MODULE 8: Input/Output Systems

# Lecture 8.2
# Input/Output Control

Prepared By:
- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering
Virginia Tech

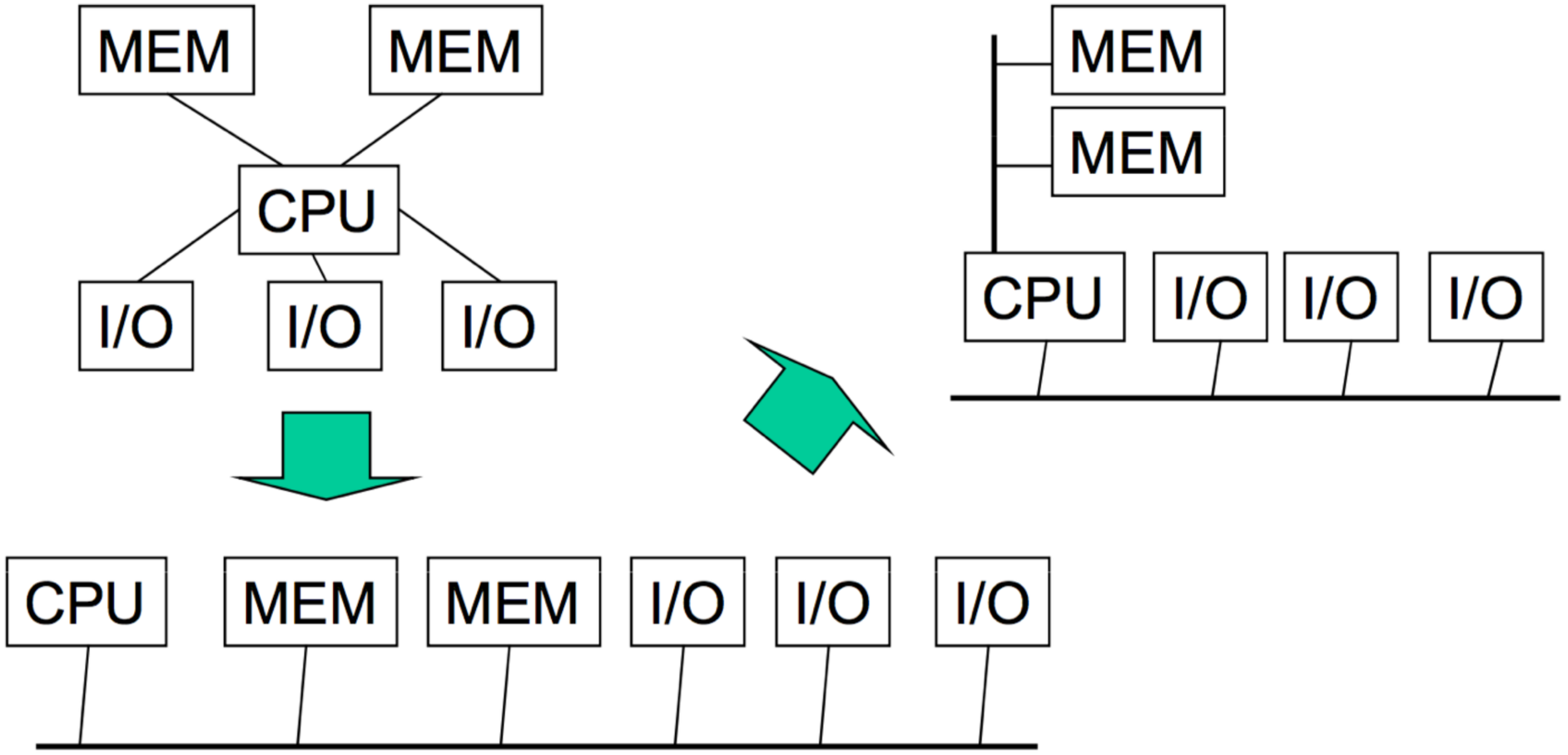
# Lecture 8.2 Objectives

- Enumerate the components of an I/O subsystem
- Identify the motivation for a bus
- Describe the operation of synchronous and asynchronous buses
- Describe the operation of four types of I/O control: programmed I/O, interrupt-driven I/O, Direct Memory Access, and channel I/O

