(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 697 37 184 T2** 2007.10.04

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 0 856 800 B1**
(21) Deutsches Aktenzeichen: **697 37 184.0**
(86) PCT-Aktenzeichen: **PCT/JP97/03061**
(96) Europäisches Aktenzeichen: **97 937 870.0**
(87) PCT-Veröffentlichungs-Nr.: **WO 1998/010354**
(86) PCT-Anmeldetag: **02.09.1997**
(87) Veröffentlichungstag der PCT-Anmeldung: **12.03.1998**
(97) Erstveröffentlichung durch das EPA: **05.08.1998**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **03.01.2007**
(47) Veröffentlichungstag im Patentblatt: **04.10.2007**

(51) Int Cl.[8]: ***G06F 17/10*** (2006.01)

(30) Unionspriorität:

23456596	**04.09.1996**	**JP**
13592297	**08.05.1997**	**JP**

(84) Benannte Vertragsstaaten:
DE, FR, GB, IT

(73) Patentinhaber:
Seiko Epson Corp., Tokyo, JP

(74) Vertreter:
Hoffmann, E., Dipl.-Ing., Pat.-Anw., 82166 Gräfelfing

(72) Erfinder:
KUBOTA, Satoshi, Suwa-shi, Nagano 392, JP;
KUDO, Makoto, Suwa-shi, Nagano 392, JP;
MIYAYAMA, Yoshiyuki, Suwa-shi, Nagano 392, JP

(54) Bezeichnung: **SCHALTKREIS, MIKROCOMPUTER UND ELEKTRONISCHER APPARAT FÜR DIE DATENVERARBEITUNG**

Beschreibung

TECHNISCHES GEBIET

[0001] Diese Erfindung betrifft eine Informationsverarbeitungsschaltung, einen Mikrocomputer und eine elektronische Anlage.

HINTERGRUNDTECHNIK

[0002] Seit kurzem steigt die Nachfrage nach einem Mikrocomputer, der in Lage ist, Summe-von-Produkten-Rechenbefehle rasch auszuführen. Wenn ein Mikrocomputer Summe-von-Produkten-Rechnungen rasch ausführen kann, kann er als ein digitaler Signalprozessor (DSP), eine spezifische Bildverarbeitungs-IC oder eine spezifische Tonverarbeitungs-IC fungieren, was die Kosten von Endprodukten senken und zur Vereinfachung von System beitragen würde.

[0003] Ein Summe-von-Produkten-Rechenbefehl wird von einem Mikrocomputer wie nachstehend beschrieben ausgeführt. Zuerst sind erste Summe-von-Produkten-Eingangsdaten in einem ersten Bereich des Speichers und zweite Summe-von-Produkten-Eingangsdaten in einem zweiten Bereich desselben gespeichert worden. Der Mikrocomputer verwendet dann zwei Adressen, die vom Inhalt interner Mehrzweckregister vorgegeben werden, um die in diesem ersten und zweiten Bereich gespeicherten ersten und zweiten Summe-von-Produkten-Eingangsdaten auszulesen. Er multipliziert dann diese ersten und zweiten Summe-von-Produkten-Eingangsdaten und eine Summe-von-Produkten-Rechenschaltung addiert das Resultat zu einem internen Register für das Resultat der Summe-von-Produkten (MAC-Register).

[0004] Ein Mikrocomputer, der einen Summe-von-Produkten-Rechenbefehl ausführen kann, ist jedoch mit den folgenden Problemen behaftet:

(1) Wenn der Mikrocomputer die Summe-von-Produkten-Rechnungen eine Mehrzahl Male ausführen soll, muss ein Programm geschrieben werden, damit eine Sequenz der gleichen Anzahl Summe-von-Produkten-Rechenbefehlen vorliegt wie die Anzahl der Male, die der Summe-von-Produkten-Rechenbefehl zu wiederholen ist. Wenn also die Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, erhöht wird, ist es erforderlich, eine entsprechende Anzahl Summe-von-Produkten-Rechenbefehlen zu speichern, was die erforderliche Speicherkapazität erhöht. Eine Technik, die zur Lösung dieses Problems in Frage käme, wäre ein Programm zu schreiben, das Summe-von-Produkten-Rechnungen ausführt, während die Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, dekrementiert wird, und dann aus dieser Schleife austritt, wenn die Anzahl der Ausführungen null erreicht. Mit dieser Technik wird jedoch die Zeit länger, die zum Ausführen jeder Summe-von-Produkten-Rechnung erforderlich ist.

(2) Wenn die Summe-von-Produkten-Rechnungen sequentiell ausgeführt werden, ist die Ausführungszeit für eine Rechnung durch die Zeit begrenzt, die erforderlich ist, um die ersten und zweiten Summe-von-Produkten-Eingangsdaten aus dem Speicher auszulesen.

(3) Bisher werden die ersten und zweiten Summe-von-Produkten-Eingangsdaten, die jeweils z. B. 16 Bits umfassen, multipliziert und das Ergebnis der Multiplikation wird zu einem 48-Bit-MAC-Register (Register für das Resultat Summe-von-Produkten) addiert. In einem solchen Fall ist es erforderlich, die Addition der 48-Bit-Daten innerhalb einer Taktperiode abzuschließen, so dass diese Addierverarbeitung ein kritischer Pfad wird. Da das MAC-Register nur 48 Bits lang ist, läuft es bald über, wenn die Summe-von-Produkten-Rechnung sehr oft auszuführen ist.

[0005] Die US-A5,596,760 offenbart eine Informationsverarbeitungsschaltung, die eine Steuerschaltung zum Empfangen von Befehlen aufweist, die einen Summe-von-Produkten-Rechenbefehl enthalten, zur Analyse der Befehle und zum Steuern der Ausführung dieser Befehle sowie eine Summe-von-Produkten-Rechenschaltung zum Ausführen einer Summe-von-Produkten-Rechnung unter der Steuerung dieser Steuerschaltung auf Basis dieses Summe-von-Produkten-Rechenbefehls. Die Summe-von-Produkten-Rechenschaltung führt eine Anzahl von Malen eine Summe-von-Produkten-Rechnung aus, die durch eine Anzahl-von-Ausführungen-Information vorgegeben ist, die im Summe-von-Produkten-Rechenbefehl enthalten ist.

[0006] Die vorliegende Erfindung wurde mit dem Ziel erarbeitet, die oben beschriebenen Probleme zu lösen, und die Bereitstellung einer Informationsverarbeitungsschaltung, eines Mikrocomputers und einer elektronischen Anlage, die so ausgelegt sind, dass sie das Speichernutzungsverhältnis eines Programms verbessern, das die Summe-von-Produkten-Rechenbefehlen verwendet, ist eine Aufgabe derselben.

[0007] Eine weitere Aufgabe dieser Erfindung ist die Bereitstellung einer Informationsverarbeitungsschaltung, eines Mikrocomputers und einer elektronischen Anlage, die Verbesserungen hinsichtlich der Geschwindigkeit, mit der Summe-von-Produkten-Rechenbefehle ausgeführt werden, ermöglichen.

OFFENBARUNG DER ERFINDUNG

[0008] Diese Aufgaben werden durch eine Informationsverarbeitungsschaltung gemäß Anspruch 1,

durch einen Mikrocomputer gemäß Anspruch 3 und eine elektronische Anlage gemäß Anspruch 4 gelöst. Bevorzugte Ausführungsformen der Erfindung sind Gegenstand der Unteransprüche.

[0009] Die Anzahl-von-Ausführungen-Information zur Vorgabe der Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, ist im Summe-von-Produkten-Rechenbefehl enthalten. Die Summe-von-Produkten-Rechenschaltung führt unter der Steuerung der Steuerschaltung Summe-von-Produkten-Rechnungen eine Anzahl von Malen aus, die von der Anzahl-von-Ausführungen-Information im Summe-von-Produkten-Rechenbefehl vorgegeben wird. Damit ist es möglich, mit einem einzigen Befehl Summe-von-Produkten-Rechnungen eine gewünschte Anzahl von Malen auszuführen. Damit kann die erforderliche Speicherkapazität für die Summe-von-Produkten-Rechnungen erheblich verringert und damit das Speichernutzungsverhältnis im Vergleich zu der Technik, die eine Sequenz der gleichen Anzahl Summe-von-Produkten-Rechenbefehlen als die Summe-von-Produkten-Rechnungen verwendet, verbessert werden. Außerdem ist es nicht erforderlich, den Summe-von-Produkten-Rechenbefehl während der Ausführung der Summe-von-Produkten-Rechnungen jedes Mal abzurufen, so dass Verzögerungen bei der Ausführung des Summe-von-Produkten-Rechenbefehls vermieden werden können.

[0010] Der Summe-von-Produkten-Rechenbefehl weist einen Operanden zur Vorgabe eines Registers aus einer Gruppe bestehend aus einem Register für die Anzahl der Male, die die Summe-von-Produkten-Rechnung auszuführen ist, einem Register für die ersten Summe-von-Produkten-Eingangsdaten und einem Register für die zweiten Summe-von-Produkten-Eingangsdaten auf. Die Steuerschaltung kann ein anderes als das eine Register gemäß einer gegebenen Regel auf Basis des Operanden, der dieses eine Register vorgibt, vorgeben. Damit ist es möglich, die Bitlänge des Befehls zu verringern und den Programmcode kompakter zu machen.

[0011] Die Informationsverarbeitungsschaltung kann ferner eine Schaltung zum Dekrementieren einer Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, aufweisen, synchron mit der Ausführung einer Summe-von-Produkten-Rechnung, wobei diese Anzahl in einem Register in der Steuerschaltung gespeichert ist, wobei die Summe-von-Produkten-Rechenschaltung Summe-von-Produkten-Rechnungen ausführt, bis die Anzahl der Ausführungen einen gegebenen Wert erreicht. Dadurch erübrigt es sich, die Anzahl der Ausführungen bei jeder Ausführung der Summe-von-Produkten-Rechnung aus dem Speicher auszulesen, was eine Verbesserung der Verarbeitungsgeschwindigkeit möglich macht. Wenn ein Interrupt während der Ausführung einer Mehrzahl Summe-von-Produkten-Rechnungen aufgetreten ist, ermöglicht diese Konfiguration auch die Wiederaufnahme der Summe-von-Produkten-Rechnungen nach der Interrupt-Verarbeitung auf Basis der im Register gespeicherten Anzahl von Ausführungen.

KURZBESCHREIBUNG DER ZEICHNUNGEN

[0012] Fig. 1 ist ein Blockdiagramm eines Konfigurationsbeispiels eines Mikrocomputers.

[0013] Fig. 2 ist ein Flussdiagramm des Funktionsablaufs von Ausführungsform 1.

[0014] Fig. 3A, Fig. 3B und Fig. 3C zeigen die Beziehungen zwischen Registern und im Speicher gespeicherten Daten.

[0015] Fig. 4 ist ein Flussdiagramm des Funktionsablaufs eines Vergleichsbeispiels.

[0016] Fig. 5A und Fig. 5B sind Diagramme, die Probleme bei diesem Vergleichsbeispiel zeigen.

[0017] Fig. 6A, Fig. 6B, Fig. 6C sind Diagramme verschiedener Ausführungsformen des Summe-von-Produkten-Rechenbefehls.

[0018] Fig. 7 ist ein Flussdiagramm des Funktionsablaufs von Ausführungsform 2.

[0019] Fig. 8A und Fig. 8B zeigen die Beziehungen zwischen Registern und im Speicher gespeicherten Daten.

[0020] Fig. 9 ist ein Flussdiagramm des Funktionsablaufs von Ausführungsform 3.

[0021] Fig. 10A ist ein Diagramm, das die von der Ausführungsform 3 angewendete Technik zum Speichern von Daten im Speicher zeigt; Fig. 10B ist ein Beispiel eines Taktdiagramms eines Vergleichsbeispiels; und Fig. 10C ist ein Beispiel eines Taktdiagramms der Ausführungsform 3.

[0022] Fig. 11A ist ein Blockdiagramm eines Konfigurationsbeispiels von Ausführungsform 4; und Fig. 11B ist ein Beispiel eines Taktdiagramms desselben.

[0023] Fig. 12 ist ein Blockdiagramm eines Konfigurationsbeispiels von Ausführungsform 5.

[0024] Fig. 13 ist ein Beispiel eines Taktdiagramms der Ausführungsform 5.

[0025] Fig. 14A und Fig. 14B sind Diagramme des Funktionsablaufs der Zustandsmaschine.

[0026] Fig. 15 ist ein Beispiel eines Taktdiagramms während der Interrupt-Erzeugung.

[0027] Fig. 16 ist ein Konfigurationsbeispiel des Mikrocomputers der Ausführungsform 6.

[0028] Fig. 17A, Fig. 17B und Fig. 17C sind beispielhafte Blockdiagramme verschiedener Bestandteile der elektronischen Anlage.

[0029] Fig. 18A, Fig. 18B und Fig. 18C sind beispielhafte Außenansichten verschiedener Bestandteile der elektronischen Anlage.

BESTE ART ZUR AUSFÜHRUNG DER ERFINDUNG

[0030] Bevorzugte Ausführungsformen dieser Erfindung werden nachstehend unter Bezugnahme auf die beiliegenden Zeichnungen beschrieben. Es sei darauf hingewiesen, dass die nachstehende Beschreibung hauptsächlich Beispiele betrifft, bei denen eine Informationsverarbeitungsschaltung gemäß dieser Erfindung in einem Mikrocomputer verwendet wird.

Ausführungsform 1

[0031] Die erste Ausführungsform dieser Erfindung führt eine Summe-von-Produkten-Rechnung in einer Summe-von-Produkten-Rechenschaltung eine Anzahl von Malen aus, die auf Basis der Anzahl-von-Ausführungen-Information vorgegeben ist, die im Summe-von-Produkten-Rechenbefehl enthalten ist.

[0032] Fig. 1 zeigt ein Blockdiagramm eines Mikrocomputers **101** mit einer internen Summe-von-Produkten-Rechenschaltung **104**. Dieses Funktionsblockdiagramm ist der Ausführungsform 1 sowie den Ausführungsformen 2, 3 und 4 gemeinsam, die später beschrieben werden. Der Mikrocomputer **101** von Fig. 1 verarbeitet 32-Bit-Daten. Die Summe-von-Produkten-Rechenschaltung **104** multipliziert erste und zweite Summe-von-Produkten-Eingangsdaten MDA und MDB von jeweils 16 Bits und addiert die 32-Bit-Daten, die das Ergebnis dieser Multiplikation darstellen, zu einem 64-Bit-MAC-Register **107**. Es ist jedoch zu beachten, dass der Anwendungsbereich dieser Erfindung weder durch die Bitlänge der vom Mikrocomputer und der Summe-von-Produkten-Rechenschaltung verarbeiteten Daten beschränkt ist, noch durch Faktoren wie die Anzahl der Mehrzweckregister im Mikrocomputer.

[0033] Der Mikrocomputer **101** von Fig. 1 weist eine Steuerschaltung **102** zur Verarbeitung von 32-Bit-Daten auf; eine Bussteuereinheit (bus control unit; BCU) **108** zum Steuern eines Busses, der den Mikrocomputer **101** mit einem Speicher **110** verbindet; die Summe-von-Produkten-Rechenschaltung **104**, die Summe-von-Produkten-Rechnungen ausführt; einen Interrupt-Kontroller **130**, der verschiedene Interrupts von innerhalb und außerhalb des Mikrocomputers empfängt und Interrupt-Anforderungen an die Steuerschaltung **102** sendet; und eine Recheneinheit (arithmetic and logic unit; ALU) **132**, die arithmetische Operationen mit den Daten wie Addition oder Subtraktion sowie logische Operationen wie ANDs, ORs und logisches Schieben ausführt.

[0034] Im vorliegenden Fall empfängt die Steuerschaltung **102** Befehle, die einen Summe-von-Produkten-Rechenbefehl enthalten, analysiert die so empfangenen Befehle und steuert die Ausführung der analysierten Befehle, wobei sie 16-Bit-Befehle verwendet. Die Steuerschaltung **102** weist ein Mehrzweckregister **103** auf, das aus sechzehn 32-Bit-Registern R0 bis R15 besteht, und einen Programmzähler (program counter; PC) **120**. Unter der Steuerung der Steuerschaltung **102** führt die Summe-von-Produkten-Rechenschaltung **104** Summe-von-Produkten-Rechnungen aus, und die ALU **132** führt arithmetische und logische Operationen aus. Die Steuerschaltung **102**, die Summe-von-Produkten-Rechenschaltung **104** und die ALU **132** bilden gemeinsam die Zentraleinheit (central processing unit; CPU).

[0035] Die Steuerschaltung **102**, die BCU **108** und die Summe-von-Produkten-Rechenschaltung **104** übertragen Daten über einen internen Datenbus **109**. Die BCU **108** verwendet einen externen Adressbus **111** und einen externen Datenbus **112** und liest aus dem Speicher **110** erste und zweite Summe-von-Produkten-Eingangsdaten MDA und MDB aus. Es wird allerdings darauf hingewiesen, dass der Anwendungsbereich dieser Erfindung nicht davon beeinflusst wird, ob der Speicher **110** innerhalb oder außerhalb des Mikrocomputers **101** angeordnet ist.

[0036] Die Summe-von-Produkten-Rechenschaltung **104** weist ein TEMPm-Register **122** und eine TEMPn-Register **124** zum vorübergehenden Halten der ersten und zweiten Summe-von-Produkten-Eingangsdaten MDA und MDB auf, einen Multiplizierer **105** zum Multiplizieren der vorübergehend gehaltenen Daten MDA und MDB, einen Addierer **106** zum Ausführen einer Addition mit dem Resultat dieser Multiplikation und das 64-Bit MAC-Register (Register für das Summe-von-Produkten-Resultat) **107** zum Halten des Resultats dieser Addition. Die Summe-von-Produkten-Rechenschaltung **104** empfängt die 16-Bit-Daten MDA und MDB, addiert das Resultat der Multiplikation daraus sowie den Inhalt des MAC-Registers **107** und speichert das Ergebnis dieser Addition im MAC-Register **107**.

[0037] Die Funktionsweise dieser Ausführungsform wird nunmehr anhand des Flussdiagramms von Fig. 2 und der Speichertabellen der Fig. 3A, Fig. 3B

und Fig. 3C beschrieben.

[0038] Vor der Ausführung des Summe-von-Produkten-Rechenbefehls werden die ersten Summe-von-Produkten-Eingangsdaten MDA_0 bis MDA_L in einem ersten Bereich **10** des Speichers und die zweiten Summe-von-Produkten-Eingangsdaten MDB_0 bis MDB_L in einem zweiten Bereich **12** gespeichert, wie aus Fig. 3A ersichtlich ist. Die Startadressen des ersten und zweiten Bereichs **10** und **12** werden zuvor in Register Rm und Rn im Mehrzweckregister **103** geladen, so dass Rm und Rn auf die Startdaten MDA_0 und MDB_0 der ersten und zweiten Summe-von-Produkten-Eingangsdaten zeigen. Die Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, wird zuvor in ein Register Rc im Mehrzweckregister **103** geladen, und außerdem wird das MAC-Register **107** initialisiert.

