XMUT202 Digital Electronics

Tutorial 6: Timer and Counter (Solution)

A. Timer and Counter in 8051

1. What are the timers in 8051 microcontrollers? Describe them briefly. [2 marks]

   Answer

   The 8051 microcontroller has two timers: Timer 0 and Timer 1. Each timer provides timing to the program and can operate in different modes, providing flexibility in terms of functionality and applications.

2. Describe the four modes of Timer 0 and 1 in 8051 microcontrollers. [4 marks]

   Answer

   Modes of Timer 0:

   Mode 0: 13-bit timer with 8 bits in TH0 and 5 bits in TL0.

   Mode 1: 16-bit timer with TH0 and TL0 forming a 16-bit timer.

   Mode 2: 8-bit auto-reload timer. TL0 is reloaded from TH0 when it overflows.

   Mode 3: Two 8-bit timers operating independently as Timer 0 (TH0, TL0) and Timer 1 (TH1, TL1).

   Modes of Timer 1:

   Mode 0: 13-bit timer with 8 bits in TH1 and 5 bits in TL1.

   Mode 1: 16-bit timer with TH1 and TL1 forming a 16-bit timer.

   Mode 2: 8-bit auto-reload timer. TL1 is reloaded from TH1 when it overflows.

   Mode 3: Two 8-bit timers operating independently as Timer 0 (TH0, TL0) and Timer 1 (TH1, TL1).

3. List and explain the characteristics of Timers 0 and 1 in 8051 when operated in modes 0 and 1 by completing the blank spaces provided in the table below. [8 marks]

	Mode 0 (13-bit Timer):	**Mode 1 (16-bit Timer)**
Timer 0	• ______________________________ • ______________________________ • ______________________________	• ______________________________ • ______________________________ • ______________________________

	• ______________________________	• ______________________________
Timer 1	• ______________________________ • ______________________________ • ______________________________ • ______________________________	• ______________________________ • ______________________________ • ______________________________ • ______________________________

Answer

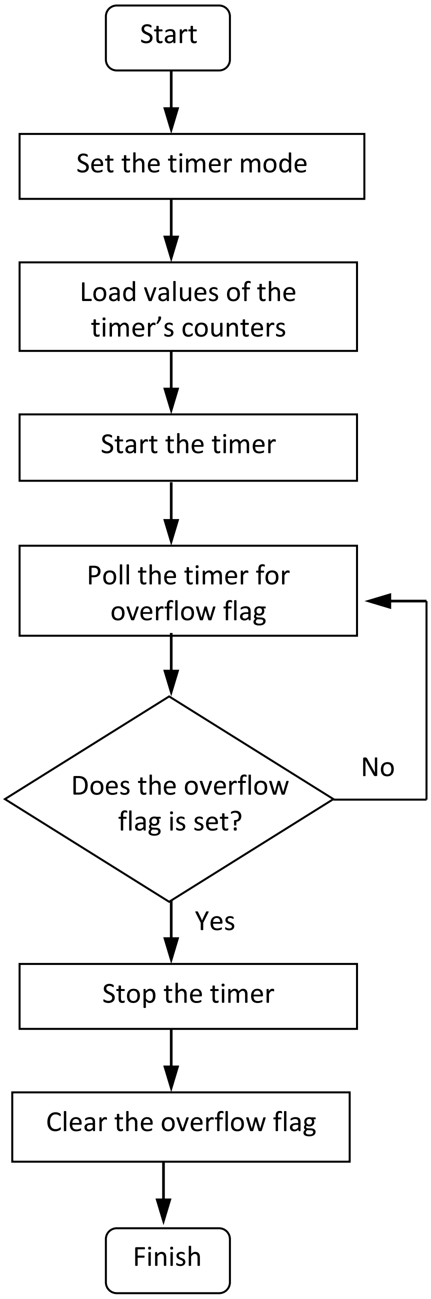
The modes of operation of the Timer in 8051 microcontrollers are listed in the table below.

