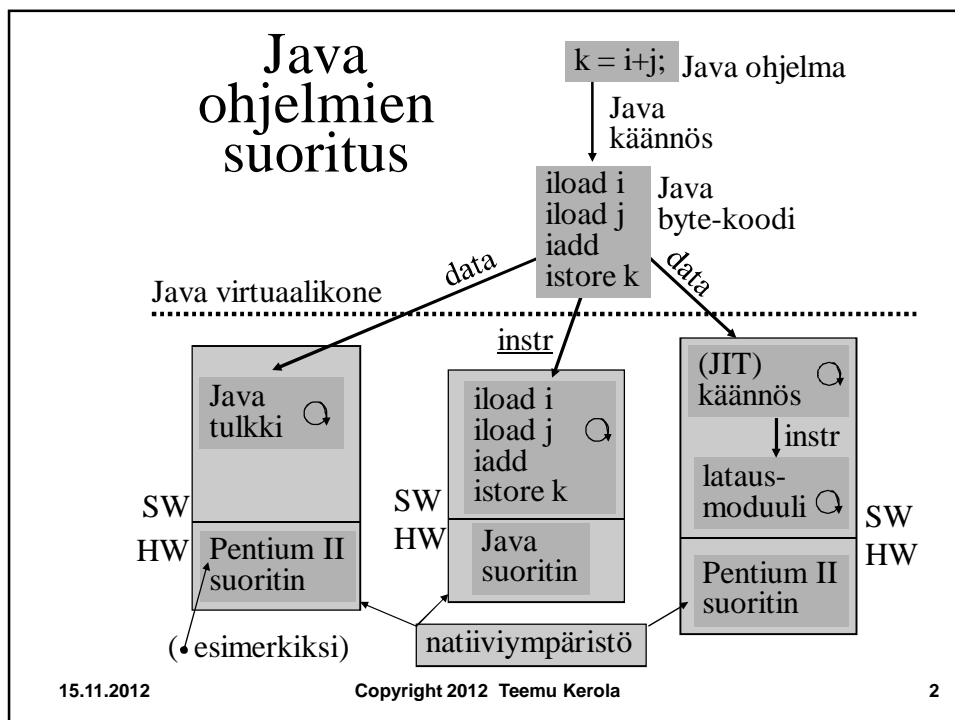


Luento 10 (verkkoluento 11) Tulkinta ja emulointi

Java ohjelman suoritus
Tavukoodi
JVM
Tulkinta
Java-suoritin
Käännös ja JIT-käännös



Java virtuaalikone (JVM)

- Hypoteettinen suoritin, toteutus eri tavoilla
- Geneerinen, sitä on ”helppo” simuloida kaikilla todellisilla suorittimilla
 - käännökseen tai tulkitsemiseen perustuva suoritus
- Useita säikeitä (thread) voi olla ”samanaikaisesti” suorituksessa
- Tietorakenteet
 - mm. virtuaalikoneen suorittimen ”rekisterit”
 - luodaan JVM:n käynnistämisen yhteydessä
- Käskyt
 - virtuaalikoneen suorittimen konekäskyt
 - 226 käskyä á 32 bittiä

15.11.2012

Copyright 2012 Teemu Kerola

3

JVM:n tietorakenteet

- JVM pino Figs 4-8, 4-9, 4-10 [Tane10]
 - kuten tavallinen AT-pino
 - koostuu useista *kehyksistä* (frames) (vrt. aktivointitietue) ja operandipinosta
 - käyttö: kehyksille ainoastaan push/pop operaatiot, operandipinon alkioille myös push/pop
 - ei tarvita yhtenäistä muistialuetta
 - allokoidaan keosta (heap)
 - toteutuksesta riippuen rajallinen tai dynaamisesti laajennettavissa
 - tila loppu ⇒ StackOverflowError, OutOfMemoryError

