Aufgaben zur Attiny-Platine

1. LEDs blinken

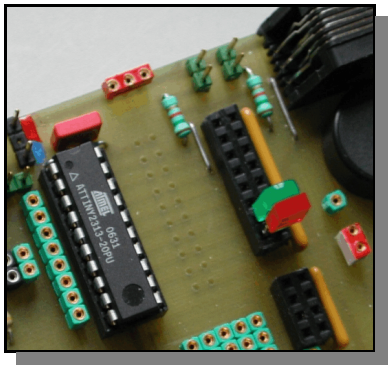


Abbildung 1

1.1 Schließen Sie eine rote LED an PortB.0 und eine grüne LED an PortB.1 an (vgl. Abb. 1). Achten Sie dabei darauf, dass die langen Anschlussbeine der LEDs zum Mikrocontroller hinweisen.

1.2 Öffnen Sie die Vorlage-Datei im Verzeichnis Source/BASCOM/Vorlage und speichern Sie sie unter einem neuen Namen (z. B. “wechselblinken”) in einem neuen Verzeichnis ab.

> ***Wichtiger Hinweis***: Benutzen Sie für ***jedes*** neue Projekt immer ein neues Verzeichnis. BASCOM legt beim Kompilieren zusätzliche Dateien an; diese liegen dann schön geordnet in diesem neuen Verzeichnis.

1.3 Ändern Sie den Kommentar am Anfang des Quelltextes ab.

> ***Dringende Empfehlung***: Schreiben Sie im Kommentar auf, welche Funktion das Programm hat und welche Ein- und Ausgänge wozu benutzt werden (sollen).

1.4 Schreiben Sie in den Bereich Hauptprogramm eine Endlosschleife, welche die rote LED blinken lässt; die An-Zeit soll doppelt so lange wie die Aus-Zeit sein.

1.5 Speichern - Kompilieren - Uploaden - Testen! Benutzen Sie ggf. die Schritt-für-Schritt-Anweisung der Kurzanleitung!

> ***Wichtiger Hinweis***: Beim Kompilieren wird der Quelltext automatisch gespeichert; dabei wird die vorige Version überschrieben. Ändern Sie also Ihren Quelltext und kompilieren ihn anschließend, so werden die Änderungen automatisch in die Datei übernommen!

1.6 Schalten Sie die Platine auch einmal aus und wieder an (ohne dabei T1 zu betätigen).

1.7 Schreiben Sie nun ein Programm, welches nur die grüne LED blinken lässt und testen Sie es aus.

1.8 Ändern Sie das Programm aus 1.7 so ab, dass An- und Auszeiten sehr klein sind (z. B. 10 ms). Was beobachten Sie nun nach dem Uploaden? Haben Sie eine Erklärung?

1.9 Schreiben Sie ein Programm, das die rote und grüne LED *abwechselnd* blinken lässt und testen Sie es aus. Diesmal sollen wieder größere An- und Auszeiten (z. B. 2 s) benutzt werden.

1.10 Nur für Freaks: Stellen Sie eine Rot-Gelb-Grün-Ampel mir der Attiny-Platine her.

1.11 Nur für Freak-Freaks: Stellen Sie ein Lauflicht mit der Attiny-Platine her.

2. Umgehen mit vollständigen Ports - Dualzahlen

2.1 Programmieren Sie eine (deutsche) Rot-Gelb-Grün Ampel (also inkl. Rot-Gelb-Phase). Überlegen Sie sich zunächst die zu den einzelnen Phasen gehörenden Dualzahlen.

Hinweis: bei der Zuordnung Rot → Bit 0; Gelb → Bit 1; Grün → Bit 2 entspricht der Rot-Gelb-Phase die Dualzahl &B00000011.

2.2 Überlegen Sie: Das Ampelprogramm aus 2.1 kann auch mit dem Bit-orientierten Befehl PortB.x erzeugt werden. Warum ist dies nicht so vorteilhaft?

