sonja Hemmersbach danken wir zudem für die Erstellung einiger Fotografien in diesem Buch.

Abschließend bleibt uns nur noch zu sagen: Zögere nicht, uns zu kontaktieren, wenn du Fragen hast. Idealerweise meldest du dich per Mail unter iotgadgets@huewe.info. Auf diesem Wege antworten wir meist innerhalb kürzester Zeit.

Und nun wünschen wir dir viel Freude bei der Lektüre unseres Buches!

Gersthofen/Augsburg, im Februar 2019

Peter Hüwe

Stephan Hüwe

2 Grundlagen

Was du in diesem Kapitel erfährst:

- Wie lassen sich Schaltungen sauber dokumentieren?
- Grundlagen der Elektrotechnik
- Welche Netzwerkprotokolle stehen zur Verfügung, um Gadgets smart zu machen?
- Einführung in Bluetooth Low Energy (BLE)
- Übersicht über die im Buch eingesetzten Programmiersprachen

Wie die Überschrift bereits verrät, behandelt dieses Kapitel Grundlagenthemen. Solltest du bereits Vorkenntnisse in den genannten Themenbereichen besitzen, kannst du dieses Kapitel überspringen und bei Bedarf hier nachschlagen, falls du bei einem deiner Projekte nicht weiterkommst.

2.1 Prototyping und Testaufbauten

Um intelligente Gegenstände für dein Zuhause entwickeln zu können, benötigst du einiges an Hard- und Software. Dazu zählen z. B. ein Arduino-Board, die dazugehörige Software zur Programmierung und diverse Komponenten, wie Sensoren, Schalter etc. (siehe Kapitel 4). In der Entwicklungsphase eines Projekts solltest du jedoch erst einmal mit Prototypen arbeiten. In diesem Abschnitt erfährst du, welche Hard- und Software du für Testaufbauten verwenden kannst.

2.1.1 Breadboarding

Während der Entwicklung und für den Hardwareaufbau von Prototypen und empfiehlt es sich, Breadboards zu verwenden. Bild 2.1 zeigt den schematischen Aufbau eines solchen Steckbretts. Es handelt es sich hierbei um eine Steckplatine aus Kunststoff mit zahlreichen Kontaktfedern, auf der elektronische Bauteile ohne Löten befestigt werden können. Diese müssen einfach nur eingesteckt werden. Dies hat den Vorteil, dass Fehlerkorrekturen und Umbauten leicht umgesetzt werden können. Wurde die Schaltung ausgiebig getestet und soll so beibehalten werden, spricht nichts dagegen, sie fest zusammenzulöten. Hierfür eignen sich Lochrasterplatinen besser als Breadboards.

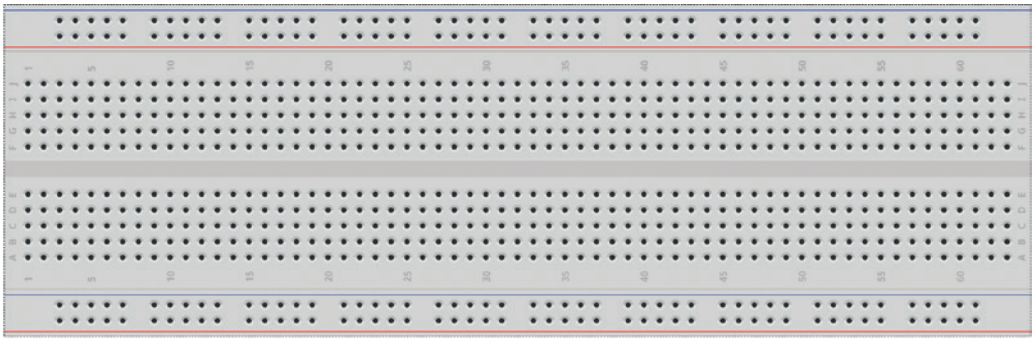


Bild 2.1 Schematischer Aufbau eines Breadboards (hier eine Darstellung der Software Fritzing)

Der Abstand zwischen den Kontaktpunkten beträgt – z. B. analog zu den GPIO-Pins auf einem Raspberry Pi – in der Regel 2,54 mm (0,1 Inch) als Normwert. Fast alle Bauteile folgen diesem Raster und können somit mühelos in die Reihen gesteckt werden. Die Kontaktpunkte sind untereinander über Netze intern verbunden, sodass einzelne Bauteile ohne weitere Drähte vernetzt werden können. Die Aufteilung der Netze kann je nach Breadboard-Typ variieren, lässt sich jedoch optisch leicht erkennen. Die Breadboards sind in unterschiedlichen Größen erhältlich.

