2009-07-02

2009年 前学期 TOKYO TECH

計算機アーキテクチャ 第一 (E)

9. 磁気ディスク, RAID

吉瀬 謙二 計算工学専攻
kise_at_cs.titech.ac.jp
W641講義室 木曜日13:20 – 14:50

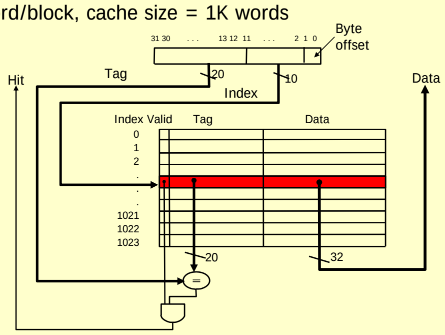
Acknowledgement

- Lecture slides for Computer Organization and Design, Third Edition, courtesy of **Professor Mary Jane Irwin**, Penn State University
- Lecture slides for Computer Organization and Design, third edition, Chapters 1-9, courtesy of **Professor Tod Amon,** Southern Utah University.

Adapted from *Computer Organization and Design*, Patterson & Hennessy, © 2005

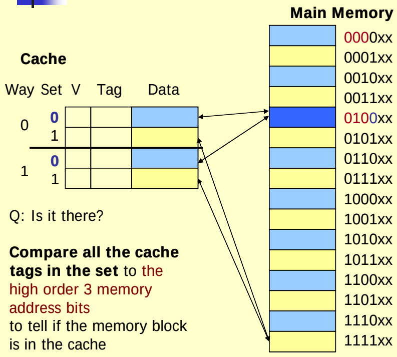
MIPS Direct Mapped Cache Example

- One word/block, cache size = 1K words



What kind of locality are we taking advantage of?

Set Associative Cache Example



Two low order bits define the byte in the word (32-b words)
One word blocks

Q: Is it there?

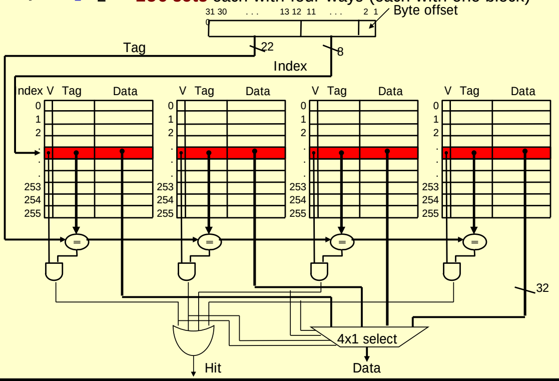
Compare all the cache tags in the set to the high order 3 memory address bits to tell if the memory block is in the cache

Q: How do we find it?

Use next 1 low order memory address bit to determine which cache set

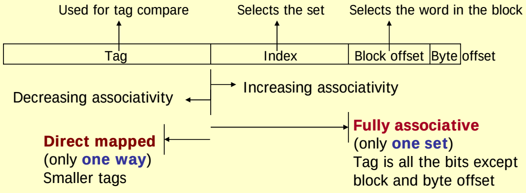
Four-Way Set Associative Cache

- 2^8 = **256 sets** each with four ways (each with one block)

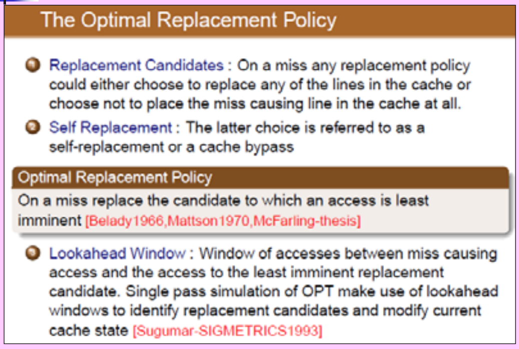


Range of Set Associative Caches

- For a fixed size cache



OPT: Optimal Replacement Policy



The Optimal Replacement Policy

1. Replacement Candidates : On a miss any replacement policy could either choose to replace any of the lines in the cache or choose not to place the miss causing line in the cache at all.
2. Self Replacement : The latter choice is referred to as a self-replacement or a cache bypass

