FRANZIS
PC & ELEKTRONIK

Ulli Sommer

- Über 80 praktische Experimente
- Grundlagenkurs zur Programmierung

Arduino

Mikrocontroller-Programmierung mit Arduino/Freeduino

Inhaltsübersicht

B Bezugsquellen

Stichwortverzeichnis

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 ditgitaler 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 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 Liefernachweise 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.

1 Mikrocontroller-Grundlagen

Bevor wir uns näher mit Arduino beschäftigen, ist es wichtig, einen allgemeinen Überblick über die Mikrocontroller zu gewinnen. Mikrocontroller werden vor allem im Bereich der Automatisierungs-, der Mess-, Steuer- und Regeltechnik eingesetzt. Der Vorteil eines Mikrocontroller-Systems ist, auf kleinstem Raum energie- und kosteneffizient physikalische Größen zu messen und zu interpretieren, um darauf aufbauend Entscheidungen zu treffen und Aktionen durchzuführen.

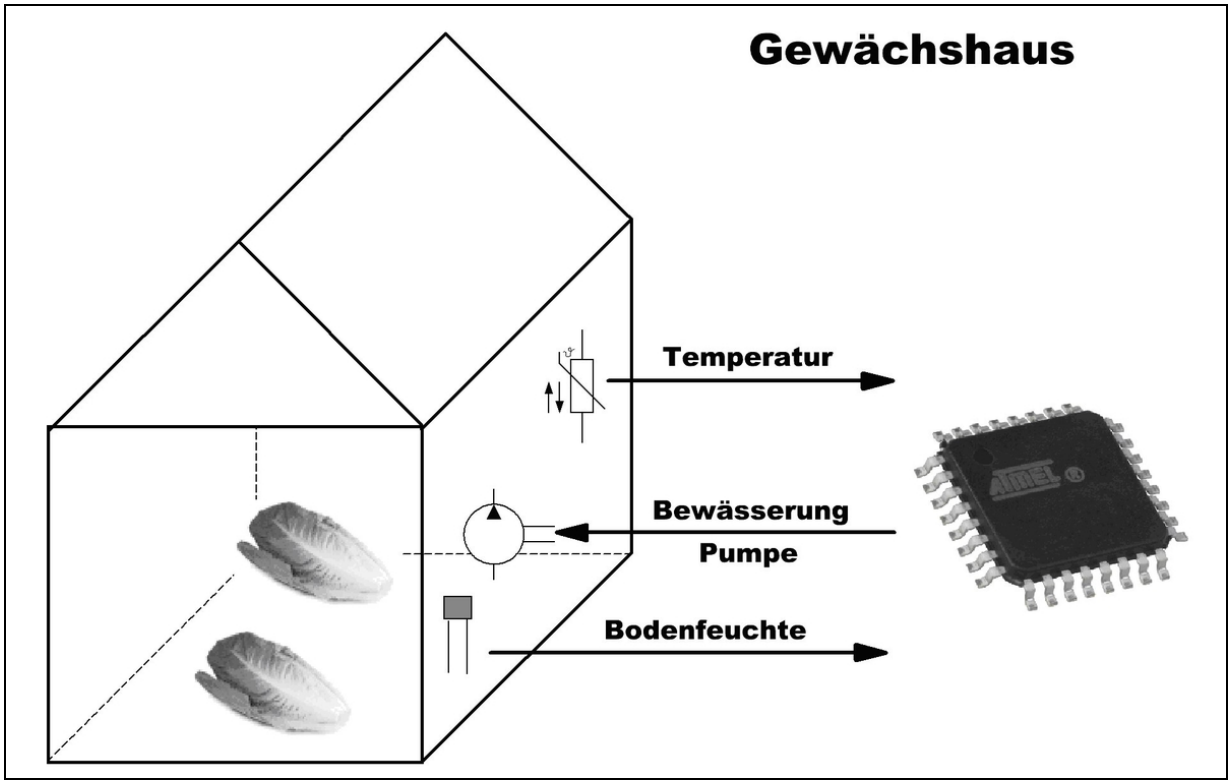

Bild 1.1 Beispiel einer Ein- und Ausgabeverarbeitung anhand eines Gewächshauses

Das Spektrum möglicher Anwendungen von Mikrocontrollern reicht vom privaten (z. B. der Steuerung eines Gewächshauses oder der Hausbelichtung) bis zum industriellen Bereich, wo komplette Anlagen mit Mikrocontroller-Systemen gesteuert, gewartet und betrieben werden können. Das obige Bild zeigt eine typische Datenverarbeitung zur Steuerung einer Bewässerungsanlage eines Gewächshauses. Der Controller nimmt dabei über Sensoren die Messwerte der Umgebungstemperatur und Bodenfeuchte auf. Die Messwerte werden durch eine digitale Logik im Mikrocontroller (kurz: *µC* oder *MC* genannt) interpretiert. Daraufhin wird die Pumpe für die Bewässerung entsprechend angesteuert.

1.1 Aufbau und Funktionsweise

Als vollwertiger Computer im Kleinstformat weist jeder Mikrocontroller – ähnlich einem PC – grundlegende Bausteine auf, die in Abb. 1.2 dargestellt sind. Grundbausteine eines jeden Mikrocontrollers sind die CPU, der Arbeitsspeicher (RAM) sowie der Programmspeicher (FLASH) und die Peripherie.

