CS 222 Computer Organization & Architecture

Lecture 35 [30.04.2019]

I/O Data Transfer techniques

John Jose

Assistant Professor

Department of Computer Science & Engineering

Indian Institute of Technology Guwahati, Assam.

I/O Data Transfer techniques

- Programmed I/O
- Interrupt-Driven I/O
- Direct Memory Access (DMA)

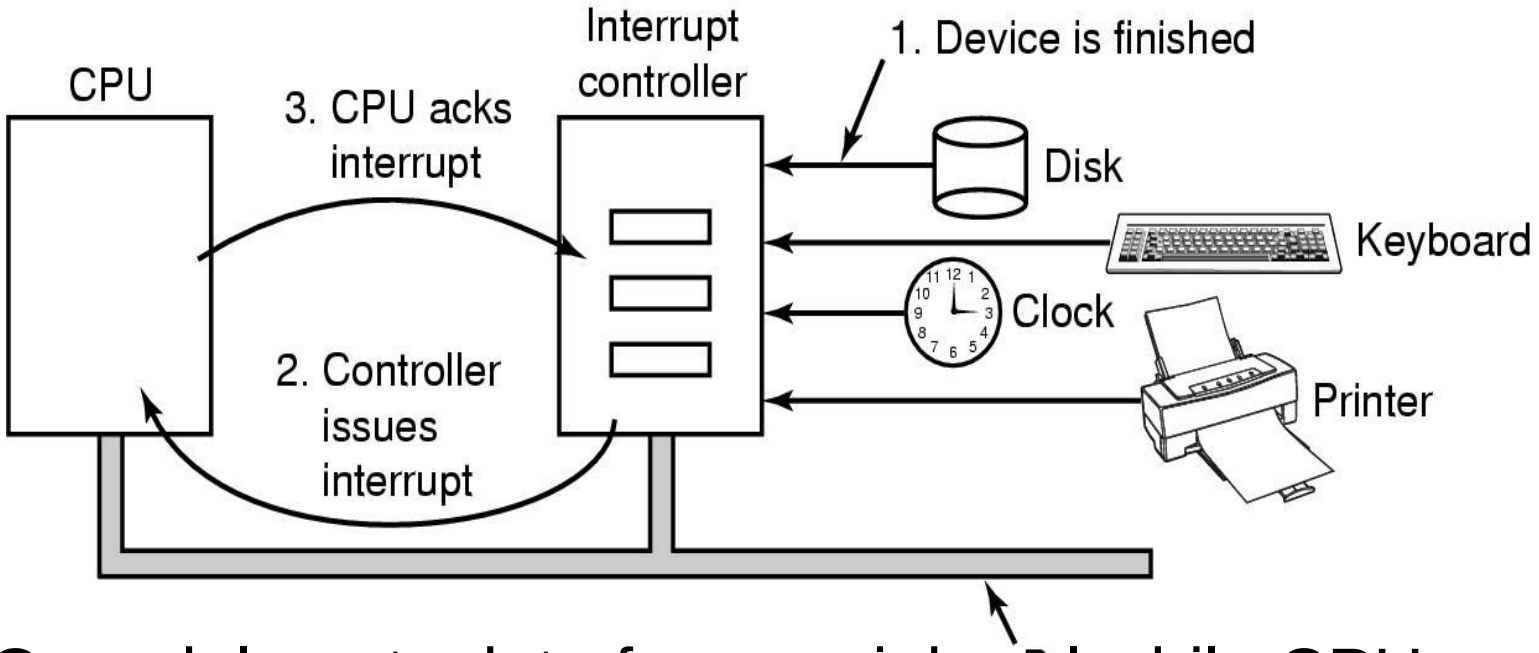
Programmed I/O

- CPU periodically check I/O status (polling)
  - If device ready, do operation
  - If error, take action
- CPU has direct control over I/O
  - Sensing status
  - Read/write commands
  - Transferring data
- CPU waits for I/O module to complete operation
- Wastes CPU time

Steps in Programmed I/O

- CPU requests I/O operation
- I/O module performs operation
- I/O module sets status bits
- CPU checks status bits periodically (polling)
- CPU may wait or come back later
- I/O module does not inform CPU directly
- I/O module does not interrupt CPU

