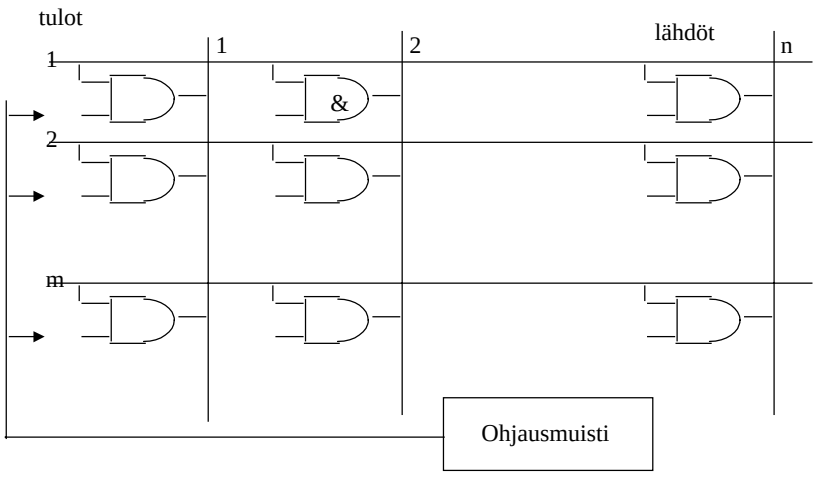
Kytkentäkentän teknologia

Kertaus
kentän rakenteeseen vaikuttavat
teknologiset tekijät
Huom. tätä ei löydy kirjasta!

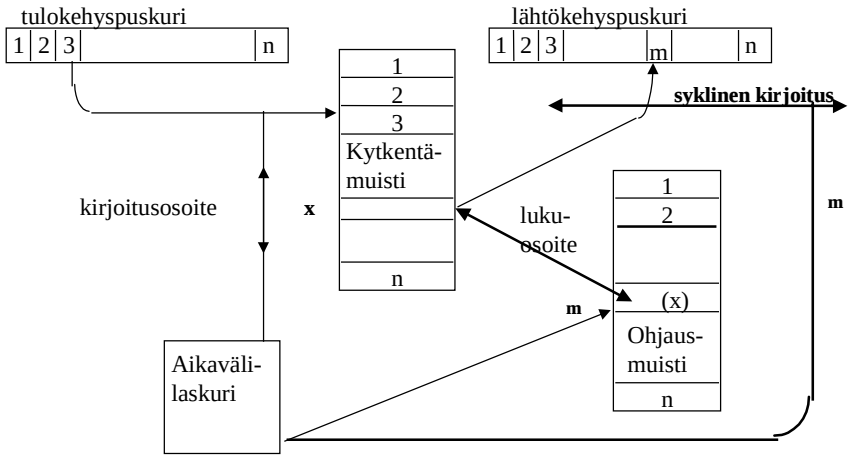
Kertaus 1 - Tilaporras - esimerkki

- **Tilakytkin on yksinkertainen ristikytkentämatriisi, jonka kytkentäpisteitä ohjaamalla voidaan informaatiovirtaa suunnata.**



Kertaus 2 - Aikaporras - sarjakirjoitus-osoiteluku

KM kirjoitetaan aikavälilaskurin ohjaamana syklisesti tulopuolen tahdissa. KM luetaan ohjausmuistin sisällön osoittamana, ohjausmuistin osoite ja lähtö vastaavat toisiaan. Ohjausmuistia luetaan syklisesti



Aikakytkinten ominaisuuksia

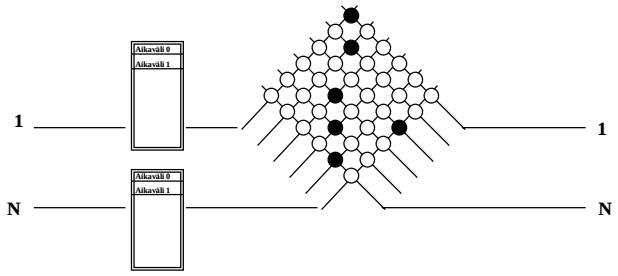
- ✔ **Tulokehyspuskuriin bitit tulevat johtojen bittinoipeudella, ne lähtevät lähtöpuskurista johdon bittinopeudella - siispä edellisestä pitää lukea aikavälit samaan tahtiin ja jälkimmäiseen kirjoittaa samaan tahtiin ja järjestyksessä.**
- ✔ **kytkentämuistiin kohdistuu kehyksen aikavälimäärän verran kirjoituksia ja sama määrä lukuoperaatioita kehyksen aikana -> kytkentämuistin nopeus on kriittinen parametri: *saatavilla oleva nopeus halutaan hyödyntää täysimääräisesti, mutta sen yli ei voida mennä ilman rinnakkaisuutta.***
- ✔ **Sarja-rinnan ja R/S -muunnos hyvä tehdä kehyspuskureissa**
- ✔ **ohjausmuistin nopeusvaatimus on hieman yli puolet kytkentämuistista, koska joskus kytkentöjä pitää myös muuttaa.**

