THE ESSENTIALS OF

Computer Organization and Architecture

THIRD EDITION

Linda Null
Julia Lobur

Chapter 7

Input/Output and Storage Systems

Chapter 7 Objectives

- Understand how I/O systems work, including I/O methods and architectures.
- Become familiar with storage media, and the differences in their respective formats.
- Understand how RAID improves disk performance and reliability, and which RAID systems are most useful today.
- Be familiar with emerging data storage technologies and the barriers that remain to be overcome.

7.1 Introduction

- Data storage and retrieval is one of the primary functions of computer systems.
  - One could easily make the argument that computers are more useful to us as data storage and retrieval devices than they are as computational machines.
- All computers have I/O devices connected to them, and to achieve good performance I/O should be kept to a minimum!
- In studying I/O, we seek to understand the different types of I/O devices as well as how they work.

7.2 I/O and Performance

- Sluggish I/O throughput can have a ripple effect, dragging down overall system performance.
  - This is especially true when virtual memory is involved.
- The fastest processor in the world is of little use if it spends most of its time waiting for data.
- If we really understand what's happening in a computer system, we can make the best possible use of its resources.

7.3 Amdahl's Law

- The overall performance of a system is a result of the interaction of all its components.
- System performance is most effectively improved when the performance of the most heavily used components is improved.
- This idea is quantified by Amdahl's Law:

$$S = \frac{1}{(1-f) + \frac{f}{k}}$$

where S is the overall speedup; f is the fraction of work performed by a faster component; and k is the speedup of the faster component.

7.3 Amdahl's Law

- Amdahl's Law gives us a handy way to estimate the performance improvement we can expect when we upgrade a system component.
- On a large system, suppose we can upgrade a CPU to make it 50% faster for $10,000 or upgrade its disk drives for $7,000 to make them 150% faster.
- Processes spend 70% of their time running in the CPU and 30% of their time waiting for disk service.
- An upgrade of which component would offer the greater benefit for the lesser cost?

7.3 Amdahl's Law

- The processor option offers a 30% speedup:

$$f = 0.70, \quad k = 1.5 \qquad S = \frac{1}{(1 - 0.7) + 0.7/1.5}$$

- And the disk drive option gives a 22% speedup:

$$f = 0.30, \quad k = 2.5 \qquad S = \frac{1}{(1 - 0.3) + 0.3/2.5}$$

- Each 1% of improvement for the processor costs $333, and for the disk a 1% improvement costs $318.

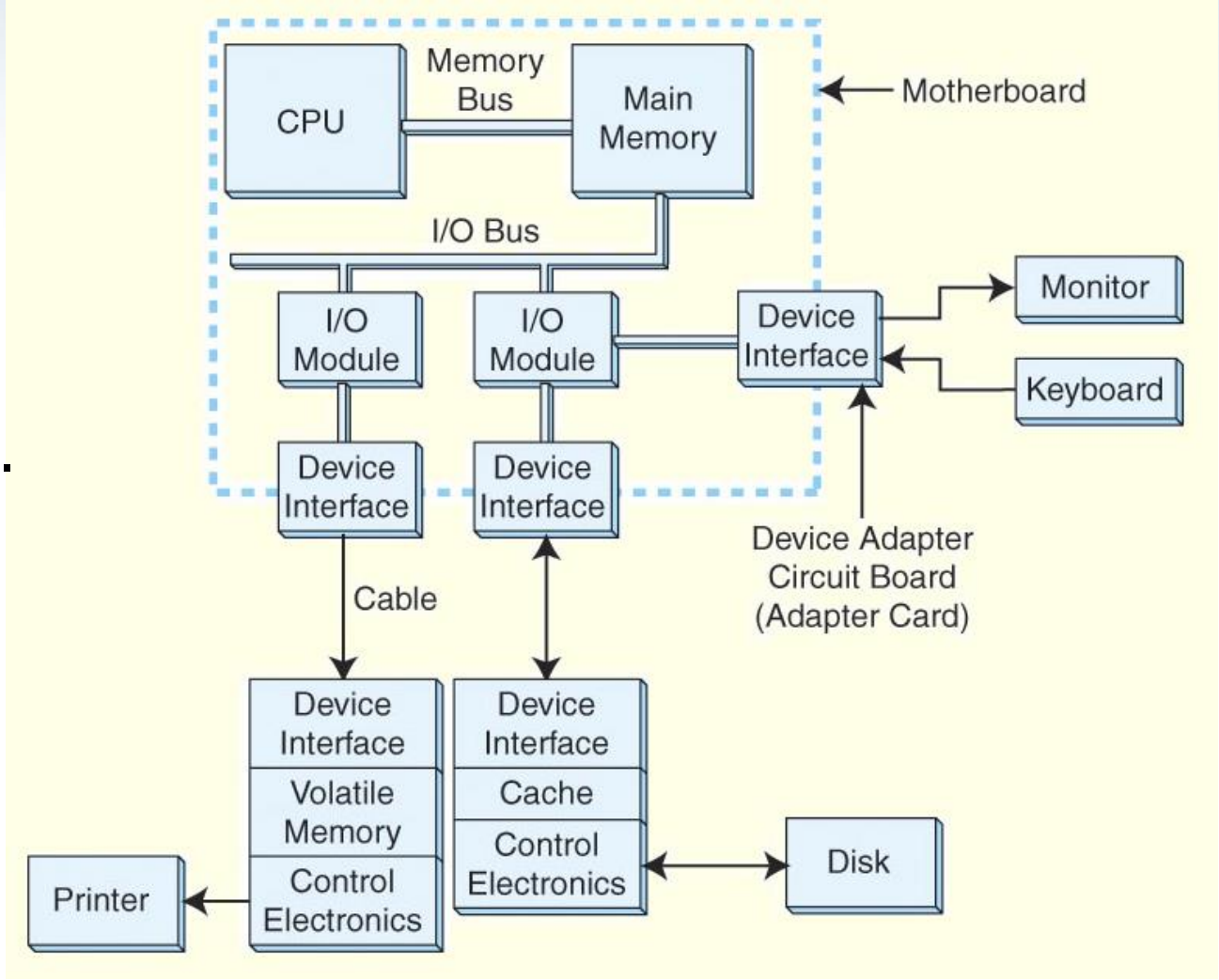
Should price/performance be your only concern?

7.4 I/O Architectures

- We define input/output as a subsystem of components that moves coded 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.

7.4 I/O Architectures

This is a model I/O configuration.



7.4 I/O Architectures

- I/O can be controlled in five general ways.
  - *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.
  - *Memory-Mapped I/O* shares memory address space between I/O devices and program memory.
  - *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 Systems

- Programmed I/O Systems using programmed I/O devote at least one register for the exclusive use of each I/O device. The CPU continually monitors each register, waiting for data to arrive. This is called polling. Thus, programmed I/O is sometimes referred to7.3 / I/O Architectures 277 as polled I/O. Once the CPU detects a “data ready” condition, it acts according to instructions programmed for that particular register.

Interrupt-Driven I/O

The devices tell the CPU when they have data to send.

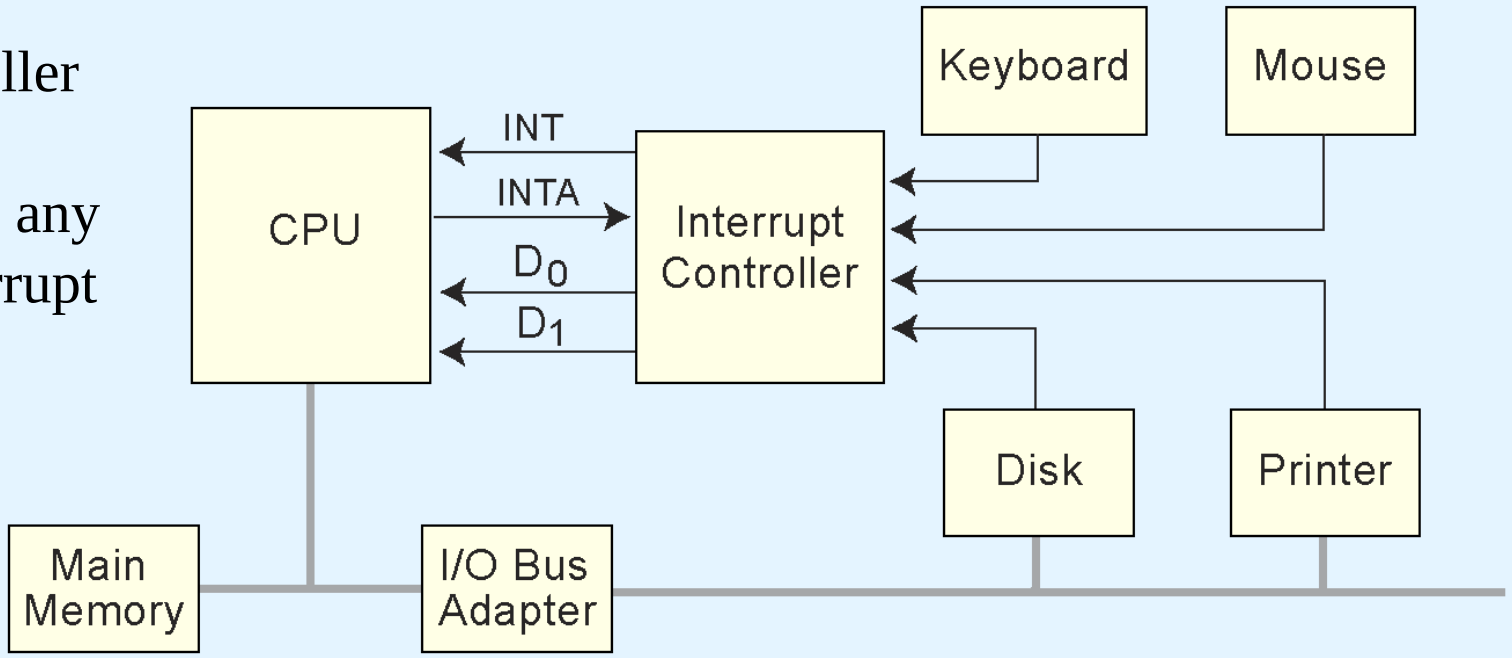
The CPU proceeds with other tasks until a device requesting service interrupts it. Interrupts are usually signaled with a bit in the CPU flags register called an interrupt flag. Once the interrupt flag is set, the operating system interrupts whatever program is currently executing, saving that program's state and variable information. The system then fetches the address vector that points to the address of the I/O service routine. After the CPU has completed servicing the I/O, it restores the information it saved from the program that was running when the interrupt occurred, and the program execution resumes.

