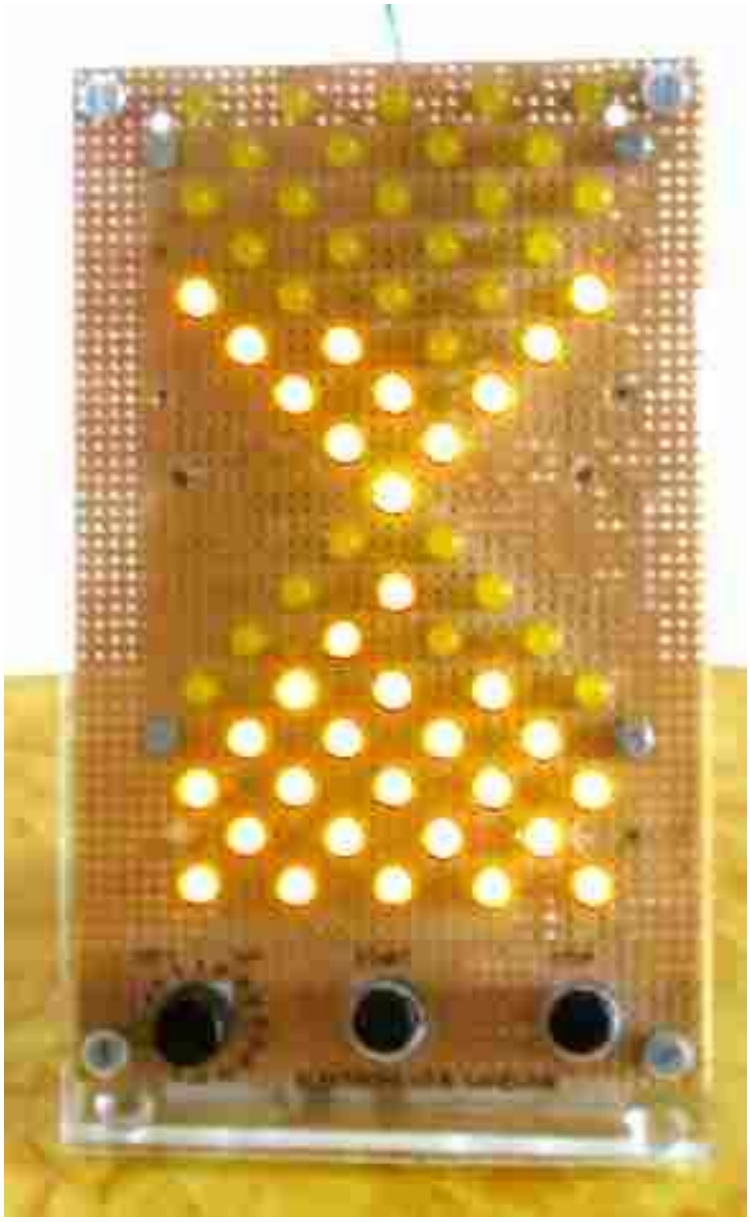
Elektronische Sanduhr



Autor: Buchgeher Stefan

Letzte Bearbeitung: 5. März 2004

Inhaltsverzeichnis

1. Einleitung

Wozu eine elektronische Sanduhr?

Nun, dieses Projekt soll nicht als eine „Elektronisierung der Welt“ missverstanden werden, sondern, als eine Übung in der Programmierung bzw. als eine Übung im Umgang mit einem PIC-Mikrocontroller.[1]
Die Übung, die Umsetzung von Ideen und die damit verbundene Suche nach Lösungsmöglichkeiten stehen hier im Vordergrund. Dies ist auch der Grund weshalb ich alle erarbeiteten Unterlagen (Schaltpläne, Listings, etc) zur freien Verfügung bereitstelle. Die von mir eingestellten Parameter lassen sich eigenen Bedürfnissen anpassen. Doch dazu mehr im Kapitel 5 (Modifikationshinweise).
Ich hoffe, dass das gesamte Projekt ausreichend dokumentiert ist, so dass auch ein nicht Elektronik-Profi mit diesem Projekt zu Recht kommt.

Bei jeder Beschäftigung, bei jedem Hobby soll natürlich auch ein sinnvolles Ergebnis der „harten Arbeit“ entstehen. So auch bei meiner Sanduhr. Was sind nun die Vorteile einer elektronischen Sanduhr gegenüber einer Herkömmlichen? Als erster großer Vorteil ist zu nennen, dass man hier die Zeit, welche verrieseln soll einstellen kann (in gewissen Grenzen natürlich!). Was passiert, wenn bei einer herkömmlichen Sanduhr die Zeit verstrichen ist? – Nichts! Und wenn man die Sanduhr nicht die ganze Zeit im Auge behält, dann bekommt man auch gar nicht mit, wenn nun die Zeit tatsächlich abgelaufen ist. Bei einer elektronischen Version kann man, sobald die *eingestellte* Zeit verstrichen ist, Aktionen einleiten. Im einfachsten Fall (so wie hier) einen Lautsprecher aktivieren, so dass man auch akustisch auf die abgelaufene Zeit hingewiesen wird, und man sich daher zwischenzeitlich anderen Aktivitäten zuwenden kann. Ich denke da zum Beispiel ans Kochen. Soll ein Kuchen etwa eine halbe Stunde im Backrohr verweilen, so stellt man diese Zeit ein und drückt die „Start-Taste“. Nun kann man sich anderen Tätigkeiten widmen, und muss nicht immer auf die Uhr sehen. Was diese Sanduhr natürlich nicht kann (oder noch nicht kann) ist, dass sie den Ofen nach der eingestellten Zeit abschaltet. Wenn man nun die Wohnung verlässt, und auf den Kuchen vergisst, so wird auch diese Sanduhr keine Wunder vollbringen! Ich will damit nur sagen, dass man schon noch im Hörbereich des Lautsprechers verweilen sollte.
Mittlerweile gibt es eine zweite Version. Diese ist mit einem Schaltausgang ausgerüstet. Somit ist es nun auch möglich mit dieser Sanduhr externe Geräte zu „steuern“. Möglicherweise müssen an diesen Geräten Anpassungen vorgenommen werden, damit sie mit der Sanduhr ein- und ausgeschaltet werden können. Der Schaltausgang verlangt jedoch, dass auf eine andere Funktion der Urversion verzichtet werden muss. Dies betrifft den Jumper zur Auswahl ob der Zustandswechsel der Leuchtdioden mit einem Beepton untermalt werden soll. Diese Dokumentation beschreibt ausführlich die Urversion. Die Version mit dem Schaltausgang wird im Kapitel 5.4 erläutert.

Eine elektronische Sanduhr hat schon seine Vorteile und soll nicht als sinnlose Spielerei abgestempelt werden. Es steht aber nicht der praktische Nutzen, sondern die Programmierung und der Umgang mit Mikrocontrollern im Vordergrund dieses Projektes!

Buchgeher Stefan

[1] Es könnte natürlich auch ein Controller einer anderen Familie verwendet werden, aber ich möchte mich hauptsächlich auf die PIC-Familie beschränken, ich bin schließlich keine Firma, und kann mir daher nicht jede Entwicklungsumgebung leisten!

2. Bedienung

Neben dem Ein/Aus-Schalter auf der Rückseite befinden sich auf der Frontplatte folgende Anzeige- und Bedienelemente:

- Anzeigefeld: 65 Leuchtdioden in Form einer herkömmlichen Sanduhr
- Wahlschalter mit 16 Positionen (demnach auch 16 verschiedene Zeiten einstellbar)
- Starttaste
- Stopptaste

Die Bedienung der Sanduhr ist einfach:
Nach dem Einschalten des Gerätes mit dem **Ein/Aus**-Schalter (auf der Rückseite) leuchten die unteren Leuchtdioden. (Vorausgesetzt, dass entweder ein **Steckernetzteil** oder eine volle Batterie angeschlossen ist!) Die gewünschte, abzulaufende Zeit wird mit dem **Wahlschalter** eingestellt. Zur Auswahl steht ein Bereich von 15 Sekunden bis 1 Stunde.[2] Sobald die **Starttaste** gedrückt wird werden die oberen Leuchtdioden aktiv und die eingestellte Zeit beginnt zu verrieseln. Mit der **Stopptaste** kann das rieseln jederzeit beendet werden. Nachdem die eingestellte Zeit vergangen ist, die unteren Leuchtdioden leuchten wieder, beginnt der **Lautsprecher**, auf der Rückseite, zu summen. Der Lautsprecher lässt sich entweder mit der Stopptaste beenden, oder man beginnt mit der Starttaste eine neue „Zeitmessung“ (mit der eingestellten Zeit).

Auf der Platine befindet sich ein Jumper. Ist dieser gesteckt, so ertönt bei jedem Zustandswechsel der Leuchtdioden ein Beep-Ton.

Zum Ein/Aus-Schalter (auf der Rückseite) ist zu sagen, dass dieser drei Positionen besitzt, wobei in der Mittelstellung die Sanduhr ausgeschaltet ist. Zeigt der Hebel des Schalters zur Frontplatte, so wird eine angeschlossene 9-V-Batterie (oder ein 9-V-Akku) zur Stromversorgung herangezogen. Zeigt der Hebel in die entgegengesetzte Richtung, so wird der Strom aus einem angeschlossenen Steckernetzteil entnommen (es genügt ein unstabilisiertes 9-V-Steckernetzteil).

3. Schaltungsbeschreibung

Beim Betrachten der Schaltpläne (ANHANG A) fällt sofort das 8x8-LED-Matrix-Feld auf, das ja auch eines der Hauptbestandteile dieses Projekts darstellt. Wo bleiben die Vorwiderstände für die Leuchtdioden? Da hier ein Multiplex-Verfahren angewendet wird und immer nur eine einzige Leuchtdiode für einen Bruchteil einer Sekunde leuchtet kann auf die Vorwiderstände verzichtet werden. Die Zeilen und Spalten der 8x8-Matrix werden über je eine Transistorstufe angesteuert. Dass immer nur eine Zeile bzw. eine Spalte aktiv ist, dafür sorgen zwei 8-aus-3-Dekoder (IC2 und IC3). Welche Leuchtdiode gerade leuchten soll wird vom Controller (IC1) bestimmt.

Die zweite wichtige Komponente ist der Controller (IC1). Hier wird ein PIC16F84 verwendet. Dieser ist leicht erhältlich und besitzt genau die Anzahl an I/O-Pins, die hier benötigt werden. Der größte Vorteil an diesem Controller ist, dass er einen Flash-Speicher besitzt und daher fast beliebig oft einfach neu programmiert werden kann. Der PIC16F84 besitzt auch ein EEPROM, welches jedoch für diese Anwendung nicht

[2] Dieser Bereich lässt sich in der Software eigenen Bedürfnissen anpassen. Dazu mehr im Kapitel 5 (Modifikations-hinweise)

benötigt wird. Die Aufgabe des Controllers ist es den gesamten Ablauf, vom Einschalten bis zur Signalisierung der abgelaufenen Zeit zu steuern. Mehr dazu im Kapitel 4.

Als Takterzeugung wird eine Standardapplikation bestehend aus einem X1,C6,C7 und R5 verwendet. Bei X1 sollte es sich um einen 4-MHz-Quarz handeln, mit einem 4-MHz-Resonator würde die Schaltung zwar auch funktionieren, die Zeiten wären aber ungenau. Bei Verwendung eines Quarzes stimmten die eingestellten Werte (Konstanten) mit den errechneten Werten überein.

Zur Erzeugung des Reset wurde ebenfalls eine einfache Standardlösung bestehend aus R4 und C8 gewählt. Die Stopptaste (S2) führt ebenfalls einen Reset aus, und versetzt den Controller daher in den Ausgangszustand.

An den I/O-Pins RB0 bis RB2 befindet sich der „Zeilendekoder“ IC2, der „Spaltendekoder“ IC3 bekommt die I/O-Pins RB3 bis RB5. Diese 6 I/O-Pins sind also für die Ansteuerung der LED-Matrix zuständig. Als Leuchtdioden werden gelbe, 5mm Standardtypen verwendet. (vgl. Stückliste im ANHANG D)

Der Lautsprecher (LS1) wird über eine Transistor-Schaltstufe (T17, R6) vom Contoller-Pin RB6 angesteuert.

Die abzulaufende Zeit wird mit einem HEX-Codierschalter (S1) eingestellt. Dieser benötigt für die 16 möglichen Schalterstellungen 4 Leitungen, und je einen Pull-up-Widerstand (Widerstandsarray R1 oder 4 Einzelwiderstände). Als Controller-Pins wurden RA0 bis RA3 verwendet.

Optional ertönt bei jedem Zustandswechsel der LEDs ein Beepton, falls der Jumper JP1 gesteckt ist. (Controller-Pin RA4).

Gestartet wird die LED-Sanduhr mit der Starttaste S3, welcher über einen Pull-Up-Widerstand (R7) mit dem Controller-Pin RB7 verbunden ist.

Die LED D65 ist mit einem Vorwiderstand (R3) direkt mit der Betriebsspannung verbunden. Diese Leuchtdiode leuchtet daher ständig. Sie ist deshalb auch in der Mitte des „LED-Feldes“ angeordnet.

Zur Stromversorgung gibt es nicht viel zu sagen. Ein Festspannungsregler vom Typ 78L05 übernimmt mit den Kondensatoren C1 und C2 die Spannungsregelung. D66 dient als Verpolungsschutz. Mit dem Ein/Aus/Ein-Schalter (S4) lässt sich nicht nur die Sanduhr ein und ausschalten. Hier wird auch bestimmt, welche Stromquelle die Sanduhr versorgen soll. Alternativ zu einem Steckernetzteil kann eine 9-V-Batterie oder ein 9-V-Akku verwendet werden. Eine automatische Ladung des Akkus ist hier nicht vorgesehen!

4. Softwarebeschreibung (der Urversion)

4.1. Allgemeines zur Software

Die Aufgaben des Controllers (IC1, PIC16F84) lassen sich grob in folgende Teilaufgaben zerlegen:

- Die 8x8-LED-Matrix ansteuern. Dies soll im Hintergrund passieren und wurde daher mit einem (Timer)-Interrupt realisiert
- Die LED-Adressen der LEDs, welche gerade leuchten sollen, in die Register POS1 bis POS32 laden
- Die Starttaste abfragen, und nachdem dieser gedrückt wurde den HEX-Codierschalter. (Die Stopptaste muss nicht extra abgefragt werden, da diese direkt mit dem Reset-Eingang verbunden ist! Beim Drücken der Stopptaste wird daher ein Reset ausgelöst und der Controller beginnt wieder von vorne)
- Den Lautsprecher nach abgelaufener Zeit aktivieren

Das Besondere bei diesem Projekt ist, dass die Adressen der 32 LEDs, welche gerade leuchten, in den RAM-Speicherregistern, mit den selbst gewählten Namen POS1 bis POS32 stehen. Im Ausgangszustand (die unteren 32 LEDs leuchten) beinhalten die RAM-Register POS1 bis POS32 die Leuchtdioden mit den Adressen der LEDs D33 bis D64. In POS1 steht also der Wert b‘10100011‘ (=Adresse der LED D33) und in POS32 der Wert b‘10000000‘ (=Adresse der LED D64). vgl Unterprogramm ZUSTAND0. Dass immer die Leuchtdioden leuchten welche in den Registern POS1 bis POS32 leuchten, dafür ist die Interrupt Service Routine (kurz ISR) zuständig. Mehr dazu in Kapitel 4.3. Nach dem Betätigen der Starttaste soll nun die Sanduhr „umgedreht“ werden, was bedeutet, dass nun die oberen 32 Leuchtdioden leuchten. In den RAM-Registern POS1 bis POS32 stehen nun die LED-Adressen der oberen 32 Leuchtdioden. In POS1 demnach b‘10111111‘ (=Adresse von LED D1) und in POS32 der Wert b‘10010100‘ (=Adresse von LED D32). Die ISR bekommt jetzt eine zusätzliche Aufgabe, sie muss jetzt bei jedem Aufruf das Zählregister AKTZEITx um 1 vermindern. Bei diesem Register handelt es sich um ein 16-Bit-Zählregister. Dieses muss daher auf zwei 8-Bit Register mit den Namen AKTZEITL und AKTZEITH aufgeteilt werden. Ist dieses 16-Bit-Register Null, so wird ein Bit (NaechstZust-Flag) gesetzt und das Hauptprogramm weis nun, dass ein Zustandswechsel der Leuchtdioden erfolgen muss. Für den Zustand, wo die oberen 32 Leuchtdioden leuchten und dem Ausgangszustand (die unteren 32 Leuchtdioden leuchten) stehen die zwei Unterprogramme ZUSTAND1 und ZUSTAND0 zur Verfügung. Für alle Zustände dazwischen ebenfalls ein eigenes Unterprogramm zu schreiben, wäre zwar eine Möglichkeit, aber mit einem PIC16F84 wahrscheinlich nicht zu realisieren, da dies zu viel Programmspeicher benötigen würde. Es geht auch anders, doch dazu mehr im Kapitel 4.4.5.

Etwas genauer soll hier noch ein Register namens SUSTATUS erklärt werden. Dieses Register ist sozusagen das Sanduhr-Statusregister, wobei nur die drei niederwertigsten Bits benötigt werden. Im Ausgangszustand besitzen alle Bits den Wert 0.

- *Bit 0:* Dieses wird gesetzt, wenn die Starttaste gedrückt wird, und bleibt solange gesetzt (High) solange der „Riesel-Vorgang“ dauert. Erst wenn die eingestellte Zeit abgelaufen ist. wird dieses Bit wieder gelöscht.
- *Bit 1*: Dieses Bit wird von der ISR gesetzt, wenn ein Zustandswechsel erfolgen soll. Nach dem Ausführen des Zustandswechsels (eigenes Unterprogramm) wird dieses Bit wieder zurückgesetzt.

- *Bit 2*: Sobald die eingestellte Zeit abgelaufen ist, wird dieses Bit gesetzt und signalisiert dem Hauptprogramm, dass der Lautsprecher aktiviert werden muss.

4.2. Hauptprogramm

Die Aufgaben des Hauptprogramms lassen sich gut mit einem Flussdiagramm beschreiben (siehe nächste Seite).
Zuerst wird der Prozessor initialisiert, und der Ausgangszustand (die unteren LEDs leuchten) hergestellt. Diese beiden Tätigkeiten werden von je einem Unterprogramm ausgeführt (INIT bzw. ZUSTAND0). Erst jetzt werden der Timer0-Interrupt und der globale Interrupt freigegeben.[3] Dafür ist das Register INTCON zuständig. Je nach benötigten Interrupts werden die entsprechenden Freigabebits (im Englischen: Enable) gesetzt. Wird ein Interrupt verwendet so muss zusätzlich zum verwendeten Interrupt auch die globale Interruptfreigabe GIE (**G**eneral **I**nterrupt **E**nable) gesetzt werden. Er ist sozusagen der Hauptschalter, der Interrupts ermöglicht. Das INTCON-Register kann einfach mit den Befehlen `movlw b‘10100000‘` und `movwf INTCON` beschrieben werden. Der Timer0-Interrupt ist jetzt eingeschaltet. Er sorgt hier bei der Sanduhr unter anderem dafür, dass der Benützer den Eindruck hat, dass die 32 leuchtenden LEDs gleichzeitig leuchten. In Wirklichkeit leuchtet immer nur eine LED für einen Bruchteil einer Sekunde. Diese Verfahren wird Multiplex genannt. Mehr dazu im Kapitel 4.3

Nun wartet das Hauptprogramm solange bis die Starttaste (S3 am Port RB7) gedrückt wird. Dieser Pin also LOW wird. Eine einfache Schleife wird dazu benötigt:

```
START       btfsc PORTB,STARTTASTE
            goto  START
```

Wurde die Starttaste gedrückt, so wird der Port RB7 low und die oben genannte Schleife wird verlassen. Das Hauptprogramm geht in eine neue Schleife, in die HAUPTSCHLEIFE über. Hier erfolgt als erste Tätigkeit das setzen von SUSTATUS.0 (vgl. Kap. 4.1 Allgemeines zur Software)
Das Bit SUSTATUS.2 („Fertigbits“) könnte gesetzt sein, und muss daher wieder gelöscht werden. Warum kann das „Fertigbit“ gesetzt sein? Das Flussdiagramm zeigt es deutlich. Wird zum Beispiel die Sanduhr mehrmals hintereinander benützt (ohne das sie dazwischen ausgeschaltet wird), beginnt das Hauptprogramm, nachdem die Starttaste wieder gedrückt wurde wieder in der Hauptschleife, und hier ist ja das „Fertigbit“ noch gesetzt.

Es leuchten noch die unteren Leuchtdioden. Eine herkömmliche Sanduhr würde man jetzt auf den Kopf stellen. Hier, bei der elektronischen Version, werden einfach die unteren LEDs „ausgeschaltet“ und die oberen LEDs „eingeschaltet“. D. h. die Adressen der oberen 32 LEDs werden in die RAM-Register POS1 bis POS32 geladen. Diese Aufgabe übernimmt das Unterprogramm ZUSTAND1.

Damit das Programm weis in welchem Zustand es sich befindet (welche LEDs gerade leuchten), dient ein Zählregister namens AKTZUSTAND. Dieses Register muss ebenfalls mit dem entsprechenden Wert geladen werden. Dieser Wert steht in der Konstante ZUSTANZ. Mit den folgenden Befehlen wird der Inhalt der Konstante

[3] Der PIC16F84 besitzt 4 Interruptquellen. Bei der elektronischen Sanduhr wird aber nur der Timer0-Interrupt benötigt.

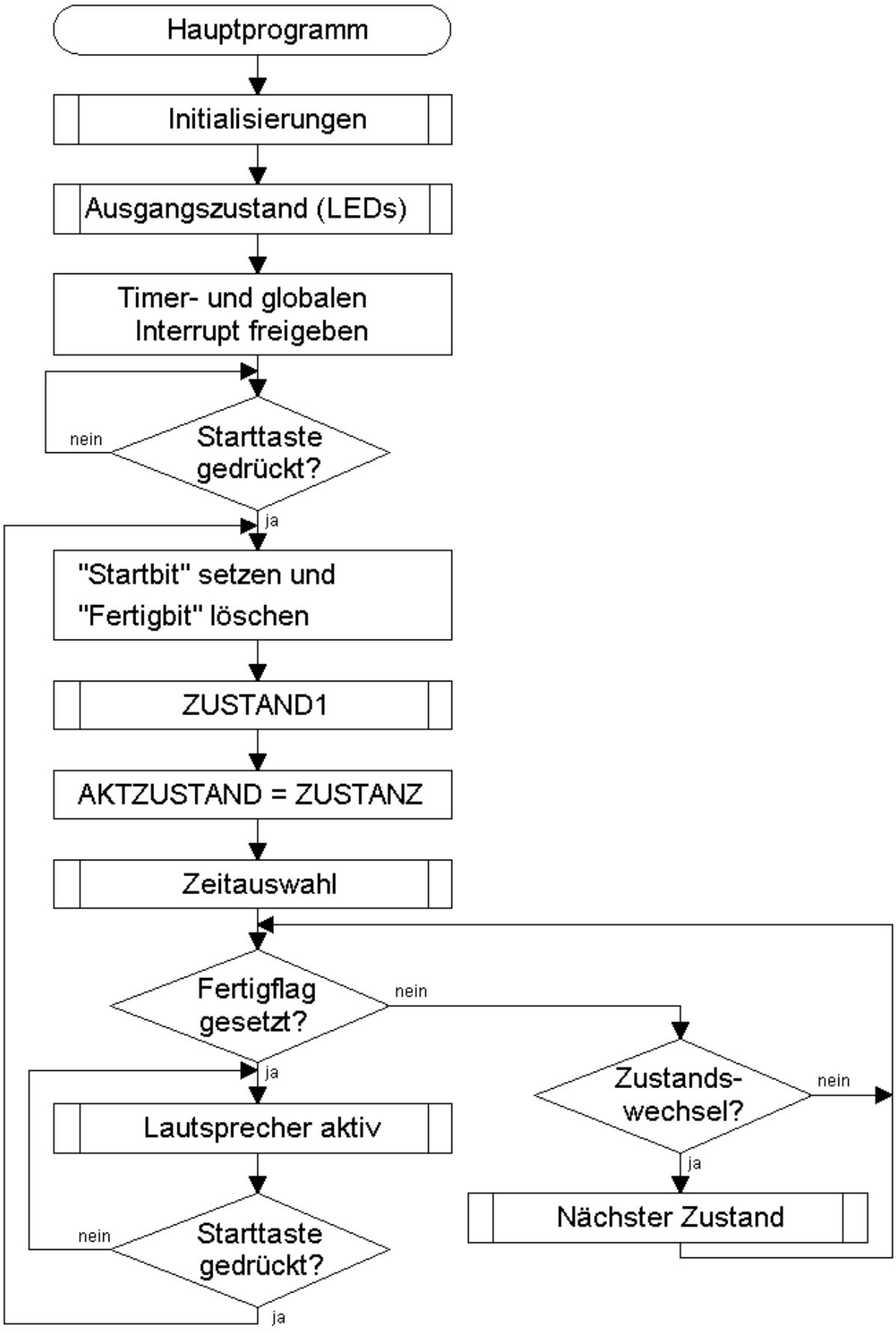


Bild 1: Flussdiagramm des Hauptprogramms

(ZUSTANZ) in das Register AKTZUSTAND kopiert:

```
movlw       ZUSTANZ
movwf       AKTZUSTAND
```

Die Konstante ZUSTANZ besitzt übrigens den Wert 112. Es gibt demnach 112 Schritte zwischen dem Zustand wo die oberen 32 Leuchtdioden leuchten und dem Zustand wo die unteren 32 leuchten.

