CS/COE1541: Intro. to Computer Architecture

Input/output subsystem

Sangyeun Cho

Computer Science Department
University of Pittsburgh

Input/output (I/O)

- I/O connects
  - User (human) and CPU (or a program running on it)
  - Environment (physical world) and CPU (or a program running on it)
  - Bottom line: I/O devices and CPU
- I/O devices have vastly different performance requirements
  - *E.g.*, Hard disk vs. mouse
  - Bandwidth (*e.g.*, block-oriented) vs. latency (*e.g.*, character-oriented)
- I/O system architecture
  - Control: Polling (I/O device is "passive") vs. interrupt (I/O device is "active")
  - Synchronous or asynchronous buses
  - Parallel or serial buses
  - DMA (direct memory access)
- I/O interfaces are driven by standards

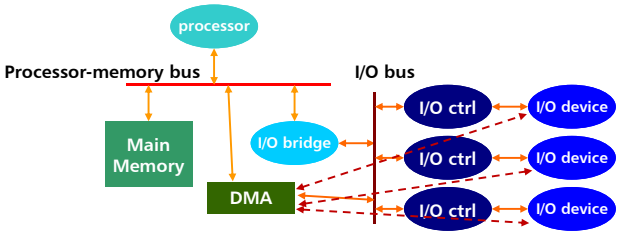
I/O devices

- To/from users
  - Display, keyboard, mouse, …
- To/from non-volatile storage
  - Hard disk, tape, flash drives, …
- To/from other computers
  - Network interface card (NIC), …
- Questions
  - What are the performance requirements of these devices?
  - Are there changing needs?

Performance aspects

- I/O devices typically have fixed bandwidth requirements
  - What is the necessary bandwidth (to/from video buffer) for a 1024x768 screen at 60Hz refresh rate?
- Two tiers
  - Between user and device
    - Display screen ↔ video card (VGA cable & signal interface)
  - Between device and CPU
    - Video card ↔ CPU (*e.g.*, AGP)
- Another bandwidth example
  - How many transactions per second can be handled?
- Latency

I/O bus



- Processor-memory bus
  - Masters: processor, I/O bridge (esp. DMA)
  - Slave: main memory
- I/O bus
  - Decouples memory and I/O bus transactions
  - Accommodates (slower) I/O devices and allows efficient data transfer using dedicated buffers and DMA

CPU-I/O interfacing

- Memory-mapped I/O
  - I/O control bits (*e.g.*, turn on, turn off, change color, …) are located in memory addresses
  - Use regular load and store to read from and write to those bits
  - These addresses are "volatile"
- Dedicated I/O address space and instructions
- In any case, I/O device control is done via well-defined "interface" – register specification

DMA (direct memory access)

- Taps into the main memory bus and I/O device buffers
  - From I/O device to main memory (*e.g.*, copying a new network packet from NIC to main memory)
  - From main memory to I/O device (*e.g.*, writing a block of data from memory to hard disk)
  - From I/O device to I/O device
  - From a main memory location to another
- Key is "programmability"
  - Source and destination
  - Total transfer size (how many bytes in total?)
  - Transfer unit (how many bytes at a time?)
  - Transfer triggering condition & transfer termination condition
  - Various event notification methods (*e.g.*, termination, error, …)
- DMA vs. CPU priority

DMA and virtual memory

- Different virtual memory pages have different physical page numbers
  - DMA operation over multiple pages can pose a problem – "what is the next page?"
- Solutions
  - DMA using VM addresses
    - With translation hardware
    - OS does not remap those pages until DMA is done
  - Partition DMA transfer into pages
    - OS chains the pages for the requester

DMA and cache coherence

- Two copies
  - One in cache
  - One in main memory
    - When transferring data from memory to I/O
    - When transferring data from I/O to memory
- Solutions
  - Do not cache I/O data
  - Flush cache whenever needed (*e.g.*, write dirty data to main memory before I/O write)
  - Similarly, invalidate cache before I/O read
  - Primitives (special instructions or registers to control the actions) are provided by hardware