Optimal Replacement Policy

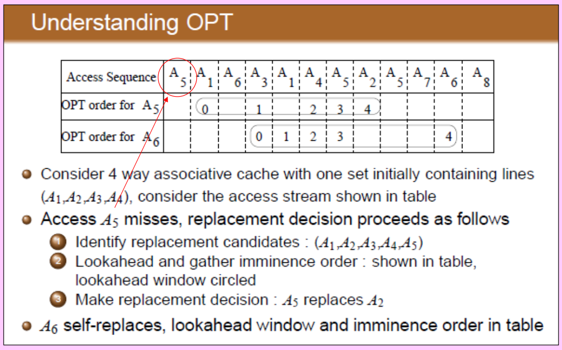
On a miss replace the candidate to which an access is least imminent [Belady1966,Mattson1970,McFarling-thesis]

3. Lookahead Window : Window of accesses between miss causing access and the access to the least imminent replacement candidate. Single pass simulation of OPT make use of lookahead windows to identify replacement candidates and modify current cache state [Sugumar-SIGMETRICS1993]

OPT: あまり切迫していないものを置き換える.

MICRO-40 Emulating Optimal Replacement with a Shepherd Cache

Optimal Replacement Policy の例



Understanding OPT

Access Sequence	A_5	A_1	A_6	A_3	A_1	A_4	A_5	A_2	A_5	A_7	A_6	A_8
OPT order for A_5		0		1		2	3	4				
OPT order for A_6				0	1	2	3				4	

- Consider 4 way associative cache with one set initially containing lines (A_1,A_2,A_3,A_4), consider the access stream shown in table
- Access A_5 misses, replacement decision proceeds as follows
  1. Identify replacement candidates : (A_1,A_2,A_3,A_4,A_5)
  2. Lookahead and gather imminence order : shown in table, lookahead window circled
  3. Make replacement decision : A_5 replaces A_2
- A_6 self-replaces, lookahead window and imminence order in table

MICRO-40 Emulating Optimal Replacement with a Shepherd Cache

Sources of Cache Misses

- **Compulsory** (**初期参照ミス**, cold start or process migration, first reference):
  - First access to a block, “cold” fact of life, not a whole lot you can do about it
  - If you are going to run “millions” of instruction, compulsory misses are insignificant
- **Conflict** (**競合性ミス**, collision):
  - Multiple memory locations mapped to the same cache location
  - Solution 1: increase cache size
  - Solution 2: increase **associativity**
- **Capacity** (**容量性ミス**):
  - Cache cannot contain all blocks accessed by the program
  - Solution: increase cache size

レポート 問題

1. SimMipsにデータキャッシュのヒット率を測定する仕組みを追加し，ヒット率を測定せよ．
   1. ***ダイレクトマップ方式***，**ラインサイズは４ワード**とする．
   2. セット数を８，１６，３２，６４，１２８，**２５６**，５１２に変更した場合のヒット率を示せ．
   3. 以前作成した sort（1000要素のランダムデータ）を含む３つのアプリケーションを作成し，そのヒット率を示すこと．
2. キャッシュのヒット率を改善する方式を実装し，その効果を示せ．
   1. 例えば，ラインサイズの変更
   2. 例えば，セットアソシアティブ方式
   3. 例えば，フルアソシアティブ方式
3. レポートはA4用紙3枚以内にまとめること．(必ずPDFとすること)（2段組，コードは小さい文字でもかまわない．）

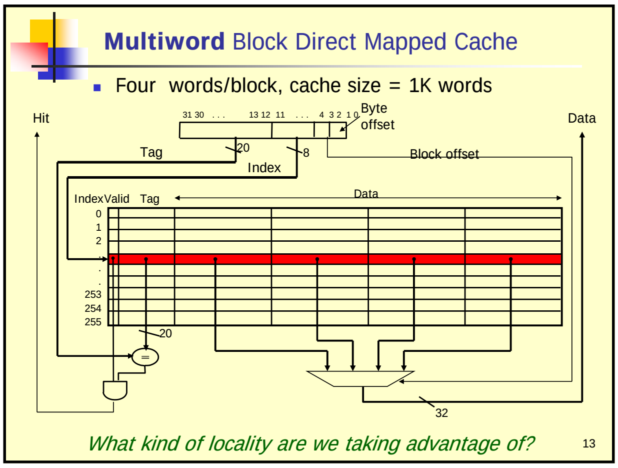
講義用の計算機環境

- 講義用の計算機
  - 131.112.16.56
  - ssh arche@131.112.16.56
    - **ユーザ名: arche**
    - **パスワードは講義時に連絡**
  - cd myname　　(例: cd 06B77777)
  - cp –r /home/arche/v0.5.5 .
  - cd v0.5.5
  - memory.cc などを修正してコンパイル，実行
- 注意点
  - 計算機演習室からは外部にsshで接続できないかもしれません．
  - Windowsからは Tera Term などを利用してください．

Adapted from *Computer Organization and Design*, Patterson & Hennessy, © 2005

レポート 提出方法

- 7月4日（午後7時）までに電子メールで提出
  - 今回は先願性は考慮しません．
  - report_at_arch.cs.titech.ac.jp
- 電子メールのタイトル
  - Arch Report [学籍番号]
  - 例：Arch Report [33_77777]
- 電子メールの内容
  - 氏名，学籍番号
  - 回答
    - PDFファイルを添付（必ずPDFとすること）
    - PDFファイルにも氏名，学籍番号を記入すること．
    - A4用紙で3枚以内にまとめること．



Multiword Block Direct Mapped Cache

- Four words/block, cache size = 1K words

What kind of locality are we taking advantage of?