Als nächstes erfolgt die Abfrage der eingestellten Zeit und das Laden der entsprechenden Register. Diese Aufgabe übernimmt wieder ein eigenes Unterprogramm (ZEITAUSWAHL), und soll daher an dieser Stelle nicht weiter beschrieben werden. (Kap. 4.4.4)

Es wurden nun alle Parameter vom Hauptprogramm erfasst und den entsprechenden Registern zugewiesen. Jetzt muss „nur“ mehr das „Sanduhr-Status“-Register ständig geprüft und die entsprechenden Aktionen ausgelöst werden: Ist das Fertigflag gesetzt muss der Lautsprecher solange summen, bis die Starttaste erneut gedrückt wird. Andernfalls wir das „Zustandswechselflag“ (SUSTATUS.1) geprüft. Ist dieses gesetzt, so erfolgt mit Hilfe des Unterprogramms NAECHSTZUST ein Zustandswechsel (vgl. Kap. 4.4.5). Andernfalls wird das „Sanduhr-Status“-Register erneut geprüft. Dieses Prüfen geschieht solange, bis das „Fertig-Flag“ gesetzt ist.

4.3. ISR (Timer 0)

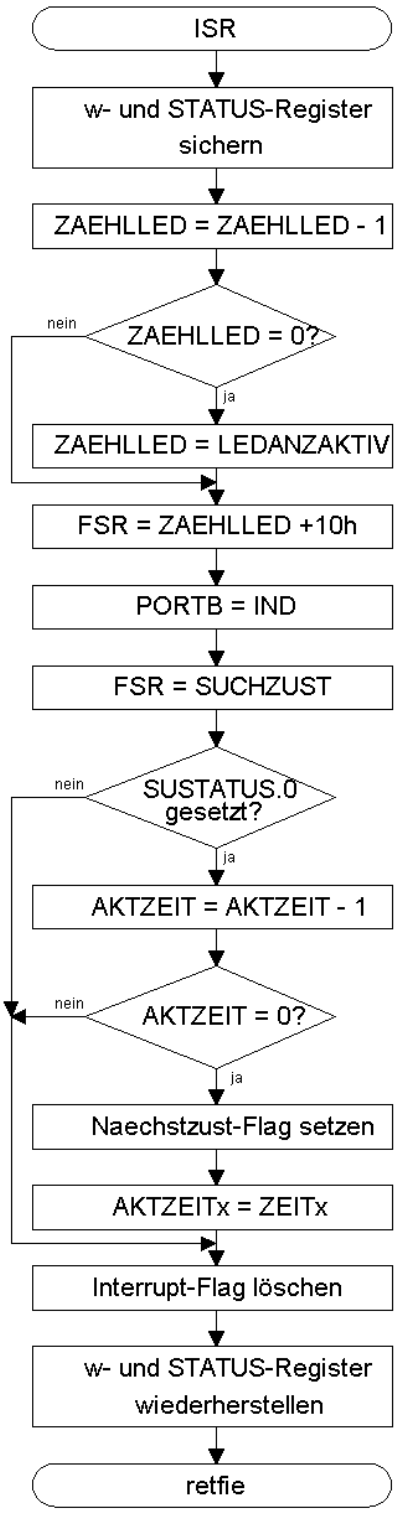
Eine ISR (**I**nterrupt **S**ervice **R**outine) ist im Prinzip ein Unterprogramm, welches aber im Gegensatz zu normalen Unterprogrammen, „unvorhergesehen“ aufgerufen wird. Hier, beim Timer 0-Interrupt jedes Mal, wenn der Timer 0 überläuft, also von 255 auf 0 wechselt. Würde zum Beispiel ein RB-Interrupt verwendet werden, so würde bei jeder Pegeländerung von RB4 bis RB7 ein Interrupt auftreten und die entsprechende ISR wird ausgeführt. Eine ISR sollte daher so kurz wie möglich sein.

Ein weiterer wichtiger Punkt bei einer ISR ist, dass das w-Register (Working- oder Arbeitsregister) und das STATUS-Register in andere Register zwischengespeichert werden müssen, falls diese in der ISR ihren Registerinhalt verändern. Der Grund dafür ist, dass eine ISR eben unvorhergesehen aufgerufen wird, und die angesprochenen Register unter Umständen zu diesen Zeitpunkten gerade benötigte Werte enthalten. Nach Ausführung der ISR springt diese zwar wieder genau an die Stelle zurück, wo sie war, bevor der Interrupt auftauchte, aber mit einem möglicherweise falschen Wert im w-Register (bzw. STATUS-Register). Das Zwischenspeichern des w-Register bzw. des STATUS-Registers wird häufig auch als PUSH bezeichnet. Das Widerherstellen von w-Register und STATUS-Register nennt man POP.

Woher weis das Programm, dass ein Interrupt aufgerufen werden muss? Dazu gibt es für jede Interruptquelle ein Kontroll-Flag. Dies wird vom Controller gesetzt wenn dieser Interrupt auftritt. (Vorausgesetzt, dass diese Interruptquelle freigegeben ist). Damit aber die ISR nicht ständig aufgerufen wird, muss dieses Bit in der ISR wieder gelöscht werden.

Nun aber zur sanduhrspezifischen Timer 0-ISR. Auch hier soll wieder ein Flussdiagramm den Ablauf etwas anschaulicher darstellen.

Die ISR wird übrigens alle 512 µs aufgerufen.[4]



Der erste Schritt ist die schon erwähnte PUSH-Befehlsfolge, also das Zwischenspeichern vom Arbeitsregister (w-Register) und dem STATUS-Register. Hierfür wurden eigens zwei Register mit den Namen w_TEMP und STAT_TEMP definiert.

```
PUSH        movwf       w_TEMP
            swapf       STAT,w
            movwf       STAT_TEMP
```

Wie schon mehrmals erwähnt werden die Leuchtdioden im Multiplex-Verfahren angesteuert. D.h. es werden nicht alle Leuchtdioden gleichzeitig betrieben sondern immer nur eine für einen Bruchteil einer Sekunde. Die Vorteile dieses Verfahrens sind: Einsparung an Leitungen und Portpins. Würde für die 64 LEDs je ein Portpin verwendet werden, würde man 64 Portpins benötigen. Mit einer 8x8-Matrix und zusätzlich noch zwei 8-aus-3-Dekoder reichen nur sechs Leitungen aus um alle 64 Leuchtdioden unabhängig voneinander anzusteuern. Ein weiterer Vorteil einer Multiplexansteuerung ist der geringere Stromverbrauch, da immer nur eine Leuchtdiode leuchtet und daher Strom benötigt.
Welche der 32 Leuchtdioden gerade leuchten soll, also am Port B ausgegeben wird, bestimmt das Zählregister ZAEHLLED. Dieses Register wird bei jedem Aufruf der ISR um eins vermindert. Hat es den Wert 0, so wird es wieder mit der Konstanten LEDANZAKTIV (=32) geladen. Im Register ZAEHLLED steht also eine Zahl zwischen 1 und 32. Diese Zahl gibt an welches POSx-Register am Port B ausgegeben werden soll. In den Registern POS1 bis POS32 stehen die Adressen der 32 aktuell leuchtenden LEDs. Um das entsprechende POSx-Register am Port B ausgeben zu können hilft uns die indirekte Adressierung. Bei dieser Adressierung wird nicht der Registername (z.B. POS1, SUSTATUS, PORTA usw.) sondern deren Adresse in das FSR-Register geladen. Im IND-Register steht dann der Registerinhalt. Beispiel: es soll mittels indirekter Adressierung auf das POS1-Register zugegriffen werden. Das POS1-Register hat die Adresse 11h (vgl. 1. Seite des Listings, ANHANG B). Das FSR-Register wird also mit dem Wert 11h geladen, und im IND-Register steht nun der Inhalt von POS1. Wozu das ganze, es geht ja einfacher mit dem Befehl *movf POS1,w* um auf das Register POS1

[4] dieser Wert ergibt sich folgendermaßen: TMR0 wird mit dem Wert 0 geladen – es dauert also 256 µs bis das Register wieder den Wert 0 besitzt, der Vorteiler besitzt den Wert 2 (vgl. Unterprogramm INIT, Kap. 4.5.1). Diese beiden Werte multipliziert ergeben den Wert 512µs bei einer Taktzeit von 1µs (bei Verwendung eines 4 MHz-Quarz)

zuzugreifen? Um die Adresse der gerade leuchtenden LED zu erhalten muss nur das Zählregister ZAEHLLED mit 10h addiert werden und in das FSR-Register geschrieben werden. Im IND-Register steht nun die Adresse der aktuellen LED. Diese muss jetzt nur mehr über das w-Register in das PORTB-Register kopiert werden. Dies ist sicherlich die einfachste Möglichkeit das entsprechende POSx-Register in das PORTB-Register zu kopieren. Diese Methode benötigt nur 5 Befehle:

```
            movlw       10
            addwf       ZAEHLLED,w
            movwf       FSR
            movf        IND,w
            movwf       PORTB
```

Ist Bit 0 von SUSTATUS gesetzt. (Die Sanduhr ist also in jenem Zustand, wo die LEDs nach unten rieseln), wird bei jedem Aufruf der ISR das 16-Bit-Zählregister AKTZEIT um eins vermindert. Ist dieses Register 0 so wird das NaechstZustand-Flag gesetzt, und dem Hauptprogramm somit mitgeteilt dass ein Zustandswechsel zu erfolgen hat. Weiters muss das AKTZEIT-Register neu geladen werden. Dieses Register gibt also die Anzahl der ISR-Aufruf bis zum nächsten Zustandswechsel an. Da es sich hierbei um ein 16-Bit-Register handelt muss es auf zwei 8-Bit-Register (AKTZEITH bzw. AKTZEITL) aufgeteilt werden.

Die sanduhrspezifischen Funktionen sind nun erledigt. Was jetzt noch fehlt, ist das Interrupt-Anforderungsbit, hier für den Timer 0-Interrupt, wieder zu löschen (T0IF im Register INTCON),

```
            bcf         INTCON,T0IF
```

und den Zustand vom Arbeits- und Statusregister wiederherstellen.

```
POP         swapf       STAT_TEMP,w
            movwf       STAT
            swapf       w_TEMP,f
            swapf       w_TEMP,w
```

Jede ISR muss mit dem Befehl *retfie* beendet werden.[5]

[5] Beim Aufruf der ISR wird automatisch das Bit GIE gelöscht, damit während der Ausführung der ISR kein weiterer Interrupt ausgelöst werden kann, was bei mehreren freigegebenen Interruptquellen durchaus möglich sein kann. Mit dem Befehl *retfie* wird zunächst an die Stelle im Programm zurückgesprungen, wo sich die Programmabarbeitung befand bevor die ISR aufgerufen wurde, und die verwendeten Interrupts werden wieder freigegeben (hier bei der Sanduhr der Timer0-Interrupt)

4.4. Unterprogramme

4.4.1. INIT

Dieses Unterprogramm initialisiert den Prozessor, in dem folgende Parameter eingestellte werden:

- Den Timer0 mit 0 voreinstellen

```
clrf        TMR0
```

- Da sich die nächsten Parameter in der Registerseite 1 befinden, muss diese zunächst selektiert werden, in dem man das RP0-Bit im Status-Register setzt

```
bsf         STAT,RP0
```

- Es wird der interne Takt und ein Vorteiler von 2 verwendet

```
movlw       b‘00000000‘
movwf       OPTREG
```

- Als nächstes werden die Ports definiert:
  - Port B: Bit 0 bis 6 Ausgänge (LED-Matrix, Lautsprecher), Bit 7 Eingang (Starttaste)
  - Port A: Alle 5 Bits sind Eingänge

```
movlw       b‘10000000‘
movwf       TRISB
movlw       b‘11111‘
movwf       TRISA
```

- Wieder die Registerseite 0 selektieren (Bit RP0 löschen)

```
bcf         STAT,RP0
```

- Das Sanduhr-Statusregister löschen

```
movlw       b‘00000000‘
movwf       SUSTATUS
```

- Das Register ZAEHLLED mit der Konstanten LEDANZAKTIV (=32) laden

```
movlw       LEDANZAKTIV
movwf       ZAEHLLED
```

- Jedes Unterprogramm muss mit dem folgenden Befehl enden

```
return
```

4.4.2. ZUSTAND0

Diese Unterprogramm lädt die Adressen der unteren 32 Leuchtdioden (LED33 bis LED64) in die Register POS1 bis POS32. Die Adressen der Insgesamt 64 Leuchtdioden stehen als Konstanten (LED1 bis LED64) zur Verfügung (vgl. Listing, ANHANG B)

4.4.3. ZUSTAND1

Wie bei ZUSTAND0, es werden jedoch die oberen 32 Leuchtdioden (LED1 bis LED32) in die Register POS1 bis POS32 geladen.

4.4.4. ZEITAUSWAHL

Das Unterprogramm für die Zeiteinstellung sieht auf dem ersten Blick etwas komplex und verwirrend aus. Man soll sich aber von der Länge nicht täuschen lassen. Zunächst muss man wissen, dass die Schalterstellung „0“ des HEX-Codierschalters (S1) am Port A einen Zustand von „x1111“ ergibt. Die einzelnen Bits also invertiert sind. Die Schalterstellung „F“ ergibt demnach den Zustand „x0000“.

Das Unterprogramm fragt zuerst das höherwertige Bit des Codierschalters ab (PORTA.3). Ist dieses gesetzt, so bedeutet dies, dass sich die Schalterstellung zwischen 0 und 7 befindet. Als nächstes wird der Portpin PORTA.2 geprüft. Ist PORTA.3 LOW, so springt es zum Label Z8bis15. Das bedeutet, dass die eingestellte Zeit zwischen Schalterposition 8 und 15 steht. Dies wird für jeden der vier Portpins durchgeführt. Ist der Codierschalter zum Beispiel in der Position 5 (dies entspricht einer eingestellten Zeit von 3 Minuten) so werden vom Programm folgenden Schritte ausgeführt. Am Port A steht folgender Zustand: x1010. Zuerst wird das höchstwertige Bit (des Codierschalters) geprüft: `btfss PORTA.3`. Dieser ist „1“, es wird also der Befehl `goto Z8bis15` übersprungen. Als nächstes erfolgt der Befehl `btfss PORTA.2`. Dieser Pin ist in unserem Beispiel „0“. Es wird daher der Befehl `goto Z4bis7` ausgeführt. Wir stehen im Programm daher auf dem Label *Z4bis7.* Nun wird mit dem Befehl *btfss PORTA.1* der Portpin mit der Wertigkeit 1 geprüft. In unserem Beispiel ist dieser wieder „1“, es wird also der Befehl `goto Z6u7` übersprungen. Als letzter Pin ist jetzt noch der niederwertigste zu testen.: `btfss PORTA,0` dieser ist „1“, folglich wird der Befehl *goto ZEIT5* ausgeführt. Nun werden mit den folgenden Befehlen die Zeitregister mit ihren Werten (Konstanten) geladen.

```
ZEIT5       movlw       ZEIT5L
            movwf       ZEITL
            movwf       AKTZEITL
            movlw       ZEIT5H
            movwf       ZEITH
            movwf       AKTZEITH
            return
```

Die Register ZEITL und AKTZEITL werden in diesem Beispiel also mit der Konstante ZEIT5L (=67d) und die Register ZEITH und AKTZEITH mit der Konstante ZEIT5H (=13d) geladen. Dies entspricht einer Zeit von etwa 1,6 Sekunden zwischen zwei Zuständen. Bei 112 Zuständen ergibt dies eine Gesamtzeit von 180 Sekunden also 3 Minuten. Mehr dazu im Kap. 5.1 (Zeitintervalle ändern)

4.4.5. NAECHSTZUST (=LED-Zustandswechsel)

Wie funktioniert nun ein Zustandswechsel? In den Registern POS1 bis POS32 stehen die Adressen der 32 Leuchtdioden, welche gerade leuchten. Es sind immer nur 32 Leuchtdioden aktiv. Bei einem Zustandswechsel muss also eine Leuchtdiode dunkel werden, und dafür eine andere leuchten. Die Adresse, jener Leuchtdiode welche erlischt, wird in den Registern POS1 bis POS32 gesucht, und durch die Adresse der neuen Leuchtdiode ersetzt. Woher weis das Programm welche Leuchtdiode erlöschen soll, und welche anstelle dieser leuchten soll? Dazu gibt es zwei Tabellen. In der Tabelle TABWIRDERSETZT stehen die Adressen der Leuchtdioden, die bei jedem Zustandswechsel erlöschen und die Tabelle TABNEUELED beinhaltet die Adressen der Leuchtdioden die bei jedem Zustandswechsel neu hinzukommen. Ein Beispiel macht diesen Zusammenhang etwas deutlicher. Hier der Übergang von Zustand 109 auf 110:

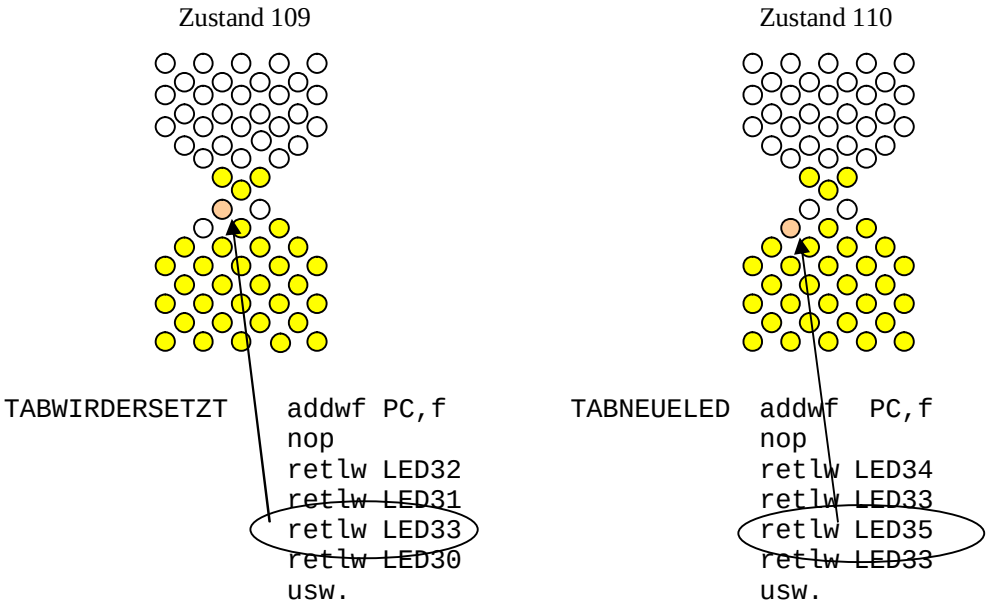


Bild 3: Zustandswechsel

Der genaue Ablauf wird wieder mit einem Flussdiagramm (auf der nächsten Seite) genauer beschrieben:

Das Register SUCHZUST wird zunächst mit POS1 geladen, im Register AKTZUSTAND steht die aktuelle Zustandsnummer von 1 bis 112. Mit Hilfe des AKTZUSTAND-Register wird nun aus der Tabelle TABWIRDERSETZT die Adresse der zu ersetzenden Leuchtdiode geholt und im Register ZWISCH zwischengespeichert. Nun beginnt der Suchvorgang. Auch hier ist uns die indirekte Adressierung behilflich. Der Registerinhalt von SUCHZUST wird nun in das FSR-Register geladen. Im IND-Register steht nun die Adresse der entsprechenden (gerade aktiven) LED, diese wird mit der Adresse, welche im Register ZWISCH steht verglichen, indem diese beide voneinander subtrahiert werden. Ist das Ergebnis der Subtraktion Null, so sind die Registerinhalte dieser beiden Register gleich. Ob das Ergebnis Null ist erkannt man daran, dass das Zero-Bit (Z) gesetzt ist. Mit Hilfe des Registers AKTZUSTAND wird nun aus der Tabelle TABNEUELED die Adresse der neuen LED geladen und in das IND-Register geladen. Im entsprechenden POSx-Register steht nun die Adresse der neuen aktiven LED. Ist das Ergebnis der Subtraktion jedoch nicht Null, so wird das Z(ero)-Bit vom Controller gelöscht. Das SUCHZUST-Register und das SUCHZAEHLER-Register werden um eins vermindert und ein erneuter Vergleichsvorgang beginnt.

Kann, aus welchem Grund auch immer, die zu suchenden LED-Adresse nicht gefunden werden, so wird mit dem SUCHZAEHLER-Register verhindert, dass irgendein anderes, zufällig passendes Register in ein POSx-Register geladen wird. Konnte die zu suchende LED-Adresse nach 32 Durchgängen nicht gefunden werden, so beginnt der gesamte Suchvorgang von vorne. Im Normalfall soll dies nicht passieren, im Entwicklungs- bzw. im Modifikations-stadium ist dies hilfreich.

Wurde also die zu suchende LED-Adresse gefunden und durch die Neue ersetzt, wird als nächstes das AKTZUSTAND-Zählregister um eins vermindert, damit beim nächsten Aufruf von NAECHSTZUST der nächste Zustand hergestellt werden kann. Wird dieses Register Null, ist die eingestellte Zeit abgelaufen. Das Fertigflag (SUSTATUS.2) wird gesetzt, und das LED-rieseln-Flag (SUSTATUS.0) wird gelöscht.

Der Zustandswechsel ist jetzt abgeschlossen (egal welchen Wert das AKTZUSTAND-Zählregister hat), das NächstZust-Flag wird daher wieder gelöscht.
Optional kann bei jedem Zustandswechsel ein Beepton erfolgen. Dazu muss der Jumper JP1 gesteckt sein. Softwaremäßig wird dies mit dem Befehl *btfsc PORTA,SUMMERSELEKT* abgefragt. Ist der Jumper gesteckt, so ist der Portpin RA4 low und die folgenden Befehle werden ausgeführt:

```
bsf   PORTB,SUMMER
call  WARTESCHLEIFE
bcf   PORTB,SUMMER
```

Es wird also der Lautsprecher für eine Zeit von 500µs (vgl. Kap. 4.4.7 WARTESCHLEIFE) eingeschaltet und danach wieder ausgeschaltet. Dies ergibt dann einen kurzen Beep-Ton.

Abgeschlossen wird das Unterprogramm (wie üblich und notwendig) mit dem *return*-Befehl.

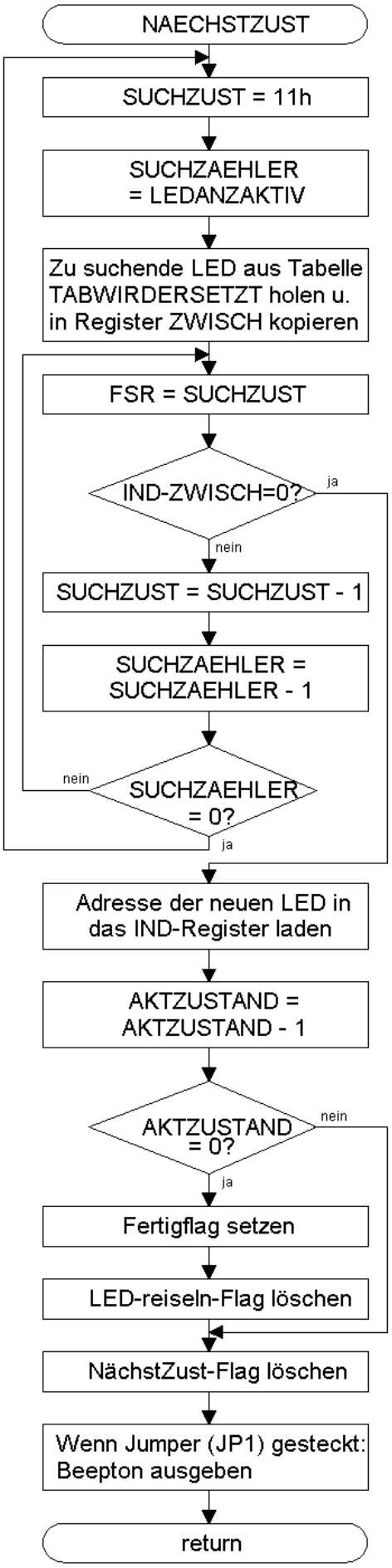


Bild 4: Flussdiagramm Zustandswechsel

4.4.6. SUMMERAKTIV

Dieses Unterprogramm erzeugt ein Rechtecksignal mit einer Frequenz von etwa 1 kHz. Dazu wird der Lautsprecher 500µs lang eingeschaltet und 500µs lang ausgeschaltet.

```
bsf   PORTB,SUMMER
call  WARTESCHLEIFE
bcf   PORTB,SUMMER
call  WARTESCHLEIFE
return
```

Diese Unterprogramm wird so oft aufgerufen bis die Starttaste erneut gedrückt, oder die Stopptaste gedrückt wird (vgl. Hauptprogramm Kap.4.2)

4.4.7. WARTESCHLEIFE

Dieses Unterprogramm erzeugt eine Zeitverzögerung von exakt 500 µs bei einem 4 MHz-Quarz.