Bus

- Connects different components in a computer system
  - CPU-memory
  - Memory-I/O
  - Chip-to-chip
  - There are "serial" buses
- Processor-memory bus vs. I/O bus
- Synchronous vs. asynchronous
- Master & slave
  - When we have multiple masters, we need "arbitration"
- Address & data
  - Separate signals or multiplexed
- Split transaction bus
  - Decouples address transfer and data transfer and allows multiple address transfers before the first data transfer takes place ⇒ increased bus utilization, high bandwidth at the cost of increased design complexity

Master & slave

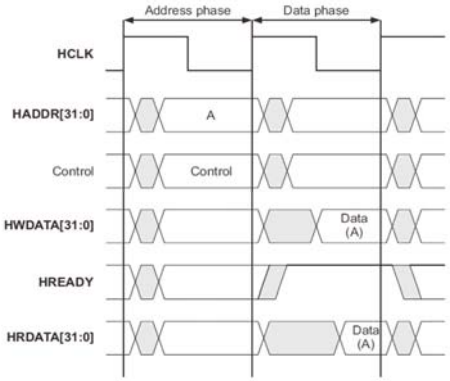
- Master
  - Bus entity that can initiate a bus transaction
  - Examples: CPU, DMA
- Slave
  - Bus entity that does not initiate a transaction by itself
  - Rather, it only responds to a request from a bus master
  - Example: Memory

Bus signals

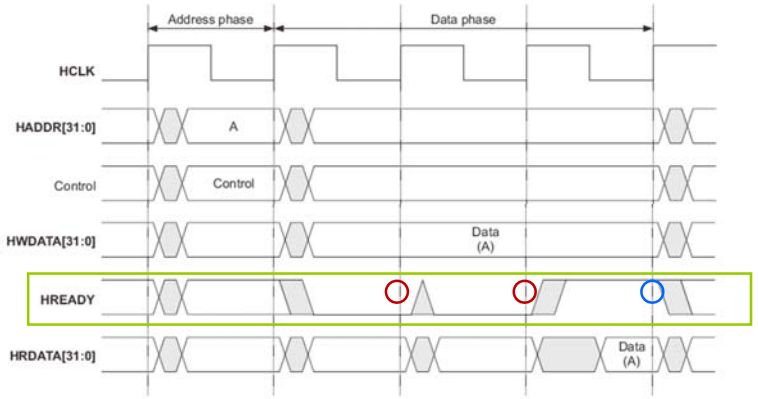
- Address
- Data
- Control
  - Signals that show or govern: bus transaction type, arbitration, data transfer timing, exception, etc.
- As to timing, address and control signals precede data for read (why?) or they can come (close) together for write

AMBA example

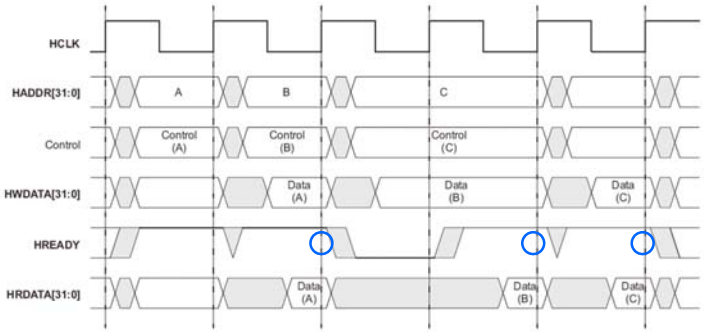
- AMBA (Advanced Microcontroller Bus Architecture) from ARM



