

COMP3301

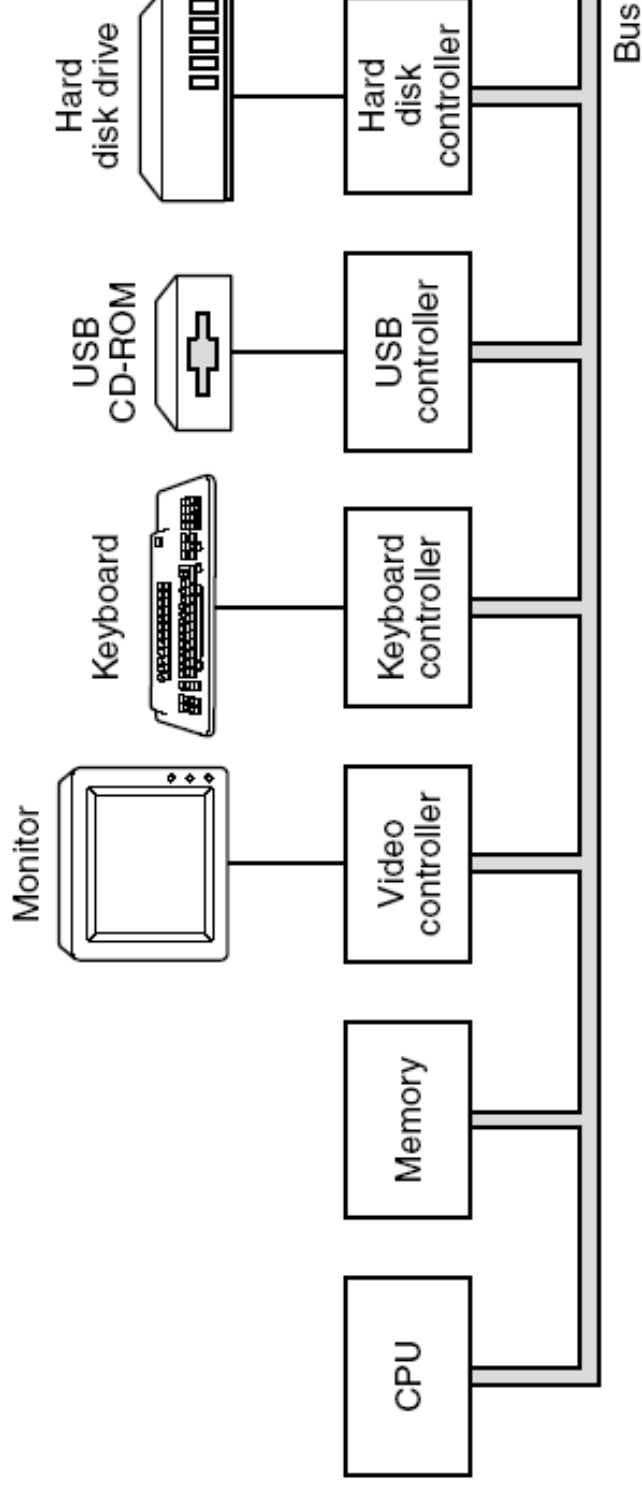
Lecture 5

Deadlock, Disks

School of Information Technology and Electrical Engineering
The University of Queensland

Device controllers

- Mechanical component
 - Connectors etc
- Electronic component
 - device controller



Resources

- The sequence of events required to use a resource:
 1. Request the resource.
 2. Use the resource.
 3. Release the resource.

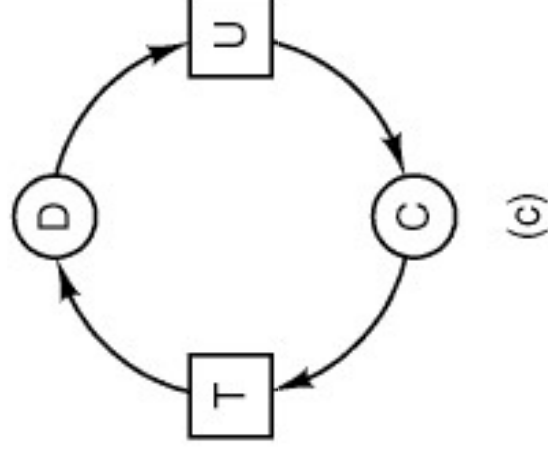
Definition of Deadlock

- *A set of processes is deadlocked if each process in the set is waiting for an event that only another process in the set can cause.*

Deadlock

- Requirements:
 - Mutual exclusion
 - Hold and wait
 - No preemption
 - Circular wait
- All must be present!

