SULAUTETTUJEN SUORITTIMIEN MUISTITYYPIT

Martti Savolainen

Ohjaaja: Timo Rahkonen

ELEKTRONIIKAN JA TIETOLIIKENNETEKNIIKAN TUTKINTO-OHJELMA

2019

Savolainen M. (2019) Sulautettujen suorittimien muistityypit. Oulun yliopisto, Elektroniikan ja tietoliikennetekniikan tutkinto-ohjelma. Kandidaatintutkielma, 46 s

TIIVISTELMÄ

Tässä kirjallisuuskatsauksessa tutustutaan sulautettujen suorittimien muistityyppeihin. Muistiteknologioiden nopea kehittyminen viime vuosina on tuonut perinteisesti sulautetuissa suorittimissa käytettyjen muistityyppien rinnalle myös uusia ratkaisuja. Ferrosähköinen muisti (FRAM) ja faasimuutokseen perustuva muisti (PRAM) edustavat tässä työssä uusia, jo kaupallistettuja muistityyppejä. Sulautettujen järjestelmien suunnittelussa energiankulutus on keskeinen kriteeri. Niinpä muistityyppien ominaisuuksia tarkastellaan ja muistityyppejä vertaillaan keskenään ennen muuta energiankulutuksen näkökulmasta. Lopuksi tutustutaan kahteen mikro-ohjaimeen, joissa käytetään eri muistityypeistä koostuvia muistikonfiguraatioita.

Tämän kirjallisuuskatsauksen perusteella voidaan todeta, että korvaamalla nykyisissä mikro-ohjaimissa yleisimmin haihtumattomana muistina käytetty flashmuisti ferromagneettisella muistilla (FRAM), voitaisiin saavuttaa energiatehokkaampi ja erityisesti langattomiin IoT-sovelluksiin paremmin soveltuva muistiratkaisu. FRAM soveltuisi yksinään käytettäväksi yhdistettynä käsky- ja datamuistina esim. energiankeruusovelluksissa. Vaikka sulautettu muisti voidaan toteuttaa pelkällä FRAM:lla, tarjoaa SRAM:iin ja FRAM:iin perustuva muistikonfiguraatio kuitenkin energiatehokkaamman ratkaisun, jonka energiankulutusta voidaan edelleen pienentää tehokkaalla muistiviittausten optimoinnilla.

Avainsanat: sulautettu, suoritin, muisti, muistityypit.

Savolainen M (2019) Sulautettujen suorittimien muistityypit. University of Oulu, Degree Program in Electrical Engineering, Bachelor's Thesis, 46 p.

ABSTRACT

The memory types used as embedded memory in microcontrollers are discussed in this literature review. There has been a rapid technical development in the area of the memory technologies during the last few years. As a result, it has emerged some new memory types that have been used in the memory configurations of the microcontrollers in parallel with more traditional memory types. The ferromagnetic memory (FRAM) and phase change memory (PRAM) are in the scope of the review. The properties of the memory types are specially considered from the perspective of energy consumption. Finally, two widely used microcontrollers representing two different memory configurations are discussed as examples of embedded memory types.

On the basis of this literature review, it seems that the replacement of the flash as a non-volatile memory by the ferromagnetic memory (FRAM) makes sense from the perspective of energy consumption. FRAM can also be implemented as unified memory which makes it a very attractive memory type for the IoT and energy-harvesting applications too. However, the memory configuration based on SRAM and FRAM is more energy efficient. The energy efficiency of SRAM-FRAM hybrid configuration can be further increased by optimizing the memory access routines.

Key words: embedded, microprocessor, memory, memory types.

SISÄLLYSLUETTELO

LYHENTEIDEN SELITYKSET

BSL	Bootstrap loader
CMOS	Complementary metal oxide semiconductor
DDR	Double data rate
DMA	Direct memory access
DRAM	Dynamic random access memory
DSP	Digital signal processing
EEPROM	Electrically erasable programmable read only memory
EPROM	Erasable programmable read only memory
FET	Field effect transistor
FFT	Fast Fourier transform
FIT	Failures in time
FN	Fowler-Nordheim
FRAM	Ferromagnetic random access memory
IoT	Internet of things
ISP	In-system programming
JTAG	Joint test action group
LEA	Low-energy accelerator
MB	Megabytes
MBps	Megabytes per second
MEMC	Memory controller
MHz	Megahertz
MIMD	Multiple instruction, multiple data
MLC	Multi layer cell
MMC	Multimedia card
MOS	Metal oxide semiconductor
MOSFET	Metal oxide semiconductor field effect transistor
MMU	Memory management unit

MROM	Maskable read only memory
MTBF	Mean time between failures
NVM	Non-volatile memory
OOB	Out of band
OTP	One time programmable
PRAM	Phase-change random access memory
PZT	Lead zirconate titanate
QLC	Quad layer cell
RF	Radio frequency
RISC	Reduced instruction set computer
ROM	Read only memory
RTOS	Real-time operating system
SBT	Strontium bismuth tantalate
SPM	Scratch pad memory
SD	Secure digital
SDRAM	Synchronous dynamic random access memory
SER	Soft error rate
SIMD	Single instruction, multiple data
SLC	Single layer cell
SPI	Serial peripheral interface
SRAM	Static random access memory
SSD	Solid-state drive
SoC	System on circuit
Tcrys	Crystallization temperature
TLB	Translation lookaside buffer
Tmelt	Melting temperature
UART	Universal asynchronous receiver-transmitter
USB	Universal serial bus

1. JOHDANTO

Vuonna 2016 ennakoitiin, että vuoteen 2020 mennessä maailmassa on 50 miljardia itsenäistä esineiden internetiin (IoT) liitettyä laitetta [36]. Energiankeruuteknologian (energy harvesting) kehittymistä ja sen tarjoamia ratkaisuja pidetään yhtenä mahdollisuutena itsenäisten IoT-laitteiden ja langattomien anturiverkkojen käyttöjännitelähteeksi. Laitteen käyttöympäristössä liikkeeseen, lämpöenergiaan, piezosähköisyyteen tai valoon perustuva energiankeruu voi mahdollistaa jatkossa elektroniikan sovellusalan laajentamisen nykyisistä IoT-sovelluksista kaikille päivittäisen elämän alueille kannettavan ja puettavan elektroniikan muodossa. Erityisen lupaavalta näyttää aurinkopaneeleista tuttu valosähköisyyteen perustuva teknologia, jolla ylletään jo 100 mW / cm2 tehotiheyksiin. [44 s. 21 – 26] Sulautettujen järjestelmien päivitettävyys langattoman yhteyden yli, niihin liittyvät tietoturvanäkökulmat, IoT-sovellusten räjähdysmäinen lisääntyminen ja erityisesti uudet energiankeruuseen perustuvat sovellukset asettavat uusia vaatimuksia myös järjestelmän sisältämälle muistille.

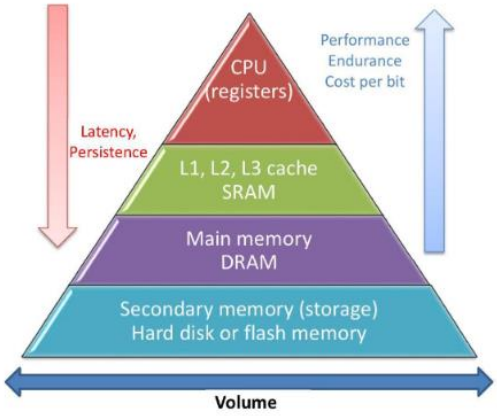
Viime vuosina muistiteknologioissa tapahtunut kehitys on tuonut perinteisesti sulautetuissa järjestelmissä käytettyjen muistityyppien rinnalle uusia kiinnostavia vaihtoehtoja. Tässä kirjallisuuskatsauksessa tutustutaan sulautetuissa suorittimissa käytettyihin muistityyppeihin ja mukaan on pyritty ottamaan myös uusia jo kaupallistettuja muistityyppejä. Suorittimelle integroitu muisti on energiatehokkuuden, huollettavuuden ja käyttöympäristön erityisvaatimusten näkökulmasta usein käyttökelpoisin muistiratkaisu.

2. MUISTI SULAUTETUSSA JÄRJESTELMÄSSÄ

Tässä kirjallisuuskatsauksessa tutustutaan sulautettujen suorittimien sisältämiin muistityyppeihin. Vaikka työssä esitetään muistien ohjelmointiin liittyviä näkökohtia, painottuu tarkastelu laitetasolle. Sulautetuissa järjestelmissä käytettäviin suorittimiin on yleensä integroitu samalle piirille suorittimen lisäksi myös muistia sekä tarvittavat I/O-rakenteet. Tällaista suorittimen lisäksi muistia ja keskeiset I/O-komponentit sisältävää kokonaisuutta kutsutaan mikro-ohjaimeksi. Integroimisasteen kasvaessa piiriä kutsutaan järjestelmäpiiriksi (SoC). Sulautetulla suorittimella viitataan tässä kirjallisuuskatsauksessa mikro-ohjaimen sisältämään suorittimeen tai mikro-ohjainpiirille mahdollisesti integroituihin apusuorittimiin. Mikro-ohjainpiirille integroitu muisti voi olla suorittimelle tai apusuorittimille integroitua muistia tai mikro-ohjaimen sisäistä apusuorittimien kesken jaettua muistia. Piirin ulkopuoliset muistit jätetään tarkastelun ulkopuolelle. [1]

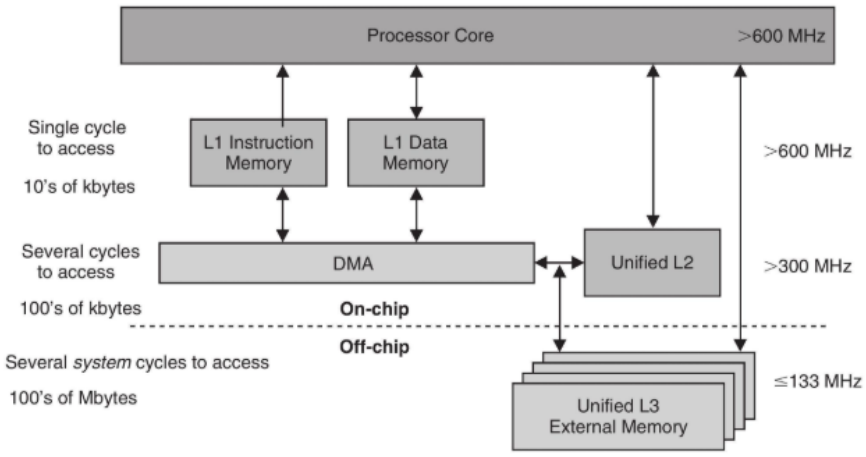
2.1 Sulautetun järjestelmän muistihierarkia

Tietokonejärjestelmän muistihierarkiaa voidaan tarkastella pyramidina (Kuva 1), jossa lähimpänä suoritinta ovat suorituskykyisimmät muistit, kuten siihen integroidut rekisterit ja välimuistit. Mikäli välimuistitasoja on useampia, on tason 1 välimuisti integroitu suorittimelle. Seuraavan muistihierarkian tason muodostaa primäärimuisti, joka tunnetaan tietokoneissa keskusmuistina sekä alimpana oleva sekundäärimuisti, jota kutsutaan myös massamuistiksi. Mitä lähempänä suoritinta muisti sijaitsee muistihierarkiassa, sitä nopeampaa ja suorituskykyisempää sen on oltava, jotta suorittimen ja muistien välisestä nopeuserosta johtuvia pullonkauloja ei synny. Siirryttäessä kauemmas suorittimesta nopeusvaatimus ei ole yhtä keskeinen. Mitä kauemmas suorittimesta hierarkiassa siirrytään sitä suuremmaksi kasvaa myös muistien koko.



Kuva 1. Tietokonejärjestelmien muistihierarkia. [2 s. 16]

Sulautetun järjestelmän muisti noudattaa periaatteessa samanlaista hierarkiaa, vaikka sen rakenne ja toteutus vaihtelevat sovelluskohtaisesti ja riippuvat mm. suoritinarkkitehtuurista (Kuva 2). Suoritinsirulle integroitu nopein muisti, josta käytetään myös nimitystä tason 1 muisti (L1) toimii täydellä kellotaajuudella, eli yksittäinen muistihaku tapahtuu yhden kellojakson aikana. Riippuen suorittimen arkkitehtuurista tason 1 muisti voi olla jaettu erillisiin käsky- ja datasegmentteihin. Tason 1 muisti on kuitenkin kooltaan pieni, joten mikäli laitteen ohjelma tarvitsee enemmän muistitilaa, se on tallennettava tason 2 muistiin, joka voi olla joko suorittimella tai sen ulkopuolella. Tason 2 muistioperaatioihin tarvitaan useampia kellojaksoja. Viiveet kasvavat myös muistin koon kasvaessa, joten tason 2 muistioperaatiot sisältävät tasoon 1 verrattuna huomattavasti pidemmät viiveet [3]. Tason 3 muisti on suoritinpiirin ulkopuolista muistia, joka jää tässä kirjallisuuskatsauksessa tarkastelun ulkopuolelle. Tason 2 muistioperaatiot sisältävät viivettä myös johtuen väyläviiveistä muistin ja suorittimen välillä. Yleensä korkeimman tason muisti on kooltaan muutamia kymmeniä kilotavuja ja tason 2 muisti muutamia satoja kilotavuja. [45 s. 183 – 185]



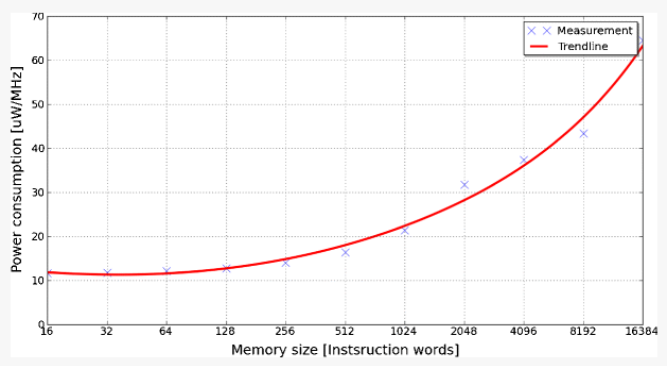
Kuva 2. Muistin hierarkiatasot sulautetussa järjestelmässä. [45 s. 185]

Puolijohdevalmistusprosessien kehittymisen myötä on pystytty kasvattamaan myös suoritinpiirille integroidun muistin kokoa ja tällä hetkellä tason 2 muistit yltävät jo muutaman megatavun kokoluokkaan (Taulukko 3). Mikro-ohjainpiirille integroitujen suorittimen sisäisten rekisterien ja mahdollisen välimuistin lisäksi siihen sisältyy haihtumatonta muistia ohjelmaa, sen tarvitsemia vakioita, sekä laitekonfiguraatiota ja mahdollista käyttöjärjestelmää varten. Mikro-ohjainpiirin sisältämä haihtuva datamuisti on antureiden sekä muiden järjestelmään liitettyjen komponenttien tuottaman tiedon tallentamista varten. Mikäli muisti on integroitu suoritin/mikro-ohjainpiirille siitä käytetään tässä kirjallisuuskatsauksessa nimitystä sulautettu muisti.

