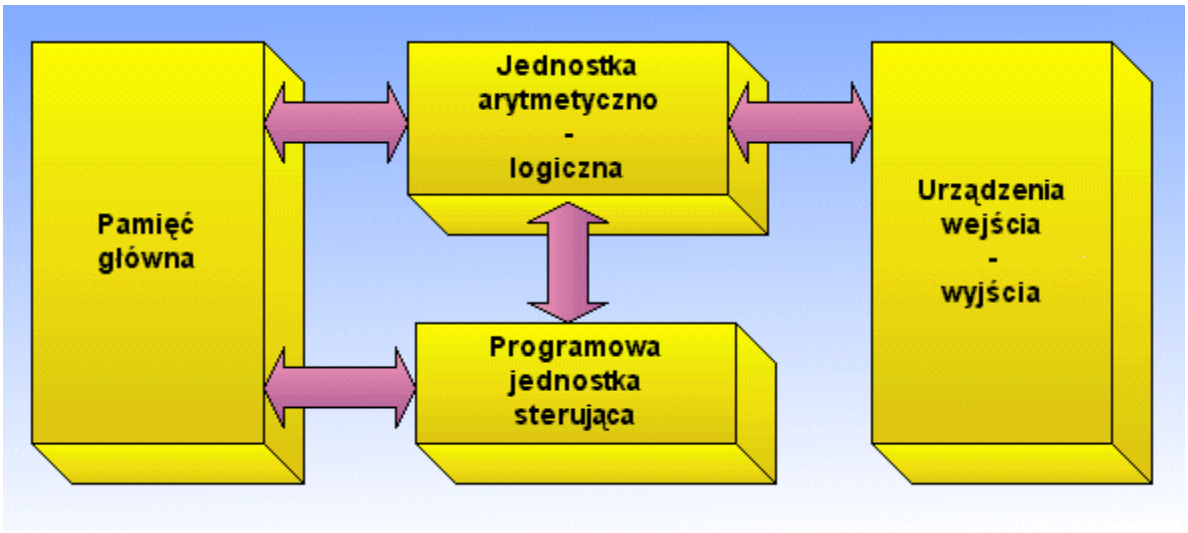
1. Koncepcja Johna von Neumanna

W 1945 roku po raz pierwszy została opublikowana **koncepcja przechowywanego programu**, która jest przypisywana jednemu z konsultantów projektu ENIAC - John'owi von Neumanowi (należy wspomnieć, że niemal w tym samym czasie podobna koncepcja została opracowana również przez Alana Turinga). Idea ta miała położyć podwaliny pod nowy komputer von Neumana - EDVAC (ang. *Electronic Discrete Variable Computer*).

W roku 1946 von Neumann w Princeton Institute for Advanced Studies rozpoczął projektowanie komputera, który wykorzystywał program przechowywany w pamięci. Komputer ten nazwano IAS i był on prototypem wszystkich następnych komputerów o ogólnym przeznaczeniu.



Rysunek 1.1. Struktura komputera IAS

Powyższy rysunek przedstawia ogólną strukturę komputera IAS, na którą składają się [1]:

- **Pamięć główna** - w tej pamięci przechowywane są dane oraz rozkazy.
- **Jednostka arytmetyczno-logiczna (ALU)** - wykonuje działania na danych binarnych.
- **Jednostka sterująca** - interpretuje i wykonuje rozkazy z pamięci.
- **Urządzenia wejścia/wyjścia** - ich pracą kieruje jednostka sterująca.

John von Neumann tak opisał niegdyś tę strukturę komputera [1]:

1. Po pierwsze, ponieważ urządzenie to jest przede wszystkim komputerem, najczęściej będzie wykonywało elementarne operacje matematyczne - dodawanie, odejmowanie, mnożenie, dzielenie. Jest więc rozsądne, że powinno posiadać wyspecjalizowane "organy" do wykonywania tych operacji.
   Należy jednak zauważyć, że chociaż powyższa zasada jako taka brzmi rozsądnie, to szczegółowy sposób jej realizacji wymaga głębokiego zastanowienia[...] W każdym przypadku *centralna, arytmetyczna* część urządzenia będzie prawdopodobnie musiała istnieć, co oznacza występowanie *pierwszej specyficznej części komputera: CA*.

2. Po drugie, logiczne sterowanie urządzeniem, to znaczy odpowiednie szeregowanie jego operacji może być najefektywniej realizowane przez centralny organ sterujący. Jeżeli urządzenie ma być *elastyczne*, to znaczy możliwie *uniwersalne*, należy rozróżniać specyficzne rozkazy związane z określonym problemem i ogólne "organy" sterujące, dbające o wykonanie tych rozkazów - czymkolwiek by one nie były. Te pierwsze muszą być w jakiś sposób

przechowywane; te drugie - reprezentowane przez określone działające części urządzenia. Przez *sterowanie centralne* rozumiemy tylko tę ostatnią funkcję, a "organy", które ją realizują, tworzą *drugą specyficzną część urządzenia: CC*.

3. Po trzecie, jakiekolwiek urządzenie, które ma wykonywać długie i skomplikowane sekwencje działań (w szczególności obliczeń), musi mieć odpowiednio dużą pamięć[...]
   (b) Rozkazów kierujących rozwiązywaniem skomplikowanego problemu może być bardzo dużo, zwłaszcza wtedy, gdy kod jest przypadkowy (a tak jest w większości przypadków). Muszą one być pamiętane [...]

4. Trzy specyficzne części CA, CC (razem C) oraz M odpowiadają neuronom skojarzeniowym w systemie nerwowym człowieka. Pozostają do przedyskutowania równoważniki neuronów *sensorycznych* (doprowadzających) i *motorycznych* (odprowadzających). Są to "organy" *wejścia i wyjścia* naszego urządzenia.
   Urządzenie musi mieć możliwość utrzymania kontaktu z wejściem i wyjściem za pomocą specjalistycznego narzędzia. Narzędzie to będzie nazwane *zewnętrznym narzędziem rejestrującym urządzenia: R* [...]
5. Po czwarte urządzenie musi być wyposażone w organy przenoszące [...] informację z R do swoich specyficznych części C i M. "Organy" te stanowią jego wejście, a więc *czwartą, specyficzną część: I*. Zobaczymy, że najlepiej jest dokonywać wszystkich przeniesień z R (poprzez I) do M, a nigdy bezpośrednio do C [...]

