COMP4611: Design and Analysis of Computer Architectures

Input/Output

I/O Devices, RAID and Network Storage

Lin Gu
CSE, HKUST

1

Motivation: Who Cares About I/O?

- CPU Performance: 60% improvement per year
- I/O system performance: < 10% improvement per year
  - sometimes limited by *mechanical* delays (disk I/O)
- 10% IO & 10x CPU => 5x Performance (lose 50%)

  10% IO & 100x CPU => 10x Performance (lose 90%)
- I/O bottleneck:

  Diminishing value of faster CPUs

2

Input and Output Devices

I/O devices are diverse with respect to
- Behavior – input, output or storage
- User – human or machine
- Data rate – the rate at which data are transferred between the I/O device and the main memory or processor

Device	Behavior	User	Data rate (Mb/s)
Keyboard	input	human	0.0001
Mouse	input	human	0.0038
Laser printer	output	human	3.2000
Graphics display	output	human	800.0000-8000.0000
Network/LAN	input or output	machine	10.0000-10000.0000
Magnetic disk	storage	machine	240.0000-2560.0000

9 orders of magnitude range

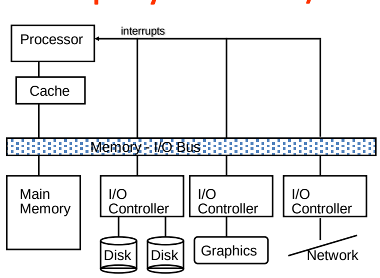
3

I/O Performance Measures

- I/O bandwidth (throughput) – amount of information that can be input (output) and communicated across an interconnect (e.g., a bus) to the processor/memory (I/O device) per unit time
  1. How much data can we move through the system in a certain time?
  2. How many I/O operations can we do per unit time?
- I/O response time (latency) – the total elapsed time to accomplish an input or output operation
  - An especially important performance metric in real-time systems
- Many applications require both high throughput and short response times

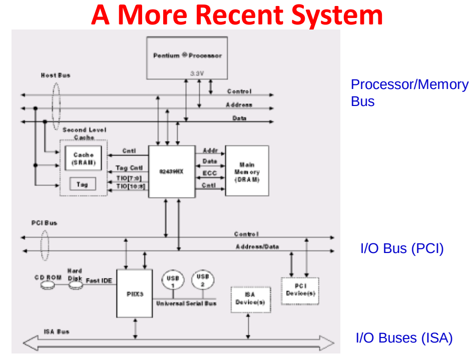
4

A Simpe System with I/O



5

A More Recent System



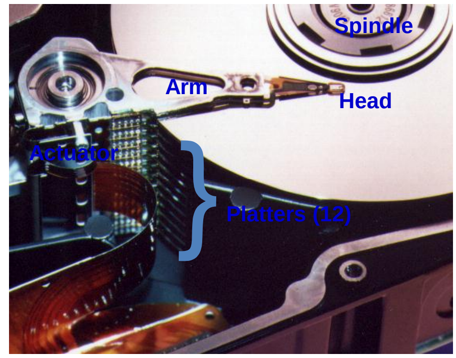
6

Introduction to I/O

- Most of the I/O devices are ***very slow*** compared to the cycle time of a CPU.
- The architecture of I/O systems is an active area of R&D.
  - I/O systems can define the performance of a system.
- *Computer architects* strive to design systems that do not tie up the CPU waiting for slow I/O systems (too many applications running simultaneously on processor).

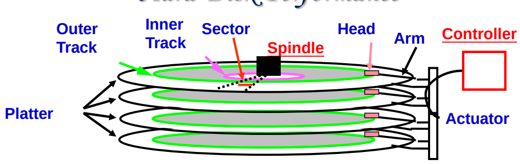
7

Hard Disk



8

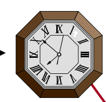
Hard Disk Performance

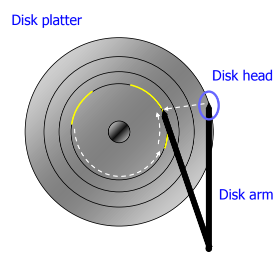


- Disk Latency = Seek Time + Rotation Time + Transfer Time + Controller Overhead
- Seek Time depends on the moving distance and moving speed of arms
- Rotation Time depends on how fast the disk rotates and how far sector is from head
- Transfer Time depends on data rate (bandwidth) of disk and size of request

9

Disk Access Time

I want block X →  → block x in memory



Disk access time =

Seek time

\+ Rotational delay

\+ Transfer time

\+ Other delays

10

Example: Barracuda 180

- 181.6 GB, 3.5 inch disk
- 12 platters, 24 surfaces
- 24,247 cylinders
- 7,200 RPM; (4.2 ms avg. latency)
- 7.4/8.2 ms avg. seek (r/w)
- 65 to 35 MB/s (internal)
- 0.1 ms controller time

Latency =

per access { Queuing Time + Controller time + Seek Time + Rotation Time +

per byte Size / Bandwidth

source: www.seagate.com

11

Disk Performance Example

Calculate time to read 64 KB (128 sectors) for Barracuda 180 X using advertised performance; sector is on outer track

Disk latency = average seek time + average rotational delay + transfer time + controller overhead

= 7.4 ms + 0.5 * 1/(7200 RPM) + 64 KB / (65 MB/s) + 0.1 ms

= 7.4 ms + 0.5 /(7200 RPM/(60000ms/M)) + 64 KB / (65 KB/ms) + 0.1 ms

= 7.4 + 4.2 + 1.0 + 0.1 ms = 12.7 ms

12

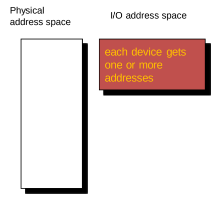
Communication of I/O Devices and Processor

How does the processor command the I/O devices?

