



Thomas
Brühlmann
5. Auflage

Arduino Praxiseinstieg

Behandelt Arduino UNO R4 und R3



UNO R4
MINIMA

IOREF

RESET

3.3V

5V

GND

VIN

POWER

ANALOG

A0

A1

A2

komplett
in Farbe

Inhaltsverzeichnis

1	Einleitung	9
1.1	5. Auflage	9
1.2	Aufbau des Buchs	9
1.3	Mehr Informationen	12
1.4	Weitere Quellen	12
1.5	Danksagung	13
2	Arduino-Plattform	15
2.1	Am Anfang war der König	15
2.2	Be a Maker	16
2.3	Arduino-Plattform	18
2.4	Hardware	19
2.5	Software	27
2.6	Installation der Software	27
2.7	Get Connected	30
2.8	Arduino-Entwicklungsumgebung	36
2.9	Arduino-Boards	45
2.10	Arduino-kompatible Boards	48
3	Startschuss	49
3.1	Das Arduino-Board	50
3.2	Steckbrett – Experimentieren ohne Löten	58
3.3	Spannung, Strom und Herr Ohm	65
3.4	Widerstand & Co	70
3.5	Programmcode	80
3.6	Testen	86
3.7	Projekt Blink	92
3.8	Projekt Wechselblinker	95
4	Eingänge und Ausgänge	99
4.1	Digitale Eingänge	100
4.2	Digitale Ausgänge	114
4.3	Analoge Welt	117

4.4	Serielle Kommunikation	131
4.5	Drahtlose Kommunikation	152
4.6	Keyboard und Maus mit USB HID (nur Arduino UNO R4)	176
4.7	Projekt: Würfel.	180
5	Sensoren	189
5.1	Sensoren	190
5.2	Projekt Kompass mit Richtungsanzeige	246
5.3	Projekt Gefrierschrankwächter (nur Arduino UNO R3)	254
5.4	Kontaktloses Fieberthermometer	257
6	Aktoren	261
6.1	Relais.	261
6.2	Servos	264
6.3	Motoren	274
6.4	Lasten schalten.	282
6.5	Projekt: Roboter mit Wii-Steuerung.	285
7	Anzeigen	299
7.1	Leuchtdiode (LED)	299
7.2	7-Segment-Anzeigen	309
7.3	LC-Display (LCD)	316
7.4	LC-Display Nokia 3310/5110	327
7.5	OLED-Display.	329
7.6	LED-Matrix	335
7.7	Projekt Geschwindigkeitsmesser fürs Fahrrad	336
7.8	Projekt Schrittzähler	345
8	Datenverarbeitung	351
8.1	Daten speichern	351
8.2	Daten ins Internet senden.	365
8.3	Projekt: Programmierbarer Signalgeber	365
8.4	Sinusgenerator.	372
8.5	Projekt Sollwerteingabe mit Drehgeber	374
9	Erweiterungen	391
9.1	Bibliotheken	391
9.2	Hardware-Erweiterungen (Shields)	400
9.3	Hardware-Adapter	409

10	Arduino im Einsatz	413
10.1	Verbindung zum Internet.	413
10.2	Verbindung zum Internet.	427
10.3	ESP-Module und Arduino.	436
10.4	XML einlesen	448
10.5	Umweltdaten sammeln.	462
10.6	Projekt Wetterstation (nur Arduino UNO R3).	471
11	Fehlersuche/Troubleshooting	493
11.1	Allgemeines Vorgehen	493
11.2	Fehler in der Schaltung.	493
11.3	Fehler im Programm.	494
11.4	Probleme mit der IDE.	495
11.5	Hallo Arduino-Board	495
12	DIY Boards und Clones	497
12.1	Boards.	497
12.2	Programmieradapter (USB-Wandler)	504
12.3	Programmierung über ICSP (nur Arduino UNO R3)	507
12.4	Arduino im Miniaturformat mit ATtiny	512
13	Tools für Praktiker	539
13.1	Hardware	539
13.2	Leiterplatten herstellen	559
A	Codereferenz	567
B	Boards	601
C	Bezugsquellen	607
	Stichwortverzeichnis	609

Einleitung

1.1 5. Auflage

Es ist das Jahr 2023 und vor Ihnen liegt die 5. Auflage von »Arduino Praxiseinstieg«. Auch über 10 Jahre nach der Erstveröffentlichung dieses Buch-Titels gehört Arduino weiterhin zu den wichtigsten Microcontroller-Boards für Maker und im Bildungsbereich.

Das Arduino-Projekt hat sich im letzten Jahr auf seine ersten Schritte zurückbesinnt und einen Bausatz mit einem Arduino UNO zum Selberlöten, das »Arduino Make Your UNO Kit« vorgestellt. Jeder Bastler kann sich nun einen eigenen UNO mit konventionellen Bauteilen von Hand zusammenbauen, um die Technik und alle Komponenten zu erlernen.

Im Frühling 2023 hat das Arduino-Team den neuen Arduino UNO R4 angekündigt.

Ende Juni 2023 wurde dann endlich der neue Arduino UNO R4 vorgestellt: das Standardmodell UNO R4 Minima und ein UNO R4 Wifi. Der UNO R4 Minima ist der Nachfolger des bisherigen UNO R3. Der UNO R4 Wifi hat, wie der Name aussagt, eine Wifi-Schnittstelle für Internet- und IoT-Anwendungen. Beide Boards basieren auf einem neuen Microcontroller und bieten eine Menge zusätzlicher Funktionen.

In dieser 5. Auflage werden die beiden neuen UNO R4 ausführlich beschrieben. Das bisherige Standardboard UNO R3 spielt aber weiterhin eine Rolle in der Arduino-Community, da bisher noch nicht alle Bibliotheken und Funktionen auf die neuen Boards umgeschrieben sind. Auch gibt es natürlich noch Tausende von Projekten, Beispielen und Tutorials, die mit einem Arduino UNO R3 betrieben werden.

Darum wird in dieser Auflage bei den Beispielen und Projekten jeweils erwähnt, für welches Board die Code-Beispiele ausgelegt und lauffähig sind.

1.2 Aufbau des Buchs

Am besten arbeitet man das Buch der Reihe nach durch, da die einzelnen Kapitel aufeinander aufbauen. In den ersten Kapiteln werden zunächst die wichtigsten Grundlagen der Hard- und Software des Arduino-Projekts beschrieben. Wer dieses Wissen bereits erworben hat, kann diese Kapitel natürlich überspringen.

In Kapitel 2 wird kurz über die Entstehung des Arduino berichtet und die Idee vom Basteln, Testen und Ausprobieren erläutert. Das Ganze ist aber kurz gehalten, da das Ziel dieses Buchs die praktische Arbeit ist. Im Anschluss beginnt dann der praktische Teil. Sie lernen zuerst die Hardware und die verschiedenen Arduino-Boards kennen. Begriffe werden erklärt und dann startet schon die eigentliche Installation der Software, der Entwicklungsumgebung. Nach erfolgreicher Installation und Verbindung mit der Arduino-Plattform schließt das Kapitel mit dem ersten Programm, dem Testprogramm Blink. Es folgt eine Einführung in die Oberfläche der Entwicklungsumgebung, und schon ist sie für die ersten Programme bereit.

Kapitel 3 startet mit einem Hardwareteil, in dem Sie die Arduino-Boards und den Schaltungsaufbau mit dem Steckbrett kennenlernen. Anschließend werden die Begriffe Strom, Spannung und Widerstand sowie die wichtigsten elektronischen Bauelemente erklärt. Als Nächstes werden die wichtigsten Begriffe rund um den Programmcode erklärt: Was ist eine Variable und wie ist eine Funktion aufgebaut? Nach der Einführung in die Struktur der Arduino-Programme wird das Testen und Debuggen des Programmcodes beschrieben. Hier wird aufgezeigt, wie man den seriellen Monitor zur Fehlersuche nutzen kann.

Digitale Eingänge lesen und Ausgänge schalten sind die nächsten Schritte in Kapitel 4. Die erste Leuchtdiode wird zum Leuchten gebracht. Im Anschluss befassen wir uns mit der analogen Welt. Es werden grundsätzliche Themen wie die Auflösung von Analogwandlern erklärt. Das erste richtige Programm liest die Sensorspannung eines Temperatursensors ein und gibt den Wert im seriellen Monitor aus. Nach dem Einlesen von analogen Werten werden analoge Signale mittels Pulsweitenmodulation ausgegeben.

Das nächste Thema in Kapitel 4 ist die serielle Kommunikation über die serielle Schnittstelle (RS232). Es werden Daten ausgegeben und eingelesen. Anschließend wird der Datentransfer über einen 2-Draht-Bus (I²C-Bus und Two-Wire-Bus) beschrieben. Daten werden von einem Master zum Slave versendet, wodurch ein Miniatur-Servo gesteuert wird. Weiter werden praktische I²C-Anwendungen wie ein serieller Sensor und eine busfähige Uhrenanwendung realisiert. Zum Schluss wird noch eine drahtlose Kommunikation mittels 433-MHz-Technologie erklärt. Ein Projekt, in dem ein elektronischer Würfel aufgebaut wird, schließt dieses Kapitel ab.

In Kapitel 5 werden Sensoren wie ein Fotowiderstand und viele Temperatursensoren sowie weitere Sensoren zur Umwelterfassung beschrieben. Danach folgt die Beschreibung sonstiger Sensoren. Mit einem Beschleunigungssensor wird eine kleine Wasserwaage realisiert. Als Kapitelprojekte werden ein elektronischer Kompass, ein Gefrierschrankwächter und ein kontaktloses Fieberthermometer realisiert.

Als Nächstes folgen in Kapitel 6 die Aktoren wie Relais und Servo. Bei einer Servoanwendung wird ein Servo zu einer analogen Temperaturanzeige umgebaut. Nun lernt der Leser die verschiedenen Motoren und deren Ansteuerung kennen. Die Erläuterung der Frage, wie man hohe Lasten schaltet, schließt das Thema Aktoren ab. Als Kapitelprojekt wird ein kleiner Roboter realisiert. Der kleine Roboter kann dabei mittels einer bekannten Komponente aus dem Spielboxbereich gesteuert werden, dem Wii Nunchuk, dem Joystick für die Wii-Spielkonsole.

Kapitel 7 behandelt die verschiedenen Anzeigeelemente. Es wird erklärt, wie man diese ansteuert und wie man die Helligkeit regeln oder fest begrenzen kann. Beispiele von Ansteuermöglichkeiten für LC-Displays und LED-Matrix und Beispiele mit Nokia-Display und OLED-Display runden das Thema Anzeigen ab. Ein Geschwindigkeitsmesser fürs Fahrrad und ein Schrittzähler für Jogger schließen dieses Kapitel ab.

Kapitel 8 beschreibt die verschiedenen Arten der Datenverarbeitung. Es werden Daten in ein EEPROM und auf eine SD-Karte geschrieben. In einem praktischen Projekt wird ein programmierbarer Signalgeber zur Erzeugung analoger Signale realisiert. Dabei lernen Sie eine neue Variante der Ansteuerung von Digitalports kennen.

In Kapitel 9 werden verschiedene Softwarebibliotheken zur Erweiterung der Arduino-Funktionalität vorgestellt. Der zweite Teil dieses Kapitels behandelt die Hardwareerweiterungen, »Shields« genannt. Einige wichtige und nützliche Shields werden etwas genauer betrachtet. Zum Abschluss dieses Kapitels wird eine praktische Lösung zur Herstellung eines Wii-Adapters vorgestellt.

Kapitel 10 beschreibt verschiedene praktische Arduino-Lösungen und wie man mittels Ethernet-Erweiterung mit dem Arduino kommunizieren kann. Zusätzlich wird der UNO R4 Wifi im drahtlosen Einsatz als Webclient und Webserver erklärt. Zum Abschluss dieses Kapitels werden Sensordaten gesammelt und an ein Auswertungstool übertragen. Im Schlussprojekt dieses Kapitels wird eine kleine Wetterstation aufgebaut, die das aktuelle Wetter und die Wettervorhersage für den kommenden Tag anzeigt. Zusätzlich gibt es noch ein Kapitel zum Download auf www.miptp.de/0661, das Wifi mit ESP8266 für den Arduino UNO R3 beschreibt.

Fehlersuche und Troubleshooting sind die Themen in Kapitel 11. Es wird gezeigt, wie man die eigene Schaltung oder das neu erstellte Programm zum Laufen bringt.

Kapitel 12 beschreibt die verschiedenen Arduino-Clones und wie man sich einen minimalen Arduino auf dem eigenen Steckbrett aufbauen kann.

In Kapitel 13 werden verschiedene Werkzeuge beschrieben, die bei den Arduino-Projekten nützlich und hilfreich sind. Neben Steckbrett, Lötkolben und Zangen werden auch Messgeräte wie Multimeter und Oszilloskop erläutert. Im Teil zum

Thema Softwaretools lernen Sie Programme zur Schaltplan- und Leiterplattenerstellung kennen.

Im Schlussteil und Anhang des Buchs finden Sie eine Codereferenz, eine Board-Übersicht, Informationen über Bezugsquellen sowie Anpassungen, die bei der Migration von älteren Arduino-Anwendungen auf die aktuelle Version 1.x nötig sind. Ein Bonus-Kapitel zum Download unter www.mitp.de/0661 listet alle Stücklisten der Beispiele und Projekte aus dem Buch auf.

1.3 Mehr Informationen

Weitere Informationen zu den Anwendungen und Beispielen im Buch sind auf der Buchwebsite erhältlich:

<http://arduino-praxis.ch>

Die Beispielskripte stehen im Downloadbereich zur Verfügung:

<http://arduino-praxis.ch/download/>

Viele Maker-Projekte, Reviews und Tipps gibt's auf der Maker-Plattform des Autors:

<https://www.555circuitslab.com/>

Für Anmerkungen oder Anregungen zu diesem Thema und die Kontaktaufnahme mit dem Autor stehen die Kontaktseite der Buchwebsite, E-Mail und Twitter bzw. X zur Verfügung.

Die E-Mail-Adresse zum Buch lautet:

kontakt@arduino-praxis.ch

Der Twitter bzw. X-Account lautet:

<http://twitter.com/arduinopraxis> oder der User [@arduinopraxis](https://twitter.com/arduinopraxis)

Im Blog zum Buch werden laufend neue und interessante Projekte sowie Produktvorstellungen aus der Arduino-Welt publiziert.

1.4 Weitere Quellen

Die größte Quelle für weitere Fragen zum Arduino ist natürlich das Internet. Zu fast jedem Problem gibt es bereits realisierte Lösungen oder Ansätze.

Für Arduino-Anwender sind folgende Websites ideale und empfehlenswerte Anlaufstellen bei Problemen und Fragen.

Arduino-Website:

<https://arduino.cc>

Make:Blog:

<https://blog.makezine.com/>

Explore & Learn Adafruit:

<https://learn.adafruit.com/>

1.5 Danksagung

Mein Dank geht vor allem an meine Familie, meine Frau Aga und meine beiden Jungs Tim und Nik. Auch während der Arbeit an der 5. Auflage dieses Buchs mussten sie wieder viele Stunden ohne mich auskommen. Die Skiferien haben sie abermals allein verbracht und auch an den Wochenenden waren viele Stunden für das Schreiben des Buchs verplant. Aber alle haben sehr viel Verständnis gezeigt und mich dabei unterstützt.

Ein Dank geht an die Arduino-Community. Ohne sie wäre dieses Buch nicht entstanden und niemand hätte originelle Lösungen und Lösungsansätze realisiert, die mich zu meinem Buch inspirierten. Viele nette und konstruktive E-Mails und Kommentare zum Buch haben mir Auftrieb für die 5. Auflage geben.

