Lecture 28
DMA, IOP and RAID

Byung-gi Kim

School of Computer Science and Engineering

Soongsil University

6. Storage and Other I/O Topics

6.1 Introduction
6.2 Dependability, Reliability, and Availability
6.3 Disk Storage
6.4 Flash Storage
6.5 Connecting Processors, Memory, and I/O Devices
6.6 Interfacing I/O Devices to the Processor, Memory, and Operating System
6.7 I/O Performance Measures: Examples from Disk and File Systems
6.8 Designing an I/O System
6.9 Parallelism and I/O: RAID
6.10 Real Stuff: Sun Fire x4150 Server

Transferring the Data between a Device and Memory

1. Polling-based transfer (= programmed I/O)

- Periodical check of device status by CPU

```
while (not ready) get_status_of_the_device;
load_data_from_I/O_device;
store_the_data_into_memory;
```

- Best with lower-bandwidth devices
- More interested in reducing the cost of the device controller and interface than in providing a high-bandwidth transfer
- Put burden of moving data and managing the transfer on the processor

2. Interrupt Driven Transfer

- **Common characteristics with programmed I/O**
  - OS still transfers data in small number of bytes.
  - Best with lower-bandwidth devices
  - More interested in reducing the cost
- **Difference**
  - I/O device informs the processor when ready
  - Relieving the processor from having to wait for every I/O
- **I/O operation**
  - OS simply works on other tasks while data is being read from or written to the device.
  - On interrupts, OS reads the status to check for errors.
  - If none, OS transfers data.
  - When I/O completed, OS can inform the program.

3. DMA (Direct Memory Access)

- Device controller transfers data directly to or from the memory without involving the processor
- High-bandwidth devices like hard disks
- Interrupt mechanism is used only on completion of the I/O transfer or when an error occurs.
- **DMA controller**
  - Special controller that transfers data between an I/O device and memory independent of the processor
  - Becomes the bus master when transferring data

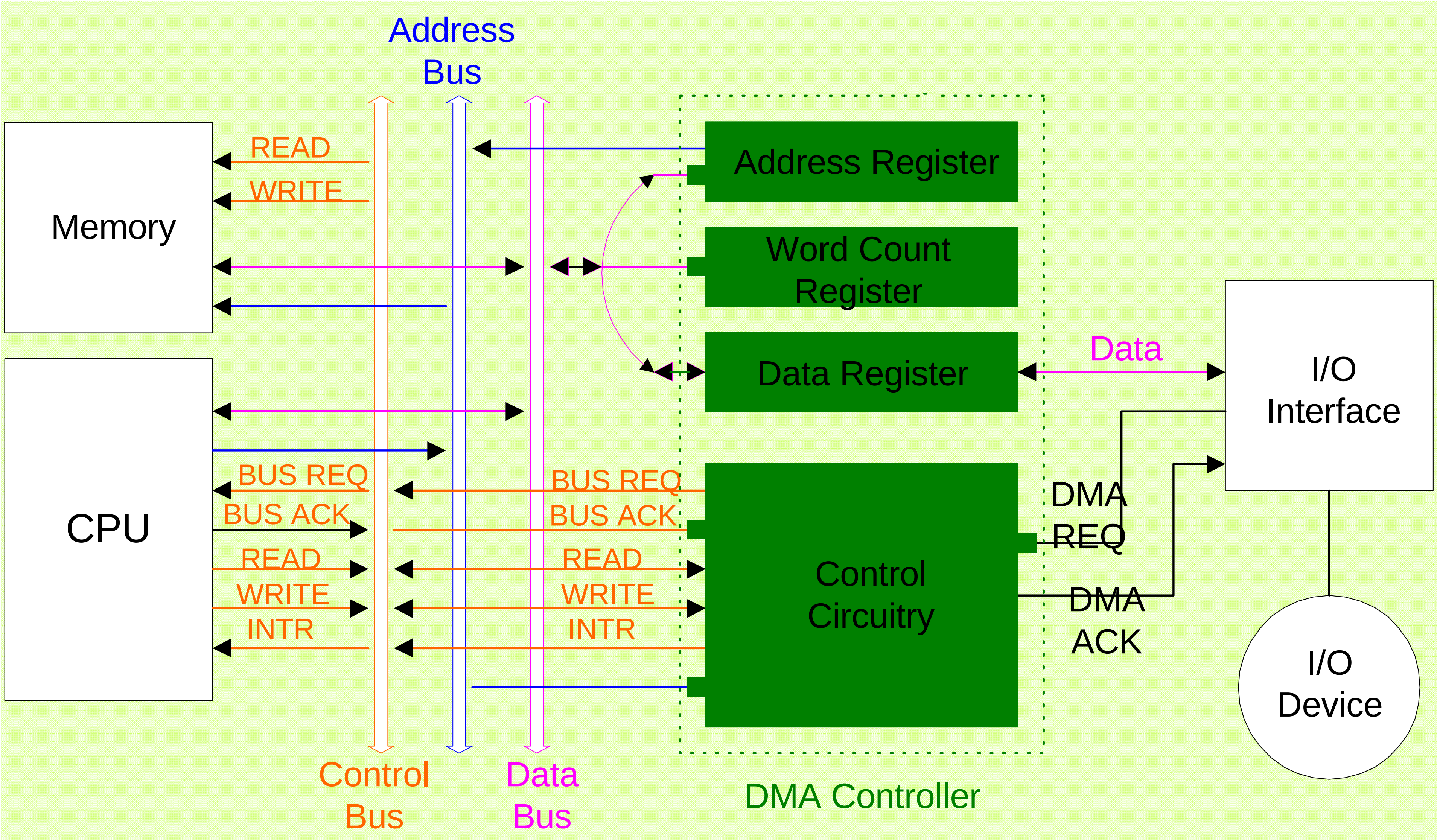
3 Steps in a DMA Transfer

1. **Initial setup of the DMA**
   - Device ID, operation, memory address, number of bytes
2. **DMA operation (cf) cycle stealing**

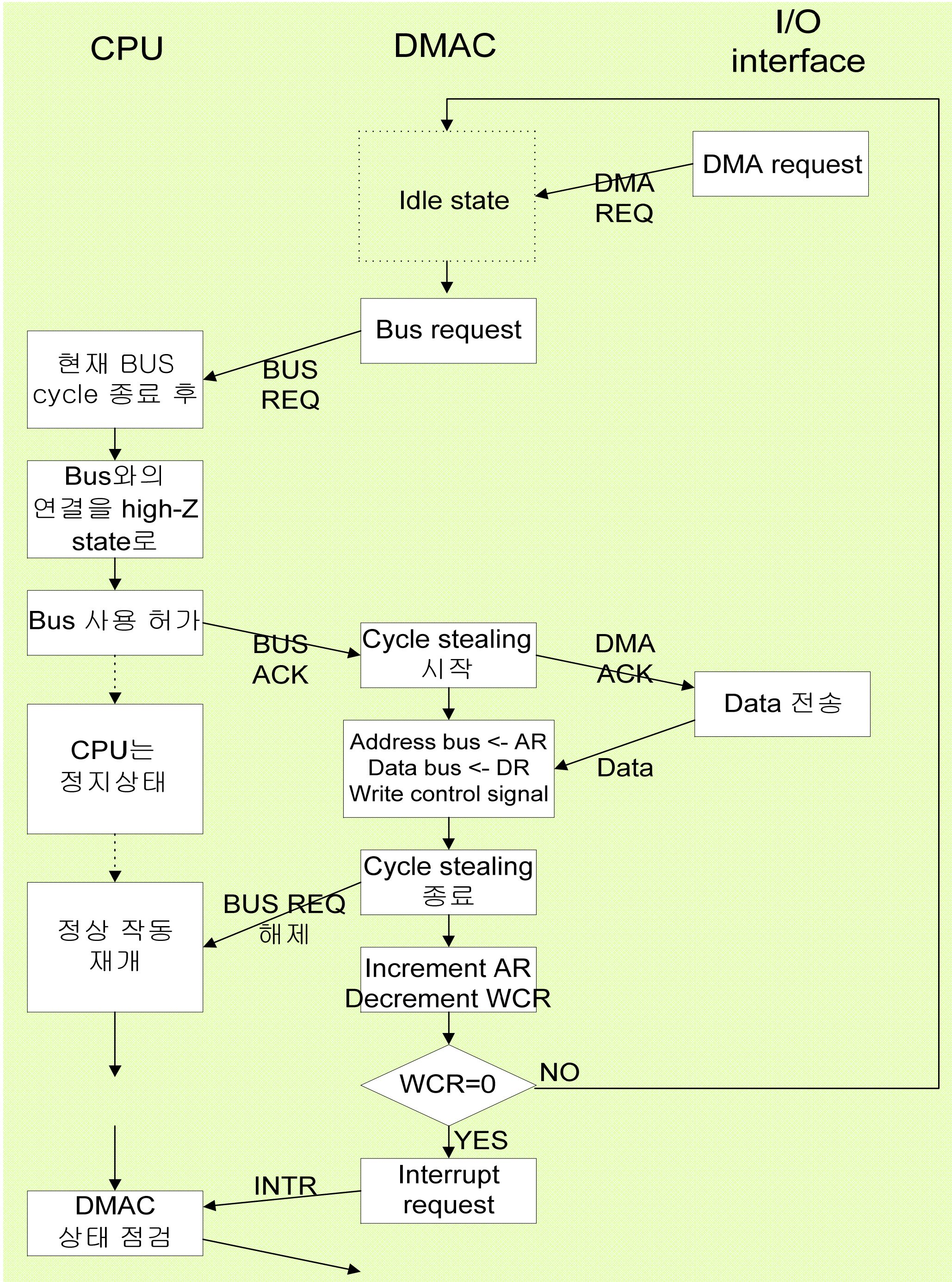
```
while (WCR>0) {acquire bus through arbitration;
               transfer data;
               WCR--; AR++;}
```

3. **Interrupt request**
   - When DMA transfer is complete
   - Processor checks whether the entire operation completed successfully.

