

# Disks

wangth (2017-2020, CC BY-SA)  
? (1996-2016)

交大資工系資訊中心

Computer Center of Department of Computer Science, NCTU

# Handbook and Manual pages

- Official guide and be found at
  - <https://www.freebsd.org/doc/handbook/disks-adding.html>
  - <https://www.freebsd.org/doc/handbook/configtuning-disk.html>

# Outline

- Interface
- Geometry
- Add new disks
  - Installation procedure
  - Filesystem check
  - Add a disk
- RAID
  - GEOM

# Disk Interfaces

- SCSI
  - Small Computer Systems Interface
- IDE (or ATA) Low Price!
  - Integrated Device Electronics (or Advanced Technology Attachment)
  - Low cost
  - Become acceptable for enterprise with the help of RAID technology
- SATA Enhancement
  - Serial ATA
- SAS Speeds up!
  - Serial Attached SCSI
- USB
  - Universal Serial Bus
  - Convenient to use

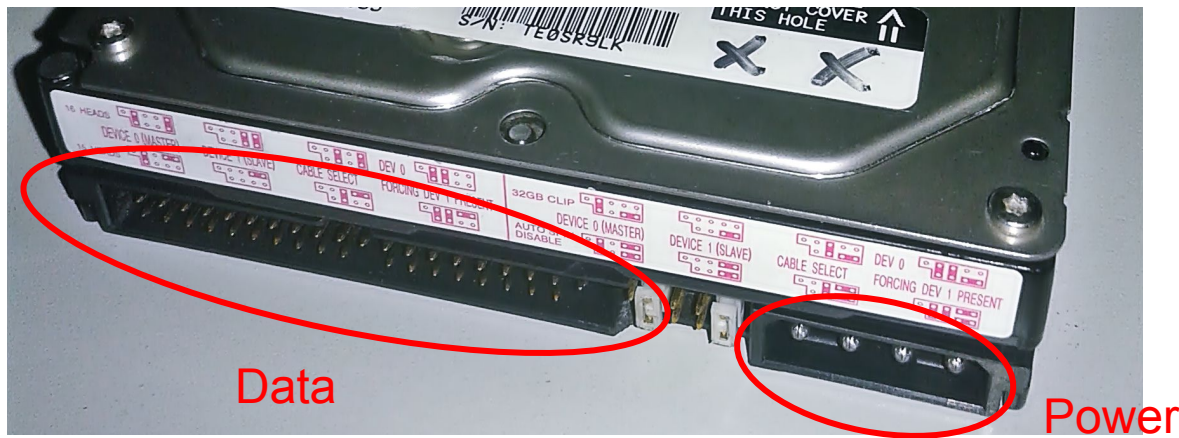
# Disk Interfaces - ATA & SATA

- ATA (AT Attachment)
  - ATA2
    - PIO, DMA
    - LBA (Logical Block Addressing)
  - ATA3, Ultra DMA/33/66/100/133
  - ATAPI (ATA Packet Interface)
    - CDROM, TAPE
  - Only one device can be active at a time
    - **SCSI support overlapping commands, command queuing, scatter-gather I/O**
  - **Master-Slave**                      **Primary Master (0) / Slave (1)**
  - 40-pin ribbon cable              **Secondary Master (2) / Slave (3)**
- SATA
  - Serial ATA
  - SATA-1 1.5Gbit/s, SATA-2 3Gbit/s, SATA-3 6Gbit/s
  - SATA 3.1, SATA 3.2 16Gbit/s, SATA 3.3, eSATA, mSATA

