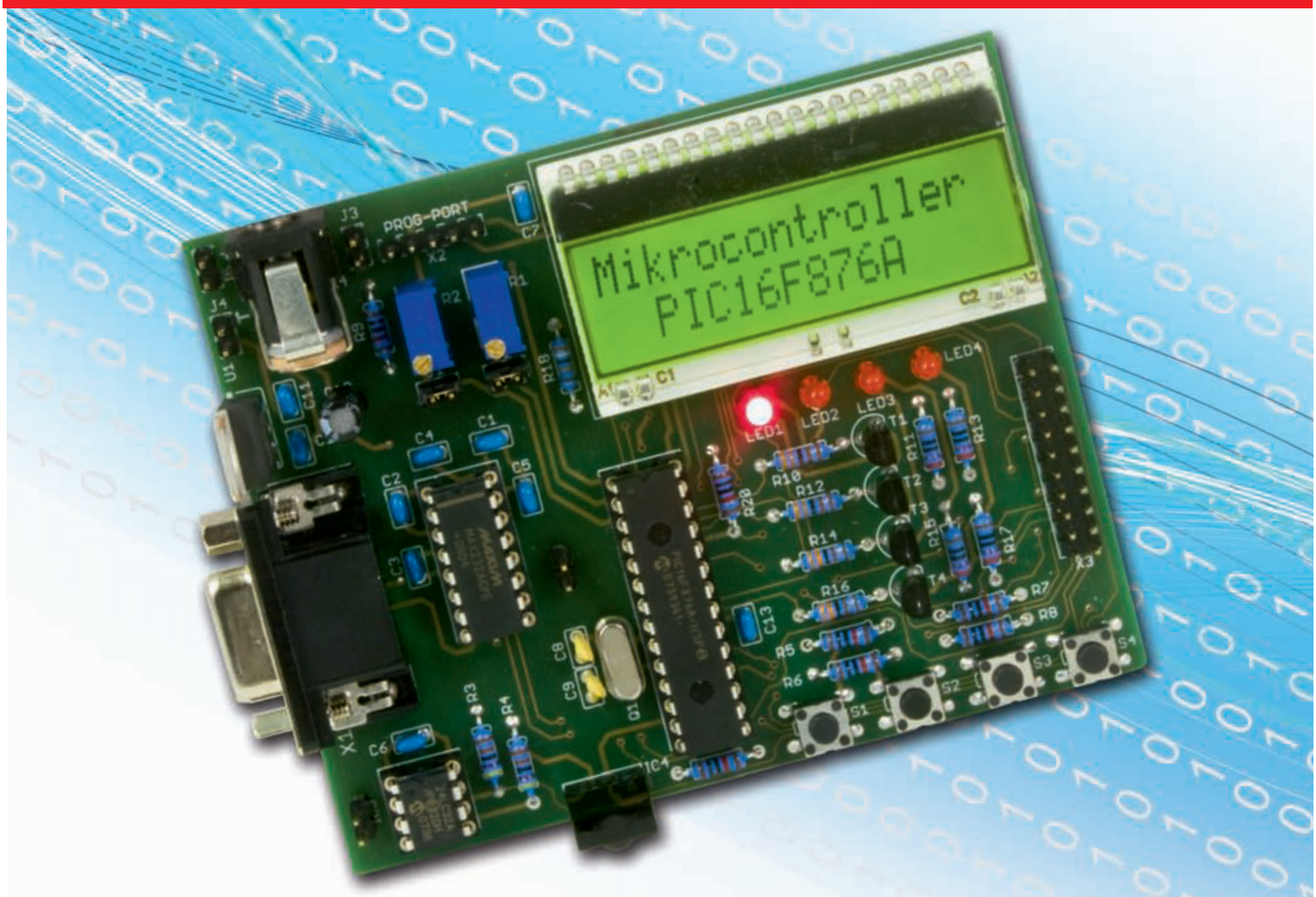
FRANZIS
PC & ELEKTRONIK



Michael Hofmann

Mikrocontroller für Einsteiger

Schaltungen entwerfen und Software programmieren

Auf CD-ROM:

- Beispielprogramme
- Layoutdaten
- Schaltpläne
- Datenblätter

Michael Hofmann

Mikrocontroller für Einsteiger

FRANZIS
PC&ELEKTRONIK

Michael Hofmann

Mikrocontroller für Einsteiger

Schaltungen entwerfen und Software programmieren

Mit 102 Abbildungen

Bibliografische Information der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte Daten sind im Internet über **http://dnb.ddb.de** abrufbar.

