Timer/Counter Programming

Instructor

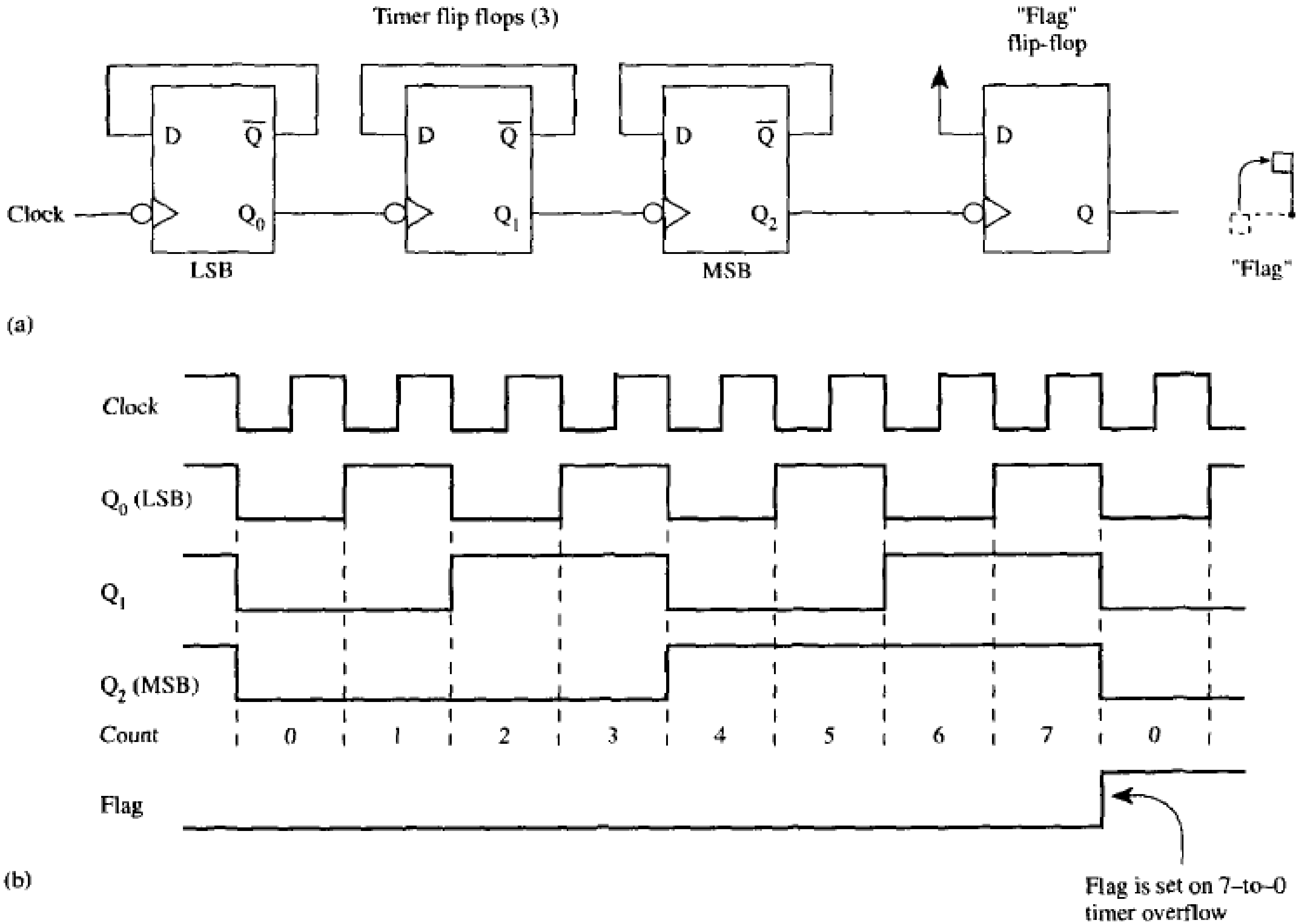
Zhizheng Wu

吴智政

School of Mechatronic Engineering and Automation

- List the timers of the 8051 and their associated registers
- Describe the various modes of the 8051 timers
- Program the 8051 timers in Assembly to generate a time delay

TIMER PROGRAMING



A three bit timers a) schematic b) timing diagram

TIMER REGISTERS

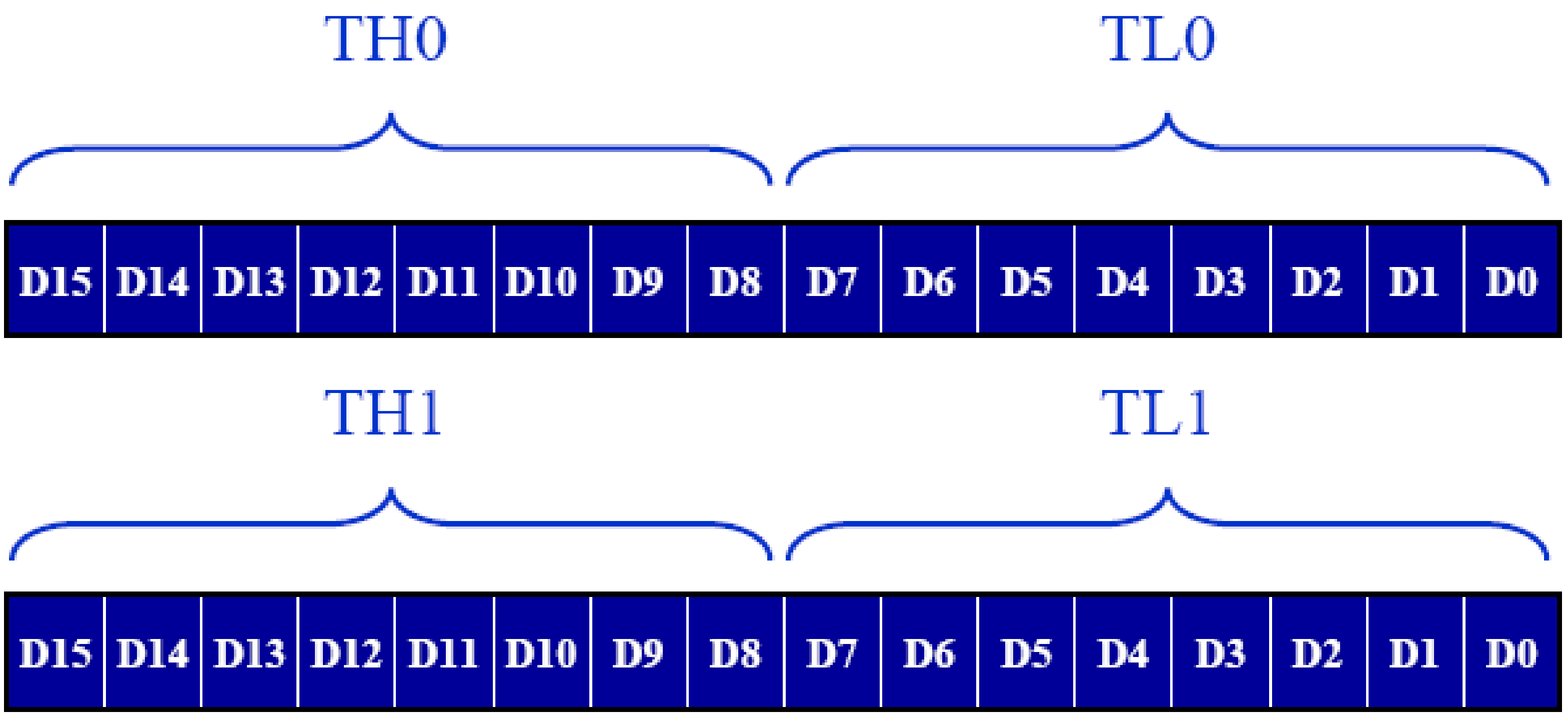
- The 8051 has two timers/counters, they can be used either as
  - Timers to generate a time delay or as
  - Event counters to count events happening outside the microcontroller
- Both Timer 0 and Timer 1 are 16 bits wide
  - Since 8051 has an 8-bit architecture, each 16-bits timer is accessed as two separate registers of low byte TLx and high byte THx
- Other two SFRs
  - TMOD (Timer Mode Register)
  - TCON (Timer Control Register)