Interrupt Driven I/O



- I/O module gets data from peripheral while CPU continues other work.
- I/O module interrupts CPU when data is ready.
- CPU requests data
- I/O module transfers data

Interrupt Driven I/O

- I/O device issues an interrupt to indicate that it needs attention of CPU
- Interrupts are special signals initialed by I/O devices to catch the attention of the processor.
- Overcomes CPU waiting
- No repeated CPU checking of device
- An I/O interrupt is asynchronous w.r.t. instruction execution
- Is not associated with any instruction so doesn't prevent any instruction from completing

Interrupt Driven I/O

- Advantages of using interrupts
  - Relieves the processor from having to continuously poll for an I/O event; user program progress is only suspended during the actual transfer of I/O data to/from user memory space
- Disadvantage – special hardware is needed to
  - Indicate the I/O device causing the interrupt and to save the necessary information prior to servicing the interrupt and to resume normal processing after servicing the interrupt

Challenges in Interrupt Driven I/O

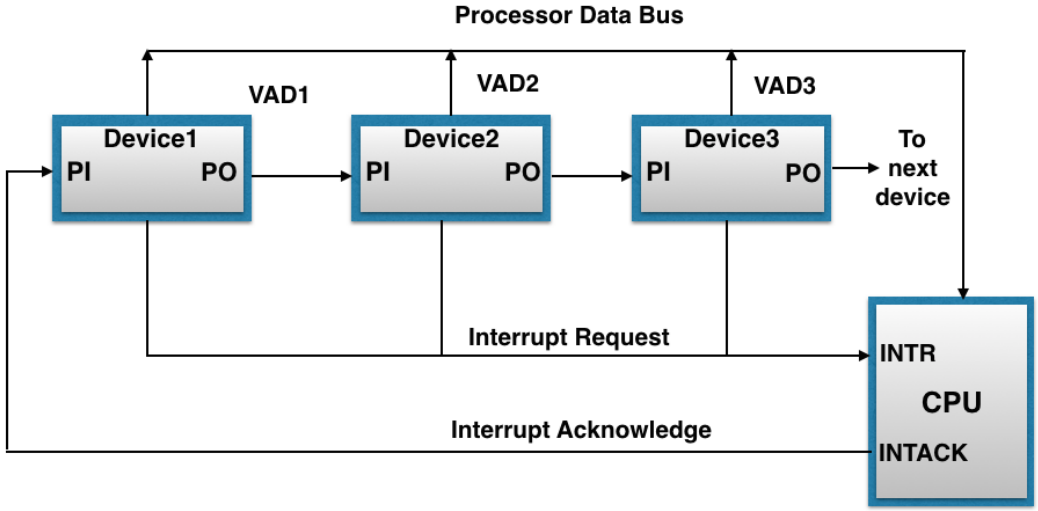
- How do you identify the module issuing the interrupt?
  - Need a way to identify the device generating the interrupt
- How do you deal with multiple interrupts?
  - Can have different urgencies (so need a way to prioritize them)