[0039] Wenn die Steuerschaltung **102** in diesem Zustand einen Summe-von-Produkten-Rechenbefehl empfängt (mit anderen Worten, wenn der Befehl, auf den der PC **120** zeigt, ein Summe-von-Produkten-Rechenbefehl ist, wie in Fig. 3A dargestellt ist), werden die verschiedenen Prozesse, die zur Ausführung dieses Summe-von-Produkten-Rechenbefehls erforderlich sind, unter der Steuerung der Steuerschaltung **102**, die diesen Summe-von-Produkten-Rechenbefehl analysiert, ausgeführt.

[0040] Mit anderen Worten, die ersten Summe-von-Produkten-Eingangsdaten MDA_0, die von der im Register Rm gespeicherten Adresse vorgegeben werden, werden über die BCU **108** aus dem Speicher **110** ausgelesen und im TEMPm-Register **122** gespeichert (Schritt S1 in Fig. 2). Auf ähnliche Weise werden MDB, auf die Rn zeigt, über die BCU **108** aus dem Speicher **110** ausgelesen und im TEMPn-Register **124** gespeichert (Schritt S2). Es ist zu beachten, dass (Rm) und (Rn) in den Schritten S1 und S2 die ersten und zweiten Summe-von-Produkten-Eingangsdaten im Speicher bedeuten, die von den in den Registern Rm und Rn gespeicherten Adressen vorgegeben werden.

[0041] Die in Rm und Rn gespeicherten Adressen werden dann um 2 inkrementiert (Schritte S3 und S4). Bei dieser Ausführungsform wird der Speicherplatz in Mindesteinheiten von 8 Bit breiten Bytes adressiert, und die ersten und zweiten Summe-von-Produkten-Eingangsdaten sind 16-Bit-Daten. Wenn die Adressen in Rm und Rn um 2 inkrementiert werden, werden deshalb Rm und Rn aktualisiert, um auf die nächsten Summe-von-Produkten-Eingangsdaten MDA_1 und MDB_1 zu zeigen, wie aus Fig. 3B ersichtlich ist.

[0042] Die im TEMPm-Register **122** und im TEMPn-Register **124** gespeicherten 16-Bit-Daten MDA_0 und MDB_0 werden vom Multiplizierer **105** multipliziert, das Ergebnis dieser Multiplikation wird vom Addierer **106** zum Inhalt des MAC-Registers **107** addiert und das Ergebnis dieser Addition im MAC-Register **107** gespeichert (Schritt S5).

[0043] Die Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist und die im Register Rc gespeichert wird, wird dann dekrementiert (Schritt S6). Mit anderen Worten, die Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, wird synchron mit der Summe-von-Produkten-Rechnung dekrementiert. Es ist zu beachten, dass die Anzahl von Malen gemäß Fig. 2 am Ende jeder Summe-von-Produkten-Rechnung zu dekrementieren ist, aber diese Dekrementierung der Anzahl von Ausführungen könnte zumindest synchron mit der Ausführung der Summe-von-Produkten-Rechnung erfolgen. Diese Dekrementierung wird von der ALU **132** z. B. in Fig. 1 ausgeführt.

[0044] Der Mikrocomputer bestimmt dann, ob die in Rc gespeicherte Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, ein gegebener Wert ist wie z. B. null oder nicht (Schritt S7). Wenn sie nicht null ist, geht die Verarbeitung auf Schritt S1 zurück. In diesem Fall sind die Adressen in Rm und Rn wie oben beschrieben um 2 inkrementiert worden, so dass die Operanden der nächsten Summe-von-Produkten-Rechnung die nächsten Summe-von-Produkten-Eingangsdaten MDA_1 und MDB_1 sind (siehe Fig. 3B).

[0045] Wenn dagegen die Summe-von-Produkten-Rechnung die in Rc eingestellte Anzahl von Malen wiederholt worden ist und Rc damit null erreicht hat, wird die im PC **120** gespeicherte Adresse um 2 inkrementiert wie in Fig. 3C dargestellt (Schritt S8). Dies stellt sicher, dass der Summe-von-Produkten-Rechenbefehl endet und auch der PC **120** auf den nächsten Befehl weist. Da die Befehlslänge bei dieser Ausführungsform 16 Bit bzw. 2 Bytes beträgt, wird dann, wenn der Wert (Adresse) im PC **120** um 2 inkrementiert wird, der nächste Befehl zwei Bytes weiter angezeigt.

[0046] Die Steuerung der Steuerschaltung **102** über die Komponenten wie die Summe-von-Produkten-Rechenschaltung **104**, die BCU **108** und die ALU **132** zur Ausführung der obigen Verarbeitung stellt sicher, dass die Summe-von-Produkten-Rechnung die gewünschte Anzahl von Malen durch einen einzigen Befehl ausgeführt werden kann.

[0047] Ein Flussdiagramm der Verarbeitung durch einen Mikrocomputer, bei dem es sich um ein Vergleichsbeispiel für diese Ausführungsform handelt, ist in Fig. 4 dargestellt. Die Schritte T1 bis T5 in Fig. 4 sind gleich den Schritten S1 bis S5 in Fig. 2. Das Vergleichsbeispiel von Fig. 4 unterscheidet sich jedoch von der Ausführungsform gemäß Fig. 2 darin,

dass es den Wert im PC **120** um 2 inkrementiert und den Summe-von-Produkten-Rechenbefehl beendet, ohne den Wert in Rc zu dekrementieren oder zu bestimmen, ob der Wert in Rc null ist oder nicht. Außerdem gibt es keine Information im Summe-von-Produkten-Rechenbefehl zur Vorgabe der Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist.

[0048] Um deshalb sicherstellen zu können, dass die Technik gemäß Fig. 4 die Summe-von-Produkten-Rechnung die gewünschte Anzahl von Malen ausführt, ist es erforderlich, das Programm mit der gleichen Anzahl Summe-von-Produkten-Rechenbefehlen und Anzahl von Malen zu schreiben, wie aus Fig. 5A ersichtlich ist, was insofern zu Problemen führt, als die Größe des Programmcodes zunimmt und ein größerer Speicher zum Speichern dieser Befehle benötigt wird.

[0049] Eine Technik zur Lösung dieser Probleme ist, ein Programm so zu schreiben, dass die Summe-von-Produkten-Rechnung ausgeführt wird, während die Anzahl C der Male, die diese Summe-von-Produkten-Rechnung auszuführen ist, dekrementiert wird (Schritte U1 und U2), und das aus dieser Schleife austritt, wenn die Anzahl C von Malen null erreicht (Schritt U3). Wenn diese Technik angewendet wird, wird jedoch die Zeit länger, die zum einmaligen Ausführen einer Summe-von-Produkten-Rechnung erforderlich ist. Mit anderen Worten, zusätzlich zur erforderlichen Zeit zur Ausführung des Summe-von-Produkten-Rechenbefehls (Schritt U1) benötigt diese Technik die Zeit, die zum Ausführen eines Befehls erforderlich ist, mit dem die Anzahl C von Malen dekrementiert wird (Schritt U2), und die Zeit, die zur Bestimmung erforderlich ist, ob die Anzahl der Ausführungen null erreicht hat oder nicht (Schritt U3). Dies bedeutet, dass die Verarbeitungsdauer mindestens zwei Taktperioden länger ist als die der Ausführungsform von Fig. 2, was zu einem Abfall der Verarbeitungsgeschwindigkeit führt.

[0050] Im Gegensatz dazu ist es bei dieser Ausführungsform nicht erforderlich, ein Programm zu schreiben, das eine Folge aus der gleichen Anzahl Summe-von-Produkten-Rechenbefehlen wie Anzahl der Ausführungen enthält. Es ist auch nicht erforderlich, ein Programm zu schreiben, das die Anzahl der Ausführungen dekrementiert und bestimmt, ob diese null ist oder nicht. Dadurch wird es möglich, die Summe-von-Produkten-Rechnung eine gewünschte Anzahl von Malen mittels eines einzigen Summe-von-Produkten-Rechenbefehls auszuführen, wobei eine effizientere Nutzung der Codegröße, eine Verkleinerung des Speichers zum Speichern der Befehle und eine Erhöhung der Verarbeiturgsgeschwindigkeit möglich werden.

[0051] Bei dieser Ausführungsform ist es nicht erforderlich, bei jeder Ausführung der Summe-von-Produkten-Rechnung einen Summe-von-Produkten-Rechenbefehl abzurufen. Dadurch werden Verzögerungen bei der Ausführung des Summe-von-Produkten-Rechenbefehls aufgrund eines Konflikts zwischen dem Auslesen der Summe-von-Produkten-Eingangsdaten und dem Abrufen des Summe-von-Produkten-Rechenbefehls vermieden und eine Verringerung der Leistungsaufnahme bedingt durch überflüssige Abrufvorgänge ermöglicht.

[0052] Es lassen sich verschiedene andere Ausführungsformen eines Summe-von-Produkten-Rechenbefehls konzipieren, der eine Vorgabe der Anzahl von Malen enthält, die die Summe-von-Produkten-Rechnung auszuführen ist.

[0053] So könnte z. B. der Summe-von-Produkten-Rechenbefehl einen 6 Bit langen Operationscode zur Vorgabe des Summe-von-Produkten-Rechenbefehls aus einer Mehrzahl Befehle und einen 4-Bit-Operanden zur Vorgabe des Registers Rc aus den 16 Mehrzweckregistern aufweisen, wie in Fig. 6A dargestellt ist. In einem solchen Fall gibt die Steuerschaltung **102** das Register Rm für die ersten Summe-von-Produkten-Eingangsdaten und das Register Rn für die zweiten Summe-von-Produkten-Eingangsdaten gemäß einer gegebenen Regel vor, die auf diesem Rc vorgebenden Operanden basiert. Wenn z. B. das Mehrzweckregister R13 vom Summe-von-Produkten-Rechenbefehl von Fig. 6A als Rc zugeordnet wird, inkrementiert die Verarbeitung R13 um +1 und +2, um die Mehrzweckregister R14 und R15 als Rm und Rn zuzuordnen. Dies ermöglicht es, die Befehlslänge auf höchstens 16 Bits zu beschränken, wodurch die Codegröße rationell genutzt und die zum Speichern der Befehle erforderliche Speichergröße verringert werden kann. Die Technik gemäß Fig. 6A ist bei einer Befehlsvorratsarchitektur, in der sämtliche Befehle eine feste Länge, z. B. 16 Bits, haben, besonders effektiv, um sicherzustellen, dass die Codegröße rationell genutzt wird. Es sei darauf hingewiesen, dass der Summe-von-Produkten-Rechenbefehl in Fig. 6A einen Operanden aufweist, der Rc vorgibt, er aber ebenso gut einen Operanden aufweisen kann, der Rm oder Rn vorgibt.

[0054] Der in Fig. 6B dargestellte Summe-von-Produkten-Rechenbefehl weist einen 6 Bit langen Operationscode, einen 4-Bit-Operanden, der Rc vorgibt, einen 4-Bit-Operanden, der Rm vorgibt, und einen 4-Bit-Operanden auf, der Rn vorgibt. Mit anderen Worten, Rc, Rm und Rn werden direkt vom Summe-von-Produkten-Rechenbefehl vorgegeben. Im Vergleich zur Technik gemäß Fig. 6A hat dies den Nachteil einer längeren Befehlslänge, aber den Vorteil, dass keine Verarbeitung erforderlich ist, um andere Operanden aus dem einen Operanden zu spezifizieren.

[0055] Bei der Technik gemäß Fig. 6C sind Rc, Rm und Rn so eingestellt, dass sie spezifische Register für die Anzahl der Ausführungen bzw. die ersten und zweiten Summe-von-Produkten-Eingangsdaten sind. In diesem Fall weist der Summe-von-Produkten-Rechenbefehl einen Operationscode auf, der diese spezifischen Register als implizite Operanden verwendet. Damit kann die Codegröße effizient genutzt werden. Es ist zu beachten, dass zwar Rc, Rm und Rn sämtlich in Fig. 6C als spezifische Register dargestellt sind, die Konfiguration aber auch nur zwei davon als spezifische Register vorsehen könnte.

[0056] Ferner sei darauf hingewiesen, dass obwohl die ersten und zweiten Summe-von-Produkten-Eingangsdaten bei dieser Ausführungsform bei jeder Ausführung der Summe-von-Produkten-Rechnung auf Basis der in den Registern Rm und Rn gespeicherten Adressen aus dem Speicher ausgelesen werden, der in Rc gespeicherte Wert als die Anzahl der Ausführungen verwendet wird. Der Grund dafür ist, dass das Auslesen der Anzahl von Ausführungen aus dem Speicher bei jeder Ausführung der Summe-von-Produkten-Rechnung einen Abfall der Verarbeitungsgeschwindigkeit verursachen würde. Nachdem bei dieser Ausführungsform die Anzahl der Ausführungen aus dem Speicher in das Register Rc geladen worden ist, wird sie von der ALU **132** dekrementiert, so dass es anders als bei den Summe-von-Produkten-Eingangsdaten nicht notwendig ist, die Anzahl der Ausführungen erneut aus dem Speicher auszulesen. Es ist jedoch zu beachten, dass der Bereich dieser Erfindung nicht auf diese Technik zur Vorgabe der Anzahl von Ausführungen beschränkt ist.

Ausführungsform 2

[0057] Die zweite Ausführungsform dieser Erfindung empfängt während der Ausführung einer Mehrzahl Summe-von-Produkten-Rechnungen einen Interrupt und nimmt die Ausführung der angehaltenen Summe-von-Produkten-Rechnungen nach der Beendigung des Interrupt wieder auf. Die Ausführungsform 2 wird nachstehend anhand des Flussdiagramms von Fig. 7 und der Speichertabellen der Fig. 8A und Fig. 8B beschrieben.

[0058] Bei der zuvor beschriebenen Ausführungsform 1 kann eine Mehrzahl Summe-von-Produkten-Rechnungen mittels eines einzigen Summe-von-Produkten-Rechenbefehls ausgeführt werden. Da jedoch die Ausführung dieser Mehrzahl Summe-von-Produkten-Rechnungen als die Ausführung eines einzigen Befehls gilt, ändert sich der Wert im PC **120** nur, wenn die gleiche Anzahl Summe-von-Produkten-Rechnungen wie die in Rc eingestellten Ausführungen abgeschlossen worden ist, so dass die Verarbeitung nicht zum nächsten Befehl weiter geht. Eine Interrupt-Anforderung vom Interruptkontroller **130** in Fig. 1 wird normalerweise an der Grenze zwischen einem Befehl und einem anderen Befehl verarbeitet. Ein Übergang zur Interrupt-Verarbeitung nimmt deshalb viel Zeit in Anspruch, wenn dieser während der Ausführung einer Mehrzahl Summe-von-Produkten-Rechnungen erfolgen sollte.

[0059] Aus diesem Grund führt die Ausführungsform 2 die in Fig. 7 dargestellte Verarbeitung aus (die Schritte V1 bis V6 in Fig. 7 sind identisch mit den Schritten S1 bis S6 in Fig. 2). Mit anderen Worten, nach der Bestimmung, ob die Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist und deren Wert in Rc gespeichert ist, null ist oder nicht (Schritt V7), bestimmt der Mikrocomputer, ob eine Interrupt-Anforderung vorliegt oder nicht (Schritt V8). Wenn keine Interrupt-Anforderung vorliegt, wird die Verarbeitung der Summe-von-Produkten-Rechnungen fortgesetzt, und die Summe-von-Produkten-Rechnung erfolgt auf der Basis der nächsten Summe-von-Produkten-Eingangsdaten. Wenn dagegen eine Interrupt-Anforderung vorliegt, wird der Summe-von-Produkten-Rechenbefehl vorübergehend beendet, ohne dass der PC **120** inkrementiert wird (die Verarbeitung von Schritt V9 entfällt). Es sei angenommen, dass eine Interrupt-Anforderung z. B. während der Summe-von-Produkten-Rechnungsverarbeitung der Summe-von-Produkten-Eingangsdaten MDA_{K-1} und MDB_{K-1} in Fig. 8A erzeugt wird. In diesem Fall hat die Verarbeitung in den Schritten V3 und V4 von Fig. 7 Rm und Rn so eingestellt, dass sie auf die nächsten Summe-von-Produkten-Eingangsdaten MDA_K und MDB_K zeigen. Der PC **120** zeigt nach wie vor auf diesen Summe-von-Produkten-Rechenbefehl und nicht auf den nächsten Befehl (siehe H1 in Fig. 8A).

[0060] Das Ende des Summe-von-Produkten-Rechenbefehls ermöglicht eine Verzweigung zu einem Interrupt-Verarbeitungsunterprogramm und damit zur Ausführung der Interrupt-Verarbeitung. Eine solche Interrupt-Verarbeitung erfolgt normalerweise an der Grenze zwischen zwei Befehlen. Vor der Verzweigung zur Interrupt-Verarbeitung stapelt der Interrupt-Handler in diesem Fall den Wert im PC **120**, der auf die Rücksprungadresse nach Beendigung der Interrupt-Verarbeitung zeigt. Wenn bei dieser Ausführungsform der Fluss jedoch bei während der Ausführung des Summe-von-Produkten-Rechenbefehls zur Interrupt-Verarbeitung verzweigt, endet der Summe-von-Produkten-Rechenbefehl ohne Inkrementieren des PC **120** um 2, wie aus den Schritten V8 und V9 in Fig. 7 zu ersehen ist. Deshalb zeigt der PC **120** immer noch auf den Summe-von-Produkten-Rechenbefehl wie in H2 von Fig. 8B dargestellt, so dass derselbe Summe-von-Produkten-Rechenbefehl nach Beendigung der Interrupt-Verarbeitung erneut ausgeführt werden kann.

[0061] Zu diesem Zeitpunkt sind die Werte in Rc,

Rm und Rn die gleichen wie die an der Abzweigung zur Interrupt-Verarbeitung, wie mit H3, H4 und H5 in Fig. 8B dargestellt ist. Dies ermöglicht eine ordnungsgemäße Wiederaufnahme der Ausführung der Summe-von-Produkten-Rechnung, die von der Interrupt-Verarbeitung unterbrochen worden war. Mit anderen Worten, vor der Unterbrechung durch den Interrupt wird die Verarbeitung bis zur Handhabung der Summe-von-Produkten-Eingangsdaten MDA_{K-1} und MDB_{K-1} abgeschlossen. Nach dem Anhalten kann die Ausführung der Summe-von-Produkten-Rechnung ab der Verarbeitung hinsichtlich MDA_K und MDB_K wieder aufgenommen werden.

