Luento 12 Yhteenveto

Keskeiset asiat
Mitä hyötyä tästä on?
Mitä seuraavaksi?
Kurssit?
Asiat?

Tavoitteet

- Ymmärtää tietokonejärjestelmän keskeiset piirteet sillä suoritettavan ohjelman näkökulmasta
- Miten tietokonejärjestelmä suorittaa sille annettua ohjelmaa?
- Minkälaista koodia suoritin ymmärtää?
- Mikä on käyttöjärjestelmän rooli?

Mitä hyötyä tästä on?

- Ohjelman suoritusnopeus perustuu suorittimen (CPU) suorittamiin konekäskyihin, ei pelkästään ohjelman korkean tason esitysmuotoon
- Ylemmän tason asioiden ymmärtäminen on helpompaa (mahdollista), kun ymmärtää alemman tason asiat

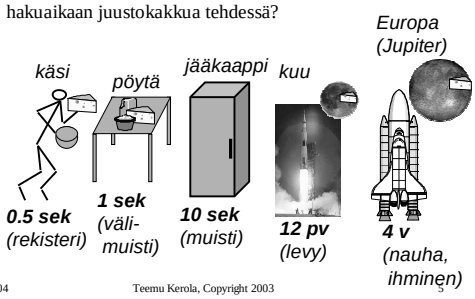
Keskeisiä asioita

- Järjestelmä kokonaisuudessaan, nopeuserot
- Esimerkkikone ja sen käyttö
- Konekielinen ohjelmointi
- Suoritin, rekisterit, väylät, muisti
  - konekäskyjen suoritussykli, keskeytykset
- Aktivointitietuepino, aliohjelmien toteutus
- Tiedon esitysmuodot (ohjelma vs. laitteisto)
- Prosessi ja sen toteutus (PCB)
- I/O laitteet
  - laiteajurit, laitekeskeytykset, levymuisti
- Ohjelmien suoritus järjestelmässä
  - käännös, linkitys, lataus, tulkinta, emulointi, simulointi

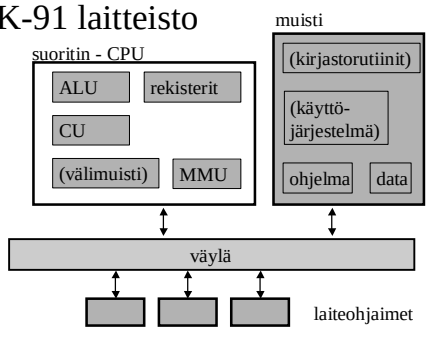
Esimerkkejä keskeisistä asioista seuraavilla kalvoilla

Nopeuserot: Teemun juustokakku

Rekisterien, välimuistin, muistin, levymuistin ja magneettinauhan nopeudet suhteutettuna juuston hakuaikaan juustokakkua tehdessä?



Esimerkkikone: TTK-91 laitteisto



TTK-91

- Rekisterit
  - Yleisrekisterit R0-R5,
  - SP – Stack Pointer, pino-osoitin
  - FP – Frame Pointer, ympäristöosoitin
- Kontrolliyksikön sisäiset rekisterit
  - PC - Program Counter, käskyosoitin
  - IR - Instruction Register, käskyrekisteri
  - TR - Temporary Register, apurekisteri
  - SR - State Register, tilarekisteri

Konekielinen ohjelmointi

```
I     DC    0
      …
      LOAD  R1, =20
      STORE R1, I

      for (int i=20; i < 50; ++i)
          T[i] = 0;

Loop  LOAD  R2, =0
      LOAD  R1, I
      STORE R2, T(R1)

      LOAD  R1, I
      ADD   R1, =1
      STORE R1, I

      LOAD  R3, I
      COMP  R3, =50
      JLES  Loop
```

muuttujat, vakiot taulukot, tietueet

muistissa, rekisterissä?

valinta, toistot aliohjelmat, SVC:t parametrit, paikalliset muuttujat

TTK-91 muistin osoitusmoodit (8)

ks. lista sivulla 50 [Häkk98]

