Interrupt Priority Management

The interrupt priority management logic indicated in Fig. 4.14 can be implemented in several ways. It does not need to be present in systems which use software priority management or simple daisy chaining, but more complex systems may require the efficiency gained by including hardware for managing the I/O interrupts. Many manufacturers have made priority management devices available and Intel is no exception. Although such a device made by one manufacturer could be used with processors made by other manufacturers, generally there are fewer compatibility problems if the CPU and interrupt priority device are produced by the same company. Therefore, we will be concerned with the Intel 8259A programmable interrupt controller (PIC), which has been specifically designed to work with the 8086/8088 as well as other members of the Intel microprocessor family.

The 8259A has been designed so that it can operate alone or in concert with other 8259As. In order to limit the initial discussion, an interrupt system involving a single 8259A device is considered first; then the discussion is extended to systems that can include as many as nine 8259As.

Interrupt System Based on a Single 8259A

The 8259A is contained in a 28-pin dual-in-line package that requires only a + 5-V supply voltage. Its organization is shown in Fig. 4.14 along with its connections to a maximum mode system. Its pins (other than the supply voltage and ground pins) are defined as follows:

D7-DO	–	For communicating with the CPU over the data bus. On a few systems bus drivers may be needed, but on other systems direct connections can be used.
INT	–	To send interrupt request signals to the CPU.
INTA	–	To receive interrupt acknowledge signals from the CPU. The 8259A assumes that an acknowledgment consists of two negative pulses, thus making it compatible with 8086/8088 systems.

RD – To signal the 8259A that it is to place the contents of the IMR, ISR, or IRR register or a priority level on the data bus. Which of these possibilities is placed on the bus depends on the state of the 8259A and is discussed below.

