1 Fragen

- Was ist ROM?

  a Das Rom ist ein Speicher, dessen Inhalt nur vom Prozessor gelesen werden kann. Sein Inhalt lässt sich durch UV-Licht löschen.

- Wie funktioniert das Status-Register? (Info in den Themen Blättern)
- Welche Register gibt es noch?
- Können wir angeschrieben Programme mitnehmen oder nur als Textdokument?

2 Aufgaben

- HEX zahlen lernen. Wie rechnent man diese mit dem Taschenrechner?
- Die erste Aufgabe gibt 35-39,40 der Frage Katalog Meist bekommt man so 20 Punkte (Die Antworten können in Stichpunkten beantwortet werden)
- Kommentare zu Übungsaufgabe 1 hinzufügen

PORT RA und TRIS RA Initialisierungssequenz

```
1     BSF     status, rp0     ;auf Bank 1 umschalten, dort sind die
2     BCF     trisa, 0        ;TRIS-Register. RA0 wird Ausgang
3     BCF     trisa, 1        ;RA1 wird ebenfalls Ausgang
4     BCF     status, rq0     ;zurueck auf Bank 0 schalten
5     ...
6     ...
7     BSF     porta, 0        ;setzt den Pegel an RA0 auf high
8     BCF     porta, 1        ;setzt den Pegel an Ra1 auf low
```

Vergleich zweier Speicherstellen

```
1   MOVF    adr1, W         ;ein Argument ins W-Register holen
2   XORWF   adr2, W         ;XOR verknuepfen und Ergebnis in
3                           ;W-Register, so bleiben adr1 und
4                           ;adr2 unveraendert
5   BTFSC   status, Zflag
6   GOTO    sindGleich
7   GOTO    sindUngleich
```

Vergleich zweier Speicherstellen auf größer / kleiner

```
1   MOVF    adr1, W         ;ein Argument ins W-Register holen
2   SUBWF   adr2, W         ;subtrahiere W von Inhalt von adr2
3                           ;und schreib Ergebnis ins
4                           ;W-Register, so bleiben adr1 und adr2
5                           ;unveraendert
6   BTFSC   status,Zflag
7   GOTO    sindGleich
8   BTFSC   status, CFlag
9   GOTO    kleiner         ;es gab einen Ueberlauf im Carry
10  GOTO    groesser
```

5 Multiplikator

In `HWert1` steht wie viele überläufe es gab.51

```
CLRF    HWert1      ;Loescht HWert1
BCF     status, 0   ;Carryflag wird geloescht Carryflag ist auf 0
MOVF    LWert1, w   ;LWert in das W-Register
RLF     LWert1      ;verdoppelt den LWert
RLF     HWert1      ;moeglicher Ueberlauf der Operation wird in
                    ;HWert von rechts geschoben. Im Carray steht
                       ↪ 0.
RLF     LWert1      ;verdoppelt den LWert
RLF     HWert1      ;moeglicher Ueberlauf der Operation wird in
                    ;HWert von rechts geschoben. Im Carray steht
                       ↪ 0.
ADDWF   LWert       ;Der urspruengliche LWert wird noch dazu
   ↪ addiert
BTFSC   status, 0   ;Es wird geprueft ob Carry 0 ist wenn nicht
INCF    HWert1      ;wird HWert1 inkrementiert.
```

Eingangsimpuls erfassen

Schreiben Sie ein Assemblerprogramm für den 16C83, das einen Eingangsimpuls (0,1 ms bis 0,5 ms) erfasst und daraus einen 8 x so langen Ausgangsimpuls erzeugt. Der Ausgangsimpuls soll erst dann erscheinen, wenn der Eingangsimpuls wieder weg ist. Die Quarzfrequenz beträgt 4 MHz; ein einfacher Befehl benoetigt somit 1 μs.