LOAD R1, 10	; R1 ← 200	
LOAD R1, =10	; R1 ← 10	
LOAD R1, @10	; R1 ← 6000	
LOAD R4, R2	; R4 ← 201	
LOAD R4, @R2	; R4 ← 11	
LOAD R5,=Tbl(R3)	; R5 ← 201	
LOAD R5, Tbl(R3)	; R5 ← 11	
LOAD R5, @Tbl(R3)	; R5 ← 300	

rekisterit

R0:	104
R1:	10
R2:	201
R3:	1
…	
SP=R6:	
FP=R7:	125

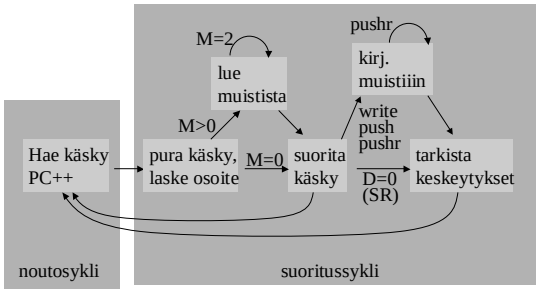
muisti-segmentti

0:	
10:	200
11:	300
200:	6000
201:	11
LIMIT:	

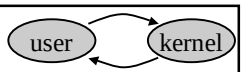
symboli-taulu

Tbl:	200
X:	10
One:	1

Käskyjen nouto- ja suoritussykli



Suorittimen tilat



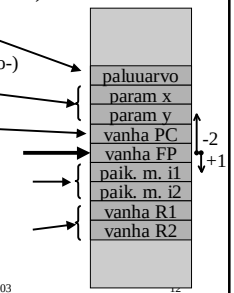
- Käyttäjätila (user mode, normal mode)
  - voi käyttää vain tavallisia käskyjä
  - voi viitata vain käyttäjän omaan muistiavaruuteen (MMU valvoo)
- Etuoikeutettu tila tai (KJ:n) ytimen tila (kernel mode, privileged mode)
  - voi käyttää kaikkia käskyjä, myös etuoikeutettuja (esim. clear_cache, iret)
  - voi viitata kaikkialle muistiin, myös käyttöjärjestelmän ytimeen (kernel)
    - voi käyttää (myös) suoria muistiosoitteita (PA)

Aktivointitietue (Aktivointitietuepino)

(activation record, activation frame)

int funcA (int x,y);

- Aliohjelman toteutusmuoto (ttk-91)
  - funktion paluuarvo (tai kaikki paluuarvot)
  - kaikkien (sisäänmeno- ja ulostulo-) parametrien arvot
  - paluuosoite
  - kutsukohdan aktivointitietue
  - kaikki paikalliset muuttujat ja tietorakenteet
  - aliohjelman ajaksi talletettujen rekistereiden alkuperäiset arvot



Aliohjelmaesimerkki (ei anim)

```
int fA (int x, y)
{
        int z = 5;
        z = x * z + y;
        return (z);
}
…
T = fA (200, R);
```

käyttö:

```
R       DC  24
…
PUSH  SP,=0 ; ret. value space
PUSH SP, =200   muistista muistiin!!
PUSH  SP, R
CALL  SP, fA    talleta PC, FP aseta PC, kutsu & paluu palauta FP, PC
POP     SP, R1  2. operandi aina rekisteri
STORE R1, T
…
```


Aliohjelma-esimerkki

```
int fA (int x, y)
{
        int z = 5;
        z = x * z + y;
        return (z);
}
…
T = fA (200, R);
```

```
retfA   EQU  -4
parX    EQU  -3
parY    EQU  -2
locZ    EQU  1
fA      PUSH  SP, =0 ; alloc Z
        PUSH  SP, R1 ; save R1          prolog
        LOAD  R1,=5; init Z
        STORE R1, locZ (FP)
        LOAD R1, parX (FP)
        MUL   R1, locZ (FP)
        ADD   R1, parY (FP)
        STORE R1, locZ (FP)
        STORE R1, retfA (FP)
        POP     SP, R1; recover R1      epilog
        SUB     SP, =1 ; free Z
        EXIT    SP, =2 ; 2 param.
```


Tiedon esitysmuodot

kokonaisluvut

liukuluvut

merkit

merkkijonot

kuvat

äänet

ei-standardoitu tieto

suorittimen ymmärtämä tieto

Lukujärjestelmiä:

Binääriluvut, oktaaliluvut,

heksadesimaaliluvut

Negatiiviset luvut (4)

arvo talletus

+57 = 0011 1001

- Etumerkkibitti erikseen
- Yhden komplementtiesitys
- Kahden komplementtiesitys
- Vakiolisäys
  - Esim lisää 127 (=2^8 –1)
    - yleensä: $2^{bittilkm}$ -1
  - Talleta etumerkittömänä

sign bit = MSB = most significant bit

luku -57 = 1011 1001 talletus-muoto

-57 = 1100 0110

"sign" bit +1

-57 = 1100 0111

"sign" bit

-57 = 0100 0110

-57 + 127 =70

+57 = 1011 1000

+57+127=184

Tiedon esitysmuodot: liukuluku

1/8 = 0.1250

1/16= 0.0625

0.1875

"+"	"15"	"0.1875" = "0.0011"
sign	exponent	mantissa or significand

- 23 bittiä mantissalle, siten että ...

1) Binääripiste (.) on heti ensimmäisen bitin jälkeen

2) Mantissa on normalisoitu: vasemmanpuolimmainen bitti on1

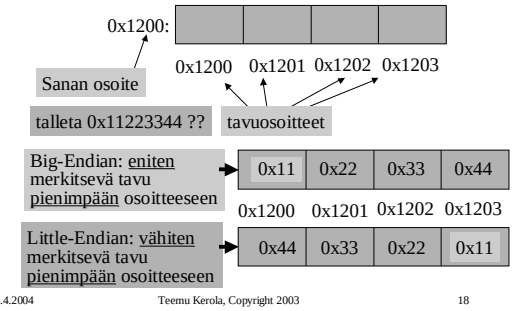
3)Vasemmanpuolimmaista (eniten merkitsevä) bittiä (1) ei talleteta (implied bit)

mantissa	eksponentti
0.0011	"15"
1.1000	"12"
1000	"12"

24 bitin mantissa!

Big vs. Little Endian (3)

- Miten monitavuiset arvot talletetaan?



Virheen korjaava Hamming koodi

Data: 100 1100 110 1100 (parillinen pariteetti)

Bitti nro: 765 4321 765 4321

1 = 001
2 = 010
3 = 011
4 = 100
5 = 101
6 = 110
7 = 111

Pariteettibitti 1 tarkistaa bittejä 1, 3, 5, 7

Pariteettibitti 2 tarkistaa bittejä 2, 3, 6, 7

Pariteettibitti 4 tarkistaa bittejä 4, 5, 6, 7

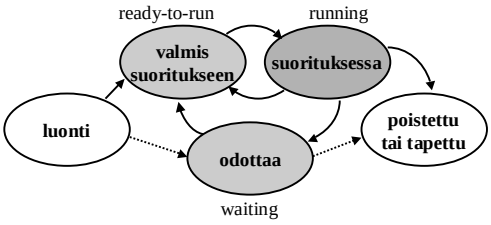
Tapahtuu virhe: bitti 6 muuttuu (flips)

Pariteettibitti 2 tarkistaa bittejä 2, 3, 6, 7: VIRHE

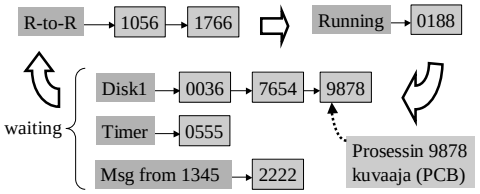
Pariteettibitti 4 tarkistaa bittejä 4, 5, 6, 7: VIRHE

2+4 = 6 ⇒ korjaa bitti nro 6

Prosessin tilat ja elinkaari

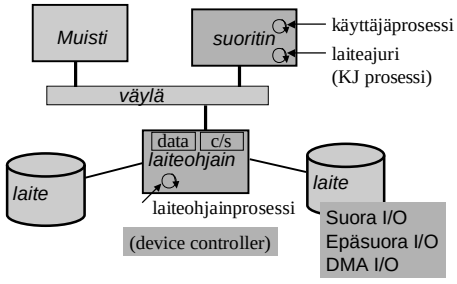


Prosessit jonoissa ja PCB



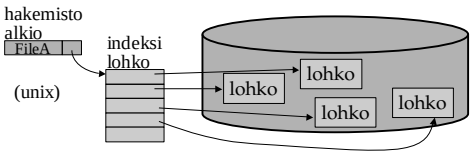
Vuoronanto:
valitse seuraava prosessi Ready-to-Run -jonosta ja
siirrä se suoritukseen CPU:lle
(kopioi tämän prosessin suorittimen tila suorittimelle)