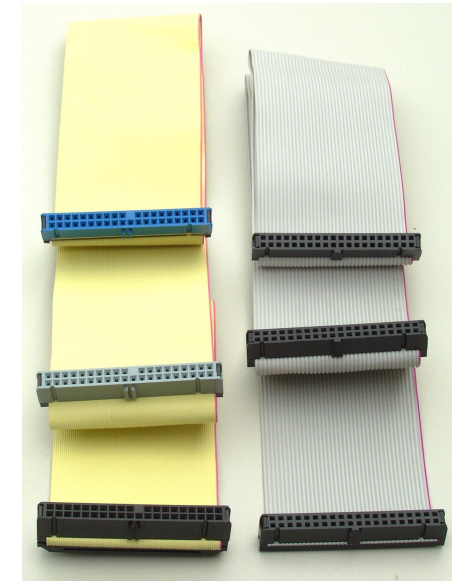


# Disk Interfaces - ATA & SATA Interfaces

- ATA interface and its cable

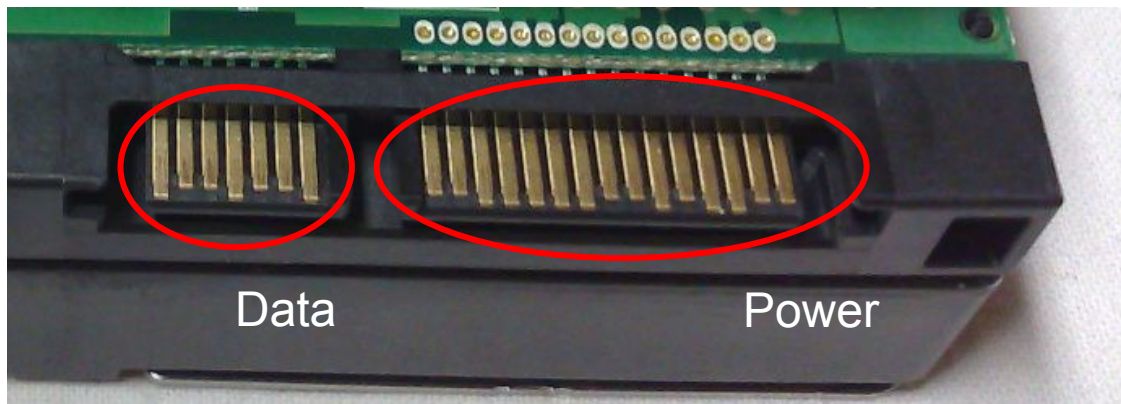


Credit: [JulianVilla26](#)



Credit: User [Smial](#) on [de.wikipedia](#)

- SATA interface and its cable



Credit: [Dsimic](#)



