POLSKA RZECZPOSPOLITA LUDOWA

URZĄD PATENTOWY PRL

OPIS PATENTOWY 115020

Patent dodatkowy
do patentu nr ———

Zgłoszono: 16.07.74 (P. 172789)

Pierwszeństwo: 18.07.73 Wielka Brytania

Zgłoszenie ogłoszono: 02.06.75

Opis patentowy opublikowano: 20.08.1982

Int. Cl.[2]
G06F 13/00
G06F 9/16

Twórca wynalazku: ———

Uprawniony z patentu: International Computers Limited, Londyn (Wielka Brytania)

System przetwarzania danych

1

Przedmiotem wynalazku jest system przetwarzania danych. W znanych systemach przetwarzania danych występuje problem przydziału obszaru pamięci dla różnych kategorii informacji w przypadku, gdy ilość informacji, jaka ma być zapamiętana w każdej kategorii, zmienia się podczas pracy systemu. Jeden ze sposobów przydziału pamięci w takim przypadku polega na przydzielaniu oddzielnego obszaru pamięciowego dla każdej kategorii informacji. Przy takim rozwiązaniu obszar pamięciowy musi być jednakże stosunkowo duży, ponieważ musi on spełniać wszystkie wymagania pamięciowe związanej z nim kategorii informacji. Prowadzi to do marnotrawienia dużego obszaru pamięci, ponieważ należy oczekiwać, że w danej chwili jedynie niektóre kategorie informacji będą potrzebowały tak dużego obszaru pamięciowego, podczas gdy inne będą potrzebowały go znacznie mniej lub wcale. Straty obszaru pamięci można zmniejszyć wprowadzając organizację, umożliwiającą zapisanie informacji do dowolnego, dostępnego obszaru pamięciowego.

Takie rozwiązanie wymaga stosowania tablicy przeznaczonej do przechowywania zapisu informującego o miejscu pamiętania każdej pozycji informacji, i stosunkowo złożonego systemu zarządzania pamięcią, który steruje wykorzystywaniem pamięci. Takie rozwiązanie prowadzi jednakże do rozproszenia informacji tej samej kategorii po całym obszarze pamięci, tak więc in-

2

formacja ta nie jest przechowywana w kolejnych komórkach pamięciowych, co w niektórych sytuacjach jest niezbędne, na przykład w przypadku, gdy informację tę stanowi mikroprogram, który zwykle jest wykonywany sekwencyjnie.

Rozwiązanie tego problemu zostało zaproponowane przez D. Knutha w książce „The Art of Computer Programming" 1969, Tom 1, str. 242. Według tego rozwiązania, informacje dwóch różnych kategorii są zapisane w dwóch stosach (lub listach), które posuwają się ku sobie od oddzielnych adresów bazowych w miarę dopływu informacji. W rozwiązaniu tym dwie kategorie informacji zajmują wspólną przestrzeń pamięci i zderzają się jedynie w przypadku, gdy całkowite zapotrzebowanie na przestrzeń pamięci jest większe niż dostępny obszar wolny. Jednakże rozwiązanie nie rozwiązuje problemu, gdy całkowite zapotrzebowanie jest większe niż dostępny obszar wolny i występuje stan „nadmiaru", po którym zostaje zakończona praca systemu.

Celem wynalazku jest rozwiązanie tego problemu i zabezpieczenie ciągłej pracy systemu nawet przy dużym zapotrzebowaniu na obszar pamięciowy.

System przetwarzania danych, w którym dwie kategorie informacji są zapisywane na dwa stosy, które w miarę zapisywania informacji są zapisy- które w miarę zapisywania informacji są rozbudowywane ku sobie, począwszy od oddzielnych

adresów bazowych, według wynalazku charakteryzuje się tym, że jeden ze stosów ma wyższy priorytet niż drugi, zaś w przypadku, gdy informacja ma być zapisana w jednym z dwóch stosów i wobec stwierdzenia braku wolnego obszaru do zapisania tej informacji, usuwany jest cały stos o priorytecie niższym dla dostarczenia wolnego obszaru dla informacji zapisywanej.

Ze stosu o priorytecie wyższym można usunąć dowolną wybraną liczbę bloków informacji.