Satz: Fotosatz Pfeifer, 82166 Gräfelfing
art & design: www.ideehoch2.de
Druck: Bercker, 47623 Kevelaer
Printed in Germany

ISBN 978-3-7723-**4318-6**

Vorwort

Mikrocontroller sind in fast allen elektronischen Geräten zu finden. Dieses Buch zeigt, dass die Verwendung und Programmierung eines Mikrocontrollers nicht schwer ist. So manchem sträuben sich die Haare, wenn das Wort *Assembler* im Zusammenhang mit der Programmierung fällt. Nach der Lektüre dieses Buchs werden Sie feststellen, dass es nicht so kompliziert ist, wie Sie vielleicht vermutet haben.

Verschiedene Beispiele zeigen die Möglichkeiten eines Mikrocontrollers. Hierfür wird der *PIC16F876A* von Microchip verwendet. Dieser Typ verfügt über verschiedene Schnittstellen und Features und man kann damit eine große Bandbreite der Funktionalität vorstellen. Die Beispiele werden zeigen, wie Eingänge abgefragt und Ausgänge geschaltet werden. Sie werden auch lernen, wie man ein Display ansteuert, um Text und Daten anzuzeigen. Sie werden auch erfahren, wie man analoge Signale misst, diese in einem seriellen EEPROM speichert und anschließend über einen PC ausliest. Ein weiteres Beispiel zeigt die Steuerung der Ausgänge mit einer Infrarotfernbedienung. Ein Teil der Beispiele kann mit der Entwicklungsumgebung *MPLAB* simuliert werden. Daher wird keine funktionsfähige Hardwareschaltung benötigt. Da aber jeder Mikrocontroller irgendwann einmal in einer Schaltung eingesetzt werden soll, wird ein kleines Entwicklungs-Board vorgestellt, mit dem alle Beispiele in einer realen Umgebung getestet werden können. Um die Arbeit und die Fehlersuche für den Aufbau zu erleichtern, können Sie über meine Homepage (www.edmh.de) eine unbestückte Leiterplatte erwerben. Falls Sie eigene Erweiterungen vornehmen wollen, finden Sie das Layout im Eagle-Format auf der beiliegenden CD-ROM. Darauf finden Sie auch verschiedene Unterlagen zum Ausdrucken, z. B. eine Befehlsübersicht und Registerbeschreibungen. Für die Programmierung des Mikrocontrollers wird der *ICD2* von Microchip verwendet. Auf der CD-ROM finden Sie aber auch eine sehr einfache Schaltung, mit der der PIC über die serielle Schnittstelle programmiert werden kann.

Ich wünsche Ihnen viel Spaß und Erfolg bei der Mikrocontrollerprogrammierung. Für Kritik, Lob und Verbesserungsvorschläge bin ich immer offen und freue mich daher auch über eine Rückmeldung an meine E-Mail-Adresse info@edmh.de.

