

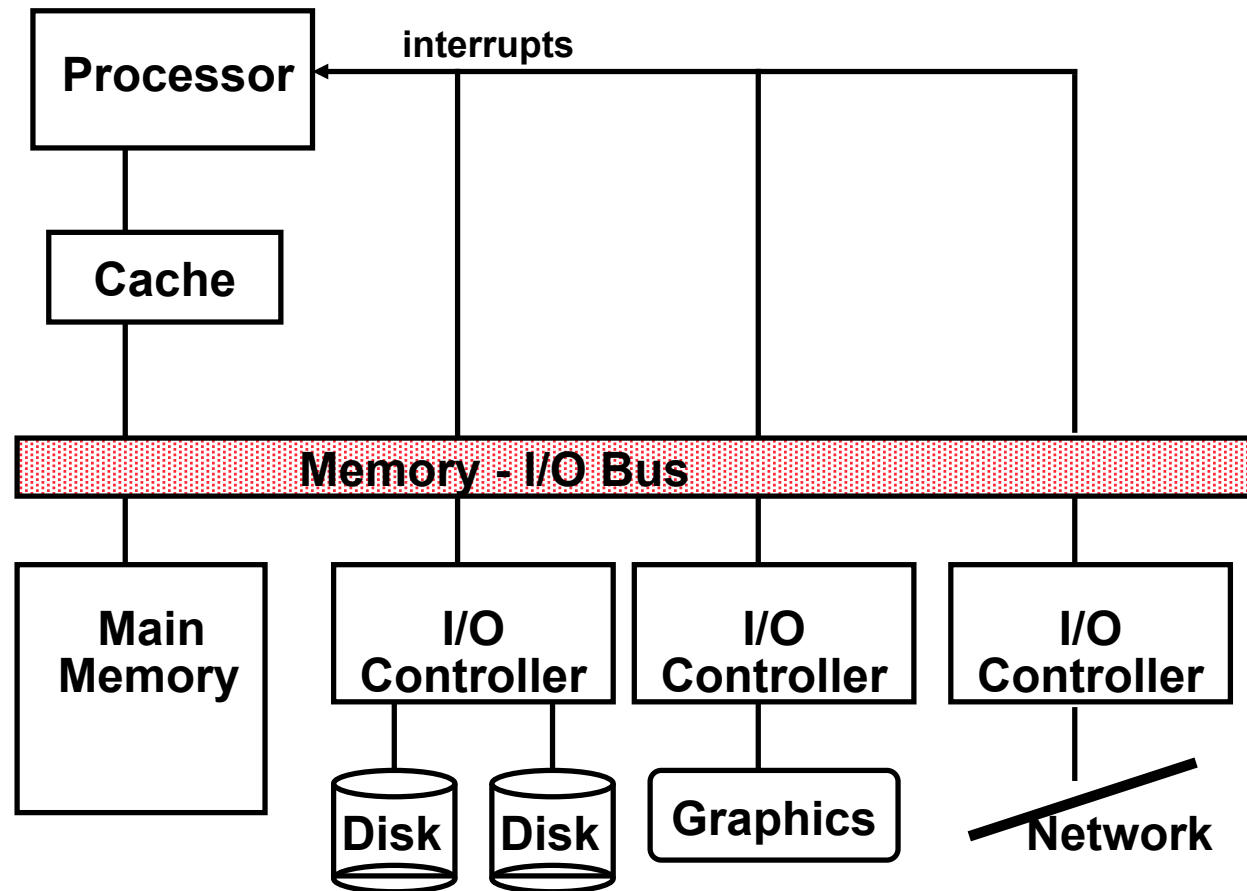
CoE3DR4

Computer Organization

Chapter 6: Storage, Networks and Other Peripherals



-
- Performance
 - Expandability
 - Resilience in the face of failure



I/O Device Characteristics

- I/O devices are characterized according to:
 - **Behavior:**
 - Input (read only).
 - Output (write only, cannot be read).
 - Storage (can be reread and usually rewritten).
 - **Partner:** Either a human or a machine at the other end of the I/O device.
 - **Data rate:** The peak rate at which data can be transferred between the I/O device and memory or CPU.

Device	Behavior	Partner	Data rate (Mbit/sec)
Keyboard	Input	Human	0.0001
Mouse	Input	Human	0.0038
Voice input	Input	Human	0.2640
Sound input	Input	Machine	3.0000
Scanner	Input	Human	3.2000
Voice output	Output	Human	0.2640
Sound output	Output	Human	8.0000
Laser printer	Output	Human	3.2000
Graphics display	Output	Human	800.0000–8000.0000
Cable modem	Input or output	Machine	0.1280–6.0000
Network/LAN	Input or output	Machine	100.0000–10000.0000
Network/wireless LAN	Input or output	Machine	11.0000–54.0000
Optical disk	Storage	Machine	80.0000–220.0000
Magnetic tape	Storage	Machine	5.0000–120.0000
Flash memory	Storage	Machine	32.0000–200.0000
Magnetic disk	Storage	Machine	800.0000–3000.0000

I/O System Performance

- I/O System performance depends on many aspects of the system (“limited by weakest link in the chain”):
 - The memory system
 - The underlying interconnection (buses)
 - The I/O device
 - The speed of the I/O software (Operating System)
 - The efficiency of the software’s use of the I/O devices
- Two common performance metrics:
 - Throughput: I/O bandwidth
 - Response time: Latency

I/O System Performance

- Throughput:
 - The number of tasks completed by the server in unit time
- Response time:
 - total elapsed time to accomplish a particular task
 - If the I/O requests are large, response time will depend on bandwidth
 - If the accesses are small, the I/O system with the lowest latency per access will deliver the best response time

Reliability and Availability

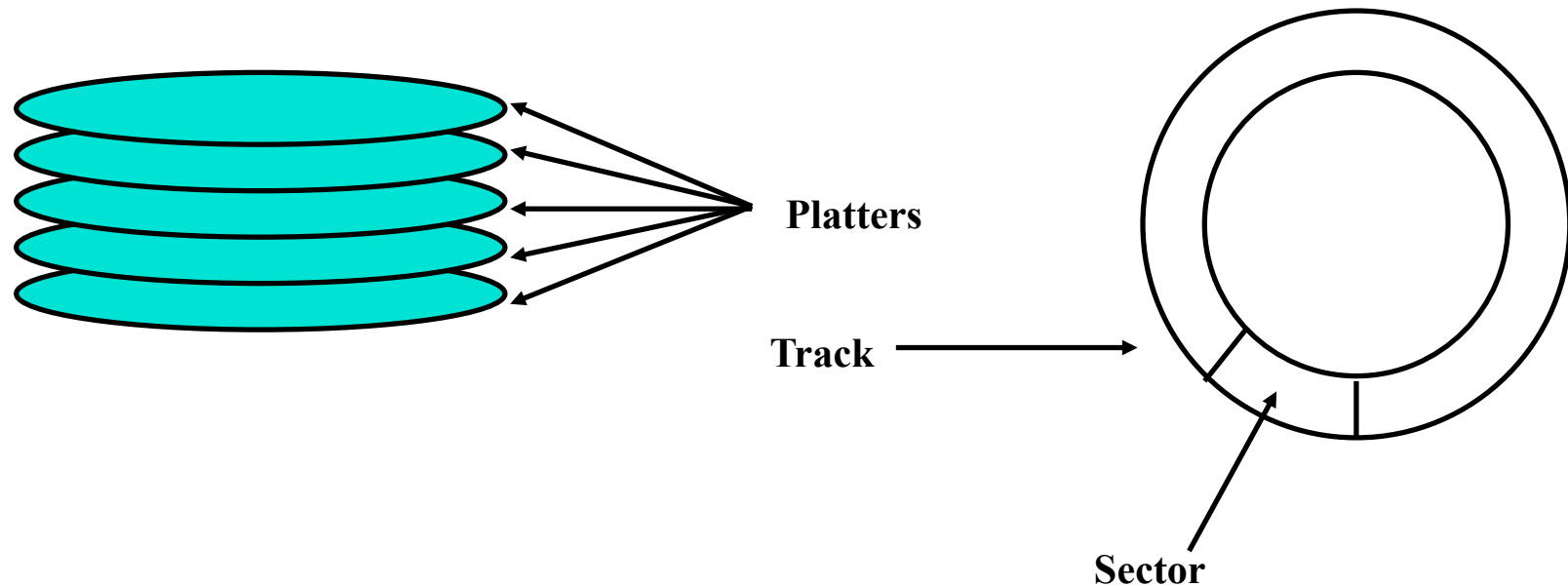
- A system can alternate between two states of delivered service:
 1. Service accomplished: service is delivered as specified
 2. Service interruption: service is different from specification
- Transition from 1 to 2: failure
- Transition from 2 to 1: restoration
- Reliability: a measure of continuous service accomplishment
- Mean time to failure (MTTF) is a reliability measure
- Service interruption is measured as mean time to repair (MTTR)
- $\text{Availability} = \text{MTTF} / (\text{MTTF} + \text{MTTR})$

-
- Three ways to improve MTTF:
 - Fault avoidance
 - prevent fault
 - Fault Tolerance (Redundancy)
 - using redundancy to allow service comply with the specification despite fault occurring
 - Fault Forecasting
 - predict when a fault is going to happen

Magnetic Disk

- Purpose:
 - Long term, nonvolatile storage
 - Large, inexpensive, and slow
 - Lowest level in the memory hierarchy
 - Rely on a rotating platter coated with a magnetic surface
 - Use a moveable read/write head to access the disk
- Typical numbers (depending on the disk size):
 - Diameter typical diameter ranges from 1 to 3.5 in
 - 10,000 to 50,000 tracks per surface
 - 100 to 500 sectors per track
 - A sector is the smallest unit that can be read or written