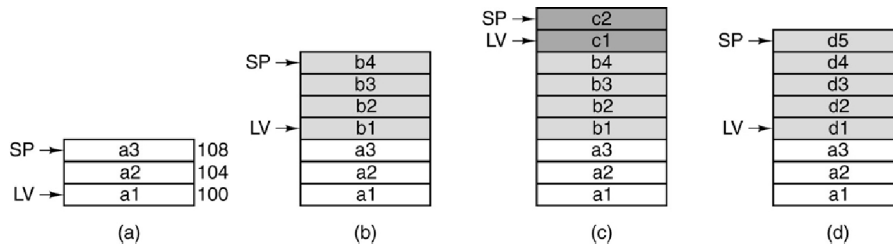
<http://java.sun.com/docs/books/vmspec/2nd-edition/html/VMSpecTOC.doc.html>

15.11.2012

Copyright 2012 Teemu Kerola

4

Fig 4-9 [Tane10]. Stacks (1)



Use of a stack for storing local variables.

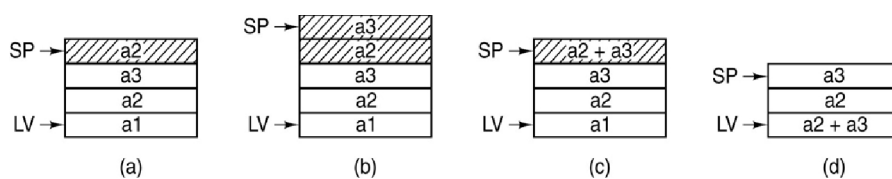
- a) While *A* is active.
- b) After *A* calls *B*.
- c) After *B* calls *C*.
- d) After *C* and *B* return and *A* calls *D*.

15.11.2012

Copyright 2012 Teemu Kerola

Keskustelevä 5

Fig 4-10 [Tane10]. Stacks (2)



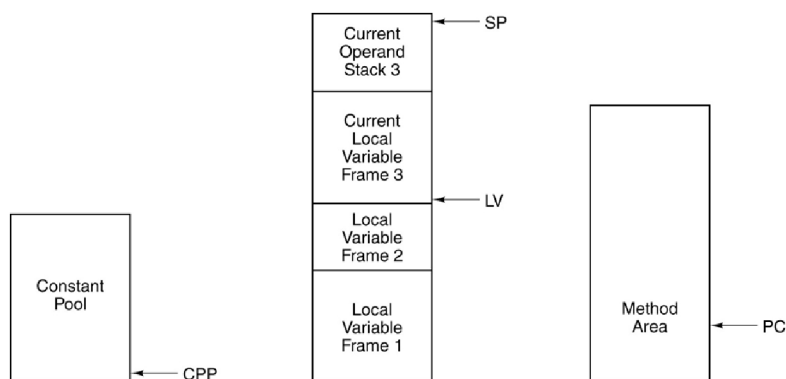
Use of an operand stack for doing an arithmetic computation.

15.11.2012

Copyright 2012 Teemu Kerola

6

Fig 4-10 [Tane10]



The various parts of the JVM memory.

15.11.2012

Copyright 2012 Teemu Kerola

7

JVM:n tietorakenteet (jatkuu)

- JVM keko (JVM heap)
 - yhteinen kaikille saman virtuaalikoneen säikeille
 - automaattinen roskienkeruu (garbage collector)
 - ei-käytössä (implisiittisesti ”vapautettu”) oleva muistialue palautetaan uusiokäyttöön (vapaaksi)
 - ei tarvita erikseen *free* operaatiota Java ohjelmassa
 - voi hidastaa suoritusta milloin vain (ongelma?)
 - toteutuksesta riippuen kiinteän kokoinen tai dynaamisesti laajennettavissa
 - ei tarvitse muodostaa yhtenäistä muistialuetta natiivijärjestelmän keossa
 - tila loppu ⇒ OutOfMemoryError

15.11.2012

Copyright 2012 Teemu Kerola

8

JVM:n tietorakenteet (jatkuu)

ks. Fig. 4-10 [Tane10]

