

```

1 ; /***** start *****/
2     device 16f84
3
4     pcl equ 2 ; program counter
5     fsr equ 4 ; Indirect data memory address pointer 0.
6     indf equ 0; Uses contents of FSR to address data memory (not a physical register).
7
8     status equ 3
9     rp0 equ 5 ; bank select
10    carry equ 0 ; carry flag - Indicates when an arithmetic carry or borrow has been generated out of the
    ↳ most significant ALU bit position.
11    zero equ 2 ; zero flag - Indicates that the result of an operation was zero.
12
13    porta equ 5
14    ra0 equ 0 ; RA0
15    portb equ 6
16
17 ; vars
18    var equ 10h
19
20    org 0
21
22    cold
23        bsf status,rp0 ; select bank 1
24        ; port initialization
25        bcf status,rp0 ; select bank 0
26    mainloop
27 ; /***** start *****/
28
29
30 ; /***** loop *****/
31    movlw 4
32    movwf foo
33    loop ; foo - 1 iterations
34        ; body
35        decfsz foo ; decrement f, skip if 0
36        goto loop ; not null
37 ; /***** loop *****/
38
39
40 ; /***** equals *****/
41    movf var1,w
42    xorwf var2,w
43    btfsc status,zero
44    goto equal
45    goto not_equal
46 ; /***** equals *****/
47
48
49 ; /***** number comparison *****/
50    movf foo,w
51    subwf bar,w ; difference
52    btfsc status,zero ; equal?
53    goto equal ; yes
54    ; no
55    btfsc status,carry; (2 complement)
56    goto fooSmaller ; C = 1 → foo < bar
57    goto fooGreater ; C = 0 → foo > bar
58 ; /***** number comparison *****/
59
60
61
62
63

```

```

64 ; /***** indirect *****/
65 movlw 12h ; adress
66 movwf fsr
67 movf indf ; get value in 12h
68 incf indf
69 ; /***** indirect *****/
70
71
72 ; /***** software wait *****/
73 ; 1 ms = 1000 µs; 1 basic command = 1 µs; commands with jump = 2 µs
74 ; initialization
75     movlw 101 ; 1 µs; number of delay iterations
76     movwf delayCounter ; 1 µs
77 wait ; 1 Iteration 3µs - 100 iterations
78     decfsz delayCounter ; 1 µs
79     goto wait ; 2 µs
80     ; time = 2 µs + 100 * 3 µs = 302 µs
81 ; /***** software wait *****/
82
83
84 ; /***** check edge *****/
85 checkEdge
86     ; read current value
87     movf porta,w
88     andlw mask ; mask clock signal on RA0
89     movwf currentValue
90     xorwf oldValue,w ; compare with oldValue
91     movwf edge
92
93     ; oldValue := currentValue
94     movf currentValue,w
95     movwf oldValue
96
97     ; edge = 0 if currentValue = oldValue
98     movf edge ; set zero-flag if edge = 0
99     btfsc status,zero ; edge 0?
100    retlw 0 ; no → no new edge
101
102    movf currentValue ; set zero-flag if currentValue = 0
103    btfss status,zero ; currentValue = 0? or zero = 1?
104    retlw 2 ; no → rising edge
105    retlw 1 ; yes → falling edge
106 ; /***** check edge *****/