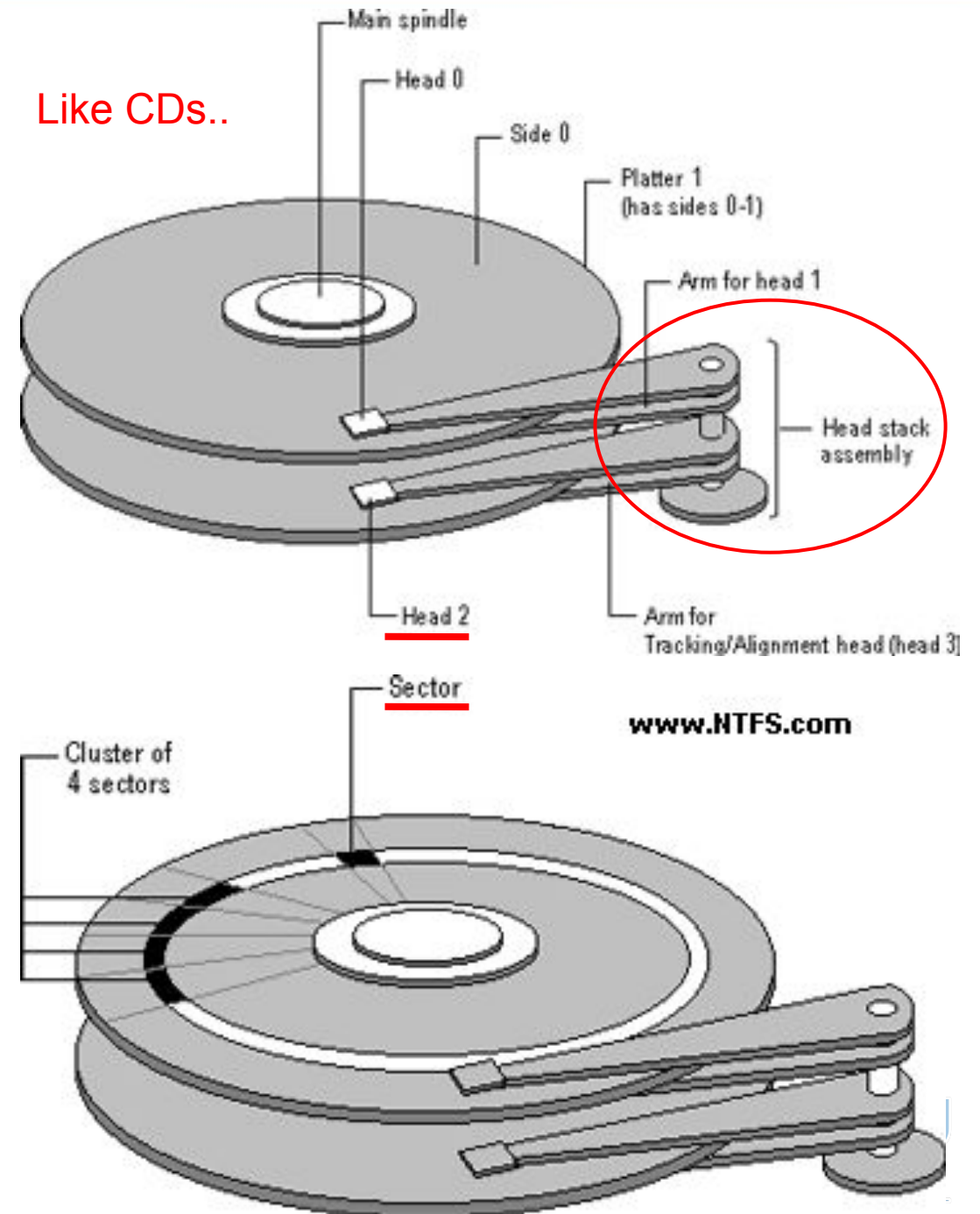
# Disk Interfaces - USB

- IDE/SATA to USB converters



# Disk Geometry (1)

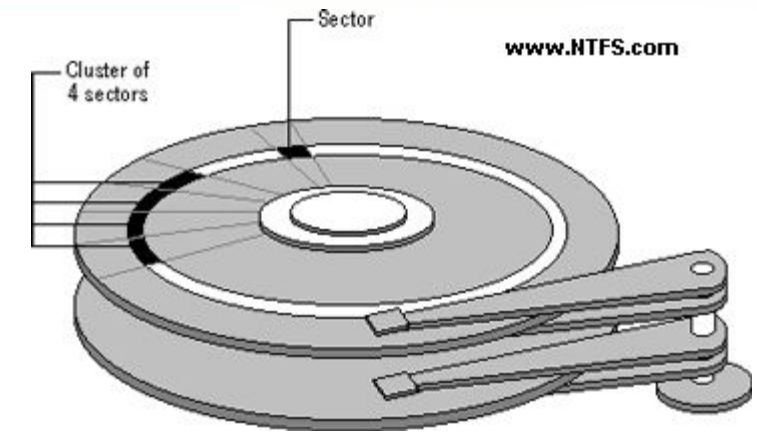
- Sector
  - Individual data block
- Track
  - circle
- Cylinder
  - circle on all platters
- Position
  - **CHS:**  
Cylinder,  
Head (0, 1, ...)  
Sector





# Disk Geometry (2)

- 40G HD
  - 4866 cylinders, 255 heads
  - 63 sectors per track, 512 bytes per sector
  - $512 * 63 * 4866 * 255 = \underline{40,024,212,480}$  bytes  
G M K
  - 1KB = 1024 bytes
  - 1MB = 1024 KB = 1,048,576 bytes
  - 1GB = 1024 MB = 1,073,741,824 bytes
  - $40,024,212,480 / 1,073,741,824 \doteq \underline{37.275}$  GB



10<sup>3</sup> vs. 2<sup>10</sup>...

# Disk Installation Procedure (in BSD...)

# Disk Installation Procedure (1)

- The procedure involves the following steps:

- Connecting the disk to the computer

- IDE: master/slave
- SATA
- SCSI: ID, terminator
- Power

- Creating device files

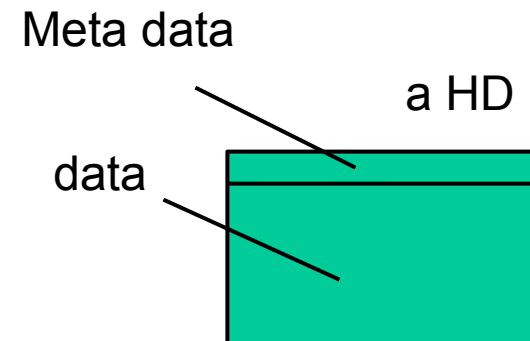
- Auto created by devfs

- Formatting the disk

- Low-level format

- Manufacturer diagnostic utility
- **Kill all** address information and timing marks on platters
- Repair bad sectors  mark the bad sectors and don't use them!