Kertaus 3 - kaksiportaiset kentät

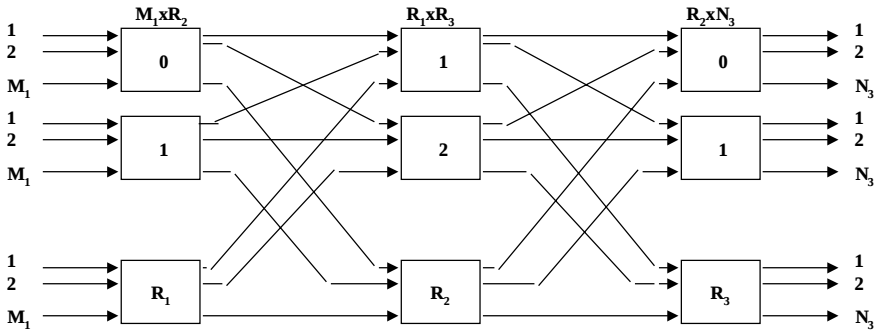
- ✔ **Kytkentäkenttä muodostetaan erilaisilla kombinaatioilla tila- ja aikakytkimiä.**
- ✔ **Kaksiportaiset kytkentäkentät muodostetaan kahdella peräkkäisellä kytkimellä:**
  - § **Aika-aika (AA)**
  - § **Aika-tila (AT)**
  - § **Tila-aika (TA)**
  - § **Tila-tila (TT)**

Kertaus 4 - tärkein 2-portainen kenttä on AT-kenttä

- ✔ **AT-kenttä on rakenteeltaan vähäestoinen, sillä aikakytkin mahdollistaa aikavälien järjestelyn niin, että kytkentä tilakentässä on estotonta.**

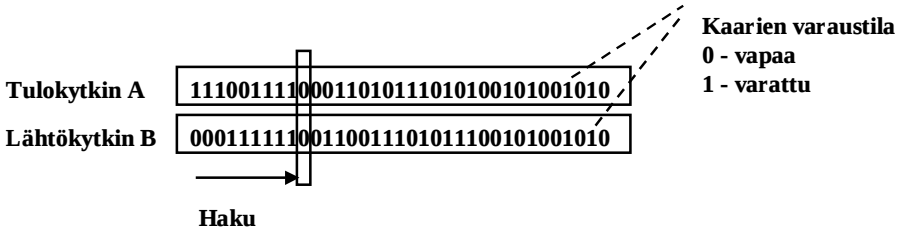


Closin -verkko



- **Porras 1: $N_1 = R_2$**
- **Porras 2: $M_2 = R_1$ ja $N_2 = R_3$**
- **Porras 3: $M_3 = R_2$**

Polun haku Closin kentässä perustuu portaiden välisten kaarien varaustilavektoreihin



Teknologia 1 - Moniportaisten kenttien ongelmat

- ✓ **Tarvitaan polun hakua**
- ✓ **Jos tarvitaan nopeaa kytkentää, vaaditaan erittäin nopea ohjaus**
- ✓ **Jos ohjaus ei ole riittävän nopea, kentän käyttöaste laskee**

- ✓ **Jakelu ei ole itsestään selvyys - itse asiassa se on hankalaa**
- ✓ **Moniaikavälikytkennät voivat synnyttää ongelmia, jos kulkuaikaviive kentässä ei ole vakio. Lisäksi esto voi kasvaa.**

