Interrupts

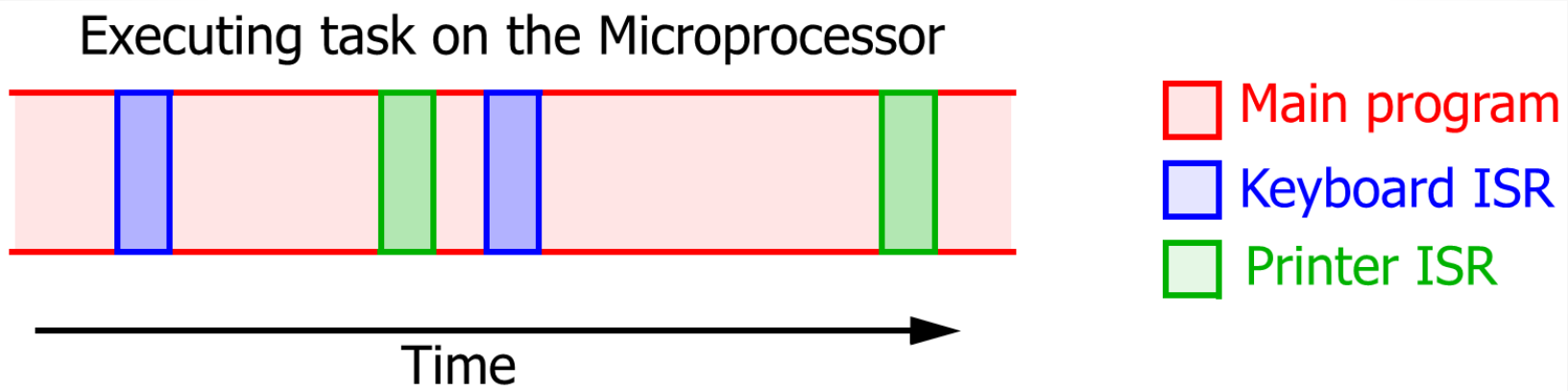
Dr. Mohammed Morsy

Interrupts

- This chapter will discuss:
  - Software Interrupt.
  - Hardware Interrupt.
  - Expanding the Interrupt Structure.
  - 8259A Programmable Interrupt Controller.

Introduction

- Interrupt is useful when interfacing I/O device that provide or require data at relatively low data transfer rates.
- Interrupt processing is an alternative to polling.



Introduction

- The Intel microprocessors support hardware interrupts through:
  - Two pins that allow interrupt requests, INTR and NMI
  - One pin that acknowledges, $\overline{INTA}$, the interrupt requested on INTR.
- And software interrupts through instructions:
  - INT, INTO, INT 3, BOUND
- Control is provided through
  - IF and TF flag bits
  - IRET and IRETD

Software Interrupts

- INT and INT 3 behave in a similar way.

INT n

  - Calls ISR located by vector n at address (n*4).

- BOUND and INTO are both conditional:

BOUND AX, DATA ;Compares AX with DATA

  - AX is compared with DATA and DATA+1, if less than a type 5 interrupt occurs.
  - AX is compared with DATA+2 and DATA+3, if greater than a type 5 interrupt occurs.

INTO

  - Checks the overflow flag (OF). If OF=1, the ISR stored in vector type number 4 is called.

- An IRET instruction returns six bytes from the stack: two for the IP, two for the CS, and two for the flags

Real Mode Interrupts

- After the execution of each instruction, the microprocessor determines whether an interrupt is active by checking, in order:
  - Other instruction executions
  - Single-step
  - NMI
  - Coprocessor segment overrun
  - INTR
  - INT

Real Mode Interrupts

- If one or more of these conditions are present, then:
  - FLAG register contents are pushed onto the stack
  - Both the interrupt (IF) and trap (TF) flags are cleared, which disables the INTR pin and the trap or single-step feature.
  - The CS and IP contents are pushed onto the stack.
  - The interrupt vector contents are fetched and loaded into CS and IP and execution starts the ISR.
  - On IRET, CS, IP and FLAGS are popped.

    IF and TF are set to the state prior to the interrupt.

Hardware Interrupts

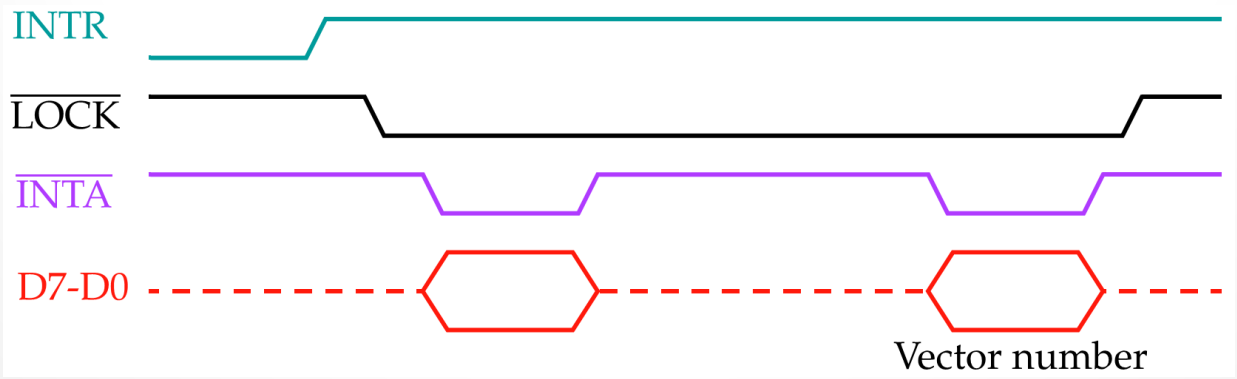
- There are two types of hardware interrupts:
  - Non-maskable interrupt (NMI).
  - Interrupt Request (INTR).
- NMI is decoded internally.
- When NMI input is activated, type 2 interrupt occurs.
- It is an edge-triggered input that requests an interrupt on the positive edge.
- Before the +ve edge, the NMI pin must remain low for at least two clocking periods.
- After the +ve edge, the NMI must remain high until it is recognized by the microprocessor.
- It is often used for parity errors and other major system faults, such as power failures.

Hardware Interrupts

- The INTR pin must be externally decoded to select a vector.
  - Any vector is possible, but the interrupt vectors between 20H and FFH are usually used (Intel reserves vectors between 00H and 1FH).
- $\overline{INTA}$ is an output of the microprocessor to signal the external decoder to place the interrupt number on data bus connections D_7-D_0.
- The INTR pin is set by an external device (8259A) and cleared in the ISR.
  - The input is automatically disabled by the microprocessor once it is recognized and re-enabled by IRET or IRETD instruction.

