Vložené otázky

#242

Otazka:

Alokační blok (Cluster)

Zaraka: Karma: 0 + -

Nejmenší logická část souborového systému. Cluster je vždy 2^n sektorů následujících za sebou.

#108

Otazka:

extent

stanly: Karma: 0 + -

-posloupnost proměnného počtu bloků uložených na disku fyzicky za sebou
-extent udává, kde na disku začíná a kolik bloků obsahuje
-snižuje se počet metadat, zrychluje se čtení velkých souborů
-používají se např v B+ stromech

x: Karma: 0 + -

x

#107

Otazka:

žurnálování, žurnál ...

Zaraka: Karma: 0 + -

Slouží pro záznamy změnu metadat/dat před jejich zápisem na disk.
- používají je např ext3, ext4, ReiserFS, NTFS
- většina dat žurnálována není (příliš velká režie)

#106

Otazka:

start systému:

stanly: Karma: 0 + -

1.BIOS
2.inicializace služeb jádra+jádro samotné
3.swapper
4.init

#105

Otazka:

dynamická změna priorit?

#104

Otazka:

jak se vypíná obsluha přerušení?

#103

Otazka:

při přepínaní kontextu se zálohuje?

? pouze poslední instrukce a registry, nebo vše? celý PCB, pamět ... atd ?

tpa: Karma: 0 + -

registry a pokud to jde, tak na zbytek pouze ukazatele

#102

Otazka:

možná implementace semaforu (je nutné zajistit aby semafor proběhl jako atomická instrukce) využije se vlastnosti spinlocku ...

```
typedef Struct {
int value;
fronta_procesu *queue;
bool lock;
} semaphore;

lock(S){
  while(testAndSet(S.lock)); // spinlock
  S.value--;               //nevim proc
  if(S.hodnota < 0){
     C = get(ready_queue); // vytahnu z fronty procesu pripravenych bezet
     append(S.queue, C);   // dam na forntu cekajicich procesu na semaforu
     S.lock = false;       // vychzim z KS
     switch();             // vzdam se jadra
     }
     else{
     S.lock = false;       // vychazim z KS
     }

// zkuste nekdo napsat jak by se zamykalo se spinlockem ...
```

Z: Karma: 0 + -

```
z přednášky č. 6
bool TestAndSet(bool &target) {
bool rv = target;
```

```
target = true;
return rv;
}
```
Tento kód popisuje chování spinlocku, který je ovšem řešen hardwarově za účelem dosažení atomicity.
S.value--; je tam proto, že při zamčení semaforu se dekrementuje jeho hodnota.

#101

Otazka:

Jak probíhá překlad adres při stránkování?

yes: Karma: 0 + -

zde si myslim, ze by vojnar chtel implementaci v nejakem pseudo kodu... aspon o prednaskach to porad naznacoval, ze bychom to meli zvladnout...

maybe: Karma: 0 + -

```
pg_num page; //cislo stranky
pg_num PageTable; //velikost tabulky
int page_item[PageTable];
fr_num frame;  //cislo ramce

if (page<=PageTable)
{
frame=page_item[base+page];
physic_adress=frame+offset;
}
else SIGSEG;
```

může být něco takového? vůbec nevím, jak si to oni představují...

#100

Otazka:

Napište pseudokód segmentace paměti.

#99

Otazka:

úrovně běhu, co znamenají? které se využívají?

stanly: Karma: 0 + -

```
0..halt
1..single user
6..reboot
2-5..víceuživatelské užití, grafický režim, síťový režim,
grafický+stíťový,.. záleží na distribuci
změna telinit N  (kde N je 0-6 viz výše)
```

Je to ono, na co se ptají? :D

viktor: Karma: 0 + -

ano je to presne ono, plus este urovne 's' a 'S'

#98

Otazka:

Co je hw přerušení, co je řadič přerušení, jak lze zakázat
přerušení, co je NMI, rozdělení na úrovně ...

bla: Karma: 0 + -

HW přerušení je mechanismus, kterým HW zařízení oznamují jádru
asynchronně vznik událostí, které je zapotřebí obsloužit.

Řadič přerušení je řadič, do kterého přicházejí žádosti o HW
přerušení. Na PC se jmenuje APIC, kde každý procesor má vlastní lokální
APIC.

Řadič může být naprogramován tak, aby maskoval určitá přerušení.
Obsluhu přerušení lze také zakázat na procesoru, případně čistě
programově v jádře.

NMI (Non-mascable interrupt) je HW přerušení, které nelze maskovat na
řadiči, ani zakázat jeho přerušení na procesoru.

Obsluha přerušení bývá rozdělena na 2 urovně:
1) Zajišťuje minimální obsluhu HW a plánuje běh obsluhy 2. úrovně.
2) Postupně řeší zaznamenaná přerušení.