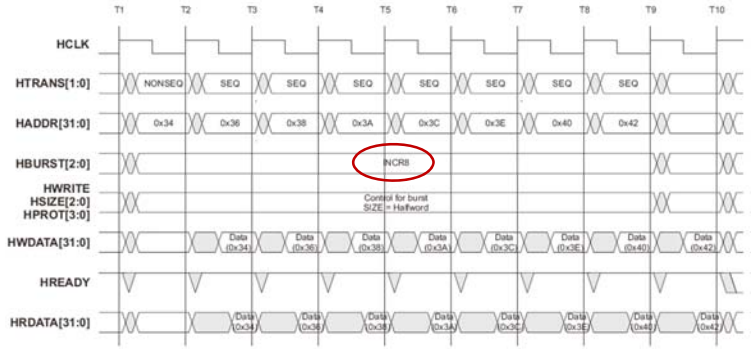
Transfer with "wait" states



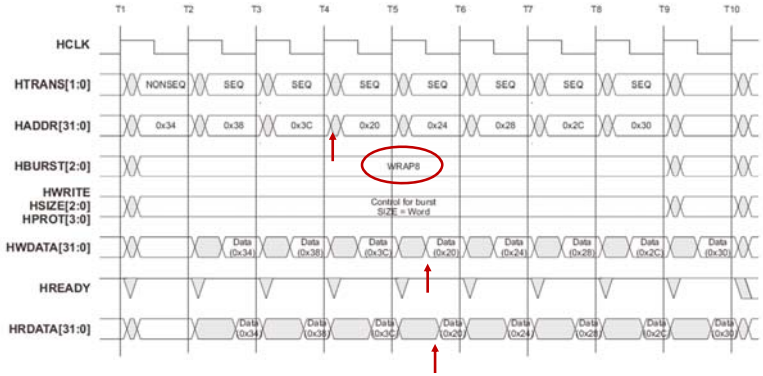
Multiple transfers



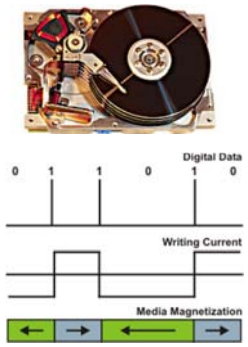
Burst, sequential (increasing)

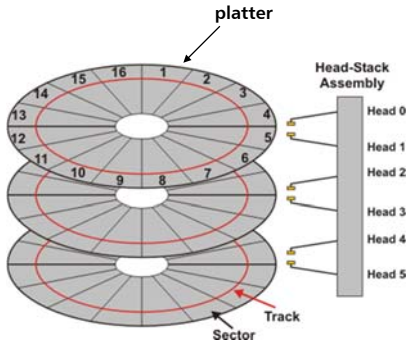


Burst, wrapping



Magnetic disk drive





- Stack of platters
- Two surfaces per platter
- Tracks & sectors

- Heads move together (single arm)
- Disk access time
  - Queueing + seek
  - Rotation + transfer

Magnetic disk drive, performance

- Queueing time
  - How many jobs are queued at the time of request
- Seek time
  - Time needed to move the arm to the target track
  - Mechanical!
- Rotation delay
  - Time for the target sector to be under the head
  - ½ rotation time on average
  - Mechanical!
- Transfer time
  - Time to read out, buffer, and transfer data
  - Has to do with rotation speed, recording density, buffering method, and interface used

Improving magnetic disk drive

- Use smart scheduling to reduce queueing time
  - Locality-based scheduling
- Faster seeking or arm movement
  - (Mechanical)
- Faster rotation speed (currently 7200rpm or higher)
  - (Mechanical)
- Denser recording
  - *E.g.*, vertical magnetization "Get perpendicular" - http://www.hitachigst.com/hdd/research/recording_head/pr/PerpendicularAnimation.html
- Larger buffer (currently 16MB for PCs)
  - Smart caching
  - Flash buffer (a.k.a. hybrid hard drive)
- Higher-bandwidth bus stanrards
  - Serial ATA or SATA: 150MB/s ⇒ 300MB/s ⇒ 600MB/s
  - C.f., parallel ATA or PATA: max 133MB/s

Magnetic disk drive, some metrics

- Capacity
  - GB
- Performance
  - Latency (ms) + transfer rate (MB/s)
- Power consumption
  - Watt
  - There are different modes (*e.g.*, spindle off)
- Form factor
  - Inch×inch×inch
- Reliability
  - MTBF (mean time between failures), POH (power-on hours)
- Vibration, thermal, EMI, …
- Cost
  - $$, MB/$