13

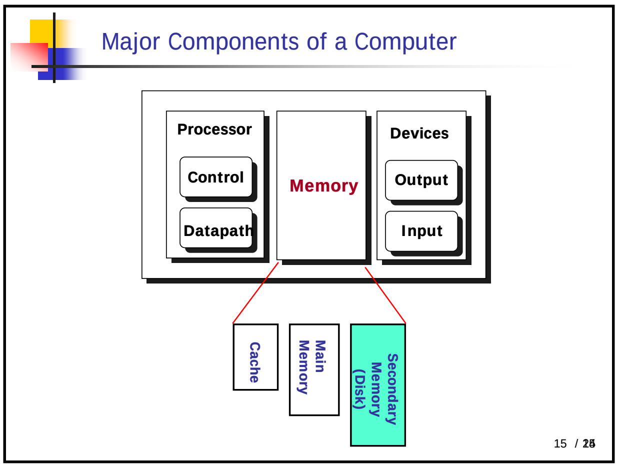
2009-07-02

2009年 前学期 TOKYO TECH

計算機アーキテクチャ 第一 (E)

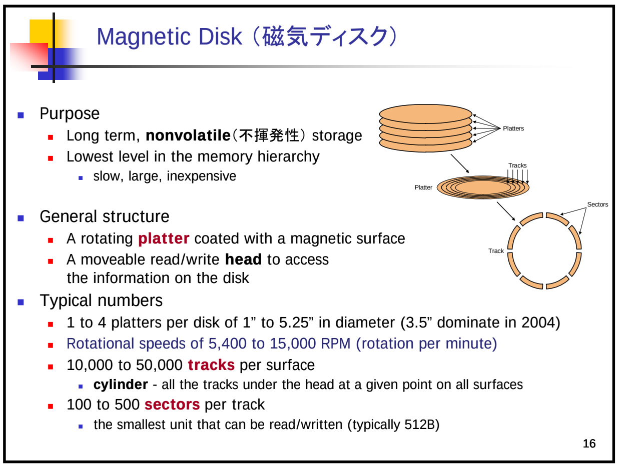
9. 磁気ディスク, RAID

吉瀬 謙二 計算工学専攻
kise_at_cs.titech.ac.jp
W641講義室 木曜日13:20 ー 14:50



Major Components of a Computer

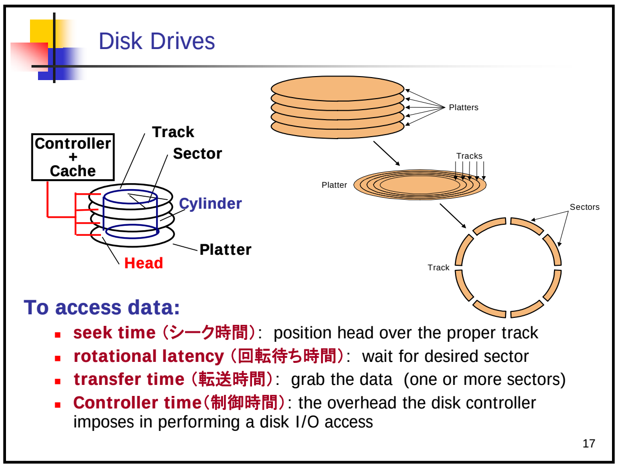
15 / 26



Magnetic Disk（磁気ディスク）

- Purpose
  - Long term, **nonvolatile**（不揮発性）storage
  - Lowest level in the memory hierarchy
    - slow, large, inexpensive
- General structure
  - A rotating **platter** coated with a magnetic surface
  - A moveable read/write **head** to access the information on the disk
- Typical numbers
  - 1 to 4 platters per disk of 1” to 5.25” in diameter (3.5” dominate in 2004)
  - Rotational speeds of 5,400 to 15,000 RPM (rotation per minute)
  - 10,000 to 50,000 **tracks** per surface
    - **cylinder** - all the tracks under the head at a given point on all surfaces
  - 100 to 500 **sectors** per track
    - the smallest unit that can be read/written (typically 512B)