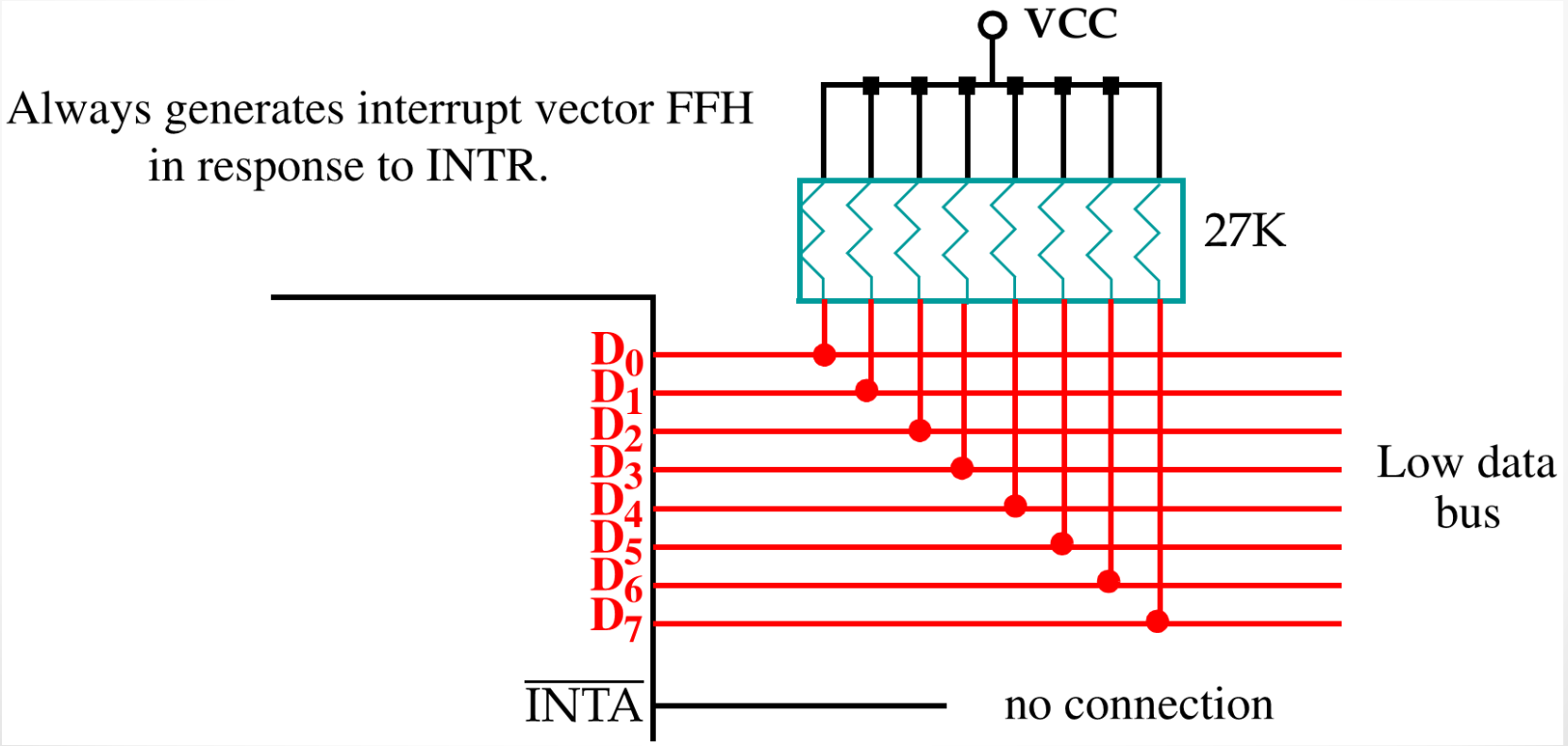
Hardware Interrupts

- The INTR pin must be externally decoded to select a vector.
- The interrupt vectors between 20H and FFH are usually used.
- INTA is an output of the microprocessor to signal the external decoder to place the interrupt number on data bus connections D_7-D_0.
- INTR pin is set by the external device and cleared in the ISR.
- Timing diagram of the handshake:



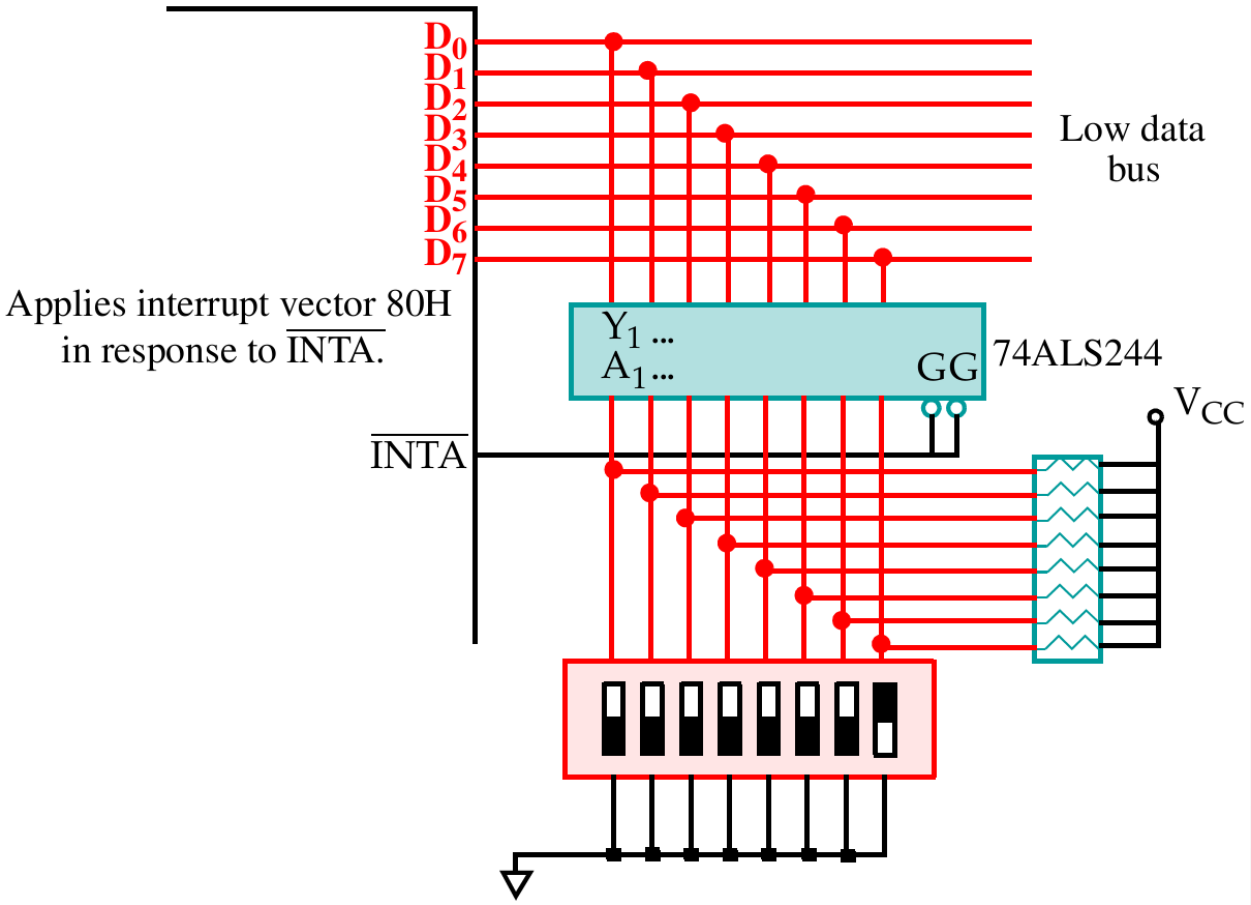
Hardware Interrupts

- The simplest method of generating an interrupt vector:



