Octrooiraad

(10) A **Terinzagelegging** (11) **7920197**

Nederland (19) **NL**

(54) **Informatieverwerkende inrichting met een op-code verlengd register.**

(51) Int.Cl³.: G06F9/06, G06F7/38, G11C8/00.

(71) Aanvrager: Western Electric Company, Incorporated te New York.

(74) Gem.: Ir. H.M. Urbanus c.s.
Vereenigde Octrooibureaux
Nieuwe Parklaan 107
2587 BP 's-Gravenhage.

(21) Aanvrage Nr. 7920197.

(86) Aanvraagnummer oorspronkelijke internationale aanvrage: PCT/US79/01045.

(22) Ingediend 6 december 1979.

(32) Voorrang vanaf 29 december 1978.

(33) Land van voorrang: Ver. St. v. Am. (US).

(31) Nummer van de voorrangsaanvrage: 974426 .

(62) - -

(43) Ter inzage gelegd 28 november 1980.

(87) Publicatiedatum oorspronkelijke internationale aanvrage: 10 juli 1980.

(87) Publicatienummer oorspronkelijke internationale aanvrage: WO80/01423.

Deze octrooiaanvrage werd ingediend als internationale octrooiaanvrage onder de bepalingen van het Verdrag tot samenwerking inzake octrooien (PCT). De aan dit blad gehechte stukken zijn een afdruk van een Nederlandse vertaling van de oorspronkelijk in een andere taal ingediende beschrijving met conclusie(s) en tekening(en). De Nederlandse octrooiaanvrage wordt geacht te zijn ingediend op de indieningsdatum van de internationale octrooiaanvrage.

VO 805

Informatieverwerkende inrichting met een op-code verlengd register

De uitvinding heeft betrekking op een centrale verwerkingseenheid (CPU) van een informatieverwerkingssysteem voor het met behulp van een reeks instructiecycli uitvoeren van een programma van opgeslagen instructies, waarbij de centrale verwerkingseenheid geschikt is voor het ontvangen van een instructie tijdens de "ophaal"-fase-instructie van een instructiecyclus en, in responsie op de instructie, het tijdens de informatie uitvoerfase van de instructiecyclus uitvoeren van een rekenbewerking op de opgeslagen informatie gespecificeerd door de instructie, waarbij elke opgeslagen instructie voorzien is van een op-codegedeelte en een adresgedeelte, waarbij de centrale verwerkingseenheid voorzien is van een instructieregister voor het tijdens de instructie-ophaalfase van de instructiecyclus ontvangen van het op-codegedeelte van de instructie en voor het opslaan van de op-code tijdens de gehele instructiecyclus en van een besturingsinrichting voor het tijdens de informatiebewerkingsfase van de instructiecyclus opwekken van commandosignalen die corresponderen met de rekenbewerking.

De op-code van een instructie is een groep van binaire cijfers (bits) die een rekenbewerking bepalen, zoals OPTELLEN, AFTREKKEN, COMPLEMENT, etc. Het instellen van de rekenbewerkingen, geformuleerd voor een CPU hangt af van de soort van bewerking die moet worden uitgevoerd, Het totale aantal verschillende bewerkingen, die door de CPU kunnen worden uitgevoerd bepaalt de waarde van de rekenbewerkingen.

Het aantal bits dat de op-code (op-codeveld) vormt is een functie van het aantal bewerkingen dat mogelijk is. Er zijn ten minste N-bits nodig voor het definiëren van 2^N of minder verschillende bewerkingen. De ontwerper van de CPU geeft aan elke bewerking een andere bitcombinatie (d.w.z. op-code). Een stuursectie van de CPU detecteert op het juiste tijdstip na elkaar de bitcombinatie en voert de juiste commandosignalen naar de vereiste bestemmingen in de CPU, teneinde de gespecificeerde bewerking uit te voeren.

Behalve het specificeren van een rekenbewerking, zal een instructie normaliter ook andere informatie geven, zoals het adres of de adressen van de geheugenplaatsen, waar een operand of de operanden zijn opge-

slagen, die worden benut voor de rekenbewerkingen. Het aantal bits, dat vereist is voor de operandadressen (adresveld) bezetten gewoonlijk het grootste deel van het aantal beschikbare bitposities in de instructie, zodat slechts een gering aantal bits overblijft voor het op-codeveld. Wanneer de ontwerper van een CPU vindt, dat de toegekende bits voor het op-codeveld onvoldoende is voor een bepaalde instelling van de rekenbewerkingen, had hij tot nu toe de keuze, of het accepteren van een kleiner aantal rekenbewerkingen, of het verlengen van de instructie.

Lange instructies zijn nadelig in kleine rekeninrichtingen, die slechts een beperkte geheugencapaciteit hebben voor het opslaan van instructies. Bovendien hebben kleine rekeninrichtingen een beperkte woordlengte (in bepaalde systemen waar de CPU de vorm heeft van een microprocessor minder dan 4 bits), terwijl een lange instructie het nadeel heeft van een groot aantal geheugenreferenties om de instructie eruit te halen en derhalve een langzamere uitvoering van de bewerking. Van het standpunt van veelzijdigheid, programmeergemak, en de werkefficiëntie, is het echter gewenst een groot aantal rekenbewerkingen te hebbem. Het probleem is derhalve een CPU te ontwikkelen voor kleine rekeninrichtingen, die geschikt is voor het uitvoeren van een groot aantal bewerkingen, met een minimale instructielengte.

Voor het in stand houden van de lengte van de instructies heeft men zich in de bekende techniek bezig gehouden met het verminderen van de lengte van het adresveld door het toepassen van een of andere vorm van verkorte adressering van geheugenplaatsen, bij voorbeeld in de vorm van indirecte adressering, relatieve adressering, of het verlengen van de adrescode. Evenwel blijft het probleem aanwezig hoe een extra vermindering van de instructielengte is te verkrijgen, wanneer een N-bit op-codeveld in de instructie meer dan 2^N-rekenbewerkingen omvat.

Het probleem wordt volgens de uitvinding opgelost door de centrale verwerkingseenheid te voorzien van een op-code-verlengregister voor het daarin opslaan van een geselecteerd op-codeverlengwoord en waarbij de besturingsinrichting zowel reageert op de gecombineerde inhoud van het instructieregister als op het op-codeverlengregister voor het opwekken van commandosignalen, die overeenkomen met de gecombineerde inhoud, waarbij de inhoud van het op-codeverlengregister alleen verandert indien de centrale verwerkingseenheid een instructie ontvangt voor het overdragen van een nieuw geselecteerd op-codeverlengwoord naar het op-code-

verlengregister.

De uitvinding voorziet in een CPU van een informatieverwerkend systeem, teneinde een programma van opgeslagen informatie door middel van een opeenvolging van instructiecycli uit te voeren, waarbij de CPU gedurende de instructie-ophaalfase van de instructiecyclus een instructie ontvangt en in responsie op de instructie, het gedurende de informatiebewerkingsfase van een instructiecyclus uitvoeren van een bewerking gespecificeerd door de instructie van opgeslagen informatie gespecificeerd door de instructie, waarbij elke opgeslagen instructie voorzien is van een op-codegedeelte en van een adres-gedeelte, waarbij de CPU voorzien is van een instructieregister voor het ontvangen van het op-codegedeelte van de instructie gedurende de instructie ophaalfase en voor het opslaan van de op-code tijdens de gehele instructiecyclus; en van een besturingsinrichting voor het leveren van de commandosignalen die corresponderen met de rekenbewerkingen gedurende de informatie-uitvoeringsfase, met het kenmerk, dat de opgeslagen informatie voorzien is van operanden en op-codeverlengwoorden en dat voorzien is in een op-codeverlengregister voor het daarin opslaan van een geselecteerd op-codeverlengwoord en waarbij de besturingsinrichting responsief is op de gecombineerde inhoud van het instructieregister en van het op-codeverlengregister voor het leveren van commandosignalen, die corresponderen met de rekenbewerkingen, die vereist zijn voor de respectieve gecombineerde inhoud, waarbij de inhoud van het op-codeverlengregister verandert wanneer de CPU een instructie ontvangt voor het overdragen van een nieuw geselecteerd op-codeverlengwoord naar het op-codeverlengregister.

De inhoud van het op-codeverlengregister, dat niet veranderd behoeft te worden bij elke instructiecyclus, wordt derhalve gecombineerd met de op-code, opgeslagen in het instructieregister, dat verandert met elke nieuwe instructie voor het vormen van een effectieve langere op-code.

Met voordeel wordt het op-codeverlengregister toegepast bij het verdelen van het aantal CPU-bewerkingen in algemene rekenbewerkingen en facultatieve CPU-eigenschappen, die de algemene bewerkingen modificeren. Het op-codeveld bepaalt derhalve in elke instructie de algemene bewerkingen, terwijl de facultatieve eigenschappen wel of niet worden geselecteerd door het op-codeverlengwoord, opgeslagen in het op-codeverlengregister.

Het is derhalve een doel van de uitvinding een CPU te verschaffen, die een rekenbewerkingsinstelling mogelijk maakt, die groter is dan die, die bepaald kan worden door het op-codeveld in een instructie.

Een ander doel van de uitvinding is te voorzien in een CPU, die de veelzijdigheid en de prestatie van een kleine rekeninrichting met een kleine woordafmeting en een beperkte instructiegeheugencapaciteit aanzienlijk verbetert.

Een ander doel van de uitvinding is te voorzien in een CPU voor het verminderen van het aantal geheugenreferenties, die vereist zijn voor het ophalen van een instructie in een informatieverwerkend systeem met een kleine woordafmeting.

Een ander doel van de uitvinding is te voorzien in een CPU, die het mogelijk maakt programmeerbare facultatieve eigenschappen toe te voegen zonder dat aanvullende op-codes nodig zijn.

Een ander doel van de uitvinding is te voorzien in een CPU, die toegepast kan worden in een goedkope microverwerkingsinrichting met een zeer grote prestatie.

De hierboven aangegeven doelen van de uitvinding zijn gerealiseerd in de verschillende uitvoeringsvormen, die naderhand zullen worden beschreven.

Het is evenwel duidelijk, dat ook andere nuttige uitvoeringsvormen kunnen worden onderworpen die minder kunnen uitvoeren dan alle hierboven genoemde doelen, maar die toch gelegen zijn binnen de principes van de uitvinding.

De uitvinding zal onderstaand aan de hand van een aantal voorbeelden en onder verwijzing naar de tekening nader worden uiteengezet. Uitdrukkelijk wordt naar voren gebracht dat de tekening uitsluitend bestemd is ten behoeve van een nadere uiteenzetting van de uitvinding en niet bestemd is om de uitvinding te beperken.

Fig. 1 toont een blokdiagram van een bekende CPU;

fig. 2 is een stromingsdiagram voor het aangeven van de wijze van verwerking gedurende een instructiecyclus;

fig. 3 toont een diagram van de opeenvolgende stappen van de CPU van fig. 1 tijdens het uitvoeren van een ALU-bewerking;

fig. 4 toont een blokdiagram van een CPU, waarin de operandbreedte wordt geregeld door de programmabesturing van de ALU;

fig. 5 toont een logicadiagram van de overdracht-in-administratie-

keten, zoals toegepast in de CPU van fig. 4;

fig. 6 toont een logicadiagram van de overdracht-uit-multiplexketen, zoals toegepast in de CPU van fig. 4;

fig. 7 is een logicadiagram van de overdracht-uit-registerketen toegepast in de CPU van fig. 4;

fig. 8 is een logicadiagram van de draai/verschuif naar rechts-administratieketen, zoals toegepast in de CPU van fig. 4;

fig. 9 is een logicadiagram van de ALU-functieblokkeerketen, zoals toegepast in de CPU van fig 4;

fig. 10 is een blokdiagram van een CPU, waarin de facultatieve kenmerken van variabele operandbreedte, automatisch laden en auto-incrementering van de geheugenaanwijsregisters worden geregeld door een programmabesturing;

fig. 11 toont een op-codeverlengregister, zoals toegepast in de CPU van fig. 10;

fig. 12 is een diagram van de door de CPU van fig. 10 achtereenvolgens uitgevoerde stappen tijdens het uitvoeren van een ALU-bewerking en de bewerking van de variabele operandbreedte en de auto-incrementerende eigenschappen;

fig. 13 is een diagram van de door de CPU van fig. 10 gevolgde stappen in adresformatie en de bewerking van de eigenschap van het automatisch laden;

fig. 14 is een blokdiagram van een deel van een CPU, waarin de overdracht van het adresregister voor het daarin opslaan van hetzij geheugenadressen of randinrichtingadressen door de programmabesturing wordt geregeld overeenkomstig de principes van de uitvinding;

fig. 15 is een blokdiagram van het instructieregister, op-codeverlengregister en de besturingsinrichting van de CPU van fig. 14 en een logicadiagram van de keten in de besturingsinrichting, die het adresregisteropdrachtsveld decodeert;

fig. 16 is een blokdiagram van een koppelingsopstelling van de CPU van fig. 14 met een RAM en drie randinrichtingen.

Thans wordt verwezen naar fig. 1, die een blokdiagram geeft van een simpele CPU 100, die in de technisch op zich bekend is en die toegepast wordt in verwerkingssystemen, zoals minicomputers en microverwerkingsinrichtingen. Alleen die gedeelten van de CPU, die essentieel zijn voor de uitleg, zijn in fig. 1 aangegeven. In de configuratie van

fig. 1 is voorzien in één enkel instructieregister (IR) 101, waarin de in uitvoering zijnde instructie is opgeslagen. Een programma van instructie dat moet worden uitgevoerd door de CPU is opgeslagen in een dood geheugen (ROM) 102, in de vorm van binaire woorden, die hierna instructiewoorden genoemd worden. In dit voorbeeld bevat elk instructiewoord acht bits. De CPU haalt teneinde een complete instructie te vormen een integraal aantal instructiewoorden uit het ROM. Elke instructie is samengesteld uit een op-code, die een bewerking aangeeft, die door de CPU kan worden uitgevoerd, en uit een adres-code, die de geheugenplaats aangeeft, waarin een operand of de operanden voor het uitvoeren van een specifieke bewerking wordt of worden opgeslagen. In de CPU van fig. 1 zijn de operanden opgeslagen in een vrij toegankelijk geheugen (RAM) 103, welke operanden in de vorm zijn van binaire woorden, die hierna operandwoorden genoemd worden. In dit voorbeeld, omvat elk operandwoord acht bits. Een operand wordt gevormd uit één operandwoord, dat uit het RAM tevoorschijn is gehaald. De IR houdt acht bits van een instructie vast, waarvan zes gereserveerd zijn voor de op-code en de overblijvende twee gereserveerd zijn voor het deel van de adrescode, dat de adresmodus specificeert. Vier adresmodi zijn beschikbaar in de CPU van fig. 1; nl. de direct mode, twee indirecte modi en de spoedinformatiemodus. De lengte van een instructie varieert afhankelijk van de toegepaste adresmodus. In onderstaande tabel I is een uitvoeringsvoorbeeld gegeven hoe een zes-bit op-code wordt toegepast voor het bepalen van achtenveertig rekenbewerkingen.

