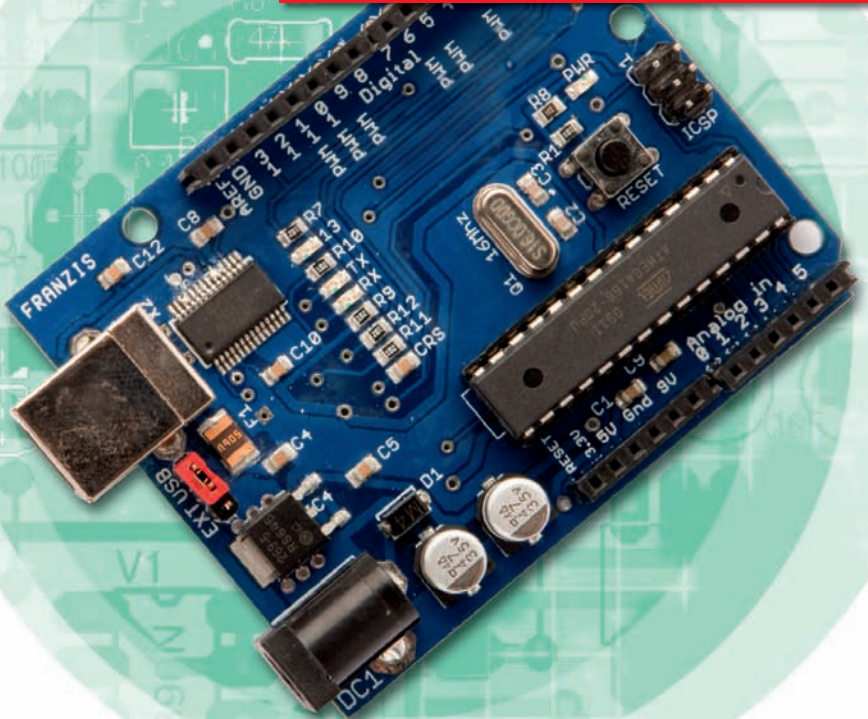


Ulli Sommer

- Über 80 praktische Experimente
- Grundlagenkurs zur Programmierung



# Arduino

Mikrocontroller-Programmierung mit Arduino/Freduino

## Auf CD-ROM:

- Open-Source-Soft- und Hardware
- Über 80 Quellcodes zu den Experimenten
- Schaltpläne und Datenblätter
- Open-Source-VB.NET-Programme zum Messen und Steuern



# Vorwort

Vielen fällt der Einstieg in die Mikrocontroller-Programmierung und die dazugehörige Elektronik schwer. Bei den meisten Mikrocontroller-Systemen muss man sich zuvor durch unzählige und für den Anfänger schwer verständliche Datenblätter wälzen. Die Programmieroberflächen sind meist viel zu kompliziert und mehr für professionelle Programmierer ausgelegt. Somit bleibt manchem der Zugang in die Welt der Mikrocontroller für immer verwehrt.

*Arduino* ist eine leicht zu verstehende Open-Source-Plattform, basierend auf einem Mikrocontrollerboard und einer Entwicklungsumgebung mit einer API (Programmier-Schnittstelle) für den Mikrocontroller. Für die Interaktion zwischen Mensch und Mikrocontroller können diverse analoge und digitale Sensoren angeschlossen werden, die die Umwelt erfassen und die Daten an den Mikrocontroller weitergeben. Der Mikrocontroller verarbeitet die eingehenden Daten, und durch das Programm entstehen neue Ausgabedaten in analoger oder digitaler Form. Hierbei sind der Kreativität des Entwicklers fast keine Grenzen gesetzt.

Die Arduino-Programmieroberfläche unterstützt den Entwickler bei seinen Vorhaben durch ihre vorgefertigten Programme und Funktionsbibliotheken. Das einfache Zusammenspiel aus Hard- und Software bildet die Basis für *Physical Computing*: die Verbindung der realen Welt mit der des Mikrocontrollers, die aus Bits und Bytes besteht. Dieses Buch zeigt Ihnen Schritt für Schritt, wie Sie den leichten Einstieg in diese Welt finden.

Viel Spaß beim Lesen und Experimentieren mit diesem Buch!

Ulli Sommer

# CD-ROM zum Buch

Diesem Buch liegt eine CD-ROM bei, die verschiedene Programme, Tools, Datenblätter und Beispiele enthält. Die CD-ROM erleichtert Ihnen das Arbeiten mit diesem Buch. Die hier abgedruckten Beispiele sind auf der CD-ROM enthalten.

## Inhalt der CD-ROM

- Arduino-Entwicklungsumgebung (IDE)
- Beispiel-Programmcode zum Lehrgang
- Diverse Tools
- Datenblätter
- Schaltpläne

## GPL (General Public License)

Sie können Ihre eigenen Programme mit anderen Anwendern über das Internet austauschen. Die Beispielprogramme stehen unter der Open-Source-Lizenz *GPL* (General Public License). Daher sind Sie berechtigt, die Programme unter den Bedingungen der GPL zu modifizieren, zu veröffentlichen und anderen Anwendern zur Verfügung zu stellen, sofern Sie Ihre Programme dann ebenfalls unter die GPL-Lizenz stellen.

## Systemvoraussetzung

Ab Pentium III-PC, Windows 98SE/ME/XP/Vista/Windows 7, Linux, Mac OS, CD-ROM-Laufwerk, Java

## Updates und Support

Arduino wird ständig weiterentwickelt. Updates können kostenlos von der Website [www.arduino.cc](http://www.arduino.cc) heruntergeladen werden (es fallen nur Ihre üblichen Online-Kosten an).

