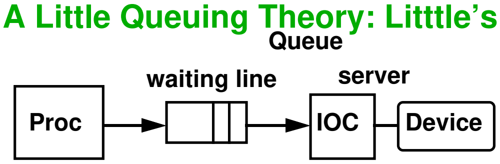
Review: Storage System Issues

- Historical Context of Storage I/O
- Storage I/O Performance Measures
- Secondary and Tertiary Storage Devices
- A Little Queuing Theory
- Processor Interface Issues
- I/O & Memory Buses
- Redundant Arrays of Inexpensive Disks (RAID)
- ABCs of UNIX File Systems
- I/O Benchmarks
- Comparing UNIX File System Performance

Review: A Little Queuing Theory: Litttle's Theorem



- Queuing models assume state of equilibrium :
  input rate = output rate
- Notation:

  r average number of arriving customers/second
  T_s average time to service a customer
  u server utilization (0..1): $u = r \times T_s$
  T_w average time/customer in waiting line
  T_q average time/customer in queue: $T_q = T_w + T_s$
  L_w average length of waiting line: $L_w = r \times T_w$
  L_q average length of queue: $L_q = r \times T_q$

- Little's Law: $r = L_q / T_q = L_w / T_w = u / T_s$
  Mean number customers = arrival rate x mean response time

Review: A Little Queuing Theory: M/G/1 and M/M/1

- Assumptions so far
  - equilibrium
  - time between two success arrivals in line are random
  - server can start on next customer immediately after finish prior
  - no limit to the waiting line: works First-In-First-Out
  - Afterward, all customers in line must complete; each avg. T_s
- Described Markovian request arrival (M for C=1exponentially random), General service distribution (no restrictions), 1 server: *M/G/1 queue*
- When Service times have C =1, *M/M/1 queue*

  $T_w = T_s \times u \times (1 + C) / (2 \times (1 - u)) = T_s \times u / (1 - u)$

  T_s average time to service a customer

  u server utilization (0..1): $u = r \times T_s$

  T_w average time/customer in waiting line
- Some confusion: waiting time = queue delay?

Review: Processor Interface Issues

- Processor interface
  - interrupts
  - memory mapped I/O
- I/O Control Structures
  - polling
  - interrupts
  - DMA
  - I/O Controllers
  - I/O Processors

Review: Relationship to Processor Architecture

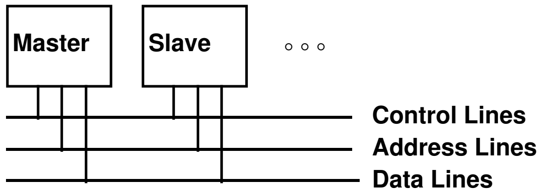
- I/O instructions and busses have disappeared
- Interrupt vectors have been replaced by jump tables
  PC <- M [IVA + interrupt number]
  PC <- IVA + interrupt number
- Interrupt stack replaced by shadow registers

  handler saves registers and re-enables higher priority interrupts
- Interrupt types reduced in number
  handler must query interrupt controller
- Caches required for processor performance cause problems for I/O
  - flushing is expensive, I/O pollutes cache
  - solution is borrowed from shared memory multiprocessors "snooping"
- virtual memory frustrates DMA
- load/store architecture at odds with atomic operations
  - load locked, store conditional
- stateful processors hard to context switch

Buses

- Bus: a shared communication link between subsystems.
  - low cost: a single set of wires is shared multiple ways
  - versatility:Easy to add new devices & peripherals may even be ported between computers using common bus
- Disadvantage: a communication bottleneck, possibly limiting the maximum I/O throughput
- Bus speed is limited by physical factors:
  - the bus length
  - the number of devices (and, hence, bus loading).
  - these physical limits prevent arbitrary bus speedup.
- 2 generic types of buses
  - I/O buses: lengthy, many types of devices connected, wide range in the data bandwidth), and follow a bus standard
  - CPU–memory buses: high speed, matched to the memory system to maximize memory–CPU bandwidth, single device
  - To lower costs, low cost (older) systems combine together
- bus transaction: sending address & receiving or sending data

Bus Protocols



Multibus: 20 address, 16 data, 5 control, 50ns Pause

Bus Master: has ability to control the bus, initiates transaction

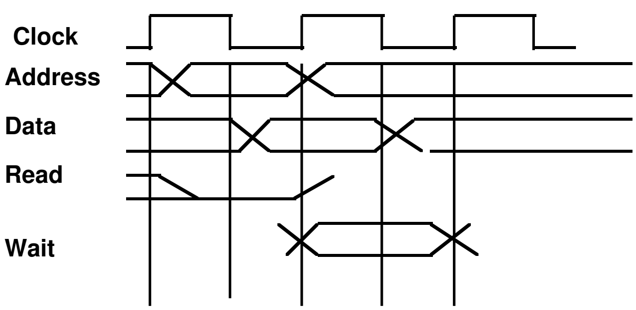
Bus Slave: module activated by the transaction

Bus Communication Protocol: specification of sequence of events and timing requirements in transferring information.

