Ievada/Izvada sistēmas

I/O un kam tas vajadzīgs

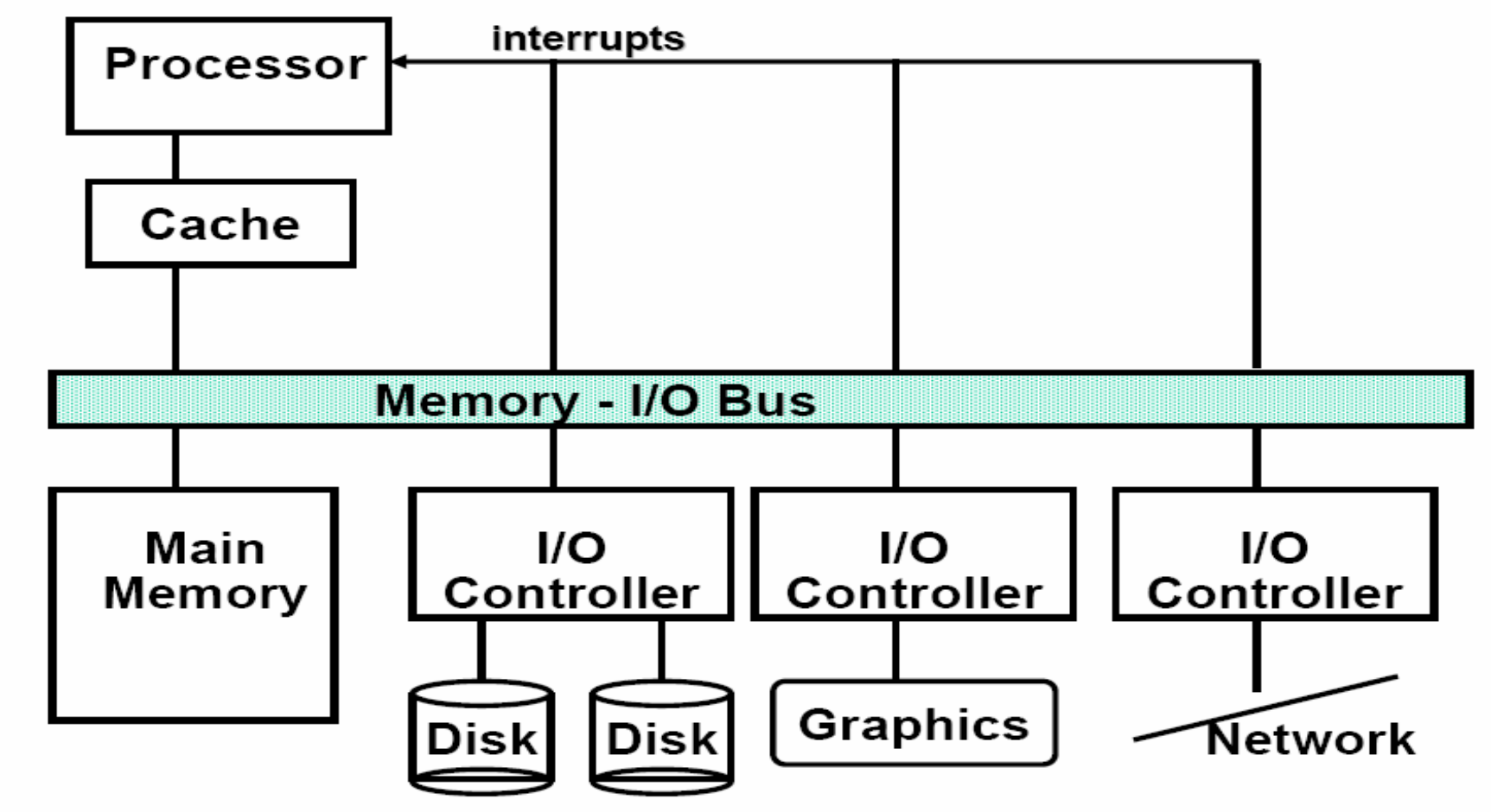
I/O iedalījums

- Ievada iekārtas (tikai lasīšanai)
- Izvada iekārtas (tikai rakstīšanai)
- Uzglabāšanas iekārtas (vienlaicīgi)
- Komunikāciju iekārtas (vienlaicīgi)

Piemēri:

Iekārta	Tips	Partneris	Josla KB/s
Pele	Ievada	Cilvēks	0.01
Monitors	Izvada	Cilvēks	60 000
Modems	Komunikāciju	Mašīna	5
LAN	Komunikāciju	Mašīna	600
Lentas	Uzglabāšanas	Mašīna	2 000
Diski	Uzglabāšanas	Mašīna	10 000

