CS/ECE 552: Input/Output

Prof. Matthew D. Sinclair

Lecture notes based in part on slides created by Mark Hill, Mikko Lipasti, David Wood, Guri Sohi, John Shen and Jim Smith

Input/Output

- Part 1
  - Motivation
  - I/O Devices
- Part 2: Reliability
- Part 3: Buses
- Part 4: Interfacing

Motivation

- I/O necessary
  - To/from users (display, keyboard, mouse)
  - To/from non-volatile media (disk, tape)
  - To/from other computers (networks)

- Key questions
  - How fast?: Affects design of interfaces
  - What are the trends?: Getting faster?

Examples

Device	I or O?	Partner	Data Rate KB/s
Mouse	I	Human	0.01
Display	O	Human	373,000
Modem	I/O	Machine	2-8
LAN	I/O	Machine	100,000
Tape	Storage	Machine	2000
Disk	Storage	Machine	2000-100,000

Humans are asymmetric!

I/O Performance

- What is performance? For I/O means many things …

- Supercomputers read/write 1GB of data
  - Want high bandwidth to vast data (bytes/sec)

- Transaction processing: many independent small I/Os
  - Want high I/O rates (I/Os per sec)
  - May want fast response times

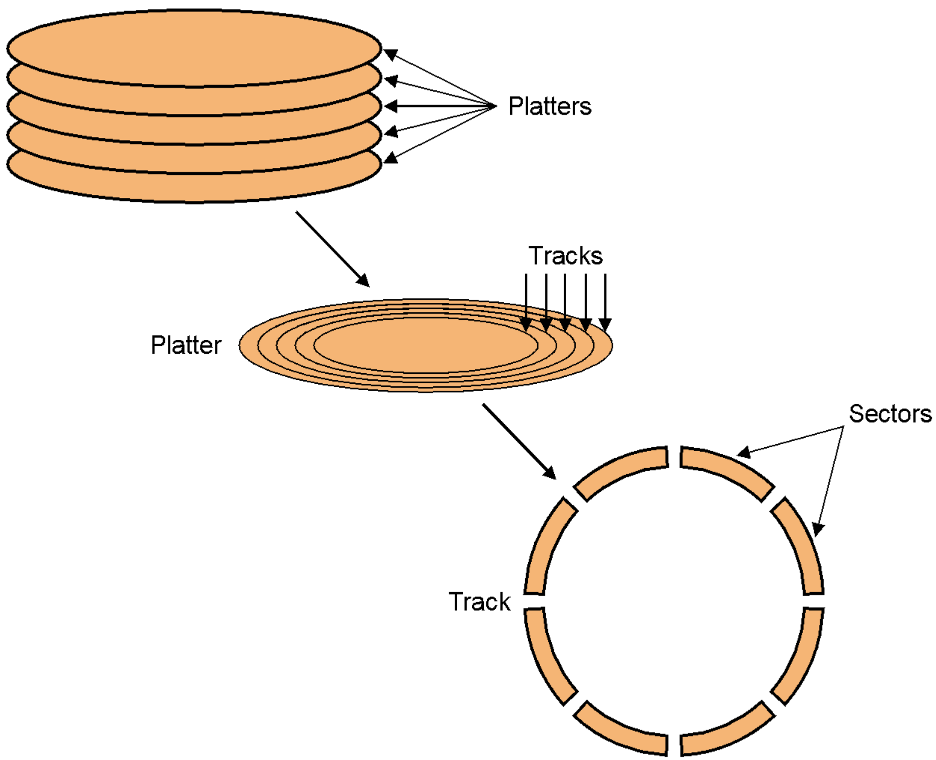
- File systems
  - Want fast response time first
  - Lots of locality

Magnetic Disks

- Stack of platters
- Two surfaces/platter
- Tracks
- Heads move together
- Sectors
- Disk access:

  Queueing delay +
  Seek time +

  Rotation time +
  transfer time



Magnetic Disks

- Seek = 10-20ms but smaller with locality
- Rotation = ½ rotation/3600rpm = 8.3ms
- Transfer = x / 2-4MB/s
  - E.g. 4kB/4MB/s = 1ms
- Remember: mechanical => ms

Disk Trends

- Disk trends
  - $/MB down (well below $.10/GB)
  - Disk diameter: 14” => 3.5” => 2.5” => 1.8” => 1”
  - Seek time reduced
  - Rotation speed increasing at high end
    - 5400rpm => 7200rpm => 10Krpm => 15Krpm
    - Slower when energy-constrained (laptop)
  - Transfer rates up
  - Capacity per platter way up (100%/year)
  - Hence, op/s/MB way down
    - High op/s demand forces excess capacity

GPU/Video Card

- Extreme bandwidth requirement just for frame buffer
  – 1920x1080 pixels x 24bits/pixel = 6.2MB
  – Refresh whole screen 60 times/sec = 373MB/s !
- 3D rendering amplifies bandwidth demand
  – Texture memory accesses, etc.
  – Result: need many GB of bandwidth
- GPUs use specialized, dedicated memory (GDDRx)
  – APUs share DDRx memory, can't keep up
- Connected via PCIe x16 to system memory

CS/ECE 552: Input/Output Part 2

Prof. Matthew D. Sinclair

Lecture notes based in part on slides created by Mark Hill, Mikko Lipasti, David Wood, Guri Sohi, John Shen and Jim Smith

RAID

- What if we need 100 disks for storage?
- MTTF = 5 years / 100 = 18 days!