7.4 I/O Architectures

This is an idealized I/O subsystem that uses interrupts.

Each device connects its interrupt line to the interrupt controller.

The controller signals the CPU when any of the interrupt lines are asserted.

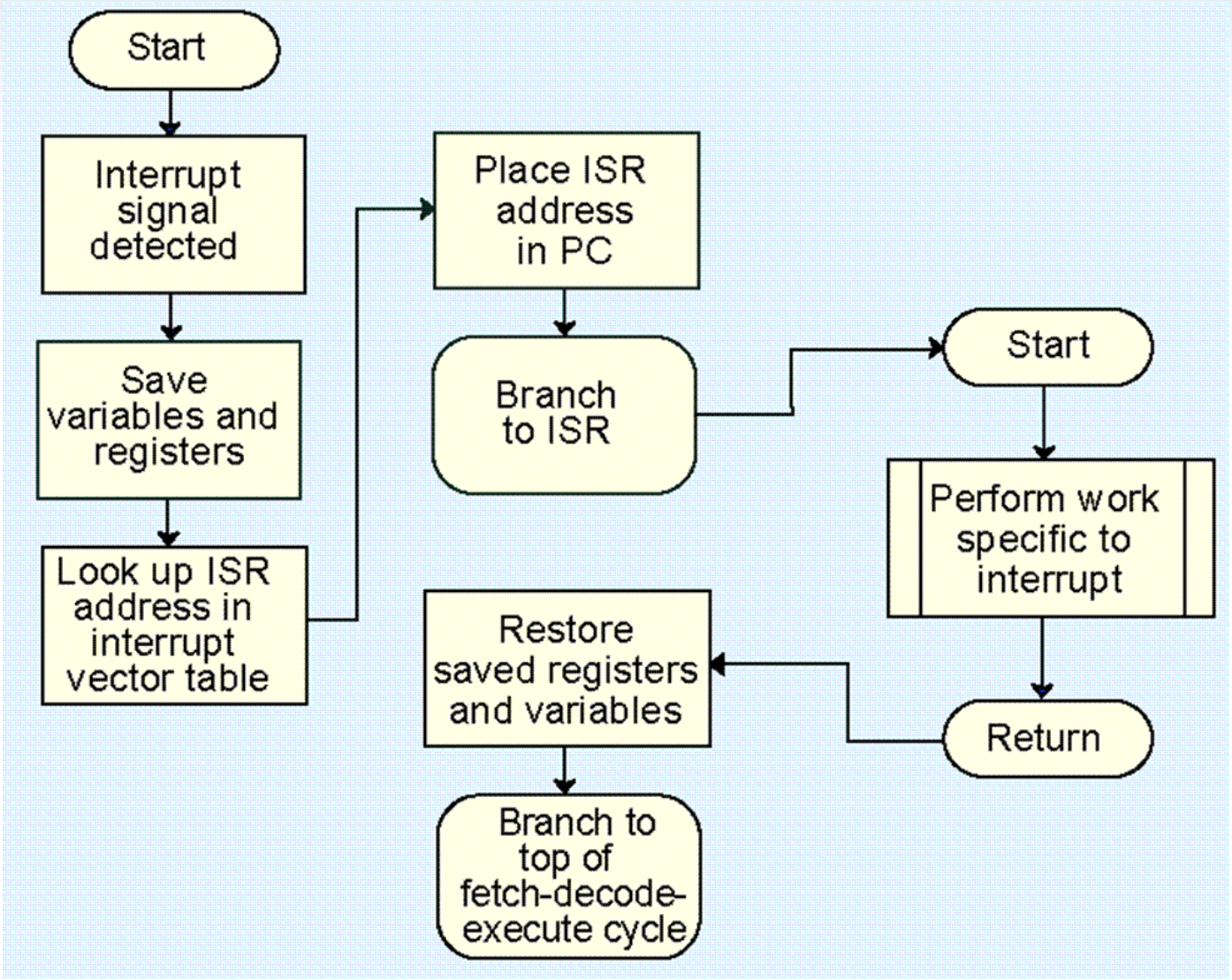


7.4 I/O Architectures

- Recall from Chapter 4 that in a system that uses interrupts, the status of the interrupt signal is checked at the top of the fetch-decode-execute cycle.
- The particular code that is executed whenever an interrupt occurs is determined by a set of addresses called *interrupt vectors* that are stored in low memory.
- The system state is saved before the interrupt service routine is executed and is restored afterward.

We provide a flowchart on the next slide.

7.4 I/O Architectures



7.4 I/O Architectures

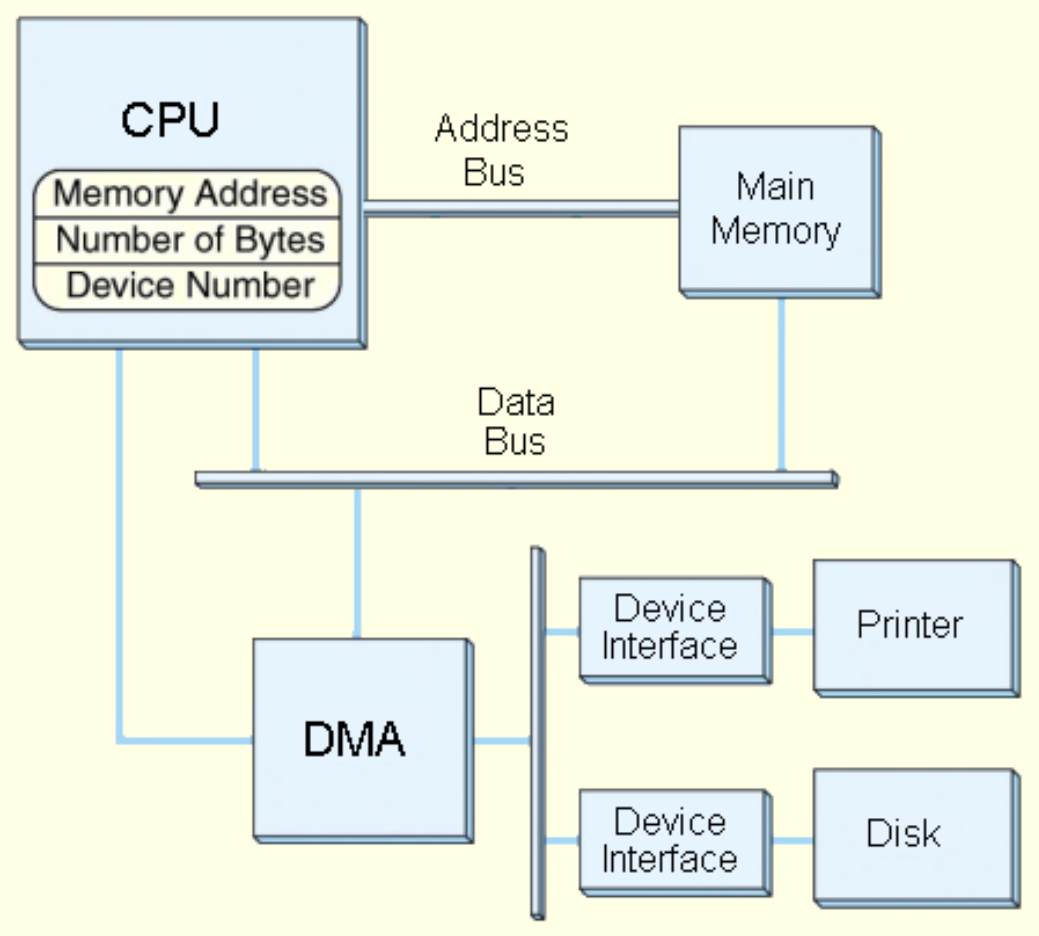
- In memory-mapped I/O devices and main memory share the same address space.
  - Each I/O device has its own reserved block of memory.
  - Memory-mapped I/O therefore looks just like a memory access from the point of view of the CPU.
  - Thus the same instructions to move data to and from both I/O and memory, greatly simplifying system design.
- In small systems the low-level details of the data transfers are offloaded to the I/O controllers built into the I/O devices.

7.4 I/O Architectures

This is a DMA configuration.

Notice that the DMA and the CPU share the bus.

The DMA runs at a higher priority and steals memory cycles from the CPU.