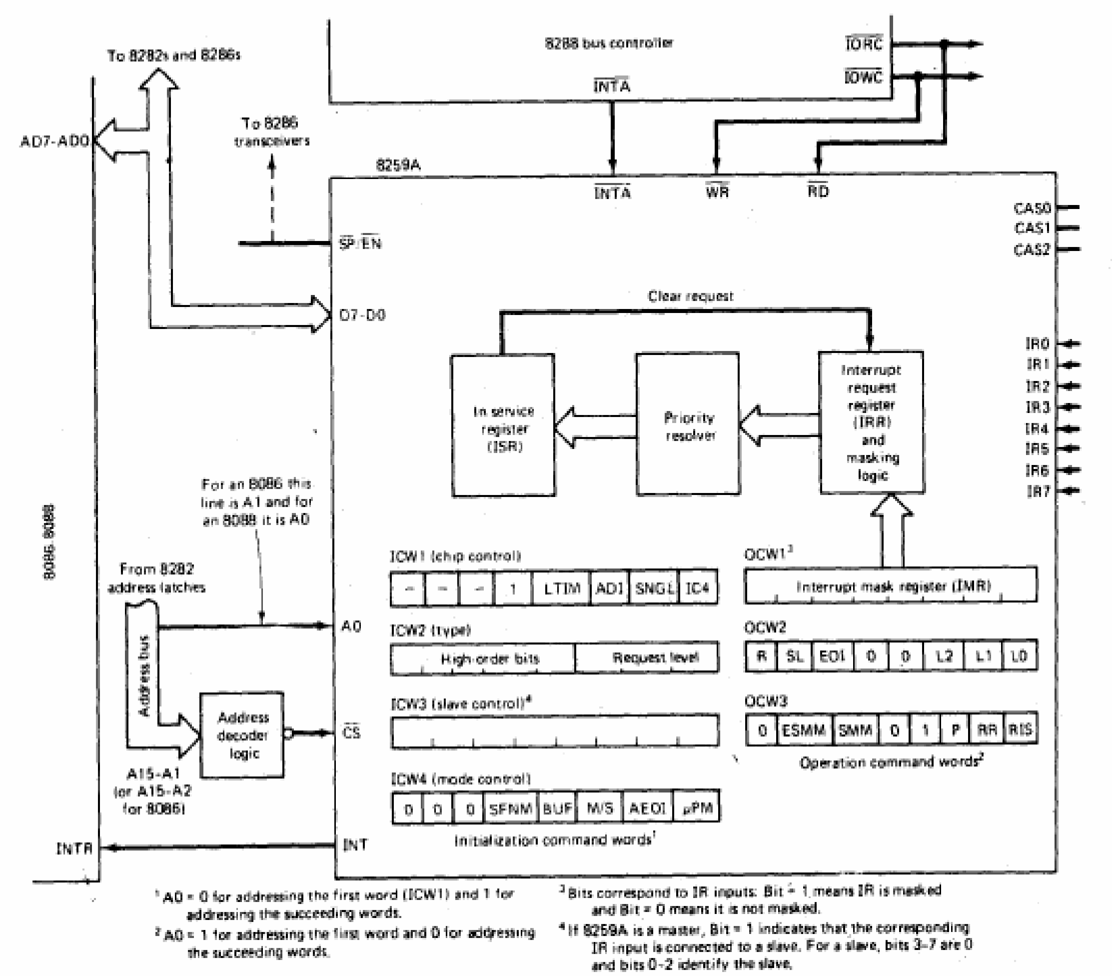

Figure 4.14: Organization of the 8259A programmable interrupt controller

WR –

To signal the 8259A that it is to accept data from the data bus and us the data to set

the bits in the command words. How the received data are distributed is discussed

later

CS – For indicating that the 8259A is being accessed. This pin is connected to the address bus through the decoder logic that compares the high-order bits of the address of the 8259A with the address currently on the address bus. Input to this pin can be combined with S2 to give the ready signal.

AO – For indicating which port of the 8259A is being accessed. Two addresses must be reserved in the I/O address space for each 8259A in the system.

IR7-IRO – For receiving interrupt requests from I/O interfaces or other 8259As referred to as slaves.

CAS2-CASO – To identify a particular slave device.

SP/EN – For one of two purposes; either as an input to determine whether the 8259A is to be a master (SP/EN = 1) or as a slave (SP/EN = 0), or as an output to disable the data bus transceivers when data are being transferred from the 8259A to the CPU. Whether the SP/EN pin is used as an input or output depends on the buffer mode discussed below.

For an 8088 the two addresses associated with an 8259A are normally consecutive, and the AO line is connected to the AO pin, but because there are only eight data pins on the 8259A and the 8086 always inputs the interrupt pointer from the lower 8 bits of its 16-bit data bus, all data transfers to and from the 8259A must be made over the lower byte of the bus. The easiest way to guarantee that all transfers will use the lower half of the bus is to connect the Al line to AO and use two consecutive even addresses, with the first being divisible by 4. However, to simplify the following discussion, the second address will be referred to as the odd address for both cases.

The control portion of the 8259A contains several programmable bits that can be viewed as being contained in seven 8-bit registers. These registers are divided into two groups, with one group containing the initialization command words (ICWs) and the other group containing the operation command words (OCWs). The initialization command words are normally set by an initialization routine when the computer system is first brought up and remain constant throughout its operation. By contrast, the operation command words are used to dynamically control the processing of interrupts.

The IRR (and its associated masking logic), priority resolver, and ISR are for receiving and controlling the interrupts that arrive at the IR7-IRO pins. The IRR latches the incoming requests and, in conjunction with the priority resolver, allows unmasked requests with sufficient priority to put a 1 on the INT pin. The priority resolver logic determines the priorities of the requests in the IRR and the ISR is for holding the requests currently being processed.

After a bit in the IRR is set to 1 it is compared with the corresponding mask bit in the IMR. If the mask bit is 0, the request is passed on to the priority resolver, but if it is 1, the request is blocked. When an interrupt request is input to the priority resolver its priority is examined and, if according to the current state of the priority resolver the interrupt is to be sent to the CPU, the INT line is activated.

Assuming that the IF flag in the CPU is 1, the CPU will enter its interrupt sequence at the completion of the current instruction and return two negative pulses over the INTA line. Upon the arrival of the first pulse, the IRR latches are disabled so that the IRR will ignore further signals on the IR7-IRO lines. This state is maintained until the end of the second INTA pulse. Also, the first INTA pulse will cause the appropriate ISR bit to be set and the corresponding IRR bit to be cleared. The second INTA pulse causes the current contents of ICW2 to be placed on D7 -DO, and the CPU uses this byte as the interrupt type. If the automatic end of interrupt (AEOI) bit in ICW4 is 1, at the end of the second INTA pulse the ISR bit that was set by the first INTA pulse is cleared; otherwise, the ISR bit is not cleared until the proper end of interrupt (EOI) command is sent to OCW2.