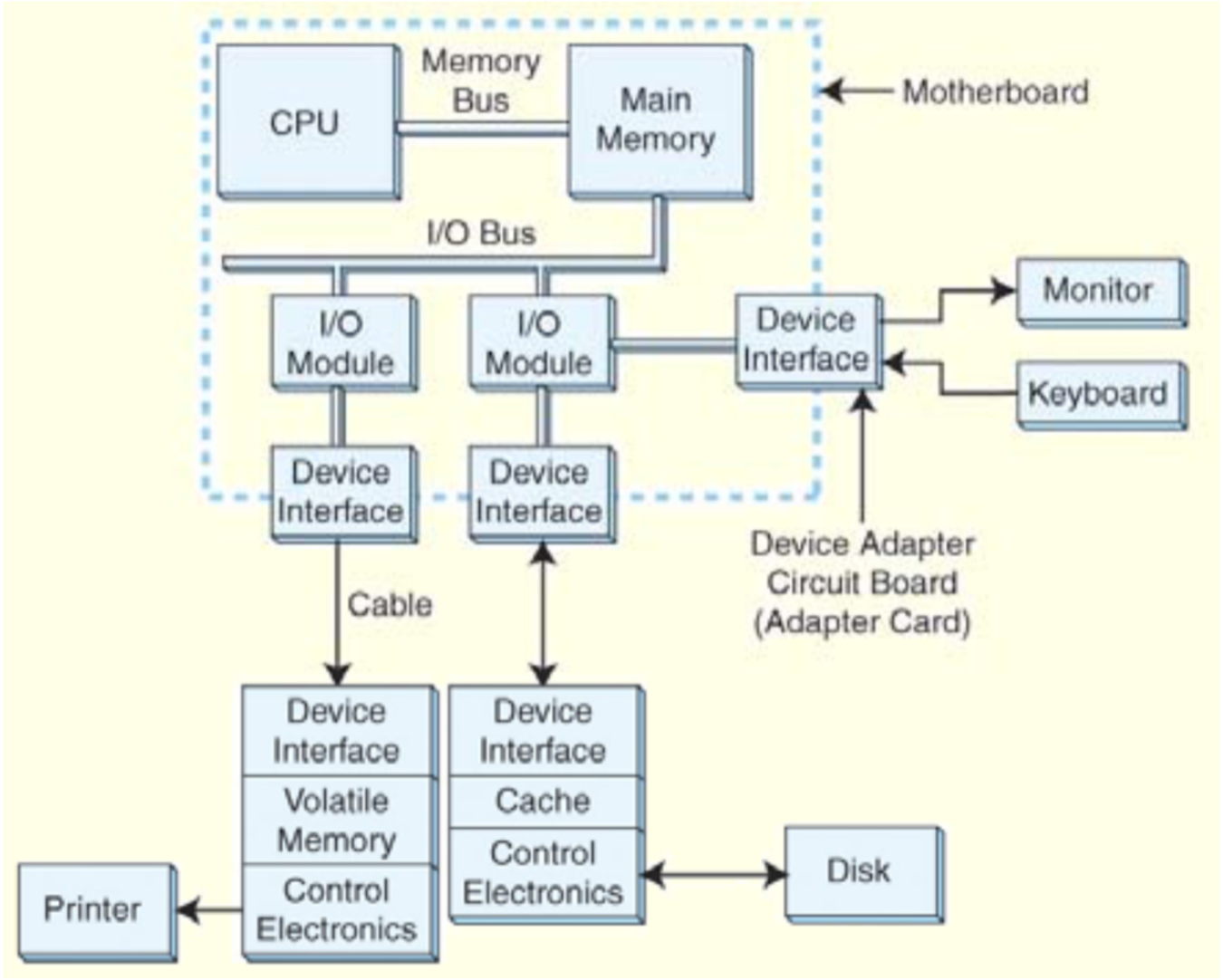
# I/O Subsystem

- A subsystem of components that moves data between external devices and a host system
- I/O subsystems include:
  - Blocks of main memory that are devoted to I/O functions
  - Buses that move data into and out of the system
  - Control modules in the host and in peripheral devices
  - Interfaces to external components such as keyboards and disks
  - Cabling or communications links between the host system and its peripherals