Deadlock Modeling

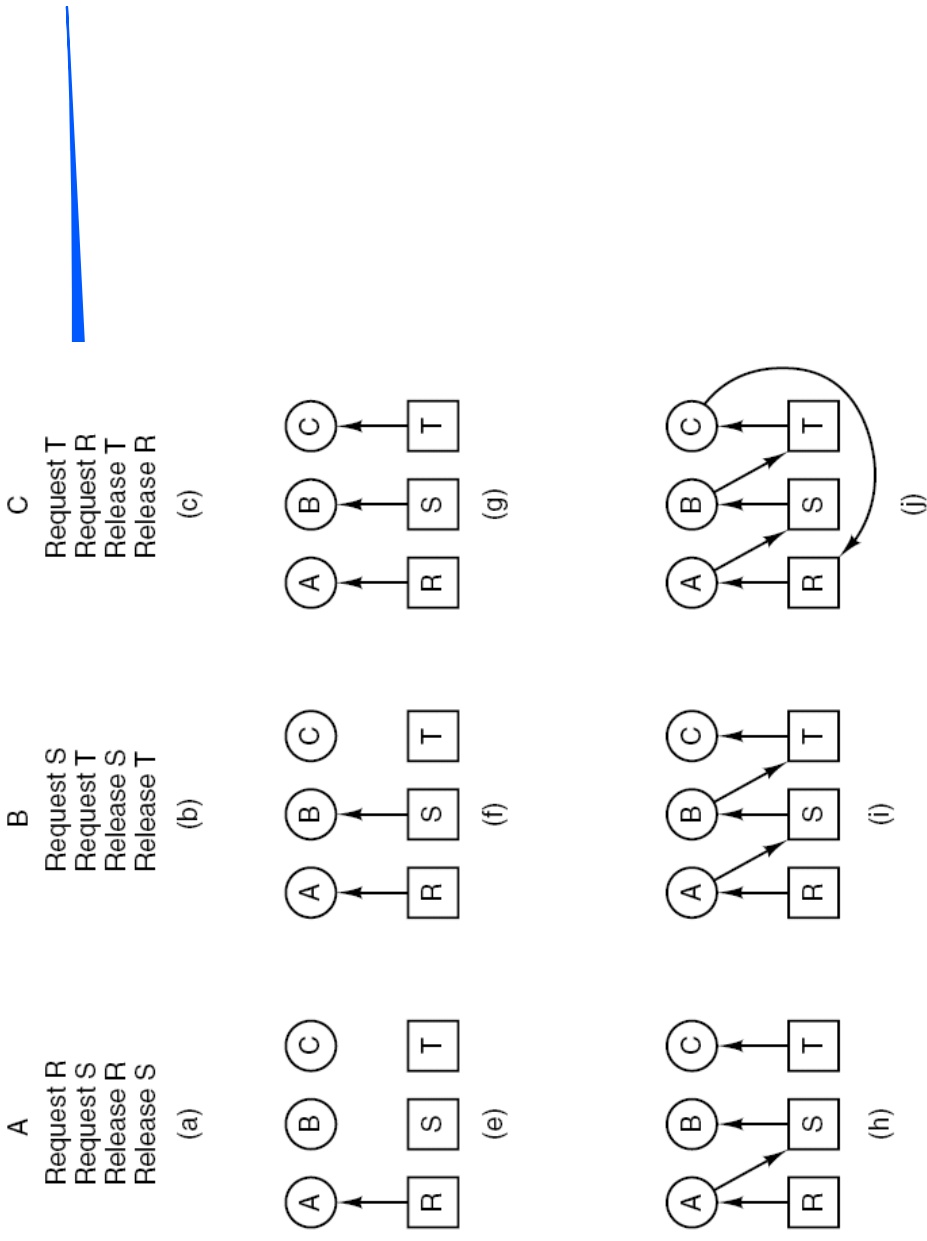


- Figure 3-9. Resource allocation graphs.
- (a) Process (A) Holding a resource (R).
- (b) Requesting a resource. (c) Deadlock.

