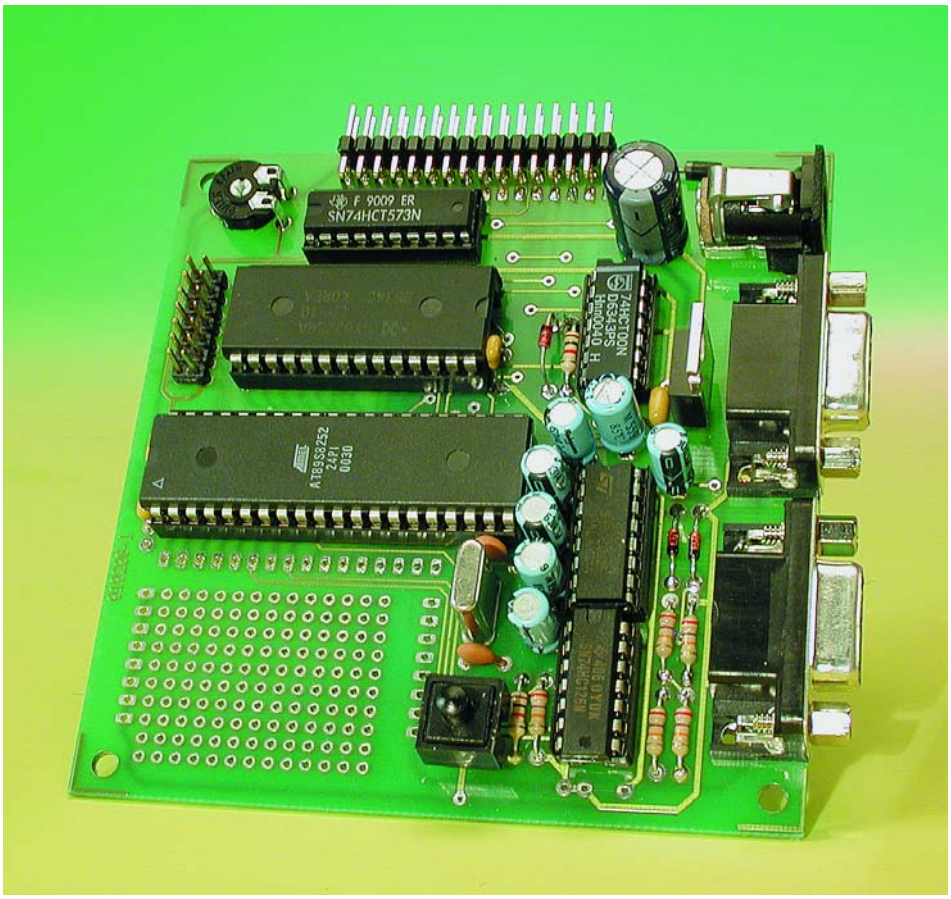
Edukative Hard- und Software

89S8252 Flash-Board

Basis für Mikrocontroller-Kurs

Von B. Kainka

Das Mikrocontroller-Board als Grundlage für einen Programmierkurs muss einfach handzuhaben sein und dennoch alle Ingredenzien eines modernen Mikrocontrollers mit Peripherie aufweisen. Dies macht es auch für alte Controllerhasen interessant!



Wer eigene Ideen in Schaltungen umsetzen will, kommt mit einem Mikrocontroller meist schneller zum Ziel als mit konventioneller Elektronik. Allerdings ist bei vielen Systemen die Programmierung relativ aufwendig. Deshalb hat Elektor das neue Flash-Board entwickelt. Der Flash-Programmspeicher und ein sorgfältiges Design des Systems garantieren eine einfache und schnelle Programmierung.

Die Vorüberlegungen zu einem neuen Mikrocontroller-Board gingen von den folgenden Zielen aus: Es sollte eine Schaltung entwickelt werden, die sowohl dem erfahrenen Anwender wie auch dem Einsteiger entgegenkommt. Das System sollte von vornherein mit unterschiedlichen Programmiersprachen von Assembler über Basic bis zu C klarkommen. Es sollte weiterhin leicht erweiterbar und einfach zu programmieren sein. Außerdem sollte ein einmal auf das Board übertragenes Programm nach dem Ausschalten erhalten bleiben. Denn damit entfällt der mühsame Weg

über das Brennen eines EPROMs. Das Board selbst sollte deshalb zugleich die Funktion eines Programmiergeräts mit übernehmen.
Es gibt viele Mikrocontroller mit internem EEPROM- oder Flash-Speicher, die prinzipiell eine einfache Programmierbarkeit versprechen. Sollte es ein PIC, ein Controller aus der 68er-Familie, ein AVR oder ein erweiterter 8051 sein? Die schwierige Frage nach dem Controllertyp wurde zugunsten der 8051er-Familie entschieden. Dafür gab es gute Gründe: Die meisten anderen Controller werden nur von einer Firma gefertigt, was im Ernstfall zu Lieferproblemen führen kann. Die 8051-Familie dagegen kann mit Recht als Industriestandard bezeichnet werden, denn sehr viele Hersteller fertigen Derivate des ursprünglich von Intel entwickelten Standards. Wenn also ein Gerät entwickelt wurde und Firma A kann eines Tages nicht liefern, dann kann man leicht auf einen Controller der Firma B oder der Firma C ausweichen. Für die 8051-Familie gibt es außerdem ein umfangreiches Software-Angebot und viel Literatur. Wer also auf den 8051-Standard setzt, kann nicht viel falsch machen. Und das gilt auch heute noch, obwohl es inzwischen neuere Konzepte gibt.
Der verwendete AT89S8252 (**Bild** 1) von Atmel entspricht in wesentlichen Teilen dem 8052 von Intel. Das bedeutet zum Beispiel, dass das bekannte BASIC-52 problemlos auf dem Chip eingesetzt werden kann. Einige zusätzliche Merkmale erfreuen den Entwickler. So gibt es einen internen Watchdog und ein Daten-EEPROM mit einer Größe von 2 K. Das Entscheidende aber ist der 8 K große Programmspeicher in Flash-Technologie. Das Programmiergerät für diesen Speicher befindet sich ebenfalls auf dem Chip. Man benötigt nicht einmal eine besondere Programmierspannung, die normale Spannung von 5 V reicht aus. Die Programmierung erfolgt seriell und getaktet über eine SPI-Schnittstelle. Nur wenige Leitungen werden benötigt, um ein Programm in den internen Flash-Bereich zu laden.