- Special I/O instructions
  - Must specify both the device (port number) and the command
- For example:

inp reg, port; register←port

out port, reg; port→register

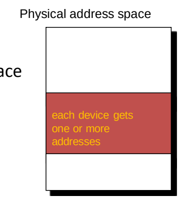


13

Communication of I/O Devices and Processor

How does the processor command the I/O devices?

- Memory-mapped I/O
  - Portions of the memory address space are assigned to I/O devices
  - Read and writes to those memory addresses are interpreted as commands to the I/O devices
  - Load/stores to the I/O address space can only be done by the OS



14

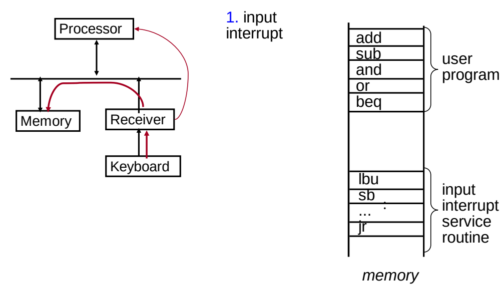
Communication of I/O Devices and Processor

How does the I/O device communicate with the processor?

- Polling – the processor periodically checks the status of an I/O device to determine its need for service
  - Processor is totally in control – it also does all the work
  - Can waste a lot of processor time due to speed differences
- Interrupt-driven I/O – the I/O device issues an interrupt to the processor to indicate that it needs attention

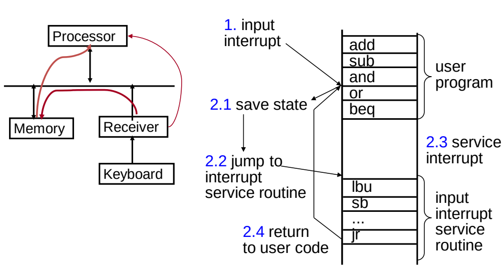
15

Interrupt-Driven Input



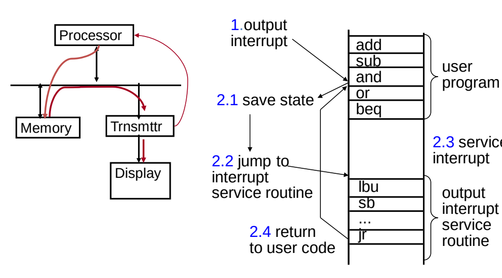
16

Interrupt-Driven Input



17

Interrupt-Driven Output



18

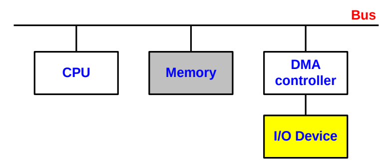
Direct-Memory Access (DMA)

- Interrupt-driven IO relieves the CPU from waiting for every IO event
- But the CPU can still be bogged down if it is used in transferring IO data.
  - Typically blocks of bytes.
- For high-bandwidth devices (like disks) interrupt-driven I/O would consume a *lot* of processor cycles

19

DMA

- DMA – the I/O controller has the ability to transfer data directly to/from the memory without involving the processor



20

DMA

- Consider printing a 60-line by 80-character page
- With no DMA:
  - CPU will be interrupted 4800 times, once for each character printed.
- With DMA:
  - OS sets up an I/O buffer and CPU writes the characters into the buffer.
  - DMA is commanded (includes the beginning address of the buffer and its size) to print the buffer.
  - DMA will take items from the buffer one-at-a-time and performs everything requested.
  - Once the operation is complete, the DMA sends a single interrupt signal to the CPU.

21

I/O Communication Protocols

- Typically *one* I/O channel controls *multiple* I/O devices.
- We need a two-way communication between the channel and the I/O devices.
  - The channel needs to send the command/data to the I/O devices.
  - The I/O devices need to send the data/status information to the channel whenever they are ready.

22

Channel to I/O Device Communication

- Channel sends the address of the device on the bus.
- All devices compare their addresses against this address.
  - Optionally, the device which has matched its address places its own address on the bus again.
    - First, it is an acknowledgement signal to the channel;
    - Second, it is a check of validity of the address.
- The channel then places the I/O command/data on the bus received by the correct I/O device.
- The command/data is queued at the I/O device and is processed whenever the device is ready.

23

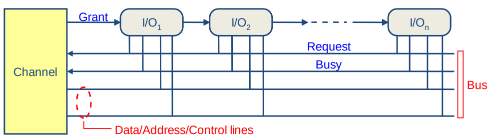
I/O Devices to Channel Communication

- The I/O devices-to-channel communication is more complicated, since now several devices may require *simultaneous* access to the channel.
  - Need arbitration among multiple devices (bus master?)
  - May prefer a priority scheme
- Three methods for providing I/O devices-to-channel communication

24

Daisy Chaining

- Two schemes
- Centralized control (priority scheme)



25

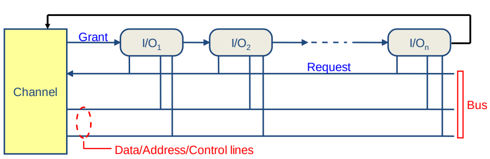
Daisy Chaining

- The I/O devices activate the request line for bus access.
- If the bus is not busy (indicated by no signal on busy line), the channel sends a Grant signal to the first I/O device (closest to the channel).
  - If the device is not the one that requested the access, it propagates the Grant signal to the next device.
  - If the device is the one that requested an access, it then sends a busy signal on the busy line and begins access to the bus.
- Only a device that holds the Grant signal can access the bus.
- When the device is finished, it resets the busy line.
- The channel honors the requests only if the bus is not busy.
- Obviously, devices closest to the channel have a higher priority and block access requests by lower priority devices.