Laiteohjain ja laiteajuri

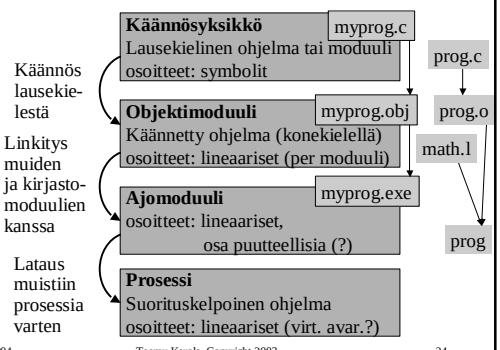


Levyjen käyttö

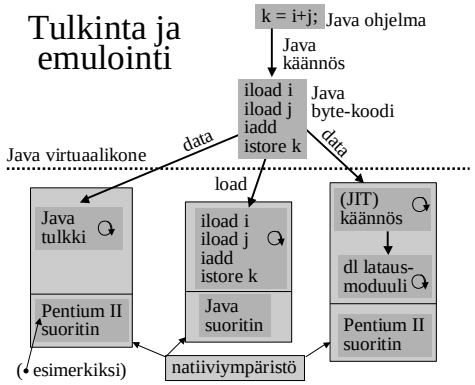
- Tiedosto koostuu useista lohkoista
  - lohko per sektori
- Levyn hakemistossa on tieto kunkin tiedoston käyttämistä lohkoista
  - luetaan lohkot annetussa järjestyksessä



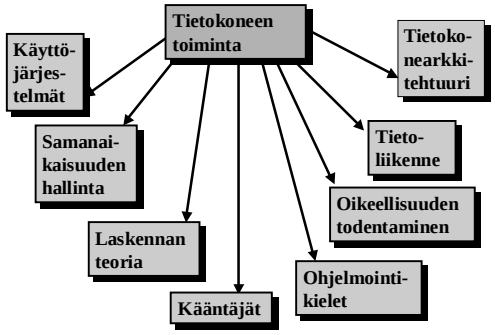
Lausekielestä suoritukseen



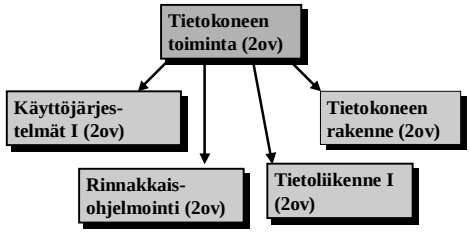
Tulkinta ja emulointi



Asioiden välisiä suhteita



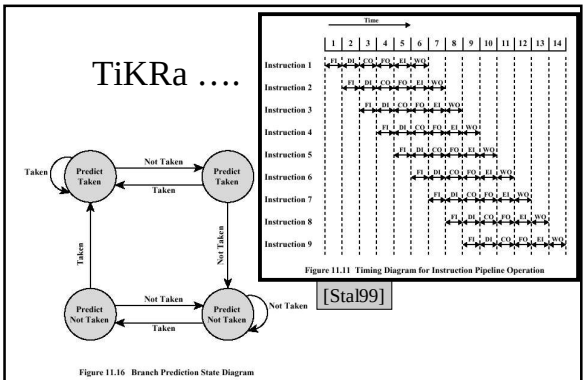
Kurssien välisiä suhteita



Tietokoneen rakenne, 2 ov

- Yksi taso alaspäin TITOsta
- Sopiva 2. vuoden opiskelijalle
- Useissa yliopistoissa yhdistetty TITOon
- ”Miten kellopulssi saa suorittimen suorittamaan konekäskyjä?”
- ”Miten suorittimen aritmetiikka on toteutettu?”
- Usea käsky on todellisuudessa suorituksessa samanaikaisesti
  - Miten tämä toteutetaan, mitä ongelmia siitä seuraa ja miten noita ongelmia ratkotaan?
- Jatkoa syventävällä tasolla
  - Tietokonearkkitehtuurit, 4 ov

TiKRa ….