Wie erzeugt man genau 500 µs? Dazu sehen wir uns zunächst den Code genauer an:

```
                movlw       .164              (1)
                movwf       DELAY             (1)
VERZOEGERUNG    decfsz      DELAY,f           (1 bzw. 2)
                goto        VERZOEGERUNG      (2)
                nop                           (1)
                return                        (2)
```

In den Klammer befinden sich die Anzahl der Zyklen die diese Befehle benötigen. Eine Besonderheit gibt es beim Befehl `decfsz DELAY,f`. Solange dieser Befehl nur die angegebene Variable (hier DELAY) dekrementiert benötigt er nur einen Zyklus. Ist die Variable DELAY 0, überspringt sie den folgenden Befehl. In diesem Fall benötigt der Befehl 2 Zyklen.
Die Schleife *VERZOEGERUNG* benötigt 3 Zyklen (`decfsz DELAY,f` und `goto VERZOEGERUNG`) und wird 164-mal durchgeführt. 3x164=492. 1 Zyklus kommt für den Sprung bei DELAY= 0 hinzu, ebenso die Zyklen für die restliche Befehle. Nun sind wir bei 498 Zyklen. Zählt man noch 2 Zyklen für den Aufruf (`CALL WARTESCHLEIFE`) dazu, so ergibt dies die gewünschten 500 Zyklen, und bei einem Takt von 1µs (bei einem 4 MHz-Quarz) ergibt dies genau 500 µs.

5. Modifikationshinweise

Die von mir gewählten Parameter sind eigenen Bedürfnissen anpassbar. Die Zeitauswahl wurde von mir so gewählt, dass sie einen weiten Bereich abdeckt. Von 15 Sekunden bis 1 Stunde. Dass, bei nur 16 Einstellmöglichkeiten nur die „wichtigsten“ Zeiten einstellbar sind liegt daher klar auf der Hand. Dies soll aber kein Nachteil sein, denn diese sind leicht änderbar. Wird nur ein Bereich von 1 bis 10 Minuten benötigt, so lässt sich dies, mit ein wenig Rechenarbeit, realisieren.

Vorraussetzung dafür ist aber, dass Sie die Möglichkeit einer Entwicklungsumgebung und einer Programmiermöglichkeit für den PIC-Baustein haben.

Nicht nur, was die Zeiteinstellung betrifft, auch die Reihenfolge wie die einzelnen Leuchtdioden nach unten rieseln lässt sich verändern, und es müssen auch nicht unbedingt 112 Schritte (Zustände) sein. Die Anzahl der Schritte ergibt sich zwangsläufig aufgrund der Reihenfolge des Rieselns. Die einzige Bedingung dafür ist aber, dass sich immer nur *eine* Leuchtdiode pro Zustandsänderung ändert, ansonst müssen Sie sich eine andere Strategie erarbeiten und das Unterprogramm „NAECHSTZUST“ entsprechen abändern.

Und wer meint es müssen immer 32 Leuchtdioden leuchten, der irrt! Es ist durchaus möglich, dass auch weniger leuchten. Sollen mehr als 32 leuchten, so sind umfangreichere Programmänderungen nötig. Abgesehen davon, ist dies auch „unrealistisch“!

Nun aber zu den einzelnen Änderungsmöglichkeiten im Detail:

5.1. Zeitintervalle ändern

Die Berechnung der Konstanten für die Zeiteinstellung soll am Beispiel von Schalterstellung 0 des HEX-Codierschalters S1 erläutert werden. In dieser Schalterstellung dauert die verrieselnde Zeit 15 Sekunden. Die entsprechenden Konstanten heißen *ZEIT0H* und *ZEIT0L*. Da es sich hier um einen 16-Bit-Wert handelt, muss dieser auf zwei Register aufgeteilt werden. Die Ziffer 0 steht für die Schalterstellung. Für die Schalterstellung 1 (30 Sekunden) lauten daher die entsprechenden Konstanten *ZEIT1H* bzw. *ZEIT1L* und bei Schalterstellung F (60 Minuten) heißen die Konstanten *ZEIT15H* bzw. *ZEIT15L*. Es müssen **nur** diese Konstanten angepasst werden!

Wie werden nun die Konstanten ermittelt, wenn die verrieselnde Zeit 15 Sekunden betragen soll?

Die Konstanten (ZEITxH und ZEITxL) beinhalten die Anzahl der ISR-Aufrufe zwischen zwei Zuständen. Es gibt hier insgesamt **112** Zustände. Die ISR wird alle 512 µs aufgerufen. Dieser Wert ergibt sich dadurch, da das TMR0-Register mit 0 geladen wird (es also wieder 256 µs dauert bis dieses Register wieder 0 wird und einen Überlauf generiert) und der Vorteiler mit 2 geladen wurde (OPTION-Register: b‘00000000‘) ,vgl. Kap 4.4.1

Es ergibt sich daher folgende Formel:

$$\text{Einstellwert} = \frac{\text{gewünschte Zeit in } \mu\text{s}}{\text{Anz. der Zustände} \cdot \text{N} \cdot \text{V} \cdot \text{TP}}$$

Einstellwert = berechnete 16-Bit Konstante
Gewünschte Zeit in µs = Hier 15000000 µs (=15 Sekunden)
Anz. der Zustände = 112
N = Zahl der Schritte bis zum TMR0-Überlauf (hier 256)
V = Vorteilerwert (hier 2)
TP = Taktperiode (1µs bei einem 4 MHz-Quarz)

Konkret für 15 Sekunden:

$$\text{Einstellwert} = \frac{15000000\ \mu\text{s}}{112 \cdot 256 \cdot 2 \cdot 1\ \mu\text{s}} = 261{,}579 \Rightarrow \text{Einstellwert} = 261$$

Dieser 16-Bit-Wert muss jetzt nur noch in zwei 8-Bit-Werte umgerechnet werden. Das höherwertige muss zusätzlich noch um 1 erhöht werden.

$$\frac{261}{256} = 1{,}019 \Rightarrow \text{Höherwertiges Byte} = 2$$

$$261 - 1 \cdot 256 = 5 \Rightarrow \text{Niederwertiges Byte} = 5$$

Die Konstanten ZEIT0H und ZEIT0L besitzen daher die Werte 2 bzw. 5 (vgl. Listing)

Die einzelnen Zeiten:

Schalterpos.	Ges.Zeit	Einstellwert (16Bit)	ZEITxH	ZEITxL
0	15 Sekunden	261	2	5
1	30 Sekunden	523	3	11
2	60 Sekunden	1046	5	22
3	90 Sekunden	1569	7	33
4	2 Minuten	2092	9	44
5	3 Minuten	3139	13	67
6	4 Minuten	4185	17	89
7	5 Minuten	5231	21	111
8	8 Minuten	8370	33	178
9	10 Minuten	10463	41	223
A	12 Minuten	12555	50	11
B	15 Minuten	15695	62	79
C	20 Minuten	20926	82	190
D	30 Minuten	31389	123	157
E	45 Minuten	47084	184	236
F	60 Minuten	62779	246	59

Anmerkung: Bei einer Änderung der Zeiten sollte auch die Frontfolie dementsprechend angepasst werden. Zu diesem Zweck steht die Datei sanduhr.**fpl** zur Verfügung. Diese wurde mit dem Programm „Fronplatten-Designer 2.0“ der Firma ABACOM erstellt.

5.2. „Riesel“-Reihenfolge der LEDs ändern

Die Reihenfolge, wie die oberen Leuchtdioden nach unten rieseln wird mit zwei Tabellen beschrieben. (siehe auch Kap. 4.4.5). In der Tabelle TABWIRDERSETZT stehen nacheinander die LED-Nummern, welche bei einem Zustandswechsel dunkel werden und in der Tabelle TABNEUELED stehen die LED-Nummern, welche jetzt neu dazukommen. Wichtig: es wird bei einem Zustandswechsel immer nur ***eine*** Leuchtdiode ausgetauscht! Die restlichen 31 Leuchtdioden bleiben leuchtend. Zu beachten ist, dass die Tabelle unten beginnt. Beim ersten Zustandswechsel (der Aufruf des Unterprogramms ZUSTAND1, nachdem die Starttaste gedrückt wurde, gilt hier nicht als Zustandswechsel!) wird also die Leuchtdiode LED3 dunkel (siehe Tabelle TABWIRDERSETZT) und die Leuchtdiode LED36 leuchtet nun anstelle von LED3. Beim zweiten Zustandswechsel wird LED36 wieder dunkel, dafür leuchtet nun LED44, beim dritten Zustandswechsel erlischt LED44 und es leuchtet nun LED53, usw. (vgl. Listing)
Der Befehl `nop` gleich nach dem `addwf PC,f` - Befehl darf bei den beiden Tabellen nicht entfernt werden!

Hier die Nummerierung der Leuchtdioden:

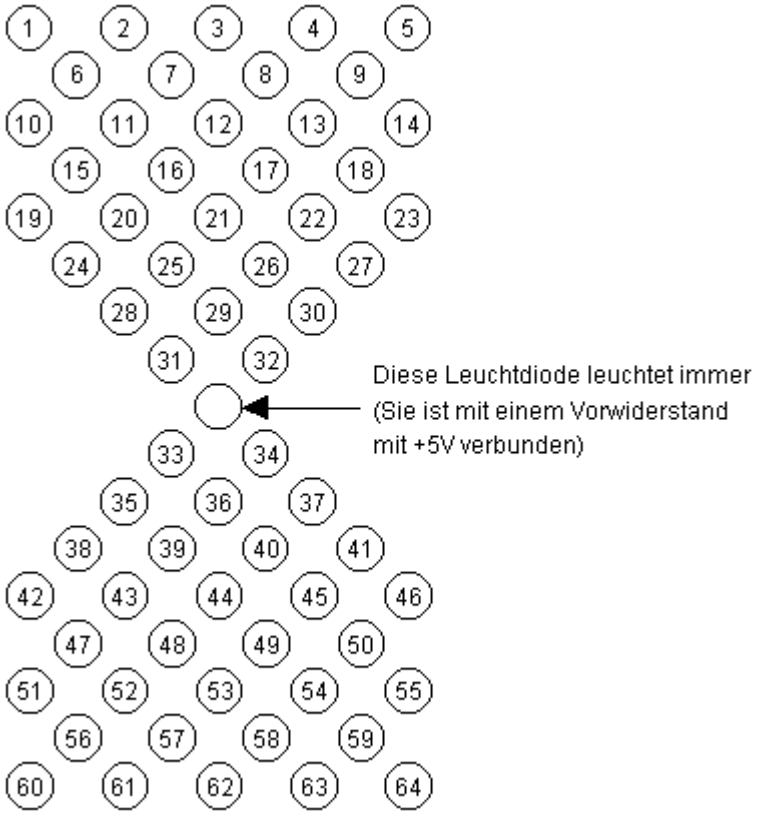


Bild 5: Nummerierung der Leuchtdioden

Weiters muss die Konstante ZUSTANZ (im Listing) angepasst werden. Diese enthält die Anzahl der Zustände. In der aktuellen Version hat sie den Wert 112 (da es ja 112

Zustände gibt). Die Konstanten für die Zeiteinstellung (ZEIT0L bis ZEIT15H) müssen ebenfalls neu berechnet werden (vgl. Kap. 5.1)

5.3. Anzahl der leuchtenden LEDs ändern

Auch die Anzahl der leuchtenden LEDs kann angepasst werden. Es können auch weniger als 32 LEDs leuchten. Mehr als 32 ist zwar möglich, wenn man zusätzliche POSx-Register definiert, aber meiner Meinung bei insgesamt 64 Leuchtdioden nicht besonders sinnvoll!

Änderungen:

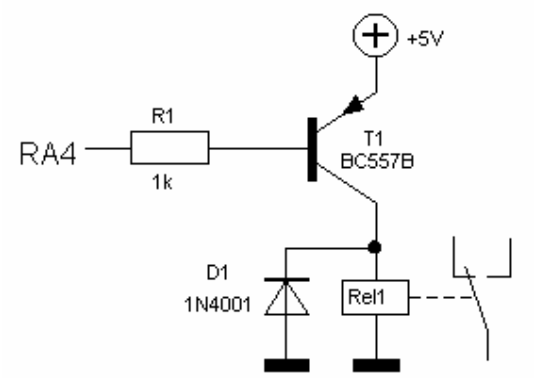
- Die Konstante LEDANZAKTIV mit der gewünschten Anzahl laden
- Die Unterprogramme ZUSTAND0 und ZUSTAND1 anpassen (nur die gewünschte Anzahl der POSx-Register mit den entsprechenden LED-Adressen laden)
- Die Tabellen TABWIRDERSETZT und TABNEUELED anpassen, und falls sich dadurch die Anzahl an Zuständen ändert auch die Konstante ZUSTANZ und die Konstanten für die Zeiteinstellung (ZEIT0H bis ZEIT15L)

Wichtig:
Bei einem Zustandswechsel wird die zu ersetzende LED in den POSx-Registern gesucht, wird diese dort nicht gefunden (Fehler in den Tabellen!!) so bleibt die Sanduhr buchstäblich stehen, und es erfolgt kein Zustandswechsel mehr!! Wird z.B. die Konstante LEDANZAKTIV mit dem Wert 20 geladen, so wird die zu ersetzende Leuchtdiode auch nur in den Registern POS1 bis POS20 gesucht, wird sie dort nicht gefunden, steht die Sanduhr!

5.4. Schaltausgang anstelle Jumper JP1

In vielen Fällen kann auf den Jumper JP1 verzichtet werden, wodurch ein I/O-Pin des PIC-Controller (IC1) frei wird und für eine zusätzliche Funktion verwendet werden kann. Für manche Anwendungen der Sanduhr ist ein Schaltausgang für die Aktivierung/Deaktivierung eines anderen Gerätes während oder nach dem Rieseln notwendig. Dies läst sich nun mit dem freigewordenen Pin realisieren, wobei natürlich Anpassungen sowohl in der Software als auch in der Hardware notwendig sind:

Hardware:
Den Pull-Up-Widerstand R2 entfernen und an den Controller-Pin RA4 z.B. folgende Transistor-Schaltstufe hinzufügen.



R1: 1k
T1: BC557B
D1: 1N4001

Bild 6: Schaltausgang

Der hier gewählte Transistor (BC557B) kann einen maximalen Kollektorstrom von 100mA liefern. Wird für das verwendete Relais mehr benötigt, so muss ein entsprechender Transistor verwendet werden.

Software:
Anhang E zeigt den Sourcecode der geänderten Version (SaUhr1v2.asm)

Geändert bzw. Neu sind:

Konstanten:

BEEPMODE: 1 = Beep bei jedem Zustandswechsel der LEDs
0 = kein Beep bei Zustandswechsel

SAUSMODE: 0 = kein Schaltausgang
1 = Beim Drücken der Starttaste wird der Schaltausgang aktiv (low); ist die eingestellte Zeit verrieselt oder mit der Stopptaste geht der Schaltausgang wieder in den inaktiven Zustand (high) zurück
2 = Der Schaltausgang geht in den aktiven Zustand (low), wenn die eingestellte Zeit verrieselt ist und geht wieder in den inaktiven Zustand (high) zurück, wenn entweder erneut die Starttaste gedrückt wird, oder durch Drücken der Stopptaste.
3 = Der Schaltausgang geht, nachdem die eingestellte Zeit verrieselt ist für eine kurze Dauer in den aktiven Zustand (low) über. (Dieser Mode erzeugt also einen Triggerimpuls). Die Impulsdauer ist dabei mit einer Konstanten einstellbar.

SAUSDAUER: Diese Konstante gibt die Impulsdauer x 500µs an, wenn die Konstante SAUSMODE den Wert 3 besitzt.

Unterprogramm INIT: Port A muss wie folgt definiert werden:
Pins RA0 bis RA3: Eingänge
Pin RA4: Ausgang
è Das Register TRISA muss demnach mit dem Wert b‘01111‘ geladen werden. Weiters muss der Schaltausgang mit HIGH vorbelegt werde

Unterprogramm NAECHSTZUST:
Die Abfrage ob der Jumper gesteckt ist entfällt, stattdessen wird anhand der Konstante BEEPMODE ein Beepton bei einem Zustandswechsel der Leuchtdioden ausgegeben.

Unterprogramm SAUSBEISTART: (Dieses Unterprogramm kommt hinzu)
Nach Drücken der Starttaste wird der Schaltausgang je nach Modus gesetzt oder rückgesetzt

Unterprogramm SAUSNACHZEIT: (Dieses Unterprogramm kommt hinzu)
Nach Ablauf der eingestellten Zeit wird der Schaltausgang je nach Modus gesetzt oder rückgesetzt

Hauptprogramm: Die Unterprogramm-Aufrufe für SAUSBEISTART und SAUSNACHZEIT an den entsprechenden Positionen

6. Nachbauhinweise

Schritt 1: (Lochraster)-Platinen bestücken

Die Elektronik dieses Projektes wurde auf mehrere Lochrasterplatinen aufgeteilt:

- Anzeige- und Bedienplatine
- Steuerplatine
- Stromversorgungsplatine

Das Bestücken einer Platine ist erst dann sinnvoll, wenn alle für diese Platine benötigten Bauteile vorhanden sind. Es sollten generell nur erstklassige und neuwertige Bauteile verwendet werden. Auf Bauteile aus ausgeschlachteten Geräten sollte grundsätzlich verzichtet werden, da ihre Funktionalität nicht gewährleistet ist, und eine unnötige Fehlersuche dadurch vermieden werden kann.

Weiters sollte ausreichend Platz und vor allem ausreichend Zeit für die Bestückung der Platinen vorhanden sein.

Anzeige- und Bedienplatine:

Die Bauelemente entsprechend Bild 7, der folgenden Reihenfolge, der Stückliste (Anhang D), und den Schaltplänen (Anhang A) bestücken wobei zunächst nur die Bauteile angelötet werden. Diese jedoch noch nicht verdrahten. Das Verdrahten erfolgt erst wenn alle Bauteile angelötet sind. Die nach dem anlöten überstehenden Anschlüsse mit einem kleinen Seitenschneider entfernen.

Reihenfolge zur Bearbeitung und Bestückung der Bedien- und Anzeigeplatine:

- 3-mm-Bohrungen für die Montage der Platine an der Frontplatte (4 Bohrungen) und für die Montage der Steuerplatine (4 Bohrungen)
- Leuchtdioden D1 bis D65 (gelb, diffus, 5 mm): **Achtung:** Polarität beachten! (der längere Anschluss ist die Kathode (plus), der kürzere demnach die Anode (minus). **Tipp:** Zuerst nur jeweils einen Anschluss der Leuchtdioden anlöten, Leuchtdioden anschließend ausrichten, und erst wenn sie mit der Anordnung der Leuchtdioden zufrieden sind denn jeweils zweiten Anschluss der Leuchtdioden anlöten.
- Taster S2 und S3 (Impulstaster, rund): **Achtung:** abgeflachte Seite beachten!
- HEX-Codierschalter (S1): **Achtung:** Einbaulage beachten!

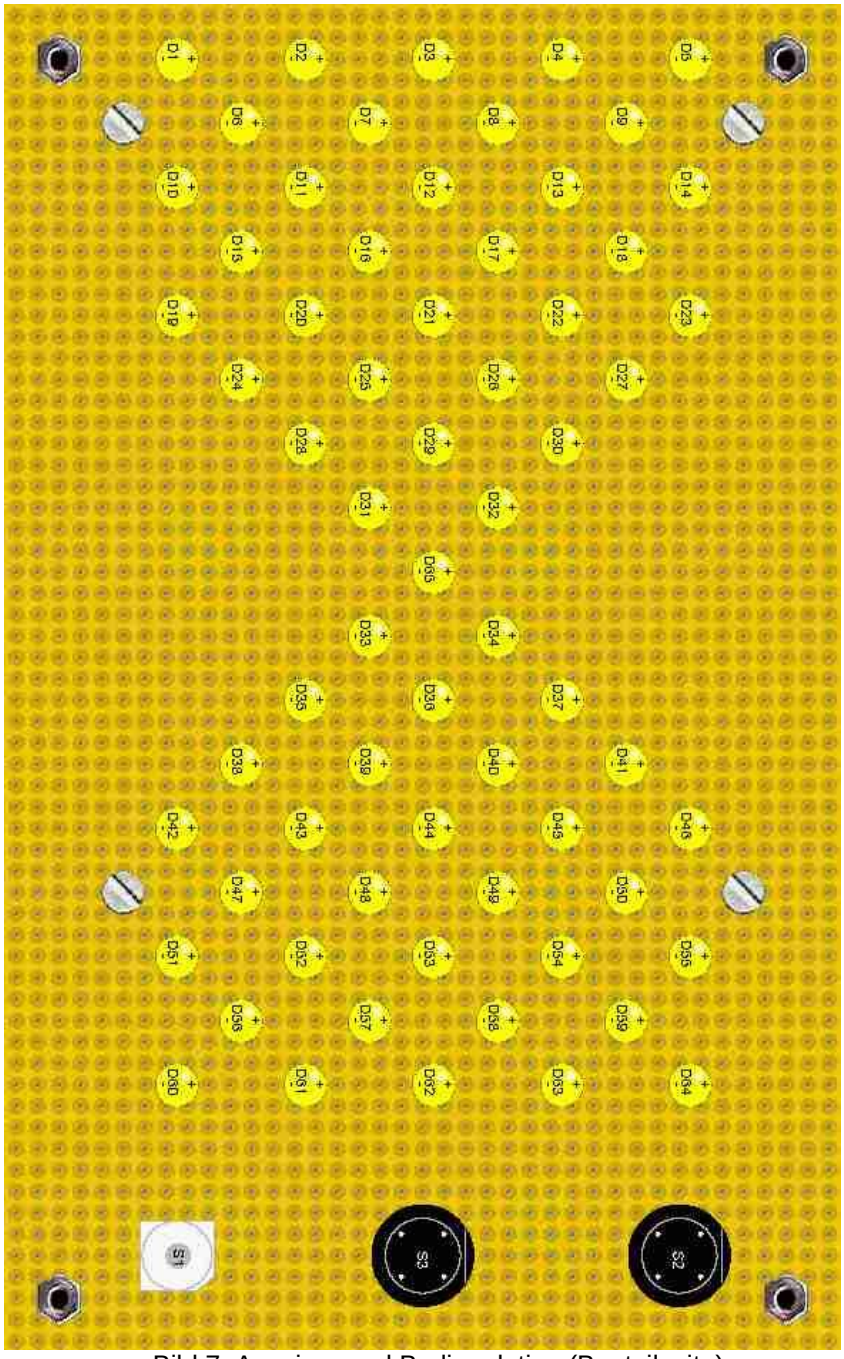


Bild 7: Anzeige- und Bedienplatine (Bauteilseite)

Nachdem alle Bauteile eingelötet wurden, die Verbindungen auf der Lötseite entsprechend Bild 8 mit dünnem Draht bzw. mit isoliertem Draht herstellen. Dabei sollte sorgfältigst gearbeitet werden, und man sollte sich dafür ausreichend Zeit nehmen. Befinden sich Verbindungen direkt nebeneinander, so können diese auch nur mit Lötzinn verbunden werden. Tipp: Bild 8 ausdrucken, und während dem Verlöten die soeben durchgeführte Verbindung im Ausdruck kennzeichnen. Auf diese Weise können Fehler durch fehlende Verbindungen vermieden werden.
Zum Schluss alle Verbindungen und Lötstellen noch einmal sorgfältig überprüfen. (Auch hierfür sollte man sich ausreichend Zeit nehmen!)