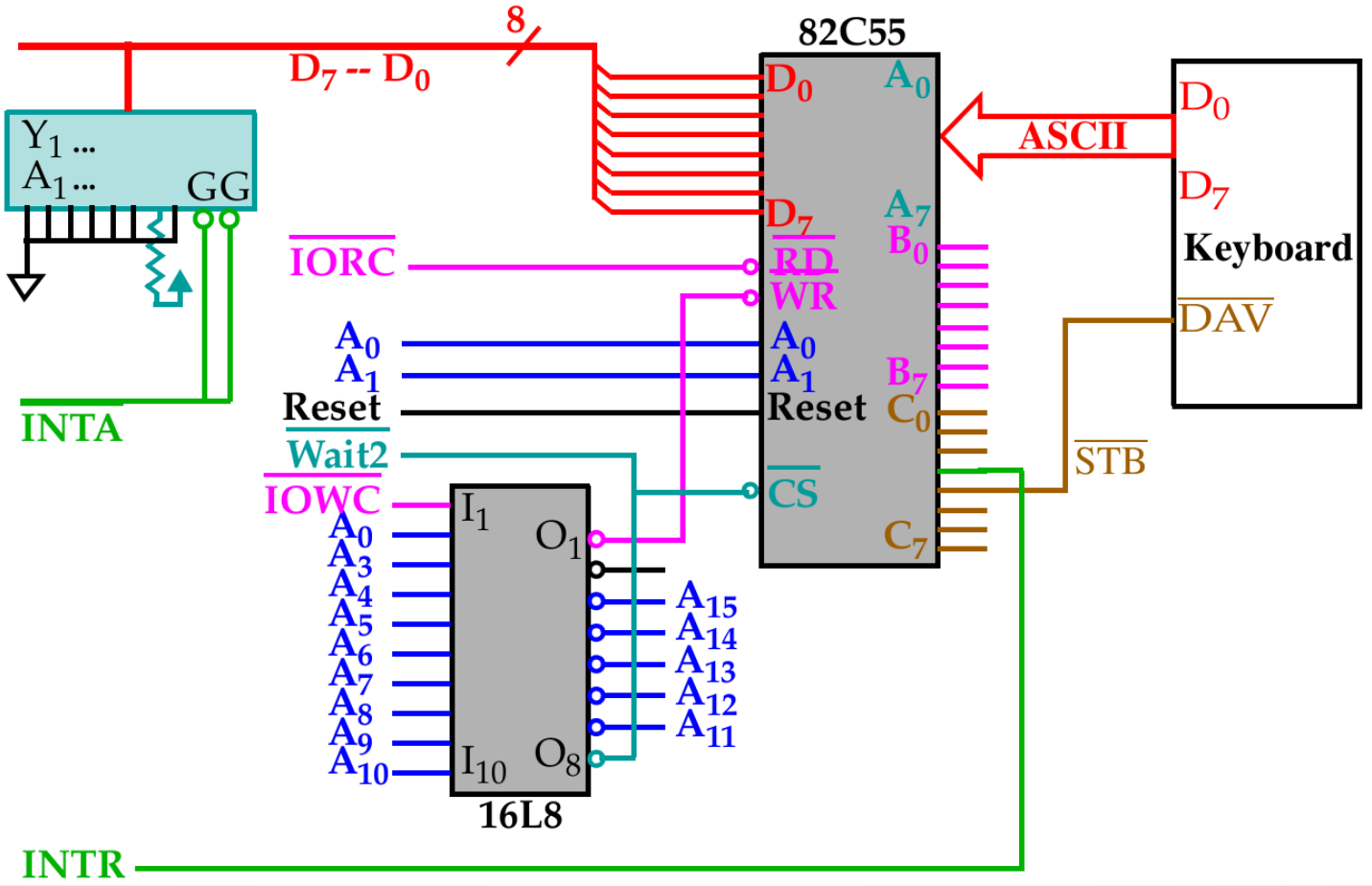
Hardware Interrupts

- Tri-state Buffer for Generating the Interrupt Vector:



Hardware Interrupts

- An Example 82C55 Interrupt Configuration:

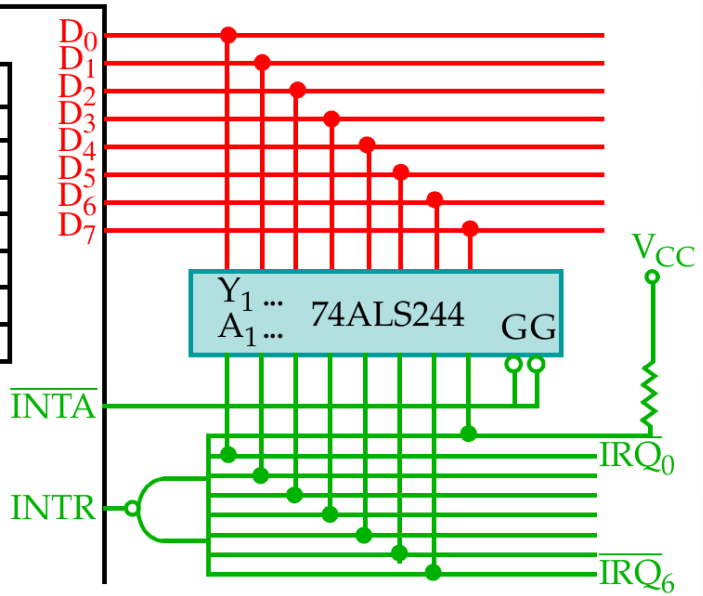


Hardware Interrupts

- Handling more than one IRQ:

IRQs

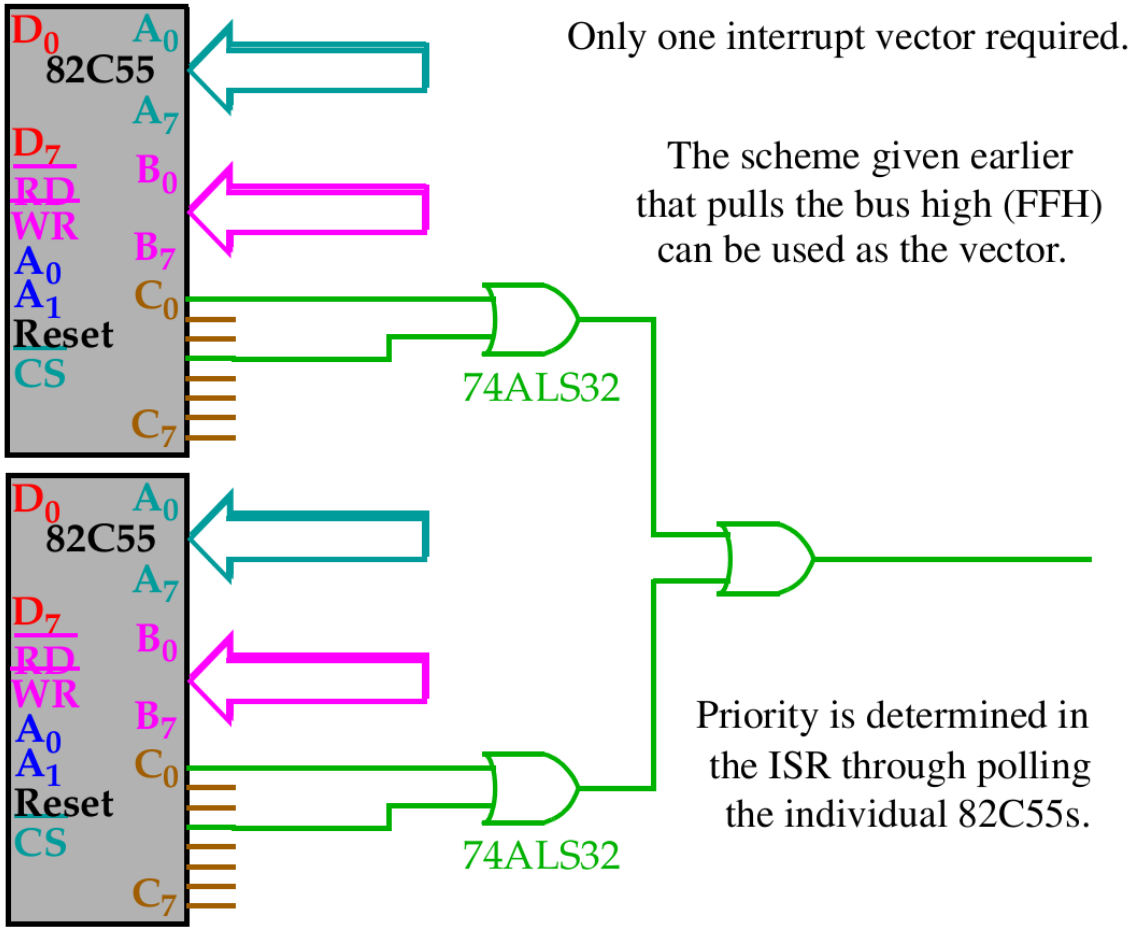
6	5	4	3	2	1	0	Vect
1	1	1	1	1	1	0	FEH
1	1	1	1	1	0	1	FDH
1	1	1	1	0	1	1	FBH
1	1	1	0	1	1	1	F7H
1	1	0	1	1	1	1	EFH
1	0	1	1	1	1	1	DFH
0	1	1	1	1	1	1	BFH



- If any of $\overline{IRQ_X}$ goes low, the NAND goes low requesting an interrupt.
- If more than one IRQ goes low, a unique interrupt vector is generated and an interrupt priority is defined.