[0062] Die oben beschriebene Ausführungsform 2 ermöglicht den Empfang eines Interrupt während der Ausführung einer Mehrzahl Summe-von-Produkten-Rechnungen und die Interrupt-Verarbeitung. Außerdem ermöglicht sie es, die Wartezeit für die Interrupt-Verarbeitung gleich der der Ausführungsform gemäß Fig. 4 zu machen.

Ausführungsform 3

[0063] Die dritte Ausführungsform dieser Erfindung ermöglicht es, erste und zweite Summe-von-Produkten-Eingangsdaten in einem einzigen Speicherzugriff aus benachbarten Speicherbereichen auszulesen, in denen diese ersten und zweiten Summe-von-Produkten-Eingangsdaten gespeichert sind. Die Funktionsweise der Ausführungsform 3 wird nunmehr unter Bezugnahme auf das Flussdiagramm von Fig. 9 und die Speichertabellen der Fig. 10A, Fig. 10B und Fig. 10C beschrieben.

[0064] Bei der Ausführungsform 3 werden erste Summe-von-Produkten-Eingangsdaten MDA und zweite Summe-von-Produkten-Eingangsdaten MDB in benachbarten Speicherplätzen gespeichert, wie aus Fig. 10A ersichtlich ist. Beispielsweise werden die Daten MDB_0 nach den Daten MDA_0 und MDA_1 sowie MDB_1 nach MDB_0 gespeichert. Mit anderen Worten, wenn N natürliche Zahlen repräsentiert, werden die ersten Summe-von-Produkten-Eingangsdaten MDA in den Adressen 4N und die zweiten Summe-von-Produkten-Eingangsdaten MDB in den Adressen 4N+2 gespeichert. Dieser Punkt ist von der in den Fig. 3A bis Fig. 3C dargestellten Konfiguration verschieden, bei der die ersten Summe-von-Produkten-Eingangsdaten MDA zusammen im ersten Speicherbereich **10** und die zweiten Summe-von-Produkten-Eingangsdaten zusammen im zweiten Speicherbereich **12** gespeichert werden.

[0065] Die Ausführungsform 3 unterscheidet sich von den Ausführungsformen 1 und 2 auch darin, dass sie zwei Register Rm und Rn anstelle von drei verwendet. Mit anderen Worten sieht die Konfiguration vor, dass MDA_0 und MDB_0 gemäß Rm ausgelesen werden, dann MDA_1 und MDB_1 gemäß einem Wert ausgelesen werden, der dem Wert von Rm plus 4 entspricht.

[0066] Nunmehr wird die Funktionsweise der Ausführungsform 3 beschrieben. Zunächst werden die ersten und zweiten Summe-von-Produkten-Eingangsdaten MDA_0 und MDB_0 aus dem Speicher ausgelesen, wie durch die im Register Rm gespeicherte Adresse vorgegeben. Die oberen 16 Bits (2 Bytes) der ausgelesenen Daten werden im TEMPm-Register **122** und die unteren 16 Bits im TEMPn-Register **124** gespeichert (Schritt W1).

[0067] Mit anderen Worten, die Datenübertragung zwischen dem Speicher **110** und der Summe-von-Produkten-Rechenschaltung **104** erfolgt bei dieser Ausführungsform über einen 32-Bit-Bus wie aus Fig. 10A ersichtlich. Das bedeutet, dass 32 Bits (4 Bytes) breite Daten durch einen einzigen Speicherzugriff ausgelesen werden können, wobei die oberen 16 Bits der ausgelesenen Daten die ersten Summe-von-Produkten-Eingangsdaten MDA_0 und die unteren 16 Bits die zweiten Summe-von-Produkten-Eingangsdaten MDB_0 werden.

[0068] Nach dem Lesen der Summe-von-Produkten-Eingangsdaten inkrementiert der Mikrocomputer den Wert von Rm um 4 und führt die Summe-von-Produkten-Rechnung aus (Schritte W2 und W3). Durch das Inkrementieren des Wertes von Rm um 4 können die nächsten Summe-von-Produkten-Eingangsdaten MDA_1 und MDB_1 vorgegeben werden, wie in Fig. 10A dargestellt ist. Es ist zu beachten, dass die Verarbeitung in den Schritten W6 bis W9 die gleiche ist wie in den Schritten V6 bis V9 von Fig. 7.

[0069] Bei den Ausführungsformen 1 und 2 werden bei jedem Speicherzugriff (eine Taktperiode) nur entweder die ersten oder die zweiten Summe-von-Produkten-Eingangsdaten MDA bzw. MDB ausgelesen, wie in Fig. 10B dargestellt ist. Da die Summe-von-Produkten-Rechnung erst ausgeführt werden kann, wenn sowohl MDA als auch MDB vorhanden sind, kann die Summe-von-Produkten-Rechnung tatsächlich nur in zwei Taktzyklen ausgeführt werden. Mit anderen Worten, die Zeit, die zur Ausführung einer einzigen Summe-von-Produkten-Rechnung erforderlich ist, ist die Zeit, die zwei Speicherzugriffsoperationen in Anspruch nehmen.

[0070] Im Gegensatz dazu ist es bei der Ausführungsform 3 möglich, sowohl die ersten als auch die zweiten Summe-von-Produkten-Eingangsdaten MDA und MDB durch einen einzigen Speicherzugriff auszulesen, wie in Fig. 10C dargestellt ist. Deshalb kann in jeder Taktperiode eine Summe-von-Produkten-Rechnung ausgeführt werden, so dass die zur Ausführung der Summe-von-Produkten-Rechnung erforderliche Zeit der Zeit entspricht, die für einen ein-

zigen Speicherzugriff anfällt. Dadurch wird eine beträchtliche Verbesserung der Verarbeitungsgeschwindigkeit möglich.

Ausführungsform 4

[0071] Die vierte Ausführungsform dieser Erfindung betrifft eine Summe-von-Produkten-Rechnung, die sich eines Pipeline-Verfahrens bedient, bei dem eine Multiplikation in einer ersten Stufe, die Addition des Multiplikationsergebnisses zu einem unteren Register für ein erstes Summe-von-Produkten-Ergebnis in einer zweiten Stufe und, wenn das Register für das erste Summe-von-Produkten-Ergebnis überläuft, eine Inkrementierung oder Dekrementierung eines oberen Registers für das zweite Summe-von-Produkten-Ergebnis in einer dritten Stufe ausgeführt wird.

[0072] Fig. 11A ist ein Blockdiagramm einer Summe-von-Produkten-Rechenschaltung gemäß Ausführungsform 4. Diese Summe-von-Produkten-Rechenschaltung weist einen Multiplizierer **105**, einen Addierer **106-1**, einen Inkrementierer/Dekrementierer **106-2**, ein ALR **107-1**, bei dem es sich um ein unteres Register eines MAC-Registers handelt (Register für das Summe-von-Produkten-Ergebnis) und ein AHR **107-2** auf, bei dem es sich um ein oberes Register des MAC-Registers handelt.

[0073] In der ersten Stufe des in Fig. 11B dargestellten Pipeline-Prozesses multipliziert der Multiplizierer **105** die ersten und zweiten Summe-von-Produkten-Eingangsdaten MDA und MDB. In einer zweiten Stufe des Pipeline-Prozesses addiert der Addierer **106-1** dann das Ergebnis der Multiplikation der ersten Stufe zu Daten, die im ALR-1 (Register für das erste Summe-von-Produkten-Ergebnis) gespeichert sind. Wenn die Addition der zweiten Stufe einen positiven Überlauf verursacht und damit ein Überlaufsignal aktiv wird, inkrementiert der Inkrementierer/Dekrementierer **106-2** in der dritten Stufe des Pipeline-Prozesses die im AHR **107-2** (Register für das zweite Summe-von-Produkten-Ergebnis) gespeicherten Daten. Wenn dagegen die Addition der zweiten Stufe einen negativen Überlauf verursacht und damit ein Zehnervorgriffssignal aktiv wird, dekrementiert der Inkrementierer/Dekrementierer **106-2** in der dritten Stufe die im AHR **107-2** gespeicherten Daten.

[0074] Bei der so konfigurierten Ausführungsform 4 wird das durch die Multiplikation von 16-Bit-Daten mit 16-Bit-Daten erhaltene Ergebnis zum 64-Bit-MAC-Register addiert, das aus dem ALR **107-1** und dem AHR **107-2** gebildet ist. Diese Addition ist unterteilt in eine untere 32-Bit-Addition und eine obere 32-Bit-Addition, wobei die untere 32-Bit-Addition in der zweiten Stufe des Pipeline-Prozesses und die obere 32-Bit-Additon (Inkrementierung oder Dekrementierung) in der dritten Stufe des Pipeline-Prozesses ausgeführt wird. Damit können folgende Auswirkungen erzielt werden:

(1) Der Addierer **106-1** kann anstatt als ein 48-Bit-Bauelement als ein 32-Bit-Bauelemeni ausgeführt sein, so dass die Probleme mit dem kritischen Pfad, die bei einem 48-Bit-Addierer auftreten würden, vermieden werden können.

(2) Die Anzahl der Bits des MAC-Registers (ALR **107-1** und AHR **107-2**) kann auf 64 Bits erhöht werden, wodurch die Wahrscheinlichkeit eines Überlaufs (Sättigung) während der Summe-von-Produkten-Rechnung verringert und außerdem die Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, auf $2^{32} - 1$ erhöht werden kann, womit praktisch unendlich viele Rechnungen ermöglicht werden. Die Ausführungsform 4 ist besonders wirksam, wenn sie mit der Ausführungsform 1 kombiniert wird. Mit anderen Worten, die Technik der Ausführungsform 1, bei der eine die Anzahl der Ausführungen vorgebende Information im Summe-von-Produkten-Rechenbefehl enthalten ist, stellt sicher, dass kein Programm mit einer Sequenz der gleichen Anzahl Summe-von-Produkten-Rechenbefehlen wie der Anzahl Ausführungen geschrieben werden muss, so dass der Benutzer eine extrem hohe Anzahl von Malen vorgeben kann, die die Summe-von-Produkten-Rechnung auszuführen ist. Bei der Ausführungsform 4 ist die Anzahl der Ausführungen der Summe-von-Produkten-Rechnung praktisch unendlich, so dass sie die Vorgabe einer derart hohen Anzahl Ausführungen verarbeiten kann.

(3) Bei der Ausführungsform 4 kann eine obere 32-Bit-Addition unter Verwendung des Inkrementierers/Dekrementierers **106-2** ausgeführt werden, der hardwaremäßig kleiner ist als ein normaler Addierer. Das bedeutet, dass die Größe der Hardware auf ein Minimum reduziert werden kann, obwohl die Größe des MAC-Registers (ALR **107-1** und AHR **107-2**) auf 64 Bits erhöht wird.

[0075] Vorzugsweise ist der Multiplizierer **105** als 17 × 17-Bit-Konfiguration bereitzustellen, um sicherzustellen, dass ein einziges Hardware-Element zur Multiplikation sowohl von Daten mit als auch ohne Vorzeichen verwendet werden kann. Das Überlaufsignal von Fig. 11A wird aktiv, wenn ein Überlauf durch den Addierer **106-1** erzeugt wird und außerdem die Daten zu diesem Zeitpunkt positiv sind. Umgekehrt wird das Zehnervorgriffssignal aktiv, wenn ein Überlauf vom Addierer **106-1** erzeugt wird, aber die Daten zu diesem Zeitpunkt negativ sind. Wenn nur Daten ohne Vorzeichen zu verarbeiten sind, ist das Zehnervorgriffsignal nicht erforderlich, so dass der Inkremenüerer/Dekrementierer **106-2** nur die Inkrementiertunktion zu erfüllen braucht.

Ausführungsform 5

[0076] Die fünfte Ausführungsform dieser Erfindung

betrifft detaillierte Beispiele der Steuerschaltung **102**, der Summe-von-Produkten-Rechenschaltung **104** und der ALU **132** in Fig. 1. Fig. 12 ist ein Blockdiagramm dieser Einheiten.

[0077] In Fig. 12 kennzeichnet I_ADDR_BUS einen Befehlsadressbus und I_DATA_BUS einen Befehlsdatenbus. Diese Busse dienen zum Auslesen von Befehlen wie einem Summe-von-Produkten-Rechenbefehl aus einem Befehlsspeicher **110-1**. D_ADDR_BUS kennzeichnet einen Datenadressbus und A_DATA_BUS einen Datenbus, und diese Busse dienen zum Auslesen von Daten wie den ersten und zweiten Summe-von-Produkten-Eingangsdaten MDA und MDB aus einem Datenspeicher **110-2**. Diese Ausführungsform verwendet also eine Buskonfiguration, die als Harvard-Architektur bezeichnet wird.

[0078] PA_BUS, PB_BUS, WW_BUS und XA_BUS sind interne Busse und AUX_BUS ist ein Bus für den Austausch von Daten zwischen der Steuerschaltung **102** und der Summe-von-Produkten-Rechenschaltung **104**. IA und DA dienen zur Ausgabe von Adressen aus der Steuerschaltung **102** (CPU) an I_ADDR_BUS bzw. D_ADDR_BUS. DIN dient zur Eingabe von Daten von D_DATA_BUS in die Steuerschaltung **102** und DOUT zur Ausgabe von Daten von der Steuerschaltung **102** an D_DATA_BUS.

[0079] Ein Befehlsdecodierer **140** empfängt und analysiert Befehle, die von I_DATA_BUS eingegeben werden, und gibt dann verschiedene Steuersignale aus, die zur Ausführung dieser Befehle erforderlich sind. So werden z. B. verschiedene Anweisungen entsprechend jedem Befehl an die Komponenten der Steuerschaltung **102** über einen Direktwertgenerator **142** gesendet. Wenn ein Interrupt vom Interruptkontroller **130** (siehe Fig. 1) empfangen wird, wird TRAP VECTOR zur Aktivierung des Interrupt-Handlers über den D_ADDR_BUS ausgegeben und ein Signal trap aktiviert (geht auf 1), um die Summe-von-Produkten-Rechenschaltung **104** zu informieren, dass ein Interrupt erzeugt worden ist. In ähnlicher Weise wird bei jedem Empfang eines Summe-von-Produkten-Rechenbefehls ein mac-Signal aktiv, um die Summe-von-Produkten-Rechenschaltung **104** zu informieren, dass ein Summe-von-Produkten-Rechenbefehl erzeugt worden ist.

[0080] Der Direktwertgenerator **142** erzeugt 32-Bit-Direktdaten, die während der Ausführung des Befehls zu verwenden sind, und konstante Daten 0, ±1, ±2 und ±4, die zur Ausführung jedes Befehls auf Basis des im Befehl enthaltenen Direktwertes erforderlich sind. Ein PC-Inkrementierer **118** inkrementiert den Wert im PC **120** bei jeder Ausführung eines Befehls. Ein Adressaddierer **144** führt unter Verwendung der in den verschiedenen Registern gespeicherten Informationen und der vom Direktwertgenerator **142** erzeugten Direktdaten eine Addition aus und erzeugt die zum Auslesen aus dem Speicher **110** erforderlichen Adressen.

[0081] Das Mehrzweckregister **103** weist sechzehn 32-Bit-Register R0 bis R15 auf. Ein SP **146** ist ein spezielles 32-Bit-Stapelzeigerregister, das einen Stapelzeiger speichert, der die Startadresse im Stapel angibt. Ein Prozessorzustandsregister (PSR) **148** ist ein 32-Bit-Register, das verschiedene Flags speichert.

[0082] Die ALU **132** führt arithmetische und logische Operationen aus; bei dieser Ausführungsform dekrementiert sie die Anzahl der Ausführungen. Wenn das Ergebnis der von der ALU **132** ausgeführten Operation null ist, aktiviert ein Null-(Zero)Detektor **134** ein Signal ALU_zero (legt es auf 1). Dies bewirkt, dass im PSR **148** ein Zero-Flag gesetzt wird, und informiert außerdem die Summe-von-Produkten-Rechenschaltung **104**, dass die Anzahl der Ausführungen null erreicht hat. Ein Busmultiplexer **121** wählt einen Bus aus PA_BUS, BP_BUS und WW_BUS und verbindet ihn mit AUX_BUS. Der Busmultiplexer **121** weist das TEMPm-Register **122** und das TEMPn-Register **124** auf und gibt die ersten und zweiten Summe-von-Produkten-Eingangsdaten MDA und MDB an die Summe-von-Produkten-Rechenschaltung **104** aus, wenn beide Daten vorhanden sind.

[0083] Die Summe-von-Produkten-Rechenschaltung **104** weist eine Zustandsmaschine **150** auf. Diese Zustandsmaschine **150** steuert den Zustand der Summe-von-Produkten-Rechenschaltung **104** auf Basis verschiedener Signale wie ALU_zero, trap und mac.

[0084] Die im Taktdiagramm von Fig. 13 dargestellten MAC-Zustände (MAC0 bis MAC8) repräsentieren die Zustände der Summe-von-Produkten-Rechenschaltung **104** (der Zustandsmaschine **150**), und ein Zustandsübergangsdiagramm davon ist in Fig. 14A dargestellt. Die Bedeutungen der Signale in diesem Zustandsübergangsdiagramm sind wie folgt:

(1) mac

[0085] Dieses Signal wird 1 (aktiv), wenn der Befehlsdecodierer **140** einen Summe-von-Produkten-Rechenbefehl empfängt.

(2) mac_end

[0086] Dieses Signal wird 1, wenn die Endbedingung für einen Summe-von-Produkten-Rechenbefehl angehoben wird; genauer gesagt, geht es auf 1, wenn mac_zero oder mac_trap 1 wird.

(3) mac_zero

[0087] Dieses Signal wird 1, wenn die Anzahl von

Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, null erreicht. In diesem Fall wird mac_zero 0, wenn der Mikrocomputer rückgesetzt wird oder wenn der MAC-Zustand MAC8 oder MAC9 wird, wie in Fig. 14B dargestellt ist. Es wird außerdem 1, wenn das Signal ALU_zero vom Nulldetektor **134** 1 wird, wenn der MAC-Zustand MAC3, MAC5 oder MAC7 ist.

(4) mac_trap

[0088] Dieses Signal wird 1, wenn ein Interrupt während der Ausführung des Summe-von-Produkten-Rechenbefehls auftritt. In diesem Fall wird mac_trap 0, wenn der Mikrocomputer rückgesetzt wird oder wenn der MAC-Zustand MAC8 oder MAC9 wird, wie in Fig. 14B dargestellt ist. Es wird außerdem 1, wenn das Signal trap vom Befehlsdecodierer **140** 1 wird, wenn der MAC-Zustand MAC5 oder MAC7 ist.

[0089] Der MAC-Zustand bleibt auf MAC0, wenn mac 0 wird, aber kein Summe-von-Produkten-Rechenbefehl ausgegeben wird, wie in Fig. 14A dargestellt ist. Andererseits ändert sich der MAC-Zustand zu MAC1, wenn mac 1 wird. Die Übergänge von MAC1 to MAC2 und von MAC2 zu MAC3 sind unbedingt (UCT) und synchron mit dem Taktsignal.

