

# Timers (Part-2)

Lecture 5

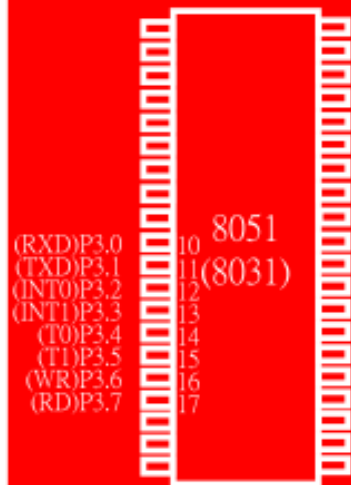
Bilal Habib  
DCSE, UET

# Outline for today

- Review of how delays are created using Timers
- How to use 'Auto Reload' mode of Timers
- How to use timers in counter mode

## PIN DESCRIPTION

### Port 3



- Port 3 can be used as input or output
  - Port 3 does not need any pull-up resistors
- Port 3 has the additional function of providing some extremely important signals

P3 Bit	Function	Pin
P3.0	RxD	10
P3.1	TxD	11
P3.2	<u>INT0</u>	12
P3.3	<u>INT1</u>	13
P3.4	T0	14
P3.5	T1	15
P3.6	<u>WR</u>	16
P3.7	<u>RD</u>	17

Serial communications

External interrupts

Timers

Read/Write signals of external memories

## PROGRAMMING TIMERS

### TMOD Register

- ❑ Both timers 0 and 1 use the same register, called TMOD (timer mode), to set the various timer operation modes
- ❑ TMOD is a 8-bit register
  - The lower 4 bits are for Timer 0
  - The upper 4 bits are for Timer 1
  - In each case,
    - The lower 2 bits are used to set the timer mode
    - The upper 2 bits to specify the operation

(MSB)

(LSB)

GATE	C/T	M1	M0	GATE	C/T	M1	M0
Timer1				Timer0			

## PROGRAMMING TIMERS

### TMOD Register (cont')

**Gating control when set.**  
Timer/counter is enable 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



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

#### Timer or counter selected

Cleared for timer operation (input from internal system clock)

Set for counter operation (input from Tx input pin)

## COUNTER PROGRAMMING

- ❑ Timers can also be used as counters counting events happening outside the 8051
  - 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 last section also applies to programming it as a counter
  - Except the source of the frequency

- Example:
- Assume that  $f_{clk} = 12 \text{ MHz}$ . What value do we need to load the timer's register if we want to have a time delay of 5 ms (milliseconds)? Show the program for timer 0 to create a pulse width of 5 ms.
- $F_{clk} = 12 \text{ MHz}$ ,
  - $F_{timer} = (1/12) * f_{clk} = 1 \text{ MHz}$
  - $T_p$  (Timer period of a timer) = 1  $\mu\text{sec}$
  - $5 \text{ msec} = 5000 \text{ usec} \Rightarrow 5000 \text{ cycles of timer clock}$

- Example 9-10:
- Assume that  $f_{clk} = 12 \text{ MHz}$ . What value do we need to load the timer's register if we want to have a time delay of 5 ms (milliseconds)? Show the program for timer 0 to create a pulse width of 5 ms.

• Sol:

Timer frequency  $F_{\text{timer}} = (1/12) * f_{clk} = 1 \text{ MHz}$

Cycle time of a Timer =  $1/F_{\text{timer}} = 1 \text{ usec}$

5msec = 5000 usec = 5000 cycles

Max = 0xFFFF (Mode1 or 16bit mode)

