

# Lernpaket **Arduino**

Mit 101 Abbildungen

**FRANZIS**

# 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 ihr vorgefertigtes Programm und ihre 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 Lernpaket zeigt Ihnen Schritt für Schritt, wie Sie den leichten Einstieg in diese Welt finden.

Viel Spaß beim Lesen und Experimentieren mit diesem Lernpaket!

Ulli Sommer

# CD-ROM zum Lernpaket

Diesem Lernpaket 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).

# 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 .....	16
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 .....	25
3.6	Arduino Pro.....	26
3.7	LilyPad .....	27
3.8	USB-Adapter .....	28
<b>4</b>	<b>Arduino Shields .....</b>	<b>29</b>
4.1	Arduino ProtoShield .....	29
4.2	Ardumoto.....	30
4.3	TellyMate .....	31
4.4	ArduPilot.....	32
4.5	Ethernet Shield .....	34
<b>5</b>	<b>Bauteile des Lernpakets .....</b>	<b>35</b>
5.1	Bauteileliste.....	35
5.2	Das Freeduino-Experimentierboard .....	36
5.3	Anschlüsse und LEDs des Freeduino-Mikrocontroller- Experimentierboards.....	36
5.4	Die Stromversorgung.....	37
5.5	Reset-Taster .....	37

5.6	ISP-Anschluss .....	37
5.7	Sicherheitshinweise.....	39
<b>6</b>	<b>Bauteile und ihre Funktion .....</b>	<b>41</b>
6.1	Leuchtdioden.....	41
6.2	Widerstände .....	41
6.3	Kondensatoren .....	43
6.4	Transistoren.....	45
6.5	Diode .....	45
6.6	Piezo-Schallwandler (Buzzer).....	46
6.7	Schaltdraht.....	46
6.8	Taster.....	46
6.9	Potenziometer.....	47
6.10	LDR .....	47
6.11	Steckbrett .....	47
<b>7</b>	<b>Die ersten Vorbereitungen (Inbetriebnahme) .....</b>	<b>49</b>
7.1	Treiberinstallation.....	49
7.2	Das Tool MProg für den FT232RL .....	51
7.3	FT232R mit MProg programmieren .....	55
7.4	Die Arduino-Software installieren.....	56
<b>8</b>	<b>Die Arduino-Entwicklungsumgebung .....</b>	<b>59</b>
8.1	Einstellungen in der Arduino-IDE.....	61
8.2	Der erste Funktionstest »ES_Blinkt«.....	62
8.3	Was haben wir getan? .....	65
<b>9</b>	<b>Arduino-Programmiergrundlagen.....</b>	<b>67</b>
9.1	Bits und Bytes.....	67
9.2	Grundsätzlicher Aufbau eines Programms.....	68
9.2.1	Sequenzieller Programmablauf .....	68
9.2.2	Interruptgesteuerter Programmablauf .....	69
9.3	Der Aufbau eines Arduino-Programms.....	70
9.4	Das erste eigene Programm mit Arduino.....	70
9.5	Arduino-Befehle und ihre Verwendung.....	72
9.5.1	Kommentare im Quelltext.....	72
<b>10</b>	<b>Weitere Experimente mit Arduino.....</b>	<b>131</b>
10.1	Der Transistor-LED-Dimmer .....	131
10.2	Softer Blinker .....	133
10.3	Taster entprellen .....	136
10.4	Einschaltverzögerung.....	141
10.5	Ausschaltverzögerung.....	142

10.6	LEDs und Arduino.....	143
10.7	Größere Verbraucher schalten.....	146
10.8	DAC mit PWM-Ports.....	148
10.9	Mit Musik geht alles besser.....	152
10.10	Romantisches Mikrocontroller-Kerzenlicht.....	156
10.11	Überwachung des Personalausgangs.....	158
10.12	RTC (Real Time Clock).....	160
10.13	Schuluhrprogramm.....	162
10.14	Lüftersteuerung.....	165
10.15	Dämmerungsschalter.....	168
10.16	Alarmanlage.....	170
10.17	Codeschloss.....	173
10.18	Kapazitätsmesser mit Autorange.....	177
10.19	Potenziometer professionell auslesen.....	179
10.20	Sensor Taster.....	181
10.21	State Machine.....	183
10.22	Ein 6-Kanal-Voltmeter mit Arduino.....	186
10.23	Spannungs-Plotter selbst programmiert.....	189
10.24	Das Arduino-Speicheroszilloskop.....	191
10.25	StampPlot der Profi-Datenlogger zum Nulltarif.....	193
10.26	Steuern über VB.NET.....	197
10.27	Temperaturschalter.....	200
<b>A</b>	<b>Anhang.....</b>	<b>203</b>
A.1	Arduino zu ATmega Pinmap.....	203
A.2	Escape-Sequenzen.....	203
A.3	ASCII-Tabelle.....	205
	<b>Bezugsquellen.....</b>	<b>210</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 I2C 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.

