

(19)



SUOMI - FINLAND

(FI)

PATENTTI- JA REKISTERIHALLITUS

PATENT- OCH REGISTERSTYRELSEN

FINNISH PATENT AND REGISTRATION OFFICE

(10) FI 940168 A7

**(12) JULKISEKSI TULLUT PATENTTIHAKEMUS
PATENTANSÖKAN SOM BLIVIT OFFENTLIG
PATENT APPLICATION MADE AVAILABLE TO THE
PUBLIC**

(21) Patentihakemus - Patentansökan - Patent application **940168**

(51) Kansainvälinen patenttiluokitus - Internationell patentklassifikation -
International patent classification
G06F 12/02

(22) Tekemispäivä - Ingivningsdag - Filing date **15.07.1992**

(23) Saapumispäivä - Ankomstdag - Reception date **13.01.1994**

(41) Tullut julkiseksi - Blivit offentlig - Available to the public **04.03.1994**

(43) Julkaisupäivä - Publiceringsdag - Publication date **13.06.2019**

(86) Kansainvälinen hakemus - **15.07.1992 PCT/US1992/005925**
Internationell ansökan - International
application

(32) (33) (31) Etuoikeus - Prioritet - Priority

15.07.1991 US 730244

(71) Hakija - Sökande - Applicant

1 • Quarterdeck Office Systems Inc., 150 Pico Boulevard, Santa Monica Cal. 90405, USA, AMERIKAN YHDYSVALLAT, (US)

(72) Keksijä - Uppfinnare - Inventor

1 • Spear, Dan, USA, AMERIKAN YHDYSVALLAT, (US)

2 • Mayer, Larry, USA, AMERIKAN YHDYSVALLAT, (US)

(74) Asiamies - Ombud - Agent

Kolster Oy Ab, Salmisaarenaukio 1, 00180 Helsinki

(54) Keksinnön nimitys - Uppfinningens benämning - Title of the invention

Muistinhallintamenetelmä

Förfarande för minneshantering

Muistinhallintamenetelmä

Keksinnön tausta

1. Keksinnön ala

5 Keksintö liittyy yleisesti tietokonemuistijärjestelmien alaan ja erityisesti muistinhallintajärjestelmään tietokoneen muistin käytön ohjaamiseksi ja optimoimiseksi.

2. Tekniikan taso

10 Tyypillinen tietokonejärjestelmä sisältää useita yksiköitä tai komponentteja. Tietokonejärjestelmät sisältävät tyypillisesti keskusyksikön (CPU), kuten mikroprosessorin. Mikroprosessori on ohjelman ohjaama laite, joka saa, dekodaa ja suorittaa käskyjä. Tietokonejärjestelmä sisältää myös tallennuskomponentteja järjestelmän käyttö-
15 ohjelmiston, sovellusohjelmakäskyjen ja datan tallentamiseksi. Nämä tallennuskomponentit voivat olla lukumuisteja (ROM), hajasaantimuisteja (RAM), levy- tai nauhamuisteja tai joitain muita sopivia tallennusvälineitä.

20 Tietokonejärjestelmä sisältää tyypillisesti myös otto/antokomponentit (I/O-komponentit) ulkoisten laitteiden kytkemiseksi mikroprosessoriin. Tietokonejärjestelmän osana voivat olla myös erityistä tarkoitusta varten olevat komponentit, kuten muistinhallintayksiköt tai laskusuoritimet.

25 Tietokoneita käytetään datankäsittelyyn. Datankäsittelyn mahdollistamiseksi ottodata täytyy tallentaa siihen asti, kunnes keskusyksikkö (CPU) käyttää sitä. Myös antodata täytyy tallentaa sen jälkeen kun se on käsitelty. Joidenkin käsittelytoimintojen aikana CPU voi tarvita myös
30 datan väliaikaista tallentamista, kun tuolle datalle suoritetaan käskyjä. Lisäksi sovellusohjelman, joka ohjaa käsittelyä, ja käyttöjärjestelmän, jonka alaisuudessa ohjelman ajo tapahtuu, täytyy olla CPU:n ulottuvilla. Tämä informaatio on tehty CPU:lle saatavilla olevaksi tallentamalla se resurssiin, joka tunnetaan "keskusmuistina".
35

Keskusmuistina tunnettu muistikomponentti on niukka resurssi, jota allokoidaan dynaamisesti käyttäjille, datalle, ohjelmille tai prosesseille. Keskusmuisti on tyyppillisesti piipohjainen muisti, kuten RAM. Monissa sovel-
 5 luksissa käytetään keskusmuistina dynaamista hajasaanti-
 muistia (DRAM). Prosessorin keskusmuisti on "tavuorgani-
 soitu". Toisin sanoen muisti on järjestetty 8-bittisten
 tavujen sekvenssiksi, ja tavu on pienin informaatioyksik-
 10 kö, joka muistista saadaan. Eräässä sopimuksessa koko ri-
 vin valitsee riviosoite, ja sarakkeisiin päästään kahdek-
 san bitin ryhmissä. Joissakin toteutuksissa saadaan ker-
 ralla 16-bittiset sanat (2 tavua) tai 32-bittiset sanat (4
 tavua).

Jotta CPU:lla olisi pääsy keskusmuistiin, se täytyy
 15 kytkeä joko suoraan tai epäsuorasti CPU:hun. Keskusmuistin
 määrää, joka voidaan kytkeä suoraan keskusyksikköön, ra-
 joittaa CPU:n osoiteväylän leveys. Osoiteväylä on kytketty
 CPU:n ja muistin välille. Muistiin pääsemiseksi CPU:n täy-
 tyy antaa osoiteväylälle arvo, joka edustaa yksinomaan
 20 yhtä ainoata muistipaikkaa. Ainutkertaisten arvojen määrä,
 joka osoiteväylälle voidaan antaa, riippuu osoiteväylän
 leveydestä. Koska osoiteväylän leveys tietyn tyyppiselle
 CPU:lle, kuten esimerkiksi jollekin erityiselle mikropro-
 sessorille, on tavallisesti kiinteästi määriteltä, CPU:n
 25 osoitettavissa olevien muistipaikkojen määrä on rajoitet-
 tu.

Joskus on toivottavaa, että CPU:lle kytketään enem-
 män muistia, kuin mitä CPU:n osoitettavissa on. Lisämuis-
 tin kytkemisen mahdollistamiseksi käytetään menetelmää,
 30 jossa CPU:lle kytketään lisämuistia epäsuorasti. Sellai-
 silla mikroprosessoreilla kuin 8088 ja 8086 on 20-bittinen
 osoiteväylä. Koska n-bittisen osoiteväylän osoitettavissa
 olevien muistipaikkojen määrä on 2^n , 20-bittisen osoiteväy-
 län osoitettavissa olevien muistipaikkojen määrä on 2^{20} ,
 35 joka on jokseenkin yhtä kuin yksi miljoona (1M). Ollakseen

suoraan CPU:n saatavilla kaiken muistin täytyy sijaita vieretysten lohkoissa, joka ei ole kooltaan suurempi kuin niiden ainutkeraisten arvojen määrä, jotka voidaan esittää osoiteväylän biteillä. 20-bittisellä osoiteväylällä kaiken
 5 muistin täytyy olla vierekkäin 1M:n lohkoissa.

Vaikka on teoreettisesti mahdollista päästä lisämuistiin jollakin epäsuoralla menetelmällä, useimmat käyttöjärjestelmät tukevat ainoastaan suoraa pääsyä muistiin. Jos käyttöjärjestelmä ei tue epäsuoria menetelmiä lisämuistiin pääsemiseksi, sovellusohjelman ja sen kaiken da-
 10 tan yhdessä tarpeellisen järjestelmäohjelmiston kanssa täytyy sisältyä vierekkäisesti järjestettyyn suorapääsyiseen lohkoon.

Kun osoiteväylän leveys rajoittaa suorasaantisen muistin määrää, järjestelmäohjelmiston varaaman muistin määrä rajoittaa sovellusohjelmia varten käytettävissä ole-
 15 van vapaan muistin määrää. Sovellusohjelmia varten käytettävissä olevan muistin määrä on siten vähemmän kuin suorasaantinen kokonaismäärä. Kun sovellusohjelmat vaativat
 20 enemmän muistia kuin niitä varten on käytettävissä, täytyy löytää menetelmä käytettävissä olevan helpposaantisen muistin kasvattamiseksi.

Tekniikan tasossa on tehty lukuisia yrityksiä käytettävissä olevan muistin määrän kasvattamiseksi. Eräs
 25 tekniikan tasossa käytetty menetelmä tunnetaan laajennusmuistin määrittämisellä (EMS). EMS varaa osan jatkomuistista, joka ei ole suoraan saatavilla, käytettäväksi laajennusmuistina ja jakaa sen sivuihin. Kytkemällä laajennusmuistin yhden sivun kerrallaan osoiteavaruuteen, joka on suoraan CPU:n saatavilla, EMS pystyy pääsemään virtuaalisesti
 30 rajoittamattomaan muistimäärään. EMS:ltä kuluu kuitenkin aikaa sivujen vaihtamiseen. Jos haluttu data ei ole suorasaantisessa muistissa sijaitsevassa EMS-sivunkehityksessä, EMS:n täytyy vaihtaa pois sivunkehityksen senhetkinen sisältö ja vaihtaa tilalle laajennusmuistista sivu, joka sisäl-

tää halutun datan. Koska tällainen sivunvaihto vaatii aikaa, tietokoneen käsittelynopeus pienenee. EMS ei myöskään ole suoraan sovellettavissa kaikkiin sovellusohjelmistoihin. Sovellusohjelma täytyy kirjoittaa erityisesti käyttämään hyväksi EMS:ää, jos se on käytettävissä.

Eräs toinen tekniikan tasossa käytetty menetelmä pohjautuu LOADALL-käskyyn. LOADALL-käsky on käsky, jolla ei ole tukea ja joka on toteutettu ainoastaan 80286-mikroprosessorissa. Koska LOADALL-käsky on toteutettu ainoastaan 80286:ssa, tämä tekniikan tason menetelmä on käytettävissä ainoastaan tietokoneissa, jotka perustuvat 80286-mikroprosessoriin.

Vielä eräs tekniikan tason menetelmä käsittää järjestelmä-ROM:issa olevan järjestelmäohjelmiston tutkimisen ja uudelleenohjelmoinnin. Tässä menetelmässä järjestelmä-ROM voidaan ohjelmoida uudelleen suorittamaan tehtävänsä ja varaamaan samalla minimimäärä muistia. Tämä menetelmä mahdollistaa muistitilan, joka oli tuhlatu tehottomasta ROM-ohjelmoinnista johtuen, käyttämisen muihin tarkoituksiin. Tämä menetelmä vaatii kuitenkin kunkin ROM:in, jolla menetelmää voidaan käyttää, kunkin version manuaalisen uudelleenohjelmoinnin.

Kaikilla näillä tekniikan tason menetelmillä on haittapuolia. Jotkin näistä menetelmistä antavat lisämuistia, mutta tuottavat tietokonejärjestelmän heikentyneen toiminnan johtuen vaiheista, jotka vaaditaan, jotta haluttu data saadaan saatavilla olevaksi. Eräät toiset menetelmät ovat käytettävissä ainoastaan järjestelmissä, jotka perustuvat erityiseen keskusyksikköön. Jotkin menetelmät ovat ROM-spesifisiä, ja ne täytyy toteuttaa erikseen kunkin ROM:in kullekin versiolle. Eräät muut menetelmät ovat spesifisiä tietyille sovellusohjelmille, eikä niitä voida soveltaa suoraan kaikelle sovellusohjelmistolle.

Keksinnön yhteenveto

Keksintö antaa lisää käytettävissä olevaa muistia CPU:n suorasaantiseen osoiteavaruuteen. Koska keksintö ei ole CPU-, ROM- tai sovellusohjelmaspesifinen eikä heikennä toimintaa yhtä paljon kuin tekniikan tason menetelmät, keksinnöllä vältetään tekniikan tason haittapuolet, ja se antaa yleisemmän sovellettavuuden sekä paremman suorituskyvyn.

Keksinnössä ROM:iin sisältyvä järjestelmäohjelmisto siirretään pois normaalista osoiteavaruudesta, ja tuota osoiteavaruutta käytetään muihin tarkoituksiin. Keksinnöllä hallitaan muistia tavalla, joka sallii ROM-osoiteavaruuden käytön muihin tarkoituksiin, kun samalla mahdollistetaan pääsy ROM:iin sisältyvään ohjelmistoon.

Keksinnön parhaana pidetyssä suoritusmuodossa järjestelmäohjelmiston sisältämä ROM sijoitetaan kartalla jatkomuistiosoitteavaruuteen, ja siihen on pääsy suojatilassa. Jatko-RAM sijoitetaan kartalla sitten osoiteavaruuteen, jossa ROM:it sijaitsivat aikaisemmin. Keksinnön parhaana pidetty suoritusmuoto mahdollistaa myös kaikkien ROM-pääsy-yritysten suuntaamisen uudelleen uudelleensijoitettuun ROM-vedokseen sekä alkuperäiseen ROM-osoiteavaruuteen sijoitetun jatko-RAM:in käytön muihin tarkoituksiin, kuten sovellusohjelmia varten.

Keksinnön parhaana pidetty suoritusmuoto on menetelmä muistimäärän, joka on osoitettavissa CPU:n normaalin muistiosoitusalueen puitteissa, lisäämiseksi, ja se mahdollistaa siten suoran ja helpon pääsyn dataan. Keksinnön parhaana pidetty suoritusmuoto sijoittaa uudelleen järjestelmä-ROM:in ja näyttö-ROM:in, mutta sitä voidaan soveltaa yhtä hyvin muihin ROM-muisteihin.

Keksinnössä suoritetaan alustusrutiini ROM-muistin sijoittamiseksi kartalla ulkoiseen muistiosoitteavaruuteen. Sieppausrutiini on asennettu sieppaamaan kaikki ROM-haut. Jatko-RAM allokoidaan sijoitettavaksi ja sijoitetaan osoi-

teavaruuteen, jonka ROM aikaisemmin varasi. Normaaaleja sovellusohjelmia, kun ne on asennettu, voidaan suorittaa tietokonejärjestelmässä. Lisä-RAM, joka on sijoitettu alkuperäiseen ROM-osoiteavaruuteen, voidaan allokoida käytettäväksi sovellusohjelmille tai muihin tarkoituksiin, 5 mukaan lukien TRS-ohjelmat (terminate-and-stay resident programs), mutta ei niihin rajoittuen. Kun yritetään pääsyä ROM-muistiin, ROM-haun sieppausrutiini sieppaa pääsyyrityksen ja suuntaa sen uudelleen uudelleensijoitettuun ROM-vedokseen. Kun uudelleensijoitetun ROM:in rutiinia kutsutaan, sen suoritusta tarkkaillaan sen varmistamiseksi, että se ei yritä päästä ROM-datataulukoihin tai alirutiineihin alkuperäisessä ROM-osoiteavaruudessa, mikä tuottaisi järjestelmän luhistumisen. Kun uudelleensijoitetun ROM:in rutiini on suoritettu loppuun, kaikki ROM-suoritukselta saadut tulokset tutkitaan ja käsitellään kaiken epätarkan datan, joka saattaisi tuottaa virheen tai järjestelmän luhistumisen, eliminoimiseksi. Toimimalla tavalla, joka ei merkittävästi heikennä toimintaa ja joka ei rajoitu spesifiseen CPU:hun, ROM:iin tai sovellusohjelmaan, keksintö tarjoaa käyttöön paljon tehokkaamman ja yleisemmin sovellettavan menetelmän käytettävissä olevan muistimäärän, johon CPU:lla on helppo pääsy, lisäämiseksi tietokonejärjestelmässä.

25 Piirustusten kuvaus

Kuvio 1 on lohkokkaavio, joka kuvaa tyypillisen tietokonejärjestelmän rakennetta.

Kuvio 2 on muistikartta, joka kuvaa tietokonejärjestelmän erästä mahdollista muistiorganisaatiota.

30 Kuvio 3 on piirros, joka kuvaa tiettyjen mikroprosessoreiden avotilassa käytettyä osoitusmenetelmää.

Kuvio 4 on piirros, joka kuvaa tiettyjen mikroprosessoreiden suojatilassa käytettyä osoitusmenetelmää.

35 Kuvio 5 on muistikartta, joka kuvaa erään tyypillisen tietokonejärjestelmän muistiorganisaatiota.

Kuvio 6 on muistikartta, joka kuvaa tekniikan tason ensimmäistä menetelmää.

Kuvio 7 on muistikartta, joka kuvaa tekniikan tason toista menetelmää.

5 Kuvio 8 on muistikartta, joka kuvaa tekniikan tason neljättä menetelmää.

Kuvio 9 on muistikartta, joka kuvaa tekniikan tason kolmatta menetelmää.

10 Kuvio 10 on lohkokaavio, joka kuvaa tekniikan tason viidettä menetelmää käyttävän tyypillisen tietokonejärjestelmän rakennetta.

Kuvio 11 on muistikartta, joka kuvaa erästä mahdollista muistiorganisaatiota tietokonejärjestelmässä, joka käyttää tekniikan tason viidettä menetelmää.

15 Kuvio 12 on muistikartta, joka kuvaa tekniikan tason viidettä menetelmää.

Kuvio 13 on muistikartta, joka kuvaa keksinnön suojatilasuoitusmuotoa ja karttasijoitusmuotoa.

20 Kuvio 14 on vuokaavio, joka kuvaa keksinnön suojatilasuoitusmuodon alustusrutiinia.

Kuvio 15 on vuokaavio, joka kuvaa keksinnön suojatilasuoitusmuodon alustuskäsittelijää.

Kuvio 16 on vuokaavio, joka kuvaa keksinnön alustusrutiinin karttasijoitusmuotoa.

25 Kuvio 17 on vuokaavio, joka kuvaa keksinnön alustuskäsittelijän karttasijoitusmuotoa.

Keksinnön yksityiskohtainen kuvaus

Tässä kuvataan menetelmä helppopääsyisen lisämuistin aikaansaamiseksi tietokonejärjestelmässä. Seuraavassa kuvauksessa lukuisia erityisiä yksityiskohtia, kuten esimerkiksi tietokonejärjestelmätyyppi, muistiosoitepaikat, muistimäärät ja väylälevyydet jne., kuvataan yksityiskohtaisesti keksinnön kuvaamiseksi perusteellisemmin. Alan ammattilaiselle on kuitenkin selvää, että keksintöä voidaan soveltaa käytäntöön ilman näitä erityisiä yksityis-

30

35

kohtia. Muissa esimerkeissä hyvin tunnettuja piirteitä ei ole kuvattu yksityiskohtaisesti, jotta ei tarpeettomasti hämärretä keksintöä.

5 Kuviossa 1 on kuvattu tyypillinen tietokonearkkitehtuuri. Tietokonejärjestelmä sisältää CPU:n 101, RAM-muistin (keskusmuistin) 102, ROM-muistin (lukumuistin) 103 sekä I/O-lohkon (otto/antolohkon) 104, jotka on kaikki
10 kytketty järjestelmäväylään 107. I/O-lohko 104 tarjoaa pääsyn muihin järjestelmiin, kuten massamuistiin 106 väylän 105 kautta.

CPU 101 ohjaa tietokonetta, suorittaa käskyjä ja käsittelee dataa. CPU 101 kommunikoi muiden komponenttien kanssa järjestelmäväylän 107 kautta. CPU vastaanottaa otodataa tietokoneen muilta komponenteilta järjestelmäväylällä 107 ja lähettää antodataa tietokoneen muille komponenteille järjestelmäväylällä. Järjestelmäväylä 107 sisältää tavallisesti osoiteväylän, dataväylän ja erilaisia
15 muita ohjauslinjoja. Osoite- ja dataväyliä leveys sekä ohjauslinjojen määrä ja tyyppi vaihtelevat tietokonejärjestelmästä toiseen.
20

Kukin tietokonejärjestelmän komponentti, mukaan luettuna RAM 102, ROM 103 ja muistikartoitettu I/O 104, sisältää useita erillisiä muistipaikkoja. Pääsyn mahdollistamiseksi CPU:lle 101 näihin paikkoihin kullekin paikalle on osoitettu erityinen osoite. Kukin osoite on
25 specifinen binaariarvoyhdistelmä, joka voidaan lähettää osoiteväylällä. Koska useimmat muistilaitteet sisältävät useampia kuin yhden paikan, osoitteet yhden ainoan muistilaitteen kaikille muistipaikoille osoitetaan tavallisesti vierekkäisten paikkojen lohkona. Näille lohkoille osoitetaan usein osoitteet (ne sijoitetaan muistikartalle) myös
30 vierekkäisellä tavalla. Karttasijoittelussa voi kuitenkin olla osoittamattomien osoitteiden välejä tai osoitteita, jotka on varattu tulevaa käyttöä varten.