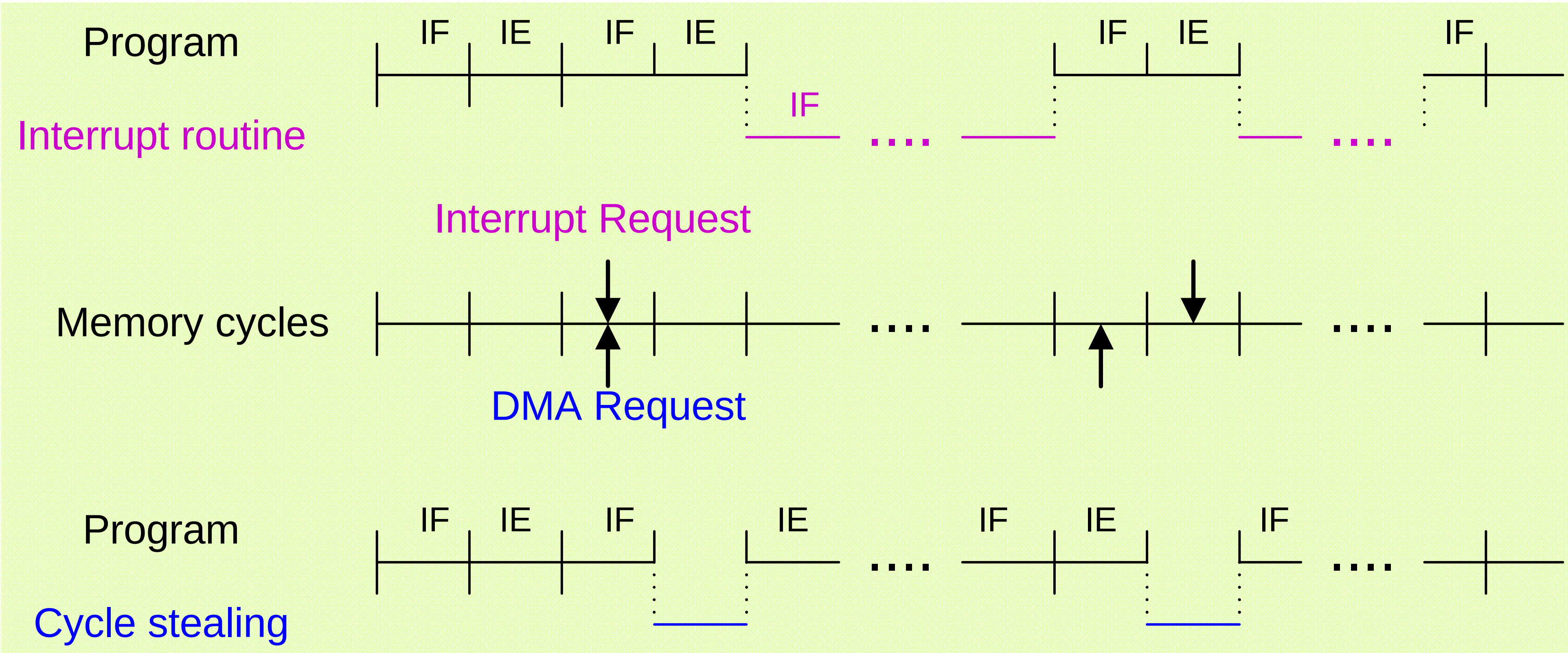
DMA Controller



DMA (Input)


CPU
DMAC
I/O interface
Idle state
DMA request
DMA REQ
Bus request
현재 BUS cycle 종료 후
BUS REQ
Bus와의 연결을 high-Z state로
Bus 사용 허가
BUS ACK
Cycle stealing 시작
DMA ACK
Data 전송
CPU는 정지상태
Address bus <- AR
Data bus <- DR
Write control signal
Data
Cycle stealing 종료
BUS REQ 해제
정상 작동 재개
Increment AR
Decrement WCR
WCR=0
NO
YES
Interrupt request
INTR
DMAC 상태 점검

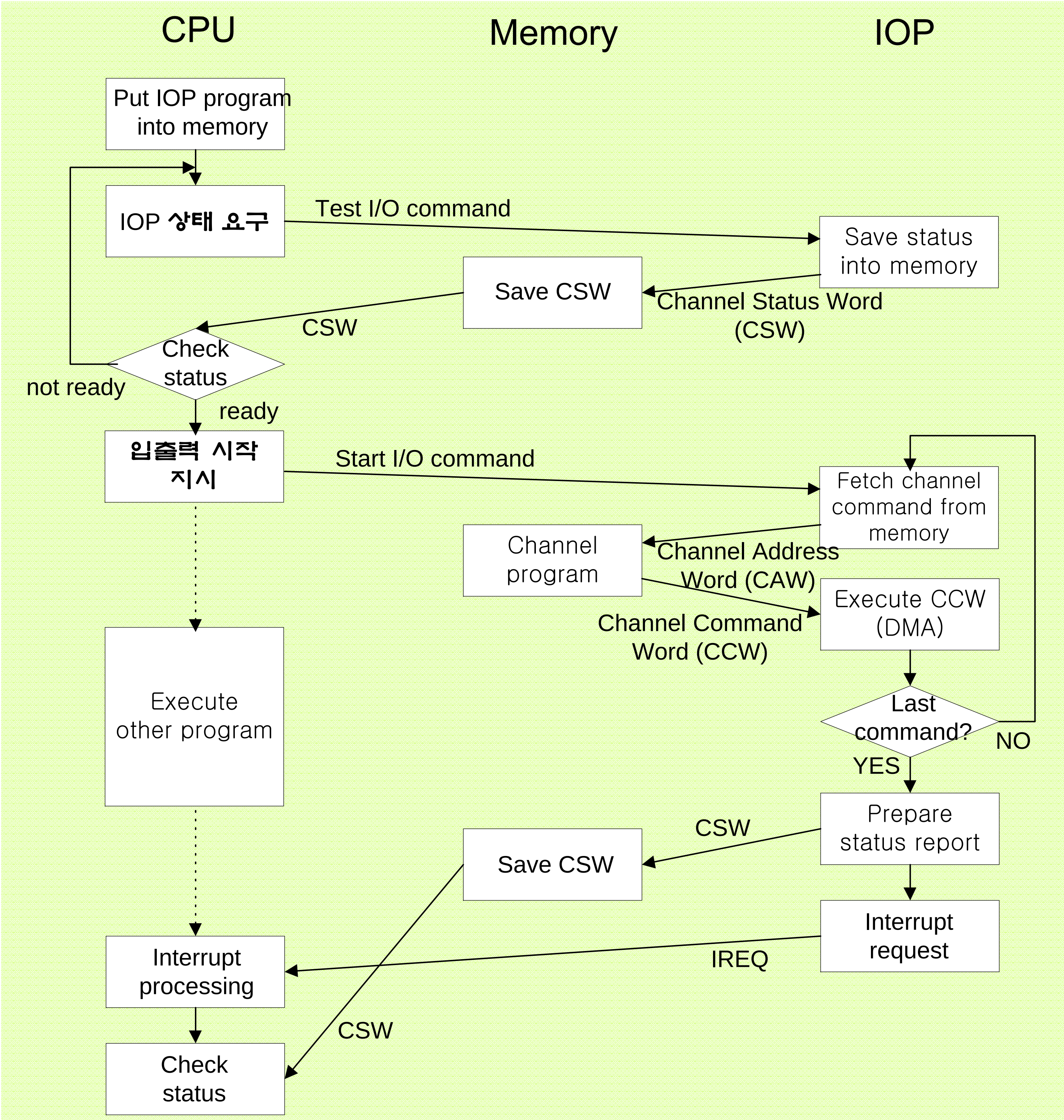
Interrupt vs. Cycle Stealing



Elaboration

- **I/O processor (IOP)**
  - I/O controller = channel controller = channel = I/O channel
  - More intelligent I/O controller
  - Reducing the need to interrupt the processor
- **I/O program**
  - Stored in the IOP or in main memory
  - I/O operations to be done as well as the size and transfer address for any reads or writes