As indicated above, the initialization command words are normally filled by an initializing routine when the system is turned on and contain the control bits that are held constant throughout the system's operation. The 8259A has an even address (AO = 0) and an odd address (AO = 1) associated with it and the initialization command words must be filled consecutively by using the even address for ICWI and the odd address for the remaining ICWs.

The definitions of the bits in ICWI are:

Bits 7-5 — Not used in an 8086/8088 system, only in an 8080 or 8085 system.

Bit 4 — Always set to 1. It directs the received byte to ICWI as opposed to OCW2 or OCW3.

Which also use the even address (A0 = 0).

Bit 3 (LTIM) — Determines whether the edge-triggered mode (LTIM = 0) or the level-triggered mode (LTIM = 1) is to be used. The edge-triggered mode causes the IRR bit to be cleared when the corresponding ISR bit is set.

Bit 2 (ADD — Not used in an 8086/8088 system, only in an 8080 or 8085 system.

Bit 1 (SNGL) — Indicates whether or not the 8259A is cascaded with other 8259As. SNGL = 1 when only one 8259A is in the interrupt system.

Bit 0 (IC4)

— Is set to 1 if an ICW4 is	to be output to during the initialization sequence. For an 8086/8088 system this bit must always be set to 1 because bit 0 in JCW4 must be set to 1.

Bits 7-3 of ICW2 are tilled from bits 7-3 of the second byte output by the CPU during the initialization of the 8259A, and bits 2-0 are set according to the level of the interrupt request, e.g., a request on IR6 would cause them to be set to 110. ICW3 is significant only in systems including more than one 8259A and is output to only if SNGL = 0. ICW4 is output to only if IC4 (ICWI) is set to 1; otherwise, the contents of ICW4 is cleared. The bits in ICW4 are defined as follows:

Bits 7-5	—	Always set to 0.
Bit 4 (SFNM)	—	If set to 1, the special fully nested mode is used. This mode is utilized in systems having more than one 8259A and is discussed below.
Bit 3 (BUF)	—	BUF = 1 indicates that the SP/EN is to be used as an output to disable the system's 8286 transceivers while the CPU inputs data from the 8259A. If no transceivers are present, BUF should be set to 0 and, in systems involving only one 8259A, a 1 should be applied to the SP/EN pin.
Bit 2 (M/S)	—	This bit is ignored if BUF = 0. For a system having only one 8259A, this bit should be 1; otherwise, it should be 1 for the master and 0 for the slaves.
Bit 1 (AEOI)	—	If AEOI = 1, then the ISR bit that caused the interrupt is cleared at the end of the second INTA pulse.
Bit 0 (µPM)	—	µPM = 1 indicates the 8259A is in an 8086/8088 system. This bit being 0 implies an 8080 or 8085 system. A typical program sequence for setting the contents of the

ICWs, which assumes that the even address of the 8259A is 0080, is:

```
MOV     AL,13H

OUT     80H,AL

MOV     AL,18H
```

```
OUT     81H,AL
MOV     AL,ODH
OUT     81H,AL
```

The first two instructions cause the requests to be edge triggered, denote that only one 8259A is used, and inform the 8259A that an ICW4 will be output. The next two instructions cause the 5 most significant bits of the interrupt type to be set to 00011. ICWS is not output to because SNGL = 1; therefore, the last two instructions set ICW4 to OD, which informs the 8259A that the special fully nested mode is not to be used, the SP/EN is used to disable transceivers, the 8259A is a master, EOI commands must be used to clear the ISR bit, and the 8259A is part of an 8086/8088 system.

There are three OCWs. The command word OCWI is used for masking interrupt requests; when the mask bit corresponding to an interrupt request is 1, then the request is blocked. OCW2 and OCWS are for controlling the mode of the 8259A and receiving EOI commands. A byte is output to OCWI by using the odd address associated with the 8259A and bytes are output to OCW2 and OCWS by using the even addresses. OCW2 is distinguished from OCWS by the contents of bit S of the data byte. If bit S is 0, the byte is put in OCW2, and if it is 1, it is put in OCWS. Both OCW2 and OCWS are distinguished from ICWI, which also uses the even address, by the contents of bit 4 of the data. If bit 4 is 0, then the byte is put on OCW2 or OCWS according to bit S. There is no ambiguity in ICW2, ICWS, ICW4, and OCWI all using the odd address because the initialization words must always follow ICWI as dictated by the initialization sequence, and an output to OCWI cannot occur in the middle of this sequence.

