FI000120117B

(12) **PATENTTIJULKAISU**
PATENTSKRIFT

(10) **FI 120117 B**

(45) Patentti myönnetty - Patent beviljats **30.06.2009**

(51) Kv.lk. - Int.kl.

G10L 19/04 (2006.01)

(21) Patenttihakemus - Patentansökning **954920**

(22) Tekemispäivä - Ingivningsdag **16.10.1995**

(24) Alkupäivä - Löpdag **13.02.1995**

(41) Tullut julkiseksi - Blivit offentlig **15.12.1995**

(86) Kv. hakemus - Int. ansökan **PCT/US1995/002043**

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

16.02.1994 US 197417 P

SUOMI – FINLAND
(FI)

PATENTTI- JA REKISTERIHALLITUS
PATENT- OCH REGISTERSTYRELSEN

(73) Haltija - Innehavare

1 • Qualcomm Incorporated, 5775 Morehouse Drive, San Diego, CA 92121, AMERIKAN YHDYSVALLAT, (US)

(72) Keksijä - Uppfinnare

1 • McDonough, John G., 11190 Caminito Innocenta, San Diego, CA 92126, AMERIKAN YHDYSVALLAT, (US)
2 • Chang, Chienchung, 11456 Cypress Terrace, San Diego, CA 92131, AMERIKAN YHDYSVALLAT, (US)
3 • Singh, Randeep, 10466 Caminito Alvarez, San Diego, CA 92126, AMERIKAN YHDYSVALLAT, (US)
4 • Sakamaki, Charles E., 12166 Via Milano, San Diego, CA 92128, AMERIKAN YHDYSVALLAT, (US)
5 • Tsai, Ming-Chang, 4427 Mistral Place, San Diego, CA 92130, AMERIKAN YHDYSVALLAT, (US)
6 • Kantak, Prashant, 3625 Earnscliff Place #23, San Diego, CA 92111, AMERIKAN YHDYSVALLAT, (US)

(74) Asiamies - Ombud

Oy Jalo Ant-Wuorinen Ab, Iso Roobertinkatu 4 - 6 A, 00120 Helsinki

(54) Keksinnön nimitys - Uppfinningens benämning

Vokooderi-ASIC
Vokoder-ASIC

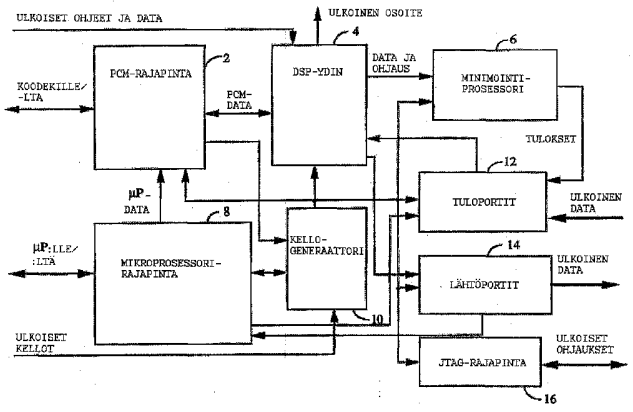
(56) Viitejulkaisut - Anförda publikationer

EP 0308817 A, EP 0291377 A, EP 0280827 A

(57) Tiivistelmä - Sammandrag

Menetelmä ja laite vokooderin toteuttamiseksi sovelluskohtaiselle integroidulle piirille (ASIC) esitetään. Laite sisältää DSP-ytimen (4), joka suorittaa laskentaa rajoitetun käskykannan (RISC) arkkitehtuurin mukaisesti. Lisäksi piiri käsittää DSP-ytimeen (4) erityisesti suunnitellun aliprosessorin, jota pidetään minimointiprosessorina (6). Lisäksi laitteeseen kuuluu erityisesti sitä varten suunniteltu lohkonormalisointipiiri.

Förfarande och anordning för förverkligande av en vokoder för en tilllämpningsspecifik integrerad krets (ASIC) förs fram. Anordningen innehåller en DSP-kärna (4), vilken utför beräkningar enligt en begränsad instruktionsmängd (RISC) arkitektur. Dessutom innehåller kretsen en speciellt för DSP-kärnan designerad slavprocessor, vilken används som en minimeringsprocessor (6). Ytterligare hör till anordningen en speciellt för densamma designerad blocknormaliseringskrets.



VOKOODERI-ASIC

KEKSINNÖN TAUSTA

I. Keksinnön ala

Esillä oleva keksintö liittyy puheenkäsittelyyn. Erityisesti esillä oleva keksintö liittyy uuteen ja kehittyneeseen menetelmään ja laitteeseen vokooderin toteuttamiseksi sovelluskohtaiselle integroidulle piirille (ASICille).

II. Liittyvän alan kuvaus

Äänen lähettäminen digitaalisilla tekniikoilla on levinnyt laajalle, erityisesti kaukoetäisyyksillä ja digitaaliradiopuhelinsovelluksissa. Jos puhetta lähetetään yksinkertaisesti näytteistämällä ja digitoimalla, vaaditaan datanopeus luokkaa 64 kilobittiä sekuntia kohden (kbps), jotta saavutetaan analogisten puhelimien puheenlaatua vastaava laatu. Kuitenkin käyttämällä puheen analysointia, sitä seuraavaa sopivaa koodausta, lähetystä ja syntetisointia vastaanottimessa, voidaan saavuttaa merkittävä datanopeuden pudotus. Laite, joka suorittaa puheen analysoinnin ja koodauksen lähettimessä ja syntetisoinnin vastaanottimessa tunnetaan vokooderina.

Viimeaikainen kehitys digitaalisten solukkopuhelinten standardeissa ja järjestelmissä on lisännyt näiden vokoodereiden tarvetta. Tarve kehittyneille vokoodausal

goritmeille, jotka käyttävät saatavilla olevan taajuuskaistan tehokkaammin ja palauttavat lähdepuheen tarkemmin, on lisännyt suuremman laskentakapasiteetin omaavien prosessorien tarvetta suorittamaan näitä monimutkaisempia vokoodausalgoritmeja.

Vokooderit käyttävät hyväkseen pitkä- ja lyhytkestoisten redundanssien esiintymistä puheessa, mikä vaatii laskennallisesti tehokkaita numeerisia toimintoja. Jotkin näistä toiminnoista sisältävät pitkäsekvenssisiä konvoluutioita, matriisin kääntämisiä, pitkäsekvenssisiä korrelaatioita ja niin edelleen. Näiden operaatioiden suorittaminen reaaliajassa aiheuttamatta sietämättömiä koodaus- ja dekoodausviiveitä puheenlähetyksessä vaatii tehokkaat laskentaresurssit.

Digitaalisten signaaliprosessorien (DSP:t) julkistaminen oli tärkeä tekijä vokooderialgoritmien reaaliaikatoteutuksissa. DSP:t ovat erityisen tehokkaita suorittaessaan aritmeettisia operaatioita, jotka ovat yleisiä vokooderialgoritmeissa. Edistymiset DSP:issa on lisännyt niiden laskennallista kapasiteettia 40 miljoonaan käskyyn sekunnissa (MIPS) ja sen yläpuolelle.

Esimerkkitarkoituksessa käytetty vokoodausalgoritmi on muuttuvanopeuksinen CELP-algoritmi, joka kuvataan yksityiskohtaisemmin US-patenttihakemuksessa sarjanro 08/004,484, joka on jätetty 14.1.1993, nimeltään ”Muuttuvanopeuksinen vokooderi” ja jossa haltijana on sama kuin tässä keksinnössä. Yllämainittu hakemus liitetään tähän viittauksella.

Taulukossa 1 alla esitetään esimerkkinä olevan vokoodausalgoritmin koodausosan ajoaikaprofiili yksittäiselle 20 millisekunnin puhekehykselle toteutettuna käyttäen tyypillistä DSP:tä. Koska esimerkkinä olevan vokoodausalgoritmin koodausosa vaatii merkittävästi enemmän käsittelyä kuin dekoodausosa, vain koodausosa on esitetty yksityiskohtaisesti taulukossa 1. Taulukossa 1 viitattua DSP:tä kellotetaan 40 Mhz:n taajuudella ja se suorittaa aritmeettiset operaatiot ja muut operaatiot yhdessä tai useassa kellojaksossa riippuen operaatioista. Ensimmäinen sarake edustaa esimerkkinä olevaa DSP:tä käyttävän vokoodausalgoritmin pääoperaatioita. Toinen sarake edustaa esimerkkinä olevaa DSP:tä käyttävän vokoodausalgoritmin vaatimien kellojaksojen määrää kunkin tietyn operaation suorittamiseksi. Kolmas sarake edustaa tietyn operaation vaatimaa prosenttiosuutta kokonaisprosessointiajasta. Esimerkkivokoodausalgoritmissa vaaditaan, että kaikki operaatiot suoritetaan 20 millisekunnissa esimerkkivokoodausalgoritmin reaaliaikaisilla operaatioilla. Tämä asettaa vaatimuksia algoritmin toteuttavan DSP:n valinnalle niin, että DSP kykenee toimimaan kellotaajuudella tai sen yläpuolella, joka vaaditaan vaaditun käsittelyn suorittamiseksi 20 millisekunnin kehyksessä. Taulukossa 1 kuvatulle tyypilliselle DSP:lle, tämä rajoittaa kellojen määrän 800000:een.

Taulukko I

Rutiini	40 Mhz kellot	% kaikista
LPC-analyysi	24810	3.89
LPC-LSP-muunnos	48780	7.65
Nopeuspäättely	330	0.05
LSP-kvantisointi	660	0.10
LSP-kvantisoinnin palautus	1110	0.17
LSP/LPC-interpolointi	34590	5.42
Äänentason etsintä	225570	35.36
Koodikirjan etsintä	262380	41.13
Synteteesi	34230	5.37
Pakkaus	5490	0.86
Yhteensä	637950	100.00

Kuten voidaan nähdä taulukosta 1, äänentasoetsintä- ja koodikirjaetsintätoiminteet kuluttavat yli 75 prosenttia käsittelyajasta vokooderialgoritmin koodausosassa. Koska laskennallisen kuormituksen enemmistö on näissä kahdessa etsintäalgoritmissa, ensisijainen tavoite tehokkaalla vokoodausalgoritmia suorittavalla ASIC:lla

on vähentää näiden kahden algoritmin suorittamiseen käytettävien kellojaksojen määrää.

Esillä olevan keksinnön mukainen menetelmä ja laite vähentävät merkittävästi näiden etsintäalgoritmien suorittamiseen tarvittavien käskyjaksojen määrää. Edelleen esillä oleva keksintö tuo esiin menetelmän ja laitteen, jotka on optimoitu vokoodausalgoritmeille erityisen merkittävien operaatioiden suorittamiseksi tehokkaammin. Esillä olevan keksinnön mukaisen menetelmän ja laitteen sovellukset eivät rajoitu esimerkkinä olevan vokoodausalgoritmin suorittamiseen tai edes puheen koodaukseen tai dekoodaukseen. On ajateltu, että menetelmät ja laite voidaan soveltaa mihin tahansa järjestelmään, joka käyttää digitaalista signaalinkäsittelyalgoritmia, kuten kaiun poistoon ja kanavatasaimiin.

KEKSINNÖN YHTEENVETO

Esillä oleva keksintö on uusi ja kehittynyt menetelmä ja laite vokoodausalgoritmin suorittamiseksi.

Tässä kuvattu esillä olevan keksinnön esimerkkisovellus on muuttuvanopeuksisen CELP-algoritmin, joka kuvataan yksityiskohtaisemmin yllämainitussa patentissa, ASIC-toteutus. Esillä olevan keksinnön ominaisuudet voidaan vastaavasti soveltaa kaikkiin lineaarisesti ennustaviin koodausalgoritmeihin (LPC). Esillä oleva keksintö tuo esiin arkkitehtuurin, joka on optimoitu toteuttamaan vokooderialgoritmi alennetulla määrällä kellotaajuuksia ja pienemmällä tehonkulutuksella. Alkuperäinen optimoinnin äärimmäinen tavoite oli minimoida tehon kulutus. Kellotuksen vähentäminen vaadit-

tavan algoritmin suorittamiseksi oli myös tekemisissä kellotaajuuden vähentämisen kanssa vähentäen suoraan tai epäsuoraan tehonkulutusta. Suora vaikutus johtuu tehon kulutuksen ja kellotaajuuden välisestä suhteesta komplementaarisessa metallipuolijohde (CMOS) -laitteessa. Epäsuora vaikutus johtuu neliöllisestä suhteesta tehonkulutuksen ja CMOS-laitteen jännitteen välillä ja kyvystä alentaa jännitettä yhdessä kellotaajuuden kanssa. Vokooderi-ASICin tehokkuuden mitta on kellojaksossa suoritettava prosessointi. Näin ollen tehokkuuden lisääminen vähentää algoritmin suorittamiseen tarvittavien kellojaksojen määrää.

Ensimmäinen menetelmä vokoodausalgoritmin tehokkuuden lisäämiseksi on erityinen DSP-ydinarkkitehtuuri. Esimerkissä oleva DSP-ydin lisää muistin läpäisyä aikaansaamalla kolme hajasaantimuistielementtiä (RAM). Kukin kolmesta RAM-elementistä on osoitettu muistiosoitteen muodostusyksikköön. Tämä muistin kolmeen osaan jakaminen mahdollistaa tehokkaan suorituksen operaatioille, kuten rekursiiviselle konvoluutiolle operandeja antamalla, laskentatuloksille ja talletuksille kaikki yhdessä jaksossa. Operandien haku, tulosten laskenta ja tallennus johdetaan niin, että täydellinen rekursiivinen konvoluutio yhdelle tulokselle suoritetaan kolmen jakson aikana, mutta uusi tulos muodostetaan joka jakso. Kolmeen osaan jaettu muisti vähentää kellojaksojen tarvetta muita toimintoja varten vokoodausalgoritmissa. Rekursiivisen konvoluution tehokas suorittaminen aikaansaa merkittävimmät säästöt vokooderialgoritmissa.

Toinen menetelmä vokoodausalgoritmin tehokkuuden lisäämiseksi on järjestää erillinen aliprosessori DSP-ytimeen, jota aliprosessoria pidetään minimointiprosessorina. Minimointiprosessori suorittaa korrelaatiot, laskee keskimääräisen neliöllisen virheen (MSE) ja

etsii minimi MSE:tä käyttäen DSP-ytimen antamaa dataa. Minimointiprosessori jakaa laskennallisesti tehokkaat korrelaatio- ja minimointitehtävät DSP-ytimen kanssa. Minimointiprosessoriin kuuluu ohjauselementti, joka valvoo minimointiprosessorin toimintaa ja joka voi supistaa MSE-minimointitehtävää tietyissä olosuhteissa. Näissä olosuhteissa jatkettu etsintä ei kykene aikaansaamaan MSE:tä alle nykyisen MSE-minimin johtuen matemaattisista rajoitteista. MSE-minimointitehtävää supistavia menetelmiä pidetään minimointiprosessorin tehosäästötoimintoina.

Kolmas menetelmä vokoodausalgoritmin tehokkuuden lisäämiseksi esimerkkisovelluksessa on järjestää tarkoitukseen sopiva laitteisto tehokkaan lohkonormalisoinnin suorittamiseksi. Vokoodausalgoritmin laskennassa on tarvetta ylläpitää korkein mahdollinen laskentatarkkuus. Järjestämällä sopiva laitteisto, lohkonormalisointi voidaan suorittaa samanaikaisesti muiden vokoodausalgoritmin operaatioiden kanssa, jolloin vokoodausalgoritmin suorittamiseen vaadittavien jaksojen määrä pienenee.

PIIRUSTUSTEN LYHYT KUVAUS

Esillä olevan keksinnön muodot, tarkoitukset ja edut tulevat selvemmiksi seuraavasta yksityiskohtaisesta kuvauksesta viitaten oheisiin piirustuksiin, joissa on samat viitenumerot kauttaaltaan ja joissa:

kuvio 1 on lohkokaavio, joka esittää esillä olevan keksinnön mukaista laitetta;

kuvio 2 on esillä olevan keksinnön toiminnan toiminnallinen esitys;

kuvio 3 on vuokaavio, joka esittää esimerkin esillä olevan keksinnön mukaisesta koodaustoiminnasta;

kuvio 4a - d ovat kaavioita, jotka esittävät vokooderin bittiallokointia eri nopeuksilla ja jotka esittävät eri nopeuksilla käytettyjen aste- ja koodikirja-alikehysten määrää;

kuviot 5a - d ovat lohkokaavioita esillä olevan keksinnön mukaisen esimerkkisovellutuksen DSP-ytimestä;

kuviot 6a - d ovat lohkokaavioita esillä olevan keksinnön mukaisen esimerkkisovellutuksen minimointiprosessorista;

kuvio 7 esittää äänenetasoetsintätoimintoa, joka suoritetaan esillä olevan keksinnön esimerkkisovellutuksessa;

kuvio 8 on vuokaavio, joka esittää esillä olevan keksinnön esimerkkisovellutuksessa suoritettavaa äänenetasoetsintätoimintoa;

kuvio 9 esittää koodikirjaetsintätoimintoa, joka suoritetaan esillä olevan keksinnön esimerkkisovellutuksessa;

kuvio 10 on vuokaavio, joka esittää esillä olevan keksinnön esimerkkisovellutuksessa suoritettavaa koodikirjaetsintätoimintoa;

kuvio 11 on lohkokaavio kooderin dekooderista, joka vastaa kooderin suodatinmuistin pitämisestä yhdessä päässä ja dekooderin toisessa päässä tietoliikennelinkkiä samassa vokooderitoiminnassa esillä olevan keksinnön mukaisesti; ja

kuvio 12 on esillä olevan keksinnön mukaisen esimerkkisovellutuksen dekooderin lohkokaavio.

SUOSITELTUJEN SUORITUSMUOTOJEN YKSITYISKOHTAINEN KUVAUS

Viitaten nyt kuvioihin, kuvion 1 DSP-ydin, joka esitetään kuvioissa 5a - d, suunnitellaan kolmiosaisen hajasaantimuistin (RAM) ympärille, (RAM A 104, RAM B 122 ja RAM C 182), lukumuistin (ROM), (ROM E 114) ja tehokkaan aritmeettis-loogisen yksikön (ALU) (ALU 143). Kolmiosainen hajasaantimuisti RAM mahdollistaa ALU:n tehokkaamman käytön ja lisääntyneen RAM-kaistanleveyden verrattuna yksittäisellä RAM:lla saavutettavaan. Haluttu ROM, ROM E 114 aikaansaa 16-bittiset vakiot. RAM-osat RAM A 104, RAM B 122 ja RAM C 182 ja ROM, ROM E 114 antavat datan ALU:un 143. RAM C 182 hyväksyy ja muodostaa 32-bittistä dataa ALU:sta ja ALU:un 143 kun taas RAM A 104 ja RAM B 122 hyväksyvät ja muodostavat 16-bittistä dataa tehden laskennan 16-bittisillä vakioilla ja tallentamisen 32-bittisillä vakioilla erittäin tehokkaaksi.

Kullakin osalla on haluttu osoitteenmuodostusyksikkö. RAM A:n 104 on osoiteyksikkö A 102, RAM B:n 122 on osoiteyksikkö B 120 ja RAM C:n 182 on osoiteyksikkö C 186 ja ROM E:n 114 on osoiteyksikkö E 112. Kukin osoiteyksikkö käsittää rekistereitä, multipleksereitä ja lisääjiä/vähentäjiä (ei esitetty). Yhden kellojakson aikana DSP-ydin 4 voi suorittaa kolme muistioperaatiota, kolme osoitepäivitystä, aritmeettisen operaation (t.s. kerro-kerää-normalisoi) ja datan siirron normalisointiprosessorille.

Komento ROM, ROM I 194 tallettaa komennot, jotka ohjaavat DSP-ytimen 4 suoritussekvenssiä. ROM:iin I 194 talletettu komentosekvenssi kuvaa DSP-ytimellä 4 suo-

ritettavat prosessointitoiminnat. ROM I 194 on haluttu osoitteenmuodostusyksikkö, IP-laskin ja pino 196.

RAM-osoitteenmuodostusyksiköt tai rekisteritiedostot, osoiteyksikkö A 102, osoiteyksikkö B 120, osoiteyksikkö C 186 antavat osoitteen ja datan vastaaville RAM-operaatioille. Data voidaan siirtää rekisteritiedostoelementistä toiseen samassa osoiteyksikössä tai vastaavaan RAM-muistiin. Esimerkkisovellutuksessa osoiteyksikkö A 102 antaa datan multiplekserin 106 kautta RAM:iin A 104, osoiteyksikkö B 120 antaa datan multiplekserin 124 kautta RAM:iin B 122 ja osoiteyksikkö C 186 antaa datan multiplekserin 180 kautta RAM:iin C 182.

Rekisteritiedostoelementit hyväksyvät välittömän datan, IMM (kuten esitetään kuvissa 5a -d), datan muista rekisteritiedostoelementeistä samassa osoiteyksikössä tai datan RAM-muistista. Tästä lähtien sanojen ”välitön data” maininta viittaa kaikissa tapauksissa ohjedekooderin 192 antamaan dataan. Esimerkkisovellutuksessa RAM A 104 antaa datan multiplekserin 100 kautta osoiteyksikköön A 102, RAM B 122 antaa datan multiplekserin 118 kautta osoiteyksikköön B 120 ja RAM C 182 antaa datan multiplekserin 184 kautta osoiteyksikköön C 186. Kukin osoiteyksikkö aikaansaa automaattisen jälkilisäyksen ja jälkivähennyksen sisäisellä lisääjällä/vähentäjällä (ei esitetty). Esimerkkisovellutuksessa osoiteyksikkö 120 B aikaansaa automaattisesti modulo-osoituksen ja kahden halutun rekisteritiedostoelementin (ei esitetty) käytön osoittimina suoramuistihaulle (DMA).

Osoiteyksikkö E 112 on optimoitu kertoimien palauttamiseen. Se sisältää perusrekisterin, joka hyväksyy välittömän datan multiplekserin 110 kautta, ja siirtore-

kisterin, joka hyväksyy välittömän datan multiplekserin 110 kautta tai datan kerääjältä (C0REG tai C1REG 166) multiplekserien 168 ja 110 kautta. Siirtorekisteri aikaansaa automaattisen jälkilisäyksen ja jälkivähennyksen sisäisellä lisääjällä/vähentäjällä (ei esitetty).

IP-laskin ja pino 196 sisältää osoiteosoittimet, jotka suorittavat ROM I:n 194 muistiosoitustoiminteen. Osoitteen sekvenssointia ohjataan käskydekooderilla 192. Osoitedata siirretään joko sisäisesti IP-laskimessa ja pinossa tai hyväksytään välittömänä datana.

Data voidaan siirtää RAM A:sta 104, RAM B:stä 122 tai RAM C:stä 122 ALU:ssa 143 oleviin rekistereihin. Dataa voidaan myös siirtää akusta (C0REG 164 tai C1REG 166) RAM A:han 104, RAM B:hen 122 tai RAM C:hen 182. Dataa voidaan siirtää OREG:stä 162 RAM C:hen 182.

RAM A hyväksyy dataa osoiteyksiköltä A 102 multiplekserin 106 kautta. RAM A 104 myös hyväksyy datan kerääjältä (C0REG tai C1REG 166) multiplekserien 168 ja 110 kautta. RAM B 122 hyväksyy datan osoiteyksiköltä B 120 multiplekserin 124 kautta. RAM B 122 myös hyväksyy datan kerääjältä (C0REG tai C1REG 166) multiplekserien 168 ja 110 kautta. RAM B myös hyväksyy datan DMA_INPUT:sta (kuten esitetään kuvioissa 5a - d) tai INREG:stä 128 multiplekserin 124 kautta. RAM C 182 hyväksyy datan osoiteyksiköltä C 186 multiplekserin 180 kautta. RAM C 182 myös hyväksyy datan kerääjältä (C0REG tai C1REG 166) multiplekserien 168 ja 180 kautta. RAM A 104 antaa datan osoiteyksikköön A 102 multiplekserin 100 kautta ja AREGiin 103 multiplekserin 108 kautta. RAM B 122 antaa datan osoiteyksikköön B 120

multiplekserin 118 kautta, RAMB_OUT:iin (kuten esitetään kuvissa 5a - d), SREG:iin 136 multiplekserin 126 kautta, BREG:iin 134 multiplekserin 116 kautta ja DREG:iin 156 multiplekserin 158 kautta.

AREG 130 hyväksyy välittömän datan, datan ROM E:stä 114 tai datan RAM A:sta 104 multiplekserin 108 kautta. BREG 134 hyväksyy välittömän datan, datan muistista ROM E 114 tai datan muistista RAM B 122 multiplekserin 116 kautta. Lisäksi BREG 134 hyväksyy datan kerääjältä (C0REG 164 tai C1REG 166) multiplekserien 168 ja 116 kautta.

C0REG 164 tai C1REG 166 hyväksyvät datan multiplekserin 148 kautta RAM C 182, summaimelta 146, loogiselta JA-elementiltä 144 tai loogiselta TAI-elementiltä 142.

Siirtoindeksirekisteri SREG 136 hyväksyy välittömän datan tai datan muistista RAM B 122 multiplekserin 126 kautta.

ALU 143 suorittaa kerto-, lisäys-, kerto-keräys-, kerto-lisäys-, kerto-vähennys-, pyöristys-, lisäys-, negaatio-, ja loogiset JA-, TAI-, tai KÄÄNNÄ-operaatiot. Tulot kertojaan 132, AREG:iin 130 ja BREG:iin 134 portitetaan (portitusta ei esitetty), mikä vähentää tehonkulutusta kertojassa 132 varmistamalla, että tulot vaihtuvat vain kun kertolasku on suoritettu. ALU 143 aikaansaa kaksi 36-bittistä kerääjää (C0REG 164 tai C1REG 166) tehokkuutta varten ja kaksi ryhmäsiirrintä, ryhmäsiirrin 140 ja ryhmäsiirrin 150, normalisoinnille. Siirrot aina 18-bitin asemiin vasemmalla tai oikealla annetaan ryhmäsiirtimelle 140 ja ryhmäsiirtimelle 150. Siirtoindeksi spesifioidaan joko nimenomaan vä-

littömän datan kautta tai tarkoitukseen sopivalla siirtoindeksirekisterillä SREG 136 multiplekserin 149 kautta. Siirtoindeksirekisteri SREG 136 yhdessä ryhmäsiirrinten 140 ja 150 kanssa, bittikäänteinen looginen TAI-elementti 160 ja OREG 162 järjestetään ylimenevän minimoimiseksi lohkonormalisoinnin suorituksessa. ALU 143 antaa tilan ohjedekooderille 192 mahdollistaen ehdolliset siirtymät perustuen C0REG:n 164 ja/tai C1REG:n 166 aritmeettiseen tai loogiseen tilaan. Esimerkiksi esimerkkisovellutuksessa C0REG 164 ja C1REG arvojen merkkejä verrataan ehdollisen hypyn aikaansaamiseksi merkin muutokselle. Hyppy syntyy, kun välitön data annetaan IP-laskimeen ja pinoon 196. Kerääjän ylivuoto ja alivuoto tunnistetaan ja saturaatio suoritetaan automaattisesti antamalla heksadesimaaliarvo 0x7FFFFFFF ylivuototapauksessa ja 0x80000001 alivuototapauksessa kahden komplementti aritmetiikan mukaisesti.

Ohjesekvenssi on hae, dekoodaa, suorita. Osoitearvo annetaan IP-laskimella ja pinolla 196 ROM I:n 194 käskemiseksi, joka vastauksena antaa ohjeet ohjedekooderille 192. Ohjedekooderi 192 vastauksena näihin tuloohjeisiin dekoodaa ohjeen ja antaa ohjaussignaalin sopiville elementeille DSP-ytimessä 4 ohjeen suorittamiseksi.

Sopiva silmukkalaskin ja pino 190 yhdessä IP-laskimen ja pinon 196 kanssa antavat pienen ylimenevän sisäkäiset alirutiinikutsut ja sisäkkäiset silmukat. Ohjeen haku estetään yksittäisen ohjesilmukan ajaksi, mikä vähentää tehonkulutusta. Silmukkalaskin ja pino 190 hyväksyy välittömän datan multiplekserin 188 kautta kiinteämittaisten silmukoiden suorittamiseksi. Lisäksi silmukkalaskin ja pino 190 hyväksyy datan kerää-

jältä (C0REG 164 C1REG 166) multiplekserien 168 ja 188 kautta vaihtuvamittaisten silmukoiden suorittamiseksi. 256-sanainen pysyvä ohjevälimuisti (ei esitetty) ROM I:ssä 194 aikaansaa pienitehoisen ohjehaun useimmin suoritetuille silmukoille ja alirutiineille. ODOTA-ohje estää ohjehaun ja ohjekoodauksen tapahtuman kestäessä, mikä vähentää tehonkulutusta. Esimerkkeihin sellaisista tapahtumista voi kuulua DMA-siirto, ajoituspoiminta PCM-rajapinnasta 2 tai ulkoinen tapahtuma.

Ulkoinen data ja ohjaus annetaan DSP-ytimeen 4 PORT_INPUT:n kautta (kuten esitetään kuvioissa 5a - d), DMA_INPUT:iin PCM-rajapinnasta 2 ja pysyvät testibitit, joita käytetään ehdollisissa hyppyohjeissa. Data annetaan ulkoisesti DSP-ytimellä 4 CREGin (kuten esitetään kuvioissa 5a - d ja 6a - b) ja RAMB_OUT:n kautta. DMA DSP-ytimen 4 ja PCM-rajapinnan 2 välillä suoritetaan tunnetulla jakso-otolla. Data C0REG:stä 164 tai C1REG:stä 166 annetaan multiplekserin 168 kautta yhdessä OUTREG_EN:n kanssa CREG:n (kuten esitetään kuvioissa 5a - d ja 6a - b) signaalin ohjedekooderilta kanssa. Aktiivinen OUTREG_EN-signaali merkitsee voimassaolevan minimointiprosessorille annetun CREG-datan läsnäoloa.

Minimointiprosessori 6, joka esitetään kuvioissa 6a - b, auttaa äänentaso- ja koodikirjaetsinnän laskennallisesti raskaissa osissa. Suorittaakseen minimointiproseduurin, minimointiprosessori 6 vastaanottaa sekvenssin havainnollisesti painotettuja tulopuheen näytteitä, joukon vahvistusarvoja ja joukon syntetisoituja puhenäytesekvenssejä DSP-ytimeltä 4. Minimointiprosessori 6 laskee syntetisoidun puheen autokorrelaation ja ristikorrelaation syntetisoidun puheen ja havainnollisesti painotettujen tulopuheen näytteiden välillä.

Näistä korrelaatioista määritetään neliöllinen keskivirhe (MSE) syntetisoidun puheen ja tulopuheen välillä syntetisoidun puheen vahvistuksen ja indeksin funktiona. Minimointiprosessori 6 raportoi indeksin ja vahvistuksen, mikä johtaa MSE:n minimiin. Tehonsäästöominaisuudet keskeyttävät MSE-laskennat, kun lisäminimointi ei ole mahdollista. Minimointiprosessori 6 kommunikoi DSP-ytimen kanssa CREG:in kautta, portti I/O:n, ja sopivat DSP-ytimen ohjeet.