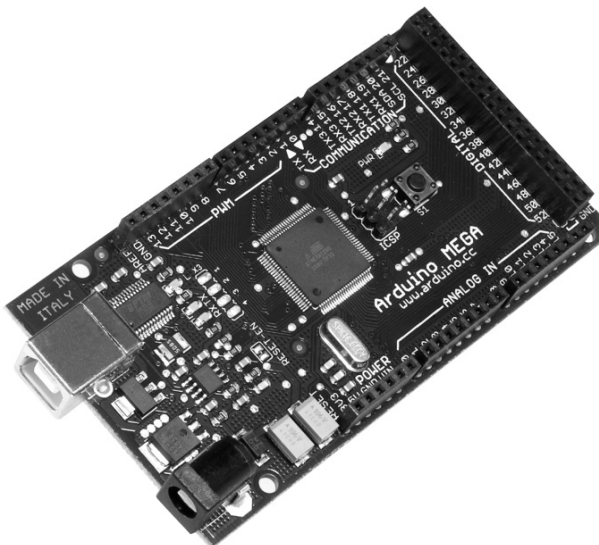
Das neue Arduino Mega Board verwendet einen leistungsstärkeren Mikrocontroller (Atmega1280) und bietet mehr Speicher, I/O-Pins und Funktionen auf einer deutlich erweiterten Platinenfläche.

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

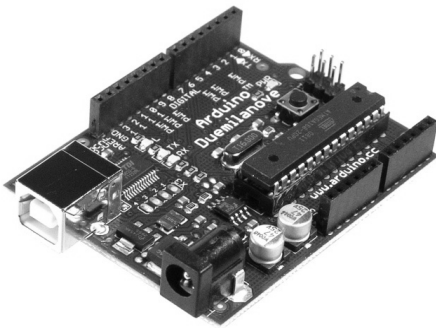


**Bild 3.1:**  
Arduino Mega  
(Quelle: Fa. Elmicro)

**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
- I2C-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 (2 KByte benötigt der Bootloader für sich)
- 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 (2 KB benötigt der Bootloader für sich)
- 1 KB (ATmega168) oder 2 KB (ATmega328) SRAM
- 512- (ATmega168) oder 1-KB-EEPROM

## 3.7 LilyPad

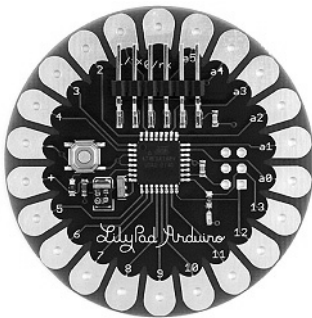


Bild 3.7: LilyPad Arduino (Quelle: Elmicro)

### 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 (2 KB benötigt der Bootloader für sich)
- 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)