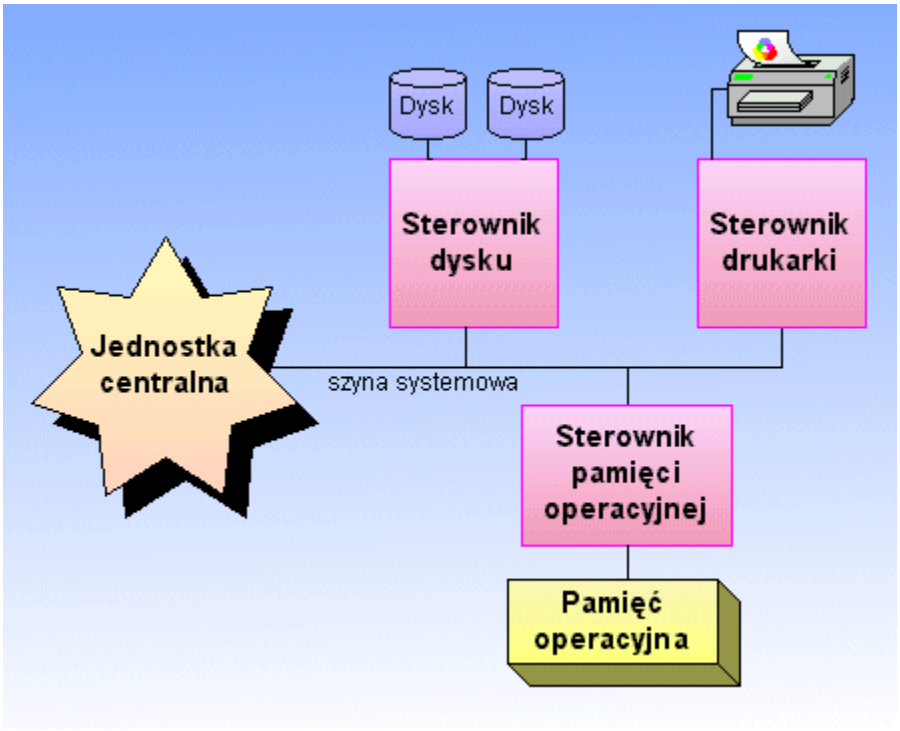
6. Po piąte, urządzenie musi mieć "organy" służące do przenoszenia [...] ze swoich specyficznych części C i M do R. Organy te tworzą jego *wyjście*, a więc *piątą specyficzną część: O*. Zobaczymy, że znowu najlepiej jest dokonywać wszystkich transferów z M (poprzez O) do R, nigdy zaś bezpośrednio z C [...]

Dzisiaj, niemal wszystkie popularne komputery klasy PC mają strukturę, którą stworzył niegdyś Neumann, dlatego też noszą miano maszyn von Neumanna.

NASTĘPNA

2. Architektura systemu komputerowego

System komputerowy składa się z co najmniej: jednej **jednostki centralnej**, kilku **sprzętowych sterowników** (ang. *device controller*), połączonych wspólną **magistralą danych** umożliwiającą współpracę ze wspólną **pamięcią** [2] (patrz rysunek poniżej).



Rysunek 1.2. Architektura systemu komputerowego

Każde urządzenie posiada swój sterownik (czy to jest dysk twardy, czy karta grafiki, czy drukarka). Sterownik ma za zadanie zapewnić uporządkowany, synchroniczny dostęp do wspólnej pamięci pomimo możliwości działania współbieżnego z jednostką centralną. Zdarza się bowiem, że jednostka centralna i sterowniki działają współbieżnie.

Ostatecznie możemy powiedzieć, że na architekturę komputera składają się:

1. **jednostka centralna (CPU)**, czyli przynajmniej jeden procesor.
2. **magistrala systemowa** poprzez którą komunikują się i przesyłają dane komponenty systemu komputerowego.
3. **wspólna pamięć**, która służy do przechowywania danych i kodu instrukcji procesora.
4. **urządzenia wejścia-wyjścia** (np. klawiatura - monitor), dzięki którym możliwa jest komunikacja z komputerem.

Zauważmy, że owa architektura jest łudząco podobna do "koncepcji przechowywanego programu" John'a von Neumanna, która została opublikowana dawno, dawno temu w Ameryce ...

NASTĘPNA

3. Procesor

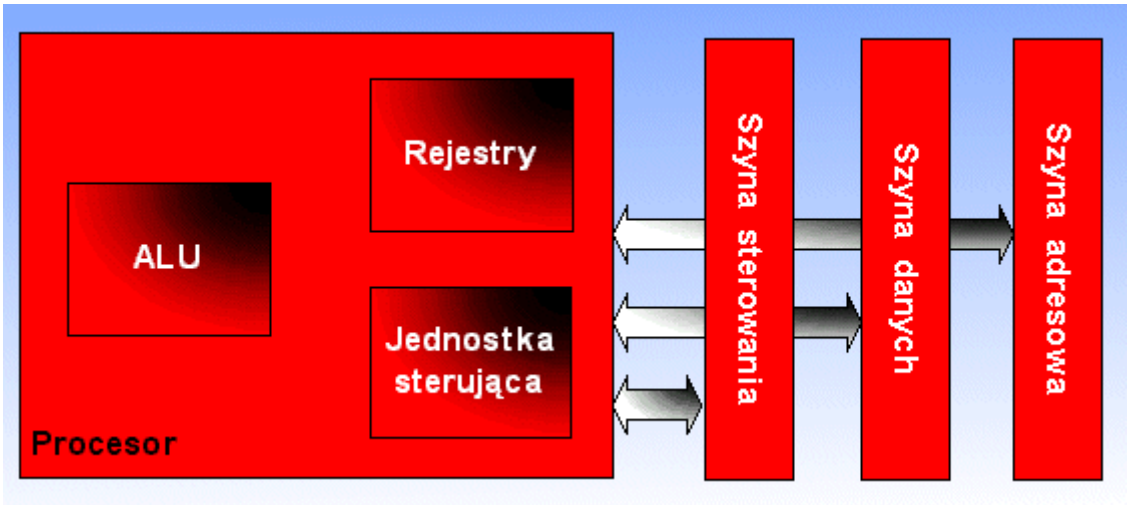
Procesor jest centralną częścią systemu komputerowego, ale czy wszyscy zdają sobie sprawę z tego, jakie zadania ten procesor właściwie powinien realizować? Otóż procesor musi:

- *Pobierać rozkazy* - w celu odczytania poleceń z pamięci.
- *Interpretować rozkazy* - polecenia należy zdekodować, aby wiedzieć jakie operacje należy wykonać.
- *Pobierać dane* - z pamięci lub modułu wejścia-wyjścia.
- *Przetwarzać dane* - przeprowadzać na danych pewne operacje arytmetyczne lub logiczne.
- *Zapisywać dane* - w pamięci lub module wejścia-wyjścia.

Aby procesor miał możliwość wykonywania powyższych zadań musi dysponować małą pamięcią wewnętrzną, która wymagana jest do czasowego przechowywania danych i rozkazów (procesor musi np. pamiętać lokalizację poprzedniego rozkazu po to, by "odnaleźć" rozkaz następny).