Hardware Interrupts

- Daisy-Chained Mechanism for Multiple IRQs:



Hardware Interrupts

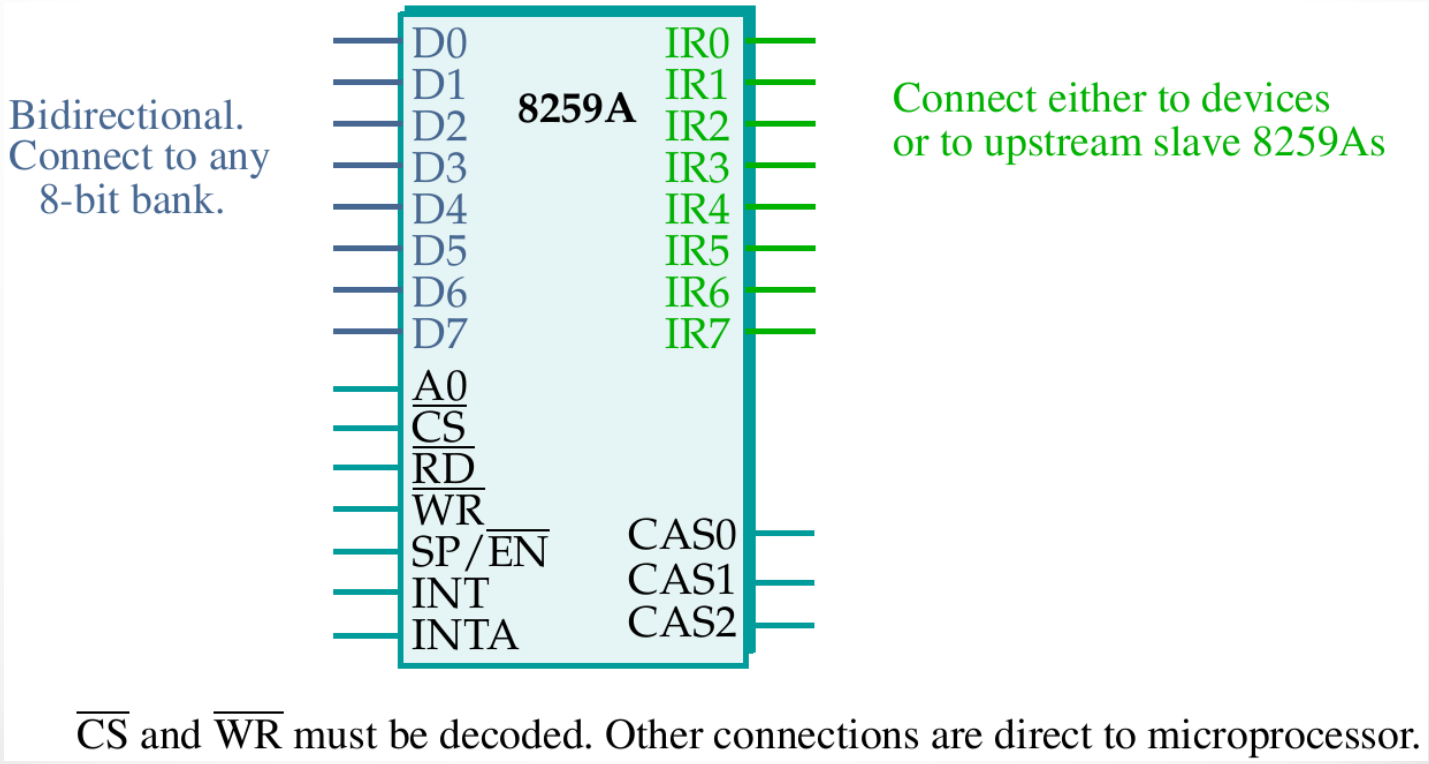
- Setting the daisy-chain priority requires additional software execution time.
- When a daisy-chain is used to request an interrupt, the data bus connections are pulled to high.
- When the INTR goes high, the hardware does not give any indication about the interrupt reason.
- The interrupt service procedure (ISP) polls the 8255s to determine which output caused the interrupt.

Hardware Interrupts

```
;Interrupt service procedure that resolves priority in a daisy-chained interrupt scheme
    CONTROL_1           EQU 03H;first 8255A
    CONTROL_2           EQU 07H;second 8255A
    MASK_1              EQU 01H;INTR B
    MASK_2              EQU 08H;INTR A
    INTERRUPT           PROC NEAR
                        PUSH AX
                        IN AL, CONTROL_1
                        TEST AL, MASK_1
                        JNZ LEVEL_0
                TEST AL, MASK_2
                JNZ LEVEL_1
                        IN AL, CONTROL_2
                        TEST AL, MASK_1
                        JNZ LEVEL_2
                TEST AL, MASK_2
                JNZ LEVEL_4
                        RET
    INTERRUPT           ENDP
```

8259A Programmable Interrupt Controller

- The 8259A adds 8 vectored priority encoded interrupts to the microprocessor.
- It can be expanded to 64 interrupt requests using one master 8259A and 8 slave units

