

Jakso 7

Tiedon muuttumattomuuden tarkistus Järjestelmän sisäinen muisti



Pariteetti

Hamming-koodi

Välimuisti

Tavallinen muisti

Historiaa

Tiedon tarkistus ⁽⁴⁾

- Tiedon oikeellisuutta ei voi tarkistaa yleisessä tapauksessa
- Laitteistovirheitä voidaan havaita ja joskus automaattisesti korjata
 - bitti voi muuttua muistissa tai tiedon siirrossa
 - muistipiirissä voi olla vika (staattinen vika)
 - sopiva alkeishiukkanen voi muuttaa bitin tiedonsiirron aikana (transientti virhe)
- Tietokannan eheys on eri asia!

Lisää
tietoa?



Tieto-
kanta
kurssit

Tiedon muuttumattomuus (2)

- Perusidea: otetaan mukaan ylimääräisiä bittejä, joiden avulla virheitä voidaan havaita ja ehkä myös korjata
- Järjestelmä suorittaa tarkistukset automaattisesti joko laitteistotasolla tai ohjelmiston avulla

Esimerkki ohjelmistotason tarkistusmerkistä ⁽²⁾

- Henkilötunnus: 120464-121C

$$120464121 \% 31 = 12$$

0123456789 ABCDEFHJKLMNPRSTUVWXY
 ↑ ↑↑

- Tarkistusmerkin avulla voidaan tarkistaa, että mikään yksi merkki ei ole väärin
 - havaitsee yhden merkin virheen
 - virhettä ei voi automaattisesti korjata!! Miksi?

Bittitason tarkistukset ⁽⁷⁾

- Muistipiirit, levyt, väylät, tiedonsiirrot 120464-121C
- Monenko bitin muuttuminen havaitaan? Hetu: 1
- Monenko bitin muuttuminen voidaan automaattisesti korjata? Hetu: 0
- Korjaamiseen tarvitaan enemmän ylimääräisiä bittejä
 - lisämuistitilan tai levytilan tarve? Hetu: +10%
 - lisäpiuhojen tarve väylällä?
- Tarkistukset/korjaukset laitteisto- vai SW-tasolla? Hetu: ohjelmistotasolla

Pariteettibitti ⁽⁹⁾

- Yksi ylimääräinen bitti per tietoalkio
 - sana, tavu, tietoliikenne paketti
- Parillinen (pariton) pariteetti: 1-bittien lukumäärä on aina parillinen (pariton)
- Havaitsee: 1 bitti
- Korjaa: 0 bittiä
- Esimerkki (parillinen pariteetti)

0010 001 0

1000 1101 1111 001 1

Hamming etäisyys ⁽³⁾

- Montako bittiä jossain koodijärjestelmässä (esim ISO Latin) esitetyllä koodilla (esim. 'A' = 0x41 = 0100 0001) täytyy muuttua, että se muuttuu johonkin toiseen (mihin tahansa) lailliseen koodiin.