```

BCD-Zähler

```
1 ; BCD counter @ Valerio Cocco
2
3     device 16f84
4
5 ; \***** labels *****\
6
7 ; status regsiter
8 status equ 3 ; adress of the status register
9 rp0 equ 5 ; bank select
10 carry equ 0 ; carry flag - Indicates when an arithmetic carry or borrow has been generated out of the
   ↳ most significant ALU bit position.
11 zero equ 2 ; zero flag - Indicates that the result of an operation was zero.
12
13 ; port a
14 porta equ 5
15 clock equ 0 ; clock signal in RA0
16 reset equ 1 ; reset in RA1
17 inhibit equ 2 ; inhibit in RA2
18 carryOut equ 3 ; carry out RA3
19 mask equ 1 ; 00000001 ; mask for the clock signal on RA0
20
21 portb equ 6
22 ; TRIS (TRIState regsiter) Used to define the direction (in/out) of port or pin.
23 trisa equ 5 ; for port a
24 trisb equ 6 ; for port b
25
26 bcdOverfVal0 equ 10 ; 0000 1010
27 bcdOverfVal1 equ 0A0h ; 1010 0000
28 bcdMask equ 0Fh ; 0000 1111, mask for the first bcd digit
29
30 ; variables
31 counter equ 10h ; 0c first aviable adress
32 currentValue equ 12h
33 oldValue equ 13h
34 edge equ 14h
35
36 ; \***** labels *****\
37
38     org 0 ; program start at adress 0
39
40 cold
41     ; initialize ports
42     bsf status,rp0 ; select bank 1
43     bcf trisa,carryOut ; set carry on port a to out
44     clrf trisb ; set port all to out
45
46     bcf status,rp0 ; select bank 0
47
48     ; read first value
49     movf porta,w ; read port a in w
50     andlw mask ; mask clock signal
51     movwf oldValue ; write w regsiter to oldValue: first comparison value
52
53 resetCNT
54     clrf counter ; init
55     bcf porta,carryOut ; reset carry
56
57     clrf portb
58     bcf porta,carryOut ; output carry 0
59
60 mainloop
61     ; output BCD
62     movf counter,w
63     movwf portb
```

```

64
65     btfsc porta,reset ; reset in = 1?
66     goto resetCNT ; yes → reset
67
68     btfsc porta,inhibit ; inhibit = 1?
69     goto mainloop ; yes → pause
70
71     call checkEdge ; edge? no → w = 0,
72     xorlw 2 ; w = 2 (rising redge)? set zero flag if w = 2
73     btfss status,zero ; w = 2?
74     goto mainloop ; no
75
76     bcf porta,carryOut ; output carry 0
77
78     ; increment BCD
79     incf counter ; first digit
80     movf counter,w
81     andlw bcdMask ; mask first digit
82     xorlw bcdOverfVal0 ; overflow on first digit?
83     btfss status,zero
84     goto mainloop ; no
85     ; yes
86     movf counter,w
87     xorlw bcdOverfVal0 ; set first nibble to 0
88     addlw 10h ; increment second nibble
89     movwf counter
90
91     xorlw bcdOverfVal1 ; overflow on second digit?
92     btfss status,zero
93     goto mainloop ; no
94     ; yes
95     clrf counter
96     bsf porta,carryOut
97
98     goto mainloop
99
100 checkEdge
101     ; read current value
102     movf porta,w
103     andlw mask ; mask clock signal on RA0
104     movwf currentValue
105     xorwf oldValue,w ; compare with oldValue
106     movwf edge
107
108     ; oldValue := currentValue
109     movf currentValue,w
110     movwf oldValue
111
112     ; edge = 0 if currentValue = oldValue
113     movf edge ; set zero-flag if edge = 0
114     btfsc status,zero ; edge 0?
115     retlw 0 ; no → no new edge
116
117     movf currentValue ; set zero-flag if currentValue = 0
118     btfss status,zero ; currentValue = 0? or zero = 1?
119     retlw 2 ; no → rising edge
120     retlw 1 ; yes → falling edge

```

BCD Siebensegment

```
1 ; BCD to seven segment display @ Valerio Cocco
2
3     device 16f84
4
5 ; \***** labels *****\
6 pcl equ 2 ; program counter
7
8 status equ 3 ; status register
9 rp0 equ 5 ; bank select
10 carry equ 0 ; carry flag - Indicates when an arithmetic carry or borrow has been generated out of the
   ↳ most significant ALU bit position.
11 zero equ 2 ; zero flag - Indicates that the result of an operation was zero.
12
13 ; input: port A input
14 porta equ 5
15 trisa equ 5
16 ra6 equ 6
17 bcdmask equ 0Fh ; 00001111
18
19 ; output: port B
20 portb equ 6
21 trisb equ 6
22
23 ; variables
24 bcdin equ 10h
25 ; \***** labels *****\
26
27     org 0
28
29 cold
30     ; init
31     bsf status,rp0 ; select bank 1
32     clrf trisb ; set port b to output, RB0 is LSB
33     bcf trisa,ra6 ; set RA6 to output: digit 0 select
34     bcf status,rp0 ; select bank 0
35
36     bsf porta,ra6
37
38 mainloop
39     movf porta,w
40     andlw bcdmask
41     movwf bcdin
42     call bcdToSsd
43     movwf portb
44     bcf porta,ra6 ; output portb to digit 0
45     bsf porta,ra6 ; lock output from protb to digit
46     goto mainloop
47
48 bcdToSsd ; Sevent Segment Display
49     movf bcdin,w
50
51     addwf pcl ; unsave: undefined behavior if bcdin > 9
52     retlw 3Fh ; 0
53     retlw 06h ; 1
54     retlw 5Bh ; 2
55     retlw 4Fh ; 3
56     retlw 66h ; 4
57     retlw 6Dh ; 5
58     retlw 7Dh ; 6
59     retlw 07h ; 7
60     retlw 7Fh ; 8
61     retlw 6Fh ; 9
62     ; retlw 0 ; B
63     ; ...
```