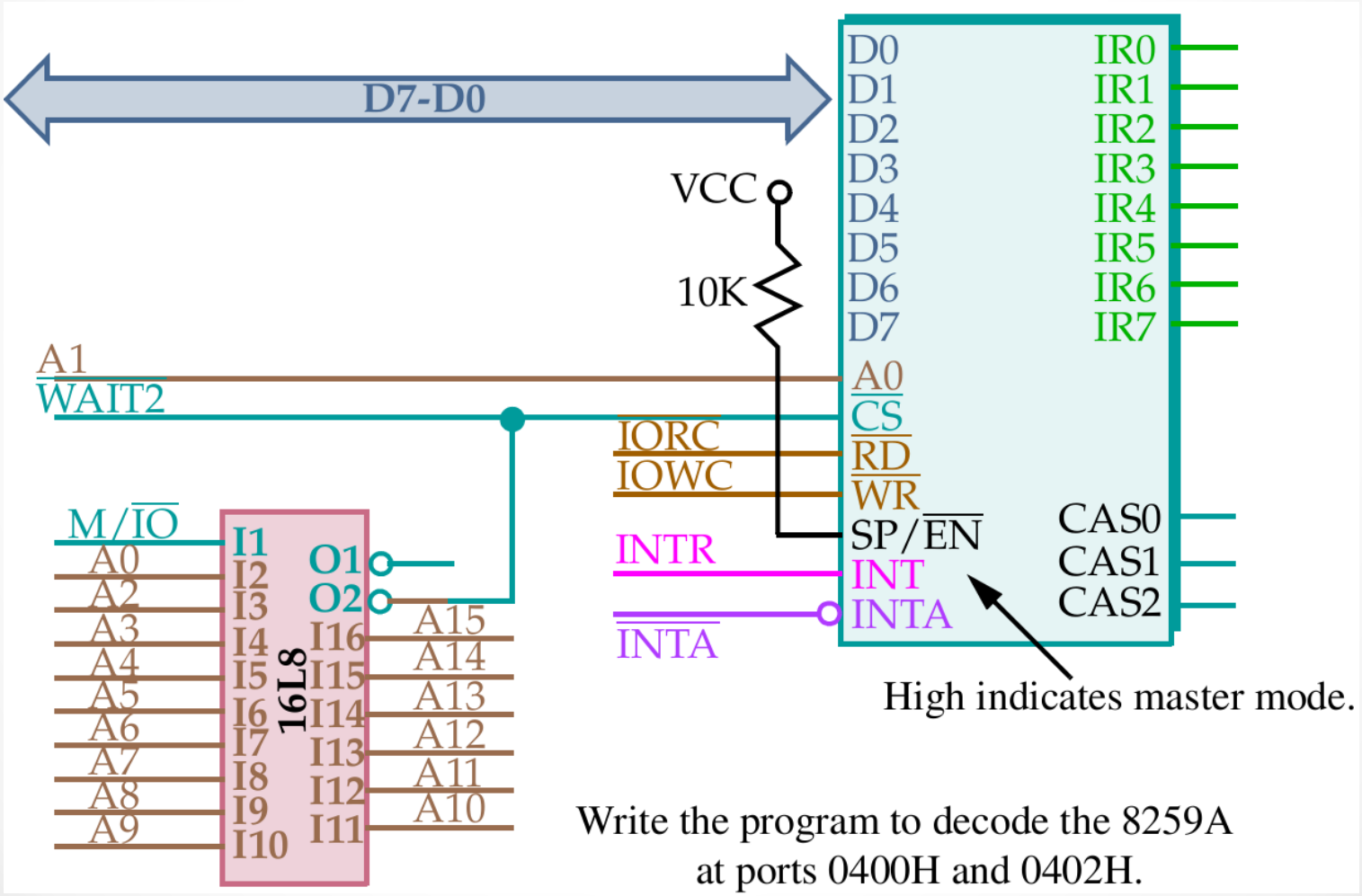


8259A Programmable Interrupt Controller

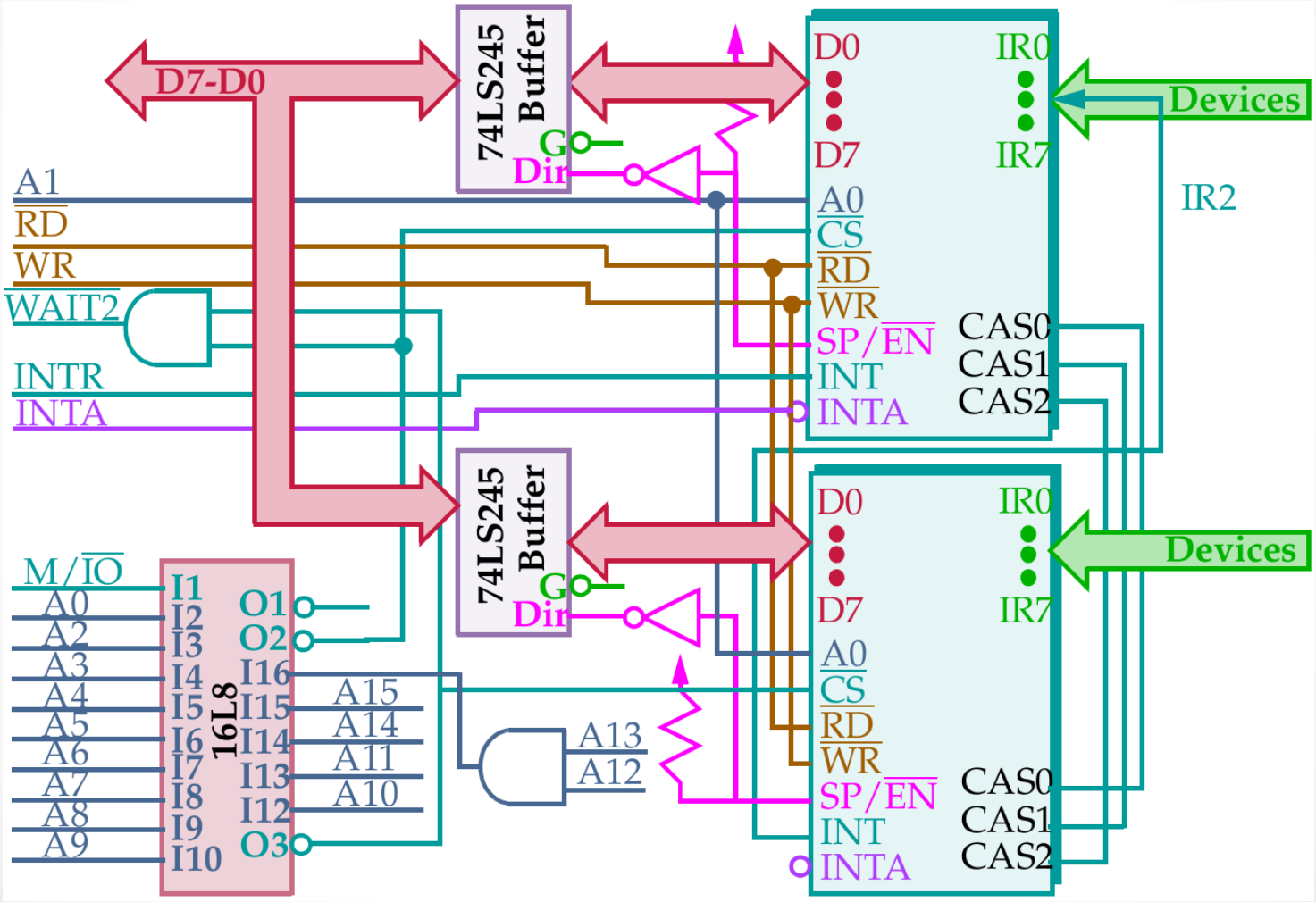
The meaning of the other connections:

- $\overline{WR}$: Connects to a write strobe signal (one of 8 for the Pentium).
- $\overline{RD}$: Connects to the $\overline{IORC}$ signal.
- INT: Connects to the INTR pin on the microprocessor.
- $\overline{INTA}$: Connects to the $\overline{INTA}$ pin on the microprocessor.
- A0: Selects different command words in the 8259A.
- $\overline{CS}$: Chip select - enables the 8259A for programming and control.
- $SP/\overline{EN}$: Slave Program (1 for master, 0 for slave)/Enable Buffer (controls the data bus transceivers when in buffered mode).
- CAS2-CAS0: Used as outputs from the master to the slaves in cascaded systems.