Reliability: RAID

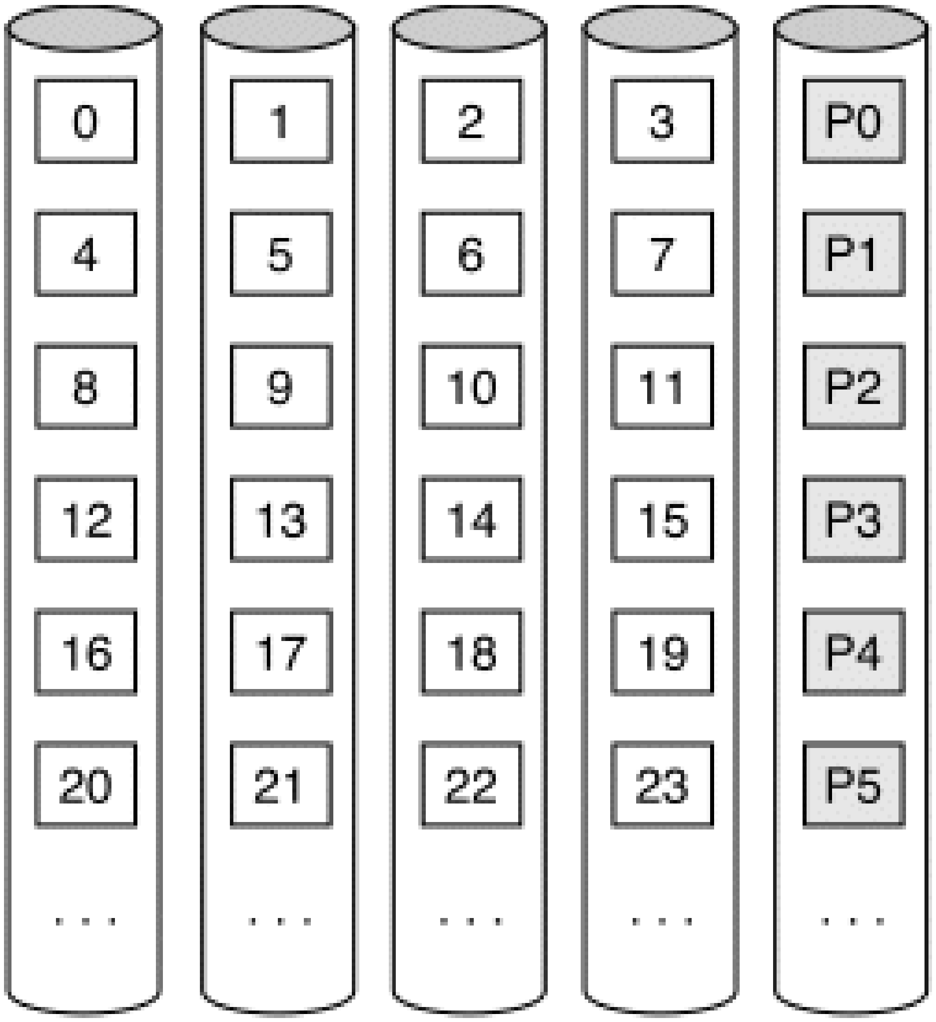
- **Error correction**: more important for disk than for memory
  - Error correction/detection per block (handled by disk hardware)
  - Mechanical disk failures (entire disk lost) most common failure mode
    - Many disks means high failure rates
    - Entire file system can be lost if files striped across multiple disks
- **RAID (redundant array of inexpensive disks)**
  - Add redundancy
  - Similar to DRAM error correction, but…
  - Major difference: which disk failed is known
    - Even parity can be used to recover from single failures
    - Parity disk can be used to reconstruct data faulty disk
  - RAID design balances bandwidth and fault-tolerance
  - Implemented in hardware (fast, expensive) or software

Levels of RAID - Summary

- **RAID-0 - no redundancy**
  - Multiplies read and write bandwidth
- **RAID-1 - mirroring**
  - Pair disks together (write both, read one)
  - 2x storage overhead
  - Multiples only read bandwidth (not write bandwidth)
- **RAID-3 - bit-level parity** (dedicated parity disk)
  - N+1 disks, calculate parity (write all, read all)
  - Good sequential read/write bandwidth, poor random accesses
  - If N=8, only 13% overhead
- **RAID-4/5 - block-level parity**
  - Reads only data you need
  - Writes require read, calculate parity, write data & parity
- **RAID-6 – diagonal parity**