16

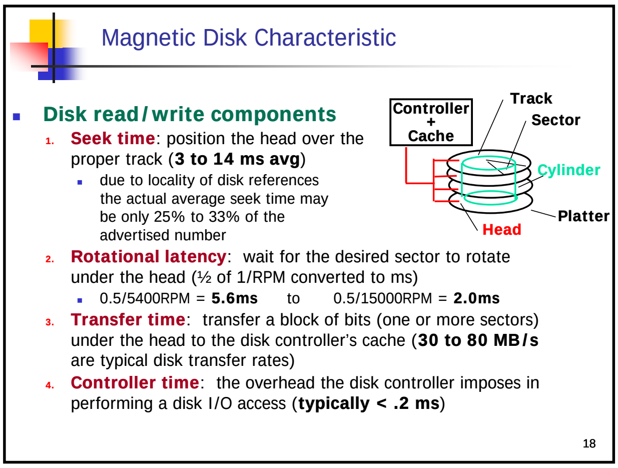


Disk Drives

To access data:

- **seek time（シーク時間）**: position head over the proper track
- **rotational latency（回転待ち時間）**: wait for desired sector
- **transfer time（転送時間）**: grab the data (one or more sectors)
- **Controller time（制御時間）**: the overhead the disk controller imposes in performing a disk I/O access

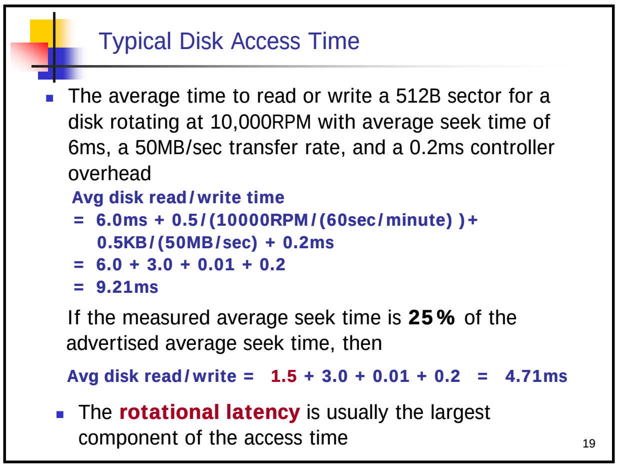
17



Magnetic Disk Characteristic

- **Disk read / write components**
  1. **Seek time**: position the head over the proper track (**3 to 14 ms avg**)
     - due to locality of disk references the actual average seek time may be only 25% to 33% of the advertised number
  2. **Rotational latency**: wait for the desired sector to rotate under the head (½ of 1/RPM converted to ms)
     - 0.5/5400RPM = **5.6ms** to 0.5/15000RPM = **2.0ms**
  3. **Transfer time**: transfer a block of bits (one or more sectors) under the head to the disk controller’s cache (**30 to 80 MB/s** are typical disk transfer rates)
  4. **Controller time**: the overhead the disk controller imposes in performing a disk I/O access (**typically < .2 ms**)

18



Typical Disk Access Time

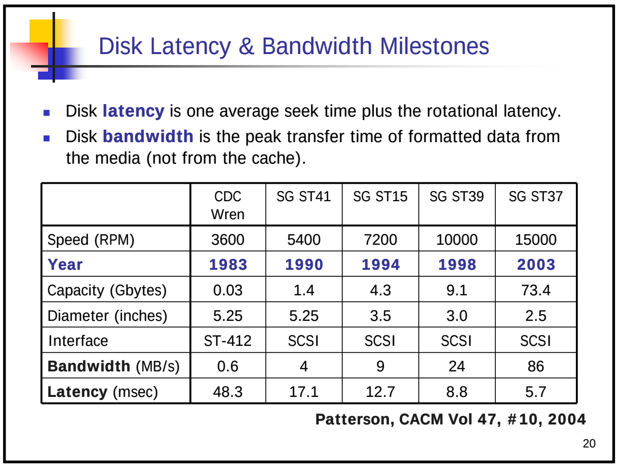
- The average time to read or write a 512B sector for a disk rotating at 10,000RPM with average seek time of 6ms, a 50MB/sec transfer rate, and a 0.2ms controller overhead

Avg disk read/write time
= 6.0ms + 0.5/(10000RPM/(60sec/minute))+ 0.5KB/(50MB/sec) + 0.2ms
= 6.0 + 3.0 + 0.01 + 0.2
= 9.21ms

If the measured average seek time is **25%** of the advertised average seek time, then