8259A Programmable Interrupt Controller Interface

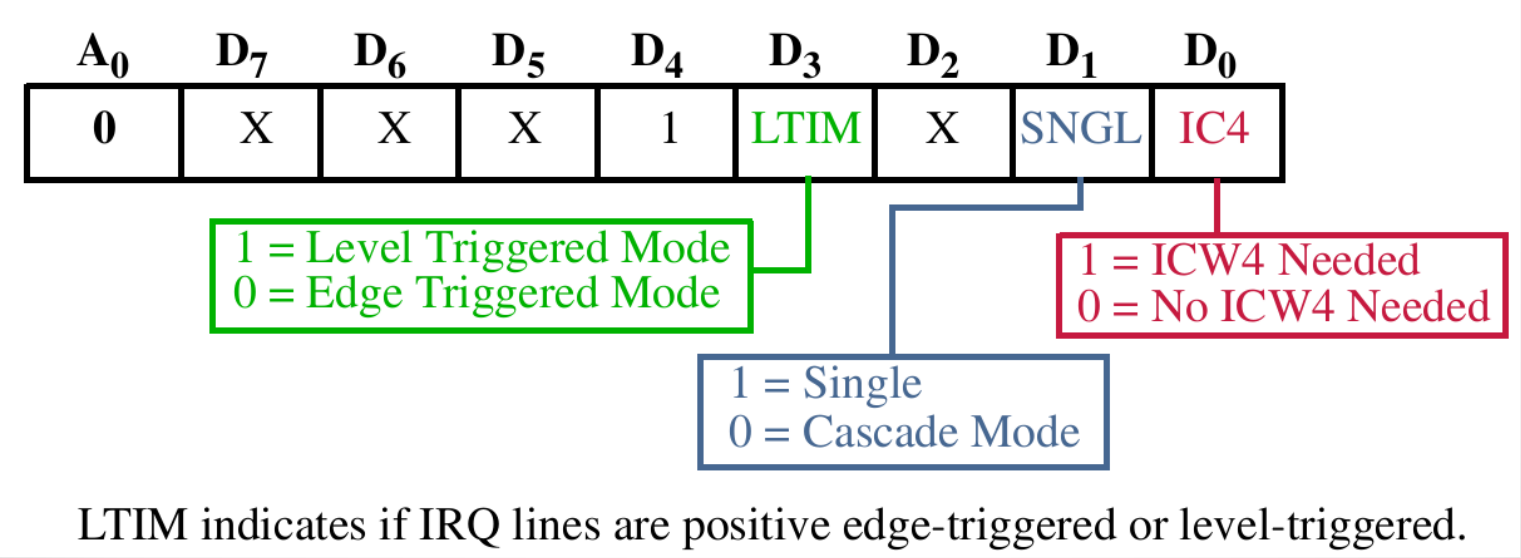


Cascading Multiple 8259A



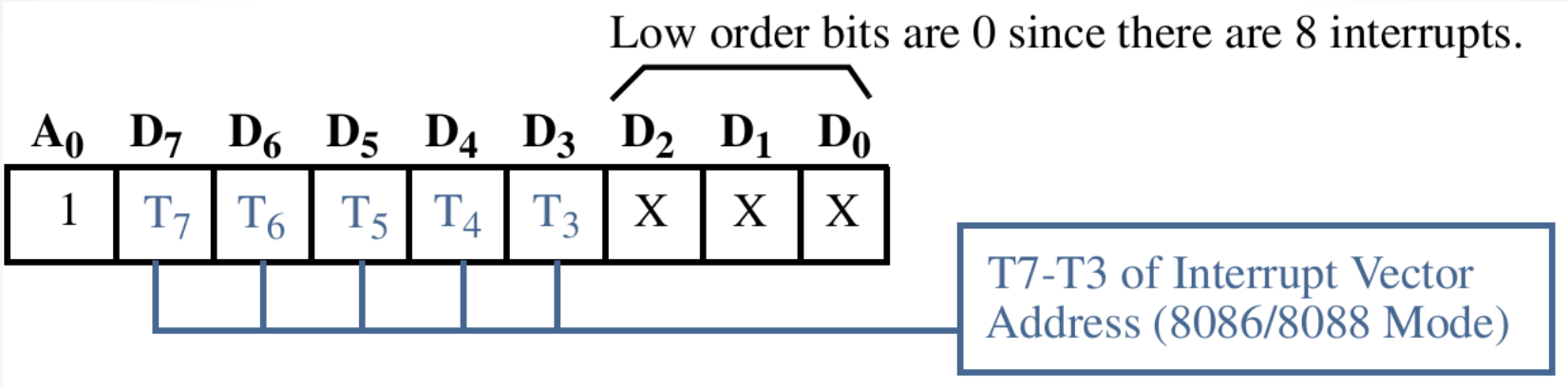
Programming the 8259A

- Programmed using Initialization (ICWs)and Operation (OCWs) Command Words.
- There are 4 ICWs.
  - At power-up, ICW1, ICW2 and ICW4 must be sent.
  - If ICW1 indicates cascade mode, then ICW3 must also be sent.
- ICW1:



Programming the 8259A

- ICW2:

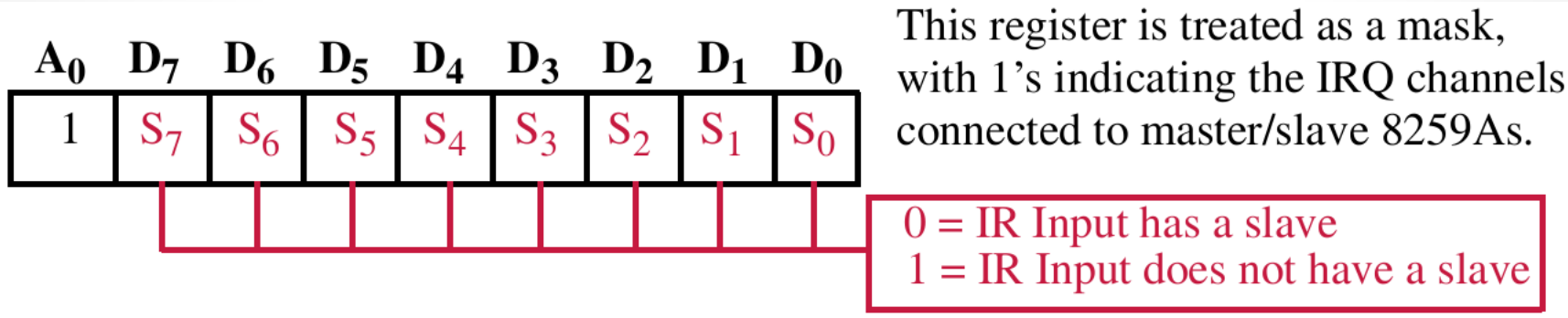


These bits determine the vector numbers used with the IRQ inputs.

For example, if programmed to generate vectors 08H-0FH, 08H is placed into these bit positions.

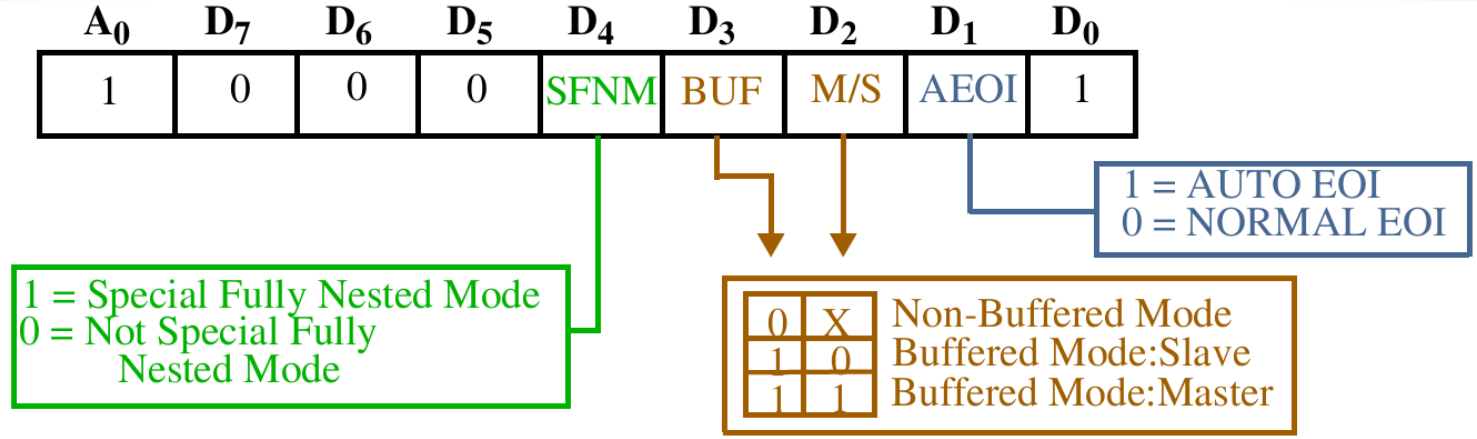
Programming the 8259A

- ICW3:



Programming the 8259A

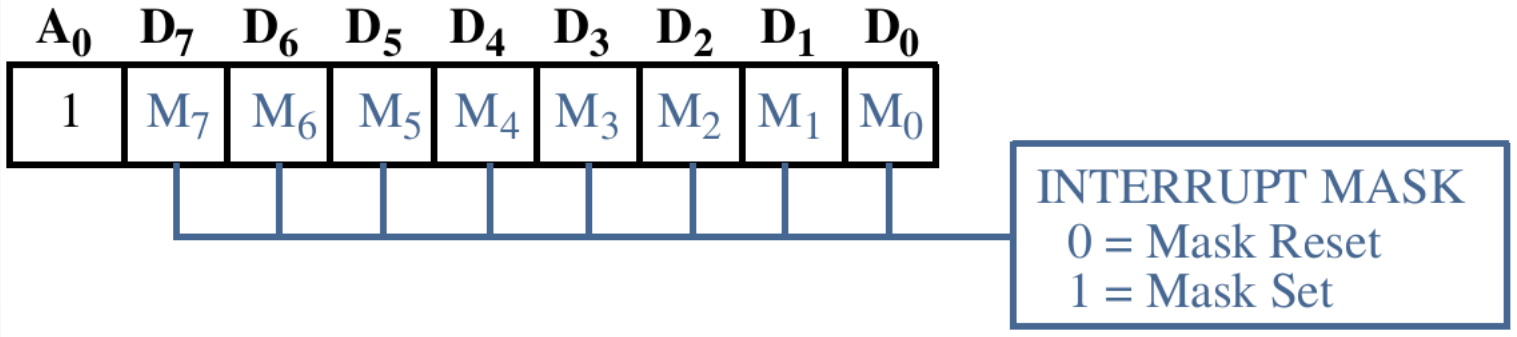
- ICW4:



- Fully nested mode allows the highest-priority interrupt request from a slave to be recognized by the master while it is processing another interrupt from a slave.
- AEOI, if 1, indicates that an interrupt automatically resets the interrupt request bit, otherwise OCW2 is consulted for EOI processing.

