

Luento 12 Yhteenveto

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

27.4.2004

Teemu Kerola, Copyright 2003

1

Tavoitteet

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

27.4.2004

Teemu Kerola, Copyright 2003

2

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

27.4.2004

Teemu Kerola, Copyright 2003

3

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

27.4.2004

Teemu Kerola, Copyright 2003

4

Nopeuserot: Teemun juustokakku

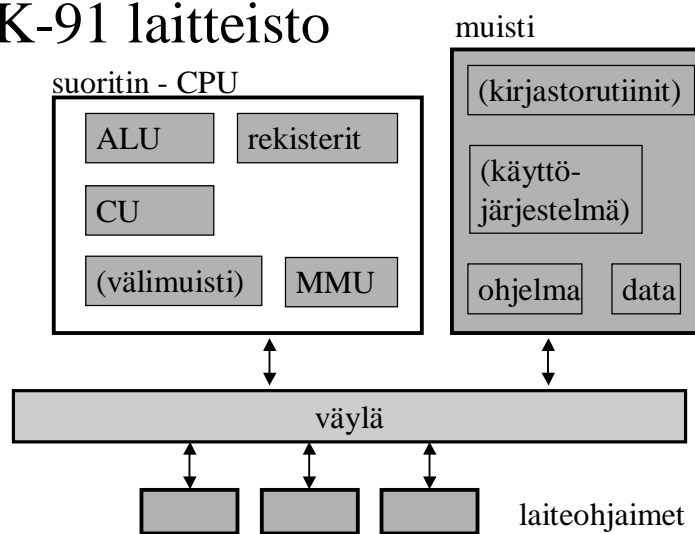
Rekisterien, välimuistin, muistin, levymuistin ja magneettinauhan nopeudet suhteutettuna juuston haku-aikaan juustokakkuja tehdessä?



27.4.2004

Teemu Kerola, Copyright 2003

Esimerkkikone: TTK-91 laitteisto



27.4.2004

Teemu Kerola, Copyright 2003

6

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

27.4.2004

Teemu Kerola, Copyright 2003

7

Konekielinen ohjelmointi

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

muuttujat, vakiot
taulukot, tietueet

muistissa, rekisterissä?

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

I	DC	0
...		
		LOAD R1, =20 STORE R1, I
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

27.4.2004

Teemu Kerola, Copyright 2003

8

TTK-91 muistin osoitusmoodit (8)

ks. lista sivulla 50
[Häkk98]

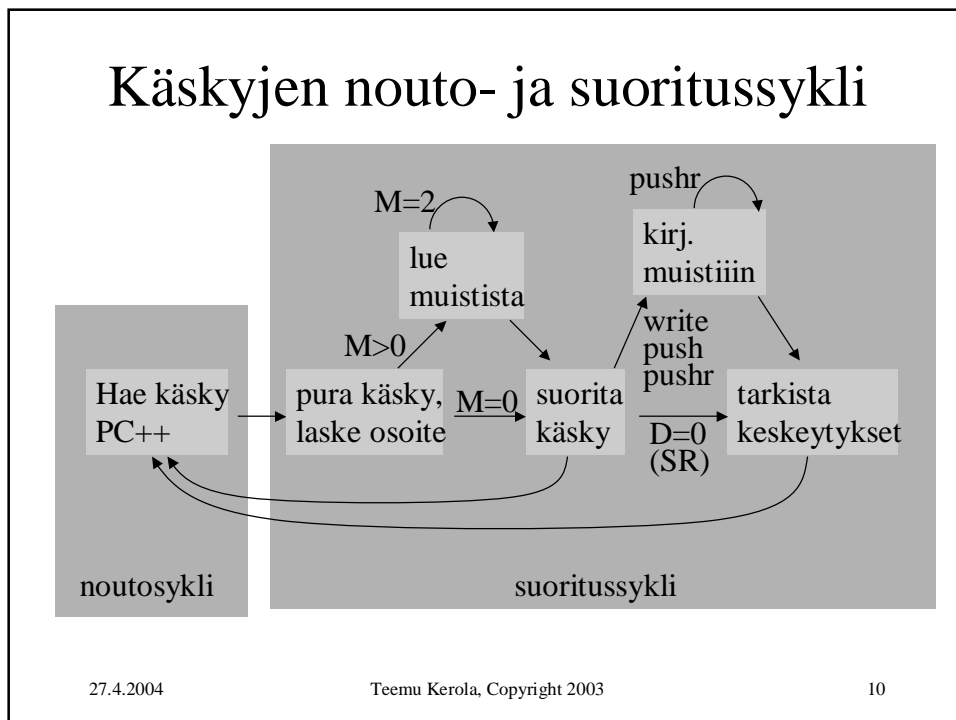
LOAD R1, 10	; R1 ← 200
LOAD R1, =10	; R1 ← 10
LOAD R1, @10	; R1 ← 6000
LOAD R4, R2	; R4 ← 201
LOAD R4, @R2	; R4 ← 11
LOAD R5, =Tbl(R3)	; R5 ← 201
LOAD R5, Tbl(R3)	; R5 ← 11
LOAD R5, @Tbl(R3)	; R5 ← 300