Michael Hofmann

Inhalt

5 Das Entwicklungs-Board

Ein Mikrocontroller ist immer eng mit der Hardware verbunden und die Software wird speziell für eine Hardwareschaltung programmiert. Programmiert man Software für einen PC, ist es nicht unbedingt notwendig zu wissen, welche Grafikkarte oder welcher Prozessor verwendet wird. Anders ist dies allerdings bei der Mikrocontrollerprogrammierung. Hier ist es sehr wichtig zu wissen, welches Display angeschlossen ist und an welchen Pin die Taster und LEDs liegen. Damit die Software auch auf echter Hardware getestet werden kann, wird im Folgenden eine Schaltung für ein Entwicklungs-Board vorgestellt, mit der alle Beispiele ausprobiert und erweitert werden können. Um den Schaltplan zum Arbeiten auszudrucken, findet man auf der CD-ROM das entsprechende Dokument im PDF-Format. Man findet außerdem den Schaltplan und das Layout im Eagle-Format. Der Schaltplan und das Layout wurden mit der Demoversion von Eagle erstellt, die man kostenfrei für private Zwecke von der Internetseite des Herstellers (www.cadsoft.de) herunterladen kann. Der Schaltplan und das Layout können so auf die eigenen Bedürfnisse angepasst und erweitert werden. Um den Aufwand für die Erstellung eines eigenen Entwicklungs-Boards zu verringern, kann über die Homepage www.edmh.de eine unbestückte Leiterplatte bestellt werden.

5.1 Schaltungsbeschreibung der Hardware

Das Entwicklungs-Board besteht aus verschiedenen Schaltungsteilen, die mit dem Mikrocontroller PIC16F876A verbunden sind. Es ist so konstruiert, dass möglichst viele Eigenschaften des Mikrocontrollers demonstriert werden können. Mit dem Entwicklungs-Board können Daten mit einem PC über die serielle Schnittstelle ausgetauscht, Daten in einem externen EEPROM über I²C abgespeichert, analoge Spannungen gemessen, Infrarotsignale analysiert und über ein Display Meldungen ausgegeben werden. Auf der Leiterplatte sind auch vier Taster und vier LEDs vorgesehen, über die man den Zustand des Mikrocontrollers beeinflussen oder anzeigen kann.

5.1.1 Netzteil

Der PIC16F876A benötigt eine Versorgungsspannung von 5 V. Damit diese immer konstant anliegt, findet man auf dem Entwicklungs-Board einen Spannungsregler. Das Entwicklungs-Board kann so auch über ein ungeregeltes Netzteil versorgt werden. Damit die Verlustleistung über dem Spannungsregler nicht zu groß wird, sollte man das Entwicklungs-Board mit einer Spannung zwischen 7 und 12 V versorgen. Die beiden Keramikkondensatoren *C10* und *C11* unterdrücken hochfrequente Störungen und der Elektrolytkondensator *C12* puffert die Ausgangsspannung.

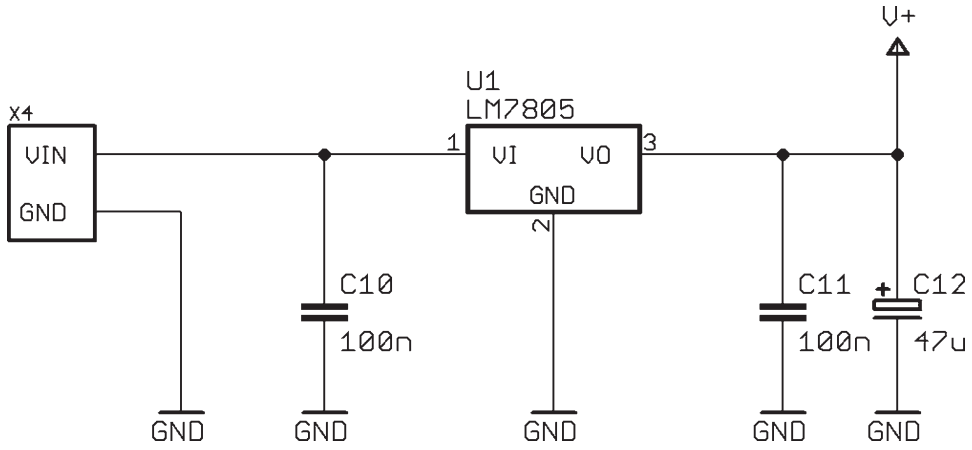


Abb. 5.1: Netzteil

5.1.2 Programmierschnittstelle

Damit die Programmierung möglichst universell erfolgen kann, wird der Mikrocontroller über eine fünfpolige Stiftleiste programmiert. Der Pin MCLR/VPP wird über einen Pull-up-Widerstand (*R9*) auf einem High-Pegel gehalten. Sollte sich das Programm, bei späteren Tests, einmal in einer Endlosschleife verfangen, kann man über den Jumper *J3* einen Reset auslösen.

