Programmable DMA Controller (8237)

Course Teacher:

Md. Obaidur Rahman, Ph.D.
Professor
Department of Computer Science and Engineering (CSE)
Dhaka University of Engineering & Technology (DUET), Gazipur.

Course ID: CSE - 4619
Course Title: Peripherals, Interfacing and Embedded Systems
Department of Computer Science and Engineering (CSE)
Islamic University of Technology (IUT), Gazipur.

Lecture References:

- Book:
  - *Microprocessors and Interfacing: Programming and Hardware*, **Author:** Douglas V. Hall
  - *Microprocessor Architecture, Programming and Applications with 8085 (Chapter-15)*, **Author:** Ramesh Gaonkor
- Lecture Materials:
  - *I/O System Design*, Dr. Esam Al_Qaralleh, CE Department, Princess Sumaya University for Technology.
  - Computer Peripherals and Interfacing Lecture, Mahmud Hasan, Stamford University, Bangladesh.

Review of I/O Types

- **Programmed I/O:** I/O between memory and the I/O device is performed by the Processor: e.g.

  IN AL,DX
  MOV [DI],AL; Transfer is through the μP - **slow!**

  - **Polling/Handshaking I/O**
    Processor checks device readiness repeatedly, e.g. in a tight loop

  - **Interrupt-driven I/O**
    Device signals its readiness by an interrupt. Processor performs I/O by executing an ISR. Otherwise processor is doing other useful work

