Drei Lösungen zur Übungsaufgabe 1: Hin-und-her

Lösung 1: Automatische Weiterschaltung nach 0,5 Sekunden (hin-und-her-1.asm)

```
;hin-und-her-1.asm
;Musterlösung Nr. 1 Automatischer Durchlauf
;09. Maerz 2011
;DH2MIC

.include "tn13def.inc"

;Es wird auf .def und .equ bewusst verzichtet ;um die Loesung moeglichst einfach darzustellen
;r16 und r17 sind fuer das Wartezeit-Unterprogramm reserviert
;r18 soll frei bleiben fuer groessere Wartezeiten mit 3 Schleifen
;r19 ist Hilfsregister fuer out-Befehle
;
;Die Ausgaenge werden zunaechst alle auf LOW gesetzt,
;Die Pins 2, 3 und 4 bleiben aber noch Eingaenge.
;Zur Anzeige wird das jeweilige Bit im Register ddrb auf 1 gesetzt und das vorherige wieder auf Null

        rjmp   Anfang
Anfang:
        ldi    r19,0b00000000   ;Alle Ausgaenge sollen auf LOW sein
        out    portb,r19

Schleife:                ;Beginn des Hauptprogramms
        cbi   ddrb,3     ;zwei mittlere LEDs an PB3 (wieder) aus
        sbi   ddrb,2     ;zwei schraege LEDs an PB2 an
        rcall Warte

        cbi   ddrb,2     ;zwei schraege LEDs an PB2 aus
        sbi   ddrb,3     ;zwei mittlere LEDs an
        rcall Warte

        cbi   ddrb,3     ;zwei mittlere LEDs aus
        sbi   ddrb,4     ;zwei andere schraege LEDs an PB4 an
        rcall Warte

        cbi   ddrb,4     ;zwei andere schraege LEDs wieder aus
        sbi   ddrb,3     ;zwei mittlere LEDs an
        rcall Warte

        rjmp  Schleife    ;zurueck zum Anfang

;Ende des Hauptprogramms
;Beginn der Unterprogramme

;Unterprogramm "Warte" ruft 20 mal das Unterprogramm "Warte25ms"

Warte:
        ldi   r18,20             ;warte 20x25ms = 0,5 sek.
Warte0:
        rcall Warte25ms      ;rufe Unterprogramm "25ms warten"
        dec   r18
        brne  Warte0
        ret
```

```
Warte25ms:              ;25ms (Millisekunden)
        ldi   r16,40
Warten1:                ;aeussere Warteschleife
        ldi   r17,250
Warten2:                ;innere Warteschleife
        dec   r17
        brne  Warten2   ;zurueck auf Warten2, solange r17 > 0
        dec r16
        brne  Warten1   ;zurueck auf Warten1, solange r16 > 0
        ret

;Ende des Programmes
```

Lösung 2: Manuelle Weiterschaltung mit der Taste (hin-und-her-2.asm)

```
;hin-und-her-2.asm
;Musterlösung Nr. 2
;09. Maerz 2011
;DH2MIC

.include "tn13def.inc"

;Es wird auf .def und .equ bewusst verzichtet, ;um die Loesung moeglichst einfach darzustellen
;r16 und r17 sind fuer das Wartezeit-Unterprogramm reserviert
;r18 soll frei bleiben fuer groessere Wartezeiten mit 3 Schleifen
;r19 ist Hilfsregister fuer out-Befehle
;
;Die Ausgaenge werden zunaechst alle auf LOW gesetzt,
;Die Pins 2, 3 und 4 bleiben aber noch Eingaenge.
;Zur Anzeige wird das jeweilige Bit im Register ddrb auf 1 gesetzt,
;wodurch der Anschluss zum Ausgang wird.