7.4 I/O Architectures

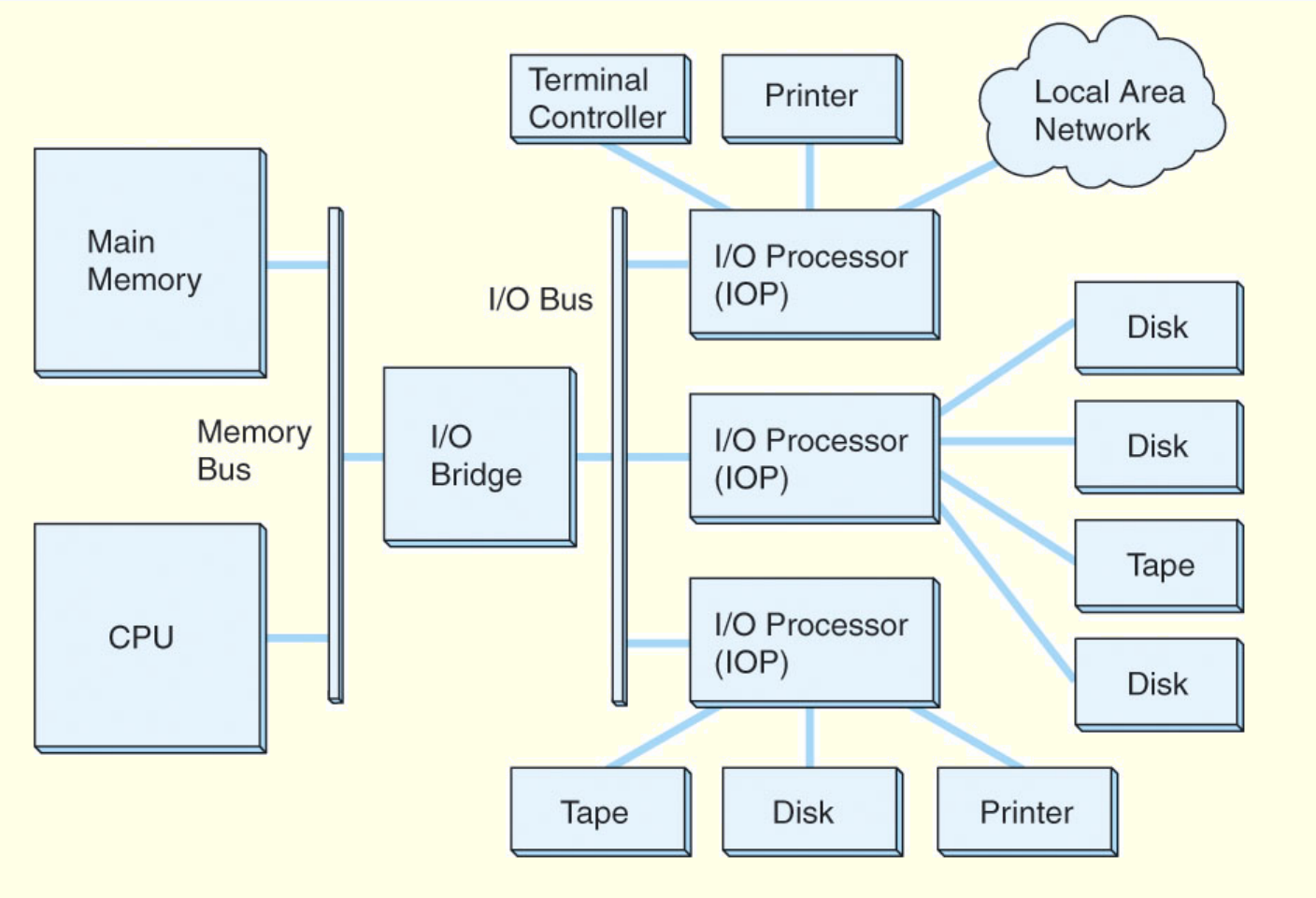
- Very large systems employ channel I/O.
- Channel I/O consists of one or more I/O processors (IOPs) that control various channel paths.
- Slower devices such as terminals and printers are combined (*multiplexed*) into a single faster channel.
- On IBM mainframes, multiplexed channels are called *multiplexor channels*, the faster ones are called *selector channels*.

7.4 I/O Architectures

- 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 independent of the host CPU.
- The host has only to create the program instructions for the I/O operation and tell the IOP where to find them.

7.4 I/O Architectures

- This is a channel I/O configuration.



7.4 I/O Architectures

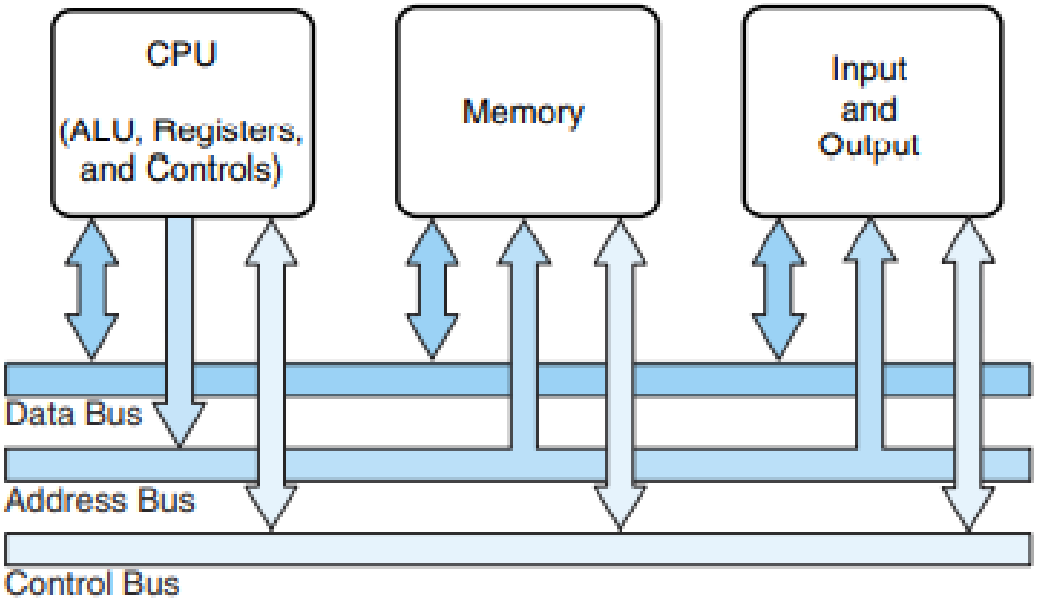
- Character I/O devices process one byte (or character) at a time.
  - Examples include modems, keyboards, and mice.
  - Keyboards are usually connected through an interrupt-driven I/O system.
- Block I/O devices handle bytes in groups.
  - Most mass storage devices (disk and tape) are block I/O devices.
  - Block I/O systems are most efficiently connected through DMA or channel I/O.

7.4 I/O Architectures

- I/O buses, unlike memory buses, operate asynchronously. Requests for bus access must be arbitrated among the devices involved.
- Bus control lines activate the devices when they are needed, raise signals when errors have occurred, and reset devices when necessary.
- The number of data lines is the *width* of the bus.
- A bus clock coordinates activities and provides bit cell boundaries.

7.4 I/O Architectures

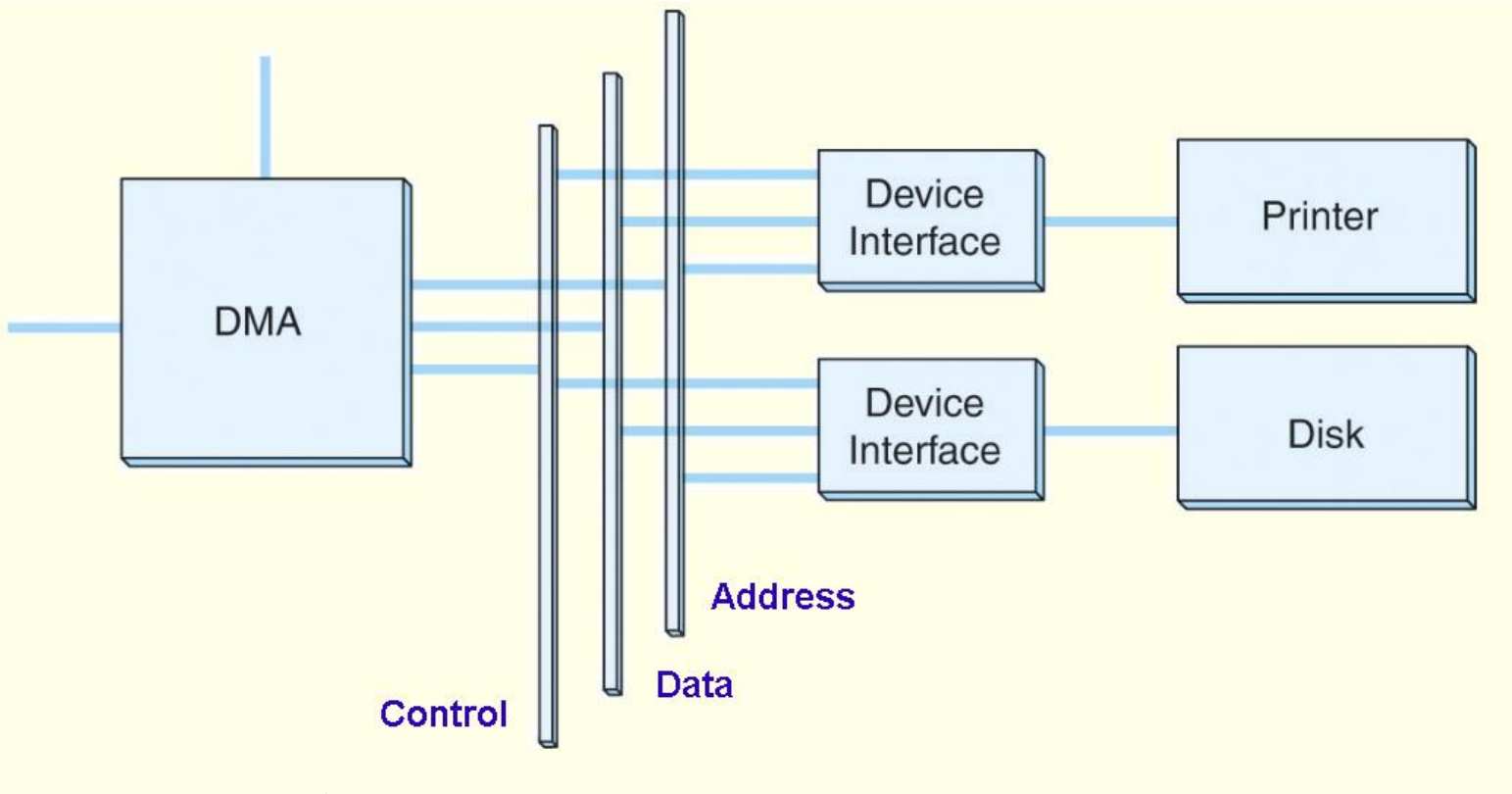
- A system bus is a resource shared among many components of a computer system.
- Access to this shared resource must be controlled. This is why a control bus is required.