T A B E L I

CPU instructie-instelling

	Instructie	Op-code
1	Tel op	000000
2	Tel op bij overdracht	000001
3	Trek af	000010
4	Trek af bij overdracht	000011
5	En	000100
6	Of	000101
7	Exclusief af	000110
8	Verplaats	000111
9	Vergelijk en tak af indien gelijk	001000
10	Vergelijk en tak af niet gelijk	001001
11	Vergelijk en tak af kleiner dan	001010

12	Vergelijk en tak af groter dan gelijk	001011
13	Test en tak af indien nul	001100
14	Test en tak af indien niet nul	001101
15	DECREMENT, tak af indien niet nul	001110
16	Stel in	001111
17	Vrijgeven	010000
18	INCREMENT	010001
19	DECREMENT	010010
20	COMPLEMENT	010011
21	Draai naar links	010100
22	Draai naar links bij overdracht	010101
23	Draai naar rechts	010110
24	Draai naar rechts met overdracht	010111
25	Verschuif naar links	011000
26	Verschuif naar links bij overdracht	011001
27	Verschuif naar rechts	011010
28	Verschuif naar rechts bij overdracht	011011
29	Stel Bit in	011100
30	Geef bit vrij	011101
31	Tak af op bit instelling	011110
32	Tak af op vrijgeefbit	011111
33	Verwissel	100000
34	Test met masker en tak af indien nul	100001
35	Test met masker en tak af indien niet nul	100010
36	Tak af	100011
37	Spring	100100
38	Roep op	100101
39	Keer terug	100110
40	Keer terug bij onderbreken	100111
41	Geef vlag register vrij	101000
42	Stel vlag register in	101001
43	Druk	101010
44	Duw	101011
45	Verwissel geheugen aanwijzer	101100
46	Laad stapel aanwijzer	101101
47	Sla stapel aanwijzer op	101110
48	Geen bewerking	101111

De programmateller (PC) 104, houdt normaliter het adres vast van het volgende instructiewoord of constante, dat uit het ROM wordt gehaald. De PC is een twaalf-bit-register, dat door een stap-voor-stap telreeks gaat en daarbij wijst op opeenvolgende in het ROM opgeslagen instructiewoorden; echter niet in responsie op een programma-overdrachtsinstructie (d.w.z. instructies 9-15, 31, 32, 34-40, 43, en 44 van tabel 1), wanneer een nieuw adres in het PC wordt geladen. Een acht-bit informatiebus 105, wordt toegepast voor het overdragen van instructies, adressen, en operanden tussen de verschillende geheugenplaatsen in de CPU. De adressen van de geheugen in het RAM 103, worden opgeslagen in een twaalf-bit geheugen-adresvergrendelinrichting (MAL) 106, die een adres ontvangt van één van een groep van registers in een geheugenaanwijsinrichting (MP) 107. De inhoud van het IR wordt gedetecteerd door een besturingsinrichting 108, die commandosignalen levert voor de rekenbewerkingen en adresm anipulaties, gespecificeerd door de gedetecteerde inhoud. De commandosignalen worden overgedragen naar verschillende bestemmingen in de CPU, via stuurlijnen 109.

De rekenkundige en logische bewerkingen worden uitgevoerd in de rekenlogica-eenheid (ALU) 110, die voorzien is van twee ingangen A en B, waaraan één of twee operanden kunnen worden toegevoerd. Een accumulator (ACC) 113, slaat ten behoeve van de ALU een operand op en slaat eveneens het resultaat op van de ALU-bewerking. De ALU is voorzien van een overdracht-in-ingang voor het ontvangen van een overdrachtssignaal en van een overdracht-uit-uitgang, waaraan het door de ALU opgewekte overdrachtssignaal wordt toegevoerd. Het ingangsoverdrachtssignaal wordt geleverd door de overdracht-in-administratieketen 111, die een +1 signaal selecteert in het geval van een INCREMENT bewerking, en een overdrachtssignaal selecteert in het geval van een DRAAI-NAAR-LINKS BIJ OVERDRACHT-bewerking, of de meest significante bit AC 7, van de accumulator selecteert in het geval van een DRAAI NAAR LINKS-bewerking. Het uitgangsoverdrachtssignaal wordt ontvangen door een overdrachtsregister (CR) 112, dat een ALU-overloop of -onderloopfunctie aangeeft. Het CR is eveneens oplaadbaar, terugzetbaar en neemt deel in de SCHUIF- en DRAAI-bewerkingen.

De ACC die een hoofdgedeelte ACM, en een slaafgedeelte ACS bezit, wordt in dit voorbeeld eveneens toegepadt als een vergrendeling voor de informatie die moet worden uitgelezen uit of moet worden ingelezen in het RAM. Een speciale schuif/draainaarrechts-keten (SRR) 114, levert

de SCHUIF-EN-DRAAI-NAAR-RECHTS-bewerkingen met betrekking tot de ACC. De SCHUIF-EN-DRAAI-NAAR-LINKS-bewerking wordt verkregen door de operand bij zichzelf op te tellen en vereist derhalve geen speciale keten. Een draai/verschuif-naar-rechts-administratieketen (RSR) 115, selecteert hetzij het overdrachts-uitgangssignaal, hetzij de minst significante bit, ACO, van de ACC die moet worden opgeladen in de meest significante bit-positie van de ACC, ten behoeve van resp. de DRAAI-NAAR-RECHTS-bij-overdracht en de DRAAI-NAAR-RECHTS-bewerkingen.

Teneinde de instructiecyclus te vormen vindt de uitvoering van elke instructie plaats met behulp van de CPU in een opeenvolging van basisstappen. De opeenvolgende stappen worden geregeld met behulp van tempeersignalen, die worden opgewekt door een klokketen, die in fig. 1 niet is aangegeven.

Thans wordt verwezen naar fig. 2, waarin een stromingsdiagram is aangegeven van de stappen in de instructiecyclus. De instructiecyclus is verdeeld in twee fasen: een instructie-ophaalfase en een informatie-uitvoerfase. De referentienummers 201 t/m 206 geven de basisstadia aan van elke fase. In de instructie-ophaalfase wordt een instructiewoord, waarvan het adres in de PC is, overgedragen van het ROM naar het IR. Het instructiewoord, zoals eerder genoemd, bezit een zes-bit op-codeveld en een twee-bit-adresmodusveld. De eerste stap in de informatie-uitvoeringsfase is de formatie van het adres of adressen van operanden, die vereist zijn voor de bewerking. De operanden worden opgeslagen in het RAM of in inwendige registers in de CPU. Eerst wordt de op-code in het IR gecodeerd met behulp van de besturingsinrichting, teneinde vast te stellen of de bewerking eenwaardig of tweewaardig is. Een eenwaardige bewerking is een bewerking die slechts één operand vereist, zodat uitsluitend één operandadres wordt gevormd. Het aldus gevormde enkele operandadres voor een enkelvoudige bewerking wordt een bestemmingsadres genoemd, omdat de resultaten van de bewerking automatisch worden opgeslagen in de geheugenplaats, die eerder bezet werd door de enkele operand. Een tweewaardige bewerking is een bewerking die twee operanden vereist, zodat twee operandadressen worden geformeerd. De adressen van de eerste operand wordt het bronadres genoemd, terwijl dat van de twee operand het bestemmingsadres wordt genoemd, omdat de resultaten van de tweevoudige bewerking automatisch zal worden opgeslagen in de geheugenplaats, die eerder bezet werd door de tweede operand. In de CPU van fig. 1, is het

bestemmingsadres in het algemeen altijd ACC.

De adresmodusbits in het IR worden eveneens gedetecteerd door de besturingsinrichting. Indien een rechtstreekse adressering is aangegeven, wordt een adres overgedragen van een geheugenplaats in het ROM naar de MAL. Indien evenwel één van de beide indirecte adresmodi wordt aangegeven, wordt de inhoud van een gespecificeerd geheugenaanwijsregister in de MP overgedragen naar de MAL. In het geval van een spoed-informatie-adresmodus wordt de operand of worden de operanden meegevoerd door het adresgedeelte van de instructie en worden derhalve geen operandadressen gevormd.

Nadat het operandadres is gevormd en overgedragen naar de MAL, kan de operand opgehaald worden uit de geheugenplaats in het RAM, dat door de MAL is aangewezen. De laatste stap in de informatie-uitvoeringsfase is het uitvoeren van de bewerking aangegeven door de op-code in het IR, dat in dit geval een ALU-functie is. Na voltooiing van de ALU-functie eindigt de instructiecyclus.

Thans wordt verwezen naar fig. 3, die een blokdiagram 300 weergeeft van de gesimplificeerde CPU van fig. 1. De referentienummers 301 t/m 310 geven de gedetailleerde stappen aan in een instructiecyclus voor het uitvoeren van de ALU en de informatie-overdrachtsbewerkingen. Niet-informatiebewerkingen, zoals AFTAKKEN, OPROEPEN, TERUGKEREN, etc. zijn geklassificeerd als gemengde instructies en zijn niet weergegeven. De tweevoudige bewerking ADD tussen een eerste operand, opgeslagen in een RAM geheugenplaats, aangewezen door een register in het MP en een tweede operand eveneens opgeslagen in de ACC kunnen door de navolgende reeks worden weergegeven:

S1 $\longrightarrow$ S2 $\longrightarrow$ S4 $\longrightarrow$ S6 $\longrightarrow$ S9 $\longrightarrow$ S10.

Onder verwijzing naar de toestanden, weergegeven door de blokken aangeduid met de referentiecijfers 301 t/m 310 in fig. 3, wordt in de toestand S1 de inhoud van het ROM geheugenplaats, geadresseerd door de inhoud van PC, overgedragen naar IR. In de toestand S2 wordt de inhoud van IR gecodeerd door de besturingsinrichting. In dit voorbeeld is verondersteld, dat de adresmodus indirect is en dat in de toestand S4, 304, de inhoud van een gespecificeerd register in het MP overgedragen wordt naar de MAL. In de toestand S6, 306, wordt de operand, opgeslagen in de RAM-geheugenplaats en gespecificeerd door het adres in de MAL, overgedragen naar de ALU, en wordt een bepaald ALU-bewerking, gespecificeerd

door de op-code (d.w.z. ADD), uitgevoerd met de hierboven aangegeven operand en de inhoud van ACC, met als resultaat de overdracht naar het hoofdgedeelte (ACM) van de ACC. In de toestand S9, 309, kan de inhoud van ACM, indien gewenst door de instructie, hetzij naar de slaag (ACS) van de ACC, hetzij naar een RAM-geheugenplaats, aangewezen door de MAL, worden overgedragen. Voor de ADD-bewerking blijft het resultaat (de som) in de ACC. In de toestand S10, 310, wordt het PC geïncrementeerd, alvorens een nieuwe instructiecyclus te beginnen.

Een voorbeeld van een enkelvoudige bewerking, STEL ACCUMULATOR IN, kan weergegeven worden door de navolgende toestandreeks;

S1 → S2 → S7 → S9 → S10.

De stappen, weergegeven door de toestanden S1 en S2 zijn dezelfde als die hierboven beschreven voor de tweevoudige bewerking. In de toestand S7, 307, wekt de ALU een acht-bitwoord op, bestaande uit enen, welk woord overgedragen wordt naar de ACM. De stappen weergegeven door de toestanden S9 en S10 zijn dezelfde als hierboven beschreven voor de tweevoudige bewerking.

Een voorbeeld van een informatie-overdrachtsbewerking, VERPLAATS, van een in het ROM opgeslagen constante, naar een register in het MP, wordt weergegeven door de navolgende toestandreeks;

S1 → S2 → S3 → S5 → S10.

De stappen, weergegeven door detoestanden S1 en S2 zijn reeds uiteengezet met betrekking tot de tweevoudige bewerking. In de toestand S3, 303, wordt de PC geïncrementeerd en verwezen naar de ROM-geheugenplaats, waar de constante wordt opgeslagen. In de toestand S5, 305, wordt de constante in de ROM-geheugenplaats, aangewezen door het PC, overgedragen naar het MP. De stap weergegeven door de toestand S10 is reeds uiteengezet met betrekking tot de tweevoudige bewerking.

De CPU-configuratie van fig. 1 kan zodanig worden gemodificeerd, dat deze behalve de in de tabel 1 aangegeven bewerkingen, ook nog andere rekenbewerkingen kan uitvoeren. Bij voorbeeld kan, zoals naderhand in detail beschreven zal worden, de ALU-sectie gewijzigd worden en geschikt gemaakt worden voor zowel vier-bit-operanden als ook voor de normale acht-bit-operanden waarbij de keuze van de operandbreedte wordt geregeld door het besturingsprogramma. Een dergelijke modificatie zou de rekenkundige, de logica en de informatieverplaatsingsbewerkingen, weergegeven door de instructies 1 t/m 35 van tabel 1, beïnvloeden. De modificatie

zou effectief 35 nieuwe bewerkingen toevoegen aan de vier-bit-operandbreedte, aan de bewerkingen die reeds zijn aangegeven in tabel I, waardoor het totaal-aantal bewerkingen op 83 komt. Voorzover het zes-bit op-codeveld in de CPU van fig. 1 slechts 64 verschillende codes toelaat, moet het opcodeveld uitgebreid worden, teneinde deze extra bewerkingen te kunnen onderbrengen. Als gevolg van het feit, dat de woordafmeting van de CPU acht-bit is, teneinde koppelingsmoeilijkheden te voorkomen, zou het noodzakelijk zijn het IR door een veelvoud van acht-bits uit te breiden. Indien elke instructie informatie moet meevoeren met betrekking tot de operandbreedte, zou de instructielengte met acht bits toenemen; dit zou een aanzienlijke toename van het ROM geheugen betekenen, teneinde het programma op te slaan en bovendien is er nog een extra ROM-referentie nodig, die moet worden toegevoegd aan elke instructiecyclus. Deze extra toename in het geheugen zal de kosten aanzienlijk doen toenemen, terwijl deze extra ROM-referenties de snelheid van de bewerking aanzienlijk doet afnemen.