Poniżej przedstawione są dwa rysunki, które ukazują bardzo uproszczoną budowę procesora. Rysunek 1.3 przedstawia trzy główne składniki procesora [1]:

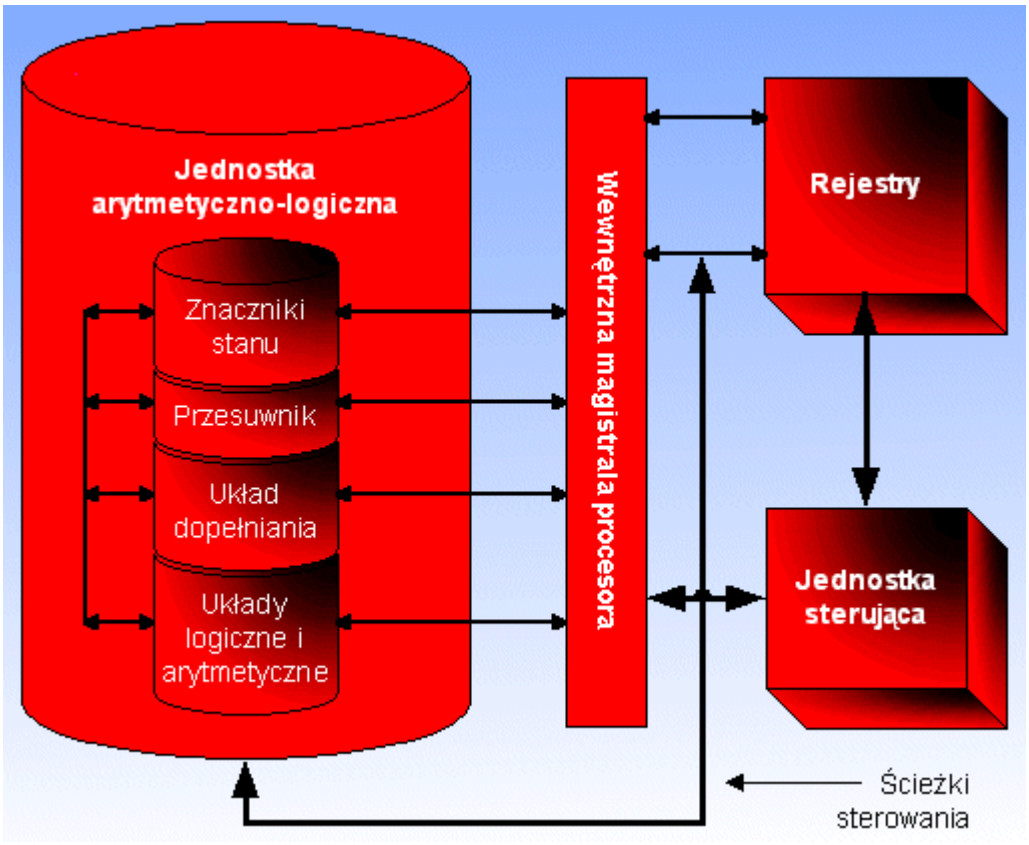
- **Jednostkę arytmetyczno-logiczną (ALU)** - ta część procesora jest odpowiedzialna za wykonanie obliczeń i przetwarzanie danych.
- **Jednostkę sterującą (CU)** - kieruje ona ruchem danych i rozkazów do/z procesora i steruje pracą ALU.
- **Rejestry** - część pamięci wewnętrznej uprzywilejowanej wykorzystywanej przez procesor do adresowania, odczytu danych, itp.



Rysunek 1.3. Procesor a magistrala systemowa

Komunikacja między procesorem a resztą systemu komputerowego odbywa się poprzez magistralę systemową, której poszczególne składniki (szyny danych, adresowa i sterowania) mogą, ale nie muszą być fizycznie od siebie odseparowane.

Nieco dokładniej strukturę wewnętrzną procesora przedstawia kolejny rysunek. Element określony jako **wewnętrzna magistrala procesora** zajmuje się przesyłaniem danych między różnymi rejestrami a ALU (w rzeczywistości ALU operuje jedynie na danych zaczerpniętych z wewnętrznej pamięci procesora) [1]. Rysunek przedstawia również elementy podstawowe ALU:



Rysunek 1.4. Procesor: struktura wewnętrzna

Na koniec zapamiętajmy, że to właśnie ALU jest "sercem" procesora. Rejestry są używane do przechowywania danych wewnętrznych w procesorze, a niektóre z nich zawierają informacje potrzebne do zarządzania porządkowaniem rozkazów (np. słowo stanu programu), pozostałe zawierają dane przeznaczone dla ALU, pamięci i modułów wejścia-wyjścia lub dane przekazane przez te jednostki. Wewnętrzne ścieżki danych są używane do przenoszenia danych pomiędzy rejestrami oraz między rejestrami a ALU. Zewnętrzne ścieżki danych łączą rejestry z pamięcią i modułami wejścia-wyjścia często za pomocą magistrali systemowej. Jednostka sterująca koordynuje wykonanie operacji wewnątrz procesora.

NASTĘPNA

4. Magistrale systemowe

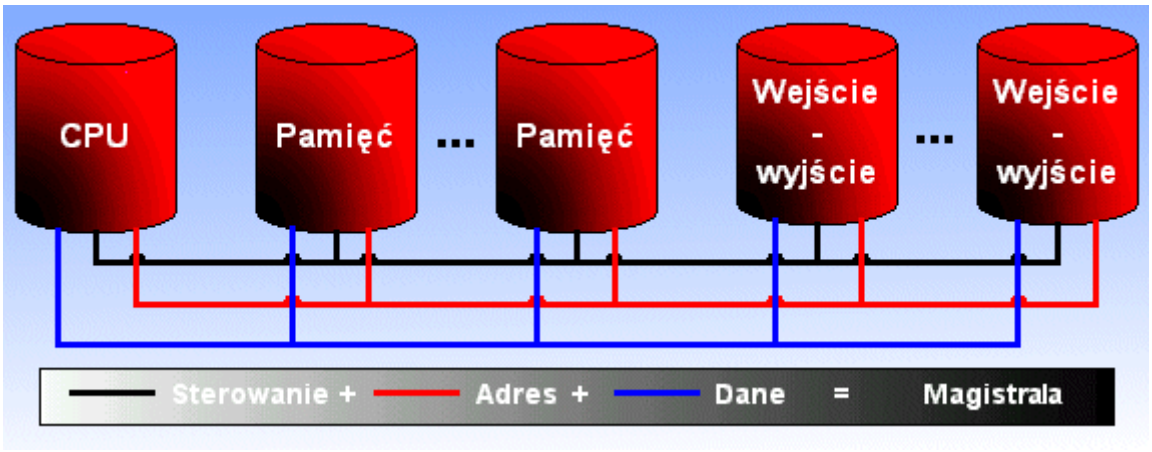
System komputerowy zawiera pewną liczbę magistrali. Najważniejsza z nich to **magistrala systemowa** (niektóre systemy wykorzystują kilka magistrali systemowych) łącząca najważniejsze podzespoły komputera tj. procesor, pamięć, układy wejścia-wyjścia [1].

Magistrala jest wspólnym nośnikiem transmisji (ang. *shared transmission medium*). Korzysta z niej pewna liczba urządzeń, które poprzez nią komunikują się wykorzystując kilka linii komunikacyjnych, (np. 16, mówimy wtedy o magistrali 16-bitowej), których liczba określają ile jednocześnie danych binarnych może transmitować magistrala. Dostęp do magistrali jest sekwencyjny - w danej chwili może korzystać z niej tylko jedno urządzenie - gdyby w tym samym czasie nadawało kilka urządzeń, ich sygnały zakłócałyby się wzajemnie, co prowadziłoby do przekłamań w transmisji danych.

(4.1) Struktura magistrali

Magistrala systemowa składa się z wielu oddzielnych linii o określonym znaczeniu lub funkcji. Linie zawarte w magistrali można podzielić na trzy grupy (patrz rysunek 1.5):

- linie danych
- linie adresów
- linie sterowania
- linie zasilania (opcjonalnie tzn. nie zawsze)



Rysunek 1.5. Połączenie magistrali z elementami systemu komputerowego

Linie danych służą do przenoszenia danych pomiędzy komponentami systemu. Linie te zwane są **szyną danych** (ang. *data bus*). Liczba linii szyny określa jej *szerokość* (przeważnie występuje 8, 16, 32, 64 linii), czyli ilość bitów, które można jednocześnie przesyłać. Szerokość szyny danych jest jednym z głównych czynników określających wydajność systemu komputerowego. Na przykład jeśli szyna danych ma szerokość 16 bitów, a wykonywany rozkaz 32 bity, to procesor chcąc nie chcąc, musi łączyć się z modułem pamięci dwukrotnie w czasie tego cyklu rozkazu [1].