## Vorbereitungen

Die vorgestellten Experimente können mit wenigen, meist preiswerten Teilen – aus der Bastelkiste oder extra gekauft – durchgeführt werden. Im Anhang finden Sie eine Liste der Teile und Lieferrachweise für den Bezug der Komponenten.

Für die Experimente und Versuche brauchen Sie weder Batterien noch eine zusätzliche Stromversorgung.

Als sinnvolle und hilfreiche Ergänzung kann ein Vielfachmessinstrument (Multimeter) und/oder eine Schnittstelle zum Computer zur Strom- und Spannungsmessung verwendet werden. Damit können zusätzliche Experimente durchgeführt werden und es sind weitere spannende Zusammenhänge erfahrbar. Außerdem ist es nützlich, eine handelsübliche Akkuzelle der Größe AA (Mignon) oder AAA (Micro) für einige Experimente der Ladetechnik zur Verfügung zu haben.

Das Buch vermittelt die wichtigsten Grundlagen der Mikrocontrollertechnik. Außerdem werden beispielhafte praktische Anwendungen vorgestellt, mit deren Hilfe es möglich wird, eigene Schaltungen und Erfindungen rund um die Mikrocontrollertechnik zu entwickeln.

Sie können Ihr Equipment auch um eine Sortimentsbox ergänzen. Darin werden alle Einzelteile griffbereit und übersichtlich aufbewahrt.

# Inhaltsverzeichnis

<b>1</b>	<b>Mikrocontroller-Grundlagen.....</b>	<b>13</b>
1.1	Aufbau und Funktionsweise .....	14
1.1.1	Die CPU.....	14
1.1.2	Arbeits- und Programmspeicher .....	15
1.2	Peripherie .....	16
1.3	Technologievergleich: RISC und CISC .....	16
1.3.1	CISC-Technologie .....	17
1.3.2	RISC-Technologie .....	17
1.3.3	Vergleich.....	17
<b>2</b>	<b>Programmierung der Mikrocontroller .....</b>	<b>19</b>
2.1	Was ist ein Programm? .....	19
2.2	Programmierung in C.....	19
<b>3</b>	<b>Eine kleine Übersicht über die ARDUINO-Mikrocontroller-Familie.....</b>	<b>21</b>
3.1	Arduino Mega .....	22
3.2	Arduino Duemilanove.....	23
3.3	Arduino Mini .....	24
3.4	Arduino Nano.....	25
3.5	Arduino Pro Mini .....	26
3.6	Arduino Pro.....	27
3.7	LilyPad .....	28
3.8	USB-Adapter .....	29
<b>4</b>	<b>Arduino Shields .....</b>	<b>31</b>
4.1	Arduino ProtoShield .....	31
4.2	Ardumoto.....	32
4.3	TellyMate .....	33
4.4	ArduPilot.....	34
4.5	Ethernet Shield .....	36
<b>5</b>	<b>Bauteile.....</b>	<b>37</b>
5.1	Teilleiste Basisexperimente .....	37
5.2	Teilleiste Zusatzexperimente (I <sup>2</sup> C, LCD ...) .....	38
5.3	Das Freeduino-Experimentierboard .....	38
5.4	Anschlüsse und LEDs des Freeduino-Mikrocontroller- Experimentierboards.....	39
5.5	Die Stromversorgung.....	40

5.6	Reset-Taster .....	40
5.7	ISP-Anschluss .....	40
5.8	Sicherheitshinweise.....	41
<b>6</b>	<b>Bauteile und ihre Funktion .....</b>	<b>43</b>
6.1	Leuchtdioden.....	43
6.2	Widerstände .....	43
6.3	Kondensatoren .....	45
6.4	Transistoren.....	47
6.5	Diode .....	47
6.6	Piezo-Schallwandler (Buzzer).....	47
6.7	Schaltdraht.....	48
6.8	Taster.....	48
6.9	Potenziometer.....	49
6.10	LDR .....	49
6.11	Steckbrett .....	50
<b>7</b>	<b>Die ersten Vorbereitungen (Inbetriebnahme) .....</b>	<b>51</b>
7.1	Treiberinstallation.....	51
7.2	Das Tool MProg für den FT232RL .....	52
7.3	FT232R mit MProg programmieren .....	57
7.4	Die Arduino-Software installieren.....	58
<b>8</b>	<b>Die Arduino-Entwicklungsumgebung .....</b>	<b>61</b>
8.1	Einstellungen in der Arduino-IDE.....	63
8.2	Der erste Funktionstest »ES_Blinkt« .....	64
8.3	Was haben wir getan? .....	67
<b>9</b>	<b>Arduino-Programmiergrundlagen.....</b>	<b>69</b>
9.1	Bits und Bytes.....	69
9.2	Grundsätzlicher Aufbau eines Programms.....	70
9.2.1	Sequenzieller Programmablauf .....	70
9.2.2	Interruptgesteuerter Programmablauf .....	71
9.3	Der Aufbau eines Arduino-Programms.....	72
9.4	Das erste eigene Programm mit Arduino.....	72
9.5	Arduino-Befehle und ihre Verwendung.....	74
9.5.1	Kommentare im Quelltext.....	74
<b>10</b>	<b>Weitere Experimente mit Arduino.....</b>	<b>133</b>
10.1	Der Transistor-LED-Dimmer .....	133
10.2	Softer Blinker .....	135
10.3	Taster entprellen.....	138
10.4	Einschaltverzögerung.....	143