Flash-Board mit Mikrocontroller 89S8252

- 32 K RAM
- 8 K Programm-Flash
- 2 K Daten EEPROM
- interner Watchdog-Timer
- Serielle Schnittstelle
- Programmierschnittstelle
- Betriebsspannung 9...12V, 5 V Spannungsregler
- LCD-Schnittstelle
- Busanschlüsse an Pfostensteckern
- Portanschlüsse an Pfostensteckern
- Experimentierfeld für Erweiterungen

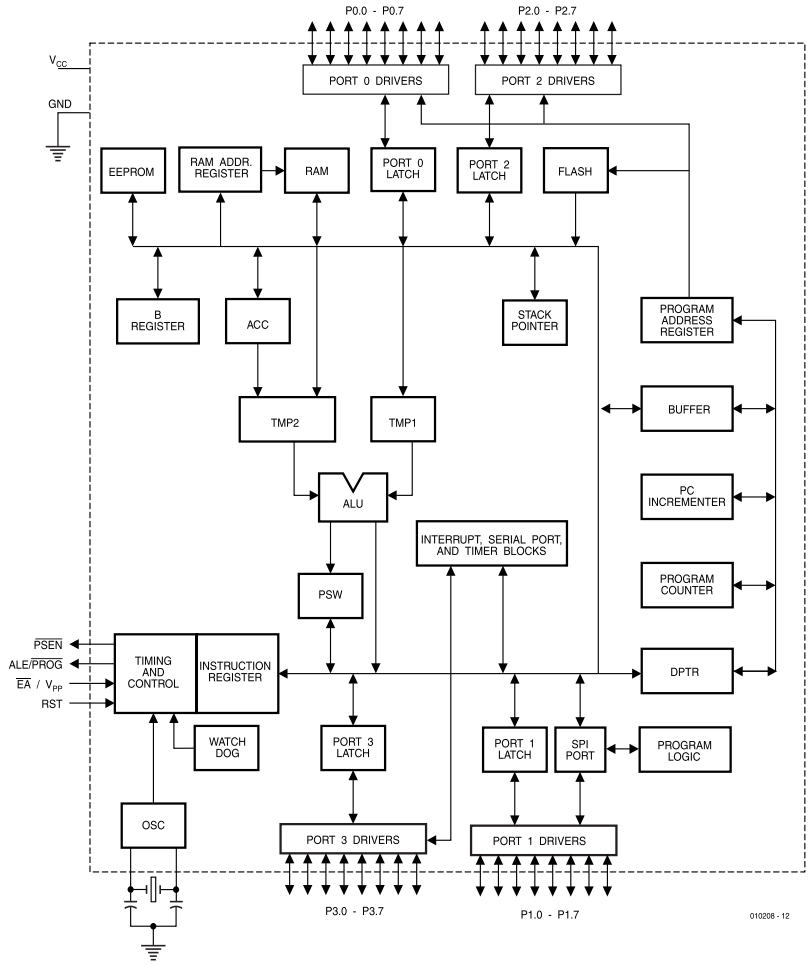


Bild 1. Blockschaltbild des Controllers AT89S8252

Das Board

Das Flash-Board erweitert den Controller um 32 K RAM, eine LCD-Schnittstelle, eine serielle und eine Programmierschnittstelle. Außerdem befindet sich auf dem Board ein Experimentierfeld mit 130 Lötpunkten. Kleine Erweiterungen vom Motortreiber über Sensoren bis zu zusätzlichen Schnittstellen können also leicht untergebracht werden. Der gesamte Daten- und Adressbus und alle Ports sind über Steckerleisten zugänglich. Deshalb lässt sich das System fast beliebig erweitern. Die Platine besitzt zwei DB9-Buchsen. Über eine findet die Kommunikation mit der seriellen Schnittstelle statt. Wie jeder Mikrocontroller aus der 8051-Familie besitzt auch der 89S8252 einen internen UART, der über einen Schnittstellentreiber MAX232 beispielsweise mit einer COM-Schnittstelle des PCs verbunden wird. Ob der PC hier Steuerkommandos an den Controller überträgt, Messdaten abfragt oder mit dem internen BASIC-Interpreter kommuniziert, hängt ganz vom Anwender und der eingesetzten Software ab. Die zweite DB9-Buchse (COM2) dient ganz allein der Programmierung des Controllers. Ein angeschlossener PC kann den Mikrocontroller zurücksetzen, in den Programmiermodus schalten, ein Programm übertragen und starten. Des ganze Board ist also praktisch fernsteuerbar. Es könnte sich irgendwo an einer unzugänglichen Stelle