Vaihtoehtona on lähteä teknologisista rajoitteista

- ✔ **Ei pyritä optimoimaan yhtä suuretta eli krosspointtien lukumäärää, vaan tarkastellaan useita rajoitteita yhtä aikaa.**
- ✔ **Miten nopeita komponentteja on tarjolla.**
- ✔ **Mikä on komponenttien ajokyky.**
- ✔ **Kuinka tiiviisti komponentteja voidaan pakata ilman että syntyy lämpöongelmia (tehonkulutus).**
- ✔ **Kuinka pitkiä väyliä kenttään joudutaan rakentamaan. Pitkät väylät laskevat kentän sisäistä nopeutta ja vaikeuttavat mm vikojen paikantamista.**
- ✔ **Halu/haluttomuus käyttää erikoiskomponentteja.**

Nopeutuvat komponentit vievät kohti matriisikenttiä

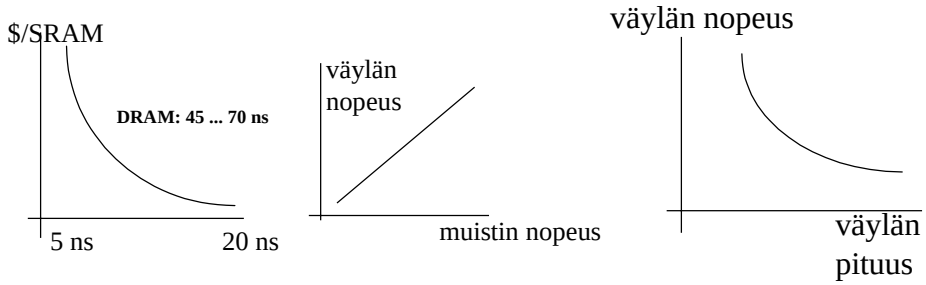
- ✓ **SRAM on nopeampaa kuin DRAM**
- ✓ **Nykyisellä SRAM tekniikalla voidaan helposti toteuttaa esim. 8k * 2M PCM matriisikenttä - tämän isompaa ei juuri kukaan haluakaan…**
  ==> Kapeakaistaverkkoissa jo 10 vuotta matriisikentät ovat vallitseva suunta.
- ✓ **Nopeutuvat loppukäyttäjän yhteydet pitävät moniporraskenttiä aina pinnalla.**

Matriisikentän ominaisuudet

- + **Täysin estoton**
- + **Ei polun hakua - aina suoraan kytkettävissä, jos lähtö vapaa**
- + **Monipistekytkennät (1 tulo --> monta lähtöä) helppoja**
- + **Vakioviive**
- + **Moniaikaväliset kytkennät helppoja**

- **- Kytkentä- ja ohjausmuistin neliöllinen kasvu**
- **- Laajakaistasiirto --> Muistin nopeus voi olla riittämätön --> Moniportaiset kentät**

Nopea muisti pitää voida hyödyntää



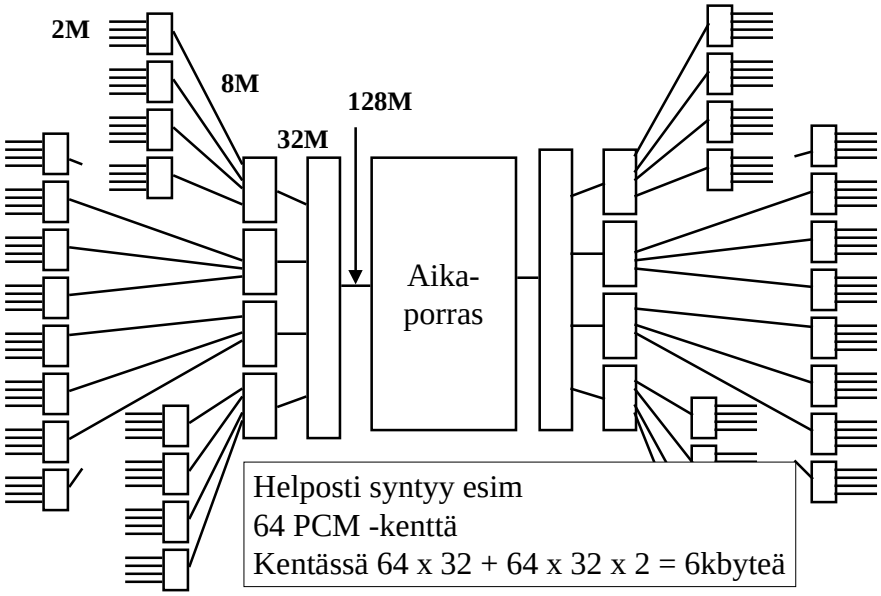
- ✓ **Hinta rajoittaa kaikkein nopeipien muistien käyttöä. Suunnitteluhetkellä valitaan yleensä komponentteja, jotka tekevät tuotteesta kohtuuhintaisen samalla kun suorituskyky on riittävä.**
- ✓ **Jotta nopeasta muistista saatais kaikki irti, väylien on oltava nopeita. Muistien ja väylien nopeudet kasvavat samassa tahdissa.**
- ✓ **Kun väylien nopeus kasvaa, niiden taloudellinen pituus laskee käänteisesti tai nopeammin.**

