



PB 169

Počítačové sítě a operační systémy

I/O systém

Vnější paměti



Hardware

- HW pro I/O je značně rozmanitý
- Existují však určité běžně používané prvky
 - port
 - sběrnice (bus)
 - řadič (host adapter, controller)
- I/O zařízení jsou řízeny I/O instrukcemi (IN, OUT)
- Adresy I/O zařízení
 - uváděné přímo v I/O instrukcích (např. IN AL, DX : DX port, AL získaný bajt)
 - I/O se mapuje na přístup k paměti (např. grafická karta, videopaměť)
- Základní způsoby ovládání I/O
 - polling, programované I/O operace
 - aktivní čekání na konec operace
 - přerušení
 - DMA



Rozmístění I/O portů v PC

I/O address range (hexadecimal)	device
000–00F	DMA controller
020–021	interrupt controller
040–043	timer
200–20F	game controller
2F8–2FF	serial port (secondary)
320–32F	hard-disk controller
378–37F	parallel port
3D0–3DF	graphics controller
3F0–3F7	diskette-drive controller
3F8–3FF	serial port (primary)

Techniky provádění I/O

- Programovaný I/O (busy-waiting)
 - opakovaně se ptám na stav zařízení
 - připraven
 - pracuje
 - chyba
- I/O řízený přerušením
 - zahájení I/O pomocí I/O příkazu
 - paralelní běh I/O s během procesoru
 - I/O modul oznamuje přerušením konec přenosu
- Direct Memory Access (DMA)
 - kopírování bloků mezi pamětí a I/O zařízením na principu kradení cyklů paměti
 - přerušování po přenosu bloku (indikace konce)



Přerušeni

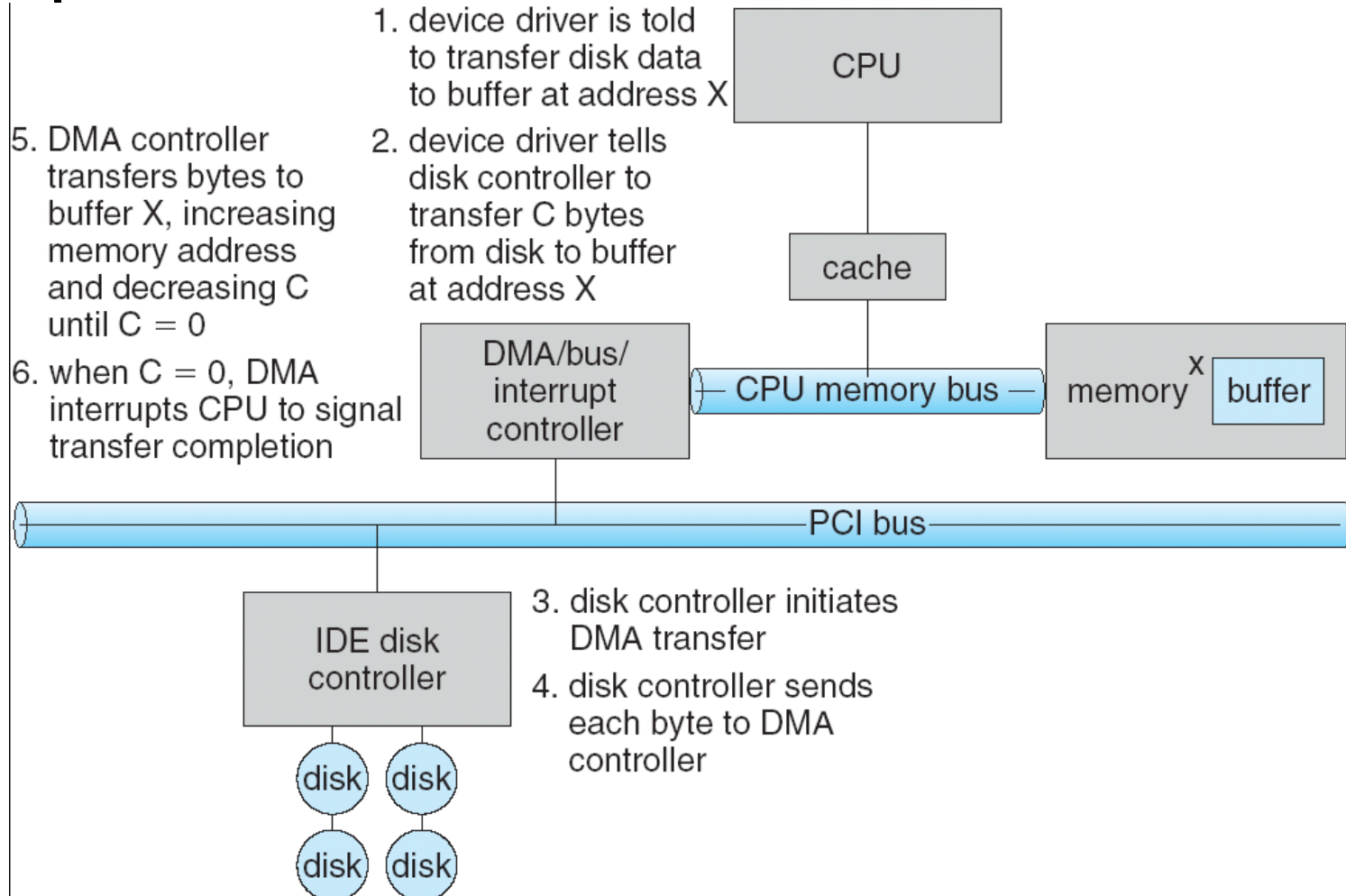
- Přerušeni obsluhuje ovladač přerušeni (kód OS)
- Maskováním lze některá přerušeni ignorovat nebo oddálit jejich obsluhu
- Patřičný ovladač přerušeni se vybírá přerušovacím vektorem
 - některá přerušeni nelze maskovat
 - přerušeni mohou být uspořádána podle priorit
- Přerušeni se používá i pro řešení výjimek (nejsou asynchronní)



DMA

- Přímý přístup do paměti (Direct Memory Access - DMA)
 - nahrazuje programovaný I/O při velkých přesunech dat
 - vyžaduje speciální DMA řadič
 - při přenosu dat se obchází procesor, přístup do paměti zajišťuje přímo DMA řadič
 - procesor a DMA soutěží o přístup k paměti

DMA: příklad



Aplikační rozhraní I/O

- Jádro OS se snaží skrýt rozdíly mezi I/O zařízeními a programátorům poskytuje jednotné rozhraní
- Dále vrstva ovladačů ukrývá rozdílnost chování I/O řadičů i před některými částmi jádra
- Některé vlastnosti I/O zařízení
 - mód přenosu dat: znakové (terminál) / blokové (disk)
 - způsob přístupu: sekvenční (modem) / přímý (disk)
 - sdílené/dedikované: klávesnice / páska
 - rychlost přenosu: vystavení, přenos, ...
 - read-write, read only, write only



Bloková a znaková zařízení

- Bloková zařízení – typicky disk
 - příkazy: read, write, seek
 - logický způsob přístupu: obecný I/O nebo souborový systém
 - možný přístup formou souboru mapovaného do paměti
- Znaková – klávesnice, myš, sériový port
 - příkazy: get, put
 - nad nimi knihovní podprogramy pro další možnosti (např. řádková editace)

