# EXPERIMENT 7: The 8051 Timers

The basic 8051 has two on-chip timers that can be used for timing durations or for counting external events (AT89C52 has three on-chip timers).

Interval timing allows the programmer to perform operations at specific instants in time. For example, in our LED flashing program the LED was turned on for a specific length of time and then turned off for a specific length of time. We achieved this through the use of time delays. Since the microcontroller operates at a specific frequency, we could work out exactly how many iterations of the time delay was needed to give us the desired delay.

However, this is cumbersome and prone to error. And there is another disadvantage; the CPU is occupied, stepping through the loops. If we use the on-chip timers, the CPU could be off doing something more useful while the timers take on the menial task of keeping track of time.

**Timers 0 and 1**

Timers 0 and 1 are fundamentally the same and both have two 8-bit registers, timer high (TH) byte and timer low (TL) byte. Both share the timer control (TCON) register and the timer mode (TMOD) register.

The timers can be configured into one of the four modes:

- **Mode 0** TH and TL come together to form a 13-bit register where TH has 5 bits. This makes the microcontroller compatible with an earlier device.
- **Mode 1** TH and TL come together to form a 16-bit register.
- **Mode 2** TL is the working 8-bit register and TH is the automatic reload register. This mode is used to define the baud rate of the serial UART interface.
- **Mode 3** TH and TL registers of both timers combine to produce three 8-bit timers.

Details of the TMOD SFR are:

TMOD

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

This register has two identical halves, the upper four bits for timer 1 and the lower four bits for timer 0. The bits M1 and M0 set the TMOD:

| M1 | M0 | |
|---|---|---|
| 0 | 0 | Mode 0 |
| 0 | 1 | Mode 1 |
| 1 | 0 | Mode 2 |
| 1 | 1 | Mode 3 |

**MODE 1 16-BIT UP COUNTER**

Figure 4.1 shows the timer in mode 1 with the TH and TL registers together forming a 16-bit register. The diagram also shows that for this microcontroller family, **the timer clock is one-twelve of the microcontroller clock**. The default value of TMOD is 00H, so C/T=0 and the peripheral defaults to being a timer rather than a pulse counter. The default value of GATE is 0 and this is inverted, so the OR output defaults to logic 1. The timer is turned on or off by TR1/0 (in TCON register), putting TR0=1 would turn timer 0 on.