PCM-nopeudet vs. muistin nopeus

Nopeus	Aikavälin kesto, ns	Bitin kesto, ns
2M	3906	488
34M	230	29
64 * 2M	61	8
128 *2M	31	4
256 * 2M	15	2

-->64… 256 PCM:ää voidaan kirjoittaa/lukea samaan muistipiiriin jatkuvasti reaaliajassa.
--> ennen kytkentää aikavälit kannattaa kääntää rinnakkaismuotoon

Helposti voidaan toteuttaa aikaporras:



Helposti syntyy esim
64 PCM -kenttä
Kentässä 64 x 32 + 64 x 32 x 2 = 6kbyteä

Edellisen esim. kytkentämuistin nopeus

Oletus: KM on yksi muistipiiri:

Koko on 64 x 32 oktettia = 2kbytea

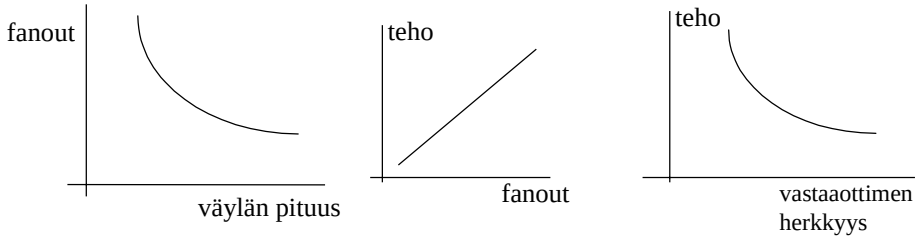
Kehyksen aika on 125 μs.