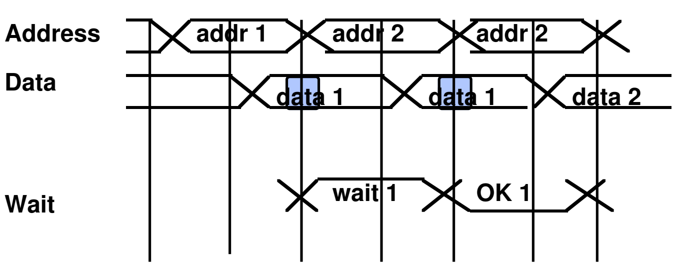
Asynchronous Bus Transfers: control lines (req., ack.) serve to orchestrate sequencing.

Synchronous Bus Transfers: sequence relative to common clock.

Synchronous Bus Protocols

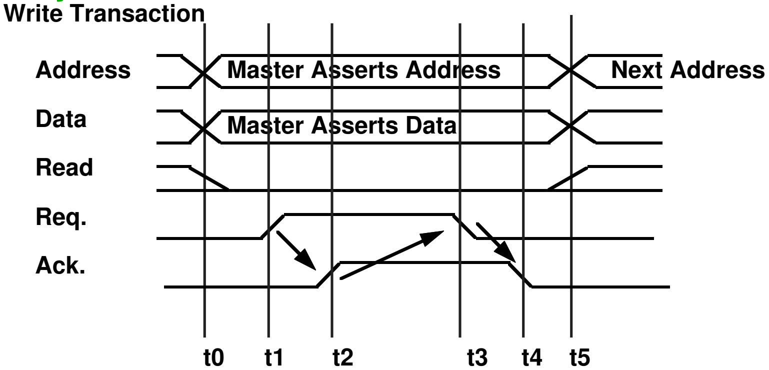


Pipelined/Split transaction Bus Protocol



Asynchronous Handshake

Write Transaction



t0 : Master has obtained control and asserts address, direction, data
Waits a specified amount of time for slaves to decode target\

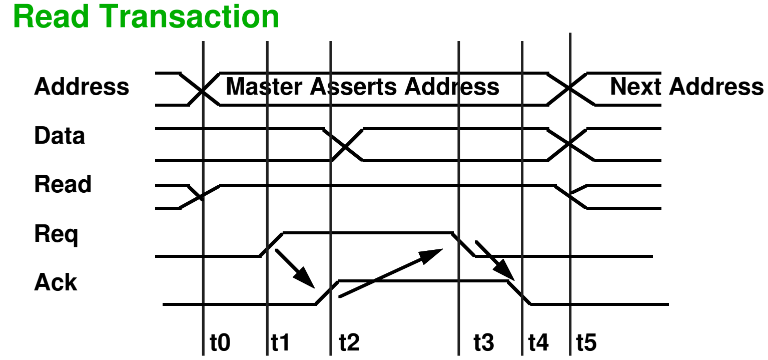
t1: Master asserts request line

t2: Slave asserts ack, indicating data received

t3: Master releases req

t4: Slave releases ack

Read Transaction



t0 : Master has obtained control and asserts address, direction, data
Waits a specified amount of time for slaves to decode target\

t1: Master asserts request line

t2: Slave asserts ack, indicating ready to transmit data

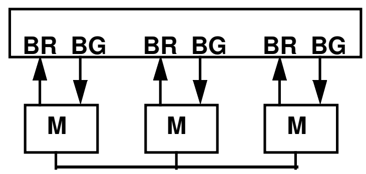
t3: Master releases req, data received

t4: Slave releases ack

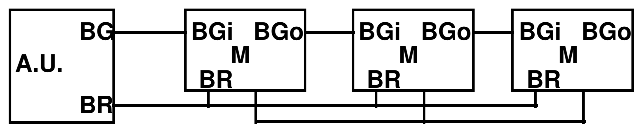
Time Multiplexed Bus: address and data share lines

Bus Arbitration

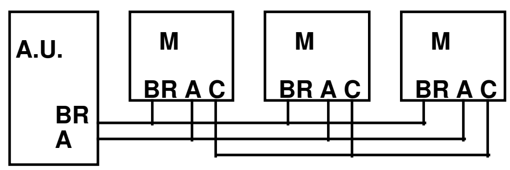
Parallel (Centralized) Arbitration



Serial Arbitration (daisy chaining)