Beginne zunächst mit Breadboards mittlerer Größe. Große Platinen lassen sich nämlich nur schwer verbauen. Kleine Boards hingegen bieten oftmals nicht ausreichend Platz, um die Gestaltung eines übersichtlichen Layouts zu ermöglichen. Bist du dir bezüglich der Netzgestaltung nicht sicher, empfiehlt es sich, zur Sicherheit nachzumessen, um Beschädigungen an Bauteilen zu vermeiden.

Gibt es keine Verbindung über das interne Netz oder müssen weitere Strecken überbrückt werden, so bietet sich die Verwendung von Drahtbrücken oder Litzen an. Diese sollten idealerweise isoliert sein. Im Handel gibt es hierfür sogenannte Jumper Wires (Bild 2.2).

Bild 2.2 Jumper Wires

Die Verwendung von Breadboards hat jedoch auch einige Nachteile. Durch das einfache Stecken von Bauteilen wird keine dauerhafte Verbindung hergestellt. Dies führt dazu, dass es zu Fehlkontakten kommt, die die Fehlersuche erschweren. Darüber hinaus hat das Breadboard eine schlechte Wärmeabfuhr, sodass Bauteile sehr warm werden können. Die Strombelastbarkeit ist ebenfalls eingeschränkt und liegt in der Regel bei ca. 1 A (Ampere).

2.1.2 Software zur Schaltplanerstellung

Es ist sinnvoll, sich vor der Umsetzung Gedanken über die Struktur und den Aufbau eines Projekts zu machen. Schaltungen, die ohne Vorplanung auf Breadboards umgesetzt werden, neigen dazu, schnell unübersichtlich und schlecht wartbar zu werden. Deshalb gibt es Softwaretools, mit denen sich Schaltungen bereits vorab virtuell zusammenstellen lassen. Im professionellen Umfeld gibt es etablierte Lösungen, mit denen sich komplexe Schaltungslayouts erstellen lassen, wie z. B. EAGLE der Firma Autodesk. Diese Werkzeuge sind sehr mächtig, jedoch für Einsteiger und Kleinprojekte meist zu komplex. Dieses Buch beschränkt sich bei der Auswahl der Werkzeuge auf zwei Produkte, die auch für Einsteiger geeignet sind: Fritzing und Virtual Breadboard.

2.1.2.1 Fritzing

Bei Fritzing handelt es sich um eine Open-Source-Initiative, deren Ursprung auf ein Projekt der FH Potsdam zurückgeht. Die Idee ist es, eine freie Software anzubieten, die bei der Entwicklung von elektronischen Projekten unterstützt. Neben der Möglichkeit, Leiterplatten und Schaltpläne zu erstellen, bietet Fritzing eine Breadboard-Ansicht an. In der mitgelieferten Bibliothek findest du zahlreiche Komponenten aus der Welt der Elektronik, von der LED bis hin zum Raspberry Pi. So kannst du deine Schaltung virtuell am PC zusammenstecken, bevor du sie baust.

Ein Großteil der im Buch gezeigten Schaltungen wurde mit Fritzing erstellt oder basiert darauf. Sofern nicht explizit gekennzeichnet, erkennst du Fritzing-Zeichnungen recht einfach am Comicstil. Die Lizenz von Fritzing und der Teile erlaubt ausdrücklich das Erstellen und Veröffentlichen von Schaltungen, sofern du auf Fritzing hinweist und auch unter gleicher Lizenz veröffentlichst. Wir danken an dieser Stelle dem Fritzing-Projekt und allen Mitwirkenden für dieses tolle Tool.