[Stal99]

Käyttöjärjestelmät I, 2 ov

- Sopiva 2. vuoden opiskelijalle
- Käyttöjärjestelmän rooli yhden prosessin valvojana
- Täsmentää ja jatkaa TITOn käyttöjärjestelmien piirteiden esittelyä
- Samanaikaiset prosessit resurssien käyttäjinä
- Systeemin resurssien jakelu
- Prosessien vuoronanto (skedulointi)
- Jatkoa perustasolla ja syventävällä tasolla
  - Käyttöjärjestelmät II, 2 ov
  - Käyttöjärjestelmämetodiikka, 3 ov

KJ ...

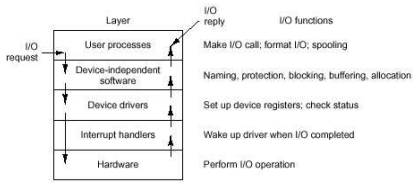
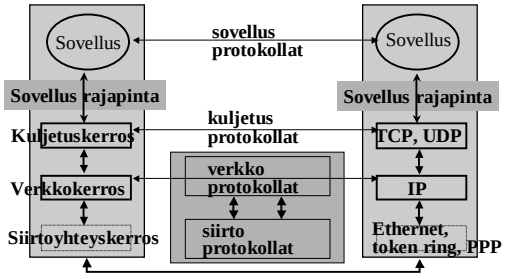


Figure 3-6. Layers of the I/O system and the main functions of each layer.

Tietoliikenne I, 2 ov

- Sopiva 2. vuoden opiskelijalle
- Tietokoneverkkojen peruspalvelut käyttäjälle ja sovelluksille
- Verkkojen tiedonsiirron perusvälineistö
- Verkkoarkkitehtuurin kerrosrakenne ja kunkin tason palvelut
- Jatkoa perustasolla ja syventävällä tasolla
  - Tietoliikenne II, 2 ov
  - Tietoliikennejärjestelmät, 3 ov

Tietoliikenne … TCP/IP -kerrosmalli



Rinnakkaisohjelmointi, 2 ov

- Sopiva: 2. vuoden opiskelijoille
- Samanaikaisuuden aiheuttamat ongelmat
  - järjestelmä kaatuu … miksi niin kävi?
- Samanaikaisuuden aiheuttamat vaatimukset systeemille
- Prosessien synkronointi eri tapauksissa
  - odottamalla vai prosessia vaihtamalla? miksi?
- Prosessien kommunikointi eri tavoin
  - yhteinen muistialue? viestit? miksi?
  - verkon ylitse?
- Jatkoa syventävällä tasolla
  - Hajautetut järjestelmät, 3 ov

RIO: synkronointiongelman ratkaisu Test-and-Set -käskyllä

- TAS Ri, L (ttk-91:n laajennus)

tämän käskyn osoite

```
Ri := mem[L]
if Ri==1 then
  {Ri := 0, mem[L] := Ri, jump *+2}
```

- Kriittinen vaihe

```
LOOP: TAS   R1, L    # L: 1 (vapaa)  0 (varattu)
      JUMP  LOOP     # odota kunnes lukko auki
      ...            # lukko on varattu (kiinni) minulle
      kriittinen vaihe: yksi prosessi kerrallaan
      ...
      LOAD  R1,=1    # avaa lukko L
      STORE R1,L
```

- Toim
  - Mikä on “paha kohta”?

Ohjelmointikielten periaatteet, 4 ov

- Ei luennoitu muutamaan vuoteen
- Lähtötiedot: OLPM, TiTo, ohjelmointilabrat
- Sopiva: 3. vuoden opiskelijat
- Ohjelmointikielten määrittelyn välineistö
- Erilaiset ohjelmointiparadigmat esimerkkikielten avulla
  - proseduraaliset kielet
  - oliokielet
  - funktionaaliset kielet
  - logiikkaohjelmointikielet
- Muita asiaan liittyviä kursseja:
  - Symbolinen ohjelmointi, tekoälykielet

