William Stallings
Computer Organization and Architecture
7th Edition

Chapter 7

Input/Output

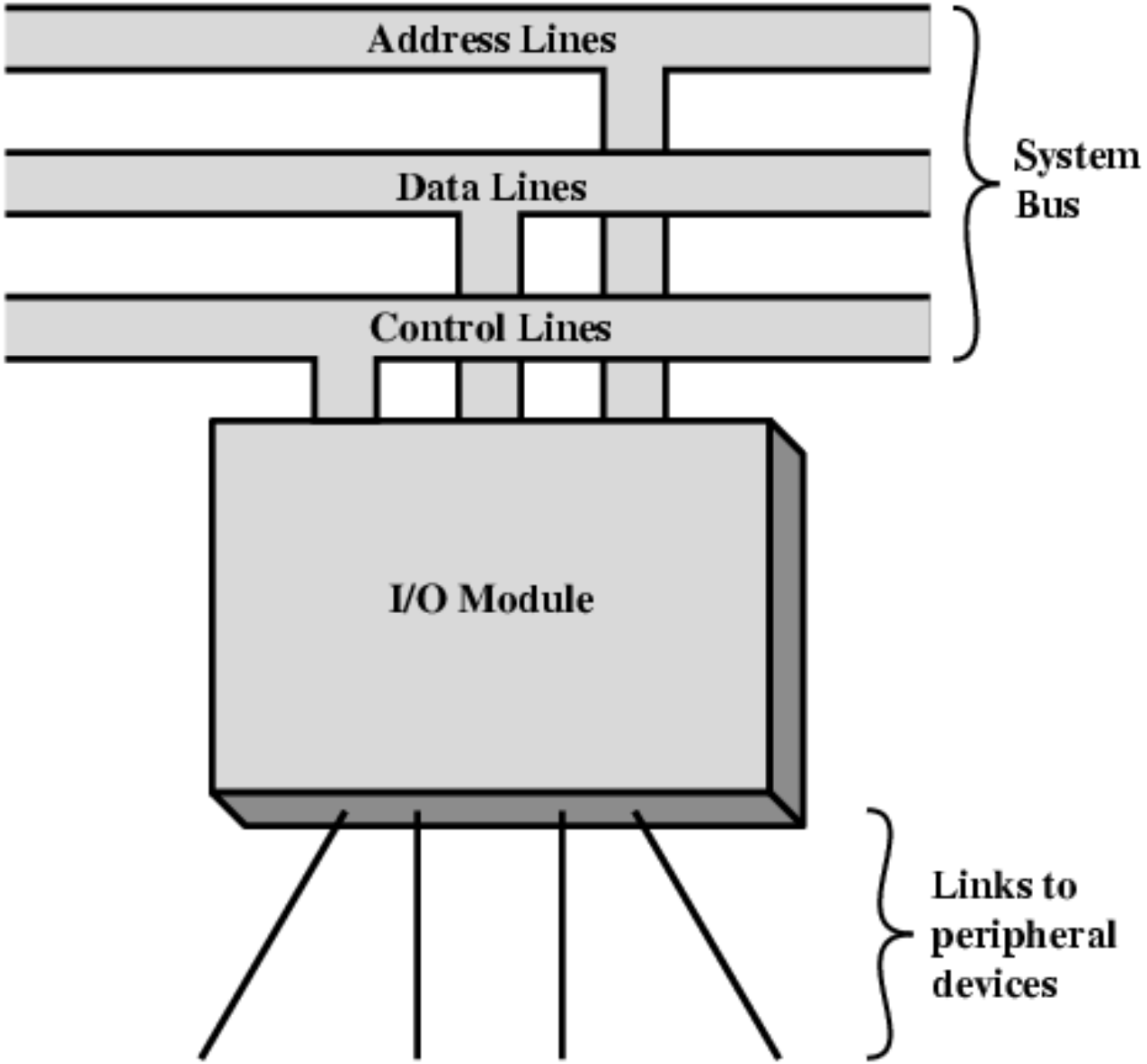
Input/Output Problems

- Wide variety of peripherals
  - Delivering different amounts of data
  - At different speeds
  - In different formats
- All slower than CPU and RAM
- Need I/O modules

