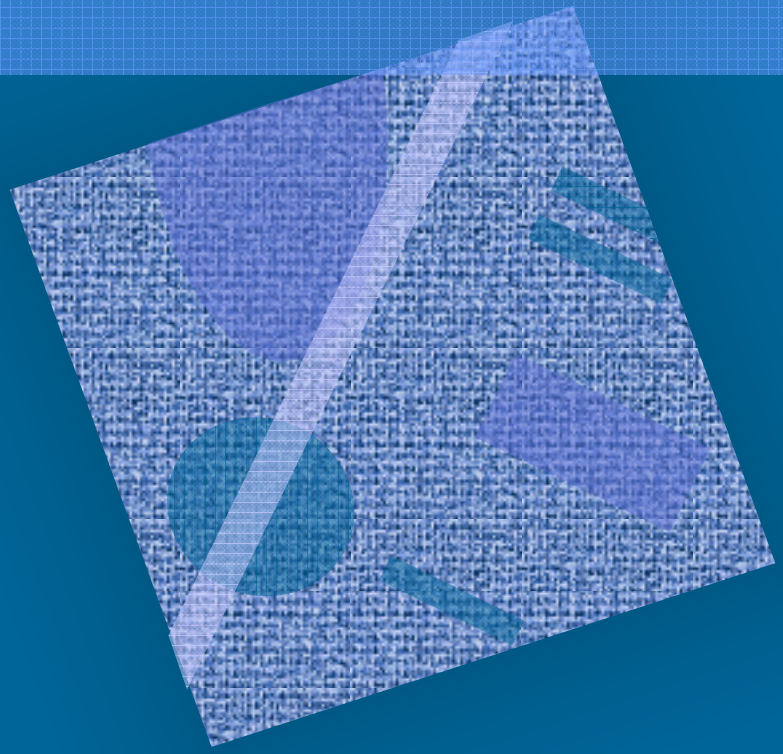
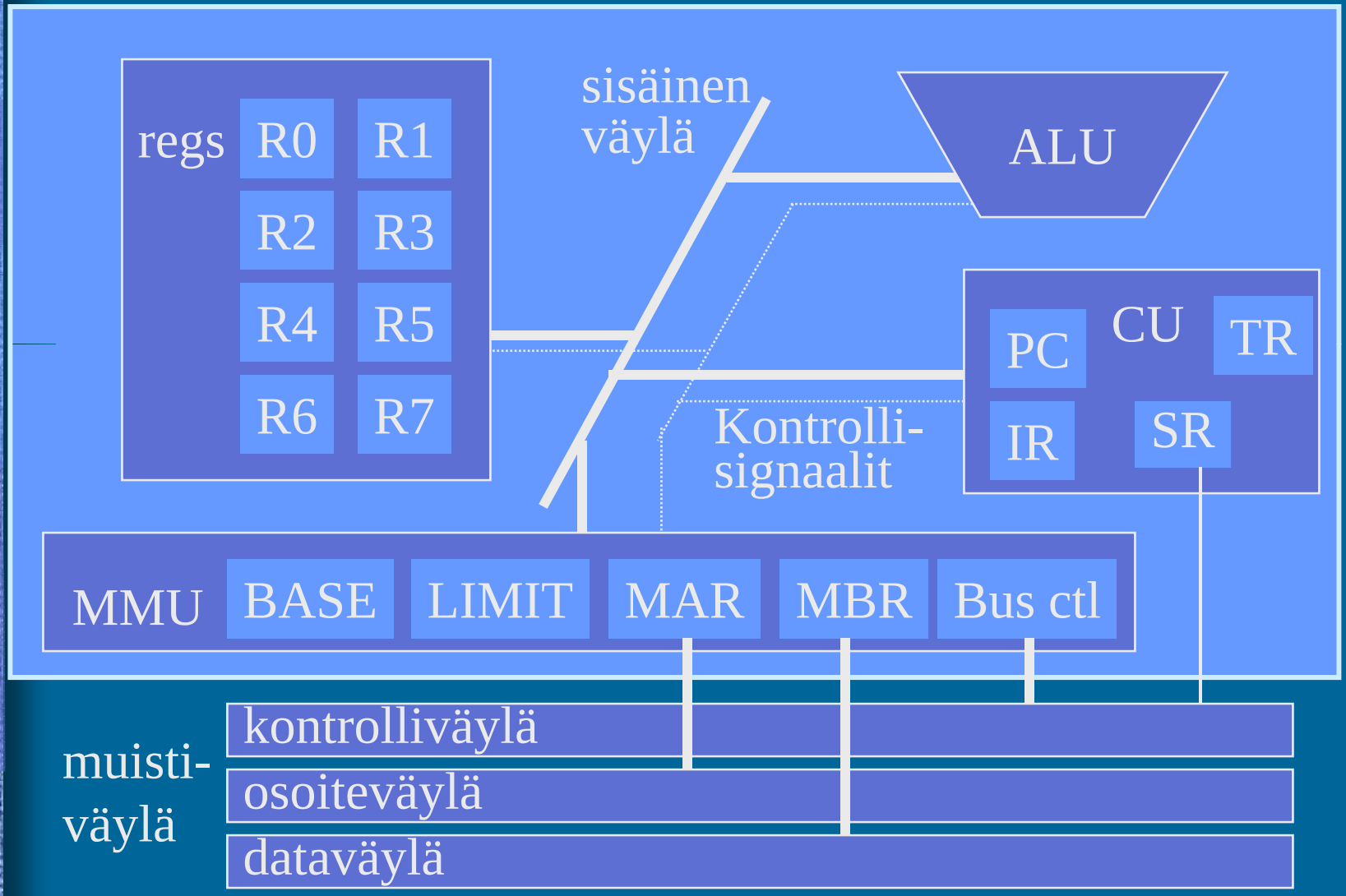
Luento 4 (verkkoluento 5)

