# Microprocessors and Microcontrollers (EE-231)

## Lecture-16

# Main Objectives

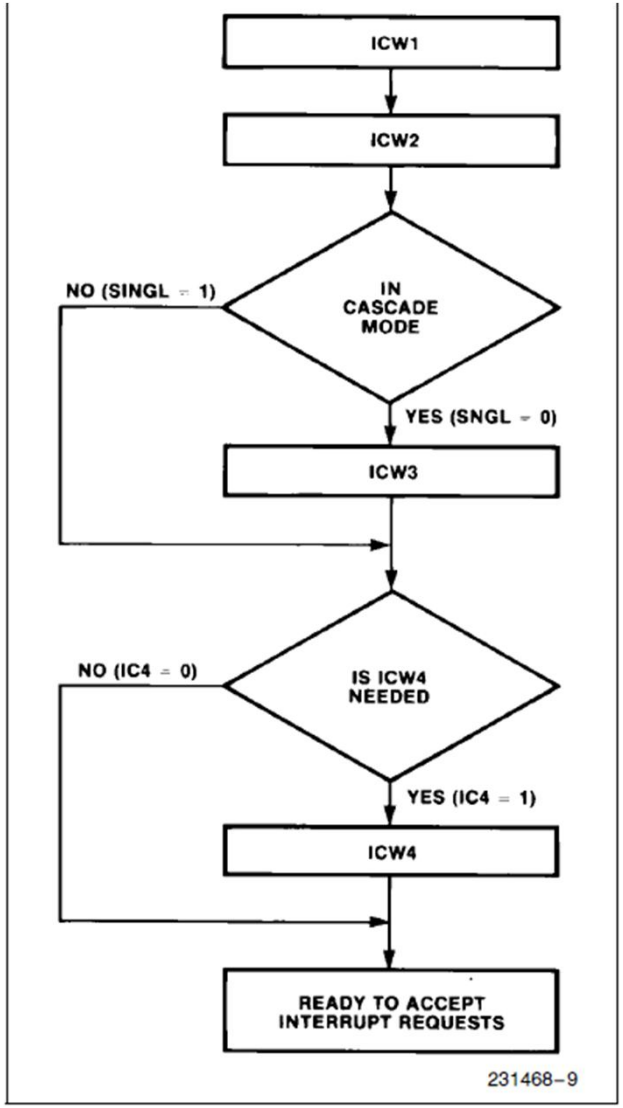
- Programming 8259A
- Direct Memory Access
  - Introduction to DMA operation
  - 8237 DMA controller

# Programming the 8259A

- 8259A is programmed by initialization and operation command words.
- **Initialization command words** (ICWs) are programmed before the 8259A is able to function in the system.
  - dictate the basic operation of the 8259A
- **Operation command words** (OCWs) are programmed during the normal course of operation.
  - OCWs control the operation of the 8259A

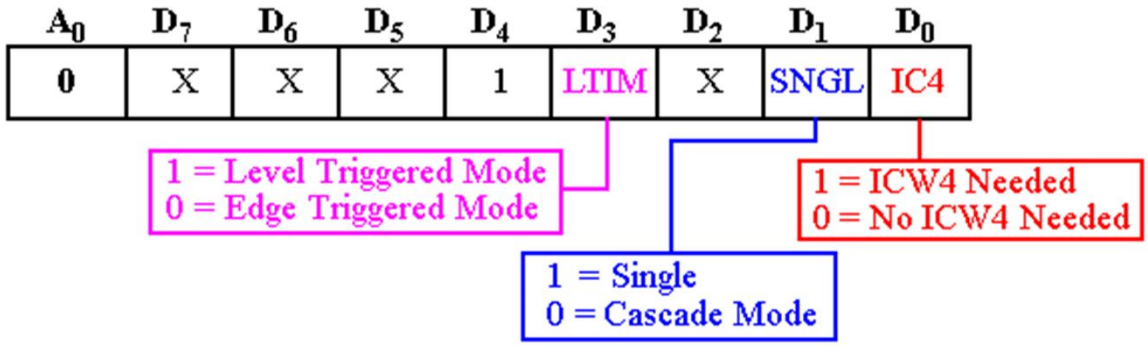
# Initialization Command Words

- the four initialization command words (ICWs) are selected when the $A_0$ pin is logic 1
- if a single 8259A is used in a system, $ICW_1$, $ICW_2$, and $ICW_4$ must be programmed when powered up
- if programmed in cascade mode, then $ICW_3$ must be programmed