Vaikka muistiteknologian kehitys on mahdollistanut entistä isompien muistien integroimisen mikro-ohjaimelle, on järjestelmään mahdollisesti kuuluvat alemman hierarkiatason muistit (taso 3) toteutettu piirin ulkopuolisena muistina. Useimmat mikro-ohjaimet mahdollistavat oikosiirron (DMA) minkä avulla tiedonsiirto ulkopuolisen muistin ja mikro-ohjaimen välillä on mahdollista isommissa lohkoissa suoraan ohi suorittimen ilman, että suoritinta tarvitsee kuormittaa tavukohtaisesti. I/O-komponentit eivät mikro-ohjainpiirille integroituna ole välttämättä yhtä suorituskykyinen ratkaisu kuin ulkoisella apusuorittimella toteutettuna ja sulautettu muisti jää valmistusteknisistä syistä johtuen yleensä kapasiteetiltaan pienemmäksi kuin mitä vastaavalla muistityypillä voitaisiin ulkoisesti toteuttaa. Tästä huolimatta suoritin- tai mikro-ohjainpiirille integroituna saadaan väyläviiveet huomattavasti pienemmiksi millä on myös vaikutusta järjestelmän tehonkulutukseen. Sulautetun

järjestelmän sovelluksissa alhaisen tehonkulutuksen lisäksi integroimisasteen kasvun myötä mahdollistuva pienenempi koko ja kompaktimpi toteutus tukevat erilaisiin käyttöympäristöihin ja huollettavuuteen liittyviä näkökohtia, joilla on sulautettujen järjestelmien suunnittelussa keskeinen merkitys. [1 s. 246 - 249]

Energiankulutuksen näkökulmasta sulautettu muisti on ulkoista muistia huomattavasti suorituskykyisempää. Mikro-ohjainpiirin ulkopuoliseen muistiin tehtävät muistiviittaukset kuluttavat keskimäärin kymmenkertaisen määrän energiaa piirin sisäisiin muistiviittauksiin verrattuna. Energiatehokkaampaan lopputulokseen päästään myös jakamalla sulautettu muisti pienempiin fyysisiin muisteihin (Kuva 3).



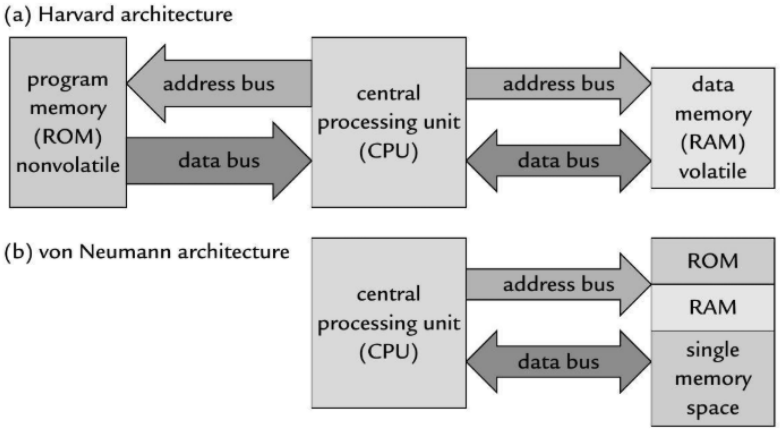
Kuva 3. Esimerkki 16-bittisen SRAM:n tehonkulutus muistin koon funktiona. [4 s. 7]

Tehokkuutta voidaan edelleen parantaa hyödyntämällä erilaisia muistin käytön optimointiin perustuvia tekniikoita. Energiankulutuksen näkökulmasta tarkasteltuna, voidaan todeta että muistin koon kaksinkertaistaminen kaksinkertaistaa myös sen energiankulutuksen. [3 s. 19-20],[4]

2.2 Muisti eri suoritinarkkitehtuureissa

Suoritinarkkitehtuurit voidaan jakaa karkeasti kahteen ryhmään, von Neumann - ja Harvard-arkkitehtuureihin. Harvard-arkkitehtuurin mukaisissa suorittimissa käskymuistille ja datamuistille on varattu erilliset muistisegmentit, joihin liittyvät myös erilliset osoite- ja dataväylät. Tämä mahdollistaa sen, että suoritin voi lukea operandia datamuistista erilliseen datamuistiin samaan aikaan kun se lukee seuraavaa

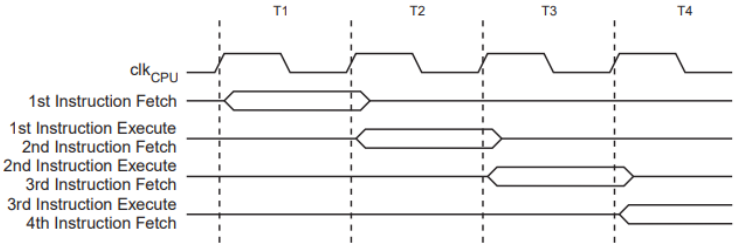
käskyä käskymuistista, mikä nopeuttaa käskyjen suorittamista. Tehokkuusvaatimuksen kasvaessa voidaan hyödyntää lisäksi moniin suorittimiin sisältyvää monitasoista liukuhihnaperiaatetta (pipeline) sekä usean suorittimen avulla toteutettua käskyjen rinnakkaista suorittamista (MIMD, SIMD). Rakenteeltaan yksinkertaisemmassa von Neumann-arkkitehtuurissa on yhteinen muistiavaruus sekä käskyille että datalle, johon liittyy vain yksi osoite- ja yksi dataväylä. [5, s. 23 – 24]



Kuva 4. Muisti Harvard- ja von Neumann-arkkitehtuureissa. [5 s. 23]

Käsky- ja datamuistin erottelu Harvard-arkkitehtuurissa merkitsee, että esim. samalla 8-bittisellä käskymuistiin viittaavalla osoitteella, joka sisältyy 32-bittiseen datamuistiin viittaavaan osoitteeseen, voidaan lukea muistista käskyä samanaikaisesti, kun suoritin kohdistaa hakua datamuistiin. Harvard-arkkitehtuurin yleisin modifioitu versio ei pidä ehdottoman tiukasti kiinni käskyjen ja datan erottelusta, vaan se käyttää apuna käskyille ja datalle varattuja erillisiä välimuisteja. Suorittaessaan välimuistissa olevaa käskyä suoritin toimii puhtaasti Harvard-mallin mukaan samalla kun se suorittaa yhteiseen muistiavaruuteen kohdistuvia operaatioita von Neumanin mallin mukaisesti. Tätä modifioitua Harvard-arkkitehtuuria (Kuva 5) sovelletaan nykyään useissa sulautetuissa suorittimissa. Monet suoritinvalmistajat käyttävät tuotteissaan molempia arkkitehtuureja. Niinpä esim. sulautetuissa järjestelmissä paljon käytetty ARM:n Cortex-M0 perustuu von Neumann -arkkitehtuuriin, kun taas vaativampiin sulautetun sovelluksiin tarkoitettu Cortex-M4 käyttää Harvard-arkkitehtuurin modifioitua versiota. Esimerkkeinä Harvard-arkkitehtuuria noudattavista suoritinarkkitehtuureista voidaan mainita myös useiden

puhelinvalmistajien omissa toteutuksissaan käyttämä ARM-arkkitehtuuri, IBM:n käyttämä Power-arkkitehtuuri sekä Intelin, Cyrixin ja AMD:n käyttämä x86-arkkitehtuuri. [1 s. 273 – 277] [7], [8], [9], [10]



Kuva 5. Rinnakkainen käskyn haku ja suoritus Harvard-arkkitehtuuria soveltavassa Atmega328-mikro-ohjaimessa. [6]

2.3 Muistiavaruuden sisäinen organisointi

Tietokonejärjestelmissä muisti organisoidaan hierarkiseksi rakenteeksi, kuten edellä on esitetty. Muistin organisoinnilla viitataan myös siihen, miten muistiavaruutta käytetään eri muistityyppien kesken, jolloin voidaan puhua muistiavaruuden sisäisestä organisoinnista. Huolimatta siitä, että muisti on organisoitu useiksi eri kokoisiksi muistialueiksi, suoritin näkee sen kuitenkin yhtenäisenä muistikarttana, jossa muistikartta määrittelee yksikäsitteisesti eri muistialueiden osoitteet. [43 s. 112]

2.3.1 Välimuisti

Päämuistin ja välimuistin välillä tapahtuvat luku- ja kirjoitusoperaatiot tapahtuvat yhden tai useamman sanan kokoisissa lohkoissa. Nämä lohkot sisältävät kopioita paljon käytetyistä muistisisällöistä sekä niiden sijainnit päämuistissa ilmoittavista tageista (tags). Kirjoitettaessa muistiin, on vastaava sisältö kirjoitettava myös tason 1 välimuistiin, mikäli sitä vastaava alkuperäinen sisältö siellä jo on. Tähän on olemassa kaksi strategiaa, ’write-through’ ja ’write-back’. Ensin mainittu tarkoittaa, että data kirjoitetaan joka kerta sekä välimuistiin että päämuistiin. Jälkimmäisellä viitataan tilanteeseen, että data on alun perin kirjoitettu vain välimuistiin ja se kirjoitetaan päämuistiin vasta kun se korvataan uudella datalla. Luettaessa muistista suoritin

tarkistaa datan ensiksi tason 1 välimuistista. Mikäli se on siellä, saadaan välimuistiosuma (cache hit) ja suoritin suorittaa tehtävänsä loppuun. Mikäli dataa ei löydy tason 1 välimuistista, saadaan välimuistihuti (cache miss) ja tieto on haettava korkeamman tason välimuisteista tai päämuistista. Tason 1 välimuisti on yleensä toteutettu Harvard-periaatteella, joten siinä on erilliset muistialueet ja sisäiset väylät käskyille ja datalle.

Suorasijoittavassa välimuistissa (direct mapping cache) välimuisti on jaettu lohkoihin, jotka sisältävät datan lisäksi kaksi tagia, joista toinen ilmoittaa onko lohko käytettävissä (valid tag) ja toinen tagi (tag) ilmoittaa päämuistin osoitealueen, josta tietoa voidaan välimuistiin kirjoittaa. Tagi johdetaan päämuistipaikan osoitteesta ja se koostuu tagista, lohkoon viittaavasta indeksistä ja lohkon sisällä muistipaikkaan viittaavasta siirtymästä (offset). Assosiatiiviseen välimuistiin on tallennettu päämuistissa sijaitsevaan dataan viittaava osoite sekä kopio datasta. Haettaessa assosiatiivisestä välimuistista tietoa sen sijaintia ei tiedetä, joten se on pääteltävä päämuistiin viittaavan osoitetiedon perusteella. Täysassosiatiivisessa välimuistissa päämuistista haettava data voidaan sijoittaa vapaasti mihin tahansa välimuistipaikkaan, kun taas joukkoassosiatiivisessa (set associative) välimuistissa sijoittelumahdollisuudet on rajoitettu muutamiin mahdollisiin paikkoihin välimuistissa. Siinä lohkot muodostavat suurempia kokonaisuuksia, joita kutsutaan joukoiksi (set). Välimuistiin viittaaminen tapahtuu lohkotasolla suorasijoittavan välimuistin periaatteella ja joukkotasolla assosiatiivisesti. Tällä saavutetaan täysassosiatiiviseen välimuistiin verrattuna se hyöty, ettei hakua tarvitse kohdistaa koko välimuistiin. [17 s. 419-420] Sulautetuissa järjestelmissä suorasijoittava välimuisti on paljon käytetty koska siinä muistihakua kohti laskettu energiankulutus on pienempi kuin joukkoassosiatiivisessa välimuistissa. [4]

2.3.2 Muistinhallinta

Muistinhallinta voi olla toteutettu joko muistiohjaimen (MEMC) tai muistinhallintayksikön avulla (MMU). Monissa suorittimissa on nykyään sisäinen muistinhallintayksikkö. Muistiohjaimen tehtävänä on mahdollistaa suorittimen kommunikoiminen välimuistin ja eri muistityyppejä edustavien muistien kanssa.

Muistiohjain välittää suorittimelta tulevan muistikutsun kohdistamalla sen kohteena olevaan fyysiseen muistiosoitteeseen. Mikäli muistiohjain liittyy vain yhteen muistityyppiin käytetään siitä nimitystä välimuistiohjain, DRAM-ohjain jne. Muistinhallintayksikkö kääntää loogiset muistiosoitteet fyysisiksi osoitteiksi, sekä ohjaa eri muistityyppien ja väylän sekä suorittimen välistä tiedonsiirtoa ja generoi keskeytykset. Muistinhallintayksikön tehtävänä on myös huolehtia muistin segmentoinnista, jolla tarkoitetaan loogisen muistiavaruuden jakamista suurempiin vaihtelevan kokoisiin osiin. Näille osille voidaan määritellä erilaisia oikeuksia, esim. jaettu resurssi, luku- ja kirjoitusoikeus tai pelkkä lukuoikeus. Mikäli muistihakua ei voida kohdistaa mihinkään segmenttiin, MMU generoi keskeytyksen. Muistinhallintayksikkö voi käyttää myös tason 1 välimuistia tai osia siitä puskureina kääntäessään loogisia osoitteita fyysisiksi (’translation lookaside buffer’ TLB). [43 s. 109 – 111]

3. SULAUTETUISSA JÄRJESTELMISSÄ KÄYTETTÄVIÄ MUISTITYYPPEJÄ

Tietokonejärjestelmissä käytettävät muistityypit voidaan luokitella monien eri ominaisuuksien tai käytettyjen valmistusteknologioiden perusteella. Tässä kirjallisuuskatsauksessa muistit luokitellaan haihtuviin ja haihtumattomiin muisteihin. Tarkastelussa keskitytään kuvaamaan yleisellä tasolla muistityyppejä, joita kaupallisissa ja yleisesti käytössä olevissa mikro-ohjaimissa käytetään osana sulautettuja järjestelmiä.

3.1 Haihtuvat muistit

Haihtuvaan muistiin tallennettu tieto häviää muistista jännitekatkoksen myötä ellei sitä ole varmennettu paristolla. Seuraavassa tarkastellaan haihtuvia muistityyppejä, joita käytetään sulautetuissa järjestelmissä.