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

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

Tbl:	200
X:	10
One:	1

27.4.2004 Teemu Kerola, Copyright 2003 9



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 konerakenteita, myös etuoikeutettuja (esim. clear_cache, iret)
 - voi viitata kaikkialle muistiin, myös käyttöjärjestelmän ytimeen (kernel)
 - voi käyttää (myös) suoria muistiosoitteita (PA)

27.4.2004
Teemu Kerola, Copyright 2003
11

Aktivoititietue (Aktivoititietuepino)

(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 aktivoititietue
 - kaikki paikalliset muuttujat ja tietorakenteet
 - aliohjelman ajaksi talletettujen rekistereiden alkuperäiset arvot

27.4.2004
Teemu Kerola, Copyright 2003
12

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
STORE R1, T ← aina rekisteri
...
        
```

27.4.2004
Teemu Kerola, Copyright 2003
13

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

      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
      SUB  SP, =1 ; free Z
      EXIT SP, =2 ; 2 param.
        
```

prolog
epilog

27.4.2004
Teemu Kerola, Copyright 2003
14

Tiedon esitysmuodot

kokonaisluvut

liukuluvut

merkit

merkkijonot

kuvat

äänet

ei-standardoitu tieto

suorittimen ymmärtämä tieto

Lukujärjestelmiä:

Binääriluvut, oktaaliluvut,

heksadesimaaliluvut

27.4.2004

Teemu Kerola, Copyright 2003

15

Negatiiviset luvut (4)

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

arvo talletus
+57 = 0011 1001

sign bit = MSB
= most significant bit

luku -57 = 1011 1001 talletusmuoto

-57 = 1100 0110

“sign” bit +1

-57 = 1100 0111

“sign” bit

-57 = 0100 0110

-57 + 127 = 70

+57 = 1011 1000

+57 + 127 = 184

27.4.2004

Teemu Kerola, Copyright 2003

16

Tiedon esitysmuodot: liukuluku

“+”	“15”	“0.1875” = “0.0011”
sign	exponent	mantissa or significand

1/8 = 0.1250
1/16 = 0.0625
0.1875

- 23 bittiä mantissalle, siten että ...

- 1) Binääripiste (.) on heti ensimmäisen bitin jälkeen
- 2) Mantissa on normalisoitu: vasemmanpuolimmainen bitti on 1
- 3) Vasemmanpuolimmaista (eniten merkitsevä) bittiä (1) ei talleteta (implied bit)

mantissa eksponentti

0.0011	“15”
1.1000	“12”
1000	“12”

24 bitin mantissa!

27.4.2004
Teemu Kerola, Copyright 2003
17

Big vs. Little Endian (3)

- Miten monitavuiset arvot talletetaan?

0x1200:

--	--	--	--

0x1200 0x1201 0x1202 0x1203

Sanan osoite tavuosoitteet

talleta 0x11223344 ??

Big-Endian: eniten merkitsevä tavu pienimpään osoitteeseen

Little-Endian: vähiten merkitsevä tavu pienimpään osoitteeseen

0x11	0x22	0x33	0x44
0x1200	0x1201	0x1202	0x1203
0x44	0x33	0x22	0x11

27.4.2004
Teemu Kerola, Copyright 2003
18

Virheen korjaava Hamming koodi

Data: 100 1100 110 1100 (parillinen pariteetti)