10.5	Ausschaltverzögerung.....	144
10.6	LEDs und Arduino.....	145
10.7	Größere Verbraucher schalten.....	148
10.8	DAC mit PWM-Ports.....	151
10.9	Mit Musik geht alles besser.....	156
10.10	Romantisches Mikrocontroller-Kerzenlicht.....	159
10.11	Überwachung des Personalausgangs.....	161
10.12	RTC (Real Time Clock).....	163
10.13	Schuluhrprogramm.....	165
10.14	Lüftersteuerung.....	169
10.15	Dämmerungsschalter.....	172
10.16	Alarmanlage.....	174
10.17	Codeschloss.....	177
10.18	Kapazitätsmesser mit Autorange.....	181
10.19	Potenziometer professionell auslesen.....	184
10.20	Sensortaster.....	186
10.21	State Machine.....	188
10.22	Ein 6-Kanal-Voltmeter mit Arduino.....	191
10.23	Spannungs-Plotter selbst programmiert.....	193
10.24	Das Arduino-Speicheroszilloskop.....	196
10.25	StampPlot, der Profi-Datenlogger zum Nulltarif.....	198
10.26	Steuern über VB.NET.....	202
10.27	Temperaturschalter.....	205
<b>11</b>	<b>Der I<sup>2</sup>C-Bus.....</b>	<b>209</b>
11.1	Bit-Übertragung.....	210
11.2	Startbedingung.....	210
11.3	Stoppbedingung.....	210
11.4	Byte-Übertragung.....	210
11.5	Bestätigung (Acknowledgment).....	211
11.6	Adressierung.....	211
11.7	7-Bit-Adressierung.....	211
<b>12</b>	<b>Arduino und der I<sup>2</sup>C-Bus-Temperatursensor LM75.....</b>	<b>213</b>
<b>13</b>	<b>I<sup>2</sup>C-Portexpander mit PCF8574.....</b>	<b>217</b>
<b>14</b>	<b>Ultraschallsensoren zur Entfernungsbestimmung.....</b>	<b>221</b>
14.1	Der SRF02-Ultraschallsensor.....	221
14.2	Auslesen der Entfernungsdaten.....	222

<b>15</b>	<b>Arduino mit GPS .....</b>	<b>225</b>
15.1	Wie viel Satelliten sind notwendig?.....	226
15.2	Wie schlieÙe ich das GPS an Arduino an?.....	227
15.3	GPS-Protokoll.....	228
<b>16</b>	<b>Stellantrieb mit Servo für Arduino .....</b>	<b>233</b>
16.1	Wie funktioniert ein Servo? .....	233
16.2	Anschluss an Arduino .....	234
<b>17</b>	<b>LC-Displays <i>LCDs</i> .....</b>	<b>237</b>
17.1	Polarisation von Displays.....	238
17.2	Statische Ansteuerung, Multiplexbetrieb .....	238
17.3	Blickwinkel 6 Uhr/12 Uhr .....	238
17.4	Reflektiv, Transfektiv, Transmissiv.....	239
17.5	Die Kontrasteinstellung des Displays .....	240
17.6	Der Zeichensatz .....	242
17.7	Pinbelegung der gängigen LCDs .....	243
17.8	So wird das Display vom Mikrocontroller angesteuert .....	244
17.9	Initialisierung der Displays.....	245
17.10	Das Display und sein Anschluss am Arduino .....	246
17.11	Die erste Ausgabe .....	248
17.12	Was haben wir genau gemacht? .....	251
<b>A</b>	<b>Anhang .....</b>	<b>253</b>
A.1	Arduino zu ATmega Pinmap .....	253
A.2	Escape-Sequenzen .....	253
A.3	ASCII-Tabelle.....	255
	<b>Bezugsquellen.....</b>	<b>259</b>
	<b>Stichwortverzeichnis .....</b>	<b>261</b>



## 2 Programmierung der Mikrocontroller

Mit der zunehmenden Integration von Halbleiterbauteilen wie Mikroprozessoren hielten Mikrocontroller immer stärker Einzug in die Anwendungsgebiete der Mess-, Steuer- und Regelungstechnik. Aber auch im Hobbybereich wurden die Mikrocontroller immer beliebter. Das liegt zum einen daran, dass heute komplexe, meist analoge Schaltungen durch einfachere digitale Mikrocontroller-Schaltungen ersetzt werden. Ein anderer ausschlaggebender Punkt ist das unschlagbare Preis-Leistungs-Verhältnis von Mikrocontrollern.

### 2.1 Was ist ein Programm?

Ein Programm ist die Beschreibung eines Informationsverarbeitungsprozesses. Im Lauf eines solchen Prozesses wird aus einer Menge von variablen oder konstanten Eingangswerten eine Menge von Ausgangswerten berechnet. Die Ausgangswerte sind entweder selbst Ziel der Informationsgewinnung oder dienen mittelbar zur Reaktion auf die Eingangswerte. Neben den eigentlichen Berechnungen kann ein Programm Anweisungen zum Zugriff auf die Hardware des Computers oder zur Steuerung des Programmflusses enthalten. Ein Programm besteht aus mehreren Zeilen sogenannten Quelltextes. Dabei enthält jede Zeile eine oder mehrere Rechen- oder Steueranweisungen. Neben diesen Anweisungen selbst bestimmt ihre Reihenfolge wesentlich die eingangs beschriebene Informationsverarbeitung. Die Ausführung der den Anweisungen entsprechenden Operationen durch den Steuercomputer erfolgt sequenziell, also nacheinander. Eine Folge von Programmanweisungen mit einem bestimmten Ziel nennt man auch *Algorithmus*.