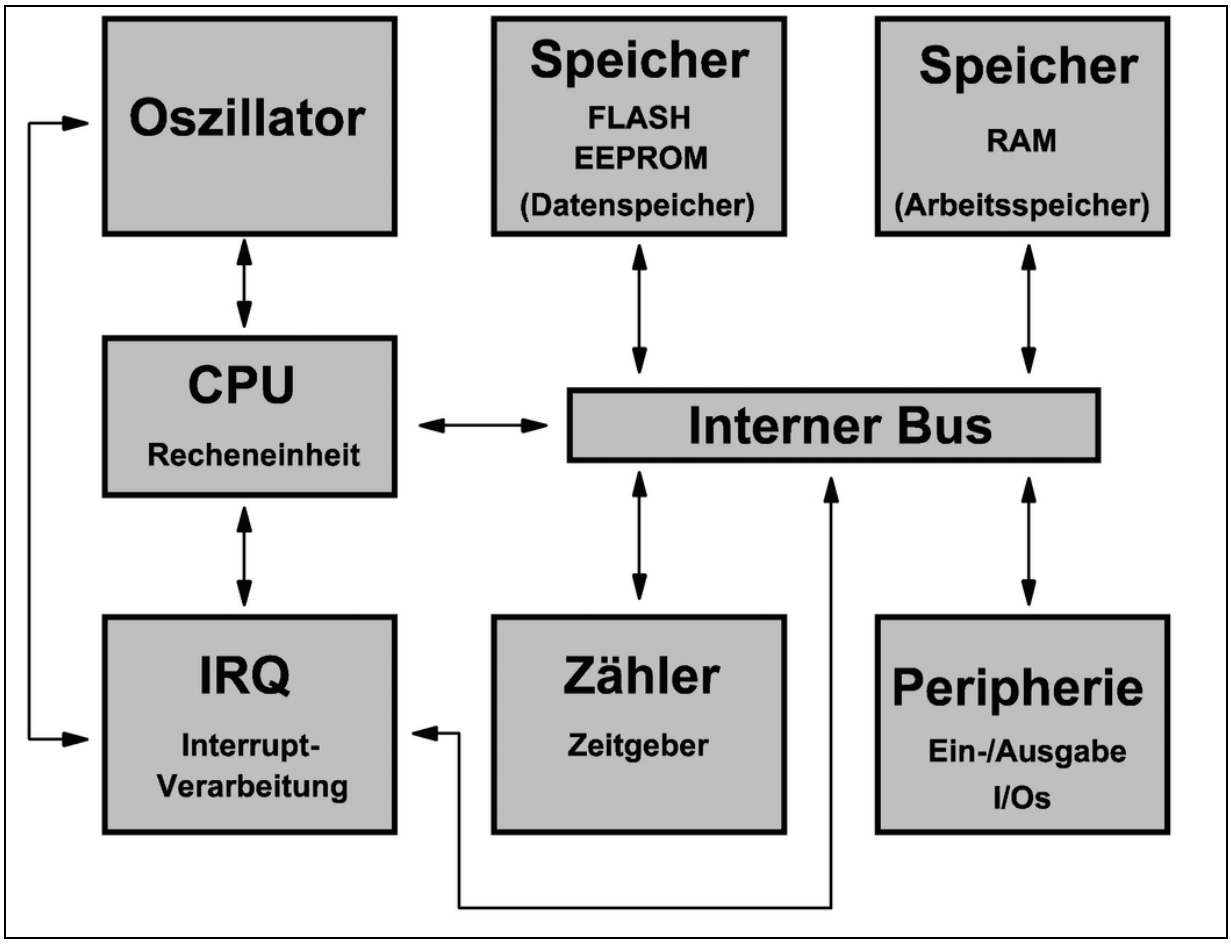

Bild 1.2 Prinzipieller Aufbau eines Mikrocontrollers

Die CPU

Die wichtigste Funktionseinheit ist die zentrale Recheneinheit, die *CPU (*engl.: Central Processing Unit). Sie kann als das »Gehirn« des Mikrocontrollers verstanden werden. Dort werden Signale ausgewertet und Befehle und arithmetische Operationen abgearbeitet.

Arbeits- und Programmspeicher

Arbeits- und Programmspeicher werden in vielen Darstellungen in der Regel logisch getrennt. Das Benutzerprogramm unseres eigenen Programms, das wir selbst geschrieben haben, wird in einem nichtflüchtigen *Flash-Speiche r*, dem Programmspeicher, abgelegt. Je nach Controllersystem kann man auf (implementieren) Programmspeicher von mehreren Kilobyte (KB) bis Megabyte (MB) zurückgreifen. Bei einigen Systemen ist es darüber hinaus möglich, den Programmspeicher durch externe Flash-Komponenten aufzustocken.