TIMER REGISTERS

SFR	Address	Description
TL0	**8AH**	Timer/Counter 0 low byte
TH0	**8CH**	Timer/Counter 0 high byte
TL1	**8BH**	Timer/Counter 1 low byte
TH1	**8DH**	Timer/Counter 1 high byte
TMOD	**89H**	Timer/Counter mode control
TCON	**88H**	Timer/Counter control

Registers used in timer operation

TIMER REGISTERS



Registers of Timer 0 and Timer 1

- **Timer 0 registers**
  - **low byte register is called TL0 (Timer 0 low byte) and the high byte register is referred to as TH0 (Timer 0 high byte)**
  - **can be accessed like any other register, such as A, B, R0, R1, R2, etc.**
  - **"MOV TL0, #4 FH" moves the value 4FH into TL0**
  - **"MOV R5, TH0" saves TH0 (high byte of Timer 0) in R5**

- **Timer 1 registers**
  - **also 16 bits**
  - **split into two bytes TL1 (Timer 1 low byte) and TH1 (Timer 1 high byte)**
  - **accessible in the same way as the registers of Timer 0**

TIMER REGISTERS

- TMOD (timer mode) register (SFR address: 89H)
  - Both timers 0 and 1 use the same register, called TMOD (timer mode), to set the various timer operation modes
  - 8-bit register
  - lower 4 bits are for Timer 0
  - upper 4 bits are for Timer 1
  - lower 2 bits are used to set the timer mode
  - upper 2 bits to specify the operation

TIMER REGISTERS

(MSB) (LSB)

GATE	C/T	M1	M0	GATE	C/T	M1	M0
Timer 1				Timer 0			

GATE Gating control when set. The timer/counter is enabled only while the INTx pin is high and the TRx control pin is set. When cleared, the timer is enabled whenever the TRx control bit is set.

C/T Timer or counter selected cleared for timer operation (input from internal system clock). Set for counter operation (input from Tx input pin).

M1 Mode bit 1

M0 Mode bit 0

M1	M0	Mode	Operating Mode
0	0	0	13-bit timer mode 8-bit timer/counter THx with TLx as 5-bit prescaler
0	1	1	16-bit timer mode 16-bit timer/counters THx and TLx are cascaded; there is no prescaler
1	0	2	8-bit auto reload 8-bit auto reload timer/counter; THx holds a value that is to be reloaded into TLx each time it overflows.
1	1	3	Split timer mode

TMOD register

TIMER REGISTERS

- M1, M0
  - 00 : Mode 0 ;13-bit timer mode
  - 01 : Mode 1 ;16-bit timer mode
  - 10 : Mode 2 ;8-bit timer mode
  - 11 : Mode 3 ;Split timer mode
- C/T bit
  - 0 : Timer mode. The clock source is the crystal oscillations. In the 8051, the frequency of the timer clock 1/12 the frequency of the oscillator.
  - 1 : Counter mode.

- **C/T bit in TMOD register**
  - used as a timer, the 8051's crystal is used as the source of the frequency
  - used as a counter, pulse outside the 8051 increments the TH, TL registers
  - counter mode, TMOD and TH, TL registers are the same as for the timer

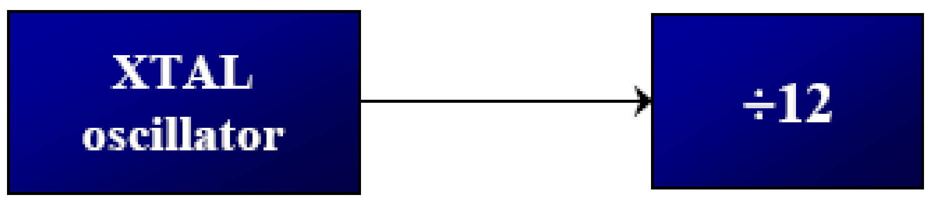
- C/T bit in TMOD register
  - C/T bit in the TMOD register decides the source of the clock for the timer
  - C/T = 0, timer gets pulses from crystal
  - C/T = 1, the timer used as counter and gets pulses from outside the 8051
    - ☞C/T = 1, the counter counts up as pulses are fed from pins P3.4 and P3.5. Pins P3.4 and P3.5 are called T0 (Timer 0 input) and T1 (Timer 1 input), respectively.
    - Timer 0, when C/T = 1, pin P3.4 provides the clock pulse and the counter counts up for each clock pulse coming from that pin
    - Timer 1, when C/T = 1 each clock pulse coming in from pin P3.5 makes the counter count up

TIMER REGISTERS

☞ Clock source for timer

- ✓ timer needs a clock pulse to tick
- ✓ if C/T = 0, the crystal frequency attached to the 8051 is the source of the clock for the timer
- ✓ frequency for the timer is always 1/12th the frequency of the crystal attached to the 8051

The crystal frequency
11.0592 MHz



$1/12 \times 11.0529$ MHz = 921.6 MHz;
T = 1/921.6 kHz = 1.085 us

- Gate
  - 0 : Start and stop of the timer are controlled through software:

```
SETB TR0        ; Start  Timer 0
SETB TR1        ; Start  Timer 1
CLR   TR0       ; Stop  Timer 0
CLR   TR1       ; Stop  Timer 1
```

  - 1 : Start and stop of the timer/counter are controlled through hardware:
    - ✓ Start and stop of the counter 0 is controlled through pin \INT0 (Pin 12 (P3.2)).
    - ✓ Start and stop of the counter 1 is controlled through pin \INT1 (Pin 13 (P3.3)).