### 2.2 Programmierung in C

C oder auch *ANSI-C* ist eine einfach zu erlernende Programmiersprache. C ist eine imperative Programmiersprache, die der Informatiker Dennis Ritchie in den frühen 70er-Jahren an den Bell Laboratories für das Betriebssystem Unix entwickelte. Seitdem ist sie weltweit stark verbreitet. Die Anwendungsbereiche von C sind sehr verschieden. Es wird z. B. zur System- und Anwendungsprogrammierung eingesetzt. Die grundlegenden Programme aller Unix-Systeme und die System-

kerne vieler Betriebssysteme sind in C programmiert. Zahlreiche Sprachen wie C++, Objective-C, C#, Java, PHP oder Perl orientieren sich an der Syntax und anderen Eigenschaften von C. Es ist also mehr als lohnenswert, sich mit dieser Programmiersprache zu beschäftigen, da man später auch leicht auf andere Mikrocontrollersysteme umsteigen kann. Für fast alle Mikrocontroller existiert ein freier C-Compiler, den die Hersteller zum Download anbieten. Das C von Arduino ist jedoch um einiges einfacher gehalten als die professionellen C-Compiler und nimmt sehr viel Arbeit ab. Vor allem um die komplizierten Hardware-Routinen muss man sich bei Arduino nicht kümmern, da sie bereits als feste Befehle in der Entwicklungsumgebung integriert sind.

## 3 Eine kleine Übersicht über die ARDUINO-Mikrocontroller-Familie

Die Arduino-Hardware verwendet ausschließlich gängige, allgemein verfügbare Bauteile. Daher ist es leicht, die Funktionsweise zu verstehen und die Schaltung an eigene Wünsche anzupassen oder Erweiterungen vorzunehmen. Den Kern bildet ein ATmega-Controller aus Atmels weitverbreiteter 8-Bit-AVR-Familie. Hinzu kommen Schaltungsteile zur Stromversorgung und eine serielle Schnittstelle. Letztere ist bei den jüngeren Arduino-Versionen als USB-Interface ausgelegt. Über diesen Anschluss erfolgt der Download der Anwenderprogramme und bei Bedarf auch die Kommunikation zwischen PC und Arduino während der Programmausführung.

Weil Arduino-Boards so einfach und universell ausgelegt sind, werden sie häufig auch schlicht als *I/O-Board* bezeichnet. Arduino stellt dem Anwender 14 digitale Ein- oder Ausgänge zur Verfügung, davon sind sechs als Analogausgang (8 Bit PWM) zu verwenden. Weitere sechs Eingänge können analoge Signale erfassen (10 Bit ADC). Bei Bedarf stehen SPI und I<sup>2</sup>C als weitere Schnittstellen zur (seriellen) Kommunikation zur Verfügung.

Es gibt Arduino-Boards in mehreren Varianten. Die Originale stammen vom Hersteller Smart Projects aus Italien. Es gibt mittlerweile auch zahllose Klone und Nachbauten von anderen Anbietern, schließlich handelt es sich um *Open Hardware*. Ein wichtiger Unterstützer des Arduino-Projekts ist Sparkfun aus Boulder, Colorado. Die Kooperation mit dem US-Partner hat eine Reihe optimierter Arduino-Boards hervorgebracht, die den Zusatz »Pro« im Namen führen. Außerdem ist mit LilyPad ein wichtiger Ableger entstanden, der das Thema *Wearable Computing* aufgreift.

Die meisten Anwender setzen auf das von Smart Projects gefertigte, handteller-große Arduino Duemilanove (Duemilanove = 2009), das den ATmega-Controller in DIP-Bauform auf einem Sockel trägt. Es unterscheidet sich nur unwesentlich vom überaus erfolgreichen Vorgänger Arduino Diecimilanove, dessen Namensgebung auf die ersten 10.000 verkauften Boards zurückgeht. Auf den Boards ist ein FTDI-Chip aufgelötet, der die USB-Schnittstelle bereitstellt.

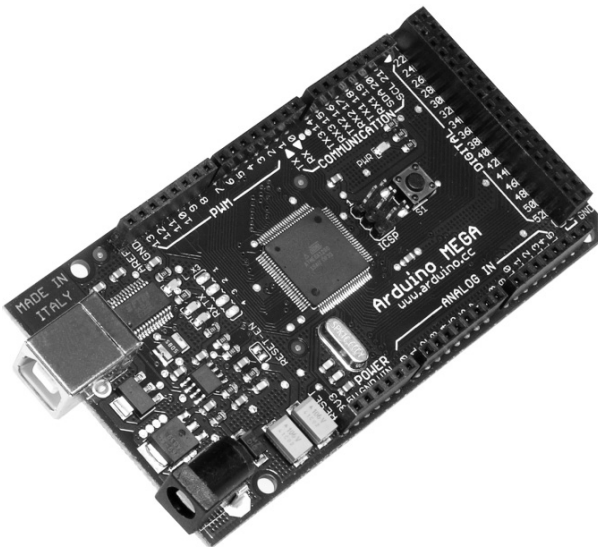
Das neue Arduino Mega Board verwendet einen leistungsstärkeren Mikrocontroller (Atmega1280) und bietet mehr Speicher, I/O-Pins und Funktionen auf einer deutlich erweiterten Platinenfläche.

Wesentlich kleiner ist Arduino Mini, ein Board im DIP24-Format. Das ganze Modul lässt sich auf einen 24-poligen DIL-Sockel stecken. Die Version Arduino Pro Mini von Sparkfun ist nahezu identisch, wird aber ohne »Beinchen« (seitliche Stifte) geliefert. Diese Module benötigen zum Programmieren einen USB-Adapter, der an der Schmalseite der Module angesteckt werden kann.