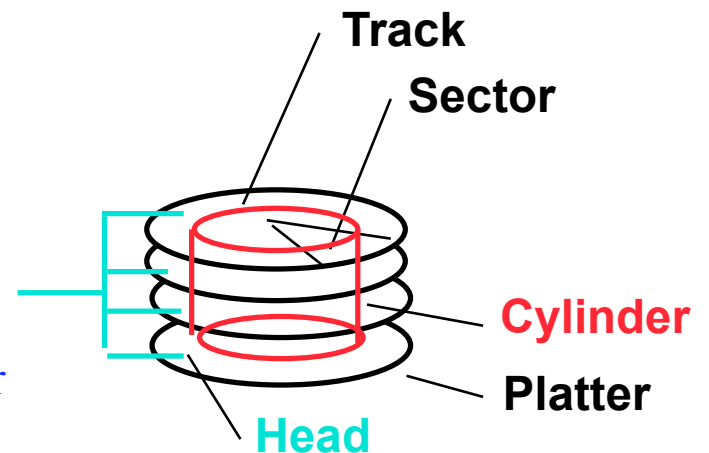
Organization of a Hard Magnetic Disk



- Traditionally all tracks have the same number of sectors
- Recently relaxed: varying number of sectors, speed varies with track location

Magnetic Disk Characteristic

- Cylinder: all the tracks under the heads at a given point on all surfaces
- Read/write data is a three-stage process:
 - Seek time: position the head over the proper track
 - Rotational latency: wait for the desired sector to rotate under the read/write head
 - Transfer time: transfer a block of bits (sector) under the read-write head
- Average seek time as reported by the industry:
 - Typically in the range of 3 ms to 12 ms
 - $(\text{Sum of the time for all possible seek}) / (\text{total \# of possible seeks})$
- Due to locality of disk reference, actual average seek time may:
 - Only be 25% to 33% of the advertised number



Typical Numbers of a Magnetic Disk

- Rotational Latency:
 - Most disks rotate at 5400 to 15000 RPM
 - Approximately 11.2 ms to 4 ms per revolution, respectively
 - An average latency to the desired information is halfway around the disk:
5.6 ms at 5400 RPM, 2 ms at 15000 RPM
- Transfer Time is a function of :
 - Transfer size (usually a sector): 512 B / sector
 - Rotation speed: 5400 RPM to 15000 RPM
 - Recording density: bits per inch on a track
 - Typical values: up to 320 MB per second

-
- A disk controller usually handles the control of the disk and transfer of data
 - Controller adds a delay called controller time (controller overhead)

Example

- 512 byte sector, rotate at 15,000 RPM, advertised seeks is 4 ms, transfer rate is 100 MB/sec, controller overhead is 0.2 ms, disk idle so no waiting time
- Disk Access Time = Seek time + Rotational Latency + Transfer time
+ Controller Time + waiting Delay
- Disk Access Time = 4 ms + 0.5 / 15000 RPM + 0.5 KB / 100 MB/s + 0.2 ms
+ 0
- Disk Access Time = 6.2 ms
- If real seeks are 1/4 advertised seeks, then its 3.2 ms, with rotation delay being the largest component

Characteristics	Seagate ST33000655SS	Seagate ST31000340NS	Seagate ST973451SS	Seagate ST9160821AS
Disk diameter (inches)	3.50	3.50	2.50	2.50
Formatted data capacity (GB)	147	1000	73	160
Number of disk surfaces (heads)	2	4	2	2
Rotation speed (RPM)	15,000	7200	15,000	5400
Internal disk cache size (MB)	16	32	16	8
External interface, bandwidth (MB/sec)	SAS, 375	SATA, 375	SAS, 375	SATA, 150
Sustained transfer rate (MB/sec)	73–125	105	79–112	44
Minimum seek (read/write) (ms)	0.2/0.4	0.8/1.0	0.2/0.4	1.5/2.0
Average seek read/write (ms)	3.5/4.0	8.5/9.5	2.9/3.3	12.5/13.0
Mean time to failure (MTTF) (hours)	1,400,000 @ 25°C	1,200,000 @ 25°C	1,600,000 @ 25°C	—
Annual failure rate (AFR) (percent)	0.62%	0.73%	0.55%	—
Contact start-stop cycles	—	50,000	—	>600,000
Warranty (years)	5	5	5	5
Nonrecoverable read errors per bits read	<1 sector per 10 ¹⁶	<1 sector per 10 ¹⁵	<1 sector per 10 ¹⁶	<1 sector per 10 ¹⁴
Temperature, shock (operating)	5°–55°C, 60 G	5°–55°C, 63 G	5°–55°C, 60 G	0°–60°C, 350 G
Size: dimensions (in.), weight (pounds)	1.0" × 4.0" × 5.8", 1.5 lbs	1.0" × 4.0" × 5.8", 1.4 lbs	0.6" × 2.8" × 3.9", 0.5 lbs	0.4" × 2.8" × 3.9", 0.2 lbs
Power: operating/idle/standby (watts)	15/11/—	11/8/1	8/5.8/—	1.9/0.6/0.2
GB/cu. in., GB/watt	6 GB/cu.in., 10 GB/W	43 GB/cu.in., 91 GB/W	11 GB/cu.in., 9 GB/W	37 GB/cu.in., 84 GB/W
Price in 2008, \$/GB	~ \$250, ~ \$1.70/GB	~ \$275, ~ \$0.30/GB	~ \$350, ~ \$5.00/GB	~ \$100, ~ \$0.60/GB

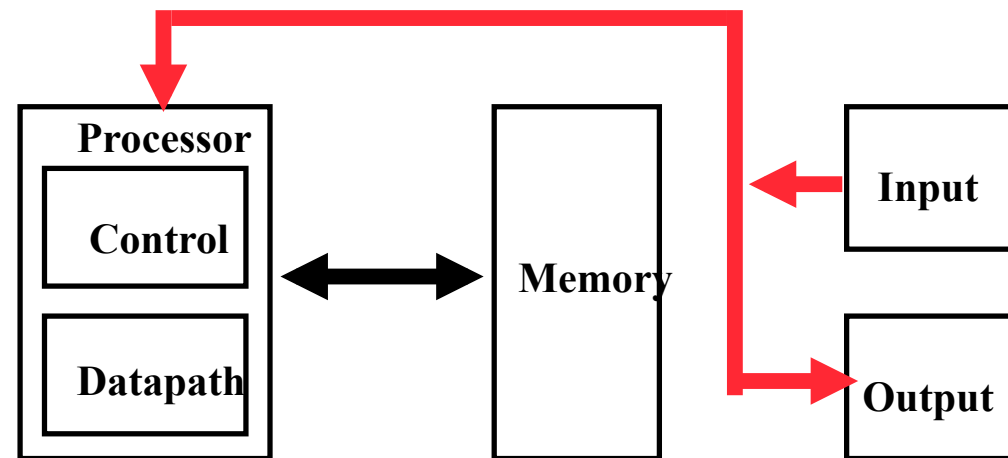
Flash Storage

- A semiconductor memory that is nonvolatile
- Latency 10-100 times faster than disk
- Smaller than disk, more power efficient
- Used on digital cameras, MP3 players
- Cost \$4 to \$10 per gigabyte (2 to 40 times higher than disk)
- Flash is popular in mobile devices because it comes in smaller capacities
- Flash is a type of electrically erasable programmable read-only memory (EEPROM)
- Flash memory bits wear out

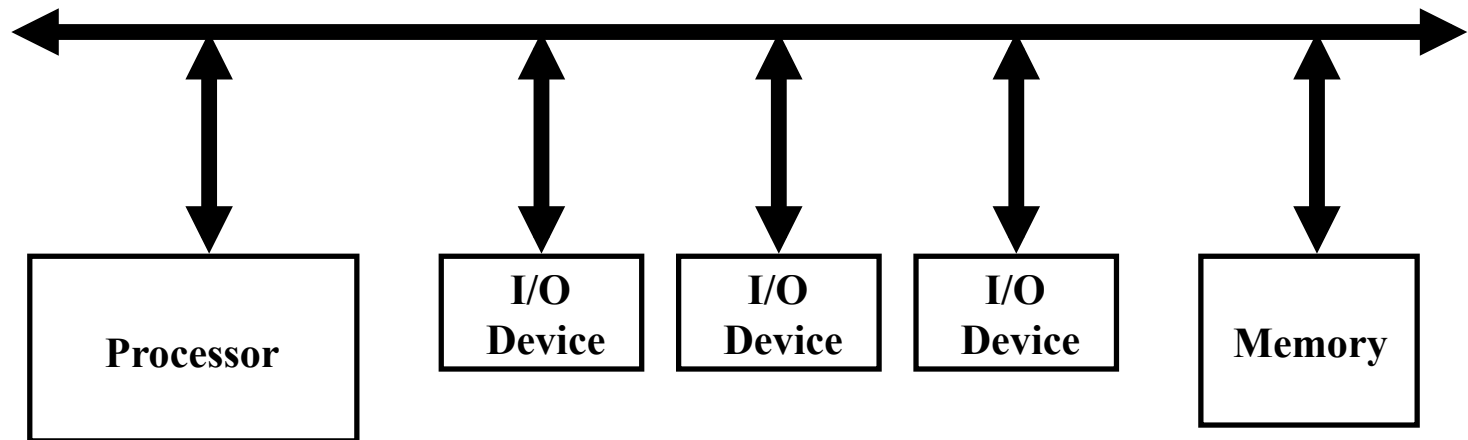
-
- In 2008 the first laptops with flash memory instead of hard disks were offered
 - Advantages: faster boot time, smaller size, longer battery life
 - Hybrid hard disks: include a flash memory (say a gigabyte) and a regular hard disk

A Bus is:

- shared communication link
- single set of wires used to connect multiple subsystems