Volgens de principes van de uitvinding is eveneens voorzien in een alternatief middel voor programmaselectie van de operandbreedte. Het zal duidelijk zijn, dat in de meeste programma's, wanneer een bepaalde operandbreedte is geselecteerd, vele bewerkingen kunnen worden uitgevoerd, alvorens het noodzakelijk is van operandbreedte te veranderen. Het gedeelte van het op-codeveld, dat de operandbreedte aangeeft, blijft constant gedurende vele instructiecycli, terwijl het gedeelte van het op-codeveld, dat de bewerkingen aangeeft, zoals aangegeven in tabel I (algemene bewerkingen) gewoon verandert met elke nieuwe instructiecyclus. Het is derhalve niet nodig aan elke instructie het niet-frequente veranderde gedeelte van de op-code toe te voeren; in plaats daarvan kan deze worden opgeslagen in een speciaal register. Voor dit doeleinde zal het specifieke register genoemd worden het op-codeverlengregister (OER). Het toepassen van het OER voor het opslaan van het gedeelte van de op-code, die de operandbreedte specificeert, betekent, dat het op-codeveld in elke instructie alleen de algemene bewerkingen behoeft aan te geven, waarvoor zes bits voldoende zijn. Door het toepassen van het OER, is het niet meer nodig de instructielengte te verlengen.

Het OER kan eveneens andere facultatieve eigenschappen, behalve de variabele operandbreedte, specificeren. Enkele van deze andere eigenschappen, zoals automatisch opladen en automatische incrementering van

de geheugenaanwijsregisters, en de overdracht van adresregisters, zullen eveneens in detail worden behandeld.

Wanneer verschillende facultatieve eigenschappen door het OER moet worden gespecificeerd, moeten er voldoende bitposities in het OER aanwezig zijn teneinde alle verschillende combinaties van de facultatieve eigenschappen te specificeren. Een groep bits, die door het OER worden opgeslagen, wordt hierna genoemd een op-codeverlengwoord. De inhoud van het OER wordt alleen veranderd indien speciale instructies dit noodzakelijk maken (toegevoegd aan de lijst van tabel I) voor het aan het OER overdragen van de nieuw op-codeverlengwoord vanuit het geheugen en voor het veranderen van een bepaalde bit in het OER. Elk op-codeverlengwoord komt overeen met een bepaalde combinatie van speciale eigenschappen, die in het programma worden toegepast. Door het toepassen van het OER voor het programmaselectie van facultatieve eigenschappen, wordt een geheugenruimte in het ROM in stand gehouden en zijn geen extra ROM-referenties in elke instructiecyclus nodig.

Variabele operandbreedte

Thans wordt verwezen naar fig. 4, waarin een blokdiagram is aangegeven van een CPU 400, die in principe gelijk is aan die van fig. 1, maar die thans gemodificeerd is, teneinde een verandering van de operandbreedte van acht bits naar vier bits mogelijk te maken. Deze modificatie omvat het delen van de ALU, de accumulator en de schuif-en-draai-naar-rechts-keten in twee onafhankelijke vier-bitsecties, die hierna genoemd zullen worden "hogere: en "lagere" secties. In fig. 4 zijn de hogere secties van de ALU (ALUH), accumulator (ACCH), en de schuif-en-draai-naar-rechts-keten (SRRH), resp. aangegeven met de referentienummers 410, 416 en 418. De lagere secties van de ALU (ALUL), de accumulator (ACCL) en de schuif-naar-rechts-en-draai-keten (SRRL) zijn resp. aangegeven met de referentiecijfers 411, 417 en 419. Voor elk van deze gevallen zijn de hogere en de lagere secties zodanig aangesloten, dat wanneer de operandbreedte acht bits is, de hogere secties werkt op de meest significante vier-bits van het operandwoord, terwijl de lagere sectie werkt op de minst significante vier-bits van het acht-bitoperandwoord. Wanneer de operandbreedte vier-bits is, blijven de hogere secties van elk gedeelte onwerkzaam, alsof de lopende instructie GEEN BEWERKING was, terwijl de lagere secties werken op de minst significante vier-bits van een operandwoord.

De overdracht-in-administratieketen (CIA) 412, voorziet normaliter in een "nul" aan de overdracht-in-ingang van de ALUL, 411. Indien de huidige instructie INCREMENT OF DECREMENT is, is het CIA-uitgangssignaal een "een". Het CIA-uitgangssignaal is in de toestand van het overdrachtsregister 413, indien de huidige instructie TEL OP bij OVERDRACHT, AFTREKKEN, of DRAAI-NAAR-LINKS-bij-OVERDRACHT, is. Voor de DRAAI-NAAR-LINKS-instructie geeft de CIA-uitgang de meest significante bit (ACC) van ACCH in het geval van een acht-bitoperand, en geeft de meest significante bit (AC3) van ACCL in het geval van een vier-bitoperand. Een voorbeeld van een logische uitvoering van de CIA is in fig. 5 weergegeven.

De overdracht-uit-multiplexer (COMX) 414, voert het overdracht-bit-ingangssignaal toe aan het overdrachtsregister 413. Het COMX-uitgangssignaal is hetzij het overdracht-uit-uitgangssignaal van de ALUH in het geval van een acht-bitoperand, hetzij het overdracht-uit-uitgangssignaal van de ALUL in het geval van een vier-bitoperand. Een voorbeeld van een logica-uitvoering van de overdracht-uit-multiplexer is in fig. 6 weergegeven.

Het overdrachtsregister (CR) 413, is een flipflop, die vanuit verschillende bronnen ingesteld kan worden, afhankelijk van de huidige instructie. De werkzame ingangen van het CR voor de verschillende instructies zijn onderstaand aangegeven:

Instructie	Werkzame CR ingang
(ALU bewerking met overdracht)	CB van COMX
Laad CR	Data Bus
Geef CR vrij	"0"
Verschuif naar rechts met overdracht	"0"
Draai naar rechts met overdracht	AC0

Een voorbeeld van een logica-uitvoering van het overdrachtsregister is in fig. 7 weergegeven.

De keten (RSR) 420 levert de meest significante bit voor de ACCH en/of ACCL in de SCHUIF-EN-DRAAI-NAAR-RECHTS-bewerkingen. In het geval van een acht-bit-operandbreedte, levert de RSR de toestand van de minst significante bit (AC0) van de ACCL naar de meest significante bitpositie (ACC) van de ACCH via de OT8-uitgang. Bovendien levert de RSR de toestand van de minst significante bit (AC4) van de ACCH naar de meest significante bit-positie (AC3) van de ACCL via de OT4-uitgang. In het

geval van een 4-bitoperandbreedte, levert de RSR-toestand van de minst significante bit (AC0) van de ACCL naar de meest significante bitpositie (AC3) van de ACCL via de OT4-uitgang. De ingangssignalen en de operatieve uitgangssignalen van de RSR voor de verschillende instructies zijn onderstaand aangegeven:

Instructie	Ingangen	Werkzame uitgangen
Verschuif naar rechts (8-bit)	0,AC4	OT8,OT4
Verschuif naar rechts (4-bit)	0	OT4
Draai naar rechts (8-bit)	AC0,AC4	OT8,OT4
Draai naar rechts (4-bit)	AC0	OT4
Verschuif naar rechts/ draai bij overdracht	Overdracht,AC4	OT8,OT4
Verschuif naar rechts/ draai bij overdracht	Overdracht	OT4

Een logica-uitvoering van de draai/verschuif-naar-rechts-administratieve keten is aangegeven in fig. 8.

De ALU-functie-blokkeerketen (ALUFI) 415 wordt benut voor het blokkeren van de ALUH, indien de operandbreedte vier-bits is. Indien de operandbreedte acht-bits is, regelt de ALU-functiestuurlijnen 422 van de besturingsinrichting 408 de keuze van de ALU-functies voor zowel de ALUH en ALUL. Wanneer de operandbreedte vier-bits is, levert een stuursignaal aan de ALUH, welk signaal correspondeert met de NIET-BEWERKING-instructie, en laat de ALUH de operanden passeren zonder te veranderen. Een voorbeeld van een logica-uitvoering van de ALU-functieblokkeerketen is weergegeven in fig. 9.

Wanneer een vier-bitoperandbreedte wordt geselecteerd voor de CPU van fig. 4, werken alleen de minst significante vier-bits van het operandwoord, terwijl de meest significante vier-bits ongebruikt blijven. Teneinde gebruik te maken van de meest significante vier-bits van het operandwoord en daarbij volledig gebruik te maken van de RAM-opslagruimte die ingericht wordt voor het opslaan van acht-bitwoorden, is het noodzakelijk te voorzien in middelen om de meest significante en de minst significante vier-bits van het operandwoord onderling te wijzigen. Een dergelijke onderlinge wijziging kan verkregen worden op verschillende manieren. Eén van de wijzen om een effectieve onderlinge verandering

van de meest en de minst significante vier-bits in een operandwoord te verkrijgen is, onder besturing van een bitpositie in het OER, de "hoge" en "lage" secties van de ALU onderling te veranderen. Wanneer bij voorbeeld een aangewezen bitpositie van het OER voor het besturen van de ALU-verwisseling een "nul" bevat gedurende de vier-bitbewerking, zal de ALUFI de ALUH blokkeren, terwijl de ALUL werkzaam wordt op de minst significante vier-bits van het operandwoord. Indien de aangewezen bitpositie een "een" bevat gedurende de vier-bitbewerking, zal de ALUFI de ALUL blokkeren, terwijl de ALUH uitsluitend op de meest significante vier-bit van het operandwoord werkt.

De meest en minst significante vier-bits van een operandwoord kan rechtstreeks worden verwisseld door het introduceren van een DRAAI-4-instructie, aan de lijst van tabel I. De CPU van fig. 4 zal na ontvangst van een dergelijke instructie een operandwoord, gespecificeerd door de instructie vanuit het RAM, overdragen aan de gedeelde accumulator, waarbij oorspronkelijk de meest significante vier-bit van het operandwoord in de ACCH en de minst significante vier-bit in de ACCL aanwezig blijft. Het operandwoord wordt dan in de gedeelde accumulator "rondgedraaid", totdat de voorgaande meest significante vier-bits aanwezig is in de ACCL en de voorgaande minst significante vier-bits aanwezig in de ACCH. Het "rondgedraaide" operandwoord in de gedeelde accumulator kan hetzij meteen worden toegepast in een ALU-bewerking, hetzij terug worden gevoerd naar het RAM.

Een programmeerbare selectie van de operandbreedte wordt verkregen met behulp van het op-codeverlengregister (OER) 421, dat in dit voorbeeld uitsluitend een-bitpositie behoeft te bevatten. Wanneer de binaire bittoestand van die bit een "een" is, is de operandbreedte voor de ALU en informatieverplaatsingsbewegingen vier-bits, voor het overige is de operandbreedte acht-bits. Een LAAD-OER-instructie wordt toegevoegd aan de lijst in tabel I, waarbij het ROM twee constanten bevat, en waarbij als op-code verlengwoorden dienst doen een "een" en een "nul". Alhoewel slechts één bit nodig is voor het specificeren van de operandbreedte, kan het OER extra bitposities omvatten voor het selecteren van andere facultatieve eigenschappen, zoals hierboven reeds is besproken.

Een alternatieve CPU-configuratie voor het uitvoeren van de variabele operandbreedte, alsook voor de andere facultatieve eigenschappen is in fig. 10 aangegeven. Alhoewel het ontwerp van de CPU uit fig. 10

principieel verschillend is van dat van fig. 4, wordt hetzelfde op-code verlengregister toegepast voor het selecteren van de facultatieve eigenschappen. Thans wordt verwezen naar fig. 10, waarin het instructieregister 1001, het ROM 1002, het RAM 1003, de informatiebus 1004 en de besturingsinrichting 1005 allen op dezelfde wijze functioneren als hun corresponderende delen in de CPU van fig. 4, met uitzondering van de woordafmeting van het ROM en RAM en de breedte van de informatiebus en het instructieregister die allen vier-bits zijn in plaats van acht-bits. De adresvergrendeling (AL) 1006 is een twaalf-bit-hoofd-slaagvergrendeling voor het opslaan van het huidige adres van een instructie of van een operand. De twaalf-bit-adres-rekenkundige eenheid (AAU) 1007, incrementeert of decrementeert het huidige adres in AL. Het interne registergeheugen (IRM) 1008, bestaat uit een groep registers, voorzien van een programmeerteller (PC) 1009, van twee geheugenaanwijsregisters B0 1010 en B1 1011 en van een stapelaanwijzer 1012. Het tijdelijke registergeheugen (TRM) 1013 omvat twee tijdelijke registers T0 1014 en T1 1015 en worden toegepast voor de gemiddelde adresberekeningen tijdens de adresformatietoestand. De vier adresmodi, beschikbaar in de CPU van fig. 4 zijn eveneens beschikbaar in de CPU van fig. 10.

De informatie-adresbus-multiplexer (DAMUX) 1015, bestuurt de overdracht van informatie in vier-bitwoorden vanuit het ROM en RAM naar de twaalf-bitregisters in het RIM en TRM, via de vier-bitinformatiebus. De DAMUX regelt de vier-bits van de informatiebus naar één van de drie vier-bitsecties (aangegeven met hogere, middelste en onderste secties) van de adresrekenbus 1026. De drie secties van de adresrekenbus zijn resp. verbonden met de hogere, de middelste en de onderste vier-bitsecties van de twaalf-bitregisters in het IRM en TRM. Een twaalf-bit-adresbus 1027 draagt de adressen van de AL over naar de AAU, RAM en ROM.

De vier-bitreken- en logica-eenheid (ALU) wordt toegepast voor het uitvoeren van alle rekenkundige en logicafuncties. De operanden voor de ALU worden opgeslagen in de vier-bit tijdelijke informatieregisters TA 1018, en TB 1019.

In tegenstelling tot de CPU van fig. 4, heeft de CPU van fig. 10 geen adresseerbare accumulator. In plaats daarvan worden door de geheugenaanwijsregisters 1010 en 1011 de RAM-geheugenplaatsen aangewezen die als accumulators worden toegepast. Derhalve heeft de CPU net zo veel accumulators als de RAM geheugenruimte heeft.

De basis-operandbreedte van de ALU in de CPU van fig. 10 heeft vier-bits; evenwel is zijn configuratie zodanig, dat de ALU-bewerkingen kunnen worden uitgevoerd op operanden, waarvan de breedten integrale meervouden zijn van de basisbreedte (bij voorbeeld 8, 12 en 16-bits). Voor de operanden met breedtes, die een meervoud zijn van de basisbreedte, werkt de ALU op vier-bitsegmenten van de operand; te beginnen met het meest significante segment, waarbij de ALU-bewerking wordt herhaald op de andere segmenten, overeenkomstig de belangrijkheid van hun significantie. Een vier-bitsegment wordt gewoonlijk "knabbel" (in het Engels nibble) genoemd. Het aantal herhalingen van een ALU-bewerking, vereist voor een gegeven operandbreedte, is gelijk aan het aantal "knabbels" in de operand.