Informacja zapisana poniżej adresu bazowego jednego ze stosów nie może być usunięta z pamięci, zaś adres bazowy może się zmieniać, aby czasowo zapobiec usunięciu części informacji przechowywanej w stosie w sąsiedztwie adresu bazowego.

Pamięć systemu jest pamięcią mikroprogramową, a informacja stanowi bloki materiału mikroprogramowego, a ponadto system jest wyposażony w mikroprogramową jednostkę sterującą, zapoczątkowywującą wykonywanie sekwencji mikrorozkazów przechowywanych w pamięci mikroprogramowej.

System według wynalazku może pracować w sposób ciągły, bez wystąpienia stanu „nadmiaru".

Przedmiot wynalazku jest przedstawiony w przykładzie wykonania na rysunku, na którym fig. 1 przedstawia schemat blokowy części systemu, fig. 2 — schemat blokowy innej części systemu, fig. 3—5 — schematy blokowe mikroprogramów systemu, a fig. 6 — modyfikację systemu według wynalazku.

Na figurze 1 przedstawiono system według wynalazku, który składa się z pamięci głównej **10**, przeznaczonej do przechowywania danych i programu, pamięci mikroprogramowej **11** i mikroprogramowej jednostki sterującej **12**. Podczas pracy systemu według wynalazku jednostka sterująca **12** pobiera rozkazy programowe z pamięci operacyjnej **10** i dla każdego z nich inicjuje odpowiednią sekwencję mikrorozkazów z pamięci mikroprogramowej **11** w celu wykonania rozkazów. Tego rodzaju mikroprogramowe sterowanie systemu przetwarzania danych nie jest oczywiście żadną nowością w tej dziedzinie, dlatego też w opisie tym nie przedstawiono struktury mikroprogramowej jednostki sterującej **12**.

Pamięć mikroprogramowa **11** ma stosunkowo małe wymiary w porównaniu z pamięcią operacyjną **10**, jest natomiast dużo od niej szybsza (mniejszy czas dostępu), dzięki czemu umożliwia prawie natychmiastowe uzyskanie dostępu do mikrorozkazów dla jednostki mikroprogramowej. Obszar **13** pamięci mikroprogramowej **11** jest zarezerwowany dla mikroprogramu podstawowego (zwanego prymitywnym interfejsem), przeznaczonego do podstawowego sterowania systemu. Mikroprogram ten jest zapisany na stałe w pamięci mikroprogramowej. Pozostały obszar **14** pamięci mikroprogramowej przechowuje kopie pewnej liczby bloków dodatkowego materiału mikroprogramowego wykorzystywanego na bieżąco przez system. Jeden z obszarów pamięci operacyjnej **10** spełnia zadanie pamięci zapasowej, w której przechowuje się kopie wszystkich bloków mikro-

programów systemu. Każdy z tych boków można przesyłać do pamięci mikroprogramowej **11**, gdy jest on wywoływany dla wykorzystania przez jednostkę mikroprogramową **12**. Przesłany blok będzie w ogólności zachodził na informację obecną juz w pamięci mikroprogramowej, dlatego też bloki mikroprogramowe będą nazywane „zakładkami". Na fig. 1 obszar zakreskowany **15** w pamięci operacyjnej **10** stanowi zakładka kopii głównej, podczas gdy odpowiadająca jej kopia w pamięci mikroprogramowej jest przedstawiona jako obszar zakreskowany **16**.

Zastosowanie obszaru zapasowego dla zakładek i możliwości zakładkowania pamięci mikroprogramowej umożliwia systemowi według wynalazku korzystanie z dużej liczby mikroprogramów bez konieczności stosowania bardzo dużej i bardzo szybkiej pamięci mikroprogramowej, która byłaby niezwykle kosztowna.

W przykładzie wykonania urządzenia według wynalazku sklasyfikowano dwie kategorie zakładek mikroprogramowych:

(I) Zakładki systemowe. Stanowią one bloki mikroprogramowe, które w rzeczywistości stanowią rozszerzenia mikroprogramu interfejsu prymitywnego w celu zwiększenia zakresu i efektywności systemu. Mogą one wykonywać na przykład funkcje nadzorcze, takie jak zmiana stron, lub mogą być wymagane do emulacji, to jest inicjacji pracy innej maszyny cyfrowej o innym kodzie rozkazów i innej organizacji. W ogólności technika zakładek systemowych jest wprowadzana przez producentów dużych maszyn cyfrowych.