26

Daisy Chaining

- Decentralized control (Round-robin Scheme)



27

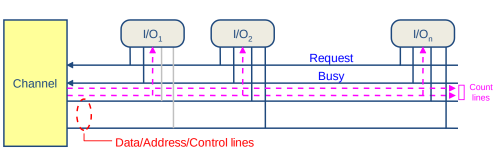
Daisy Chaining

- The I/O devices send their request.
- The channel activates the Grant line.
- The first I/O device which requested access accepts the Grant signal and has control over the bus.
  - Only the devices that have received the grant signal can have access to the bus.
- When a device is finished with an access, it checks to see if the request line is activated or not.
- If it is activated, the current device sends the Grant signal to the next I/O device (Round-Robin) and the process continues.
  - Otherwise, the Grant signal is deactivated.

28

Polling

- The channel interrogates (polls) the devices to find out which one requested access:



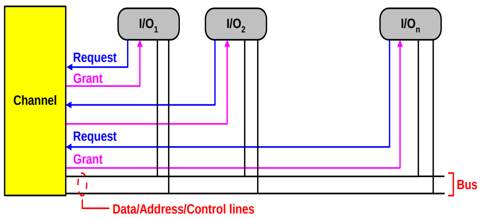
29

Polling

- Any device requesting access places a signal on request line.
- If the busy signal is off, the channel begins polling the devices to see which one is requesting access.
  - It does this by sequentially sending a count from 1 to n on $\log_2 n$ lines to the devices.
- Whenever a requesting device matches the count against its own number (address), it activates the busy line.
- The channel stops the count (polling) and the device has access over the bus.
- When access is over, the busy line is deactivated and the channel can either continue the count from the last device (Round-Robin) or start from the beginning (priority).

30

Independent Requests



31

Independent Requests

- Each device has its own Request-Grant lines:
  - Again, a device sends in its request, the channel responds by granting access
  - Only the device that holds the grant signal can access the bus
  - When a device finishes access, it lowers it request signal.
  - The channel can use either a Priority scheme or Round-Robin scheme to grant the access.

32

I/O Buses

- Connect I/O devices (channels) to memory.
  - Many types of devices are connected to a bus.
  - Have a wide range of bandwidth requirements for the devices connected to a bus.
  - Typically follow a bus standard, e.g., PCI, SCSI.
- Clocking schemes:
  - Synchronous: The bus includes a clock signal in the control lines and a fixed protocol for address and data relative to the clock
  - Asynchronous: The bus is self-timed and uses a handshaking protocol between the sender and receiver

33

I/O Buses

Synchronous buses are fast and inexpensive, but

- All devices on the bus must run at the same clock rate.
- Due to clock-skew problems, buses cannot be long.
- *CPU-Memory* buses are typically implemented as synchronous buses.
  - The front side bus (FSB) clock rate typically determines the clock speed of the memory you must install.

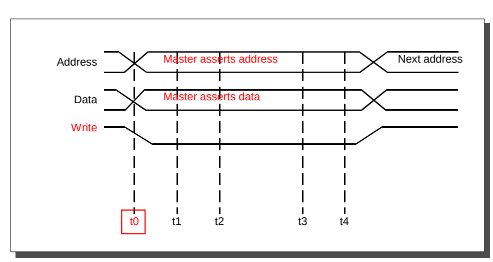
34

I/O Buses

- Asynchronous buses are self-timed and use a handshaking protocol between the sender and receiver.
- This allows the bus to accommodate a wide variety of devices and to lengthen the bus.
- I/O buses are typically asynchronous.
  - A master (e.g., an I/O channel writing into memory) asserts address, data, and control and begins the handshaking process.

35

I/O Buses

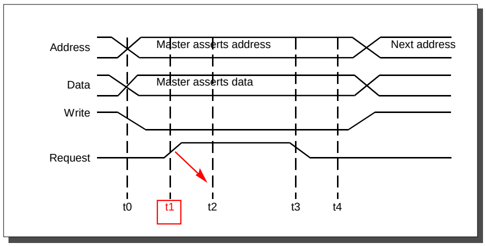


Asynchronous write: master asserts address, data, write buses.

36

I/O Buses

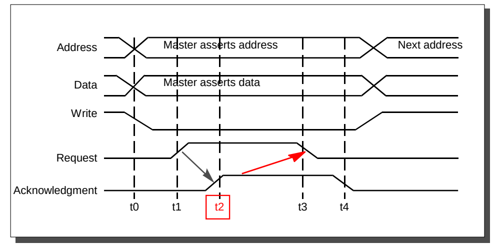
Asynchronous write: master asserts request, expecting acknowledgement later.