Kirjoituksia 2k/125 μs

Lukuja 2k/125 μs

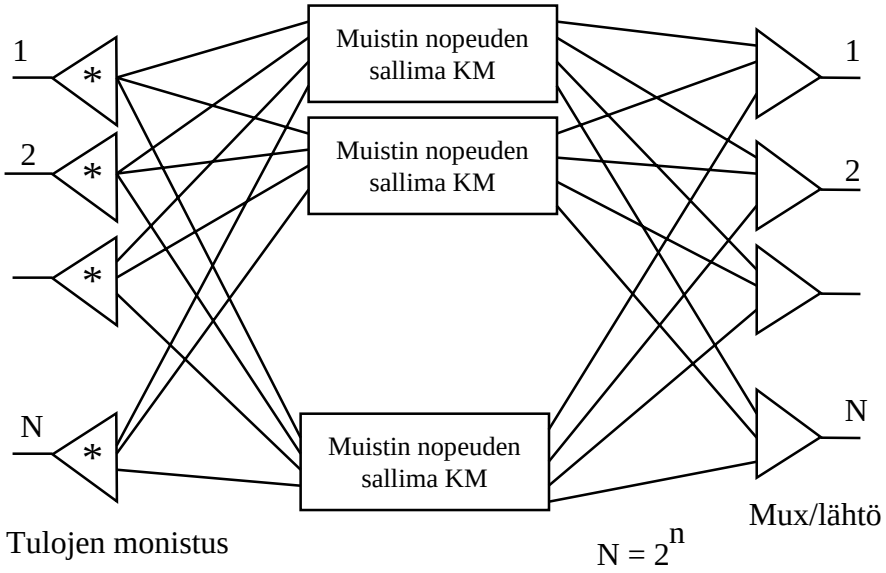
Muistioperaatioita 4k/125 μs = 1/30 ns

Kentän kuluttamaa tehoa täytyy rajoittaa, jotta lämpöongelmia ei synny



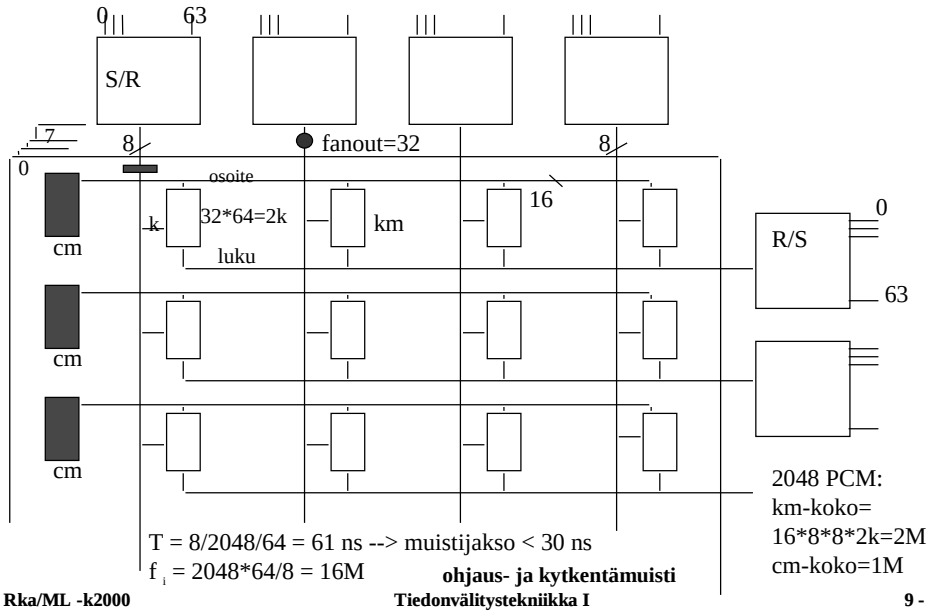
- ✓ **Mitä pitempää väylää annetun nominaalisen ajokyvyn komponentin täytyy ajaa, sitä pienempi on todellinen ajokyky.**
- ✓ **Mitä useampaa komponenttia yksi komponenttilähtö syöttää, sitä enemmän tehoa ja virtaa kuluu.**
- ✓ **Tehon kulutusta voidaan vähentää, jos käytettävissä on herkempiä vastaanottimia.**

Looginen rakenne-esitys matriisikentälle

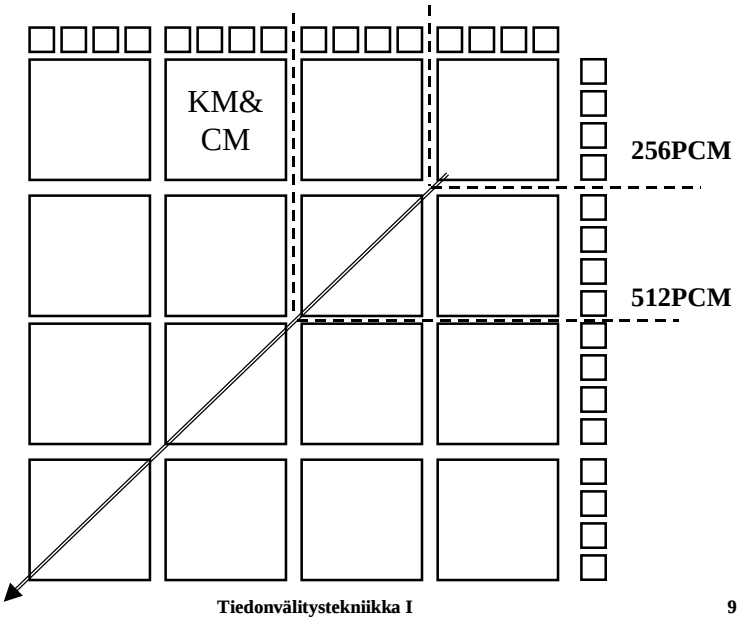


$N = 2^n$

Esimerkki matriisikentästä (DX 200)