Hard disk product families

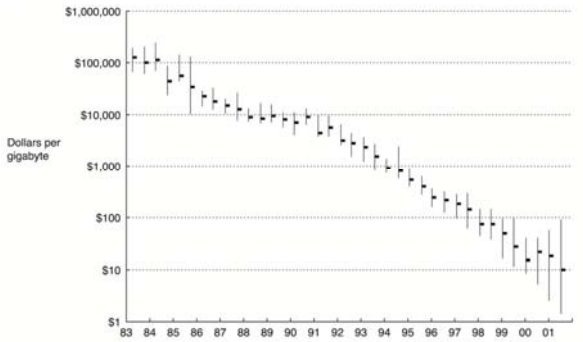
- PC/workstation (3.5-in)
  - Performance, price, and capacity
  - 200~1,000GB
- Notebook (2.5-in)
  - Power, capacity, and form factor
  - 40~250GB
- Embedded products
  - Cost sensitivity and reliability
  - 80~250GB (PVR)
- Micro-drive (<1-in)
  - Form factor, power, and cost
  - 2~8GB

Example specification

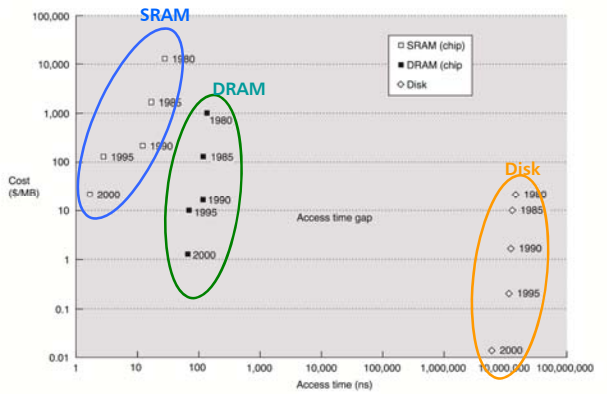
Drive Configuration	Interface	Serial ATA II 3.0Gbps
	Buffer DRAM Size	16 Mbytes
	Bytes per Sector	512
Performance Specification	Read Seek Time(typical)	
	Track to Track	0.8 ms
	Average	8.9 ms
	Full Stroke	18 ms
	Average Latency	4.17 ms
	Rotational Speed	7,200 rpm
	Data Transfer Rate	
	Media to/from Buffer (max.)	135 Mbits/sec
	Buffer to/from Host(max.)	300 MB/sec
	Drive Ready Time(typical)	10 sec
Reliability Specification	Non-recoverable Read Error	1 sector in 10^{14} bits
	MTBF	600,000 POH
	Start/Stop Cycles (Ambient)	50,000
	Component Design Life	5 years
Acoustics(Average Sound Power)	Idle	2.3 Bel
	Random Read/Write	2.7 Bel

Power Requirements	Voltage	+5V±5%,+12V±10%
	Spin Up Current (max.)	2.0 A
	Read/Write On-Track(typ.)	8.0 W
	Seek(typ.)	9.5 W
	Idle (typ.)	6.5 W
	Standby(typ.)	1.5/1.0 W Power consumption with/without slumber mode
	Sleep(typ.)	0.5/1.0 W Power consumption with/without slumber mode
Physical Dimension	Height	1 in
	Width	4 in
	Depth	5.75 in
	Weight	1.13 lb