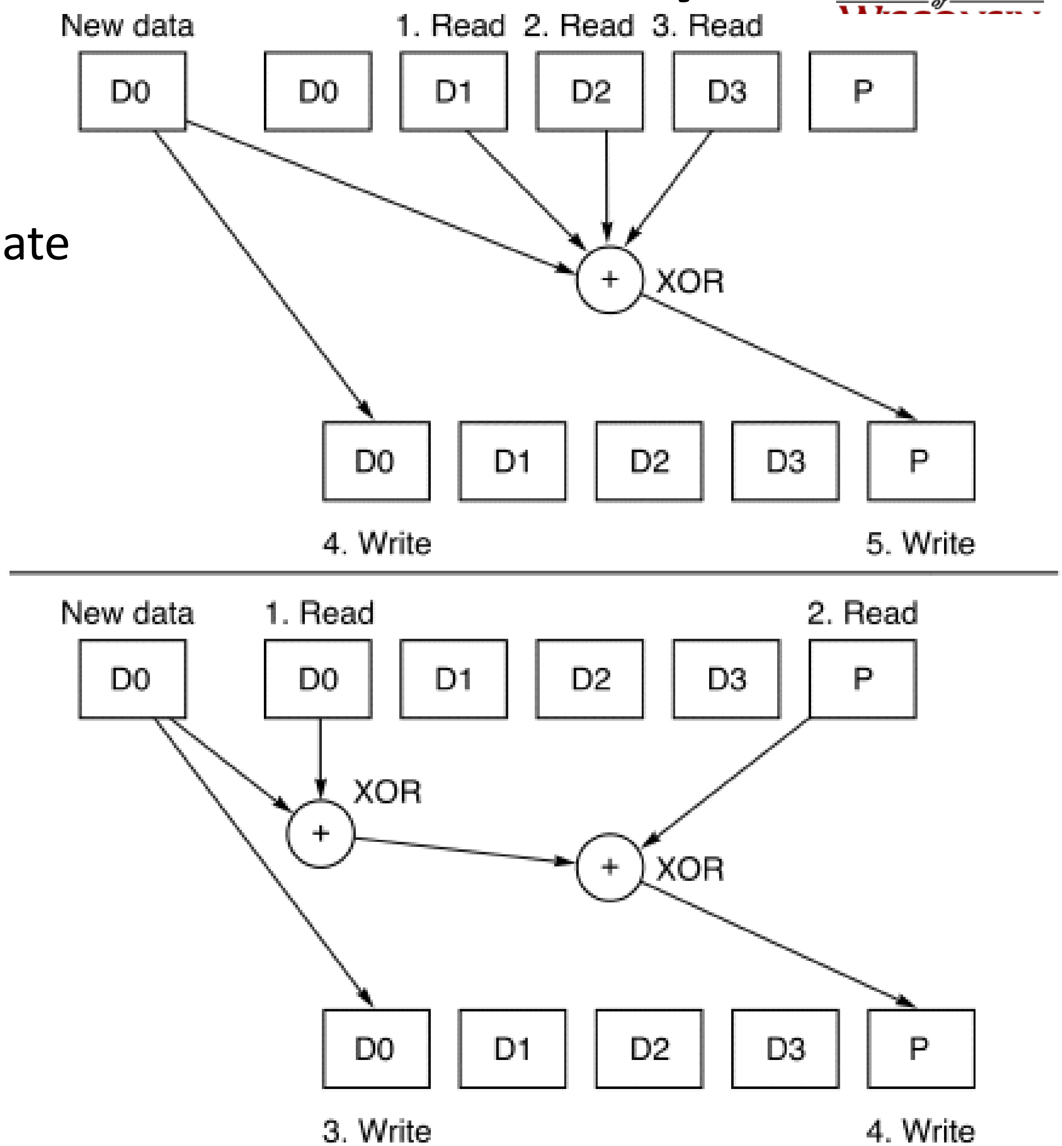
RAID-3: Bit-level parity

- **RAID-3 - bit-level parity**
  - dedicated parity disk
  - N+1 disks, calculate parity (write all, read all)
  - Good sequential read/write bandwidth, poor random accesses
  - If N=8, only 13% overhead



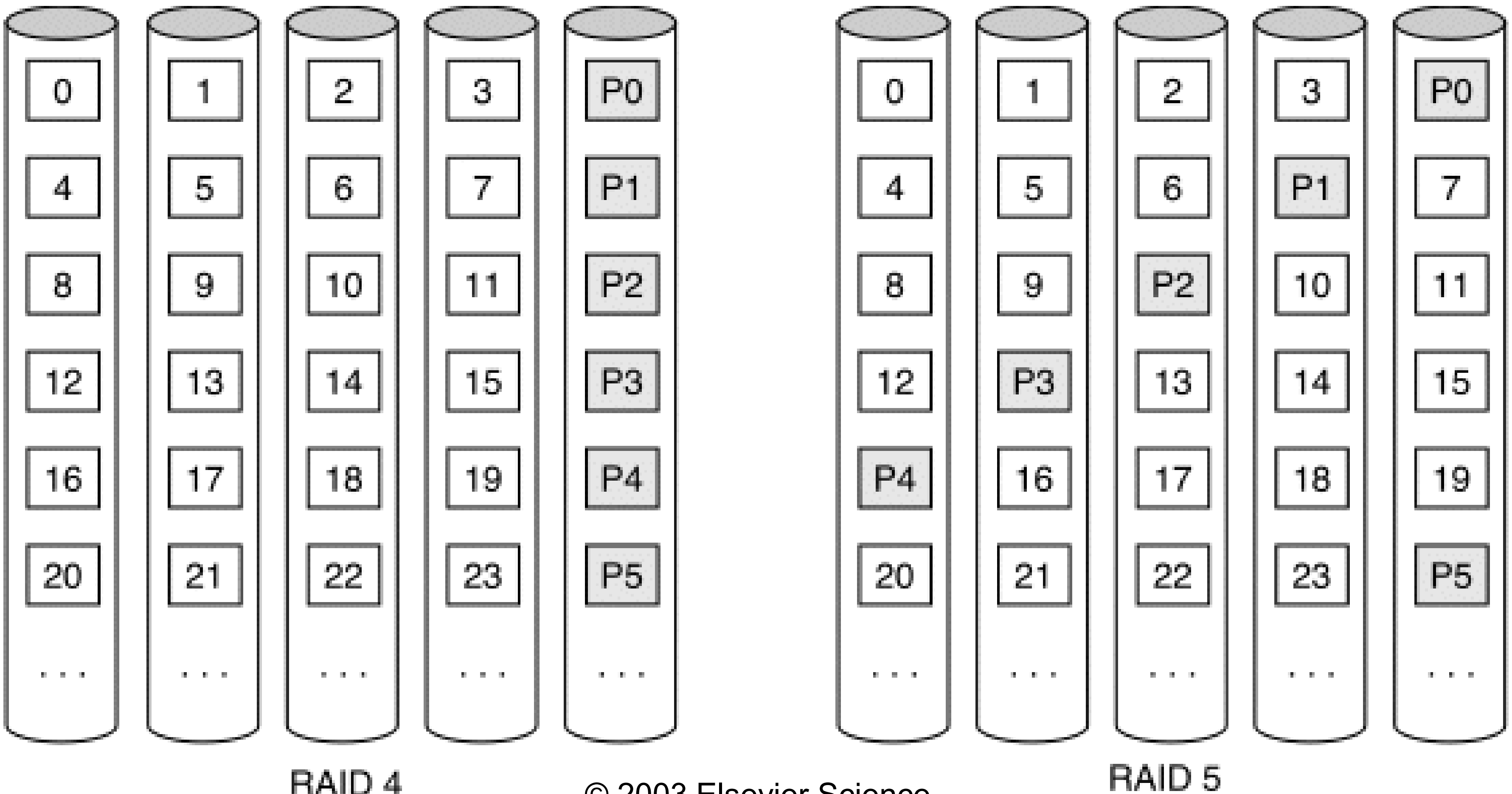
RAID 4/5 - Block-level Parity

- **RAID-4/5**
  - Reads only data you need
  - Writes require read, calculate parity, write data&parity
- Naïve approach
  1. Read all disks
  2. Calculate parity
  3. Write data&parity
- Better approach
  - Read data&parity
  - Calculate parity
  - Write data&parity
- Still worse for small **writes** than RAID-3

