

Countdown-Timer

Präzises Timing mit AT90S1200

Von Andy Morell

andy@morell.freeserve.co.uk

Das Projekt, das ein Intervall von maximal 99 Minuten herunterzählt, ist einfach zu realisieren. Der Countdown wird in einem 2-Digit-Display angezeigt. Nach Ablauf der eingestellten Zeit ertönt ein Summer.

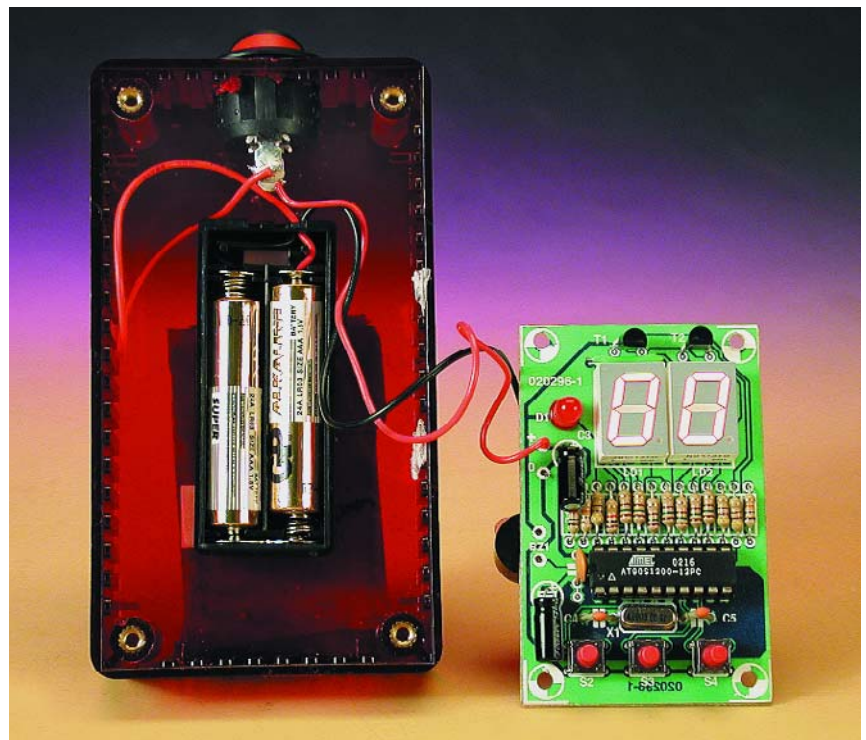
Eigenschaften

- Countdown-Intervall von 1...99 Minuten
- Zwei 7-Segment-LED-Displays
- Steuerung mit drei Tasten
- Niedrige Stromaufnahme
- 3-V-Betrieb
- Mikrocontroller AT90S1200

Die Bedienung des Timers ist die Einfachheit selbst. Das Intervall wird mit den Tastern S4 in Minuten und S3 (x 10 Minuten) eingestellt. Jeder Tastendruck inkrementiert die eingestellte Zahl. Beim zehnten Tastendruck springt die Zahl wieder auf null. Nach der Bedienung der beiden Tasten erscheint das Intervall im Display. Man drückt auf die Start-Taste und der Timer läuft. Die rote LED D1 signalisiert dies sofort. Wenn das Intervall abgelaufen ist, blinkt das Display und ein Summer gibt für etwa eine Minute Laut. Dann stoppt die Sleep-Funktion den Summer. Ein neuer Countdown-Zyklus kann erst nach einem Reset (Timer aus- und wieder einschalten.) gestartet werden.

Hardware

Das Schaltbild des Countdown-Timers ist in **Bild 1** zu sehen. Der Mikrocontroller AT90S1200 nutzt seinen internen Timer0, um ein genaues Zeitsignal von einer Sekunde respektive einer Minute zu generieren. Der Controller, der von einem 4-MHz-Quarz getaktet wird, ist billig, leicht erhältlich und arbeitet klaglos an einer (Batterie-) Versorgungsspannung bis hinunter zu 3 V. Die beiden Displays sind parallel an dem



Controllerport PB angeschlossen. Der freie Anschluss PB7 wird vom aktiven DC-Summer genutzt. Da die Displays parallel angeschlossen sind, müssen sie gemultiplext werden. Diese Aufgabe übernehmen die beiden Portleitungen PD5 und PD6, die über die PNP-Transistoren T1 und T2 die gemeinsamen Anoden der HD11310 ansteuern. Der Strom durch die Segmente wird von R3...R9 begrenzt.