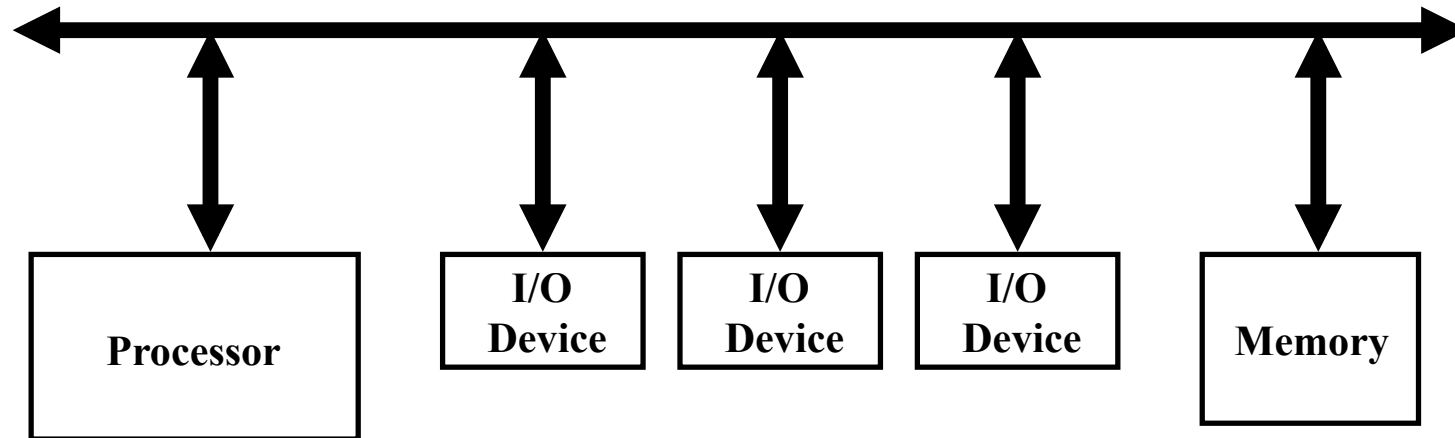


Advantages of Buses



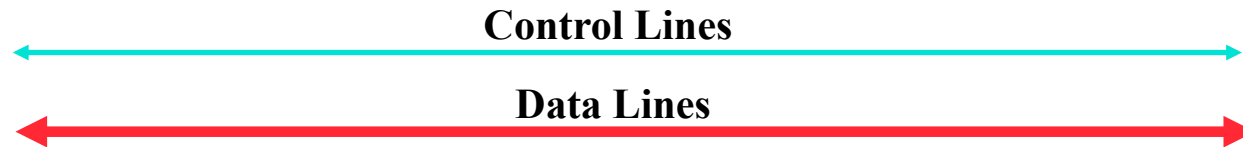
- **Versatility:**
 - New devices can be added easily
 - Peripherals can be moved between computer systems that use the same bus standard
- **Low Cost:**
 - A single set of wires is shared in multiple ways
- **Manage complexity by partitioning the design**

Disadvantage of Buses



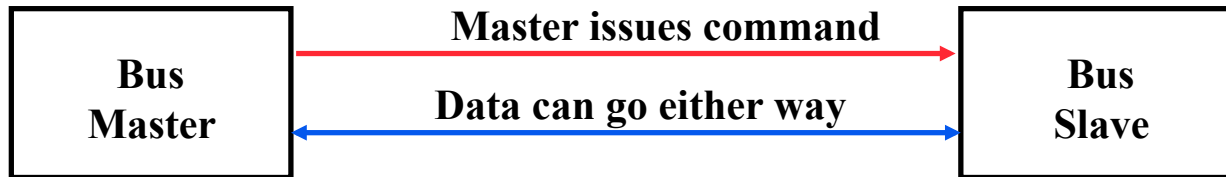
- It creates a communication bottleneck
 - The bandwidth of that bus can limit the maximum I/O throughput
- The maximum bus speed is largely limited by:
 - The length of the bus
 - The number of devices on the bus
 - The need to support a range of devices with:
 - Widely varying latencies
 - Widely varying data transfer rates

The General Organization of a Bus



- **Control lines:**
 - Signal requests and acknowledgments
 - Indicate what type of information is on the data lines
- **Data lines** carry information between the source and the destination:
 - Data and Addresses
 - Complex commands

Master versus Slave



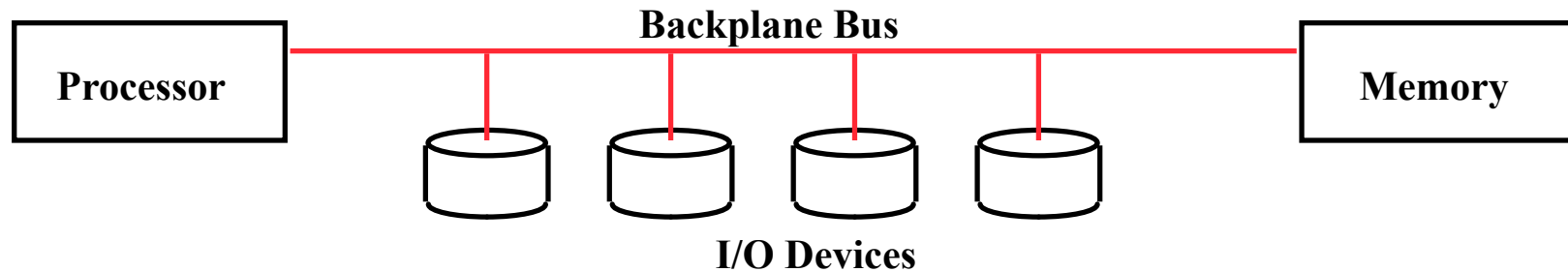
- A bus transaction includes two parts:
 - Issuing the command (and address) – request
 - Transferring the data – action
- Master is the one who starts the bus transaction by:
 - issuing the command (and address)
- Slave is the one who responds to the address by:
 - Sending data to the master if the master ask for data
 - Receiving data from the master if the master wants to send data

Types of Buses

- Processor-Memory Bus (design specific)
 - Short and high speed
 - Only need to match the memory system
 - Maximize memory-to-processor bandwidth
 - Connects directly to the processor
 - Optimized for cache block transfers
- I/O Bus (industry standard)
 - Usually is lengthy and slower
 - Need to match a wide range of I/O devices
 - Connects to the processor-memory bus or backplane bus
- Backplane Bus (standard or proprietary)
 - Backplane: an interconnection structure within the chassis
 - Allow processors, memory, and I/O devices to coexist
 - Cost advantage: one bus for all components

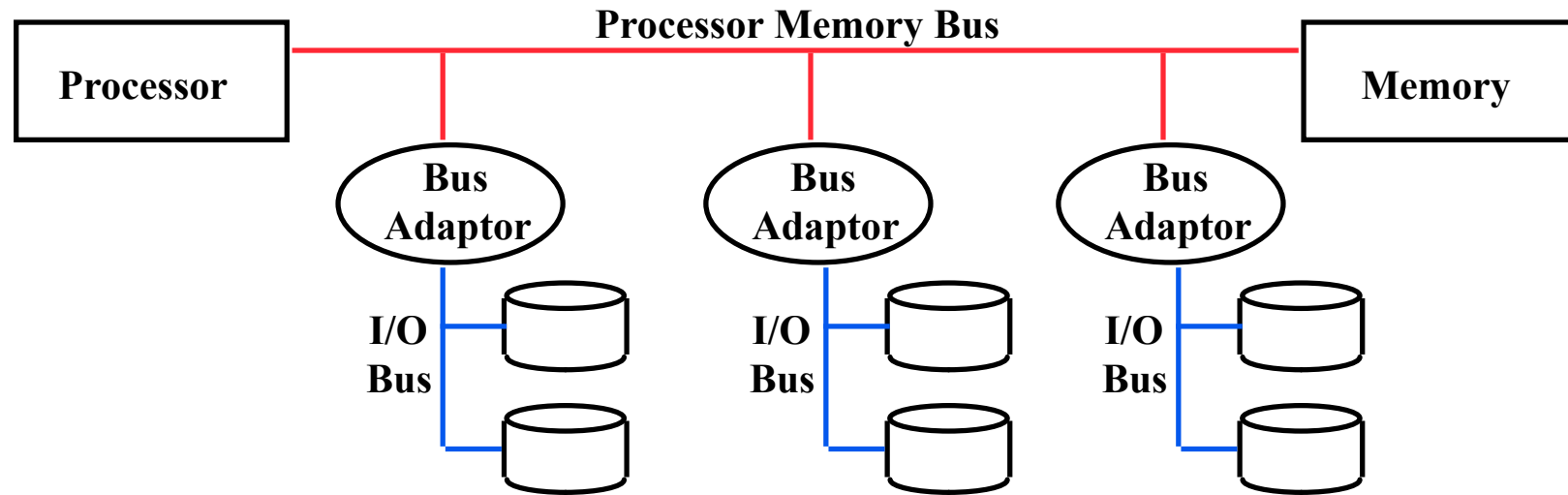
Attributes ↓	Name →	PCI	SCSI	FireWire	USB
Type of bus		Backplane	Parallel I/O	Serial I/O	Serial I/O
Standard designation		PCI	ANSI X3.131	IEEE 1394	USB 2.0
Typical application domain		System	Fast I/O	Fast I/O	Low-cost I/O
Bus width (data bits)		32-64	8-32	2	1
Peak bandwidth (MB/s)		133-512	5-40	12.5-50	0.2-15
Maximum number of devices		1024*	7-31 [#]	63	127 ^{\$}
Maximum span (m)		< 1	3-25	4.5-72 ^{\$}	5-30 ^{\$}
Arbitration method		Centralized	Self-select	Distributed	Daisy chain
Transceiver complexity or cost		High	Medium	Medium	Low