`Pegel prüfen bis eine 1 kommt`

```
Label1
  BTFSC   porta, 0
  GOTO    Label1        ;warten auf low
Label2
  BTFSS   porta, 0
  GOTO    Label2        ;wenn das ueberspringt gab es eine
     ↪ ssteigende Flanke
                        ;Ab jetzt messen -> solange incrementieren

;8-bitzaehler geht bis 512 s

Label3
  INCF    Dauer, 1      ;1 Takt
  BTFSC   porta, 0      ;1
```

```
14    GOTO    Label3        ;2 Vier Takte fuer den letzten 3
15
16  ;Ausgabe
17  Label3
18    MOVLW   8             ;ein Register wird auf 8 gesetzt
19    MOVWF   Schleife      ;ist eine Variable
20    BSF     porta, 1
21  Label5
22    MOVF    Dauer, W
23    MOVWF   Counter
24  Label4
25    DECF    Counter
26    BTFSS   status,zero
27    GOTO    Label4
28    DECTSZ  Schleife
29    GOTO    Label5
30    BCF     porta, 1
31    GOTO    Label2
```

PIC-Programmierung Übungsaufgabe Nr 1

Aufgabe:

An einer Kegelbahn sollen die geworfenen Kegel als Zahlenwert angezeigt werden. In einem Unterprogramm werden dazu die liegenden Kegel gezählt und als BCD-Wert im W-Register dem Hauptprogramm zurückgegeben. Das Hauptschleife des Hauptprogramms besteht nur aus dem Unterprogrammaufruf. Eine Ausgangroutine ist nicht vorgesehen.

Hardwarebeschreibung:

Jeder Kegel, der richtig steht, schließt einen Schaltkontakt gegen Masse. Fällt der Kegel, liefert der entsprechende Kontakt somit ein High-Signal

Zuordnung Kegel → Eingänge:

RB 0

RB1 RB2

RB3 RA4 RB 4

RB5 RB6

RB7

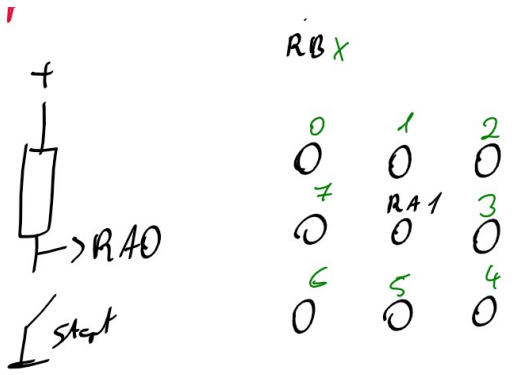
Sensor für Kugel am Lift:

RA0

```
1  Start
2      BSF     status, rp0
3      MOVLW   255
4      MOVWF   06/trisb
5      MOVLW   255
6      MOVWF   trisa
7      BCF     status, rp0
8      GOTO    Hauptprogramm
9
10 Hauptprogramm
11     CALL    Unterprogramm
12     GOTO    Hauptprogramm
13
14 Unterprogramm
15     CLRF    counter
16     BTFSC   RA0
17     RETLW   0
18     BTFSC   RA4
19     INCF    counter
20     MOVLW   8
21     MOVWF   loopCnt
22
23 Schleife
24     BTFSC   RB, 0
25     INCF    counter
26     RRF     RB
27     DECFSZ  loopCnt
28     GOTO    Schleife
29
30     MOVF    counter, W
31     RETURN
```

Gib das Bild eines Würfels aus

Es soll gezählt werden wie oft der RA0 Schalter gedrückt wird. Diese Zahl sollen mit einem Würfel ausgegeben werden.




```
1  Start
2      CLRF    counter
3      BSF     status, rp0
4      MOVLW   0           ;beide Zeilen koennte
5      MOVWF   06          ;man auch mit CLRF 06 ersetzten
6      BCF     05,1
7      BCF     status, rp0
8
9  Hauptprogramm
10     MOVLW   1
11     MOVWF   counter
12
13 Loop
14     BTFSC   RA,0
15     GOTO    Ausgabe
16     INCF    counter
17     MOVLW   7
18     XORWF   counter,w
19     BTFSC   status, 2
20     GOTO    Hauptprogramm
21     GOTO    Loop
22
23 Ausgabe
24     MOVF    counter, w
25     CALL    Tabelle1
26     MOVWF   portb
```

```
27      MOVF    counter, w
28      CALL    Tabelle2
29      MOVWF   porta
30      GOTO    Hauptprogramm
31
32  Tabelle1
33      ADDWF   pcl
34      nop
35      RETLW   0
36      RETLW   00010001b
37      RETLW   00010001b
38      RETLW   01010101b
39      RETLW   01010101b
40      RETLW   11011101b
41
42  Tabelle1
43      ADDWF
44      nop
45      RETLW   00000010b
46      RETLW   00000000b
47      RETLW   00000010b
48      RETLW   00000000b
49      RETLW   00000010b
50      RETLW   00000000b
```