In het voorbeeld van fig. 10 heeft het OER zes bitposities, die ingedeeld zijn overeenkomstig fig. 11. In fig. 11 specificeert een twee-bitveld, voorzien van bits B0 1101 en B1 1102, de vier facultatieve operandbreedtes van vier-bits, acht-bits, twaalf-bits en zestienbits. De andere vier-bitposities van het OER worden toegepast om andere speciale eigenschappen te specificeren, die naderhand beschreven zullen worden. De codering van de bits B0 en B1 en het aantal herhalingen, nodig voor het voltooien van de ALU-bewerking voor elke toestand zijn onderstaand aangegeven:

b1, b0	Operandbreedte	ALU herhalingen
00	16-bits	4
01	4-bits	1
10	8-bits	2
11	12-bits	3

Zoals in fig. 10 is aangegeven, wordt het aantal herhalingen van een ALU-bewerking, gespecificeerd door de inhoud van het OER, bestuurd door de twee-bitteller 1021, en de comparator 1022, die de toestand van de teller vergelijkt met die van B0-B1. Wanneer de toestand van de teller en die van B0-B1 met elkaar overeenkomen, zullen de herhalingen van de ALU-bewerking stoppen.

De details van een ALU-bewerking in de CPU van fig. 10 kan beter begrepen worden, wanneer men het toestandsdiagram 1200, weergegeven in fig. 12, nader beschouwt. In fig. 12 zijn de instructie-ophaaltoestanden 1201 en de adresinformatietoestanden 1202 reeds besproken in verband met fig. 2. Na de adresformatie wordt het bronadres voor een tweevoudige be-

werking opgeslagen in het tijdelijke register T0. Wanneer de bewerking van de huidige instructie een enkelvoudige of tweevoudige bewerking is, is het bestemmingsadres in het tijdelijke register T1. Opgemerkt moet worden dat in de CPU van fig. 10 een bestemmingsadres altijd vereist is voor een tweevoudige op bewerking, in tegenstelling tot de CPU van fig. 4, waarin de accumulator altijd de veronderstelde bestemming is voor tweevoudige bewerkingen.

In de toestand ALU1, 1203 wordt de inhoud van T0 (bronadres) overgedragen aan het hoofd (ALM) van de adresvergrendeling (1023 in fig. 10). Indien de huidige instructie een enkelvoudige bewerking aangeeft, is de volgende toestand ALU5, 1207; overigens is de volgende toestand ALU2, 1204. Gedurende de overgangsperiode van ALU1 naar ALU2, wordt de inhoud van ALM overgedragen naar de slaaf (ALS) van de adresvergrendeling (1024 in fig. 10) en wordt in de toestand ALU2 de operand, opgeslagen in de RAM-geheugenplaats aangewezen door ALS, overgedragen naar het tijdelijk informatieregister TB. In de toestand ALU3, 1205, wordt het adres in ALS geïncrementeerd en overgedragen naar zowel T0 als ALM, en indien een adres facultatieve eigenschap, bij voorbeeld auto-incrementering die nu zal worden beschreven wordt vrijgegeven, wordt het geïncrementeerde adres eveneens overgedragen aan het aanwijsregister B0. Indien de huidige instructie een onmiddellijke informatie-adressering aangeeft voor de tweede operand van de tweevoudige bewerking (d.w.z. de tweede operand zelf wordt toegevoerd in het adresveld van de huidige instructie) wordt de volgende toestand ALU4, 1206. In de ALU4 wordt de inhoud van ALS overgedragen naar PC, voor het herstellen van het ROM-adres voor de volgende informatieconstante of instructie. Voor de andere adresseermodi wordt de volgende toestand ALU5, 1207, waarbij het bestemmingsadres is opgeslagen in T1, overgedragen naar ALM en wordt de tweebitteller geïncrementeerd, waarbij de teller vrijgegeven is aan het begin van de instructiecyclus. Gedurende de overdracht van de toestand ALU5 naar de toestand ALU6 wordt de inhoud van ALM overgedragen naar ALS. In de toestand ALU6, 1208, wordt de inhoud van de RAM-positie aangewezen door ALS overgedragen naar het tijdelijk informatieregister TA. De volgende toestand is ALU7, 1209, waarbij ALS wordt geïncrementeerd, en zijn inhoud wordt overgedragen naar T1 en naar B1, althans indien de auto-incrementerende eigenschap wordt vrijgegeven. De ALU-bewerking, aangegeven door de huidige instructie wordt uitgevoerd op twee operanden

geleverd door TA en TB in het geval van een tweevoudige bewerking, of door een enkele operand, geleverd door TA in het geval van een enkelvoudige bewerking. De volgende toestand is ALU8, 1210, waarbij het resultaat van de ALU-bewerking (ALU-UIT) wordt opgeslagen in de RAM-geheugenplaats, aangewezen door ALS. De toestand van de teller wordt dan vergeleken met de toestand van B0-B1 in het OER. Indien het comparatoruitgangssignaal waar is, wordt de ALU-bewerking voltooid; maar indien het comparatoruitgangssignaal vals is, keert de CPU terug naar de toestand ALU1 en wordt de ALU-bewerking herhaald.

Het automatisch laden van de aanwijsregisters

Een andere facultatieve eigenschap van de CPU in fig. 10, welke eigenschap geselecteerd kan worden met de programmabesturing van het OER, is het automatisch laden van de aanwijsregisters B0 en B1. Deze eigenschap is alleen ter beschikking voor de rechtstreekse adresseermodus, waarbij het volledige operandadres aanwezig is in de adrescode van de instructie. Wanneer het automatisch laden wordt vrijgegeven, worden de operandadressen, geleverd door de huidige instructie, automatisch opgeslagen in de gespecificeerde aanwijsregisters ten tijde van het voltooien van de instructiecyclus. Het automatisch laden van B0 en B1 wordt geregeld door de toestand van resp. de bitposities AL0 (1105 van fig. 11) en AL1 (1106 van fig. 11) in het OER, waarbij een "een"-toestand met automatisch laden vrijgeeft.

Het automatisch laden wordt uitgevoerd tijdens de adresformatietoestand van deinformatie-uitvoeringsfase. Een toestandsdiagram van de adresinformatietoestand voor de CPU van fig. 10 is weergegeven in fig. 13. Een voorbeeld van adresinformatie zal thans worden uiteengezet aan de hand van fig. 13.

Tijdens de instructie-ophaalfase, weergegeven door het blok 1301, wordt een instructiewoord, omvattende een op-code en een adresmoduscode, overgedragen van het ROM naar het IR. De CPU gaat dan naar de eerste adresinformatietoestand AF1, 1302. De baan die gevolgd wordt door de CPU door de adresinformatiestappen hangt af van de adresseermodus, aangegeven door de adresmoduscode in het IR, en hangt af of de bewerking, aangegeven door de op-code in het IR, enkelvoudig of tweevoudig is. Een D-type flipflop (DFF), 1025, in de CPU van fig. 10, regelt of het adres, geformeerd voor een tweevoudige bewerking, een bron of bestemmingsadres is. Voorafgaande aan de formatie van het bronadres van een

tweevoudige bewerking, wordt DFF vrijgegeven (DFF = 0); doch voorafgaande aan de formatie van het enkele operandadres van een enkelvoudige bewerking of het bestemmingsadres van een tweevoudige bewerking, wordt DFF ingesteld (DFF = 1). Indien de rechtstreekse adresseermodus het bronadres is worden de meest significante (bovenste) vier-bits van TO alle ingesteld op "een" en worden, op overeenkomstige wijze, indien de rechtstreekse adresseermodus een bestemmingsadres is, de meest significante vier-bits van T1 alle ingesteld op "een".

De CPU gaat vervolgens naar de toestand AF2, 1303. Indien de adresseermodus een indirecte adressering is, wordt de inhoud van BO overgedragen naar T1, indien DFF is ingesteld, of wordt de inhoud van BO overgedragen naar TO en wordt de inhoud van B1 overgedragen naar T1, indien DFF vrijgegeven is. De CPU gaat vervolgens rechtstreeks naar de toestand AF6, 1307. Indien de adresseermodus een onmiddellijke informatie is, wordt de toestand van PC overgedragen aan TO en gaat de CPU meteen naar de toestand AF6.

In de toestand AF2 wordt, voor het geval van een rechtstreekse adressering, de inhoud van PC overgedragen naar ALM, waarbij de inhoud van PC, zijnde het adres van de ROM-geheugenplaats, de eerste "knabbel" van het operandadres bevat. Gedurende de overgangstoestand van de toestand AF2 naar de toestand AF3, 1304, wordt de inhoud van ALM overgedragen naar ALS. In de toestand AF3 wordt ALS geïncrementeerd en wordt zijn inhoud overgedragen naar PC. Indien een andere "knabbel" van het ROM is vereist voor het completeren van de formatie van het adres, wordt het geïncrementeerde adres in ALS eveneens overgebracht naar ALM. De inhoud van de ROM-geheugenplaats, aangewezen door ALS, wordt dan overgedragen naar de onderste vier bits van TO (TOL) indien DFF isingesteld, of aan de onderste vier bits van T1 (T1L) indien DFF is vrijgegeven. Gedurende de overdracht van de toestand AF3 naar de toestand AF4, 1305, wordt de inhoud van ALM overgedragen naar ALS. In de toestand AF4. wordt ALS opnieuw geïncrementeerd, en wordt zijn inhoud overgedragen naar PC. De inhoud van de ROM-geheugenplaats, aangewezen door ALS, wordt overgedragen naar de middelste vier bits van TO (TOM) indien DFF is ingesteld, of naar de middelste vier bits van T1 (T1M) indien DFF is vrijgegeven. Opgemerkt wordt dat de bovenste vier bits van een operandadres in de directe adresseermodus "1111" is.

Indien het automatisch laden niet vrijgegeven is voor het hetzij

B0, of B1, gaat de CPU van de toestand AF4 rechtstreeks naar de toestand AF6. Overigens is de volgende toestand AF5, 1306. In de toestand AF5 wordt de inhoud van T0 overgebracht naar B0 indien DFF is ingesteld, en wordt het automatisch laden vrijgegeven voor B0 (d.w.z. AL0 = 1), of wordt de inhoud van T1 overgebracht naar B1 indien DFF is vrijgegeven en wordt het automatisch laden vrijgegeven voor B1 (d.w.z. AL1 = 1).

In de toestand AF6 wordt het gecompleteerde operandadres in T0 overgedragen naar ALM. Indien DFF is ingesteld in de toestand AF6, wordt de adresformatie gecompleteerd en gaat de CPU naar de ALU-functie, weergegeven door het blok 1309. Indien evenwel DFF is vrijgegeven, aangevende dat de adresinformatievoor een tweevoudige bewerking is incompleet, gaat de CPU eerst naar de toestand AF7, waarbij DFF is ingesteld en gaat vervolgens naar de toestand AF1 en herhaalt de adresformatiestappen voor het bestemmingsadres.

Auto-incrementering van aanwijsregisters

Een andere facultatieve kenmerk van de CPU van fig. 10, welke eigenschap wel of niet kan worden vrijgegeven met behulp van het OER, is het auto-incrementeren van de aanwijsregisters B0 en B1. Indien deze eigenschap wordt vrijgegeven voor een specifiek geheugenaanwijsregister, zal het adres, opgeslagen in dat register, automatisch voortschuiven (incrementeren) aan het einde van elke instructiecyclus, teneinde te wijzen naar de geheugenplaats van de eerste "knabbel" van de volgende operand. Het automatisch incrementeren van B0 en B1 wordt geregeld door de toestand van resp. de OER-bitposities AI0 (1103 van fig. 11) en AI1 (1104 van fig. 11). Een "een"-toestand geeft de auto-incrementering vrij.

Wanneer deze toestand is vrijgegeven, vindt het auto-incrementeren plaats van B0 en B1 tijdens de ALU-functietoestand van de informatie-uitvoerfase, zoals weergegeven door het toestandsdiagram van fig. 12. In fig. 12 wordt, zoals reeds bovenstaand in verband met de variabele operandbreedte wanneer de CPU in de toestand ALU3, 1205 is, een bronadres, opgeslagen in de ALS die wijst naar een "knabbel" van een eerste operand (verondersteld wordt een tweevoudige bewerking), geïncrementeerd, teneinde te wijzen naar de volgende "knabbel" van dezelfde operand, indien de ALU-bewerking herhaald wordt bij de volgende "knabbel". Indien de ALU-bewerking voltooid is voor alle "knabbels" van dezelfde operand, wijst de ALS naar de eerste "knabbel" van een nieuwe eerste operand.

Het geïncrementeerde adres wordt dan overgedragen naar TO en ALM. Indien de auto-incrementering van BO vrijgegeven is, (d.w.z. AIO = 1) wordt het geïncrementeerde adres eveneens overgedragen van ALS naar TO. In de toestand ALU7, 1209 wordt een bestemmingsadres, aanwezig in ALS en wijzende naar een "knabbel" van de tweede operand van een tweevoudige bewerking, geïncrementeerd en overgedragen naar T1. Indien de auto-incrementering van B1 vrijgegeven is (d.w.z. AI1 = 1) wordt het geïncrementeerde adres eveneens overgedragen van ALS naar B1. De ALU-bewerking wordt dan uitgevoerd op de corresponderende "knabbels" van de eerste en tweede operanden die resp. aanwezig zijn in TA en TB. Het resultaat van de ALU-bewerking wordt opgeslagen in de geheugenplaats, die daarvóór bezet was door de corresponderende "knabbel" van de tweede operand. De ALU-bewerking wordt indien noodzakelijk, herhaald, op aanvullende "knabbels" van de eerste en tweede operanden. Opgemerkt moet worden, dat wanneer het autoïncrementeren vrijgegeven is voor een bepaald aanwijsregister, zijn inhoud bij elke herhaling van de ALU-bewerking zodanig wordt geïncrementeerd dat aan het einde van de instructiecyclus het aanwijsregister altijd wijst naar de eerste "knabbel" van de volgende operand.

Toewijzing van adresregisters

Een andere facultatieve eigenschap, dat bestuurd kan worden door middel van het OER is de toewijzing van adresregisters. In sommige CPU-configuraties delen de RAM en randinrichtingen (bij voorbeeld een teletype-toetsenbord, kaartlaser, drukinrichting, magnetische bandaandrijving, etc.), gekoppeld aan de CPU, niet alleen hetzelfde adresveld in de instructie, maar bovendien delen zij dezelfde adressen. D.w.z. dat een gegeven adres kan refereren aan hetzij een RAM-geheugenplaats, hetzij aan een randinrichting. Identificatie van een adres, zoals een RAM-geheugenplaats of een randinrichting kan verkregen worden door afzonderlijk gescheiden RAM-referentie- en randreferentie-instructies (bij voorbeeld LEES RAM UIT, LEES RAND UIT, etc. te bepalen. Teneinde de op-codes instand te houden kan dezelfde instructie worden toegepast door te verwijzen naar zowel de RAM als de randinrichtingen. Derhalve zal de instructie aangeven of het adres voor een RAM-geheugenplaats is, of voor een randinrichting door te refereren naar hetzij een adresregister dat specifiek gereserveerd is voor het opslaan van een geheugenadres, hetzij een adresregister specifiek gereserveerd voor het opslaan van een rand-

inrichtingadres.

