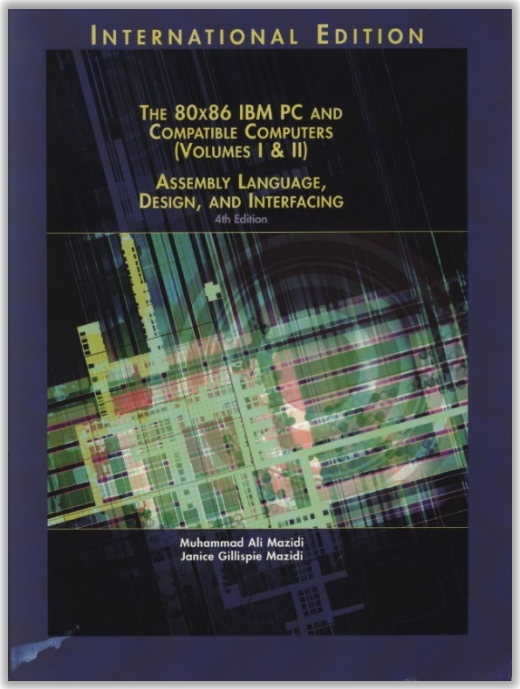
Lecture 09: Interrupts & 8259

Reference Book:

The 80x86 IBM PC and Compatible Computers

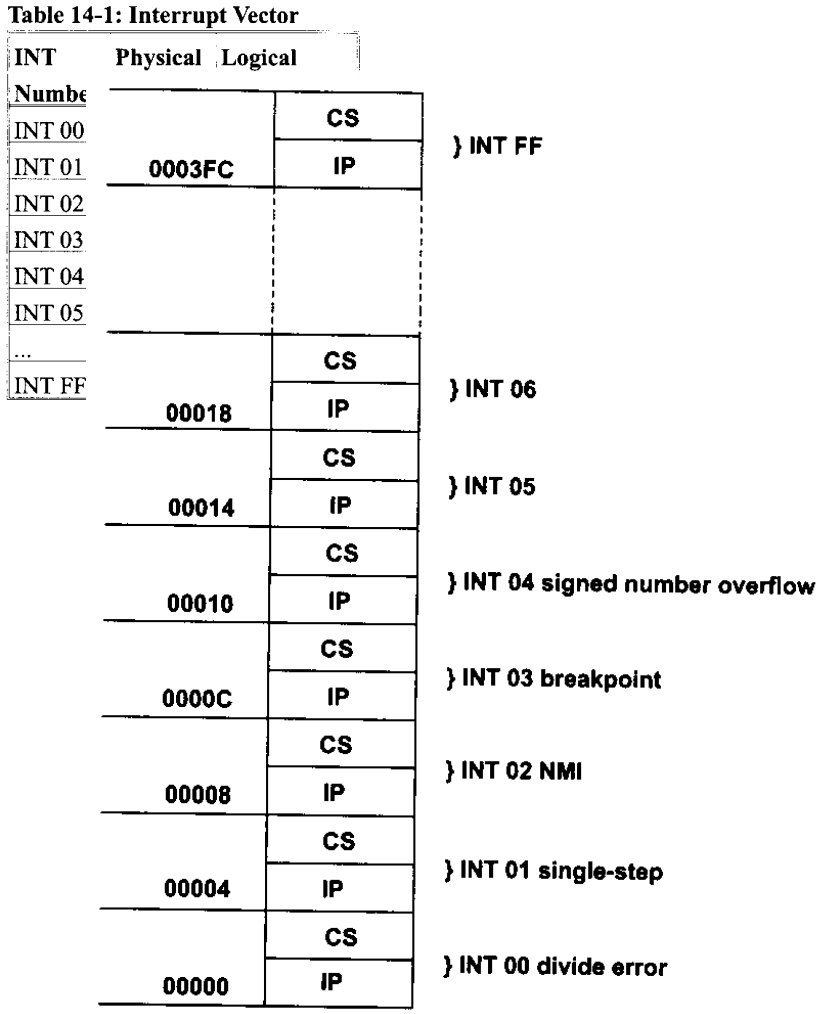
Chapter 14

Interrupts and the 8259 Chip

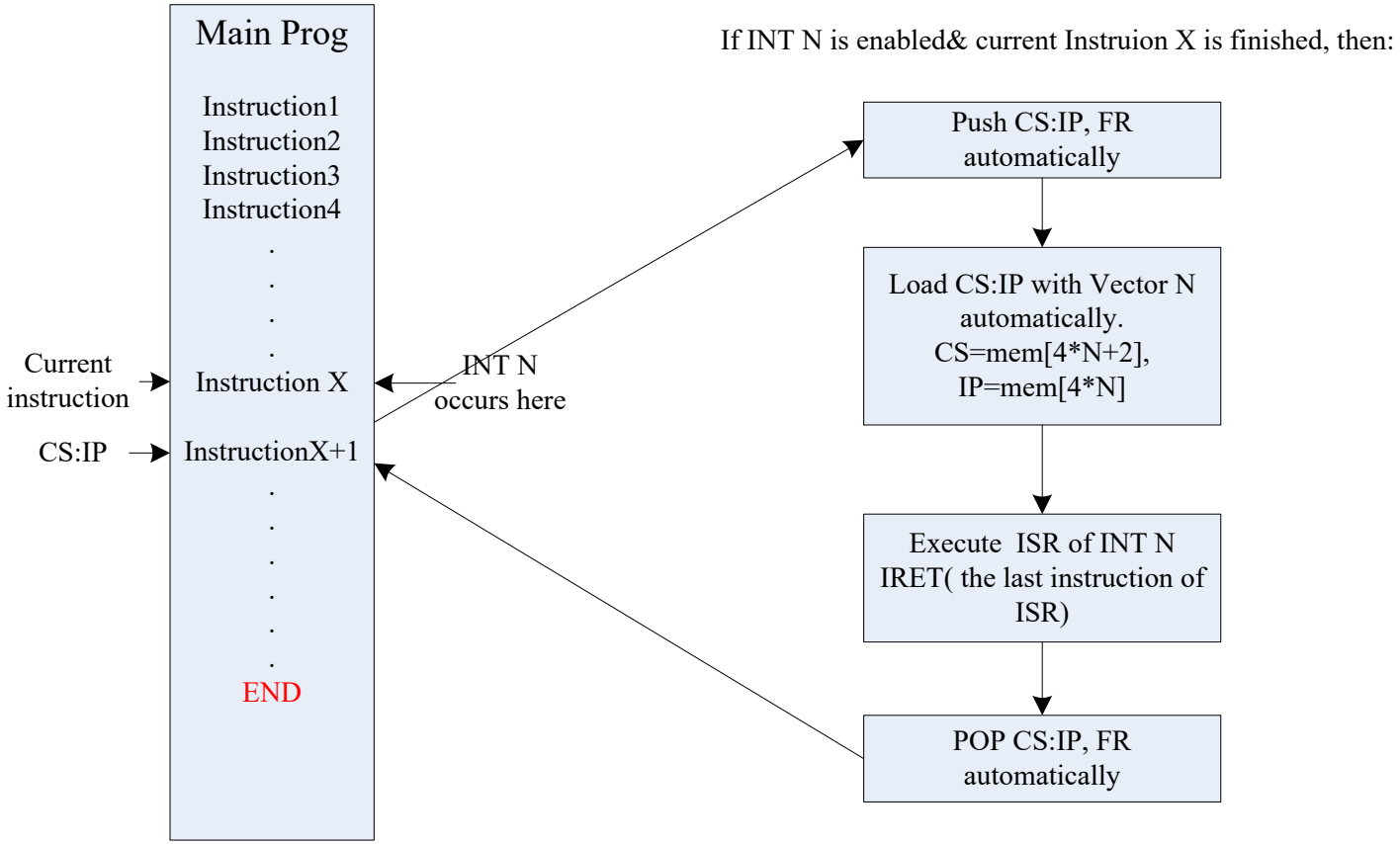


Interrupts in 8086/8088

- 256 **interrupt types** in total
  - INT 00 ~ INT 0FFh
- Type * 4 = PA of **interrupt vector**
  - The first 1KB is used to store interrupt vectors, called **Interrupt Vector Table** (IVT)
- Interrupt vector points the entrance address of the corresponding **interrupt service routine (ISR)**



Main Program and ISR



An ISR is launched by an interrupt event (internal 'int xx' **or** external NMI and INTR) . So ISR is 'separated' from main.