Bitti nro: 765 4321 765 4321

Pariteettibitti 1 tarkistaa bittejä 1, 3, 5, 7

Pariteettibitti 2 tarkistaa bittejä 2, 3, 6, 7

Pariteettibitti 4 tarkistaa bittejä 4, 5, 6, 7

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

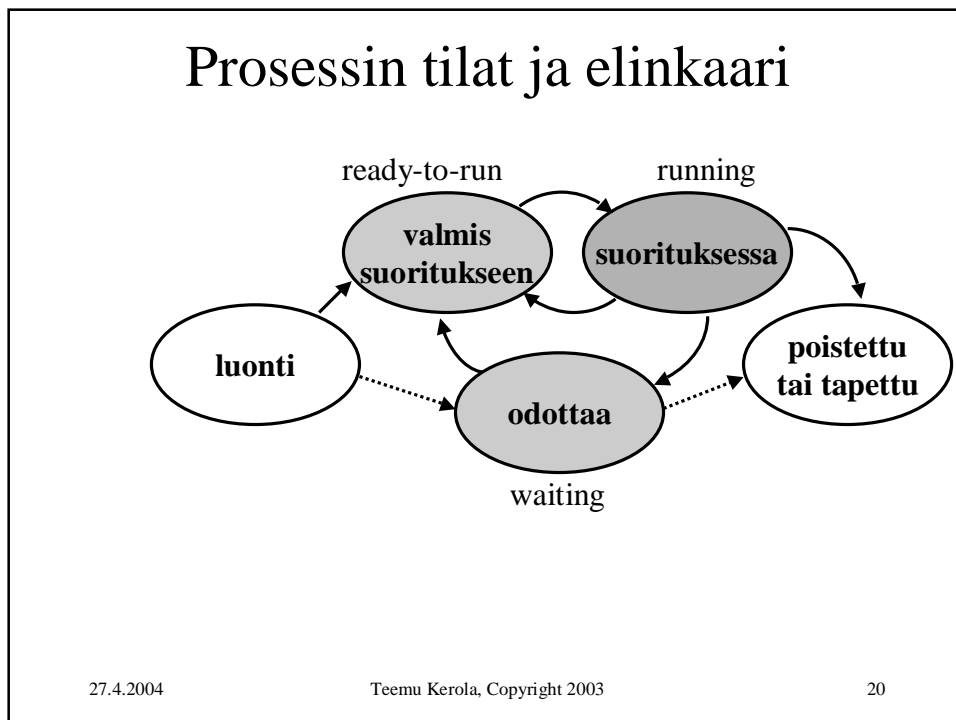
Tapahtuu virhe: bitti 6 muuttuu (flips)

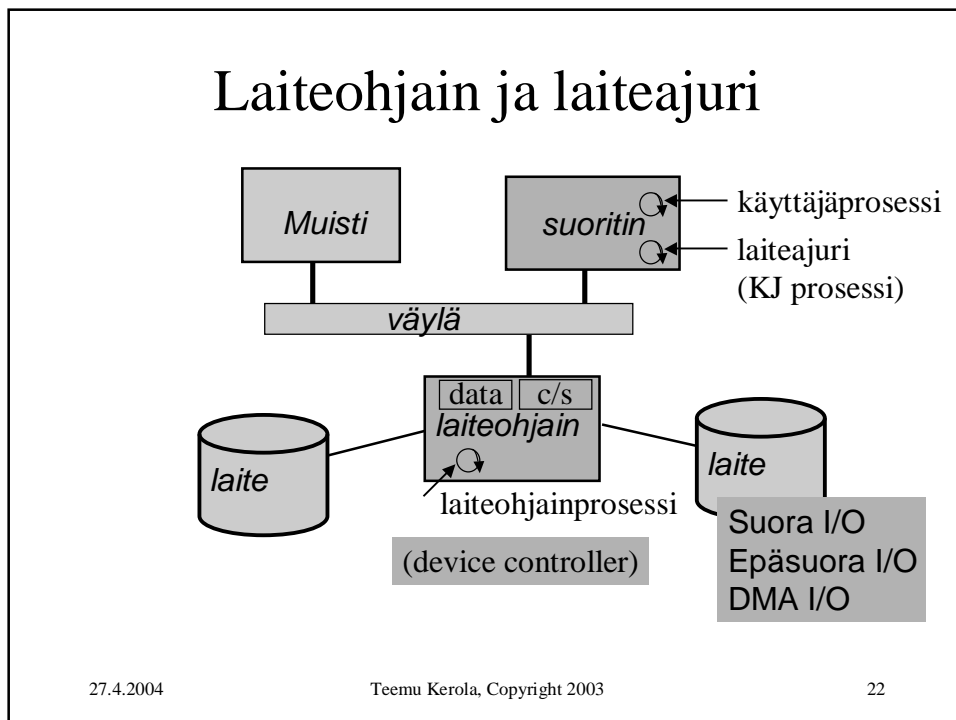
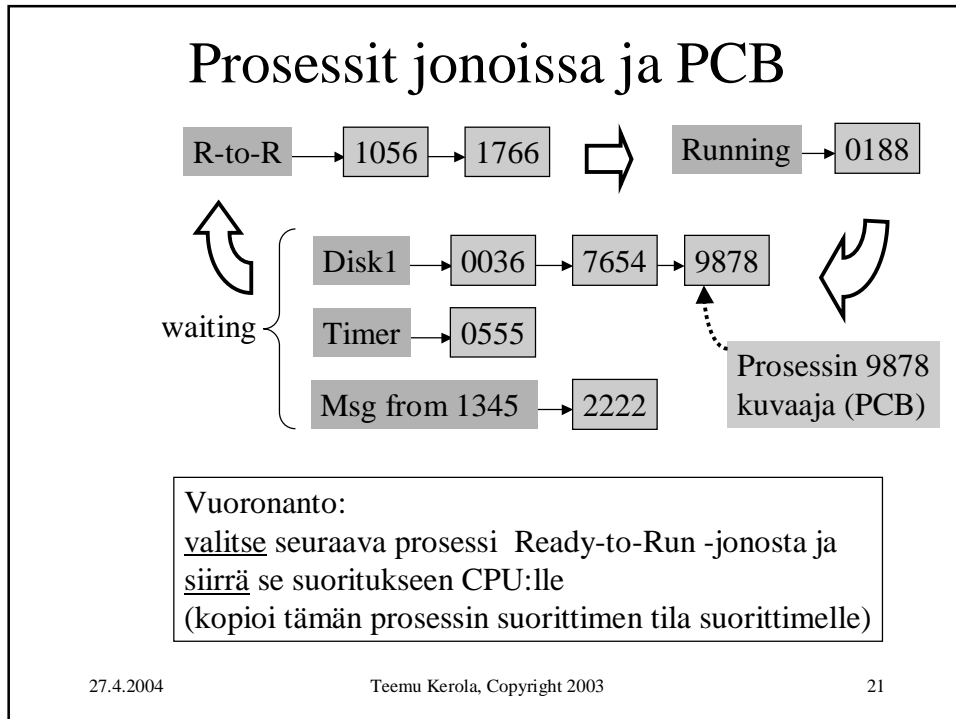
Pariteettibitti 2 tarkistaa bittejä 2, 3, 6, 7: **VIRHE**