'A' = 0x41 = 0100 0001

2 bittiä

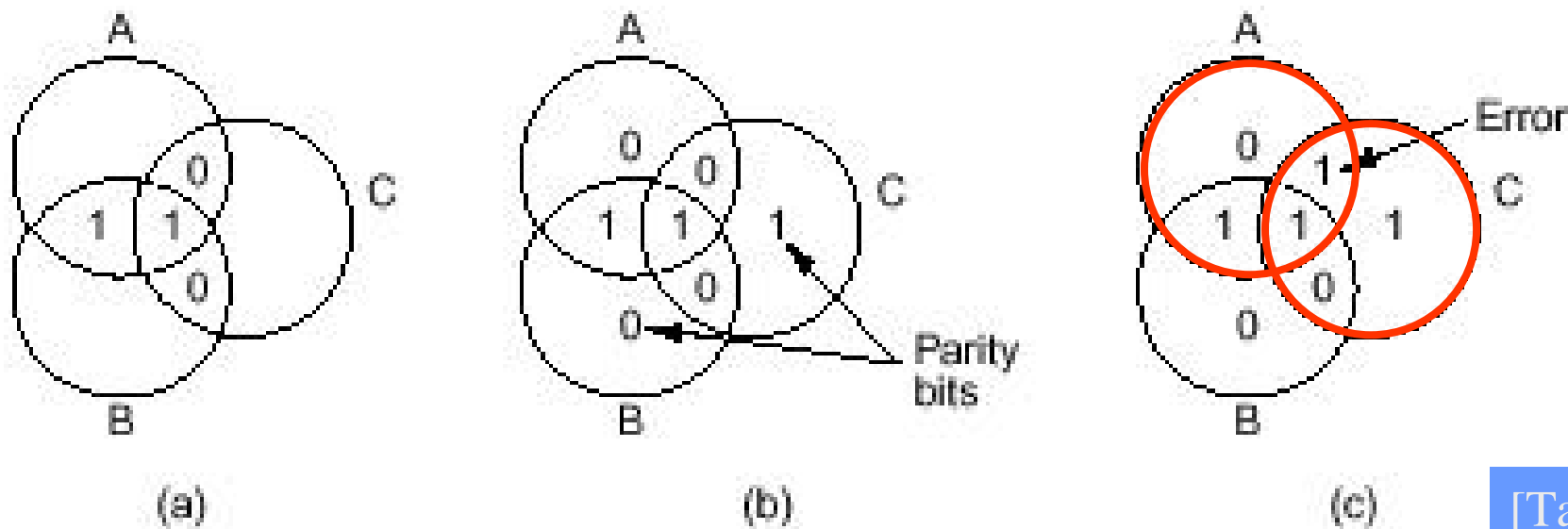
'B' = 0x42 = 0100 0010

1 bittiä

'C' = 0x43 = 0100 0011

- ISO Latin-1:n Hamming etäisyys: 1
- Pariteettibitin kanssa Hamming etäisyys: 2

Hamming koodi (2)



[Tane99]

Figure 2-14. (a) Encoding of 1100. (b) Even parity added. (c) Error in AC.

Kukin databitti (4 kpl)
kuuluu erilaisiin
pariteettijoukkoihin (3 kpl)

Joukot A ja C
havaitsevat virheen
ja siten paikallistavat
virheellisen bitin

Hamming koodi ⁽⁹⁾

- Käytetään useampia pariteettibittejä
- Havaitsee: 2:n bitin muuttuminen
- Korjaa: 1 bitin muuttuminen

Data:

100 1100

4 bittiä dataa,
3 pariteettibittiä

Bitti nro:

765 4321

Kaikki bitit nro 2^i ovat pariteettibittejä,
muut ovat databittejä (numerot alkavat 1:stä)

Kutakin data-bittiä n tarkistavat ne pariteettibitit
joiden summana n voidaan esittää. Parillinen pariteetti.

$6 = 4 + 2 \Rightarrow$ databittiä 6 tarkistavat par. bitit 4 ja 2

Virheen korjaava Hamming koodi ⁽⁸⁾

(ECC)

Data:

100 1100

110 1100

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

Tapahtuu virhe: bitti 6 muuttuu (flips)

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

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

$2+4 = 6 \Rightarrow$ korjaa bitti nro 6

Virheen korjaava Hamming koodi (ilman animaatioita)

Data:

100 1100

110 1100

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

Tapahtuu virhe: bitti 6 muuttuu (flips)

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

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

$2+4 = 6 \Rightarrow$ korjaa bitti nro 6

CRC - Cyclic Redundancy Code ⁽⁷⁾

- Tiedonsiirrossa käytetty tarkistusmenetelmä
- Tarkistussumma (16 bittiä) isolle tietojoukolle
 - laske $CRC = f(\text{viesti} * 2^{16}) \% (X^{16} + X^{15} + X^2 + 1)$
(huom! polynomijako)
 - lähetä viesti ja CRC
 - vastaanota viesti ja CRC
 - laske CRC ja tarkista oliko oikein (pitäisi olla 0!)
 - jos pielessä, niin pyydä uudelleenlähetyistä

CRC-CCITT CRCs detect:

All single- and double-bit errors

All errors of an odd number of bits

All error bursts of 16 bits or less

In summary, 99.998% of all errors

Virheiden tarkistusmenetelmien käyttöalueet

- Mitä lähempänä suoritinta, sitä tärkeämpää tiedon oikeellisuus on
- Sisäinen väylä, muistiväylä
 - virheet korjaava Hamming koodi
- Paikallisverkko
 - uudelleenlähetyksen vaativa CRC
 - kun tulee virheitä, niin niitä tulee yleensä paljon
 - Hamming koodi ei riitä kuitenkaan
 - pariteettibitti päästää läpi 2 virheen paketit

Laitteiden monistaminen ⁽⁶⁾

- Monta muistipiiriä, samat tiedot
- Monta suoritinta, samat käskyt
- Monta laitteistoa, samat ohjelmat
- Eri tyyppiset laitteistot, samankaltaiset ohjelmat (samat speksit, samat syötteet)
- Äänestysmenettely: enemmistö voittaa
 - monimutkainen, hidas?
 - virheelliseksi havaittu laitteisto suljetaan pois häiriköimästä automaattisesti?

Lentokoneet, avaruussukkula, ydinvoimala, ...

Välimuisti (cache) ⁽³⁾

- Ongelma: keskusmuisti on aika kaukana suorittimesta

rekisterin viittausaika: X
muistin viittausaika: 10X

- Ratkaisu: välimuisti lähelle suoritinta

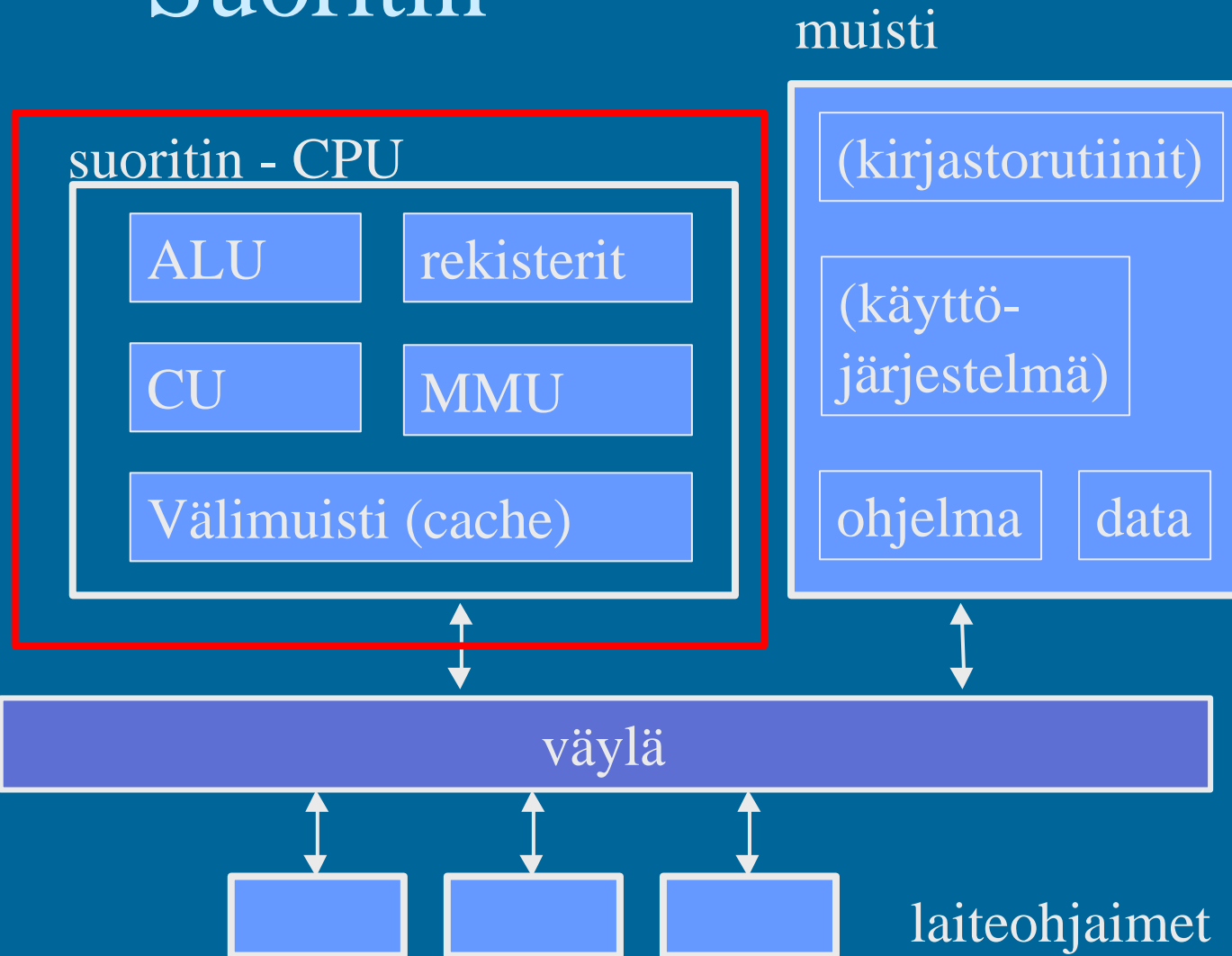
- pidetään siellä (kopioita) viime aikoina viitatuista keskusmuistin alueista