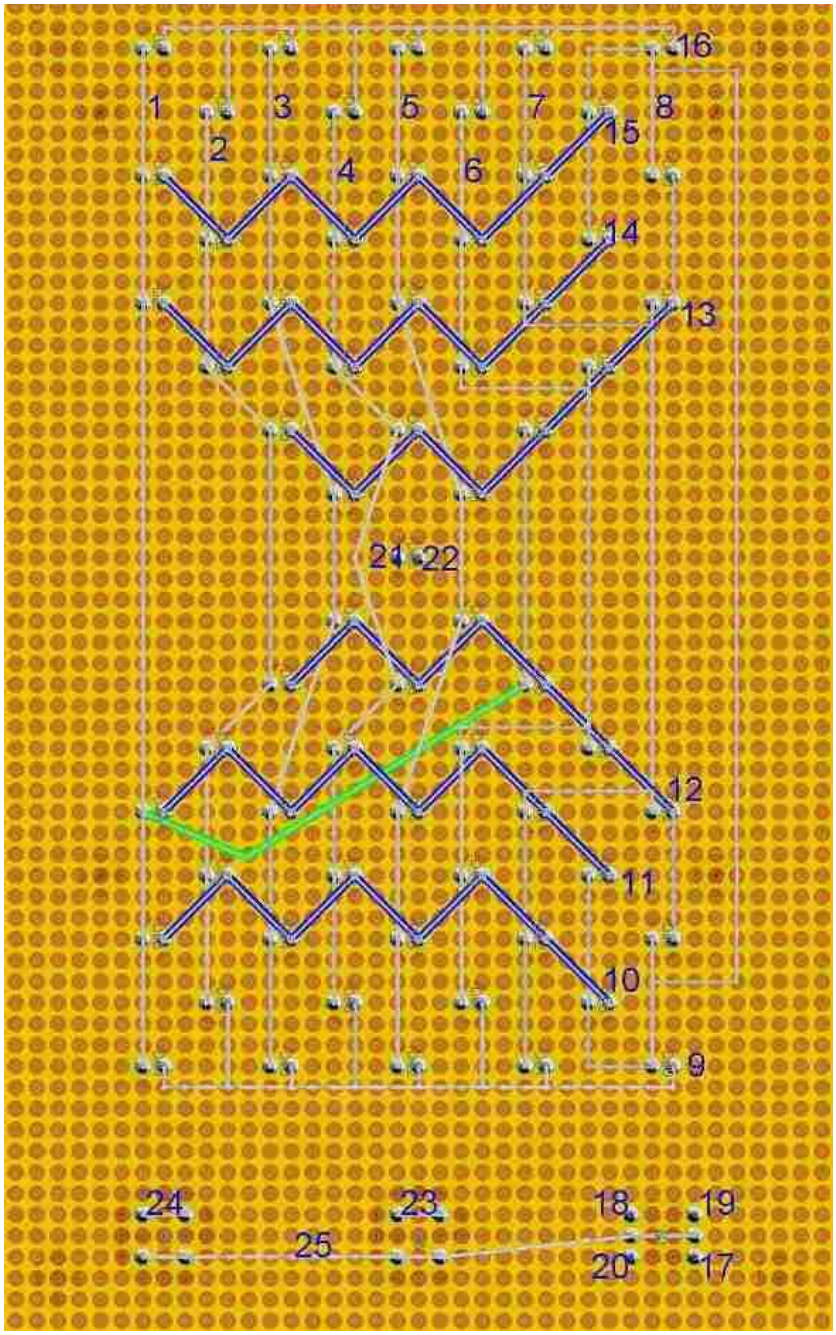


Bild 8: Anzeige- und Bedienplatine (Lötseite)

Steuerplatine:

Die Bauelemente entsprechend Bild 9, der folgenden Reihenfolge, der Stückliste (Anhang D), und den Schaltplänen (Anhang A) bestücken wobei zunächst nur die Bauteile angelötet werden. Diese jedoch noch nicht verdrahten. Das Verdrahten erfolgt erst wenn alle Bauteile angelötet sind. Die nach dem anlöten überstehenden Anschlüsse mit einem kleinen Seitenschneider entfernen.

Reihenfolge zur Bearbeitung und Bestückung der Steuerplatine:

- Platine auf eine Größe von 100 x 81 mm zuschneiden und die geschnittenen Flächen abschleifen
- 3-mm-Bohrungen für die Montage dieser Platine an der Anzeige- und Bedienplatine
- 16 Drahtbrücken: **Achtung:** Bei den Transistoren T1 bis 16 (BC557B) befinden sich 14 der 16 Drahtbrücken.

- Widerstände R2 (10k), R3 (680 Ohm), R4 (10k), R5 (270 Ohm), R6 (4k7), R7 bis R15 (je 10k), R16 bis R23 (je 100k), R24 bis R31 (je 10k) und R32 bis R39 (je 100k): **Tipp**: Vor dem Einlöten des Widerstandes diesen überprüfen, auch wenn die Bauteile in einem Regal sortiert sind. (z.B. mit einem Multimeter). Die Praxis hat gezeigt, dass sich hin und wieder doch falsche Bauteilwerte in das Regal eingeschlichen haben. Dies gilt nicht nur für Widerstände, sondern auch für Dioden, Kondensatoren, Transistoren usw.
- IC-Fassungen für IC1 (18polig), IC2 und IC3 (je 16-polig): **Tipp 1:** Obwohl es bei den Fassungen elektrisch gesehen egal ist wie die Fassungen eingelötet sind, sollte man doch die Fassung so einlöten, dass die Kerbe in die richtige Richtung zeigt. Dies erleichtert das spätere Einsetzten der ICs bzw. erleichtert die Arbeit bei einem IC-Tausch. **Tipp 2:** Beim Einlöten der Fassungen sollte man wie folgt vorgehen: Fassung an der einzusetzenden Stelle lagerichtig einsetzen und zunächst nur einen beliebigen Eckpin anlöten. Fassung kontrollieren und eventuell Nachlöten. Sitzt die Fassung ordentlich, den gegenüberliegenden Pin anlöten. Fassung wieder kontrollieren und eventuell nachlöten. Erst wenn Sie mit der Lage der Fassung zufrieden sind, die restlichen Pins anlöten. **Achtung:** Bei der Fassung für IC1 (18polig) den mittleren Steg vorsichtig entfernen.
- Anstelle einer IC-Fassung können im Notfall auch Buchsenleisten verwendet werden. Hier empfiehlt sich wegen der fehlenden Kerbe diese mit einem Filzstift oä. auf der Platine zu markieren.
- Widerstandsarray R1 (4x10k): **Achtung:** Polarität beachten! Anstelle des Arrays können auch 4 Einzelwiderstände (je 10k) verwendet werden. In Bild 3 sind beide Varianten eingezeichnet!
- Keramikkondensatoren C6 und C7 (je 22pF)
- Keramikkondensatoren C3 bis C5 (je 100nF): **Achtung:** C3 befindet sich unter IC1 (bzw. in dessen IC-Fassung)
- Transistoren T1 bis T16 (BC557B) und T17 (BC547B): **Achtung:** abgeflachte Seite beachten!
- 2polige Stiftleiste (JP1)
- Elko C8 (10µF/25V): **Achtung:** Polarität beachten
- Quarz X1 (4 MHz)

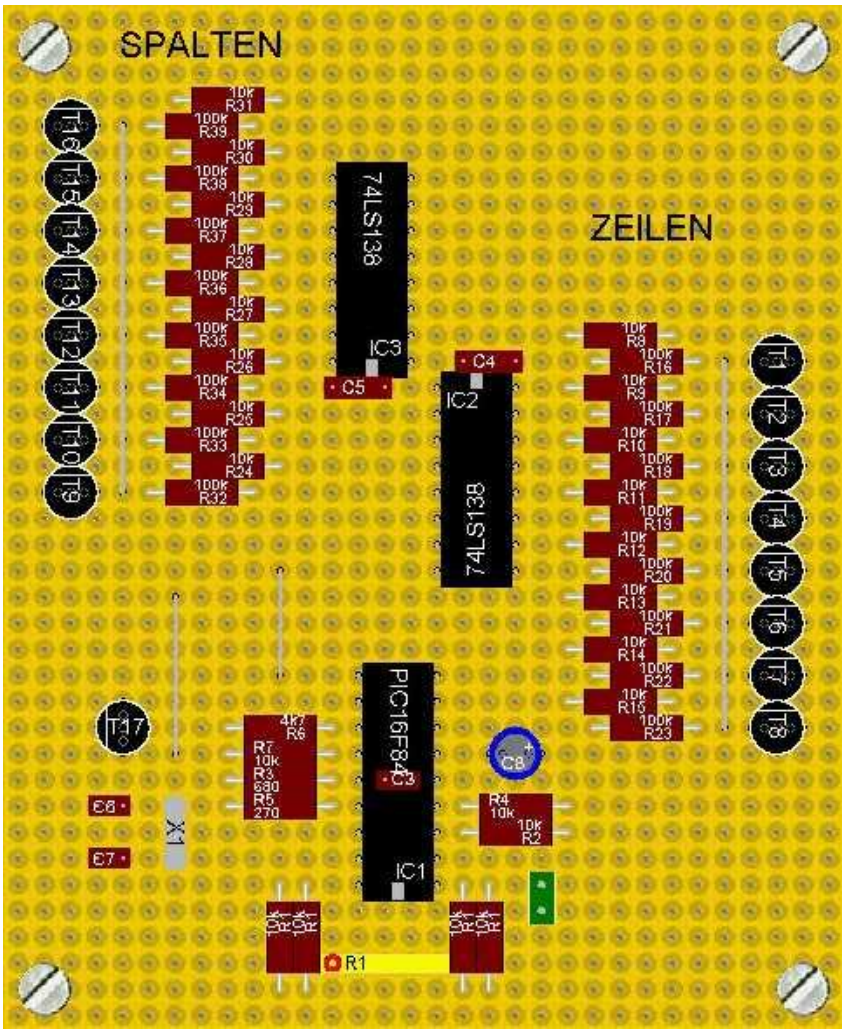


Bild 9: Steuerplatine (Bauteilseite)

Nachdem alle Bauteile eingelötet wurden, die Verbindungen auf der Lötseite entsprechend Bild 10 mit einem dünnen Draht bzw. mit isolierten Drähten herstellen. Dabei sollte sorgfältigst gearbeitet werden, und man sollte sich dafür ausreichend Zeit nehmen. Befinden sich Verbindungen direkt nebeneinander, so können diese auch nur mit Lötzinn verbunden werden. **Tipp:** Bild 10 ausdrucken, und während dem Verlöten die soeben durchgeführte Verbindung im Ausdruck kennzeichnen. Auf diese Weise können Fehler durch fehlende Verbindungen vermieden werden.
Zum Schluss alle Verbindungen und Lötstellen noch einmal sorgfältig überprüfen. (Auch hierfür sollte man sich ausreichend Zeit nehmen!)

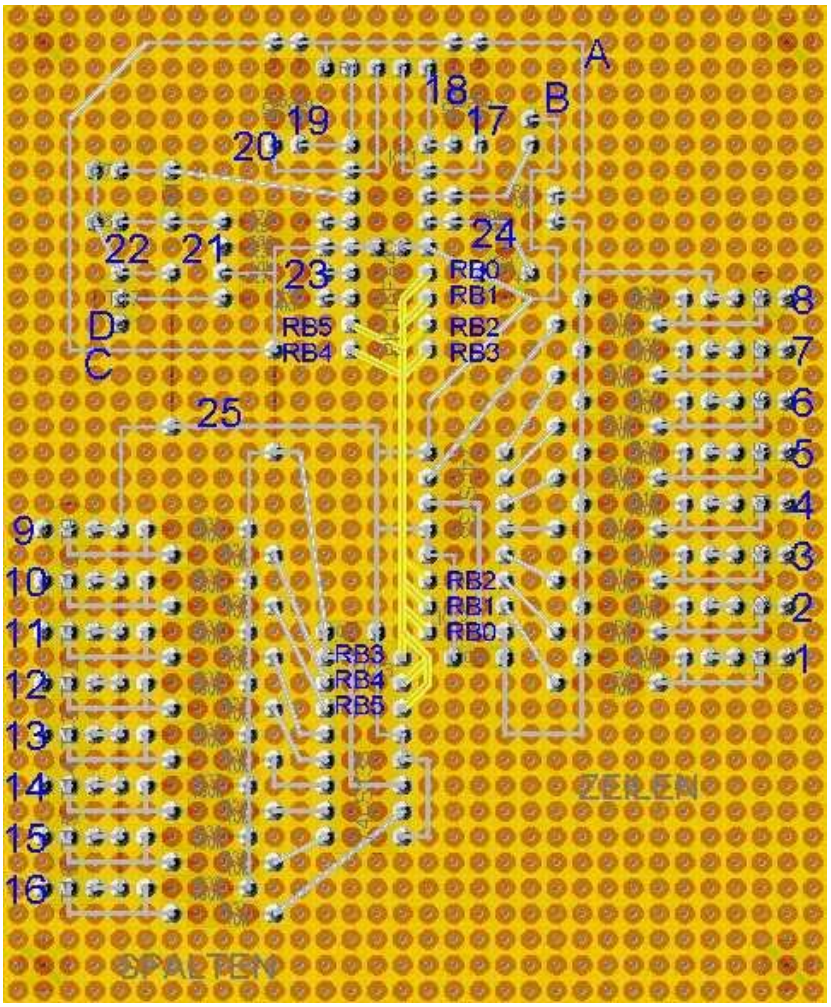


Bild 10: Steuerplatine (Lötseite)

Stromversorgungsplatine:

Die Bauelemente entsprechend Bild 11, der folgenden Reihenfolge, der Stückliste (Anhang D), und den Schaltplänen (Anhang A) bestücken wobei, zunächst nur die Bauteile angelötet werden. Diese jedoch noch nicht verdrahten. Das Verdrahten erfolgt erst wenn alle Bauteile angelötet sind. Die nach dem anlöten überstehenden Anschlüsse mit einem kleinen Seitenschneider entfernen.

Reihenfolge zur Bearbeitung und Bestückung der Steuerplatine:

- Platine auf eine Größe von 59 x 46 mm zuschneiden und die geschnittenen Flächen abschleifen
- 3-mm-Bohrungen für die Montage der Platine und des Batteriehalters
- Diode D66 (1N4001): **Achtung:** Polarität beachten!
- Keramikkondensatoren C2 (100 nF)
- Spannungsregler IC4 (78L05): **Achtung:** Polarität beachten!
- Elko C1 (10µF/25V): **Achtung:** Polarität beachten!
- Printbuchse
- Kippschalter S4
- Batteriehalter für 9-V-Block: Batteriehalter zunächst mit 2 Senkkopfschraube M3 x 10 und Mutter M3 befestigen und anschließend die Kontakte löten

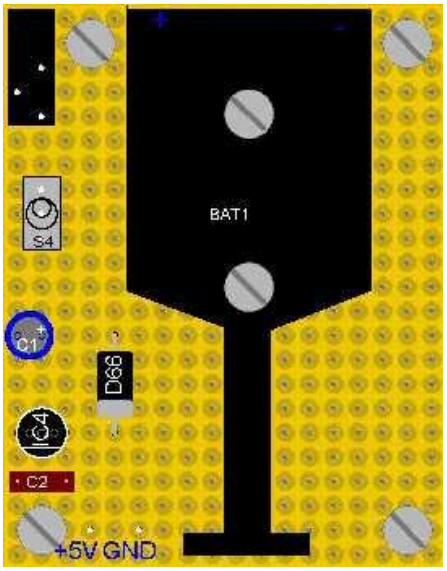


Bild 11: Stromversorgungsplatine (Bauteilseite)

Nachdem alle Bauteile eingelötet wurden, die Verbindungen auf der Lötseite entsprechend Bild 12 mit einem dünnen Draht herstellen. Dabei sollte sorgfältigst gearbeitet werden, und man sollte sich dafür ausreichend Zeit nehmen. Befinden sich Verbindungen direkt nebeneinander, so können diese auch nur mit Lötzinn verbunden werden. **Tipp:** Bild 12 ausdrucken, und während dem Verlöten die soeben durchgeführte Verbindung im Ausdruck kennzeichnen. Auf diese Weise können Fehler durch fehlende Verbindungen vermieden werden.
Zum Schluss alle Verbindungen und Lötstellen noch einmal sorgfältig überprüfen. (Auch hierfür sollte man sich ausreichend Zeit nehmen!)

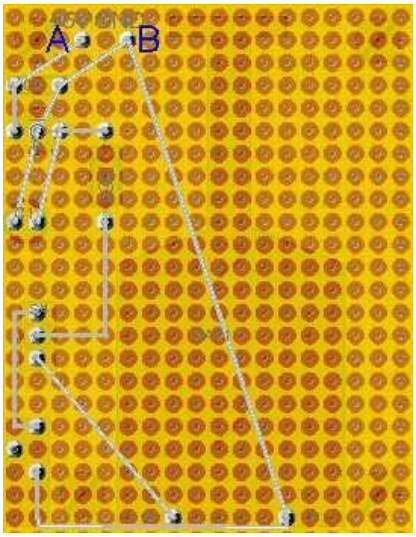


Bild 12: Stromversorgungsplatine (Lötseite)

Schritt 2: (Lochraster)-Platinen miteinander verbinden

Die elektrischen Verbindung der Platinen miteinander erfolgt mit isolierten Drähten. Dazu eignen sich z.B. die Adern eines bunten Flachbandkabels. Es sollten verschiedene Farben verwendet werden. Dies erleichtert eine mögliche Fehlersuche. Weiters sollten diese Drähte nicht zu kurz gewählt werden.

Anzeige- und Bedienplatine mit der Steuerplatine verbinden:

Hier sind 25 Verbindungen nötig. Diese sind mit den Zahlen 1 bis 25 auf den Lötseiten dieser beiden Platinen gekennzeichnet.

Tipp: Da diese Verbindungen unterschiedlich lange sind sollte wie folgt vorgegangen werden. Die zu verbindenden Platinen nebeneinander legen, wobei bei beiden Platinen die Lötseite nach oben zeigt. (Die Steuerplatine sollte dabei links von der Anzeige- und Bedienplatine liegen). Ein Ende eines Drahtes ca. 2 mm abisolieren, Litzen verdrillen und verzinnen. Dieses Ende an z.B. Nr. 1 der Anzeige- und Bedienplatine anlöten. Die Länge dieses Drahtes ermitteln, wobei eine Reserve von einigen Zentimetern hinzugefügt werden soll. Draht mit einem kleinen Seitenschneider trennen. Dieses (zweite) Drahtende wieder ca. 2 mm abisolieren, Litzen verdrillen, verzinnen und an die mit 1 gekennzeichnete Position auf der Steuerplatine anlöten. Diese Schritte für die restlichen Verbindungen wiederholen.

Steuerplatine mit der Stromversorgungsplatine verbinden:

Hier sind 2 Verbindungen nötig (+5V und GND). Diese sind mit den Buchstaben A und B auf den Lötseiten dieser beiden Platinen gekennzeichnet. Die Drahtlänge sollte ca. 10 cm betragen. Es ist sinnvoll für die +5V-Leitung (Buchstabe A) einen roten Draht und für die GND-Leitung (Buchstabe B) einen schwarzen Draht zu verwenden. Dies erleichtert möglicherweise eine spätere Fehlersuche!

Lautsprecher mit Steuerplatine verbinden:

Hier sind 2 Verbindungen nötig. Diese sind mit den Buchstaben C und D auf der Lötseite der Steuerplatinen gekennzeichnet. Die Drahtlänge sollte ca. 14 cm betragen. Hier sind die Farben der Drähte beliebig. Auch die Anschlussfolge am Lautsprecher ist beliebig.

Schritt 3: Tests

- In die fertig bestückte Steuerplatine die ICs (IC1 bis IC3) noch **nicht** einsetzen.
- Zuerst mit einem Multimeter prüfen, ob zwischen Betriebsspannung und Masse kein Kurzschluss herrscht. (Multimeter im Mode „Durchgangstester“ an den Verbindungs-leitungen zwischen Steuerplatine und Stromversorgungsplatine). Ist kein Kurzschluss feststellbar, als nächstes am z.B. IC-Sockel für IC1 an den Pins 5 (GND) und 14 (Ub) messen. Diese Messung kann auch am IC-Sockel für IC2 oder IC3 an den Pins 8 (GND) und 16 (Ub) durchgeführt werden. Multimeter nach wie vor im Mode „Durchgangstester“. Eventuell festgestellte Kurzschlüsse müssen natürlich aufgespürt und entfernt werden!
- Kippschalter (S4) in Mittelposition bringen
- Steckernetzteil auf 9V einstellen und mit einem passenden Gegenstück in die Buchse der Stromversorgungsplatine anstecken. Steckernetzteil am Netz an-stecken.
- Sanduhr mit den Kippschalter (S4) in Richtung Printbuchse einschalten – Nur die mittlere Leuchtdiode (D65) auf der Anzeige- und Bedienplatine muss aufleuchten. Ist dies nicht der Fall, so wurde vermutlich beim Steckernetzteil der der Stecker verkehrt angesteckt. – Diesen daher umdrehen und wieder anstecken. – Leuchtet nun die zuvor genannte Leuchtdiode? – Nein! – Prüfen, ob die Verbindungsleitungen zwischen Steuerplatine und Anzeige- und

Bedienplatine korrekt sind (Verbindungen 21 und 22) bzw. ob die Leuchtdiode richtig gepolt ist.

- Spannung an IC1 zwischen den Pins 5 (GND) und 14 (Ub) messen. Wird hier (bei angestecktem Netzteil und Kippschalter S4 in Richtung Printbuchse) keine oder eine grob abweichende Spannung als 5V gemessen, so liegt ein Bestückungs- und/oder Verdrahtungsfehler vor, welcher unbedingt aufgespürt und beseitigt werden muss.
- **Programmierten** Mikrocontroller PIC16F84 (IC1) sowie IC2 und IC3 (je 74LS138) im **ausgeschalteten** Zustand (Kippschalter S4 in Mittelstellung) einsetzen. **Achtung**: Auf die Polarität achten!
- Sanduhr wieder einschalten (Kippschalter S4 in Richtung Printbuchse). Nun müssen die unteren Leuchtdioden aufleuchten. Ist das nicht der Fall, so beginnen Sie mit der Fehlersuche. Ist der Mikrocontroller (IC1, PIC16F84) auch wirklich programmiert? – Mit einem fabrikfrischen (unprogrammierter) Mikrocontroller kann die Sanduhr nicht funktionieren!
- Den HEX-Codierschalter (S1) in Stellung 0 drehen.
- Starttaste (mittlere Taste, S3) drücken. – Nun müssen die unteren Leuchtdioden dunkel werden, während die oberen Leuchtdioden aufleuchten. Weiters beginnt der Verrieselungsvorgang. D. h. Von den oberen Leuchtdioden wird nach und nach eine dunkel während eine untere nicht leuchtende Leuchtdiode aufleuchtet. Dieser Vorgang dauert 15 Sekunden (wenn sich der HEX-Codierschalter S1 in Position 0 befindet). Bei gestecktem Jumper (JP1) ertönt bei jedem Zustandswechsel der Leuchtdioden (d.h., wenn eine Leuchtdiode dunkel wird, und dafür eine andere aufleuchtet) ein Beepton. Ist dieser Verrieselungsvorgang beendet (es leuchten wieder alle unteren Leuchtdioden, während alle oberen Leuchtdioden dunkel sind) ertönt ein akustisches Signal.
- Erfolgt nach dem Drücken der Starttaste keine Reaktion, so sollten zunächst die Verbindungsleitungen 23 und 25 (zwischen Steuerplatine und Anzeige- und Bedienplatine) überprüft werden.
- Stopptaste (rechte, äußere Taste, S2) drücken. Ist der Verrieselungsvorgang noch nicht beendet, so wird er nun abgebrochen. Nun leuchten wieder nur die unteren Leuchtdioden. War der Verrieselungsvorgang schon beendet (der Lautsprecher erzeugte einen Ton), so wird dieser Ton beendet.
- Erfolgt nach dem Drücken der Stopptaste keine Reaktion, so sollte zunächst die Verbindungsleitung 24 (zwischen Steuerplatine und Anzeige- und Bedienplatine) überprüft werden.
- Den HEX-Codierschalter (S1) in Stellung 1 drehen.
- Starttaste (mittlere Taste, S3) drücken. – Nun sollte der Verrieselungsvorgang 30 Sekunden dauern
- Die letzten beiden Schritte für alle 16 Stellungen des HEX-Codierschalter (S1) über-prüfen. Die Verrieselnde Zeit muss dabei der Tabelle im Abschnitt *5.1. Zeitintervalle ändern* entsprechen.
- Sanduhr ausschalten (Kippschalter S4 in Mittelposition bringen)
- Eine voll geladene 9V-Batterie oder einen voll geladenen 9V-Akku in den Batteriehalter einlegen. **Achtung**: Polarität beachten! (Der Batteriehalter besitzt eine entsprechende Kennzeichnung)
- Den HEX-Codierschalter (S1) in Stellung 0 drehen. – Verrieselungsszeit: 15 Sekunden
- Sanduhr mit den Kippschalter (S4) in Richtung Elko (C1) einschalten – Nun wird die Sanduhr von der Batterie bzw. vom Akku versorgt. Auch hier müssen nun die unteren Leuchtdioden leuchten.