Input/Output Module

- Interface to CPU and Memory
- Interface to one or more peripherals

Generic Model of I/O Module



I/O Module Functions

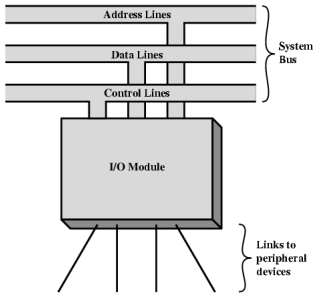
- Control and timing
- Processor communication
- Device communication
- Data buffering
- Error detection

External Devices

- Human readable
  - —Screen, printer, keyboard
- Machine readable
  - —Monitoring and control
- Communication
  - —Modem
  - —Network Interface Card (NIC)

I/O Steps

- CPU checks I/O module device status
- I/O module returns status
- If ready, CPU requests data transfer
- I/O module gets data from device
- I/O module transfers data to CPU
- Variations for output, DMA, etc.



Input Output Techniques

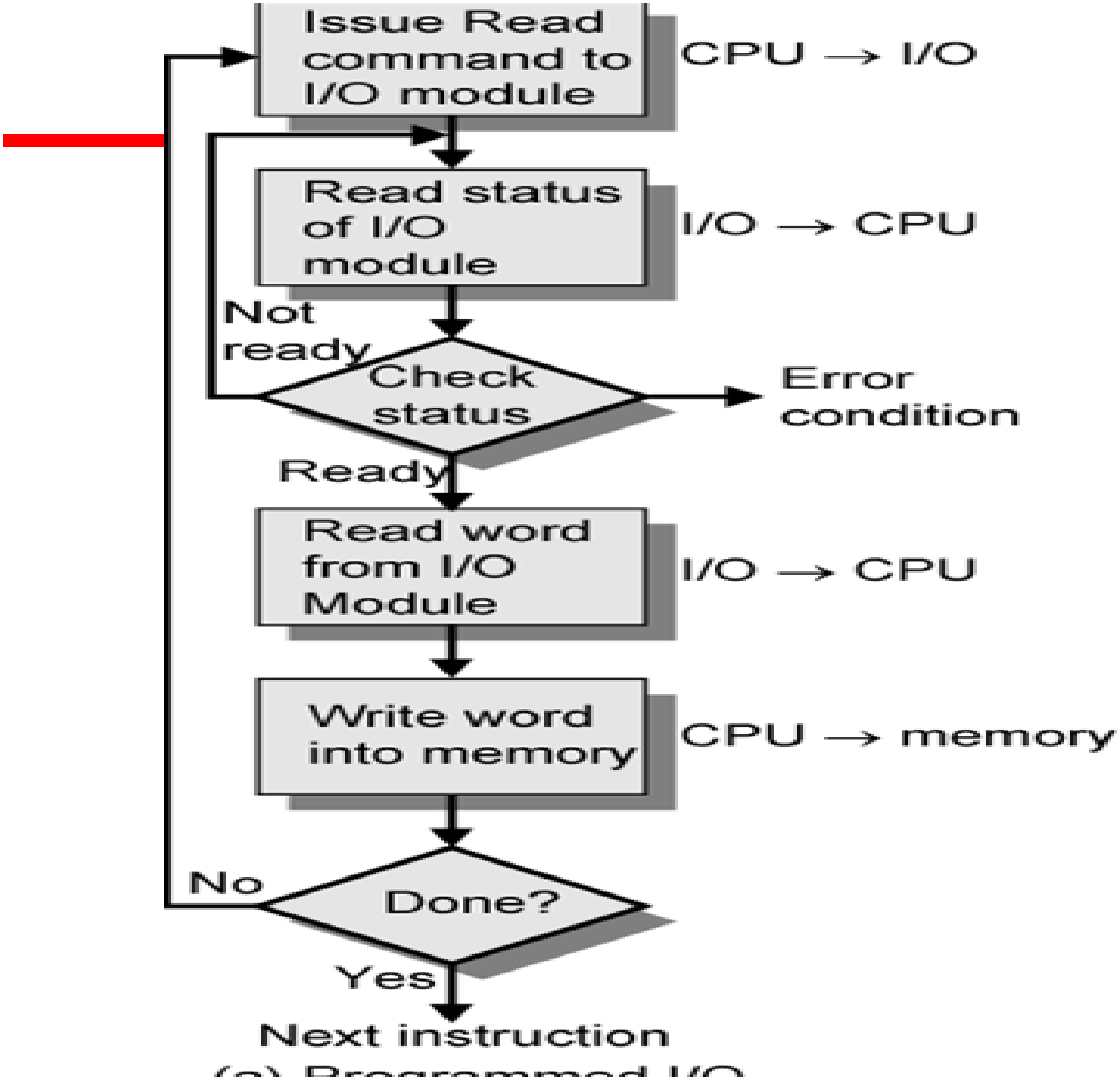
- Programmed
- Interrupt driven
- Direct Memory Access (DMA)

