

**Department of Computer System Engineering**

**COMPUTER DESIGN LAB**

**ENCS 411**

**Report 3**

**Experiment No. 8**

***Programmable Interval Timer (PIT)***

**Student Name:**

**Ibrahim Abu Farha 1110024**

**Othman Alkhamra 1110017**

**Instructor: Dr. Ahmad Afaneh**

**Teaching Assistant: Eng. Yazan Abu Farha**

**Section: 3**

**Date: 16/05/2014**

# **Introduction:**

## Programmable Interval Timer (PIT):

The [Intel](http://en.wikipedia.org/wiki/Intel) 8253 and 8254 are [Programmable Interval Timers](http://en.wikipedia.org/wiki/Programmable_Interval_Timer) (PITs), which perform timing and counting functions. In other words, it is counters which generate signal depending on programmable count or using trigger input**.** [1][2]

## The Internal Structure of the Programmable Interval Timer:



**Fig.1.1: 8254 PIT Internal Structure [1]**

As shown in **Fig.1.1**, the programmable interval timer has three counters, called channels. Each channel can be programmed to operate in one of six modes. Once programmed, the channels can perform their tasks independently. All channels are driven by a 1.19MHz oscillator signal. Each “tick” of the PIT generates hardware interrupt request 0. [1][3]

Furthermore, a PIT has a Control Word Register which contains the programmed information which will be sent -by the [microprocessor](http://en.wikipedia.org/wiki/Microprocessor)- to the device. It defines how the PIT logically works. Each access to these ports takes about 1 µs. The control word register contains 8 bits, labeled D7..D0 (D7 is the [MSB](http://en.wikipedia.org/wiki/Most_significant_bit)) see Fig.1.2. [1]



**Fig.1.2: Control Word Register Description [6]**

In Fig.1.1, a Read/Write Logic also appears which has 5 pins, which are listed below:

* $\overbar{RD}$: read signal
* $\overbar{WR}$: write signal
* $\overbar{CS}$: chip select signal
* A0, A1: address lines

Moreover, the Data Bus Buffer block contains the logic to buffer the data bus to / from the microprocessor, and to the internal registers. It has 8 input pins, usually labelled as D7…D0, where D7 is the [MSB](http://en.wikipedia.org/wiki/Most_significant_bit). [1]

## Programmable Interval Timer Modes:

As mentioned before, a PIT has six different operation modes. PITs may be one-shot or periodic.

* One-shot timers will signal only once and then stop counting. Periodic timers signal every time they reach a specific value and then restart, thus producing a signal at periodic intervals.
* Periodic timers are typically used to invoke activities that must be performed at regular intervals.

Counters are usually programmed with fixed intervals that determine how long the counter will count before it signals. The interval determines how long the counter will count before it will output a signal. [2]

The six modes of Programming Interval Timer are:

### Mode 0: Interrupt on Terminal Count:

Allows the 8254 counter to be used as an events counter. In this mode, the output becomes a logic 0 when the control word is written and remains there until N plus the number of programmed counts. For example, if a count of 5 is programmed, the output will remain a logic 0 for 6 counts beginning with N. Note that the gate (G) input must be a logic 1 to allow the counter to count. If G becomes a logic 0 in the middle of the count, the counter will stop until G again becomes a logic 1 see **Fig.1.3.** [4]



**Fig.1.3: Mode 0 [4]**

### Mode 1: Hardware Re-triggerable One-Shot:

Causes the counter to function as a retriggerable, monostable multivibrator (one-shot). In this mode the G input triggers the counter so that it develops a pulse at the OUT connection that becomes a logic 0 for the duration of the count. If the count is 10, then the OUT connection goes low for 10 clocking periods when triggered. If the G input occurs within the duration of the output pulse, the counter is again reloaded with the count and the OUT connection continues for the total length of the count **see Fig.1.4**.[4]



**Fig.1.4: Mode 1 [4]**

### Mode 2: Rate Generator:

Allows the counter to generate a series of continuous pulses that are one clock pulse wide. The separation between pulses is determined by the count. For example, for a count of 10, the output is a logic 1 for nine clock periods and low for one clock period. This cycle is repeated until the counter is programmed with a new count or until the G pin is placed at a logic 0 level. The G input must be a logic 1 for this mode to generate a continuous series of pulses **see Fig.1.5**. [4]



**Fig.1.5: Mode 2 [4]**

### Mode 3: Square Wave Generator:

Generates a continuous square wave at the OUT connection, provided that the G pin is a logic 1. If the count is even, the output is high for one half of the count and low for one half of the count. If the count is odd, the output is high for one clocking period longer than it is low. For example, if the counter is programmed for a count of 5, the output is high for three clocks and low for two clocks **see Fig.1.6**. [4]



**Fig.1.6: Mode 3 [4]**

### Mode 4: Software Triggered Strobe

Allows the counter to produce a single pulse at the output. If the count is programmed as a 10, the output is high for 10 clocking periods and low for one clocking period. The cycle does not begin until the counter is loaded with its complete count. This mode operates as a software triggered oneshot. As with modes 2 and 3, this mode also uses the G input to enable the counter. The G input must be a logic 1 for the counter to operate for these three modes **see Fig.1.7**. [4]



**Fig.1.7: Mode 4 [4]**

### Mode 5: Hardware Re-triggerable Strobe

A hardware triggered one-shot that functions as mode 4, except that it is started by a trigger pulse on the G pin instead of by software. This mode is also similar to mode 1 because it is retriggerable **see Fig.1.8**. [4]



**Fig.1.8: Mode 5 [4]**

# **References:**

[1] <http://en.wikipedia.org/wiki/Intel_8253>

**Access Date: 15 – 05 – 2014**

[2] <http://en.wikipedia.org/wiki/Programmable_interval_timer>

**Access Date: 15 – 05 – 2014**

[3] Programmable Interval Timer MDA-8086 Kit – PPI Application, Experiment 8.

**Access Date: 15 – 05 – 2014**

[4] Intel Microprocessor Architecture, Programming and Interfacing 8th Edition, Barry B. Brey

**Access Date: 15 – 05 – 2014**

[5] <http://wiki.osdev.org/Programmable_Interval_Timer>

**Access Date: 16 – 05 – 2014**