- **The case of GATE = 0, 1 in TMOD**
  - GATE = 0, the timer is started with instructions "SETB TR0" and "SETB TR1”
  - GATE = 1, the start and stop of the timers are done externally through pins P3.2 and P3.3
  - allows us to start or stop the timer externally at any time via a simple switch

TIMER REGISTERS

- Timer Control (TCON) special function register (SFR address: 88H)
  - ➢ Contains the TR0, TR1, TF0, and TF1 bits. The TCON is bit addressable.

For Timer 0

SETB TR0	=	SETB TCON.4
CLR TR0	=	CLR TCON.4
SETB TF0	=	SETB TCON.5
CLR TF0	=	CLR TCON.5

For Timer 1

SETB TR1	=	SETB TCON.6
CLR TR1	=	CLR TCON.6
SETB TF1	=	SETB TCON.7
CLR TF1	=	CLR TCON.7

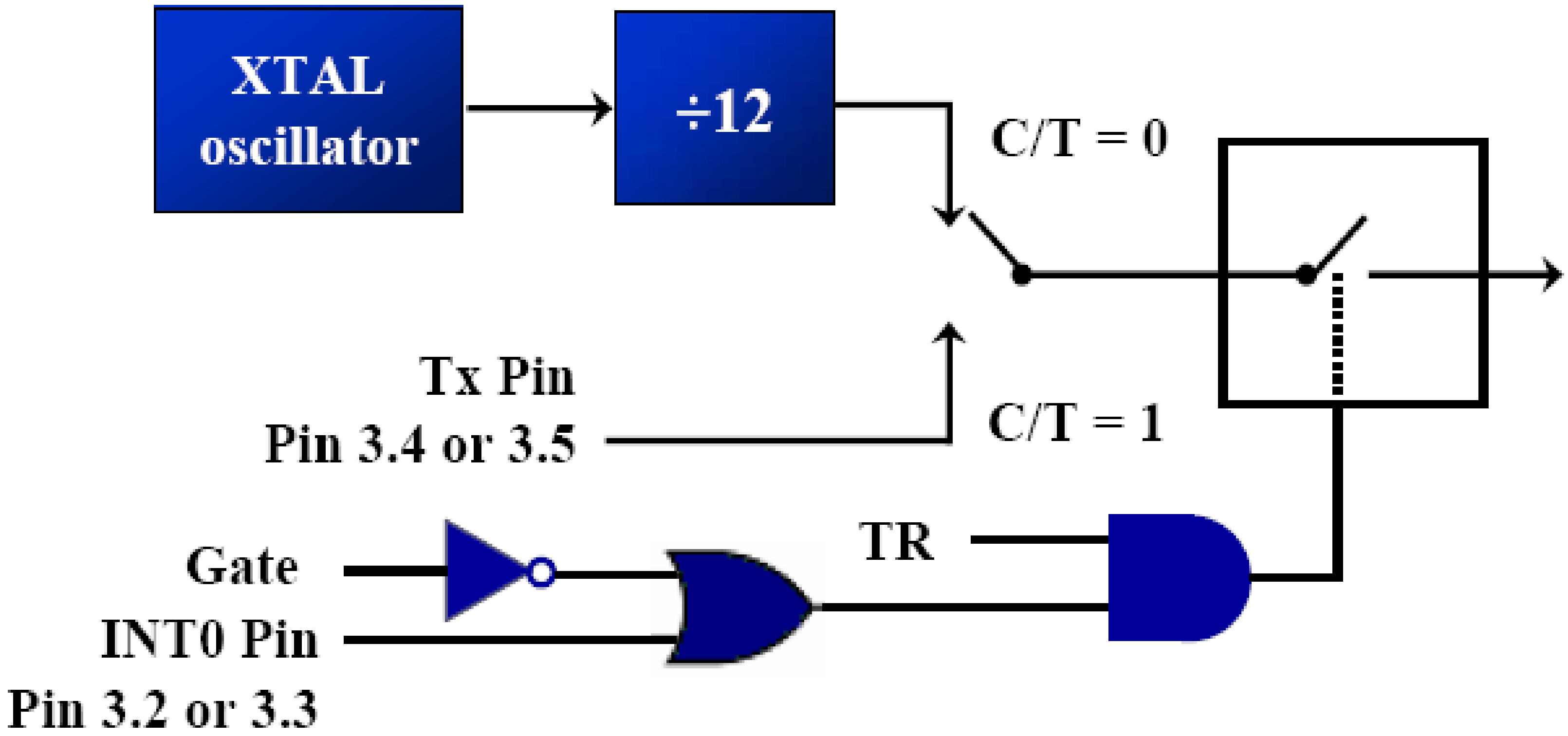
TCON: Timer/Counter Control Register

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

For timer

For interrupt

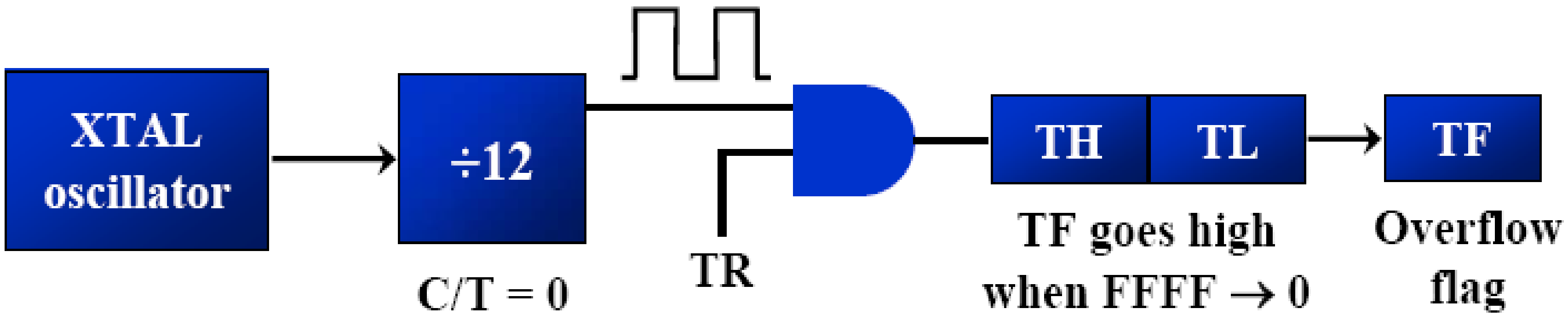
TIMER REGISTERS



Operating diagram of the timers

- **Mode 1 programming**
  - 16-bit timer, values of 0000 to FFFFH
  - TH and TL are loaded with a 16-bit initial value
  - timer started by "SETB TR0" for Timer 0 and "SETB TR1" for Timer l
  - timer count ups until it reaches its limit of FFFFH, then rolls over from FFFFH to 0000H
  - sets TF (timer flag)
  - when this timer flag is raised, can stop the timer with "CLR TR0" or "CLR TR1“
  - after the timer reaches its limit and rolls over, the registers TH and TL must be reloaded with the original value and TF must be reset to 0