# Evolution



# I/O Architecture

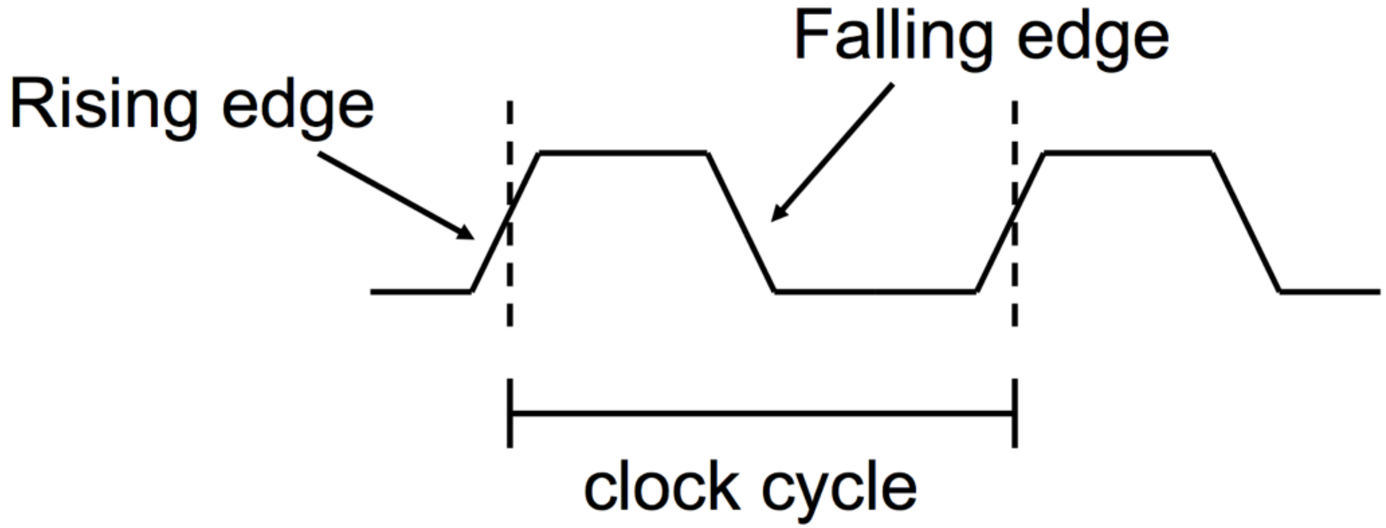


# A Bus

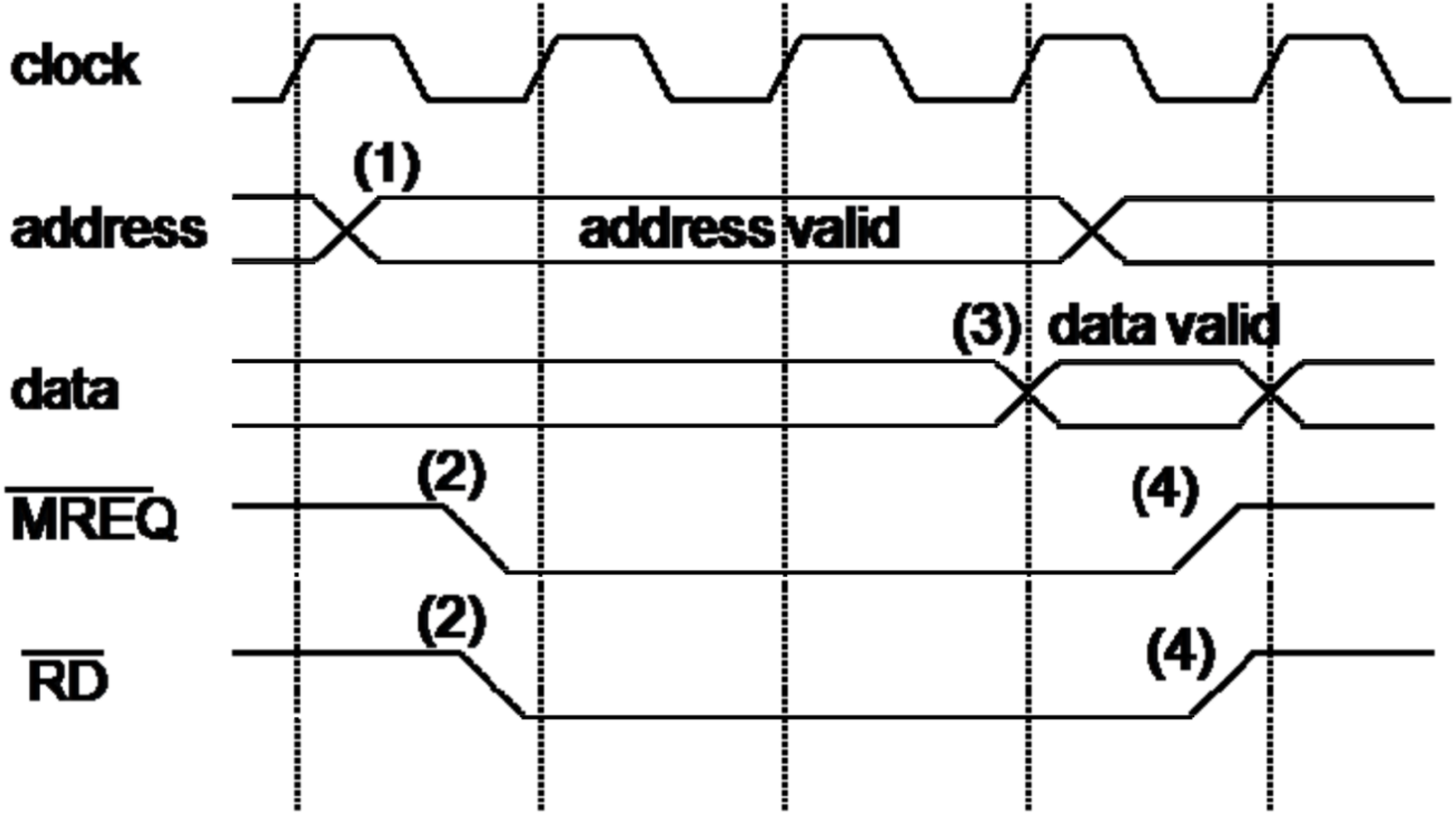
- A bus is a common pathway that interconnects a number of devices
  - Consists of physical parts (connectors, wires) and a bus protocol
  - Eliminates the need to connect each device to every other device
- One device sends data at any given time, typically one device receives (but all listen)
- One device serves as a bus master, with the remaining devices as slaves (roles change at different times)
- Synchronous and asynchronous bus architectures