Polling



Bus Options

Option	*High performance*	*Low cost*
Bus width	Separate address & data lines	Multiplex address & data lines
Data width	Wider is faster (e.g., 32 bits)	Narrower is cheaper (e.g., 8 bits)
Transfer size	Multiple words has less bus overhead	Single-word transfer is simpler
Bus masters	Multiple (requires arbitration)	Single master (no arbitration)
Split transaction?	Yes—separate Request and Reply packets gets higher bandwidth (needs multiple masters)	No—continuous connection is cheaper and has lower latency
Clocking	Synchronous	Asynchronous

1990 Bus Survey

	VME	FurtureBus	Multibus II	IPI	SCSI
Signals	128	96	96	16	8
Addr/Data mux	no	yes	yes	n/a	n/a
Data width	16 - 32	32	32	16	8
Masters	multi	multi	multi	single	multi
Clocking	Async	Async	Sync	Async	either
MB/s (0ns, word)	25	37	20	25	1.5 (asyn) 5 (sync)
150ns word	12.9	15.5	10	=	=
0ns block	27.9	95.2	40	=	=
150ns block	13.6	20.8	13.3	=	=
Max devices	21	20	21	8	7
Max meters	0.5	0.5	0.5	50	25
Standard	IEEE 1014	IEEE 896.1	ANSI/IEEE 1296	ANSI X3.129	ANSI X3.131

1993 I/O Bus Survey

Bus	SBus	TurboChannel	MicroChannel	PCI
Originator	Sun	DEC	IBM	Intel
Clock Rate (MHz)	16-25	12.5-25	async	33
Addressing	Virtual	Physical	Physical	Physical
Data Sizes (bits)	8,16,32	8,16,24,32	8,16,24,32,64	8,16,24,32,64
Master	Multi	Single	Multi	Multi
Arbitration	Central	Central	Central	Central
32 bit read (MB/s)	33	25	20	33
Peak (MB/s)	89	84	75	111 (222)
Max Power (W)	16	26	13	25

1993 MP Server Memory Bus Survey

Bus	Summit	Challenge	XDBus
Originator	HP	SGI	Sun
Clock Rate (MHz)	60	48	66
Split transaction?	Yes	Yes	Yes?
Address lines	48	40	??
Data lines	128	256	144 (parity)
Data Sizes (bits)	512	1024	512
Clocks/transfer	4	5	4?
Peak (MB/s)	960	1200	1056
Master	Multi	Multi	Multi
Arbitration	Central	Central	Central
Addressing	Physical	Physical	Physical
Slots	16	9	10
Busses/system	1	1	2
Length	13 inches	12? inches	17 inches

Improving Bandwidth of Secondary Storage

- **Processor performance growth phenomenal**
- **I/O?**

"I/O certainly has been lagging in the last decade."
Seymour Cray, Public Lecture (1976)

"Also, I/O needs a lot of work."
David Kuck, Keynote Address, (1988)

What have Magnetic Disk Designers been doing?

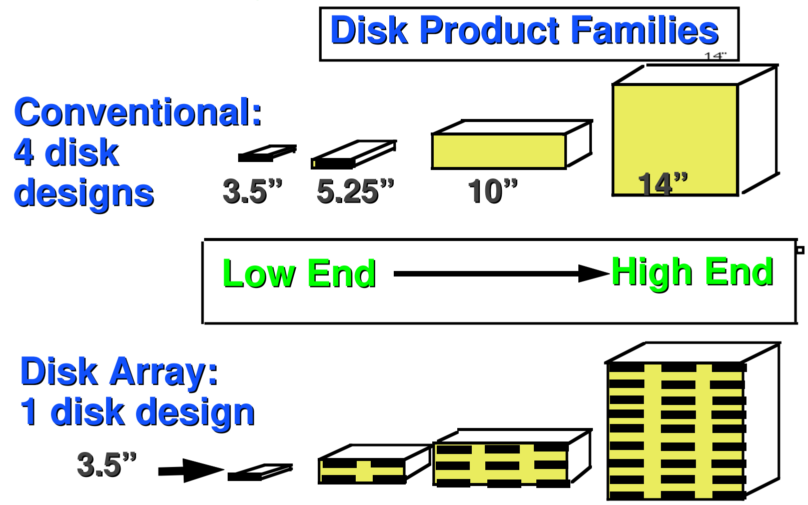
- Capacity: Improving 25% per year
- $/MB: Improving 25% per year
- Evolving to smaller physical size

14" ➡ 10" ➡ 8" ➡ 5.25" ➡ 3.5" ➡ 2.5" ➡ 1.8" ➡ 1"?

Smaller and cheaper vs. Larger and faster

Can smaller disks be used to close gap in performance between disks and CPUs?