Matriisikenttä kasvaa neliöllisesti



Matriisikentän selityksiä

- ✓ **S/R - Sarja-rinnan muunnin. Tulevat aikavälit käännetään rinnakkaismuotoon, jotta kentän sisäisten väylien nopeus pysyisi kohtuullisena.**
- ✓ **R/S - ennen lähtöjä aikavälit käännetään takaisin sarjamuotoon.**
- ✓ **64 PCM:n S/R + R/S on toteutettu yhdelle pistoyksikölle. Käytännöllistä, koska PCM:t ovat kaksisuuntaisia.**
- ✓ **Yhteen kytkentälohkoon voidaan liittää max neljä S/R+R/S:ää. Määrä valitaan vaaditun kapasiteetin mukaan (64, 128, 192 tai 256 PCM:ää).**
- ✓ **Yksi S/R+R/S syöttää max kahdeksaa rinnakkaista kytkentälohkoa. Lohkojen määrä valitaan installaatiossa vaaditun kapasiteetin mukaan (n * 256 PCM:ää).**
- ✓ **Kentän max koko on 2048 PCM:ää. Esimerkki on DX 200 -järjestelmästä.**
- ✓ **Jos tänään halutaan rakentaa isompi kenttä (esim 8K PCM:ää) , valitaan toisenlaisia SRAM:ja, periaate olisi samantapainen (nykyinen maksimikenttä).**

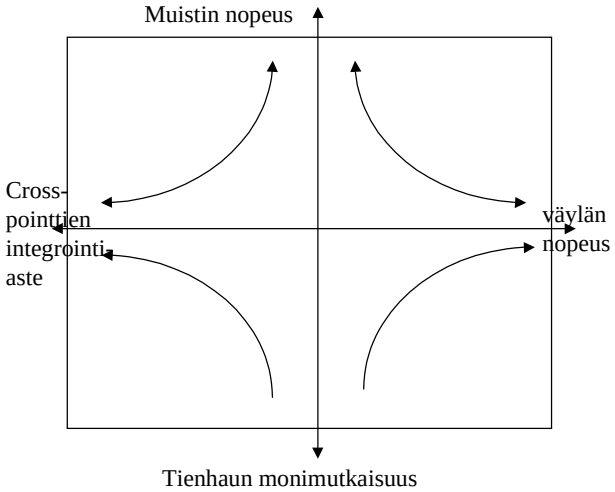
Matriisikentän toiminta

- ✔ **Kirjoitus S/R:stä tapahtuu kaikkiin väylän varrella oleviin kytkentämuisteihin kaikissa “päällekkäisissä“ lohkoissa hyödyntäen S/R:n ajokykyä ja lohkon väyläpurskurin ajokykyä.**
- ✔ **Sama sisältö monistuu max 4x8=32 paikkaan.**
- ✔ **CM sisältö toimii lukuosoitteena kytkentämuistille.**
- ✔ **KM(CM(Lähtö-PCM, lähtö-tsl)) = lähtö**
- ✔ **CM(Lähtö-PCM, lähtö-tsl)**
  - § **lohkossa 2 sisältöbittiä valitsee KM-piirin ja**
  - § **5+6=11 bittiä KM-piirin muistipaikan.**
  - § **loput 3 bittiä osoittavat lohkoa (yksikin bitti riittäisi - tämä lohko/ei-tämä lohko**

Lisää matriisikentän lukuarvoja

- Kytkettävien aikavälien määrä kehyksen aikana: = 2048 x 32 = 64k
- Kytkentämuistissa max 32 x 64 k byteä = 2M
- Kytkentämuisteista luetaan maksimikentässä vain joka 32. Muistipaikka, eli keskimääräinen nopeusvaatimus on pienempi kuin pahimman tapauksen mukaan laskettu muistin nopeusvaatimus.
- Ohjausmuistin max koko = 2ksanaa x 4 x 8 x 8 = 4 x 4 x 8 x 8 kbyteä = 1 Mbyteä.

Kentän teknologiset tradeoffit



Kun tien hakua pyritään yksinkertaistamaan ja siis kytkentää nopeuttamaan ->
- väylänopeudet kasvaa-> tarvitaan nopeampaa muistia
- crosspointtien integrointiaste kasvaa -> tarvitaan nopeampaa muistia

Kun riittävän nopeaa muistia ei ole
--> moniportaiset kentät

Hidas tien haku (BBand)
--> kentän käyttöaste voi laskea
--> ***muistin minimointi voi olla hyödytöntä***