Referring back to our earlier discussion, bits L2-LO of OCW2 are for designating an IR level, bit 5 is for giving EOI commands, and bits 6 and 7 are for controlling the IR levels. Recall that when the AEOI bit in ICW4 is 1, the ISR bit, which is set by the interrupt request, is reset automatically at the end of the second INTA pulse, but if AEOI = 0, then the ISR bit must be explicitly cleared by an EOI command, which consists of sending an OCW2 with bit 5 equal to 1. When an EOI command is given the meanings of the four possible combinations of bit 7, the R (rotate) bit, and bit 6, the SL (set level) bit, are:

R	SL	
0	0	Nonspecific, normal priority mode
0	1	Specifically clears the ISR bit indicated by L2-LO
1	0	Rotate priority so that a device after being serviced has the lowest priority
1	1	Rotate priority until position specified by L2-LO is lowest

The bits in OCW2 are only temporarily retained by the 8259A until the actions specified by them are carried out. This statement is particularly important with regard to the EOI bit. Let us now examine these possibilities in greater detail beginning with the normal priority mode (00).

Ordinarily, a request on IRO has the highest priority, one on IRI has the next highest priority, and so on. When the first INTA pulse arrives the priority resolver allows only the unmasked request having the highest priority to set its ISR bit. Because the 3 least significant bits of ICW2, where ICW2 indicates the interrupt type and determines the interrupt pointer address, are determined by which ISR bit is set, the address of the interrupt routine depends on which ISR bit is set. Therefore, the interrupt routine associated with the device connected to the highest-priority IR pin is begun first and the other requests must wait until further interrupts are allowed.

Under the normal priority mode, if ISRn is set, the priority resolver will not recognize any requests on IR7 through IR(n+l), but will recognize unmasked requests on IR(n -1) through IRO, Consequently, if the IF flag in the CPU has been reset to 1, requests having higher priority than the one being processed may cause the current interrupt routine to be interrupted while those having lower priorities are kept waiting. The lower-priority requests are processed according to their priorities as the higher-priority ISR bits are cleared. If AEOI = 1, the ISR bit corresponding to an interrupt is automatically cleared at the end of the second INTA pulse. When AEOI = 0 the ISR must be cleared by the interrupt routine by setting bit 5 of OCW2.

As an example of the normal priority mode, suppose that initially AEOI = 0 and all ISR and IMR bits are clear. Also suppose that, as shown in Fig. 4.15, requests occur simultaneously on IR2 and IR4, then a request arrives at IRI, and last a request arrives at IR3 and that these are the only requests. First, ISR2 will be set and the interrupt routine associated with IR2 will start executing. After this routine resets the IF flag to 1 and when the IRI request is made, ISRI will be set and the IRI routine will be executed in its entirety. While it is executing it should reset the IF bit to 1 and send the necessary command to clear ISRI. Upon the return to the IR2 routine, ISR2 is cleared. Then ISR4 is set and its routine is begun. While this routine is executing IR3is made. It is acknowledged as soon as IF is reset to 1, and ISR3 is set. Then the IR3 routine is initiated. Before the IR3 routine is completed it should clear ISR3 and set IF. The return is made to the IR4 routine, which should clear ISR4 before returning to the IR2 routine. The IR2 routine, having already cleared the ISR2 bit, would simply return to the interrupted program. (Note that if IF is not reset to 1 within the interrupt routine, further interrupts will not be processed until the routine is completed, i.e., the IRET instruction is encountered.)

Although a 1 sent to bit 5 of OCW2 normally causes the highest-priority ISR bit (i.e., the last ISR bit to be set) to be cleared, any ISR bit can be explicitly cleared by sending an OCW2 with the R, SL, and EOI bits set to Oil and putting the number of the bit to be cleared in L2-L0. If

0 1 1 0 0 0 1 1

is sent to OCW2, then ISR3 will be cleared.