Identifying Interrupting Module

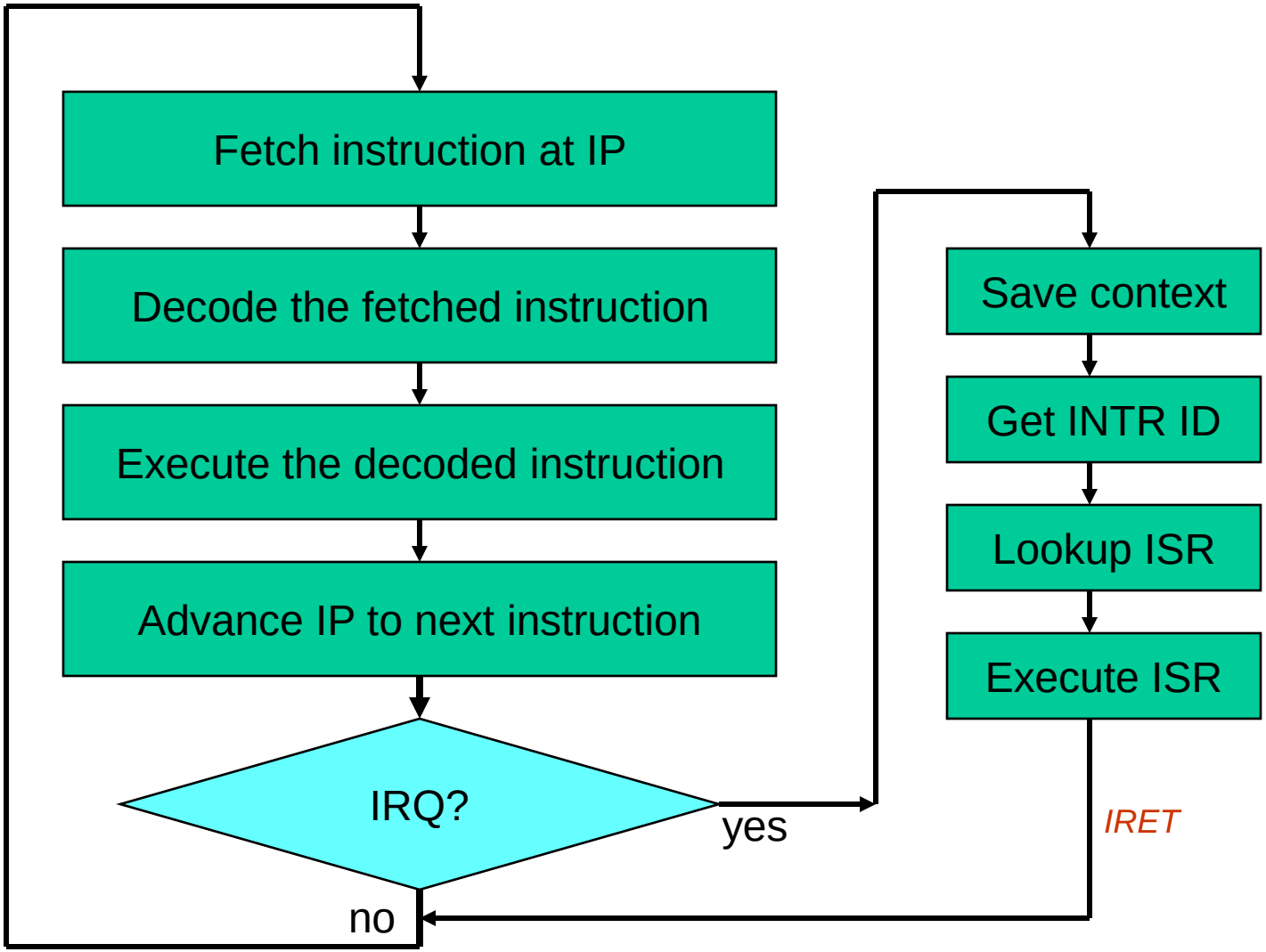
- Different line for each module
  - Limits number of devices
- Software poll
- Bus Master
  - Module must claim the bus before it can raise interrupt

Identifying Interrupting Module

- CPU asks each module in turn (Slow) Daisy Chain or Hardware poll
  - Interrupt Acknowledge sent down a chain
  - Module responsible places vector on bus
  - CPU uses vector to identify handler routine