# Bus Clock

- Clock signal may be used to synchronize bus operations – Generally derived from the master clock (but may be slower)
- It may take several clock cycles to perform a bus transaction (such as a memory read): a bus cycle



# Synchronous Bus Example



# Synchronous Bus Example (cont'd)

- The CPU wants to perform a memory read

1. CPU places the address it wants to read from onto the address lines of the bus
2. CPU asserts MREQ and RD lines
    1. MREQ tells the memory it is selected for the operation
    2. RD specifies that a Read operation is desired
3. After a fixed number of clock cycles, CPU reads the data from the data lines of the bus
4. CPU de-asserts MREQ and RD, releasing the bus

# CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

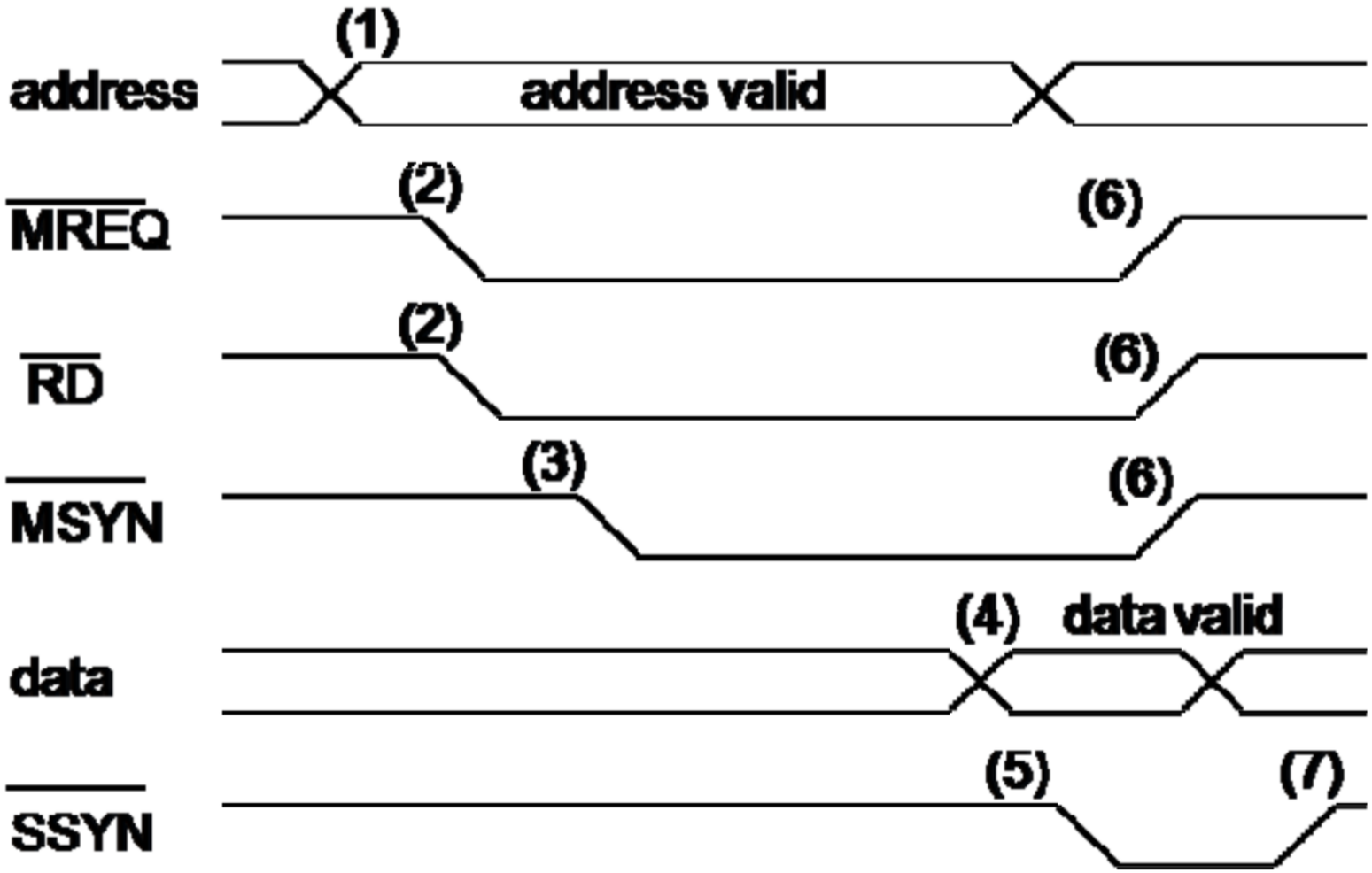
- Enumerate the components of an I/O subsystem
- Identify the motivation for a bus and describe the operation of synchronous buses

If you have any difficulties, please review the lecture video before continuing.