(II) Zakładki użytkownika. Stanowią one bloki mikroprogramowe, przeznaczone do wykonywania zadań specjalnych, wymaganych często przy określonych zastosowaniach, na przykład przy obliczaniu pierwiastka kwadratowego. W ogólności zakładki te są pisane przez użytkownika systemu, a nie przez producenta.

Klasyfikacja ta jest oczywiście arbitralna i przyjęto ją jedynie dla wygody.

Przesyłaniem zakładek pomiędzy pamięcią operacyjną **10** a pamięcią mikroprogramową **11** steruje tablica zakładek **17**, która w rzeczywistości stanowi część pamięci operacyjnej **10** i jest określana zawartością dwu rejestrów: rejestru adresu bazowego tablicy zakładek **18**, który przechowuje adres **VTBA**, będący adresem początku tablicy zakładek w pamięci operacyjnej, i rejestru długości **19** tablicy zakładek. Tablica zakładek **17** przechowuje po jednym elemencie każdej zakładki systemu. Każdy taki element składa się z pola **VL**, pola **VA** i pola **VSA**.

Pole **VL** określa długość zakładki (czyli liczbę mikrorozkazów w zakładce). W ogólności różne zakładki mają różną długość.

Pole **VA** określa adres początkowy zakładki w pamięci mikroprogramowej. Jeżeli zakładka nie jest aktualnie obecnie w pamięci mikroprogramowej, pole to jest zerowane. Pole **VSA** określa adres początkowy kopii głównej zakładki w pamięci operacyjnej.

Jeden z takich elementów **20** tablicy, przeznaczony dla kopii **15** i **16** zakładek przedstawiono

na fig. 1, przy czym zależności pól **VL, VA** i **VSA** zakładek **15** i **16** zaznaczono strzałkami.

Gdy program systemu wymaga wykorzystania określonej zakładki mikroprogramowej, wówczas generuje on rozkaz wywołania, który umieszcza deskryptor w rejestrze deskryptorów **21**. Deskryptor ten składa się z bitu **VT** oraz pola **VN**.

Bit **VT**, określa rodzaj zakładki. VT = 0 określa zakładkę użytkownika, a VT = 1 określa zakładkę systemową. Pole **VN** określa położenie w tablicy zakładek elementu wymaganej zakładki.

Pole **VN** jest podawane na komparator **22**, który porównuje jego wartość z długością **VTL** tablicy zakładek przechowywaną w rejestrze **19**. Jeżeli **VN** jest większe od **VTL**, oznacza to wystąpienie błędu i wygenerowanie na linii **23** sygnału przerwania, który powoduje wejście do odpowiedniego programu przerwania w obszarze **13**. Jeżeli natomiast **VN** nie jest większe od **VTL**, wówczas wartość **VN** jest podawana na sumator **24**, który dodaje ją do wartości **VTBA**, pochodzącej z rejestru **18**, co ma na celu utworzenie adresu odpowiedniego elementu w tablicy zakładek **17**. Następnie odczytuje się pole **VA** tego elementu i wykorzystuje się je do zaadresowania pamięci mikroprogramowej **11**. Zakładając, że kopia pożądanej zakładki jest aktualnie obecna w pamięci mikroprogramowej, wykonywany jest skok do zakładki przechowywanej w tej pamięci. Jeżeli jednakże kopia wymaganej zakładki nie jest aktualnie obecna w pamięci mikroprogramowej **11**, wtedy wartością **VA** będzie zero, co oznacza uzyskanie dostępu do zerowej komórki pamięci mikroprogramowej. W komórce tej jest przechowywany rozkaz skoku, który powoduje wykonanie skoku do specjalnego programu zakładki, w obszarze **13** interfejsu prymitywnego przy czym program ten steruje ładowaniem kopii wymaganej zakładki z pamięci operacyjnej **10** do pamięci mikroprogramowej **11**.