Síťová zařízení

- Přístup k nim se značně liší jak od znakových, tak od blokových zařízení
 - proto mívají samostatné rozhraní OS
- Unix i Windows obsahující rozhraní nazývané „sockets“
 - separují síťové protokoly od síťových operací
 - přístup jako k souborům (včetně funkce *select*)
- Existuje celá řada přístupů k síťovým službám
 - Pipes (roury), FIFOs, streams, queues, mailboxes



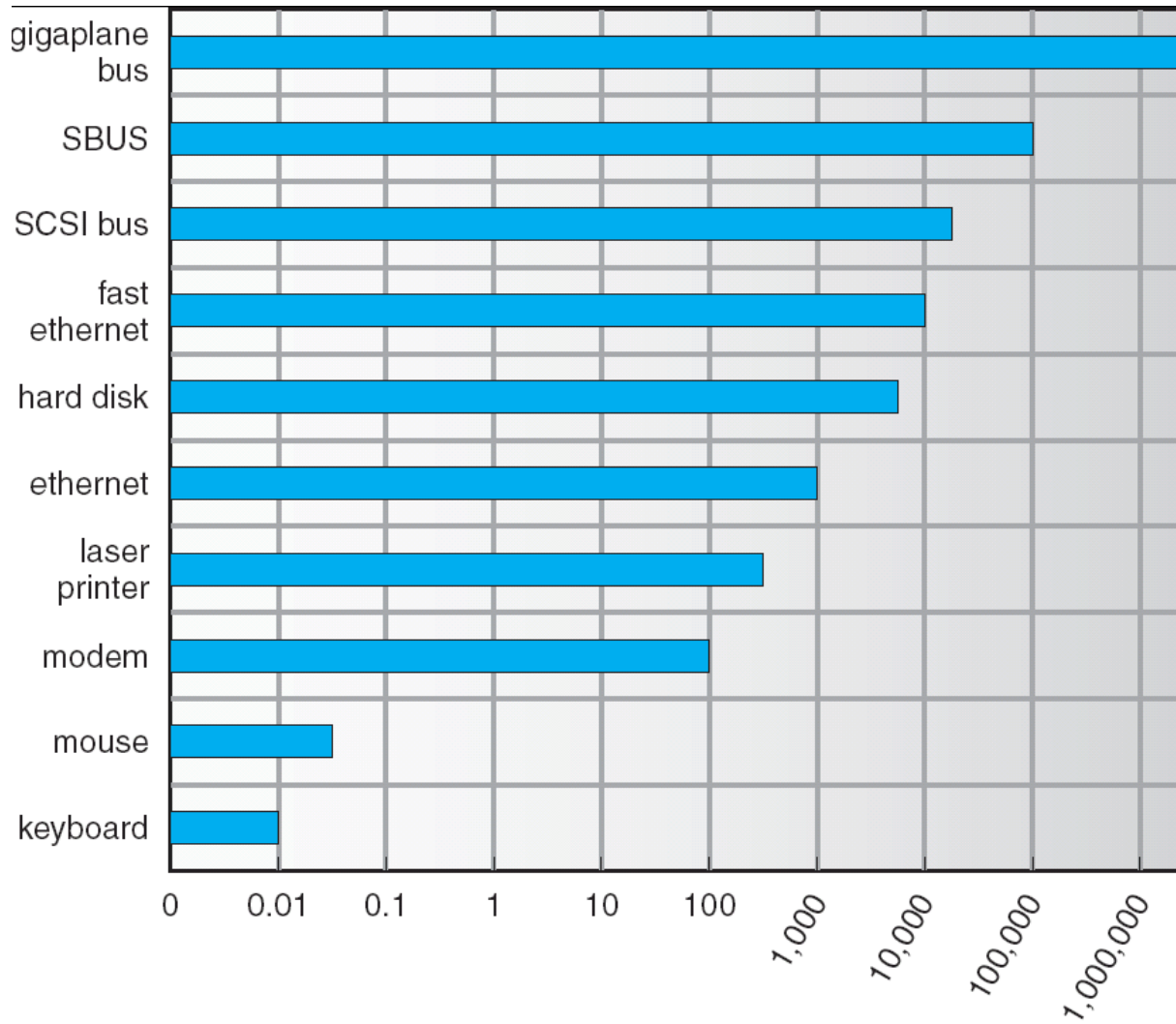
Blokující a neblokující I/O

- Blokující
 - z hlediska procesu synchronní
 - proces čeká na ukončení I/O
 - snadné použití (programování), snadné porozumění (po provedení operace je hotovo to co jsem požadoval)
 - někdy však není dostačující (z důvodu efektivity)
- Neblokující
 - řízení se procesu vrací co nejdříve po zadání požadavku
 - vhodné pro uživatelské rozhraní, bufferovaný I/O
 - bývá implementováno pomocí vláken
 - okamžitě vrací počet načtených či zapsaných znaků
- Asynchronní
 - proces běží souběžně s I/O
 - konec I/O je procesu hlášen signály
 - obtížné na programování, složité používání, ale v případě vhodně promyšleného programu velice efektivní

I/O subsystém v jádru

- Plánování
 - některé I/O operace požadují řazení do front na zařízení
 - některé OS se snaží o „spravedlnost“
- Vyrovnání (vyrovnávací paměti), buffering
 - ukládání dat v paměti v době přenosu k/ze zařízení
 - řeší rozdílnost rychlosti
 - řeší rozdílnost velikosti datových jednotek

Příklad: Sun Enterprise 6000





I/O Subsystem v jádru

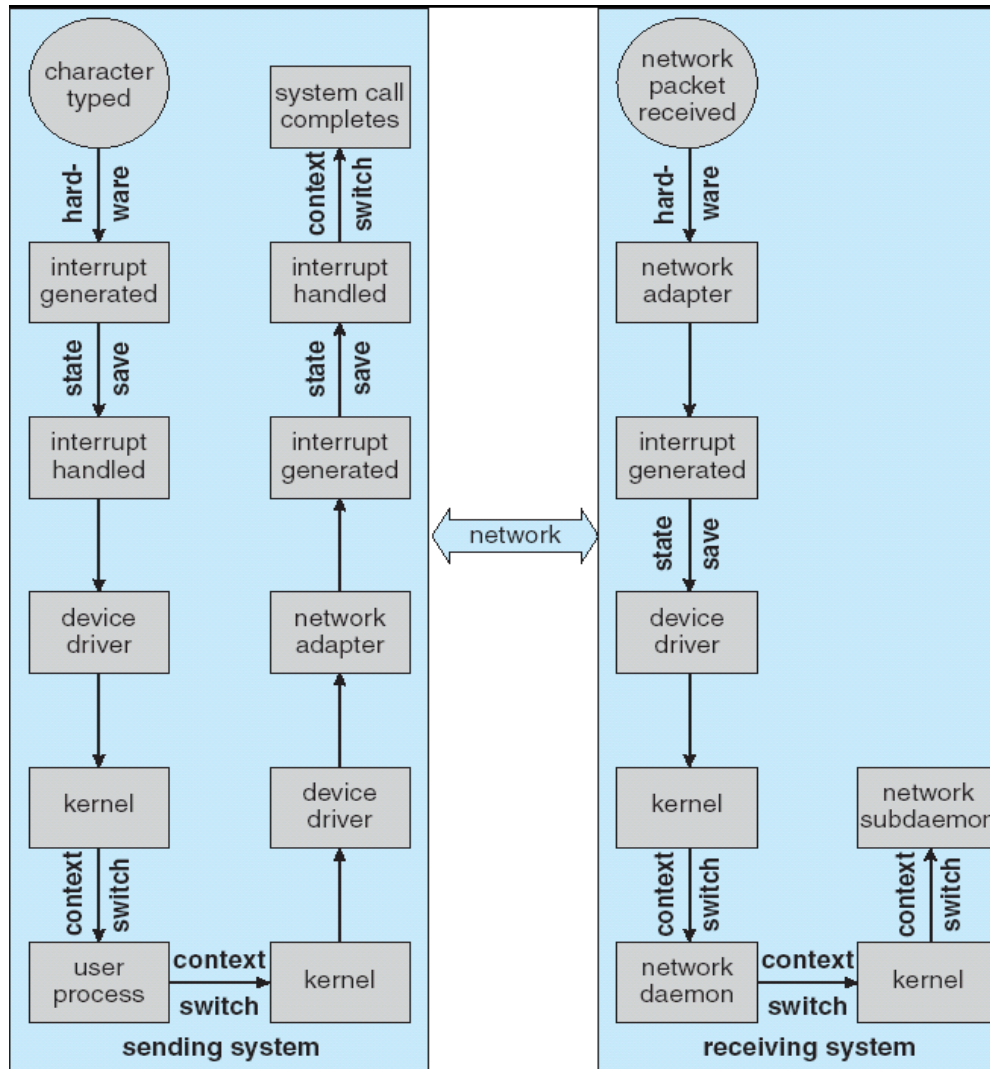
- Caching
 - rychlá paměť udržuje kopii dat
 - vždy pouze *kopii*
 - caching je klíčem k dosažení vysokého výkonu
- Spooling
 - udržování *fronty dat* určených k výpis na zařízení
 - pokud zařízení může vyřizovat požadavky pouze sekvenčně
 - typicky tiskárna
- Rezervace zařízení
 - exkluzivita přístupu k zařízení pro proces
 - rezervace / uvolnění – volání systému
 - pozor na uváznutí (deadlock)