Du kannst Fritzing kostenfrei unter *http://fritzing.org/home* herunterladen. Fritzing liefert bereits im Standard eine große Anzahl von Bauteilen mit. Sollte dir ein Bauteil fehlen, liefern einige Hersteller zu ihren Bauteilen auch passende Fritzing-Elemente, die du in die Software importieren kannst. Auch wenn die letzte Version von Fritzing im Juni 2016 veröffentlicht wurde, wird die Teilebibliothek regelmäßig aktualisiert und neue Bauteile werden hinzugefügt. Fritzing lädt diese Aktualisierung bei jedem Start nach.

Fritzing ist nahezu selbsterklärend. Fritzing liefert auf seiner Website zahlreiche Tutorials für den tieferen Einstieg. An dieser Stelle soll daher nur eine kurze Einführung gegeben werden, damit du direkt beginnen kannst. Die Software ist grob in zwei Bereiche aufgeteilt: in den Arbeitsbereich in der Mitte und in den Bauteilebereich auf der rechten Seite (Bild 2.3). In der Bauteileübersicht kannst du das gewünschte Bauteil suchen und dann mit Drag & Drop auf die Steckplatine ziehen und dort entsprechend verkabeln. Über das Eigenschaftenfenster rechts unten lassen sich die Bauteile noch im Detail weiter konfigurieren.

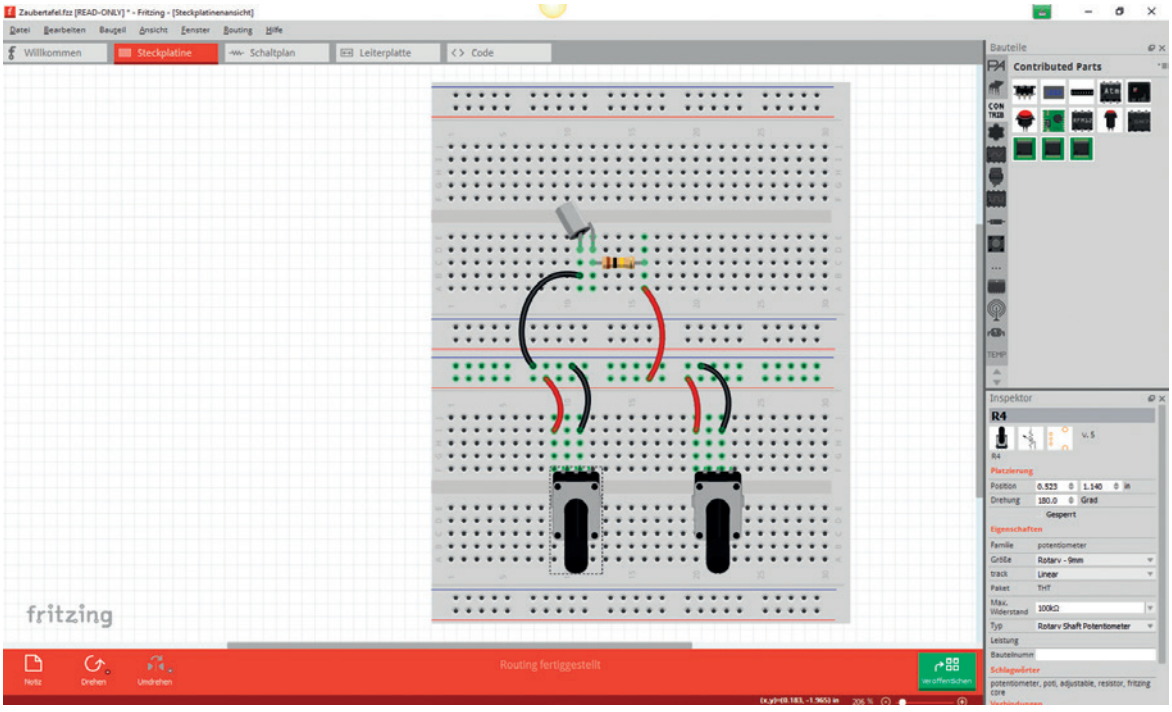


Bild 2.3 Bedienoberfläche von Fritzing

Neben der Steckplatinen-Ansicht kannst du dir die erstellte Schaltung auch als Schaltplan oder Leiterplatte anzeigen lassen (Bild 2.4). Fritzing bietet noch viele weitere nützliche Funktionen, wie z. B. das Veröffentlichen von Schaltungen oder die Direktbestellung von Leiterplatten. Schaltungen lassen sich bequem in gängige Formate wie PDF oder PNG exportieren und somit weitergeben oder wie in diesem Buch zur Illustration abdrucken.