Programming the 8259A

- The Operation Command Words (OCWs) are used to direct the operation of the 8259A.
- OCW1:



OCW1 is used to read or set the interrupt mask register.

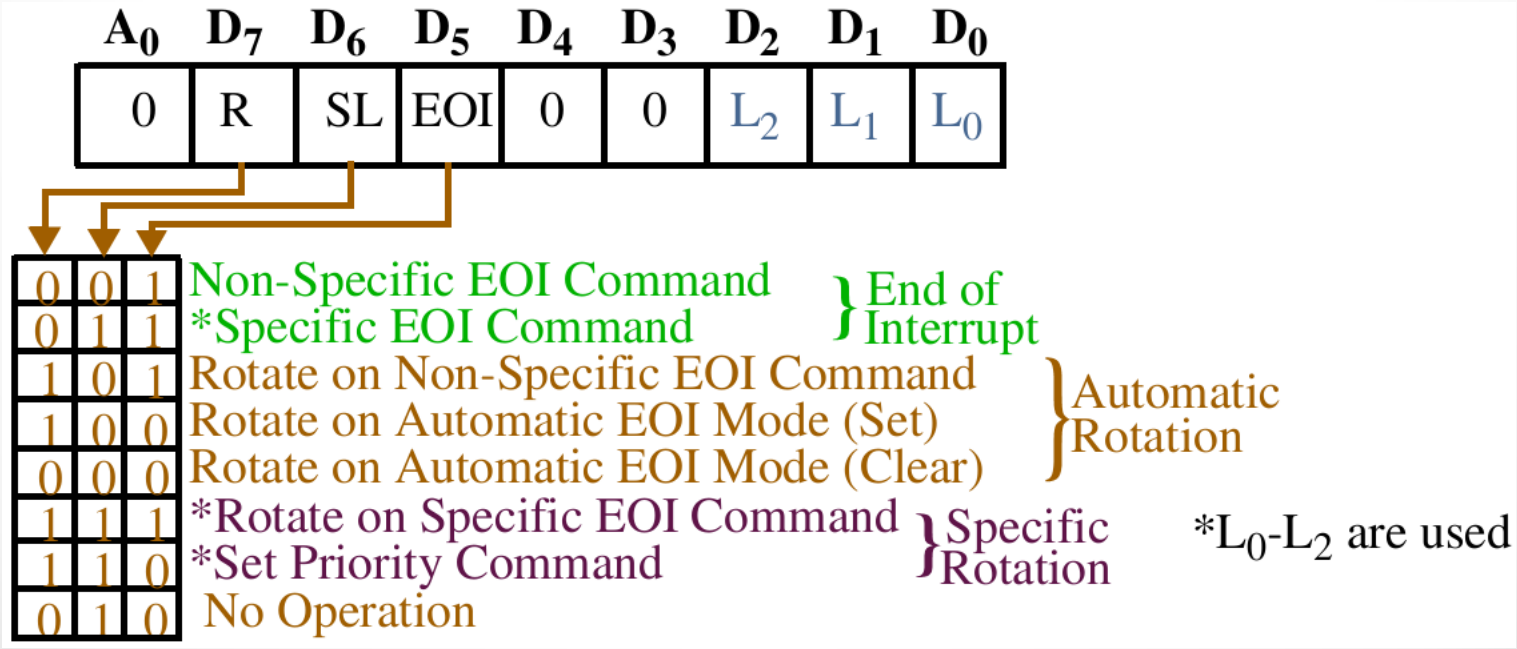
If a bit is set, it will turn off (mask) the corresponding interrupt input.

Programming the 8259A

- OCW2:

Only programmed when the AEOI mode in ICW4 is 0.

Allows you to control priorities after each interrupt is processed

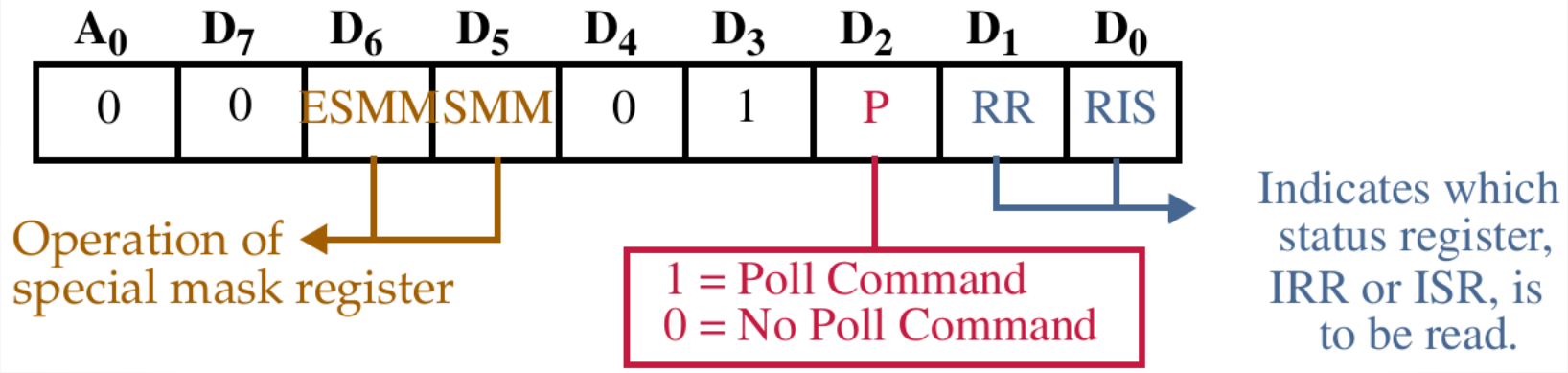


Programming the 8259A

- OCW2:
  - Non-specific EOI: Here, the ISR sets this bit to indicate EOI. The 8259A automatically determines which interrupt was active and re-enables it and lower priority interrupts.
  - Specific EOI: ISR resets a specific interrupt request given by L_2-L_0.
  - Rotate commands cause priority to be rotated w.r.t. the current one being processed.
  - Set priority: allows the setting of the lowest priority interrupt (L_2-L_0).

Programming the 8259A

- OCW3:



  - If polling is set, the next read operation will read the poll word.

If the leftmost bit is set in the poll word, the rightmost 3 bits indicate the active interrupt request with highest priority.