Frequenzteiler

```
1 ; frequency divider @ Valerio Cocco
2
3     device 16f84
4
5 ; \***** labels *****\
6 status equ 3 ; status register
7 rp0 equ 5 ; bank select
8 carry equ 0 ; carry flag - Indicates when an arithmetic carry or borrow has been generated out of the
    ↳ most significant ALU bit position.
9 zero equ 2 ; zero flag - Indicates that the result of an operation was zero.
10
11 porta equ 5
12 output equ 1 ; out an ra1
13 trisa equ 5
14
15 currentValue equ 10h
16 oldValue equ 11h
17 edge equ 12h
18
19 mask equ 00000001b
20 ; \***** labels *****\
21
22 cold
23     bsf status,rp0 ; select bank 1
24     bcf trisa,output
25     bcf status,rp0 ; select bank 0
26
27     ; read initial value
28     movf porta,w
29     andlw mask
30     movwf oldValue
31
32 mainloop
33     call checkEdge
34     xorlw 2 ; rising edge?
35     btfss status,zero
36     goto mainloop ; no
37     ; yes
38     movlw 10b
39     xorwf porta
40     goto mainloop
41
42 checkEdge
43     movf porta,w
44     andlw mask
45     movwf currentValue
46     xorwf oldValue,w
47     movwf edge
48
49     ; oldValue := currentValue
50     movf currentValue,w
51     movwf oldValue
52
53     ; edge = 0 if currentValue = oldValue
54     movf edge ; set zero-flag if edge = 0
55     btfsc status,zero ; edge 0?
56     retlw 0 ; no → no new edge
57
58     movf currentValue ; set zero-flag if currentValue = 0
59     btfss status,zero ; currentValue = 0? / zero = 1?
60     retlw 2 ; no → rising edge
61     retlw 1 ; ja → falling edge
```

```

1 ; km counter
2
3     device 16f84
4
5 ; \***** labels *****\
6 pcl equ 2 ; program counter
7
8 status equ 3 ; status register
9 rp0 equ 5 ; bank select
10 carry equ 0 ; carry flag - Indicates when an arithmetic carry or borrow has been generated out of the
   ↳ most significant ALU bit position.
11 zero equ 2 ; zero flag - Indicates that the result of an operation was zero.
12
13 fsr equ 4 ; Indirect data memory address pointer 0.
14
15 indf equ 0; Uses contents of FSR to address data memory (not a physical register).
16
17 ; input: port A input
18 porta equ 5
19 trisa equ 5
20 ra0 equ 0
21 ra1 equ 1
22 ra2 equ 2
23 ra3 equ 3
24
25 ; literals
26 sigMask equ 00001000b ;
27
28 ; output: port B
29 portb equ 6
30 trisb equ 6
31
32
33 ; variables
34 edge equ 8h
35 bcdin equ 9h
36 signalCounter equ 10h
37 currentVal equ 11h
38 oldVal equ 12h
39 mCounter equ 13h
40 kmCounter equ 16h
41
42 ; \***** labels *****\
43
44     org 0
45
46 cold
47     ; init ports
48     bsf status,rp0 ; select bank 1
49     clrf trisb ; set port b to output, RB0 is LSB
50     bcf trisa,ra0 ; set RA0 to output: digit 0 select
51     bcf trisa,ra1 ; digit 1 select
52     bcf trisa,ra2 ; digit 2 select
53     bsf trisa,ra3 ; sig in
54     bcf status,rp0 ; select bank 0
55
56     clrf portb
57     ; lock every digit select
58     bsf porta,ra0
59     bsf porta,ra1
60     bsf porta,ra2
61
62     ; init variables
63     clrf signalCounter

```

```

64     clrf mCounter ; 1x m at 13h
65     clrf 14h ; 1xx m
66     clrf 15h ; km
67     clrf kmCounter ; 1x km at 16h
68
69     ; read first value
70     movf porta,w ; read port a in w
71     andlw sigMask
72     movwf oldVal
73
74 outputSsd
75     ; output digit 0
76     movf 14h,w
77     movwf bcdin
78     call bcdToSsd
79     movwf portb
80     bcf porta,ra0
81     bsf porta,ra0
82     ; digit 1
83     movf 15h,w
84     movwf bcdin
85     call bcdToSsd
86     movwf portb
87     bcf porta,ra1
88     bsf porta,ra1
89     ; digit 2
90     movf kmCounter,w
91     movwf bcdin
92     call bcdToSsd
93     movwf portb
94     bcf porta,ra2
95     bsf porta,ra2
96
97 mainloop
98     ; check signal
99     call checkEdge ; edge? no → w = 0,
100    xorlw 2 ; w = 2 (rising redge)? set zero flag if w = 2
101    btfss status,zero ; w = 2?
102    goto mainloop ; no
103    incf signalCounter
104    movf signalCounter,w
105    xorlw 5 ; counted 5 signals? 5 signals → 10 m
106    btfss status,zero
107    goto mainloop
108
109    clrf signalCounter ; reset signal counter
110    ; increment meter counter
111    movlw mCounter,w ; meter counter
112    movwf fsr
113 bcdinc
114     incf indf
115     movf indf,w
116     xorlw 10 ; bcd overflow?
117     btfss status,zero
118     goto outputSsd ; no
119     ; yes
120     clrf indf ; clear overflown bcd digit
121     ; increment next digit
122     movf fsr,w
123     xorlw kmCounter ; out of bounds?
124     btfsc status,zero
125     goto outputSsd ; yes
126     ; no
127     incf fsr
128     goto bcdinc
129