In de bekende CPU's waarin het gereserveerde adresregister wordt toegepast, wordt het aantal geheugenadresregisters en randadresregisters bepaald door het ontwerp van de CPU. D.w.z. dat de ontwerper van de CPU beslist over het aantal van elk type adresregister, gebaseerd op het soort van toepassing waarvoor de CPU werd ontwikkeld. Indien de CPU wordt toegepast in een geheugen met een intensieve toepassing, heeft deze meer geheugenadresregisters en minder randadresregisters. Het is duidelijk, dat een vaste toewijzing van adresregisters de veelzijdigheid van een CPU beperkt.

Een optimale veelzijdigheid kan worden verkregen in zowel geheugen-intensieve als rand-intensieve toepassingen door de toewijzing van het adresregister, onder invloed van een besturingsprogramma, variabel te maken. Een dergelijke facultatieve eigenschap, die ervoor moet zorgen, dat de toewijzing van adresregisters herhaaldelijk wordt vervangen tijdens een programma-uitvoering, is in het bijzonder van voordeel bij programma's die in bepaalde delen zeer geheugen-intensief zijn en in andere delen zeer rand-intensief zijn.

In fig. 14 is een blokdiagram aangegeven van een deel van een CPU 1400, waarmee een programmeerbare aanwijzing mogelijk is van verschillende adresregisters, in overeenstemming met de principe van deze uitvinding. Een groep van acht-twaalf-bitadresregisters, B0-B7 resp. aangegeven door 1401 t/m 1408, kunnen elk worden aangewezen voor het opslaan van hetzij een geheugenadres, hetzij een randinrichtingadres. De functies van de informatiebus 1409, programmateller 1410, instructieregister 1411, komen in hoofdzaak overeen met de functies, zoals uiteen is gezet in verband met de CPU van dig. 4. In het op-codeverlengregister (OER) 1412, wordt een op-codeverlengwoord opgeslagen, welk woord omvat de selectie van facultatieve CPU-eigenschappen, zoals hierboven beschreven eigenschappen, alsook de aanwijzing van de adresregisters voor het opslaan van RAM-adressen, en waarbij het opslaan van randinrichtingadressen wordt gespecificeerd door de toestanden van drie bitposities in het OER. De besturingsinrichting 1413 decodeert gelijktijdig de inhoud van zowel het IR alsook het OER (in feite gebeurt dat na elkaar) en wekt commandosignalen op, die overeenkomen met de processorbewerking, vereist door de gecombineerde inhoud van het IR en het OER. De adresmultiplexer 1418 koppelt het adresregister, aangewezen door de instructie, aan de twaalf-bitadres-

uit-bus 1417.

Wanneer een instructie verwijst naar een particulier adresregister, geeft de besturingsinrichting het geschikte adresregisterselectielijn 1415 vrij en zorgt de adres-MUX-besturingslijn ervoor, dat het adres opgeslagen wordt in dat register, dat beschikbaar is op de adres-uis-bus. Op hetzelfde tijdstip decodeert de besturingsinrichting de inhoud van OER, teneinde vast te stellen of het adresregister waarnaar verwezen wordt een RAM-adres bevat of een randinrichtingadres. Indien het een RAM-adres bevat, wordt de RAM-selectiebesturingslijn 1419 vrijgegeven; overigens wordt de randselectiebesturingslijn 1420 vrijgegeven. De lees/schrijf-besturingslijn 1421 levert een signaal aan het RAM, teneinde aan te geven of een geheugenreferentie-informatie in het RAM moet schrijven of informatie uit het RAM moet lezen.

In fig. 15 is een logicadiagram aangegeven van een deel van de besturingsinrichting 1500, die het adresregisteraanwijsveld AR0-AR2 van het OER decodeert. Indien de instructie de referentie van één van de adresregisters B0-B7 opvraagt, zal het instructiedecodeergedeelte 1503 ervoor zorgen, dat zowel de I/O-instructie-vrijgeeflijn 1504 als de adresregisterselectielijn 1505 gaan naar de toestand "een". In het voorbeeld van fig. 15 worden van de acht adresregisters B0-B7 twee gelijktijdig door de drie bits (AR0-AR2) in het adresregisteraanwijsveld van het OER aangewezen. De registers B0 en B1 zijn altijd gereserveerd voor de RAM-adressen, terwijl de registers B2 en B3 toegewezen worden om het RAM-adres, indien de bit AR2, 1504 een "nul" is, en toegewezen worden aan de randinrichtingadressen indien de bit AR2 een "een" is. Op een overeenkomstige wijze bepaalt de toestand van de bit AR1 de aanwijzing van de registers B4 en B5, en bepaalt de toestand van de bit AR0 de aanwijzing voor de registers B6 en B7. De keten, samengesteld uit logische poorten G1-G10 (referentienummers 1506-1515) detecteert de toestanden van de AR0-AR2, en die van de adresregisterselectielijn 1505, teneinde de juiste toestanden te leveren op de RAM-selectielijn of de randselectielijn, afhankelijk van de aanwijzing van het geselecteerde adresregister.

In fig. 16 is een voorbeeld weergegeven van een ingang/uitgang (I/O) koppelopstelling van de CPU van fig. 14 met een RAM en drie randinrichtingen. In dit voorbeeld is verondersteld, dat de CPU slechts één I/O-poort bezit, waarbij de RAM 1602 zijn eigen inwendige adresdecodeer-

inrichting bezit, en waarbij de drie randinrichtingen 1603-1605 alle ingangsinrichtingen zijn die informatie leveren aan de CPU. Indien de CPU een ingangsinformatie-instructie uitvoert en verwijst naar één van de adresregisters, komt het adres, opgeslagen in het aangewezen adresregister, beschikbaar via de adres-uit-uitgang van de CPU naar de randadresdecodeerinrichting (PAD) 1606 en aan de adresingang van het RAM. In dit voorbeeld is verondersteld, dat het adres, geleverd aan de adres-uit refereert aan zowel de randinrichting 2, 1604, en aan een geheugenplaats in het RAM, teneinde te kiezen tussen deze twee, waarbij de besturingsinrichting in de CPU het adresregisteraanwijsveld (AR0-AR2) van het OER detecteert, teneinde te bepalen of het geselecteerde adresregister toegewezen wordt aan het RAM of aan de randinrichtingen. De inhoud van het OER is tijdens een voorafgaande instructie opgeladen.

Indien het geselecteerde adresregister toegewezen wordt aan de randinrichtingen, wordt het RAM geblokkeerd via de RAM-selectie-uitgang, terwijl de PAD vrijgegeven wordt via de randselectie-uitgang. de PAD activeert de inrichting 2 en zorgt ervoor, dat de randinrichting-multiplexer 1607 wordt gekoppeld aan de inrichting 2 en aan de I/O-poort van de CPU. Indien evenwel het geselecteerde adresregister werd toegewezen aan het RAM, is de PAD geblokkeerd en is het RAM vrijgegeven. Een leessignaal, geleverd door de lees/inschrijfuitgang, zal er derhalve voor zorgen, dat de inhoud van de geadresseerde geheugenplaats in het RAM wordt overgedragen aan de CPU via de I/O-poort.

C O N C L U S I E S

1. Centrale verwerkingseenheid van een informatieverwerkend systeem voor het uitvoeren van een programma van opgeslagen instructies door middel van een reeks van instructiecycli, waarbij de centrale verwerkingseenheid ingericht is voor het ontvangen van een instructie gedurende de instructie-ophaalfase van een instructiecyclus, en, in responsie op deinstructie, het gedurende de informatie excecutiefase van de instructiecyclus uitvoeren van een rekenbewerking op de opgeslagen informatie aangegeven door de instructie, waarbij elke opgeslagen instructie voorzien is van een op-codegedeelte en van een adresgedeelte, waarbij de centrale verwerkingseenheid voorzien is van een instructieregister voor het tijdens de instructie-ophaalfase van de instructiecyclus ontvangen van het op-codegedeelte van de instructie en voor het opslaan van de op-code tijdens de gehele instructiecyclus en van een besturingsinrichting voor het tijdens de informatie-uitvoeringsfase van de instructiecyclus opwekken van commandosignalen die overeenkomen met de rekenbewerking, met het kenmerk, dat de centrale verwerkingseenheid (100) voorzien is van een op-codeverlengregister (421) voor het daarin opslaan van een geselecteerd op-codeverlengwoord en dat de besturingsinrichting (408) responsief is op de gecombineerde inhoud van het instructieregister (401) en van het op-codeverlengregister (421) voor het opwekken van commandosignalen, die overeenkomen met de respectieve gecombineerde inhoud, waarbij deinhoud van het op-codeverlengregister alleen verandert indien de centrale verwerkingseenheid (100) een instructie ontvangt voor het overdragen van een nieuw geselecteerd op-codeverlengwoord aan het op-codeverlengregister (421).

2. Centrale verwerkingseenheid volgens conclusie 1, voorzien van een geheugen voor het opslaan van instructies voor informatie, waarbij het geheugen voorzien is van meerdere geheugenplaatsen, waarbij elke geheugenplaats een afzonderlijk geheugenadres bezit, waarbij de centrale verwerkingseenheid voorzien is van een adresbus, die gekoppeld is aan een adresdecodeerinrichting, die samenwerkt met het geheugen en responsief is op een adres, aanwezig in de adresbus voor het selecteren van de geheugenplaats overeenkomstig het adres, met het kenmerk, dat de centrale verwerkingseenheid voorzien is van een rekenkundige logica-eenheid (411) die reageert op selectieve commandosignalen overeenkomstig de inhoud

van het op-codeverlengwoordregister en van het instructieregister van de besturingsinrichting voor het ontvangen van één of meer operanden vanuit het geheugen en voor het uitvoeren van een rekenkundige of logica-bewerking op de operand of operanden, bepaald door de gecombineerde inhoud van het instructieregister en het op-codeverlengregister, waarbij de rekenkundige logica-eenheid (411) geschikt is voor het verwerken van operanden met één of meer N-bitsegmenten, waarbij het aantal van N-bitsegmenten in een operand bepaald wordt door het op-codeverlengwoord, opgeslagen in het op-codeverlengregister.

3. Centrale verwerkingseenheid volgens conclusie 2, met het kenmerk, dat de centrale verwerkingseenheid 2N-bitoperandwoorden ontvangt vanuit het geheugen en de rekenkundige logica-eenheid die voorzien is van een bovenste en een onderste gedeelte voor het behandelen van het meest significante N-bitsegment en het minst significante N-bitsegment van een 2 N-bitoperandwoord uit het geheugen gehaald is wanneer een aangewezen bitpositie in het op-codeverlengregister een eerste binaire toestand omvat, waarbij de beide gedeelten van de rekenkundige logica-eenheid worden vrijgegeven voor het verwerken van het gehel 2 N-bitoperandwoord, en wanneer de aangewezen bitpositie de andere binaire toestand omvat, waarbij het bovenste gedeelte van de rekenkundige logica-eenheid geblokkeerd is en het onderste gedeelte vrijgegeven is bewerkingen uit te voeren op het minst significante N-bitsegment van het operandwoord en de meest significante N-bitsegmenten onveranderd blijven.

4. Centrale verwerkingseenheid volgens conclusie 3, met het kenmerk, dat een keten aangebracht is voor het verwisselen van de meest significante en de minst significante N-bitsegmenten van een 2 N-bitoperandwoord.

5. Centrale verwerkingseenheid volgens conclusie 2, voorzien van ten minste één bidirectioneel ingang/uitgangspoort, die geschikt is om te worden gekoppeld aan een aantal randinrichtingen, die elk een verschillend randinrichtingsadres hebben en gekoppeld zijn aan een deel van het geheugen, waarbij enkele van de geheugenplaatsen in het gedeelte van het geheugen adressen bezitten die gemeenschappelijk zijn voor de randinrichtingadressen; van een adresuitgang gekoppeld aan de adresbus en geschikt om te worden gekoppeld aan een randinrichtingadresdecodeerinrichting, die samenwerkt met de randinrichtingen, waarbij de randinrichtingadresdecodeerinrichting responsief is op een adres, toegevoerd

aan de adresuitgang voor het selecteren van een randinrichting, dat wordt gekoppeld aan de ingang/uitgangspoort en aan een deel van de geheugenadresdecodeerinrichting die responsief is op het adres aan de adresingang voor het selecteren van een geheugenplaats in het gedeelte van het geheugen waarin toegang verzocht wordt door de centrale verwerkingseenheid via de ingangs/uitgangspoort; van een eerste uitgang, die geschikt is om te worden gekoppeld aan de randadresdecodeerinrichting voor het vrijgeven van de randadresdecodeerinrichting indien de eerste uitgang in een eerste binaire toestand is en voor het blokkeren van de randadresdecodeerinrichting indien de eerste uitgang in een tweede binaire toestand is; van een tweede uitgang, die geschikt is om te worden gekoppeld aan het gedeelte van de geheugenadresdecodeerinrichting voor het vrijgeven van het gedeelte van de geheugenadresdecodeerinrichting indien de tweede uitgang in een eerste binaire toestand is en voor het blokkeren van het gedeelte van de geheugenadresdecodeerinrichting indien de tweede uitgang in een tweede binaire toestand is; en van een aantal adresregisters voor het daarin opslaan van adressen, waarbij ten minste één van de adresregisters toegewezen is voor het opslaan van hetzij een geheugenadres dat overeenkomt met een geheugenplaats in het gedeelte van het geheugen het ervan een randinrichtingadres, met het kenmerk, dat het op-codeverlengregister de toewijzing regelt voor elk toewijsbaar adresregister indien een instructie wordt uitgevoerd door de centrale verwerkingseenheid voor het overdragen van de inhoud van een bepaald adresregister aan de adresuitgang; dat de besturingsinrichting responsief is op de inhoud van het op-codeverlengregister voor het leveren van een eerste binaire toestand aan de eerste uitgang en van een tweede binaire toestand aan de tweede uitgang, indien het bepaalde adresregister is aangewezen voor het opslaan van een randinrichtingadres; en dat de besturingsinrichting een tweede binaire toestand levert aan de eerste uitgang en een eerste binaire toestand aan de tweede uitgang indien het bepaalde adresregister is aangewezen voor het opslaan van een geheugenadres, dat overeenkomt met een geheugenplaats in het gedeelte van het geheugen.