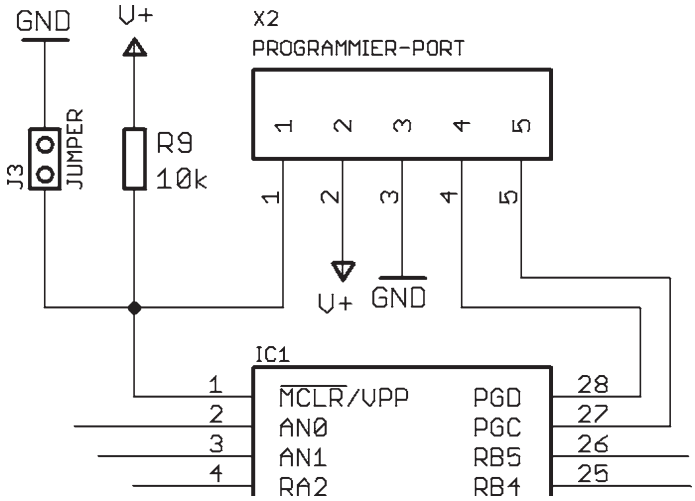


Abb. 5.2: Programmierschnittstelle

5.1.3 Taktgenerierung

Der Arbeitstakt des Prozessors wird über einen 4-MHz-Quarz erzeugt. Damit er anschwingt, sind zwei Keramikkondensatoren (*C8* und *C9*) am Quarz erforderlich.

7 Die Timer

Ein wichtiger Bestandteil eines Mikrocontrollers sind die Timer. Timer sind getaktete Zähler, die durch einen Takt hoch- oder heruntergezählt werden. Timer können für die unterschiedlichsten Zwecke verwendet werden. So wird z. B. für den Watchdog ein Timer benötigt oder man kann einen Timer für eine Warteschleife verwenden. Es können Zeiten zwischen zwei Impulsen bestimmt oder es können Impulse mit einer vorgegebenen Dauer ausgegeben werden. Jeder Timer verfügt über ein Register, in dem ein Wert gespeichert wird, der dann aufwärts oder abwärts gezählt wird. Bei einem Über- oder Unterlauf dieses Registers kann dann ein Interrupt ausgelöst werden. Einen Timer könnte man sich auch selbst programmieren, indem man ein beliebiges Register mit einem Wert lädt und dann in einer Schleife so lange herunterzählt, bis der Wert 0 ist. Dies ist eine einfache Methode, kurze Verzögerungszeiten in einem Programm zu implementieren. Bei dem folgenden Beispiel wird Pin RA2 für die Dauer von ca. 100 µs auf einen High-Pegel gesetzt. Für den Fall, dass der Prozessor mit einem Takt von 4 MHz versorgt wird, muss man in das Register *COUNTER* den dezimalen Wert 32 laden. Das ist ungefähr 100/3, da der Befehl *decfsz* 1 Befehlszyklus (= 1 µs) und der Befehl *goto* 2 Befehlszyklen (= 2 µs) benötigt.

Beispiel:

```
bsf PORTA, 2        ;setze Pin RA2 auf High-Pegel
movlw D‘32‘         ;lade das Register Counter mit 32
movwf COUNTER
decfsz COUNTER, F   ;verringere den Wert um 1
goto $-1            ;zähle so lange herunter, bis der Wert 0 ist
bcf PORTA, 2        ;setze Pin RA2 auf Low-Pegel
```

Sehr kurze Verzögerungen von wenigen Mikrosekunden lassen sich sehr gut über mehrere *nop*-Befehle realisieren. Ein *nop* benötigt für die Bearbeitung einen Befehlszyklus und entspricht bei einem 4-MHz-Takt einer Bearbeitungszeit von einer Mikrosekunde. Mit diesem Beispiel kann eine Zeitverzögerung bis ca. 765 µs (= 3 * 255) ohne den Einsatz eines Timers problemlos verwirklicht werden. Wenn eine Zeit größer als 765 µs implementiert werden soll, müsste man auf diese Weise schon zwei Register verwenden.