Minimointiprosessorin 6 toiminta määritetään ohjauksella 220. Ohjaus 220 käsittää laskimen indeksiarvojen seuraamiseksi, rekisterit optimaalisen äänentaso- tai koodikirjaetsintätuloksen pitämiseksi, osoitteenmuodostuspiirin RAM X 212 osoittamiseksi ja tulo-/lähtöpiiristön. Lisäksi ohjauselementti 220 on vastuussa valintasignaalien ohjauksesta multiplekserеillä 224, 234, 230 ja 246 ja kytkee rekisterit 210, 214, 226, 228, 236, 238, 244 ja 250. Ohjaus 220 myös seuraa eri arvoja minimointiprosessorin 6 elementeissä, ohjaa tehonsäästötoimintoja, jotka kaventavat hakuja tiettyjen ennalta määrättyjen hakukeskeytysolosuhteiden alaisena ja ohjaa vahvistusarvojen kierrätystä kierrätyspuskurissa 259.

Edelleen ohjaus 220 on vastuussa tulo-/lähtötoiminteiden suorittamisesta. Ohjaus 220 on vastuussa minimointitulosten antamisesta DSP-ytimelle (t.s. paras äänentasoviive ja äänentasovaihtelu tai paras koodikirjaindeksi ja koodikirjavahvistus määritettynä vastaavissa hauissa) tuloporttien 12 kautta. OUTREG_EN-signaali annetaan ohjauselementtiin 220 rekisterille 210 annetavan datan voimassaolon ja läsnäolon kerääjäsignaalissa CREG:ssä osoittamiseksi. Ohja-

us 220 vastauksena generoi kytkentäsignaalin ja antaa sen rekisteriin 210 datan vastaanottamiseksi.

OUTPORT_EN (kuten esitetään kuvioissa 5a - d ja 6a - b) ja PORT_ADD (kuten esitetään kuvioissa 5a - d ja 6a - b) signaalit annetaan ohjauselementtiin DSP-ytimeltä 4. PORT_ADD-signaali antaa osoitteen minimointiprosessorille 6. Minimointiprosessori 6 hyväksyy datan CREG:ltä kun PORT_ADD spesifioi datan minimointiprosessorille 6 ja OUTPORT_EN osoittaa voimassaolevan PORT_ADD arvon. Ohjaus ja data annetaan minimointiprosessorille 6, kuten yllä kuvataan.

Viitataan kuvioon 1, joka on esimerkkilohkodiagrammi esillä olevan keksinnön mukaisesta arkkitehtuurista. PCM-rajapinta 2 vastaanottaa ja antaa koodekkiin (ei esitetty) pulssikoodimoduloidun (PCM) puhenäytedatan, jotka esimerkkisovelluksessa ovat μ-laki tai A-laki kompandoitua näytedataa tai lineaarista näytedataa. PCM-rajapinta 2 vastaanottaa ajoitustiedon kellogeneraattorilta 10 ja vastaanottaa data- ja ohjaustiedon mikroprosessorirajapinnalta 8.

PCM-rajapinta 2 antaa DSP-ytimeen 4 PCM-puhenäytedatan, jonka se vastaanotti koodekilta (ei esitetty) koodattavaksi. PCM-rajapinta 2 vastaanottaa DSP-ytimeltä 4 puhenäytedatan, joka annetaan koodekille (ei esitetty). PCM-data siirretään DSP-ytimen 4 ja PCM-rajapinnan välillä DMA:n kautta. PCM-rajapinta 2 antaa ajoitustiedon kellogeneraattorille perustuen koodekilta (ei esitetty) vastaanotettujen näytteiden ajoitukseen.

DSP-ydin 4 antaa datan ja ohjaustiedon yhteisprosessorilleen, minimointiprosessorille 6. Lisäksi DSP-ydin 4 antaa datan lähtöporttiin 14 ja vastaanottaa datan tuloporteista 12. DSP-ydin 4 vastaanottaa ajoitustiedon kellogeneraattorilta 10. Lisäksi DSP-ydin 4 kykenee antamaan ulkoista osoitetietoa ja vastaanottamaan ulkoisia ohjeita ja dataa.

Minimointiprosessori 6 vastaanottaa ajoitustiedon kellogeneraattorilta ja vastaanottaa datan ja ohjauksen DSP-ytimeltä 4. Minimointiprosessori 6 antaa minimointiproseduurien tulokset DSP-ytimeen 4 tuloporttien 12 kautta.

Kellogeneraattori 10 antaa ajoitustiedon kaikkiin muihin lohkoihin. Kellogeneraattori 10 vastaanottaa ulkoiset kellosignaalit ja vastaanottaa ajoitustiedon mikroprosessorirajapinnasta 8 ja PCM-rajapinnasta 2.

Ryhmätestitoimintaryhmä (JTAG) -rajapinta 16 antaa mahdollisuuden testata ASIC:in toiminta. JTAG-rajapinta 16 vastaanottaa ulkoisen datan ja ohjaustiedon ja antaa ulkoista dataa.

Lähtöportit 14 vastaanottavat dataa DSP-ytimeltä 4 ja antavat tämän datan mikroprosessori rajapintaan 8 ja voivat myös antaa datan ulkoisiin laitteisiin (ei esitetty).

Tuloportit 12 vastaanottavat dataa mikroprosessori rajapinnasta 8 ja minimointiprosessorilta 6 ja antavat tämän datan DSP-ytimeen 4. Tuloportit 12 voivat myös

vastaanottaa dataa ulkoisilta laitteilta (ei esitetty) ja antaa tämän datan mikroprosessorirajapintaan 8.

Mikroprosessorirajapinta 8 vastaanottaa ja antaa mikroprosessoriin (ei esitetty) dataa ja ohjaustietoa. Tämä tieto annetaan muille lohkoille.

Esillä olevan keksinnön esimerkkisovellutuksessa vokooderi-ASIC suorittaa vaihtuvanopeuksisen CELP-algoritmin, joka esitetään yksityiskohtaisesti samaan aikaan vireillä olevassa US-patenttihakemuksessa sarjanro 08/004,484, jätetty tammikuun 14. 1993, nimeltään "Variable Rate Vocoder" ja siirretty esillä olevan keksinnön haltijalle.

Kuvio 2 esittää ASIC:issa suoritetut päätoiminteet. Viitaten kuvioon 2, koodattavat näytteet annetaan vokooderi-ASICiin PCM- rajapinnan 30 kautta koodekilta (ei esitetty). Seuraavaksi näytteet annetaan dekompandoivaan elementtiin 32, joka muuntaa μ-laki- tai A-laki-näytteet lineaarisiksi näytteiksi. Lineaarisessa muodossa annetut näytteet annetaan edelleen dekompandoivan elementin 32 kautta muuttumattomina. Lineaariset näytteet annetaan lähetysäänen prosessointielementtiin 34, joka sisältää toiminnallisesti äänikäyttöisen kytkimen (VOX) 36, äänitasauselementin 38, QCELP-koodauselementin 40 ja äänitaajuus(DTMF)-tunnistuselementin 41. Lähetysäänen prosessointielementti 34 antaa koodatut puhepaketit mikroprosessorirajapinnan 42 kautta mikroprosessoriin (ei esitetty), joka on ulkoinen ASIC:iin.

Koodatut puhepaketit annetaan mikroprosessorilta (ei esitetty) mikroprosessorirajapinnan 42 kautta vastaan-

ottavaan äänikäsittelyelementtiin 44, jossa ne dekoodataan puhenäytteiksi. Vastaanottava äänikäsittelyelementti 44 käsittää toiminnallisesti, QCELP-koodauselementin 46, äänitasaimen 48 ja DTMF-generointielementin 47. Dekoodatut näytteet annetaan kompandoivaan elementtiin 50, joka muuntaa lineaariset näytteet μ-laki- tai A-laki-muotoon tai antaa lineaariset näytteet muuttumattomina PCM-rajapintaan 30. ASIC antaa dekoodatut puhenäytteet PCM-rajapinnan kautta koodekkiin (ei esitetty), joka on ulkoinen ASIC:iin.

Dekompandointioperaatio, joka esitetään kuviossa 2 dekompandoivana elementtinä 32 ja kompandointioperaatio, joka esitetään kuviossa 2 kompandoivana elementtinä 50 suoritetaan DSP-ytimellä 4, mikä esitetään kuvioissa 5a - d. Lähetysäänen prosessointioperaatiot, jotka esitetään kuviossa 2 lähetysäänen prosessointielementtinä 34 suoritetaan DSP-ytimellä 4 ja minimointiprosessorilla 6, mikä esitetään kuvioissa 6a - b. Äänen vastaanotto-operaatiot, jotka esitetään kuviossa 2 äänen vastaanottoelementtinä 44, suoritetaan DSP-ytimellä 4, mikä esitetään kuvioissa 5a - d.

Esimerkkisovelluksessa koodekilta (ei esitetty) annetut näytteet 8-bittisessä μ-laki- tai 8-bittisessä A-laki-muodossa muunnetaan 14-bittiseen lineaariseen muotoon. μ-lain ja lineaarisen välinen yhteys esitetään yhtälössä 1 alla:

$$2Y = -1^{S}(33+2M)2^{N} - 33 \qquad (1)$$

missä Y on lineaarinen arvo (-4015.5 - 4015.5), N on eksponentti (0 - 7), M on voimakkuus (0 - 15) ja S on

merkki (0 positiiviselle; 1 negatiiviselle). A-lain ja lineaarisen välinen yhteys esitetään yhtälöillä 2 ja 3 alla:

$$2Y = -1^{S}(1+2M)^{2}2Y \qquad \text{N=0:lle} \qquad (2)$$

$$2Y = -1^{S}(33+2M)2^{N} \qquad \text{N=1,...,7:lle} \qquad (3)$$

missä Y on lineaarinen arvo (-4032 - 4032), N, M, S kuten yllä.

Viitaten kuvioihin 5a - d, kuvion 2 PCM-rajapinnan 30 kautta annetut näytteet muunnetaan lineaariseen muotoon etsintätaulukkovälineillä, jotka on tallennettu ROM E:hen 114. Edullisessa sovellutuksessa puolenkokoisia 128x14 μ-laki-lineaari- ja A-laki-lineaarietsintätaulukkoja käytetään muunnoksen suorittamiseen. Edullisessa sovelluksessa hyödynnetään täyskokoisia muunnostaulukoita, joilla on yhtälössä 4 esitetyt ominaisuudet.

$$ROM(n+128) = -ROM(n), \qquad 0 \le n \le 127 \qquad (4)$$

Kaikkien DC-komponenttien poistaminen tulopuhesignaalista vaaditaan ennen autokorrelaatiokertoimien ja LPC-kertoimien laskentaa. DC:n poisto-operaatio tehdään DSP-ytimessä 4 vähentämällä alipäästösuodatettu puhenäytekeskiarvo, DC-ennakko, kustakin tulonäytteestä ikkunassa. Näin ollen DC-ennakko nykyiselle kehykselle on painotettu keskiarvo nykyisen ja edellisen kehyksen näytekeskiarvosta. DC-ennakon laskenta esitetään yhtälöllä 5 alla:

$$DC\_ennakko = a(keskimääräinen(aikaisempi\_kehys)) + (1-a)keskimääräinen(nykyinen\_kehys) \quad (5)$$

missä a=0.75 esimerkkisovellutuksessa. Alipäästösuodatusta käytetään suurten epäjatkuvuuskohtien estämiseksi kehysten rajoilla. Tämä operaatio suoritetaan DSP-ytimessä 4 tallettamalla näytekeskiarvo nykyiselle kehykselle ja aikaisemmalle kehykselle yhdessä RAM-elementeistä (t.s. RAM A 104, RAM B 122 tai RAM C 182) ROM E 114 saadun interpolointitekijän kanssa. Lisäys suoritetaan summaimella 146 ja kertolasku kertojalla 132. DC:n estofunktio voidaan kytkeä tai estää mikroprosessorin ohjauksessa.

DC-vapaa tulopuhesignaali, s(n), ikkunoidaan puhesekvenssien katkomisen kiinteiksi kehyksiksi vaikutusten pienentämiseksi. Esimerkkisovellutuksessa käytetään Hamming-ikkunointifunktiota. Kehyspituudella L_A = 160 ikkunoitu puhe $s_w(n)$ lasketaan alla yhtälöllä 6 esitetyn mukaisesti:

$$s_w(n) = s(n+60)W_H(n), \quad 0 \le n \le L_A - 1 \quad (6)$$

missä Hamming-ikkuna määritetään yhtälöillä 7 ja 8 alla.

$$W_H(n) = 0.54 - 0.46\cos[2\pi n/(L_A - 1)], \quad 0 \le n \le L_A - 1 \quad (7)$$

$$= 0, \quad \text{muutoin} \quad (8)$$

Edullisessa sovelluksessa, koska $W_H(n)$ on parillissymmetrinen funktio, 80 kertoimen tarkastustaulu, puolet Hamming-ikkunan kertoimista, talletetaan ROM E:hen

114. Seuraavaksi voidaan suorittaa ikkunointi antamalla ikkunakerroin $W_H(n)$ ROM E:stä 114, osoiteyksikön E 112 antaman osoitearvon mukaisesti multiplekserin 108 kautta AREG:iin 130. AREG 130 antaa tämän datan kertojan 132 ensimmäiseen tuloon. Puhenäyte s(n+60) annetaan RAM B:llä 112 BREG:iin 134. BREG 134 antaa tämän datan kertojan 132 toiseen tuloon. Kertojan 132 lähtönä on ikkunoitu puhenäyte, joka annetaan C0REG:iin 164 ja sitten RAM C:hen 182 multiplekserin 168 kautta.

Lohkonormalisointiproseduuri sisältää kaksi toiminnallista osaa: normalisointikertoimen määrittäminen ja aiotun datan normalisointi. Esimerkkisovellutuksessa data talletetaan kahden komplementtinotaatiolla. Ikkunoidut näytteet lasketaan ylläolevan yhtälön 7 mukaan. Normalisointiohjeella DETNORM ROM I:stä 194 saatava ikkunoitu C0REG:ssä 164 muodostettu näytearvo alistetaan seuraavalle proseduurille. Jos arvo C0REG:ssä 164 on negatiivinen, niin inversioelementti 152 invertoi numeron binäärinumerot ja antaa invertoidun arvon summaimen 146 ensimmäiseen tuloon. Summaimen 146 toiseen tuloon annetaan nolla multiplekserien 137 ja 138 kautta. Seuraavaksi summain 146 lisää ykkösen, sen kantotulon kautta (ei esitetty) bittiinvertoituun arvoon. Jos arvo C0REG:ssä 164 on positiivinen, arvo annetaan multiplekserien 168 ja 154 kautta inversioelementtiin 152 ja summaimeen 146 koskemattomana. Tämän proseduurin tarkoituksena on laskea C0REG:in itseisarvo. Itseisarvo sitten annetaan bitittäin loogisen TAI-elementin 160 ensimmäiseen tuloon. Bitittäin loogisen TAI-elementin 160 toinen tulo annetaan OREG:llä 162. Yllä kuvattu itseisarvon laskenta esitetään alla yhtälöissä 8a -c.

$$OREG_{uusi} = (ABS(C0REG))TAI(OREG_{vanha}) \qquad (8a)$$

missä

$$ABS(C0REG) = C0REG, \qquad C0REG \geq 0 \qquad (8b)$$

$$= -C0REG, \qquad C0REG < 0 \qquad (8c)$$

Yllä yhtälöillä 8a - c kuvattu operaatio voidaan vastaavasti suorittaa C1REG:llä 166 käyttäen DETNORM-ohjetta, jos normalisoitavaksi aiottu data on C1REG:ssä 166. OREG 162 tyhjennetään normalisointioperaation alussa. Tämä proseduuri toistetaan kaikille ikkunoiduille näytteille (aiottu data) niin, että operaation lopussa OREG:iin 162 talletettava arvo edustaa kaikkien ikkunoitujen itseisarvojen bitittäin loogista TAI:ta. Eniten merkitsevien bittien joukosta OREG:ssä 162 määritetään skaalauskerroin, koska arvo OREG:ssä 162 on suurempi tai yhtäsuuri kuin suurin arvo ikkunoitujen näytteiden lohkossa. OREG:ssä 162 oleva arvo siirretään multiplekserin 168 kautta RAM C:hen 182. Seuraavaksi arvo ladataan C0REG:iin 164. Normalisointikerroin määritetään laskemalla C0REG:n 164 arvon vasemmalle tai oikealle siirtojen määrä, joka vaaditaan, että ikkunoidun datan siirto tällä määrällä aikaansaa arvot, joilla on haluttu huippuarvo seuraavaa operaatiota varten. Tätä skaalauskerrointa pidetään myös normalisointikertoimena. Koska normalisointi suoritetaan siirtojen kautta, normalisointikerroin on kahden potenssi.

Ikkunoitujen näytteiden tarkkuuden pitämiseksi korkeimmalla mahdollisella tasolla, aiotut arvot kerrotaan normalisointikertoimella niin, että suurin arvo varaa maksimimäärän bittejä, joka on järjestetty seuraavaa operaatiota varten. Koska normalisointikerroin on kahden potenssi, aiotun datan normalisointi voidaan saavuttaa yksinkertaisesti suorittamalla joukko siirtoja normalisointikertoimen määrityksen mukaisesti.

Normalisointikerroin annetaan RAM B:llä 122 multiplekserin 126 kautta SREG:iin 136. Ikkunoidut näytteet annetaan sitten RAM C:stä 182 multiplekserin 158 kautta DREG:iin 156. Tämän jälkeen DREG 156 antaa nämä arvot ryhmäsiirtimelle 150 multiplekserin 154 kautta ja estää invertterin 152, jossa ne siirretään ryhmäsiirtimeen 150 SREG:llä multiplekserin 149 kautta annetun normalisointikertoimen mukaisesti. Ryhmäsiirtimen 150 lähtö annetaan estetyn lisäyselementin 146 ja multiplekserin 148 kautta C0REG:iin 164. Normalisoidut ikkunoidut näytteet siirretään sitten multiplekserin 168 kautta ja annetaan multiplekserien 124 ja 180 kautta RAM B:hen 122 ja RAM C:hen 182, vastaavasti. Tämä johtaa kahteen kopioon normalisoiduista näytteistä RAM:ssa, tehden seuraavan autokorrelaatiokerrointen laskennan tehokkaammaksi.

P+1 autokorrelaatiokerrointen R(k) laskenta suoritetaan allaolevan yhtälön 9 mukaan.

$$R(k) = \sum_{m=0}^{L_A-1-k} s_W(m)s_W(m+k), \qquad 0 \le k \le P \qquad (9)$$

missä P formanttiennakointisuodattimen aste kooderissa. Esimerkkisovellutuksessa P = 10. Normalisoidut ikkunoidut näytteet $s_w(m)$ annetaan RAM B:llä 122 multiplekserin 116 ja BREG:in kautta kertojan 132 ensimmäiseen tuloon. Normalisoitujen ikkunoitujen näytteiden viivästetyt versiot $s_w(m+k)$ annetaan RAM C:llä 182 multiplekserin 148 ja C1REG:n 166, RAM A:n 104, ja AREG:n 130 kautta kertojan 132 toiseen tuloon. Kukin R(k) on L_A - k arvoista kerätty. Kunkin R(k) ensimmäistä iteraatiota varten kertojan 132 lähtö annetaan kerääjään C0REG 164 multiplekserin 138 ja ryhmäsiirtimen 140 kautta ja summaimen 146 ensimmäiseen tuloon.

Multiplekseri 154 antaa nollan summaimen 146 toiseen tuloon estetyn ryhmäsiirtimen 150 kautta. Seuraaville iteraatioille, kertojan 132 lähtö annetaan summaimen 146 ensimmäiseen tuloon multiplekserin 138 ja ryhmäsiirtimen 140 kautta. Summaimen 146 toiseen tuloon annetaan C0REG:n 164 lähtö multiplekserien 168 ja 154, estetyn inversioelementin ja estetyn ryhmäsiirtimen 150 kautta. Tämä proseduuri toistetaan kullekin R(k):lle.

Mikään autokorrelaatiokertoimista ei ole suurempi kuin R(0). Esimerkkisovellutuksessa R(0):n laskennan jälkeen määritetään normalisointikerroin. Seuraavaksi tämä normalisointikerroin annetaan jo lasketulle R(0):lle ja muille autokorrelaatiokertoimille kun ne on laskettu. Autokorrelaatiokertoimien normalisointi tässä vaiheessa maksimoi tarkkuuden LPC-kertoimien seuraavalle laskennalle.

Edeten nyt lohkoon 62 kuvassa 3, LPC-kertoimet lasketaan lyhytaikaisen korrelaation (redundanssien) poistamiseksi puhenäytteistä.

Formanttiennakointisuodattimella, jonka aste on P, on siirtofunktio A(z), joka esitetään allaolevalla yhtälöllä 10.

$$A(z) = 1 - \sum_{i=1}^{P} a_i z^{-1}, \qquad P = 10 \tag{10}$$

Kukin LPC-kerroin a_i lasketaan normalisoidun ikkunoidun tulopuheen autokorrelaatioarvoista. Tehokasta iteratiivistä menetelmää, *Durbinin rekursiota* (katso Rabiner, L.R. ja Schafer, R.W., *"Digital Processing of*

Speech Signals", Prentice-Hall, 1978) käytetään esimerkkisovellutuksessa LPC-kertoimien laskennassa. Tämä iteratiivinen menetelmä kuvataan yhtälöissä 11 - 17 alla.

$$E^{(0)} = R(0), \qquad i = 1 \tag{11}$$

$$k_i \left[R(i) - \sum_{j-1}^{i-1} a_j^{(i-1)} R(i-j) \right] \Big/ E^{(i-1)} \tag{12}$$

$$a_i^{(i)} = k_i \tag{13}$$

$$a_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)}, \qquad 1 \le j \le i-1 \tag{14}$$

$$E^{(i)} = (1 - k_i^2) E^{(i-1)} \tag{15}$$

Jos i<P, niin kasvata i:tä ja jatka 12:een (16)

lopulliset LPC-kerroinarvot ovat :

$$a_j^{'} = a_j^{(P)}, \qquad 1 \le j \le P \tag{17}$$

Durbinin iteratiivinen algoritmi toimii ainoastaan kun tulosignaalilla on nollakeskiarvo, mikä vaatii, että DC-ennakko poistetaan ennen autokorrelaatiolaskennan suorittamista yllä kuvatulla tavalla.

Esimerkkisovellutuksessa 15 Hz:n kaistaleveyden laajennusta käytetään formanttiennustesuodattimen stabiilisuuden varmistamiseksi. Tämä voidaan tehdä skaalaamalla formanttisyntetisointisuodattimen navat asteittain sisäänpäin. Kaistanleveyden laajennus saavutetaan skaalaamalla LPC-kertoimet allaolevan yhtälön 18 mukaan:

$$a_i = \beta^i a_i^{'}, \qquad 1 \le i \le P \tag{18}$$

missä β=0.9883 esimerkkisovellutuksessa. 10 kaistanlaajennuskerrointa β^i, $1 \leq i \leq P$ talletetaan tarkastustaulukkoon, joka on annettu ROM E:ssä 114. Lohkon 62 funktiot suoritetaan DSP-ytimessä 4.

Nyt edetään lohkoon 64 kuvassa 3, koodausnopeus nykyiselle puhenäytekehykselle määritetään. Nopeuspäätös perustuu puheaktiivisuuden mittaukseen kussakin kehyksessä. Puheaktiivisuus vuorostaan mitataan kehysenergian mukaan, R(0), mikä esitettiin aiemmin. Koodausnopeuden määrittämiseksi kehykselle i, kehysenergiaa verrataan kolmeen kynnykseen, jotka määritetään allaolevilla yhtälöillä 19 - 21:

$$T_1(B_i) = -(5.544613 \bullet 10^{-6})B_i^2 + (4.047152)B_i + 362 \quad (19)$$

$$T_2(B_i) = -(1.529733 \bullet 10^{-6})B_i^2 + (8.750045)B_i + 1136 \quad (20)$$

$$T_3(B_1) = -(3.957050 \bullet 10^{-6})B_i^2 + (18.899622)B_i + 3347 \quad (21)$$

missä taustakohinataso B_i kehykselle i päivitetään kussakin kehyksessä, kuten kuvataan alla yhtälöllä 22.

$$B_i = \min\left[R_{i-1}(0), 160000, \max\left[1.00547B_{i-1}, B_{i-1} + 1\right]\right] \quad (22)$$

Nopeus on kahdeksasosa, jos kehysenergia on alle kaikkien kolmen kynnyksen, neljännes, jos kehysenergia on $T_1(B_i)$:n ja $T_2(B_i)$:n välissä ja puolet, jos kehysenergia on $T_2(B_i)$:n ja $T_3(B_i)$:n välissä, ja täysi, jos energia on $T_3(B_i)$:n yläpuolella. Paitsi kun nopeus määrätään mikroprosessorilla (ei esitetty) mikroprosessorirajapinnan 8 kautta tiettyyn nopeuteen, datanopeus voidaan nostaa mihin tahansa nopeuteen, mutta voidaan laskea vain yksi askel kehystä kohden (t.s. puolesta nopeudesta neljäosanopeuteen). DSP-ytimessä 4 näihin

kynnyksiin liittyvät kertoimet annetaan etsintätaulukossa ROM E:ssä 114.

Murtonopeusalgoritmi annetaan myös edullisessa sovelluksessa, mikä mahdollistaa maksimaalisen keskinopeuden rajoittamisen mikroprosessorilla (ei esitetty) mikroprosessorirajapinnan 8 kautta. Annetulla nopeuden rajoituskertoimella S vokooderin maksimaalinen keskinopeus rajoitetaan (2S+1)/[2(S+1)]:een rajoittamalla peräkkäisten täyden nopeuden kehysten määrää. Lohkon 64 toiminnat suoritetaan DSP-ytimessä 4.

Edeten nyt kuvion 3 lohkoon 66, kaistaa laajentavat LPC-kertoimet muunnetaan viivaspektripari(LSP)taajuuksiksi (katso Soong ja Juang "Line Spectrum Pair (LSP) and Speech Data Compression", ICASSP, 1984). LSP-taajuudet on esitetty merkittävästi parantavan lähetys- ja tallennusominaisuuksia ja voidaan koodata tehokkaammin kuin LPC-kertoimet. LSP-taajuudet, jotka edustavat kymmentä LPC-kerrointa ovat yhtälöissä 23 ja 24 esitettyjen polynomien kymmenen juurta.

$$P'(\omega) = \cos 5(2\pi\omega) + p'_1 \cos 4(2\pi\omega) + \ldots p'_4 \cos(2\pi\omega) + 0.5 p'_5 \quad (23)$$

$$Q'(\omega) = \cos 5(2\pi\omega) + q'_1 \cos 4(2\pi\omega) + \ldots + q'_4 \cos(2\pi\omega) + 0.5 p'_5 \quad (24)$$

missä p'_i ja q'_i välillä $1 \leq i \leq 5$ lasketaan DSP-ytimessä 4 allaolevien yhtälöiden 25 - 27 mukaan:

$$p'_0 = q'_0 = 1 \quad (25)$$

$$p'_i = -a_i - a_{11-i} - p'_{i-1} \qquad 1 \leq i \leq 5 \quad (26)$$

$$q'_i = -a_i + a_{11-i} + q'_{i-1} \qquad 1 \leq i \leq 5 \quad (27)$$

P'(ω):n juuret saadaan olettamalla, että siinä on ainakin yksi juuri kunkin π/32 radiaanin välein. P'(ω) arvioidaan jokaisella π/32 radiaanilla 0:sta π radiaaniin. Juuren saaminen π/32 radiaanin välein johtaa funktion P'(ω) merkin vaihtumiseen väliajoin.

Jos juuri saadaan, binäärihaku suoritetaan juuren eristämiseksi alueelle, jonka koko on π/256. Binäärihaku sisältää P'(ω):n laskennan kyseisen alueen keskellä juuren sisältävän puolikkaan selvittämiseksi. Seuraavaksi aluetta kavennetaan ainoastaan siihen puolikkaaseen, joka sisältää juuren. Tämä prosessi jatkuu kunnes juuri on eristetty alueelle, jonka koko on π/256. Lineaarinen interpolointi suoritetaan seuraavaksi juuren paikan selvittämiseksi alueella kooltaan π/256 radiaania.

LSP-taajuuksien järjestysominaisuus varmistaa, että Q'(ω):n yksi juuri sijaitsee kunkin P'(ω):n juuriparin välissä. Q'(ω):n viides juuri on P'(ω):n viidennen juuren ja π radiaanin välissä. Binäärihaku, joka kuvattiin yllä, suoritetaan P'(ω):n kunkin juuriparin välissä ja P'(ω):n viidennen juuren ja π radiaanin välissä Q'(ω):n juurten määrittämiseksi.

Koska binäärihaku jatkuu ja juuren paikka on eristetty π/256:n radiaanin kokoisen alueen sisälle, vaaditaan tasavälein sijoitetut 256 kosiniarvoa 0:n ja π:n välille. Kosiniarvot talletetaan etsintätaulukkoon ROM E:ssä 114. P'(ω):n ja Q'(ω): juurien löytämiseksi vaaditaan funktioiden 48 ja 30 maksimiarvojen määritys. Nämä funktioarvioinnit ovat laskennallisesti vaativin

osa muunnettaessa LPC-vakiot LSP-taajuuksille. Lohkon 66 funktiot suoritetaan DSP-ytimessä 4.

Edeten nyt lohkoon 68 kuviossa 3, LSP-taajuudet kvantisoidaan. Kukin LSP-taajuus keskeistetään siihen liittyvän ennakkoarvon ympärille. Ennen kvantisointia tämä ennakko vähennetään kustakin siihen liittyvästä LSP-taajuudesta kvantisointia varten vaadittavien bittien määrän vähentämiseksi. Tämä ennakko lasketaan kuten on esitetty yhtälössä 28 alla.

$$Ennakko_i = 0.5i/(P+1) = (0.\overline{45})i \qquad (28)$$

Kun ennakko on vähennetty LSP-taajuudesta, LSP-taajuus kvantisoidaan käyttäen differentiaalista pulssikoodimodulaatiokvantisoijaa (DPCM). DPCM-kvantisoijaa käytetään, koska LSP-taajuudet vaihtuvat hitaasti ajan suhteen, ja muutoksen kvantisointi LSP-taajuuksissa vähentää kvantisointiin vaadittavien bittien määrää verrattuna määrään, joka vaaditaan LSP-taajuuksien suoraan kvantisointiin. DSP-ytimessä 4 LSP-taajuuden ennakkoarvot talletetaan etsintätaulukkoon ROM E:ssä 114. Käytettyjen kvantisointibittien määrä ja kvantisointiaskeleen koko ovat funktioita joista 10 LSP-taajuutta kvantisoidaan ja ovat koodausnopeuden funktioita. Kvantisointibittien määrä ja kvantisointiaskeleen koko talletetaan etsintätaulukkoon ROM E:ssä 114 kullekin LSP-taajuudelle kullakin koodausnopeudella. Ylläkuvattu tarkastustaulukko esitetään taulukossa II. Esimerkiksi LSP-taajuus ω_1 täydellä nopeudella kvantisoidaan neljällä bitillä ja askelkoolla 0.025.

