Organizacja i architektura komputerów

Wykład nr 1: Wprowadzenie

Piotr Bilski

O wykładowcy

- dr hab. inż. Piotr Bilski, prof. PW
- Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska
- Adresy e-mail:

  pbilski@ire.pw.edu.pl

  pbilski@elka.pw.edu.pl
- Terminy konsultacji:

  po wcześniejszym uzgodnieniu mailowym

Harmonogram zajęć

1. Wprowadzenie, historia komputerów
2. Budowa systemu komputerowego. Magistrala. Cykl rozkazowy
3. Układy logiczne. Arytmetyka komputera
4. Lista rozkazów procesora
5. Sposoby adresowania
6. Struktura procesora
7. Układ sterujący. Mikrooperacje
8. Hierarchia pamięci. Pamięć podręczna
9. Pamięć główna i zewnętrzna
10. Urządzenia wejścia/wyjścia
11. Komputery o zredukowanej liście rozkazów
12. Architektury superskalarne
13. Architektury równoległe
14. Architektura IA-64

Zasady zaliczenia

Egzamin pod koniec semestru (oceniany w skali od 0 do 50 pkt.).

Ocena końcowa obliczana na podstawie punktów z egzaminu:

>= 26 pkt. – ocena 3

>= 31 pkt. – ocena 3,5

>= 36 pkt. – ocena 4

>= 41 pkt. – ocena 4,5

>= 46 pkt. – ocena 5

Zalecana literatura

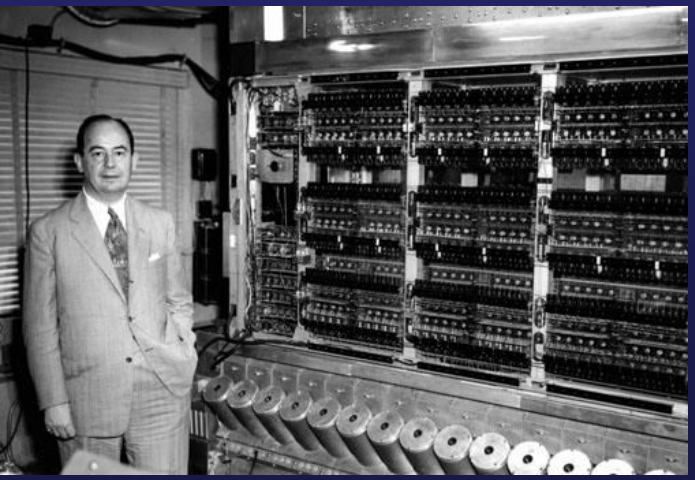
- B.S. Chalk, „Organizacja i architektura komputerów”, WNT, Warszawa, 1998
- W. Stallings, „Organizacja i architektura systemu komputerowego”, WNT, Warszawa, 2004
- M. Morris Mano, „Architektura komputerów”, WNT, Warszawa, 1988
- J. Ogrodzki, „Wstęp do systemów komputerowych”, Oficyna Wydawnicza Politechniki Warszawskiej, 2005
- S. Kozielski, Z. Szczerbiński, „Komputery równoległe”, WNT, Warszawa, 1993
- Z. Pogoda, „Mikroprocesory RISC rodziny PowerPC”, Wydawnictwo Pracowni Komputerowej Jacka Skalmierskiego, Gliwice 1995
- P. Metzger, „Anatomia PC : architektura komputerów zgodnych z IBM PC”, Wyd. Helion, Gliwice 2004

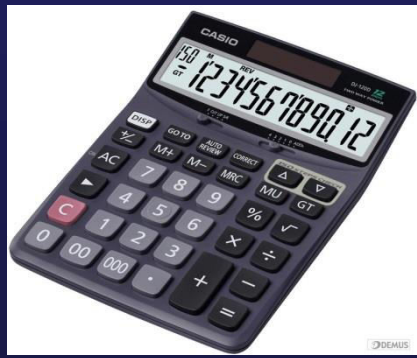
Organizacja a architektura systemu komputerowego

- Organizacja określa jednostki operacyjne i połączenia pomiędzy nimi, stanowiące realizację architektury
- Architektura określa atrybuty (cechy) systemu komputerowego widoczne dla programisty

Definicja komputera

- **Komputer** (ang. *computer*, dawne nazwy: *mózg elektronowy*, *elektroniczna maszyna cyfrowa*, *maszyna matematyczna*) to w najszerszym znaczeniu maszyna licząca, służąca do przetwarzania informacji, które da się zapisać w formie ciągu cyfr, albo sygnału ciągłego.





Klasyfikacja komputerów

Wielkość i złożoność zbioru rozkazów

- RISC
- CISC
- VLIW

Przeznaczenie

- Uniwersalne
- Problemowo-wyspecjalizowane
- Specjalistyczne

Sposób przetwarzania danych

- Szeregowe (skalarne)
- Równoległe
  - Macierzowe
  - Wektorowe
  - Wieloprocesorowe
  - Wielordzeniowe

Szerokość rejestrów/szyny danych

- 8-bitowe
- 32-bitowe
- 16-bitowe
- 64-bitowe

Schemat funkcjonalny komputera

Fazy rozwoju komputerów

1. Komputery oparte na lampach próżniowych (1946-1957)
2. Komputery oparte na tranzystorach (1958-1964)
3. Komputery o strukturze SSI, MSI (1965-1971)
4. Komputery o strukturze LSI (1972-1979)
5. Komputery o strukturze VLSI (1980-83)
6. Komputery o strukturze ULSI (1984-???)
7. Nowe architektury: molekulane, kwantowe, neurokomputery

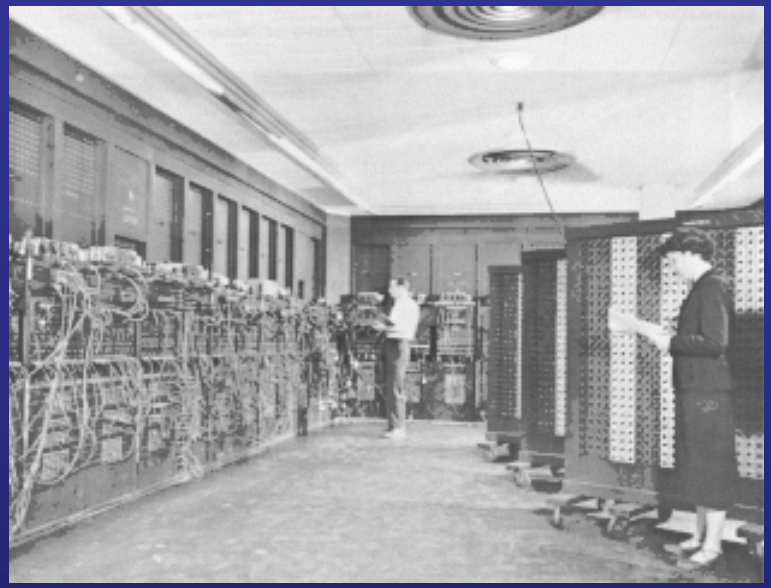
Fazy rozwoju procesorów (Intel)

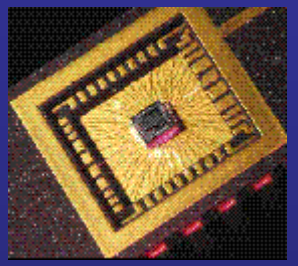
1. Procesory 8-bitowe (8008-8080)
2. Procesory 16-bitowe (8088, 8086, 80188, 80186, 80286)
3. Pierwsze procesory 32-bitowe (80386)
4. Rodzina 486 (80486)
5. Rodzina Pentium (80586)
6. Rodzina Pentium Pro (80686)
7. Rodzina Pentium IV
8. Procesory 64-bitowe (Pentium IV Extreme)
9. Procesory wielordzeniowe (Dual Core, Core2Duo, Core2Quad, X2, X4, i7)

Pentium i PowerPC

- PowerPC:
  - skonstruowany przez konsorcjum IBM-Apple-Motorola
  - Najlepszy procesor RISC
  - Modele: 601, 603, 604, 620, G3, G4
  - Obecnie spotykany w urządzeniach sieciowych, drukarkach (Kyocera) i konsolach (PS3, Nintendo Wii)
- Pentium:
  - skonstruowany przez firmę Intel
  - Klasyczny superskalarny przedstawiciel architektury x86
  - Pentium, Pentium II, Pentium Pro, Pentium IV, IA-64 (64-bitowy!)

ENIAC (J.P. Eckert, J.W. Maulchy - 1946)





- Uznawany za pierwszy komputer na świecie
- Obliczenia w systemie dziesiętnym (brak pamięci)
- Waga – 30 ton, w strukturze 20 tys. lamp próżniowych, 5000 op/s, moc pobierana: 140 kW
- Przeznaczenie: obliczenia dla wojska (balistyka rakiet, wykonalność bomby wodorowej)

Komputery komercyjne (od 1951)

701, 702 (IBM)

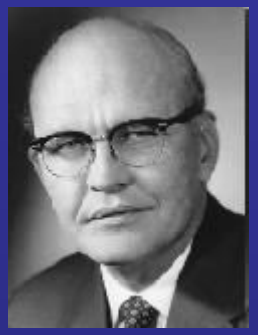
UNIVAC I (Sperry-Rand Corporation)

Cechy:

Jednostka centralna zrealizowana na lampach próżniowych

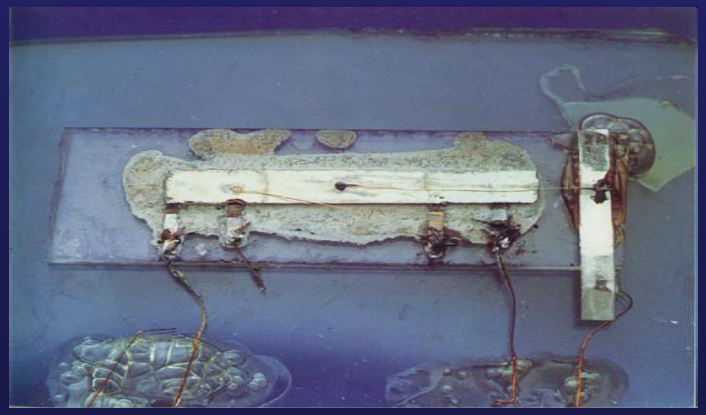
Pamięć operacyjna w postaci krążków ferrytowych lub lampach elektrostatycznych

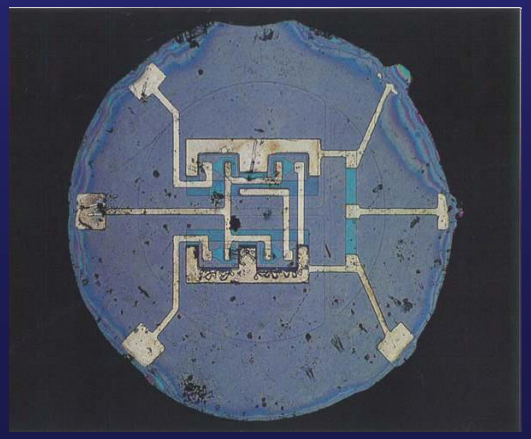
Pierwszy układ scalony (1958, J. S. Kilby, R.N. Noyce)



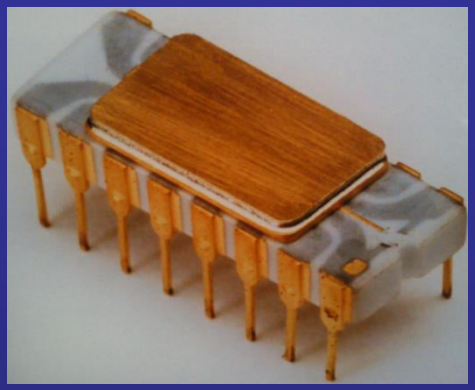
Układ germanowy zawierał pięć elementów (tranzystory, rezystory i kondensatory)

Układ krzemowy zawierał dziewięć elementów (tranzystory i rezystory)



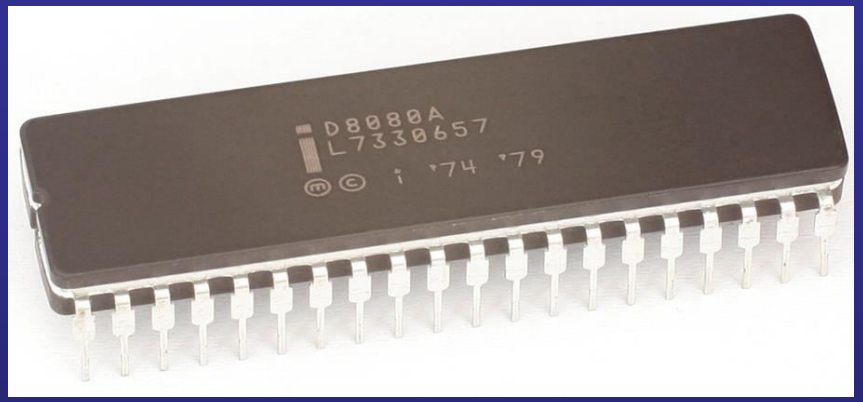


Pierwszy mikroprocesor (1971)



- Opracowany w firmie Intel, oznaczenie 4004 (autor: Ted Hoff)
- Zbudowany z 2300 tranzystorów
- Zaimplementowana operacja dodawania dwóch liczb 4-bitowych
- Taktowany zegarem 100 kHz

Pierwszy mikroprocesor ogólnego przeznaczenia (1974)



- Oznaczenie 8080
- Procesor 8-bitowy
- Częstotliwość zegara: 2 MHz
- 6000 tranzystorów w układzie
- 64 kB adresowalnej pamięci

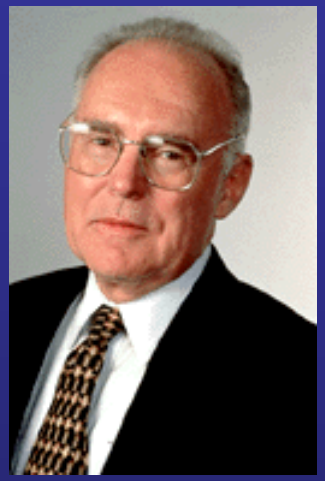
Komputer Apple II (1977)

- Jako pierwszy na świecie wyświetlał kolorową grafikę
- Otwarta architektura (łatwa rozbudowa)
- Procesor MOS 6502 (1MHz do 3 MHz)
- Pamięć RAM 4KB, max. 64 KB
- System operacyjny WOZ Integer Basic

Komputer IBM PC/XT (1981)

- Model 5150
- Procesor Intel 8088 (4,77 MHz) później (w trybie turbo) do 14 MHz, opcjonalnie koprocesor 8087
- Pamięć RAM – pierwotnie 128kB, max. 640 kB
- Magistrala ISA 8-bitowa
- Zastąpiony modelami IBM PC/AT oraz IBM PC/XT/286

Prawo Moore'a (1965)



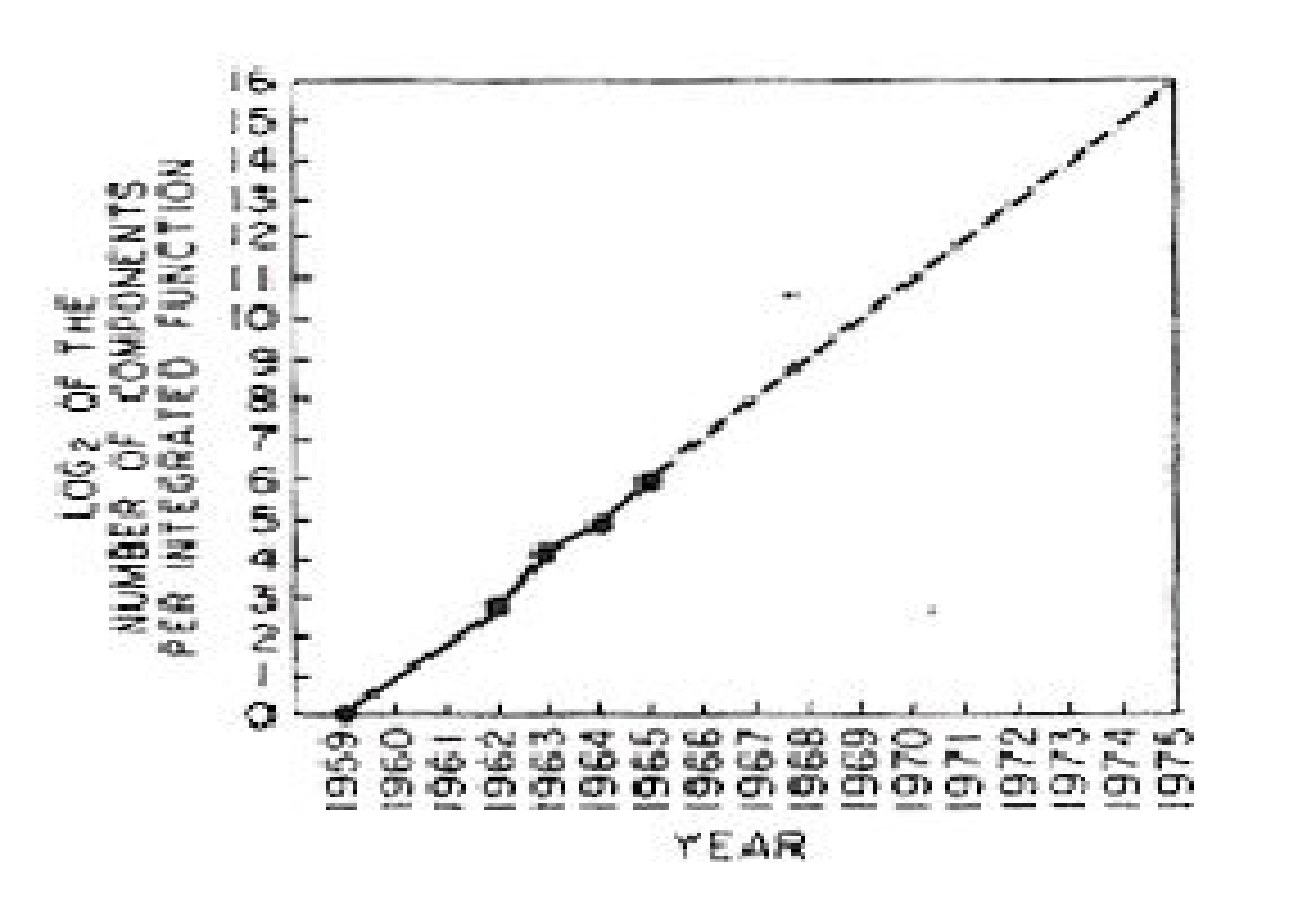
Gordon Moore (ur. 1929, San Francisco, Kalifornia), doktorat z fizyki w 1954 r. Jeden z założycieli korporacji Intel w 1968 r.

„Ekonomicznie optymalna liczba tranzystorów w układzie scalonym będzie się podwajać co 18 miesięcy"

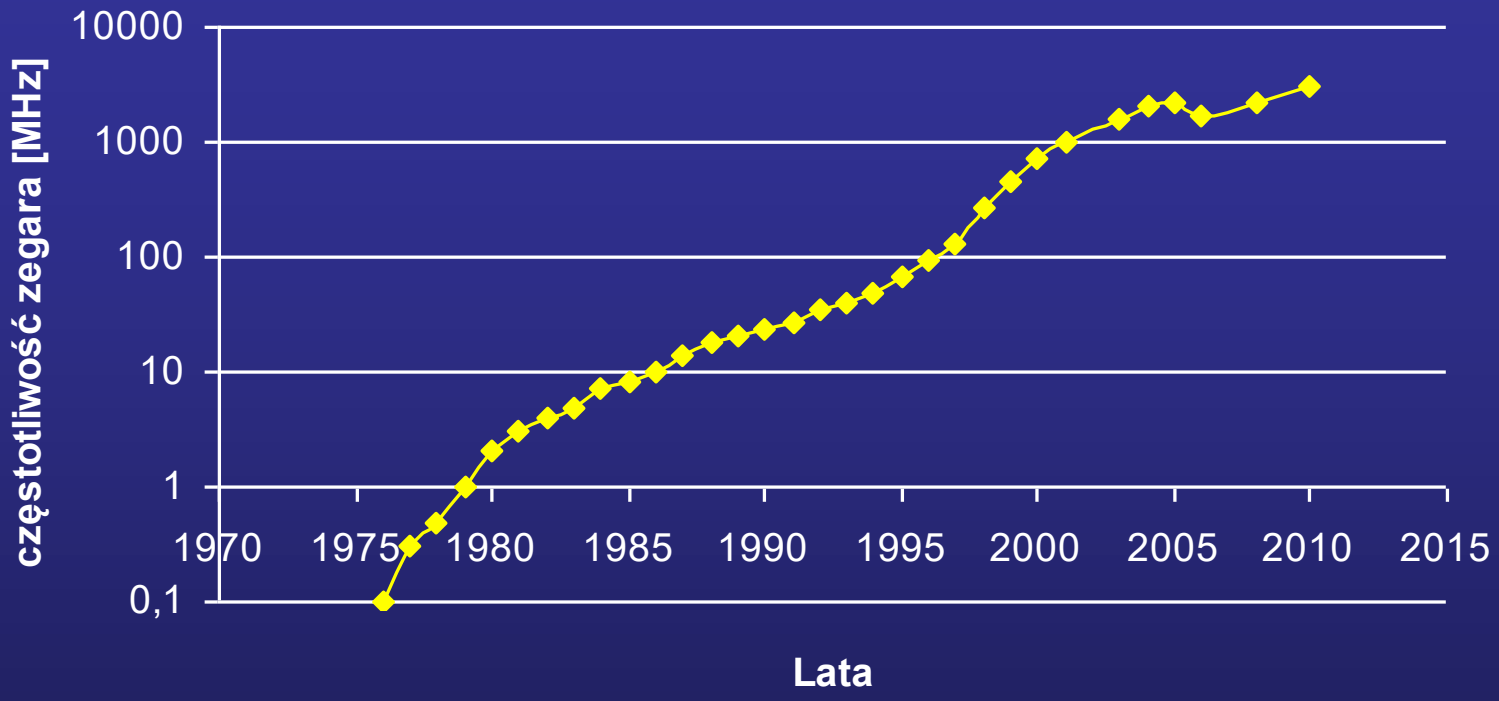
„Moc obliczeniowa mikroprocesorów przy stałym koszcie będzie się podwajać co 18 miesięcy"

Prawo Moore'a (c.d.)

- Oryginalny rysunek z artykułu Moore'a (1965)

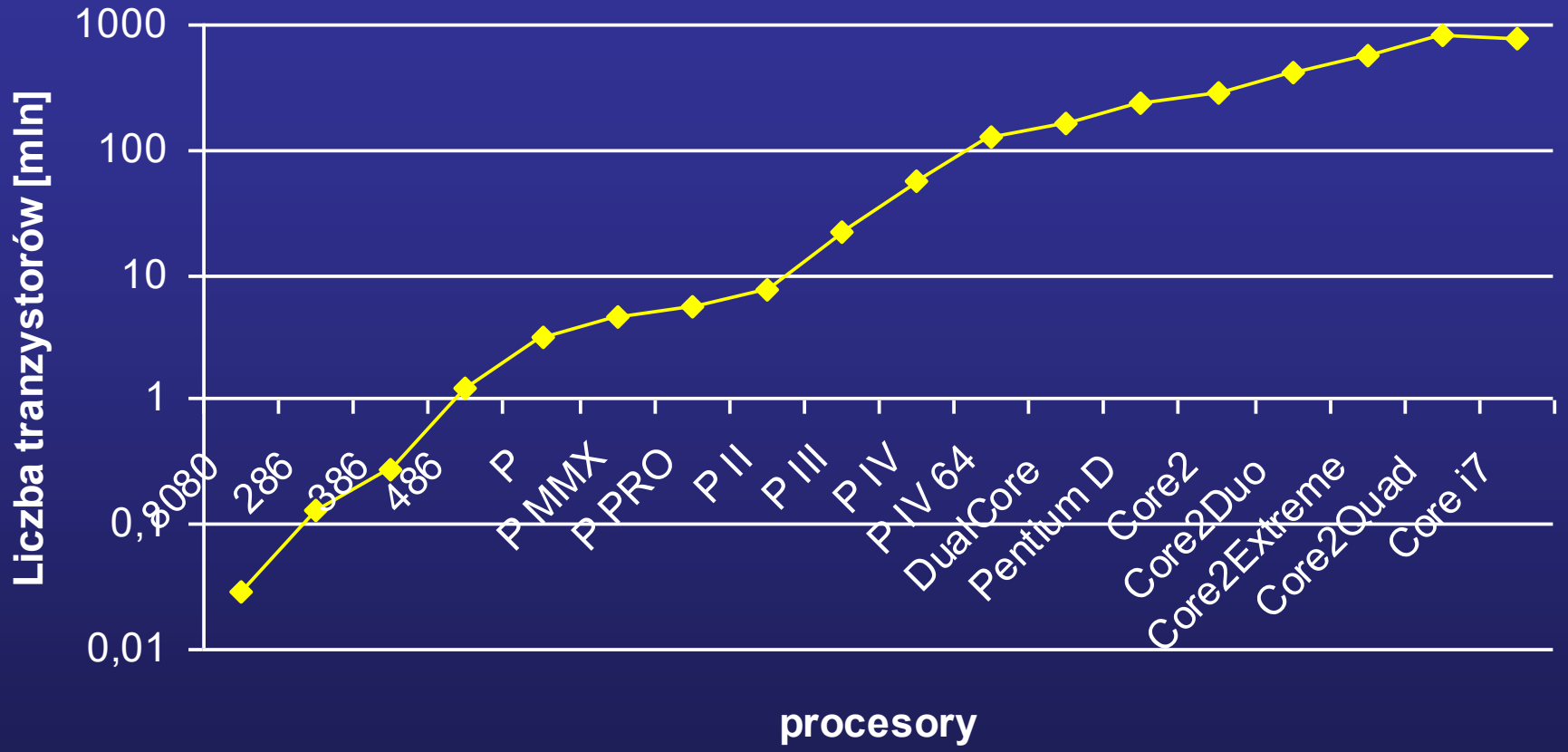


Prawo Moore'a (c.d.)