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

2+4 = 6 ⇒ korjaa bitti nro 6

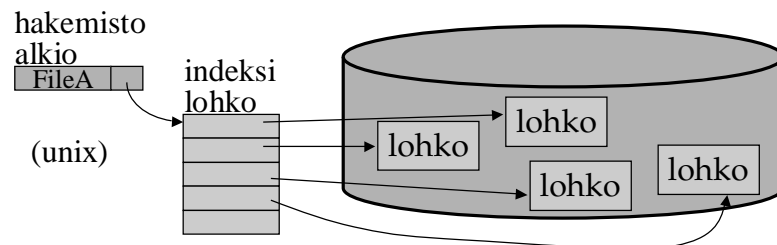
27.4.2004 Teemu Kerola, Copyright 2003 19





Levyjen käyttö

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

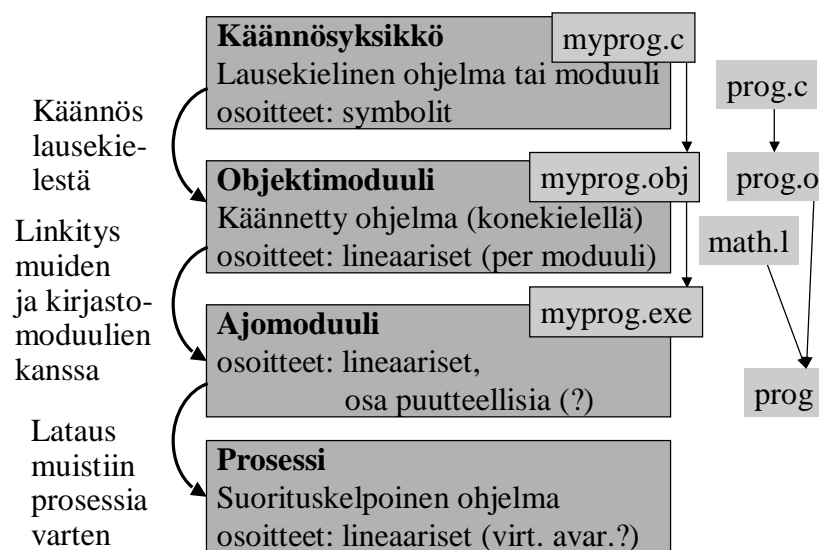


27.4.2004

Teemu Kerola, Copyright 2003

23

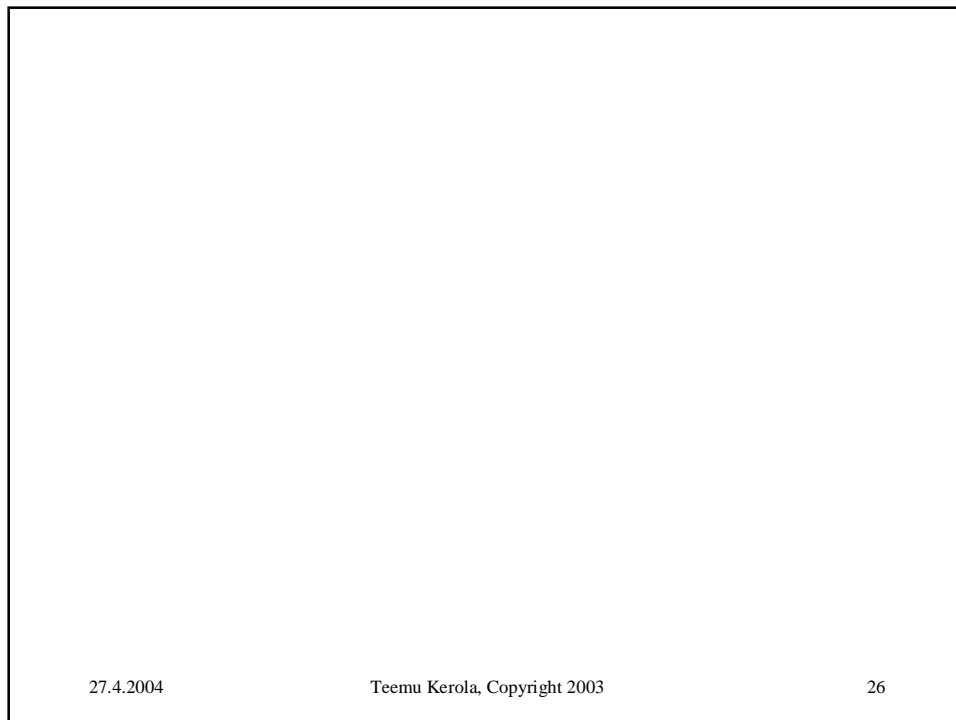
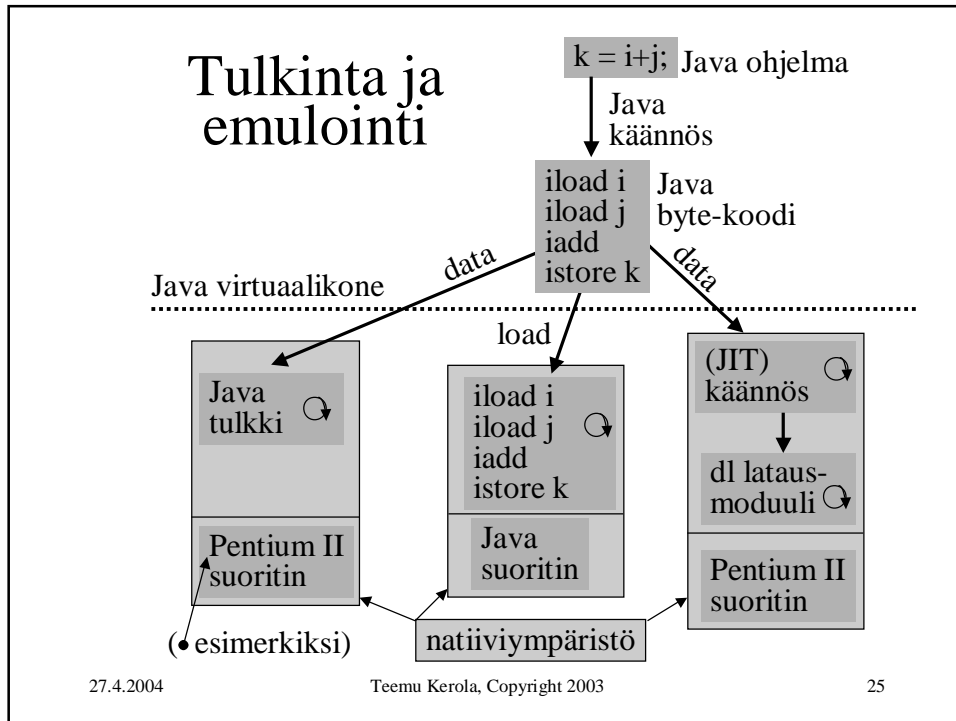
Lausekielestä suoritukseen