Výkon

- I/O je nejvýznamnějším faktorem výkonu celého systému
 - CPU musí provádět ovladače a programy I/O části jádra
 - při přerušení se přepíná kontext
 - provádí se kopírování dat
 - zvláště významný je síťový provoz

Příklad: Síťová aplikace

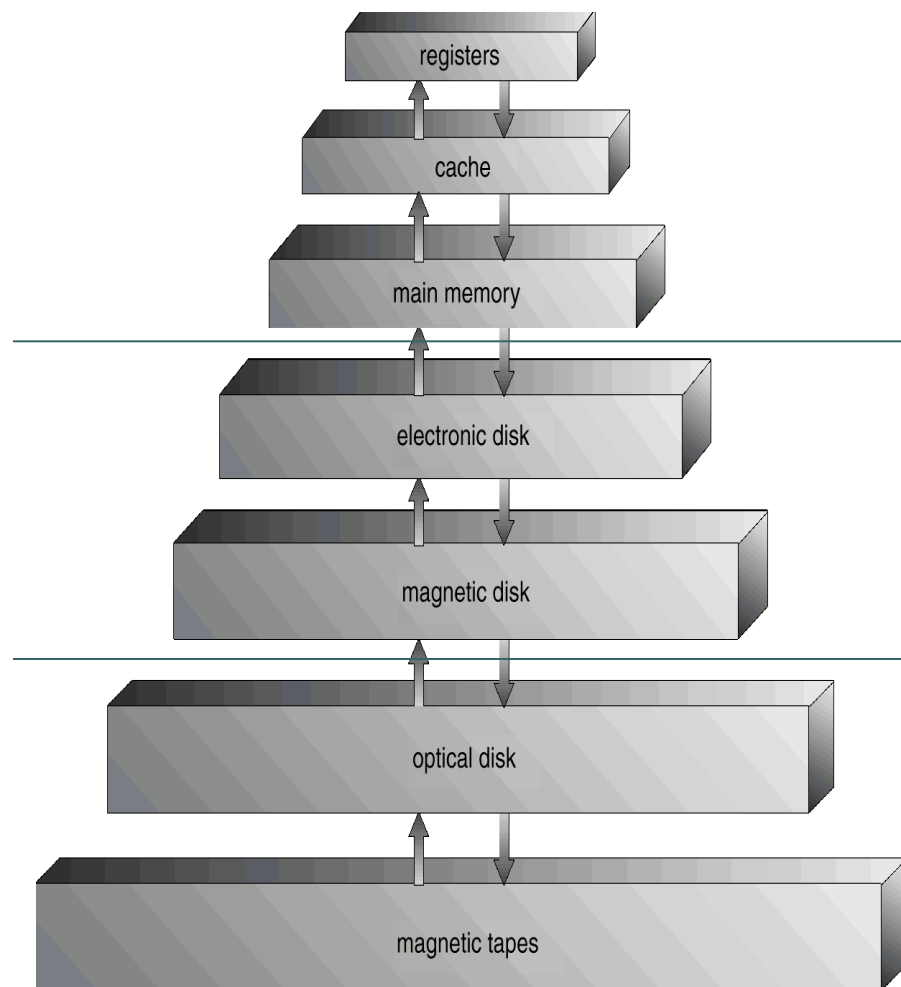


Zvyšování výkonu

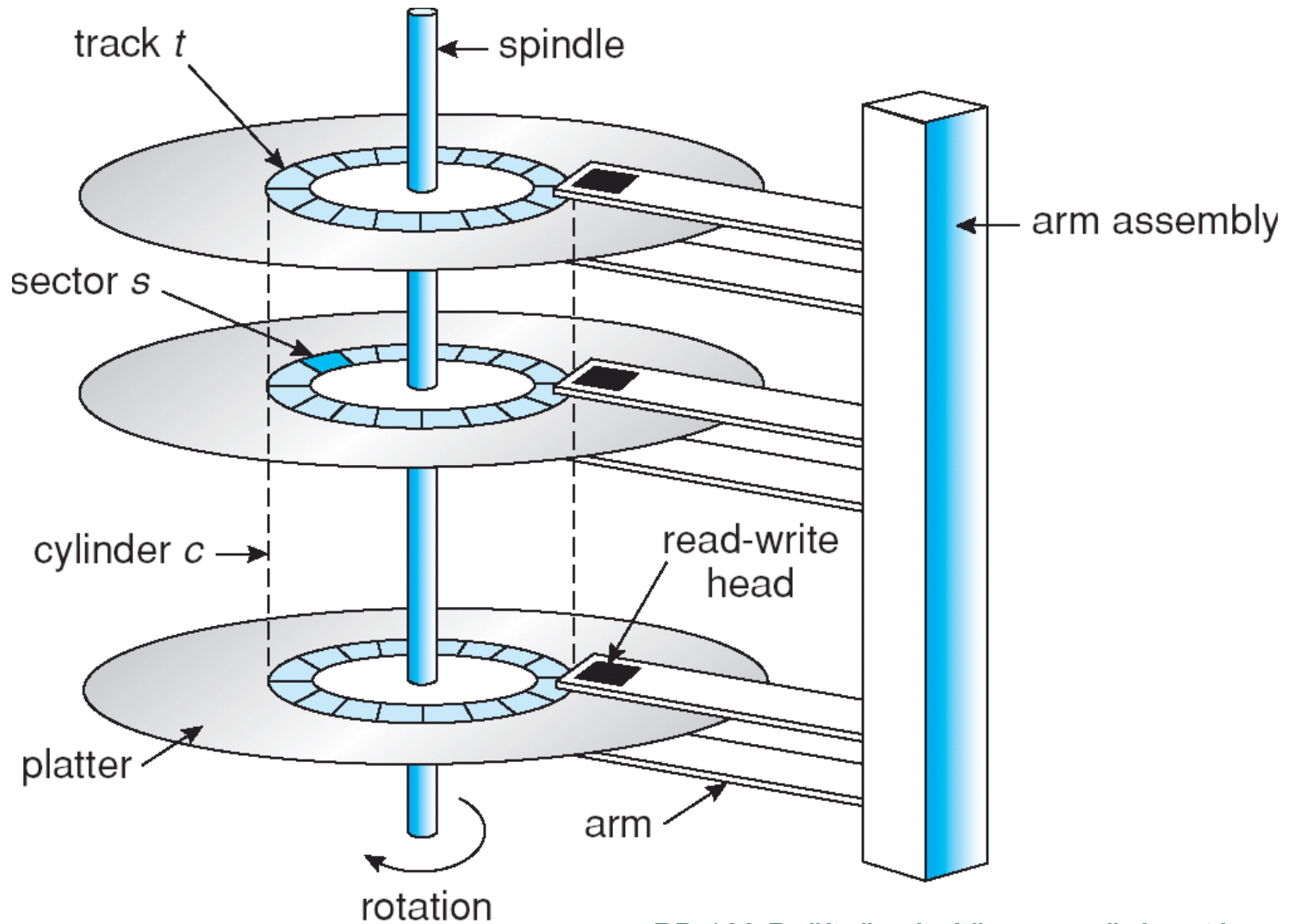
- Omezujeme počet přepnutí kontextu
- Omezujeme zbytečné kopírování dat
- Omezujeme počet přerušení tím, že přenášíme delší bloky
- Využíváme všech výhod (funkcí) moderních řadičů
- Používáme co nejvíce DMA
- Všechny komponenty kombinujeme s cílem dosažení co nejvyšší propustnosti
 - CPU, paměť, sběrnice, I/O zařízení