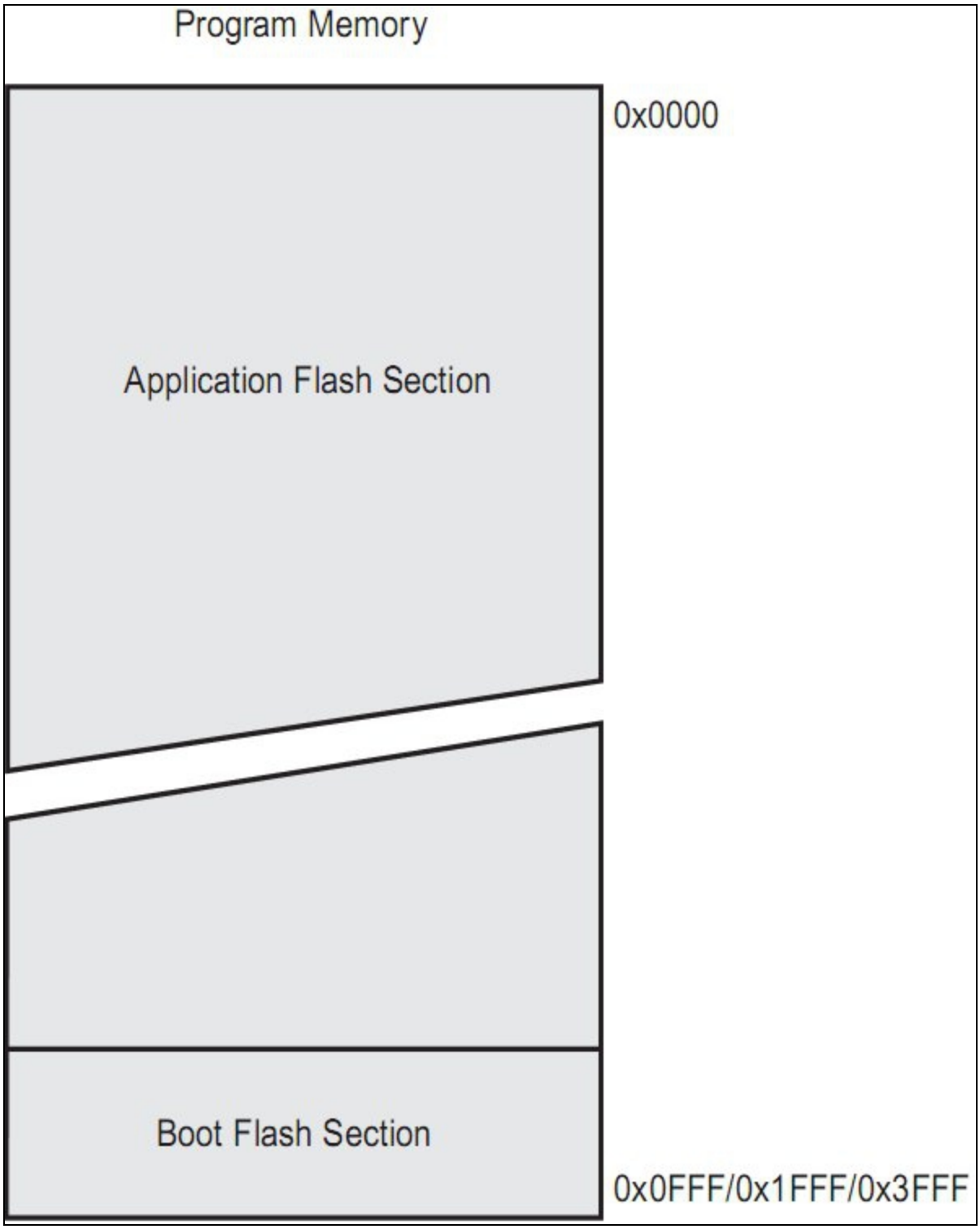

Bild 1.3 Der Flashspeicher des Mikrocontrollers ATmega168PA (Quelle: ATMEL-Datenblatt)

Der Arbeitsspeicher dient zur temporären Ablage von Rechen-, Mess- und Steuergrößen. Hier werden die Ergebnisse der Berechnungen zur Programmlaufzeit abgelegt. Ziel ist, möglichst schnell auf eine begrenzte Anzahl von Daten zugreifen zu können. Dieser RAM-Speicher (engl.: Random Access