Taulukko II

LSP	Täysino-peus	Puoli-nopeus	Neljä-osanope-us	Kahdeksas-osanopeus
ω_1	4:±0.025	2:±0.15	1:±0.01	1:±0.01
ω_2	4:±0.04	2:±0.15	1:±0.01	1:±0.01
ω_3	4:±0.07	2:±0.03	1:±0.01	1:±0.01
ω_4	4:±0.07	2:±0.03	1:±0.01	1:±0.01
ω_5	4:±0.06	2:±0.02	1:±0.01	1:±0.01
ω_6	4:±0.06	2:±0.02	1:±0.01	1:±0.01
ω_7	4:±0.05	2:±0.02	1:±0.01	1:±0.01
ω_8	4:±0.05	2:±0.02	1:±0.01	1:±0.01
ω_9	4:±0.04	2:±0.02	1:±0.01	1:±0.01
ω_{10}	4:±0.04	2:±0.02	1:±0.01	1:±0.01

Kvantisoinnin ja interpoloinnin jälkeen suoritetaan testi, jolla varmistetaan, että formanttisuodatin on stabiili kvantisoinnin vaikutusten jälkeen. LSP-taajuudet on erotettava ainakin 80 Hz:llä saatavan formanttisuodattimen stabiliteetin varmistamiseksi. Jos jokin LSP-taajuuksista erotetaan alle 80 Hz:n taajuudella viereisestä LSP-taajuudesta, suoritetaan kaistan laajennus uudelleen. Lohkon 68 funktiot suoritetaan DSP-ytimessä 4.

Edeten nyt lohkoon 70 kuvassa 3, LSP-taajuudet alipäästösuodatetaan kvantisointivaikutusten vähentämiseksi yhtälössä 29 esitettävällä tavalla.

$$\hat{\omega}_i(nykyinen) = SM\hat{\omega}_i(aikaisempi) + (1-SM)\omega_i(nykyinen) \qquad (29)$$

missä esimerkkisovellutuksessa SM=0 täydellä nopeudella, SM=0.125 puolella nopeudella. Neljännes- ja kahdeksasosanopeudella, jos peräkkäisten neljäsosa- tai kahdeksasosanopeuden kehysten määrä on alle 10, SM=0.125 ja SM=0.9 muutoin.

LSP-taajuudet interpoloidaan kullekin äänentasoalikehykselle. Äänentasoalikehykselle interpoloituja LSP-taajuuksia käytetään vastaavalle parille koodikirja-alikehyksiä, paitsi kahdeksasosanopeudella. LSP-taajuudet interpoloidaan yhtälön 30 mukaisesti alla.

$$\hat{\omega}_i(nykyinen) = \alpha_\omega \hat{\omega}_i(aikaisempi) + (1-\alpha_\omega)\hat{\omega}_i(nykyinen) \qquad (30)$$

missä painot α_ω talletetaan etsintätaulukkoon ROM E:ssä 114 kullekin äänentasoalikehykselle ja kullekin koodikirja-alikehykselle kaikilla nopeuksilla. Yllä kuvattu etsintätaulukko esitetään taulukossa III. Esimerkiksi α_ω täydellä nopeudella on 0.75 äänentasoalikehykselle 1 ja koodikirja-alikehyksille 1 ja 2.

Taulukko III

Äänentasoalikehys	Koodikirja-alikehys	Täysinopeus	Puolinopeus	Neljäsosanopeus	Kahdeksasosanopeus
	1				α_ω=0.375
1	1 ja 2	α_ω=0.75	α_ω=0.625	α_ω=0.375	
2	3 ja 4	α_ω=0.50	α_ω=0.125		
3	5 ja 6	α_ω=0.25			

4	7 ja 8	α_ω=1.00			

Interpoloidut LSP-taajuudet muunnetaan takaisin LPC-kertoimiksi käytettäväksi äänentaso- ja koodikirjahauissa. LPC-kertoimet $P_A(z)$:sta ja $Q_A(z)$:sta lasketaan, kuten esitetään alla yhtälöissä 31-33.

$$A(z) = \left[P_A(z) + Q_A(z)\right]/2 \qquad (31)$$

missä:

$$P_A(z) = (1+z^{-1})\prod_{j=1}^{5}\left[1-2z^{-1}\cos(2\pi\omega_{2j-1})+z^{-2}\right] \qquad (32)$$

$$Q_A(z) = (1-z^{-1})\prod_{j-1}^{5}\left[1-2z^{-1}\cos(2\pi\omega_{2j})+z^{-2}\right] \qquad (33)$$

Esimerkkisovellutuksessa Taylorin sarja lasketaan DSP-ytimessä 4 kosiniarvojen laskemiseksi $P_A(z)$ ja $Q_A(z)$. Taylorin sarjan laskenta aikaansaa tarkemmat kosiniarvot kuin mitä aikaisemmin kuvatussa juurihaussa käytetyt. $P_A(z)$:n ja $Q_A(z)$:n arvot lasketaan DSP-ytimessä 4 suorittamalla neliöllisten polynomien konvoluutio, kuten esitetään yhtälöissä 32-33 yllä. Lohkon 70 funktiot suoritetaan DSP-ytimessä 4.

Edeten nyt lohkoon 72 kuviossa 3, suoritetaan vertaileva analyysi syntetisoimalla äänentasohakuoperaatio. Tämä perusteellinen hakuproseduuri esitetään lohkojen 72-74 muodostamalla silmukalla. Äänentasoennustus tehdään esimerkkisovellutuksessa äänentasoalikehyksille kaikilla muilla paitsi kahdeksasosanopeudella. Kuviossa 7 esitetty äänentasokooderi käyttää syntetisointianalysointimenetelmää määrittääkseen äänentasoennustusparametrit (t.s. äänentasoviiveen, L, ja äänentasovahvistuksen, b). Valitut parametrit ovat ne, jot-

ka minimoivat MSE:n havainnollisesti painotetun tulopuheen ja syntetisoidun puheen, joka on generoitu käyttäen äänentasoennustusparametreja, välillä.

Esillä olevan keksinnön edullisessa sovelluksessa käytetään ehdotonta havainnollista painotusta äänentasoennustusparametrien palauttamisessa, kuten esitettynä kuviossa 7. Kuviossa 7, havainnollinen painotussuodatin vasteineen esitetään yhtälöllä 34 alla.

$$W(z) = \frac{A(z)}{A(z/\zeta)} \qquad (34)$$

toteutetaan suodattimien 320 ja 324 kaskadina. Ehdoton havainnollinen painotus vähentää laskennallista kompleksisuutta havainnollisessa painotussuodattimessa käyttämällä suodattimen 320 lähdön uudelleen avoimen silmukan formanttijäännöksenä. Tämä suodattimen yhtälön 34 jakaminen kahteen osaan eliminoi yhden suodatinoperaation äänentasohaussa.

Tulopuhenäytteet s(n) siirretään läpi formanttiennustussuodattimen 320, jonka kertoimet ovat LSP-interpoloinnin ja aikaisemmin kuvatun kuvion 3 lohkon 70 LSP-LPC-muunnoksen seurauksena saatavat LPC-kertoimet. Formanttiennustussuodattimen 320 lähtö on avoimen silmukan formanttijäännös, $p_o(n)$. Avoimen silmukan formanttijäännös, $p_o(n)$, siirretään painotetun formanttisynteesisuodattimen 324 läpi alla yhtälössä 35 esitetyn siirtofunktion avulla.

$$H(z) = \frac{1}{A(z/\zeta} = \frac{1}{1 - \sum_{i=1}^{10} a_i (z/\zeta)^{-1}}, missä\ \zeta = 0.8 \qquad (35)$$

Painotetun formanttisynteesisuodattimen 324 lähtö on havainnollisesti painotettua puhetta, x(n). Painotetun formanttisynteesisuodattimen 324 suodattimen alkutilan tai suodattimen muistin vaikutus poistetaan vähentämällä painotetun formanttisynteesisuodattimen 324 nollatulovaste (ZIR) painotetun formanttisynteesisuodattimen 324 lähdöstä. Painotetun formanttisynteesisuodattimen 324 ZIR:n laskenta suoritetaan ZIR-elementissä 328. ZIR, $a_{ZIR}(n)$, vähennetään havainnollisesti painotetusta puheesta, x(n), summaimessa 326. Kunkin äänentasoalikehyksen alussa ZIR-elementin 328 ja painotetun formanttisynteesisuodattimen 324 suodatinmuistit ovat identtiset.

Äänentasohaussa nollaosuus koodikirjavektorista oletetaan senhetkiselle alikehykselle avoimen silmukan formanttijäännöksen laskennassa. Äänentasohaku suoritetaan käyttäen sekä aikaisemmin kuvattua avoimen silmukan äänentasojäännöstä, $p_o(n)$, että suljetun silmukan formanttijäännöstä, $p_c(n)$. Suljetun silmukan formanttijäännökset, $p_c(n)$, ovat äänentasosynteesisuodattimessa 322 edellisten äänentasoalikehyksien aikana uudelleenmuodostettuja näytteitä. Äänentasosynteesisuodattimen 322 siirtofunktio esitetään yhtälössä 36 alla.

$$\frac{1}{P(z)} = \frac{1}{(1 - bz^{-L})} \qquad (36)$$

missä äänentasoviive L ja äänentasovahvistus b lasketaan äänentasohakuproseduurien aikana aikaisemmille äänentasoalikehyksille. Äänentasosynteesisuodattimen 322 tulo on herätevektori, $C_b(n)$, joka generoidaan kertomalla koodikirjasyöte, $C_b(I)$, indeksille I ja koodikirjavahvistus G, joka lasketaan koodikirjahakuproseduurien aikana edellisille koodikirja-

alikehyksille. Näin saatua formanttijäännöstä kutsutaan suljetun silmukan formanttijäännökseksi.

Esimerkkisovellutuksessa $p_c(n)$ muodostuu 143 suljetun silmukan uudelleenmuodostetusta näytteestä. Haku siirtyy laajalti avoimen silmukan hausta laajalti suljetun silmukan hakuun äänentasoalikehyksen mittaisella ikkunalla. L_P:n mittaiselle äänentasoalikehykselle n vaihtelee välillä $-L_{MAX}$ = 143 ja L_P = -17. Esimerkkisovellutuksessa L_P = 40 täydellä nopeudella, L_P = 80 puolella nopeudella ja L_P = 160 neljännesnopeudella. Esimerkkisovellutuksessa näytteet $p_c(n)$ ja $p_o(n)$ talletetaan vierekkäisesti, kuten esitetään kertojalla 332 mahdollistaen $p_c(n)$ ja $p_o(n)$ arvojen syötön sekventiaalisesti yksittäisenä lohkona näytteitä L_P+143 mittaisina. Näytteet talletetaan RAM B:hen 122.

Formanttijäännös p(n) muodostuu arvoista $p_c(n)$ ja $p_o(n)$, ja se siirretään läpi painotetun formanttisynteesisuodattimen 330, jonka siirtofunktio esitetään yhtälössä 37 alla.

$$H(z) = \frac{1}{A(z/\zeta)}, missä\ \zeta = 0.8 \qquad (37)$$

Painotettua formanttisynteesisuodatinta 330 käytetään painotettujen syntetisoitujen puhenäytteiden, $y_L(n)$, sekvenssin L_P generoimiseen kullekin äänentasoviiveen L arvolle. Painotettu syntetisoitu puhenäytesekvenssi, $y_L(n)$, ja painotettu puhenäytesekvenssi, $x_p(n)$, annetaan minimointiprosessoriin 334. Koska painotetun formanttisynteesisuodattimen 324 alkutilan vaikutukset on vähennetty, painotetun formanttisynteesisuodattimen 330 impulssivaste on laskettava. $Y_L(n)$:n laskenta tuo

mukanaan painotetun formanttisynteesisuodattimen 330 kiertävän impulssivasteen sopivan alisekvenssin kanssa p(n):ssä painotettujen syntetisoitujen puhenäytteiden, $y_L(n)$, saamiseksi äänentasoviiveelle L.

Konvoluutio suoritetaan rekursiivisella tavalla DSP-ytimessä 4, kuten esitetään kuvioissa 5a - d. Joukko $y_L(n)$:n arvoja lasketaan kullekin äänentasoviivearvolle L=17 - L=143. Painotetun formanttisynteesisuodattimen 330 impulssivaste, h(n), lävistetään ensimmäisiin 20 näytteeseen esimerkkisovellutuksessa ja talletetaan RAM A:han 104. Formanttijäännös, p(n), talletetaan RAM B:hen 122. Ensimmäisen äänentasoviiveen L=17 konvoluutio suoritetaan ei-rekursiivisella tavalla kuten esitetään alla yhtälössä 38.

$$y_{17}(n) = h(n) * P_{17}(n) = \sum_{i=0}^{19} h(i)\, p_{17}(n-i), \quad 0 \le n < L_p \qquad (38)$$

missä L_p on äänentasoalikehyksen mitta.

Ensimmäinen sekvenssi $y_{17}(n)$ lasketaan ja talletetaan RAM C:hen 142. Seuraavat sekvenssit, $y_L(n)$, äänentasoviiveille L=18 - L=143 lasketaan rekursiivisesti, kuten esitetään yhtälöissä 39-41 alla.

$$y_L(n) = h(0)\, p(-L) \qquad n = 0 \qquad (39)$$

$$y_L(n) = y_{L-1}(n-1) + h(n)\, p(-L) \qquad 1 \le n < 20, \quad 17 < L \le 143 \qquad (40)$$

$$y_L(n) = y_{L-1}(n-1) \qquad 20 \le n < L_p \qquad (41)$$

Huomaa alla yhtälössä 42 esitetty riippuvuus.

$$p_L(n) = p(n-L) = p_{L-1}(n-1), \qquad 17 < L \le 143, \ 0 \le n < L_p \qquad (42)$$

Rekursiivisen konvoluution tehokkuutta optimoidaan jakamalla RAM:t kolmeen osaan, joista kuhunkin kuuluu sopiva osoiteyksikkö lataus- ja tallennusoperaatioiden ohjaamiseksi. Tätä pidetään kolmiosaisena RAM:na. Konvoluutioarvojen laskenta yhtälössä 40 ja tuloksen muodostaminen kussakin kellojaksossa on siten mahdollista. Esimerkiksi yhdessä kellojaksossa $y_{18}(10)$ lasketaan, $y_{18}(9)$ talletetaan, $y_{17}(10)$ haetaan ja h(10) haetaan. Näin ollen yhtälö 40 voi tuottaa tuloksen kussakin jaksossa. Tuloksen tuottaminen yhtälöstä 41 kussakin jaksossa on myös mahdollista. Esimerkiksi yhdessä jaksossa $y_{18}(24)$ lasketaan, $y_{17}(24)$ haetaan ja $y_{18}(23)$ talletetaan.

Kyky suorittaa yhtälöt 40 ja 41 ilman tarvetta siirtää aikaisemmin laskettuja puhesynteesinäytteitä $y_{L-1}(n-1)$ samaan lähde-RAM:iin kunkin äänentasoviivepäivityksen yhteydessä vaatii tallennusstrategian ja laitteistokyvyn, jota pidetään ping-pongina, jossa lähde- ja kohde-RAM:eja vaihdellaan kunkin viivepäivityksen yhteydessä. Laskettaessa $y_L(n)$:n arvoja parillisille äänentasoviivearvoille, $y_{L-1}(n-1)$, arvot haetaan ensimmäisestä kolmesta RAM:stä ja tulokset talletetaan toiseen kolmesta RAM:stä. Laskettaessa $y_L(n)$:n arvoja parittomille äänentasoviivearvoille, $y_{L-1}(n-1)$, arvot haetaan toisesta kolmesta RAM:stä ja tulokset talletetaan ensimmäiseen kolmesta RAM:stä. Tämä ping-pong eliminoi tarpeen siirtää uutena lasketut $y_L(n)$:n arvot samaan RAM:ään ja aikaisemmin lasketut $y_{L-1}(n-1)$:n arvot siitä pois kunkin äänentasoviivepäivityksen välillä.

Kuvio 8 on vuokaavio konvoluutio-osuuden suorittamisesta (sekä alkuperäiselle ei-rekursiiviselle että rekursiiviselle konvoluutiolle) esimerkkisovellutuksen äänentasohaussa. Lohkossa 350 äänentasoviive L alustetaan alimpaan arvoonsa, joka esimerkkisovellutuksessa on 17. Näytenumero n ja suodatinindeksi m asetetaan

nollaksi ja arvot $y_{17}(n)$ asetetaan nollaksi. Lohkot 352 - 360 muodostavat alkukonvoluutiosilmukan, joka lasketaan ei-rekursiivisella tavalla. Alkukonvoluution laskenta, jossa L=17, suoritetaan allaolevan yhtälön 43 mukaisesti.

$$y_{17}(n) = \sum_{i=0}^{19} h(i)\, p_{17}(n-i), \quad 0 \le n < L_p \qquad (43)$$

Alkukonvoluution laskennassa käytetään kiinteänmittaisia silmukoita laskennallisen kompleksisuuden vähentämiseksi. Tällä tavalla vältetään ylimääräisyys, joka vaaditaan vaihtuvamittaisen silmukkarakenteen muodostamiseksi yhtälön 43 sisemmässä silmukassa (lohkot 356-360). Kukin arvo $y_{17}(n)$ lähetetään minimointiprosessorille laskennan jälkeen.

Lohko 352 testaa näyteindeksin n. Jos n vastaa äänentasoalikehyksen pituutta L_p, alkukonvoluutio on täydellinen ja edetään lohkoon 362. Jos lohkossa 352 n on pienempi kuin äänentasoalikehyksen pituus, edetään lohkoon 356. Lohko 356 testaa indeksin m. Jos m on yhtäsuuri kuin suodattimen impulssivasteen pituus, 20 esimerkkisovellutuksessa, iteraatio on suoritettu ja edetään lohkoon 354, jossa m asetetaan nollaksi ja kasvatetaan n:ää. Seuraavaksi vuo palaa lohkoon 352. Jos lohkossa 356 m on pienempi kuin impulssivasteen pituus, edetään lohkoon 360, missä kerätään osittaiset summat. Edetään lohkoon 358, jossa kasvatetaan m:ää ja edetään lohkoon 356.

Lohkoilla 352-360 muodostetun silmukan alkukonvoluutioon liittyvät operaatiot suoritetaan DSP-ytimessä, jossa on annettu sopiva peräkkäinsuoritus tulojen kerää-

miseksi, kuten esitetään lohkossa 360, kullekin kellojaksolle. Seuraavat operaatiot kuvaavat laskennan peräkkäinsuoritusta ja ne tapahtuvat DSP-ytimessä 4 yhden kellojakson aikana. Suodattimen vastearvo h(m+1) haetaan RAM A:sta 104 ja annetaan AREG:iin 130. Formanttijäännösarvo p(n-17) haetaan RAM B:stä 122 ja annetaan BREG:iin 134. Osasumma $y_{17}(n+m+1)$, joka on C0REG:ssä 164 annetaan RAM C:hen 182 multiplekserien 168 ja 180 kautta. Osasumma $y_{17}(n+m+1)$ annetaan RAM C:llä 182 DREG:iin 156 multiplekserin 158 kautta. Arvot h(m) ja p(n-17) AREG:ssä 130 ja BREG:ssä 134, vastaavasti, annetaan kertojaan 132. Kertojan 132 lähtö annetaan multiplekserin 138 kautta ryhmäsiirtimeen 140, joka normalisoi arvon SREG:n 136 multiplekserin 149 kautta antaman skaalausarvon mukaisesti. SREG:ssä 136 olevaa arvoa tarvitaan p(n-17):n sekvenssin normalisointiin. Tämän normalisointikertoimen käyttö p(n-17):n ja h(m):n tuloon johtaa samaa vaikutukseen kuin p(n-17):n normalisointi, koska tulon täysi tarkkuus ylläpidetään ennen normalisointia ryhmäsiirtimessä. Normalisoitu arvo annetaan summaimen 146 ensimmäiseen tuloon. Osasumma, $y_{17}(n+m)$ annetaan DREG:llä 156 multiplekserin 154, estetyn invertterin 152 ja ryhmäsiirtimen 150 kautta summaimen 146 toiseen tuloon. Summaimen 146 lähtö annetaan multiplekserin 148 kautta C0REG:iin 164. Kun indeksi saavuttaa suurimman sallitun arvonsa lohkossa 352, alkukonvoluutio on suoritettu ja RAM C:ssä 182 olevat osasummat ovat nyt konvoluution lopullisia tuloksia.

Kun alkukonvoluutio on suoritettu, edetään lohkoon 362, jossa suoritetaan rekursiivinen konvoluutio jäljelläolevien äänentasoviivearvojen laskemiseksi.

Lohkossa 362 näyteindeksi n asetetaan nollaksi ja äänentasoviiveindeksi L:ää kasvatetaan. Vuo jatkaa lohkoon 364. Lohko 364 testaa L:n. Jos L on suurempi kuin suurin äänentasoviivearvo, 143 esimerkkisovellutuksessa, vuo jatkaa lohkoon 366, jossa äänentasohakuoperaatio keskeytyy. Jos L on pienempi tai yhtäsuuri kuin 143, vuo jatkaa lohkoon 368. Lohko 368 ohjaa edellä kuvattua ping-pong-operaatiota. Lohkossa 368 testataan L:ää sen parillisuuden tai parittomuuden määrittämiseksi. Jos L on parillinen, vuo jatkaa lohkoon 378 (Tapauksena I kuvattu operaatio). Jos L on pariton, niin vuo jatkaa lohkoon 370 (Tapauksena II kuvattu operaatio).

Tapaus I:(äänentasoviiveen L parilliset arvot)

Lohkossa 378 $y_L(0)$ lasketaan yhtälön 39 mukaisesti. Osoiteyksikkö A 102 antaa osoitearvon RAM A:lle 104, joka vastauksena antaa h(0):n multiplekserin 108 kautta AREG:iin 130. Samassa kellojaksossa osoiteyksikkö B 120 antaa osoitearvon RAM B:lle 122, joka vastauksena antaa p(-L):n multiplekserin 116 kautta BREG:iin 134. Seuraavan kellojakson aikana AREG 130 antaa h(0):n ja BREG antaa p(-L):n kertojalle 132, jossa kaksi arvoa kerrotaan ja tulo annetaan multiplekserin 138 kautta ryhmäsiirtimeen 140. Ryhmäsiirrin 140 SREG:n multiplekserin 149 kautta antaman arvon mukaisesti normalisoi tulon ja antaa normalisoidun tulon summaimen 146 ensimmäiseen tuloon. Summaimen 146 toiseen tuloon annetaan nolla multiplekserin 154, estetyn invertterin 152 ja ryhmäsiirtimen 152 kautta. Summaimen 146 lähtö annetaan C0REG:iin 164 multiplekserin 148 kautta. Saman kellojakson aikana $y_{L-1}(0)$ ja h(1) haetaan RAM B:stä 122 ja RAM A:sta 104 ja annetaan DREG:hen 156 ja

AREG:hen 130 multiplekserien 158 ja 108 kautta, vastaavasti.

Lohkossa 380 syntetisoitua puhenäyteindeksiä n lisätään. Ohjauslohkossa 382, jos syntetisoitu puhenäyteindeksi on alle 20, niin edetään lohkoon 384.

Lohkossa 384 uusi $y_L(n)$ arvo lasketaan kussakin kellojaksossa yhtälön 40 mukaan. Sopiva asetus, joka vaaditaan ennen lohkon 384 ensimmäistä iteraatiota arvojen $y_{L-1}(n-1)$ ja $h(n)$ alustamiseksi, saatiin lohkossa 378 yllä kuvatulla tavalla. Sopiva puhdistus vaaditaan myös lohkon 384 viimeisen iteraation jälkeen lopullisen arvon $y_L(19)$ tallettamiseksi.

Lohkon 384 ensimmäisessä iteraatiossa $y_L(0)$, joka on laskettu lohkossa 378, on läsnä C0REG:ssä 164. C0REG 164 antaa $y_L(0)$:n multiplekserien 168 ja 180 kautta RAM C:hen 182 talletettavaksi yhdessä RAM C:hen 182 osoiteyksikkö C:stä 186 annetun osoitteen kanssa. $y_L(0)$ annetaan minimointiprosessorille 334 samalla kun se annetaan RAM C:hen 182.

Lohkossa 384 seuraavat operaatiot suoritetaan yhdessä kellojaksossa. $y_{L-1}(n)$ arvo annetaan RAM B:llä 122 osoiteyksikkö B:n 120 antaman osoitteen mukaisesti multiplekserien 116 ja 158 kautta DREG:iin 156. Impulssivaste $h(n+1)$ annetaan RAM A:lla 104 osoiteyksikkö A:n 102 antaman osoitteen mukaisesti multiplekserin 108 kautta AREG:iin. DREG 156 antaa $y_{L-1}(n-1)$:n multiplekserin 154, estetyn invertterielementin 152 ja ryhmäsiirtimen 150 kautta summaimen 146 ensimmäiseen tuloon. AREG 130 antaa $h(n)$:n ja BREG antaa $p(-L)$:n kertojaan 132, jossa kaksi arvoa kerrotaan ja tulo annetaan kertojalla 132 multiplekserin 138 kautta ryhmä-

siirtimeen 140. Ryhmäsiirrin 140 SREG:n 136 antaman arvon mukaisesti normalisoi tulon ja antaa normalisoidun tuloarvon summaimen 146 toiseen tuloon. Summaimen 146 lähtö annetaan multiplekserin 148 kautta C0REG:iin 164. C0REGin 164 arvo, joka on laskettu edellisessä iteraatiossa, annetaan multiplekserien 168 ja 180 kautta RAM C:hen 182 talletettavaksi ja minimointiprosessoriin 334.

Lohkossa 380 syntetisoitua puhenäyteindeksiä n lisätään. Ohjauslohkossa 382, jos syntetisoidun puhenäytteen indeksi n on yhtäsuuri kuin 20, lasketaan $y_L(19)$ viimeisessä iteraatiossa, annetaan multiplekserien 168 ja 124 kautta RAM B:hen 122, talletettavaksi rengaspuskuriin ja minimointiprosessoriin 334 ennen etenemistä lohkoon 390.

Tapauksen I loppu

Tapaus II:(äänentasoviiveen L parittomat arvot)

Lohkossa 370 $y_L(0)$ lasketaan yhtälön 39 mukaisesti. Osoiteyksikkö A 102 antaa osoitearvon RAM A:han 104, joka vastauksena antaa h(0):n multiplekserin 108 kautta AREG:iin 130. Samassa kellojaksossa osoiteyksikkö B 120 antaa osoitearvon RAM B:hen 122, joka vastauksena antaa p(-L):n multiplekserin 116 kautta BREG:iin 134. Seuraavan kellojakson aikana AREG 130 antaa h(0):n ja BREG antaa p(-L):n kertojalle 132, jossa kaksi arvoa kerrotaan ja tulo annetaan multiplekserin 138 kautta ryhmäsiirtimeen 140. Ryhmäsiirrin 140 SREG:n multiplekserin 149 kautta antaman arvon mukaisesti normalisoi tulon ja antaa normalisoidun tulon summaimen 146 ensimmäiseen tuloon. Summaimen 146 toiseen tuloon an-

netaan nolla multiplekserin 154, estetyn invertterin 152 ja ryhmäsiirtimen 152 kautta. Summaimen 146 lähtö annetaan C0REG:iin 164 multiplekserin 148 kautta. Saman kellojakson aikana $y_{L-1}(0)$ ja h(1) haetaan RAM C:stä 182 ja RAM A:sta 104 ja annetaan DREG:iin 156 ja AREG:iin 130 multiplekserien 158 ja 108 kautta, vastaavasti.

Lohkossa 372 syntetisoidun puhenäytteen indeksiä n lisätään. Ohjauslohkossa 374, jos syntetisoidun puhenäytteen indeksi n on alle 20 niin edetään lohkoon 376.

Lohkossa 376 uusi $y_L(n)$:n arvo lasketaan kussakin kellojaksossa yhtälön 40 mukaan. Sopiva asetus, joka vaaditaan ennen lohkon 376 ensimmäistä iteraatiota arvojen $y_{L-1}(n-1)$ ja h(n) alustamiseksi, saatiin lohkossa 370 yllä kuvatulla tavalla. Sopiva puhdistus vaaditaan myös lohkon 376 viimeisen iteraation jälkeen lopullisen arvon $y_L(19)$ tallettamiseksi.

Lohkon 376 ensimmäisessä iteraatiossa $y_L(0)$, joka on laskettu lohkossa 370, on läsnä C0REG:ssä 164. C0REG 164 antaa $y_L(0)$:n multiplekserien 168 ja 180 kautta RAM B:hen 122 talletettavaksi yhdessä osoiteyksikkö B:n 120 RAM B:hen 122 antaman osoitteen kanssa. $y_L(0)$ annetaan minimointiprosessorille 334 samalla kun se annetaan RAM B:hen 122.

Lohkossa 376 seuraavat operaatiot suoritetaan yhdessä jaksossa. $y_{L-1}(n)$ arvo annetaan RAM C:llä 182 osoiteyksikkö C:n 186 antaman osoitteen mukaisesti multiplekserin 158 kautta DREG:iin 156. Impulssivastearvo h(n+1) annetaan RAM A:lla 104 osoiteyksikkö A:n 102

antaman osoitteen mukaisesti multiplekserin 108 kautta AREG:iin. DREG 156 antaa $y_{L-1}(n-1)$:n multiplekserin 154, estetyn invertterielementin 152 ja ryhmäsiirtimen 150 kautta summaimen 146 ensimmäiseen tuloon. AREG 130 antaa h(n):n ja BREG antaa p(-L):n kertojaan 132, jossa kaksi arvoa kerrotaan ja tulo annetaan kertojalla 132 multiplekserin 138 kautta ryhmäsiirtimeen 140. Ryhmäsiirrin 140 SREG:n 136 antaman arvon mukaisesti normalisoi tulon ja antaa normalisoidun tuloarvon summaimen 146 toiseen tuloon. Summaimen 146 lähtö annetaan multiplekserin 148 kautta C0REG:iin 164. C0REGin 164 arvo, joka on laskettu edellisessä iteraatiossa, annetaan multiplekserien 168 ja 124 kautta RAM B:hen 122 talletettavaksi ja minimointiprosessoriin 334.

Lohkossa 372 syntetisoidun puheen näyteindeksiä n lisätään. Ohjauslohkossa 382, jos syntetisoidun puheen näytteen indeksi n on yhtäsuuri kuin 20, lasketaan $y_L(19)$ viimeisessä iteraatiossa, annetaan multiplekserien 168 ja 124 kautta RAM B:hen 122 talletettavaksi rengaspuskuriin RAM B:ssä 122 ja minimointiprosessoriin 334 ennen etenemistä lohkoon 390.

Tapauksen II loppu

Ennen lohkon 390 ensimmäistä iteraatiota $y_{L-1}(19)$ haetaan rengaspuskurista RAM B:ssä 122 ja ladataan BREG:iin 134, seuraavaksi $y_{L-1}(19)$ siirretään BREG:stä 134 C0REG:iin 164, jonka jälkeen $y_{L-1}(20)$ haetaan rengaspuskurista RAM B:ssä 122 ja ladataan BREG:iin 134.