Muistikartta on hyödyllinen muistin organisaation ymmärtämiseksi tietokonejärjestelmässä. Tyypillinen muistikartta alkaa alhaisimmasta osoitteesta, johon mikroprosessorilla on pääsy, ja ulottuu korkeimpaan osoitteeseen, johon mikroprosessori voi päästä. Eräs esimerkki muistikartasta on esitetty kuviossa 2. Alhaisimmat muistin osoitteet, jotka on osoitettu alueena 201 kuviossa 2 esitettyssä kartassa, on allokoitu keskusmuistille RAM 102. Seuraavaksi korkeimmat osoitteet, jotka on osoitettu alueena 202, on allokoitu I/O-lohkolle 104. Alueena 203 osoitetut muistiosoitteet ovat käyttämättömiä. Käyttämättömien osoitteiden yläpuolella alue 204 sisältää ROM:eille allokoitua muistiosoitteita. Alueet 205 ja 206 ovat muistiosoitteita, jotka on allokoitu vastaavasti lisä-RAM:ille ja lisä-ROM:ille. Kuvio 2 on annettu ainoastaan esimerkkinä sen vapauden kuvaamiseksi, jolla tietokonejärjestelmän muistiarkkitehtuuri voidaan suunnitella. Se ei edusta minkään erityisen tietokonejärjestelmän todellista muistikarttaa.

Tietokonejärjestelmän toimintaa ohjaa sarja käskyjä, jotka tunnetaan "käyttöjärjestelmänä". Käyttöjärjestelmää käytetään ohjaamaan tietokonejärjestelmän perustoimintoja, kuten ottoa/antoa, ja se on tyypillisesti tallennettuna pysyvään muistin osaan, kuten ROM-muistiin 103. Esimerkkeihin käyttöjärjestelmistä kuuluvat MS-DOS ja PC-DOS.

Tietokonejärjestelmiä käytetään suorittamaan sovellusohjelmia. Esimerkkeihin sovellusohjelmista kuuluvat tekstinkäsittelijät, laskutaulukot, piirrosohjelmat, tietokannat jne. Tietyt sovellusohjelmat voivat olla tallennettuina ROM:iin. Yleensä sovellusohjelmat tallennetaan kuitenkin massamuistilaitteeseen, kuten esimerkiksi levyasemaan.

Kun tietokonejärjestelmä alustetaan, sovellusohjelmat, jotka CPU:n 101 on määrä suorittaa, siirretään myös

massamuistista 106 RAM:iin 102. Data, johon CPU:n 101 on tarkoitus kohdistaa toimenpiteitä, tallennetaan myös RAM:iin 102.

5 Sovellusohjelmien kokoa ja määrää, käyttöjärjestelmän kokoa ja sen datan määrää, joka on CPU:n 101 saatavilla, rajoittaa keskusmuistin koko. Rajaton muistimäärä olisi ihanteellinen, mutta se on mahdoton sekä taloudellisesti että teknisesti. Ensinnäkin on muistimäärälle rajana se, mihin käyttäjällä on varaa. Toiseksi on olemassa teknisiä rajoituksia muistimäärälle, jota tietty tietokone
10 voi tukea. Tietyn tietokoneen tukema muistimäärä riippuu CPU:n erityisestä tyypistä, jolle tietokone perustuu.

Esimerkiksi IBM PC perustuu 8088-mikroprosessorille. 8088-mikroprosessorissa on 20-bittinen osoiteväylä.
15 Tietokoneen osoitettavissa oleva muistimäärä on sen osoiteväylän leveyden funktio. CPU, jossa on n-bittinen osoiteväylä, voi päästä suoraan muistin 2^n tavuun tai sanaan. 8088-mikroprosessori voi siten päästä suoraan noin miljoonaan (1M) muistin tavuun tai sanaan. Vaikka on kehitetty
20 menetelmiä, jotka mahdollistavat epäsuoran pääsyn lisämuistiin CPU:n osoiteväylän rajojen ulkopuolella, tällaiset menetelmät ovat usein hankalia ja tehottomia. Siten on toivottavaa pitää niin paljon kuin mahdollista suorasaantisesta muistiavaruudesta käytettävissä sovellusohjelmiston käyttöä varten, jotta mahdollistetaan sovellusohjelmiston toimiminen niin tehokkaasti kuin mahdollista.
25

Vaikka olisi ideaalista varata niin paljon suorasaantista muistiavaruutta kuin mahdollista sovellusohjelmistoa varten ja pitää sovellusohjelmisto niin pienenä
30 kuin mahdollista sen varmistamiseksi, että se sopii suorasaantiseen muistiavaruuteen, näiden tavoitteiden toteuttaminen ei ole aina mahdollista. Koska vaaditaan järjestelmäohjelmisto sovellusohjelmiston hallintaa ja ohjausta varten, sen täytyy olla myös muistissa. Myös kun sovellus-

ohjelmiston mutkikkuus ja käyttäjädatan määrä kasvavat, sovellusohjelmisto vaatii enemmän muistia.

Lisäksi tietokonejärjestelmissä käytetään usein TSR-ohjelmia tunnettuja ohjelmia. Tällaiset ohjelmat antavat "kuumia avaimia" ja "putkahdusikkunoita", ja niitä käytetään suorittamaan taustatehtäviä, kuten esimerkiksi näyttämään kellonaika kuvanäytön nurkassa tai tarkkailemaan levyaseman aktiivisuutta. TSR-ohjelmia on kirjoitettu monille muille sovelluksille, ja käyttäjät haluavat usein pitää useita TSR-ohjelmia samanaikaisesti tietokoneessaan. Koska kukin TSR-ohjelma vaatii muistitilaan, johon se on sijoitettuna, TSR-ohjelmien lisääminen järjestelmään kasvattaa myös muistitarpeita.

Kuvio 5 kuvaa tyypillisen tietokoneen, nimittäin 8088/8086-mikroprosessoriperheeseen perustuvan ja MS-DOS:in alaisuudessa toimivan tietokoneen, kuten IBM PC:n, muistikarttaa. Kuvion 5 muistikartta ei ole ainoastaan mahdollinen muistikartta, vaan se on esimerkki tyypillisestä muistikartasta. Muistikartta on organisoitu alhaalta ylös muistipaikan nolla (501) ollessa pohjalla ja kartan jatkuessa korkeimpaan paikkaan muistissa sen huipulla. Muistikartassa on kolme perusaluetta. Ensimmäinen alue sisältää alimmat 640K muistia, ja siihen viitataan perinteisenä muistina 502. Perinteinen muisti 502 muodostuu kokonaan RAM:ista, joka sallii sekä luku- että kirjoitustoiminnot, vaikka kaikki järjestelmät eivät sisällä 640K:n muistia kokonaisuudessaan, ja jonkin verran muistiavaruutta voi jäädä käyttämättä.

Varhaisissa mikrotietokoneissa oli 16-bittiset osoiteväylät, mikä antaa 64K osoiteavaruutta. Kun sovelusten vaatima muistimäärä kasvoi, mikrotietokoneiden oli ylitettävä 16-bittisen osoiteväylän rajoitukset. Siten tuotiin markkinoille segmentoidulla osoituskaaviolla varustettu IBM PC, joka tulee 20-bittistä osoiteväylää. 20-bittinen osoiteväylä antaa 1024K:n tai 1M:n osoiteavaruutta.

den, joka on 16 kertaa 16-bittisen osoiteväylän osoiteavaruus. 1M näytti olevan erinomaisen suuri määrä muistia verrattuna 16-bittisellä osoiteväylällä varustettujen tietokoneiden 64K:n määrään. Johtuen muistipiirien korkeasta hinnasta tuohon aikaan asetettiin lisäksi mielivaltaisen raja RAM-muistille määrään 640K. 384K:n osoiteavaruutta kohdasta 640K kohtaan 1M varattiin tulevaa käyttöä varten, ensisijaisesti ROM:eille. Koska sen jälkeen tulleet tietokonemallit on suunniteltu olemaan taaksepäin yhteensopivia alkuperäisen IBM PC:n kanssa ja käyttämään samaa PC-DOS/MS-DOS-käyttöjärjestelmää, 640K:n raja rajoittaa edelleen sovelluksiin käytettävää muistimäärää moderneissa tietokoneissa.

Perinteinen muisti 502 on tärkeä osa tietokonejärjestelmää. Perinteistä muistia käytetään järjestelmäohjelmiston, sovellusohjelmiston, käyttäjädatan ja muun koodin ja datan, mukaan lukien TSR:t ja laiteohjaimet, tallentamiseen. Kuten kuviossa 5 on kuvattu, MS-DOS käyttää muistin alinta osaa oman koodinsa 506 ja siihen liittyvän datan 507 tallentamiseen. Sen yläpuolelle MS-DOS tallentaa sovellusohjelmiston, TSR:t ja laiteohjaimet, jotka on kuvattu kollektiivisesti osana 512.

Toinen muistin perusalue on 384K varattua muistia 503, joka on muistiosoitteissa 640K RAM:in rajan yläpuolella ja 1024K alapuolella. Varatun muistialueen 503 varaavat pääasiassa ROM:it, jotka ovat ainoastaan luettavia laitteita. Varatulla muistialueella olevat ROM:it sisältävät järjestelmä-ROM:in 504, näyttö-ROM:in 505 ja ehkä ROM:in 514 muita oheislaitteita, kuten kiintolevyasemia tai verkkorajapintoja, varten. Järjestelmä-ROM 504 sisältää koodin ja datan, jotka tukevat tietokoneen perustoimintoja. Näyttö-ROM 505 sisältää koodin ja datan kuvanäyttöön liittyvien toimintojen tukemiseksi. ROM:it 514 muita oheislaitteita varten tukevat noiden laitteiden käyttöä.

ROM:ien lisäksi varattu muistialue 503 sisältää myös muita tyyppejä erikoistunutta muistia. Eräs varatulla muistialueella 503 oleva muistityyppi on RAM, jota käytetään kuvakehyspuskureille. Kuvakehyspuskuriä käytetään tallentamaan informaatio, jota sillä hetkellä näytetään kuvanäytöllä. Informaatio voidaan yleensä sekä lukea kuvakehyspuskurista että kirjoittaa siihen, mutta kuvakehyspuskuriä ei voida tavallisesti käyttää tallentamaan yleistä informaatiota, koska mikä tahansa kuvakehyspuskuriin kirjoitettu informaatio muuttaa kuvanäytön ilmeä, ellei kuvakehyspuskuri ole epäaktiivinen ja pysy sellaisena.

Kuvakehyspuskureita varten käytetyt varatun muistin 503 alueet riippuvat erityiseen tietokonejärjestelmään asennetun näyttösovittimen erityisestä tyypistä. Kuten kuviossa 5 on kuvattu, jos EGA- tai VGA-näyttösovitin on asennettuna, grafiikkaa varten käytetty 64K:n kuvakehyspuskuri 508 varaa muistiavaruuden kohdasta 640K kohtaan 704K. EGA- tai VGA-sovittimet käyttävät myös 32K:n kuvakehyspuskuriä tekstin näyttämiseen. Tämä 32K:n kuvakehyspuskuri sijaitsee kohdasta 704K kohtaan 736K yksivärinäytölle ja kohdasta 736K kohtaan 768K väriinäytölle. Jos MDA-näyttösovitin on asennettuna, 16K:n kuvakehyspuskuri 509 varaa muistiavaruuden kohdasta 704K kohtaan 720K. Jos Hercules-näyttösovitin on asennettuna, 64K:n kuvakehyspuskuri varaa muistiavaruuden kohdasta 704K kohtaan 768K. Jos CGA-näyttösovitin on asennettuna järjestelmään, kuvakehyspuskuri 510 varaa 32K muistiavaruutta kohdasta 736K kohtaan 768K. Jos EGA- tai VGA-näyttösovitin on asennettu, muistin lisämäärää käytetään näyttö-ROM:ia varten. Tyypillisesti 16K muistiavaruutta kohdasta 768K kohtaan 784K käytetään EGA- tai VGA-näyttö-ROM:ia 505 varten, vaikka näyttö-ROM 505 voi olla kooltaan erilainen ja/tai sijaita eri osoitetilassa.

Järjestelmä ROM 504, joka tukee tietokoneen perustoimintoja, varaa tyypillisesti esimerkiksi ylimmät 64K

varatusta muistialueesta 503 kohdasta 960K kohtaan 1024K. Jäljelle jäävä tila varatusta muistialueesta 503 on joko käyttämättömänä tai sitä käytetään muihin tarkoituksiin, mukaan lukien ROM:it, jotka tukevat muita oheislaitteita, tai EMS-sivunkehys.

Kolmas muistin perusalue on jatkomuisti 511, joka sisältää kaiken muistin määrän 1M:n yläpuolella. Koska 8088/8086-mikroprosessoreilla on ainoastaan 20-bittinen osoiteväylä, ne voivat osoittaa muistia ainoastaan määrän 1M, eivätkä pysty helposti tukemaan jatkomuistia 511. Mikroprosessorit, joissa on 24-bittinen osoituskyky, kuten 80286, voivat osoittaa muistia jopa 16M, mukaan lukien 15M:n jatkomuisti 511 varatun muistin 503 ja perinteisen muistin 502 muodostaman 1M määrän lisäksi. Mikroprosessorit, joissa on 32-bittinen osoituskyky, kuten 80386 ja 80486, voivat osoittaa muistia jopa 4G, mikä sisältää 4095M jatkomuistia 511 varatun muistin 503 ja perinteisen muistin 502 määrän 1M lisäksi.

Juuri määrän 1M yläpuolella sijaitsevaan jatkomuistin 511 alueeseen viitataan joskus HMA-alueena (high memory area) 513. HMA-alue 513 muodostuu 65 520 muistisanasta, jotka ovat osoitettavissa käyttäen tekniikan tason HMA-tekniikkaa.

Eräs kuvion 5 tekniikan tason muistikarttajärjestelmän haittapuoli on, että ainoastaan 1 megatavu muistia on suoraan saatavissa. Tuosta 1 megatavusta RAM-osoitukselle on asetettu mielivaltainen raja 640K. Usein käyttäjä vaatii, että enemmän kuin 640K RAM-muistia on suoraan osoitettavissa. Koska RAM-osoituksen mielivaltaista 640K:n rajaa ei voida muuttaa, tekniikan tason yritykset lisä-RAM-muistin aikaansaamiseen ovat keskittyneet RAM-rajan ja 1 megatavun rajan välisen varatun muistin 384K:n määrän hyväksikäyttöön. Alla kuvataan useita näistä tekniikan tason menetelmistä.

Tekniikan taso

Ensimmäinen tekniikan tason menetelmä käsittää välittömästi perinteisen muistin 602 640K rajan yläpuolella olevan rajatun muistin 603 allokoimisen käytettäväksi lisäksi tulevana perinteisenä muistina. Kuvioon 5 viitaten, varatun muistin osoiteavaruus kohdasta 640K kohtaan 704K, joka on kuvattu osana 601, on allokoitu EGA/VGA-kuvakehyspuskurille. Varattu muisti kohdasta 704K kohtaan 736K on allokoitu MDA/Hercules-kuvakehyspuskurille. Varatun muistin osoiteavaruus kohdasta 736K kohtaan 768K on allokoitu CGA-kuvakehyspuskurille. Useimmissa tietokonejärjestelmissä on kuitenkin ainoastaan yhdentyyppinen näyttösovitin. Jos tietokonejärjestelmässä on CGA- tai MDA/Hercules-näyttösovitin mutta siinä ei ole EGA/VGA-näyttösovitinta, EGA/VGA-kuvakehyspuskurille kohdasta 640K kohtaan 704K allokoitu muisti, joka on kuvattu osana 601, jää käyttämättä ja voidaan allokoida käytettäväksi uudelleen sovellusohjelman toimesta tai perinteisen muistin 602 jatkeena. Jos tietokonejärjestelmässä on ainoastaan CGA-näyttösovitin eikä siinä ole MDA/Hercules- tai EGA/VGA-sovitinta, MDA/Hercules- ja EGA/VGA-kuvakehyspuskureille kohdasta 640K kohtaan 736K ulottuva muistitila jää käyttämättä, ja perinteistä muistia 602 voidaan jatkaa kohtaan 736K.

Jotta perinteisen muistin 602 raja voidaan työntää yli kohdan 640K, täytyy ensiksi määrittää järjestelmään asennettujen näyttösovittimien tyyppi tai tyypit. Riippuen siinä olevien näyttösovittimien tyypeistä sopiva määrä käyttämätöntä RAM:ia ulkoisesta muistista täytyy sijoittaa kartalla johonkin käyttämättömään osoitetilaan, joka on kohdan 640K yläpuolella mutta sen vieressä. Sitten muistinallokointiproseduureja, joiden tarkoituksena on antaa muistia sovellusohjelmistolle ja TSR:ille, täytyy modifioida, niin että sallitaan 640K rajan takana olevan muistin allokoiminen.

Vaikka tämä menetelmä lisää helpposaantisen muistin määrää, sillä on lukuisia haittapuolia. Ensinnäkään se ei ole käyttökelpoinen tietokonejärjestelmissä, joissa on EGA- tai VGA-näytöt, jotka ovat lisääntyvässä määrin suosittuja. Toiseksi myös järjestelmissä, joissa ei ole EGA- tai VGA-näyttöjä, ainoastaan rajoitettu muistimäärä voidaan tällä menetelmällä tehdä saatavilla olevaksi. Järjestelmissä, joissa on MDA- tai Hercules-näytöt, voidaan saatavilla olevaksi tehdä ainoastaan 64K. Myös järjestelmissä, joissa on ainoastaan CGA-näyttö, voidaan saatavilla olevaksi tehdä ainoastaan 96K.

Toinen tekniikan tason menetelmä on samanlainen kuin ensimmäinen. Se käsittää myös perinteisen muistin 640K rajan työntämisen ylöspäin varatun muistin alueelle, kuten on kuvattu kuviossa 6. Se eroaa ensimmäisestä tekniikan tason menetelmästä kuitenkin siinä, että se saa aikaan myös varatun muistin, jota käytetään muihin tarkoituksiin, sijoittamisen uudelleen. Tämän menetelmän käytöstä tuloksena oleva muistikartta on kuvattu kuviossa 7. Tällä menetelmällä näyttö-RAM 703 ja näyttö-ROM 704 samoin kuin muu I/O-muisti 705 ja TSR-muisti 706 sijoitetaan uudelleen varatulle muistialueelle muistipaikkoihin, jotka ovat niin korkealla kuin mahdollista, jotta saadaan niin paljon kuin mahdollista käyttämättömästä varatusta muistiosoitteavaruudesta saataville perinteisessä muistissa 701 olevan 640K viereen. Käytännössä perinteisen muistin 701 640K:n raja voidaan työntää niinkin kauas kuin kohtaan 900K.

Tämän menetelmän toteuttamiseksi täytyy määrittää järjestelmään asennettujen näyttösovittimien tyyppi. Muut I/O-laitteet, jotka varaavat myös osan varatusta muistialueesta 708, täytyy myös määrittää. Sitten näyttö-RAM 703, näyttö-ROM 704 ja mahdollinen muu I/O-muisti 705 ja TSR-muisti 706 täytyy sijoittaa kartalla uudelleen alueelle, joka on juuri järjestelmä-ROM:in 707 alapuolella vara-

tun muistialueen 708 yläpäässä. Sitten kaikki kekeytykset ja muut osoittimet näyttö-RAM:ille 703, näyttö-ROM:ille 704, I/O-muistille 705 ja TSR-muistille 706 täytyy suunnata uudelleen heijastamaan uutta muistitilaa, johon tämä
 5 muisti on sijoitettu uudelleen. Seuraavaksi täytyy karttasijoittaa sopiva määrä käyttämätöntä RAM:ia ulkoisesta muistista lähellä kohtaa 640K olevaan varatun muistin saatavilla olevan osoitealuelohkoon. Lopuksi muistinallokointiproseduureja sovellusohjelmistolle ja TSR:ille täytyy
 10 modifioida, jotta sallitaan 640K rajan yläpuolella olevan varatun muistin osan, johon viitataan täytemuistina 702 ja joka on tehty saatavilla olevaksi, allokointi.

Tällä toisella tekniikan tason menetelmällä on myös haittapuolia. Se sallii enimmillään pääsyn 260K:n määrään
 15 varattua muistia. Koska sovellusohjelmalla on usein suora pääsy näyttö-RAM:iin 703, on myös vaikeaa siepata aina kaikki pääsy-yritykset ja suunnata ne uudelleen muistitilaan, johon näyttö-RAM 703 on uudelleensijoitettuna.

Kolmas tekniikan tason menetelmä toimii hiukan eritavoin kuin ensimmäiset kaksi. Se on kuvattu kuviossa 8. Koska TSR:t, laiteohjaimet ja verkkorajapintaohjelmat 803, jotka on normaalisti sijoitettu perinteiseen muistiin 801
 20 alle 640K rajan, varaavat jonkin verran perinteistä muistiavaruutta, jonka voisi muutoin käyttää sovellusohjelmistoto, enemmän muistia olisi käytettävissä sovellusohjelmistolle, jos nämä ohjelmat voitaisiin siirtää perinteisen
 25 muistin 801 ulkopuolella olevaan muistiin. Kolmas tekniikan tason menetelmä lisää sovellusohjelmistolle käytettävissä olevaa perinteisen muistin 801 määrää siirtämällä
 30 TSR:t, laiteohjaimet ja verkkorajapintaohjelmat 803 perinteisestä muistista 801 varattuun muistiin 802.

Tämän menetelmän toteuttamiseksi käyttämättömän varatun muistin 806 määrä täytyy ensiksi määrittää. Myös
 35 TSR:ien, laiteohjainten ja verkkorajapintaohjelmien 803 varaama perinteisen muistin määrä täytyy määrittää. Sitten

riittävä määrä allokoimatonta RAM:ia laajennusmuistista täytyy sijoittaa kartalla varatun muistin alueelle, jotta saadaan hankituksi muistitilaa TSR:lle, laiteohjaimille ja verkkorajapintaohjelmille. Seuraavaksi TSR:t, laiteohjaimet ja verkkorajapintaohjaimet 803 täytyy sijoittaa uudelleen käytettävissä olevaan muistiin 806 varatulla muisti-alueella 802.