[0090] MAC3 ändert sich zu MAC9, wenn mac_end 1 wird, wodurch außerdem mac_end auf 0 rückgesetzt wird (siehe Fig. 14B). Nach dem Übergang zu MAC9 kehrt der Zustand zu MAC1 zurück, wenn mac auf 1 liegt, oder zu MAC0, wenn mac auf 0 liegt. Wenn mac_end 0 ist, ändert sich MAC3 zu MAC4.

[0091] Der Übergang von MAC4 auf MAC5 ist unbedingt und synchron mit dem Taktsignal. Während dieser Zeit ist es möglich, dass mac_zero aufgrund der Dekrementierung der Anzahl von Ausführungen 1 wird (siehe E22 in Fig. 13). Bei MAC5 erfolgt eine Bestimmung, ob mac_end 1 ist oder nicht; wenn es 1 ist, ändert sich der Zustand zu MAC8 und geht dann von MAC8 auf MAC0 oder MAC1 zurück. Wenn mac_end dagegen 0 ist, ändert sich der Zustand von MAC5 zu MAC6.

[0092] Der Übergang von MAC6 zu MAC7 ist bedingungslos und synchron mit dem Taktsignal. Während dieser Zeit ist es möglich, dass mac_zero aufgrund der Dekrementierung der Anzahl von Ausführungen 1 wird (siehe E24 und E26 in Fig. 13). Bei MAC7 erfolgt eine Bestimmung, ob mac_end 1 ist oder nicht; wenn es 1 ist, ändert sich der Zustand zu MAC8; wenn es 0 ist, geht er zu MAC6 zurück.

[0093] Wenn die Anzahl der Male, die die Summe-von-Produkten-Rechnung auszuführen ist, z. B. auf 0 eingestellt worden ist, ändert sich der MAC-Zustand zunächst in der Folge: MAC0, MAC1, MAC2, dann MAC3. Da mac_end dann 1 wird (mac_zero wird 1), ändert sich der Zustand von MAC3 zu MAC9 und dann zu MAC0 (oder MAC1).

[0094] Wenn die Anzahl der Ausführungen auf 1 eingestellt worden ist, ändert sich der MAC-Zustand zunächst in der Folge: MAC0, MAC1, MAC2, MAC3, dann MAC4. Die Anzahl der Ausführungen wird beim Übergang von MAC4 zu MAC5 dekrementiert, so dass mac_end 1 wird. Als Ergebnis ändert sich der MAC-Zustand von MAC4 to MAC5, MAC8, dann MAC0 (oder MAC1).

[0095] Wenn die Anzahl der Ausführungen auf 2 eingestellt worden ist, ändert sich der MAC-Zustand in der Folge: MAC0, MAC1, MAC2, MAC3, MAC4, MAC5, MAC6, MAC7, MAC8, dann MAC0 (oder MAC1). Mit anderen Worten, die Anzahl der Ausführungen wird zwischen MAC4 und MAC5 und zwischen MAC6 und MAC7 dekrementiert, so dass sie null wird. Es ist zu beachten, dass dann, wenn die Anzahl der Ausführungen auf 3 oder mehr eingestellt worden ist, die Operation der Änderung von MAC6 zu MAC7 und zurück zu MAC6 so oft wiederholt wird, bis die Anzahl der Ausführungen null wird.

[0096] Wenn eine Interrupt-Anforderung aufgetreten ist, erfolgt eine Bestimmung, ob in dem Punkt, in dem der Zustand sich zu MAC5 oder MAC7 geändert hat, mac_trap 1 geworden ist oder nicht (mac_end ist 1 geworden), und der Zustand ändert sich zu MAC8.

[0097] Ein charakteristisches Merkmal der Zustandsmaschine **150** dieser Ausführungsform ist, dass der MAC-Zustand auf Basis des aktiv werdenden Signals mac_end zum Ausgangszustand MAC0 (oder MAC1) zurückkehrt, wenn die Summe-von-Produkten-Rechnung die gewünschte Anzahl von Malen ausgeführt worden ist (mac_zero wird 1) oder wenn eine Interrupt-Anforderung aufgetreten ist (mac_trap wird 1). Diese Konfiguration ermöglicht es, Zustandsübergänge, die verwendet werden, wenn die Summe-von-Produkten-Rechnung die gewünschte Anzahl von Malen ausgeführt worden ist, als Zustandsübergänge bei Auftreten einer Interrupt-Anforderung zu nutzen. Dadurch kann die Konfiguration der Zustandsmaschine **150** vereinfacht werden.

[0098] Die Funktionsweise dieser Ausführungsform wird nunmehr anhand von Fig. 13 beschrieben. Fig. 13 ist ein Taktdiagramm, das den Fall zeigt, in dem die Anzahl der Male, die die Summe-von-Produkten-Rechnung auszuführen ist, auf 3 eingestellt worden ist. Das bedeutet, dass sich in diesem Fall der MAC-Zustand von MAC0 zu MAC1, MAC2, MAC3, MAC4, MAC5, MAC6, MAC7, MAC8, dann MAC0 ändert. Bei dieser Ausführungsform fungiert das Mehrzweckregister R13 als ein Register für die Anzahl der Ausführungen, und 3 ist darin als die Anzahl der Ausführungen eingestellt (siehe E0 in

Fig. 13). R14 und R15 dienen als Register für die ersten und zweiten Summe-von-Produkten-Eingangsdaten MDA und MDB, und die Startadressen **110h** und **230h** für die MDA und MDB enthaltenden Speicherbereiche sind darin gespeichert (siehe E1 und E2).

[0099] Wenn der Befehlsdecodierer **140** von Fig. 12 einen Summe-von-Produkten-Rechenbefehl empfängt, wird mac 1 und der MAC-Zustand ändert sich von MAC0 zu MAC1.

[0100] Die in R13 gespeicherte Anzahl von Ausführungen wird dann über den PB_BUS an die ALU **132** ausgegeben (E3). Die ALU **132** addiert null zu dieser Anzahl von Ausführungen (E4). Mit dieser Addition von null wird geprüft, ob die Anzahl der Ausführungen ursprünglich auf null eingestellt worden war oder nicht. Wenn sie null ist, wird ALU_zero 1 und der Summe-von-Produkten-Rechenbefehl endet (siehe MAC3 und MAC9 in Fig. 14A).

[0101] Die in R14 gespeicherte Adresse **100h** wird dann über den XA_BUS an den D_ADDR_BUS ausgegeben (E5 und E6). Die ersten Summe-von-Produkten-Eingangsdaten MDA (**110h**) werden auf Basis dieser Adresse aus dem Speicher **110** ausgelesen (E7). In ähnlicher Weise wird die in R15 gespeicherte Adresse **230h** über den XA_BUS an den D-ADDR_BUS ausgegeben (E8 und E9) und die zweiten Summe-von-Produkten-Eingangsdaten MDB (**230h**) werden auf Basis dieser Adresse aus dem Speicher **110** ausgelesen (E10). Der Multiplizierer **105** multipliziert diese Werte MDA und MDB (E11), der Addierer **106-1** addiert das Multiplikationsergebnis (E12), und das Additionsergebnis wird in ALR **107-1** gespeichert (E13). Wenn durch diese Addition ein Überlauf oder Zehnervorgriff erzeugt wird, führt der Inkrementierer/Dekrementierer **106-2** eine Inkrementierung oder Dekrementierung aus (E14) und speichert das Ergebnis in AHR **107-2** (E15).

[0102] Die in R14 und R15 gespeicherten Adressen **110h** und **230h** werden außerdem über den XA_BUS an den Adressaddierer **144** ausgegeben (E5 und E8). Der Adressaddierer **144** addiert 2 zu jeder dieser Adressen (E16 und E17) und gibt die Ergebnisse dieser Additionen über den WW_BUS an die Register R14 und R15 zurück (E18 und E19). Dies ändert die in R14 und R15 gespeicherten Adressen zu **112h** und **232h** (E20 und E21), wodurch das Lesen der nächsten Summe-von-Produkten-Eingangsdaten MDA (**112h**) und MDB (**232h**) möglich wird.

[0103] Die ALU **132** dekrementiert der Anzahl der Ausführungen bei MAC4 von 3 auf 2 (E22). Somit wird die dekrementierte Anzahl der Ausführungen an den PB_BUS ausgegeben, dann vom PB_BUS zurückgegeben und in die ALU **132** eingegeben (E23). Die ALU **132** dekrementiert dann die Anzahl der Ausführungen von 2 auf 1 (E24). Die so dekrementierte Anzahl der Ausführungen wird zurückgegeben und in die ALU **132** eingegeben (E25). Die ALU **132** dekrementiert dann die Anzahl der Ausführungen von 1 auf 0 (E26). ALU_zero wird 1, da die Anzahl der Ausführungen 0 erreicht hat (E27). Der MAC-Zustand ändert sich von MAC6 zu MAC7, MAC8, dann MAC0 (E28), wodurch die Ausführung des Summe-von-Produkten-Rechenbefehls beendet wird. Zu diesem Zeitpunkt wird die Anzahl der Ausführungen, die auf 0 dekrementiert worden ist, über den WW_BUS in R13 gespeichert (E29 und E30).

[0104] Die Funktionsweise dieser Ausführungsform bei Auftreten eines Interrupt wird nunmehr anhand des Taktdiagramms von Fig. 15 beschrieben. Bei der in Fig. 15 beispielhaft dargestellten Situation tritt ein Interrupt auf, und das Signal trap wird 1, wenn der MAC-Zustand MAC3 ist (E1 in Fig. 15). In diesem Fall führt diese Ausführungsform den gleichen Prozess aus wie den, wenn im nächsten Zustand, der MAC4 ist, kein Interrupt auftritt. Wenn sich der MAC-Zustand zu MAC5 ändert, erfolgt zunächst die Verarbeitung zur Änderung von MAC5 zu MAC8, dann MAC0 (F2).

[0105] Mit anderen Worten, diese Ausführungsform bringt den MAC-Zustand in den ursprünglichen Zustand MAC0 (oder MAC1) zurück, nachdem die Inhalte der Register R14 und R15 zu Inhalten geändert worden sind, die verwendet werden, wenn die Ausführung nach dem Ende der Interrupt-Verarbeitung wieder aufgenommen wird (F3, F4, F5 und F6). Dies stellt sicher, dass die Ausführung der Summe-von-Produkten-Rechnungen nach dem Ende der Interrupt-Verarbeitung auf Basis der Summe-von-Produkten-Eingangsdaten MDA und MDB in den Adressen **112h** und **232h** auf geeignete Weise wieder aufgenommen werden kann.

[0106] Bei dieser Ausführungsform geht der MAC-Zustand nach der Dekrementierung der Anzahl von Malen, die die Summe-von-Produkten-Rechnung auszuführen ist, in den ursprünglichen Zustand zurück (F7). Deshalb wird die Anzahl der Ausführungen, die nach der Dekrementierung 2 beträgt, in R13 gespeichert (F8 und F9), wodurch die Wiederaufnahme der Ausführung der verbliebenen Summe-von-Produkten-Rechnungen nach Beendigung der Interrupt-Verarbeitung möglich ist.

[0107] Diese Verarbeitung ermöglicht die Implementierung der verschiedenen Prozesse, die in den Abschnitten über die Ausführungsform 1, 2 und 4 beschrieben worden sind. Zur Implementierung der Verarbeitung gemäß Ausführungsform 3 sei darauf hingewiesen, dass das Auslesen der Summe-von-Produkten-Eingangsdaten MDA und MDB aus dem Speicher in einem einzigen Speicherzugriff (1 Takt) erfolgen könnte.

Ausführungsform 6

[0108] Die sechste Ausführungsform betrifft einen Mikrocomputer, bei dem diese Erfindung angewendet wird.

[0109] Wie aus Fig. 16 ersichtlich ist, handelt es sich bei einem Mikrocomputer **700** gemäß der Ausführungsform 6 um einen 32-Bit-Mikrocomputer, der aufweist: eine CPU (Steuerschaltung, Summe-von-Produkten-Rechenschaltung und eine ALU) **710**, einen ROM **720**, einen RAM **730**, eine Hochfrequenz-Schwingungsschaltung **910**, eine Niederfrequenz-Schwingungsschaltung **920**, eine Reset-Schaltung **930**, einen Vorskalierer **940**, eine Zeitgeberschaltung mit einem programmierbaren 16-Bit-Zeitgeber **950**, einem programmierbaren 8-Bit-Zeitgeber **960** und einem Taktgeber **970**, eine Datenübertragungssteuerschaltung mit einem intelligenten Direktspeicherzugriff-(direct memory access; DMA)Kontroller **980** und einem hochschnellen DMA-Kontroller **990**, einen Interrupt-Kontroller **800**, eine serielle Schnittstelle **810**, eine BCU **740**, eine analoge Schnittstellenschaltung mit einem A/D-Wandler **830** und einem D/A-Wandler **840**, eine E/A-Schaltung mit einem Eingangsanschluss **850**, einem Ausgangsanschluss **860** und einem E/A-Anschluss **870**, Busse **750** und **760**, die diese Komponenten miteinander verbinden, und Stifte **890**.

[0110] Bei diesem Mikrocomputer **700** handelt es sich um einen Computer mit verringertem Befehlsvorrat (reduced instruction set computer; RISC), der auf einem Ein-Chip-Halbleitersubstrat ausgebildet ist und 32-Bit-Daten verarbeiten kann. Er verwendet eine Architektur mit Pipeline- und Lade-/Speicherverfahren und führt im Wesentlichen sämtliche seiner Befehle innerhalb einer einzigen Taktperiode aus. Alle Befehle sind innerhalb einer festen 16-Bit-Länge definiert, was eine extrem kleine Größe des Befehlscodes zulässt.

[0111] Wie oben im Hinblick auf die Ausführungsformen 1 bis 5 beschrieben ist die CPU **710** so ausgelegt, dass eine Mehrzahl Summe-von-Produkten-Rechnungen durch einen einzigen Summe-von-Produkten-Rechenbefehl ausführen kann. Das bedeutet, dass der Mikrocomputer **700** zu solcher Verarbeitung eingesetzt werden kann, die nur durch einen DSP (Digitalsignalprozessor) oder eine spezifische Bild- oder Tonverarbeitungs-IC ausgeführt werden könnte, was eine Kostensenkung und eine Verringerung der Größe einer elektronischen Anlage ermöglicht, in der dieser Mikrocomputer **700** installiert ist.

Ausführungsform 7

[0112] Die siebte Ausführungsform dieser Erfindung betrifft eine elektronische Anlage, die einen der oben in Zusammenhang mit den Ausführungsformen 1 bis 6 beschriebenen Mikrocomputer aufweist.

[0113] Fig. 17A zeigt ein internes Blockdiagramm eines Pkw-Navigationssystems, das ein Typ einer elektronischen Anlage ist, und Fig. 18A ist eine externe Ansicht desselben. Eine Fernbedienung **510** dient zur Betätigung dieses Pkw-Navigationssystems, und ein Positionsdetektionsabschnitt **520** detektiert die Position des Fahrzeugs auf Basis von Informationen eines weltweiten Navigationssystems (global positioning system; GPS) oder Gyroskops. Informationen wie eine Landkarte sind auf einer CD-ROM **530** (Informationsspeichermedium) gespeichert. Ein Bildspeicher **540** fungiert als Arbeitsspeicher während der Bildverarbeitung und von diesem erzeugte Bilder werden für den Fahrer auf einem Bildausgabeabschnitt **550** ausgegeben. Ein Mikrocomputer **500** empfängt Daten von Dateneingabequellen wie der Fernbedienung **510**, dem Positionsdetektionsabschnitt **520** und der CD-ROM **530**, führt verschiedene Prozesse aus und verwendet ein Ausgabegerät wie den Bildausgabeabschnitt **550** zur Ausgabe der so verarbeiteten Daten.

[0114] Bei Pkw-Navigationssystemen wird bis jetzt die Bildverarbeitung (Grafikverarbeitung) von DSPs oder spezifischen Bildverarbeitungs-ICs ausgeführt. Das bedeutet, dass in solchen elektronischen Anlagen zwei Prozessoren vorgesehen werden müssen, wie z. B. ein Rechner mit komplexem Befehlsvorrat (complex instruction set computer; CISC) und ein DSP, was das System kompliziert macht. Die Verwendung der in den Abschnitten über die Ausführungsformen 1 bis 6 beschriebenen Mikrocomputer ermöglicht die effiziente Ausführung einer Mehrzahl Summe-von-Produkten-Rechnungen, wodurch die für ein Pkw-Navigationssystem erforderliche Bildverarbeitung implementiert werden kann, ohne dass ein DSP vorgesehen werden muss.

[0115] Ein internes Blockdiagramm einer Spielmaschine, bei der es sich um einen anderen Typ einer elektronischen Anlage handelt, ist in Fig. 17B und die externe Ansicht desselben in Fig. 18B dargestellt. Diese Spielmaschine verwendet einen Bildspeicher **590** als Arbeitsspeicher zum Erzeugen von Spielebildern und -tönen auf Basis von Daten wie den Bedienungsinformationen des Spielers von einem Spielkontroller **560**, einem Spielprogramm auf einer CD-ROM **570** und Spielerinformationen auf einer IC-Karte **580** und verwendet für ihre Ausgabe einen Bildausgabeabschnitt **610** und einen Tonausgabeabschnitt **600**. Dieser Mikrocomputer **500** dieses Beispiels verwendet die Summe-von-Produkten-Rechenfunktionen, die für die Ausführungsformen 1 bis 6 beschrieben worden sind, um eine dreidimensionale Bildverarbeitung wie Koordinatentransformation, perspektivische Transformation und Abschneiden sowie eine Tonverarbeitung wie Tonkomprimierung und

-expansion auszuführen.

[0116] Ein internes Blockdiagramm eines Druckers, bei dem es sich um einen weiteren Typ einer elektronischen Anlage handelt, ist in Fig. 17C und die externe Ansicht desselben in Fig. 18C dargestellt. Dieser Drucker verwendet einen Bitmusterspeicher **650** als Arbeitsspeicher zum Erzeugen eines Druckbildes auf Basis von Bedienungsinformationen von einem Bedienungsfeld **620** und Schriftzeicheninformationen von einem Codespeicher **630** sowie einem Schriftartenspeicher **640** und verwendet einen Druckausgabeabschnitt **660** zu dessen Ausgabe. Dieser Drucker hat außerdem ein Anzeigefeld **670**, um den Benutzer über seinen Zustand und Modus zu informieren. Der Mikrocomputer **500** dieses Beispiels verwendet die Summe-von-Produkten-Rechenfunktionen, die für die Ausführungsformen 1 bis 6 beschrieben worden sind, um Gerade, Kreisbögen und maßstäbliche Bilder zu zeichnen.

