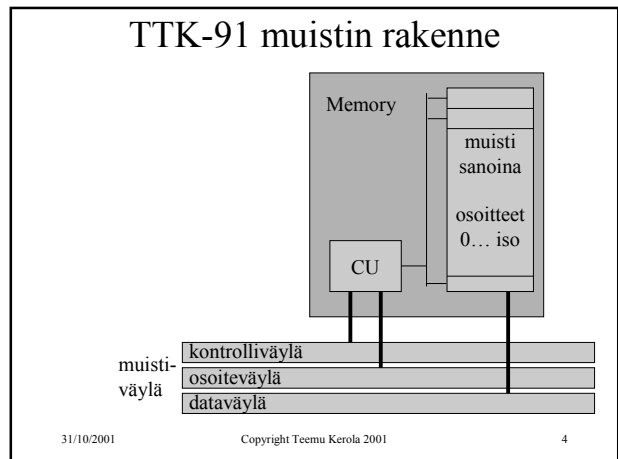
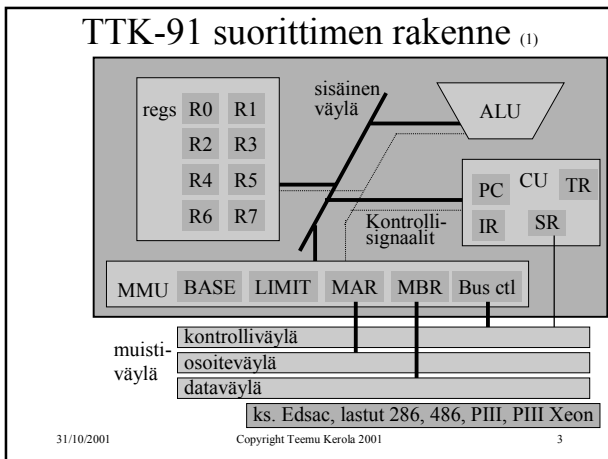
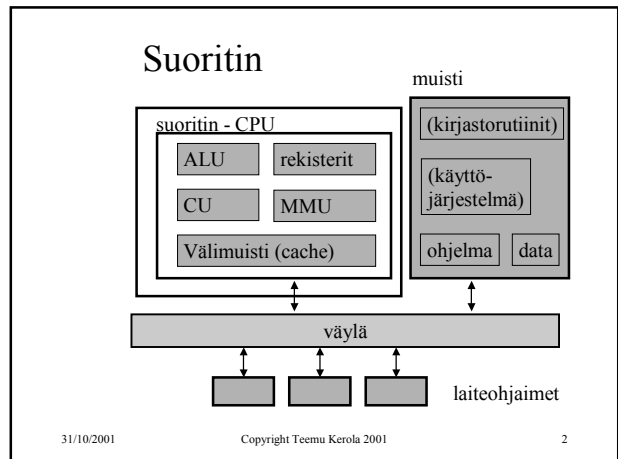


## Luento 5 Suoritin ja väylä

Suorittimen rakenne  
Väylän rakenne  
Käskeyjen suoritusykyli  
Poikkeukset ja keskeytykset  
KOKSI:n rakenne

31/10/2001 Copyright Teemu Kerola 2001 1



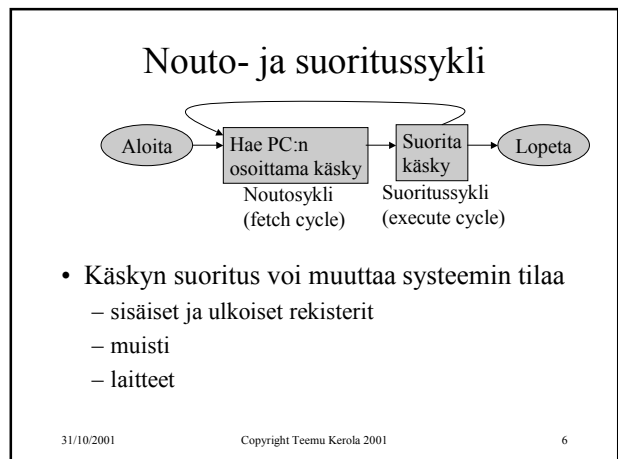
### Käskeyjen nouto- ja suoritusykyli (5)

- Hae PC:n osoittama konekäskey muistista
  - lisää samalla PC:n arvoa yhdellä
- Suorita konekäskey
  - jos (ehdollinen) hyppykäskey, niin PC:n arvo voi vielä muuttua

Suoritin ei näe mitään suurempia kokonaisuuksia kuin konekäskeyjä!

