Lecture-54

INTEL 8253: Programmable Interval Timer

Intel 8253 programmable Timer/ counter is a specially designed chip for Intel microcomputer applications which require timing and counting operations. These timing and counting functions can be implemented through software. For example, let in an application, microprocessor is required to execute N different tasks and these tasks are to be executed at an interval of T seconds. The software solution would be to call a delay routine of T seconds after a task (say for example, 'i') is completed and then do next task (say 'j'). In a software delay subroutine, either a register or a register pair is to initialized and decremented continuously till it become zero. In order to maintain the precision of the delay, it will not be possible for the microprocessor to execute any other task during this interval. If there are more such tasks, then microprocessor will be busy most of the time to execute the delay routines. If microprocessor has to perform some other useful task during (calculation), which is common in control applications, then it is very difficult.

The other possible solution is use of external timer. The µC may start this timer with a programmable value after executing the task 'i', then µC is free to do something else. This external timer may be down counter or up counter. The external timer after a delay of T seconds interrupts the µP. The µC executes task 'j' once it gets this interrupt. Such external device is called a programmable timer.

The Intel 8253 is a programmable counter/timer chip designed for use as an Intel µC peripheral. INTEL 8253 chip consists of three

identical 16-bit timers TM_0, TM_1 and TM_2. The timers are basically 16-bit down counters and counts HIGH to LOW transition at CLK input. Each timer may be programmed to operator in one of the six modes, independent of the mode of operation of the other two timers. The timers are software programmable. The maximum clock input to the timer is 2.6 MHz. The main applications of Intel 8253 are as follows:

1. Interrupt a time sharing operating system at evenly spaced intervals so that it can switch a program. For example, in a microprocessor based data acquisition system, it is required to measure parameter(s) at regular intervals; the timer can be be programmed such that it generates interrupts at regular intervals.
2. Programmable one shot generator. It is used to generate a pulse (strobe) after a delay of desired period.
3. Serve as a programmable baud rate generator. The chip may be used to generate high frequency pulse train for setting baud rate in serial communication.
4. Measure time delays between external events. Due to high frequency input to the timer, it can also be used to measure the time interval between two events.
5. Count the number of times an event occurs. The chip can be used as a counter if external system is generating a pulse every time an event occurs.
6. Causes the processer to be interrupted after a programmed number of external events have occurred.
7. Real time clock.