Das LilyPad-Board von Leah Buechley (in Zusammenarbeit mit Sparkfun) ist auch Arduino-kompatibel und verfolgt einen ganz eigenen Zweck. LilyPad und Zubehör sind dafür ausgelegt, in Kleidung eingenäht zu werden, um dort eine möglichst enge Symbiose von Technik und Künstler zu realisieren. Die charakteristische runde Form des LilyPad-Arduinos erregt ebenso Aufmerksamkeit wie die Farbgebung und die kreisförmige Anordnung der Kontakte. Zum Einsatz kommt hier die Low-Power-Version (3,3 V) des ATmega168. Zahlreiche kleine Peripherieplatinen (Sensoren, LEDs, Taster ...) ergänzen LilyPad zu einem ganzen System unter dem Motto »Elektronik mit der Nähmaschine« .

Über weitere Board-Versionen und Zubehörteile informieren Sie die Arduino-Projektseite (siehe Links) und die Produktseiten von SparkFun Electronics.

## 3.1 Arduino Mega



**Bild 3.1:** Arduino Mega  
(Quelle: Fa. Elmicro)

**Technische Daten:**

- ATmega1280 Mikrocontroller
- 128 KB Flash
- 8 KB RAM, 4 KB EEPROM
- 16-MHz-Takt
- 54 digitale I/O-Pins, davon 14 als PWM nutzbar
- 4 Hardware-UARTs
- I<sup>2</sup>C-Interface, SPI
- 16 analoge Eingänge (10 Bit)
- USB-Interface, Spannungsversorgung, Bootloader etc. wie beim Arduino Duemilanove
- Abmessungen ca. 101 mm x 53 mm x 12 mm

## 3.2 Arduino Duemilanove



**Bild 3.2:** Arduino Duemilanove  
(Quelle: Elmicro)

**Technische Daten:**

- ATmega328 Mikrocontroller
- 32 KB Flash (davon 2KB für Bootloader)
- 2 KB RAM, 1 KB EEPROM
- 16-MHz-Takt
- 14 digitale I/O-Pins, davon 6 als PWM nutzbar
- sechs analoge Eingänge (10 Bit)
- On-Board-USB-Schnittstelle mit FT232RL von FTDI

- 5 V Betriebsspannung, Speisung über USB oder über Spannungsregler (7 V bis 12 V Eingangsspannung)
- Abmessungen ca. 69 mm x 53 mm x 12 mm
- Bootloader im Lieferzustand bereits installiert, Download ohne Programmieradapter möglich

### 3.3 Arduino Mini



Bild 3.3: Arduino Mini (Quelle: Elmicro)

#### Technische Daten:

- ATmega168 Mikrocontroller mit 16-MHz-Quartztakt
- Programmierung über USB-Adapter (ARDUINO/USB, USB-Adapter mit FTDI-Chip)
- 512 Byte EEPROM
- 1 KB SRAM
- 16 KB FLASH (2 KB benötigt der Bootloader für sich)
- Betriebsspannung 5 V
- 14 Digitale I/Os, sechs davon können zur PWM-Erzeugung genutzt werden
- acht analoge 10-Bit-Eingänge
- Versorgungsspannung 7 V bis 9 V

## 3.4 Arduino Nano



Bild 3.4: Arduino Nano (Quelle: Elmicro)

### Technische Daten:

- ATmega328 oder ältere Version 168 mit 16-MHz-Quarztakt
- Programmierung über USB-»On Board Chip«
- Autoreset-Funktion
- 5-V-Technik
- 14 Digitale I/Os, sechs davon können zur PWM-Erzeugung genutzt werden
- acht analoge 10-Bit-Eingänge
- 32 KB oder 16 KB FLASH
- 1 KB SRAM
- 512 oder 1 KByte EEPROM
- Ausgangsstrom pro I/O max. 40 mA
- Versorgungsspannung 6 V bis 20 V
- Abmessungen: 18 mm x 43 mm

## 3.5 Arduino Pro Mini



Bild 3.5: Arduino Pro Mini (Quelle: Elmicro)

### Technische Daten:

- ATmega328 mit 16-MHz-Quarztakt (Genauigkeit 0,5 %)
- Programmierung über USB-Adapter (ARDUINO/USB)
- Autoreset-Funktion
- Diese Version gibt es in 5-V- und 3,3-V-Technik
- Ausgangsstrom max. 150 mA
- Überlastschutz
- Verpolungsschutz
- Versorgungsspannung 5 V bis 12 V
- Power und Status LED bereits »On Board«
- Abmessungen: 18 mm x 33 mm
- Gewicht weniger als 2 g



## 3.6 Arduino Pro



**Bild 3.6:** Arduino Pro (Quelle: Elmicro)

### Technische Daten:

- ATmega328 und ältere ATmega168 mit 16-MHz-Quarztakt
- Programmierung über USB-Adapter (ARDUINO/USB)
- Diese Version gibt es in 5-V- und 3,3-V-Technik
- 14 Digital-I/O-Pins (sechs davon als PWM nutzbar)
- sechs analoge 10-Bit-Eingänge
- Versorgungsspannung 3,35 V bis 12 V (3,3-V-Version)
- Versorgungsspannung 5 V bis 12 V (5-V-Version)
- Ausgangsstrom pro Digitalport 40 mA
- 32 KB oder 16 KB (ATmega168) FLASH
- 1 KB (ATmega168) oder 2 KB (ATmega328) SRAM
- 512- (ATmega168) oder 1-KB-EEPROM