C, Pascal
Smalltalk
Scheme, ML
Prolog

Ohjelmointikielten kääntäjät, 5 ov

- Luennoidaan satunnaisesta muutaman vuoden välein
- Lähtötiedot: OLPM, ohjelmointilabrat
- Sopiva: 3. vuoden opiskelijat
- Ohjelmointikielten kääntäjien tyypit
  - rekursiivisesti etenevä jäsentelijä
- Kääntäjän osat
  - selaaja
  - jäsentelijä
  - semantiikan analyysi, koodin generointi

lex
yacc

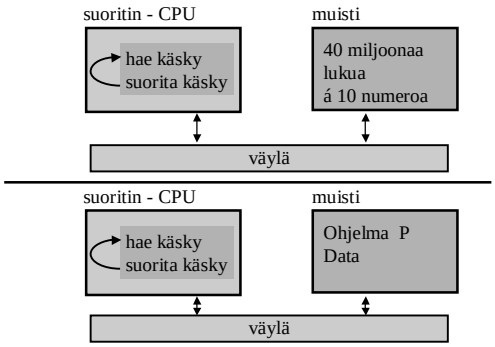
Spesifioinnin ja verifioinnin perusteet, 2 ov

- Lähtötiedot: hajautuksen ja samanaikaisuuden problematiikka
- Sopiva: 3. vuoden opiskelijalle
- Mallinnetaan prosesseja siirtymäsysteemeillä
  - askel: konekäsky? metodi? tapahtuma? ohjelma?
- Automaattisen verifioinnin periaatteet
- Yksinkertaisten protokollien verifiointi
- Jatkoa syventävällä tasolla
  - Ohjelmien semantiikka, 3 ov
  - Automaattinen verifiointi, 3 ov

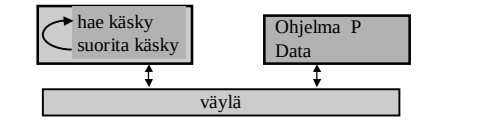
Ohjelmoinnin ja laskennan perusmallit (OLPM), 2 ov

- Lähtötiedot: matematiikkaa
  - appro tai diskr. mat., ... + tira?
- Sopiva: 1. vuoden (2. vuoden?) opiskelijalle, joka on opiskellut jo matematiikkaa
- Laskennalliset ongelmat, niiden luokittelu
- Äärelliset automaatit ja säännölliset kielet
- Kieliopit
- Turingin kone
- Jatkoa syventävällä tasolla
  - Laskennan teoria, 3 ov

Laskennan teorian perusta (1)



Laskennan teoriaa … (4)



Muistin sisältö ennen P:n suoritusta: X = hyvin iso kokonaisluku (200M numeroa?)

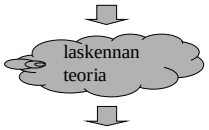
Muistin sisältö P:n suorituksen jälkeen: Y = joku toinen hyvin iso luku

P on kokonaislukuarvoinen funktio P: $\mathbb{N} \rightarrow \mathbb{N}$

Ohjelman P esitysmuoto muistissa: Iso kokonaisluku, P$\in \mathbb{N}$

Laskennan teoriaa ... (4)

- Mielivaltaisten ohjelmien ominaisuuksia voi päätellä kokonaislukujen ja niiden välisten funktioiden ominaisuuksista



- Todistettuja lauseita ohjelmien ominaisuuksista
  - pätevät kaikille tietokoneille
  - nyt ja tulevaisuudessa

27.4.2004 Teemu Kerola, Copyright 2003 43

Laskennan teoriasta ja algoritmianalyysistä todistettuja lauseita (3)

- Valitaanpa mikä tahansa aikaraja tai muistin koko, niin aina on olemassa sellainen ongelma, että
  - (1) siihen on olemassa ratkaisu ja
  - (2) kaikki ongelman ratkaisevat ohjelmat vievät enemmän aikaa tai muistitilaa kuin ennalta annettu raja
- On olemassa sellaisia ongelmia, että niitä ei voi ratkaista millään tietokoneella
- On olemassa suuri joukko tunnettuja vaikeita ongelmia, joista ei vielä tiedetä, kuinka vaikeita ne oikeastaan ovat

$$\mathbf{P} \stackrel{?}{=} \mathbf{NP}$$

27.4.2004 Teemu Kerola, Copyright 2003 44

--
Luennon 12
ja
koko kurssin
loppu
--

http://www.retroweb.com/apollo_retrospective.html

http://lue.kurssikokeeseen.edu/ajoissa.html

27.4.2004 Teemu Kerola, Copyright 2003 45