Lohkossa 390 lasketaan uusi $y_L(n)$ kussakin kellojaksossa yhtälön 41 mukaan. Seuraavat operaatiot suoritetaan yhdessä kellojaksossa. $y_{L-1}(n-2)$ annetaan BREG:illä 134 C0REG:iin 164. $y_{L-1}(n-3)$ haetaan rengas-

puskurista RAM B:ssä 122 ja ladataan BREG:iin 134. C0REG:ssä oleva $y_{L-1}(n-1)$ esitetään minimointiprosessorille 334. Lohkon 390 viimeisen iteroinnin jälkeen, $y_{L-1}(L_P-2)$ poistetaan rengaspuskurista RAM B:ssä 122. Lisäämällä elementti rengaspuskurin kuhunkin äänentasoviiveeseen ja poistamalla elementti rengaspuskurin kustakin äänentasoviiveestä, pidetään rengaspuskurin kokona L_P-19.

Rengaspuskurin toteutus saadaan erityisten osoiterekisterien kautta osoiteyksikössä B 120, joka määrää pisteiden kiertämisen niin, että sekventiaalista muistia voidaan osoittaa automaattisesti rengasmaisesti.

Lohkossa 386, syntetisoidun puhenäytteen indeksiä n lisätään. Ohjauslohkossa 388, jos syntetisoidun puhenäytteen indeksi n on pienempi kuin L_P, edetään uudelleen lohkoon 390. Jos syntetisoidun puhenäytteen indeksi n on yhtäsuuri kuin L_P, niin senhetkiselle äänentasoviiveelle on laskettu kaikki $y_L(n)$ arvot ja palataan lohkoon 362.

Ennen syntetisoitujen puhenäytteiden $y_L(n)$ laskentaa, L_P:n mittainen sekvenssi havainnollisesti painotettuja puhenäytteitä, $X_p(n)$, annetaan minimointiprosessorille 334. Kuten aikaisemmin esitettiin, syntetisoitujen puhenäytteiden $y_L(n)$ L_P:n mittaiset sekvenssit viivearvoille L=17 - L=143 annetaan minimointiprosessorille 334 alku- ja rekursiivisten konvoluutiolaskentojen aikana. Syntetisoidut puhenäytteet annetaan jaksoittain minimointiprosessorille 334. Minimointiprosessori 334 laskee kunkin syntetisoitujen puhenäytteiden $y_L(n)$ sekvenssin autokorrelaation ja ristikorrelaation kunkin syntetisoitujen puhenäytteiden $y_L(n)$ sekvenssin ja havainnollisesti painotettujen puhenäytteiden $x_p(n)$

sekvenssin välillä. Näistä korrelaatioarvoista minimointiprosessori 334 sitten laskee suhteellisen mitan MSE:stä jokaisen syntetisoidun puhenäytesekvenssin, $y_L(n)$, ja havainnollisesti painotettujen puhenäytteiden, $x_p(n)$, välillä. Kullekin äänentasoviiveelle L, lasketaan MSE syntetisoidun puhenäytesekvenssin kaikilla mahdollisilla äänentasovahvistuksilla b. Minimointiprosessori 334 löytää minimi-MSE:n kaikkien äänentasoviivearvojen L yli ja kaikkien mahdollisten äänentasovahvistusarvojen yli ylläpitämällä MSE-minimiä nykyisen äänentasoviiveen L ja nykyisen äänentasovahvistuksen b kautta. Äänentasoviive-estimaatti $\hat{L}$ ja äänentasovahvistusestimaatti-indeksi $\hat{b}$ vastaten MSE-minimiä ylläpidetään myös minimointiprosessorilla 334. Jokaista uutta MSE-arvoa verrataan minimointiprosessorilla ylläpidettyyn MSE:hen. Jos uusi MSE on pienempi kuin minimi-MSE, niin minimi-MSE korvataan uudella MSE-arvolla ja äänentasoviive-estimaatti $\hat{L}$ ja äänentasovahvistusestimaatti-indeksi $\hat{b}$ päivitetään vastaamaan uutta minimiä MSE:tä. MSE-minimi ja vastaava äänentasoviive-estimaatti $\hat{L}$ ja äänentasovahvistusestimaatti-indeksi $\hat{b}$, joita ylläpidetään minimointiprosessorissa 334, alustetaan kussakin äänentasoalikehyksessä käyttäen ensimmäistä negatiivista MSE-arvoa, joka on laskettu äänentasoalikehyksen aikana. Kaikkien äänentasoviivearvojen L ja äänentasovahvistusarvojen b käyttämisen jälkeen, äänentasoviive-estimaatti $\hat{L}$ ja äänentasovahvistusestimaatti-indeksi $\hat{b}$ ovat optimaalinen äänentasoviiveindeksi ja optimaalinen äänentasovahvistusindeksi, vastaavasti senhetkiselle äänentasoalikehykselle. Minimointiprosessori 334 antaa optimaalisen äänentasoviiveen $\hat{L}$ ja optimaalisen äänentasovahvistusindeksin $\hat{b}$ DSP-ytimeen 4. Optimaalinen äänentasoviive $\hat{L}$ ja optimaalinen äänentasovahvistusindeksi $\hat{b}$ ovat ne, jotka johtavat minimi-MSE:hen havainnollisesti painotettujen puhenäytteiden $x_p(n)$ ja painotettujen syntetisoitujen puhenäytteiden $y_L(n)$ vä-

lillä. MSE on äänentasoviiveen L ja äänentasovahvistuksen b funktio, kuten alla yhtälöissä 44 - 46 kuvataan:

$$MSE(L,b) = \sum_{n=0}^{L_p-1}\left(x_p(n) - by_L(n)\right)^2 \quad (44)$$

$$= \sum_{n=0}^{L_p-1}\left(x_p(n)\right)^2 - 2b\sum_{n=0}^{L_p-1}\left(x_p(n)y_L(n)\right) + b^2\sum_{n=0}^{L_p-1}\left(y_L(n)\right)^2 \quad (45)$$

$$= Ex_px_p - 2bEx_py_L + b^2Ey_Ly_L \quad (46)$$

Ex_px_p on havainnollisesti painotettujen puhenäytteiden $x_p(n)$ autokorrelaatio. Ex_py_L on havainnollisesti painotettujen puhenäytteiden $x_p(n)$ ja painotettujen syntetisoitujen puhenäytteiden $y_L(n)$ välinen ristikorrelaatio. Ey_Ly_L on painotettujen syntetisoitujen puhenäytteiden $y_L(n)$ autokorrelaatio.

Havainnollisesti painotettujen puhenäytteiden $x_p(n)$ autokorrelaatio Ex_px_p ei ole äänentasovahvistuksen b funktio. Ex_px_p pysyy vakiona kunkin äänentasoalikehyksen aikana ja siksi se ei vaikuta optimaalisen äänentasoviiveen valintaan. Yhtälöiden 44 - 46 minimointi äänentasoviiveen L ja äänentasovahvistuksen b suhteen on vastaava kuin allaolevan yhtälön 47 minimointi.

$$MSE(L,b) = -2bEx_py_L + b^2Ey_Ly_L \quad (47)$$

Minimointiprosessori 334 laskee painotetun syntetisoidun puhenäytesekvenssin $y_L(n)$ autokorrelaatiot, Ey_Ly_L, ja havainnollisesti painotetun puhenäytesekvenssin $x_p(n)$ ja painotetun syntetisoidun puhenäytesekvenssin $y_L(n)$ väliset ristikorrelaatiot Ex_py_L. Kullekin korre-

laatioparille (Ex_py_L, Ey_Ly_L) minimointiprosessori 334 laskee suhteellisen MSE:n ylläolevan yhtälön 47 mukaan joukolle äänentasovahvistusarvoja b. Korrelaatioiden Ex_py_L, ja Ey_Ly_L laskenta suoritetaan samanaikaisesti minimointiprosessorissa 334. Suhteelliset MSE-arvot lasketaan ja päätökset liittyen MSE-minimointiin tehdään äänentasoviiveelle L, kun taas korrelaatioarvot lasketaan äänentasoviiveelle L + 1.

Kuviot 6a ja 6b esittävät minimointiprosessorin 334 esimerkkisovellutusta. Havainnollisesti painotetut puhenäytteet, $x_p(n)$, annetaan DSP-ytimellä 4 lukkopiiriin 210 talletettavaksi RAM X:ään 212 ohjauksen 220 antaman osoitteen mukaisesti. Skaalatut äänentasovahvistusarvot, -2b, arvolla b=0.25 - b=2.0, 0.25 askeleen välein annetaan multiplekserin 260 kautta DSP-ytimellä 4 talletettavaksi lukkopiireihin 264, 268, 272, 276, 280, 284, 288 ja 292 vastaavasti. Vastaavat skaalatut äänentasovahvistusarvot, b^2, annetaan multiplekserin 260 kautta DSP-ytimellä 4 talletettavaksi lukkopiireihin 262, 266, 270, 274, 278, 282, 286 ja 290, vastaavasti. Multiplekseri 260 antaa arvot suoraan lukkopiiriin 262. Lukkopiiri 262 antaa arvot suoraan lukkopiiriin 264. Lukkopiiri 276 antaa arvot rekisteriin 278 multiplekserin 294 kautta. Rekisteri 290 antaa arvot suoraan lukkopiiriin 292 ja niin edelleen. Arvojen siirtäminen lukkopiirien 262 - 292 kautta ja multiplekserin 294 kautta mahdollistaa arvojen antamisen multiplekserin 260 kautta kaikkiin lukkopiireihin rengaspuskurissa 259. Havainnollisesti painotettujen puhenäytteiden $x_p(n)$ ja arvojen -2b ja b^2 tallentamisen jälkeen annetaan painotettujen syntetisoitujen puhenäytteiden $y_L(n)$ sekvenssit lukkopiiriin 210. Painotetut syntetisoitujen puhenäytteet $y_L(n)$ annetaan lukkopiirillä 210 kertojan 216 kahteen tuloon, joka tuottaa

painotettujen syntetisoitujen puhenäytteiden neliöt $(y_L(n))^2$. Lisäksi lukkopiiri 210 antaa painotetut syntetisoidut puhenäytteet $y_L(n)$ kertojan 218 ensimmäiseen tuloon. RAM X 212 antaa havainnollisesti painotetut puhenäytteet $x_p(n)$ lukkopiirin 214 kautta kertojan 218 toiseen tuloon. Kertoja 218 laskee tuloarvot $x_p(n)y_L(n)$. Uusi neliö $(y_L(n))^2$ ja uusi tulo $x_p(n)y_L(n)$ lasketaan jokaisessa jaksossa kertojilla 216 ja 218, vastaavasti. Näyteindeksi n vaihtelee 0:sta L_p - 1:een kullakin äänentasoviiveellä L.

Painotettujen syntetisoitujen puhenäytteiden toiset potenssit $(y_L(n))^2$ annetaan kerääjään 221. Tuloarvot $x_p(n)y_L(n)$ annetaan kerääjään 231. Kerääjä 221 laskee L_p neliöiden summat kullekin äänentasoviiveelle L. Kerääjä 231 laskee L_p tuloarvot kullekin äänentasoviiveelle L.

Ennen uutta äänentasoviivettä, rekisteriin 226 annetaan nolla multiplekserin 224 kautta. Seuraavaksi kerääjä 221 on valmis laskemaan autokorrelaation Ey_Ly_L nykyiselle äänentasoviiveelle L. Kerääjässä 221 neliöt $(y_L(n))^2$ annetaan summaimen 222 ensimmäiseen tuloon. Ajonaikainen kokonaissumma annetaan lukkopiirillä 226 summaimen 222 toiseen tuloon. Uutena laskettu ajonaikainen kokonaissumma annetaan summaimella 222 multiplekserin 224 kautta lukkopiiriin 226 tallennusta varten. Kaikkien L:n L_p-arvojen keräämisen jälkeen annetaan autokorrelaatio Ey_Ly_L lukkopiiriin 228 tallentamista varten.

Ennen kutakin uutta äänentasoviivettä lukkopiiriin 236 annetaan nolla multiplekserin 234 kautta. Seuraavaksi kerääjä 231 on valmis laskemaan ristikorrelaation Ex_py_L nykyiselle äänentasoviiveelle L. Kerääjässä 231 tulo-

arvot $x_p(n)y_L(n)$ annetaan summaimen 232 ensimmäiseen tuloon. Ajonaikainen kokonaissumma annetaan lukkopiirillä 236 summaimen 232 toiseen tuloon. Uutena lasketu ajonaikainen kokonaissumma annetaan summaimella 232 multiplekserin 234 kautta lukkopiiriin 236. Kaikkien L:n L_p-arvojen keräämisen jälkeen annetaan ristikorrelaatio Ex_py_L lukkopiiriin 238 tallettamista varten.

Yhtälöllä 47 kuvattu MSE lasketaan seuraavaksi kahdessa jaksoprosessissa alla kuvatulla tavalla.

Ensimmäisessä kahdesta jaksosta lukkopiiri 238 antaa havainnollisesti painotettujen puhenäytteiden ja painotettujen syntetisoitujen puhenäytteiden välisen ristikorrelaation Ex_py_L multiplekserin 230 kautta kertojan 240 ensimmäiseen tuloon. Rengaspuskuri 259 antaa skaalatun äänentasovahvistuksen -2b kertojan 240 toiseen tuloon multiplekserin 296 kautta. Tulo $-2bEx_py_L$ annetaan kertojalla 240 summaimen 242 ensimmäiseen tuloon. Summaimen 242 toiseen tuloon annetaan nolla multiplekserin 246 kautta. Summaimen 242 lähtö annetaan lukkopiiriin 244 tallettamista varten. Arvot rengaspuskurin lukkopiireissä 262 - 292 kierrätetään antamalla lukkopiirin 276 lähtö lukkopiiriin 278 multiplekserin 294 kautta ja antamalla lukkopiirin 292 lähtö lukkopiiriin 262 multiplekserin 260 kautta. Tämän kierron jälkeen lukkopiirit 262, 264, 266, 268, 270, 272, 274, 276, 278, 280, 282, 284, 286, 288, 290 ja 292 sisältävät arvot, jotka aiemmin olivat lukkopiireissä 292, 262, 264, 266, 268, 270, 272, 274, 276, 278, 280, 282, 284, 286, 288 ja 290, vastaavasti. Äänentasohaussa rengaspuskuri koostuu lukkopiireistä 262 - 292 ja rengaspuskurin multipleksereistä 260 ja 294. Kiertämällä arvot rengaspuskurissa 259, lukkopiiri 292 antaa -2b:n ja b^2:n ensimmäisessä ja toisessa jaksossa vastaavasti.

Toisessa kahdesta jaksosta lukkopiiri 228 antaa painotettujen syntetisoitujen puhenäytteiden autokorrelaation Ey_Ly_L multiplekserin 230 kautta kertojan 240 ensimmäiseen tuloon. Rengaspuskuri 259 antaa skaalatun äänentasovahvistusarvon b^2 kertojan 240 toiseen tuloon multiplekserin 296 kautta. Tulo $b^2Ey_Ly_L$ annetaan kertojalla 240 summaimen 242 ensimmäiseen tuloon. Summaimen 242 toiseen tuloon annetaan lukkopiirin 244 lähtö, $-2bEx_py_L$, multiplekserin 246 kautta. Summain 242 antaa $-2bEx_py_L + b^2Ey_Ly_L$ lukkopiiriin 244 tallettamista varten. Arvot rengaspuskurin 259 lukkopiireissä 262 - 292 sitten kierretään ylläkuvatulla tavalla.

Kaksi ylläkuvattua jaksoprosessia toistetaan kaikille kahdeksalle skaalattujen äänentasovahvistusarvojen parille $(-2b, b^2)$. MSE-arvon $-2bEx_py_L + b^2Ex_py_L$ laskentaa seuraavan kahden jakson aikana lasketaan uusi MSE-arvo käyttämällä uutta paria arvoista $-2b$ ja b^2. Ennen lukkopiirin 244 päivittämistä uudella MSE-arvolla, nykyistä MSE-arvoa verrataan nykyiseen MSE-minimiin, joka on talletettu lukkopiiriin 250 nykyistä äänentasoalikehystä varten. Nykyinen MSE-arvo $-2bEx_py_L + b^2y_Ly_L$ annetaan lukkopiirillä 244 vähentäjän 248 positiiviseen tuloon. Lukkopiiri 250 antaa senhetkisen MSE-minimin vähentäjän 248 negatiiviseen tuloon. Ohjaus 220 seuraa erotuksen tulosta vähentäjältä 248. Jos ero on negatiivinen, nykyinen MSE-arvo on uusi MSE-minimi nykyiselle äänentasoalikehykselle ja talletetaan lukkopiiriin 250 ja vastaava äänentasoviiveestimaatti $\hat{L}$ sekä äänentasovahvistusestimaatti-indeksi $\hat{b}$ päivitetään ohjauksessa 220. Jos ero on ei-negatiivinen, nykyinen MSE-arvo jätetään huomiotta.

Ennen kutakin äänentasoalikehystä, DSP-ydin 4 antaa komennon minimointiprosessorille 334 ilmoittaen ohja-

ukselle 220, että uusi äänentasoalikehys on tulossa. Komennon vastaanoton yhteydessä nykyinen äänentasoviive ja nykyinen äänentasovahvistusindeksi asetetaan 0:ksi ohjauksessa 220. Ennen uutta painotettujen syntetisoitujen puhenäytteiden sekvenssiä puhenäytteet annetaan minimointiprosessorille 334, DSP-ydin 4 antaa komennon minimointiprosessorille 334 ilmoittaen ohjaukselle 220, että uusi painotettujen syntetisoitujen puhenäytteiden sekvenssi on tulossa. Tämän komennon vastaanoton yhteydessä nykyinen äänentasoviive ja nykyinen äänentasovahvistusindeksi kasvatetaan 1:llä ohjauksella 220 vastaten äänentasoviiveen lisäystä 1 ja äänentasovahvistusindeksin lisäystä 0.25. Kun painotettujen syntetisoitujen puhenäytteiden ensimmäinen sekvenssi annetaan minimointiprosessorille 334, nykyinen äänentasoviive ja nykyinen äänentasovahvistusindeksi vastaavat 1:stä vastaten äänentasoviivettä L=17 ja normalisoitua äänentasovahvistusta b=0.25. Lisäksi ennen kutakin äänentasoalikehystä, nykyinen äänentasoviive-estimaatti $\hat{L}$ ja nykyinen äänentasovahvistusestimaatti-indeksi $\hat{b}$ asetetaan nollaksi, mikä osoittaa mitätöntä äänentasoviivettä ja äänentasovahvistusta. Kunkin äänentasoalikehyksen aikana ohjaus 220 tunnistaa ensimmäisen negatiivisen MSEn lukkopiirissä 244. Tämä arvo talletetaan lukkopiiriin 250 ja vastaavat äänentasoviive-estimaatti $\hat{L}$ ja äänentasovahvistusestimaatti-indeksi $\hat{b}$ päivitetään ohjauksessa 220. Tämä tehdään MSE-minimin alustamiseksi lukkopiirissä 250 kullakin äänentasoalikehyksellä. Ellei negatiivista MSE-arvoa saada äänentasoalikehyksen aikana, äänentasoviive-estimaatti $\hat{L}$ ja äänentasovahvistusestimaatti-indeksi $\hat{b}$ ovat nollia äänentasoalikehyksen lopussa. Nämä estimaatit annetaan ohjauksella 220 DSP-ytimeen 4. Jos DSP-ydin vastaanottaa mitättömän äänentasoviive-estimaatin, optimaalinen äänentasovahvistus asetetaan nollaksi, $\hat{b}$ = 0 vastaten nolla-MSE:tä. Kun äänentasosuodattimen äänentasovahvistus on asetettu nol-

laan, äänentasoviiveellä ei ole seurausta. Jos DSP-ydin 4 vastaanottaa voimassaolevan äänentasoviive-estimaatin $\hat{L}$, niin tätä arvoa käytetään optimaalisena äänentasoviiveenä ja optimaalisena äänentasovahvistuksena tulee olemaan 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75 ja 20 äänentasovahvistusestimaatti-indekseille 1:stä 8:aan, vastaavasti.

Äänentasohaussa yhtälön 47 MSE-funktion MSE(L,b) luonne mahdollistaa laskennallisten säästöjen aikaansaamisen. Nykyisen äänentasoviiveen jäljelle jäävät laskennat voidaan keskeyttää kun on määritetty, että jäljelle jäävät MSE-arvot, jotka vielä on laskematta nykyiselle äänentasoviiveelle, eivät voi johtaa MSE-arvoon, joka on pienempi kuin lukkopiiriin 250 talletettu MSE-minimi. Esimerkkisovellutuksessa minimointiprosessorissa 334 käytetään kolmea tekniikkaa laskennan säästämiseksi äänentasohaussa. MSE-funktiot, MSE(L,b), ovat neliöllisiä b:n suhteen. Yksi neliöllinen yhtälö muodostetaan kutakin äänentasoviivettä L varten. Kaikki nämä neliölliset yhtälöt kulkevat origon kautta, b=0 ja MSE(L,b)=0. Äänentasovahvistusarvo b=0 kuuluu mahdollisten vahvistusarvojen joukkoon, vaikkakaan sitä ei tarkoituksella haeta äänentasohakuoperaatiossa.

Ensimmäiseen laskennan säästämismenetelmään kuuluu MSE-arvojen laskennan keskeyttäminen nykyisen äänentasoviiveen äänentasohakuproseduurissa kun Ex_py_L on negatiivinen. Kaikki äänentasovahvistusarvot ovat positiivisia, mikä varmistaa sen, että nolla on yläraja MSE-minimille kullakin alikehyksellä. Ex_py_L:n negatiivinen arvo johtaisi positiiviseen MSE-arvoon ja olisi näin ollen alioptimaalinen.

Toiseen laskennan säästämismenetelmään kuuluu jäljellä olevien MSE-arvojen laskennan keskeyttäminen nykyisen äänentasoviiveen äänentasohakuproseduurissa perustuen MSE-funktion neliölliseen ominaisuuteen. MSE-funktio, MSE(L,b), lasketaan äänentasovahvistusarvoille, jotka kasvavat monotonisesti. Kun positiivinen MSE-arvo lasketaan nykyiselle äänentasoviiveelle, kaikki jäljellä olevat MSE-laskennat nykyiselle äänentasoviiveelle keskeytetään, koska kaikki jäljellä olevat MSE-arvot olisivat samoin positiivisia.

Kolmanteen laskennan säästämismenetelmään kuuluu jäljellä olevien MSE-arvojen laskennan keskeyttäminen nykyisen äänentasoviiveen äänentasohakuproseduurissa perustuen MSE-funktion neliölliseen ominaisuuteen. MSE-funktio, MSE(L,b), lasketaan äänentasovahvistusarvoille, jotka kasvavat monotonisesti. Kun nykyisessä äänentasoviiveessä laskettiin MSE-arvo, joka määritettiin uudeksi MSE-minimiksi, kaikki jäljellä olevat MSE-laskennat nykyiselle äänentasoviiveelle keskeytetään, koska mikään jäljellä olevista MSE-arvoista ei voi olla pienempi kuin MSE-minimi. Kolme ylläkuvattua laskennan säästämismenetelmää aikaansaavat merkittäviä tehonsäästöjä minimointiprosessorissa 334.

Lohkossa 76 äänentasoarvot kvantisoidaan. Kullekin äänentasoalikehykselle valitut parametrit $\hat{b}$ ja $\hat{L}$ muunnetaan lähetyskoodeiksi PGAIN ja PLAG. Optimaalinen äänentasovahvistusindeksi on kokonaislukuarvo välillä 1 ja 8, sisältyen. Optimaalinen äänentasoviive $\hat{L}$ on kokonaislukuarvo välillä 1 ja 127, sisältyen.

PLAGin arvo riippuu sekä arvosta $\hat{b}$ että $\hat{L}$. Jos $\hat{b}=0$, niin PLAG=0. Muutoin PLAG=$\hat{L}$. Näin ollen PLAG esitetään käyttämällä seitsemää bittiä. Jos $\hat{b}=0$, niin

PGAIN=0. Muutoin PGAIN=$\hat{b}$ - 1. Näin ollen PGAIN esitetään käyttämällä kolmea bittiä. Huomaa, että sekä $\hat{b}$=0 että $\hat{b}$=1 antaa PGAIN=0. Nämä kaksi tapausta erotetaan PLAGin arvolla, joka on nolla ensimmäisessä ja ei-nolla toisessa tapauksessa.

Kaikilla muilla paitsi kahdeksasosanopeudella kukin äänentasoalikehys käsittää kaksi koodikirja-alikehystä. Kullekin koodikirja-alikehykselle optimaalinen koodikirjaindeksi î ja optimaalinen koodikirjavahvistus $\hat{G}$ määritetään koodikirjahakuproseduurissa lohkossa 80. Kahdeksasosanopeudelle määritetään vain koodikirjaindeksi ja yksi koodikirjavahvistus ja koodikirjaindeksi poistetaan ennen lähetystä.

Viitaten kuvioon 9 esimerkkisovellutuksessa ärsykekoodikirja, joka annetaan koodikirjalla 400 sisältää 2^M koodivektoria, missä M=7.

Esimerkkisovellutuksessa rengaskoodikirja koostuu 128:sta arvosta, jotka annetaan taulukossa IV alla. Arvot merkitään desimaaleina ja talletetaan ROM E:een 114.

Taulukko IV

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
c(n)	0.0	-2.0	0.0	-1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

n	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
c(n)	0.0	-1.5	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.5

n	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
c(n)	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	1.5	1.0	0.0	1.5	2.0	0.0	0.0

n	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
c(n)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.5	0.0	0.0

n	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
c(n)	-1.5	1.5	0.0	0.0	-1.0	0.0	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-2.5	0.0

n	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
c(n)	0.0	0.0	0.0	1.5	0.0	0.0	0.0	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0

n	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
c(n)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.5	3.0	-1.5	-2.0	0.0	-1.5	-1.5

n	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
c(n)	1.5	-1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Koodikirjavektorin indeksin I ja koodikirjavahvistuksen G valinnassa käytettävä menetelmä on analyysi-synteesimenetelmä, joka vastaa äänentasohakuproseduurissa käytettävää. Valittu koodikirjaindeksi Î ja valittu koodikirjavahvistus $\hat{G}$ ovat niiden sallitut arvot, jotka minimoivat yhtälön 50 neliöllisen keskivirheen MSE(I,G) painotetun syntetisoidun puheen $y_I(n)$ ja havainnollisesti painotetun puheen välillä äänentasoestimaatin, $x_c(n)$, ollessa poistettu. Viitaten kuvioon 9, painotettu puhe, $x_c(n)$, muodostetaan seuraavasti. Samaa äänentasohaussa nykyiselle alikehykselle havainnollisesti painottua puhetta x(n) käytetään tässä. Johtuen äänentasohaun puuttumisesta kahdeksasosanopeudella, x(n) generoidaan koodikirjahaussa tällä nopeudella. x(n) annetaan summaimen 410 ensimmäiseen tuloon. Käyttäen optimaalista äänentasoviivettä $\hat{L}$ ja optimaalista äänentasovahvistusta $\hat{b}$, jotka saatiin nykyisen alikehyksen äänentasohaussa ja äänentasosuodattimen nykyisistä tilamuisteista 506, lasketaan äänentasosynteesisuodattimen 506 ZIR, $p_{zir}(n)$ ZIR-elementissä 406. ZIR siirretään painotetun syntteesisuodattimen 408 kautta ja lähtö $pa_{zir}(n)$ vähennetään x(n):stä summaimessa 410. Summaimen 410 lähtö, $x_c(n)$, normalisoidaan ja annetaan minimointiprosessorille 412.

Koodikirjavektori $C_I(n)$ annetaan koodikirjalla 400 vastauksena koodikirjaindeksiin I yhtälön 53 mukaisesti. Painotetun synteesisuodattimen 404 impulssivastetta h(n), joka on määritetty jo nykyisen äänentasoalikehyksen äänentasohakuproseduurissa, käytetään esimerkkisovellutuksessa. Kuitenkin kahdeksasosanopeudella h(n) lasketaan koodikirjahaussa painotetussa synteesisuodattimessa 404. Esimerkkisovellutuksessa impulssivaste h(n) typistetään ensimmäiseen 20:een näytteeseen.

Johtuen koodikirjan rekursiivisesta luonteesta, käytetään äänentasohaussa käytettyä proseduuria vastaavaa rekursiivista konvoluutioproseduuria koodikirjahaussa. Konvoluutio lasketaan alla olevalla yhtälöllä 52 esitetyllä tavalla.

$$y_1(n) = \sum_{i=0}^{19} h(i) C_I(n-i),\ 0 \le I < 128,\ 0 \le n < L_C \qquad (52)$$

Koodikirjaindeksi I, $C_I(n)$ määritetään yhtälöllä 53 esitetyllä tavalla.

$$C_I(n) = \begin{cases} c\big((n-I) \bmod 128\big) & n-I \ge 0 \\ c\big(128+(n-I)\big) & n-I < 0 \end{cases} \quad 0 \le I < 128,\ 0 \le n < L_C \qquad (53)$$

Täydellinen konvoluutio suoritetaan indeksille I=0 yhtälön 54 mukaan. Lähtö $y_0(n)$ talletetaan RAM C:hen 182. Lopuille indekseille I=1 - I=127, konvoluutio suoritetaan rekursiivisesti, kuten esitetään yhtälöissä 55-57 alla.

$$y_0(n) = h(n) * C_0(n) = \sum_{i=0}^{19} h(i) C_0(n-i), \qquad 0 \le n < L_c \qquad (54)$$

$$y_I(n) = \begin{cases} h(0)C_I(0) & (55) \\ y_{I-1}(n-1) + h(n)C_I(n) & (56) \\ y_{I-1}(n-1) & (57) \end{cases}$$

Kuten äänentasohaussa, koodikirjahaun rekursiivisen konvoluution tehokkuutta optimoidaan kolmeen osaan jaetuilla RAM:eilla ja ROM E:llä 114 DSP-ytimessä 4.

Konvoluutioarvojen laskenta yhtälössä 56 ja tuloksen muodostaminen kussakin kellojaksossa on siten mahdollinen. Esimerkiksi yhdessä jaksossa lasketaan $y_{18}(10)$, $y_{18}(9)$ talletetaan, $y_{17}(10)$ haetaan ja h(10) haetaan. Näin yhtälö 56 yllä voi tuottaa tuloksen kussakin kellojaksossa. Tuloksen tuottaminen kussakin kellojaksossa yhtälölle 57 on myös mahdollista. Esimerkiksi yhdessä jaksossa $y_{18}(24)$ lasketaan, $y_{17}(24)$ haetaan ja $y_{18}(23)$ talletetaan.