3.1.1 DRAM

DRAM (Dynamic Random Access Memory) on ollut jo yli 30 vuoden ajan pääasiallinen muistityyppi tietokoneiden keskusmuisteissa ja erilaisten tietokonejärjestelmien datamuistina. DRAM:n muistikapasiteetti on kasvanut samaa tahtia puolijohdeteknologian kehityksen kanssa, joka Mooren lain mukaan on merkinnyt kapasiteetin kaksinkertaistumista aina 18 kk välein. Kehitys ei ole näkynyt ainoastaan muistikapasiteetin, vaan myös sulautetuille järjestelmille tärkeän tiedonsiirtonopeuden kasvuna ja virran kulutuksen pienenemisenä sekä toiminta- että lepotilassa. [11, s. 1461]

DRAM edustaa asynkronista haihtuvaa luku-kirjoitusmuisti -tyyppiä, jossa bitti tallennetaan yksittäisen muistisolun sisältämään kondensaattoriin. Muistisoluun kuuluu kondensaattorin lisäksi myös kytkimenä toimiva MOSFET-transistori. Muisti koostuu muistisolujen muodostamasta matriisista, jossa jokaiseen muistisoluun voidaan viitata aktivoimalla ensin rivi ja sen jälkeen sarake, joiden risteyskohdassa muistisolu sijaitsee. Koska bitti on tallennettu kondensaattoriin, täytyy sitä virkistää jotta tieto ei katoa virrankatkaisun yhteydessä mutta myös jokaisen lukuoperaation

jälkeen. Tämä selittää DRAM:in kohtalaisen suurta energiankulutusta. Koska muistisolu koostuu yhdestä transistorista ja kondensaattorista on DRAM:in valmistaminen halpaa ja sillä voidaan toteuttaa edullisesti suuria muistikokonaisuuksia suhteellisen pienelle puolijohdepinta-alalle. [12] DRAM:in virkistysvaatimus asettaa tiukat ehdot siinä esiintyville vuotovirroille. Muistisolun sisältämän transistorin ja kondensaattorin vuotovirtojen huomioonottamisen lisäksi korostuvat pakkaustiheyden kasvaessa myös valmistusprosessin tarkkuuteen liittyvät vaatimukset. [13]

Koska tiedon säilyminen DRAM:ssa perustuu muistikondensaattoreiden jatkuvaan virkistämiseen, on DRAM:n ongelmana energiankulutus erityisesti lepotilassa. Tämä ilmenee varsinkin muistikapasiteetin kasvaessa, jolloin muistisolun jännitteen tulee laskea samassa suhteessa muistisolun fyysisten mittojen kutistuessa. Tästä seuraa, että myös kynnysjännite laskee ja virkistysjännitteen saturoitumisen seurauksena tehonkulutus kasvaa lepotilassa. [11 s. 1461 – 1464]

3.1.2 SDRAM

Tiedonsiirron nopeuttamiseksi on DRAM:sta kehitetty kellosignaaliin synkronoitu SDRAM, joka toimii muuten samalla periaatteella kuin DRAM mutta kykenee antamaan peräkkäistä dataa ulos peräkkäisillä kellojaksoilla. Hakuaika ei synkronisella DRAM:lla (SDRAM) kuitenkaan ole välttämättä nopeampi, koska muisti odottaa aina kellojakson reunaa ennen kuin sille voi antaa komentoja. DDR-SDRAM kaksinkertaistaa siirrettävän tiedon määrän SDRAM:iin verrattuna, koska siinä tiedonsiirto tapahtuu sekä kellojakson nousevalla että laskevalla reunalla. Myös DDR2-tyypin SDRAM siirtää tietoa kellojakson molemmilla reunoilla. Se mahdollistaa kuitenkin nopeamman tiedonsiirron, koska DDR2-SDRAM toimii ulkoista kellotaajuutta puolet pienemmällä taajuudella, mikä antaa mahdollisuuden kasvattaa ulkoista kellotaajuutta ilman, että muistin sisäinen kellotaajuus aiheuttaa ongelmia. Tämä mahdollistaa sen, että samalla valmistustekniikalla voidaan valmistaa lähes puolta nopeampia muistipiirejä. DDR3-tyypin SDRAM:ssa suorituskykyä ja tehonkulutusta on edelleen parannettu. Sen suorituskyky on DDR2:een verrattuna kaksinkertainen ja se toimii pienemmällä käyttöjännitteellä

kuin DDR2 tukien samalla 64-bittistä tiedonsiirtoa. Lisäksi se kykenee lukemaan jokaisella sisäisellä kellojaksolla 8 bittiä tietoa. DDR4-tyyppinen SDRAM toimii vieläkin pienemmällä, 1,2 V käyttöjännitteellä sekä antaa mahdollisuuden käyttää jopa 3200 MHz:n kellotaajuutta. [14], [15]

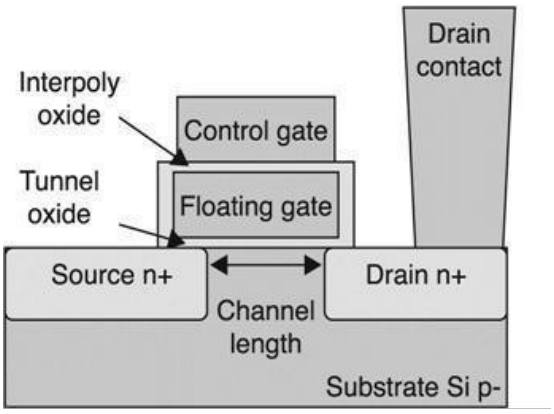
3.1.2 SRAM

DRAM-muistia nopeampaa SRAM:ia (Static random access memory) ei tarvitse virkistää, koska siinä tieto tallennetaan kiikkupiiriin, joka säilyttää tilansa kunnes laitteesta katkaistaan jännite. Tiedon katoaminen muistista käyttöjännitteen katkaisun jälkeen voidaan estää paristovarmennuksella. Paristovarmennettu SRAM-muisti on käytännössä haihtumatonta muistia ja hyvin käyttökelpoinen pientä tehonkulutusta edellyttäviin sovelluksiin, koska siinä vuotovirrat ovat pieniä, pariston vuotovirtojen suuruusluokkaa [16 s. 76]. SRAM-muisti on nopeampaa kuin DRAM johtuen siitä, että data voidaan lukea suoraan muistin lähdöstä ilman erityistoimenpiteitä. SRAM:iin voidaan viitata koko osoiteväylän leveydeltä toisin kuin DRAM:iin, jossa osoite joudutaan kirjoittamaan kahdessa osassa, mikä sekin tekee SRAM:sta nopeamman. Johtuen siitä, että yhteen muistisoluun tarvitaan useita transistoreita, tarvitsee SRAM kuitenkin 6-8 kertaa enemmän tilaa kuin DRAM. Tästä johtuen myös SRAM:n tehonkulutus on DRAM:ia suurempi. Nopeutensa vuoksi SRAM:ia käytetään paljolti puskurimuistina ja välimuistina. Sulautetuissa järjestelmissä mikro-ohjaimelle integroitu sulautettu datamuisti on usein toteutettu SRAM:lla. [16 s. 417], [18]

Alun perin kahdella transistorilla toteutettu muistisolun sisältämä kiikku tehdään nykyään kuudella transistorilla. Vaikka sen tehonkulutus on pieni, se on erittäin vakaa ja sietää hyvin muistissa käytetyn polykidepii-teknologian riesana aiemmin olleita alfa-hiukkasia. Se, että muistisolu koostuu useasta transistorista kasvattaa kuitenkin SRAM:n kokoa DRAM:iin verrattuna. Näistä seikoista johtuen SRAM:ia käytetään sulautetuissa järjestelmissä ennemmin suoritinpiirille integroituna välimuistina kuin itsenäisenä muistipiirinä. [18]

3.2 Haihtumattomat muistit

Kannettavan elektroniikan, IoT:n ja älypuhelinkehityksen myötä on muun kuin magneettiseen tallennukseen perustuvan haihtumattoman muistin kysyntä voimakkaasti lisääntynyt. Muuhun kuin magneettiseen tallennukseen perustuvat haihtumattomat muistit on toteutettu käytännössä viimeisen 40 vuoden ajan perustuen nk. kelluvan hilan konseptiin (Kuva 1). EPROM ja EEPROM olivat pitkään ainoat haihtumattomat muistiratkaisut, jotka sovelsivat kelluvan hilan periaatetta. Siinä MOS-transistorin rakennetta on muutettu siten, että hilan ja kanavamateriaalin väliin on muodostettu eristeellä ympäröity kelluva hila (floating gate). Hilaan injektoituva varaus säilyy eristeen eristyskyvystä riippuen hyvinkin pitkän ajan.



Kuva 6. MOS-transistori kelluvalla hilalla. [39 s. 41]

Muistiin ”kirjoitettu” varaus luetaan mittaamalla nieluvirtaa. Neutraali (positiivisesti varattu) kelluva hila edustaa loogista tilaa ’1’ ja negatiivisesti varattu tilaa ’0’. [37]

3.2.1 EEPROM

EPROM on kerran ohjelmoitava lukumuisti, joka ohjelmoidaan yleensä valmistajan toimesta. Se voidaan tyhjentää vain voimakkaan UV-valon avulla, mikä edellyttää kalliita kotelointiratkaisuja. EEPROM voidaan tyhjentää sähköisesti, joten sen uudelleen ohjelmoiminen on helpompaa. Nykyaikaisen EEPROM:in kestävyys on luokkaa 10^6 kirjoitusjaksoa. EPROM:sta on kehitetty myös MROM (Maskable

ROM) ja OTP-ROM (One time programmable ROM), jotka voidaan valmistaa edullisemmassa muovikotelossa. EPROM-muisteja on aiemmin käytetty lähinnä ohjelmien tallentamiseen kun taas EEPROM soveltuu myös käyttäjän toimesta tapahtuvaan vakioiden ja taulukoiden tallentamiseen. Ohjelmiston päivitettävyysvaatimukset sekä EEPROM:in lukuviiveisiin verrattuna jopa satakertaiset kirjoitusviiveet ovat tehneet EPROM:sta ja EEPROM:sta sulautettuna käskymuistina huonosti nykyisiin järjestelmiin soveltuvan. Flashmuistin ominaisuudet tekevät siitä ylivertaisen EEPROM:iin verrattuna, joskin EEPROM:iin voidaan kirjoittaa tavu kerrallaan, mikä ei ole periaatteessa mahdollista NAND-flashiin. Pienehköjä EEPROM-muisteja käytetään kuitenkin edelleen jossain määrin flashin rinnalla laitteistokonfiguraatioiden tallentamiseen. [37]

3.2.2 Flashmuisti

Flashmuisti perustuu samalle kelluvan hilan periaatteelle, kuin sen syrjäyttämät EPROM ja EEPROM. Flashteknologioissa on tällä hetkellä kaksi pääsuuntausta, NOR-flash ja NAND-flash. NOR-flash on pääasiallinen flashmuistityyppi ohjelman ja käyttäjädatan tallentamiseen, kun halutaan mahdollistaa rinnakkaismuotoinen tiedonsaanti. NAND-flash mahdollistaa vain sarjaliitynnän mutta NOR-flashia halvemman hinnan ja suuremman pakkaustiheyden ansiosta se on käyttökelpoisempaa massamuisti- ja muistikorttisovelluksissa. NOR-tyypin flashissa muistisolujen MOS-transistorit on rinnakkain yhdistetty yhteiseen maatasoon ja niiden nielut on kytketty bittilinjoihin. Varauksen säilymistä kelluvassa hilassa edistää hilan ja kanavamateriaalin välinen suuri energiakaistaero. Sulautettujen järjestelmien yhteydessä erityisesti korkean tason integraatiota toteuttavat järjestelmäpiirit hyötyvät NOR-flashteknologian nopeudesta ja yhteensopivuudesta logiikkaprosessien kanssa. [37 s. 44 - 46]

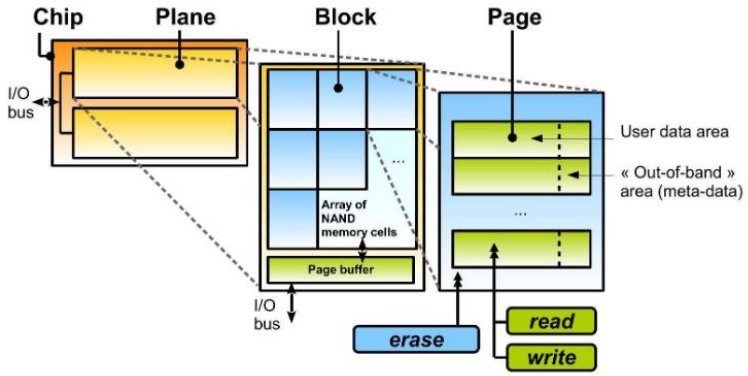
NOR-flasheja hitaammin kehittyneet NAND-flashmuistit ovat nykyään magneettisen tallennuksen rinnalla kilpailevaa massamuistiteknologiaa. Yksittäiset muistisolut perustuvat samaan kelluvan hilan periaatteeseen kuin NOR-flashit mutta ne on yhdistetty eri tavalla. NAND-flashmuisti on rakennettu yhdistämällä muistisolut sarjaan maan ja bittilinjan välillä. Tämän ansiosta pakkaustiheys on NOR-flasheja

suurempi mutta koska jokaiseen muistisoluun kohdistuvat luku- ja kirjoitusoperaatiot kulkevat läpi muiden sarjaan kytkettyjen muistisolujen ketjun, kasvavat viiveet huomattavasti. Tavallisesti sarjaan kytkettyjen muistisolujen lukumäärä on 16 tai 32, jolloin NOR-flashien muutaman kymmenen nanosekunnin luokkaa olevat viiveet kasvavat NAND-flasheissa mikrosekuntien luokkaan. Tämä käytännössä estää niiden käytön rinnakkaisliitettävinä hajasaantimuisteina jolloin ainoaksi tarkoituksenmukaiseksi tavaksi jää käyttää niitä sarjaliitettävinä muisteina. NAND-flashien ohjelmointi tapahtuu Fowler-Nordheim -tunnelointitekniikalla (FN), joka on NOR-flashien ohjelmoinnissa käytettävään 'Channel hot-electron injection' -menetelmään verrattuna noin 300 kertaa hitaampaa. Hitaus on seurausta hilan kanavamateriaalista erottavan tunnelioksidikerroksen yli vaikuttavasta sähkökentästä. Toisaalta FN-tunnelointi mahdollistaa kirjoitusoperaation pienellä virrankulutuksella, joten NAND-flashit soveltuvat erinomaisesti massamuistisovelluksiin. NAND-flashien pakkaustiheys on NOR-flasheja puolet suurempi, mikä myös tekee NAND-flasheista niitä paremmin massamuistiksi soveltuvia. [37 s. 48]

Koska NOR-flashin lukuviiveet ovat 20 ns luokkaa, se soveltuu hyvin ohjelmamuistiksi. NOR-flashilla kirjoittaminen on lisäksi mahdollista tavu kerrallaan toisin kuin NAND-flashilla, joissa kirjoitetaan sivu kerrallaan. NOR-flashin kirjoitusviiveet ovat kuitenkin huomattavasti suuremmat. NAND-flashin halvempi hinta, suurempi tallennustila ja NOR-flashia pienemmät lukunopeudet mutta toisaalta myös pienemmät kirjoitusviiveet tekevät siitä hyvin massamuistiksi soveltuvan muistityypin. Näistä seikoista johtuu, että NAND-flash on pääasiallinen flashteknologia, joka on myös kehittynyt nopeammin. NAND-flashia hyödynnetään massamuistina SSD-kovalevyissä, USB-avaimina, SD- ja MMC-muistikortteina. Tässä kirjallisuuskatsauksessa flashillä viitataan jatkossa NAND-flashiin. [2 s. 23]

Flashin luku-, kirjoitus- ja pyyhkimisoperaatiot kohdistuvat isompiin kokonaisuuksiin, joita kutsutaan tasoiksi, sivuiksi ja lohkoiksi (Kuva 4). Flashmuistipiirin tasot koostuvat lohkoista, jotka puolestaan jakaantuvat sivuihin. Muistiin liittyvät kirjoitus- ja lukuoperaatiot tapahtuvat multipleksatun 8- tai 16-

bittisen I/O-väylän kautta, mikä mahdollistaa sen että käskyt, osoitteet ja data jakavat saman väylän. [2 s. 18]



Kuva 7. Flashmuistin hierarkinen rakenne. [2 s. 18]