Avg disk read/write = 1.5 + 3.0 + 0.01 + 0.2 = 4.71ms

- The **rotational latency** is usually the largest component of the access time

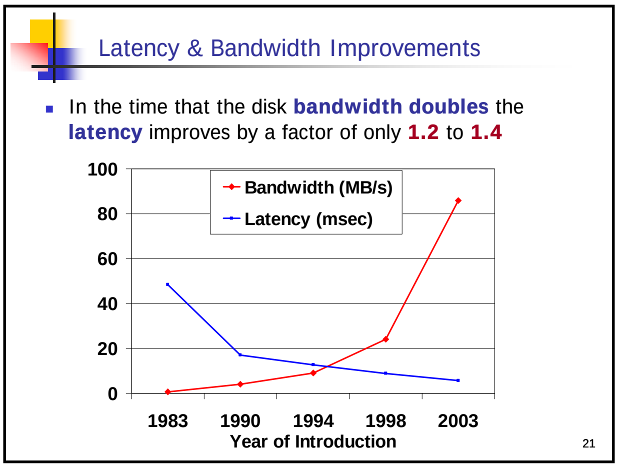


Disk Latency & Bandwidth Milestones

- Disk **latency** is one average seek time plus the rotational latency.
- Disk **bandwidth** is the peak transfer time of formatted data from the media (not from the cache).

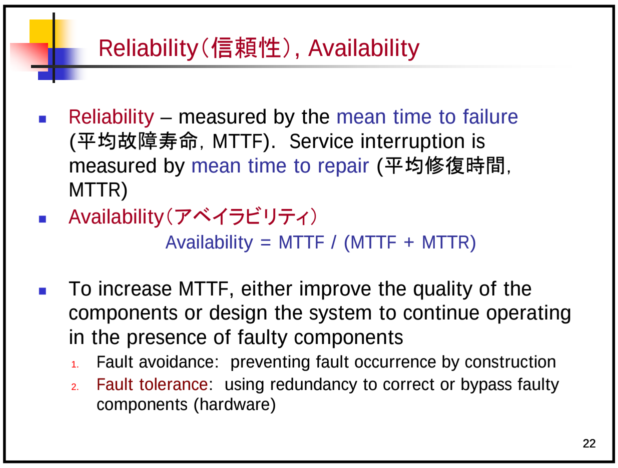
	CDC Wren	SG ST41	SG ST15	SG ST39	SG ST37
Speed (RPM)	3600	5400	7200	10000	15000
Year	**1983**	**1990**	**1994**	**1998**	**2003**
Capacity (Gbytes)	0.03	1.4	4.3	9.1	73.4
Diameter (inches)	5.25	5.25	3.5	3.0	2.5
Interface	ST-412	SCSI	SCSI	SCSI	SCSI
Bandwidth (MB/s)	0.6	4	9	24	86
Latency (msec)	48.3	17.1	12.7	8.8	5.7

Patterson, CACM Vol 47, #10, 2004



Latency & Bandwidth Improvements

- In the time that the disk **bandwidth doubles** the **latency** improves by a factor of only **1.2** to **1.4**

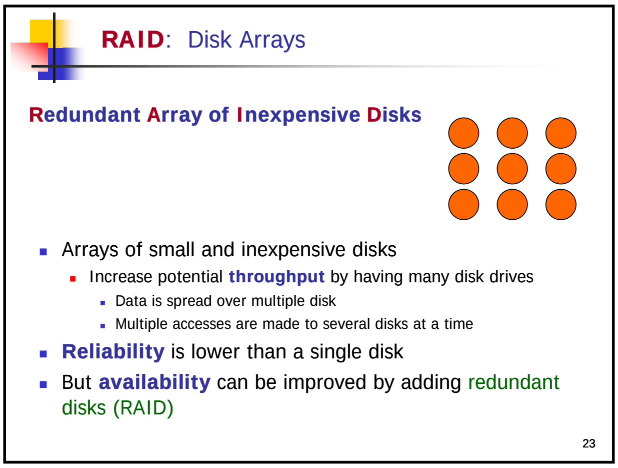


Reliability（信頼性）, Availability

- Reliability – measured by the mean time to failure （平均故障寿命, MTTF). Service interruption is measured by mean time to repair （平均修復時間, MTTR)
- Availability（アベイラビリティ）

  Availability = MTTF / (MTTF + MTTR)