7.1 Der 8-Bit-Timer (Timer0)

Um längere Zeiten zu verwirklichen, ist es sinnvoll, einen Timer zu verwenden. Der große Vorteil eines Timers ist, dass der Takt, mit dem das Timer-Register hoch- oder

9 Anzeige von Daten auf einem Display

In vielen Fällen sollen die Daten im Innern des Mikrocontrollers dem Benutzer des Geräts mitgeteilt werden. In einfachen Fällen geschieht dies über eine Leuchtdiode, die z. B. das Überschreiten eines Schwellenwerts anzeigt. Will man aber wissen, wie weit der Schwellenwert überschritten wurde oder wie hoch der aktuelle Wert ist, ist die Darstellung mittels LEDs sehr aufwendig oder unpraktisch. Durch ein Display kann man dem Benutzer auf verschiedene Arten die Ergebnisse der Berechnungen mitteilen. Mithilfe der Anzeige auf dem Display ist es möglich, die verschiedensten Zeichen darzustellen. Es gibt Displays in unterschiedlichen Größen für die unterschiedlichsten Anwendungen – angefangen bei den kleinen Displays, die ca. 8 Zeichen darstellen können, bis hin zu komplexen Grafikdisplays, die beliebige Bilder und Schriften in Farbe darstellen können. Da die farbigen Grafikdisplays in der Regel verwendet werden, um komplexe Zusammenhänge darzustellen, benötigt man hierzu auch größere Mikrocontroller mit ausreichend Speicher. Die kleinen PIC-Mikrocontroller sind nicht für die Ansteuerung solcher Farbdisplays geeignet und werden daher sinnvollerweise auch nur für die Ansteuerung von einfachen Displays verwendet. Gebräuchlich ist ein Display mit einer Größe von 2 x 16 Zeichen. Das klingt nach „wenig“, aber man wird sich wundern, wie viele Informationen man damit darstellen kann. Im Folgenden werden anhand eines Displays die prinzipielle Funktionsweise und die Ansteuerung erklärt. Die Ansteuerung eines anderen Displaytyps erfolgt auf ähnliche Art und Weise und kann mithilfe der Beispiele problemlos auf verschiedene Displays umgesetzt werden.

9.1 Der Displaycontroller

Das Display auf dem vorgestellten Entwicklungs-Board ist von der Firma Electronic Assembly aus der EA-DOG-Reihe. Die Wahl fiel auf diese Firma, da man die Möglichkeit hat, das Display getrennt von der Hintergrundbeleuchtung zu kaufen. Das hat den Vorteil, dass man die Darstellungsfarbe des Textes nach seinen Wünschen sehr flexibel anpassen kann. Außerdem gibt es zu diesen Displays eine Erklärung in Deutsch. Das Display verfügt über 2 Zeilen mit je 16 Zeichen. Jedes Zeichen wiederum besteht aus 5 Punkten in horizontaler Richtung und 8 Punkten in vertikaler Richtung. In Summe verfügt das Display über 2 * 16 * 5 * 8 = 1.280 Punkte. Diese Punkte, auch *Pixel* genannt, müssen alle einzeln ansteuerbar sein. Um jedes Pixel einzeln anzusteuern, sind also 1.280 Leitungen nötig. Der PIC verfügt aber nur über 22 I/O-Pins, die noch nicht einmal für ein Zeichen ausreichen würden. Um nun dennoch ein solches Display

11 Messung des Widerstands und der Leistung

Im vorherigen Beispiel wurde die Spannung an einem analogen Eingang gemessen und ihr Wert auf dem Display dargestellt. Bei vielen Anwendungen in der Elektrotechnik soll nicht nur eine Spannung angezeigt werden, sondern auch der Strom durch einen Verbraucher. Das folgende Beispiel zeigt, wie die Spannung und der Strom eines Verbrauchers bestimmt werden und wie man aus den beiden Werten den Widerstand (R = U / I) und die Leistung (P = U * I) berechnet.