A Computer System with One Bus: Backplane Bus



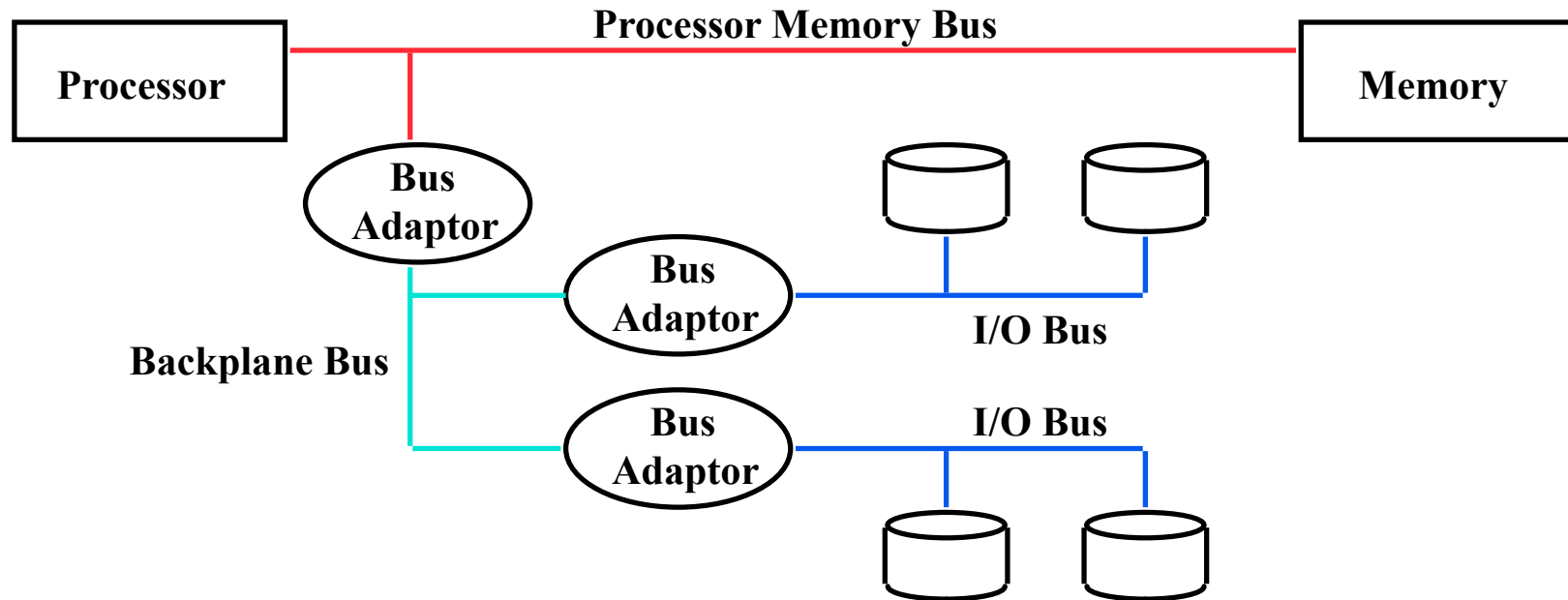
- A single bus (the backplane bus) is used for:
 - Processor to memory communication
 - Communication between I/O devices and memory
- Advantages: Simple and low cost
- Disadvantages: slow and the bus can become a major bottleneck

A Two-Bus System



- I/O buses tap into the processor-memory bus via bus adaptors:
 - Processor-memory bus: mainly for processor-memory traffic
 - I/O buses: provide expansion slots for I/O devices
- Apple Macintosh-II
 - NuBus: Processor, memory, and a few selected I/O devices
 - SCCI Bus: the rest of the I/O devices

A Three-Bus System



- A small number of backplane buses tap into the processor-memory bus
 - Processor-memory bus is used for processor memory traffic
 - I/O buses are connected to the backplane bus
- Advantage: loading on the processor bus is greatly reduced

What defines a bus?

Transaction Protocol

Timing and Signaling Specification

Bunch of Wires

Electrical Specification

**Physical / Mechanical Characteristics
– the connectors**

Synchronous and Asynchronous Bus

- Synchronous Bus:
 - Includes a clock in the control lines
 - A fixed protocol for communication that is relative to the clock
 - Advantage: involves very little logic and can run very fast
 - Disadvantages:
 - Every device on the bus must run at the same clock rate
 - To avoid clock skew, they cannot be long if they are fast
 - Processor-memory buses are often synchronous
- Asynchronous Bus:
 - It is not clocked
 - It can accommodate a wide range of devices
 - It can be lengthened without worrying about clock skew
 - It requires a handshaking protocol to coordinate transmission of data

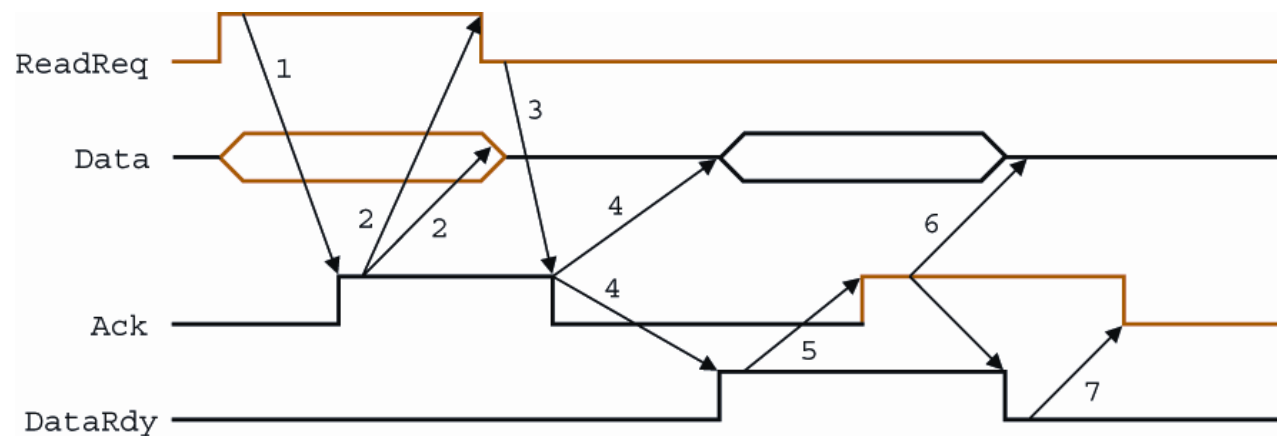
Asynchronous Bus

- Example: a device is requesting a word of data from the memory system
- Assume that there are three control lines:
 1. ReadReq: used to indicate a read request for memory. The address is put on the data lines at the same time
 2. DataRdy: used to indicate that the data word is now ready on the data lines.
 - In an output transaction, the memory will assert this signal since it is providing the data
 - In an input transaction, an I/O device would assert this signal since it would provide data
 3. Ack: used to acknowledge the ReadReq or DataRdy signal of the other party
- In an asynchronous protocol, the control signals ReadReq and DataRdy are asserted until the other party indicates that the control lines have been seen and the data lines have been read

Asynchronous Bus

I/O device signals a request by raising ReadReq and putting the addr on the data lines

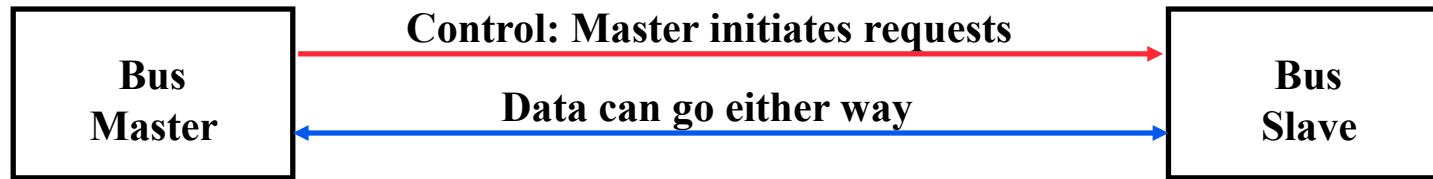
1. Memory sees ReadReq, reads addr from data lines, and raises Ack
2. I/O device sees Ack and releases the ReadReq and data lines
3. Memory sees ReadReq go low and drops Ack
4. When memory has data ready, it places it on data lines and raises DataRdy
5. I/O device sees DataRdy, reads the data from data lines, and raises Ack
6. Memory sees Ack, releases the data lines, and drops DataRdy
7. I/O device sees DataRdy go low and drops Ack



Increasing the Bus Bandwidth

- Separate versus multiplexed address and data lines:
 - Address and data can be transmitted in one bus cycle if separate address and data lines are available
 - Cost: (a) more bus lines, (b) increased complexity
- Data bus width:
 - By increasing the width of the data bus, transfers of multiple words require fewer bus cycles
 - Cost: more bus lines
- Block transfers:
 - Allow the bus to transfer multiple words in back-to-back bus cycles
 - Only one address needs to be sent at the beginning
 - The bus is not released until the last word is transferred
 - Cost: (a) increased complexity
(b) decreased response time for request

Arbitration: Obtaining Access to the Bus



- One of the most important issues in bus design:
 - How is the bus reserved by a devices that wishes to use it?
- Chaos is avoided by a master-slave arrangement:
 - Only the bus master(s) can control access to the bus:
 - It initiates and controls all bus requests
 - A slave responds to read and write requests
- The simplest system:
 - Processor is the only bus master
 - All bus requests must be controlled by the processor
 - Major drawback: the processor is involved in every transaction

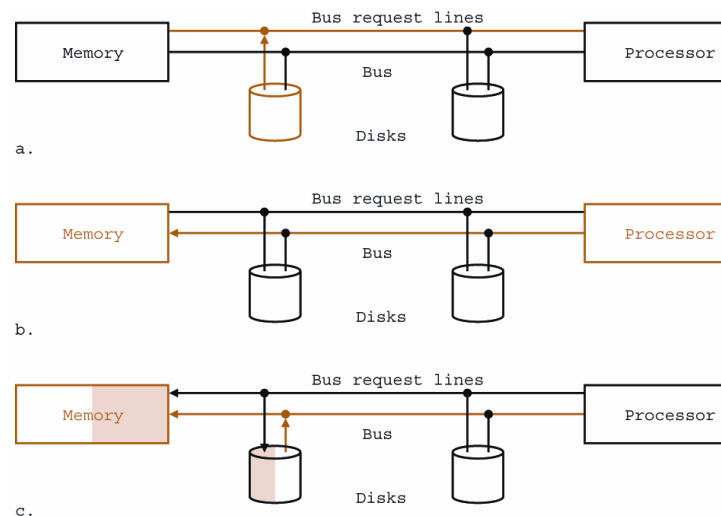
Single Master Bus Transaction

All bus requests are controlled by the processor
it initiates the bus cycle on behalf of the requesting
device