## 3.7 LilyPad

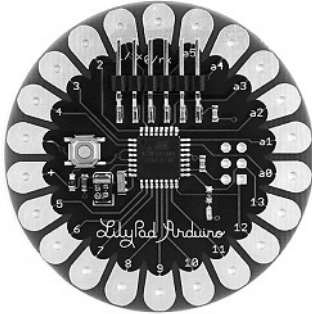


Bild 3.7: LilyPad Arduino (Quelle: Elmicro)

### Technische Daten:

- ATmega328V und ältere ATmega168V mit 16-MHz-Quarztakt
- Programmierung über USB Adapter (ARDUINO/USB)
- Spannungsversorgung 2,7 V bis 5,5 V
- 14 Digital-I/O-Pins (sechs davon als PWM nutzbar)
- sechs analoge 10-Bit-Eingänge
- Ausgangsstrom pro Digitalport 40 mA
- 32 KB oder 16 KB (ATmega168) FLASH
- 1 KB (ATmega168) oder 2 KB (ATmega328) SRAM
- 512- (ATmega168) oder 1-KB-EEPROM

## 3.8 USB-Adapter



**Bild 3.8:** USB-Adapter mit FTDI-Chip (Quelle: Elmicro)

Diesen Programmieradapter gibt es in 3,3-V- und in 5-V-Ausführung.

Der Adapter wird zum Programmieren der Arduino-Borads ohne USB-Anschluss benötigt. Die Pinbelegung entspricht den Original-Arduino-Spezifikationen. Er kann auch zur Kommunikation (virtuelle serielle Schnittstelle) verwendet werden. Dieses Feature muss man für eigene Entwicklungen einfach haben. Es ermöglicht, einen Sketch auf das Board zu laden, ohne die Reset-Taste zu drücken.

# Stichwortverzeichnis

## Symbole

#Define-Anweisungen 81  
 10-k $\Omega$ -Potenziometer 246  
 4-Bit-Mode 245

## A

abs(x) 95  
 Abstrakte Maschine 188  
 ADC 16, 119, 184, 191, 193, 205  
 ADC-Ausgabe 252  
 ADC-Kanäle 192  
 Adresse 209  
 Akkus 234  
 Alarm 174  
 Alarmanlage 174  
 Ampel 188  
 analog 151  
 Analogeingang 186, 193, 251  
 analogRead 120  
 analogRead() 128, 184  
 analogWrite 124  
 Analysator 237  
 Anode 43  
 ANSI-C 19  
 Ansteuerung 244  
 Arbeitsspeicher 14, 15  
 Arduino Diecimila 63  
 Arduino-Duemilanove-Board 38  
 Ardumoto 32  
 ArduPilot 34  
 Arithmetik 80

Arrays 78  
 dynamisches 79  
 ASCII-Tabelle 255  
 ASCII-Zeichen 105, 242  
 Atmega1280 22  
 Auflösung 120  
 Ausgang 113

## B

Basis B 47  
 Batterien 8  
 Baudrate 102  
 BIN 105  
 binary digits 69  
 Blickwinkel 238  
 Boolean 76  
 Breadboard 38, 50  
 Byte 76, 105  
 Byte() 93

## C

C 19  
 CD-ROM 7  
 Char 76  
 Char() 93  
 Checksumme 192  
 CISC 17  
 CISC-Technologie 16  
 Codeschloss 177  
 Comport 63  
 Comport-Nummer 57  
 COM-Schnittstelle 58  
 constrain(x, a, b) 95  
 Continue 92  
 cos(rad) 99

C-Programmierung 74  
 CPU 14  
 CRC 192

## D

DAC 16  
 Dämmerungsschalter 172  
 Daten drahtlos übermitteln 35  
 Datenbits 108  
 Datentransfer 244  
 Datentypen 75, 78  
 Datenverarbeitung 13  
 DA-Wandler 151  
 DEC 104  
 Decrement 80  
 delay 67, 127  
 delay() 130, 168, 251  
 Digitalport 133  
 digitalRead 114, 124, 142  
 digitalWrite 67, 114, 124  
 Diode 47  
 Display 244, 246  
 Display() 251  
 Displaycontroller 242  
 do while 88  
 DOP-Wert 226  
 Dot-Matrix 237  
 Dot-Matrix-Displays 240  
 do-while-Schleife 141  
 Download 66  
 Drift 184  
 Durchlassrichtung 145

**E**

E12-Reihe 146  
 E24-Reihe 44  
 Eingang 113  
 Einschaltverzögerung 143  
 Einstellungen 64  
 Elektrolytkondensator 46  
 else if 83  
 Emitteranschluss 47  
 Entfernungsmesser 221  
 Entstörung 149  
 Entwicklungsumgebung  
 (IDE) 61  
 Escape-Sequenzen 253  
 Ethernet Shield 36  
 EVA 70  
 EXT 40

**F**

Farad 45  
 Farbcode 44  
 Flash-Speicher 15  
 Float 77  
 Float() 93  
 for 86, 88  
 Freilaufdiode 151  
 Frequenz 156  
 FSTN-Technik 238  
 FT232R 58  
 FT232RL 51, 57  
 FTDI-Treiber 52  
 Funkstrecke 35  
 Funktionen 91  
 Mathematische 93  
 Funktionsdefinition 76

**G**

Gerätemanager 63  
 Geschweifte Klammer 75  
 Getriebemotor 233  
 Global Positioning System  
 (GPS) 225  
 GPL 7

GPS-Protokoll 228  
 GPS-Signale 226  
 Grad Celsius 110  
 Grad Fahrenheit 110  
 Graphen 198  
 Grenzfrequenz 152  
 Grundlagen 69