Na figurze 2 przedstawiono fragment systemu według wynalazku, w którym program zakładek umieszcza zakładki pochodzące z pamięci głównej na dwu stosach **25** i **26** w pamięci mikroprogramowej **11**, zgodnie z określonym rodzajem zakładki. Zakładki systemowe są umieszczane na stosie **25**, który jest rozbudowywany w górę w pamięci mikroprogramowej (to jest w kierunku wzrastających wartości adresów), począwszy od adresu bazowego **SB**. Zwykle, wymieniony adres bazowy jest równy pierwszemu wolnemu adresowi leżącemu powyżej granicy interfejsu prymitywnego. Zakładki użytkownika są umieszczone na stosie **26**, który jest rozbudowywany ku dołowi pamięci mikroprogramowej, począwszy od adresu bazowego **UB**, który może stanowić górną granicę pamięci. Tak więc zakładki te są wprowadzane na dwa stosy, które powiększają się jeden w kierunku drugiego, aż do chwili ich ewentualnego spotkania. W tym ostatnim przypadku stos **25** zakładek systemowych ma priorytet, który umożliwia mu zapisywanie swoich elementów na stosie **26** zakładek użytkownika. Procedura ta zostanie opisana niżej.

Program zakładek wykorzystuje zbiór rejestrów **27**, które mogą w rzeczywistości być przechowywane w pierwszych miejscach zakładki **17** (fig. 1). Rejestry te przechowują następujące wartości:

UB — adres bazowy stosu **26** zakładek użytkownika.

UP — wskaźnik pierwszego, wolnego adresu na początku stosu zakładek użytkownika.

SP — wskaźnik pierwszego wolnego adresu na początku stosu **25** zakładek systemowych

SB — adres bazowy stosu zakładek systemowych.

ST — całkowita liczba zakładek systemowych na stosie zakładek systemowych.

Zależność między tymi rejestrami i ich położenie w pamięci mikroprogramowej przedstawiono na fig. 2.

Zawartości rejestrów **UP** i **SP** są odejmowane i zwiększane o jedność w układzie odejmującym **28** w celu otrzymania wartości X = UP — SP = 1, która jak zostanie to pokazane, będzie reprezentowała ilość wolnego obszaru między początkami stosów **25** i **26**, do którego można będzie zapisywać dalsze zakładki.

Pierwszą operacją programu zakładek jest zbadanie zawartości pola **VT** w rejestrze deskryptora **21** (fig. 1) w celu określenia rodzaju deskryptora. Jeżeli VT = 0, co oznacza zakładkę użytkownika, wówczas wykonywana jest część programu zakładek przedstawiona na fig. 3, jeżeli natomiast VT = 1, co oznacza zakładkę systemową, wtedy jest wykonywana część programu zakładek przedstawiona na fig. 4.

W przypadku zakładki użytkownika (fig. 3) wartość **VL** pochodząca z aktualnie adresowanego elementu tablicy zakładek **17** jest porównywana (blok **30**) z wartością wolnego obszaru **X** otrzymaną z układu **28** w celu określenia, czy wolny obszar pamięci mikroprogramowej zawarty pomiędzy początkami stosów jest wystarczająco duży do przechowania w nim nowej zakładki. Jeżeli **VL** jest mniejszy od lub równy **X**, nowa zakładka może być natychmiast ładowana do komórek od UP — VL + 1 do **UP** pamięci mikroprogramowej, tak aby zwiększyć stos zakładek użytkownika w kierunku ku dołowi. W tym samym czasie tablica zakładek **17** jest aktualizowana na zasadzie zapisania adresu początkowego UP — VL + 1 nowej zakładki do pola **VA**.

Następnie jest aktualizowany rejestr adresu wskaźnika **UP** (blok **32**) na zasadzie odjęcia od jego zawartości wartości **VL**. Stanowi to zakończenie programu zakładek dla tego przypadku.