MODES OF THE TIMERS



Operating diagram of mode 1

- **Steps to program in mode 1**

1. load TMOD, select mode 1 for timer 0 or timer 1
2. load the TL0 and TH0
3. start timer
4. monitor the timer flag (TF) with "JNB"
5. get out of the loop when TF=1
6. clear TF
7. reload the TL0 and TH0
8. go back to Step 3

MODES OF THE TIMERS

Example 1: In the following program, we are creating a square wave of 50% duty cycle (with equal portions high and low) on the P1.5 bit. Timer 0 is used to generate the time delay

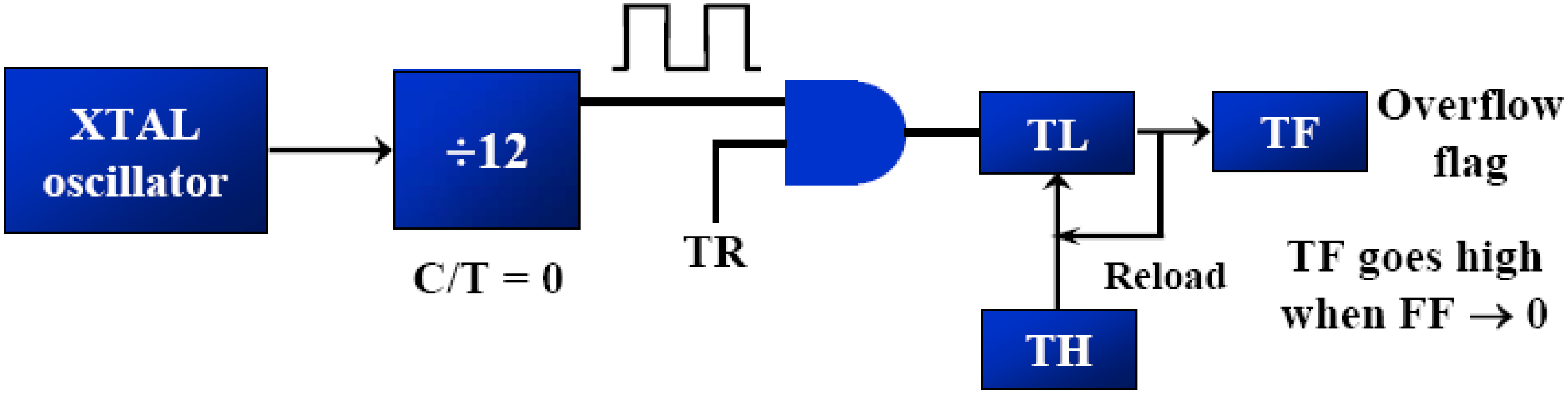
```
MOV TMOD,#01            ;Timer 0, mode 1(16-bit mode)
HERE: MOV TL0,#0F2H     ;TL0 = F2H, the Low byte
MOV TH0,#0FFH           ;TH0 = FFH, the High byte
CPL P1.5                ;toggle P1.5
ACALL DELAY
SJMP HERE               ;load TH, TL again

DELAY:                  ;delay using Timer 0
SETB TR0                ;start Timer 0
AGAIN: JNB TF0,AGAIN    ;monitor Timer 0 flag until ;it rolls over
CLR TR0                 ;stop Timer 0
CLR TF0                 ;clear Timer 0 flag
RET

END
```

- **Mode 2 programming**
  - **8-bit timer, allows values of 00 to FFH**
  - **TH is loaded with the 8-bit value**
  - **a copy is given to TL**
  - **timer is started by ,"SETB TR0" or "SETB TR1”**
  - **starts to count up by incrementing the TL register**
  - **counts up until it reaches its limit of FFH**
  - **when it rolls over from FFH to 00, it sets high TF**
  - **TL is reloaded automatically with the value in TH**
  - **To repeat, clear TF**
  - **mode 2 is an auto-reload mode**

MODES OF THE TIMERS



Operating diagram of mode 2

- **Steps to program in mode 2**

1. **load TMOD, select mode 2**
2. **load the TH**
3. **start timer**
4. **monitor the timer flag (TF) with "JNB"**
5. **get out of the loop when TF=1**
6. **clear TF**
7. **go back to Step 4 since mode 2 is auto-reload**

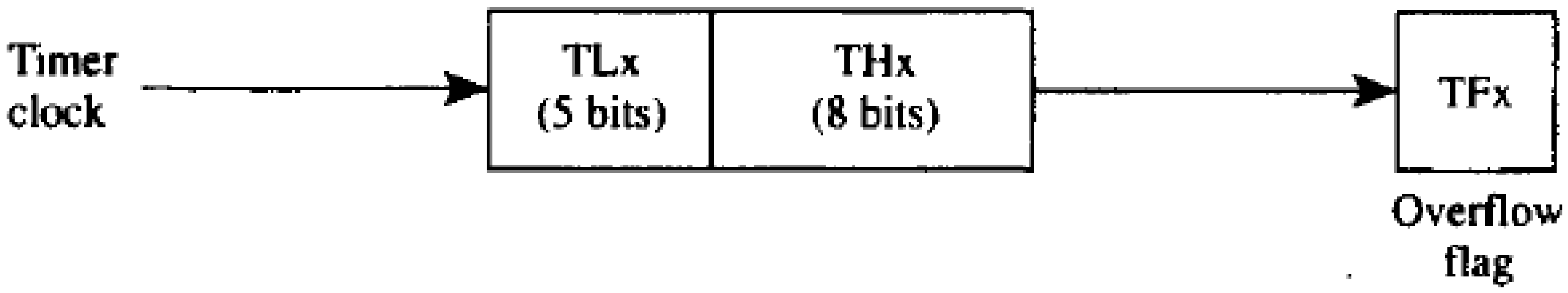
MODES OF THE TIMERS

Example 2: In the following program, we are creating a square wave of 50% duty cycle on the P1.5 bit. Timer 1 in mode 2 is used to generate the time delay.

```
        MOV  TMOD, #20H      ;T1/8-bit/auto reload
        MOV  TH1, #5         ;TH1 = 5
        SETB  TR1            ;start the timer 1
BACK:   JNB  TF1, BACK       ;till timer rolls over
        CPL  P1.5            ;toggle P1.5
        CLR  TF1             ;clear Timer 1 flag
        SJMP  BACK           ;mode 2 is auto-reload
```

