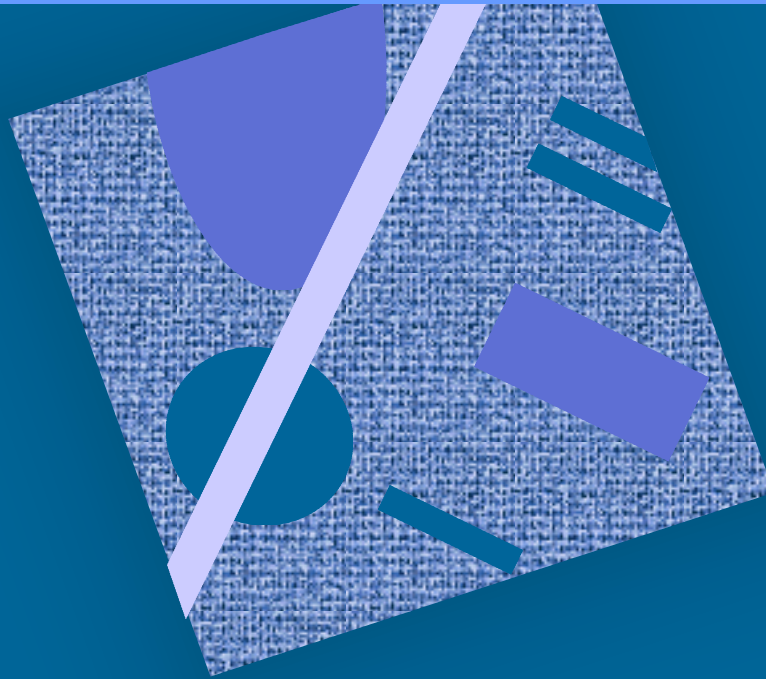


## Tiedon esitysmuodot



Lukujärjestelmät

Kokonaisluvut

Liukuluvut

Merkit, merkkijonot

Totuusarvot

Kuvat, äänet, hajut(?)

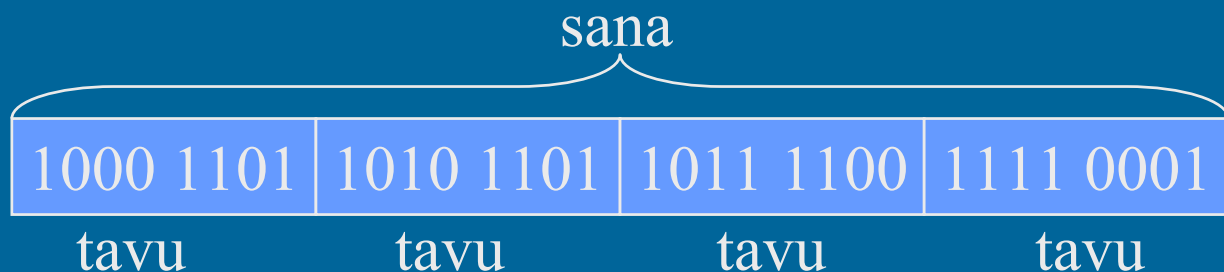
Ohjelman esitysmuoto

# Tiedon tyypit

- Kommunikointi ihmisen kanssa
  - kuva, ääni, merkit, ...
- Laitteiston sisäinen talletus
  - kuvaformatit, ääniformatit, pakkausstandardit, ...
  - kokonaisluvut, liukuluvut, merkit, merkistöt
  - ohjelmat
- Suorittimen omana lajinaan ymmärtämät tyypit
  - on olemassa konekäskyjä tälle tietotyypille
  - kokonaisluvut
  - liukuluvut (useimmat suorittimet nykyään)
  - totuusarvot (jotkut suorittimet)
  - merkit (jotkut suorittimet)
  - konekäskyt

# Tiedon esitys laitteistossa (4)

- Kaikki tieto koneessa on binääribitteinä (0 tai 1)
  - binäärijärjestelmän numerot: 0, 1
  - helppo toteuttaa piireillä
  - helppo suunnitella logiikkaa Boolean algebran avulla
- Muisti jaettu tasapituisiin sanoihin (word)
  - sana = word = 32 bittiä (16 bittiä, 64 bittiä, ...)
- Usein sana on jaettu tasapituisiin 8-bittisiin tavuihin (byte)



# Tiedon esitys laitteistossa (4)

- Tietoa siirretään muistiväylää pitkin sanoina
  - joskus useampi kuin yksi sana kerrallaan (lohko)
- Prosessorin rekisterit ovat yleensä yhden tai kahden sanan mittaisia
  - 1 sana: kokonaisluku, pieni liukuluku
  - 1 sana: 1 merkki tai 4 merkkiä
  - 2 sanaa: pitkä kokonaisluku, iso liukuluku

# Tiedon esitys <sup>(7)</sup>

- Kysymys: miten esittää eri tyyppisiä tietoja?
- Vastaus: koodataan ne biteiksi
  - kaikki tieto on koneessa bitteinä
- Kaikelle käsitellylle tiedolle on omat koodausmenetelmänsä
  - kaikkia koodausmenetelmiä ei ole standardoitu
  - samalla tietotyypille voi olla useita koodausmenetelmiä
    - kokonaisluvut, liukuluvut, merkit, merkkijonot, kuvat, ...
  - ongelma: ymmärtävätkö koneet toisiaan?
    - tiedon esitysmuotoa voidaan joutua muuttamaan kun tietoa siirretään koneelta toiselle