[0117] Es sei darauf hingewiesen, dass der Mikrocomputer gemäß dieser Erfindung außer bei den oben beschriebenen bei verschiedenen anderen Einheiten einer elektronischen Anlage, etwa ein Zelltelefon, ein Personal Handyphone System (PSH; Mobilfunktelefonsystem mit geringer Reichweite), Audiogeräte, elektronische Notebooks, elektronische Rechenmaschinen, Endgeräte für Kassensysteme (point of sales, POS), Geräte mit Sensorbildschirmen, Projektoren, Textverarbeitungssysteme, Personal Computer, Fernsehgeräte und Videorecorder entweder mit Bildsuchern oder Monitoren angewendet werden kann.

[0118] Außerdem ist zu beachten, dass diese Erfindung nicht auf die oben beschriebenen Ausführungsformen 1 bis 7 beschränkt ist; sie kann auf vielerlei verschiedene Arten innerhalb des hierin definierten Gültigkeitsbereichs der Erfindung modifiziert werden.

[0119] So ist z. B. die Technik zur Vorgabe der Anzahl von Malen, die die Summe-von-Produkten-Rechnung durch einen Summe-von-Produkten-Rechenbefehl auszuführen ist, nicht auf die oben beschriebenen Ausführungsformen beschränkt und kann auf verschiedene Arten modifiziert werden.

[0120] In ähnlicher Weise ist die Informationsverarbeitungsschaltung dieser Erfindung besonders wirksam, wenn sie in einem Mikrocomputer, vor allem einem RISC-Mikrocomputer verwendet wird; sie kann aber auch in anderen Anwendungen eingesetzt werden.

[0121] Ferner ist die Konfiguration des Summe-von-Produkten-Rechenbefehls nicht auf die oben beschriebenen Ausführungsformen beschränkt, sondern kann ebenfalls auf verschiedene Weise modifiziert werden.

Patentansprüche

1. Informationsverarbeitungsschaltung, umfassend:
eine Steuerschaltung (**102**) zum Empfang von Befehlen, die einen Summe-von-Produkten-Rechenbefehl umfassen, zur Analyse der Befehle und zur Steuerung der Ausführung der Befehle; und
eine Summe-von-Produkten-Rechenschaltung (**104**) zur Ausführung einer Summe-von-Produkten-Berechnung unter der Steuerung durch die Steuerschaltung (**102**) auf der Grundlage des Summen-von-Produkten-Rechenbefehls;
wobei die Summe-von-Produkten-Rechenschaltung (**104**) eine Summe-von-Produkten-Rechnung eine Anzahl von Malen ausführt, die durch Anzahl-von-Ausführungen-Information spezifiziert ist, welche in dem Summe-von-Produkten-Rechenbefehl enthalten ist;
dadurch gekennzeichnet, daß der Summe-von-Produkten-Rechenbefehl einen Operationscode und einen Operaden zur Spezifizierung eines Registers von einem ersten, einem zweiten und einem dritten einer Mehrzahl von Register umfaßt, von denen das erste Register ein Register (Rc) für die Anzahl von Malen ist, mit der die Summe-von-Produkten-Rechnung ausgeführt werden soll, das zweite Register ein Register (Rm) für erste Summe-von-Produkten-Eingangsdaten ist und das dritte Register ein Register (Rn) für zweite Summen-von-Produkten-Eingabedaten ist; und
die Steuerschaltung (**102**) nach Maßgabe einer gegebenen Regel auf der Basis des Operanden, der das eine Register spezifiziert, ein anderes als das eine unter dem ersten Register, dem zweiten Register und dem dritten Register spezifiziert.

2. Informationsverarbeitungsschaltung nach Anspruch 1, ferner umfassend:
eine Schaltung (**132**) zum Dekrementieren der Anzahl von Malen, mit der die Summe-von-Produkten-Rechnung ausgeführt werden soll, synchron mit der Ausführung einer Summe-von-Produkten-Rechnung, wobei die Anzahl in einem Register gespeichert ist, welches in der Steuerschaltung (**102**) enthalten ist;
wobei die Summe-von-Produkten-Rechenschaltung (**104**) Summe-von-Produkten-Rechnungen ausführt, bis die Anzahl der Ausführungen einen gegebenen Wert erreicht.

3. Mikrocomputer, der auf einem Halbleitersubstrat integriert ist, umfassend:
die Informationsverarbeitungsschaltung nach Anspruch 1 oder 2; und
wenigstens eine Komponente aus der Gruppe aus einer Bussteuerschaltung (**102**), einem Speicher (**110**), einem Interruptkontroller (**130**), einer Zeitgeberschaltung, einer analogen Interfaceschaltung, einer Datenübertragungssteuerschaltung (**102**), und einer

Eingabe/Ausgabeschaltung.

4. Elektronische Anlage, umfassend:
den Mikrocomputer nach Anspruch 3;
eine Eingabequelle für von dem Mikrocomputer zu verarbeitende Daten; und
eine Ausgangseinrichtung zur Ausgabe von Daten, die von dem Mikrocomputer bearbeitet wurden.

Es folgen 19 Blatt Zeichnungen

Anhängende Zeichnungen

FIG. 1

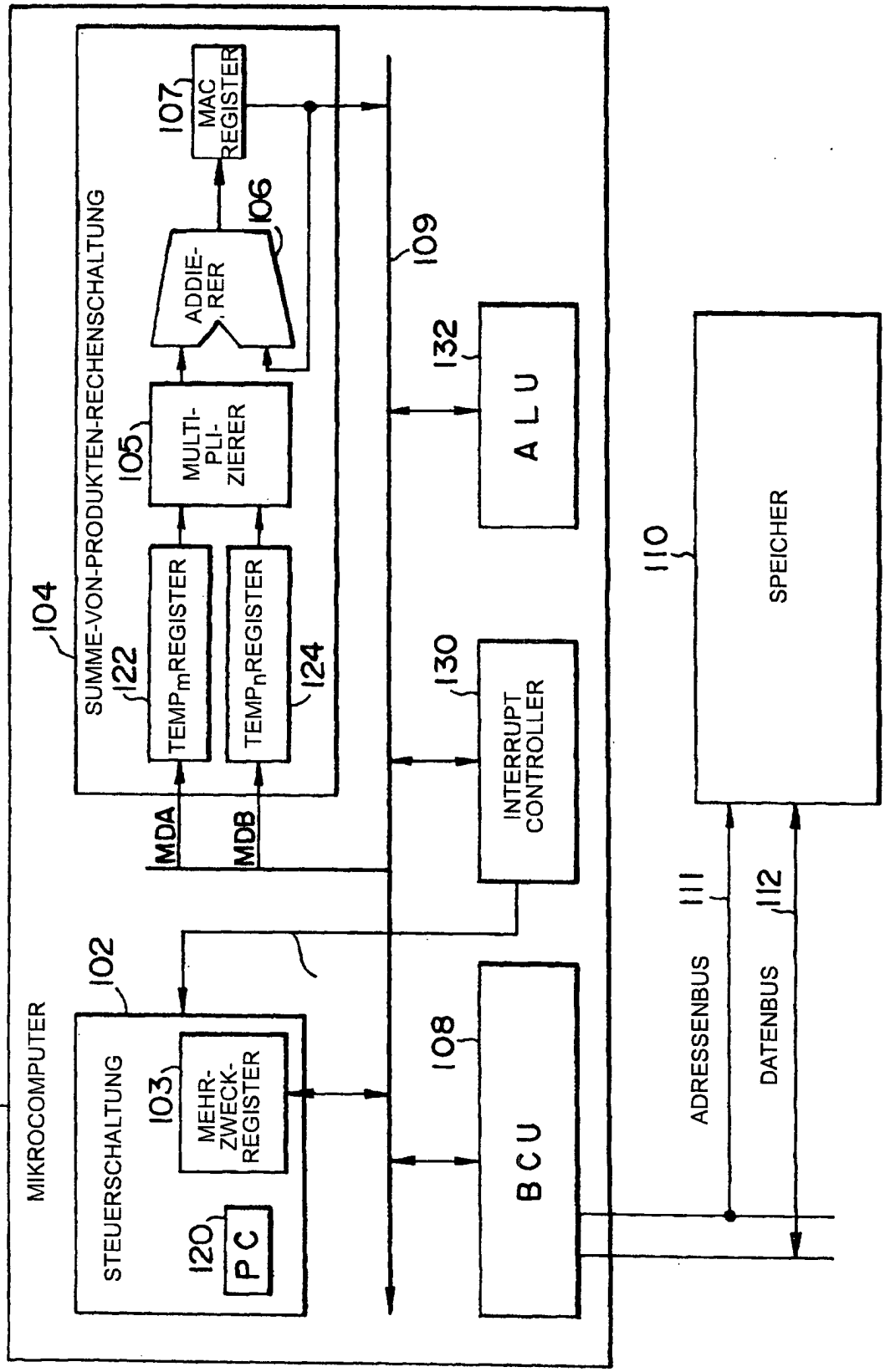


FIG.2

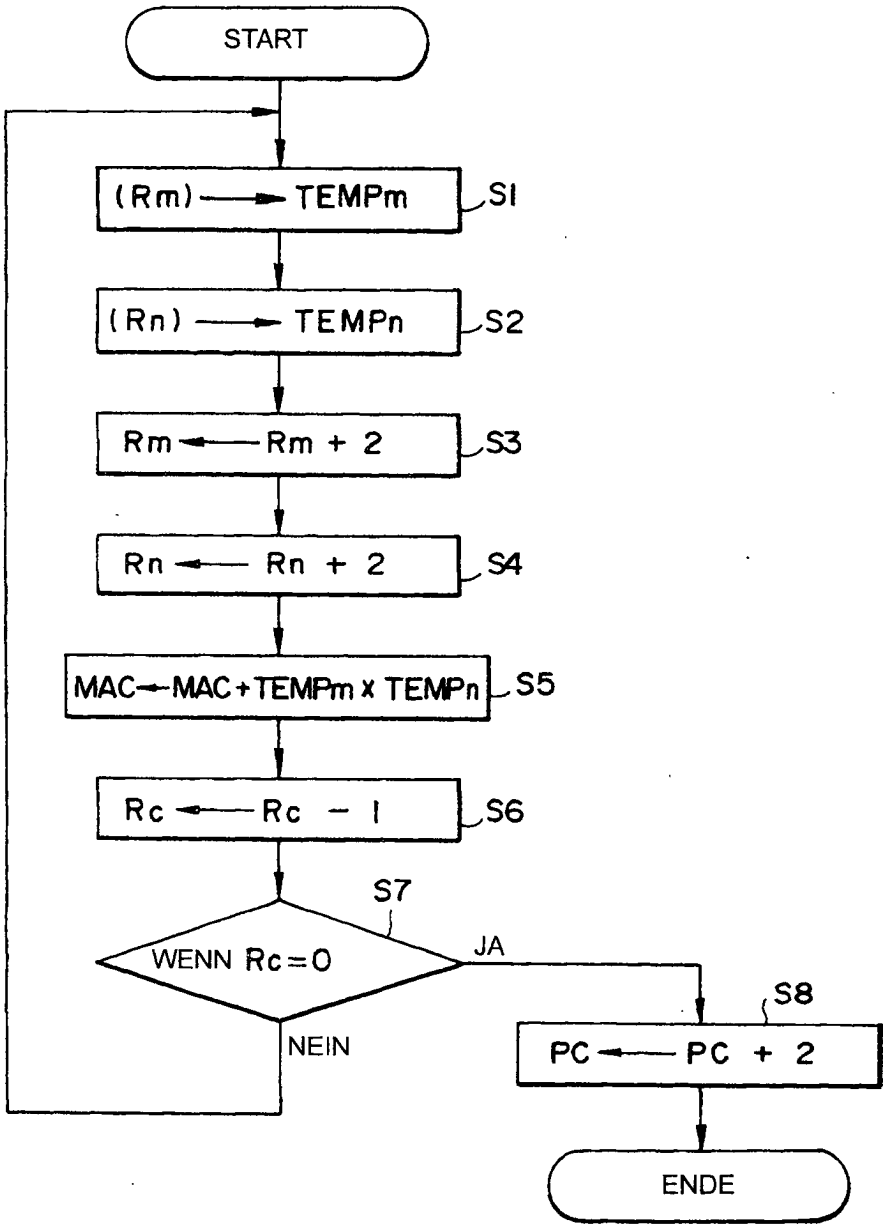

START
(Rm) → TEMPm
S1
(Rn) → TEMPn
S2
Rm ← Rm + 2
S3
Rn ← Rn + 2
S4
MAC ← MAC + TEMPm x TEMPn
S5
Rc ← Rc − 1
S6
S7
WENN Rc = 0
JA
NEIN
S8
PC ← PC + 2
ENDE

FIG.3A

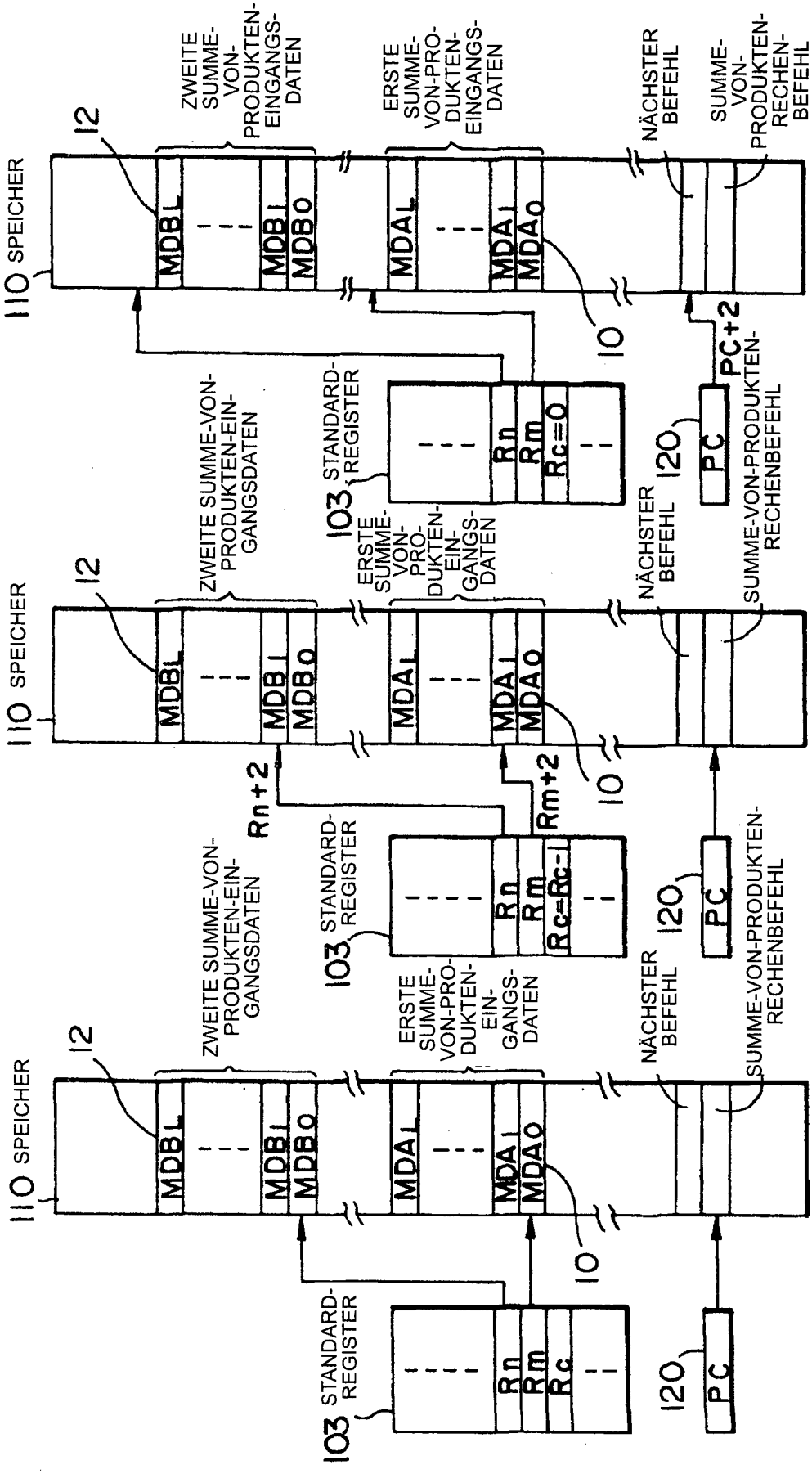


FIG.3B

FIG.3C

FIG. 4

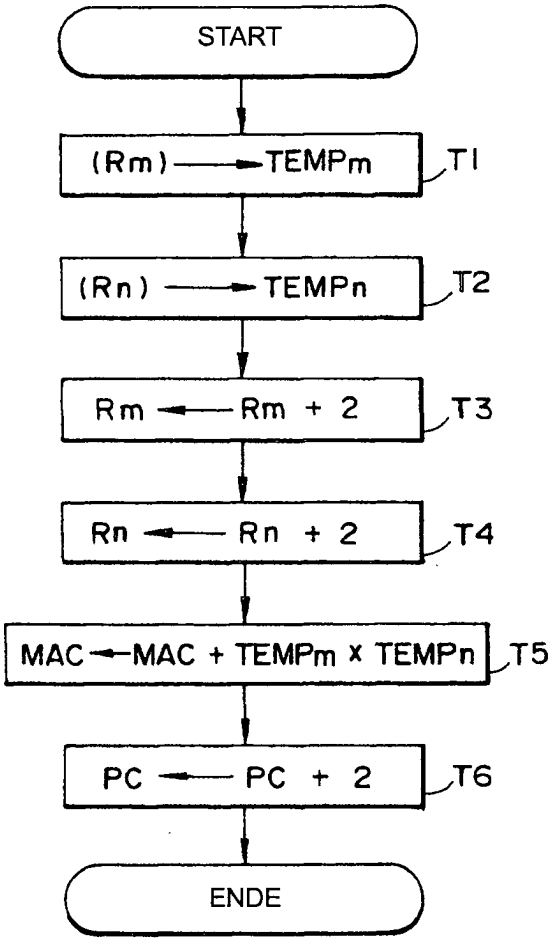

START
(Rm) → TEMPm
T1
(Rn) → TEMPn
T2
Rm ← Rm + 2
T3
Rn ← Rn + 2
T4
MAC ← MAC + TEMPm x TEMPn
T5
PC ← PC + 2
T6
ENDE

FIG. 5A

```
MAC    %Rs1 ,  %Rs2
MAC    %Rs1 ,  %Rs2
MAC    %Rs1 ,  %Rs2
MAC    %Rs1 ,  %Rs2
```