# Asynchronous Bus

- Does not use the bus clock
  - More flexible: a faster master/slave pair responds faster than a slower master/slave pair
- Master places control, data, address on the bus and then asserts master synchronization (MSYN)
- Slave performs its job then asserts slave synchronization (SSYN) when done
- Master then de-asserts MSYN Slave then de-asserts SSYN

# Asynchronous Bus Example



# Asynchronous Bus Example (cont'd)

- Again, the CPU wants to perform a memory read:

1. CPU places the address it wants to read from onto the address lines of the bus
2. CPU asserts MREQ and RD lines
3. CPU (master) asserts MSYN
4. Memory places data on bus data lines
5. Memory (slave) asserts SSYN
6. Master de-asserts MSYN (also MREQ and RD)
7. Slave de-asserts SSYN

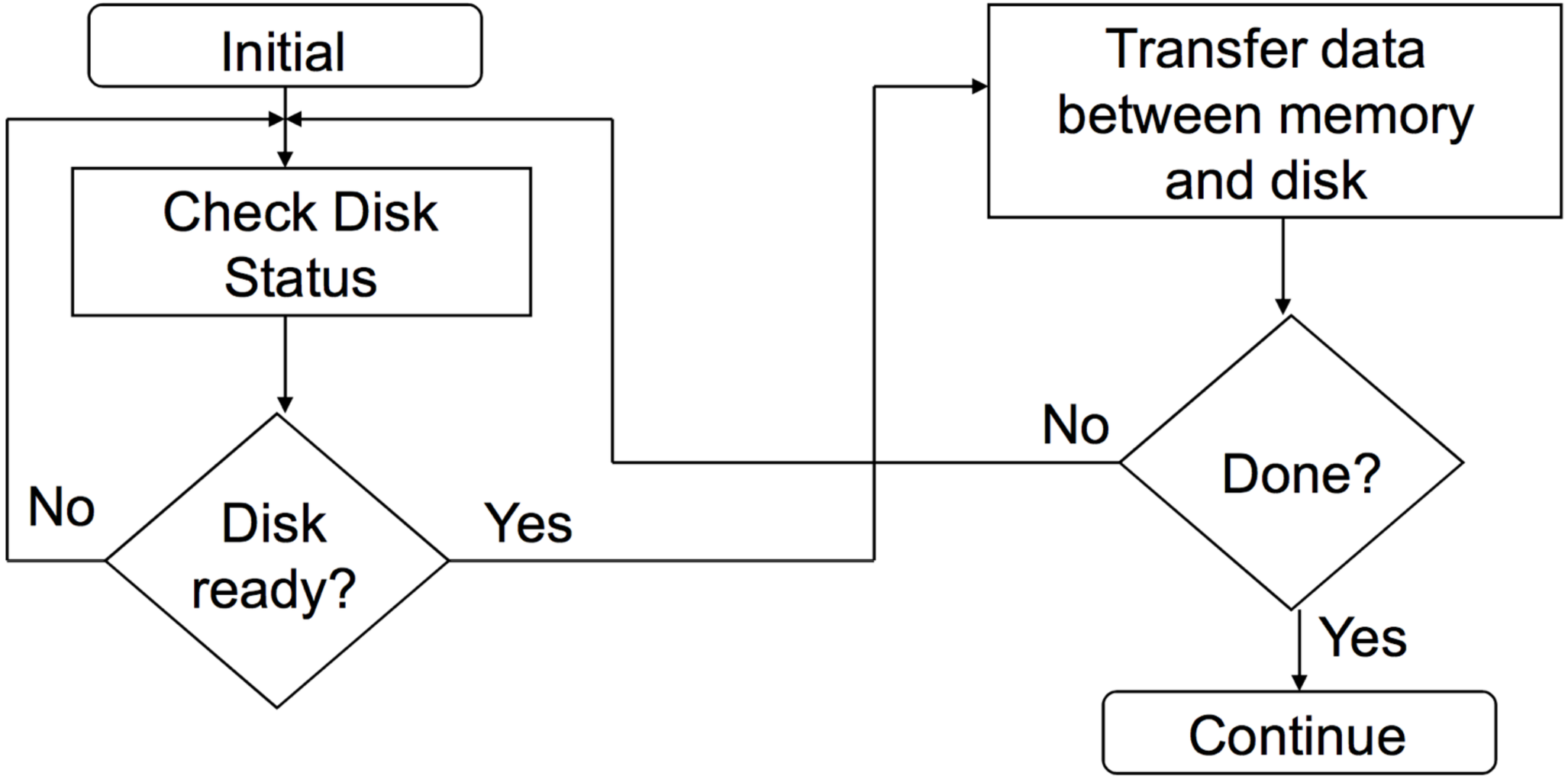
# Methods for Managing I/O

- Programmed I/O
  - Reserves a register for each I/O device
  - Each register is continually polled to detect data arrival
- Interrupt-Driven I/O
  - Allows the CPU to do other things until I/O is requested
- Direct Memory Access (DMA)
  - Offloads I/O processing to a special-purpose chip that takes care of the details.
- Channel I/O
  - Uses dedicated I/O processors