;Zuvor wird das vorherige Datenrichtungsbit wieder auf Null gesetzt,
;d.h. der Anschluss ist wieder ein Eingang

        rjmp   Anfang
Anfang:
        ldi    r19,0b00000000   ;Alle Ausgaenge sollen auf LOW sein
        out    portb,r19

Schleife:               ;Beginn des Hauptprogramms

        cbi   ddrb,3    ;zwei mittlere LEDs an PB3 (wieder) aus
        sbi   ddrb,2    ;zwei schraege LEDs an PB2 an
        rcall Taste

        cbi   ddrb,2    ;zwei schraege LEDs an PB2 aus
        sbi   ddrb,3    ;zwei mittlere LEDs an PB3 an
        rcall Taste

        cbi   ddrb,3    ;zwei mittlere LEDs aus
        sbi   ddrb,4    ;zwei andere schraege LEDs an PB4 an
        rcall Taste

        cbi   ddrb,4    ;zwei andere schraege LEDs wieder aus
        sbi   ddrb,3    ;zwei mittlere LEDs an
        rcall Taste

        rjmp  Schleife   ;zurueck zum Anfang

;Ende des Hauptprogramms
```

```
;Beginn der Unterprogramme

;Unterprogramm "Taste" prueft den Status der Taste an PB1
;Beim Aufruf des Unterprogrammes wurde gerade ein neuer Zusatnd
;eingestellt (Taste wurde gedrueckt). Deshalb muss zuerst gwartet
;werden, dass die Taste wieder geloest wird.
;Erst jetzt warten wir wieder darauf, dass die Taste wieder
;gedrueckt wird und wir das Unterprogramm beenden koennen, um
;den neuen Zustand einzustellen.

Taste:
noch_low:
        sbis  pinb,1      ;normal ist Taste noch gedrueckt wenn Unterprogramm
                          ;aufgerufen wird (kurz nach dem neuen Zustand der LEDs)
        rjmp  noch_low    ;immer noch gedrueckter Zustand der Taste
        rcall Warte25ms   ;warte 25ms Prellen ab, wenn Taste geloest wurde
noch_high:
        sbic  pinb,1       ;ok, jetzt wurde Taste gedrückt
        rjmp  noch_high   ;bleibe in der Schleife solange Taste geloest ist
        rcall Warte25ms   ;warte noch 25ms Prellen ab, nachdem Taste gedrueckt wurde
        ret               ;return und zeige neuen Zustand an

;Unterprogram "Warte25ms" erzeugt eine Wartezeit von 0,025 Sekunden

Warte25ms:          ;25ms (Millisekunden)
   ldi   r16,40
Warten1:            ;aeussere Warteschleife
   ldi   r17,250
Warten2:            ;innere Warteschleife
        dec   r17
        brne  Warten2     ;zurueck auf Warten2, solange r17 > 0
        dec r16
        brne  Warten1     ;zurueck auf Warten1, solange r16 > 0
        ret

;Ende des Programmes
```

Lösung 3: Manuelle Weiterschaltung mit der Taste und gemeinsames Anzeige-unterprogramm (hin-und-her-3.asm)

```
;hin-und-her-3.asm
;Musterlösung Nr. 3 Weiterschaltung mit der Taste und einheitliches Ausgabeprogramm
;09. Maerz 2011
;DH2MIC

.include "tn13def.inc"

;Es wird auf .def und .equ bewusst verzichtet
;um die Loesung moeglichst einfach darzustellen
;r16 und r17 sind fuer das Wartezeit-Unterprogramm reserviert
;r18 soll frei bleiben fuer groessere Wartezeiten mit 3 Schleifen
;r19 ist Hilfsregister fuer out-Befehle
;r20 dient der Datenausgabe
;
;Die Ausgaenge werden zunaechst alle auf LOW gesetzt,
;Die Pins 2, 3 und 4 werden auch als Ausgaenge deklariert.
;Zur Anzeige wird ein Bitmuster ins Register r20 geschrieben,
;damit alle drei Portbits mit einem out-Befehl auf einmal
;geschrieben werden können.