Difference =  $0xFFFF_{\text{hex}} - 5000_{\text{decimal}} = EC77_{\text{hex}}$

**TH0 = 0xEC, TL0 = 0x77**



## Auto-Reload Mode (Mode-2)

## PROGRAMMING TIMERS

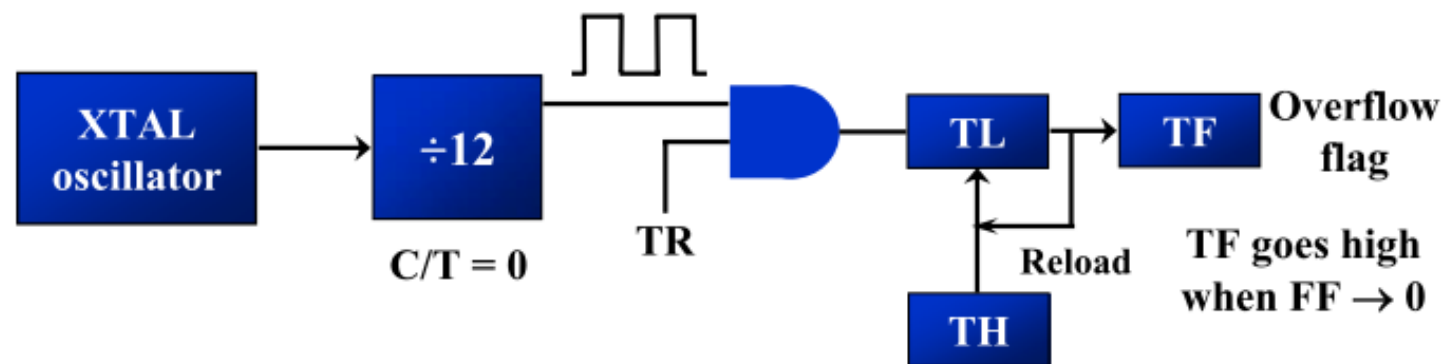
### Mode 2 Programming

- ❑ The following are the characteristics and operations of mode 2:
  1. It is an 8-bit timer; therefore, it allows only values of 00 to FFH to be loaded into the timer's register TH
  2. After TH is loaded with the 8-bit value, the 8051 gives a copy of it to TL
    - Then the timer must be started
    - This is done by the instruction `SETB TR0` for timer 0 and `SETB TR1` for timer 1
  3. After the timer is started, it starts to count up by incrementing the TL register
    - It counts up until it reaches its limit of FFH
    - When it rolls over from FFH to 00, it sets high the TF (timer flag)

## PROGRAMMING TIMERS

### Mode 2 Programming (cont')

4. When the TL register rolls from FFH to 0 and TF is set to 1, TL is reloaded automatically with the original value kept by the TH register
  - To repeat the process, we must simply clear TF and let it go without any need by the programmer to reload the original value
  - This makes mode 2 an auto-reload, in contrast with mode 1 in which the programmer has to reload TH and TL



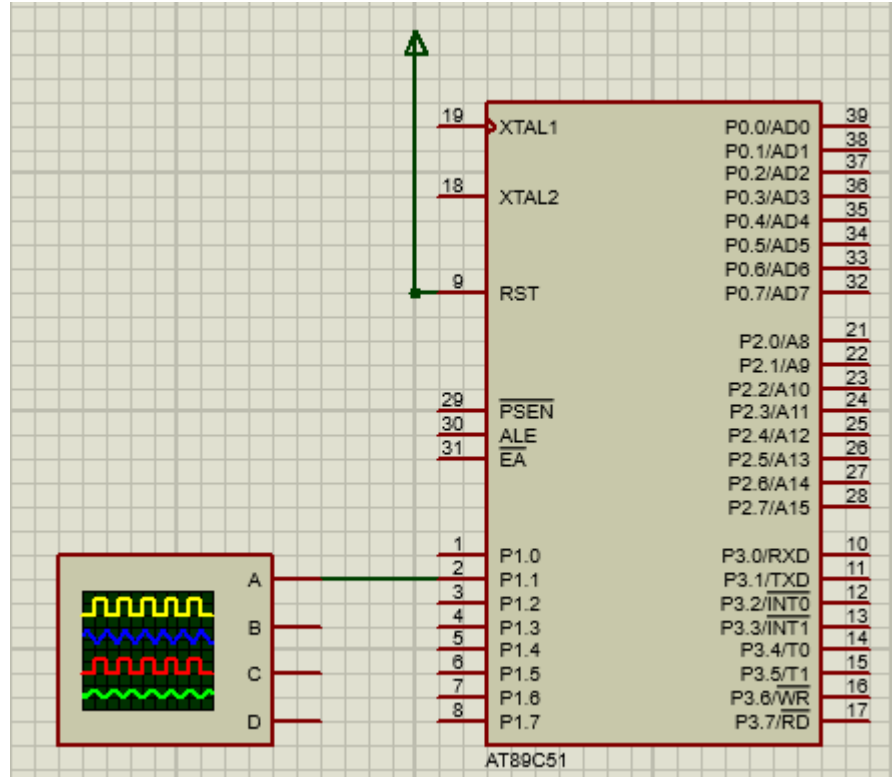
## PROGRAMMING TIMERS

### Mode 2 Programming

#### Steps to Mode 2 Program

- ❑ To generate a time delay
  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
    - Get out of the loop when TF goes high
  5. Clear the TF flag
  6. Go back to Step4, since mode 2 is auto-reload