välimuistin viittausaika: 2X

- Jokainen muistiviite on nyt seuraavanlainen

- jos data ei ole välimuistissa, niin hae se sinne
 - suoritin odottaa tällä aikaa
- tee viittaus dataan (käskyyn) välimuistissa
- (talleta muutettu tieto keskusmuistiin)

Suoritin



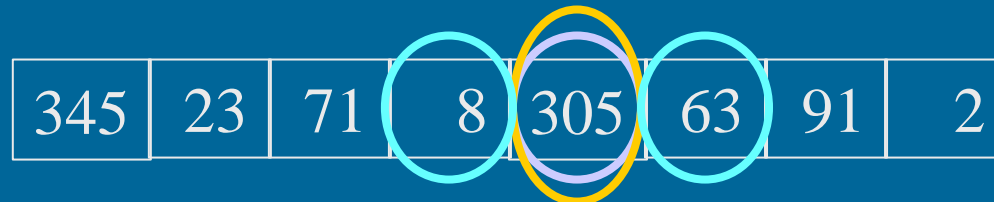
Välimuisti

Fig. 4.13 [Stal99]

- Tuntumaton suorittimelle
 - jos viitattu tieto ei saatavilla, niin suoritin vain odottaa muutaman kellopulssin ajan...
- Toteutettu usein nopeammalla teknologialla kuin keskusmuisti (tavallinen muisti)
- Toteutettu usein samalla mikropiirillä kuin suoritin
- Silti iso aikaero: välimuisti 2X, muisti 10X
 - ratkaisu: monitasoiset välimuistit: L1, L2, L3
 - L2 eri piirillä, mutta isompi kuin L1
- TTK-91 koneessa ei ole välimuistia

Miksi välimuisti toimii? ⁽⁵⁾

- Paikallisuusperiaate: tietyssä aikavälillä muistiviitteet kohdistuvat vain koko muistin pieneen osajoukkoon (locality principle)
- Ajallinen paikallisuus: on todennäköistä, että vähän aikaa sitten viitattuun tietoon viitataan uudelleen kohta (temporal locality)
- Alueellinen paikallisuus: on todennäköistä, että vähän aikaa sitten viitatus tiedon lähelle viitataan lähitulevaisuudessa (spatial locality)



Miten välimuisti toimii? ⁽³⁾

Fig. 4.13 [Stal99]

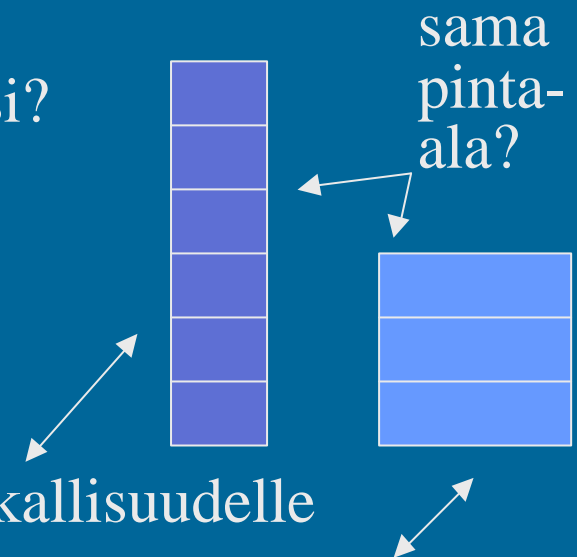
- Välimuistissa on useita keskusmuistin lohkoja
 - esim. 4 sanaa / lohko
- Kun tulee ”välimuistihuti” (viitattu data ei välimuistissa), niin koko tuon datan sisältämä lohko (”rivi”) kopioidaan muistista välimuistiin
- Uusille lohkoille tehdään tilaa poistamalla vanhoja lohkoja, joihin ei toivottavasti heti viitata
 - ne lohkot, joihin on kirjoitettu, täytyy kopioida muistiin