Natürlich möchte ich auch einen Dank an das Arduino-Core-Team aussprechen. Die Realisierung dieses Open-Source-Projekts ist eine Bereicherung für die Hardwaregemeinde. Die Idee einer Open-Source-Plattform, offener Hardware und kostenloser Entwicklungstools ist einfach großartig.

Zum Schluss möchte ich mich wiederum bei meiner Lektorin Sabine Schulz bedanken. Sie hat mir die nötige Zeit gegeben, um diese 5. Auflage zu schreiben.

Arduino-Plattform

2.1 Am Anfang war der König

Arduin von Ivrea war in den Jahren 1001 bis 1015 der Markgraf von Ivrea, einer italienischen Stadt in der Region Piemont. Dieser Graf Arduin, in den Jahren 1002 bis 1004 auch König von Italien, ist der indirekte Namensgeber des Arduino-Projekts. Indirekt, weil er das Projekt nicht selbst so getauft hat, dies haben Dozenten am *Interaction Design Institute Ivrea* (IDII) getan. Andere meinen, dass die Bar in der Nähe des Instituts der Namensgeber für das Arduino-Board sei.

Tatsache ist, dass die Dozenten und Studenten vor dem Problem standen, dass es zum damaligen Zeitpunkt keine einfachen und kostengünstigen Microcontroller-Systeme für Designstudenten und deren Design- und Kunstprojekte gab. Zusammen mit Elektronikingenieuren wurde daher eine erste Serie von Arduino-Boards produziert. Als Entwicklungsumgebung stand die Processing-Umgebung Pate, und daraus wurde eine Programmiersprache für den Arduino entwickelt.

Durch die Offenheit dieser Plattform – die Entwicklungsumgebung ist kostenlos verfügbar und die Hardware kann man fertig kaufen, oder man stellt sie sich selbst her – hat sich die Arduino-Community schnell vergrößert. Viele Studenten und Bastler haben Lösungen und eigene Boards realisiert und die Projektinformationen und Daten im Internet publiziert.

Die Vorstellung des Arduino-Projekts vor fast 20 Jahren hat eine richtige Selbstbau-, DIY- und Makerszene hervorgerufen. In den letzten Jahren haben viele findige Tüftler neue Board-Varianten erstellt und Firmen bieten Produkte, einzelne Komponenten oder ganze Bausätze zum Verkauf an, wobei Stromlaufpläne und auch Programme für diese fertigen Produkte und Lösungen bereitgestellt und von den Bastlern weiterverwendet werden dürfen.

Aus den Originaldaten des Standardboards des Arduino sind viele neue Projekte und Elektronik-Produkte auf den Markt gekommen und viele auch wieder verschwunden.

Das Arduino-Projekt hat zahlreiche Schüler, Studenten und junge Leute animiert, sich mit Elektronik zu beschäftigen und eine berufliche Karriere im technischen Umfeld zu starten.

Auch geht das Arduino-Projekt nochmals auf seine Ursprünge zurück und hat im Herbst 2022 einen Arduino-Bausatz zum Selberlöten herausgebracht. Mit dem Projekt »Arduino Make Your UNO Kit« wird ein Schritt zurück in die Zukunft gestartet.

<https://blog.arduino.cc/2022/11/03/are-you-ready-to-go-back-to-the-future/>

2.2 Be a Maker

Basteln, Spaß und Lernen ist das Motto bei den Arduino-Anwendungen. Die Anwendungen sollen nämlich nicht nur von Ingenieuren realisiert werden, die die Schaltungsentwicklung an Hochschulen erlernt haben. Die Entwicklung von Anwendungen soll auf praktischem Weg durch Aufbauen und Testen erfolgen. Das ingenieurmäßige Vorgehen durch Planen und Berechnen ist hier nicht das Wesentliche. Eine Lösung sollte schnell als Prototyp aufgebaut sein, und anschließend kann man überprüfen, ob das Resultat auch der Idee entspricht und ob die Anwendung das gewünschte Verhalten aufweist. Natürlich geht dabei auch mal etwas schief, zum Beispiel brennt ein Widerstand durch oder der Servo bleibt am Anschlag stehen.

Das Basteln und Probieren sollte Spaß machen. Dabei sollte der Kreativität freier Lauf gelassen werden. Beim Aufbau der Prototypen können plötzlich wieder neue Ideen entstehen, und vielleicht verändern sich auch die im Voraus angedachten Vorgaben.

Beim Entwickeln von Elektronikprojekten kommen oftmals auch vorhandene Geräte zum Einsatz, die man im Haushalt findet. Einen automatischen Raumlüfterfrischer, den es in verschiedenen Discountergeschäften oder Drogeriemärkten zu kaufen gibt, nutzt man üblicherweise im Haus oder in der Wohnung, um die Luft zu erfrischen. Nimmt man diesen auseinander und versieht ihn mit einer intelligenteren Logik in Form eines Arduino-Boards, entsteht aus einem einfachen Gegenstand ein ganz neues Produkt. Das Resultat ist in diesem Fall ein automatischer Luftbefeuchter für das heimische Terrarium. Mit einem intelligenten Regensensor kann die Lüfterfrischerlösung sogar zu einer internetbasierten Pflanzenbewässerung erweitert werden.

Wie das Beispiel zeigt, kann man für seine Bastelprojekte mit dem Arduino viele Gegenstände aus dem Haushalt verwenden oder diese umbauen und um eine Logik erweitern.

Das Basteln und Entwickeln mit Arduino ist eine Sache, die Spaß macht und die Vorstellungskraft und Kreativität fördert. Dabei muss nicht alles im Voraus geplant und berechnet werden. Es soll auseinandergelöst, geforscht und gebastelt werden, damit viele spannende und auch lustige Anwendungen entstehen. Wäh-

renddessen gilt es, immer wieder verschiedene Herausforderungen und Hürden zu nehmen. Läuft das Arduino-Programm und funktioniert das Zusammenspiel mit den externen Sensoren und Aktoren, kommt der nächste Schritt: der Aufbau der Lösung in einer stabilen Form.



Abb. 2.1: Indianerfigur aus Abfallteilen einer Elektronikproduktion

Elektronik-Basteln als Hobby hat sich zu einem richtigen Markt entwickelt. Viele Online-Elektronikplattformen und Community-Websites bilden die sogenannte Maker-Community und liefern Anleitungen, Ideen, Bausätze und Komponenten für den Selbstbau. Stellvertretend für diese Anbieter sind Adafruit Industries (<https://www.adafruit.com>) und Sparkfun (<https://www.sparkfun.com>) zu erwähnen. Beide Firmen bieten auf ihren Websites viele Tutorials und Anleitungen für Arduino-Anwender. Damit man umgehend mit den Experimenten und Basteleien loslegen kann, liefern sie auch alle nötigen Bauteile und Werkzeuge. In den Onlineshops können sich Bastler alle Teile, vom einzelnen Widerstand über Leiterplatten und Stecker bis zum kompletten Toolkit mit Bauteilen, Zangen und Lötkolben besorgen. Weltweit finden Messen für Maker und Bastler statt, die als Maker Faire bezeichnet werden, auf denen sich Hobbyanwender und Profis treffen und austauschen. Auf der Maker-Faire-Website (<https://makerfaire.com/>) werden jeweils die kommenden Messen angekündigt. In Europa ist speziell die

Maker Faire Rome (<https://www.makerfairerome.eu>) zu erwähnen, auf der sich die Maker aus ganz Europa treffen.

Das Schöne an diesem Hobby ist, dass man in das Elektronik-Basteln nicht viel Geld investieren muss. Viele Teile hat man bereits im Haushalt oder im Hobbykeller. Zu Beginn kauft man sich meist als Grundausstattung ein Arduino-Board, installiert die kostenlose Entwicklungsumgebung und schon kann man die ersten Tutorials durcharbeiten und originelle Lösungen aufbauen. So könnte beispielsweise eine dimmbare LED-Lampe mit einem Lampengehäuse aus einer leeren Milchflasche das heimische Wohnzimmer verschönern.

Mit den Projekten kommen die Ideen, und die dann noch fehlenden Bauteile, wie beispielsweise einen Infrarotsensor für berührungslose Lichtschalter, kauft man jeweils von Fall zu Fall.

Neben den verschiedenen Anbietern von Tutorials und den nötigen Werkzeugen und Bauteilen fördert auch eine immer größer werdende Community im Internet das Tinkering. Das Arduino-Forum (<https://forum.arduino.cc/>), die erste Adresse bei Problemen oder Fragen rund um Arduino, bietet für jeden Arduino-Bastler, egal ob Anfänger oder Profi, viele Ideen und Unterstützung. Zusätzlich stehen beim Arduino-Playground viele Anleitungen, Tutorials und Beispielprogramme für eigene Projekte bereit. Im Wiki kann jeder Anwender seine Projekte und Lösungen vorstellen und dokumentieren.

Die Do-it-yourself-Plattform Instructables (<https://www.instructables.com>) gehört zu den größten Bastel-Portalen im Internet. Diese Selbstbauplattform wird von vielen Bastlern genutzt. Tausende von Usern präsentieren hier ihre Selbstbauprojekte. Genaue Anleitungen, meist durch Bilder oder Videos ergänzt, unterstützen den Bastler beim Nachbau des vorgestellten Projekts. Der Benutzer findet für fast jedes Problem eine Anleitung oder die Beschreibung einer bereits realisierten Lösung. Kompetente Benutzer unterstützen den Anfänger oder Einsteiger bei Fragen und Problemen.

2.3 Arduino-Plattform

Die Arduino-Plattform besteht aus zwei Teilen: einem Hardwareteil und einem Softwareteil.

Der Hardwareteil der Arduino-Plattform besteht aus einem Microcontroller-Board, das mittels Ein- und Ausgänge die Verbindung zur physischen Welt herstellt. Die digitalen und analogen Eingänge können Daten von Sensoren wie Schaltern, Temperatursensoren oder GPS-Modulen einlesen. Die Ausgänge können Leuchtdioden, Relais, Servos oder Motoren ansteuern.

Die Entwicklungsumgebung, auch IDE (Integrated Development Environment) genannt, ist der Softwareteil. Die Entwicklungsumgebung basiert auf Processing

und ist die Plattform, auf der die ausführbaren Programme erstellt werden, die dann von dem Hardwareteil ausgeführt werden.

Die »Programmiersprache« ähnelt C++ und ist so ausgelegt, dass man Programme ohne allzu tiefe Programmierkenntnisse erstellen kann.

Die Software selbst wird auf einem Computer installiert und ist für verschiedene Plattformen (Windows, Mac OS und Unix/Linux) verfügbar.

Das erstellte Programm, auch »Sketch« genannt, wird via USB-Port von der Entwicklungsumgebung in den Speicher des Arduino-Boards geladen. Diese physische Verbindung zum Hardwareteil muss während der Entwicklungs- und Testphase immer vorhanden sein. Die Übertragung erfolgt seriell.

Gleichzeitig dient die Verbindung auch als Hilfsmittel bei der Fehlersuche. Mithilfe eines seriellen Monitors können die Zustände der Ein- und Ausgänge oder die von den Sensoren gelieferten Daten überwacht werden.

Nach der Entwicklung kann die Verbindung zur Entwicklungsumgebung getrennt werden, und der Arduino kann als unabhängiger Minicomputer mit der Außenwelt kommunizieren.

Falls aber Daten zur Weiterverarbeitung oder zur Anzeige an den Computer gesendet werden müssen, bleibt die serielle Verbindung über den USB-Anschluss bestehen.

2.4 Hardware

Die Hardware des Arduino-Projekts ist eine Leiterplatte, die mit verschiedenen Elektronik-Bauteilen bestückt ist. In der Praxis wird diese Leiterplatte auch als »Board« bezeichnet.

Im Arduino-Projekt stehen verschiedene Boards zur Verfügung, die je nach Anforderungen (Baugröße, Anzahl digitaler Ein- und Ausgänge, Speicherbedarf) verwendet werden können.

Auf jedem Board sind die nötigen elektronischen Komponenten wie Schaltkreise, Widerstände, Kondensatoren etc. platziert. Zusätzlich besitzen die Boards verschiedene Stecker und Schnittstellen für die Verbindung mit der Außenwelt.

Die Spannungsversorgung erfolgt entweder über ein separates, externes Netzteil oder über die angeschlossene USB-Schnittstelle. Die USB-C-Schnittstelle liefert eine Spannung von 5 Volt über die das Board und die externen Sensoren und Module versorgt werden.

Über die USB-Schnittstelle erfolgt neben der soeben beschriebenen Spannungsversorgung auch die Kommunikation mit der Entwicklungsumgebung, sprich das Hochladen von Programmen (Sketche) und die Überwachung via seriellen Monitor.

Das Standardboard des Arduino (Stand August 2023) heißt Arduino UNO R4. Alle Erklärungen und Beispiele in diesem Buch beziehen sich auf den Arduino UNO R4 Minima. Einzelne Code-Beispiele funktionieren aber nur mit dem UNO R3, da noch nicht alle Bibliotheken für das gerade herausgekommene Modell R4 umgeschrieben sind.

Bei den Beispielen wird jeweils erwähnt, auf welchem Modell der Code lauffähig ist.

Der Arduino UNO R4 kann man momentan (Stand September 2023) nur über Shop von Arduino kaufen.

Der Arduino UNO R3 war jahrelang das Standard-Board und kann weiterhin von verschiedenen Anbietern (siehe Anhang C, »Bezugsquellen«) bezogen werden.

Zusätzlich gibt es eine große Anzahl von kompatiblen Boards (Clones) bei vielen Händlern und Anbietern. Diese bieten auch Bausätze oder nur die Leiterplatten des UNO R3 an.

Neben den Einzelkomponenten und den komplett aufgebauten Boards besteht für den fortgeschrittenen Elektronikbastler auch die Möglichkeit, ein Board mittels CAD-Programm selbst zu erstellen. Auf der Arduino-Website stehen hierzu der Stromlaufplan und die Leiterplattendaten zur Verfügung.

In den letzten zehn Jahren haben viele innovative Bastler Arduino-Boards in zahlreichen Projekten realisiert. Da neben den unter Open-Source-Lizenz vertriebenen Boards auch das eigentliche Gehirn des Arduino, der programmierbare ATmega-Microcontroller, frei verfügbar ist, kann man mit einem Microcontroller mit Bootloader und ein paar externen Komponenten schnell einen einfachen Arduino aufbauen. Der Bootloader ist ein kleines Programm, das im Microcontroller gespeichert ist und das Hochladen von Programmen erlaubt. Das Bootloader-Programm wird jeweils bei Anschluss der Spannungsversorgung am Microcontroller gestartet.

Der UNO R4 ist mit einem modernen 32-Bit Microcontroller ausgerüstet. Zum aktuellen Zeitpunkt sind von diesem Board noch keine Nachbauten im Internet verfügbar.

2.4.1 Arduino Uno R4

<https://store.arduino.cc/collections/boards/products/uno-r4-minima>

Der Arduino UNO R4 ist das aktuelle Standard-Board des Arduino-Projekts. In Abbildung 2.2 ist das Model UNO R4 Minima abgebildet.

Auf dem Arduino UNO R4 sind verschiedene Stecker und Anschlussbuchsen montiert. Die Position der verschiedenen Anschlussbuchsen ist fix definiert und stellt die Kompatibilität mit früheren und zukünftigen Boards sicher.