Please do it offline...



Format (metadata + data)  
v.s. fast format (metadata only)

# Disk Installation Procedure (2)

- Partitioning (and Labeling) the disk
  - Allow the disk to be treated as a group of independent data area
  - e.g. root, home, swap partitions
  - Former Suggestions:
    - /var, /tmp
      - separate partition (for backup issue)
    - Make a copy of root filesystem for emergency
- Establishing logical volumes
  - Combine multiple partitions into a logical volume
  - Related to RAID
  - Software RAID technology
    - GEOM: [geom\(4\)](#), [geom\(8\)](#)
    - ZFS: [zpool\(8\)](#), [zfs\(8\)](#), [zdb\(8\)](#)

# Disk Installation Procedure (3)

- Creating UNIX filesystems within disk partitions
  - Use "**newfs**" to install a filesystem for a partition
  - Establish all filesystem components
    - A set of inode storage cells
    - A set of data blocks
    - A set of superblocks
    - A map of the disk blocks in the filesystem
    - A block usage summary

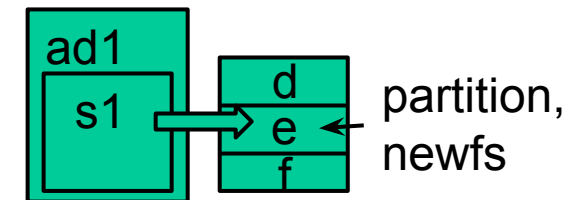


# Disk Installation Procedure (4)

- Superblock contents
  - The length of a disk block
  - Size and location of inode table
  - Disk block map
  - Usage information
  - Other filesystem's parameters
- sync
  - The *sync() system call* forces a write of dirty (modified) buffers in the block buffer cache out to disk.
  - The *sync utility* can be called to ensure that all disk writes have been completed before the processor is halted in a way not suitably done by `reboot(8)` or `halt(8)`.

# Disk Installation Procedure (5)

- mount
  - Bring the new partition to the filesystem tree
  - mount point can be any directory (empty)
  - `$ mount /dev/ad1s1e /home2`
- Setting up automatic mounting
  - Automount at boot time
    - `/etc/fstab`
    - `$ mount -t ufs /dev/ad2s1a /backup`
    - `$ mount -t cd9600 -o ro,noauto /dev/acd0c /cdrom`



Mount CD  
Also for ISO image file

```
liuyh@NASA:/etc> cat fstab
# Device          Mountpoint  Fstype    Options    Dump    Pass#
/dev/ad0s1b      none       swap      sw         0        0
/dev/ad2s1b      none       swap      sw         0        0
/dev/ad0s1a      /          ufs       rw         1        1
/dev/acd0        /cdrom     cd9660    ro,noauto  0        0
/dev/ad2s1a      /backup    ufs       rw,noauto  2        2
csduty:/bsdhome /bsdhome   nfs       rw,noauto  0        0
```

Mount from the network; talk about it in “NFS”...

Usually: 2, 1 for root;  
No write = 0

# Disk Installation Procedure (6)

- Setting up swapping on swap partitions
  - [swapon](#), [swapoff](#), [swapctl](#)
    - `$ swapon -a`
      - mount all partitions for swap usage
  - [swapinfo](#), [pstat](#)

```
nctucs [~] -wangth- swapinfo
Device          1K-blocks  Used   Avail   Capacity
/dev/da0p2      2097152    42772  2054380  2%
```

# fsck – check and repair filesystem (1)

- System crash will cause
  - Inconsistency between memory image and disk contents
- fsck
  - Examine all local filesystem listed in /etc/fstab at boot time. (fsck -p)
  - Automatically correct the following damages:
    - Unreferenced inodes
    - Inexplicably large link counts
    - Unused data blocks not recorded in block maps
    - Data blocks listed as free but used in file
    - Incorrect summary information in the superblock
    - [fsck\(8\)](#)、[fsck\\_ffs\(8\)](#)
      - Check if filesystem is clean...
      - 1: clean (ro)
      - 0: dirty (rw)
    - [ffsinfo\(8\)](#): dump metadata

# fsck – check and repair filesystem (2)

- Run fsck in manual to fix serious damages

There is no guarantee that fsck will fully recover your disk.