„Gdyby technologia samochodowa od 1949 r. przyspieszała w takim samym tempie, jak komputerowa, nowoczesny samochód ważyłby 60g, kosztował 40$, miał bagażnik o pojemności1,5 mln L, zużywał 1l paliwa na 600 tys. kilometrów i osiągał prędkość 2 160 000/h"

Wzrost skali integracji w czasie

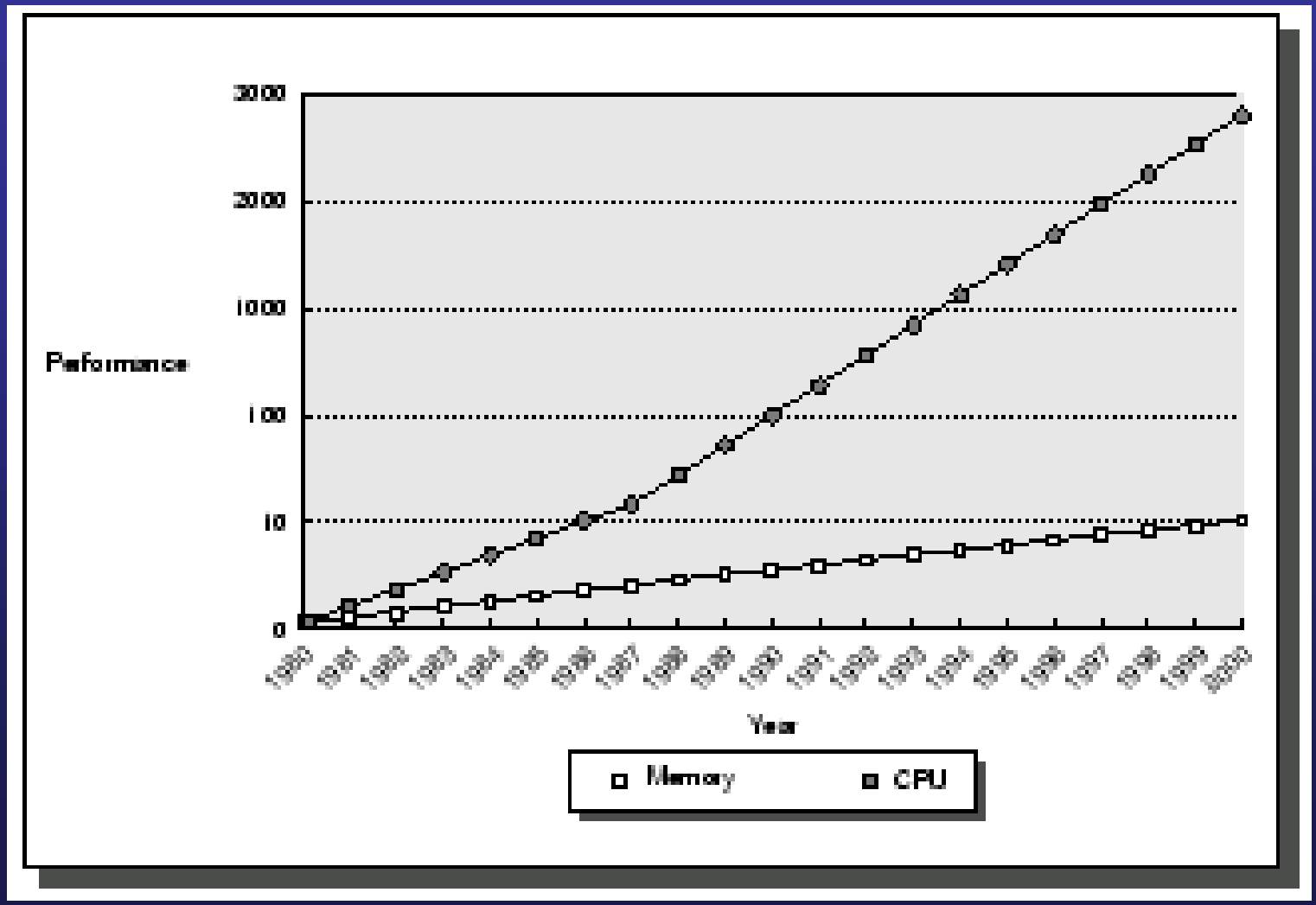


Uwaga: w czasie premiery architektury Core i, równolegle wprowadzono procesory serwerowe Tukwila (następcy procesorów Itanium oraz Itanium 2), które mają maksymalnie 2 mld tranzystorów!!

Przerwa wydajnościowa

- Postęp wydajności procesorów i pamięci nie był równomierny
- Częstotliwości pracy zegara są znacząco większe od częstotliwości pracy pamięci
- Istnieją liczne metody kompensacji tej nierówności:
  - zwiększanie częstotliwości pracy pamięci
  - zwiększanie wielkości pamięci podręcznej
  - modyfikacja kolejności wykonywania rozkazów

Ilustracja przerwy wydajnościowej

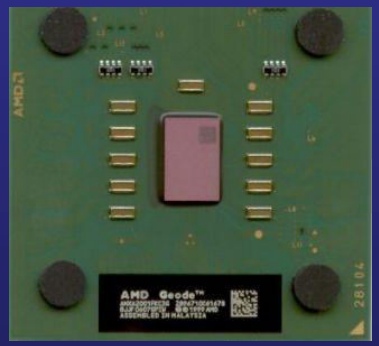


Problem ograniczeń fizycznych

- Rozmiar tranzystorów nie może być zmniejszany w nieskończoność!
- Duży problem stanowi ciepło wydzielane przez procesor (problem chłodzenia!)
- Rdzeń procesora ma kluczowe znaczenie dla obliczeń oraz wydzielanego ciepła

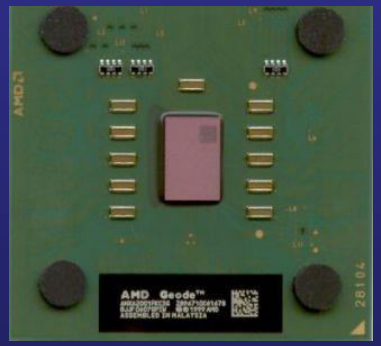
Porównanie architektur jedno- i wieloprocesorowych

Jeden rdzeń

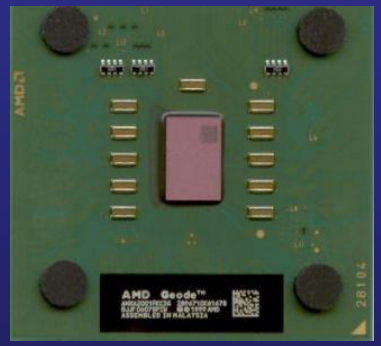


cache

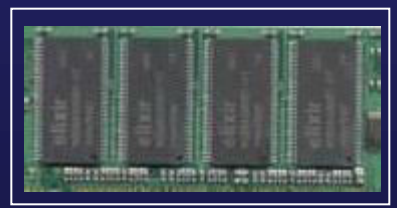
Multiprocesor



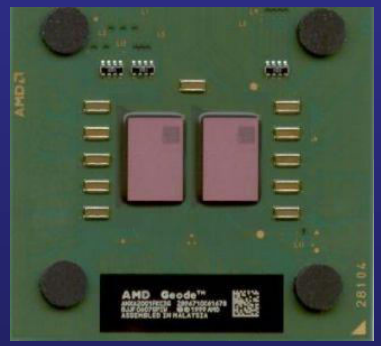
cache



cache



Wiele rdzeni

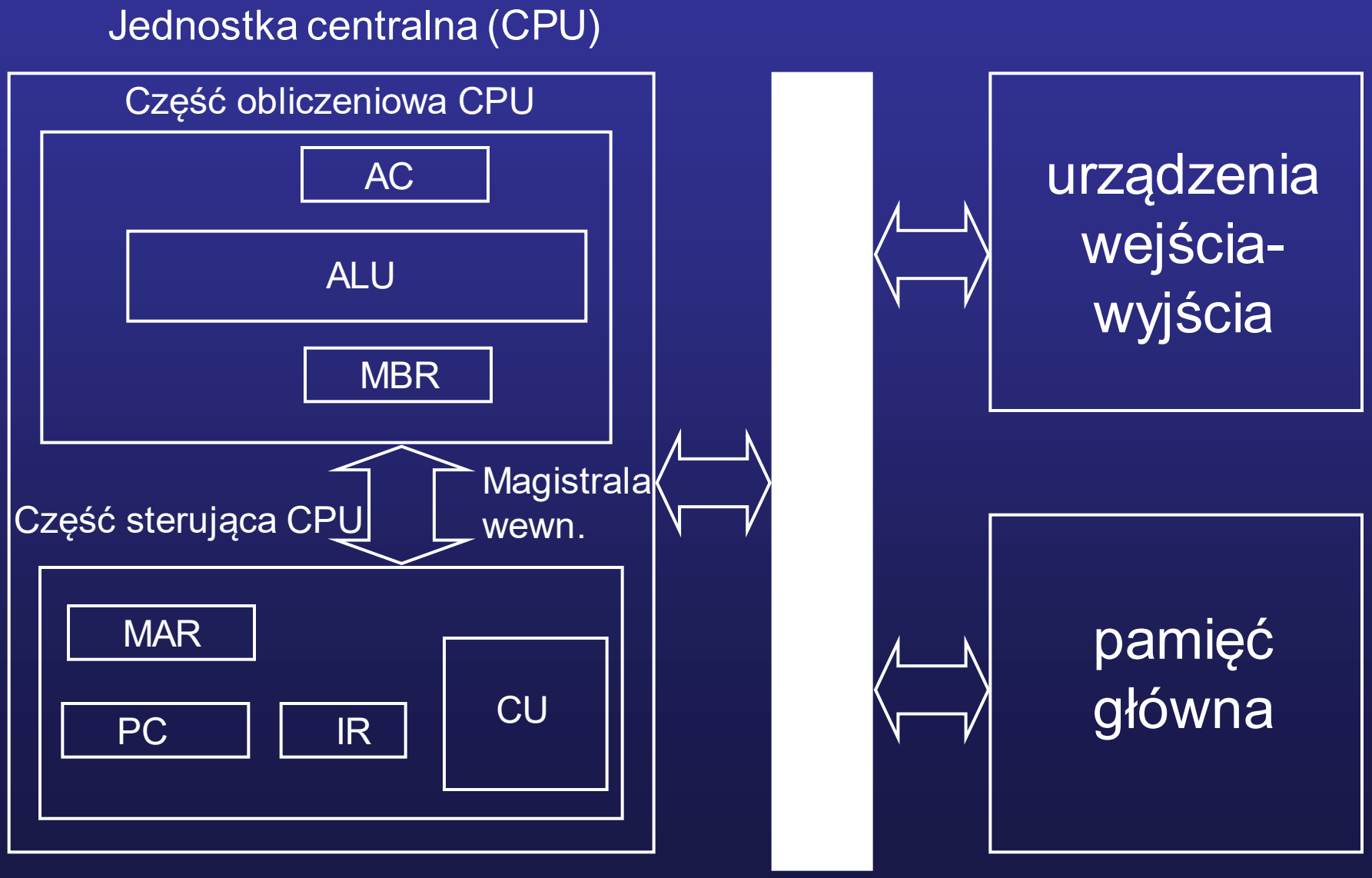


cache

Architektura von Neumanna (1945)

- Uniwersalna architektura, na której opierają się współczesne komputery
- Pierwszy raz zrealizowana w postaci komputera IAS (1952)
- Struktura funkcjonalna:
  - Jednostka centralna składająca się z jednostki arytmetyczno-logicznej i sterującej
  - Pamięć główna do przechowywania danych i rozkazów
  - Urządzenia wejścia-wyjścia

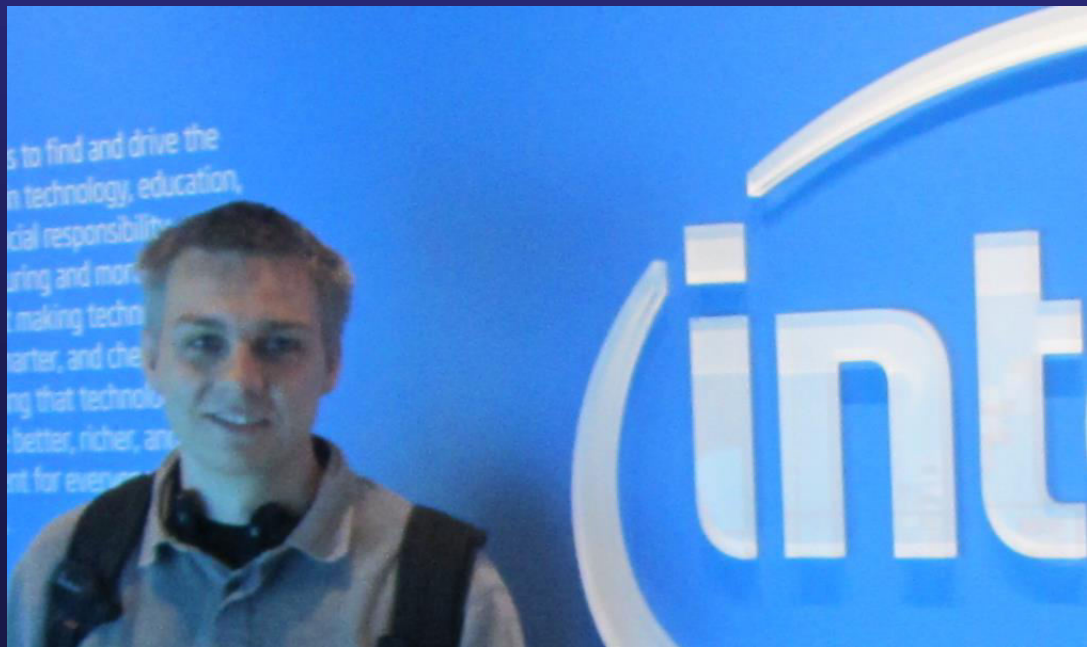
Architektura von Neumanna



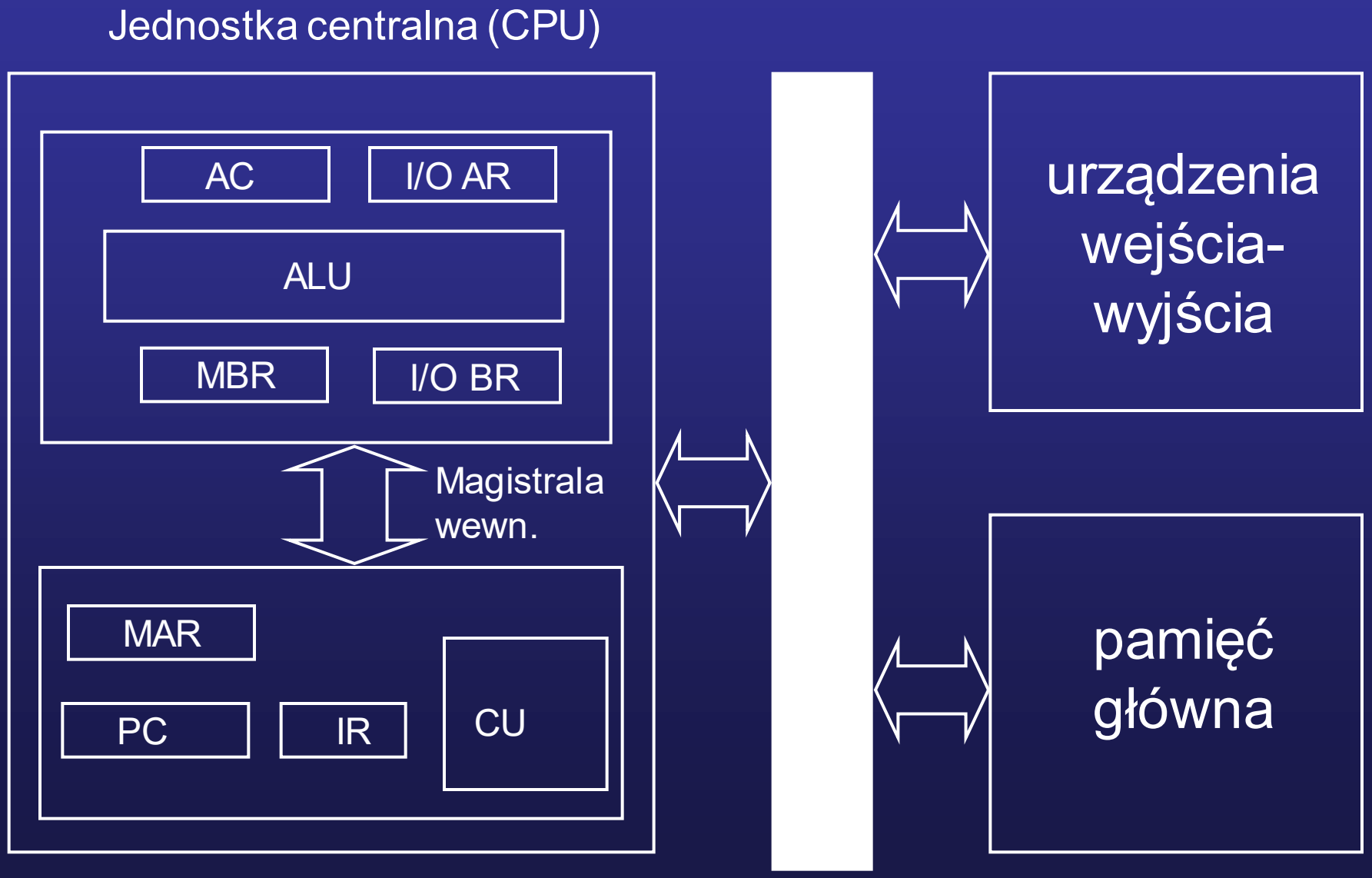
Architektura komputerów

Wykład nr 2: Procesor, cykl programu, przerwania, magistrala systemowa

Piotr Bilski



Maszyna von Neumanna



Pojęcie programu

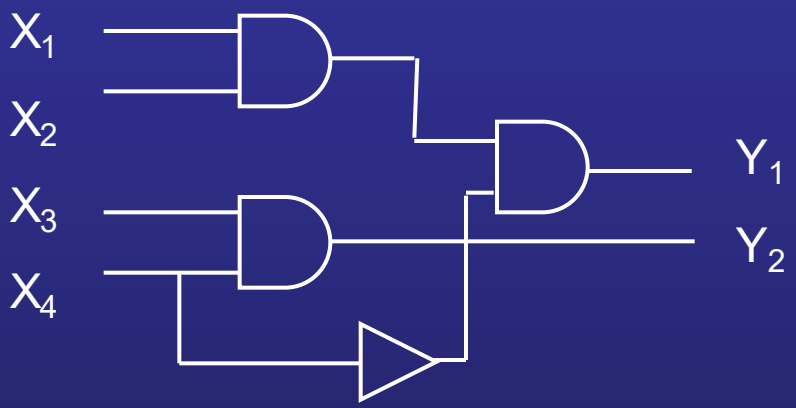
- Programem nazywamy zbiór instrukcji, których wykonanie w określonej kolejności zapewnia odpowiednie przetworzenie informacji.
- Instrukcja (rozkaz) jest słowem maszynowym zawierającym informację o wykonywanej operacji, miejscu w pamięci gdzie znajdują się operandy, rezultat i następny rozkaz.

Wymagania odnośnie systemu komputerowego przetwarzającego program (von Neumann)

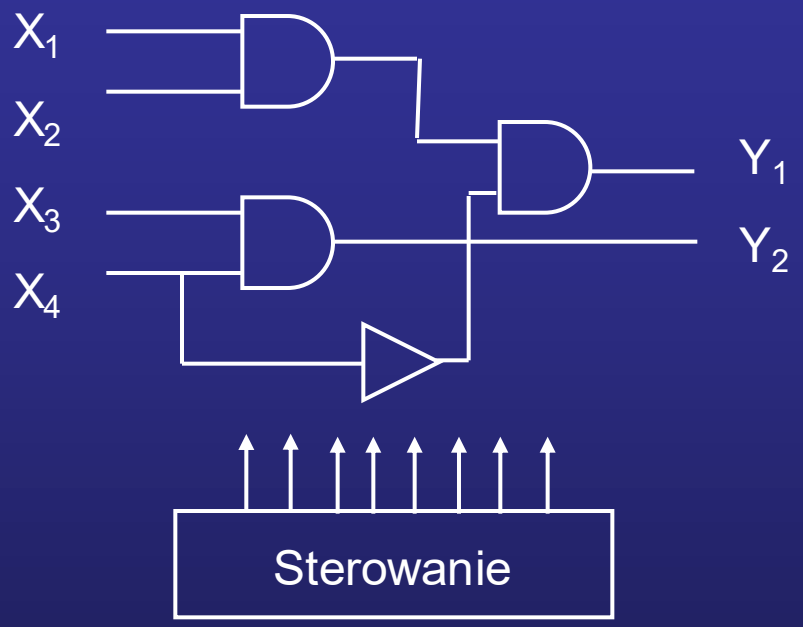
System komputerowy musi:

- Mieć skończoną i funkcjonalnie pełną listę rozkazów;
- Mieć możliwość wprowadzenia programu do systemu komputerowego poprzez urządzenia zewnętrzne i przechowywać instrukcje w pamięci w sposób identyczny jak danych;
- Dane i instrukcje powinny być jednakowo dostępne dla procesora (poprzez jednoznaczne adresy w pamięci);
- Informacja jest przetwarzana dzięki sekwencyjnemu odczytywaniu instrukcji z pamięci komputera i wykonywaniu tych instrukcji w procesorze.

Program vs. „program” sprzętowy



Hard-wired program



0000: move 4

0001: add 5

0010: store 6

0011: stop

Działanie komputera

Język przesłań rejestrowych

- Symbole złożone z wielkich liter oznaczają zawartość
- M – pamięć
- A, MAR itd. – rejestr
- $\leftarrow$ przepisanie
- ( ) – adres
- 0:7 – zakres bitów słowa pamięci lub rejestru, którego dotyczy operacja

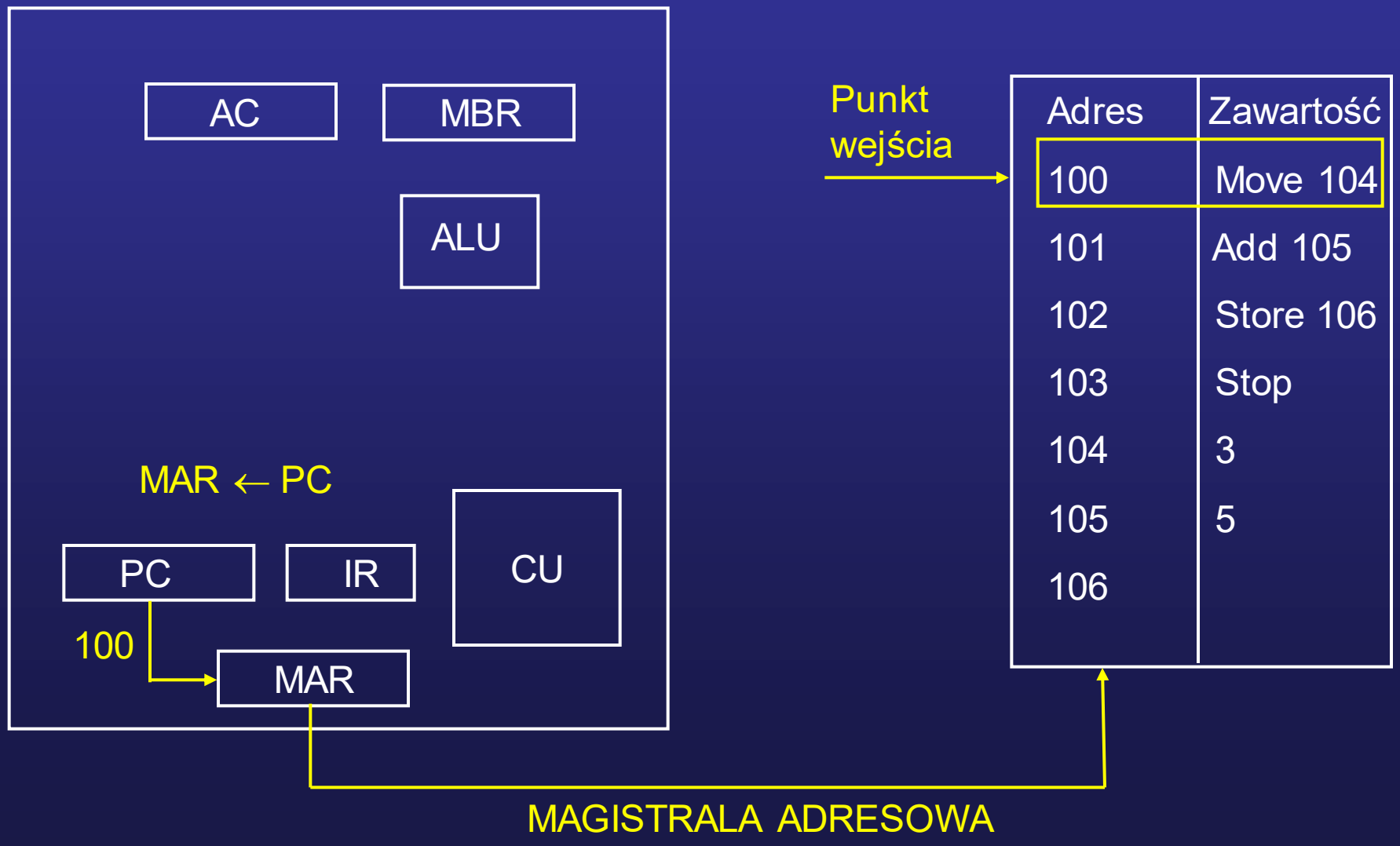
Np.: MAR $\leftarrow$ PC

MBR $\leftarrow$ M(MAR)

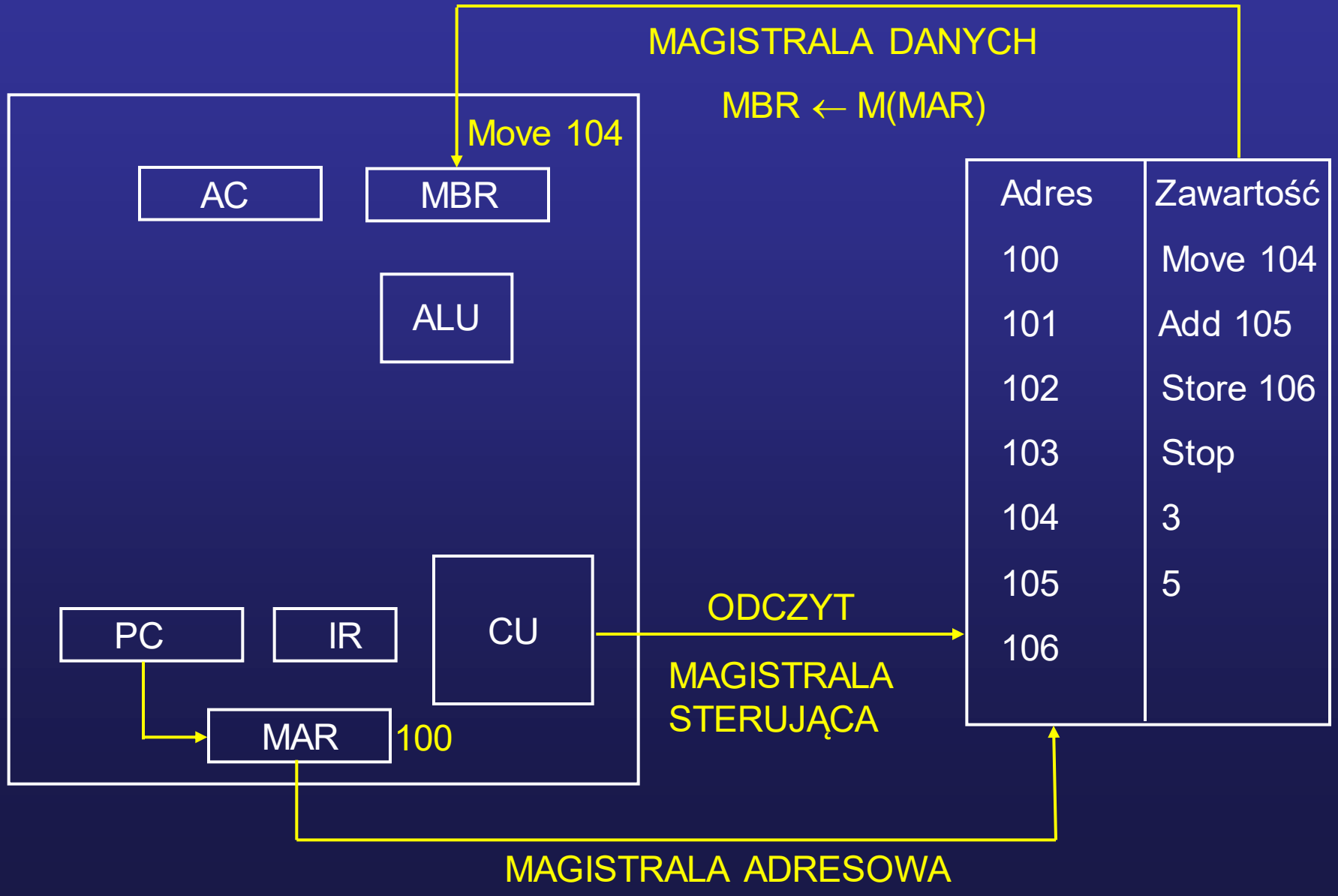
Cykl pobrania rozkazu

- Licznik programu (PC) przechowuje adres następnej instrukcji do pobrania (na początku jest to tzw. punkt wejścia)
- Procesor pobiera instrukcję spod adresu wskazywanego przez PC
- Wartość PC zwiększana jest o 1 (chyba że trzeba inaczej – np. skok)
- Instrukcja ładowana jest to rejestru rozkazów (IR)
- Procesor dekoduje instrukcję i wykonuje operację z nią związaną