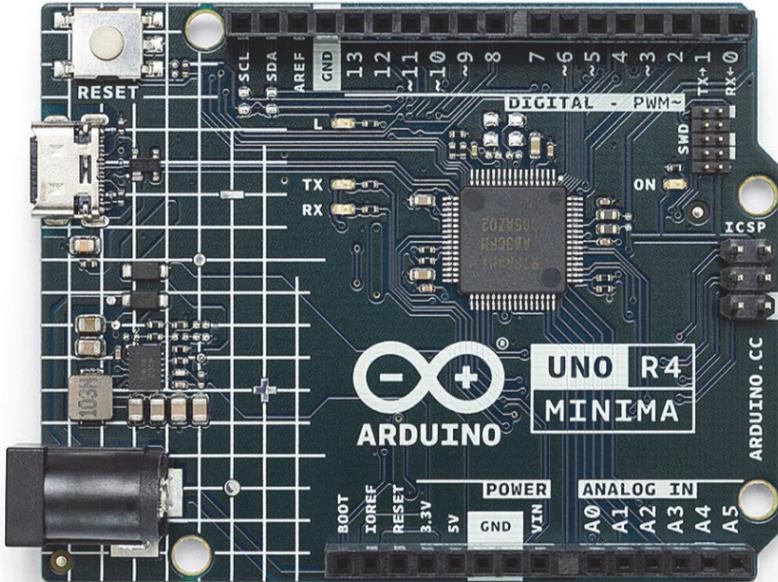


Abb. 2.2: Arduino UNO R4 Minima (Quelle: www.arduino.cc)

In Abbildung 2.3 sind die Anschlussmöglichkeiten auf der Leiterplatte des Arduino Uno gekennzeichnet.

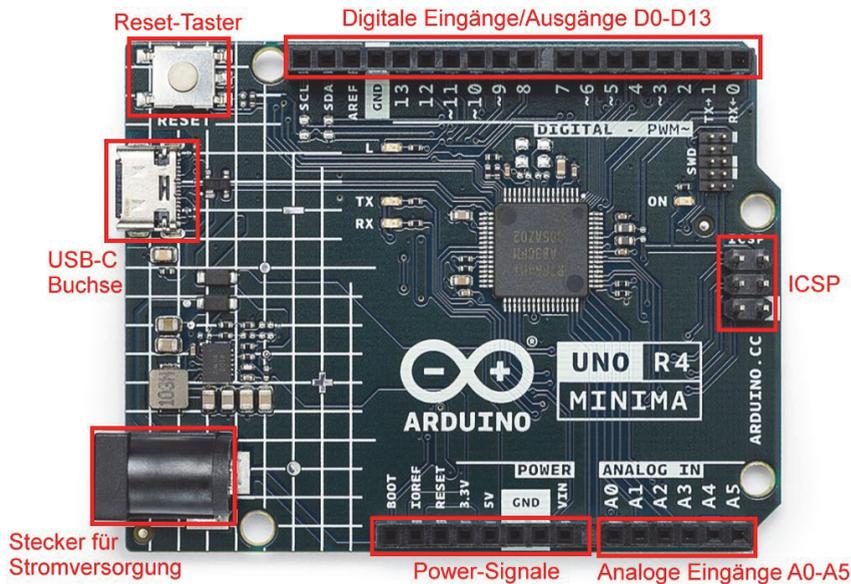


Abb. 2.3: Arduino UNO R4: Anschlussmöglichkeiten

Nachfolgend werden die einzelnen Anschlussmöglichkeiten beschrieben.

USB-Buchse: Der USB-Anschluss (USB-C) wird für die Programmierung mit der Entwicklungsumgebung und zur Kommunikation mit dem angeschlossenen Rechner verwendet. Gleichzeitig kann das Board über den USB-Anschluss mit Strom versorgt werden.

Stecker für externe Stromversorgung: Über eine 2,1-mm-Buchse kann das Arduino-Board per externem Steckernetzteil oder Batterie mit Spannung versorgt werden. Die Stromversorgung über den USB-Port wird dabei automatisch deaktiviert.

Vorteilhaft ist vor allem die höhere Leistung, die aus der externen Versorgungsspannung bezogen werden kann, beispielsweise zur Versorgung von Aktoren und Sensoren. Im Falle eines Kurzschlusses wird nicht der USB-Port des angeschlossenen Rechners belastet, sondern der Überlastschutz der externen Stromversorgung.

Steckerleisten für digitale Ein- und Ausgänge (D0-D13): Einreihige Anschlussbuchsen, auf die Erweiterungsplatinen (Shields) oder Steckerleisten gesteckt werden können. Alle vorhandenen digitalen Signale und Schnittstellen sind über diese Anschlussleisten verfügbar.

Steckerleiste für analoge Eingänge (A0-A5): Einreihige Anschlussbuchsen für den Anschluss von 6 analogen Eingangssignalen.

ICSP: Die Stiftreihe für das ICSP (In-Circuit Serial Programming) erlaubt die Programmierung von Sketchen (Programmen) ohne Bootloader mittels eines externen Programmiergeräts, dazu mehr in Kapitel 10.

Neben den Anschlussmöglichkeiten sind auf der Leiterplatte des Arduino Uno Bauteile und Funktionen vorhanden, die in Abbildung 2.4 dargestellt sind.

Reset-Taster: Mit dem Reset-Taster kann der Microcontroller zurückgesetzt werden. Nach einem Reset erfolgt ein Neustart des Boards.

Stromversorgung: Mit der Schaltung der Stromversorgung wird die über den 2.1 mm-Hohlstecker zugeführte Versorgungsspannung auf 5 V geregelt.

Microcontroller: Der Microcontroller ist das Gehirn des Arduino UNO, er führt die Programme aus und verarbeitet die Ein- und Ausgangssignale.

SWD-Anschluss: Über den SWD-Anschluss kann der Entwickler einen externen Debugger (J-Link-Debugger) anschließen, um die Funktionen des Microcontrollers zu überwachen und zur vertieften Fehlersuche im System.

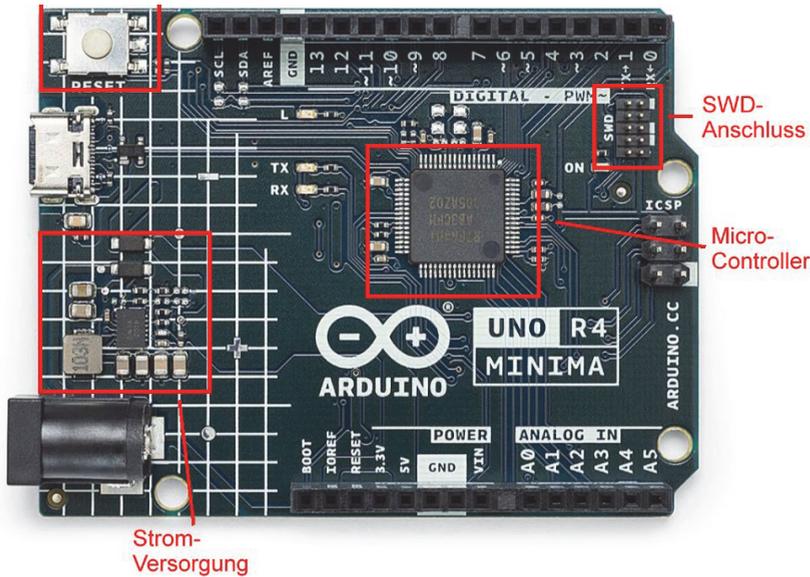


Abb. 2.4: Arduino UNO R4 Minima: Bauteile und Funktionen

In Tabelle 2.1 sind die technischen Daten des Arduino UNO R4 aufgelistet.

Beschreibung	Detailldaten
Microcontroller	R7FA4M1AB3CFM#AA0 (Renesas) 48 MHz Arm Cortex-M4
Spannungsversorgung	6–24 VDC
Betriebsspannung	5 VDC und 3,3 VDC (intern über Spannungsregler generiert)
Digitale Ein-/Ausgänge	14 (D0-D13) davon 6 als PWM (D3, D5, D6, D9, D10, D11)
Analoge Eingänge	6 (A0-A5)
Strom pro digitalem Pin	8 mA DC
Flash Memory	256 KB
SRAM	32 KB
EEPROM	8 KB
Taktfrequenz	48 MHz
USB-Schnittstelle	ja
Reset-Taster	ja
Onboard-ICSP-Stecker	ja
Abmessungen Board (L x B)	70 x 53 mm

Tabelle 2.1: Technische Daten Arduino UNO R4

2.4.2 Arduino UNO R4 WIFI

<https://store.arduino.cc/products/uno-r4-wifi>

Gleichzeitig mit dem UNO R4 Minima wurde noch ein zweites Board vorgestellt: der Arduino UNO R4 WIFI (Abbildung 2.5).

Wie die Typenbezeichnung »WIFI« aussagt, hat dieses Board eine WIFI-Schnittstelle eingebaut, die mit einem ESP32 realisiert wurden.

Der Arduino UNO R4 WIFI eignet sich somit als Webserver oder Webclient und für den Einsatz als IOT- oder Sensor-Node mit WLAN-Schnittstelle.

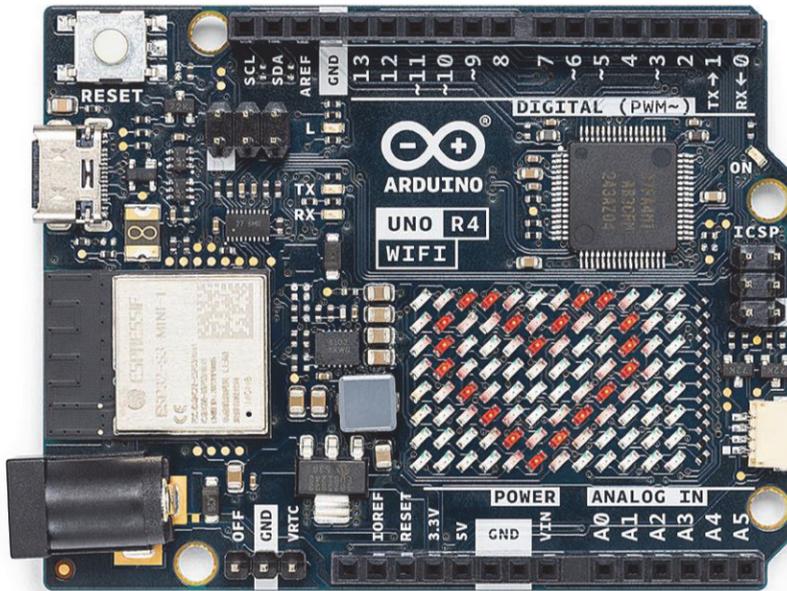


Abb. 2.5: Arduino UNO R4 WIFI (Quelle: www.arduino.cc)

In Kapitel 10 wird die WIFI-Funktionalität des UNO R4 WIFI im Detail erklärt.

2.4.3 Arduino UNO R3

<https://store.arduino.cc/collections/boards/products/arduino-uno-rev3>

Der Arduino UNO R3 ist der weitverbreitete Vorgänger des UNO R4. In Abbildung 2.6 ist das Board abgebildet.

In Abbildung 2.7 sind die Anschlussmöglichkeiten auf der Leiterplatte des Arduino UNO R3 gekennzeichnet.

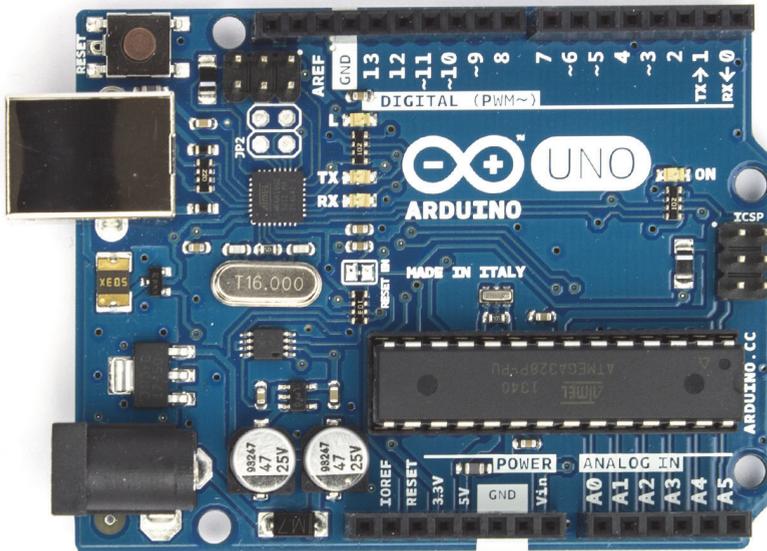


Abb. 2.6: Arduino Uno Rev 3 (Quelle: www.arduino.cc)

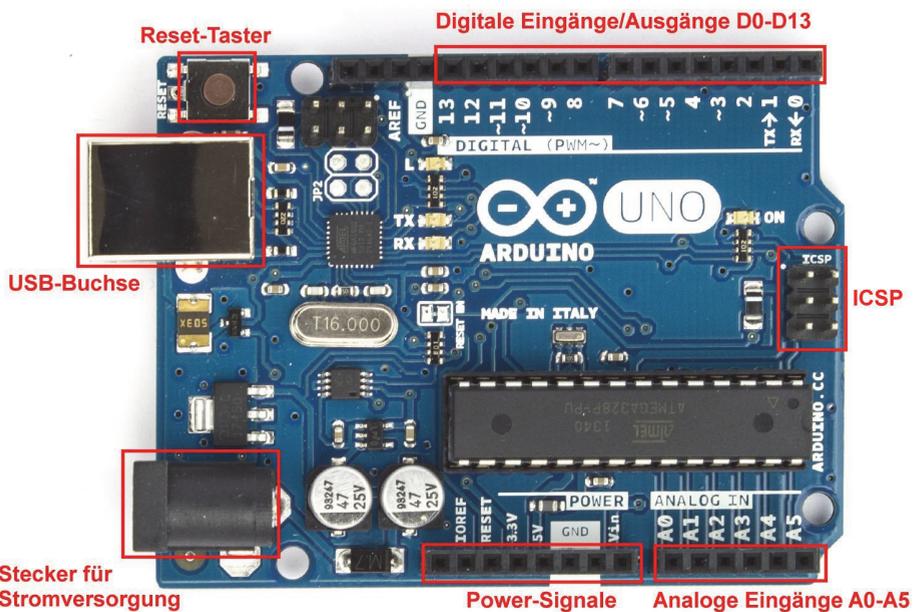


Abb. 2.7: Arduino UNO R3: Anschlussmöglichkeiten

In Abbildung 2.8 sind die Bauteile und Funktionen des UNO R3 abgebildet.

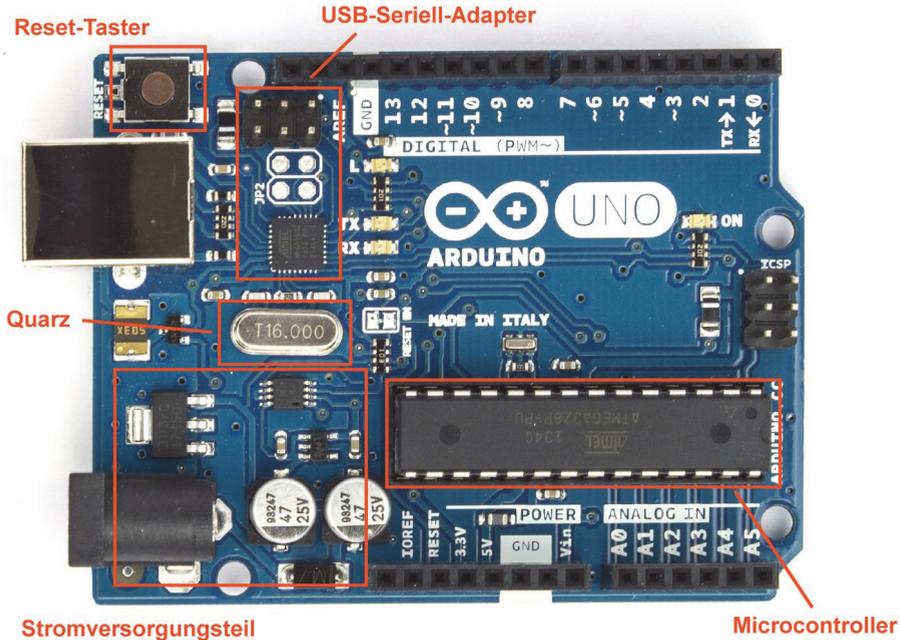


Abb. 2.8: Arduino UNO R3: Bauteile und Funktionen

Reset-Taster: Mit dem Reset-Taster kann der Microcontroller zurückgesetzt werden. Nach einem Reset erfolgt ein Neustart des Boards.