Suoritin ja väylä



Käskyjen suoritussykli
Suorittimen tilat
Poikkeukset ja keskeytykset

TTK-91 suorittimen rakenne



Käskyn suorittaminen

- Titokone näyttää vaiheet simuloimalla
- Store R4, @10(R1) ; R1 =20, R4=15
  - Käskyn nouto ja ohjelmalaskurin kasvatus:
    - PC -> MAR, mem -> MBR, (odota), MBR -> IR
    - PC+1 -> PC
      - PC -> ALU1, 1-> ALU2, +, (odota), ALU -> PC (??)
  - Käskyn suoritus:
    - R1 -> ALU1, 10 -> ALU2, +, (odota), ALU-> TR
    - TMP -> MAR, mem -> MBR, (odota),
      MBR -> MAR, R4 -> MBR, MBR -> mem, (odota)
  - Huom: yhteensä 3 muistiviitettä

TTK-91 konekäskyn rakenne

- Käskyn esitys bittitasolla on **aina**:

(pienehkö) muisti-osoite tai vakio

Rj = käskyn ensimmäinen operandi

Ri = indeksirekisteri (0 ≡ ei käytetä)

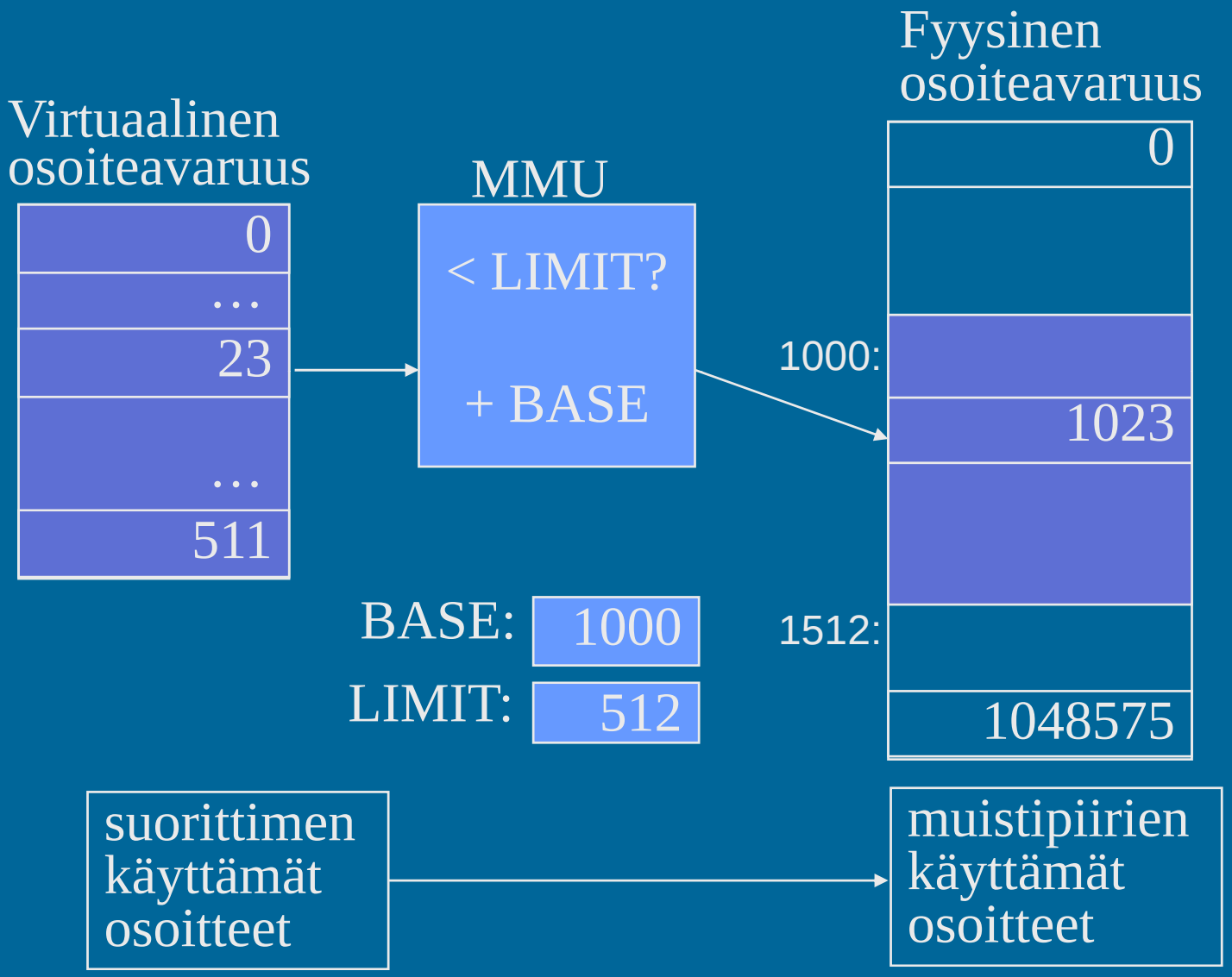
M = muistinoutojen määrä toiseen operandiin (addressing mode)
(ennen mahdollista muistiin talletusta)
00 eli 0 kpl, välitön osoitus (STORE: suora osoitus)