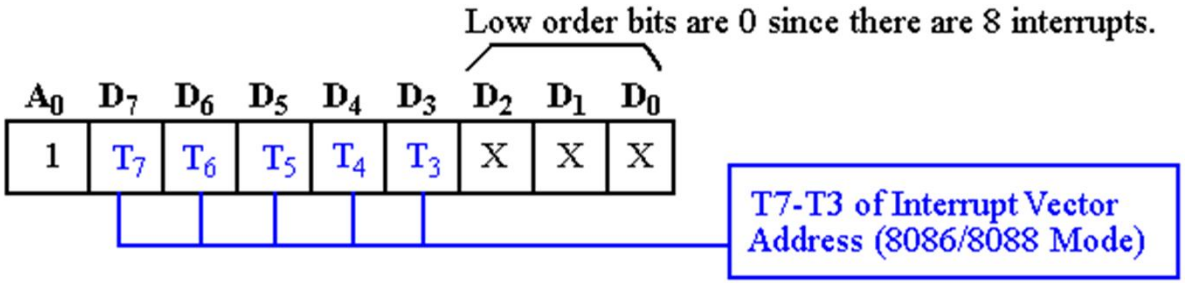
# ICW Descriptions - ICW1

- **$ICW_1$** programs basic operation of the 8259A.
  - selects single or cascade operation by programming the SNGL bit
  - if cascade operation is selected, $ICW_3$ must also be programmed
  - the LTIM bit determines whether interrupt request inputs are positive edge-triggered or level-triggered



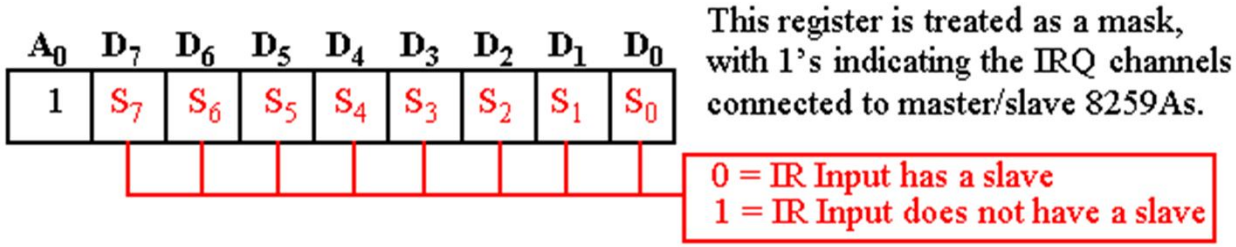
# ICW Descriptions – ICW2

- **$ICW_2$** selects the vector number used with the interrupt request inputs.
  - if programming 8259A so it functions at vector locations 08H–0FH, place 08H into this command word
  - if programming for vectors 70H–77H, place 70H in this ICW



# ICW Descriptions – ICW3

- **$ICW_3$** is used only when $ICW_1$ indicates the system is operated in cascade mode.
  - this ICW indicates where the slave is connected to the master.
  - For example, if two slaves are connected using $IR_0$ and $IR_1$, the master is programmed with an $ICW_3$ of 03H
  - one slave is programmed with an $ICW_3$ of 00H and the other with an $ICW_3$ of 01H

