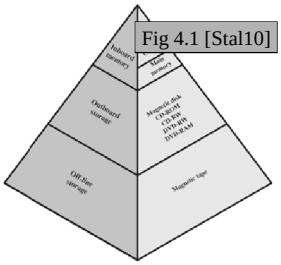
Luento 8 (verkkoluento 9)
Järjestelmän ulkoinen muisti I/O

Muistihierarkia
Kiintolevyt
I/O:n toteutus

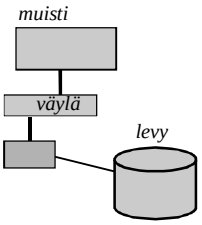
Muistihierarkia

- Ulkoinen muisti (levymuisti) on halvempaa toteuttaa per t
- Ulkoinen muisti on paljo hitaampaa kuin sisäinen r
- Aika/tila optimointi
  - suuret tietomäärät täytyy (kannattaa) kustannussyist ulkoisessa muistissa
  - pienet tietomäärät täytyy (kannattaa) tehokkuussyist sisäisessä muistissa
- Kaiken viitatun tiedot tule suoritusaikana olla sisäise muistissa!



Virtuaalimuisti

- Osa muistihierarkiaa
- Vastaus ongelmaan
  - ”yhtä suuri” kuin levymuisti ja ”yhtä nopea” kuin keskusmuisti?
- Kaksitasoinen:
  - keskusmuistissa kulloinkin käytössä olevat alueet
  - levyllä kaikki tiedot
  - kopiointi tarvittaessa
    - sivunpuutoskeskeytys



Virtuaalimuistin toteutus

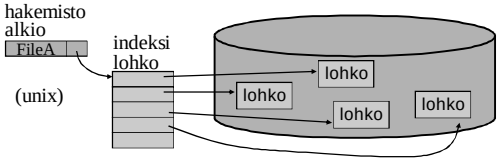
- Toteutustavat
  - kanta- ja rajarekisterit
  - sivutus, (segmentointi ja sivuttava segmentointi)
- Pääosa toteutuksesta KJ:n ohjelmistotasolla
- Laitteistotuki
  - MMU – muistinhallintayksikkö
    - Virtuaaliosoite -> fyysinen muistiosoite
  - Osoitteenmuutospuskurista (TLB) löytyy viimeksi tehdyt osoitteenmuutokset (vrt. välimuisti)
    - osoitetta ei tarvitse laskea usealla konekäskyllä, jos se löytyy TLB:stä

Tiedostojärjestelmä

- KJ:n osa, hallitsee kaikkia tiedostoja
- Valvoo oikeuksia tiedostoa avattaessa
- Muuntaa tiedostonimet fyysisiksi osoitteiksi
- Ylläpitää taulukoita, joista näkee mitä kohtaa mistäkin tiedostosta kukin prosessi on käsittelemässä
- Tiedostojärjestelmä lukee ja kirjoittaa tiedostoja suurina kerralla käsiteltävinä lohkoina (0.5-8 KB?)
  - käyttäjätason prosessit käsittelevät tiedostoja tavuittain eikä niiden tarvitse tietää tiedoston todellista fyysistä rakennetta (KJ:n laiteajuri huolehtii siitä)

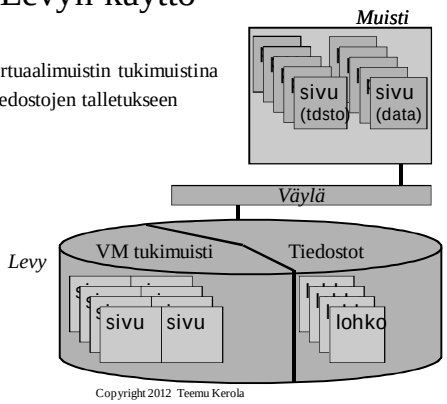
Tiedoston talletus levylle

- Tiedosto koostuu useista lohkoista
  - lohko = 1 tai usea levyn sektori
- Levyn hakemisto
  - tiedoston lohkot
  - luetaan lohkot annetussa järjestyksessä



Levyn käyttö

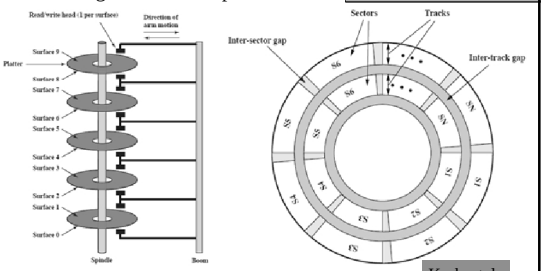
- Virtuaalimuistin tukimuistina
- Tiedostojen talletukseen



Levymuistin saantiaika

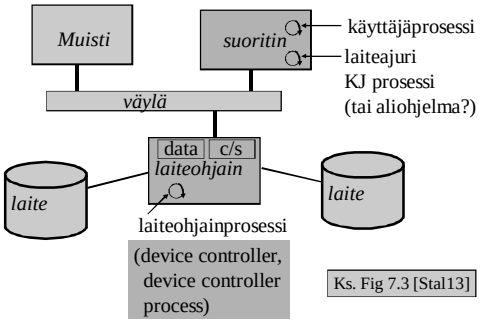
- Tiedon (lohkon) osoite: levypinta + ura + sektori
  - laiteajuri etsii KJ-taulukoista loogisen osoitteen perusteella