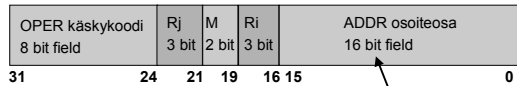
Suoritin ei tiedä mitään ohjelmista!

31/10/2001 Copyright Teemu Kerola 2001 5



## TTK-91 konekäskyn rakenne

- Käskyn esitys bittitasolla on aina:



Rj = käskyn ensimmäinen operandi

Ri = indeksirekisteri ( $R0 \equiv 0$ )

M = muistinoutojen määrä toiseen operandiin (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  $\rightarrow$  poikkeustilanne)

muistiosoite tai  
(pienehkö) vakio

(addressing mode)

31/10/2001

Copyright Teemu Kerola 2001

7

## Nouto- ja suoritussykli tarkemmin <sup>(5)</sup>

- Noutovaihe
  - muistista MBR:n kautta IR:ään
  - Lisää 1 PC:hen
- Käskyn purku ja muistiosoitteen (EA) lasku
  - OPER, Rj, M, Ri, ADDR
  - $TR \leftarrow (Ri) + ADDR$  (pelkkä ADDR, jos  $Ri=R0$ )
- Operandin nouto Ei kaikilla käskyillä
- ALU operaatio
  - tulos rekisteriin R0-R7 tai TR:ään (STORE, PUSH)
- Muistiin talletus Ei kaikilla käskyillä
  - muistiin MBR:n kautta

ks. TTK-91  
suorittimen  
rakennekuva

31/10/2001

Copyright Teemu Kerola 2001

8

## Käskyn noutovaihe <sup>(4)</sup>

ks. TTK-91 suorittimen rakennekuva

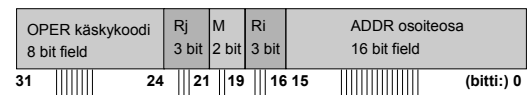
- Vie PC:n arvo MAR:iin
- Aseta muistin lukusignaali kontrolliväylälle asentoon "lue"
- Odota, kunnes muistipiiri toimittaa väylän kautta uuden arvon MBR:ään
- Siirrä konekäsky MBR:stä IR:ään

31/10/2001

Copyright Teemu Kerola 2001

9

## Käskyn purku ja tehollisen muistiosoitteen (EA) laskemisvaihe



- Purku automaattisesti langoitettuna IR:stä
- Muistiosoitteen lasku, tulos TR:ään
  - jos  $Ri=0$ , niin  $TR \leftarrow ADDR$
  - muutoin  $TR \leftarrow (Ri)+ADDR$ 
    - ALU suorittaa laskutoimituksen
  - Effective Address (EA) on nyt TR:ssä

31/10/2001

Copyright Teemu Kerola 2001

10

## Operandin luku vaihe <sup>(4)</sup>

ks. TTK-91 suorittimen rakennekuva

- Vie muistiosoite MAR:iin
- Aseta muistin lukusignaali kontrolliväylälle asentoon "lue"
- Odota kunnes muistipiiri toimittaa väylän kautta uuden arvon MBR:ään
- Siirrä sana MBR:stä TR:ään
  - (tai suoraan johonkin laiterekisteriin R0-R7)

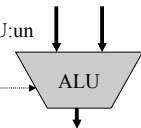
31/10/2001

Copyright Teemu Kerola 2001

11

## ALU operaatio -vaihe <sup>(10)</sup>

- Lähtötilanne ks. TTK-91 suorittimen rakennekuva
  - käsky haettu ja purettu osiin IR:ssä
  - 1. operandi rekisterissä ( $R0, \dots, R7$ )
  - 2. operandi TR:ssä
- Käskyn suoritus ALU:ssa
  - vie operandit sisäistä väylää pitkin ALU:un
  - anna ALU:lle sopiva ohjaussignaali
    - add, mul, copyLeft, comp, ...
  - odota, että tulos valmis
  - talleta tulos rekisteriin, MBR:ään, PC:hen ja/tai SR:ään



Tässä tapahtuu tietokoneen tekemä työ,  
kaikki muu on hallintoa

31/10/2001

Copyright Teemu Kerola 2001

12

### Tuloksen muistiin kirjoitus -vaihe <sup>(5)</sup>

ks. TTK-91 suorittimen rakennekuva

- Vie muistiosoite MAR:iin
- Vie kirjoitettava sana MBR:ään
- Aseta kirjoitussignaali kontrolliväylälle asentoon "kirjoita muistiin"
- Odota kunnes sana siirretään muistiin väylää pitkin ja väylän kontrollisignaali kertovat muistiinkirjoittamisen tapahtuneen

Lisää tietoa?