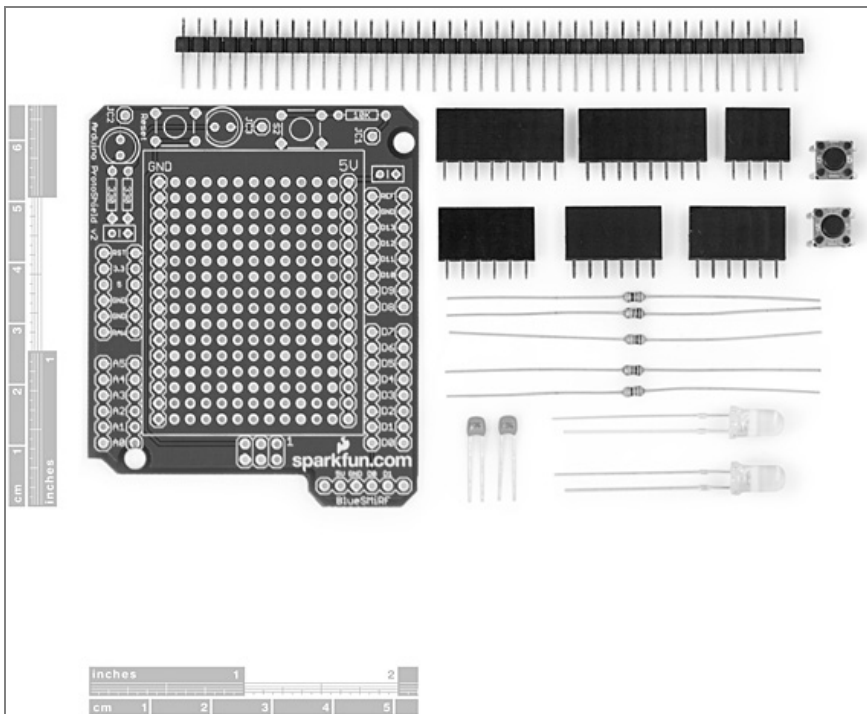
Diesen Programmieradapter gibt es in 3,3-V- und in 5-V-Ausführung.

Der Adapter wird zum Programmieren der Arduino-Boards 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.

## 4 Arduino Shields

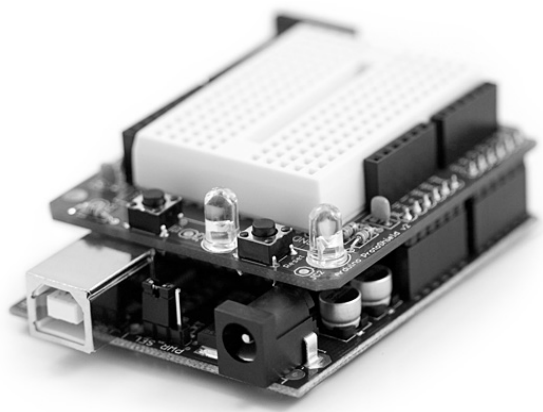
Es gibt eine Menge verschiedener Erweiterungs-Boards, die mit den Arduino-Boards verwendet werden können. Wenn man sich im Internet umsieht, findet man fast monatlich neue Boards und nützliche Erweiterungen. Die Erweiterungs-Boards werden in der »Arduino-Gemeinde« *Shields* genannt und besitzen alle den gleichen Formfaktor. Das hat den Vorteil, dass man sie einfach auf die Arduino-Boards aufstecken kann. Ausgenommen sind die kleinen Units und das LilyPad.

### 4.1 Arduino ProtoShield



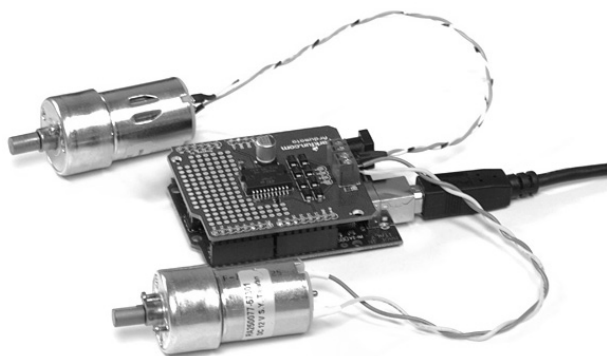
**Bild 4.1:** Arduino-ProtoShield-Kit (Quelle: Fa. SparkFun)

Für eigene Basteleien ohne Lötarbeit bietet sich das ProtoShield an. Es ermöglicht Experimente auf einem kleinen Steckbrett.



**Bild 4.2:** Arduino Duemilanove mit ProtoShield (Quelle: SparkFun)

## 4.2 Ardumoto

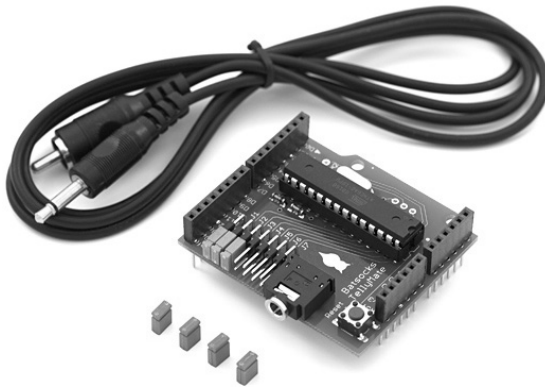


**Bild 4.3:** Ardumoto – Motor Driver Shield (Quelle: SparkFun)

Das Motor Driver Shield *Ardumoto* ist ideal, um kleine Motoren anzusteuern. Die Anschlussdrähte der Motoren werden einfach an die Schraubklemmen des Motor-Shields angeschlossen und ein kleines Programm lässt die Motoren mit gewünschter Geschwindigkeit und Richtung drehen.