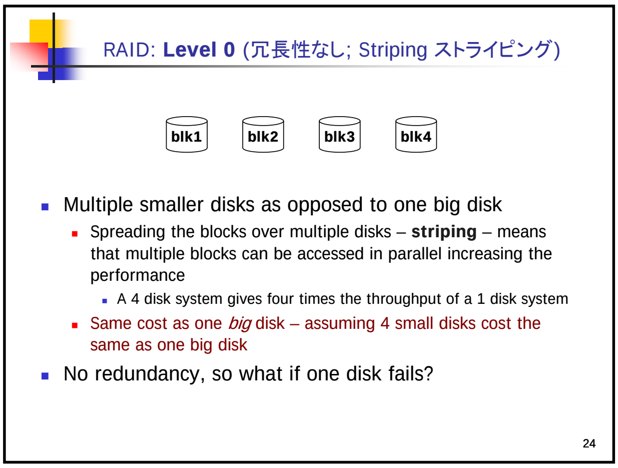
- To increase MTTF, either improve the quality of the components or design the system to continue operating in the presence of faulty components
  1. Fault avoidance: preventing fault occurrence by construction
  2. Fault tolerance: using redundancy to correct or bypass faulty components (hardware)



RAID: Disk Arrays

Redundant Array of Inexpensive Disks

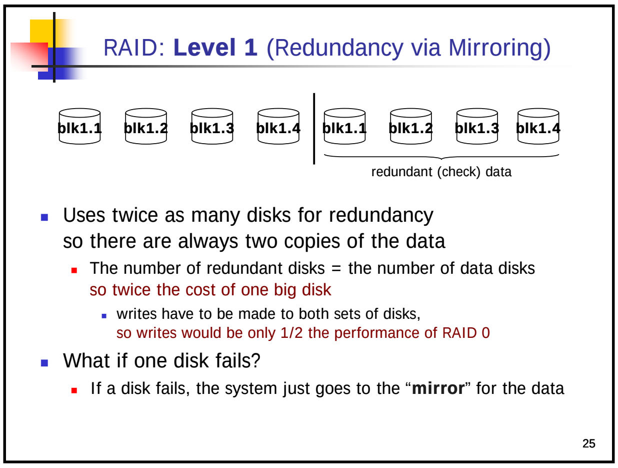
- Arrays of small and inexpensive disks
  - Increase potential **throughput** by having many disk drives
    - Data is spread over multiple disk
    - Multiple accesses are made to several disks at a time
- **Reliability** is lower than a single disk
- But **availability** can be improved by adding redundant disks (RAID)



RAID: **Level 0** (冗長性なし; Striping ストライピング)

blk1
blk2
blk3
blk4

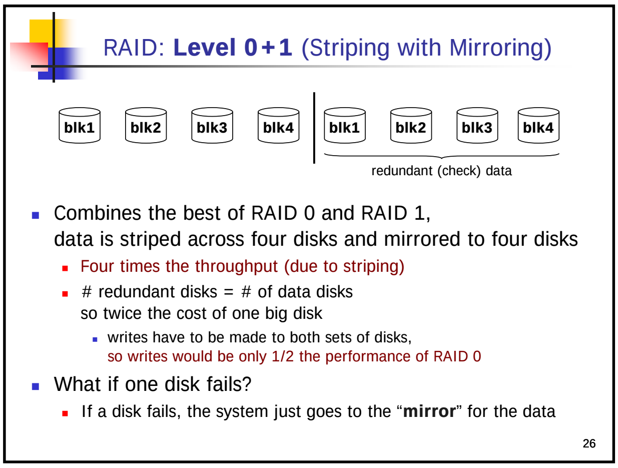
- Multiple smaller disks as opposed to one big disk
  - Spreading the blocks over multiple disks – **striping** – means that multiple blocks can be accessed in parallel increasing the performance
    - A 4 disk system gives four times the throughput of a 1 disk system
  - Same cost as one *big* disk – assuming 4 small disks cost the same as one big disk
- No redundancy, so what if one disk fails?



RAID: **Level 1** (Redundancy via Mirroring)