11.1 Die Strommessung

Da der interne A/D-Wandler des Mikrocontrollers keine Ströme messen kann, muss man die Signale über analoge Schaltungsteile in eine verwendbare Form bringen. Um den Strom durch einen Verbraucher zu messen, wird in der Regel ein Widerstand, der viel kleiner als der Widerstand des Verbrauchers ist, in Serie geschaltet und die daran abfallende Spannung gemessen. In *Abb. 11.1* ist eine prinzipielle Schaltung dargestellt, die zeigt, wie der Strom eines Verbrauchers gemessen werden kann. Die Schaltung soll nur als Beispiel dienen und ist in der Praxis nur bedingt empfehlenswert, da der Fehler bei kleinen Widerständen relativ hoch ist.

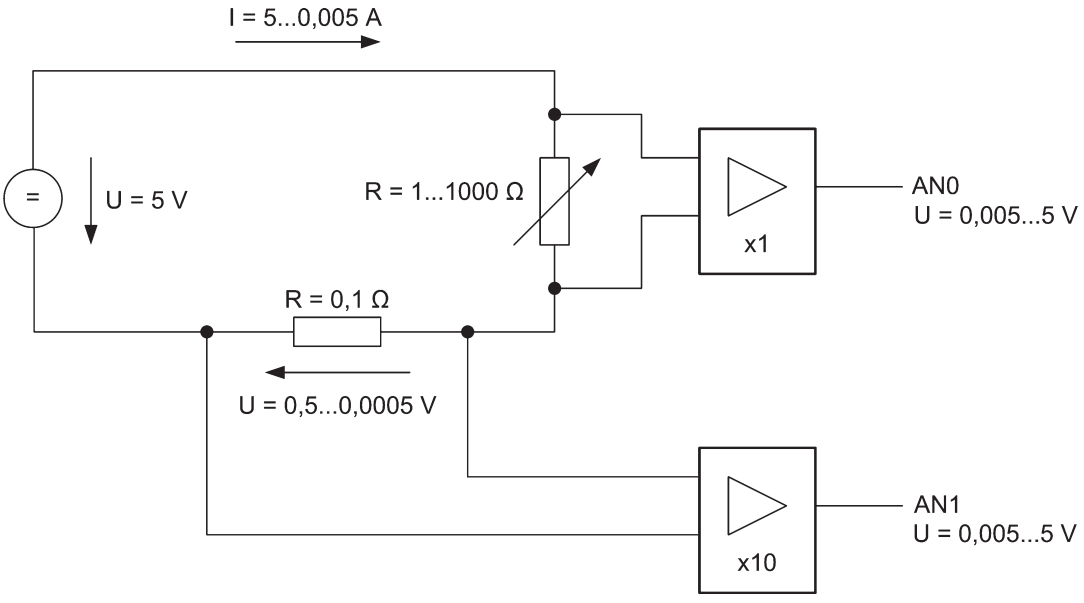


Abb. 11.1: Strommessung

12 Datenübertragung über die serielle Schnittstelle (RS-232)

In den vorherigen Beispielen wurde der Mikrocontroller immer als eigenständiges Bauteil betrieben, das dem Benutzer Informationen nur über ein Display oder Leuchtdioden mitteilt. Häufig möchte man den Mikrocontroller aber auch mit einem PC verbinden und Daten austauschen. Dies kann z. B. der Fall sein, wenn der Mikrocontroller Messdaten sammelt und diese dann zu einem PC übertragen werden sollen, um sie dort mit der höheren Rechenleistung bearbeiten zu können. Aufgrund der deutlich höheren Rechenleistung des Computers kann man die gewonnenen Messdaten einfach in Graphen auf dem Monitor darstellen. Ein anderes Beispiel, bei dem der PC Daten zum Mikrocontroller übertragen muss, ist die Steuerung einer externen Hardware. Man programmiert auf dem PC eine grafische Benutzeroberfläche, mit der man das externe Gerät steuern will, und verbindet die Hardware mit dem PC über die serielle Schnittstelle. Jetzt hat man die Möglichkeit, z. B. eine Temperaturregelung über den PC fernzusteuern. Dem Mikrocontroller wird nur noch mitgeteilt, wie die gewünschte Temperatur sein soll. Er übernimmt dann selbstständig die Regelung.