Kyky suorittaa yhtälöt 56 ja 57 tarvitsematta siirtää painotettuja syntetisoituja puhenäytteitä $y_{I-1}(n-1)$ samaan lähde-RAM:iin kunkin koodikirjaindeksipäivityksen välillä, vaatii talletusstrategian ja laitteisto-ominaisuudet, joita pidetään ping-pongina, jossa lähde- ja kohde-RAM:it vaihdetaan kunkin indeksipäivityksen välillä. Laskettaessa $y_I(n)$ arvoja parillisille koodikirjaindeksiarvoille, $y_{I-1}(n-1)$ arvot haetaan ensimmäisestä kolmesta RAM:sta ja tulokset talletetaan toiseen kolmesta RAM:sta. Laskettaessa $y_I(n)$ arvoja parittomille koodikirjaindeksiarvoille, $y_{I-1}(n-1)$ arvot haetaan toisesta kolmesta RAM:sta ja tulokset talletetaan ensimmäiseen kolmesta RAM:sta. Tämä ping-pong eliminoi tarpeen siirtää aiemmin lasketut $y_{I-1}(n-1)$ ar-

vot samaan RAM:iin kunkin koodikirjaindeksipäivityksen välillä.

Kuvio 10 on vuokaavio rekursiivisen konvoluutio-osuuden suorittamisesta (sekä alkuperäiselle ei-rekursiiviselle että rekursiiviselle konvoluutiolle) esimerkkisovellutuksen koodikirjahaussa. Lohkossa 414 koodikirjaindeksi I alustetaan alimpaan arvoonsa, joka esimerkkisovellutuksessa on 0. Näytenumero n ja indeksi m asetetaan nollaksi ja arvot $y_0(n)$ asetetaan nollaksi. Lohkot 416 - 424 muodostavat alkukonvoluutiosilmukan, joka lasketaan ei-rekursiivisesti. Alkukonvoluution laskenta, jossa I=0, suoritetaan alla olevan yhtälön 58 mukaisesti.

$$y_0(n) = \sum_{i=0}^{19} h(i) C_0(n-i), \qquad 0 \le n < L_C \qquad (58)$$

Alkukonvoluution laskennassa käytetään kiinteänmittaisia silmukoita laskennallisen kompleksisuuden vähentämiseksi. Tällä tavalla vältetään ylimenevä, joka vaaditaan vaihtuvamittaisen silmukkarakenteen muodostamiseksi yhtälön 58 sisemmässä silmukassa (lohkot 320-324). Kukin arvo $y_0(n)$ lähetetään minimointiprosessorille 412 laskennan jälkeen.

Lohko 416 testaa näyteindeksin n. Jos n vastaa koodikirja-alikehyksen pituutta L_C, ensimmäinen konvoluutio on täydellinen ja edetään lohkoon 426. Jos lohkossa 416 n on pienempi kuin koodikirja-alikehyksen pituus, edetään lohkoon 420. Lohko 420 testaa indeksin m. Jos m on yhtäsuuri kuin suodattimen impulssivasteen pituus, 20 esimerkkisovellutuksessa, iteraatio on suoritettu ja edetään lohkoon 418, jossa m asetetaan 0:aan

ja kasvatetaan n:ää. Seuraavaksi palataan lohkoon 416. Jos lohkossa 420 m on pienempi kuin impulssivasteen pituus, 20, edetään lohkoon 424, missä kerätään osittaiset summat. Edetään lohkoon 422, jossa kasvatetaan indeksiä m ja edetään lohkoon 420.

Lohkoilla 414-424 muodostetun silmukan alkukonvoluution liittyvät operaatiot suoritetaan DSP-ytimessä 4, jossa annetaan sopiva peräkkäinsuoritus tulojen keräämiseksi, kuten esitetään lohkossa 424. Seuraavat operaatiot kuvaavat laskennan peräkkäinsuoritusta ja tapahtuvat DSP-ytimessä 4 yhden kellojakson aikana. Suodattimen vastearvo h(m+1) haetaan RAM A:sta 104 ja annetaan AREG:iin 130. Koodikirjavektoriarvo $C_I(n)$ haetaan ROM E:stä 114 ja annetaan BREG:iin 134. Osasumma $y_0(n+m-1)$, joka on C0REG:ssä 164 annetaan RAM C:hen 182 multiplekserien 168 ja 180 kautta. Osasumma $y_0(n+m+1)$ annetaan RAM C:llä 182 DREG:iin 156. Arvot h(m) ja $C_I(n)$ AREG:ssä 130 ja BREG:ssä 134, vastaavasti, annetaan kertojaan 132. Kertojan 132 lähtö annetaan multiplekserin 138 ja estetyn ryhmösiirtimen 140 kautta summaimen 146 ensimmäiseen tuloon. Osasumma $y_0(n+m)$ annetaan DREG:llä 156 multiplekserin 154, estetyn invertterin 152 ja estetyn ryhmäsiirtimen 150 kautta summaimen 146 toiseen tuloon. Esimerkkisovellutuksessa keskileikattu Gaussin koodikirja $C_I(n)$ sisältää nolla-arvojen enemmistön. Edun saamiseksi tästä tilanteesta ja tehon säästämiseksi, DSP-ydin 4 tarkastaa ensin onko koodikirjavektori nolla lohkossa 424. Jos se on nolla, kerto- ja lisäysaskel, jotka normaalisti suoritetaan lohkossa 424 ja jotka selitettiin yllä, jätetään väliin. Tämä proseduuri eliminoi kerto- ja lisäysoperaatioita karkeasti ottaen 80% ajasta säästäen siten tehoa. Summaimen 146 lähtö annetaan multiplekserin 148 kautta C0REG:iin 164. Tämä arvo C0REG:ssä sitten annetaan multiplekserien 168 ja 180

kautta RAM C:hen 182. Kun indeksi n saavuttaa suurimman sallitun arvonsa lohkossa 416, alkukonvoluutio on suoritettu ja RAM C:ssä 182 olevat osasummat ovat nyt konvoluution lopullinen tulos.

Kun alkukonvoluutio on suoritettu, edetään lohkoon 426, jossa suoritetaan rekursiivinen konvoluutio jäljellä olevien koodikirjaindeksiarvojen laskemiseksi.

Lohkossa 462 näyteindeksi n asetetaan nollaksi ja koodikirjaindeksi I:tä kasvatetaan. Edetään lohkoon 428. Lohko 428 testaa I:n. Jos I on suurempi tai yhtäsuuri kuin 128, joka on suurin koodikirjaindeksiarvo esimerkkisovellutuksessa, edetään lohkoon 430, jossa koodikirjahakuoperaatio päättyy. Jos I on pienempi tai yhtäsuuri kuin 127, edetään lohkoon 432. Lohko 432 ohjaa edellä kuvattua ping-pong-operaatiota. Lohkossa 432 testataan I:tä sen parillisuuden tai parittomuuden määrittämiseksi. Jos I on parillinen, edetään lohkoon 442 (Tapauksena I kuvattu operaatio). Jos I on pariton, edetään lohkoon 434 (Tapauksena II kuvattu operaatio).

<u>Tapaus I: (koodikirjaindeksin I parilliset arvot)</u>

Lohkossa 442 $y_I(0)$ lasketaan yhtälön 55 mukaisesti. Osoiteyksikkö A 102 antaa osoitearvon RAM A:lle 104, joka vastauksena antaa h(0):n multiplekserin 108 kautta AREG:iin 130. Samassa kellojaksossa osoiteyksikkö E 112 antaa osoitearvon ROM E:lle 114, joka vastauksena antaa $C_I(0)$ multiplekserin 116 kautta BREG:iin 134. Seuraavan kellojakson aikana AREG 130 antaa h(0):n ja BREG antaa $C_I(0)$:n kertojalle 132, jossa kaksi arvoa kerrotaan ja tulo annetaan multiplekserin 138 estetyn

ryhmäsiirtimen 140 kautta summaimen 146 ensimmäiseen 146 tuloon. Summaimen 146 toiseen tuloon annetaan nolla multiplekserin 154, estetyn invertterielementin 152 ja ryhmäsiirtimen 152 kautta. Summaimen 146 lähtö annetaan C0REG:iin 164 multiplekserin 148 kautta. Saman kellojakson aikana $y_{I-1}(0)$ ja h(1) haetaan RAM B:stä 122 ja RAM A:sta 104 ja annetaan DREG:iin 156 ja AREG:iin 130 multiplekserien 158 ja 108 kautta, vastaavasti.

Lohkossa 444 syntetisoitua puhenäyteindeksiä n lisätään. Ohjauslohkossa 446, jos syntetisoitu puhenäyteindeksi n on alle 20, niin edetään lohkoon 448.

Lohkossa 448 uusi $y_I(n)$:n arvo lasketaan kussakin kellojaksossa yhtälön 56 mukaan. Sopiva asetus, joka vaaditaan ennen lohkon 448 ensimmäistä iteraatiota arvojen $y_{I-1}(n-1)$ ja h(n) alustamiseksi, saatiin lohkossa 442 yllä kuvatulla tavalla. Sopiva puhdistus vaaditaan myös lohkon 448 viimeisen iteraation jälkeen lopullisen arvon $y_I(19)$ tallettamiseksi.

Lohkon 448 ensimmäisen iteraatio $y_I(0)$, joka on laskettu lohkossa 442, on läsnä C0REG:ssä 164. C0REG 164 antaa $y_I(0)$:n multiplekserien 168 ja 180 kautta RAM C:hen 182 talletettavaksi yhdessä osoiteyksikkö C:n 186 RAM C:hen 182 antaman osoitearvon kanssa. $y_I(0)$ annetaan minimointiprosessorille 412 samalla kun se annetaan RAM C:hen 182.

Lohkossa 448 seuraavat operaatiot suoritetaan yhdessä kellojaksossa. $y_{I-1}(n)$ arvo annetaan RAM B:llä 122 osoiteyksikkö B:n 120 antaman osoitteen mukaisesti multiplekserien 116 ja 158 kautta DREG:iin 156. Im-

pulssivastearvo h(n+1) annetaan RAM A:lla 104 osoiteyksikkö A:n 102 antaman osoitteen mukaisesti multiplekserin 108 kautta AREG:iin. DREG 156 antaa $y_{I-1}(n-1)$ multiplekserin 154, estetyn invertterielementin 152 ja ryhmäsiirtimen 150 kautta summaimen 146 ensimmäiseen tuloon. AREG 130 antaa h(n):n ja BREG antaa $C_I(n)$:n kertojaan 132, jossa kaksi arvoa kerrotaan ja tulo annetaan kertojalla 132 multiplekserin 138 kautta, estetyn ryhmäsiirtimen 140 kautta, summaimen 146 toiseen tuloon. Summaimen 146 lähtö annetaan multiplekserin 148 kautta C0REG:iin 164. C0REG:n 164 arvo, joka on laskettu edellisessä iteraatiossa, annetaan multiplekserien 168 ja 180 kautta RAM C:hen 182 talletettavaksi ja minimointiprosessoriin 412.

Ohjauslohkossa 446, jos syntetisoidun puhenäytteen indeksi n on yhtäsuuri kuin 20, niin lasketaan $y_I(19)$ viimeisessä iteraatiossa ja annetaan multiplekserien 168 ja 124 kautta RAM B:hen 122 talletettavaksi rengaspuskuriin ja minimointiprosessoriin 412 ennen etenemistä lohkoon 454.

Tapauksen I loppu

Tapaus II:(koodikirjaindeksin I parittomat arvot)

Lohkossa 434 $y_I(0)$ lasketaan yhtälön 55 mukaisesti. Osoiteyksikkö A 102 antaa osoitearvon RAM A:lle 104, joka vastauksena antaa h(0) multiplekserin 108 kautta AREG:iin 130. Samassa kellojaksossa osoiteyksikkö E 112 antaa osoitearvon ROM E:hen 114, joka vastauksena antaa $C_I(0)$ multiplekserin 116 kautta BREG:iin 134. Seuraavan kellojakson aikana AREG 130 antaa h(0):n ja BREG antaa $C_I(0)$:n kertojalle 132, jossa kaksi arvoa

kerrotaan ja tulo annetaan multiplekserin 138 ja estetyn ryhmäsiirtimen 140 kautta summaimen 146 ensimmäiseen tuloon. Summaimen 146 toiseen tuloon annetaan nolla multiplekserin 154, estetyn invertterin 152 ja ryhmäsiirtimen 152 kautta. Summaimen 146 lähtö annetaan C0REG:iin 164 multiplekserin 148 kautta. Saman kellojakson aikana $y_{I-1}(0)$ ja h(1) haetaan RAM C:stä 182 ja RAM A:sta 104 ja annetaan DREG:iin 156 ja AREG:iin 130 multiplekserien 158 ja 100 kautta, vastaavasti.

Lohkossa 436 syntetisoidun puhenäytteen indeksiä n lisätään. Ohjauslohkossa 438, jos syntetisoidun puhenäytteen indeksi on alle 20 edetään lohkoon 440.

Lohkossa 440 uusi $y_I(n)$:n arvo lasketaan kussakin kellojaksossa yhtälön 56 mukaan. Sopiva asetus, joka vaaditaan ennen lohkon 440 ensimmäistä iteraatiota arvojen $y_{I-1}(n)$ ja h(n) alustamiseksi, saatiin lohkossa 434 yllä kuvatulla tavalla. Sopiva puhdistus vaaditaan myös lohkon 440 viimeisen iteraation jälkeen lopullisen arvon $y_I(19)$ tallettamiseksi.

Lohkon 440 ensimmäisessä iteraatiossa $y_I(0)$, joka on laskettu lohkossa 434, on läsnä C0REG:ssä 164. C0REG 164 antaa $y_I(0)$:n multiplekserien 168 ja 180 kautta RAM B:hen 122 talletettavaksi yhdessä osoiteyksikkö B:n 120 RAM B:hen 122 antaman osoitteen kanssa. $y_I(0)$ annetaan minimointiprosessorille 412 samalla kun se annetaan RAM B:hen 122.

Lohkossa 440 seuraavat operaatiot suoritetaan yhdessä kellojaksossa. $y_{I-1}(n)$ arvo annetaan RAM C:llä 182 osoiteyksikkö C:n 186 antaman osoitteen mukaisesti

multiplekserin 158 kautta DREG:iin 156. Impulssivaste h(n+1) annetaan RAM A:lla 104 osoiteyksikkö A:n 102 antaman osoitteen mukaisesti multiplekserin 108 kautta AREG:iin. DREG 156 antaa $y_{I-1}(n-1)$:n multiplekserin 154, estetyn invertterielementin 152 ja ryhmäsiirtimen 150 kautta summaimen 146 ensimmäiseen tuloon. AREG 130 antaa h(n):n ja BREG antaa $C_I(n)$:n kertojaan 132, jossa kaksi arvoa kerrotaan ja tulo annetaan kertojalla 132 multiplekserin 138 ja ryhmäsiirtimen 140 kautta summaimen 146 ensimmäiseen tuloon. Summaimen 146 lähtö annetaan multiplekserin 148 kautta C0REG:iin 164. C0REG:n 164 arvo, joka on laskettu edellisessä iteraatiossa, annetaan multiplekserien 168 ja 124 kautta RAM B:hen 122 talletettavaksi ja minimointiprosessoriin 412.

Lohkossa 436 syntetisoidun puheen näyteindeksiä n lisätään. Ohjauslohkossa 438, jos syntetisoidun puhenäytteen indeksi n on yhtäsuuri kuin 20, viimeisessä iteraatiossa laskettu $y_I(19)$ annetaan multiplekserien 168 ja 124 RAM B:hen 122 talletettavaksi rengaspuskuriin ja minimointiprosessoriin 412 ennen etenemistä lohkoon 454.

Tapauksen II loppu

Ennen lohkon 454 ensimmäistä iteraatiota $y_{I-1}(19)$ haetaan rengaspuskurista RAM B:ssä 122 ja ladataan BREG:iin 134, seuraavaksi $y_{I-1}(19)$ siirretään BREG:stä 134 C0REG:iin 164, jonka jälkeen $y_{I-1}(20)$ haetaan rengaspuskurista RAM B:ssä 122 ja ladataan BREG:iin 134.

Lohkossa 454 lasketaan uusi $y_I(n)$ kussakin kellojaksossa yhtälön 57 mukaan. Seuraavat operaatiot suoritetaan yhdessä kellojaksossa. $y_{I-1}(n-2)$ annetaan BREG:llä 134 C0REG:iin 164. $y_{I-1}(n-3)$ haetaan rengaspuskurista RAM B:ssä 122 ja ladataan BREG:iin 134. C0REG:ssä oleva $y_{I-1}(n-1)$ esitetään minimointiprosessorille 412. Lohkon 454 viimeisen iteroinnin jälkeen, $y_{I-1}(L_C-2)$ poistetaan rengaspuskurista RAM B:ssä 122. Lisäämällä elementti rengaspuskurin kuhunkin äänentasoviiveeseen ja poistamalla elementti rengaspuskurin kustakin äänentasoviiveestä, pidetään rengaspuskurin kokona L_C-19.

Rengaspuskurin toteutus RAM B:ssä 122 saadaan erityisten osoiterekisterien kautta osoiteyksikössä B 120, joka määrää ympärikiertopisteet niin, että sekventiaalista muistia voidaan osoittaa automaattisesti rengasmaisella tavalla.

Lohkossa 450, syntetisoidun puhenäytteen indeksiä n lisätään. Ohjauslohkossa 452, jos syntetisoidun puhenäytteen indeksi n on pienempi kuin L_C, edetään uudelleen lohkoon 454. Jos syntetisoidun puhenäytteen indeksi n on yhtäsuuri kuin L_C, niin senhetkiselle koodikirjaindeksille L on laskettu kaikki $y_I(n)$ arvot ja palataan lohkoon 426.

Ennen syntetisoitujen puhenäytteiden $y_I(n)$ laskentaa, L_C:n mittainen sekvenssi havainnollisesti syntetisoituja puhenäytteitä $x_c(n)$ annetaan minimointiprosessorille 412. Kuten aikaisemmin esitettiin, syntetisoitujen puhenäytteiden $y_I(n)$ L_c:n mittaiset sekvenssit koodikirjaindeksiarvoille I=0 - I=127 annetaan minimointiprosessorille 412 alku- ja rekursiivisten konvoluutiolaskentojen aikana. Syntetisoidut puhenäytteet an-

netaan jaksottain minimointiprosessorille 412. Minimointiprosessori 412 laskee kunkin syntetisoitujen puhenäytteiden $y_I(n)$ sekvenssin autokorrelaation ja ristikorrelaation kunkin syntetisoitujen puhenäytteiden $y_I(n)$ sekvenssin ja havainnollisesti painotettujen puhenäytteiden $x_c(n)$ sekvenssin välillä. Näistä korrelaatioarvoista minimointiprosessori 412 laskee MSE:n suhteellisen arvon kaikkien syntetisoitujen puhenäytteiden $y_I(n)$ sekvenssin ja havainnollisesti painotettujen puhenäytteiden $x_c(n)$ sekvenssin välillä. Kullekin koodikirjavektori-indeksille I, lasketaan MSE syntetisoitujen puhenäytteiden $y_L(n)$ sekvenssin kaikille mahdollisille koodikirjavahvistusarvoille G. Minimointiprosessori 412 löytää MSE-minimin kaikkien yli ja kaikkien mahdollisten koodikirjavahvistusarvojen G yli ylläpitämällä MSE-minimiä nykyisen koodikirjavektori-indeksiarvon I ja koodikirjavahvistusarvon G kautta. Koodikirjavektori-indeksiestimaatti $\hat{I}$ ja koodikirjavahvistusindeksiestimaatti $\hat{G}$ vastaten MSE-minimiä ylläpidetään myös minimointiprosessorilla 412. Jokaista uutta MSE-arvoa verrataan minimointiprosessorilla 412 ylläpidettyyn MSE-arvoon. Jos uusi MSE-arvo on pienempi kuin minimi-MSE, minimi-MSE korvataan uudella MSE:llä ja koodikirjavektori-indeksiestimaatti $\hat{I}$ ja koodikirjavahvistusestimaatti-indeksi $\hat{G}$ päivitetään vastaamaan uutta MSE-minimiä. MSE-minimi ja vastaava koodikirjavektori-indeksiestimaatti $\hat{I}$ ja koodikirjavahvistusestimaatti-indeksi, $\hat{G}$, joita ylläpidetään minimointiprosessorissa 412 alustetaan kussakin koodikirjakehyksessä käyttäen ensimmäistä MSE-arvoa, joka on laskettu koodikirja-alikehyksen aikana. Kaikkien koodikirjavektori-indeksien, I, ja koodikirjavahvistusarvojen G käyttämisen jälkeen, koodikirjavektori-indeksiestimaatti $\hat{I}$ ja koodikirjavahvistusestimaatti-indeksi, $\hat{G}$, ovat optimaalinen koodikirjavektori-ineksi ja optimaalinen koodikirjavahvistusindeksi, vastaavasti, senhetkiselle koodikirja-alikehykselle.

Minimointiprosessori 412 antaa optimaalisen koodikirjavektori-indeksin Î ja optimaalisen koodikirjavahvistusestimaatti-ineksin $\hat{G}$ DSP-ytimeen 4 tuloporttien 12 kautta. Optimaalinen koodikirjavektori-indeksi Î ja optimaalinen koodikirjavahvistusestimaatti-indeksi $\hat{G}$ ovat ne, jotka johtavat minimi-MSE:hen havainnollisesti painotettujen puhenäytteiden $x_C(n)$ ja painotettujen syntetisoitujen puhenäytteiden $y_I(n)$ välillä. MSE on koodikirjaindeksin I ja koodikirjavahvistuksen G funktio, kuten alla yhtälöissä 59 - 61 kuvataan:

$$MSE(I,G) = \sum_{n=0}^{L_C-1}\left(x_C(n) - Gy_I(n)\right)^2 \quad (59)$$

$$= \sum_{n=0}^{L_C-1}\left(x_C(n)\right)^2 - 2G\sum_{n=0}^{L_C-1}\left(x_C(n)y_i(n)\right) + G^2\sum_{n=0}^{L_C-1}\left(y_I(n)\right)^2 \quad (60)$$

$$= Ex_Cx_C - 2GEx_Cy_I + G^2Ey_1y_1 \quad (61)$$

Ex_Cx_C on havainnollisesti painotettujen puhenäytteiden $x_C(n)$ autokorrelaatio. Ex_Cy_I on havainnollisesti painotettujen puhenäytteiden $x_C(n)$ ja painotettujen syntetisoitujen puhenäytteiden $y_I(n)$ välinen ristikorrelaatio. Ey_Iy_I on painotettujen syntetisoitujen puhenäytteiden $y_I(n)$ autokorrelaatio.

Havainnollisesti painotettujen puhenäytteiden $x_C(n)$ autokorrelaatio Ex_Cx_C ei ole koodikirjavahvistuksen G funktio. Ex_Cx_C pysyy vakiona kunkin koodikirjaalikehyksen aikana ja siksi ei vaikuta optimaalisen koodikirjavektori-indeksin ja optimaalisen koodikirjavahvistuksen valintaan. Yhtälöiden 59 - 61 minimointi koodikirjavektori-indeksin I ja koodikirjavahvistuksen G suhteen on vastaava kuin allaolevan yhtälön 62 minimointi.

$$MSE(I,G) = -2GEx_C y_I + G^2 Ey_I y_I \qquad (62)$$

Minimointiprosessori 412 laskee painotettujen syntetisoitujen puhenäytesekvenssien $y_I(n)$ autokorrelaatiot Ey_Iy_I ja havainnollisesti painotetun puhenäytesekvenssin $x_c(n)$ ja painotettujen syntetisoitujen puhenäytesekvenssien $y_I(n)$ väliset ristikorrelaatiot Ex_cy_I. Kullekin korrelaatioparille (Ex_cy_I, Ey_Iy_I) minimointiprosessori 412 laskee suhteellisen MSE:n ylläolevan yhtälön 62 mukaan joukolle koodikirjavahvistusarvoja G. Korrelaatioiden Ex_cy_I, ja Ey_Iy_I laskenta suoritetaan samanaikaisesti minimointiprosessorissa 412. Suhteelliset MSE-arvot lasketaan ja päätökset liittyen MSE-minimointiin tehdään koodikirjavektori-indeksille I, kun taas korrelaatioarvot lasketaan koodikirjavektori-indeksille I + 1.

Kuviot 6a ja 6b esittävät minimointiprosessorin 412 esimerkkisovellutusta. Havainnollisesti painotetut puhenäytteet $x_c(n)$ annetaan DSP-ytimellä 4 lukkopiiriin 210 talletettavaksi RAM X:ään 212 ohjauksen 220 antaman osoitteen mukaisesti. Kaksi koodikirjavahvistusarvojoukkoa annetaan DSP-ytimellä 4 minimointiprosessoriin 412. Yksi joukko on positiivisille koodikirjavahvistusarvoille G ja toinen joukko on negatiivisille koodikirjavahvistusarvoille -G. Täydellä nopeudella ja puolinopeudella skaalatut koodikirjavahvistusarvot -2G arvoille G=-4dB - G=+8dB +4db:n askelein annetaan DSP-ytimellä 4 multiplekserin 260 kautta talletettaviksi lukkopiireihin 292, 288, 284 ja 280, vastaavasti. Neljäsosanopeudella ja kahdeksasosanopeudella skaalatut koodikirjavahvistusarvot -2G arvoille G=-4.0dB - G=+2.0dB +2dB:n askelein annetaan DSP-ytimellä 4 multiplekserin 260 kautta talletettaviksi lukkopiireihin 292, 288, 284 ja 280, vastaavasti. Vastaavat skaalatut

koodikirjavahvistusarvot G^2 annetaan multiplekserin 260 kautta talletettaviksi lukkopiireihin 290, 286, 282 ja 278, vastaavasti. Täydellä nopeudella ja puolinopeudella, skaalatut koodikirjavahvistusarvot 2G arvoille G=-4.0dB - G=+8.0dB +4.0db:n askelein annetaan multiplekserin 260 kautta talletettaviksi lukkopiireihin 276, 272, 268 ja 264, vastaavasti. Neljäsosanopeudella ja kahdeksasosanopeudella, skaalatut koodikirjavahvistusarvot 2G arvoille G=-4.0dB - G=+2.0dB +2dB:n askelein annetaan multiplekserin 260 kautta talletettaviksi lukkopiireihin 276, 272, 268 ja 264, vastaavasti. Vastaavat skaalatut koodikirjavahvistusarvot G^2 annetaan multiplekserin 260 kautta talletettaviksi lukkopiireihin 274, 270, 266 ja 262, vastaavasti. Multiplekseri 260 antaa arvot suoraan lukkopiiriin 262. Lukkopiiri 262 antaa arvot suoraan lukkopiiriin 264. Lukkopiiri 276 antaa arvot lukkopiiriin 278 multiplekserin 294 kautta. Lukkopiiri 290 antaa arvot suoraan lukkopiiriin 292 ja niin edelleen. Arvojen siirtäminen lukkopiirien 262 - 292 ja multiplekserin 294 kautta mahdollistaa arvojen antamisen multiplekserin 260 kautta kaikkiin lukkopiireihin rengaspuskurissa 259. Koodikirjahaussa järjestetään kaksi rengaspuskuria rengaspuskuriin 259. Havainnollisesti painotettujen puhenäytteiden $x_c(n)$ tallentamisen ja koodikirjavahvistusarvojen tallentamisen jälkeen annetaan painotettujen syntetisoitujen puhenäytteiden $y_I(n)$ sekvenssit lukkopiiriin 210. Painotetut syntetisoidut puhenäytteet $y_I(n)$ annetaan lukkopiirillä 210 kertojan 216 kahteen tuloon, joka tuottaa painotettujen syntetisoitujen puhenäytteiden toisen potenssit $(y_I(n))^2$. Lisäksi lukkopiiri 210 antaa painotetut syntetisoidut puhenäytteet $y_I(n)$ kertojan 218 ensimmäiseen tuloon. RAM X 212 antaa havainnollisesti painotetut puhenäytteet $x_c(n)$ lukkopiirin 214 kautta kertojan 218 toiseen tuloon. Kertoja 218 laskee tuloarvot $x_c(n)y_I(n)$. Uusi toisen potenssi $(y_I(n))^2$ ja uusi tulo $x_c(n)y_I(n)$ laske-

taan jokaisessa jaksossa kertojilla 216 ja 218, vastaavasti. Näyteindeksi n vaihtelee 0:sta L_c-1:een kullakin koodikirjavektori-indeksillä I.

Painotettujen syntetisoitujen puhenäytteiden toiset potenssit $(y_I(n))^2$ annetaan kerääjään 221. Tuloarvot $x_c(n)y_I(n)$ annetaan kerääjään 231. Kerääjä 221 laskee L_c-neliöiden summat kullekin koodikirjavektori-indeksille I. Kerääjä 231 laskee L_c-tuloarvojen summat kullekin koodikirjavektori-indeksille I.

Ennen jokaista uutta koodikirjavektori-indeksiä, lukkopiiriin 226 annetaan nolla multiplekserin 224 kautta. Seuraavaksi kerääjä 221 on valmis laskemaan autokorrelaation Ey_Iy_I nykyiselle koodikirjavektori-indeksille I. Kerääjässä 221 neliöt $(y_I(n))^2$ annetaan summaimen 222 ensimmäiseen tuloon. Kokonaisajo annetaan lukkopiirillä 226 summaimen 222 toiseen tuloon. Uutena laskettu kokonaisajo annetaan summaimella 222 multiplekserin 224 kautta lukkopiiriin 226 talletettavaksi. Kaikkien koodikirjavektori-indeksi I:n L_c-arvojen keräämisen jälkeen annetaan autokorrelaatio Ey_Iy_I lukkopiiriin 228 tallettamista varten.

Ennen kutakin uutta koodikirjavektori-indeksiä lukkopiiriin 236 annetaan nolla multiplekserin 234 kautta. Seuraavaksi kerääjä 231 on valmis laskemaan ristikorrelaation Ex_cy_I nykyiselle koodikirjavektori-indeksille I. Kerääjässä 231 tuloarvot $x_c(n)y_I(n)$ annetaan summaimen 232 ensimmäiseen tuloon. Ajonaikainen kokonaissumma annetaan lukkopiirillä 236 summaimen 232 toiseen tuloon. Uutena laskettu kokonaisajo annetaan summaimella 232 multiplekserin 234 kautta lukkopiiriin 236 tallennusta varten. Kaikkien koodikirjavektori-indeksi I:n L_c-arvojen keräämisen jälkeen annetaan

ristikorrelaatio Ex_cy_I lukkopiiriin 238 tallettamista varten.

Yhtälöllä 62 kuvattu MSE lasketaan seuraavaksi kahdessa jaksoprosessissa allakuvatulla tavalla.