High-Level View of a System Bus

7.4 I/O Architectures

This is a generic DMA configuration showing how the DMA circuit connects to a data bus. (shows the individual component buses.)



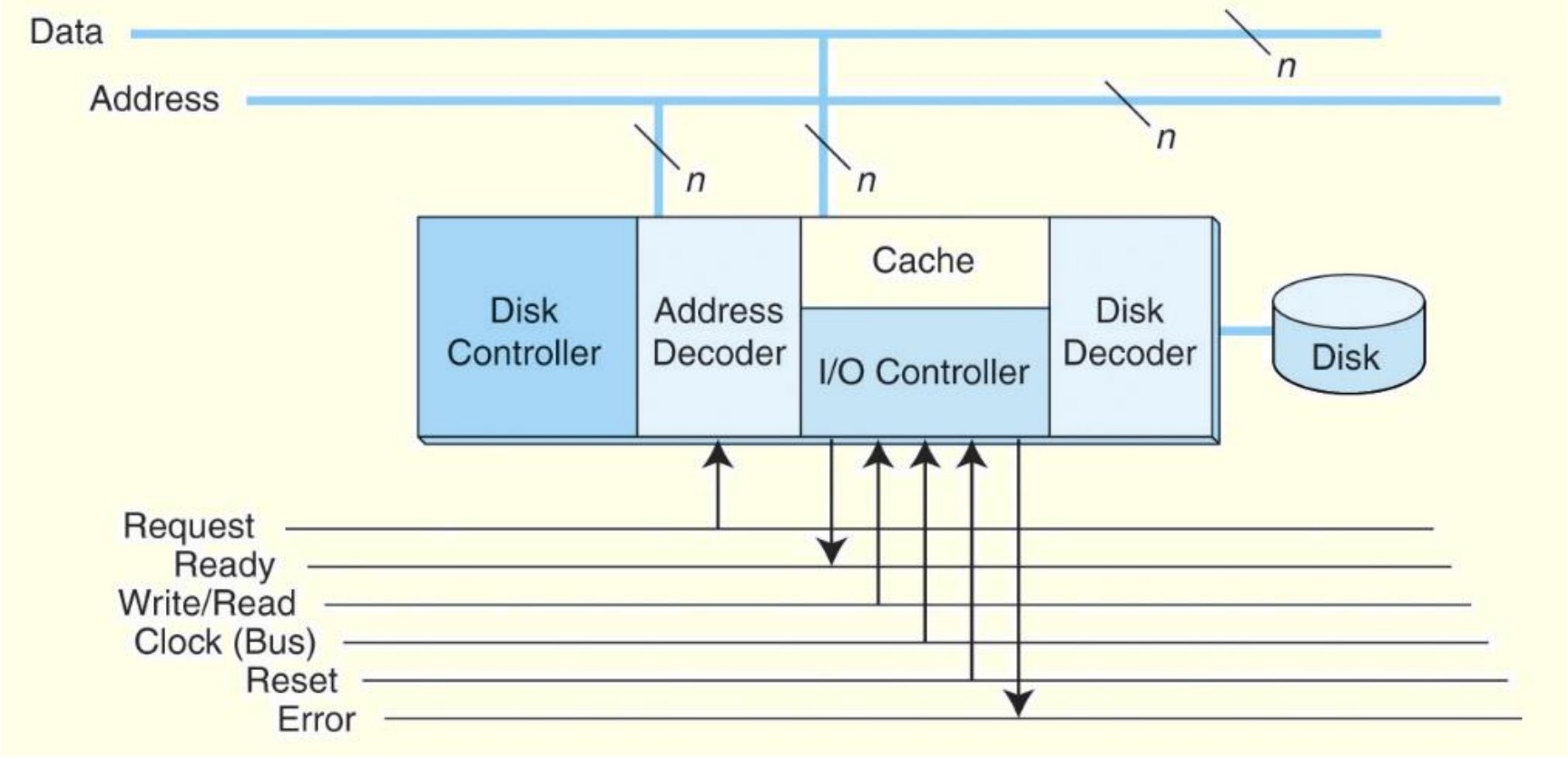
DMA Configuration Showing Separate Address, Data, and Control Lines

7.4 I/O Architectures

- A Disk Controller Interface
- The next slide gives details of how the disk interface connects to all three buses. The address and data buses consist of a number of individual conductors, each of which carries one bit of information. The number of data lines determines the width of the bus. A data bus having eight data lines carries one byte at a time. The address bus has a sufficient number of conductors to uniquely identify each device on the bus.

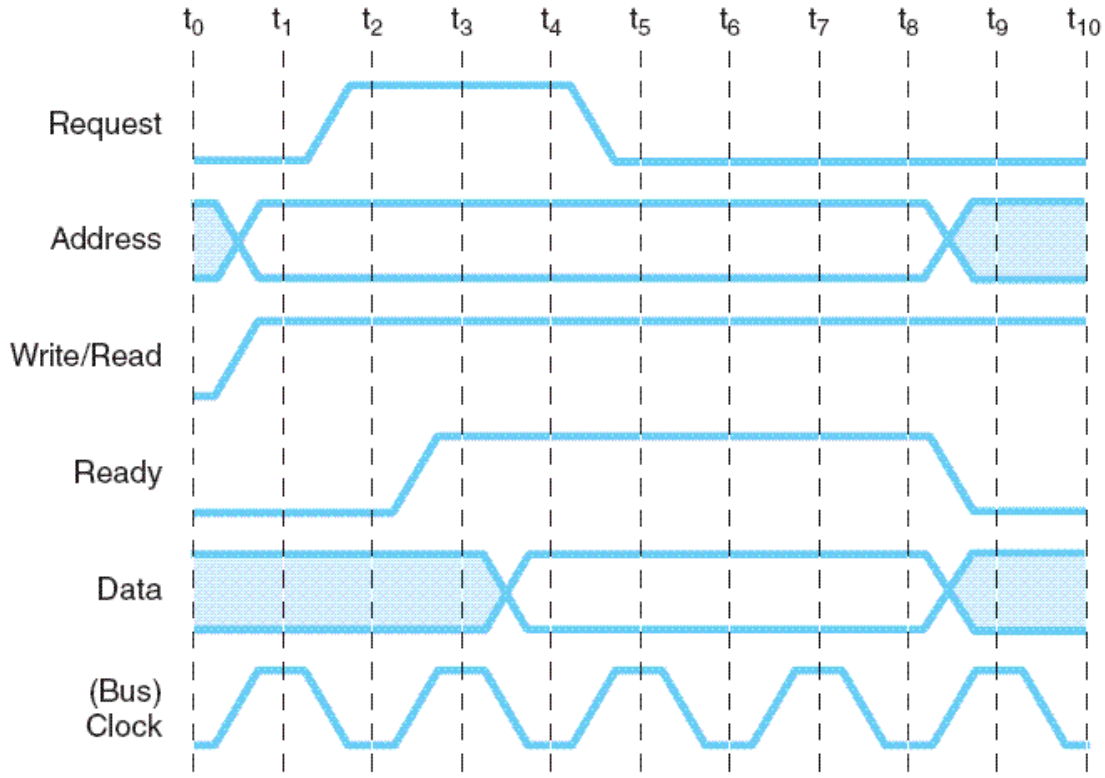
7.4 I/O Architectures

This is how a bus connects to a disk drive.



A Disk Controller Interface with Connections to the I/O Bus

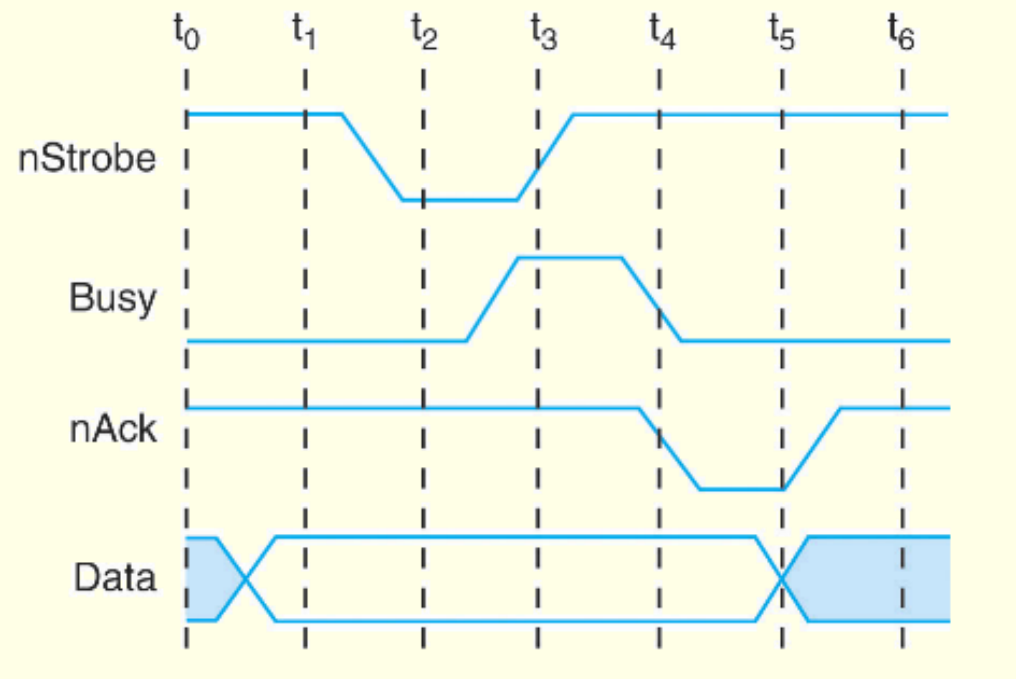
Timing diagrams, such as this one, define bus operation in detail.