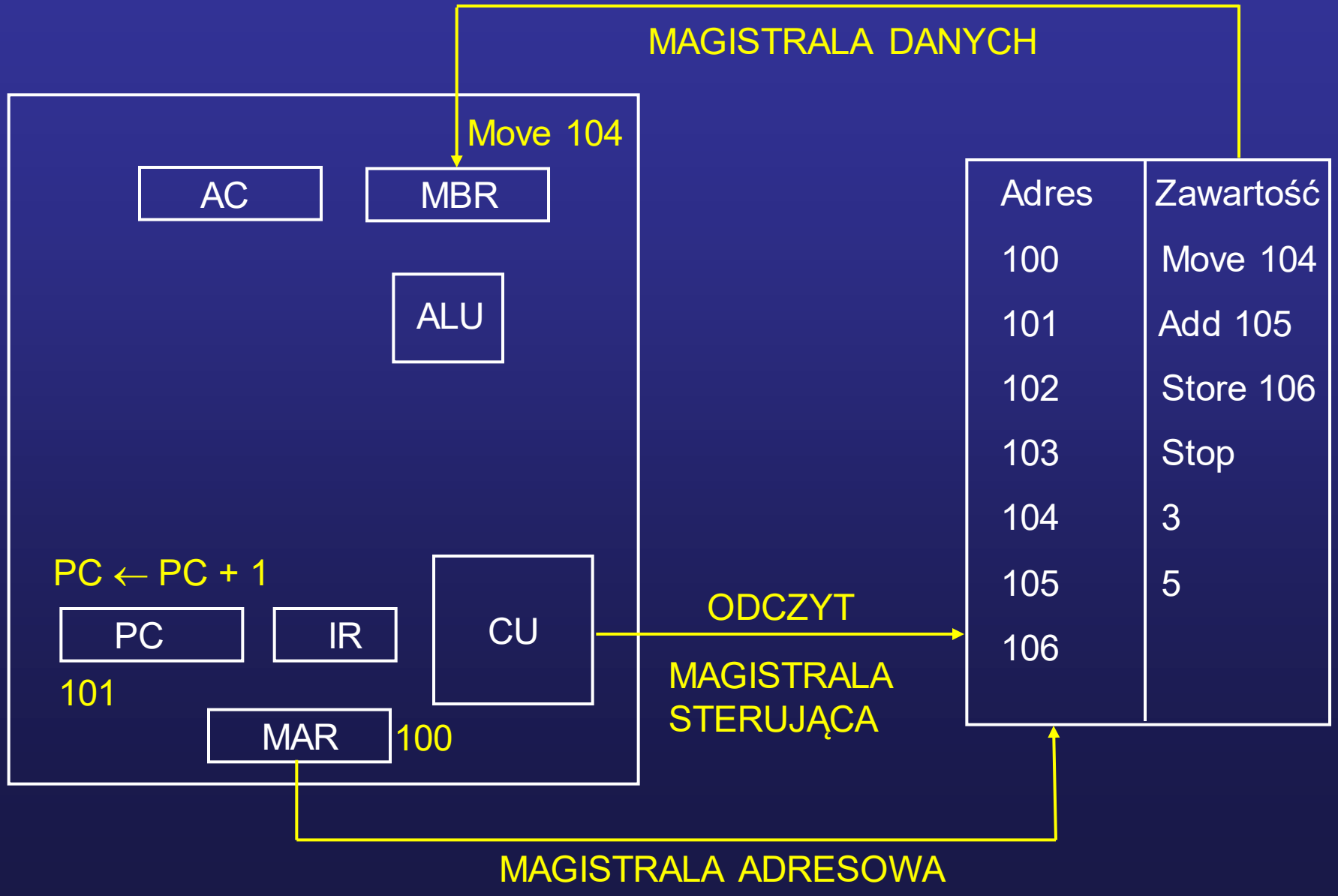
Ilustracja cyklu pobrania rozkazu



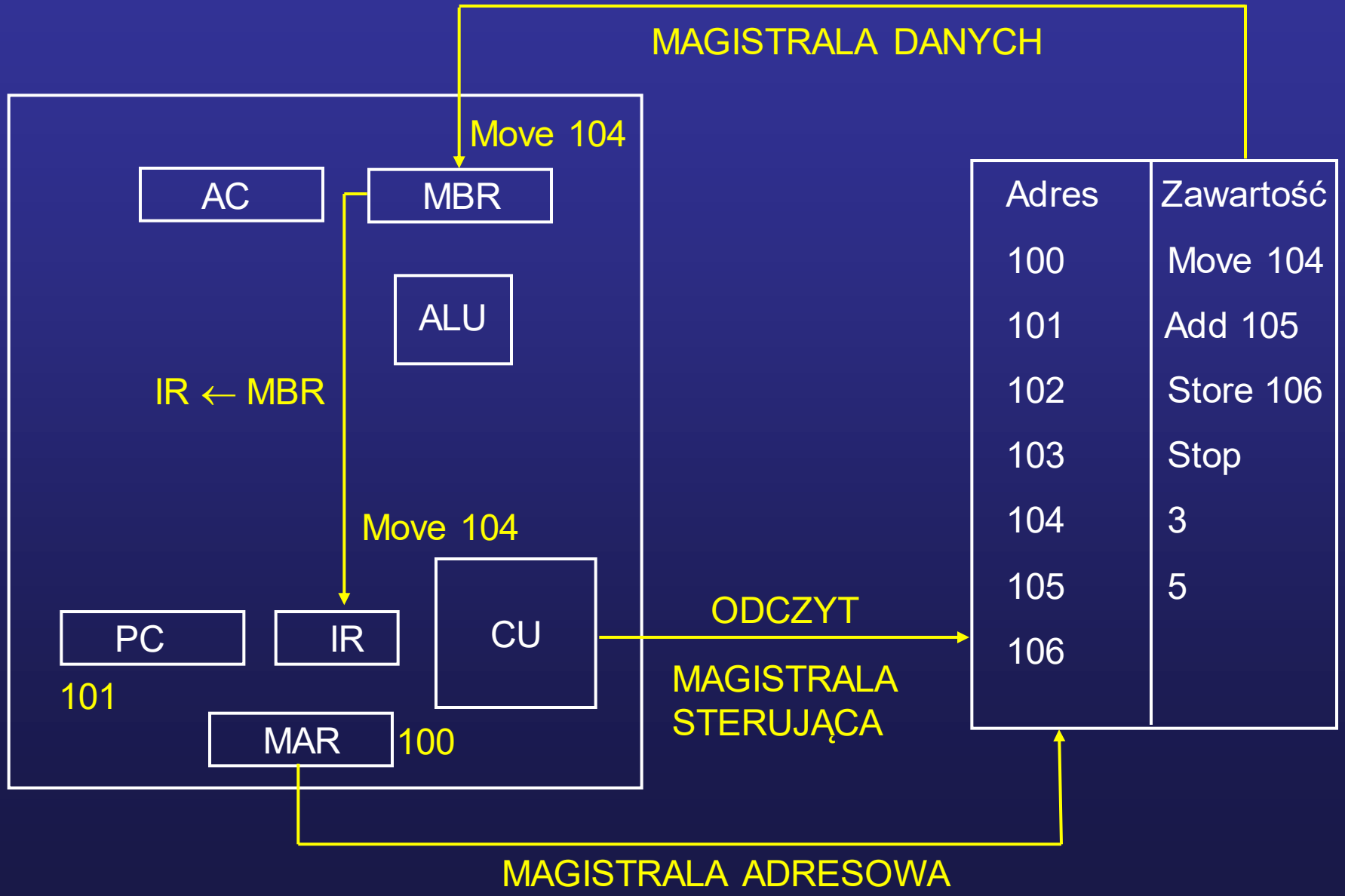
Ilustracja cyklu pobrania rozkazu



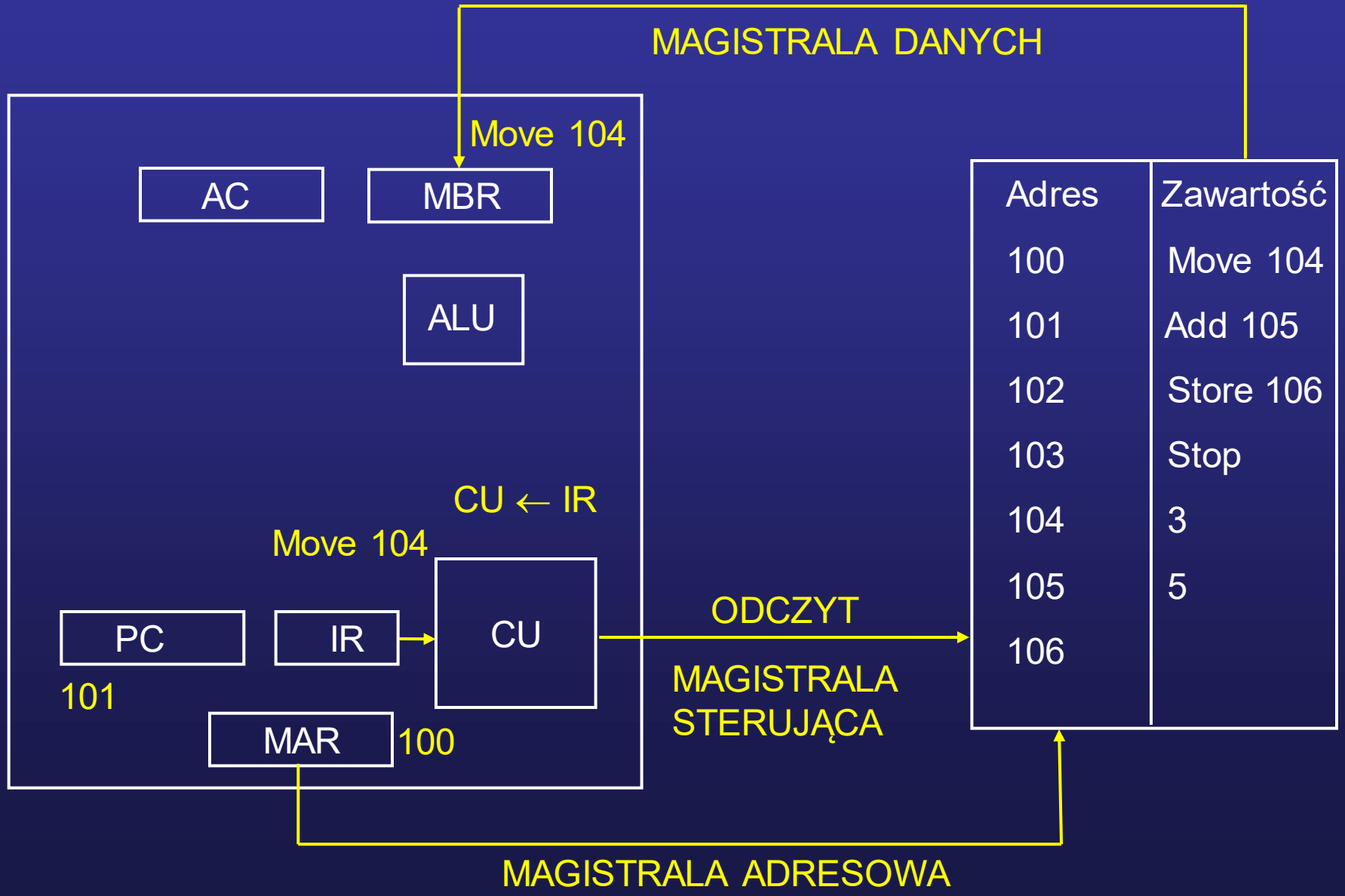
Ilustracja cyklu pobrania rozkazu



Ilustracja cyklu pobrania rozkazu



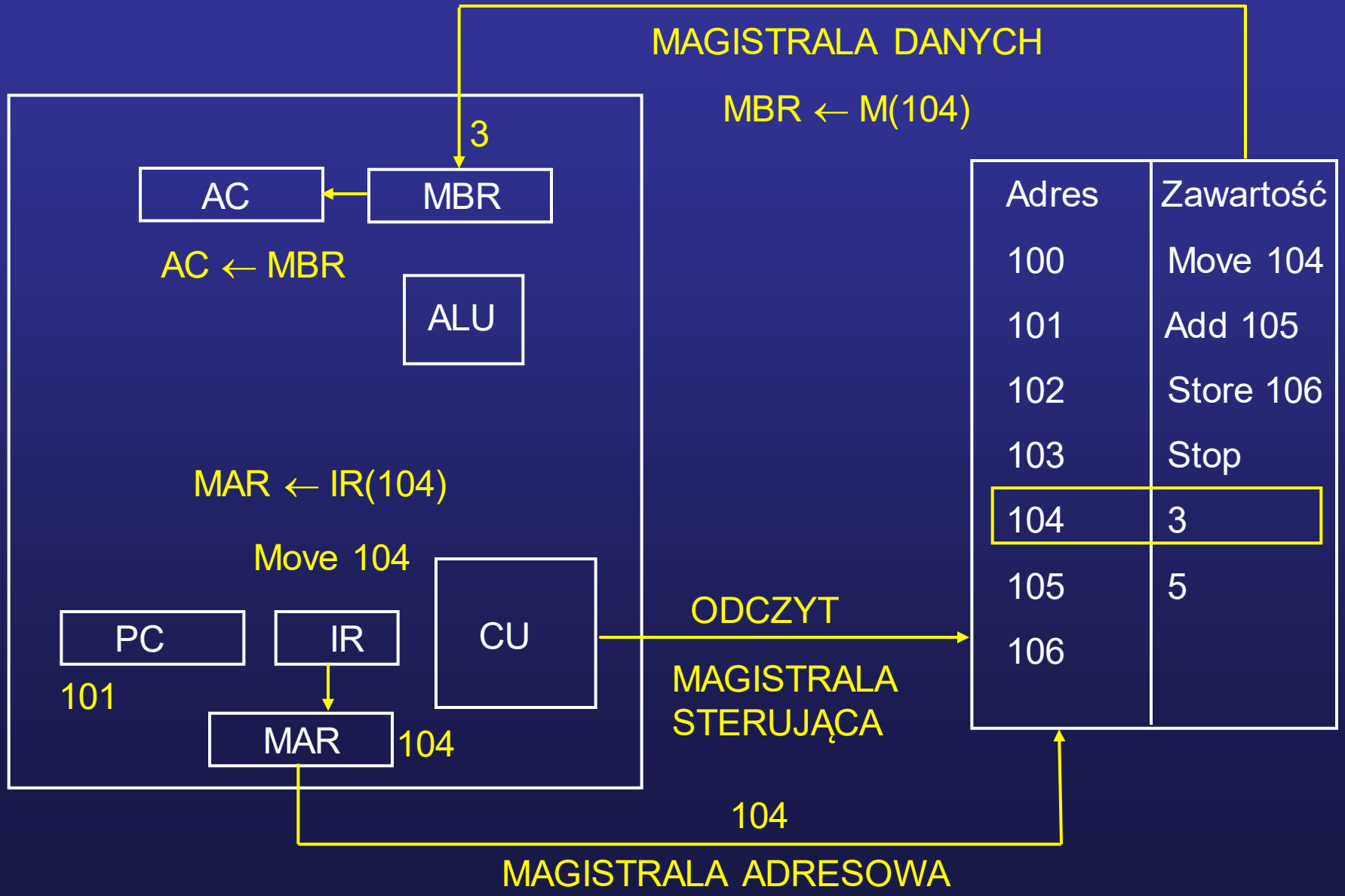
Ilustracja cyklu pobrania rozkazu



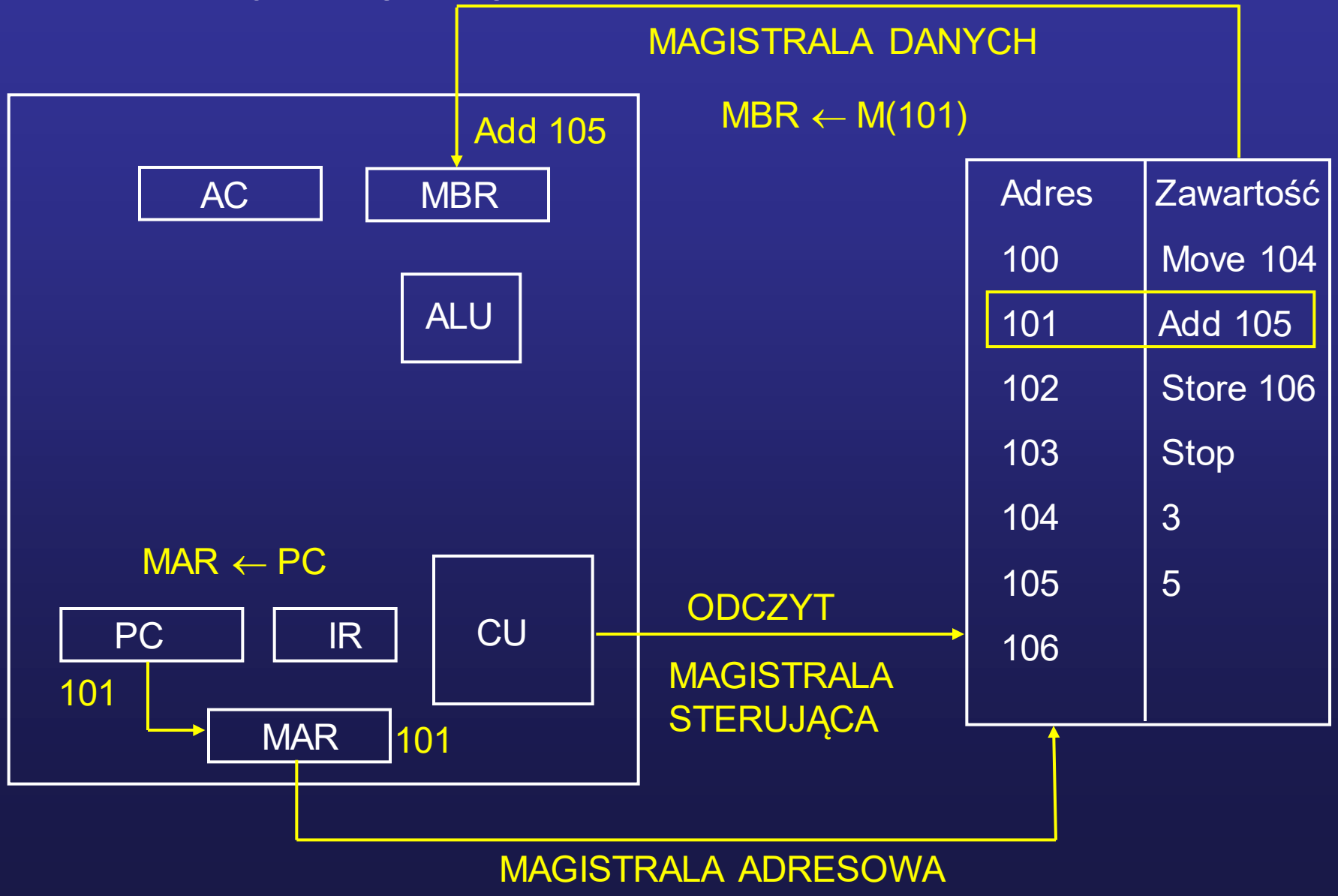
Cykl wykonania rozkazu

- Procesor – pamięć
  - transfer danych pomiędzy CPU a pamięcią
- Procesor – wejście/wyjście
  - transfer danych pomiędzy CPU a modułem wejścia/wyjścia
- Przetwarzanie danych
  - działania arytmetyczne lub logiczne na danych
- Sterowanie
  - zmiana kolejności wykonania operacji (np. skok)
- Kombinacja powyższych