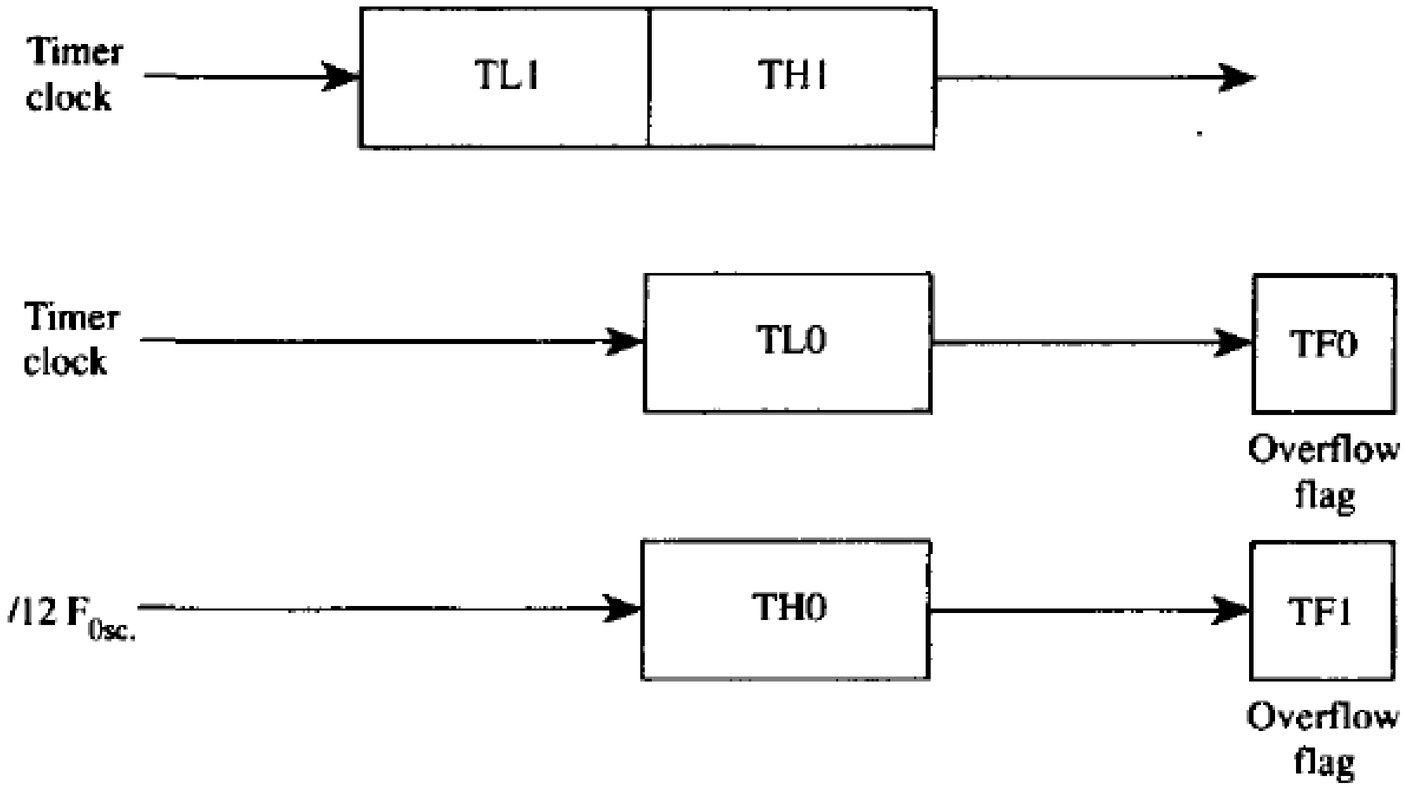
- **Mode 0**
  - **works like mode 1**
  - **13-bit timer instead of 16bit**
  - **13-bit counter hold values 0000 to 1FFFH**
  - **when the timer reaches its maximum of 1FFFH, it rolls over to 0000, and TF is set**

MODES OF THE TIMERS



Timer mode 0

- **Mode 3 (split timer mode)**
  - **Timer 0 is split into two 8-bit timers**
  - **TL0 and TH0 are separated timers with overflows setting TF0 and TF1 bits respectively**
  - **Timer 1 can be used in other modes, however it will not affect the overflow bit and produce the interrupt. E.g. baud rate generator**

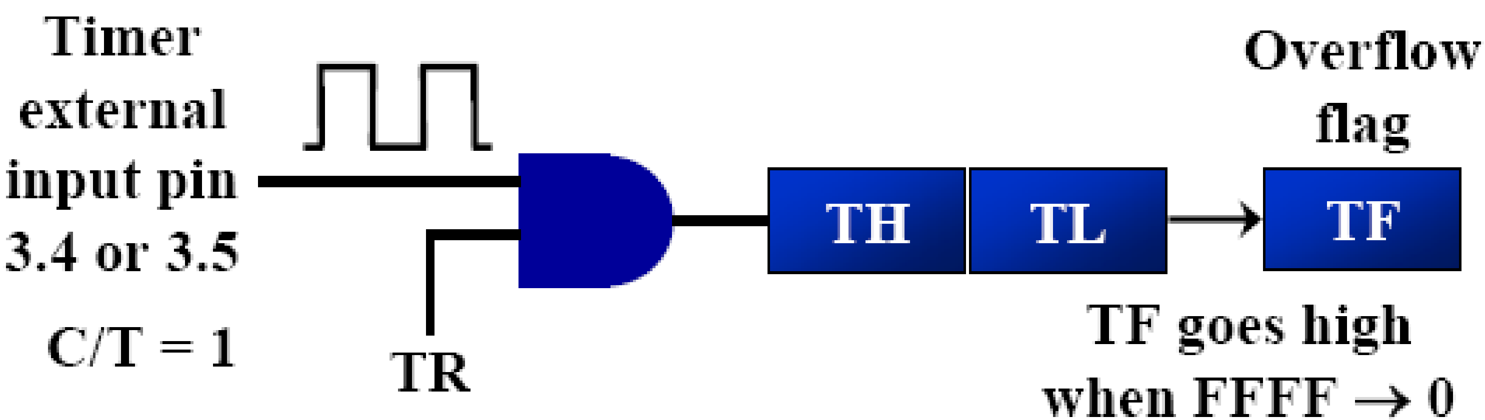

Timer clock
TL1
TH1
Timer clock
TL0
TF0
Overflow flag
/12 $F_{0sc.}$
TH0
TF1
Overflow flag