➔

Tietokoneen rakenne-kurssi

31/10/2001 Copyright Teemu Kerola 2001 13

### TTK-91 Nouto- ja suoritussykli vähän tarkemmin <sup>(1)</sup>

31/10/2001 Copyright Teemu Kerola 2001 14

### MMU:n toiminta <sup>(2)</sup>

ks. TTK-91 suorittimen rakennekuva

- Ohjelman käyttämät muistiosoitteet (VA) ovat näennäisiä, välillä 0 ... LIMIT-1
  - ne eivät ole samoja osoitteita kuin keskusmuisti käyttää
- MAR:iin menevä arvo VA ei käytetä suoraan, vaan se tarkistetaan ja muokataan ensin
  - Tarkista, onko VA ∈ [0, LIMIT-1]. Jos ei ole, niin aseta SR:n bitti M päälle ja lopeta käskyn suoritus
  - Lisää VA:han BASE ja laita tämä arvo (PA) MAR:iin

VA = virtual address, PA = physical address = BASE+VA

31/10/2001 Copyright Teemu Kerola 2001 15

### TTK-91 virtuaalimuisti

31/10/2001 Copyright Teemu Kerola 2001 16

### Virtuaalimuistin osoitteenmuunnos menetelmiä <sup>(4)</sup>

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

Lisää tietoa?

➔

käyttö-järjestelmä kurssit

31/10/2001 Copyright Teemu Kerola 2001 17

### Keskeytystilanteet <sup>(3)</sup>

- Mikä tahansa tilanne, jonka käsittely vaatii poikkeuksen käskyjen normaaliin suoritusjärjestykseen
- Rakkaalla lapsella on monta nimeä:
  - poikkeus, keskeytys, virhetilanne, trappi, ...
  - exception, interrupt, fault, trap, failure, ...
  - SCV, KJ-kutsu, ...
- Jatkossa yleisnimi keskeytys tarkoittaa kaikkia näitä eri tapauksia tai tyyppejä

31/10/2001 Copyright Teemu Kerola 2001 18

## Keskeytysten käsittely <sup>(4)</sup>

- Jokainen mahdollinen keskeytystyyppi on ennalta tunnettu
- Jokaiselle keskeytystyypille on oma käyttöjärjestelmän tuntema `interrupt handler` keskeytyskäsittelyrutiini
- Jokaisen käskyn suorituksen jälkeen tarkistetaan keskeytysten olemassaolo SR:stä ja haaraututaan keskeytyskäsittelijään tarvittaessa
  - joskus keskeytykset on estetty (ttk-91:ssä SR:n bitti D)
  - paluu käsittelijästä "return-from-interrupt" käskyllä
- ”Yllättävä aliohjelmakutsu”

31/10/2001

Copyright Teemu Kerola 2001

19

## Keskeytystyyppejä <sup>(3)</sup>

- Käskyn aiheuttamat virhetilanteet
- Käskyn aiheuttamat muut poikkeustilanteet
  - kyseessä ei siis ole virhetilanne
- Ulkoapäin (muualta kuin CPU:lta) tulleet

31/10/2001

Copyright Teemu Kerola 2001

20

## Käskyn aiheuttamat virhetilanteet <sup>(5)</sup>

- Virheellinen käskyn tai datan osoite
- Tuntematon käsky (opcode)
- Nollalla jako
- Kokonaisluvun tai liukuluvun yli/alivuoto
- Käytetty osoite ei ole muistissa (MMU)

31/10/2001

Copyright Teemu Kerola 2001

21

## Käskyn aiheuttamat muut poikkeustilanteet <sup>(3)</sup>

- SVC käsky
- I/O konekäsky
- Trace keskeytys

31/10/2001

Copyright Teemu Kerola 2001

22

## Ulkoapäin (muualta kuin suorittimelta) tulleet keskeytykset <sup>(3)</sup>

- Kellolaitekeskeytys (esim. joka 10 ms)
- Laitekeskeytys (esim. levy I/O valmis)
- Laitteistovirhe (esim. virhe väylän tiedonsiirrossa)

31/10/2001

Copyright Teemu Kerola 2001

23

## Keskeytyskäsittelijä

- Osa käyttöjärjestelmää
- Ennen keskeytyskäsittelijän aloittamista asetetaan suoritin ja MMU `(supervisor state)` käyttöjärjestelmätilaan
  - SR:n bitti P on päällä => etuoikeutettu tila eli käyttöjärjestelmä tila
  - käyttöjärjestelmätilassa saa viitata mihin tahansa kohtaan muistia (MMU: BASE=0, LIMIT="hyvin iso")
  - käyttöjärjestelmätilassa saa käyttää kaikkia konekäskyjä
- Käsittelijästä paluun yhteydessä MMU:n tila ja suorittimen tila asetetaan ennalleen