Setzten von Ports

```
1   Start
2       BSF     status, rq0
3       MOVLW   0
4       MOVWF   trisb           ;RB0-RB7 wird Ausgang
5       BSF     trisa, 0        ;RA0 wird Eingang
6       BCF     status, rq0
7
8   Hauptprogramm
9       ;CLRF    RB              ;Setz alle Ausgaenge auf 0
10      BTFSS   RA, 0
11      GOTO    Hauptprogramm
12      CLRF    counter         ;setzt counter auf 0
13      INCF    counter         ;bring counter auf 1
14      MOVLW   counter
15      MOVWF   real_counter    ;setzt real_counter auf counter
16      CLRW                    ;setzt W-Register auf 0
17      GOTO    Schleife
```

```
18
19 Schleife
20     BTFSS    RA, 0
21     GOTO     Hauptprogramm
22     BTFSC    counter, 0
23     ADDLW    1
24     RRF      counter
25     GOTO     Schleife
26
27 Ausgabe
28     CALL     Tabelle
29     MOVWF    portb
30     INCF     real_counter
31     MOVLW    real_counter
32     MOVWF    counter
33     GOTO     Schleife
34
35 Tabelle
36     ADDWF    pcl
37     RETLW    1000 0000b
38     RETLW    1000 0001b
39     RETLW    1000 0011b
40     RETLW    1000 0111b
41     RETLW    1000 1111b
42     CLRF     real_counter
43     RETLW    1001 1111b
```

Testprog1

```
1  ;*******************************************************
2  ; Testprog1.src
3  ;*******************************************************
4    device 16f84
5  ;Symbol definieren
6  status     equ 3
7  zero       equ 2
8  rp0        equ 5
9  trisa      equ 5
10 trisb      equ 6
11 porta      equ 5
12 portb      equ 6
13
14 ;Hex-Zahlen: h am Ende, bei Zahlen mit Buchstaben an erster
     ↪ Stelle eine 0 davor
```

```
15  wert       equ 0ch
16  alterw       equ 13
17  counter    equ 14
18
19
20  org        0
21
22  ;Einsprung beim Einschalten (Power on)
23  cold
24    bsf      status,rp0   ;auf Bank 1 umschalten
25    movlw      0
26    movwf      trisb          ;PortB wird komplett als Ausgang
        ↪ geschaltet
27    bcf        trisa,3          ;RA3 wird Ausgang (Carry)
28    bcf      status,rp0   ;zurueck auf Bank 0
29
30  ;Definieren von alterw mit aktuellem Wert an RA0
31    movf       porta,w          ;PortA lesen
32    andlw      00000001b
33    movwf      alterw
34
35  ;Hauptschleife
36  loop
37    clrf       counter          ;Reset und Startwert
38    clrf       portb
39
40  loop1
41  ;Reset aktiv?
42    btfss      porta,1          ;Reseteingang
43    goto       loop
44  ;Inhibit aktiv?
45    btfsc      porta,2          ;Inhibiteingang
46    goto       loop1
47
48  ;Takteingang lesen
49    movf       porta,w          ;PortA komplett eingelesen
50    andlw      1              ;Nur R0 ist von Interesse
51
52    xorwf      alterw,w     ;Wenn beide gleich, keine Flanke
53    btfsc      status,zero    ;Beide gleich, Zero gesetzt
54    goto       loop1          ;Nichts passiert
55    movlw    1
56    xorwf      alterw           ;Beinhaltet neuen Pegel an RA0
57    btfss      alterw,0
58    goto       loop1
```

```
59
60    ;Richtige Flanke gefunden
61      bcf     porta,3
62      incf      counter          ;Zaehler erhoehen
63      movf      counter,w
64      movwf     portb
65      btfss     status,zero     ;Zaehlerueberlauf
66      goto      loop1
67      bsf     porta,3         ;Carryausgang setzen
68      goto      loop1
69
70      end
```