PROGRAMMED I/O

- Data transfer operations are completely controlled by **CPU** i.e. CPU executes a program that
  - initiates,
  - directs and
  - terminates the I/O operation

PROGRAMMED I/O

- Useful where h/w costs need to be minimized.
- Entire I/O is handled by CPU
- **STEPS**
  - 1. Read I/O devices status bit
  - 2. Test status bit to determine if device is ready
  - 3. If device not ready return to step 1
  - 4. During interval when device is not ready CPU simply wastes its time until device is ready



(a) Programmed I/O

Programmed I/O

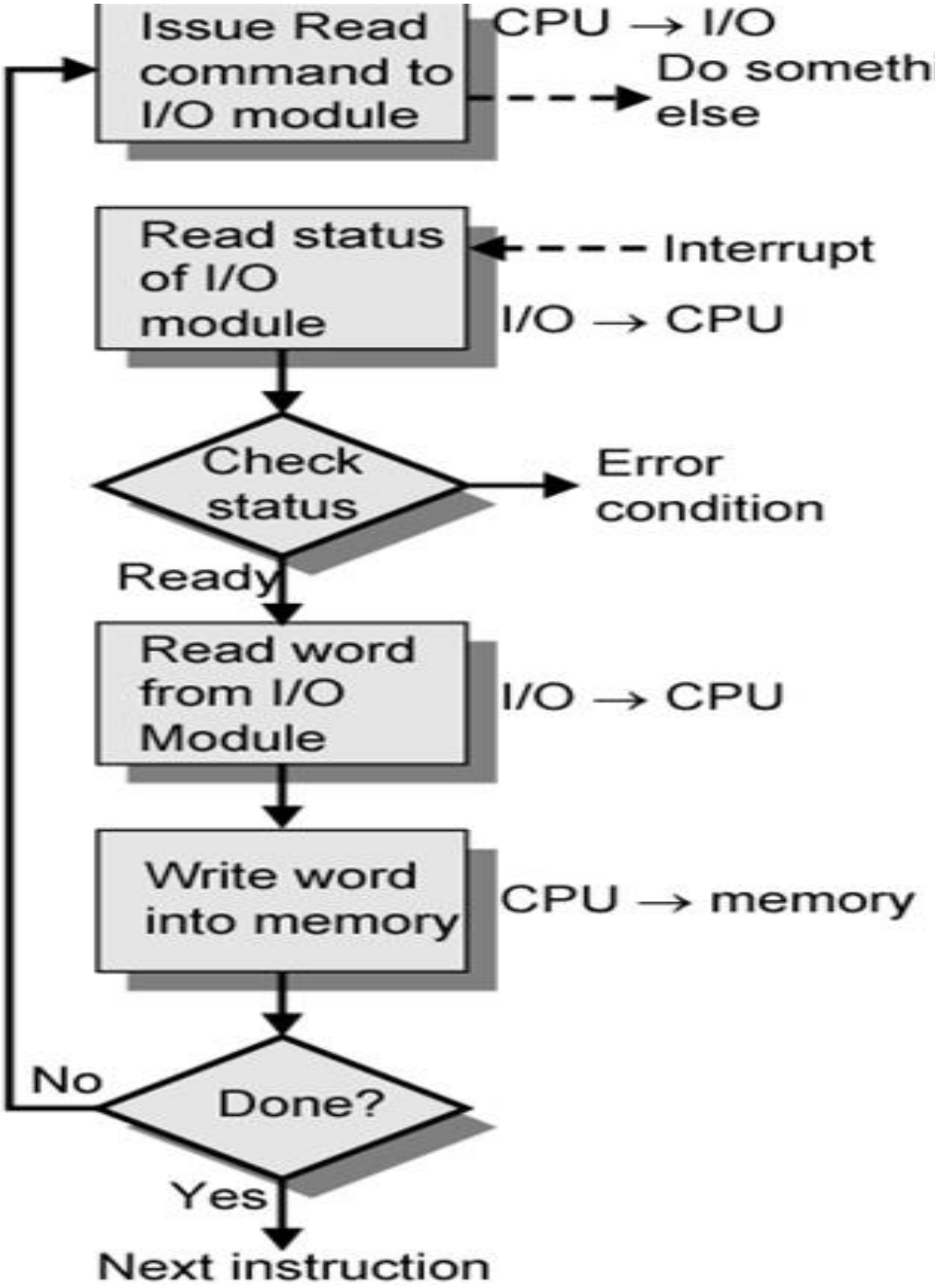
- CPU has direct control over I/O
  - Sensing status
  - Read/write commands
  - Transferring data
- CPU waits for I/O module to complete operation
- Wastes CPU time

Programmed I/O - detail

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

INTERRUPT DRIVEN I/O

- Major drawback of programmed I/O is **busy waiting**
- Speed of I/O devices is much slower than CPU
- Performance of CPU goes down as it has to repeatedly check for device status
- **Solution**: Switch to another task if device is not ready and thus use interrupt mechanism


Issue Read command to I/O module
CPU → I/O
Do something else
Read status of I/O module
Interrupt
I/O → CPU
Check status
Error condition
Ready
Read word from I/O Module
I/O → CPU
Write word into memory
CPU → memory
No
Done?
Yes
Next instruction

Interrupt Driven I/O

- Overcomes CPU waiting
- No repeated CPU checking of device
- I/O module interrupts when ready