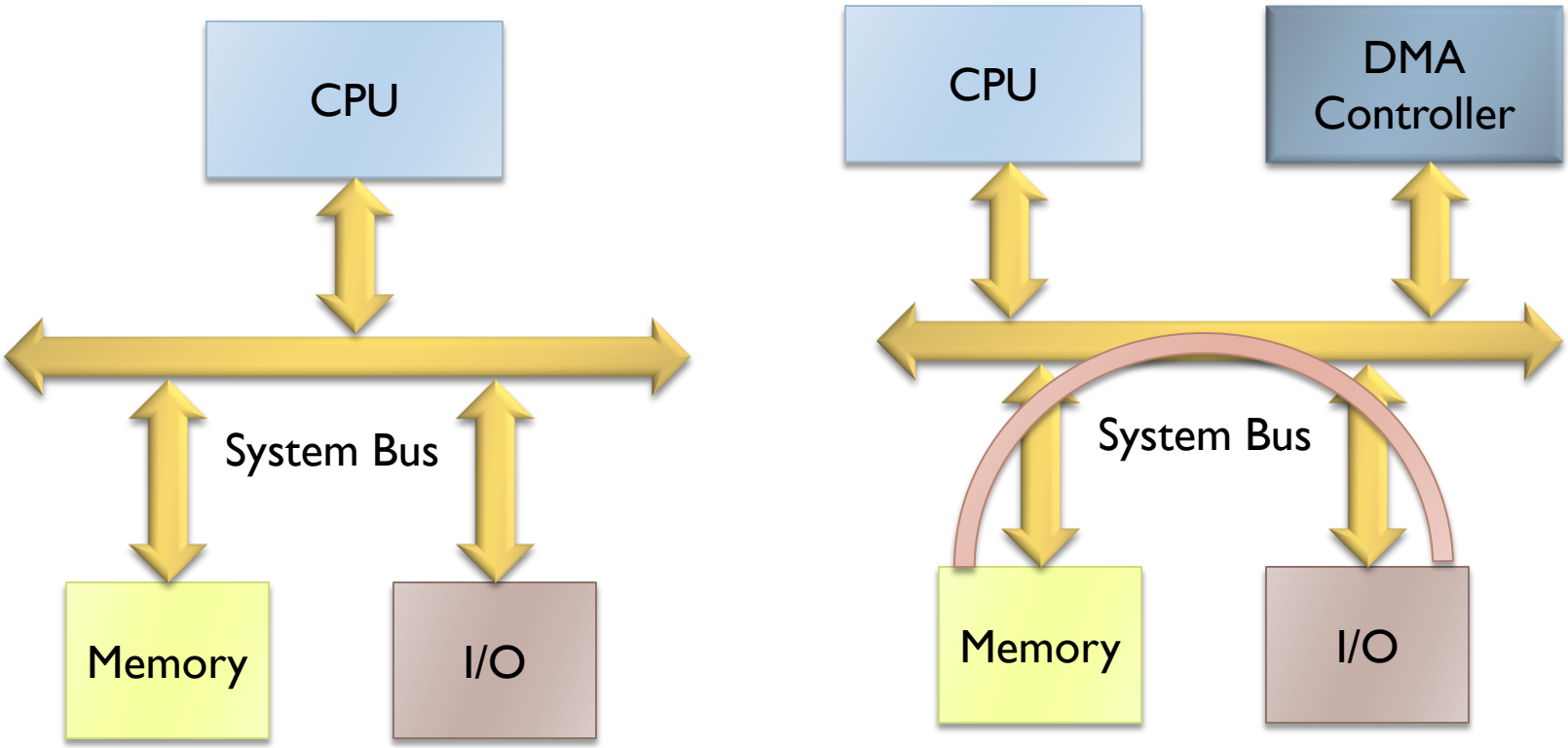
Review of I/O Types

- Problems with programmed I/O
  - Processor wastes time for polling
  - Lets take example of Key board
    - Waiting for a key to be pressed,
    - Waiting for it to be released
    - May not satisfy timing constraints associated with some devices : **Disk read or write**

Basic Idea of DMA Technique

- Frees the processor of the data transfer responsibility
- Avoids the slow speed of programmed I/O when moving large amounts of data between memory and a peripheral
- Data transfer is coordinated by a DMA controller- not the processor
- Avoids the bottleneck of having to channel data through the mP
- Uses the 3 mP buses, so the mP is unable to use them temporarily
- Speed is limited only by those of the memory and the DMAC

Basic Idea of DMA Technique



Basic Idea of DMA Technique

- Direct Memory Access (DMA) is a technique that allows the direct data transfer between memory and I/O devices keeping the microprocessor temporary disabled.
- Used for transfer of data between the RAM and some external peripheral device without the use of the **Microprocessor**.
- Under normal circumstances you would write a program to perform the input/output using a microprocessor. This is known as ***Programmed I/O*** since you are programming it.
- For typical microprocessors 1 (one) byte of data transfer between RAM and I/O take 5 – 10 microseconds.
- This is also regarded inadequate for most high-speed applications.
- The use of DMA would typically reduce this time to 1 (one) microsecond per byte as it doesn't get slowed down by microprocessor interpreting each instruction.

DMA Applications

- Wherever large amounts of data need to be transferred fast between memory and an I/O peripheral device, e.g.
  - Hard disk, CD
  - Video memory to refresh display
  - Sound cards
  - Network cards