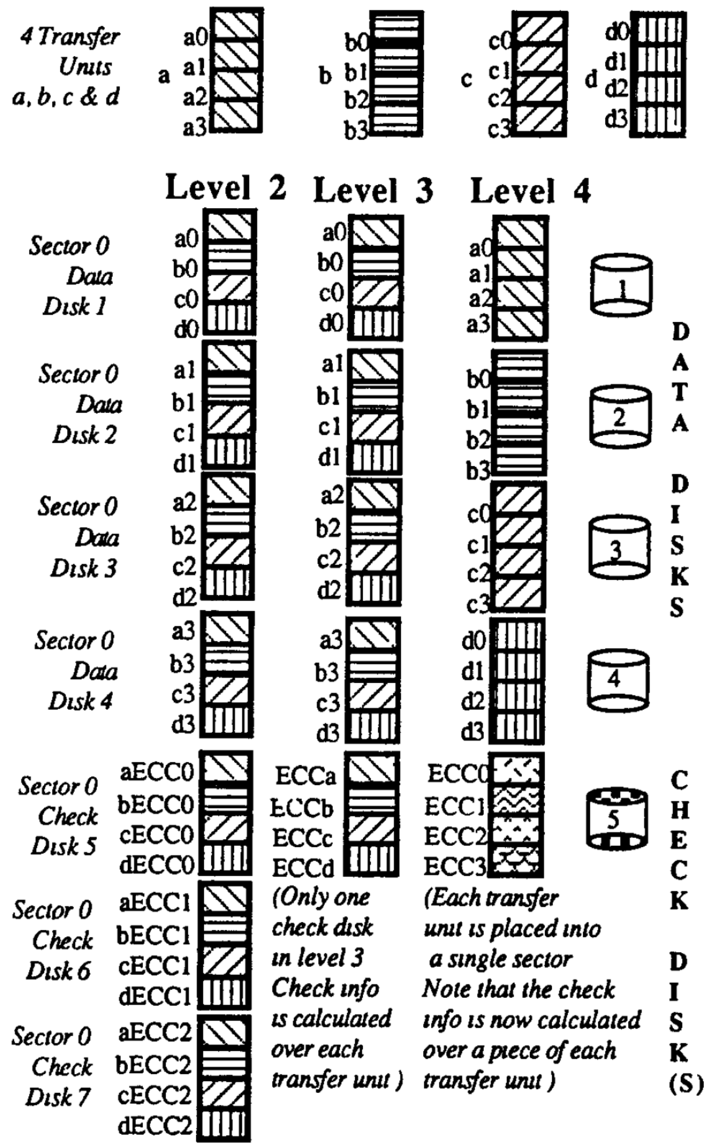


RAID-4 vs RAID-5

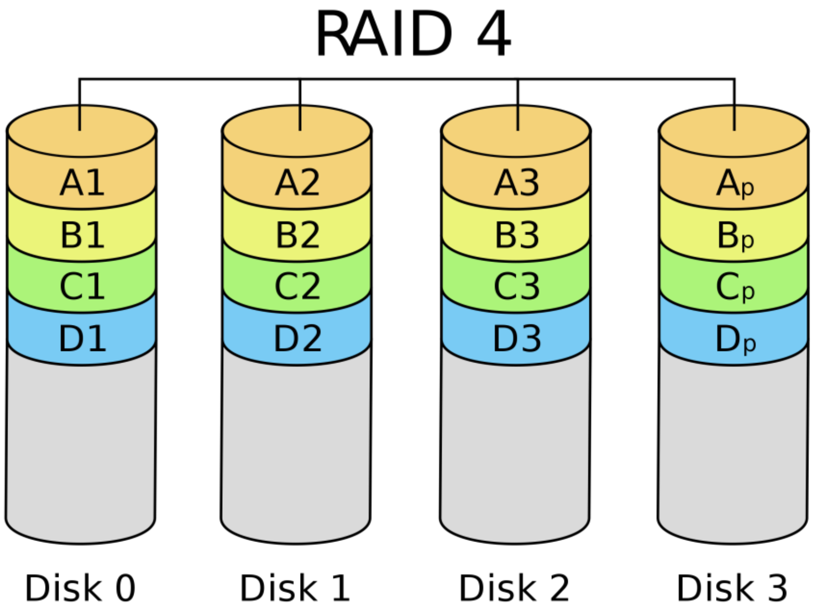
- RAID-5 rotates the parity disk, avoid single-disk bottleneck

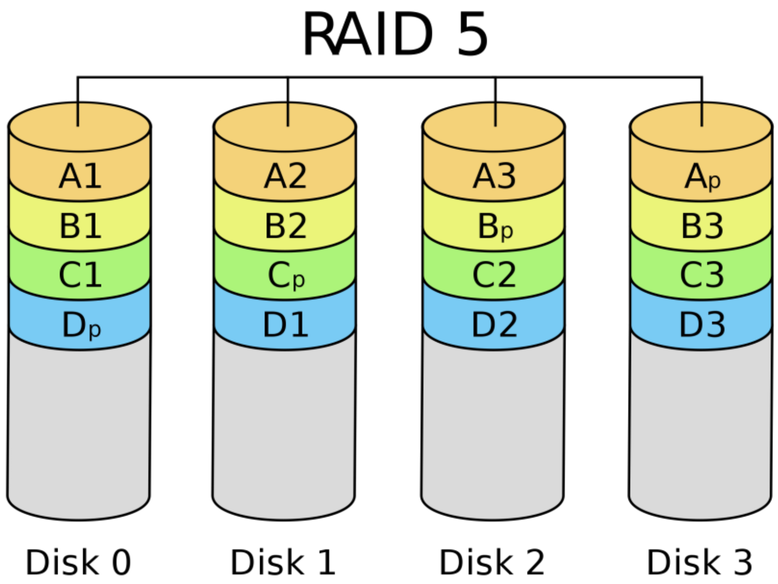


From the original paper:



In color: RAID 4 vs. RAID 5





Images: Wikipedia

In color: RAID 6

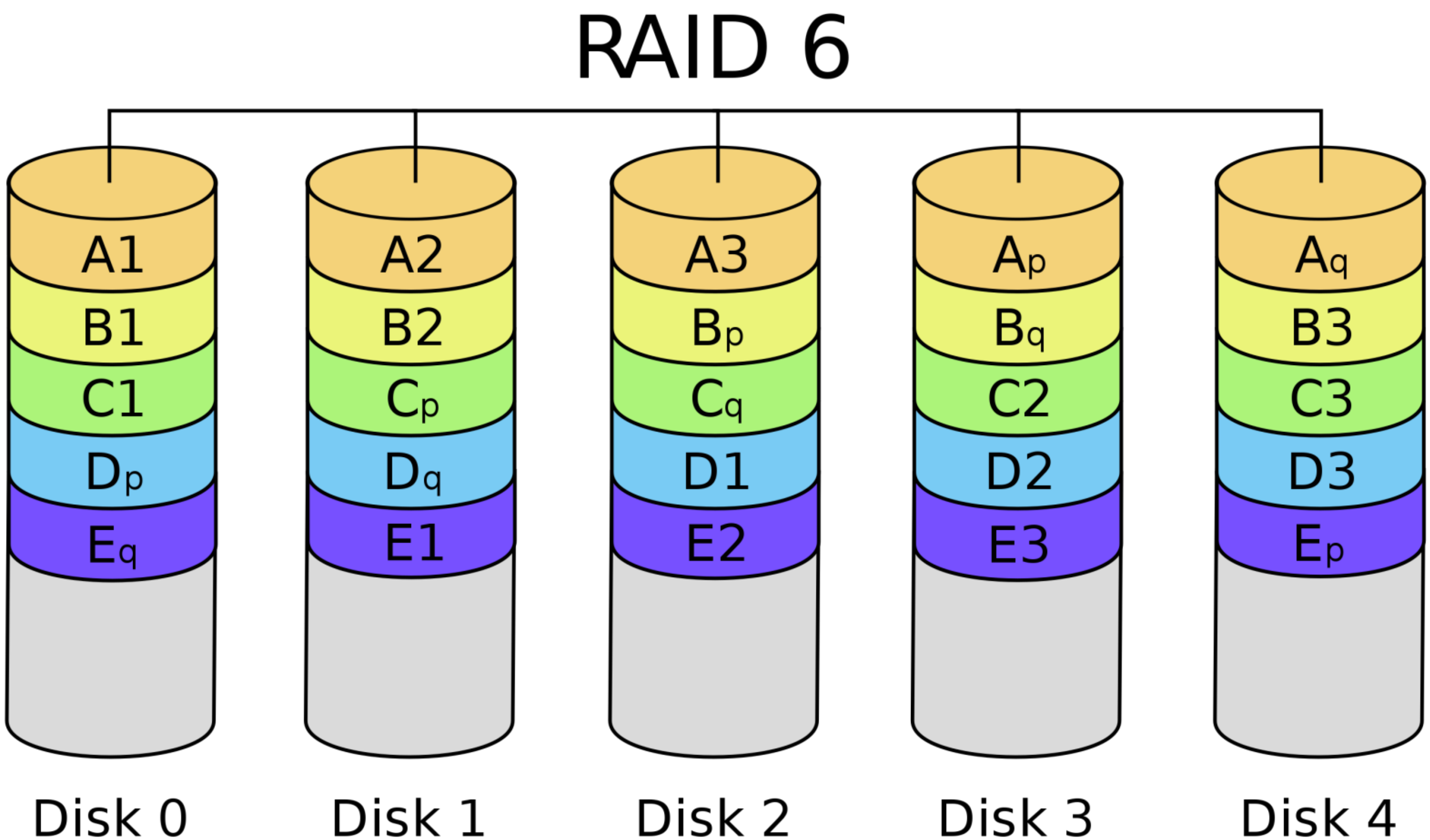
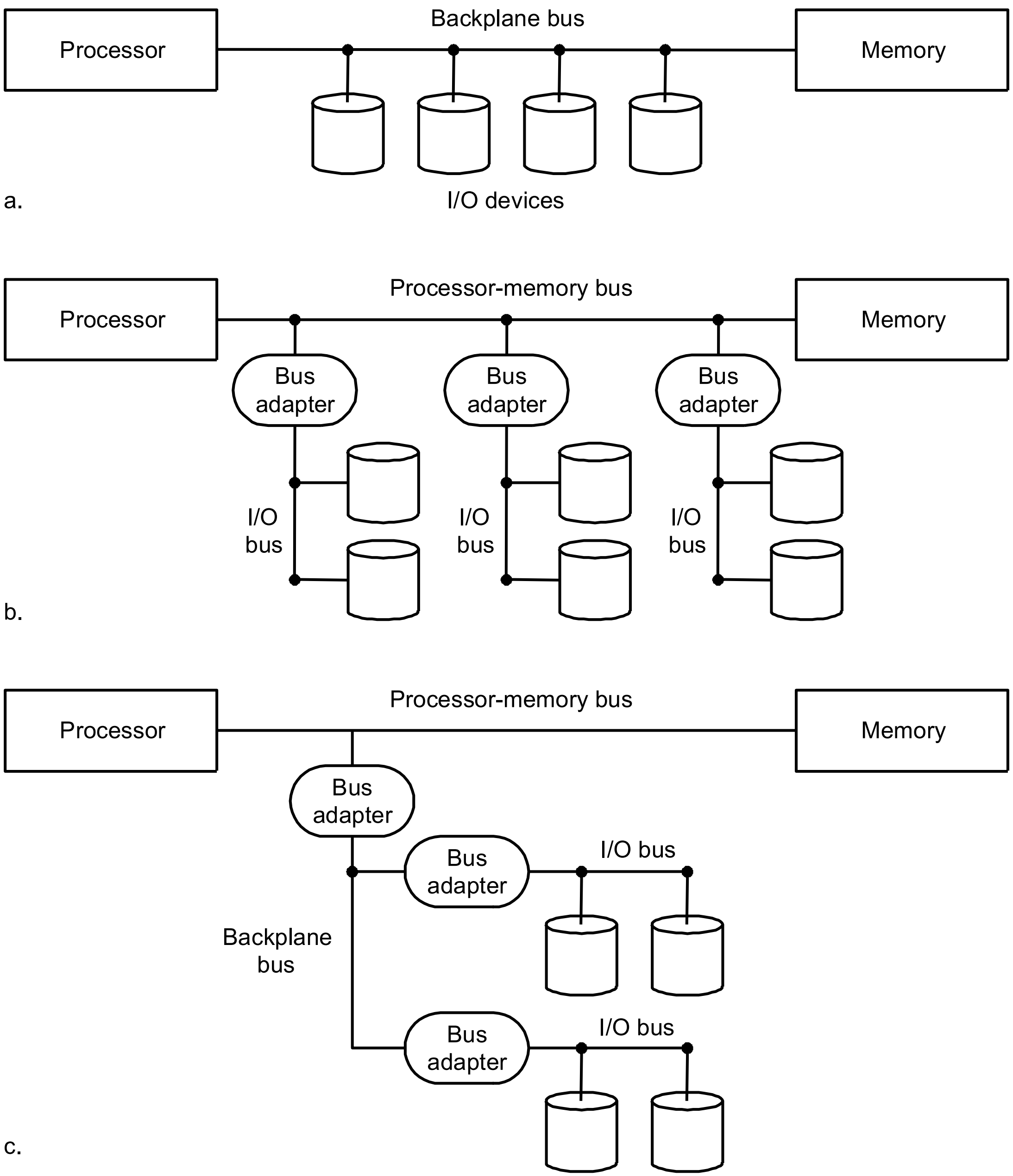