01 eli 1 kpl, suora osoitus (STORE: epäsuora osoit.)
10 eli 2 kpl, epäsuora osoitus (STORE: epäkelpo arvo)
(11 eli 3 kpl, epäkelpo arvo → poikkeustilanne)

Keskustele

Käskyn purku ja tehollisen muisti-osoitteen (EA) laskemisvaihe

- Purku automaattisesti langoitettuna käskyrekisteristä (IR)
- Muistiosoitteen lasku, tulos tilapäisrekisteriin (TR)
  - jos Ri=0, niin TR ← ADDR
  - muutoin TR ← (Ri)+ADDR
    - ALU suorittaa laskutoimituksen
    - jos ADDR = 0, niin TR ← (Ri)
  - Effective Address (EA) on nyt TR:ssä
- EA:ta voi käyttää datana sellaisenaan tai muistiosoitteena

TTK-91 Kanta ja rajarekisterit



Keskustele

Virtuaalimuistin osoitteenmuunnosmenetelmiä

- Kanta- ja rajarekisteriin perustuva
  - base ja limit rekisterit (esim. ttk-91, 8086, ...)
- Sivuttava
  - sivutaulut
  - virtuaaliavaruus jaettu saman kokoisiin sivuihin
- Segmentoiva
  - virtuaaliavaruus jaettu ohjelman mukaan erillisiin eri kokoisiin segmentteihin
    - koodi segmentti, data segmentti, …