- JVM metodialue (JVM Method Area)
 - yhteinen kaikille JVM säikeille
 - vastaa tavallista kääntäjän tuottamaa koodisegmenttiä
 - loogisesti osa JVM kekoa
 - toteutuksesta riippuen kiinteän kokoinen tai dynaamisesti laajennettavissa
 - tila loppu \Rightarrow OutOfMemoryError

15.11.2012

Copyright 2012 Teemu Kerola

9

JVM:n tietorakenteet (jatkuu)

ks. Fig. 4-10 [Tane10]

- Javan suoritusaikainen vakioallas (runtime constant pool)
 - joka luokalle (class) ja liittymälle (interface)
 - suoritusaikainen esitystapa tiedoston *class constant_pool* -taulukolle
 - vastaa vähän tavallista symbolitaulua
 - useita erilaisia vakioita (käännösaikaiset literaalit, suor. aikana ratkottavat attribuutit, ...)
 - talletetaan JVM metodialueelle
 - tila loppu \Rightarrow OutOfMemoryError

15.11.2012

Copyright 2012 Teemu Kerola

10

JVM:n tietorakenteet (jatkuu)

- Natiivimetodien pilot
(Native Method Stacks)
 - toteutus voi käyttää tavallisia pinoja ("C stacks") sellaisten natiivimetodien tukena, jota ei ole kirjoitettu Javalla
 - käytetään myös Java tulkin toteutuksessa
 - ei JVM toteutuksissa, joissa ei natiivimetoodeja
 - toteutuksesta riippuen kiinteän kokoinen tai dynaamisesti laajennettavissa
 - tila loppu ⇒ StackOverflowError, OutOfMemoryError

15.11.2012

Copyright 2012 Teemu Kerola

11

JVM:n tietorakenteet (jatkuu)

- JVM rekisterit ks. Fig. 4-10 [Tane10]
 - PC osoittaa johonkin JVM metodialueelle
 - CPP osoittaa vakioaltaaseen
 - LV on paikallisten muuttujien kantaosoite (vähän kuten FP ttk-91:ssä)
 - SP osoittaa JVM operandipinon huipulle
 - kaikki rekisterit implisiittisiä, niitä ei erikseen nimetä JVM konekäskyissä

15.11.2012

Copyright 2012 Teemu Kerola

12

JVM:n tietorakenteet (jatkuu)

ks. Figs 4-12, 4-13 [Tane10]

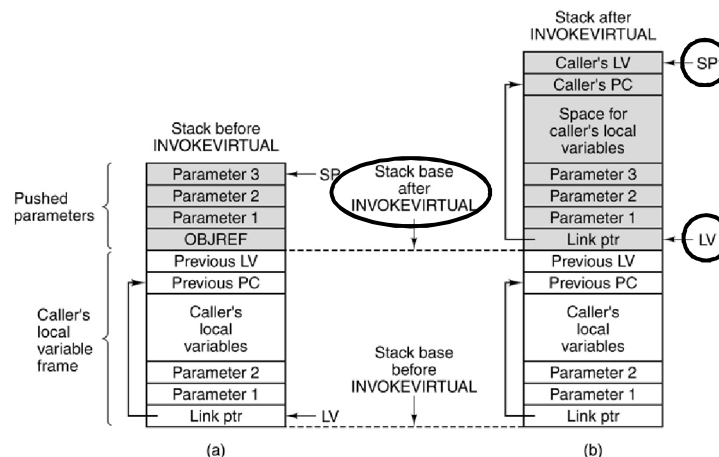
- JVM kehys (frame, raami)
 - talletetaan JVM pinoon, luodaan metodin kutsun yhteydessä, vapautetaan metodista poistuttaessa
 - paikalliset muuttujat
 - parametrit, paluuarvo ja välitulokset
 - dynaamisen linkityksen toteutusväline
 - keskeytysten toteutusväline

15.11.2012

Copyright 2012 Teemu Kerola

13

Tane10 Fig 4-12 The JVM Instruction Set (2)