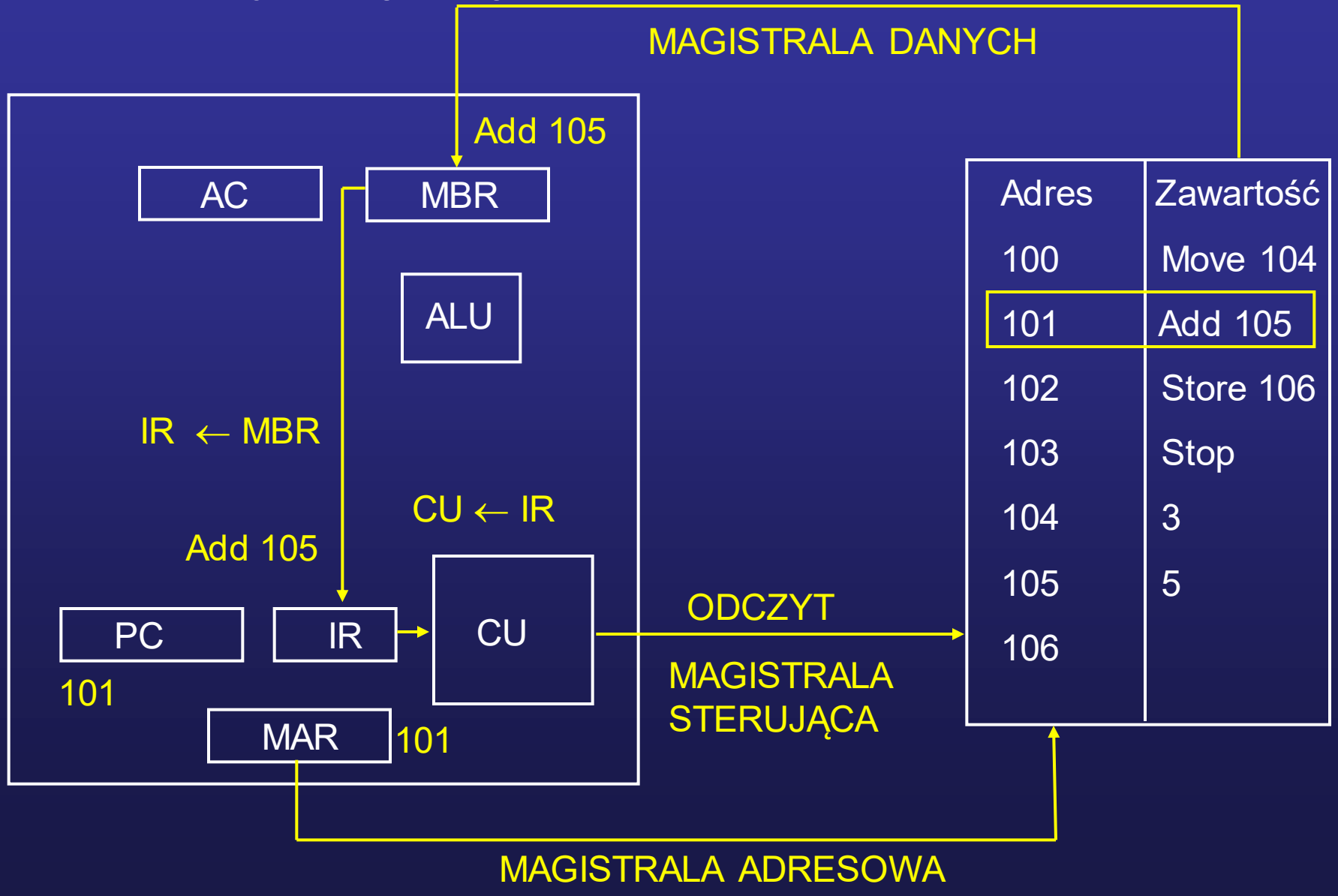
Ilustracja cyklu wykonania rozkazu



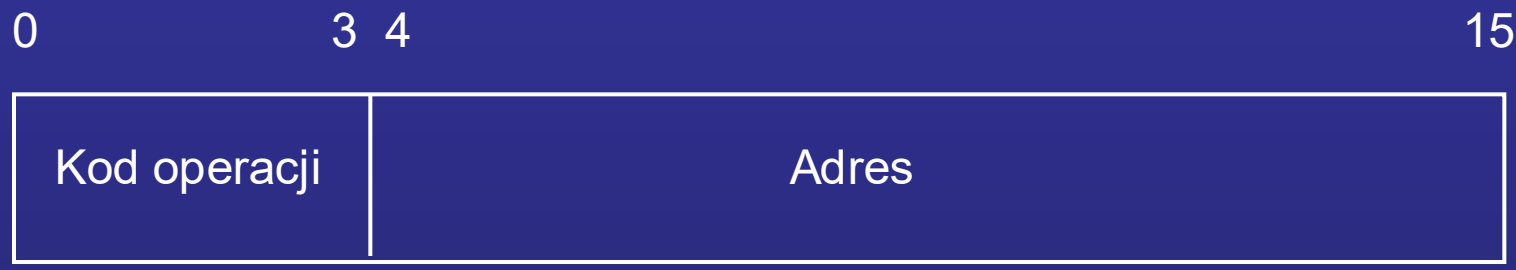
Następny cykl pobrania rozkazu

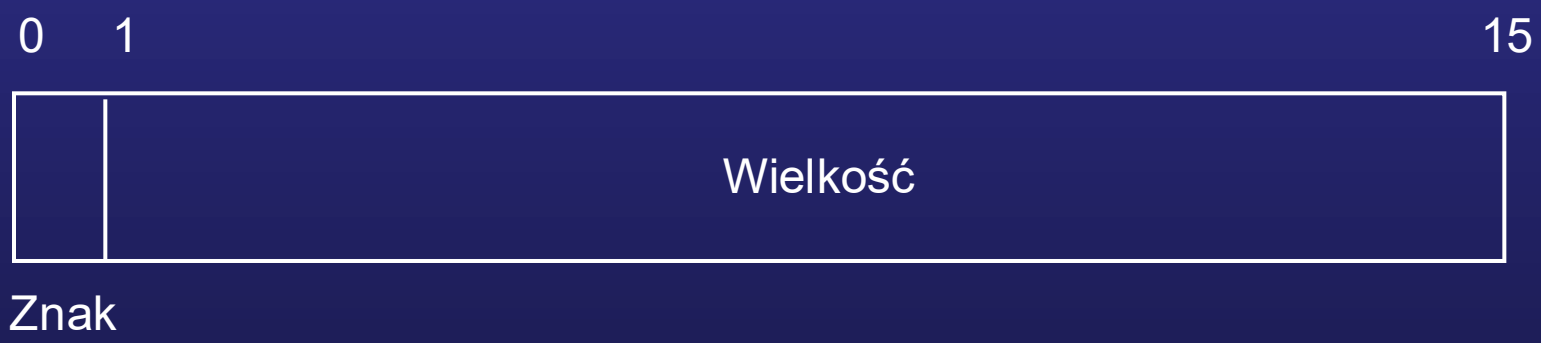


Następny cykl pobrania rozkazu



Format rozkazu





rozkaz argument

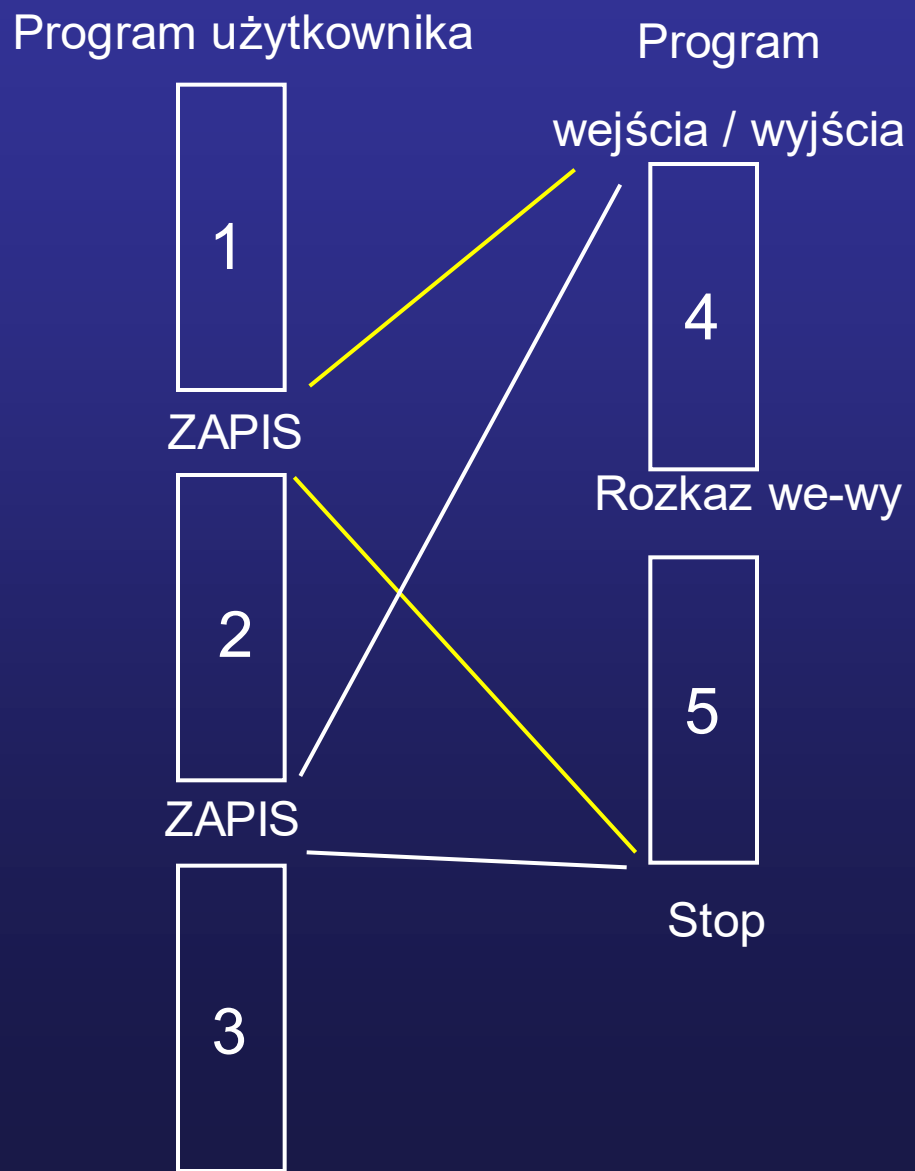
Np. Move 104 - 0101000001101000

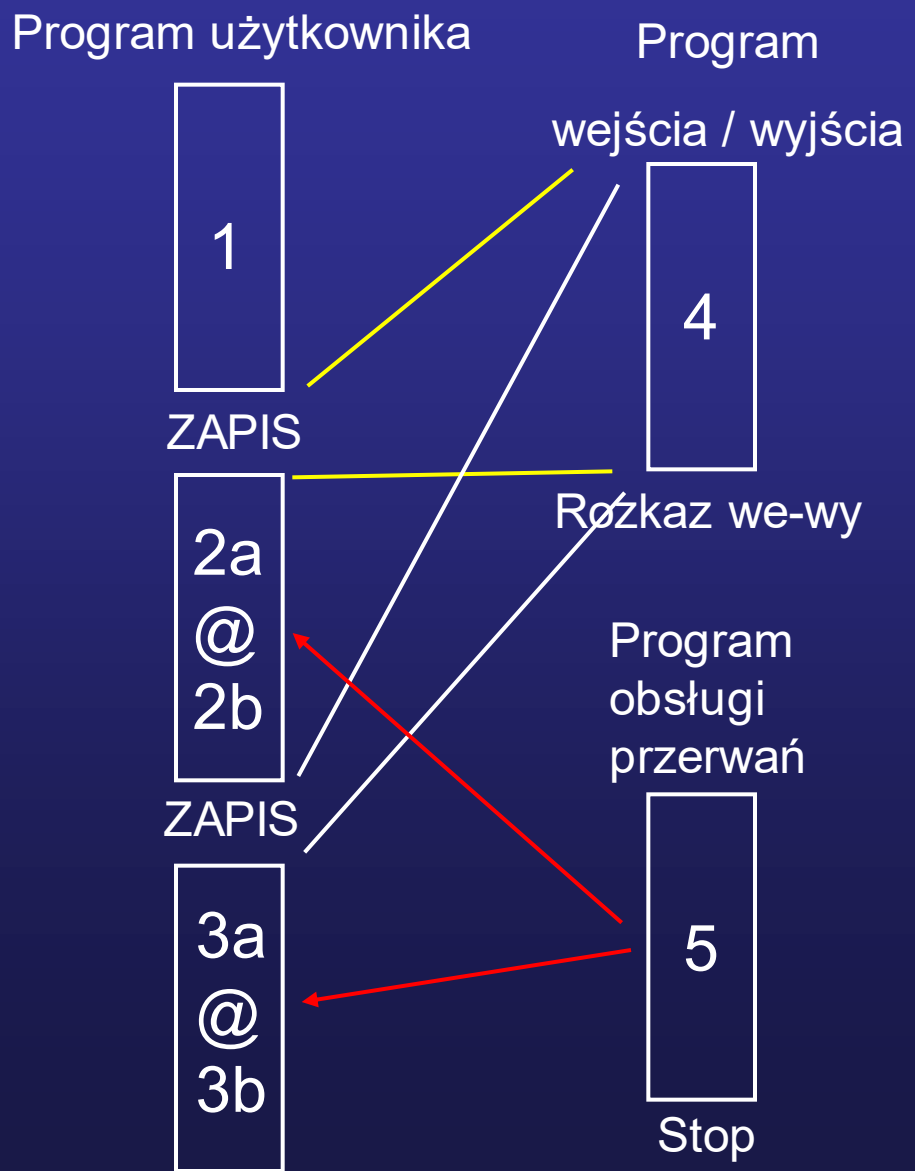
Graf stanów cyklu rozkazu

Przerwania

- Mechanizm pozwalający innym komponentom zakłócenie normalnego porządku przetwarzania
- Programowe
  - np. przepełnienie, dzielenie przez zero
- Zegarowe
  - Generowane przez wewnętrzny zegar procesora
  - Używane przy wywłaszczaniu procesów
- Wejścia/wyjścia
  - Z kontrolera wejścia/wyjścia
- Awaria sprzętu
  - Błąd parzystości pamięci

Idea zastosowania przerwań





Cykl przerwania

- Procesor cyklicznie sprawdza, czy nie wystąpiło przerwanie
  - Wskazane jest to przez sygnał przerwania
- Jeśli nie ma przerwania, pobierany jest następny rozkaz
- Jeśli jest przerwanie:
  - Zawiesza się wykonanie wykonywanego programu
  - Zapisywany jest jego kontekst
  - Licznik programu ustawiany jest na adres pierwszej instrukcji programu obsługi przerwań
  - Przerwanie jest przetwarzanie
  - Następuje przywrócenie poprzedniego kontekstu i powraca się do wykonywania programu użytkownika

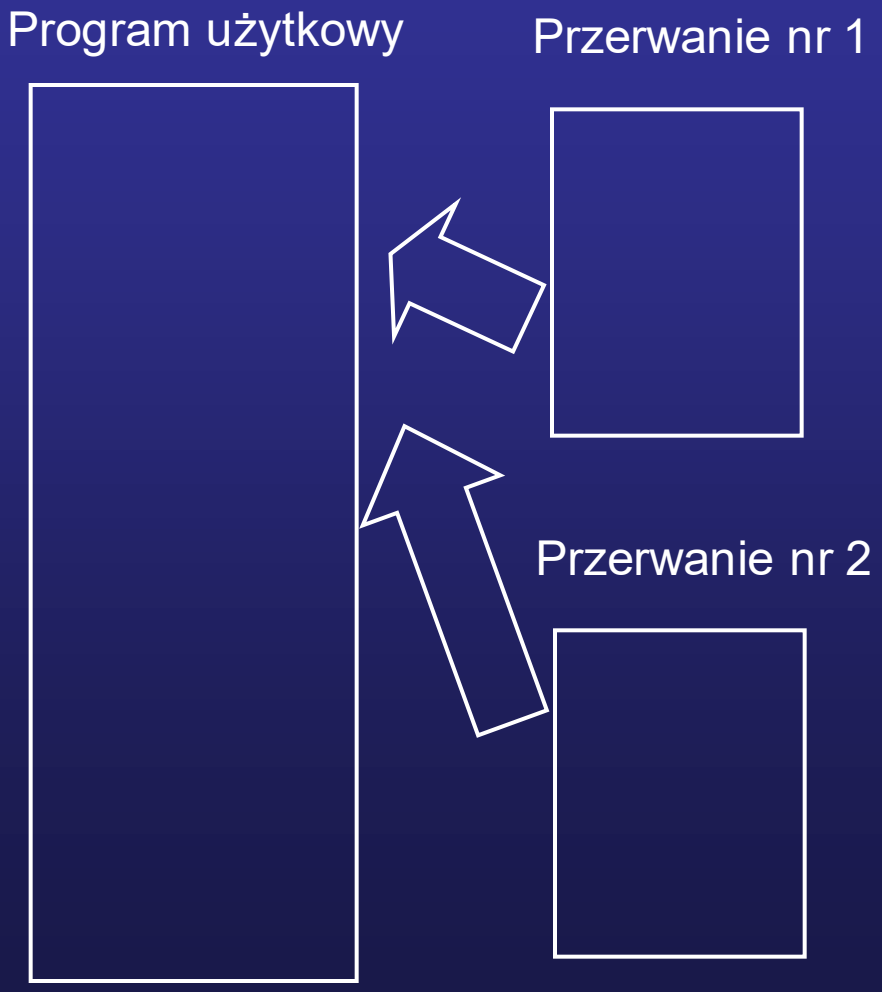
Przerwania wielokrotne

Istnieją dwie metody obsługi przerwań wielokrotnych

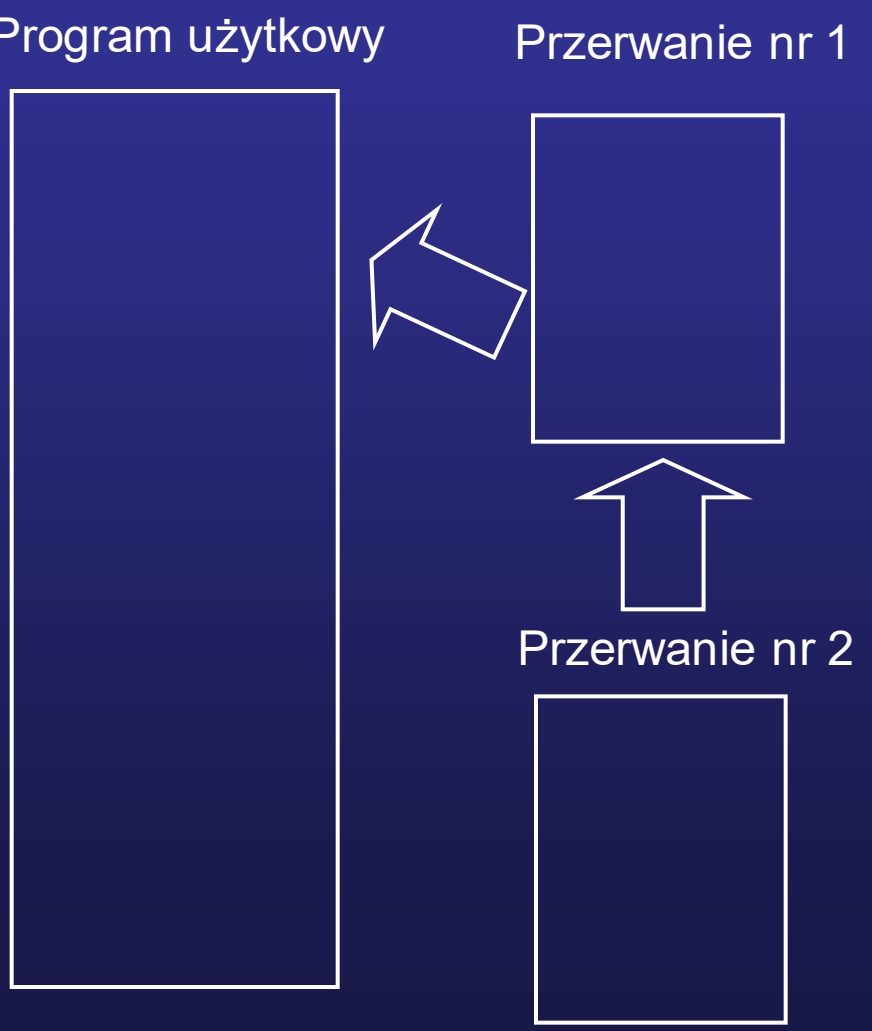
- Przerwania zablokowane
  - Procesor ignoruje inne przerwania podczas obsługi przerwania
  - Przerwania czekają w kolejce, po zakończeniu obsługi przerwania sprawdza się, czy nie ma następnego
  - Przerwania są obsługiwane w kolejności wystąpienia
- Priorytety
  - Obsługa przerwania o niskim priorytecie może zostać zakłócona przez przerwanie o wyższym priorytecie
  - Po zakończeniu obsługi przerwania o wyższym priorytecie następuje powrót do obsługi przerwania o niższym priorytecie

Ilustracja obsługi przerwań wielokrotnych

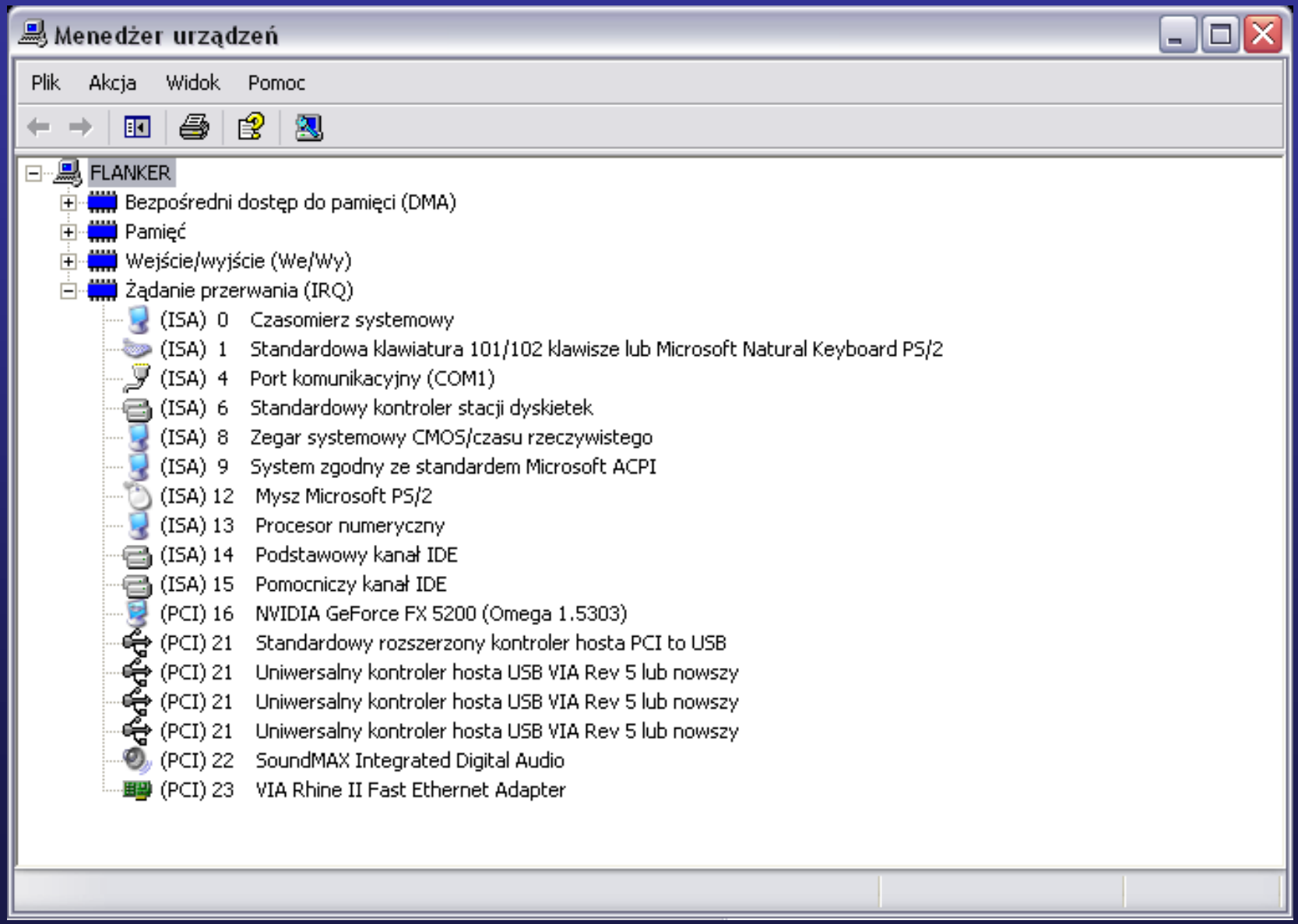
Obsługa sekwencyjna



Obsługa priorytetowa



Przykład przypisania przerwań



Przepływ danych w modułach komputerowych

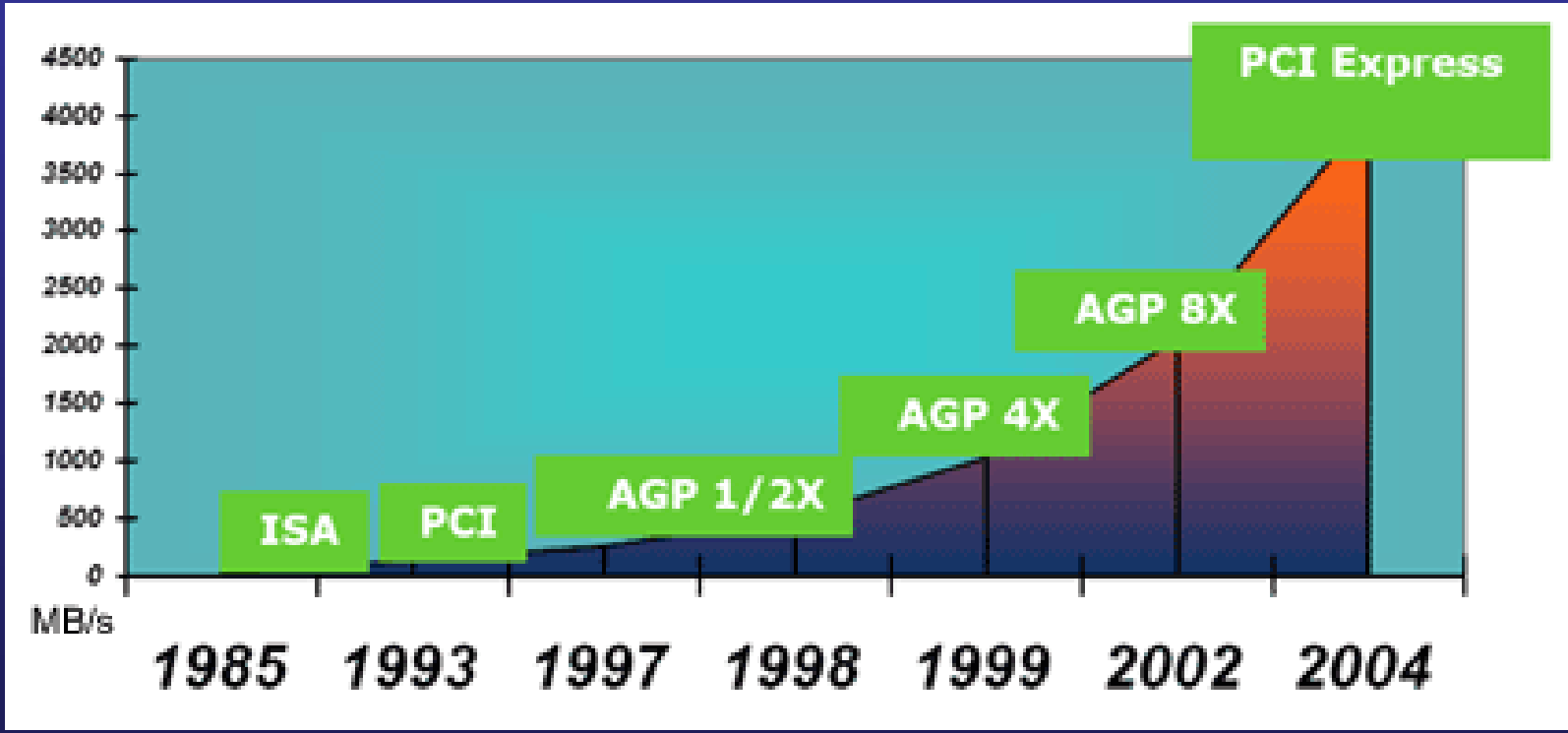
Magistrala

- Jest to droga zapewniająca komunikację z urządzeniami
- Składa się z wielu dróg komunikacyjnych (linii) trzech rodzajów: danych, adresowych i sterowania
- Szerokość magistrali to maksymalna liczba bitów, którą można przesłać jednocześnie (równa liczbie linii)

Najważniejsze magistrale

- ISA (1985)
- PCI (1993)
- AGP 1x/2x (1997-1998)
- AGP 4x (1999)
- AGP 8x (2002)
- PCI Express (2004)
- SCSI (1979)
- Magistrale przemysłowe (PROFIBUS, IEC-625)

Przepustowość magistral



Działanie magistrali

Komunikacja między dwoma modułami

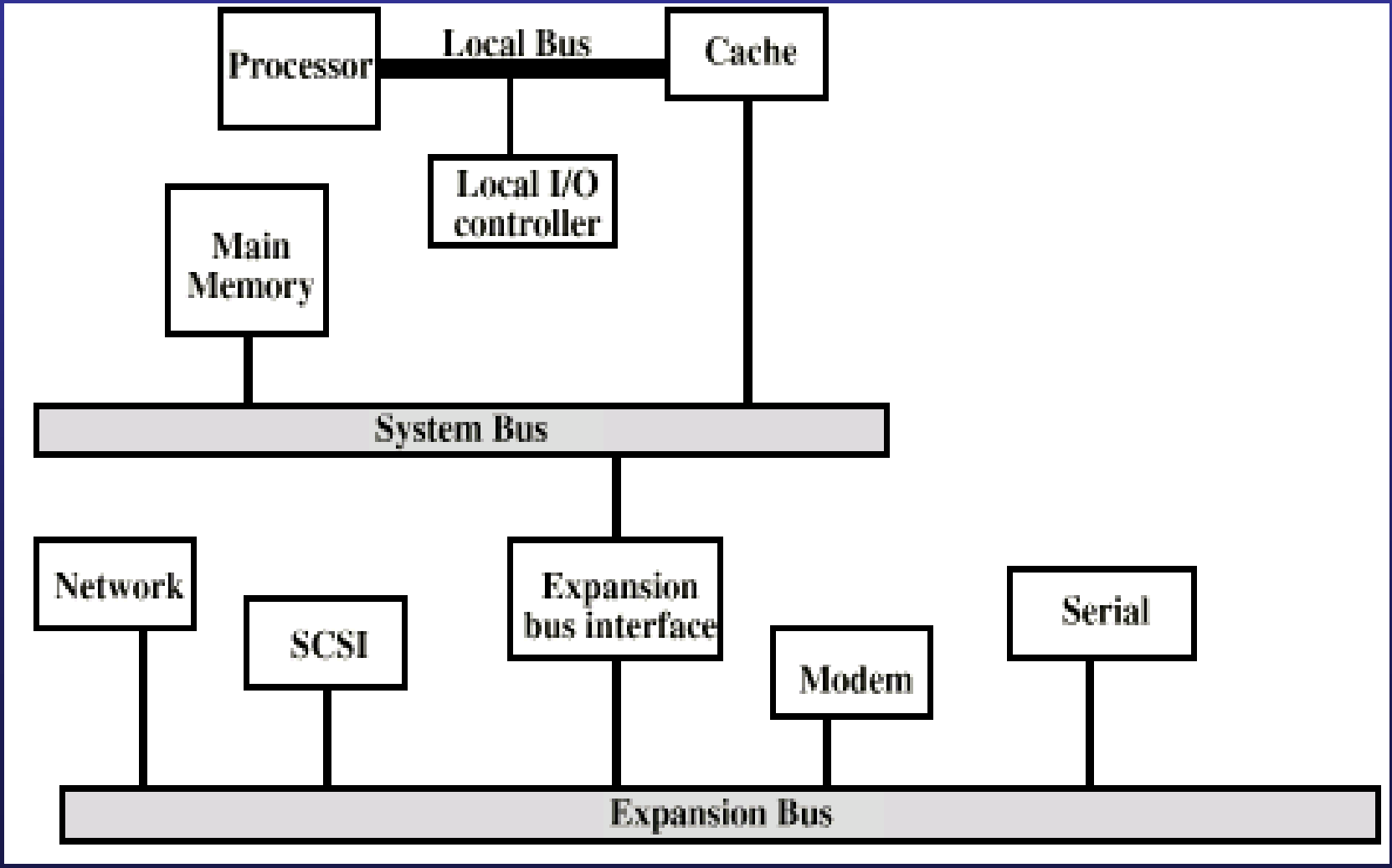
Wysyłanie danych:

1. Uzyskanie dostępu do magistrali
2. Przekazanie danych

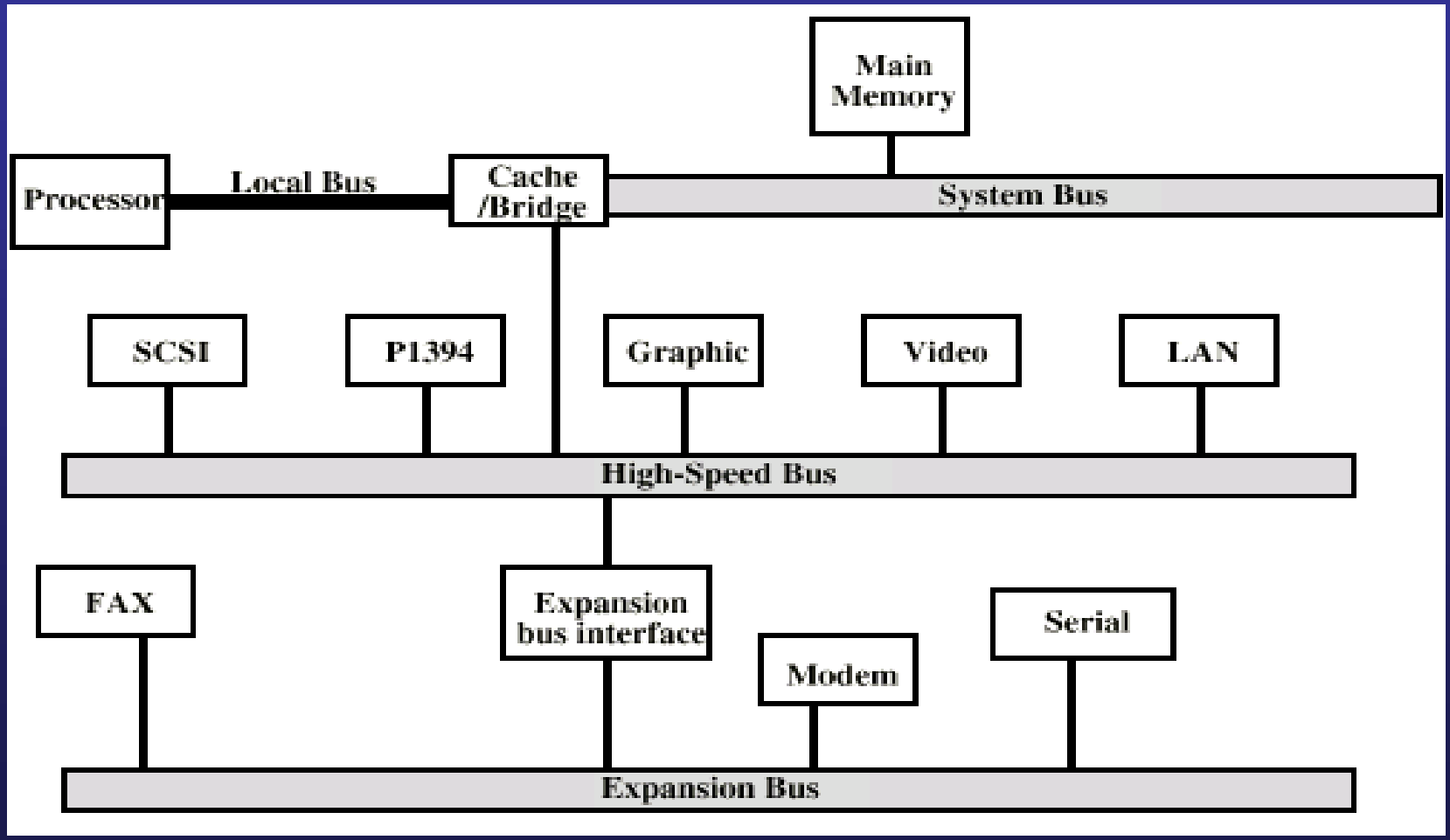
Odbieranie danych

1. Uzyskanie dostępu do magistrali
2. Przekazanie informacji o potrzebie uzyskania danych (poprzez linie sterujące)
3. Oczekiwanie na przesłanie danych