Flashmuistisivu jakautuu käyttäjän tallentamalle tiedolle varattuun alueeseen sekä OOB-alueeseen, joka sisältää tallennettua dataa koskevaa metatietoa, kuten pariteetti-informaation virheenkorjausta varten ja osoitteen käännöstä varten tarvittavat tiedot. Uusissa SLC- ja MLC-flasheissa sivuista on muodostettu 32, 64 tai 128 sivun lohkoja. Niiden sisältämien käyttäjän tallentamalle datalle varattujen alueiden koot voivat olla 4096 tai jopa 8192 tavua, jolloin OOB-alueet ovat 128 tavun kokoisia. Flashmuistipiiri sisältää 1, 2 tai 4 tasoa joissa voi olla vaihteleva määrä lohkoja. Yhden tason sisältämien lohkojen muodostamaa kokonaisuutta kutsutaan myös NAND-matriisiksi. Sivu pitää NAND-matriisin lisäksi sisällään sivupuskurin tai sivurekisterin, jonka kautta sivu liittyy I/O-väylään. [2]

Lukuoperaatioiden yhteydessä koko sivu siirretään puskurimuistiin ja kirjoitusoperaatiossa I/O-väylältä vastaanotettava tieto luetaan puskuriin sekä siirretään lopuksi NAND-matriisiin. Tyhjennys tapahtuu lohkotasolla, jolloin kaikki lohkon sisältämät sivut pyyhitään. Vaikka yleensä luku- ja kirjoitusoperaatiot koskevat kokonaisia sivuja, tukevat jotkin piirit myös sivujen osittaista kirjoittamista ja lukemista. [2 s. 26 - 29] NAND-flashin sisäisestä rakenteesta johtuen ennen kirjoittamista täytyy sivulla mahdollisesti jo oleva tieto pyyhkiä pois. Sivun tyhjentäminen merkitsee koko lohkon tyhjentämistä, koska flashissa tyhjennys tapahtuu lohkoittain. Pyyhkimis-kirjoittamisoperaatioon liittyy merkittäviä viiveitä, koska se edellyttää monimutkaisia muistinhallintatoimenpiteitä (Taulukko 1). Perinteinen SLC flashmuisti kykenee tallentamaan yhden bitin. Kasvattamalla

puolijohderajapinnan kynnysjännitetasojen määrää on mahdollista tallentaa useampia bittejä yhteen muistisoluun. Kynnysjännitetasojen määrän kasvattaminen nelinkertaiseksi mahdollistaa kahden bitin tallentamisen yhteen soluun (MLC-flash).

Taulukko 1. Flashmuistiin kohdistuvien luku-, kirjoitus- ja pyyhkimisoperaatioiden viiveitä ja tehonkulutusarvoja [2 s. 30], *[18]

Type	Page size (bytes)	Block size (KB)	Latency (us)			Power consumption (mW)				Source
			Read	Write	Erase	Read	Write	Eras.	Idle	
SLC	4096	256	20	300	1200	19.1	56.0	25.5	13.3	[GRU 09]*
SLC	2048	128	20	200	2000	58.8	78.4	47.6	17.0	[GRU 09]*
SLC	2048	128	20	200	2000	41.1	59.9	35.5	7.1	[GRU 09]*
SLC	2048	128	20	200	500	27.2	35.0	25.3	2.9	[GRU 09]*
SLC	2048	128	20	200	1200	29.9	35.0	20.0	2.9	[GRU 09]*
SLC	2048	128	20	200	2000	35.3	55.2	30.9	2.7	[GRU 09]*
SLC	2048	128	25	200	2000	49.5 - 99			3.3	[LIU 10]
SLC	512	16	12	200	2000	-	-	-	-	[LIM 06]
SLC	2048	256	60	800	1500	49.5 - 99			3.3	[LEE 11]
SLC	2048	128	77	252	1500	-	-	-	-	[KIM 12b]
SLC	4096	512	25	200	700	66 - 165				[LEE 14]
SLC	2048	128	25	200	1500	33 - 165			3.3	[PAR 08]
MLC	4096	1024	30	300-1500	3500	75.9	94.7	70.6	8.5	[GRU 09]*
MLC	4096	512	40	300-1500	3600	66.3	82.3	57.0	11.2	[GRU 09]*
MLC	2048	256	20	300-1100	2800	54.0	58.9	42.4	12.7	[GRU 09]*
MLC	4096	512	110	400-2000	2800	112.0	132.2	111.8	27.3	[GRU 09]*
MLC	4096	512	165.6	905.8	1500	-	-	-	-	[KIM 12b]

Kynnysjännitetasojen määrää on mahdollista kasvattaa edelleen, jolloin päästään moninkertaiseen tallennustiheyteen, kunhan kynnysjännitetasot asetetaan tarkasti niin etteivät ne mene limittäin. [40 s. 160]

Taulukkoon 1 kerättyjä SLC- ja MLC-flashmuisteihin liittyviä luku-, kirjoitus- ja pyyhkimisviiveitä on tarkasteltu vertailemalla eri sivu- ja lohkokokoja edustavia flasheja. Tutkimuskirjallisuudesta ja datalehdistä peräisin olevat tiedot osoittavat, että viiveissä on suuria vaihteluita. Taulukossa lähteenä käytetyn simulaatiotutkimuksen (GRU 09) tulokset vahvistavat tämän samalla kun ne toteavat sulautetun järjestelmän suunnittelun kannalta keskeisen ongelman, valmistajien

datalehdissä antamat tiedot ovat ilmoitettujen viiveiden osalta usein harhaanjohtavia. [19 s. 10]

NAND-flashtekniikka edustaa tällä hetkellä ylivoimaisesti kustannustehokkainta muistiratkaisua haihtumattomissa muisteissa. NAND-flashin pakkaustiheyttä voidaan edelleen kasvattaa siirtymällä kolmiulotteiseen NAND-arkkitehtuuriin. Vertikaaliseen NAND-valmistusteknologiaan verrattuna sen etu on, että valmistuksessa selvitään pienemmällä maskimäärällä, mikä alentaa valmistuskustannuksia. Kyetäkseen kilpailemaan kustannustehokkuudessa planaarisen NAND-teknologian kanssa sen kerroslukumäärää on pyritty kasvattamaan. Monikerrostekniikan ohella myös monisiruarkkitehtuuri (multi-die architecture) on mahdollistanut pakkaustiheyksien kasvattamisen entisestään. [21]

Teknologisen kehityksen vauhti on ollut nopeaa flashmuisteissa. Neljä vuotta sitten Intel kertoi kehittäneensä Micronin kanssa 32-kerroksisen 1 teratavun MLC-flashmuistin, jossa yhteen muistisoluun voidaan tallentaa 4 bittiä. Viime vuonna Intel tiedotti kaupallistaneensa QLC-flashin, jossa yhteen muistisoluun kyetään tallentamaan 4 bittiä samalla kun se kertoi päättäneensä 64-kerroksiseen rakenteeseen perustuvan toisen sukupolven 3D NAND-flashin testiohjelman ja aloittaneensa 96-kerroksisen 3D NAND-flashin kehitystyön. [20] Luku- ja kirjoitusviiveisiin liittyvät tutkimukset ovat osoittaneet, että MLC-piirien viiveet voivat vaihdella merkittävästi johtuen niiden mikroarkkitehtuurista (Taulukko 1). Uusien piirinvalmistustekniikoiden mahdollistamissa monikerroksisissa NAND-flasheissa on todettu, että vaikka monisiru- ja monikerrosarkkitehtuuri parantaa kirjoitusoperaatioiden suorituskykyä yli 90%, saadaan lukuoperaatioiden tehokkuutta kasvatettua vain noin 10%. Tämän on todettu johtuvan kuitenkin pääasiassa I/O-väylään, ei itse flashiin liittyvistä pullonkauloista. [21 s. 2]

Flashin heikko kirjoituskestävyys on yksi siihen liittyvistä keskeisistä ongelmista. Uudelleenkirjoituskertojen määrä vaihtelee tekniikoittain siten, että kun SLC-soluun voidaan kirjoittaa 100 000 kertaa, niin MLC-soluun voidaan kirjoittaa vain noin 4000 kertaa. Lukemisen suhteen ei ole samanlaisia rajoituksia. Flasheista koostuvalla SSD-levyllä tietoa järjestellään sisäisesti siten, että uudelleenkirjoituskertojen

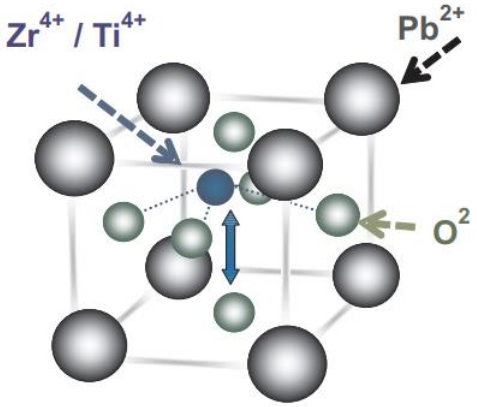
lukumäärää kontrolloidaan kierrättämällä dataa solujen välillä, millä pyritään siihen että muistisoluja käytetään mahdollisimman tasaisesti. [22] Teknologian kehittämisen nykyvaiheessa 24-kerroksisen 3D NAND-flashin yhdistäminen MLC-flashmuistin kanssa mahdollistaa monoliittisen muistimoduulin rakentamisen, joka tällä hetkellä näyttää kustannustehokkaimmalta haihtumattoman muistin ratkaisulta. Sitä käytetään mm. SSD-sovelluksissa. [37 s. 67] Flashmuistin kestoa rajoittava kirjoitus- ja pyyhkimiskertojen rajallinen määrä johtuu siitä, että jokaisen kirjoitus- ja pyyhkimisoperaation myötä MOS-rakenteen oksidikerrokseen jää loukkuun elektroneja, jotka laskevat ajan myötä pn-liitoksen kynnysjännitettä. Flashmuistin luku- ja kirjoitusoperaatioihin liittyy myös muita rajoitteita, joista tallennettuun bittiin liittyvä invertoitumismahdollisuus (bitflips) voi merkitä että kirjoitettu ja luettu data poikkeavat toisistaan. Tämä on seurausta luku- ja pyyhkimisoperaatioihin liittyvien korkeiden jännitteiden aiheuttamasta epästabiiliudesta, mikä korostuu pakkaustiheyden kasvun myötä kutistuvien fyysisten mittojen seurauksena. Ongelman ratkaisemiseksi on kehitetty erilaisia muistinhallinta- ja virheenkorjausmetodeja. [2 s. 30 - 37]

3.2.2 Ferrosähköinen muisti

Ferrosähköinen muisti (FeRAM, F-RAM tai FRAM) edustaa uusien muistiteknologioiden joukossa kypsintä, jo kaupallistettua teknologiaa. Ferrosähköinen muisti perustuu ferrosähköisen materiaalin kykyyn polarisoitua pysyvästi sähkökentässä. Ferrosähköisissä muisteissa hyödynnetään pääasiassa kahta toteutustapaa. Toinen perustuu perovskiittisen kiderakenteen omaavaan ferrosähköiseen lyijy-zirkonium-titanaattiin (PZT), toinen harvinaisempi toteutustapa Strontium-bismutti-tantalaatti -kerrosrakenteeseen (SBT). PZT-toteutuksessa materiaalin kiderakenteen joutuminen sähkökenttään tuottaa pysyvän polarisaation jo 1,5 – 3V jännitteellä. Tähän vaadittava materiaalinpaksuus on vain 70-100 nm, mikä tekee siitä käyttökelpoisen haihtumattoman muistiteknologian matalajännitteisiin älykortti- ja RF-tunnistussovelluksiin sekä ylipäätään sulautettuihin kannettaviin sovelluksiin. PZT-toteutuksessa kiteinen rakenne muuttuu sähkökentän vaikutuksesta siten, että titaaniatomi voi siirtyä sähkökentän vaikutuksesta kahteen pysyvään paikkaan, joista toinen on kiderakenteessa happitason ylä- ja toinen alapuolella.

Pysyvä polarisaatio on näiden siirtymien lopputuloksena syntynyt residuaalinen polarisaatio, joka on luokkaa 10-30 uC/cm2. Ferrosähköinen muistisolu voidaan toteuttaa joko integroimalla materiaali eristemateriaalin ympäröimäksi ferrosähköiseksi kondensaattoriksi tai ferrosähköiseksi FET-transistoriksi, jossa muistisoluun yhdistyy muistin valintaelementti. CMOS-valmistusteknologiaan yhdistettynä planaarisen kondensaattoriarkkitehtuurin korvaaminen kolmiulotteisella ferrosähköisellä kondensaattorilla antaa mahdollisuuden kasvattaa jatkossa ferrosähköisen muistin pakkaustiheyttä huomattavasti. Ferrosähköisen materiaalin tuominen CMOS-valmistusprosessiin on kuitenkin haastavaa ja vaatii lisää tutkimustyötä. [37 s. 55 - 57]

FRAM-muistisolua voidaan tarkastella dipolikondensaattorina, jossa ohut PZT-kalvo on kahden elektrodin välissä. Kytkemällä kondensaattoriin jännite muuttuu kalvon kiderakenne pysyvästi, vaikka kondensaattorin elektrodien välinen sähkökenttä lakkaisikin vaikuttamasta. Sähkökentän vaikutuksesta PZT-kalvon kiderakenteessa tapahtuva anionien ja kationien siirtymä merkitsee, että kun sähkökenttä alkaa vaikuttaa kiderakenteeseen ja sen suunta on Kuvan 8 tilanteessa alhaalta ylös, liikkuvat positiivisesti varautuneet Ti- tai Zr-ionit sähkökentän suunnassa ja asettuvat stabiileihin paikkoihin happitason yläpuolelle. Sähkökentän vaikuttaessa ylhäältä alas, anionit siirtyvät happitason alapuolella olevaan pysyvään tilaan. Sähköinen dipoli syntyy, koska kumpikaan mainituista tiloista ei sijaitse yksikköhilan keskipisteessä. [37 s. 820 – 821]



Kuva 8. PZT-kiteen rakenne [23 s. 2]

Syntyneet uudet polarisaatiotilat vastaavat FRAM:iin tallennettua loogista tilaa ’1’ ja ’0’. Kiderakenteen muutokset ovat pysyviä, joten bitti säilyy muistissa vaikka sähkökentän vaikutus lakkaisi. Muistisolun lukeminen tapahtuu siten, että kiderakenteeseen kytketään sähkökenttä, joka emittoi kiderakenteesta riippuen pienen tai suuren indusoituneen varauksen. Tätä varausta verrataan tunnettuun referenssivaraukseen, minkä perusteella voidaan arvioida kiteen tila (joko ’1’ tai ’0’). Koska lukeminen saa aikaan kiteen uudelleen polarisoitumisen ja loogisen tilan muuttumisen, on se palautettava kirjoittamalla tila lukuoperaation jälkeen uudestaan. [23]