Figs 6.5&6.2 [Stal13]



 Keskustele

Laiteohjain (I/O Moduuli)



(device controller, device controller process)

Ks. Fig 7.3 [Stal13]

Laiteohjaimen rekistereihin viittaaminen omilla konekäskyillä

- Käskyssä annetaan laiteohjaimen identifikaatio ja laiterekisterin nro (oma I/O osoiteavaruus)
- Vaikea laajentaa käyttöä uusiin laitteisiin, joilla ”laiterekisterit” voivat olla hyvinkin erilaisia
- Suorittimen konekäskyjä ei voi muuttaa

x86: IN, OUT
INS, OUTS

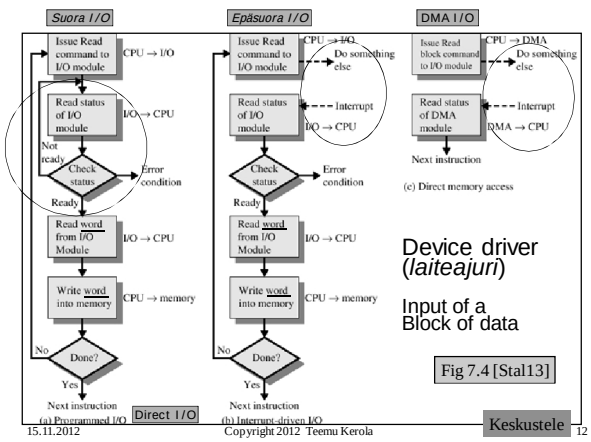
Ttk-91:
IN, OUT

Muistiinkuvattu I/O

- Laiteajuri lukee/kirjoittaa laiteohjaimella olevia rekistereitä (data, status/kontrolli) tavallisilla muistin luku/kirjoitus käskyillä
  - ei tarvita erillisiä I/O-konekäskyjä!
  - laiteohjaimella olevat ”laiterekisterit” ovat samanlaista viitattavaa muistia kuin ”normaali muisti”
  - muistisoitteen ensimmäiset bitit (ei siis käskykoodi) valitsevat, mille laitteelle (vai tavallisen muistiin) viittaus kohdistuu

```
load  R1,=DiskRd
store R2, DiskCtr
```

```
DiskCtr EQU  0x80000001
```

Device driver (*laiteajuri*)

Input of a Block of data

Fig 7.4 [Stal13]

Direct I/O

 Keskustele

Esimerkki: kirjoittimen laiteajuri ttk-91 koneelle

- Laitteella voi tulostaa kokonaislukuja yksi kerrallaan
- Muistiinkuvattu I/O, suora I/O
- Laiteportti
  - kontrollirekisteri muistipaikka 1048576 = 0x80000
  - tilarekisteri muistipaikka 1048577 = 0x80001
  - datarekisteri muistipaikka 1048578 = 0x80002
- Laiteajuri Print toimii etuoikeutetussa tilassa
- Kutsu:

```
PUSH  SP, =0       ; space for return value
PUSH  SP, X        ; parameter to print
SVC   SP, =Print   ; returns Success/Failure
POP   SP, R1
JNZER R1, TakeCareOfTrouble
```

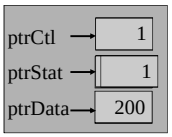
15.11.2012 Copyright 2012 Teemu Kerola 13

Esim: laiteajurin toteutus

```
ptrCtr   DC  1048576 ; control register address
ptrStat  DC  1048577 ; status register address
ptrData  DC  1048578
retVal   EQU -3
parData  EQU -2
```

Solution with no timeout

Oleta: SVC:n ja IRET:n toteutus samalla tavalla kuin CALL ja EXIT



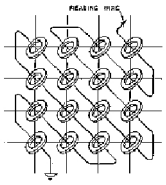
See: driver.k91

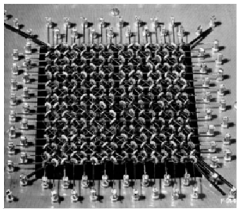
```
Print   PUSHR SP              ;save regs
        LOAD  R1, parData(FP)
        STORE R1, @ptrData ; data to print
        LOAD  R1, =0
        STORE R1, @ptrStat  ; init (clear) state register
        LOAD  R1, =1
        STORE R1, @ptrCtr  ; give command to print

Wait    LOAD  R1, @PtrStat ; check state register
        JNZER R1, Done
        JUMP  Wait          ; wait until I/O done

Done    LOAD  R1, =0        ; return "Success"
        STORE R1, retVal(FP)
        POPR  SP            ; recover regs
        IRET  SP, =1
```

15.11.2012 Copyright 2012 Teemu Kerola 14

-- Luennon loppu --

- Ferriittirengas (core) teknologia
  - 1952, Jay Forrester & Bob Everett, MIT (Whirlwind)
  - tieto säilyy ilman virtaa
  - ei häiriinny säteilystä (avaruus, sotilasteknologia)
  - 1955, valtaa markkinat Williams Tube'lta
  - Käytössä vielä 1970-luvulla, nyt vain nimi jäljellä





15.11.2012 Copyright 2012 Teemu Kerola 15