LUENTO 8

VIRTUAALIMUISTI

Stallings, Luku 8.1

1

Sisältö

- Ohjelman suoritus virtuaalimuistissa
- Sivutus ja sivutaulut
- Osoitemuunnospuskuri TLB

- Segmentointi
- Segmentointi ja sivutus yhdistettynä
- Yhteiskäytöstä

2

Suoritus virtuaalimuistissa

- Ohjelman loogiset osoitteet muutetaan fyysisiksi osoitteiksi vasta ajonaikana
  - prosessin paikka muistissa vaihtelee, sillä ei vaikutusta osoitemuunnokseen
  - MMU
- KJ käsittelee ohjelmaa sivuina tai kääntäjä jakaa ohjelman segmentteihin, jotka KJ voi sijoitella vapaasti muistiin
  - KJ:n kirjanpito osien sijainnista prosessin sivutaulussa tai segmenttitaulussa

3

Suoritus virtuaalimuistissa

- Kaikkien sivujen / segmenttien ei tarvitse olla muistissa yhtäaikaa
  - riittää, että suoritettava osa ja sen data muistissa
  - paikallisuus, suoritus viihtyy samoilla alueilla
  - Kirjan kuva 8.1 näyttää muistiviitteiden jakautumista
- Laitteisto (MMU) ja KJ huolehtivat, että tarvittavat osat oikeaan aikaan muistissa
  - MMU huomaa puuttumisen
  - KJ noutaa muistiin

= Virtuaalimuisti

4

Suoritus virtuaalimuistissa

- Sivu- / segmenttitaulun alkiossa läsnäolobitti, josta käy ilmi onko sivu / segmentti muistissa
- Lataaja tuo aluksi muistiin vain muutaman sivun / segmentin (ennaltanouto) tai ei yhtään (tarvenouto)
- Prosessi CPU:lle ð
  - MMU:hun sivu- / segmenttitaulun
  - fyysinen muistiosoite
  - TLB:n sisällön mitätöinti

Paikallisuuden vaikutus

5

Puutoskeskeytys **(memory / page fault)**

- Jos viitattu osoite ei ole muistissa, MMU aiheuttaa keskeytyksen
- KJ siirtää keskeytyksen aiheuttaneen prosessin (A) Blocked-tilaan
- KJ etsii sivulle / segmentille vapaan paikan muistista
- KJ käynnistää ohjaimen siirtämään puuttuvaa sivua / segmenttia ko. paikkaan
- Siirron aikana CPU suorittaa muita prosesseja

6

Puutoskeskeytys

- Kun siirto valmis, ohjain keskeyttää suorituksessa olevan prosessin B
- KJ päivittää prosessin A sivu/segmenttitaulun ja siirtää prosessin A Ready-tilaan
- Suoritus palaa takaisin prosessiin B
- Kun prosessi A taas aikanaan suoritukseen, se viittaa uudestaan äskeiseen osoitteeseen
  - nyt viitatun mp:n sisältö muistissa

7

Virtuaalimuistin etuja

- Muistia käytetään tehokkaammin hyväksi
  - kustakin prosessista vain tarvittava osa muistissa
  - montako sivua/prosessi pidetään muistissa yhtäaikaa?
    - lokaalit ja globaalit algoritmit
- Prosessoria käytetään (kenties) tehokkaammin
  - moniajoastetta voi nostaa
    - muistiin mahtuu paremmin, ei ruuhkautumista

8

Virtuaalimuistin etuja

- Voi ajaa fyysistä muistia suurempia ohjelmia
  - ohjelmoijan ei tarvitse huolehtia kerrostuksesta
- Osoiteavaruus voi olla valtaisa verrattuna todelliseen muistin määrään
  - esim. 32 bittiä => 4GB:n osoiteavaruus
  - hyöty?
- Looginen osoiteavaruus saa sisältää 'reikiäkin'
  - vain tarvittavat osat kuvataan fyysiseen muistiin