Ensimmäisessä kahdesta jaksosta lukkopiiri 238 antaa havainnollisesti painotettujen puhenäytteiden ja painotettujen syntetisoitujen puhenäytteiden välisen ristikorrelaation Ex_cy_I multiplekserin 230 kautta kertojan 240 ensimmäiseen tuloon. Ohjaus 220 seuraa lukkopiirin 238 antamaa Ex_cy_I:tä. Jos Ex_cy_I on ei-negatiiviinen, niin lukkopiiri 292 antaa skaalatun koodikirjavahvistusarvon -2G kertojan 240 toiseen tuloon multiplekserin 296 kautta. Tulo $-2GEx_cy_I$ annetaan kertojalla 240 summaimen 242 ensimmäiseen tuloon. Jos Ex_cy_I on negatiivinen, niin lukkopiiri 276 antaa skaalatun koodikirjavahvistusarvon 2G kertojan 240 toiseen tuloon multiplekserin 296 kautta. Tulo $2GEx_cy_I$ annetaan kertojalla 240 summaimen 242 ensimmäiseen tuloon. Summaimen 242 toiseen tuloon annetaan nolla multiplekserin 246 kautta. Summaimen 242 lähtö annetaan lukkopiiriin 244 tallettamista varten. Ex_cy_I:n merkki talletetaan ohjauksessa 220. Ykkösen ja nollan merkit Ex_cy_I:lle vastaavat Ex_cy_I:n negatiivisia ja positiivisia arvoja vastaavasti. Arvot lukkopiireissä 262 - 276 kierretään antamalla lukkopiirin 276 lähtö lukkopiiriin 262 multiplekserin 260 kautta. Tämän kierron jälkeen lukkopiirit 262, 264, 266, 268, 270, 272, 274 ja 276 sisältävät arvot, jotka aiemmin olivat lukkopiireissä 276, 262, 264, 266, 268, 270, 272 ja 274, vastaavasti. Arvot lukkopiireissä 278-292 kierretään antamalla lukkopiirin 292 lähtö lukkopiiriin 278 multiplekserin 294 kautta. Tämän kierron jälkeen lukkopiirit 278, 280, 282, 284, 286, 288, 290 ja 290 sisältävät arvot, jotka aikaisemmin sisältyivät lukkopiireihin 292, 278, 280, 282, 284, 286, 288 ja 290 vastaavasti. Yksi rengaspus-

kuri koostuu lukkopiireistä 262 - 276 ja multiplekseristä 260. Toinen rengaspuskuri koostuu rekistereistä 276 - 292 ja multiplekseristä 294. Kiertämällä arvot rengaspuskurin 259 ensimmäisessä kahdesta puskurista, lukkopiiri 292 antaa -2G:n ja G^2:n ensimmäisessä ja toisessa jaksossa, vastaavasti. Kiertämällä arvot rengaspuskurin 259 toisessa kahdesta puskurista, lukkopiiri 276 antaa 2G:n ja G^2:n ensimmäisessä ja toisessa jaksossa vastaavasti. Kullekin korrelaatio- ja ristikorrelaatioarvoparille, annetaan vain yksi joukko koodikirjavahvistuspareja rengaspuskurilla 259. Joukko koodikirjavahvistuspareja annetaan rengaspuskurilla käsittäen lukkopiirit 262 - 276 ja multiplekserin 260 Ex_cy_I:n negatiivisille arvoille. Joukko koodikirjavahvistuspareja annetaan rengaspuskurilla koostettuna lukkopiireillä 278 - 292 ja multiplekserillä 294 Ex_cy_I:n ei-negatiivisille arvoille.

Toisessa kahdesta jaksosta lukkopiiri 228 antaa Ey_Iy_I:n multiplekserin 230 kautta kertojan 240 ensimmäiseen tuloon. Multiplekserin 296 kautta lukkopiirit 276 - 292 antavat koodikirjavahvistusarvon G^2 kertojan 240 toiseen tuloon Ex_cy_I:n negatiivisille ja ei-negatiiviselle arvoille, vastaavasti. Tulo $G^2Ex_cy_I$ annetaan kertojalla 240 summaimen 242 ensimmäiseen tuloon. Summaimen 242 toiseen tuloon annetaan lukkopiirin 244 lähtö $\pm 2GEx_cy_I$ multiplekserin 246 kautta. Summain 242 antaa $\pm 2GEx_cy_I+G^2Ey_Iy_I$ lukkopiiriin 244 talletettamista varten. Arvot rengaspuskurin 259 lukkopiireissä 262 - 292 sitten kierretään ylläkuvatulla tavalla.

Kaksi ylläkuvattua jaksoprosessia toistetaan kaikille neljälle koodikirjavahvistusarvojen parille ($\pm$ 2G, G^2) jokaiselle koodikirjaindeksille I. Nykyisen MSE-arvon $2GEx_cy_I + G^2Ex_cy_I$ laskentaa seuraavan kahden jakson ai-

kana lasketaan uusi MSE-arvo käyttämällä arvojen ±2G ja G^2 seuraavaa paria. Ennen lukkopiirin 244 päivittämistä uudella MSE-arvolla, nykyistä MSE-arvoa verrataan minimi-MSE:hen nykyistä koodikirja-alikehystä varten, joka on tallennettu lukkopiiriin 250. Nykyinen MSE-arvo $\pm 2Ex_C y_I + G^2 Ex_I y_I$ annetaan lukkopiirillä 244 vähentäjän 248 positiiviseen tuloon. Lukkopiiri 250 antaa senhetkisen minimin MSE-arvon vähentäjän 248 negatiiviseen tuloon. Ohjaus 220 seuraa erotuksen tulosta vähentäjältä 248. Jos ero on negatiivinen, nykyinen MSE-arvo on uusi MSE-minimi nykyiselle koodikirja-alikehykselle ja talletetaan lukkopiiriin 250 ja vastaava koodikirjavektori-indeksiestimaatti Î sekä koodikirjavahvistusestimaatti-indeksi $\hat{G}$ päivitetään ohjauksessa 220. Jos ero on ei-negatiivinen, nykyinen MSE-arvo jätetään huomiotta.

Ennen kutakin koodikirja-alikehystä, DSP-ydin 4 antaa komennon minimointiprosessorille 412 ilmoittaen ohjaukselle 220, että uusi äänentasoalikehys on tulossa. Tämän komennon vastaanoton yhteydessä nykyinen koodikirjavektori-indeksi ja nykyinen koodikirjavahvistusindeksi asetetaan 0:ksi ohjauksessa 220. Ennen kunkin uuden painotettujen syntetisoitujen puhenäytteiden sekvenssin puhenäytteiden antamista minimointiprosessorille 412, DSP-ydin antaa komennon minimointiprosessorille 412 ilmoittaen ohjaukselle 220, että uusi painotettujen syntetisoitujen puhenäytteiden sekvenssi on tulossa. Tämän komennon vastaanoton yhteydessä nykyinen koodikirjavektori-indeksi ja nykyinen koodikirjavahvistusindeksi kasvatetaan 1:llä ohjauksessa 220 vastaten koodikirjavektori-indeksin lisäystä 1 ja koodikirjavahvistuksen lisäystä 2 dB tai 4 dB riippuen nopeudesta. Kun painotettujen syntetisoitujen puhenäytteiden ensimmäinen sekvenssi annetaan minimointiprosessorille 412, nykyinen koodikirjavektori-indeksi

ja nykyinen koodikirjavahvistusindeksi vastaavat 1:stä vastaten koodikirjavektori-indeksiä ja koodikirjavahvistusta G=-8 dB tai G=-4 dB riippuen nopeudesta. Kunkin koodikirja-alikehyksen aikana ensimmäinen MSE-arvo talletetaan lukkopiiriin 250 ja vastaava koodikirjavektori-indeksiestimaatti Î ja vastaava koodikirjavahvistusestimaatti-indeksi $\hat{G}$ päivitetään ohjauksessa 220. Tämä tehdään MSE-minimin alustamiseksi lukkopiirissä 250 kullakin koodikirja-alikehyksellä. MSE-minimiestimaatteja vastaavat koodikirjavektori-indeksi ja koodikirjavahvistusindeksi annetaan ohjauksella 220 DSP-ytimeen 4 yhdessä ristikorrelaation, Ex_cy_I, merkin kanssa vastaten MSE-minimiä. DSP-ytimen 4 vastaanottaessa nollan Ex_cy_I:n merkiksi, se asettaa optimaalisen koodikirjavahvistuksen G:ksi. DSP-ytimen 4 vastaanottaessa ykkösen Ex_cy_I:n merkiksi, se asettaa optimaalisen koodikirjavahvistuksen -G:ksi. DSP-ydin 4 käyttää ohjauksen 220 antamaa koodikirjavektori-indeksiestimaattia ja koodikirjavahvistusestimaatti-indeksiä optimaalisen koodikirjavektorin ja optimaalisen koodikirjavahvistuksen määrittämiseksi. Täydellä nopeudella ja puolinopeudella optimaalinen koodikirjavahvistus G on -4 dB, 0 dB, +4 dB ja +8 dB koodikirjavahvistusindekseille välillä $\hat{G}$=1 - $\hat{G}$=4, vastaavasti. Neljäsosanopeudella ja kahdeksasosanopeudella optimaalinen koodikirjavahvistus G on -4 dB, -2 dB, 0 dB ja +2 dB koodikirjavahvistusindekseille välillä $\hat{G}$=1 - $\hat{G}$=4, vastaavasti.

Koodikirjahaussa yhtälön 62 MSE-funktion, MSE(I,G), luonne mahdollistaa laskennallisten säästöjen aikaansaamisen. Nykyisen koodikirjavektorin jäljelle jäävät laskennat voidaan keskeyttää, kun on määritetty, että jäljelle jäävät MSE-arvot, jotka vielä on laskematta nykyiselle koodikirjavektorille, eivät voi johtaa MSE-arvoon, joka on pienempi kuin lukkopiiriin 250 talle-

tettu nykyinen MSE-minimi. Esimerkkisovellutuksessa minimointiprosessorissa 412 käytetään kolmea tekniikkaa laskennan säästämiseksi koodikirjahaussa. MSE-funktiot MSE(I,G) ovat neliöllisiä G:ssä. Yksi neliöllinen yhtälö muodostetaan kutakin koodikirjavektori-indeksiä I varten. Kaikki nämä neliölliset yhtälöt kulkevat origon kautta, G=0 ja MSE(I,G)=0.

Ensimmäiseen laskennan säästämismenetelmään kuuluu joko positiivisten tai negatiivisten koodikirja-arvojen haku riippuen Ex_cy_I:n merkistä. Negatiivinen Ex_cy_I:n arvo ja negatiivinen vahvistusarvo johtavat termin $-2GEx_cy_I$:n negatiiviseen arvoon yhtälössä 62. Positiivinen Ex_cy_I:n arvo ja positiivinen vahvistusarvo myös johtavat termin $-2GEx_cy_I$:n negatiiviseen arvoon yhtälössä 62. Koska termi $G^2Ey_Iy_I$ yhtälössä 62 on aina positiivinen, termin $-2GEx_cy_I$:n negatiivinen arvo yleensä minimoi MSE:n. Rengaspuskuriin 259 annetaan kaksi joukkoa koodikirjavahvistuspareja, toinen positiivisin koodikirjavahvistusarvoin ja toinen negatiivisin koodikirjavahvistusarvoin. Tällä tavoin joudutaan käyttämään ainoastaan neljää paria vahvistusarvoja kahdeksan vahvistusparin sijaan kullekin koodikirjavektori-indeksille I.

Toiseen laskennan säästämismenetelmään kuuluu jäljellä olevien MSE-arvojen laskennan keskeyttäminen nykyisen koodikirjavektorin koodikirjahakuproseduurissa perustuen MSE-funktion neliölliseen ominaisuuteen. MSE-funktio MSE(I,G) lasketaan koodikirjavahvistusarvoille, jotka kasvavat monotonisesti. Kun positiivinen MSE-arvo lasketaan nykyiselle koodikirjavektorille, kaikki jäljellä olevat MSE-laskennat nykyiselle koodikirjavektorille keskeytetään, koska vastaavat MSE-arvot ovat suurempia kuin nykyinen MSE-arvo.

Kolmanteen laskennan säästämismenetelmään kuuluu jäljellä olevien MSE-arvojen laskennan keskeyttäminen nykyisen koodikirjaindeksivektorin koodikirjahakuproseduurissa perustuen MSE-funktion neliölliseen ominaisuuteen. MSE-funktio MSE(I,G) lasketaan koodikirjavahvistusarvoille, jotka kasvavat monotonisesti. Kun nykyisessä koodikirjavektorissa lasketaan MSE-arvo, jota ei määritetty uudeksi MSE-minimiksi, ja kun nykyisessä koodikirjavektorissa laskettiin MSE-arvo, joka määritettiin uudeksi MSE-minimiksi, kaikki jäljellä olevat MSE-laskennat nykyisessä koodikirjavektorissa keskeytetään, koska jäljellä olevat MSE-arvot eivät voi olla pienempiä kuin uusi MSE-minimi. Kolme ylläkuvattua laskennan säästämismenetelmää aikaansaavat merkittäviä tehonsäästöjä minimointiprosessorissa 412.

Lohkossa 84 koodikirja-arvot kvantisoidaan. Lohko 86 tarkistaa onko kaikki koodikirja-alikehykset käsitelty. Jos kaikkia koodikirja-alikehyksiä ei ole käsitelty, niin palataan lohkoon 80. Jos kaikki koodikirja-alikehykset on käsitelty, niin edetään lohkoon 88. Lohko 88 tarkistaa onko kaikki äänentasoalikehykset käsitelty. Jos kaikkia äänentasoalikehyksiä ei ole käsitelty, niin palataan lohkoon 70. Jos kaikki äänentasoalikehykset on käsitelty, niin edetään lohkoon 90.

Lohkossa 90 koodatut tulokset pakataan tiettyyn muotoon. Täydellä nopeudella 22 bittiä dataa luetaan mikroprosessorilla (ei esitetty). Puolella nopeudella luetaan 10 tavua, 5 neljännesnopeudella ja 2 kahdeksasosanopeudella. Täydellä nopeudella generoidaan 11 pariteetin tarkistusbittiä virheenkorjauksen ja tunnistuksen aikaansaamiseksi täyden nopeuden 18:lle tärkeimmälle bitille.

Kooderin, lähettimessä, on ylläpidettävä dekooderin tilaa, vastaanottimessa, suodatinmuistien päivittämiseksi, joita vuorostaan käytetään kooderissa äänentaso- ja koodikirjahakuproseduureissa. Esimerkkisovelluksessa kooderiin kuuluu dekooderiversio, jota käytetään kunkin koodikirja-alikehyksen jälkeen.

Seuraavat dekoodausoperaatiot suoritetaan DSP-ytimessa 4 kooderin osana. Viitaten kuvioon 11, nykyiselle koodikirja-alikehykselle määritettyjä optimaalista koodikirjavektori-indeksiä Î ja optimaalista koodikirjavahvistusta $\hat{G}$, käytetään skaalatun koodikirjavektorin $C_d(n)$ generoimiseksi. Kaikilla muilla paitsi kahdeksasosanopeudella nykyiselle koodikirja-alikehykselle määritetty koodikirja 502 annetaan optimaalisen koodikirjaindeksin, Î, kanssa ja vastauksena annetaan vastaava herätevektori kertojan 504 ensimmäiseen tuloon. Kahdeksasosanopeuden tapauksessa generoidaan näennäissatunnainen sekvenssi $C_d(n)$:ää varten näennäissatunnaisvektorigeneraattorilla 500 ja annetaan kertojan 504 ensimmäiseen tuloon. Optimaalinen koodikirjavahvistus $\hat{G}$, joka on määritetty nykyiselle koodikirja-alikehykselle annetaan kertojan 504 toiseen tuloon. Tämä sekvenssi generoidaan samalla näennäissatunnaisgenerointioperaatiolla kuin mitä käytetään vastaanottimen dekooderissa.

Skaalatut koodikirjavektorit $C_d(n)$ annetaan äänentasosynteesisuodattimeen 506, joka generoi formanttijäännöksen $P_d(n)$. Äänentasosynteesisuodatinmuistit alustetaan lopputilalla, joka saadaan viimeisestä generoidusta puhenäytteestä. Äänentasosynteesisuodatin 506 käyttää optimaalista äänentasoviivettä $\hat{L}$ ja optimaalista äänentasovahvistusta $\hat{b}$, jotka on määritetty nykyiselle äänentasoalikehykselle. Kahdeksasosanopeudel-

le optimaalinen äänentasovahvistus asetetaan 0:ksi. Äänentasosynteesisuodatinmuistien lopputila ylläpidetään puheen generoimiseksi seuraavaa äänentasoalikehystä varten, kuten yllä mainittiin, käytettäväksi peräkkäisissä äänentasohauissa ja dekoodausoperaatioissa kooderissa.

Painotettu formanttisynteesisuodatin 508 generoi lähdön $Y_d(n)$ formanttijäännöksestä $P_d(n)$. Tämä suodatin alustetaan lopputilalla, joka saadaan viimeisestä generoidusta puhenäytteestä. LPC-vakiot, jotka on laskettu interpoloiduista LSP-arvoista nykyiselle alikehykselle, käytetään tämän suodattimen kertoimina. Tämän suodattimen lopputila talletetaan puheen generointia varten seuraavalle koodikirja-alikehykselle ja käytettäväksi seuraavissa äänentaso- ja koodikirjahauissa.

Dekoodausoperaatio, joka näytetään lohkoilla 44 ja 50 kuviossa 2, suoritetaan DSP-ytimessä 4. ASIC vastaanottaa paketin tietyssä muodossa mikroprosessorilta (ei esitetty) mikroprosessorirajapinnan 42 kautta. DSP-ydin 4 dekoodaa datan tässä paketissa ja käyttää sitä syntetisoimaan puhenäytteet, jotka annetaan koodekkiin (ei esitetty) PCM-rajapinnan 2 kautta. DSP-ytimessä vastaanotettu paketti puretaan puhenäytteiden syntetisoimiseen tarvittavan datan saamiseksi. Data sisältää koodausnopeuden, LSP-taajuudet ja äänentaso- ja koodikirjaparametrit vastaaville alikehyksille tällä nopeudella. Puhenäytteiden synteesi vastaanotetusta pakettidatasta suoritetaan DSP-ytimessä 4 ja esitetään kuviossa 12.

Viitaten kuvioon 12 optimaalinen koodikirjavektoriindeksi Î ja optimaalinen koodikirjavahvistus $\hat{G}$ vas-

taten nykyistä koodikirja-alikehystä ovat käytössä dekooderissa skaalattujen koodikirjavektorien $C_d(n)$ generoimiseksi. Paitsi kahdeksasosanopeudella, koodikirja 522 annetaan yhdessä nykyistä koodikirja-alikehystä vastaavan optimaalisen koodikirja-indeksin $\hat{I}$ kanssa ja vastauksena annetaan vastaava herätevektori kertojan 524 ensimmäiseen tuloon. Kahdeksasosanopeuden tapauksessa näennäissatunnainen sekvenssi $C_d(n)$:lle generoidaan näennäissatunnaisvektorigeneraattorilla 520 ja annetaan kertojan 524 ensimmäiseen tuloon. Tämä sekvenssi generoidaan samalla näennäissatunnaisgenerointioperaatiolla kuin mitä käytetään vastaanottimen dekooderissa. Optimaalinen koodikirjavahvistusarvo $\hat{G}$ vastaten nykyistä koodikirja-alikehystä annetaan kertojan 524 toiseen tuloon.

Skaalatut koodikirjavektorit $C_d(n)$ annetaan äänentasosynteesisuodattimeen 526, joka generoi formanttijäännöksen $P_d(n)$. Äänentasosynteesisuodattimen muistit alustetaan lopputilalla, joka saadaan viimeisestä generoidusta puhenäytteestä. Äänentasosynteesisuodatin 526 käyttää optimaalista äänentasoviivettä $\hat{L}$ ja optimaalista äänentasovahvistusta $\hat{b}$ vastaten nykyistä äänentasoalikehystä. Kahdeksasosanopeudella optimaalinen äänentasovahvistus asetetaan 0:ksi. Äänentasosynteesisuodattimen lopputila talletetaan käytettäväksi puheen generoinnissa seuraavalle äänentasoalikehykselle, kuten mainittiin yllä.

Painotettu formanttisynteesisuodatin 528 generoi lähdön $Y_d(n)$ formanttijäännöksestä $P_d(n)$. Tämä suodatin alustetaan lopputilalla, joka saadaan viimeisestä generoidusta puhenäytteestä. LPC-vakioita, jotka on laskettu interpoloiduista LSP-arvoista nykyiselle alikehykselle, käytetään suodattimen kertoimina. Suodatti-

men lopputila talletetaan puheen käytettäväksi generointiin seuraavalle koodikirja-alikehykselle.

Dekoodattu puhe $Y_d(n)$ annetaan jälkisuodattimeen 530, joka esimerkkisovellutuksessa on pitkäkestoinen jälkisuodatin perustuen LPC-kertoimiin nykyiselle dekoodattavalle alikehykselle. Jälkisuodatin 530 suodattaa uudelleenmuodostetut puhenäytteet $Y_d(n)$ ja antaa suodatetun puheen vahvistuksen ohjaukseen 532. Vahvistuksen ohjaus 532 ohjaa lähtöpuheen $s_d(n)$ tasoa ja sillä voidaan suorittaa automaattinen vahvistuksen ohjaus (AGC).

Edellä oleva edullisten sovellutusten kuvaus annetaan, jotta alan ammattimies voisi valmistaa tai käyttää esillä olevan keksintöä. Näiden sovellutusten eri modifikaatiot ovat alan ammattimiehille ilmeisiä ja tässä kuvatut yleiset periaatteet ovat sovellettavissa muihin sovellutuksiin käyttämättä keksimiskykyä. Näin ollen esillä olevaa keksintöä ei ole tarkoitettu rajata tässä esitettyihin sovellutuksiin vaan sille tulee suoja laajin piiri, joka on yhdenmukainen tässä esitettyjen periaatteiden ja piirteiden kanssa.

PATENTTIVAATIMUKSET

1. Digitaalisen signaaliprosessorin (DSP) ydinvälineet (4) käytettäväksi digitoidun puheen mallintamisessa, useiden arvojen rekursiivista konvoluutiota varten, **tunnettu** siitä, että käsittää laskentavälineet (143) mainitun rekursiivisen laskennan suorittamiseksi ja joukon hajasaantimuisteja (RAM:t) (104, 122, 182), jotka ovat alttiita laskentavälineille, jokainen RAM on vastaavan arvon tallentamista varten siten, että arvot voidaan hakea RAM:eista rinnakkain toistensa kanssa laskentavälineillä prosessointia varten.

2. Patenttivaatimuksen 1 mukaiset digitaalisen signaaliprosessorin (DSP) ydinvälineet (4) mainitun rekursiivisen laskennan suorittamiseksi ensimmäisen sekvenssin ja toisen sekvenssin mukaisesti, jossa mainittu joukko hajasaantimuisteja (104, 122, 182) käsittää:

   ensimmäinen hajasaantimuisti (RAM) (104) mainitun ensimmäisen sekvenssin tallentamiseksi ja antamiseksi;

   toinen hajasaantimuisti (RAM) (122) mainitun toisen sekvenssin tallentamiseksi ja antamiseksi;

   kolmas hajasaantimuisti (RAM) (182) ylimääräisen sekventiaalisen datan tallentamiseksi ja antamiseksi.

3. Patenttivaatimuksen 2 mukaiset digitaalisen signaaliprosessorin (DSP) ydinvälineet (4), jossa mainittu ensimmäinen RAM (104) on ensimmäisen arvojoukon, joka edustaa painosyntetisoituja puhe-

näytteitä, tallentamista varten, mainittu toinen RAM (122) on toisen arvojoukon, joka edustaa formanttisuodattimen impulssivasteita, tallentamista varten, ja mainittu kolmas RAM (182) on kolmannen arvojoukon, joka edustaa formanttijäännösten sekvenssiä, tallentamista varten.

4. Laite vokooderin toteuttamiseksi, **tunnettu** siitä, että käsittää:

jonkin patenttivaatimuksen 1-3 mukaiset digitaalisen signaaliprosessorin (DSP) ydinvälineet rekursiivisen konvoluutiolaskennan suorittamista varten ja mainitun rekursiivisen konvoluutiolaskennan tuloksen antamista varten; ja

minimointiprosessorivälineet (6), jotka ovat erilliset mainituista DSP-ydinvälineistä (4), mainitun rekursiivisen konvoluutiolaskennan tuloksen vastaanottamiseksi ja minimointietsinnän suorittamiseksi mainitun rekursiivisen konvoluutiolaskennan mainitun tuloksen mukaisesti rinnakkain DSP-ydinvälineiden (4), jotka suorittavat rekursiivisia konvoluutiolaskentoja, kanssa.

5. Patenttivaatimuksen 4 mukainen laite, joka lisäksi käsittää lohkonormalisointipiirin (136, 140, 150, 160, 162), joka sisältää siirtorekisterin digitoidun puheen näytteiden normalisoimiseksi ilman jakamista.

6. Patenttivaatimuksen 4 tai patenttivaatimuksen 5 mukainen laite, jossa:

mainituilla digitaalisen signaalin prosessoinnin (DSP) ydinvälineillä (4) on sisääntulo digitoidun äänidatan sekvenssin vastaanottamista varten, DSP-ydinvälineet generoivat datan synteettisten versioiden vastaavan sekvenssin; ja mainituilla minimointiprosessointivälineillä (6) on sisääntulo kytkettynä mainittuihin DSP-ydinvälineisiin (4) datan syntetisoitujen versioiden sekvenssin vastaanottamiseksi, minimointiprosessorivälineet (6) määräävät minimieron datan syntetisoidun version ja digitoidun äänidatan välillä ja takaavat sen määräävän funktion rinnakkain DSP-ydinvälineiden (4) kanssa, jotka takaavat sen generointifunktion.

7. Patenttivaatimuksen 6 mukainen laite, joka lisäksi käsittää digitoidun äänirajapinnan (2), jolla on sisääntulo ensimmäisen äänidatan vastaanottamiseksi ja ulostulo, jossa mainittu DSP-ydinvälineiden sisääntulo on kytketty mainittuun digitoidun äänirajapinnan (2) ulostuloon.

8. Patenttivaatimuksen 7 mukainen laite, joka lisäksi käsittää mikroprosessorirajapinnan (8), jolla on sisääntulo mikroprosessoridatan vastaanottamiseksi ja on ulostulo, jossa mainitulla digitoidulla äänirajapinnalla (2) on toinen sisääntulo ja jossa mainittu digitoidun äänirajapinnan (2) toinen sisääntulo on kytketty mainittuun mikroprosessorirajapinnan (8) ulostuloon.

9. Jonkin patenttivaatimuksen 4-8 mukainen laite, joka lisäksi käsittää kellogeneraattorin (10), jolla on sisääntulo kellosignaalin vastaanottamista var-

ten ja on ulostulo, jossa mainituilla DSP-ydinvälineillä (4) on toinen sisääntulo kytkettynä mainittuun kellogeneraattorin ulostuloon.

10. Jonkin patenttivaatimuksen 4-9 mukainen laite, jossa mainitut minimointiprosessorivälineet (6) vastaanottavat puhenäytteiden ensimmäisen sekvenssin ja syntetisoitujen puhenäytteiden toisen sekvenssin mainituilta digitaalisen signaaliprosessorin (DSP) ydinvälineiltä ja määräävät pienimmän keskimääräisen neliövirheen (MSE) sekvensseistä, minimointiprosessorivälineet (6) käsittävät:

logiikkavälineet toisen sekvenssin autokorrelaation määräämiseksi;

logiikkavälineet ristikorrelaation määräämiseksi ensimmäisen sekvenssin ja toisen sekvenssin välillä; ja

logiikkavälineet, jotka ovat alttiita autokorrelaatiolle ja ristikorrelaatiolle pienimmän MSE:n määräämiseksi.

11. Mobiilipuhelin, joka sisältää jonkin patenttivaatimuksen 4-10 mukaisen laitteen.

12. Menetelmä digitoidun äänidatan syntetisoimiseksi, joka käsittää takaavan rekursiivisen konvoluution ensimmäisen sekvenssin, joka edustaa syntetisoituja puhenäytteitä mobiilipuhelinjärjestelmässä, ja toisen sekvenssin, joka edustaa puhenäytteitä, välillä, **tunnettu** siitä, että menetelmä käsittää vaiheet:

suoritetaan alkukonvoluutio;

testataan onko vaiheindeksi parillinen;

noudetaan mainittu ensimmäinen sekvenssi ensimmäisestä muistielementistä (104);

noudetaan mainittu toinen sekvenssi toisesta muistielementistä (122);

noudetaan osittainen konvoluutiotulos kolmannesta muistielementistä (182), kun mainittu vaiheindeksi on parillinen;

suoritetaan osittainen konvoluutio mainitun noudetun ensimmäisen sekvenssin, mainitun noudetun toisen sekvenssin ja mainitun noudetun osittaisen konvoluutiotuloksen mukaisesti;

tallennetaan mainitun osittaisen konvoluution tulos mainittuun toiseen muistielementtiin (122), kun mainittu vaiheindeksi on parillinen;

noudetaan osittainen konvoluutio mainitusta toisesta muistielemetistä (122), kun mainittu vaiheindeksi on pariton;

suoritetaan osittainen konvoluutio mainitun noudetun ensimmäisen sekvenssin, mainitun noudetun toisen sekvenssin ja mainitun noudetun osittaisen konvoluution mukaisesti;

tallennetaan mainitun osittaisen konvoluution tulos mainittuun kolmanteen muistielementtiin (182), kun mainittu vaiheindeksi on pariton; ja

prosessoidaan konvoluutiotulosta sekvenssien välisen eron minimoimiseksi digitoidun äänidatan tarkan synteesin antamiseksi.

13. Patenttivaatimuksen 12 mukainen menetelmä,

joka lisäksi käsittää aikaisemmin generoidun konvoluutiotuloksen noutamisen rengaspuskurista (259).

Patentkrav

1. En digital signalprocessors (DSP) kärnmedel (4) för modellering av digitalt tal, för rekursiv konvolution av flera värden, kännetecknat därav, att den omfattar räknemedel (143) för utförande an den nämnda rekursiva uträkningen och en grupp direktminnen (RAM) (104, 122, 182)m som är åtkomliga för räknemedlen, vart och ett av RAM:en är för förvaring av ett motsvarande värde så att värdena kan sökas ur RAM:en parallellt med varandra för processering med räknemedlen.

2. En digital signalprocessors (DSP) kärnmedel (4) enligt patentkrav 1, för genomföring av nämnda rekursiva uträkning enligt en första sekvens och en andra sekvens, där den nämnda gruppen direktminnen (104, 122, 182) omfattar:
ett första direktminne (RAM) (104) för förvaring och givande av den första sekvensen;
ett andra direktminne (RAM) (122) för förvaring och givande av den andra sekvensen;
ett tredje direktminne (RAM) (182) för förvaring och givande av extra sekventiellt data.

3. En digital signalprocessors (DSP) kärnmedel (4) enligt patentkrav 2, där nämnda första RAM (104) är för förvaring av en första värdegrupp som representerar viktsyntetiserade talprov, nämnda andra RAM (122) är förförvaring av en andra värdegrupp som representerar ett formantfilters impulssvar, och nämnda tredje RAM (182) är för förvaring av en tredje värdegrupp som representerar en sekvens formantrester.

4. Anordning för förverkligande av en vokoder, **kännetecknad** därav, att den omfattar:

den digitala signalprocessorns (DSP) kärnmedel enligt något av patentkraven 1-3, för genomföring av rekursiv konvolutionsuträkning och för givande av den rekursiva konvolutionsuträkningens resultat; och minimeringsprocesseringsmedel (6) som är skilda från de nämnda DSP-kärnmedlen (4) för mottagning av den nämnda rekursiva konvolutionsuträkningens resultat och för utförande av minimeringssökningen enligt det nämnda resultatet av den nämnda konvolutionsuträkningen parallellt med DSP-kärnmedlen (4) som utför rekursiva konvolutionsuträkningar.

