Data storage

- A challenge in **every age**: from the first stone carvings to the digital revolution.
- **Principles:** safety, validity, integrity.
- **Data:** piece of information.
- **Digital data:** bits, bytes, k, M, G, T, P, E ...

Virtualization

- A way of **decoupling data storage** from its physical appearance.
- At **any levels** of data storage hierarchy.
- Not a magic word, but a set of real technical solutions: e.g. volume management, device mapping, clustered file systems.
- Layers **hide details** beneath, **offer services** above.

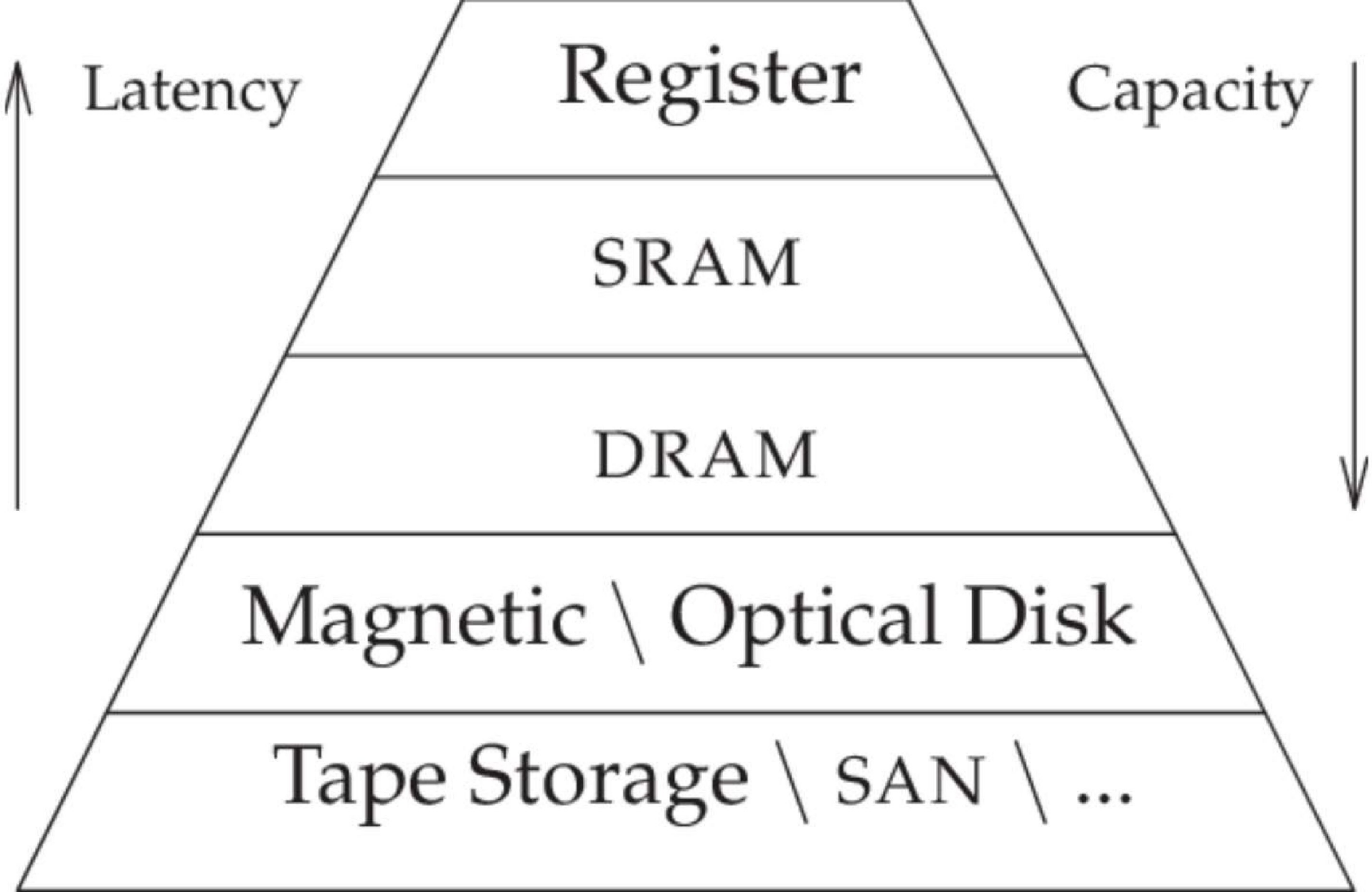
[1] https://en.wikipedia.org/wiki/Storage_virtualization]

Virtualization methods

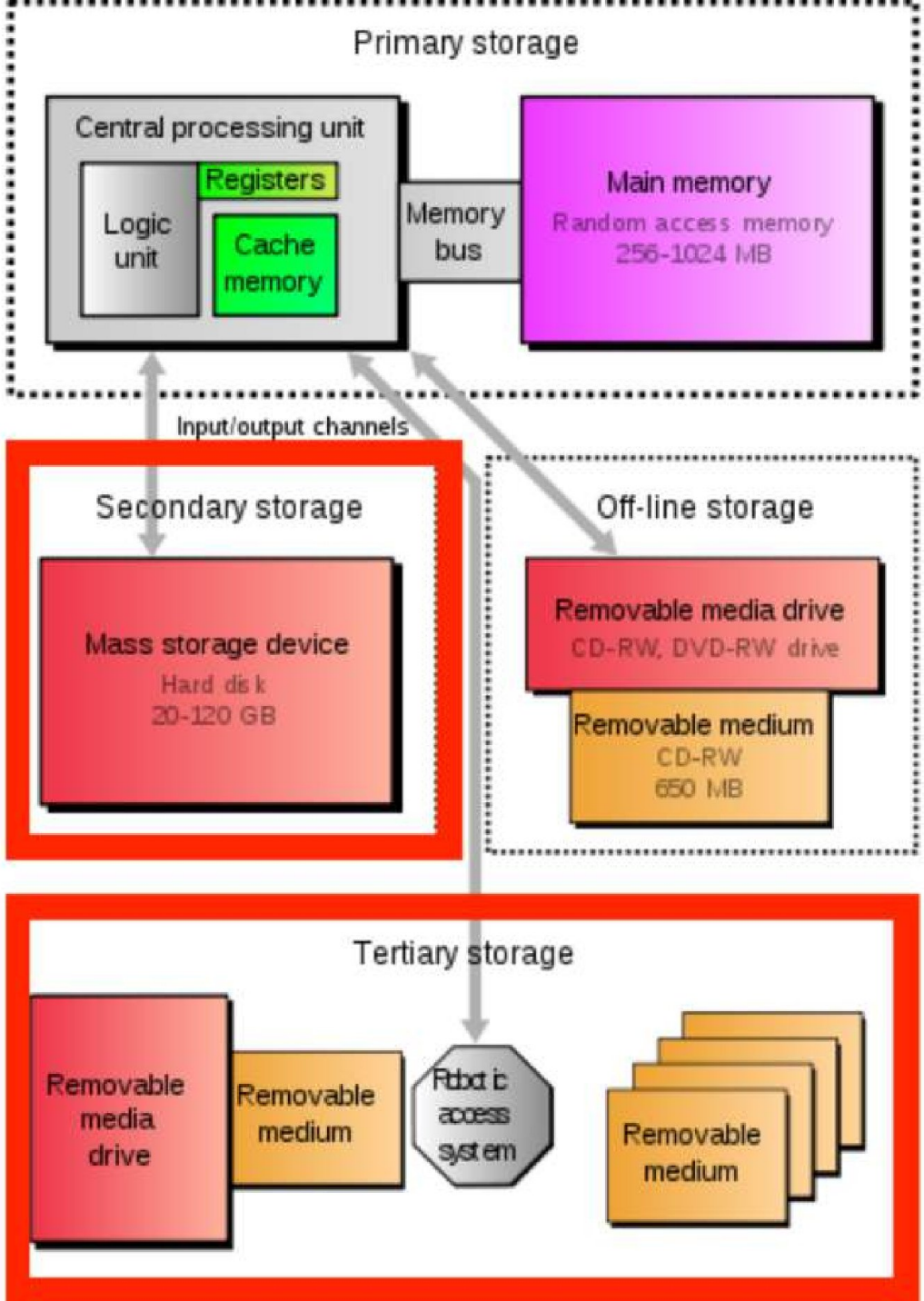
Block device based: it creates a mapping between the provider (storage) and user of data (host) on block level

- Host based (LVM, LDM);
- Storage device based: RAID controllers;
- Network based: iSCSI, FC;

File system based: it offers generalized access to different file systems in a uniform way, e.g. VFS.


Latency
Capacity
Register
SRAM
DRAM
Magnetic \ Optical Disk
Tape Storage \ SAN \ ...

apacity



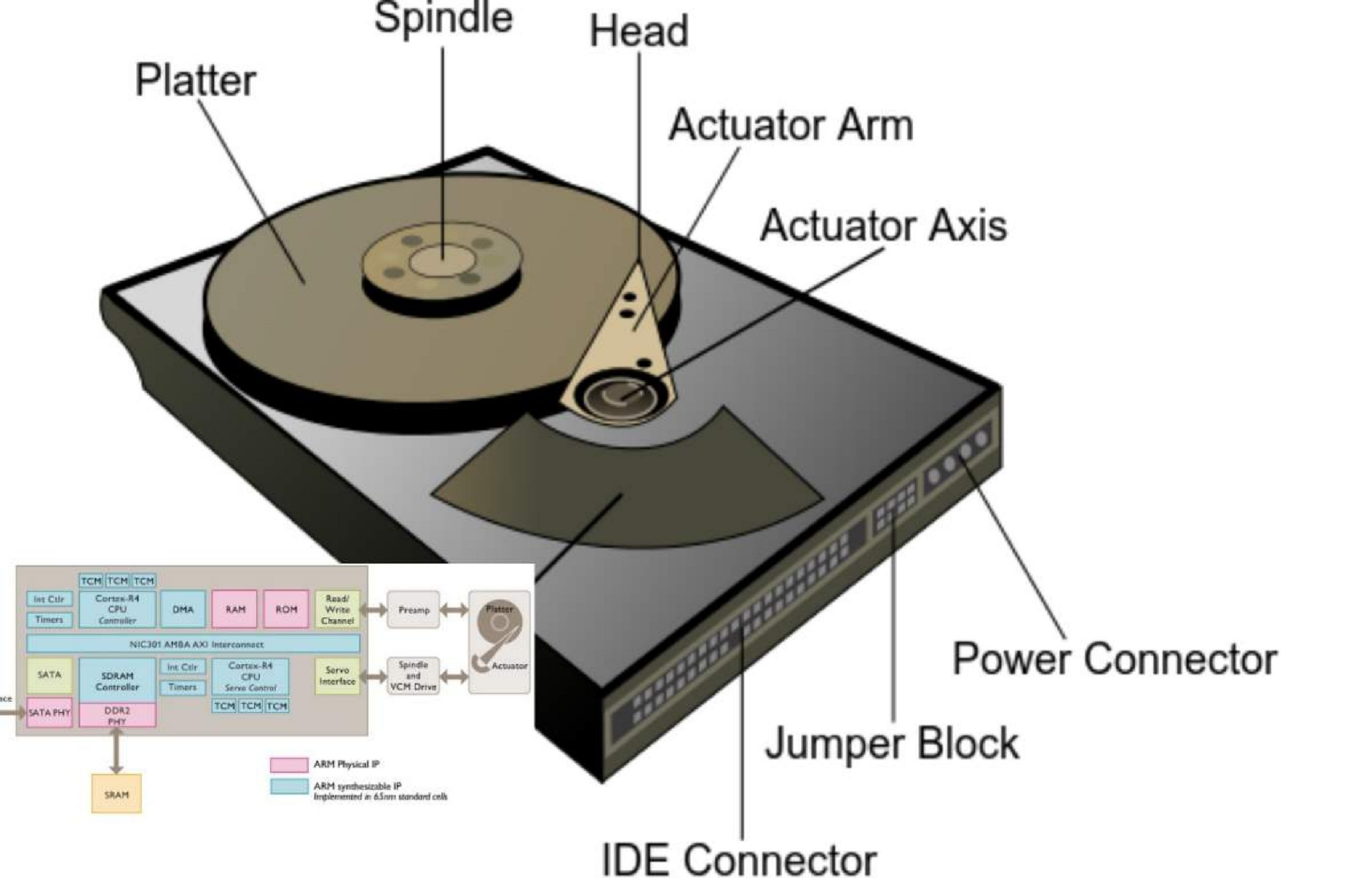
High

Volun

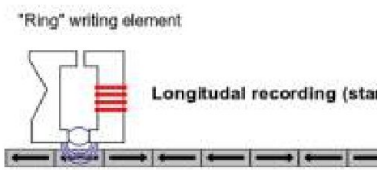
Stora

Sto

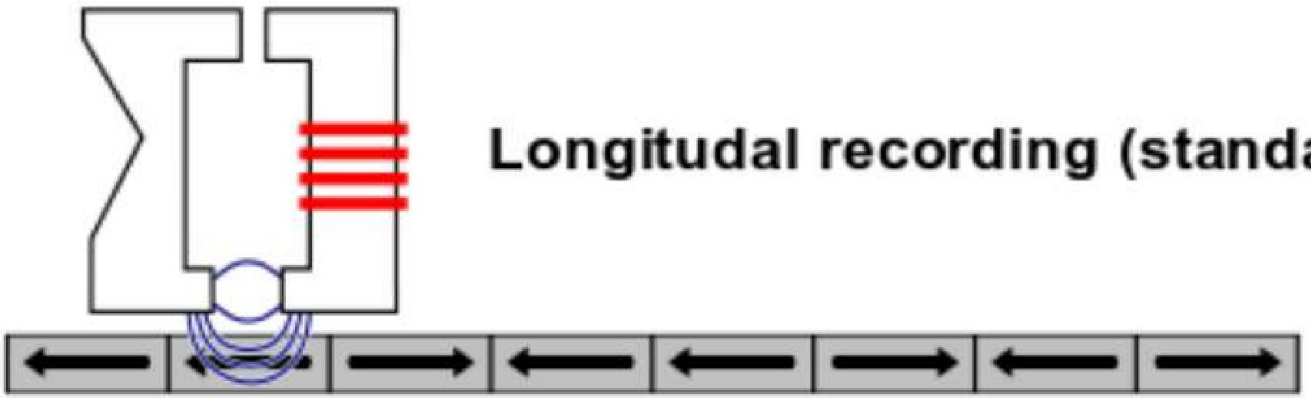
Ph
(dis


Spindle
Head
Platter
Actuator Arm
Actuator Axis
Power Connector
Jumper Block
IDE Connector
TCM
Int Ctlr
Timers
Cortex-R4 CPU Controller
DMA
RAM
ROM
Read/ Write Channel
Preamp
Platter
NIC301 AMBA AXI Interconnect
SATA
SDRAM Controller
Int Ctlr
Timers
Cortex-R4 CPU Servo Control
Servo Interface
Spindle and VCM Drive
Actuator
SATA PHY
DDR2 PHY
SRAM
ARM Physical IP
ARM synthesizable IP
Implemented in 65nm standard cells

- **Magnetic recording**:
  - Magnetic cover on the platters;
  - **Direction changes** in magnetization store bits;
- Disk controller is used to control the rotation as well as the head positioning;
- Data is stored in **blocks** from 512 Byte (classic) to 4096 Byte (modern);
- **Architecture:** media - buffer - interface;
- **Longitudinal** and **vertical** recording.



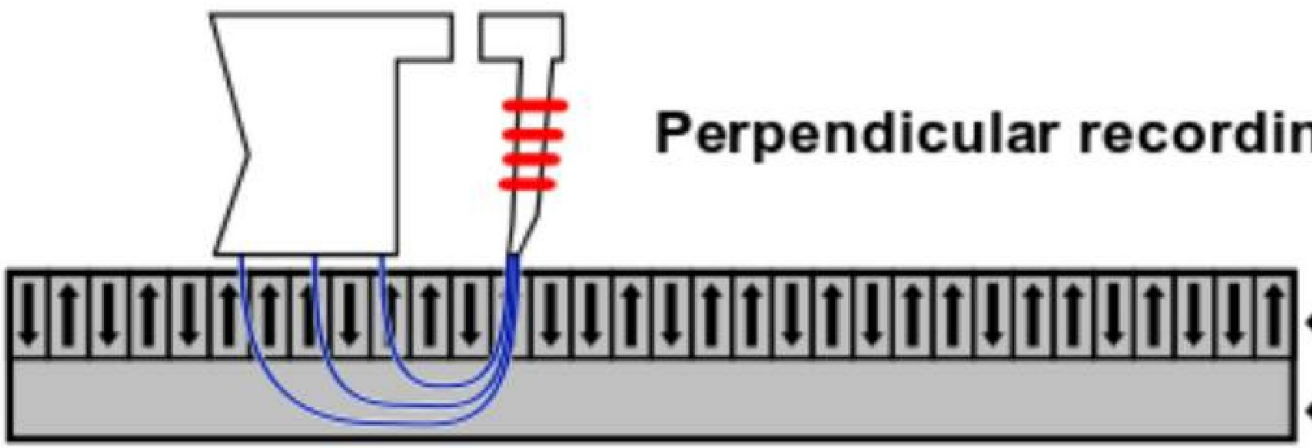
"Ring" writing element



Longitudal recording (standard)

Recording layer

"Monopole" writing element



Perpendicular recording

Recording layer

Additional layer

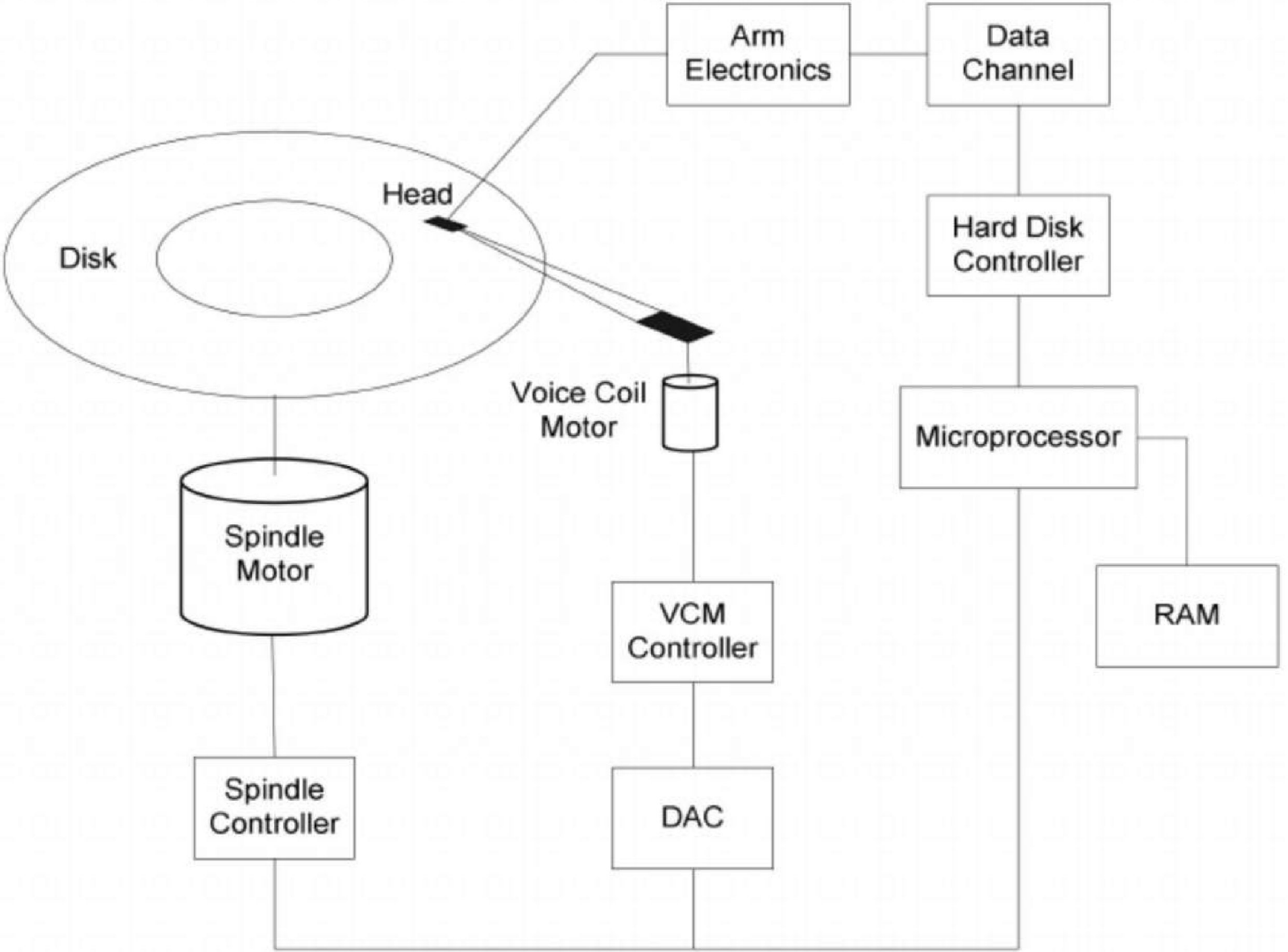
IDE connector

Characterized by

- **capacity** (Bytes, 1-10 Tbytes)
  - OS reports less for many reasons;
  - error correction: 10% of space;
  - redundancy + FS structures;
  - decimal or binary prefix;
- **speed** (rpm, 4200-15000 rpm)
- **latency** (sec, 2-6 ms);
- **throughput** (bit/s, 1-3 Gbit/s);
- **form factor** (2.5" vs 3.5");
- **energy consumption** (W);

- **Des**
  s, 54
- **Lap**
  420
- **Ente**

- **Desktop:** 4-12 Tbytes, 3.5", 0.5 Gbit/s, 5400-10000 rpm;
- **Laptop:** 1-4 Tbytes, 2.5", 0.5 Gbit/s, 4200-7200 rpm;
- **Enterprise:** 1-12 Tbytes, any, 1.6 Gbit/s, 10000-15000 rpm;


Arm Electronics
Data Channel
Head
Disk
Hard Disk Controller
Voice Coil Motor
Microprocessor
Spindle Motor
VCM Controller
RAM
Spindle Controller
DAC

Interfaces:

the physical + logical link

- IDE, EIDE;
- ATA, PATA, SATA;
- SCSI;
- SAS;
- FC.

Magnetic tapes

- Has a long history, since 1951, IBM + DEC were pioneers.
- Allows **sequential data** storage.
- Operations: record, play, fast forward, rewind.
- BOT ... EOT.
- Media write: **linear** vs. **helical** tracks.
- **Exposed tapes** ... **cartridges** (with flash memory).
- Access time: 3 magnitude longer than those of HDD's.
- **Compression**: redundant data compressed in a limited window buffer (2:1 typical).
- **Encryption**: for protecting data. **Key management!**

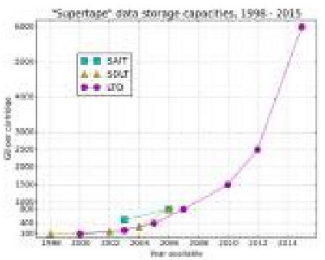
Linear Tape Open (LTO):

LTO1, 100 GB, 200 GB, 20 MB/s;
LTO2, 200 GB, 400 GB, 40 MB/s;
...
LTO7, **6 TB**, 15 TB, 750 MB/s;
LTO8, **12 TB**, 30 TB, 900 MB/s;
....
One step backward compatibility.

Error correction: CRC + repeated read.
Data blocks: Mbytes of block size.
Designed for data archiving, i.e. 15-30 years.
Data structures: data + metadata.
.tar format.
Labels, BAR-codes.
Cleaning tapes.

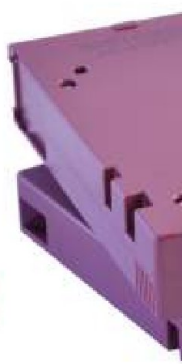
Tape drives + **tape libraries**:

- Take tapes in parallel.
- Collect a huge set of cartridges.
- Offer **standard interfaces** to hosts.



[4] https://en.wikipedia.org/wiki/Tape_drive

- Has a long history, since 1951, IBM + DEC were pioneers.
- Allows **sequential data** storage.
- Operations: record, play, fast forward, rewind.
- BOT ... EOT.
- Media write: **linear** vs. **helical** tracks.
- **Exposed tapes** ... **cartridges** (with flash memory).
- Access time: 3 magnitude longer than those of HDD's.
- **Compression**: redundant data compressed in a limited window buffer (2:1 typical).
- **Encryption**: for protecting data. **Key management!**



Linear Tape Open (LTO):

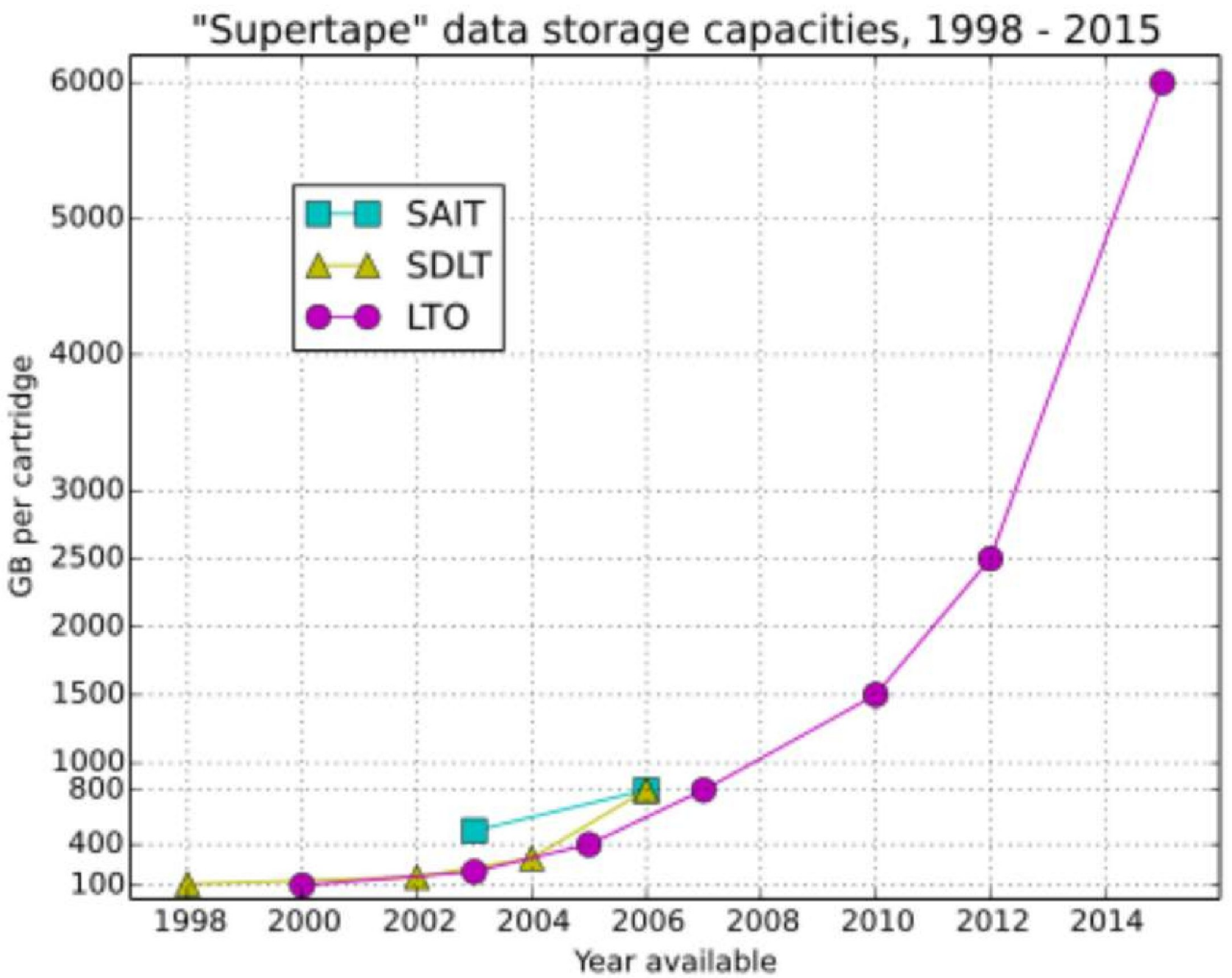
LTO1, **100 GB**, 200 GB, 20 MB/s;
LTO2, **200 GB**, 400 GB, 40 MB/s;

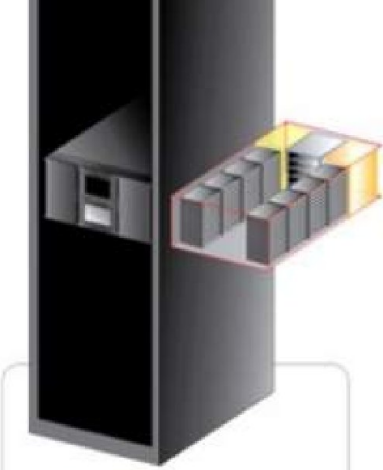
...

LTO7, **6 TB**, 15 TB, 750 MB/s;
LTO8, **12 TB**, 30 TB, 900 MB/s;

....

One step backward compatibility.

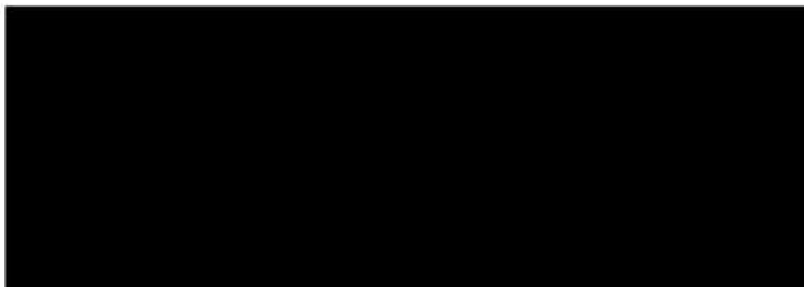

"Supertape" data storage capacities, 1998 - 2015
GB per cartridge
Year available
SAIT
SDLT
LTO
6000
5000
4000
3000
2500
2000
1500
1000
800
400
100
1998
2000
2002
2004
2006
2008
2010
2012
2014

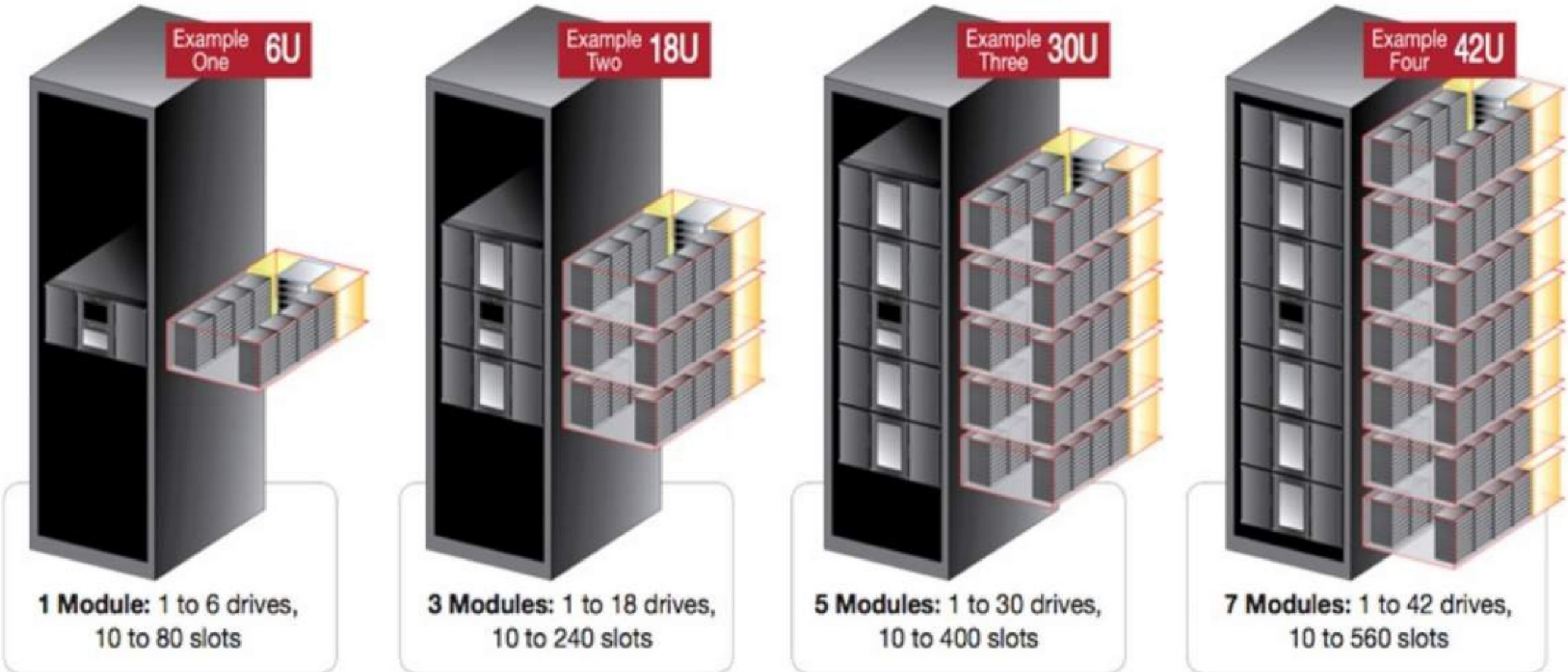


1 Module: 1 to 6 drives, 10 to 80 slots

Tape drives + **tape libraries**:

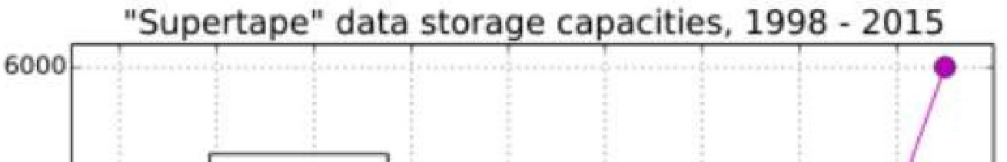
- Take tapes in parallel.
- Collect a huge set of cartridges.
- Offer **standard interfaces** to hosts.




Example One 6U
Example Two 18U
Example Three 30U
Example Four 42U
1 Module: 1 to 6 drives, 10 to 80 slots
3 Modules: 1 to 18 drives, 10 to 240 slots
5 Modules: 1 to 30 drives, 10 to 400 slots
7 Modules: 1 to 42 drives, 10 to 560 slots

One step backward compatibility.

Error correction: CRC + repeated read.
Data blocks: Mbytes of block size.
Designed for data archiving, i.e. 15-30 years.
Data structures: data + metadata.
.tar format.
Labels, BAR-codes.
Cleaning tapes.



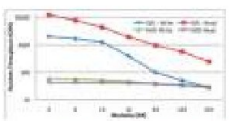
Solid State Drives

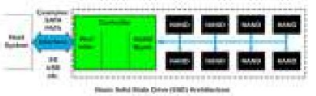
- A sort of **flash memory** (NAND), meant to replace (suplement) HDDs. Same **size**, similar **form factors** (exception M.2), **same interfaces**:
  - **SAS**, 12 Gbit/s;
  - **PCI-e 3.0**, 31 Gbit/s;
- Non-volatile, stores data when losing power.
- Stores data **as electrical charges**.

Controller:
- R/W cache + **battery**;
- **bad block** mapping;
- **wear leveling**;

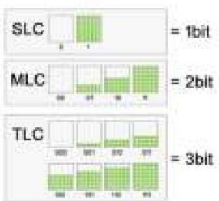
Advantages:
- fast random read;
- no moving parts;
- no spin-up time;
- less energy consuming;
- silent.

Disadvantages:
- high cost of capacity;
- limited amount of writes;
- faster, no moving parts, but have **endurance time**: 1-2 years (except Intel's 3D Point);
- expensive;
- sophisticated controller.





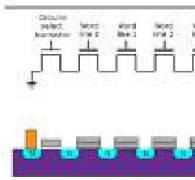
- Information is stored in **floating gate transistors**, i.e. read and written, holds charge for a long period.
- **Levels** of data storage:
  - single level cell: **1 bit/cell**, fast, endurable (50000-100000 erases), small;
  - 2-level cell: **2 bits/cell**, slower, less endurable (1000-10000 erases), medium;
  - 4-level cell: **4 bits/cell**, slower, less endurable, large.
- **Capacities:** 32 Gbyte - 4 Tbyte.
- Combined HDD + SSD devices.



[4] https://en.wikipedia.org/wiki/Solid-state_drive

Solid State Drives

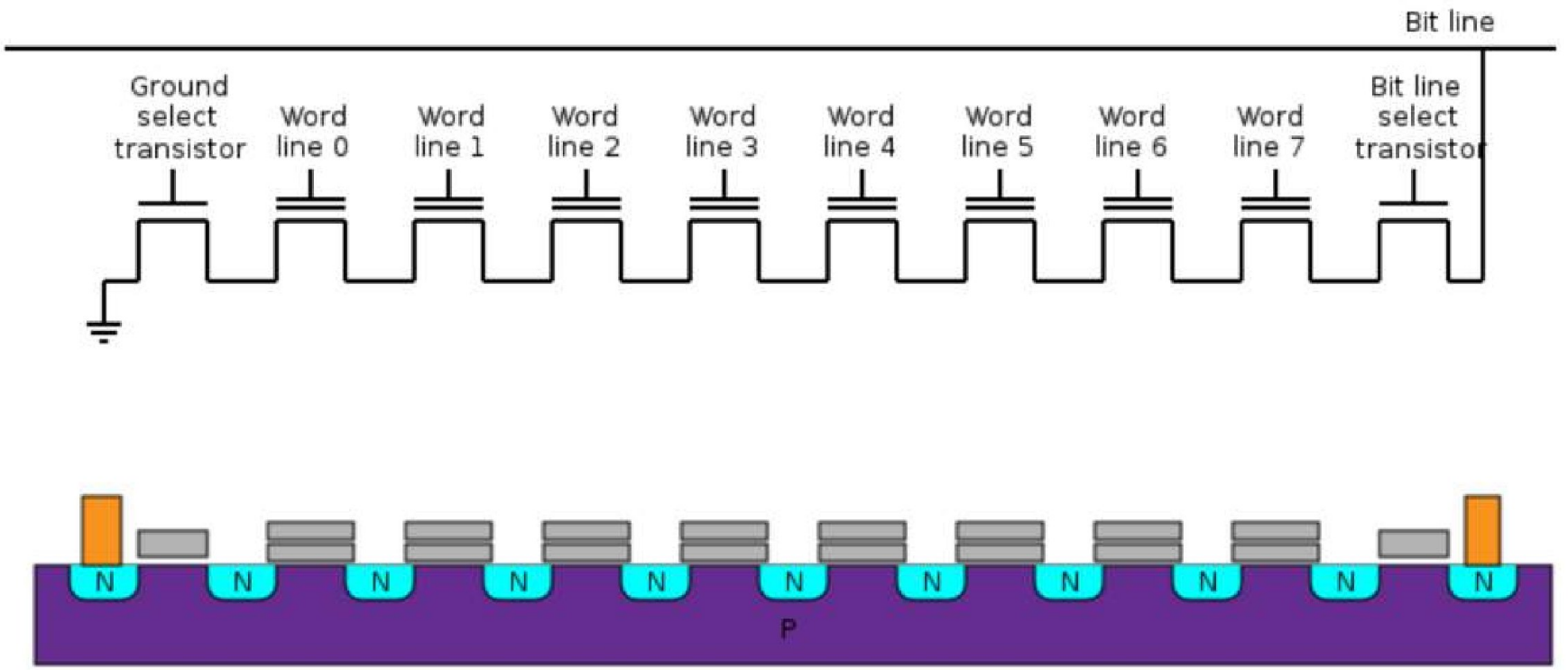
- A sort of **flash memory** (NAND), meant to replace (suplement) HDDs. Same **size**, similar **form factors** (exception M.2), **same interfaces**:
  - **SAS**, 12 Gbit/s;
  - **PCI-e 3.0**, 31 Gbit/s;
- Non-volatile, stores data when losing power.
- Stores data **as electrical charges**.



Advantages:
- fast random read;
- no moving parts;
- no spin-up time;
- less energy consuming;
- silent.

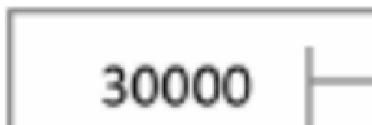
Disadvantages:
- high cost of capacity;
- limited amount of writes;
- faster, no moving parts, but, have **endurance time**: 1-2 years (except Intel's 3D Point);

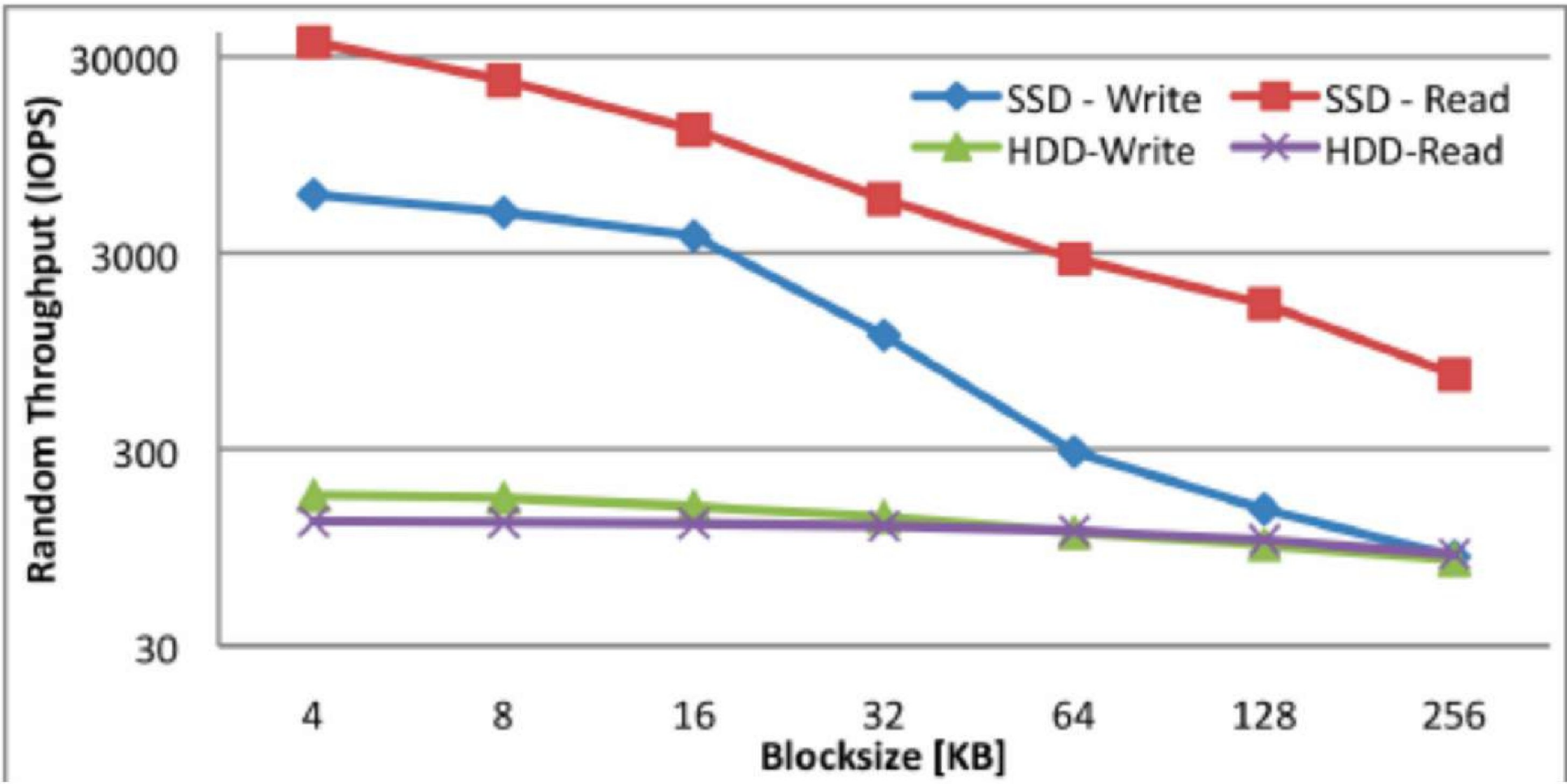
Controller:
- R/W cach
- **bad bloc**
- **wear** leve


Bit line
Ground select transistor
Word line 0
Word line 1
Word line 2
Word line 3
Word line 4
Word line 5
Word line 6
Word line 7
Bit line select transistor
N
P

Advantages:

- fast random read;
- no moving parts;
- no spin-up time;
- less energy consuming;
- silent.