Es gibt verschiedene serielle Schnittstellen. Sie unterscheiden sich darin, wie sie die Daten übertragen. Die bekanntesten Schnittstellen, die standardmäßig am PC vorhanden sind, sind die USB- und die RS-232-Schnittstelle. Die USB-Schnittstelle ist mittlerweile kaum noch wegzudenken, da man hier hohe Übertragungsraten erzielen kann. Die RS-232-Schnittstelle findet man meist nur noch bei Desktop-PCs und sehr wenigen Laptops. Leider ist die USB-Schnittstelle relativ komplex in der Ansteuerung und daher für viele kleine Mikrocontroller, wie auch den PIC16F876A, zu aufwendig. Das Protokoll und die Ansteuerung der RS-232-Schnittstelle sind sehr einfach und unkompliziert und können so auch mit einem Mikrocontroller leicht realisiert werden. Meist reicht die geringe Übertragungsgeschwindigkeit der RS-232-Schnittstelle vollkommen aus, da nur wenige Daten übertragen werden sollen. Typische Geschwindigkeiten der RS-232-Schnittstelle liegen zwischen ca. 10 und 100 kBaud. Es geht auch noch ein wenig schneller, jedoch ist dann bei manchen PCs eine zusätzliche Einsteckkarte erforderlich. Will man eine Kommunikation zwischen einem Mikrocontroller und einem Laptop herstellen, der über keine RS-232-Schnittstelle verfügt, hat man die Möglichkeit, einen Adapter an einen USB-Port anzuschließen. Die Elektronik in diesem Adapter stellt dann dem Rechner einen sogenannten *virtuellen COM-Port* zur Verfügung und regelt die Kommunikation über die serielle RS-232-Schnittstelle. Für den Benutzer sieht dann alles so aus, als würde der PC über eine RS-232-Schnittstelle verfügen.

13 Datenübertragung über den I²C-Bus

Die Abkürzung *I²C* steht für *Inter-Integrated Circuit* und bedeutet, dass dieser Bus für die Kommunikation zwischen verschiedenen Bauteilen (ICs) verwendet wird. Die Datenübertragung erfolgt über lediglich zwei Leitungen: eine Datenleitung (SDA) und eine Taktleitung (SCL). Häufig findet man auch die Bezeichnung *TWI* für *Two-Wire-Interface* (Zweidrahtverbindung). Die Schnittstelle wurde von Philips Semiconductors (jetzt NXP Semiconductors) entwickelt, hat mittlerweile eine große Verbreitung gefunden und wird von vielen ICs unterschiedlicher Hersteller unterstützt. Die Daten werden als 8-Bit-Werte mit einer Geschwindigkeit von bis zu 3,4 Mbit/s übertragen. Die Standardgeschwindigkeit von I²C liegt bei 100 kbit/s. Diese Geschwindigkeit ist in der Regel für die Übertragung weniger Steuerregister im IC ausreichend. Daher wird die Schnittstelle sehr häufig verwendet, um die Register in den Bausteinen mit den entsprechenden Daten zu beschreiben. Da die Möglichkeiten der I²C-Schnittstelle umfangreich und vielseitig sind, werden in diesem Buch nur das Prinzip und die einfache Datenübertragung über die Schnittstelle erklärt. Die ausführliche Spezifikation des I²C-Buses findet man auf der Internetseite von NXP Semiconductors oder auf der beigelegten CD-ROM.