;Weil dabei aber manchmal der Anschluss PB3 als Eingang
;geschaltet werden muss, damit dort KEINE LED leuchtet,
;wird diese Info im Bit 7 dem Anzeigeunterprogramm mitgeteilt.

        rjmp   Anfang
Anfang:
        ldi    r19,0b00000000   ;Alle Ausgaenge sollen auf LOW sein
        out    portb,r19
        ldi    r19,0b00011100   ;PB2,PB3 und PB4 sollen Ausgaenge werden
        out    ddrb,r19         ;und das passiert hiermit

Schleife:           ;Beginn des Hauptprogramms

        ldi   r20,0b00011000    ;zwei schraege LEDs an PB2 an
                                ;und ddrb,3 auf LOW (Eingang)
        rcall Ausgabe
        rcall Taste

        ldi   r20,0b10010100    ;zwei mittlere LEDs an PB3 an
                                ;und ddrb,3 auf HIGH (Ausgang)
        rcall Ausgabe
        rcall Taste

        ldi   r20,0b00001100    ;zwei schraege LEDs an PB4 an
                                ;und ddrb,3 auf LOW (Eingang)
        rcall Ausgabe
        rcall Taste

        ldi   r20,0b10010100    ;zwei mittlere LEDs an PB3 an
                                ;und ddrb,3 auf HIGH (Ausgang)
        rcall Ausgabe
        rcall Taste

        rjmp  Schleife          ;zurueck zum Anfang

;Ende des Hauptprogramms
```

```
;Beginn der Unterprogramme

;Unterprogramm "Taste" prueft den Status der Taste an PB1
;Beim Aufruf des Unterprogrammes wurde gerade ein neuer Zusatnd
;eingestellt (Taste wurde gedrueckt). Deshalb muss zuerst gwartet
;werden, dass die Taste wieder geloest wird.
;Erst jetzt warten wir wieder darauf, dass die Taste wieder
;gedrueckt wird und wir das Unterprogramm beenden koennen, um
;den neuen Zustand einzustellen.

Ausgabe:
        cbi   ddrb,3      ;auf Verdacht PB3 als Eingang deklarieren
        sbrc  r20,7       ;ueberspringe naechten Befehl, wenn Annahme richtig war
                          ;sbrc besteht aus (S),(B),(R),(C) und bedeutet: Skip (S)
                          ;wenn Bit (B)(7) in Register (R)(r20) ist Cleared (C)
        sbi   ddrb,3      ;na gut, dann setzen wir das Bit eben auf 1
        andi  r20,0b01111111 ;loesche das Bit 7 fuer die Ausgabe an das PORTB-Register
        out   portb,r20   ;schreibe neues Bitmuster an alle Ausgaenge
        ret

Taste:
noch_low:
        sbis  pinb,1          ;normal ist Taste noch gedrueckt wenn Unterprogramm
                              ;aufgerufen wird (kurz nach dem neuen Zustand der LEDs)
        rjmp  noch_low        ;immer noch gedrueckter Zustand der Taste
        rcall Warte25ms       ;warte 25ms Prellen ab, wenn Taste geloest wurde
noch_high:
        sbic  pinb,1          ;ok, jetzt wurde Taste gedrückt
        rjmp  noch_high       ;bleibe in der Schleife solange Taste geloest ist
        rcall Warte25ms       ;warte noch 25ms Prellen ab, nachdem Taste gedrueckt wurde
        ret                   ;return und zeige neuen Zustand an

;Unterprogram "Warte25ms" erzeugt eine Wartezeit von 0,025 Sekunden

Warte25ms:                    ;25ms (Millisekunden)
        ldi   r16,40
Warten1:                      ;aeussere Warteschleife
        ldi   r17,250
Warten2:                      ;innere Warteschleife
        dec   r17
        brne  Warten2         ;zurueck auf Warten2, solange r17 > 0
        dec r16
        brne  Warten1         ;zurueck auf Warten1, solange r16 > 0
        ret

;Ende des Programmes
```