Kolmannella tekniikan tason menetelmällä on myös haittapuolia. Ensinnäkin se voi käyttää ainoastaan allokoimatonta varattua muistialuetta 806. Se ei voi käyttää varattua muistialuetta 807, joka on allokoitu ROM:eille, kuvakehyspuskureille tai muuhun käyttöön. Myös TSR:ien, laiteohjainten ja verkkorajapintaohjelmien 803 uudelleen-sijoittelu täytyy tehdä huolellisesti sen varmistamiseksi, että kaikki viittaukset niihin suunnataan uudelleen niiden uusiin paikkoihin 804, 805.

Neljäs tekniikan tason menetelmä liittyy siihen, mitä kutsutaan ylämuistialueeksi (HMA). Tämän menetelmän käytöstä tuloksena oleva muistikartta on kuvattu kuviossa 9. Tämä menetelmä perustuu anomaliaan segmentoidussa osoituskaaviossa, joka on toteutettu 80286:ssa. Segmentoitu osoituskaavio käsittää 16-bittisen segmentin yhdistämisen 16-bittiseen siirrokseen 20-bittisen osoitteen tuottamiseksi. Vaikka 20-bittisen osoitteen tulisi teoreettisesti rajoittua 1M:ään, tietyt segmenttien ja siirrosten yhdistelmät voivat tuottaa osoitteet, jotka ulottuvat yli 1M rajan. Vaikka 8088- ja 8086-mikroprosessorit suunniteltiin rajoittamaan mahdollinen osoitteiden alue 1M:ään, 80286-mikroprosessorissa ei ole tällaista suojamekanismia. Koska 80286-mikroprosessori ei pidä tiukasti 1M:n rajaa muisti-osoitteissa, tätä menetelmää voidaan käyttää 80286-mikroprosessorin yhteydessä mahdollistamaan pääsy 64K:n lisämuistiin, joka tunnetaan HMA-alueena 903 ja joka sijaitsee juuri 1M:n yläpuolella.

Tämän menetelmän käyttämiseksi rekisteriin, joka tunnetaan "segmenttirekisterinä" ja jota käytetään tallentamaan ensimmäinen 16-bittinen segmentti, kun käytetään hyväksi segmentoitua osoituskaaviota, ladataan ensiksi
 5 FFFFH. Sitten rekisteriin, joka tunnetaan "siirrosrekisterinä", ladataan arvo väliltä 10H ja FFFH. Sitten mikroprosessori suorittaa HMA-saannin ikäänkuin se olisi saanti tavanomaisesta muistista 901.

Tähän neljänteen tekniikan tason menetelmään liittyy haittoja. Koska tämä menetelmä ei toimi tietokoneilla, jotka perustuvat 8088- tai 8086-mikroprosessoreihin, sitä ei voida käyttää varhaisissa PC-laitteissa tai niiden kanssa yhteensopivissa laitteissa. Myös tällä menetelmällä hankitun muistin määrä rajoittuu määrään 64K. Lisäksi so-
 10 vellusohjelmiston täytyy olla erityisesti kirjoitettua HMA:n allokointia ja siihen pääsyä varten, ja se ei saa olla ristiriidassa HMA:han pääsyn rajoitusten kanssa, mukaan lukien rajoitukset segmenttiaritmetiikassa ja levyjen lukemisessa ja kirjoittamisessa suoraan HMA:han. Lisäksi
 15 tämä menetelmä tunnetaan laajasti, ja on kirjoitettu ohjelmistoa, joka käyttää tätä menetelmää lisämuistin hankkimiseen, ja niinpä tällä menetelmällä hankittu muisti ei todennäköisesti ole saatavilla muuhun käyttöön useimmissa tietokonejärjestelmissä.

Viides tekniikan tason menetelmä käsittää muistin jakamisen sivuihin ja sen varmistamisen, että halutun datan sisältävä sivu on saatavilla olevassa muistissa sillä hetkellä, kun saantiin ryhdytään. Tästä menetelmästä on kaksi variaatiota. Ensimmäistä kutsutaan laajennusmuistin
 25 määrittämisestä (EMS). EMS on kuvattu kuviossa 12. EMS-pankki kytkee sivut jatkomuistista sivunkehyykseen 1203 varatulla muistialueella 1202, niin että mahdollistetaan suurten jatkomuistimäärien käyttö, vaikkakin ainoastaan yksi sivu tai muutamia sivuja on jollakin annetulla hetkellä
 30 1M:n alapuolella olevassa muistiavaruudessa. EMS:n yhtey-

dessä käytettynä jatkomuistiin viitataan laajennusmuistina 1204. Toista variaatiota kutsutaan virtuaalimuistiksi. Virtuaalimuisti on kuvattu kuviossa 10. Eräs mahdollinen virtuaalimuistilla varustetun järjestelmän muistikartta on esitetty kuviossa 11. Virtuaalimuisti sallii informaation, jota ei voida sisällyttää fyysiseen muistiin 1002, tallentamisen I/O-laitteeseen, kuten esimerkiksi kiintolevyasemaan. Data jaetaan sivuihin ja ladataan levyasemalta fyysiseen muistiin ainoastaan, kun CPU 1001 sitä tarvitsee.

10 Suuri määrä datasivuja voidaan sitten tallentaa kiintolevyille, kun taas ainoastaan muutamia sivuja tarvitsee pitää fyysisessä muistissa 1002.

EMS:n käyttämiseksi laajennusmuisti 1204 täytyy jakaa sivuihin. Myös alue varatussa muistissa täytyy allokoida käytettäväksi sivunkehyksenä 1203. Sivuhakemistoa, joka sisältää niiden sisällön ja niiden paikan laajennusmuistissa 1204, täytyy ylläpitää. Kun EMS on asennettu ja toimii, ohjelmiston, joka tukee EMS:ää, täytyy tarkkailla muistiin pääsyä sen varmistamiseksi, että haluttu EMS-data on EMS-sivunkehysessä 1203. Jos haluttu EMS-data ei ole EMS-sivunkehysessä 1203 hakuhetkellä, ohjelmiston, joka tukee EMS:ää, täytyy määrittää, mikä sivunkehyksen 1203 sivu tai mitkä sivut täytyy pankkikytkeä pois sivunkehysestä tilan tekemiseksi halutulle datalle. Sitten sen täytyy löytää sopiva sivu laajennusmuistista 1204. Sopiva sivu täytyy sitten ladata sivunkehyksen 1203 saatavilla olevaksi sivuksi. Sitten muistisaanti täytyy suunnata uudelleen oikeaan osoitteeseen sivunkehysessä 1203, niin että mahdollistetaan halutun datan lukeminen tai kirjoittaminen.

Virtuaalimuistijärjestelmän käyttämiseksi kiintolevyille allokoitu virtuaalimuisti 1003 täytyy jakaa sivuihin. Sivuhakemistoa, joka sisältää niiden sisällöt ja niiden sijainnin muistissa, täytyy ylläpitää. Kun virtuaalimuistijärjestelmä on alustettu ja toimii, sen täytyy tark-

kailla muistisaantia sen varmistamiseksi, että haluttu data on fyysisessä muistissa 1002. Jos haluttu virtuaalimuistin data ei ole fyysisessä muistissa 1002 hakuhetkellä, virtuaalimuistijärjestelmän täytyy määrittää, mille muistin sivulle tai sivuille ainakin todennäköisesti on päästävä tulevaisuudessa, ja sen täytyy tallentaa ne takaisin kiintolevylle. Sitten sen täytyy löytää sopiva sivu kiintolevyn virtuaalimuistista 1003. Sopiva sivu tai sopivat sivut täytyy sitten ladata fyysisen muistin 1002 valituksi sivuksi tai valituiksi sivuiksi. Sitten muistisaanti täytyy suunnata uudelleen oikeaan osoitteeseen vasta ladatulla sivulla tai ladatuilla sivuilla, niin että mahdollistetaan halutun datan lukeminen tai kirjoittaminen.

Vaikka EMS- ja virtuaalimuistijärjestelmissä on samankaltaisuuksia, virtuaalimuistijärjestelmät toimivat yleensä paljon automaattisemmin ja tuntumattomammin sovellusohjelmistolle, kuin EMS toimii. Sovellusohjelmisto ei ehkä ole edes tietoinen siitä, että virtuaalimuistijärjestelmä on mukana ja toimii tietokonejärjestelmässä, kun taas sovellusohjelmiston ei ainoastaan pidä tunnistaa EMS:n olemassaolo, vaan sen täytyy erityisesti tukea EMS:ää ja suunnata EMS:n toimintaa.

Vaikka tämä viides tekniikan tason menetelmä pystyy teoreettisesti mahdollistamaan pääsyn lisättyyn muistimäärään, sillä on myös haittapuolia. Muistisaantiin käytetty aika lisääntyy tällä menetelmällä, koska CPU:n aikaa täytyy omistaa muistinhallinnalle. Tallenteita täytyy jatkuvasti päivittää sivujen paikan osoittamiseksi sekä sivunkehyksessä 1203 että laajennusmuistissa 1204 ja myös niiden sisällön osoittamiseksi. Nämä toiminnot ovat poissa ajasta, jonka CPU voisi omistaa muille tehtäville, ja siksi ne heikentävät toimintaa.

Kuudes tekniikan tason menetelmä käsittää ROM-muistien kokoonpuristamisen ja optimoinnin. Tätä menetelmää ei voida soveltaa yleisesti, vaan se täytyy suorittaa manuaa-

lisesti kunkin ROM:in kullekin versiolle. Tämän menetelmän tarkoituksena on pienentää ROM-muistien varaaman muistitilan määrää, niin että ROM:eille allokoitu mutta niiden käyttämättä jättämä muistitila voidaan allokoida uudelleen muihin tarkoituksiin. ROM-muistien varaamaa muistitilaa pienennetään etsimällä ja eliminoimalla tehoton koodaus ja osat ROM:ista, joita ei koskaan haeta.

Koska tätä menetelmää täytyy käyttää eri tavoin kullekin erityiselle ROM:ille, menetelmän yksityiskohtia ei voida kuvata yleisesti. Tämän menetelmän käyttämiseksi täytyy tutkia huolellisesti olemassa oleva ROM-koodi ja -data ja paikallistaa kaikki alueet, joita tulisi muuttaa tai jotka tulisi eliminoida. Sitten kun nämä muutokset on tehty, ROM-koodi ja -data täytyy koota uudelleen. Uusi pienempi ROM-koodi täytyy sitten sijoittaa kartalla alkuperäisen ROM:in yläosaan. Täytyy varautua myös tällä menetelmällä käytettäväksi saadun muistitilan allokoimiseen käytettäväksi muihin tarkoituksiin.

Tämän menetelmän ilmeinen haittapuoli on, että sitä ei voida soveltaa yleisesti. Tähän menetelmään perustuvat tuotteet tulevat vanhentuneiksi niin pian kuin uusi ROM-versio julkistetaan. Myös merkittävä määrä aikaa, vaivaa ja taitoa vaaditaan kunkin ROM:in kunkin version kokoonpuristamisen ja optimoinnin suorittamiseen.

Seitsemäs tekniikan tason menetelmä käsittää lisämuistin aikaansaamisen väliaikaisesti varatun muistin alueelle TSR:ien, joiden on määrä jäädä pysyvästi varatun muistin alueelle, alustamiseksi. Sijoittamalla TSR:t muistiin, joka on sijoitettu varatun muistin alueelle, perinteistä muistitilaa, jota ne muuten olisivat voineet käyttää, voi käyttää sovellusohjelmisto. TSR:t vaativat kuitenkin tyypillisesti enemmän muistia silloin, kun niitä alustetaan, kuin silloin, kun ne pelkästään jäävät paikalleen. Joskus, kuten esimerkiksi silloin kun EMS on asennettu ja sen sivunkehys varaa muistitilaa varatun muistin

alueella, ei ehkä ole käytettävissä riittävästi varatun muistin tilaa, jotta tiettyjen TSR:ien alustaminen olisi mahdollista. Tämä tekniikan tason menetelmä voittaa tämän ongelman lainaamalla tilaa varatun muistin tilasta, joka on allokoitu EMS-sivunkehykselle. Sen jälkeen kun TSR suorittaa loppuun alustuksensa eikä tilaa enää tarvita, tila palautetaan EMS:lle.

Tällä tekniikan tason menetelmällä on myös haittapuolia. Ensinnäkin menetelmä on sovellettavissa ainoastaan TSR:ille. Joissakin järjestelmissä ei ehkä ole lainkaan TSR:iä tai on ainoastaan muutamia TSR:iä, ja ne hyötyvät vähän tästä menetelmästä. Tästä menetelmästä saadun hyödyn määrä vaihtelee myös riippuen erityisestä TSR:stä. Jos TSR ei vaadi lisämuistia alustamista varten, tästä menetelmästä voidaan saada vähän hyötyä.

Kahdeksas tekniikan tason menetelmä käsittää LOAD-ALL-komennon käytön avotilaohjelmien ajamiseksi 80286-mikroprosessorin suojatilassa. Sen tarkoituksena on emuloida EMS:ää, lukuun ottamatta sitä tosiasiaa, että 80286-mikroprosessorilla ei ole luontaista muistin karttasijoittelukykyä avotilassa. Kun avotilaohjelma pyrkii lataamaan segmenttirekistereitä, tapahtuu yleinen suojauspoikkeama (general protection exception). Tämä menetelmä käyttää yleisen suojauspoikkeaman käsittelijää (GPE-käsittelijää) segmenttirekisterin muuttamiseksi valitsimeksi, niin että odottamattomasta prosessoritilasta tehdään ohjelmalle tuntumaton. Näiden ominaisuuksien aikaansaamiseksi 80286-mikroprosessorissa keksintö nojautuu LOADALL-komentoon.

LOADALL-komento on 80286-mikroprosessorin dokumentoitamaton piirre, jolla ei ole tukea. 8088-, 8086-, 80386- tai 80486-mikroprosessorit eivät tue sitä. Se toteutettiin ilmeisesti 80286-mikroprosessorissa ainoastaan mikroprosessorin valmistamisen ja testaamisen apuvälineenä. Siten ei ole mitään takeita siitä, että edes tulevaisuudessa valmistetut 80286-mikroprosessorit tukevat LOADALL-käskyä.

LOADALL-käskyä ei ole koskaan tarkoitettu käytettäväksi järjestelmissä eikä sovellusohjelmistossa.

Tekniikan tason kahdeksannen menetelmän käyttämiseksi täytyy suorittaa seuraava proseduuuri. Ensiksi keskeytykset täytyy asettaa estoon. Toiseksi täytyy säästää 102 tavua alkaen paikasta 00800H. Kolmanneksi täytyy säästää kaikki rekisterit. Seuraavaksi halutut rekisteriarvot mukaan lukien määrittelijävälimuistin kantaosoitteet, jotka määrittelevät saannin kohteena olevan ulkoisen muistin alueen, täytyy ladata paikasta 00800H alkaviin muistin 102 tavuun. Tämän jälkeen täytyy suorittaa LOADALL-toimituskoodi (0FH05H). Seuraavaksi täytyy suorittaa luku jatko-muistista tai kirjoitus siihen käyttäen ainoastaan osoitesiiirrosta ja muuttamatta segmentin kantaosoitetta. Seuraavaksi tallennetut arvot kaikille rekistereille täytyy hakea ja ladata paikasta 00800H alkaviin muistin 102 tavuun. Seuraavaksi täytyy suorittaa LOADALL-toimituskoodi (0FH05H) vanhan rekisterin ja määrittelijävälimuistin sisällön palauttamiseksi. Sitten muistin paikasta 00800H alkavien 102 tavun sisältö täytyy hakea ja palauttaa. Lopuksi täytyy jälleen antaa sallinta keskeytyksille.

Tekniikan tason kahdeksannella menetelmällä on monia haittapuolia. Ensinnäkin LOADALL-käsky toimii ainoastaan 80286-pohjaisissa tietokonejärjestelmissä, ei 8088-, 8086-, 80386 tai 80486-pohjaisissa tietokonejärjestelmissä. LOADALL-käsky on dokumentoimaton, eikä sillä ole tukea, eikä ole mitään takeita siitä, että LOADALL-käsky toimii tulevaisuudessa tuotetuissa 80286-mikroprosessoreissa. Yleisesti ottaen ei voida hyväksyä käytäntöä, että luotetaan laitteistopiirteisiin, joita ei ole dokumentoitu ja joilla ei ole tukea. Lisäksi LOADALL-käsky vaatii datan manipulointia absoluuttisessa muistiosoitteessa 00800H. Ellei suoriteta toimenpiteitä sen varmistamiseksi, että tällä muistin alueella oleva data säilytetään ennen LOADALL-käskyn suorittamista, järjestelmäohjelmisto voi vahin-

goittua. Tämä menetelmä tuottaa myös yleiset suojauspoikkeamat kaikenlaisien järjestelmäoperaatioiden aikana. Näitä poikkeamia täytyy palvella, mikä kuluttaa CPU:n aikaa ja siten olennaisesti heikentää järjestelmän toimintaa

5 enimmän osan ajasta.

Vielä eräs tekniikan tason menetelmä liittyy 16K:n RAM-korttiin, jota käytetään Apple II-tietokoneessa. Apple II -tietokone perustuu 6502-mikroprosessoriin. 6502-mikroprosessorissa on 16-bittinen osoiteväylä, joka tukee 64K:n

10 osoiteavaruutta. Johtuen RAM-laitteiden korkeasta hinnasta siihen aikaan, kun Apple II tuotiin markkinoille, kuitenkin ainoastaan alemmat 48K:n osoiteavaruudesta tarkoitettiin RAM-käyttöön. 48K:n RAM-muistin yläpuolella oli 4K:n alue, joka oli omistettu I/O-muistiksi. Osoiteavaruuden

15 yläpäässä sijaitseva jäljelle jäävä 12K:n alue tarkoitettiin ROM-käyttöön.

Kun sovellusohjelmisto alkoi vaatia yhä kasvavat määrät RAM-muistia, tarvittiin menetelmä RAM:in laajentamiseksi yli 48K rajan. RAM:in määrän kasvattamiseksi tuotiin markkinoille 16K:n RAM-kortti. 16K:n RAM-kortti sisälsi 16K RAM-muistia, joka voitiin sijoittaa kartalle

20 64K:n avaruuden ylimmiksi 16K:n tavuksi korvaamaan 4K:n I/O-muistia ja 12K:n ROM-muistia. 16K:n lisä-RAM:in käytön mahdollistamiseksi I/O-muisti ja ROM oli ensiksi poistettava ylimmästä 16K:n osoitetilasta. Tämä poistaminen voitiin suorittaa muuttamalla muistilaitteisiin liittyvää osoitteendekoodauslogiikkaa. Sitten 16K:n RAM-muistia voitiin sijoittaa kartalla 16K:n osoitetilaan antamalla sallinta sen osoitteendekoodauslogiikalle.

25

16K RAM-kortin toiminnan ollessa sallittuna 64K:n osoiteavaruus täytettiin 64K:n RAM-muistilla. 4K:n I/O-muistiin tai 12K:n ROM-muistiin ei ollut pääsyä tuona aikana. Koska I/O-muisti ja ROM eivät olleet käytettävissä, sovellusohjelmiston oli ylläpidettävä tietokonejärjestelmän täysi ohjaus, kun 16K RAM-kortin toiminta oli sallit-

30

35

tuna, ja varmistettava, että I/O-muistin ja ROM-muistin toiminta jälleen sallittiin ennen ohjauksen luovuttamista. Sovellusohjelmisto ei myöskään voinut päästä I/O-laitteisiin tai ROM-rutiineihin, kun 16K RAM-kortin toiminta oli sallittuna. Jos sovellusohjelmisto luovutti ohjauksen tai yritti päästä I/O-laitteisiin tai ROM-rutiineihin, kun 16K RAM-kortin toiminta oli sallittuna, järjestelmä luhistui, koska halutut I/O-laitteet ja järjestelmä-ROM eivät olleet tietokonejärjestelmän osoiteavaruudessa tuohon aikaan. Koska menetelmä vaati tällaista huolellista ohjausta ja valvontaa sovellusohjelmiston toimesta, se ei varmasti ollut sovellusohjelmiston toiminnalle tuntumaton. Sovellusohjelmiston täytyi olla kirjoitettu erityisesti tunnistamaan 16K RAM-kortin olemassaolo ja käyttämään sen piirteitä huolellisesti.

Keksintö ylittää tämän tekniikan tason menetelmän rajoitukset tarjoamalla menetelmän ROM-osoitettilan käyttämiseksi uudelleen tavalla, joka on sovellusohjelmistolle tuntumaton. Ei ole minkäänlaista tarvetta sille, että sovellusohjelmisto kirjoitettaisiin erityisesti tukemaan keksintöä tai edes olemaan tietoinen sen olemassaolosta ja toiminnasta. Keksintö antaa siten paljon käyttökelpoisemman ja monipuolisemman tavan käyttää uudelleen ROM-osoitettilaa kuin tekniikan tason menetelmät.