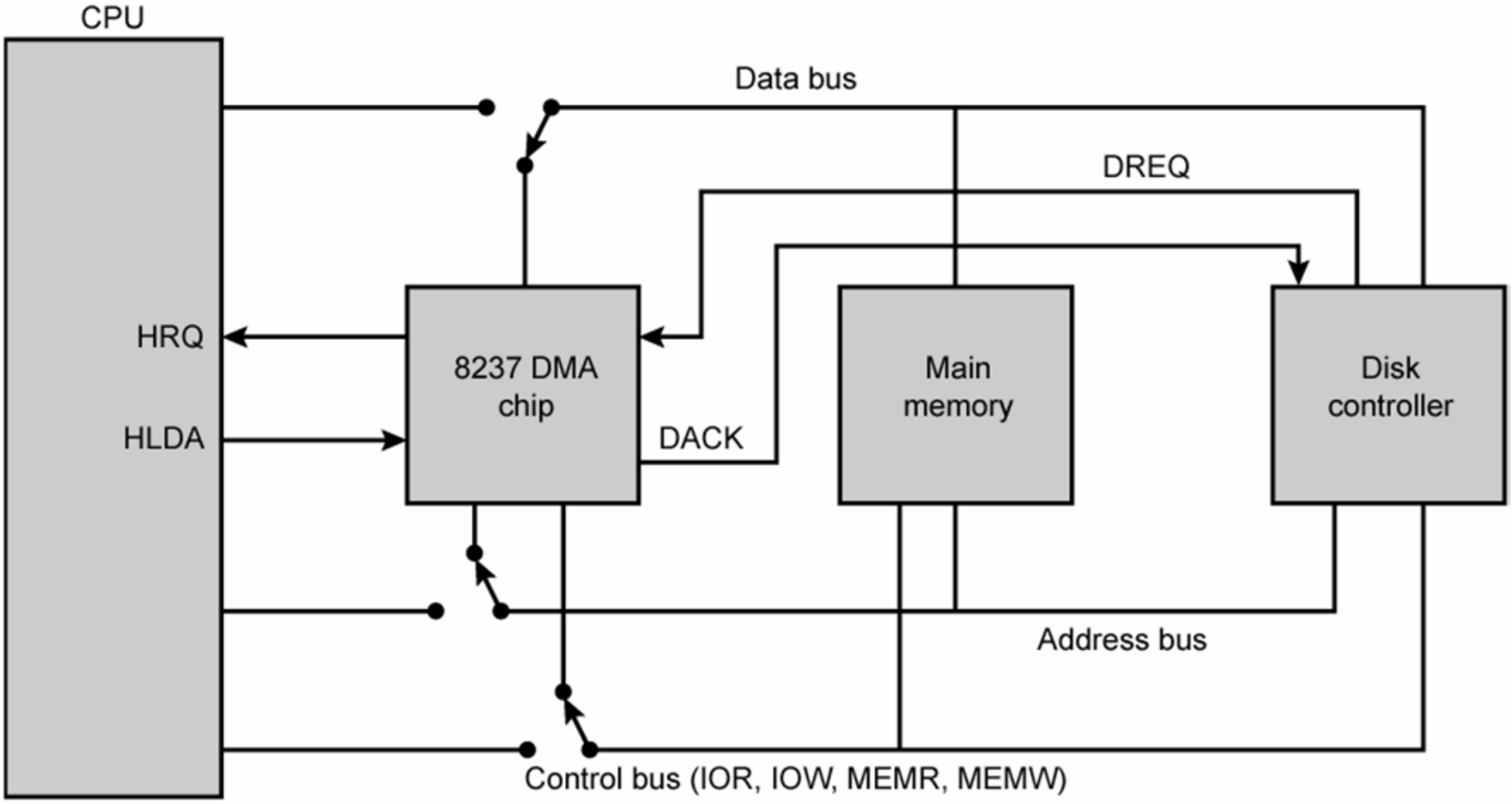
DMA Controller

- A ***DMA Controller chip*** is may be built into the microprocessor or may be found external.
- We can see that the DMA controller has to perform operations that are very similar to operations performed by a microprocessor.
- They are thus generally designed and built by microprocessor manufacturers.
- Two control signals are used to **request and acknowledge** DMA transfer.
  - 1. **HOLD** pin is an input that is used to request a DMA action.
  - 2. **HLDA** pin is an output that is used to acknowledge a DMA action.