I/O sistēmas arhitektūra



I/O iekārtu tipi

- Simbolu iekārtas:
  - Apmainās ar simboliem
  - Nav adresējamas
  - Piem. klaviatūra, printeri
- Bloku iekārtas:
  - Apmainās ar datu blokiem
  - Adresē bloku veidā
  - Piem. Diski, tīkla iekārtas
- DMA (Ditect Memory Access):
  - Iespēja iekārtai strādāt ar atmiņu bez CPU piesaistes
  - Parasti pielieto bloku iekārtām
- Atmiņas adresēs iekļautās:
  - Iekārtu reģistri ir daļa no atmiņas adresēm un nevajag speciālas komandas (piem. sparc)
- I/O porti:
  - Katram kontrolierim tiek piešķirta speciāla adrese (ports) kas ***nav atmiņas adrese***
- IRQ pārtraukumu līnijas:
  - Fiziski signāli IRQ kontrolierim kas signalizē par iekārtas gatavību veikt kādu darbību.

I/O problēmas

- Vai tādas vispār ir?
  - Sistēmas ātrdarbības uzlabojumus ierobežo lēnākā iekārta (Amdala likums)
    - 10% IO & 10x CPU => 5x kopējā veiktspēja (- 50%)
    - 10% IO & 100x CPU => 10x kopējā veiktspēja (- 90%)
  - Visi grib ātru piekļuvi I/O, VM pieprasījumlapošana, datoru tīkls, failu piekļuve....
  - Kopumā veidojas I/O sastrēgumi “bottleneck” un arvien lielāka laika daļa jāpavada gaidot uz I/O apstrādi.
  - I/O var neņemt vērā ***ja CPU ir ko darīt*** kamēr kāds process gaida uz I/O
  - ***PC parasti nav ko darīt*** kamēr jāgaida uz I/O (serveriem tas gan var būt savādāk, bet lai izdarītu konteksta pārslēgšanu vajag vairāk RAM un tas arī prasa laiku)

I/O problēmas

- Pamatā problēma ir I/O un CPU ātrdarbības plaisa:
  - Ja CPU gaida uz I/O tad tas nevar veikt neko citu.
  - Ja CPU dara ko citu kamēr veic I/O tad kā sinhronizēt CPU un I/O?
- Varētu lieto pārtraukumus kuri paziņo CPU par I/O notikumiem uz kuriem CPU reaģēs nosūtot nākamo datu porciju un atkal pārslēdzoties uz citiem darbiem līdz nākamajam pārtraukumam...
- Bet var būt pietiekami ātras iekārtas (salīdzināms ar CPU ātrdarbību) kas tādā gadījumā pārslogos CPU ar saviem pārtraukumiem (CPU tikai pārslēgsies starp uzdevumiem un nevarēs veikt neko derīgu)
- Masveida datu apmaiņai tāpēc cenšas lieto DMA. Šādā gadījumā CPU pasaka iekārtai kur iegūt/nolikt datus un iekārta tikai darba beigās paziņo par darba veikšanu.

I/O programmēšana

- Senatnē I/O varēja programmēt tieši bet..
- Šodien daudzuzdevumu un daudzlietotāju vidēs tas nav pieļaujams dēļ nepieciešamības nodrošināt efektīvu iekārtu izmantošanu, visu procesu vienlīdzību piekļuvē, kļūdu apstrādi.... un tāpēc šodien I/O veic OS

Kopnes

- Kopne ir:
  - Koplietošanas komunikācijas resurss
  - Vadu kopa kas savieno vairākas apakšsitēmas
  - Pamata rīks ar kura palīdzību var veidojot lielas un sarežģītas sistēmas
- Pozitīvi:
  - Daudzpusīgs pielietojums (var pieslēgt dažādas iekārtas, pārvietot iekārtas starp datorsistēmām...)
  - Neliela cena (visus vadus koplieto visas iekārtas, vieni un tie paši vadi dara vairākas lietas)
- Negatīvi:
  - Tās ir sistēmas šaurās vietas
  - Max. ātrumu nosaka fiziskie izmēri un pieslēgto iekārtu skaits.
  - Prasa saskaņot ļoti dažādu (pēc latentuma un caurlaides spējas) iekārtu darbību.