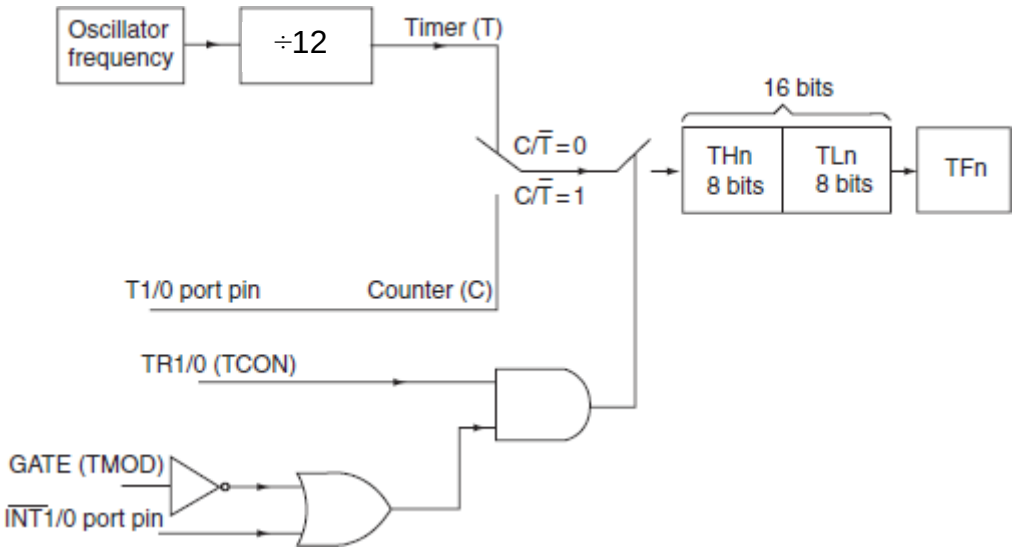



**Figure 4.1** Circuit for timer 0/1 operating as a 16-bit up-counter in mode 1

In mode 1, the TH and TL registers in timer 0 or timer 1 join to form a 16-bit up counter. The counter can be loaded with a base number from which the timer can increment upwards towards the 16-bit maximum of 65535 (FFFFH). The time taken to count from the base number to the maximum count value is the required delay. Figure 4.2 shows the method of achieving the required delay.

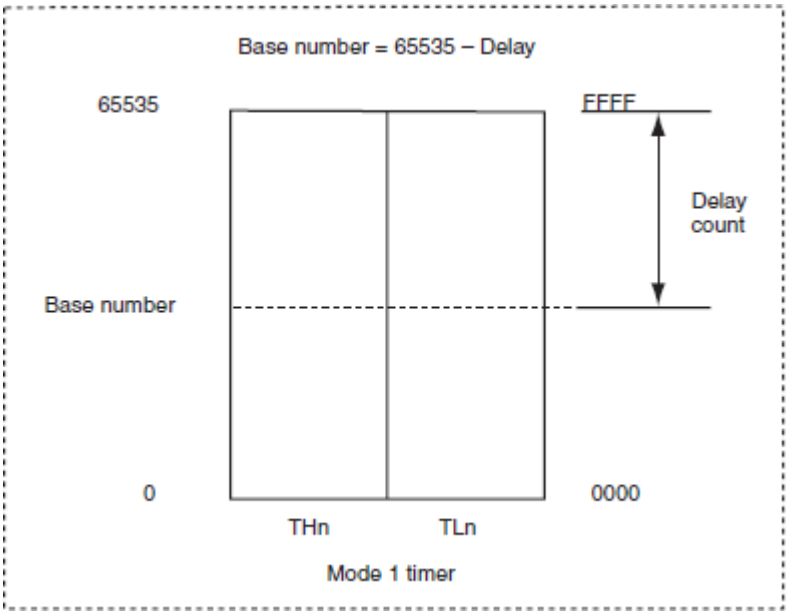



**Figure 4.2** Determination of delay count for a mode 1 timer

**Example 4.1** A 1 kHz square-wave signal is to be generated from pin 7 on port 1. The microcontroller clock frequency is 11.0592 MHz.

(a) Determine the required delay time.

(b) Using timer 0 determine the base numbers that must go into TH0 and TL0.

**Solution**

The required waveform is shown in Figure 4.3.

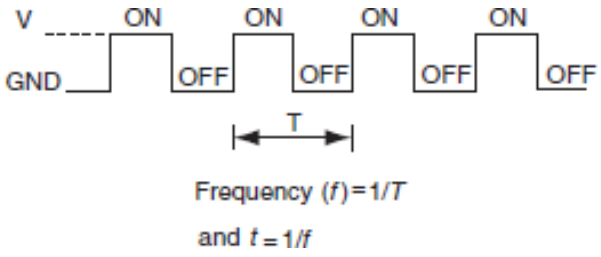



**Figure 4.3** Square-wave signal to be generated at port 1, pin 7

(a) One cycle time T of the required square-wave signal equals 1/frequency T=1/1000 Hz = 1ms

Delay time = T/2 = 0.5ms

(b) Timer clock = microclock/12

= 11.0592MHz/12 = 0.9216 Mhz

Timer cycle time = 1/0.9216MHz = 1.0851μs Delay count = (delay time) / (timer cycle time)

= 0.5ms / 1.0851 μs =460.8 ≅ 461 (nearest whole number) Base number = 65535 – delay count

= 65535 – 461 = 65074

*65074 decimal = FE32 hexadecimal or a different way, TH0 = whole number of 65074/256 65074/256 = 254.1953 = whole number of 254 TH0 = 254 assembly language MOV TH0, #0FEH TL0 = (remainder of 65074/256 )x256 0.1953x256 = 49.9968 ≅ 50

TL0 = 50 decimal and assembly language MOV TL0, #32H

**MODE 2 EIGHT-BIT UP-COUNTER**

The instruction **MOV TMOD, #02H** would put timer 0 into mode 2 defining an 8-bit timer using TL0 as the working register and TH0 as the automatic reload register. The circuit arrangement is shown in Figure 4.4.