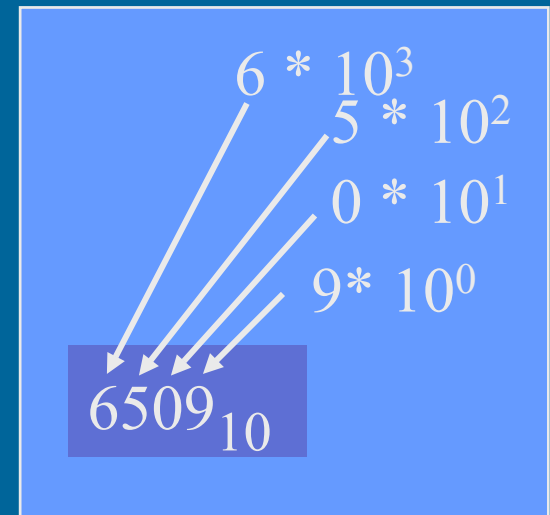
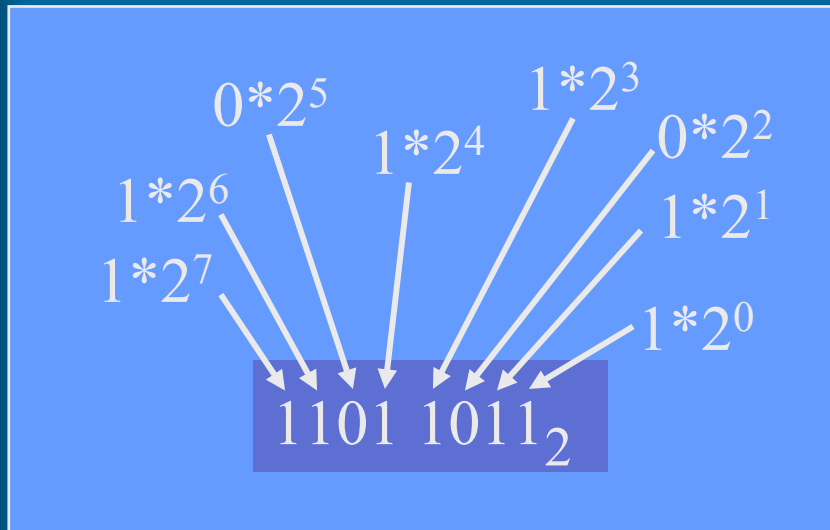
# Suorittimen ymmärtämä tieto (9)

- Kaikki tieto koneessa on koodattuna biteiksi
- Muistissa voidaan esittää kaikki tieto sovitulla esitystavalla (koodauksella)
- Suoritin osaa tehdä operaatioita joillakin esitystavoilla koodatuille tiedoille
  - kokonaisluvut ja liukuluvut (aina)
  - totuusarvot, merkit ja merkkijonot (joskus)
  - kuvat ja äänet (ei yleensä ellei erikoistunut suoritin)
  - hajut (ei vielä)
- Muiden tietojen käsittely tapahtuu ohjelmallisesti
  - esim. merkkejä voidaan käsitellä kokonaislukuoperaatioilla ja aliohjelmilla

TTK-91:  
kokonaisluvut

# Binäärijärjestelmä (2)

- Kantaluku 2, numerot 0 ja 1
  - numeroiden painoarvot oikealta vasemmalle:  
 $1=2^0$ ,  $2=2^1$ ,  $4=2^2$ ,  $8=2^3$ ,  $16=2^4$ ,  $32=2^5$ , ...
  - kymmenjärjestelmässä painoarvot ovat  
 $1=10^0$ ,  $10=10^1$ ,  $100=10^2$ ,  $1000=10^3$ , ...



# Binäärilukuesimerkkejä

$$\begin{array}{cccc} +32 & +16 & +8 & \\ \swarrow & \swarrow & \swarrow & \swarrow \\ 0011 & 1001 & = ? & = 57_{10} \end{array}$$

$$\begin{array}{cccc} & & +2 & \\ & & \swarrow & \swarrow \\ 0000 & 0011 & = ? & = 3_{10} \end{array}$$

$$\begin{array}{cccc} +64 & +16 & +4 & \\ \swarrow & \swarrow & \swarrow & \swarrow \\ 0101 & 0101 & = ? & = 85_{10} \end{array}$$