25 Keksintö

Keksintö tarjoaa käyttöön lisämuistitilaa, jota voidaan käyttää tukemaan RAM:ia, allokoimalla uudelleen ROM-muistiosoitteet varatulla muistialueella välillä 640K ja 1 megatavu. Alkuperäisiä muistiosoitteita voidaan siten käyttää osoittamaan sovelluksia, TSR:iä jne., jotka on tallennettu RAM-muistiin.

Esimerkinomaisesti kuvataan parhaana pidetty suoritusmuoto sovellettuna tietokonejärjestelmään, joka perustuu 8088/8086-mikroprosessoriperheeseen (joka sisältää myös 80286-, 80386- ja 80486-mikroprosessorit mutta ei

rajoitu niihin) ja toimii MS-DOS- tai PC-DOS-käyttöjärjestelmän alaisuudessa. Keksinnön soveltaminen ei kuitenkaan rajoitu tällaiseen tietokonejärjestelmään. Keksintö ei vaadi mitään erityistä näyttösovitinkonfiguraatiota vaan toimii kaikilla näyttösovittimilla. Lisäksi keksintö ei tue pelkästään joidenkin erityisten ROM:ien erityisiä versioita vaan on yleisesti sovellettavissa useimmille ROM-muisteille. Lisäksi keksintö ei rajoitu käytettäväksi jonkin erityisen sovellusohjelmiston yhteydessä, vaan sitä voidaan käyttää tuntumattomasti useimmille sovellusohjelmitoille. Keksinnöllä vältetään myös koko järjestelmän suorituskyvyn olennainen heikkeneminen. Keksintö on siten suuri parannus suhteessa tekniikan tason menetelmiin lisätä käytettävissä olevan muistin määrää tietokonejärjestelmissä.

Keksinnön soveltamista tyypilliseen MS-DOS-tietokonejärjestelmään on kuvattu kuviossa 13. Järjestelmä-ROM 1301 ja näyttö-ROM 1302 on sijoitettu uudelleen jatkomuistiin 1303, ja niiden aikasemmin varaamalta tilalta on poistettu allokointi. Käyttämättömän varatun muistin 1304 määrää kasvatetaan siten käyttämättömäksi varatuksi muistiksi 1305. Keksintö sallii pääsyn järjestelmä-ROM:iin 1306 ja näyttö-ROM:iin 1307 joko suojatilassa tai karttasijoittamisella takaisin varattuun muistiin.

Keksintö käyttää hyväksi segmentoitua osoittamista sekä tietokonejärjestelmissä käytettyjen mikroprosessorien "avotilaa" ja "suojatilaa". Segmentoitu osoittaminen ja avo- ja suojatila kuvataan alla yksityiskohtaisesti.

Segmentoitu osoittaminen on kuvattu kuviossa 3. Segmentoidussa osoittamisessa osoite määritellään kahdessa osassa. Ensimmäiseen osaan viitataan segmenttinä 301, ja se on 16-bittinen binaariarvo. Toiseen osaan viitataan siirroksena 302, ja se on myös 16-bittinen binaariarvo. Saannin kohteena olevan erityisen osoitteen 305 määrittämiseksi fyysisessä muistissa 306 mikroprosessori suorittaa

ensiksi segmenttirekisterin sisällön kertomisen 303 arvol-
la 10H, mikä vastaa sen siirtämistä 4 bittiä vasemmalle,
ja sitten suorittaa tuon arvon laskemisen yhteen 304 siir-
rosrekisterin sisällön kanssa. Siirtämällä segmenttirekis-
5 teriä ennen sen summaamista siirrosrekisteriin kehitetään
20 bittiä leveä osoite. CPU:n 20-bittisen osoiteväylän
kautta saatavissa olevan muistin määrä on 2^{20} , joka on yhtä
kuin 1024K tai 1M. Tätä osoitusmenetelmää käytetään myös
80286-, 80386- ja 80486-mikroprosessoreissa, kun niitä
10 käytetään "avotilassa".

80286-, 80386- ja 80486-mikroprosessoreita voidaan
käyttää joko "avotilassa" tai "suojatilassa". Suojatilalla
on useita etuja, joita ei ole avotilassa. Ensinnäkin
80286:lla suojatila sallii pääsyyn 16 megatavuun muistia,
15 mikä on 16 kertaa avotilassa osoitettavissa olevan muistin
määrä. 80386:lla ja 80486:lla suojatila kasvattaa osoitus-
rajoja vielä enemmän. Suojatilassa 80386 ja 80486 voivat
osoittaa 4096M tai 4G, joka on jokseenkin 4 miljardia
muistisanaa.

20 Keksintö voidaan toteuttaa muodossa, jota kutsutaan
"suojatilasuoritusmuodoksi", tai muodossa, jota kutsutaan
"karttasijoitusmuodoksi".

Suojatila

Suojatilassa osoittaminen on kuvattu kuviossa 4.
25 16-bittisen segmenttirekisterin sijasta suojatila käyttää
16-bittistä valitsinta 401. Valitsin osoittaa tulokohtaa
403 määrittelijätaulukossa 404. Määrittelijätaulukot si-
sältävät määrittelijät, jotka spesifioivat kantaosoitteen.
80286:n määrittelijät antavat 24-bittisen kantaosoitteen,
30 kun taas 80386:n ja 80486:n määrittelijät antavat 32-bit-
tisen kantaosoitteen. Nämä kantaosoitteet suorittavat sa-
man tehtävän kuin 8088/8086:n segmentit mutta ovat moni-
puolisempia. Spesifioidun osoitteen 405, johon on päästä-
vä, määrittämiseksi muistissa 406 lisätään määrittelijä-

taulukosta 404 saatuun kantaosoitteeseen siirros 402, joka on samanlainen kuin 8088/8086:ssa käytetty.

80386:ssa ja 80486:ssa on lisäominaisuutena "virtuaalinen 86-tila" ("V86-tila"). Virtuaalitilassa mikroprosessorin laskema osoite ei ehkä ole osoiteväylälle annettu todellinen osoite. Kun 80386 tai 80486 laskee osoitteen virtuaalitilassa, se laskee ainoastaan 32-bittisen lineaarisen osoitteen. Lineaarinen osoite viedään sitten mikroprosessorin sivutusmekanismille, joka suorittaa lisäksi laskutoimitukset lopullisen fyysisessä muistissa olevan osoitteen määrittämiseksi. Sivutusmekanismi sallii muistin karttakuvaamisen muotoon, joka eroaa todellisen fyysisen muistin karttasijoittelusta. Jakamalla fyysisen muistin sivuihin virtuaalitila sallii, että fyysisen muistin kullakin 4K:n lohkolla on oma virtuaalinen osoitteensa. Alla esitetyssä kuvauksessa termi "avotila" voidaan vaihtaa termin "virtuaalitila" kanssa. Parhaana pidetyssä suoritustavassa keksintö käyttää virtuaalitilaa.

Keksinnön suojatilasuoritustavoitto mahdollistaa varatulla muistialueella 503 olevan muistitilan, joka on alunperin allokoitu järjestelmä- ja näyttö-ROM:ille 504 ja 505, uudelleenallokoinnin. Lukemalla ROM:it suojatilassa keksintö mahdollistaa ROM-osoitteen käyttöön muihin tarkoituksiin avotilassa.

Keksintö toteutetaan alustamalla järjestelmäkonfiguraatio, allokoimalla uudelleen alkuperäinen ROM-osoitte-tila sen tekemiseksi muihin tarkoituksiin käytettäväksi, sieppaamalla ROM-haut, niin että ne voidaan suunnata uudelleenallokoituun ROM-osoitte-tilaan, sekä käsittelemällä ROM-haut siten, että ROM-haut voidaan suorittaa loppuun, ja järjestelmä voi jatkaa suoritusta ROM-haun jälkeen.

Keksinnön tietokonejärjestelmän alustus on kuvattu kuvion 14 vuokaavion yhteydessä. Alustusrutiini alkaa vaiheesta 1401. Vaiheessa 1402 ROM:it, jotka on määrää siirtää, paikannetaan alkuperäiseen osoitteen tilaansa kuvion 5

varatun muistin alueella 503. Vaiheessa 1403 keskeytyskäsittelijät, joilla on pääsy ROM-koodiin, identifioidaan.

Vaiheessa 1404 identifioidaan ROM-datataulukkojen osoitteet ja pituudet. Vaiheessa 1405 identifioidaan sopiva allokoinen suojatila osoitetila uudelleensijoitettujen ROM:ien käytettäväksi. Tämän osoitetilan täytyy olla aukoton ja kooltaan sopiva tukemaan täysin ROM-osoitusta. Tämä osoitetila on se, jossa uudelleensijoitetut ROM:it tulevat toimimaan.

10 Vaiheessa 1406 identifioitu osoitetila jatkamuistissa allokoidaan ROM:eille, niin että estetään osoitetilan ristiriitainen käyttö muihin tarkoituksiin. ROM:it sijoitetaan kartalla allokoituun muistiosoitettiin.

15 Valinnaisessa vaiheessa 1407 globaalit ja paikalliset määrittelijätaulukot (GDT/LDT) esiallokoidaan uudelle osoitetilalle, johon ROM:it on sijoitettu kartalla.

Vaiheessa 1408 perinteinen muisti allokoidaan pitämään kopio ROM-datataulukoista. Vaiheessa 1409 ROM-datataulukot kopioidaan perinteiseen muistiin. Vaiheessa 1410 osoitetila perinteisessä muistissa allokoidaan keskeytyskäsittelijärutiinille. Vaiheessa 1411 keskeytyskäsittelijärutiini asennetaan sitten allokoituun osoitetilaan. Vaiheessa 1412 osoittimet ROM-datataulukoihin suunnataan uudelleen perinteisessä muistissa olevaan kopioon.

25 Valinnaisessa vaiheessa 1413 ROM-osoitteet katetaan jatkumuistilla. Vaiheessa 1414 ohjaus palautetaan käyttöjärjestelmälle.

Jos kaikkia viittauksia ROM-datataulukoihin ei voida löytää tai jos on toivottavaa, että vältetään kaikki mahdollisuudet, että niitä kaikkia ei voida löytää, voidaan toteuttaa eräs edellä esitetyn proseduurin variaatio. Sen jälkeen kun kaikki ROM-datataulukot on paikannettu joko niiden sijaintia koskevan julkaistun informaation perusteella tai ROM-muistien sisältöä analysoimalla, osa alkuperäisestä ROM-osoitetilasta, jonka ne varaavat, voi-

30

35

daan merkitä niille varatuksi. Sitten suojatilan piirteitä voidaan käyttää palauttamaan ROM-datataulukot niiden alkuperäisille paikoille alkuperäisessä ROM-osoitetilassa. ROM-datataulukot sijoitetaan kartalla alkuperäisiin paikkoihinsa alkuperäisessä ROM-osoitetilassa. Mikroprosessori asetetaan suojatilan rengas nollaan. Sekä globaalit että paikalliset määrittelijätaulukot (GDT/LDT) asetetaan ROM-datataulukkojen varaamalle osalle alkuperäisestä ROM-osoitetilasta. Prosessori saatetaan poistumaan suojatilasta ja palaamaan avotilaan. Alkuperäinen ROM-osoitetila lukuun ottamatta ROM-datataulukoiden varaamia osia jätetään allokoimatta, jotta sallitaan sen käyttö muihin avotilatarkoituksiin. Sopiva allokoimaton jatko-RAM täytyy löytyä täyttämään varatussa muistissa 503 oleva alkuperäinen ROM-osoitetila, jolta on poistettu allokointi. Tämä RAM allokoidaan sitten ja sijoitetaan kartalla alkuperäiseen ROM-osoitetilaan varatussa muistissa.

Kun kaikki tarpeelliset vaiheet on suoritettu tietokonejärjestelmän alustamiseksi käytettäväksi keksinnön suojatilasuoritusmuodossa, RAM, joka on sijoitettu kartalla alkuperäiseen ROM-osoitetilaan varatussa muistissa 503, voidaan käyttää mihin tahansa lukuisista eri tarkoituksista, mukaan lukien TSR:ien, laiteohjainten tai verkkorajapintaohjelmien sijoittaminen uudelleen. Uusi RAM pysyy kartalla sijoitettuna alkuperäiseen ROM-osoitetilaan tietokonejärjestelmän toiminnan kaikkien vaiheiden aikana mukaan lukien ROM-kutsujen ajat.

Järjestelmän uudelleenkonfiguroinnin jälkeen ROM-hakusieppaaja on mukana, jotta mahdollistetaan varatussa muistissa 503 olevan alkuperäisen ROM-osoitetilan uudelleenallokointi. Kun mikä tahansa ohjelmisto järjestelmässä pyrkii pääsemään ROM:iin, ROM-hakusieppaajan täytyy siepata pääsy-yritys ja aktivoida ROM-hakukäsittelijä varmistamaan, että ROM-koodiin tai -dataan voidaan päästä. Koska useimpiin ROM-rutiineihin päästään ohjelmistokeskeytysten

kautta, helpoin tapa siepata ROM-haut on modifioida edellä kuvatulla tavalla keskeytystaulukkoa. Koska jokin sovellusohjelmisto voi yrittää päästä ROM:iin suoraan käyttämättä keskeytyksiä, on toivottavaa tutkia sovellusohjelmistosta viittaukset ROM-osoitteisiin ja suunnata nämä viittaukset uudelleen, niin että ne hakevat uuden ROM-vedoksen jatkomuistista.

Jos käytetään edellä kuvattua vaihtoehtoista proseduuria säilyttää ROM-datataulukot niiden alkuperäisillä paikoilla, voidaan tietenkin sallia kaikki ROM-haut, jotka yrittävät päästä ROM-datataulukoihin, eikä niitä tarvitse siepata ja suunnata uudelleen.

ROM-hakukäsittelijä tarjoaa käyttöön tavan päästä uuteen ROM-vedokseen jatkomuistissa 511, sisältää ROM-suorituksen ja palauttaa tietokonejärjestelmän tilaan, jossa se voi jatkaa sovellusohjelmiston suoritusta sen jälkeen, kun ROM-haku on suoritettu loppuun.

ROM-hakukäsittelijän toimintaa kuvaava vuokaavio on esitetty kuviossa 15. Prosessi alkaa vaiheesta 1501. Vaiheessa 1502 ROM-hakukäsittelijä tietyille keskeytyspalvelurutiinille, jota haetaan, saa CPU:n siirtymään suojatilaan suojaustasolla, joka tunnetaan rengas nollana. Valinnaisessa vaiheessa 1503 palvelu keskeytykselle suoritetaan ilman, että mennään ROM:iin, ja ohjaus palautetaan kutsujalle. Jos valinnaista vaihetta ei suoriteta, jatketaan seuraavasti.

Vaiheessa 1504 oton avotilasegmenttirekisterit asetetaan heijastamaan uusia suojatilavalitsimia, jos se on tarpeen. Vaiheessa 1505 osoite ROM:ille paluuta varten asetetaan suojatilapinoon.

Vaiheessa 1506 vaihdetaan renkaan kolme suojatilan CS:IP osoittamaan uudelleensijoitetun ROM:in tulokohtaa. Vaiheessa 1507 asetetaan sopiva IOPL. Vaiheessa 1508 siirrytään renkaan kolme suojatilaan uudelleensijoitetun ROM:in tulokohdassa.

Vaiheessa 1509 ROM:in annetaan tehdä suoritus ja käsitellä mahdolliset virheet, joita voi esiintyä. Vaiheessa 1510 ROM palauttaa ohjauksen vaiheessa 1505 asetettuun osoitteeseen. Vaiheessa 1511 siirrytään takaisin suojatilaan siivousrutiiniin.

Vaiheessa 1512 muutetaan avotilan CS:IP osoittamaan kutsujan paluusoitteeseen. Jos tarpeen, vaiheessa 1513 muutetaan paluusoittimia osoittamaan uudelleensuunnattuun ROM-dataan. Vaiheessa 1514 siirrytään takaisin avotilaan kutsujan paluusoitteeseen.

Karttasijoitus-suoritusmuoto

Keksinnön karttasijoitus-suoritusmuoto mahdollistaa myös alunperin ROM:eille allokoitun osoitetilan varatussa muistissa 503 uudelleenallokoinnin muuhun käyttöön. Keksinnön karttasijoitus-suoritusmuoto ei kuitenkaan vaadi suojatilan käyttöä, ja se on siksi yhteensopiva 8088- ja 8086-mikroprosessoreiden kanssa, jotka eivät tue suojatilatointia. Vaikka keksinnön karttasijoitus-suoritusmuoto ei vaadi 80386- tai 80486-mikroprosessoreiden suojatilaa, se vaatii kyvyn sijoitella muistiavaruus kartalle uudelleen. 80386:lla ja 80486:lla tämän kyvyn antaa V86-virtuaalitila. 8088-, 8086- ja 80286-mikroprosessoreissa tämän kyvyn voi antaa joko laajennusmuistin määrittelykortti (EMS-kortti) tai jokin muu muistinhallintalaitteisto.

Keksinnön alustusrutiinin karttasijoitus-suoritusmuotoa kuvaava vuokaavio on esitetty kuviossa 16. Prosessi alkaa vaiheesta 1601. Vaiheessa 1602 ROM:it, joiden osoitetila on määrä allokoita uudelleen, paikallistetaan.

Vaiheessa 1603 identifioidaan keskeytyskäsitteijät, joilla on pääsy ROM-koodiin. Vaiheessa 1604 ROM-datataulukoiden osoitteet ja pituudet identifioidaan.

Vaiheessa 1605 allokoidaan avotilan osoitetila EMS-sivunkehykselle, jossa ROM:it tullaan suorittamaan. Valinnaisessa vaiheessa 1606 allokoidaan jatkomuisti, jota on määrä käyttää kattamaan ROM-osoitteet.

Datataulukot ROM:eista kopioidaan perinteiseen muistiin vaiheessa 1607. Vaiheessa 1608 keskeytyskäsitte-
 lijäsieppaajat asetetaan paikalleen. Vaiheessa 1609 osoit-
 timet ROM-dataan suunnataan uudelleen osoittamaan datako-
 5 pioihin perinteisessä muistissa. Valinnaisesti vaiheessa
 1610 ROM-osoitteet katetaan jatkomuistilla. Ohjaus palau-
 tetaan käyttöjärjestelmälle vaiheessa 1611.

Jos kaikkia viittauksia ROM-datataulukoihin ei
 voida löytää tai jos on toivottavaa, että vältetään kaikki
 10 mahdollisuudet, että niitä kaikkia ei löydetä, voidaan
 toteuttaa edellä esitetyn proseduurin eräs variaatio. Sen
 jälkeen kun kaikki ROM-datataulukot on paikallistettu joko
 niiden sijaintia koskevan julkaistun informaation perus-
 teella tai ROM-muistien sisältöä analysoimalla, osa alku-
 15 peräisestä ROM-osoitetilasta, jonka ne varaavat, voidaan
 merkitä niille varatuksi. ROM-datataulukot sijoitetaan
 kartalla takaisin alkuperäisiin paikkoihinsa alkuperäises-
 sä ROM-osoitetilassa. Sitten alkuperäinen ROM-osoitetila
 varatussa muistissa, lukuun ottamatta ROM-datataulukkojen
 20 varaamaa tilaa, jätetään allokoimatta, jotta sallitaan sen
 käyttö muihin avotilatarkoituksiin. Sopivaa allokoimatonta
 jatko-RAM-muistia täytyy löytää täyttämään alkuperäinen
 varatussa muistissa oleva ROM-osoitetila, jolta on pois-
 tettu allokointi. Tämä RAM täytyy sitten allokoida. Tämä
 25 RAM täytyy sijoittaa kartalla allokoimattomaan ROM-osoi-
 tilaan varatussa muistissa 503.

Kun kuvion 16 alustusrutiini on suoritettu loppuun,
 tietokonetta voidaan käyttää normaalisti, ja muistitila,
 jonka alunperin varasivat ROM:it, voidaan allokoida muuhun
 30 käyttöön, mukaan lukien TSR:t, laiteohjaimet ja verkkora-
 japintaohjelmat.

Karttasijoitus suoritusmuodon ROM-hakusieppaajan
 täytyy siepata kaikki käskyt, jotka viittaavat ROM:eihin,
 jotta estetään suorituksen joutuminen harhaan. Koska
 35 useimmat sovellusohjelmistot pääsevät ROM-koodiin ohjel-

mistokeskeytysten kautta, keskeytystaulukon modifiointi mahdollistaa ROM-koodiin kohdistuvien kutsujen sieppaamisen ja suuntaamisen uudelleen. On toivottavaa myös suunnata uudelleen viittaukset ROM-datataulukkoihin, ja siten ohjelmisto täytyy tutkia kaikkien ROM-datataulukkoihin kohdistuvien hakuyritysten paikallistamiseksi, ja muutokset ROM-datataulukko-osoittimiin täytyy tehdä sen varmistamiseksi, että haut suunnataan uudelleen oikein. Jos käytetään edellä kuvattua vaihtoehtoista proseduuria pitää ROM-datataulukot alkuperäisillä paikoillaan, kaikki ROM-haut, jotka yrittävät päästä ROM-datataulukkoihin, voidaan sallia, eikä niitä tarvitse siepata ja suunnata uudelleen. Kun ROM-hakusieppaaja sieppaa yritetyn ROM-haun, se siirtää suorituksen ROM-hakukäsittelijälle.

15 Keksinnön ROM-hakukäsittelijän karttasijoitussuoritusmuoto on kuvattu kuvion 17 vuokaaviossa. Vuokaavio aloittaa toiminnan vaiheesta 1701.