IOP



CPU
Memory
IOP
Put IOP program into memory
IOP 상태 요구
Test I/O command
Save status into memory
Save CSW
Channel Status Word (CSW)
CSW
Check status
not ready
ready
입출력 시작 지시
Start I/O command
Fetch channel command from memory
Channel program
Channel Address Word (CAW)
Execute CCW (DMA)
Channel Command Word (CCW)
Last command?
NO
YES
Execute other program
Prepare status report
CSW
Save CSW
Interrupt request
IREQ
Interrupt processing
CSW
Check status

DMA Controller vs. I/O Processor

	DMAC	IOP
own instruction set and I/O program	no	yes
interrupt request	at the end of every block transfer	at the end of I/O program
implementation	special-purpose processor (usually single-chip and nonprogrammable)	general-purpose processor (general-purpose microprocessor, which runs a specialized I/O program)

Comparison of I/O Transfer Methods

	Polling	Interrupt-driven	DMA
I/O-to-memory transfer	through CPU	through CPU	direct
Interrupt	no	every byte or word	every block
Overhead	busy waiting	context switches	bus cycles
Data transfer	by instruction execution	by instruction execution	cycle stealing
Unit of transfer	byte or word	byte or word	block

6.9 Parallelism and I/O: RAIDs

Example: Impact of I/O on System Performance

- Benchmark
  - 90 sec. (CPU time) + 10 sec. (I/O time)
- Double the number of CPUs/2 years and I/O unchanged

[Answer]

Years	CPU time	I/O time	Elapsed time	% I/O time	Speedup
0	90 sec	10 sec	100 sec	10%	100/100 = 1.0
2	90/2 = 45 sec	10 sec	55 sec	18%	100/55 = 1.8
4	45/2 = 23 sec	10 sec	33 sec	31%	100/33 = 3.0
6	23/2 = 11 sec	10 sec	21 sec	47%	100/21 = 4.7

RAID

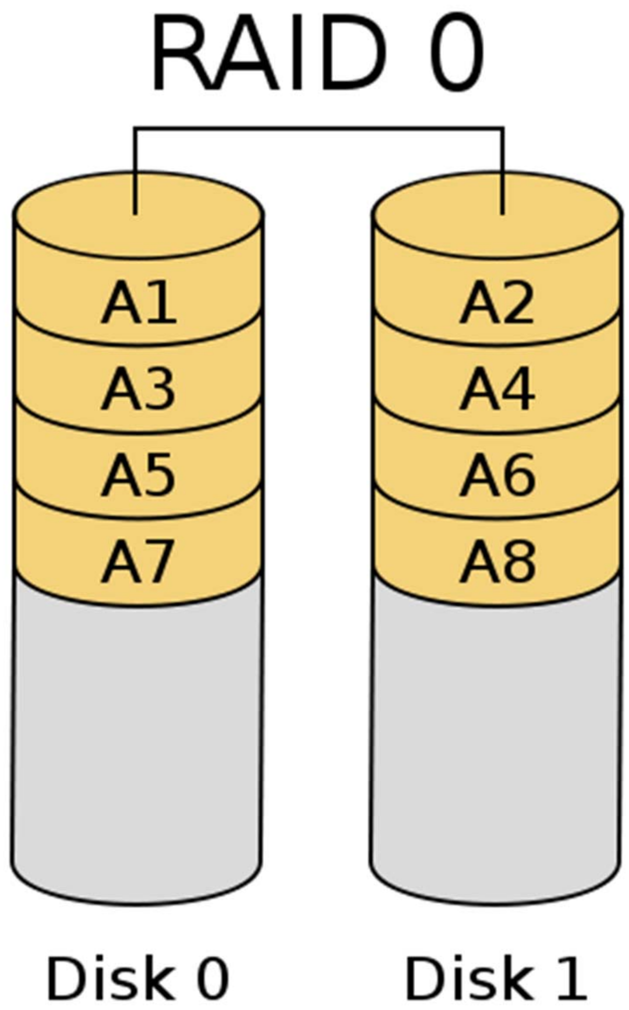
- Redundant Arrays of Inexpensive Disks
- Replacing a few large disks with many small disks
- Increase potential throughput by having many disk drives
  - Data is spread over multiple disk
  - Multiple accesses are made to several disks at a time
- Reliability is lower than a single disk
  - But availability can be improved by adding redundant disks
  - Lost information can be reconstructed from redundant information
  - Dependability is more affordable with RAID
- [ref] D. Patterson, G. Gibson and R. Katz, "A case for redundant arrays of inexpensive disks (RAID)," EECS/UCB Technical Report , UCB/CSD-87-391, Dec. 1997.

Berkeley RAID-1

- **RAID-I (1989)**
  - Consisted of a Sun 4/280 workstation with 128 MB of DRAM, four dual-string SCSI controllers, 28 5.25-inch SCSI disks and specialized disk striping software
- Today RAID is > 32 billion dollar industry, 80% nonPC disks sold in RAIDs (in 2009)

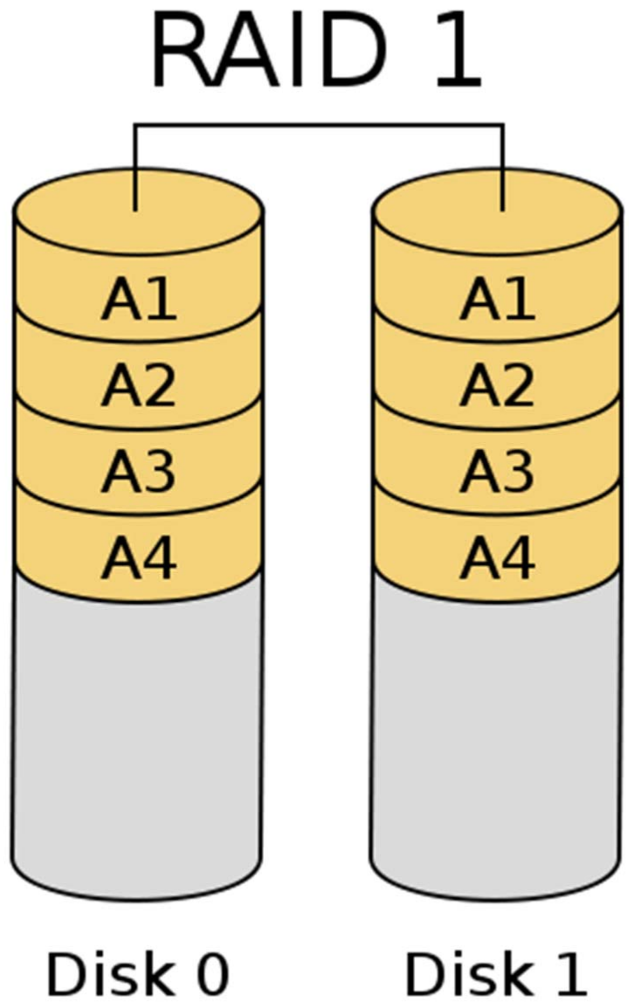
No Redundancy (RAID 0)

- No redundancy = "AID"
- Block-level striping without parity or mirroring
- Multiple blocks can be accessed in parallel increasing the performance
- No redundancy, so what if one disk fails?
  - Failure of one or more disks is more likely as the number of disks in the system increases