Memory) ist in der Regel deutlich kleiner als der Flash-Speicher, dafür aber um ein Vielfaches schneller. Die Werte des RAM werden zur Laufzeit erzeugt und sind im Gegensatz zum Flash-Speicher flüchtig, d. h., dass im RAM nach einem Neustart des Controllers keine Werte gespeichert sind.

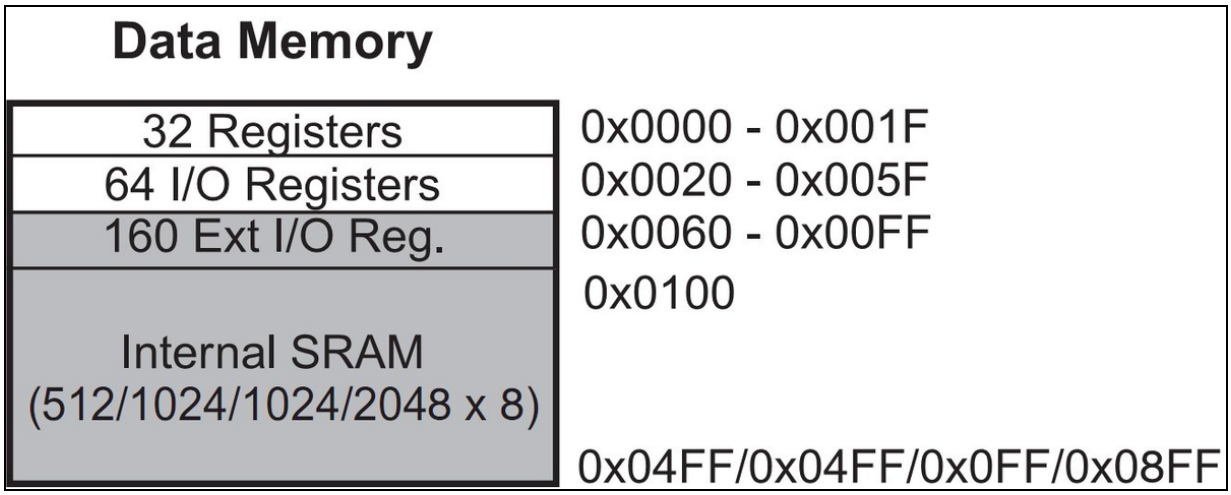

Bild 1.4 Der RAM-Speicher des Mikrocontrollers ATmega168PA (Quelle: ATMEL-Datenblatt)