Deadlock Avoidance (1)

1. A requests R
2. B requests S
3. C requests T
4. A requests S
5. B requests T
6. C requests R
deadlock

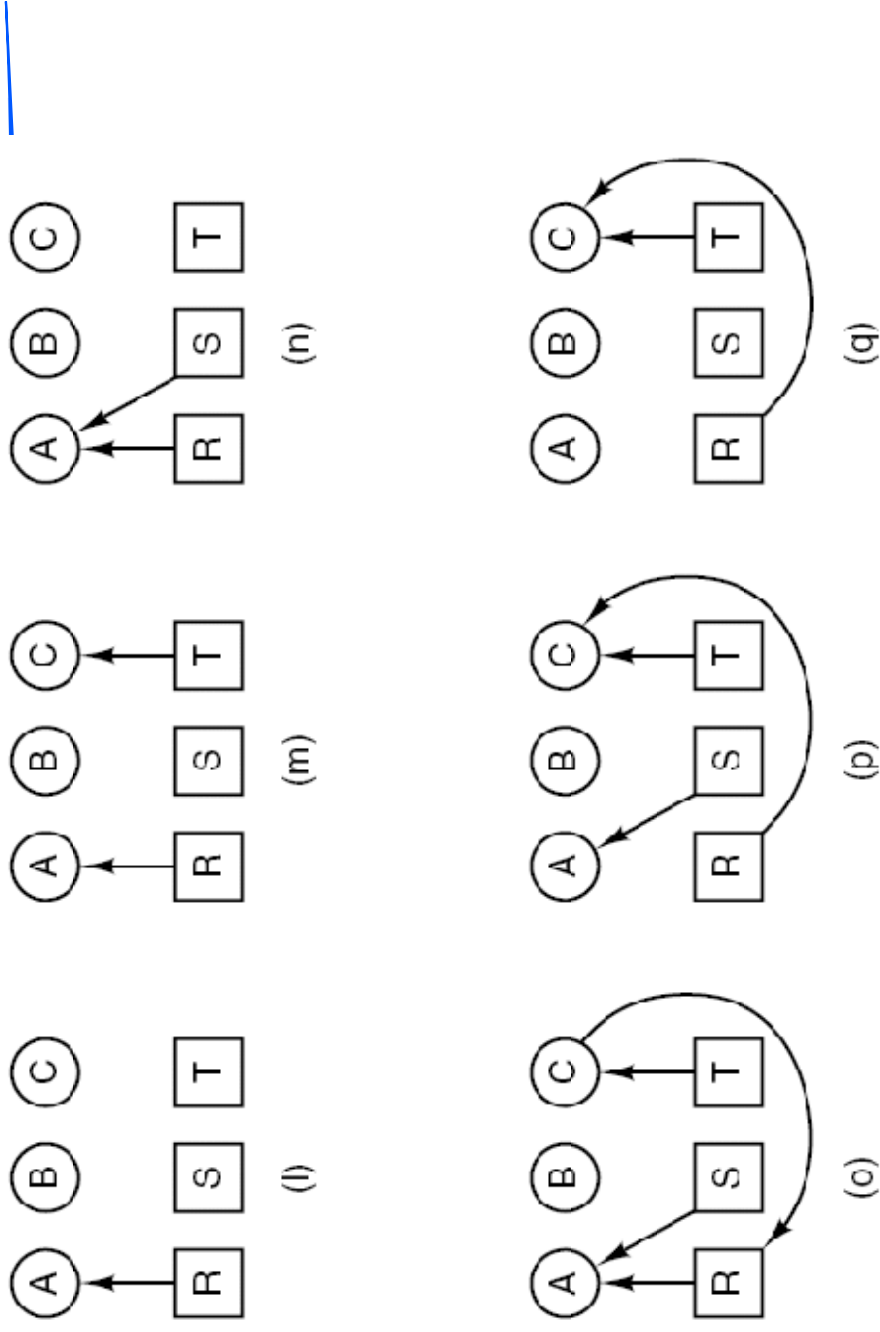
(d)



- Figure 3-10. An example of how deadlock occurs and how it can be avoided.

Deadlock Avoidance (2)

1. A requests R
 2. C requests T
 3. A requests S
 4. C requests R
 5. A releases R
 6. A releases S
- no deadlock



- Figure 3-10. An example of how deadlock occurs and how it can be avoided.