Testprog2

```
1  ;*******************************************************
2  ;  Testprog1.src
3  ;*******************************************************
4      device 16f84
5  ;Symbol definieren
6  status  equ 3
7  zero    equ 2
8  rp0     equ 5
9  trisa   equ 5
10 trisb   equ 6
11 porta   equ 5
12 portb   equ 6
13
14 ;Hex-Zahlen: h am Ende, bei Zahlen mit Buchstaben an erster
       ↪ Stelle eine 0 davor
15 wert    equ 0ch
16 alterw  equ 13
17 counter equ 14
18
19
20     org 0
21
22 ;Einsprung beim Einschalten (Power on)
23 cold
24     bsf status,rp0   ;auf Bank 1 umschalten
25     movlw   0
26     movwf   trisb        ;PortB wird komplett als Ausgang
           ↪ geschaltet
27     bcf     trisa,3      ;RA3 wird Ausgang (Carry)
```

```
28      bcf     status,rp0      ;zurueck auf Bank 0
29
30  ;Definieren von alterw mit aktuellem Wert an RA0
31      movf    porta,w     ;PortA lesen
32      andlw   00000001b
33      movwf   alterw
34
35  ;Hauptschleife
36  loop
37      clrf    counter     ;Reset und Startwert
38      clrf    portb
39  loop1
40
41  ;Reset aktiv?
42      btfss   porta,1     ;Reseteingang
43      goto    loop
44  ;Inhibit aktiv?
45      btfsc   porta,2     ;Inhibiteingang
46      goto    loop1
47
48  ;Takteingang lesen
49      movf    porta,w     ;PortA komplett eingelesen
50      andlw   1       ;Nur R0 ist von Interesse
51
52      xorwf   alterw,w    ;Wenn beide gleich, keine Flanke
53      btfsc   status,zero ;Beide gleich, Zero gesetzt
54      goto    loop1       ;Nichts passiert
55      movlw   1
56      xorwf   alterw      ;Beinhaltet neuen Pegel an RA0
57      btfss   alterw,0
58      goto    loop1
59
60  ;Richtige Flanke gefunden
61      bcf     porta,3     ;Carryausgang zuruecksetzen
62      incf    counter     ;Zaehler erhoehen
63      movlw   0fh
64      andwf   counter,w
65      xorlw   10
66      btfss   status,zero
67      goto    ausgabe
68      movlw   6
69      addwf   counter
70      movlw   0a0h
71      xorwf   counter,w
72      btfss   status,zero
```

```
73      goto    ausgabe
74      clrf    counter
75      bsf  porta,3
76
77  ausgabe
78      movf    counter,w
79      movwf   portb
80      goto    loop1
81
82      end
```

Testprog3

```
1   ;*******************************************************
2   ;  Testprog1.src
3   ;*******************************************************
4       device 16f84
5   ;Symbol definieren
6   status  equ 3
7   zero    equ 2
8   rp0     equ 5
9   trisa   equ 5
10  trisb   equ 6
11  porta   equ 5
12  portb   equ 6
13  pcl equ 2
14
15  ;Hex-Zahlen: h am Ende, bei Zahlen mit Buchstaben an erster
        ↪ Stelle eine 0 davor
16  wert    equ 0ch
17  alterw  equ 13
18  counter equ 14
19
20
21      org 0
22
23  ;Einsprung beim Einschalten (Power on)
24  cold
25      bsf status,rp0   ;auf Bank 1 umschalten
26      movlw   0
27      movwf   trisb        ;PortB wird komplett als Ausgang
            ↪ geschaltet
28      bcf     trisa,3      ;RA3 wird Ausgang (Carry)
29      bcf     status,rp0   ;zurueck auf Bank 0
```

```
30
31 ;Definieren von alterw mit aktuellem Wert an RA0
32     movf    porta,w     ;PortA lesen
33     andlw   00000001b
34     movwf   alterw
35
36 ;Hauptschleife
37 loop
38     clrf    counter     ;Reset und Startwert
39     clrf    portb
40 loop1
41
42 ;Reset aktiv?
43     btfss   porta,1     ;Reseteingang
44     goto    loop
45 ;Inhibit aktiv?
46     btfsc   porta,2     ;Inhibiteingang
47     goto    loop1
48
49 ;Takteingang lesen
50     movf    porta,w     ;PortA komplett eingelesen
51     andlw   1       ;Nur R0 ist von Interesse
52
53     xorwf   alterw,w    ;Wenn beide gleich, keine Flanke
54     btfsc   status,zero ;Beide gleich, Zero gesetzt
55     goto    loop1       ;Nichts passiert
56     movlw   1
57     xorwf   alterw      ;Beinhaltet neuen Pegel an RA0
58     btfss   alterw,0
59     goto    loop1
60
61 ;Richtige Flanke gefunden
62     bcf porta,3         ;Carryausgang zuruecksetzen
63     incf    counter     ;Zaehler erhoehen
64     movlw   0fh
65     andwf   counter,w
66     xorlw   10
67     btfss   status,zero
68     goto    ausgabe
69     movlw   6
70     addwf   counter
71     movlw   0a0h
72     xorwf   counter,w
73     btfss   status,zero
74     goto    ausgabe
```

```
75      clrf    counter
76      bsf     porta,3
77
78  ;Ausgabe auf Siebensegment ausgeben
79  ausgabe
80
81
82      movf    counter,w
83      call    convert
84      movwf   portb
85      bcf     porta,4
86      bsf     porta,4
87      swapf   counter,w
88      call    convert
89      movwf   portb
90      bcf     porta,5
91      bsf     porta,5
92      goto    loop1
93
94  ;Setzt Binaerzaehler in Bitmuster fuer Siebensegment um
95  convert
96      andlw   15
97      addwf   pcl     ;w=offset, der zum PCL addiert wird
98      retlw   3fh     ;0
99      retlw   06h     ;1
100     retlw   5bh     ;2
101     retlw   4fh     ;3
102     retlw   66h
103     retlw   6dh
104     retlw   7dh     ;...
105     retlw   07h
106     retlw   7fh
107     retlw   6fh     ;9
108     retlw   00h     ;ungueltig
109     retlw   00h
110     retlw   00h
111     retlw   00h
112     retlw   00h
113     retlw   00h
114
115     end
```