**H**

Halbbyte 69  
 Halbleitermaterial 49  
 HAL-Prinzip 68  
 Hardware 21  
 UART 102  
 HD44780 240  
 HD44780-/KS0066-  
 Standard 248  
 H-DOP 226  
 HEX 104  
 high 67  
 Highbyte 192

**I**

I/O-Board 21  
 I<sup>2</sup>C 213  
 I<sup>2</sup>C-Bus 209  
 IDE (siehe  
 Entwicklungsumgebung)  
 61  
 if 83  
 IF 145  
 if – else 82  
 Induktionsspannung 151  
 Informationsverarbeitungs-  
 prozesses 19  
 Initialisierung 245  
 Inkrement 80  
 Input-Konfiguration 113  
 Int() 93  
 Integer 77  
 Interruptus 71  
 ISP-Anschluss 40

**K**

Kapazität 45  
 Kapazitätsmesser 181  
 Kathode 43  
 Keramikkondensator 46  
 Kleinsignaltransistor 47,  
 148  
 Kohleschichtwiderstand 43  
 Kollektor 47  
 Kommunikation 100  
 Kompilieren 62  
 Kondensator 45, 181  
 Konstante 81  
 Kontrast 240, 248  
 Kontrasteinstellung 240  
 Kontrollstrukturen 81

**L**

Lautsprecher 47  
 LCD 241, 248  
 Pinbelegung 243  
 lcd.print() 251  
 lcd.setCursor() 251  
 LCD-Modul 240, 245, 246  
 LDR 49, 169, 172, 174  
 LED 145, 148, 149  
 LED-Doppelblitzer 146  
 Leuchtdiode 43  
 Library 251  
 Lichtempfindlichkeit 171  
 LM75 213  
 Lokale 76  
 Long 77  
 Long() 93  
 loop() 251  
 Löschbefehle 255  
 Lowbyte 192  
 Lüftersteuerung 169

**M**

map(x, fromLow,  
 fromHigh, toLow, toHigh)  
 96

- max(x, y) 94
- MAX232 227
- Melodien 157
- Menü 62
- Messgeräte 181
- micros() 127, 131, 168
- Mikrocontrollerboard 63
- Mikrosekunden 131
- millis() 128, 130, 251
- min(x, y) 93
- Modellbauservo 233
- Modellflugzeug 34
- Modularität 90
- MProg 52, 53, 56
- Multiplexbetrieb 238
  
- N**
- Neu 62
- Nibble 69
- noDisplay() 251
- Not(!)-Funktion 135
- NTC 241
  
- O**
- OCT 105
- Öffnen 62
- Operator 80, 82
- Oszilloskop 197
- Output-Konfiguration 113
  
- P**
- Parameter 107
- Parity Bit 108
- PCF8574 217
- Peripherie 16
- Pfeiltasten 254
- Philips 209
- Physikalische Größen 13
- Piezo-Schallwandler 47, 126, 156, 165
- Pinmapping 253
- pinMode 67, 113
- Polarisation 238
- Polarisator 237
- Port 148
- Portexpander 217
- Potenzialfreier Kontakt 151
- Potenziometer 49, 122, 161, 171, 184, 198, 233, 240
- pow(base, exponent) 97
- Power ON-LED 39
- Programm übertragen 62
- Programmierungsumgebung 58
- Programmierung 57
  - prozedurale 70
  - sequenzielle 70
- Programmspeicher 14, 15
- ProtoShield 32
- Prozedur 70
- Puffer 103, 109
- Pull-down-Widerstand 117
- Pull-up-Widerstand 113, 116, 117, 118, 181
- PWM (Pulse Width Modulation) 122, 133
- PWM-Signal 151, 155
- PWM-Wert 136
  
- Q**
- Quittungstöne 157
  
- R**
- RAM 16
- random(min, max) 128
- randomSeed(seed) 128
- RC-Glied 137, 151
- RC-Tiefpass 152
- Referenzspannung 119, 120
- Reflektiv 239
- Relais 151
- Reset-Taster 40
- Ringbuffer 111
- Rippel 152
- RISC 17
- RISC-Technologie 16
- Routinen 90
- RTC 163, 165
  
- S**
- Satelliten 226
- Schaltdraht 48
- Schleifen 86
- Schnittstelle 8
- Schutzdiode 116
- SCL 210
- SDA 210
- seed 128
- Semikolon 75
- Sensortaster 186
- Serial.available() 103
- Serial.begin(Baudrate) 102
- Serial.end() 103
- Serial.flush() 104
- Serial.print() 100, 104
- Serial.println() 100, 106
- Serial.read() 103
- Serial.write() 107
- Serielle Ein-/Ausgabe 100
- Serielle Übertragung 108
- Servo 233
- Shields 31
- Signal 196
- Siliziumdiode 47
- sin(rad) 98
- Sinusfunktion 135, 138
- Sinustabellen 137
- Smart Project 21
- Software UART 111
- Sortimentsbox 8
- Soundbefehl 157
- Spannung 124
- Spannungs-Plotter 193, 198
- Speichern 62
- Speicheroszilloskop 196
- Spikes 138
- Spreizwiderstand 246
- sq(x) 97
- Sqrt(x) 98
- SRF02 221