Step 1: Disk wants to use the bus so it generates a bus request to processor

Step 2: Processor responds and generates appropriate control signals

Step 3: Processor gives slave (disk) permission to use the bus



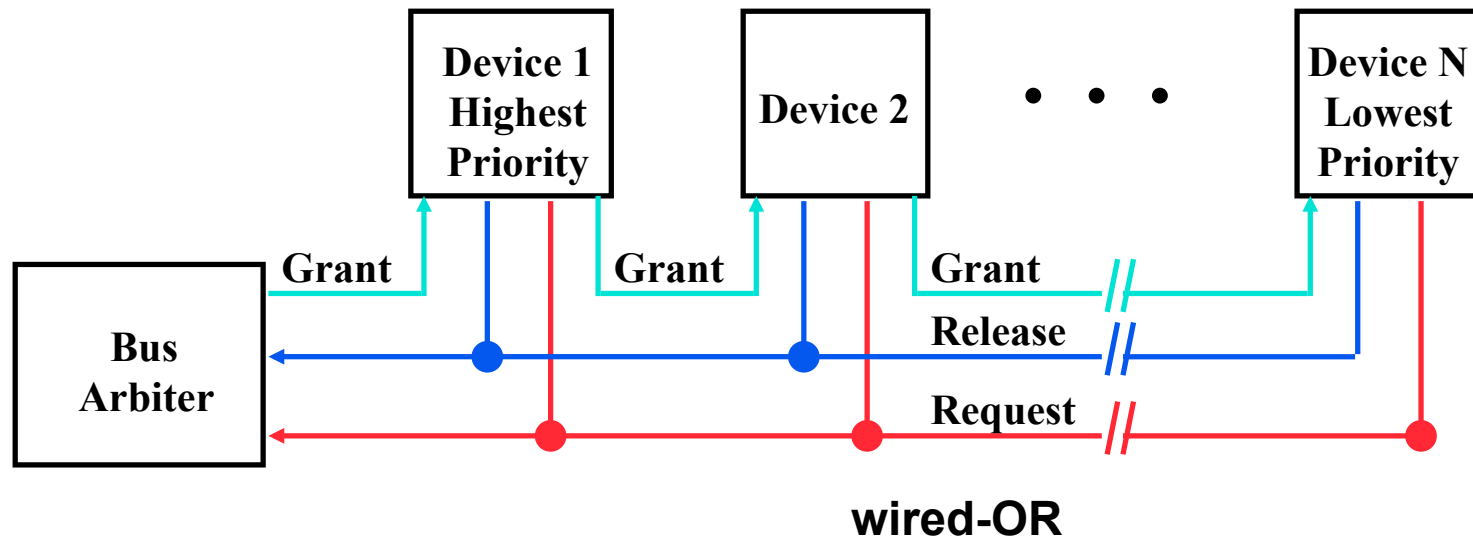
Multiple Potential Bus Masters: the Need for Arbitration

- Bus arbitration: deciding which bus master gets to use the bus next
- Bus arbitration scheme:
 - A bus master wanting to use the bus asserts the bus request
 - A bus master cannot use the bus until its request is granted
 - A bus master must signal to the arbiter after finish using the bus
- Bus arbitration schemes usually try to balance two factors:
 - Bus priority: the highest priority device should be serviced first
 - Fairness: Even the lowest priority device should never be completely locked out from the bus

Multiple Potential Bus Masters: the Need for Arbitration

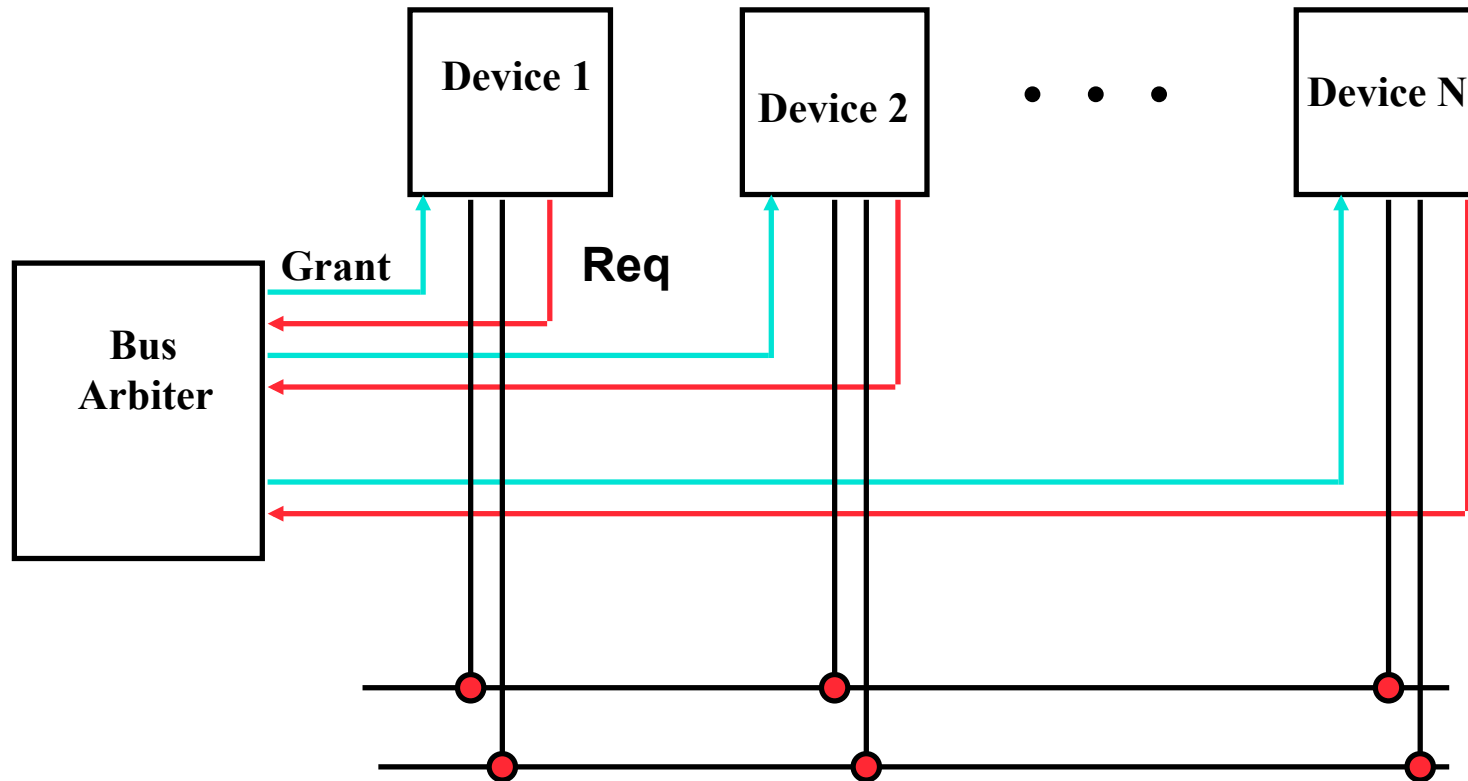
- Bus arbitration schemes can be divided into four broad classes:
 - Daisy chain arbitration: bus grant line runs through the devices from the highest priority to lowest (priorities are determined by the position on the bus)
 - Centralized, parallel arbitration: see next-next slide
 - Distributed arbitration by self-selection: each device wanting the bus places a code indicating its identity on the bus.
 - By examining the bus, devices can determine the highest priority device
 - There is no need for central arbitrator
 - Distributed arbitration by collision detection: Ethernet uses this.
 - Each device independently requests the bus
 - Simultaneous requests result in collision
 - Collision is detected and a scheme for selecting among the colliding parties is used

The Daisy Chain Bus Arbitrations Scheme



- Advantage: simple
- Disadvantages:
 - Cannot assure fairness:
 - A low-priority device may be locked out indefinitely
 - The use of the daisy chain grant signal also limits the bus speed

Centralized Parallel Arbitration



- Used in essentially all processor-memory busses and in high-speed I/O busses
- Used in PCI

Responsibilities of the Operating System

- The operating system acts as the interface between:
 - The I/O hardware and the program that requests I/O
- Three characteristics of the I/O systems:
 - The I/O system is shared by multiple program using the processor
 - I/O systems often use interrupts (externally generated exceptions) to communicate information about I/O operations.
 - Interrupts must be handled by the OS because they cause a transfer to supervisor mode
 - The low-level control of an I/O device is complex:
 - Managing a set of concurrent events
 - The requirements for correct device control are very detailed

Operating System Requirements

- Provide protection to shared I/O resources
 - Guarantees that a user's program can only access the portions of an I/O device to which the user has rights
- Provides abstraction for accessing devices:
 - Supply routines that handle low-level device operation
- Handles the interrupts generated by I/O devices
- Provide equitable access to the shared I/O resources
 - All user programs must have equal access to the I/O resources
- Schedule accesses in order to enhance system throughput

OS and I/O Systems Communication Requirements

- The Operating System must be able to prevent:
 - The user program from communicating with the I/O device directly
- If user programs could perform I/O directly:
 - Protection to the shared I/O resources could not be provided
- Three types of communication are required:
 - The OS must be able to give commands to the I/O devices
 - The I/O device must be able to notify the OS when the I/O device has completed an operation or has encountered an error
 - Data must be transferred between memory and an I/O device