 Vaiheessa 1702 siirrytään suojatilaan erityistä haettavaa keskeytyspalvelurutiinia varten olevaan käsittelijään. Valinnaisessa vaiheessa 1703 palvelu suoritetaan menemättä ROM:iin, ja järjestelmä palaa silloin kutsujaan. Jos valinnaista vaihetta 1703 ei suoriteta, prosessi jatkuu seuraavasti.

25 Vaiheessa 1704 säästetään vallitseva sivunkehysten karttasijoittelu, valinnaisesti kutsujan avotilapinoon. Vaiheessa 1705 asetetaan ROM:ille osoite paluuta varten avotilapinoon. Vaiheessa 1706 muutetaan avotilan CS:IP osoittamaan uudelleensijoitetun ROM:in tulokohtaan. Vaiheessa 1707 ROM sijoitetaan kartalla sivunkehukseen. Vaiheessa 1708 siirrytään avotilaan uudelleensijoitetun ROM:in tulokohdassa. ROM:in sallitaan suorittaa tehtävänsä vaiheessa 1709.

 Vaiheessa 1710 ohjaus palaa vaiheessa 1705 asetettuun osoitteeseen. Järjestelmä siirtyy takaisin suojati-

laan siivousrutiiniin. Vaiheessa 1712 muutetaan avotilan CS:IP osoittamaan kutsujan paluusoitteeseen.

Jos tarpeen, muutetaan paluusoittimet osoittamaan uudelleensuunnattuun ROM-dataan vaiheessa 1713. Alkuperäinen sivunkehysten karttasijoitus palautetaan vaiheessa 1714, valinnaisesti kutsujan avotilapinoon säästetystä datasta. Vaiheessa 1715 järjestelmä siirtyy takaisin avotilaan kutsujan paluusoitteeseen.

Vaikka keksinnön toteutus käyttäen CPU:n suojatilaa tarjoaa lukuisia etuja, ROM:eja käytetään normaalisti avotilassa (tai V86-tilassa) eikä suojatilassa. Yhteensopivuuden säilyttämiseksi järjestelmäohjelmiston ja sovellusohjelmiston kanssa, jotka odottavat ROM-muistien olevan saatavilla avotilassa, täytyy suorittaa vaiheita, jotka simuloivat avotilatoimintaa ROM:eille, joihin pääsy tapahtuu suojatilassa.

Keksintö voidaan toteuttaa sisältämään menetelmä yritysten ladata segmenttirekisteri pyydystämiseksi. Yrityksen ladata segmenttirekisteri, kun CPU on suojatilassa, ilmaisee CPU, ja se tuottaa automaattisesti kutsun yleisen suojauspoikkeaman käsittelijälle (GPE-käsittelijälle, engl.: General Protection Exception Handler). GPE-käsittelijä palauttaa käskyn paikan, joka loukkasi suojatilan sääntöjä.

Tämän tapahtuman kompensoimiseksi GPE-käsittelijän määrittelemässä paikassa sijaitseva käsky tutkitaan ja dekodataan. Dekoodaus voidaan suorittaa hakemalla käskyn toimituskoodi hakutaulukosta, joka sisältää kaikkien käskysten toimituskoodit. Analysoimalla käskyä voidaan määrittää käskyn tarkoitettu vaikutus avotilassa, ja tarkoitettua vaikutusta voidaan simuloida suojatilassa. Koska tarkoitettu vaikutus on saatu, koodi, joka tuotti virheen, voidaan saada toimimaan oikein, ja suoritus voidaan siirtää takaisin tuolle koodille, jotta sallitaan sen suorittaminen loppuun.

Sen varmistamiseksi, että keksintö toimii oikein, I/O-käskyt voidaan pyydystää muuttamalla I/O-etuoikeustaso (IOPL) suuremman etuoikeuden tasolle (jota edustaa alhaisempi etuoikeustason numero), niin että kaikkien IOPL-herkkien I/O-käskyjen suorittaminen tuottaa yleisen suojauspoikkeamavirheen. Tuloksena oleva virhe voidaan käsitellä sen varmistamiseksi, että se ei ole ristiriidassa keksinnön toiminnan kanssa.

Vaikka keskeytyslippu tietyillä mikroprosessoreilla toimii tavalla, joka vaatii hiukan huomiota, keskeytyslippu voi toimia eri tavoin, kun I/O-etuoikeudet kielletään, kuten silloin, kun pyydytetään I/O-käskyt. Tiedyt mikroprosessorikäskyt, jotka liittyvät keskeytyslippuun, ovat I/O-etuoikeustasoherkkiä (IOPL-herkkiä). Järjestelmissä, jotka perustuvat 80286- tai 80386-mikroprosessoreihin, yritys suorittaa CLI-käsky (nollaa keskeytyksensallintalippu) tai STI-käsky (asettaa keskeytyksensallintalippu), kun I/O-etuoikeudet kielletään, tuottaa yleisen suojauspoikkeamavirheen.

Keksintö sisältää myös menetelmän, jonka tarkoituksena on voittaa vaikeudet tarkkailla ohjelmiston yrityksiä muuttaa keskeytyslippu tai emuloida oikeaa keskeytyslippukäyttäytymistä. Vaikeudet syntyvät POPF-käskyn ja IRET-käskyn (keskeytyksen paluu) epäasianmukaisesta toiminnasta suojatilassa. POPF-käskyn oletetaan poimivan ylimmän sanan pinosta ja sijoittavan sen sisällön CPU:n lippurekisteriin. IRET-käskyn oletetaan poimivan ylimmät kolme sanaa pinosta ja sijoittavan kolmannen sanan sisällön CPU:n lippurekisteriin. Suojatilassa POPF-käsky ja IRET-käsky eivät kumpikaan muuta keskeytyslippua eivätkä kehitä yleistä suojauspoikkeamaa. Siten GPE-käsittelijän käyttö emuloimaan keskeytyslipun avotilakäyttäytymistä, kun tietokone on suojatilassa, ei tuota täydellistä emulointia eikä tuota oikeaa toimintaa.

Aikaisemmin kukaan ei ole pystynyt tuottamaan tyydyttävää avotilaemulointia, kun ollaan suojatilassa, johtuen kyvyttömyydestä pyydystää POPF-käskyt. DOS Protected Mode Interface Specification, Version 1.0 toteaa: "...(A)siakkaat eivät voi käyttää IRET(D)- tai POPF-käskyä keskeytyslipun muuttamiseen, koska nämä käskyt ha-
 5 kevat fyysisen keskeytyslipun ja koska CPU jättää ne huomiotta johtuen asiakkaan etuoikeustasosta".

Eräänä keksinnön piirteenä on saada POPF- ja IRET-käskyt kehittämään yleinen suojauspoikkeama. Keksintö antaa avotilakäyttäytymisen oikean emuloinnin, kun ollaan suojatilassa, pyydystämällä POPF-käskyn tai IRET-käskyn käyttäen vaihtoehtoista menetelmää sen sijaan, että luote-
 10 taan yksinomaan GPE-käsittelijään.

Koska POPF-käskyä tai IRET-käskyä käytetään tavallisesti ennustettavan koodisekvenssin osana ohjelmassa, keksintö käyttää hyväksi tätä säännönmukaisuutta POPF-käskyn pyydystämiseksi.
 15

Normaalisti ohjelma säästää lippurekisterin sisällön käyttäen PUSHF-käskyä, lukitsee keskeytykset käyttäen CLI- tai STI-käskyä, suorittaa sekalaisen koodin, kun keskeytykset on lukittu, ja palauttaa sitten lippurekisterin alkuperäisen sisällön käyttäen POPF-käskyä.
 20

POPF-käskyn pyydystämiseksi pinoraja asetetaan yhtäsuureksi kuin vallitseva pino-osoitin, milloin tahansa esiintyy CLI- tai STI-käskyn suorittamisesta tuloksena oleva yleinen suojauspoikkeama. Tällaisella pinorajalla POPF- tai IRET-käskyn suorittaminen saa aikaan pinorajan ylittämisen ja kehittää sillä tavoin yleisen suojauspoikkeaman.
 25
 30

Keksintö ei rajoitu tietokonejärjestelmiin, joilla on kuviossa 5 esitetyllä tavalla karttana esitetyt muistiosoitteet. Esimerkiksi ROM:eille varatut muistiosoitteet voivat olla eri paikoissa. Esimerkiksi sellaisessa käyt-

töjärjestelmässä kuin PS/2 muistiosoitteet ROM:ejä varten voidaan varata sellaisista paikoista kuten 0E000.

5 Vaikka keksintö käyttää hyväksi suojatilaan karttasijoitusmenetelmässä, tätä ei vaadita. On mahdollista käyttää jotain muuta muistinhallintajärjestelmää, joka ei vaadi, että siirrytään suojatilaan karttasijoitusmenetelmää varten.

10 Siten on aikaansaatu menetelmä, joka antaa enemmän käytettävissä olevaa muistia, joka on helposti saataville, tavalla, joka ei merkittävästi heikennä toimintaa ja joka ei ole CPU-, ROM- tai sovellusspesifinen.

Patenttivaatimukset:

1. Tietokonejärjestelmässä, jossa on prosessori, joka pystyy toimimaan avo- ja suojatilassa, ja jossa on
5 ensimmäinen muistiosoitteiden alue, joka on varattu luku-
muistille (ROM:it), sekä lisämuistiosoitteita, jotka on
osoitettu perinteiseksi muistiksi ja jatkomuistiksi, mene-
telmä sanotun ensimmäisen muistiosoitteiden alueen käyt-
tämiseksi hyväksi sanotun prosessorin toimesta muihin
10 muistilaitteisiin pääsemiseksi, t u n n e t t u siitä,
että se sisältää vaiheet, joissa:

identifioidaan sanottujen ROM:ien osoitteet sano-
tulla ensimmäisellä muistiosoitteiden alueella,

15 identifioidaan keskeytyskäsitteijät sanottuihin
ROM:eihin pääsemiseksi,

identifioidaan minkä tahansa sanottuun ROM:iin tal-
lennetun datataulukon osoitteet,

20 allokoidaan sanotut ROM:it toiselle osoitteiden
alueelle, joka on sanotun ensimmäisen alueen ja sanotun
perinteisen muistin ulkopuolella ja karttasijoitetaan sa-
notut ROM:it sanotulle toiselle muistiosoitteiden
alueelle,

kopioidaan sanottu ROM-datataulukko sanottuun pe-
rinteiseen muistiin,

25 allokoidaan osoitetila sanotussa perinteisessä
muistissa sanotuille keskeytyskäsitteijöille,

kopioidaan sanotut keskeytyskäsitteijät allokoituun
osoitetilaan,

30 varustetaan sanotut keskeytyskäsitteijät valitsi-
milla, jotka osoittavat sanottuun ROM-datataulukoon sano-
tussa perinteisessä muistissa, sekä

katetaan sanotulla ensimmäisellä alueella olevat
sanotut ROM-osoitteet jatkomuistilla.

2. Patenttivaatimuksen 1 mukainen menetelmä, tunnettu siitä, että se sisältää lisäksi vaiheet, joissa:

5 luodaan globaalit ja paikalliset määrittelijätaulukot sanotuille ROM:eille sanotulla toisella osoitteiden alueella.

3. Patenttivaatimuksen 1 mukainen menetelmä, tunnettu siitä, että se sisältää lisäksi vaiheet, joissa:

10 asetetaan sanottu prosessori sanottuun suojatilaan ensimmäisellä suojaustasolla,

asetetaan paluuosoite sanotun suojatilan pinoon, muutetaan koodisegmentti:käskyosoitin (CC:IP) osoittamaan uudelleensijoitettuun tulokohtaan,

15 asetetaan sanottu prosessori sanotun suojatilan toiselle suojaustasolle,

suoritetaan sanottu ROM-saanti, palautetaan ohjaus sanottuun paluuosoitteeseen, muutetaan sanottu CS:IP kutsujan paluuosoitteeksi, 20 tutkitaan sanotun ROM-saannin sanotun suorituksen anto ja modifioidaan sanottua antoa, jos se on tarpeen, sekä

asetetaan sanottu prosessori sanottuun avotilaan.

25 4. Tietokonejärjestelmässä, jossa on prosessori, joka pystyy toimimaan avo- ja suojatilassa, ja jossa on ensimmäinen muistiosoitteiden alue, joka on varattu luku- muistille (ROM:it), menetelmä sanotun ensimmäisen muistiosoitteiden alueen käyttämiseksi hyväksi sanotun prosessorin toimesta muihin muistilaitteisiin pääsemiseksi, 30 tunnettu siitä, että se sisältää vaiheet, joissa:

identifioidaan sanottujen ROM:ien osoitteet sanotulla ensimmäisellä muistiosoitteiden alueella,

identifioidaan keskeytyskäsitteijät sanottuihin ROM:eihin pääsemiseksi,

identifioidaan minkä tahansa sanottuun ROM:iin tallennetun datataulukon osoitteet,

allokoidaan avotilaosoitetila laajennusmuistin määrittelyn (EMS) sivunkehykselle, jossa ROM:it voidaan suorit-
5 rittaa,

kopioidaan sanottu ROM-datataulukko perinteiseen muistiin,

allokoidaan osoitetila sanotussa perinteisessä muistissa sanotuille keskeytyskäsitteilyille,

10 asetetaan sanotut keskeytyskäsitteilyt keskeytystaulukkoon sanotussa allokoitussa osoitetilassa,

muutetaan sanotuissa keskeytyskäsitteilyissä olevia osoittimia sanottuun ROM-datataulukkoon osoittamaan sanotun ROM-datataulukon kopioihin sanotussa perinteisessä
15 muistissa sekä

katetaan sanotulla ensimmäisellä alueella olevat sanotut ROM-osoitteet jatkomuistilla.

5. Patenttivaatimuksen 4 mukainen menetelmä,
tunnetaan siitä, että se sisältää lisäksi vaiheet,
20 joissa:

asetetaan sanottu prosessori sanottuun suojatilaan ensimmäisellä suojaustasolla,

säästetään sanotun EMS-sivunkehyksen vallitseva karttasijoittelu,

25 asetetaan paluusoite sanotun avotilan pinoon,
muutetaan koodisegmentti:käskyosoitin (CS:IP) osoittamaan uudelleensijoitetun ROM:in tulokohtaan,

asetetaan sanottu prosessori sanottuun avotilaan, suoritetaan koodi sanotussa ROM:issa,

30 palautetaan ohjaus sanottuun paluusoitteeseen,
muutetaan sanottu CS:IP kutsujan paluusoitteeksi,
tutkitaan sanotun ROM-saannin sanotun suorituksen antoa ja modifioidaan sanottua antoa, jos se on tarpeen,

35 palautetaan sanotun sivunkehyksen alkuperäinen karttasijoittelu sekä

asetetaan sanottu prosessori sanottuun avotilaan.

6. Menetelmä keskeytyslipun hallitsemiseksi tietokonejärjestelmässä, t u n n e t t u siitä, että se sisältää vaiheet, joissa:

5 ilmaistaan tilaemulaattorilla keskeytyksen käsittelykäskeyn suorittaminen,

asetetaan sanotulla tilaemulaattorilla sanotussa tietokonejärjestelmässä olevan pinon pinoraja,

10 suoritetaan sanotussa tietokonejärjestelmässä pinoon yhteydessä oleva käsky mennä sanottuun pinoon sanotun pinorajan takana,

ilmaistaan sanotussa tietokonejärjestelmässä sanotun menemisen aiheuttama sanotun rajan rikkominen sekä

modifioidaan keskeytyslippua, kun sanottu rikkominen on ilmaistu.

15 7. Patenttivaatimuksen 6 mukainen menetelmä, t u n n e t t u siitä, että sanottu käsky on keskeytykseen yhteydessä oleva käsky.

20 8. Patenttivaatimuksen 6 mukainen menetelmä, t u n n e t t u siitä, että sanottu pinoraja asetetaan pino-osoittimen vallitsevaan arvoon.

9. Patenttivaatimuksen 7 mukainen menetelmä, t u n n e t t u siitä, että sanottu keskeytykseen yhteydessä oleva käsky on keskeytyksen nollauskäsky (CLI).

25 10. Patenttivaatimuksen 7 mukainen menetelmä, t u n n e t t u siitä, että sanottu keskeytykseen yhteydessä oleva käsky on keskeytyksen asetuskäsky (STI).

30 11. Patenttivaatimuksen 6 mukainen menetelmä, t u n n e t t u siitä, että sanottu pinoon yhteydessä oleva käsky on poimi liput -käsky (POPF), joka poimii useita pinoonvientejä sanotusta pinosta.

12. Patenttivaatimuksen 6 mukainen menetelmä, t u n n e t t u siitä, että sanottu pinoon yhteydessä oleva käsky on keskeytyksen paluukäsky (IRET), joka poimii useita pinoonvientejä sanotusta pinosta.

13. Patenttivaatimuksen 6 mukainen menetelmä, tunnettu siitä, että sanotun pinoon yhteydessä olevan käskyn suorittaminen ja sanotun pinorajan sanottu rikkominen tuottavat yleisen suojauspoikkeaman.

Patentkrav:

1. Förfarande i ett datorsystem, vilket har en processor som förmår arbeta på ett reellt och skyddat sätt och är försedd med ett första område minnesadresser som reserverats för läsminnen (ROM) och ytterligare minnesadresser som betecknats som konventionellt minne och expanderbart minne, för utnyttjande av det första området minnesadresser medelst processorn för åtkomst av andra minnesanordningar, k ä n n e t e c k n a t därav, att det omfattar steg, vid vilka man

identifierar adresser för nämnda ROM i det första området minnesadresser;

identifierar avbrottshanterare för åtkomst av nämnda ROM;

identifierar adresser hos någon datatabell som upprevaras i nämnda ROM;

tilldelar nämnda ROM ett andra område av adresser ytterom det första området och det konventionella minnet och kartlägger det andra området minnesadresser i nämnda ROM;

kopierar ROM-datatabellen i det konventionella minnet;

tilldelar adressutrymme i det konventionella minnet för avbrottshanterarna;

kopierar avbrottshanterarna i det tilldelade adressutrymmet;

förser avbrottshanterarna med selektorer vilka pekar på ROM-datatabellen i det konventionella minnet;

överlagrar ROM-adresserna i det första området med det expanderade minnet.

2. Förfarande enligt patentkrav 1, k ä n n e t e c k n a t därav, att det ytterligare omfattar steg, vid vilka man upprättar globala och lokala deskriptorbellor för nämnda ROM i det andra adressområdet.

3. Förfarande enligt patentkrav 1, k ä n n e -
t e c k n a t därav, att det ytterligare inkluderar steg,
vid vilka man

5 placerar processorn i det skyddade arbetssättet vid
en första skyddsnivå;

placerar en återhoppadress på en stack i det skyd-
dade arbetssättet;

ändrar en kodsegment:instruktionsvisare (CS:IP) så
att den pekar på en flyttbar ingångspunkt;

10 placerar processorn vid en andra nivå i det skydda-
de arbetssättet;

utför nämnda ROM-åtkomst;

returnerar styraren till återgångsadressen;

ändrar nämnda CS:IP till en anrops-återgångsadress;

15 granskar uteffekten för utförande av ROM-åtkomsten
och modifierar uteffekten ifall behövt;

placerar processorn i det reella arbetssättet.

4. Förfarande i ett datorsystem, vilket har en pro-
cessor som förmår arbeta på ett reellt och skyddat sätt
20 och är försedd med ett första område minnesadresser som
reserverats för läsminnen (ROM) för utnyttjande av det
första området minnesadresser medelst processorn för åtkomst
av andra minnesanordningar, k ä n n e t e c k n a t
därav, att det omfattar steg, vid vilka man

25 identifierar adresser för nämnda ROM i det första
området minnesadresser;

identifierar avbrottshanterare för åtkomst av nämnda
da ROM;

30 identifierar adresser hos någon datatabell som upp-
bevaras i nämnda ROM;

tilldelar reellt arbetssätt-adressutrymme för en
expanderad minnesspecifikations(EMS)-ram, vari ROM kommer
att exekvera;

35 kopierar ROM-datatabellen i ett konventionellt min-
ne;

tilldelar adressutrymme i det konventionella minnet
för avbrottshanterarna;

placerar avbrottshanterarna i en avbrottstabell i
det tilldelade adressutrymmet;

5 modifierar visare i avbrottshanterarna i förhållan-
de till ROM-datatabellen för pekande på kopior av ROM-da-
tatabellen i det konventionella minnet;

överlagrar ROM-adresserna i det första området med
det expanderade minnet.

10 5. Förfarande enligt patentkrav 4, k ä n n e -
t e c k n a t därav, att det ytterligare inkluderar steg,
vid vilka man

placerar processorn i det skyddade arbetssättet vid
en första skyddsnivå;

15 sparar den löpande kartläggningen av EMS-sidramen;
inställer en återföringsadress på en stack av det
reella arbetssättet;

ändrar en kodsegment:instruktionsvisare (CS:IP)
så att den pekar på en ingångspunkt i ett flyttbart ROM;

20 placerar processorn i det reella arbetssättet;
exekverar kod i nämnda ROM;

återför styrning till nämnda återföringsadress;

ändrar nämnda CS:IP till en anrops-återgångsadress;

granskar uteffekten för utförande av nämnda ROM-

25 åtkomst och modifierar uteffekten ifall behövt;

återupprättar ursprungliga kartläggningen av sidra-
men;

placerar processorn i det reella arbetssättet.