FIG. 1

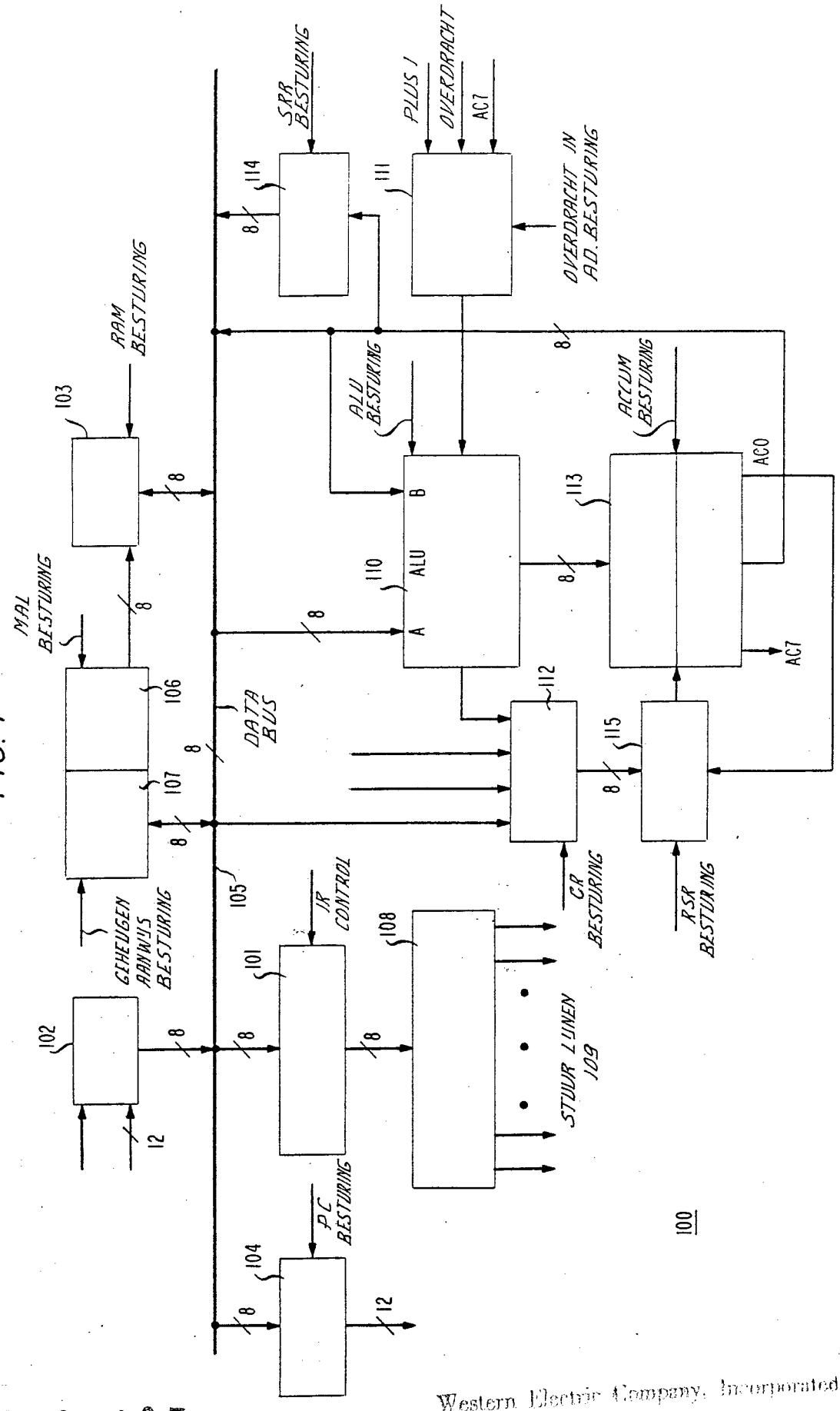


FIG. 2

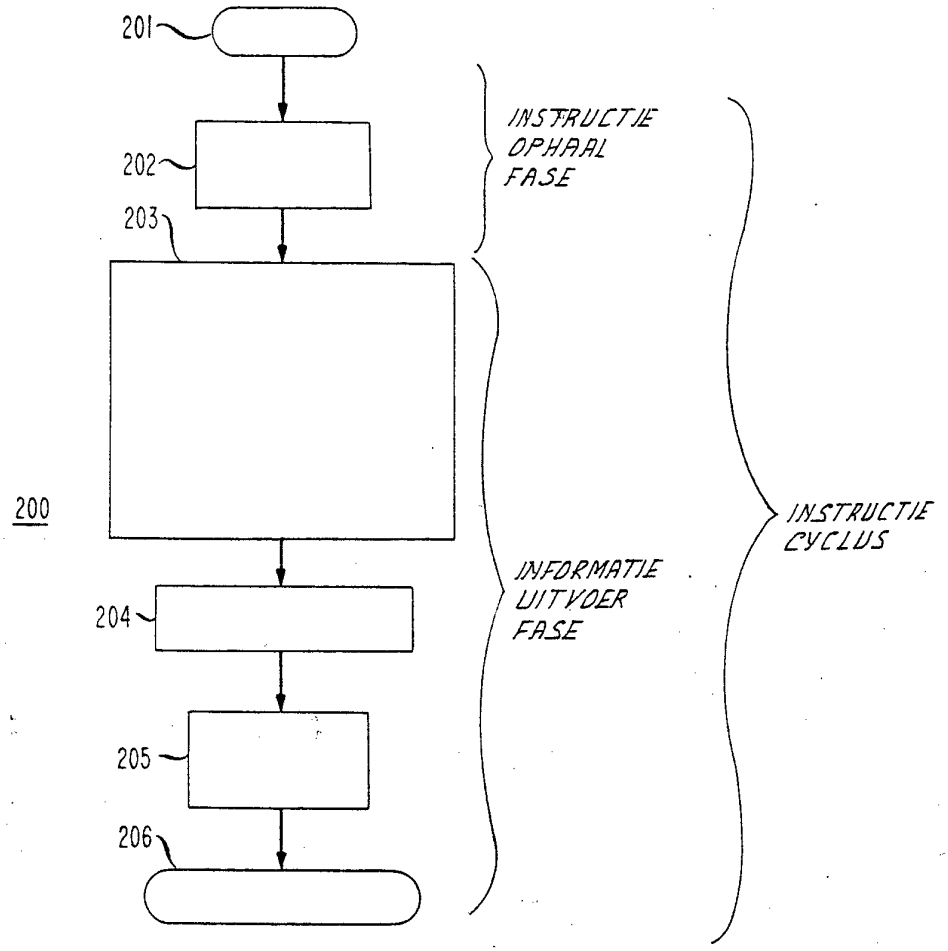


FIG. 9

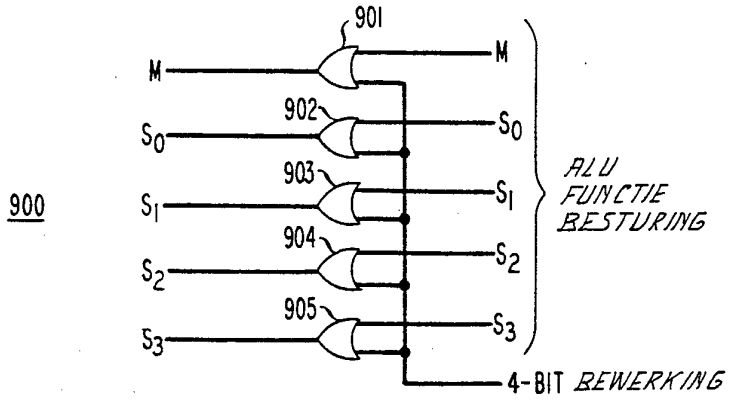
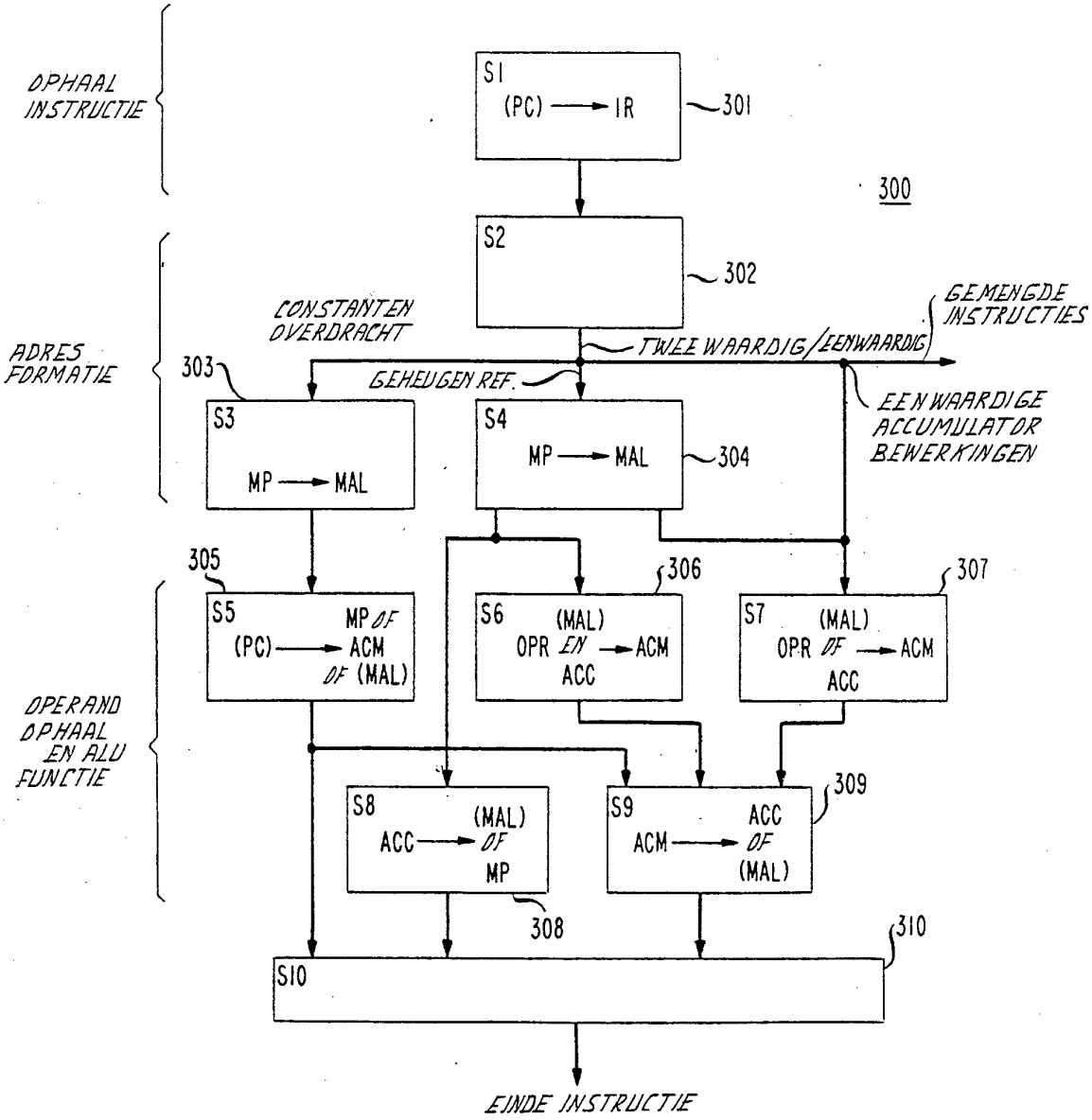