DMA Controller

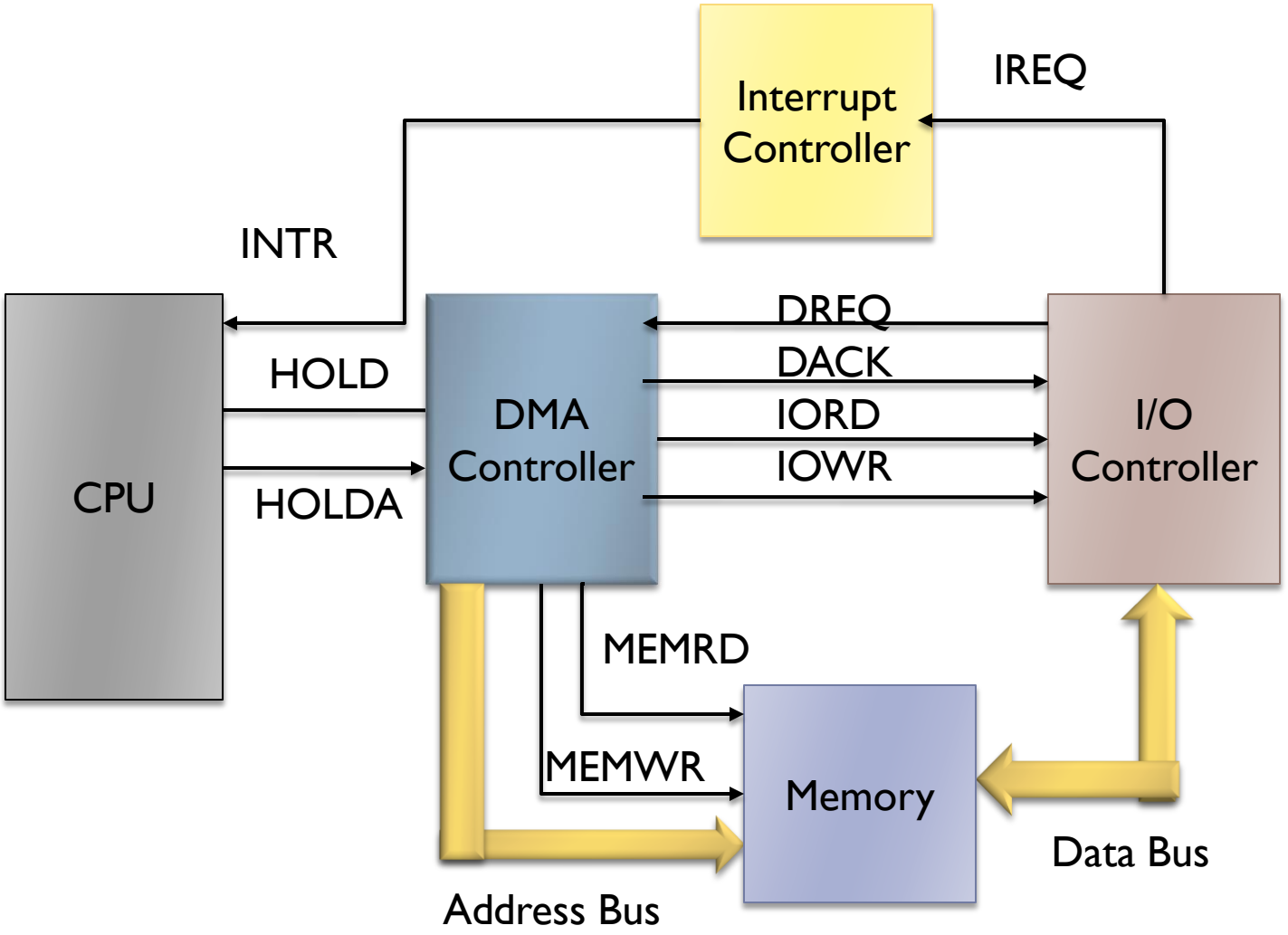
- DMA is implemented using a DMA controller
- DMA controller
  - Acts as slave to processor
  - Receives instructions from processor
  - Example: Reading from an I/O device
    - Processor gives details to the DMA controller
    - I/O device number
    - Main memory buffer address
    - Number of bytes to transfer
    - Direction of transfer (memory → I/O device, or vice versa)