37

I/O Buses

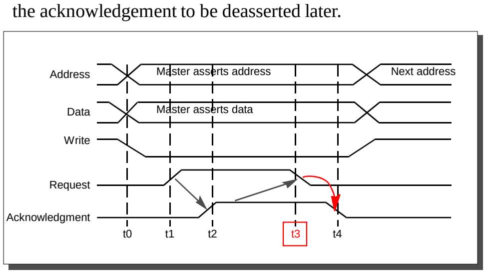
Asynchronous write: slave (memory) asserts acknowledgment, expecting request to be deasserted later.



38

I/O Buses

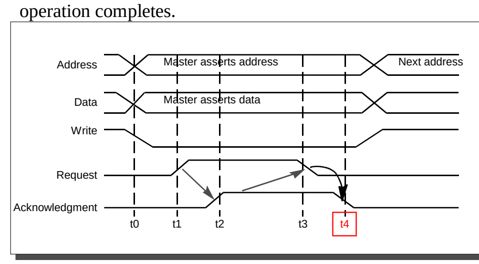
Asynchronous write: master deasserts request and expects the acknowledgement to be deasserted later.



39

I/O Buses

Asynchronous write: slave deasserts acknowledgement and operation completes.



40

I/O Bus Examples

- Multiple master I/O buses:

	Sun-S bus	IBM MicroChannel	PCI	SCSI 2
Data width	32 bits	32 bits	32 to 64 bits	8 to 16 bits
Clock rate	16 to 25 MHZ	Asynchronous	33 MHZ	10 MHZ or Asynch.
32-bit reads bandwidth	33 MB/Sec	20 MB/Sec	33 MB/Sec	20 MB/sec or 6 MB/Sec
Peak Bandwidth	89 MB/Sec	75 MB/Sec	132 MB/Sec	20 MB/sec or 6 MB/Sec

41

I/O Bus Examples

- Multiple master CPU-memory buses:

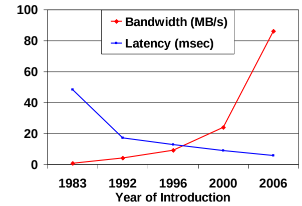
	HP Summit	SGI Challenge	Sun XDBus
Data width	128 bits	256 bits	144 bits
Clock rate	60 MHZ	48 MHZ	66 MHZ
Peak Bandwidth	960 MB/Sec	1200 MB/Sec	1056 MB/Sec

42

RAID (Redundant Array of Inexpensive Disks)

Disk Latency & Bandwidth Improvements

- Disk latency is one average seek time plus the rotational latency
- Disk bandwidth is the peak transfer rate of formatted data
- In the time that the disk bandwidth doubles the latency improves by a factor of only 1.2 to 1.4



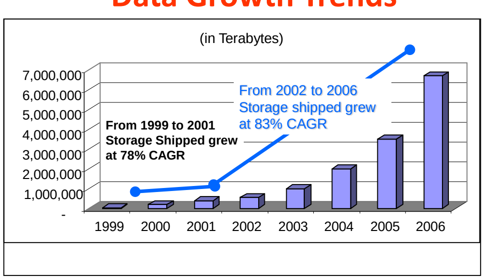
Media Bandwidth/Latency Demands

- Bandwidth requirements
  - High quality video
    - Digital data = (30 frames/s) × (640 x 480 pixels) × (24-b color/pixel) = 221 Mb/s (27.625 MB/s)
  - High quality audio
    - Digital data = (44,100 audio samples/s) × (16-b audio samples) × (2 audio channels for stereo) = 1.4 Mb/s (0.175 MB/s)
- Latency issues
  - How sensitive is your eye (ear) to variations in video (audio) rates?
  - How can you ensure a constant rate of delivery?
  - How important is synchronizing the audio and video streams?
    - 15 to 20 ms early to 30 to 40 ms late is tolerable

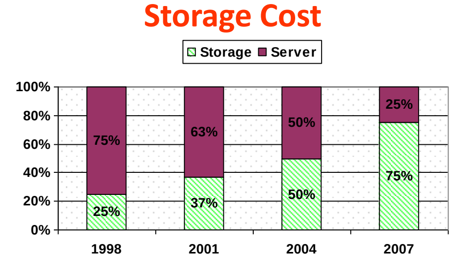
Storage Pressures

- Storage capacity growth estimates: 60-100% per year
  - Growth of e-business, e-commerce, and e-mail ⇨ now common for organizations to manage hundreds of TB of data
  - Mission critical data must be continuously available
  - Regulations require long-term archiving
  - More storage-intensive applications on market
- Storage and Security are leading **pain points** for the IT community
- Managing storage growth effectively is a challenge

Data Growth Trends



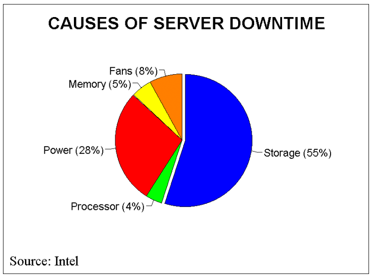
Storage Cost



Storage cost as proportion of total IT spending as compared to server cost

Availability/Reliability and Performance are EXTREMLY important

Importance of Storage Reliability



RAID

- To increase the availability and the performance (bandwidth) of a storage system, instead of a single disk, a set of disks (disk arrays) can be used.
- Similar to memory interleaving, data can be spread among multiple disks (*striping*), allowing simultaneous access to the data and thus improving the throughput.
- However, the reliability of the system drops (n devices have $1/n$ the reliability of a single device).

