Modes

The following text describes all possible modes. The modes used in the MZ-700 and set by the monitor's startup are mode 0, mode 2, and mode 3.

- **Mode 0 (Interrupt on Terminal Count)**

The counter will be programmed to an initial value and afterwards counts down at a rate equal to the input clock frequency. When the count is equal to 0, the OUT pin will be a logical 1. The output will stay a logical 1 until the counter is reloaded with a new value or the same value or until a mode word is written to the device. Once the counter starts counting down, the GATE input can disable the internal counting by setting the GATE to a logical 0.

- **Mode 1 (Programmable One-Shot)**

In mode 1, the device can be setup to give an output pulse that is an integer number of clock pulses. The one-shot is triggered on the rising edge of the GATE input. If the trigger occurs during the pulse output, the 8253 will be retriggered again.

- **Mode 2 (Rate Generator)**

The counter that is programmed for mode 2 becomes a "divide by n" counter. The OUT pin of the counter goes to low for one input clock period. The time between the pulses of going low is dependent on the present count in the counter's register. I mean the time of the logical 1 pulse.
For example, suppose to get an output frequency of 1,000 Hz (Hertz), the period would be 1 / 1,000 s = 1 ms (millisecond) or 1,000 µs (microseconds). If an input clock of 1 MHz (Mega-Hertz) were applied to the clock input of the counter #0, then the counter #0 would need to be programmed to 1000 µs. This could be done in decimal or in BCD. (The period of an input clock of 1 MHz is 1 / 1,000,000 = 1 µs.)
The formula is: **n=f_i divided by f_{out}.**
f_i = input clock frequency, f_{out} = output frequency, n = value to be loaded.

My example: $f_i = 1$ MHz $= 1 \times 10^6$ Hz, $f_{out} = 1$ kHz $= 1 \times 10^3$ Hz.

$n = 1 \times 10^6$ Hz / 1×10^3 Hz $= 1 \times 10^3 = 1{,}000$. This is the decimal value to be loaded or the hexadecimal value $03E8. The following program example uses the decimal load count.

```
B000 3E35     LD      A,$35          ; load control word
                                     ; for counter 0 mode 2
B002 3207E0   LD      ($E007),A      ; into port $E007
                                     ; for BCD count
B005 2104E0   LD      HL,$E004       ; address to the port
                                      ; of counter 0
B008 3E00     LD      A,$00
B00A 77       LD      (HL),A         ; load least significant
                                     ; byte of 1000 first
B00B 3E10     LD      A,$10
B00D 77       LD      (HL),A         ; load most significant
                                     ; byte of 1000 last
B00E 3E01     LD      A,1
B010 3208E0   LD      ($E008),A      ; start counter 0 is only
                                     ; Necessary for the MZ-700.
                                     ; Not necessary for
                                    ; counter #1 and #2
; The counter is now initialized and the output frequency
; will be 1000 Hz if the input frequency is 1 MHz.
```

If the count is loaded between output pulses, the present period will not be affected. A new period will occur during the next count sequence.

- **Mode 3 (Square Wave Generator)**

Mode 3 is similar to the mode 2 except that the output will be high for half the period and low for half. If the count is odd, the output will be high for (n + 1) / 2 and low for (n - 1) / 2 counts.

For example, I'll setup counter #0 for a square wave frequency of 10 kHz (kilo-Hertz), assuming the input frequency is 1 MHz.

Please refer to the formula described at mode 2.
$1 \times 10^6 / 10 \times 10^3 = 100$. This is the decimal value to be loaded or the hexadecimal value $0064. The following program example uses the binary load count.

```
B000 3E35     LD      A,$36          ; load control word
                                     ; for counter 0 mode 3
B002 3207E0   LD      ($E007),A      ; into port $E007
                                     ; for binary count
B005 2104E0   LD      HL,$E004       ; address to the port
                                     ; of counter 0
B008 3E00     LD      A,$64          ; equals to
                                     ; 100 microseconds
```

```
                                        ; for 10,000 Hz
B00A 77       LD      (HL),A            ; load least significant
                                        ; byte of $0064 first
B00B 3E10     LD      A,$00
B00D 77       LD      (HL),A            ; load most significant
                                        ; nyte of $0064 last
B00E 3E01     LD      A,1
B010 3208E0   LD      ($E008),A         ; start counter 0 is only
                                        ; necessary for the MZ-700.
                                        ; Not necessary for counter
                                        ; #1 and #2
; The counter is now initialized and the output frequency
; will be 10 kHz if the input frequency is 1 MHz.
```

- **Mode 4 (Software Triggered Strobe)**

In this mode the programmer can set up the counter to give an output timeout starting when the register is loaded. On the terminal count, when the counter equals to 0, the output will go to a logical 0 for one clock period and then returns to a logical 1. First the mode is set, the output will be a logical 1.