- Blocks claimed by more than one file
  - Blocks claimed outside the range of the filesystem
  - Link counts that are too small
  - Blocks that are not accounted for
  - Directories that refer to unallocated inodes
  - Other errors
- fsck will suggest you the action to perform
    - Delete, repair, ...



# Adding a disk to FreeBSD (1)

## 1. Check disk connection

- Look system boot message

```
ada3: 238475MB <Hitachi HDS722525VLAT80 V360A6MA> at ata1-slave UDMA100
```

## 2. Use [gpart\(8\)](#) to create a partition on the new HD

Line, speed

- \$ gpart create -s GPT ada3
- \$ gpart add -t freebsd-ufs -a 1M ada3

## 3. Use [newfs\(8\)](#) to construct new UFS file system

- \$ newfs -U /dev/ada3p1

## 4. Make mount point and mount it

- \$ mkdir /home2
- \$ mount -t ufs /dev/ada3p1 /home2
- \$ df

## 5. Edit /etc/fstab

# Adding a disk to FreeBSD (2)

- If you forget to enable soft-update when you add the disk
  - \$ umount /home2
  - \$ tunefs -n **enable** /dev/ada3p1
  - \$ mount -t ufs /dev/ada3p1 /home2
  - \$ mount

```
/dev/ada0p2 on / (ufs, local, soft-updates)
/dev/ada1p1 on /home (ufs, local, soft-updates)
procfs on /proc (procfs, local)
/dev/ada3p1 on /home2 (ufs, local, soft-updates)
```

# GEOM

## Modular Disk Transformation Framework

交大資工系資訊中心

Computer Center of Department of Computer Science, NCTU

# Handbook and Manual pages

- Official guide and be found at
  - <http://www.freebsd.org/doc/handbook/geom.html>

# GEOM - (1)

- Support
  - ELI – [geli\(8\)](#): cryptographic GEOM class
  - JOURNAL – [gjournal\(8\)](#): journaled devices Journalize (logs) before write
  - LABEL – [glabel\(8\)](#): disk labelization
  - MIRROR – [gmirror\(8\)](#): mirrored devices Software RAID1
  - STRIPE – [gstripe\(8\)](#): striped devices Software RAID0
  - ...



# GEOM - (2)

- GEOM framework in FreeBSD

- Major RAID control utilities
- Kernel modules (/boot/kernel/geom\_\*)
- Name and Providers ← devices

Logical  
volumes ↗

- "manual" or "automatic"
- Metadata in the last sector of the providers



- Kernel support

- {glabel,gmirror,gstripe,g\*} load/unload
  - device GEOM\_\* in kernel config
  - geom\_\*\_enable="YES" in /boot/loader.conf

- (1) On demand load/unload kernel modules
  - load automatically at booting
- (2) Build-in kernel and recompile

# GEOM - (3)

- LABEL **Bundle by name instead of bundle by provider**
  - Used for GEOM provider labelization
  - Kernel
    - device GEOM\_LABEL **e.g. ad0s1d => usr**
    - geom\_label\_load="YES"
  - glabel (for new storage) **glabel label ... => Create permanent labels**  
**glabel create ... => Create transient labels**
    - \$ glabel label -v usr da2
    - \$ newfs /dev/label/usr **/dev/label/usr**
    - \$ mount /dev/label/usr /usr
    - \$ glabel stop usr **Stop using the name**
    - \$ glabel clear da2 **Clear metadata on provider**
  - UFS label (for an using storage)
    - \$ tuneufs -L data /dev/da4s1a **"data" is a name**
    - \$ mount /dev/ufs/data /mnt/data

# GEOM - (4)

- MIRROR
  - Kernel
    - device GEOM\_MIRROR
    - geom\_mirror\_load="YES"
  - gmirror
    - \$ gmirror label -v -b round-robin data da0
    - \$ newfs /dev/mirror/data ← logical volume called "data", using HD: da0, ...
    - \$ mount /dev/mirror/data /mnt
    - \$ gmirror insert data da1 ← Add a HD into the volume
    - \$ gmirror forget data ← Remove non-existent HDs
    - \$ gmirror insert data da1
    - \$ gmirror stop data
    - \$ gmirror clear da0