2.3 Programmieren Sie eines dieser Lauflichter, z. B.

"Kid"	"Wow"	Ihre Kreation
10000001	00000000	
01000010	00000001	
00100100	00000011	
00011000	00000111	
und zurück usw.	...	
	11111111	
	11111110	
	11111100	
	...	

3. Sound

3.1 Verbinden Sie den Ausgang PortB.0 mit dem Beeper (vgl. Abb. 2). Laden Sie das Programm aus Aufgabe 1.7 auf den Attiny. Sie werden ein periodisches Knacken vernehmen. Beachten Sie auch die Information auf der nächsten Seite!

3.2 Erzeugen Sie Programme mit unterschiedlich hohen Tönen. Kann man auch Töne im Ultraschallbereich erzeugen?

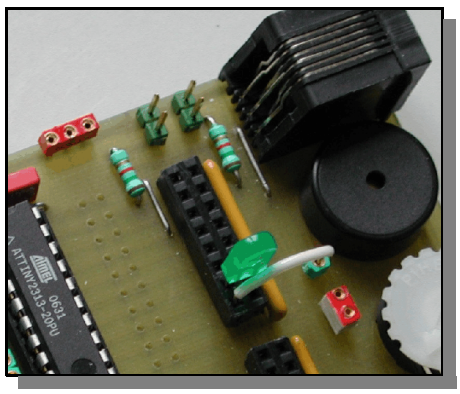


Abbildung 2

Information: Durch das Ein- und Ausschalten wird eine Membran im Beeper bewegt. Dies erzeugt die Knackgeräusche. Wird die Membran sehr rasch hin und her bewegt, so hört man einen Ton. Der Ton ist um so höher, je schneller die Membran sich bewegt.

3.3 Nur für Freak-Freak-Freaks: Erzeugen Sie einen Tatü-Tatü-Sound (nicht: Tatü-Tata!). Um unterschiedlich hohe Töne innerhalb einer einzigen Endlos-Schleife zu erzeugen, kann man eine Variable für die (halbe) Schwingungsdauer, eine Zählvariable und eine Verzweigung einsetzen.

4. Eingänge und Datenrichtungsregister

Bei den folgenden Aufgaben ist es wichtig, auf die korrekte Einstellung der Datenrichtungsregister zu achten. In der Vorlage-Datei finden wir folgende Standardeinstellungen:

```
'******************* Initialisierung ************************

Ddrb = &B11111111   'Port B als Ausgangsport
Ddrd = &B01110000   'D4, D5, D6 als Ausgang; Rest als Eingang
Portd = &B10001111  'Eingänge von D auf high legen
```

Diese Einstellungen sind für viele, aber eben nicht für alle Aufgaben geeignet. Kontrollieren Sie deswegen bei jeder Aufgabe nach, ob die Standardeinstellungen geeignet sind und ändern Sie sie ggf. ab.

4.1 Schließen Sie eine LED an PortB.0 an (vgl. Abb. 1). Diese LED soll über den Taster T0 ein- und ausgeschaltet werden. Genauer: Solange der Taster T0 gedrückt ist, soll die LED leuchten, sonst nicht.

4.2 Die LED aus 4.1 soll nun über eine Fotodiode an PortB.7 ein- und ausgeschaltet werden. Die Fotodiode stellt bei Lichteinfall eine gut leitende Verbindung zwischen seinen Anschlüssen her. Die Schaltung der Fotodiode ist in Abb. 3 dargestellt; für die Fotodiode wurde als Schaltsymbol ein Pfeil benutzt: kurzes Bein links, langes Bein rechts!