The pin-configuration of Intel 8253 is shown in fig.10.1

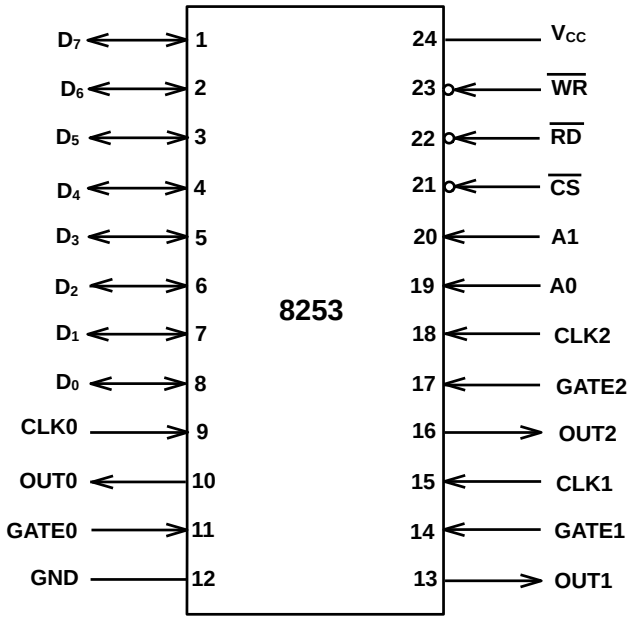

Fig.10.1 Pin Configuration of Intel 8253

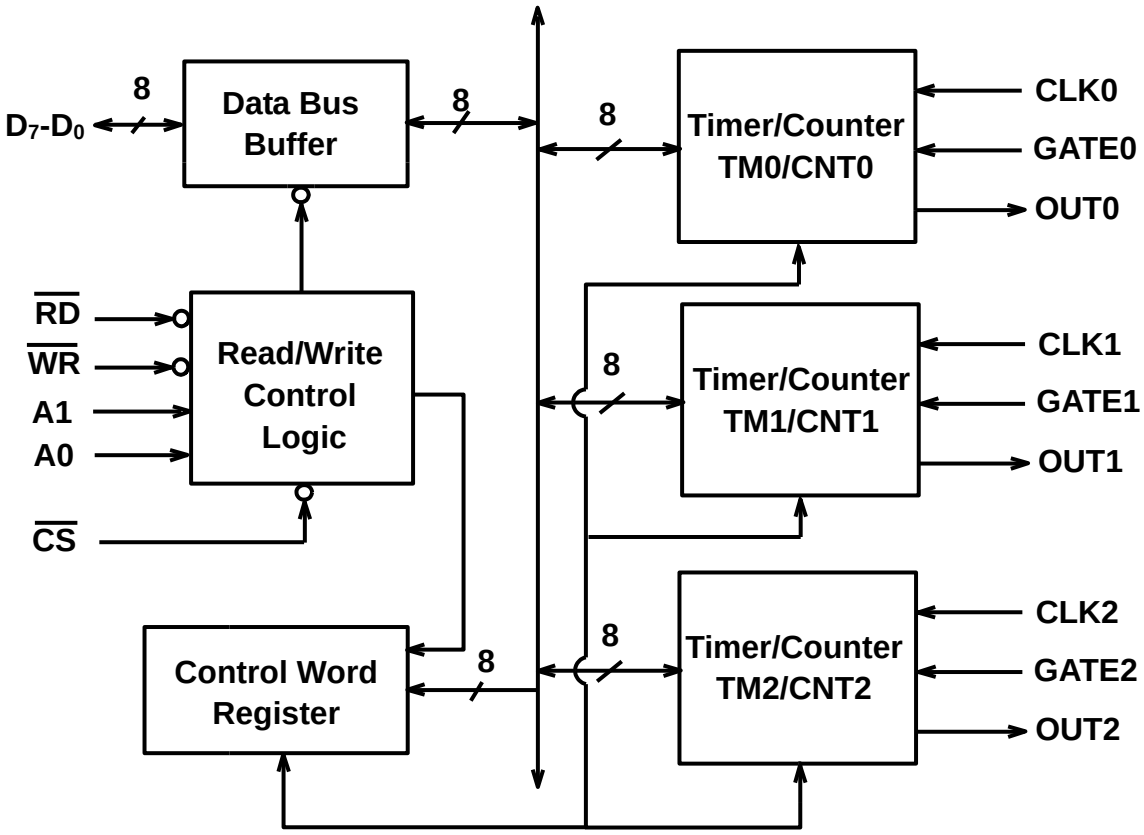

Fig.10.2 Functional Block Diagram of 8253

Functional Description & Pin Details:

The functional block diagram is shown in fig.10.2. The different blocks are described below:

Data Bus buffer:

The data bus buffer is bidirectional, 8-bit buffer and is used to interface the timer 8253 to the system data bus. The operation of this buffer is controlled by the chip select line ($\overline{CS}$) which tells the timer chip that the microprocessor is communicating with it, i.e., trying to transfer information to or from it even though $\overline{CS}$ is part of the READ/WRITE control logic. Data is transmitted or received by the buffer upon execution of IN PORT or OUR PORT instruction from CPU. The data bus buffer has three basic functions:

(i) Programming the modes of 8253.

(ii) Loading the count value in timers

(iii) Reading the count value from timers.

The data bus buffer is connected to processor using D_7-D_0 pins which are bidirectional. The data transfer takes place through these pins. These pins will be in high-impedance (or tri-state) condition until the 8253 is selected by a LOW on $\overline{CS}$ and either the read operation requested by a LOW on the $\overline{RD}$ input or a write operation requested by the $\overline{WR}$ input going LOW.

Read/ Write Logic:

It accepts inputs for the system control bus and in turn generates the control signals for overall device operation. It is

enabled or disabled by $\overline{CS}$ so that no operation can occur to change the function unless the device has been selected by the system logic.

$\overline{CS}$:

The chip select input is used to enable the communication between 8253 chip and the microprocessor by means of data bus. A low on $\overline{CS}$ enables the data bus buffers, while a high disables the buffer. The $\overline{CS}$ input does not have any affect on the operation of three timers once they have been initialized. The normal configuration of a system employs some decoding logic which actives $\overline{CS}$ line, whenever a specific set of addresses that correspond to 8253 appear on the address bus.

$\overline{RD}$ & $\overline{WR}$:

The read ($\overline{RD}$) and write ($\overline{WR}$) pins control the direction of data transfer on the 8-bit bus. When the $\overline{RD}$ input pin is low, then CPU is reading data from 8253 in the form of counter value. When $\overline{WR}$ pin is low, then CPU is writing data to 8253 in the form of mode information or loading initial count values to counters. The $\overline{RD}$ & $\overline{WR}$ should not both be low simultaneously. When $\overline{RD}$ & $\overline{WR}$ signals are HIGH, the data bus buffer is disabled and no data transfer takes place between processor and 8253.