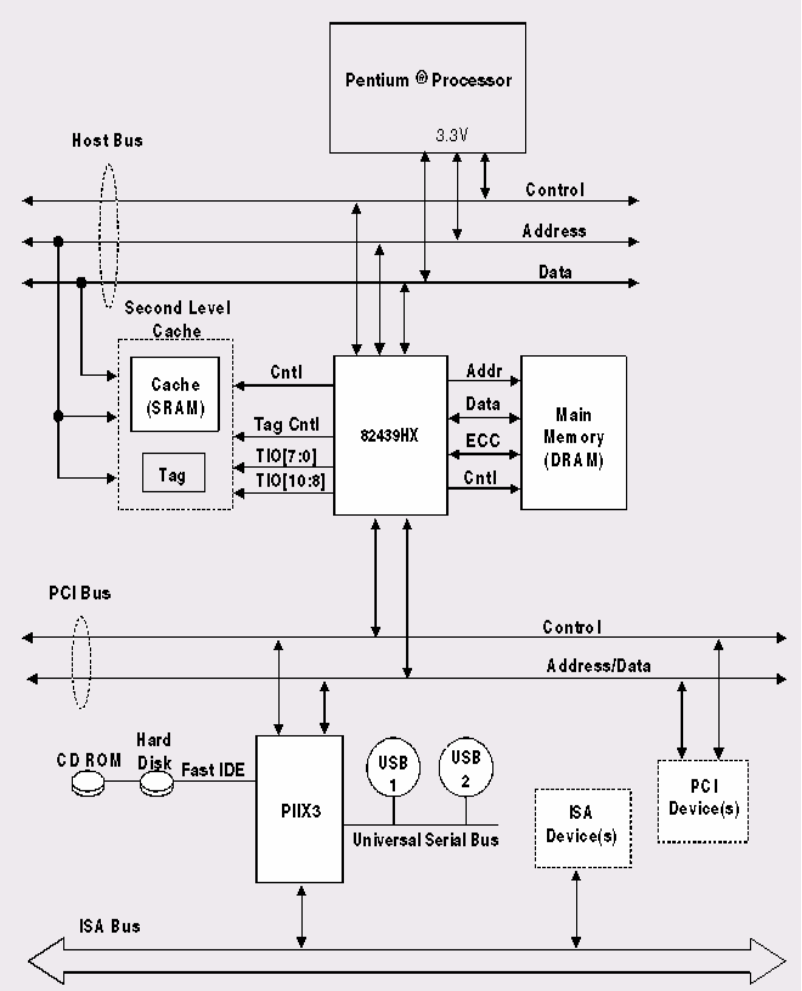
Kopnes

- Satur minimumā:
  - Datu daļu
    - Novada informāciju starp izejas un mērķa mezgliem
    - Satur datus ***vai*** adreses ***vai*** sarežģītas komandas
  - Vadības daļu
    - Signalizācija par pieprasījumiem un atbildēm uz tiem
    - Norāda ***kas*** ir datu daļā

Kopnes

- Kopnes cikls sastāv no divām daļām:
  - Komandas (un adreses) izstādīšanu pieprasījuma daļā
  - Datu apmaiņas darbības daļā
- Vadītājs ir tas kurš uzsāk kopnes ciklu:
  - Izstādot komandu (un adresi)
- Vadāmais ir tas kurš atbild uz šo adresi ar:
  - Datu nosūtīšanu ja tā tika pieprasīta no vadītāja puses
  - Datu saņemšanu ja tā tika pieprasīta no vadītāja puses