Four Possible Strategies

1. Ignore the problem
2. Detection & Recovery – identify when deadlocks occur & recover
3. Dynamic Deadlock Avoidance by careful resource allocation
4. Prevention, by structurally negating one of the four conditions necessary to cause deadlock

“Ostrich” Algorithm

- Stick your head in the sand and ignore the problem.
- The most commonly adopted solution! (eg. MINIX, Linux)
- If it is rare, then not worth the effort to fix it – system will crash & problem goes away.
- Not good enough for safety-critical systems.

Detection & Recovery

- Keep a graph of requested & allocated resources
- Identify cycles
- Kill processes until the cycle is broken
- Even easier – look for “zombie” process groups that have been blocked for a long time (eg. an hour)

Deadlock Prevention

- Negate one of the conditions.
- Mutual exclusion – can't always negate, but spoolers can help (only one process ever has access to the printer, and it never needs any other resource)
- No hold and wait
 - – request all resources before starting, and get all or none.
 - Hard to know what is needed, resources locked for a long time.
 - OR Release all before asking for new resource

Deadlock Prevention

- No pre-emption: hard to negate this without undesirable properties. Requires ability to “roll-back” a process.
- Avoid circular wait – number all resources, and only ask in ascending order. Alternatively, can’t ask for anything lower in number than currently held.

Deadlock Prevention

Condition	Approach
Mutual exclusion	Spool everything
Hold and wait	Request all resources initially
No preemption	Take resources away
Circular wait	Order resources numerically

- Figure 3-12. Summary of approaches to deadlock prevention.

Deadlock Avoidance

- Banker's Algorithm
- Based on small-town bank with limited loan funds for multiple lines of credit.
- Before allocating any resources, ensure that there are enough resources for at least one task to complete (loan to be completed and repaid)

The Banker's Algorithm for a Single Resource

	Has	Max
A	0	6
B	0	5
C	0	4
D	0	7

Free: 10

(a)

	Has	Max
A	1	6
B	1	5
C	2	4
D	4	7

Free: 2

(b)

	Has	Max
A	1	6
B	2	5
C	2	4
D	4	7

Free: 1

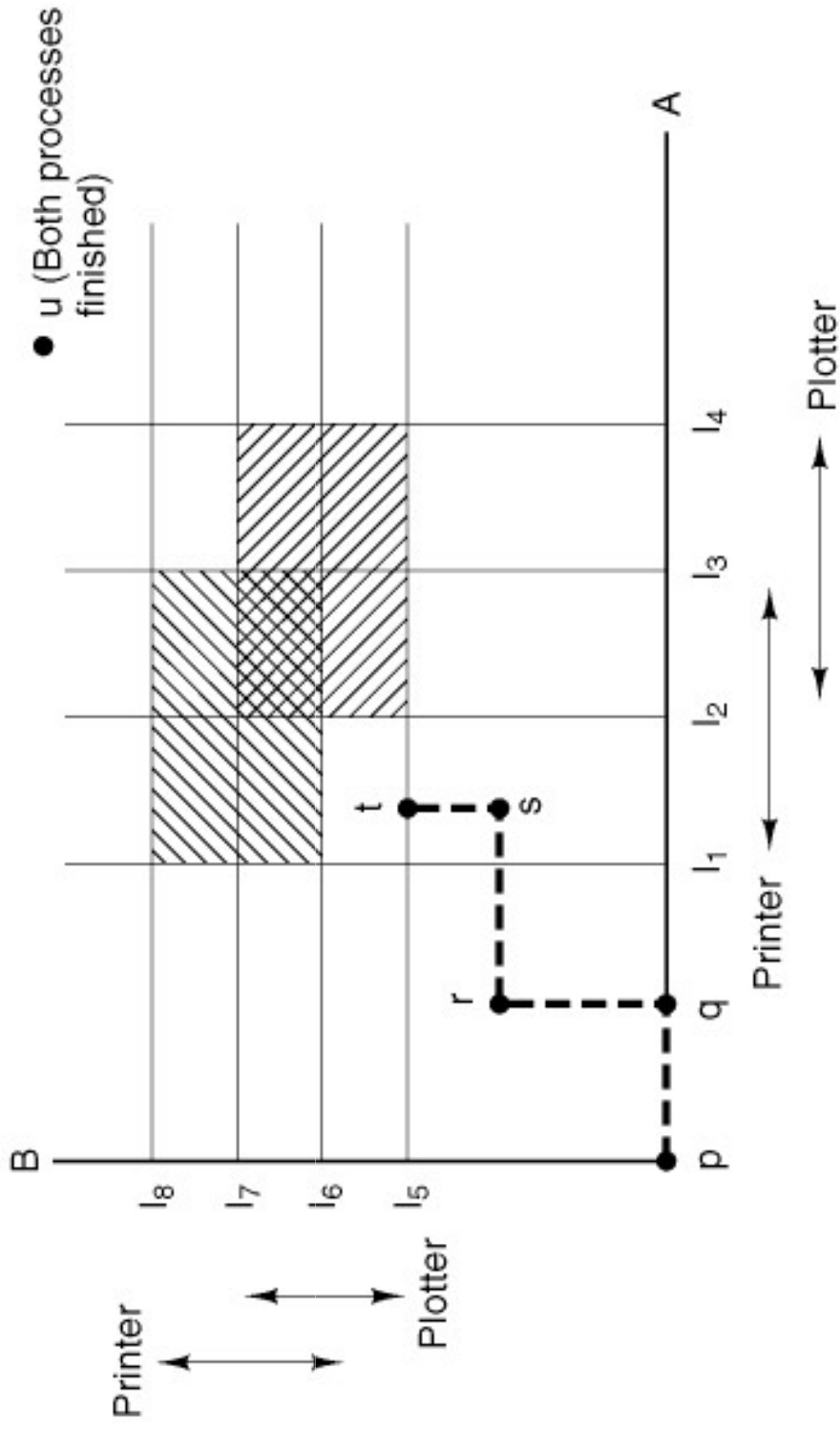
(c)