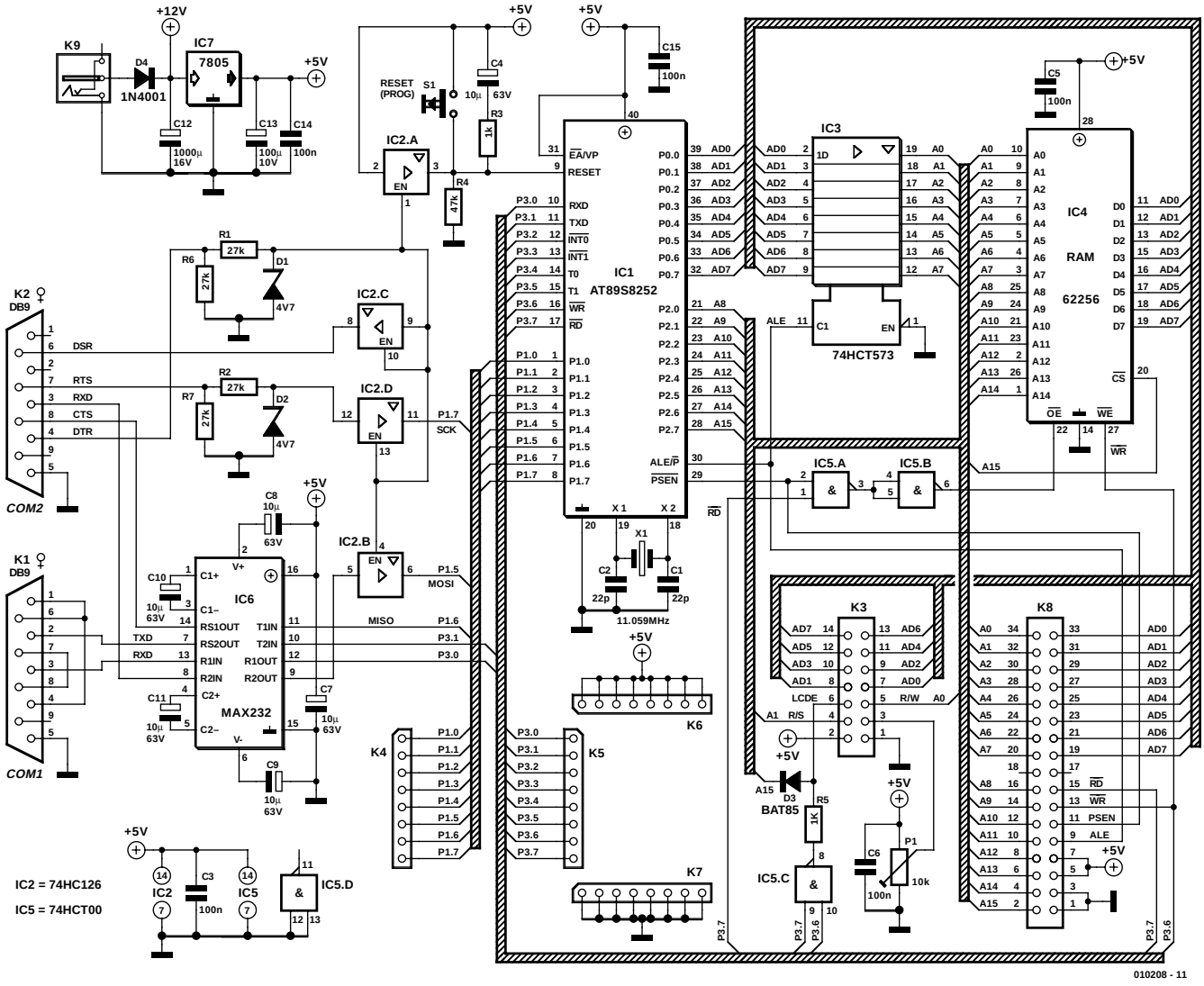


Bild 2. Ein klassisches, gut ausgebautes Mikrocontrollersystem

befinden und trotzdem problemlos neu programmiert werden.

Die erste DB9-Buchse COM1 wird mit einer seriellen Schnittstelle des PCs verbunden. Der Anwender schreibt und kompiliert dann sein Programm, überträgt es an den Controller und testet es auf dem Board. Nur wenn das übertragene Programm nach einer Kommunikation mit dem PC ruft, muss auch K1 verwendet werden. Dabei hat man zwei Möglichkeiten. Entweder steckt man das Anschlusskabel um oder verwendet zwei COM-Schnittstellen des PCs. Dies ist an vielen PCs möglich, wenn zwei COM-Schnittstelle vorhanden sind und die Maus einen PS2-Anschluss belegt oder am USB arbeitet. Die parallele Verwendung von zwei COM-Schnittstellen erweist sich oft als besonders praktisch und komfortabel. Programmieren und Testen gehen Hand in Hand, wenn die benötigten Tools parallel eingesetzt werden.