# Programmed I/O

- CPU polls each device in some order to determine whether it requires servicing
  - If device is busy, CPU checks status continuously until device becomes available (busy-wait)
- Simplest, least expensive scheme
- Poor resource utilization – processor may spend considerable time in busy-wait cycles
- High-priority devices are not checked until CPU is finished with current task
  - Only one device active at a time
- Mismatch between speed of CPU and of I/O devices

# Programmed I/O Example



# CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

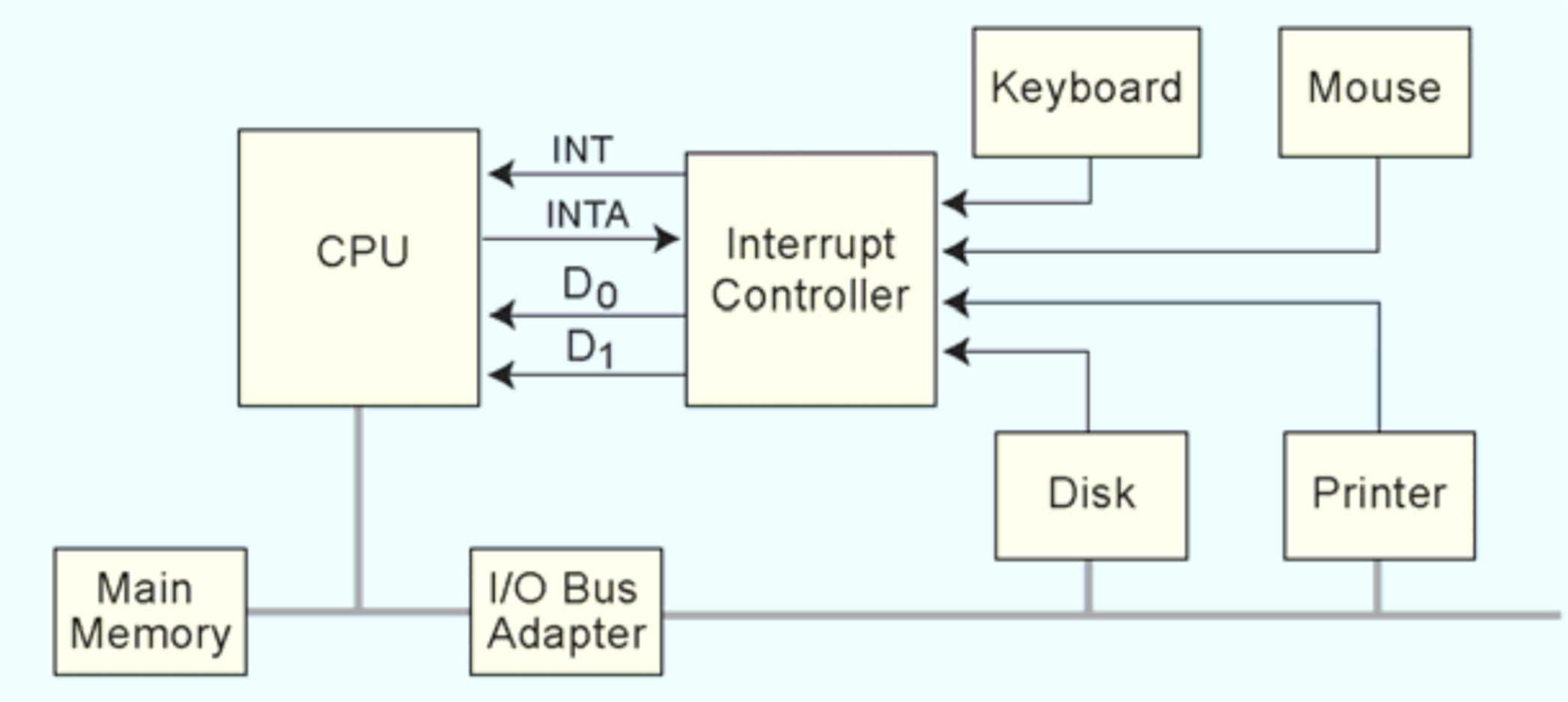
- Identify the motivation for a bus and describe the operation of asynchronous buses
- Describe the operation of programmed I/O

If you have any difficulties, please review the lecture video before continuing.