```



```

130 checkEdge
131     ; read current value
132     movf porta,w
133     andlw sigMask ; mask clock signal on RA0
134     movwf currentVal
135     xorwf oldVal,w ; compare with oldVal
136     movwf edge
137
138     ; oldVal := currentVal
139     movf currentVal,w
140     movwf oldVal
141
142     ; edge = 0 if currentVal = oldVal
143     movf edge ; set zero-flag if edge = 0
144     btfsc status,zero ; edge 0?
145     retlw 0 ; no → no new edge
146
147     movf currentVal ; set zero-flag if currentVal = 0
148     btfss status,zero ; currentVal = 0? or zero = 1?
149     retlw 2 ; no → rising edge
150     retlw 1 ; yes → falling edge
151
152 bcdToSsd ; Sevent Segment Display
153     movf bcdin,w
154
155     addwf pcl ; unsave: undefined behavior if bcdin > 9
156     retlw 3Fh ; 0
157     retlw 06h ; 1
158     retlw 5Bh ; 2
159     retlw 4Fh ; 3
160     retlw 66h ; 4
161     retlw 6Dh ; 5
162     retlw 7Dh ; 6
163     retlw 07h ; 7
164     retlw 7Fh ; 8
165     retlw 6Fh ; 9
166     ; retlw 0 ; A
167     ; retlw 0 ; B
168     ; ...

```

Impuls Verlängerer

```

1     device 16f84
2
3     ; \***** labels *****\
4     status equ 3
5     rp0 equ 5
6
7     porta equ 5
8     sigIn equ 0 ; RA0
9     sigOut equ 1 ; RA1
10
11    ; vars
12    msCounter equ 10h
13
14    ; \***** labels *****\
15
16    org 0
17
18    cold
19    bsf status,rp0 ; bank 1
20    bcf prota,sigOut
21    bcf status,rp0 ; bank 0
22
23    goto mainloop

```

```

24
25 makeOut
26     ; check if ms Counter between 2 & 6
27     infc msCounter ; 1 µs
28     decfsz msCounter
29     goto waitOutInit
30     goto mainloop ; decfsz = 0
31
32 waitOutInit
33     bcs prota,sigOut ; 1µs
34     movlw 100 ; 1 µs; number of deplay iterations
35     movwf deplayCounter ; 1 µs
36 waitOut ; iteration = 5 µs
37     decfsz deplayCounter ; 1 µs (2 µs)
38     goto waitOut ; 2 µs
39     nop ; 1 µs
40     nop ; 1 µs
41     goto makeOut ; 2 µs
42
43 mainloop
44     bcf prota,sigOut ; sigOut = 0
45     btfss porta,0 ; high sig? / 2 µs
46     goto mainloop ; no
47 count
48     movlw 31 ; 1 µs; number of deplay iterations
49     movwf deplayCounter ; 1 µs
50 wait ; iteration = 3 µs
51     decfsz deplayCounter ; 1 µs (2 µs)
52     goto wait ; 2 µs
53
54     ; 100 µs
55
56     btfss porta,0 ; still high?
57     goto makeOut ; no
58
59     infc msCounter
60     goto count

```

Lauflicht

```

1  status equ 3
2  rp0 equ 5
3  carry equ 0
4  zero equ 2
5  pcl equ 2
6
7  porta equ 5
8  portb equ 6
9  trisa equ 5
10 trisb equ 6
11
12 cold
13     bsf status,rp0 ;Auf Bank1 umschalten
14     bsf trisa,0 ;RA0 auf Eingang setzen
15     clrf trisb ;PortB alles auf Ausgang
16     bcf status,rp0 ;Auf Bank 0 umschalten
17     bsf portb,0 ;Port RB0 auf 1 Setzen
18 start
19     btfsc porta,0 ;Wenn signal an Port0 erkannt
20     goto loop ;in den Loop springen, ansonsten bei
21     goto start ;Start weitermachen
22
23 loop
24     btfsc status,carry ;Wenn Carry auf 1
25     rlf portb ;2 mal shiften

```

```
26  rlf portb ;andernfalls nur  
27  goto start ;1 mal shiften
```