USB-Seriell-Adapter: Dieses Schaltungsteil wandelt das USB-Signal in ein serielles Signal um, das der Microcontroller verarbeiten kann. Die seriellen Signale sind an den digitalen Pins D0 (RX) und D1 (TX) verfügbar.

Quarz: Der Schwingquarz ist das Bauteil, das den Takt für den Microcontroller bereitstellt.

Stromversorgungsteil: Im Stromversorgungsteil wird die Versorgungsspannung, die über den schwarzen 2,1-mm-Klinenstecker zugeführt wird, auf 5 V geregelt. Erfolgt die Spannungsversorgung über den USB-Anschluss, so stellt dieses Schaltungsteil die saubere Umschaltung zwischen USB-Versorgung und externer Versorgung sicher.

Microcontroller: Der Microcontroller ist quasi das Gehirn des Arduino-Boards. Dieser kleine Minicomputer führt die Arduino-Programme aus und verarbeitet die Ein- und Ausgangssignale. Beim Arduino UNO wird ein Microcontroller vom Typ ATmega328 vom Hersteller Atmel eingesetzt.

In Tabelle 2.2 sind die technischen Daten des Arduino UNO R3 aufgelistet.

Beschreibung	Detailldaten
Microcontroller	ATmega328
Spannungsversorgung	7–12 VDC
Betriebsspannung	5 VDC und 3,3 VDC (intern über Spannungsregler generiert)
Digitale Ein-/Ausgänge	14 (davon 6 als PWM-Ausgänge)
Analoge Eingänge	6
Strom pro digitalem Pin	40 mA DC
Flash Memory	32 KB (ATmega328), wobei 0,5 KB vom Bootloader belegt werden
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Taktfrequenz	16 MHz
USB-Schnittstelle	ja
Reset-Taster	ja
Onboard-ICSP-Stecker	ja
Abmessungen Board (L x B)	70 x 53 mm

Tabelle 2.2: Technische Daten Arduino UNO R3

2.5 Software

Für die Entwicklung der Arduino-Programme muss auf dem lokalen Rechner eine Entwicklungsumgebung installiert werden. Die Arduino-Entwicklungsumgebung ist ein Java-Programm und kann kostenlos von der Arduino-Website heruntergeladen werden. Die Software ist für die drei gängigen Betriebssysteme (Windows, macOS und Linux) verfügbar und wird laufend aktualisiert und erweitert. Mit dem Download und der Installation der Software erhält man eine komplette Entwicklungsumgebung mit Code-Editor, Dokumentation und einer Anzahl von Beispielen sowie Standardbibliotheken für verschiedene Anwendungen. Die Installation und Konfiguration dieser Entwicklungsumgebung wird im nachfolgenden Abschnitt beschrieben.

2.6 Installation der Software

Im Downloadbereich der Arduino-Website (<https://www.arduino.cc/en/software>) stehen die aktuellen Versionen der Arduino-Entwicklungsumgebungen als Archivdatei zur Verfügung. Zum Zeitpunkt der Drucklegung dieses Buches (Herbst 2023) lautete die aktuelle Programmversion 2.2.1.

2.6.1 Installation

Die nachfolgende Installation bezieht sich auf die Betriebssysteme Windows und macOS. Die Installation unter Linux ist etwas komplizierter und von Distribution zu Distribution leicht unterschiedlich. Auf der Arduino-Website sind detaillierte Installationsanleitungen, Problembeschreibungen und Zusatzinformationen für die Installation auf den einzelnen Linux-Distributionen beschrieben (<http://playground.arduino.cc/Learning/Linux>).

Zum Download klickt man auf den Downloadlink der jeweiligen Betriebssystemversion und speichert die Software in einem temporären Verzeichnis auf dem Rechner ab. Die Installationsdatei hat in der aktuellen Programmversion eine Dateigröße von rund von 160 MB.

2.6.2 Installation Windows

Für Windows-Benutzer stehen mehrere Versionen zum Download bereit: die Installationsdatei für Win10/11 als EXE-Datei, ein MSI-Paket und eine ZIP-Version. Nach erfolgreichem Download der Arduino-Umgebung kann die Software ausgeführt oder mit einem Archivprogramm entpackt werden.

Die Variante mit der ausführbaren Installationsdatei (EXE) erfordert Administrationsrechte auf dem Rechner. Bei dieser Variante werden auch alle notwendigen Daten installiert. Die Installationsroutine installiert die Arduino IDE standardmäßig ins Verzeichnis

```
C:\Program Files\Arduino IDE
```

Die ZIP-Version kann ohne Administratorrechte auf den Rechner kopiert, entpackt und ausgeführt werden.

Beim ersten Start der Entwicklungsumgebung kann das Windows-System ein Hinweis bezüglich der Firewall Windows Defender anzeigen. Diese Meldung kann dann mit »Zugriff zulassen« bestätigt werden.

Praxis-Tipp

Für erfahrene Anwender empfehle ich die Installation der Entwicklungsumgebung jeweils in einen eigenen Ordner mit Versionsnummer (C:\Programme\arduino-2-0-3).

Auf diese Art kann in einer Version eine Einstellung oder Erweiterung vorgenommen werden, ohne dass die »produktive« Hauptversion beeinträchtigt ist. Die ist praktisch, wenn man mit verschiedenen Board-Varianten arbeitet und dabei die jeweiligen Board-Pakete verwendet (siehe dazu Kapitel 12).

2.6.3 Installation macOS

Unter macOS verschiebt man das ausführbare Programm nach dem Download ins Verzeichnis /Programme oder /Applications. Die ausführbare Datei heißt `Arduino.app` (Abbildung 2.9).

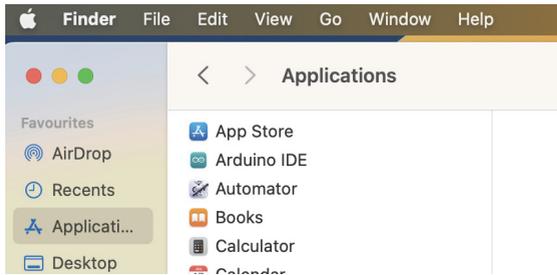


Abb. 2.9: macOS: Ausführbares Programm

Somit ist die Entwicklungsumgebung installiert beziehungsweise auf den Rechner kopiert. Falls die Installation der ZIP-Variante ausgeführt wird, muss noch ein USB-Treiber installiert werden. Für die Installationsvariante ist die Treiber-Installation automatisch erledigt. Unter macOS ist keine Treiberinstallation notwendig.

2.6.4 Installation des USB-Treibers unter Windows 10/11

Der erforderliche Treiber für die USB-Schnittstelle ist bereits bei der Installation der Entwicklungsumgebung mitkopiert worden und liegt im Verzeichnis

`arduino-verzeichnis\drivers\`

Um den Treiber zu installieren, muss eine Verbindung mit dem Arduino-Board hergestellt werden. Zu diesem Zweck wird ein USB-Kabel an einem verfügbaren USB-Anschluss des Rechners angeschlossen und mit dem USB-Port auf dem Arduino-Board verbunden. Auf dem Arduino-Board leuchtet die Leuchtdiode »On« und zeigt an, dass das Board mit Spannung versorgt wird.

Sofort startet auf dem Rechner eine Installationsroutine, die versucht, den Treiber für das Arduino-Board zu installieren. Kurze Zeit später meldet die Installationsroutine, dass der Treiber nicht installiert werden konnte.

Der Treiber muss nun manuell nachinstalliert werden.

1. Auf dem Rechner `START|SYSTEMSTEUERUNG` auswählen.
2. `GERÄTE-MANAGER` auswählen.
3. Unter `ANSCHLÜSSE (COM & LPT)` erscheinen der angeschlossene Arduino und der COM-Port.
4. Arduino-Board anklicken, rechte Maustaste.

5. Option TREIBERSOFTWARE AKTUALISIEREN auswählen.
6. Auf die Frage, wie die Treibersoftware gesucht wird, Option AUF DEM COMPUTER NACH TREIBERSOFTWARE SUCHEN auswählen.
7. Oben genannten Treiberpfad (ORDNER/DRIVERS) auswählen.
8. Nach Bestätigung wird die Treiberinstallation gestartet.
9. Anschließend meldet das System die erfolgreiche Installation des Treibers.

Im Geräte-Manager erscheint nun das Arduino-Board mit korrekt installiertem Treiber und der zugeordneten Schnittstelle (Abbildung 2.10)



Abb. 2.10: Geräte-Manager mit Arduino Uno und USB-Port

Hinweis

Damit dieser Port im Geräte-Manager angezeigt wird, muss der Rechner mit dem Arduino-Board verbunden sein.

2.6.5 Installation des USB-Treibers unter macOS

Die Installation eines USB-Treibers ist für den Arduino UNO und Arduino Mega 2560 nicht erforderlich. Für ältere Arduino-Boards sowie für alle Arduino-Clones mit einem FTDI-Chip muss der Treiber installiert werden. Der Treiber für den USB-Port kann von der Website des Treiberherstellers geladen werden (<https://www.ftdichip.com/Drivers/VCP.htm>).

Nach einem Doppelklick auf die Treiberdatei startet der Installationsprozess. Befolgen Sie zunächst die Installationsanweisungen und starten Sie zum Schluss den Rechner neu.

2.6.6 Installation des USB-Treibers unter Linux

Der aktuellste Treiber für Linux kann auf der Website des Chip-Herstellers FTDI heruntergeladen werden (<http://www.ftdichip.com/Drivers/VCP.htm>). Nachdem die Archivdatei `ftdi_sio.tar.gz` in einem temporäres Verzeichnis entpackt ist, folgen Sie den Beschreibungen in der README-Datei.

2.7 Get Connected

Mit der Installation der Entwicklungsumgebung (IDE) und des USB-Treibers ist auf dem Rechner bereits alles Nötige für die Entwicklung von Arduino-Anwendungen vorhanden.

Die Kommunikation zwischen der Entwicklungsumgebung erfolgt über den eingerichteten virtuellen COM-Port, wobei die USB-Schnittstelle als physisches Übertragungsmedium verwendet wird.

2.7.1 Verbindungskabel

Für die Verbindung des Rechners mit dem Arduino-Board kann ein Standard-USB-Kabel verwendet werden. Je nach Board-Variante ist hierzu auf der Seite des Arduino ein USB-Stecker vom Typ B (Arduino Uno) oder ein USB-Stecker vom Typ B-Mini erforderlich. Am Rechner ist meist ein USB-Stecker vom Typ A im Einsatz.

1. Schließen Sie das USB-Kabel am Rechner an.
2. Verbinden Sie das Arduino-Board mit dem USB-Kabel.

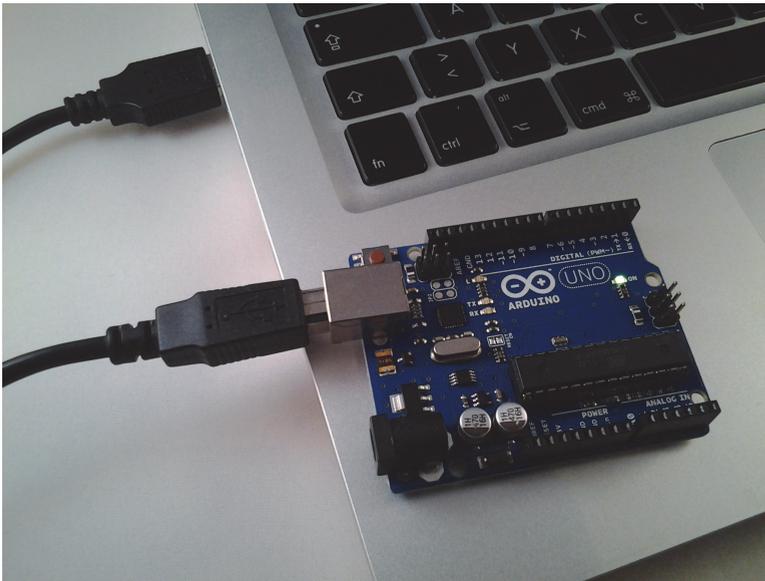


Abb. 2.11: Arduino: USB-Verbindung mit dem Rechner

Während der Realisierung eines Projekts und dem Entwickeln des Programmcodes wird das Arduino-Board meist über den USB-Anschluss des angeschlossenen Rechners versorgt (Abbildung 2.11).

2.7.2 Verbindung und »Blink«

Wenn das Board mit dem Rechner verbunden ist, wird die Entwicklungsumgebung (IDE) mittels eines Doppelklicks auf das Icon gestartet. Die Arduino-IDE wird mit einem Skriptfenster geöffnet (Abbildung 2.12).



Abb. 2.12: Arduino-IDE

Je nach Sprache der Windows-Version öffnet sich die Entwicklungsumgebung in englischer Sprache. Über den Menüpunkt FILE|PREFERENCES kann die Sprache auf Deutsch umgeschaltet werden (Abbildung 2.13).

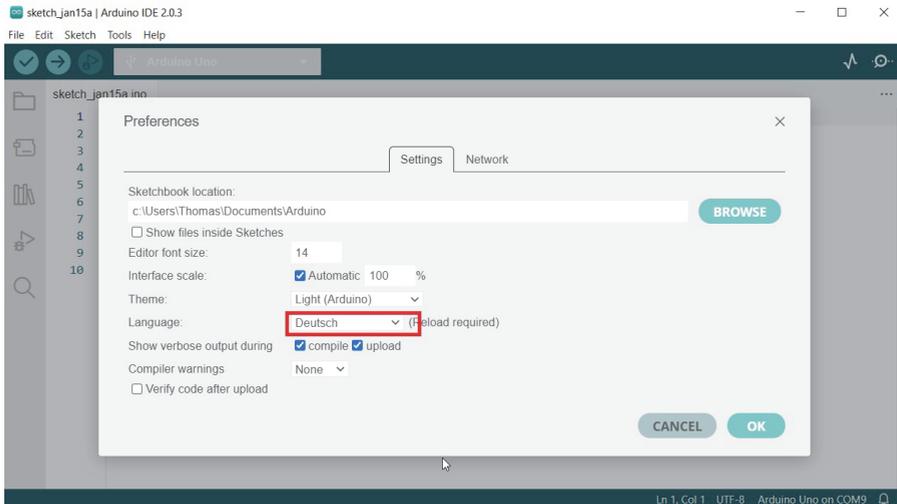


Abb. 2.13: Arduino IDE - Sprache

Nach der Bestätigung mit OK wird das Programm nochmals geladen und die Menüpunkte sind in Deutsch dargestellt. Nun muss in der Entwicklungsumgebung über den Menüpunkt WERKZEUGE|BOARD das verwendete Arduino-Board ausgewählt werden (Abbildung 2.14).