- Starttaste (mittlere Taste, S3) drücken. Der Verrieselungsvorgang beginnt

Schritt 4: Frontplatte vorbereiten

Frontplatte (große Acrylglas-Scheibe) nach Bild 7 und folgender Anleitung vorbereiten:

- Frontplatte auf eine Größe von 175 x 100 mm zuschneiden.
- Bohrungen entsprechend Bild 7 anzeichnen und bohren

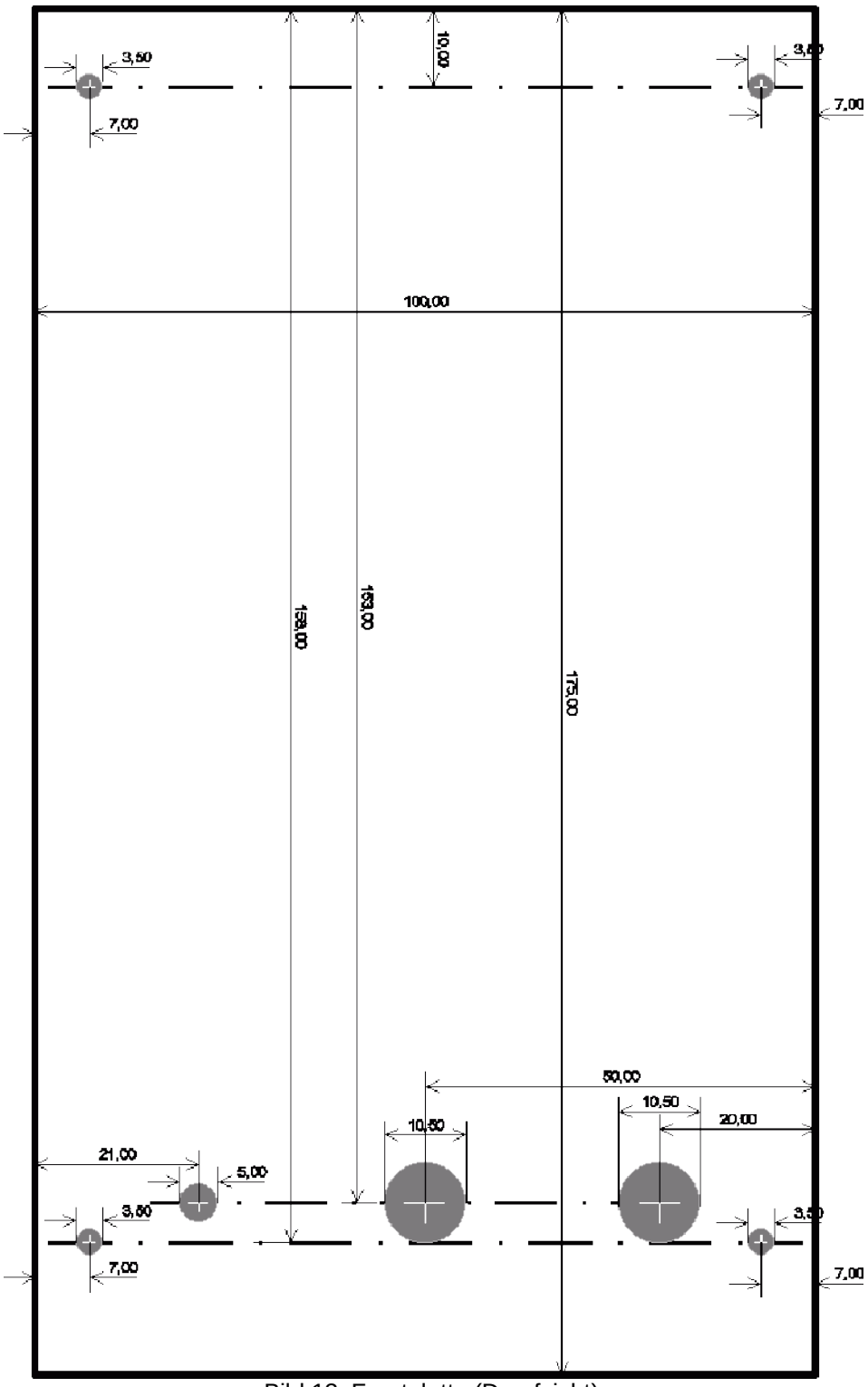


Bild 13: Frontplatte (Draufsicht)

- Die vier 3,5-mm-Bohrungen an den Ecken dienen zur Montage der Anzeige- und Bedienplatine mittels Senkkopfschrauben und muss daher gesenkt werden. (von oben)
- Anhang C auf eine Frontfolie (z.B. die in der Stückliste angegebenen, bei Conrad erhältlich) ausdrucken, zurechtschneiden und so auf die Frontplatte kleben, dass die Bohrungen der Frontfolie mit denen der Frontplatte übereinstimmen.

Schritt 5: Bodenplatte vorbereiten

Bodenplatte (kleine Acrylglas-Scheibe) nach Bild 8 und folgender Anleitung vorbereiten:

- Bodenplatte auf eine Größe von 100 x 100 mm zuschneiden.
- Bohrungen entsprechend Bild 8 anzeichnen und bohren. Alle Bohrungen haben einen Durchmesser von 3,5 mm

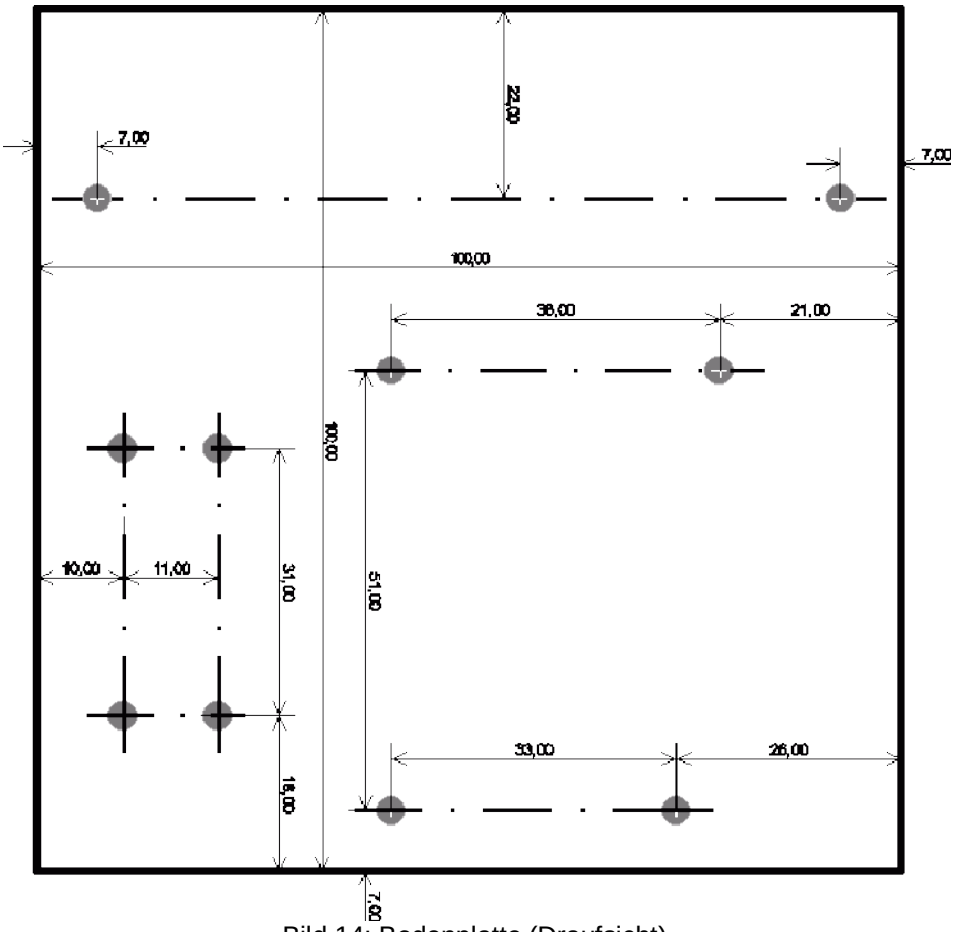


Bild 14: Bodenplatte (Draufsicht)

- Alle Bohrungen von unten senken
- An den Ecken je einen Gerätefuß kleben

Schritt 6: Montagewinkel vorbereiten

2 Montagewinkeln aus je einem 30x12 mm großen etwa 0,5 – 1 mm dicken Aluminium nach Bild 9 herstellen und zu einem Winkel von etwa 70 Grad biegen.

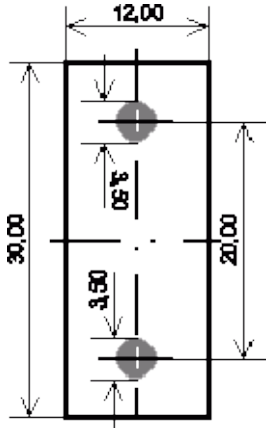


Bild 15: Montagewinkel

Schritt 7: Zusammenbau

Die soweit erfolgreich getestete Sanduhr und die in den Schritten 4 bis 6 vorbereiteten Komponenten nach Bild 10 und der folgender Anleitung zusammenbauen:

- Steuerplatine mit Anzeige- und Bedienplatine mit je 4 Zylinderkopfschrauben M3x16, Distanzrollen 8mm und Mutter M3 verbinden. Im Bild 10 mit A gekennzeichnet.
- Stromversorgungsplatine mit je 4 Senkkopfschrauben M3x16, Distanzrolle 5mm und Mutter M3 auf der Bodenplatte befestigen. Im Bild 10 mit B gekennzeichnet.
- Lautsprecher (LS1) mit zumindest je 2 gegenüberliegenden Senkkopfschrauben M3x20, Distanzrolle 8mm und Mutter M3 auf der Bodenplatte befestigen. Im Bild 10 mit C gekennzeichnet.
- Anzeige- und Bedienplatine (mit der bereits montierten Steuerplatine) auf die Frontplatte mit je 4 Distanzbolzen M3x8, Senkkopfschrauben M3x6 und Mutter M3 montieren. Im Bild 10 mit D gekennzeichnet. Achtung: Bei den beiden unteren muss noch zwischen Bedien- und Anzeigeplatine und Mutter M3 je ein Montagewinkel eingefügt werden.
- Bodenplatte mit je 2 Senkkopfschrauben M3x10 und Mutter M3 an die beiden restlichen Enden der Montagewinkel befestigen. Im Bild 10 mit E gekennzeichnet.
- Steckachse und Kunststoff-Zeigerknopf auf den HEX-Codierschalter drücken. Achtung: Vergewissern Sie sich dass der Zeiger auf die richtige Zeit der Skala zeigt.

Geschafft! Nun Sollte Ihre Sanduhr so wie auf den Bildern 10 und 11 aussehen.

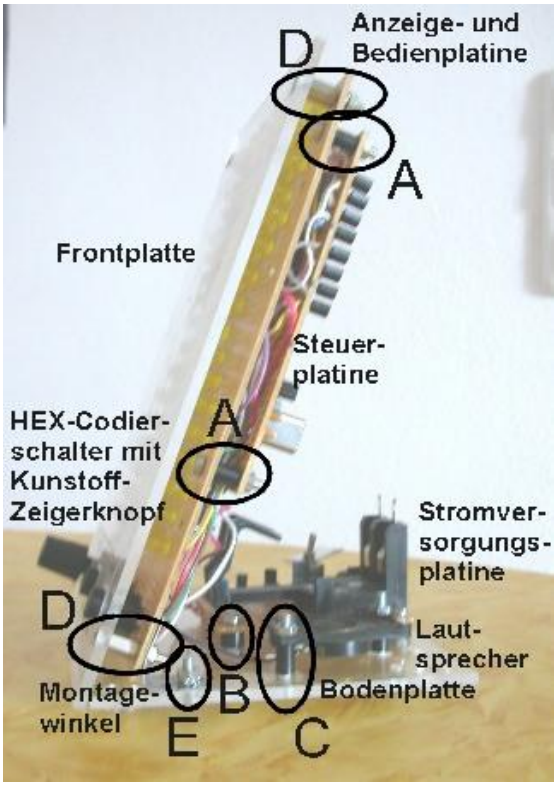


Bild 16: Zusammenbau

Das folgende Bild zeigt die Sanduhr im Betrieb.

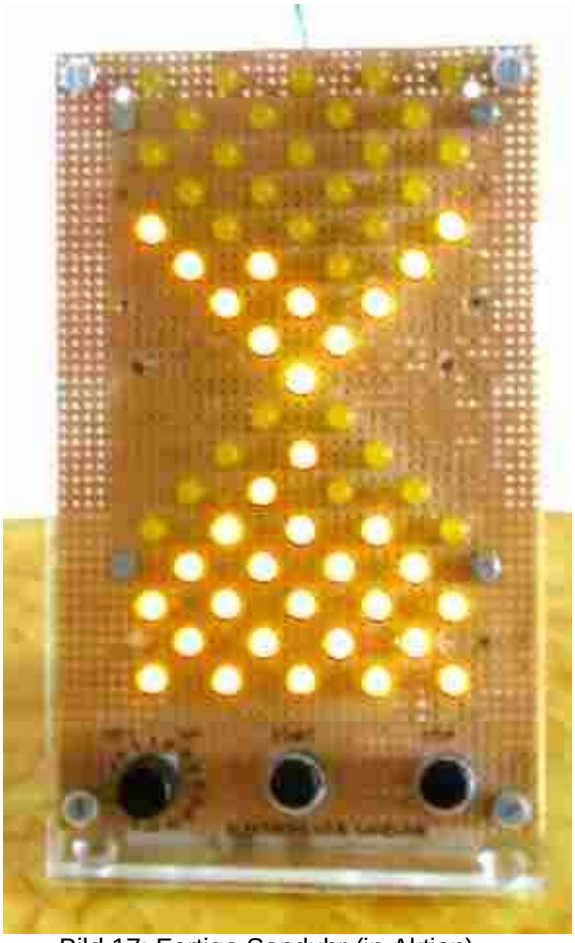
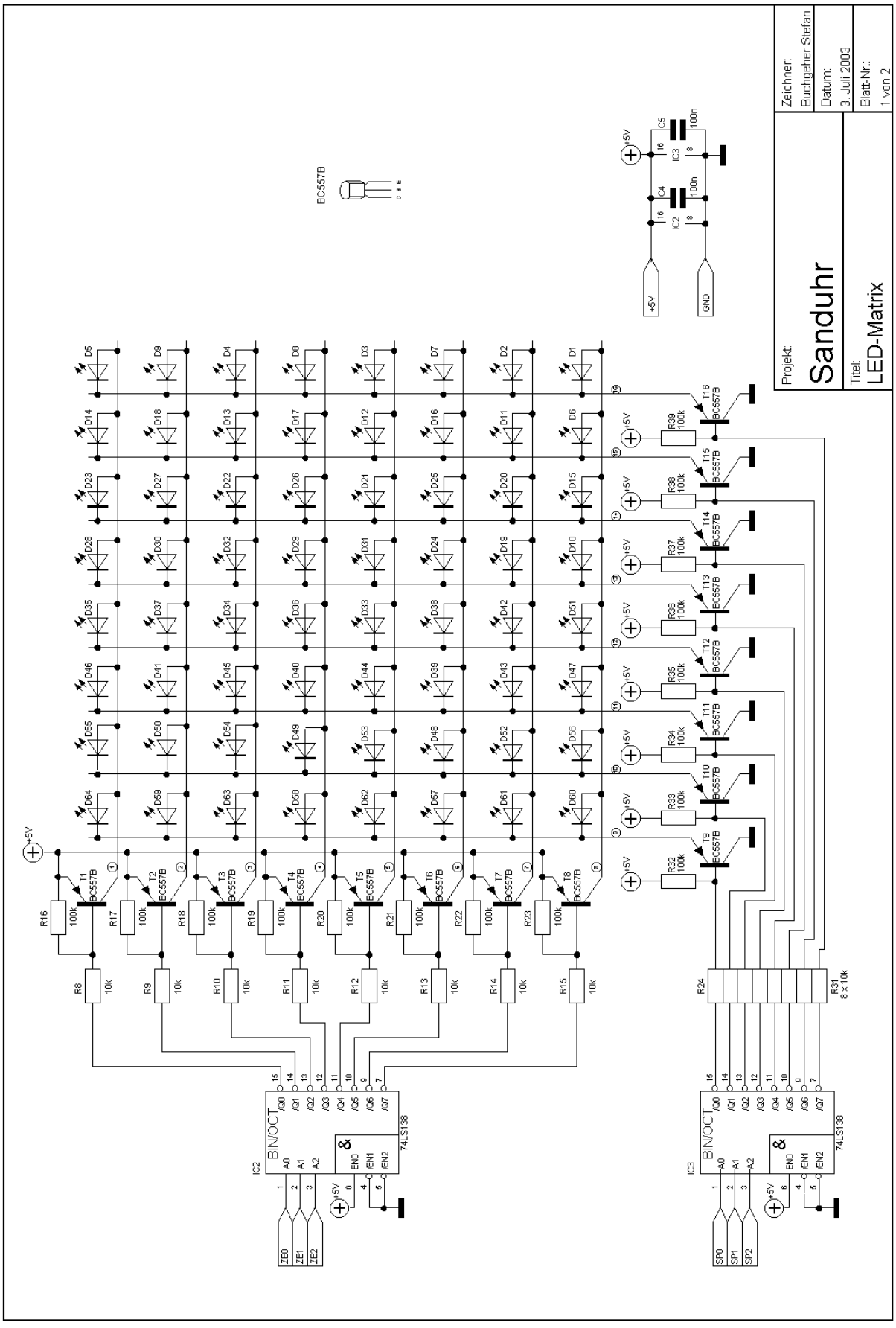
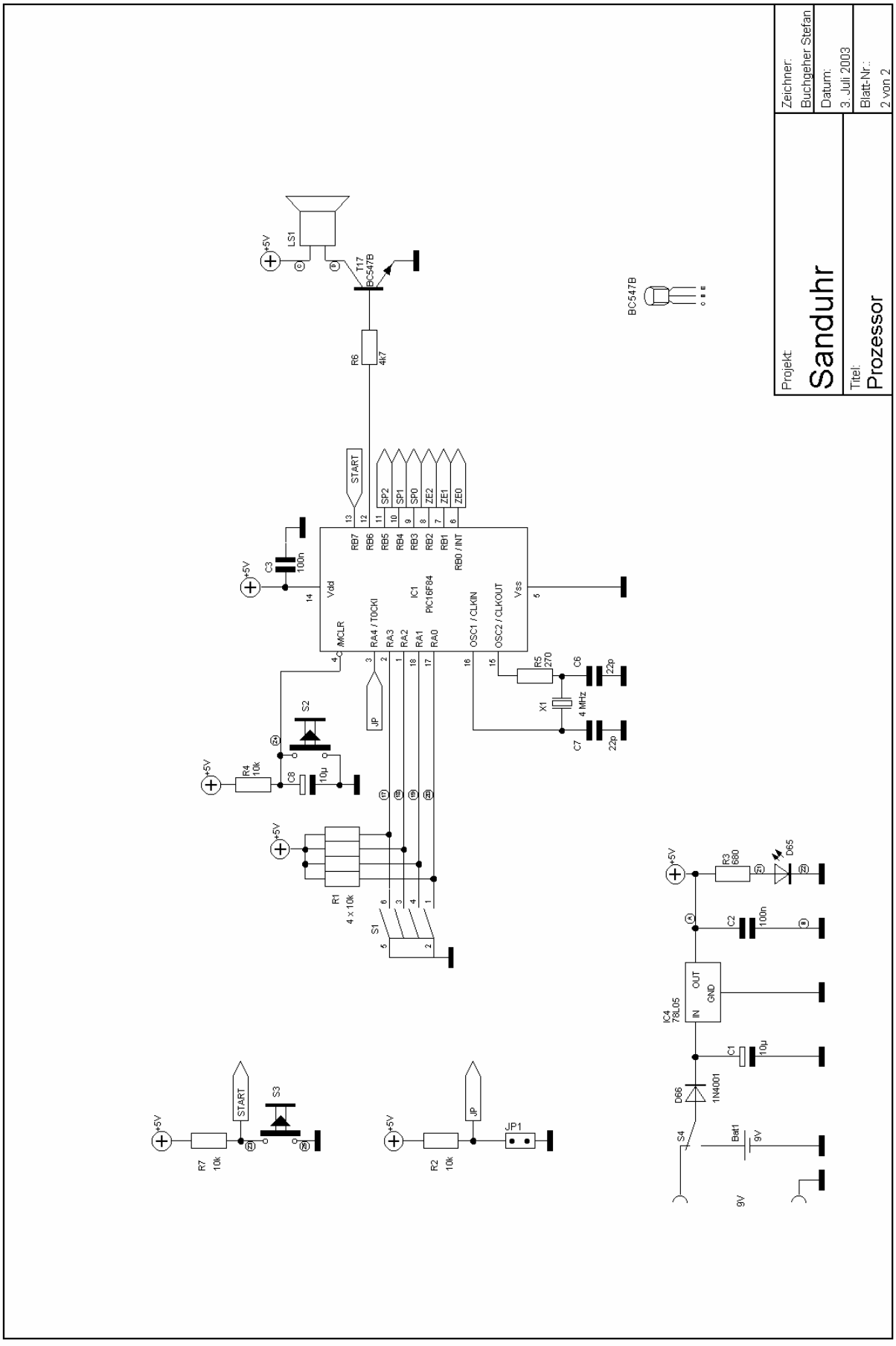


Bild 17: Fertige Sanduhr (in Aktion)

Anhang A: Schaltpläne




Zeichner:
Buchgeher Stefan
Datum:
3. Juli 2003
Blatt-Nr.:
2 von 2
Projekt:
Sanduhr
Titel:
Prozessor
BC547B
LS1
T17
BC547B
R6
4k7
START
SP2
SP1
SP0
ZE2
ZE1
ZE0
C3
100n
IC1
PIC16F84
Vdd
Vss
MCLR
RA4 / T0CKI
OSC1 / CLKIN
OSC2 / CLKOUT
RB7
RB6
RB5
RB4
RB3
RB2
RB1
RB0 / INT
R5
270
C6
22p
X1
4 MHz
C7
22p
JP
S2
R4
10k
C8
10µ
R1
4 x 10k
S1
R3
680
D65
C2
100n
IC4
78L05
IN
GND
OUT
C1
10µ
D66
1N4001
S4
Bat1
9V
9V
R7
10k
START
S3
R2
10k
JP
JP1
+5V