# Bonäärilukujen laskutoimitukset

+	0	1
0	0	1
1	1	10

$$\begin{array}{r}
 11 \\
 101101 \\
 + 1100 \\
 \hline
 111001
 \end{array}$$

$$\begin{array}{r}
 45 \\
 + 12 \\
 \hline
 57
 \end{array}$$

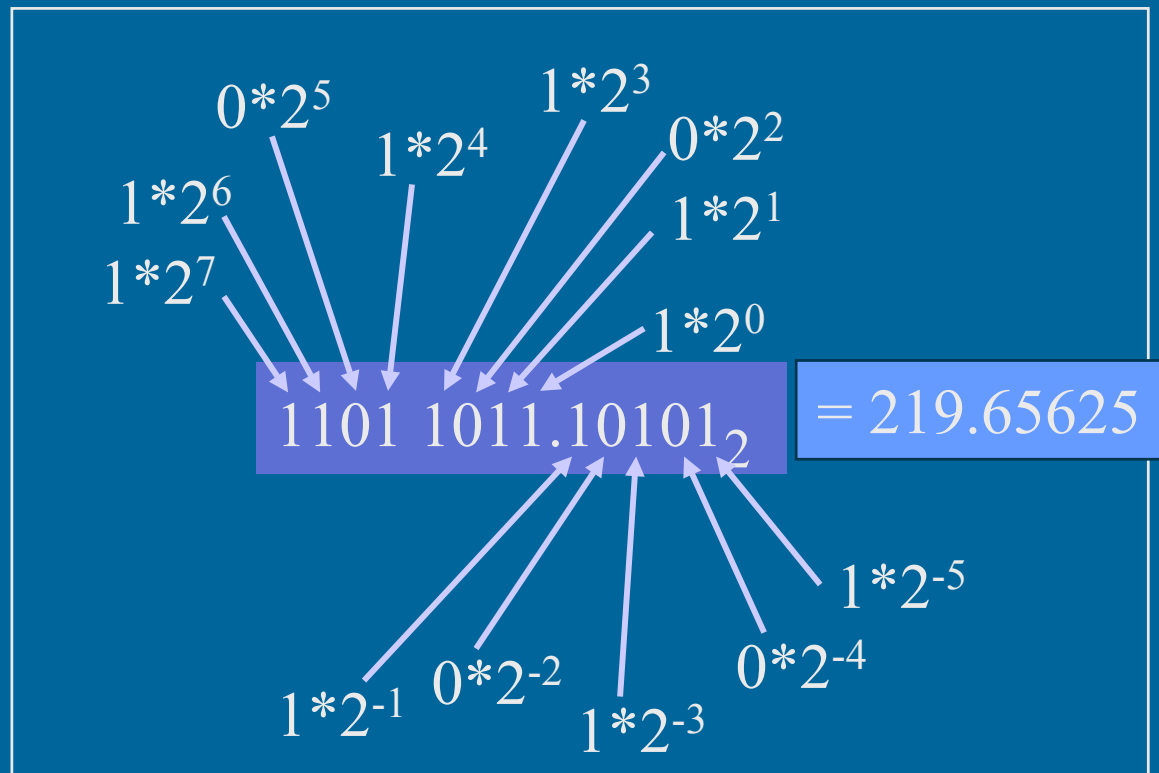
*	0	1
0	0	0
1	0	1

$$\begin{array}{r}
 101 \\
 * 101 \\
 \hline
 101 \\
 + 101 \\
 \hline
 11001
 \end{array}$$

$$\begin{array}{r}
 5 \\
 * 5 \\
 \hline
 = 25
 \end{array}$$

# Binääripiste (2)

- Binääriluvuilla voi olla myös binääriosaa (vrt. desimaaliosa)



# Binääripiste-esimerkkejä

$+4$   $+1$   $+0.5 = 2^{-1}$   
 $+0.125 = 2^{-3}$

$0101.101 = ?$   $= 5.625_{10}$

$+4$   $+2$   $+0.125 = 2^{-3}$   
 $+0.0625 = 2^{-4}$

$0110.0011 = ?$   $= 6.1875_{10}$

$0110.0010 = ?$   $= 6.1250_{10}$

$?? = 6.1500_{10}$

?

# Muunnokset lukujärjestelmien välillä <sup>(5)</sup>

- 2-järjestelmä  $\Rightarrow$  10-järjestelmä
  - esitettiin jo edellä
- 10-järjestelmä  $\Rightarrow$  2-järjestelmä
  - kokonaisosa ja desimaaliosa erikseen
  - kokonaisosa:
    - jaa toistuvasti 2:lla, kunnes 0 jäljellä
    - ota jakojäännökset käännettyssä järjestyksessä

# 10-järj $\Rightarrow$ 2-järj kokonaislukuesimerkki <sup>(11)</sup>

$$57_{10} = ?$$

$$57/2 = 28 \text{ jää } 1$$

$$28/2 = 14 \text{ jää } 0$$

$$14/2 = 7 \text{ jää } 0$$

$$7/2 = 3 \text{ jää } 1$$

$$3/2 = 1 \text{ jää } 1$$

$$1/2 = 0 \text{ jää } 1$$

loppu

$$= 11\ 1001_2$$

$$= 0011\ 1001_2$$

# 10-järj $\Rightarrow$ 2-järj desimaaliosa $\Rightarrow$ binääriosaa

- Kerrotaan toistuvasti desimaaliluvun desimaaliosa 2:lla, kunnes
  - desimaaliosa = 0 (tarkka binääriesitys)
  - tarpeeksi numeroita haluttuun tarkkuuteen
- Tulos saadaan ottamalla saatujen desimaalilukujen kokonaisosat (0 tai 1) lasketussa järjestyksessä

# 10-järj $\Rightarrow$ 2-järj binääriosaesimerkki

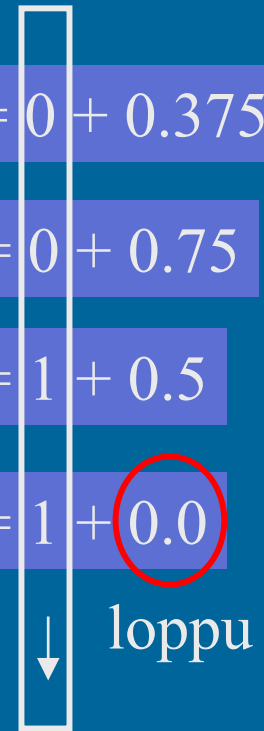
$$0.1875_{10} = ?$$

$$2 * 0.1875 = 0.375 = 0 + 0.375$$

$$2 * 0.375 = 0.75 = 0 + 0.75$$

$$2 * 0.75 = 1.5 = 1 + 0.5$$

$$2 * 0.5 = 1.0 = 1 + 0.0$$



$$= 0.0011_2$$

$$= 0.001100000000000000000000_2$$

# Heksadesimaaliesitys <sup>(6)</sup>

- Binäärilukuja käyttö on tarpeellista, mutta niitä on ikävä kirjoittaa
  - liikaa numeroita
- Kirjoitetaan ne 16-järjestelmässä eli heksadesimaalijärjestelmässä
- 4 bittiä vastaa aina yhtä 16-järjestelmän numeroa
- Yksi 16-järjestelmän numero vastaa aina 4 bittiä
- 16-järjestelmän numerot ovat:  
0,1,2,3,4,5,6,7,8,9, A, B, C, D, E ja F