Categories of Interrupts

- Hardware (external) interrupts
  - Maskable (from INTR)
  - Non-maskable (from NMI)
- Software (internal) interrupts
  - Using the **INT** instruction
  - Predefined *conditional* (*exception*) interrupts

Hardware Interrupts

- Non-maskable interrupt
  - Trigger: NMI pin, input-signal, rising edge and two-cycle high activate
  - TYPE: INT 02
  - Not affected by the **IF**
  - Reasons:
    - E.g., RAM parity check error, interrupt request from co-CPU 8087

Hardware Interrupts

- Maskable interrupt
  - Trigger: INTR pin, input-signal, high active
  - TYPE: No predefined type
  - IF = 1, enable; IF = 0, disable
    - **STI** sets IF, **CLI** clears IF
  - Reasons:
    - Interrupt requests of external I/O devices

Procedure for Processing Maskable Interrupts

- CPU responds to INTR interrupt requests
  - External I/O devices send interrupt requests to CPU
  - CPU will check INTR pin on the last cycle of an instruction: if the INTR is high and IF = 1, CPU responds to the interrupt request
  - CPU sends two negative pluses on the ~INTA pin to the I/O device
  - After receiving the second ~INTA, I/O device sends the interrupt type *N* on the data bus

Procedure for Processing Maskable Interrupts

- CPU executes the ISR of INT *N*
  - CPU reads the *N* from data bus
  - Push the **FR** in stack
  - Clear **IF** and **TF**
  - Push the CS and IP of the next instruction in stack
  - Load the ISR entrance address and moves to the ISR
  - At the end of the ISR, **IRET** will pop IP, CS and FR in turn, CPU returns to previous program and proceeds

Software Interrupts

- INT xx instruction
  - An ISR is called upon instruction such as "INT xx"
    - E.g., int 21h ; Dos service
  - CPU always responds and goes execute the corresponding ISR
    - Not affected by the IF
  - You can "CALL" any ISR by using the INT instruction

Difference between INT & CALL

- CALL FAR can jump anywhere within 1MB vs. INT jumps to a fix location (finding the corresponding ISR)
- CALL FAR is in the sequence of instructions vs. an external interrupt can come in at any time
- CALL FAR cannot be masked (disabled) vs. an external interrupt can be masked
- CALL FAR saves CS:IP of next instruction vs. INT saves FR + CS:IP of next instruction
- last instruction: RETF vs. IRET

Software Interrupts

- Predefined conditional interrupts
  - "INT 00" (divide error)
    - Reason: dividing a number by zero, or quotient is too large
  - "INT 01" (single step)
    - If TF = 1, CPU will generates an INT 1 interrupt after executing each instruction for debugging
  - "INT 03" (breakpoint)
    - When CPU hits the breakpoint set in the program, CPU generates INT 3 interrupt for debugging
  - "INT 04" (signed number overflow)
    - **INTO** instruction
    - Check the OF after an arithmetic instruction