Paměťová hierarchie

- Primární paměti
 - nejrychlejší
 - energeticky závislé
 - Cache, hlavní (operační) paměť
- Sekundární paměti
 - středně rychlé
 - energeticky nezávislé
 - Také nazývané „on-line storage“
 - flash disky, magnetické disky
- Terciální paměti
 - levná typicky vyměnitelná média
 - pomalé
 - energeticky nezávislé
 - také nazývané „off-line storage“
 - floppy disky, magnetické pásky, optické disky



Magnetické disky





Struktura disku

- Diskové mechanismy se adresují jako velká 1-dimensionální pole logických bloků
 - logické bloky jsou nejmenší jednotkou přenosu dat
- 1-dimensionální pole logických bloků je zobrazováno do sektorů disku sekvenčně
 - sektor 0
 - první sektor na první stopě vnějšího cylindru
 - zobrazování pokračuje po této stopě, potom po ostatních stopách tohoto cylindru, a potom po cylindrech směrem ke středu



Plánování disku

- OS je odpovědný za efektivní používání hardware
 - pro disky: co nejrychlejší přístup a co největší šířka pásma
- Doba přístupu (*access time*) je dána:
 - dobou vystavení (*seek time*) – na cylindr se stopou s adresovaným sektorem
 - dobou rotačního zpoždění – dodatečná doba do průchodu adresovaného sektoru pod čtecí/zápisovou hlavou
- Minimalizace doby vystavení
 - doba vystavení \approx vystavovací vzdálenosti
 - řeší plánování činnosti disku
- Rotační zpoždění
 - shora omezeno konstantou
- Šířka pásma
 - počet přenesených bytů / doba od zadání skupiny požadavků do jejich ukončení
 - převzatý pojem z telekomunikací



Plánování disku

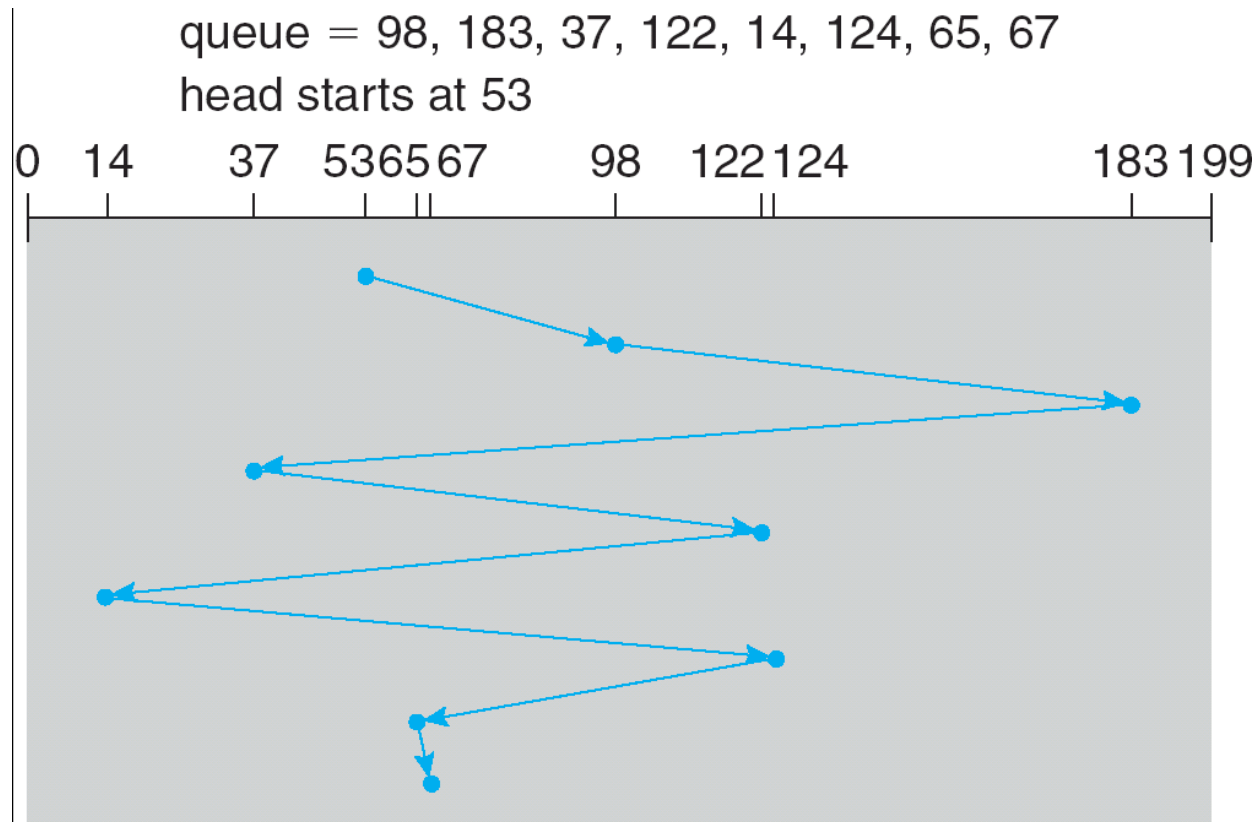
- Existuje celá řada algoritmů pro plánování přístupu na disk.
- Příklad: vzorová fronta požadavků na přístup k disku (máme cylindry 0-199).