	Mode 0 (13-bit Timer):	**Mode 1 (16-bit Timer)**
Timer 0	• Timer 0 is configured as a 13-bit timer. • It operates as a software programmable counter/timer with a maximum count of 8191. • It does not have any hardware interrupts associated with it. • It's suitable for basic timing functions.	• Timer 0 is configured as a 16-bit timer. • It operates as a software programmable counter/timer with a maximum count of 65535. • Mode 1 is commonly used for time delays and other applications where a longer timer duration is needed. • It generates an interrupt when it overflows.
Timer 1	• Like Mode 0 for Timer 0, Timer 1 is configured as a 13-bit timer. • It operates as a software programmable counter/timer with a maximum count of 8191. • It does not have any hardware interrupts associated with it. • Mode 0 for Timer 1 is suitable for basic timing functions.	• Like Mode 1 for Timer 0, Timer 1 is configured as a 16-bit timer. • It operates as a software programmable counter/timer with a maximum count of 65535. • Mode 1 for Timer 1 is commonly used for time delays and applications requiring a longer timer duration. • It generates an interrupt when it overflows

4. Using a flow chart diagram, explain the steps for configuring and running a timer programmed in 8051 microcontrollers. [6 marks]

   Answer

   The following flowchart diagram shows the step-by-step process for configuring and running the timer in 8051 microcontrollers.



5. Describe the function of watchdog timer in 8051 microcontrollers. What is its main use?

[2 marks]

Answer

- Microcontrollers' watchdog timers (WDT's) will automatically reset the system (moving PC to 0, etc.) if they are allowed to overflow.
- This will allow us to reset our microcontroller even if the program hangs up on some routine i.e. useful to let us automatically recover from unexpected 'crashes.'

B. Timer and Counter Programming

1. The following shows crystal frequency for three different 8051-based microcontroller systems. Find the period of the machine cycle in each case. [6 marks]

   a. 11.0592 MHz b. 16 MHz c. 20 MHz

Solution

a. f_{osc} = 11.0592 MHz/12 = 921.6 kHz; Machine Cycle (MC) = 1/921.6 kHz = 1.085 μs.

b. f_{osc} = 16 MHz/12 = 1.333 MHz; MC = 1/1.333 MHz= 0.75 μs.

c. f_{osc} = 20 MHz/12 = 1.66 MHz; MC = 1/1.66 MHz= 0.60 μs.

2. For an 8051-microcontroller system of 11.0592 MHz, find how long it takes to execute each of the following instructions. [14 marks]

   a. `MOV R3, #55`

   b. `DEC R3`

   c. `DJNZ R2, target`

   d. `LJMP`

   e. `SJMP`

   f. `NOP`

   g. `MUL AB`

Solution

The machine cycle for a 8051 microcontroller system of 11.0592 MHz is:

$$t_{osc} = \frac{\text{(clock/cycle)}}{f_{osc}} = \frac{12}{11.0592 \times 10^6} = 1.085\ \mu\text{s}$$

The table in Appendix shows machine cycles for each of the above instructions. Therefore, we have:

	Instruction	**Machine cycles**	**Time to execute**
a	`MOV R3, #55`	1	1 x 1.085 μs = 1. 085 μs
b	`DEC R3`	1	1 x 1.085 μs = 1. 085 μs
c	`DJNZ R2,`	2	2 x 1.085 μs = 2.17 μs

	target		
d	LJMP	2	2 x 1.085 μs = 2.17 μs
e	SJMP	2	2 x 1.085 μs = 2.17 μs
f	NOP	1	L x 1.085 μs = 1.085 μs
G	MUL AB	4	4 x 1.085 μs = 4. 34 μs

3. Find the delay introduce in the DELAY subroutine in the following program if the crystal frequency is 11.0592 MHz. [5 marks]

Program:

```
        MOV A, #55H       ;load register A with 0x55H
AGAIN:  MOV P1, A         ;issue value in register A to Port 1
        ACALL DELAY       ;jump to time delay subroutine
        CPL A             ;complement register A
        SJMP AGAIN        ;keep doing this indefinitely
        ;time delay subroutine
DELAY:  MOV R3, #200H     ;load R3 with 200
HERE:   DJNZ R3, HERE     ;stay here until R3 become 0
        RET               ;return to main program
```

Solution

The machine cycle for a 8051 microcontroller system of 11.0592 MHz is:

$$t_{osc} = \frac{(\text{clock/cycle})}{f_{osc}} = \frac{12}{11.0592 \times 10^6} = 1.085\ \mu\text{s}$$

From the table of instruction sets for the 8051 microcontrollers, we have the following machine cycles for each of the instructions in the DELAY subroutine.