FRAM edustaa haihtumatonta muistiteknologiaa, joka käyttäytyy samantyyppisesti kuin SRAM. Tämä mahdollistaa sen käytön monenlaisissa sulautettujen järjestelmien sovelluksissa. [24 s. 1] FRAM:n kirjoitusnopeudet ovat SRAM:n luokkaa, tosin tämä on mahdollista huomattavasti pienemmällä tehonkulutuksella. Kirjoituskestävyydessä FRAM on ylivoimainen Flashiin verrattuna. Tässä kirjallisuuskatsauksessa esiteltävässä TI:n MSP430-sarjan suorittimissa FRAM:iin voidaan kirjoittaa 10^15 kertaa, kun flashmuisti kestää tyypillisesti 10^5 kirjoituskertaa [23 s. 5]. Haihtumattomana muistina FRAM korvaa flashteknologiaa, joka perustuu paljon tehoa kuluttavaan varauspumppuperiaatteeseen. FRAM:n tehonkulutus tallennettua bitti kohti on 250 kertaa pienempi kuin flashilla ja kirjoittaminen onnistuu matalammalla jännitteellä. FRAM:n kirjoitusjännite on 1,5 V luokkaa, kun flash-muisteissa kirjoittamiseen vaaditaan 10-14 V jännite. [25] FRAM mahdollistaa SRAM:ia muistuttavan hajasaantimuistin toteuttamisen, koska datan osoittaminen luku- ja kirjoitusoperaatioissa sana- tai tavutasolla ei edellytä muistin segmentointia. Tässä kirjallisuuskatsauksessa lähemmin käsiteltävässä Texas Instrumentsin mikro-ohjaimessa voidaan kirjoitus- ja lukuoperaatiot kohdistaa FRAM:iin mikro-ohjainpiirin tarjoamalla 8MHz:n maksiminopeudella jolloin tyypilliset kirjoitusnopeudet ovat 2 MBps luokkaa, kun flasheissa päästään 14 Mbps:iin. FRAM:in etuna SRAM:iin verrattuna on myös, ettei se vaadi muistin tyhjentämistä ennen kirjoittamista. [26]

Ferrosähköisen muistin polarisaatio-ominaisuudet ovat lämpötilariippuvia. Muistisolu depolarisoituu, eli datan kirjoittamisen yhteydessä tapahtunut polarisoituminen palautuu lähtötilaan, kun lämpötila lähestyy Curie-lämpötilaa. Täydellinen depolarisaatio on tapahtunut, kun lämpötila on noin 430 astetta (transition temperature), mikä on hyvä ottaa huomioon valmistusprosessissa, esim. juotettaessa piirejä. Depolarisaatio ei kuitenkaan ole pysyvä tila, vaan materiaalin kyky polarisoitua uudelleen palautuu lämpötilan laskiessa Curie-lämpötilan alapuolelle. Toinen seikka, joka tekee ferromuistista lämpötilariippuvia on, että sen pitkäkestoinen altistuminen korkeille lämpötiloille vahvistaa tallennetun loogisen tilan pysyvyyttä, mistä seuraa että vastakkaisen loogisen tilan tallentaminen muistiin on vaikeaa. Toisin kuin depolarisaatiossa tämä ominaisuus ei palaudu lämpötilan laskiessa, vaan siitä seuraa pysyvä polarisaatiotilan stabiloituminen. Stabiloituminen tapahtuu alhaisemmassa lämpötilassa kuin depolarisaatio ja sitä voidaan myös käyttää hyväksi, kun kirjoitettu tieto halutaan tallentaa pysyvästi. Näin FRAM voi toimia myös lukumuistina. [23 s. 3]

FRAM:in integroiminen mikro-ohjainpiirille on mahdollistanut kokonaan uuden sukupolven sulautettujen ratkaisujen kehittämisen. FRAM:n flashiin verrattuna matala tehonkulutus, suuret luku- ja kirjoitusnopeudet (Taulukko 2) [26 s. 575], muistin pysyvyys ja luotettavuus sekä mahdollisuus käyttää sitä niin ohjelmamuistina kuin datamuistina ovat luoneet edellytyksiä myös langattomien sovellusten toteuttamiselle entistä kustannustehokkaammin.

Taulukko 2. FRAM:n kirjoitusnopeus ja keskimääräinen tehonkulutus flashiin ja SRAM:iin verrattuna [26 s. 575]

	NOR Flash	FRAM	SRAM
Non-volatile (Retains data)	Yes	Yes	No
Write speed (13KB)	1s	10ms	<10ms
Average power (uA/MHz)	260	100	<60

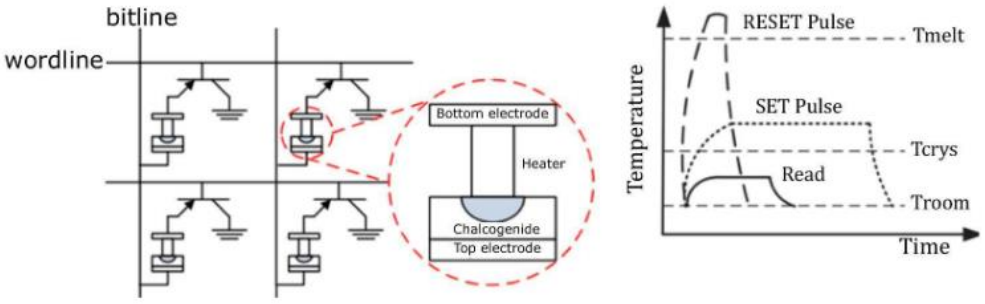
Langattomassa järjestelmässä FRAM-muisti mahdollistaa muistin joustavan jakamisen järjestelmän käyttämien erilaisten tiedonsiirtoprotokollien tarpeisiin. Perinteisesti flashilla tai RAM:lla toteutettujen suoritin- tai ohjainpiirille integroitujen erillisten ohjelma- ja datamuistien käyttö saattaa pakottaa valitsemaan tarkoitukseen huonosti soveltuvan ja kalliin suorittimen. FRAM mahdollistaa paremmin myös järjestelmien tietoturvan huomioimisen, koska firmwaren ohjelmistopäivitykset ja salasanasuojaukset on helpompi toteuttaa pysyvään, koko tuotteen elinkaaren kestävään muistiin pohjautuvassa järjestelmässä. FRAM:n matala tehonkulutus mahdollistaa järjestelmän koteloimisen ilman, että paristonvaihtomahdollisuutta tarvitsee ottaa huomioon. Tämä tekee siitä kompaktimman ja myös helpottaa erilaisten käyttöolosuhteiden huomioonottamista. [26]

Kuten aiemmin on todettu, FRAM:ssa tyypillinen kirjoituskertojen lukumäärä on 10^15 kun flashin kirjoituskertojen määrä on vain 10^5 luokkaa. Koska FRAM:ssa jokaiseen lukuoperaatioon kuitenkin liittyy myös uudelleenkirjoitus, lukuja ei voi suoraan verrata toisiinsa. Arvioitaessa FRAM:n elinkaarta, voidaan arvioida huonoimman skenaarion pohjalta (peräkkäisiä FRAM:iin viittaavia JMP-käskyjä ilman välissä suoritettavia muita käskyjä), että jokaiseen yksittäiseen muistipaikkaan kohdistuvien mahdollisten kirjoituskertojen lukumäärä merkitsee vähintään 72 vuoden minimikestävyyttä, joka on huomattavasti pidempi aika kuin tuotteen elinkaari. [23] Tilapäisen virheen esiintymistiheyttä voidaan arvioida joko käyttämällä käsitettä MTBF (Mean time between failures) tai FIT (Failures in time). SER (Soft error rate) tarkoittaa laitteessa tai järjestelmässä esiintyvää tai ennustettua satunnaisten virheiden lukumäärää suhteessa aikayksikköön. Haihtumattomissa muisteissa satunnaiset virheet ovat tyypillisesti neutronien aiheuttamia alfapartikkelien ja gammasäteiden sivuvaikutuksia. Koska FRAM:ssa tiedon säilyminen perustuu varauksen säilymisen sijasta kiteen polarisoitumiseen ei siinä tapahdu neutronien vaikutuksesta samanlaista tiedon häviämistä kuin SRAM:ssa. Tämä on todettu säteilytestauksissa, joiden perusteella FRAM:n SER on kertaluokkaa alempi kuin SRAM:ssa. [23 s. 7]

3.2.3 PRAM

Uudemmista muistityypeistä tässä kirjallisuuskatsauksessa esitellään FRAM:n lisäksi myös faasimuutosmuisti (PCM, PRAM). Sille on ominaista pieni fyysinen koko, hyvät satunnaishakuominaisuudet, kohtalainen kirjoitustehokkuus, pysyvyys ja NAND-flashia parempi kestävyys. Toisin kuin kirjoitettaessa flashiin, PRAM:ssa muistia ei tarvitse pyyhkiä ennen kirjoittamista. [2 s. 174]

PRAM perustuu germaniumantimonitelluridin faasimuutokseen kiteisen ja amorfisen olomuodon välillä. Tämä muutos saadaan aikaiseksi lämmittämällä ainetta kahden elektrodin välissä (Kuva 9) [2 s. 174].



Kuva 9. PRAM-muistisolun rakenne.

Aine vaihtaa olomuotoa nopeasti elektrodeihin kytketyn sähköisen pulssin lämmittävästä vaikutuksesta. Muutos ilmenee aineessa tapahtuvana resistanssin muuttumisena. Lyhyt pulssi tuottaa kohtalaisen suurella jännitteellä amorfisen tilan lämpötilan ylittäessä sulamispisteen (Tmelt), mikä vastaa loogista tilaa ’0’ (RESET). Vastaavasti matalajännitteinen pitkä pulssi tuottaa kiteisen tilan lämpötilan laskiessa kiteytymislämpötilan (Tcrys) alapuolelle, mikä vastaa loogista tilaa ’1’ (SET). Muistisolun kirjoitusnopeus määräytyy pidemmän SET-pulssin mukaan. Lukeminen perustuu siihen, että SET- ja RESET-tilojen välillä ilmenee luokkaa 10^2 – 10^4 oleva ero resistansseissa. PRAM-muistiin liittyy haasteita faasimuutosaineen ja elektrodin rajapinnassa esiintyvän lämpölaajenemisen ja supistumisen takia. Tämä rajoittaa PRAM:n kirjoituskestävyyden suuruusluokkaan 10^8, mikä on flashia parempi mutta DRAM:ia (10^15) huonompi. PRAM:in pitkä kirjoitusviive estää sen

käytön välimuistina nopeaa kirjoittamista vaativissa sovelluksissa. Sen sijaan sitä voidaan hyvin käyttää DRAM:ia korvaavana primääri- tai sekundäärimuistina. [2 s. 174 - 175] PRAM:in luokkaa 300 ns oleva kirjoitusviive johtuu faasimuutosaineen kohtalaisen hitaasta kiteytymisestä sekä aineen herkkyydestä häiritseville lämpötilavaikutuksille [27 s. 12].

4. SULAUTETTU MUISTI

Eri muistityyppien integroiminen suorittimelle tai mikro-ohjaimelle pakottaa ottamaan huomioon suoritinpiirin valmistukseen liittyvät valmistustekniset näkökohdat sekä mm. yhteensovittamisen suorittimen logiikkatasojen kanssa. Muistin koko ei valmistusprosessien kehittymisestä huolimatta yllä vielä saman muistityypin ulkoisen muistin tasolle. Tästä johtuen paljon muistikapasiteettia edellyttävissä sovelluksissa joudutaan turvautumaan ulkoisiin ja väyläviiveistä johtuen hitaampiin muisteihin. Sulautettujen sovellusten osalta tärkeät ominaisuudet, kuten energiankulutus, rakenteen kompaktius, päivitettävyys ja nopeus ovat sulautetun muistin vahvuuksia.

Ilman käyttöjärjestelmää toimiva sulautettu järjestelmä on perinteisesti suunniteltu toteuttamaan yhtä tai muutamaa tehtävää. Nykyään sulautettu järjestelmä kuitenkin on yhä useammin monta suoritinta käsittävä, paljon laskentatehoa sisältävä, useita tehtäviä samanaikaisesti suorittava tietokonejärjestelmä, joka perustuu reaaliaikaiseen käyttöjärjestelmään (RTOS). Siihen liittyy myös vaatimuksia koskien järjestelmän päivitettävyyttä verkon yli, ohjelmoitavuutta sekä tietoturvaratkaisuja. Tämä kaikki edellyttää ennen kaikkea suurempaa haihtumattoman muistin määrää. Mikro-ohjainta valittaessa yhdistyy käyttötarkoitusta ajatellen optimaalinen suoritin harvoin optimaaliseen muistin kokoon, joten usein täytyy tehdä kompromisseja ja hyväksyä jommankumman yliresursointi. Valintaa ohjaa yleensä muistin koko ja suorittimeksi valikoituu käyttötarkoitukseen nähden liian tehokas suoritin.

4.1 Sulautetun muistin erityispiirteitä

Sulautettu muisti koostuu haihtuvasta ja haihtumattomasta muistista. Haihtumattomaan muistiin tallennetaan sovellusohjelmat, käynnistystiedostot, laitekonfiguraatio, käskyjen tarvitsemat muuttumattomat parametrit sekä mahdollinen käyttöjärjestelmä. Haihtuva muisti toimii työmuistina, johon järjestelmän tuottama data tallennetaan. Teknisen kehityksen nykyisessä vaiheessa sulautetun muistin toteutukset perustuvat pääasiassa flashin käyttöön haihtumattomana ja SRAM:n käyttöön haihtuvana muistina [28]. SRAM:ia käytetään

myös suoritinta lähellä olevana nopeana muistina, kuten tason 1 välimuistina. Vaikka SRAM on ylivoimaisesti yleisin haihtuva sulautettu muistityyppi, on myös DRAM:ia integroitu menestyksekkäästi suorittimeen. Uusiin 40 nm perustuviin mikro-ohjaimiin on kyetty integroimaan jo 8 MB flashiä ja jopa 3,5 MB RAM:ia. Sulautetun DRAM:n tehonkulutus on luokkaa 1/5-1/8 sulautetun SRAM:in tehonkulutukseen verrattuna. Suoritinpiirille integroitu DRAM sietää myös sulautettua SRAM:ia paremmin ympäristön taholta tulevia, muistin loogisia tiloja häiritseviä säteilyvaikutuksia (SER, Soft error rate). [28] Kuuteen transistoriin (6T) perustuva SRAM-muisti yhdessä vertikaaliseen kondensaattoriin pohjautuvan DRAM:in kanssa mahdollistavat 256 MB luokkaa olevat haihtuvat sulautetut muistit, joita voidaan käyttää nopeana välimuistina. [27]

Flashin asema sulautettuna käskymuistina on vahva. Suoritinsirulle integroituna sillä saadaan toteutettua 32 MB muisti, jonka käyttöjännite on 1,3V ja jonka kellotaajuus yltää 200MHz:een. Sulautettu flashmuisti pienentää järjestelmän suunnitteluun kuuluvaa aikaa ja kustannuksia, koska ohjelmointi on helppoa senkin jälkeen, kun mikro-ohjain on jo asennettu laitteeseen. Uusista muistiteknologioista sulautetulla PRAM:lla on kyetty saavuttamaan 4 MB muistikapasiteetti 1,2 V käyttöjännitteellä, 12 ns hakuaika ja 10^6 jakson kirjoituskestävyys. Monissa sulautetuissa sovelluksissa jo laajasti käyttöön otettu sulautettu FRAM kykenee 4 MB tallennuskapasiteettiin 1,8 V käyttöjännitteellä. Vertailun vuoksi todettakoon, että ulkoisena muistina FRAM:lla päästään samalla käyttöjännitetasolla 128 MB tallennuskapasiteettiin. [27 s. 11 – 12]