- **Mode 5 (Hardware Triggered Strobe)**

In this mode the rising edge of the trigger input will start the counting of the counter. The output goes low for one clock at the terminal count. The counter is retriggerable, thus meaning that if the trigger input is taken low and then high during a count sequence, the sequence will start over.

When the external trigger input goes to a logical 1, the timer will start to time out. If the external trigger occurs again, prior to the time completing a full timeout, the timer will retrigger.

Programmable Interrupt Controller 8259A

- If we are working with an 8086, we have a problem here because the 8086 has only two interrupt inputs, NMI and INTR.
- If we save NMI for a power failure interrupt, this leaves only one interrupt for all the other applications. For applications where we have interrupts from multiple source, we use an external device called a ***priority interrupt controller*** (PIC) to the interrupt signals into a single interrupt input on the processor.

Architecture and Signal Descriptions of 8259A

- The architectural block diagram of 8259A is shown in fig1. The functional explication of each block is given in the following text in brief.
- **Interrupt Request Register (RR)**: The interrupts at IRQ input lines are handled by Interrupt Request internally. IRR stores all the interrupt request in it in order to serve them one by one on the priority basis.
- **In-Service Register (ISR)**: This stores all the interrupt requests those are being served, i.e. ISR keeps a track of the requests being served.

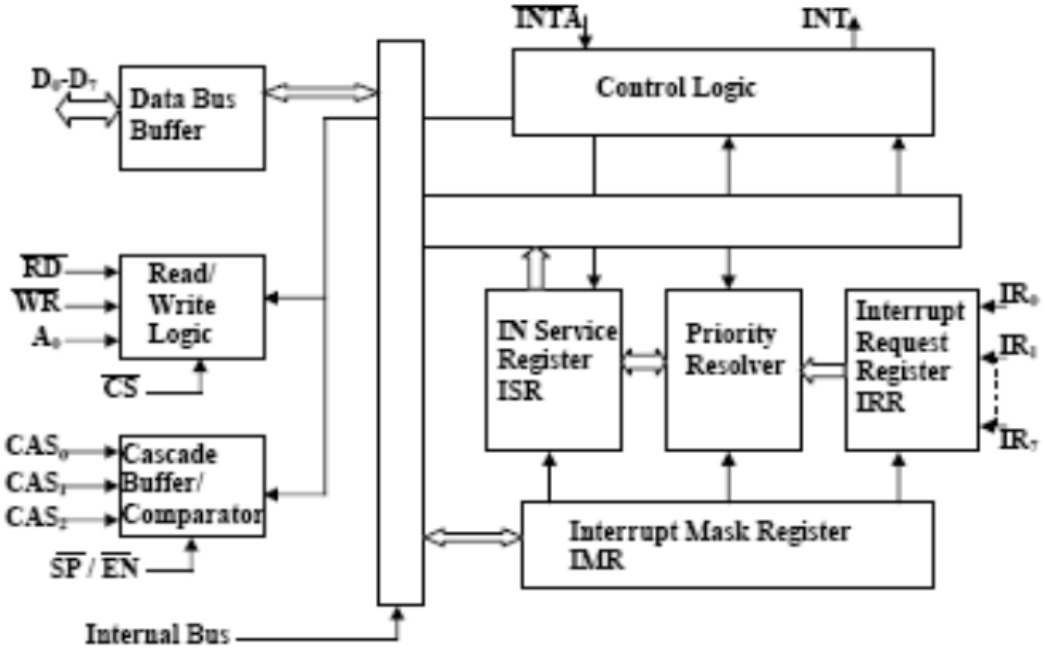

Fig:1 8259A Block Diagram

- **Priority Resolver :** This unit determines the priorities of the interrupt requests appearing simultaneously. The highest priority is selected and stored into the corresponding bit of ISR during INTA pulse. The IR0 has the highest priority while the IR7 has the lowest one, normally in fixed priority mode. The priorities however may be altered by programming the 8259A in rotating priority mode.
- **Interrupt Mask Register (IMR)** : This register stores the bits required to mask the interrupt inputs. IMR operates on IRR at the direction of the Priority Resolver.
- **Interrupt Control Logic**: This block manages the interrupt and interrupt acknowledge signals to be sent to the CPU for serving one of the eight interrupt requests. This also accepts the interrupt acknowledge (INTA) signal from CPU that causes the 8259A to release vector address on to the data bus.