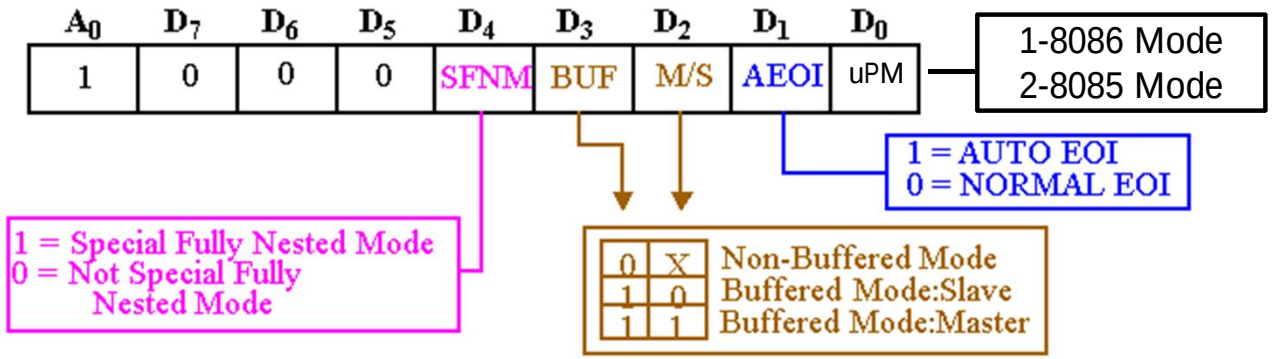



**ICW3 (Slave Device)**

| A0 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | ID2 | ID1 | ID0 |

# ICW Descriptions – ICW4

- **$ICW_4$** is programmed for use with the 8086–Pentium 4 processors, and the rightmost bit must be logic 1 to select operation with them.
- Remaining bits are programmed as follows:
  - **SFNM**—Selects the special fully nested mode of operation if logic 1 is placed in this bit
  - **BUF** and **M/S**—Buffered and master slave are used together to select buffered operation or non-buffered operation for the 8259A as a master or a slave



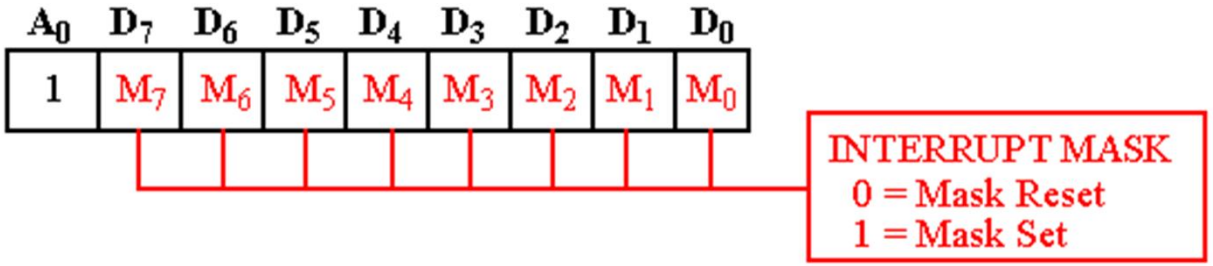
# ICW Descriptions – ICW4

- **AEOI**—Selects automatic or normal end of interrupt. The EOI commands of OCW2 are
  used only if AEOI mode is not selected by $ICW_4$
- if AEOI is selected, the interrupt automatically resets the interrupt request bit and does not modify priority
- this is the preferred mode of operation for the 8259A and reduces the length of the interrupt service procedure

# Operation Command Words

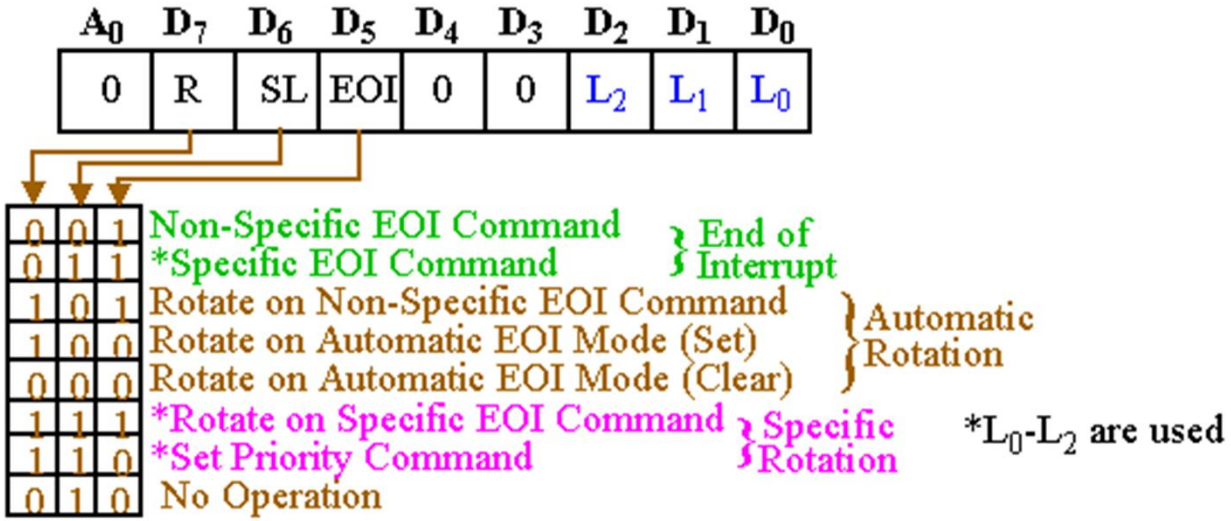
- used to direct 8259A operation once it is programmed via ICW
- OCWs are selected when the $A_0$ pin is at logic 0 level
- except $OCW_1$, which is selected when $A_0$ is logic 1