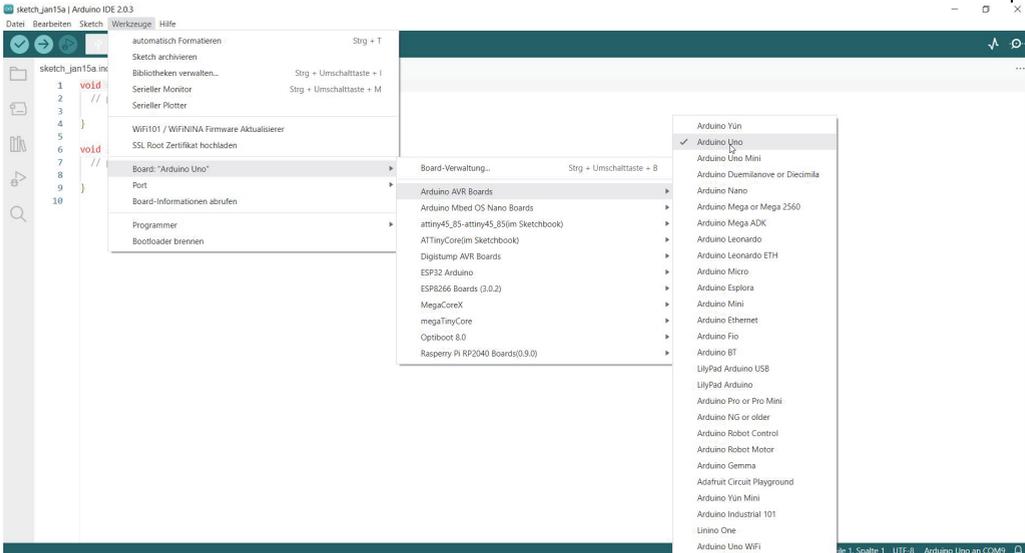


Abb. 2.14: Arduino-IDE – Board-Auswahl

Hinweis

Falls bei der erstmaligen Kompilierung eines erstellten Sketches eine Fehlermeldung erscheint, ist der Grund dafür oftmals eine falsche Board-Auswahl.

Im zweiten Schritt muss die vorher installierte virtuelle COM-Schnittstelle via USB-Port ausgewählt werden. Die Konfiguration erfolgt über den Menüpunkt WERKZEUGE|PORT (Abbildung 2.15).

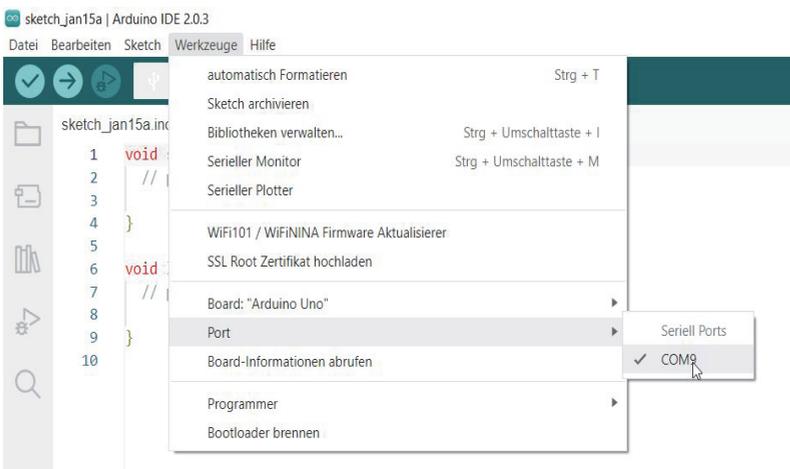


Abb. 2.15: Arduino-IDE – Auswahl COM-Port (Windows)

Zu beachten ist, dass die zu verwendende serielle Schnittstelle nur in der Auflistung der »Serial ports« erscheint, wenn das Arduino-Board mittels USB-Kabel mit dem Rechner verbunden ist. Andernfalls erscheint lediglich die Standardschnittstelle, die für diesen Einsatz nicht verwendet werden kann.

Hinweis

Nach der Erstinstallation und dem Start wird die Entwicklungsumgebung in englischer Sprache gestartet. Die Umschaltung der Sprache wird im nachfolgenden Abschnitt beschrieben.

Für macOS sieht die Auswahl der seriellen Schnittstelle wie in Abbildung 2.16 dargestellt aus.

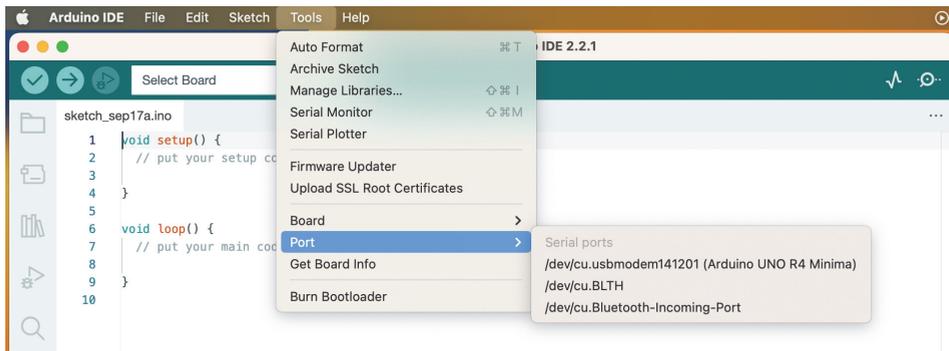


Abb. 2.16: Arduino-IDE – Auswahl Port (Mac OS)

Der Port ist mit `/dev/cu.usbxx` bezeichnet. Für den ersten Funktionstest und den Start des ersten lauffähigen Arduino-Programms sind in der Entwicklungsumgebung bereits verschiedene Testprogramme abgelegt.

Hinweis

Programme in der Arduino-IDE werden als »Sketche« bezeichnet und besitzen immer die Dateiendung `.ino`.

Öffnen Sie über DATEI|BEISPIELE|01.BASICS das Programm `Blink`. Dieses Testprogramm steuert den digitalen Ausgangspin 13 des Arduino an und lässt die bereits auf der Platine vorhandene Leuchtdiode (LED) im Sekundentakt blinken.

Nachdem das Testprogramm geöffnet wurde, wird es über den Menüpunkt SKETCH|ÜBERPRÜFEN/KOMPILIEREN kompiliert und anschließend über SKETCH|HOCHLADEN in den Microcontroller auf dem Arduino-Board geladen. Das erfolg-

reiche Hochladen wird in der Entwicklungsumgebung mit einer entsprechenden Erfolgsmeldung quittiert (Abbildung 2.17).

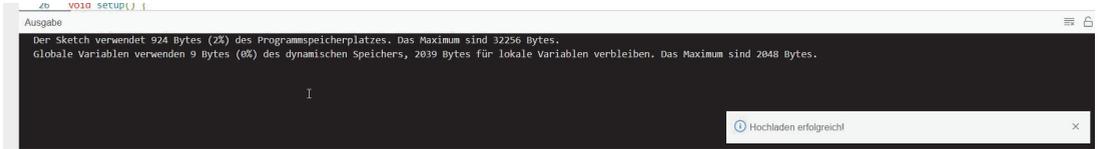


Abb. 2.17: Upload-Test Sketch

Auf dem Board blinkt nun die Leuchtdiode an Pin D13 (Abbildung 2.18).



Abb. 2.18: Arduino B1ink

Herzlichen Glückwunsch zu Ihrem ersten lauffähigen Arduino-Programm!

2.7.3 Projekt B1ink

Das erste Arduino-Programm B1ink eignet sich ideal für kleine Anwendungen, bei denen eine Leuchtdiode blinkt und so den Betrieb eines Geräts meldet. Damit kann beispielsweise eine Attrappe einer Alarmanlage realisiert werden, die durch Blinken der Leuchtdiode den aktiven Betriebsstatus anzeigt.

Abbildung 2.19 zeigt das Projekt B1ink, wobei eine externe Leuchtdiode auf einem Steckbrett angesteuert wird. Für das Projekt werden nur wenige Bauteile benötigt.

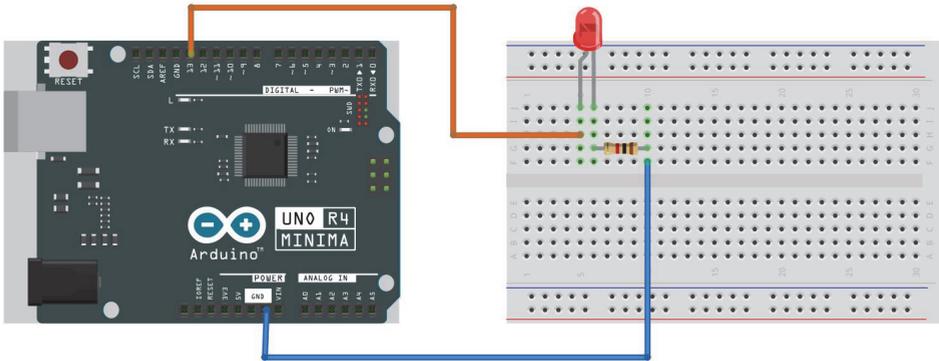


Abb. 2.19: Projekt Blink auf Steckbrett

Stückliste (Projekt Blink)

- 1 Arduino-Board
- 1 Steckbrett
- 1 Widerstand 1 kOhm (braun, schwarz, rot)
- 1 LED rot oder Farbe nach Wahl
- Anschlussdrähte

Sobald der Steckbrett-Aufbau gemäß Abbildung 2.19 erfolgt ist, leuchtet die rote LED im gleichen Takt wie die Onboard-Leuchtdiode an D13.

Glückwunsch zum ersten erfolgreichen Steckbrett-Projekt!

In Kapitel 3 werden die Grundlagen zu den Bauteilen und zum Steckbrett erklärt. Anschließend wird das Projekt Blink genauer untersucht.

2.8 Arduino-Entwicklungsumgebung

2.8.1 Voreinstellungen

Mit dem Ausführen des Blink-Testprogramms in der Arduino-Entwicklungsumgebung ist die IDE nun betriebsbereit und lauffähig.

Beim erstmaligen Start werden die Standardeinstellungen geladen. Diese Einstellungen definieren den Speicherort der Sketche (Sketchbook), die Sprache der Oberfläche, die Schriftgröße im Editor und weitere Einstellungen. Wer gerne in einem Dark-Modus arbeitet, kann diese Einstellung unter FARBDISEIGN auswählen.

Wie bereits erwähnt, ist die Sprache in der Entwicklungsumgebung beim allerersten Start auf Englisch gesetzt. Diese kann in den Einstellungen auf Deutsch umgeschaltet werden.

Die Einstellungen sind im Menü wie folgt zu finden:

Englisch: FILE|PREFERENCES

Deutsch: DATEI|EINSTELLUNGEN

Alle weiteren Erklärungen zur Oberfläche der Entwicklungsumgebung beziehen sich auf die deutschsprachige Darstellung.

In Abbildung 2.20 ist das Menü mit den Einstellmöglichkeiten dargestellt.

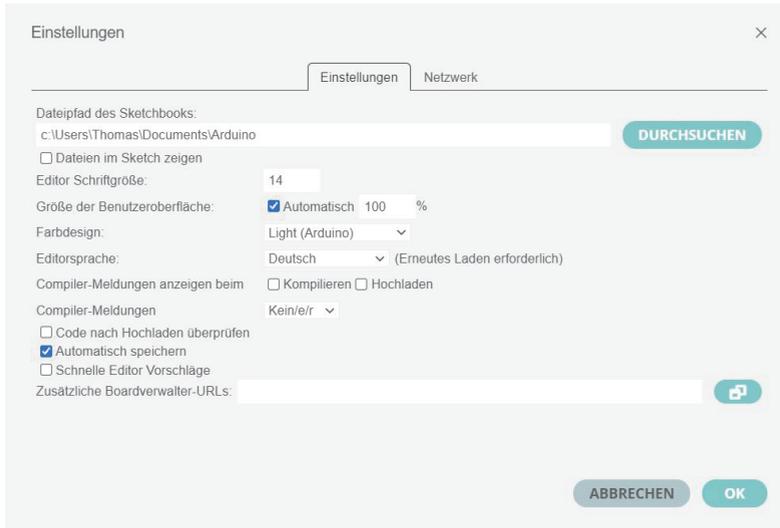


Abb. 2.20: Entwicklungsumgebung – Einstellungen

2.8.2 Aufbau Entwicklungsumgebung

Auf dem Bildschirm ist die Arduino-Entwicklungsumgebung in drei Teilbereiche gegliedert:

- Menü- und Symbolleiste
- Editor für Programmcode-Erstellung
- Ausgabefenster für Meldungen und Systeminformationen

2.8.3 Menü- und Symbolleiste

Die Symbolleiste der Arduino-Entwicklungsumgebung besteht aus mehreren grafischen Symbolen, über die man die Hauptfunktionen der IDE aufrufen kann.



Abb. 2.21: Arduino-IDE – Symbolleiste

Oberhalb der grafischen Symbolleiste ist die Menüleiste mit den Textlinks zu allen verfügbaren Funktionen sichtbar.



Abb. 2.22: Menüleiste

Überprüfen



Die Funktion ÜBERPRÜFEN kompiliert den aktuellen Programmcode und überprüft die Syntax sowie die Verfügbarkeit der im aktuellen Sketch verwendeten Bibliotheken (Libraries). Nach der Kompilierung wird im Konsolenfenster das Resultat angezeigt. Im Falle eines Fehlers wird eine Fehlermeldung mit Verweis auf die fehlerhafte Codezeile ausgegeben. Konnte der Programmcode fehlerfrei kompiliert werden, erscheint im Konsolenfenster eine Erfolgsmeldung sowie eine Angabe über die Größe des kompilierten Codes. Zusätzlich erfolgt eine Angabe über den maximal verfügbaren Speicherplatz im Arduino-Microcontroller.

Neben dem eigentlichen Programmcode benötigt der Arduino-Microcontroller zusätzlich Platz für das Bootloader-Programm. Der Bootloader benötigt nur rund 2 KB. Somit steht dem Anwender des UNO R4 noch genügend Platz für Programme zur Verfügung.

Beim Einsatz des ATmega328 (Arduino UNO R3) stehen dem Nutzer insgesamt nur 32 KB an Flashspeicher zur Verfügung. Nach Abzug des Platzbedarfs für den Bootloader sind bei diesem Microcontroller demnach rund 30 KB Speicherplatz für eigenen Programmcode vorhanden.

Hochladen



Mit der Funktion HOCHLADEN wird der aktuelle Sketch kompiliert und anschließend auf den Arduino-Microcontroller hochgeladen.

Während des Programmuploads blinken auf dem angeschlossenen Arduino-Board die Kommunikations-Leuchtdioden »RX« und »TX«. Nach erfolgreichem Upload werden eine Bestätigungsmeldung sowie die Angabe der Sketchgröße in der Konsole ausgegeben. Das Arduino-Board führt das hochgeladene Programm nach erfolgreichem Upload aus.

Debug



Die Funktion DEBUG wird für das Standardboard Arduino UNO noch nicht unterstützt.

Boardswahl



Mit Klick auf die Boardswahl öffnet sich ein Fenster, worüber das verwendete Board sowie der Port ausgewählt werden können.

Serieller Plotter



Der serielle Plotter erlaubt die Darstellung von analogen Signalen.

Serial Monitor



Der serielle Monitor ist als Symbol ganz rechts in der Symbolleiste aufrufbar oder kann über den Menüpunkt WERKZEUGE geöffnet werden (Abbildung 2.23).

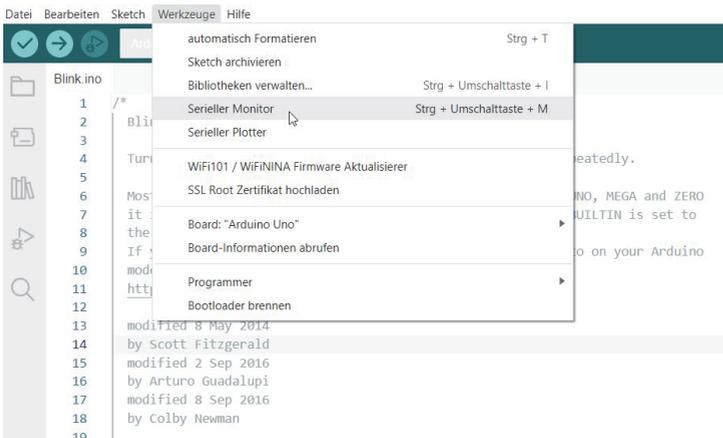


Abb. 2.23: Serieller Monitor

Nach Aufruf öffnet sich der serielle Monitor in einem zusätzlichen Fenster in der Entwicklungsumgebung. Hier wird die Kommunikation auf der seriellen Schnitt-

stelle dargestellt. Über ein Eingabefeld können Daten seriell an das Arduino-Board verschickt werden.