Dependability Measures

- Reliability: mean time to failure (MTTF)
- Service interruption: mean time to repair (MTTR)
- Mean time between failures
  - MTBF = MTTF + MTTR
- Availability = MTTF / (MTTF + MTTR)
- Improving Availability
  - Increase MTTF: fault avoidance, fault tolerance, fault forecasting
  - Reduce MTTR: improved tools and processes for diagnosis and repair

Array Reliability

- Reliability of N disks = Reliability of 1 Disk ÷ N

50,000 Hours ÷ 70 disks = 700 hours

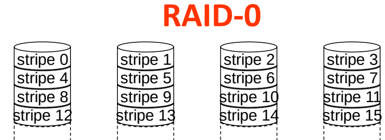
Disk system Mean Time To Failure (MTTF): Drops from 6 years to 1 month!

•Arrays without redundancy too unreliable to be useful!

RAID

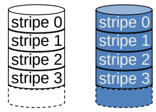
- A disk array's availability can be improved by adding redundant disks:
  - If a single disk in the array fails, the lost information can be reconstructed from redundant information.
- These systems have become known as RAID - Redundant Array of Inexpensive Disks.
  - Depending on the number of redundant disks and the redundancy scheme used, RAIDs are classified into levels.
  - 6 levels of RAID (0-5) are accepted by the industry.
  - Level 2 and 4 are not commercially available, they are included for clarity

RAID-0



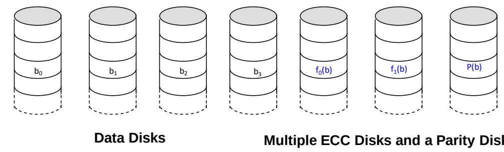
- Striped, non-redundant
  - Parallel access to multiple disks
  - ➔ Excellent data transfer rate
  - ➔ Excellent I/O request processing rate (for large stripes) if the controller supports independent Reads/Writes
  - ➔ Not fault tolerant (**AID**)
- Typically used for applications requiring high performance for non-critical data (e.g., video streaming and editing)

RAID-1 - Mirroring



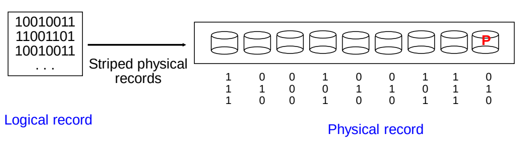
- Called mirroring or shadowing, uses an extra disk for each disk in the array (most costly form of redundancy)
- Whenever data is written to one disk, that data is also written to a redundant disk: good for reads, fair for writes
- If a disk fails, the system just goes to the mirror and gets the desired data.
- Fast, but very expensive.
- Typically used in system drives and critical files
  - Banking, insurance data
  - Web (e-commerce) servers

RAID-2: Memory-Style ECC



- Multiple disks record the (error correcting code) ECC information to determine which disk is in fault
- A parity disk is then used to reconstruct corrupted or lost data
  Needs $\log_2$(number of disks) redundancy disks
- Least used since ECC is irrelevant because most new Hard drives support built-in error correction

RAID-3 - Bit-interleaved Parity



- Use 1 extra disk for each array of *n* disks.
- Reads or writes go to all disks in the array, with the extra disk to hold the parity information in case there is a failure.
- The parity is carried out at bit level:
  - A parity bit is kept for each bit position across the disk array and stored in the redundant disk.
  - Parity: sum modulo 2.
    - parity of 1010 is 0
    - parity of 1110 is 1

Or use XOR of bits

RAID-3 - Bit-interleaved Parity

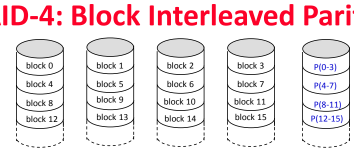
- If one of the disks fails, the data for the failed disk must be recovered from the parity information:
  - This is achieved by subtracting the parity of good data from the original parity information:
  - Recovering from failures takes longer than in mirroring, but failures are rare, so is okay
  - Examples:

Original data	Original Parity	Failed Bit	Recovered data
1010	0	101X	|0-0| = 0
1010	0	10X0	|0-1| = 1
1110	1	111X	|1-1| = 0
1110	1	11X0	|1-0| = 1

RAID-4 - Block-interleaved Parity

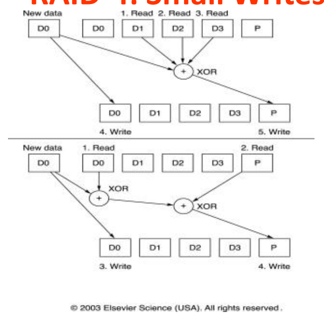
- In RAID 3, every read or write needs to go to all disks since bits are interleaved among the disks.
- Performance of RAID 3:
  - Only one request can be serviced at a time
  - Poor I/O request rate
  - Excellent data transfer rate
  - Typically used in large I/O request size applications, such as imaging or CAD
- RAID 4: If we distribute the information block-interleaved, where a disk sector is a block, then for normal reads different reads can access different segments in parallel. Only if a disk fails we will need to access all the disks to recover the data.

