8259A PROGRAMMABLE INTERRUPT C ONTROLLER (8259A/8259A-2)

Features

- Y 8086, 8088 Compatible
- Y MCS-80, MCS-85 ompatible
- Y Eight-Level Priority Controller
- Y Expandable to 64 Levels
- Y Programmable Interrupt Modes
- Y Individual Request Mask Capability
- Y Single a 5V Supply (No Clocks)
- Y Available in 28-Pin DIP and 28-Lead PLCC Package

4.6.1 FUNCTIONAL DESCRIPTION

Intel 8259A Programmable Interrupt Controller handles up to eight vectored priority interrupts for the CPU. It is cascadable for up to 64 vectored priority interrupts without additional circuitry. It is packaged in a 28-pin DIP, uses NMOS technology and requires a single a 5V supply. Circuitry is static, requiring no clock input.

The 8259A is designed to minimize the software and real time overhead in handling multi-level priority inter- rupts. It has several modes, permitting optimization for a variety of system requirements.

The 8259A is fully upward compatible with the Intel 8259. Software originally written for the 8259 will operate the 8259A in all 8259 equivalent modes (MCS-80/85, Non- Buffered, Edge Triggered).

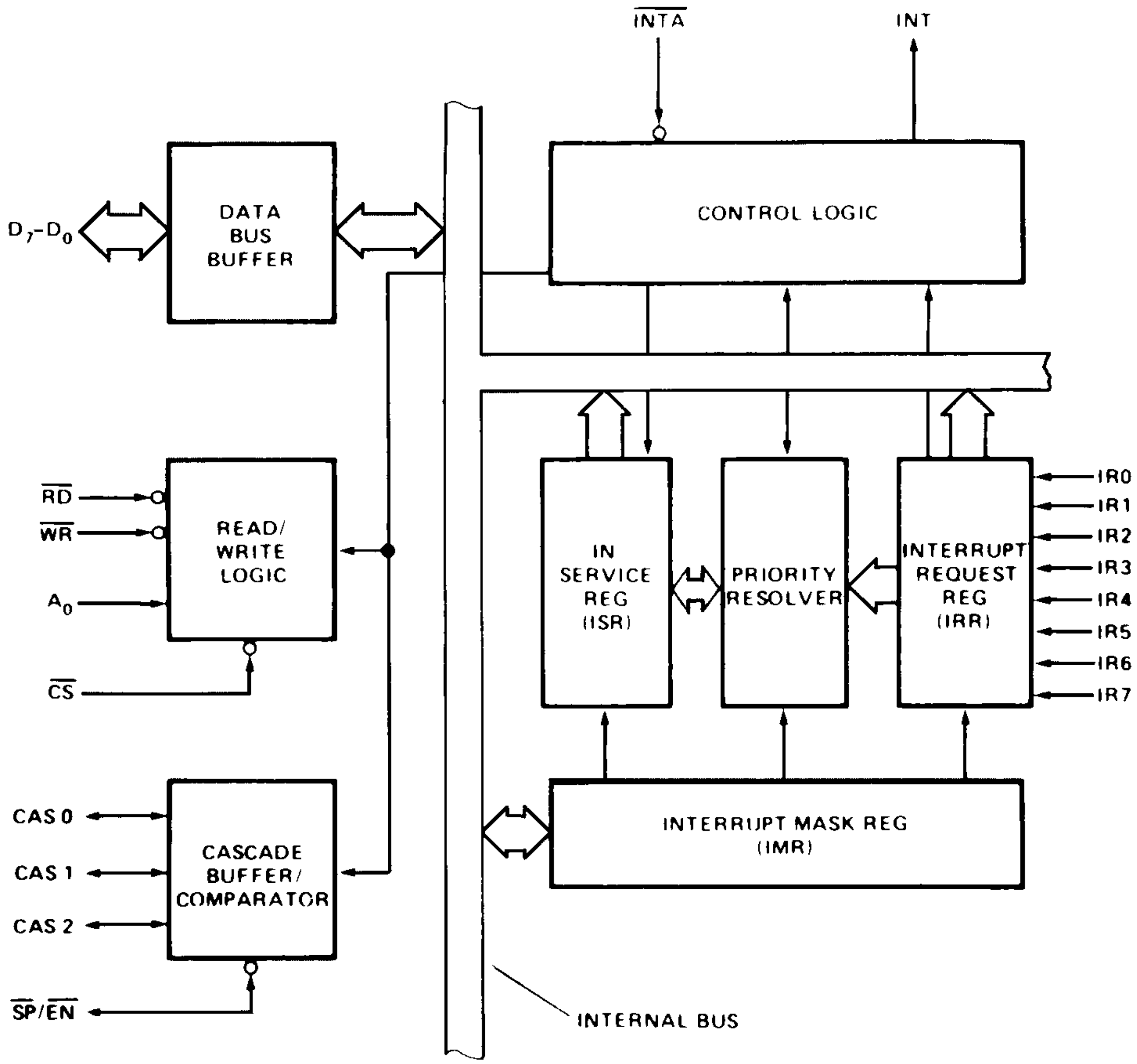
Interrupts in Microcomputer Systems

Microcomputer system design requires that I.O de- vices such as keyboards, displays, sensors and oth- er components receive servicing in a an efficient manner so that large amounts of the total system tasks can be assumed by the microcomputer with little or no effect on throughput.

The most common method of servicing such devic- es is the Polled approach. This is where the proces- sor must test each device in sequence and in effect „„ask"" each one if it needs servicing. It is easy to see that a large portion of the main program is looping through this continuous polling cycle and that such a method would have a serious detrimental effect on system throughput, thus limiting the tasks that could be assumed by the microcomputer and reducing the cost effectiveness of using such devices

A more desirable method would be one that would allow the microprocessor to be executing its main program and only stop to service peripheral devices when it is told to do so by the device itself. In effect, the method would provide an external asynchronous input that would inform the processor that it should complete whatever instruction that is currently being executed and fetch a new routine that will service the requesting device. Once this servicing is com- plete, however, the processor would resume exactly where it left off.

This method is called Interrupt . It is easy to see that system throughput would drastically increase, and thus more tasks could be assumed by the micro- computer to further enhance its cost effectiveness.



The Programmable Interrupt Controller (PIC) func- tions as an overall manager in an Interrupt-Driven system environment. It accepts requests from the peripheral equipment, determines which of the in- coming requests is of the highest importance (priori- ty), ascertains whether the incoming request has a higher priority value than the level currently being serviced, and issues an interrupt to the CPU based on this determination.

Each peripheral device or structure usually has a special program or „„routine"" that is associated with its specific functional or operational requirements; this is referred to as a „„service routine"". The PIC, after issuing an Interrupt to the CPU, must somehow input information into the CPU that can „„point"" the Program Counter to the service routine associated with the requesting device. This „„pointer"" is an ad- dress in a vectoring table and will often be referred to, in this document, as vectoring data.

Source : http://nprcet.org/e%20content/Misc/e-Learning/IT/IV%20Sem/CS%202252-Microprocessors%20and%20Microcontrollers.pdf