Image: Wikipedia

CS/ECE 552: Input/Output Part 3

Prof. Matthew D. Sinclair

Lecture notes based in part on slides created by Mark Hill, Mikko Lipasti, David Wood, Guri Sohi, John Shen and Jim Smith



Buses in a Computer System

Buses

- Bunch of wires
  - Arbitration
  - Control
  - Data
  - Address
  - Flexible, low cost
  - Can be bandwidth bottleneck

Buses

- Types
  - Processor-memory
    - Short length, fast (speed), custom
  - I/O
    - Long length, slower (speed), standard
  - Backplane
    - Medium length, medium (speed), standard

Buses

- Synchronous – has clock
  - Everyone watches clock and latches at appropriate phase
  - Transactions take fixed or variable number of clocks
  - Faster but clock limits length
  - E.g. processor-memory
- Asynchronous – requires handshake
  - More flexible
  - I/O

Buses

- Synchronous vs. asynchronous
  - Must distribute clock and deal with skew
  - Simple handshake
  - Backward compatibility difficult, esp. with slow devices
  - No metastability problems (FSD)

Buses

- Improving bandwidth
  - Wider bus
  - Block transfer to exploit spatial locality
  - Separate address/data lines
  - Split transactions (multiple concurrent requests)
  - Pipelined in-order responses
  - Out-of-order responses (add transaction ID)

Bus Arbitration

- One or more bus masters, others slaves
  - Bus request
  - Bus grant
  - Priority
  - Fairness

- Implementations
  - Centralized vs. distributed

CS/ECE 552: Input/Output Part 4

Prof. Matthew D. Sinclair

Lecture notes based in part on slides created by Mark Hill, Mikko Lipasti, David Wood, Guri Sohi, John Shen and Jim Smith

Interfacing

- Three key characteristics
  - Multiple users/programs share I/O resource
  - Overhead of managing I/O can be high
  - Low-level details of I/O devices are complex

- Three key functions
  - Virtualize resources – protection, scheduling
  - Use interrupts (similar to exceptions)
  - Device drivers

Interfacing

- How do you give I/O device a command?
  - Memory-mapped load/store
    - Special addresses not for memory
    - Send commands as data
    - Cacheable?
  - I/O commands
    - Special opcodes
    - Send over I/O bus

Interfacing

- How do I/O devices communicate w/ CPU?
  - Poll on devices
    - Waste CPU cycles
    - Poll only when device active?
    - Not very popular in modern systems
  - Interrupts
    - Similar to exceptions, but asynchronous
    - Info in cause register
    - Possibly vectored interrupt handler

Interfacing

- Transfer data
  - Polling and interrupts – by CPU
  - OS transfers data

- Too many interrupts?
  - Use DMA so interrupt only when done
  - Use I/O channel – extra smart DMA engine
    - Offload I/O functions from CPU

Interfacing

- Caches and I/O
  - I/O in front of cache – slows CPU
  - I/O behind cache – cache coherence?
  - OS must invalidate/flush cache first before I/O