- Figure 3-13. Three resource allocation states:
(a) Safe. (b) Safe. (c) Unsafe.

Deadlock Avoidance

- Resource Trajectories look at allocation and de-allocation of resources.
- Some areas are impossible (mutual exclusion).
- Some areas inevitably lead to deadlock so avoid these.

Resource Trajectories



Deadlock Avoidance

- Information for bankers algorithm is often not available
- Similarly for Resource trajectories
- Requires operating system to know in detail about process behaviour
- Most useful for analysing a specific system to see if deadlock is possible, and then to avoid it.

Disks

- We concentrate on “real” physical disks, not devices such as RAM disk, or “solid-state” (flash memory) disk.
- A simple one-sided disk consists of a circular, rotating platter, covered in magnetic coating.
- A disk head can magnetise a small section of the surface in one of two directions (N-S poles), storing a 1 or a 0.

Disks

- This disk head can be moved radially out from the centre, at one particular radius it sees all of the bits in one CYLINDER as the platter spins under the head.
- Each cylinder is divided into sectors (say 8 per cylinder), and each sector contains one disk block, around 1kbyte. A whole block is read or written at once.

Multiple heads

- Disks can typically record on both sides of a platter (two heads, one per surface), and often have multiple platters. They might have 8 or more heads, reading 4 or more platters.
- Typically, all the heads are mechanically linked so all read the same cylinder at the same time.
- In multi-head drives, a block might be all on one platter, or may be spread across all platters.

Disk Addresses

- A particular disk block can be addressed by the particular cylinder (0-16383), head (0-15) and sector (0-63) on the disk, but the problem is that these quantities are different for every disk, and choosing appropriate sizes either uses too many bits or not enough bits in the address.

Logical Block Addresses

- More modern controllers typically don't worry about the physical location on the disk, they simply refer to logical block addresses: 0,1,2,.. to a large number.

Multi-zone disks

- Sectors near the centre of a disk are smaller than sectors near the edge.
- Modern disks are multi-zone, with perhaps 20 or more zones, with more sectors per cylinder in each zone, as you move towards the outside of the disk. Physical addresses are no longer visible to the user, only the controller.

Disk capacities

- 1 Gigabyte of computer memory usually means 2^{30} bytes = 1.07×10^9 .
- A 1 Gigabyte disk is usually 10^9 , (so the disk seems bigger).
- Sometimes now see 1 Gibibyte (GiB) to mean 2^{30} bytes.