- **Data Bus Buffer** : This tristate bidirectional buffer interfaces internal 8259A bus to the microprocessor system data bus. Control words, status and vector information pass through data buffer during read or write operations.
- **Read/Write Control Logic**: This circuit accepts and decodes commands from the CPU. This block also allows the status of the 8259A to be transferred on to the data bus.
- **Cascade Buffer/Comparator**: This block stores and compares the ID's all the 8259A used in system. The three I/O pins CASO-2 are outputs when the 8259A is used as a master. The same pins act as inputs when the 8259A is in slave mode. The 8259A in master mode sends the ID of the interrupting slave device on these lines. The slave thus selected, will send its preprogrammed vector address on the data bus during the next INTA pulse.
- **CS**: This is an active-low chip select signal for enabling RD and WR operations of 8259A. INTA function is independent of CS.
- **WR** : This pin is an active-low write enable input to 8259A. This enables it to accept command words from CPU.
- **RD** : This is an active-low read enable input to 8259A. A low on this line enables 8259A to release status onto the data bus of CPU.
- **D0-D7** : These pins from a bidirectional data bus that carries 8-bit data either to control word or from status word registers. This also carries interrupt vector information.
- **CAS0 – CAS2 Cascade Lines** : A signal 8259A provides eight vectored interrupts. If more interrupts are required, the 8259A is used in cascade mode. In cascade mode, a master 8259A along with eight slaves 8259A can provide upto 64 vectored interrupt lines. These three lines act as select lines for addressing the slave 8259A.
- **PS/EN** : This pin is a dual purpose pin. When the chip is used in buffered mode, it can be used as buffered enable to control buffer transreceivers. If this is not used in buffered mode then the pin is used as input to designate whether the chip is used as a master (SP =1) or slave (EN = 0).
- **INT** : This pin goes high whenever a valid interrupt request is asserted. This is used to interrupt the CPU and is connected to the interrupt input of CPU.
- **IR0 – IR7 (Interrupt requests)** :These pins act as inputs to accept interrupt request to the CPU. In edge triggered mode, an interrupt service is requested by raising an IR pin from a low to a high state and holding it high until it is acknowledged, and just by latching it to high level, if used in level triggered mode.

Pin Diagram

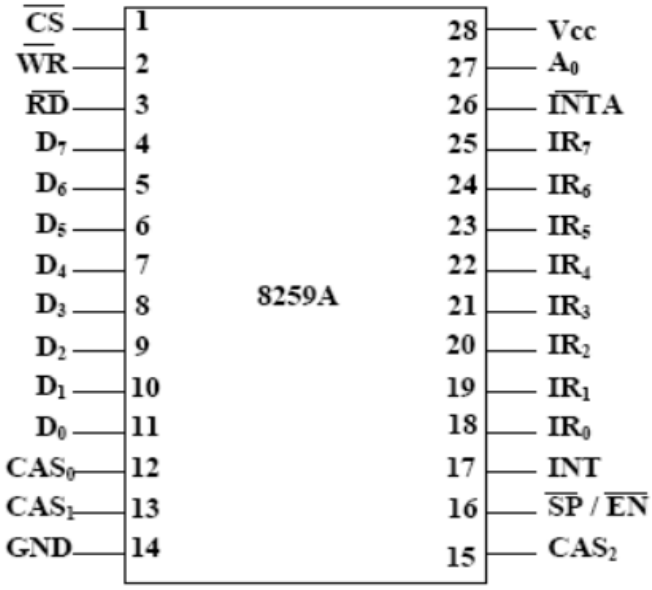

Fig : 8259 Pin Diagram

INTA (Interrupt acknowledge): This pin is an input used to strobe-in 8259A interrupt vector data on to the data bus. In conjunction with CS, WR and RD pins, this selects the different operations like, writing command words, reading status word, etc.

- The device 8259A can be interfaced with any CPU using either polling or interrupt. In polling, the CPU keeps on checking each peripheral device in sequence to ascertain if it requires any service from the CPU. If any such service request is noticed, the CPU serves the request and then goes on to the next device in sequence.
- After the entire peripheral device are scanned as above the CPU again starts from first device.
- This type of system operation results in the reduction of processing speed because most of the CPU time is consumed in polling the peripheral devices.
- In the interrupt driven method, the CPU performs the main processing task till it is interrupted by a service requesting peripheral device.
- The net processing speed of these type of systems is high because the CPU serves the peripheral only if it receives the interrupt request
- If more than one interrupt requests are received at a time, all the requesting peripherals are served one by one on priority basis.
- This method of interfacing may require additional hardware if number of peripherals to be interfaced is more than the interrupt pins available with the CPU.