Suoritinpiirille integroidun välimuistin energiankulutus on tyypillisesti 25-50% koko suoritinpiirin energiankulutuksesta. Samassa suhteessa kuluu sirupinta-alaa. Vaikka tällä ei tietokoneissa välttämättä ole kovin suurta merkitystä, niin sulautetuissa järjestelmissä sekä muistiin käytetty puolijohdepinta-ala että energiankulutus ovat keskeisiä tekijöitä. Tästä johtuen monet valmistajat ovat korvanneet sulautettujen järjestelmien suorittimissaan laiteohjatun välimuistin työmuistilla (Scratch pad memory, SPM). Se on tason 1 välimuistin tapaan lähinnä suoritinta sijaitseva, SRAM:lla toteutettu, nopea väliaikainen muisti laskutoimitusten ja tiedon tilapäiseen tallentamiseen. Se ei kuitenkaan välimuistin tapaan sisällä yleensä kopiota päämuistin tietosisällöstä. SPM on mukana mm. ARM10E, Analog Devices:n

ADSPTS201S, Motorolan M-core MMC221 ja TI:n TMX320C6xxx -suorittimissa. Laiteohjatun välimuistin korvaaminen SPM-työmuistilla parantaa sulautetulle muistille tärkeää energiatehokkuutta. Tutkimuskirjallisuudessa on raportoitu tutkimustuloksista, jotka viittaavat SPM-muistin vuotovirtojen hallintaan kehitetyn muistinhallinta-algoritmin sekä SRAM-NVM -hybriditoteutuksen avulla saavutettavista parannuksista. Kokeellisten tulosten perusteella näin saavutettiin 18,17% pienemmät hakuajat, 24,29% pienempi energiankulutus sekä 37,34% pienempi vuotovirtojen aiheuttama hukkateho kuin samankokoiselle puolijohdealalle toteutetulla SRAM:lla. [29 s. 1094 – 1101]

Sulautetuille järjestelmille on tyypillistä, että niiden muistitarve on dynaamista ja ennakoimatonta. Järjestelmän liittyminen langattomiin tiedonsiirtoverkkoihin tuottaa jatkuvasti muuttuvan määrän muistiviittauksia ja tallennustarpeita, joita ei voi ottaa tarkasti huomioon ennakolta. Lisäksi antureiden syöttämä reaaliaikaisesti muuttuva tieto yhdessä käyttäjän tai sovelluksen reaaliaikaisuuteen perustuvien toimenpiteiden kanssa edellyttävät dynaamista muistinhallintaa. [3 s. 9] Ohjelmamuistin osalta erityisesti lisääntynyt tarve ottaa huomioon käyttäjän mahdollisuuksia muokata sovelluksia käyttöympäristön ehdoilla, asettaa kasvavan muistitarpeen lisäksi vaatimuksia myös muistin ohjelmoitavuudelle. [28] Ohjelmointiin ja I/O-laitteisiin liittyviä näkökohtia tarkastellaan seuraavassa lyhyesti käyttämällä esimerkkinä kahta yleisesti käytettyä mikro-ohjainta.

Sulautetun muistin myötä järjestelmän nopeus, kaistanleveys ja luotettavuus paranevat samalla kun sen tehonkulutus, suunnittelu- ja valmistuskustannukset sekä fyysinen koko pienenevät. [27] Piirin ulkopuoliseen muistiin verrattuna sulautetun muistin nopeus on hintaa ja muistin kokoa tärkeämpi kriteeri. Vaikka sulautetun muistin alhainen tehonkulutus on monissa sovelluksissa ensiarvoisen tärkeää, on siihen pyrkiminen käyttöjännitettä alentamalla ongelmallista. Käyttöjännitteen laskeminen yhdessä suuremman pakkaustiheyden myötä pienentyneiden fyysisten mittojen kanssa johtaa suurempiin kynnysjännitteen vaihteluihin muistisolun transistoreissa. Käyttöjännitteen laskeminen 100 mV:lla merkitsee vuotovirtojen kasvua kertaluokkaa suuremmiksi, mikä johtaa tehonkulutuksen kasvuun. Näiden ongelmien ratkaisemiseksi on kehitetty adaptiivisia tehonsyöttöratkaisuja, kahteen

käyttöjännitteeseen tai kynnysjännitteeseen perustuvia piirejä sekä eri paksuisiin oksidikerroksiin perustuvia MOSFET-teknologioita. [27 s. 10]

4.2 Mikro-ohjainten sulautetut muistikonfiguraatiot

Oheiseen taulukkoon (Taulukko 3) on kerätty tietoja muutamissa eri suoritinarkkitehtuureja edustavissa kaupallisissa suorittimissa käytetyistä muistityypeistä. Käskyjen ja vakioiden tallentamiseen käytetään sekä pienemmän suorituskyvyn 8-bittisissä mikro-ohjaimissa että raskaampaa laskentatehoa vaativissa teollisuuden sulautetuissa DSP-suorittimissa pääasiassa flashmuistia. Haihtuvana muistina käytetään yleisimmin nopeaa SRAM:ia. Joissain mikro-ohjaimissa SRAM on paristovarmennettu, kuten ST-Microelectronicsin ARM-suorittimeen perustuvassa STM32:ssa. Uusista muistiratkaisuista FRAM on mukana ainakin Texas Instrumentsin ja Renesasin mikro-ohjaimissa. Microchipin DSP-suorittimen mikro-ohjaimeen yhdistävässä dsPIC33 -ohjaintoteutuksessa käytetään haihtumattomana muistina 24 kB:n PRAM:ia. Flashin pitkälti korvaama EEPROM on mukana flashin rinnalla vielä joidenkin valmistajien mikro-ohjaimissa, joista mainittakoon taulukossa esiintyvät Microchipin PIC24F32 sekä Silicon Labsin C8051. [30], [31], [32], [33], [34], [35]

Taulukko 3. Sulautetun muistin määrä ja muistityypit eräissä mikro-ohjaimissa (*1 paristovarmennettu SRAM, *2 32 kB käskyille ja 4 kB datalle)

Mikro-ohjainsarja / piiri	**Muisti** (kB)				**Ydin**
	Haihtumaton		***Haihtuva***		
Atmel ATSAM3U2	Flash	128	SRAM	36	ARM Cortex M3
Cypress S6J3200	Flash	4000	SRAM	512	ARM Cortex R5
Intel Quark D2000	Flash OTP flash	32 + 4 *2 8	SRAM	8	Intel Quark
Microchip PIC24F32	Flash EEPROM	32 0,512	SRAM	2	PIC24
Microchip dsPIC33	PRAM	24	SRAM	16 + 4	dsPIC
TI MSP430FR	FRAM	256	SRAM	8	MSP430
Freescale Kinetis L	Flash	256	SRAM	32	ARM Cortex
Rabbit 6000	SRAM *1	32	SRAM	1000	Rabbit 6000
STM32 F2	Flash SRAM *1 OTP	1000 4 0,528	SRAM	0,128	ARM Cortex
Silicon Labs C8051F37	EEPROM Flash	0,512 16	SRAM	1	8051

Kaikissa kaupallisissa mikro-ohjaimissa käytetään useampaa kuin yhtä muistityyppiä. Useampia apusuorittimia käsittävissä mikro-ohjaimissa voi kullakin suorittimella olla oma käskymuisti suorittimen useimmin tarvitsemia operaatioita varten. Tällöin se on toteutettu yleensä SRAM:lla ja muita käskyjä sekää dataa varten on varattu yhteinen jaettu muisti, joka on toteutettu tavallisesti flashilla. Sulautetuissa järjestelmissä tarvitaan monesti useita tiedonsiirtoprotokollia, mikä kasvattaa SRAM:in tarvetta useiden pinomuistien ja verkkojen välisten tiedonsiirtorutiinien takia. Siirryttäessä entistä kehittyneempiin piirivalmistusprosesseihin kasvavat myös suoritinpiirien nopeudet ja tällöin tarvitaan yhä useammin myös tason 2 välimuistia, mikä sekin edellyttää suurempaa SRAM-muistin määrää. [28]

Seuraavassa tarkastellaan lyhyesti kahta mikro-ohjainta, jotka toimivat esimerkkeinä erilaisia muistikonfiguraatioita edustavista sulautetun muistin toteutuksista. Valituista esimerkeistä toinen edustaa von Neumann -arkkitehtuuria, toinen modifioitua Harvard-arkkitehtuuria. Vaikka valitut tuotesarjat ovat kummankin valmistajan tapauksessa periaatteessa keskenään hyvin samankaltaisia, esiintyy

tuoteperheiden sisällä eroja liittyen muistityyppeihin ja muistin määriin. Tästä johtuen tarkasteluun on valittu kummassakin tapauksessa yksi mikro-ohjain, josta voidaan esittää eksaktimpi kuvaus. Kuvauksessa on kiinnitetty huomiota muistitoteutuksen ohella myös ohjelmointiin ja I/O-rajapintaan sekä energiankulutukseen. Ensimmäisenä luodaan lyhyt katsaus SRAM- ja FRAM-muisteja sisältävään Texas Instrumentsin MSP 430-sarjan mikro-ohjaimeen MSP 430 FR5994.

4.2.1 Sulautettu FRAM-SRAM -muistikonfiguraatio: TI:n MSP 430 -sarja

Texas Instrumentsin MSP430-sarja on analogisiin ja digitaalisiin mittaussovelluksiin kehitetty mikro-ohjainperhe, jonka suoritin perustuu von Neuman -arkkitehtuuria soveltavaan 16-bittiseen RISC-arkkitehtuuriin. Se soveltuu matalan tehonkulutuksensa ansiosta erinomaisesti käytettäväksi sulautetuissa järjestelmissä. MSP430FR -sarja sisältää ferromagneettista FRAM-muistia mallista riippuen 128-258 kB sekä 2-8kB SRAM:ia. [33]

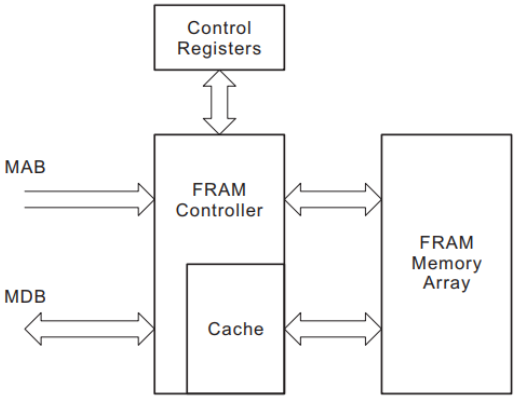
Kuten aiemmin on todettu FRAM voisi toimia yhdistettynä käsky- ja datamuistina, jolloin mikro-ohjaimessa ei tarvittaisi välttämättä muita muistityyppejä. Sen rinnalla on kuitenkin järkevää käyttää energiatehokkaampaa SRAM:ia, jossa lukujakson viiveet eivät muodostu yhtä suuriksi. FRAM-SRAM -hybriditoteutuksen energiankulutusta voidaan edelleen pienentää järkevällä muistinhallinnalla, jossa ohjelman toiminnalliset lohkot sijoitetaan muistiin mahdollisimman energiatehokkaasti. Tutkimuskirjallisuudessa on raportoitu samaan tuoteperheeseen kuuluvan TI:n MSP430FR5739 mikro-ohjaimen suorituskyvyn kaksinkertaistumisesta ja energiankulutuksen laskusta 20% pelkästään FRAM:lla toteutettuun ratkaisuun verrattuna. [36 s. 264 – 266]

MSP430FR5994 sisältää 256 kB FRAM-muistia ja 8 kB SRAM:ia. Suorittimen sisältämä kohtalaisen suuri FRAM mahdollistaa kehittyneiden sovellusten tallentamisen piirille sekä helpottaa langattomien ohjelmapäivitysten tekoa. MSP430FR5994:n SRAM on jaettu kolmeen sektoriin, joista kaksi ensimmäistä ovat

2kB:n kokoisia ja kolmas, 4kB:n sektori on jaettu LEA-moduulin (Low-energy accelerator) kanssa. [33 s. 66 - 73]

MSP430FR5994:ssa on 4 tehonsäästötilaa (LPM0 - LPM3), joista mikro-ohjain saadaan heräämään keskeytyspyynnöillä. Laitteen suoritettua vaaditut rutiinit se palaa takaisin tehonsäästötilaan. Erityisesti mittaus- ja DSP-sovelluksissa käyttökelpoinen LEA-energiansäästötoiminto mahdollistaa suurta laskentatehoa vaativien FFT-muunnosten, digitaalisen suodatuksen ja matriisien kertolaskujen toteuttamisen pienellä tehonkulutuksella. LEA perustuu matalatehoiseen apusuorittimeen, joka hoitaa paljon laskenta-aikaa vaativien funktioiden laskennan häiritsemättä pääsuorittimen toimintaa. Kun laskenta on valmis, se lähettää keskeytyspyynnön pääsuorittimelle. LEA:ssa on tehokkuuden ja käytettävyyden parantamiseksi erillinen DSP-kirjasto, jossa tulo- ja lähtöpuskureihin osoittavia käskyjä käytetään. DSP-kirjasto sisältää funktioita, jotka on optimoitu suorittamaan useita tavallisimpia digitaalisen signaalinkäsittelyn operaatioita. Käyttäjäystävällisyyttä on pyritty ottamaan huomioon hyödyntämällä makroja, jotka automaattisesti valitsevat ja ottavat käyttöön funktioihin tarvittavat LEA-moduulit. LEA mahdollistaa myös omien filttereiden generoimisen Pythonin tai Matlabin avulla. [34]

FRAM:iin kohdistuvat operaatiot ovat aina myös kirjoitusoperaatioita, koska lukemisen yhteydessä tieto on kirjoitettava uudelleen, kuten aiemmin on todettu. MSP430FR5994 käyttää neljän sanan välimuistia (64 bittiä) käskyjen puskuroimiseen ennen niiden suorittamista (Kuva 10). [38 s. 301]



Kuva 10. FRAM-muistin ohjaus.