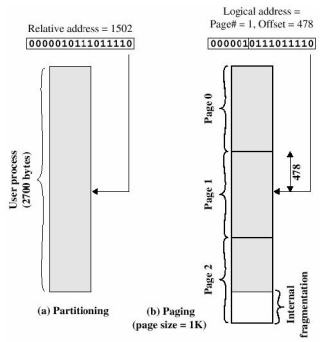
9

SIVUTAULU

10

Sivutus

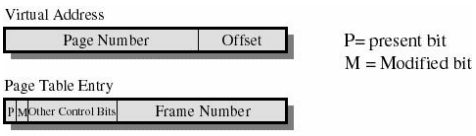
Kuva 7.11



11

Sivutaulu

Kuva 8.2a



- Jokaisella prosessilla oma sivutaulu
  - missä sivutiloissa tämän prosessin sivut sijaitsevat
- Jokaisessa alkiossa läsnäolobitti P
  - P=1: sivu muistissa, alkiossa sivutilan numero
  - P=0: sivu ei muistissa, alkiossa esim. tieto missä sivu sijaitsee tukimuistissa (suoraan/epäsuorasti)

12

Sivutaulu

- Jokaisessa sivutaulun alkiossa muutettu-bitti M (modified)
  - M=1: sivun sisältö muuttunut muistissa, sivu kirjoitettava levylle, jos varaus vapautetaan
  - M=0: sivua ei muutettu, ei tarvitse kirjoittaa levylle sivutilaa vapautettaessa
- Sivutaulun alkiossa mahd. myös muuta tietoa
  - käyttötapabitti: R / RW
  - suojaustasobitti/bitit: KJ:n sivu / tav. prosessin sivu
  - milloin sivuun viitattu viimeeksi tai viitelaskuri
    - poistoalgoritmit tarvitsevat näitä

13

Osoitemuunnos

- Prosessien sivutaulut tavallisesti eri kokoisia ja voivat olla suuria
  - koko riippuu sivukoosta ja ohjelman koosta
  - koko taululle ei voi varata tilaa MMU:sta
    - sivutaulu muistissa ja osa jopa levyllä
    - sivutaulun fyysinen osoite PCB:ssä
- MMU:ssa sivutaulurekisteri PTR, jossa suoritettavan prosessin sivutaulun fyysinen alkuosoite

14

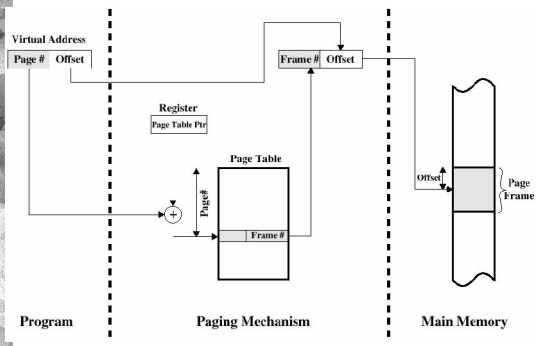
Osoitemuunnos

- MMU jakaa loogisen osoitteen pariksi (sivunro,siirtymä)
  - Esim. kun sivukoko 1024 B (= 2^{10})
    - 10 viimeistä bittiä siirtymä
    - alkuosa sivunumero
- MMU korvaa sivunumero-bitit sivutaulusta löytyvillä sivutilannumero-biteillä
  - ts. MMU katenoi sivutilanumeron ja siirtymän bitit
- Helppo laitetoiminto

15

Osoitemuunnos

Kuva 8.3



16

Lisää sivutauluista

17

Lisää sivutauluista

- Monet järjestelmät sallivat suuren virtuaaliosoiteavaruuden
  - looginen osoite esim. 32 tai 64 bittiä
- Jokaisella prosessilla suuri sivutaulu
  - jos 32-bittinen osoite ja sivukoko 4KB (12 bittiä), niin sivuja 2^{20} = 1M kappaletta
  - jokainen alkio useita tavuja, esim. 4 B, joten sivutaulu 4 MB
- Myös sivutaulu jaetaan sivuihin ja myös sivutaulun osia voidaan pitää levyllä
  - riittää, kun suorituksessa olevaan osaan liittyvät sivut muistissa