(cache block
cache line)

Välimuistin ominaisuuksia (5)

- Koko
 - isommasta löytyy enemmän dataa, mutta se voi olla hitaampi
- Kuvausfunktio
 - miten löytää data välimuistista?
- Poistoalgoritmi
 - mikä lohko poistetaan seuraavaksi?
- Kirjoituspolitiikka
 - miten käsitellä WRITE käskyjä?
- Lohkon (rivin) koko
 - monta pientä: hyvä ajalliselle paikallisuudelle
 - muutama iso: hyvä alueelliselle paikallisuudelle

Fig. 4.13 [Stal99]



Muistin toteutus ⁽⁶⁾

- Eri teknologioita eri tasoisiin muisteihin
- RAM - Random-Access Semiconductor Memory
 - anna osoite ja lue/kirjoita signaali
 - mistä vaan voi lukea/kirjoittaa samassa ajassa
 - virta pois \Rightarrow tiedot häviävät (volatile memory)

Huom: kaikki nykyiset muistit ovat ”random access”

RAM:n kaksi eri teknologiaa (2)

- DRAM: dynaaminen RAM, halvempi, hitaampi, tietoja pitää virkistää vähän väliä (esim. joka 2 ms)
 - tavallinen keskusmuisti (1975-..) useimmissa koneissa
 - toteutettu kondensaattoreilla, jotka ”vuotavat” ...
- SRAM: static RAM, kalliimpi (~20x), nopeampi (~10x), ei vaadi tietojen virkistämistä
 - välimuisti useimmissa koneissa
 - muisti superkoneissa (esim. Cray C-90)
 - toteutettu samanlaisilla logiikkaportteilla (gate) kuin prosessorikin
 - CMOS valmistusteknologia
(Complementary Metal Oxide Semiconductor)

Muistin toteutus (7)

- ROM - Read-Only Memory
 - tieto säilyy virran katkettua (non-volatile)
 - voi käytössä vain lukea, ei voi kirjoittaa
 - esim. järjestelmän alustustiedot (BIOS)
 - kirjoitus lastun valmistusaikana, Mask-ROM
 - huono puoli: kerran väärin, aina väärin
 - päivitys: laita valmistajalta saatu uusi lastu paikalleen
 - tietoa voi lukea mistä vain samassa ajassa (random access)
 - yleensä hitaampi kuin RAM (~10x)

Kirjoitettavia ROM-muisteja (6)

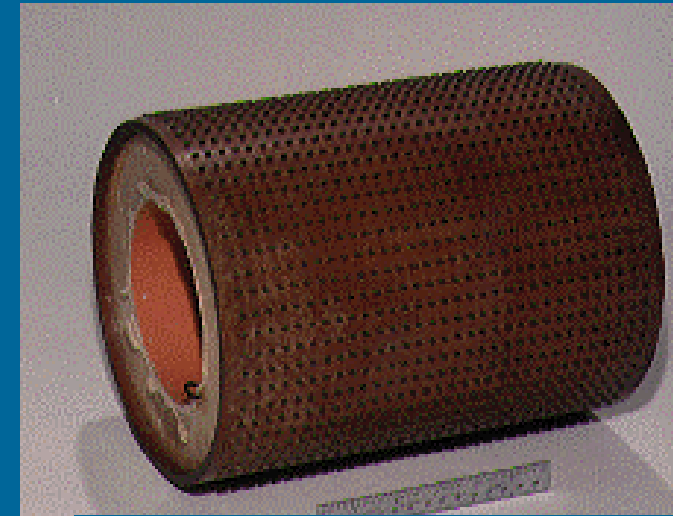
- PROM - Programmable ROM
 - kerran kirjoitettava
 - tiedon päivitys: ”polta” tiedot tyhjään PROM:iin
- EPROM - Erasable PROM
 - tietoja ei voi päivittää sana kerrallaan
 - vanhat tiedot voidaan (kaikki!) poistaa 20 min. UV-säteilyllä, jonka jälkeen päivitettyt tiedot voidaan ladata
- EEPROM - Electronically Erasable PROM
 - tietojen pyyhkiminen tavukohtaisesti elektronisesti
- FLASH EEPROM memory
 - tietojen pyyhkiminen nopeasti kerralla elektronisesti
 - normaalijännitteellä
 - nopeampi kuin EEPROM