Testprog4

```
;*******************************************************
;  Testprog1.src
;*******************************************************
    device 16f84
;Symbol definieren
status  equ 3
zero    equ 2
carry   equ 0
rp0     equ 5
trisa   equ 5
trisb   equ 6
porta   equ 5
portb   equ 6
pcl equ 2
temp    equ 7

;Hex-Zahlen: h am Ende, bei Zahlen mit Buchstaben an erster
    ↪ Stelle eine 0 davor
wert    equ 0ch
alterw  equ 13
counter equ 14


    org 0

;Einsprung beim Einschalten (Power on)
cold
    bsf status,rp0  ;auf Bank 1 umschalten
    movlw   0
    movwf   trisb       ;PortB wird komplett als Ausgang
        ↪ geschaltet
    bcf     trisa,3     ;RA3 wird Ausgang (Carry)
    bcf     status,rp0  ;zurueck auf Bank 0

;Definieren von alterw mit aktuellem Wert an RA0
    movf    porta,w     ;PortA lesen
    andlw   00000001b
    movwf   alterw

;Hauptschleife
loop
    clrf    counter     ;Reset und Startwert
    clrf    portb
loop1

```

```
44 ;Reset aktiv?
45     btfss    porta,1      ;Reseteingang
46     goto     loop
47 ;Inhibit aktiv?
48     btfsc    porta,2      ;Inhibiteingang
49     goto     loop1
50
51 ;Takteingang lesen
52     movf     porta,w      ;PortA komplett eingelesen
53     andlw    1        ;Nur R0 ist von Interesse
54
55     xorwf    alterw,w     ;Wenn beide gleich, keine Flanke
56     btfsc    status,zero  ;Beide gleich, Zero gesetzt
57     goto     loop1        ;Nichts passiert
58     movlw    1
59     xorwf    alterw       ;Beinhaltet neuen Pegel an RA0
60     btfss    alterw,0
61     goto     loop1
62
63 ;Richtige Flanke gefunden
64     bcf porta,3           ;Carryausgang zuruecksetzen
65     incf     counter      ;Zaehler erhoehen
66     movlw    0fh
67     andwf    counter,w
68     xorlw    10
69     btfss    status,zero
70     goto     ausgabe
71     movlw    6
72     addwf    counter
73     movlw    0a0h
74     xorwf    counter,w
75     btfss    status,zero
76     goto     ausgabe
77     clrf     counter
78     bsf porta,3
79
80 ;Ausgabe auf Siebensegment ausgeben
81 ausgabe
82
83     movf     counter,w
84     call     convert
85 ;alternative Methode um RB0 frei zu bekommen besteht darin, die
       ↪ Ruecksprungadressen in der Tabelle zu aendern (verdoppeln)
86      movwf  temp
87      bcf    status,carry
```

```
88        rlf     temp,w
89      movwf     portb
90      bcf porta,4
91      bsf porta,4
92      swapf     counter,w
93      call      convert
94       movwf    temp
95        bcf     status,carry
96        rlf     temp,w
97      movwf     portb
98      bcf porta,5
99      bsf porta,5
100     goto      loop1
101
102 ;Setzt Binaerzaehler in Bitmuster fuer Siebensegment um
103 convert
104     andlw     15
105     addwf     pcl       ;w=offset, der zum PCL addiert wird
106     retlw     3fh       ;0
107     retlw     06h       ;1
108     retlw     5bh       ;2
109     retlw     4fh       ;3
110     retlw     66h
111     retlw     6dh
112     retlw     7dh       ;...
113     retlw     07h
114     retlw     7fh
115     retlw     6fh       ;9
116     retlw     00h       ;ungueltig
117     retlw     00h
118     retlw     00h
119     retlw     00h
120     retlw     00h
121     retlw     00h
122
123     end
```