Die Displays HD1131 gibt es in verschiedenen Ausführungen. Das Suf-

fix deutet auf die Farbe, in unserem Fall mit O auf „Super-Rot“ hin. Die Helligkeit kann durch ein weiteres Kürzel (zum Beispiel L) im Display-Kode beschrieben werden. Normalerweise verwendet man Strombegrenzungswiderstände von 100...120 Ω . Wenn man - wie hiermit empfohlen - sehr helle Display einsetzt, darf der Widerstandswert auf 150 Ω erhöht werden. Über die eingesparte Energie freut sich die Batterie. Die drei Ports PD2...PD4 für die Taster sind mit Pull-up-Widerstän-

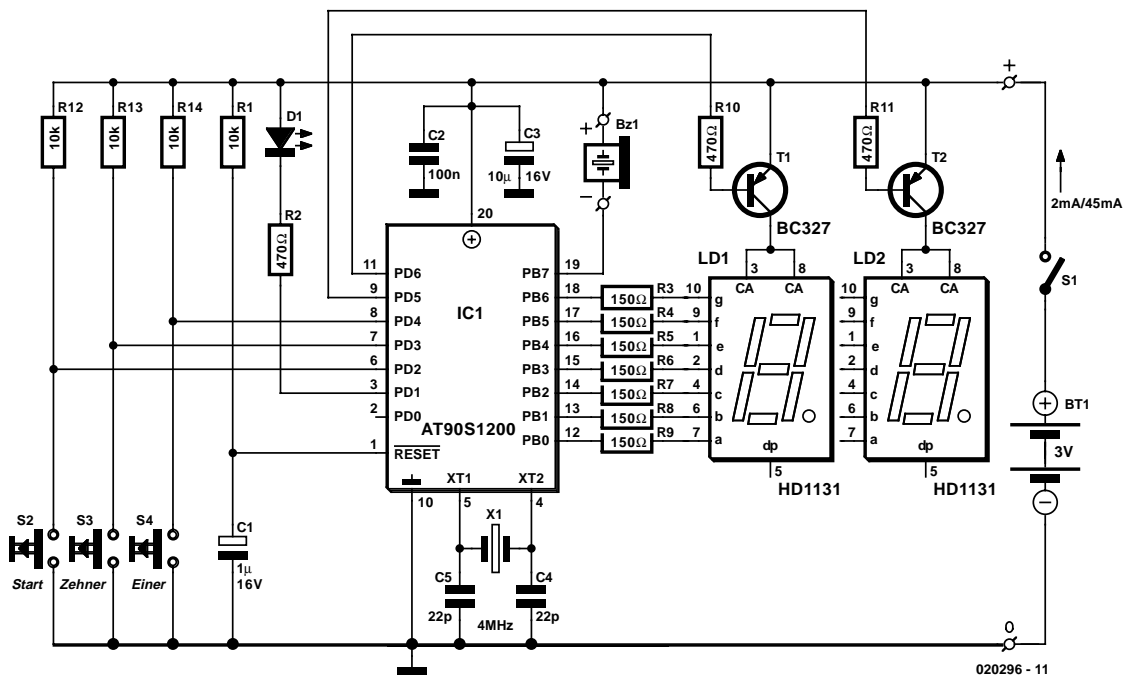


Bild 1. Schaltplan des Countdown-Timers

den R12...R14 ausgestattet. Der Reset-Impuls wird beim Einschalten von R1 und C1 erzeugt. Die Stromaufnahme liegt in Ruhe bei 2 mA, mit Displays steigt dieser Wert auf rund 45 mA.