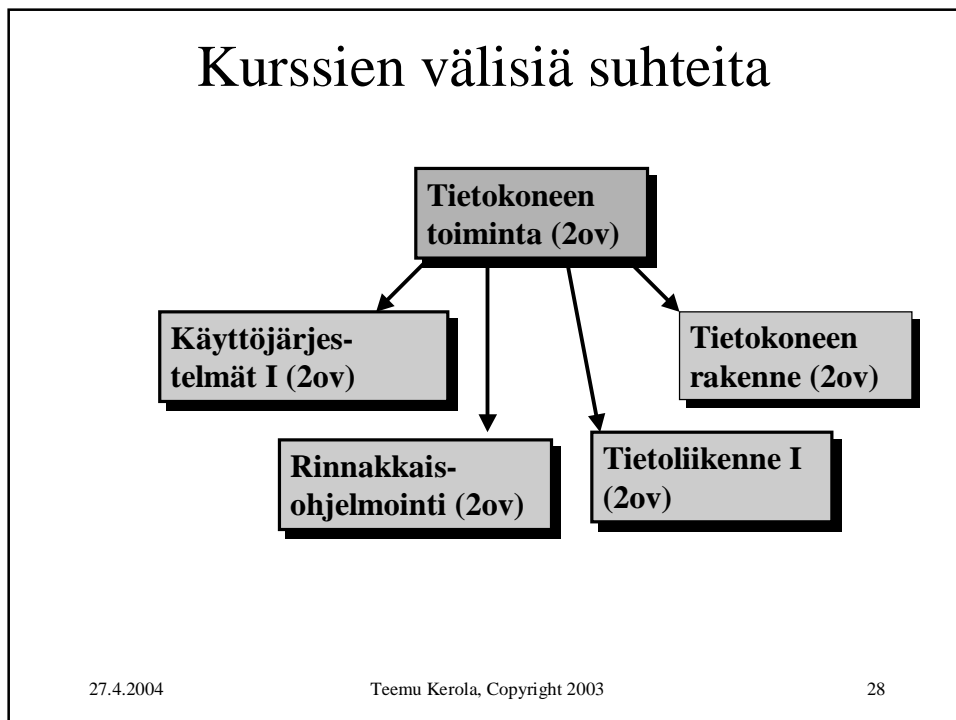
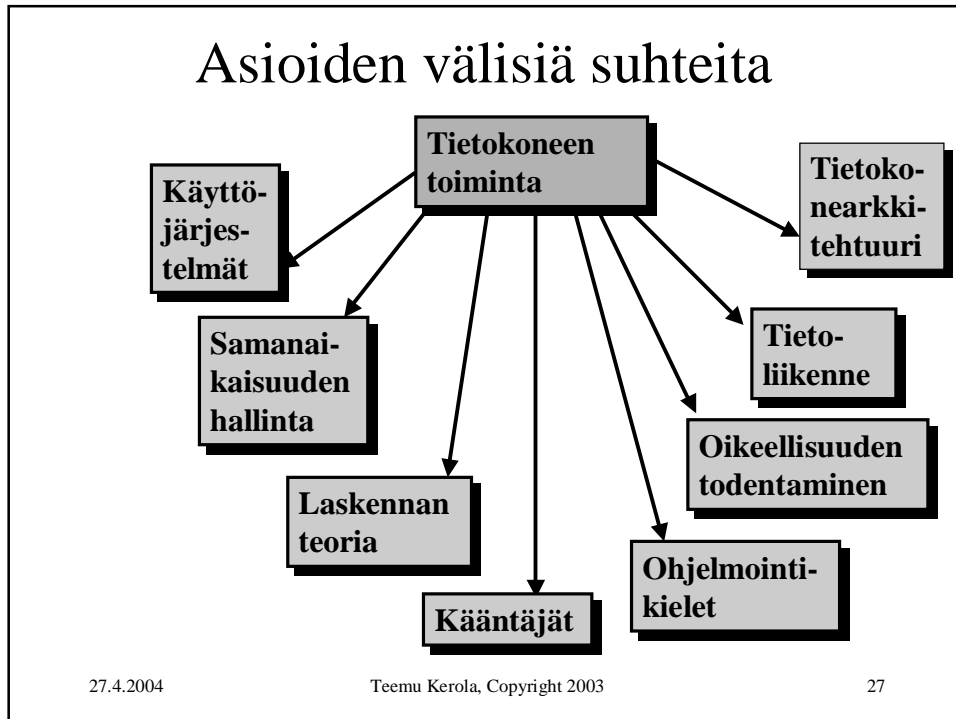


27.4.2004

Teemu Kerola, Copyright 2003

24





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

27.4.2004

Teemu Kerola, Copyright 2003

29

TiKRa

```

    graph TD
      PT((Predict Taken)) -- Taken --> PT
      PT -- Not Taken --> PNT((Predict Not Taken))
      PNT -- Taken --> PT
      PNT -- Not Taken --> PNT
    
```

Figure 11.16 Branch Prediction State Diagram

Figure 11.11 Timing Diagram for Instruction Pipeline Operation

[Stal99]

27.4.2004

Teemu Kerola, Copyright 2003

30

Käyttöjärjestelmät I, 2 ov

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

27.4.2004

Teemu Kerola, Copyright 2003

31

KJ ...