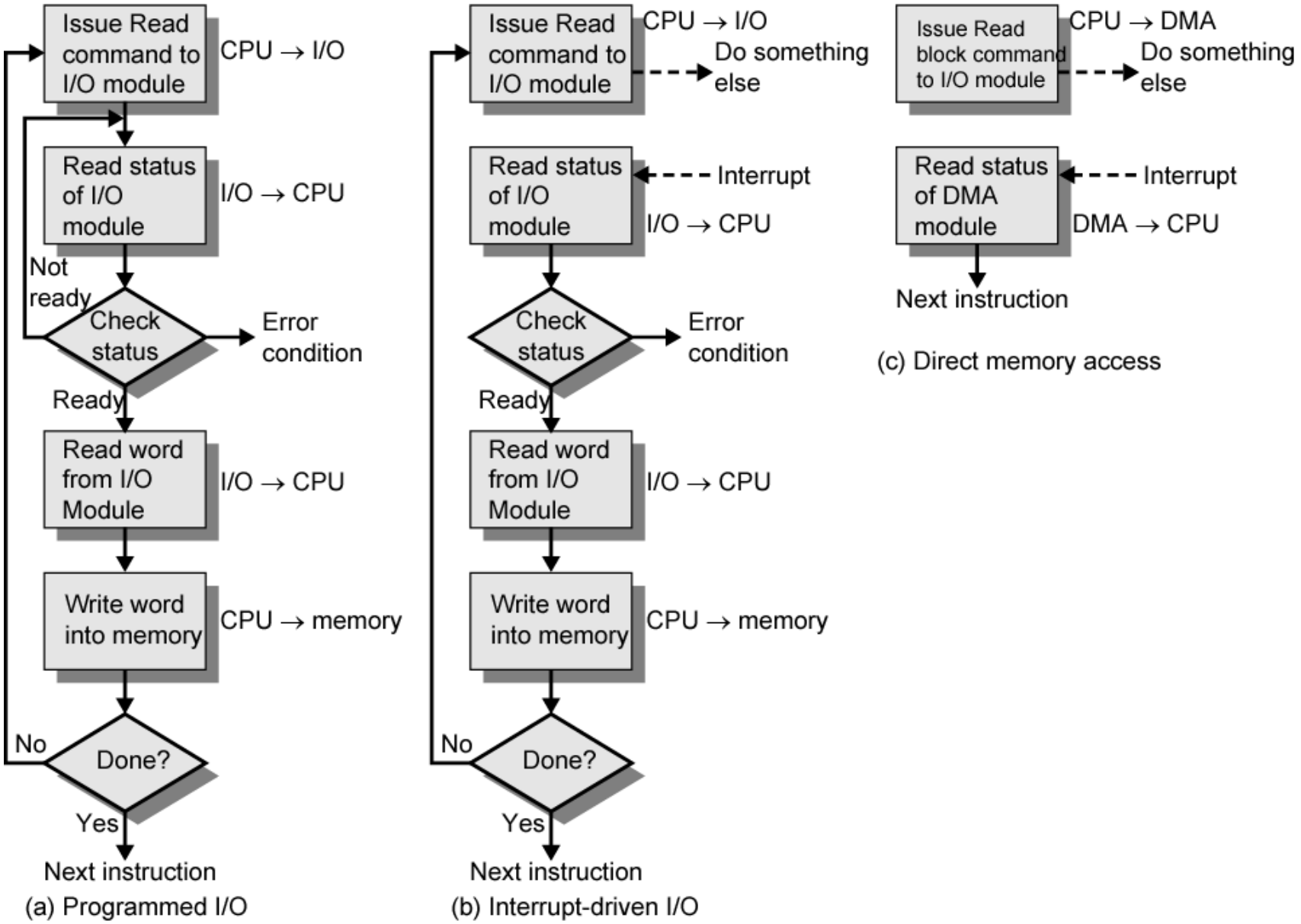
Interrupt Driven I/O
Basic Operation

- CPU issues read command
- I/O module gets data from peripheral whilst CPU does other work
- I/O module interrupts CPU
- CPU requests data
- I/O module transfers data

CPU Viewpoint

- Issue read command
- Do other work
- Check for interrupt at end of each instruction cycle
- If interrupted:-
  - Save context (registers)
  - Process interrupt
    - Fetch data & store