# GEOM - (5)

- STRIPE

- Kernel

- device GEOM\_STRIPE
- geom\_stripe\_load="YES"

- gstripe

- \$ gstripe label -v -s 131072 data da0 da1 da2 da3
- \$ newfs /dev/stripe/data
- \$ mount /dev/stripe/data /mnt
- \$ gstripe stop data
- \$ gstripe clear da0

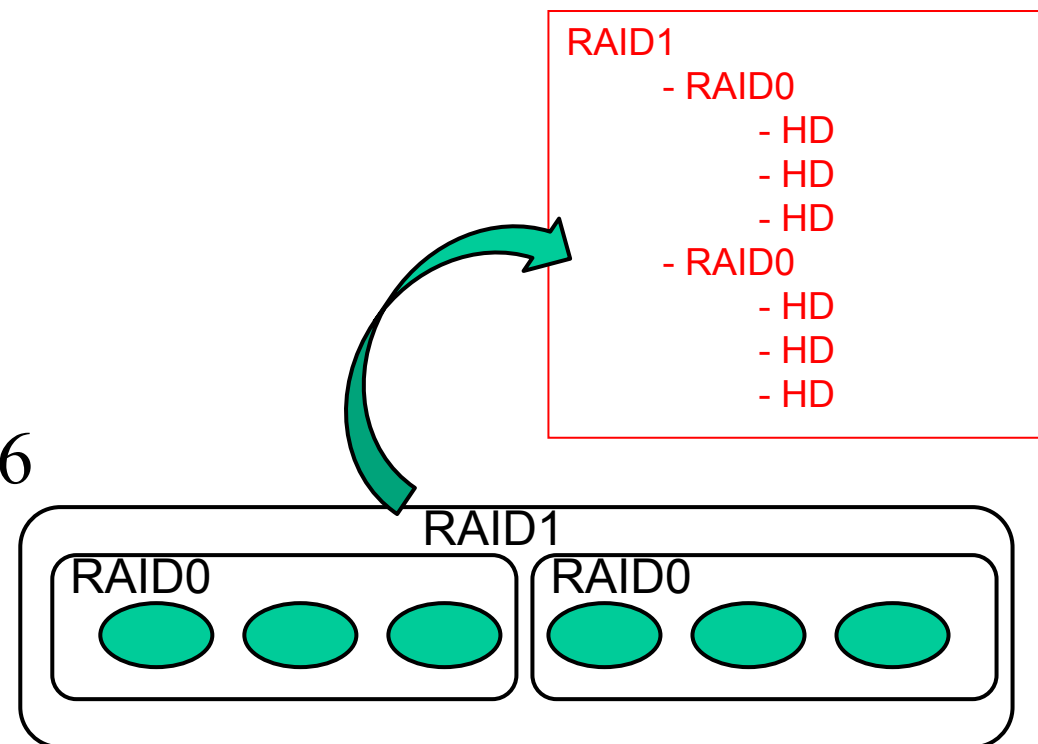
← Create logical volume "data",  
which stripe da0~da3 HDs

# Appendix



# RAID - (1)

- Redundant Array of Inexpensive Disks
  - A method to combine several physical hard drives into one logical unit
    - e.g. HD1, HD2 v.s D:\ in windows
- Depending on the type of RAID, it has the following benefits:
  - Fault tolerance
  - Higher throughput
  - Real-time data recovery
- RAID Level
  - RAID 0, 1, 0+1, 2, 3, 4, 5, 6
  - Hierarchical RAID



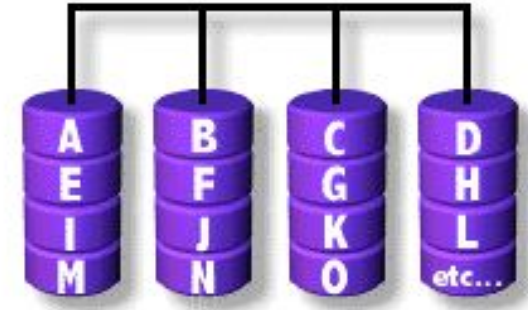
# RAID - (2)

- Hardware RAID
  - There is a dedicate controller to take over the whole business
  - RAID Configuration Utility after BIOS
    - Create RAID array, build Array
- Software RAID
  - GEOM
    - CACHE, CONCAT, ELI, JOURNAL, LABEL, MIRROR, MULTIPATH, NOP, PART, RAID3, SHSEC, STRIPE, VIRSTOR
  - ZFS
    - JBOD, STRIPE
    - MIRROR
    - RAID-Z, RAID-Z2, RAID-Z3