Die technischen Daten entsprechen den verwendeten Motortreiber-ICs L298. Das Datenblatt finden Sie auf der mitgelieferten CD-ROM.

## 4.3 TellyMate



**Bild 4.4:** TellyMate  
(Quelle: SparkFun)

Das TellyMate ist wohl das genialste Shield, das es für Arduino gibt. Seine Einsatzmöglichkeiten sind fast unbegrenzt. Es ermöglicht, Daten (ADC, IOs usw.) oder auch einfach nur Texte oder Grafiken auf dem TV-Bildschirm darzustellen. Es macht unseren heimischen Fernseher zum Arduino-Display. Der Arduino-Mikrocontroller verwendet zur Kommunikation mit TellyMate die serielle Schnittstelle.

### Features:

- Arduino TV-Ausgabe
- PAL-oder NTSC-Composite-Video
- Aufsteckbares Arduino-Shield
- Arbeitet mit `Serial.println ()` usw.
- 38 x 25 Zeichen
- Darstellung der Zeichen schwarz, weiß
- Einfache Grafiken
- Einfache Programmierung

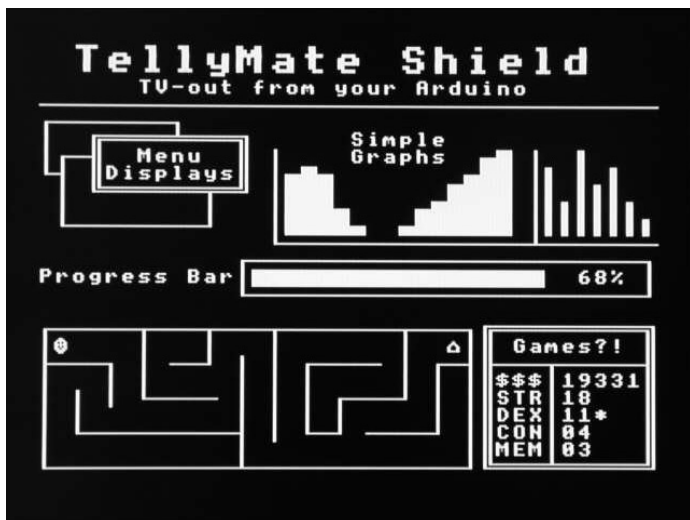


Bild 4.5: TellyMate in Aktion (Quelle: SparkFun)

## 4.4 ArduPilot

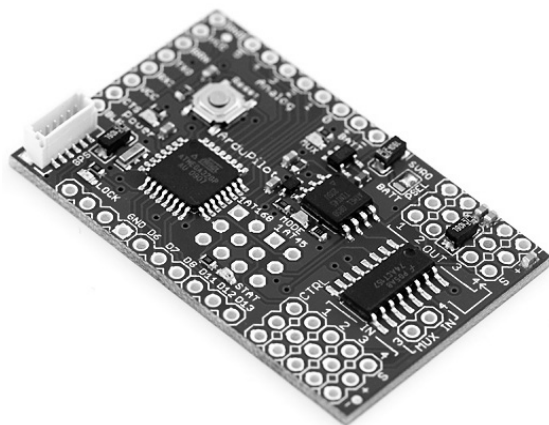


Bild 4.6: ArduPilot –  
 Arduino-kompatibler UAV-  
 Controller ATmega328  
 (Quelle: SparkFun)

Für die Modellflieger ist der ArduPilot ein äußerst interessantes Spielzeug. Es ermöglicht das autonome Fliegen eines Modellflugzeugs.

Mehr dazu finden Sie unter <http://diydrones.com>.