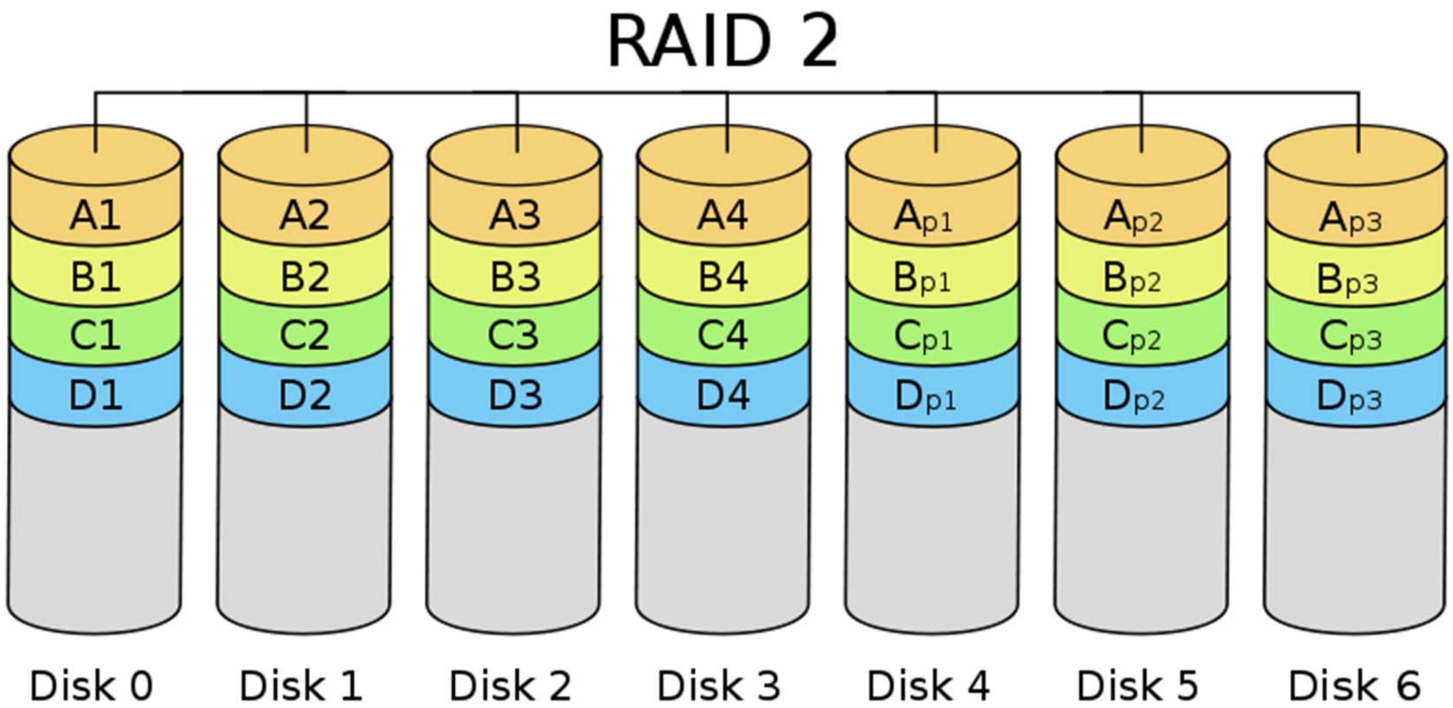
Mirroring (RAID 1)

- Each disk is fully duplicated onto its "mirror"
  - On disk failure, read from mirror
  - Very high availability can be achieved
- Bandwidth reduced on write:
  - Write data to both data disk and mirror disk
  - 1 Logical write = 2 physical writes
- Most expensive solution
  - 100% capacity overhead



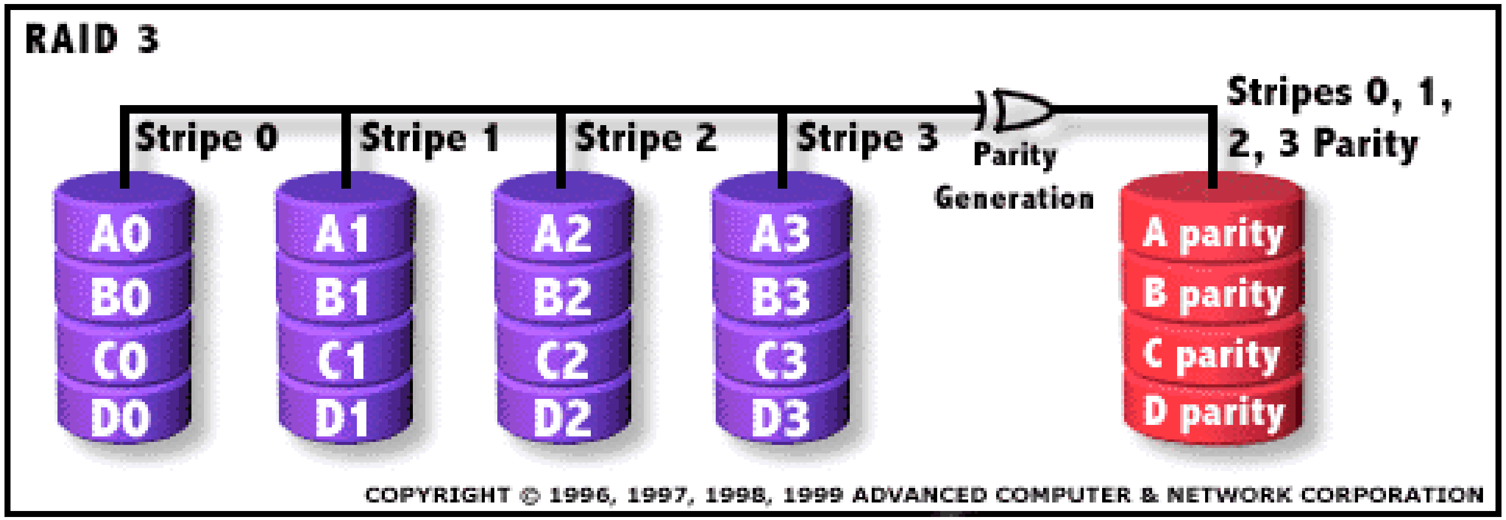
Error Detecting and Correcting Code (RAID 2)

- N + E disks (e.g., 10 + 4)
- Split data at bit level across N disks
- Generate E-bit ECC
- Too complex, not used in practice



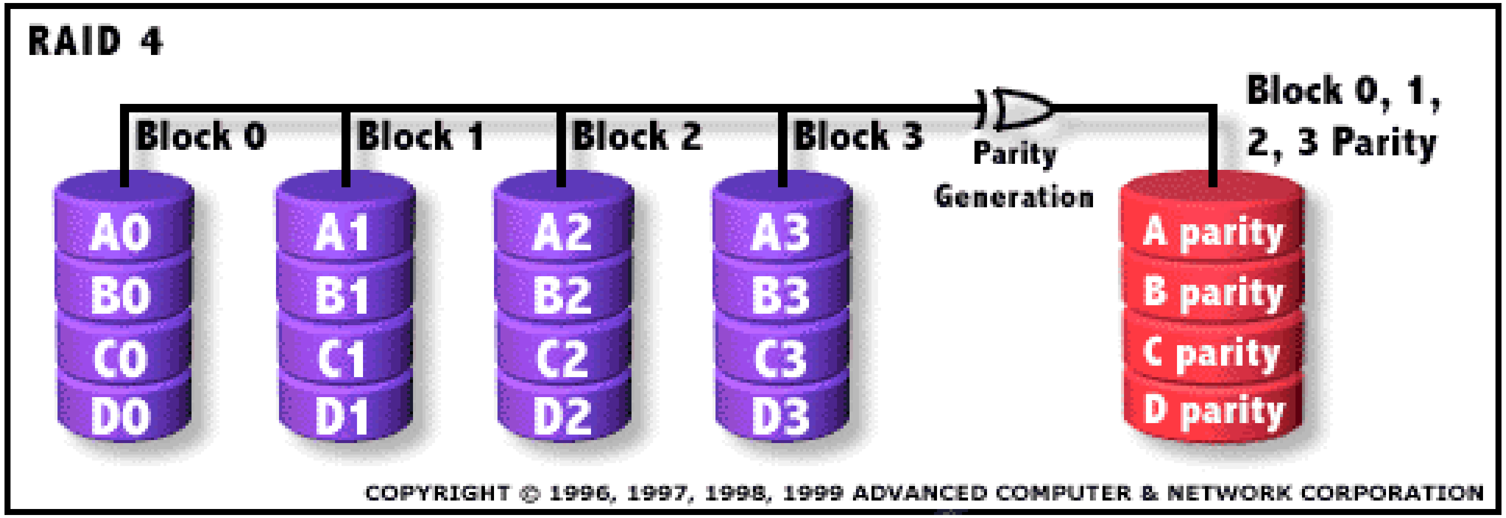
Bit-Interleaved Parity (RAID 3)

- Byte-level striping with dedicated parity
- All disk spindle rotation is synchronized
- Data is striped such that each sequential byte is on a different disk.
- Popular in applications with large data sets, such as multimedia and some scientific codes



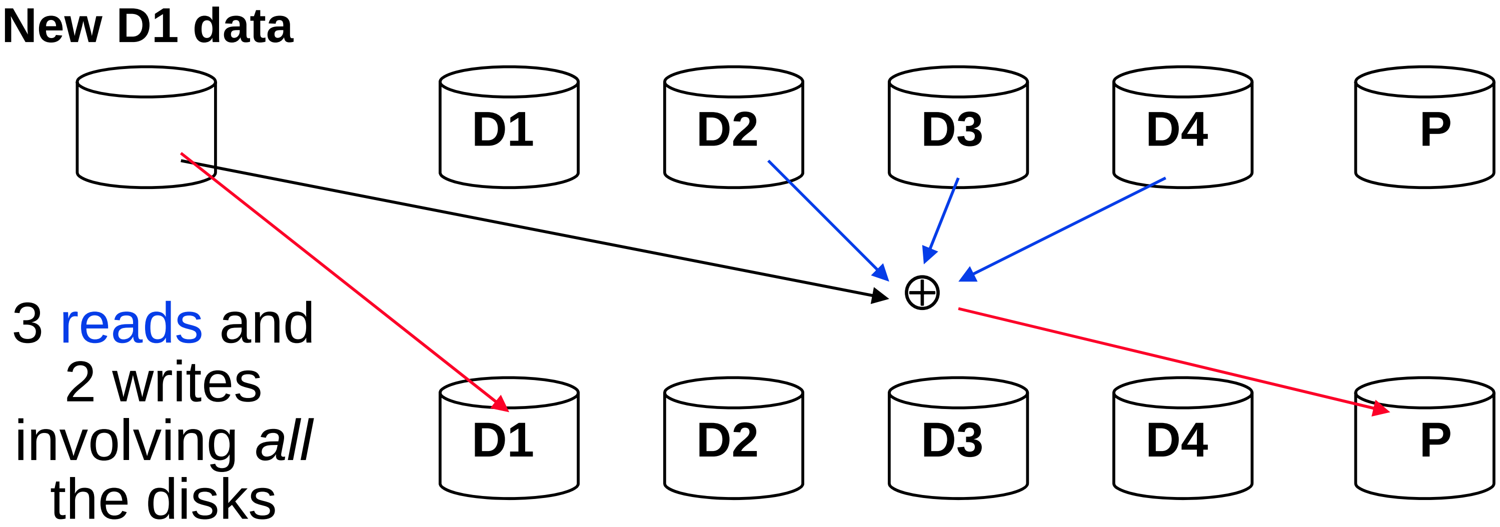
Block-Interleaved Parity (RAID 4)