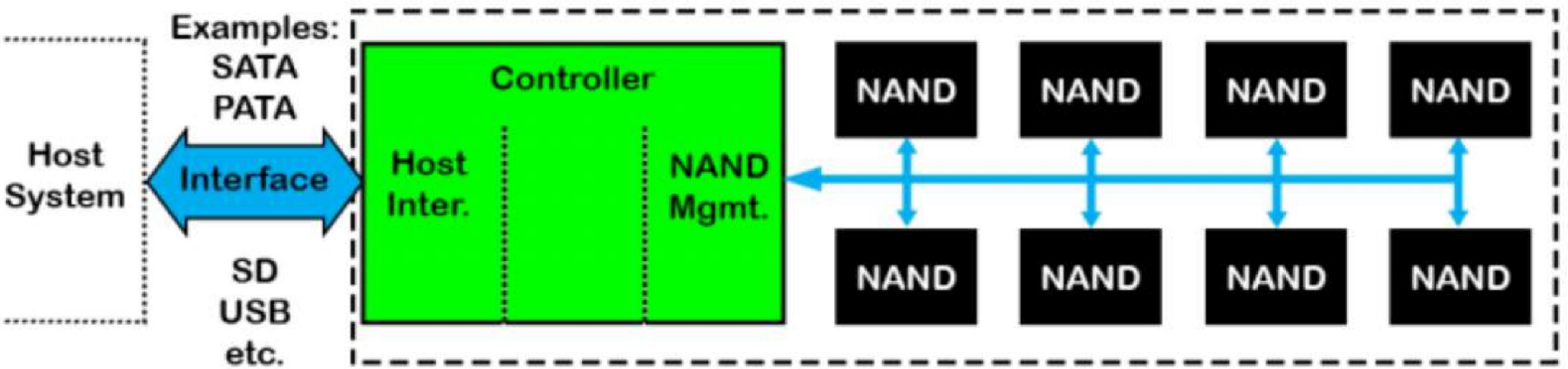

30000
3000
300
30
Random Throughput (IOPS)
SSD - Write
SSD - Read
HDD-Write
HDD-Read
4
8
16
32
64
128
256
Blocksize [KB]

Disadvantages:

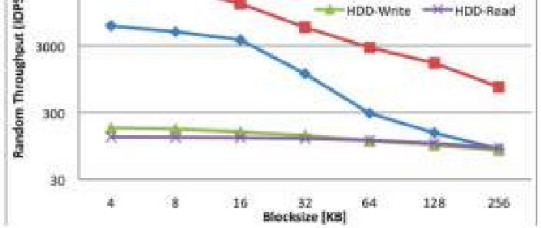
- high cost of capacity;
- limited amount of writes;
- faster, no moving parts, but, have **endurance time**: 1-2 years (except Intel's 3D Point);
- expensive;
- sophisticated controller.

Controller:

- R/W cache + **battery**;
- **bad block** mapping;
- **wear** leveling;

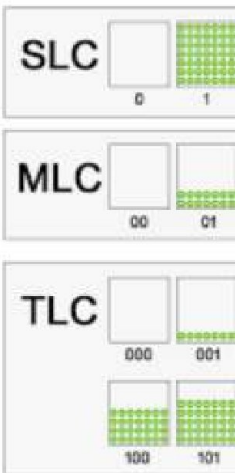


Basic Solid State Drive (SSD) Architecture

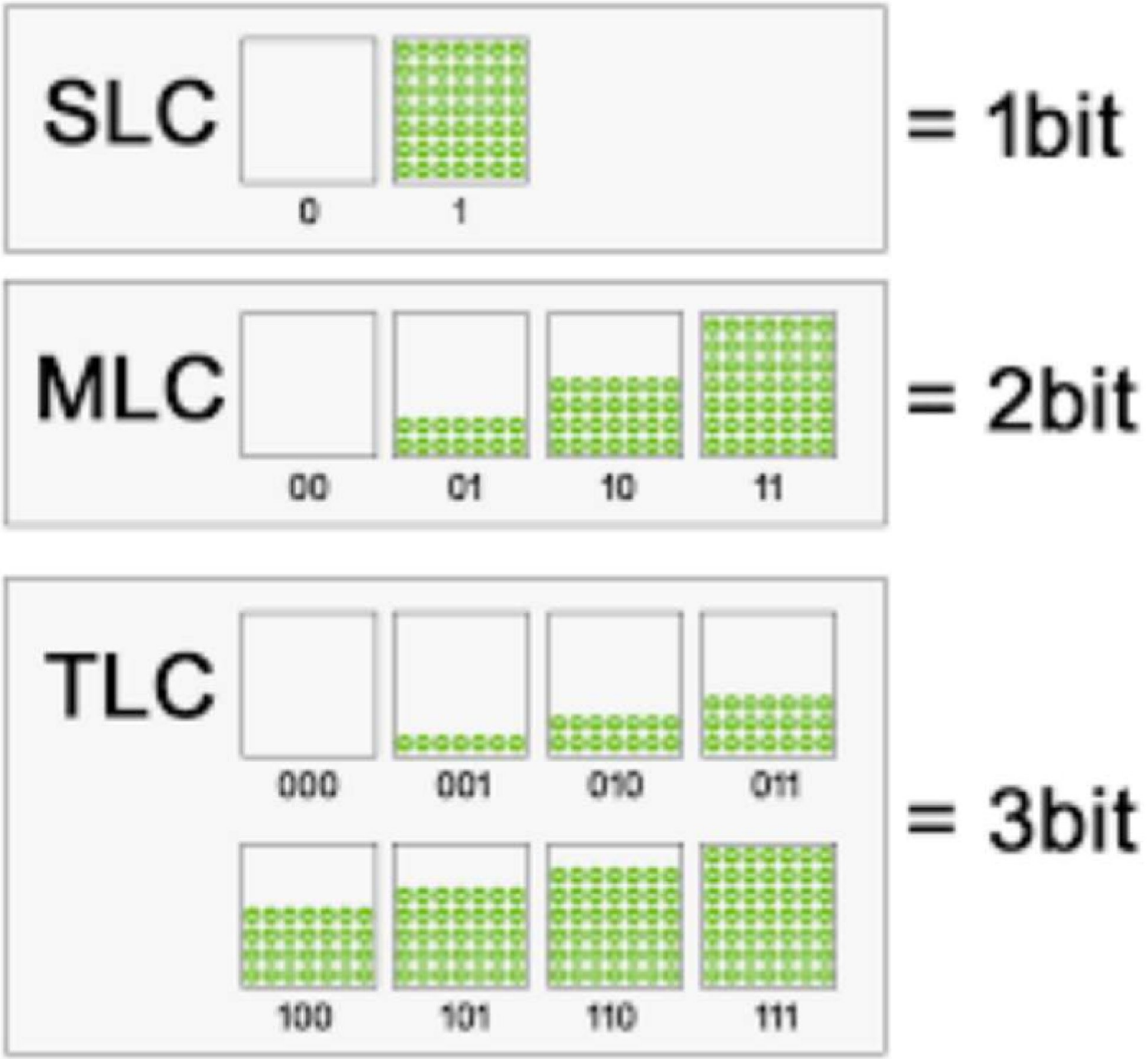


- sophisticated controller.

- Information is stored in **floating gate transistors**, i.e. read and written, holds charge for a long period.
- **Levels** of data storage:
  - single level cell: **1 bit/cell**, fast, endurable (50000-100000 erases), small;
  - 2-level cell: **2 bits/cell**, slower, less endurable (1000-10000 erases), medium;
  - 4-level cell: **4 bits/cell**, slower, less endurable, large.
- **Capacities:** 32 Gbyte - 4 Tbyte.
- Combined HDD + SSD devices.



[4] https://en.wikipedia.or


SLC
0
1
= 1bit
MLC
00
01
10
11
= 2bit
TLC
000
001
010
011
100
101
110
111
= 3bit

Storage structures

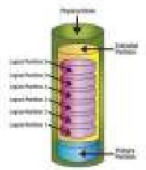
compound storage devices built upon simple storage media

Devices

- Storage components appear as **devices** on operating system level.
- Corresponding drivers (= interface between the kernel and the devices) take care of them.
- Linux kernel: **/dev** files
- major numbers: kernel, SCSI, **sd**, 8, 65, etc.
- minor numbers: used by the driver, for sd it indicates disk and partition (0 disk, 1, part, 2 part, 16 disk, etc);

[5] https://www.oreilly.com/library/view/linux-device-drivers/0596005903/ch03s02.html

Disk partitioning



- Is a process of **slicing the disk** into individual units of variable sizes and purposes.
- Partitions are stored in **partition tables** written on the device:
  - Master Boot Record (MBR) scheme, 16 byte entries;
  - GUID Partition Table (GPT) scheme, 512 byte entries;
- In some cases it can be resized.
- **Important to take care of partition table!**
- **Formatting: creating structures over the devices.**

[6] https://en.wikipedia.org/wiki/Disk_partitioning

RAID

Practice #3: Creating RAID

http://wiki.rcte-it.com/wiki/Direct_Attached_Storage

```
yum install mdadm
mdadm --create --help

mdadm --create /dev/md1 --level=1 \
 --raid-devices=2 /dev/sdb /dev/sdc

cat /proc/mdstat
mdadm --detail /dev/md1
ls -la /dev/md1
lsblk
blkid -t /dev/md1

mdadm --fail /dev/md1 /dev/sdc
mdadm --remove /dev/md1 /dev/sdc
mdadm --add /dev/md1 /dev/sdd
mdadm --add-spare /dev/md1 /dev/sdd

mdadm --stop /dev/md1
mdadm --assemble --scan

mdadm --build /dev/md2 --level=1 \
--raid-devices=2 /dev/sde /dev/sdf
mdadm --stop /dev/md1 /dev/md2
mdadm --assemble --scan

mdadm --zero-superblock /dev/sdd
mdadm --assemble --scan
mdadm --zero-superblock /dev/sdb /dev/sdc

mdadm --create /dev/md2 --level 5 \
--raid-devices 3 /dev/sdb /dev/sdc /dev/sdd
mdadm --detail /dev/md2
mdadm --stop /dev/md2
mdadm --examine --scan

mdadm --add /dev/md2 /dev/sde
mdadm --grow /dev/md2 --raid-devices 4
mdadm --detail /dev/md2

mdadm --fail /dev/md2 /dev/sdd /dev/sdb ...

mdadm --create /dev/md2 --level 5 \
--raid-devices 3 /dev/sdb /dev/sdc /dev/sdd
mdadm --add /dev/md2 /dev/sde
mdadm --grow /dev/md2 --level 6
lsblk /dev/md2

mdadm /dev/md2 --fail /dev/sdd \
--remove /dev/sdd
mdadm /dev/md2 --add /dev/sdd
```

Just a Bunch of Disks (JBoD)

= disk expansion units

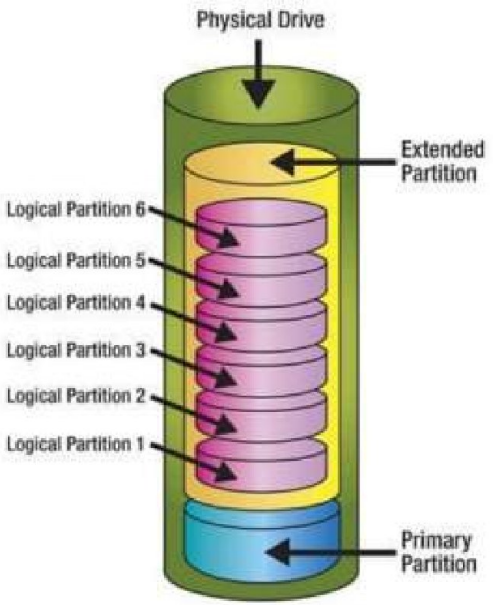
- It offers very simple disk organizing features;
- No significant added value, individual disks are connected through **intelligent expansion units**.
- These units are built up of as follows:
  - basic enclosure;
  - backplane with one or more expander chips;
  - redundant power supply;
  - proper connection, like SAS or SATA;
- Simple operations may apply:
  - concatenation = spanning;
  - mirroring.

Devices

- Storage components appear as **devices** on operating system level.
- Corresponding drivers (= interface between the kernel and the devices) take care of them.
- Linux kernel: **/dev** files
- major numbers: kernel, SCSI, **sd**, 8, 65, etc.
- minor numbers: used by the driver, for sd it indicates disk and partition (0 disk, 1, part, 2 part, 16 disk, etc);

[5] https://www.oreilly.com/library/view/linux-device-drivers/0596000081/ch03s02.html

Disk partitioning



- Is a process of **slicing the disk** into individual units of variable sizes and purposes.
- Partitions are stored in **partition tables** written on the device:
  - Master Boot Record (**MBR**) scheme, 16 byte entries;
  - GUID Partition Table (**GPT**) scheme, 512 byte entries;
- In some cases it can be resized.
- **Important to take care of partition table!**
- **Formatting: creating structures over the devices.**

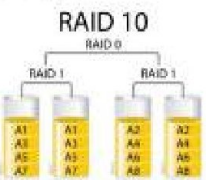
[6] https://en.wikipedia.org/wiki/Disk_partitioning

RAID

- **R**edundant **A**rray of **I**nexpensive/**I**ndependent **D**isks.
- What was expensive? Mainframe vs. PC worlds.
- Spread data across disks. **Single logical disk**.
- Reengineered structure: **Striped data storage**, **parity calculation**.
- Designed for:
  - **speedup** devices;
  - **size** increase;
  - **error recovery**.
- Redundancy: **useful** vs. **useless**.
- RAID levels...
- **Use the corresponding RAID to the purpose!**
- SCSI RAIDs do better. Why?
  - multiple devices on bus;
  - device subdivision, device hierarchy.

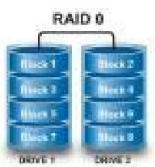
- **Hardware** vs. **software** based;
- **Rebuild** process and rebuild time;
- Limitations:
  - correlated device failures;
  - unrecoverable read errors;
  - failure during rebuild;
  - lack of atomicity of writes;
  - write cache reliability.
- TAPE RAID: mirror and concatenate.

Hot spare!
≠
RAID

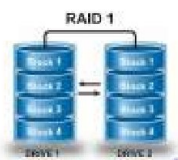


Special RAID levels:
- RAID 1+0: mirrors first, then concatenate;
- RAID 0+1: concatenates first, then mirrors;

RAID0:
- Concatenation and striping disks;
- **Benefit:** speedup.
- **Drawback:** failure of any device causes the whole structure to fail;



RAID1:
- Mirroring, no striping;
- Identically written disks;
- **Benefits:** security, data written 2 places;
- **Drawback:** performance, slower R/W;



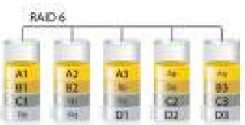
RAID2:
- It has historical relevance;
- Bit-level striping and parity;
- Parity is written on dedicated drive;

RAID3:
- Has no practical relevance.
- Byte-level striping and parity;
- Dedicated parity disk;

RAID4:
- Has no practical relevance.
- Block-level striping and parity;
- Dedicated parity disk;
- Benefits: fast IO;
- Drawback: extra parity disk;

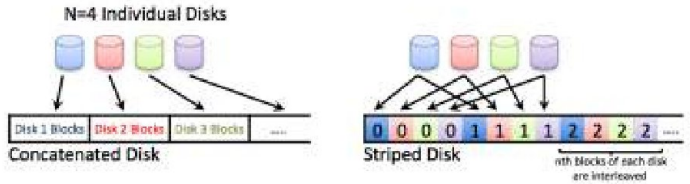
RAID5:
- Most widely used;
- Block-level striping;
- Distributed parity;
- **Benefits:** fast IO, secure for 1 device failure;
- **Drawback:** slow rebuild time;

RAID6:
- Widely used;
- Block-level striping;
- Distributed parities (2);
- **Benefits:** fast IO, secure for 2 devices failure;
- **Drawbacks:** even slower rebuild time, 2 extra devices;



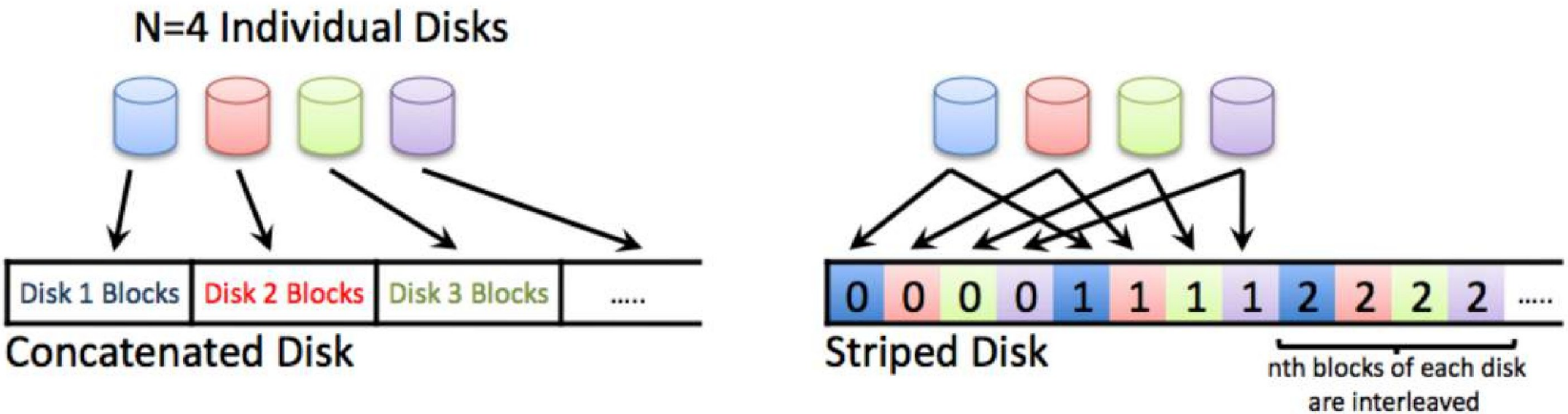
[7] https://docs.fedoraproject.org/en-US/Fedora/14/html/Storage_Administration_Guide/index.html
[8] https://media.techtarget.com/searchNetworking/Downloads/IncidentResponseChapter10.pdf

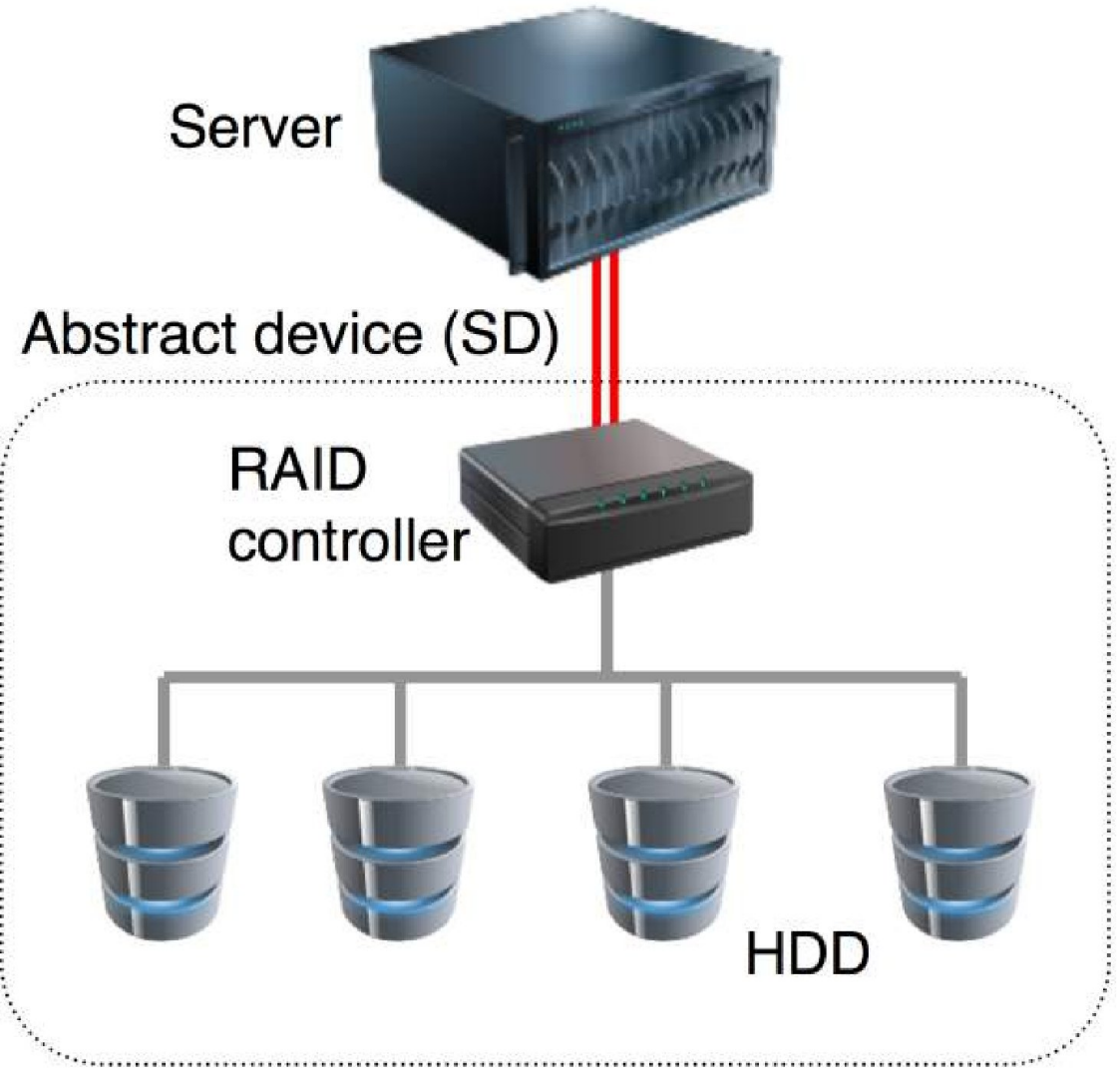
- **R**edundant **A**rray of **I**nexpensive/**I**ndependent **D**isks.
- What was expensive? Mainframe vs. PC worlds.
- Spread data across disks. **Single logical disk**.
- Reengineered structure: **Striped data storage**, **parity calculation**.
- Designed for:
  - **speedup** devices;
  - **size** increase;
  - **error recovery**.
- Redundancy: **useful** vs. **useless**.
- RAID levels...
- **Use the corresponding RAID to the purpose!**
- SCSI RAIDs do better. Why?
  - multiple devices on bus;
  - device subdivision, device hierarchy.

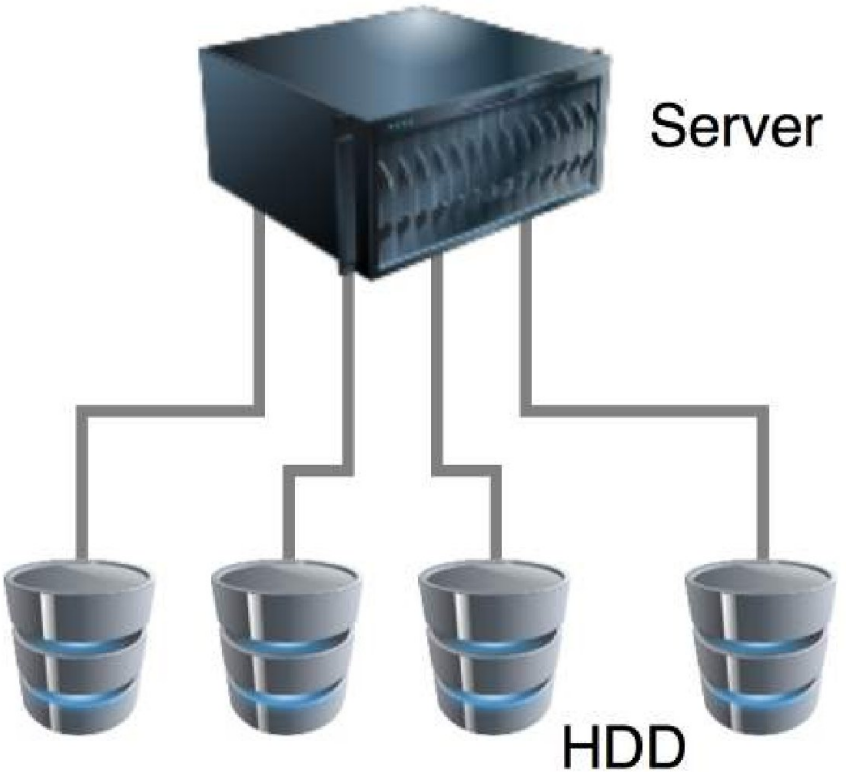


Hot spar
≠
RAID

RAID0: **RAID1:** **RAID2:** **RAID3:** **RAID**


N=4 Individual Disks
Disk 1 Blocks
Disk 2 Blocks
Disk 3 Blocks
.....
Concatenated Disk
0 0 0 0 1 1 1 1 2 2 2 2 .....
Striped Disk
nth blocks of each disk
are interleaved


Server
Abstract device (SD)
RAID
controller
HDD

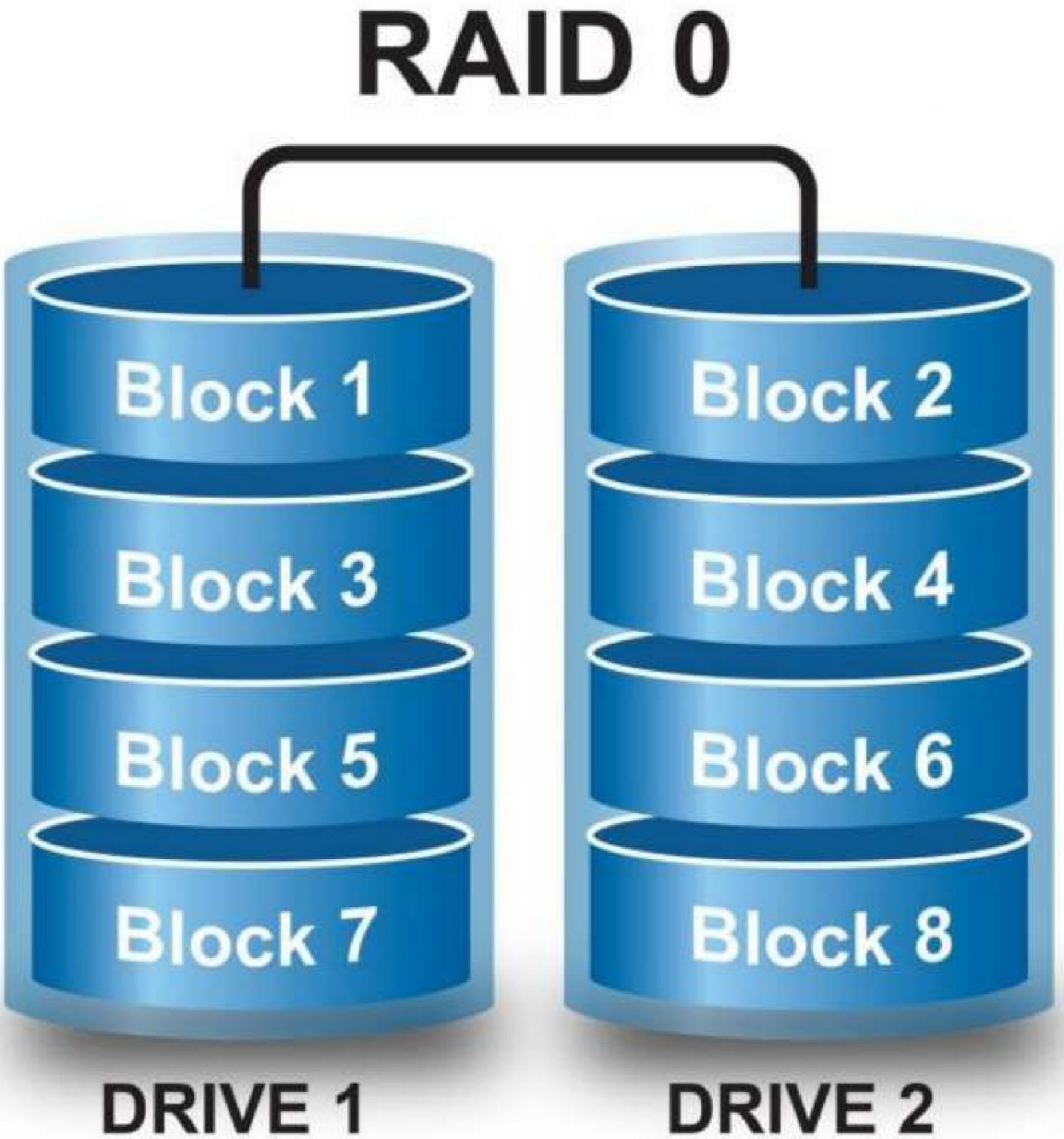

Server
HDD

RAID0:

- **Concatenation** and **striping** disks;
- **Benefit:** speedup.
- **Drawback:** failure of any device causes the whole structure to fail;

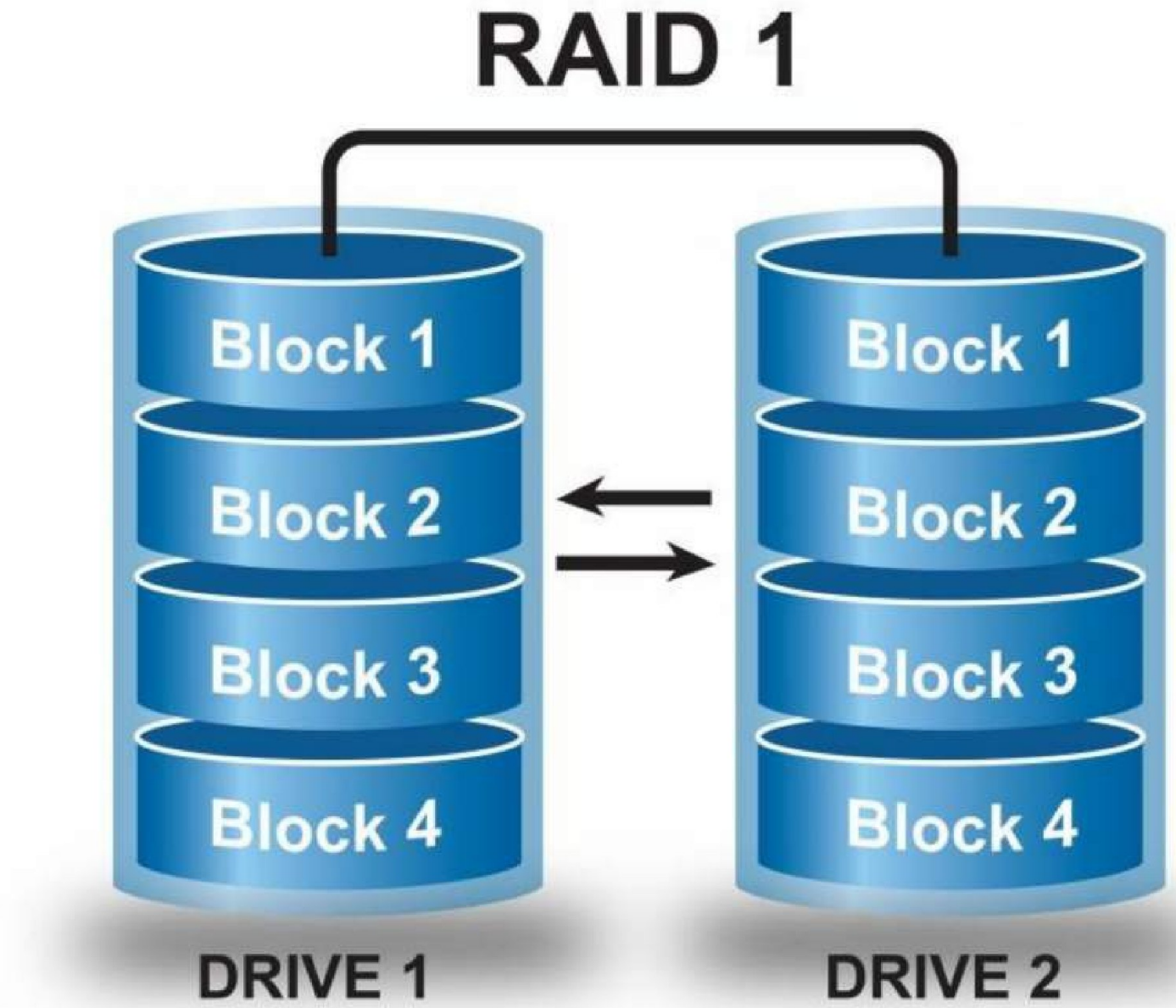
RAID1:

- **Mirr**
- Iden
- **Ben**
  writt
- **Draw**
  slow


RAID 0
Block 1
Block 3
Block 5
Block 7
Block 2
Block 4
Block 6
Block 8
DRIVE 1
DRIVE 2

RAID1:

- **Mirroring**, **no striping**;
- Identically written disks;
- **Benefits:** security, data written 2 places;
- **Drawback:** performance, slower R/W;



Mirrored Data to both Drives

[7] http

striping;
en disks;
rity, data
;
formance,

RAID2:

- It has historical relevance;
- Bit-level striping and parity;
- Parity is written on dedicated drive;

RAID3:

- Has prac relev
- Byte stripi parit
- Dedi disk;

torical
e;
striping
y;
written
ated

RAID3:

- Has no practical relevance.
- Byte-level striping and parity;
- Dedicated parity disk;

RAID4:

- Has no relevanc
- Block-le and **par**
- **Dedicat disk**;
- Benefits
- Drawba parity di

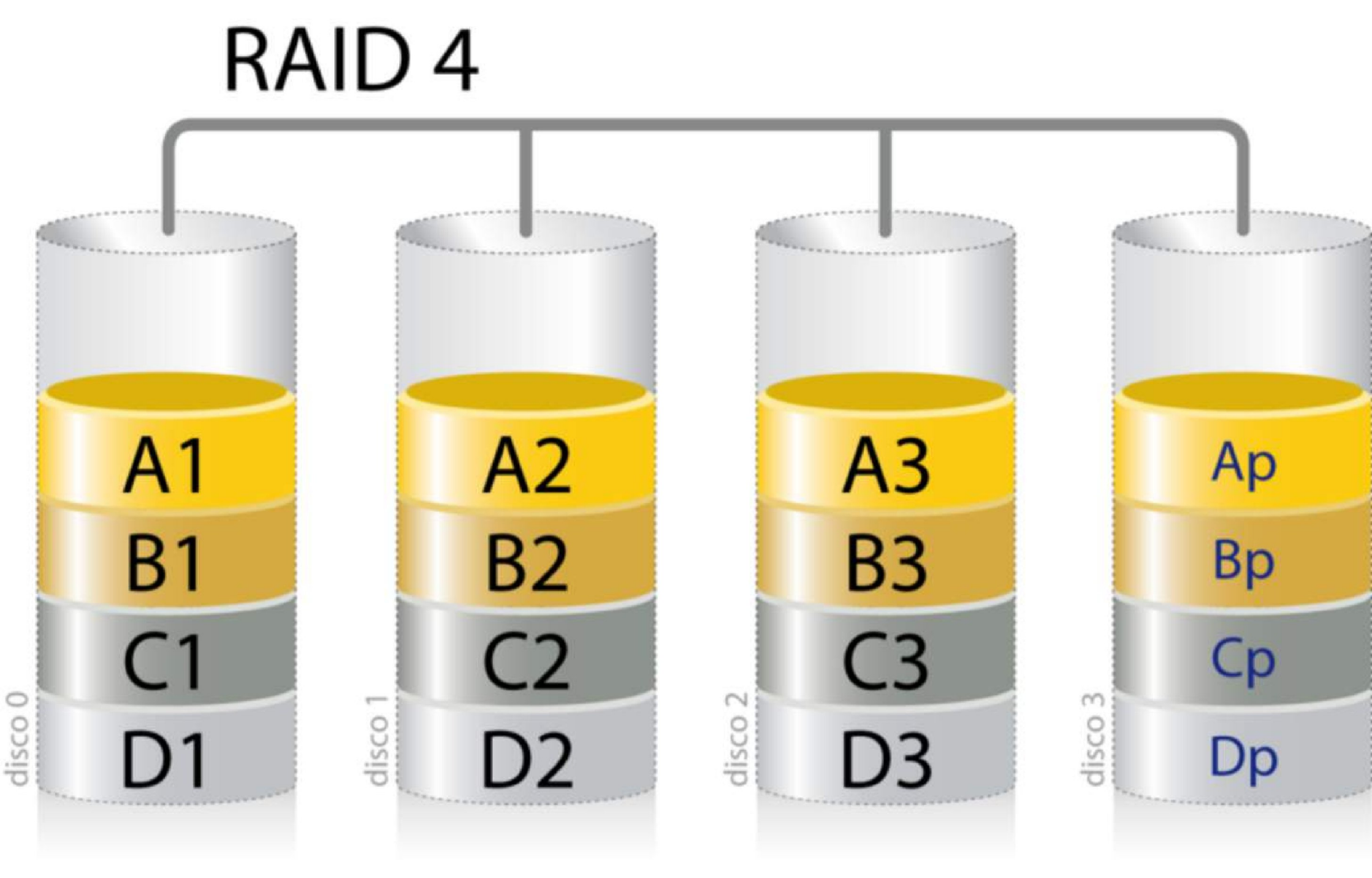
no
ical
ance.
-level
ng and
y;
cated parity

RAID4:

- Has no practical relevance.
- Block-level **striping** and **parity**;
- **Dedicated parity disk**;
- Benefits: fast IO;
- Drawback: extra parity disk;

RAID5:

- Most
- Block
- **Distr**
- **Bene**
  secur
  failure
- **Draw**
  rebuil


RAID 4
A1
B1
C1
D1
A2
B2
C2
D2
A3
B3
C3
D3
Ap
Bp
Cp
Dp
disco 0
disco 1
disco 2
disco 3

cal

iping

ity

O;

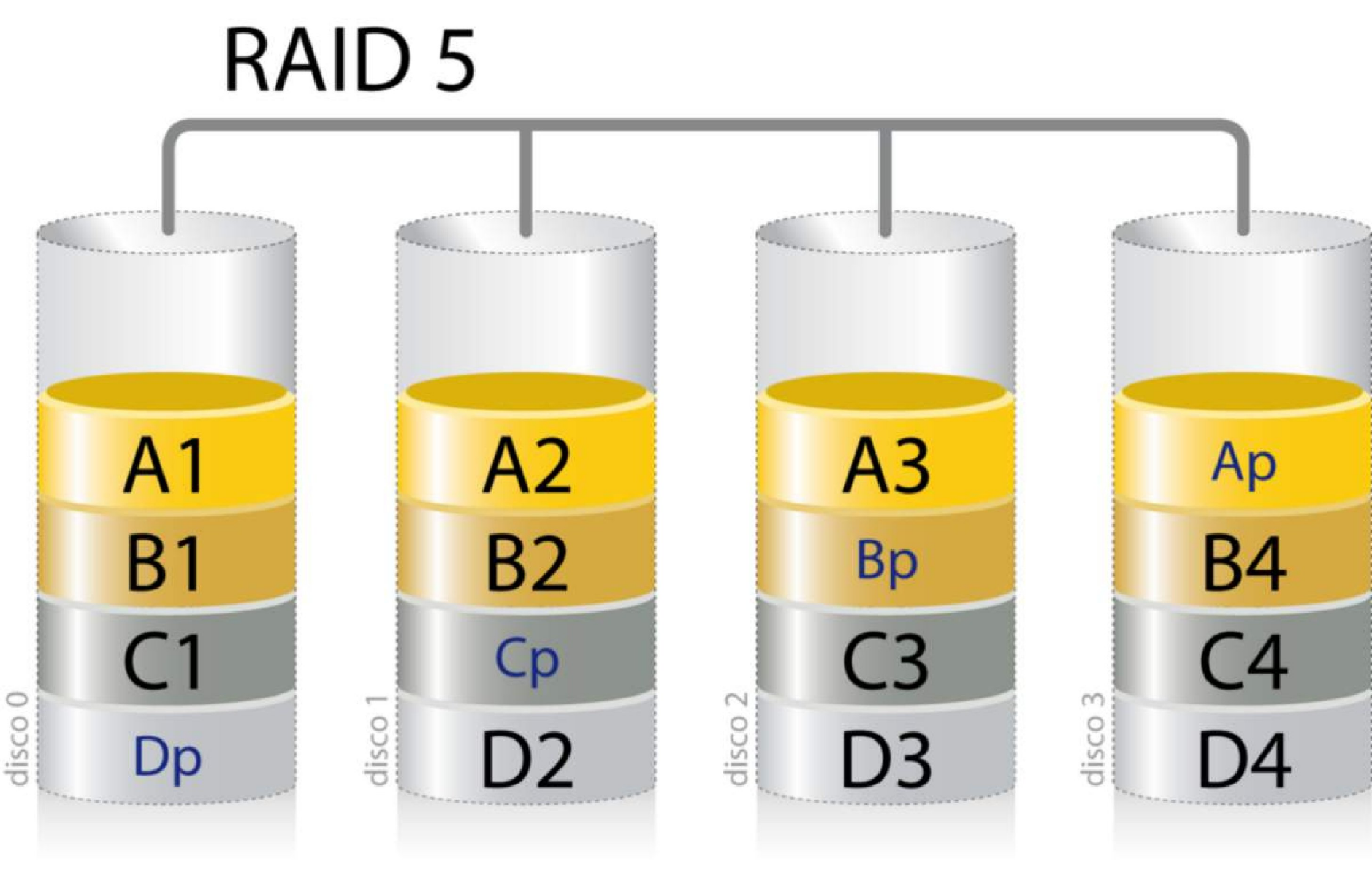
ra

RAID5:

- Most widely used;
- Block-level **striping**;
- **Distributed parity**;
- **Benefits:** fast IO, secure for 1 device failure;
- **Drawback:** slow rebuild time;

RAID6:

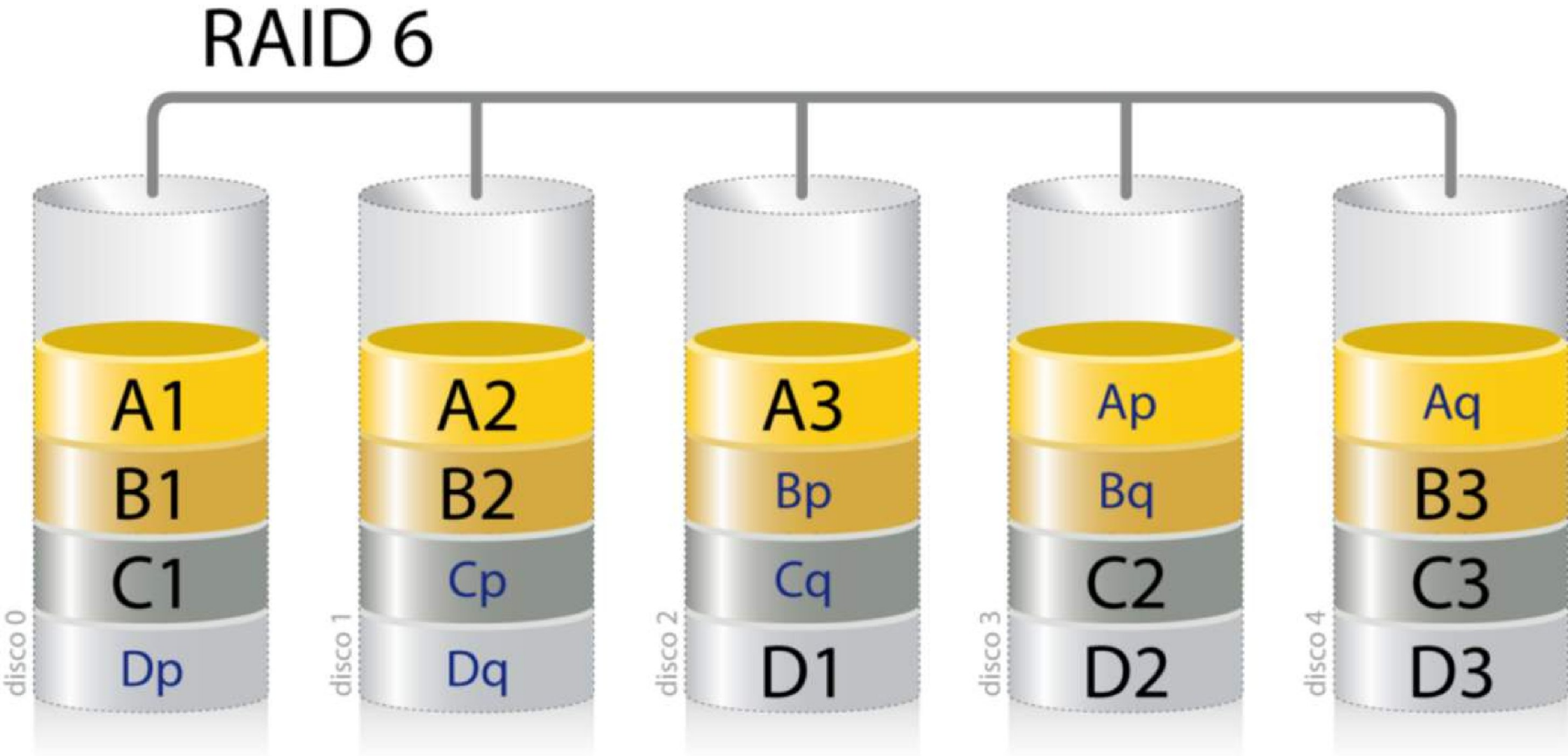
- Wid
- Bloc
- **Dist**
- **Ben**
  secu
  failu
- **Dra**
  slow
  extra


RAID 5
A1
B1
C1
Dp
A2
B2
Cp
D2
A3
Bp
C3
D3
Ap
B4
C4
D4
disco 0
disco 1
disco 2
disco 3

used;
triping;
parity;
st IO,
device
slow

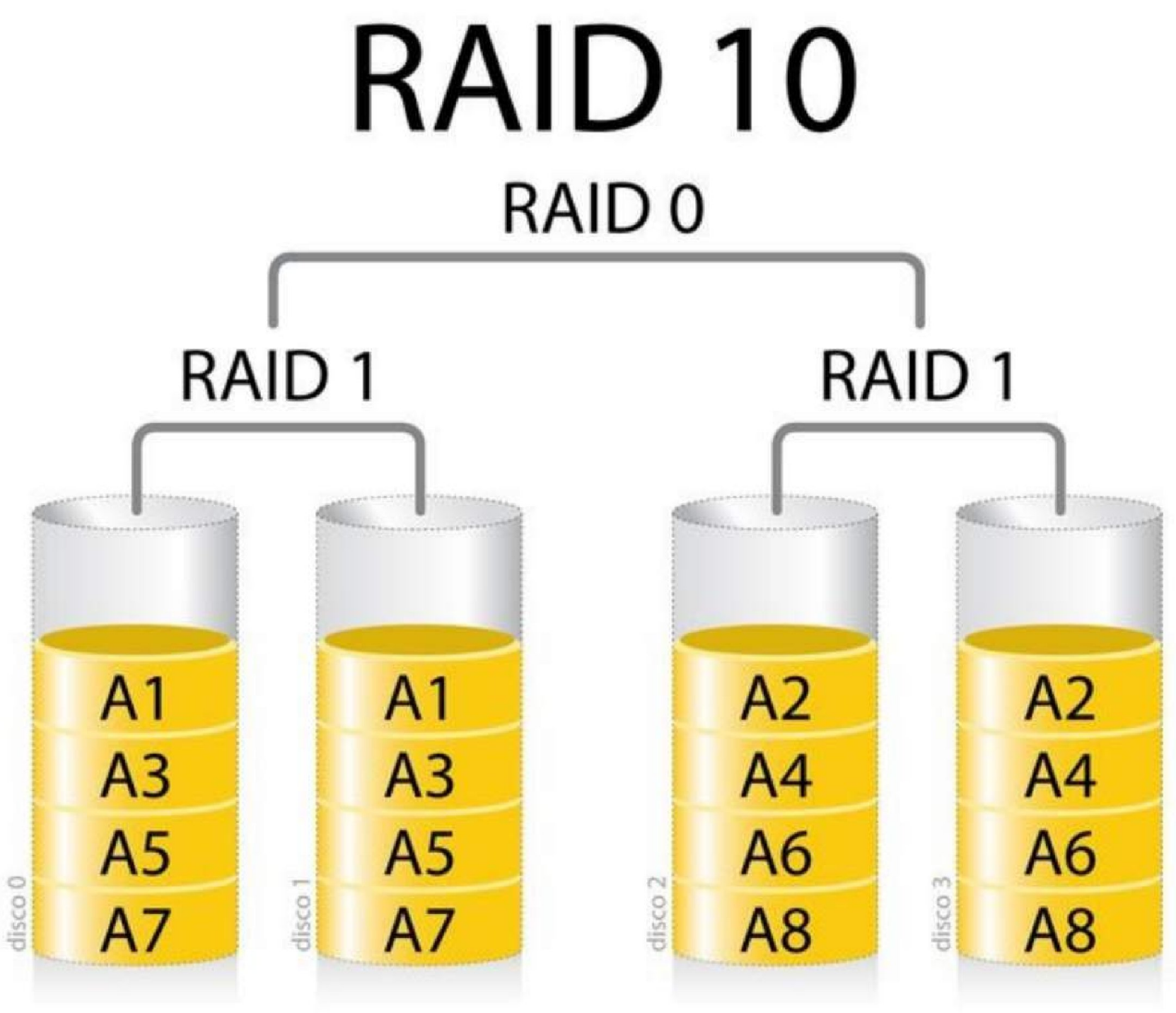
RAID6:

- Widely used;
- Block-level **striping**;
- **Distributed parities (2)**;
- **Benefits:** fast IO, secure for 2 devices failure;
- **Drawbacks:** even slower rebuild time, 2 extra devices;


RAID 6
A1
B1
C1
Dp
disco 0
A2
B2
Cp
Dq
disco 1
A3
Bp
Cq
D1
disco 2
Ap
Bq
C2
D2
disco 3
Aq
B3
C3
D3
disco 4

Special RAID levels:

- RAID 1+0: mirrors first, then concatenate;
- RAID 0+1: concatenates first, then mirrors;


RAID 10
RAID 0
RAID 1
RAID 1
A1
A3
A5
A7
A1
A3
A5
A7
A2
A4
A6
A8
A2
A4
A6
A8
disco 0
disco 1
disco 2
disco 3

Hot spare!