Lisää tietoa?  käyttö-järjestelmä kurssi

Lisää tietoa?  käyttö-järjestelmä kurssi

Keskeytysten käsittely

- Jokainen mahdollinen keskeytystyyppi on ennalta tunnettu, eli *mitään todella yllättävää ei tapahdu*
- Jokaiselle keskeytystyypille on oma käyttöjärjestelmän tuntema keskeytyskäsittelyrutiini

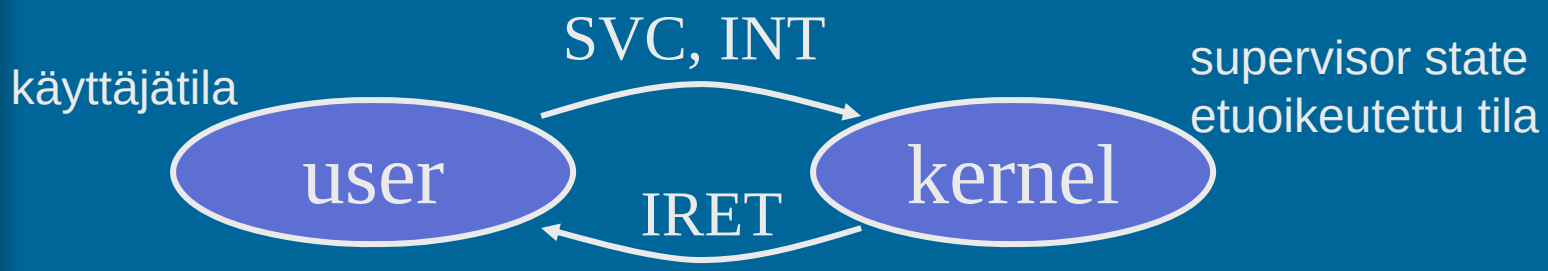
interrupt handler

- Jokaisen käskyn suorituksen jälkeen tarkistetaan keskeytysten olemassaolo SR:stä ja haaraudutaan keskeytyskäsittelijään tarvittaessa
  - joskus keskeytykset on estetty (ttk-91:ssä SR:n bitti D)
  - paluu käsittelijästä ”return-from-interrupt-handler” käskyllä (esim. IRET, tms)
- ”Yllättävä aliohjelmakutsu”

Keskeytyskäsittelijä

- Osa käyttöjärjestelmää
- Ennen keskeytyskäsittelijään hyppäämistä asetetaan suoritin ja MMU etuoikeutettuun käyttöjärjestelmätilaan (supervisor state, kernel st.)
  - SR:n bitti P on päällä => etuoikeutettu tila eli käyttöjärjestelmä tila (P = Priviledged)
  - käyttöjärjestelmätilassa saa viitata mihin tahansa kohtaan muistia (MMU: BASE=0, LIMIT="hyvin iso")
  - käyttöjärjestelmätilassa saa käyttää kaikkia konekäskyjä (esim. IRET tai ClearCache)
- Käsittelijästä paluun yhteydessä MMU:n tila ja suorittimen tila (bitti P) asetetaan ennalleen