„Płaska” struktura magistrali (ISA)



Hierarchiczna struktura magistrali



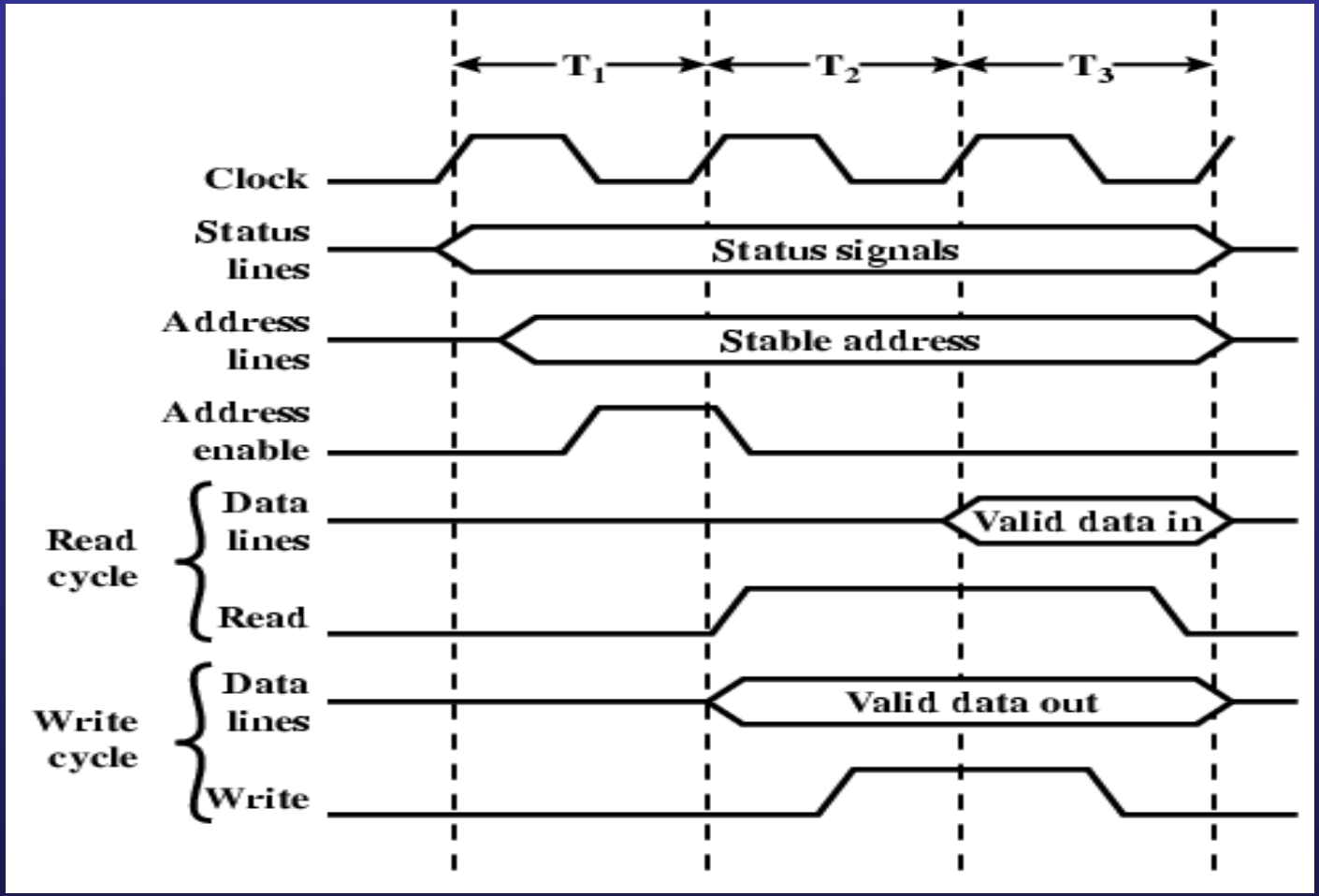
Rodzaje magistral

Przeznaczenie
- Specjalistyczna
- Multipleksowana

Metoda arbitrażu
- Scentralizowana
- Rozproszona

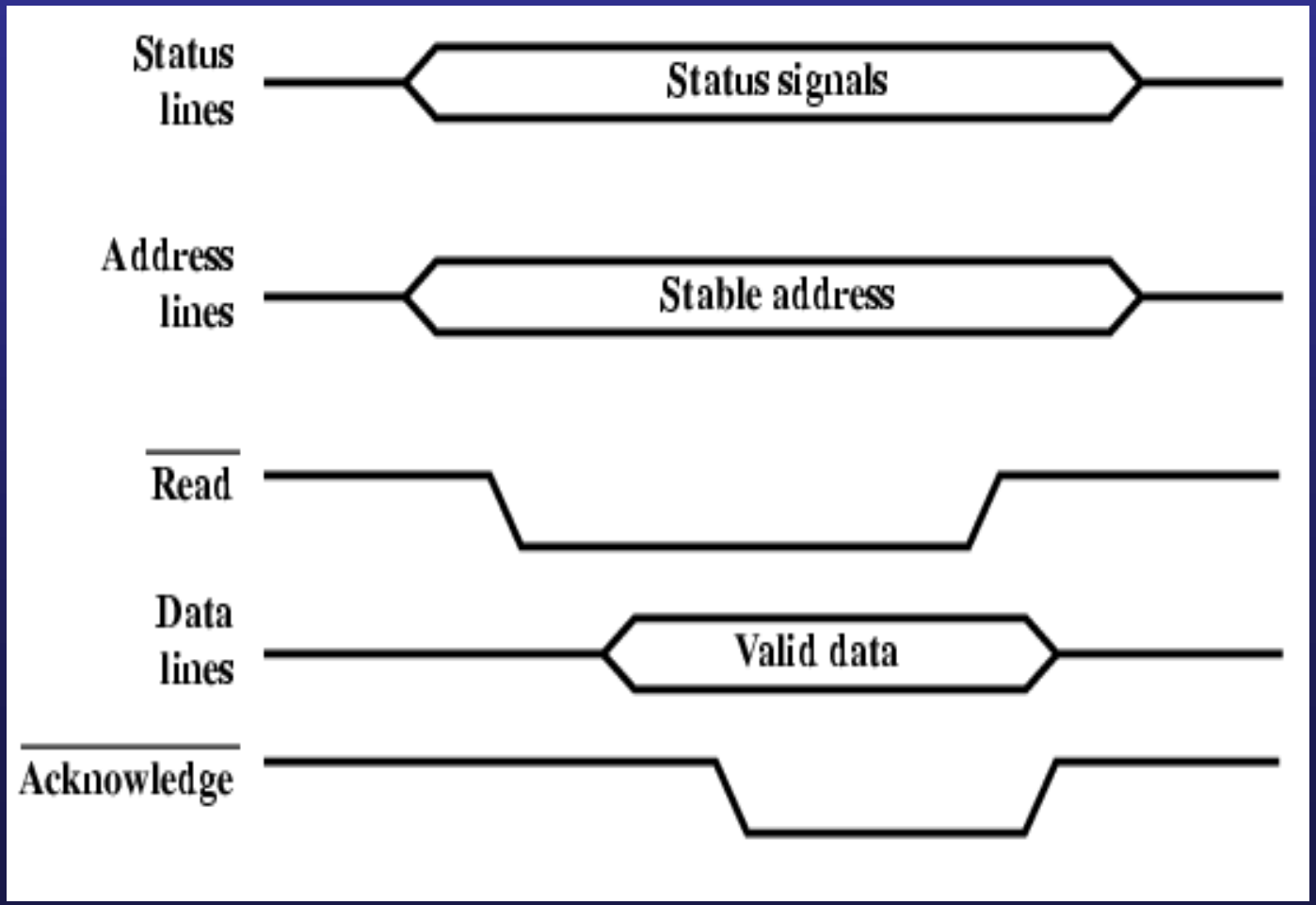
Koordynacja czasowa
- Synchroniczna
- Asynchroniczna

Koordynacja synchroniczna



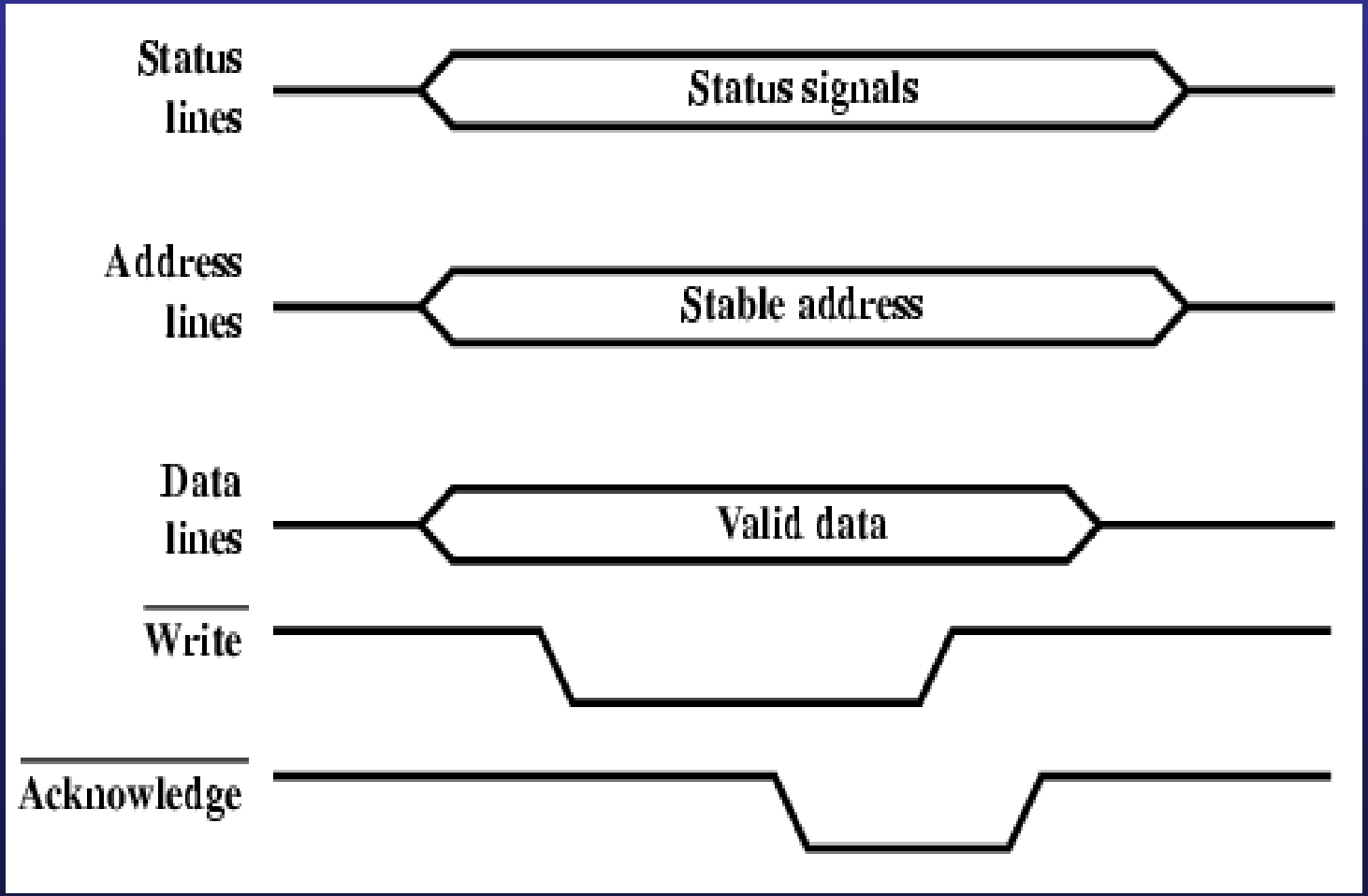
Koordynacja asynchroniczna

Odczyt



Koordynacja asynchroniczna

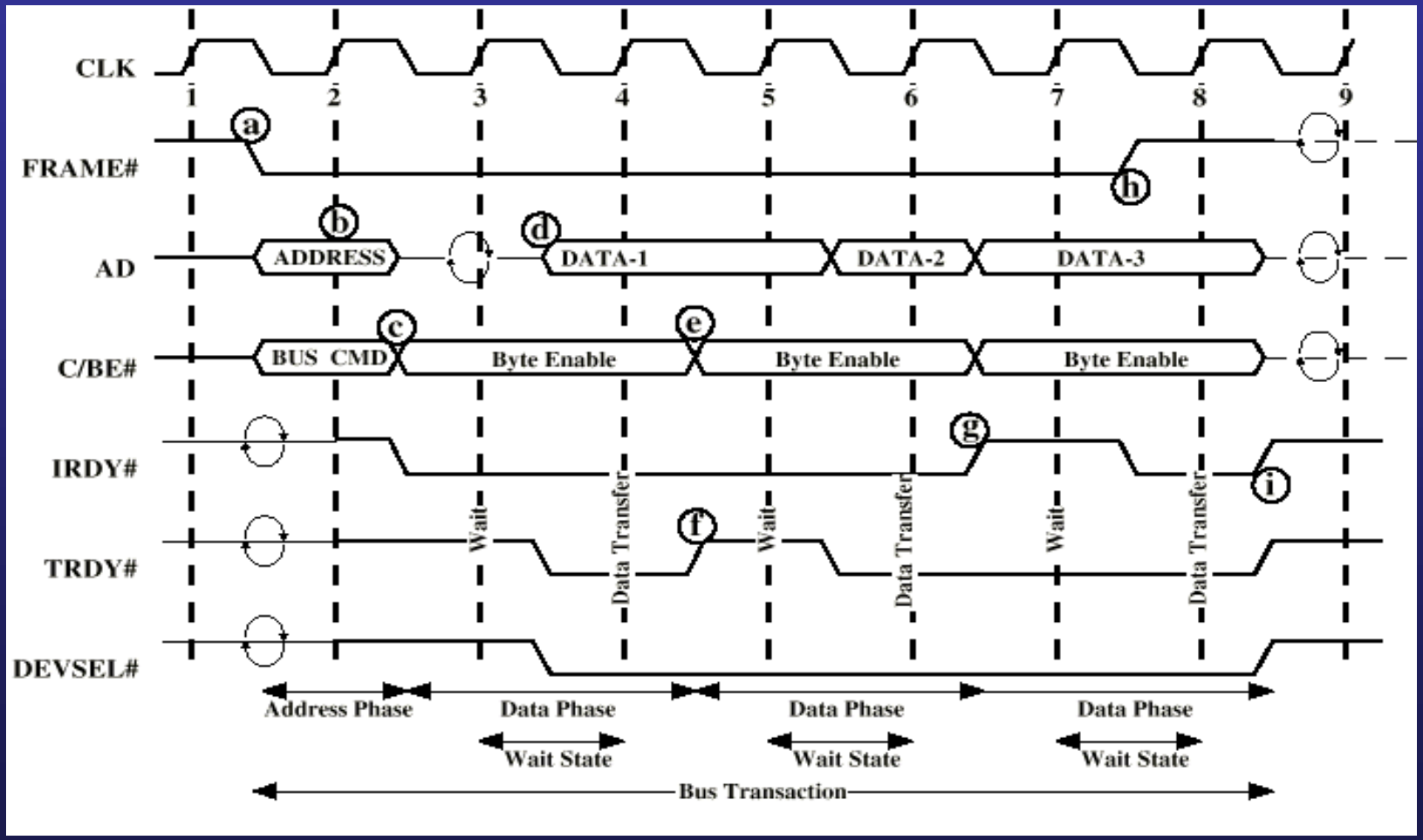
Zapis



Magistrala PCI

- Zaproponowana przez Intela w 1990 r.
- Częstotliwość pracy: 66 MHz
- Pracuje w konfiguracji 32 lub 64-bitowej
- Komunikacja z procesorem i pamięcią przez mosty (bridge), możliwa struktura hierarchiczna
- Obowiązujące linie sygnałowe:
  - systemowe (np. zegar, reset)
  - adresowe i danych (32)
  - sterowania interfejsu
  - arbitrażowe (indywidualne!)
  - informacja o błędach

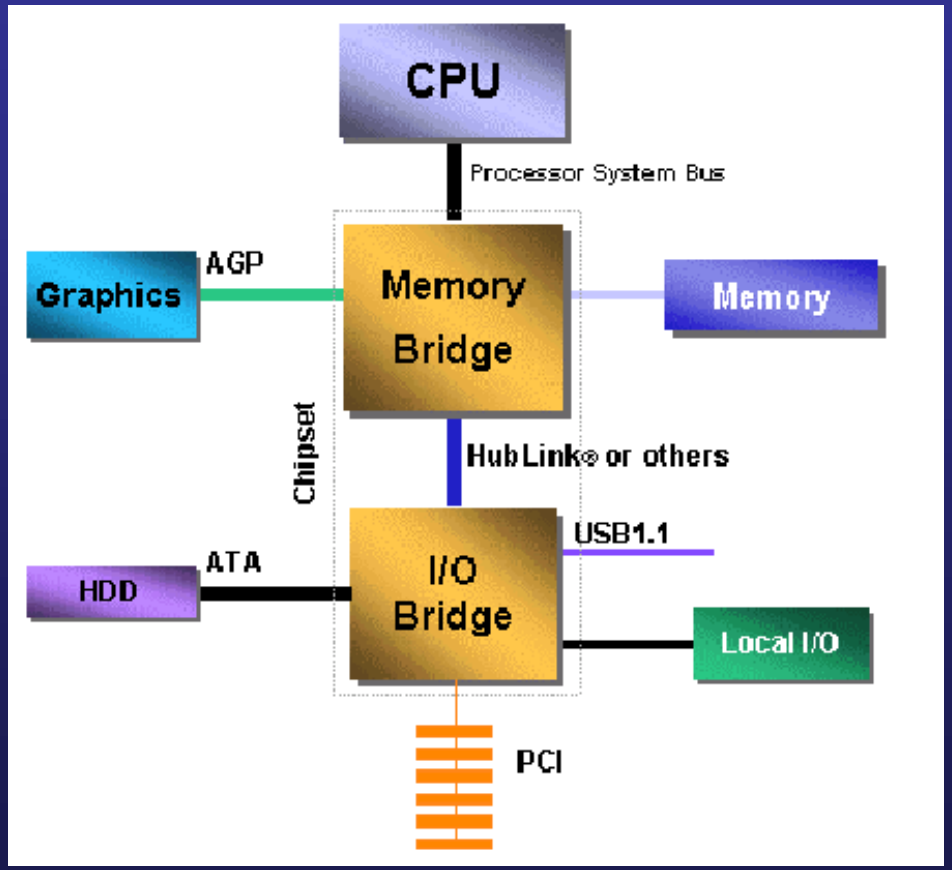
Odczyt z magistrali PCI

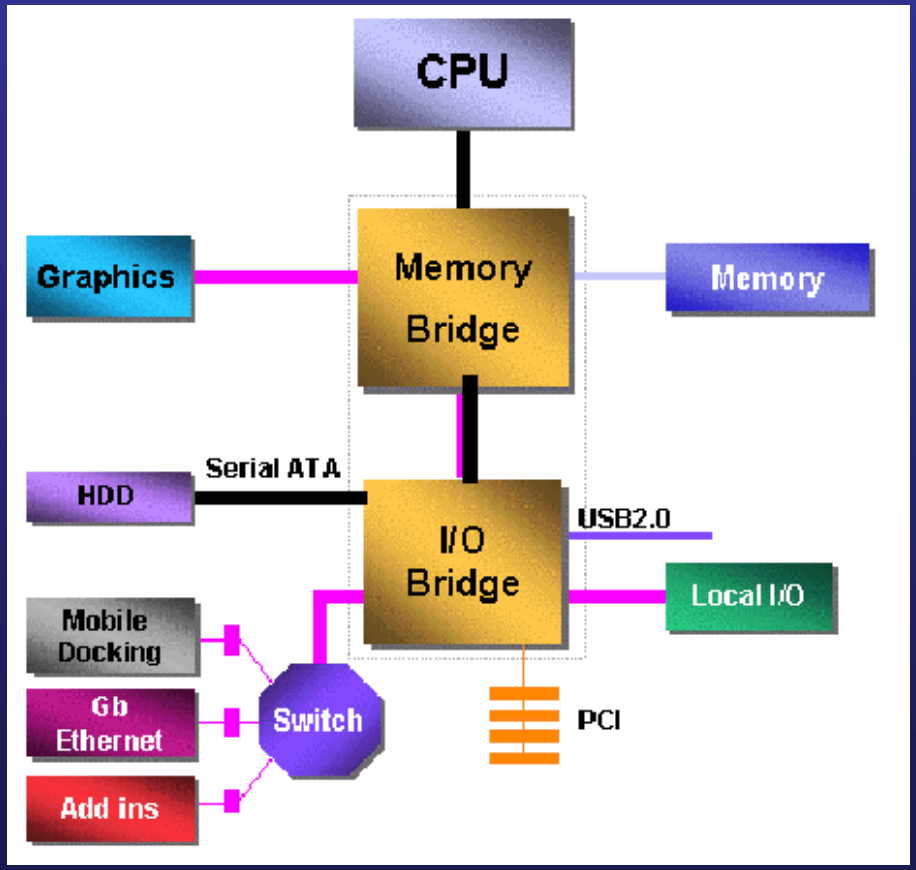


Magistrala PCI Express

- Połączenia typu punkt-punkt
- Wprowadzenie przełączników (switch)
- Częstotliwość taktowania: 2,5 GHz
- Skalowalna przepustowość magistrali – od 1x do 32x (w zależności od liczby wykorzystywanych kanałów/slotów)
- Prędkość 250 MB/s dla pojedynczego slotu (dla karty graficznej 16x PCIe jest to 4 GB/s)
- Wsteczna zgodność z PCI

Architektura komputera z PCI Express





Organizacja i architektura komputerów

Wykład nr 3: Układy logiczne, arytmetyka komputera

Piotr Bilski

Jednostka arytmetyczno-logiczna (ALU)

- Realizuje operacje arytmetyczno-logiczne
- Obliczenia wykonywane są na liczbach w kodzie dwójkowym (algebra Boole'a)

Reprezentacja liczb całkowitych

- Liczby całkowite:

$$61_{10} = 0\ 0\ 1\ 1\ 1\ 1\ 0\ 1_2$$

- Liczby rzeczywiste:

$$1{,}6328125_{10} = 1,\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0_2$$

Liczby całkowite (reprezentacja stałopozycyjna)

- Reprezentacja dla liczby *A* pozbawionej znaku

$$A = \sum_{i=0}^{n-1} 2^i \cdot a_i$$

- Reprezentacja znak-moduł

$$A = s + \sum_{i=0}^{n-2} 2^i \cdot a_i$$

- Reprezentacja uzupełnienia do dwóch (U2)

$$A = -2^{n-1} \cdot a_{n-1} + \sum_{i=0}^{n-2} 2^i \cdot a_i$$

Przykłady (znak-moduł)

$21_{10} = 00010101_2$

$-21_{10} = 10010101_2$

$0_{10} = 00000000_2$

$0_{10} = 10000000_2$

- Zero ma podwójną reprezentację
- Dodawanie i odejmowanie wymaga osobnej analizy znaków i modułów

Właściwości reprezentacji U2

Cecha	Znak-moduł	U2
Zakres	$-2^{n-1}-1$ do $2^{n-1}-1$	-2^{n-1} do $2^{n-1}-1$
Reprezentacje zera	Dwie	Jedna
Negacja	Zmiana MSB	Uzupełnienie do 2
Zwiększenie liczby bitów	Dodatkowe bity = 0 MSB $\leftarrow$ znak	Dodatkowe bity wypełniane znakiem
Przepełnienie	Analiza znaków i modułów	Gdy znaki operandów są równe, a wyniku - przeciwny
Odejmowanie	Osobny algorytm	Dodawanie liczby przeciwnej

Operacje arytmetyczne na liczbach całkowitych (1)

Negacja:

Znak-moduł

00010101 (21)

+10000000

10010101 (-21)

U2

00010101 (21)

11101010 NEG(21)

+00000001

11101011 (-21)

Operacje arytmetyczne na liczbach całkowitych (2)

Rozszerzenie bitowe:

Znak-moduł

00010101 (21) 8b

00000000 00010101 (21) 16b

10010101 (-21)

10000000 00010101 (-21) 16b

U2

00010101 (21) 8b

00000000 00010101 (21) 16b

11101011 (-21) 8b

11111111 11101011 (-21) 16b

Operacje arytmetyczne na liczbach całkowitych (3)

Dodawanie:

Znak-moduł

00011001 (25)
+00001101 (13)
00100110 (38)

10011001 (-25)
+00001101 (13)
10001100 (-12)

x0011001
x1110010 NEG(13)
0001011
+0000001
10001100 (-12)

10011001 (-25)
+10001101 (-13)
10100110 (-38)

10001101 (-13)
+00011001 (25)
00001100 (12)

x0001101
x1100110 NEG(25)
1110011 R
00001100 NEG(R)

Operacje arytmetyczne na liczbach całkowitych (4)

Dodawanie:

U2

```
  00011001 (25)
+ 00001101 (13)
  00100110 (38)
```

```
  11100111 (-25)
+ 00001101 (13)
  11110100 (-12)
```

```
  11100111 (-25)
+ 11110011 (-13)
  11011010 (-38)
```

```
  11110011 (-13)
+ 00011001 (25)
  00001100
```

Operacje arytmetyczne na liczbach całkowitych (5)

- Mnożenie

Liczby całkowite bez znaku: 7x5

0111 (7) 4b ← mnożna

0101 (5) 4b ← mnożnik

00000111

00000000

00011100

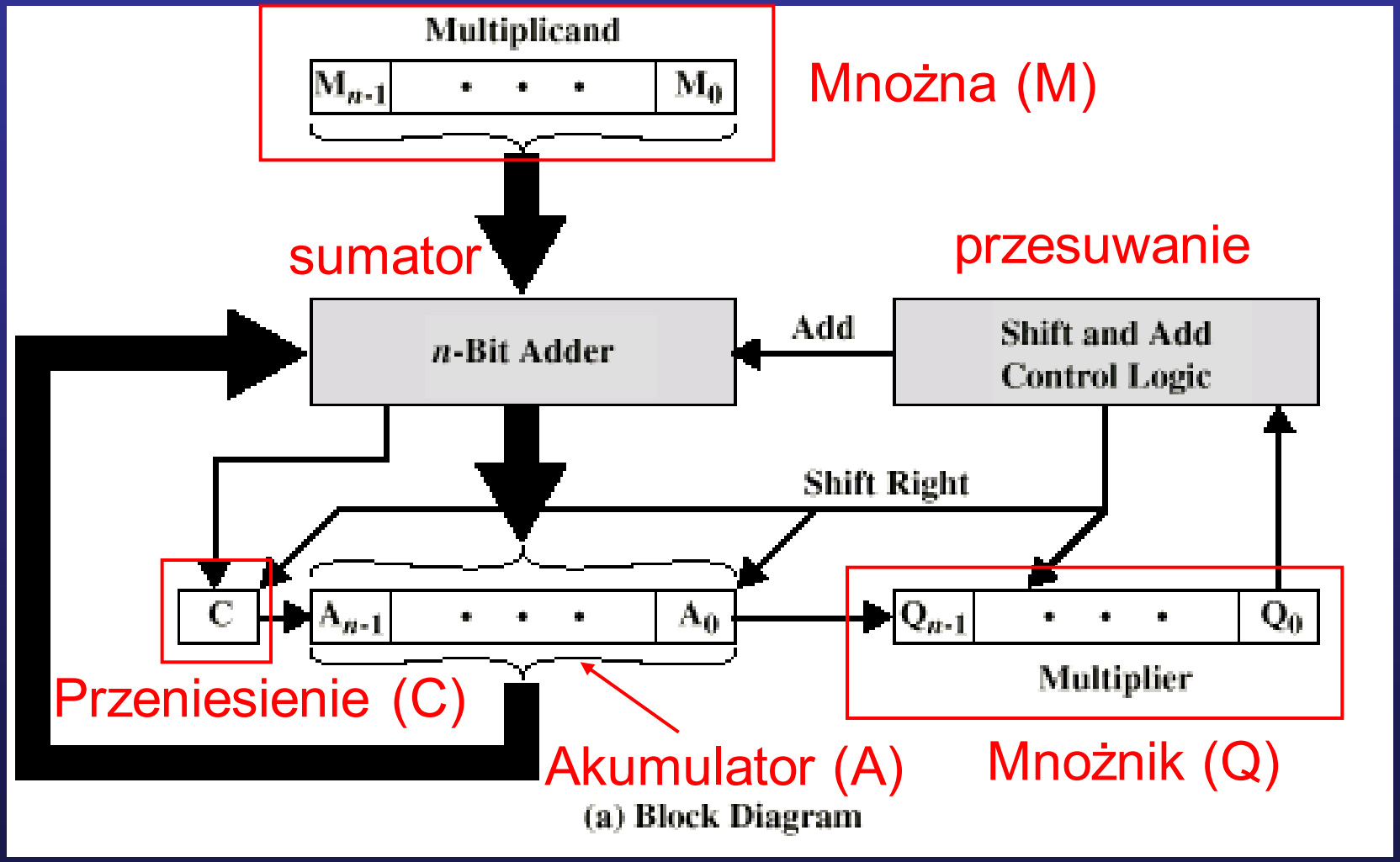
00000000

00100011 (35) 8b !!