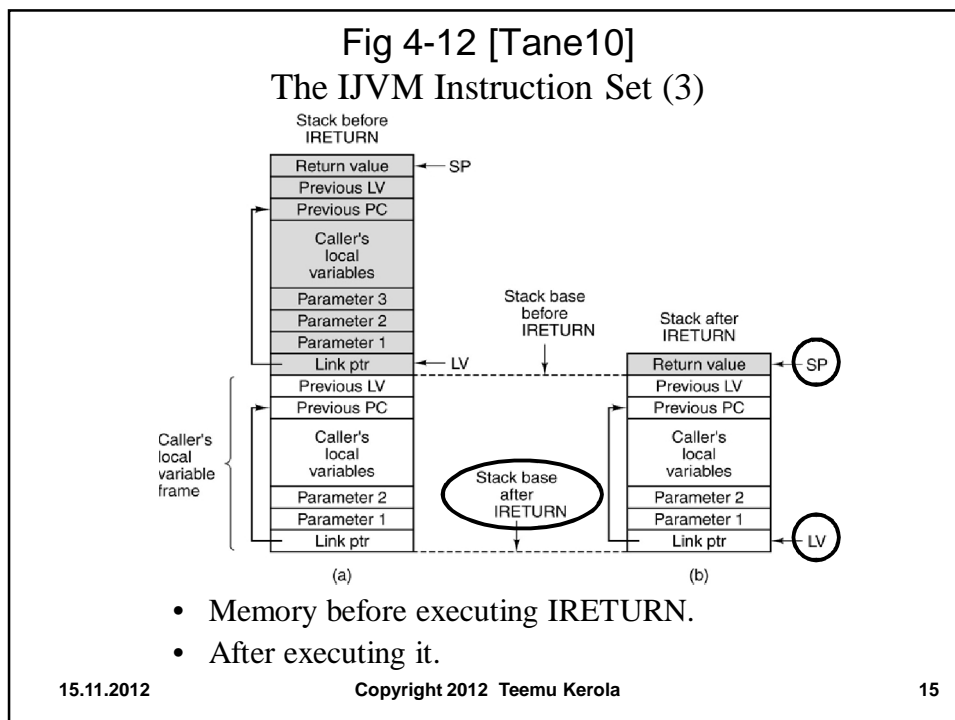


- Memory before executing INVOKEVIRTUAL.
- After executing it.

15.11.2012

Copyright 2012 Teemu Kerola

14



JVM kehyksen data ⁽¹⁾

- Paikalliset muuttujat sisältävä taulukko ks. Fig. 4-13 [Tane10]
 - viittaukset indeksoituna (0, 1, 2, ...) rekisterin LV suhteen
 - indeksit sanoina
 - kaksi sanaa vaativa muuttuja (long, double) sijoitetaan kahteen peräkkäiseen (32 bittiseen) sanaan
 - big-endian talletus
- Parametrit, paluarvon ja välitulokset sisältävä operandipino
 - SP osoittaa pinon huipulle
 - pinoarkkitehtuuri (vs. rekisteriarkkitehtuuri)

JVM:n tiedon osoitusmoodit ⁽⁴⁾

- Välitön operandi `iINC 2 (34)` `Java: xLoc += 34;`
- Indeksoitu `iINC (2)34` `tehollinen muistiosoite (LV) + 2`
- Pino-osoitus `iADD` `Java: a1 = a2+a3;`
 ks. Fig. 4-9 [Tane99] `korvaa pinon kaksi päällä olevaa kokonaislukua niiden summalla`
- Taulukko-osoitus pinon kautta `aLOAD 1`
`iLOAD 2`
`iALOAD`
`iSTORE 3` `Korvaa pinon pinnalla olevat taulukon alkuosoite ja indeksi k.o. taulukon alkiolla`
`Java: a = T[i];`