Die Schaltung in **Bild 2** orientiert sich in weiten Teilen am üblichen Entwurf für 8051er-Systeme. Ein Adresslatch 74HCT573 (IC3) liefert die unteren acht Adressen für das 32 K RAM (IC4). Da die $\overline{CS}$-Leitung des RAMs mit der Adressleitung A15 verbunden ist, belegt der Speicher nur den Bereich 0...7FFFh, der obere Bereich ab 8000h ist frei für Erweiterungen. Die $\overline{WR}$-Leitung für die Steuerung von Schreibzugriffen wird direkt vom Prozessor angesteuert. Die $\overline{OE}$-Leitung für Lesezugriffe jedoch erhält ein Signal, das mit Hilfe zweier Gatter des 74HCT00 (IC5) aus einer UND-Verknüfung des PSEN-Signals für Programmzugriffe und des WR-Signals für Datenzugriffe gebildet wird. Im Normalfall verwenden 8051-Controller parallel liegende Adressbereiche für Programme und Daten. Diese Trennung wird hier aufgehoben. Der Controller verwendet zwar meist ein Programm in internen FLASH-ROM mit einer Größe von 8 K und benutzt das RAM nur zum Ablegen von Daten. Es ist jedoch auch möglich, Programmteile in Bereichen des RAMs oberhalb 2000h zu verwenden. Da die selben Bereiche auch als Datenbereich zugänglich sind, können selbstmodifizierende Programme geschrieben werden. Der Prozessor schreibt seinen eigenen Kode ins RAM und führt ihn dann aus.

Das interne EEPROM des Prozessors ist übrigens ein reiner Datenspeicher, der parallel zum externen RAM liegt. Das EEPROM muss durch Software aktiviert werden. Dann hat man die Möglichkeit, hier dauerhaft

Daten abzulegen. Die unteren 2 K des externen RAMs sind in diesem Fall inaktiv.
Das LC-Display ist ebenfalls am Datenbus angeschlossen und belegt den Bereich oberhalb 8000h. Eine NAND-Verknüpfung von $\overline{RD}$ und $\overline{WR}$ liefert das Enable-Signal für das Display. Eine einfache Dioden-Logik sperrt dieses Signal im Adressbereich bis 7FFFh. Das Display belegt daher folgende Adressen:

8000h: Kommando schreiben
8001h: Kommando lesen
8002h: Daten schreiben
8003h: Daten lesen

Dieser Bereich spiegelt sich bis FFFFh. Man kann also ebenso den Bereich F000h bis F003h verwenden.
Die serielle Schnittstelle mit den Leitungen TXD und RXD ist wie üblich über einen Schnittstellenbaustein MAX232 mit dem Anschluss zum PC verbunden. Es wird kein Handshake verwendet. Interne Verbindungen einzelner Handshakeleitungen nehmen Rücksicht auf die Bedürfnisse einiger Terminalprogramme, die eine Signalisierung der Empfangsbereitschaft erwarten.
Der Aufbau des Flash-Boards ist kinderleicht und sollte in einer Stunde erledigt sein. Für alle ICs darf man (gute) Fassungen verwenden, so dass man später beim Bestücken (wie bei den Dioden und Elkos) nur auf die richtige Orientierung achten muss. Für K8 werden entweder vier „nackte“ achtpolige einreihige oder aber ein 2-reihiger 34-poliger Pfostenverbinder mit Wanne und Orientierungsnase außen verwendet. Die mittleren Pins sind dann nicht belegt (siehe **Tabelle**).