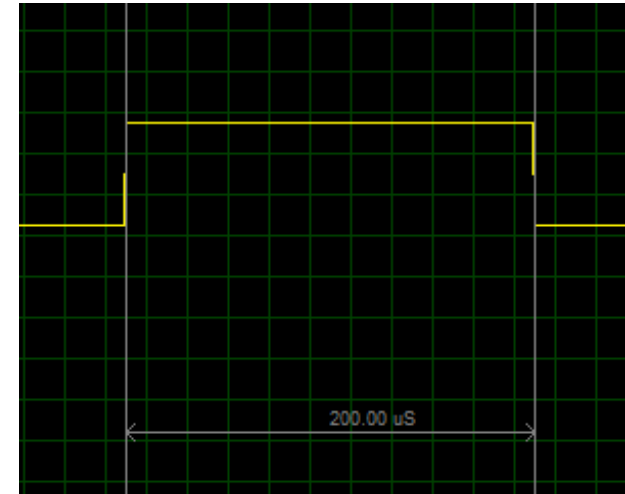
# Mode-2 of Timer Auto-Reload Mode



```
#include <reg51.h>
#include <stdio.h>
sbit pin = P1^1;
void start_timer(void)
{
    TR0 = 1;
}
void timer0_ISR(void) interrupt 1
{
    pin^=1;
}

void init_timer(void)
{
    TMOD = 0x02; // 8-bit Auto Reload mode, Timer0
    TH0 = 0x38; // Always load 55d in TL0.
    IE = 0x82;
}

void main(void)
{
    init_timer();
    start_timer();
    while (1)
    {} //Do nothing
}
```



# Counter Mode

## COUNTER PROGRAMMING

- ❑ Timers can also be used as counters counting events happening outside the 8051
  - 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 last section also applies to programming it as a counter
  - Except the source of the frequency

## COUNTER PROGRAMMING

### C/T Bit in TMOD Register

- ❑ The C/T bit in the TMOD registers decides the source of the clock for the timer
  - When  $C/T = 1$ , the timer is used as a counter and gets its pulses from outside the 8051
    - The counter counts up as pulses are fed from pins 14 and 15, these pins are called T0 (timer 0 input) and T1 (timer 1 input)

#### Port 3 pins used for Timers 0 and 1

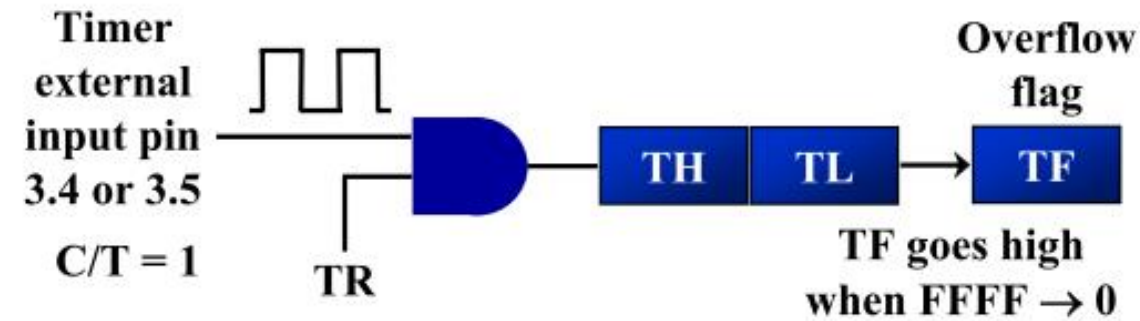
Pin	Port Pin	Function	Description
14	P3.4	T0	Timer/counter 0 external input
15	P3.5	T1	Timer/counter 1 external input