30 6. Förfarande för avbrytande av flagghantering i
ett datorsystem, k ä n n e t e c k n a t därav, att det
omfattar steg, vid vilka man

medelst en arbetssättemulator detekterar ett ut-
förande av en instruktion för behandling av ett avbrott;

35 medelst arbetssättemulatorn inställer en stackgräns
hos en stack i datorsystemet;

i datorsystemet utför en till stacken hänförd instruktion för att möjliggöra åtkomst till stacken ytterom stackgränsen;

5 i datorsystemet detekterar ett överskridande av stackgränsen till följd av den nämnda åtkomsten; och modifierar en avbrottsflagga då denna överskridning detekterats.

7. Förfarande enligt patentkrav 6, k ä n n e -
t e c k n a t därav, att instruktionen är en till ett
10 avbrott hänförd instruktion.

8. Förfarande enligt patentkrav 6, k ä n n e -
t e c k n a t därav, att stackgränsen inställs till ett föreliggande värde hos en stackvisare.

9. Förfarande enligt patentkrav 7, k ä n n e -
15 t e c k n a t därav, att den till avbrottet hänfödda instruktionen är en klar avbrottsinstruktion (CLI).

10. Förfarande enligt patentkrav 7, k ä n n e -
t e c k n a t därav, att den till avbrottet hänfödda instruktionen är en inställd avbrottsinstruktion (STI).

20 11. Förfarande enligt patentkrav 6, k ä n n e -
t e c k n a t därav, att den till stacken hänfödda instruktionen är en utplockningsflagginstruktion (POPF), vilken utplockar ett flertal stackingångar ur stacken.

25 12. Förfarande enligt patentkrav 6, k ä n n e -
t e c k n a t därav, att den till stacken hänfödda instruktionen är en avbrytande återföringsinstruktion (IRET), vilken plockar ut ett flertal stackingångar ur stacken.

30 13. Förfarande enligt patentkrav 6, k ä n n e -
t e c k n a t därav, att utförandet av den till stacken hänfödda instruktionen och överskridandet av stackgränsen resulterar i ett allmänt skyddsundantag.