Aufbau der Software

Werfen wir einen Blick auf sind die Flussdiagramme des Hauptprogramms (Bild 2) sowie der Interrupt-Routine (Bild 3). Auch der Pro-

grammkode, der als Hex- und Sourcefile per Download auf der Elektor-Website gratis und im Elektor-Service auch auf Diskette (EPS 020296-11) oder in einem fertig programmierten Controller (EPS 020296-41) erhältlich ist, sollte auf dem Bildschirm oder in gedruckter Form in Ihrem Blickfeld sein. Nach dem Reset setzt der Mikrocontroller Stack-Pointer, damit Adressen und so weiter gespeichert werden können, wenn Subroutinen (insbesondere die hier eingesetzte Interrupt-Routine) aufgerufen werden. Das Steuerregister TCCR0 von Timer0 wird auf eine Verteilung von 256 gesetzt, so dass bei einem 4-MHz-Quarz der Timer0-Interrupt genau im Sekundentakt erfolgt. Dazu muss das Register mit 0b00000100 oder 04hex beschrieben werden. Auf einen Interrupt hin wird die Routine DISPLAY_DIGITS aufgerufen. Es werden 61 Interrupts im Register *time out* gezählt. Der relevante Abschnitt der Route lautet:

```

DISPLAY_DIGITS:

IN SAVE_STATUS, SREG
CPI FLAG_1, $FF
BRNE ALARM_NOT_SET

INC TIME_OUT
CPI TIME_OUT, 61 ;1 SECOND
BREQ TIMER_OUT
    
```

Damit der Timer0-Interrupt erscheint, muss außerdem Bit 1 (TOIE0) im TIMSK-Register gesetzt werden. Nun ist alles für den Inter-