read-mostly memory

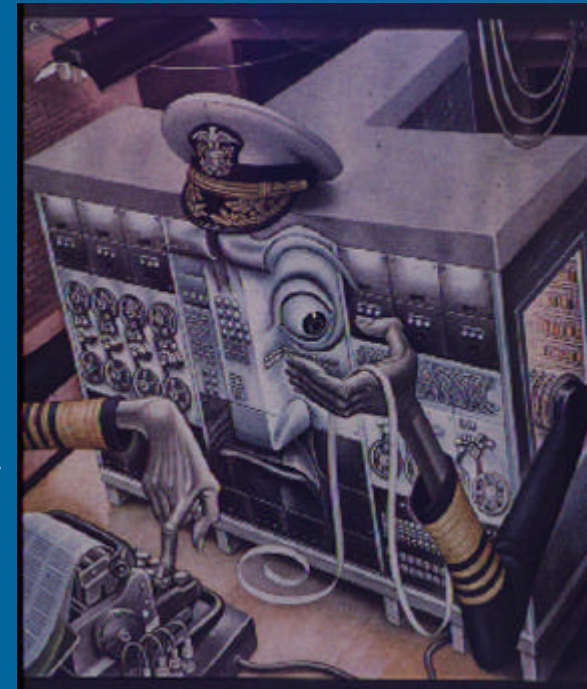
BIOS, CIH-virus

Muistien historiaa

- Rumpumuisti
 - 1939, ABC, Atanasoff-Berry Computer, Iowa State College.
 - lähinnä laskin ...
 - 1951, Aiken Mark III
 - erilliset rumpumuistit koodille ja datalle



30 numeroa á 50 bittiä



Artzybasheff
Time cover
1951

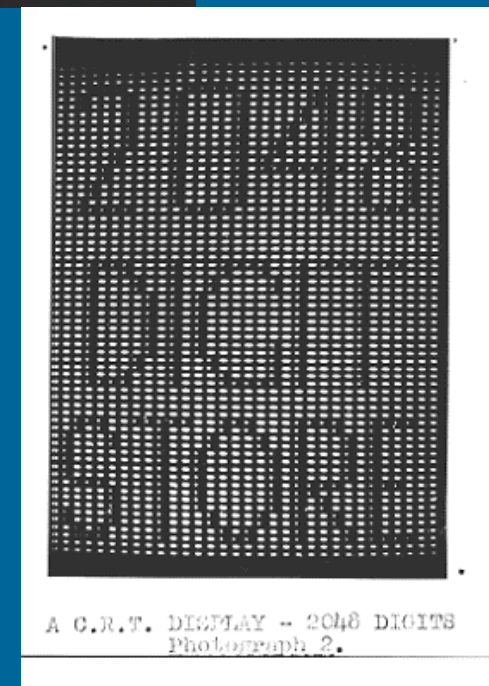
Muistien historiaa

- Williams Tube
 - 1946, Williams & Kilburn
 - katodisädeputki
 - ensimmäinen suuri ”RAM” muisti