# Interrupt-driven I/O

- CPU only accesses a device when it needs servicing (no busy-wait cycles)
- Interrupt line used to catch the attention of the CPU
- Stylized algorithms for handling interrupts (change little from one architecture to another)
- Interrupt Flag (IF) in the processor status register determines whether a CPU is accepting interrupts
- Priorities are assigned to interrupts
    - Non-maskable interrupt for handling potentially catastrophic events

# Interrupt-driven Architecture



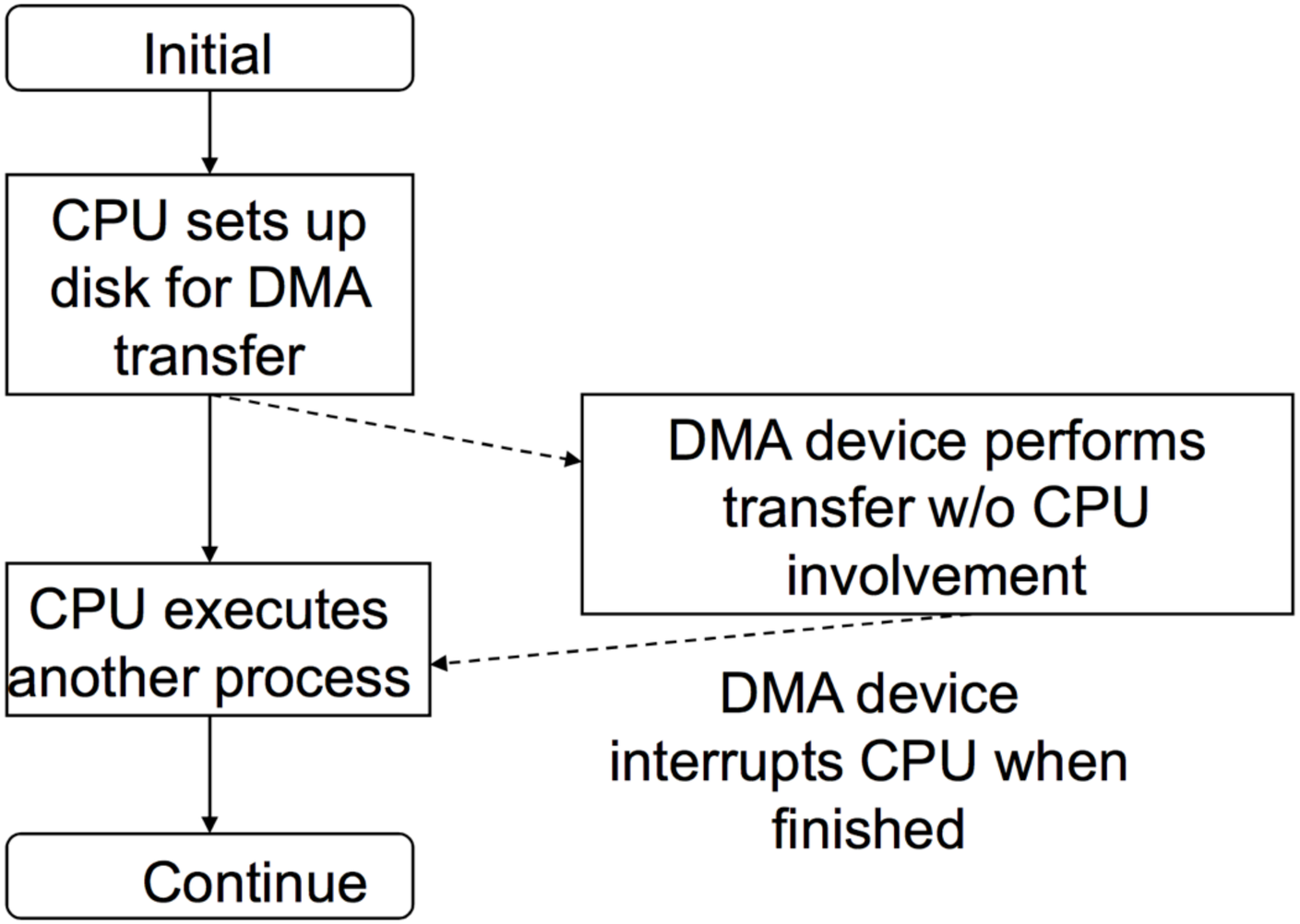
# Processing an Interrupt

- When an interrupt is received, the processor status register and the program counter are pushed onto the stack
- Program counter is loaded with the address of appropriate interrupt service routine (ISR)
- Processor status register contains the IF
    - Must disable interrupt at least for the duration of the first instruction of the ISR
- Processor executes the interrupt
- Execution of interrupted program resumes

# Direct Memory Access (DMA)

- Devices may transfer data to/from memory without the intervention of the processor
- Implemented through the use of a DMA controller (a special-purpose processor)
- Cycle-stealing DMA – DMA controller acquires the bus, transfer a single word and relinquishes the bus
    - Allows CPU (and other devices) to use the bus during a DMA transfer