#97

Otazka:

Mikrojádra - popis, výhody, nevýhody

nik: Karma: 0 + -

minimalizuju rozsah jadra, jednoduche rozhranie, jednoduche abstrakcie, maly
pocet sluzieb
vacsina sluzieb je implementovana mimo jadro v tzv. serveroch, teda nebezi v
privilegovanom rezime, teda je to bezpecnejsie a flexibilnejsie
nevyhody: nizsia flexibilita?

nik: Karma: 0 + -

*nie flexibilita ale efektivita

Cup: Karma: 0 + -

nevyhoda je vyssi rezije.

jabaduba: Karma: 0 + -

Minimalizují rozsah jádra, jednoduché rozhraní s jednoduchými abstrakcemi a malým počtem služeb = pouze základní správa procesoru, I/O, paměti a meziprocesorové komunikace

výhody:
flexibilita - více současně běžících služeb
zabezpečení - servery neběží v privilegovaném režimu, chyba nevede hned k selhání OS

nevýhody:
vyšší režie - vyšší problém u mikrojader 1.generace, lepší u 2. generace - ale stále přetrvává

katjes: Karma: 0 + -

jabaduba ma ty vyhody a nevyhody spravne

#96

Otazka:

OS - definice, role, cíle atd.

bla: Karma: 0 + -

Operační systém je program (případně soubor programů), které tvoří spojující mezivrstvu mezi HW počítače (který může být virtualizován) a uživatelem (aplikačními programy uživatele).

Cílem OS je maximálně využít zdroje počítače a zjednodušit práci s ním.

OS je správcem prostředků počítače a tvůrcem prostředí pro uživatele a jejich programy (poskytuje standardní rozhraní a abstrakce).

OS se dělí na jádro, systémové knihovny a utility a textové a/nebo grafické uživatelské rozhraní.

#95

Otazka:

Vše, co víte o deadlocku (cca za 15b)

jabaduba: Karma: 0 + -

Vzniká když více procesů chtějí dvě zařízení. A nastane situace že 1. proces si zabere jedno zařízení a 2. zařízení druhé a oba čekají na jejich druhé zařízení které je již zabrané.
řešení: výlučný přístup, postupné přidělování prostředků, odebrání zařízení po určité době

podmínky uváznutí:
vzájemné vyloučení při používání prostředků
vlastnictví alespoň jednoho zdroje a čekání na další
prostředky vraci pouze proces po dokončení jejich využití

cyklická závislost na sebe čekajících procesů

prevence:
-zruší se některá z platnosti podm. uváznutí
1.u prostředků, které umožňují (současný) sdílený přístup, nejsou zámky zapotřebí
2.proces může žádat o prostředky pouze pokud žádné nevlastní
3.pokud proces požádá o prostředky, které nemůže momentálně získat, je pozastaven, všechny prostředky jsou mu odebrány a čeká se, až mu mohou být všechny potřebné prostředky přiděleny
4.prostředky jsou očíslovány a je možné je získávat pouze od nejnižších čísel k vyšším

vyhýbání:
- procesy před spuštěním deklarují určité informace o způsobu, jakým budou využívat zdroje: v nejjednodušším případě se jedná o maximální počet současně požadovaných zdrojů jednotlivých typu.
- předem známé informace o možných požadavcích jednotlivých procesu a o aktuálním stavu přidělování se využijí k rozhodování o tom, které požadavky mohou být uspokojeny (a které musí počkat) tak, aby nevzniklo cyklické cekání.

zotavení a uváznutí:
-ukoncení všech nebo nekterých zablokovaných procesu,
-odebrání zdroje nekterým procesum, anulace jejich nedokoncených operací (rollback) a pozdeji restart.

#94

Otazka:

cituji(z fb, snad dotycne nebude vadit): loni jsem měla otázku NTFS... co by ste k tomu kdo napsali? ono tam toho totiž moc není.

Cup: Karma: 0 + -

Také z fb: Nakreslil MFT, trosku ji popsal a potom nakreslil rozlozeni v te tabulce.

este tam napriklad mozes dodat, ze ma zurnalovanie....

Vyuziva istu modifikaciu B+ stromov, nazyvanych tiez H-stromy. Na rozdiel od Unixu metadata obsahuju aj nazov suboru. Ak sa specifikacia a pristupove prava nevojdu do vyhradneho priestoru, alokuje si dalsie riadky. Kazdy subor ma defaultne "nachystany" jeden riadok v tabulke. Oblast pre data obsahuje adresu zaciatku extendu a to ako aj logicku tak aj fyzicku a velkost. Opat, ak je pocet riadkov adresuje sa priamo, ak nie alokuju sa dalsie riadky na ktore sa odkazuje, podobne ako v i-uzloch.

?+ to, že je to proprietární FS od Microsoftu (docela podstatná informace :D)

hostar: Karma: 0 + -

tady je obrazek:
http://pages.cs.wisc.edu/~bart/537/lecturenotes/figures/mft-entry-extent.gif

---