RAID-4: Block Interleaved Parity



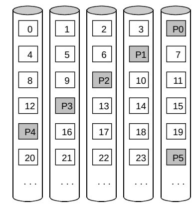
- Allow for parallel access by multiple I/O requests
- Doing multiple small reads is now faster than before.
- A write, however, is a different story since we need to update the parity information for the block.
- Large writes (full stripe), update the parity:
  $P' = d0' \oplus d1' \oplus d2' \oplus d3'$;
- Small writes (eg. write on d0), update the parity:
  $P = d0 \oplus d1 \oplus d2 \oplus d3$
  $P' = d0' \oplus d1 \oplus d2 \oplus d3 = d0' \oplus d0 \oplus P$;
- However, writes are still very slow since parity disk is the bottleneck.

RAID-4: Small Writes



RAID-5 - Block-interleaved Distributed Parity

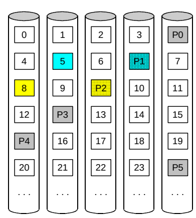
- To address the write deficiency of RAID 4, RAID 5 distributes the parity blocks among all the disks.



RAID 5

RAID-5 - Block-interleaved Distributed Parity

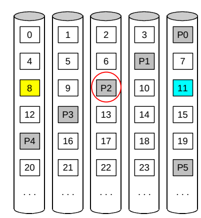
- This allows *some* writes to proceed in parallel
  - For example, writes to blocks 8 and 5 can occur simultaneously.



RAID 5

RAID-5 - Block-interleaved Distributed Parity

- However, writes to blocks 8 and 11 cannot proceed in parallel.



RAID 5

Performance of RAID-5 - Block-interleaved Distributed Parity

- **Performance of RAID-5**
  - I/O request rate: excellent for reads, good for writes
  - Data transfer rate: good for reads, good for writes
  - Typically used for high request rate, read-intensive data lookup
  - **File and Application servers, Database servers, WWW, E-mail, and News servers, Intranet servers**
- The most versatile and widely used RAID.

RAID-6 – Row-Diagonal Parity

- To handle 2 disk errors
  - In practice, another disk error can occur before the first problem disk is repaired
- Use p-1 data disks, 1 row-parity disk, 1 diagonal-parity disk
- If any two of the p+1 disks fail, data can still be recovered

Data Disk 0	Data Disk 1	Dada Disk 2	Dada Disk 3	Row Parity Disk	Diagonal Parity Disk
0	1	2	3	4	0
1	2	3	4	0	1
2	3	4	0	1	2
3	4	0	1	2	3

Dependability

Definitions

- Examples on why precise definitions so important for reliability
- **Is a programming mistake a fault, error, or failure?**
  - **Are we talking about the time it was designed or the time the program is run?**
  - **If the running program doesn't exercise the mistake, is it still a fault/error/failure?**
- **If an alpha particle hits a DRAM memory cell, is it a fault/error/failure if it doesn't change the value?**
  - **Is it a fault/error/failure if the memory doesn't access the changed bit?**
  - **Did a fault/error/failure still occur if the memory had error correction and delivered the corrected value to the CPU?**

IFIP Standard terminology

- **Computer system *dependability*: quality of delivered service such that reliance can be justifiably placed on the service**
- ***Service* is observed *actual behavior* as perceived by other system(s) interacting with this system's users**
- **Each module has ideal *specified behavior*, where *service specification* is agreed description of expected behavior**
- **A system *failure* occurs when the actual behavior deviates from the specified behavior**
- **failure occurred because an *error*, a defect in module**
- **The cause of an error is a *fault***
- **When a fault occurs it creates a *latent error*, which becomes *effective* when it is activated**
- **When error actually affects the delivered service, a failure occurs (time from error to failure is *error latency*)**

Fault v. (Latent) Error v. Failure

- **An error is manifestation *in the system* of a fault, a failure is manifestation *on the service* of an error**
- **If an alpha particle hits a DRAM memory cell, is it a fault/error/failure if it doesn't change the value?**
  - **Is it a fault/error/failure if the memory doesn't access the changed bit?**
  - **Did a fault/error/failure still occur if the memory had error correction and delivered the corrected value to the CPU?**
- **An alpha particle hitting a DRAM can be a fault**
- **If it changes the memory, it creates an error**
- **Error remains latent until affected memory word is read**
- **If the effected word error affects the delivered service, a failure occurs**

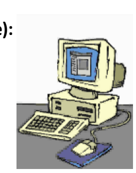
Fault Categories

1. Hardware faults: Devices that fail, such alpha particle hitting a memory cell
2. Design faults: Faults in software (usually) and hardware design (occasionally)
3. Operation faults: Mistakes by operations and maintenance personnel
4. Environmental faults: Fire, flood, earthquake, power failure, and sabotage

- Also by duration:

1. Transient faults exist for limited time and not recurring
2. Intermittent faults cause a system to oscillate between faulty and fault-free operation
3. Permanent faults do not correct themselves over time

Fault Tolerance vs Disaster Tolerance

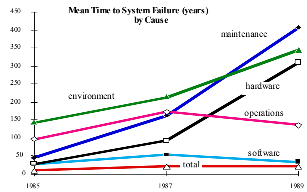


- **Fault-Tolerance (or more properly, Error-Tolerance):** mask local faults (prevent errors from becoming failures)
  - RAID disks
  - Uninterruptible Power Supplies
  - Cluster Failover
- **Disaster Tolerance:** masks site errors (prevent site errors from causing service failures)
  - Protects against fire, flood, sabotage,..
  - Redundant system and service at remote site.
  - Use design diversity

From Jim Gray's "Talk at UC Berkeley on Fault Tolerance " 11/9/00

Case Studies - Tandem Trends

Reported MTTF by Component



		1985	1987	1990	
SOFTWARE	2	53	33	Years	
HARDWARE	29	91	310	Years	
MAINTENANCE	45	162	409	Years	
OPERATIONS	99	171	136	Years	
ENVIRONMENT		142	214	346	Years
SYSTEM		**8**	**20**	**21**	**Years**

Problem: Systematic Under-reporting

From Jim Gray's "Talk at UC Berkeley on Fault Tolerance " 11/9/00

HW Failures in Real Systems: Tertiary Disks

A cluster of 20 PCs in seven 7-foot high, 19-inch wide racks with 368 8.4 GB, 7200 RPM, 3.5-inch IBM disks. The PCs are P6-200MHz with 96 MB of DRAM each. They run FreeBSD 3.0 and the hosts are connected via switched 100 Mbit/second Ethernet. Data collected during 18 months of operation.