Usprawnienia metody mnożenia

- Każdy wynik cząstkowy od razu sumowany (mniej rejestrów!)
- Mnożenie przez zero to tylko przesunięcie!

Realizacja sprzętowa mnożenia liczb bez znaku



Przykład działania realizacji sprzętowej

C	A	Q	M	
0	0000	0101	0111	wartości początkowe
0	0111	0101	0111	dodaj
0	0011	1010	0111	przesuń
0	0001	1101	0111	przesuń
0	1000	1101	0111	dodaj
0	0100	0110	0111	przesuń
0	0010	0011	0111	przesuń
0	0010	0011	0111	wynik

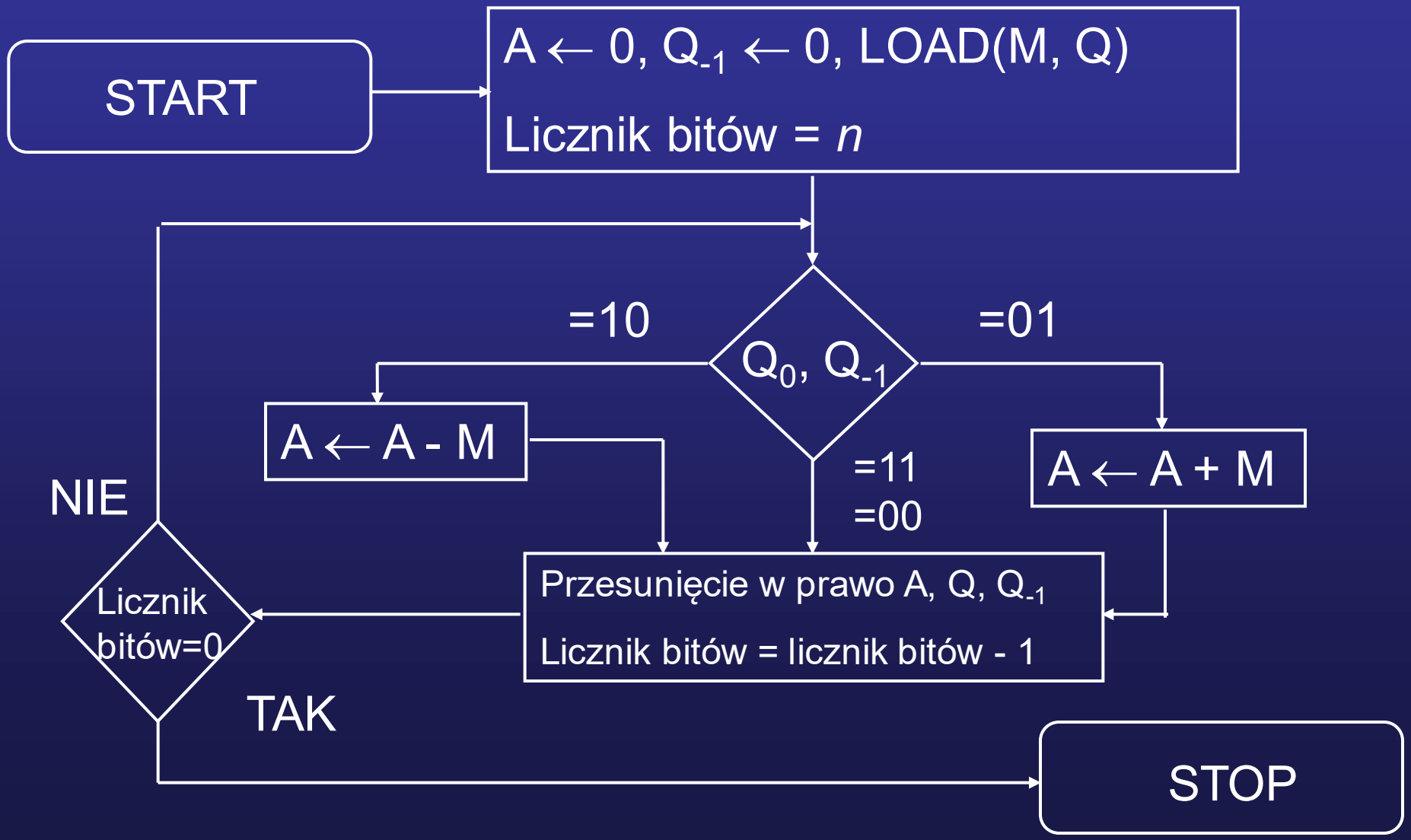
Mnożenie w reprezentacji U2

- Liczby całkowite ze znakiem: -7x3

1001 (-7) ← mnożna
0011 (3) ← mnożnik
11111001
11110010
11101011 (-21)

Inna interpretacja przesuwania binarnego
Liczba ujemna musi być reprezentowana w kodzie U2

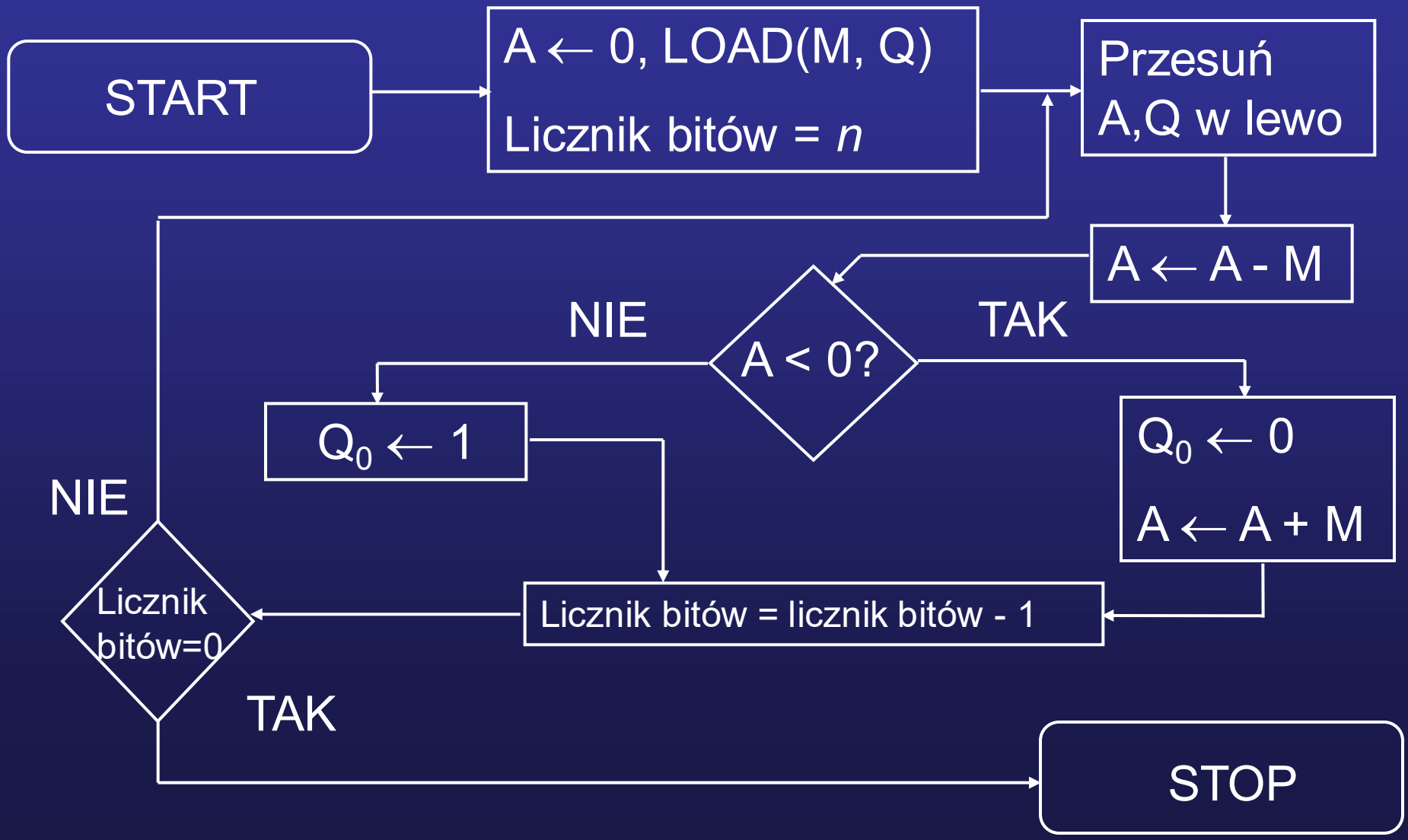
Realizacja mnożenia w kodzie U2 – algorytm Bootha



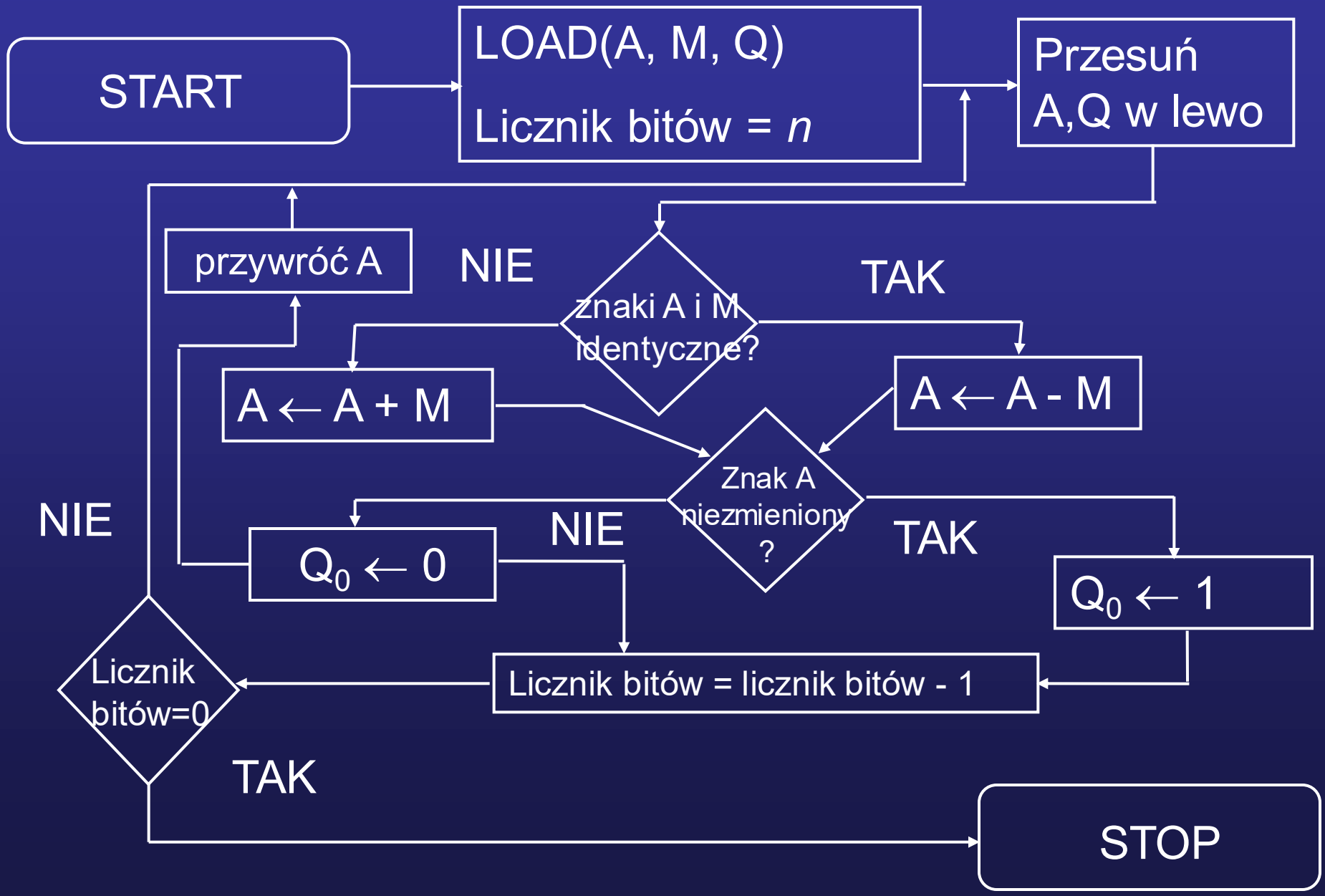
Przykład mnożenia w kodzie U2 (operacja 7 x 3)

A	Q	Q_{-1}	M	
0000	0011	0	0111	wartości początkowe
1001	0011	0	0111	odejmij
1100	1001	1	0111	przesuń arytmetycznie
1110	0100	1	0111	przesuń arytmetycznie
0101	0100	1	0111	dodaj
0010	1010	0	0111	przesuń arytmetycznie
0001	0101	0	0111	przesuń arytmetycznie
0001	0101	0	0111	wynik

Dzielenie liczb całkowitych bez znaku



Dzielenie liczb w kodzie U2



Przykład dzielenia w kodzie U2 (operacja –7/3)

A	Q	M	
1111	1001	0011	wartości początkowe
1111	0010	0011	przesuń
0010			dodaj
1111	0010	0011	przywróć
1110	0100	0011	przesuń
0001			dodaj
1110	0100	0011	przywróć
1100	1000	0011	przesuń
1111			dodaj
1111	1001	0011	ustaw $Q_0 = 1$
1111	0010	0011	przesuń
0010			dodaj
1111	0010	0011	przywróć
1111	0010	0011	wynik

Reprezentacja zmiennopozycyjna

- Służy do reprezentacji liczb bardzo małych oraz bardzo dużych
- Liczba taka ma postać:

$$A = \pm m \cdot b^{\pm c}$$

gdzie:

m – mantysa

c – cecha (wykładnik)

b - podstawa

Przykłady

1.24×10^{7} (1.24e7)

5.82×10^{-21}

$0.010110 \times 2^{110101}$

0.001001×16^{101}

Format 32-bitowej liczby zmiennopozycyjnej:

Zapis liczby binarnej w formacie zmiennopozycyjnym

$1{,}6328125 \times 2^{20} = 1.1010001 \times 2^{10100}$

0,6328125 x 2	1,265625	1	0,265625
0,265625 x 2	0,53125	0	0,53125
0,53125 x 2	1,0625	1	0,0625
0,0625 x 2	0,125	0	0,125
0,125 x 2	0,25	0	0,25
0,25 x 2	0,5	0	0,5
0,5 x 2	1,0	1	0

Zapis liczby binarnej w formacie zmiennopozycyjnym

$1{,}6328125 \times 2^{20} = 1.1010001 \times 2^{10100}$

0	10010011	10100010000000000000000

- Wykładnik jest liczbą przesuniętą o 127, więc 20 = 127 + 20 = 147 (j.w.)

Normalizacja liczby w formacie zmiennopozycyjnym

- Wykładnik jest tak zmieniany, aby pierwsza cyfra mantysy przed przecinkiem była niezerowa
- Ponieważ niezerowa cyfra to 1, nie trzeba jej przechowywać

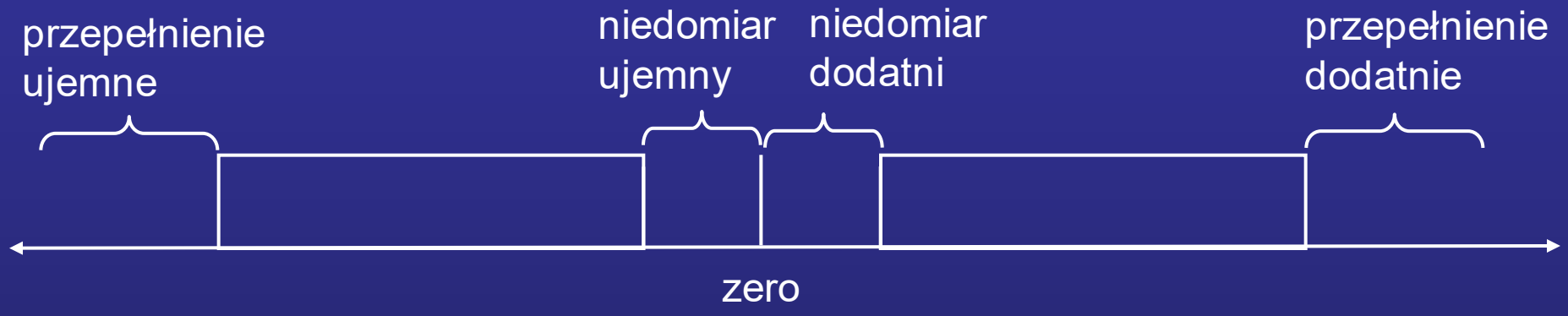
Zakres i dokładność liczb zmiennopozycyjnych

Dla liczby 32-bitowej:

- Wykładnik jest 8-bitowy, zatem zakres liczb to $+-2^{256}$
- Mantysa jest 23-bitowa, zatem dokładność wynosi $2^{-23} = 1.2 \times 10^{-7}$

Konieczny jest kompromis pomiędzy dokładnością i zakresem

Gęstość i zakres liczb w formacie zmiennopozycyjnym



Norma IEEE 754

- Stosowany standard przechowywania liczb zmiennopozycyjnych
- Dotyczy liczb 32- i 64-bitowych
- Cecha o długości, odpowiednio, 8 i 11 bitów
- Domyślna podstawa wynosi 2
- Przewidziane formaty rozszerzone dla obliczeń pośrednich

Wartości specjalne w IEEE 754

- $c = 0, m = 0$ – dodatnie lub ujemne zero
- $c = 11111111, m = 0$ – dodatnia lub ujemna nieskończoność
- $c = 0, m \neq 0$ - liczba zdenormalizowana (bit na lewo od przecinka jest zerem!)
- $c = 11111111, m \neq 0$ - NaN

Arytmetyka zmiennopozycyjna

Podstawowe operacje:

$$A = m_a \cdot b^{c_a}; \quad B = m_b \cdot b^{c_b}$$

$$A + B = (m_a \cdot b^{c_a - c_b} + m_b) \cdot b^{c_b},$$

$$A - B = (m_a \cdot b^{c_a - c_b} - m_b) \cdot b^{c_b},$$

$$A \cdot B = (m_a \cdot m_b) \cdot b^{c_a + c_b},$$

$$\frac{A}{B} = \left(\frac{m_a}{m_b} \right) \cdot b^{c_a - c_b}$$

Dodawanie i odejmowanie

1. Sprawdzenie zer
2. Wyrównanie mantys
3. Dodanie lub odjęcie mantys
4. Normalizacja wyniku

Przykład:

$(123 \times 10^{0})+(456 \times 10^{-2})=(123 \times 10^{0})+(4{,}56 \times 10^{0}) =$
$= 127{,}56 \times 10^{0}$

Mnożenie i dzielenie

1. Sprawdzenie zer
2. Dodanie wykładników i odjęcie wartości przesunięcia od sumy
3. Sprawdzenie przepełnienia lub niedomiaru wykładnika
4. Mnożenie mantys z uwzględnieniem znaków (postać znak-moduł!)
5. Zaokrąglanie i normalizacja wyniku iloczynu

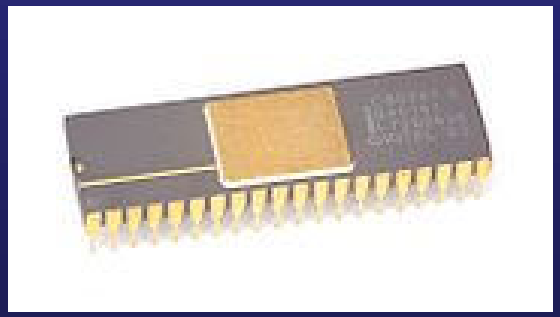
Jednostka zmiennoprzecinkowa

- Odpowiada za wykonywanie operacji na liczbach rzeczywistych
- Obecnie zintegrowany w procesorach ogólnego przeznaczenia (niekoniecznie w systemach wbudowanych!)
- W systemach bez koprocesora konieczna emulacja za pomocą biblioteki na liczbach stałopozycyjnych (fixed-point), realizowana przez ALU

Koprocesor x87 (np. 80287)

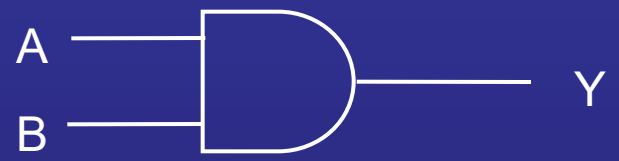
- Osiem rejestrów roboczych st0-st7 zorganizowanych w stos
- Osobne rejestry znaczników i flagowe
- Przykładowe operacje:

```
fld1
fadd st1
fxch st3
```



Układy logiczne (1)

Podstawowe bramki:

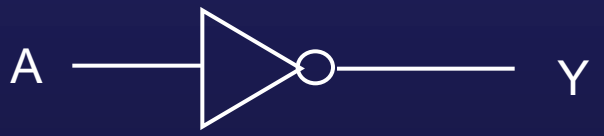


Y = A *AND* B

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Y = A *OR* B

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

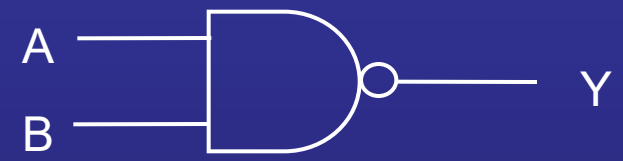


A = *NOT* Y

A	Y
1	0
0	1

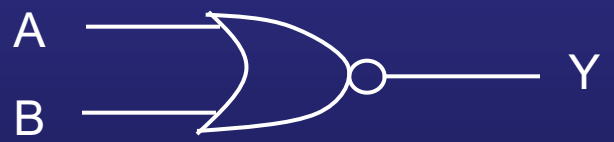
Układy logiczne (2)

Dodatkowe bramki:



Y = A *NAND* B

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0



Y = A *NOR* B

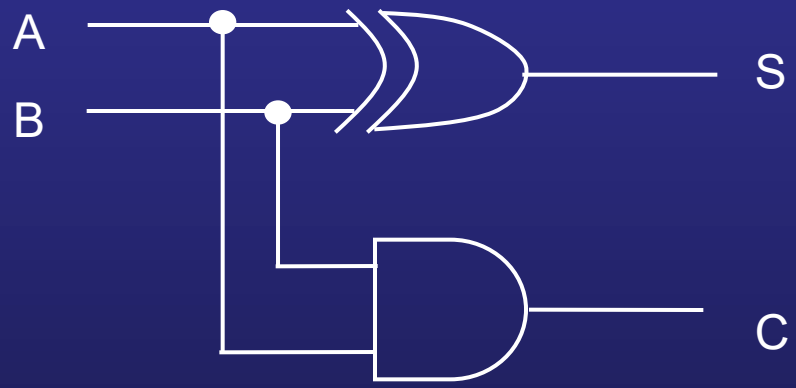
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Y = A *XOR* B

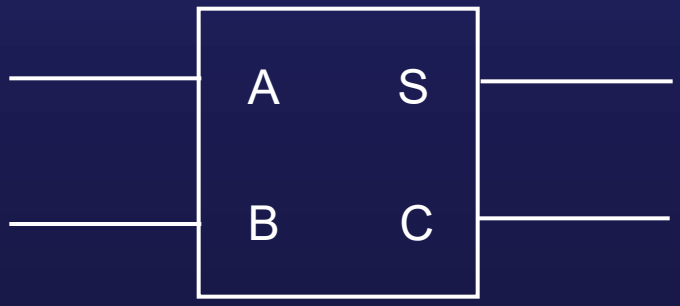
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Układy logiczne (3)