≠

RAID

Where configuration is stored?

- **configuration files** (/etc/mdadm.conf);
- **superblocks**
  - 256 Byte;
  - stores metadata (RAID level, e.g.)

https://raid.wiki.kernel.org/index.php/RAID_superblock_fo

Storage architectures

the way we construct network of elementary storage devices and storage structures

Direct Attached Storage (DAS)

- Not a network!
- Storage device **directly connected to** and **exclusively used by** the host;
- Most of the protocols are used (SCSI, SAS, ATA, SATA, FC, USB, etc.);
- It is not necessary to be a small device! E.g. SUN 10000 + DAS;
- External or internal;
- Smallest latency, closest to server, but limited complexity.

SAN

Storage Area Networks (SAN)

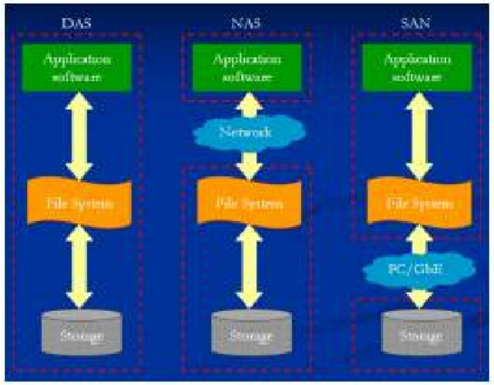
a network that provides access to block level devices

Components

https://arkit.co.in/san-switch-basic/
https://en.wikipedia.org/wiki/Storage_area_network
https://slideplayer.com/slide/5117247/

Network Attached Storage (NAS)

file level data storage



Unified Storage

offers FCoE, iSCSI, NFS, CIFS in the same box

Performance difference!

https://www.computerweekly.com/feature/Unified-Storage-FAQ
http://www.eternus-dx.com/eternus-overview/unified-storage/

Practice#4: Take a look at SAN devices

Case study: measuring Eternus DX200

show enclosure-status -type all
show hardware-information
show fru-ce -type cm0
show disks
show raid-groups
show eco-mode
show volumes
set volume -volume-name OEVOL01
show fc-parameters
show iscsi-parameters
show host-wwn-names
show host-sas-addresses
test iscsi-ping -port 10 -ip 192.168.2.10 -count 2

show network
show smi-s
show storage-system-name
show raid-tuning
show cache-parameters
show extreme-cache

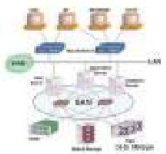
Direct Attached Storage (DAS)

- **Not a network!**
- Storage device **directly connected to** and **exclusively used by** the host;
- Most of the protocols are used (SCSI, SAS, ATA, SATA, FC, USB, etc.);
- It is not necessary to be a small device! E.g. SUN 10000 + DAS;
- External or internal;
- Smallest latency, closest to server, but limited complexity.

SAN

Storage Area Networks (SAN)

a network that provides access to block level devices



- Basic purpose: **connects servers to disk and tape devices**;
- OS senses devices as if they were directly attached;
- Offers **only block level** operations, no higher structures;
- From 2000s, developed from the mainframe world where multiple servers were connected with storage devices, eliminating Single Point of Faliure;
- **Separate network** from LAN: own topology and devices;
- Hosts do not own resources! **Conflicts.**
- As shared device QoS is needed:
  - bandwidth;
  - latency.

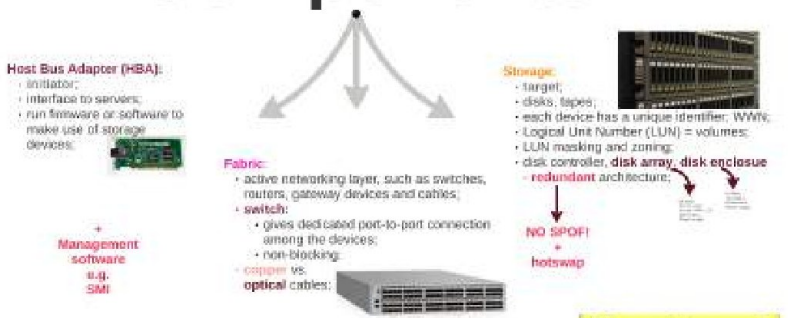
Major SAN types:

- FC-SAN;
- Ethernet-SAN;

SAN variations:

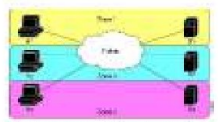
- in-band: data + control;
- out-of-band: data || control (Ethernet)

Components



Zoning:

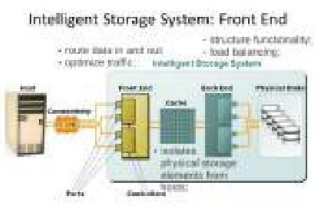
- **soft** - implemented by software;
- hard - implemented by hardware;
- WWN - address based;
- port - port based;



Logical Unit Number (LUN): individually addressable R/W device, volume

- logical disk units;
- tapes;

Controller, Target, Disk, Slice nomenclature;



https://arkit.co.in/san-switch-basic/
https://en.wikipedia.org/wiki/Storage_area_network
https://slideplayer.com/slide/1517247/

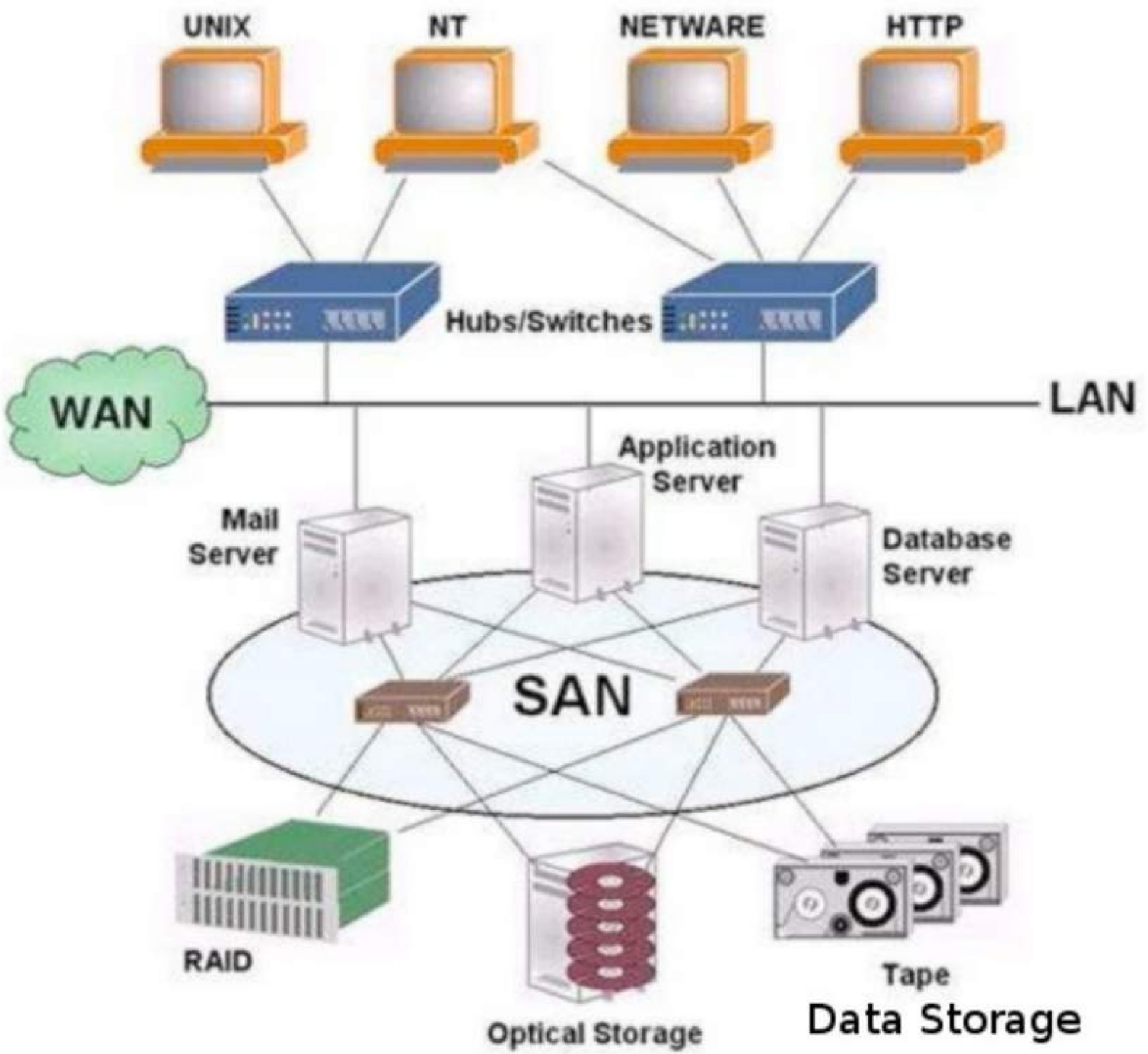
- Basic purpose: **connects servers to disk and tape devices**;
- OS senses devices as if they were directly attached;
- Offers **only block level** operations, no higher structures;
- From 2000s, developed from the mainframe world where multiple servers were connected with storage devices, eliminating Single Point of Faliure;
- **Separate network** from LAN: own topology and devices;
- Hosts do not own resources! **Conflicts.**
- As shared device QoS is needed:
  - bandwidth;
  - latency.

Major SAN types:

- FC-SAN;

SAN variatio

- in-band: d
- out-of-ba
  (Ethernet


UNIX
NT
NETWARE
HTTP
Hubs/Switches
WAN
LAN
Application Server
Mail Server
Database Server
SAN
RAID
Optical Storage
Tape
Data Storage

Components

Host Bus Adapter (HBA):
- initiator;
- interface to servers;
- run firmware or software to make use of storage devices;

+
Management software
e.g.
SMI

Fabric:
- active networking layer, such as switches, routers, gateway devices and cables;
- **switch:**
  - gives dedicated port-to-port connection among the devices;
  - non-blocking;
- copper vs. **optical** cables;

Storage:
- target;
- disks, tapes;
- each device has a unique identifier: WWN;
- Logical Unit Number (LUN) = volumes;
- LUN masking and zoning;
- disk controller, **disk array**, **disk enclosue**
  - **redundant** architecture;

NO SPOF!
+
hotswap

- Chassis;
- Controller(s);
- Cache: RAM + NV;
- Enclosures;
- Power supply;

- Chassis;
- Backplane;
- Connectors;
- Power supply;

tions:

- HBAs;
- fabrics;
- disks + structures;

Host Bus Adapter (HBA):

- initiator;
- interface to servers;
- run firmware or software to make use of storage devices;

Fabric:

- active networking layer, such as switches, routers, gateway devices and cables;
- **switch:**
  - gives dedicated port-to-port connection among the devices;
  - non-blocking;
- **copper** vs. **optical** cables;

Storage:

- **target**;
- disks, tapes;
- each device has a unique identifier: WWN;
- Logical Unit Number (LUN) = volumes;
- LUN masking and zoning;
- disk controller, **disk array**, **disk enclosue**
  - **redundant** architecture;

- Chassis;
- Controller(s);
- Cache: RAM + NV;
- Enclosures;
- Power supply;

- Chassis;
- Backplane;
- Connectors;
- Power supply;


NO SPOF!
+
hotswap

y and

e.g.
SMI

optical ca

SAN variations:

- in-band: data + control;
- out-of-band: data || control (Ethernet)

Z

Intelligent Storage System: Front End

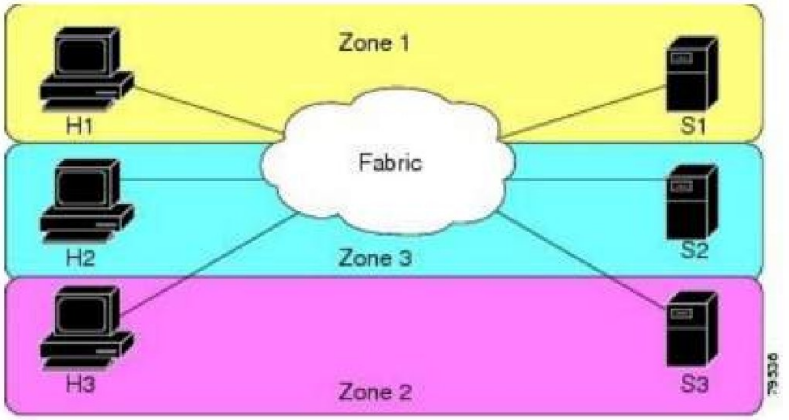
- structure functionality;

Major SAN types:

- FC-SAN;
- Ethernet-SAN;

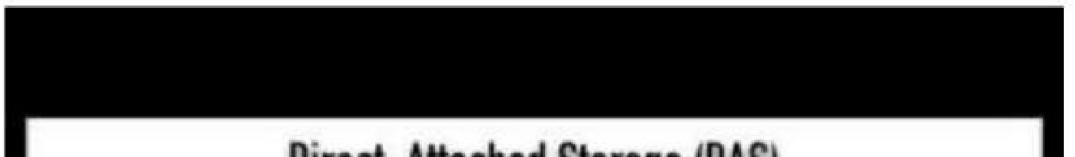
- HBAs;
- fabrics;
- disks + structures;

creating logical groups of devices



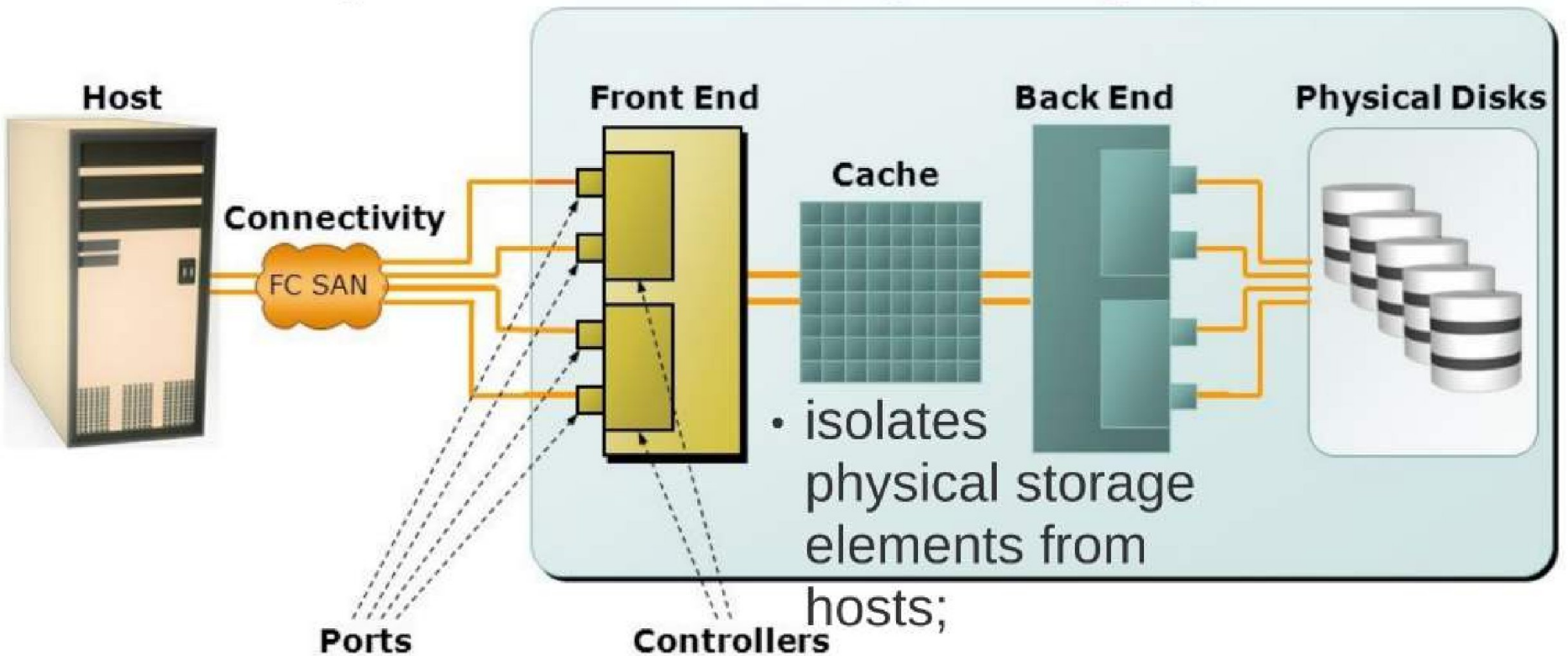
Zoning:

- **soft** - implemented by software;
- hard - implemented by hardware;
- WWN - address based;
- port - port based;



Intelligent Storage System: Front End

- route data in and out;
- optimize traffic;
- structure functionality;
- load balancing;



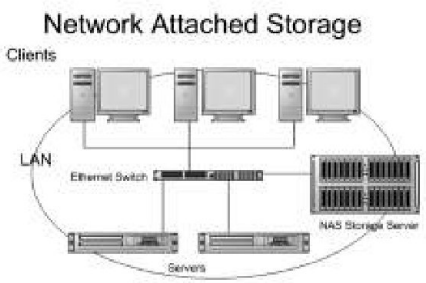
Network Attached Storage (NAS)

file level data storage

- History began in 1980s with file system sharing protocols: NFS, Novell;
- Network appliances with similar internal structure as SANs (disk pools, storage structures, volumes, etc.);
- Client sees it as a **file server**;
- Allow file system access: NFS, SMB/CIFS, Novell FS, SFTP, etc.;
- Requires simple IP/Ethernet network;
- Offer **some sort of redundancy**: power supply, controllers, connections, etc;
- **Clustered versions**: distributed data and metadata storage on multiple NAS'es, because
  - extra processing power can be added;
  - device failures will not disrupt the system.
- Can be rack mounted or standalone;

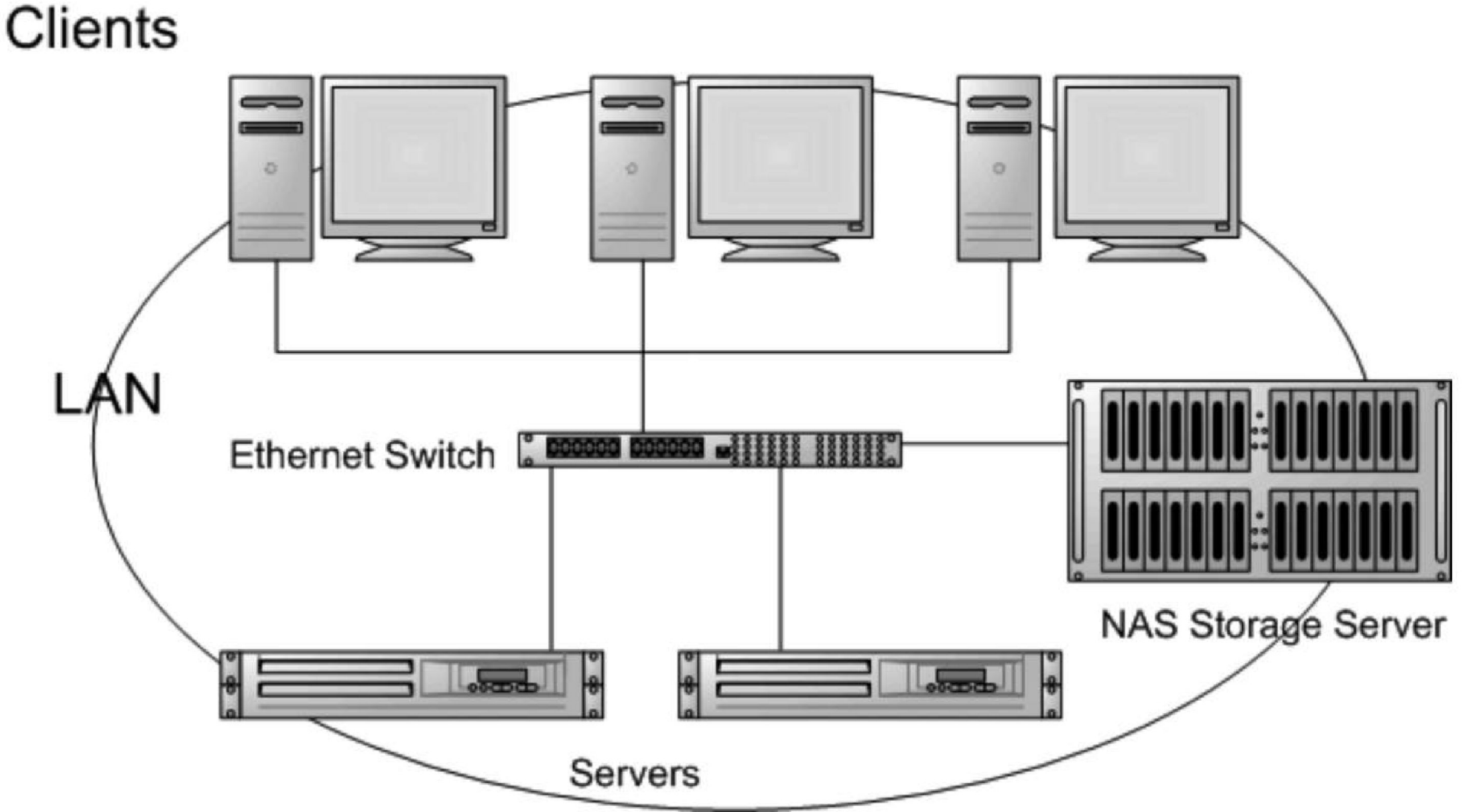
NAS management:

- Similar to that of SAN's: create structures, such as RAID groups, RAIDs;
- Create **NAS volumes**: user or backup (creating FS structures take longer time);
- Access control:
  - users, groups, access lists, shares, (can be taken from auth servers);
  - standards based access control (POSIX, Windows ACL);




Seagate NAS
BlackArmor

Network Attached Storage



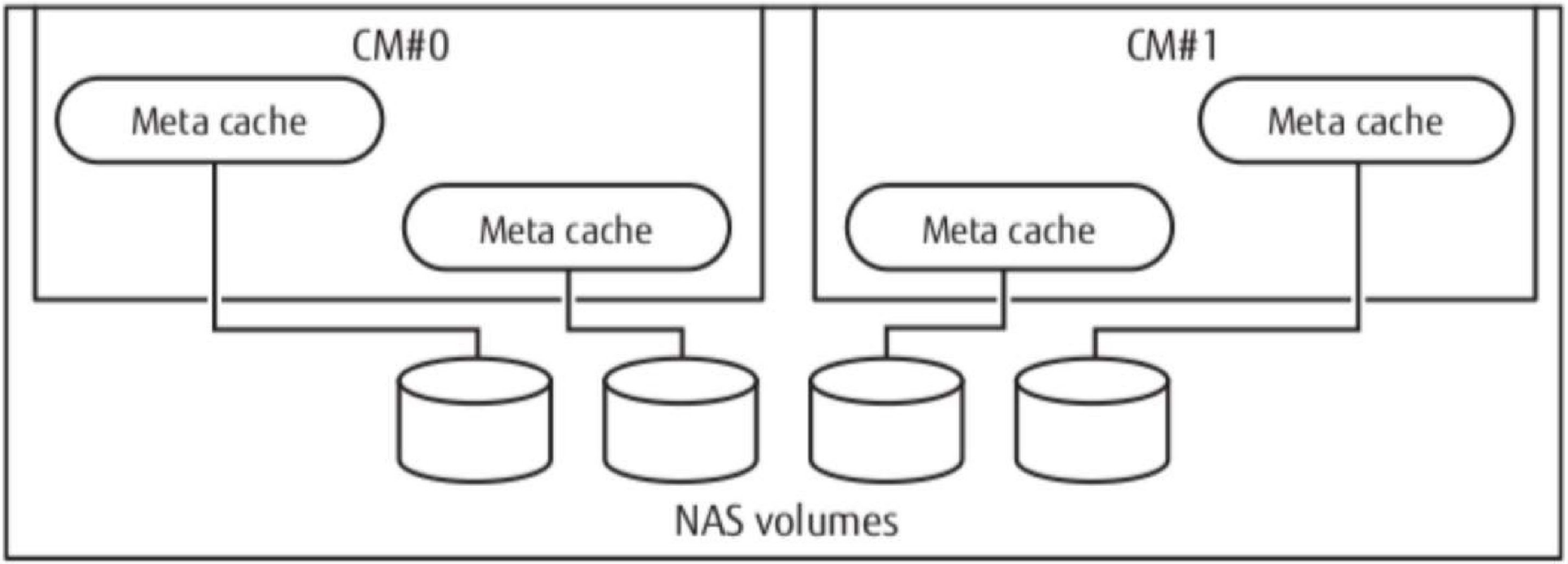
NAS management:

- Similar to that of SAN's: create structures, such as RAID groups, RAIDs;
- Create **NAS volumes**: user or backup (creating FS structures take longer time);
- Access control:
  - **users, groups, access lists, shares,** (can be taken from auth servers);
  - standards based access control (POSIX, Windows ACL);

NAS management:

- Similar to that of SAN's: create structures, such as RAID groups, RAIDs;
- Create **NAS volumes**: user or backup (creating FS structures take longer time);
- Access control:
  - **users, groups, access lists, shares,** (can be taken from auth servers);
  - standards based access control (POSIX, Windows ACL);

NAS caches: **metadata** + **larger** + **write-back on sync requests**.



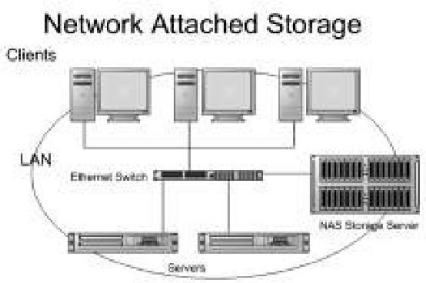
Network Attached Storage (NAS)

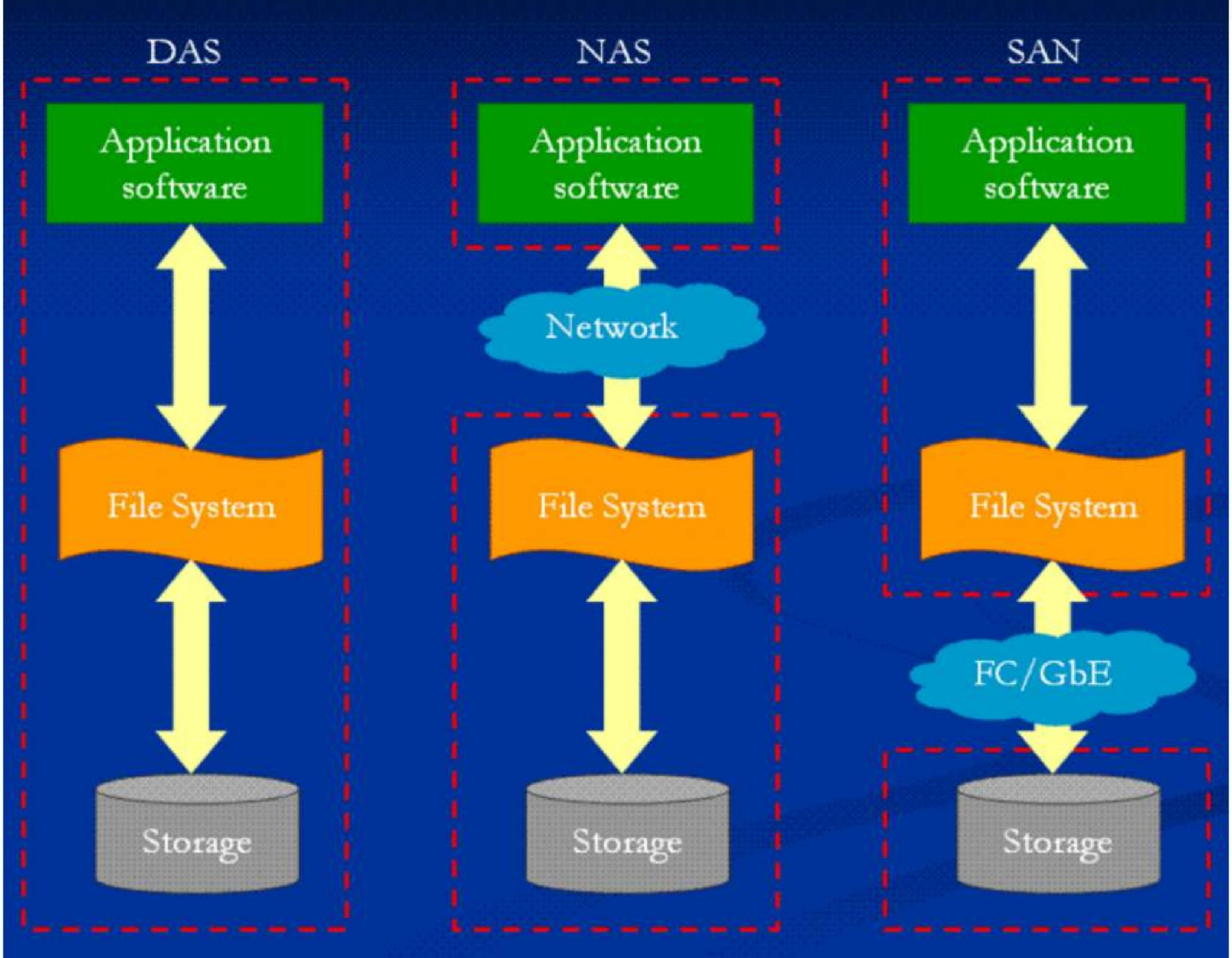
file level data storage

- History began in 1980s with file system sharing protocols: NFS, Novell;
- Network appliances with similar internal structure as SANs (disk pools, storage structures, volumes, etc.);
- Client sees it as a **file server**;
- Allow file system access: NFS, SMB/CIFS, Novell FS, SFTP, etc.;
- Requires simple IP/Ethernet network;
- Offer **some sort of redundancy**: power supply, controllers, connections, etc;
- **Clustered versions**: distributed data and metadata storage on multiple NAS'es, because
  - extra processing power can be added;
  - device failures will not disrupt the system.
- Can be rack mounted or standalone;

NAS management:

- Similar to that of SAN's: create structures, such as RAID groups, RAIDs;
- Create **NAS volumes**: user or backup (creating FS structures take longer time);
- Access control:
  - users, groups, access lists, shares, (can be taken from auth servers);
  - standards based access control (POSIX, Windows ACL);




DAS
NAS
SAN
Application software
Application software
Application software
Network
File System
File System
File System
FC/GbE
Storage
Storage
Storage

Unified Storage

offers FCoE, iSCSI, NFS, CIFS in the same box

Conceptual difference:

- **block:**
  - physical data concepts referring to the organization of data on disk drives, i.e. tracks, sectors;
  - shorter access times, shorter data messages;
- **files:**
  - logical data concepts made up of many blocks;
  - more random data;
  - longer access times, longer data messages;

Transformation is done by file systems!

Performance difference!

Variations:

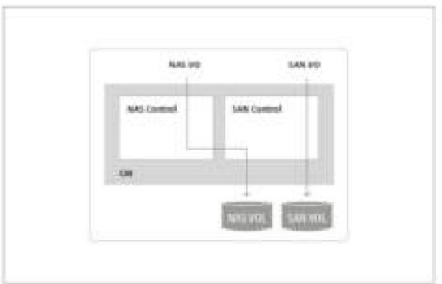
- SAN blocks sit on top of shared devices. Translation is done by storage controller on shared Thin Provisioning Volumes, TPVs.
- NAS gateway attached to SANs.
- Separate controllers for SAN and NAS functions.
- **Shared controllers to manage SAN and NAS:**
  - Ethernet as carrier;
  - FCoE, iSCSI, NFS, CIFS protocols;
- STaaS cloud storage, combines on-premises and cloud storage.

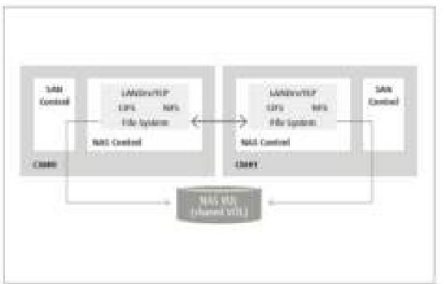
Advantages:

- Save space, device, management efforts, network costs;
- leverages utilization between NAS and SAN;
- **Conform to STaaS providers;**

Disadvantages:

- Performance penalties;





https://www.computerweekly.com/feature/Unified-Storage-FAQ
http://www.eternus-dx.com/eternus-overview/unified-storage/

Performa
differenc

Conceptual difference:

- **block:**
  - physical data concepts referring to the organization of data on disk drives, i.e. tracks, sectors;
  - shorter access times, shorter data messages;
- **files:**
  - logical data concepts made up of many blocks;
  - more random data;
  - longer access times, longer data messages;

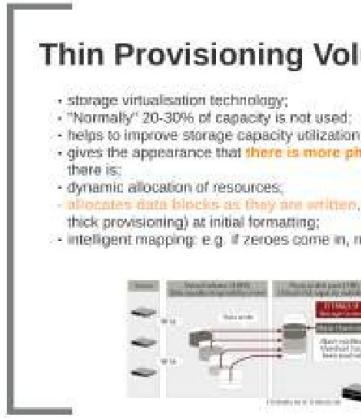
Transformation is done by file systems!

ce

!

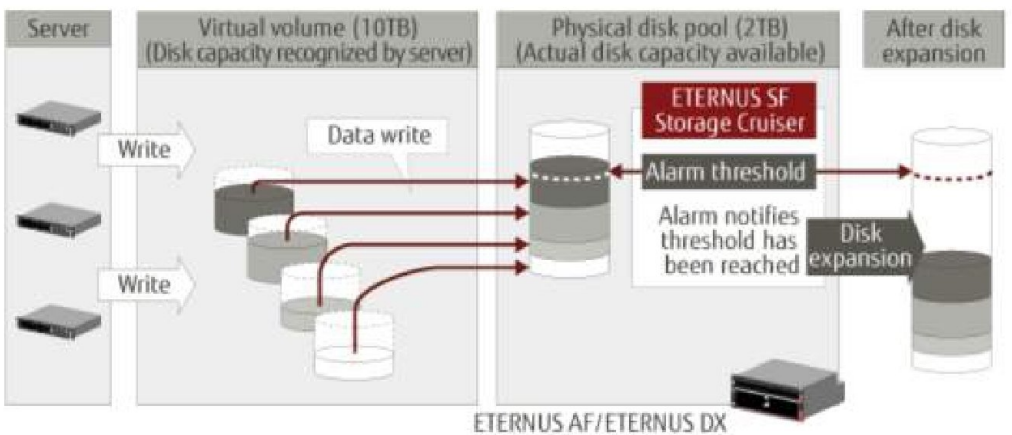
Variations:

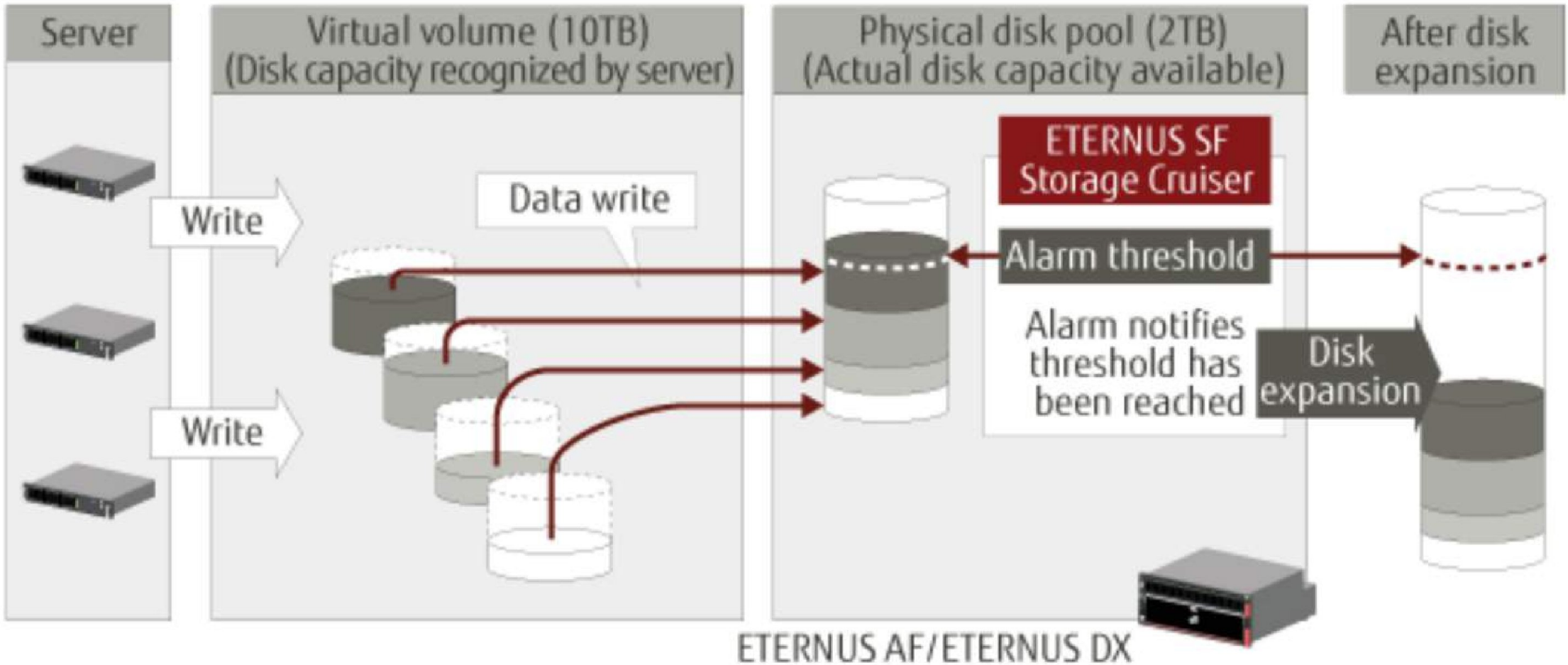
- SAN blocks sit on top of shared devices. Translation is done by storage controller on shared Thin Provisioning Volumes, **TPV**s.
- NAS gateway attached to SANs.
- Separate controllers for SAN and NAS functions.
- **Shared controllers to manage SAN and NAS**:
  - Ethernet as carrier;
  - FCoE, iSCSI, NFS, CIFS protocols;
- STaaS cloud storage, combines on-premises and cloud storage.