Time	Salient Bus Signal	Meaning
t_0	Assert Write	Bus is needed for writing (not reading)
t_1	Assert Address	Indicates where bytes will be written
t_2	Assert Request	Request write to address on address lines
t_3	Assert Ready	Acknowledges write request, bytes placed on data lines
t_4–t_7	Data Lines	Write data (requires several cycles)
t_8	Lower Ready	Release bus

7.5 Data Transmission Modes

- Bytes can be conveyed from one point to another by sending their encoding signals simultaneously using *parallel data transmission* or by sending them one bit at a time in *serial data transmission*.
  - Parallel data transmission for a printer resembles the signal protocol of a memory bus:



7.5 Data Transmission Modes

- In parallel data transmission, the interface requires one conductor for each bit.
- Parallel cables are fatter than serial cables.
- Compared with parallel data interfaces, serial communications interfaces:
  - Require fewer conductors.
  - Are less susceptible to attenuation.
  - Can transmit data farther and faster.

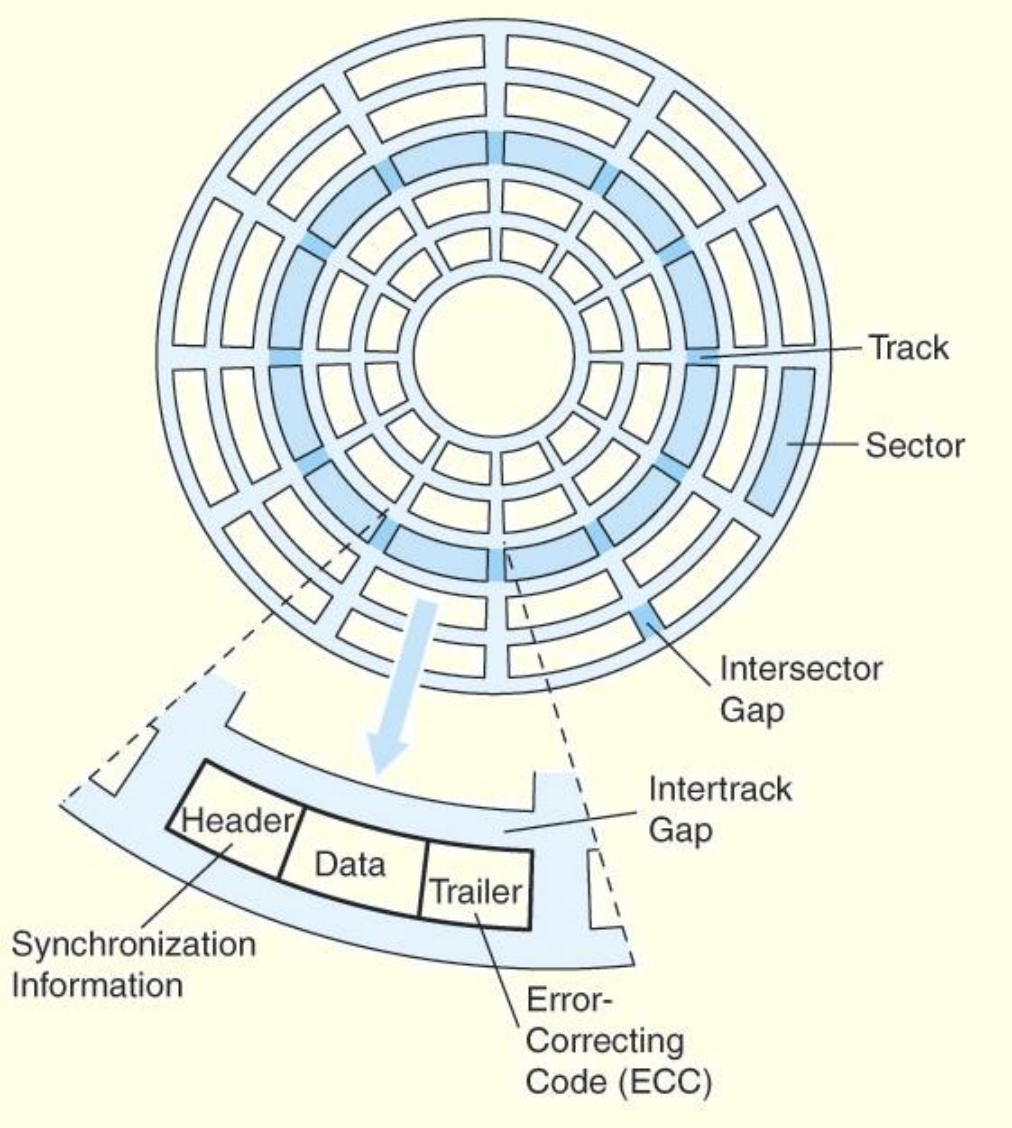
Serial communications interfaces are suitable for time-sensitive (*isochronous*) data such as voice and video.

7.6 Magnetic Disk Technology

- Magnetic disks offer large amounts of durable storage that can be accessed quickly.
- Disk drives are called *random* (or *direct*) *access storage devices,* because blocks of data can be accessed according to their location on the disk.
  - This term was coined when all other durable storage (e.g., tape) was sequential.
- Magnetic disk organization is shown on the following slide.

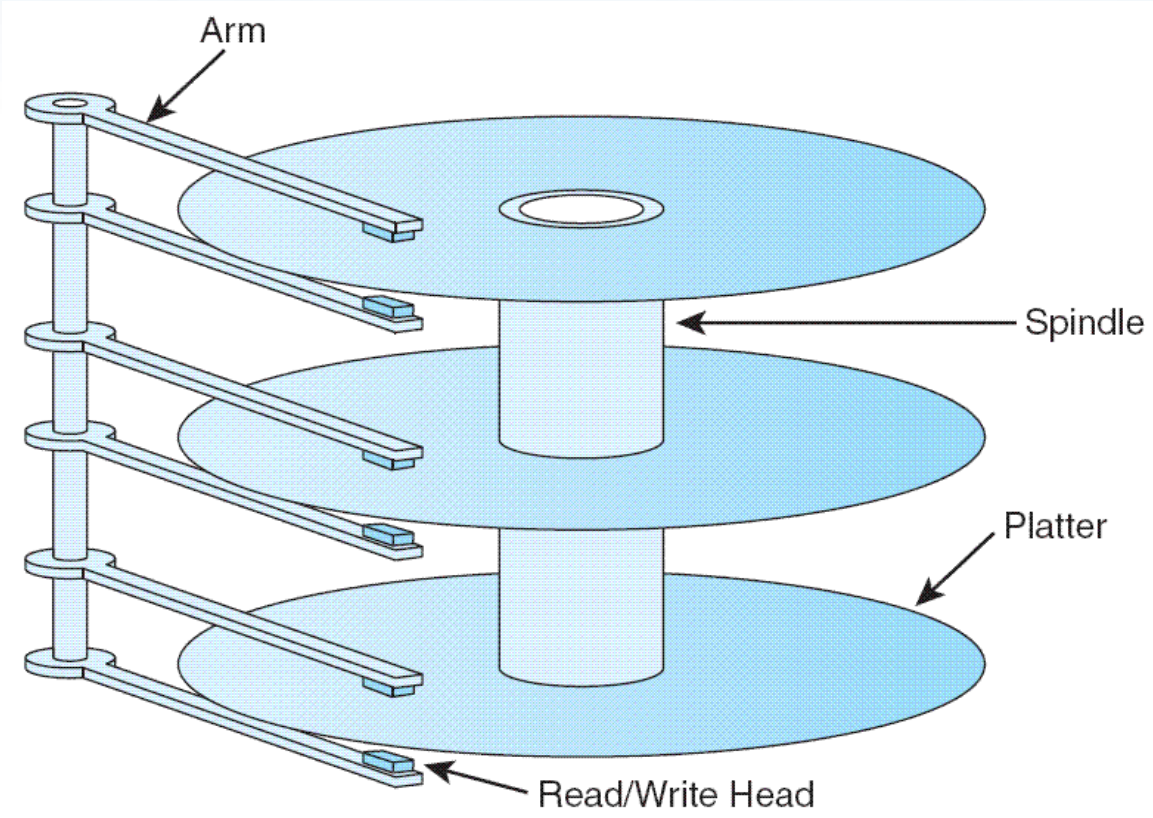
7.6 Magnetic Disk Technology

Disk tracks are numbered from the outside edge, starting with zero.



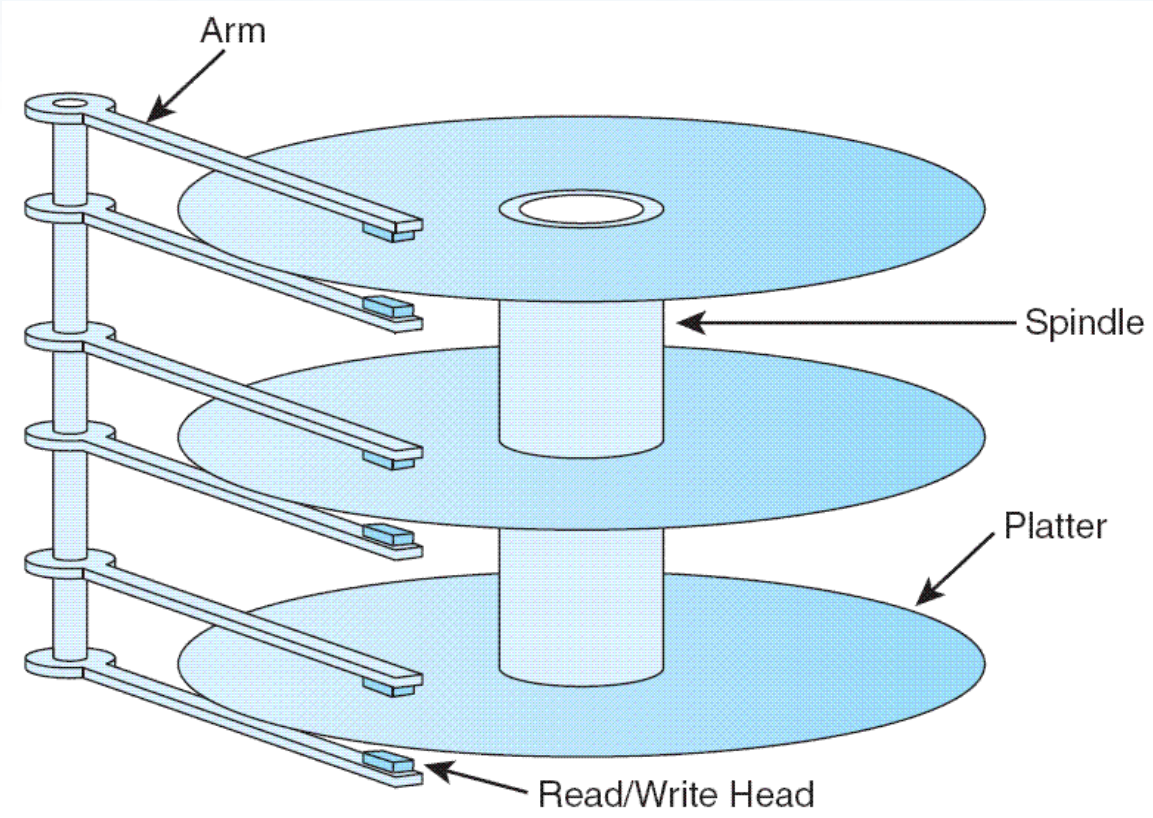
7.6 Magnetic Disk Technology

- Hard disk platters are mounted on spindles.
- Read/write heads are mounted on a comb that swings radially to read the disk.