Linie adresowe wykorzystywane są do określania skąd i dokąd przepływają informacje przesyłane szyną danych. Na przykład procesor, który chce odczytać dane z pamięci, umieszcza adres tych danych na szynie adresowej, a chcąc zapisać efekt ich wykonania wysyła na szynę adresową adres pamięci (lub modułu we-wy), do którego chce przesłać tenże wynik [1].
Istotnym parametrem również tej szyny jest jej *szerokość*. Szerokość szyny adresowej określa

maksymalną pojemność pamięci systemu, a ściślej rzecz ujmując przestrzeń adresową, która nie musi być pamięcią, ale np. urządzeniem we-wy. Zwykle bity mniej znaczące służą do adresowania pamięci lub określonych portów wewnątrz modułu we-wy, zaś bity bardziej znaczące do wybrania określonego modułu (np. pamięci lub we-wy).

Linie sterowania zapewniają regulację dostępu do linii danych i linii adresowych oraz umożliwiają sterowanie ich wykorzystaniem [1]. Sygnały sterujące przesyłane pomiędzy modułami systemu komputerowego zawierają [1]:

- rozkazy, które określają jakie operacje mają być wykonane
- informacje regulujące czas (taktujące), określające ważność danych i adresów

Reasumując, działanie magistrali systemowej opiera się na następujących zasadach:

- Jeśli pewien moduł (nazwijmy go modułem X) zamierza przesłać dane do drugiego modułu (powiedzmy modułu Y), to musi:
  1. uzyskać dostęp do magistrali
  2. przekazać dane za pośrednictwem magistrali

- Jeśli natomiast pewien moduł (moduł Y) zamierza pozyskać dane od drugiego modułu (modułu X), to musi:
  1. uzyskać dostęp do magistrali
  2. przekazać zapotrzebowanie do modułu X przez odpowiednie linie sterowania i adresowe

NASTĘPNA

5. Pamięć wewnętrzna

(5.1) Podstawowe cechy pamięci

Dla zrozumienia istoty pamięci oraz sposobu ich wykorzystywania przez system komputerowy należy je uszeregować ze względu na następujące własności [1]:

I. **Położenie pamięci:**
   - *pamięć wewnętrzna* - składają się na nią: pamięć główna, pamięć procesora (rejestry, pamięć podręczna)
   - *pamięć zewnętrzna* - urządzenia dostępne przez sterownik wejścia/wyjścia (np. dyski twarde).

II. **Pojemność pamięci:** zwykle wyrażana w bajtach (1 bajt=8 bitów) lub w słowach (zależy jakimi słowami posługuje się procesor, np. 64 bity)

III. **Jednostka transferu** (ang. *transfer unit*) określa ilość danych transmitowanych w jednostce czasu (np. bajt/sekundę). W przypadku pamięci wewnętrznej jednostka transferu z reguły jest równa szerokości szyny danych (ale nie zawsze!).

IV. **Sposób dostępu:**
   - *dostęp sekwencyjny* - pamięć zorganizowana jest za pomocą jednostek danych zwanych rekordami, a dostęp (zapis i odczyt) do wymaganej komórki pamięci (rekordu) jest możliwy tylko w określonej sekwencji liniowej i odbywa się za pośrednictwem rekordów pośrednich (każdy rekord pośredni musi zostać sprawdzony i ewentualnie odrzucony jako niewłaściwy) w wyniku czego czas dostępu do różnych rekordów znacznie się różni (przykład: napędy taśmowe);
   - *dostęp bezpośredni* - tak jak poprzednio zapis i odczyt realizowany jest przez ten sam mechanizm, jednak w tym wypadku poszczególne bloki mają unikatowy adres oparty na lokalizacji fizycznej, a dostęp odbywa się przez bezpośredni dostęp do najbliższego otoczenia, po którym następuje sekwencyjne poszukiwanie, liczenie lub oczekiwanie w celu osiągnięcia szukanej lokalizacji, do której czas dostępu jest zmienny (przykład: pamięci dyskowe);
   - *dostęp swobodny* - każdy rekord pamięci ma unikatowy, fizycznie wbudowany mechanizm sterowania, a czas dostępu do kolejnych rekordów nie zależy od poprzednich operacji dostępu i jest stały, dzięki czemu każdy rekord pamięci jest dostępny bezpośrednio (przykład: pamięć główna);
   - *dostęp skojarzeniowy* - rodzaj dostępu swobodnego, umożliwiający porównanie i badanie zgodności wybranych bitów wewnątrz słowa, przy czym operacja ta wykonywana jest dla wszystkich słów jednocześnie, słowo jest wyprowadzane na podstawie nie tylko adresu, ale i części swojej zawartości (przykład: pamięć podręczna);

V. **Wydajność:**
   - *czas dostępu* - w przypadku pamięci o dostępie swobodnym jest to czas niezbędny do zrealizowania operacji odczytu/zapisu; w przypadku pozostałych pamięci jest czasem potrzebnym na umieszczenie mechanizmu zapisu/odczytu w żądanym miejscu;
   - *czas cyklu pamięci* - jest sumą czasu dostępu i czasu jaki musi upłynąć zanim będzie możliwy kolejny dostęp (ten dodatkowy czas jest potrzebny do regeneracji danych w komórce pamięci RAM, a także do zaniku sygnałów przejściowych);
   - *Szybkość transferu* - szybkość z jaką dane mogą być wprowadzane do jednostki pamięci lub z niej wyprowadzane (dla pamięci o dostępie swobodnym 1/(czas cyklu));

VI. **Własności fizyczne:**
   - *ulotna/nieulotna>* - w pamięci ulotnej informacja zanika naturalnie lub jest tracona w wyniku odłączenia zasilania;
     w pamięci nieulotnej informacja raz zapisana nie zmienia się niezależnie od zasilania;
   - *wymazywalna/niewymazywalna* zawartość pamięci niewymazywalnej nie może być zmieniana

(pamięć typu ROM);

VII. **Organizacja:** fizyczne uporządkowanie bitów w celu uformowania słów.

(5.2) Pamięci stałe

Pamięć **ROM** (ang. *Read Only Memory*) jak sama nazwa wskazuje jest pamięcią tylko do odczytu, informacja zapisana jest na trwałe, a nowe dane nie mogą być zapisywane. Zaletą pamięci ROM jest to, że program znajduje się cały czas w pamięci głównej i nigdy nie wymaga ładowania z urządzeń pamięci wtórnej.

Swoistą "mutacją" pamięci ROM jest pamięć **EPROM** (optycznie wymazywalna programowalna pamięć stała), która jest odczytywana i zapisywana elektrycznie. Przed operacją zapisu wszystkie komórki zostają skasowane poprzez naświetlenie specjalnego układu (wewnątrz obudowy) promieniowaniem ultrafioletowym [1]. Pamięć EPROM jest droższa od pamięci ROM, jednakże daje nowe możliwości kilkukrotnego zapisu.