18

2-tasoinen sivutaulu

- Ylin hakemisto mahtuu yhteen sivuun, aina muistissa

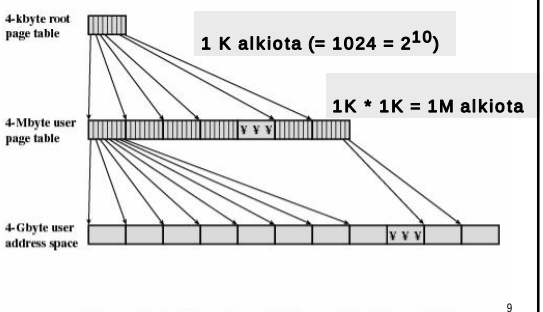
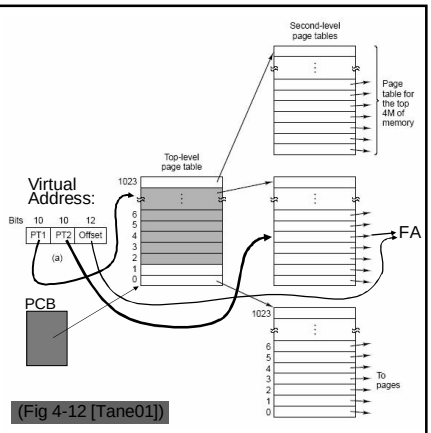


Figure 8.4 A Two-Level Hierarchical Page Table

9

Moni-tasoinen sivu-taulu

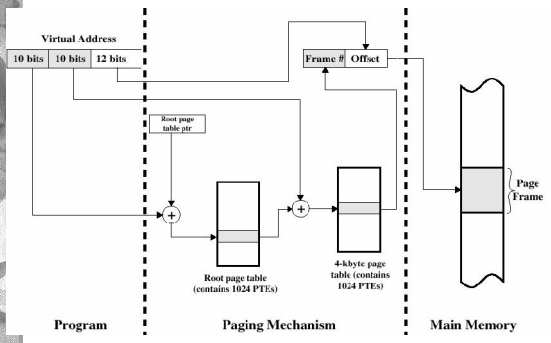


(Fig 4-12 [Tane01])

20

2-tasoinen sivutaulu

Kuva 8.5



21

KÄÄNTEINEN SIVUTAULU

22

Käänteinen sivutaulu

- Fyys.muisti pienempi kuin virtuaaliavaruus
- Kirjaa sivutilakohtaisesti mikä sivu sijaitsee ko. sivutilassa
  - vain yksi globaali käänteinen sivutaulu
  - yksi alkio per fyysinen sivutila, jossa ko. sivutilassa sijaitsevan sivun numero
- Jokaisella prosessilla sivu 0, 1, 2, ...
  miten tiedetään minkä prosessin sivu kyseessä?
  - alkiossa myös prosessin numero (pid)
  - MMU:hun rekisteri, jossa suoritettavan prosessin pid

23

Käänteinen sivutaulu

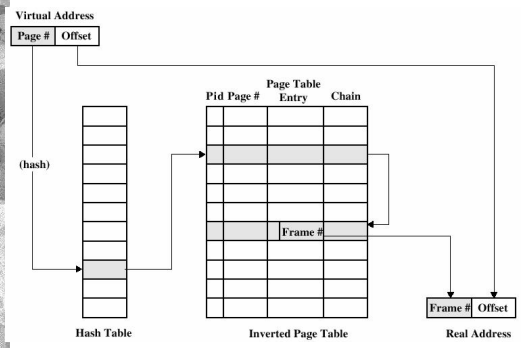
- Käänteisen ST:n indeksit sivutilan numeroita, mutta prosessin osoitteesta käy ilmi
  sivunumero ja siirtymä
  - etsittävä sisällön perusteella
- Etsintä peräkkäishakuna tehotonta
- Käytetään apuna hajautustaulua
  - käytä hajautinta indeksin laskentaan, etsi kotisolusta
  - samaan hajautusosoitteeseen kuvautuvat alkiot linkitetty toisiinsa
- Jos sivun tiedot ei listassa,
  aiheuta sivunpuutoskeskeytys
- Käyttökelpoinen vain jos TLB riittävän suuri

24

Käänteinen sivutaulu

4. ed

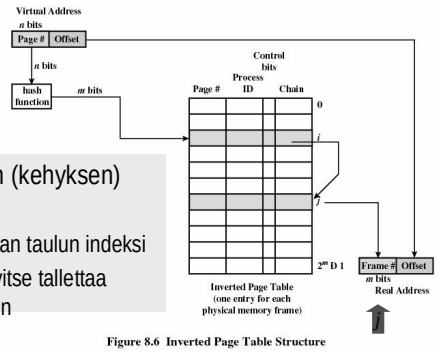
Kuva 8.6



25

Käänteinen sivutaulu v.2

5. ed



- Sivutilan (kehyksen) numero
  - Suoraan taulun indeksi
  - Ei tarvitse tallettaa tauluun