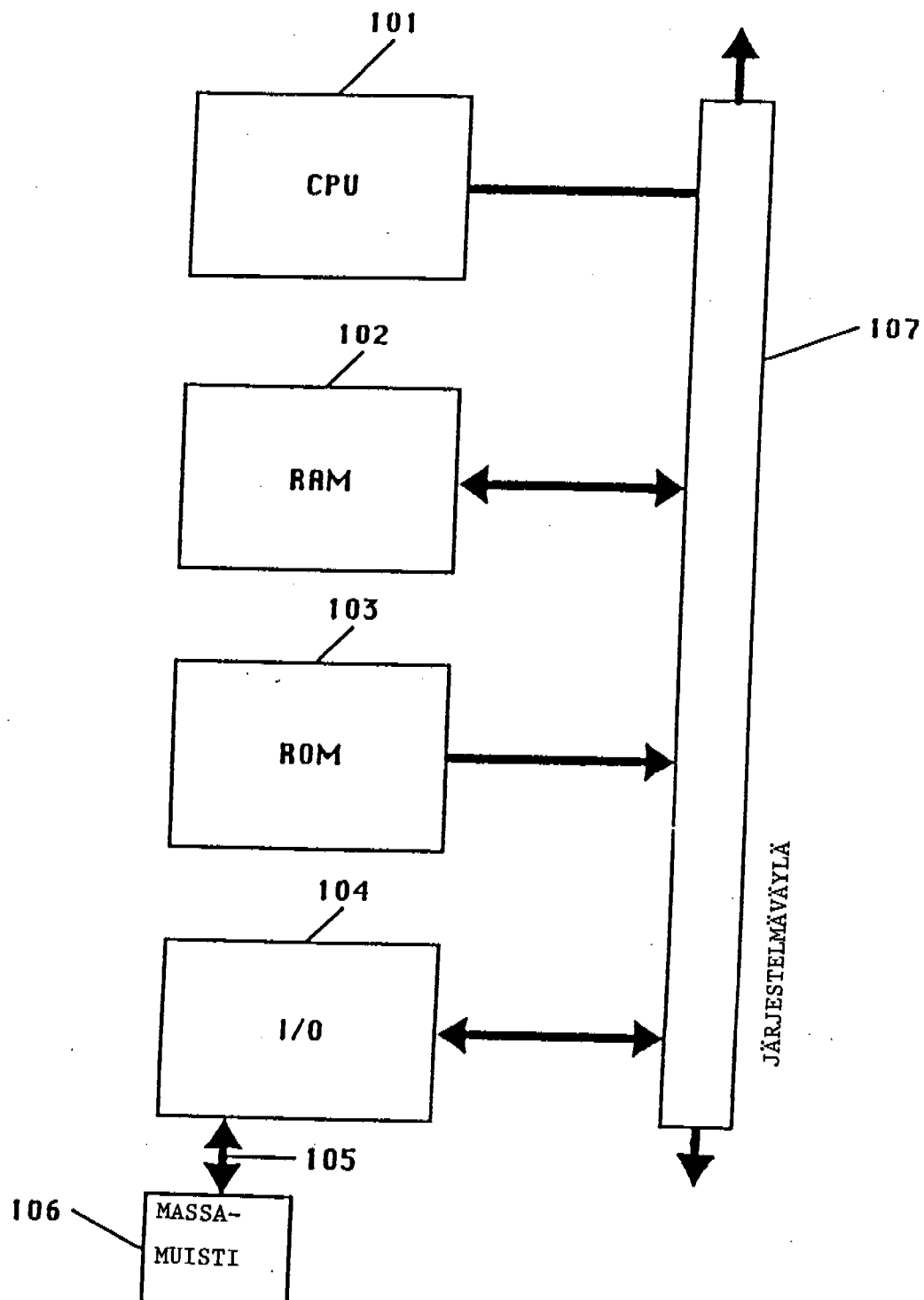


Fig. 1

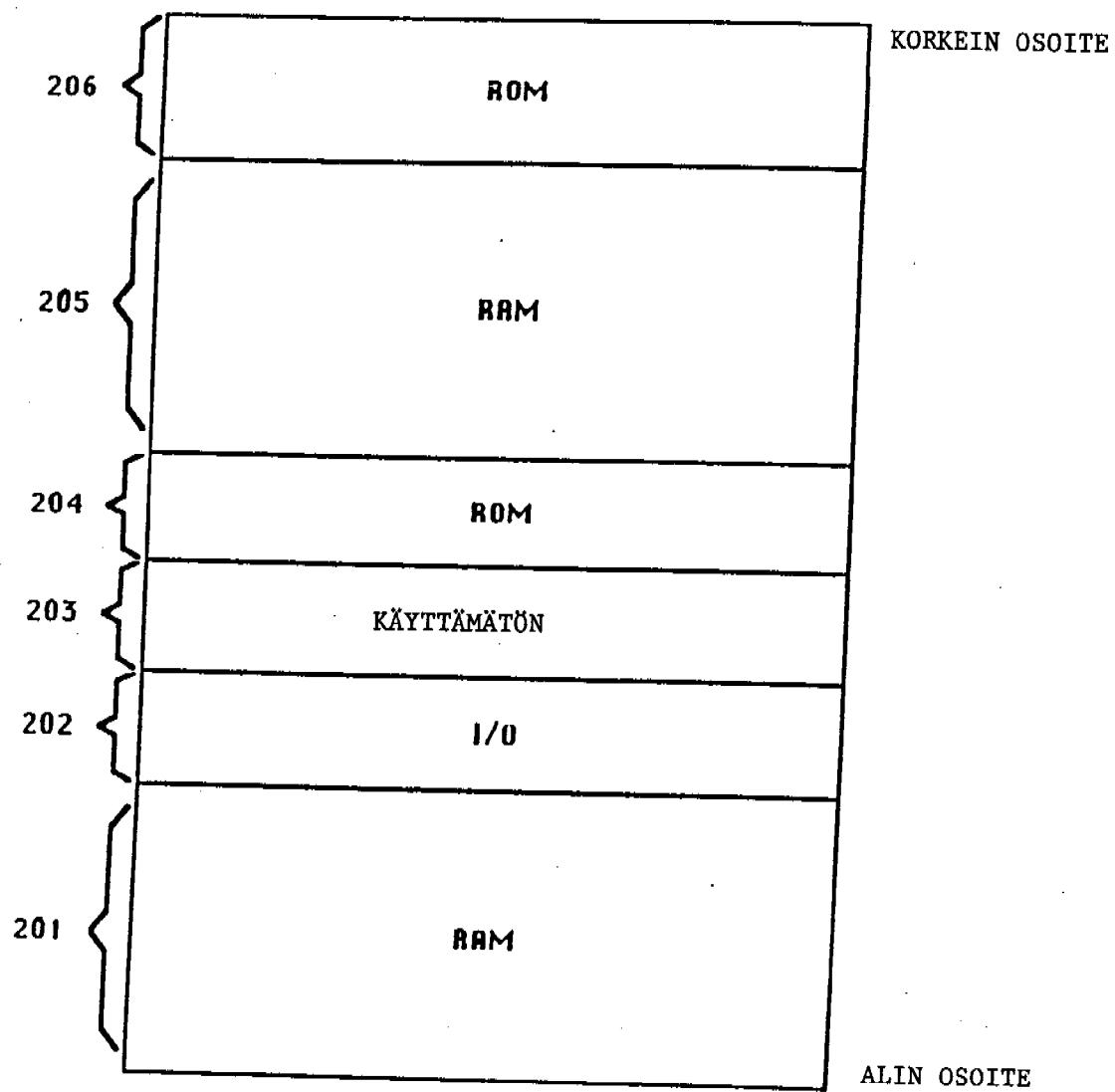


Fig. 2

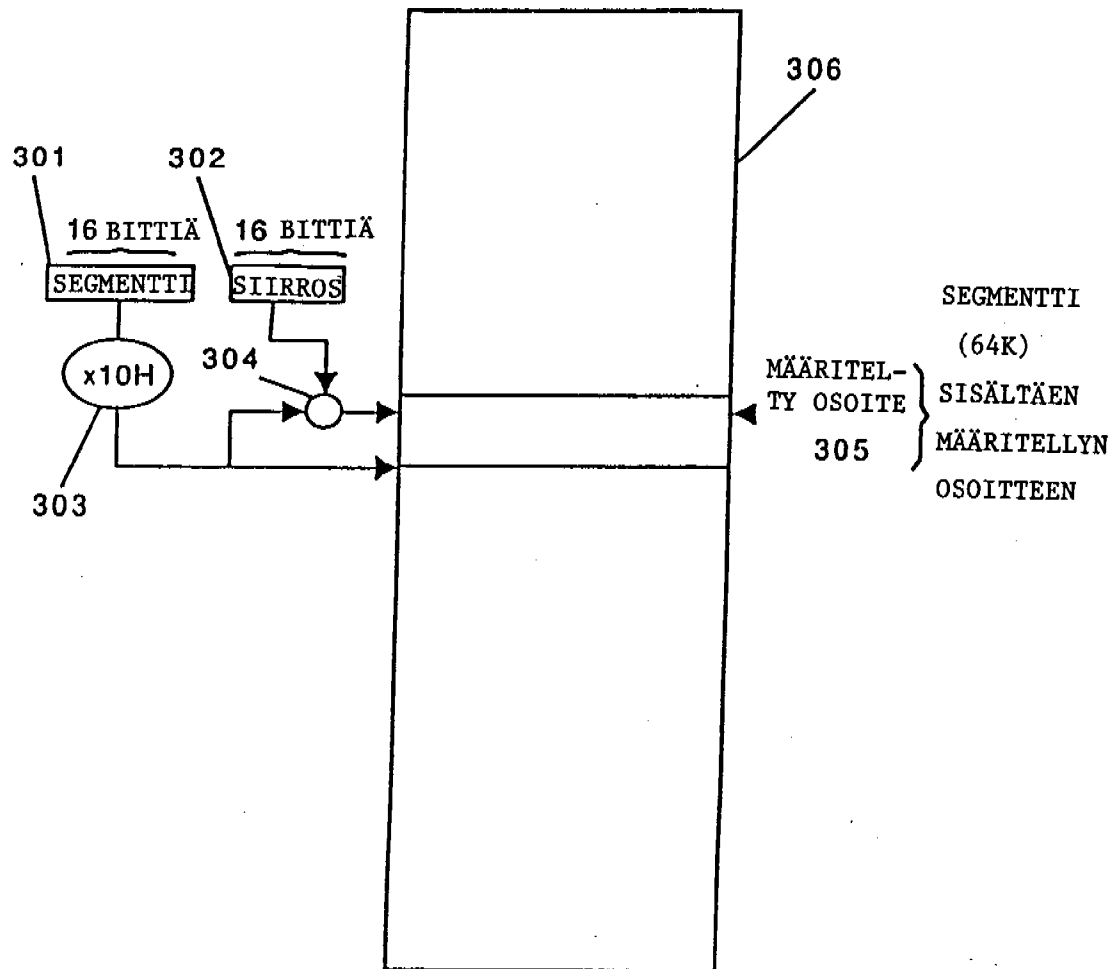


Fig. 3

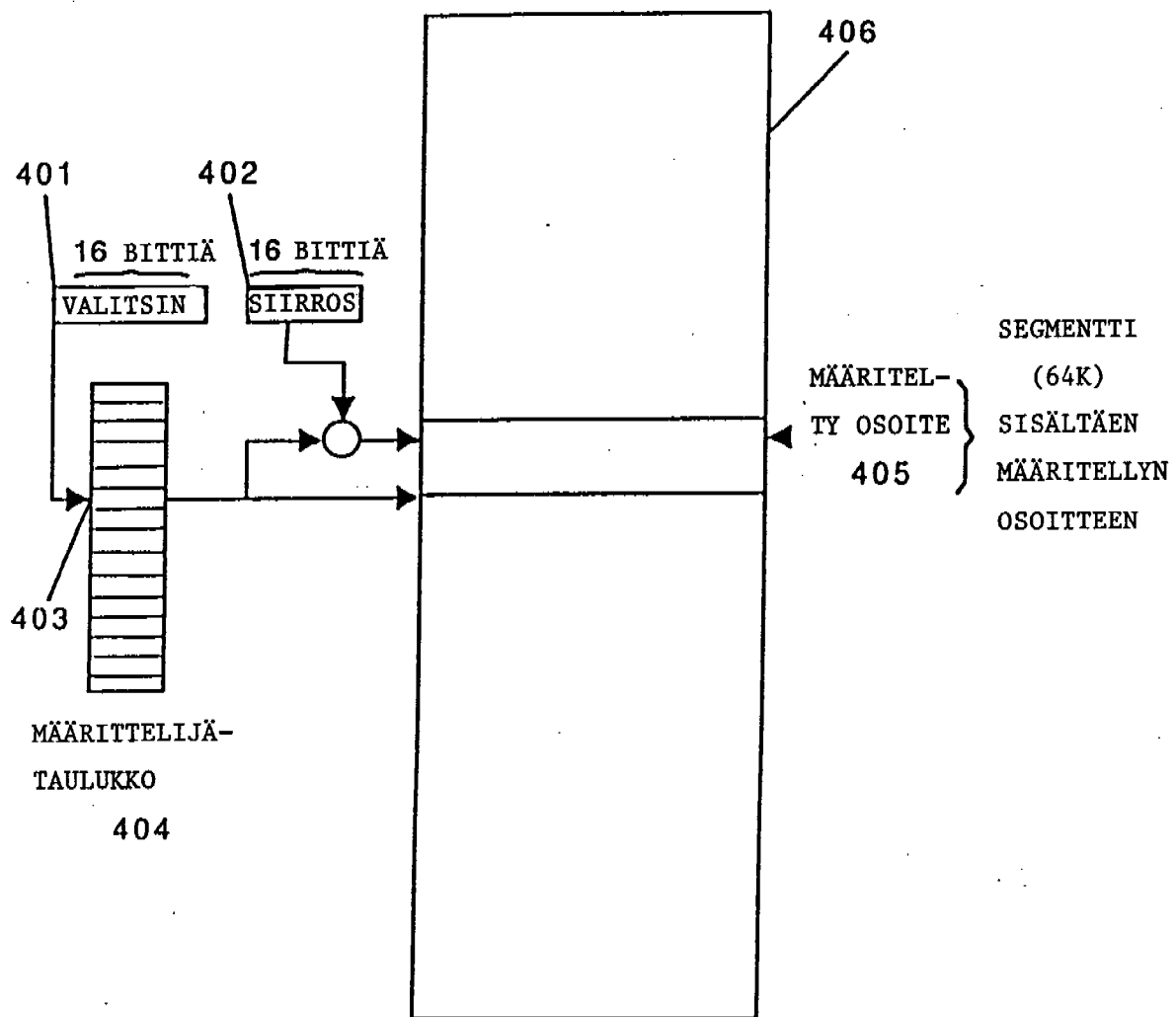


Fig. 4

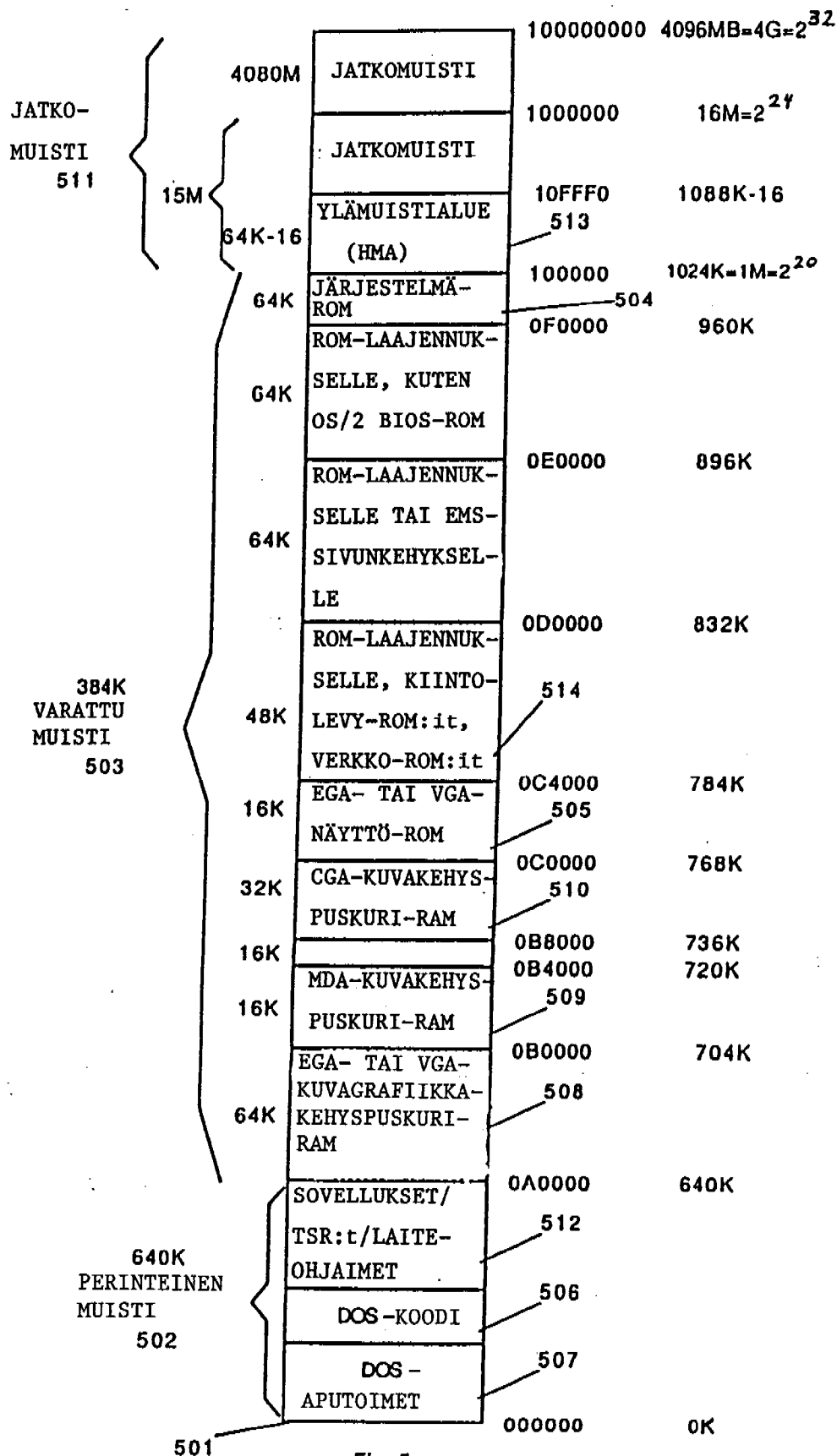


Fig. 5

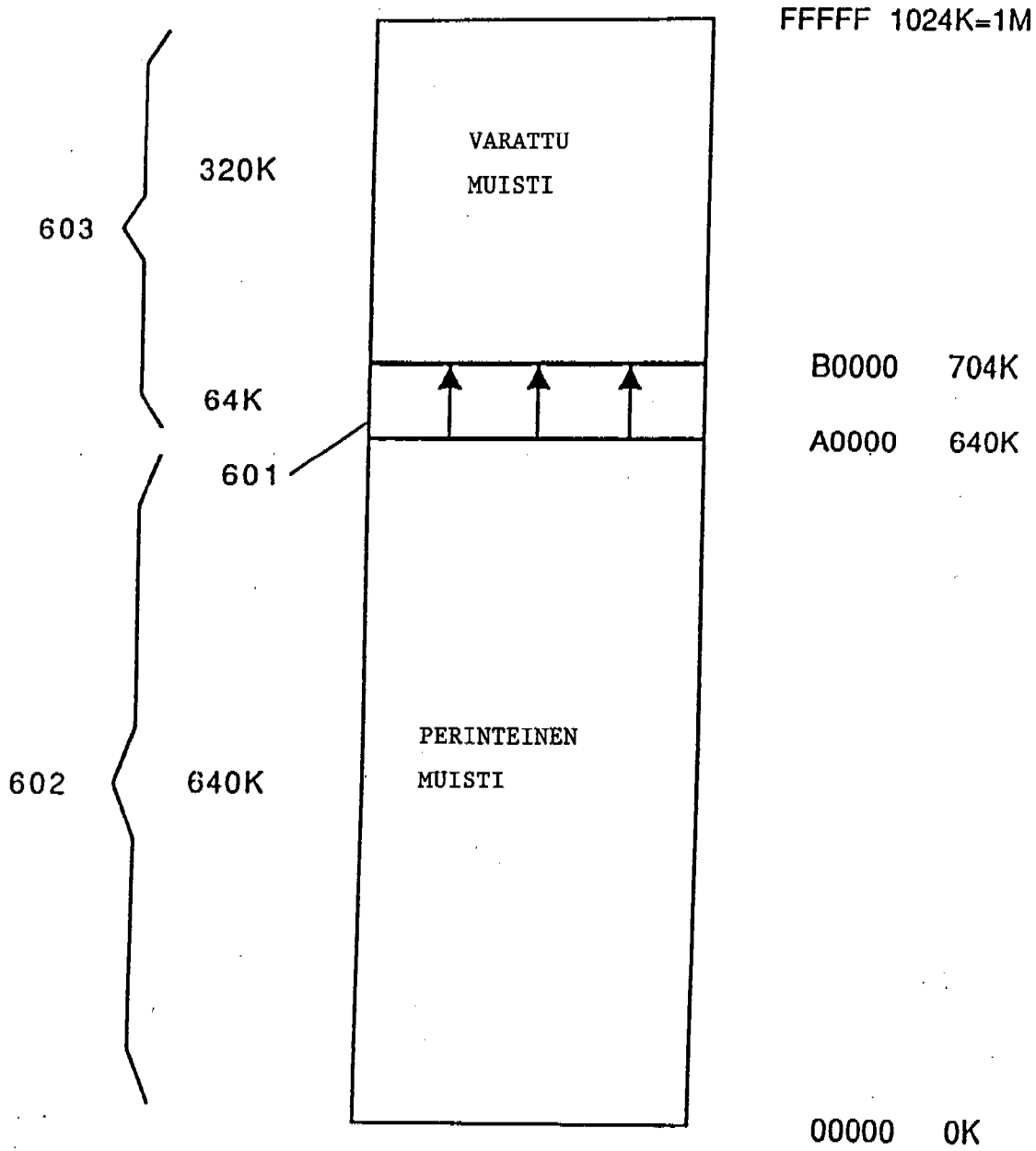


Fig. 6

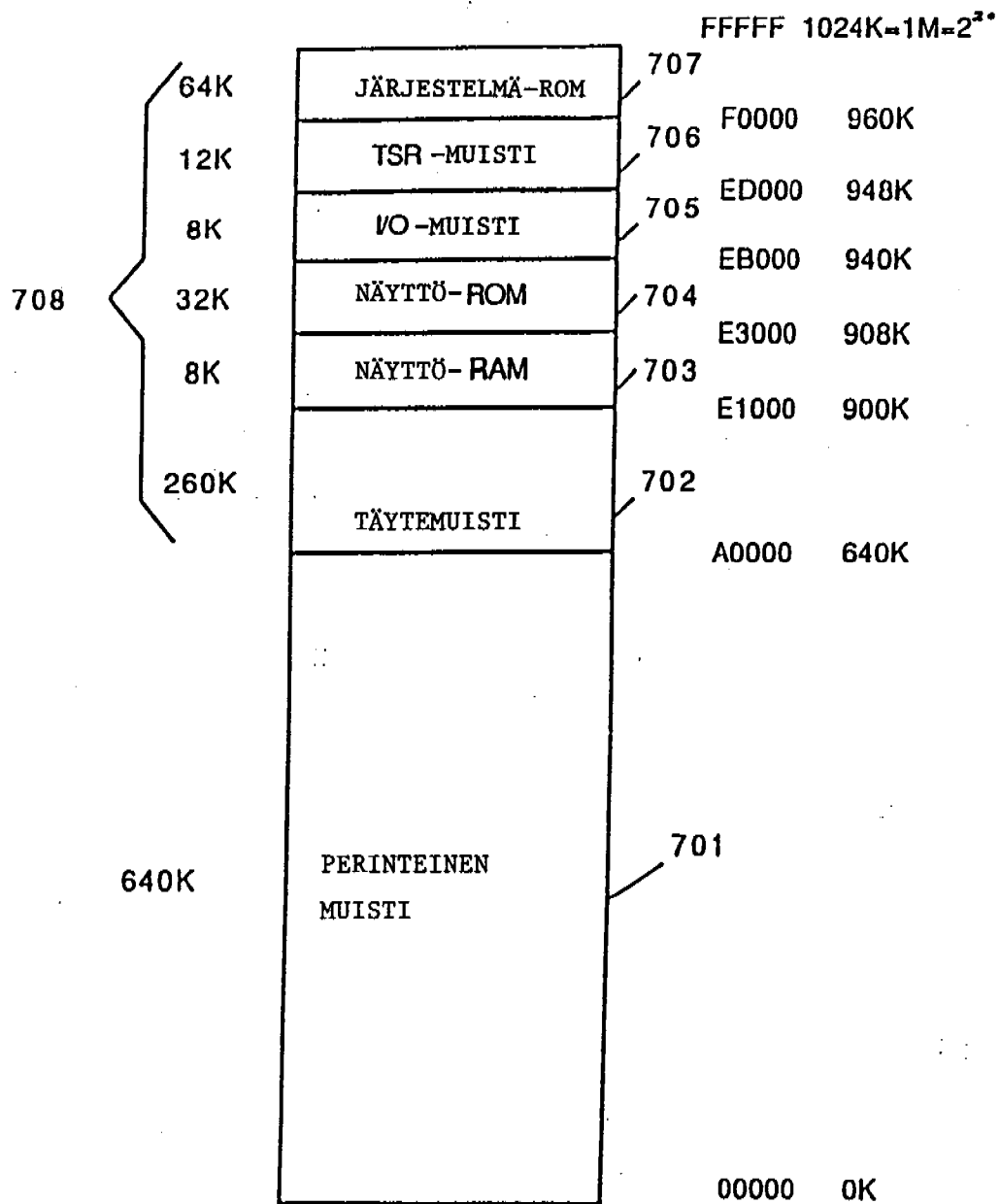


Fig. 7

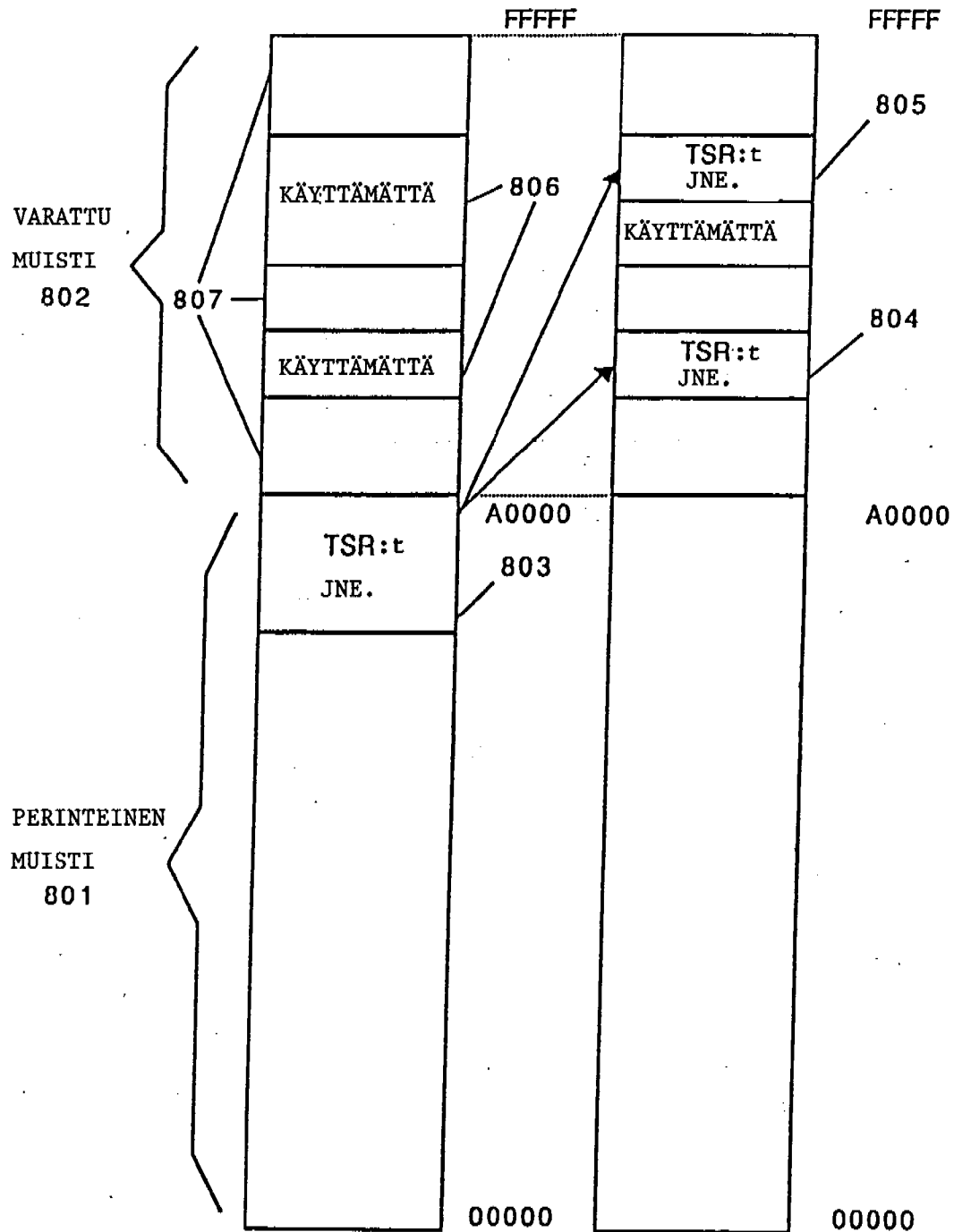


Fig. 8

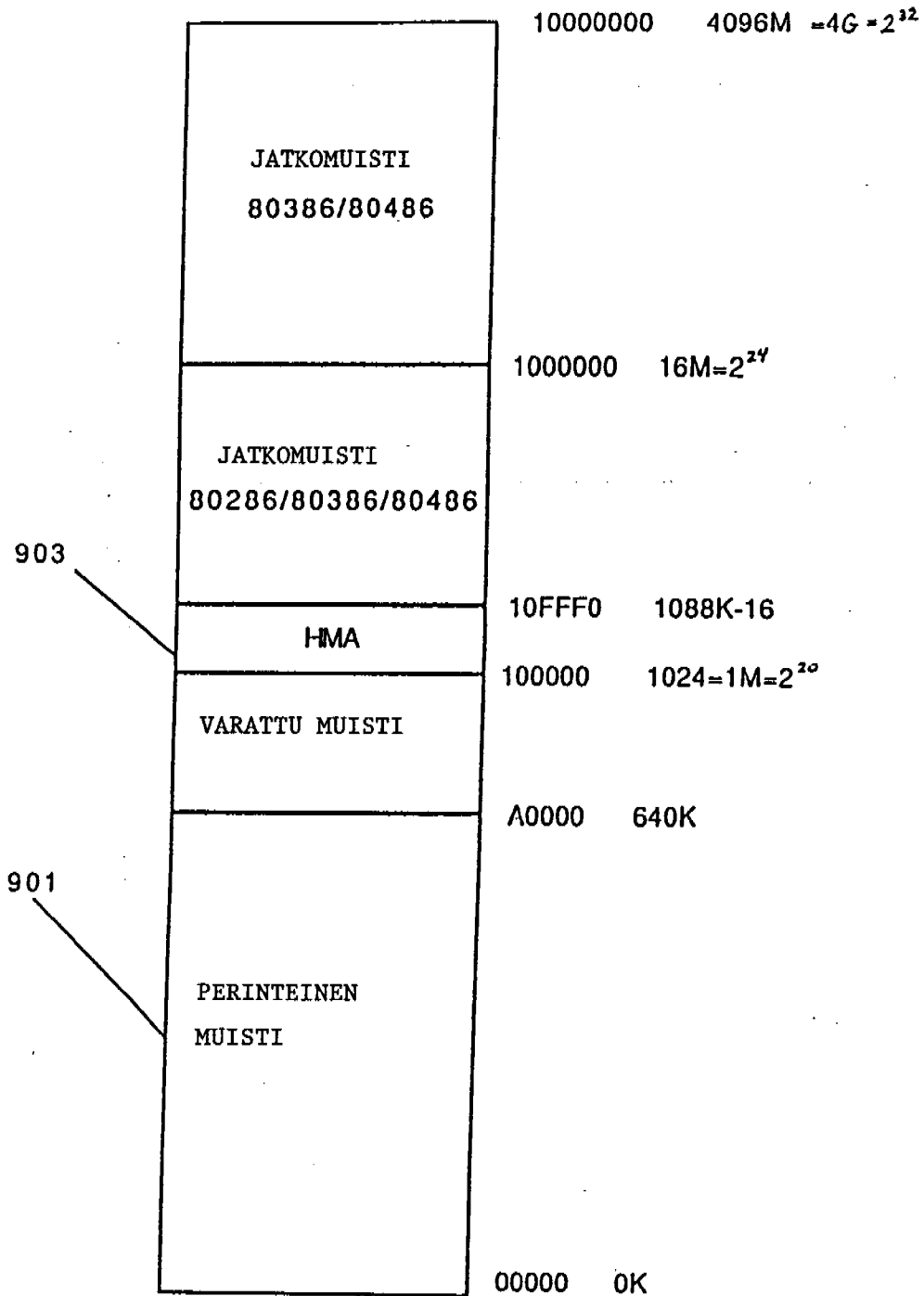