FIG. 5B

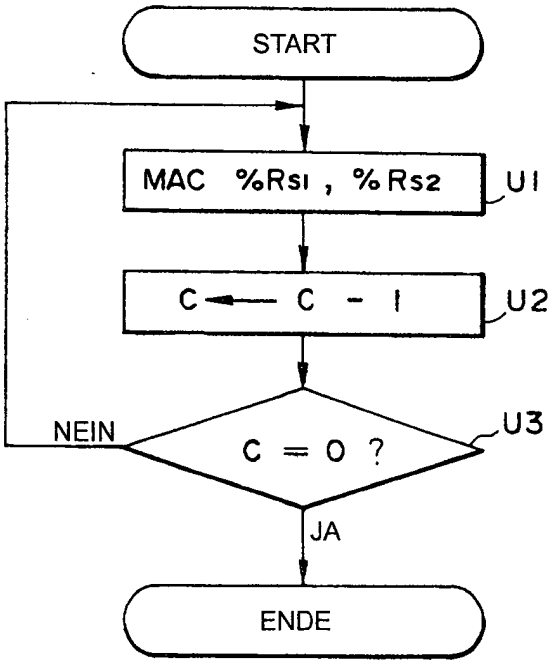


FIG.6A

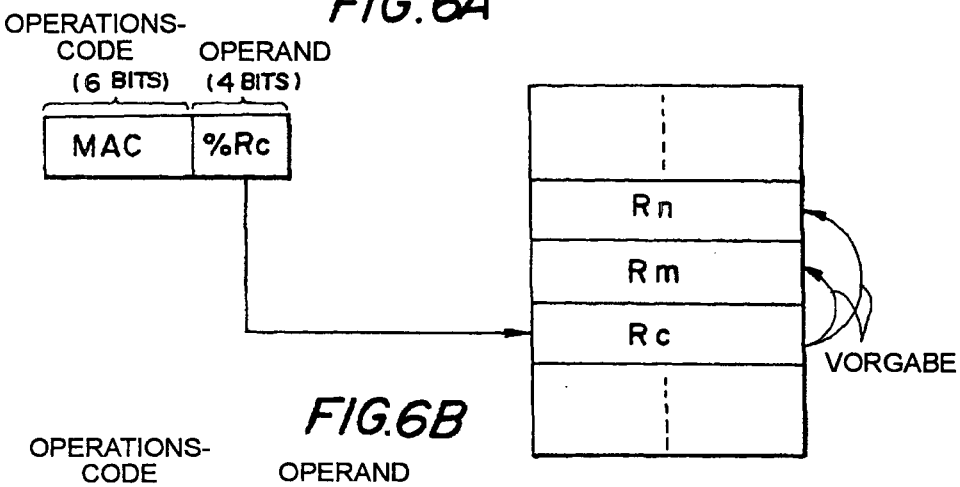


FIG.6B

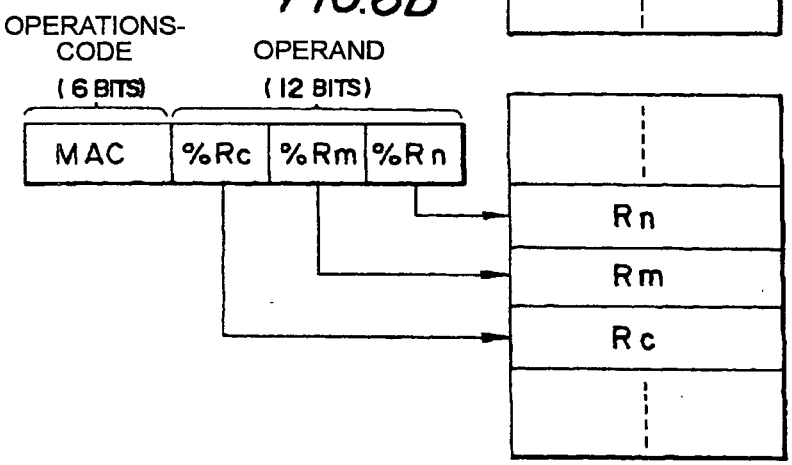


FIG.6C

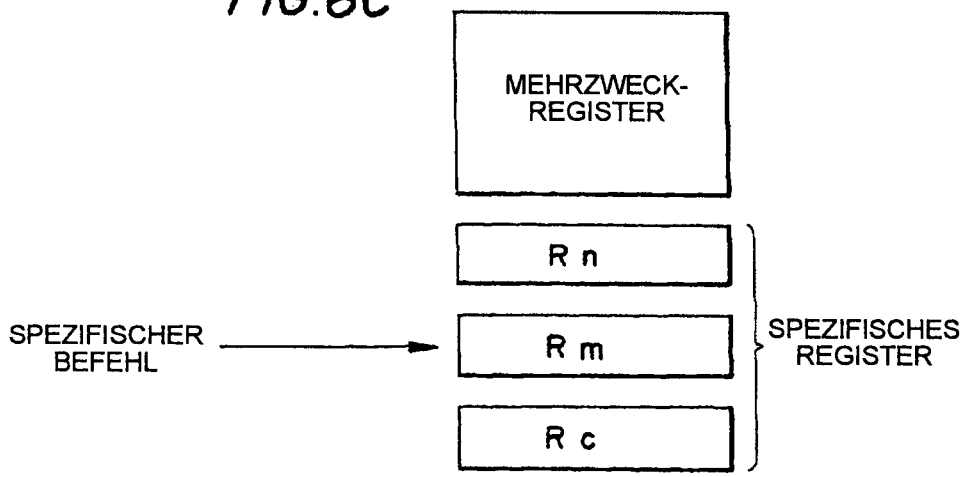
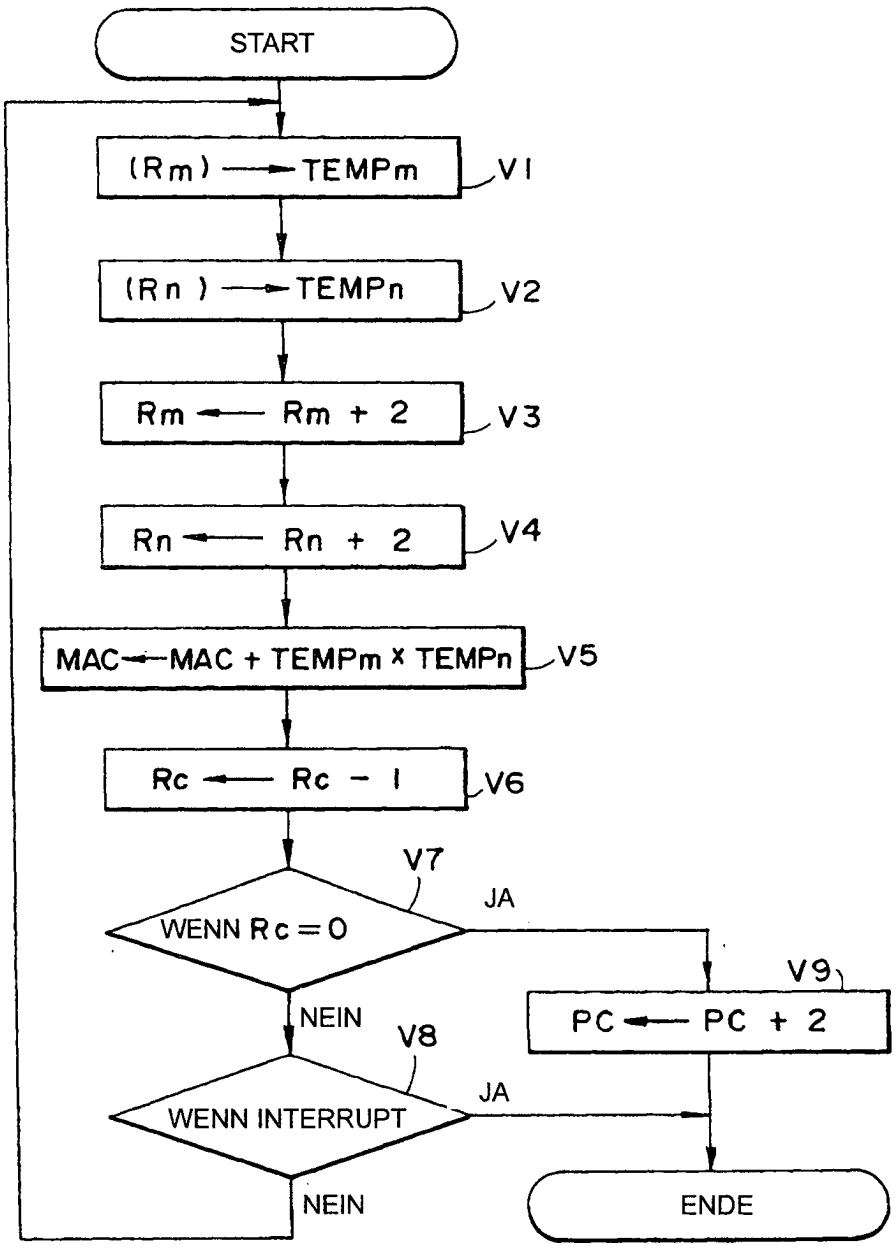


FIG. 7


START
(Rm) → TEMPm
V1
(Rn) → TEMPn
V2
Rm ← Rm + 2
V3
Rn ← Rn + 2
V4
MAC ← MAC + TEMPm x TEMPn
V5
Rc ← Rc - 1
V6
V7
WENN Rc = 0
JA
NEIN
V8
WENN INTERRUPT
JA
NEIN
V9
PC ← PC + 2
ENDE

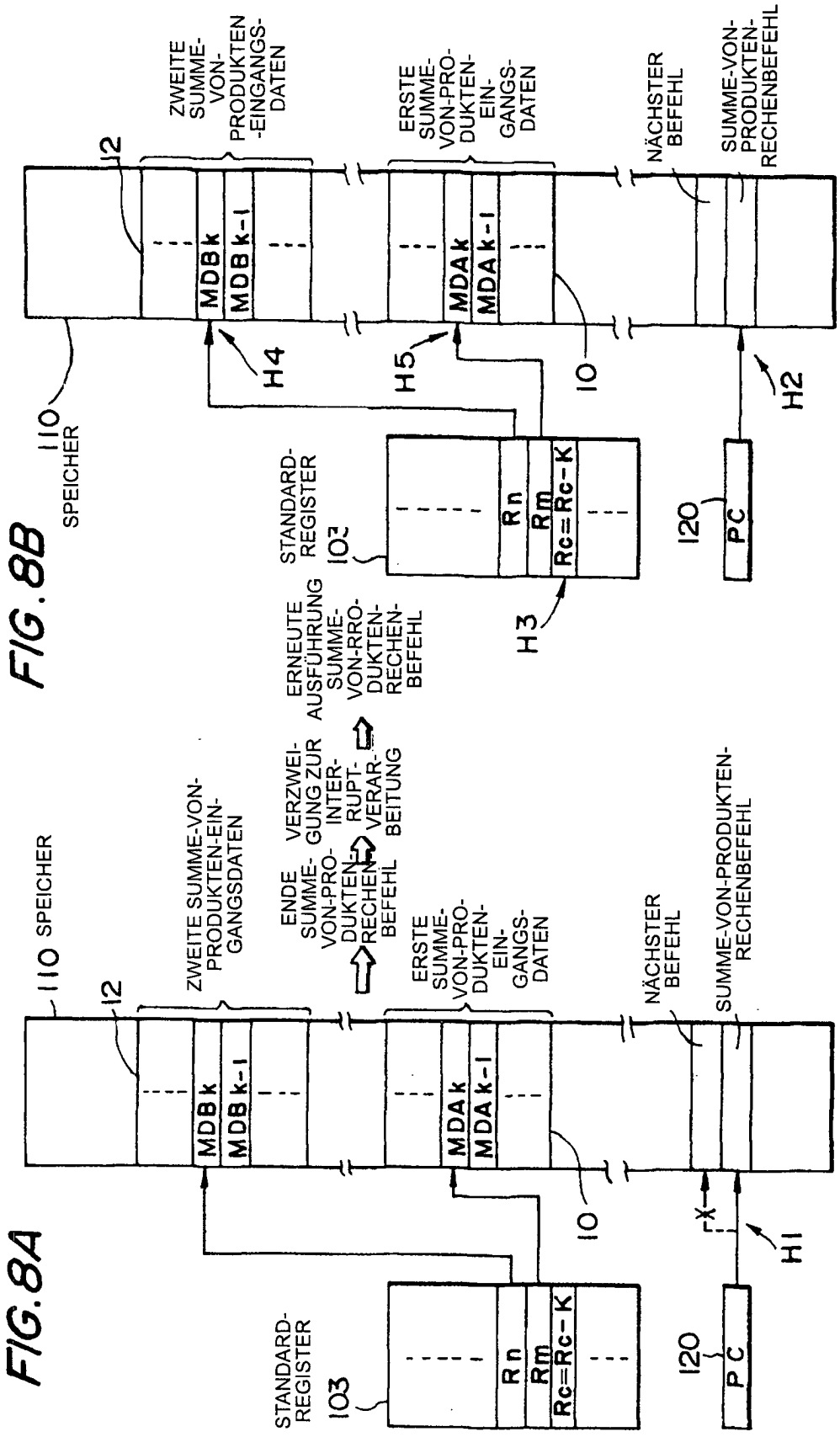


FIG. 8A
110 SPEICHER
12
ZWEITE SUMME-VON-PRODUKTEN-EINGANGSDATEN
MDB k
MDB k-1
ENDE SUMME-VON-PRODUKTEN-RECHEN-BEFEHL
VERZWEIGUNG ZUR INTERRUPT-VERARBEITUNG
ERSTE SUMME-VON-PRODUKTEN-EINGANGS-DATEN
MDA k
MDA k-1
NÄCHSTER BEFEHL
SUMME-VON-PRODUKTEN-RECHENBEFEHL
10
X
H1
STANDARD-REGISTER
103
Rn
Rm
Rc=Rc-K
120
PC
FIG. 8B
110 SPEICHER
12
ZWEITE SUMME-VON-PRODUKTEN-EINGANGS-DATEN
MDB k
MDB k-1
ERSTE SUMME-VON-PRODUKTEN-EINGANGS-DATEN
MDA k
MDA k-1
NÄCHSTER BEFEHL
SUMME-VON-PRODUKTEN-RECHENBEFEHL
10
H4
H5
H2
STANDARD-REGISTER
102
Rn
Rm
Rc=Rc-K
H3
ERNEUTE AUSFÜHRUNG SUMME-VON-RRO-DUKTEN-RECHEN-BEFEHL
120
PC

FIG. 9

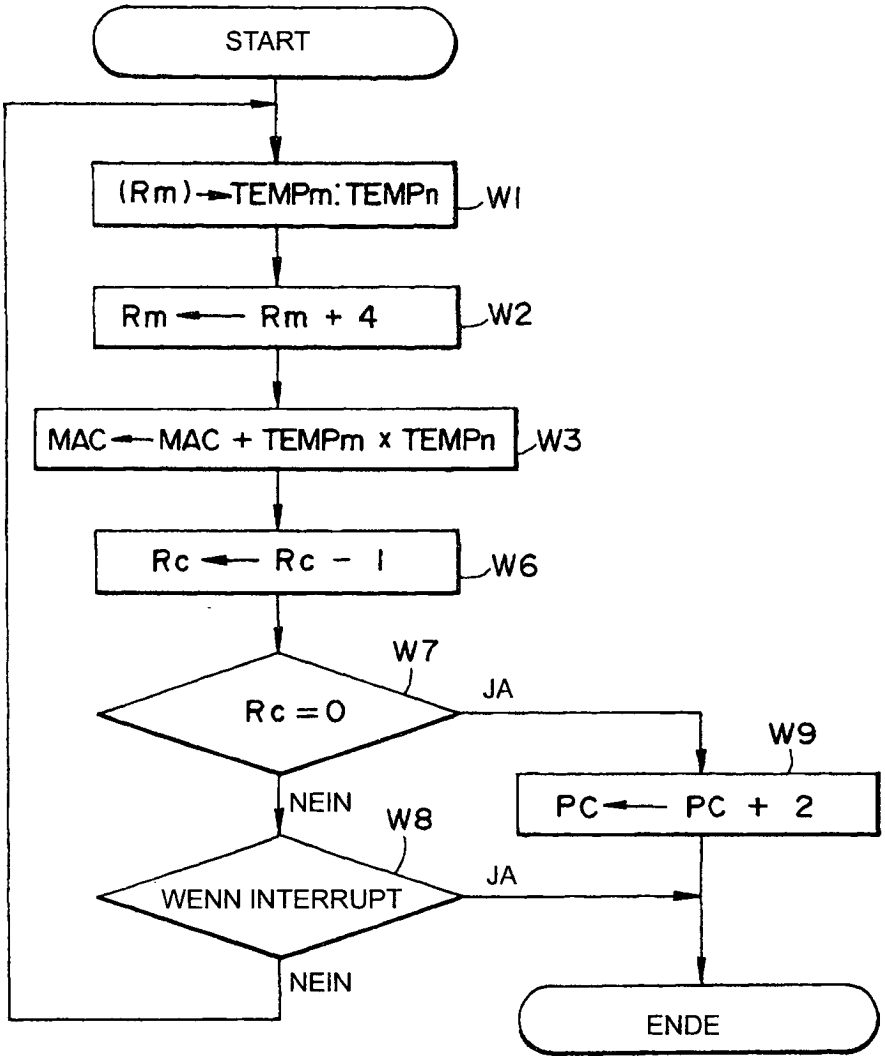

START
(Rm)→TEMPm:TEMPn
W1
Rm ← Rm + 4
W2
MAC ← MAC + TEMPm x TEMPn
W3
Rc ← Rc - 1
W6
W7
Rc = 0
JA
W9
PC ← PC + 2
NEIN
W8
WENN INTERRUPT
JA
NEIN
ENDE