blk1.1
blk1.2
blk1.3
blk1.4
blk1.1
blk1.2
blk1.3
blk1.4
redundant (check) data

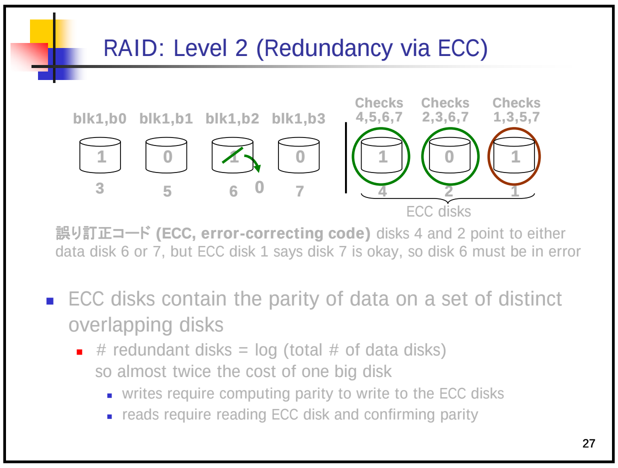
- Uses twice as many disks for redundancy so there are always two copies of the data
  - The number of redundant disks = the number of data disks so twice the cost of one big disk
    - writes have to be made to both sets of disks, so writes would be only 1/2 the performance of RAID 0
- What if one disk fails?
  - If a disk fails, the system just goes to the "**mirror**" for the data



RAID: **Level 0+1** (Striping with Mirroring)

blk1
blk2
blk3
blk4
blk1
blk2
blk3
blk4
redundant (check) data

- Combines the best of RAID 0 and RAID 1, data is striped across four disks and mirrored to four disks
  - Four times the throughput (due to striping)
  - # redundant disks = # of data disks so twice the cost of one big disk
    - writes have to be made to both sets of disks, so writes would be only 1/2 the performance of RAID 0
- What if one disk fails?
  - If a disk fails, the system just goes to the "**mirror**" for the data

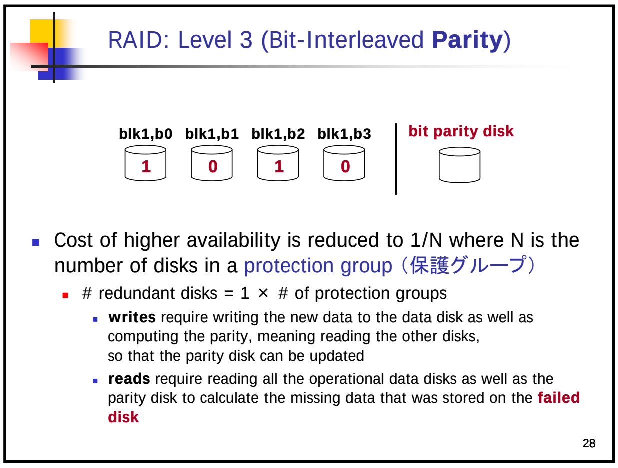


RAID: Level 2 (Redundancy via ECC)

blk1,b0
blk1,b1
blk1,b2
blk1,b3
Checks 4,5,6,7
Checks 2,3,6,7
Checks 1,3,5,7
1
0
1
0
1
0
1
3
5
6
0
7
4
2
1
ECC disks

誤り訂正コード (ECC, error-correcting code) disks 4 and 2 point to either data disk 6 or 7, but ECC disk 1 says disk 7 is okay, so disk 6 must be in error

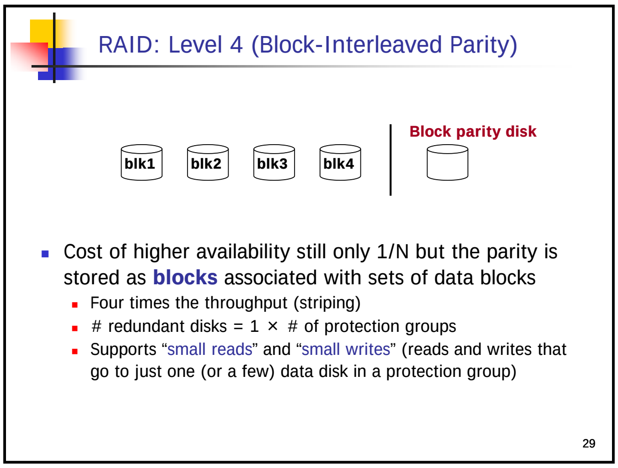
- ECC disks contain the parity of data on a set of distinct overlapping disks
  - # redundant disks = log (total # of data disks) so almost twice the cost of one big disk
    - writes require computing parity to write to the ECC disks
    - reads require reading ECC disk and confirming parity