1.2 Peripherie

Als »Peripherie« bezeichnet man jene Komponenten eines Mikrocontrollers, die nicht durch die CPU und Speicherbausteine abgedeckt werden. Insbesondere Komponenten, die eine Schnittstelle zur Außenwelt darstellen, wie digitale Ein- und Ausgänge (kurz: I/O für Input/Output), werden zu den Peripheriebausteinen gezählt. Die meisten Mikrocontroller bieten eine Vielzahl von Ein- und Ausgängen mit verschiedenen Funktionen wie digitale, aber auch analoge Ein- und Ausgänge (ADC und DAC).

1.3 Technologievergleich: RISC und CISC

Die Charakterisierung der RISC- und CISC-Technologie ist schon ein tiefer gehender Einblick in die Digital- und Mikrocontrollertechnik. AVR-Controller wie der Arduino basieren auf der RISC-Technologie. Der folgende Abschnitt bietet einen Überblick über die RISC- und die CISC-Technologie.

CISC-Technologie

Bei der CISC-Technologie wird der Programmspeicher in den RAM des Systems geladen und teilt sich diesen mit dem Programmspeicher. Man spricht auch davon, dass sich Programmcode und Daten derselben Speicher teilen. Das war insbesondere bei den ersten Computersystemen sinnvoll, da Arbeitsspeicher teuer war.

Ein weiteres, für Mikrocontroller viel entscheidenderes Merkmal ist der Aufbau von Befehlen. Ein CISC-Rechner verfügt über ein großes Sortiment an teils sehr speziellen Befehlen. In der Digitaltechnik ist ein Befehl eine Reihenfolge von bestimmten Bytes. Ein Byte kann 256 (0 bis 255) verschiedene Zustände annehmen. Um mehr als 256 verschiedene Befehle zu implementieren, benötigt man weitere Bytes. So kann es sein, dass ein spezialisierter Befehl aus mehreren Bytes (z. B. fünf Byte) besteht. Das Laden dieses Befehls dauert länger als das Laden eines Befehls, der nur ein Byte lang ist.

RISC-Technologie

Man hat festgestellt, dass bei CISC-Rechnern in der Regel etwa 90 % eines Quelltextes aus nur 30 verschiedenen Befehlen bestehen. Auf dieser Grundlage entstand der Gedanke, in der CPU weniger dafür aber kurze und schnelle Befehle zu implementieren. So findet man auf RISC-Mikrocontrollern in der Regel keine Befehle, die aus mehr als drei oder vier Bytes bestehen. Damit verfügt man nicht mehr über so viele spezialisierte Befehle und muss diese aus mehreren kurzen zusammensetzen. Um dabei mindestens die gleiche Leistungsfähigkeit zu erzielen wie bei einem CISC-Rechner, verfügen die meisten RISC-Rechner über eine große Anzahl von Registern. Ein Register ist ein in der CPU befindlicher temporärer, extrem schneller Speicher. Ein weiterer Gegensatz zur CISC-Technologie ist eine klare physikalische und logische Trennung zwischen Programm- und Datenspeicher.

Vergleich

Bei einem CISC-Rechner hat man eine Vielzahl spezialisierter Befehle, die in der Regel jedoch eine lange Abarbeitungszeit beanspruchen. Deutlich kürzere Abarbeitungszeiten erreichen die Befehle eines RISC-Rechners. Ein Nachteil dieser Technologie ist jedoch, dass hier die spezialisierten Befehle durch mehrere Befehle nachgebildet werden müssen. Die Vor- und Nachteile der CISC- oder RISC-Technologie halten sich etwa die Waage. Außerdem sollte beachtet werden, dass es keinen reinen RISC- und keinen reinen CISC-Rechner gibt.