98, 183, 37, 122, 14, 124, 65, 67

Hlavička disku vystavena na pozici 53

FCFS

- Celkem přesun o 640 cylindrů





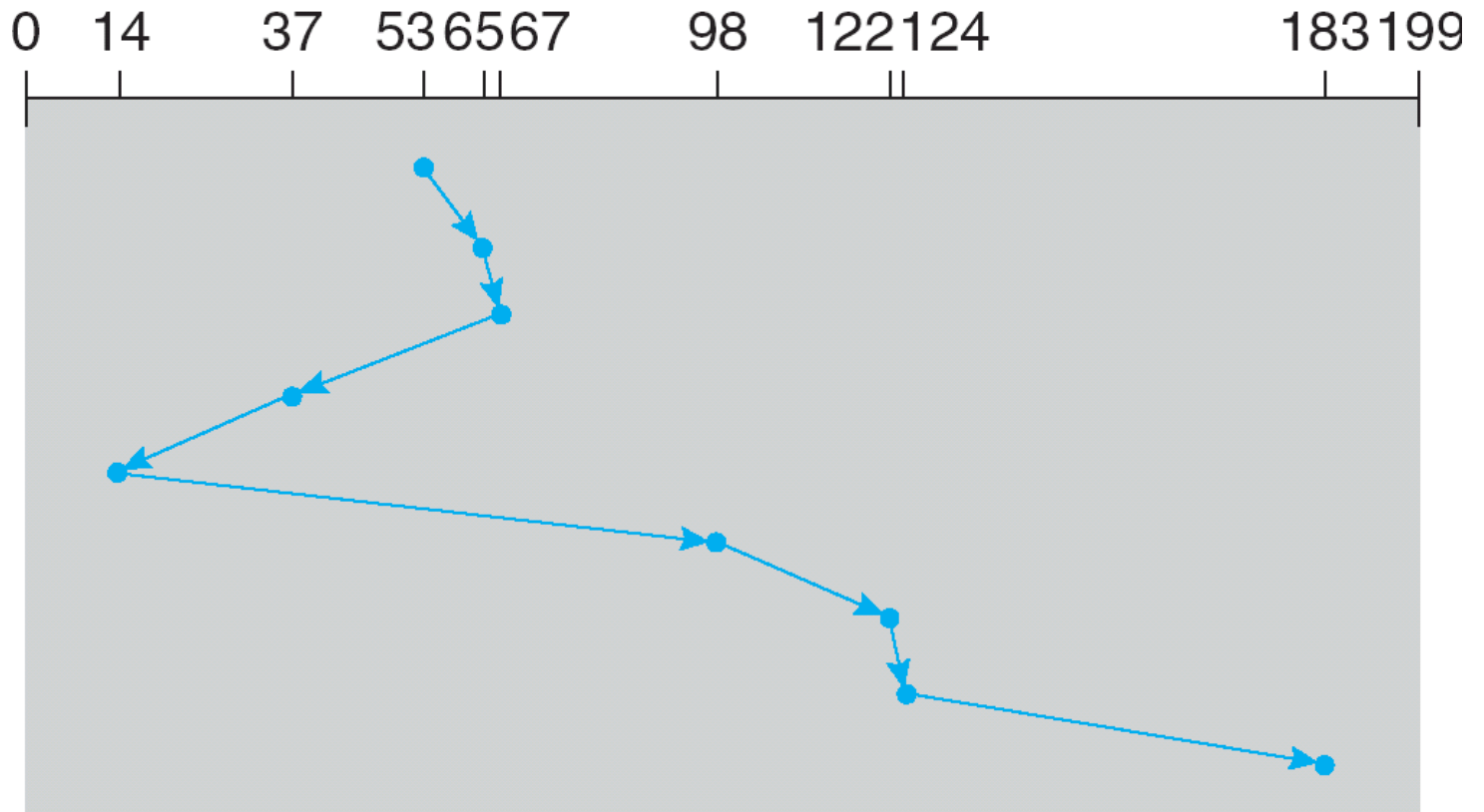
SSTF

- Z fronty požadavků vybírá ten požadavek, který vyžaduje minimální dobu vystavení od současné pozice hlavičky
- SSTF (*shortest seek time first*) algoritmus je variantou algoritmu SJF (*shortest job first*); může způsobit stárnutí požadavků.
- Náš příklad vyžaduje přesun o 236 cylindrů

SSTF

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53





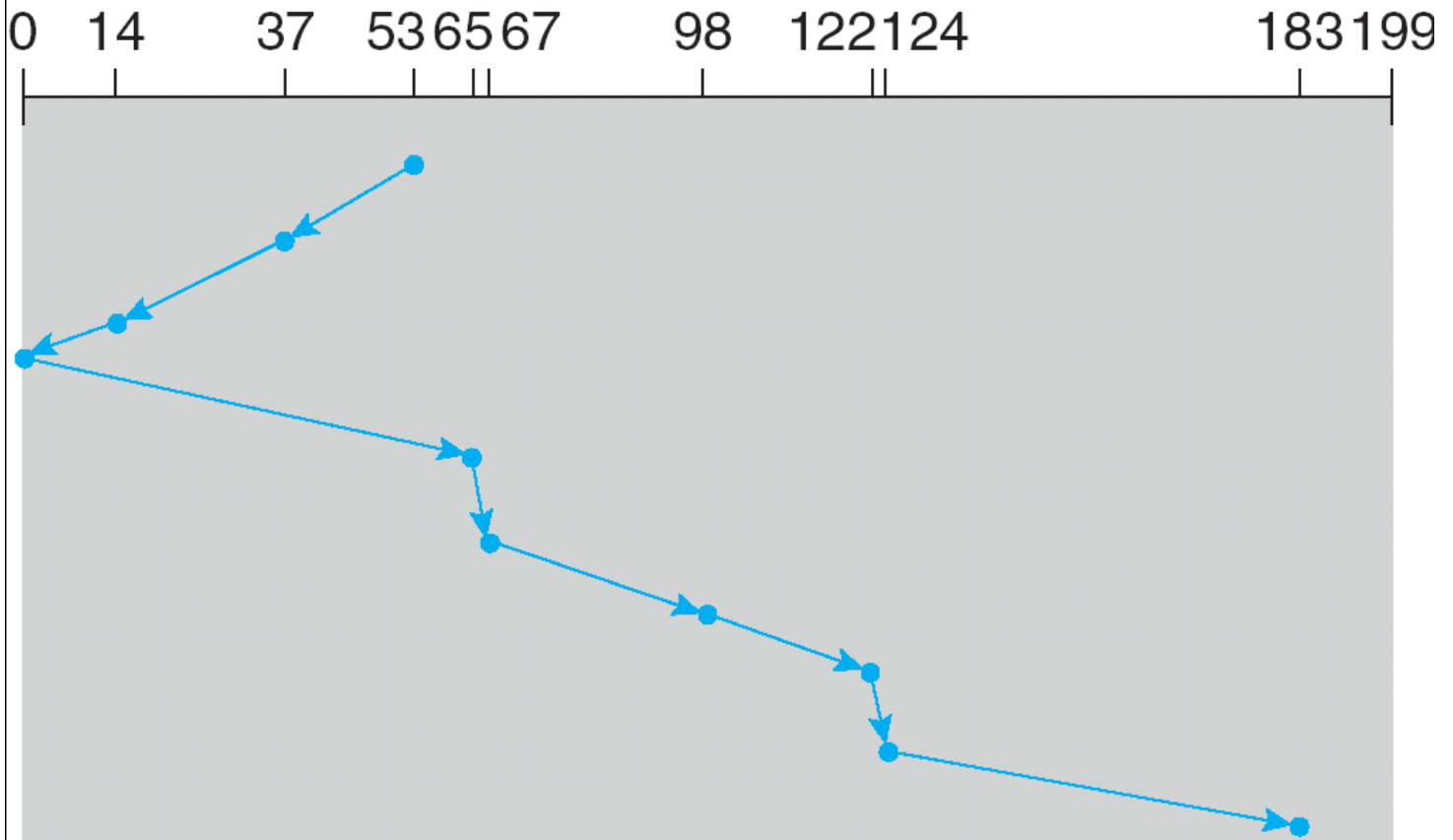
SCAN

- Hlavička disku začíná na jedné straně disku a přesunuje se při splňování požadavků ke druhé straně disku. Pak se vrací zpět a opět plní požadavky.
- Někdy nazývané *algoritmus typu výtah*.
- Náš příklad vyžaduje přesun o 208 cylindrů.