## XBee ZNet



**Bild 4.7:** XBee ZNet 2.5 OEM-Modul  
(Quelle: SparkFun)

Wer drahtlos Daten übermitteln möchte, sollte sich die XBee-Module zulegen. Diese stellen eine drahtlose serielle (UART-)Verbindung her. Man könnte damit zwei Arduino-Boards oder einen PC und ein Arduino-Board über eine Funkstrecke kommunizieren lassen.

### Technische Daten des ZigBee-Funkmoduls:

- Betriebsspannung: 2,8 V bis 3,4 V
- Frequenz: ZigBee Standard, 2,4 GHz ISM-Band
- Sendeleistung: 0 dbm (1 mW)
- Empfindlichkeit: -92 dbm
- Reichweite: 30 m Indoor / 100 m Outdoor (abhängig von Umgebungsbedingungen)
- Stromaufnahme TX: 45 mA, RX: 50 mA, Stand-by: 10 uA
- Datenrate (über Funk): 250.000 bps
- Datenrate (Interface): 1.200-115.200 bps
- serielle Schnittstelle: 0 V/3,3 V zum Anschluss an den PC ist ein 3,3-V-Pegelwandler (MAX3232) zwingend erforderlich
- Standard: kompatibel zu ZigBee/802.15.4
- Topologien: Point-To-Point, Point-To-Multipoint
- Abmessungen: 24,38 mm x 27,61 mm x 4mm, 2-mm-Raster

## 4.5 Ethernet Shield



**Bild 4.8:** Ethernet Shield  
(Quelle: Solarbotics)

Das Arduino Ethernet Shield ermöglicht, ein Arduino-Board mit dem Netzwerk/Internet zu verbinden. Es basiert auf dem Wiznet W5100 Ethernet-Chip. Der Wiznet W5100 bietet ein Netzwerk-(IP-)Stack, der TCP und UDP unterstützt. Er ermöglicht zudem bis zu vier gleichzeitige Socket-Verbindungen. Arduino bietet eine umfangreiche Bibliothek und verschiedene Beispielprogramme an, um den Einstieg in die Netzwerkwelt zu erleichtern.

# 5 Bauteile des Lernpakets

Wir haben in den letzten Kapiteln einiges über Mikrocontroller, Arduino und deren Anwendungsgebiete sowie über die Programmierung erfahren. Nun ist es an der Zeit, sich an die Experimente mit Arduino zu wagen. Es sind folgende Teile im Lernpaket vorhanden:

## 5.1 Bauteileliste

- 1x Freeduino-Mikrocontrollerboard
- 1x Steckbrett Tiny
- 2x Printtaster RM 2,54
- 1x LDR A9060-13 ( $R_{100} = 5 \text{ k}\Omega$ )
- 1x NPN-Transistor BC548C
- 1x Silizium-Dioden 1N4148
- 1x Piezo-Schallwandler
- 1x LED rot
- 1x LED grün
- 2x LED gelb
- 3x Widerstand  $1,5 \text{ k}\Omega$
- 1x Widerstand  $4,7 \text{ k}\Omega$
- 1x Widerstand  $47 \text{ k}\Omega$
- 1x Widerstand  $10 \text{ k}\Omega$
- 1x Widerstand  $68 \text{ k}\Omega$
- 1x Trimmwiderstand linear  $10 \text{ k}\Omega$  PT10
- 1x Kondensator  $1 \mu\text{F}$
- 1x Wickelschaltdraht

## 5.2 Das Freeduino-Experimentierboard

Die kleine Freeduino-Experimentierplatine entspricht dem Arduino-Duemilano-Board. Freeduino ist die Freeware-Version von Arduino, aber vollständig kompatibel zu Arduino Duemilano. Nachfolgend werden wir es ganz umgangssprachlich »Mikrocontroller« und »Mikrocontrollerboard« nennen.

Das Board besitzt einen USB-Anschluss, der die Verbindung zwischen PC und AVR herstellt. Über den USB-Anschluss werden sowohl die Programme *Sketches* als auch die Daten aus unserem Arduino-Programm zum PC (serielle Schnittstelle) übertragen. Die Platine dient durch die bereits vorhandene Hardware-Ausstattung als Basis für die Experimente in diesem Lernpaket. Für die größeren Experimente benötigen wir das kleine Steckbrett *Breadboard*, auf das Sie die Bauteile aufstecken und mit den Freeduino-Board verdrahten können.