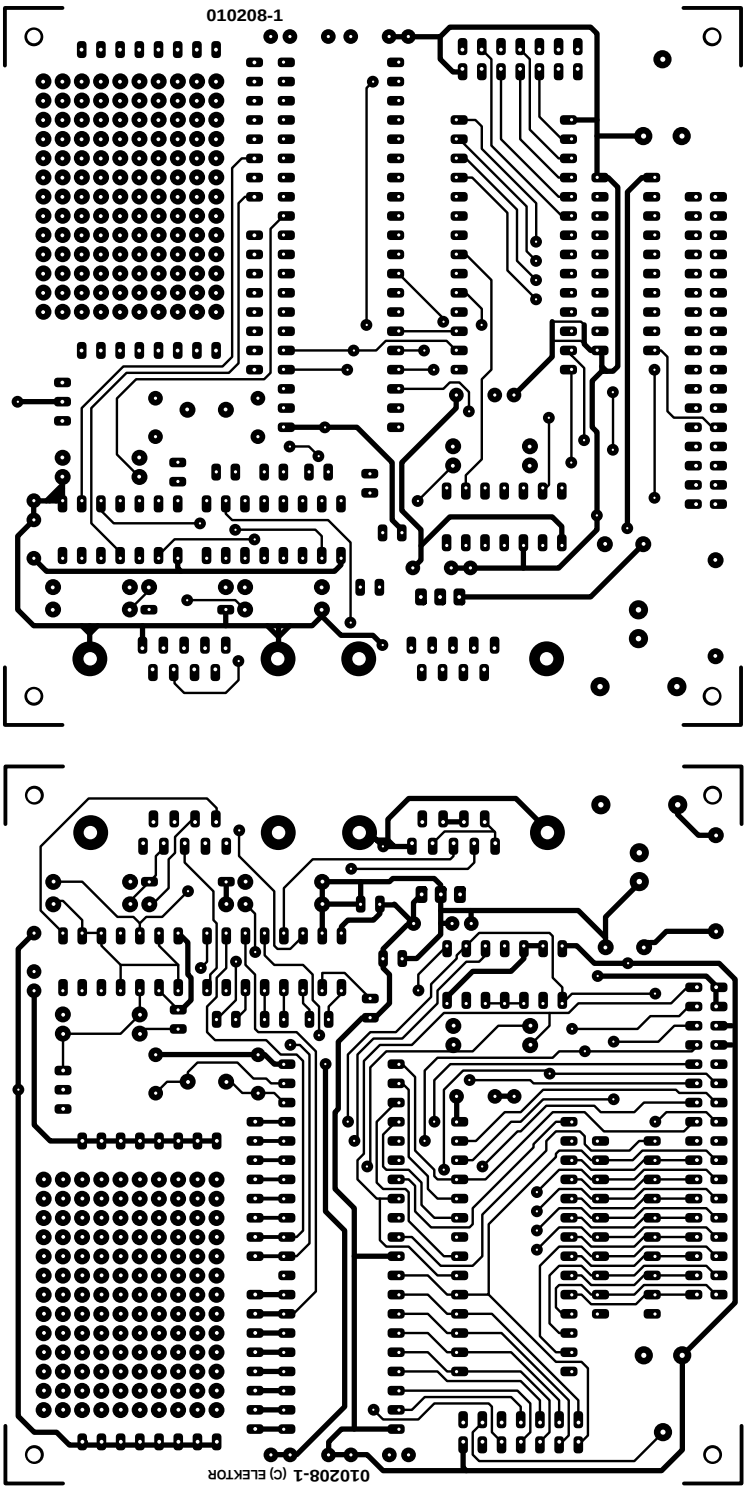


Bild 3a. Platinenlayout ...

Programmierung

Die Programmierschnittstelle des Controllers verwendet eine SPI-Schnittstelle mit folgenden Leitungen:

SCK (P1.7): Taktleitung
MISO (P1.6): Daten lesen
MOSI (P1.5): Daten schreiben
RST: Reset-Leitung

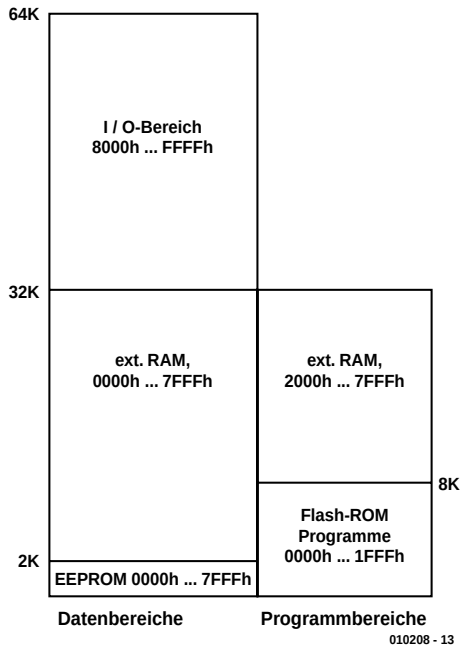


Bild 4. Interne und externe Speicher des Mikrocontrollerboards.

Die Programmierung erfolgt bei aktiver Reset-Leitung. Ein Programmiertool erzeugt also einen Reset-Zustand und schreibt dann Daten über MOSI, wobei jedes Bit über SCK in ein internes Schieberegister des Prozessors übertragen wird. In gleicher Weise kann über MISO der Erfolg der Programmierung überwacht oder ein Programm gelesen werden. Alle Signale werden direkt über RS232-Leitungen erzeugt, die Spannungspegel um ±10 V liefern. Daher ist eine Anpassung an die 5-V-Pegel des Prozessors notwendig. Die noch freie Hälfte des MAX232 übernimmt die Pegelanpassung der Datenleitungen MOSI und MISO. Das Clock- und das Reset-Signal dagegen werden mit einer Zenerdioden-Spannungsbegrenzung und Tristate-Buffern im 74HC126 (IC2) aufbereitet. Dabei wurde berücksichtigt, dass die betreffenden Portanschlüsse natürlich in einem eigenen Mikrocontroller-Programm eine andere Funktion ausüben können. Die Programmierschnittstelle wird deshalb völlig abgeschaltet, wenn sie nicht benötigt wird. Die Buffer befinden sich dann im hochohmigen Zustand und sind sozusagen gar nicht da. Erst wenn über die DTR-Leitung ein Prozessor-Reset ausgeführt wird, werden auch die übrigen Buffer aktiv. Dies geschieht übrigens nicht, wenn ein Reset über den Taster auf dem Board ausgelöst wird. Parallel zum Reset-Taster arbeitet auch eine Pufferstufe im 74HC126. Über DTR wird praktisch ein P-Kanal-FET mit offenem Drain gesteuert, der wie ein zweiter Reset-Taster arbeitet. Das Reset-Signal wird also immer nur aktiv

Anschlussbelegung K8

1	GROUND
2	A15
3	GROUND
4	A14
5	+5V
6	A13
7	+5V
8	A12
9	ALE
10	A11
11	PSEN
12	A10
13	$\overline{WR}$
14	A9
15	$\overline{RD}$
16	A8
17	NOT CONNECTED
18	NOT CONNECTED
19	AD7
20	A7
21	AD6
22	A6
23	AD5
24	A5
25	AD4
26	A4
27	AD3
28	A3
29	AD2
30	A2
31	AD1
32	A1
33	AD0
34	A0