8237 DMA Usage of Systems Bus

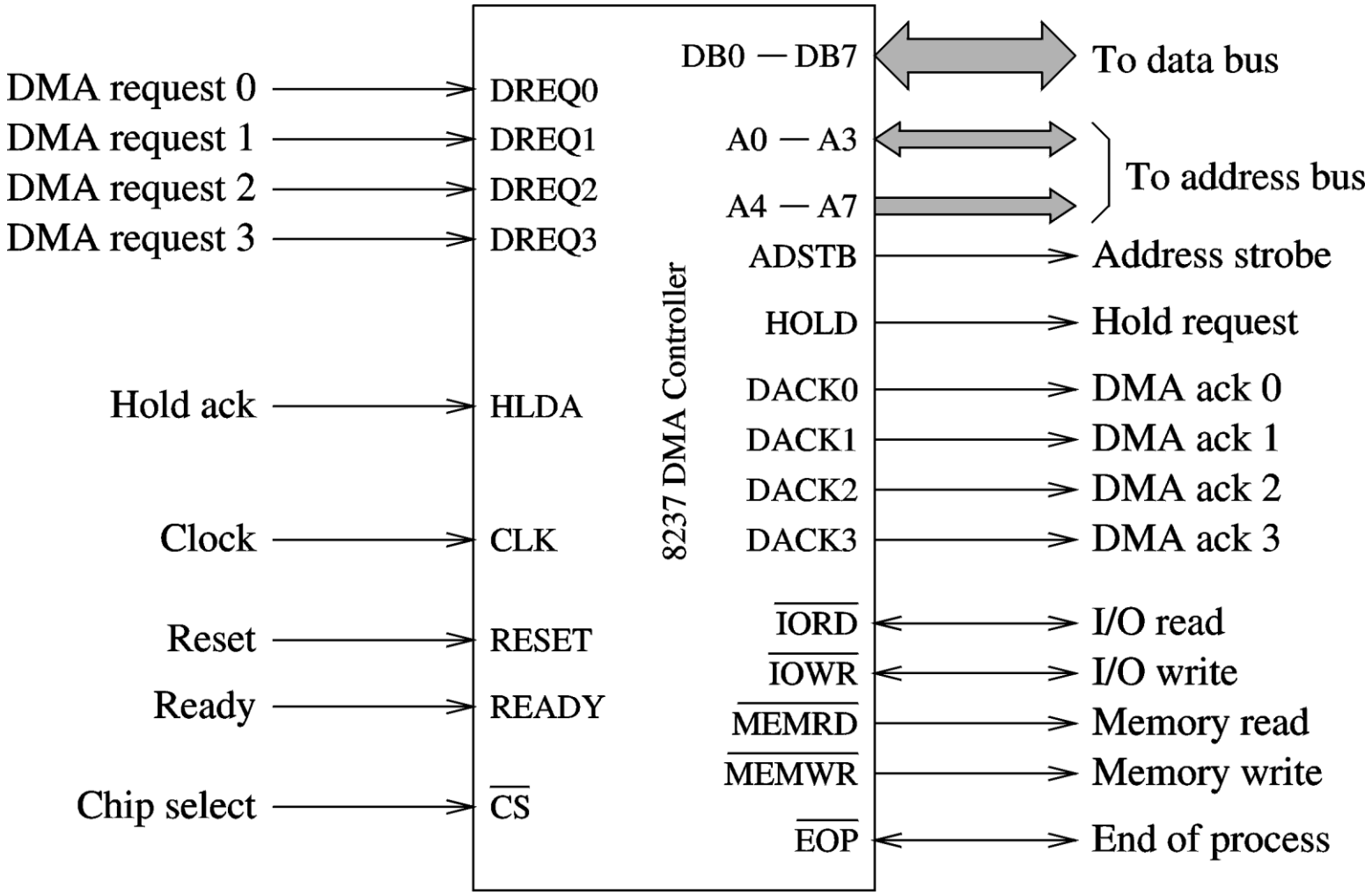


DACK = DMA acknowledge
DREQ = DMA request
HLDA = HOLD acknowledge
HRQ = HOLD request