Three Techniques for Input of a Block of Data



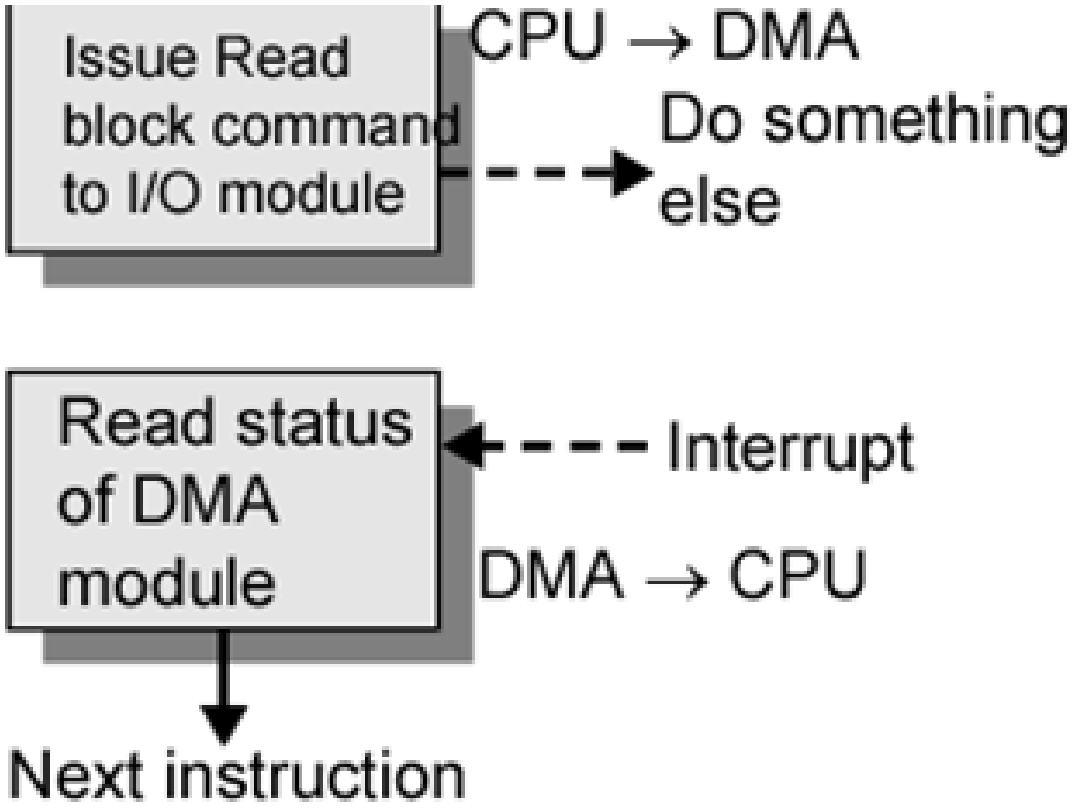
Direct Memory Access

- Interrupt driven and programmed I/O require active CPU intervention
  - —Transfer rate is limited
  - —CPU is tied up

- Direct memory access (**DMA**) is a feature of computer systems that allows certain hardware subsystems to access main system memory (RAM) independently of the central processing unit (CPU).