In addition to the normal priority mode discussed above, OCW2 can rotate the priority by assigning bottom priority to any one of the IR levels. In this case the other priorities will follow as if the normal ordering had been rotated. For instance, if the lowest priority is given to IR4, then the order of priorities will be:

IR5, IR6, IR7, IR0, IR1, IR2, IR3, IR4

(i.e., IR5 is rotated into the top-priority position). A rotation by one can be obtained by letting the combination for the R and SL bits be 10. If the R and SL bit combination is II, then the IR level with the lowest priority is the one specified by L2-LO. If IR5 currently has top priority and

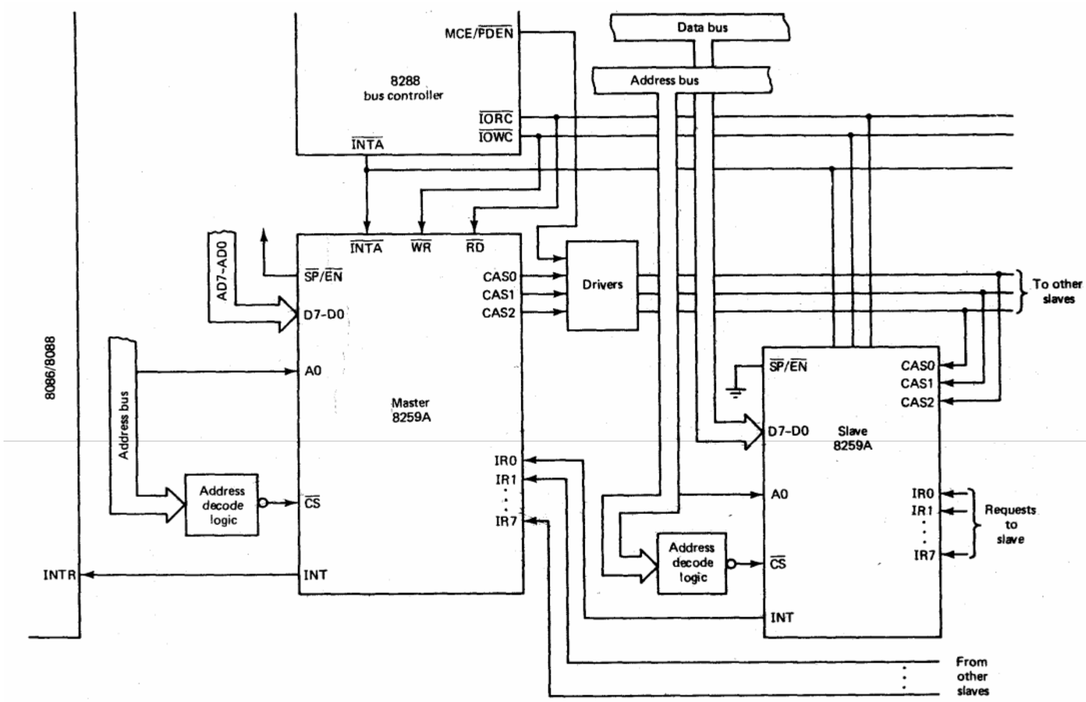

Figure 4.15: Actions taken in the normal operating mode when a typical sequence of interrupts occurs.

1 0 1 0 0 0 0 0

is sent to OCW2, then the new priority ordering would be

IR6, IR7, IR0, IR1, IR2, IR3, IR4, IR5

1 1 1 0 0 0 1 0

but if

is sent, the new ordering would be

IR3, IR4, IR5, IR6, IR7, IR0, IR1, IR2

The R and SL bits may also have significance when EOI = 0. In this case R = 1 and SL = 0 cause automatic rotations when AEOI = 1, and R = SL = 0 turns off this action so that automatic rotations do not take place. R = SL = 1 and EOI = 0 result in the lowest priority being designated by L2-LO without an EOI command being sent. The remaining combination, R = 0 and SL = 1, causes no action.

In OCW3 the ESMM (enable special mask mode) and SMM (special mask mode) bits can be used to negate the priority modes discussed above. If a byte is sent to OCW3 in which both ESMM and SMM are set to 1, then unmasked interrupt requests are processed as they arrive (provided that the processor's IF bit is 1) and the priority order is ignored. By subsequently sending a byte to OCW3 in which ESMM = 1 and SMM = 0, a switch back to the priority ordering of interrupts can be made. If a byte with the ESMM bit equal to 0 is sent to OCW3, then the SMM bit will have no effect and the special mask mode will not change.

The P (polling) bit is used to place the 8259A in polling mode. This mode assumes that the CPU is not accepting interrupts (IF = 0) and it is necessary for the interrupt requests in the IRR to be polled. When the P bit is 1 the next RD signal would cause" the appropriate bit in the ISR to be set just as if INTA signal had been received, and would return to the AL register in the CPU a byte of the form

I	—	—	—	—	W2	W1	W0

where I = 1 indicates that an interrupt is present and W2, Wl, and W0 give the IR level of the highest-priority interrupt. For example, if P = 1, the priority or daring is

IR3, IR4, IR5, IR6, IR7, IR0, IR1, IR2

there are unmasked interrupts on IR4 and IRI, and the instruction

```
IN    AL, 80H
```

(where 0080 is the even address of the 8259A) is executed, then

1	—	—	—	—	1	0	0

is input to the AL register.