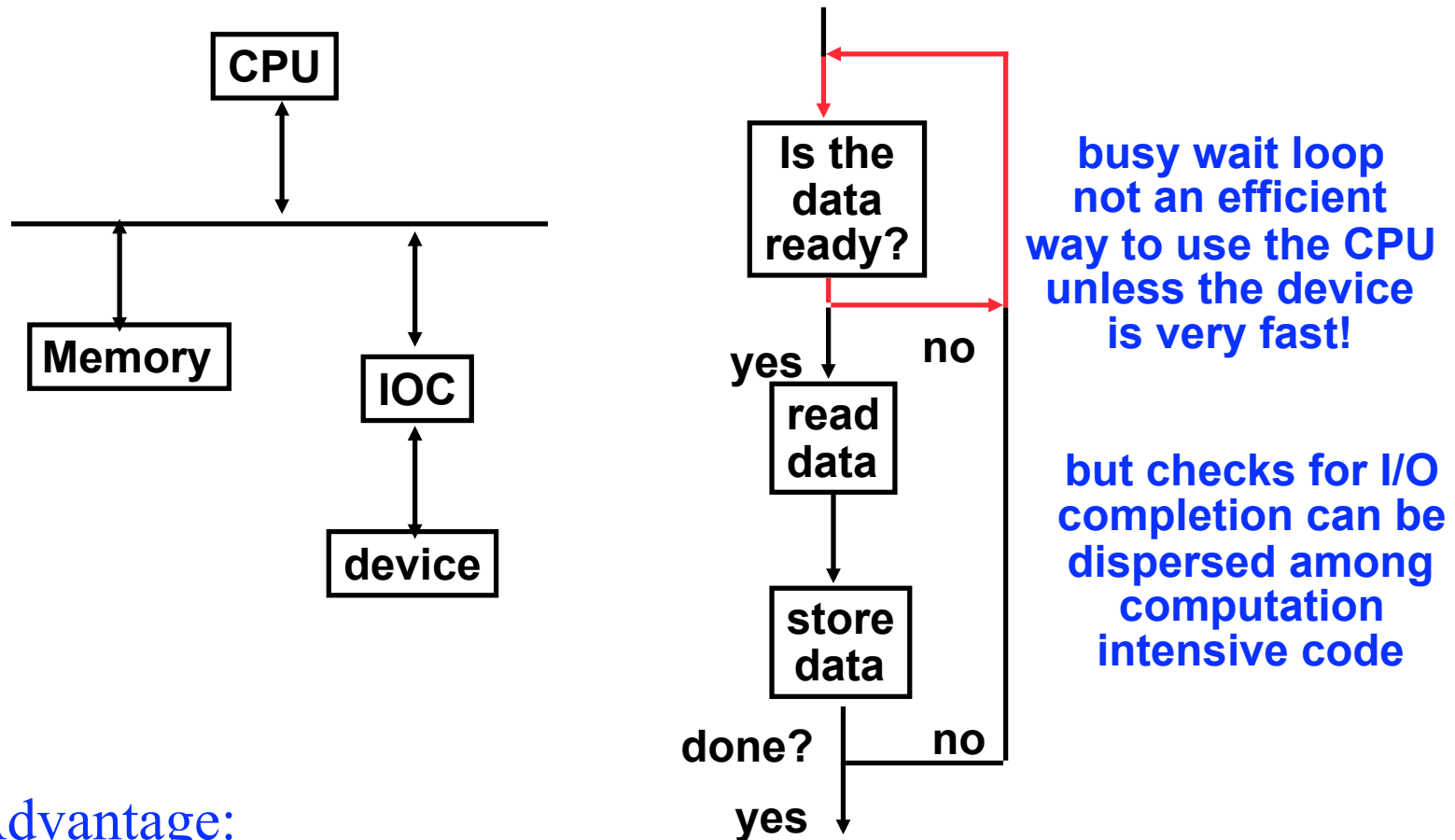
Giving Commands to I/O Devices

- To give a command to an I/O device, processor must be able to address the device
- Two methods are used to address the device:
 - Special I/O instructions
 - Memory-mapped I/O
- Special I/O instructions specify:
 - Both the device number and the command word
 - Device number: the processor communicates this via a set of wires normally included as part of the I/O bus
 - Command word: this is usually send on the bus's data lines
- Memory-mapped I/O:
 - Portions of the address space are assigned to I/O device
 - Read and writes to those addresses are interpreted as commands to the I/O devices
 - User programs are prevented from issuing I/O operations directly:
 - The I/O address space is protected by the address translation

I/O Device Notifying the OS

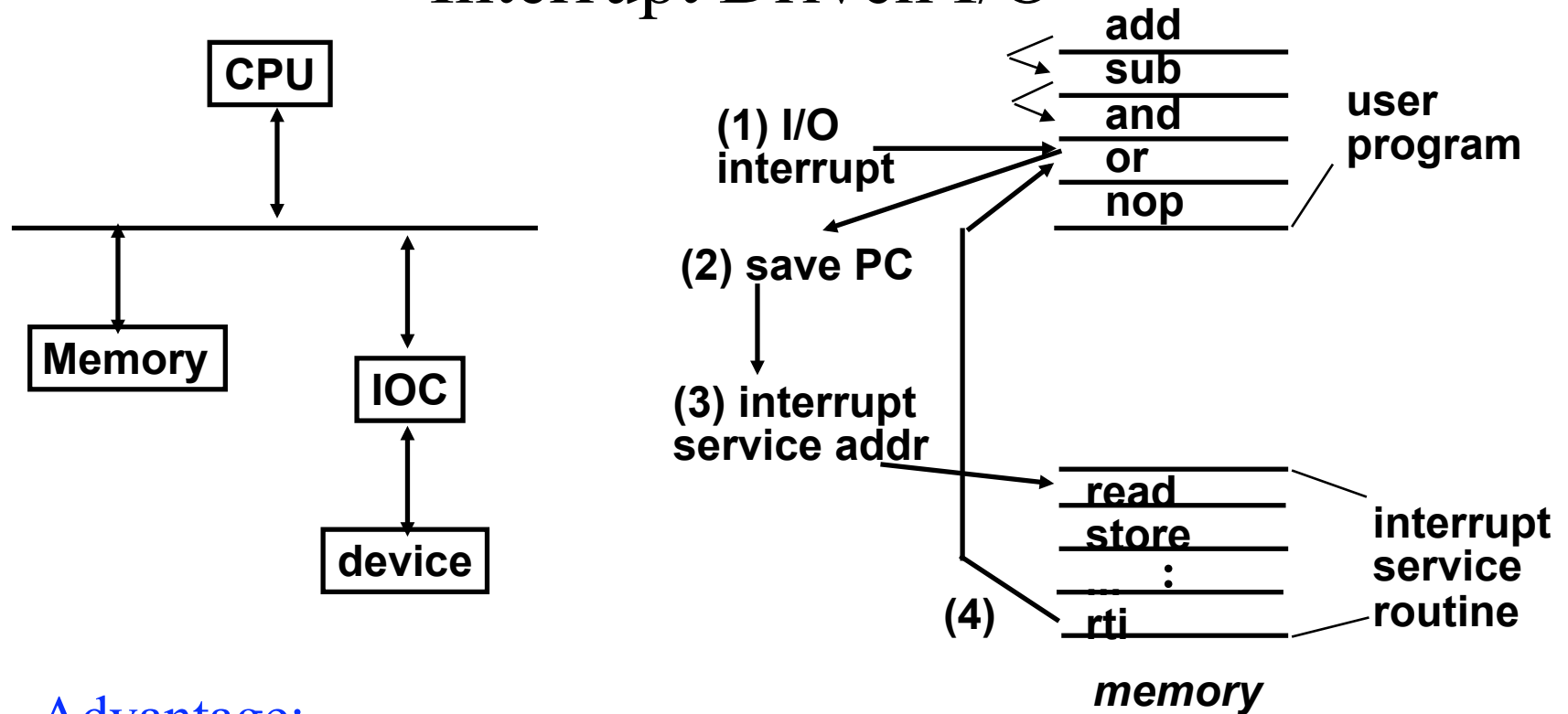
- The OS needs to know when:
 - The I/O device has completed an operation
 - The I/O operation has encountered an error
- This can be accomplished in two different ways:
 - Polling:
 - The I/O device put information in a status register
 - The OS periodically check the status register
 - I/O Interrupt:
 - Whenever an I/O device needs attention from the processor, it interrupts the processor from what it is currently doing.

Polling: Programmed I/O



- Advantage:
 - Simple: the processor is totally in control and does all the work
- Disadvantage:
 - Polling overhead can consume a lot of CPU time

Interrupt Driven I/O



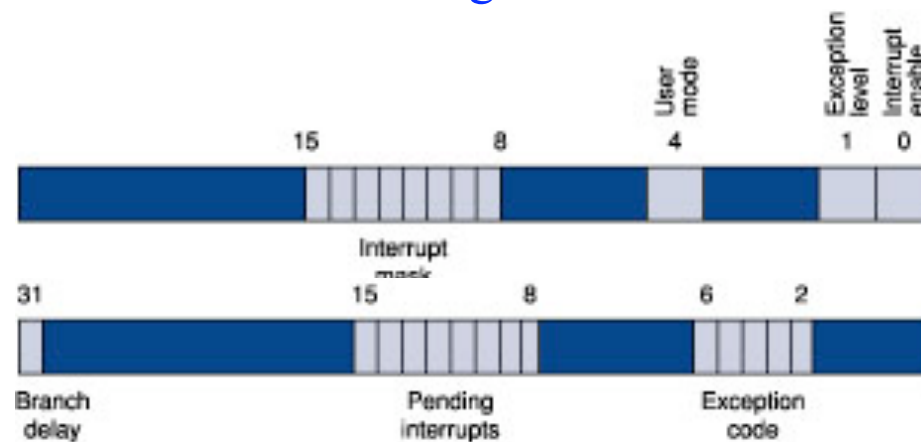
- Advantage:
 - User program progress is only halted during actual transfer
- Disadvantage, special hardware is needed to:
 - Cause an interrupt (I/O device)
 - Detect an interrupt (processor)
 - Save the proper states to resume after the interrupt (processor)

I/O Interrupt

- An I/O interrupt is just like the exceptions except:
 - An I/O interrupt is asynchronous
 - Further information needs to be conveyed
- An I/O interrupt is asynchronous with respect to instruction execution:
 - I/O interrupt is not associated with any instruction
 - I/O interrupt does not prevent any instruction from completion
 - You can pick your own convenient point to take an interrupt
- I/O interrupt is more complicated than exception:
 - Needs to convey the identity of the device generating the interrupt
 - Interrupt requests can have different urgencies:
 - Interrupt request needs to be prioritized

Interrupt priority levels

- To deal with different priorities of I/O devices, most interrupt mechanisms have several levels of priority
- The Status register determines who can interrupt the computer
- If the interrupt bit is 0, none can interrupt
- Interrupt mask: there is a bit corresponding to each bit in the pending interrupt field of the Cause register
- To enable the corresponding interrupt, there must be a 1 in the mask field at that position
- Once an interrupt occurs the operating system can find the reason in the exception code field of the Cause register