- Block-level striping with dedicated parity
- In RAID 3, every access goes to all disks.
- Each entire block is written onto a disk.
- Rely on error detection field to catch errors on read, not on the parity disk
- Allows small independent reads to different disks simultaneously



Small Writes

- **RAID 3 writes**



- **RAID 4 small writes**

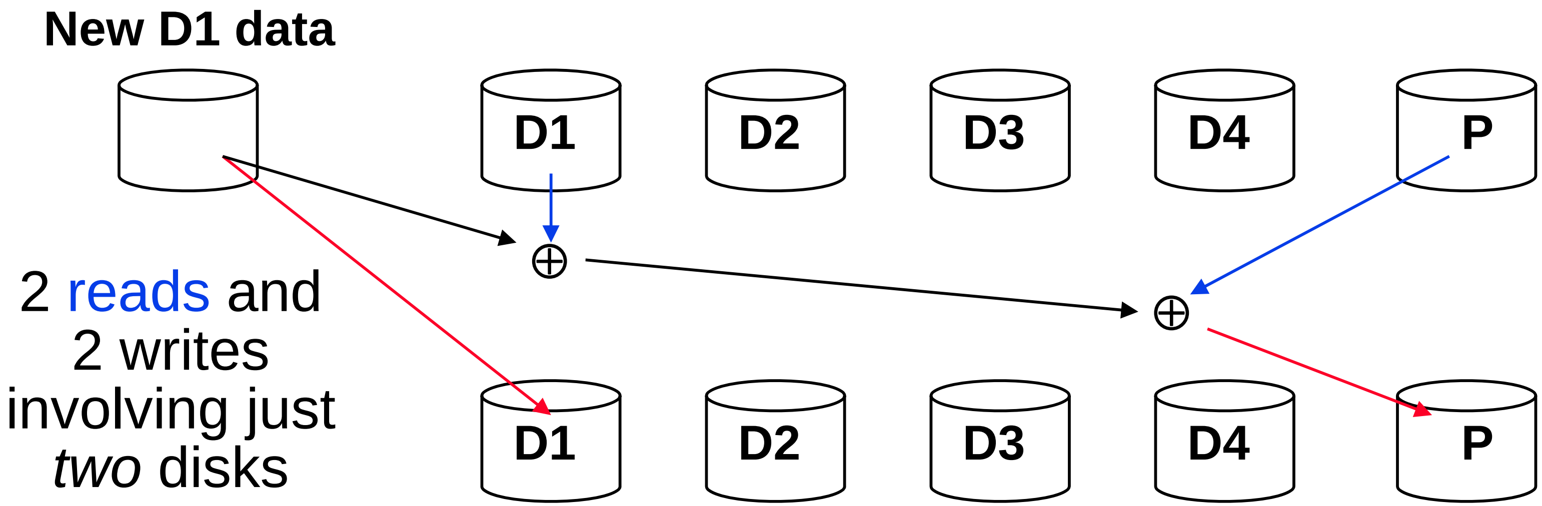


Figure 6.13

Distributed Block-Interleaved Parity (RAID 5)

- Block-level striping with distributed parity
- Like RAID 4, but parity blocks distributed across disks
  - Avoids parity disk being a bottleneck
- Widely used

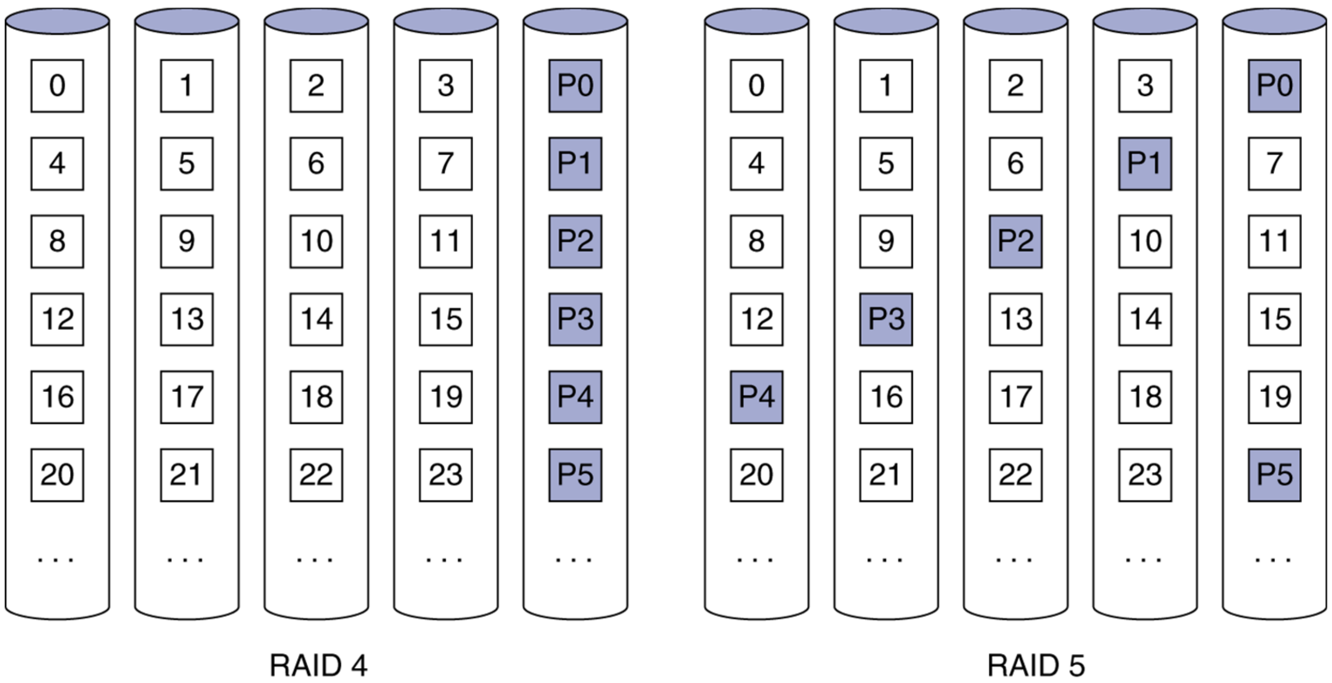
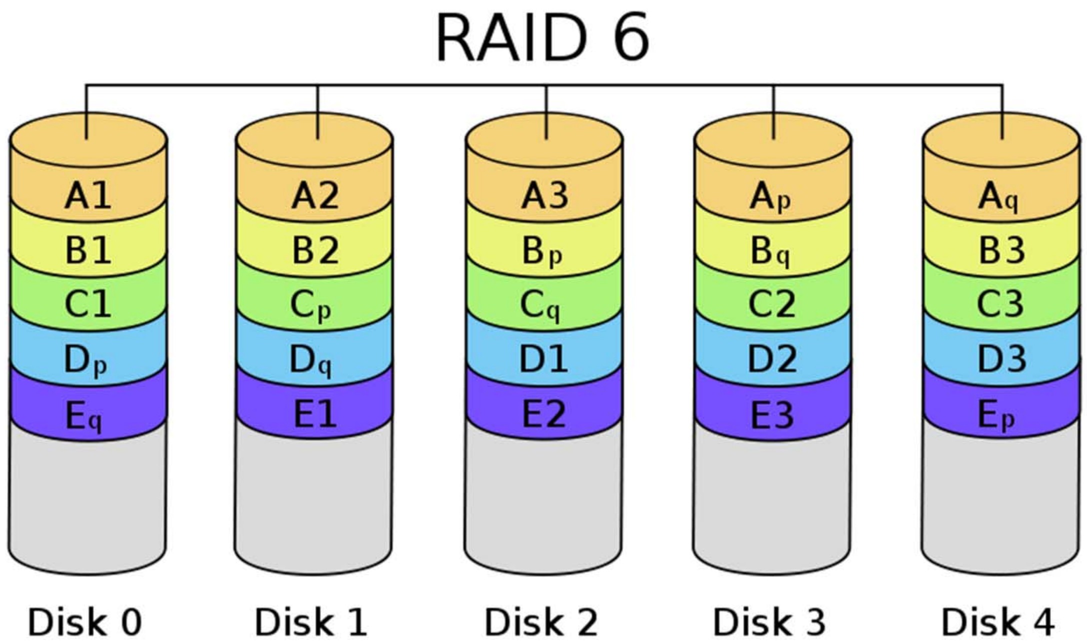