Käskyhaut kohdistuvat FRAM:iin vasta kun kaikki käskyt on noudettu välimuistista. Käskyjen lukeminen välimuistiin tapahtuu yhden FRAM:iin kohdistuvan lukuoperaation aikana, mikä tarkoittaa, että suoritettavia käskyjä luetaan FRAM:sta joka neljännen sanan mittaisen käskyn hakemisen yhteydessä. Tämä voidaan ohjelmassa muuttaa haluttaessa tapahtuvaksi joka käskyn suorittamisen yhteydessä. JMP-käskyjen yhteydessä voidaan kuitenkin käyttää myös nop-käskyjä (no operation) kun seuraavan haarautumiskäskyn yhteydessä halutaan viitata välimuistin ulkopuolelle (FRAM:iin). [23 s. 6] FRAM-ohjaimen sisältämä välimuisti on toteutettu kaksisuuntaisena assosiatiivisena välimuistina, jossa on 4 kpl 64-bittisiä rivejä. Kun käskyjen osoitteet osuvat samalle välimuistin riville, voidaan käskyt suorittaa ilman keskeytystiloja. [38 s. 302 - 304]

Kaikki käskyt on MSP 430FR5994:ssa haarautumiskäskyjä lukuun ottamatta toteutettu rekisterikäskyinä, joissa lähdeoperandiin voidaan viitata seitsemällä eri osoitustavalla ja kohdeoperandiin neljällä. Suorittimeen on integroitu 16 rekisteriä, joihin kohdistuvat rekisteristä-rekisteriin -operaatiot suoritin kykenee suorittamaan yhden kellojakson aikana. Neljää ensimmäistä rekisteriä (ohjelmalaskuri, pino-osoitin, tilarekisteri ja vakiogeneraattori) lukuun ottamatta kaikki rekisterit ovat yleiskäyttöisiä. [38 s. 65]

FRAM:n ohjelmointi on mahdollista JTAG:n, BSL:n, Spy-Bi-Ware -rajapinnan kautta tai ROM:ssa olevan käyttäjän luoman sovelluksen avulla. FRAM:n ohjelmointi JTAG-rajapinnan kautta tapahtuu joko 4-signaalisen JTAG:in kautta tai kahta pinniä käyttävän Spy-Bi-Ware -rajapinnan kautta. Ohjelmointi ja debuggaus perustuvat makroihin joilla JTAG-rajapinnan tiedonsiirtoa ohjataan operoimalla

JTAG:in käsky-, data- ja osoiterekistereitä. Ohjelmointi voidaan suorittaa tietokoneeseen kytkettävän kehitysalustan avulla käyttämällä Texas Instrumentsin Code Composer Studiota. Spy-Bi-Ware -rajapinnan kautta tapahtuva ohjelmointi on periaatteessa kahta signaalia käyttävä JTAG-rajapinta, joka sisältää 4-signaalisen JTAG:in kaksitie-signaaliksi muuttavan logiikan. Käynnistyksenlataaja (BSL) mahdollistaa FRAM:n ja SRAM:in lukemisen ja ohjelmoimisen UART -sarjaväylän kautta. Pääsy voidaan suojata 256-bittisellä käyttäjän määrittelemällä salasanasuojauksella. Kolmas ohjelmointivaihtoehto perustuu käyttäjän luomaan sovellukseen, jonka avulla FRAM:ia voidaan ohjelmoida käyttäen UART:ia, SPI:tä tai mitä tahansa tiedonsiirtotapaa. Kaikki ohjelmointitavat mahdollistavat in-system -ohjelmoinnin. [38 s. 302], [41], ks. myös [42]

MSP430FR5994:n 80-pinnisessä versiossa on 9 I/O-porttia, joissa on yhteensä 68 kaksisuuntaista vapaasti konfiguroitavaa I/O-linjaa, joita jokaista voidaan lukea ja kirjoittaa itsenäisesti. Kaikkien yksittäisten I/O-bittien ohjelmoiminen on mahdollista jokaisessa I/O-linjassa. Ylös- tai alasvetovastukset voidaan kussakin I/O-linjassa kytkeä itsenäisesti päälle tai pois päältä. I/O-porttien P1 – P8 kaikille linjoille on mahdollista asettaa sisääntulosignaalin reunalla lähetettävä keskeytyspyyntö ja porttien P1 ja P2 linjat voidaan ohjelmoida lähettämään keskeytyspyyntö joko signaalin nousevalla tai laskevalla reunalla. Kaikki I/O-linjat voidaan myös herättää tehonsäästötiloista LPM3.5 ja LPM4.5. [35 s. 364], [33 s. 74]

Yksittäiseen I/O-porttiin kohdistuvat luku- ja kirjoitusoperaatiot voidaan suorittaa tavun tai sanan levyisinä. Jälkimmäisessä tapauksessa porteista muodostetaan pareja, jotka nimetään PA (portit P1 ja P2), PB (portit P3 ja P4) jne. Samalla tavalla myös operaatioihin liittyvät rekisterit nimetään, lukuun ottamatta keskeytysvektorirekisterejä, joita kutsutaan kuten käytettäessä yksittäisiä portteja (P1IV, P2IV jne.). Kirjoitettaessa porttiin PA, kaikki 16 bittiä kirjoitetaan. Porttiin voidaan kirjoittaa myös tavun levyisiä käskyjä, jolloin portissa valmiina oleva sana korvautuu vain kirjoitettavan tavun leveydeltä ja loppuosa säilyy ennallaan. Sama periaate on voimassa myös lukuoperaatioiden osalta. Tosin on muistettava, että kirjoitettaessa kohderekisteriin tavua I/O-portista, se tallentuu vähiten merkitsevän tavun paikalle ja eniten merkitsevä tavu pyyhkiytyy pois. Käyttäjän toimesta

konfiguroitavia I/O-portteja ohjataan input-, output- ja suuntarekistereillä (PxIN, PxOUT, PxDIR). [38 s. 312 - 313]

Seuraavassa tarkastellaan Atmelin ATmega-sarjan mikro-ohjainta esimerkkinä kolmesta muistityypistä koostuvasta sulautetusta muistista.

4.2.2 Sulautettu flash-SRAM-EEPROM -muistikonfiguraatio: Atmelin ATmega -sarja

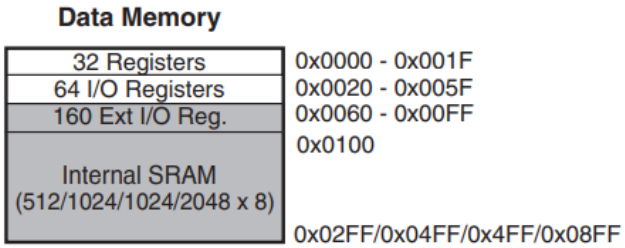
Atmelin AVR-arkkitehtuuriin kuuluvat ATmega -mikro-ohjaimet perustuvat 8-bittiseen modifioituun Harvard-arkkitehtuuriin. RISC-prosessorin ympärille rakennettuna Atmega-mikro-ohjaimet suorittavat useimmat käskyt yhden kellojakson aikana ja kaikkia yleiskäyttöisiä rekistereitä voidaan käyttää sekä käskyjen lähde- että kohderekistereinä. ATmega -mikro-ohjaimissa on ohjelmamuistina 32kB flashmuisti, johon voidaan kirjoittaa suorittimen ollessa kytkettynä järjestelmään (ISP, In-system programming). Lisäksi piirille on integroitu 1kB EEPROM- sekä 2kB SRAM-muistit. Suoritin sisältää 23 yleiskäyttöistä I/O-linjaa sekä 32 yleiskäyttöistä 8-bittistä rekisteriä, joita voidaan käyttää niin datan lähde- ja kohderekistereinä kuin data- tai osoiterekistereinä. Atmega-mikro-ohjain on mahdollista ohjelmoida myös ilman ohjelmointilaitetta. Erityisesti AVR-suorittimiin liittyvä suorittimen ja käskymuistin välinen yhteys mahdollistaa käskymuistiin kohdistuvien hakujen käsittelyn lukumuistityyppisenä datana. Tämä merkitsee, että merkkijonojen ja funktiotaulukoiden tyyppiseen vakiomuotoiseen dataan kohdistuvat haut ovat mahdollisia ilman, että data täytyy ensin tallentaa datamuistiin. Tämä vapauttaa niukkaa ja tehoa kuluttavaa datamuistia luku- ja kirjoitusoperaatioiden muuttujia varten. [7], [39]

Atmega -sarjaan kuuluvassa Atmega328-mikro-ohjaimessa on kolme lineaarista muistiavaruutta, jotka sijaitsevat flashissa, SRAM:ssa ja EEPROM:ssa. Mikro-ohjainpiirille integroitua 2k x 16 muotoon organisoitua flashia käytetään ISP-ohjelmoitavana ohjelmamuistina. Se on jaettu kahtia käynnistyksenlataus- ja sovellusohjelmaosioihin. Ulkoista ohjelmamuistia ei voi käyttää, vaan kun flashin käynnistysosioon on ladattu latausohjelma (Boot loader), piiri voidaan ohjelmoida sen avulla. Flashin käynnistysosioon kuuluva käynnistyslatausohjelma sisältää

’Read-While-Write Self-Programming’ -mekanismin, joka mahdollistaa ohjelmakoodin tai päivityksen lataamisen minkä tahansa rajapinnan kautta. Ohjelmamuistina toimivan flashin päivittäminen tapahtuu sivu kerrallaan, mitä ennen sivu on tallennettava tilapäiseen puskurimuistiin ja korvattava sivu on tyhjennettävä. Mikro-ohjaimen sisältämistä 32 yleiskäyttöisestä rekisteristä yhtä 16-bittistä rekisteriä voidaan käyttää flashiin viittaavana osoiteosoittimena. Useimmat käskyt noudattavat yhden 16-bittisen sanan formaattia ja jokaiseen ohjelmamuistin osoitteeseen liittyy 16- tai 32-bittinen käsky. [6]

Ennen kuin muistisivu voidaan kirjoittaa flashiin, on keskeytystaulukko sekä ohjelmarutiini siirrettävä käynnistyksenlataus -osioon. Siirrettävä sivu osoitetaan RAM:ssa Y-osoittimen avulla ja flashin ensimmäiseen kirjoitettavaan muistipaikkaan osoitetaan z-osoittimella. Ohjelmalaskurin sisältämät paluuosoitteet tallennetaan pinomuistiin, joka on SRAM:ssa ja jolle asettaa ylärajan vain SRAM:n koko. Kaikkiin käyttäjän kirjoittamiin ohjelmiin tulee sisältyä pino-osoittimen alustaminen, jotta luku- ja kirjoitusoperaatioita voidaan kohdistaa I/O-avaruuteen. [6]

Atmega328:n SRAM sisältää 2048x8-bittisen datamuistin, 32 yleiskäyttöistä rekisterimuistipaikkaa (r0-r31), 64 muistipaikkaa I/O-käskyjä varten sekä 160 rekisterin laajennetun I/O-muistin (Kuva 11). Ohjelmoitavat I/O-rekisterit sisältävät sekä ohjelmoitavien laitteiden että suorittimen ohjaus- ja tilarekisterit. [6]



Kuva 11. ATmega328:n Datamuistikartta SRAM:ssa.

Datamuistina käytettävässä SRAM:ssa on käytössä viisi erilaista osoitustapaa: suora osoitus, epäsuora osoitus, epäsuora osoitus siirroksella, epäsuora osoitus esidekrementoinnilla ja jälki-inkrementoinnilla. Keskeytykset on toteutettu I/O-avaruudessa olevien ohjausrekisterien avulla. Kaikille keskeytyksille on oma keskeytysvektori keskeytysvektoritaulukossa, jonka mukaan niiden prioriteetti määräytyy. I/O-avaruus käsittää 64 osoitetta oheislaitteisiin liittyviin

ohjausrekistereihin, SPI:iin ja I/O-toimintoihin viittaamiseksi. I/O-muistiin voidaan viitata suoraan tai datamuistin rekisteritiedostojen 0x20 – 0x5F kautta. Atmega328 sisältää myös laajennetun I/O-avaruuden 0x60 – 0xFF SRAM:ssa. Kaikki suorittimen 32 yleiskäyttöistä rekisteriä on yhdistetty myös datamuistiosoitteisiin, jotka viittaavat 32 ensimmäiseen muistipaikkaan datamuistiavaruudessa. [6 s. 28]

EEPROM-muisti kestää luku- ja pyyhkimisoperaatioita kymmenkertaisen määrän flashiin verrattuna. Se on ATmega328:ssa toteutettu erillisenä datamuistialueena, jota operoidaan EEPROMIN osoite-, data- ja ohjausrekisterien avulla. Toisin kuin flashiin, EEPROM:iin voidaan kohdistaa luku- ja kirjoitusoperaatioita tavuittain. Ennen kuin flashiin kirjoitetaan, on sivu ensin pyyhittävä. ATmega328:ssa samalla pyyhkiytyy myös EEPROM, mikä täytyy huomioida asettamalla lukkobitti (EESAVE), ellei EEPROM:in sisältöä haluta menettää. Samoin kuin piirilevytoteutuksessa, myös suoritinpiirille integroitu EEPROM on herkkä käyttöjännitteen vaihteluille. Käyttöjännitteen laskun aiheuttama, kirjoittamisen aikana tapahtuva korruptoituminen voidaan kuitenkin estää pitämällä AVR Reset -pinniä alhaalla kirjoittamisen ajan. [6 s. 30]

SPI-tiedonsiirto otetaan käyttöön Atmega-mikro-ohjaimissa asettamalla PRSPI-bitti nollaksi. Isäntä- ja orjalaitteen kommunikointi tapahtuu kahden 8-bittisen siirtorekisterin välillä isäntälaitteen kellosignaalin (SCK) tahdistamana. Isäntälaite alustaa liikenteen vetämällä halutun orjalaitteen valintasignaalin (SS) alas. Isäntä- ja orjalaite valmistelevat lähetettävän datan siirtorekistereihin ja isäntälaite generoi kellosignaalin jolloin tiedonsiirto alkaa isäntälaitteelta MOSI-linjaa pitkin ja orjalaitteelta MISO-linjaa pitkin. Jokaisen datapaketin jälkeen isäntälaite synkronoi orjan lähettämällä valintasignaalin ’1’. [6 s. 282]

5. POHDINTAA

Pohdittaessa sulautettuun muistiin liittyviä keskeisiä vaatimuksia nousevat alhainen energiankulutus ja luotettavuus tärkeiksi tekijöiksi. IoT-sovellusten lisääntyminen sekä energiankeruuteknologioiden kehittyminen luovat osaltaan kysyntää uusille muistityypeille mutta myös erilaisille muistioperaatioiden optimointiin tähtääville energiatehokkaille muistinhallintaratkaisuille. Sulautettujen suorittimien suunnittelijat ovat vuosien ajan tehneet töitä kehittääkseen suorituskykyisen satunnaishakumuistin, joka sopisi yhteiseksi muistityypiksi sekä käskyille että datalle ja jossa haihtumattomuus yhdistyisi matalaan energiankulutukseen. [27 s. 11] Tässä kirjallisuuskatsauksessa mukana ollut FRAM edustaa luotettavuutta haihtumattomana muistityyppinä, minkä merkitys korostuu erityisesti käyttöjännitevaihteluille alttiissa energiankeruusovelluksissa mutta tietoturvaan liittyvistä syistä johtuen myös muissa sulautetuissa sovelluksissa. FRAM:n käyttö yhdistettynä sulautettuna muistina energiankeruusovelluksissa voisikin olla hedelmällinen jatkotutkimuksen kohde.

Vaikka tässä kirjallisuuskatsauksessa pyrittiin muodostamaan mahdollisimman kattava kuva sulautetuissa suorittimissa käytetyistä muistityypeistä, valikoitui mukaan suhteellisen pieni joukko erilaisia muistityyppejä sisältäviä suorittimia. Flashin ja SRAM:in vahva asema sulautettuna muistina selittää tätä osaltaan. Eri muistityyppien vertailu on ongelmallista erityisesti valmistajien antamien viiveisiin ja energiankulutukseen liittyvien tietojen keskinäiseen vertailtavuuteen liittyvien ongelmien takia. Tämän kirjallisuuskatsauksen puitteissa ongelma nousi esille erityisesti flashien osalta [19].