Bratem bliźniakiem pamięci EPROM jest **EEPROM** (elektrycznie wymazywalna programowalna pamięć stała). Pamięć ta może być zapisana bez wymazania poprzedniej zawartości: aktualizowane są tylko bajty adresowane.

Kolejną powszechnie stosowaną pamięcią stałą jest **pamięć błyskawiczna** (ang. *flash memory*), która wykorzystuje metodę wymazywania elektrycznego. Cała pamięć może być wymazana w ciągu kilku sekund, ponadto możliwe jest wymazanie zawartości tylko niektórych bloków pamięci, a nie całego układu.

(5.3) Pamięci RAM

Pamięć o dostępie swobodnym **RAM** (ang. *Random Access Memory*) pozwala w stosunkowo łatwy sposób na odczytywanie/zapisywanie danych z/do pamięci. Zapis oraz odczyt odbywają się za pomocą sygnałów elektrycznych. Ważną cechą pamięci RAM jest jej ulotność, pamięć RAM potrzebuje źródła zasilania, a w przypadku jego braku dane ulegają skasowaniu. Pamięć RAM można podzielić na statyczną i dynamiczną.

Pamięć **DRAM** (ang. *Dynamic RAM*) zbudowana jest z kondensatorowych komórek, które przechowują ładunek elektryczny. Obecność lub brak ładunku są interpretowane jako logiczne "1" lub "0". Kondensatory po pewnym czasie rozładowują się, dlatego też konieczne jest okresowe odświeżanie ładunku (czyli zachowanie danych). Pamięć DRAM jest wykorzystywana w komputerach jako pamięć główna.

Pamięć **SRAM** (ang. *Static RAM*) przechowuje wartości binarne wykorzystując mechanizmy przerzutników. SRAM przechowuje dane dopóty, dopóki jest zasilany. Zalety: duża szybkość, brak konieczności odświeżania. Wada: wysoka cena.

NASTĘPNA

6. Pamięć masowa

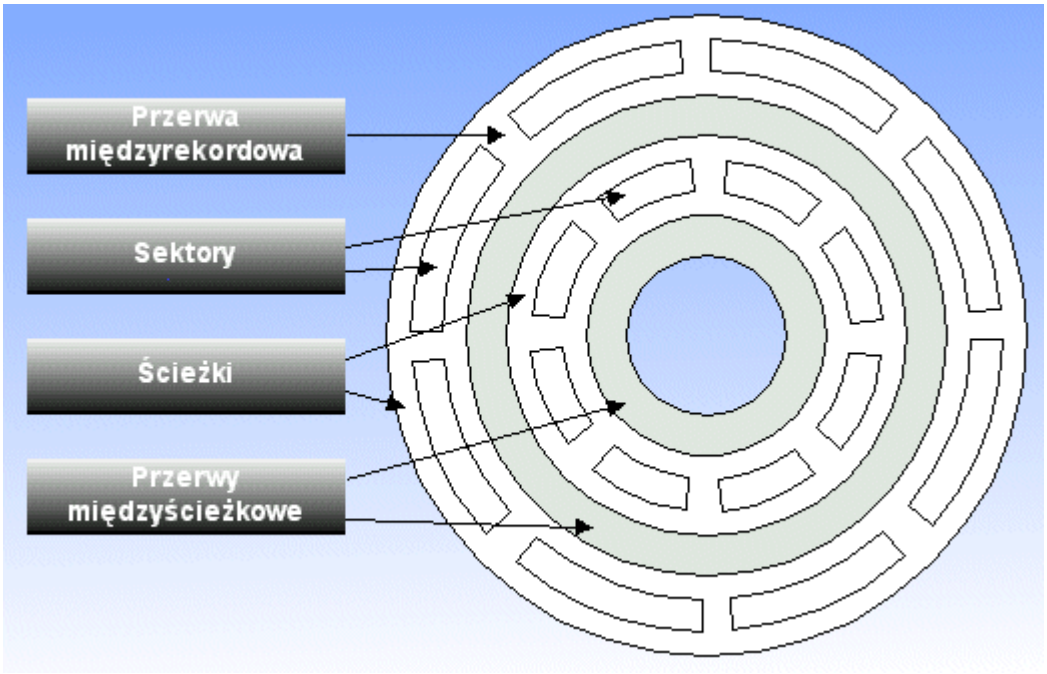
(6.1) Dysk magnetyczny

Dysk magnetyczny ma kształt okrągłej płyty i pokryty jest materiałem magnetycznym. Dane są na nim zapisywane i odczytywane za pomocą głowicy, którą stanowi w tym wypadku cewka elektryczna. W czasie operacji odczytu/zapisu wiruje dysk, zaś głowica pozostaje nieruchoma.

Podczas zapisu wykorzystywane jest pole magnetyczne, wytwarzane przez prąd płynący w cewce. Po nadejściu żądania zapisu, do głowicy wysyłane są impulsy, które powodują zapisanie określonych wzorów magnetycznych na powierzchni płyty, znajdującej się nad głowicą (zależnie od polaryzacji głowicy wzory te są różne). Odczyt polega na wykorzystaniu zjawiska przepływu prądu elektrycznego pod wpływem pola magnetycznego wirującego, względem głowicy dysku. Tak więc dane odczytane rozpoznawane są dlatego, że generowany jest prąd tej samej biegunowości, jak proąd użyty przy zapisie.

Organizacja i formatowanie danych

Organizacja danych na płycie magnetycznej ma postać koncentrycznego zespołu pierścieni, nazywanych **ścieżkami**, z których każda ma taką samą szerokość jak głowica (patrz rysunek 1.6.). Sąsiednie ścieżki są od siebie oddzielone, dzięki czemu błędy wynikające z niewłaściwego ustawienia głowicy lub interferencji pola magnetycznego zostają zminimalizowane. Aby uprościć układy elektroniczne współpracujące z głowicą, na każdej ścieżce przechowywana jest taka sama liczba bitów, a wielkość tę nazywa się **gęstością** i wyraża w bitach na cal.



Rysunek 1.6. Organizacja dysku magnetycznego

Ponieważ zapis i odczyt danych z dysku odbywa się pewnymi blokami, dane są przechowywane także w postaci bloków nazywanych **sektorami**. Na ścieżkę przypadać może do 100 sektorów, a ich długość może być zmienna lub ustalona. Poszczególne sektory wewnątrz ścieżek na dysku magnetycznym są identyfikowane dzięki dodatkowym danym kontrolnym, zapisywanym na tymże dysku. Te dodatkowe dane pozwalają na określenie początku i końca każdego sektora względem sektora startowego.

Jednym z ważniejszych parametrów jest szybkość dysku, którą określają dwa czynniki:

1. **Tempo przesyłania** (ang. *transfer rate*) - oznacza szybkość, z jaką dane są transmitowane pomiędzy dyskiem a komputerem.

2. **Czas ustalania położenia głowicy** (ang. *positioning time*, nazywany też ang. *random access time*). Na wielkość tę składają się:

    - *czas wyszukiwania* (ang. *seek time*), czyli czas przesuwania głowicy do odpowiedniego cylindra (cylinder jest to kilka ścieżek umieszczonych fizycznie nad sobą na kilku talerzach dyskowych)
    - *opóźnienie obrotowe*, czyli czas potrzebny na to, aby odpowiedni sektor, przeszedł nad głowicą.