FIG. 10A

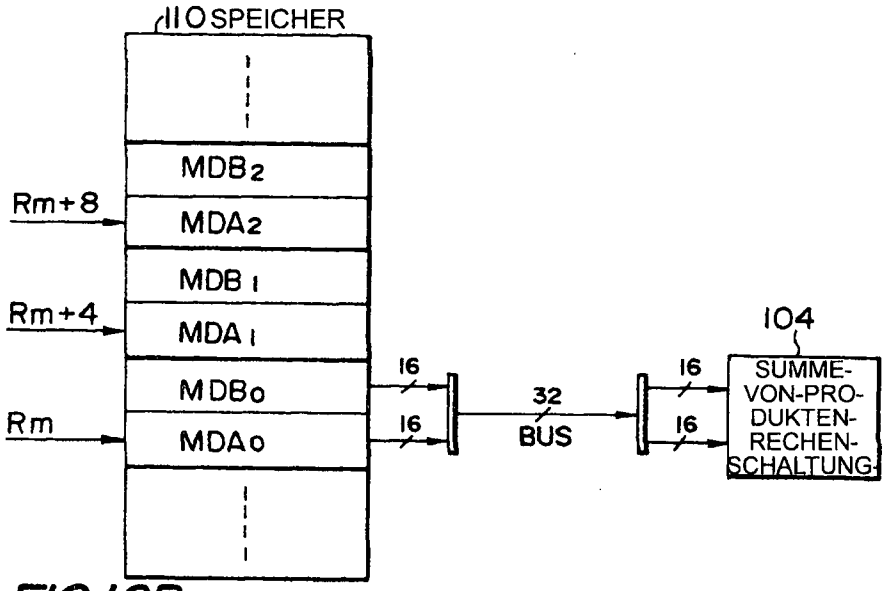


FIG. 10B

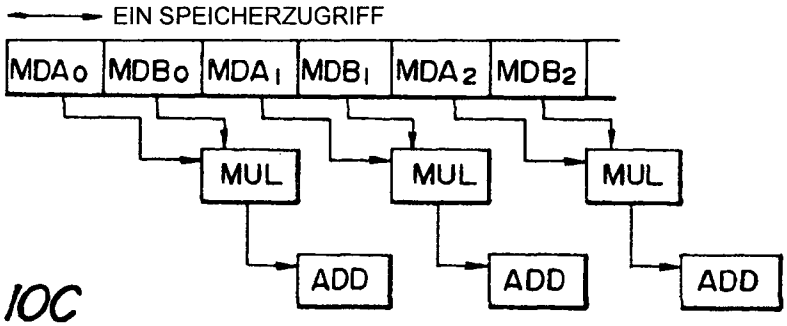


FIG. 10C

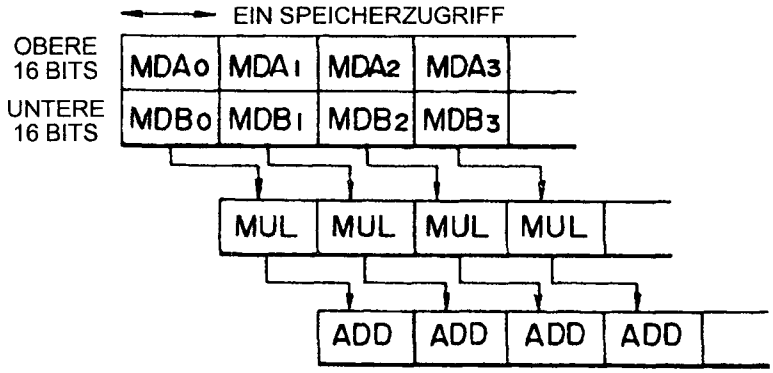


FIG. 11A

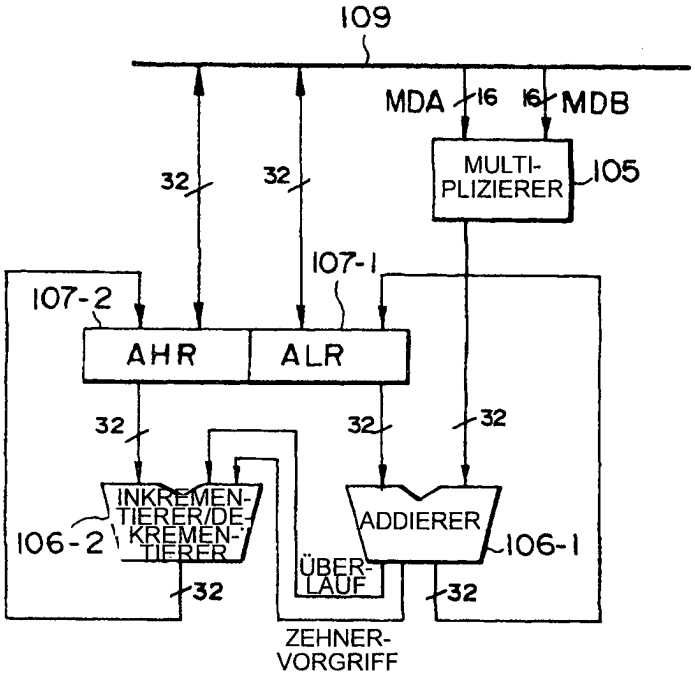


FIG. 11B

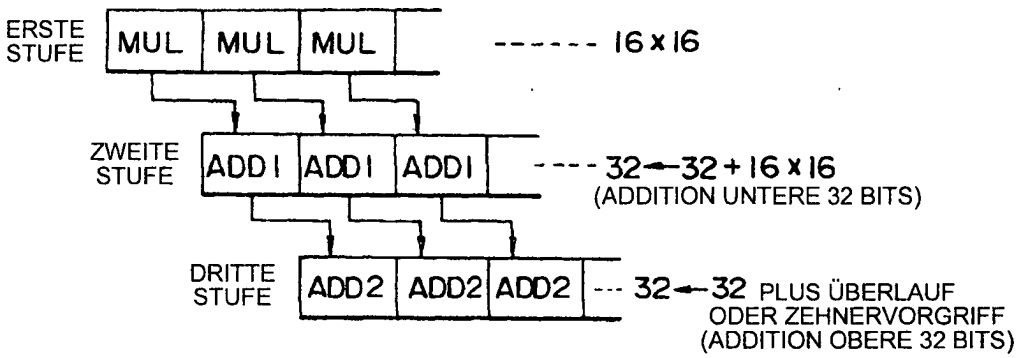


FIG. 12

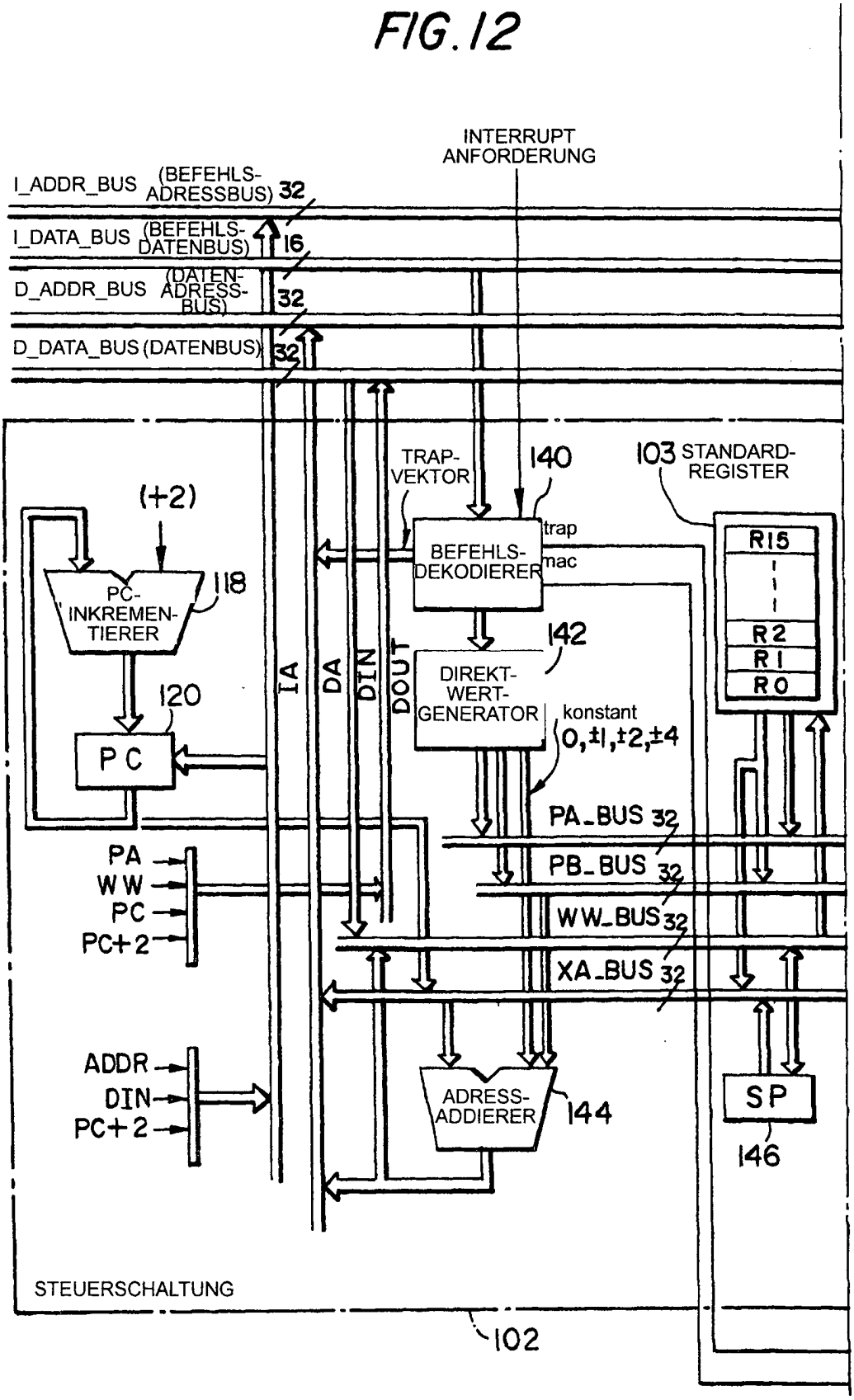

INTERRUPT ANFORDERUNG
I_ADDR_BUS (BEFEHLS-ADRESSBUS) 32
I_DATA_BUS (BEFEHLS-DATENBUS) 16
D_ADDR_BUS (DATEN-ADRESS-BUS) 32
D_DATA_BUS (DATENBUS) 32
TRAP-VEKTOR
140
103 STANDARD-REGISTER
(+2)
PC-INKREMEN-TIERER
118
BEFEHLS-DEKODIERER
trap
mac
R15
R2
R1
R0
142
DIREKT-WERT-GENERATOR
konstant
0,±1,±2,±4
120
PC
IA
DA
DIN
DOUT
PA_BUS 32
PB_BUS 32
WW_BUS 32
XA_BUS 32
PA
WW
PC
PC+2
ADDR
DIN
PC+2
ADRESS-ADDIERER
144
SP
146
STEUERSCHALTUNG
102

FIG. 12 (FORTS.)

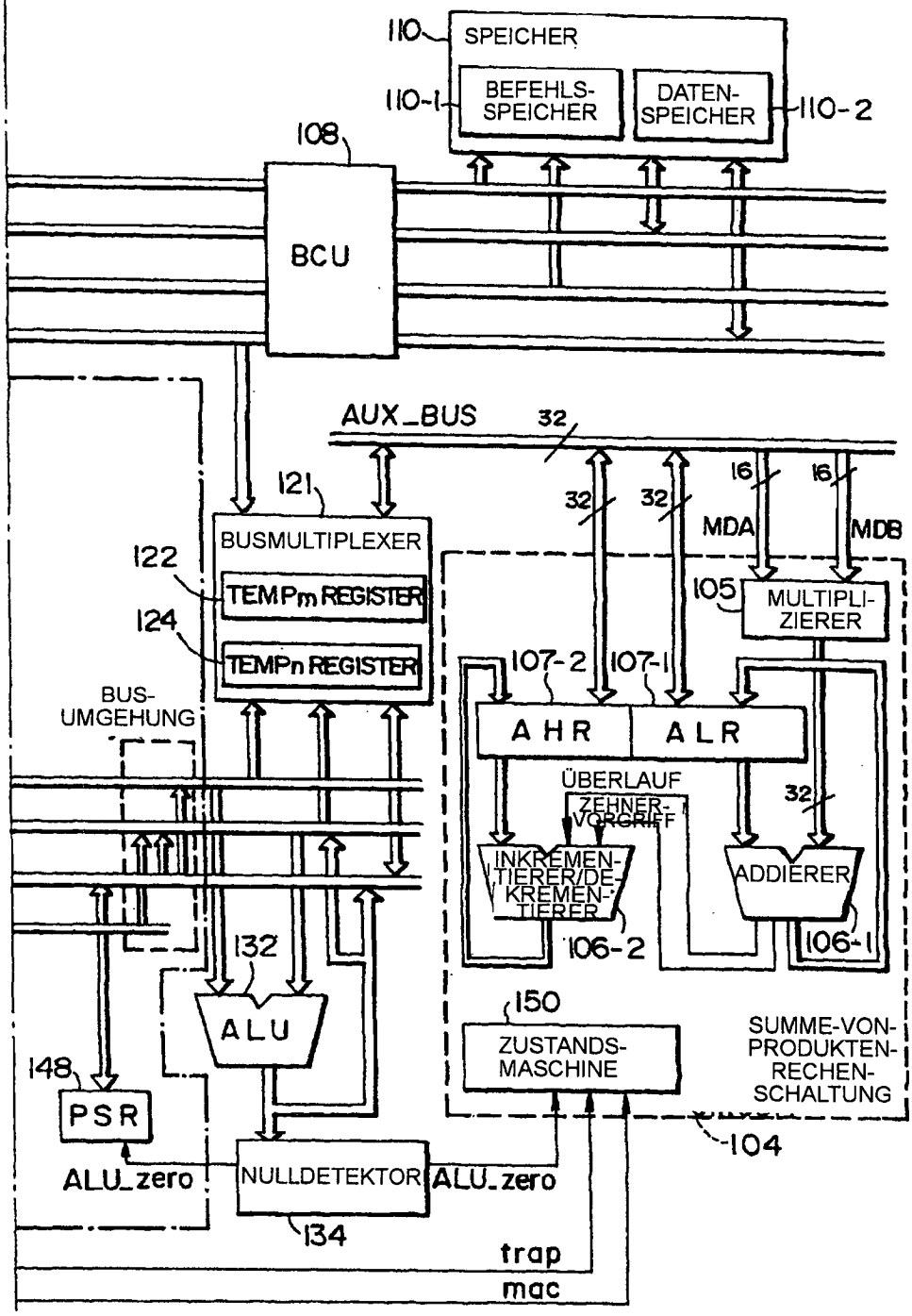

110
SPEICHER
110-1
BEFEHLS-SPEICHER
DATEN-SPEICHER
110-2
108
BCU
AUX_BUS
32
121
BUSMULTIPLEXER
122
TEMPm REGISTER
124
TEMPn REGISTER
BUS-UMGEHUNG
32
32
16
16
MDA
MDB
105
MULTIPLI-ZIERER
107-2
107-1
AHR
ALR
ÜBERLAUF
ZEHNER-VORGRIFF
32
INKREMEN-TIERER/DE-KREMEN-TIERER
ADDIERER
106-2
106-1
132
ALU
150
ZUSTANDS-MASCHINE
SUMME-VON-PRODUKTEN-RECHEN-SCHALTUNG
104
148
PSR
ALU_zero
NULLDETEKTOR
ALU_zero
134
trap
mac

FIG. 13

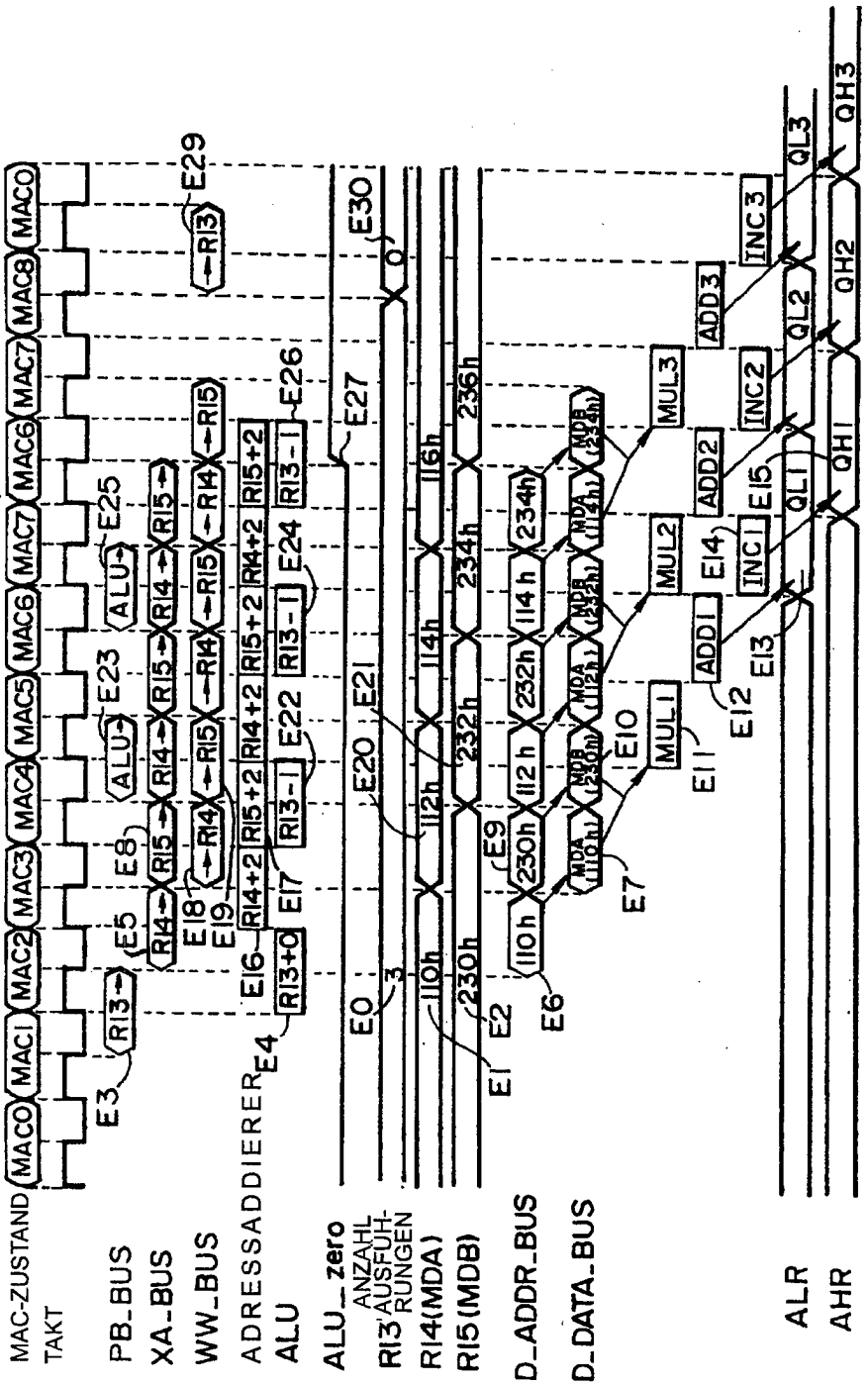

MAC-ZUSTAND
TAKT
PB_BUS
XA_BUS
WW_BUS
ADRESSADDIERER
ALU
ALU_zero
ANZAHL AUSFUHRUNGEN
R13
R14(MDA)
R15(MDB)
D_ADDR_BUS
D_DATA_BUS
ALR
AHR
MAC0
MAC1
MAC2
MAC3
MAC4
MAC5
MAC6
MAC7
MAC8
R13
R14
R15
R13+0
R14+2
R15+2
R13-1
ALU
E0
E1
E2
E3
E4
E5
E6
E7
E8
E9
E10
E11
E12
E13
E14
E15
E16
E17
E18
E19
E20
E21
E22
E23
E24
E25
E26
E27
E28
E29
E30
110h
230h
112h
232h
114h
234h
116h
236h
MDA
MDB
MUL1
MUL2
MUL3
ADD1
ADD2
ADD3
INC1
INC2
INC3
QL1
QL2
QL3
QH1
QH2
QH3