DMA Controller

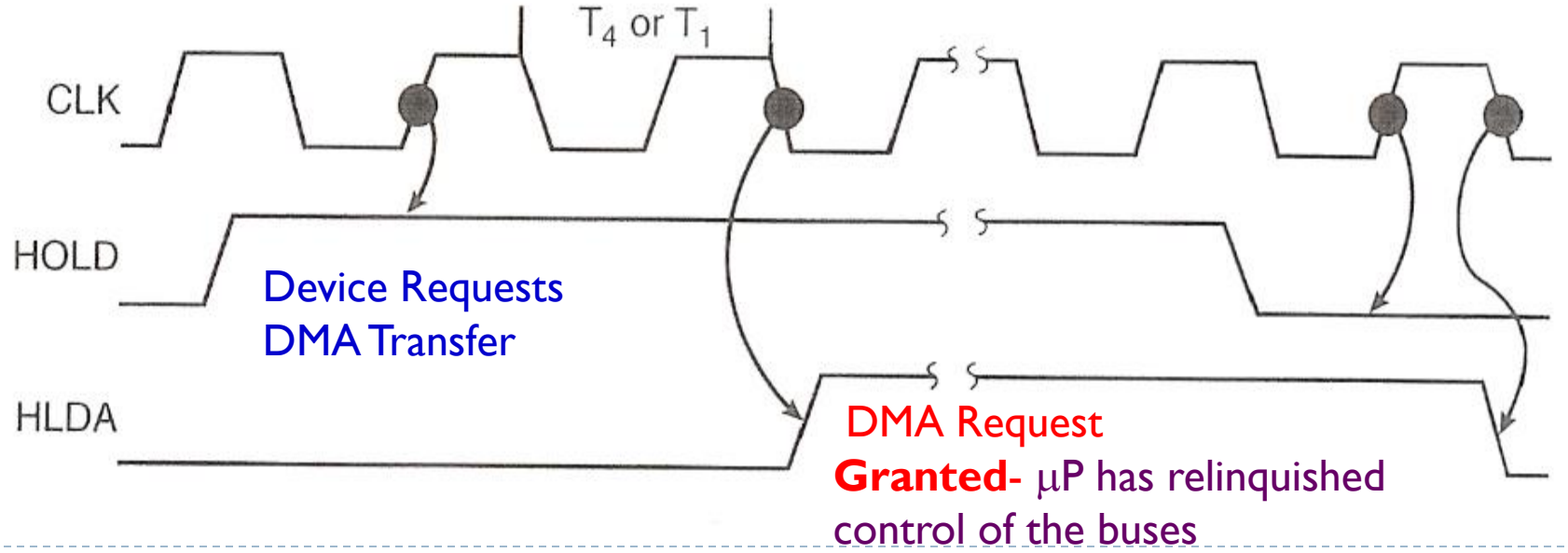


8237 is Programmable DMA Controller



DMA: HOLD and HOLDA Signals

- IF HOLD = logic-1, DMA action is requested.
- Microprocessor responds to it by suspending its execution and placing its address, data and control buses at **high-impedence state.**
- **Then the I/O devices can access the system buses.**



DMA: HOLD and HOLDA Signals

- HOLD: DMA to CPU
  - DMA Sends HOLD High to CPU
  - I (DMA) want BUS Cycles
- HOLDA
  - CPU send HOLDA
  - BUS is granted to DMA to do the transfer
  - DMA is from Slaves to Master mode
- HOLD Low to CPU
  - I (DMA) finished the transfer
- Cycle Stealing if One BUS
- Other wise Separate process independent of processing

DMA Control Signals

- Because during a DMA both memory and an I/O device may be accessed simultaneously, the DMAC may need to generate:
  - #MRDC and #IOWC (simultaneously) for memory to I/O device transfers
  - #IORC and #MRWC (simultaneously) for I/O device to memory transfers

Type	Advantages	Disadvantages
Polling	- Fastest response to device request - Simplest hardware and software	- Wasted processor resources (always waiting)
Interrupt	- More efficient use of processor time (Processor executes program- checks for interrupts only at the end of every instructions)	- Delay in response time to interrupt (interrupt latency) - Overhead due to interrupt processing, e.g. saving return address & registers, context switching - Increased cost and complexity of hardware and software
DMA	- Fastest data transfer rates (approaching those determined by memory/device access time) Address generation by fast DMAC hardware- not by processor software	- Need a DMAC device - Highest cost and complexity in hardware and software

Thank You !!