Figure 8.6 Inverted Page Table Structure

26

Osoitemuunnospuskuri TLB

27

Osoitemuunnospuskuri TLB

- Koska sivutaulu muistissa, jokaiseen muunnokseen näyttäisi liittyvän

  yksi ylimääräinen muistinouto
  - hae sivutaulun alkio MMU:hun
  - tee osoitemuunnos
  - nouda / talleta ko. fyysiseen osoitteeseen
- Ratkaisu: osoitemuunnospuskuri TLB
  - Translation Lookaside Buffer
- MMU pitää tallessa edellisissä muunnoksissa tarvitut sivutaulun tiedot
  - paikallisuus: sitä tarvitaan het'kohta uudestaan
  - sivutaulun alkiota ei tarvitse noutaa joka kerta

28

Etsintä TLB:stä

- TLB nopeiden rekistereiden joukko
  - esim. 32 rekisteriä
- Käyttää assosiatiivista hakua
  - vertailu kohdistuu kaikkiin alkioihin yhtäaikaa
  - etsintä nopeaa
  - laitetoiminto!
- Kaikilla prosesseilla sivut 0, 1, 2, …,

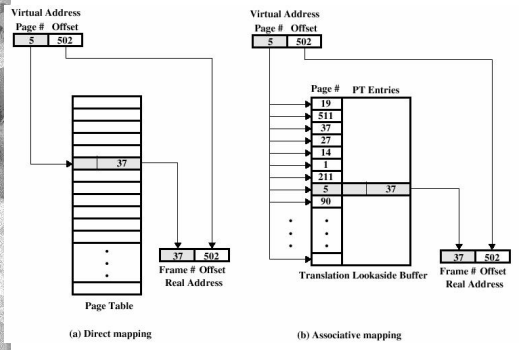
  mutta eri sivutiloissa
  - è TLB tyhjennettävä prosessin vaihdossa
- TLB:n alkiossa (ainakin)

  sivunro, sivutaulun alkio, validibitti V

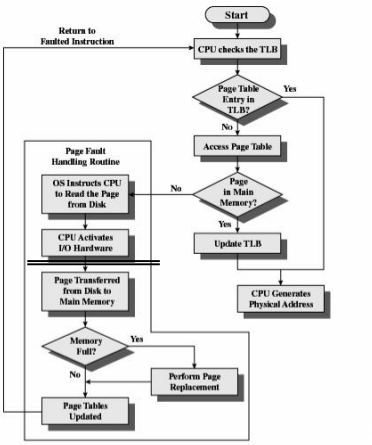
29

Etsintä TLB:stä

Kuva 8.9



30



Kuva 8.8.

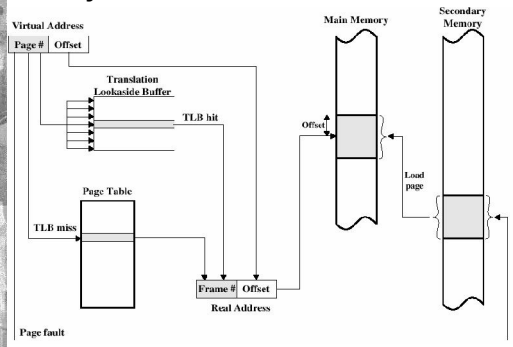
TLB ja osoitemuunnos

```
sivunro  = loog. osoitteen alkubitit
siirtymä = loog. osoitteen loppubitit
Jos sivun tiedot ei TLB:ssä tai V=0,
   nouda TLB:hen sivutaulun alkio
   osoitteesta PTR + sivunro
Jos P=0, aiheuta sivunpuutoskeskeytys
Fyys.os = Katenoi(Sivutilanro,Siirtymä)
```

- Kun keskeytys käsitelty, sama osoite tulee viitattavaksi uudelleen
  - Esim. PC:n kasvatus vasta osoitemuunnoksen jälkeen