Dyski magnetyczne współpracują zarówno z głowicami nieruchomymi (jedna głowica zapisu/odczytu na jedną ścieżkę) oraz z głowicami ruchomymi - istnieje wtedy tylko jedna głowica zapisu/odczytu na talerz (np. współczesne dyski twarde).

(6.2) Pamięć optyczna

Dyski CD i CD-ROM są dziś najpowszechniej wykorzystywanym nośnikiem informacji. Płyty CD i CD-ROM są wykonane w ten sam sposób: dysk składa się z czterech warstw: warstwy nośnej z poliwęglanu (plastik), warstwy barwnika (topi się podczas zapisu), warstwy dobrze odbijającej światło (np. aluminium) i lakierowej warstwy ochronnej. Płyta CD posiada spiralny rowek prowadzący (ang. *groove*), który naprowadza laser, wskazuje mu drogę. Informacja zarejestrowana cyfrowo nanoszona jest, w postaci małych zagłębień (ang. *pit*), na powierzchnię odbijającą. Po raz pierwszy wykonuje się to za pomocą dobrze zogniskowanego światła lasera o dużej mocy (podczas zapisu barwnik wytapia się, odsłaniając odbijającą światło powierzchnię) - tak powstaje dysk wzorcowy, który służy jako matryca do tłoczenia kopii [1]. Powierzchnia z naniesioną informacją jest zabezpieczona bezbarwnym lakierem.

Odczyt z płyty odbywa się z wykorzystaniem lasera małej mocy wbudowanego w napęd CD. Laser świeci na wirującą płytę (wprawianą w ruch przez silniczek napędu), pokrytą przezroczystą warstwą lakieru. Zmiana natężenia odbitego światła, zarejestrowana przez komórki światłoczułe, oznacza, że promień napotkał zagłębienie, a więc daną, która następnie zostaje przekonwertowana na sygnał cyfrowy.

Aby umożliwić stałą prędkość odczytu danych przez laser, stosuje się napędy, które obracają krążki [1]:

- **ze stałą prędkością kątową** (ang. *constant angular velocity - CAV*).
  Dysk jest podzielony na pewną liczbę sektorów oraz na szereg koncentrycznych ścieżek. Zaletą tego rozwiązania jest łatwe adresowanie z użyciem ścieżki i sektora. Także przesunięcie głowicy pod określony adres wymaga niewielkiego jej ruchu i oczekiwania aż szukany sektor znajdzie się nad głowicą. Na ścieżkach wewnętrznych i zewnętrznych można umieścić taką samą liczbę danych, dlatego też metoda ta nie znalazła szerszego zastosowania w napędach CD-ROM.
- **ze stałą prędkością liniową** (ang. *constant linear velocity - CLV*).

  Dysk obraca się więc wolniej podczas odczytu danych z zewnętrznych sektorów płyty, zaś w pobliżu jej środka "przyspiesza". Taka konstrukcja sprawia, że pojemność ścieżek znajdujących się na obrzeżach dysku znacznie wzrasta.

Dane na dysku CD-ROM, podobnie jak dla dysków magnetycznych, są zorganizowane w bloki.

Bloki te składają się z pól [1]:

- *Sync* (synchronizacja) - 12 bajtowe pole identyfikujące początek bloku (składa się z bajtu 0, 10 bajtów 1 i bajtu 0).
- *Header* (nagłówek) - 4 bajtowe pole zawierające adres bloku i bajt trybu. Tryb 0 to czyste pole danych, tryb 1 - wykorzystanie kodu korekcyjnego dla 2048 bajtów danych, tryb 2 to 2236 bajtów danych bez kodu korekcji.
- *Data* (dane) - dane to dane, ale do 2048 bajtów.
- *Auxiliary* (pomocnicze) - dodatkowe dane użytkownika w trybie 2 lub 288 bajtowy kod korekcyjny w trybie 1.

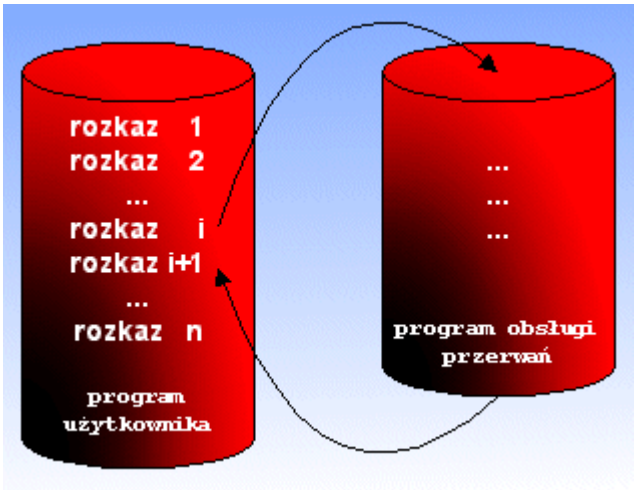
Zależnie od gęstości ścieżek płyty CD-ROM umożliwiają zapis od 550 do 870 MB danych.

NASTĘPNA

7. Przerwania

Przerwania są ważnym elementem funkcjonowania systemu komputerowego. Nazwa odnosi się do tego, że przerwania mają w zwyczaju przerywać normalną pracę programów. Zapytacie: po co? Cierpliwości, już niedługo wszystko się wyjaśni.

A oto jak w ogólnym zarysie działają przerwania: wyobraźmy sobie, że pewne urządzenie zewnętrzne jest gotowe do obsługi (czyli np. przyjmowania danych od procesora). Wtedy moduł I/O (dołączony do tego urządzenia) wysyła sygnał **żądania przerwania** do procesora. W odpowiedzi procesor zawiesza działanie bieżącego programu (rysunek 1.7.) i wykonuje skok do programu obsługującego to urządzenie, czyli **programu obsługi przerwania** (ang. *interrupt handler*). Po obsłużeniu urządzenia następuje powrót do programu, który został przerwany [1]. Pamiętajmy jeszcze, że zawieszenie programu użytkownika, a następnie powrót do miejsca, gdzie nastąpiło przerwanie jego działania, realizuje procesor i system operacyjny.



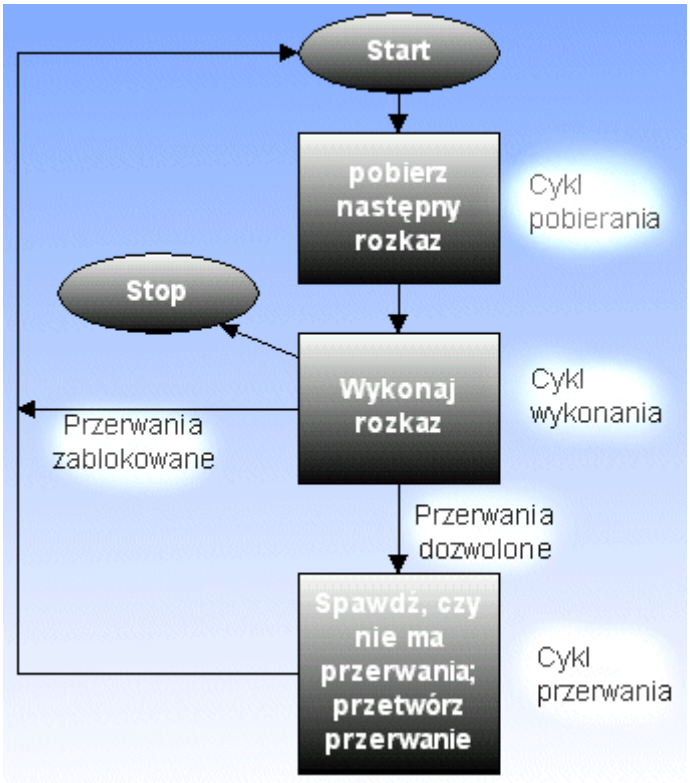
Rysunek 1.7. Przekazywanie sterowania przez przerwania

Ponieważ przerwania powinny być obsługiwane szybko, system posługuje się tablicą wskaźników do procedur obsługujących przerwania. Owa tablica nazywana jest **wektorem przerwań** (ang. *interrupt vector*) i jest indeksowana jednoznacznym numerem urządzenia.