Figure 6.14

P + Q Redundancy (RAID 6)

- Block-level striping with double distributed parity
- Two independent parities
- Recovery from a second failure
- Storage overhead is twice that of RAID 5
- Six disks accesses for small write



RAID: Summary

- In case of 4 data disks

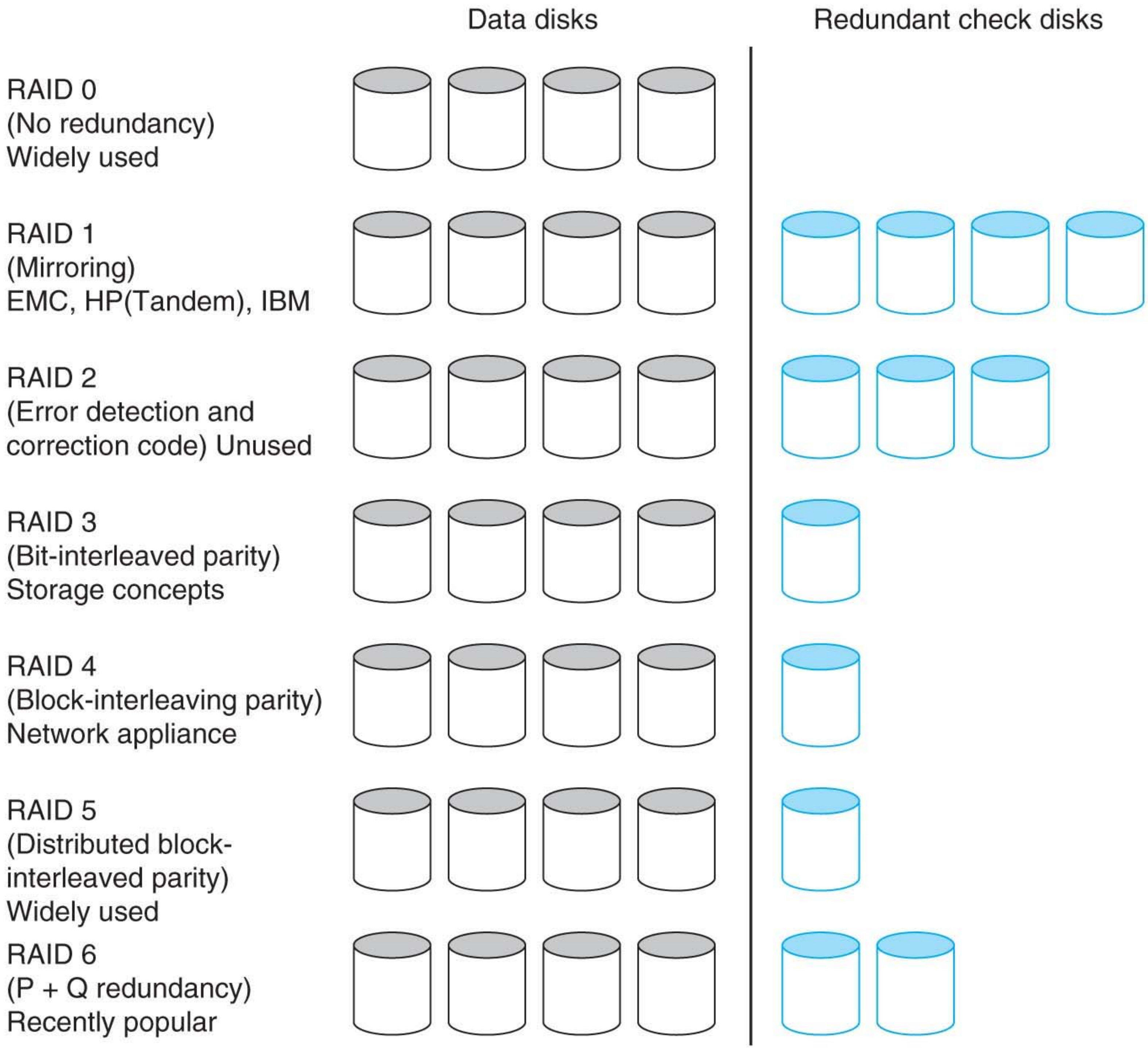


Figure 6.12

Supplement

DMA and the Memory System

- If DMA writes to a memory block that is cached
  - Cached copy becomes stale
- If write-back cache has dirty block, and DMA reads memory block
  - Reads stale data
- Solutions to ensure cache coherence
  1. Routing all I/O activity through the cache
     - Expensive and a large negative performance impact
  2. Cache flush
     - Having the OS invalidate all the entries in the cache for an I/O input or force write-backs for an I/O output
  3. Snooping cache controller
     - Providing hardware to *selectively* invalidate cache entries
  4. Non-cacheable memory locations for I/O

6.7 I/O Performance Measures: Examples from Disk and File Systems

Transaction Processing I/O Benchmarks

- **Characteristics of TP applications**
  - Both a response time requirement and a performance measurement based on throughput
  - Chiefly concerned with I/O rate (disk accesses/sec) than data rate (bytes/sec)
- **Transaction Processing Council (TPC) benchmarks**
  - TPC-APP: B2B application server and web services
  - TCP-C: on-line order entry environment
  - TCP-E: on-line transaction processing for brokerage firm
  - TPC-H: decision support — business oriented ad-hoc queries
  - ***http://www.tpc.org***

File System and Web I/O Benchmarks

- **SPECSFS (SPEC System File System)**
  - Synthetic workload for NFS server, based on monitoring real systems
  - Throughput-oriented benchmark but with important response time requirements
- **SPECWeb**
  - Simulating multiple clients requesting both static and dynamic pages from server, as well as clients posting data to the server
- **SPECPower**
  - Power and performance characteristics of small servers
- **filebench**
  - A file system benchmark framework from Sun
  - Instead of a standard workload, it provides a language that lets you describe the workload you'd like to run on your file systems

6.8 Designing an I/O System

- **Two primary types of specifications**
  1. Latency constraints
  2. Bandwidth constraints
- **General approach**
  1. Find the weakest link in the I/O system.
  2. Configure it to sustain required bandwidth.
     - Workload and configuration limit may dictate the weakest link.
  3. Determine the requirements for the rest of the system and configure them to support this bandwidth.
- **Example in Section 6.10**
  - Analysis of the I/O system of the Sun Fire x4150 server

6.10 Real Stuff: Sun Fire x4150 Server

- **19-inch rack**
  - 19 inches wide(482.6 mm)
- **Rack mount = subrack = shelf**
  - Computers designed for the rack
- **Rack unit = unit (U)**
  - 1.75 inches (44.45 mm)
  - The most popular 19-inch rack is 42 U high
    - 42 x 1.75 = 73.5 inches high
- **1U computer = 1U server = pizza box**
  - The smallest rack mount computer
  - 19 inches wide and 1.75 inches tall