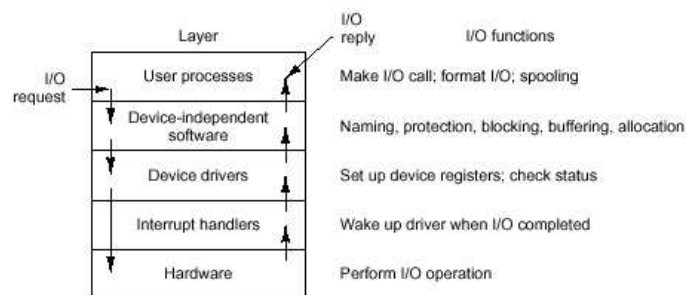


Figure 3-6. Layers of the I/O system and the main functions of each layer.

27.4.2004

Teemu Kerola, Copyright 2003

32

Tietoliikenne I, 2 ov

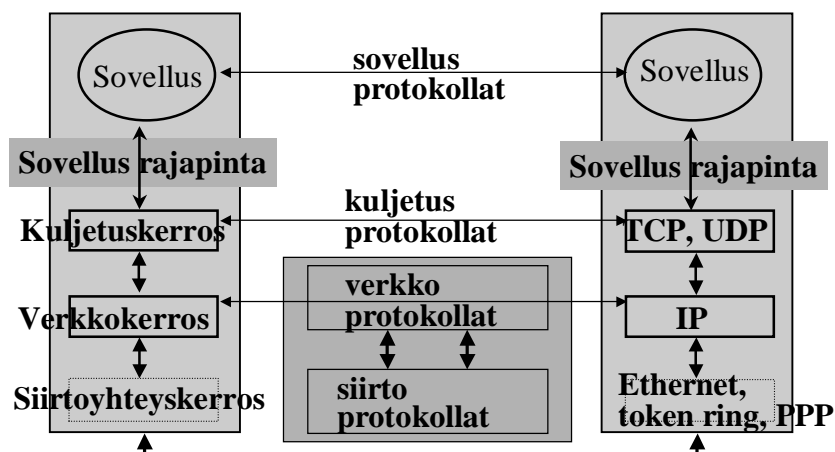
- Sopiva 2. vuoden opiskelijalle
- Tietokoneverkkojen peruspalvelut käyttäjälle ja sovelluksille
- Verkkojen tiedonsiirron perusvälineistö
- Verkkoarkkitehtuurin kerrosrakenne ja kunkin tason palvelut
- Jatkoa perustasolla ja syventävällä tasolla
 - Tietoliikenne II, 2 ov
 - Tietoliikennejärjestelmät,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

27.4.2004

Teemu Kerola, Copyright 2003

33

Tietoliikenne ... TCP/IP -kerrosmalli



27.4.2004

Teemu Kerola, Copyright 2003

34

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

27.4.2004

Teemu Kerola, Copyright 2003

35

RIO: synkronointiongelman ratkaisu Test-and-Set -käskyllä

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

```

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

tämän
käskyn
osoite