Występowanie przerwań modyfikuje nieco schemat cyklu rozkazowego (patrz rysunek 1.8.). Do cyklu rozkazu dodawany jest **cykl przerwania**. W trakcie cyklu przerwania procesor sprawdza, czy nie pojawiły się jakieś przerwania. Jeśli przerwania nie są wykonywane, procesor pobiera następny rozkaz danego programu. Jeśli natomiast wystąpi zgłoszenie przerwania procesor [1]:

1. Zawiesza wykonywanie bieżącego programu i zachowuje jego kontekst. Adres następnego rozkazu (zawartość licznika), a także dane zawarte w rejestrach zrzucane są na stos systemowy.
2. Ustawia licznik na początkowy adres programu obsługi przerwań

Następnie procesor pobiera pierwszy rozkaz z programu obsługi przerwań, który podejmuje niezbędne do obsłużenia przerwania działania.



Rysunek 1.8. Cykl rozkazu z przerwaniami

Warto wspomnieć, że zaawansowane architektury przerwań pozwalają na obsługę nowego przerwania przed zakończeniem poprzedniego. System operacyjny korzysta ze specjalnego schematu priorytetów, który określa jak ważna jest obsługa danego urządzenia. Przerwanie o wyższym priorytecie będzie obsługiwane również wtedy, gdy jest w trakcie wykonywane przerwanie o priorytecie niższym, zaś przerwania o tym samym lub niższym priorytecie będą *maskowane* (ang. *masked interrupt*).

NASTĘPNA

8. Obsługa wejścia-wyjścia

Istnieją trzy podstawowe sposoby realizacji operacji wejścia-wyjścia (ang. *input-output* - IO):

- **Programowanego wejścia-wyjścia** - komunikacja odbywa się pomiędzy procesorem a modułem I/O. Procesor za pośrednictwem specjalnego programu ma możliwość sterowania operacją we-wy, wysyłania rozkazu zapisu/odczytu i transferu danych. Wadą tego rozwiązania jest fakt, że procesor musi czekać na zakończenie operacji wejścia-wyjścia przez moduł I/O.

- **Wejście-wyjście sterowane przerwaniami** - procesor wydaje rozkaz I/O, później zajmuje się wykonywaniem innych rozkazów, gdy moduł I/O zakończy wykonywanie swojego zadania przerywa czynność procesora.

  W obu powyższych rozwiązaniach procesor odpowiada za odczyt danych z pamięci głównej i za zapis tych danych.

- **Bezpośredni dostęp do pamięci** (ang. *direct memory access - DMA)*) - tryb ten charakteryzuje się tym, że moduł I/O i pamięć główna wymieniają dane bez udziału procesora.

(8.1) Programowe wejście-wyjście

Urządzenia wejścia-wyjścia są zwykle połączone z systemem poprzez moduły wejścia-wyjścia. Oczywiście każde urządzenie posiada unikatowy identyfikator i adres, który wystawia procesor chcąc odwołać się do urządzenia. Fakt, że procesor wystawia adres urządzenia zmusza moduły wejścia-wyjścia do jego identyfikacji.

Jeżeli procesor, pamięć główna i moduły wejścia-wyjścia używają wspólnej magistrali możliwe są dwa tryby adresowania [1]:

1. **Odwzorowany w pamięci** - ta sama przestrzeń adresowa przeznaczona jest dla komórek pamięci i urządzeń I/O. Procesor wykonuje takie same instrukcje maszynowe niezależnie od tego, czy odwołuje się do pamięci, czy do urządzeń I/O. Tak więc, na magistrali potrzebna jest tylko jedna linia zapisu i jedna linia odczytu.

2. **Izolowane wejście-wyjście** - przestrzeń adresowa jest podzielona - inną pulę adresów mają urządzenia I/O, a inną komórki pamięci. Tak więc, część linii magistrali wykorzystywana jest do odczytu/zapisu komórek pamięci, a część do adresowania modułów I/O, dzięki czemu możliwe jest wykorzystanie całej przestrzeni adresowej zarówno przez pamięć jak i urządzenia I/O.

(8.2) WE-WY sterowane przerwaniami

Wadą programowanego wejścia-wyjścia był długi czas oczekiwania przez procesor na gotowość wybranego modułu I/O. Dlatego też wykorzystuje się wejście-wyjście sterowane przerwaniami, którego działanie można przedstawić następująco [1]:

1. Procesor wydaje modułowi rozkaz wejścia-wyjścia, po czym przechodzi do wykonywania "ciekawszych" zadań (np. do obliczania całki Fouriera).
2. Kiedy moduł I/O jest gotowy do wymiany danych z procesorem (czyli np. wczytał już dane z urządzenia peryferyjnego), wysyła żądanie obsługi przerwania i czeka aż procesor będzie gotowy do wymiany danych.
3. Gdy następuje przerwanie od modułu I/O procesor zachowuje kontekst bieżącego programu

(np. obliczania całki Fouriera) tj. min. zawartość licznika programu i rejestrów procesora i zgłasza chęć transmisji.
4. Moduł I/O wystawia dane na magistralę, po czym jest gotowy do kolejnej operacji wejścia-wyjścia.
5. Procesor transmituje dane, a następnie odnawia kontekst programu (np. całki Fouriera) i wznawia jego działanie.

Wejście-wyjście sterowane przerwaniami jest wydajniejsze niż programowane, jednakże i w tym rozwiązaniu traci się dużo czasu procesora, który musi nadzorować każde słowo transmisji pomiędzy pamięcią a modułami I/O.

(8.3) DMA

Metoda **bezpośredniego dostępu do pamięci** DMA (ang. *Direct Memory Access*) stosowana jest zawsze tam, gdzie muszą być przenoszone duże ilości danych przy minimalnym obciążeniu procesora.

Metoda ta wymaga zainstalowania dodatkowego modułu DMA na magistrali systemowej. Moduł ten potrafi przejmować od procesora sterowanie systemem. Dochodzi to tego, gdy procesor chce odczytać/zapisać dane. Wydaje wtedy rozkaz modułowi DMA, wysyłając następujące informacje [1]:

- czy potrzebny jest odczyt, czy zapis?
- adres urządzenia I/O wymaganego do komunikacji;
- adres początkowej komórki pamięci, do której nastąpi zapis/odczyt;
- liczbę słów, które należy odczytać/zapisać

Procesor powraca do wykonywania swoich zadań (np. programu użytkowego), zaś moduł DMA zajmuje się transmisją danych pomiędzy modułem I/O a pamięcią z pominięciem procesora. Gdy moduł DMA zakończy przesył danych, wysyła sygnał przerwania do procesora. Procesor jest więc używany tylko na początku i końcu transmisji.