6. YHTEENVETO

Vaikka nykyisissä sulautetuissa suorittimissa ja mikro-ohjaimissa yleisin muistikonfiguraatio perustuu flashiin ja SRAM:iin, käytetään sulautettuna muistina edelleen myös EEPROM:ia ja DRAM:ia. Flash on kuitenkin käytännössä syrjäyttänyt EEPROM:in haihtumattomana muistina. Tähän ovat vaikuttaneet mm. järjestelmien päivitettävyysvaatimukset, sekä EEPROM:in flashia huomattavasti pidemmät kirjoitusviiveet. SRAM:n nopeus ja matalampi energiankulutus tekee siitä DRAM:ia paremmin sulautetuksi muistiksi soveltuvan. Koska sulautetulle muistille alhainen energiankulutus ja nopeus ovat suurta tallennuskapasiteettia tärkeämpiä ominaisuuksia, ei SRAM:ia vähemmän puolijohdepinta-alaa vievän DRAM-muistisolun pienempi koko ole merkityksellinen tekijä.

Tässä kirjallisuuskatsauksessa tutustutaan myös kahteen uutta haihtumatonta muistiteknologiaa edustavaan muistityyppiin, ferrosähköiseen FRAM:iin ja faasimuutokseen perustuvaan PRAM:iin, joista FRAM edustaa pidemmälle kaupallisettua teknologiaa. FRAM:in etuna flashiin verrattuna on huomattavasti parempi kirjoituskestävyys, suurempi kirjoitusnopeus sekä matalammasta kirjoitusjännitteestä johtuva pienempi tehonkulutus. FRAM:ia voidaan käyttää SRAM:in tyyppisenä hajasaantimuistina, koska se ei edellytä muistin segmentointia. Tästä johtuen sitä voidaan käyttää periaatteessa myös SRAM:ia korvaavana datamuistina. FRAM:in etuna SRAM:iin nähden on, että kirjoittaminen ei edellytä ensin tiedon pyyhkimistä. FRAM:in ominaisuuksista johtuen sitä voidaan käyttää yhdistettynä data- ja käskymuistina. PRAM asettuu kirjoituskestävyydessä flashin ja DRAM:n välille. Kirjoitusviiveiden takia PRAM soveltuu kuitenkin huonosti SRAM:ia korvaavaksi sulautetuksi muistiksi. Sen sijaan sen käyttö ulkopuolisena sekundäärimuistina on perusteltua.

Sulautetun muistin ominaisuudet eivät riipu pelkästään käytetyistä muistityypeistä. Integroitaessa muistia suoritinpiirille täytyy ottaa huomioon myös valmistusteknologiaan liittyvät näkökohdat. Vaikka valmistusprosessit ovat kehittyneet viime vuosina nopeasti, ei sulautetun muistin koko yllä ulkoisen muistin tasolle. 3D-valmistusteknologioiden kehittyminen on kuitenkin mahdollistanut myös muistikapasiteetiltaan suurempien muistien integroimisen suorittimelle.

Tässä kirjallisuuskatsauksessa tarkastelluista sulautettujen suorittimien ja mikro-ohjaimien muistityypeistä voidaan todeta, että kaikkien tarkasteltujen mikro-ohjaimien muistikonfiguraatiot koostuvat vähintään kahdesta muistityypistä. Tarkasteltavina olleiden muistityyppien välillä on suuria eroja niiden kirjoituskestävyydessä, energiankulutuksessa ja kirjoitusviiveissä. Johtuen siitä, että sulautettuna datamuistina käytetään useimmin SRAM:ia ei haihtuvien muistityyppien osalta ole tämän kirjallisuuskatsauksen puitteissa ollut mahdollista esittää vertailevia arvioita eri muistityyppien käytöstä sulautettuna muistina. Kirjallisuuskatsauksessa mukana olleiden, eri muistikonfiguraatioita edustavien suorittimien osalta voidaan kuitenkin todeta, että SRAM soveltuu matalahkon energiankulutuksensa ja nopeutensa ansiosta erinomaisesti haihtuvaksi sulautetuksi muistiksi.

Haihtumattomien muistityyppien osalta FRAM osoittautuu tämän kirjallisuuskatsauksen perusteella varteenotettavaksi vaihtoehdoksi flashille. Vaikka FRAM voisi toimia tutkijoiden tavoittelemana yhdistettynä sulautettuna muistina, on juuri energiatehokkuuden näkökulmasta tarkasteltuna perusteltua kuitenkin käyttää sen rinnalla SRAM:ia.

Koska kaupallisissa mikro-ohjaimissa sulautettu muisti koostuu useammasta muistityypistä, tarkasteltiin tässä kirjallisuuskatsauksessa sulautettua muistia useamman muistityypin yhdistelminä. Tätä varten valittiin kaksi mikro-ohjainta, jotka edustavat eri suoritinarkkitehtuureja ja erilaisia muistikonfiguraatioita. Näitä mikro-ohjaimia tarkasteltiin esimerkinomaisesti huomioimalla muistityyppien tehtävien lisäksi niiden ohjelmointiin liittyviä näkökohtia.

LÄHDELUETTELO

[1] Noergaard T. (2013) Embedded systems architecture – A comprehensive guide for engineers and programmers. Elsevier, Oxford. https://www.scribd.com/read/282542777/Embedded-Systems-Architecture-A-Comprehensive-Guide-for-Engineers-and-Programmers?mode=standard#

[2] Boukhobza J. & Olivier P. (2017) Flash memory integration – Performance and energy considerations. Elsevier Ltd, Oxford. https://www.scribd.com/read/341634402/Flash-Memory-Integration-Performance-and-Energy-Issues?mode=standard#

[3] Alonso D.A., Mamagkakis S., Poucet C., Péon-Quirós M., Bartzas A., Catthoor F. & Soudris D. (2015) Dynamic memory management for embedded systems. Springer International Publishing, Switzerland. https://www.researchgate.net/publication/283781405_Dynamic_Memory_Management_for_Embedded_Systems/download

[4] Artes A., Ayala J.L., Huisken J. & Catthoor F. (2013) Survey of Low-Energy Techniques for Instruction Memory Organisations in Embedded Systems. J Sign Process Syst 70:1-19, Springer Science + Business Media, Berlin. https://link-springer-com.pc124152.oulu.fi:9443/article/10.1007%2Fs11265-012-0694-2

[5] Davies J. (2008) MSP430 Microcontroller basics. Newnes, Elsevier Ltd., Oxford. https://www.scribd.com/read/282519710/MSP430-Microcontroller-Basics?mode=standard#

[6] megaAVR Data sheet (2018) Microchip Technology Inc. Chandler. http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf

[7] Harvard Architecture (2019). Science Direct, Elsevier B.V., Amsterdam. https://www.sciencedirect.com/topics/computer-science/harvard-architecture (luettu: 23.2.2019)

[8] AMD64 Architecture Programmer's Manual (2012). Advanced Micro Devices, Inc. http://developer.amd.com/wordpress/media/2012/10/24593_APM_v2.pdf

[9] Power Architecture Processors (2019). NXP Semiconductors, Eindhoven. https://www.nxp.com/products/processors-and-microcontrollers/power-architecture-processors:POWER-ARCHITECTURE?lang=en&lang_cd=en& (luettu: 30.3.2019)

[10] ARM Community, ARM Limited, Cambridge. https://community.arm.com/ (luettu 5.3.2019)

[11] Riho Y. & Nakazato K. (2014) Partial acces mode: New method for reducing power consumption of dynamic random access memory. IEEE Transactions on VLSI systems, Vol. 22, No. 7, July 2014, 1461, New York. https://ieeexplore-ieee-org.pc124152.oulu.fi:9443/stamp/stamp.jsp?tp=&arnumber=6620974

[12] Drepper U. (2007) What every programmer's should know about memory - Part 1. LWN.net, Eklektix, Inc., https://lwn.net/Articles/250967/ (luettu: 23.3.2019)

[13] Semiconductor technology online, http://maltiel-consulting.com/Semiconductor_technology_memory.html (luettu: 20.3.2019)

[14] Ziff Davis, LLC, PCMag Digital Group, https://www.pcmag.com/encyclopedia/term/50982/sdram (luettu: 11.4.2019)

[15] Main memory: DDR4 & DDR5 SDRAM. Jedec Committee. Arlington. https://www.jedec.org/category/technology-focus-area/main-memory-ddr3-ddr4-sdram

[16] Heath S. (2003) Embedded systems design. EDN Series for design engineers. Newnes, Elsevier Science, Oxford. https://www.scribd.com/read/282484311/Embedded-Systems-Design?mode=standard

[17] Noergaard T. (2013) Embedded systems architecture – A comprehensive guide for engineers and programmers. Newnes, Elsevier Inc., Oxford. https://www.scribd.com/read/282542777/Embedded-Systems-Architecture-A-Comprehensive-Guide-for-Engineers-and-Programmers?mode=standard#

[18] https://fi.wikipedia.org/wiki/SRAM (luettu 7.4.2019) SRAM

[19] Grupp L.M. Caulfield A. & Coburn J. (2009) Characterizing flash memory: anomalies, observations and applications. Micro'09, December 12-16, New York. https://cseweb.ucsd.edu/~swanson/papers/Micro2009FTest.pdf

[20] Micron and Intel extend their leadership in 3D NAND-flash memory (2018). News release, Intel Newsroom, Intel Corporation, Santa Clara. https://newsroom.intel.com/news-releases/micron-intel-extend-their-leadership-3d-nand-flash-memory/#gs.Lx0IRDBF

[21] Jung M., Wilson E.H., Donofrio D. Shalf J., Kandemir M.T. (2012) NANDFlashSim: Intrinsic latency variation aware NAND Flash memory system modeling and simulation at microarchitecture level. IEEE, New York. https://ieeexplore-ieee-org.pc124152.oulu.fi:9443/stamp/stamp.jsp?tp=&arnumber=6232389

[22] https://fi.wikipedia.org/wiki/SSD (luettu 17.4.2019) SSD

[23] Thanigai P. & Goh W. (2014) MSP430 FRAM Quality and reliability. Application report, SLAA526A, Texas Instruments Inc., Dallas. http://www.ti.com/lit/an/slaa526a/slaa526a.pdf

[24] Thanigai P. & Goh W. (2014) MSP430 FRAM technology – How to and best practices. Application report, SLAA628, Texas Instruments Inc., Dallas. http://www.ti.com/lit/an/slaa628/slaa628.pdf

[25] FRAM FAQs (2014). Texas Instruments Inc., Dallas. http://www.ti.com/lit/ml/slat151/slat151.pdf

[26] Kim M., Lee J., Kim Y., Song Y.H. (2018) An analysis of energy consumption under various memory mappings for FRAM-based IoT devices. IEEE, New York. https://ieeexplore-ieee-org.pc124152.oulu.fi:9443/stamp/stamp.jsp?tp=&arnumber=8355212

[27] Itoh K. (2011) Embedded memories: Progress and a look into the future. IEEE Design & Test of computers, January/February 2011, s. 10-13, IEEE, New York. https://ieeexplore-ieee-org.pc124152.oulu.fi:9443/stamp/stamp.jsp?tp=&arnumber=5708258

[28] Jew T. (2015) Embedded microcontroller memories – Application memory usage. IEEE, New York. https://ieeexplore-ieee-org.pc124152.oulu.fi:9443/stamp/stamp.jsp?tp=&arnumber=7150284

[29] Hu J., Xue C.J., Zhuge Q., Tseng W., Sha E. (2013) Data allocation optimization for hybrid scratch pad memory with SRAM and nonvolatile memory. IEEE Transactions on VLSI Systems, Vol. 21, No. 6, IEEE, New York. https://ieeexplore-ieee-org.pc124152.oulu.fi:9443/stamp/stamp.jsp?tp=&arnumber=6248275

[30] Embedded system articles (2012-2018). Microcontroller.com, CPU Technologies https://microcontroller.com/news/embedded_news.asp (luettu: 22.3.2019)

[31] Rabbit 6000 Microprocessor User's Manual (2013). Digi International Inc., Hopkins. https://www.digi.com/resources/documentation/digidocs/pdfs/90001108.pdf

[32] dsPIC33F 16-bit Microcontrollers and digital signal controllers (2012). Microchip Technology Inc., Chandler. http://ww1.microchip.com/downloads/en/DeviceDoc/75018c.pdf

[33] MSP430FR599x Mixed-signal Microcontrollers (2018). Texas Instruments Inc., Dallas. http://www.ti.com/lit/ds/symlink/msp430fr5994.pdf

[34] 32-bit Microcontroller Traveo Family S6J3200 Series Hardware Manual (2018). Cypress Semiconductor, San Jose. https://www.cypress.com/file/254046/download

[35] Intel Quark Microcontroller D2000 – Features. Intel Corporation, Santa Clara. https://www.intel.com/content/www/us/en/embedded/products/quark/mcu/d2000/overview.html

[36] Jayakumar H., Raha A. & Raghunathan V. (2016) Energy-aware memory mapping for hybrid FRAM-SRAM MCUs in IoT edge devices. In: International Conference on Embedded Systems, IEEE Computer Society, s. 264-269, Washington D.C. https://ieeexplore-ieee-org.pc124152.oulu.fi:9443/stamp/stamp.jsp?tp=&arnumber=7434963&tag=1

[37] Bez R. & Pirovani A. (2014) Overview of non-volatile memory technology: markets, technologies and trends. In: Nishi, Yoshio (toim.) Advances in non-volatile memory and storage technology [verkkodokumentti]. Elsevier, Woodhead Publishing Series in Electronic and optical materials: Number 64. Cambridge. s. 33 – 76. Saatavissa: https://www.scribd.com/read/282662305/Advances-in-Non-volatile-Memory-and-Storage-Technology?mode=standard# (luettu: 15.2.2019)

[38] MSP 430FR4xx and MSP430FR2xx Family – User's Guide (2018). Texas Instruments Inc., Dallas. http://www.ti.com/lit/ug/slau445h/slau445h.pdf

[39] Microchip Technology Inc. . https://www.microchip.com/ (luettu: 5.3.2019)

[40] Micheloni R. & Crippa L. (2014) Multi-bit NAND Flash memories for ultra high density storage devices. In: Nishi Y. (toim.) Advances in non-volatile memory and storage technology. Elsevier, Woodhead Publishing Series in Electronic and optical materials: Number 64. Cambridge. s. 158 – 228. https://www.scribd.com/read/282662305/Advances-in-Non-volatile-Memory-and-Storage-Technology?mode=standard#

[41] MSP 430 programming with the JTAG interface (2019). User's guide, SLAU320AF, Texas Instruments Inc., Dallas. http://www.ti.com/lit/ug/slau320af/slau320af.pdf

[42] MSP 430 flash devices bootloader (BSL) (2019). User's guide, SLAU319W, Texas Instruments Inc., Dallas. http://www.ti.com/lit/ug/slau319w/slau319w.pdf

[43] Ganssle J., Noergaard T., Eady F., Huddleston C., Edwards L., Katz D.J., Gentile R., Arnold K., Hyder K., Perrin B. (2008) Embedded hardware – Know it all. Newnes, Elsevier Inc., Amsterdam.

[44] Schuss C. (2017). Measurement techniques and results aiding the design of photovoltaic energy harvesting systems. Acta Universitatis Ouluensis, C Technica 619, Oulu. http://jultika.oulu.fi/files/isbn9789526215914.pdf

[45] Katz D.J. & Gentile R. (2008) Memory systems. In: Ganssle J., Noergaard T., Eady F., Huddleston C., Edwards L., Katz D.J., Gentile R., Arnold K., Hyder K., Perrin B. (2008) Embedded hardware – Know it all. Newnes, Elsevier Inc., Amsterdam.