15.11.2012

Copyright 2012 Teemu Kerola

Keskustele 17

JVM käskyt

- Peruslaskutoimitukset `ks. Fig. 4-11 [Tane06]`
 - add, sub, mul, div, rem, neg
- Boolean
 - and, or, xor, shl, shr, ushr
- Pinon hallinta
 - dup, pop, swap, tauluk. luonti, esitystavan muutokset
- Load/Store
 - load, aload, store, astore, push-käskyt
- Vertailut
- Kontrollinsiirrot
- Muut

15.11.2012

Copyright 2012 Teemu Kerola

18

Fig 4-11 [Tane10]
The JVM Instruction Set (1)

Hex	Mnemonic	Meaning
0x10	BIPUSH <i>byte</i>	Push <i>byte</i> onto stack
0x59	DUP	Copy top word on stack and push onto stack
0xA7	GOTO <i>offset</i>	Unconditional branch
0x60	IADD	Pop two words from stack; push their sum
0x7E	IAND	Pop two words from stack; push Boolean AND
0x99	IFEQ <i>offset</i>	Pop word from stack and branch if it is zero
0x9B	IFLT <i>offset</i>	Pop word from stack and branch if it is less than zero
0x9F	IF_ICMPEQ <i>offset</i>	Pop two words from stack; branch if equal
0x64	IINC <i>varnum const</i>	Add a constant to a local variable
0x15	ILOAD <i>varnum</i>	Push local variable onto stack
0xB6	INVOKEVIRTUAL <i>disp</i>	Invoke a method
0x60	IOR	Pop two words from stack; push Boolean OR
0xAC	IRETURN	Return from method with integer value
0x36	ISTORE <i>varnum</i>	Pop word from stack and store in local variable
0x64	ISUB	Pop two words from stack; push their difference
0x13	LDC_W <i>index</i>	Push constant from constant pool onto stack
0x00	NOP	Do nothing
0x57	POP	Delete word on top of stack
0x5F	SWAP	Swap the two top words on the stack
0xC4	WIDE	Prefix instruction; next instruction has a 16-bit index

The JVM instruction set. The operands *byte*, *const*, and *varnum* are 1 byte. The operands *disp*, *index*, and *offset* are 2 bytes.

15.11.2012

Copyright 2012 Teemu Kerola

Keskusteleva 19

Fig 4-11 [Tane10], Compiling Java to JVM (1)

<code>i = j + k;</code>	1	ILOAD j	// i = j + k	0x15 0x02
<code>if (i == 3)</code>	2	ILOAD k		0x15 0x03
<code> k = 0;</code>	3	IADD		0x60
<code>else</code>	4	ISTORE i		0x36 0x01
<code> j = j - 1;</code>	5	ILOAD i	// if (i == 3)	0x15 0x01
	6	BIPUSH 3		0x10 0x03
	7	IF_ICMPEQ L1		0x9F 0x00 0x0D
	8	ILOAD j	// j = j - 1	0x15 0x02
	9	BIPUSH 1		0x10 0x01
	10	ISUB		0x64
	11	ISTORE j		0x36 0x02
	12	GOTO L2		0xA7 0x00 0x07
	13	L1: BIPUSH 0	// k = 0	0x10 0x00
	14	ISTORE k		0x36 0x03
	15	L2:		

(a)

(b)

(c)

- A Java fragment.
- The corresponding Java assembly language.
- The JVM program in hexadecimal.

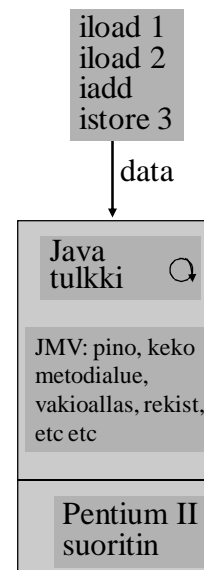
15.11.2012

Copyright 2012 Teemu Kerola

20

Java tulkki

- Emuloi JVM konekielen käskyjä (byte-koodia)
- Yksi (byte-koodi) käsky kerrallaan
- JVM rekisterit ja muistialueet emuloitu tulkin tietorakenteina muistissa
 - vrt. KOKSI ja TTK-91
- Hidasta, mutta joustavaa