# RAID 0 (normally used)

(500GB+500GB=1TB)

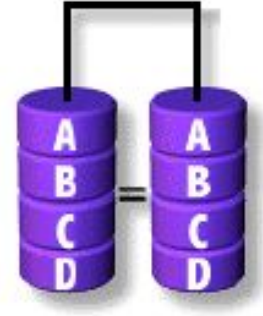
- Stripped data into several disks
- Minimum number of drives: 2 e.g. HD1 (500GB), HD2 (500GB)  
v.s. D:\ in windows (1TB)
- Advantage
  - Performance increase in proportional to n **theoretically**
  - Simple to implement parallel file io from/to different HDs
- Disadvantage
  - No fault tolerance
- Recommended applications
  - Non-critical data storage
  - Application requiring high bandwidth (such as video editing)



# RAID 1 (normally used)

(500GB+500GB=500GB)

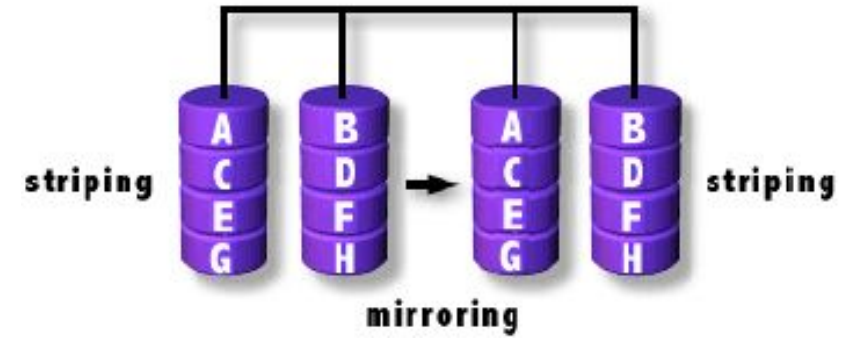
- Mirror data into several disks
- Minimum number of drives: 2
- Advantage
  - 100% redundancy of data
- Disadvantage
  - 100% storage overage
  - Moderately slower write performance
- Recommended application Caused by double check mechanisms on data...
  - Application requiring very high availability (such as home)



# RAID 0+1 (normally used)

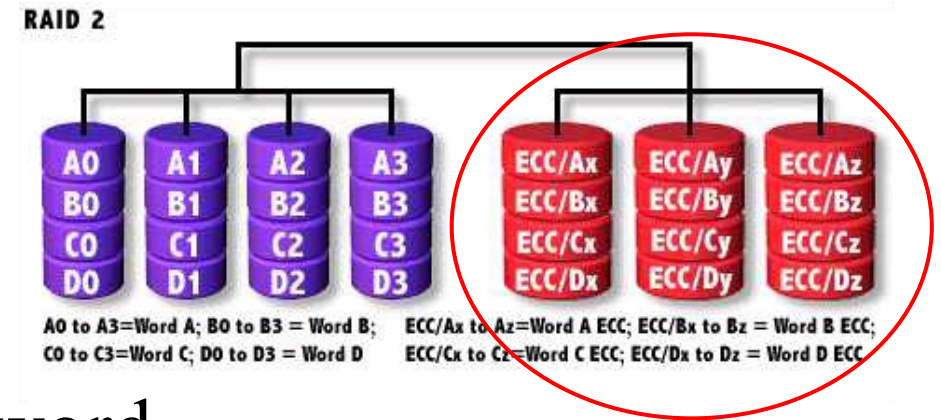
$((500\text{GB}+500\text{GB})+(500\text{GB}+500\text{GB})=1\text{TB})$

- Combine RAID 0 and RAID 1
- Minimum number of drives: 4



RAID1, RAID1  
Them RAID0 above it

# RAID 2



- Hamming Code ECC Each bit of data word
- Advantage
  - "On the fly" data error correction
- Disadvantage
  - Inefficient
  - Very high ratio of ECC disks to data disks
- Recommended applications
  - No commercial implementations exist / not commercially viable