Fig. 9

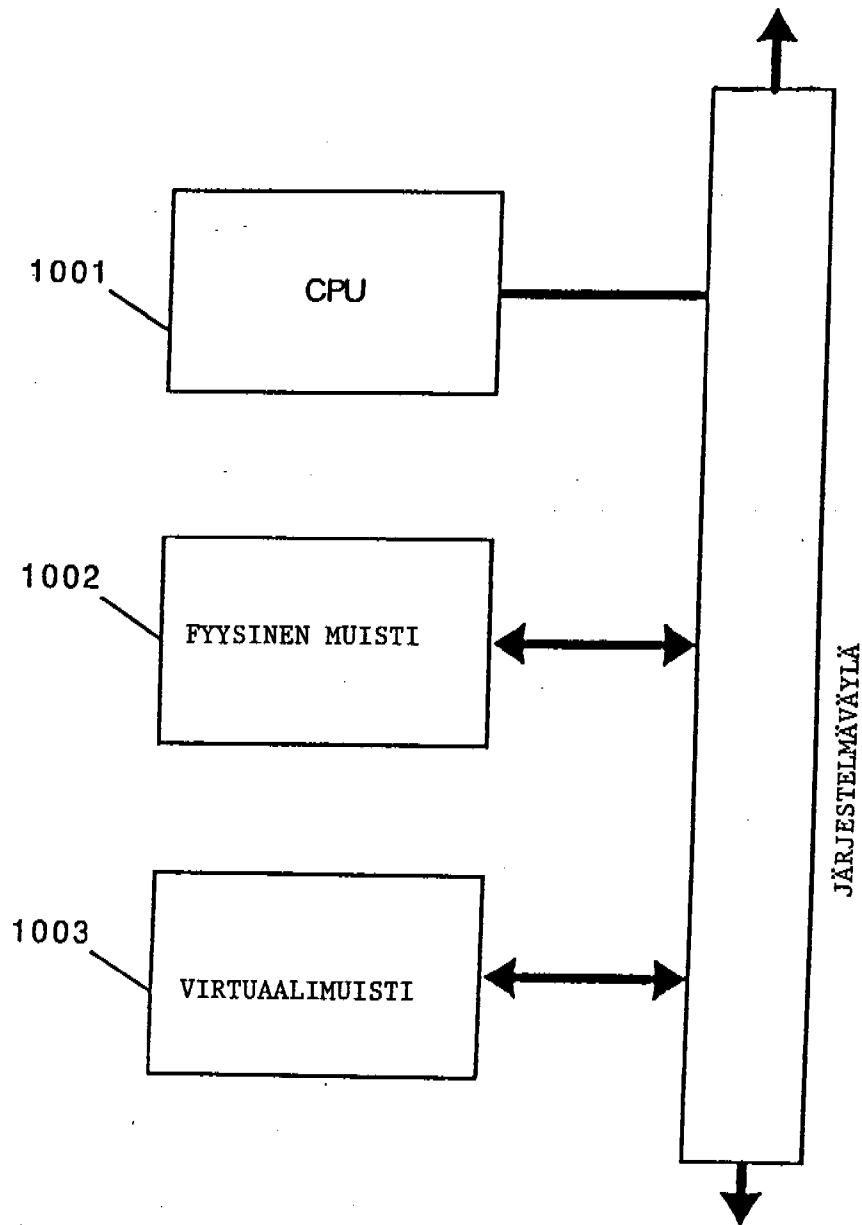


Fig. 10

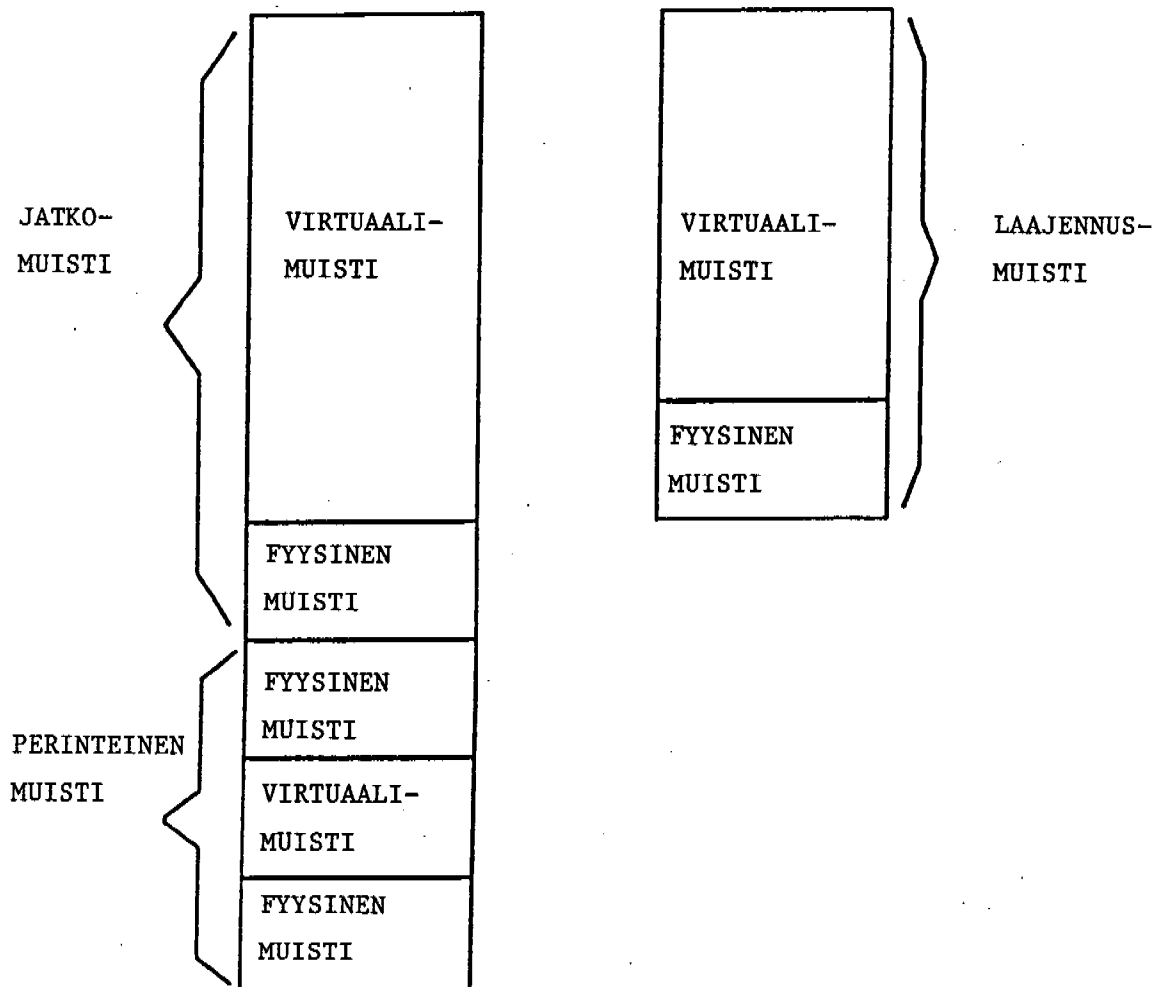


Fig. 11

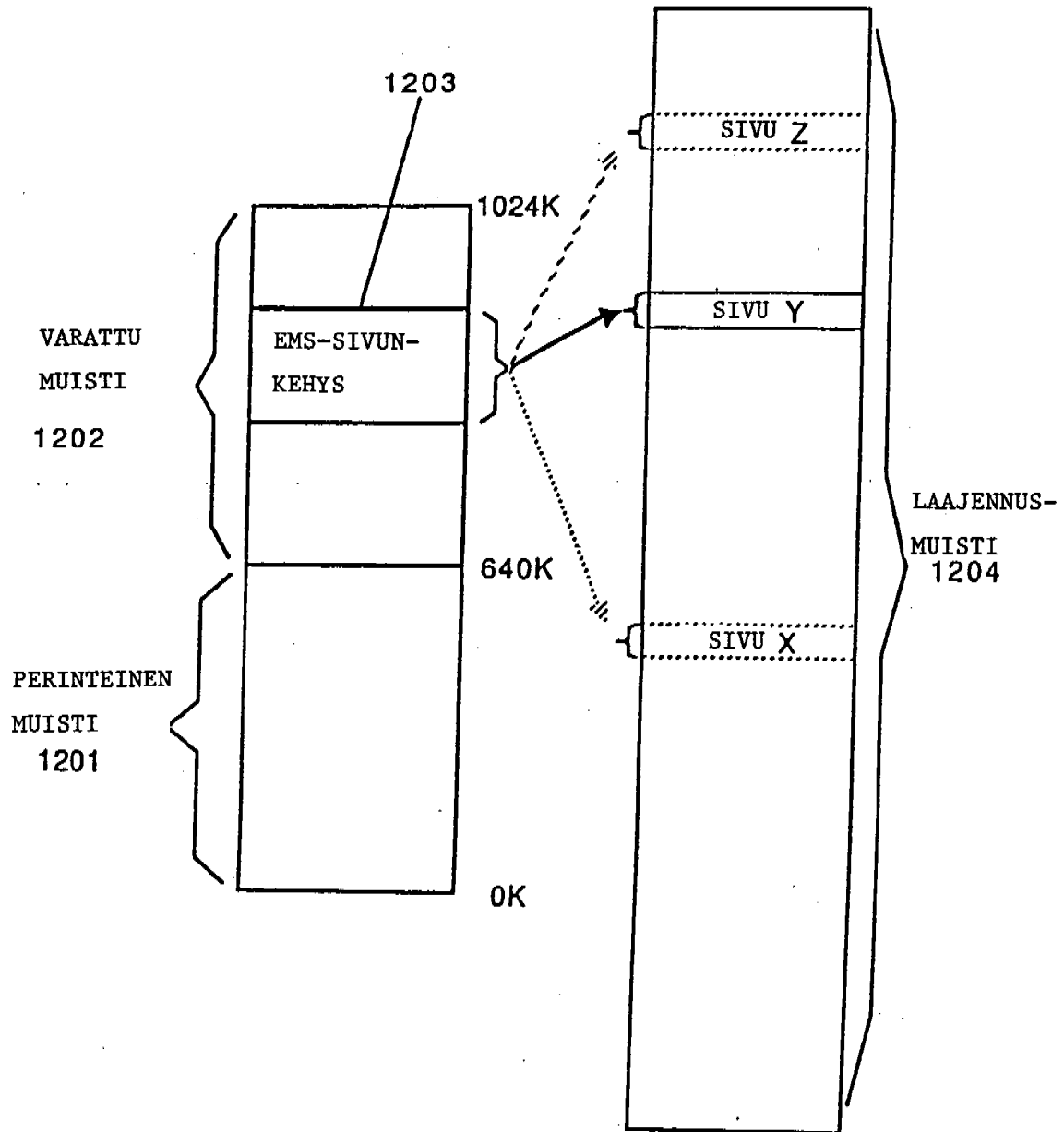


Fig. 12

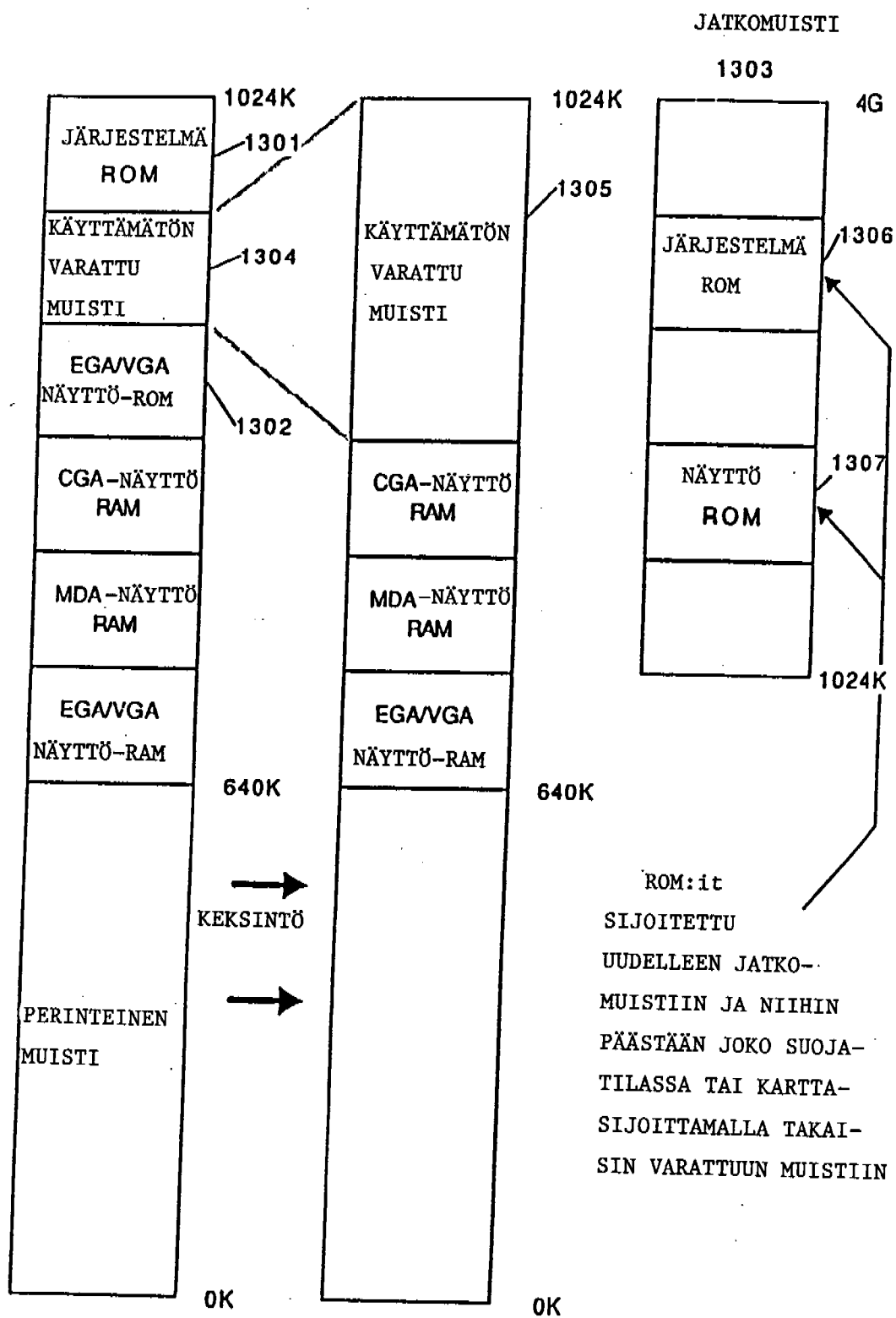


Fig. 13

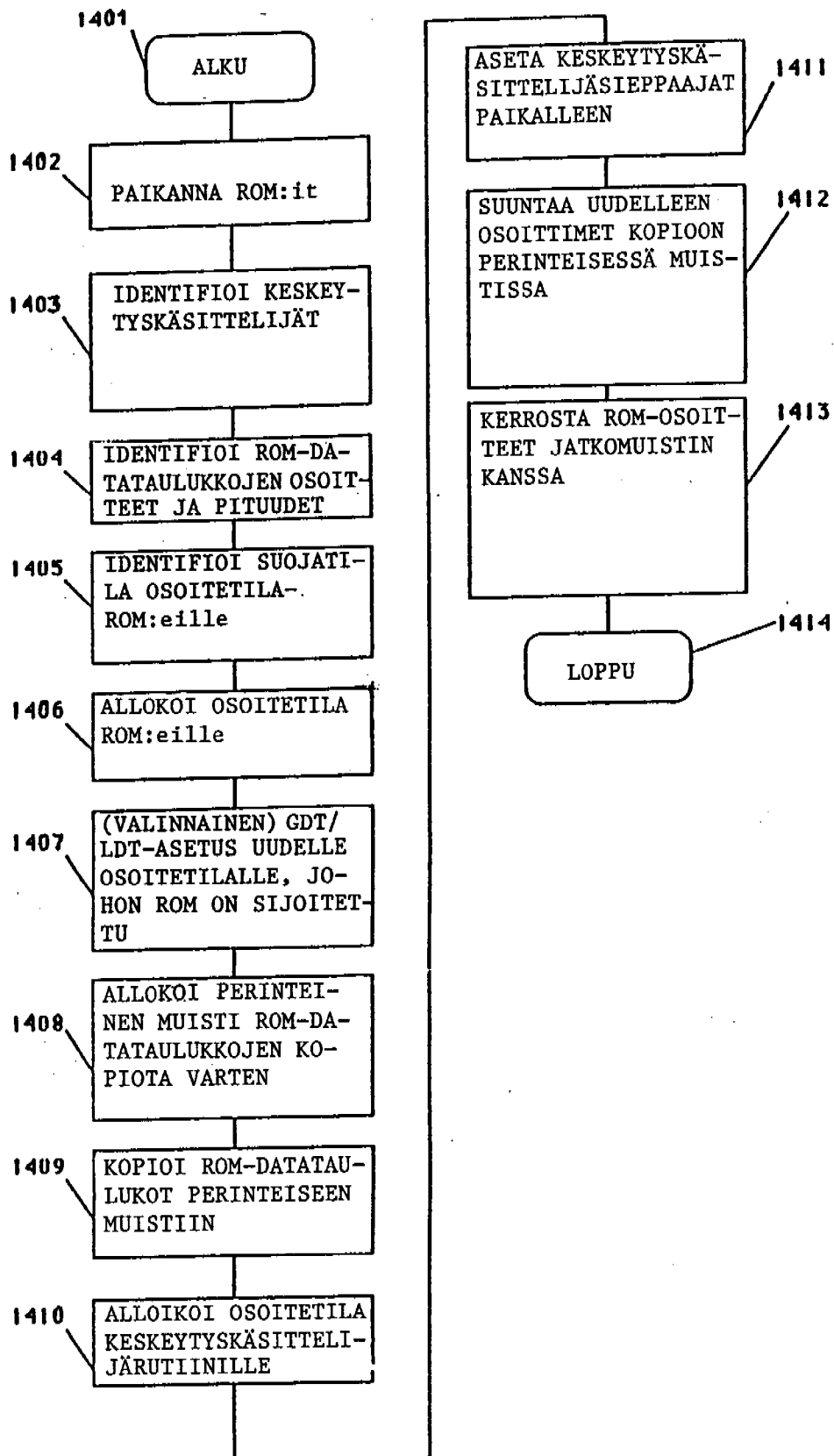


Fig. 14

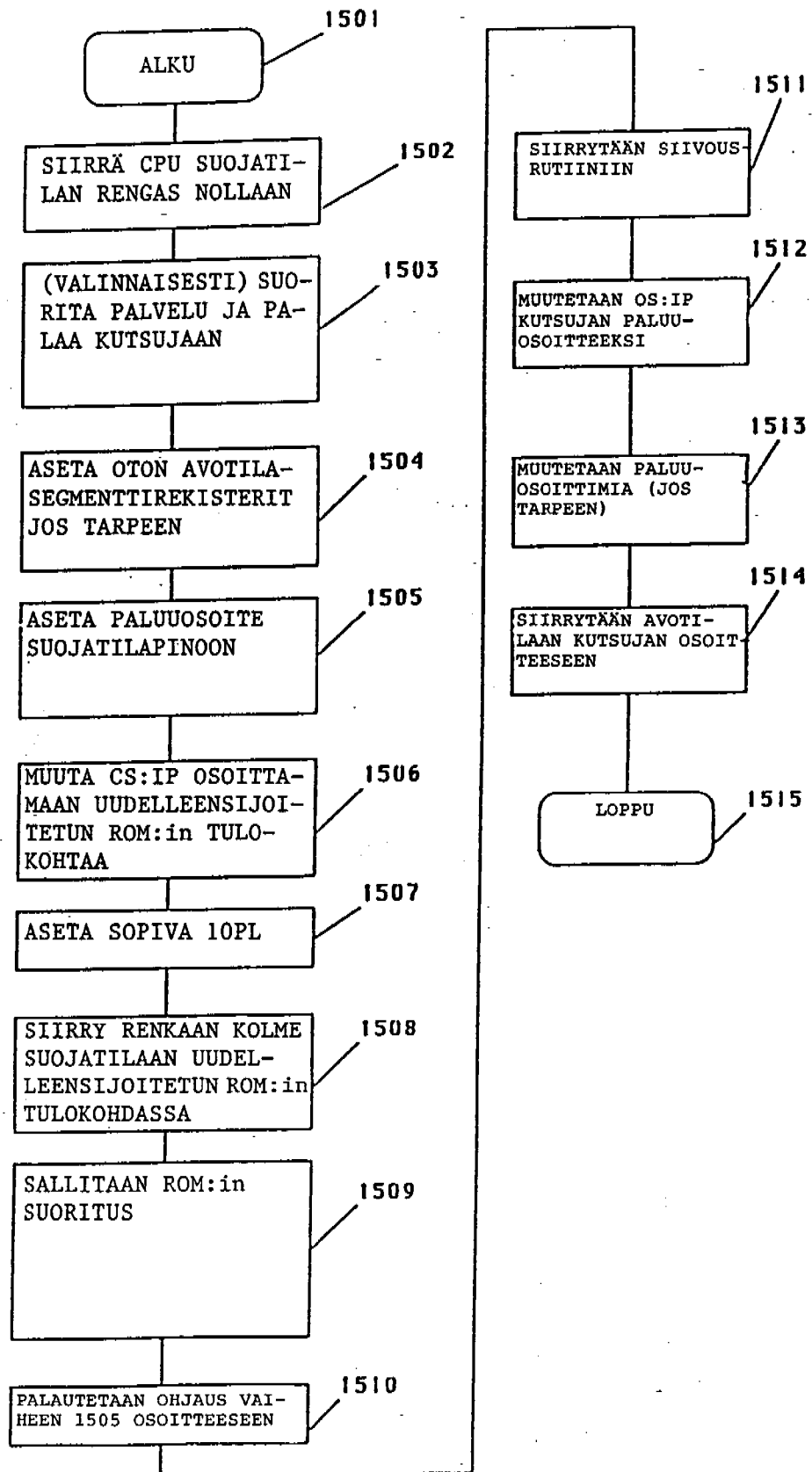


Fig. 15

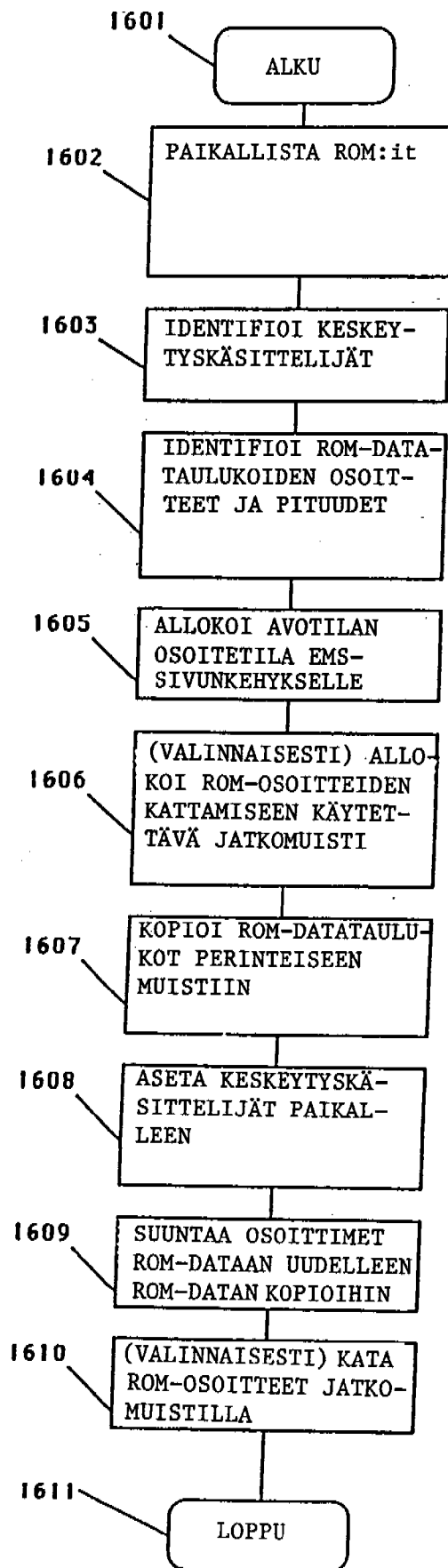


Fig. 16

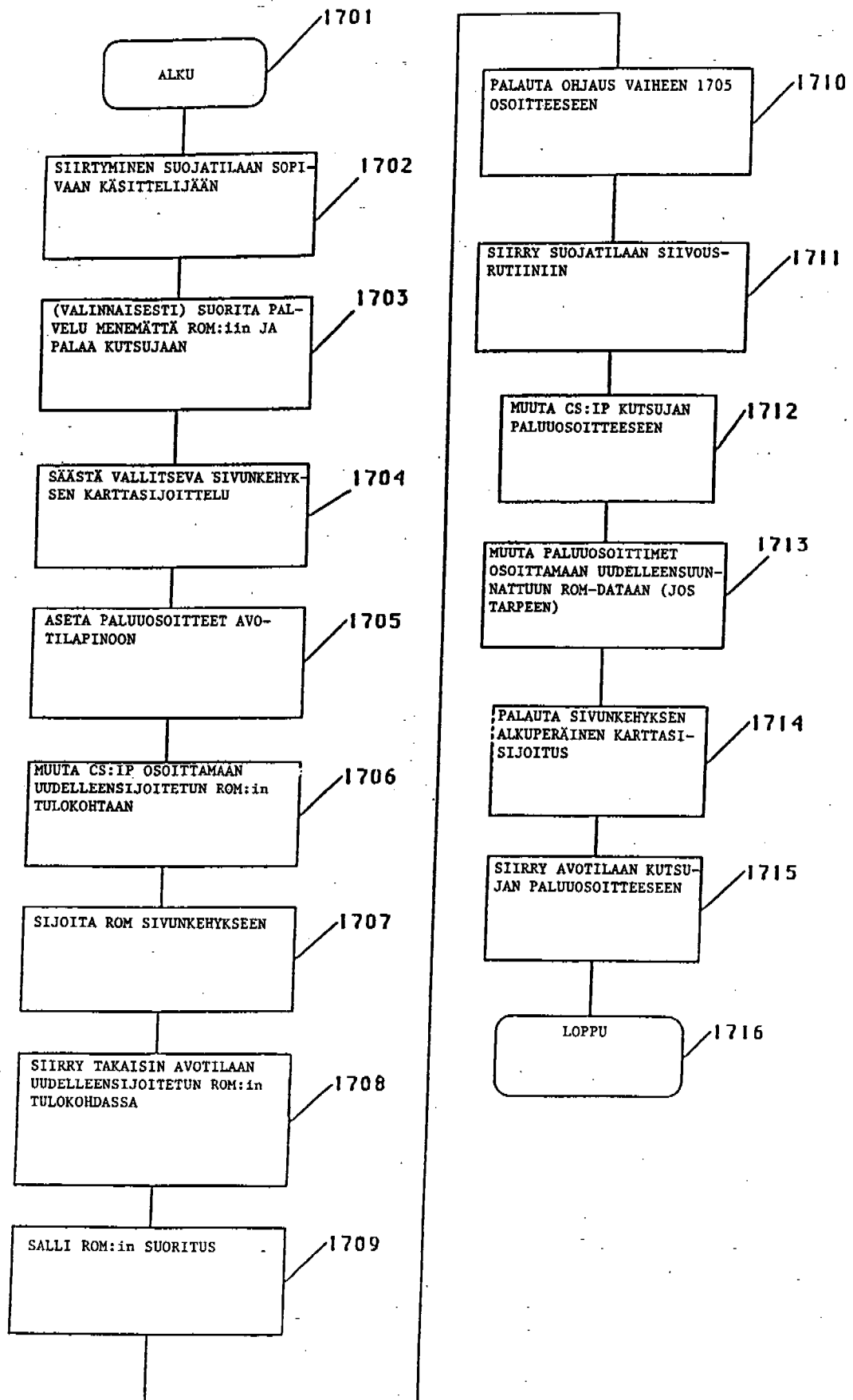


Fig. 17