Thin Provisioning Volumes

- storage virtualisation technology;
- "Normally" 20-30% of capacity is not used;
- helps to improve storage capacity utilization;
- gives the appearance that **there is more physical resource** as there is;
- dynamic allocation of resources;
- **allocates data blocks as they are written**, not in advance (i.e. thick provisioning) at initial formatting;
- intelligent mapping: e.g. if zeroes come in, no space is allocated;



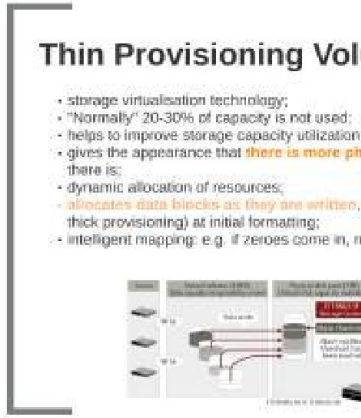

Server
Virtual volume (10TB)
(Disk capacity recognized by server)
Physical disk pool (2TB)
(Actual disk capacity available)
After disk expansion
Write
Write
Data write
ETERNUS SF
Storage Cruiser
Alarm threshold
Alarm notifies threshold has been reached
Disk expansion
ETERNUS AF/ETERNUS DX

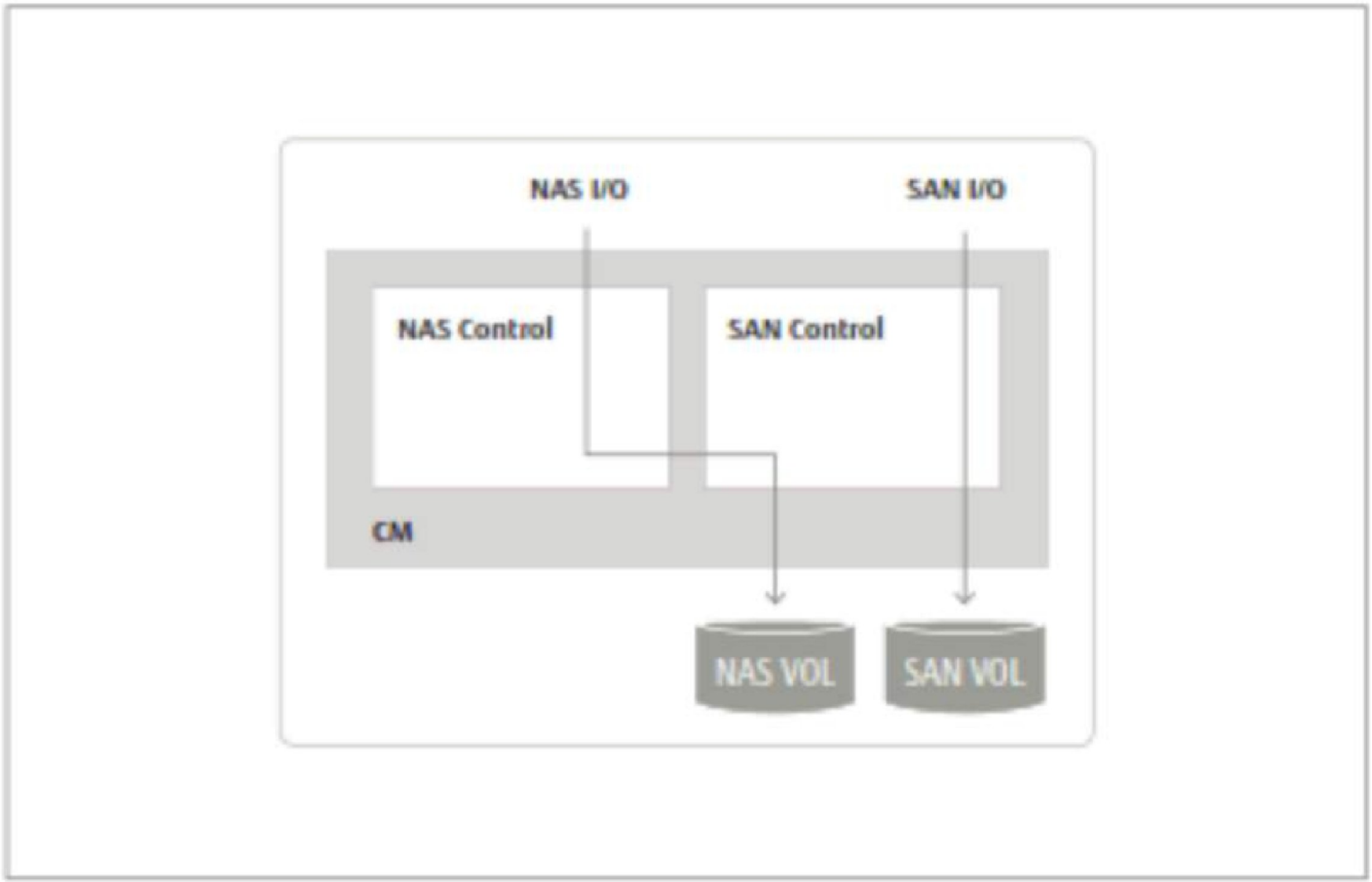
ce

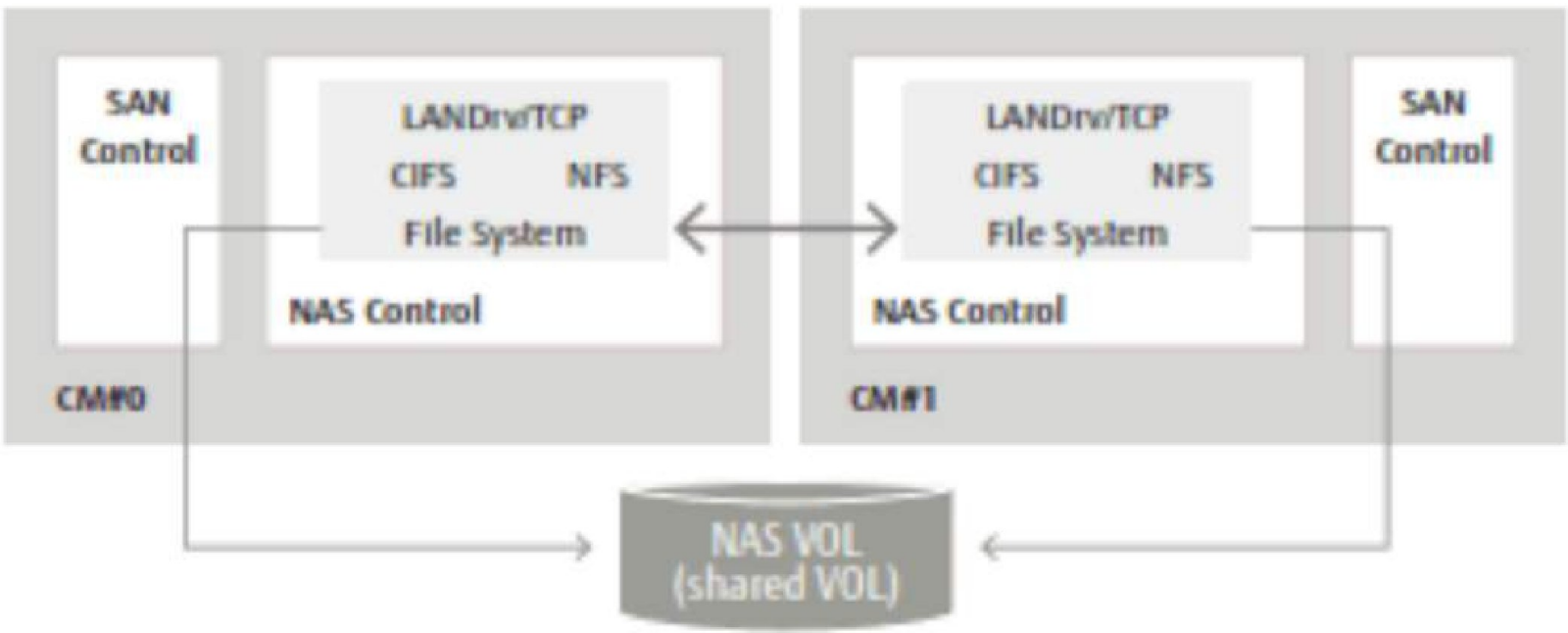
!

Variations:

- SAN blocks sit on top of shared devices. Translation is done by storage controller on shared Thin Provisioning Volumes, **TPV**s.
- NAS gateway attached to SANs.
- Separate controllers for SAN and NAS functions.
- **Shared controllers to manage SAN and NAS**:
  - Ethernet as carrier;
  - FCoE, iSCSI, NFS, CIFS protocols;
- STaaS cloud storage, combines on-premises and cloud storage.




NAS I/O
SAN I/O
NAS Control
SAN Control
CM
NAS VOL
SAN VOL


SAN
Control
LANDrv/TCP
CIFS
NFS
File System
NAS Control
CM#0
LANDrv/TCP
CIFS
NFS
File System
NAS Control
SAN
Control
CM#1
NAS VOL
(shared VOL)

Advantages:

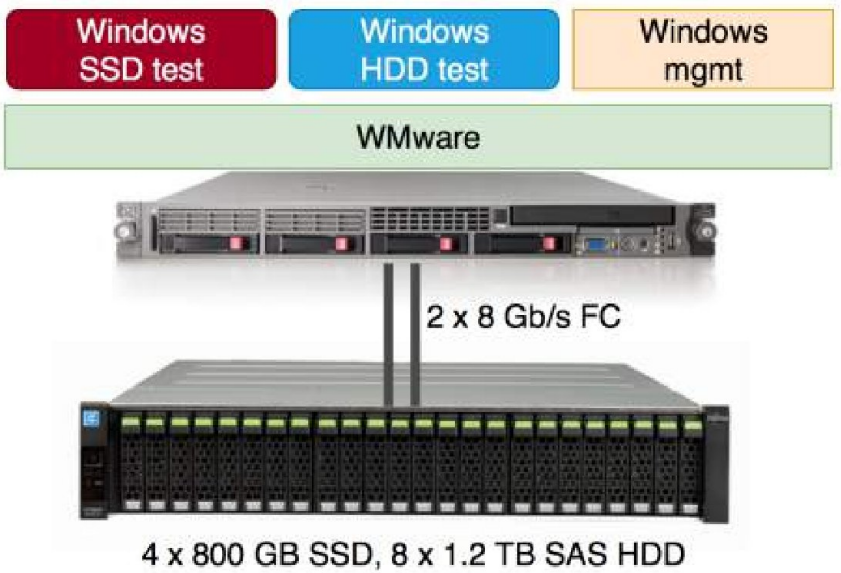
- Save space, device, management efforts, network costs;
- leverages utilization between NAS and SAN;
- **Conform to STaaS providers;**

Disadvantages:

- Performance penalties;

Case study: measuring Eternus DX200

The configuration...



The results...

Test#1:

- 2 x **SSD** in RAID1;
- 2 x **SSD** in RAID0 cache;
- 6 x **HDD** in RAID5;
- 1 x hotspare;

Test#2:

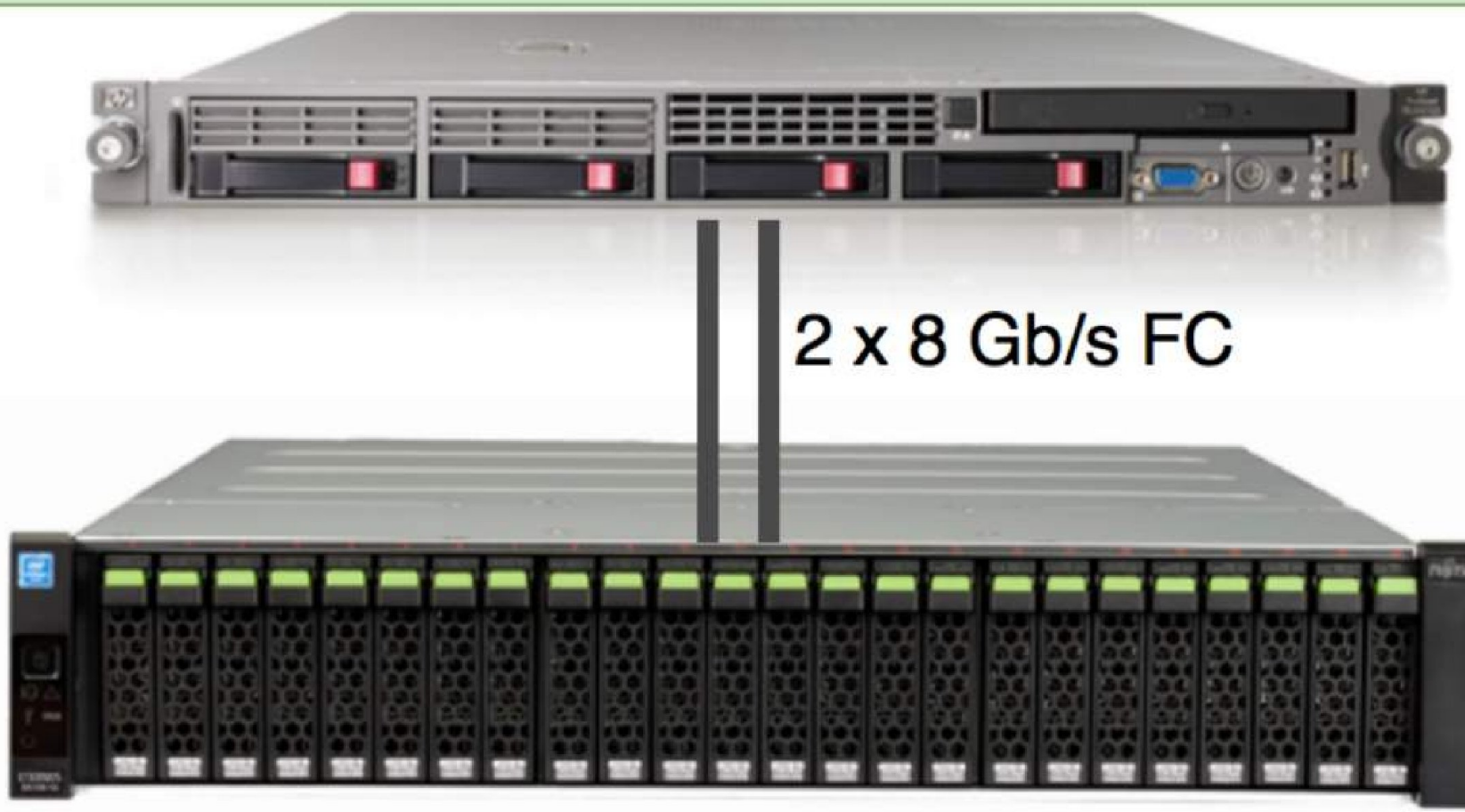
- 4 x **SSD** in RAID5;
- 8 x **HDD** in RAID6;
- no hotspare + no cache;

Windows SSD test

Windows HDD test

Windows mgmt

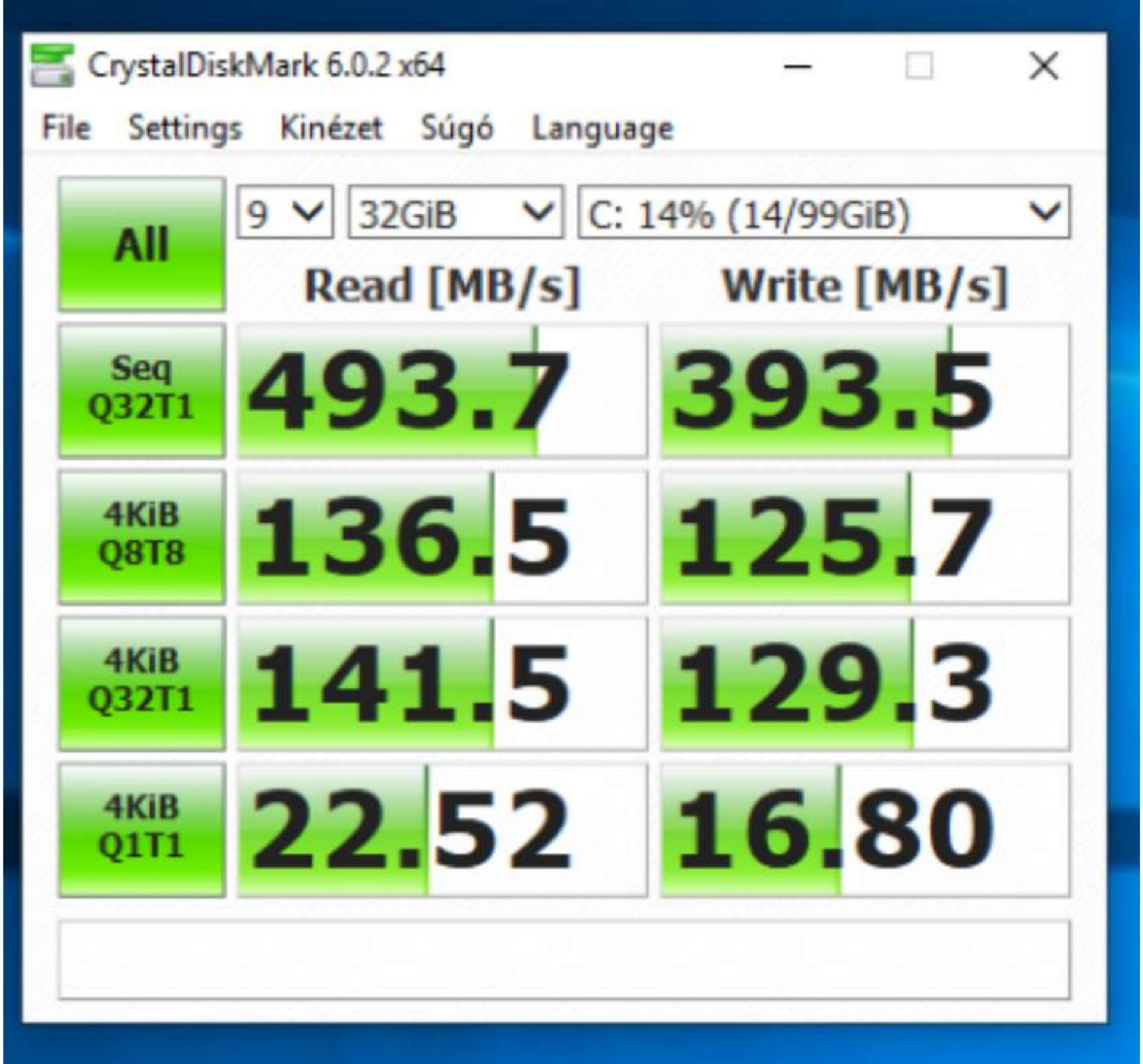
WMware

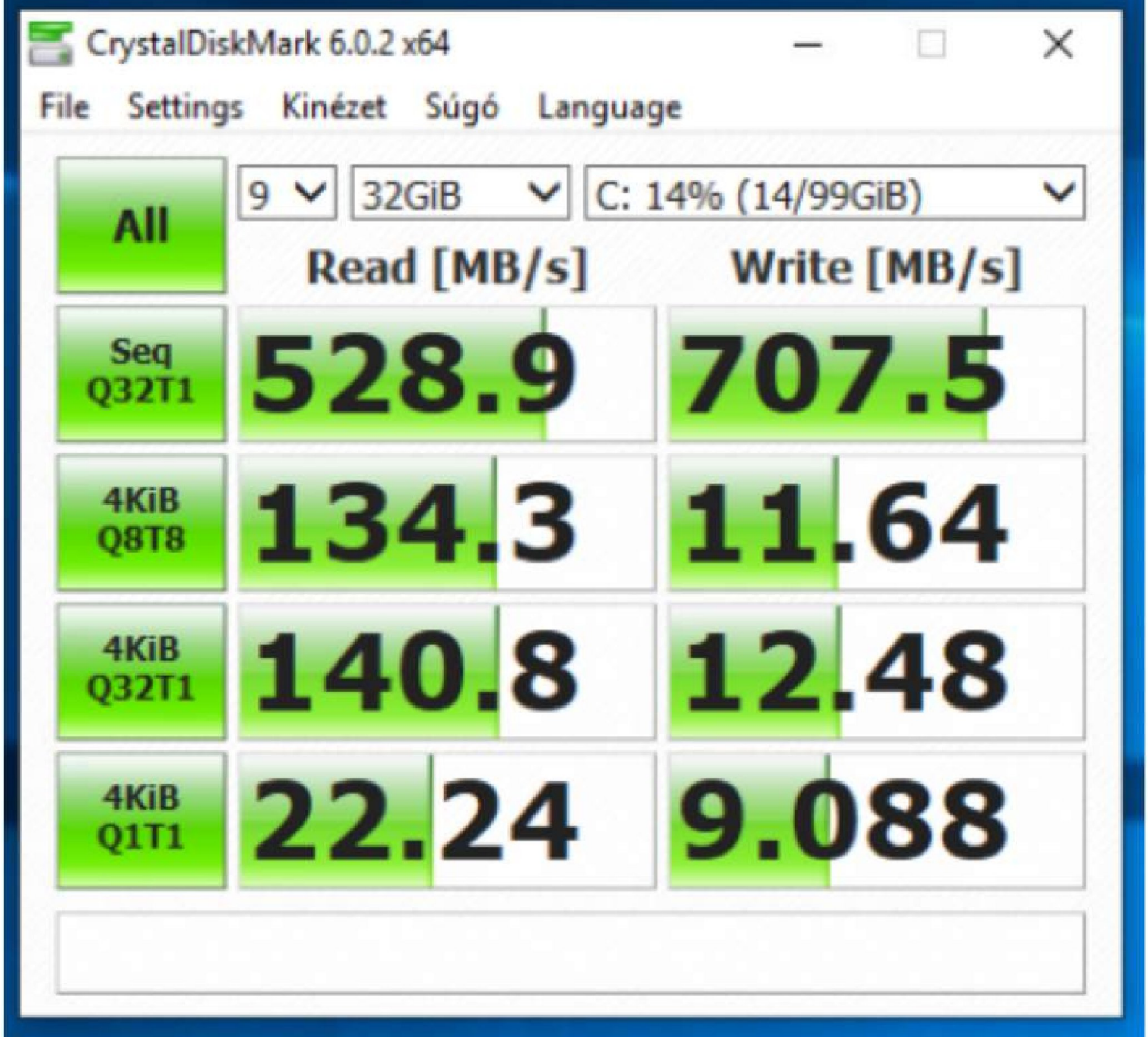


4 x 800 GB SSD, 8 x 1.2 TB SAS HDD

Test#1:

- 2 x **SSD** in RAID1;
- 2 x **SSD** in RAID0 cache;
- 6 x **HDD** in RAID5;
- 1 x hotspare;


CrystalDiskMark 6.0.2 x64
File Settings Kinézet Súgó Language
All
9
32GiB
C: 14% (14/99GiB)
Read [MB/s]
Write [MB/s]
Seq Q32T1
493.7
393.5
4KiB Q8T8
136.5
125.7
4KiB Q32T1
141.5
129.3
4KiB Q1T1
22.52
16.80


CrystalDiskMark 6.0.2 x64
File Settings Kinézet Súgó Language
All
9
32GiB
C: 14% (14/99GiB)
Read [MB/s]
Write [MB/s]
Seq Q32T1
528.9
707.5
4KiB Q8T8
134.3
11.64
4KiB Q32T1
140.8
12.48
4KiB Q1T1
22.24
9.088

Test#1:

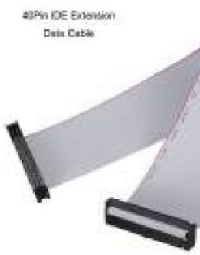
- 2 x **SSD** in RAID1;
- 2 x **SSD** in RAID0 cache;
- 6 x **HDD** in RAID5;
- 1 x hotspare;

Test#2:

- 4 x **SSD** in RAID5;
- 8 x **HDD** in RAID6;
- no hotspare + no cache;

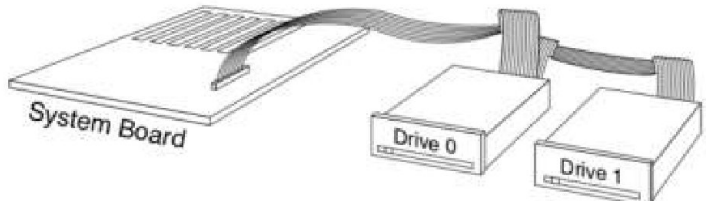
Attachment (PATA

- An old PC standard initiated in 1986;
- Made disk controllers independent from drives;
- ATA 0 [**3.3 MB/s**] - 7 [**133 MB/s**];
- **I**ntegrated **D**evice **E**lectronics, EIDE at Western Digital;
- Originally **16-bits** wide, then **22**, **28**;
- Maximum drive capacity 2^28 x 512 = 128 GBytes;
- **2 drives attached** to **parallel bus** (master, slave, cable select, OS arbitration);
- Traditional **40-pin ribbon cable + 4 for electricity**;
- Traditionally **serialized command sequence**;



PATA commands:

- 8 bit commands, from $00 to $FF;

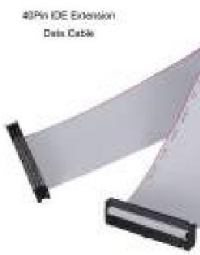


40Pin IDE Extension

Data Cable

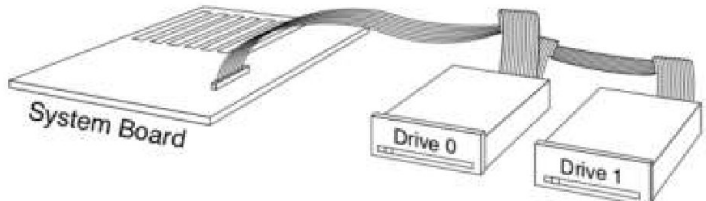
Attachment (PATA

- An old PC standard initiated in 1986;
- Made disk controllers independent from drives;
- ATA 0 [**3.3 MB/s**] - 7 [**133 MB/s**];
- **I**ntegrated **D**evice **E**lectronics, EIDE at Western Digital;
- Originally **16-bits** wide, then **22**, **28**;
- Maximum drive capacity 2^28 x 512 = 128 GBytes;
- **2 drives attached** to **parallel bus** (master, slave, cable select, OS arbitration);
- Traditional **40-pin ribbon cable + 4 for electricity**;
- Traditionally **serialized command sequence**;

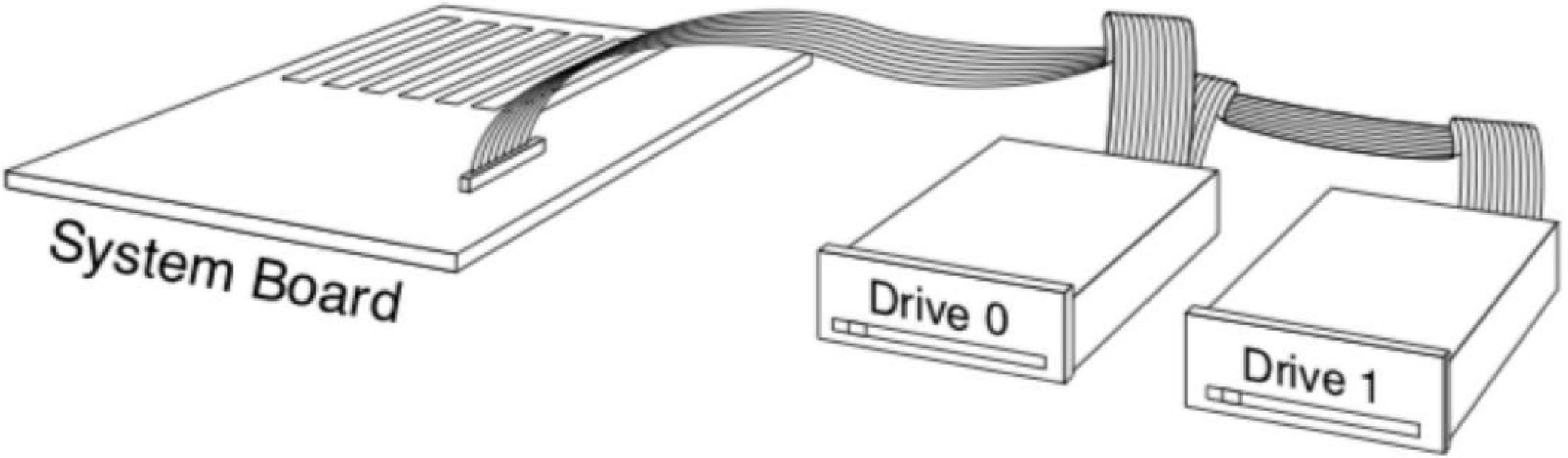


PATA commands:

- 8 bit commands, from $00 to $FF;



d sequence,

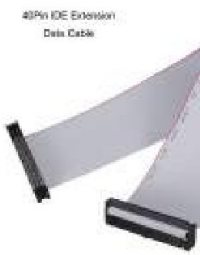


tus, Data;

ATA commands

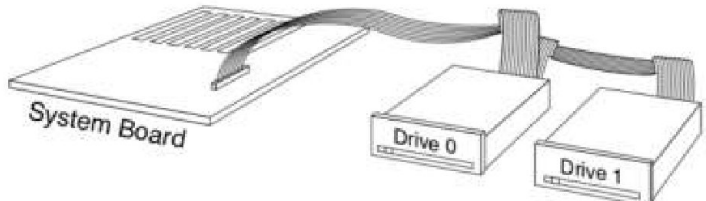
Attachment (PATA

- An old PC standard initiated in 1986;
- Made disk controllers independent from drives;
- ATA 0 [**3.3 MB/s**] - 7 [**133 MB/s**];
- **I**ntegrated **D**evice **E**lectronics, EIDE at Western Digital;
- Originally **16-bits** wide, then **22**, **28**;
- Maximum drive capacity 2^28 x 512 = 128 GBytes;
- **2 drives attached** to **parallel bus** (master, slave, cable select, OS arbitration);
- Traditional **40-pin ribbon cable + 4 for electricity**;
- Traditionally **serialized command sequence**;

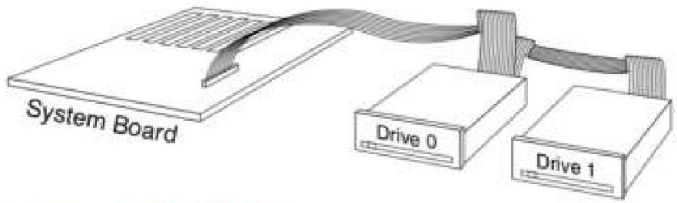


PATA commands:

- 8 bit commands, from $00 to $FF;



- Traditional **40-pin ribbon cable + 4 for electricity**;
- Traditionally **serialized command sequence**;



PATA commands:

- 8 bit commands, from $00 to $FF;
- IO registers: **Command**, **Control**, **Status**, **Data**;
- Command registers are used to send ATA commands over the parallel ports;
- Status is received in status registers;
- ATA-8 contains a much broader advanced feature set;
- **Protocols:**
  - PIO read, write;
  - DMA transfer;
  - Non-data commands.

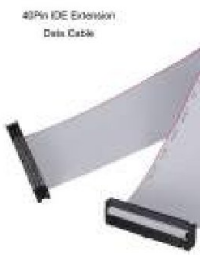
http://www.t13.org/documents/uploadeddocuments/docs2006/d1699r3f-ata8-acs.pdf
ftp://ftp.seagate.com/acrobat/reference/111-1c.pdf

General command set:

- EXECUTE DEVICE DIAGNOSTIC
- FLUSH CACHE
- IDENTIFY DEVICE
- READ DMA
- READ MULTIPLE
- **READ SECTOR(S)**
- READ VERIFY SECTOR(S)
- SET FEATURES
- SET MULTIPLE MODE
- WRITE DMA
- WRITE MULTIPLE
- **WRITE SECTOR(S)**

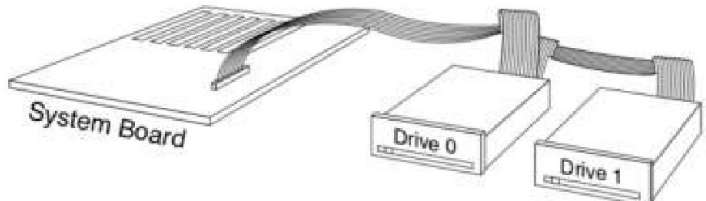
Attachment (PATA

- An old PC standard initiated in 1986;
- Made disk controllers independent from drives;
- ATA 0 [**3.3 MB/s**] - 7 [**133 MB/s**];
- **I**ntegrated **D**evice **E**lectronics, EIDE at Western Digital;
- Originally **16-bits** wide, then **22**, **28**;
- Maximum drive capacity 2^28 x 512 = 128 GBytes;
- **2 drives attached** to **parallel bus** (master, slave, cable select, OS arbitration);
- Traditional **40-pin ribbon cable + 4 for electricity**;
- Traditionally **serialized command sequence**;



PATA commands:

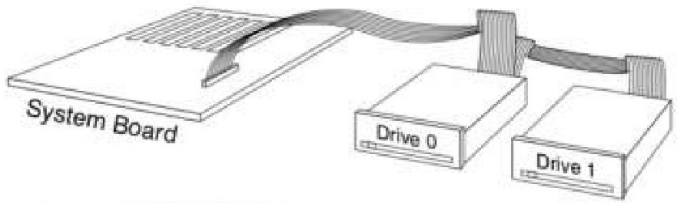
- 8 bit commands, from $00 to $FF;



Tagged Command Queuing:

- One request at a time to drives;
- As in case of SCSI, **sending multiple commands at the same time;**
- Due to the high number of interrupts this solution required lot of CPU power and was not really efficient;

- Traditional **40-pin ribbon cable + 4 for electricity**;
- Traditionally **serialized command sequence**;



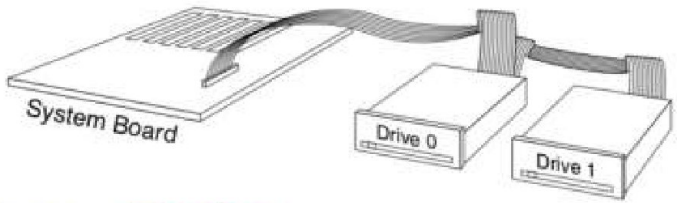
PATA commands:

- 8 bit commands, from $00 to $FF;
- IO registers: **Command**, **Control**, **Status**, **Data**;
- Command registers are used to send ATA commands over the parallel ports;
- Status is received in status registers;
- ATA-8 contains a much broader advanced feature set;
- **Protocols:**
  - PIO read, write;
  - DMA transfer;
  - Non-data commands.

http://www.t13.org/documents/uploadeddocuments/docs2006/d1699r3f-ata8-acs.pdf
ftp://ftp.seagate.com/acrobat/reference/111-1c.pdf

Command Block registers							
1	0	0	0	0	Data register		1F0
1	0	0	0	1	Error register	Feature register	1F1
1	0	0	1	0	Sector Count register		1F2
1	0	0	1	1	Sector Number register		1F3
1	0	1	0	0	Cylinder Low register		1F4
1	0	1	0	1	Cylinder High register		1F5
1	0	1	1	0	Drive/Head register		1F6
1	0	1	1	1	Status register	Command register	1F7

- Traditional **40-pin ribbon cable + 4 for electricity**;
- Traditionally **serialized command sequence**;



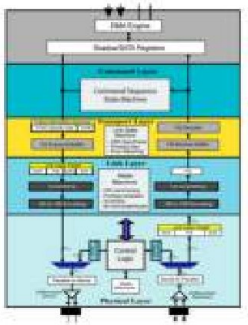
PATA commands:

- 8 bit commands, from $00 to $FF;
- IO registers: **Command**, **Control**, **Status**, **Data**;
- Command registers are used to send ATA commands over the parallel ports;
- Status is received in status registers;
- ATA-8 contains a much broader advanced feature set;
- **Protocols:**
  - PIO read, write;
  - DMA transfer;
  - Non-data commands.

http://www.t13.org/documents/uploadeddocuments/docs2006/d1699r3f-ata8-acs.pdf
ftp://ftp.seagate.com/acrobat/reference/111-1c.pdf

Serial ATA

- It is **serialized version** of Parallel ATA standard;
- Serialization helps reducing cabling structure and increasing clock speed -> increased transfer speeds: SATA1 [**150 MB/s**], SATA2 [**300 MB/s**], SATA3 [**600 MB/s**], SATA Express, 3.2 [**2 GB/s**];
- 100% software compatible with PATA standard, enhanced command set;
- Dedicated **p2p crossbar** connection among the host bus and drives, eliminating the bus (like data network);
- Frame based protocol;
- Hot-plug, hot-swap;
- Layered communication structure;



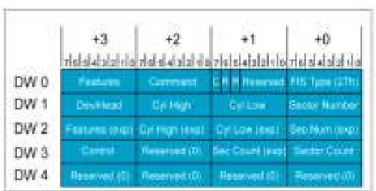
Link layer:
- 8b/10b encoding;
- converts data streams into frames;
- CRC;
- flow control;

Transport layer:
- managing FIS;

SATA Port Multipliers:
- Up to 15 drives;
- Cost efficient connection of drives;
- Share the HBA bandwidth;

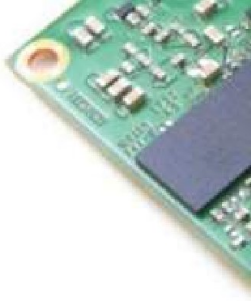
Benefits:
- lower pin count;
- larger performance, full duplex;
- simple configuration;
- lower voltage;

Frame Information Structures:
- Generated in the transport layer;
- Data movement setup;
- Read and write data;
- Used to control IO;
- Carries **contents of ATA command registers**;



Native Command Queuing:
- **Multiple IO commands** in parallel (like SCSI);
- Due to mechanical constraints it is not efficient to serve IO requests in the order of their issue = **elevator scheduling problem**;
- Eliminate unnecessary travels = significantly improve performance;
- MCQ allows **out of order** IO command completion;
- Reordering of 32 IO commands;
- Command order **received** vs. command order **optimized for shortest completion time**;
- minimize **rotation** and **head movement**;

https://www.mindshare.com/files/ebooks/SATA%20Storage%20Technology.pdf



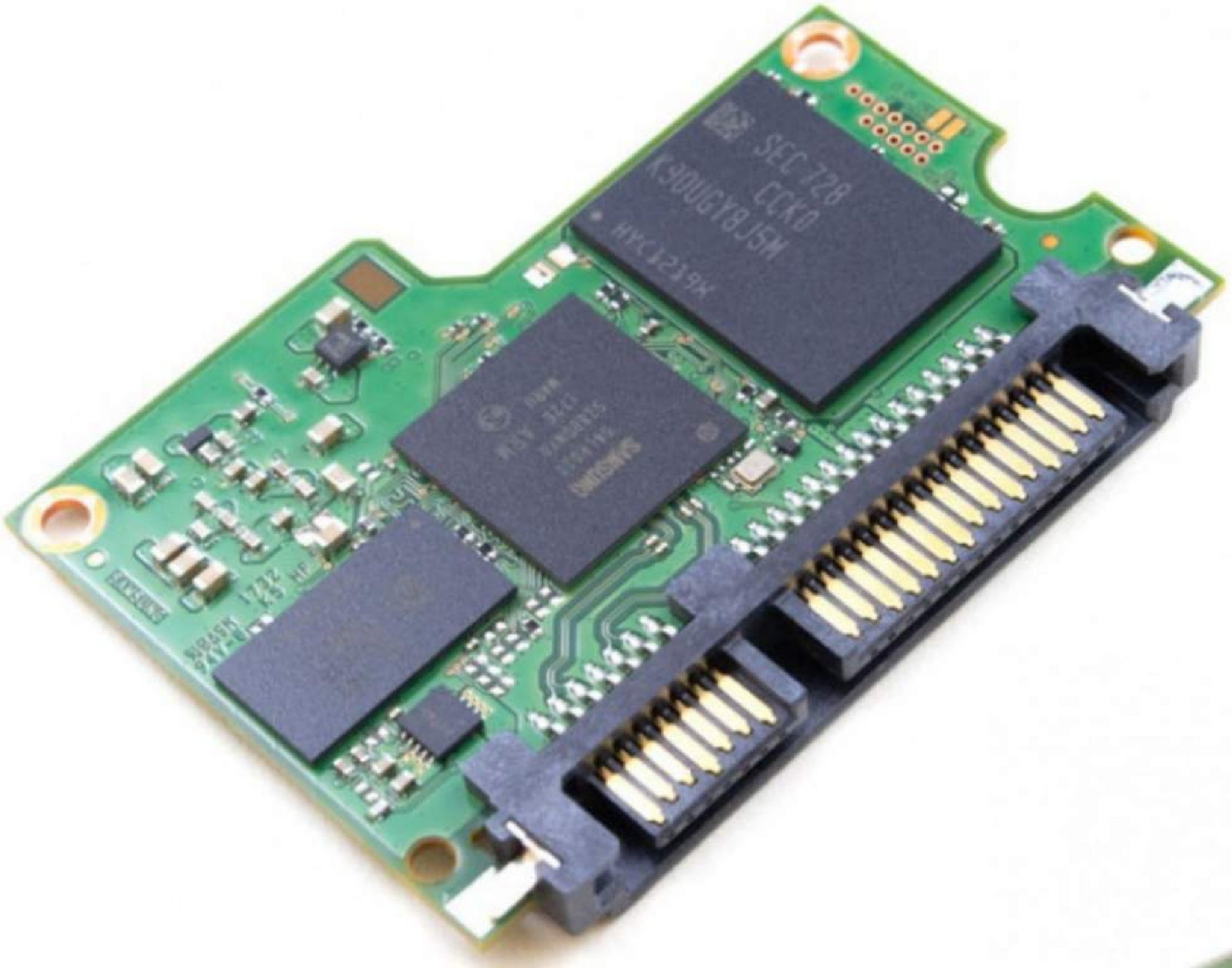
- It is **serialized version** of Parallel ATA standard;
- Serialization helps reducing cabling structure and increasing clock speed -> increased transfer speeds: SATA1 [**150 MB/s**], SATA2 [**300 MB/s**], SATA3 [**600 MB/s**], SATA Express, 3.2 [**2 GB/s**];
- 100% software compatible with PATA standard, enhanced command set;
- Dedicated **p2p crossbar** connection among the host bus and drives, eliminating the bus (like data network);
- Frame based protocol;
- Hot-plug, hot-swap;
- Layered communication structure;

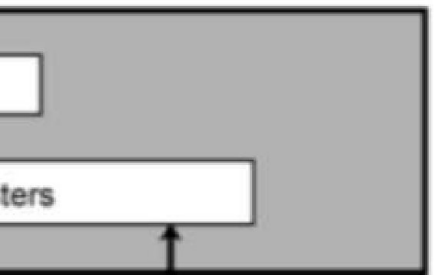
SATA Port Multipliers:

- Up to 15 drives;
- Cost efficient connection of drives;
- Share the HBA bandwidth;

SATA PM connectivity: one port, one cable, four drives, 4 ports = 16 drives

Benefits:

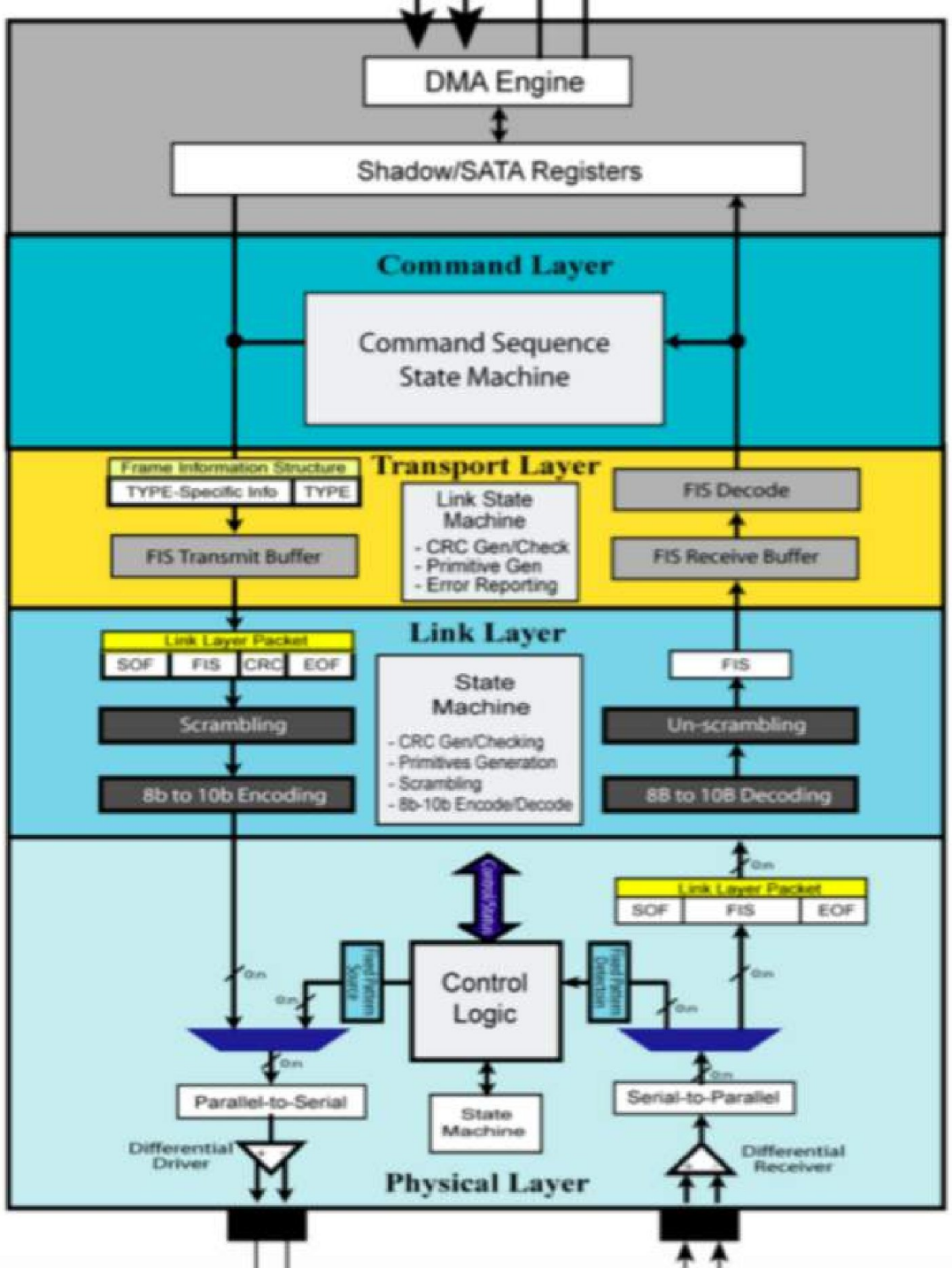

SEC 728
CCK0
K3U6Y8J5M


ters

Benefits:

- lower pin count;
- larger performance, full duplex;
- simple configuration;
- lower voltage;

ost
ork);



Link la

- 8b/
- con
  fran
- CR
- flow

Transp

Link layer:

- 8b/10b encoding;
- converts data streams into frames;
- CRC;
- flow control;

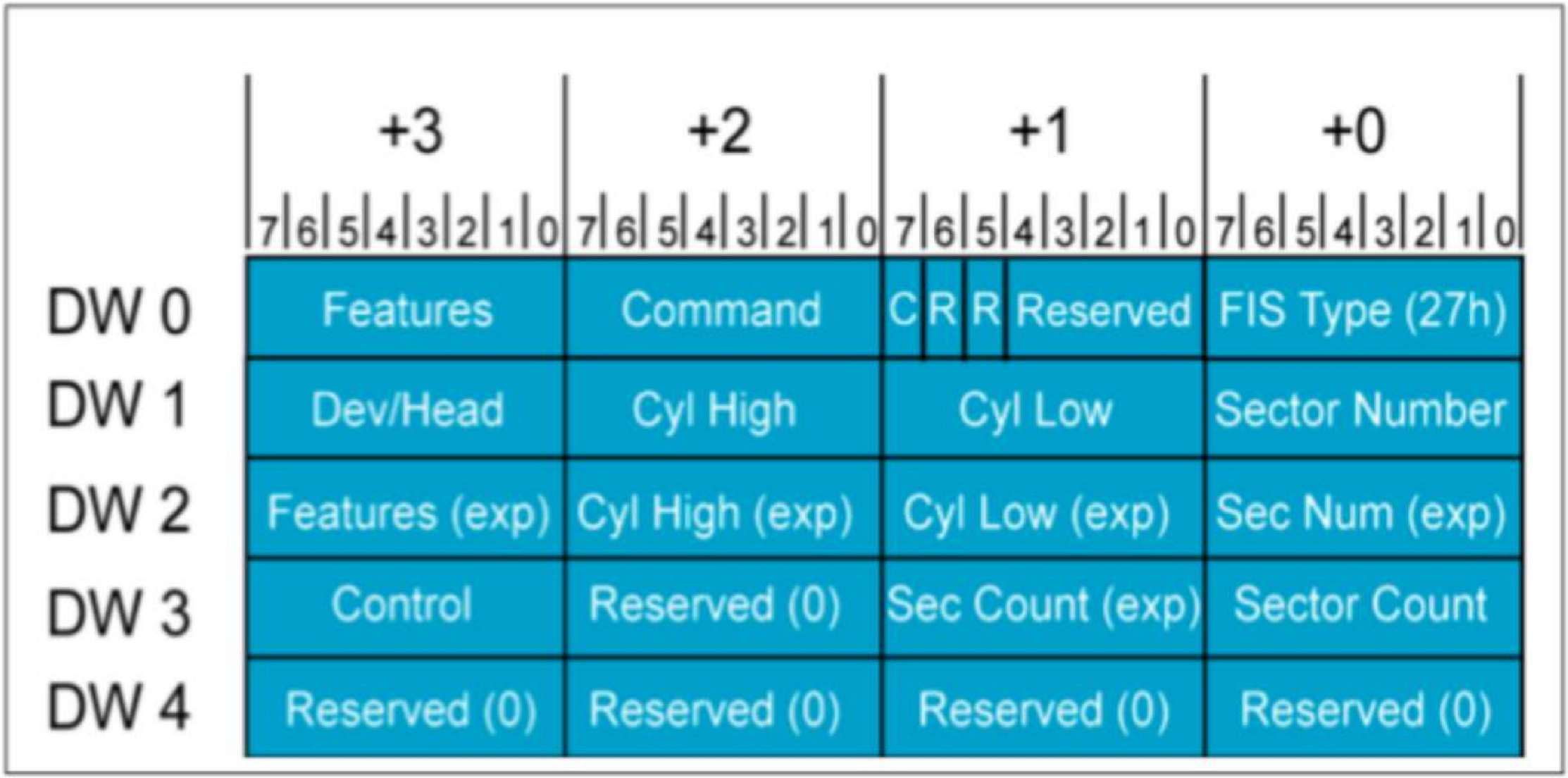
Transport layer:

Transport layer:

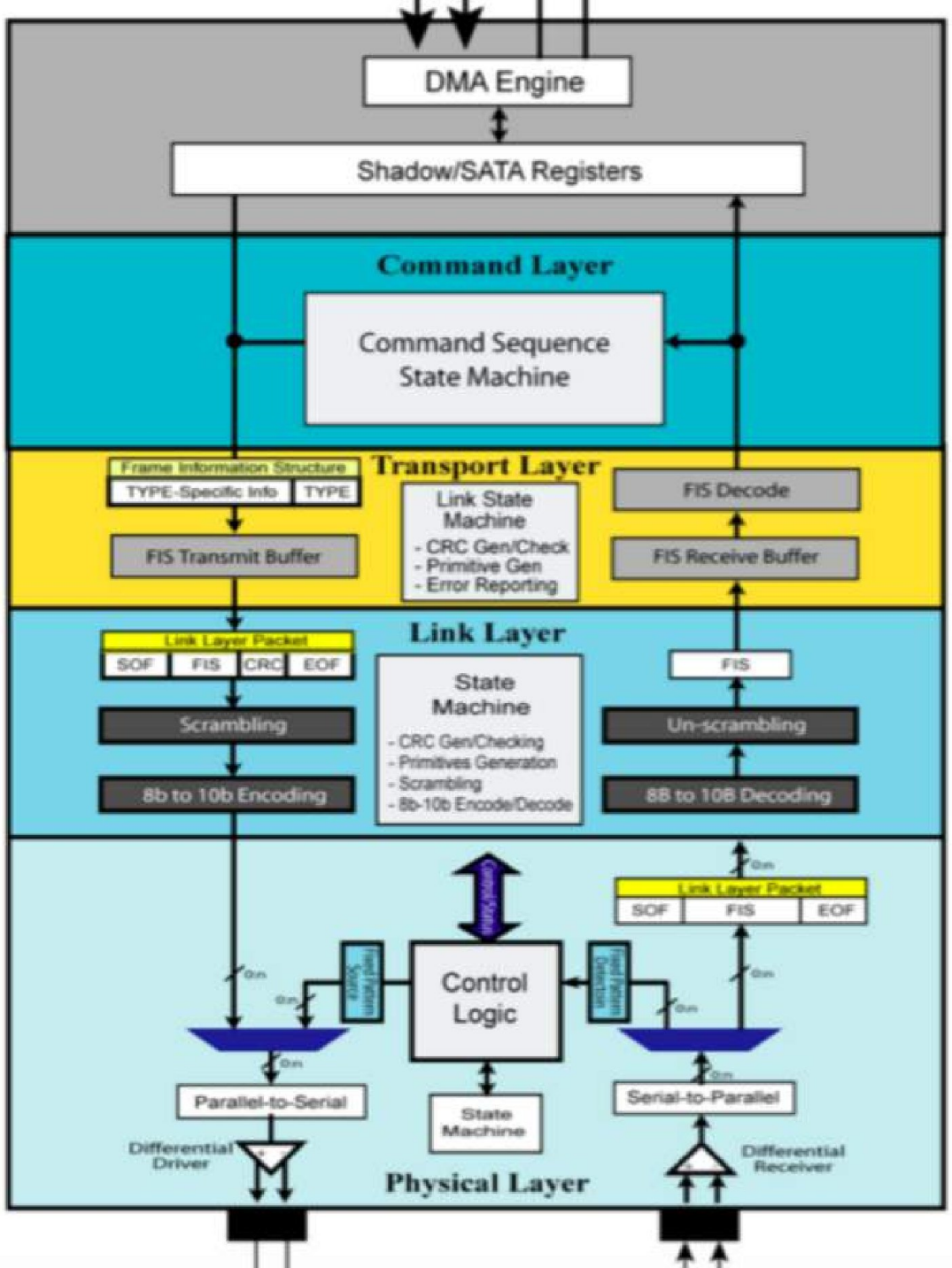
- managing FIS;

Frame Information Structures:

- Generated in the transport layer;
- Data movement setup;
- Read and write data;
- Used to control IO;
- Carries **contents of ATA command registers**;


+3
+2
+1
+0
7 6 5 4 3 2 1 0
DW 0
Features
Command
C R R Reserved
FIS Type (27h)
DW 1
Dev/Head
Cyl High
Cyl Low
Sector Number
DW 2
Features (exp)
Cyl High (exp)
Cyl Low (exp)
Sec Num (exp)
DW 3
Control
Reserved (0)
Sec Count (exp)
Sector Count
DW 4
Reserved (0)
Reserved (0)
Reserved (0)
Reserved (0)

ost
ork);



Link la

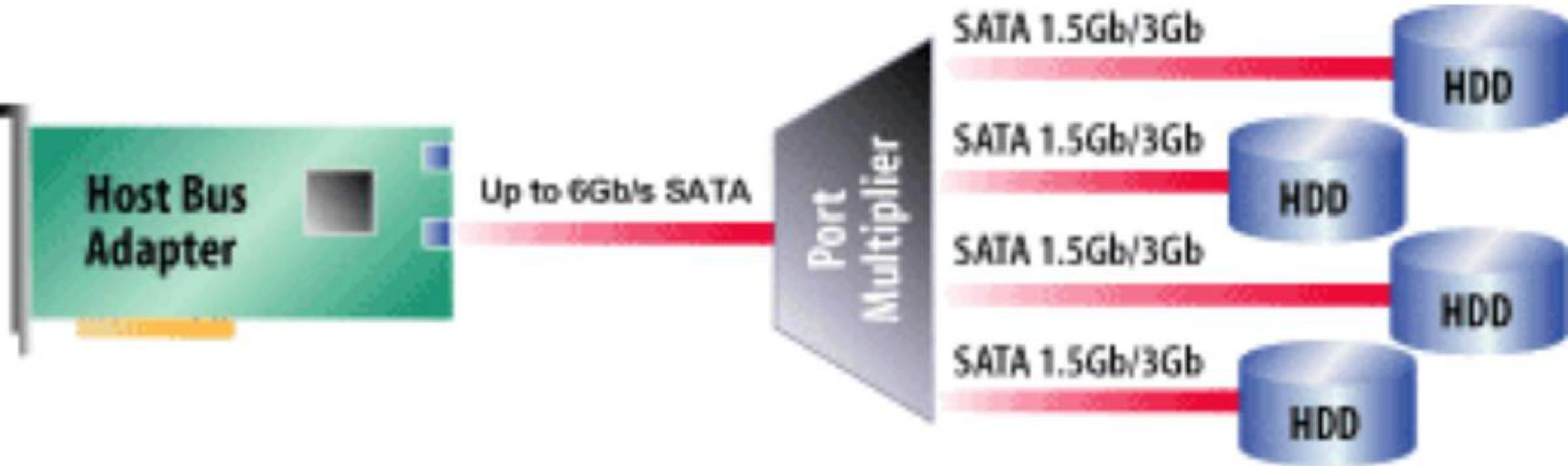
- 8b/
- con
  fran
- CR
- flow

Transp

ayered communication stru

SATA Port Multipliers:

- Up to 15 drives;
- Cost efficient connection of drives;
- Share the HBA bandwidth;



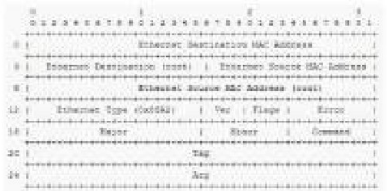
SATA PM connectivity: one port, one cable, four drives; 4-ports = 16 drives

ull

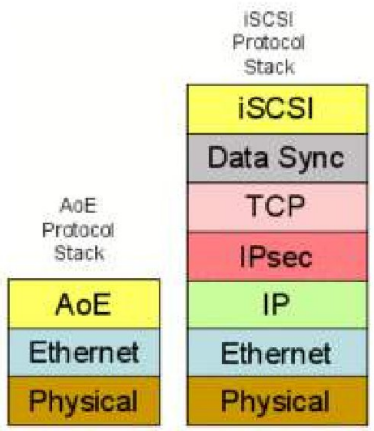
Native Command Queuing:

- **Multiple IO commands** in parallel (like SCSI);
- Due to mechanical constraints it is not efficient to serve IO requests in the order of their issue = **elevator scheduling problem**;
- Eliminate unnecessary travels = significantly improve performance;
- MCQ allows **out of order** IO command completion;
- Reordering of 32 IO commands;
- Command order **received** vs. command order **optimized for shortest completion time**;
- minimize **rotation** and **head movement**;

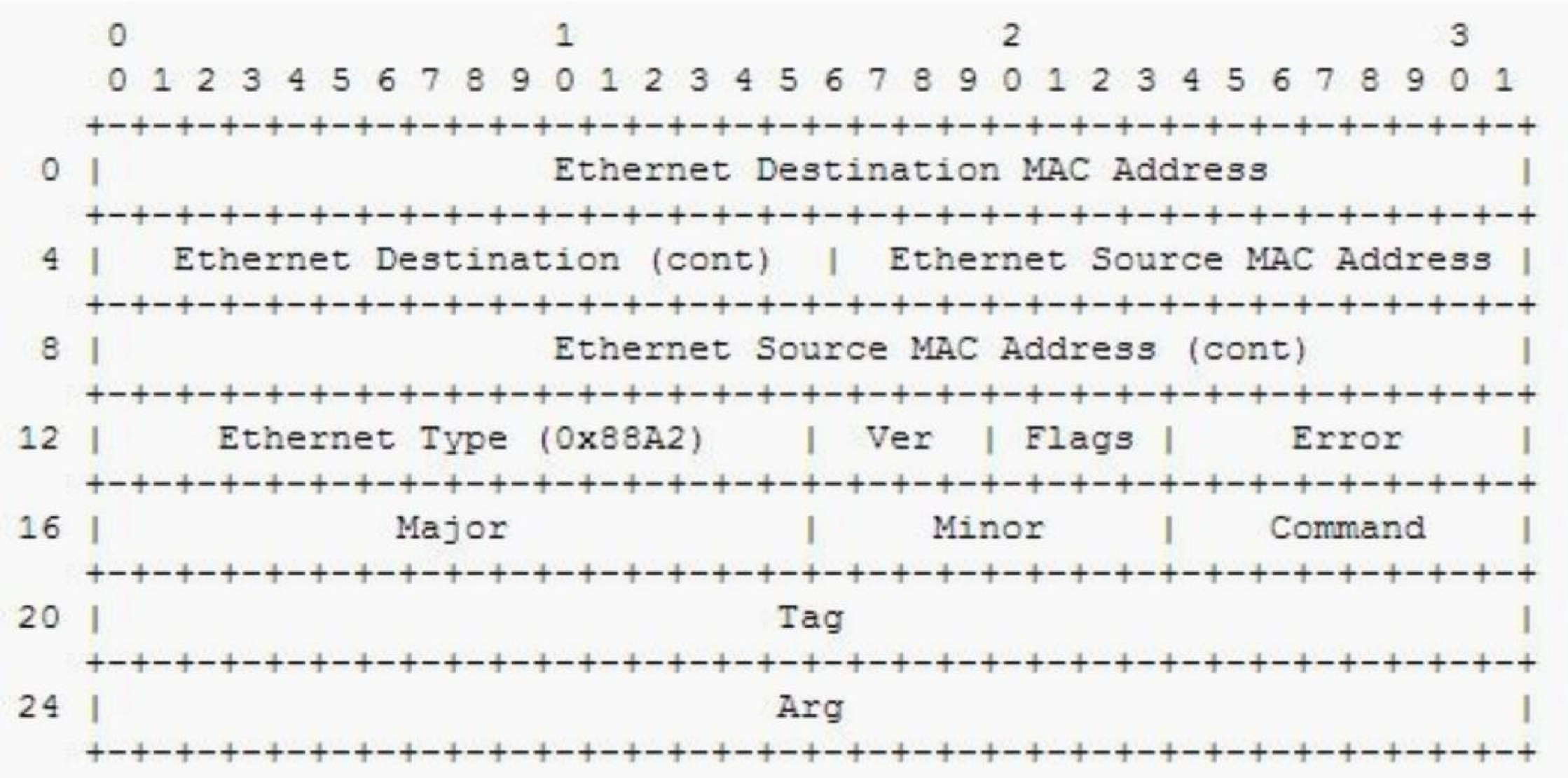
ATA over Ethernet



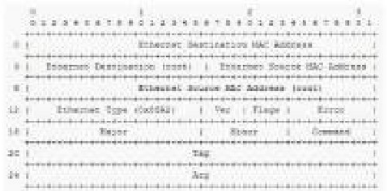
- Simplicity attracts! :)
- Encapsulates **ATA commands into Ethernet carrier**;
- Carries ATA commands information to be put into registers;
- Data is sent in 8 KBytes = 16 blocks;
- Replaces traditional ATA transport layer (bus and cabling) to Ethernet;
- No need for multipathing;
- Lightweight protocol stack;
- Non-routing protocol;
- **AoE initiator** - **AoE target**;



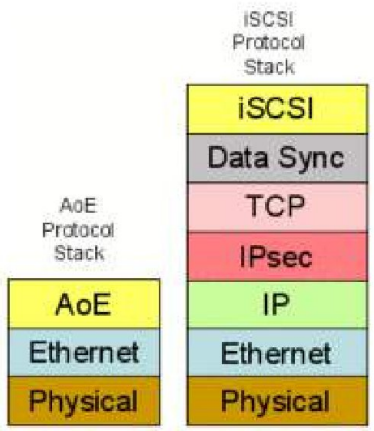
https://www.linuxjournal.com/content/mastering-ata-over-ethernet
http://storagegaga.com/aoe-all-about-ethernet/
http://download.alyseo.com:81/pub/partners/Coraid/Docs/AoE/AoEDescription.pdf


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
0
Ethernet Destination MAC Address
4
Ethernet Destination (cont)
Ethernet Source MAC Address
8
Ethernet Source MAC Address (cont)
12
Ethernet Type (0x88A2)
Ver
Flags
Error
16
Major
Minor
Command
20
Tag
24
Arg

ATA over Ethernet



- Simplicity attracts! :)
- Encapsulates **ATA commands into Ethernet carrier**;
- Carries ATA commands information to be put into registers;
- Data is sent in 8 KBytes = 16 blocks;
- Replaces traditional ATA transport layer (bus and cabling) to Ethernet;
- No need for multipathing;
- Lightweight protocol stack;
- Non-routing protocol;
- **AoE initiator** - **AoE target**;

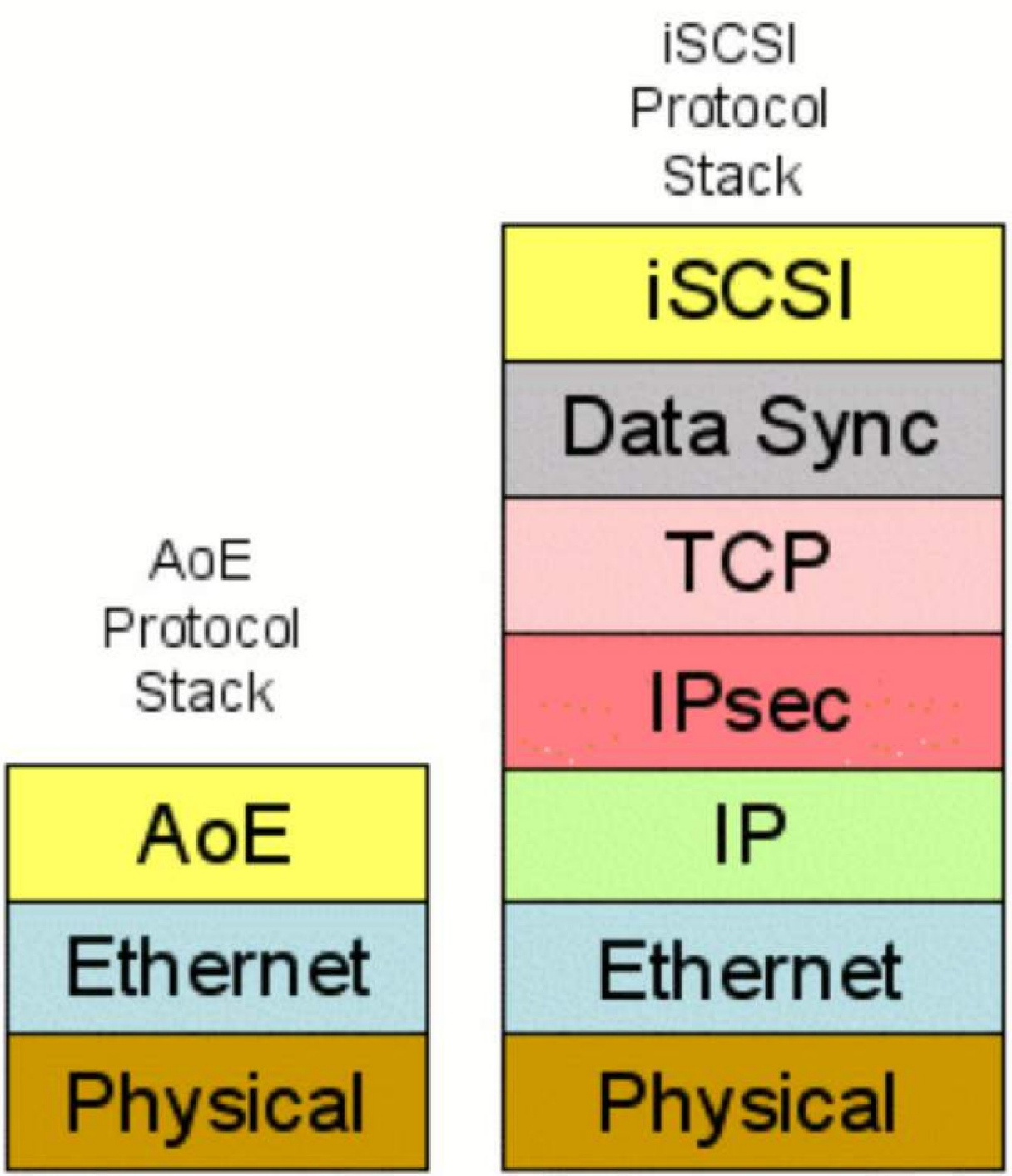


https://www.linuxjournal.com/content/mastering-ata-over-ethernet
http://storagegaga.com/aoe-all-about-ethernet/
http://download.alyseo.com:81/pub/partners/Coraid/Docs/AoE/AoEDescription.pdf

oE devices

=10

vel=1 --raid-
v/etherd/e0.1



SCSI Protocol

- History starts at 1978 (Alan Shugart, SASI), continues at 1986 (Small Computer System Interface, SCSI-1, first standardization);
- **Parallel bus standard** to ease the communication between the host and the low level disk drives;
- Uses 8/16 data + parity + 9 control signals;
- Originally:
  - 8 bit + parity wide;
  - 8 or 16 drives could be attached;
  - 10 MHz.
- **Interface names:** SCSI-1 [5 MB/s], Fast SCSI [10 MB/s], Fast Wide SCSI [20 MB/s], Ultra Wide SCSI [40 MB/s], Ultra-320 SCSI [320 MB/s], and Ultra-640 SCSI [640 MB/s];
- **SCSI initiator** - **SCSI target(s)**;
- Bus used as **state machine**: bus-free, arbitration, device selection, command, reselection (target disconnect in long operations), data, message, status;

Connectors:

- SCSI Parallel Interface;
- Fibre Channel;
- Serial Attached SCSI;
- HD Mini SAS;

SCSI command families:

- **non-data** commands;
- **reading** data from target;
- **writing** data from the initiator to target;
- **bidirectional**.

Device types:

- 5-bit field sent in **Inquiry**;
- broad range: scanners, printers, block disks, CD-ROMS, floppies... $00-$1F

Device identification:

- manually set: 0-7, 0 - bootable disk, 7 - initiator;
- BIOS set;
- slot-set in storage shelves;
- ID discovery for SAS.

Command Descriptor Block

READ command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation Code (08h)							
1	Logical Unit Number			Logical Block Address (MSB)				
2	Logical Block Address							
3	Logical Block Address (LSB)							
4	Transfer Length							
5	Vendor Unique		Reserved				Flag	Link

Difference between **SCSI** and **ATA**:

- On non-protocol level: **Enterprise Storage** vs **Personal Storage**;
- **Internal CPU driven IO** vs **Programmed IO** (lack of DMA);
- **Variable block size** vs **Fixed block size**;
- **Parallel queuing** vs **Serial command set**;
- **15** drives vs **2 x 2** drives;
- **larger buffers/speed** vs **smaller buffers/speed**;

http://pages.cs.wisc.edu/~remzi/Classes/838/Fall2001/Papers/scsi-ata.pdf

Test unit ready
Inquiry
Request sense
Send/Receive diagnostic
Start/Stop unit
Read capacity
Format unit
Read (four variants)
Write (four variants)
Log sense
Mode sense
Mode select

https://en.wikipedia.org/wiki/SCSI
http://fundasbykrishna.blogspot.com/2013/04/scsi-fundamentals-2.html

- History starts at 1978 (Alan Shugart, SASI), continues at 1986 (Small Computer System Interface, SCSI-1, first standardization);
- **Parallel bus standard** to ease the communication between the host and the low level disk drives;
- Uses 8/16 data + parity + 9 control signals;
- Originally:
  - 8 bit + parity wide;
  - 8 or 16 drives could be attached;
  - 10 MHz.
- **Interface names:** SCSI-1 **[5 MB/s]**, Fast SCSI **[10 MB/s]**, Fast Wide SCSI **[20 MB/s]**, Ultra Wide SCSI **[40 MB/s]**, Ultra-320 SCSI **[320 MB/s]**, and Ultra-640 SCSI **[640 MB/s]**;
- **SCSI initiator** - **SCSI target(s)**;
- Bus used as **state machine**: bus-free, arbitration, device selection, command, reselection (target disconnect in long operations), data, message, status;

Command Desc

READ command

Device types:


adaptec

Device types:

- 5-bit field sent in **Inquiry**;
- broad range: scanners, printers, block disks, CD-ROMS, floppies... $00-$1F

Device identification:

- manually set: 0-7, 0 - bootable disk, 7 - initiator;
- BIOS set;
- slot-set in storage shelves;
- ID discovery for SAS.

sg_raw -r 512 /dev/sdb 28 00

http://sg.danny.cz/s

Connectors:

- SCSI Parallel Interface;
- Fibre Channel;
- Serial Attached SCSI;
- HD Mini SAS;

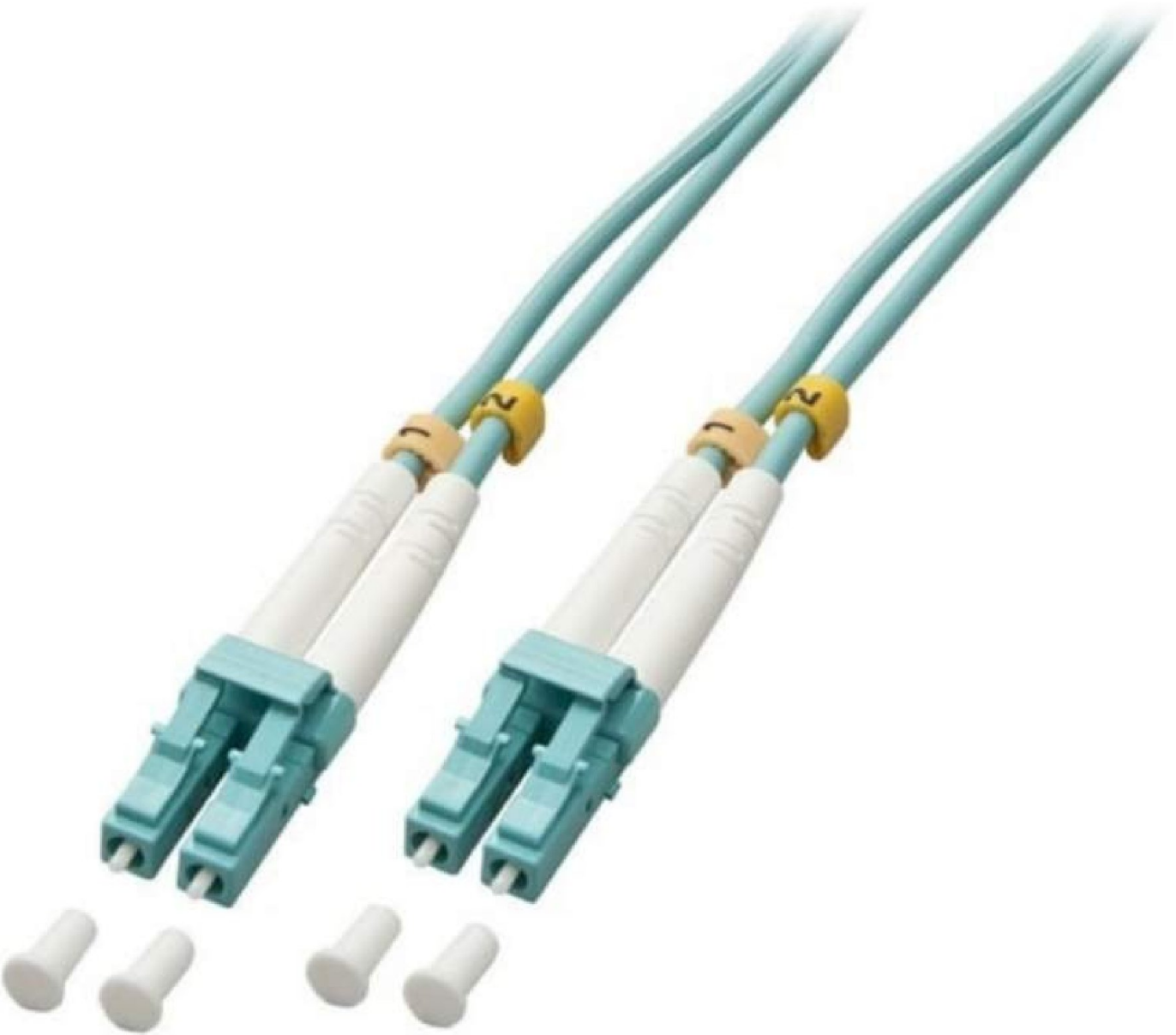
sg_raw -r 512 /dev/sdb 28 00

http://sg.danny.cz/s

Connectors:

- SCSI Parallel Interface;
- Fibre Channel;
- Serial Attached SCSI;
- HD Mini SAS;

ce;



sg_raw -r 512 /dev/sdb 28 00

http://sg.danny.cz/s

Connectors:

- SCSI Parallel Interface;
- Fibre Channel;
- Serial Attached SCSI;
- HD Mini SAS;

- HD Mini SAS,

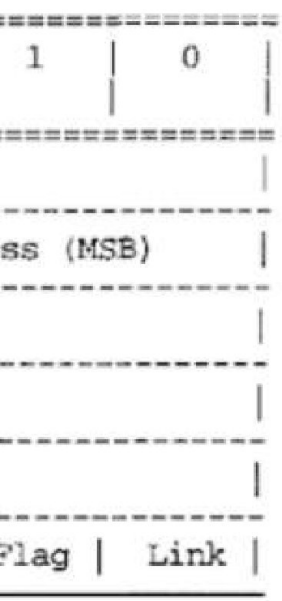
DSISEDIMS

SCSI command families:

- **non-data** commands;
- **reading** data from target;
- **writing** data from the initiator to target;
- **bidirectional**.

scriptor Block Test unit ready

lock



e vs **Personal**

IO (lack of DMA);

;

- Test unit ready
- Inquiry
- Request sense
- Send/Receive diagnostic
- Start/Stop unit
- Read capacity
- Format unit
- Read (four variants)
- Write (four variants)
- Log sense
- Mode sense
- Mode select

Command Descriptor Block

READ command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation Code (08h)							
1	Logical Unit Number			Logical Block Address (MSB)				
2	Logical Block Address							
3	Logical Block Address (LSB)							
4	Transfer Length							
5	Vendor Unique		Reserved				Flag	Link

Difference between **SCSI** and **ATA**:

3	Logical Block Address (LSB)				
4	Transfer Length				
5	Vendor Unique	Reserved		Flag	Link

Difference between **SCSI** and **ATA**:

- On non-protocol level: **Enterprise Storage** vs **Personal Storage**;
- **Internal CPU driven IO** vs **Programmed IO** (lack of DMA);
- **Variable block size** vs **Fixed block size**;
- **Parallel queuing** vs **Serial command set**;
- **15** drives vs **2 x 2** drives;
- **larger buffers/speed** vs **smaller buffers/speed**;

http://pages.cs.wisc.edu/~remzi/Classes/838/Fall2001/Papers/scsi-ata.pdf

SCSI

pot.com/2013/04/scsi-fundamentals-2

Serial Attached SCSI

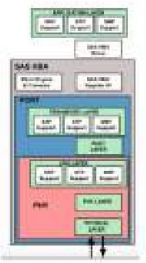
- **P2P crossbar** attached version of SCSI;
- **Serializes transport** between host adapter **expanders**, and drives;
- Certain compatibility with SATA;
- SAS1 [**300 MB/s**], SAS2 [**600 MB/s**], SAS3 [**1.5 GB/s**], SAS4 [**3 GB/s**];
- Identifies ports, HAs, devices with unique WWN;
- **narrow** and **wide** ports;

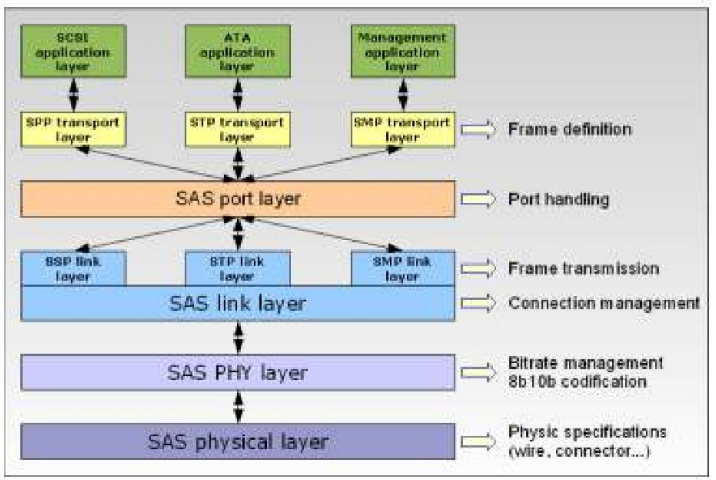
Serial SCSI Protocol:
- Wraps SCSI commands into serialized frames;
- Read Command, Data from target, Response;
- Write Command, XFER_RDY, Data to Target, Response;
- Nondata, Command, Response.

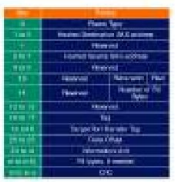
SCSI Management Protocol:
- management frames to set up and inquire about SAS topology;
- set up and manage SAS routing;
- like RIP, BGP, or OSPF in case of Ethernet;

SATA Transport Protocol:
- Wraps SATA commands;
- Tunneling;
- Uses underlying SAS layers for transport;
- FIS.