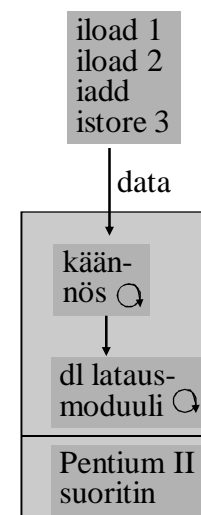
15.11.2012

Copyright 2012 Teemu Kerola

21

Käännös natiivikoneelle

- (a) Käännetään tavukoodi suoraan natiivikoneen konekielelle ja suoritetaan se normaalin ohjelman tapaan
- (b) Käännetään tavukoodi ensin korkean tason kielelle (esim C), joka sitten käännetään natiivikoneen konekielelle
 - alkuosa riittää tehdä kerran
 - loppuosa on jo valmiina yleensä
- Ongelma: ei dynaamista linkitystä



15.11.2012

Copyright 2012 Teemu Kerola

22

Java JIT käänнос

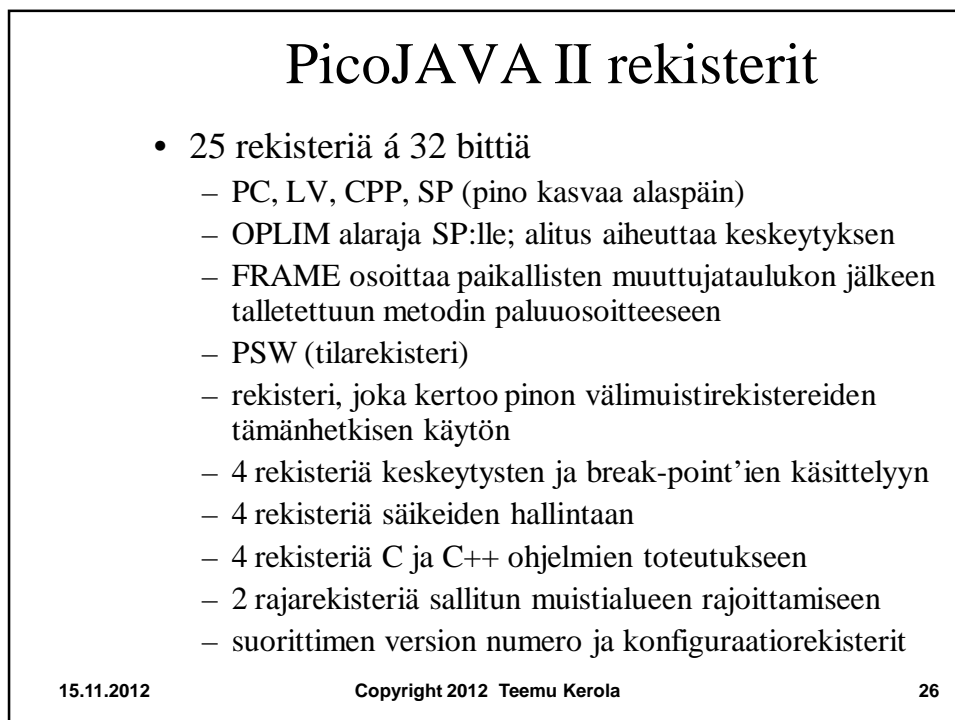
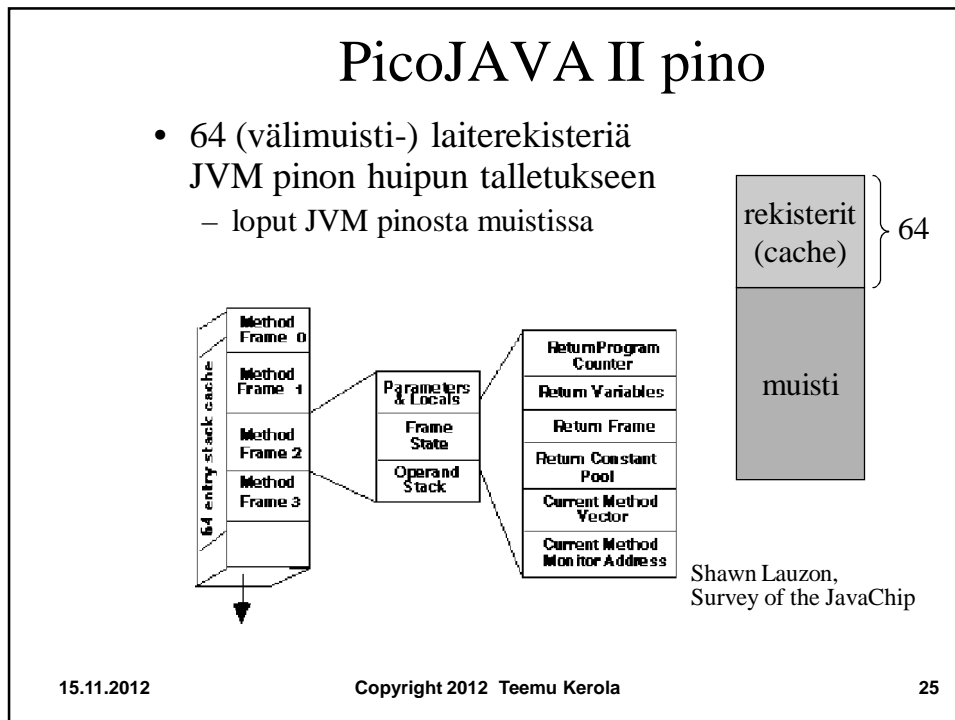
- JIT = Just-in-Time
- Emulointi ja/tai käänнос tilanteesta riippuen
- Käännä luokka natiivikonekielelle dynaamisesti linkitettäväksi moduuliksi, mutta vasta juuri ennen luokan metodin kutsua
- Tarvitsee paljon muistia
- Voi hidastaa suoritusta, jos käännökseen menee enemmän aikaa kuin tulkitsemiseen
 - käänнос vasta 2. kutsukerralla?
- JVM rekisterit ja muistialueet emuloitu tulkin tietorakenteina, joita natiivikoodi myös käyttää