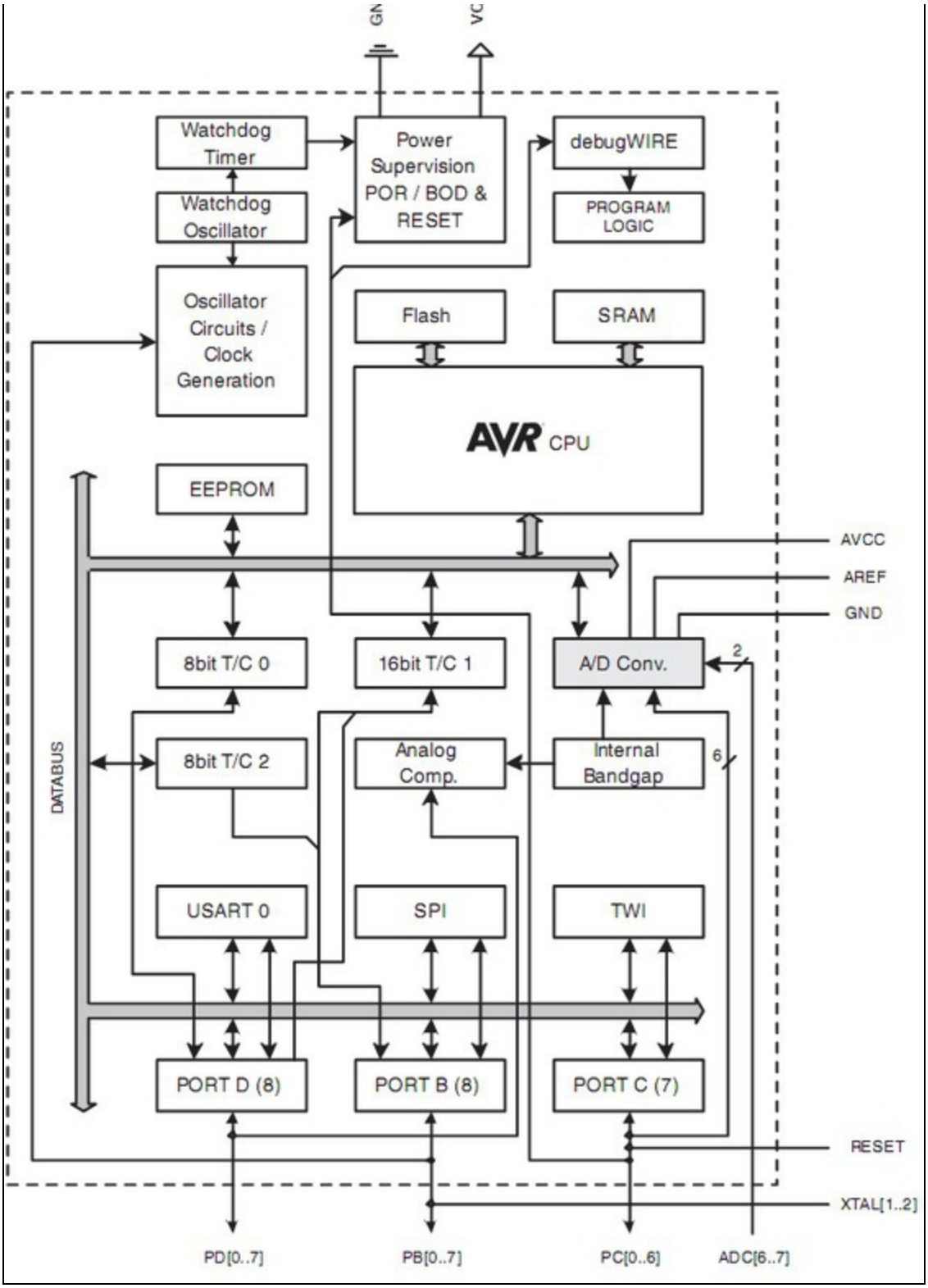

Bild 1.5 Das Blockschaltbild der Mikrocontroller-Blockschemata – hier sind die internen Strukturen des Controllers gut zu erkennen. (Quelle: ATMEL-Datenblatt)

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 Systemkerne 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^2C 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, handtellergroße 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