- StampPlot 198  
 Startbit 108  
 State Machine 188  
 Steckbrett 50  
 Steuern 202  
 Steuerung Gewächshaus 13  
 Stiftleiste 248  
 STNs (Super-Twisted-Nematics) 238  
 Stopp 62  
 Stoppbit 108  
 String 78, 108  
 Strom 148  
 Stromversorgung 40, 233  
 Sub Routine 90  
 switch case 85  
 Syntaxfehler 65
- T**
- tan(rad) 100  
 Taster 48, 138  
 Tasterzustand 114  
 Tastverhältnis 122  
 Teileliste 37, 38  
 TellyMate 33  
 Temperaturschalter 205  
 Temperatursensoren 213  
 Terminal 62, 64, 109  
 Terminal-Ausgaben 254  
 Terminal-Befehle 254  
 Terminal-Programm 73  
 Tiefpass 151  
 Timer 169
- TN(Twisted-Nematic)-Displays 238  
 Toleranzangabe 43  
 tone() 156  
 Tonerzeugung 156  
 Toolbar 62  
 Transfektiv 239  
 Transistor 47, 148, 149, 151  
 Transistor-LED-Dimmer 133  
 Transmissiv 239  
 Treiber 51  
 Trimmwiderstand 49  
 Türöffner 177  
 Typenkonvertierung 104  
 Typenumwandlung 93
- U**
- UART 100, 112  
 UART-Schnittstelle 51, 111, 193  
 Uhr 163  
 Uhrzeit 168  
 Ultraschallsensor 221  
 Umgebungstemperaturbereich 241  
 Unsigned Char 77  
 Unsigned int 77  
 Unsigned Long 77  
 USB 40  
 USB-Buchse 40  
 USB-Chip 51  
 USB-seriell-Wandler 58  
 USB-zu-Seriell-Adapter 52
- V**
- Variablen 75  
   lokale 75  
   globale 76  
 Variablen-Namen 75  
 VB.NET 192, 194, 196, 202  
 VB.NET-Programm 137  
 V-DOP 226  
 Vergleich 80  
 Verstärkung 148  
 Vf 145, 205  
 Virtueller Comport 51  
 Visualisieren 191  
 void loop() 67, 91  
 void setup() 67, 91  
 Voltmeter 193  
 Vorwiderstand 145  
 VT100 253
- W**
- while 88  
 Widerstände 43  
 Wire-Bibliothek 215  
 Wiznet W5100 36
- X**
- XBee 35
- Z**
- Zeichenattribute 254  
 Zeichensatz 242  
 Zeit 130  
 Zufallszahlengenerator 161

Ulli Sommer

# Arduino

**Arduino ist ein Mikrocontroller-System, das aus einem Mikrocontroller der Firma Atmel und einer Open-Source-Entwicklungs-umgebung, die auf einem vereinfachten C-Dialekt basiert, besteht.**

Der Mikrocontroller wird über den PC programmiert und kann eigenständig oder in Verbindung mit dem PC agieren. Es können für die Interaktion zwischen Mensch und Mikrocontroller diverse Sensoren angeschlossen werden, die unsere Umwelt erfassen und die Daten an den Mikrocontroller weitergeben. Der Mikrocontroller verarbeitet mit seinem Programm die Daten, und es können Ausgaben getätigt oder z. B. Aktuatoren gesteuert werden. Der Kreativität des Entwicklers sind dabei keine Grenzen gesetzt.

Die Arduino-Programmieroberfläche unterstützt den Entwickler bei seinen Vorhaben durch ihre vorgefertigten Programme und Funktionsbibliotheken. Das einfache Zusammenspiel aus Hard- und Software bildet die Basis für Physical Computing, also die Verbindung der realen Welt mit der Welt des Mikrocontrollers, die aus Bits und Bytes besteht. Dieses Buch bietet Ihnen einen unkomplizierten Einstieg in diese Welten.

Die ersten Kapitel vermitteln Ihnen die Programmierung mit Arduino. Die C-Befehle werden anhand kleiner Beispiele verdeutlicht, Hard- und Software werden detailliert erklärt. Schließlich setzen Sie Ihre neu erworbenen Kenntnisse in Experimenten kreativ und spielerisch in Mess-, Steuer- und Regelanwendungen ein. Nach der Lektüre dieses Buchs werden Sie in der Lage sein, Ihre eigenen Ideen selbstständig umzusetzen.

## CD-Inhalt:

- Open-Source-Soft- und Hardware
- Über 80 Quellcodes zu den Experimenten
- Schaltpläne und Datenblätter
- Open-Source-VB.NET-Programme zum Messen und Steuern mit Arduino/Freeduino

## Aus dem Inhalt:

- Mikrocontroller-Grundlagen
- Mikrocontroller-Programmierung mit Arduino/Freeduino
- Aufbauanleitung zu jedem Experiment
- Von den Grundlagen bis zur eigenen Applikation
- Entwickeln Sie Ihre eigenen Anwendungen und damit praktisch Ihr eigenes Spezial-IC: sei es eine spezielle Alarmanlage, ein Messgerät oder eine Robotersteuerung
- **Über 80 praktische Experimente:**

Den USB-Brückenchip FT232RL einrichten und anwenden, Ein-/Aus-schaltverzögerung, Temperaturschalter, Kapazitätsmessgerät, Schuluhr mit RTC, 6-Kanal-Digitalvoltmeter, Lüftersteuerung, Datenaustausch zwischen VB.NET und Arduino, Sensortaster, Statemaschine, Daten aufzeichnen mit Stamp PLOT, Digitales Speicheroszilloskop, Bewegungsmelder-Alarmanlage, Dämmungsschalter, romantisches Kerzenlicht, Musikplayer, Datenplotter mit VB.NET, serielle Ein- und Ausgabe, Experimente mit LCD-Displays und vieles mehr

ISBN 978-3-645-65034-2



Euro **29,95** [D]