Suorittimen tilan muuttaminen



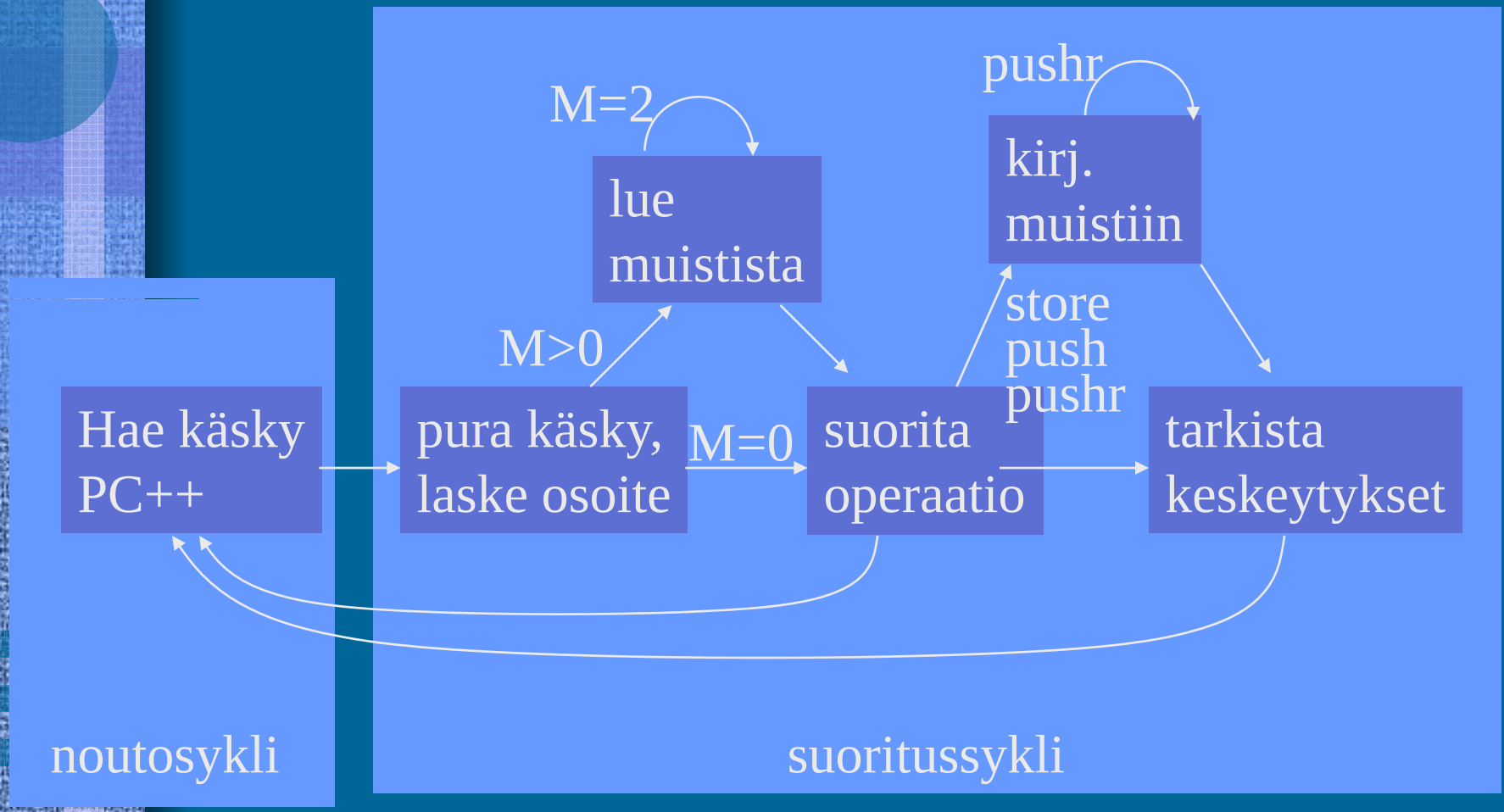
- Käyttäjätila → etuoikeutettu tila
  - keskeytys tai suora KJ:n palvelupyyntö (SVC käsky)
  - keskeytyskäsittelijä tarkistaa onko (oliko) oikeutta tilan vaihtoon

(interrupt handler)

- Etuoikeutettu tila → käyttäjätila
  - etuoikeutettu konekäsky “return from interrupt handler” esim. IRET (Pentium II)
  - palauttaa kontrollin keskeytyneeseen kohtaan ja suorittimen tilan keskeytystä edeltäneeseen tilaan

Keskustele

TTK-91 Nouto- ja suoritussykli vielä vähän tarkemmin



Keskustele

Titokone - TTK-91 koneen simulaattori

- Tavallinen Javalla kirjoitettu ohjelma
- TTK-91 koneen osat tietorakenteina
  - rekisterit, MMU, CU, muisti
- Simuloi käskyjen suoritussykliä käsky kerrallaan
- Toteuttaa myös TTK-91 koneen käyttöjärjestelmän osat osana tavallista ohjelmaa
  - assembler kääntäjä, lataaja, debugger, kesk. käsittelijät
- Graafinen käyttöliittymä