Der serielle Monitor eignet sich auch ideal als Hilfsmittel bei der Programmerstellung. In diesem Fall werden die Daten, Zustände oder Variablenwerte mittels Prozeduraufruf an den seriellen Port verschickt und sind im Infofenster sichtbar. Dieses Vorgehen ist vor allem beim Debugging von fehlerhaften Codezeilen und der Darstellung von an den Analogeingängen anliegenden Signalen sehr hilfreich.

Praxis-Tipp

Beim Neuerstellen eines Sketches über die Funktion »NEU« wird ein neues Arduino-Programm mit Grundstruktur erstellt. Die Grundstruktur beinhaltet nur die Funktionen `setup()` und `loop()`. Falls man jeweils eigene Einstellungen, Kommentare und Standarddefinitionen im Sketch verwenden möchte, muss man diese nicht manuell einfügen. Im Beispielsketch `01.Basics/BareMinimum` werden die Einstellungen für einen neuen Sketch eingetragen. Die Datei `BareMinimum.ino` muss über das Filesystem aufgerufen werden. Der Pfad zur Datei lautet:

```
Arduino-Programmpfad\resources\app\node_modules\arduino-ide-  
extension\Examples\01.Basics\BareMinimum\BareMinimum.ino
```

2.8.4 Editor

Der Editor ist das zentrale Element in der Arduino-Entwicklungsumgebung und dient zur Erstellung des Programmcodes. Funktionen wie Syntaxerkennung (SYNTAX HIGHLIGHTING), also das automatische Erkennen von einzelnen Codeanweisungen und die farbige Darstellung dieser Anweisungen, unterstützen den Programmierer bei der Code-Erstellung. Zusätzlich zeigt der Editor beim Programmieren zusammengehörige Klammerpaare an (Abbildung 2.24).

```
void loop() {  
  if (irrecv.decode(&results)) {  
    // ...  
    digitalWrite(ledPin1, HIGH); // Taste 1 LED1 EIN  
  }  
}
```

Abb. 2.24: Editor mit Code Syntax Highlighting

Die zentrale Ablage der Sketche erfolgt im Sketchbook, also dem Skizzenbuch. Für jeden Sketch wird ein eigener Unterordner erstellt. Über das Menü FILE|SKETCHBOOK kann man die bereits erstellten Sketche aufrufen.

Der physische Ablageort des Sketchbooks wurde im vorherigen Abschnitt beschrieben.

Beim Neuanlegen eines Sketches vergibt der Editor jeweils einen eigenen Dateinamen. Um die Übersicht zu behalten, sollten Sie eine eigene Namenskonvention verwenden. Beim Experimentieren kommt schnell eine größere Anzahl von Sketchen mit unterschiedlichen Versionen zustande.

2.8.5 Ausgabefenster

Das Ausgabefenster ist der Bereich mit dem schwarzen Hintergrund unterhalb des Editors. Dieser Bereich dient zur Ausgabe von Informationen des Systems an den Benutzer. Das Ausgabefenster ist nur ein Informationsbereich, der Benutzer kann hier keine Eingaben vornehmen.

Zu beachten ist, dass das Ausgabefenster in früheren Versionen der Entwicklungsumgebung auch die Konsole für den seriellen Monitor beinhaltete. Diese Funktion wird später erklärt und ist in der aktuellen Version der IDE in einem zusätzlichen Fenster untergebracht.

Meldungen über erfolgreiche Aktionen mit dem Arduino-Board, wie eine erfolgreiche Kompilierung oder ein Sketch-Upload, werden als Info ausgegeben (Abbildung 2.25).



Abb. 2.25: Ausgabe nach erfolgreicher Kompilierung

Meldungen über eine fehlerhafte Programmierung oder eine fehlende Schnittstelle werden im Ausgabefenster in leuchtenden Farben dargestellt (Abbildung 2.26).



Abb. 2.26: Ausgabefenster mit Fehlermeldung

Bei der Kompilierung von fehlerhaftem Programmcode erscheint wie beschrieben eine Fehlermeldung im Ausgabefenster.

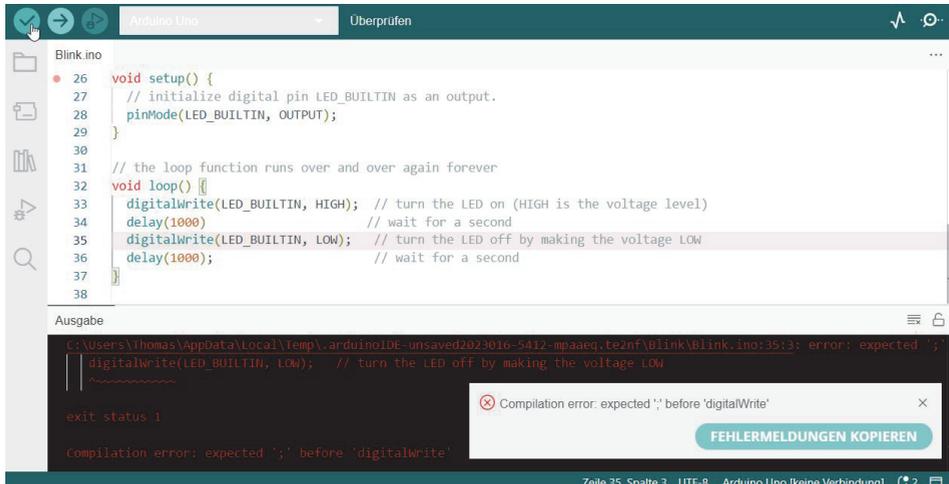


Abb. 2.27: Ausgabe bei Kompilierung mit Fehler im Code

In Abbildung 2.27 wird auf ein fehlendes Semikolon vor der Anweisung `digitalwrite` hingewiesen. Die Fehlermeldung informiert den Programmierer hier über diesen Fehler und macht eine ungefähre Angabe zur Position der fehlerhaften Zeile. Der eigentliche Fehler ist dann in Zeile 34 zu finden.

Hinweis

Leider sind die Fehlermeldungen nicht immer sehr aussagekräftig. In diesem Fall muss der Programmierer den fehlerhaften Code mit weiteren Mitteln debuggen. Das Debuggen von fehlerhaftem Arduino-Code wird in Kapitel 3 näher beschrieben.

2.8.6 Bibliotheken verwalten

Software-Bibliotheken (englisch: libraries) sind in sich geschlossene Programme mit zusätzlichen Funktionen. Diese Bibliotheken erweitern den Funktionsumfang eines Arduino-Boards auf einfache Art und Weise, ohne dass man den Code für eine Funktion selber neu entwickeln muss.

Im Internet gibt es viele spezifische Bibliotheken für Arduino, die von engagierten Entwicklern geschrieben wurden. In der Arduino IDE ist bereits eine Anzahl von Standardbibliotheken integriert. Unter WERKZEUGE|BIBLIOTHEKEN VERWALTEN oder über die Schnellauswahl am linken Rand kann man nach Bibliotheken suchen und neue Bibliotheken hinzufügen (Abbildung 2.28).

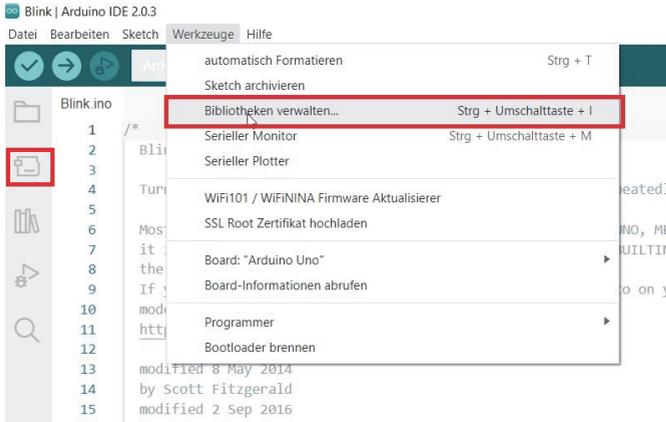


Abb. 2.28: Arduino IDE - Bibliotheken verwalten

Nach Aufruf der Funktion öffnet sich in der linken Spalte ein neuer Bereich mit der Suchfunktion für Bibliotheken (Abbildung 2.29).

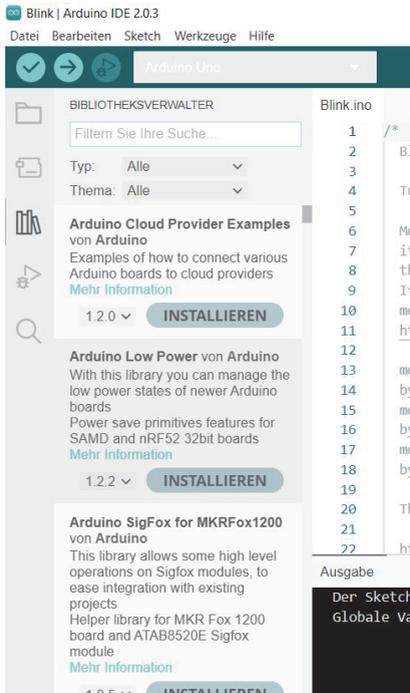


Abb. 2.29: Arduino IDE – Bibliotheken verwalten

Die integrierte Suche der Bibliotheksverwaltung erlaubt die Suche nach verfügbaren Bibliotheken. Im Beispiel wird eine Bibliothek für GPS gesucht (Abbildung 2.30).

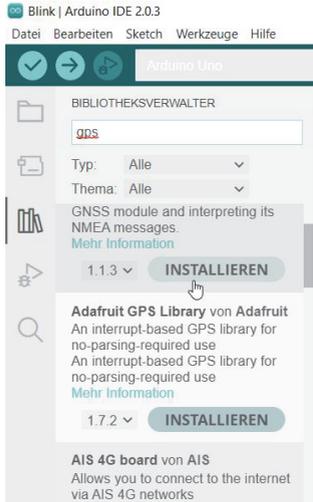


Abb. 2.30: Arduino IDE – Bibliotheken suchen

Die gewünschte Bibliothek kann anschließend über die Funktion **INSTALLIEREN** ins System integriert werden. Falls Abhängigkeiten zu anderen Bibliotheken vorhanden sind, werden diese vor der Installation angezeigt. Im Falle der GPS-Bibliothek besteht eine Abhängigkeit zur SD-Card-Bibliothek. Dies wird in einem Hinweisfenster angezeigt (Abbildung 2.31).



Abb. 2.31: Arduino IDE – Bibliotheken verwalten

Nach erfolgreicher Installation der Bibliothek können verfügbare Funktionsbeispiele über **DATEI|BEISPIELE** aufgerufen werden. Beispiele von hinzugefügten Bibliotheken findet man dabei im Bereich »Beispiel aus eigenen Bibliotheken« (Abbildung 2.32).



Abb. 2.32: Arduino IDE – installierte Bibliotheken

Der Code der Bibliotheken wird im Sketch-Ordner im Unterordner »libraries« gespeichert (Abbildung 2.33).

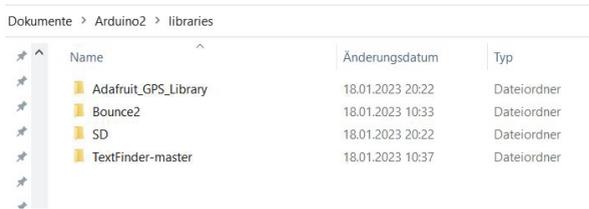


Abb. 2.33: Arduino IDE – installierte Bibliotheken

Viele Funktionsbibliotheken für Arduino findet man über die Internet-Suche auf den verschiedensten Plattformen. Hier wird die Bibliothek meist als Paket im ZIP-Format angeboten. Unter SKETCH|BIBLIOTHEK EINBINDEN|ZIP-BIBLIOTHEK HINZUFÜGEN kann man die gepackte Bibliothek in die Entwicklungsumgebung installieren (Abbildung 2.34).

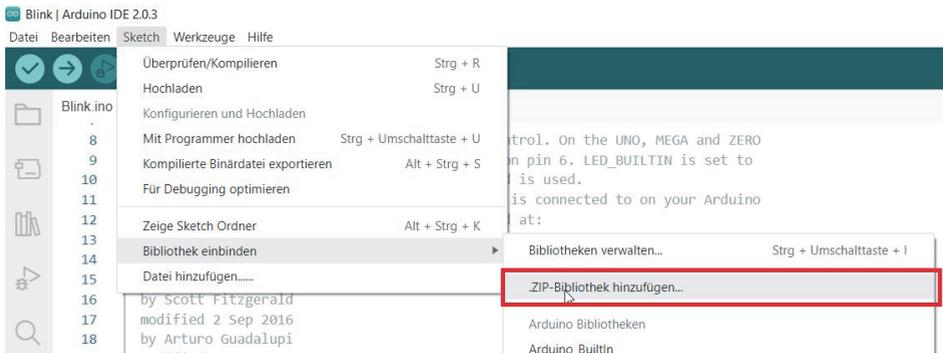


Abb. 2.34: Arduino IDE –Bibliothek als ZIP hinzufügen

Die ZIP-Datei kann auch entpackt und dann direkt im Bibliotheks-Ordner abgespeichert werden. Weitere Informationen zu Bibliotheken findet man im Kapitel 9.

2.9 Arduino-Boards

Neben dem Standardboard Arduino UNO sind im Arduino-Projekt in der Kategorie »Classic Family« etliche weitere Boards verfügbar, die über den Arduino-Shop oder einen Hardwarelieferanten gekauft werden können. Zusätzlich zu den klassischen Arduino-Boards sind viele weitere Boards mit Sonderfunktionen verfügbar.

Nachfolgend werden einige klassische Boards für den Einsteiger vorgestellt. Einen Vergleich der gängigsten Arduino-Boards finden Sie in Anhang B. Alle aktuellen Arduino-Boards sind jeweils im Hardwarebereich der Arduino-Website aufgelistet:

<http://arduino.cc/en/Main/Hardware>

2.9.1 Arduino Leonardo

<http://arduino.cc/en/Main/ArduinoBoardLeonardo>

Der Arduino Leonardo wurde im Mai 2012 vorgestellt. Das Board basiert auf einem ATmega32u4-Microcontroller. Dieser neue Microcontroller übernimmt auch die Kommunikation mit der USB-Schnittstelle. Das Leonardo-Board kann als USB-Device genutzt werden und eine Maus, eine Tastatur oder einen Joystick emulieren.

Die volle Unterstützung dieses Boards ist mit der IDE-Version 1.0.1 und höher vorhanden. Dazu stehen in der Entwicklungsumgebung etliche Beispiel-Sketches für die neue USB-Device-Funktionalität zur Verfügung.