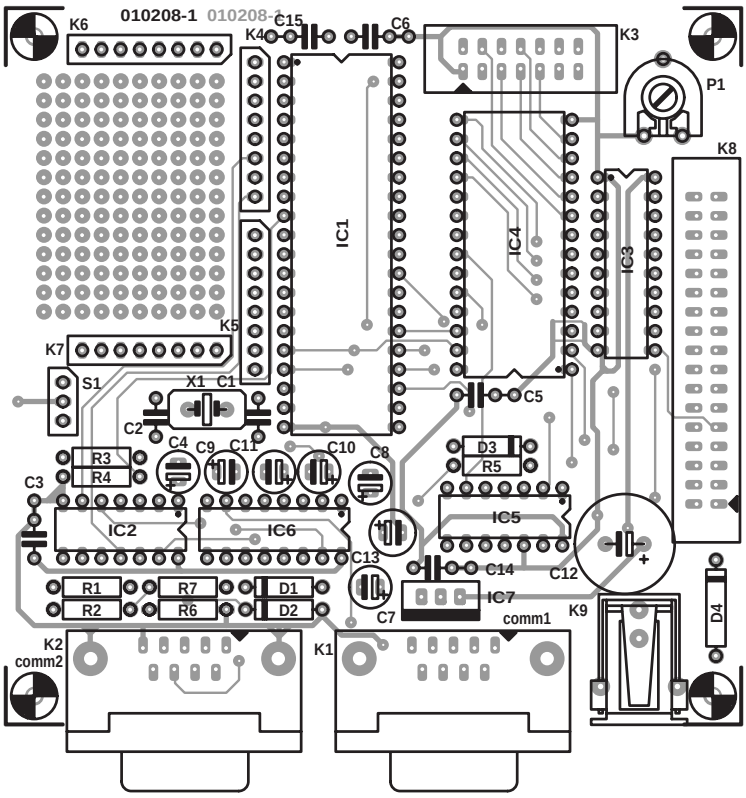


Bild 3b. ... und Bestückungsplan.

Stückliste

Widerstände:
R1,R2,R6,R7 = 27 k
R3,R5 = 1 k
R4 = 47 k
P1 = 10 k Trimmpoti

Kondensatoren:
C1,C2 = 22 p
C3,C5,C6,C14,C15 = 100 n keramisch
C4,C7...C11 = 10 µ/63 V stehend
C12 = 1000 µ/16 V stehend
C13 = 100 µ/10 V stehend

Halbleiter:
D1,D2 = Z-Diode, 4V7, 500 mW
D3 = BAT85
D4 = 1N4001
IC1 = AT89S8252 24PC
IC2 = HC126
IC3 = 74HCT573
IC4 = 62256 120 ns
IC5 = 74HCT00
IC6 = MAX232
IC7 = 7805

Außerdem:
K1,K2 = 9-polige Sub-D-Buchse, gewinkelt für Platinenmontage
K3 = 2·7-poliger Pfostenverbinder mit Wanne
K4...K7 = 1·8-poliger Pfostenverbinder, male
K8 = 2·17-poliger Pfostenverbinder mit Wanne
K9 = Buchse für Steckernetzteil
S1 = Drucktaster mit Arbeitskontakt
X1 = Quarz 11,0592 MHz
Platine EPS 010208-1
Diskette EPS 010208-11
(siehe Service-Seiten in der Heftmitte)

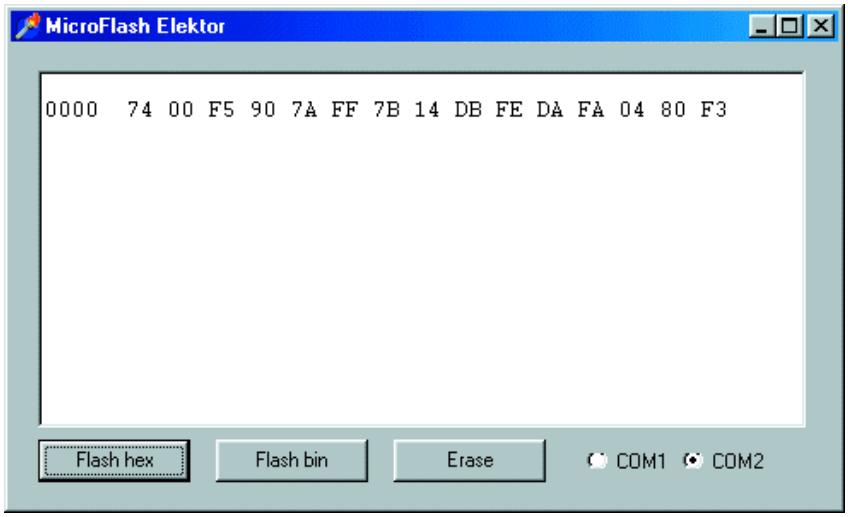


Bild 5. Das Downloadprogramm

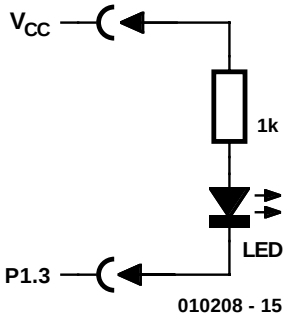
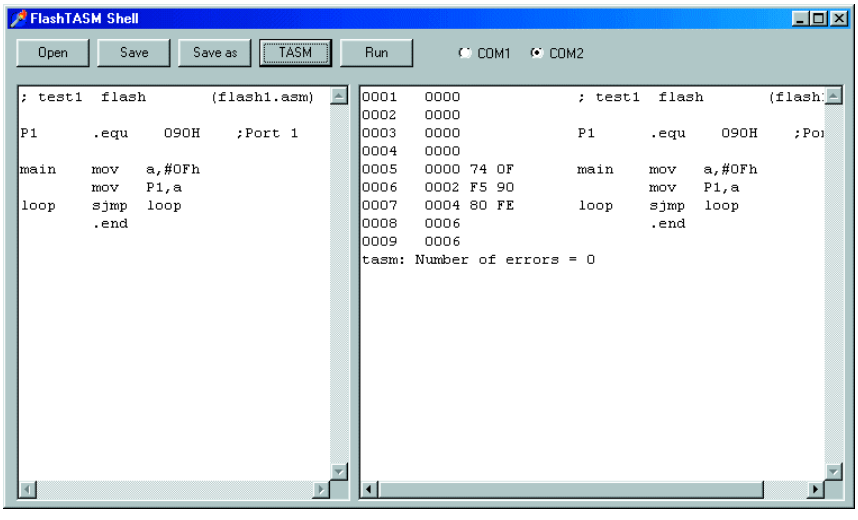