Ks. Processor.java Titokoneen koodissa:

titokone.jar\fi\hu\cs\titokone\Processor.java

(http://www.cs.helsinki.fi/group/nodes/kurssit/tito/Processor.java.txt)

TTK-91 käskyn suoritussykli

hae käsky simuloidusta muistista

IR = mem[PC]

pura käsky osiin (OPER, Rj, M, Ri, ADDR) ja
laske osoiteosan arvo TR (ADDR tai regs[Ri]+ADDR)

ADDR = IR mod 32768

TR = regs[Ri] +ADDR

tee tarvittava määrä (M) operandin
hakuja muistista rekisteriin TR

TR = mem[TR]

valitse aliohjelma operaatiokoodin (OPER) perusteella

if (opcodeOK[OPER] = FALSE) then SR.U = 1;

simuloi konekäskyn suorituksen muutokset
rekistereihin (R0…R7, SR, PC, MAR, MBR)

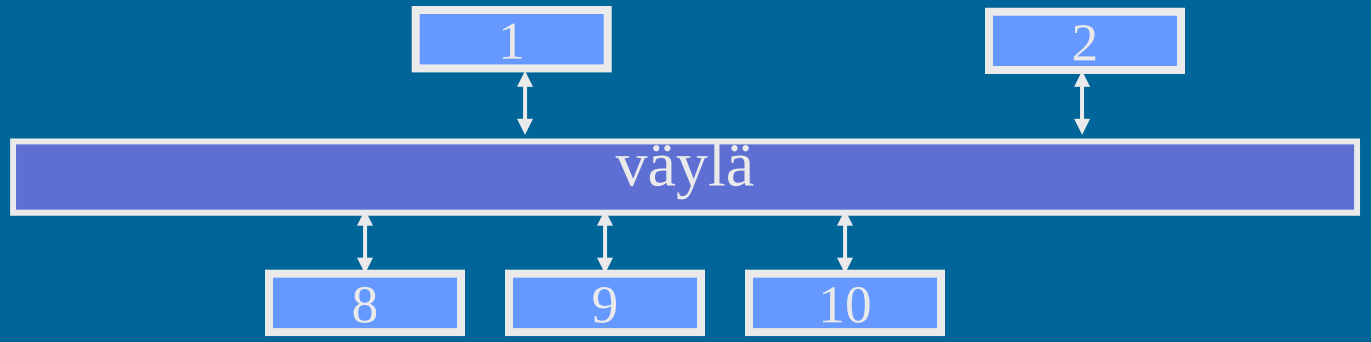
ADD Rj, M ADDR(Ri) ⇒ regs[Rj] += TR;

lopeta suoritus jos SVC tai keskeytys

SR.O = …

Simulaattori C:llä: http://www.cs.helsinki.fi/group/nodes/kurssit/tito/simu/simu.c