13.1 Funktionsweise der I²C-Schnittstelle

Jedes IC, das an den I²C-Bus angeschlossen wird, hat eine eindeutige Adresse (Device Address). In der Regel ist der Mikrocontroller der Bus-Master, der die Kommunikation regelt. Der Master spricht dann über die Adresse die angeschlossenen ICs (Slaves = Sklaven) an. Nach der Startsequenz überträgt der Master die Slave-Adresse des anzusprechenden Gerätes und teilt dem Gerät mit, ob Daten an das Gerät gesendet oder aus dessen internen Speicher gelesen werden sollen. Dies erfolgt über das letzte Bit der Adresse. Soll vom Slave gelesen werden, wird das letzte Bit auf High-Pegel gesetzt. Sollen Daten an den Slave übertragen werden, wird das Bit auf Low-Pegel gelegt. Erkennt ein Slave anhand der Adresse, dass er angesprochen wurde, bestätigt er den korrekten Empfang, indem er das nächste Bit (Acknowledge) auf 0 setzt. Danach wird in den meisten Fällen die interne Adresse des anzusprechenden Registers im IC übertragen. Im Anschluss daran werden die Daten, die in das Register geschrieben oder aus dem Register ausgelesen werden sollen, übertragen. In *Abb. 13.1* ist ein Beispiel einer Übertragungssequenz dargestellt.

14 Schalten über eine Infrarot-Fernbedienung

Jedes Fernsehgerät und jeder Satellitenempfänger wird mit einer Infrarot-Fernbedienung gesteuert. Teilweise ist eine Programmumschaltung ohne Fernbedienung nicht mehr möglich. Eine Fernsteuerung hat in der Regel zwischen 20 und 30 Tasten, wodurch man sich diese Anzahl der Tasten an dem zu steuernden Gerät sparen kann. Um sich diese Möglichkeit der Fernsteuerung zunutze zu machen, wird in diesem Kapitel ein Beispiel vorgestellt, mit dessen Hilfe das Infrarotprotokoll (IR-Protokoll) ausgewertet werden kann. Im Beispiel wird abhängig vom Tastendruck eine LED geschaltet. Würde man anstelle der LED ein Relais schalten lassen, wäre es möglich, einen Verbraucher mit 230 V Netzspannung (z. B. eine Lampe) zu schalten. Auf diese Weise kann man relativ einfach eine Lampe fernsteuern. Um Energie zu sparen, kann man diese Schaltung auch vor eine Steckdosenleiste schalten und so mehrere Verbraucher (z. B. Fernseher, Receiver, CD-ROM-Player etc.) mit einem Tastendruck auf der Fernbedienung vom Netz trennen. Dadurch verbrauchen die angeschlossenen Geräte keinen Strom mehr und die Kosten werden reduziert.

Leider konnten sich die Hersteller von IR-Fernbedienungen nicht auf einen einheitlichen Standard für die Übertragung einigen und so gibt es nun viele verschiedene Codes, mit denen die Geräte fernbedient werden. In diesem Buch wird daher ein Code verwendet, der verhältnismäßig weit verbreitet ist und gern von Bastlern für die Steuerung eigener Entwicklungen verwendet wird. Es handelt sich dabei um den *RC5-Code*, der von der Firma Philips entwickelt wurde. Sollte man in seiner Fernbedienungssammlung keine Fernbedienung mit einem RC5-Code finden, kann man sich für ca. 10 € im Fachhandel eine Universalfernbedienung kaufen, mit der nahezu jeder Code gesendet werden kann. Der Code wird in der Regel über eine dreistellige Ziffer in die Fernbedienung programmiert. Um den richtigen Code zu finden, sollte man sich das fertige Beispiel in den Mikrocontroller laden. Dann kann man einen automatischen Suchlauf auf der Universalfernbedienung starten. Wenn ein RC5-Code gesendet wird, schaltet eine LED um und man kann diesen Code für die weitere Steuerung verwenden.

14.1 Das RC5-Protokoll

Um eine möglichst stabile Übertragung zu gewährleisten, werden bei dem RC5-Protokoll nicht einfach nur High- und Low-Pegel übertragen, sondern es wird für jedes Bit eine Signaländerung gesendet. Ebenfalls wird das Infrarotsignal noch mit einer Frequenz von

Sachverzeichnis