SCSI Command Set

From Wikipedia – Approximately 60 commands

- **Test unit ready:** Queries device to see if it is ready for data transfers (disk spun up, media loaded, etc.).
- **Inquiry:** Returns basic device information, also used to "ping" the device since it does not modify sense data.
- **Request sense:** Returns any error codes from the previous command that returned an error status.
- **Send diagnostic and Receive diagnostic results:** runs a simple self-test, or a specialised test defined in a diagnostic page.
- **Start/Stop unit:** Spins disks up and down, load/unload media.

SCSI (cont)

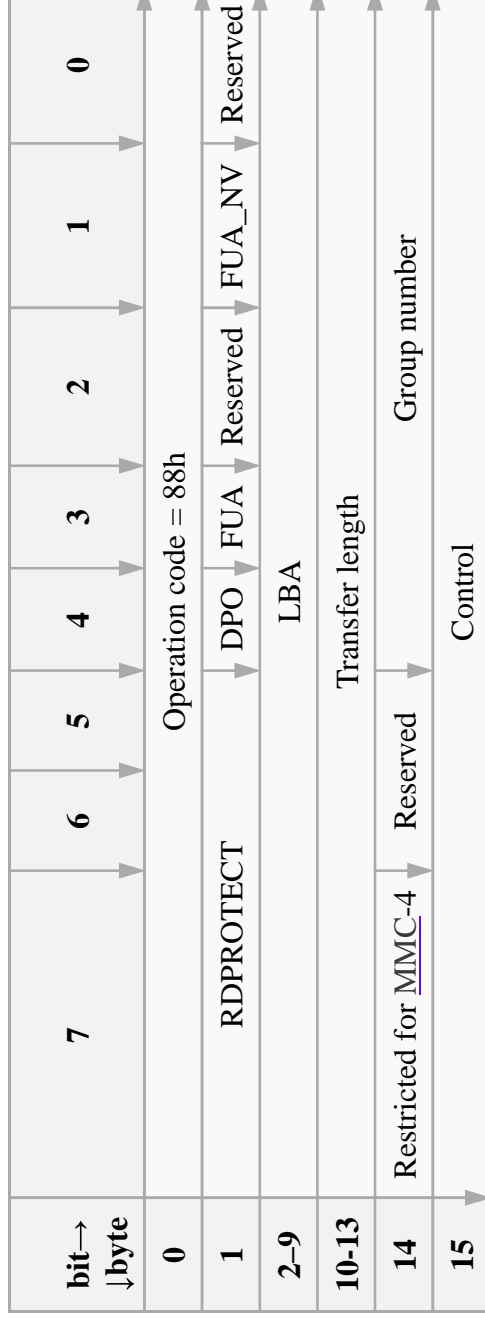
- **Read capacity:** Returns storage capacity.
- **Format unit:** Sets all sectors to all zeroes, also allocates logical blocks avoiding defective sectors.
- **SCSI Read Format capacities:** Retrieve the data capacity of the device.
- **Read (several variants):** Reads data from a device.
- **Write (several variants):** Writes data to a device.
- **Log sense:** Returns current information from log pages.
- **Mode sense:** Returns current device parameters from mode pages.
- **Mode select:** Sets device parameters in a mode page.

SCSI – Read 16

Read (16)

The **Read(16)** command is similar to the [Read\(12\)](#) command except that it has a 64-bit [LBA](#) field.

And is thus capable of addressing 8 388 608 PiB ([ATA-6](#) is capable of max 128 PiB). The [CDB](#) structure is:



IDE Registers

Register	Read Function	Write Function
0	Data	Data
1	Error	Write Precompensation
2	Sector Count	Sector Count
3	Sector Number (0-7)	Sector Number (0-7)
4	Cylinder Low (8-15)	Cylinder Low (8-15)
5	Cylinder High (16-23)	Cylinder High (16-23)
6	Select Drive/Head (24-27)	Select Drive/Head (24-27)
7	Status	Command

(a)

- Figure 3-23. (a) The control registers of an IDE hard disk controller. The numbers in parentheses are the bits of the logical block address selected by each register in LBA mode.

IDE Registers

7	6	5	4	3	2	1	0
1	LBA	1	D	HS3	HS2	HS1	HS0

LBA: 0 = Cylinder/Head/Sector Mode

1 = Logical Block Addressing Mode

D: 0 = master drive

1 = slave drive

HSn: CHS mode: Head select in CHS mode

LBA mode: Block select bits 24 - 27

(b)

- Figure 3-23. (b) The fields of the Select Drive/Head register.