Testprog5

```
1  ;*******************************************************
2  ;  Testprog1.src
3  ;*******************************************************
4      device 16f84
5  ;Symbol definieren
```

```
6  status  equ 3
7  zero    equ 2
8  carry   equ 0
9  rp0     equ 5
10 trisa   equ 5
11 trisb   equ 6
12 porta   equ 5
13 portb   equ 6
14 pcl equ 2
15 ;temp   equ 7
16
17 intcon  equ 0bh
18 inte    equ 4
19 intf    equ 1
20 gie equ 7
21
22 option  equ 1
23 intedg  equ 6
24
25 ;Hex-Zahlen: h am Ende, bei Zahlen mit Buchstaben an erster
       ↪ Stelle eine 0 davor
26 ;wert   equ 0ch
27 ;alterw equ 13
28 counter equ 0ch
29
30
31     org 0
32
33 ;Einsprung beim Einschalten (Power on)
34 cold
35 ;    bsf status,rp0  ;auf Bank 1 umschalten
36 ;    movlw   0
37 ;    movwf   trisb       ;PortB wird komplett als Ausgang
       ↪ geschaltet
38 ;    bcf     trisa,3     ;RA3 wird Ausgang (Carry)
39 ;    bcf     status,rp0  ;zurueck auf Bank 0
40
41
42
43     goto    main
44     nop
45     nop
46     nop
47 intup
48     btfss   intcon,intf ;war es ein RB0-Signal?
```

```
49      goto    intend      ;nein, Fehler
50      call    zaehlen
51      bcf intcon,intf ;Interrupt RB0 wird bearbeitet
52  intend
53      retfie
54
55  main
56      bsf status,rp0
57      movlw   00000001b
58      movwf   trisb
59      bcf trisa,3
60      bcf option,intedg
61      bcf status,rp0
62      bsf intcon,inte
63      bcf intcon,intf
64      bsf intcon,gie
65
66  ;Definieren von alterw mit aktuellem Wert an RA0
67  ;   movf    porta,w     ;PortA lesen
68  ;   andlw   00000001b
69  ;   movwf   alterw
70
71
72
73  ;Hauptschleife
74  loop
75      clrf    counter     ;Reset und Startwert
76      clrf    portb
77
78
79  loop1
80
81  ;Reset aktiv?
82      btfss   porta,1     ;Reseteingang (0 = Reset)
83      goto    loop
84  ;Inhibit aktiv?
85      btfsc   porta,2     ;Inhibiteingang
86      goto    loop2
87      bsf intcon,inte ;kein Inhibit -> RB0 Interrupt aktiv
88      goto    loop1
89
90  loop2
91      bcf intcon,inte ;RB0 Interrupt sperren
92      goto    loop1
93
```

```
94
95  ;Takteingang lesen
96  ;    movf    porta,w     ;PortA komplett eingelesen
97  ;    andlw   1       ;Nur R0 ist von Interesse
98
99  ;    xorwf   alterw,w    ;Wenn beide gleich, keine Flanke
100 ;    btfsc   status,zero ;Beide gleich, Zero gesetzt
101 ;    goto    loop1       ;Nichts passiert
102 ;    movlw   1
103 ;    xorwf   alterw      ;Beinhaltet neuen Pegel an RA0
104 ;    btfss   alterw,0
105 ;    goto    loop1
106
107 ;Richtige Flanke gefunden
108 zaehlen
109     bcf porta,3         ;Carryausgang zuruecksetzen
110     incf    counter     ;Zarehler erhoehen
111     movlw   0fh
112     andwf   counter,w
113     xorlw   10
114     btfss   status,zero
115     goto    ausgabe
116     movlw   6
117     addwf   counter
118     movlw   0a0h
119     xorwf   counter,w
120     btfss   status,zero
121     goto    ausgabe
122     clrf    counter
123     bsf porta,3
124
125 ;Ausgabe auf Siebensegment ausgeben
126 ausgabe
127
128     movf    counter,w
129     call    convert
130 ;alternative Methode um RB0 frei zu bekommen besteht darin, die
    ↪ Ruecksprungadressen in der Tabelle zu aendern (verdoppeln)
131     movwf   portb
132     bcf porta,4
133     bsf porta,4
134     swapf   counter,w
135     call    convert
136     movwf   portb
137     bcf porta,5
```

```
138     bsf  porta,5
139     return
140
141 ;Setzt Binaerzaehler in Bitmuster fuer Siebensegment um
142 convert
143     andlw   15
144     addwf   pcl     ;w=offset, der zum PCL addiert wird
145     retlw   7eh     ;0
146     retlw   0ch     ;1
147     retlw   0b6h        ;2
148     retlw   9eh     ;3
149     retlw   0cch
150     retlw   0dah
151     retlw   0fah        ;...
152     retlw   0eh
153     retlw   0feh
154     retlw   0deh        ;9
155     retlw   00h     ;ungueltig
156     retlw   00h
157     retlw   00h
158     retlw   00h
159     retlw   00h
160     retlw   00h
161
162     end
```