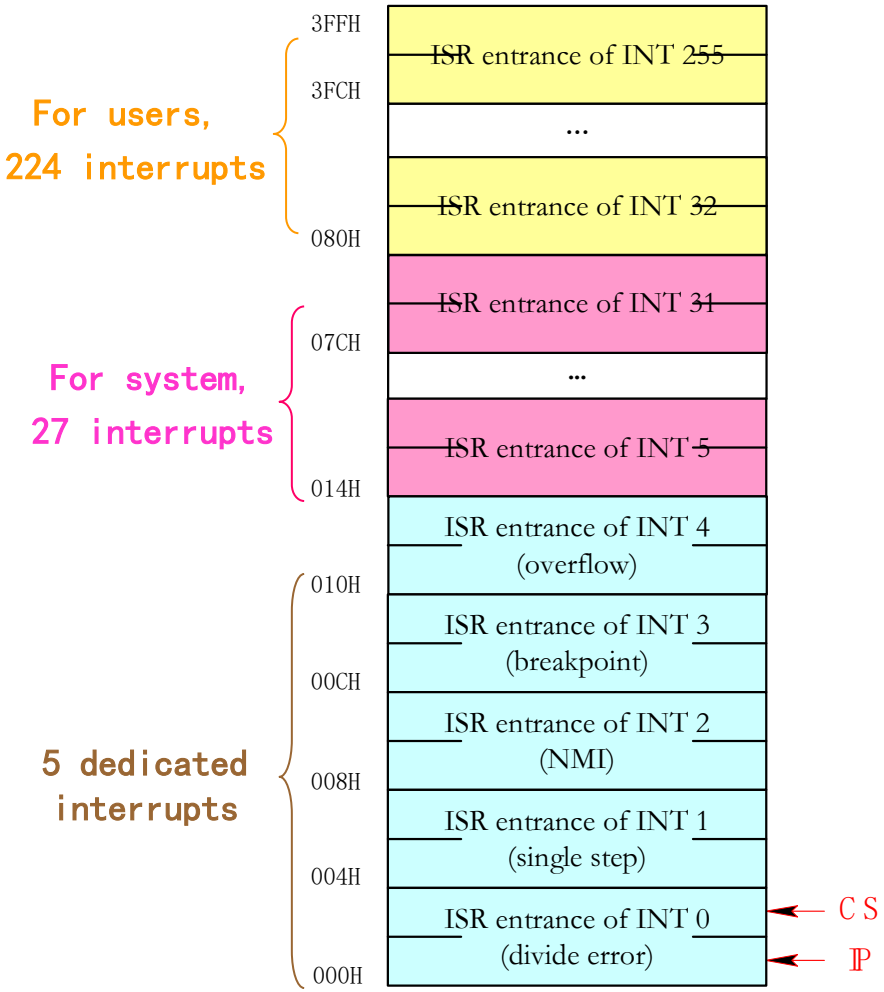
```
;How to clear TF?
    PUSHF
    POP AX
    AND AX 0FEFFH
    PUSH AX
    POPF
```

```
MOV  AX,0009H
ADD AX,0080H
INTO
```

Procedure for Processing Non-Maskable & Software Interrupts

- For NMI
  - CPU checks NMI, generates INT 02 interrupt automatically regardless of **IF** and executes to the ISR
- For software (internal) interrupts
  - CPU generates INT *N* interrupt automatically and executes to the corresponding ISR

Interrupt Vector Table of 8086/8088



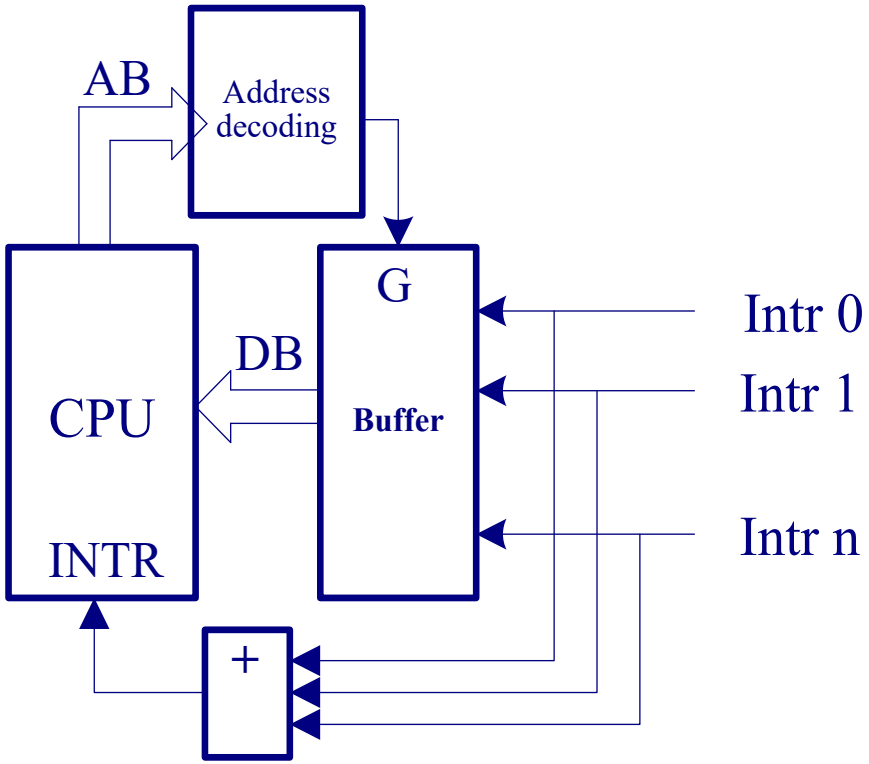
- 256 interrupts
  - ❖ 0 ~ 4 dedicated
  - ❖ 5 ~ 31reserved for system use
    - □ 08H ~ 0FH: 8259A
    - □ 10H ~ 1FH: BIOS
  - ❖ 32 ~ 255 reserved for users
    - □ 20H ~ 3FH: DOS
    - □ 40H ~ FFH: open

Interrupt Priority

- INT instruction has higher priority than INTR and NMI
- NMI has higher priority than INTR
- For different external interrupt requests, different strategies can be used to determine their priorities.