Manufacturing advantages of disk arrays



cs252 4.19

Arrays of Disks to Close the Performance Gap

Replace Small Number of Large Disks

With Large Number of Small Disks!

cs252 4.20

	IBM 10.4" 3390	IBM 3.5" 0061	x 70
Data Capacity	23 GBytes	320 MBytes	23 GBytes
Volume	97 cu. ft.	0.1 cu. ft.	11 cu. ft.
Power	3 KW	11 W	1 KW
Data Rate	15 MB/s	1.5 MB/s	120 MB/s
I/O Rate	600 I/Os/s	55 I/Os/s	3900 IOs/s
Cost	≈$200K	≈$2K	≈$150K
MTTDL (hrs)	250K	50K	???

cs252 4.21

Disk Arrays have potential for:

- large data and I/O rates
- high MB per cu. ft.
- high MB per KW

But is there a catch?

cs252 4.22

Array Reliability

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

50,000 Hours ÷ 70 disks = 700 hours

Disk system MTTF: Drops from
6 years to 1 month!

• Arrays Too Unreliable to be useful!

Improving Reliability with Redundancy

• Add redundant drives to handle failures

Redundant
Array of
Inexpensive
Disks

RAID

• Redundancy offers 2 advantages:

1) Data not lost:
Reconstruct data onto new disk
2) Continuous operation
in the presence of failure

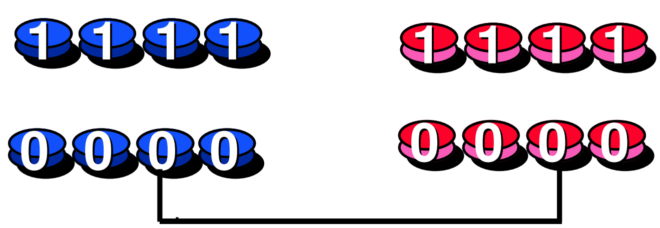
Several RAID organizations

Mirroring/Shadowing	***(Level 1 RAID)***
ECC	***(Level 2 RAID)***
Parity	***(Level 3 RAID)***
Rotated Parity	***(Level 5 RAID)***

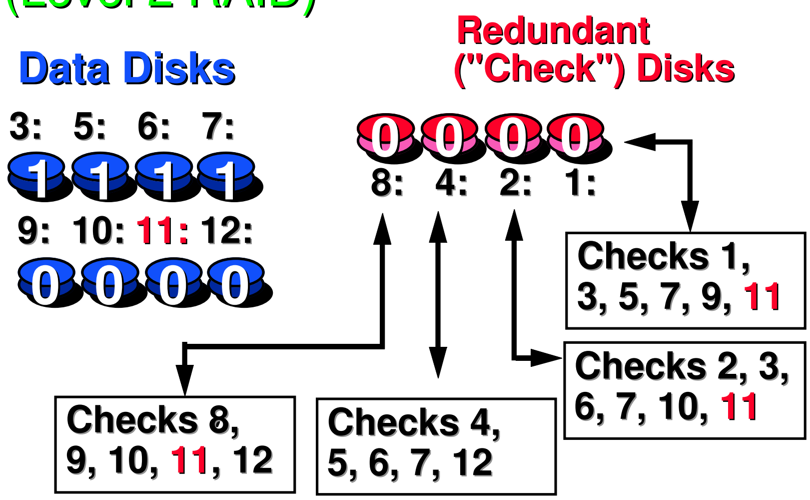
Redundancy via Mirroring (Level 1 RAID)

Data Disks

Redundant ("Check") Disks

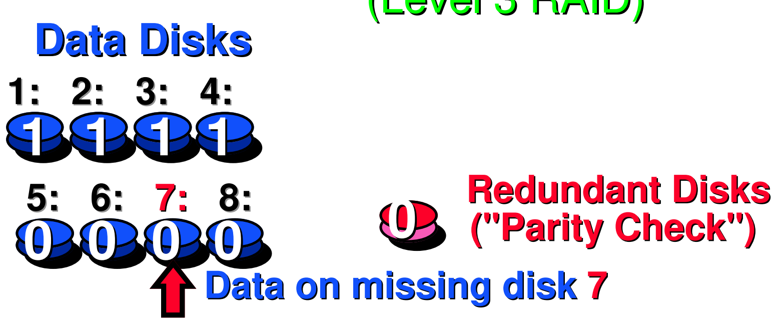


Redundancy via ECC
(Level 2 RAID)



cs252 4.27

Redundancy via Parity
(Level 3 RAID)



Reed/Solomon code per sector identifies failed disk

cs252 4.28

Redundancy via Rotated Parity
(Level 5 RAID)