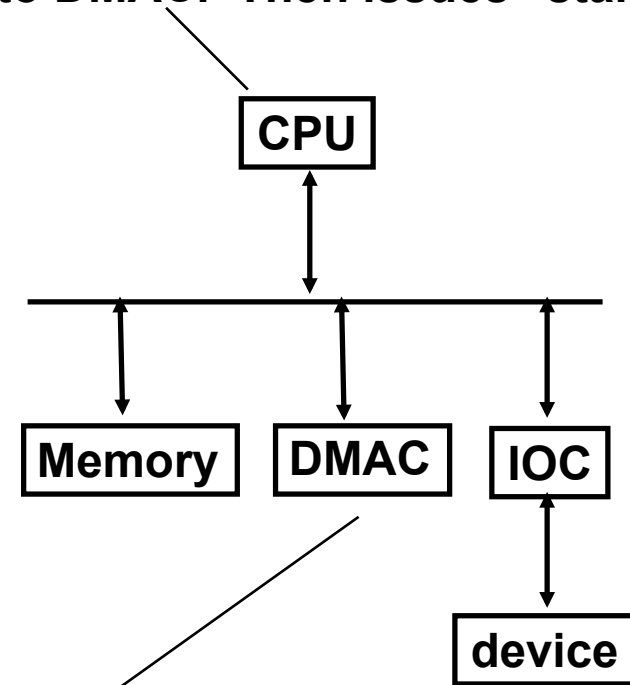
Interrupt priority levels

- Steps in handling an interrupt:
 1. Logically AND pending interrupt field and the interrupt mask to see which enabled interrupt could be the culprit
 2. Select the higher priority interrupt
 3. Save the interrupt mask field of the Status register
 4. Change the interrupt mask field to disable all interrupt of equal or lower priority
 5. Save the processor state needed to handle the interrupt
 6. To allow higher-priority interrupts, set the interrupt enable bit of the Status register to 1
 7. Call the appropriate interrupt routine
 8. Before restoring state, set the interrupt enable bit of the Status register to 0. This allows you to restore the interrupt mask filed

Delegating I/O Responsibility from the CPU: DMA

- **Direct Memory Access (DMA):** a mechanism that provides a device controller the ability to transfer data directly to or from the memory without involving the processor
- **Direct Memory Access (DMA):**
 - External to the CPU
 - Act as a master on the bus
 - Transfer blocks of data to or from memory without CPU intervention

CPU sends a starting address, direction, and length count to DMAC. Then issues "start".



DMAC provides handshake signals for Peripheral Controller, and Memory Addresses and handshake signals for Memory.

RAID

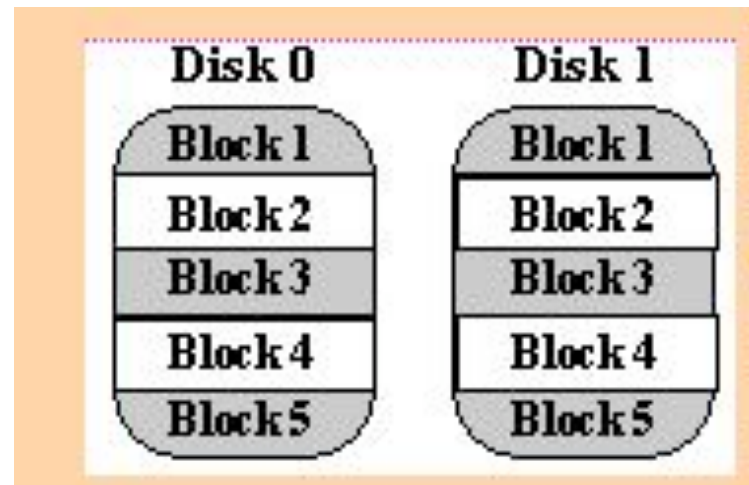
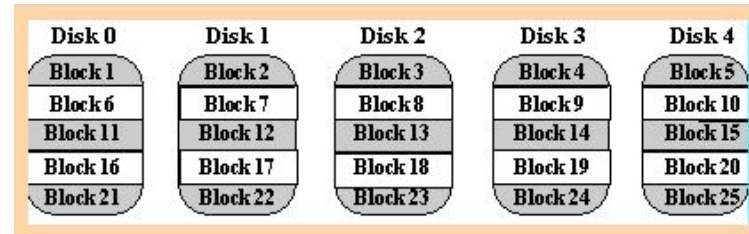
- Replace Small Number of Large Disks with Large Number of Small Disks! (1988 Disks)
- Disk Arrays have potential for large data and I/O rates, high MB per cu. ft., high MB per KW, but what about reliability?
- Reliability of N disks = Reliability of 1 Disk \div N
- Example: 50,000 Hours \div 70 disks = 700 hours
 - Disk system MTTF: Drops from 6 years to 1 month!
- Arrays (without redundancy) too unreliable to be useful!

RAID

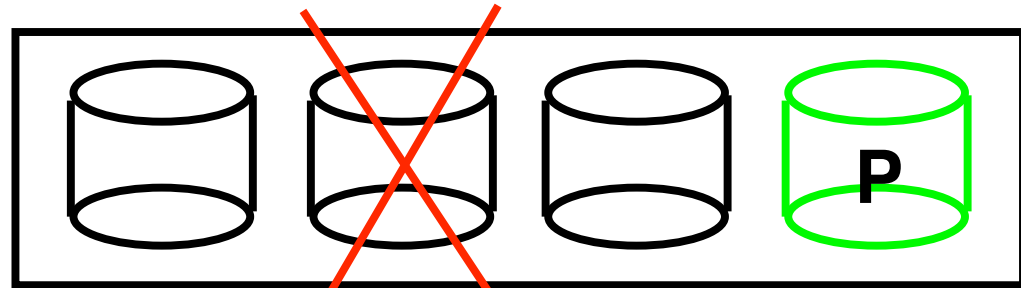
- RAID == Redundant Array of Inexpensive Disks
 - the idea is to maintain multiple copies of the data.
- Files are "striped" across multiple disks
- Redundancy yields high data availability
- Disks will still fail
- Contents reconstructed from data redundantly stored in the array
 - ⇒ Capacity penalty to store redundant info
 - ⇒ Bandwidth penalty to update redundant info

Disk Dependability

- RAID 0 - No redundancy - stripe data over multiple disks but still only one copy of any particular data item.
- RAID 1 - Simple mirroring - write all data to two disks, read back from either one of the disks.
- RAID 2: uses an error detection and correction scheme (no longer in use).



RAID 3: Parity Disk



Striped physical records

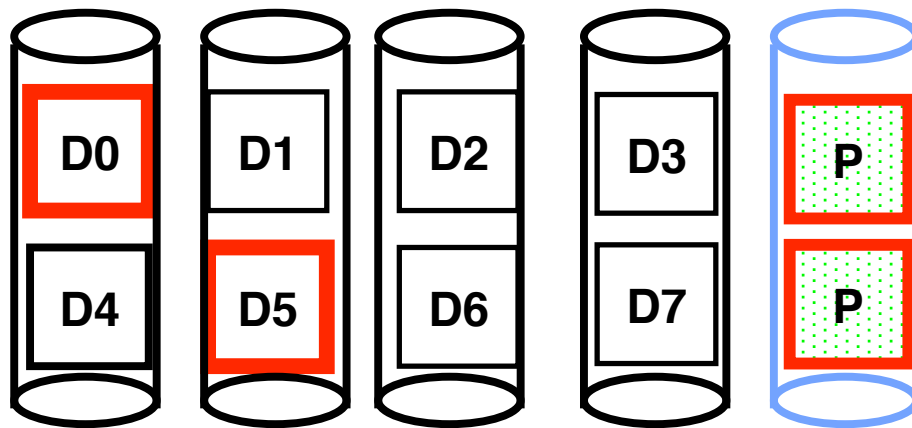
1	1	1	1
0	1	0	1
1	0	1	0
0	0	0	0
0	1	0	1
0	1	0	1
1	0	1	0
1	1	1	1

P contains sum of other disks per stripe mod 2 (“**parity**”)
If disk fails, subtract P from sum of other disks to find missing information

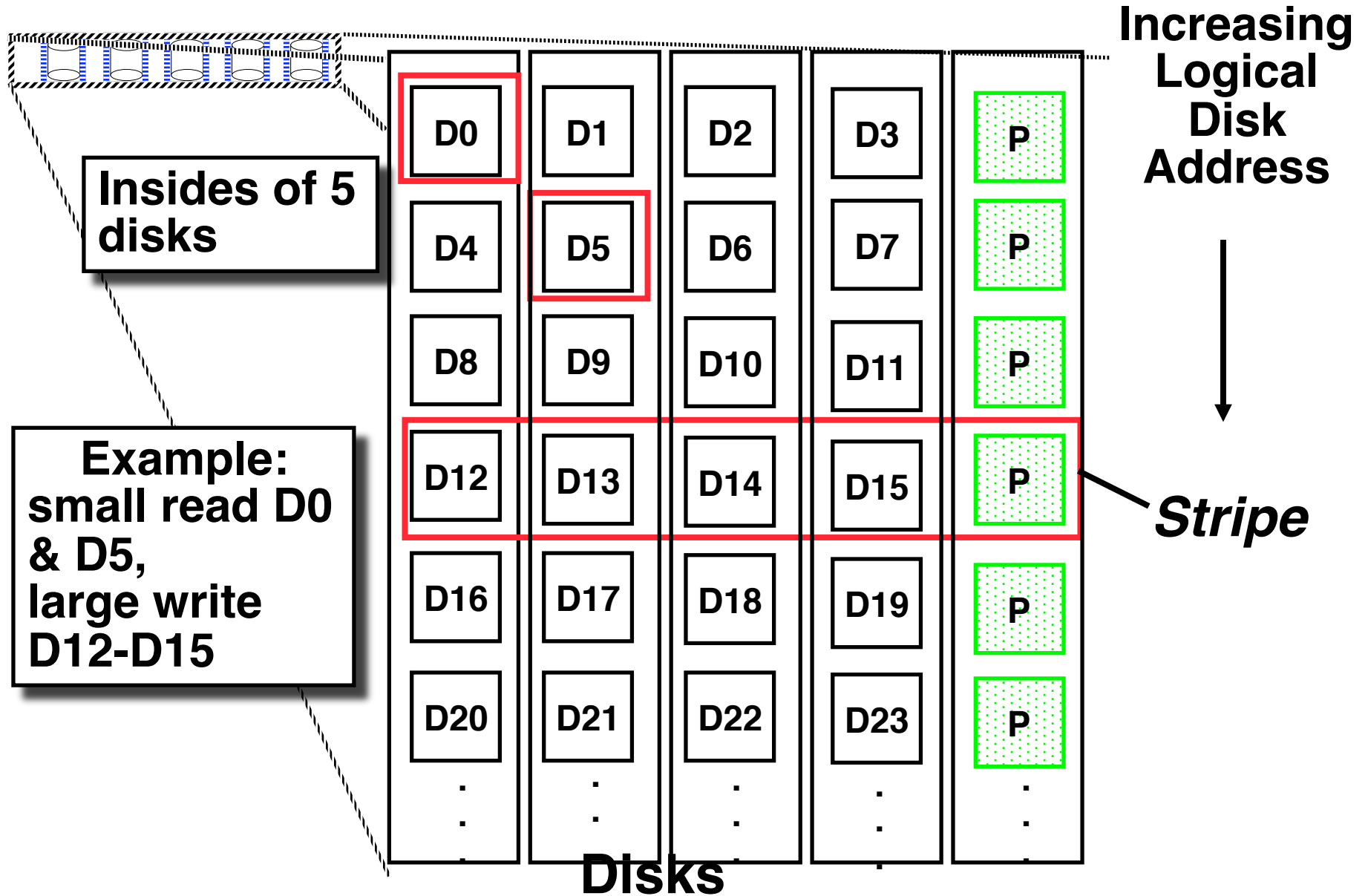
RAID 3: Parity Disk

- Sum computed across protection group to protect against hard disk failures, stored in P disk
- Read and writes go to all the disks in the group
- Logically, a single high capacity, high transfer rate disk: good for large transfers
- Wider arrays reduce capacity costs, but decreases availability
- 33% capacity cost for parity in this configuration