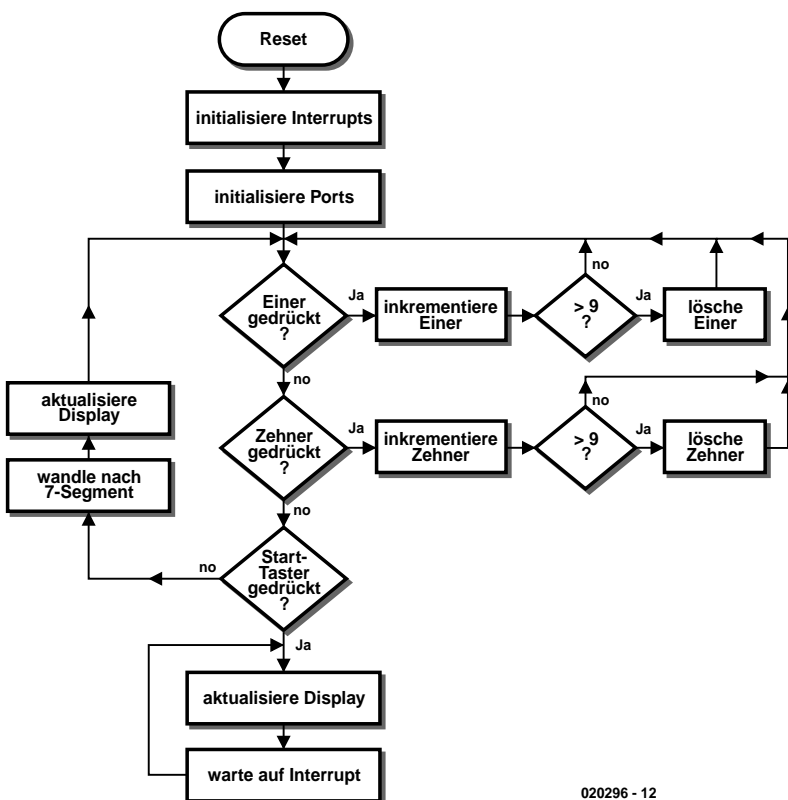


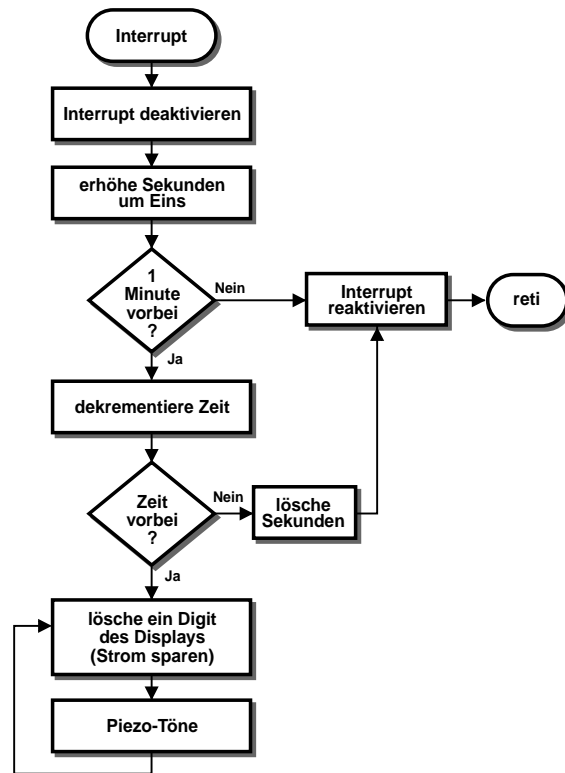
Bild 2. Flussdiagramm des Hauptprogramms

rupt vorbereitet. Da er nun nicht sofort aktiv werden soll, legen wir ihn mit dem CLI-Befehl (Clear Interrupt) lahm.

Als Nächstes werden die verschiedenen Ports als Ein- oder Ausgänge definiert. Es ist einfach zu sehen, wie die Taster arbeiten. Je nachdem, ob man T3 oder T4 drückt, startet die entsprechende Routine, die den Wert inkrementiert. Dabei wird garantiert, dass der Wert die Neun nicht überschreitet und in der Anzeige keine hexadezimalen Werte (A...F) zu sehen sind. Bevor die Start-Taste gedrückt wird, befindet sich das Programm in einer Schleife, in der die Routine *Get Seven Segment* dem Wert das entsprechende 7-Segment-Muster zuordnet und das Display ansteuert.

Beim Druck auf den Start-Taster verzweigt das Programm in die Countdown-Abteilung. Jetzt wird der Interrupt wieder eingeschaltet, so dass die Interrupt-Service-Routine ISR sich zu jeder Sekunde bemerkbar macht. Das Programm befindet sich in einer Schleife, in der bei jedem Durchlauf (also einmal pro Sekunde) die Displayanzeige erneuert wird, wobei natürlich auch stets die erwähnte Umwandlung von dezimalem Wert in ein 7-Segment-Muster stattfindet.

In der Interrupt-Routine wird zunächst der Interrupt abgeschaltet. Der Wert für die Sekunden wird erhöht und kontrolliert, ob eine Minute erreicht ist. Ist das nicht der Fall, wird der Interrupt wieder freigeschaltet und der Rücksprung ins Hauptprogramm durchgeführt. Beim 60. Sekundenimpuls ist die Minute voll, der Countdown-Wert wird vermindert und kontrolliert, ob das Intervall abgelaufen ist. Wenn dies noch nicht geschehen ist, geht es, nachdem der Sekundenanzähler auf Null gesetzt und der Interrupt



020296 - 13

Bild 3. Die Interrupt-Service-Routine

wieder freigegeben wurden, wieder ins Hauptprogramm. Die Routine *Convert To BCD* verhindert, dass die Displays andere alphanumerische als 0...9 zeigen. Entdeckt die Software, dass COUNT0 und COUNT1 null sind, verlässt sie ihre gewohnten Bahnen, aktiviert sie für eine Minuten den Summer. Dann fällt der Controller in den Schlaf

und kann nur durch ein Reset wieder erweckt werden.

Timerbau

Der Aufbau des Countdown-Timers auf der Platine, deren Layout und Bestückungsplan in **Bild 4** zu sehen ist, birgt keinerlei Probleme. Zuerst wird die Drahtbrücke über LD1