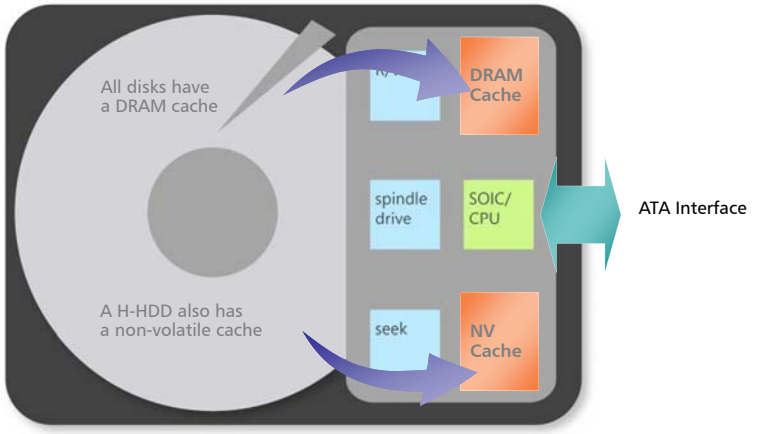
Price per GB trend



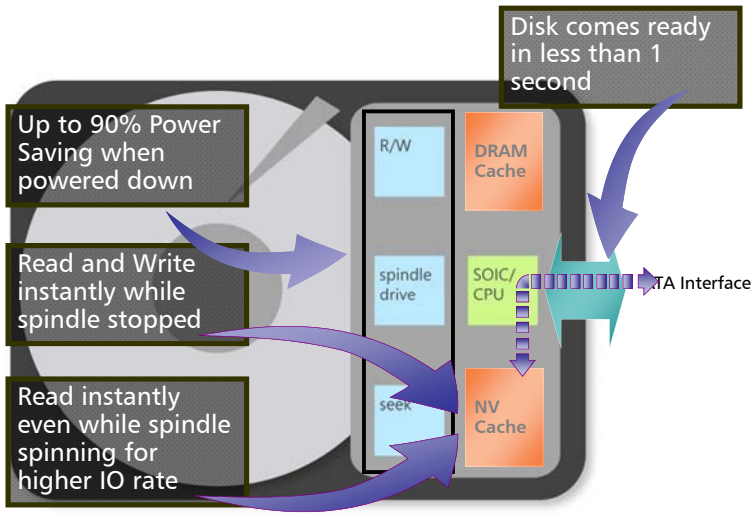
Cost vs. access time



Hybrid hard drive



Hybrid hard drive



HHD boot/resume (Windows)

- During shutdown or hibernate all the disk sectors needed to boot or resume are pinned into the NV cache
- On next power on the BIOS POST runs and the disk is powered on but the spindle won't be ready for 2~4 seconds
- BIOS can read data from the NV cache and all boot process I/O can be read from the NV cache
- Once the rotating media is ready I/O can be satisfied by both NV cache and rotating media for optimized read performance

Power saving mode (Windows)

- SuperFetch buffers disk data in system DRAM to fulfill reads
- Write I/Os buffered in NV cache while disk is spun down
- Disk spins up only when
  - Read cache miss
  - NV cache full
- The disk spin-down and continues to use the NV cache

Hybrid hard drive summary

- Basic idea
  - Implement a 64~128MB NV cache using NAND flash memory
    - The NV cache size will increase as NAND flash memory price goes down
- Potential benefits
  - Reduce boot time and resume time
  - Extend spindle down time ⇒ lower power
  - Higher reliability
- Problem
  - Cost
- Full flash-based hard drives are already on the market

Solid-state disk (SSD)

- Instead of magnetic media accessed via mechanical parts (spindle motor, arm, …), SSD uses only solid-state components (NAND flash chips) to store information
- NAND flash memories are optimized for storage applications (larger read/write data unit than other addressable memories like SRAM and DRAM)
  - C.f. NOR flash has been favored to store codes
- NAND flash memories are susceptible to write endurance
  - Various wear-leveling techniques have been developed
  - Hidden in flash translation layer or FTL
- A fun video: http://www.youtube.com/watch?v=96dWOEa4Djs

Optical discs

- Optical disc drives have become a standard archive/backup device for personal computing
- Based on optics/laser technology
- For writing, "phase-change" materials used (*i.e.*, two different reflection rate)

~700MB

~4.7GB

~27GB

Storage devices trend

- Hard disk
  - ~~We will see 1TB hard disks become mainstream in PC soon~~
    - 2× capacity increase per year
  - Perpendicular recording has been fully adopted
  - Hybrid disks has been used somewhat (from notebook market)
  - Solid-state disks are gaining momentum (Samsung's 64GB SSD was <$200 @Amazon, 11/06/2009)
- Optical disk
  - CD speed of ~52× (saturated)
  - Writeable DVD drives prevalent now (~16×, speed saturated)
  - Standard battle ended; Blu-ray group (Sony, Samsung, …) won over HD-DVD (previously AOD) group (Toshiba, NEC, …)
- Flash memory cards
  - 16GB cards and 8GB USB drives are available
  - Device (NAND flash) capacity has doubles every year (will slow down)