- **$OCW_1$** is used to set and read the interrupt mask register.
  - when a mask bit is set, it will turn off (mask) the corresponding interrupt input
  - the mask register is read when $OCW_1$ is read
  - because the state of the mask bits is unknown when 8259A is first initialized, $OCW_1$ must be programmed after programming the ICW upon initialization



# OCW-Descriptions-OCW2

- **$OCW_2$** is programmed only when the AEOI mode is not selected for the 8259A.



# OCW-Descriptions-OCW2

- It Selects the way 8259A responds to an interrupt. Modes are listed as follows:
  - **Nonspecific End-of-Interrupt**—Command sent by the interrupt service procedure to signal the end of the interrupt
  - 8259A determines which interrupt level was active and resets the correct interrupt status register bit
  - resetting the bit allows the interrupt to take action again or a lower priority interrupt to take effect
  - **Specific End-of-Interrupt**—A command that allows a specific interrupt request to be reset
  - exact position determined with bits $L_2$–$L_0$ of $OCW_2$

# OCW-Descriptions-OCW2

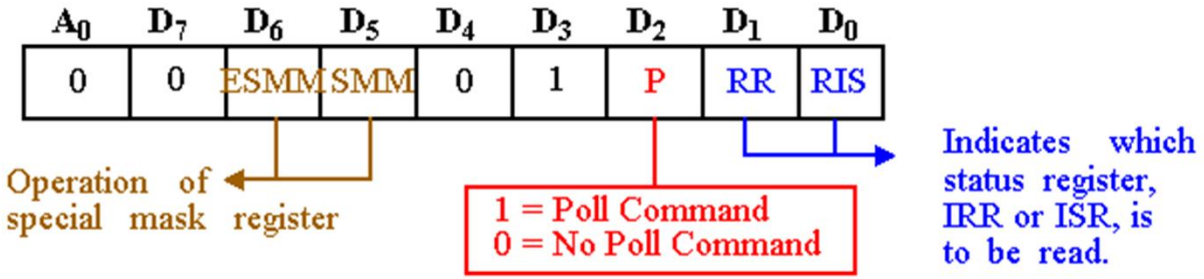
- **$OCW_2$** interrupt modes:
  - **Rotate-on-Nonspecific EOI**—functions exactly like the Nonspecific End-of-Interrupt command, except it rotates interrupt priorities after resetting the interrupt status register bit
  - the level reset by this command becomes the lowest priority interrupt
  - if $IR_4$ was just serviced by this command, it becomes the lowest priority interrupt input and $IR_5$ becomes the highest priority
  - **Rotate-on-Automatic EOI**—A command that selects automatic EOI with rotating priority
  - must only be sent to the 8259A once if this mode is desired.
  - if this mode must be turned off, use the clear command

# OCW-Descriptions-OCW2

- **$OCW_2$** interrupt modes:
  - **Rotate-on-Specific EOI**—Functions as the specific EOI, except that it selects rotating priority.
  - **Set priority**—Allows the programmer to set the lowest priority interrupt input using the $L_2$–$L_0$ bits.