Półsumator

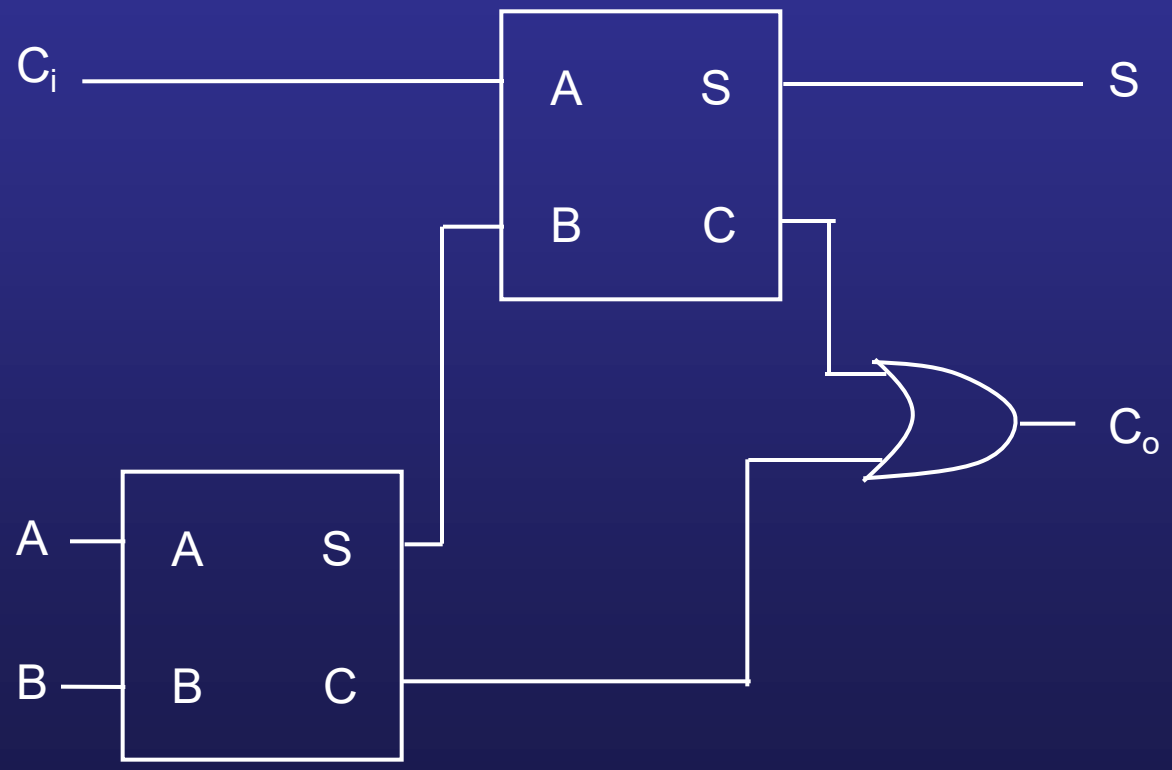


A	B	S	C
0	0	0	0
0	1	1	0
1	0	-	-
1	1	-	-



Układy logiczne (4)

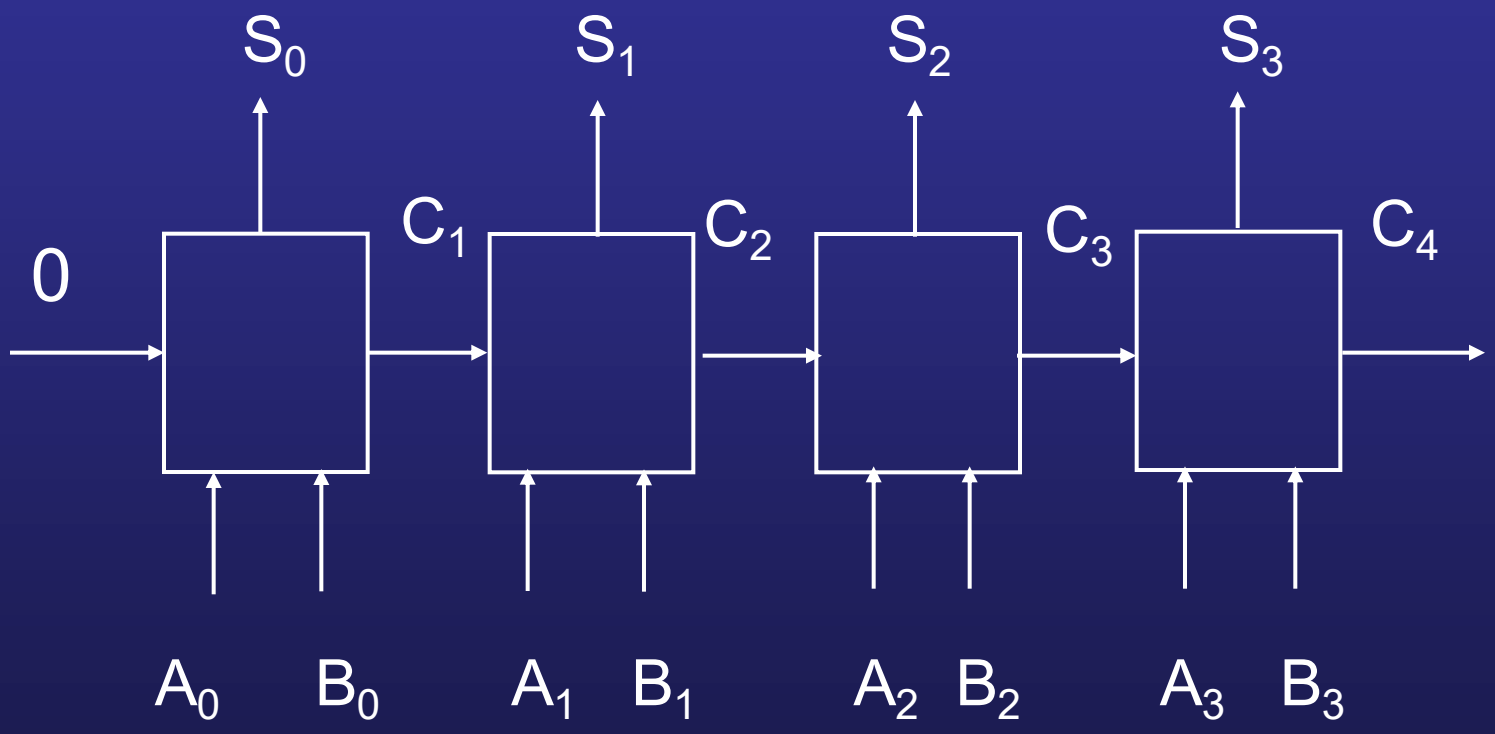
Sumator



A	B	C_i	S	C_o
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Układy logiczne (5)

Sumator 4-bitowy



Organizacja i Architektura Komputerów

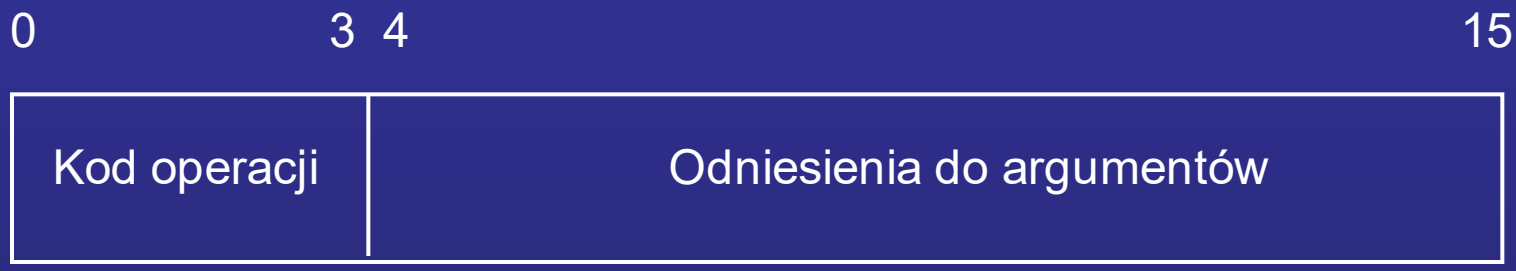
Wykład nr 4: Lista rozkazów procesora

Piotr Bilski

Wykonywanie programu przez komputer

- Procesor wykonuje rozkazy maszynowe (zrozumiałe dla niego)
- Programista tworzy program w języku symbolicznym, wysokiego lub niskiego poziomu
- W czasie kompilacji następuje zamiana rozkazów języka symbolicznego na rozkazy języka maszynowego

Elementy rozkazu maszynowego



- Kod operacji
- Wskazanie na argumenty (dane wejściowe operacji)
- Odniesienie do wyniku (jeśli uzasadnione)
- Wskazanie na następny rozkaz

Argumenty i wyniki przechowywane są w:

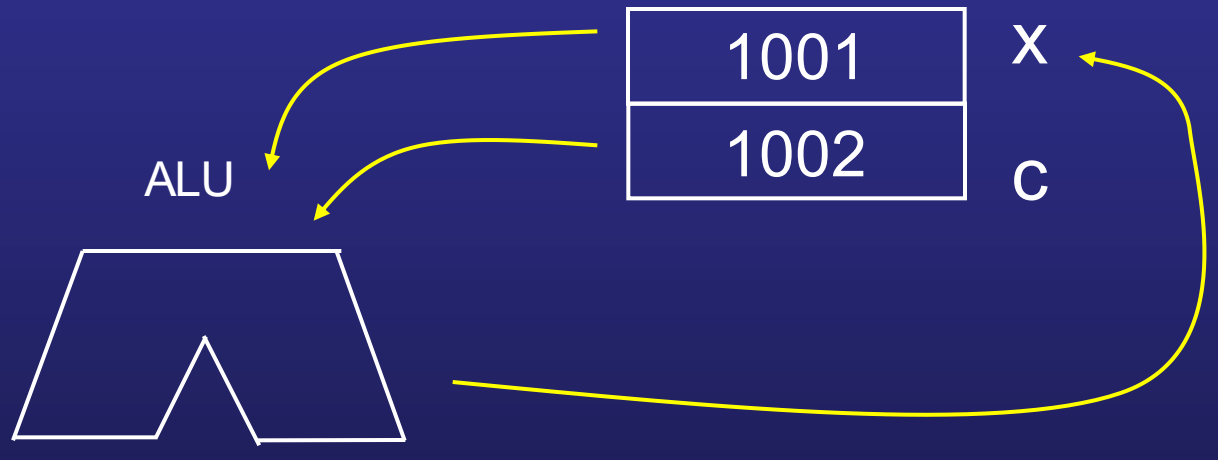
- Pamięci (głównej, podręcznej, wirtualnej)
- Rejestrach procesora (akumulator, rejestry ogólnego przeznaczenia)
- Urządzeniach wejścia/wyjścia (dysk twardy, drukarka)

Rodzaje rozkazów

- Przetwarzanie danych (operacje arytmetyczne i logiczne)
- Przechowywanie danych (rozkazy związane z dostępem do pamięci)
- Przemieszczanie danych (operacje wejścia/wyjścia)
- Sterowanie (testowanie wyników, rozgałęzienia w kodzie)

Związek pomiędzy rozkazami symbolicznymi i maszynowymi

x = x + c;



LOAD 1001

ADD 1002

STORE 1001

Liczba adresów w rozkazie

Rozkaz	Działanie
SUB Y,A,B	Y←A-B
MPY T,D,E	T←D*E
ADD T,T,C	T←T+C
DIV Y,Y,T	Y←Y/T

3 adresy

Rozkaz	Działanie
MOVE Y,A	Y←A
SUB Y,B	Y←Y-B
MOVE T,D	T←D
MPY T,E	T←T*E
ADD T,C	T←T+C
DIV Y,T	Y←Y/T

2 adresy

Rozkaz	Działanie
LOAD D	AC←D
MPY E	AC←AC*E
ADD C	AC←AC+C
DIV Y	AC←AC/Y

1 adres

Y=(A-B)/(C+D*E)

Liczba adresów w rozkazie (c.d.)

a = b + c

- Trzy adresy:

  ADD a,b,c

- Dwa adresy:

  MOVE a,b

  ADD a,c

- Jeden adres:

  LOAD b

  ADD c

  STOR a

Problemy projektowania listy rozkazów

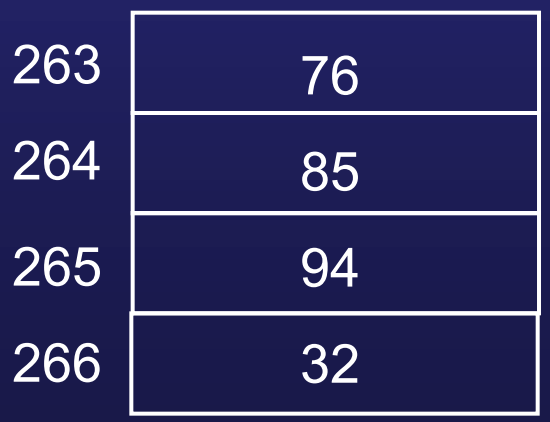
- Ile i jakie operacje do wykonania przez procesor?
- Jakie rodzaje danych (argumenty, wyniki)?
- Jaki format rozkazu (długość, liczba adresów)?
- Ile rejestrów i jakie?
- Jakie tryby adresowania?

Argumenty

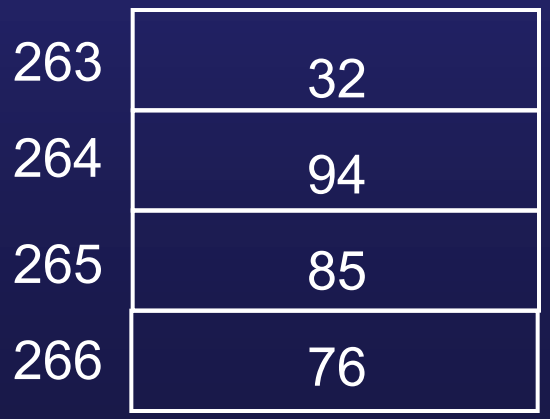
- Adresy (liczby całkowite bez znaku)
- Liczby (dane numeryczne) – całkowite, zmiennopozycyjne, dziesiętne
- Znaki (kody ASCII / IRA, EBCDIC itp.)
- Dane logiczne (pojedyncze bity)

Przechowywanie danych w komputerze

- Zapis danych wielobajtowych w pamięci może być cienko-, grubo- lub dwukońcowy (little endian, big endian, bi-endian)
- Różnica pomiędzy zapisem danych polega na kolejności zapisu bajtów danej w pamięci, np. liczba szesnastkowa 76859432h da się zapisać:



Big endian



Little endian

Grubo- a cienkokońcowość

Grubokońcowość

- Łatwiej sortować wyrównane łańcuchy znaków
- Drukowanie znaków kodu ASCII bez przekształceń
- Liczby całkowite i znaki w tym samym porządku
- Sun SPARC, maszyny RISCowe, Motorola 680x0

Cienkokońcowość

- Łatwe przekształcenie liczby dłuższej na krótszą
- Łatwiej przeprowadzane są działania arytmetyczne
- Intel 80x86, Pentium, Alpha

Dwukońcowość

- Oba standardy zrozumiałe
- Wykorzystywany w PowerPC

Przykłady grubo- i cienkokońcowości w formatach plików

Big endian:

- **Adobe Photoshop**
- **IMG (GEM Raster)**
- **JPEG**
- **MacPaint**
- **SGI (Silicon Graphics)**
- **Sun Raster**

Little endian:

- **BMP (Windows, OS/2 Bitmaps)**
- **GIF**
- **PCX (PC Paintbrush)**
- **TGA (Targa)**
- **Microsoft RTF (Rich Text Format)**

Bi-endian:

- **Microsoft RIFF (.WAV & .AVI)**
- **TIFF**
- **XWD (X Window Dump)**

Rodzaje danych Pentium

- Dane o długości wielokrotności bajtu (słowo – 2 B, podwójne słowo – 4 B itd.)
- Zgodność formatów z normą IEEE 754
- Nie ma konieczności przechowywania danych pod adresami wyrównanymi parzyście
- Liczby całkowite bez znaku (8, 16, 32, 64 bity) - adresy
- Liczby całkowite ze znakiem (8,16, 32, 64 bity) w reprezentacji U2
- Liczby zmiennopozycyjne (pojedyncza, podwójna i podwójna rozszerzona precyzja)

Rodzaje danych Pentium (c.d.)

- Nieupakowana binarna reprezentacja liczb dziesiętnych (jedna cyfra w bajcie)
- Upakowana binarna reprezentacja liczb dziesiętnych (dwucyfrowa)
- Wskaźnik (adres 32-bitowy)
- Łańcuch bajtów

Rodzaje danych PowerPC

- Dane o długości 8, 16, 32, 64 bitów
- Wyrównanie adresów danych do parzystego bajtu nie jest wymagane (choć czasem stosowane)
- PowerPC jest typu bi-endian
- Przechowywane: liczby bez znaku i ze znakiem (bajt, półsłowo, słowo, podwójne słowo), liczby zmiennopozycyjne (IEEE 754), łańcuch bajtów (do 128 B)

Klasyfikacja operacji

- Transfer danych (np. STORE, LOAD, SET, PUSH, POP)
- Arytmetyczne (ADD, SUB, NEG, INC, MUL)
- Logiczne (AND, OR, NOT, TEST, SHIFT, ROTATE)
- Przeniesienie sterowania (JUMP, HALT, EXEC)
- Wejście/wyjście (READ, WRITE)
- Konwersja (TRANS, CONV)

Transfer danych

- Cel: przeniesienie danych z jednego miejsca w drugie
- Wymaga: określenia lokacji w pamięci (adres wirtualny?), sprawdzenia pamięci podręcznej, wydania polecenia zapisu/odczytu
- Przykładowe polecenia: LOAD, STORE (w wersjach short, long, halfword itp.)

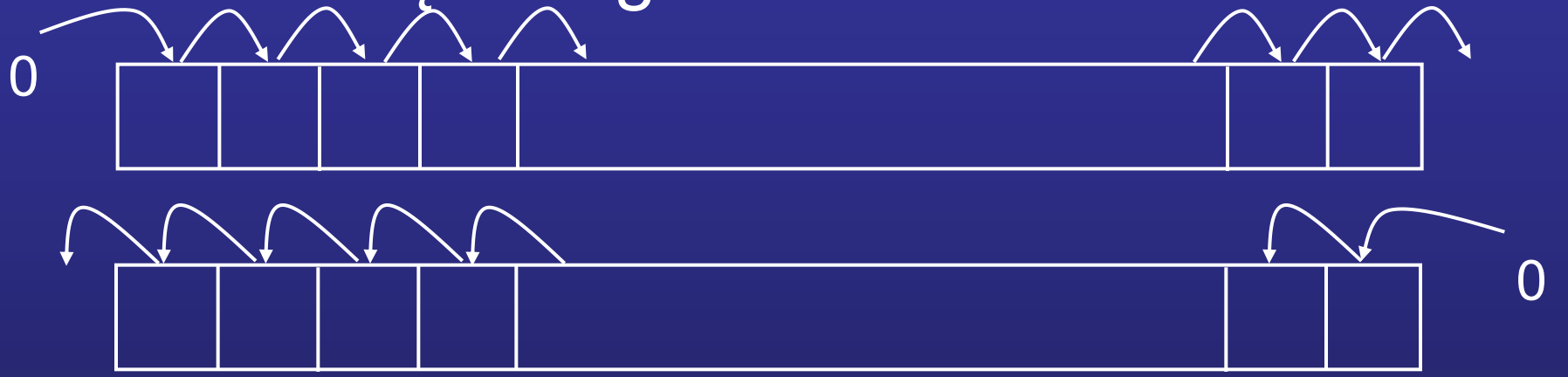
Operacje logiczne

- Argumenty traktowane są jako ciąg bitów
- Najczęściej stosowane operacje: AND, OR, XOR, NOT
- Ciągi bitowe traktowane jako maski:

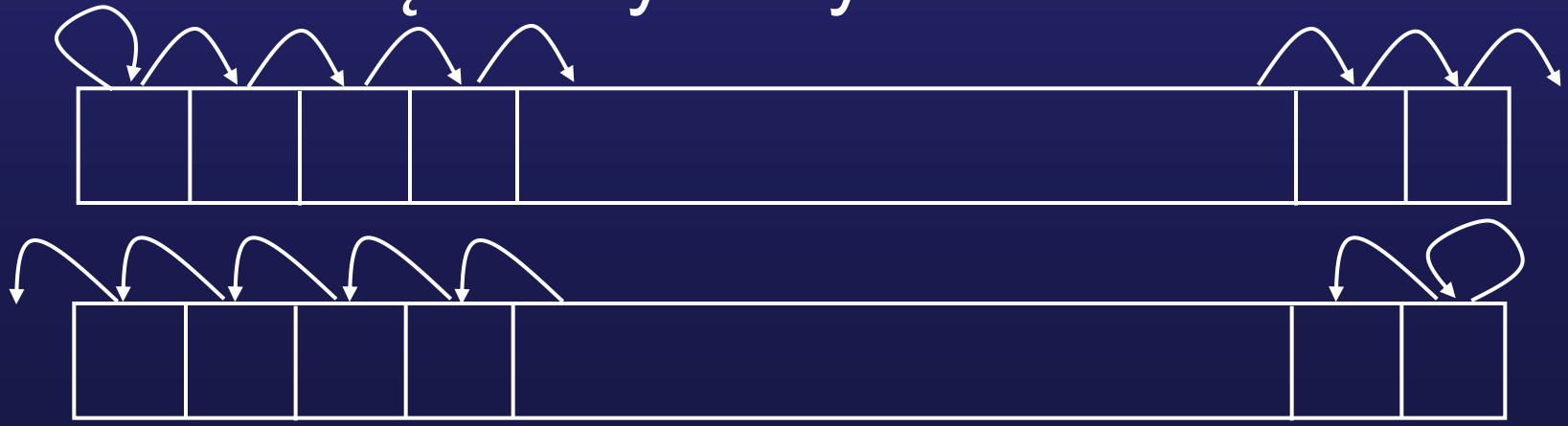
A_1 = 10100101	A_1 = 10100101
AND	XOR
A_2 = 11110000	A_2 = 11111111
10100000	01011010

Operacje logiczne (c.d.)

- Przesunięcie logiczne



- Przesunięcie arytmetyczne



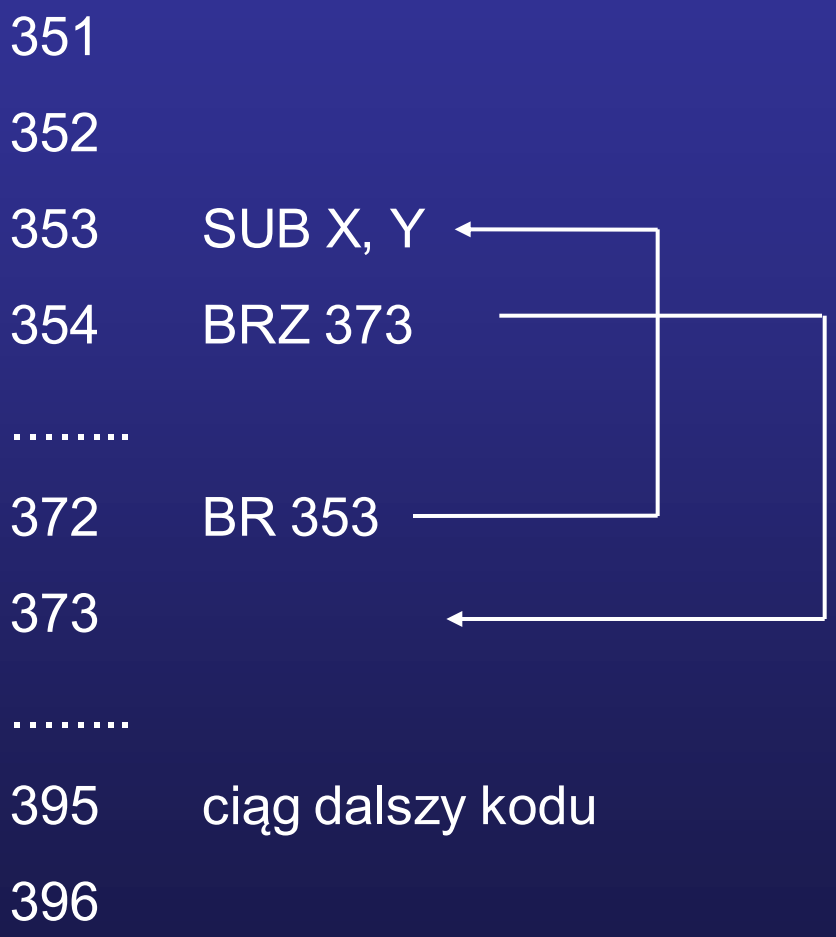
Przekazywanie sterowania

- Związane jest z kolejnością wykonywania rozkazów
- Obejmują skoki oraz wykonywanie jednej operacji w pętli
- Przekazywanie sterowania może mieć charakter warunkowy i bezwarunkowy

Rozgałęzienie warunkowe

- Istnieje wielobitowy kod przechowujący wyniki operacji stanowiących warunek dokonania rozgałęzienia, np. ze względu na znak wyniku, wystąpienie przepełnienia lub wyzerowanie wyniku
- Druga metoda to zawarcie warunku rozgałęzienia w instrukcji skoku
- Rozgałęzienie może być stosowane w obu kierunkach

Przykład rozgałęzienia



BRZ – wykonaj skok, jeśli wynik jest zerem

BR – wykonaj skok bezwarunkowo

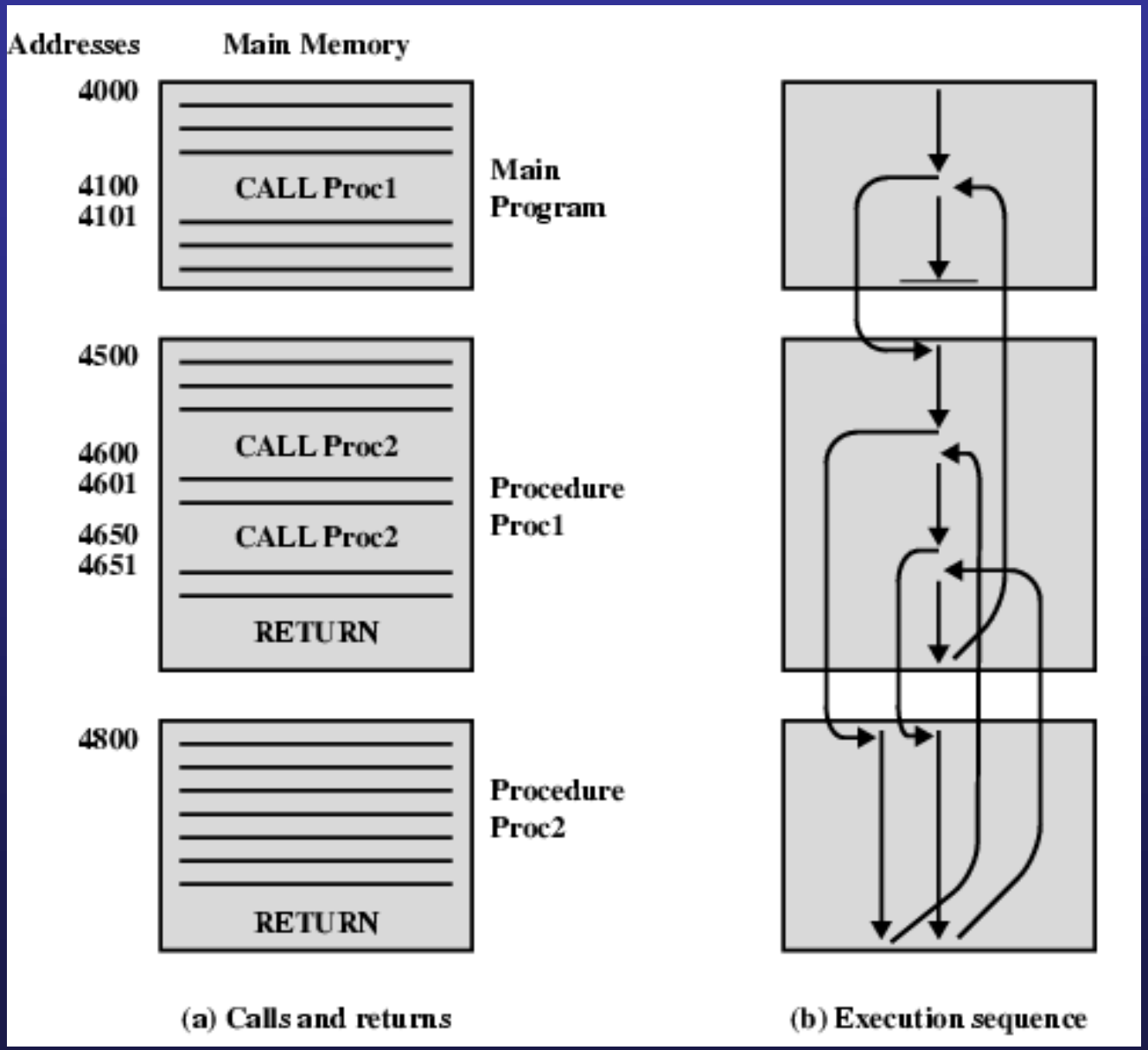
Procedury

- Są to fragmenty kodu stanowiące osobne moduły
- Ich wykorzystanie umożliwia zwiększenie elastyczności kodu
- Wymagają dwóch rozkazów: rozkaz wywołania i powrotu
- Ta sama procedura może być wywoływana wielokrotnie z różnych lokacji
- Możliwe jest zagnieżdżanie procedur