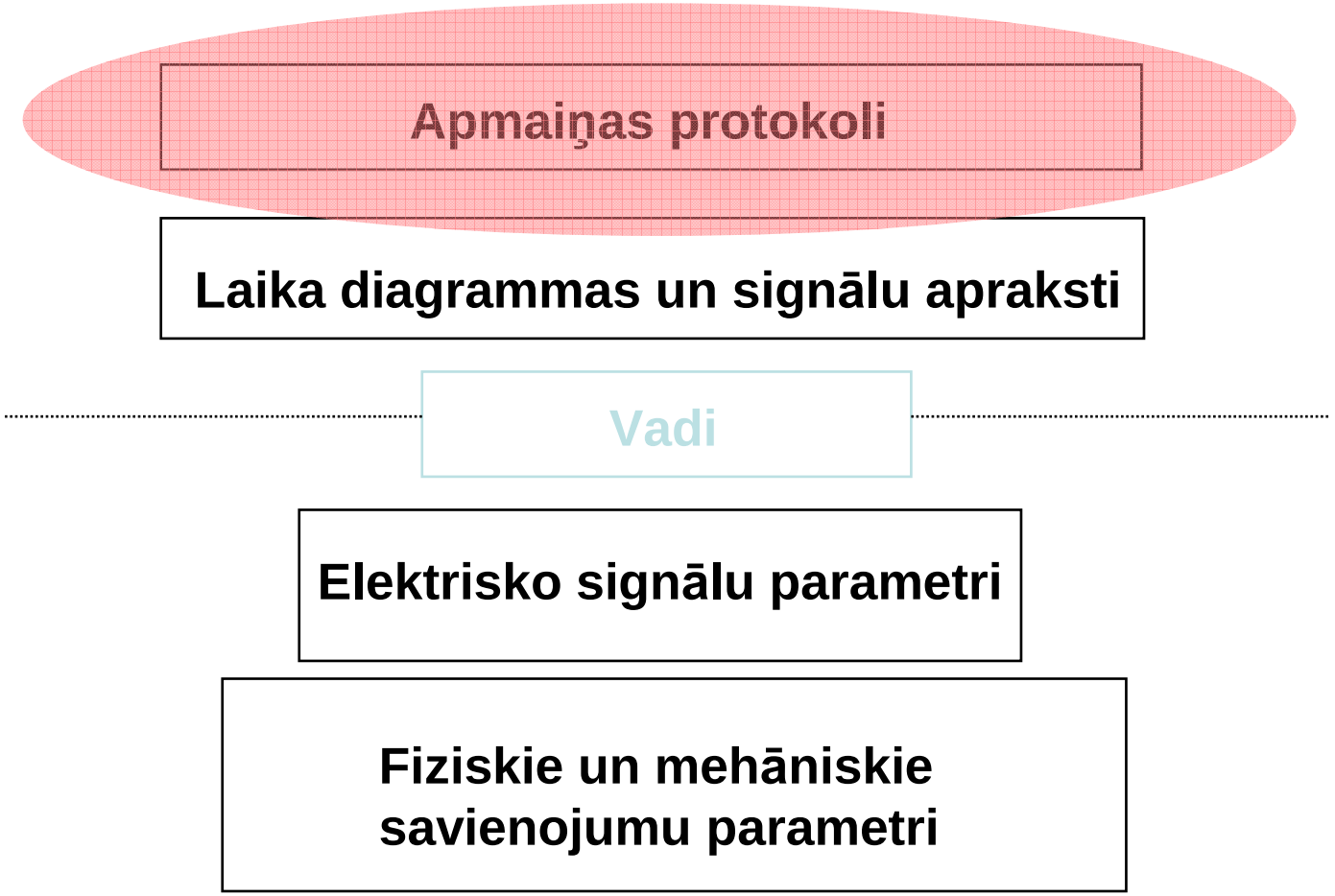
Kopnes



Atmiņas kopne
- Īsa un ātra
- Specifiskas dotajai realizācijai
- Atbilst kešu bloku apmaiņas veidiem un izmēriem

I/O kopne
- Garākas un lēnākas
- Industrijas standarti
- Jāspēj savienot ļoti dažādas iekārtas
- Ar tiltu palīdzību slēdzas pie CPU<>RAM kopnes

Kopņu loģiskās sastāvdaļas



Kopņu pamati

- Taktēšana (vai ir vienota takts)?
  - Sinhronas: taktētas, īsas vai zemas frekvences
  - Asinhronas: netaktētas, lieto rokspiešanu “handshaking”
  - Isosinhronas: liels joslas platums, pakešu apstrādes ideoloģija
- Komutācija (kad tiek iegūta/atgriezta kopnes vadība)?
  - Atomiska “Atomic”: kopne tiek turēta kamēr netiek izpildīta visa darbība – lēni
  - Sadalīta “Split-transaction”: kopne tiek atbrīvota starp pieprasījumu –un atbildi
- Arbitrāža (kā izlemt kurš iegūs kopni nākamais)?
  - Pārklātā: izdara arbitrāžu kamēr notiek datu apmaiņa
  - Ziedlapķēdes: tuvākā iekārta iegūst augstāku prioritāti
  - Sadalīta: VAI shēmas, zemākās prioritātes iekārta atkāpjas
- Citi parametri
- – Vienotas/dalītas datu un adrešu līnijas, platums, šaltspārraides iespēja

Uzglabāšanas sistēmas

- Galvenais parametrs ir.....?
- Pamatā attīstība notiek pateicoties pārejai no pakešapstrādes uz multiprogrammu sistēmām (kopš 50iem) un ieviešot visuresošo skaitļošanu (kopš 90iem) ....
- Rezultāti ir mazākas, lētākas, uzticamākas(?), mazāk energoprasīgas, ietilpīgākas, hierarhiski vadāmas uzglabāšanas iekārtas.