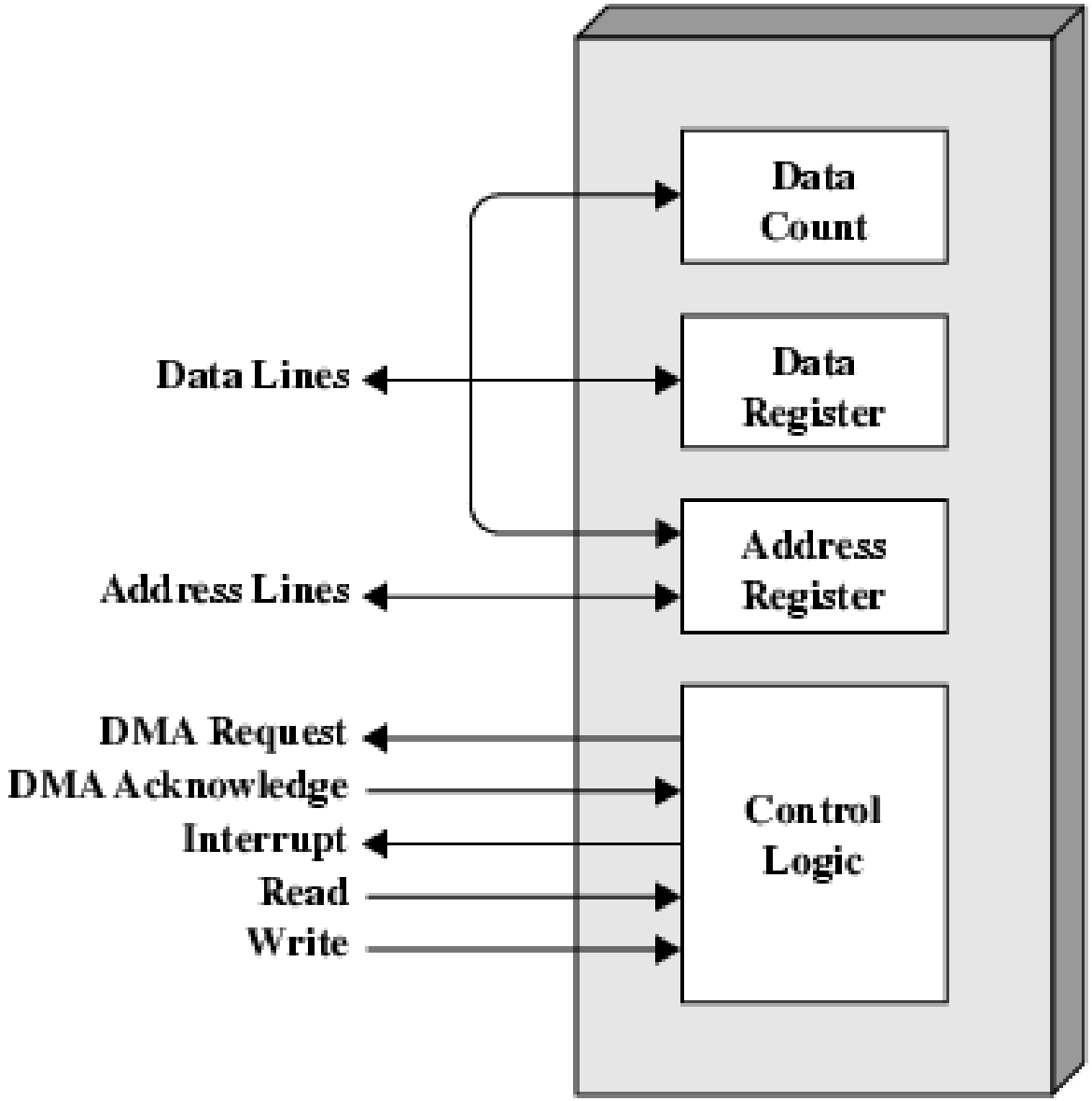
DMA Function

- Additional Module (hardware) on bus
- DMA controller takes over from CPU for I/O



(c) Direct memory access

Typical DMA Module Diagram



DMA Operation

- 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

DMA data transfer modes:

- Burst Mode
- Cycle Stealing Mode
- Transparent Mode

DMA data transfer modes

1. DMA block transfer/Burst Mode

A block of data of **arbitrary length** is transferred in a single **burst**

- Burst - Temporary high-speed data transmission mode used to facilitate sequential data transfer at maximum throughput.

2. Cycle stealing mode

- DMA controller is allowed to use system bus to transfer **one word of data at a time**, after which it must return control of the bus to the CPU.
- The DMA module uses the system bus only when the **processor does not need it**, or it must force the processor to suspend operation temporarily.
- Referred to as **cycle stealing**, because the DMA module in effect steals a bus cycle.

DMA data transfer modes:

3.Transparent DMA

DMA is allowed to steal only those cycles **when CPU is not using system bus**

Advantages of DMA

- High transfer rates
  - fewer CPU cycles for each transfer.
  - DMA speedups the memory operations by bypassing the involvement of the CPU.
- Work overload on the CPU decreases.

Disadvantages of DMA

- DMA transfer requires a **DMA controller** to carry out the operation
- More **expensive** system
- **Synchronization mechanisms** must be provided in order to avoid accessing **non**-updated information from RAM
  - **Cache coherence problem**