Timer mode 3

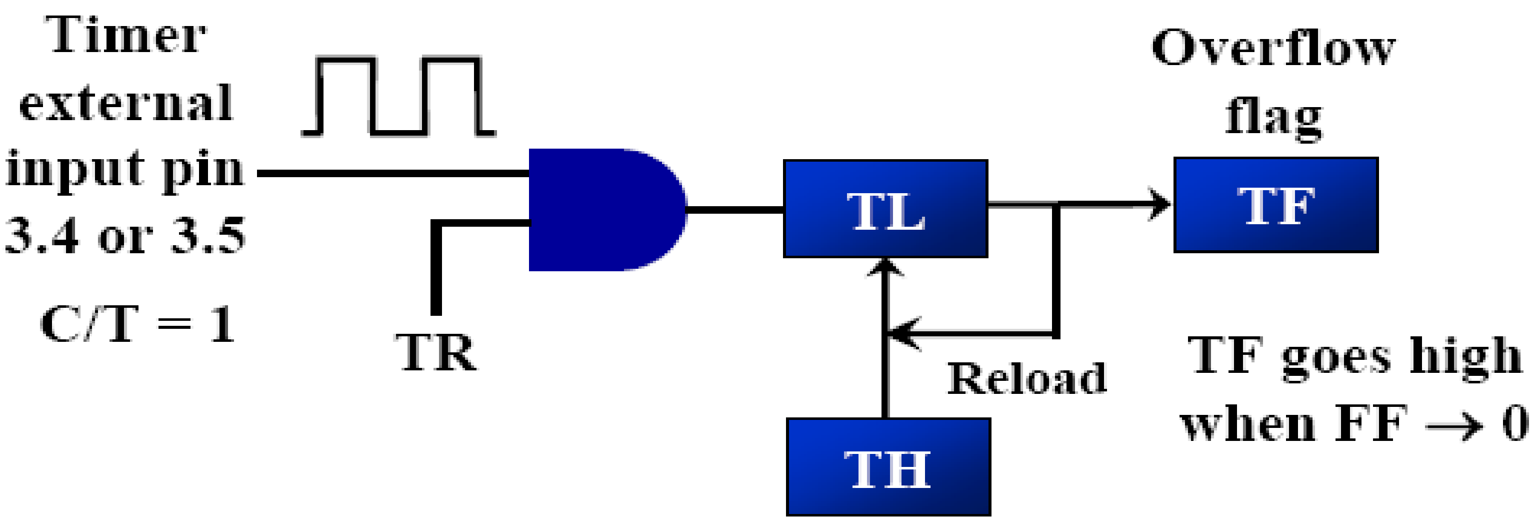
- **Counter operation**
  - Timers can also be used as counters counting events happening outside the 8051 by setting C/T = 1
  - When it is used as a counter, it is a pulse outside of the 8051 that increments the TH, TL registers
  - TMOD and TH, TL registers are the same as for the timer discussed previously
  - Programming the timer in the different modes also applies to programming it as a counter except the source of the frequency

MODES OF THE TIMERS

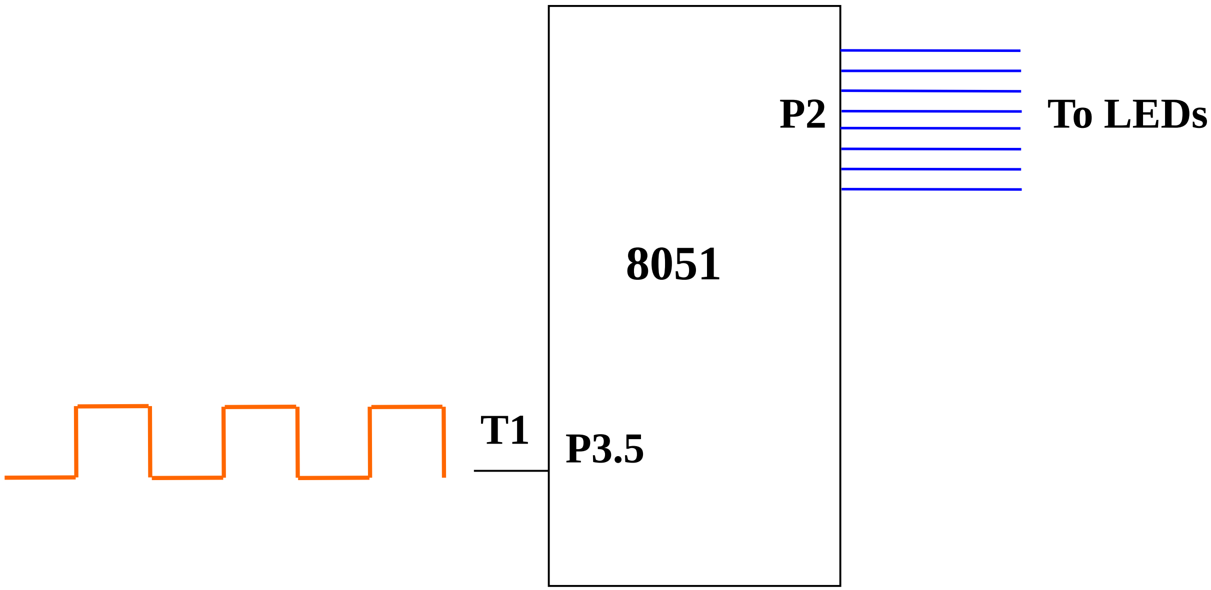
Timer with external input (Mode 1)



Timer with external input (Mode 2)



- **Example 3:** Assuming that clock pulses are fed into pin T1, write a program for counter 1 in mode 2 to count the pulses and display the state of the TL1 count on P2, which connects to 8 LEDs.



Solution:

```
        MOV TMOD, #01100000B    ;counter 1, mode 2, C/T=1
        MOV TH1, #0             ;clear TH1
        SETB  P3.5              ;make T1 (pin P3.5) input
AGAIN:  SETB TR1                ;start the counter
BACK:   MOV A, TL1              ;get copy of TL
        MOV P2, A               ;display it on port 2
        JNB TF1, Back           ;keep doing, if TF = 0
        CLR TR1                 ;stop the counter 1
        CLR TF1                 ;make TF=0
        SJMP AGAIN              ;keep doing it
```

- To generate a time delay using timer mode 0,1 or 3
  - ①Load the TMOD value register indicating which timer (timer 0 or timer 1) is to be used and which timer mode (0, 1 or 3) is selected
  - ②Load registers TL and TH with initial count value
  - ③Start the timer
  - ④Keep monitoring the timer flag (TF) with the **JNB TFx, target** instruction to see if it is raised
  - ⑤Get out of the loop when TF becomes high
  - ⑥Stop the timer
  - ⑦Clear the TF flag for the next round
  - ⑧Go back to Step 2 to load TH and TL again

- **Finding values to be loaded into the timer**
  - **XTAL = 11.0592 MHz (12MHz)**
  - **divide the desired time delay by 1.085ms (1ms) to get *n***
  - **65536 – *n* = *N***
  - **convert *N* to hex yyxx**
  - **set TL = xx and TH = yy**

Example 4: Assuming XTAL = 11.0592 MHz, write a program to generate a square wave of 50 Hz frequency on pin P2.3.