7.6 Magnetic Disk Technology

- The rotating disk forms a logical cylinder beneath the read/write heads.
- Data blocks are addressed by their cylinder, surface, and sector.



7.6 Magnetic Disk Technology

- There are a number of electromechanical properties of hard disk drives that determine how fast its data can be accessed.
- *Seek time* is the time that it takes for a disk arm to move into position over the desired cylinder.
- *Rotational delay* is the time that it takes for the desired sector to move into position beneath the read/write head.
- Seek time + rotational delay = *access time*.

7.6 Magnetic Disk Technology

- *Transfer rate* gives us the rate at which data can be read from the disk.
- *Average latency* is a function of the rotational speed:

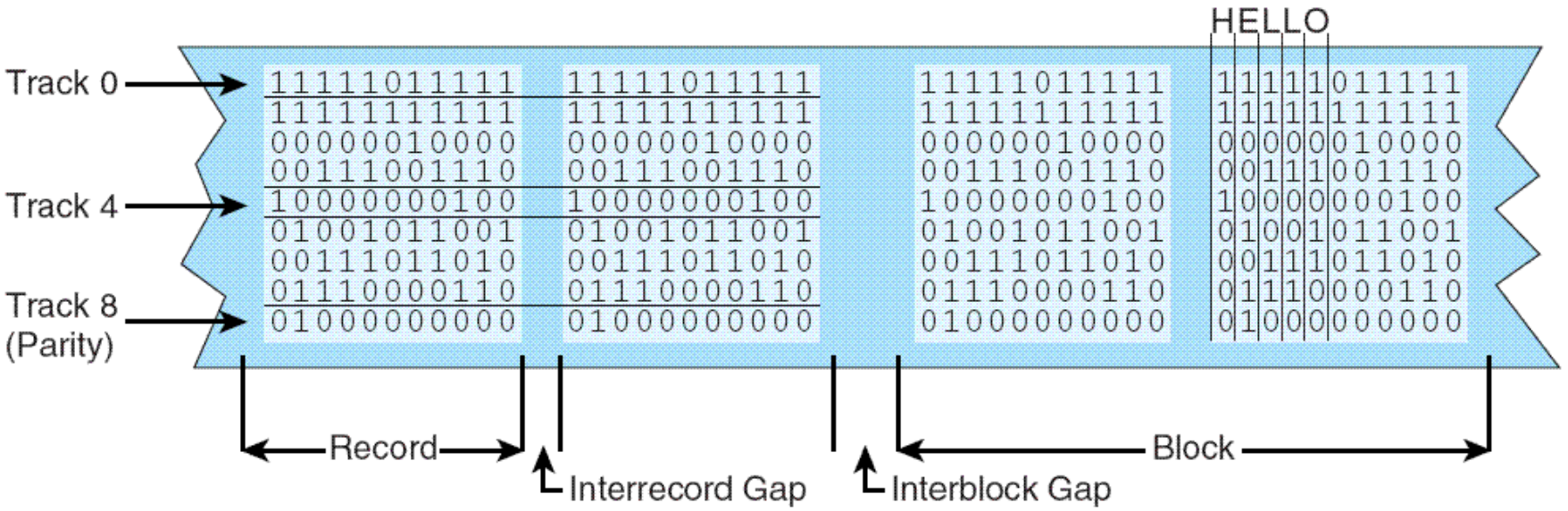
$$\frac{\frac{60 \text{ seconds}}{\text{disk rotation speed}} \times \frac{1000 \text{ ms}}{\text{second}}}{2}$$

- *Mean Time To Failure (MTTF)* is a statistically-determined value often calculated experimentally.
  - It usually doesn't tell us much about the actual expected life of the disk. *Design life* is usually more realistic.

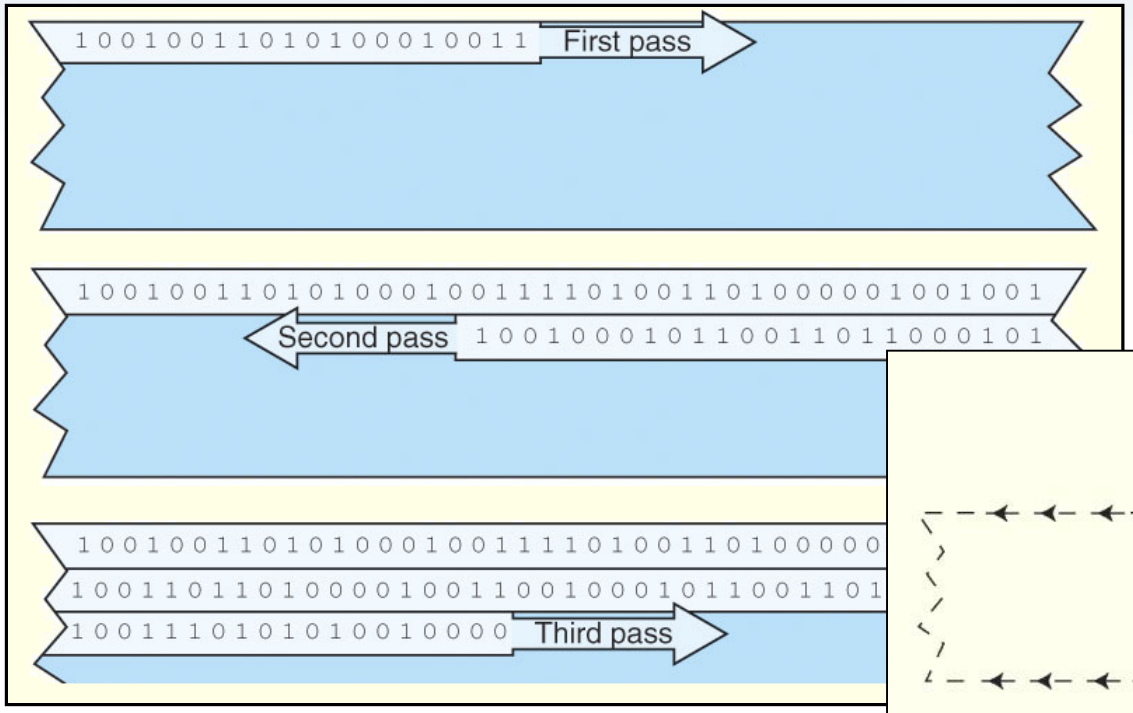
Figure 7.15 in the text shows a sample disk specification.

7.8 Magnetic Tape

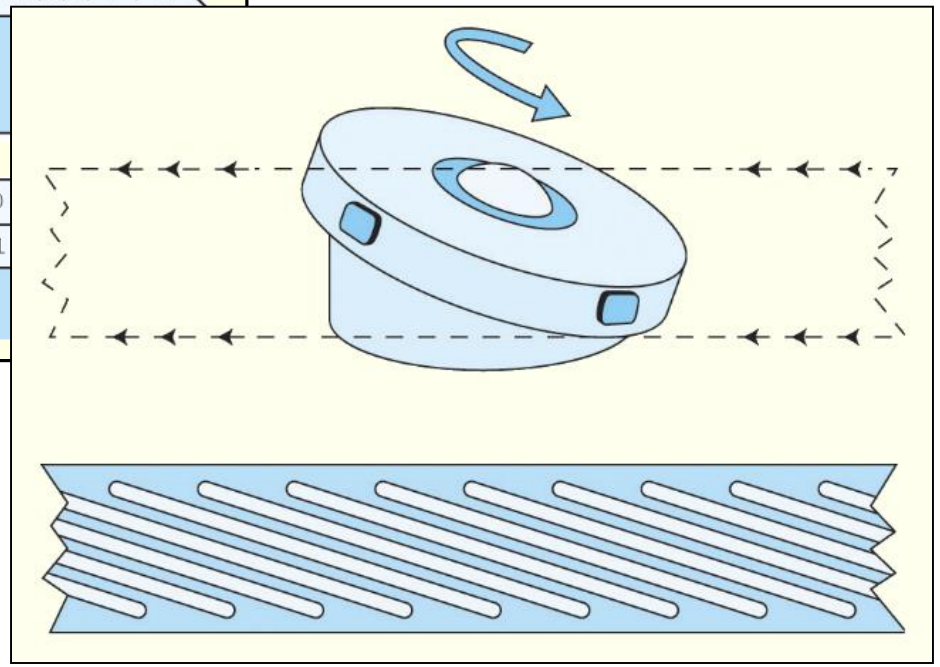
- First-generation magnetic tape was not much more than wide analog recording tape, having capacities under 11MB.
- Data was usually written in nine vertical tracks:



7.8 Magnetic Tape



← Serpentine



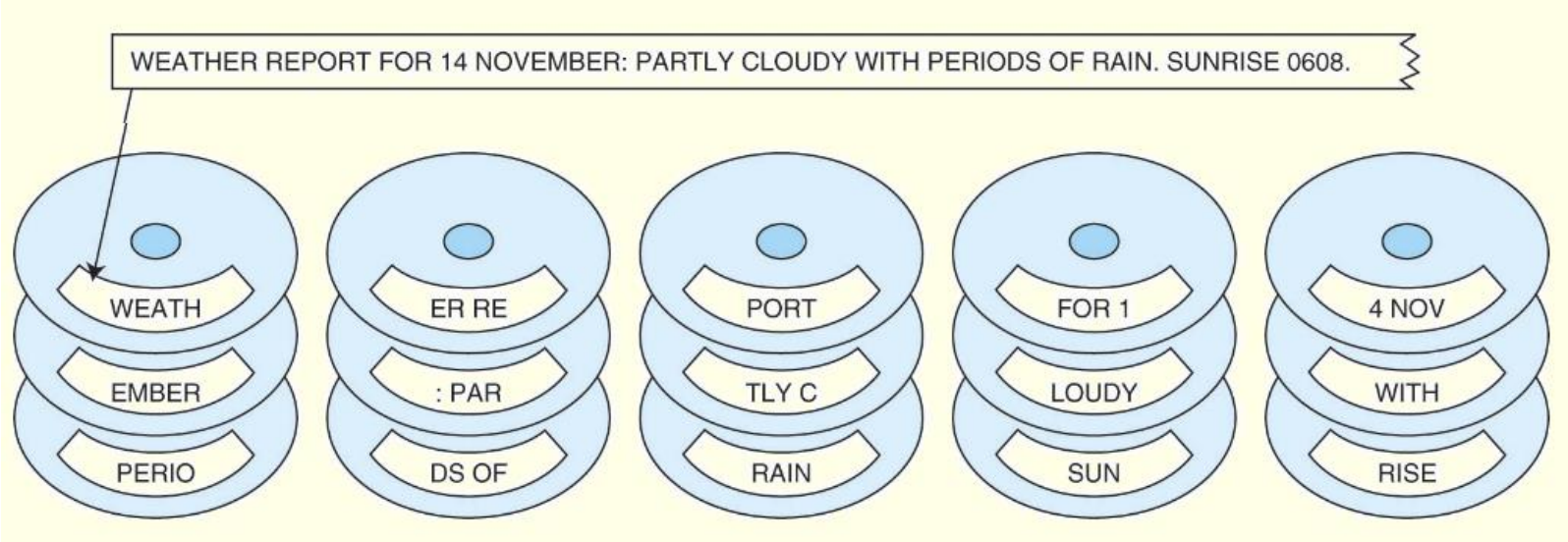
Helical Scan →

7.9 RAID

- RAID, an acronym for *Redundant Array of Independent Disks* was invented to address problems of disk reliability, cost, and performance.
- In RAID, data is stored across many disks, with extra disks added to the array to provide error correction (redundancy).
- The inventors of RAID, David Patterson, Garth Gibson, and Randy Katz, provided a RAID taxonomy that has persisted for a quarter of a century, despite many efforts to redefine it.

7.9 RAID

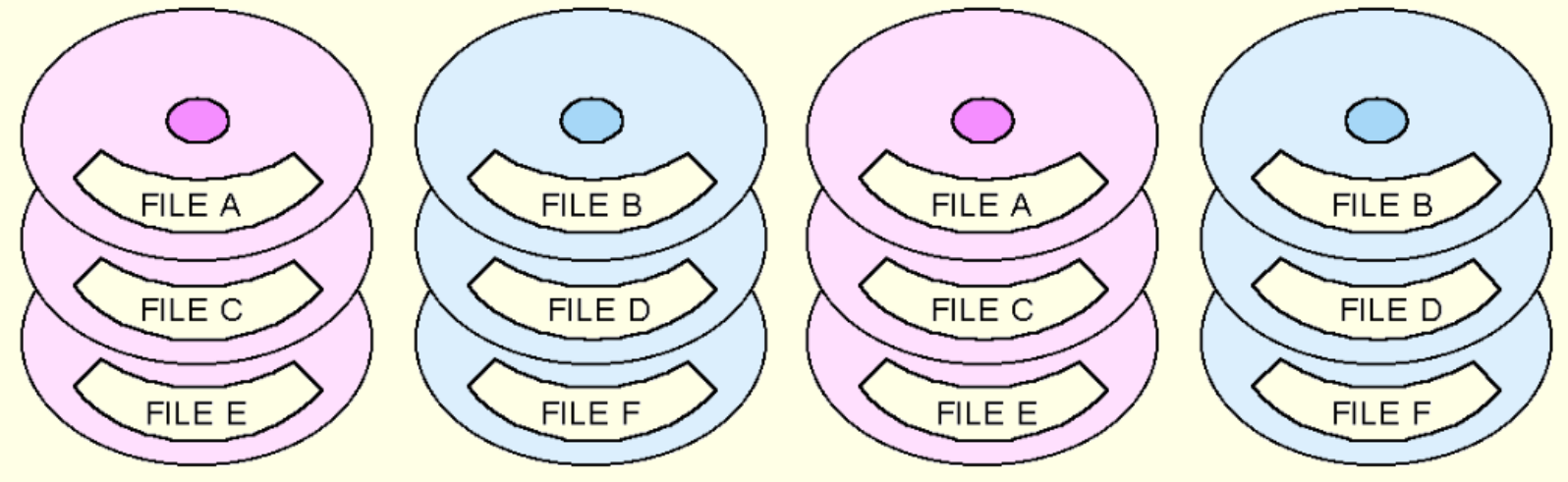
- RAID Level 0, also known as *drive spanning*, provides improved performance, but no redundancy.
  - Data is written in blocks across the entire array



  - The disadvantage of RAID 0 is in its low reliability.

7.9 RAID

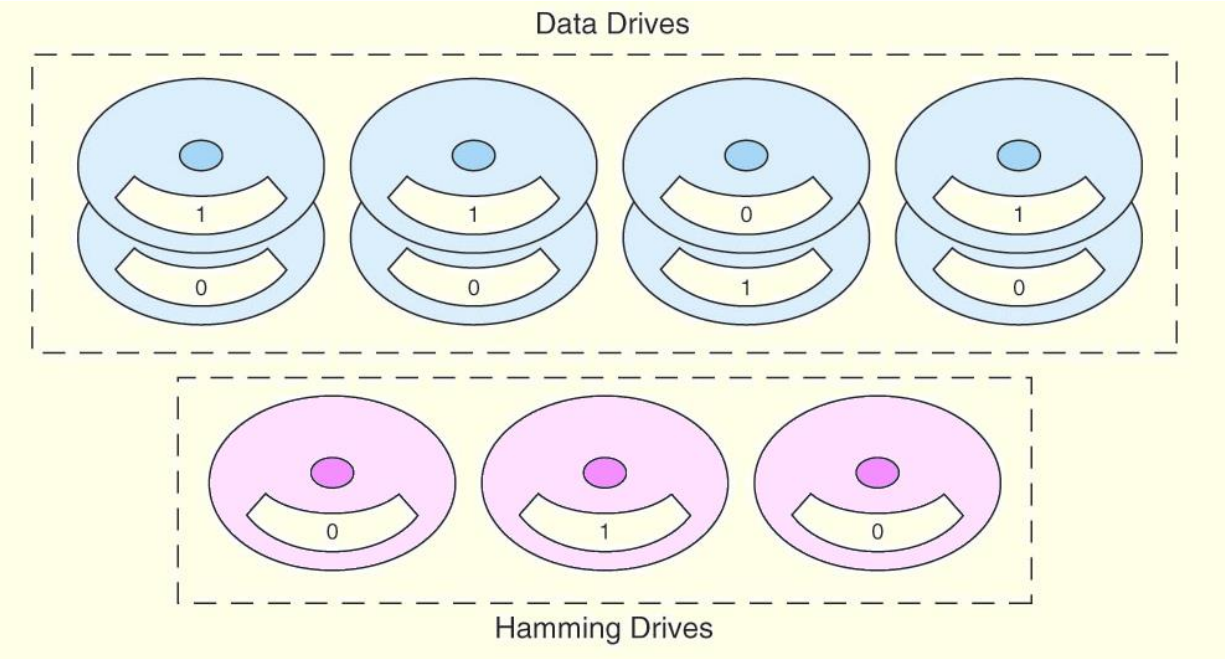
- RAID Level 1, also known as *disk mirroring*, provides 100% redundancy, and good performance.
  - Two matched sets of disks contain the same data.



  - The disadvantage of RAID 1 is cost.

7.9 RAID

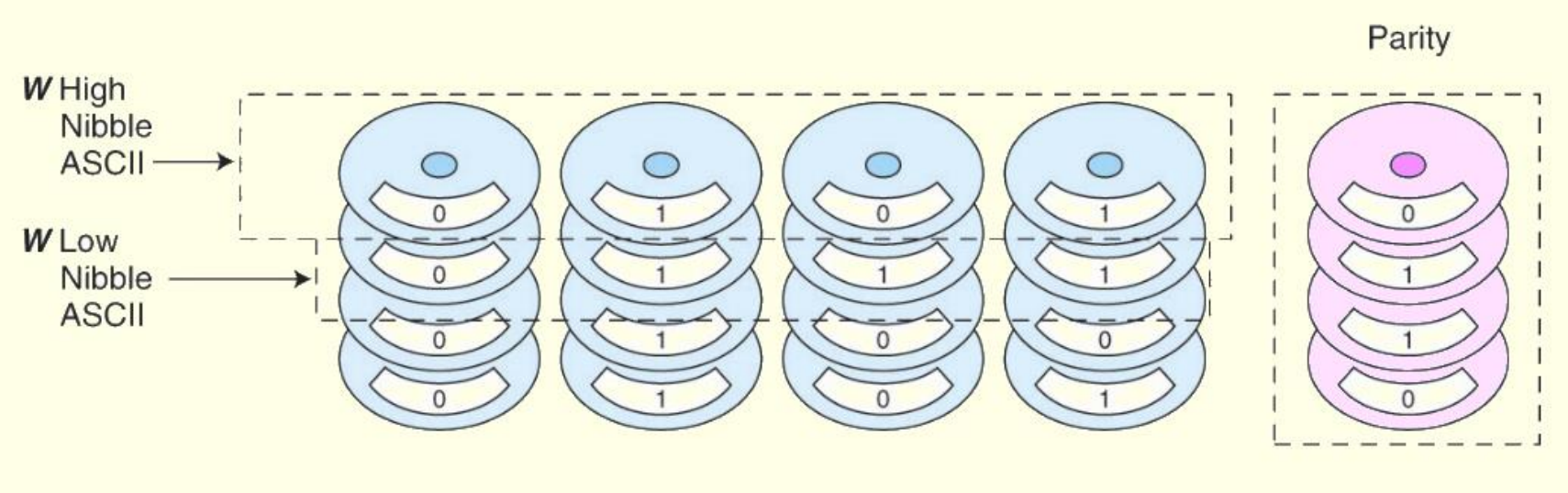
- A RAID Level 2 configuration consists of a set of data drives, and a set of Hamming code drives.
  - Hamming code drives provide error correction for the data drives.