10	11	12	13	14	15
----	----	----	----	----	----



# Heksadesimaaliesimerkkejä <sup>(11)</sup>

binääri: 0100 0111 1001 1010 1111

16-järj: 4 7 9 A F = 479AF<sub>16</sub>

= 0004 79AF<sub>16</sub> = 0x 479AF

16-järj: 120ADF<sub>16</sub> 1 2 0 A D F

binääri: 0001 0010 0000 1010 1101 1111

# Oktaaliesimerkkejä <sup>(13)</sup>

Numerot: 0, 1, 2, 3, 4, 5, 6, 7

binääri: 01 000 111 100 110 101 111

8-järj: 1 0 7 4 6 5 7 = 1074657<sub>8</sub>

= 0001074657<sub>8</sub> = 01074657

8-järj: 120371<sub>8</sub>

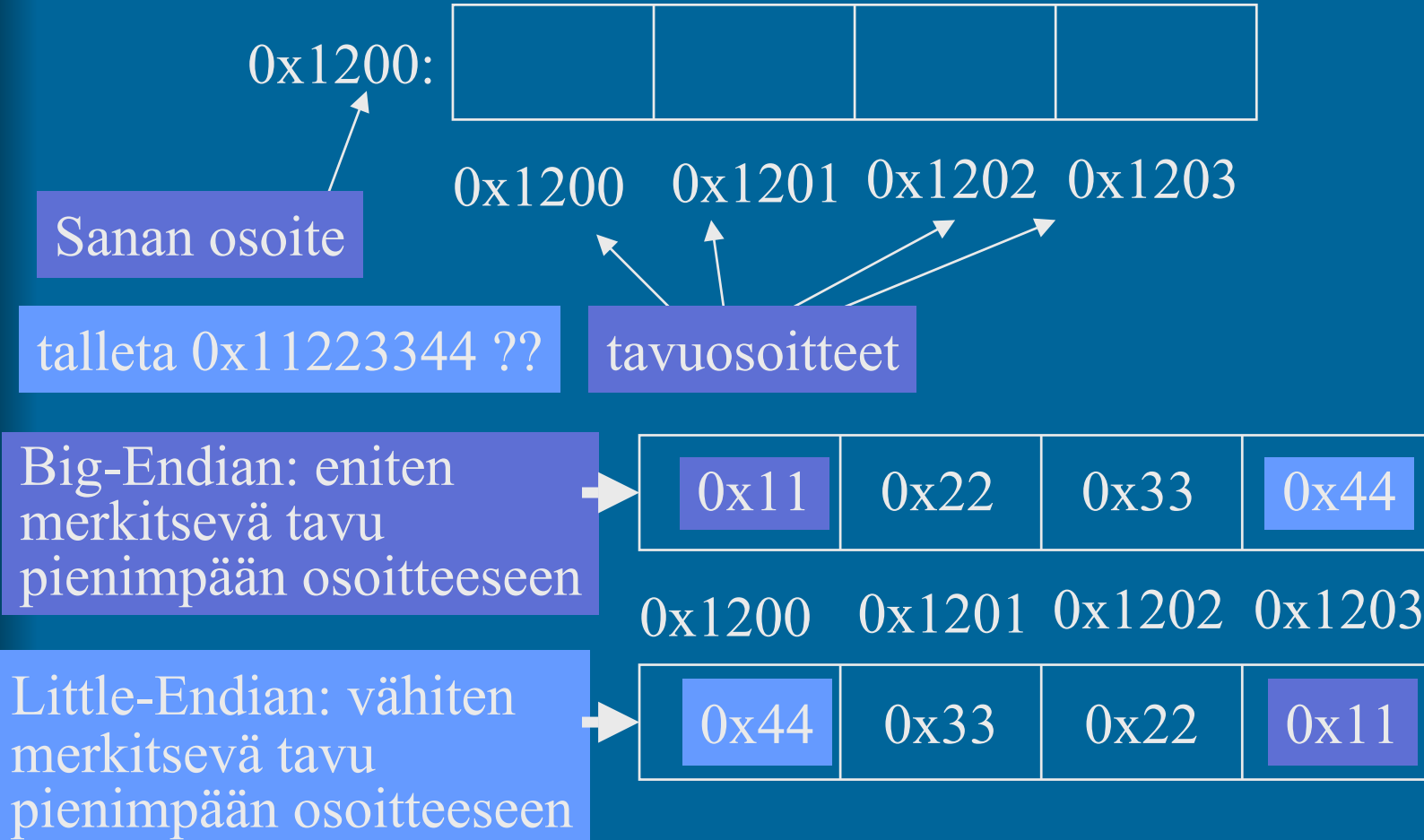
1 2 0 3 7 1

binääri:

001 010 000 011 111 001

# Big vs. Little Endian (3)

- Miten monitavuiset arvot talletetaan?



# Big vs. Little Endian <sup>(5)</sup>

- Monitavuisen tiedon (sana-) osoite on sama molemmissa tapauksissa
- Tavujen sisäinen järjestys vaihtelee
- Suorittimen suunnittelija päättää
  - Matematiikkapiirien tulee tietää miten luvut esitetty
  - Täytyy ottaa huomioon siirrettäessä tietoa verkon yli
- Power-PC: bi-endian - molemmat moodit käytössä
  - voidaan valita ohjelmakohtaisesti
  - etuoikeutetussa tilassa voidaan vielä valita erikseen
  - suoritin osaa laskea kummallakin tavalla talletetuilla luvuilla

TTK-91: big-endian

# Kokonaislukujen esitysmuoto (8)

Kaikki esitetty biteillä 0 ja 1  
ei etumerkkejä  
ei desimaalipistettä

$$57 = 0011\ 1001$$

+32 +16 +8 +1

- Etumerkittömät kokonaisluvut helppoja
- Positiiviset luvut helppoja
  - normaali binäärilukuesitys

- **Negatiiviset luvut**

- -57, -256 ???
- Eräs tapa

sign bit = MSB  
= most significant bit

$$-57 = \underline{1}011\ 1001$$

$$+57 = \underline{0}011\ 1001$$

# Negatiiviset luvut (8)

$$57 = 0011\ 1001$$

- Etumerkkibitti erikseen

sign bit = MSB  
= most significant bit

$$-57 = \underline{1}011\ 1001$$

talletusmuoto

- Yhden komplementtiesitys



komplementti

$$-57 = \underline{1}100\ 0110$$

- Kahden komplementtiesitys

$$-57 = \underline{1}100\ 0111$$

+1

“sign” bit

komplementit

- Vakiolisäys

- lisää 127 ( $= 2^{**}8-1$ )
- tai jokin muu luku

$$-57 = \underline{0}100\ 0110$$

$$-57 + 127 = 70$$

# Yhden ja kahden komplementti

- Yhden komplementti:

- ykköset nolliksi ja nollat ykkösiksi

ones complement:  $-0 = 1111\ 1111$   
 $-1 = 0000\ 0000$   
 $-57 = 1100\ 0110$

- Kahden komplementti

- ykköset nolliksi ja nollat ykkösiksi
- ja lisätään vielä ykkönen

```
57 = 0011 1001
-57 = 1100 0110
      +      1
=====
      1100 0111
```

$+2 = 0000\ 0010$   
 $+1 = 0000\ 0001$   
 $+0 = 0000\ 0000$   
 $-0 = 0000\ 0000$   
 $-1 = 1111\ 1111$   
 $-2 = 1111\ 1110$

# Kahden komplementti

- Useimmiten käytössä  $+57 = 0011\ 1001$
- Etu: vain yksi nolla
  - Yhden komplementissa 2 nollaa
    - $+0 = 0000\ 0000$   $-0 = 1111\ 1111$
- Helpot muunnokset: arvo  $\leftrightarrow$  esitysmuoto
  - Miten arvo  $-75$  esitetään?
    - $75 = 0100\ 1011 \Rightarrow$
    - $-75 = 1011\ 0100 + 0000\ 0001 = 1011\ 0101$
  - Mitä arvoa esitysmuoto  $1100\ 1100$  tarkoittaa?
    - $1100\ 1100 - 1 = 1100\ 1011 \Rightarrow 0011\ 0100 = 52$   
(eli  $52 = 0011\ 0100$ , kahden komplementtimuodossa  $= 1100\ 1011 + 1 = 1100\ 1100$ )
    - tai:  $-(1100\ 1100) = 0011\ 0011 + 1 = 0011\ 0100 = 52$



- Kahden komplementtimuodossa vähennyslasku korvautuu yhteenlaskulla

$$- 57 - 52 = 57 + (-52);$$

$$-57 + 52$$

$$\begin{array}{r}
 57: 0011\ 1001 \\
 -52: \underline{1100\ 1100} \ + \\
 1\ 0000\ 0101 = 5
 \end{array}$$

$$\begin{array}{r}
 -57: 1100\ 0111 \\
 52: \underline{0011\ 0100} \ + \\
 1111\ 1011 = -5
 \end{array}$$

# Liukuluvut

- Tietokoneessa ei ole realilukuja tai rationaalilukuja (matemaattiset käsitteet)
- Aina rajallinen esityksen tarkkuus
  - lukuja  $\pi$ ,  $\text{SQRT}(2)$  tai  $1/3$  ei voi esittää tarkasti
  - luvut 1.000000000 ja luvut 1.000000001 ovat yhtäsuuria (joissakin esityksissä)
- Yleinen realilukuja vastaava esitysmuoto on liukulukuesitysmuoto float, double, real
  - 32 bittiä, noin 7-8 desimaalinumeron tarkkuus
  - 64 bittiä, noin 16-17 desimaalinumeron tarkkuus

# Liukulukujen esitys (4)

$$+1.23 = +1.23 * 10^0$$

$$+123.0 = +1.23 * 10^2$$

$$+0.123 = +1.23 * 10^{-1}$$

$$-0.000\ 000\ 000\ 123 = -1.23 * 10^{-10}$$

$$+123\ 000\ 000\ 000\ 000 = +1.23 * 10^{14}$$

“+”	“14”	“1.23”
-----	------	--------

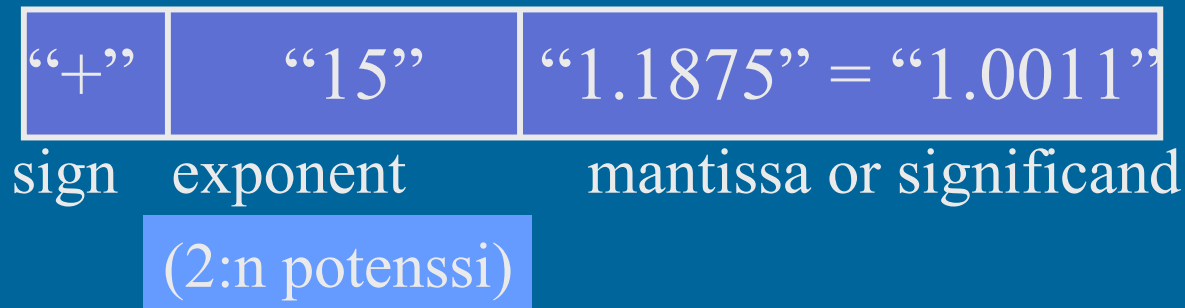
sign exponent

mantissa or significand

(exponentti)

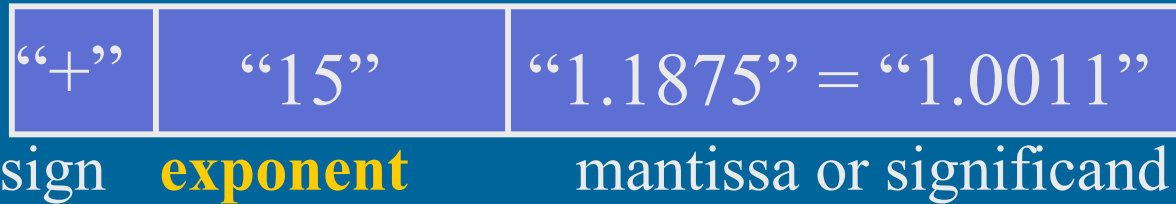
(mantissa)

# IEEE 32-bit Floating Point Standard <sup>(3)</sup>



- Etumerkki
  - 1 bitti etumerkille, 1  $\Rightarrow$  “-”, 0  $\Rightarrow$  “+”
  - etumerkkibitti  $S \Rightarrow$  etumerkin arvo =  $(-1)^S$