- T = 1/50 Hz = 20 ms
- 1/2 of it for the high and low portions of the pulse = 10 ms
- 10 ms / 1.085 us = 9216
- 65536 - 9216 = 56320 in decimal = DC00H
- TL = 00 and TH = DCH
- The calculation for 12MHz crystal uses the same steps

Example 5: Assuming XTAL = 11.0592 MHz, write a program to generate a square wave of 50 Hz frequency on pin P2.3.

```
MOV TMOD,#10H                ;Timer 1 mode 1 (16-bit)
AGAIN: MOV TL1,#00           ;TL1 = 00, Low byte
MOV TH1,#0DCH                ;TH1 = 0DCH, High byte

SETB TR1                     ;start Timer 1
BACK: JNB TF1,BACK           ;stay until timer rolls over
CLR TR1                      ;stop Timer 1
CPL P2.3                     ;compliment P2.3 to get hi, lo
CLR TF1                      ;clear Timer 1 flag
SJMP AGAIN                   ;reload timer since
                             ;mode 1 is not auto reload


END
```

GENERATING A TIME DELAY

Example 6: The following program generates a square wave on pin P 1.5 continuously using Timer 1 for a time delay. Find the frequency of the square wave if XTAL = 11.0592 MHz. In your calculation do not include the overhead due to the timer setup instructions in the loop.

```
        MOV TMOD, #10         ;Timer 1, mod 1 (16-bit mode)
AGAIN:  MOV TL1, #34H         ;TL1=34H, low byte of timer
        MOV TH1, #76H         ;TH1=76H, high byte timer
        SETB TR1              ;start the timer 1
BACK:   JNB TF1, BACK         ;till timer rolls over
        CLR TR1               ;stop the timer 1
        CPL P1.5              ;toggle P1.5
        CLR TF1               ;clear timer flag 1
        SJMP AGAIN            ;is not auto-reload
```

Solution:

Since FFFFH – 7634H = 89CBH + 1 = 89CCH and 89CCH = 35276 clock count and 35276 $\times$ 1.085 us = 38.274 ms for half of the square wave. The frequency = 13.064Hz.

- **Generating a large time delay**
  - **size of the time delay depends**
    - ✓**crystal frequency**
    - ✓**timer's 16-bit register in mode 1**
  - **largest time delay is achieved by making both TH and TL zero**
  - **what if that is not enough?**

Example 7: Modify TL and TH in Example 6 to get the largest time delay possible. Find the delay in ms. In your calculation, exclude the overhead due to the instructions in the loop.

```
        MOV TMOD, #10       ;Timer 1, mod 1 (16-bit mode)
AGAIN:  MOV TL1, #00H       ;TL1=34H, low byte of timer
        MOV TH1, #00H       ;TH1=76H, high byte timer
        SETB TR1            ;start the timer 1
BACK:   JNB TF1, BACK       ;till timer rolls over
        CLR TR1             ;stop the timer 1
        CPL P1.5            ;toggle P1.5
        CLR TF1             ;clear timer flag 1
        SJMP AGAIN          ;is not auto-reload
```

Solution:

Making TH and TL both zero means that the timer will count from 0000 to FFFF, and then roll over to raise the TF flag. As a result, it goes through a total Of 65536 states. Therefore, we have delay = $(65536 - 0) \times 1.085$ us = 71.1065ms.

Example 8: Examine the following program and find the time delay in seconds. Exclude the time delay due to the instructions in the loop.

```
        MOV TMOD, #10H          ;Timer 1, mod 1
        MOV R3, #200            ;for multiple delay
AGAIN:  MOV TL1, #08H           ;TL1=08,low byte of timer
        MOV TH1, #01H           ;TH1=01,high byte
        SETB TR1                ;Start timer 1
BACK:   JNB TF1, BACK           ;until timer rolls over
        CLR TR1                 ;Stop the timer 1
        CLR TF1                 ;clear Timer 1 flag
        DJNZ R3, AGAIN          ;if R3 not zero then reload timer
```

Solution:

TH-TL = 0108H = 264 in decimal and 65536 – 264 = 65272. Now 65272 $\times$ 1.085 μs = 70.820 ms, and for 200 of them we have 200 $\times$ 70.820 ms = 14.164024 seconds.

- To generate a time delay using timer mode 2
  1. Load the TMOD value register indicating which timer (timer 0 or timer 1) is to be used, and the timer mode (mode 2) is selected
  2. Load the TH registers with the initial count value
  3. Start timer
  4. Keep monitoring the timer flag (TF) with the JNB TFx, target instruction to see whether it is raised
  5. Get out of the loop when TF goes high
  6. Clear the TF flag
  7. Go back to Step4, since mode 2 is auto reload

Example 9: Assume XTAL = 11.0592 MHz, find the frequency of the square wave generated on pin P1.0 in the following program

```
        MOV TMOD, #20H       ;T1/8-bit/auto reload
        MOV TH1, #5          ;TH1 = 5
        SETB TR1             ;start the timer 1
BACK:   JNB TF1, BACK        ;till timer rolls over
        CPL P1.0             ;toggle P1.0
        CLR TF1              ;clear Timer 1 flag
        SJMP BACK            ;mode 2 is auto-reload
```

Solution:

In mode 2 we do not need to reload TH since it is auto-reload. Now (256 - 05) $\times$ 1.085 us = 251 $\times$ 1.085 us = 272.33 us is the high portion of the pulse. As a result T = 2 $\times$ 272.33 us = 544.67 us and the frequency = 1.83597 kHz

Example 10: Find the frequency of a square wave generated on pin P1.0.

```
        MOV TMOD, #2H           ;Timer 0, mod 2 (8-bit, auto reload)
        MOV TH0, #0
AGAIN:  MOV R5, #250            ;multiple delay count
        ACALL DELAY
        CPL P1.0
        SJMP AGAIN
DELAY:  SETB TR0                ;start the timer 0
BACK:   JNB TF0, BACK           ;stay timer rolls over
        CLR TR0                 ;stop timer
        CLR TF0                 ;clear TF for next round
        DJNZ R5, DELAY
        RET
```