  - RAID 2 performance is poor and the cost is relatively high.

7.9 RAID

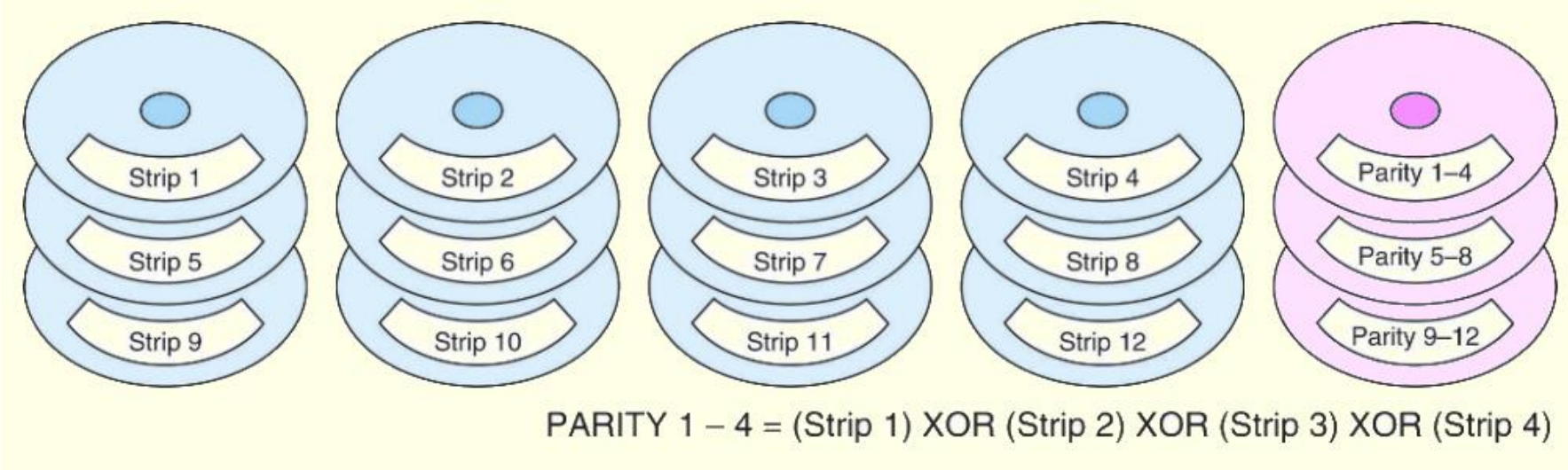
- RAID Level 3 stripes bits across a set of data drives and provides a separate disk for parity.
  - Parity is the XOR of the data bits.



  - RAID 3 is not suitable for commercial applications, but is good for personal systems.

7.9 RAID

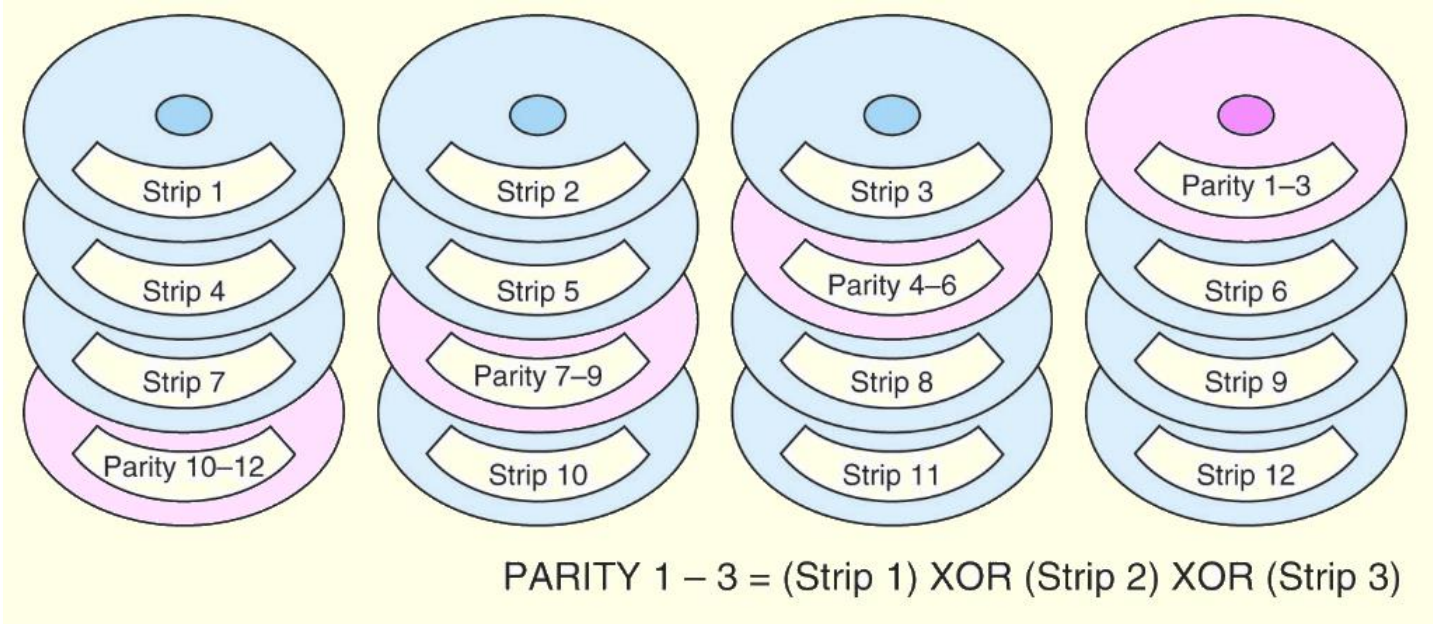
- RAID Level 4 is like adding parity disks to RAID 0.
  - Data is written in blocks across the data disks, and a parity block is written to the redundant drive.



  - RAID 4 would be feasible if all record blocks were the same size.

7.9 RAID

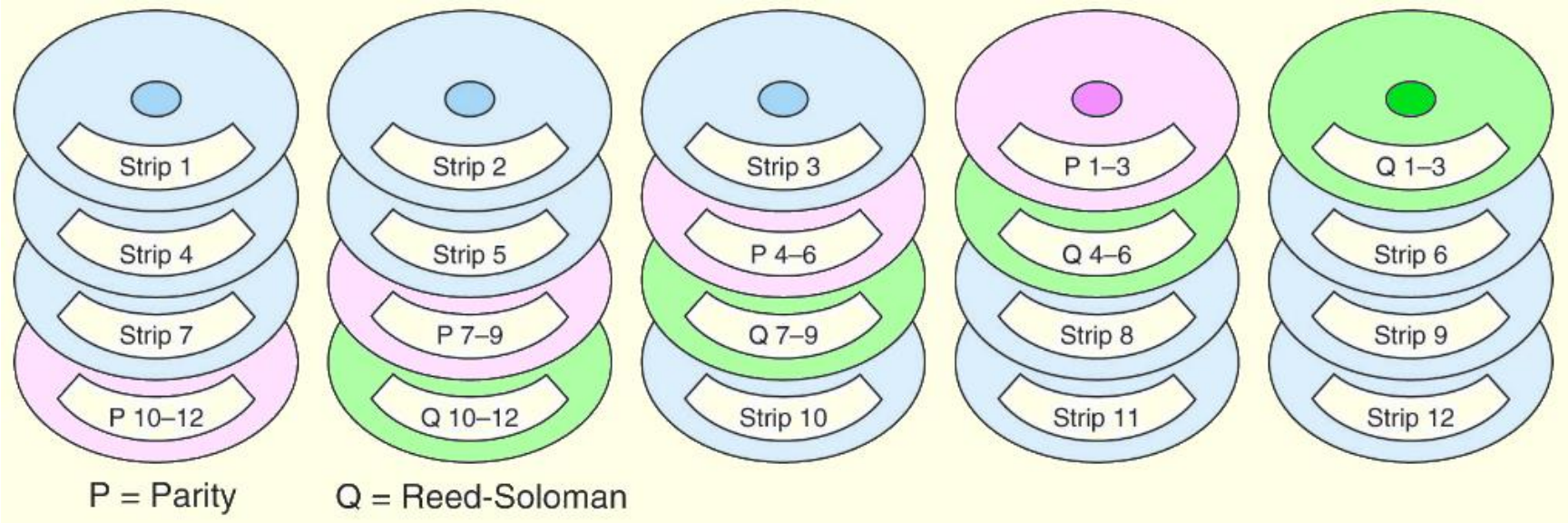
- RAID Level 5 is RAID 4 with distributed parity.
  - With distributed parity, some accesses can be serviced concurrently, giving good performance and high reliability.



  - RAID 5 is used in many commercial systems.

7.9 RAID

- RAID Level 6 carries two levels of error protection over striped data: Reed-Soloman and parity.
  - It can tolerate the loss of two disks.



  - RAID 6 is write-intensive, but highly fault-tolerant.