Disk Read/Write Time

- Seek Time – time for the disk arm with heads to move to the correct cylinder.
- The rotational delay (given by disk speed in RPM) – the time for the sector to rotate under the heads to start of block.
- Transfer time – to read the block off disk, transfer to a buffer, and signal CPU.

Transfer time

- Seek time - maximum and average
- Rotational delay - maximum of 1 revolution, average 0.5 revolutions.
- Rotational delay for one sector, and transfer time over controller interface (might be overlapped, but usually check first for errors)

RAID

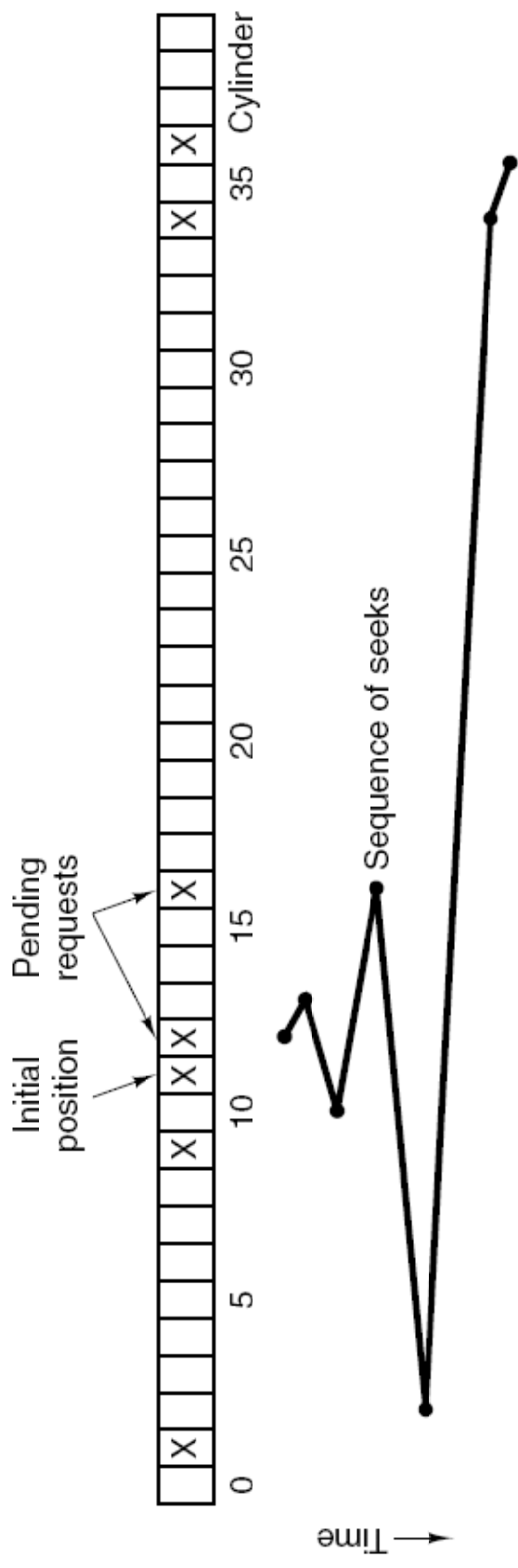
- Redundant Array of Independent Disks.
- Different disk blocks can be interleaved on different disks – improves speed. This is striping, RAID level 0.
- RAID 1 – data is duplicated, on multiple disks providing reliability.
- A single block can be spread across multiple drives, eg. 8 drives contain one bit of each byte, another 3 drives provide an ECC, so any one drive can fail, and the system still work. 11 drives gives 8x speedup and much higher reliability.

Disk Scheduling

- If there are many pending read/write requests to a disk, the order in which they are serviced can greatly improve speed.
- Try to read “next closest” pending read, either in same cylinder, or with a small seek.