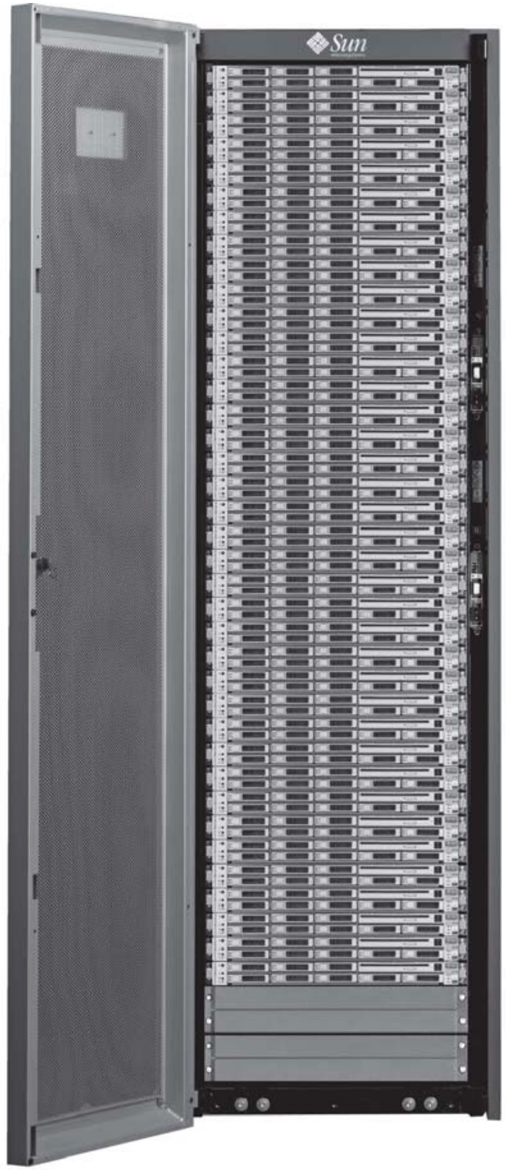


Figure 6.15

Back to chapter overview

Sun Fire x4150 1U Server

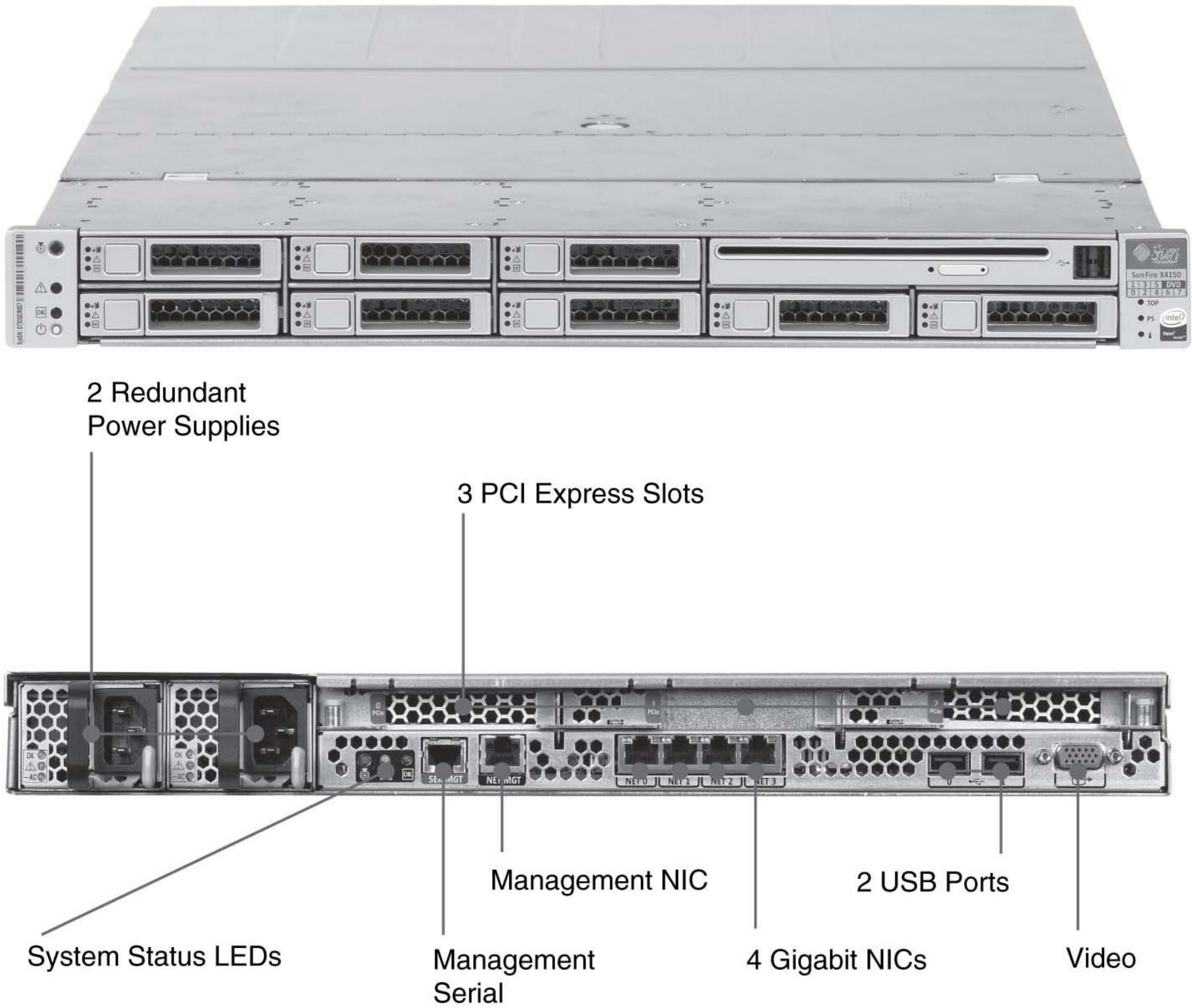


Figure 6.16

Sun Fire x4150

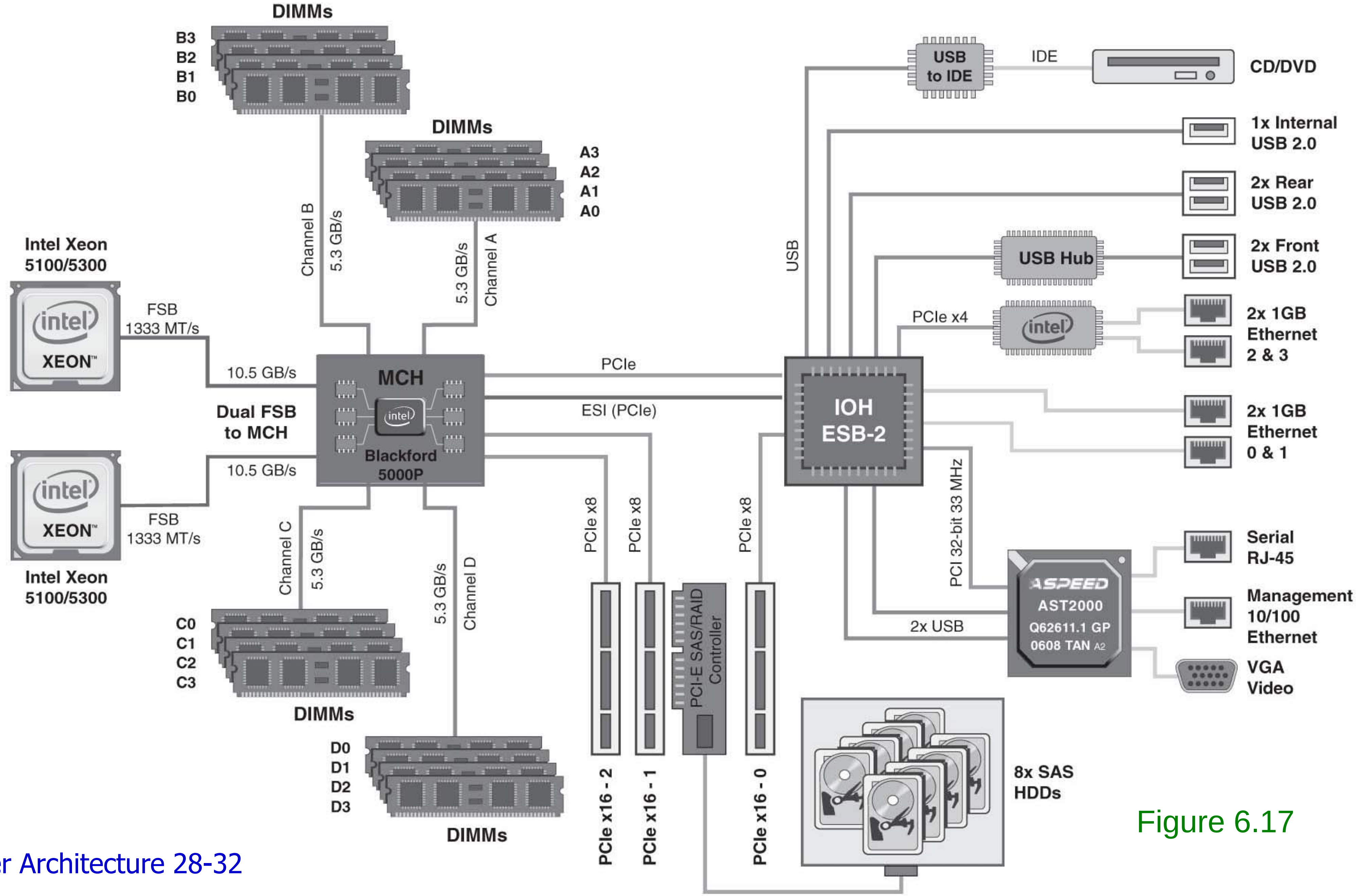


Figure 6.17

Example: I/O System Design

- Workload of 64 KB reads
  - User program: 200,000 instructions per I/O
  - OS: 100,000 instructions per I/O operation
- Sun Fire x4150
  - Each processor: 1 billion instructions per second
  - FSB: 10.6 GB/sec peak
  - DRAM DDR2 667MHz: 5.336 GB/sec
  - PCI Express x8 ports: 8 × 250MB/sec = 2GB/sec
  - 2.5″ SAS disks: 15,000 rpm, 2.9ms avg. seek time, 112MB/s transfer rate
- Ignore disk conflicts
- RAID controller is not the bottleneck
- **Find maximum sustainable I/O rate for a fully loaded Sun Fire x4150 for random reads and sequential reads.**