Anhang B: SaUhr1v1.asm

```
;*****************************************************************************************
;** Elektronische Sanduhr V1                                                            **
;**                                                                                     **
;** Sanduhr mit PIC16F84 und 65 LEDs in Form einer 8x8-Matrix und eine in der Mitte der **
;** Sanduhr, welche über einen Vorwiderstand mit Ub (=+5V) verbunden ist.               **
;** Unterschiedliche Zeiteinstellung (von 15 Sekunden bis 60 Minuten), Summer nach Ablauf **
;** der eingestellten Zeit, Beep-Ton bei jedem Zustandswechsel (wenn Jumper gesteckt)   **
;**                                                                                     **
;** Entwickler: Buchgeher Stefan                                                        **
;** Entwicklungsbeginn der Software: 21. Juli 2001                                      **
;** Funktionsfaehig seit: 28. Oktober 2001                                              **
;** Letzte Bearbeitung: 5. Maerz 2004                                                   **
;*****************************************************************************************


               List    p=PIC16F84

;******************************** Register (in Registerseite 0) ******************************
IND            equ     0                    ;Register für indirekte Adressierung
TMR0           equ     1                    ;Timer0-Register
PC             equ     2                    ;Programmzaehler
STAT           equ     3                    ;Statusregister
FSR            equ     4                    ;FSR-Register
PORTA          equ     5                    ;PortA-Register
PORTB          equ     6                    ;PortB-Register
INTCON         equ     0B                   ;Interrupt-Register


;******************************** Register (in Registerseite 1) ******************************
OPTREG         equ     1                    ;OPTION-Register
TRISA          equ     5                    ;Richtungsregister PortA
TRISB          equ     6                    ;Richtungsregister PortB


;******************************** Eigene Register (in Registerseite 0) ***********************
ZAEHLLED       equ     10                   ;Zählvariable (32 bis 0) gibt die Aktuelle
                                            ;  POS-Nr der leuchtende LED an
POS1           equ     11                   ;die folgenden 32 Register beinhalten die
POS2           equ     12                   ;  aktuellen "Adressen" der aktiven
POS3           equ     13                   ;  (leuchtenden) LEDs
POS4           equ     14
POS5           equ     15
POS6           equ     16
POS7           equ     17
POS8           equ     18
POS9           equ     19
POS10          equ     1A
POS11          equ     1B
POS12          equ     1C
POS13          equ     1D
POS14          equ     1E
POS15          equ     1F
POS16          equ     20
POS17          equ     21
POS18          equ     22
POS19          equ     23
POS20          equ     24
POS21          equ     25
POS22          equ     26
POS23          equ     27
POS24          equ     28
POS25          equ     29
POS26          equ     2A
POS27          equ     2B
POS28          equ     2C
POS29          equ     2D
POS30          equ     2E
POS31          equ     2F
POS32          equ     30

SUSTATUS       equ     31                   ;Sanduhr-Statusregister
                                            ;  Bit 0: 1 ... Starttaste wurde gedrueckt,
                                            ;              Körner(LEDs) rieseln nach unten
                                            ;         0 ... Starttaste nicht gedrückt
                                            ;  Bit 1: Hilfsflag: gesetzt: nächster Zustand
                                            ;  Bit 2: gesetzt: Zeit ist abgelaufen
```

```
AKTZEITH        equ     32                      ;aktuell abgel. Zeit zw. 2 Zustandsaenderungen
                                                ;  (High-Byte)
ZEITH           equ     33                      ;eingestellte Zeit zwischen 2 Zuständen
                                                ;  (High-Byte)
AKTZEITL        equ     34                      ;wie vorher, jedoch für das LOW-Byte
ZEITL           equ     35                      ;wie vorher, jedoch für das LOW-Byte

AKTZUSTAND      equ     36                      ;Aktueller Zustand der LEDs (Konstante ZUSTANZ
                                                ;  bis 0)
ZWISCH          equ     37                      ;Zwischenspeicher für auszutauschende LED
DELAY           equ     38                      ;Zählregister für Zeitverzögerungsschleife
w_TEMP          equ     39                      ;Zwischenspeicher für w-Register bei ISR
STAT_TEMP       equ     3A                      ;Zwischenspeicher für STATUS-Register bei ISR
SUCHZUST        equ     3B                      ;Hilfsregister
SUCHZAEHLER     equ     3C                      ;Zählregister von 32 bis 0, verhindert,
                                                ;das ausserhalb von POS bis POS32 gesucht wird


;******************************** Bits in Registern ******************************************
C               equ     0                       ;Carrybit im Statuswort-Register
Z               equ     2                       ;Zerobit im Statuswort-Register
RP0             equ     5                       ;Seitenauswahlbit im Statuswort-Register
T0IF            equ     2                       ;TMR0-Interruptflag im INTCON-Register


;******************************** Ziele der Registeroperationen ******************************
w               equ     0                       ;w ist Zielregister
f               equ     1                       ;f ist Zielregister


;******************************** Eigene Bits in Registern ***********************************
STARTTASTE      equ     7                       ;Port B, Bit 7
SUMMER          equ     6                       ;Port B, Bit 6
SUMMERSELEKT    equ     4                       ;Port A, Bit 4


;******************************** Konstanten *************************************************
LED1            equ     b'10111111'             ;"Adresse" von LED D1
LED2            equ     b'10110111'             ;"Adresse" von LED D2
LED3            equ     b'10100111'             ;"Adresse" von LED D3, usw.
LED4            equ     b'10010111'
LED5            equ     b'10000111'
LED6            equ     b'10111110'
LED7            equ     b'10101111'
LED8            equ     b'10011111'
LED9            equ     b'10001111'
LED10           equ     b'10111100'
LED11           equ     b'10110110'
LED12           equ     b'10100110'
LED13           equ     b'10010110'
LED14           equ     b'10000110'
LED15           equ     b'10111101'
LED16           equ     b'10101110'
LED17           equ     b'10011110'
LED18           equ     b'10001110'
LED19           equ     b'10110100'
LED20           equ     b'10110101'
LED21           equ     b'10100101'
LED22           equ     b'10010101'
LED23           equ     b'10000101'
LED24           equ     b'10101100'
LED25           equ     b'10101101'
LED26           equ     b'10011101'
LED27           equ     b'10001101'
LED28           equ     b'10000100'
LED29           equ     b'10011100'
LED30           equ     b'10001100'
LED31           equ     b'10100100'
LED32           equ     b'10010100'
LED33           equ     b'10100011'
LED34           equ     b'10010011'
LED35           equ     b'10000011'
LED36           equ     b'10011011'
LED37           equ     b'10001011'
LED38           equ     b'10101011'
LED39           equ     b'10101010'
LED40           equ     b'10011010'
LED41           equ     b'10001010'
```

```
LED42           equ     b'10110011'
LED43           equ     b'10110010'
LED44           equ     b'10100010'
LED45           equ     b'10010010'
LED46           equ     b'10000010'
LED47           equ     b'10111010'
LED48           equ     b'10101001'
LED49           equ     b'10011001'
LED50           equ     b'10001001'
LED51           equ     b'10111011'
LED52           equ     b'10110001'
LED53           equ     b'10100001'
LED54           equ     b'10010001'
LED55           equ     b'10000001'
LED56           equ     b'10111001'
LED57           equ     b'10101000'
LED58           equ     b'10011000'
LED59           equ     b'10001000'
LED60           equ     b'10111000'
LED61           equ     b'10110000'
LED62           equ     b'10100000'
LED63           equ     b'10010000'
LED64           equ     b'10000000'
ZUSTANZ         equ     .112                    ;Anzahl der Zustände
LEDANZAKTIV     equ     .32                     ;Anzahl der leuchtenden LEDs

ZEIT0H          equ     .2                      ;High- und Low-Byte
ZEIT0L          equ     .5                      ;  für 15 Sekunden
ZEIT1H          equ     .3                      ;High- und Low-Byte
ZEIT1L          equ     .11                     ;  für 30 Sekunden
ZEIT2H          equ     .5                      ;High- und Low-Byte
ZEIT2L          equ     .22                     ;  für 60 Sekunden
ZEIT3H          equ     .7                      ;High- und Low-Byte
ZEIT3L          equ     .33                     ;  für 90 Sekunden
ZEIT4H          equ     .9                      ;High- und Low-Byte
ZEIT4L          equ     .44                     ;  für 2 Minuten
ZEIT5H          equ     .13                     ;High- und Low-Byte
ZEIT5L          equ     .67                     ;  für 3 Minuten
ZEIT6H          equ     .17                     ;High- und Low-Byte
ZEIT6L          equ     .89                     ;  für 4 Minuten
ZEIT7H          equ     .21                     ;High- und Low-Byte
ZEIT7L          equ     .111                    ;  für 5 Minuten
ZEIT8H          equ     .33                     ;High- und Low-Byte
ZEIT8L          equ     .178                    ;  für 8 Minuten
ZEIT9H          equ     .41                     ;High- und Low-Byte
ZEIT9L          equ     .223                    ;  für 10 Minuten
ZEIT10H         equ     .50                     ;High- und Low-Byte
ZEIT10L         equ     .11                     ;  für 12 Minuten
ZEIT11H         equ     .62                     ;High- und Low-Byte
ZEIT11L         equ     .79                     ;  für 15 Minuten
ZEIT12H         equ     .82                     ;High- und Low-Byte
ZEIT12L         equ     .190                    ;  für 20 Minuten
ZEIT13H         equ     .123                    ;High- und Low-Byte
ZEIT13L         equ     .157                    ;  für 30 Minuten
ZEIT14H         equ     .184                    ;High- und Low-Byte
ZEIT14L         equ     .236                    ;  für 45 Minuten
ZEIT15H         equ     .246                    ;High- und Low-Byte
ZEIT15L         equ     .59                     ;  für 60 Minuten


;******************************** Konfigurations-Bits ****************************************
_lp_osc         equ     h'3FFC'
_xt_osc         equ     h'3FFD'
_hs_osc         equ     h'3FFE'
_rc_osc         equ     h'3FFF'

_wdt_off        equ     h'3FFB'
_wdt_on         equ     h'3FFF'

_pwrt_off       equ     h'3FFF'
_pwrt_on        equ     h'3FF7'

_cp_off         equ     h'3FFF'
_cp_on          equ     h'000F'

        __config        _hs_osc & _wdt_off & _pwrt_off & _cp_off

                ORG     0x000
                goto    Beginn
```

```
                ORG     0x004
                goto    ISR

;******************************** Tabellen ***************************************************
TABWIRDERSETZT addwf   PC,f
                nop
                retlw   LED32
                retlw   LED31
                retlw   LED33
                retlw   LED30
                retlw   LED34
                retlw   LED28
                retlw   LED37
                retlw   LED34
                retlw   LED24
                retlw   LED35
                retlw   LED33
                retlw   LED27
                retlw   LED41
                retlw   LED37
                retlw   LED34
                retlw   LED23
                retlw   LED38
                retlw   LED35
                retlw   LED33
                retlw   LED19
                retlw   LED29
                retlw   LED36
                retlw   LED25
                retlw   LED36
                retlw   LED26
                retlw   LED40
                retlw   LED36
                retlw   LED22
                retlw   LED39
                retlw   LED36
                retlw   LED20
                retlw   LED45
                retlw   LED40
                retlw   LED36
                retlw   LED18
                retlw   LED43
                retlw   LED39
                retlw   LED36
                retlw   LED15
                retlw   LED47
                retlw   LED43
                retlw   LED39
                retlw   LED36
                retlw   LED14
                retlw   LED50
                retlw   LED45
                retlw   LED40
                retlw   LED36
                retlw   LED10
                retlw   LED36
                retlw   LED21
                retlw   LED44
                retlw   LED36
                retlw   LED16
                retlw   LED44
                retlw   LED36
                retlw   LED17
                retlw   LED49
                retlw   LED44
                retlw   LED36
                retlw   LED11
                retlw   LED48
                retlw   LED44
                retlw   LED36
                retlw   LED13
                retlw   LED54
                retlw   LED49
                retlw   LED44
                retlw   LED36
                retlw   LED6
                retlw   LED52
                retlw   LED48
```

```
                retlw   LED44
                retlw   LED36
                retlw   LED9
                retlw   LED59
                retlw   LED54
                retlw   LED49
                retlw   LED44
                retlw   LED36
                retlw   LED1
                retlw   LED56
                retlw   LED52
                retlw   LED48
                retlw   LED44
                retlw   LED36
                retlw   LED5
                retlw   LED44
                retlw   LED36
                retlw   LED4
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED2
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED7
                retlw   LED58
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED8
                retlw   LED57
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED12
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED3

TABNEUELED      addwf   PC,f
                nop
                retlw   LED34
                retlw   LED33
                retlw   LED35
                retlw   LED33
                retlw   LED37
                retlw   LED34
                retlw   LED41
                retlw   LED37
                retlw   LED34
                retlw   LED38
                retlw   LED35
                retlw   LED33
                retlw   LED46
                retlw   LED41
                retlw   LED37
                retlw   LED34
                retlw   LED42
                retlw   LED38
                retlw   LED35
                retlw   LED33
                retlw   LED36
                retlw   LED40
                retlw   LED36
                retlw   LED39
                retlw   LED36
                retlw   LED45
                retlw   LED40
                retlw   LED36
                retlw   LED43
                retlw   LED39
                retlw   LED36
                retlw   LED50
                retlw   LED45
                retlw   LED40
                retlw   LED36
                retlw   LED47
```

```
                retlw  LED43
                retlw  LED39
                retlw  LED36
                retlw  LED51
                retlw  LED47
                retlw  LED43
                retlw  LED39
                retlw  LED36
                retlw  LED55
                retlw  LED50
                retlw  LED45
                retlw  LED40
                retlw  LED36
                retlw  LED44
                retlw  LED36
                retlw  LED49
                retlw  LED44
                retlw  LED36
                retlw  LED48
                retlw  LED44
                retlw  LED36
                retlw  LED54
                retlw  LED49
                retlw  LED44
                retlw  LED36
                retlw  LED52
                retlw  LED48
                retlw  LED44
                retlw  LED36
                retlw  LED59
                retlw  LED54
                retlw  LED49
                retlw  LED44
                retlw  LED36
                retlw  LED56
                retlw  LED52
                retlw  LED48
                retlw  LED44
                retlw  LED36
                retlw  LED64
                retlw  LED59
                retlw  LED54
                retlw  LED49
                retlw  LED44
                retlw  LED36
                retlw  LED60
                retlw  LED56
                retlw  LED52
                retlw  LED48
                retlw  LED44
                retlw  LED36
                retlw  LED53
                retlw  LED44
                retlw  LED36
                retlw  LED58
                retlw  LED53
                retlw  LED44
                retlw  LED36
                retlw  LED57
                retlw  LED53
                retlw  LED44
                retlw  LED36
                retlw  LED63
                retlw  LED58
                retlw  LED53
                retlw  LED44
                retlw  LED36
                retlw  LED61
                retlw  LED57
                retlw  LED53
                retlw  LED44
                retlw  LED36
                retlw  LED62
                retlw  LED53
                retlw  LED44
                retlw  LED36

;******************************** ISR - Timer0 ************************************************
```

```
;*****************************************************************************************
;** Interrupt Service Routine:                                                           **
;**                                                                                      **
;** w-Register (=Arbeitsregister) und Status-Register zwischenspeichern. In ZAEHLLED steht  **
;** die Nummer der auszugebende LED. Die Adresse dieser LED aus dem RAM laden (POS1 bis     **
;** POS32) und am Port B ausgeben. Da das FSR-Register auch im Hauptprogramm verwendet wird, **
;** den Inhalt vor Aufruf der ISR wiederherstellen (SUCHZUST). Wenn Bit SUSTATUS.0 gesetzt  **
;** (d.h. die LEDs rieseln nach unten) die Zählregister AKTZEITL bzw. AKTZEITH bei jedem ISR **
;** Aufruf um eins vermindern. Bei AKTZEITH=0 und AKTZEITL=0 ein Zustandswechsel der LEDs   **
;** erfolgen, daher das Bit SUSTATUS.1 setzen, und die Register AKTZEITL und AKTZEITH neu   **
;** laden.                                                                               **
;** Das Timer-Interrupt-Flag T0IF wider löschen, und das w- bzw. Status-Register wiederher- **
;** stellen.                                                                             **
;*****************************************************************************************
ISR
PUSH            movwf   w_TEMP                  ;w-Register
                swapf   STAT,w                  ;  und auch das
                movwf   STAT_TEMP               ;  Status-Register retten

                decfsz  ZAEHLLED,f              ;Zaehlvar. um 1 erniedrigen
                goto    weiter
                movlw   LEDANZAKTIV             ;wenn ZAEHLLED=0, ZAEHLLED wieder mit der
                movwf   ZAEHLLED                ;Konstante LEDANZAKTIV (=32) laden
weiter          movlw   10
                addwf   ZAEHLLED,w
                movwf   FSR
                movf    IND,w                   ;akt. LED-Adresse aus RAM laden
                movwf   PORTB                   ;und am Port ausgeben

                movf    SUCHZUST,w
                movwf   FSR

                btfss   SUSTATUS,0              ;Wenn SUSTATUS.0 gesetzt
                goto    fertig

                decfsz  AKTZEITL,f              ;Akt. Zeit Low von
                goto    fertig                  ;LED-Zustandsänderungen erniedrigen
                decfsz  AKTZEITH,f              ;  High-Byte erniedrigen
                goto    fertig
                bsf     SUSTATUS,1              ;wenn 0: NaechstZust-Flag setzen
                movf    ZEITL,w                 ;und Register neu laden
                movwf   AKTZEITL                ;  ZEITL -> AKTZEITL
                movf    ZEITH,w                 ;  bzw.
                movwf   AKTZEITH                ;  ZEITH -> AKTZEITH

fertig          bcf     INTCON,T0IF             ;T0-Interrupftflag loeschen

POP             swapf   STAT_TEMP,w             ;Status-Register
                movwf   STAT                    ;  und
                swapf   w_TEMP,f                ;  w-Register
                swapf   w_TEMP,w                ;  wieder herstellen

                retfie


;****************************** Unterprogramme ***************************************

;*****************************************************************************************
;** Initialisierung des Prozessor:                                                       **
;**     + TMR0-Vorteiler mit 2                                                           **
;**     + Ports: Port A: Eingang                                                         **
;**              Port B: Bit 0-6: Ausgang, Bit 7: Eingang                                **
;**     + Diverse Register laden                                                         **
;*****************************************************************************************
INIT            clrf    TMR0                    ;Timer0 auf 0 voreingestellt
                bsf     STAT,RP0                ;Registerseite 1
                movlw   b'00000000'             ;interner Takt, Vorteiler = 2 an TMR0
                movwf   OPTREG
                movlw   b'10000000'
                movwf   TRISB                   ;Port B : Bit 0-5 : Ausgang (LED-Matrix)
                                                ;         Bit 6   : Ausgang (Summer)
                                                ;         Bit 7   : Eingang (Starttaste)
                movlw   b'11111'
                movwf   TRISA                   ;Port A : Eingang
                bcf     STAT,RP0                ;Registerseite 0
                movlw   b'00000000'
                movwf   SUSTATUS                ;Sanduhr-Statusregister = 0
                movlw   LEDANZAKTIV             ;ZAEHLLED mit der Konstante LEDANZAKTIV
```

```
                movwf   ZAEHLLED                ;laden (=32)
                return


;*****************************************************************************************
;** Ausgangszustand der LEDs (die unteren 32 LEDs leuchten, diese daher in die Register POS1 **
;** bis POS32 laden)                                                                      **
;*****************************************************************************************
ZUSTAND0        movlw   LED33
                movwf   POS1
                movlw   LED34
                movwf   POS2
                movlw   LED35
                movwf   POS3
                movlw   LED36
                movwf   POS4
                movlw   LED37
                movwf   POS5
                movlw   LED38
                movwf   POS6
                movlw   LED39
                movwf   POS7
                movlw   LED40
                movwf   POS8
                movlw   LED41
                movwf   POS9
                movlw   LED42
                movwf   POS10
                movlw   LED43
                movwf   POS11
                movlw   LED44
                movwf   POS12
                movlw   LED45
                movwf   POS13
                movlw   LED46
                movwf   POS14
                movlw   LED47
                movwf   POS15
                movlw   LED48
                movwf   POS16
                movlw   LED49
                movwf   POS17
                movlw   LED50
                movwf   POS18
                movlw   LED51
                movwf   POS19
                movlw   LED52
                movwf   POS20
                movlw   LED53
                movwf   POS21
                movlw   LED54
                movwf   POS22
                movlw   LED55
                movwf   POS23
                movlw   LED56
                movwf   POS24
                movlw   LED57
                movwf   POS25
                movlw   LED58
                movwf   POS26
                movlw   LED59
                movwf   POS27
                movlw   LED60
                movwf   POS28
                movlw   LED61
                movwf   POS29
                movlw   LED62
                movwf   POS30
                movlw   LED63
                movwf   POS31
                movlw   LED64
                movwf   POS32
                return


;*****************************************************************************************
;** Zustand nach drücken der Starttaste. Die oberen 32 LEDs leuchten, diese daher in die     **
;** Register POS1 bis POS32 laden                                                          **
;*****************************************************************************************
```

```
ZUSTAND1        movlw   LED1
                movwf   POS1
                movlw   LED2
                movwf   POS2
                movlw   LED3
                movwf   POS3
                movlw   LED4
                movwf   POS4
                movlw   LED5
                movwf   POS5
                movlw   LED6
                movwf   POS6
                movlw   LED7
                movwf   POS7
                movlw   LED8
                movwf   POS8
                movlw   LED9
                movwf   POS9
                movlw   LED10
                movwf   POS10
                movlw   LED11
                movwf   POS11
                movlw   LED12
                movwf   POS12
                movlw   LED13
                movwf   POS13
                movlw   LED14
                movwf   POS14
                movlw   LED15
                movwf   POS15
                movlw   LED16
                movwf   POS16
                movlw   LED17
                movwf   POS17
                movlw   LED18
                movwf   POS18
                movlw   LED19
                movwf   POS19
                movlw   LED20
                movwf   POS20
                movlw   LED21
                movwf   POS21
                movlw   LED22
                movwf   POS22
                movlw   LED23
                movwf   POS23
                movlw   LED24
                movwf   POS24
                movlw   LED25
                movwf   POS25
                movlw   LED26
                movwf   POS26
                movlw   LED27
                movwf   POS27
                movlw   LED28
                movwf   POS28
                movlw   LED29
                movwf   POS29
                movlw   LED30
                movwf   POS30
                movlw   LED31
                movwf   POS31
                movlw   LED32
                movwf   POS32
                return


;*****************************************************************************************
;** Eingestellte Zeit (HEX-Schalter) abfragen und die Register ZEIT bzw. AKTZEIT sowohl für  **
;** das High als auch für das Low-Byte laden. Einstellbare Zeiten sind:                      **
;**     15, 30 sec 1,1 1/2,2,3,4,5,8,10,12,15,20,30,45,60 min                               **
;*****************************************************************************************
;
ZEITAUSWAHL
                btfss   PORTA,3             ;Wenn RA3 gesetzt:
                goto    Z8bis15             ;  Schalter zw. 8 und 15 ist eingestellt
                btfss   PORTA,2             ;  sonst, wenn RA2 gesetzt
                goto    Z4bis7              ;  Schalter zw. 4 und 7 ist eingestellt
                btfss   PORTA,1             ;  sonst, wenn RA1 gesetzt
                goto    Z2u3                ;  Schalter entweder Position 2 oder 3
```