Jeżeli okaże się, że **VL** jest większy od **X** (blok **30**), oznacza to, że nowa zakładka nie zmieści się w dostępnym obszarze pamięciowym. W celu stworzenia wystarczająco dużego obszaru dla nowej zakładki, są usuwane wszystkie zakładki umieszczone aktualnie na stosie zakładek użytkownika (blok **33**). Po usunięciu każdej zakładki odpowiadający jej element przechowywany w tablicy zakładek **17** jest aktualizowany na zasadzie zerowania pola **VA**, co oznacza brak zakładki w pamięci mikroprogramowej. Następnie jest aktualizowany wskaźnik **UP** (blok **34**) na zasadzie nadania mu wartości równej **UB**. Następnie wartość

VL jest ponownie porównywana z X (blok 35). Jeżeli VL jest nadal zbyt duże nawet po usunięciu wszystkich zakładek użytkownika, wówczas program zakładek nie może już nic więcej dokonać i generowany jest sygnał przerwania. Jeżeli jednakże VL jest teraz mniejszy lub równy X, to program zakładek może być zakończony w sposób opisany wyżej (bloki 31 i 32).

W przypadku zakładki systemowej (fig. 4) wartość VL jest ponownie porównywana z X (blok 40) w celu określenia, czy dostępny wolny obszar pamięciowy jest wystarczająco duży do zapamiętania zakładki. Jeżeli VL jest mniejszy lub równy X, wtedy zakładka może być natychmiast załadowana (blok 41) do komórek SP do SP + VL — — 1 pamięci mikroprogramowej tak, aby wydłużyć stos zakładek systemowych w kierunku do góry. W tym samym czasie jest aktualizowana tablica zakładek 17 na zasadzie zapisania adresu początkowego SP nowej zakładki do pola VA. Na końcu jest aktualizowany rejestr adresu wskaźnikowego SP (blok 42) na zasadzie dodania do niego wartości VL, a wartość ST (liczba zakładek systemowych w stosie) jest zwiększana o jedność. Operacja ta kończy wykonywanie programu zakładek dla tego przypadku.

Jeżeli VL jest większe od X (blok 40), oznacza to, że nowa zakładka systemowa nie zmieści się w dostępnym obszarze pamięciowym. Jednakże stos 25 zakładek systemowych ma priorytet nad stosem zakładek użytkownika, tak więc w celu utworzenia miejsca dla nowej zakładki systemowej usuwane są (blok 43) wszystkie zakładki przechowywane bieżąco na stosie 26 zakładek użytkownika. W miarę usuwania każdej zakładki, odpowiadający jej element tablicy zakładek 17 jest aktualizowany na zasadzie zerowania pola VA. Następnie jest aktualizowany wskaźnik UP (blok 44) na zasadzie nadawania mu wartości równej UB. Następnie wartość VL jest porównywana ponownie z X (blok 45). Jeżeli VL jest nadal zbyt duże nawet po usunięciu wszystkich zakładek użytkownika, wówczas generowany jest sygnał przerwania. Jeżeli jednakże VL jest obecnie mniejsze lub równe X, wtedy program zakładek może zakończyć się jak poprzednio (bloki 41, 42).

Z powyższego opisu wynika jasno, że zakładki użytkownika są usuwane automatycznie, gdy obszar zajmowany przez nie jest potrzebny dla nowych zakładek użytkownika lub dla zakładek systemowych. Z drugiej strony, zakładki sytemowe mogą być usuwane jedynie przez rozkaz specjalny „zeruj zakładkę systemową", który inicjuje odpowiedni program w mikroprogramie interfejsu prymitywnego. W ten sposób można usunąć dowolną liczbę zakładek systemowych w oparciu o zasadę „ostatnia do, pierwsza z", przy czym liczba R zakładek usuwanych jest określana przez wymieniony rozkaz.

Na figurze 5 przedstawiono mikroprogram wykonujący rozkaz „zeruj zakładkę systemową". Krok pierwszy polega na porównaniu (blok 51) wartości R (jest to liczba zakładek systemowych, które będą usuwane) i ST (liczba zakładek systemowych w pamięci mikroprogramowej). Jeżeli R jest większe od ST, oznacza to oczywiście błąd i jest generowany sygnał przerwania. W innym przypadku jest wykonywany blok 52 polegający na badaniu zera wartości R. Jeżeli R jest różne od zera, wtedy jest wykonywany blok 53, polegający na usunięciu jednej zakładki systemowej z początku stosu 25 zakładek systemowych i zaktualizowaniu odpowiadającego jej elementu tablicy zakładek na zasadzie wyzerowania pola VA. Następnie są aktualizowane rejestry 27 (blok 54) na zasadzie odejmowania długości VL z zakładki usuniętej od SP, i zmniejszenia ST o jedność. Wartość R jest również zmniejszana o jedność. Następnie następuje powrót do bloku 52, gdzie jest badana wartość R (czy R = 0). Jeżeli R = 0, oznacza to, że usunięto wymaganą liczbę zakładek systemowych i program zostaje zakończony. Jeżeli R jest różne od zera, wtedy pętla 53, 54, 52 jest powtarzana do chwili wyzerowania się R.