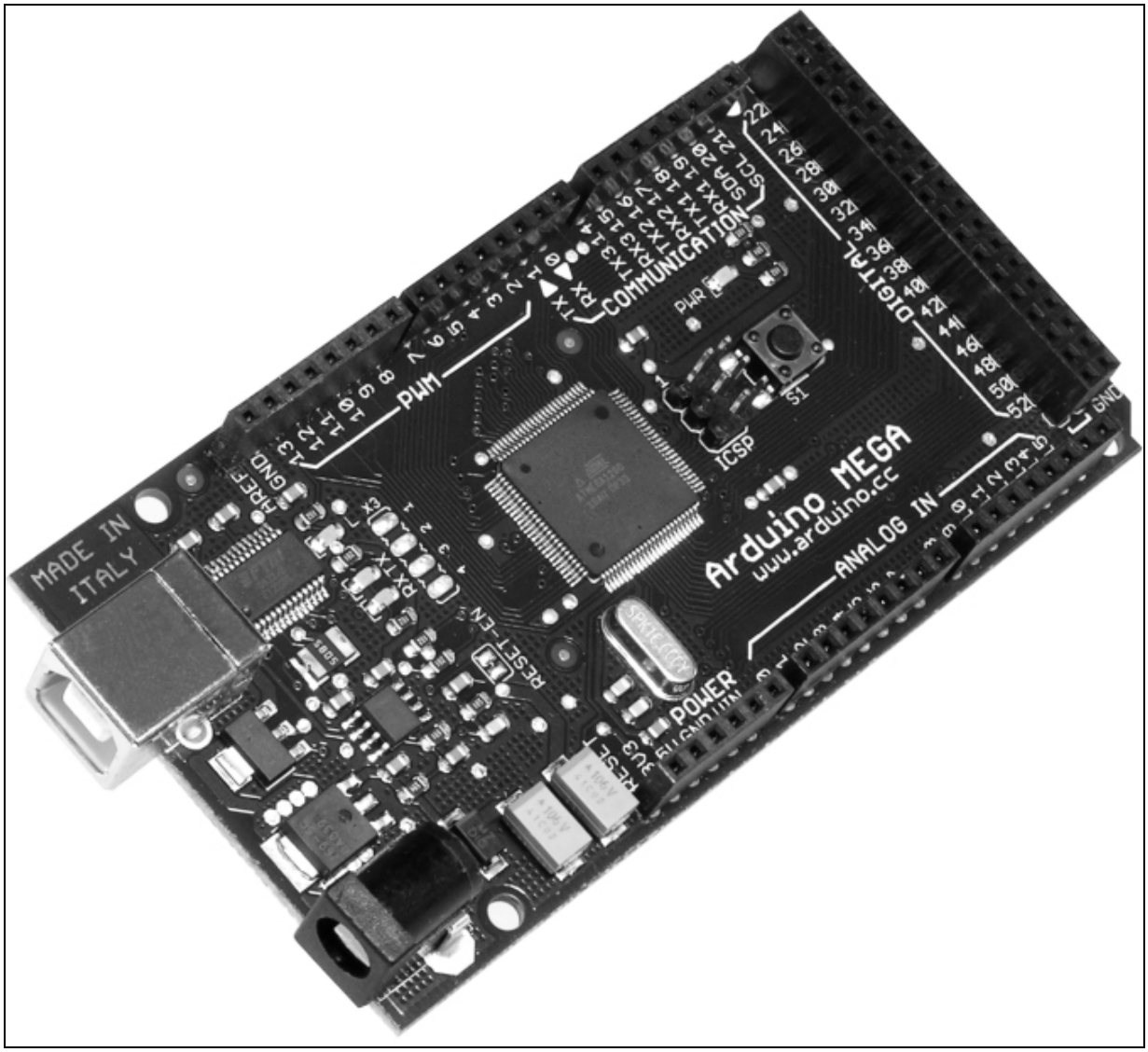

Bild 3.1 Arduino Mega (Quelle: Fa. Elmicro)

Technische Daten:

- ATmega1280 Mikrocontroller
- 128 KB Flash
- 8 KB RAM, 4 KB EEPROM
- 16-MHz-Takt