# OCW-Descriptions-OCW3

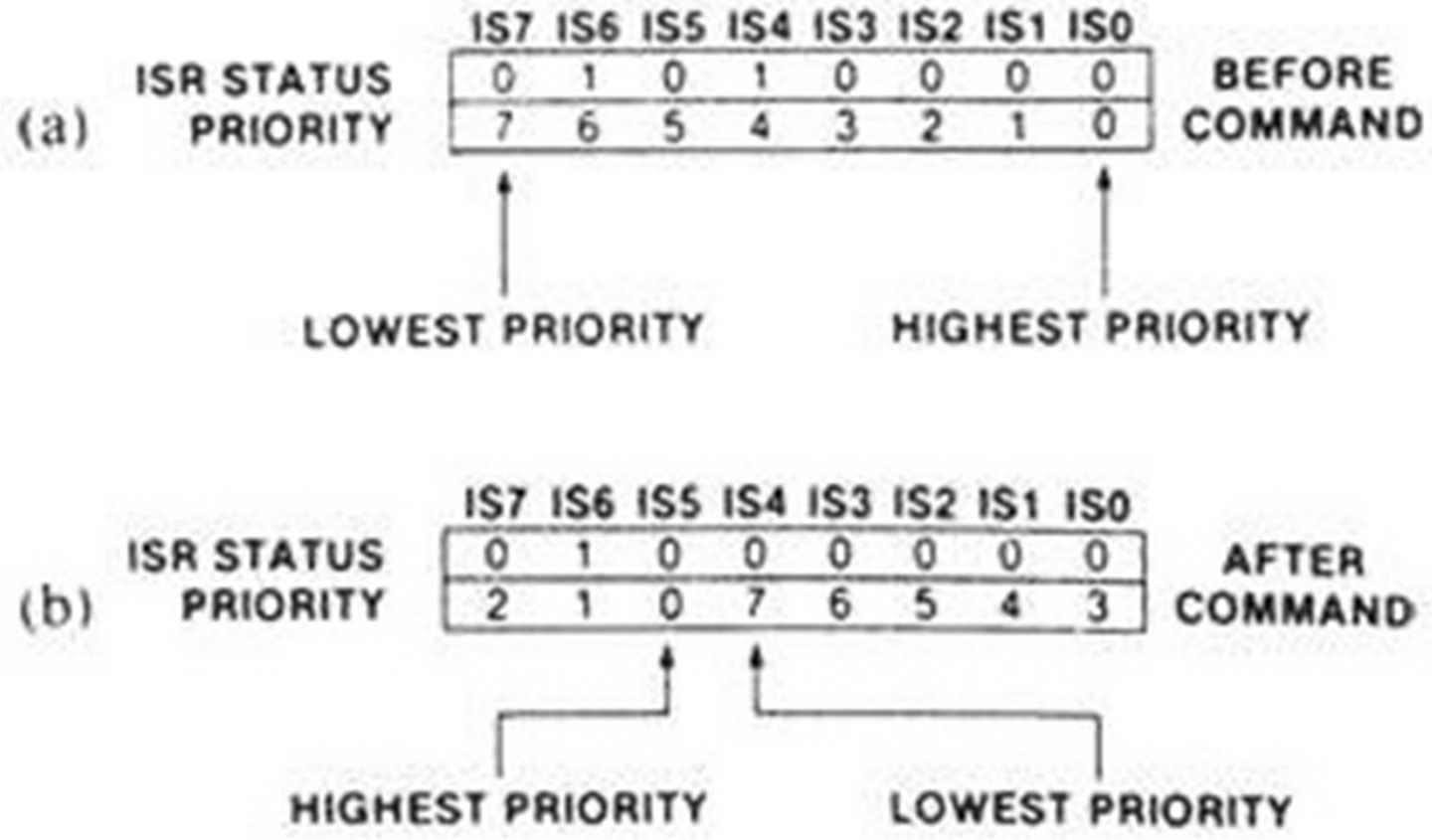
- **$OCW_3$** selects register to be read, operation of the special mask register & poll command.
  - if polling is selected, the P bit must be set and then output to the 8259A; the next read operation will read the poll word
  - the rightmost three bits of the word indicate the active interrupt request with the highest priority
  - the leftmost bit indicates if there is an interrupt and must be checked to determine whether the rightmost three bits contain valid information



# Status Register

- Three status registers are readable in 8259A:
- Interrupt request register (IRR).
  - an 8-bit register that indicates which interrupt request inputs are active
- In-service register (ISR).
  - an 8-bit register that contains the level of the interrupt being serviced
- interrupt mask register (IMR).
  - an 8-bit register that holds the interrupt mask bits and indicates which interrupts are masked off

The 8259A in-service register (ISR). (a) Before $IR_4$ is accepted and (b) after $IR_4$ is accepted. (Courtesy of Intel Corporation.)