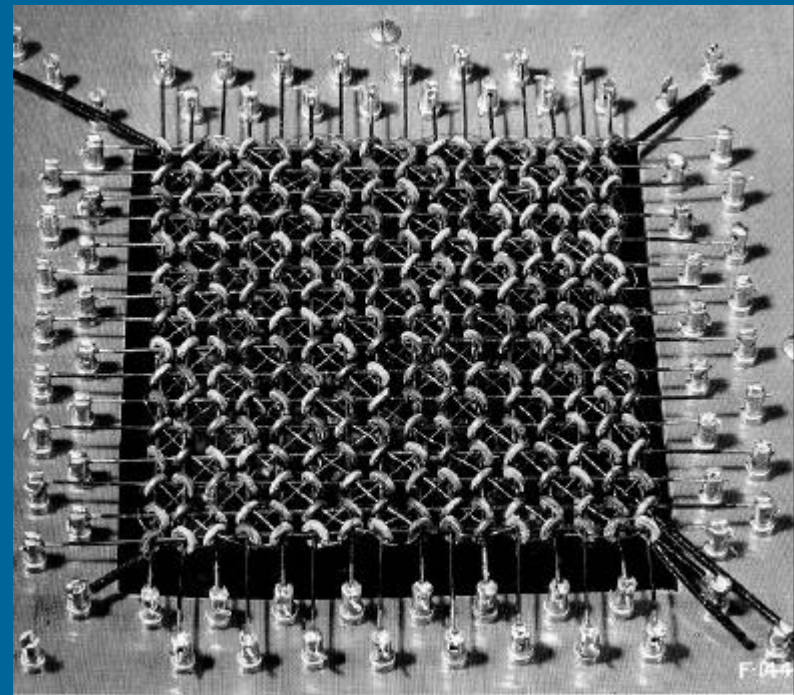
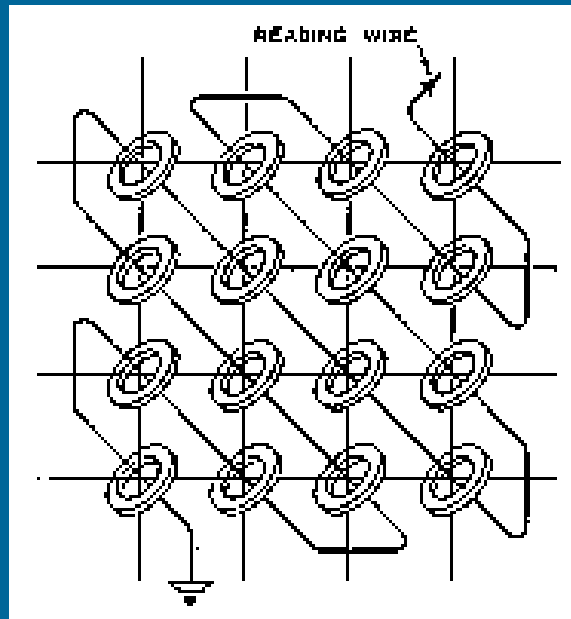
Tom Kilburn holding a Cathode Ray Tube