2.9.2 Arduino Mega 2560

<http://arduino.cc/en/Main/ArduinoBoardMega2560>

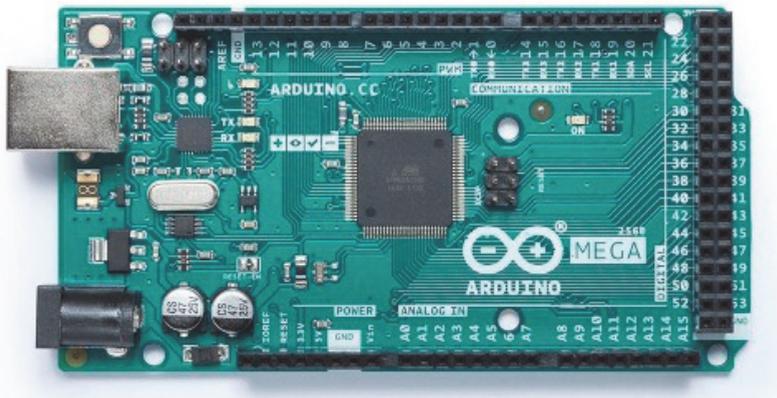


Abb. 2.35: Arduino Mega 2560 (Quelle: www.arduino.cc)

Der Arduino Mega ist die Powervariante unter den Arduino-Boards. Die aktuelle Mega-Version heißt Arduino Mega 2560 und ersetzt die bisherige Version Arduino Mega. Der Arduino Mega 2560 wird von einem ATmega2560 gesteuert und besitzt 54 digitale Ein- und Ausgänge, von denen 15 Ports als PWM eingesetzt werden können. Zusätzlich stehen dem Anwender 16 analoge Ports und 4 serielle Schnittstellen zur Verfügung.

Der Arduino Mega wird eingesetzt, wenn die Anzahl der benötigten Ports die Möglichkeiten eines Arduino Uno überschreitet (beispielsweise bei Projekten mit einer Vielzahl von LED-Anzeigen) oder der Aufwand für externe Porterweiterungen zu hoch ist.

Da die Anordnung der Steckerleisten bei Arduino Uno und Arduino Mega identisch ist, sind die meisten Standard-Shields (Erweiterungsplatinen) weiterhin verwendbar.

Beschreibung	Detailldaten
Microcontroller	ATmega2560
Spannungsversorgung	7–12 VDC
Betriebsspannung	5 VDC und 3,3 VDC (intern durch FTDI-Controller generiert)
Digitale Ein-/Ausgänge	54 (davon 15 als PWM-Ausgänge)
Analoge Eingänge	16
Strom pro digitalem Pin	40 mA DC
Flash Memory	256 KB wobei 8 KB vom Bootloader belegt werden
SRAM	8 KB
EEPROM	4 KB
Taktfrequenz	16 MHz
USB-Schnittstelle	ja
Reset-Taster	ja
Onboard-ICSP-Stecker	ja

Tabelle 2.3: Technische Daten Arduino Mega 2560

2.9.3 Arduino Nano

<https://store.arduino.cc/products/arduino-nano>

Wie der Name bereits sagt, ist das Arduino Nano eine Leiterplattenversion von geringer Baugröße. Seine Abmessungen betragen 18,5x43 mm. Auf dem Board stehen alle nötigen Komponenten inklusive USB-Schnittstelle und Spannungsregler zur Verfügung.

Das Arduino Nano besitzt dieselbe Funktionalität wie ein Standard-Board, aber statt sechs besitzt es sogar acht analoge Ports. Durch den Aufbau der beiden Stiftreihen am Rand kann das Arduino Nano ideal beim Testen der Entwicklung auf Steckbrettern (Breadboards) eingesetzt werden.

Neben dem klassischen Arduino Nano stehen im Arduino-Shop in der Kategorie »Nano-Family« weitere Modelle zur Verfügung. Zu erwähnen wäre hier der

soeben herausgebrachte Arduino Nano ESP32 mit Wifi-Schnittstelle für einfache und kostengünstige IoT- und Web-Anwendungen.

<https://store.arduino.cc/products/nano-esp32>

2.9.4 Arduino Mini 05

<https://docs.arduino.cc/retired/boards/arduino-mini-05>

Auch der Arduino Mini 05 (Abbildung 2.36) ist eine Arduino-Variante von geringer Baugröße. Das Board wurde in der Vergangenheit als Arduino Mini bezeichnet. Zwischenzeitlich ist der Mini 05 abgekündigt, aber bei vielen Händlern sind diese Boards und Nachbauten (Clones) meist zu günstigen Preisen verfügbar.

Die Abmessungen des Boards betragen 20x30 mm und passen auf einen DIL24-Sockel. Der Arduino Mini 05 hat dieselbe Portanzahl wie das Arduino Nano. Im Gegensatz zur Nano-Version besitzt er jedoch keine USB-Schnittstelle mit dazugehörigem USB-Anschluss.

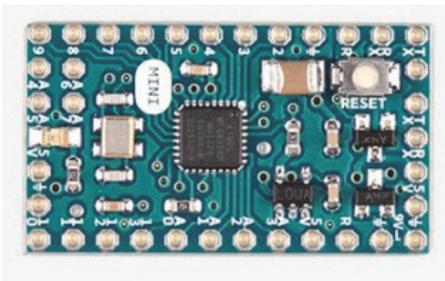


Abb. 2.36: Arduino Mini

Die Programmierung des Arduino Mini 05 erfolgt über die serielle Schnittstelle. Für den Einsatz muss ein passender USB-Adapter (beispielsweise der Mini-USB-Adapter) verwendet werden.

Das Arduino Mini hat sein Einsatzgebiet in Kleinstanwendungen oder als Steckmodul auf einer größeren Leiterplatte oder bei Anwendungen auf dem Steckbrett.

2.10 Arduino-kompatible Boards

Neben den originalen Arduino-Boards aus Italien gibt es auf dem Markt viele mit Arduino UNO R3 kompatible Boards, die von Makern und Hardware-Entwicklern realisiert wurden. Diese sogenannten Arduino-Clones basieren auf der Schaltung des Original-Arduino und haben oftmals die gleichen Abmessungen wie das Original. Durch unterschiedliche Farben der Leiterplatten und einer eigenen Board-Bezeichnung sind diese Board-Varianten meist gut als Arduino-Clones erkennbar. Einige dieser Arduino-kompatiblen Boards werden in Kapitel 12 genauer vorgestellt.

Stichwortverzeichnis

- 24LC64 355
- 2,1-mm-Power-Jack 53, 55
- 2-Wire 139
- 2-Wire-Bus 147
- 433-MHz-Technologie siehe 433 MHz
- 5-mm-Buchse 61
- 64-Bit-Adresse 205
- 74HC595 313
- 7-Segment-Anzeige 309
- 8-Bit-Modus 317
- .ino siehe Dateierdung
- A**
- A3144 240
- A/D-Wandler 117
- AA-Batterien siehe Stromversorgung
- abs() 586
- Abstandssensor 230
- Addition 589
- Adresse
 - I2C-Bus 208
- Aktor 189, 261
- Analog/Digital-Wandler 117
- Analoge Signale 118
- Analoge Welt 117
- analogRead() 118
- Analogskala 270
- Analogwert 267, 384
- analogWrite() 125
- and 590
- Anode 309
- Anschlussbelegung
 - Servo 265
 - Wii Nunchuk 410
- Anschlusskabel
 - Wii 288
- Anschlussmöglichkeiten siehe Arduino UNO
- Anschlusspin 77
- Anschlussschema
 - Microcontroller 603
- Ansteuerung
 - LED 299
 - Relais 282
- Anweisung
 - wiederkehrend 568
- Anzeige 189
- Anzeigeelement 303
- Arduino 18
 - Beispielsammlung 49
 - Board 50
 - Bootloader-Programm 20
 - Community 391
 - Entwicklungsumgebung 18, 36
 - Ethernet-Schnittstelle 365
 - externe Stromversorgung 22
 - Flash Speicher 351
 - Forum 49
 - Internetanwendungen 413
 - Mega 46
 - Mini 05 48
 - Minimalschaltung 497
 - Nano 47, 48
 - Plattform 18
 - Programmiersprache 80
 - Reset-Schaltung 497
 - Sketch-Struktur 85
 - Stromlaufplan 50
 - Stromversorgung 56
 - Webclient 422
 - Webserver 419
- arduino_secrets.h 429
- Arduino-Board
 - Zuordnung Pins Microcontroller 603
- Arduino Clone
 - Sippino 503
- Arduino-Clone
 - Helvetino 501
- Arduino-Clones siehe Clones
- Arduino ISP 510
- Arduino-Kompass siehe Kompass
- Arduino Leonardo 46
- Arduino UNO R3 24
- Arduino UNO R4 20
- Arduino UNO R4 WIFI 414, 427, 428
- Arithmetik 589
- Array
 - abfragen 578
 - Datentyp 577
 - Größe 577
- array 83
- Asterisk
 - Zeigertabelle 576

- ATmega
 - Microcontroller 20
- ATmega168
 - Pinbelegung 605
- ATmega2560 46
 - Microcontroller 601
- ATmega328 26, 52
 - Microcontroller 601
- ATmega32u4 46
- attachInterrupt() 594
- ATtiny
 - blink 520
 - Serial() 518
 - serielle Schnittstelle 518
 - Taktfrequenz 520
- ATtiny45 514
- ATtiny84 514
- ATtiny ISP Shield 527
- Auflösung 121
 - 12 Bit 121
 - A/D-Wandler 118
- Ausgabefenster 41
- Ausgabeformat
 - serielle Schnittstelle 598
- Ausgabemöglichkeit
 - serielle Schnittstelle 598
- Ausgang
 - analog 592
 - digital 591
 - setzen 114
- Außentemperatur 200
- available() 597
- B**
- Balkengrafik 308
- Bananenstecker 61
- Barebone Breadboard 499
- Basisstrom 283
- Basiswiderstand 283
- Bauelement
 - gepoltes 73
- Bedieneinheit 285
- begin() 596
- Beispielskript 12
- Berechnungsformel 70
 - Widerstandsleiterschaltung 371
- Berührungssensor 303
- Beschleunigung 231
- Beschleunigungssensor 231
- Bezugsquelle 607
 - Arduino-Boards 20
 - Einzelkomponente 607
- Bibliothek 137, 391
 - Bounce 108
 - PString 397
 - Stepper 398
 - TinyGPS 396
- Biegelehre 543
- Binärwert 574
- Bitmuster 309
- bitRead() 311
- Black Wing Adapter
 - Spannungsversorgung siehe Steckbrett
- Blink siehe Projekt Blink
- Blockbatterie 286
- BME280 225
- Board 18, 50
 - Boardvarianten 601
- Boards
 - Arduino kompatibel 48
- Bodenfeuchtigkeit 217
- Boolean
 - Datentyp 574
- boolean 83
- Bootloader 20, 52
- Bounce-Bibliothek 108
- Bouncing 107
- Breadboard 58, 496
- break 585
- Breakout-Board 147, 162
- Buchsenleiste 53, 55
- Busadresse 355
- Busteilnehmer 141
- Byte 573
- byte 82
- C**
- C++ 19
- CAD 20
- CAM-Jobs 562
- case 584
- Char
 - Datentyp 574
- char 82
- Clone 20
- Code-Debugging 90
- Common Anode 313
- Community-Website 17
- constrain 587
- continue 585
- Continuous Rotation Servo 274
- cos 588
- D**
- DAC siehe Digital/Analog-Wandler
- Datalogger 359
- Datenformat 560
 - Gerber 560
- Datenleitung 318
- Datenlogger 351, 361

Datentyp 81, 572
 Datentypkonvertierung 580
 Debuggen 87, 494
 Debug-Methode 92
 define
 Definitionsanweisung 571
 Definitionsanweisung
 define 571
 delay() 595
 delayMicroseconds() 595
 detachInterrupt() 595
 Dezimalpunkt 309
 DFRobot Motor Shield 405
 DHCP 418
 DHT11 221
 DHT22 221
 Digital/Analog-Wandler 129, 366
 DAC 129
 Digitalanzeige 544
 Digitaler Ausgang 114
 Digitaler Eingang 100
 entprellen 107
 lesen 102
 Pullup 101
 digitalRead() 116
 digitalRead(Pin-Nummer) 102
 digitalWrite() 115
 DIL 78
 DIL24-Sockel 48
 Dimmer 302
 Diode 73, 299
 Display-Controller 316
 Division 589
 DIY Shield 401
 do ... while 584
 Double
 Datentyp 574
 double 82
 Drahtlose Kommunikation 152
 Drain 284
 Drehbegrenzung 375
 Drehbereich 72
 Drehgeber 374
 Drehposition 265
 Drehrichtung
 Motor 275
 Druckschalter 230
 Drucksensor 351
 Drucktaster 80, 181
 DS1054 548
 DS1054Z 549
 DS1307 148
 DS1820 204
 DSO 547
 Dual In-Line siehe DIL
 Durchlassspannung 74, 299

E

Eagle
 CAD 556
 Format 557
 Gerber 562
 Editor 40
 EEPROM
 I2C-Baustein 355
 lesen 351
 schreiben 351
 Speicherplatz 354
 Eingang 99
 Eingangspuls
 Puls messen 592
 Eingangswiderstand
 digitaler Eingang 100
 Einzelfunktion 90
 Einzelkomponente
 Bezugsquelle 607
 Elektrolytkondensator 73
 Elektromotor 274
 Elektronikbauteil 70
 Elektroniklabor 49
 Elektronik-LötKolben 542
 elektronischer Schaltplan siehe Stromlauf-
 plan
 Elektrotechnik 66
 Grundkenntnisse 49
 else 583
 Empfangsbuffer
 löschen 599
 serielle Schnittstelle 597
 Empfangspuffer
 serielle Schnittstelle 133
 Empfangssignal 99
 Encoder siehe Rotary-Encoder
 end() 597
 Endschalter 101, 107
 Entprellen
 Tiefpass 108
 Entwicklungsumgebung 18
 Ausgabefenster 41
 Download 27
 Editor 40
 Hochladen 38
 IDE 36
 kostenlos 15
 macOS 29
 Menü- und Symbolleiste 37
 serieller Monitor 39
 Software 27
 Verifizieren 38
 Windows 29
 Erweiterung
 Schraubklemmen 409
 Terminal Block Shield 409

Ethernet-Anwendung 404
 Ethernet-Bibliothek 392, 413
 Ethernet-Controller 404
 Ethernet-Einstellungen 419
 Ethernet-Schnittstelle 365
 Ethernet Shield 400, 404, 413
 Experimentierplatine 58
 Lochrasterplatinen 540
 Experimentierschaltung 496
 externe Versorgung siehe Stromversorgung

F

Fahrgeschwindigkeit siehe Fahrradcomputer
 Fahrradcomputer 336
 false 574
 Fehler
 Programm 494
 Schaltung 493
 Fehlermeldung
 kompilieren 494
 Fehlerquelle 493
 Fehlersuche 493
 Feldeffekttransistor 282
 FET 282
 Feuchtesensor 216
 Fieberthermometer 257
 Flachzange 543
 Flash Memory
 Arduino Boards 601
 Flash-Speicher 351
 Fließkommazahl 573
 Float
 Datentyp 573
 float 82
 Floweronly 403
 Flowerpads 502
 flush() 599
 Flüssigkristallanzeige siehe LC-Display
 for 583
 Fotowiderstand 187, 190
 Freilaufdiode 74, 261
 Fritzing 553
 Code 553
 Gerber-Export 561
 Leiterplatte 553
 Schaltplan 553
 Steckplatine 553
 FTDI 498, 504
 Anschlussbelegung 506
 FTDI-Kabel 505
 Funksteckdosen 152
 Funktion 83
 Aufbau 568
 Aufruf 568
 mathematische 586
 Funktionsaufruf 83
 Funktionsdefinition 568

G

Galvanisch getrennt 263
 Ganzzahlig 572
 Arduino 217
 Gateway 414
 Gebäudesteuerung 261
 Gefrierschrankwächter 176
 Gepoltes Bauelement 73
 Geräte-Manager 495
 Gerber 560
 Gerbv 563
 Geschwindigkeit
 Motor 275
 Geschwindigkeitsmesser 336
 Gesetz
 Spannung 49
 Strom 49
 Widerstand 49
 Gleichstrommotor 274
 Glühbirne 308
 GPS-Modul 351, 405
 GPS Shield 405
 GPSVisualizer 405
 Grundfrequenz
 PWM 593
 Grundkenntnis
 Elektrotechnik 49
 Grundschialtung
 Ansteuerung LED 299
 Grundstruktur 567