Reliability issues in storage

- We don't want to lose valuable data
- Fault is the cause of an error
  - When a fault occurs, it creates a latent error, which can later manifest itself
- System failure occurs because of a manifested error
- Example
  - Alpha particle hits DRAM cell (fault)
  - Bit inversed (error) – latent until it is read
  - Error is propagated and affects the delivered service (failure)

MTTF, MTTR, …

- MTTF (mean time to failure)
  - 1/MTTF: rate of failures
- MTTR (mean time to repair)
- MTBF (mean time between failures)
  - MTTF + MTTR
- Module availability = MTTF/MTBF

Reliability improvement

- Fault avoidance
  - How to prevent, by construction, fault occurrences
- Fault tolerance
  - How to provide, by redundancy, continued service in spite of faults
- Error removal
  - How to minimize, by verification, the presence of latent errors
- Error forecasting
  - How to estimate, by evaluation, the presence, creation, and consequences of errors

RAID

- Redundant array of inexpensive dirks
- Disk array with striped data can provide high bandwidth
  - But not necessarily smaller latency
- What about reliability?
  - Error rate with 100 hard disks is 100 times higher than that of 1 disk

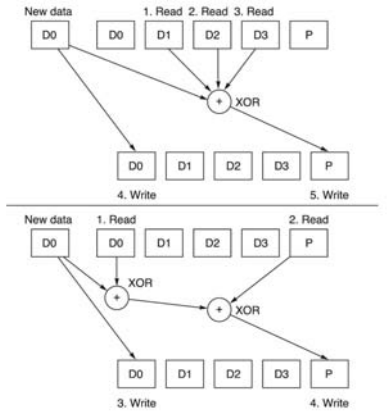
RAID 0 and 1

- RAID 0
  - Data striped
  - No provision for reliability
  - This is not really a RAID – it's AID
- RAID 1
  - Data mirrored
    - Every bit is written in two different disks
  - Expensive…

RAID 2 and 3

- RAID 2
  - Bit interleaved parity
  - Memory style ECC (error correction code)
  - Still expensive
- RAID 3
  - Byte-interleaved parity
  - Add 1 check disk for N disks (overhead ~1/N)

Optimizing small writes

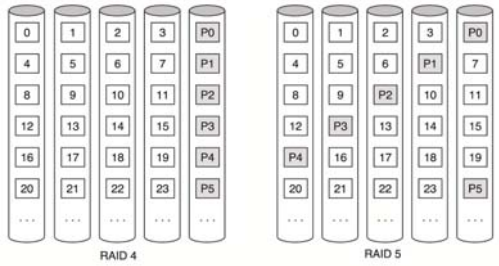


3R + 2W

2R + 2W

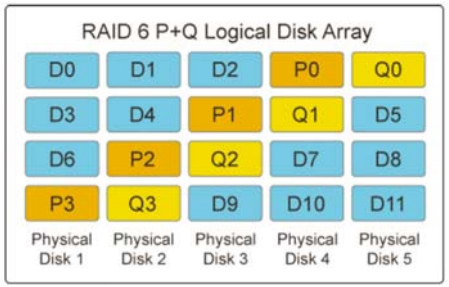
RAID 4 and 5

- RAID 4
  - Block-interleaved parity
  - 1 check disk for N data disks
  - Check is done on a block (with the original data block possibly residing in a single disk)
  - Allows parallel reads
- RAID 5
  - Same as RAID 4 but check blocks are also interleaved



RAID 6

- P+Q redundancy
- Added more parity for improved error recovery



Berkeley's RAID prototype

- c. 1989
- Sun 4/280 workstation w/ 128MB of DRAM
- Four dual-string SCSI controllers
- 28 5.25-in SCSI disks and specialized disk striping software