Bild 6. Der erste Test mit einer LED an einer Portleitung.

hochgezogen, nie jedoch aktiv auf Masse gelegt. Dies gelingt, weil der Eingang des Buffers permanent an Vcc liegt, während der Enable-Eingang von der Programmierschnittstelle gesteuert wird.

Die Programmierung des Controllers ist ein relativ komplexer Vorgang, bei dem zunächst bestimmte Steuerbytes übertragen werden müssen. Der Anwender muss sich jedoch nicht näher damit beschäftigen, weil ein fertiges Windows-Programmiertool vorliegt. Das Programm *MikroFlash.exe* befindet sich auf der EPS-Diskette 010208-11 und kann auch über das Internet von der Elektor-Homepage bezogen werden. Die Bedienung ist einfach. Man aktiviert die gewünschte Schnittstelle COM1 oder COM2 und wählt zunächst das Dateiformat (Intel-Hex oder Binär) aus. Es öffnet sich ein Dateimenü, in dem man eine passende Programmdatei auswählt. Nach der Dateiauswahl startet die Programmierung, wobei der Fortschritt im Textfenster beobachtet werden kann. Mit dem Ende der Übertragung schaltet das Programm den Controller automatisch wieder in den RUN-Modus und startet so das Programm.



Listing 1. Das Assembler-Testprogramm lässt die LED aufblinken.

Der erste Test

Ein erstes Programmbeispiel liegt auf der Diskette vor. Das Programm *Flashtest1.hex* oder *Flashtest1.bin* enthält eine kurze Programmierschleife mit aufsteigenden Portausgaben. Nach der Übertragung findet man Rechtecksignale an allen Port1-Pins. Hat man ein Oszilloskop zur Hand, lässt sich die korrekte Funktion leicht erkennen. Aber auch eine LED mit Vorwiderstand erfüllt ihren Zweck. Sie wird hier zwischen VCC und einen Port-1-Anschluss gelegt. Man erkennt ein langsames Blinken an P1.7 und jeweils die doppelte Frequenz an P1.6, P1.5 und so weiter. Das Beispiel wurde übrigens in Assembler geschrieben. **Listing 1** zeigt das erste Beispielprogramm *flash1.asm*.

```
; test1  flash
#include 8051.H
         .org 0000H

main    mov    a,#00

next    mov    P1,a
        mov    r2,#255
loop1   mov    r3,#20
loop2   djnz   r3,loop2
        djnz   r2,loop1
        inc    a
        sjmp   next
        .end
```

Wenn das Board erfolgreich getestet ist, steht der eigenen Programmentwicklung nichts mehr im Wege. Der Anwender kann dabei auf bereits vorhandene Entwicklungswerkzeuge zurückgreifen. Ganz egal, ob Programme in Assembler, Basic-52, Pascal oder C entwickelt werden sollen, das Board spielt mit.

Bei der Verwendung von Basic-52 werden übrigens sozusagen die Rollen vertauscht. Basic-Programme werden nämlich aus der Sicht des Interpreters als Daten behandelt. Der Interpreter selbst ist ein 8 K großes Programmfile, das direkt in den Controller geladen wird. Das Programm stammt aus dem legendären BASIC-52-AH und wurde von Intel freigegeben. Die Software kann daher ebenfalls mit der Programmdiskette beziehungsweise im Internet weitergegeben werden. Inzwischen gibt es ja auch die verbesserte Version Basic52-V1.3 (Elektor 2/2001), die ebenfalls einsetzbar ist. Bei der Übertra-

gung in den Controller sollte man nicht die Geduld verlieren, denn 8 KByte seriell zu übertragen und einzeln ins Flash-Rom zu brennen, dauert etwa eine Minuten. Wer möchte, kann das System dauerhaft als BASIC-Rechner benutzen. Allerdings bleiben die BASIC-Programme nach dem Ausschalten nicht erhalten, da sie im RAM liegen. Wer Software für den Dauereinsatz entwickeln möchte, sollte Assembler oder C verwenden. Nicht jede Anwendung erfordert ein RAM. Vor allem einfache Assemblerprogramme kommen oft ohne aus. Das Board kann dann zum Beispiel mit einer Teilbestückung nur mit IC1, IC2 und IC3 verwendet werden. Für eine fertige Anwendung lässt sich die Bestückung noch weiter reduzieren. Nun reicht allein der Mikrocontroller IC1, es handelt sich dann um eine echte Einchip-Anwendung. Man kann natürlich auch noch einen Schritt weiter gehen und den programmierten Chip aus der Fassung nehmen und in einer eigenen Platine einsetzen. Das Board ist dabei zunächst Entwicklungssystem und Programmiergerät zugleich. Ein Beispiel soll diese Vorgehensweise erläutern.