When P = 0, the contents of IRR or ISR can be read into the AL by setting RR = 1 and executing the instruction

IN AL, 80H

If at the time the IN instruction is executed RIS = 0, then IRR is input; otherwise, ISR is read. The contents of IMR can always be read by using the of 8259A, e.g., for the address assignment indicated above,

IN AL, 81H

would input the contents of IMR to AL.

Because the OCW3 bits (except for ESMM) are used to specify whether or not the 8259A is in special mask mode and which information is to be put on the data bus during a read, these bits are retained until they are reset by the next output to OCW3. For example, if P = 0, RR = 1, and RIS= 0, any read from the even address of the 8259A before a new byte is sent to OCW3 will cause IRR to be read.

As a final note regarding the internal workings of the 8259A, let us examine what happens when noise interferes with a request. An IR input must remain high until after the trailing edge of the first INTA pulse. If it does not, then the 8259A will simulate a 1 on IR7. Therefore, provided that no device is connected to IR7. Requests on this line would indicate improper drops in signals on the other request lines and the interrupt routine associated with IR7 would serve as a noise "cleanup" routine. If a device is connected to IR7, this noise detection scheme could still be used because an IR7 request from a device will set ISR7, while a noise-related request on IR7 will not affect ISR7. Thus, the IR7 routine could distinguish between the two events by reading the ISR and testing bit 7.

Interrupt System Based on Multiple 8259As

A multiple 8259A interrupt system is diagrammed in Fig. 4.16. In this figure data bus drivers are not shown, but they could be inserted. Although the SP/EN pin on the master 8259A is connected to the data bus transceivers, a 0 is applied to the SP/EN pins of the slaves. Only one slave is shown, but up to seven more slaves could be similarly connected into the system, permitting up to 64 distinct interrupt request lines. When designing the address decoder logic, each 8259A must be given its own address pair in the I/O address space. The drivers inserted in the CAS2-CASO lines may or may not be needed, depending on the proximity of the master to the slaves.

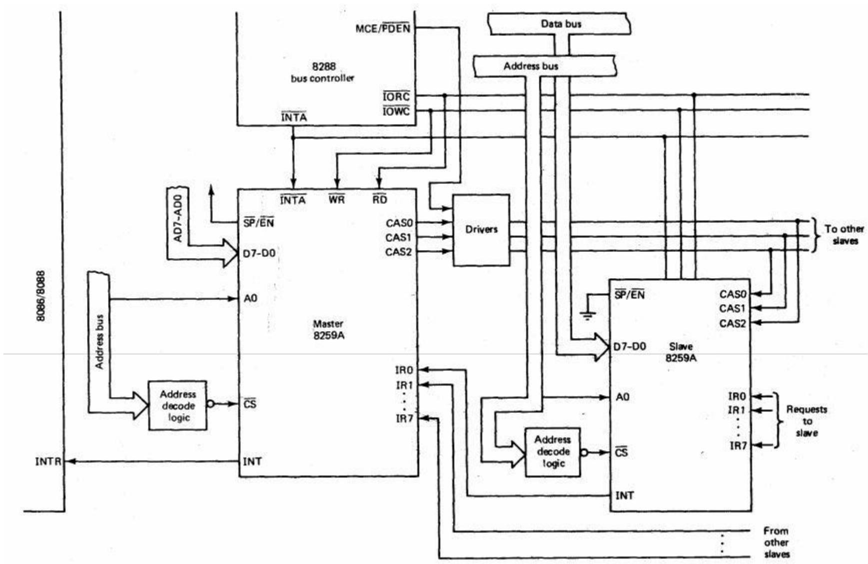

Figure 4.16

In a multiple 8259A system the slaves must be initialized as well as the master. The master would be initialized in the same way as indicated above except that SNGL would be set to 0 and ICW3 would need to be filled. A 1 would be put in each ICW3 bit for which the corresponding IR bit is connected to a slave and 0s would be put in the remaining bits. The SFNM bit may be set to 1 to activate the special fully nested mode. The SNGL bit should also be set to 0 when initializing the slaves. Thus, an ICW3 will be required for each slave, but for a slave ICW3 has a different meaning. For a slave, ICW3 has the form

0	0	0	0	0	ID2	ID1	ID0

where the 3 least significant bits provide the slave with an identification number. The identification number given to the slave should be the same as the number of the master request line to which its INT pin is connected.