FIG. 14A

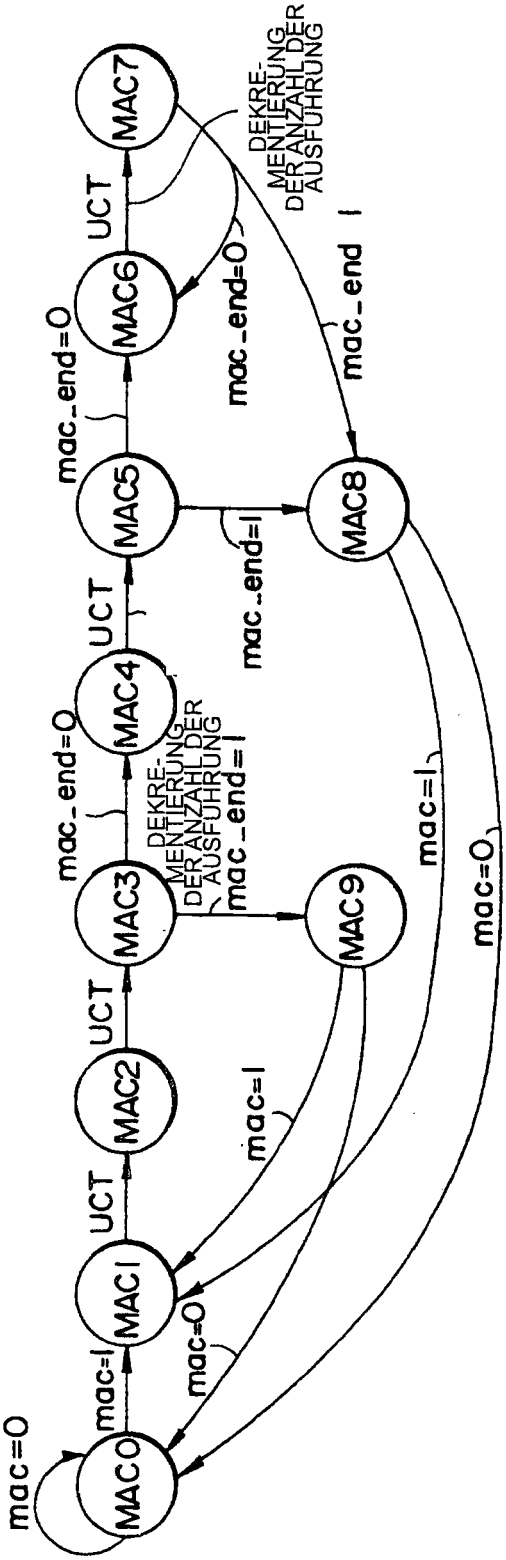


FIG. 14B

		INAKTIVER (=0) ZUSTAND	AKTIVER (=1) ZUSTAND
mac_end (mac_zero oder mac_trap)	mac_zero	RESET oder MAC8 oder MAC9	ALU_zero UND (MAC3 oder MAC5 oder MAC7)
	mac_trap	RESET oder MAC8 oder MAC9	trap UND (MAC5 ODER MAC7)

FIG. 15

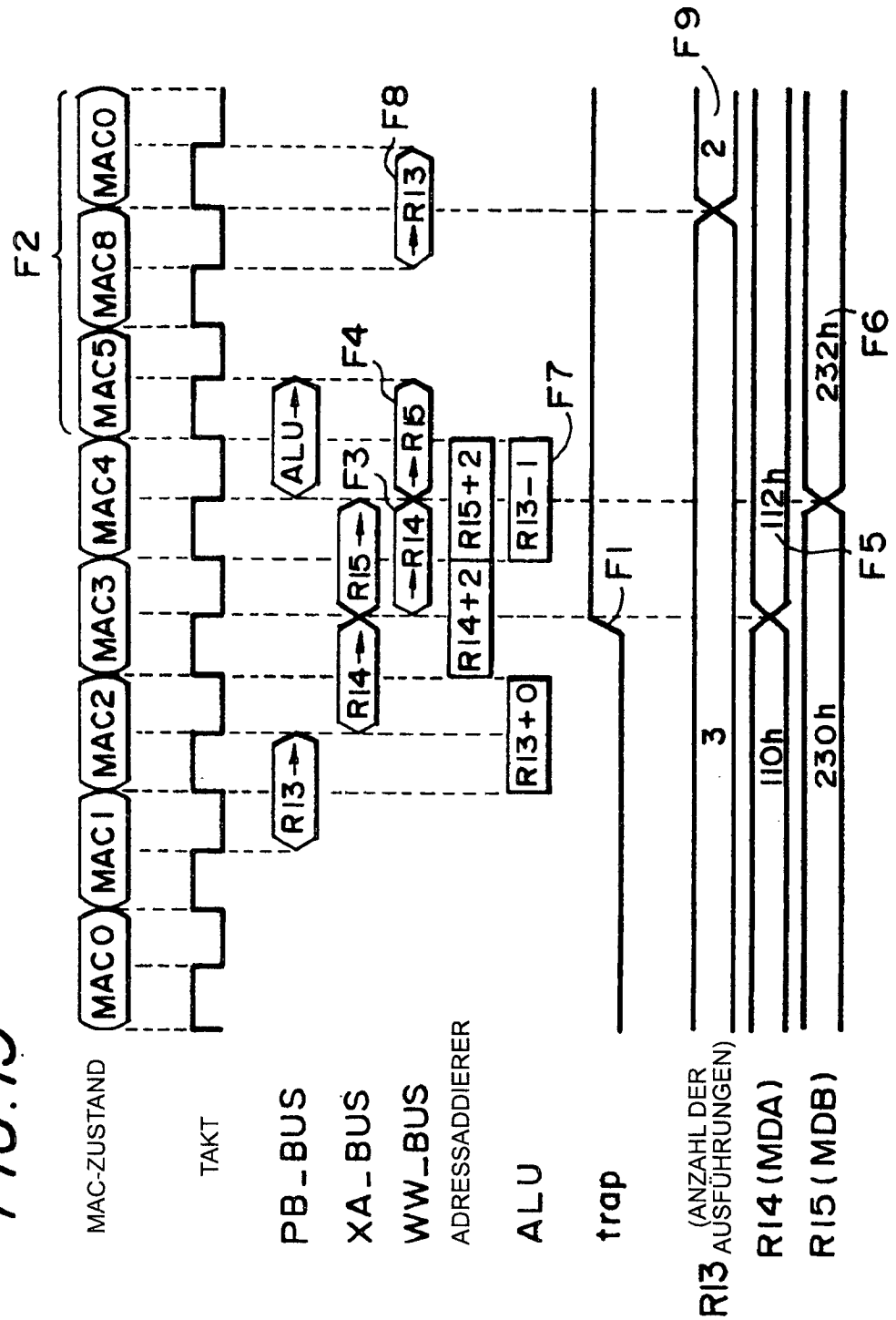

MAC-ZUSTAND
TAKT
PB_BUS
XA_BUS
WW_BUS
ADRESSADDIERER
ALU
trap
R13 (ANZAHL DER AUSFÜHRUNGEN)
R14 (MDA)
R15 (MDB)
MAC0
MAC1
MAC2
MAC3
MAC4
MAC5
MAC8
MAC0
F1
F2
F3
F4
F5
F6
F7
F8
F9
R13→
ALU→
R14→
R15→
←R14
←R15
←R13
R14+2
R15+2
R13+0
R13-1
3
110h
230h
112h
232h
2

FIG. 16

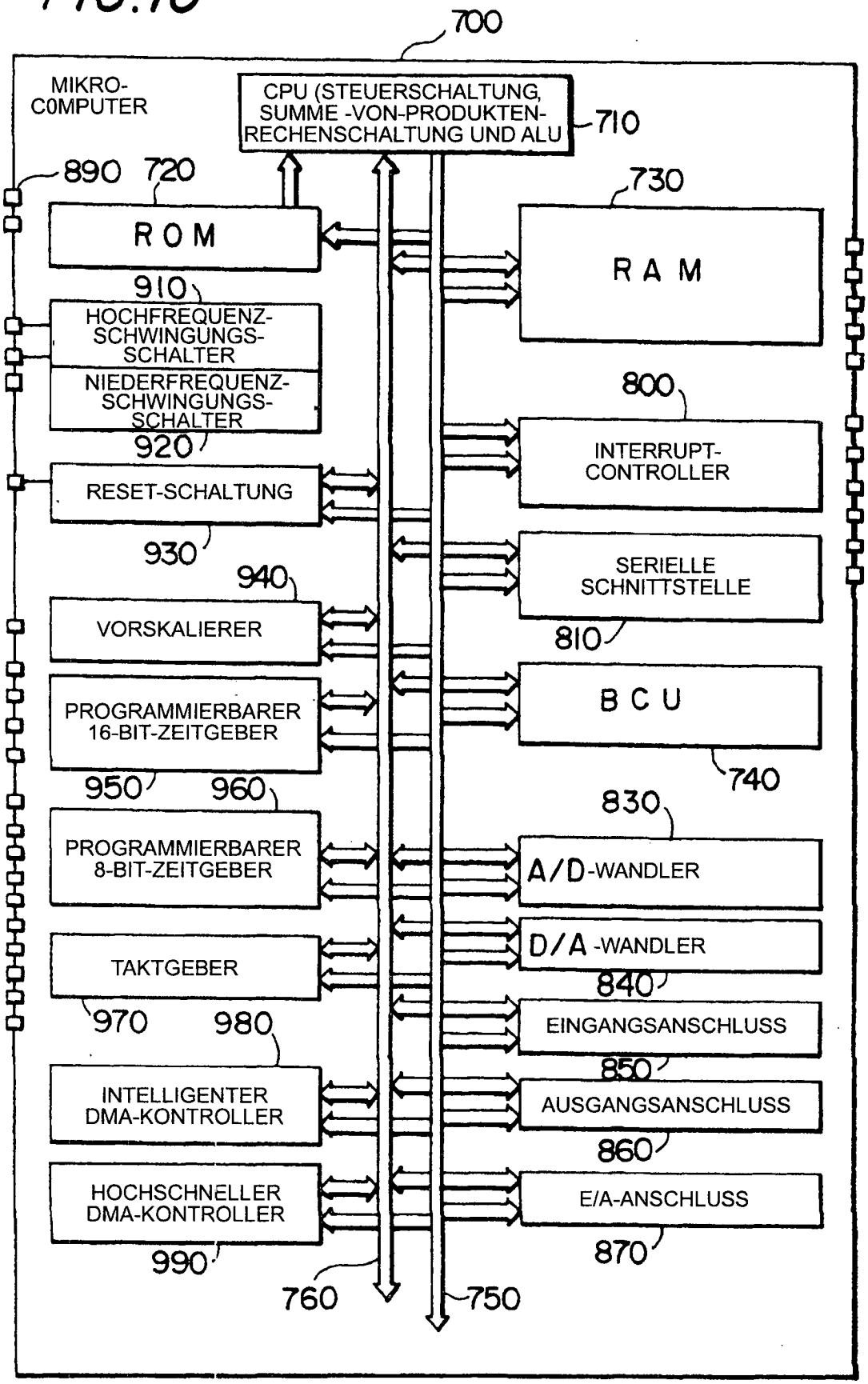

700
MIKRO-
COMPUTER
CPU (STEUERSCHALTUNG, SUMME -VON-PRODUKTEN- RECHENSCHALTUNG UND ALU
710
890
720
ROM
730
RAM
910
HOCHFREQUENZ-SCHWINGUNGS-SCHALTER
NIEDERFREQUENZ-SCHWINGUNGS-SCHALTER
920
RESET-SCHALTUNG
930
940
VORSKALIERER
PROGRAMMIERBARER 16-BIT-ZEITGEBER
950
960
PROGRAMMIERBARER 8-BIT-ZEITGEBER
TAKTGEBER
970
980
INTELLIGENTER DMA-KONTROLLER
HOCHSCHNELLER DMA-KONTROLLER
990
800
INTERRUPT-CONTROLLER
SERIELLE SCHNITTSTELLE
810
BCU
740
830
A/D-WANDLER
D/A-WANDLER
840
EINGANGSANSCHLUSS
850
AUSGANGSANSCHLUSS
860
E/A-ANSCHLUSS
870
760
750

FIG. 17A

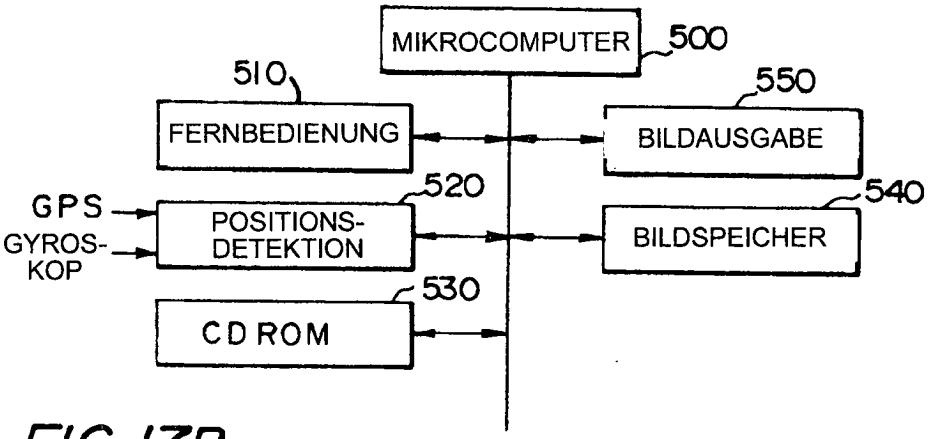


FIG. 17B

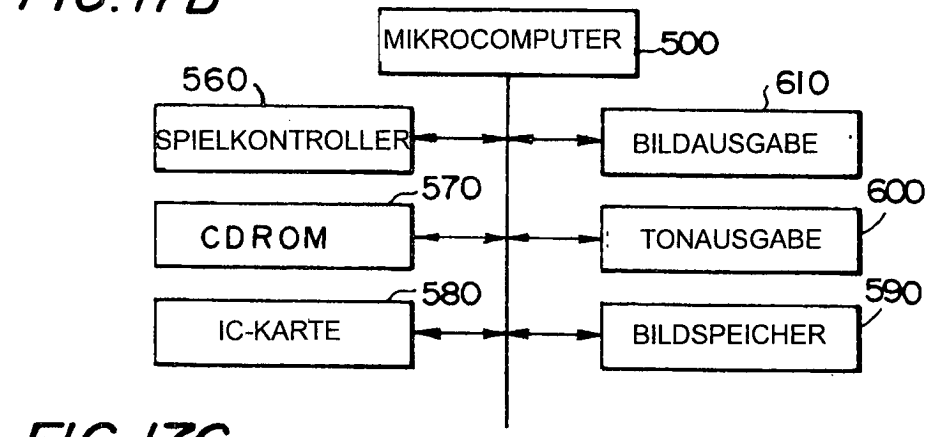


FIG. 17C

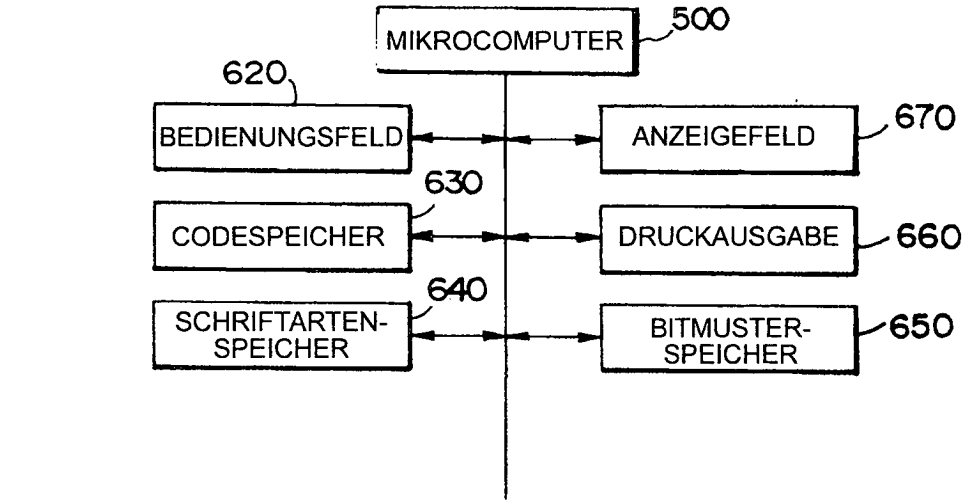


FIG. 18A

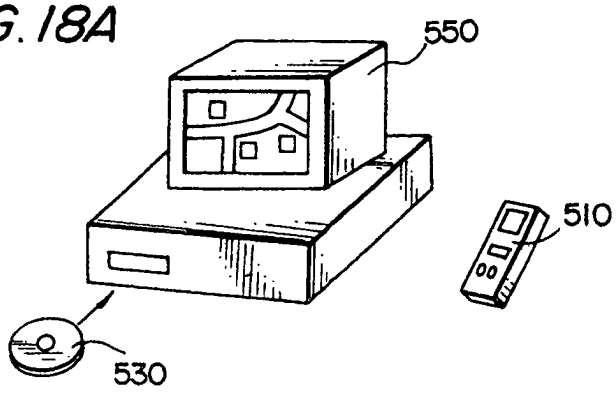


FIG. 18B

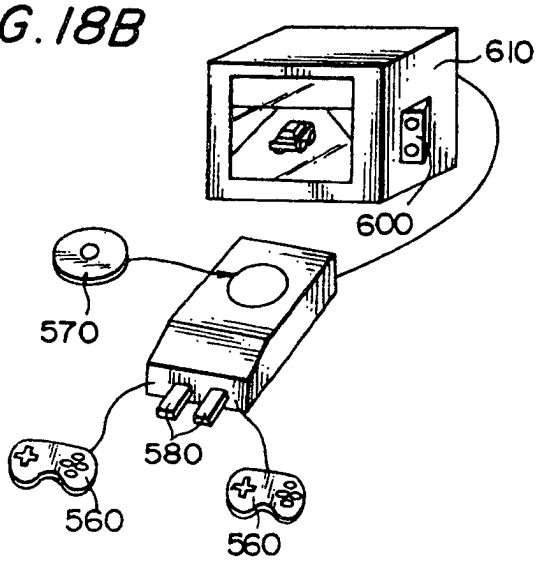


FIG. 18C