Bauen Sie einfache Experimente mit dem beiliegenden Experimentiermaterial auf und laden Sie die vorbereiteten Programme über USB in den Mikrocontroller. Spielend leicht lernen Sie die Funktionen der Bauteile und zugleich die Grundlagen der Mess- und Steuerungstechnik sowie die Programmierung mit Visual Basic Dot.Net und den Umgang mit Arduino.

## 5.3 Anschlüsse und LEDs des Freeduino-Mikrocontroller-Experimentierboards

Alle Arduino-Anschlüsse sind über Stiftleisten erreichbar. Eine Power ON-LED *PWR* signalisiert, dass unser Mikrocontrollerboard mit Strom versorgt wird. Eine LED auf der Platine, die fest mit dem Arduino-Digital-Pin 13 verbunden ist, trägt den Namen *L*. In die Stiftleisten können die Drähte und Bauteile direkt eingesteckt werden. Die LEDs mit der Bezeichnung *TX* und *RX* zeigen den Datenverkehr auf der seriellen Schnittstelle an. Sobald Kommunikation über die UART-Schnittstelle stattfindet, blinken die LEDs rhythmisch mit.

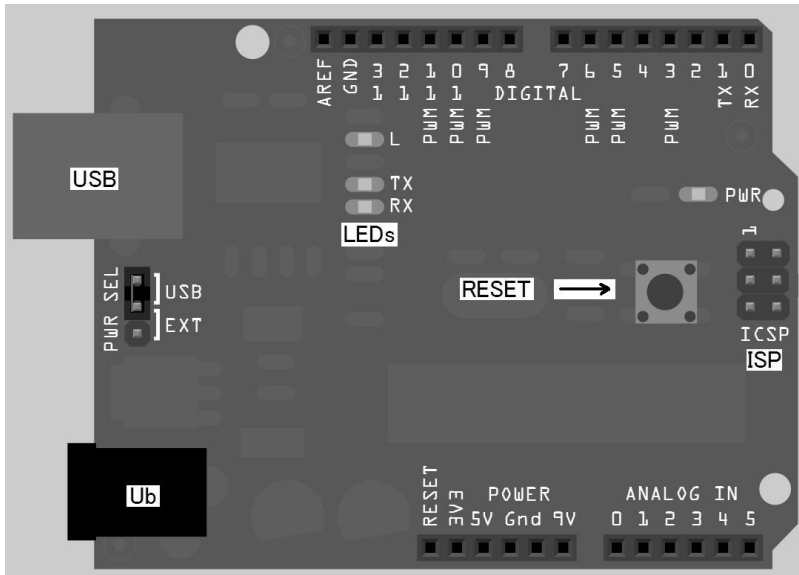


Bild 5.1: Die Freeduino-Experimentierplatine

## 5.4 Die Stromversorgung

Die Stromversorgung kann wahlweise über USB oder über ein Steckernetzgerät (ca. 500 mA) erfolgen. Dazu muss nur der Jumper (die Steckbrücke) mit der Bezeichnung *PWR SEL* wahlweise nach *USB* oder *EXT* gesteckt werden. Möchten wir die Stromversorgung über USB auswählen, stecken wir den Jumper Richtung USB-Buchse. Bei größeren Verbrauchern am Mikrocontrollerboard ist davon jedoch abzuraten, da es sich negativ auf den verwendeten USB-Port auswirken kann.

## 5.5 Reset-Taster

Der Reset-Taster bewirkt einen Neustart des Programms. Er hat die gleiche Wirkung, als wenn das Board vom Netz getrennt und wieder angesteckt wird.

## 5.6 ISP-Anschluss

Der ISP-Anschluss dient zum Programmieren des Mikrocontrollers über einen ISP-Programmer und zum Aufspielen des Bootloaders. Für unsere Experimente benötigen wir diesen Anschluss nicht. Der Bootloader ist bereits ab Werk installiert.

**TIPP:**

Benutzen Sie zum Anschluss des Freeduino-Boards einen USB-HUB. Sollten Sie doch einmal versehentlich beim Experimentieren einen Kurzschluss verursachen, ist meist nur der USB-HUB defekt und nicht der USB-PORT des PCs.