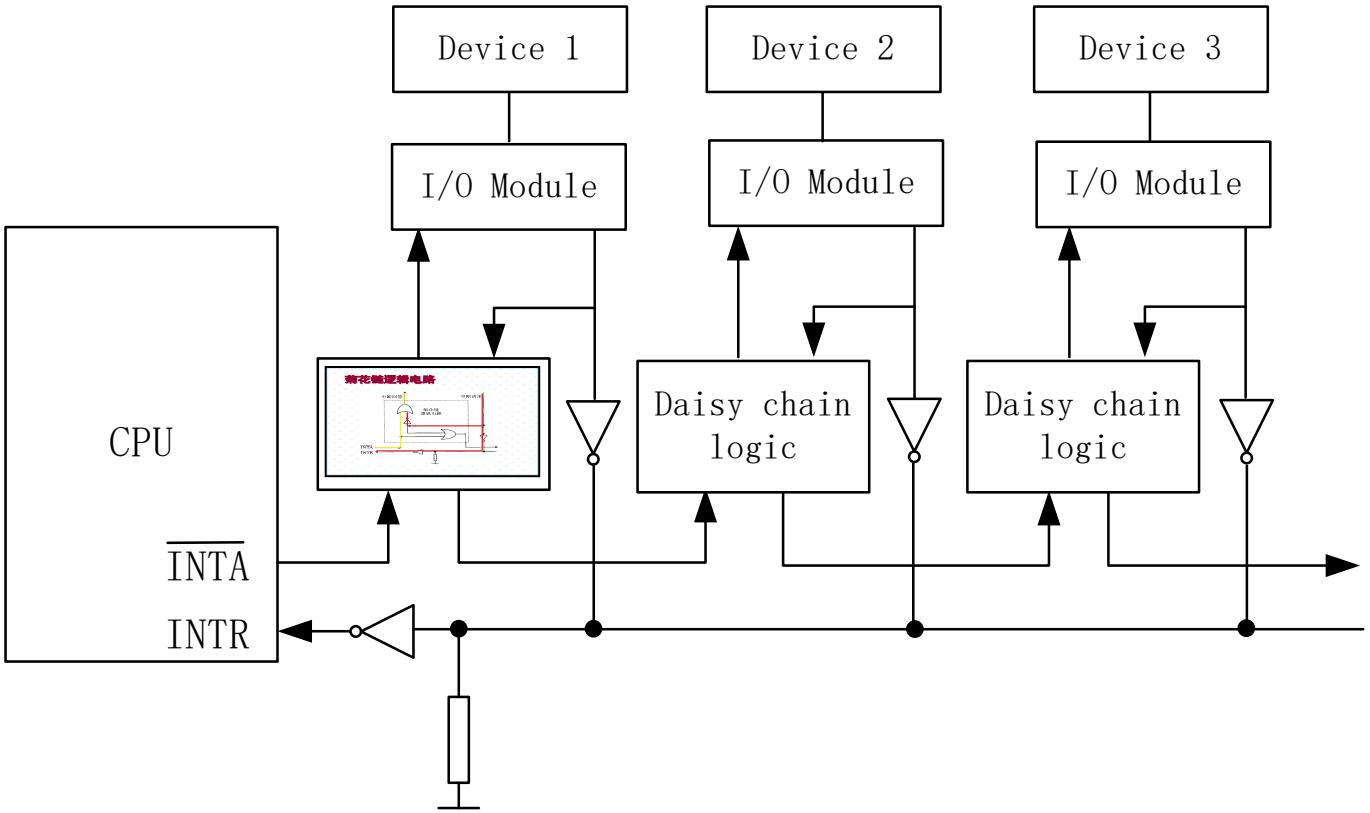
Priority of INTR Interrupts

- Software polling
  - The sequence of checking determines the priority



Priority of INTR Interrupts

- Hardware checking
  - The location in the daisy chain counts



Priority of INTR Interrupts

- Vectored interrupt controller
  - E.g., 8259

8259

- 8259 is Programmable Interrupt Controller (PIC)
- It is a tool for managing the interrupt requests.
- 8259 is a very flexible peripheral controller chip:
  - PIC can deal with up to 64 interrupt inputs
  - interrupts can be masked
  - various priority schemes can also programmed.
- originally (in PC XT) it is available as a separate IC
- Later the functionality of *(two PICs)* is in the motherboards chipset.
- In some of the modern processors, the functionality of the *PIC* is built in.

FIGURE 1 Block diagram and pin definitions for the 8259A Programmable Interrupt Controller (PIC). (Courtesy of Intel Corporation.)