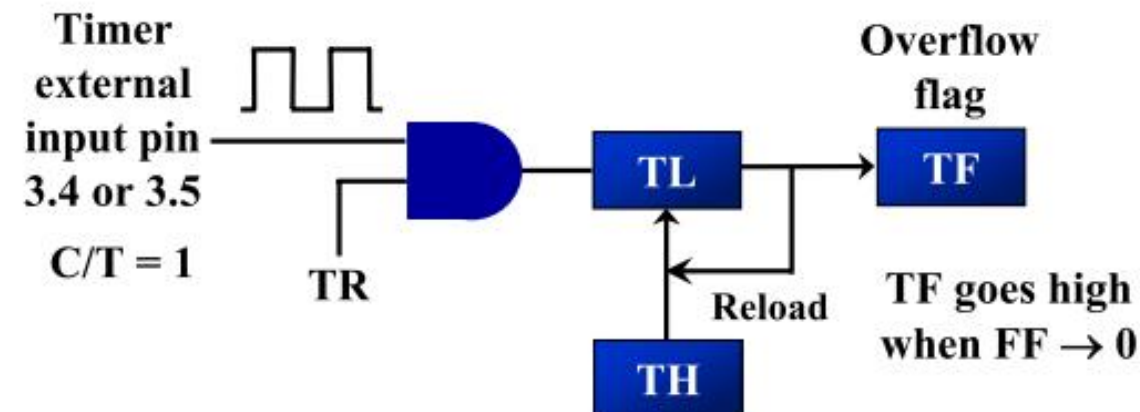
# COUNTER PROGRAMMING

## C/T Bit in TMOD Register (cont')

### Timer with external input (Mode 1)



### Timer with external input (Mode 2)



## COUNTER PROGRAMMING

### TCON Register

- ❑ TCON (timer control) register is an 8-bit register

TCON: Timer/Counter Control Register

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
-----	-----	-----	-----	-----	-----	-----	-----

The upper four bits are used to store the TF and TR bits of both timer 0 and 1

The lower 4 bits are set aside for controlling the interrupt bits

The diagram illustrates the hardware setup for the AT89C51 microcontroller. The microcontroller is represented by a central block with its pinout labeled. The pins are connected as follows:

- XTAL1** (Pin 19) and **XTAL2** (Pin 18) are connected to a common ground point.
- RST** (Pin 9) is connected to a common ground point through a 10k resistor (R1).
- PSEN** (Pin 29), **ALE** (Pin 30), and **EA** (Pin 31) are connected to a common ground point.
- P1.0** (Pin 1), **P1.1** (Pin 2), **P1.2** (Pin 3), **P1.3** (Pin 4), **P1.4** (Pin 5), **P1.5** (Pin 6), **P1.6** (Pin 7), and **P1.7** (Pin 8) are connected to the segments of the 7-segment display.
- P0.0/AD0** (Pin 39), **P0.1/AD1** (Pin 38), **P0.2/AD2** (Pin 37), **P0.3/AD3** (Pin 36), **P0.4/AD4** (Pin 35), **P0.5/AD5** (Pin 34), **P0.6/AD6** (Pin 33), and **P0.7/AD7** (Pin 32) are connected to a common ground point.
- P2.0/A8** (Pin 21), **P2.1/A9** (Pin 22), **P2.2/A10** (Pin 23), **P2.3/A11** (Pin 24), **P2.4/A12** (Pin 25), **P2.5/A13** (Pin 26), **P2.6/A14** (Pin 27), and **P2.7/A15** (Pin 28) are connected to a common ground point.
- P3.0/RXD** (Pin 10), **P3.1/TXD** (Pin 11), **P3.2/INT0** (Pin 12), **P3.3/INT1** (Pin 13), **P3.4/T0** (Pin 14), **P3.5/T1** (Pin 15), **P3.6/WR** (Pin 16), and **P3.7/RD** (Pin 17) are connected to a common ground point.

The AT89C51 microcontroller is shown with its pinout table:

Pin	Function
1	P1.0
2	P1.1
3	P1.2
4	P1.3
5	P1.4
6	P1.5
7	P1.6
8	P1.7
9	RST
10	P3.0/RXD
11	P3.1/TXD
12	P3.2/INT0
13	P3.3/INT1
14	P3.4/T0
15	P3.5/T1
16	P3.6/WR
17	P3.7/RD
18	XTAL2
19	XTAL1
21	P2.0/A8
22	P2.1/A9
23	P2.2/A10
24	P2.3/A11
25	P2.4/A12
26	P2.5/A13
27	P2.6/A14
28	P2.7/A15
29	PSEN
30	ALE
31	EA
32	P0.7/AD7
33	P0.6/AD6
34	P0.5/AD5
35	P0.4/AD4
36	P0.3/AD3
37	P0.2/AD2
38	P0.1/AD1
39	P0.0/AD0

```
#include <reg51.h>
#include <stdio.h>
sbit input_pin = P3^4;

void start_timer(void)
{
    TR0 = 1;
}

void init_timer(void)
{
    TMOD = 0x06; // Timer0 as counter (8-bit auto-reload mode)
    TH0 = 0;
    input_pin = 1; // Make P3.4 as input
}

void main(void)
{
    init_timer();
    start_timer();
    while (1)
    {
        P1 = TL0;
    }
}
```