Summary – I/O

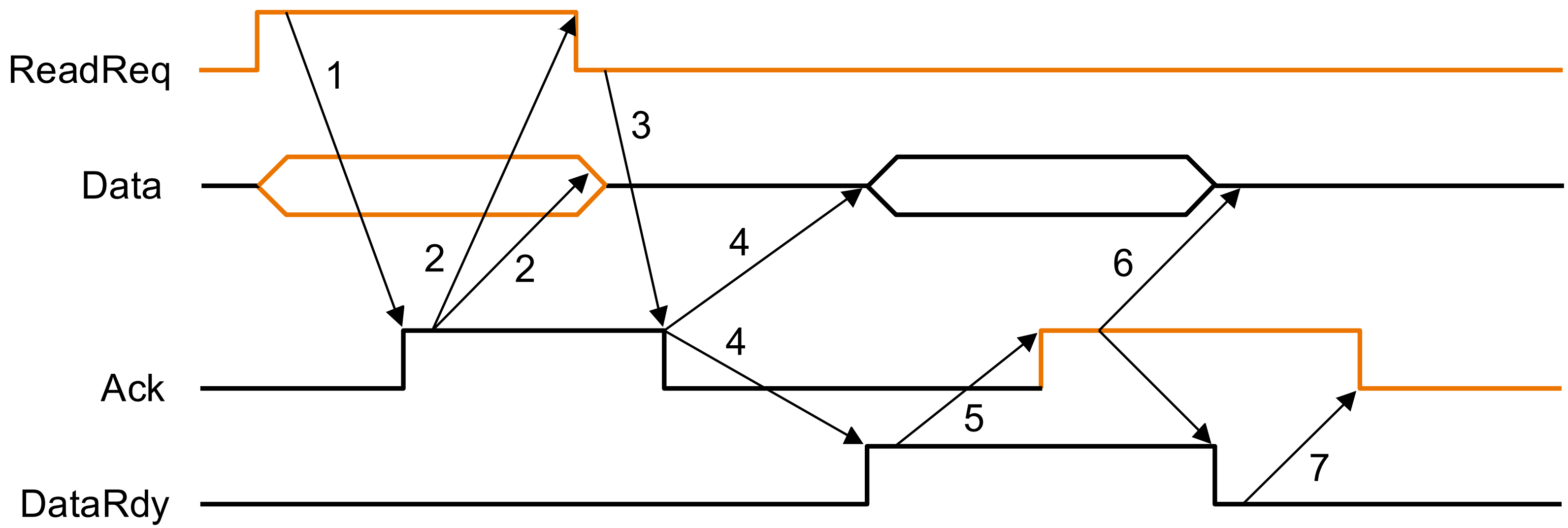
- I/O devices
  - Human interface – keyboard, mouse, display
  - Nonvolatile storage – hard drive, tape
  - Communication – LAN, modem
- Buses
  - Synchronous, asynchronous
  - Custom vs. standard
- Interfacing
  - Interrupts, DMA, cache coherence
  - O/S: protection, virtualization, multiprogramming

Backup

Buses

- Bus standards: ISA, PCI, PCI-X, AGP, …
- Currently PCIe 2.x
  - Serial, point-to-point topology
  - Bidirectional differential lanes (4 wires each)
  - 5GHz signaling rate per lane
  - 8b/10b encoding for DC balance, clock recovery
  - 5Gbit/sec x 10bit/byte = 500 MB/s per lane per direction
  - x1-x16 lanes per slot
- PCIe 3.0: 8GHz, 128/130b encoding