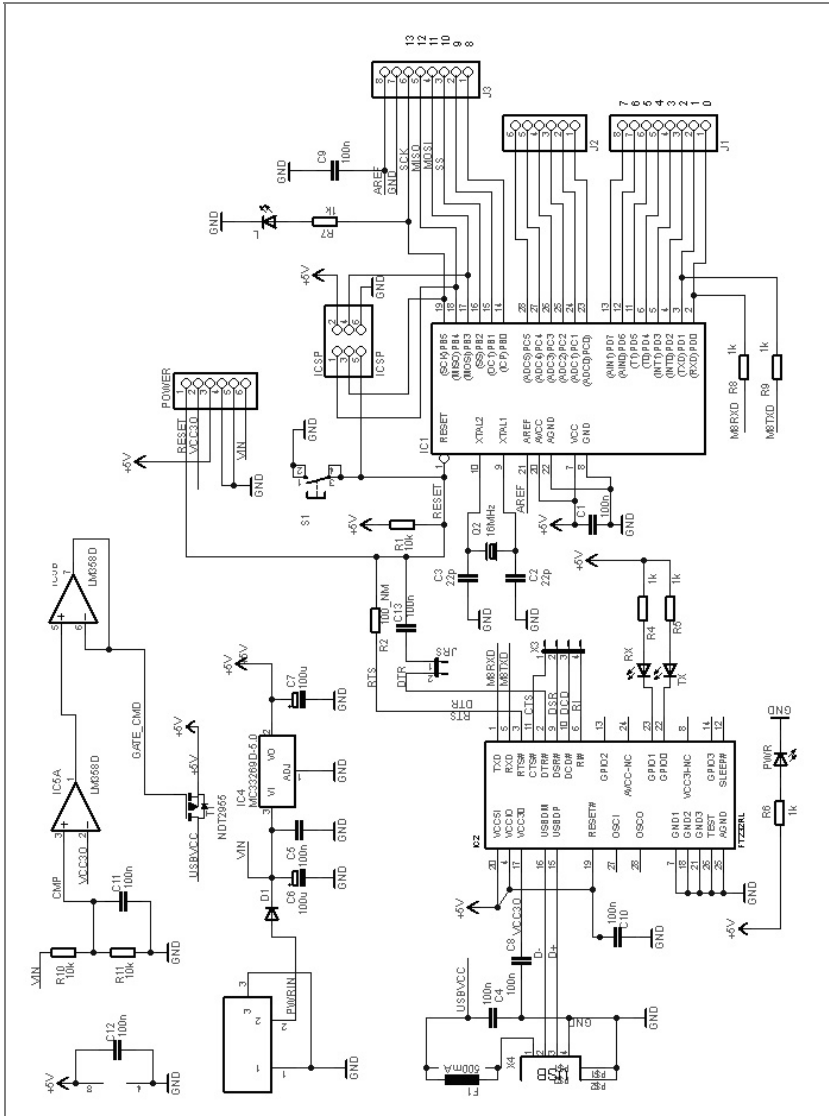


Bild 5.2: Schaltplan der Freeduino-Experimentierplatine

## 5.7 Sicherheitshinweise

Die Freeduino-Platine ist weitgehend gegen Fehler abgesichert, sodass es kaum möglich ist, den PC zu beschädigen. Die Anschlüsse der USB-Buchse sind auf der Platinenunterseite nicht isoliert. Wenn Sie die Platine auf einen metallischen Leiter stellen, kann es daher zu einem höheren Strom kommen, was den PC und die Platine beschädigen könnte. Nach der USB-Spezifikation sollte es an den USB-Downports eine Strombegrenzung geben, sodass eigentlich nichts passieren sollte. Allerdings besteht die Schutzschaltung in einzelnen Fällen aus kleinen Widerständen, die dann wie eine Sicherung durchbrennen. Beachten Sie deshalb bitte die folgenden Sicherheitsregeln:

- Vermeiden Sie metallische Gegenstände unter der Platine oder isolieren Sie die gesamte Unterseite mit einer nichtleitenden Schutzplatte oder Isolierband.
- Halten Sie Netzteile, andere Spannungsquellen oder führende Leiter mit mehr als 5 V von der Experimentierplatine fern.
- Schließen Sie die Platine nach Möglichkeit nicht direkt an den PC an, sondern über einen Hub. Dieser enthält meist eine zusätzliche wirksame Schutzschaltung. Wenn doch etwas passiert, wird im Normalfall nur der Hub und nicht der PC beschädigt.