RAID: Level 3 (Bit-Interleaved **Parity**)

blk1,b0
blk1,b1
blk1,b2
blk1,b3
1
0
1
0
bit parity disk

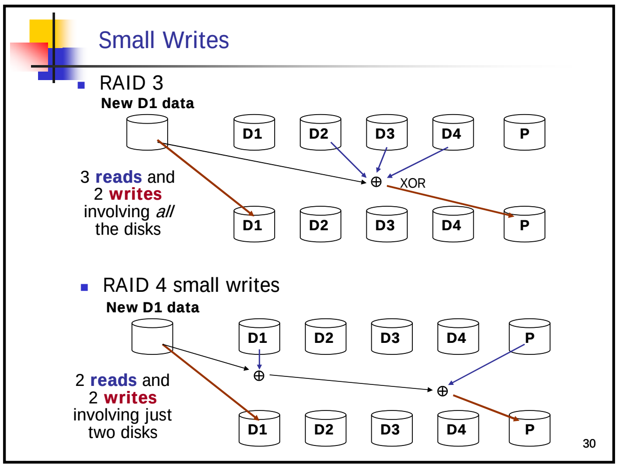
- Cost of higher availability is reduced to 1/N where N is the number of disks in a protection group（保護グループ）
  - # redundant disks = 1 × # of protection groups
    - **writes** require writing the new data to the data disk as well as computing the parity, meaning reading the other disks, so that the parity disk can be updated
    - **reads** require reading all the operational data disks as well as the parity disk to calculate the missing data that was stored on the **failed disk**



RAID: Level 4 (Block-Interleaved Parity)

blk1
blk2
blk3
blk4
Block parity disk

- Cost of higher availability still only 1/N but the parity is stored as **blocks** associated with sets of data blocks
  - Four times the throughput (striping)
  - # redundant disks = 1 × # of protection groups
  - Supports "small reads" and "small writes" (reads and writes that go to just one (or a few) data disk in a protection group)



Small Writes

- RAID 3

New D1 data
D1
D2
D3
D4
P
⊕
XOR
D1
D2
D3
D4
P

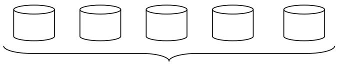
3 **reads** and
2 **writes**
involving *all*
the disks

- RAID 4 small writes

New D1 data
D1
D2
D3
D4
P
⊕
⊕
D1
D2
D3
D4
P

2 **reads** and
2 **writes**
involving just
two disks

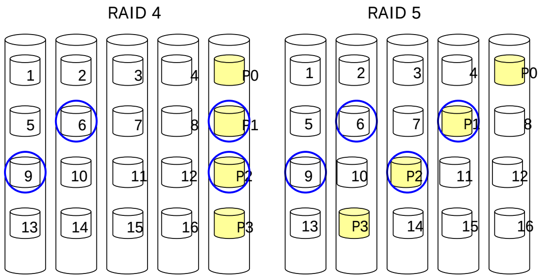
RAID: Level 5 (Distributed Block-Interleaved Parity)



one of these assigned as the block parity disk

- Cost of higher availability still only 1/N but the parity block can be located on any of the disks so there is no single bottleneck for writes
  - Still four times the throughput (striping)
  - # redundant disks = 1 × # of protection groups
  - Supports "**small reads**" and "**small writes**" (reads and writes that go to just one (or a few) data disk in a protection group)
  - Allows **multiple simultaneous writes**

Distributing Parity Blocks



RAID 4

1	2	3	4	P0
5	6	7	8	P1
9	10	11	12	P2
13	14	15	16	P3

RAID 5

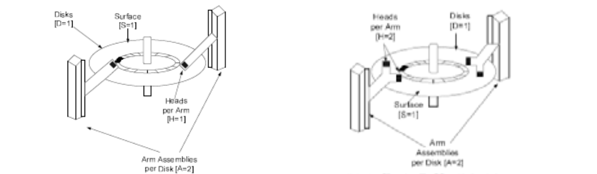
1	2	3	4	P0
5	6	7	P1	8
9	10	P2	11	12
13	P3	14	15	16

- By distributing parity blocks to all disks, some small writes can be performed **in parallel**

Disk and RAID **Summary**

- Four components of disk access time:
  - Seek Time: advertised to be 3 to 14 ms but lower in real systems
  - Rotational Latency: 5.6 ms at 5400 RPM and 2.0 ms at 15000 RPM
  - Transfer Time: 30 to 80 MB/s
  - Controller Time: typically less than .2 ms
- RAIDs can be used to improve availability
  - RAID 0 and RAID 5 – widely used in servers, one estimate is that 80% of disks in servers are RAIDs
  - RAID 1 (mirroring) – EMC, Tandem, IBM
  - RAID 3 – Storage Concepts
  - RAID 4 – Network Appliance
- RAIDs have enough redundancy to allow continuous operation

Intra-Disk Parallelism: An Idea Whose Time Has Come, ISCA2008



(a) A $D_1A_2S_1H_1$ disk drive.

(b) A $D_1A_2S_1H_2$ disk drive.

Figure 1. Example design points within the $DASH$ intra-disk parallelism taxonomy.

Adapted from *Computer Organization and Design*, Patterson & Hennessy, © 2005