Istnieje możliwość zmiany adresu bazowego SB za pomocą odpowiedniego rozkazu, co ma na celu chwilowe potraktowanie zakładek systemowych jako część interfejsu prymitywnego (zapobiega to usunięciu tych zakładek ze stosu). W przypadku tego rodzaju zmiany adresu bazowego SB należy również zmienić wartość ST.

W odmianie systemu według wynalazku, przedstawionej na fig. 6, może wystąpić trzecia kategoria zakładek. Tę trzecią kategorię mogą stanowić na przykład zakładki emulacyjne, uważane uprzednio za zakładki systemowe. W odmianie tej zakładki emulacyjne są zapisywane na trzeci stos 61 w pamięci mikroprogramowej, który rozpoczyna się od adresu bazowego EB, większego od adresu bazowego UB stosu zakładek użytkownika, i jest rozbudowywany w kierunku dwu pozostałych stosów. Korzystne jest, że stos 61 zakładek emulacyjnych ma wyższy priorytet od stosu 26 zakładek użytkownika i od stosu 25 zakładek systemowych, dzięki czemu stos 61 może zapisywać swą informację na obu pozostałych stosach. Jednakże, nie można zapisywać zakładek emulacyjnych na mikroprogramie interfejsu prymitywnego lub na zakładkach systemowych traktowanych chwilowo jako taki mikroprogram poniżej adresu SB.

W zbiorze 27 występują dwa dodatkowe rejestry, które przechowują adres bazowy (EB) stosu 61 i adres wskaźnikowy (EP), wskazujący pierwszą, wolną komórkę na początku stosu 61. Deskryptor przechowywany w rejestrze 21 (fig. 1) musi mieć dwubitowe pole VT w celu identyfikowania trzech różnych rodzajów zakładek, a ponadto program zakładek musi być rozszerzony w celu umożliwienia ładowania zakładek emulacyjnych. Do zerowania zakładek emulacyjnych służy dodatkowy program „zeruj", podobny do programu przedstawionego na fig. 5.

W innej odmianie systemu według wynalazku, system obejmuje dwa procesory, które wykorzystują tę samą pamięć mikroprogramową 11, przy czym każdy procesor ma przydzielony oddzielny obszar pamięci mikroprogramowej, w którym jest

przechowywany mikroprogram tego procesora. Procesory te korzystają również ze wspólnej pamięci operacyjnej **10**. W tym przypadku zwiększona jest tablica zakładek **17**, dzięki czemu każdy jej element obejmuje pola **VL, VA, VSA** dla zakładek związanych z jednym z procesorów, i podobny zbiór pól dla zakładek procesora drugiego. Ponadto każdy z procesorów jest wyposażony w zbiór rejestrów **27**.

Jakkolwiek wynalazek opisano w powiązaniu z zakładkowaniem mikroprogramu w pamięci mikroprogramowej, to jednak może być on wykorzystywany wszędzie tam, gdzie do pamięci zapisuje się dwie lub więcej kategorii informacji.

Zastrzeżenia patentowe

1. System przetwarzania danych, w którym dwie kategorie informacji są zapisywane na dwa stosy, które w miarę zapisywania informacji są rozbudowywane ku sobie, począwszy od oddzielnych adresów bazowych, **znamienny tym,** że jeden ze stosów **(25)** ma wyższy priorytet niż drugi **(26),** zaś w przypadku, gdy informacja ma być zapisana w jednym z dwóch stosów i wobec stwierdzenia braku wolnego obszaru **(X)** do zapisania tej informacji, usuwany jest cały stos **(26)** o priorytecie niższym dla dostarczenia wolnego obszaru dla informacji zapisywanej.