Instruction		**Machine Cycle**
DELAY:	MOV R3, #200H	1
HERE:	DJNZ R3, HERE	2
	RET	2

Therefore, we have a total time delay for the DELAY subroutine of:

$t_{delay} = t_{MOV} + 200 \times (t_{DJNZ}) + t_{RET}$

= [1 + (200 x 2) + 2] x 1.085 μs = 436.255 μs

4. Describe the following assembly language program for the 8051 microcontrollers that set Timer 0 to work in Mode 1 and count from 0FF00H. Hint: use the structure of the program provided to guide you writing each of the code descriptions. [7 marks]

Program:

```
                ; Set timer mode
                MOV TMOD, #01H
                ; Load initial value for the counter
                MOV TH0, #0FFH
                MOV TL0, #00H

                ; Start timer
                SETB TR0
                ; Wait for timer to overflow
OVERFLOW:       JNB TF0, OVERFLOW
                ; When timer overflows, stop it
                CLR TR0
                CLR TF0
```

Solution

Explanations of the program given above:

- MOV TMOD, #01H

  This instruction sets the Timer Mode register (TMOD) to 01H. This configures Timer 0 in mode 1 (16-bit timer mode), which means it can count up to 65,535 (i.e. 2^16 - 1) in a 16-bit counter.

- MOV TH0, #0FFH and MOV TL0, #00H

  These instructions load the initial values for the high byte (TH0) and low byte (TL0) of the Timer 0 register. The initial value here is 65,535 (i.e. FF00H in hex).

- SETB TR0

  This instruction sets the TR0 bit (Timer Run bit) in the Timer Control register (TCON). This starts Timer 0 and begins the counting process.

- OVERFLOW

  This is a label to mark a point in the code where the timer might have overflowed.

- JNB TF0, OVERFLOW

  This instruction checks if the Timer Flag 0 (TF0) bit in the Timer Control register is set. If it's not set (meaning the timer hasn't overflowed), the code jumps back to the overflow label.

- CLR TR0

  Once the timer flag is set, this instruction clears the TR0 bit, which stops the timer.

- CLR TF0

  This instruction clears the TF0 bit, preparing the flag for the next cycle.

5. With careful set up and configuration, the timers in the 8051 microcontrollers are capable for producing precise timing for execution of a program. You are required to design an assembly language program that will turn on and off an LED connected to Port P1.1 every 1 s.

   a. Assume the crystal oscillator frequency is 11.0592 MHz, calculate the time taken to execute 1 machine cycle. [4 marks]

   b. Determine the value to be loaded into the timer counter special function register, so it will produce a delay of 1 ms. [4 marks]

   c. Write an assembly-language program to generate a square wave on port P1.1 with an ON time (high time) of 1 ms. Use Timer 1 and configure it as Mode 1. [12 marks]

Solution

a. Assuming the crystal frequency, f is 11.0592 MHz, the timing of the crystal oscillator clock needed for the 8051 microcontrollers, t_{osc} is:

$$t_{osc} = \frac{(\text{clock/cycle})}{f_{osc}} = \frac{12}{11.0592 \times 10^6} = 1.085 \times 10^{-6}\ \text{s}$$

b. Knowing that one cycle will take 1.085 µs, then the number of counts that will be required for 1 ms delay is:

$$\text{Delay Count} = \frac{t_{delay}}{t_{osc}} = \frac{1 \times 10^{-3}}{1.085 \times 10^{-6}} = 921$$

Thus, in hexadecimal, this value in the Timer 1 in Mode 1 (i.e. 16-bit timer) is:

$$\text{Reload Value} = (2^{16} - 1) - 921 = 64{,}614 = \text{0xFC66H}$$

Thus, the count FC66H to be loaded into TH and TL registers of the Timer 1 i.e. TH = 0xFCH and TL = 0x66H.

c. The following assembly language program configures and sets up Timer 1 in Mode 1 and produces square wave for 1 ms at the Port P1.1 as described above.

   Program:

```
MOV TMOD, #10H          ;configure Timer 1 in Mode 1
REPEAT:
MOV TL1, #66H           ;load count in registers TH1 and TL1
MOV TH1, #FCH
SETB P1.1               ;set P1.1 high to get ON part of
                        ;square wave
SETB TR1                ;start Timer 1
HERE:
JNB TF1, HERE           ;wait until timer overflows
CLR P1.1                ;clear P1.1 to get OFF part of square wave
CLR TR1                 ;stop timer
CLR TF1                 ;clear overflow flag
SJMP REPEAT             ;reload timer registers and repeat
```