Storing
2048 bits
on
a CRT in
1947



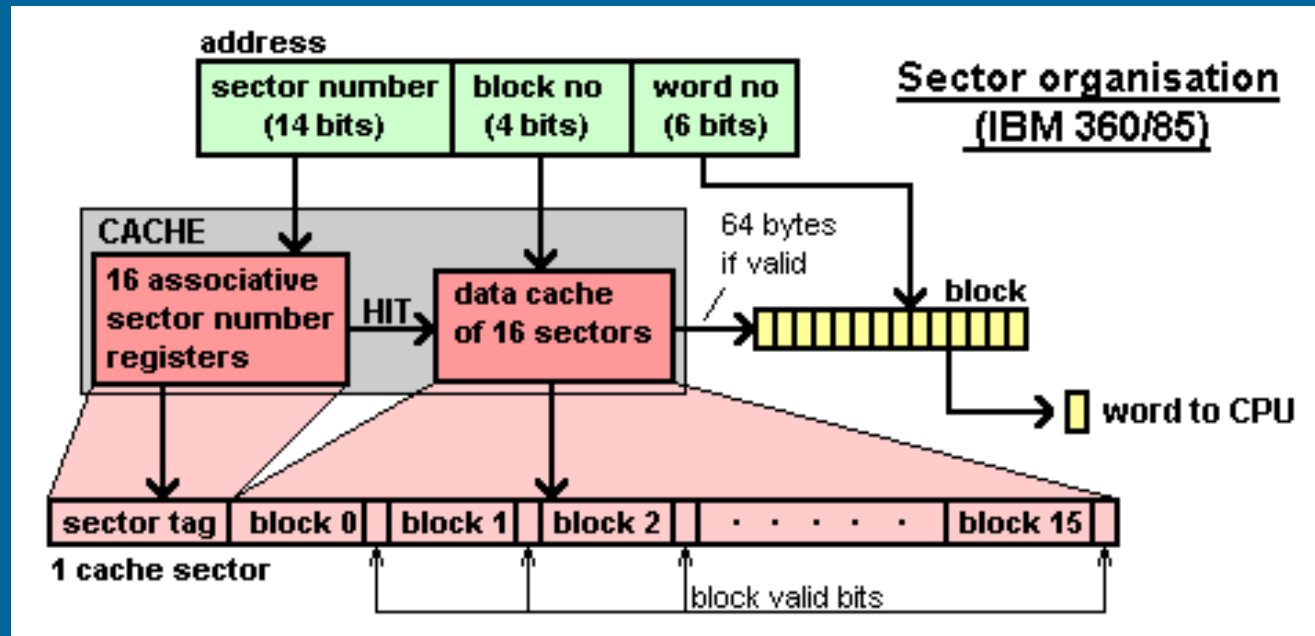
Muistien historiaa

- Ferriittirengas (core) teknologia
 - 1952, Jay Forrester & Bob Everett, MIT (Whirlwind)
 - tieto säilyy ilman virtaa
 - ei häiriinny säteilystä (avaruus, sotilasteknologia)
 - 1955, valtaa markkinat Williams Tube' lta



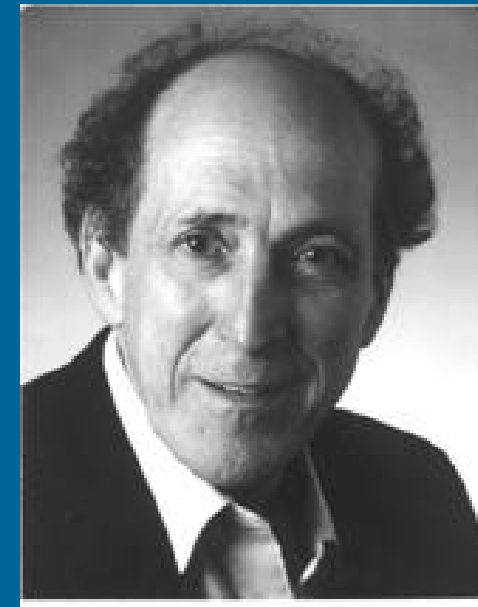
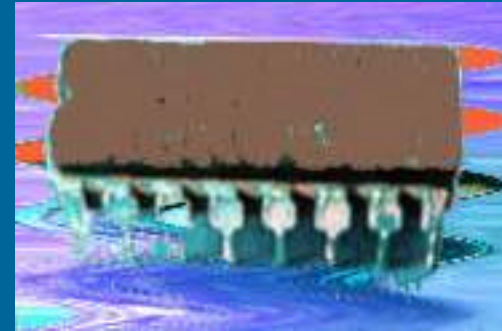
Muistien historiaa

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



Muistien historiaa

- DRAM (1966, Robert Dennard, IBM)
 - Intel 1103 (1970)
 - John Reed
 - 1 Kbit
 - valtaa markkinat ferriittirengasmuisteilta 1972
- SRAM (1970, Fairchild Corp)



Robert Dennard

Muistien historiaa

- PROM
 - ???
- EPROM
 - 1971, Dov Frohman, Intel 1701
- EEPROM,
 - 1980, Intel 2816
- Flash EEPROM
 - 1984, Lai et al, Intel

Muistien historiaa

- OROM - optical ROM
 - 1990, James Russell
(Russell keksi myös CD-ROM:n)
 - 1998, Wond-OROM-a
 - 128 MB/kortti plus lukulaite
 - ei liikkuvia osia
 - sama nopeus kuin CD-ROM:lla
(siis aika hidas!)
 - pieni virrankulutus
 - sopii kannettaviin laitteisiin



--
Jakson
7
loppu
--

Whirlwind Project,
MIT, 1946- ...