Diski

Diski

- Termini:
  - Pievads – kustina sviras (piedzen ar magnētiskā lauka palīdzību)
  - Plates - parasti vairākas, ar magnētisku informācijas ierakstu/nolasīšanu no abām plates pusēm
  - Celiņi - informāciju parasti ieraksta cilindriskos celiņos kas savukārt tiek iedalīti sektoros (512B - var būt maināms lielums bet kopskatā tas ir 2^n)
  - Svira - kustina galviņas virs platēm izvēloties celiņus (meklēšana “seek” = virsmas izvēle, cilindra izvēle, gaidīšana līdz sektors nonāk zem galvas)
  - Galvas – ieraksta/nolasīšanas elementi
  - Vārpsta – tas uz kā nostiprinātas plates un piedzen tās (parasti 7 200 – 15 000 apgr.^-1)

Diski piekļuves laiks

- Vidējais meklēšanas laiks ”averege seek time” atkarīgs no to cik jāpārvieto svira (standarta etalonuzdevums pieņemot ka piekļuves ir pilnīgi gadījuma rakstura parasti ~8ms)
  - Meklēšana ir vislielākais ļaunums (iekustināt, kustība, bremzēšana, pozicionēšana) tāpēc veido/uztur FS kas lokalitāti laikā pārvērš ***fiziskā lokalitātē***.
- Rotācijas laiks (1/2 apgr. pie 7200 apgr.^-1 ir 4.7 ms)
- Pārraides laiks - atkarīgs no joslas platuma (ieraksta blīvuma, rotācijas ātruma, pieprasījuma lieluma) - ārējie celiņi 1,7x ātrāki nekā iekšējie.
- Kontroliera liektēriņu laiks (lasīšana uz priekšu, keši (to algoritmi))
- Disk Latency = Seek Time + Rotation Time + Transfer Time + Controller Overhead

HDD attīstība

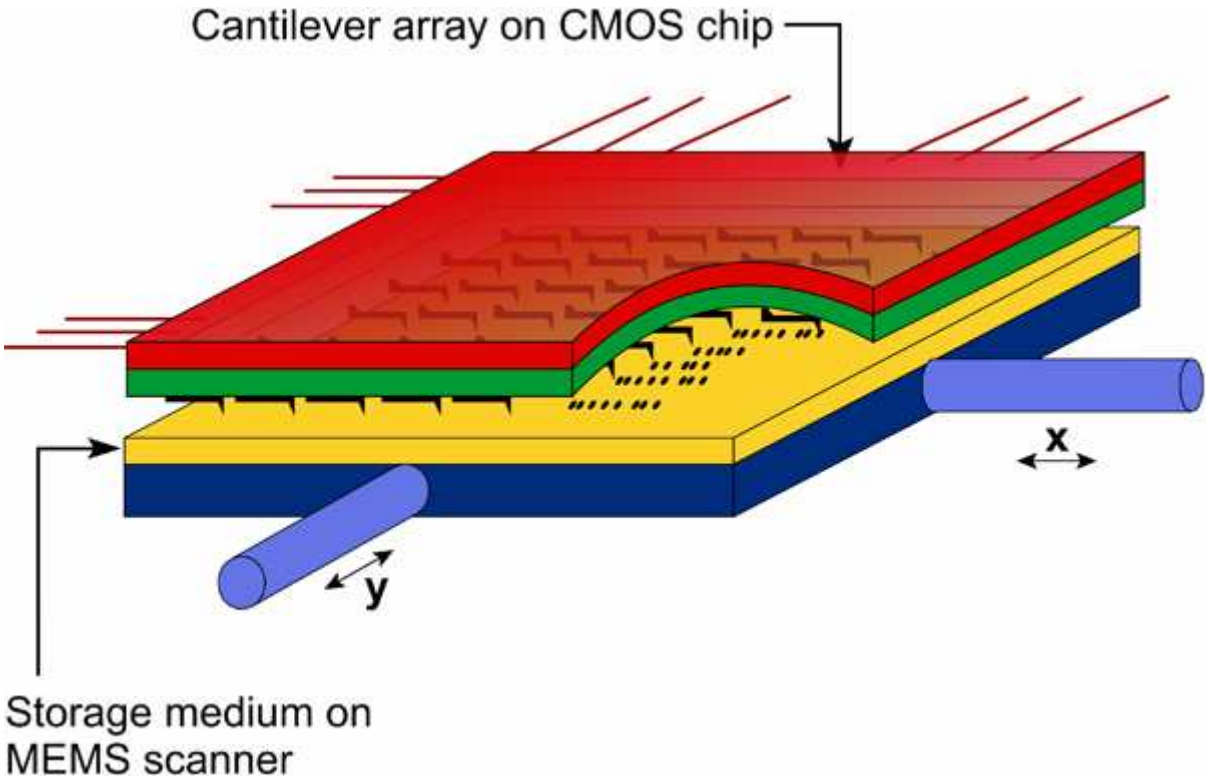
- Ietilpība
  + 100%/gadā (2X / 1. gadā)
- Pārraides ātrums
  + 40%/gadā (2X / 2. gados)
- Rotācijas un meklēšanas laiku summa
  - ***8%/ gadā (1/2 10 gados)***
- Izmaksas par MB (MB/$)
  > 100%/gadā (2X / 1. gadā)
- Mazāk elektronikas un lielāks ieraksta blīvums (Bits Per Square Inch = BPT x TPI)
- Joprojām RAM piekļuve ir 100 000x ātrāka lai gan joslas platuma atšķirība ir tikai ~50x