In mode 2, the working register is only 8 bits wide and so the base number is 8 bits wide. When the TL register rolls over it is automatically reloaded with the contents of the TH register, whose loaded contents remain the same, so the base number goes into the TH register.

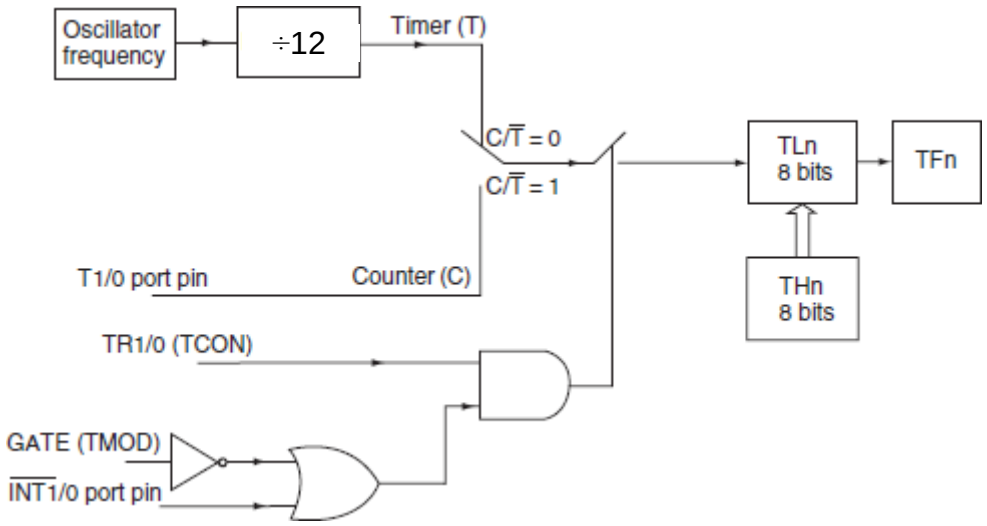



**Figure 4.4** Circuit for timer 1/0 to operate as an 8-bit up-counter in mode 2

**Example 4.3** AT89C52 microcontroller having an 11.0592 MHz clock is to be used to generate a 5 kHz square-wave signal from pin 7 of port 1. Write a suitable program to achieve this.

**Solution**

$Square-wave\ cycle\ time = \frac{1}{5KHz}$=0.2 ms

$Delay\ required\ for\ a\ square\ wave = half\ the\ cycle\ time$ = 0.1 $ms$ = 100μ$s$

$Timer\ 0\ clock = \frac{mikroclock}{12} = \frac{11.0592}{12}$

$Timer\ 0\ clock$ cycle= $\frac{12}{11.0592}$=1.0851 μ$s$

$Delay\ count = \frac{delay\ time}{timer\ clock\ cycle\ time} = \frac{100\mu s}{1.0851\mu s}$=92.1574

$Mode\ 2\ timer\ base\ number = 255 - delay\ count$

(255 = $maximum\ value\ of\ 8-bit\ register$)

= 255 − 92.1574 = 162.8426 ≅ 163 (*to nearest whole number*)

= $A3\ hex$

A3 hex is to go into TH0

A3 hex is to go into TL0

TL0 could start with its default value of 00H since the first half cycle would not be seen on an oscilloscope screen! The line MOV TL0, #0A3H can be left out of the program since after the first half cycle TL0 will automatically be reloaded with A3H from TH0.