SAS and **SCSI**:
- **P2P** vs **multidrop** bus;
- **no termination** vs **terminated bus**;
- **65535** devices vs **8/16** devices;
- dedicated **full bandwidth** vs. **shared bus** bandwidth;







SAS and **SATA**:
- **multiple initiators** vs **single initiator**;
- **TCQ** vs **NCQ**;
- **SCSI command set** vs **ATA command set**;
- **multipath** IO vs **single path** IO;
- **1.6V** signaling vs **0.6V** signaling;
- up to **10 meters** vs **1 meter** cabling;

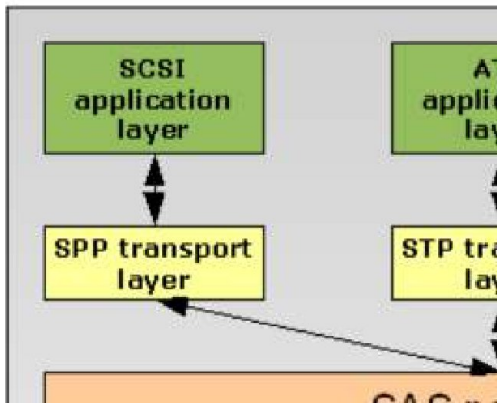
https://www.also.com/pub/pdf/hp_serial_attached_scsi.pdf
https://www.mindshare.com/files/ebooks/SAS%20Storage%20Architecture.pdf

Serial Attached

- **P2P crossbar** attached version of SCSI;
- **Serializes transport** between host adapter **expanders**, and drives;
- Certain compatibility with SATA;
- SAS1 [**300 MB/s**], SAS2 [**600 MB/s**], SAS3 [**1.5 GB/s**], SAS4 [**3 GB/s**];
- Identifies ports, HAs, devices with unique WWN;
- **narrow** and **wide** ports;

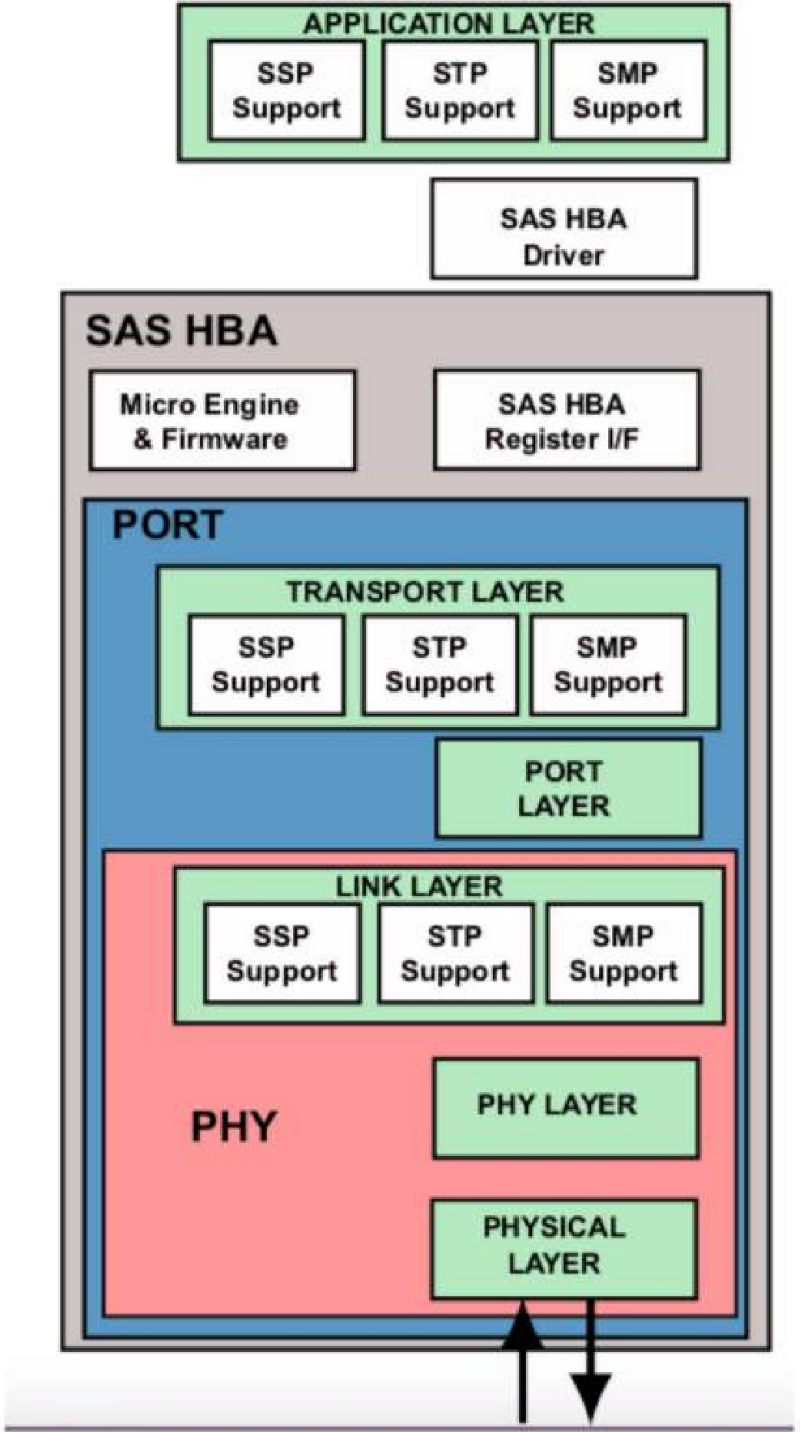
Serial SCSI
- Wraps S serialized
- Read Co Respons
- Write Co to Target
- Nondata

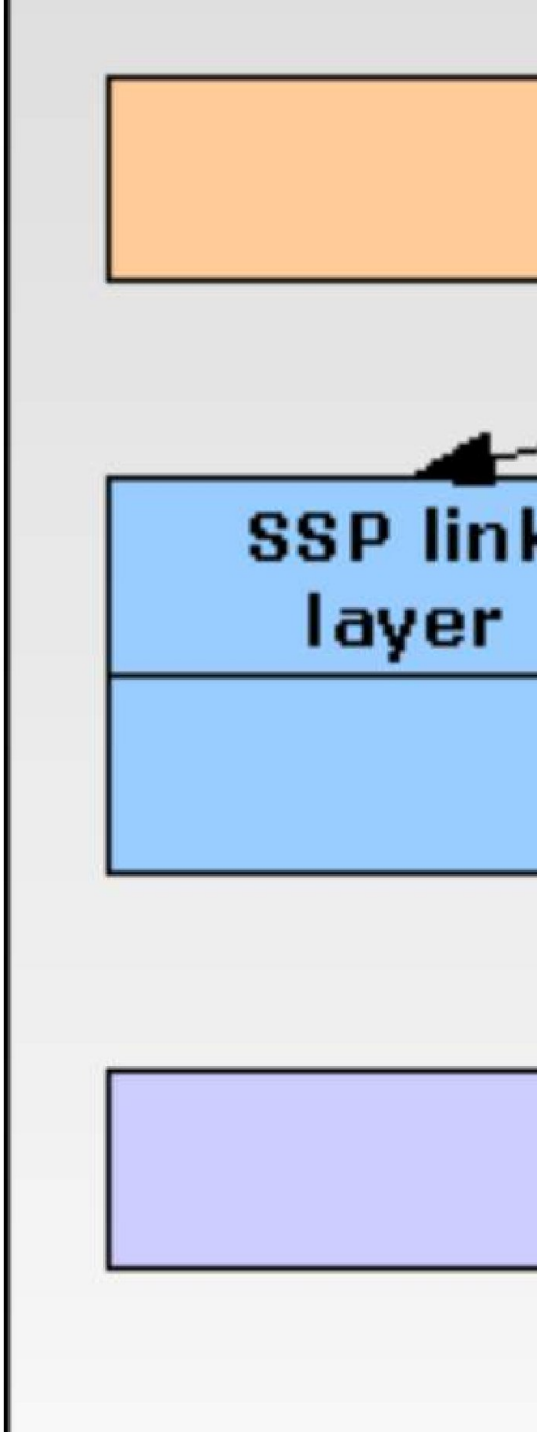
SAS and **SCSI**:
- **P2P** vs **multidrop** bus;

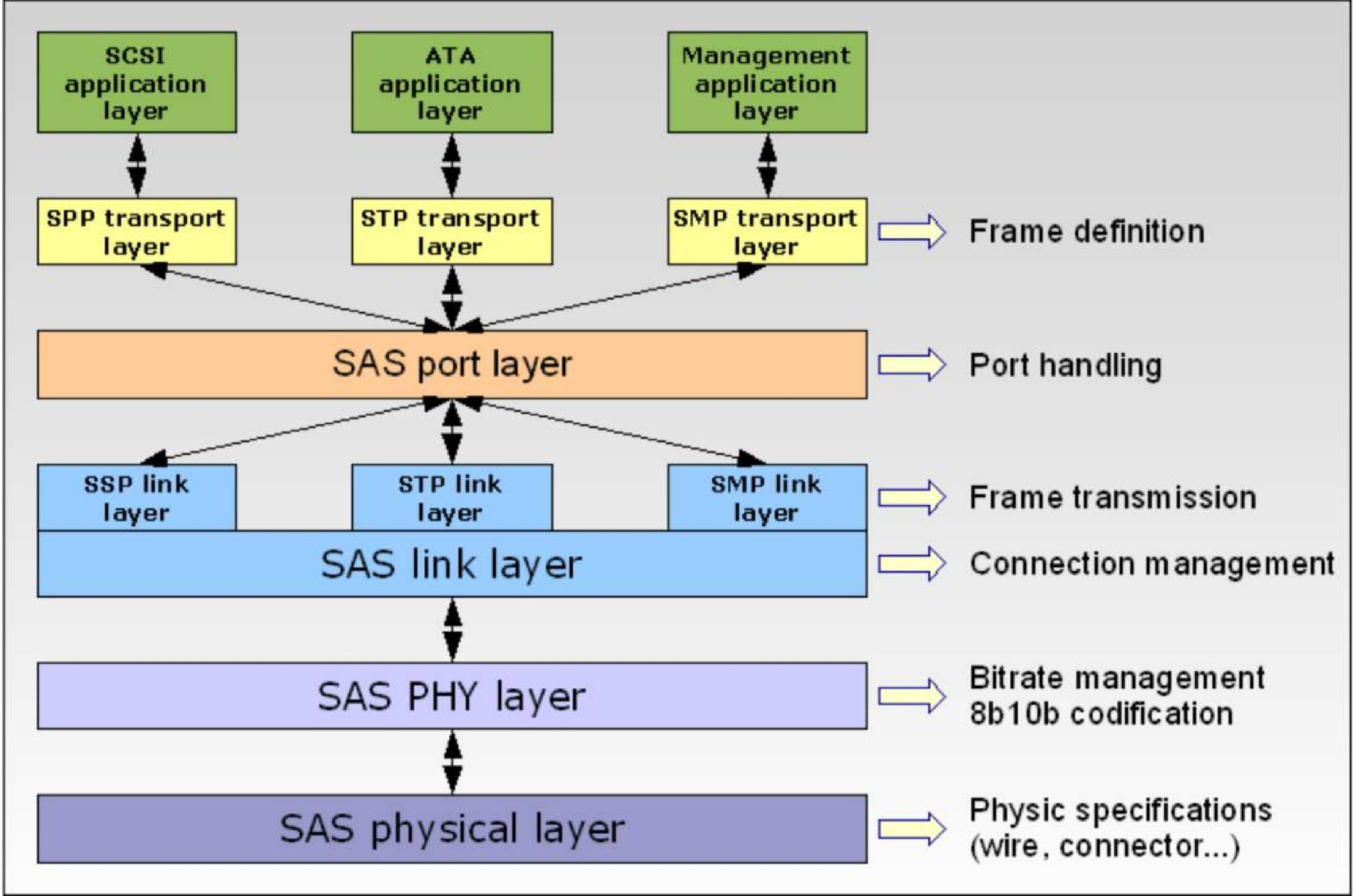


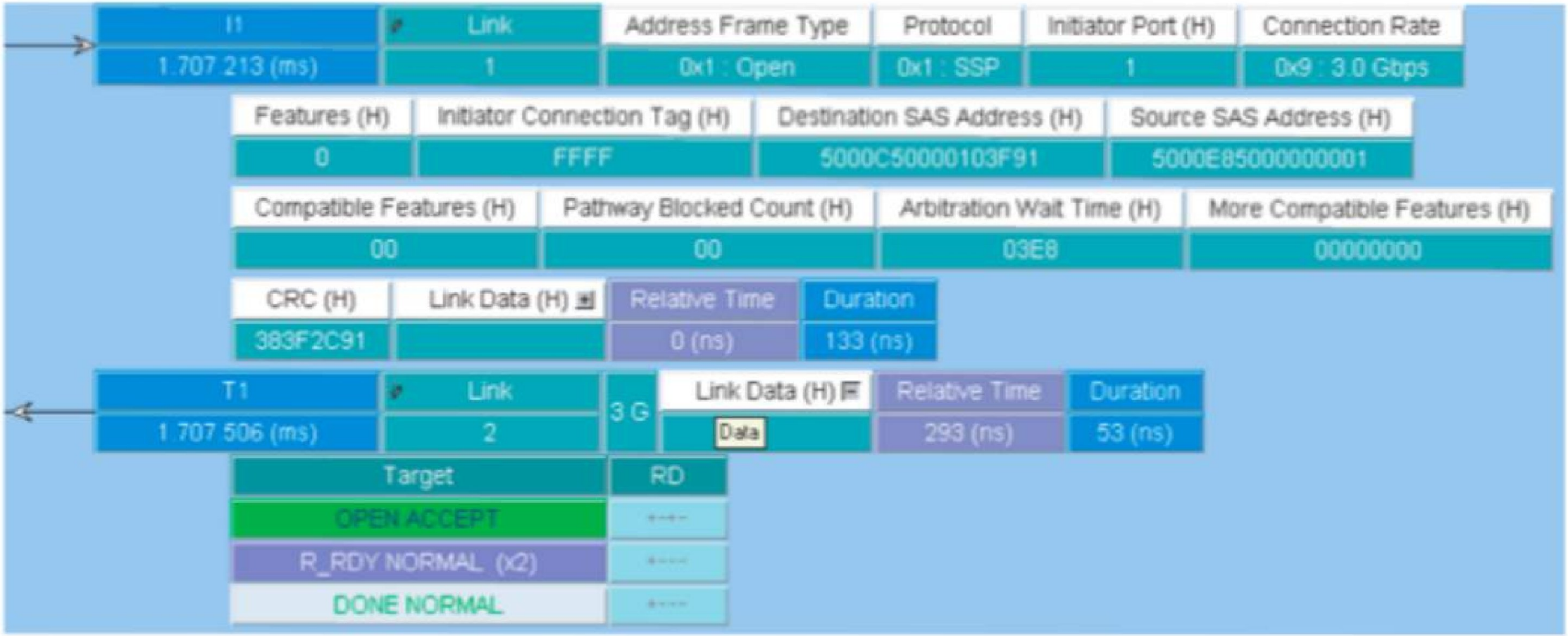
ed bus;

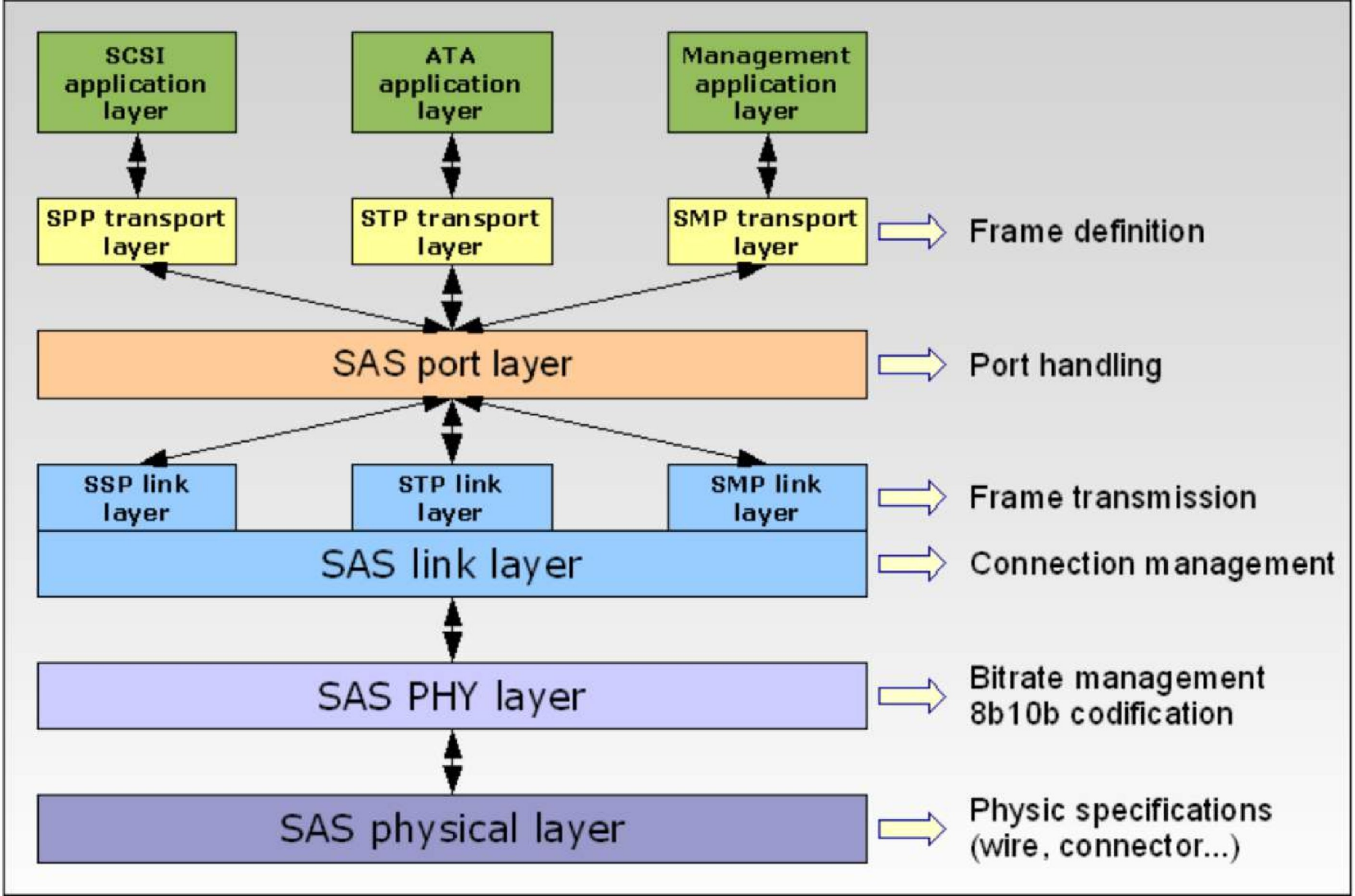
es;

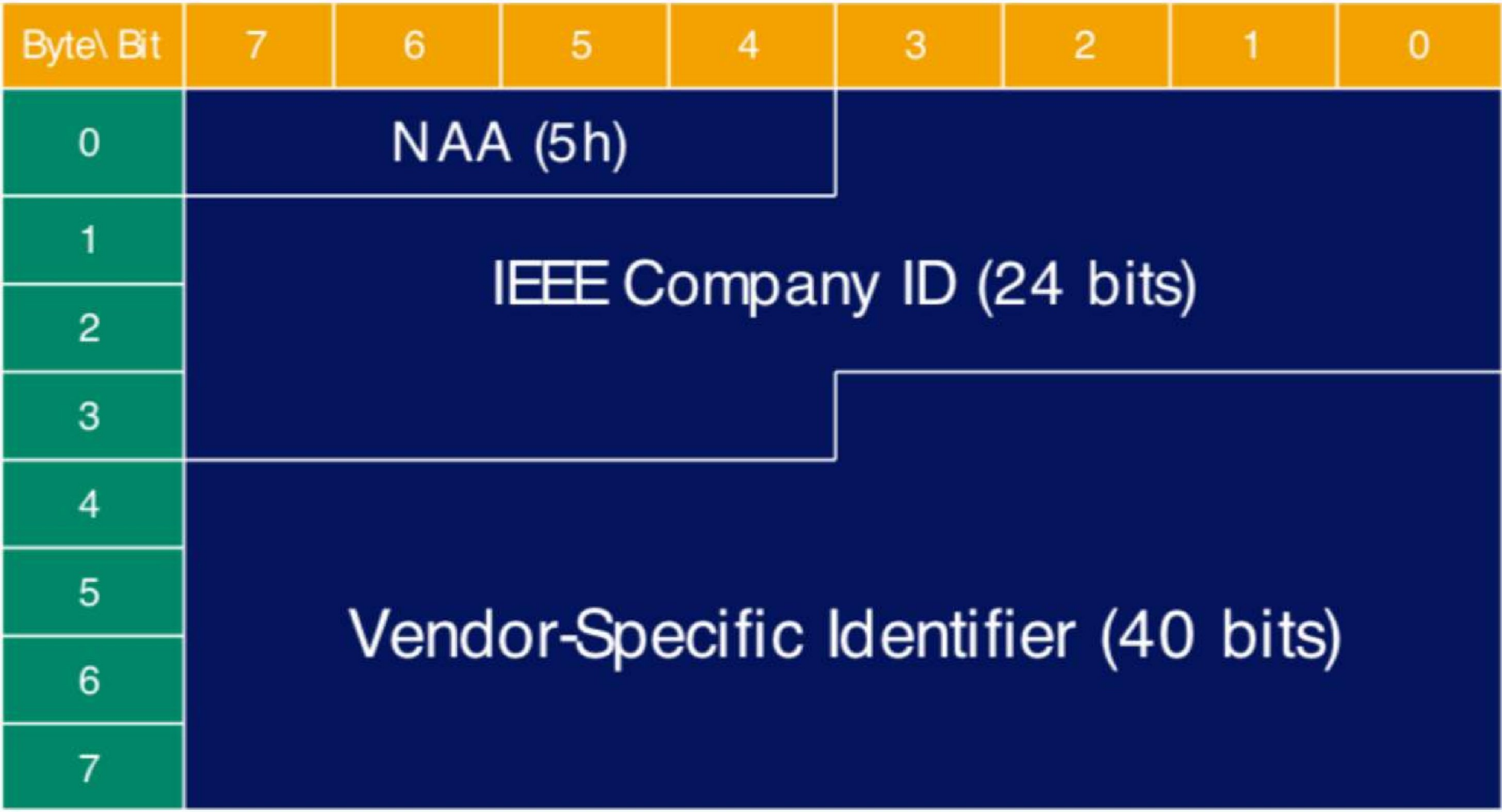

APPLICATION LAYER
SSP Support
STP Support
SMP Support
SAS HBA Driver
SAS HBA
Micro Engine & Firmware
SAS HBA Register I/F
PORT
TRANSPORT LAYER
SSP Support
STP Support
SMP Support
PORT LAYER
LINK LAYER
SSP Support
STP Support
SMP Support
PHY
PHY LAYER
PHYSICAL LAYER

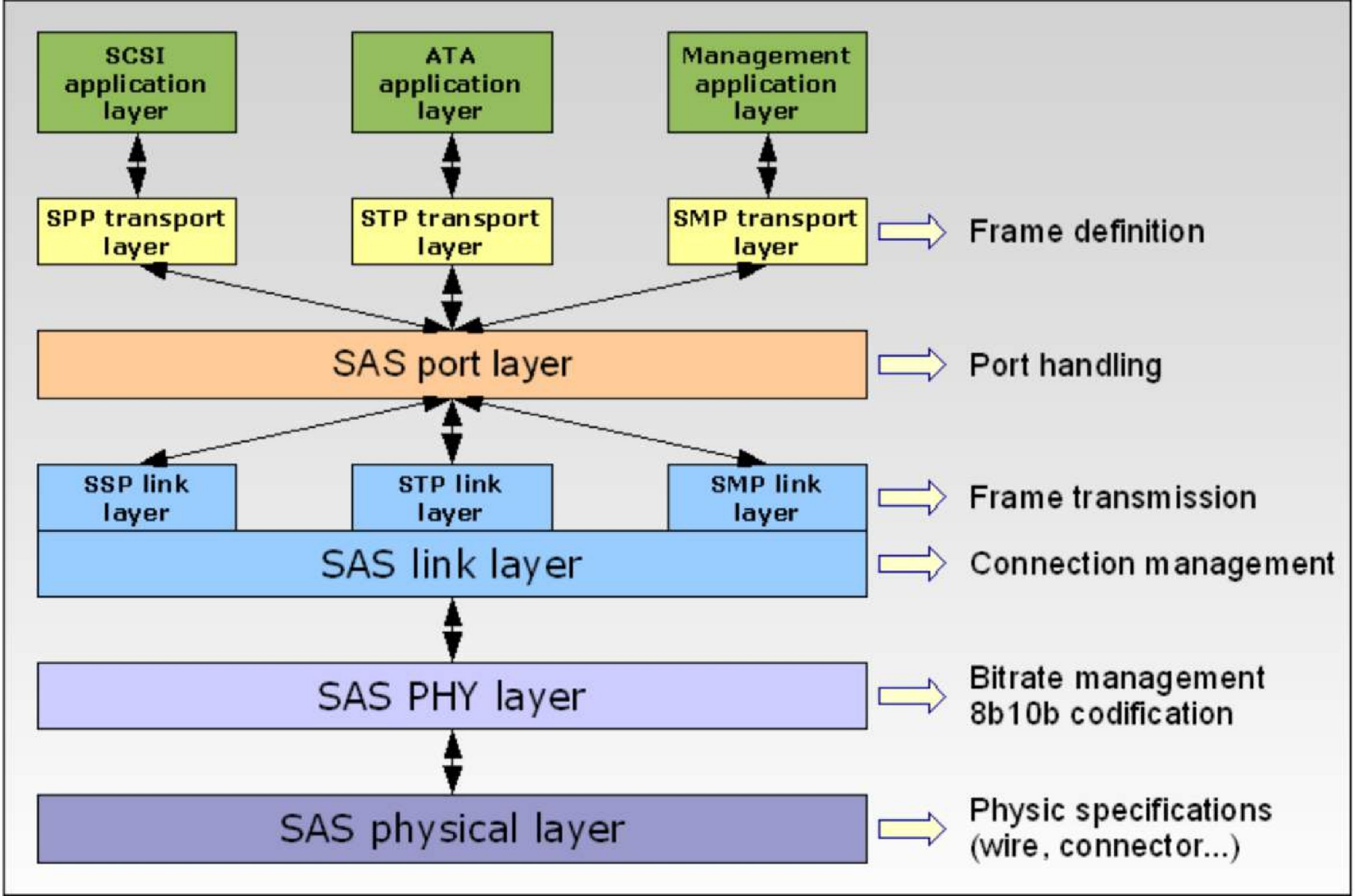

SSP link
layer


SCSI application layer
ATA application layer
Management application layer
SPP transport layer
STP transport layer
SMP transport layer
Frame definition
SAS port layer
Port handling
SSP link layer
STP link layer
SMP link layer
Frame transmission
SAS link layer
Connection management
SAS PHY layer
Bitrate management
8b10b codification
SAS physical layer
Physic specifications
(wire, connector...)

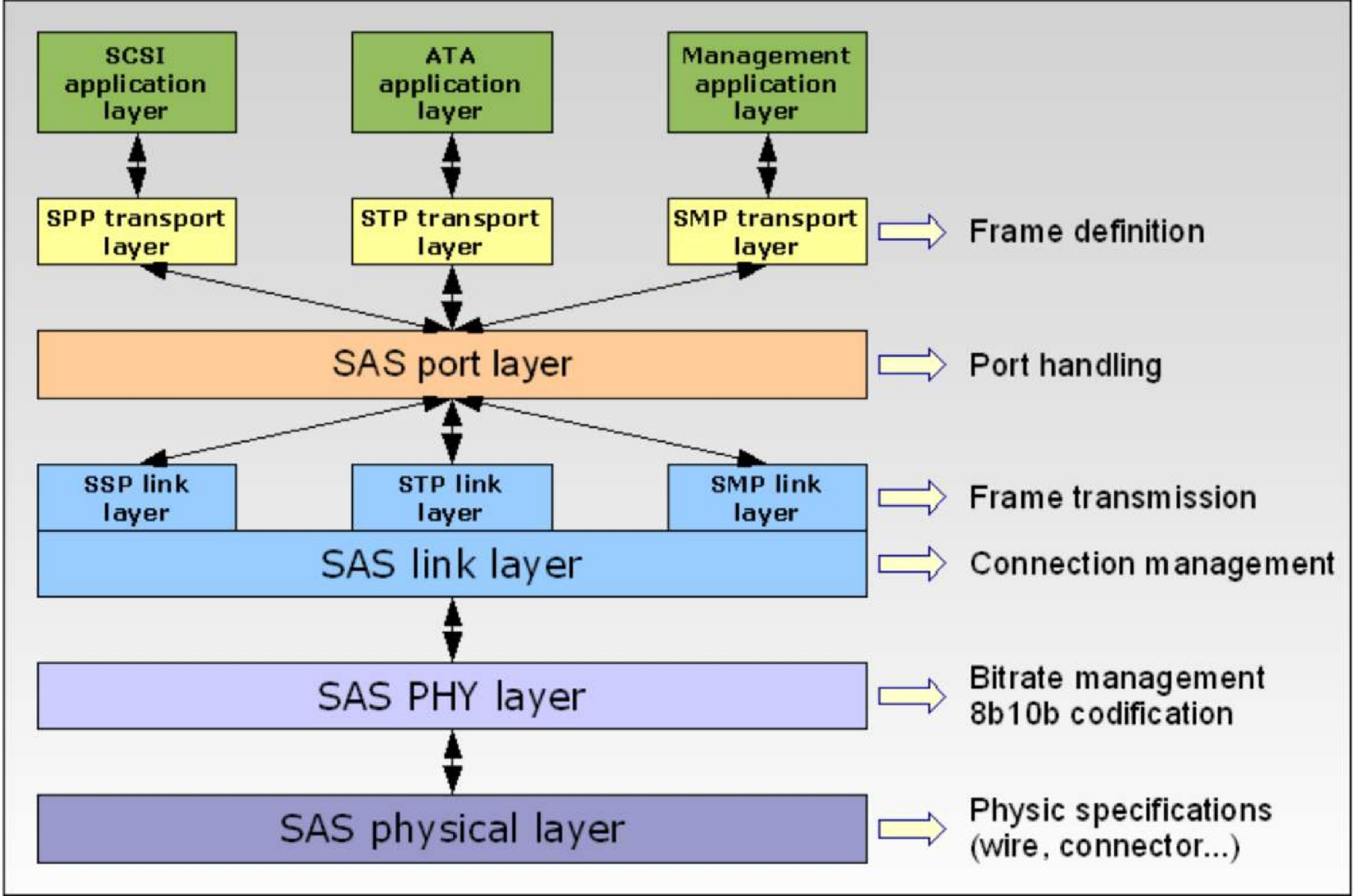

I1
Link
Address Frame Type
Protocol
Initiator Port (H)
Connection Rate
1.707 213 (ms)
1
0x1 : Open
0x1 : SSP
1
0x9 : 3.0 Gbps
Features (H)
Initiator Connection Tag (H)
Destination SAS Address (H)
Source SAS Address (H)
0
FFFF
5000C50000103F91
5000E85000000001
Compatible Features (H)
Pathway Blocked Count (H)
Arbitration Wait Time (H)
More Compatible Features (H)
00
00
03E8
00000000
CRC (H)
Link Data (H)
Relative Time
Duration
383F2C91
0 (ns)
133 (ns)
T1
Link
3 G
Link Data (H)
Relative Time
Duration
1.707 506 (ms)
2
Data
293 (ns)
53 (ns)
Target
RD
OPEN ACCEPT
R_RDY NORMAL (x2)
DONE NORMAL


SCSI application layer
ATA application layer
Management application layer
SPP transport layer
STP transport layer
SMP transport layer
Frame definition
SAS port layer
Port handling
SSP link layer
STP link layer
SMP link layer
Frame transmission
SAS link layer
Connection management
SAS PHY layer
Bitrate management
8b10b codification
SAS physical layer
Physic specifications
(wire, connector...)


Byte\ Bit
7
6
5
4
3
2
1
0
0
NAA (5h)
1
IEEE Company ID (24 bits)
2
3
4
Vendor-Specific Identifier (40 bits)
5
6
7


SCSI application layer
ATA application layer
Management application layer
SPP transport layer
STP transport layer
SMP transport layer
Frame definition
SAS port layer
Port handling
SSP link layer
STP link layer
SMP link layer
Frame transmission
SAS link layer
Connection management
SAS PHY layer
Bitrate management
8b10b codification
SAS physical layer
Physic specifications
(wire, connector...)

Byte	Field(s)		
0	Frame Type		
1 to 3	Hashed Destination SAS address		
4	Reserved		
5 to 7	Hashed Source SAS address		
8 to 9	Reserved		
10	Reserved	Retransmit	Rsvd
11	Reserved	Number of Fill Bytes	
12 to 15	Reserved		
16 to 17	Tag		
18 to19	Target Port Transfer Tag		
20 to 23	Data Offset		
24 to m	Information Unit		
m to (n-3)	Fill bytes, if needed		
(n-3) to n	CRC		


SCSI application layer
ATA application layer
Management application layer
SPP transport layer
STP transport layer
SMP transport layer
Frame definition
SAS port layer
Port handling
SSP link layer
STP link layer
SMP link layer
Frame transmission
SAS link layer
Connection management
SAS PHY layer
Bitrate management
8b10b codification
SAS physical layer
Physic specifications
(wire, connector...)

Serial SCSI Protocol:

- Wraps SCSI commands into serialized frames;
- Read Command, Data from target, Response;
- Write Command, XFER_RDY, Data to Target, Response;
- Nondata, Command, Response.

SATA Transport Protocol:

- Wraps SATA commands;
- Tunneling;
- Uses underlying SAS layers for transport;
- FIS.

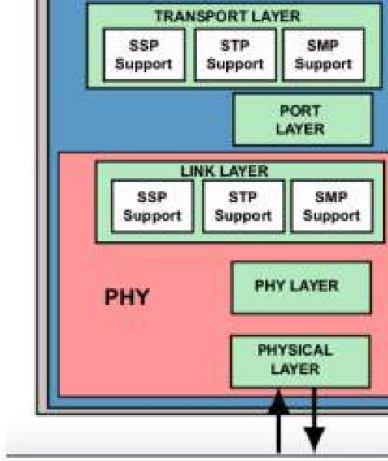
- **narrow** and **wide** ports;

SAS and **SCSI**:

- **P2P** vs **multidrop** bus;
- **no termination** vs **terminated bus**;
- **65535** devices vs **8/16** devices;
- dedicated **full bandwidth** vs. **shared bus** bandwidth;

SAS and **SATA**:

- dedicated **full bandwidth** vs. **shared bus** bandwidth;



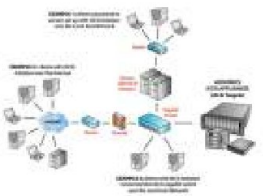
SAS and **SATA**:

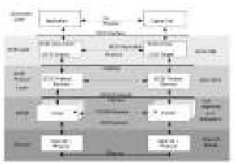
- **multiple initiators** vs **single initiator**;
- **TCQ** vs **NCQ**;
- **SCSI command set** vs **ATA command set**;
- **multipath** IO vs **single path** IO;
- **1.6V** signaling vs **0.6V** signaling;
- up to **10 meters** vs **1 meter** cabling;

https://www.also.com/pub/pdf/hp
https://www.mindshare.com/files

iSCSI Protocol

- Internet SCSI;
- **Block level access over TCP/IP network**;
- Ports 860/3260;
- Purposes:
  - storage consolidation;
  - disaster recovery;
  - iSCSI **initiator** - iSCSI **target**;
- Types:
  - hardware;
    - dedicated;
    - offload engines;
  - software solutions;
- RFC 3720;





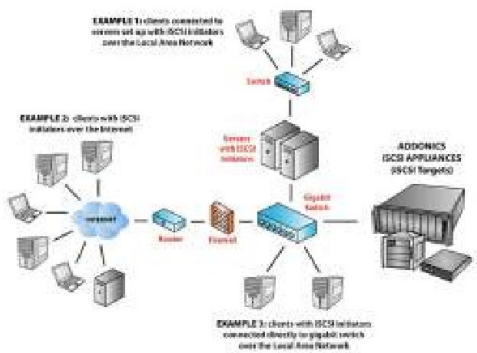
Addressing:
- ISCSI Qualified Name;
- iqn-yyyy-mm-reversed_domain:opt
- iqn-**2019-04-hu.uni-obuda**:**storage1**

Security:
- Challenge Handshake Authentication Protocol;
- filtering;
- multipathing;
- logical vs physical isolation;
- SSL over IP;
- **no zoning!**

Protocol extensions:
- **MC/S**: Allows multiple TCP/IP connections to set up an iSCSI session;
- Needs additional flow control in the protocol stack!
- Data layer + Basic Header Segments;

https://www.snia.org/sites/default/education/tutorials/2011/spring/networking/HufferdJohn-IP_Storage_Protocols-iSCSI.pdf

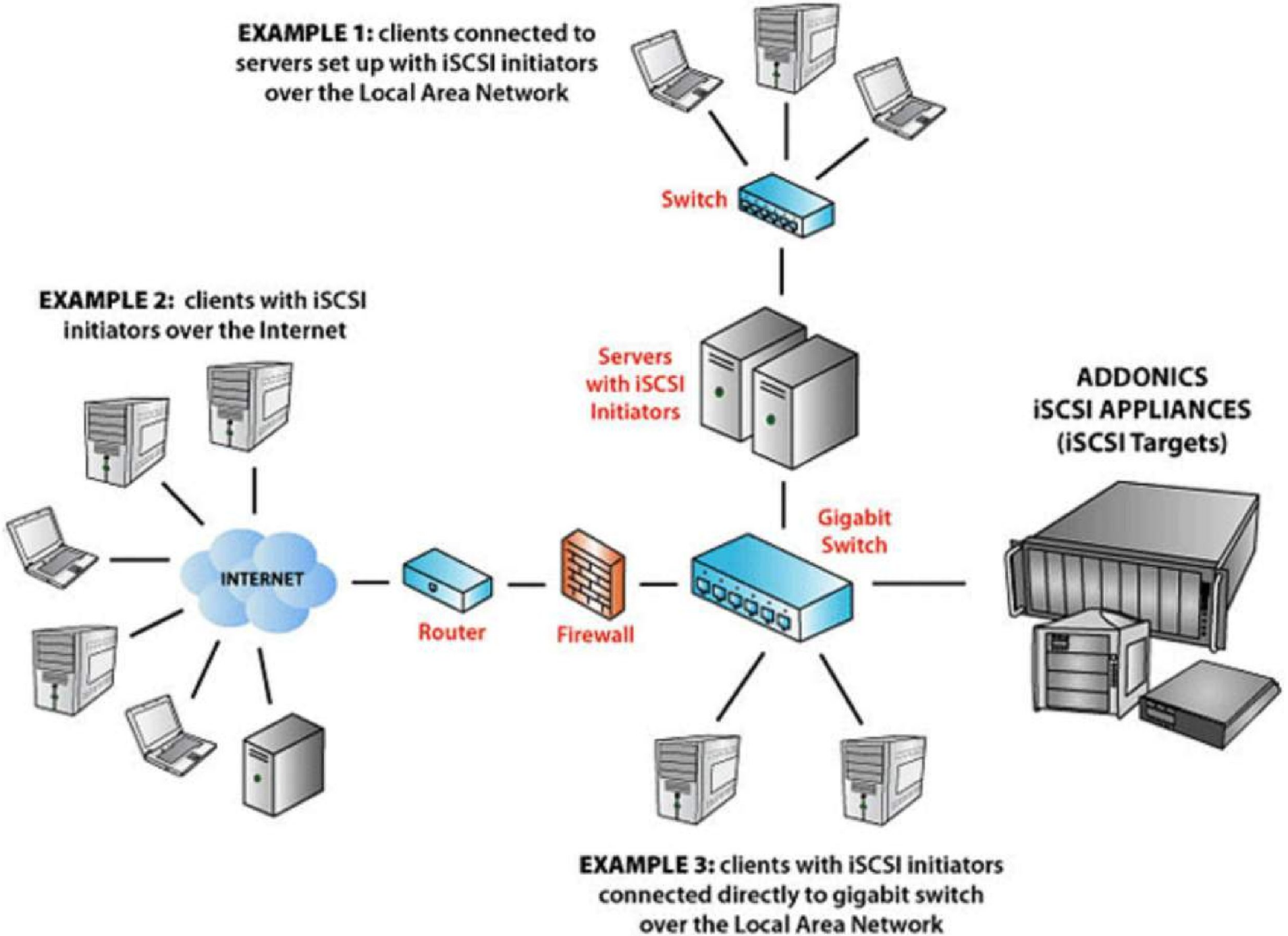
- Internet SCSI;
- **Block level access over TCP/IP network**;
- Ports 860/3260;
- Purposes:
  - storage consolidation;
  - disaster recovery;
  - iSCSI **initiator** - iSCSI **target**;
- Types:
  - hardware;
    - dedicated;
    - offload engines;
  - software solutions;
- RFC 3720;

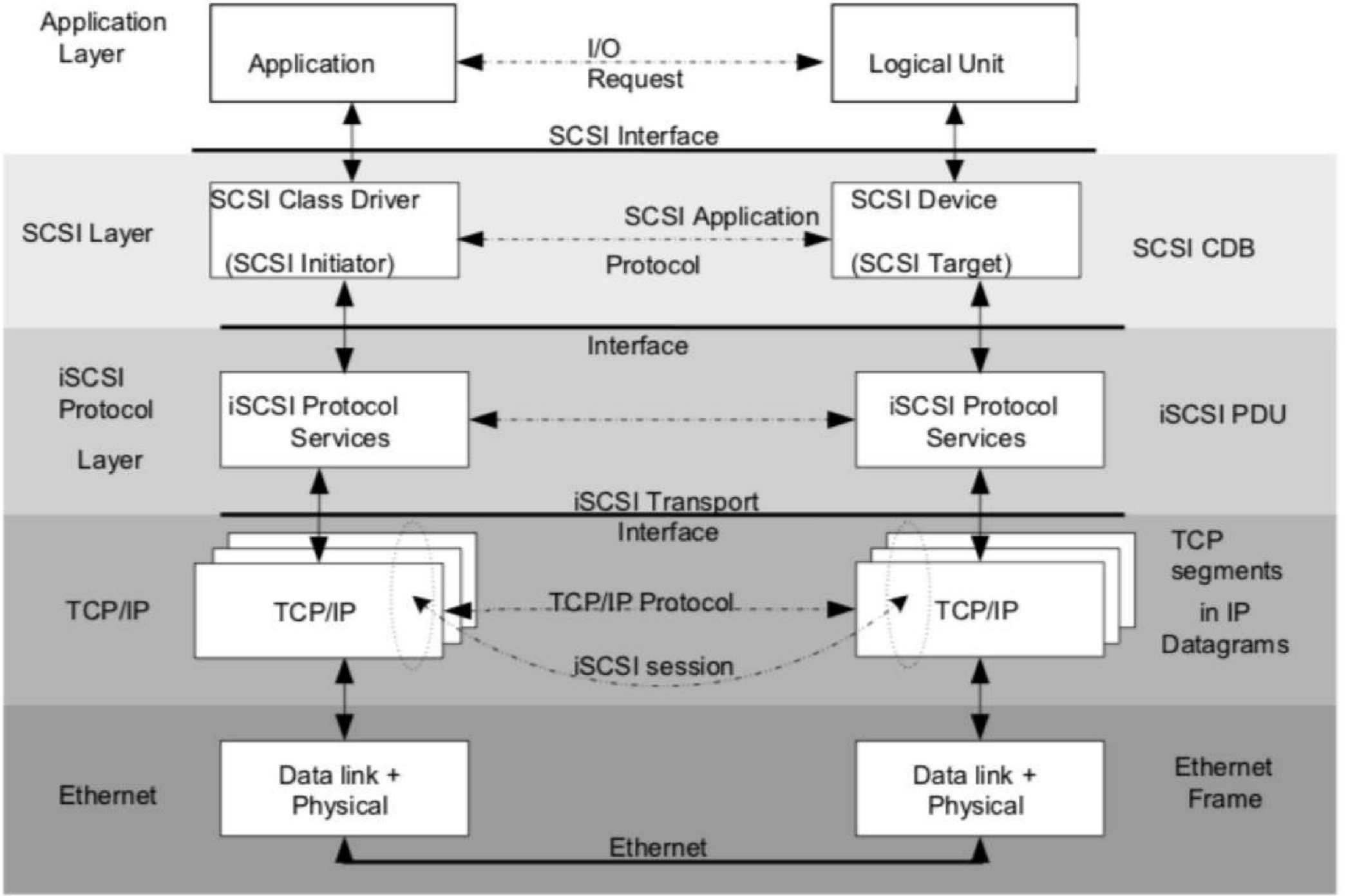


Address
- iSCS
- iqn-yy
- iqn-20

Security
- Challe
- filterin
- multip
- logica
- SSL o
- **no zo**

Protocol
- **MC/S** conne iSCSI
- Need in the


EXAMPLE 1: clients connected to servers set up with iSCSI initiators over the Local Area Network
Switch
EXAMPLE 2: clients with iSCSI initiators over the Internet
Servers with iSCSI Initiators
ADDONICS iSCSI APPLIANCES (iSCSI Targets)
Gigabit Switch
INTERNET
Router
Firewall
EXAMPLE 3: clients with iSCSI initiators connected directly to gigabit switch over the Local Area Network


Application Layer
Application
I/O Request
Logical Unit
SCSI Interface
SCSI Layer
SCSI Class Driver
(SCSI Initiator)
SCSI Application Protocol
SCSI Device
(SCSI Target)
SCSI CDB
Interface
iSCSI Protocol Layer
iSCSI Protocol Services
iSCSI Protocol Services
iSCSI PDU
iSCSI Transport Interface
TCP/IP
TCP/IP
TCP/IP Protocol
TCP/IP
iSCSI session
TCP segments in IP Datagrams
Ethernet
Data link + Physical
Data link + Physical
Ethernet Frame
Ethernet

Addressing:

- ISCSI Qualified Name;
- iqn-yyyy-mm-reversed_domain:opt
- iqn-**2019-04**-**hu.uni-obuda**:**storage1**

Security:

- Challenge Handshake Authentication Proto

- iSCSI Qualified Name;
- iqn-yyyy-mm-reversed_domain:opt
- iqn-**2019-04**-**hu.uni-obuda**:**storage1**

Security:

- **C**hallenge **H**andshake **A**uthentication **P**rotocol;
- filtering;
- multipathing;
- logical vs physical isolation;
- SSL over IP;
- **no zoning!**

Protocol extensions:

- **MC/S**: Allows multiple TCP/IP connections to set up an

Protocol extensions:

- **MC/S**: Allows multiple TCP/IP connections to set up an iSCSI session;
- Needs additional flow control in the protocol stack!
- Data layer + Basic Header Segments;

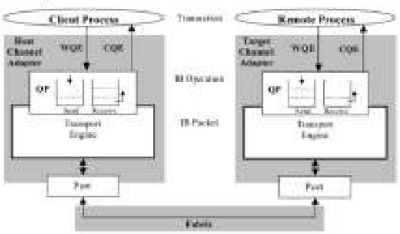
Byte	0	1	2	3
Bit	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
0 to 3	.|I| Opcode	Opcode-Specific Fields		
4 to 7	Total AHS Length	Data Segment Length		
8 to 15	Logical Unit Number (LUN) or Opcode-Specific Fields			
16 to 19	Initiator Task Tag (ITT) or Opcode-Specific Fields			
20 to 31	Opcode-Specific Fields			
32 to 47	Command Descriptor Block (CDB) or Opcode-Specific Fields			

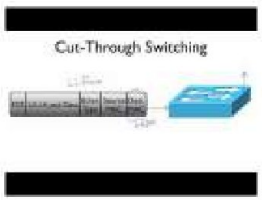
Infiniband

- A storage data network optimized for high bandwidth and low latency;
- Infiniband Trade Association including major companies;
- SDR [2Gbps, 5µs], DDR [4Gbps, 2.5µs], QDR [8Gbps, 1.3µs], FDR [14Gpbs, 700ns], EDR [24Gbps, 500ns], HDR [48Gbps, <500ns];
- Ports can be **trunked to multiply bandwidth**: 1x, 4x, 12x;
- **Non-blocking** switching;
- Switched network, with **Host Channel Adapter**, and **Target Channel Adapter**;
- Physical connections: CX4, SFP, CXP;
- **4k** packets;

Virtual Interface Architecture:

- distributed messaging technology;
- it offloads traffic control from the client to dedicated execution queues;
- **WQP** is assigned to the transmission and reception side;
- Work Queue Entries -> Completion Queue Entries;





Messages:

- RDMA;
- Channel send and receive;
- Transaction;
- Multicast message;
- Atomic operation.

Layers:

- **Physical:** cables + interfaces;
- **Link:** Local IDs, Virtual Lanes (VL0-15);
- **Network:** Global Route Headers between subnets, IPv6 headers are used;
- **Transport:** in-order delivery, partitioning, channel multiplexing;

Why IB is faster?

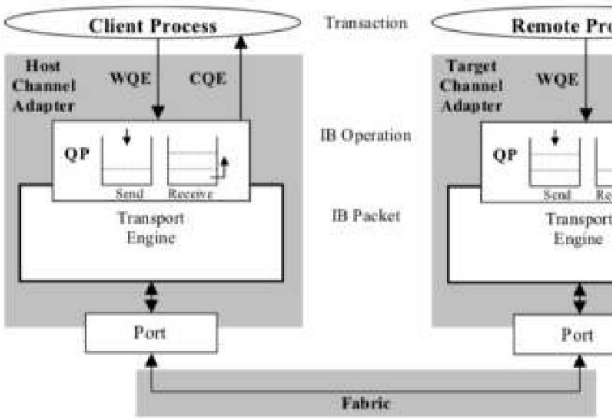
- **Store and forward** switch: stores entire packets and makes decision then;
- **Cut through** switch: stores only headers, not full data frames -> **low latency**;

https://en.wikipedia.org/wiki/Cut-through_switching

https://www.mellanox.com/pdf/whitepapers/IB_Intro_WP_190.pdf
https://www.arista.com/assets/data/pdf/Infiniband-vs-ethernet.pdf

Infiniband

- A storage data network optimized for high bandwidth and low latency;
- **I**nfiniband **T**rade **A**ssociation including major companies;
- SDR **[2Gbps, 5µs]**, DDR **[4Gbps, 2.5µs]**, QDR **[8Gbps, 1.3µs]**, FDR **[14Gpbs, 700ns]**, EDR **[24Gbps, 500ns]**, HDR **[48Gbps, <500ns]**;
- Ports can be **trunked to multiply bandwidth**: **1x, 4x, 12x**;
- **Non-blocking** switching;
- Switched network, with **Host Channel Adapter**, and **Target Channel Adapter**;
- Physical connections: **CX4, SFP, CXP**;
- **4k** packets;



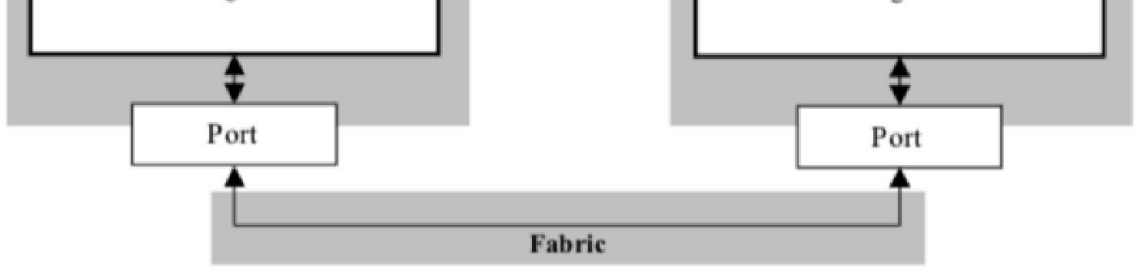
Messages:

- RDMA;

Layers:

Messages:

- RDMA;
- Channel send and receive;
- Transaction;
- Multicast message;
- Atomic operation.