Alternatīvas (Flash)

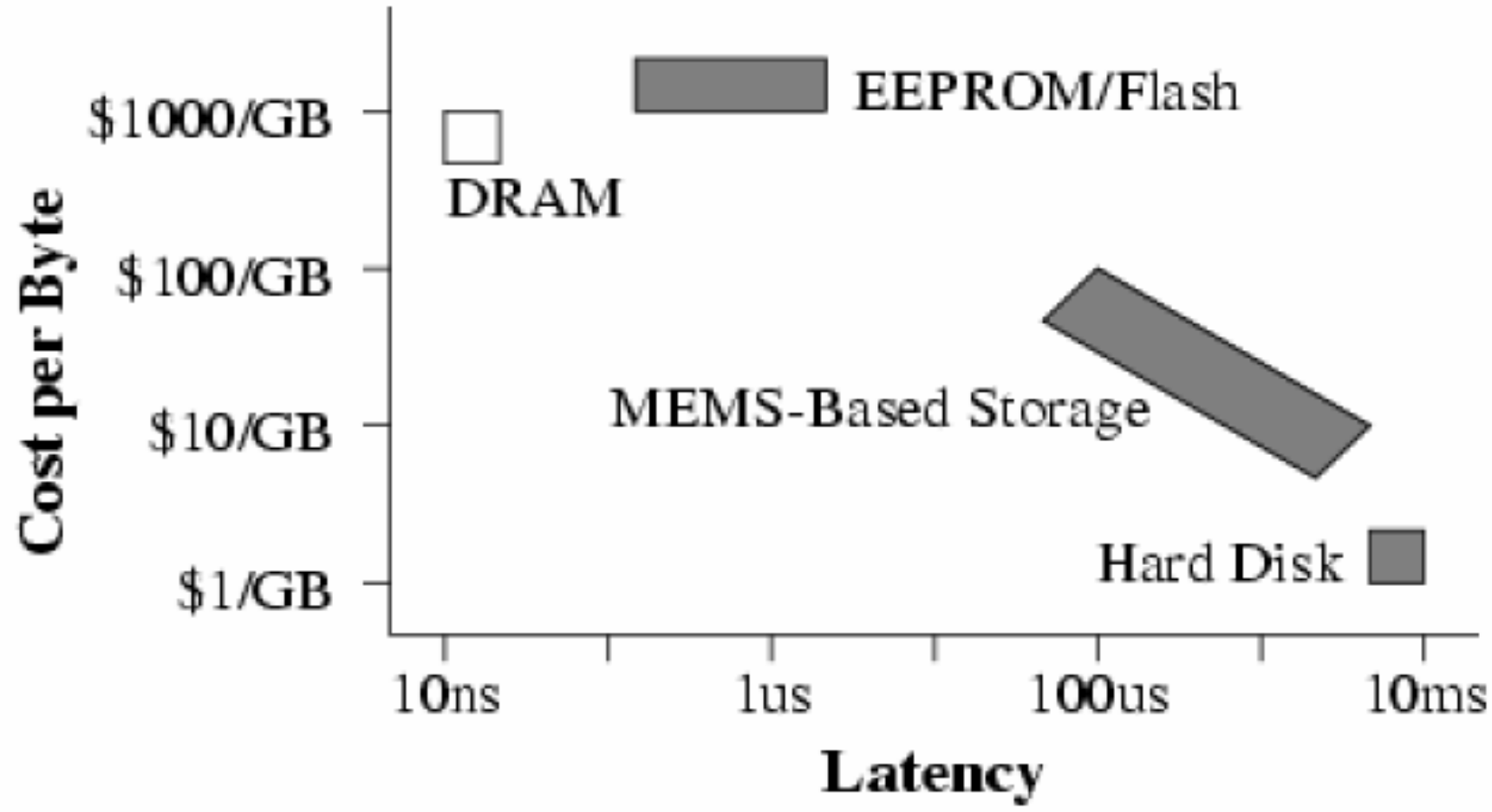
- Pusvadītāju iekārtas
  - 100,000 ieraksta/dzēšanas cikli
  - Nodroses (Standby) strāva = 100uA, ieraksta strāva = 45mA
  - Pārraides ātrums 3.5 MB/s, nolasīšanas piekļuves laiks 65-150ns
  - Compact Flash (2002) 256MB=$73 512MB=$170, 1GB=$560
  - Compact Flash (2004) 256MB=$39 512MB=$80 1GB=$146 2GB=$315 4GB=$800
  - Compact Flash (2005) 256MB=$21 512MB=$36 1GB=$61 2GB=$139 4GB=$276
  - Compact Flash (2006) 512MB=$35 1GB=$55 2GB=$90 4GB=$220
- Flash vs. HDD
  - Praktiski momentāna izeja no nodroses režīma
  - Iespējama (un pat ieteicama) efektīva gadījuma rakstura piekļuve
  - Noturīgs pret triecieniem un vibrāciju (1000G darba režīmā)

Alternatīvas (MEMS)

- Micro-Electro-Mechanical Systems



Alternatīvas



Ja nepietiek ar vienu disku?