# IEEE 32-bit FP Standard



8 bittiä eksponentille, lisättynä 127:llä

(biased form)

exponent = 5

store

$5+127 = 132 = 1000\ 0100$

exponent = -1

store

$-1+127 = 126 = 0111\ 1110$

exponent = 0

store

$0+127 = 127 = 0111\ 1111$

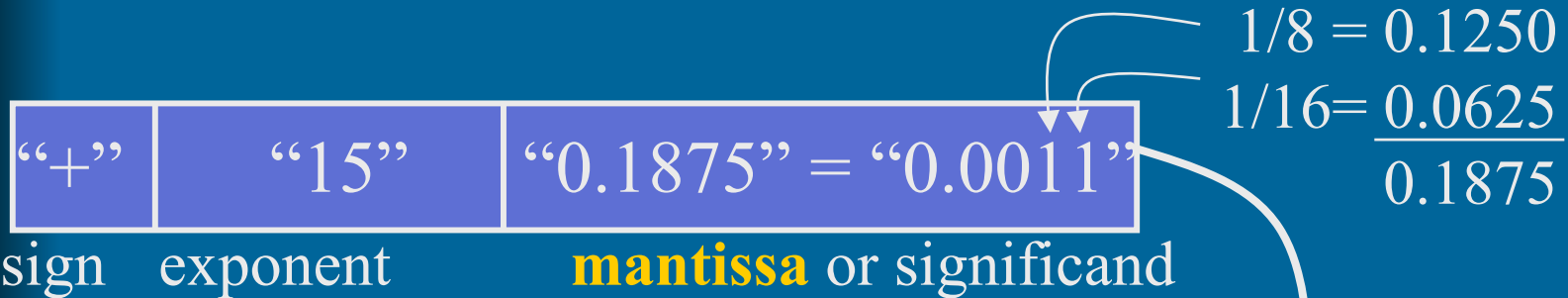
esitysmuodot 0 and 255 erikoistapauksia: hyvin pieni, ääretön, ei luk

– talletettu arvoalue: **1 - 254**  $\Rightarrow$  todellinen arvoalue: **-126 - +127**

Entä kun eksponentti on **15**?

$127 + 15 = 142$   
 $= 1000\ 1110$

# IEEE 32-bit FP Standard (7)



- 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, piilobitti)

mantissa eksponentti

0.0011      “15”

1.1000      “12”

1000      “12”

24 bitin mantissa!

# IEEE 32-bit FP Values <sup>(8)</sup>

$$23.0 = +10111.0 * 2^0 = +1.0111 * 2^4 = ?$$

$$4+127=131$$

0	1000 0011	011 1000 0000 0000 0000 0000
---	-----------	------------------------------

sign            exponent            mantissa or significand  
1 bit            8 bits                            23 bits

$$1.0 = +1.0000 * 2^0 = ?$$

$$0+127 = 127$$

0	0111 1111	000 0000 0000 0000 0000 0000
---	-----------	------------------------------

sign            exponent            mantissa or significand  
1 bit            8 bits                            23 bits

# IEEE 32-bit FP Values (6)

0	1000 0000	111 1000 0000 0000 0000 0000
---	-----------	------------------------------

sign  
1 bit

exponent  
8 bits

mantissa or significand  
23 bits

X = ?

$$X = (-1)^0 * 1.1111 * 2^{(128-127)}$$

$$= + 1.1111_2 * 2$$

$$= (1 + 1/2 + 1/4 + 1/8 + 1/16) * 2$$

$$= (1 + 0.5 + 0.25 + 0.125 + 0.0625) * 2$$

$$= 1.9375 * 2$$

$$= 3.875$$



# Merkit <sup>(4)</sup>

- Yleensä 1 tavu per merkki
- ASCII, 7 bittiä/merkki (+ tark. bitti?)  
`'A' = 0x41, 'a' = 0x61, LF = 0x0A`
- EBCDIC, 8 bittiä/merkki
- ISO/IEC 8859-15 ('Latin-9'),
  - 8-bittiä/merkki, 256 eri merkkiä käytössä
  - mukana myös ä, ö, š, €

Lisää tietoa: kts.

<http://www.tieke.fi/edisty/edis699/stand699.htm>

# UCS ja Unicode <sup>(5)</sup>

- UCS - Universal Character Set
- Samat merkistöt, eri standardit
- 2 tavua eli 16 bittiä per merkki
  - 65536 merkkiä koko maailmassa käytössä oleville n. 200000 symbolille
- Kontrollimerkit
  - 0x0000-001F and 0x0080-009F
  - 0x007F = DELETE, 0x0020 = SPACE
- UCS:ssä myös 8-bittiset koodi ”rivit”
  - eri alueille tai tarkoitukseen (zone) omat 8-bittiset koodinsa

# UCS ja Unicode

- Merkit välillä 0x0000-00FF samassa järjestyksessä kuin Latin-9 merkistössä
  - 16-bittisen UCS:n ”rivi 00” = 8-bittinen Latin-9
- Myös muut aakkoset:
  - I-zone = Kanji (0x4E00-9FFF, 20992 merkkiä)
- Ei omia konekäskyjä, manipulointi aliohjelmilla

# Merkkijonot

- Yleensä peräkkäin talletettu joukko tavuja
- Lisäksi tarvitaan jollain tavalla koodata merkkijonon pituus
  - laitetaan loppuun erikoismerkki
    - C-kieli: `'\0'` = `0x00`
  - toteutetaan tietueena 

20	"Ei yleensä nyt enää!"
----	------------------------

pituus merkkijono
  - ei omia konekäskyjä, manipulointi aliohjelmilla
    - kokonaisluku- ja bittimanipulointikäskyt
    - joissakin koneissa `"strcpy"`- ja `strcmp`-käskyt

# Totuusarvot (4)

- Boolean TRUE ja FALSE
- Yleensä koodattu TRUE=1, FALSE=0
  - muttei aina!
  - totuusarvolauseke **A and B** = kokonaislukulauseke **A\*B**
- Usein Boolean arvo per sana
  - loput 31 bittiä nollia
  - ohjelmointikielten Boolean-muuttujat
- Joskus pakatussa muodossa 32/arvoa per sana
- Ei omia konekäskyjä, manipulointi aliohjelmilla
  - kokonaisluku ja bittimanipulointikäskyt
  - haluttu käsky ”**JTRUE ...**” voidaan toteuttaa käskynä ”**JPOS ...**”  
(jos TRUE =1)