Interrupt Service Routine



Direct Memory Access

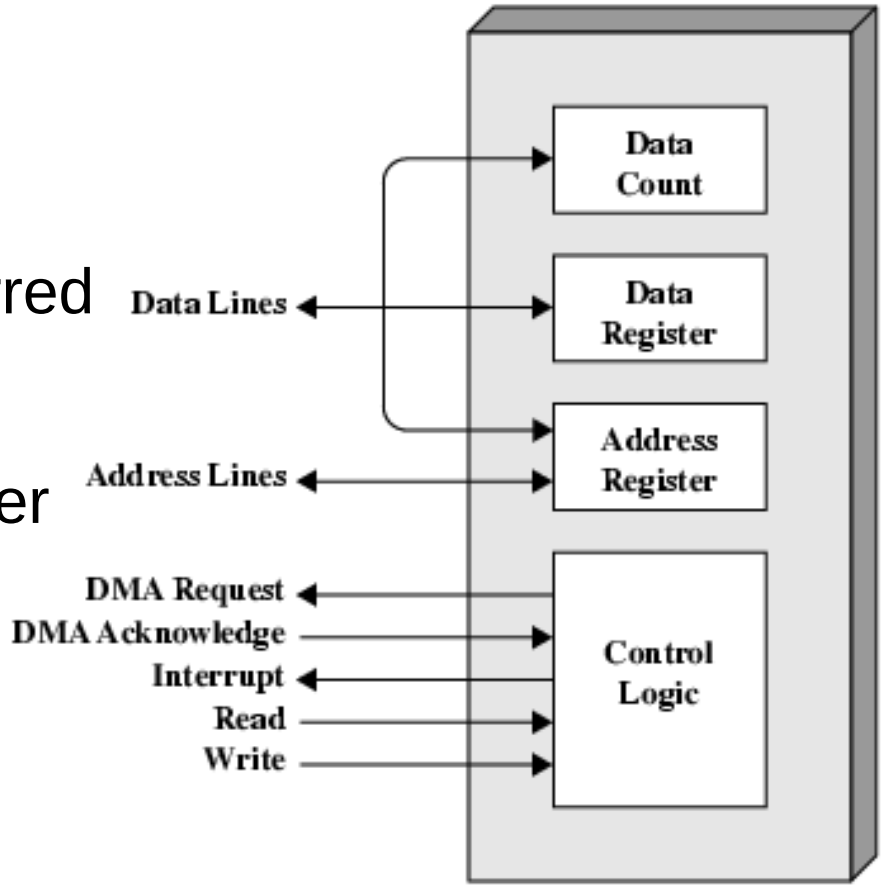
- Interrupt driven and programmed I/O require active CPU intervention
- For high-bandwidth devices (like disks) interrupt-driven I/O would consume a lot of processor cycles
- DMA is an additional module (hardware) on bus
- DMA controller takes over from CPU for I/O operations

Direct Memory Access

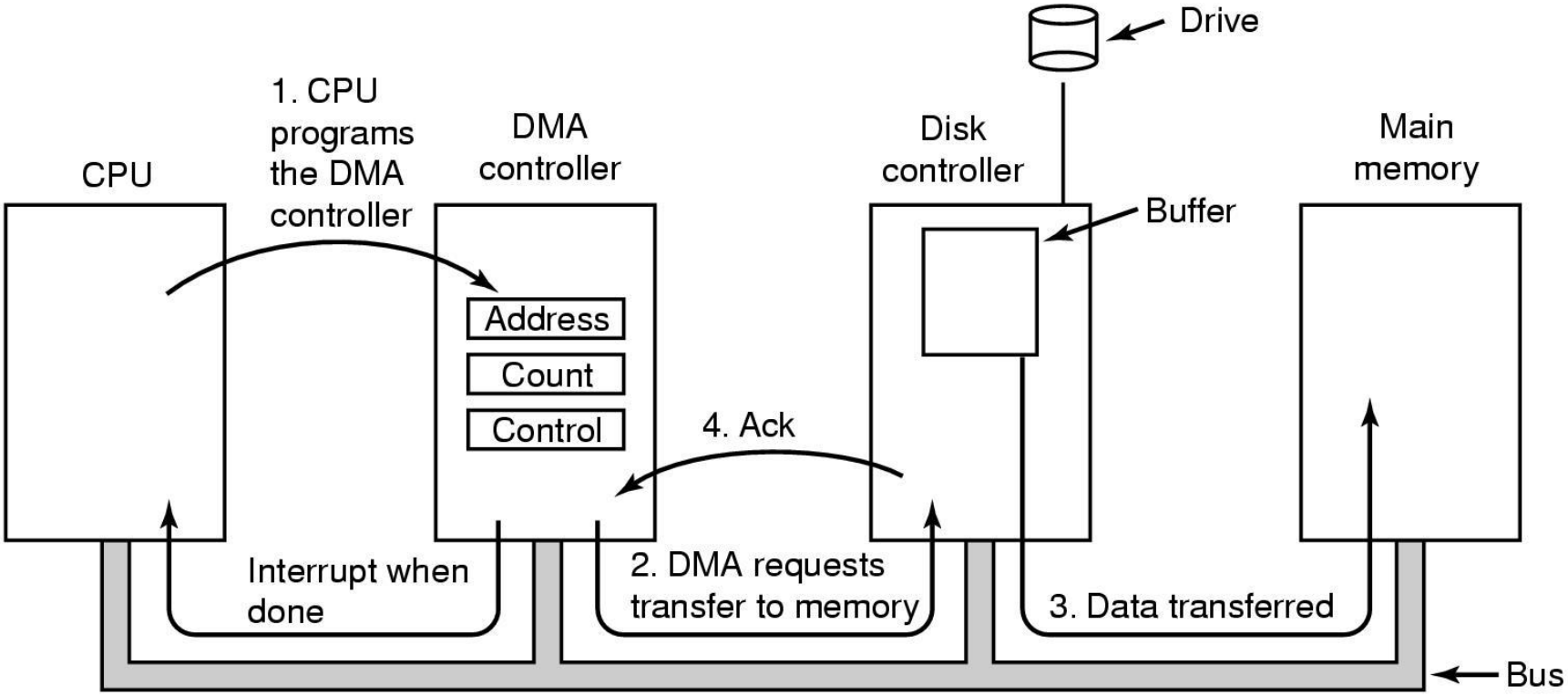
- DMA controller has the ability to transfer large blocks of data directly to/from the memory without involving the processor.
  - The processor initiates the DMA transfer by supplying the I/O device address, the operation to be performed, the memory address destination/source, the number of bytes to transfer
  - The DMA controller manages the entire transfer (possibly thousand of bytes in length), arbitrating for the bus
  - When the DMA transfer is complete, the DMA controller interrupts the processor to let it know that the transfer is complete