- Disku masīvi potenciāli:
  - Sniedz lielāku I/O veiktspēju (IOPS)
  - Sniedz lielāku blīvumu (MB/m^3)
  - Sniedz mazāku jaudas patēriņu (MB/Kw)
  - Bet uzticamība?
- N Disku uzticamība = (1. Diska uzticamība)/N
- MTTF = Mean Time to Failure
- 50 000 h ÷ 70 diskiem = 700 stundas
- Disku masīvam MTTF samazinās no 6. gadiem uz 1. mēnesi!
- ***Masīvi bez redundances ir praktiski nelietojami***

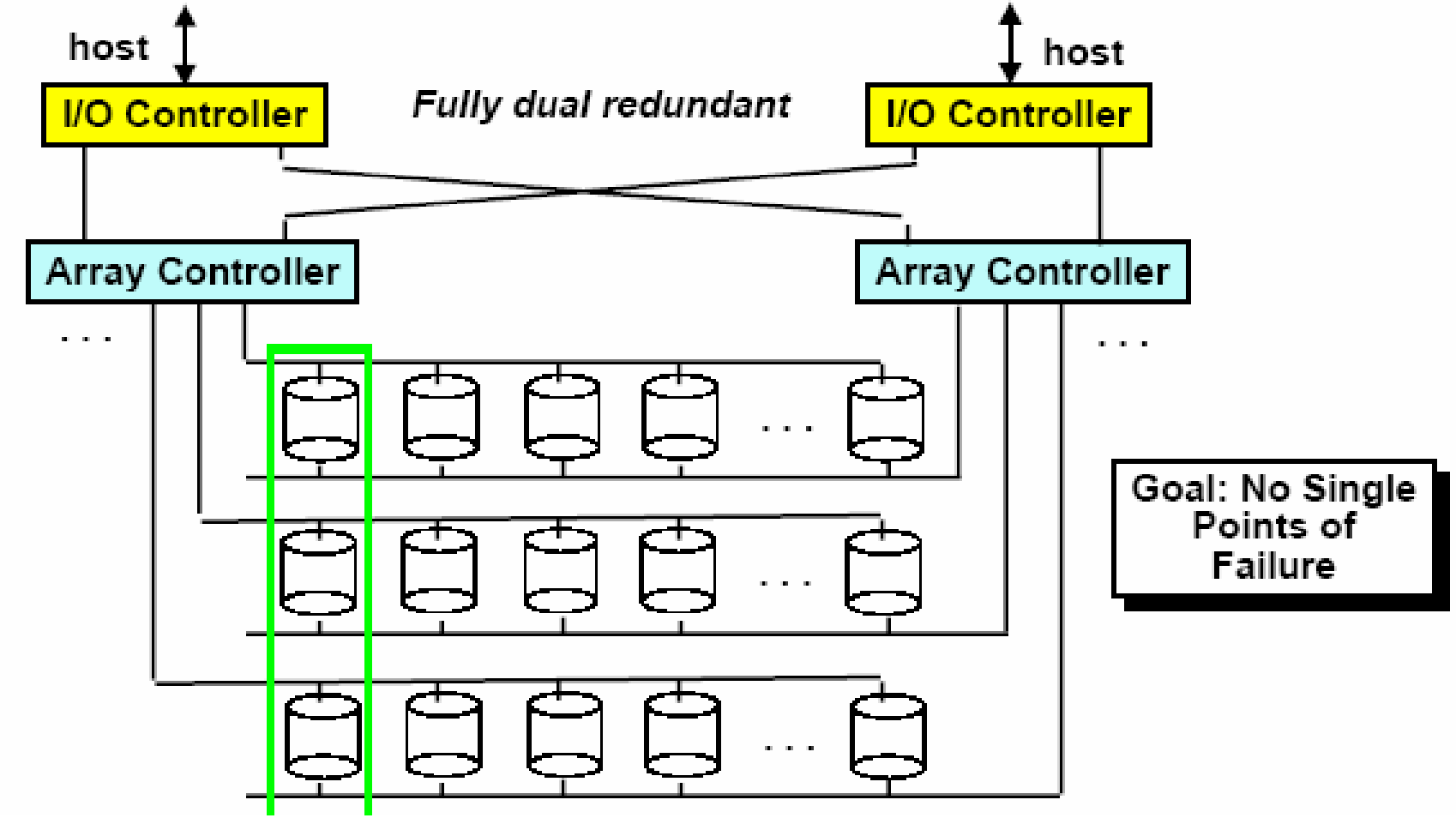
Uzticamība/Pieejamība

- Cilvēks jebkuru sistēmu iztver kā divos stāvokļos esošu:
  - Normāls serviss (kaut kad būs atteikums)
  - Servisa pārtraukums (kaut kad notiks atjaunošana)
- MTTR - mean time to repare
- ***Pieejamība*** (%) = MTTF / (MTTF + MTTR) – MTBF
- Tas ir cik % no sistēmas darba laika tā strādās (99.999% – ... min downtime gadā)
- ***Uzticamība*** - varbūtība ka sistēma strādās izvēlētajā laikā dotajos apstākļos. Obligāti jāzina kādam periodam dati ir doti, kas ir atteikums un kādi ir apstākļi pie kuriem mēra. 99,999% - 0.001 atteikums 100 gados
- ***Izvairīšanās*** no defektiem “fault avoidance” – panāk ar konstruēšanas palīdzību
- Defektu ***noturību*** “fault tolerance” – panāk ar dublēšanu
- ***Atkļūdošanu*** “error removal” – panāk ar verifikāciju
- Kļūdu ***prognozēšanu*** “error forecasting” – panāk ar atestāciju “evaluation”

RAID

- Redundant Arrays of (Inexpensive) Disks
- Tiek nodrošināta ***pieejamība*** kaut arī diski "***mirs"*** tāpat
- Dati tiek atjaunoti no dublētās informācijas
  - Papildus tēriņi dublējošās informācijas izveidei, atjaunošanai
  - Papildus vieta dublējošās informācijas glabāšanai
- RAID1 – spoguļošana (100% virstēriņi)
- RAID0 – nav dublēšanas (bet ir veiktspēja 2x)
- RAID3 – vismaz viens "lieks" paritātes disks (diemžēl vienalicīgai tas ir arī ***šaurā vieta***)