[Answer-1]

- **Maximum I/O rate of 1 processor =**

$$\frac{\text{Instruction execution rate}}{\text{Instructions per I/O}} = \frac{1 \times 10^9}{(200+100) \times 10^3} = 3.333 \frac{\text{I/Os}}{\text{second}}$$

  - I/O rate of 8 processors = 3.333 x 8 = 26,667 IOPS
- **Random reads (as disk I/O rate)**
  - Time per I/O at disk = Seek + rotational time + Transfer time

$$= \frac{2.9}{4}\text{ms} + 2.0\text{ms} + \frac{64\text{KB}}{112\text{MB/sec}} = 3.3\text{ms}$$

  - Each disk: 1 s/3.3 ms IOPS = 303 IOPS
  - 8 disks: 303 x 8 = 2,424 random reads per second
- **Sequential reads (need transfer time only)**
  - Each disk: 112MB/s / 64KB = 1750 IOPS
  - 8 disks: 1750 x 8 = 14,000 sequential reads per second

[Answer-2]

- **Max I/O rate of PCI Express x8 =**

$$\frac{\text{PCI bandwidth}}{\text{Bytes per I/O}} = \frac{2 \times 10^9}{64 \times 10^3} = 31{,}250 \text{ IOPS}$$

- **DRAM I/O rate =**

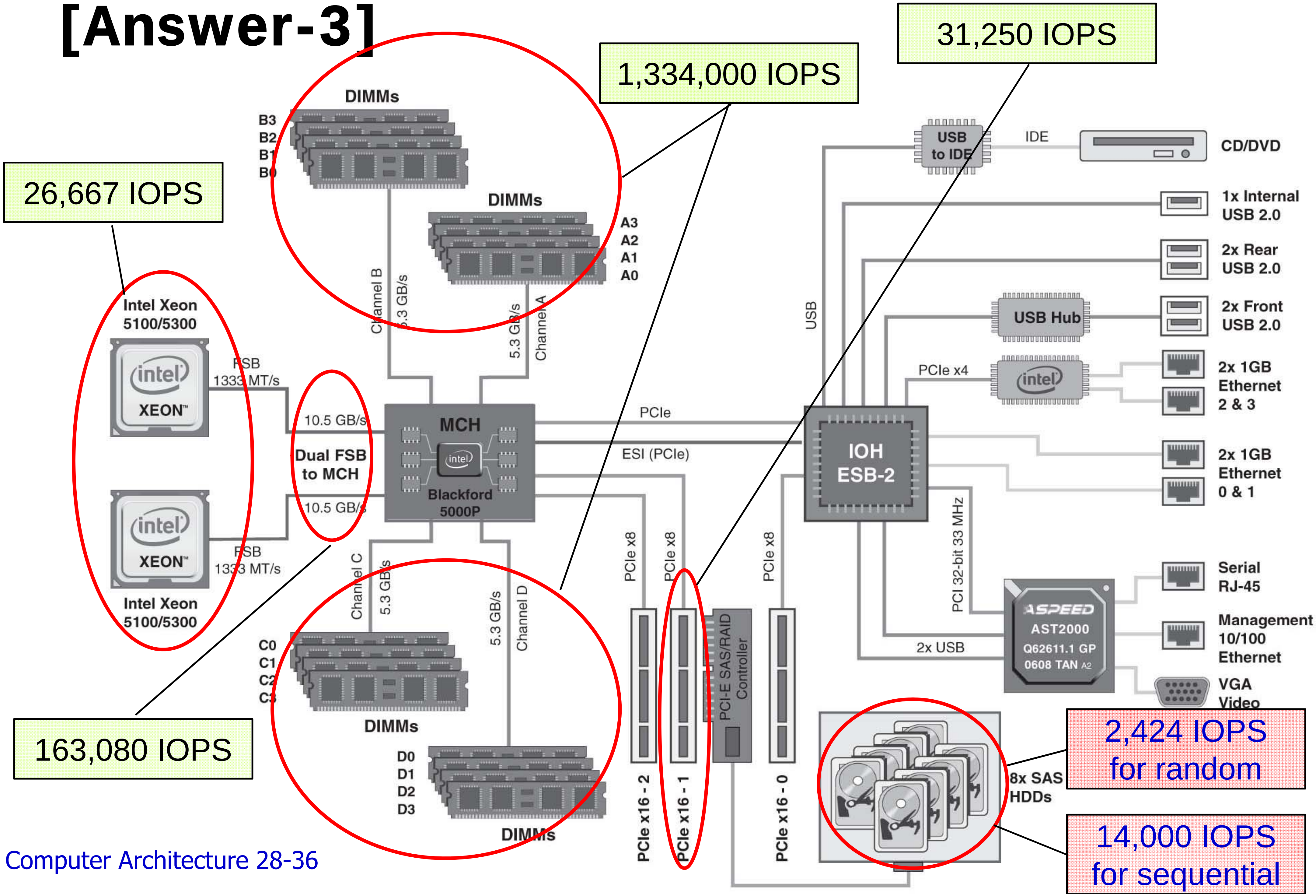
  Bandwidth of a DIMM = 667 MHz x 2 x 4 bytes = 5,336 MB/sec

  5,336 MB/sec / 64KB = 83,375 IOPS

  16 DIMMs => 83,375 x 16 = 1,334,000 IOPS

- **FSB I/O rate**
  - Assume we can sustain half the peak rate = 10.5 GB/s x 0.5 ≈ 5.3 GB/s
  - 5.3 GB/s / 64KB = 81,540 IOPS per FSB
  - 163,080 IOPS for 2 FSBs
- **Weakest link => disks**
  - 2,424 IOPS random
  - 14,000 IOPS sequential

[Answer-3]


26,667 IOPS
1,334,000 IOPS
31,250 IOPS
163,080 IOPS
2,424 IOPS for random
14,000 IOPS for sequential
DIMMs
B3
B2
B1
B0
A3
A2
A1
A0
Channel B
5.3 GB/s
Channel A
Intel Xeon 5100/5300
intel XEON
FSB 1333 MT/s
10.5 GB/s
Dual FSB to MCH
MCH
Blackford 5000P
PCIe
ESI (PCIe)
Channel C
Channel D
C0
C1
C2
C3
D0
D1
D2
D3
PCIe x8
PCIe x16 - 2
PCIe x16 - 1
PCIe x16 - 0
PCI-E SAS/RAID Controller
IOH ESB-2
USB
USB to IDE
IDE
CD/DVD
1x Internal USB 2.0
2x Rear USB 2.0
USB Hub
2x Front USB 2.0
PCIe x4
2x 1GB Ethernet 2 & 3
2x 1GB Ethernet 0 & 1
PCI 32-bit 33 MHz
ASPEED AST2000 Q62611.1 GP 0608 TAN A2
2x USB
Serial RJ-45
Management 10/100 Ethernet
VGA Video
8x SAS HDDs

Idle and Peak Power of Sun Fire x4150

	Components			System			
Item	Idle	Peak	Number	Idle		Peak	
Single Intel 2.66 GHz E5345 socket, Intel 5000 MCB/IOH chip set, Ethernet controllers, power supplies, fans, . . .	154 W	215 W	1	154 W	37%	215 W	39%
Additional Intel 2.66 GHz E5345 socket	22 W	79 W	1	22 W	5%	79 W	14%
4 GB DDR2-667 5300 FBDIMM	10 W	11 W	16	160 W	39%	176 W	32%
73 GB SAS 15K Disk drives	8 W	8 W	8	64 W	15%	64 W	12%
PCIe x8 RAID Disk controller	15 W	15 W	1	15 W	4%	15 W	3%
Total	—	—	—	415 W	100%	549 W	100%

Figure 6.18

6.12 Fallacies and Pitfalls

[Fallacy] *The rated mean time to failure of disks is 1,200,000 hours or almost 140 years, so disks practically never fail.*

[Fallacy] *Disk failure rates in the field match their specifications.*

[Fallacy] *A GB/sec interconnect can transfer 1 GB of data in 1 second.*

[Pitfall] *Trying to provide features only within the network versus end to end.*

[Pitfall] *Moving functions from the CPU to the I/O processor, expecting to improve performance without a careful analysis.*

[Pitfall] *Using magnetic tapes to back up disks.*

[Fallacy] *Operating systems are the best place to schedule disk accesses.*

[Pitfall] *Using the peak transfer rate of a portion of the I/O system to make performance projections or performance comparisons.*

6.13 Concluding Remarks

- I/O systems are evaluated on several different characteristics.
  - Dependability
  - Variety of I/O devices supported
  - Maximum number of I/O devices
  - Cost
  - Performance
    - Measured both in latency and in throughput
- Storage and networking demands are growing at unprecedented rates [Lyman and Varian, 2003]
  - Amount of information created in 2002 was 5 exabytes (5×10^{18} bytes)
    - Equivalent to 500,000 copies of the text in the U.S. Library of Congress
  - Total amount of information in the world was doubling every 3 years