15.11.2012 Copyright 2012 Teemu Kerola 23

Java suoritin: Sun PicoJAVA II

- Suorittimen määrittely, jonka mukaisessa koneessa byte-koodi -muodossa olevia ohjelmia voidaan sellaisenaan suorittaa
- Valinnainen välimuisti ja liukulukusuoritin
- Kaikki 226 JVM konekäskyä
 - jotkut käskyt toteutettu aliohjelmilla, jotka aktivoidaan keskeytyskäsittelemekanismin avulla
- Myös 115 muuta konekäskyä käyttöjärjestelmän ja muiden ohjelmointikielten toteuttamiseksi

15.11.2012 Copyright 2012 Teemu Kerola 24



PicoJAVA ylim. käskyt

- Read/write ylimääräisille rekistereille
- Osoittimien manipulointikäskyt
 - mitä tahansa muistialuetta voidaan suoraan lukea/kirjoittaa
 - tarvitaan C/C++ varten
- C/C++ aliohjelmien kutsu ja paluukäskyt
- Natiivi HW manipulointi
 - tyhjä välimuisti (osittain? kokonaan?), ...
- Muut käskyt
 - power on/off, ...

15.11.2012

Copyright 2012 Teemu Kerola

27

PicoJAVA toteutuksia ⁽²⁾

- Sun microJAVA 701
 - valinnainen välimuisti
 - oma muistiväylä
 - PCI väylä muille laitteille
 - 16 ohjelmitavaa I/O johdinta
 - näppäimet, LEDit, ...
 - 3 ohjelmitavaa ajastinta (⇒ kellolaitekeskeytykset)
 - suunnattu halpuihin kannettaviin laitteisiin (kämment mikro, PDA - Personal Digital Assistant)
- Sun ultraJAVA
 - nopeampi, parempi, kalliimpi, ...
 - suunnattu grafiikka- ja multimediasovelluksiin

15.11.2012

Copyright 2012 Teemu Kerola

28

Muita Java-suorittimia

- JEM (Rockwell Collins)
- PSC1000 (Patriot Scientific)
 - dSys (Saksa), lääketieteellisiä laitteita
- MJ501 (LG Semicon)
 - TV, älykortit
- JSR-001, Real-Time Specification for Java (Java Community Process, "Sun Microsystems")
 - aJile: aJ-80, aJ-100, älykkäät liikkuvat laitteet



15.11.2012

Copyright 2012 Teemu Kerola

29

Sun MAJC

- MAJC - Microprocessor Architecture for Java Computing
 - suoritinarkkitehtuurin määrittely
 - tavoitteena suuri nopeus Java, C ja C++ sovelluksille
 - suunnattu multimediasovelluksiin verkossa
 - tukee hyvin JIT-käännöstä

15.11.2012

Copyright 2012 Teemu Kerola

30

MAJC toteutus: MAJC 5200

- 1-4 suoritinta (2 suorittimen lastu, v. 1999)
- Useiden (peräkkäin kutsuttavien) metodien samanaikainen suoritus eri suorittimilla
 - ennakoivalle (speculative) suoritukselle oma kasa
 - peruutus (rollback), jos ennakoitu suoritus meni pieleen
- 4 säiettä suorituksessa per suoritin
 - säikeen vaihto nopeampaa kuin muistista luku!
 - laiterekisterit 4:lle säikeelle! (hyper-threaded processor)
 - välimuistin hudin aikana suoritetaan muita säikeitä
- VLIW arkkitehtuuri – 4 konekäskyä samanaikaisesti
- Suunnattu interaktiiviseen TV:hen, virtuaalitodellisuussovelluksiin, ...

15.11.2012

Copyright 2012 Teemu Kerola

31

15.11.2012

Copyright 2012 Teemu Kerola

32

TTK-91 Emulointi

- TTK-91 konekielen emulointi
- Tietokoneen komponentti
- Yksi käsky kerrallaan
- TTK-91 koneen rekisterit ja muisti emuloitu tulkin (Titokone) tietorakenteina

```
load R1, 234
add R1, =5
mul R1, R2
```

data

TTK-91
emulaat-
tori

Pentium II
suoritin

ks. simulaattorin koodi, proj. Titokone:

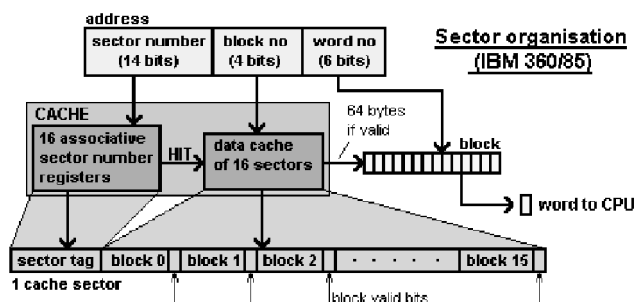
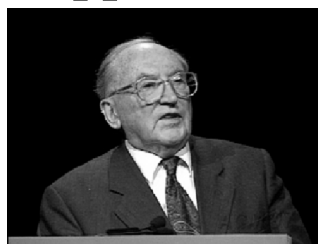
15.11.2012

Copyright 2012 Teemu Kerola

Keskustele 33

-- Luennon 10 loppu --

- Välimuisti
(1965, Maurice Wilkes)
 - IBM S/360 Model 85
 - 1968
 - 256 lohkoa á 64 tavua



15.11.2012

Copyright 2012 Teemu Kerola

34