Die LED soll leuchten, wenn kein Licht auf die Fotodiode fällt. (Dämmerungsschalter)

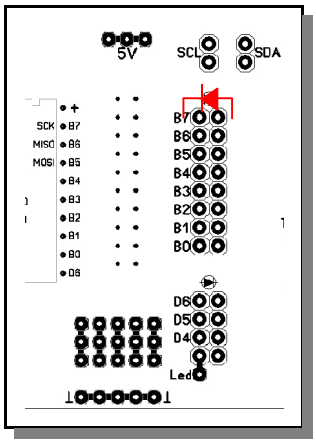


Abbildung 3

4.3 Wie muss das Programm aus 4.2 abgeändert werden, damit die LED leuchtet, wenn Licht auf die Diode fällt (und sonst nicht)? Testen Sie Ihre Idee aus.

4.4 Der Attiny soll die Anzahl der Tastendrücke beim Taster T0 an PortB anzeigen; PortB muss entsprechend mit 8 LEDs bestückt werden.

Beachten Sie:

- Ein Tastendruck dauert immer eine gewisse Zeit: Selbst bei flotten Menschen bleibt der Taster für mehrerer Millisekunden geschlossen; für den Attiny ist das aber eine halbe Ewigkeit!
- Informationsbox zum Prellen von Schaltern
- Mit `do <CR> loop until PinD.2=1` kann man den Mikrocontroller solange warten lassen, bis der Eingang D.2 auf 1 liegt.

4.5 Für Freaks: Jetzt soll mit dem Taster T0 dauerhaft geschaltet werden. Genauer heißt das: Wenn der Taster T0 kurz betätigt ist, soll die LED eingeschaltet werden und erst dann wieder ausgehen, wenn T0 ein zweites mal (kurz) betätigt wird (usw.).

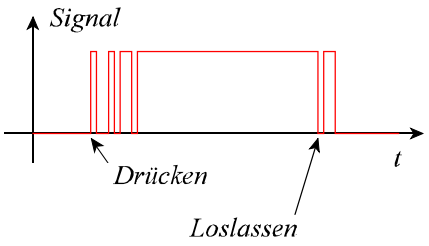
Beachten Sie:

- Die LED soll nicht nur einmal ein- bzw. ausgeschaltet werden können. Wir müssen also mit einer Endlosschleife arbeiten.
- Der Attiny muss sich den Schaltzustand der LED merken; dazu benutzen wir eine Variable.
- Informationsbox zum Prellen von Schaltern
- Mit `do <CR> loop until PinD.2=1` kann man den Mikrocontroller solange warten lassen, bis der Eingang D.2 auf 1 liegt.

4.6 Für Freak-Freaks: Je nachdem ob T0 und T1 betätigt wird, soll ein hoher oder tiefer Ton zu hören sein.

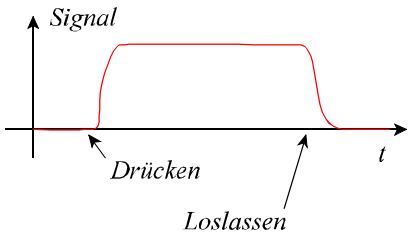
Information zum Prellen von Schaltern

Die meisten mechanischen Schalter (und dazu gehören auch unsere Taster T0 und T1) führen mehr als einen Schaltvorgang durch, auch wenn sie nur ein einziges Mal betätigt werden. Bei einem Taster wird das Schaltverhalten beim Drücken in etwa so aussehen:



Das liegt u. A. daran, dass sich eine gut leitende Verbindung nicht sofort, sondern erst nach einer kurzen Verzögerung einstellt. Dieses instabile Verhalten beim Ein- oder Ausschalten eines mechanischen Schalters bezeichnet man als **Prellen**.

Das Prellen eines Schalters kann unterdrückt werden, indem man parallel zum Schalter einen Kondensator einbaut. Die folgende Abbildung für das Schaltverhalten zeigt, dass das Prellen nicht mehr auftritt, allerdings das Signal etwas verzögert ist.



Auf unserer Platine ist der Taster T0 durch eine Kondensator entprellt, der Taster T1 hingegen nicht.