DMA Module Diagram

- CPU tells DMA controller:-
  - Read/Write
  - Device address
  - Starting address of memory block for data
  - Amount of data to be transferred
- CPU carries on with other work
- DMA controller deals with transfer
- DMA controller sends interrupt when finished



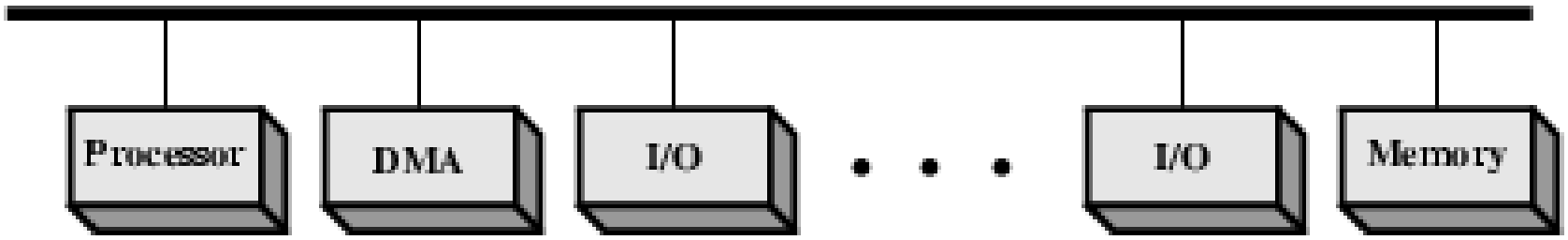
Disk to Memory Copy via DMA



Modes of DMA operation

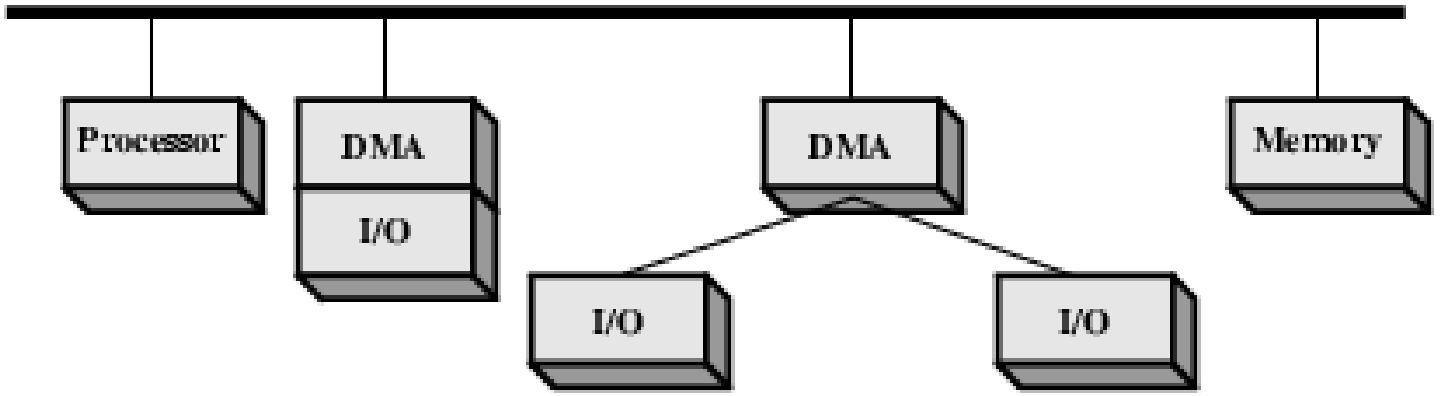
- **Cycle Stealing**
  - DMA controller acquires control of bus
  - Transfers a single word and releases the bus
  - The CPU is slowed down due to bus contention
  - Responsive but not very efficient
- **Burst Mode**
  - DMA Controller acquires control of bus
  - Transfers all the data and then only releases the bus
  - The CPU is suspended or works with cache
  - Efficient but interrupts may not be serviced in a timely way

DMA Configurations



- Single Bus, Detached DMA controller
- Each transfer uses bus twice
  - I/O to DMA then DMA to memory
- CPU is suspended twice

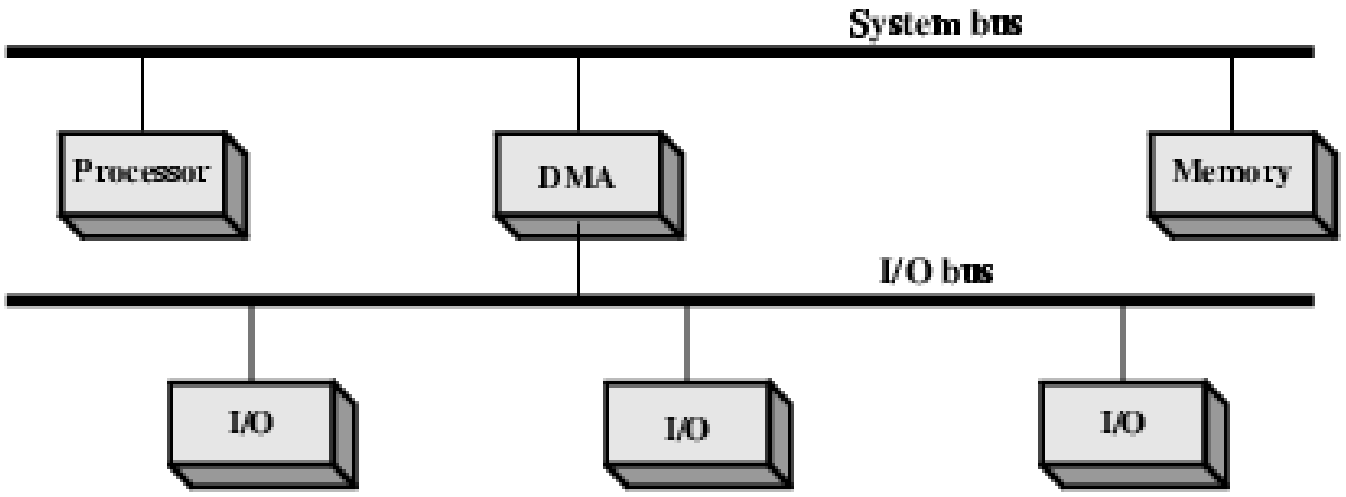
DMA Configurations



(b) Single-bus, Integrated DMA-I/O

- Single Bus, Integrated DMA controller
- Controller may support >1 device
- Each transfer uses bus once
  - DMA to memory
- CPU is suspended once

DMA Configurations



(c) I/O bus

- Separate I/O Bus
- Bus supports all DMA enabled devices
- Each transfer uses bus once
  - DMA to memory
- CPU is suspended once

johnjose@iitg.ac.in
http://www.iitg.ac.in/johnjose/