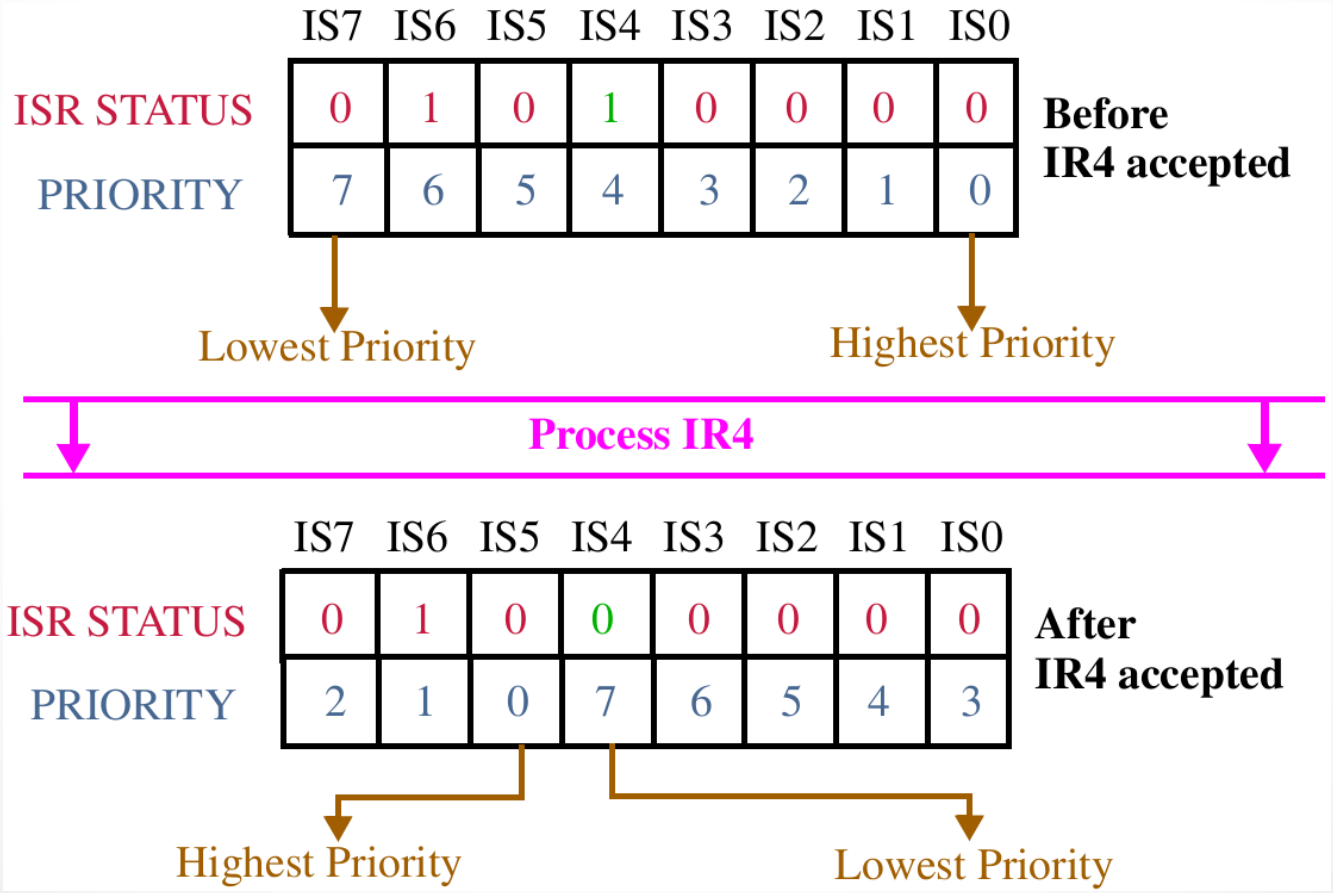
  - Allows ISR to service highest priority interrupt.

Programming the 8259A

- There are three status registers, Interrupt Request Register (IRR), In-Service Register (ISR) and Interrupt Mask Register (IMR).
  - IRR: Indicates which interrupt request lines are active.
  - ISR: Level of the interrupt being serviced.
  - IMR: A mask that indicates which interrupts are on/off.

Programming the 8259A

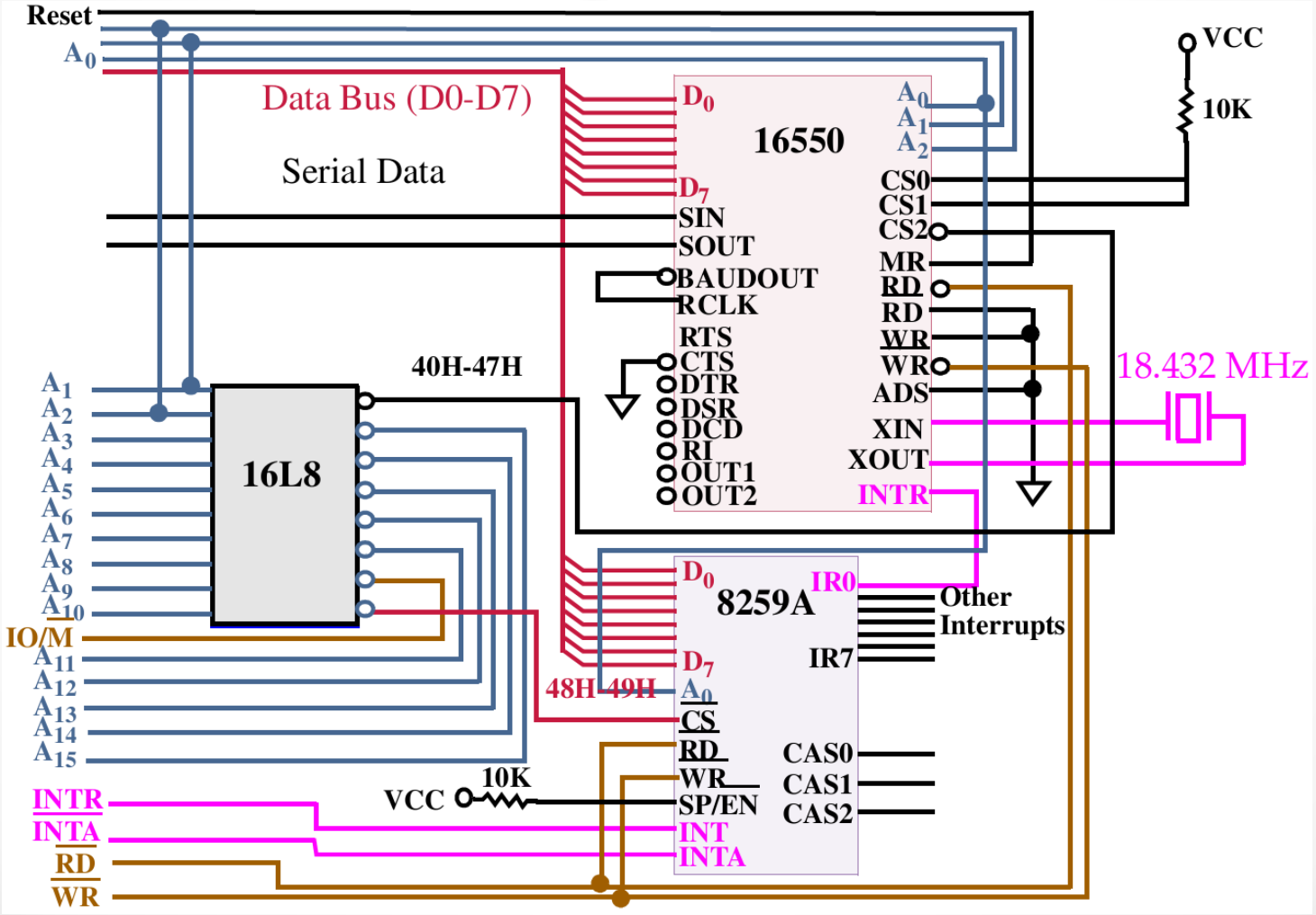
- ISR update procedure with rotating priority configured.



Example: Interfacing 16550 UART using 8259A

- In the following configuration the 16550 is connected to the 8259A through IR0.
- An interrupt is generated, if enabled through the interrupt control register, when either:
  - The transmitter is ready to send another character.
  - The receiver has received a character.
  - An error is detected while receiving data.
  - A modem interrupt occurs.
- The16550 is decoded at 40H and 47H.
- The 8259A is decoded at 48H and 49H.

Interfacing 16550 UART using 8259A



16550 UART Interrupts