SCAN

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53





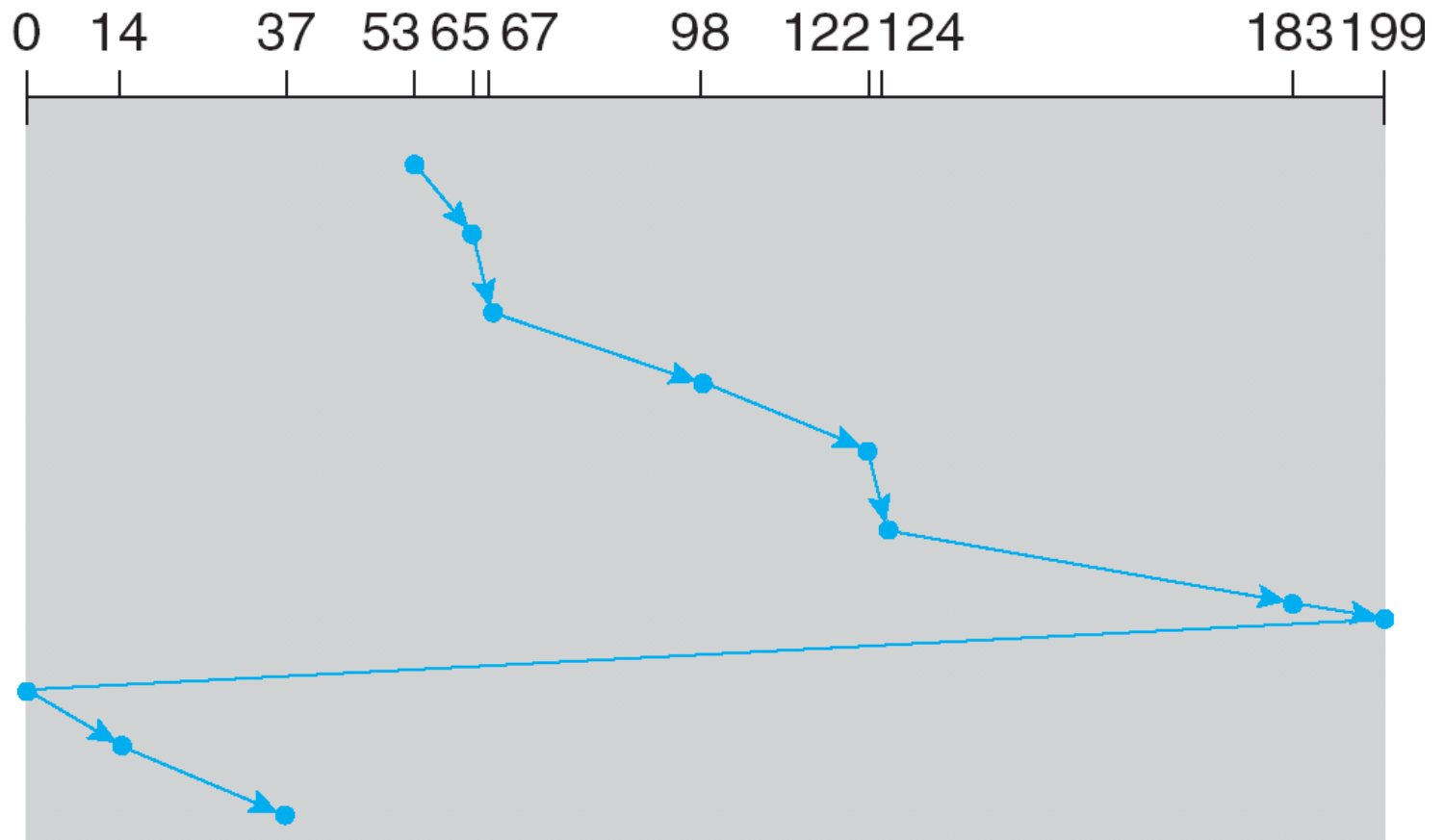
C-SCAN

- Poskytuje jednotnější čekací dobu než SCAN
- Hlavička se posouvá z jednoho konce disku na druhý a zpracovává požadavky. Potom se vrací zpět bez vyřizování požadavků a opět začíná vyřizovat požadavky z prvního konce.
- Cylindry považuje za kruhový seznam, který za posledním cylindrem pokračuje opět prvním cylindrem.