32

TLB ja osoitemuunnos Kuva 8.7



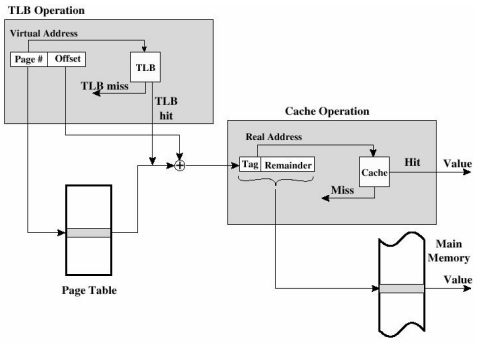
33

TLB:n alustus

- Kun suoritettava prosessi vaihtuu, TLB:n vanha sisältö mitätöitävä
  - PTR osoittamaan uuden prosessin sivutauluun
  - nollattava TLB:n alkioiden validibitit V=0
- Koska TLB suhteellisen pieni, tarvitaan sopiva laitetason algoritmi, jonka perusteella valitaan korvattava alkio
  - TLB:n alkiossa mahd. myös laitetason viitelaskureita: poista se, jota ei ole aikoihin käytetty

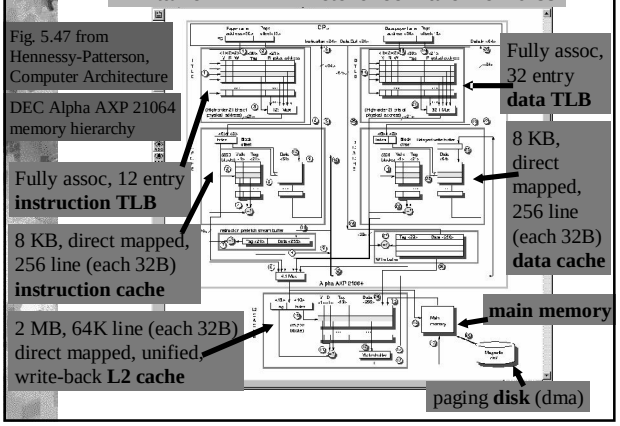
34

TLB ja välimuisti Kuva 8.10



35

TLB tarkemmin? ⟶ Tietokoneen rakenne -kurssi

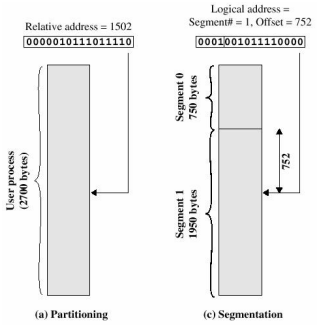


SEGMENTOINTI

37

Segmentointi

Kuva 7.11



38

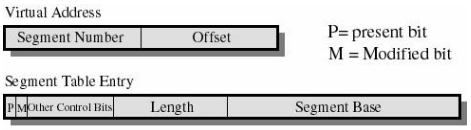
Segmentointi

- Perusideoiltaan samanlainen kuin sivutus, 'yksiköt' vain keskenään erikokoisia segmenttejä
- Jokaisella prosessilla oma segmenttitaulu
  - Kertoo missä tämän prosessin segmentit sijaitsevat
- Osoitemuunnos segmenttitaulun avulla
- TLB:n käyttö kuten edellä
- MMU:n rekisterissä nyt segmenttitaulun fyysinen osoite

39

Segmenttitaulu

Kuva 8.2b

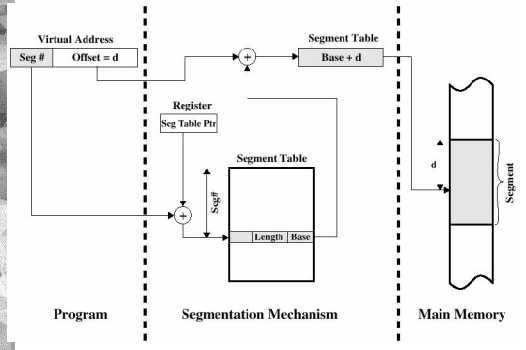


- Jokaisessa alkiossa läsnäolobitti P
  - P=1: segmentti muistissa
- alkiossa fyysinen alkuosoite sekä pituus
- Jokaisessa alkiossa muutettu-bitti M kuten sivutaulussa
- Jokaisessa alkiossa myös muuta tietoa kuten sivutaulussa