- RAID5 – paritātes informācija "izsmērēta" pa visiem diskiem (nav vienas šaurās vietas un ir mazāks darbību apjoms ieraksta laikā
- Kombinācijas no šiem pamata veidiem
- RAID ***neatbrīvo no rezerves kopiju veidošanas nepieciešamības***!

Sistēmas līmeņa pieejamība



Kas veic I/O

- I/O apstrādi veic
  - Lietotnes
  - DMA
  - I/O procesori (IOPs)

Kas veic I/O

- CPU
  - Tieši izpilda visas I/O darbības
  - Atmiņas adresēs iekļautā I/O “Memory Mapped”
  - Speciālas ISA I/O komandas (x86, IBM 370)
  - Pārtraukumu sistēma, programmaptauja, hibrīdās shēmas(reāllaika sistēmām)
  - Liels virstēriņš un potenciāli notiek kešu satura maiņa
  - Nav problēmu ar datu koherenci
- I/O procesors (IOP vai kanāla procesors)
  - (speciāls vai vispārējas nozīmes) procesors kas speciāli veltīts I/O darbībām
  - Ātri
  - Var būt pārmērība “overkill”, kešu koherences problēmas
    - I/O redzēs vecus datus izvadot (atmiņa var nebūt atjaunota)
    - CPU redz vecus datus kešā jo I/O sistēma atjauno tikai atmiņas saturu

Kas veic I/O

- *DMAC (direct memory access controller)*
  - Var pārvietot datus uz/no atmiņas sākot ar uzstādīto adresi
  - ātri un vienkārši
  - Vienalga var būt koherences problēmas
  - Jāpieslēdz pa tiešo atmiņas kopnei

Mājās

- http://www.storagereview.com/map/lm.cgi/seek
- http://www.cs.ucla.edu/~kohler/class/05s-osp/notes/notes12.html
- http://www.zurich.ibm.com/st/storage/concept.html#
- http://en.wikipedia.org/wiki/Hard_disk
- http://en.wikipedia.org/wiki/Availability
- http://www.google.com/search?q=define:reliability
- http://www.ecs.umass.edu/ece/koren/architecture/Raid/basicRAID.html
- http://www.cs.rtu.lv/Pubs/Cipa/Arhit2003/WPad/ARH8_1.doc
- http://www.cs.rtu.lv/Pubs/Cipa/Arhit2003/WPad/ARH11_1A.DOC