

Read BCD typing:

```
; get BCD number from terminal  
; Entry: A=high nibble, B=low  
; Exit: A  
; e.g. A = 31H, B = 39H, or we  
; typed 19 with PC keyboard,  
; return value will be A = 19H  
;
```

Read BCD typing:

; get BCD number from terminal

```
getBCD:      CALL getch;  '1'=31H  
             ANL A,#0FH  
             PUSH ACC ; A = 01  
             CALL getch ; '9'=39H  
             ANL A,#0FH  
             MOV B,A ; A= 09  
             POP ACC ; A = 01
```

Read BCD typing:

```
SWAP A ; A = 10
```

```
ORL A,B ; A = 19H
```

```
RET
```

```
; test code with BCD typing
```

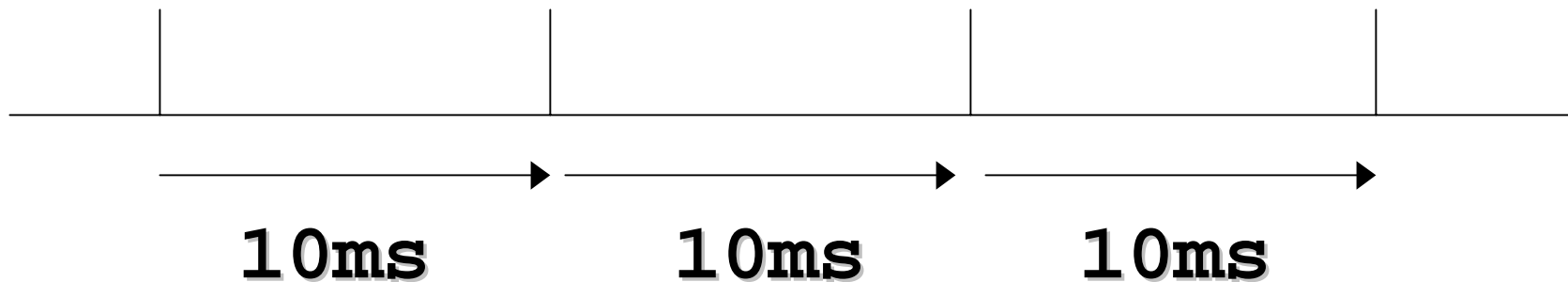
```
main: CALL getBCD
```

```
CALL printBCD; or phex
```

```
JMP main
```

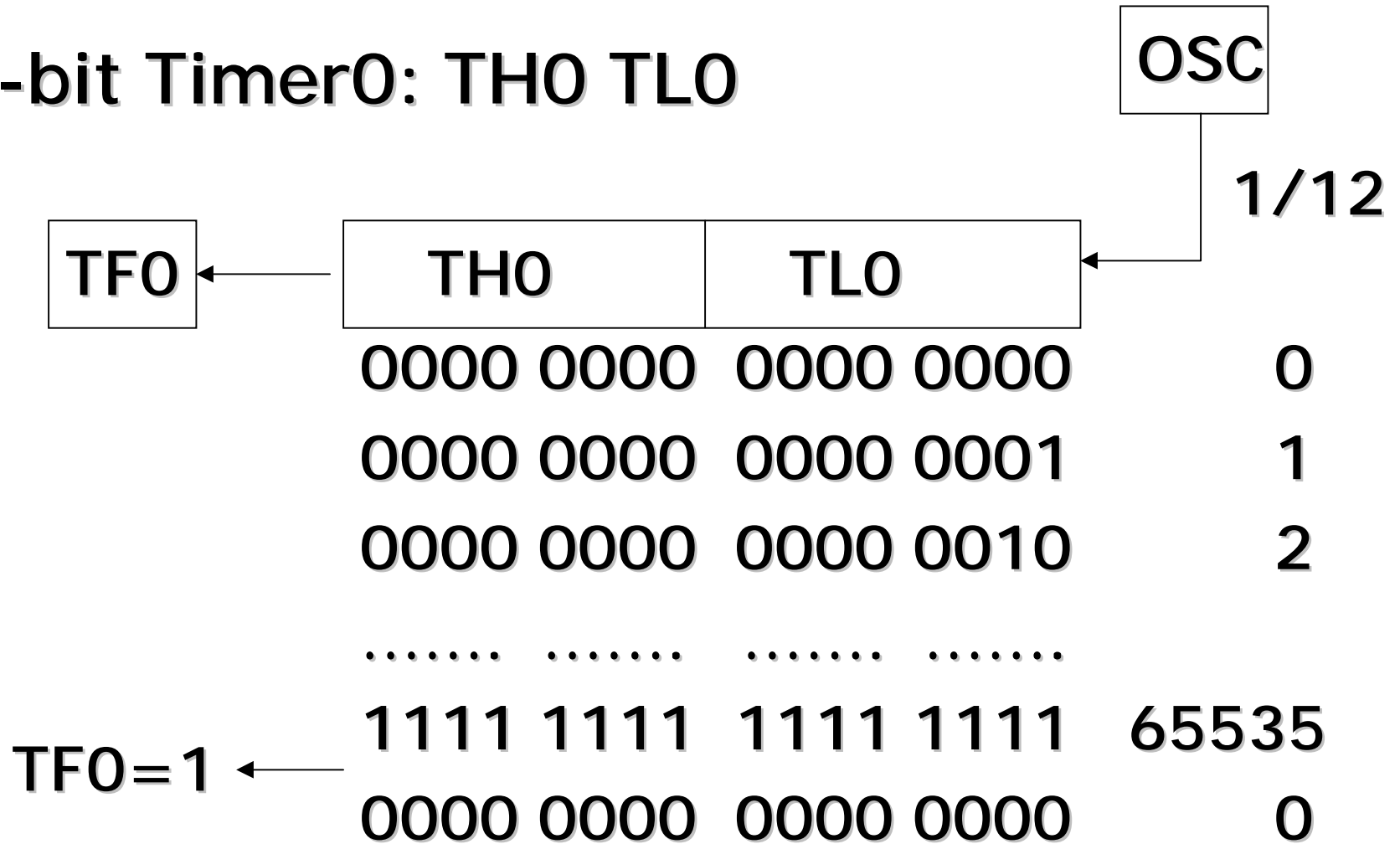
Timer interrupt:

- ;** to produce 10ms system tick
- ;** for real-time control
- ;**



Using Hardware Timer

16-bit Timer0: TH0 TLO



Using Hardware Timer

8051SBC has 11.0592MHz oscillator. The clock frequency for timer is $11.0592\text{MHz}/12 = 921,600 \text{ Hz}$

To produce 10ms or 100Hz timer overflow, the timer clock must be divided by 100,
 $921,600\text{Hz}/100\text{Hz}=9216$ Load value is $65536-9216=56320$ (DC00H)

Timer interrupt:

```
CSEG AT 8000H
```

```
JMP main
```

```
ORG 800BH; interrupt vector for  
timer 0
```

```
servicetimer0: INC tick
```

```
ORL TH0,#0DCH
```

```
RETI
```

```
main: ORL TMOD,#1;set modeltimer0
```

Timer interrupt:

```
MOV TH0,#0DCH
```

```
MOV TL0,#00H
```

```
SETB EA ; enable all
```

```
; interrupt
```

```
SETB ET0 ; enable timer0
```

```
; interrupt
```

```
SETB TR0 ; run timer 0
```

```
JMP $ ; jump here
```

Timer interrupt:

```
; let's test timer0 interrupt  
; using tick variable  
                DSEG AT 30H  
tick:          DS 1  
                CSEG  
  
; and our debug LED, making the  
; LED blinks every 0.5s
```

Timer interrupt:

```
LEDblink:  MOV A, tick
           CJNE A,#50,exit; 50x10ms
           MOV tick,#0
           CPL P1.7

exit:      RET

; to enable timer interrupt
; set bit EA, ET0 and run timer0
; timer mode set to 16-bit
```

Digital clock program:

DSEG AT 30H

```
tick:          DS  1  
sec100:       DS  1 ; 100 count=1s  
sec:         DS  1 ; 0-59  
min:         DS  1 ; 0-59  
hour:        DS  1 ; 0-23
```

Digital clock program:

```
                SETB EA ; enable all
; interrupt
                SETB ET0 ; enable timer0
; interrupt
                SETB TR0 ; run timer 0
wait:          MOV A,tick
                JNB ACC.0, wait
                MOV tick,#0 ; reset tick
```

Digital clock program:

```
CALL updateclock
```

```
JMP wait
```

```
updateclock:
```

```
INC sec100
```

```
MOV A,sec100
```

```
CJNE A,#100,checksec
```

```
MOV sec100,#0
```

```
INC sec
```

Digital clock program:

```
Checksec: MOV A,sec
          CJNE A,#60,checkmin
          MOV sec,#0
          INC min
checkmin: MOV A,min
          CJNE A,#60,checkhour
          MOV min,#0
          INC hour
```

Digital clock program:

```
Checkhour:  MOV A, hour
            CJNE A, #24, checkday
            MOV hour, #0
checkday:   RET
```

```
; test print the clock on screen
; e.g.  17:59:59
```

Digital clock program:

Practices-

- 1 modify sec, min and hour updated with BCD increment,
- 2 use printBCD to display clock on screen
- 3 add time setting from terminal with getBCD or getHEX
- 4 add DAY value for MON to SUN, say 1-7