5. Anordning enligt patentkrav 4, som vidare omfattar en blocknormaliseringskrets (136, 140, 150, 160, 162) som innehåller ett skiftregister för normalisering av digitala talprov utan delning.

6. Anordning enligt patentkrav 4 eller patentkrav 5, där:

nämnda digitala signalprocesserings (DSP) kärnmedel (4) har en ingång för mot tagning av en sekvens digitaliserat audiodata, DSP-kärnmedlen genererar en motsvarande sekvens av syntetiserade versioner av datat; och nämnda minimeringsprocesseringsmedel (6) har en ingång kopplad till de nämnda DSP-kärnmedlen (4) för mottagning av sekvensen av de syntetiserade versionerna av datat, då minimeringsprocesseringsmedlen (6) bestämmer minimiskillnaden mellan den syntetiserade versionen av datat och det digitaliserade audiodatat och garanterar dess bestämmande funktion paral-

lellt med DSP-kärnmedlen (4) som garanterar dess genereringsfunktion.

7. Anordning enligt patentkrav 6, som vidare omfattar ett digitaliserat ljudgränssnitt (2) med en ingång för mottagning av ett första ljuddata och en utgång, där nämnda DSP-kärnmedels (4) ingång är kopplad till nämnda digitaliserade ljudgränssnitts (2) utgång.

8. Anordning enligt patentkrav 7, som vidare omfattar ett mikroprocessorgränssnitt (8), som har en ingång för mottagning av mikroprocessordata och en utgång, där det nämnda digitala ljudgränssnittet (2) har en andra ingång och där det nämnda digitaliserade ljudgränssnittets (2) andra ingång är kopplad till nämnda mikroprocessorgränssnitts (8) utgång.

9. Anordning enligt något av patentkraven 4-8, som vidare omfattar en klockgenerator (10), som har en ingång för mottagning av en klocksignal och en utgång, där nämnda DSP-kärnmedel (4) har en andra ingång kopplad till nämnda klockgenerators utgång.

10. Anordning enligt något av patentkraven 4-9, där de nämnda minimeringsprocessormedlen (6) mottar en första sekvens av talprov och en andra sekvens av syntetiserade talprov från nämnda digitala signalprocessors (DSP) kärnmedel (4) och bestämmer den minsta medelkvadratavvikelsen (MSE) för sekvenserna, då minimeringsprocessormedlen (6) omfattar:

logikmedel för bestämning av den andra sekvensens autokorrelation;

logikmedel för bestämning av en korskorrelation mellan den första sekvensen och den andra sekvensen; och

logikmedel som är mottagliga för autokorrelationen och korskorrelationen för bestämning av ett minimi-MSE.

11. En mobiltelefon som omfattar en anordning enligt något av patentkraven 4-10.

12. Ett förfarande för syntetisering av digitaliserat audiodata, som omfattar utförande av rekursiv konvolution mellan en första sekvens som representerar syntetiserade talprov i ett mobiltelefonsystem och en andra sekvens som representerar talprover, kännetecknat därav, att förfarandet omfattar faserna för:

utförande av en startkonvolution;

testning av om stegindexet är jämnt;

hämtning av den nämnda första sekvensen från ett första minneselement (104);

hämtning av den nämnda andra sekvensen från ett andra minneselement (122);

hämtning av ett partiellt konvolutionsresultat från ett tredje minneselement (182) då det nämnda stegindexet är jämnt;

utförande av en partiell konvolution enligt den nämnda hämtade första sekvensen, den nämnda hämtade andra sekvensen och det nämnda hämtade partiella konvolutionsresultatet;

sparning av resultatet av den nämnda partiella konvolutionen i det nämnda andra minneselementet (122) då det nämnda stegindexet är jämnt;

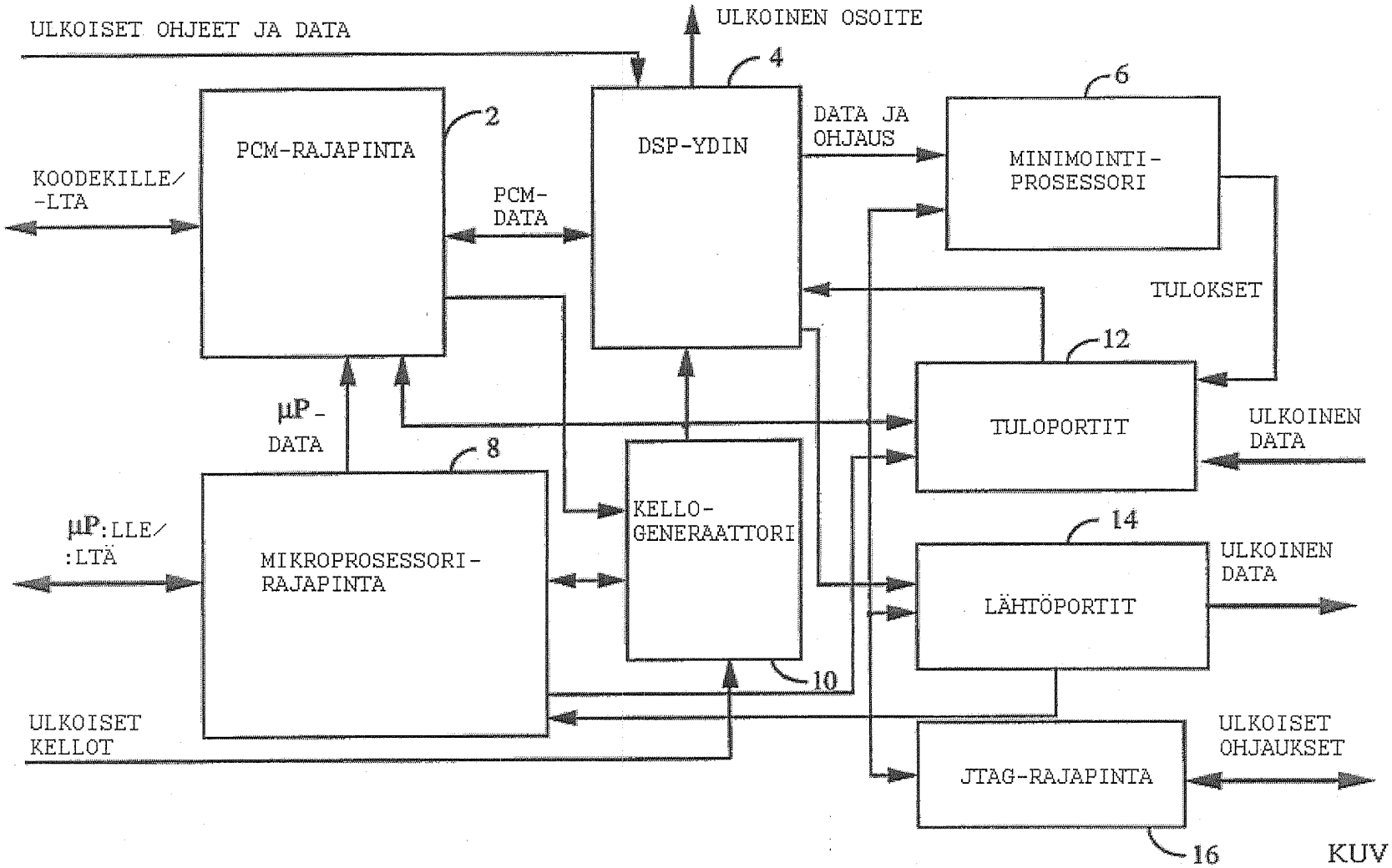
hämtning av en partiell konvolution från det nämnda andra minneselementet (122) då det nämnda stegindexet är udda;

utföring av en partiell konvolution enligt den nämnda hämtade första sekvensen, den nämnda hämtade andra sekvensen och den nämnda hämtade partiella konvolutionen;

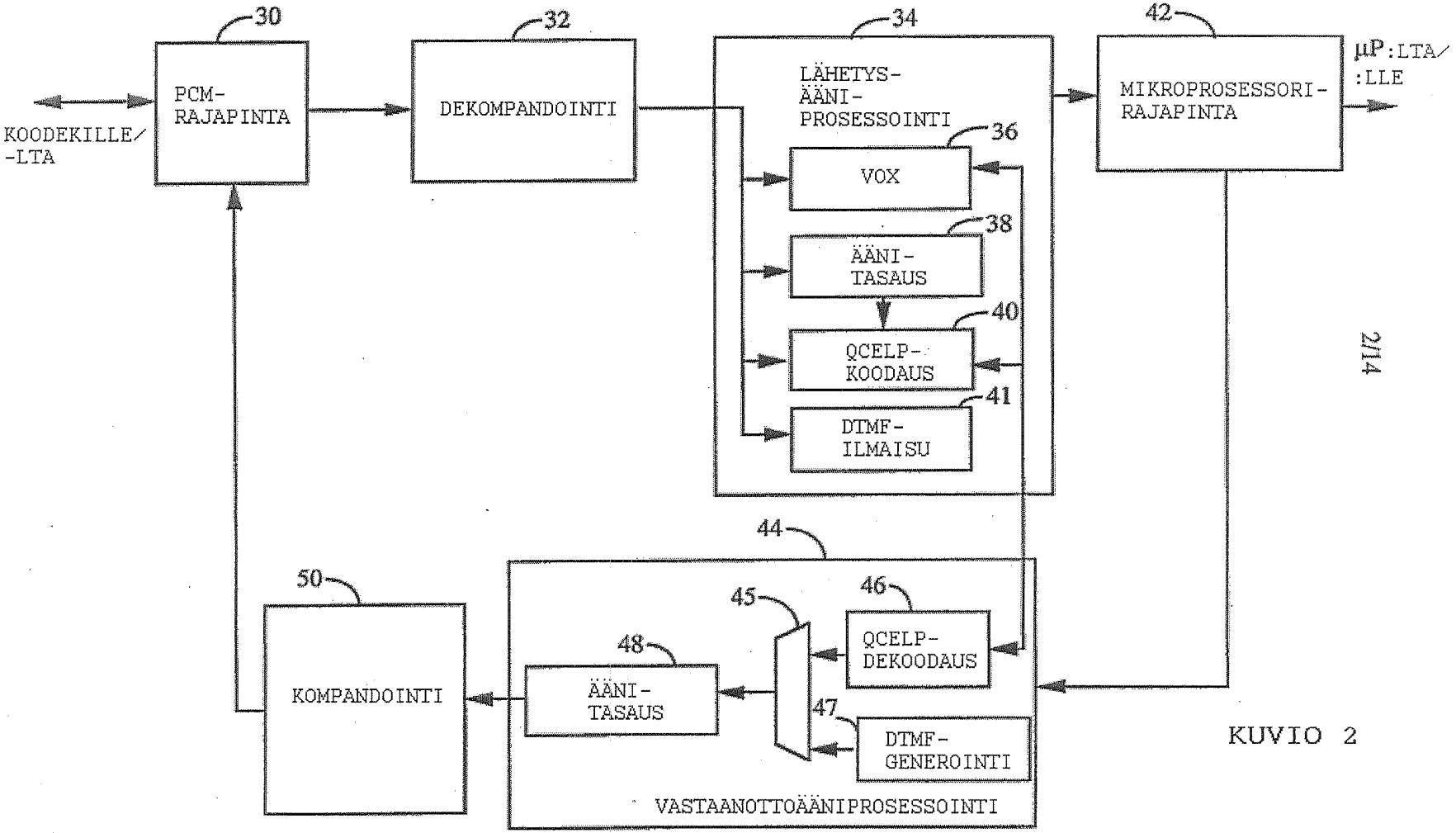
sparning av resultatet av den nämnda partiella konvolutionen i det nämnda tredje minneselementet (182) då det nämnda stegindexet är udda; och

processering av konvolutionsresultatet, för att minimera skillnaden mellan sekvenserna för att ge en exakt syntes av det digitaliserade audiodatat.

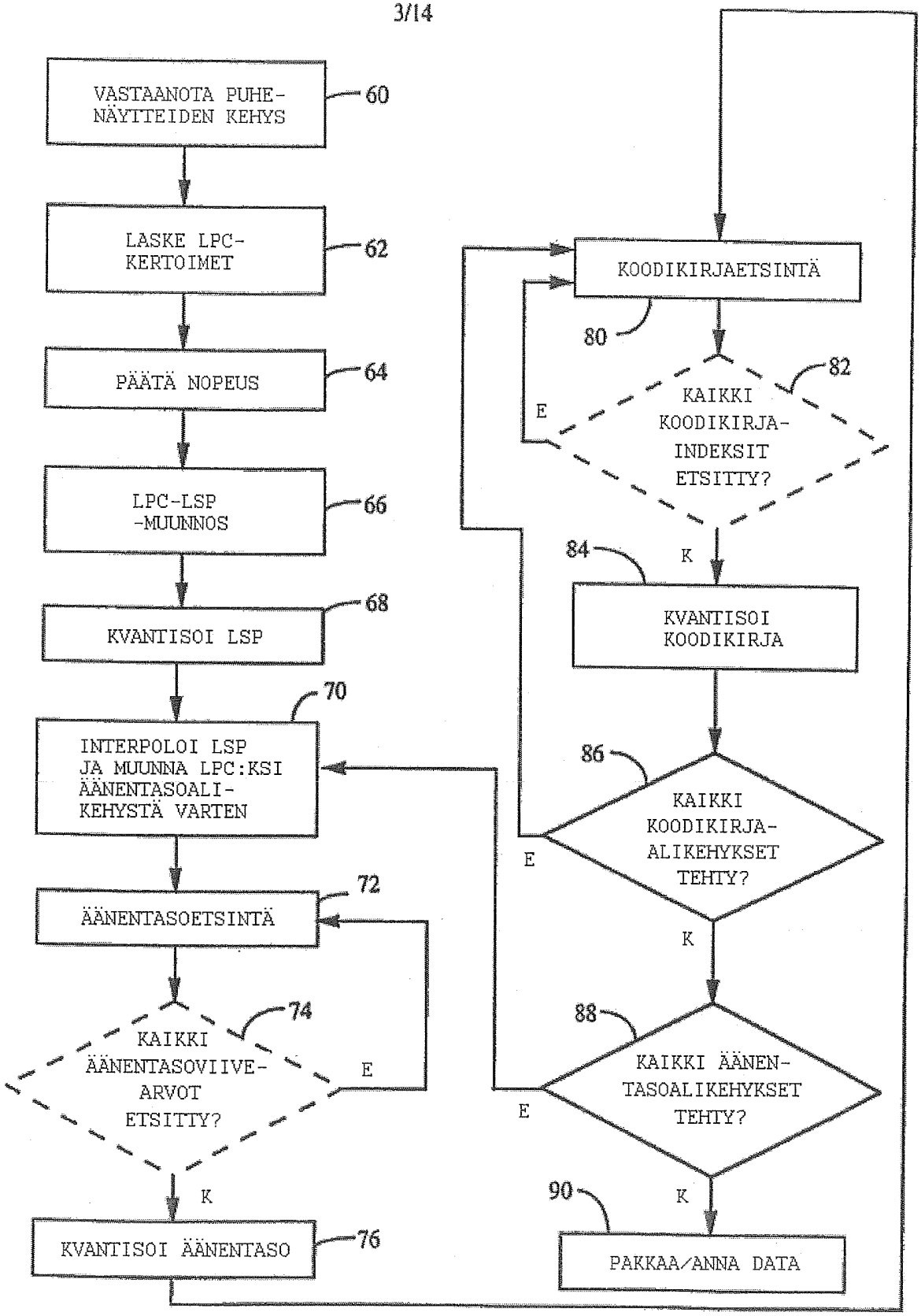
13. Förfarande enligt patentkrav 12, som vidare omfattar hämtning av ett tidigare genererat konvolutionsresultat från en cirkulär buffert.


ULKOISET OHJEET JA DATA
ULKOINEN OSOITE
4
6
2
PCM-RAJAPINTA
DSP-YDIN
DATA JA OHJAUS
MINIMOINTI-PROSESSORI
KOODEKILLE/ -LTA
PCM-DATA
TULOKSET
12
µP-DATA
TULOPORTIT
ULKOINEN DATA
8
KELLO-GENERAATTORI
µP:LLE/ :LTÄ
14
MIKROPROSESSORI-RAJAPINTA
ULKOINEN DATA
LÄHTÖPORTIT
10
ULKOISET KELLOT
ULKOISET OHJAUKSET
JTAG-RAJAPINTA
16

KUVIO 1

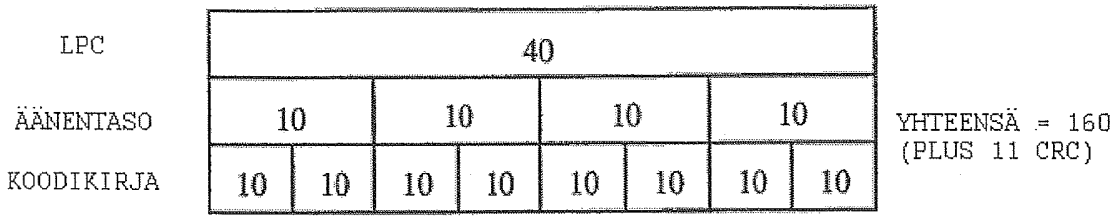

30
PCM-
RAJAPINTA
KOODEKILLE/
-LTA
32
DEKOMPANDOINTI
34
LÄHETYS-
ÄÄNI-
PROSESSOINTI
36
VOX
38
ÄÄNI-
TASAUS
40
QCELP-
KOODAUS
41
DTMF-
ILMAISU
42
MIKROPROSESSORI-
RAJAPINTA
µP:LTA/
:LLE
44
45
46
QCELP-
DEKOODAUS
48
ÄÄNI-
TASAUS
47
DTMF-
GENEROINTI
50
KOMPANDOINTI
VASTAANOTTOÄÄNIPROSESSOINTI

KUVIO 2


VASTAANOTA PUHE-NÄYTTEIDEN KEHYS
60
LASKE LPC-KERTOIMET
62
PÄÄTÄ NOPEUS
64
LPC-LSP -MUUNNOS
66
KVANTISOI LSP
68
INTERPOLOI LSP JA MUUNNA LPC:KSI ÄÄNENTASOALI-KEHYSTÄ VARTEN
70
ÄÄNENTASOETSINTÄ
72
KAIKKI ÄÄNENTASOVIIVE-ARVOT ETSITTY?
74
E
K
KVANTISOI ÄÄNENTASO
76
KOODIKIRJAETSINTÄ
80
KAIKKI KOODIKIRJA-INDEKSIT ETSITTY?
82
E
K
KVANTISOI KOODIKIRJA
84
KAIKKI KOODIKIRJA-ALIKEHYKSET TEHTY?
86
E
K
KAIKKI ÄÄNEN-TASOALIKEHYKSET TEHTY?
88
E
K
PAKKAA/ANNA DATA
90

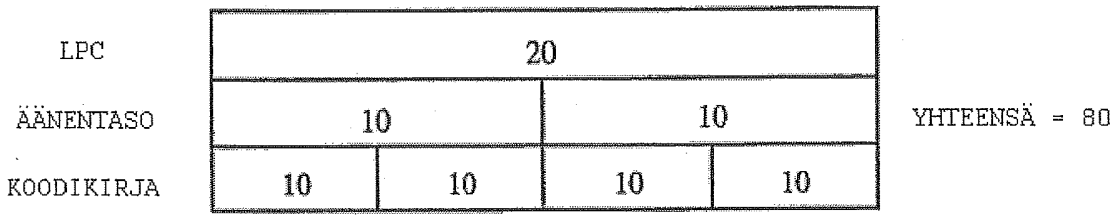
KUVIO 3

TÄYSI NOPEUS



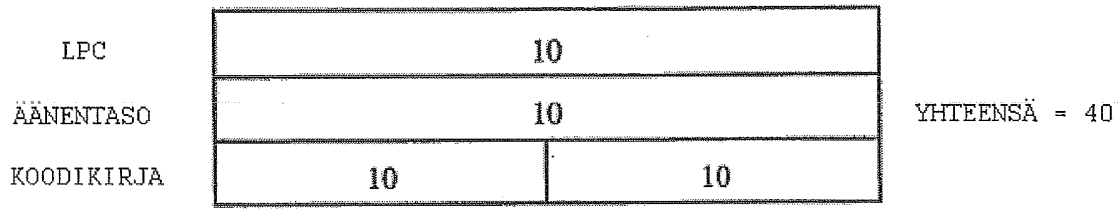
KUVIO 4a

PUOLINOPEUS



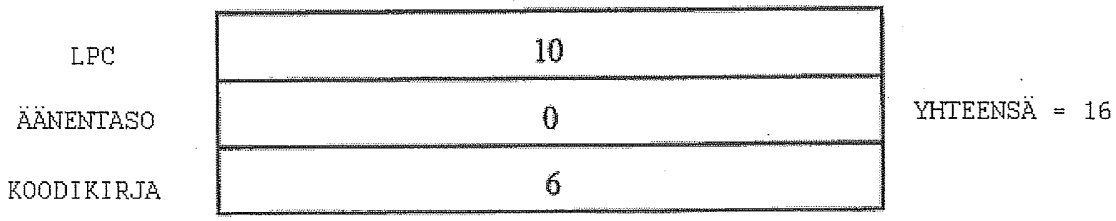
KUVIO 4b

NELJÄSOSANOPEUS

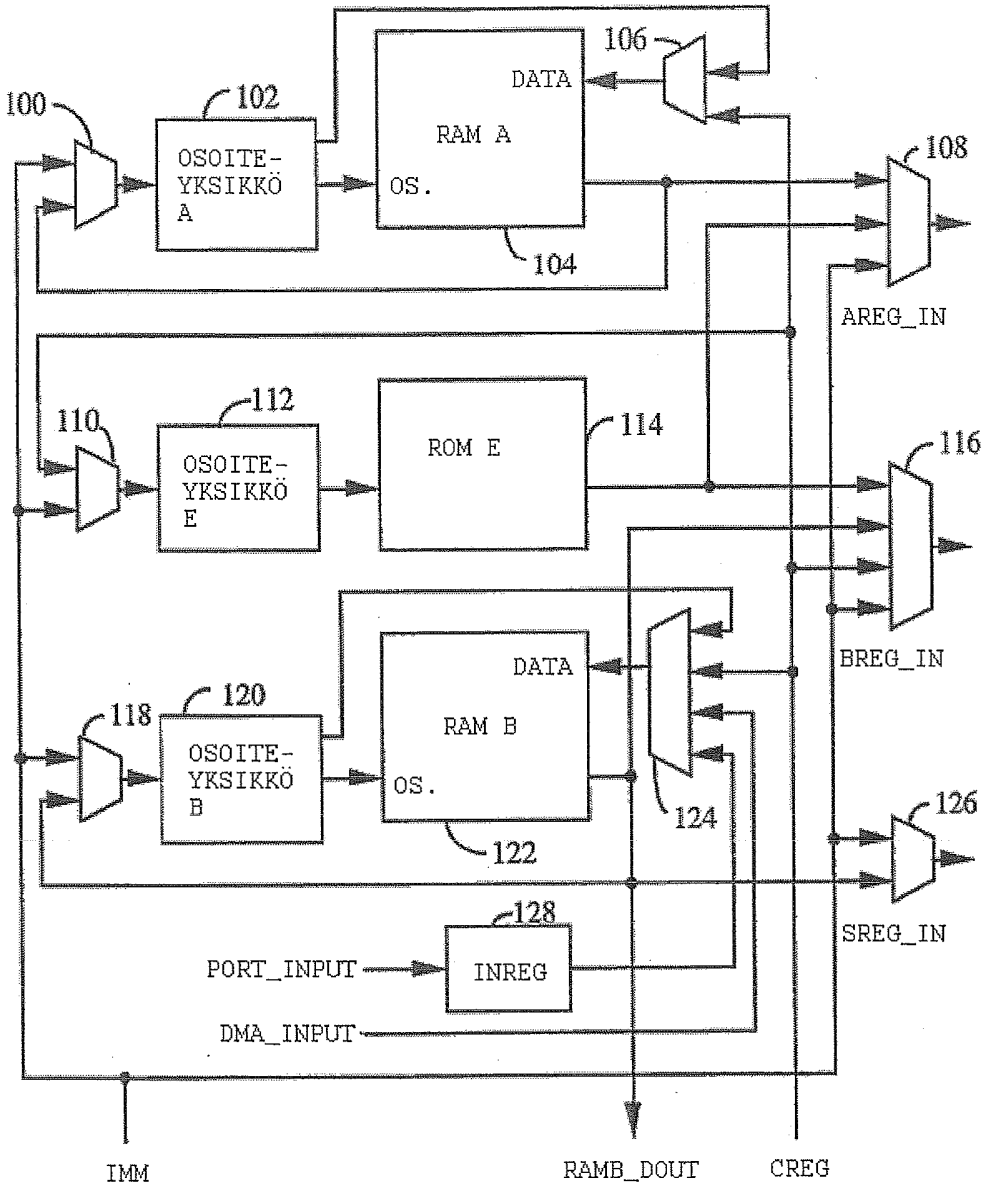


KUVIO 4c

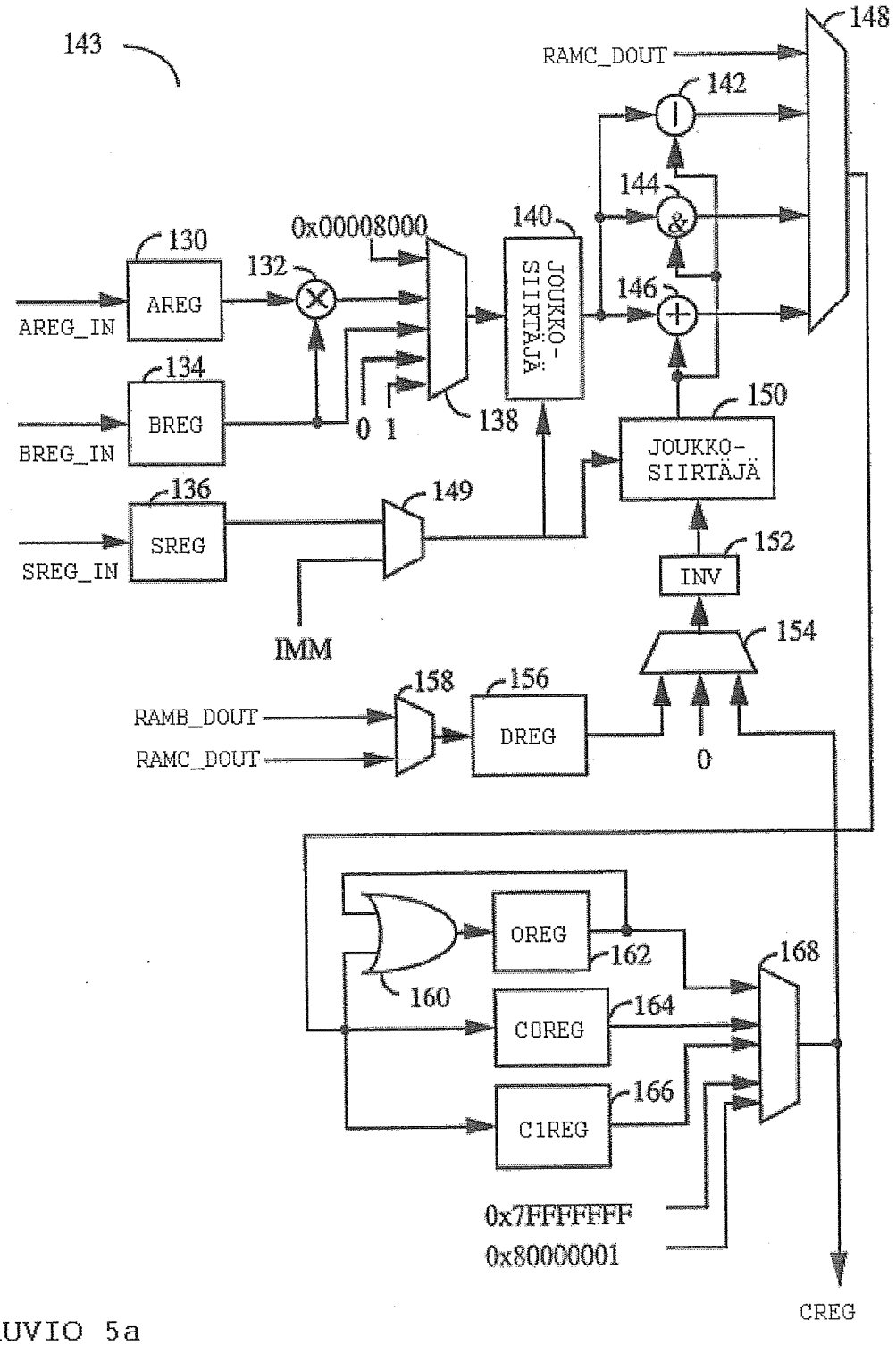
KAHDEKSASOSANOPEUS



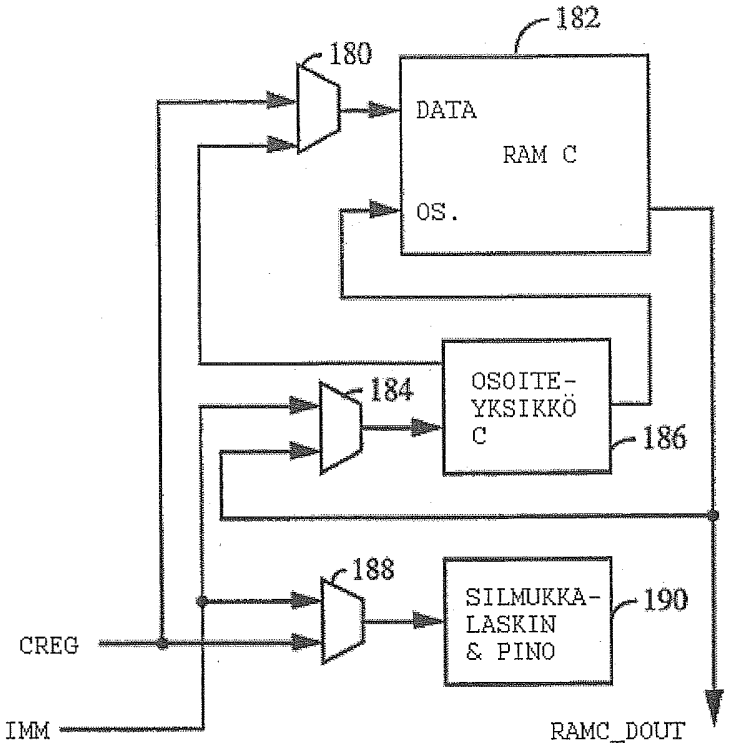
KUVIO 4d


100
102
OSOITE-
YKSIKKÖ
A
DATA
RAM A
OS.
104
106
108
AREG_IN
110
112
OSOITE-
YKSIKKÖ
E
ROM E
114
116
BREG_IN
118
120
OSOITE-
YKSIKKÖ
B
DATA
RAM B
OS.
122
124
126
SREG_IN
128
PORT_INPUT
INREG
DMA_INPUT
IMM
RAMB_DOUT
CREG