Disk Arm Scheduling

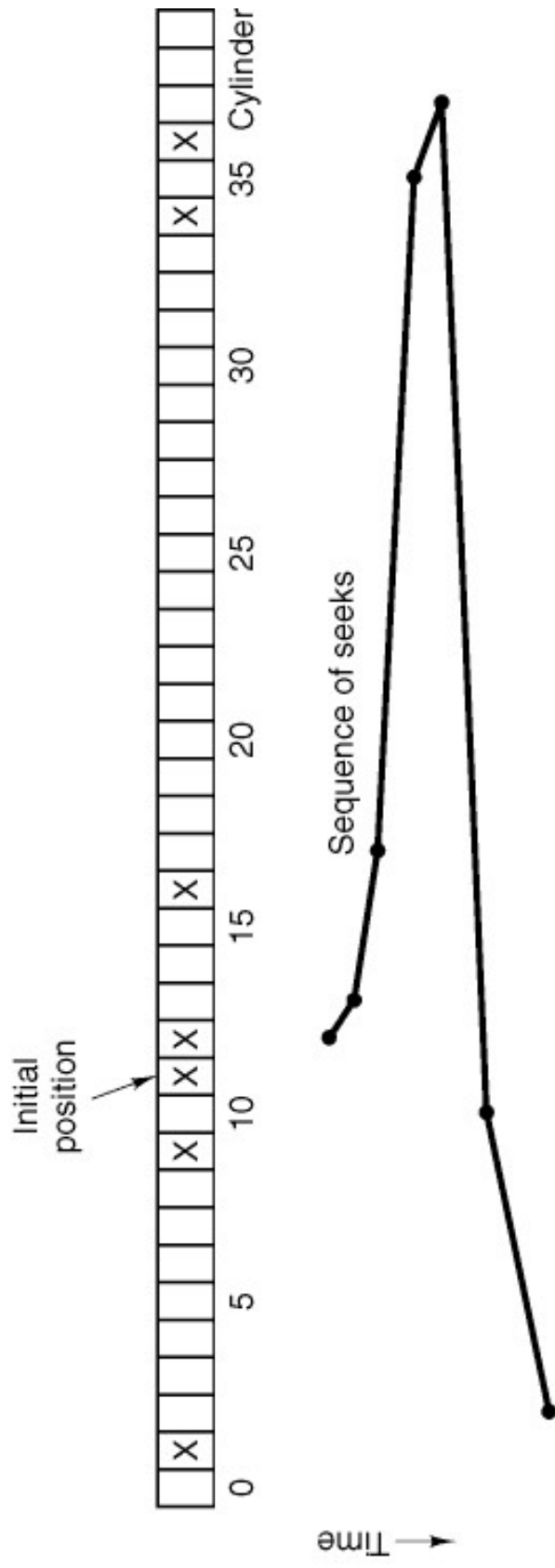
Algorithms (1)



- Figure 3-21. Shortest Seek First (SSF) disk scheduling algorithm.

Disk Arm Scheduling

Algorithms (2)



- Figure 3-22. The elevator algorithm for scheduling disk requests.

Common Hard Drive Errors

1. Programming error
 - request for nonexistent sector
2. Transient checksum error
 - caused by dust on the head
3. Permanent checksum error
 - disk block physically damaged
4. Seek error
 - the arm sent to cylinder 6 but it went to 7
5. Controller error
 - controller refuses to accept commands

Bad Blocks

- Usually some parts of the disk don't work (millions of blocks, expect a few to be faulty).
- Either the file system, or the disk controller keeps a list of bad blocks.
- Controller may keep a few "spare" tracks to substitute for bad blocks, invisible to operating system.

Disk caches

- Memory is cheap, so disks can easily read a whole cylinder into a cache, since subsequent reads are often from the same cylinder.
- Also commonly keep a cache of recently read blocks.
- Operating system (disk device driver) may also keep a cache in main memory.

Structure of files on disk

- Dealt with when we explore file systems – many different formats.

Tutorial Question 1.

- Maxtor 7Y250M0
 - 9ms average seek time
 - 7200rpm
 - 8MB buffer
 - 150MB/s transfer rate
- How long does it take to
 - move 1 byte?
 - 4Kbytes?
 - enough to double minimum transaction time (average seek+rotation)?

Tutorial Question 2.

- MATSUSHITA DVD-R UJ-825
 - 8x DVD (10.57MB/s)
 - 24x CD (3.52MB/s)
 - 2MB buffer
 - 45ms average seek*
 - Rotation 1400rpm*
- How long does it take to
 - move 1 byte?
 - 4Kbytes?
 - enough to double minimum transaction time (average seek+rotation)?

Summary

- **Deadlock**
 - Definition
 - How it is caused
 - How to deal with it
- **Disks**
 - A key I/O device
 - What is looks like to the CPU
 - What affects its performance