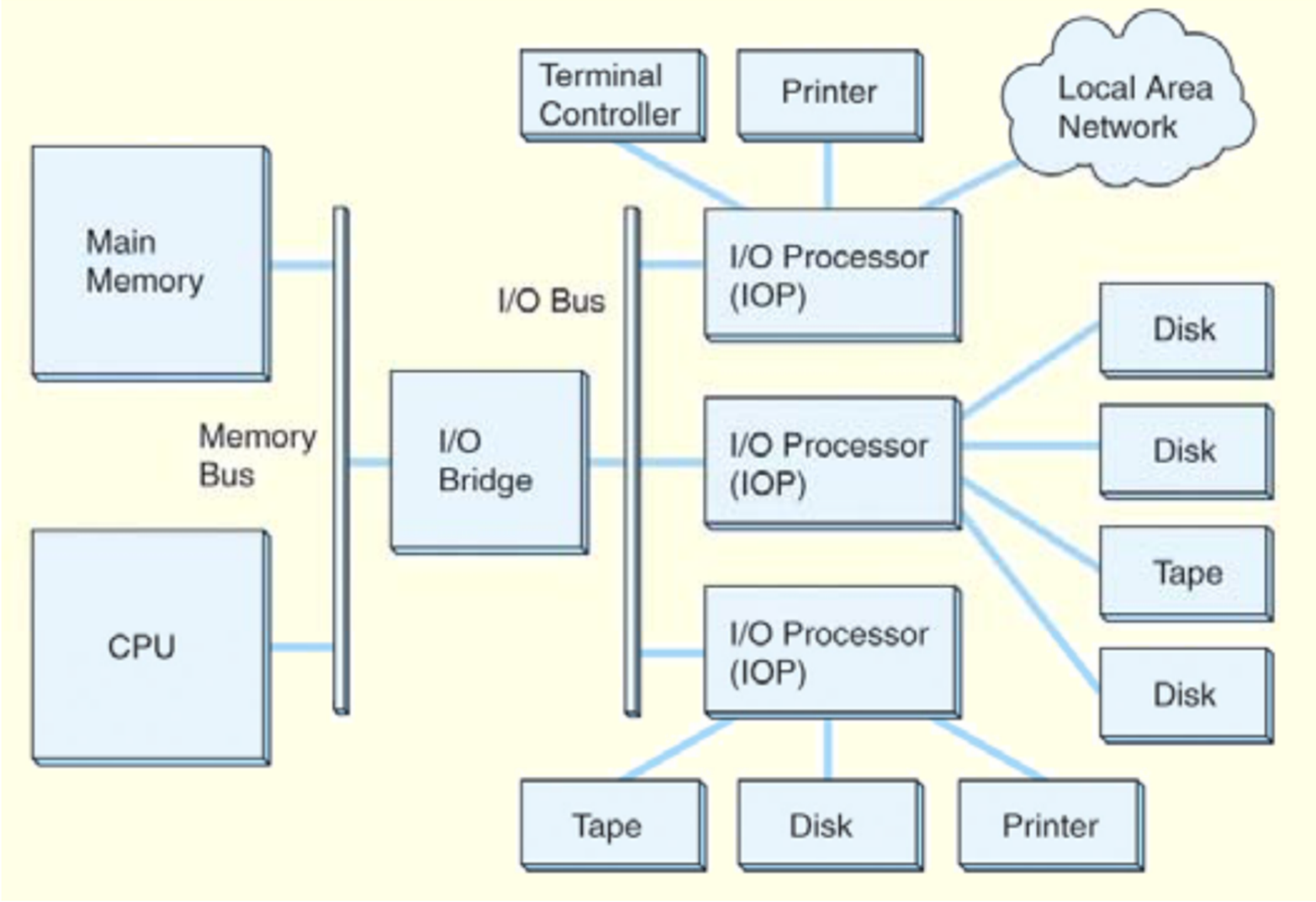
# DMA Example



# Channel I/O

- Channel I/O consists of one or more I/O processors (IOPs) that control various channel paths
- Slower devices such as terminals and printers are combined (multiplexed) into a single faster channel
- Channel I/O is distinguished from DMA by the intelligence of the IOPs
- The IOP negotiates protocols, issues device commands, translates storage coding to memory coding, and can transfer entire files or groups of files independently of the host CPU

# Channel I/O Architecture



# CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- Describe the operation of interrupt-driven I/O, Direct Memory Access, and channel I/O

If you have any difficulties, please review the lecture video before continuing.

# Summary

- A bus is a common pathway that interconnects a number of devices, eliminating the need to connect each device to all others
- Buses can operate synchronously or asynchronously
- In programmed I/O the CPU polls each device in order to determine whether it needs servicing
- In interrupt-driven I/O the CPU only accesses the device when it needs servicing
- In DMA and channel I/O devices may transfer data to/from memory without requiring intervention by the CPU

MODULE 8: Input/Output Systems

# Lecture 8.2
# Input/Output Control

Prepared By:
- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering
Virginia Tech