31/10/2001

Copyright Teemu Kerola 2001

24

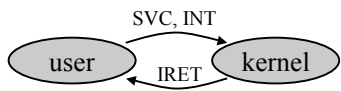
### Suorittimen tilat <sup>(6)</sup>



- 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 konekäskyjä, myös etuoikeutettuja (esim. clear\_cache, iret)
  - voi viitata kaikkialle muistiin, myös käyttöjärjestelmän ytimeen (kernel)
    - voi käyttää (myös) suoria muistiosoitteita (PA)

31/10/2001 Copyright Teemu Kerola 2001 25

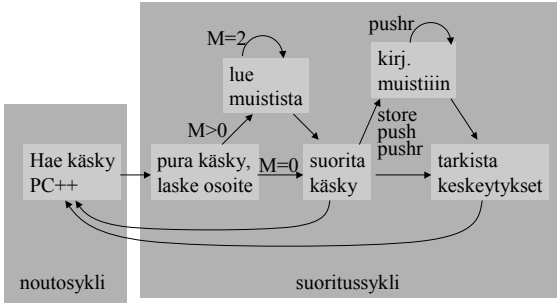
### Suorittimen tilan muuttaminen <sup>(6)</sup>



- Käyttäjätila → etuoikeutettu tila
  - keskeytys tai suora KJ:n palvelupyyntö (SVC käsky)
  - keskeytyskäsitteijä 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

31/10/2001 Copyright Teemu Kerola 2001 26

### TTK-91 Nouto- ja suoritusyksi vielä vähän tarkemmin



31/10/2001 Copyright Teemu Kerola 2001 27

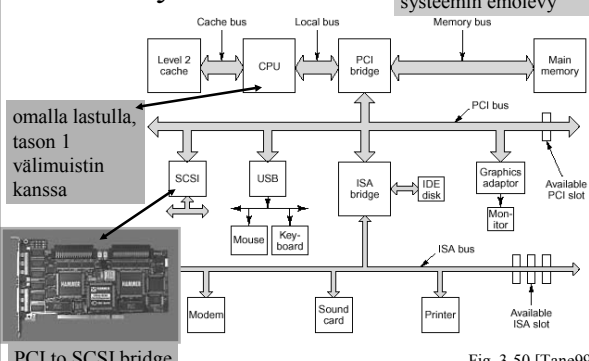
### Väylät <sup>(5)</sup>

- Tiedon siirtoa varten laitteistossa
- Yksi kirjoittaja kerrallaan
- Toteutettu johdinkimppuina
- Eri tasoilla
  - suorittimen sisällä "sisäinen väylä" (internal bus)
  - muistiväylä suorittimen ja muistin välillä (memory bus)
  - I/O-väylä muistiväylän ja I/O-laitteiden välillä (I/O bus)
- Useita eri tapoja yhdistellä edellä olevia

31/10/2001 Copyright Teemu Kerola 2001 28

### Väylähierarkia

Tyypillinen Pentium II systeemin emolevy



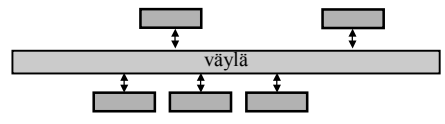
omalla lastulla, tason 1 välimuistin kanssa

PCI to SCSI bridge

Fig. 3-50 [Tane99]

31/10/2001 Copyright Teemu Kerola 2001 29

### Väylät <sup>(5)</sup>



- Kullakin laitteella oma osoite
- Yksi lähettää, kaikki kuulevat, vain "oikea" laite vastaanottaa
- Paljon erilaisia
- Lähellä suorittinta

Lisää tietoa? Tietokoneen rakenne -kurssi

31/10/2001 Copyright Teemu Kerola 2001 30

## TTK-91 koneen KOKSI simulaattori <sup>(6)</sup>

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

ks. suoritussyklin toteutus Koksissa (6 kalvoa)

31/10/2001

Copyright Teemu Kerola 2001

31

## TTK-91 käskyn suoritussykli <sup>(5)</sup>

```

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 % 65536   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;
lopetta suoritus jos SVC tai keskeytys  SR.O = ...
    
```

31/10/2001

Copyright Teemu Kerola 2001

32

## -- Luennon 5 loppu --

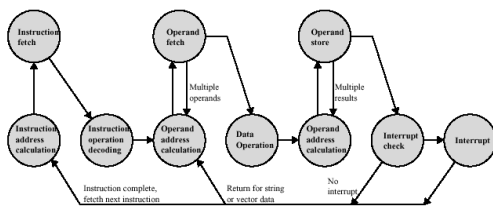


Figure 3.12 Instruction Cycle State Diagram, With Interrupts

[Stal99]

31/10/2001

Copyright Teemu Kerola 2001

33