Bild 2.4 Wechsel des Ansichtsmodus in Fritzing

2.1.2.2 Virtual Breadboard

Ein weiteres hilfreiches Tool ist Virtual Breadboard (VBB). Es handelt sich dabei um ein ähnliches Werkzeug wie Fritzing, wie bereits der Aufbau der Software vermuten lässt (Bild 2.5). Im Gegensatz zu Fritzing steht VBB unter keiner freien Lizenz und bietet einen Modus zur Simulation und zum Debugging von Schaltungen. Das Tool kannst du unter *http://www.virtualbreadboard.com* herunterladen.

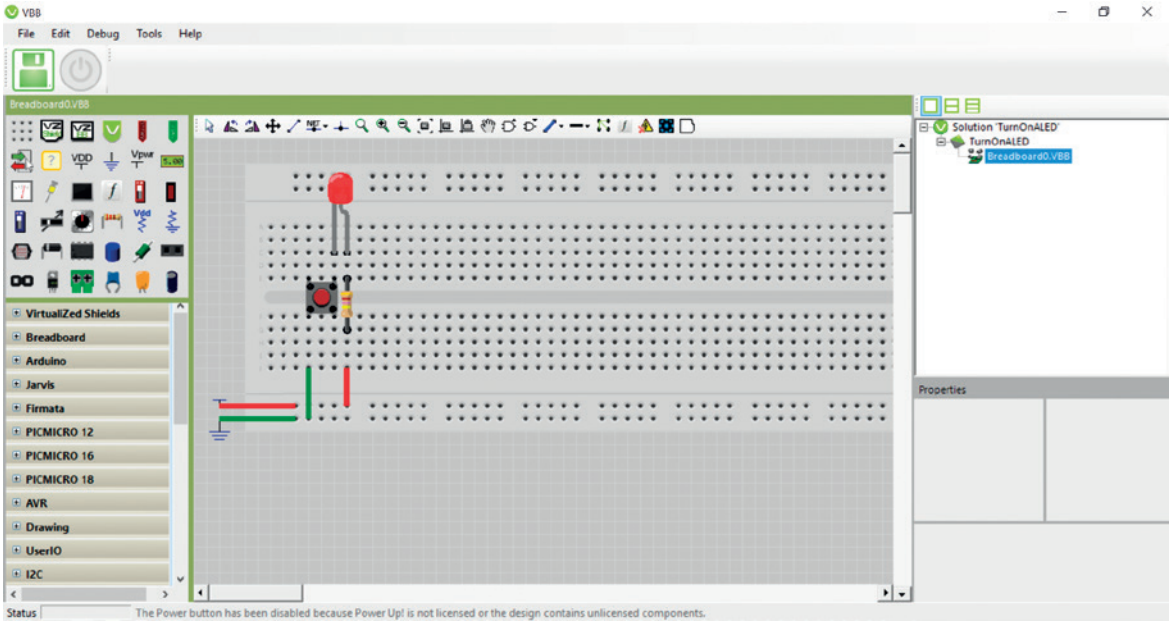


Bild 2.5 Projektansicht bei Virtual Breadboard

2.2 Elektrotechnische Grundlagen

Da wir in den Beispielprojekten dieses Buches auch Hardware ansprechen möchten, benötigst du einige elektrotechnische Grundlagen. Dieser Abschnitt fasst – ohne lange Herleitung oder theoretische Ausführungen – die wichtigsten Informationen und Konzepte für dich zusammen. Dieses Wissen reicht aus, um die Projekte im Buch nachvollziehen zu können.

2.2.1 Begriffserklärungen und Definitionen

Um sich im Kauderwelsch der Produktdatenblätter zurechtzufinden, enthält die folgende Tabelle eine Übersicht der wichtigsten Begriffe, die einem darin begegnen.