Component	Total in System	Total Failed	% Failed
SCSI Controller	44	1	2.3%
SCSI Cable	39	1	2.6%
SCSI Disk	368	7	1.9%
IDE Disk	24	6	25.0%
Disk Enclosure -Backplane	46	13	28.3%
Disk Enclosure - Power Supply	92	3	3.3%
Ethernet Controller	20	1	5.0%
Ethernet Switch	2	1	50.0%
Ethernet Cable	42	1	2.3%
CPU/Motherboard	20	0	0%

How Realistic is "5 Nines"?

- **HP claims HP-9000 server HW and HP-UX OS can deliver 99.999% availability guarantee "in certain pre-defined, pre-tested customer environments"**
  - **Application faults?**
  - **Operator faults?**
  - **Environmental faults?**
- **Collocation sites (lots of computers in 1 building on Internet) have**
  - **1 network outage per year (~1 day)**
  - **1 power failure per year (~1 day)**
- **Microsoft Network unavailable recently for a day due to problem in Domain Name Server: if only outage per year, 99.7% or 2 Nines**

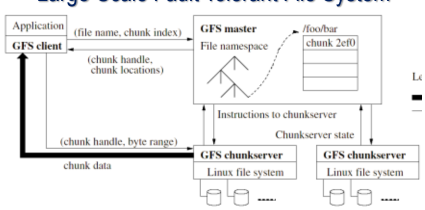
Data Processing for Today's Web-Scale Services

Application tasks	Data size	Computation	Network
Web crawl	800TB	Highly parallel	High bandwidth
Data analytics	200TB	Intensive	High bandwidth, low latency
Orkut (social network)	9TB	Parallelizable	Low latency
Youtube	Estimated multi-petabytes	Intensive, parallelizable	Very high bandwidth, low latency
e-business (e.g., Amazon)	Estimated multi-petabytes	Intensive	High bandwidth, very low latency

- Petabytes of data and demanding computation
- Network performance is essential!

Chang, F et al. Bigtable: a distributed storage system for structured data. In Proceedings of the 7th Symposium on Operating Systems Design and Implementation (Seattle, Washington, November 06 - 08, 2006). 205-218. http://baijia.info/showthread.php?tid=4

Large-Scale Fault Tolerant File System



- **A distributed file system at work (GFS)**
  - **Single master and numerous slaves communicate with each other**
  - **File data unit, "chunk", is up to 64MB. Chunks are replicated.**
- **Requires extremely high network bandwidth, very low network latency**

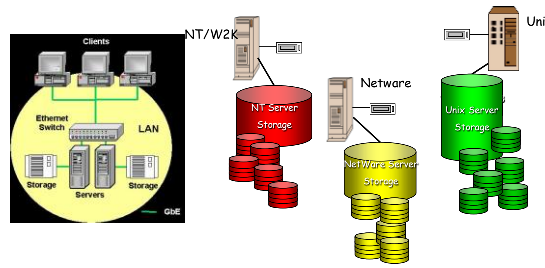
Network Storage Systems

Which Storage Architecture?

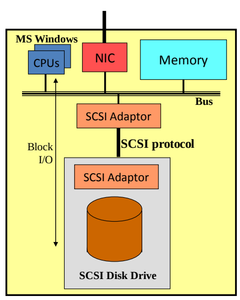
- DAS - Directly-Attached Storage
- NAS - Network Attached Storage
- SAN - Storage Area Network

Storage Architectures

(Direct Attached Storage (DAS))

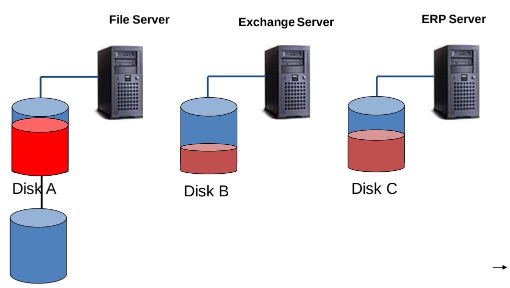


DAS



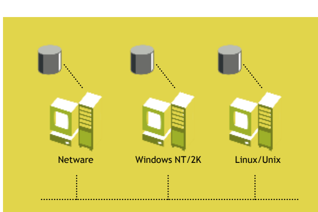
Traditional Server

Storage Architectures
(Direct Attached Storage (DAS))



The Problem with DAS

•Direct Attached Storage (DAS)



- Data is bound to the server hosting the disk
- Expanding the storage may mean purchasing and managing another server
- In heterogeneous environments, management is complicated

Storage Architectures

(Direct Attached Storage (DAS))