A_0 & A_1:

These two input lines allow the microprocessor to specify which one of the internal register in the 8253 is going to be used for the data transfer. Fig shows how these two lines are used to select either the control word register or one of the 16-bit counters. For example, if there is a ‘1’ on both A_0 & A_1, and a ‘0’ on $\overline{WR}$, then the processor is

writing a control word to the control word register. These two pins are usually connected to the address bus lines, normally of the same name (A_0 & A_1).

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_1	A_0	Operation
0	1	0	0	0	Load counter '0'
0	1	0	0	1	Load counter '1'
0	1	0	1	0	Load counter '2'
0	1	0	1	1	Write mode word
0	0	1	0	0	Read TM_0
0	0	1	0	1	Read TM_1
0	0	1	1	0	Read TM_2
0	0	1	1	1	No- operation 3- state
1	X	X	X	X	Disable -- state
0	1	1	X	X	No- operation 3- state

Control Word Register:

The control word register is used to program the timers in different modes and control their operations. It is selected when A_0 and A_1 pins are '1 1'. If $\overline{WR}$ is also low, it accepts information from the data bus buffer and stores it in control word register. The information stored then controls the operation mode of each counter, selection of binary or BCD counting and the loading of each counting and the loading of each count register. This register can be written into only; no read operation of this content is available. The format will be discussed later on.

Counters:

Each of the timers has three pins associated with it. These are clock (CLK) input, the gate (GATE) control input and the output (OUT).

CLK:

This clock input pin provides 16-bit timer with the signal that causes the timer to decrement. The maximum clock frequency input is 2.6MHz. Note that the counters operate at HIGH to LOW transition (the negative edge) of this clock input. If the signal on this pin is generated by a fixed frequency oscillator then the user has implemented a standard timer for getting the desired delay. The count value to be loaded, in this case, determines the delay produced by the timer. If the input signal is a string of randomly occurring pulses, then it is called implementation of a counter. In this case, these clock pulses occur whenever an event takes place. Whenever a negative transition occurs at CLK input, the count value is decremented. From initial and value one can count the number of times an event has taken place in an interval.

GATE:

The gate input pin is used to initiate or enable counting. The exact effect of the gate signal depends on which of the six modes of operation is chosen.

OUTPUT:

The output pin provides an output from the timer. Its actual use depends on the mode of operation of the timer. The counter can be read "**on the fly**" without inhibiting gate pulse or clock input.

The timers are 16-bit down counters. These timers are loaded with an initial value before triggering. Every high to low transition at CLK input decrements the count by one. The operation of timers can be best understood by considering the internal structure of timer. Each timer is having three elements- input element (IE), counting element (CE) and output element (OE). These three elements are connected as shown in fig.10.3.

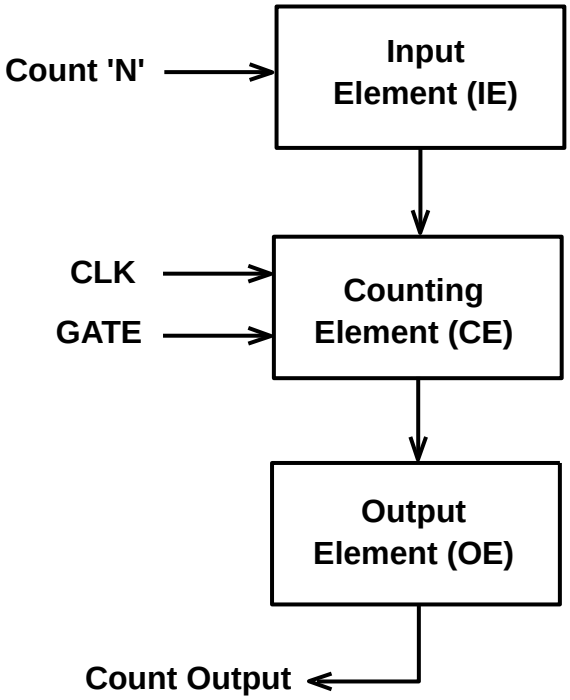

Fig.10.3 Internal Structure of Timer

The count value ‘N’ loaded into the timer is stored in input element and it remains in this element until it is reloaded with another count value. Depending on the mode in which the timer is programmed, this count value is transferred to counting element at an appropriate clock input. Once the value is transferred to CE, the down counting takes place in this element. The current count value at any moment is available at the input of output element. Output element is

basically a latch which, by default, is transparent and the output of latch follows the input, i.e, the current count value. The value can be latched in output element by issuing a proper control word. Once the output is latched, the internal down counting continues without affecting the output. The processor always read the output of output element to get the remaining count value. Once the timer is read, the output element automatically becomes transparent and starts following the internal count value. The count value can be latched again. If an attempt is made to latch the count value without reading the earlier latched value, it will be ignored.