Begriff	Erklärung
VCC	Versorgungsspannung (innerhalb von Schaltungen meist 5 V, 3,3 V oder 1,8 V)
GND	Ground/Masse (0 V), das Gegenstück zu VCC
CLK	Clock (Eingangstakt)
NC	Not Connected (nicht angeschlossen): Diese Pins werden einfach ignoriert.
GPIO	General Purpose Input/Output (Ein-/Ausgang)
#RESET	Reset-Leitung, die das System zurücksetzt
High „1“	Ein Signal wird als „High“ bzw. logische 1 erkannt, wenn eine hohe Spannung im Vergleich zu VCC anliegt (z. B. ≥ 60 % von VCC).
Low „0“	Ein Signal wird als „Low“ bzw. logische 0 erkannt, wenn eine geringe Spannung im Vergleich zu VCC anliegt (z. B. ≤ 15 % von VCC).
Floating	Sollte ein Wert zwischen den Grenzen von „High“ und „Low“ anliegen, ist das Verhalten meist undefiniert, d. h., mal liegt „High“, mal „Low“ an. Dies kann z. B. vorkommen, wenn eine Spannung einbricht oder nicht verbunden ist.
Tri-State „Z“	Neben den Zuständen „High“ und „Low“ kann ein Signal auch noch den Tri-State-Zustand „Z“ annehmen (auch oft als „hochohmig“ bezeichnet). Dem Ausgang ist hierbei ein extrem hoher Widerstand vorgeschaltet, sodass er effektiv das Signal nicht mehr beeinflussen kann.
Active Low, #SIGNAL	Manche Signale haben eine invertierte Logik. Das bekannteste Beispiel hierfür ist #RESET: Erkennt ein #RESET-Eingang eine logische 0, wird der Reset ausgelöst. Liegt eine logische 1 an, passiert nichts. Diese umgekehrte Logik erlaubt eine relativ einfache und stromsparende Anbindung mehrerer Teilnehmer an ein Signal. Diese Signale sind entweder mit einer Raute (#) oder mit einem Schrägstrich (/) vor oder über dem Signalnamen gekennzeichnet. Weitere Beispiele: #IRQ (Interrupt), #CS (ChipSelect) #CE (ChipEnable)

Begriff	Erklärung
Pull-up	Hoher Widerstand zwischen einer Signalleitung und VCC, der das Signal auf „High“ zieht. Möchte ein Teilnehmer das Signal auf „Low“ ziehen, braucht er nur kurz seinen Ausgang mit GND zu verbinden. Typische Werte: 10 kΩ, 1 kΩ
Pull-down	Genau wie Pull-up, nur eben umgekehrt
I2C, TWI	Inter-Integrated Circuit, Two Wire Interface: „langsamer“ serieller Datenbus mit den Leitungen SDA und SCL (siehe Abschnitt 4.2.2)
SPI	Serial Peripheral Interface: „schneller“ serieller Datenbus mit den Leitungen MISO, MOSI, SCLK und #CS (siehe Abschnitt 4.2.1)
UART	Auch bekannt als serielle Schnittstelle, nur mit 5 V bzw. 3,3 V (siehe Abschnitt 4.2.3)
OneWire	Sehr langsamer serieller Datenbus, jedoch mit der Besonderheit, dass man neben GND wirklich nur eine Leitung braucht (für Kommunikation und Spannungsversorgung)

2.2.2 Vorsichtsmaßnahmen im Umgang mit Spannungen

Wie die meisten technischen Geräte benötigen deine Gadgets eine Stromversorgung mit passender Spannung (Volt) und einer geeigneten Stromstärke (Ampere). Typischerweise funktionieren die meisten Boards und Schaltungen mit 1,8 V, 3,3 V oder 5 V und benötigen bis zu 1 A (meistens deutlich weniger). Sie sind damit erst einmal relativ ungefährlich.

In der Regel passiert nichts, wenn man die Pins eines Boards mit bloßen Händen anfasst, solange man mit diesen Spannungen und Stromstärken arbeitet. Dennoch sollte man dies im stromführenden Zustand nach Möglichkeit vermeiden, speziell mit feuchten Händen. Hat man mit größeren Stromstärken und höheren Spannung zu tun, z. B. im Umgang mit Relais oder Motortreibern und den dazugehörigen Steckernetzteilen, sollte deutlich mehr Vorsicht an den Tag gelegt werden.