Layers:

- **Physical:** cables + interfaces;
- **Link:** Local IDs, Virtual Lanes (VL0-15);
- **Network:** Global Route Headers between subnets, IPv6 headers are used;
- **Transport:** in-order delivery, partitioning, channel multiplexing;

.mellanox.com/pdf/whitepapers/IB_Intro_WP_

arista.com/assets/data/pdf/Infiniband-vs-ethe

Virtual Interface Architecture:

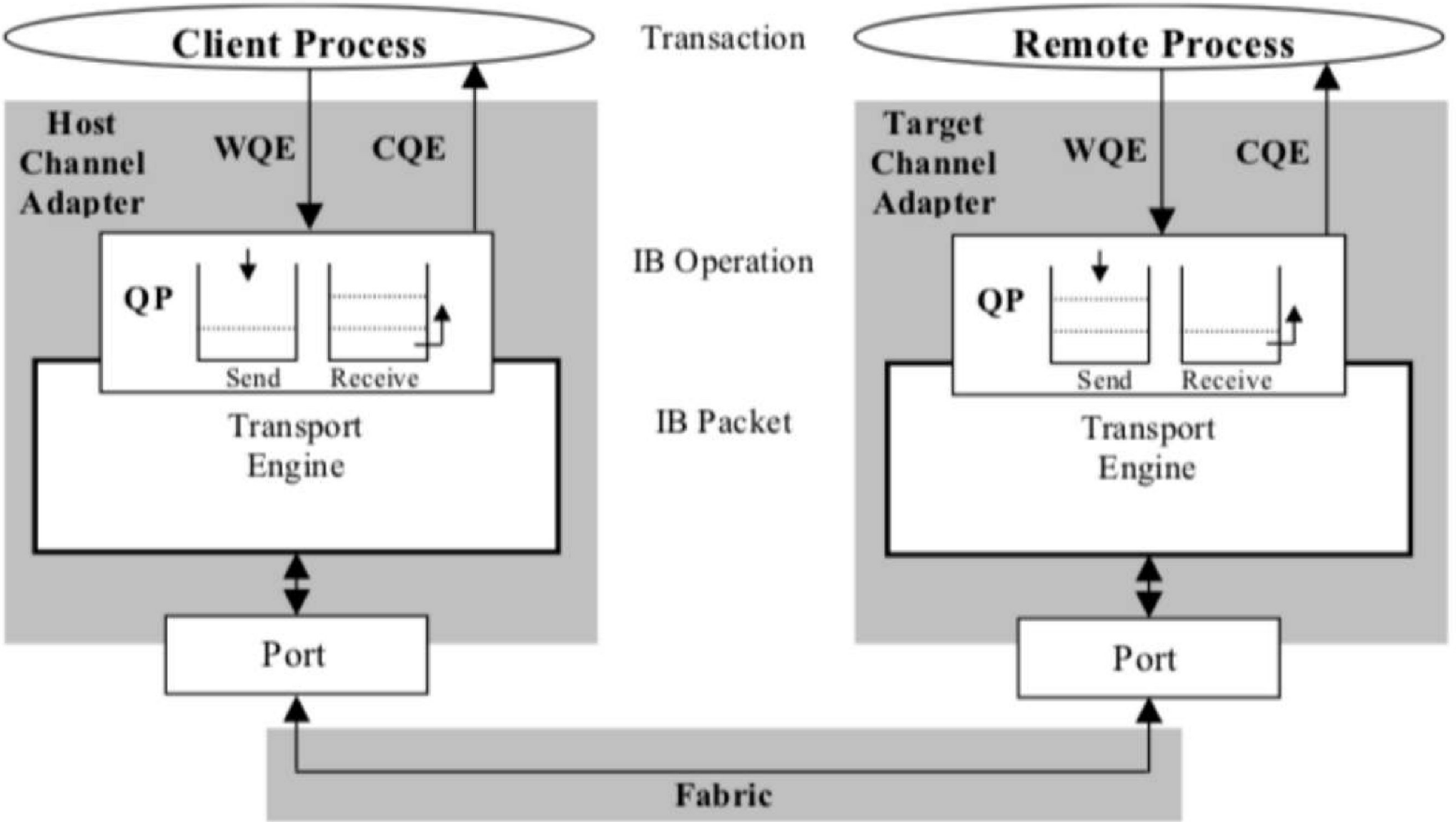
- distributed messaging technology;
- it offloads traffic control from the client to dedicated execution queues;
- **WQP** is assigned to the transmission and reception side;
- **W**ork **Q**ueue **E**ntries -> **C**ompletion **Q**ueue **E**ntries;

Why IB is faster?

- **Store and forward** switch: stores entire packets and makes decision then;
- **Cut through** switch: stores only headers, not full data frames -> **low latency**;

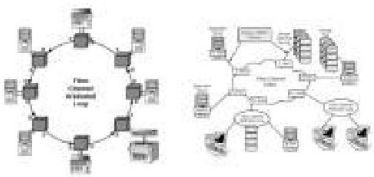
https://en.wikipedia.org/wiki/Cut-through_switching

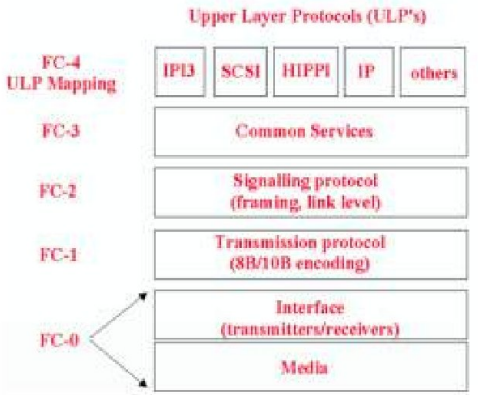
ptel, and


Client Process
Transaction
Remote Process
Host Channel Adapter
WQE
CQE
Target Channel Adapter
WQE
CQE
QP
IB Operation
QP
Send
Receive
Send
Receive
Transport Engine
IB Packet
Transport Engine
Port
Port
Fabric

Fibre Channel Protocol

- Delivering **SCSI command set** over FC network.
- Besides SCSI: HIPPI, ATM, IP, NVMe.
- Bandwidth: 1-32 Gb/s. We have 4 and 8 Gb/s.
- **Addressing:**
  - 3 byte addresses are used
  - FFFFFA - fabric manager service
  - FFFFFC - fabric name service
  - FFFFFD - fabric controller service
  - FFFFFE - fabric login service
- **Principal switch:** responsible for managing and distributing IDs within the domain.
- **Flow control:** throttling traffic.
- **Topologies:**
  - FC-PP: point-to-point;
  - FC-AL: arbitrated loop, ring;
  - FC-SW: switched fabric.





Protocol exchanges:

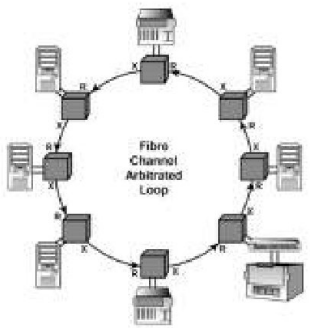
- **Fabric Login (FLOGI):** node enters the fabric;
- **Port Login (PLOGI):** session between initiator port and targer port;
- **Process login (PRI):** sending SCSI commands.

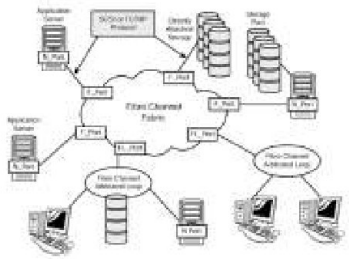
Frame structure:

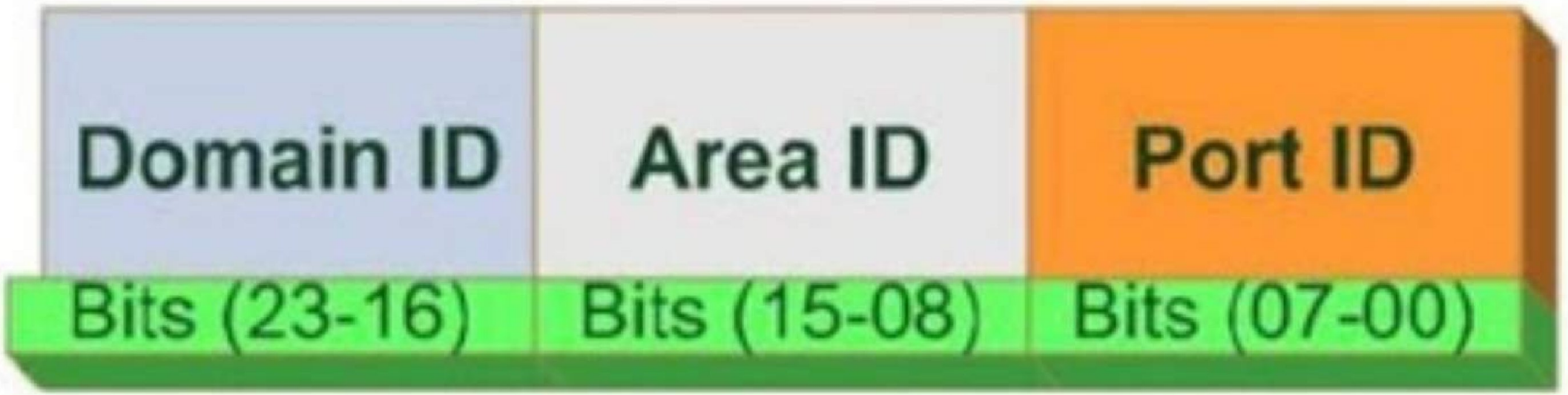
- Exchange: a session;
- Sequence: a set of frames;
- Frame:
  - data;
  - control (e.g. FSPF).

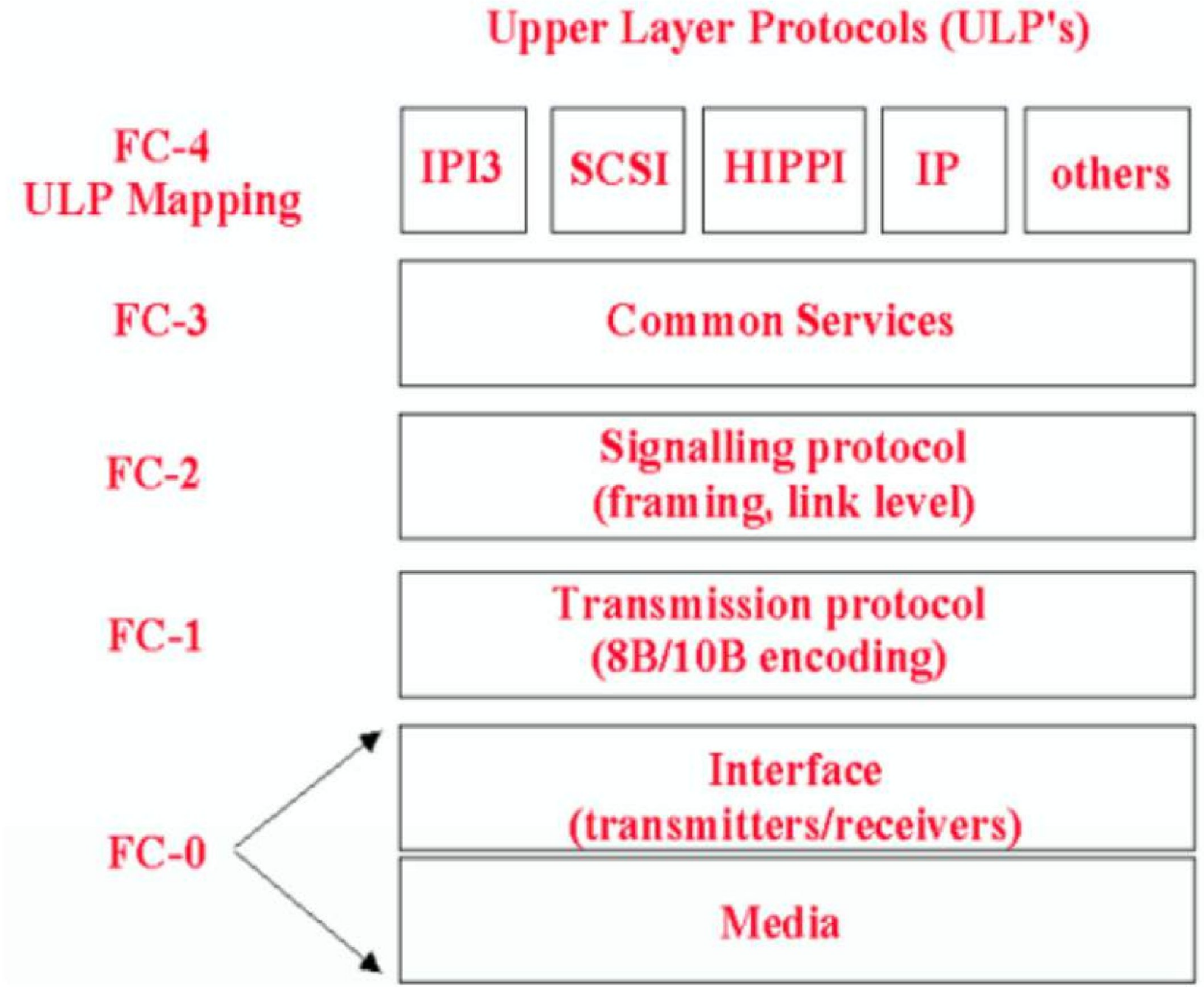
http://www.tsmtutorials.com/2016/08/fc-san-protocols.html
https://en.wikipedia.org/wiki/Fibre_Channel

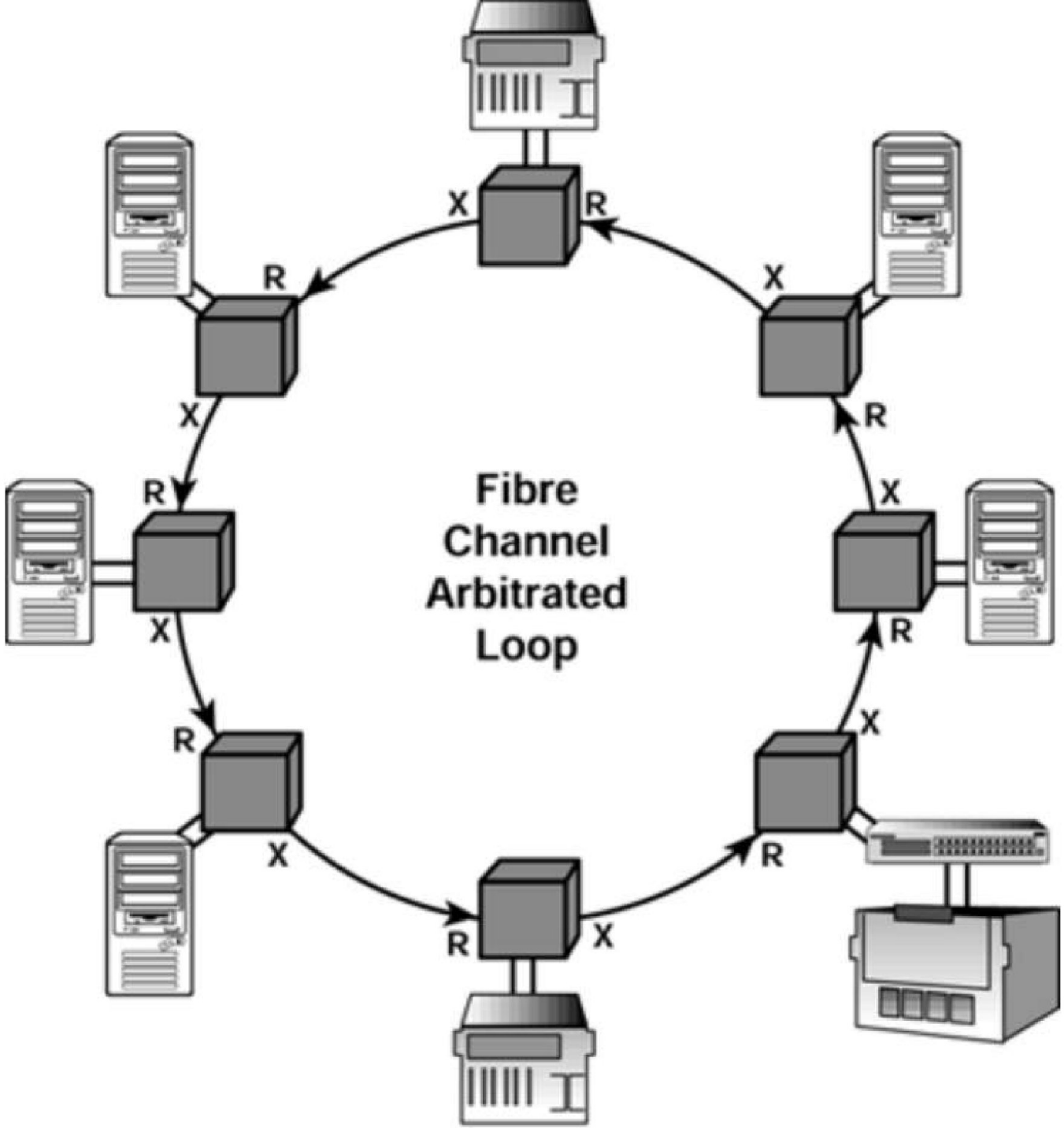
- Delivering **SCSI command set** over FC network.
- Besides SCSI: HIPPI, ATM, IP, NVMe.
- Bandwidth: **1-32 Gb/s**. We have 4 and 8 Gb/s.
- **Addressing:**
  - 3 byte addresses are used
  - **FFFFFA** - fabric manager service
  - **FFFFFC** - fabric name service
  - **FFFFFD** - fabric controller service
  - **FFFFFE** - fabric login service
- **Principal switch:** responsible for managing and distributing IDs within the domain.
- **Flow control:** throttling traffic.
- **Topologies:**
  - **FC-PP**: point-to-point;
  - **FC-AL**: arbitrated loop, ring;
  - **FC-SW**: switched fabric.

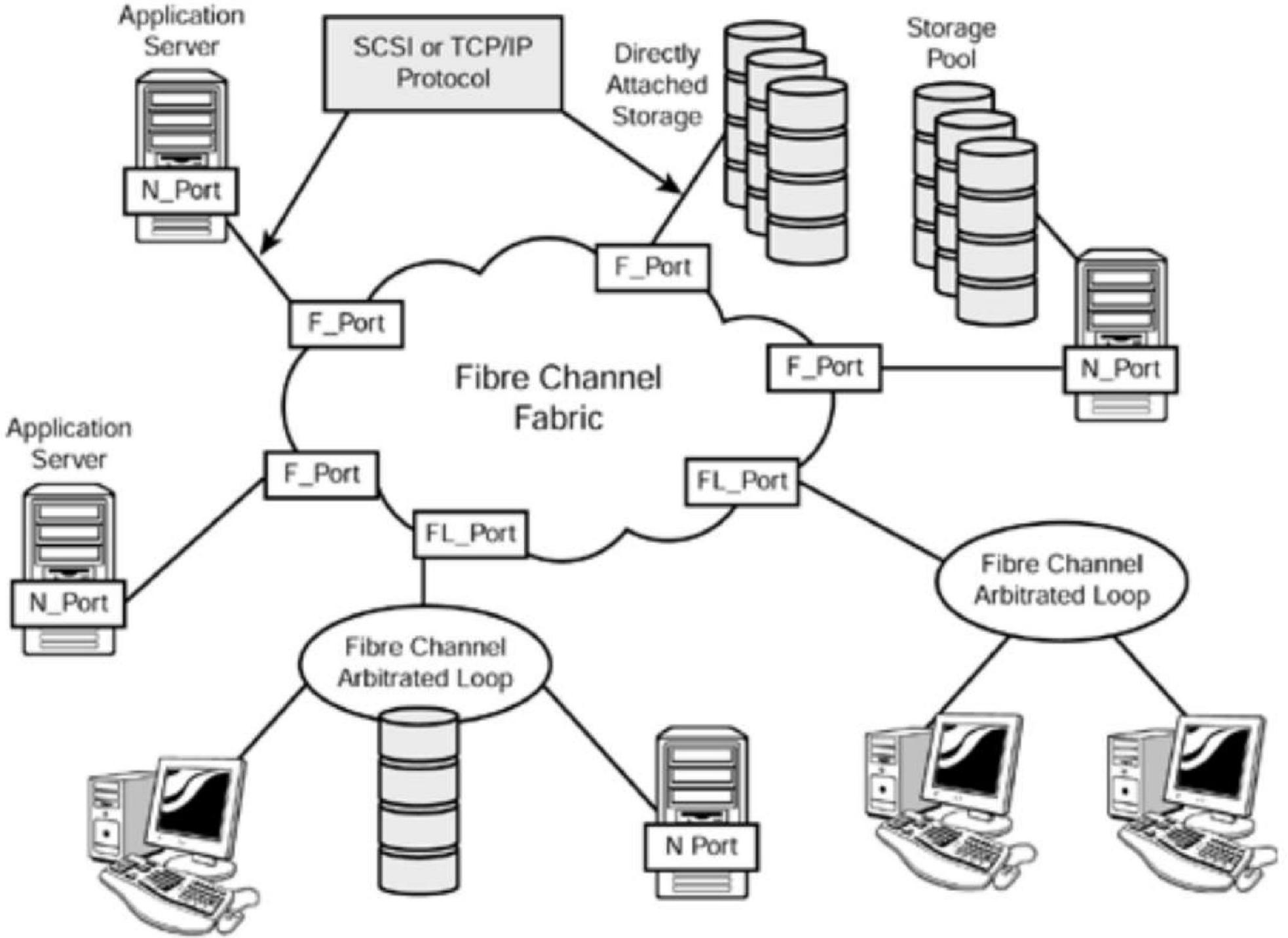





Domain ID
Area ID
Port ID
Bits (23-16)
Bits (15-08)
Bits (07-00)

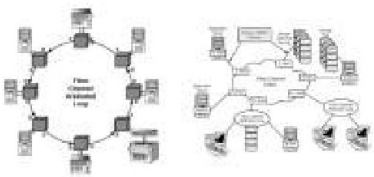

Upper Layer Protocols (ULP's)
FC-4
ULP Mapping
IPI3
SCSI
HIPPI
IP
others
FC-3
Common Services
FC-2
Signalling protocol
(framing, link level)
FC-1
Transmission protocol
(8B/10B encoding)
FC-0
Interface
(transmitters/receivers)
Media

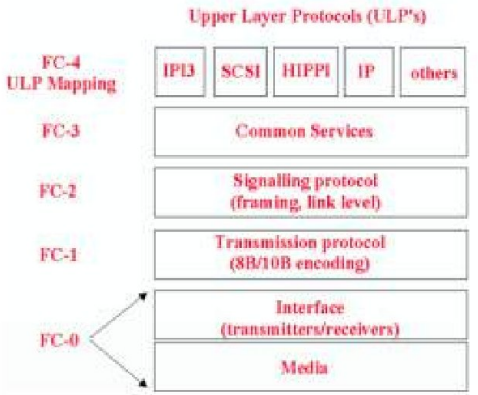

Fibre
Channel
Arbitrated
Loop


Application Server
SCSI or TCP/IP Protocol
Directly Attached Storage
Storage Pool
N_Port
F_Port
F_Port
F_Port
F_Port
N_Port
Fibre Channel Fabric
Application Server
F_Port
FL_Port
FL_Port
N_Port
Fibre Channel Arbitrated Loop
Fibre Channel Arbitrated Loop
N Port

Fibre Channel Protocol

- Delivering **SCSI command set** over FC network.
- Besides SCSI: HIPPI, ATM, IP, NVMe.
- Bandwidth: 1-32 Gb/s. We have 4 and 8 Gb/s.
- **Addressing:**
  - 3 byte addresses are used
  - FFFFFA - fabric manager service
  - FFFFFC - fabric name service
  - FFFFFD - fabric controller service
  - FFFFFE - fabric login service
- **Principal switch:** responsible for managing and distributing IDs within the domain.
- **Flow control:** throttling traffic.
- **Topologies:**
  - FC-PP: point-to-point;
  - FC-AL: arbitrated loop, ring;
  - FC-SW: switched fabric.





Protocol exchanges:

- **Fabric Login (FLOGI):** node enters the fabric;
- **Port Login (PLOGI):** session between initiator port and targer port;
- **Process login (PRI):** sending SCSI commands.

Frame structure:

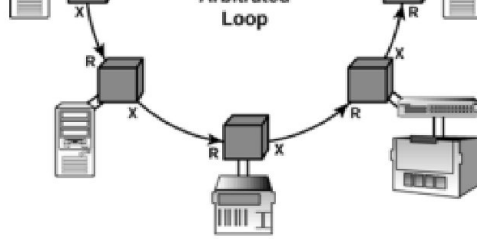
- Exchange: a session;
- Sequence: a set of frames;
- Frame:
  - data;
  - control (e.g. FSPF).

http://www.tsmtutorials.com/2016/08/fc-san-protocols.html
https://en.wikipedia.org/wiki/Fibre_Channel

Frame structure:

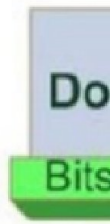
- Exchange: a session;
- Sequence: a set of frames;
- Frame:
  - data;
  - control (e.g. FSPF).

- **FC-SW**: switched fabric.



Protocol exchanges:

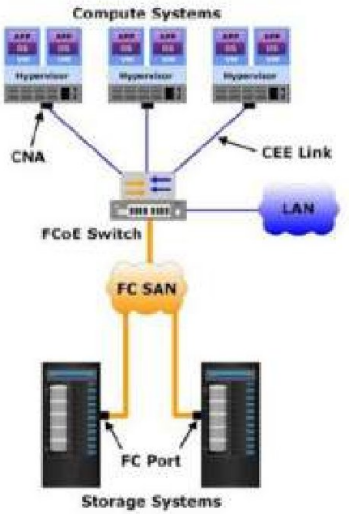
- **Fabric Login (FLOGI):** node enters the fabric;
- **Port Login (PLOGI):** session between initiator port and targer port;
- **Process login (PRI):** sending SCSI commands.

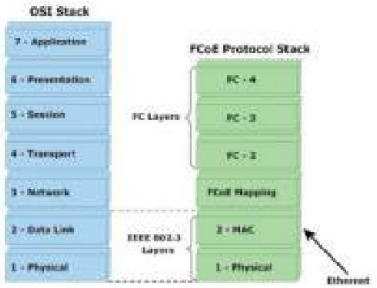


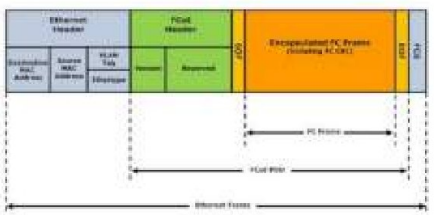
http://www.tsmtutorials.com/

FC over Ethernet Protocol

- Encapsulates **FC data into Ethernet frames**;
- Uses **multi-functional HBAs** and **switches**;
- **Adapters, cables, switches:**
  - **Converged Network Adapter**: combines Ethernet and FC functionality in the same device;
  - **Software FCoE Adapter**: uses kernel modules to process FCoE traffic;
- **Fibre Channel Forwarder:**
  - Resides in a combined Ethernet and FC switch;
  - Provides FC services: zoning, name resolving, etc.
- The same port types: **VN_Port, VF_Port, VE_Port**.





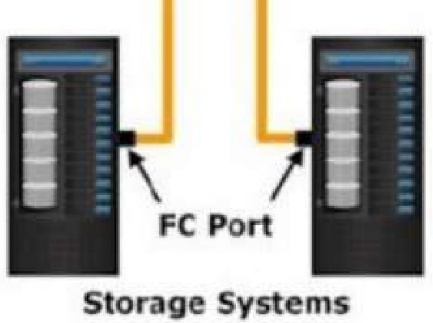


FCoE addressing:

- MAC addresses are used;
- jumbo frames: 2112 bytes vs. 1500 bytes;
  - **SPMA**: compute servers provide MAC addresses;
  - **FPMA**: fabrics provide MAC addresses (MAC:FC addresses concatenated);
- **Discovery (FIP)** - **Login** - **Data forwarding**.

Discovery and Login:
- Multicast Frame Initializing Packet messages to find FCF;
- FCFs send FIP Advertisement frames;
- Nodes send FLOGI requests to the closest FCF;

ed Ethernet and FC switch;
s: zoning, name resolving, etc.
N_Port, VF_Port, VE_Port.



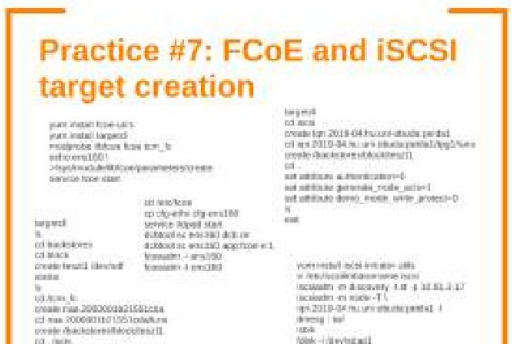
FCoE addressing:

- MAC addresses are used;
- jumbo frames: 2112 bytes vs. 1500 bytes;
  - **SPMA**: compute servers provide MAC addresses;
  - **FPMA**: fabrics provide MAC addresses (MAC:FC addresses concatenated);
- **Discovery (FIP)** - **Login** - **Data forwarding**.

Discovery and Login:

- Multicast Frame Initializing Packet messages to find FCF;
- FCFs send FIP Advertisement frames;
- Nodes send FLOGI requests to the closest FCF;

Practice #7: FCoE and iSCSI target creation



Discovery and Login:

- Multicast Frame Initializing Packet messages to find FCF;
- FCFs send FIP Advertisement frames;
- Nodes send FLOGI requests to the closest FCF;

Volume management

create flexible volumes and mappings on nodes

Logical Volume Management

a more flexible way of allocating block device resources than in traditional way, i.e. partitioning

- Divides physical space to chunks called Physical Extents (PE);
- Map PEs to Logical Extents (LE);
- Elementary terms:
  - **Physical Volume(s) (PV)**: a set of devices formulating the physical space;
  - **Volume Group (VG)**: a pool of resources, sum of all PEs serving the same purpose;
  - **Logical Volume(s) (LV)**: a set of dynamically allocated LEs;
- PE + LE have fixed size: 4MB.
- Structure: first MegaByte of each PVs stores the same LVM metadata;

LE allocation policies:

- contiguous: LEs are adjacent and ordered;
- cling: LEs are on same physical devices;
- normal: indiscriminate allocation;
- anywhere: random allocation.

Operations on LVs:

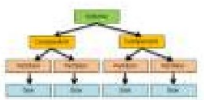
- Concatenation of PVs;
- Mirroring: map multiple PEs to a single LE;
- **Growing and shrinking LVs**;
- Growing VGs by adding PVs;
- creating **snapshots** (copy on write LE mapping);
- Hybrid volumes:
  - similar to hierarchical storage systems;
  - bcache, dm-cache, Fusion Drive, etc.

- Originally LVM is planned for **on host use**.
- **Clustered LVM**:
  - PVs are located on different hosts;
  - A shared and lock-managed metadata service is used to store shared metadata;
- **CLVM**: metadata communication flows through the lock manager;
- **HA-LVM**: uses the default file locking mechanism, and avoids metadata contention, no concurrent accesses, useful in master-slave configs.

https://en.wikipedia.org/wiki/Logical_volume_management
https://en.wikipedia.org/wiki/Logical_Volume_Manager_(Linux)
http://www.cybesphoton.com/questions/question/what-is-the-difference-between-lvm-and-raid

Logical Disk Manager

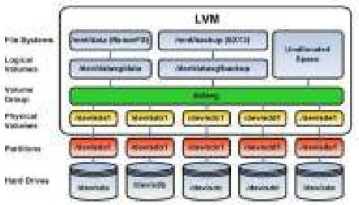
- Implements Logical Volume Management in MS Windows environment;
- By Veritas and MS;
- **Volume types:**
  - **Basic** volumes: data storage is limited to a physical disk + no partitioning;
  - **Dynamic** volumes: flexible volumes on the same physical disk, or on multiple disks:
    - Partitionable;
    - Striped volumes;
    - SPAN volumens;
  - Limited to **32 dynamic volumes**;
- Uses special partition tables + special partition bounds, i.e. 1MB bound.

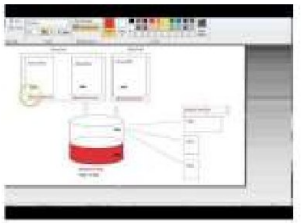


https://en.wikipedia.org/wiki/Logical_Disk_Manager

Logical Volume Management

a more flexible way of allocating block device resources than in traditional way, i.e. partitioning





- Divides physical space to chunks called Physical Extents (**PE**);
- Map PEs to Logical Extents (**LE**);
- Elementary terms:
  - **Physical Volume(s) (PV)**: a set of devices formulating the physical space;
  - **Volume Group (VG)**: a pool of resources, sum of all PEs serving the same purpose;
  - **Logical Volume(s) (LV)**: a set of dynamically allocated LEs;
- PE + LE have fixed size: **4MB**.
- Structure: first MegaByte of each PVs stores the same LVM metadata;

LE allocation policies:

- **contiguous**: LEs are adjacent and ordered;
- **cling**: LEs are on same physical devices;
- **normal**: indiscriminate allocation;
- **anywhere**: random allocation.

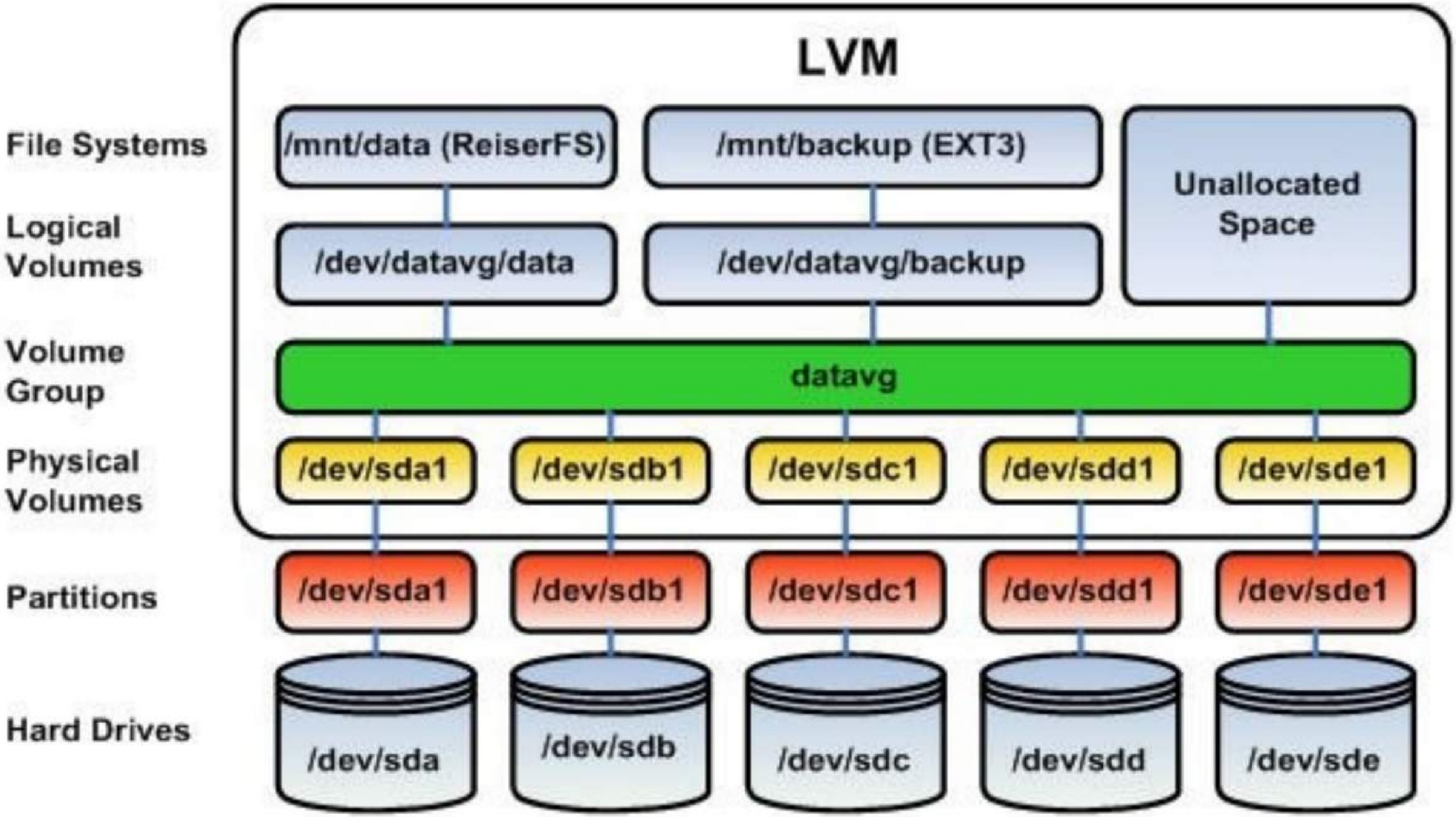
Operations on LVs:

- Concatenation of PVs;
- Mirroring: map multiple PEs to a single LE;
- **Growing and shrinking LVs**;
- Growing VGs by adding PVs;
- creating **snapshots** (copy on write LE mapping);
- **Hybrid volumes**:
  - similar to hierarchical storage systems;
  - bcache, dm-cache, Fusion Drive, etc.

- Originally LVM is planned for **on host use**;
- **Clustered LVM**:
  - PVs are located on different hosts;
  - A shared and lock-managed metadata service is used to store shared metadata;
- **CLVM**: metadata communication flows through the lock manager;
- **HA-LVM**: uses the default file locking mechanism, and avoids metadata contention, no concurrent accesses, useful in master-slave configs.

https://en.wikipedia.org/wiki/Logical_volume_management
https://en.wikipedia.org/wiki/Logical_Volume_Manager_(Linux)
http://www.cyberphoton.com/questions/question/what-is-the-

- Divides physical space to chunks called Physical Extents (**PE**);
- Map PEs to Logical Extents (**LE**);
- Elementary terms:
  - **Physical Volume(s) (PV)**: a set of devices formulating the physical space;
  - **Volume Group (VG)**: a pool of resources, sum of all PEs serving the same purpose;
  - **Logical Volume(s) (LV)**: a set of dynamically allocated LEs;
- PE + LE have fixed size: **4MB**.
- Structure: first MegaByte of each PVs stores the same LVM metadata;


LVM
File Systems
Logical Volumes
Volume Group
Physical Volumes
Partitions
Hard Drives
/mnt/data (ReiserFS)
/mnt/backup (EXT3)
Unallocated Space
/dev/datavg/data
/dev/datavg/backup
datavg
/dev/sda1
/dev/sdb1
/dev/sdc1
/dev/sdd1
/dev/sde1
/dev/sda1
/dev/sdb1
/dev/sdc1
/dev/sdd1
/dev/sde1
/dev/sda
/dev/sdb
/dev/sdc
/dev/sdd
/dev/sde

- Divides physical space to chunks called Physical Extents (**PE**);
- Map PEs to Logical Extents (**LE**);
- Elementary terms:
  - **Physical Volume(s) (PV)**: a set of devices formulating the physical space;
  - **Volume Group (VG)**: a pool of resources, sum of all PEs serving the same purpose;
  - **Logical Volume(s) (LV)**: a set of dynamically allocated LEs;
- PE + LE have fixed size: **4MB**.
- Structure: first MegaByte of each PVs stores the same LVM metadata;

LE allocation policies:

- **contiguous**: LEs are adjacent and ordered;
- **cling**: LEs are on same physical devices;
- **normal**: indiscriminate allocation;
- **anywhere**: random allocation.

- Originally LVM is planned fo

same LVM metadata;

Operations on LVs:

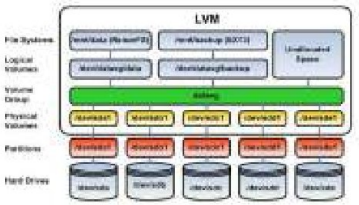
- Concatenation of PVs;
- Mirroring: map multiple PEs to a single LE;
- **Growing and shrinking LVs**;
- Growing VGs by adding PVs;
- creating **snapshots** (copy on write LE mapping);
- **Hybrid volumes:**
  - similar to hierarchical storage systems;
  - bcache, dm-cache, Fusion Drive, etc.

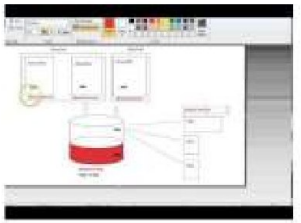
https://en.wikipedia.org/wiki/Logica
https://en.wikipedia.org/wiki/Logica

- Divides physical space to chunks called Physical Extents (**PE**);
- Map PEs to Logical Extents (**LE**);
- Elementary terms:
  - **Physical Volume(s) (PV)**: a set of devices formulating the physical space;
  - **Volume Group (VG)**: a pool of resources, sum of all PEs serving the same purpose;
  - **Logical Volume(s) (LV)**: a set of dynamically allocated LEs;
- PE + LE have fixed size: **4MB**.
- Structure: first MegaByte of each PVs stores the same LVM metadata;

Logical Volume Management

a more flexible way of allocating block device resources than in traditional way, i.e. partitioning





- Divides physical space to chunks called Physical Extents (**PE**);
- Map PEs to Logical Extents (**LE**);
- Elementary terms:
  - **Physical Volume(s) (PV)**: a set of devices formulating the physical space;
  - **Volume Group (VG)**: a pool of resources, sum of all PEs serving the same purpose;
  - **Logical Volume(s) (LV)**: a set of dynamically allocated LEs;
- PE + LE have fixed size: **4MB**.
- Structure: first MegaByte of each PVs stores the same LVM metadata;

LE allocation policies:

- **contiguous**: LEs are adjacent and ordered;
- **cling**: LEs are on same physical devices;
- **normal**: indiscriminate allocation;
- **anywhere**: random allocation.

Operations on LVs:

- Concatenation of PVs;
- Mirroring: map multiple PEs to a single LE;
- **Growing and shrinking LVs**;
- Growing VGs by adding PVs;
- creating **snapshots** (copy on write LE mapping);
- **Hybrid volumes**:
  - similar to hierarchical storage systems;
  - bcache, dm-cache, Fusion Drive, etc.

- Originally LVM is planned for **on host use**;
- **Clustered LVM**:
  - PVs are located on different hosts;
  - A shared and lock-managed metadata service is used to store shared metadata;
- **CLVM**: metadata communication flows through the lock manager;
- **HA-LVM**: uses the default file locking mechanism, and avoids metadata contention, no concurrent accesses, useful in master-slave configs.

https://en.wikipedia.org/wiki/Logical_volume_management
https://en.wikipedia.org/wiki/Logical_Volume_Manager_(Linux)
http://www.cyberphoton.com/questions/question/what-is-the-

ere: random allocation.