-
- Allows independent reads to different disks simultaneously
 - RAID 4 works well for small reads
 - Small writes (write to one disk):
 - Option 1: read other data disks, create new sum and write to Parity Disk
 - Option 2: since P has old sum, compare old data to new data, add the difference to P
 - Small back to back writes are limited by Parity Disk: Write to D0, D5 both also write to P disk



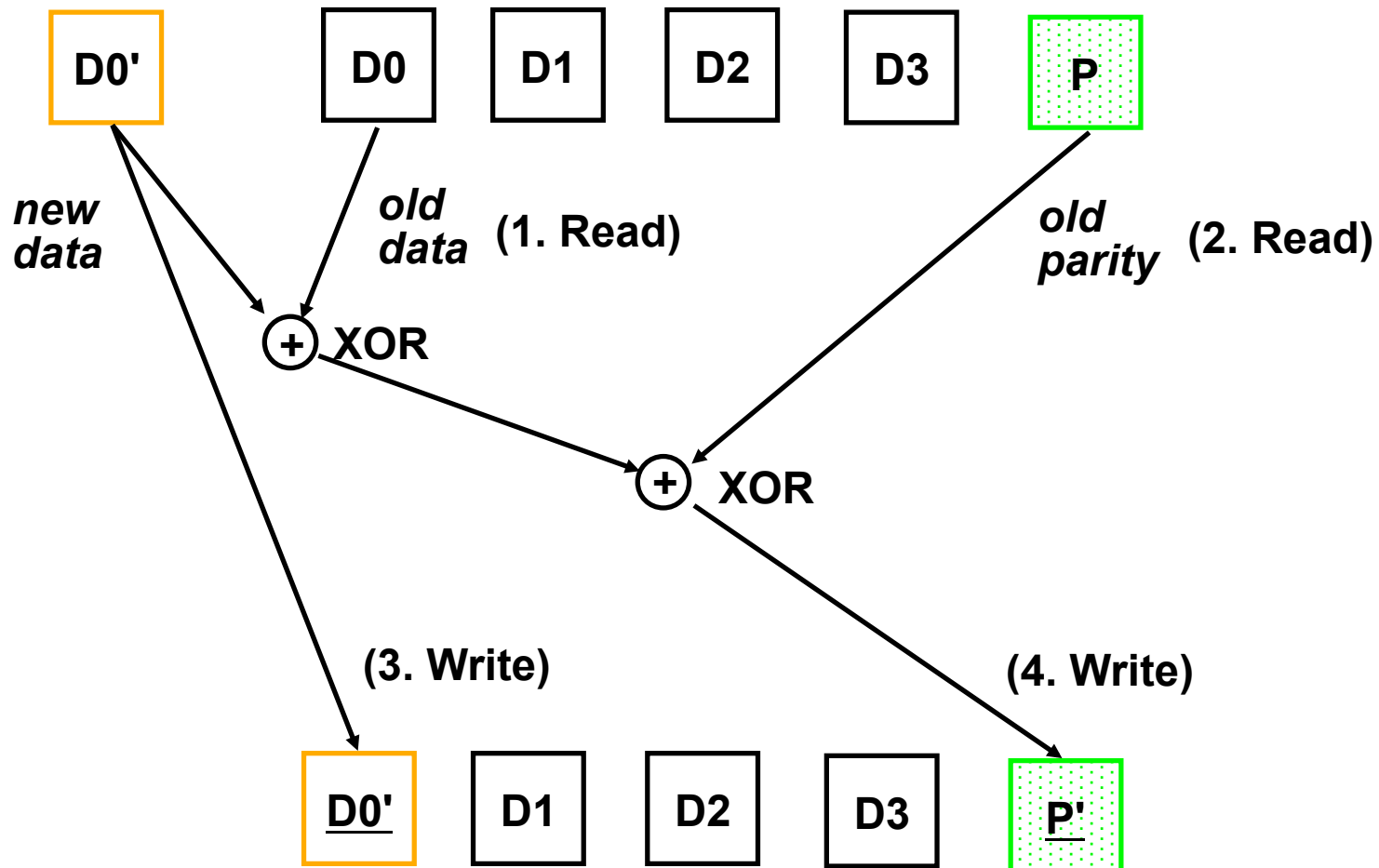
Disks RAID 4: High I/O Rate Parity



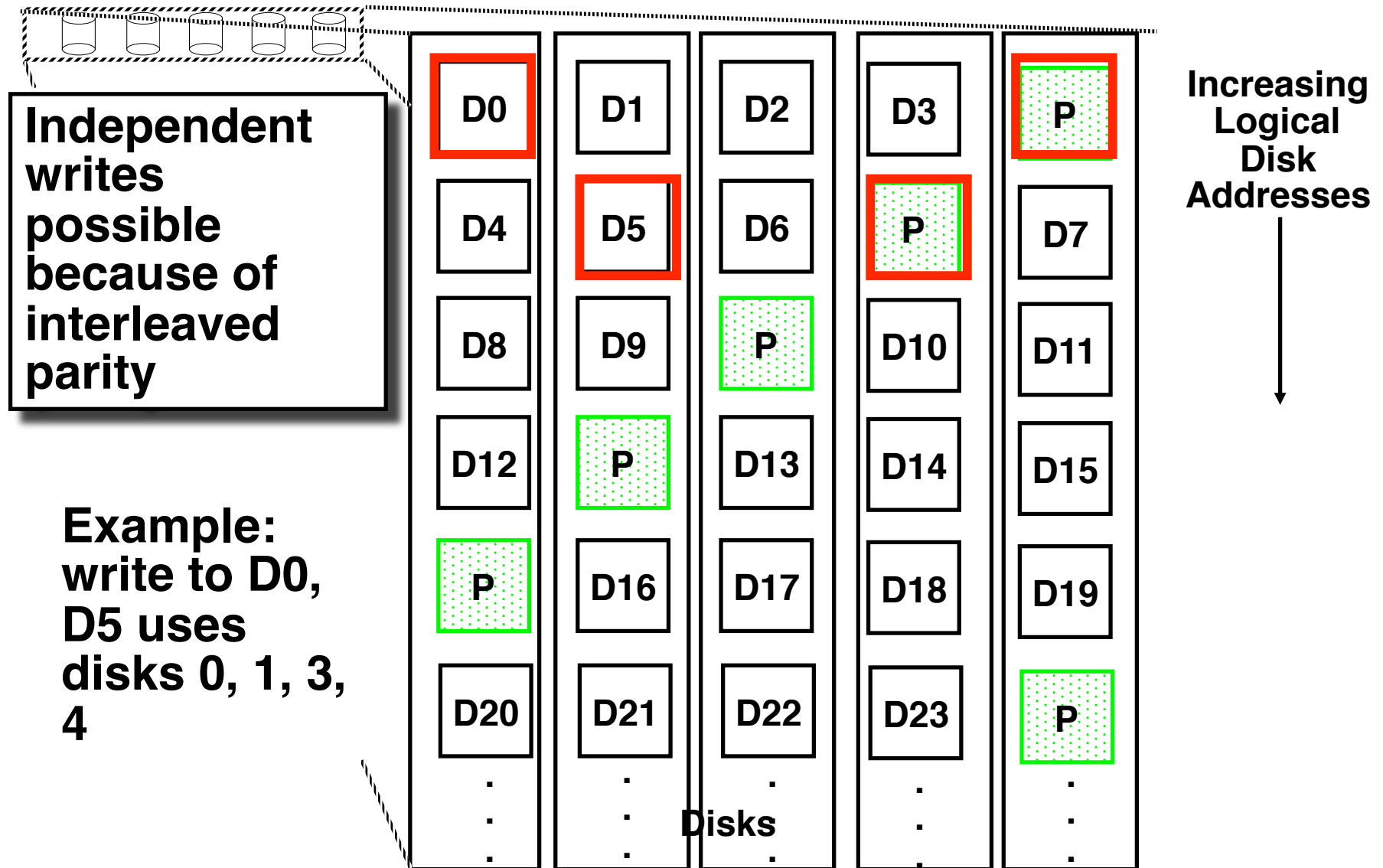
Problems of Disk Arrays: Small Writes

RAID-4: Small Write Algorithm

1 Logical Write = 2 Physical Reads + 2 Physical Writes



RAID 5: High I/O Rate Interleaved Parity



RAID 6

- Parity based schemes protect against a single self-identifying failure
- When a single failure correction is not sufficient, parity can be generalized to have a second calculation over the data and another check disk of information
- The small write short cut works but there will be six disk accesses.

RAID summary

- RAID 1 and RAID 5 are widely used
- Weaknesses of RAID system:
 - Array design should allow the failed disk to be replaced without having to turn off the system (hot swapping)
 - Another failure could occur during repair so the repair time affects the chance of losing data (it is better to have a standby spare)
 - Correlated failure due to problem with the environment (e.g., failure of air conditioning)