Testprog6

```
1  ;***********************************************
2  ;    INTERRUPTS
3  ;***********************************************
4
5      device  16f84
6
7  intcon  equ 0bh
8  inte    equ 4
9  intf    equ 1
10 gie equ 7
11
12 status  equ 3
13 rp0 equ 5
14
15 option  equ 1
16 intedg  equ 6
```

```
17
18 counter equ 0ch
19
20 ;Startadressen fuer Programm
21     org 0
22 cold
23     goto    main
24     nop
25     nop
26     nop
27 intup
28     btfss   intcon,intf ;war es ein RB0-Signal?
29     goto    intend      ;nein, Fehler
30     bcf     intcon,intf ;Interrupt RB0 wird bearbeitet
31     incf    counter     ;Zaehler erhoehen
32 intend
33     retfie
34
35
36 main
37 ;Interrupts initialisieren
38
39     bsf     intcon,inte     ;RB0-Interrupt freigeben
40     bcf     intcon,intf     ;Interruptflag loeschen
41     bsf     status,rp0      ;auf Bank 1 umschalten
42     bcf     option,intedg   ;auf fallende Flanke pruefen
43     bcf     status,rp0      ;zurueck auf Bank 0 schalten
44
45     bsf intcon,gie  ;Globales Interrupt enable Bit
46
47 loop
48     goto    loop
```

Testprog7

```
1 ;*********************************
2 ;   AUFGABE 12
3 ;*********************************
4
5 device 16f84
6 ;Symbole definieren
7 delaycnt    equ
8
9 delay166
```

```
10      movlw   52
11      movwf   delaycnt
12  delay166_a
13      decfsz  delaycnt
14      goto delay166_a
15      return
16
17
18  delay625
19      movlw   206
20      movwf   delaycnt
21  delay625_a
22      decfsz  delaycnt
23      goto    delay625_a
24      return
25
26  warte_null
27      btfss   rb,4            ;Nulldurchgang?
28      goto    warte_null
29  ;Nulldurchgang gefunden
30      call    delay166
31  ;Schalter einlesen
32      movf    portb,w         ;Schalter = 0?
33      andlw   15              ;Obere 4 Bits auf 0 setzen
34      btfsc   status,zero
35      goto    warte_null
36
37      movwf   loopcnt
38
39  loop
40      call    delay625
41      decfsz  loopcnt
42      goto    loop
43
44  ;TRIAC einschalten
45      bsf     rb,5
46      call    delay166
47      bcf     rb,5
48      goto    warte_null
```