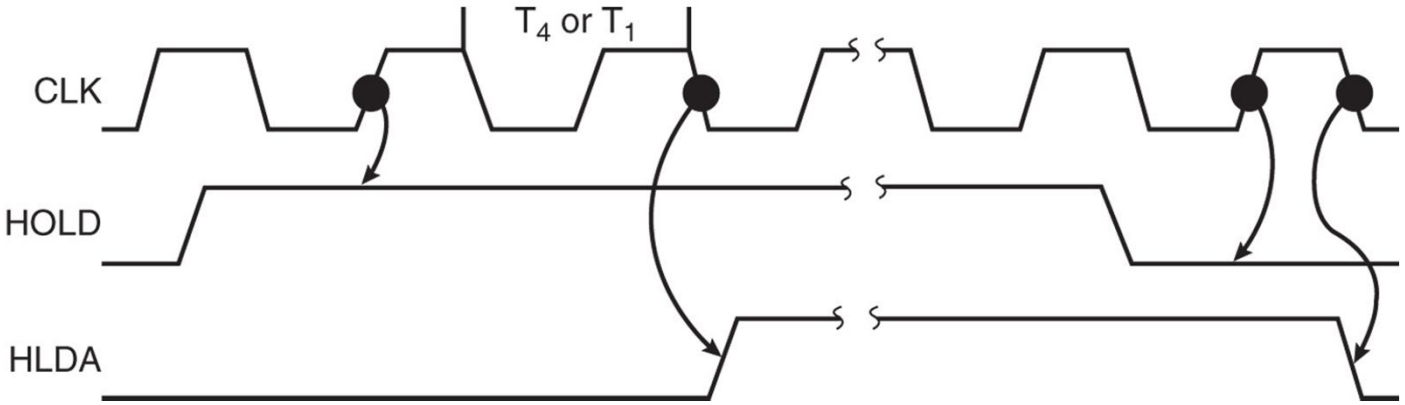


# Introduction to DMA

- The DMA I/O technique provides direct access to the memory while the microprocessor is temporarily disabled.
- It avoids the slow speed of programmed I/O when moving large amounts of data between memory and a peripheral
- DMA operations involve transferring data between memory and disk systems and video systems.
- Disk memory includes floppy, fixed, and optical disk storage. Video systems include digital and analog monitors.

# BASIC DMA OPERATION

- Two control signals are used to request and acknowledge a direct memory access (DMA) transfer in the microprocessor-based system.
  - the **HOLD** pin is an input used to request a DMA action
  - the **HLDA** pin is an output that acknowledges the DMA action




  - HOLD is sampled in any clocking cycle
  - when the processor recognizes the hold, it stops executing software and enters hold cycles
  - HOLD input has higher priority than INTR or NMI but less than RESET