✓ Advantages

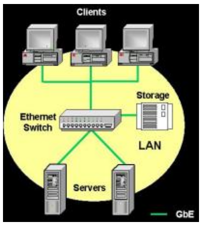
- Low cost
- Simple to use
- Easy to install

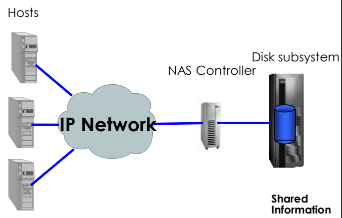
✓ Disadvantages

- No shared resources
- Difficult to backup
- Limited distance
- Limited high-availability options
- Complex maintenance

Solution for small organizations only

Storage Architectures
(Network Attached Storage (NAS))



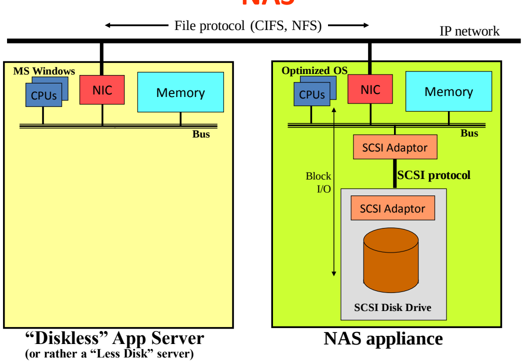


NAS (Network Attached Storage)

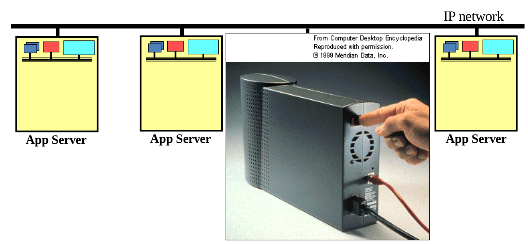
What is it?

NAS devices contain embedded processors that run specialized OS or micro kernel that understands networking protocols and is optimized for particular tasks, such as file service. NAS devices usually deploy some level of RAID storage.

NAS



The NAS Network



NAS - truly an appliance

More on NAS

- NAS Devices can easily and quickly attach to a LAN
- NAS is platform and OS independent and appears to machines as another server
- NAS Devices provide storage that can be addressed via standard file system (e.g., NFS, CIFS) protocols

Storage Architectures
(Network Attached Storage (NAS))

✓ Advantages

- Easy to install
- Easy to maintain
- Shared information
- Unix, Windows file sharing
- Remote access

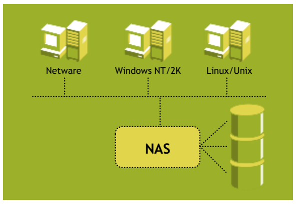
✓ Disadvantages

- Not suitable for databases
- Storage islands
- Not-very-scalable solution
- NAS controller is a bottle neck
- Vendor-dependable

Suitable for file based application

Some NAS Problems

•Network Attached Storage (NAS)



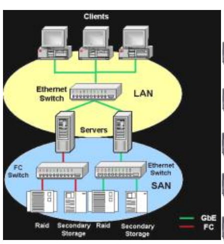
- Each appliance represents a larger island of storage
- Data is bound to the NAS device hosting the disk and cannot be accessed if the system hosting the drive fails
- Storage is labor-intensive and thus expensive
- Network is bottleneck

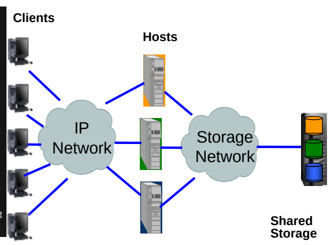
Some Benefits of NAS

- Files are easily shared among users at high demand and performance
- Files are easily accessible by the same user from different locations
- Demand for local storage at the desktop is reduced
- Storage can be added more economically and partitioned among users— reasonably scalable
- Data can be backed up from the common repository more efficiently than from desktops
- Multiple file servers can be consolidated into a single managed storage pool

Storage Architectures

(Storage Area Networks (SAN))





SAN (Storage Area Network)

what is it?

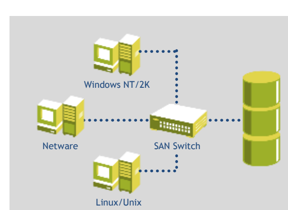
In short, SAN is essentially just another type of network, consisting of storage components (instead of computers), one or more interfaces, and interface extension technologies. The storage units communicate in much the same form and function as computers communicate on a LAN.

Advantages of SANs

- Superior Performance
- Reduces Network bottlenecks
- Highly Scalable
- Allows backup of storage devices with minimal impact on production operations
- Flexibility in configuration

Additional Benefits of SANs

•Storage Area Network (SAN)



- Server Consolidation
- Storage Consolidation
- Storage Flexibility and Management
- LAN Free backup and archive
- Modern data protection (change from traditional tape backup to snap-shot, archive, geographically separate mirrored storage)

Additional Benefits of SANs

- Disks appear to be directly attached to each host
- Provides potential of direct attached performance over Fibre Channel distances (Uses block level I/O)
- Provides flexibility of multiple host access
  - Storage can be partitioned, with each partition dedicated to a particular host computer
  - Storage can be shared among a heterogeneous set of host computers
- Economies of scale can reduce management costs by allowing administration of a centralized pool of storage and allocating storage to projects on an as-needed basis
- SAN can be implemented within a single computer room environment, across a campus network, or across a wide area network