When a slave puts a 1 on its INT pin this signal is sent to the appropriate IR pin on the master. Assuming that the IMR and priority resolver do not block this signal, it is sent to the CPU through the INT pin on the master. When the CPU returns the INTA signal the master will not only set the appropriate ISR bit and clear the corresponding IRR bit, it will also check the corresponding bit in ICW3 to determine whether or not the interrupt came from a slave. If so, the master will place the number of the IR level on the CAS2-CASO lines; if not, it will put the contents of ICW2on the data bus and no signals will be applied to the CAS2- CASO lines. The INTA signal is also received by all of the slaves, but only that slave whose ID matches the number sent to it by the master over the CAS2-CASO lines will accept the signal. In the selected slave the appropriate ISR bit will be set, the corresponding IRR bit will be cleared, and its ICW2 will be put on the data bus. Because ICW2 contains the interrupt type, it is important that unique combinations be put in the master and slave ICW2s during the initialization process. EOI commands are required for both the master and the slave if their AEOI bits are 0.

Except for the response to the INTA signal discussed above, the actions taken by all 8259A devices in the system are the same. Also, their modes are controlled and their registers are read in the same way. There is one exception, however. If the SFNM bit in the ICW4 of the master is initialized to 1, the master will enter the special fully nested mode. This mode is ordinarily used with the normal priority mode and AEOI = 0. In this case, the master will allow unmasked requests of sufficient priority to be passed on to the INT pin even if the corresponding ISR bit is already 1. This means that if a higher-priority request arrives at a slave while one or more of the slave's requests is being processed, the new request will be allowed to send its INT signal through the master. When using the special fully nested mode, two EOI commands may need to be sent by the interrupt routine. First, a nonspecific EOI command would be sent to the slave that caused the interrupt and then the slave's ISR would be tested. If and only if the ISR contains all 0s would a nonspecific EOI command be sent to the master. Therefore, assuming that slave 1 is connected to IRI and slave 2 is connected to IR2 of a master, they are the only slaves, and the highest priority in all three 8259As is assigned to IRQ, the fully nested order of priorities would be:

Highest priority
Master: IR0
Slave 1: IR0, IR1, IR2, IR3, IR4, IR5, IR6, IR7
Slave 2: IR0, IR1, IR2, IR3, IR4, IR5, IR6, IR7
Master: IR3, IR4, IR5, IR6, IR7
Lowest priority

The masks in the master and slaves may, of course, be used to block out some of the requests.

PC Bus and Interrupt System

The PC Bus uses a bus controller, address latches, and data transceivers (bidirectional data buffers).

- Bus controller:(Intel 8288 Bus Controller) coordinates activities on bus. It converts CPU status and clock signal into bus control signals. These control signals direct operations of latches, data transceivers, and the I/O bus

- Address latches: these are buffers for address lines. They serve two purposes, fill the speed gap between the CPU and other devices; and allow the CPU pins to be used for other purposes.

- Data transceivers: bidirectional data buffers

- Interrupt processing: It follows the steps:

- When external device requests an interrupt, the CPU initiates a special sequence of bus cycles, called *interrupt-acknowledge sequence*

- The external device recognizes the interrupt-acknowledge sequence by decoding the bus control signals

- Once the external device recognizes the acknowledge, it then places the interrupt vector number on the data bus (through interrupt controller, in the case of IBM PC)

- After the CPU receives the interrupt vector, it begins the standard *interrupt-initiation sequence*:forming the interrupt vector address; then starting execution of the interrupt handler routine.

Intel 8259 interrupt controller: The 8088 processor has only two interrupt control inputs, nonmaskable interrupt (NMI) and interrupt request (INTR). NMI are interrupts that cannot be masked (examples: memory parity error, power loss). The interrupt controller has several functions:

- it receives interrupt requests from up to eight different sources.
- it prioritizes and queues interrupts that come at the same time
- it masks interrupt requests when instructed by the CPU
- it processes the CPU's interrupt-acknowledge signal by sending the interrupt vector number to the CPU.