Es soll ein kleiner Roboter mit Mikrocontroller-Steuerung entwickelt werden. Zunächst baut man die erforderlichen Sensoren und Treiber auf das Experimentierfeld des Boards, entwickelt dann das Steuerprogramm, nimmt - sobald die Funktion zufriedenstellend ist - den Mikrocontroller aus der Fassung und baut eine eigene, möglichst kleine Steuerplatine für den Roboter. Der Controller benötigt dabei nicht viel mehr als einen Quarz und ein paar Kondensatoren. Und während dann der Roboter vollautomatisch seine Kreise auf dem Teppich zieht, steckt in der Entwicklungsplatine schon wieder ein neuer Prozessor für eine neue Aufgabe. Vielleicht wird es diesmal ein spezielles Messgerät für das Elektroniklabor.

(010208-1)rg

Vorschau
Basiskurs Mikrocontroller

Immer wieder wachsen neue Generationen technisch Interessierter heran. Aber auch der bereits erfahrene Elektronik-Freund lernt nie aus. In letzter Zeit gab es häufig Anfragen bei Elektor: Wie gelingt ein einfacher Einstieg in die Mikrocontroller-Technik? Unsere Antwort ist: mit dem neuen ***Basiskurs Mikrocontroller***, der im Januarheft 2002 beginnt und voraussichtlich sechs Folgen haben wird.

Zwar gibt es bereits viele Bücher zum Thema und das Internet ist voll von konkreten Anwendungen. Aber meist werden schon Grundkenntnisse vorausgesetzt. Für den Einsteiger ist es schwierig, eine Orientierung zu finden. Welchen Controller soll man nehmen, welche Software einsetzen? Wo liegen die Grenzen bestimmter Systeme, welche Programmiersprache eignet sich für welche Aufgabe? Der Basiskurs soll Licht in den Dschungel der Mikrocontoller-Technik bringen.

Der Kurs verwendet das in diesem Heft vorgestellte Flash-System. Damit ergeben sich viele Möglichkeiten für den Einsatz unterschiedlicher Programmiersprachen. Der Kurs wird Assembler, Basic und C verwenden. Damit kann jeder die spezifischen Stärken und Schwächen dieser Sprachen vergleichen. Aber es soll nicht nur eine Einführung in die Programmiersprachen werden, sondern es stehen typische Anwendungsfälle für Mikrocontroller im Mittelpunkt. Verfahren und Techniken im Bereich Messen, Steuern und Regeln werden ebenso vermittelt wie die Besonderheiten des verwendeten Controllers und ganz allgemein der 8051-Familie.

Welche Vorteile bringt die Teilnahme an diesem Kurs?

Hier wird versucht, den möglichen Nutzen in Stichworten zusammenzufassen.

Der Anwender erhält eine gründliche Einführung in allgemeine Techniken der Mikrocontroller-Anwendung, von der Speichernutzung über die Verwendung von Ports und Schnittstellen bis zur Ansteuerung komplexer Peripheriebausteine. Ein gründlicher Vergleich unterschiedlicher Programmiersprachen vermittelt Entscheidungssicherheit bei der Wahl geeigneter Werkzeuge. Die im Laufe des Kurses entwickelten Programme lassen sich auf fast beliebigen 8051-Systemen einsetzen. Der Kursteilnehmer wird zunehmend in die Lage versetzt, eigene Ideen mit einem Mikrocontroller umzusetzen.

Es werden folgende Programmiersprachen verwendet:

– Assembler (genauer: der bekannte Sharewareassembler TASM mit einer aufgesetzten Windows-Oberfläche)
– Basic-52 (auch die neue Version 1.3 mit einem Windows-Editor- und Terminalprogramm)
– Der freie C-Compiler Reads51 der Firma Rigel, der ebenfalls unter Windows läuft

Wer jetzt neugierig geworden ist, kann die Software READS51 direkt aus dem Internet laden: (http://www.rigelcorp.com/). Die Firma Rigel Corporation, Gainesville, FL, USA stellt Mikrocontroller-Produkte unter anderem für den Ausbildungsbereich her, die mit freier Software für nicht-kommerzielle Anwendungen unterstützt werden. Elektor hat freundlicherweise die Erlaubnis bekommen, die Software auch für den Basiskurs Mikrocontroller einzusetzen. Empfehlenswert ist auch das Basic-Terminalprogramm RbHost und die umfangreichen Informationen auf der Homepage von Rigel.

Der Basiskurs Mikrocontroller soll diese drei Programmiersprachen vergleichend verwenden, gleiche oder ähnliche Aufgaben werden also in allen drei Sprachen gelöst. Man kann dann Besonderheiten, Geschwindigkeit, Programmierstile und so weiter vergleichen. Eine ganz genaue Inhaltsübersicht des Kurses existiert zwar noch nicht, wohl aber ein grober Rahmen der einzelnen Themen. führen. Stichworte sind:

Die ersten Schritte in den Programmiersprachen, Portzugriffe, Zeitschleifen

– Verwendung der seriellen Schnittstelle
– Anwendung der Timer des 8051, Zähler, Frequenzmesser, PWM
– Ansteuerung des LCD, Textausgabeeinheit mit RS232
– Anschluss und Ansteuerung eines A/D-Wandlers und eines D/A-Wandlers
– Erweiterungen über den I2C-Bus, Portexpander, A/D-D/A, Echtzeituhr
– Besonderheiten des 89S8252, das interne 2-K-EEPROM und der Watchdogtimer