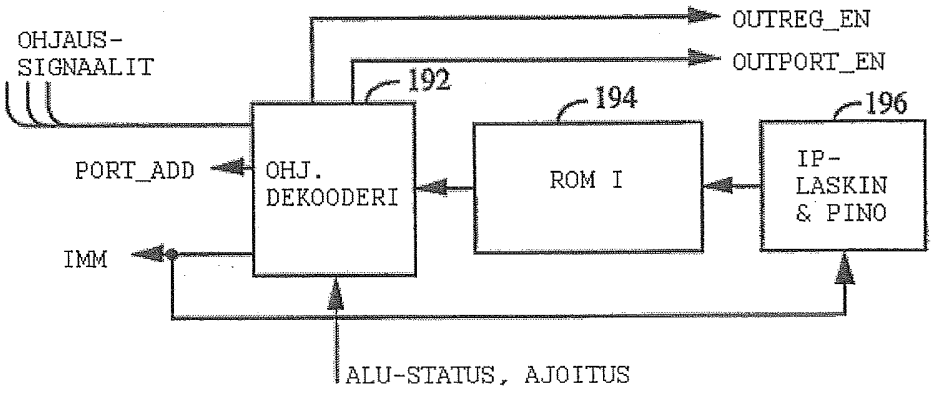
KUVIO 5a


143
148
RAMC_DOUT
142
144
&
140
JOUKKO-
SIIRTÄJÄ
0x00008000
130
AREG
132
AREG_IN
146
+
134
BREG
BREG_IN
0 1
138
150
JOUKKO-
SIIRTÄJÄ
136
SREG
149
SREG_IN
152
INV
154
IMM
158
156
RAMB_DOUT
RAMC_DOUT
DREG
0
OREG
162
168
160
COREG
164
C1REG
166
0x7FFFFFFF
0x80000001
CREG

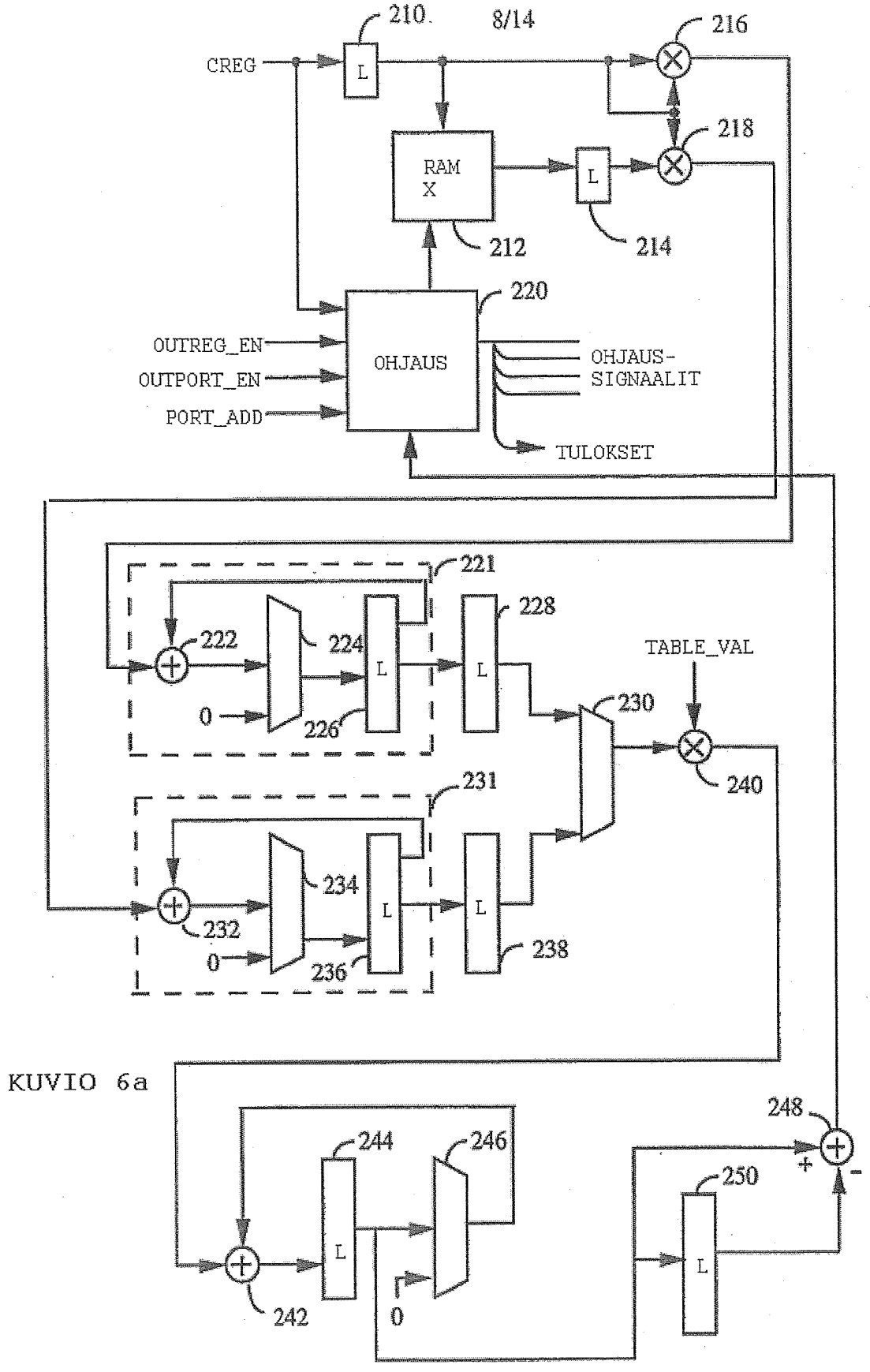
KUVIO 5a



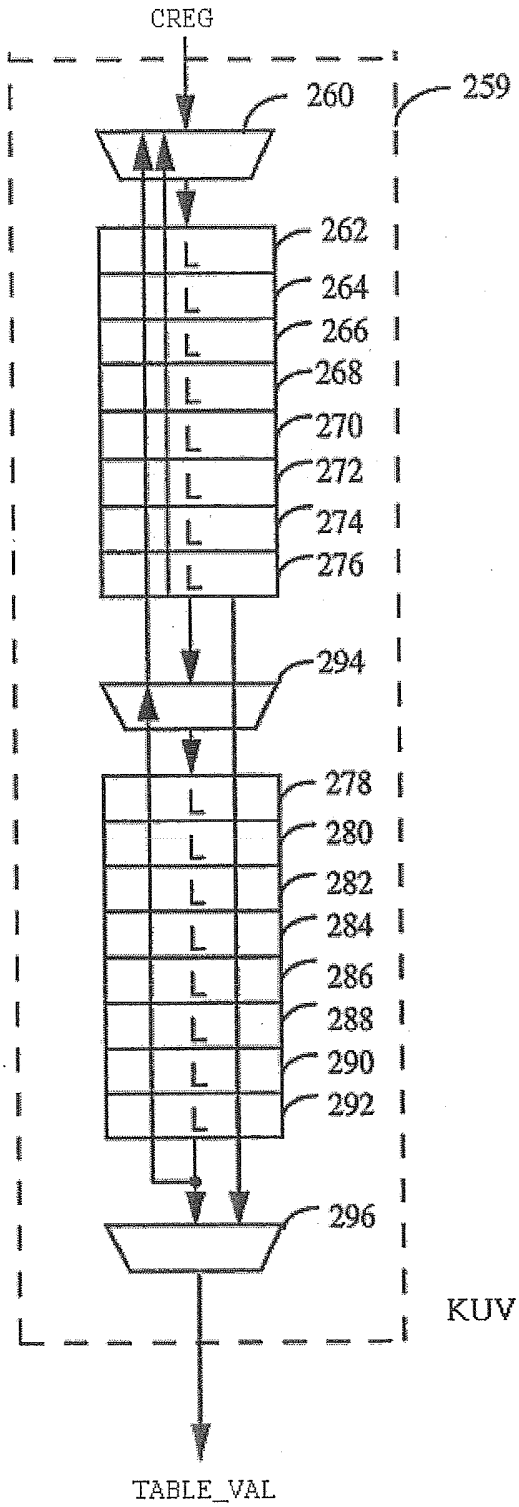
KUVIO 5c



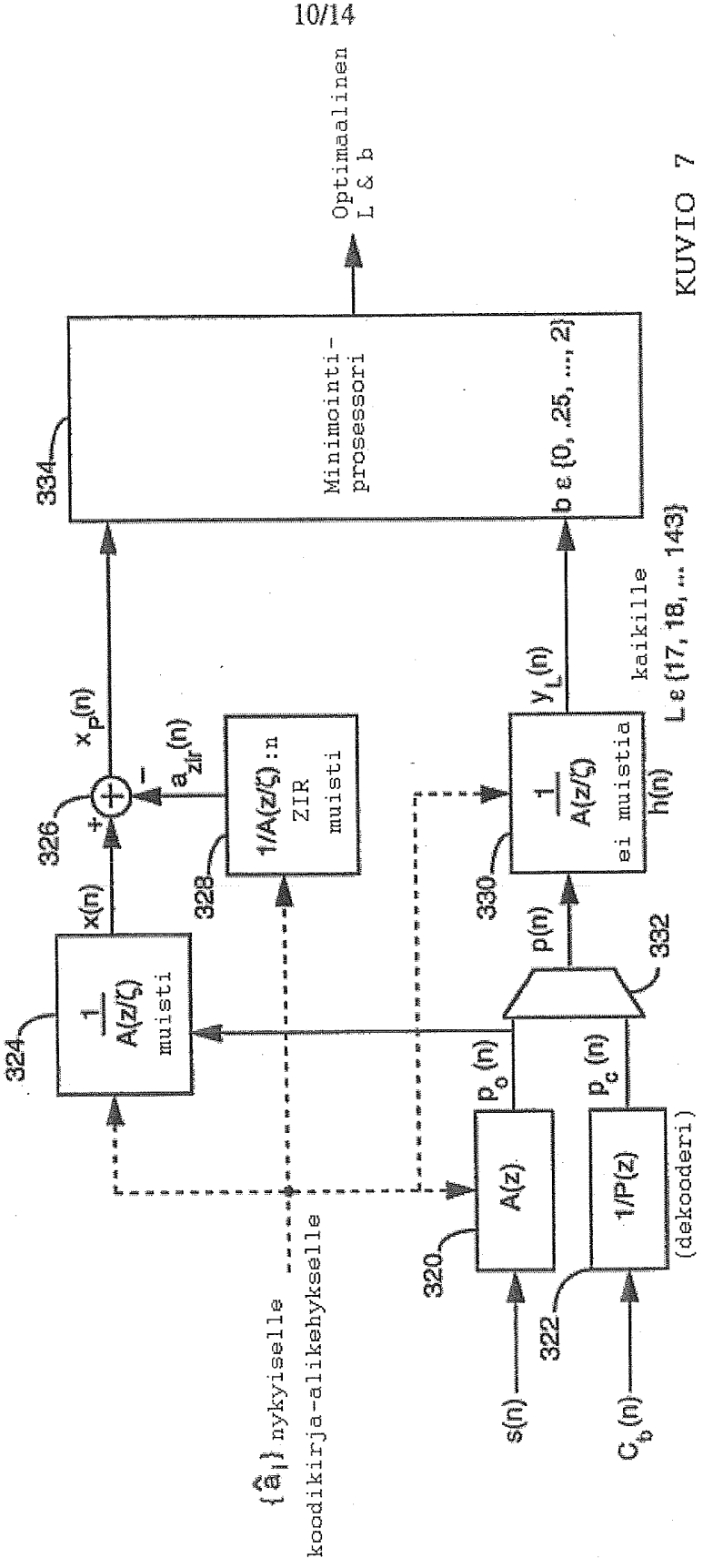
KUVIO 5d


210
CREG
L
216
218
RAM
X
L
212
214
220
OUTREG_EN
OUTPORT_EN
PORT_ADD
OHJAUS
OHJAUS-
SIGNAALIT
TULOKSET
221
222
224
228
TABLE_VAL
0
226
230
240
231
234
232
0
236
238
248
244
246
250
242
0
+
-

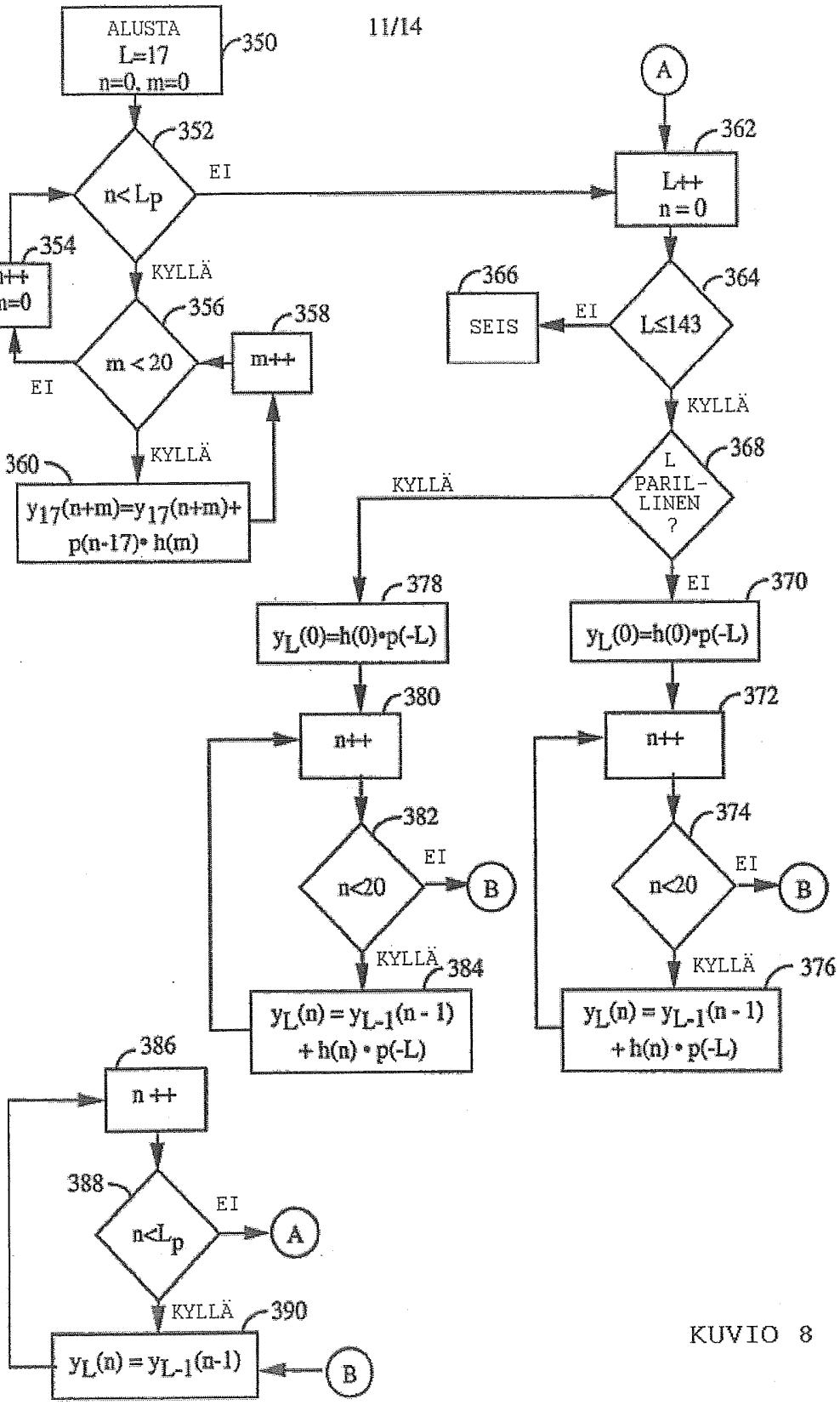
KUVIO 6a


CREG
260
259
262
264
266
268
270
272
274
276
294
278
280
282
284
286
288
290
292
296
TABLE_VAL

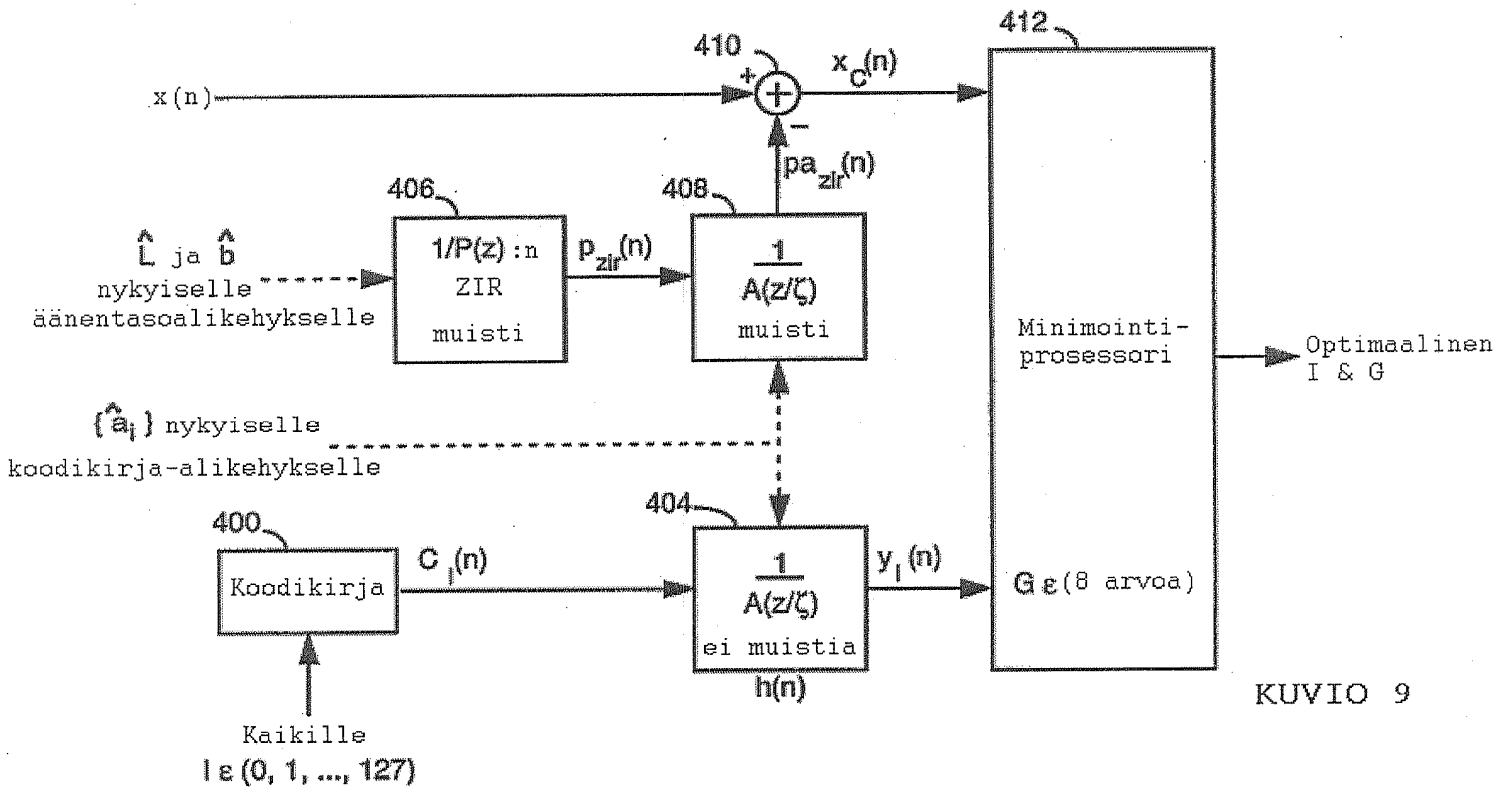
KUVIO 6b


Optimaalinen
L & b
Minimointi-
prosessori
334
b ε {0, .25, ..., 2}
L ε {17, 18, ... 143}
kaikille
y_L(n)
1/A(z/ζ)
ei muistia
h(n)
330
p(n)
332
x_p(n)
326
a_zir(n)
1/A(z/ζ) :n
ZIR
muisti
328
x(n)
1/A(z/ζ)
muisti
324
p_o(n)
p_c(n)
A(z)
1/P(z)
(dekooderi)
320
322
s(n)
C_b(n)
{â_i} nykyiselle
koodikirja-alikehykselle

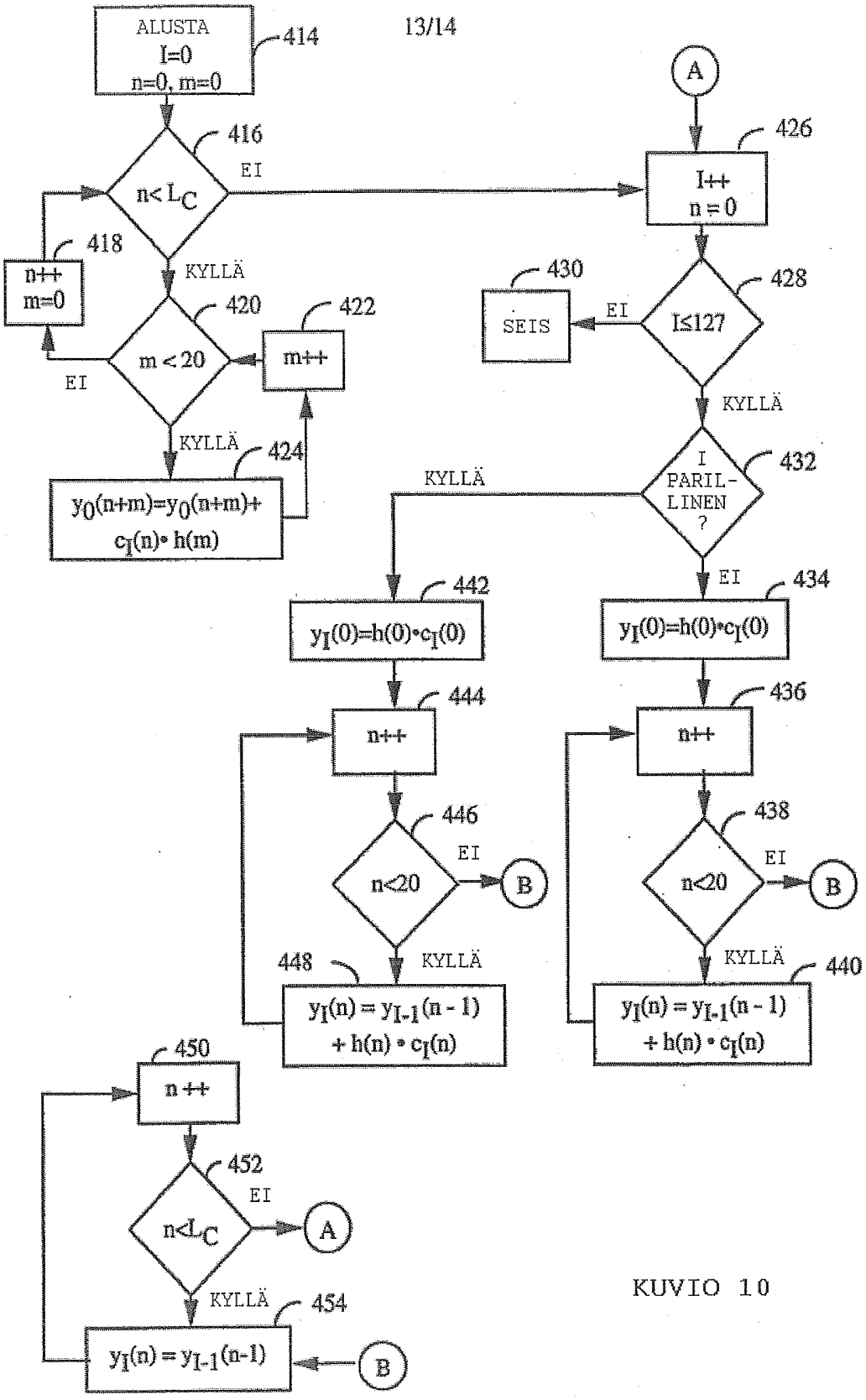
KUVIO 7


ALUSTA
L=17
n=0, m=0
350
352
n< Lp
EI
KYLLÄ
354
n++
m=0
356
m < 20
EI
KYLLÄ
358
m++
360
y17(n+m)=y17(n+m)+
p(n-17)• h(m)
A
362
L++
n = 0
364
L≤143
EI
366
SEIS
KYLLÄ
368
L
PARIL-
LINEN
?
KYLLÄ
EI
370
yL(0)=h(0)•p(-L)
372
n++
374
n<20
EI
B
KYLLÄ
376
yL(n) = yL-1(n - 1)
+ h(n) • p(-L)
378
yL(0)=h(0)•p(-L)
380
n++
382
n<20
EI
B
KYLLÄ
384
yL(n) = yL-1(n - 1)
+ h(n) • p(-L)
386
n ++
388
n<Lp
EI
A
KYLLÄ
390
yL(n) = yL-1(n-1)
B

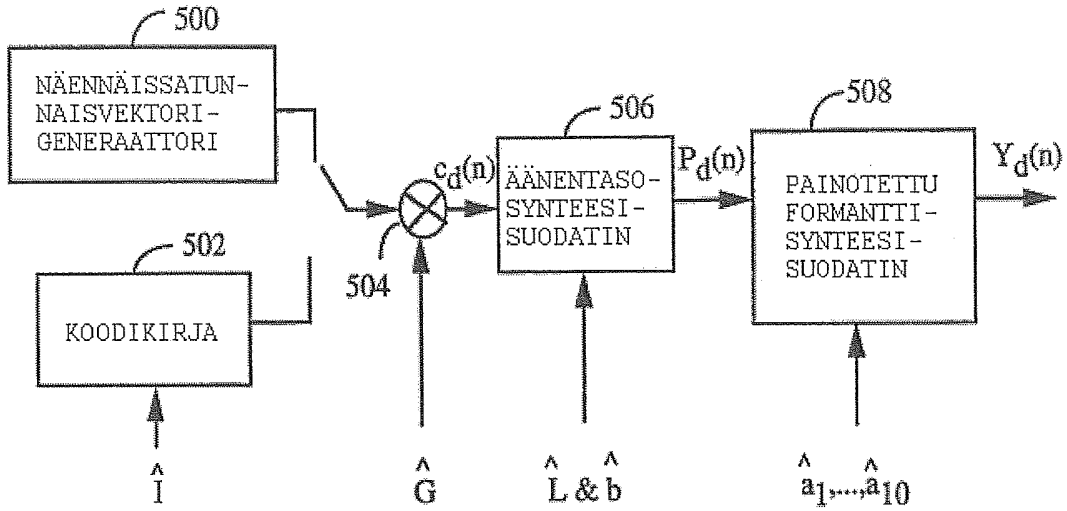
KUVIO 8


x(n)
410
+
−
$x_C(n)$
412
$pa_{zir}(n)$
406
408
$\hat{L}$ ja $\hat{b}$
nykyiselle
äänentasoalikehykselle
1/P(z) :n
ZIR
muisti
$p_{zir}(n)$
$\frac{1}{A(z/\zeta)}$
muisti
Minimointi-
prosessori
Optimaalinen
I & G
$\{\hat{a}_i\}$ nykyiselle
koodikirja-alikehykselle
404
400
Koodikirja
$C_I(n)$
$\frac{1}{A(z/\zeta)}$
ei muistia
h(n)
$y_I(n)$
G ε (8 arvoa)
Kaikille
I ε (0, 1, ..., 127)

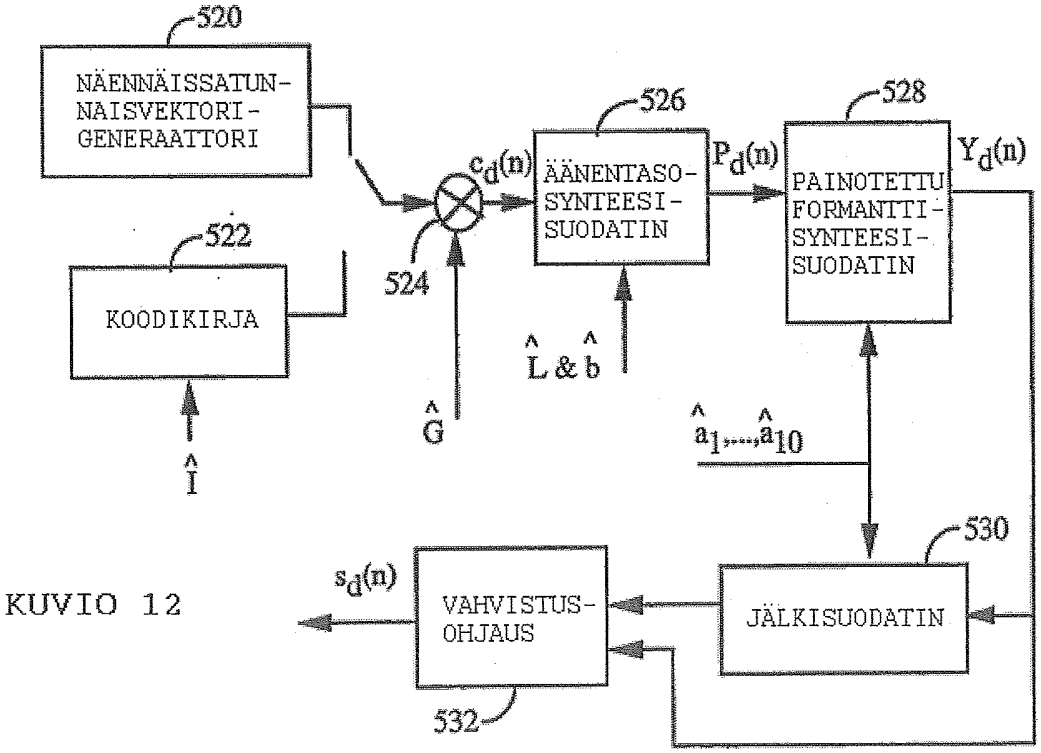
KUVIO 9


ALUSTA
I=0
n=0, m=0
414
416
n< L_C
EI
KYLLÄ
418
n++
m=0
420
m < 20
EI
422
m++
KYLLÄ
424
$y_0(n+m)=y_0(n+m)+$
$c_I(n)\cdot h(m)$
A
426
I++
n = 0
428
I≤127
EI
430
SEIS
KYLLÄ
432
I PARIL-LINEN ?
KYLLÄ
EI
434
$y_I(0)=h(0)\cdot c_I(0)$
436
n++
438
n<20
EI
B
KYLLÄ
440
$y_I(n) = y_{I-1}(n-1)$
$+ h(n) \cdot c_I(n)$
442
$y_I(0)=h(0)\cdot c_I(0)$
444
n++
446
n<20
EI
B
448
KYLLÄ
$y_I(n) = y_{I-1}(n-1)$
$+ h(n) \cdot c_I(n)$
450
n ++
452
n<L_C
EI
A
KYLLÄ
454
$y_I(n) = y_{I-1}(n-1)$
B

KUVIO 10



KUVIO 11



KUVIO 12