# Kuvat

- Monta kuvastandardia
  - yleisyys, siirrettävyys, pakkaustiheys
  - näyttöä varten tarvittavan laskennan määrä
- Kuvatiedoston alussa otsake kertoo talletusformaatin
- Viiva- ja vektorikuvat
  - kuva koodattuna objekteina
    - ympyrä, monikulmio, käyrä, alueen väri
- Rasterikuvat
  - kuva koodattuna pisteinä
    - kunkin pisteen väri koodattu esim. 24 bitillä

# Kuvat

- Kuvat ovat yleensä pakattu mahdollisimman vähän tilaa vievää muotoon
  - optimoitu tilan, ei laskennan mukaan
  - purkaminen voi vaatia paljon laskentaa
- GIF, JPEG, TIFF, BMP, ....
- Ei omia konekäskyjä, manipulointi aliohjelmilla

# Videokuva

- Vie hyvin paljon muistitilaa
- Talletus kuva kerrallaan, esim. 25 kuvaa/sek
  - 1 sekunti hyvälaatuista videokuva pakkaamattomassa muodossa 20 MB
- Talletus ”incrementaalisesti”
  - kun seuraava kuva poikkeaa edellisestä vain vähän
  - talleta vain muutokset edelliseen



# Videostandardit

- MPEG (Moving Pictures Expert Group)
- AVI (Audio Visual Interleave)
- MOV, INDEO, FLI, GL, DVD, ...
- ei omia konekäskyjä, manipulointi aliohjelmilla
- tai erikoisprosessoreilla (GPU), joiden käskykanta suunniteltu (jonkin standardin mukaisten kuvien) kuvankäsittelyyn
  - grafiikkakorteilla

# Grafiikkakortit

- Esim. 4-64 MB (dual-port) muistia
  - 2 lukua/kirjoitusta samanaikaisesti
  - tai 'tavallista' , mutta hyvin nopeaa RAMia
- Nopea väylä (ennen PCI, nyt AGP) suorittimelle
- Näytönohjaus monitoristandardien (VGA, XGA, RGB, ..) mukaisesti
- Oma suoritin (GPU)
  - lukee videodataa ja generoi näytettävän kuvan näyttöpuskuriin, josta monitori sen näyttää
- Voi olla integroitu emolevyn kanssa

# Äänet

- Täydellinen äänidata
  - 44100 näytettä/sek, 16 b/näyte, 88KB/sek
- Syntetisoitu ääni
  - MIDI-käskeyjä
    - Music Instrument Digital Interface
    - ”Soita nuotti N voimakkuudella V”
  - ei omia konekäskeyjä, manipulointi aliohjelmilla
  - tai erikoisprosessoreilla, joiden käskeykanta suunniteltu äänen käsittelyyn
    - äänikortit

# Äänikortit

- Esim. 4-64 MB VRAM- tai RAM-muistia
- nopea väylä (esim. PCI) suorittimelle
- oma suoritin, joka lukee äänidataa ja generoi äänet kaiuttimille tai vahvistimeen
  - kaiuttimet tai vahvistin kiinni äänikortilla
- Voi olla integroitu emolevyn tai grafiikkakortin kanssa

# Maku, haju, tunto ja muu data <sup>(3)</sup>

- Tähtien kirkkaus, hajut, ks. HS artikkeli 5.5.2000  
veneeseen tyyppi, tunteen palo, ....
- Sovelluskohtaisesti, ei vielä yleisiä standardeja
  - kokonaisluvut (diskreetti data)
  - liukuluvut (jatkuva data)
- Ei omia konekäskyjä, manipulointi omilla aliohjelmilla

# Konekäskyjen esitysmuoto muistissa (4)

- Konekohtainen, jokaisella omansa
- Käskyt ovat 1 tai useamman tavun mittaisia
  - SPARC, kaikki käskyt: 1 sana eli 4 tavua
  - PowerPC, kaikki käskyt: 1 sana eli 4 tavua
  - Pentium II: 1-16 tavua, paljon variaatioita
- Käskyillä on yksi tai useampi muoto, kussakin tietty määrä erilaisia kenttiä
  - opcode, Ri, Rj, Rk, osoitusmoodi
  - pitkä tai lyhyt vakio

TTK-91, kaikki käskyt: 1 sana, 1 muoto

# TTK-91 konekäskyn rakenne

- Käskyn esitys bittitasolla on aina:



Rj = käskyn ensimmäinen operandi

Ri = indeksirekisteri

M = muistinoutojen määrä toiseen operandiin  
(ennen mahdollista muistiin talletusta)

00 eli 0 kpl, rekisteri tai välitön osoitus

01 eli 1 kpl, suora osoitus

10 eli 2 kpl, epäsuora osoitus

( 11 eli 3 kpl, epäkelpo arvo → poikkeustilanne )

muistiosoite tai  
(pienehkö) vakio

(addressing  
mode)

# Konekäskyn operandit ja tulos

- Tulos: rekisteri  $R_j$ 
  - paitsi WRITE- tai PUSH-käskyissä muistipaikan sisältö
- Ensimmäinen operandi: rekisteri  $R_i$
- Toinen operandi
  - laske ensin arvo  $R_i + ADDR$  ja käytä sitä sellaisenaan ta  
käytä sitä muistisoitteena
  - arvo:  $R_i + ADDR$
  - muistipaikan  $M[R_i + ADDR]$  sisältö
  - muistipaikan  $M[M[R_i + ADDR]]$  sisältö

jos  $R_i = R_0$ ,  
niin pelkkä ADDR

Kone-  
kielen  
tiedon  
osoitus-  
moodit



# Taulukkojen esitysmuoto

- Peräkkäisrakenteena, kuten esimerkit aikaisemmin
- riveittäin tai sarakkeittain
- ei omia konekäskyjä, manipulointi aliohjelmilla tai loopeilla
  - paitsi ns. vektorikoneet, joilla on omia konekäskyjä vektorioperaatioita varten
- Indeksoitu tiedonosoitusmoodi tukee yksiulotteisten taulukoiden käyttöä