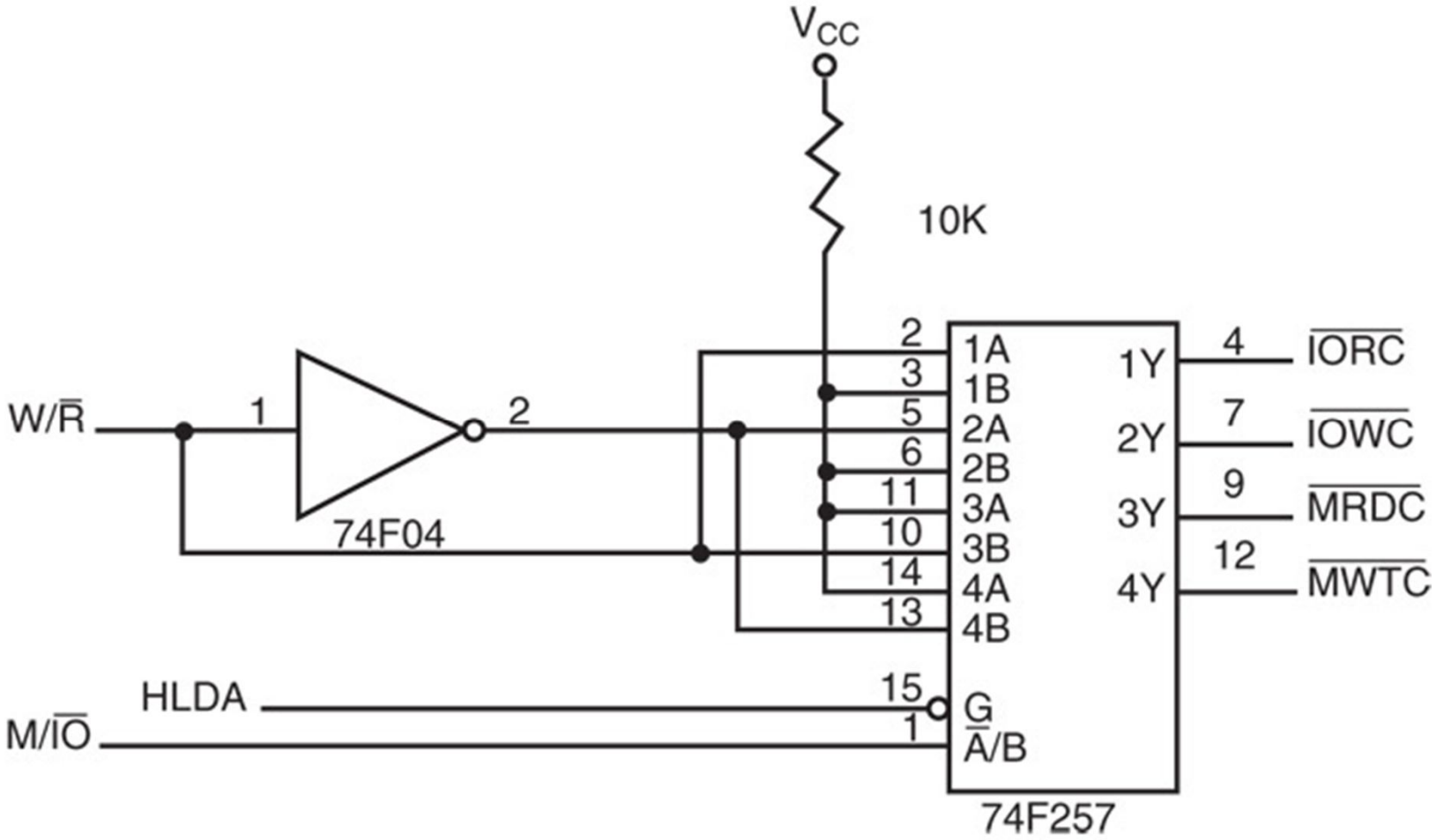
# BASIC DMA OPERATION

- HLDA becomes active to indicate the processor has placed its buses at high-impedance state.
  - as can be seen in the timing diagram, there are a few clock cycles between the time that HOLD changes and until HLDA changes
- HLDA output is a signal to the requesting device that the processor has relinquished control of its memory and I/O space.
  - one could call HOLD input a DMA request input and HLDA output a DMA grant signal

# Basic DMA Definitions

- Direct memory accesses normally occur between an I/O device and memory without the use of the microprocessor.
  - a **DMA read** transfers data from the memory to the I/O device
  - A **DMA write** transfers data from an I/O device to memory
- Memory & I/O are controlled simultaneously.
  - which is why the system contains separate memory and I/O control signals
  - A DMA read causes the MRDC and IOWC signals to activate simultaneously, transferring data from memory to the I/O device
  - A DMA write causes the MWTC and IORC signals to both activate.
  - The DMA controller provides memory with its address, and controller signal (DACK) selects the I/O device during the transfer.
  - 8086/8088 require a controller or circuit for control bus signal generation, since it does not have separate I/O and memory signals