Jak już wspomnieliśmy moduł DMA przejmuje działanie procesora w komunikacji pomiędzy pamięcią a modułami I/O. Dlatego też konieczny jest dostęp do magistrali. I tu pojawiają się dwie opcje:

1. moduł DMA używa magistrali tylko wówczas, gdy nie potrzebuje jej procesor;
2. moduł DMA wymusza czasowe zawieszenie operacji procesora;

Ta ostatnia metoda jest bardziej popularna i określa się ją mianem **wykradania cyklu** ze względu na fakt, że moduł DMA zajmuje cykl magistrali [1].

Pamiętajmy, że "wykradanie cyklu" nie jest równoważne przerwaniu - procesor zatrzymuje się na jeden cykl magistrali bez zachowywania kontekstu. Oczywiście "wykradanie cyklu" sprawia, że procesor pracuje nieco wolniej, ale dla dużych bloków danych metoda DMA jest daleko bardziej efektywna niż opisane wcześniej *programowane wejście-wyjście* i *wejście-wyjście sterowane przerwaniami*.

NASTĘPNA

9. Mechanizmy ochrony wejścia-wyjścia, pamięci, jednostki centralnej

(9.1) Ochrona wejścia-wyjścia

Działania systemu mogą zostać zakłócone - zdają sobie dobrze z tego sprawę osoby, które pisały programy w asemblerze procesora 8051 - przez program użytkownika, który wyda nieodpowiedni rozkaz I/O.

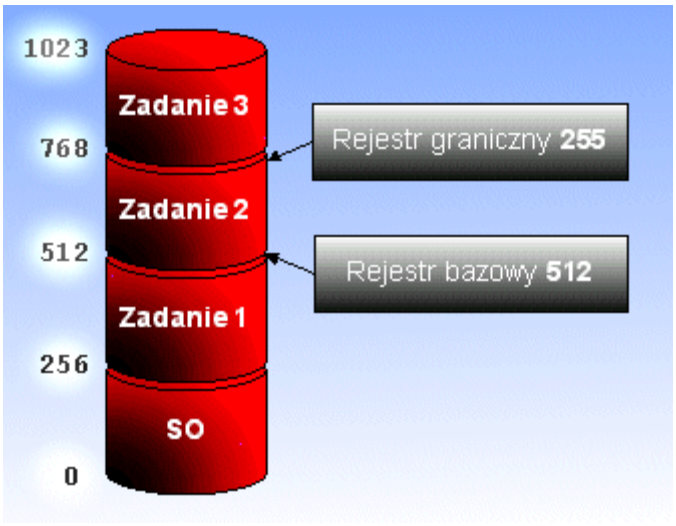
Niedozwolone operacje I/O prowadzą do nieokreślonych zachowań systemu komputerowego. Dlatego też, aby ustrzec się przed wykonaniem niedozwolonej operacji wejścia-wyjścia przyjęto, że wszystkie rozkazy I/O są uprzywilejowane, zaś użytkownicy mogą z nich korzystać jedynie za pośrednictwem systemu operacyjnego. Uzyskanie pełnej ochrony wejścia-wyjścia wymaga uniemożliwienia przejęcia przez program użytkownika kontroli nad systemem komputerowym w trybie systemowym (ang. *system mode*). Tryb systemowy wykorzystywany jest przez system operacyjny i charakteryzuje się tym, że pozwala na uzyskanie pełnej kontroli nad systemem komputerowym, w tym również umożliwia wykonanie uprzywilejowanych instrukcji (w trybie użytkownika jest to niemożliwe) [2].

(9.2) Ochrona pamięci

Podstawowymi jednostkami pamięci, które muszą być chronione są wektor przerwań oraz systemowe procedury obsługi przerwań. Zaniedbanie tej ochrony mogłoby doprowadzić do sytuacji, w której program użytkownika zastąpiłby systemowe procedury obsługi przerwań skokami do własnego obszaru, przechwytując w ten sposób sterowanie od procedury obsługi przerwania, pracującej w trybie systemowym [2]. To doprowadzić mogłoby do nieprawidłowego działania systemu komputerowego oraz np. przebiegu buforowania.

Bezpieczeństwo wektora przerwań i systemowych procedur obsługi przerwań jest bardzo istotne. Ale co zrobić, gdy chcemy chronić całą pamięć systemu operacyjnego przed wpływem "z zewnątrz" oraz obszary pamięci kilku różnych programów użytkowych? Otóż z pomocą przychodzą rozwiązania sprzętowe.

Najpopularniejsza metoda ochrony pamięci programu opiera się na wykorzystaniu dwóch rejestrów: bazowego (ang. *base*) i granicznego (ang. *limit*). Rejestr bazowy przechowuje podstawowy, najmniejszy, dopuszczalny, fizyczny adres pamięci, zaś rejestr graniczny zawiera rozmiar pamięci (patrz rysunek 1.9.).



Rysunek 1.9. Rejestr bazowy i graniczny

Dzięki ww. rejestrom dowiadujemy się, jakie są dopuszczalne adresy programu oraz uzyskujemy możliwość ochrony pamięci poza tymi adresami.

Ochronę tę sprawuje jednostka centralna porównując dla każdego adresu wygenerowanego w trybie użytkownika, jaka jest zawartość opisanych powyżej rejestrów [2]. Jakakolwiek ingerencja wykonana przez program użytkownika do pamięci innego programu lub programu pracującego w trybie systemowym, kończy się wygenerowaniem przez system błędu. Program użytkownika jest więc chroniony przed zmodyfikowaniem kodu lub struktur danych własnych oraz systemu operacyjnego.

(9.3) Ochrona jednostki centralnej

System operacyjny powinien sprawować stałą kontrolę nad komputerem, dlatego też należy zapobiec sytuacji, w której program użytkownika wpadłby w nieskończoną pętlę permanentnie odbierając kontrolę SO. Wykorzystuje się do tego **czasomierz** (ang. *timer*). Timer pozwala na generowanie przerwania po wyznaczonym czasie - czas ten może być stały lub zmienny (przyrostowo o pewien krok).

Tak więc system operacyjny, przed oddaniem sterowania do programu użytkownika, ustawia timer na przerwanie. Gdy czasomierz wywołuje przerwanie, sterowanie wraca do systemu operacyjnego, który decyduje: czy wystąpił błąd, czy też należy dać programowi użytkownika więcej czasu [2]. W ten łatwy sposób można zapobiec zbyt długiemu działaniu programów lub wymusić zakończenie działania programu po upływie określonego czasu.

Czasomierz wykorzystywany jest również do podziału czasu procesora. Umożliwia on przełączanie się programów użytkowych co pewien określony czas (stały lub zmienny), stwarzając w ten sposób wrażenie pracy równoległej.

Wydawać by się mogło, że głównym zadaniem czasomierza powinno być odmierzanie czasu względem pewnej wartości początkowej. To również potrafi czasomierz, ale jest to tylko "skutek uboczny" jego faktycznego przeznaczenia.

NASTĘPNA