FIG. 3



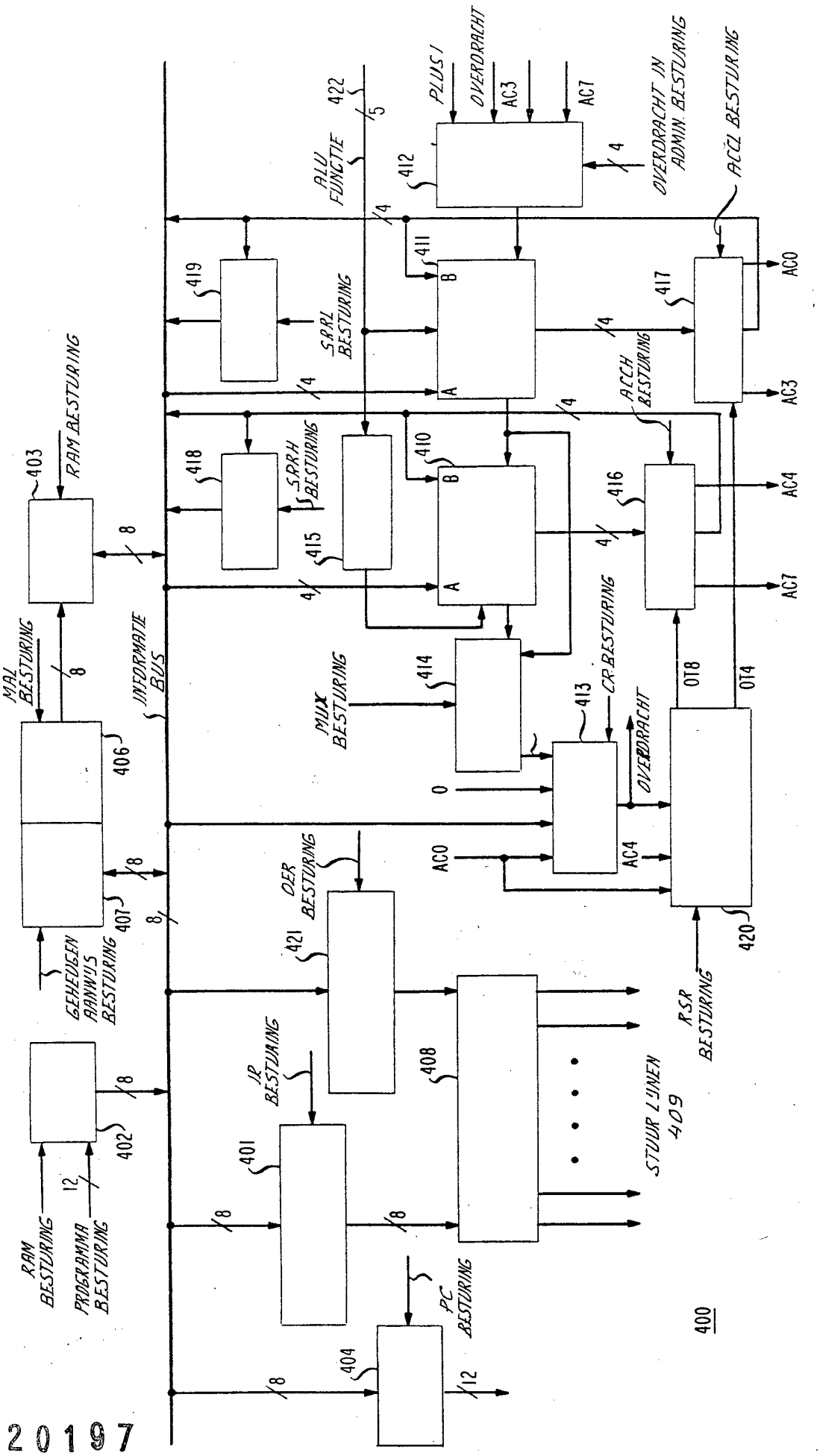


FIG. 4

7920197

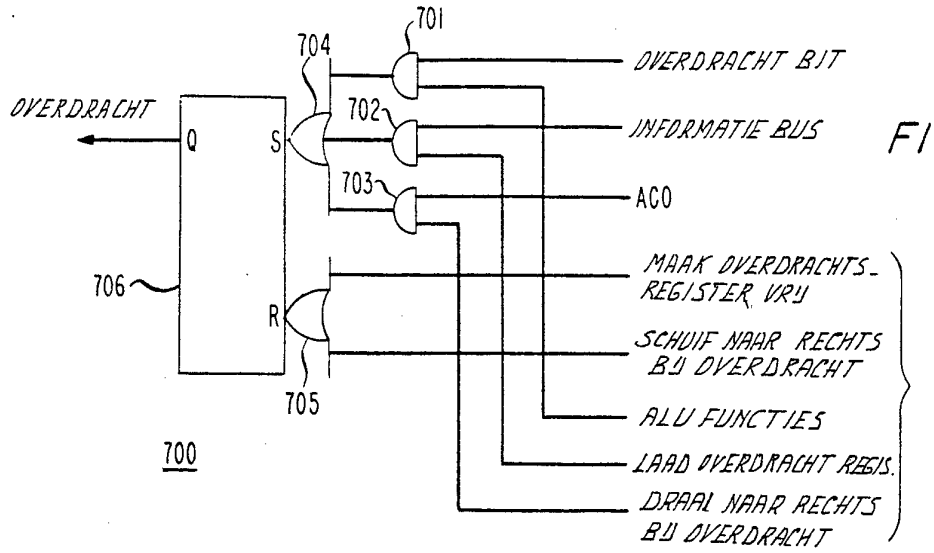


FIG. 7

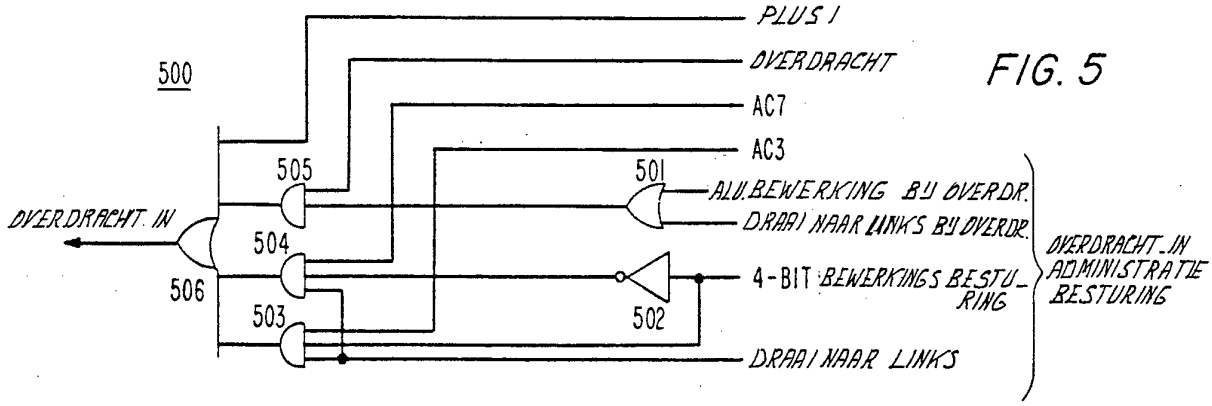


FIG. 5

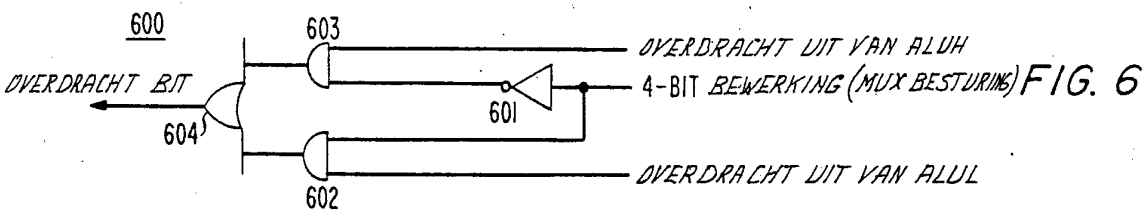


FIG. 6

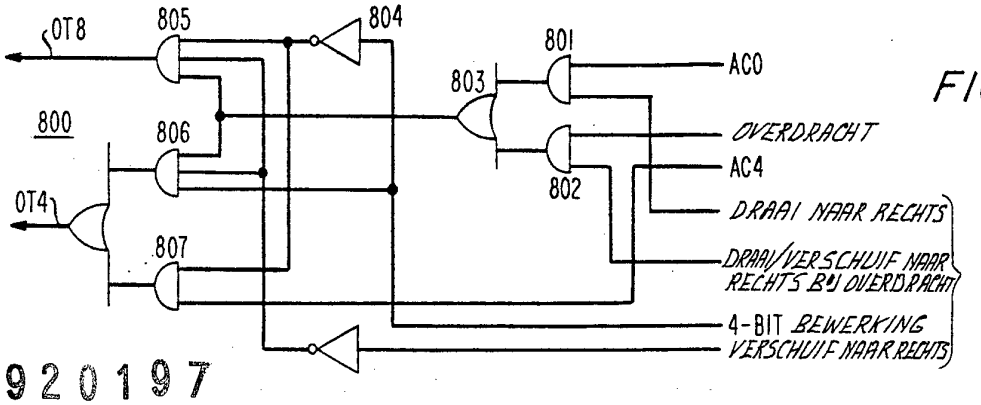


FIG. 8

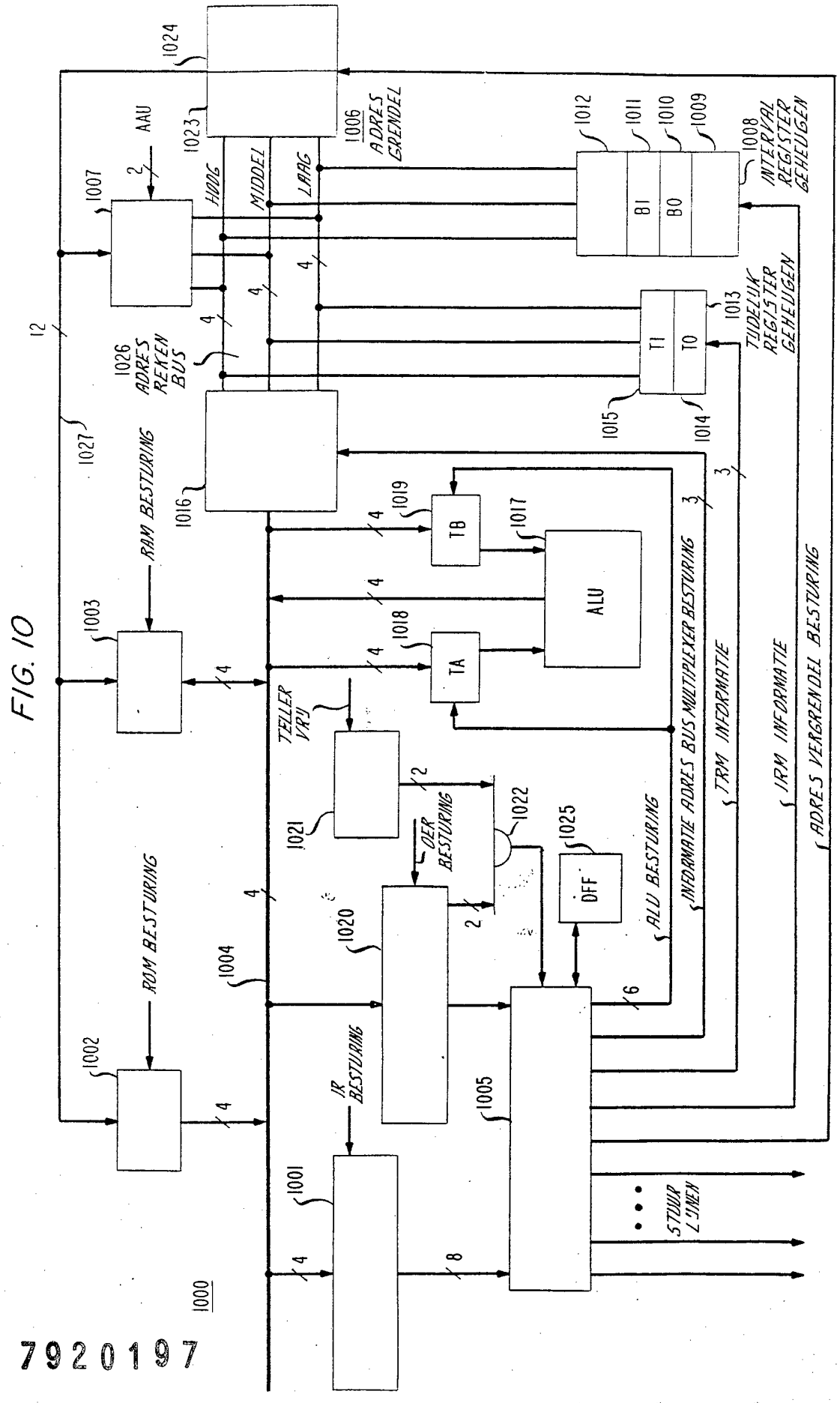


FIG. 10

7920197

Western Electric Company, Incorporated