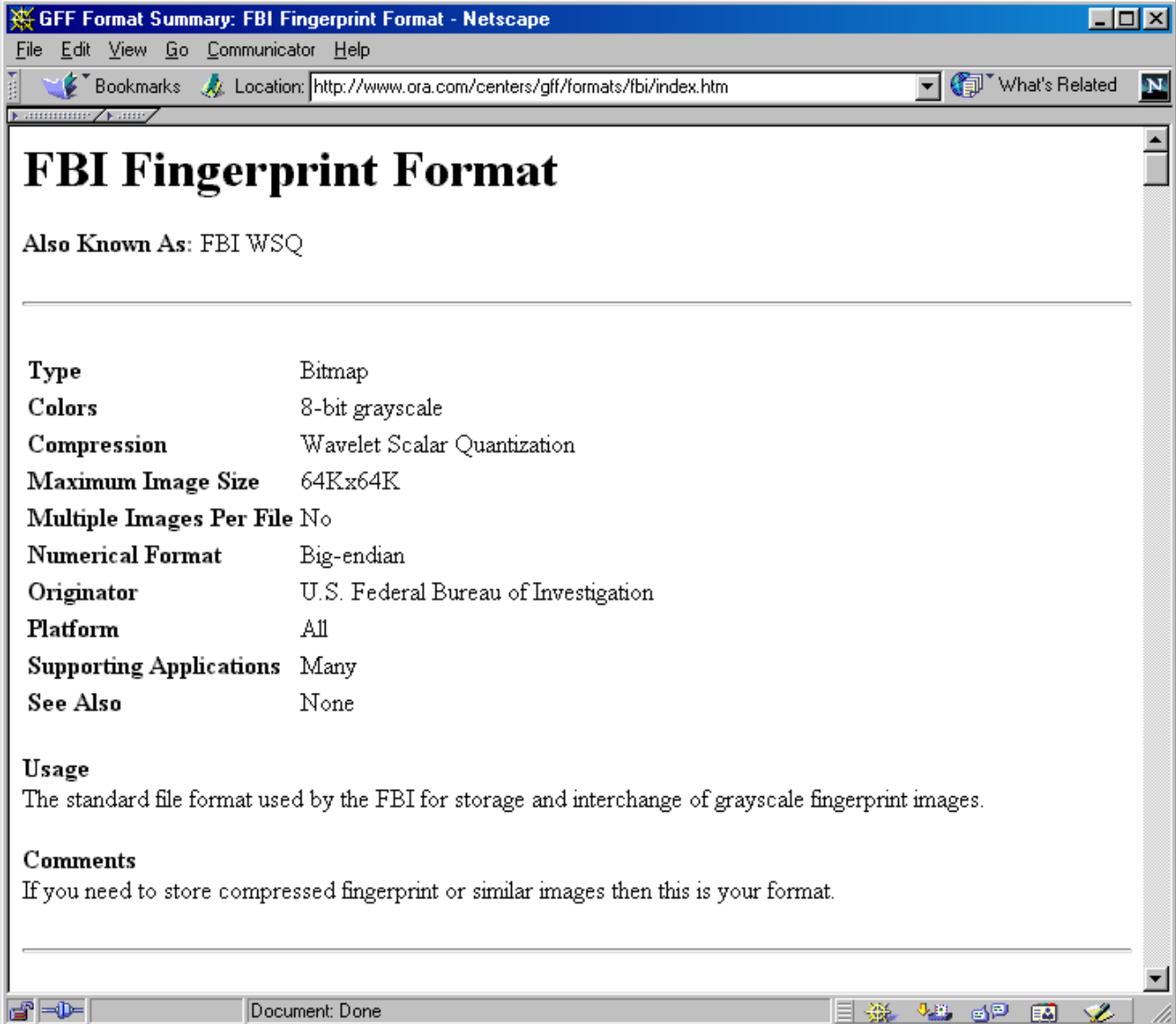
# Tietueiden esitysmuoto

- Tietueet peräkkäisrakenteena
- Osoite on jonkin osoitemuuttujan arvo
- Ei omia konekäskyjä, manipulointi aliohjelmilla tai kääntäjän generoimien vakiolisäysten avulla
- Indeksoitu tiedonosoitusmoodi tukee tietueiden käyttöä

# Olioiden esitysmuoto

- Kuten tietueet, yleensä varattu keosta (heap)
- Useat olion kentistä sisältävät vuorostaan osoitteen keosta suoritusaikana varattuun toiseen olioon
- Metodit ovat aliohjelmien osoitteita
- Ei omia konekäskyjä, manipulointi aliohjelmilla

# -- Jakson 6 loppu --



The screenshot shows a Netscape browser window with the title "GFF Format Summary: FBI Fingerprint Format - Netscape". The address bar contains the URL "http://www.ora.com/centers/gff/formats/fbi/index.htm". The main content area displays the following information:

## FBI Fingerprint Format

Also Known As: FBI WSQ

---

<b>Type</b>	Bitmap
<b>Colors</b>	8-bit grayscale
<b>Compression</b>	Wavelet Scalar Quantization
<b>Maximum Image Size</b>	64Kx64K
<b>Multiple Images Per File</b>	No
<b>Numerical Format</b>	Big-endian
<b>Originator</b>	U.S. Federal Bureau of Investigation
<b>Platform</b>	All
<b>Supporting Applications</b>	Many
<b>See Also</b>	None

**Usage**  
The standard file format used by the FBI for storage and interchange of grayscale fingerprint images.

**Comments**  
If you need to store compressed fingerprint or similar images then this is your format.

The browser's status bar at the bottom shows "Document: Done" and various system icons.