40

Osoitemuunnos

Kuva 8.12



41

Huomioita

- Segmenttitaulun alkiossa alkuosoite ja pituus
  - segmentin kokoa helppo kasvattaa/pienentää dynaamisesti
    - saattaa vaatia segmentin uudelleensijoittamista
  - osoitteen oikeellisuus tarkistettavissa MMU:ssa
- Segmentit erikokoisia, syntyy ulkoista pirstoutumista
  - Varaus/vapautus ei niin tehokasta kuin sivuilla
  - Muistin tiivistämistarvetta
- Segmentti sopiva suojauksen yksikkö
  - ohjelmoija määrittelee segmentit ja käyttöoikeudet
  - käyttötapa kopioitu segmenttitaulun alkioon

42

Segmentointi ja sivutus yhdistettynä

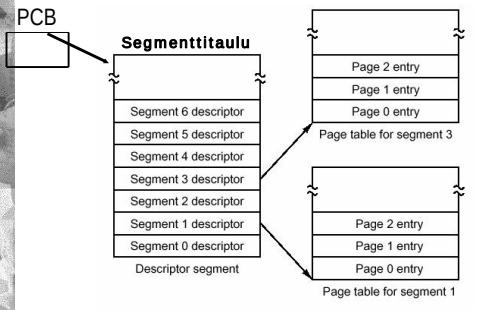
43

Sivuttava segmentointi

- Eräät laitteistot ja KJ:t jakavat myös segmenttejä sivuiksi, ohessa eräs tapa
  - muistia helpompi hallita samankokoisina sivuina
  - ei ulkoista pirstoutumista
  - ei tiivistämistarvetta
- Jokaisella prosessilla
  - oma segmenttitaulu ja
  - yksi sivutaulu per segmentti
- Vrt. 2-tasoinen sivutaulu

44

Sivuttava segmentointi Tan01 4-39



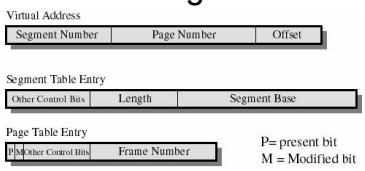
45

Sivuttava segmentointi

- Looginen osoite jakautuu nyt kolmeen osaan
  - segmenttinumero
    - segmentin sivutaulun fyysinen osoite löytyy segmenttitaulun tästä alkiosta
  - sivunumero
    - sivunumeroa vastaava sivutilan numero löytyy sivutaulun tästä alkiosta
  - siirtymä
    - sivulla viitattu sana näin kaukana sivun alusta
- Myös segmenttitaulua / sivutaulua voidaan sivuttaa

46

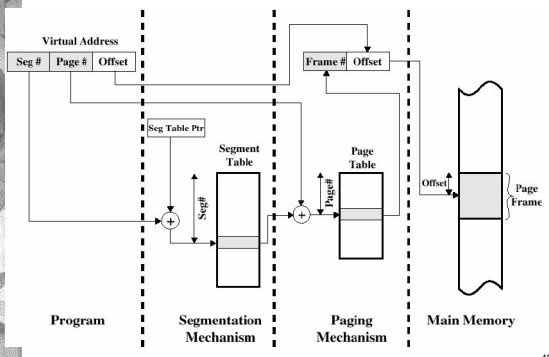
Sivuttava segmentointi Kuva 8.2c



- Segmentin alkuosoitteen (segment base) paikalla ko. segmentin sivutaulun fyysinen osoite
- P-bitti ja M-bitti vain sivutaulun alkiossa
- Yhteiskäytön ja suojauksen data luonnollisimmin segmenttitaulun alkiossa
  - oikeudet annettu koko segmenttiin (sen kaikkiin sivuihin)

47

Osoitemuunnos Kuva 8.13



48

Hyötyjä

- Ratkaisee dynaamisen linkittämisen ongelmakohdat
  - uuden segmentin (dynaaminen) linkittäminen tarkoittaa vain uuden alkion lisäämistä segmenttitauluun
- Segmentin koko voi kasvaa sivu kerrallaan, eikä segmentille tarvitse etsiä uutta paikkaa fyysisessä muistissa
- Yhteiskäyttö ja käyttöoikeudet voi määrittää segmenttikohtaisiksi
  - kauniisti loogisten kokonaisuuksien mukaan
  - useita erilaisia suojaustasoja