C-SCAN

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53





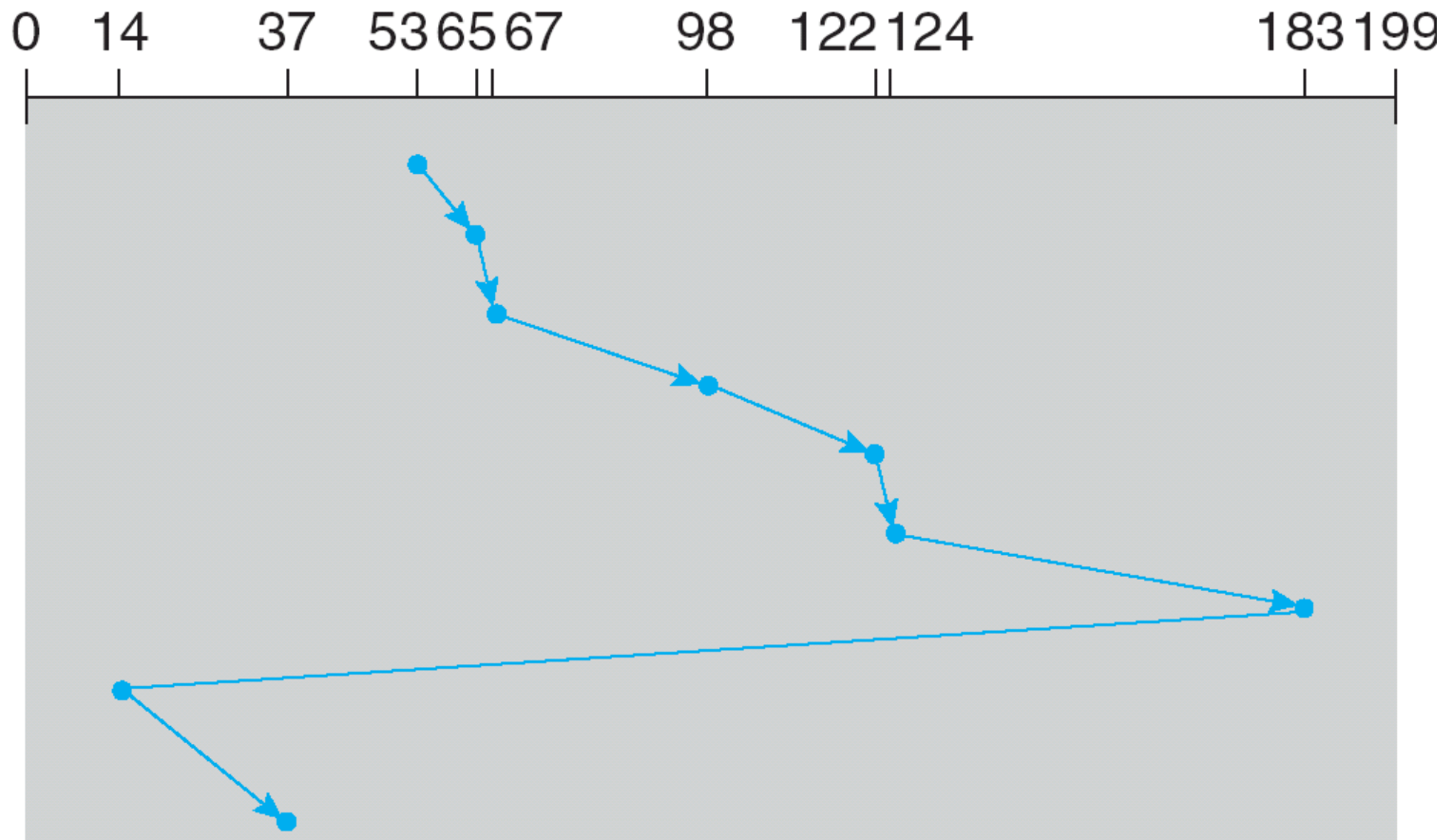
C-LOOK

- Obdoba C-SCAN, ale hlavička jen potud do kraje, pokud existují požadavky.
- Pak se vrací zpět

C-LOOK

queue 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



Výběr algoritmu

- SSTF je přirozený, má přirozené chápání
- SCAN a C-SCAN jsou vhodnější pro těžkou zátěž disku
- Výkon závisí na počtu a typech požadavků
- Požadavky na disk mohou být ovlivněny metodami organizace souborů v souborovém systému
- Plánovací algoritmus by měl být napsán jako samostatný modul, aby plánovací algoritmus OS bylo možné zaměňovat
- Častá implicitní volba bývá SSTF nebo LOOK

Moderní HW

- U moderních disků nemusí být známé mapování logických bloků na fyzické adresy
- Disku předáme skupinu požadavků a disk si pořadí optimalizuje sám
- OS přesto může mít zájem na vlastním řazení požadavků
 - prioritizace I/O operací z důvodu výpadků stránek
 - Pořadí operací zápisu dat a metadat souborového systému



Technologie RAID

- RAID: Redundant Arrays of Independent (Inexpensive) Disks
 - organizace disků řízená tak, že poskytuje objem jednoho disku
 - s velkou kapacitou a rychlostí díky tomu, že mnoho disků pracuje paralelně
 - s velkou spolehlivostí, data se uchovávají redundantně, lze je obnovit i po poruše některého z disků
- Pravděpodobnost, že některý disk z množiny N disků selže je mnohem vyšší, než pravděpodobnost, že selže jediný disk
 - $N = 100$ disků, každý má MTTF = 100 000 hodin (cca 11 let), celý systém bude mít MTTF = 1000 hodin (cca 41 dní)
 - techniky na bázi redundance chránící před ztrátou dat jsou pro systémy s velkým počtem komponent (disků) kritické
- Původní záměr
 - levná alternativa nahrazující velké drahé disky
 - „I“ je interpretováno jako „independent“

RAID: zvýšení spolehlivosti

○ Redundance

- nadbytečnost, doplňková informace použitelná pro obnovu informace po poruše (disku)

○ Zrcadlení (stínování), Mirroring (shadowing)

- každý disk je duplikován, 1 logický disk je tvořen 2 fyzickými disky
- každý zápis se provede na obou discích, čte se z jednoho disku
- jestliže se jeden disk porouchá, data jsou k dispozici na druhém disku
 - ke ztrátě dat dojde při výpadku obou disků, když zrcadlový disk selže dříve, než se systém opraví
- průměrná doba do ztráty dat závisí na průměrné době do poruchy a průměrné doby opravy
 - Např. MTTF = 100 000 hodin, průměrná doba opravy 10 hodin, dává u zrcadlené dvojice disků průměrnou dobu ztráty dat $500 \cdot 10^6$ hodin (čili 57 000 let), když budeme ignorovat požáry apod.



RAID: zvýšení výkonu

- Dva hlavní cíle paralelismu v diskových systémech
 - zvýšení propustnosti vyvážením zátěže malými přístupy
 - paralelizace velkých přístupů s cílem zkrácení doby odpovědi
- Zvýšení přenosové rychlosti paralelním zápisem do více disků (dělení, striping)
 - bit-level striping
 - dělení bitů každého bytu mezi samostatné disky
 - v poli 8 disků se zapisuje bit i každého bytu na disk i
 - čtení dat probíhá 8x rychleji než z jednoho disku
 - vystavení je delší než v případě jednoho disku
 - dnes se bit-level striping de facto už nepoužívá
 - blok-level striping
 - systém s n disky, blok souboru i se zapisuje na disk $(i \bmod n) + 1$
 - požadavky na různé bloky se mohou realizovat paralelně, jestliže bloky leží na různých discích
 - požadavek na dlouhou posloupnost bloků může použít všechny disky paralelně

Úrovně RAID, přehled

RAID Level 0: Žádná redundance, jen souběžnost

RAID Level 1: Spolehlivost dosažená zrcadlením disků

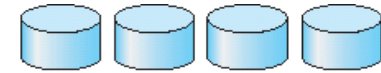
RAID Level 2: Hamming code error correction

RAID Level 3: 1 kontrolní disk na skupinu, dělení bitů

RAID Level 4: Nezávislé operace read/write, dělení bloků

RAID Level 5: Data/parity přes všechny disky (více souběžný přístup)

RAID Level 6: Odolnost při více než jedné poruše disku



(a) RAID 0: non-redundant striping.



(b) RAID 1: mirrored disks.



(c) RAID 2: memory-style error-correcting codes.



(d) RAID 3: bit-interleaved parity.



(e) RAID 4: block-interleaved parity.



(f) RAID 5: block-interleaved distributed parity.