- Originally LVM is planned for **on host use**;
- **Clustered LVM**:
  - PVs are located on different hosts;
  - A shared and lock-managed metadata service is used to store shared metadata;
- **CLVM**: metadata communication flows through the lock manager;
- **HA-LVM**: uses the default file locking mechanism, and avoids metadata contention, no concurrent accesses, useful in master-slave configs.

ume_management

ume_Manager_(Linux)

Logical Disk Manager

- Implements Logical Volume Management in MS Windows environment;
- By Veritas and MS;
- **Volume types:**
  - **Basic** volumes: data storage is limited to a physical disk + no partitioning;
  - **Dynamic** volumes: flexible volumes on the same physical disk, or on multiple disks:
    - Partitionable;
    - Striped volumes;
    - SPAN volumens;
  - Limited to **32 dynamic volumes**;
- Uses special partition tables + special partition bounds, i.e. 1MB bound.

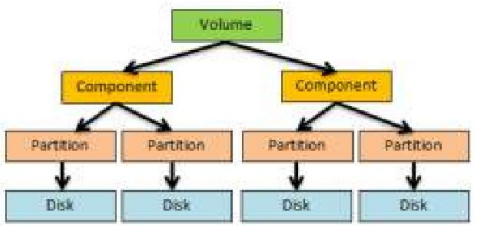


Figure 6.3

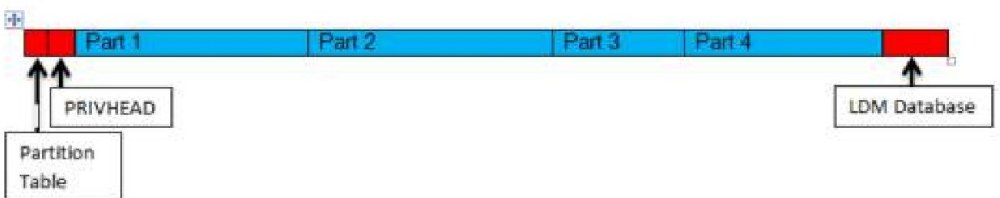


Figure 1.3

https://en.wikipedia.org/wiki/Logical_Disk_Manager

- Implements Logical Volume Management in MS Windows environment;
- By Veritas and MS;
- **Volume types:**
  - **Basic** volumes: data storage is limited to a physical disk + no partitioning;
  - **Dynamic** volumes: flexible volumes on the same physical disk, or on multiple disks:
    - Partitionable;
    - Striped volumes;
    - SPAN volumens;
  - Limited to **32 dynamic volumes**;
- Uses special partition tables + special partition bounds, i.e. 1MB bound.

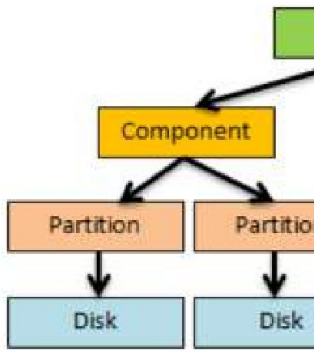


Figure 6.3

same

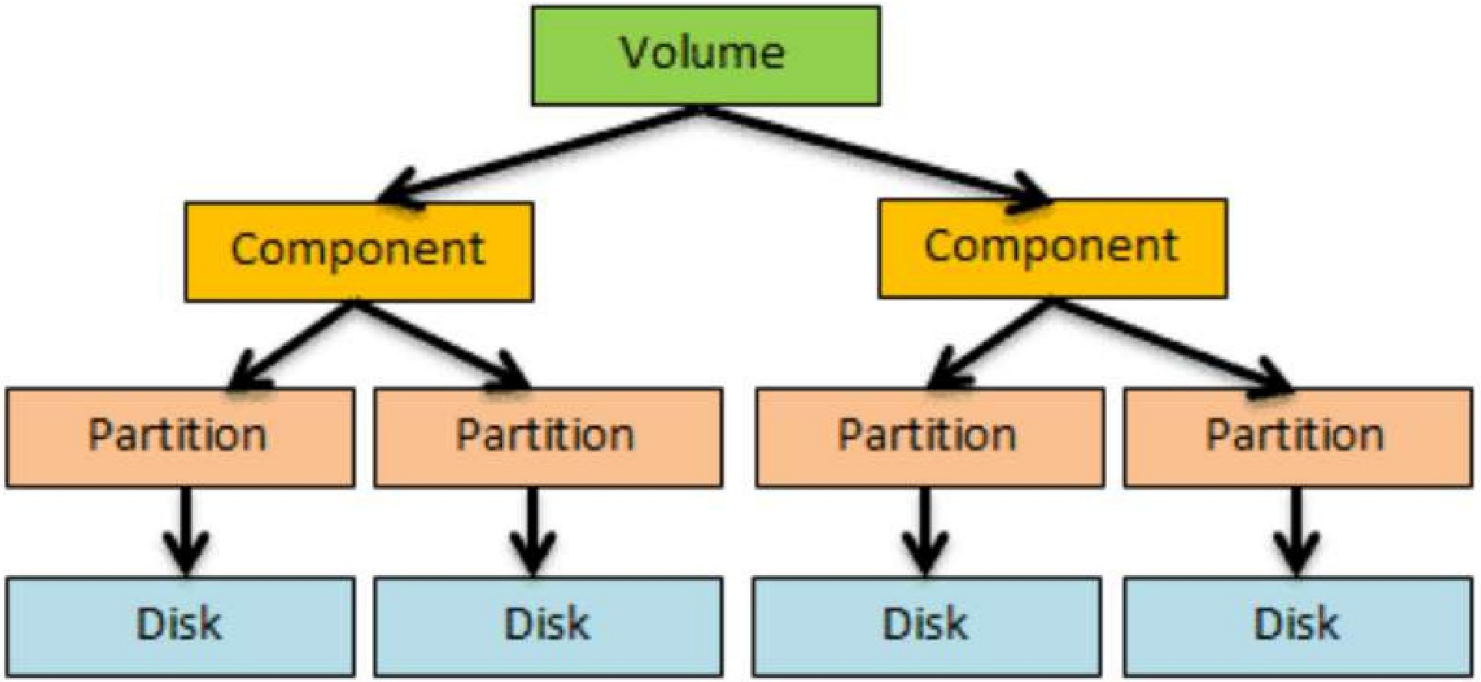


Figure 6.3

bounds,

Names:

- **caps sensitive;**
- **caps insensitive.**

ance:

stored on remote server.

Based on the underlying media:

- **disk** file systems (optical, HDD, SSD);
- **tape** file systems (**linear, mixed data and metadata**);
- **RAM** file systems;
- **database** file systems (**stored in RDBM**);
- **device** file systems (**/proc, /sys**);
- **nested** file systems (**disk images**);
- etc.

Arch

- **lo**
  in
  w

- There are plenty of them (>100), **it would worth a full semester course**.
- Stores the data organized into **files** (the analogy of paper documents), and **directories** (the analogy of paper folders).
- Since 1961.
- **Classification...**
- **Metadata...**
- **Slack space:**
  - block allocation;
  - files rarely end on block limits;
  - the average amount of space left between the file end and the last block end;
  - on average: block size / 2 / file, file number x block size / 2;
- **Fragmentation:**
  - **free space** fragmentation;
  - **file** fragmentation;
- File versioning (e.g. VMS);
- **File System journaling** (e.g. ext3);

Metadata:

- creation, modification time;
- creator;
- size: block allocation vs byte count;
- etc.

- There are plenty of them (>100), **it would worth a full semester course**.
- Stores the data organized into **files** (the analogy of paper documents), and **directories** (the analogy of paper folders).
- Since 1961.
- **Classification...**
- **Metadata...**
- **Slack space:**
  - block allocation;
  - files rarely end on block limits;
  - the average amount of space left between the file end and the last block end;
  - on average: block size / 2 / file, file number x block size / 2;
- **Fragmentation:**
  - **free space** fragmentation;
  - **file** fragmentation;
- File versioning (e.g. VMS);
- **File System journaling** (e.g. ext3);

FS journaling:

- **protecting metadata against data corruption**;
- metadata changes first go into a **circular buffer**, i.e. a **journal**;
- they are committed to media as atomic operations;
- damaged journal entries cause a **roll back** into last consistent state;

els;

- **caps insensitive**.

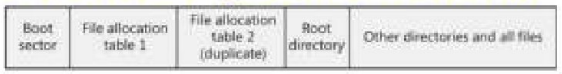
FS maintenance:

- multiple IOs -> different portion change;
- incomplete write operations -> neglected structural parts (e.g. orphan inodes, allocated blocks, etc.);
- **consistency checks, reparations**;
- **defragmentation**;

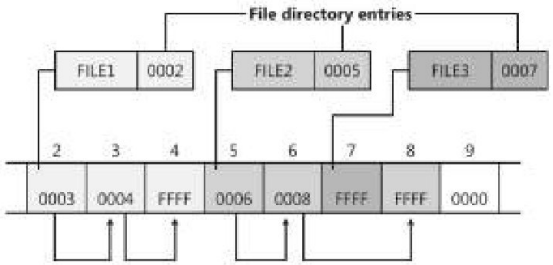
File Allocation Table (FAT) File System

- One of the oldest FS'es from the 1970s;
- **Simple**, focuses only on storing data;
- **Little endian** byte order;
- **Still wide-spread** due to its simplicity;
- Versions: FAT12, FAT16, FAT32, i.e. the width of an entry in the FAT table;
- Structures and **divides space into clusters**: i.e. contiguous space of blocks:
  - cluster size: **4 kByte** as default;
  - stores file data in **cluster chains**;
  - that are not necessarily adjacent;
- **File Allocation Tables**: maps/indices of data region (cluster index or delimiter, **0xFFFF**);
  - 0: FAT identifier;
  - 1: End-of-Chain (**EOC**) delimiter, **0xFFFFFFF**);
  - value **0**: free cluster;
  - value **EOC**: end of cluster chain;
  - value **any**: cluster index;
- **Directory entries:**
  - **32-byte** long entries;
  - root FS in a special region;
  - original **8.3** file name notation, long names trick;

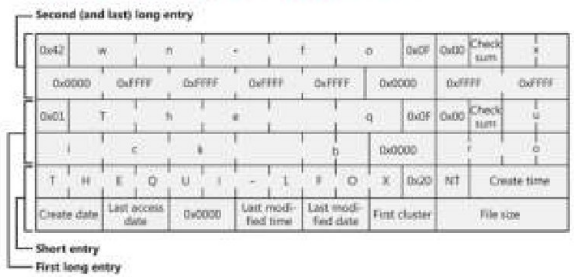
Device layout structure:



Cluster chains:



Directory entries:

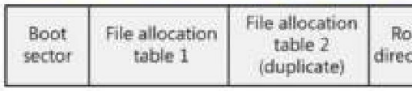


https://social.technet.microsoft.com/wiki/contents/articles/6771.the-fat-file-system.aspx
https://en.wikipedia.org/wiki/Design_of_the_FAT_file_system#FAT

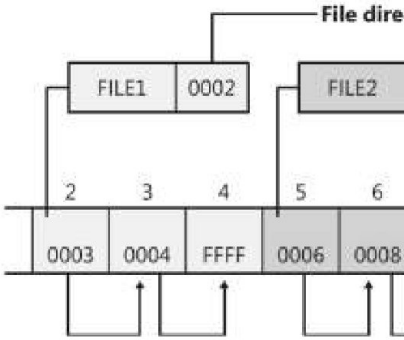
File System

- One of the oldest FS'es from the 1970s;
- **Simple**, focuses only on storing data;
- **Little endian** byte order;
- **Still wide-spread** due to its simplicity;
- Versions: FAT12, FAT16, FAT32, i.e. the width of an entry in the FAT table;
- Structures and **divides space into clusters**: i.e. contiguous space of blocks:
  - cluster size: **4 kByte** as default;
  - stores file data in **cluster chains**;
  - that are not necessarily adjacent;
- **File Allocation Tables**: maps/indices of data region (cluster index or delimiter, **0xFFFF**);
  - 0: FAT identifier;
  - 1: End-of-Chain (**EOC**) delimiter, **0xFFFFFFF**);
  - value **0**: free cluster;
  - value **EOC**: end of cluster chain;
  - value **any**: cluster index;
- **Directory entries:**
  - **32-byte** long entries;
  - root FS in a special region;
  - original **8.3** file name notation, long names trick;

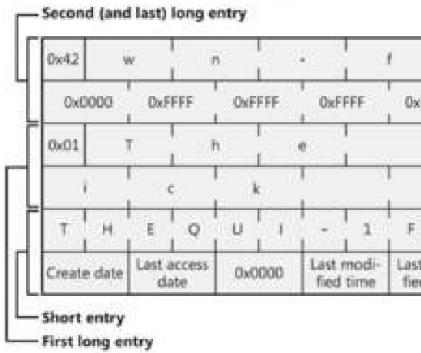
Device layout



Cluster chain



Directory entr



https://social.technet.microsoft.com/wiki/contents/articles/6771.the-fat-file-s
https://en.wikipedia.org/wiki/Design_of_the_FAT_file_system#FAT

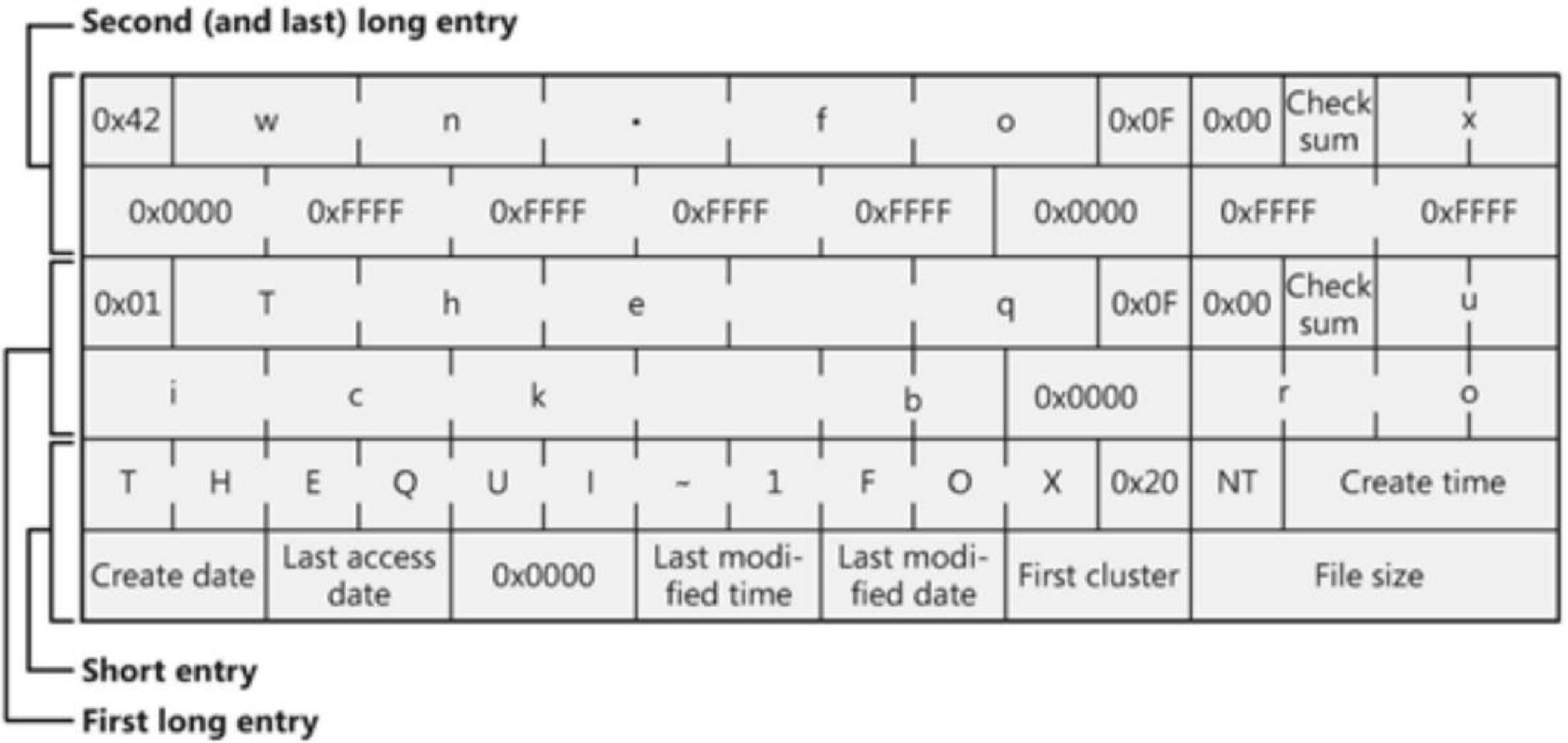
Device layout structure:

Boot sector	File allocation table 1	File allocation table 2 (duplicate)	Root directory	Other directories and all files

Cluster chains:

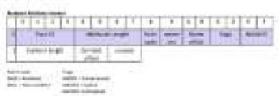
File directory entries

Directory entries:



New Technologies File System (NTFS)

- Since 1993, as a replacement of FAT; more flexible.
- New features:
  - **Variable cluster size**, default to 4 kBytes, up to 2 MBytes;
  - **Journaling** ($LogFile);
  - Hard links;
  - **Streams**: a logical sequence of file records;
  - **File compression and sparse files**;
  - Volume Shadow Copy based on Copy on Write;
  - ACLs: Directory ACL, Security ACL;
  - Encryption;
  - Quotas;
  - **Resizing**;
- NTFS structure...



- Metadata: **Master File Table (MFT) + MFT mirror**;
- Stores ACLs, creation times, size info, data blocks, etc.
- 1 kB entries;
- NTFS metafiles (0-26, 27- regular files);
- **MFT structure:**
  - header (block size, cluster size, type, etc.);
  - attribute headers (standard, short name, long name, etc.);
  - attributes (length, type, value, etc);
  - resident vs. non-resident files/attributes;
- Data stored in **either MFT** (up to 900 bytes) or **clusters**.
- **Runs**: cluster intervals.
- Uses **B-trees** instead of tables and lists.

http://www.ntfs.com/ntfs_basics.htm
https://en.wikipedia.org/wiki/NTFS
https://www.yumpu.com/en/document/read/11722944/ntfs-cheat-sheet-writeblocked

$Upcase	Upcase table	10	Converts lowercase characters to matching Unicode uppercase characters.
$Extend	NTFS extension file	11	Used for various optional extensions such as quotas, reparse point data, and object identifiers.
		12-15	Reserved for future use.

- Metadata: **Master File Table (MFT) + MFT mirror**;
- Stores ACLs, creation times, size info, data blocks, etc.
- 1 kB entries;
- NTFS metafiles (0-26, 27- regular files);
- **MFT structure:**
  - header (block size, cluster size, type, etc.);
  - attribute headers (standard, short name, long name, etc.);
  - attributes (length, type, value, etc);
  - resident vs. non-resident files/attributes;
- Data stored in **either MFT** (up to 900 bytes) or **clusters**.
- **Runs**: cluster intervals.
- Uses **B-trees** instead of tables and lists.

cs.htm
TFS
cument/read/11722944/ntfs-cheat-sheet-writeblocked

NTFS Boot sector

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Jump Instruction			OEM ID								Bytes/ Sector		Sect/ clust	res	
10	0x000000			unused		Media desc	0x0000		Sect / track		Number heads			Hidden Sectors		
20	unused								Total Sectors							
30	Logical Cluster of $MFT								Logical Cluster of $MFTMirr							
40	Clust / File record segment			Clusters / Index Block					Volume Serial Number							
50	Checksum			Boot Code												
60	Boot Code															
.	Boot Code															
.	Boot Code															
1E0	Boot Code															
1F0	Boot Code														55	AA

NTFS files

$Upcase	Upcase table	10	Converts lowercase characters to matching Unicode uppercase characters.
$Extend	NTFS extension file	11	Used for various optional extensions such as quotas, reparse point data, and object identifiers.
		12-15	Reserved for future use.

- Metadata: **Master File Table (MFT) + MFT mirror**;
- Stores ACLs, creation times, size info, data blocks, etc.
- 1 kB entries;
- NTFS metafiles (0-26, 27- regular files);
- **MFT structure:**
  - header (block size, cluster size, type, etc.);
  - attribute headers (standard, short name, long name, etc.);
  - attributes (length, type, value, etc);
  - resident vs. non-resident files/attributes;
- Data stored in **either MFT** (up to 900 bytes) or **clusters**.
- **Runs**: cluster intervals.
- Uses **B-trees** instead of tables and lists.

cs.htm
TFS
cument/read/11722944/ntfs-cheat-sheet-writeblocked

NTFS files

File	Name	$MFT record #	Description
$Mft	Master File Table	0	Contains one base file record for each file and folder on an NTFS volume. If the allocation information for a file or folder is too large to fit within a single record, other file records are allocated as well.
$MftMirr	MFT mirror	1	Guarantees access to the MFT in case of a single-sector failure. It is a duplicate image of the first four records of the MFT.
$LogFile	Log file	2	Contains information used by NTFS for faster recoverability. The log file is used by Windows Server 2003 to restore metadata consistency to NTFS after a system failure. The size of the log file depends on the size of the volume, but you can increase the size of the log file by using the Chkdsk command.
$Volume	Volume	3	Contains information about the volume, such as the volume label and the volume version.
$AttrDef	Attribute definitions	4	Lists attribute names, numbers, and descriptions.
.	Root file name index	5	The root folder.
$Bitmap	Cluster bitmap	6	Represents the volume by showing free and unused clusters.
$Boot	Boot sector	7	Includes the BPB used to mount the volume and additional bootstrap loader code used if the volume is bootable.
$BadClus	Bad cluster file	8	Contains bad clusters for a volume.
$Secure	Security File	9	Contains unique security descriptors for all files within a volume.
$Upcase	Upcase table	10	Converts lowercase characters to matching Unicode uppercase characters.
$Extend	NTFS extension file	11	Used for various optional extensions such as quotas, reparse point data, and object identifiers.
		12-15	Reserved for future use.

File Record Segment Header

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	F	I	L	E	Update Seq array offset		Update Seq array size		$LogFile Sequence Number							
1	Seq no		Hard Link Count		1st attrib offset		Flags		Used size of file record				Allocated size of file record			
2	File reference to base file record								Next attrib ID				MFT Record No			
3	default location of update seq array (size determined by seq size)						Reserved for update sequence array?									
	Reserved for sequence array?								Common location of 1st attrib							

Resident Attribute Header

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Type ID				Attribute Length				Form code	name len	Name offset		flags		Attrib ID	
1	Content length				Content offset		unused									

Form code
0x00 = Resident
0x01 = Non resident

Flags
0x00FF = Compressed
0x8000 = Sparse
0x4000 = Encrypted

$File_Name

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Parent Directory								Date Created							
10	Date Modified								Date MFT Modified							
20	Date Accessed								Logical file size							
30	Size on disk								Flags*				Reparse value			
40	Name len	Name type	Name (variable length)													

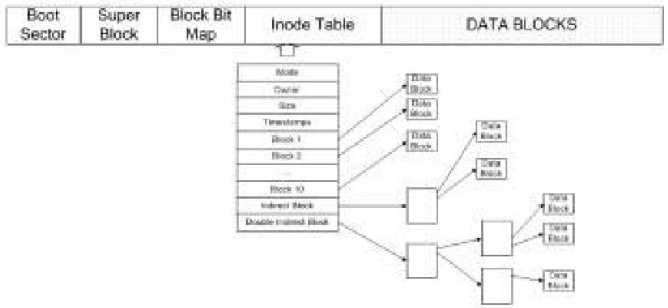
0x4000 = Encrypted

$Data (Standard Header with data run, may be resident or non resident, non resident shown here)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Type ID 0x80				Attribute Length				Form code	name len	Name offset		flags		Atrib ID	
10	Start virtual cluster number								Ending virtual cluster number							
20	Runlist offset		Compression unit size		0x0000				Size of attribute content							
30	size on disk of attribute content								Initialized size of attribute content							
40	Data runlists															

Unix File Systems (UFS)

- Multiple variations: UFS, EXT2, EXT3, ReiserFS, etc.
- Represents a different principle:
  - No letter tagged devices;
  - **single tree-structure**;
  - each partition, device is mounted under the / tree;
- Metadata stored in **i-nodes**.
- Directory entries are text files containing file names and i-node assignments.
- Multiple references to same i-nodes are hard links.
- **"Everything is a file:"**
  - directories and files;
  - device files, sockets, pipes, links;



UFS structure:

- boot block: serves for OS boot;
- **superblock**: describe FS geometry, version, parameters;
- **cylinder groups**: superblock backup, cylinder group header, number of i-nodes, and data blocks;
- **i-node bitmap**: allocation of i-nodes;
- **data bitmap**: allocation of data nodes;
- **i-node table**;
- **data blocks**;

Linux Ext4 file system features:

- journaling;
- data stored in extents, i.e. contiguous blocks, like cluster runs in NTFS;
- dynamic resizing;
- B-tree like directory structure;
- compression, encryption;

https://www.nongnu.org/ext2-doc/ext2.html

Unix File System

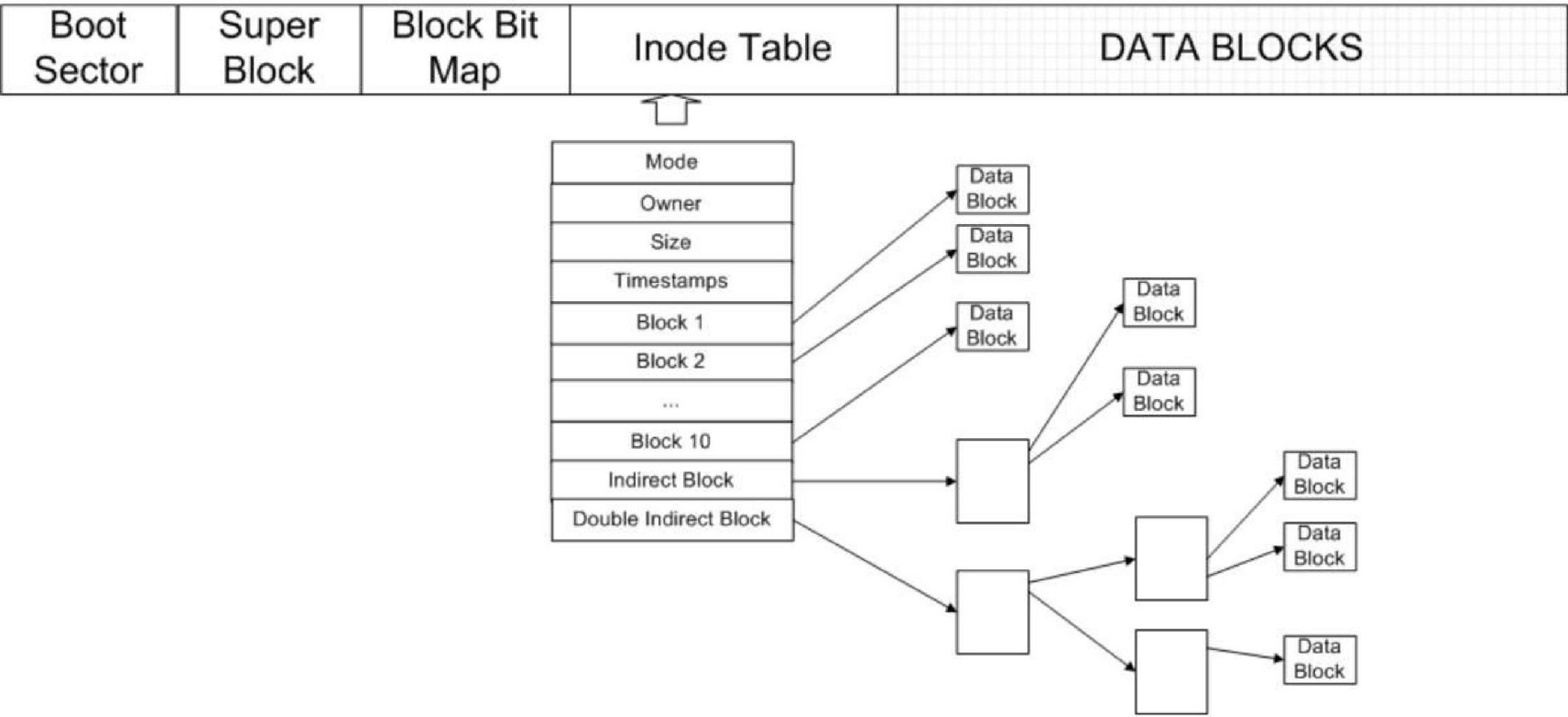
- Multiple variations: UFS, EXT2, EXT3, ReiserFS, etc.
- Represents a different principle:
  - No letter tagged devices;
  - **single tree-structure**;
  - each partition, device is mounted under the / tree;
- Metadata stored in **i-nodes**.
- Directory entries are text files containing file names and i-node assignments.
- Multiple references to same i-nodes are hard links.
- **"Everything is a file:"**
  - directories and files;
  - device files, sockets, pipes, links;

UFS structure:

UFS structure:

- boot block: serves for OS boot;
- **superblock**: describe FS geometry, version, parameters;
- **cylinder groups**: superblock backup, cylinder group header, number of i-nodes, and data blocks;
- **i-node bitmap**: allocation of i-nodes;
- **data bitmap**: allocation of data nodes;
- **i-node table**;
- **data blocks**;

https://www.nongnu.org


Boot Sector
Super Block
Block Bit Map
Inode Table
DATA BLOCKS
Mode
Owner
Size
Timestamps
Block 1
Block 2
...
Block 10
Indirect Block
Double Indirect Block
Data Block
Data Block
Data Block
Data Block
Data Block
Data Block
Data Block
Data Block

Linux Ext4 file system features:

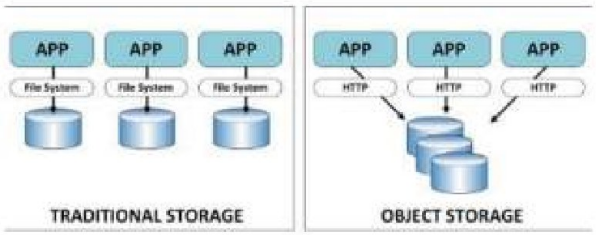
- journaling;
- data stored in extents, i.e. contiguous blocks, like cluster runs in NTFS;
- dynamic resizing;
- B-tree like directory structure;
- compression, encryption;

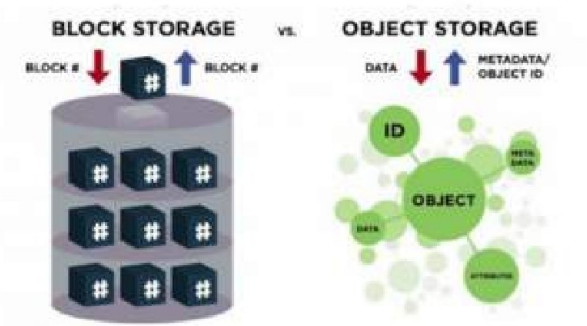
Distributed, Network and Global File Systems

- Huge areas on their own;
- Eliminate single points of failure;
- Allow **multiple clients to access the same file data structure**;
- Metadata protection, lock and sharing;
- **Virtual and shared name spaces** across local file systems;
- **Variants:**
  - **shared disk** file systems: sharing a common area on disks (NFS, Lustre);
  - **distributed** file systems: use network protocols to share objects (GFS, GPFS);

Object Storage

- block storage: data on physical device;
- file storage: data in human readable structures;
- **object storage:** some sort of **associative storage** over **HTTP**;
- Design principles:
  - simplicity;
  - robustness;
  - partial associativity;





Terms:

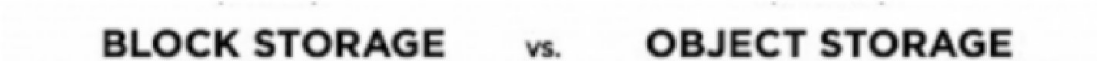
- **Bucket:** A data storage pool, e.g. a device;
- **Objects:** Data units to be stored, like files, but have properties:
  - GID;
  - **partially associative GID matching;**
  - any parameters that help indexing;
- optimized for large amounts of data: large number of large files;

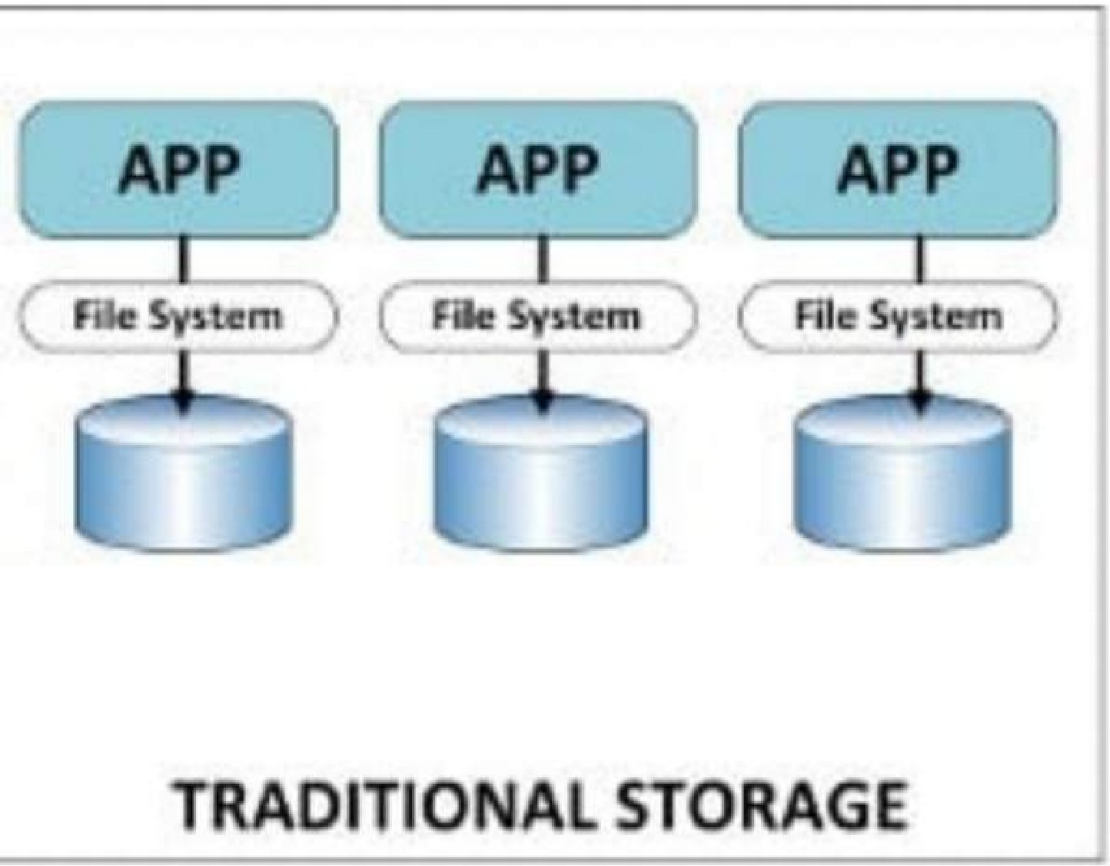
Object Storage

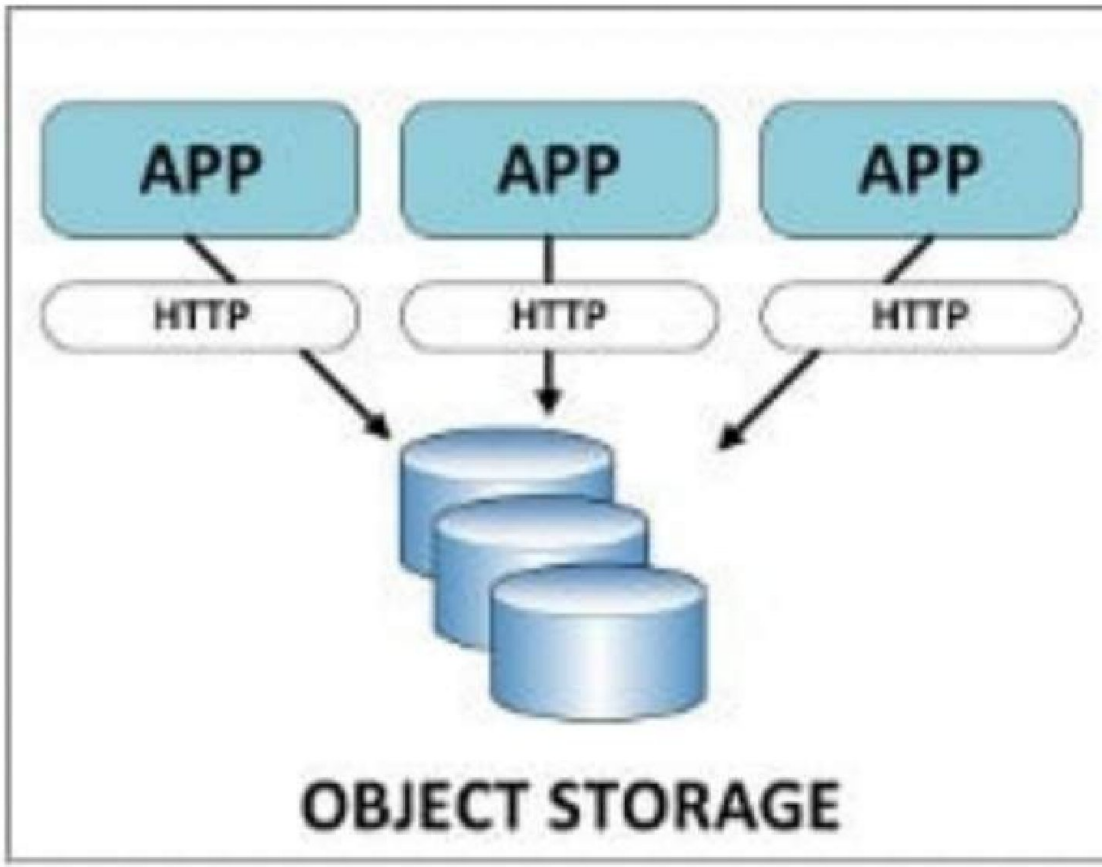
- block storage: data on physical device;
- file storage: data in human readable structures;
- **object storage:** some sort of **associative storage** over **HTTP**;
- Design principles:
  - simplicity;
  - robustness;
  - partial associativity;

Terms:

- **Bucket:** A data
  device;
- **Objects:** Data
  files, but have

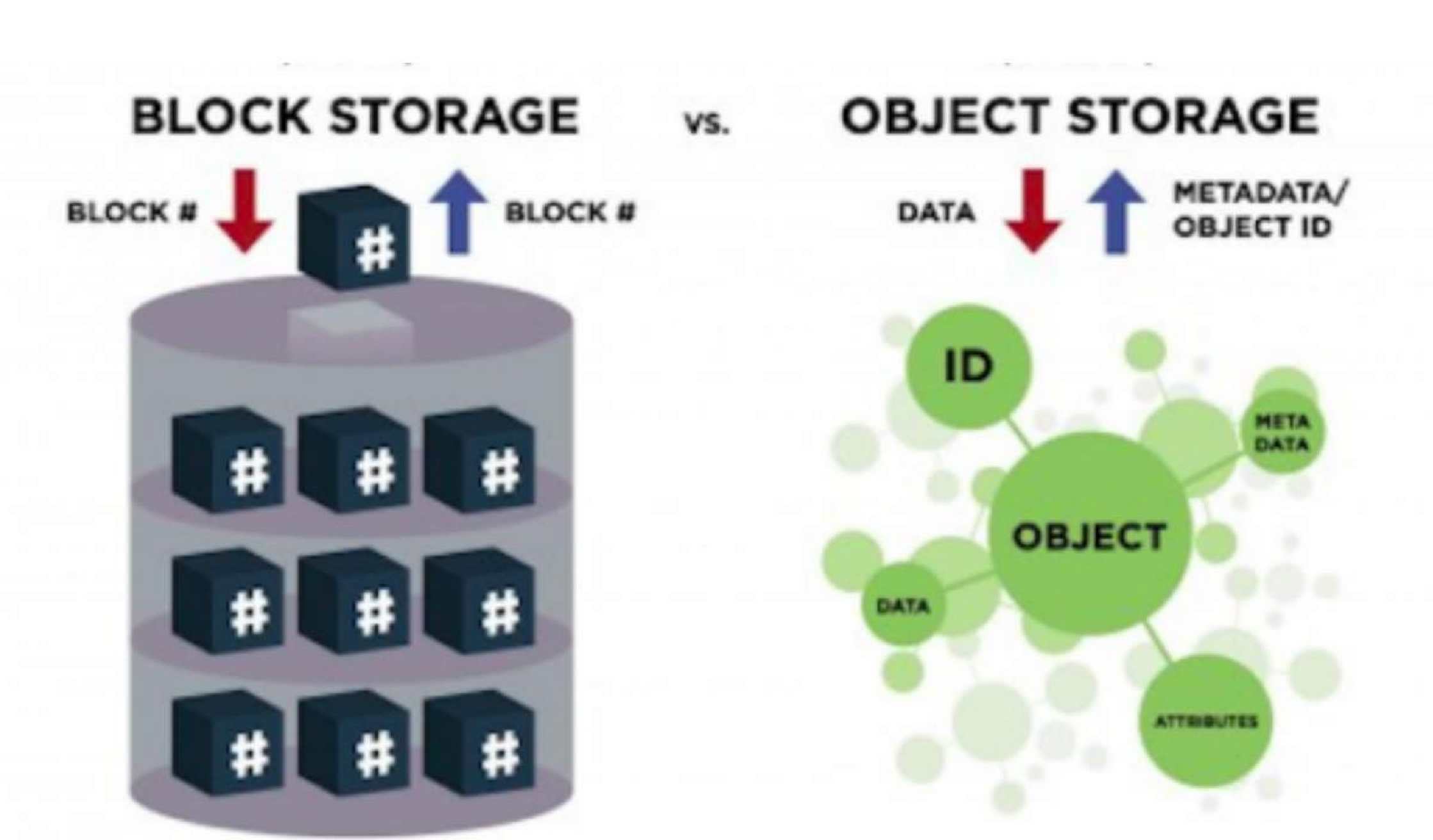



APP
APP
APP
File System
File System
File System
TRADITIONAL STORAGE


APP
APP
APP
HTTP
HTTP
HTTP
OBJECT STORAGE

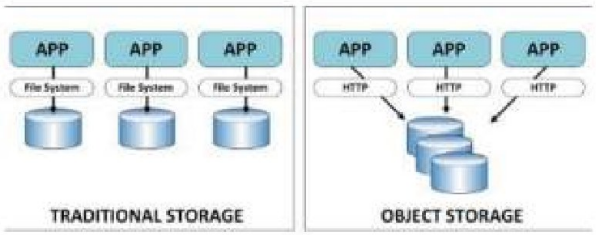
Terms:

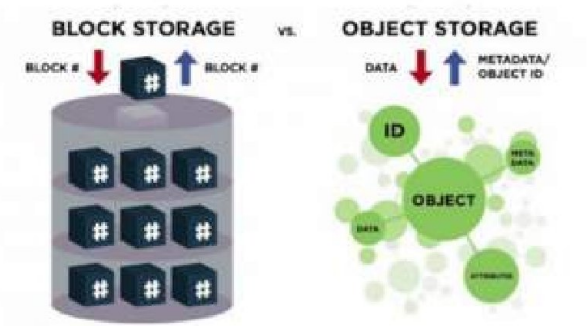
- **Bucket:** A data storage pool, e.g. a device;
- **Objects:** Data units to be stored, like files, but have properties:
  - GID;
  - **partially associative GID matching;**
  - any parameters that help indexing;
- optimized for large amounts of data: large number of large files;


BLOCK STORAGE
vs.
OBJECT STORAGE
BLOCK #
BLOCK #
DATA
METADATA/
OBJECT ID
ID
META
DATA
OBJECT
DATA
ATTRIBUTES

Object Storage

- block storage: data on physical device;
- file storage: data in human readable structures;
- **object storage:** some sort of **associative storage** over **HTTP**;
- Design principles:
  - simplicity;
  - robustness;
  - partial associativity;





Terms:

- **Bucket:** A data storage pool, e.g. a device;
- **Objects:** Data units to be stored, like files, but have properties:
  - GID;
  - **partially associative GID matching;**
  - any parameters that help indexing;
- optimized for large amounts of data: large number of large files;