Async. Handshake Example

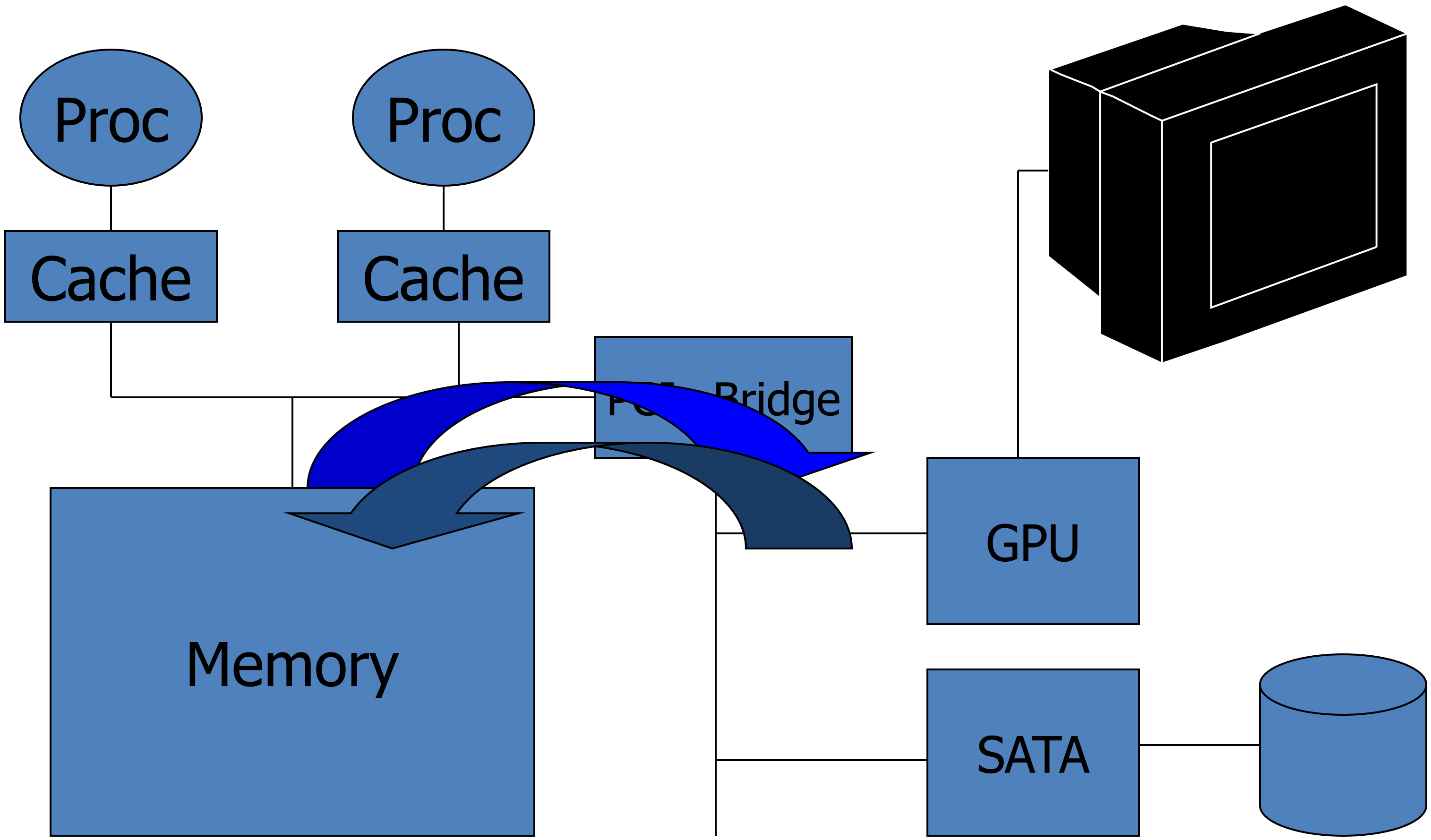


(1) Request made & (2) request send

(3) Request deasserted & (4) ack deasserted

(5) Data sent & (6) Data rec'd & (7) ack deasserted

Direct Memory Access (DMA)



DMA (cont'd)

- DMA
  - CPU sets up
    - Device ID, operation, memory address, # of bytes
  - DMA
    - Performs actual transfer (arb, buffers, etc.)
  - Interrupt CPU when done

- Typical I/O devices that use DMA
  - Hard drive, SSD, NIC, GPU

Interfacing Summary

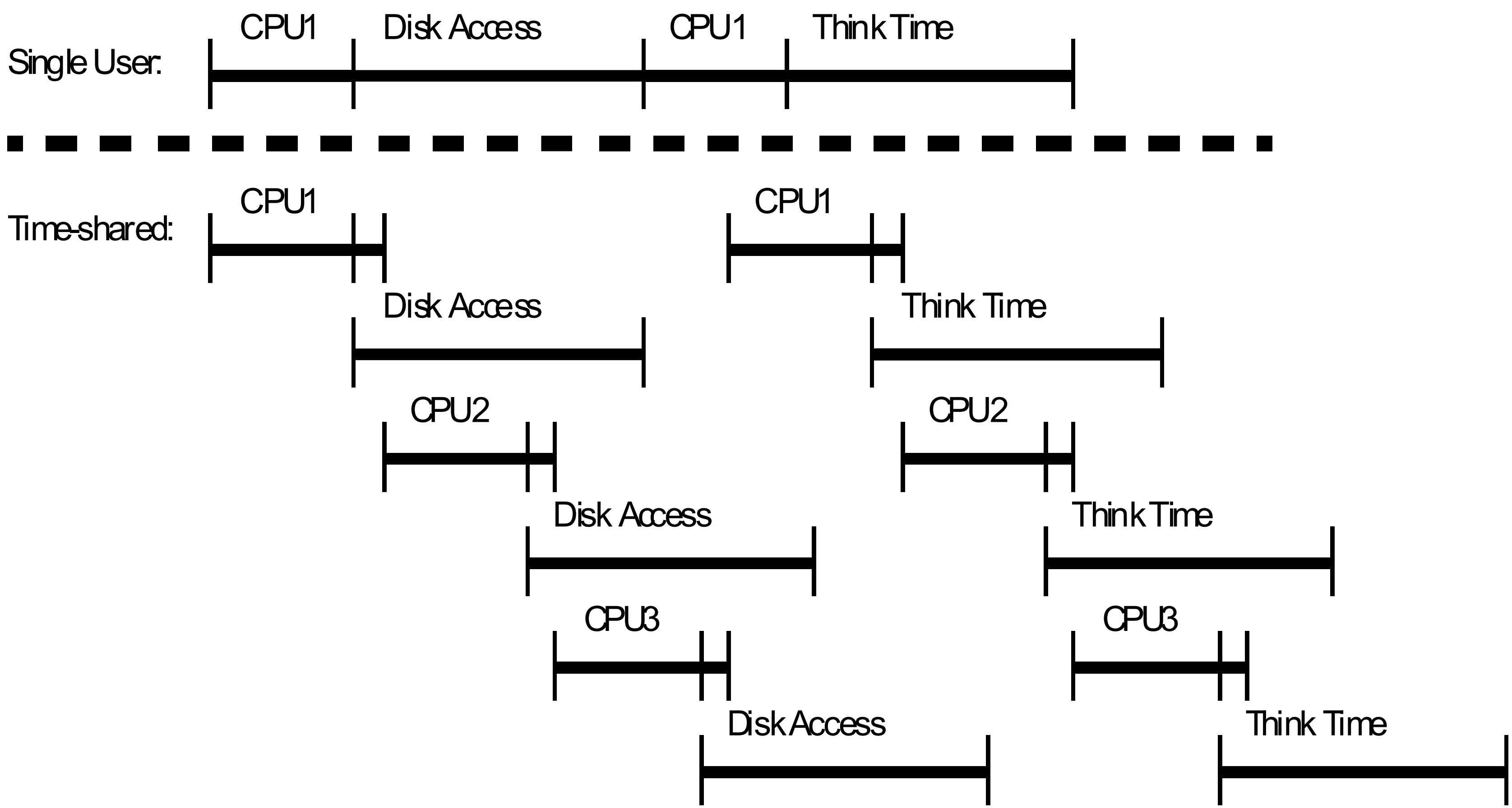
I/O Device Communication

- Control Flow Granularity
  - Coarse-grained (deep adapters, e.g. channels)
- Mechanics of Control Flow
  - Outbound Control Flow
    - Programmed I/O
  - Inbound Control Flow
    - Polling
- Mechanics of Data Flow
  - Programmed I/O
    - Software Cache Coherence

Software Interfacing

- I/O access provided by OS
  - Syscall interface between program and OS
  - OS checks protections, runs device drivers
  - Suspends current process, switches to other
  - I/O interrupt fielded by O/S
  - O/S completes I/O and makes process runnable
  - After interrupt, run next ready process
- Multiprogramming

Multiprogramming



I/O System Example