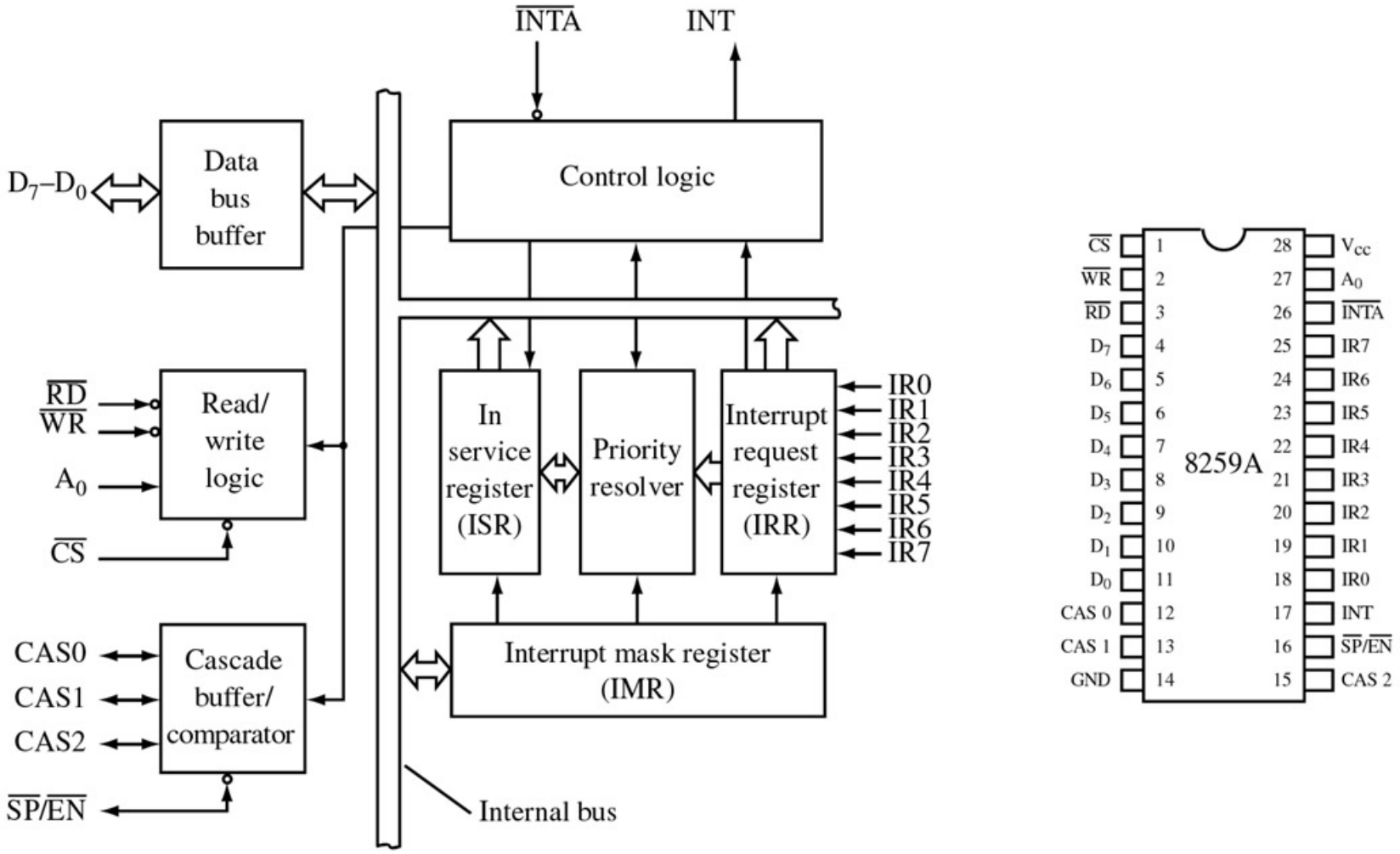
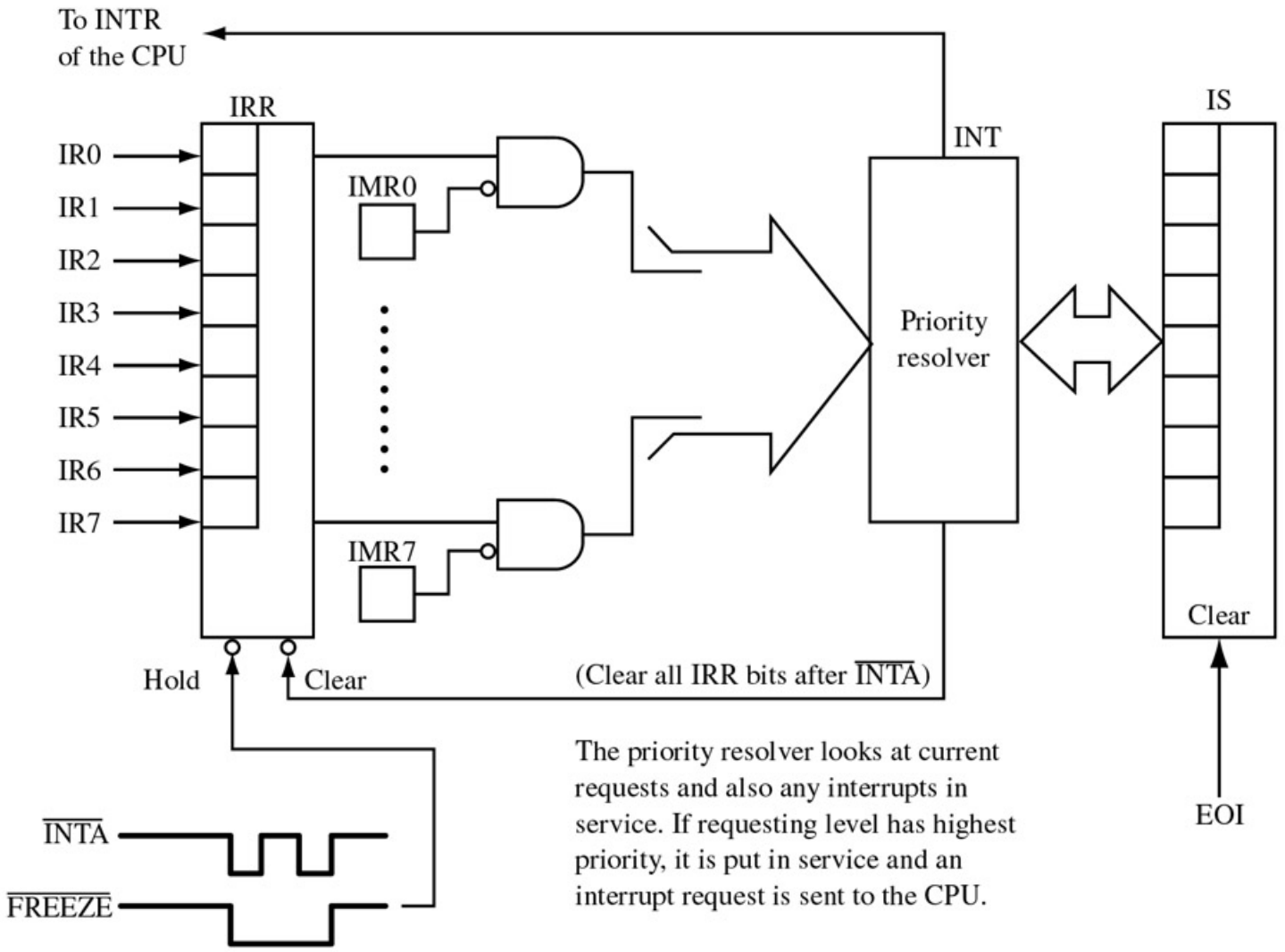
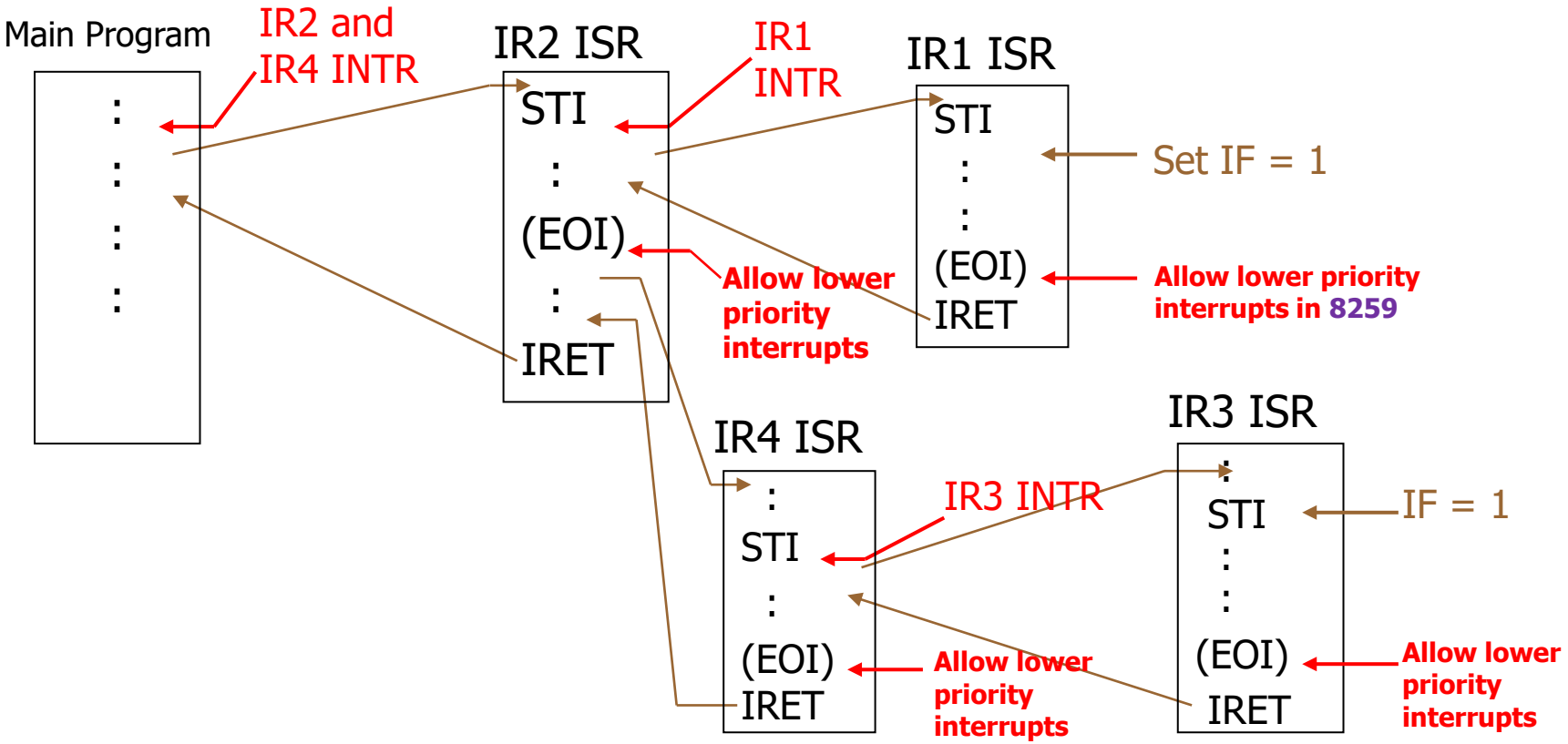


FIGURE 2 All interrupt requests must pass through the PIC's interrupt request register (IRR) and interrupt mask register (IMR). If put in service, the appropriate bit of the in-service (IS) register is set.



Interrupt Nesting

- Higher priority interrupts can interrupt lower interrupts



Exp. 3: Interrupts