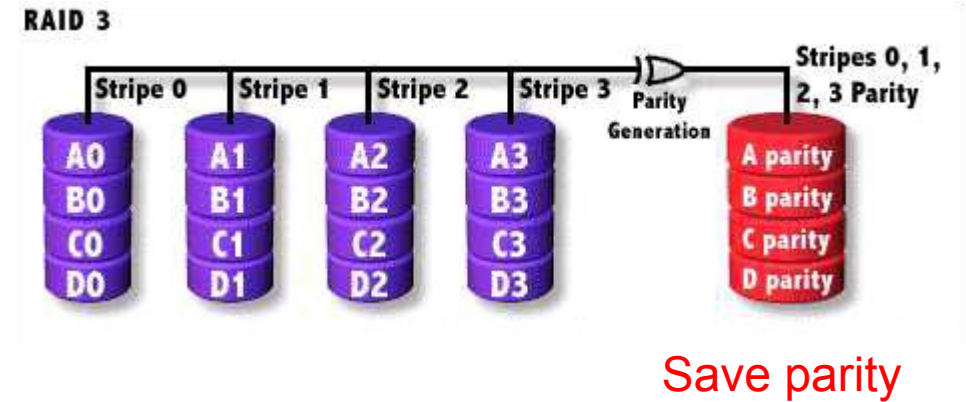
Read, check if correct, then read



# RAID 3

RAID1 if two HDs

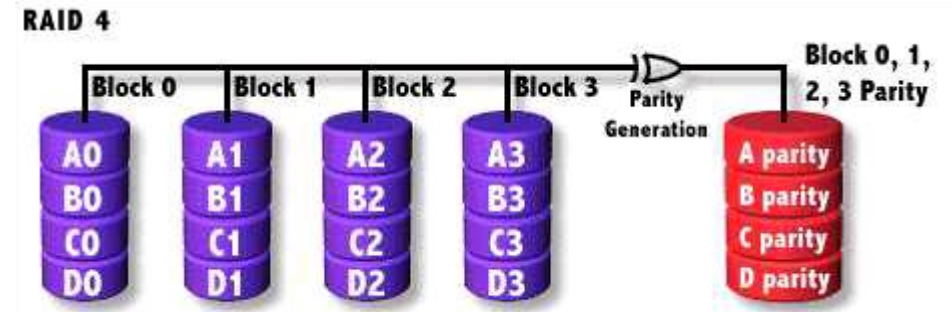
- Parallel transfer with Parity
- Minimum number of drives: 3
- Advantage
  - Very high data transfer rate
- Disadvantage
  - Transaction rate equal to that of a single disk drive at best
- Recommended applications
  - Any application requiring high throughput



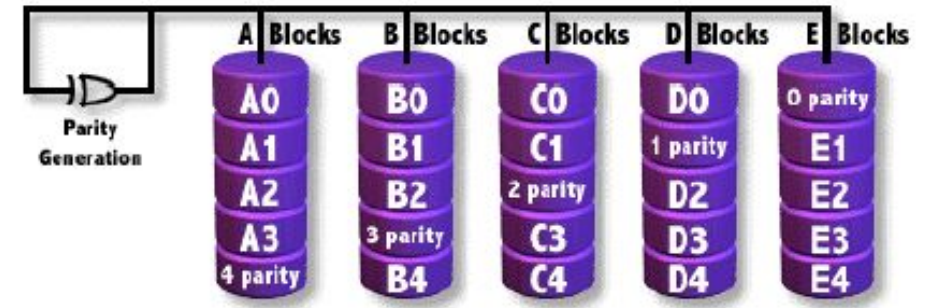
# RAID 4

- Similar to RAID3
- RAID 3 V.S RAID 4
  - Byte Level V.S Block Level
  - Block interleaving
    - Small files (e.g. 4k)

Block normally 512bytes (4k for WD HDs)



# RAID 5 (normally used)



- Independent Disk with distributed parity blocks
- Minimum number of drives: 3
- Advantage **Parallel file I/O**
  - Highest read data rate
  - Medium write data rate
- Disadvantage
  - Disk failure has a medium impact on throughput
  - Complex controller design
  - When one disk failed, you have to rebuild the RAID array

Origin from RAID3

Can tolerate only 1 HD failure

# RAID 6 (normally used)

- Similar to RAID5
- Minimum number of drives: 4
- 2 parity checks, 2 disk failures tolerable.

Slower than RAID5 because of storing 2 parities...