Väylät

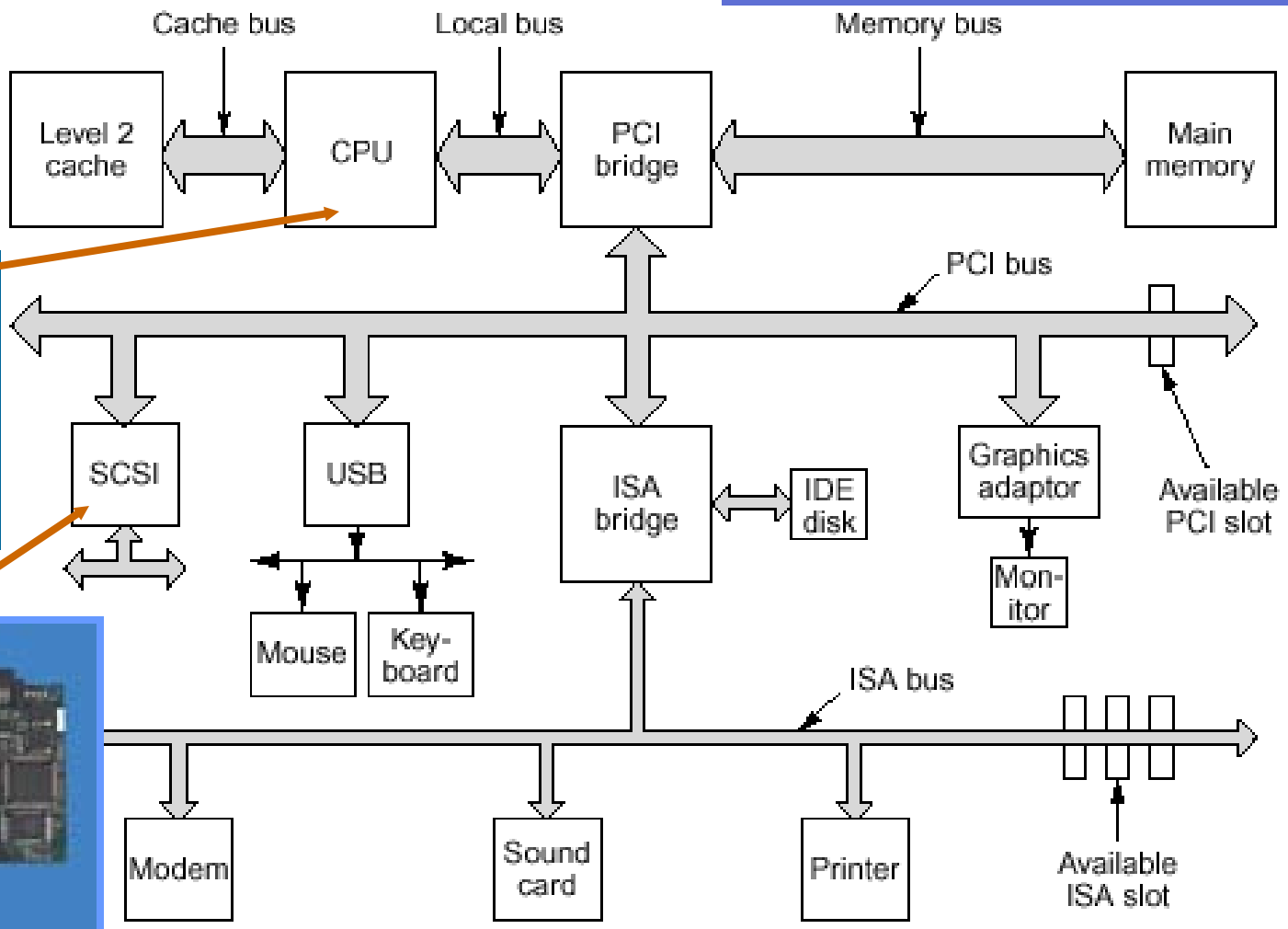


- Kullakin laitteella oma osoite
- Yksi lähettää, kaikki kuulevat, vain ”oikea” laite vastaanottaa
- Paljon erilaisia
- Lähellä suoritinta olevat ovat nopeampia

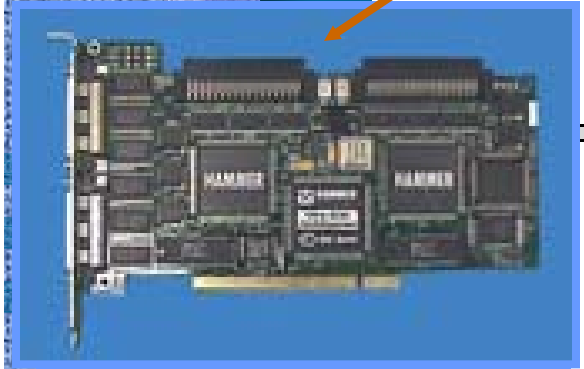
Lisää tietoa? → Tietokoneen rakenne -kurssi

Väylähierarkia

Tyypillinen Pentium II systeemin emolevy



omalla lastulla, tason 1 välimuistin kanssa



PCI to SCSI bridge

Fig. 3-50 [Tane99]

-- Luennon 4 loppu --

ESKO, 1960 ensimmäinen Suomessa rakennettu tietokone, vanhentunut jo valmistuessaan, 20 yhteenlaskua per sek.

- Ohjelmakoodi luettiin reikänauhoilta (10 kpl)
- Aliohjelmakutsu toteutettu kontrollin siirrolla toiseen reikänauhanlukijaan