Grundsätzlich gilt: Der Zugang zum Stromnetz/zur Netzspannung ist nicht notwendig und zudem gefährlich. Ist es unausweichlich, dass z. B. 230-V-Geräte geschaltet werden (wie z. B. Lampen und Geräte), so bietet es sich stattdessen an, Funksteckdosen zu verwenden. Diese können z. B. gefahrlos über einen 433-MHz-Funksender mit dem Raspberry Pi oder per WLAN geschaltet werden.

Generell gilt die VDE-Regel: Trennen und auf Spannungsfreiheit prüfen, bevor du Arbeiten durchführst. Speziell größere Kondensatoren können noch einiges an Ladung besitzen, auch wenn du die Stromzufuhr getrennt hast.

Kurzschlüsse und falsche Polungen von Bauteilen sind unbedingt zu vermeiden. Diese können zu Beschädigungen an Bauteilen und auch zu Bränden führen. Um Haustiere und Kinder zu schützen sowie die Schaltungen vor ungewollten Kurzschlüssen zu bewahren, empfiehlt es sich, jede Schaltung in ein Gehäuse zu verpacken. Dies erhöht auch die Wahrscheinlichkeit, dass die Schaltung im Wohnzimmer vom Partner akzeptiert wird.

2.2.3 Statische Aufladung vermeiden

Durch Kunststofffasern in Bodenbelägen und Kleidung kann man sich statisch aufladen. Das hat jeder schon einmal erlebt, wenn er an einen Türgriff fasst und einen kleinen Schlag bekommt. Bei der Berührung elektronischer Bauteile oder des Mikrocontrollers kann es dabei zu irreparablen Beschädigungen kommen, auch wenn man selbst nichts davon merkt. Im professionellen Bedarf gibt es spezielle Hilfsmittel, wie Schuhe, Armbänder und Matten, die die Aufladung verhindern sollen. Für unseren Bereich reicht es, sich kurz zu entladen, bevor du z. B. eine Schaltung zusammensteckst oder den Raspberry berührst. Hierzu genügt es, z. B. das PC-Gehäuse oder einen Heizkörper anzufassen.

2.2.4 Ohmsches Gesetz

Das wohl wichtigste Gesetz der Elektronik ist das ohmsche Gesetz.[1] Es dient als Grundlage zur Erklärung des Zusammenhangs zwischen Stromstärke, Spannung und Widerstand.

Durch eine Schaltung fließt Strom. Die Spannung, die in Volt (V) gemessen wird, regelt dabei, wie schnell dieser fließen kann, und ist die treibende Kraft der Ladungsbewegung. Der Stromfluss, also die Menge an Energie, die durch die Schaltung fließt, wird in Ampere (A) gemessen. Soll der Fluss gebremst werden, braucht es einen Gegenspieler. Beim Strom ist das der Widerstand, gemessen in Ohm (Ω). Mit Widerstand wird in der Regel das Bauteil bezeichnet. Jedoch hat auch jedes Material (z. B. ein Kupferdraht) einen spezifischen Widerstandswert.

Das ohmsche Gesetz ist eine Gleichung, die folgenden Zusammenhang herstellt:

- Der Quotient aus Spannung und Stromstärke definiert den Widerstand.
- Dieser Widerstand ist unabhängig von Spannung und Stromstärke und damit konstant.

$$Widerstand = \frac{Spannung}{Stromstärke} \rightarrow R = \frac{U}{I} \qquad \text{Formel 2.1}$$

[1] Benannt nach dem deutschen Physiker Georg Simon Ohm (1789 – 1854)

Die Abkürzungen für Spannung, Widerstand und Stromstärke lauten:

- U ist die Spannung in Volt.
- R ist der Widerstand in Ohm.
- I ist die Stromstärke in Ampere.

Durch Umformen erhält man folgende Gleichungen:

$$U = R \times I \qquad \text{Formel 2.2}$$

$$I = \frac{U}{R}$$

Diese Gleichung hilft z. B. bei der Berechnung des benötigten Widerstandes zum Betreiben einer LED. Mit zwei bekannten Werten kann der dritte jeweils berechnet werden.