FIG. 14

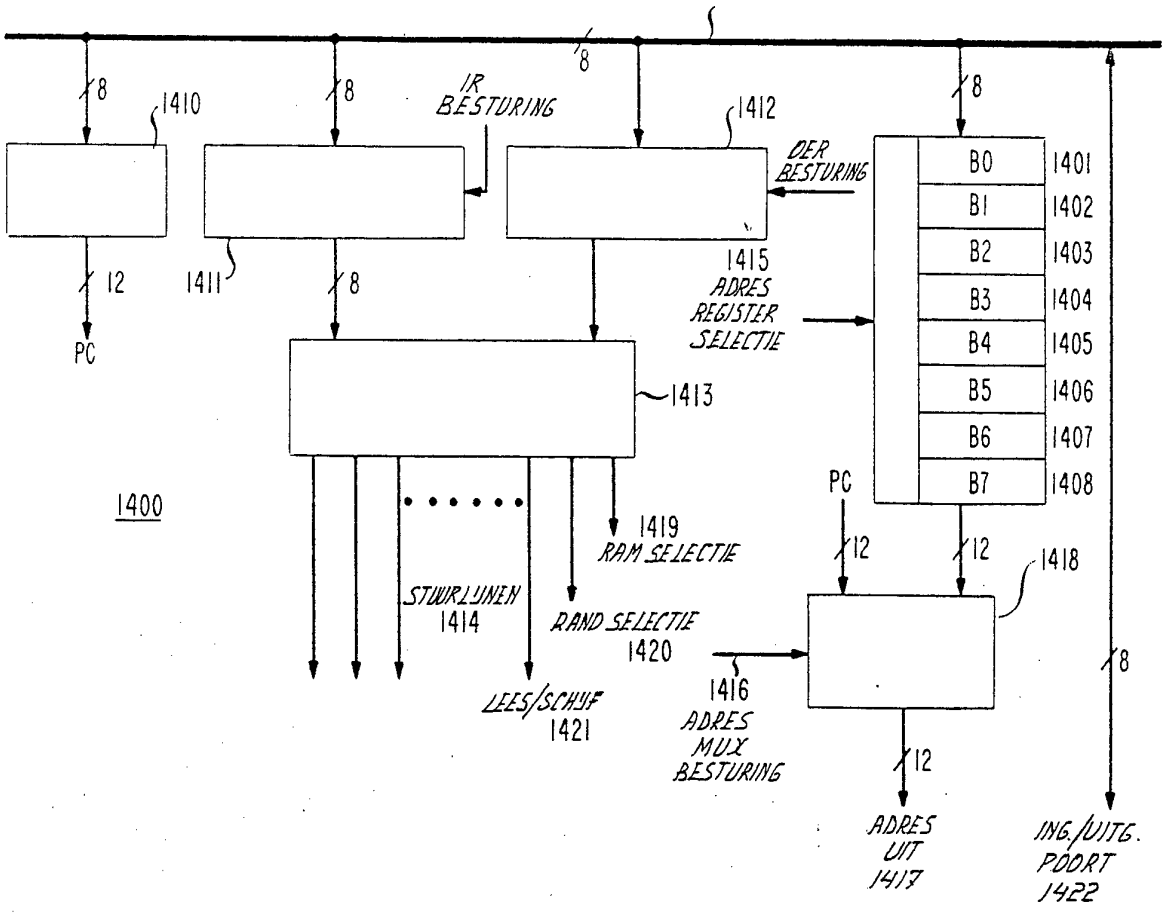


FIG. 11

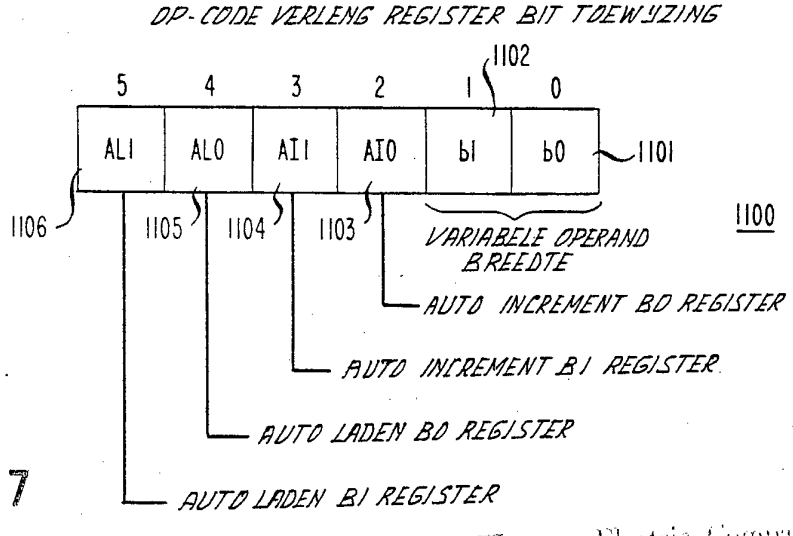
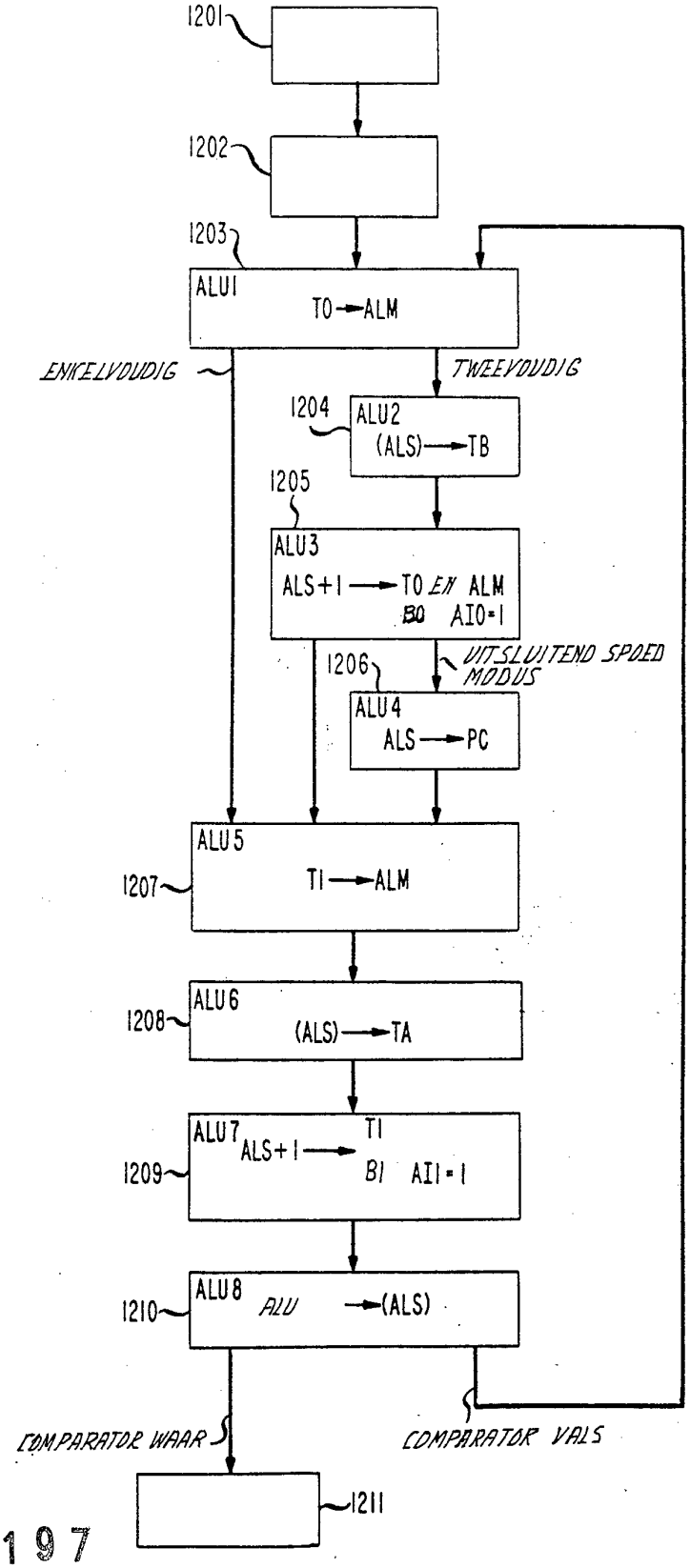
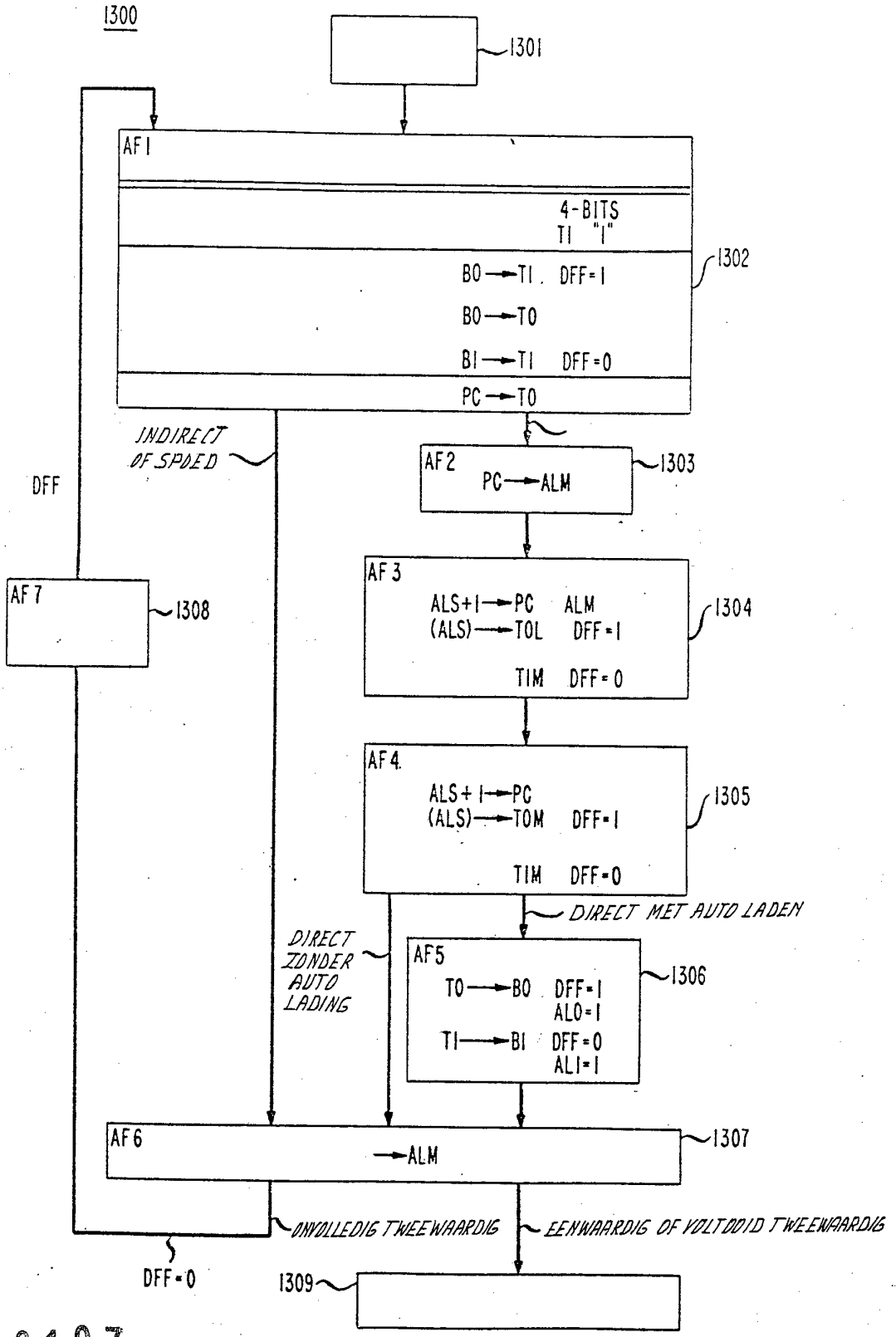


FIG. 12



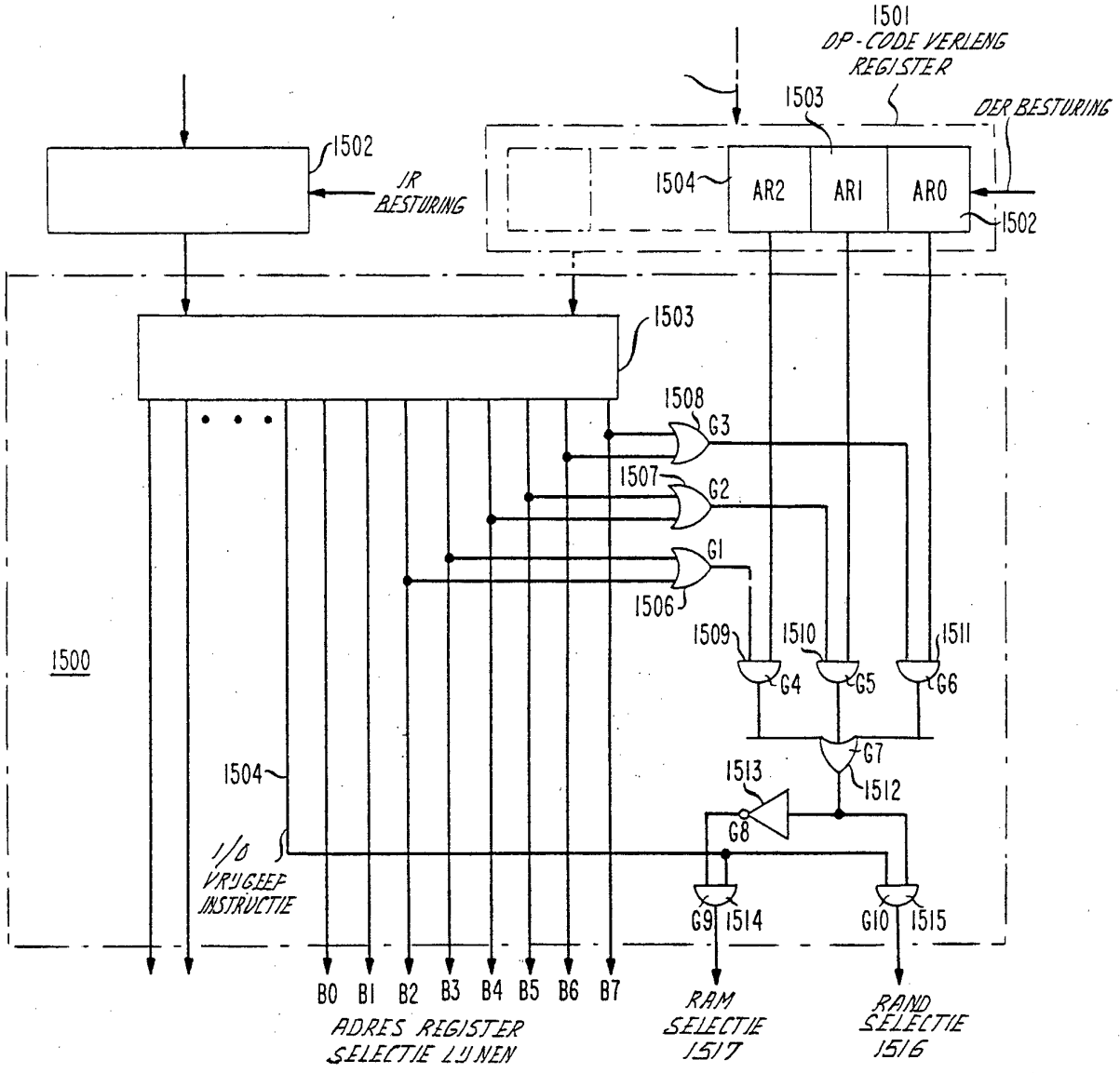
7920197

FIG. 13



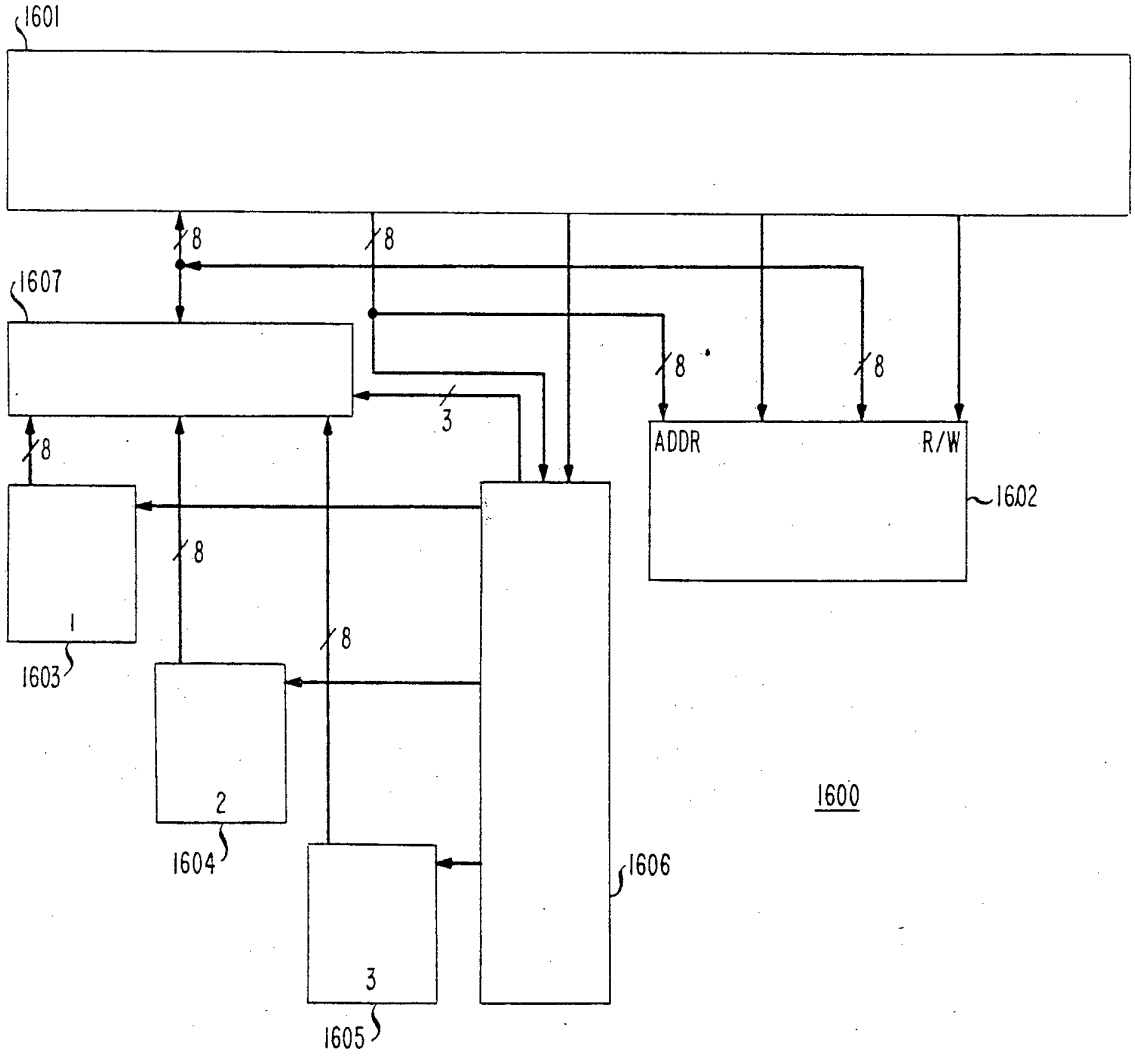
7920197

FIG. 15



7920197

FIG. 16



7920197