2. System według zastrz. 1, **znamienny tym,** że ze stosu **(25)** o priorytecie wyższym można usunąć dowolną wybraną liczbę bloków informacji.

3. System według zastrz. 1, albo 2, **znamienny tym,** że informacja zapisana poniżej adresu bazowego **(SB)** jednego ze stosów **(25)** nie może być usunięta z pamięci, zaś adres bazowy **(SB)** może się zmieniać, aby czasowo zapobiec usunięciu części informacji przechowywanej w stosie **(25)** w sąsiedztwie adresu bazowego.

4. System według zastrz. 3, **znamienny tym,** że pamięć jest pamięcią mikroprogramową **(11),** a informacja stanowi bloki materiału mikroprogramowego, a ponadto system jest wyposażony w mikroprogramową jednostkę sterującą **(12),** zapoczątkowywującą wykonywanie sekwencji mikrorozkazów przechowywanych w pamięci mikroprogramowej **(11).**

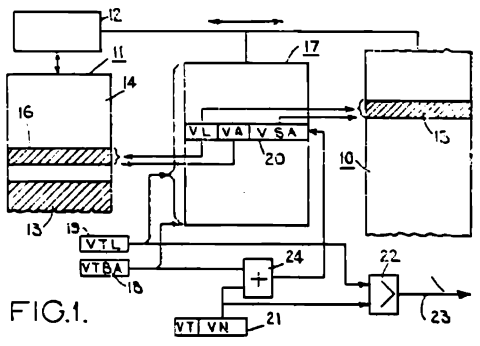

FIG.1.

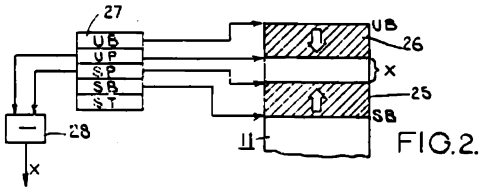

FIG.2.

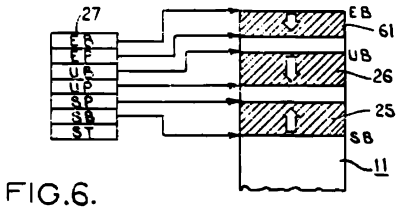

FIG.6.

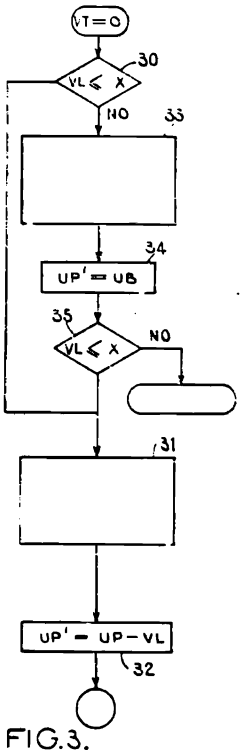

FIG.3.

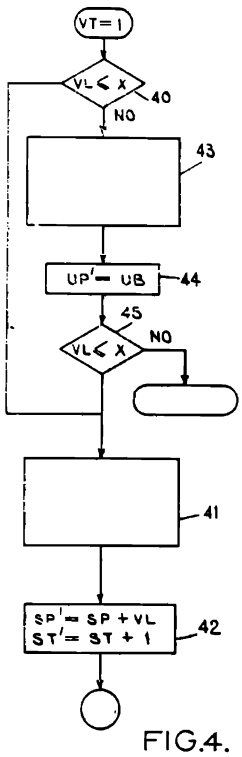

FIG.4.

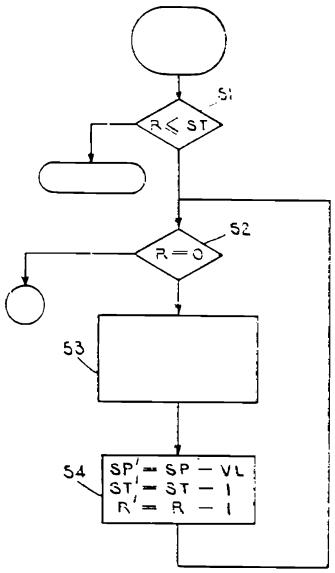

FIG.5.

PZGraf. Koszalin D-548 100 A-4

Cena 100 zł