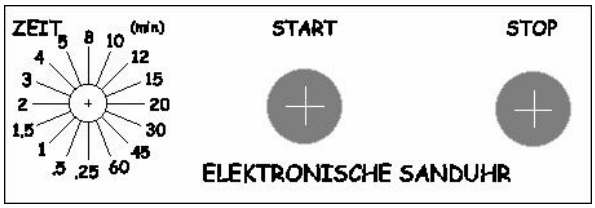
```
                btfss   PORTA,0             ;  sonst, wenn RA0 gesetzt
                goto    ZEIT1               ;  Schalter ist in Position 1
                goto    ZEIT0               ;  sonst Schalter ist in Position 0

Z2u3            btfss   PORTA,0             ;Wenn RA0 gesetzt
                goto    ZEIT3               ;  Schalter ist in Position 3
                goto    ZEIT2               ;  sonst ist Schalter in Position 4

Z4bis7          btfss   PORTA,1             ;  analog zu vorher
                goto    Z6u7
                btfss   PORTA,0
                goto    ZEIT5
                goto    ZEIT4
Z6u7            btfss   PORTA,0
                goto    ZEIT7
                goto    ZEIT6

Z8bis15         btfss   PORTA,2
                goto    Z12bis15
                btfss   PORTA,1
                goto    Z10u11
                btfss   PORTA,0
                goto    ZEIT9
                goto    ZEIT8

Z10u11          btfss   PORTA,0
                goto    ZEIT11
                goto    ZEIT10

Z12bis15        btfss   PORTA,1
                goto    Z14u15
                btfss   PORTA,0
                goto    ZEIT13
                goto    ZEIT12
Z14u15          btfss   PORTA,0
                goto    ZEIT15
                goto    ZEIT14

ZEIT0           movlw   ZEIT0L              ;Konstante ZEIT0L (Low-Byte für 15 sec)
                movwf   ZEITL               ;  laden und die Register ZEITL
                movwf   AKTZEITL            ;  und AKTZEITL damit laden
                movlw   ZEIT0H              ;Konstante ZEIT0H (High-Byte für 15 sec)
                movwf   ZEITH               ;  laden und die Register ZEITH
                movwf   AKTZEITH            ;  und AKTZEITH damit laden
                return
ZEIT1           movlw   ZEIT1L              ;Konstante ZEIT1L (Low-Byte für 30 sec)
                movwf   ZEITL               ;  laden und die Register ZEITL
                movwf   AKTZEITL            ;  und AKTZEITL damit laden
                movlw   ZEIT1H              ;Konstante ZEIT1H (High-Byte für 30 Sec)
                movwf   ZEITH               ;  laden und die Register ZEITH
                movwf   AKTZEITH            ;  und AKTZEITH damit laden
                return
ZEIT2           movlw   ZEIT2L              ;wie vorher
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT2H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT3           movlw   ZEIT3L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT3H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT4           movlw   ZEIT4L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT4H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT5           movlw   ZEIT5L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT5H
                movwf   ZEITH
                movwf   AKTZEITH
```

```
                return
ZEIT6           movlw   ZEIT6L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT6H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT7           movlw   ZEIT7L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT7H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT8           movlw   ZEIT8L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT8H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT9           movlw   ZEIT9L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT9H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT10          movlw   ZEIT10L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT10H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT11          movlw   ZEIT11L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT11H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT12          movlw   ZEIT12L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT12H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT13          movlw   ZEIT13L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT13H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT14          movlw   ZEIT14L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT14H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT15          movlw   ZEIT15L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT15H
                movwf   ZEITH
                movwf   AKTZEITH
                return

;*****************************************************************************************
;** LED-Zustandswechsel:                                                                **
;** Die zu ersetzende LED (LED-Adresse) aus Tabelle TABWIRDERSETZT holen, diese Adresse im   **
;** RAM (POS1 bis POS32) suchen und durch die neue LED aus Tabelle TABNEUELED ersetzen.      **
;** Beep-Ton, wenn Eingang (RA4) low (Jumper gesteckt) Flag wieder löschen                  **
;*****************************************************************************************
```

```
NAECHSTZUST movlw     11                    ;Adresse von POS1 in
            movwf   SUCHZUST                ;Zaehlregister SUCHZUST laden

            movlw   LEDANZAKTIV             ;Anzahl der leuchtenden (Aktiven) LEDs in den
            movwf   SUCHZAEHLER             ;SUCHZAEHLER laden

            movf    AKTZUSTAND,w            ;die zu ersetzende LED (LEDADRESSE)
            call    TABWIRDERSETZT          ;  aus Tabelle holen
            movwf   ZWISCH                  ;  und in ZWISCH speichern

SUCHEN      movf    SUCHZUST,w
            movwf   FSR
            movf    IND,w                   ;Tabellenwert (=ZWISCH) von IND subtrahieren
            subwf   ZWISCH,w
            btfsc   STAT,Z                  ;Z(ero)-Flag prüfen
            goto    NEUELED                 ;Wenn Z(ero)-Flag gesetzt:ZWISCH=IND -> LED,
                                            ;  auf Welche das FSR zeigt austauchen
            incf    SUCHZUST,f              ;SUCHZUST erhöhen und
            decfsz  SUCHZAEHLER,f           ;Such-Zaehler -1
            goto    SUCHEN
            goto    NAECHSTZUST             ;wenn 0 -> zu ersetzende LED konnte nicht ge-
                                            ;  funden werden, -> FEHLER! -> Gesamten
                                            ;  Suchvorgang wiederholen

NEUELED     movf    AKTZUSTAND,w
            call    TABNEUELED
            movwf   IND

            decfsz  AKTZUSTAND,f            ;Zustandszähler um 1 erniedrigen
            goto    FLAGLOESCHEN
            bsf     SUSTATUS,2              ;FertigFlag setzen und
            bcf     SUSTATUS,0              ;LED-rieseln-Status-Bit löschen
FLAGLOESCHEN bcf    SUSTATUS,1              ;NaechstZust-Flag wieder löschen

BEEPTON     btfsc   PORTA,SUMMERSELEKT      ;wenn Summerselekt low (Jumper gesteckt)
            goto    ZURUECK
            bsf     PORTB,SUMMER            ;Beep-Ton
            call    WARTESCHLEIFE           ;in Form eines Impulses
            bcf     PORTB,SUMMER            ;ausgeben

ZURUECK     return


;*****************************************************************************************
;** Summer summt                                                                        **
;** Ausgabe einer (Rechteck-)Frequenz von ca. 1 kHz                                     **
;*****************************************************************************************
SUMMERAKTIV bsf     PORTB,SUMMER            ;Summer einschalten
            call    WARTESCHLEIFE
            bcf     PORTB,SUMMER            ;Summer ausschalten
            call    WARTESCHLEIFE
            return


;*****************************************************************************************
;** Warteschleife                                                                       **
;** Verzoegerungszeit: 0.5 ms bei 4 MHz                                                 **
;*****************************************************************************************
WARTESCHLEIFE movlw   .164
            movwf   DELAY
VERZOEGERUNG decfsz DELAY,f
            goto    VERZOEGERUNG
            nop
            return


;******************************** Hauptprogramm ***************************************

;*****************************************************************************************
;** Aufgaben des Hauptprogramms                                                         **
;** + Aufrufen von INIT (Initialisierungen                                              **
;** + Aufrufen von ZUSTAND0 (Ausgangszustand                                            **
;** + Interrupts freigeben                                                              **
;** + warten bis Starttaste gedrückt                                                    **
;** + Aufrufen von ZUSTAND1                                                             **
;** + AKTZUSTAND=ZUSTANZ                                                                **
;** + Aufrufen von ZEITAUSWAHL                                                          **
;** + Wenn Zeit ist abgelaufen Summer aktivieren und auf erneute Betaetigung der Starttaste   **
;**   warten                                                                            **
```

```
;** + Wenn Zeit ist nicht abgelaufen, warten bis das Zustandsänderungsflag gesetzt ist. Dann  **
;**   NAECHSTZUST aufrufen                                                                    **
;*******************************************************************************************
Beginn          call    INIT                    ;Initialisierungen
                call    ZUSTAND0                ;Unteren 32 LEDs leuchten (Ausgangszustand)
                movlw   b'10100000'             ;TMR0 freigeben durch
                movwf   INTCON                  ;Setzen von GIE und T0IE im INTCON Register

START           btfsc   PORTB,STARTTASTE        ;Warten, bis Starttaste gedrueckt
                goto    START

HAUPTSCHLEIFE   bsf     SUSTATUS,0              ;Wenn Starttaste gerückt, Statusbit setzen
                bcf     SUSTATUS,2              ;  und Fertig-Flag löschen
                call    ZUSTAND1                ;  und die LEDs "springen" nach oben

                movlw   ZUSTANZ                 ;die Konstante ZUSTANZ (Anzahl der möglichen
                movwf   AKTZUSTAND              ;  LED-Zustände) ins Register AKTZUSTAND kopieren

                call    ZEITAUSWAHL

WDH2            btfss   SUSTATUS,2              ;wenn Fertig-Flag gesetzt
                goto    WEITER

SCHLEIFE        call    SUMMERAKTIV             ;Summer aktiv (Signalisierung der
                                                ;  abgelaufenen Zeit)
                btfsc   PORTB,STARTTASTE        ;warten, bis Starttaste gedrueckt
                goto    SCHLEIFE                ;Starttaste nicht gedrückt
                goto    HAUPTSCHLEIFE           ;Starttaste gedrückt

WEITER          btfss   SUSTATUS,1              ;Wenn Zustandsänderungs-Flag gesetzt
                goto    WDH2

                call    NAECHSTZUST             ;erfolgt ein Zustandswechsel der LEDs
                goto    WDH2

                end
```

Anhang C: Frontfolie



Maßstab: 1 : 1 (Originalgröße)

Anhang D: Stücklisten

Anzeigeplatine

Nr.	Bezeichnung	St	Liefe-rant	Best. Nr.	E-Preis	Bemerkungen
D1-D65	LED 5mm diffus, gelb	65	Conrad	184900	0,07	
S1	HEX-Codierschalter	1	Conrad	705497	4,29	
S2,S3	Impulsschalter, schwarz rund	2	Conrad	708461	0,79	
	Lochraster-Platine (160 x 100 mm)	1	Conrad	527769	3,25	

Steuerplatine

Nr.	Bezeichnung	St	Liefe-rant	Best. Nr.	E-Preis	Bemerkungen
R5	Widerstand 270 Ohm	1	Conrad	403108	0,11	
R3	Widerstand 680 Ohm	1	Conrad	403237	0,11	
R6	Widerstand 4k7	1	Conrad	403334	0,11	
R2,R4,R7-R15, R24-R31	Widerstand 10k	23	Conrad	403377	0,09	
R1	Widerstandsarray (4 x 10k) oder 4 Einzelwiderstände	1	Conrad	416029	0,15	
R16-R23, R32-R39	Widerstand 100k	16	Conrad	403490	0,09	
C6, C7	Keramikkondensator 22pF	2	Conrad	457167	0,14	
C3-C5	Keramikkondensator 100nF	3	Conrad	453358	0,25	
C8	Elko 10µF/25V	1	Conrad	460524	0,14	
T17	Transistor BC547B	1	Conrad	155012	0,21	
T1-T16	Transistor BC557B	16	Conrad	155098	0,14	
IC1	PIC16F84	1	Conrad	146773	10,83	Programmiert mit SaUhr1v1.hex
IC2, IC3	74LS138	2	Conrad	169463	0,85	
X1	Quarz 4 MHz	1	Conrad	182087	1,15	
JP1	Stiftleiste 2polig	1				
	IC-Präzisions-Sockel 16polig	2	Conrad	189626	0,57	Für IC2 und IC3
	IC-Präzisions-Sockel 18polig	1	Conrad	189634	0,57	Für IC1
	Lochraster-Platine (100 x 81 mm)	1	Conrad	527769	3,25	Teil einer Euro-Platine

Stromversorgungsplatine

Nr.	Bezeichnung	St	Liefe-rant	Best. Nr.	E-Preis	Bemerkungen
C2	Keramikkondensator 100nF	1	Conrad	453358	0,25	
C1	Elko 10µF/25 V	1	Conrad	460524	0,14	
D66	1N4001	1	Conrad	162213	0,09	
IC4	78L05	1	Conrad	183024	0,5	
S4	Miniatur Kippschalter (1pol, Ein/Aus/Ein)	1	Conrad	701513	2,32	
	Printbuchse	1	Conrad	733989	0,57	Passt zum Steckernetzgerät
BAT1	Einbau-Batteriehalter (9V-Block)	1	Conrad	522546	1,95	
	Senkkopfschraube M3x10	2				Für Batteriehalter
	Mutter M3	2				Für Batteriehalter
	Platine (Lochraster 59x46mm)	1	Conrad	527769	3,26	Teil einer Euro-Platine

Sonstiges

Nr.	Bezeichnung	St	Liefe-rant	Best. Nr.	E-Preis	Bemerkungen
LS1	Miniatur-Lautsprecher	1	Conrad	335400	2,90	
	Kunststoff-Zeigerknopf (schwarz)	1	Conrad	717495	0,72	Für HEX-Codierschalter (S1)
	Steckachse (schwarz)	1	Conrad	425419	0,14	Für HEX-Codierschalter (S1)
	Steckernetzgerät PA300, unstabilisiert	1	Conrad	518305	6,47	
	Acrylglas-Scheibe klar (100 x 100 mm)	1	Conrad	530778	1,23	Bodenplatte
	Acrylglas-Scheibe klar (200 x 100 mm)	1	Conrad	530816	2,03	Frontplatte
	Gehäusefüße	4	Conrad	540161	0,21	Auf Bodenplatte kleben
	Frontfolie	1	Conrad	529672	5,74	Bedruckt mit Sanduhr.fpl oder Frontplatte.doc (lt. Anhang C)
	Alustück 30x12x1 mm	2				Montagewinkel
	Zylinderkopfschraube M3x16	4				
	Senkkopfschraube M3x6	4				
	Senkkopfschraube M3x10	2				
	Senkkopfschraube M3x16	4				

	Senkkopfschraube M3x20	4				
	Distanzrolle Kunststoff 5mm	4				
	Distanzrolle Kunststoff 8mm	8				
	Distanz M3x8 mm	4				
	Mutter M3	18				