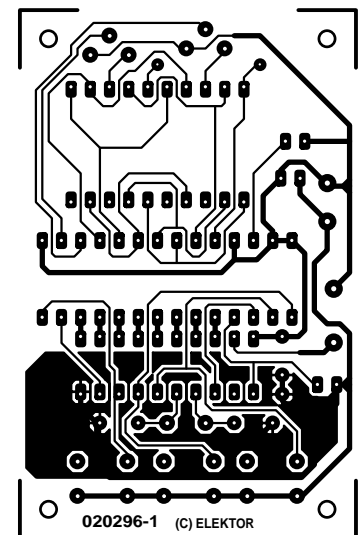
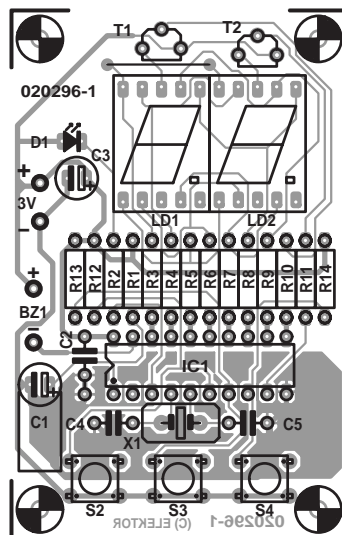
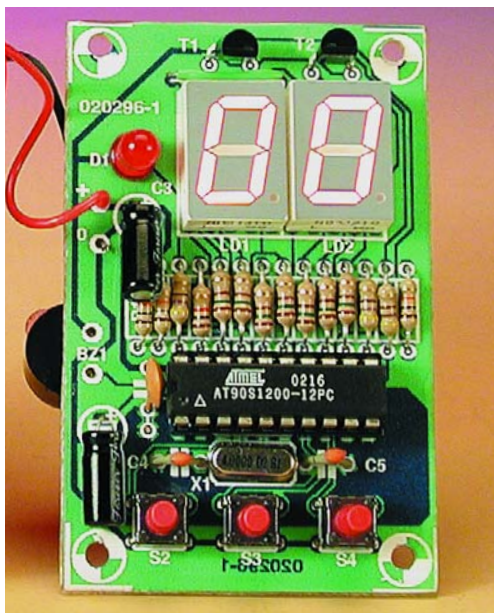


Bild 4. Platinenlayout für den Timer

Stückliste

Widerstände:

R1,R12,R13,R14 = 10 k
R2,R10,R11 = 470 Ω
R3...R9 = 150 Ω

Kondensatoren:

C1 = 1 μ/16 V stehend
C2 = 100 n
C3 = 10 μ/16 V stehend
C4,C5 = 22 p

Halbleiter:

DI = LED, rot, 3 mm
IC1 = AT90S1200-12PC
(programmiert EPS 020296-41)
LD1,LD2 = HD11310

T1,T2 = BC327

Außerdem:

BT1 = 2 Mignon- oder Lady-Zellen
mit Halterung
Bz1 = 5-V-Gleichstromsummer
S1 = Ein/Aus-Schalter
S2...S4 = Taster I-an
X1 = Quarz 4 MHz, niedrige
Bauhöhe
Software auf Diskette (EPS020296-11)
oder als Gratis-Download
Platinen-Download (020296-1)
Fertigplatine und
Bezugsquellenhinweise siehe
www.elektor.de/pcbs/pcbs.htm

angebracht, danach die Widerstände, die Kondensatoren (achten Sie auf die Polarität von C1 und C2!), danach setzt man eine 20-polige DIL-Fassung für den Controller und den Quarz ein. Für den Anschluss von Piezo-Summer und Batterien

bringt man noch ein paar Lötnägel oder ausreichend lange Stückchen Schaltaht an.

Je nach Gehäuseausführung kann es erforderlich sein, Displays und Taster zu „verlängern“. Ideal dazu eignen sich einreihige Fassungen mit lan-

gen Beinchen, die man mit etwas „Distanz“ festlötet. Bei den Tastern reichen auch steife Drahtstückchen. Die LED verfügt über ausreichend Beinfreiheit, um sich einen Platz im Gehäusedeckel zu sichern.

Sind diese Fragen der Displaybefestigung gelöst und die Steifenkontakte gefestigt, lötet man noch die beiden Transistoren fest, wirft noch einen abschließenden Blick über die Lötarbeiten, verpackt die Schaltung im Gehäuse, schließt Batterien (über den externen Ein/Aus-Schalter) und den Summer an, und fertig ist der Countdown-Timer.

(02096)rg

Downloads

Unter www.elektor.de/dl/dl.htm

stehen für dieses Projekt folgende Dateien zum Download bereit:

- 020296-11 (Programm des Countdown-Timers, Source- und Hex-Code)
- 020296-1 (Layout der Platine im PDF-Format)