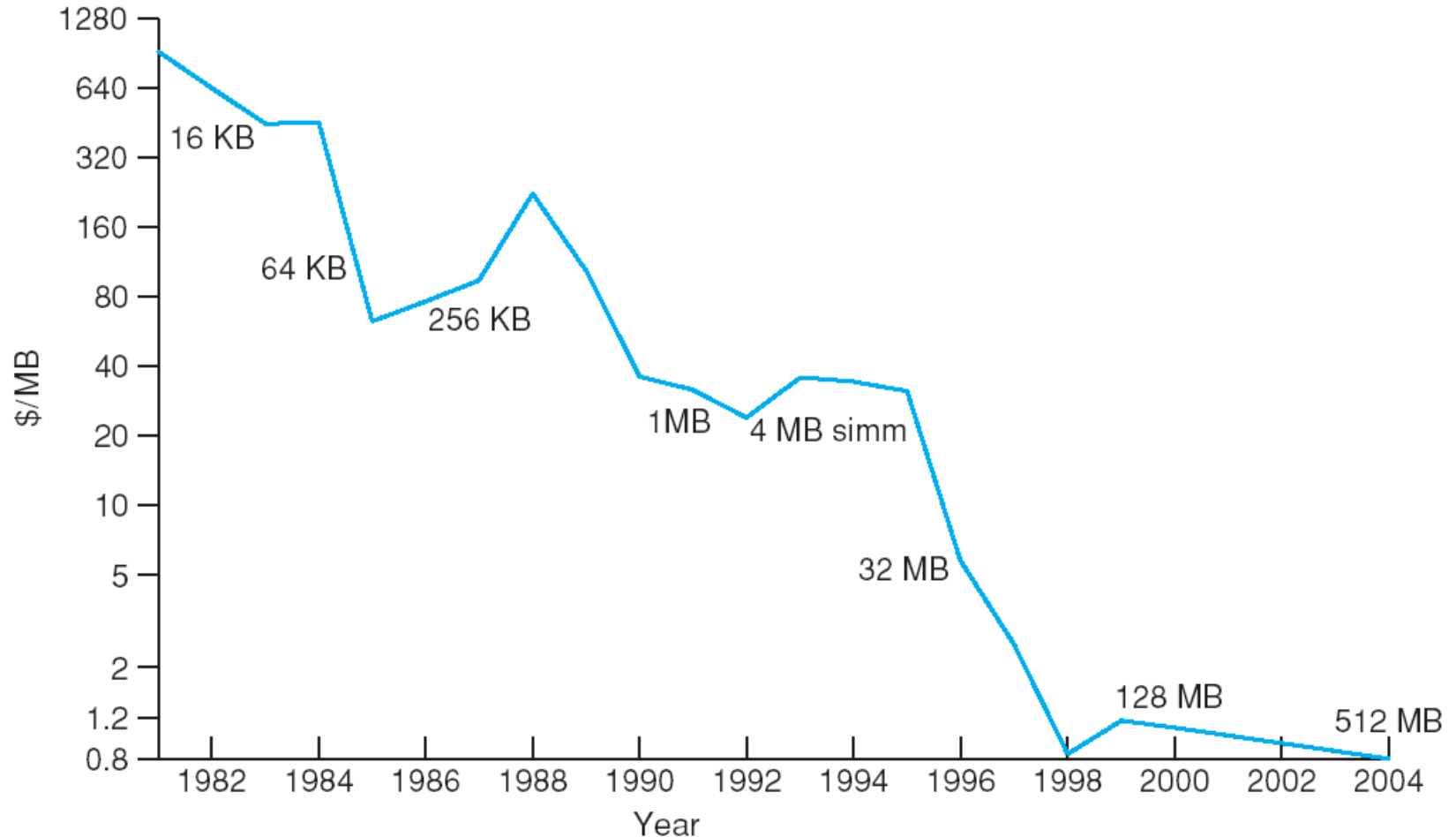
(g) RAID 6: P + Q redundancy.



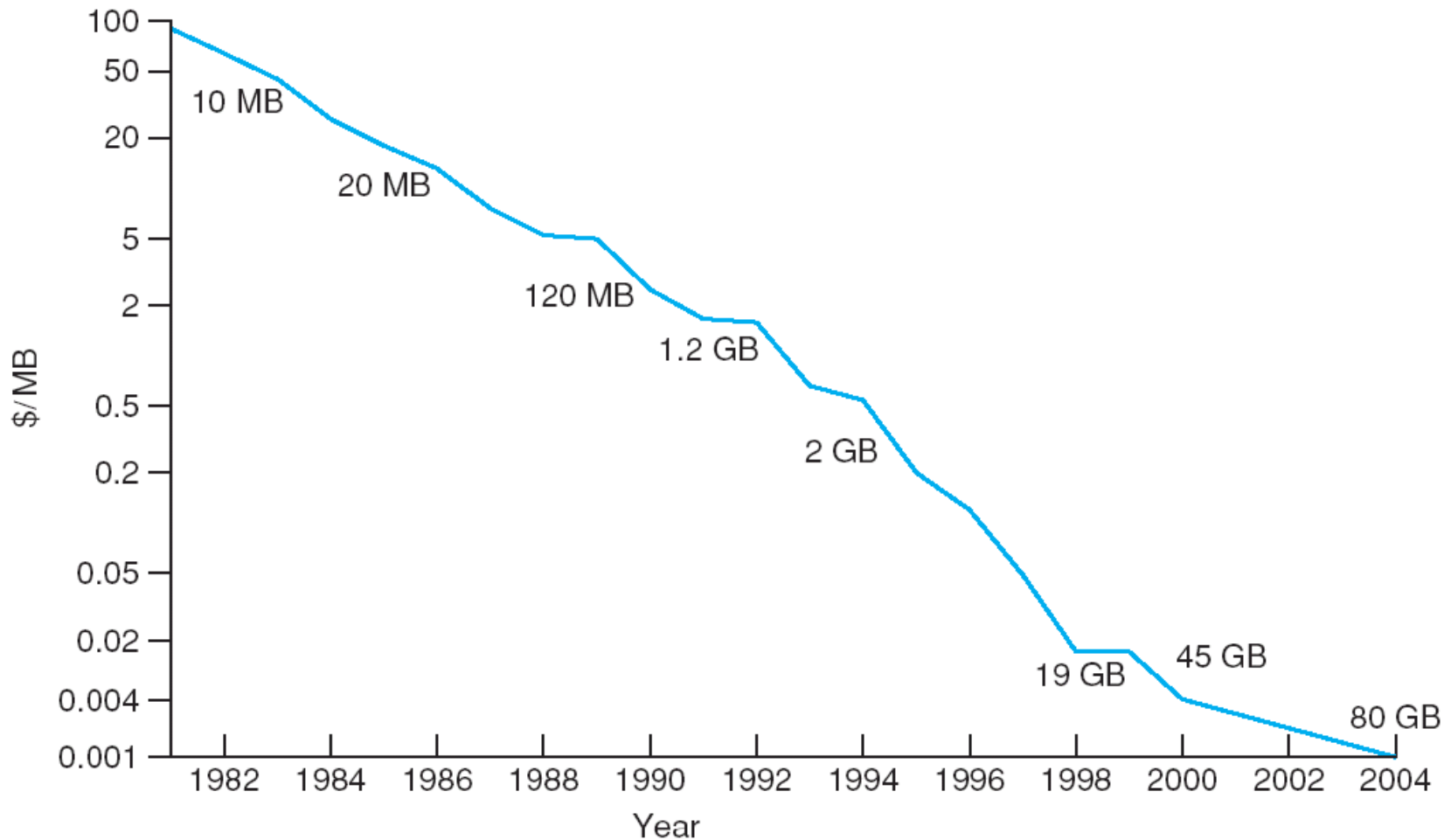
API

- Většina OS pracuje s vyměnitelnými disky (např. disketami, ZIP disky) stejně jako s pevnými disky
 - nový svazek je formátován a na disku se generuje prázdný souborový systém
- Pásy
 - jsou prezentované jako „holé“ (*raw*) paměťové médium
 - aplikace na páskách nemají k dispozici souborový systém
 - otevírají celý páskový mechanismus jako zařízení
 - páskové mechanismy se vesměs nesdílejí, jsou dedikované konkrétní aplikaci
 - OS nepodporují na páskách souborové systémy, aplikace musí samy řešit jak používat bloky dat
 - pásku pak obvykle může používat pouze aplikace, pro kterou byla páska vytvářena (protože jako jediná zná strukturu dat na pásce)

Cena MB RAM (1981 – 2004)



Cena MB pevných disků (1981 – 2004)



Cena MB pásek (1981 -2004)