```

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
  
```

- Toimi...
- Mikä on “paha kohta”?

27.4.2004

Teemu Kerola, Copyright 2003

36

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 C, Pascal
 - oliokielet Smalltalk
 - funktionaaliset kielet Scheme, ML
 - logiikkaohjelmointikielät Prolog
- Muita asiaan liittyviä kursseja:
 - Symbolinen ohjelmointi, tekoälykielet

27.4.2004

Teemu Kerola, Copyright 2003

37

Ohjelmointikielten kääntäjät, 5 ov

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

27.4.2004

Teemu Kerola, Copyright 2003

38

Spesifioinnin ja verifioinnin perusteet, 2 ov

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

27.4.2004

Teemu Kerola, Copyright 2003

39

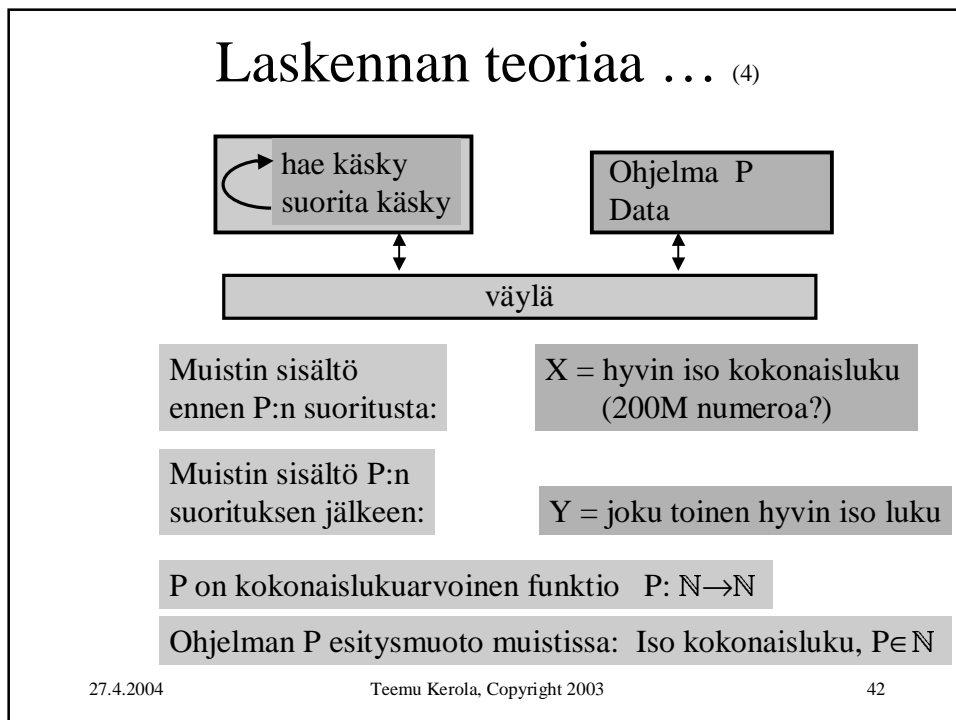
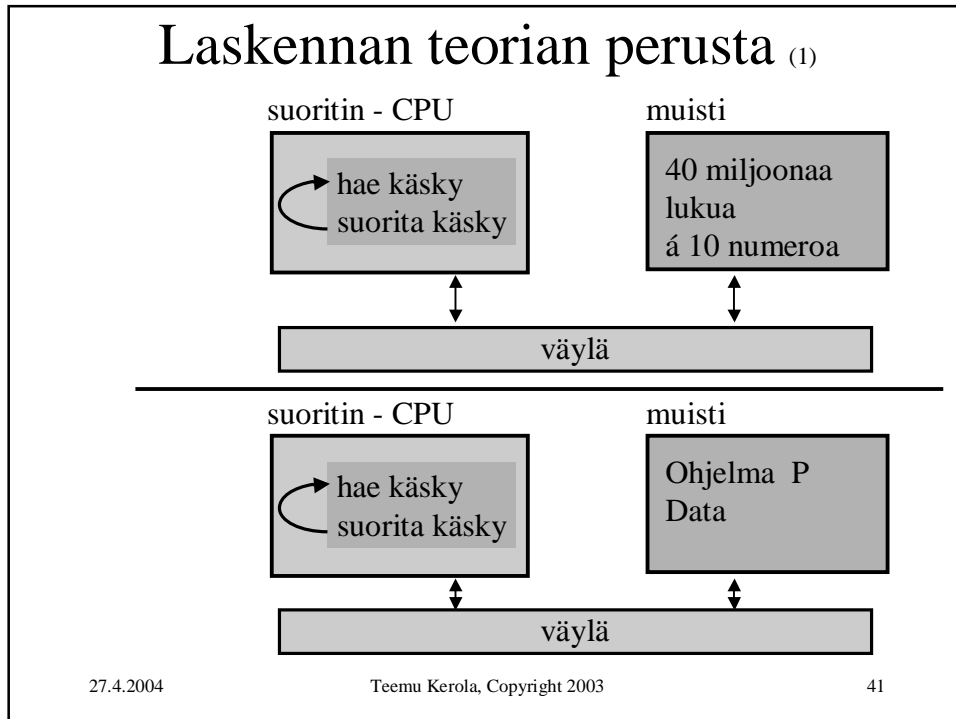
Ohjelmoinnin ja laskennan perusmallit (OLPM), 2 ov

- Lähtötiedot: matematiikkaa
 - appro tai disk. 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

27.4.2004

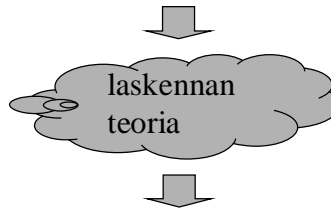
Teemu Kerola, Copyright 2003

40



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

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

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