**A circuit that generates system control signals for 8086 in a DMA environment.**

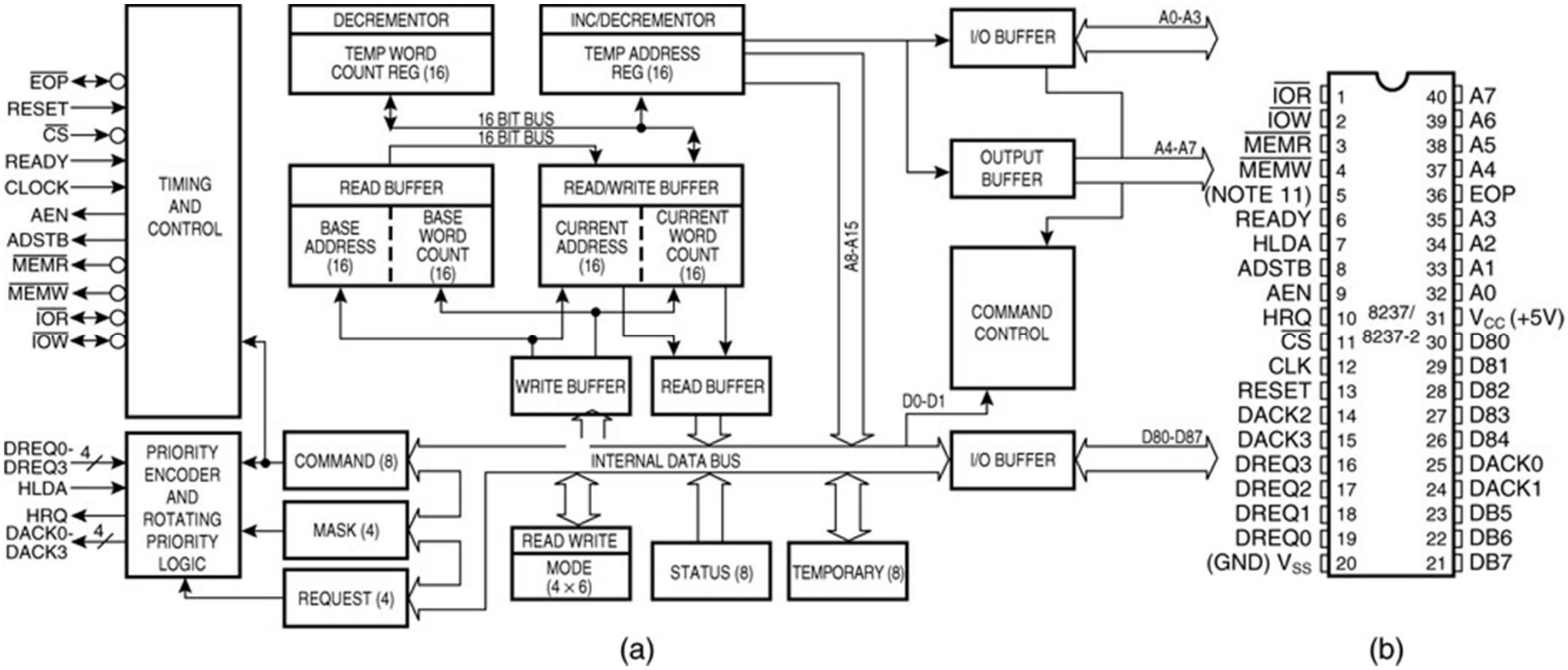


# DMA Speed

- Data transfer speed is determined by speed of the memory device or a DMA controller.
- In many cases, the DMA controller is the one that *slows* the speed of the system when transfers occur.
- The switch to serial data transfers in modern systems has made DMA is less important.
- The serial PCI Express bus transfers data at rates exceeding DMA transfers.
- The SATA (serial ATA) interface for disk drives uses serial transfers at the rate of 300 Mbps and has replaced DMA transfers for hard disks
- Serial transfers on main-boards between components can approach 20 Gbps for the PCI Express connection.

# THE 8237 DMA CONTROLLER

- The 8237 supplies memory & I/O with control signals and memory address information during the DMA transfer.
  - actually a special-purpose microprocessor whose job is high-speed data transfer between memory and I/O



# THE 8237 DMA CONTROLLER

- 8237 is not a discrete component in modern microprocessor-based systems.
  - it appears within many system controller chip sets
- 8237 is a four-channel device compatible with 8086/8088, adequate for small systems.
- 8237 is capable of DMA transfers at rates up to 1.6M bytes per second.
  - each channel is capable of addressing a full 64K-byte section of memory and transfer up to 64K bytes with a single programming

# THE 8237 DMA CONTROLLER

- **HLDA**
- A **hold acknowledge** signals 8237 that the microprocessor has relinquished control of the address, data, and control buses.
- **DREQ0–DREQ3**
- DMA request inputs are used to request a transfer for each of the four DMA channels.
- The polarity of these inputs is programmable, so they are either active-high or active-low inputs.
- **I/OR and I/OW**
- **I/O read** and **I/O write** are bidirectional pins used during programming and during a DMA write cycle.

# THE 8237 DMA CONTROLLER

- **EOP**
- End-of-process is a bidirectional signal used as an input to terminate a DMA process or as an output to signal the
  end of the DMA transfer.
- Often used to interrupt a DMA transfer at the end of a DMA cycle
- **A0-A3**
- These address pins select an internal register during programming and provide part of the DMA transfer address during a DMA action.
- Address pins are outputs that provide part of the DMA transfer address during a DMA action
- **HRQ**
- Hold request is an output that connects to the HOLD input of the microprocessor
- in order to request a DMA transfer.

# THE 8237 DMA CONTROLLER

- **$DACK_0$–$DACK_3$**
- DMA channel acknowledge outputs acknowledge a channel DMA request.
- These outputs are programmable as either active-high or active-low signals.
- DACK outputs are often used to select the DMA- controlled I/O device during the DMA transfer.
- **AEN**
- Address enable signal enables the DMA address latch connected to the DB7–DB0 pins on the 8237.
- **ADSTB**
- Address strobe functions as ALE, except it is used by the DMA controller to latch address bits A15–A8 during the DMA transfer