H

Halbleiterrelais 263
 Halbleiterübergang 304
 Hall-Sensor 239
 Halogenlampe 282
 Hardwareerweiterung siehe Shield
 Hardwareteil 18
 Hauptprogramm 84
 Loop() 568
 HD44780 316
 Heißleiter 192
 Helligkeit
 LED 302
 Helligkeitssteuerung 309
 Helvetino 501
 HIGH
 Konstante 582
 Hilfsmittel 539
 HMC6352 246
 Hyperterminal 91

I

I2C 139
 Protokoll 140
 I2C-Bus 140

- I2C-Scanner 216
 - IC 58, 77
 - ICSP 22, 53, 55, 507
 - IDE 18
 - if 583
 - In-Circuit Serial Programming siehe ICSP
 - Induktiver Schalter 230
 - Infrarot 75
 - Input
 - Konstante 582
 - Installation
 - Linux 28
 - macOS 28
 - Software 27
 - Windows 28
 - Instructables 18
 - int 82
 - Integer 81
 - Datentyp 572
 - Integrierter Temperatursensor 196
 - Integrierte Schaltung siehe IC
 - Interaction Design Institute Ivrea
 - IDII 15
 - Internet Protocol siehe IP
 - Interrupt 594
 - CHANGE 594
 - FALLING 594
 - LOW 594
 - Mode 594
 - Nummer 594
 - RISING 594
 - Intranet 414
 - IP 414
 - IP-Adresse 414
 - IR-Diode 75
 - Ivrea 15
- J**
- Java
 - Entwicklungsumgebung 27
 - JeeNode 164
 - Jumper-Wire 59
- K**
- Kaltleiter 195
 - Kapazitiver Schalter 230
 - Kathode 309
 - Kathodenstrahlröhre 546
 - KiCad 557
 - Kiloohm 72
 - Klammer
 - geschweifte 569
 - runde 569
 - Klammern
 - eckige 569
 - KML-Format 405
 - Kolophonium
 - Lötzinn 543
 - Kommentar 571
 - einzeilig 571
 - Kommentarblock 571
 - Kommunikation 19
 - serielle 131
 - Kompass 234
 - Kompass-Sensor 234
 - Kompilieren 41, 90
 - Fehlermeldung 42
 - Komponente 19
 - Kondensator 73
 - Konfiguration
 - Setup() 567
 - Konstante 581
 - Konstantendeklaration 581
 - Konstantstromquelle 300
 - Spannungsregler 301
 - Transistor 300
 - Kontaktlose Sensor siehe MLX90614
 - Kontrollstruktur 583
 - Konvention
 - Programmierung 569
 - Kreativität 16
 - Kühlung 283
- L**
- L293 276
 - Labornetzgerät 494
 - Lagerung 49
 - Lampe 261
 - Last
 - schalten 282
 - Layoutdaten 557
 - LC-Display 316
 - Ansteuerung 316
 - Bibliothek 318
 - HD44780 316
 - LDR 190
 - Widerstandswert 190
 - LED 74, 299, 303
 - LED-Matrix 335, 428
 - Leiterbahn 58
 - Leiterplatten 50, 559, 564
 - OSH Park 564
 - Prototypen 559
 - Leiterplattenversion 47
 - Leuchtdiode siehe LED
 - Library
 - Bibliothek 137
 - Lichtemission 75
 - Lichtstrom 308
 - Lieferant siehe Bezugsquelle
 - LiquidCrystal 317
 - lcd() 318

- Lithium Backpack 406
- LM317 301
- LM335 200
- LM35 197
 - Temperatursensor 122
- LM75 147, 208
- Lochmaster 540
- Lochrasterplatine 58, 540
 - Lochstreifenplatine 540
- Logischer Operator 590
- Long
 - Datentyp 573
- long 82
- loop() 85
- Loop-Funktion
 - Loop() 568
- Lötgerät 78
- LötKolben 49, 542
- Löt-punkt 58
- Lötstation
 - geregelt 543
- Lötzinn 543
- LOW
 - Konstante 582
- Luftdruck 225
- Lufterfrischer 16
- Luftfeuchtigkeit 217
- M**
- MAC-Adresse 414
- macOS 28
- Magician Chassis 288
- Magnet 79, 261
- Magnetismus siehe Hall-Sensor
- Maker 16
- Maker Faire 17, 18
- map 587
- map() 267
- Master
 - I2C-Bus 140
- Matrixanzeige 313
- max() 586
- MCP4725 384
- Mega
 - Board 601
- Megaohm 72
- Memory Card 357
- Messbereich
 - Multimeter 545
- Messgerät 49
- Messwerterfassungssystem 351
- Metalloxid-Halbleiter-Feldeffekttransistoren
 - siehe MOSFET
- Microcontroller 18, 52
 - Anschlussbelegung 52
 - Systeme 15
- micros() 596
- millis() 595
- min() 586
- Mindstorms 211
- Mini 05 48
- Minimalschaltung 497
- Miniroboter
 - Servo 274
- Mini-USB-Adapter 48
- MintyBoost 57
- Mittelwert 202
- MLX90614 257
- Modellbauservo 264
 - PWM 265
- Monitor
 - serieller 87
- MOSFET 283
- Motor 261, 274
- Motor Shield 282
- Motorsteuerung 279
- Multimeter 544
- Multiplikation 589
- N**
- Nachkommastelle
 - Float 573
- Nano 47
- Native USB 132
- Neigung 231
- Networked Cat 189
- Netzgerät 49
 - programmierbares 372
- Netzteil 61
- Netzwerkverbindung 414
- Nokia 3310/5110 327
- not 590
- notone() 593
- NPN 76
- NTC 192
- O**
- Oberflächenmontage 78
 - SMD 147
- Ohm 72
- Ohmsche Gesetz 66
- OLED 329
- OneWire
 - Bibliothek 205
- Opendrain 313
- Open Source
 - Boards 20
- Operator
 - logischer 590
- Optokoppler 112
- or 590
- OSH Park 564

- Oszilloskop 546
- Output
 - Konstante 582
- P**
- Parallelschaltung 69
- Parameter 83
 - Funktionsaufruf 568
- PCF8574 313
- PCF8583 siehe DS1307
- PDC8544 327
- Perma-proto 499
- Pflanzenbewässerung 16
- Piezoelement 345
- Pinbelegung
 - Microcontroller 605
- PING))) Ultrasonic Sensor 230
- PING-Sensor 231
- pinMode
 - Input 591
 - Output 591
- pinMode() 591
- Plattform 19
- PNP 76
- Port
 - PWM 593
- Porterweiterung 313
- Port Expander 313
- Portmanipulation 369
- Portnummer 495
- Potentiometer 72, 267
- pow() 588
- Power-Leuchtdiode 282, 308
- preferences.txt 495
- Prellen 107
- Prellverhalten 108
- print() 598
- println() 599
- Problem
 - COM-Port 495
 - IDE 495
- Processing 18
- Programmablauf 87
- Programmcode 80
- Programmierung
 - Konvention 569
- Programmstruktur 567
- Projekt Blink 35
- Protonly 402
- Protoshield 92, 187, 400
 - Floweronly 403
 - Protonly 402
- Prototyp 16
 - Aufbau 16
- Prototyp-Leiterplatten 499
- Prüfprogramme 86
- Prüfung 90
- PString 397
- PTC 195
 - Grundschialtung 196
- Pulsweite 265
- Pulsweitenmodulation 99, 593
- Pulszeit 118
- Pushbutton siehe Drucktaster
- PWM 593
 - Ausgangsfrequenzen 126
 - Ports 126
 - Signal 125
- Q**
- Quarz 73
- R**
- random() 588
- randomSeed 588
- read() 598
- Receive (rx) 137
- Rechteck 129
- Reed-Kontakt 230
- Reed-Relais 230
- Referenzspannung 117
- Regelsystem 261
- Relais 79, 261
- Relaiskontakt 261
- RESET 53, 55
- return 586
- RFBoard 503
- RFID-Reader 351
- RF-Komponente 298
- RFM12B 161
- RFM12B-Board 175
- Richtungsänderung 231
- Richtungssteuerung 279
- Rigol 548
 - DS1052 549
- Roboter 269, 285, 286
 - Stückliste 286
- Roboterfahrzeug 285
- Rotary-Encoder 374
 - Signale 375
- Router 414
- Row-Column Scanning 335
- RS232 131
- RS-274X siehe Gerber
- RTC 428
 - Real Time Clock 428
- Rückgabewert 83
 - Funktion 568
 - Typ 568
- S**
- Sägezahn 129
- Schalter 79

- Schaltersensor 230
- Schaltkontakt 282
- Schaltkreis 49
- Schaltrelais 261
- Schaltransistor 282
- Schaltungsaufbau 49, 60
- Schaltungsentwicklung 16
- Schließer 261
- Schnittstelle
 - Eingänge und Ausgänge 99
 - serielle 596
- Schraubklemme 408
- Schreib- und Lesezyklus 354
- Schrittmotor 274
 - Ansteuerung 399
- Schrittzähler 345
- SCL 140
- SDA 140
- SD-Card-Schaltung 364
- SD Card Shield 406
- SD-Karte 357
- Segment
 - 7-Segment-Anzeige 309
- Seitenschneider 543
- Selbstbausensoren 217
- Semikolon 571
- Senden (tx) 137
- Sendesignal 99
- Sensor 189
- Sensorfeedback 308
- Serial1 132
- SerialLCD 320
- Serial Monitor 39
- Serielle Kommunikation 131, 596
- Serielle Schnittstelle 131
 - native Schnittstelle 132
 - Ports 596
 - RS232 131
 - UART 132
- Serienwiderstand 299
- Servo 264
 - Anschlusskabel 265
 - durchlaufend 286
 - Roboter 274
 - Umbau durchlaufender Servo 274
- Servoachse 269
- Servo-Bibliothek 266
- Servomotor 274
- setup() 85
- Shield 400
 - Eagle-Format 401
 - Motor Shield 405
 - SD Card Shield 406
- Shift-Register 313
- Signalausgabe
 - analog 365
 - Signalgeber 365
 - Signalglättung 73
 - Signalkopplung 73
 - sin 588
 - Sinus 129
 - Sinusgenerator 372
 - Sinussignal 129, 372
 - Sippino 503
 - Sketch 19, 81
 - Grundstruktur 567
 - Sketchbook 36
 - Sketch-Upload 41
 - Skizzenbuch 40
 - Slave
 - I2C-Bus 140
 - SMD 77, 78
 - SN754410 276
 - SO-8 147
 - SoftSerial 394
 - Software 27
 - Softwarebibliothek 391
 - SoftwareSerial 138
 - Softwareteil 18
 - Solid State Relay 263
 - Sollwert 265
 - Regelsystem 366
 - Sollwertgeber 390
 - Source 284
 - Spalte 335
 - Spannung 66
 - Spannungsregler 301
 - LM317 301
 - Spannungsreglern 64
 - Spannungsteiler 112
 - Speicherbedarf
 - String 576
 - Speicherplatz 354
 - Sperrrichtung 74
 - SPI 99, 406
 - Spielkonsole
 - Wii 231
 - SpikenzieLabs 503
 - SPI-Kommunikation
 - Ports 54, 56
 - Spule 79
 - sq 588
 - sqrt 588
 - SRAM
 - Arduino-Boards 601
 - SSD1306 330
 - SSR 263
 - Standardbauelement 70
 - Standardbibliothek 392
 - Standardmodul
 - KiCad 558
 - Standortinformationen 351

Statuscode 420
 Steckanschluss 58
 Steckbrett 58, 539
 Steckbrett-Adapter
 Spannungsversorgung siehe Steckbrett
 Steckdose 49
 Steckerleiste 22
 Steckernetzteil 22
 Steinhart-Hart-Gleichung 193
 Stellglied 261
 Stepper 398
 Steuerelektronik
 Servo 265
 String 575
 Array 576
 String-Größe 576
 string 83
 Strom 66
 Stromfluss
 begrenzen 299
 Stromkreis 66
 Stromlaufplan 50
 Stromversorgung 57
 Stromverstärkungsfaktor 76
 Struktur 85
 Subtraktion 589
 Surface mounted device siehe SMD
 switch 584
 Syntaxerkennung 40

T

Taktfrequenz 601
 tan 588
 Taster 80, 107
 TCP 414
 Temperaturschwellenschalter 366
 Temperatursensor 211
 DS1820 204
 LM35 197
 NTC 192
 Testaufbau 540
 Testen 86
 Testprogramm 34, 495
 TextFinder 397
 Tiefpass 108
 Timer 128
 TinyGPS 396
 TIP110 275
 TMP275 213
 TMP36 199
 Tonausgabe 129, 593
 Tool 539
 Transceiver siehe RFM12B
 Transistor 76
 bipolar 282
 Transmission Control Protocol siehe TCP

Troubleshooting 493
 true 574
 Two-Wire 139
 Typen 81
 Typenbezeichnung
 Bauelemente 71

U

UART 132
 UBE 283
 Überspannung
 Schutz 261
 Übertragungsgeschwindigkeit
 serielle Kommunikation 131
 serielle Schnittstelle 596
 Uhrenbaustein 148
 ULP
 Eagle 557
 Ultraschallsensor 230
 Umrechnungsformel
 Fahrenheit 123
 Umwandlerschaltung 365
 Umweltdaten 462
 Universalmessgerät 544
 UNO R3
 Board 601
 UNO R4 WIFI 427
 UNO siehe Arduino UNO
 Unsigned
 Datentyp 573
 USB-Adapter 48
 USB-Device 46
 USB-Kabel 31
 USB-Port 19
 USB-Schnittstelle
 Stromverbrauch 19
 USB-Serial-Breakout-Board 505
 USB-Seriell-Wandler 498
 USB-Stecker
 Typ A 31
 Typ B 31
 USBTinyISP 508
 USB-Treiber 29
 Linux 30
 macOS 30
 User Language Program 557

V

Variable 81, 580
 Name 580
 Variablendeklaration 81
 Setup() 567
 Ventil 261
 Verbindungskabel 31
 Vergleichsoperation 589
 VG400 218

Vin 53, 55
VirtualColorMixer
 Fritzing 554
Virtuelle COM-Schnittstelle 33
void 84
 Typ 568
Vollumdrehung 274
Vorgehen
 Fehlersuche 493
Vorwiderstand 283

W

Wasserstandsmesser 366
Wasserwaage 232
Wave Shield 406
Websites 12
Wechselspannung 263
Wechselspannungslast 282
Weichlöten 542
Wert
 Konstanten 581
Wertebereich 82
while 584
Widerstand 66, 71
Widerstandsleiter 366
Widerstandsleiterschaltung 371
Wifi 365, 429, 434
 Webclient 429
 Webserver 434
WifiS3 414
Wifi-Schnittstelle 427
Wii
 Anschlusschema 288
 I2C-Bus 288

Nunchuk 189, 231, 286
 Remote 231
Wii Nunchuk 399
Wii-Nunchuk-Adapter 410
Windlicht 533
Windows 28
Windows 10/11 29
Wire-Bibliothek 140, 392
 wire.begin() 392
 wire.send() 393
Würfel 180
Würfelschaltung 181

X

X-Koordinate 546

Y

Y-Koordinate 546

Z

Zange
 Seitenschneider 543
Zeiger
 Pointer 576
Zeile 335
Zeilennummer 494
Zeitfunktion 595
Zufallszahl
 PRNG 588
 Pseudorandom Number Generator 588
Zusatzboard 496
Zuweisung
 gemischt 589
 zusammengesetzt 590