Anhang E: SaUhr1v2.asm

```
;*************************************************************************************************
;** Elektronische Sanduhr V1 mit Schaltausgang anstelle Jumper                                  **
;**                                                                                             **
;** Sanduhr mit PIC16F84 und 65 LEDs in Form einer 8x8-Matrix und eine in der Mitte der         **
;** Sanduhr, welche über einen Vorwiderstand mit Ub (=+5V) verbunden ist.                       **
;** Unterschiedliche Zeiteinstellung (von 15 Sekunden bis 60 Minuten), Summer nach Ablauf      **
;** der eingestellten Zeit.                                                                     **
;**                                                                                             **
;** Aenderungen/Ergaenzungen:                                                                   **
;** - Beep-Ton bei jedem Zustandswechsel im Code mittel Konstante einstellbar                   **
;** - Konfigurierbarer Schaltausgang (Port-Pin 0RA0) anstelle des Jumper (JP1)                  **
;**                                                                                             **
;** Entwickler: Buchgeher Stefan                                                                **
;** Entwicklungsbeginn der Software: 21. Juli 2001                                              **
;** Funktionsfaehig seit: 28. Oktober 2001                                                      **
;** Letzte Bearbeitung (der Ergaenzung): 5. Maerz 2004                                          **
;*************************************************************************************************


              List    p=PIC16F84

;******************************** Register (in Registerseite 0) ******************************
IND           equ     0                   ;Register für indirekte Adressierung
TMR0          equ     1                   ;Timer0-Register
PC            equ     2                   ;Programmzaehler
STAT          equ     3                   ;Statusregister
FSR           equ     4                   ;FSR-Register
PORTA         equ     5                   ;PortA-Register
PORTB         equ     6                   ;PortB-Register
INTCON        equ     0B                  ;Interrupt-Register


;******************************** Register (in Registerseite 1) ******************************
OPTREG        equ     1                   ;OPTION-Register
TRISA         equ     5                   ;Richtungsregister PortA
TRISB         equ     6                   ;Richtungsregister PortB


;******************************** Eigene Register (in Registerseite 0) ***********************
ZAEHLLED      equ     10                  ;Zählvariable (32 bis 0) gibt die Aktuelle
                                          ;  POS-Nr der leuchtende LED an
POS1          equ     11                  ;die folgenden 32 Register beinhalten die
POS2          equ     12                  ;  aktuellen "Adressen" der aktiven
POS3          equ     13                  ;  (leuchtenden) LEDs
POS4          equ     14
POS5          equ     15
POS6          equ     16
POS7          equ     17
POS8          equ     18
POS9          equ     19
POS10         equ     1A
POS11         equ     1B
POS12         equ     1C
POS13         equ     1D
POS14         equ     1E
POS15         equ     1F
POS16         equ     20
POS17         equ     21
POS18         equ     22
POS19         equ     23
POS20         equ     24
POS21         equ     25
POS22         equ     26
POS23         equ     27
POS24         equ     28
POS25         equ     29
POS26         equ     2A
POS27         equ     2B
POS28         equ     2C
POS29         equ     2D
POS30         equ     2E
POS31         equ     2F
POS32         equ     30

SUSTATUS      equ     31                  ;Sanduhr-Statusregister
                                          ;  Bit 0: 1 ... Starttaste wurde gedrueckt,
```

```
                                        ;              Körner(LEDs) rieseln nach unten
                                        ;        0 ... Starttaste nicht gedrückt
                                        ;  Bit 1: Hilfsflag: gesetzt: nächster Zustand
                                        ;  Bit 2: gesetzt: Zeit ist abgelaufen

AKTZEITH        equ     32              ;aktuell abgel. Zeit zw. 2 Zustandsänderungen
                                        ;  (High-Byte)
ZEITH           equ     33              ;eingestellte Zeit zwischen 2 Zuständen
                                        ;  (High-Byte)
AKTZEITL        equ     34              ;wie vorher, jedoch für das LOW-Byte
ZEITL           equ     35              ;wie vorher, jedoch für das LOW-Byte

AKTZUSTAND      equ     36              ;Aktueller Zustand der LEDs (Konstante ZUSTANZ
                                        ;  bis 0)
ZWISCH          equ     37              ;Zwischenspeicher für auszutauschende LED
DELAY           equ     38              ;Zählregister für Zeitverzögerungsschleife
w_TEMP          equ     39              ;Zwischenspeicher für w-Register bei ISR
STAT_TEMP       equ     3A              ;Zwischenspeicher für STATUS-Register bei ISR
SUCHZUST        equ     3B              ;Hilfsregister
SUCHZAEHLER     equ     3C              ;Zählregister von 32 bis 0, verhindert,
                                        ;das ausserhalb von POS bis POS32 gesucht wird
TEMP            equ     3D              ;temporaeres Hilfsregister


;******************************** Bits in Registern ****************************************
C               equ     0               ;Carrybit im Statuswort-Register
Z               equ     2               ;Zerobit im Statuswort-Register
RP0             equ     5               ;Seitenauswahlbit im Statuswort-Register
T0IF            equ     2               ;TMR0-Interruptflag im INTCON-Register


;******************************** Ziele der Registeroperationen ****************************
w               equ     0               ;w ist Zielregister
f               equ     1               ;f ist Zielregister


;******************************** Eigene Bits in Registern *********************************
STARTTASTE      equ     7               ;Port B, Bit 7
SUMMER          equ     6               ;Port B, Bit 6
;SUMMERSELEKT   equ     4               ;Port A, Bit 4 (Urversion)
SAUSGANG        equ     4               ;Port A, Bit 4 (Version mit Schaltausgang)


;******************************** Konstanten ***********************************************
LED1            equ     b'10111111'     ;"Adresse" von LED D1
LED2            equ     b'10111110'     ;"Adresse" von LED D2
LED3            equ     b'10111100'     ;"Adresse" von LED D3, usw.
LED4            equ     b'10111010'
LED5            equ     b'10111000'
LED6            equ     b'10110111'
LED7            equ     b'10111101'
LED8            equ     b'10111011'
LED9            equ     b'10111001'
LED10           equ     b'10100111'
LED11           equ     b'10110110'
LED12           equ     b'10110100'
LED13           equ     b'10110010'
LED14           equ     b'10110000'
LED15           equ     b'10101111'
LED16           equ     b'10110101'
LED17           equ     b'10110011'
LED18           equ     b'10110001'
LED19           equ     b'10100110'
LED20           equ     b'10101110'
LED21           equ     b'10101100'
LED22           equ     b'10101010'
LED23           equ     b'10101000'
LED24           equ     b'10100101'
LED25           equ     b'10101101'
LED26           equ     b'10101011'
LED27           equ     b'10101001'
LED28           equ     b'10100000'
LED29           equ     b'10100011'
LED30           equ     b'10100001'
LED31           equ     b'10100100'
LED32           equ     b'10100010'
LED33           equ     b'10011100'
LED34           equ     b'10011010'
LED35           equ     b'10011000'
```

```
LED36          equ     b'10011011'
LED37          equ     b'10011001'
LED38          equ     b'10011101'
LED39          equ     b'10010101'
LED40          equ     b'10010011'
LED41          equ     b'10010001'
LED42          equ     b'10011110'
LED43          equ     b'10010110'
LED44          equ     b'10010100'
LED45          equ     b'10010010'
LED46          equ     b'10010000'
LED47          equ     b'10010111'
LED48          equ     b'10001101'
LED49          equ     b'10001011'
LED50          equ     b'10001001'
LED51          equ     b'10011111'
LED52          equ     b'10001110'
LED53          equ     b'10001100'
LED54          equ     b'10001010'
LED55          equ     b'10001000'
LED56          equ     b'10001111'
LED57          equ     b'10000101'
LED58          equ     b'10000011'
LED59          equ     b'10000001'
LED60          equ     b'10000111'
LED61          equ     b'10000110'
LED62          equ     b'10000100'
LED63          equ     b'10000010'
LED64          equ     b'10000000'
ZUSTANZ        equ     .112                ;Anzahl der Zustände
LEDANZAKTIV    equ     .32                 ;Anzahl der leuchtenden LEDs

ZEIT0H         equ     .2                  ;High- und Low-Byte
ZEIT0L         equ     .5                  ;  für 15 Sekunden
ZEIT1H         equ     .3                  ;High- und Low-Byte
ZEIT1L         equ     .11                 ;  für 30 Sekunden
ZEIT2H         equ     .5                  ;High- und Low-Byte
ZEIT2L         equ     .22                 ;  für 60 Sekunden
ZEIT3H         equ     .7                  ;High- und Low-Byte
ZEIT3L         equ     .33                 ;  für 90 Sekunden
ZEIT4H         equ     .9                  ;High- und Low-Byte
ZEIT4L         equ     .44                 ;  für 2 Minuten
ZEIT5H         equ     .13                 ;High- und Low-Byte
ZEIT5L         equ     .67                 ;  für 3 Minuten
ZEIT6H         equ     .17                 ;High- und Low-Byte
ZEIT6L         equ     .89                 ;  für 4 Minuten
ZEIT7H         equ     .21                 ;High- und Low-Byte
ZEIT7L         equ     .111                ;  für 5 Minuten
ZEIT8H         equ     .33                 ;High- und Low-Byte
ZEIT8L         equ     .178                ;  für 8 Minuten
ZEIT9H         equ     .41                 ;High- und Low-Byte
ZEIT9L         equ     .223                ;  für 10 Minuten
ZEIT10H        equ     .50                 ;High- und Low-Byte
ZEIT10L        equ     .11                 ;  für 12 Minuten
ZEIT11H        equ     .62                 ;High- und Low-Byte
ZEIT11L        equ     .79                 ;  für 15 Minuten
ZEIT12H        equ     .82                 ;High- und Low-Byte
ZEIT12L        equ     .190                ;  für 20 Minuten
ZEIT13H        equ     .123                ;High- und Low-Byte
ZEIT13L        equ     .157                ;  für 30 Minuten
ZEIT14H        equ     .184                ;High- und Low-Byte
ZEIT14L        equ     .236                ;  für 45 Minuten
ZEIT15H        equ     .246                ;High- und Low-Byte
ZEIT15L        equ     .59                 ;  für 60 Minuten

;Ergaenzung fuer Schaltausgang anstelle des Jumpers JP1 - Start
BEEPMODE       equ     1                   ;1 = Beep bei jedem Zustandswechsel der LEDs
                                           ;0 = kein Beep bei Zustandswechsel

SAUSMODE       equ     1                   ;0 = kein Schaltausgang

                                           ;1 = Schaltausgang aktiv: Start des Rieselns
                                           ;    Schaltausgang deaktiv: Ablauf der Zeit
                                           ;                           oder Stopptaste

                                           ;2 = Schaltausgang aktiv: Ablauf der Zeit
                                           ;    Schaltausgang deaktiv: Stopptaste oder
                                           ;                           neuer Start
```

```
                                        ;3 = kurzer Impuls bei Abgelaufener Zeit

SAUSDAUER       equ     .20             ;Impulsdauer x 0.5 (in ms) nur bei SAUSMODE = 3
;Ergaenzung fuer Schaltausgang anstelle des Jumpers JP1 - Ende


;******************************** Konfigurations-Bits ***************************************
_lp_osc         equ     h'3FFC'
_xt_osc         equ     h'3FFD'
_hs_osc         equ     h'3FFE'
_rc_osc         equ     h'3FFF'

_wdt_off        equ     h'3FFB'
_wdt_on         equ     h'3FFF'

_pwrt_off       equ     h'3FFF'
_pwrt_on        equ     h'3FF7'

_cp_off         equ     h'3FFF'
_cp_on          equ     h'000F'

        __config        _hs_osc & _wdt_off & _pwrt_off & _cp_off

                ORG     0x000
                goto    Beginn
                ORG     0x004
                goto    ISR


;******************************** Tabellen *************************************************
TABWIRDERSETZT addwf   PC,f
               nop
               retlw   LED32
               retlw   LED31
               retlw   LED33
               retlw   LED30
               retlw   LED34
               retlw   LED28
               retlw   LED37
               retlw   LED34
               retlw   LED24
               retlw   LED35
               retlw   LED33
               retlw   LED27
               retlw   LED41
               retlw   LED37
               retlw   LED34
               retlw   LED23
               retlw   LED38
               retlw   LED35
               retlw   LED33
               retlw   LED19
               retlw   LED29
               retlw   LED36
               retlw   LED25
               retlw   LED36
               retlw   LED26
               retlw   LED40
               retlw   LED36
               retlw   LED22
               retlw   LED39
               retlw   LED36
               retlw   LED20
               retlw   LED45
               retlw   LED40
               retlw   LED36
               retlw   LED18
               retlw   LED43
               retlw   LED39
               retlw   LED36
               retlw   LED15
               retlw   LED47
               retlw   LED43
               retlw   LED39
               retlw   LED36
               retlw   LED14
               retlw   LED50
               retlw   LED45
               retlw   LED40
```

```
                retlw   LED36
                retlw   LED10
                retlw   LED36
                retlw   LED21
                retlw   LED44
                retlw   LED36
                retlw   LED16
                retlw   LED44
                retlw   LED36
                retlw   LED17
                retlw   LED49
                retlw   LED44
                retlw   LED36
                retlw   LED11
                retlw   LED48
                retlw   LED44
                retlw   LED36
                retlw   LED13
                retlw   LED54
                retlw   LED49
                retlw   LED44
                retlw   LED36
                retlw   LED6
                retlw   LED52
                retlw   LED48
                retlw   LED44
                retlw   LED36
                retlw   LED9
                retlw   LED59
                retlw   LED54
                retlw   LED49
                retlw   LED44
                retlw   LED36
                retlw   LED1
                retlw   LED56
                retlw   LED52
                retlw   LED48
                retlw   LED44
                retlw   LED36
                retlw   LED5
                retlw   LED44
                retlw   LED36
                retlw   LED4
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED2
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED7
                retlw   LED58
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED8
                retlw   LED57
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED12
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED3

TABNEUELED addwf        PC,f
                nop
                retlw   LED34
                retlw   LED33
                retlw   LED35
                retlw   LED33
                retlw   LED37
                retlw   LED34
                retlw   LED41
                retlw   LED37
                retlw   LED34
                retlw   LED38
                retlw   LED35
```

```
retlw  LED33
retlw  LED46
retlw  LED41
retlw  LED37
retlw  LED34
retlw  LED42
retlw  LED38
retlw  LED35
retlw  LED33
retlw  LED36
retlw  LED40
retlw  LED36
retlw  LED39
retlw  LED36
retlw  LED45
retlw  LED40
retlw  LED36
retlw  LED43
retlw  LED39
retlw  LED36
retlw  LED50
retlw  LED45
retlw  LED40
retlw  LED36
retlw  LED47
retlw  LED43
retlw  LED39
retlw  LED36
retlw  LED51
retlw  LED47
retlw  LED43
retlw  LED39
retlw  LED36
retlw  LED55
retlw  LED50
retlw  LED45
retlw  LED40
retlw  LED36
retlw  LED44
retlw  LED36
retlw  LED49
retlw  LED44
retlw  LED36
retlw  LED48
retlw  LED44
retlw  LED36
retlw  LED54
retlw  LED49
retlw  LED44
retlw  LED36
retlw  LED52
retlw  LED48
retlw  LED44
retlw  LED36
retlw  LED59
retlw  LED54
retlw  LED49
retlw  LED44
retlw  LED36
retlw  LED56
retlw  LED52
retlw  LED48
retlw  LED44
retlw  LED36
retlw  LED64
retlw  LED59
retlw  LED54
retlw  LED49
retlw  LED44
retlw  LED36
retlw  LED60
retlw  LED56
retlw  LED52
retlw  LED48
retlw  LED44
retlw  LED36
retlw  LED53
retlw  LED44
retlw  LED36
```

```
                retlw   LED58
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED57
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED63
                retlw   LED58
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED61
                retlw   LED57
                retlw   LED53
                retlw   LED44
                retlw   LED36
                retlw   LED62
                retlw   LED53
                retlw   LED44
                retlw   LED36


;******************************** ISR - Timer0 ***************************************

;*************************************************************************************************
;** Interrupt Service Routine:                                                                   **
;**                                                                                             **
;** w-Register (=Arbeitsregister) und Status-Register zwischenspeichern. In ZAEHLLED steht      **
;** die Nummer der auszugebende LED. Die Adresse dieser LED aus dem RAM laden (POS1 bis         **
;** POS32) und am Port B ausgeben. Da das FSR-Register auch im Hauptprogramm verwendet wird,    **
;** den Inhalt vor Aufruf der ISR wiederherstellen (SUCHZUST). Wenn Bit SUSTATUS.0 gesetzt      **
;** (d.h. die LEDs rieseln nach unten) die Zählregister AKTZEITL bzw. AKTZEITH bei jedem ISR    **
;** Aufruf um eins vermindern. Bei AKTZEITH=0 und AKTZEITL=0 ein Zustandswechsel der LEDs       **
;** erfolgen, daher das Bit SUSTATUS.1 setzen, und die Register AKTZEITL und AKTZEITH neu       **
;** laden.                                                                                      **
;** Das Timer-Interrupt-Flag T0IF wider löschen, und das w- bzw. Status-Register wiederher-     **
;** stellen.                                                                                    **
;*************************************************************************************************
ISR
PUSH            movwf   w_TEMP                  ;w-Register
                swapf   STAT,w                  ;  und auch das
                movwf   STAT_TEMP               ;  Status-Register retten

                decfsz  ZAEHLLED,f              ;Zaehlvar. um 1 erniedrigen
                goto    weiter
                movlw   LEDANZAKTIV             ;wenn ZAEHLLED=0, ZAEHLLED wieder mit der
                movwf   ZAEHLLED                ;Konstante LEDANZAKTIV (=32) laden
weiter          movlw   10
                addwf   ZAEHLLED,w
                movwf   FSR
                movf    IND,w                   ;akt. LED-Adresse aus RAM laden
                movwf   PORTB                   ;und am Port ausgeben

                movf    SUCHZUST,w
                movwf   FSR

                btfss   SUSTATUS,0              ;Wenn SUSTATUS.0 gesetzt
                goto    fertig

                decfsz  AKTZEITL,f              ;Akt. Zeit Low von
                goto    fertig                  ;LED-Zustandsänderungen erniedrigen
                decfsz  AKTZEITH,f              ;  High-Byte erniedrigen
                goto    fertig
                bsf     SUSTATUS,1              ;wenn 0: NaechstZust-Flag setzen
                movf    ZEITL,w                 ;und Register neu laden
                movwf   AKTZEITL                ;  ZEITL -> AKTZEITL
                movf    ZEITH,w                 ;  bzw.
                movwf   AKTZEITH                ;  ZEITH -> AKTZEITH

fertig          bcf     INTCON,T0IF             ;T0-Interrupftflag loeschen

POP             swapf   STAT_TEMP,w             ;Status-Register
                movwf   STAT                    ;  und
                swapf   w_TEMP,f                ;  w-Register
                swapf   w_TEMP,w                ;  wieder herstellen

                retfie
```

```
;******************************** Unterprogramme ***********************************************

;***********************************************************************************************
;** Initialisierung des Prozessor:                                                            **
;**     + TMR0-Vorteiler mit 2                                                                **
;**     + Ports: Port A: Urversion: alle Bits Eingang                                         **
;**                      Version mit Schaltausgang: Bit 0-3: Eingang, Bit 4: Ausgang          **
;**              Port B: Bit 0-6: Ausgang, Bit 7: Eingang                                     **
;**     + Diverse Register laden                                                              **
;***********************************************************************************************
INIT            clrf    TMR0                ;Timer0 auf 0 voreingestellt
                bsf     STAT,RP0            ;Registerseite 1
                movlw   b'00000000'         ;interner Takt, Vorteiler = 2 an TMR0
                movwf   OPTREG
                movlw   b'10000000'
                movwf   TRISB               ;Port B: Bit 0-5: Ausgang (LED-Matrix)
                                            ;       Bit 6  : Ausgang (Summer)
                                            ;       Bit 7  : Eingang (Starttaste)
;Urversion - Start
;               movlw   b'11111'            ;Port A: Eingang (Urversion)
;               movwf   TRISA
;               bcf     STAT,RP0            ;Registerseite 0
;Urversion - Ende

;Version mit Schaltausgang - Start
                movlw   b'01111'            ;Port A: Bit 0-3: Eingang (Zeitauswahl)
                                            ;       Bit 4  : Ausgang (Schaltausgang)
                movwf   TRISA
                bcf     STAT,RP0            ;Registerseite 0
                bsf     PORTA,SAUSGANG      ;Schaltausgang initialisieren setzen
;Version mit Schaltausgang - Ende

                movlw   b'00000000'
                movwf   SUSTATUS            ;Sanduhr-Statusregister = 0
                movlw   LEDANZAKTIV         ;ZAEHLLED mit der Konstante LEDANZAKTIV
                movwf   ZAEHLLED            ;laden (=32)
                return


;***********************************************************************************************
;** Ausgangszustand der LEDs (die unteren 32 LEDs leuchten, diese daher in die Register POS1 **
;** bis POS32 laden)                                                                          **
;***********************************************************************************************
ZUSTAND0        movlw   LED33
                movwf   POS1
                movlw   LED34
                movwf   POS2
                movlw   LED35
                movwf   POS3
                movlw   LED36
                movwf   POS4
                movlw   LED37
                movwf   POS5
                movlw   LED38
                movwf   POS6
                movlw   LED39
                movwf   POS7
                movlw   LED40
                movwf   POS8
                movlw   LED41
                movwf   POS9
                movlw   LED42
                movwf   POS10
                movlw   LED43
                movwf   POS11
                movlw   LED44
                movwf   POS12
                movlw   LED45
                movwf   POS13
                movlw   LED46
                movwf   POS14
                movlw   LED47
                movwf   POS15
                movlw   LED48
                movwf   POS16
                movlw   LED49
                movwf   POS17
```

```
                movlw   LED50
                movwf   POS18
                movlw   LED51
                movwf   POS19
                movlw   LED52
                movwf   POS20
                movlw   LED53
                movwf   POS21
                movlw   LED54
                movwf   POS22
                movlw   LED55
                movwf   POS23
                movlw   LED56
                movwf   POS24
                movlw   LED57
                movwf   POS25
                movlw   LED58
                movwf   POS26
                movlw   LED59
                movwf   POS27
                movlw   LED60
                movwf   POS28
                movlw   LED61
                movwf   POS29
                movlw   LED62
                movwf   POS30
                movlw   LED63
                movwf   POS31
                movlw   LED64
                movwf   POS32
                return


;*****************************************************************************************
;** Zustand nach drücken der Starttaste. Die oberen 32 LEDs leuchten, diese daher in die     **
;** Register POS1 bis POS32 laden                                                            **
;*****************************************************************************************
;
ZUSTAND1        movlw   LED1
                movwf   POS1
                movlw   LED2
                movwf   POS2
                movlw   LED3
                movwf   POS3
                movlw   LED4
                movwf   POS4
                movlw   LED5
                movwf   POS5
                movlw   LED6
                movwf   POS6
                movlw   LED7
                movwf   POS7
                movlw   LED8
                movwf   POS8
                movlw   LED9
                movwf   POS9
                movlw   LED10
                movwf   POS10
                movlw   LED11
                movwf   POS11
                movlw   LED12
                movwf   POS12
                movlw   LED13
                movwf   POS13
                movlw   LED14
                movwf   POS14
                movlw   LED15
                movwf   POS15
                movlw   LED16
                movwf   POS16
                movlw   LED17
                movwf   POS17
                movlw   LED18
                movwf   POS18
                movlw   LED19
                movwf   POS19
                movlw   LED20
                movwf   POS20
                movlw   LED21
                movwf   POS21
```

```
                movlw   LED22
                movwf   POS22
                movlw   LED23
                movwf   POS23
                movlw   LED24
                movwf   POS24
                movlw   LED25
                movwf   POS25
                movlw   LED26
                movwf   POS26
                movlw   LED27
                movwf   POS27
                movlw   LED28
                movwf   POS28
                movlw   LED29
                movwf   POS29
                movlw   LED30
                movwf   POS30
                movlw   LED31
                movwf   POS31
                movlw   LED32
                movwf   POS32
                return


;*****************************************************************************************
;** Eingestellte Zeit (HEX-Schalter) abfragen und die Register ZEIT bzw. AKTZEIT sowohl fuer **
;** das High als auch für das Low-Byte laden. Einstellbare Zeiten sind:                    **
;**    15, 30 sec 1,1 1/2,2,3,4,5,8,10,12,15,20,30,45,60 min                               **
;*****************************************************************************************
ZEITAUSWAHL
                btfss   PORTA,3             ;Wenn RA3 gesetzt:
                goto    Z8bis15             ;  Schalter zw. 8 und 15 ist eingestellt
                btfss   PORTA,2             ;  sonst, wenn RA2 gesetzt
                goto    Z4bis7              ;  Schalter zw. 4 und 7 ist eingestellt
                btfss   PORTA,1             ;  sonst, wenn RA1 gesetzt
                goto    Z2u3                ;  Schalter entweder Position 2 oder 3
                btfss   PORTA,0             ;  sonst, wenn RA0 gesetzt
                goto    ZEIT1               ;  Schalter ist in Position 1
                goto    ZEIT0               ;  sonst Schalter ist in Position 0

Z2u3            btfss   PORTA,0             ;Wenn RA0 gesetzt
                goto    ZEIT3               ;  Schalter ist in Position 3
                goto    ZEIT2               ;  sonst ist Schalter in Position 4

Z4bis7          btfss   PORTA,1             ;  analog zu vorher
                goto    Z6u7
                btfss   PORTA,0
                goto    ZEIT5
                goto    ZEIT4
Z6u7            btfss   PORTA,0
                goto    ZEIT7
                goto    ZEIT6

Z8bis15         btfss   PORTA,2
                goto    Z12bis15
                btfss   PORTA,1
                goto    Z10u11
                btfss   PORTA,0
                goto    ZEIT9
                goto    ZEIT8

Z10u11          btfss   PORTA,0
                goto    ZEIT11
                goto    ZEIT10

Z12bis15        btfss   PORTA,1
                goto    Z14u15
                btfss   PORTA,0
                goto    ZEIT13
                goto    ZEIT12
Z14u15          btfss   PORTA,0
                goto    ZEIT15
                goto    ZEIT14

ZEIT0           movlw   ZEIT0L              ;Konstante ZEIT0L (Low-Byte für 15 sec)
                movwf   ZEITL               ;  laden und die Register ZEITL
                movwf   AKTZEITL            ;  und AKTZEITL damit laden
                movlw   ZEIT0H              ;Konstante ZEIT0H (High-Byte für 15 sec)
```

```
                movwf   ZEITH                   ;  laden und die Register ZEITH
                movwf   AKTZEITH                ;  und AKTZEITH damit laden
                return
ZEIT1           movlw   ZEIT1L                  ;Konstante ZEIT1L (Low-Byte für 30 sec)
                movwf   ZEITL                   ;  laden und die Register ZEITL
                movwf   AKTZEITL                ;  und AKTZEITL damit laden
                movlw   ZEIT1H                  ;Konstante ZEIT1H (High-Byte für 30 Sec)
                movwf   ZEITH                   ;  laden und die Register ZEITH
                movwf   AKTZEITH                ;  und AKTZEITH damit laden
                return
ZEIT2           movlw   ZEIT2L                  ;wie vorher
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT2H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT3           movlw   ZEIT3L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT3H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT4           movlw   ZEIT4L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT4H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT5           movlw   ZEIT5L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT5H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT6           movlw   ZEIT6L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT6H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT7           movlw   ZEIT7L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT7H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT8           movlw   ZEIT8L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT8H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT9           movlw   ZEIT9L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT9H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT10          movlw   ZEIT10L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT10H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT11          movlw   ZEIT11L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT11H
                movwf   ZEITH
                movwf   AKTZEITH
```

```
                return
ZEIT12          movlw   ZEIT12L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT12H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT13          movlw   ZEIT13L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT13H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT14          movlw   ZEIT14L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT14H
                movwf   ZEITH
                movwf   AKTZEITH
                return
ZEIT15          movlw   ZEIT15L
                movwf   ZEITL
                movwf   AKTZEITL
                movlw   ZEIT15H
                movwf   ZEITH
                movwf   AKTZEITH
                return


;*****************************************************************************************
;** LED-Zustandswechsel                                                                  **
;** Die zu ersetzende LED (LED-Adresse) aus Tabelle TABWIRDERSETZT holen, diese Adresse im   **
;** RAM (POS1 bis POS32) suchen und durch die neue LED aus Tabelle TABNEUELED ersetzen.      **
;** Beep-Ton, wenn Eingang (RA4) low (Jumper gesteckt) Flag wieder löschen                 **
;*****************************************************************************************
NAECHSTZUST     movlw   11                      ;Adresse von POS1 in
                movwf   SUCHZUST                ;Zaehlregister SUCHZUST laden

                movlw   LEDANZAKTIV             ;Anzahl der leuchtenden (Aktiven) LEDs in den
                movwf   SUCHZAEHLER             ;SUCHZAEHLER laden

                movf    AKTZUSTAND,w            ;die zu ersetzende LED (LEDADRESSE)
                call    TABWIRDERSETZT          ;  aus Tabelle holen
                movwf   ZWISCH                  ;  und in ZWISCH speichern

SUCHEN          movf    SUCHZUST,w
                movwf   FSR
                movf    IND,w                   ;Tabellenwert (=ZWISCH) von IND subtrahieren
                subwf   ZWISCH,w
                btfsc   STAT,Z                  ;Z(ero)-Flag prüfen
                goto    NEUELED                 ;Wenn Z(ero)-Flag gesetzt:ZWISCH=IND -> LED,
                                                ;  auf Welche das FSR zeigt austauchen
                incf    SUCHZUST,f              ;SUCHZUST erhöhen und
                decfsz  SUCHZAEHLER,f           ;Such-Zaehler -1
                goto    SUCHEN
                goto    NAECHSTZUST             ;wenn 0 -> zu ersetzende LED konnte nicht ge-
                                                ;  funden werden, -> FEHLER! -> Gesamten
                                                ;  Suchvorgang wiederholen

NEUELED         movf    AKTZUSTAND,w
                call    TABNEUELED
                movwf   IND

                decfsz  AKTZUSTAND,f            ;Zustandszähler um 1 erniedrigen
                goto    FLAGLOESCHEN
                bsf     SUSTATUS,2              ;FertigFlag setzen und
                bcf     SUSTATUS,0              ;LED-rieseln-Status-Bit löschen
FLAGLOESCHEN    bcf     SUSTATUS,1              ;NaechstZust-Flag wieder löschen

;Urversion - Start
;BEEPTON        btfsc   PORTA,SUMMERSELEKT      ;wenn Summerselekt low (Jumper gesteckt)
;               goto    ZURUECK
;               bsf     PORTB,SUMMER            ;Beep-Ton
;               call    WARTESCHLEIFE           ;in Form eines Impulses
;               bcf     PORTB,SUMMER            ;ausgeben
;Urversion - Ende
```

```
;Version mit Schaltausgang - Start
BEEPTON         clrw
                addlw   BEEPMODE              ;Ist die Konstante BEEPMODE gesetzt

                btfsc   STAT,Z
                goto    ZURUECK
                bsf     PORTB,SUMMER          ;Beep-Ton
                call    WARTESCHLEIFE         ;in Form eines Impulses
                bcf     PORTB,SUMMER          ;ausgeben
;Version mit Schaltausgang - Ende

ZURUECK         return


;*****************************************************************************************
;** Summer summt                                                                        **
;** Ausgabe einer (Rechteck-)Frequenz von ca. 1 kHz                                     **
;*****************************************************************************************
SUMMERAKTIV     bsf     PORTB,SUMMER          ;Summer einschalten
                call    WARTESCHLEIFE
                bcf     PORTB,SUMMER          ;Summer ausschalten
                call    WARTESCHLEIFE
                return


;*****************************************************************************************
;** Warteschleife                                                                       **
;** Verzoegerungszeit: 0.5 ms bei 4 MHz                                                 **
;*****************************************************************************************
WARTESCHLEIFE   movlw   .164
                movwf   DELAY
VERZOEGERUNG    decfsz  DELAY,f
                goto    VERZOEGERUNG
                nop
                return


;*****************************************************************************************
;** Schaltausgang beim Druecken der Starttaste je nach Modus setzen oder ruecksetzen    **
;**     SAUSMODE = 0:  kein Schaltausgang                                               **
;**     SAUSMODE = 1:  Schaltausgang aktiv (low)                                        **
;**     SAUSMODE = 2:  Schaltausgang deaktiv (high)                                     **
;**     SAUSMODE = 3:  Schaltausgang bleibt unverändert                                 **
;*****************************************************************************************
SAUSBEISTART    clrw
                addlw   SAUSMODE              ;Konstante SAUSMODE in das Arbeitsregister
                btfsc   STAT,Z                ;Ist das Z(ero)-Flag gesetzt -> Mode 0
                goto    SAUSSTARTENDE         ;  -> Unterprogramm verlassen
                movwf   TEMP                  ;Arbeitsregister in temporaeres Register
                decf    TEMP,f                ;  kopieren, und um 1 vermindern
                btfsc   STAT,Z                ;Ist das (Z)ero-Flag gesetzt -> Mode 1
                goto    SAUSSTARTMODE1        ;  -> SAUSSTARTMODE1
                decf    TEMP,f                ;Andernfalls wieder um 1 vermindern
                btfsc   STAT,Z                ;Ist das (Z)ero-Flag gesetzt -> Mode 2
                goto    SAUSSTARTMODE2        ;  -> SAUSSTARTMODE2
                decf    TEMP,f                ;Andernfalls wieder um 1 vermindern
                btfsc   STAT,Z                ;Ist das (Z)ero-Flag gesetzt -> Mode 3
                goto    SAUSSTARTMODE3        ;  -> SAUSSTARTMODE3
                goto    SAUSSTARTENDE         ;Andernfalls falscher Mode -> Unterprogramm
                                              ;  verlassen

SAUSSTARTMODE1  bcf     PORTA,SAUSGANG        ;Schaltausgang im Mode 1 aktiv
                goto    SAUSSTARTENDE

SAUSSTARTMODE2  bsf     PORTA,SAUSGANG        ;Schaltausgang im Mode 1 nicht aktiv
                goto    SAUSSTARTENDE

SAUSSTARTMODE3  goto    SAUSSTARTENDE         ;Schaltausgang im Mode 3 unveraendert

SAUSSTARTENDE   return


;*****************************************************************************************
;** Schaltausgang nach Ablauf der Zeit je nach Modus setzen oder ruecksetzen            **
;**     SAUSMODE = 0:  kein Schaltausgang                                               **
;**     SAUSMODE = 1:  Schaltausgang deaktiv (high)                                     **
;**     SAUSMODE = 2:  Schaltausgang aktiv (low)                                        **
;**     SAUSMODE = 3:  kurzer low-Impuls (SAUSDAUER x 0.5 ms)                           **
;*****************************************************************************************
```

```
SAUSNACHZEIT   clrw
               addlw   SAUSMODE             ;Konstante SAUSMODE in das Arbeitsregister
               btfsc   STAT,Z               ;Ist das Z(ero)-Flag gesetzt -> Mode 0
               goto    SAUSENDEENDE         ;  -> Unterprogramm verlassen
               movwf   TEMP                 ;Arbeitsregister in temporaeres Register
               decf    TEMP,f               ;  kopieren, und um 1 vermindern
               btfsc   STAT,Z               ;Ist das (Z)ero-Flag gesetzt -> Mode 1
               goto    SAUSENDEMODE1        ;  -> SAUSENDEMODE1
               decf    TEMP,f               ;Andernfalls wieder um 1 vermindern
               btfsc   STAT,Z               ;Ist das (Z)ero-Flag gesetzt -> Mode 2
               goto    SAUSENDEMODE2        ;  -> SAUSENDEMODE2
               decf    TEMP,f               ;Andernfalls wieder um 1 vermindern
               btfsc   STAT,Z               ;Ist das (Z)ero-Flag gesetzt -> Mode 3
               goto    SAUSENDEMODE3        ;  -> SAUSENDEMODE3
               goto    SAUSENDEENDE         ;Andernfalls falscher Mode -> Unterprogramm
                                            ;  verlassen

SAUSENDEMODE1  bsf     PORTA,SAUSGANG       ;Schaltausgang deaktivieren
               goto    SAUSENDEENDE

SAUSENDEMODE2  bcf     PORTA,SAUSGANG       ;Schaltausgang aktivieren
               goto    SAUSENDEENDE

SAUSENDEMODE3  bcf     PORTA,SAUSGANG       ;Schaltausgang aktivieren ...
               movlw   SAUSDAUER            ;  ... Impulsdauer ...
               movwf   TEMP
SAUSSCHLEIFE1  decfsz  TEMP,f
               goto    SAUSSCHLEIFE2
               goto    SAUSENDEENDE
SAUSSCHLEIFE2  call    WARTESCHLEIFE
               goto    SAUSSCHLEIFE1

SAUSENDEENDE   bsf     PORTA,SAUSGANG       ;  ... Schaltausgang wieder deaktivieren
               return


;******************************** Hauptprogramm *************************************************

;************************************************************************************************
;** Aufgaben des Hauptprogramms:                                                               **
;** + Aufrufen von INIT (Initialisierungen)                                                    **
;** + Aufrufen von ZUSTAND0 (Ausgangszustand                                                   **
;** + Interrupts freigeben                                                                     **
;** + warten bis Starttaste gedrückt                                                           **
;** + Bei Version mit Schaltausgang: Schaltausgang je nach Modus setzen oder rücksetzen        **
;** + Aufrufen von ZUSTAND                                                                     **
;** + AKTZUSTAND=ZUSTANZ                                                                       **
;** + Aufrufen von ZEITAUSWAHL                                                                 **
;** + Wenn Zeit ist abgelaufen Summer aktivieren, bei Version mit Schaltausgang: Schalt-       **
;**   ausgang je nach Modus setzen oder rücksetzen und auf erneute Betaetigung der Starttaste **
;**   warten                                                                                   **
;** + Wenn Zeit ist nicht abgelaufen, warten bis das Zustandsänderungsflag gesetzt ist. Dann **
;**   NAECHSTZUST aufrufen                                                                     **
;************************************************************************************************
Beginn         call    INIT                 ;Initialisierungen
               call    ZUSTAND0             ;Unteren 32 LEDs leuchten (Ausgangszustand)
               movlw   b'10100000'          ;TMR0 freigeben durch
               movwf   INTCON               ;Setzen von GIE und T0IE im INTCON Register

START          btfsc   PORTB,STARTTASTE     ;Warten, bis Starttaste gedrueckt
               goto    START

HAUPTSCHLEIFE  bsf     SUSTATUS,0           ;Wenn Starttaste gerückt, Statusbit setzen
               bcf     SUSTATUS,2           ;  und Fertig-Flag löschen ...

;Version mit Schaltausgang - Start
               call    SAUSBEISTART         ;..Schaltausgang je nach Modus setzen
                                            ;  oder ruecksetzen ...
;Version mit Schaltausgang - Ende

               call    ZUSTAND1             ;  ... und die LEDs "springen" nach oben

               movlw   ZUSTANZ              ;die Konstante ZUSTANZ (Anzahl der möglichen
               movwf   AKTZUSTAND           ;  LED-Zustände) ins Register AKTZUSTAND kopieren

               call    ZEITAUSWAHL

WDH2           btfss   SUSTATUS,2           ;wenn Fertig-Flag gesetzt ...
               goto    WEITER
```

```
;Version mit Schaltausgang - Start
               call    SAUSNACHZEIT          ;  ... Schaltausgang je nach Modus setzen
                                             ;  oder ruecksetzen ...
;Version mit Schaltausgang - Ende

SCHLEIFE       call    SUMMERAKTIV           ;  ... und Summer aktivieren (Signalisierung
                                             ;  der abgelaufenen Zeit)
               btfsc   PORTB,STARTTASTE      ;warten, bis Starttaste gedrueckt
               goto    SCHLEIFE              ;Starttaste nicht gedrückt
               goto    HAUPTSCHLEIFE         ;Starttaste gedrückt

WEITER         btfss   SUSTATUS,1            ;Wenn Zustandsänderungs-Flag gesetzt
               goto    WDH2

               call    NAECHSTZUST           ;erfolgt ein Zustandswechsel der LEDs
               goto    WDH2

               end
```