49

Yhteiskäyttö ja suojaus

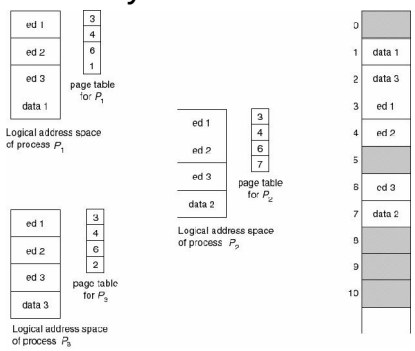
50

Yhteiskäyttö

- Jos usea prosessi ajaa samaa koodia, riittää siitä yksi kopio muistissa
  - jokaisen prosessin sivu / segmenttitaulussa viitteet yhteisen koodin ja datan sisältäviin sivutiloihin / segmentteihin
  - mutta kullakin prosessilla omat tilat yksityiselle datalle
- Koodin oltava vapaakäyntistä
  - ei saa muuttua
- Sivutus ei paras tapa yhteiskäyttöä ajatellen:
  - sivu 'kokoyksikkö', ei looginen yksikkö
  - käyttöoikeudet vaikea rajata funktiotasolla

51

Yhteiskäyttö: editori



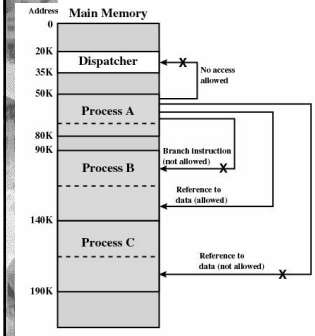
52

Segmentointi ja yhteiskäyttö

- Segmentointi kaunis tapa yhteiskäyttöä ajatellen
  - ohjelman jakaminen eripituisiin segmentteihin loogisempaa kuin jakaminen tasapitkiksi sivuiksi
  - esim. yhteiskäyttöön tarkoitettu data omaksi segmentiksi, yksityinen data omaksi segmentiksi
- Ohjelmoija kertoo kääntäjälle
  - haluamastaan segmenttijaosta
  - haluamistaan käyttöoikeuksista
- Kääntäjä muodostaa tällä perusteella ohjelman loogiset osoitteet
  - segmentti ja siirtymä sen sisällä

53

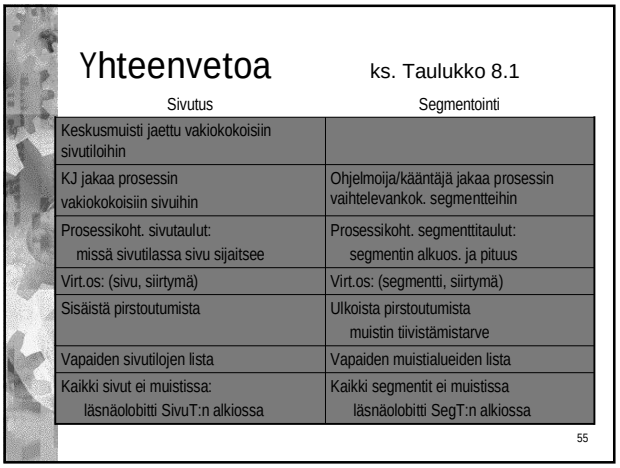
Erilaisia suojausryhmiä kuva 8.14



- Mahdollinen luokittelu:
- Ei saa käyttää lainkaan
- Saa käyttää vain dataa
- Saa käyttää vain koodia (???)
- Saa käyttää sekä koodia että dataa

Figure 8.14 Protection Relationships Between Segments

54


Yhteenvetoa
ks. Taulukko 8.1
Sivutus
Segmentointi
Keskusmuisti jaettu vakiokokoisiin sivutiloihin
KJ jakaa prosessin vakiokokoisiin sivuihin
Ohjelmoija/kääntäjä jakaa prosessin vaihtelevankok. segmentteihin
Prosessikoht. sivutaulut: missä sivutilassa sivu sijaitsee
Prosessikoht. segmenttitaulut: segmentin alkuos. ja pituus
Virt.os: (sivu, siirtymä)
Virt.os: (segmentti, siirtymä)
Sisäistä pirstoutumista
Ulkoista pirstoutumista muistin tiivistämistarve
Vapaiden sivutilojen lista
Vapaiden muistialueiden lista
Kaikki sivut ei muistissa: läsnäolobitti SivuT:n alkiossa
Kaikki segmentit ei muistissa läsnäolobitti SegT:n alkiossa
55