In einer Schaltung fließt 0,5 A, der Widerstand beträgt 10 Ω. Welche Spannung liegt an?
Lösung: U = 10 Ω × 0,5 A = 5 V

2.3 Netzwerktechnik

Während früher noch aufwendig Kabel durch das komplette Haus gelegt werden mussten, um Dinge miteinander zu vernetzen, steht heutzutage eine Vielzahl von Standards und Technologien zur Verfügung, um deine Gadgets drahtlos zu verbinden. Damit steht dem Smart Home selbst in Altbauten nichts mehr im Wege. Obwohl alle der hier vorgestellten Technologien demselben Zweck der Vernetzung dienen, könnten sie teilweise unterschiedlicher nicht sein. Speziell beim Thema Reichweite wird dies sehr deutlich. Von ein paar Metern bis hin zu weltweitem Empfang ist alles möglich. Im Folgenden stellen wir einige wichtige Smart Home-Netzwerktechnologien kurz vor.

2.3.1 WLAN

Über WLAN braucht man heutzutage nicht mehr viele Worte zu verlieren. Fast jeder hat sein eigenes WLAN zu Hause und benutzt es täglich mit Smartphone, Tablet und Computer. Mittels WLAN Repeater ist das Zuhause meist schon perfekt ausgeleuchtet, sodass jeder Winkel im Haus zumindest akzeptablen Empfang hat. Genau dies ist der große Vorteil von WLAN für die Smart Gadgets: WLAN ist verfügbar, hat keine speziellen weiteren Anschaffungskosten und erlaubt dir, deine Smart Gadgets direkt vom Smartphone oder Tablet aus zu steuern oder Nachrichten von ihnen zu erhalten. Vom heimischen WLAN ins

Internet zu gelangen ist dann auch kein Problem mehr. Damit stehen dir alle Möglichkeiten offen.

Auch auf Gadget-Seite ist WLAN mittlerweile kostengünstig und einfach zu benutzen. Entweder rüstest du es per Shield nach oder greifst auf die oft eingebaute WLAN-Lösung (z. B. beim Raspberry Pi Zero W oder ESP8266) zurück – und schon ist dein Gadget ein vollwertiges Mitglied deines Netzwerkes mit voller Bandbreite und Übertragungsgeschwindigkeit.

Warum dann überhaupt noch auf andere Lösungen ausweichen, wenn WLAN so ideal ist? WLAN hat den entscheidenden Nachteil, dass es relativ stromhungrig ist, sowohl aufseiten der Hardware beim Senden und Empfangen als auch beim Verarbeiten der Daten im Protokoll. Somit kann es einen Akku relativ schnell leer saugen. Damit ist es für Gadgets, die nicht direkt am Stromnetz hängen, eher ungeeignet, und selbst für Geräte am Stromnetz kann sich dieser Verbrauch je nach Anzahl auf der Stromrechnung bemerkbar machen.[2]

Dass man dennoch stromsparende Gadgets mit WLAN aufbauen kann, zeigen wir in Abschnitt 5.8.4.3.

Auf einen Blick

Pro und contra:

(+) Standardtechnologie

(+) nah an Computer, Tablet und Smartphone

(+) überall vorhanden, einfach in bestehende Infrastruktur integrierbar

(+) Apps/Tools/Webserver lassen sich einfach betreiben.

(+) hohe Geschwindigkeiten/hohe Bandbreite

(–) hoher Stromverbrauch

(–) Overhead bei der Verarbeitung

(–) WLAN-Netzwerke in städtischen Gebieten sind relativ überlastet.

Datenrate:

Theoretisch bis zu mehreren Gigabit pro Sekunde (Gbit/s)

Reichweite:

Bis zu 100 m, in Gebäuden teilweise deutlich weniger

Anwendungsgebiete:

Universell einsetzbar im Heimbereich

Sonstiges:

Mit dem ESP8266 bzw. ESP32 und dem Raspberry Pi Zero W lässt sich jedes Gadget kostengünstig mit WLAN ausstatten.

[2] Ein ESP8266 verbraucht im Dauerbetrieb mit WLAN ca. 75 mA mit gelegentlichen Spitzen von 300 mA. Gehen wir von 200 mA im Durchschnitt aus, ergibt sich über das Jahr ein Verbrauch von 8 – 10 kWh, damit ca. 2 € Stromkosten (ohne Berücksichtigung des Steckernetzteils).