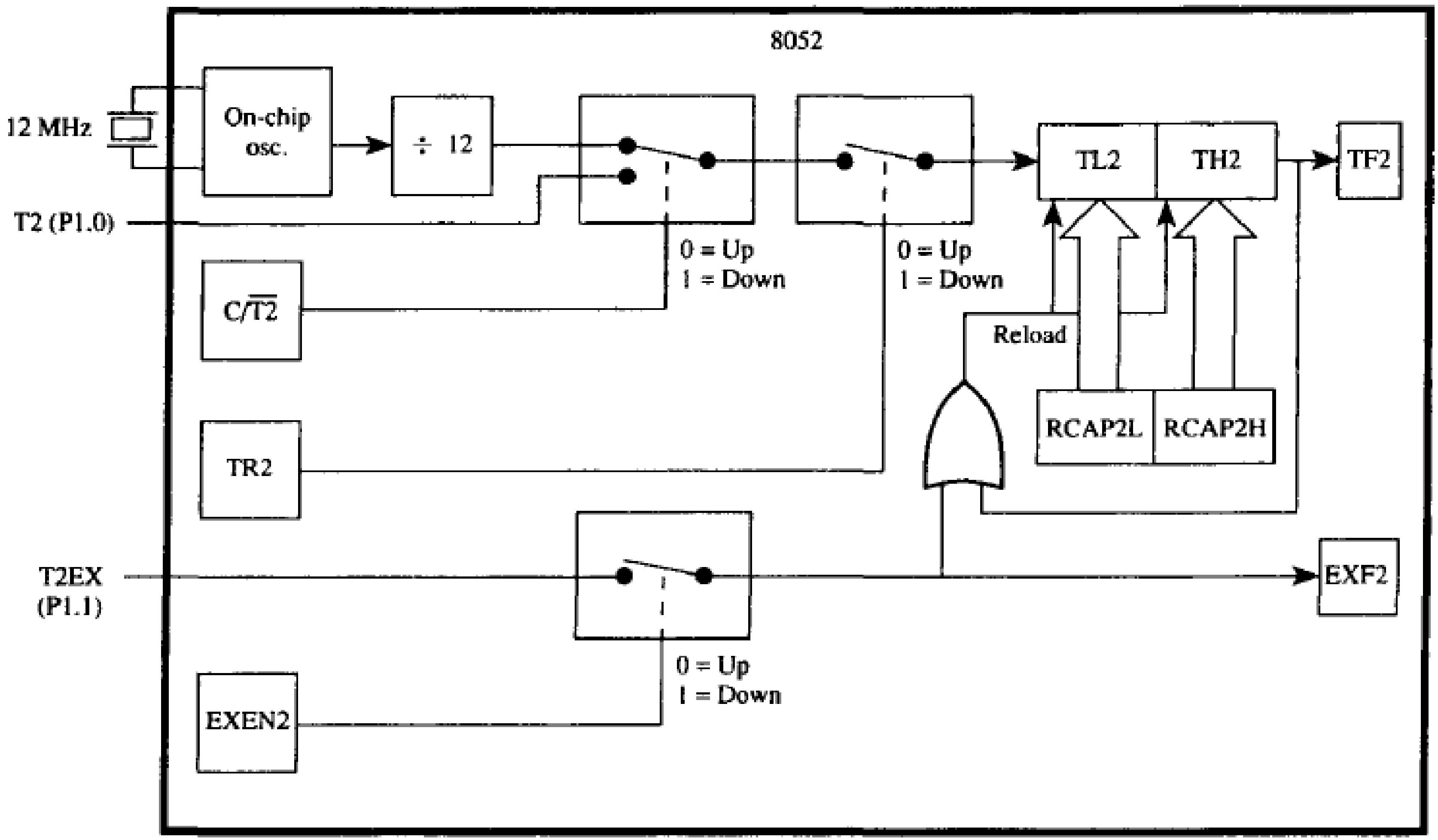

Solution: T = 2 (250 × 256 × 1.085 us) = 138.88ms, and frequency = 72 Hz

- **Using Windows calculator to find TH, TL**
  - Windows scientific calculator can be use to find the TH, TL values
  - Lets say we would like to find the TH, TL values for a time delay that uses 35,000 clocks of 1.085μs
    - open scientific calculator and select decimal
    - enter 35,000
    - select hex - converts 35,000 to hex 88B8H
    - select +/- to give -35000 decimal (7748H)
    - the lowest two digits (48) of this hex value are for TL and the next two (77) are for TH

- **Assemblers and negative values**
  - **can let the assembler calculate the value for TH and TL which makes the job easier**
  - **"MOV TH1, # -100", the assembler will calculate the -100 = 9CH**

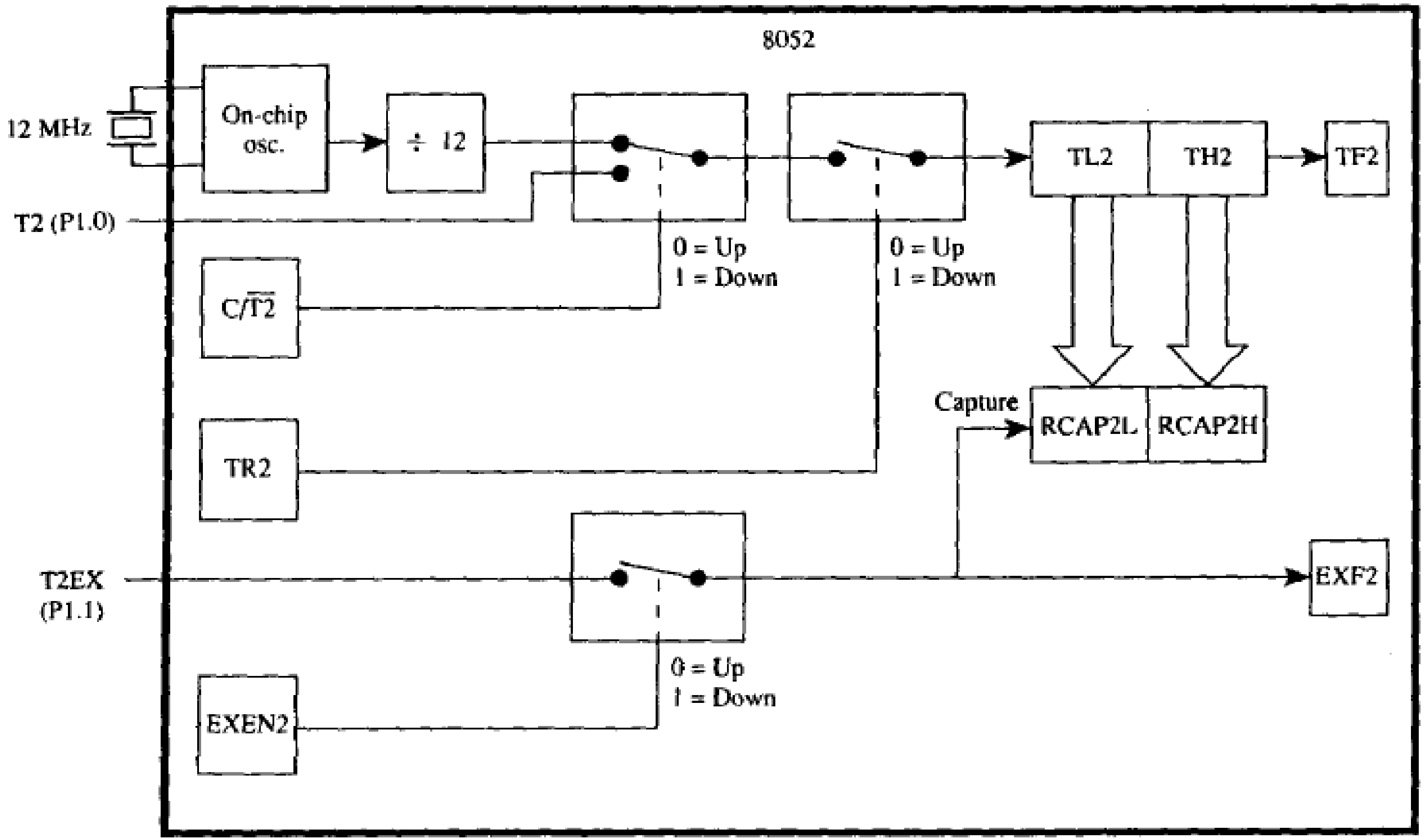
- **8052 timer 2 (for information only)**
  - **8052 has the third timer**
  - **Including five registers as TH2, TH1, T2CON, RCAP2H and RCAP2L**
  - **Three modes of operations:**
    - **Auto-reload (16 bits)**
    - **Capture**
    - **Baud rate generator (will be discussed in next lecture)**

GENERATING A TIME DELAY



Timer 2 in 16-bit auto-reload mode

GENERATING A TIME DELAY



Timer 2 in 16-bit capture mode