Procedura a miejsce powrotu

- Procedura może być wywołana z wielu lokacji
- Możliwe jest zagnieżdżanie wywołań
- Wywołanie procedury wiąże się z przechowaniem rozkazu powrotu:
  - w rejestrze
  - na początku wywoływanej procedury
  - na stosie (najlepsza opcja, umożliwia działanie procedur wielowejściowych, np. rekurencja)

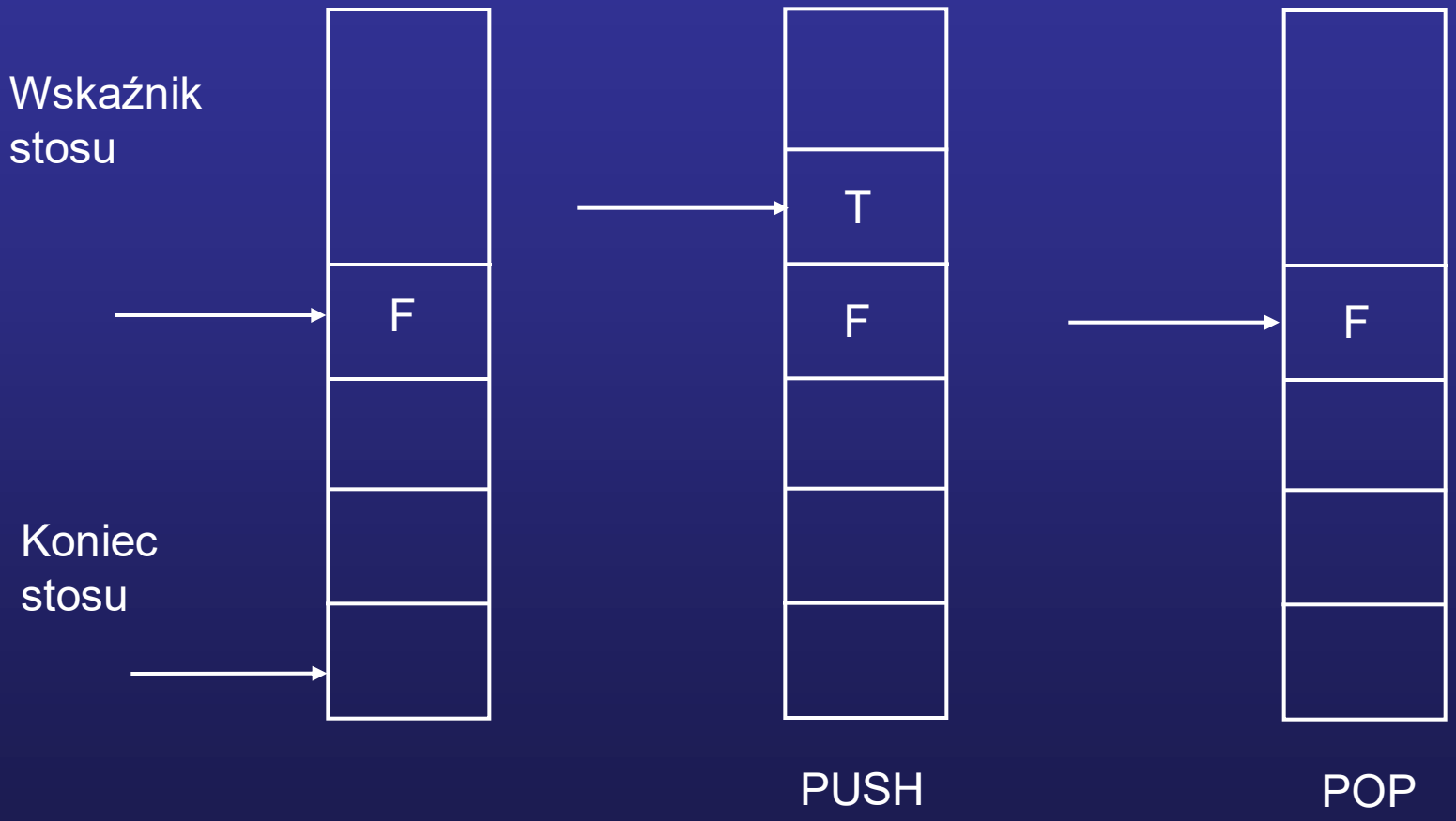
Wywołanie procedury



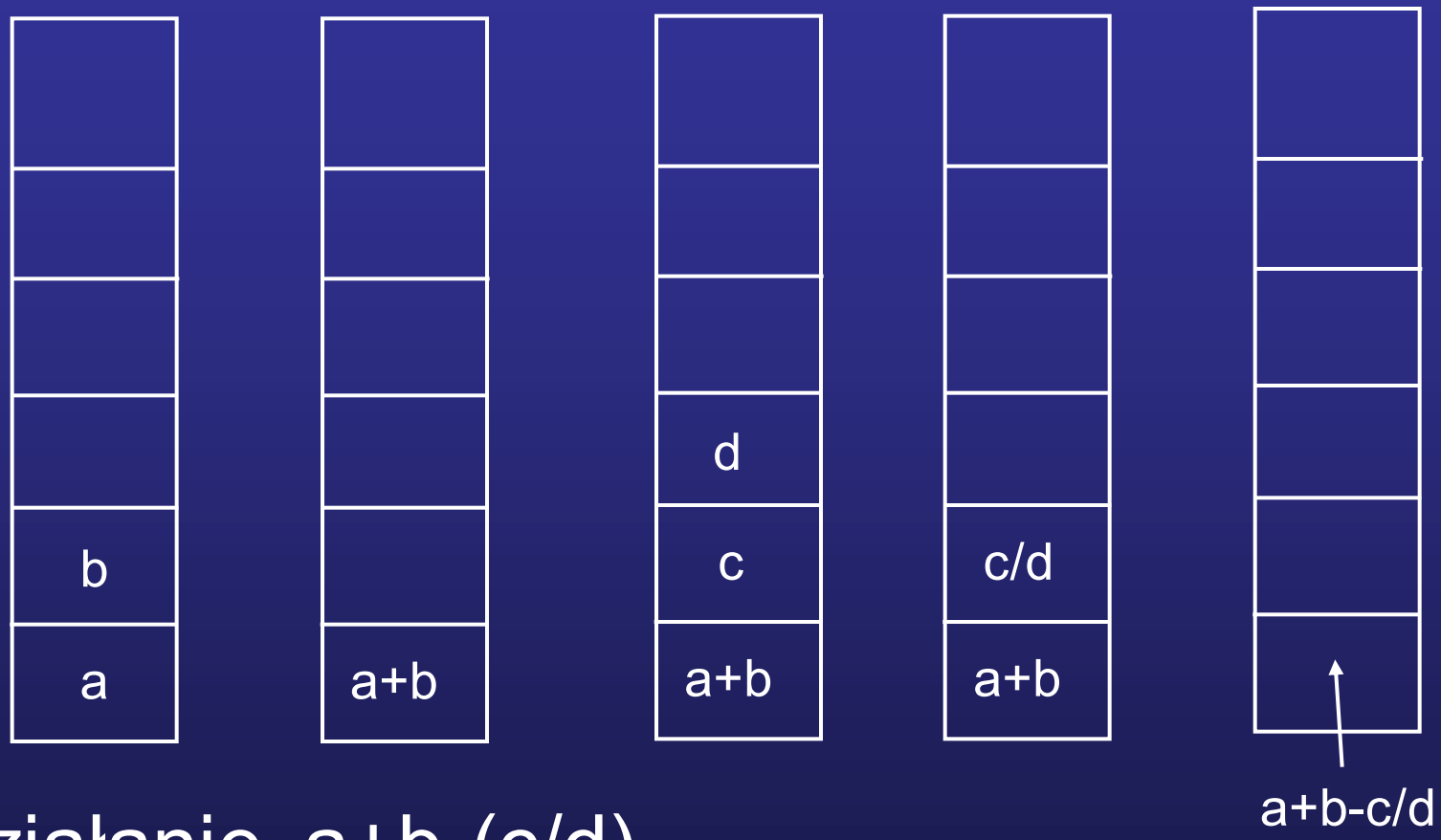
Stos

- Jest to wyróżniona porcja pamięci przechowująca dane, zorganizowana w strukturze kolejki LIFO
- W wielu procesorach istnieje rejestr działający jako wskaźnik stosu (Motorola 68000)
- Główne operacje na stosie: PUSH, POP

Przykład implementacji stosu



Wykonywanie działań na stosie



- Działanie a+b-(c/d)
- W odwrotnej notacji polskiej: ab+cd/-

Ramka stosu

- Jest to zestaw parametrów łącznie z adresem powrotu
- Umożliwia wywoływanie procedur zagnieżdżonych z przechowywaniem parametrów wejściowych i wyjściowych na stosie

Ilustracja ramki stosu

Procedura A

Procedura A wywołała B

Ramka stosu w procesorze Pentium

- Wykorzystywana przez polecenie ENTER, CALL
- Rozkaz ENTER wspomaga kompilatory w realizacji procedur zagnieżdżonych
- Działanie odwrotne wykonuje polecenie LEAVE
- Wskaźnik ramki przechowywany jest w rejestrze EBP, wskaźnik stosu w rejestrze ESP
- Przykład wywołania CALL:

```
PUSH EBP

MOV EBP, ESP

SUB ESP, miejsce_w_pamięci
```

Rozkazy MMX

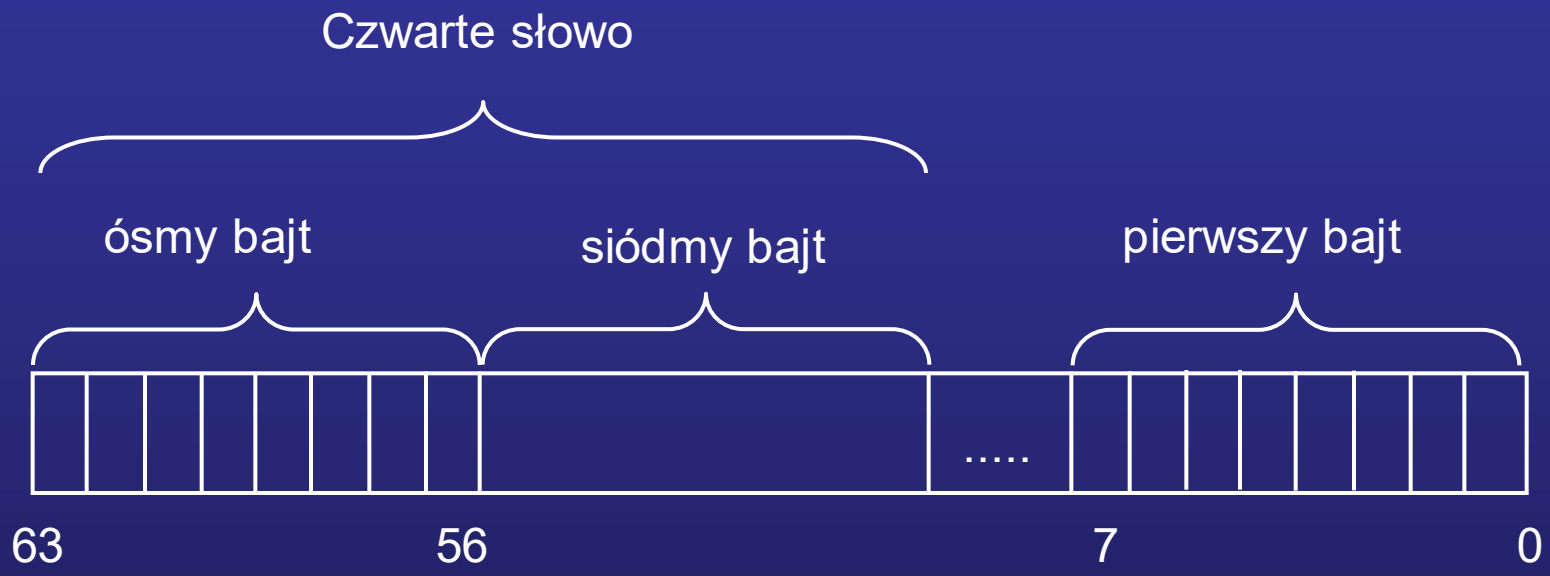
- Wprowadzone w 1996 r. do procesorów Pentium
- W pierwszej wersji było to 57 rozkazów typu SIMD
- Służą do wykonywania operacji na liczbach całkowitych
- Przeznaczenie – zastosowania multimedialne (gry komputerowe, obróbka grafiki i dźwięku)
- MMX wykorzystuje cztery nowe rodzaje danych: upakowany bajt, upakowane słowo, upakowane podwójne słowo, upakowane poczwórne słowo

Przykłady rozkazów MMX

- Arytmetyczne: PADD, PMUL, PMADD
- Logiczne: PAND, PNDN, POR, PXOR
- Porównanie: PCMPEQ, PCMPGT
- Konwersja: PUNPCKH, PUNPCKL

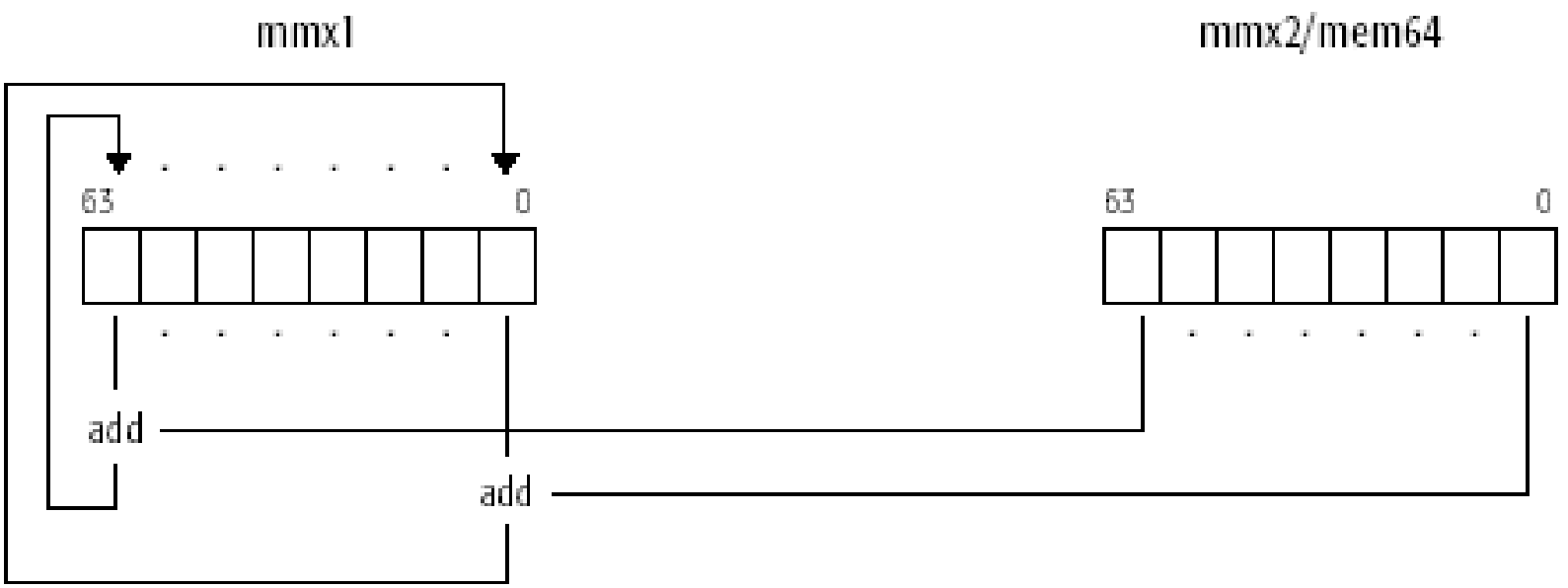
- Wszystkie rozkazy mają przyrostki decydujące o danych na jakich wykonywana jest operacja: B, W, D, Q

Dodatkowe rejestry MMX



- Osiem 64-bitowych rejestrów MM0 do MM7
- Ze względu na kompatybilność wsteczną, rejestry MMX są widoczne przez starsze oprogramowanie jako rejestry zmiennoprzecinkowe

Przykładowa operacja MMX



Arytmetyka MMX

- Nasycenie zamiast przepełnienia

```
  1111 0000 0000 0000
+0011 0000 0000 0000
-----------------------
 10010 0000 0000 0000      przepełnienie

  1111 0000 0000 0000
+0011 0000 0000 0000
-----------------------
 10010 0000 0000 0000

  1111 1111 1111 1111      nasycenie
```

Korzyści z wykorzystania MMX

Operacja	Przyspieszenie*
Efekt echa	5,9
Transpozycja macierzy	2
Operacje arytmetyczne i logiczne na wektorach	6
Rysowanie fraktali (2D)	1,5
Biliniowe mapowanie teksturami (3D)	7
Filtr medianowy	3,8
Transformata Haara 2x2	2,2
Obliczenie normy L1	3,3
Przekształcenie 3D	3,1

* - w stosunku do kodu w C wykorzystującego tradycyjną architekturę

Instrukcje SSE (Streaming SIMD Extensions)

- Wprowadzone w 1999 r. (Pentium 3)
- Nowe 70 instrukcji do operacji na liczbach zmiennoprzecinkowych
- Dodatkowe 8 rejestrów 128-bitowych, adresowanych bezpośrednio: XMM0 – XMM7 (oraz rejestr kontrolny MXCSR). Każdy rejestr przechowuje 4 liczby zmiennoprzecinkowe 32-bitowe

SSE (c.d.)

- Nowy typ danych: 4-elementowy wektor liczb zmiennoprzecinkowych pojedynczej precyzji
- Operacje mogą być upakowane (PS - na wszystkich liczbach wektorów), lub skalarne (SS - tylko na pierwszych elementach)
- Przykład:

xmm0 = [X1 X2 X3 X4] xmm1 = [Y1 Y2 Y3 Y4]

ADDPS(xmm0,xmm1) =

[X1+Y1 X2+Y2 X3+Y3 X4+Y4]

Instrukcje 3DNow!

- Wprowadzone w 1997 r. przez firmę AMD
- Zestaw 21 nowych instrukcji do obliczeń zmiennoprzecinkowych typu SIMD
- Wykorzystanie w zastosowaniach multimedialnych (grafika wysokiej rozdzielczości, gry komputerowe, CAD/CAM)
- Rozszerzenia: Enchanced 3DNow!, 3DNow Professional

Instrukcje SSE2

- Wprowadzone w 2001 r. (Intel Pentium IV, Athlon 64, Sempron 754, Transmeta Efficeon)
- Zestaw dodatkowych 144 instrukcji, do których obsługi służy 16 128-bitowych rejestrów (XMM0 – XMM15) – do operacji na liczbach całkowitych
- Możliwe operacje na liczbach zmiennoprzecinkowych 64-bitowych (koprocesory x87 działają na liczbach 80-bitowych) i całkowitych 128-bitowych

Kolejne zestawy instrukcji

- SSE3 (Prescott New Instructions) – 13 nowych rozkazów, w tym do arytmetyki zespolonej (od 2004 r., Pentium IV Prescott, Athlon 64 E)
- SSSE3 (Supplemental Streaming SIMD Extension 3) – 16 nowych instrukcji na liczbach całkowitych (od 2005 r. Xeon, Intel Core 2, AMD Phenom)
- SSE4 – 54 nowe instrukcje w dwóch grupach (47 i 7), w tym rozkazy całkowitoliczbowe modyfikujące rejestr EFLAGS (nowość!), obecne w Intel Core 2, Celeron Controe, Penryn

Kolejne zestawy instrukcji (c.d.)

- SSE5 – planowane przez AMD do wprowadzenia w 2009 r.: nowe rozkazy zmienno- i stałoprzecinkowe. Ostatecznie wprowadzono trzy grupy rozkazów: XOP, FMA4, CVT16 (kompatybilne z AVX). Zaimplementowane w procesorach Buldozzer (2011). Instrukcje zawierają nawet 4 argumenty!
- AVX (Advanced Vector Extensions) – wprowadzone przez Intela w 2011 r. w drugiej generacji rodziny Core: 16 nowych rejestrów 256-bitowych YMM0-YMM15 + 19 instrukcji działających wyłącznie na tych rejestrach (przechowujących 8 liczb typu float lub 4 liczby typu double)

Co dalej?

- AVX-2 – rozszerzenie zestawu AVX w procesorach Haswell (4. generacja rodziny Core, obecne w AMD Ryzen), 30 nowych instrukcji działających na rejestrach YMM
- AVX-512 – od 2015 r. rozszerzenie AVX wymagające rejestrów ZMM (512 bitów, zawierają rejestry YMM i poprzednie SSE)
- AES-NI – wprowadzone w 2008 roku, umożliwiają sprzętowe szyfrowanie i deszyfrowanie blokową metodą AES

Organizacja i Architektura Komputerów

Wykład nr 5: Sposoby adresowania

Piotr Bilski

Skąd potrzeba różnych trybów adresowania?

- Przestrzeń pamięci rzeczywistej lub wirtualnej jest zwykle większa od przestrzeni adresowanej przez argument (zbyt mała liczba bitów)
- Konieczny jest kompromis pomiędzy zakresem adresów i prostotą uzyskiwania adresu

Adres rzeczywisty (efektywny) a adres wirtualny

- Adres rzeczywisty jest adresem fizycznego miejsca w pamięci
- Adres wirtualny (efektywny - AE) jest adresem przechowywanym jako argument rozkazu
- Proces translacji adresu wirtualnego na rzeczywisty jest zależny od organizacji pamięci

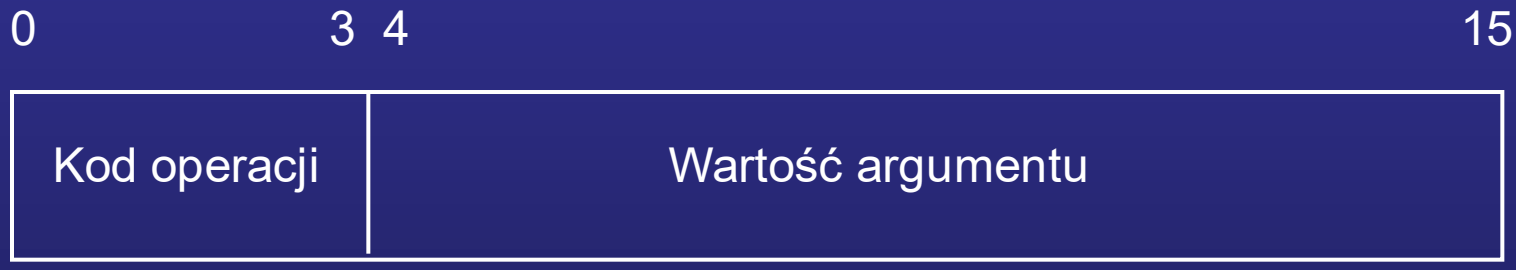
Konwencja zapisu

- AE – adres efektywny
- R – zawartość pola adresowego w rozkazie odnoszącym się do rejestru
- A – zawartość pola adresowego w rozkazie
- (X) – zawartość lokacji X

Tryby adresowania

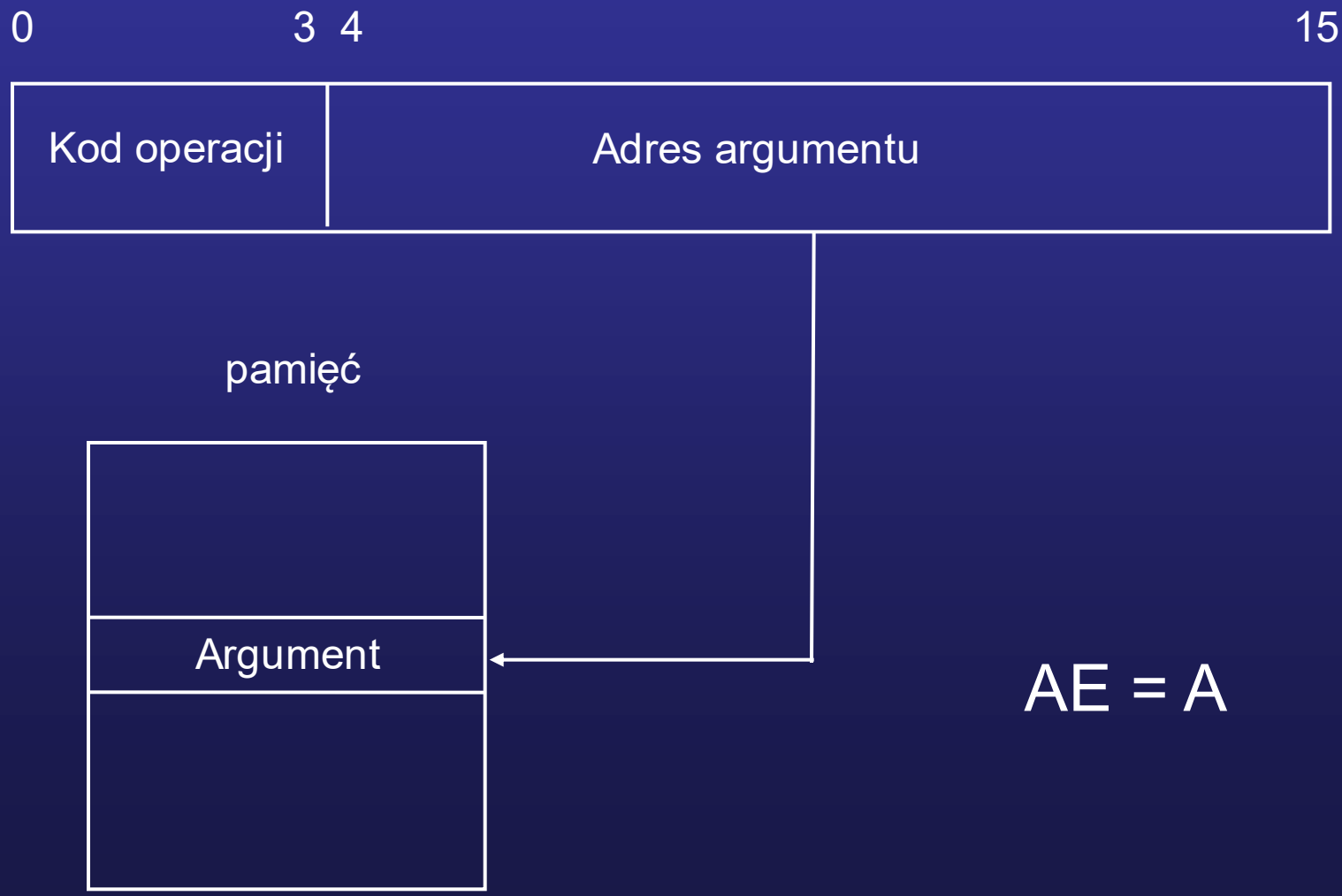
- Natychmiastowy
- Bezpośredni
- Pośredni
- Rejestrowy
- Rejestrowy pośredni
- Z przesunięciem
- Stosowy

Tryb natychmiastowy



Argument jest przechowywany w rozkazie w postaci liczby o reprezentacji U2. Liczba ta nie może być zbyt duża

Tryb bezpośredni



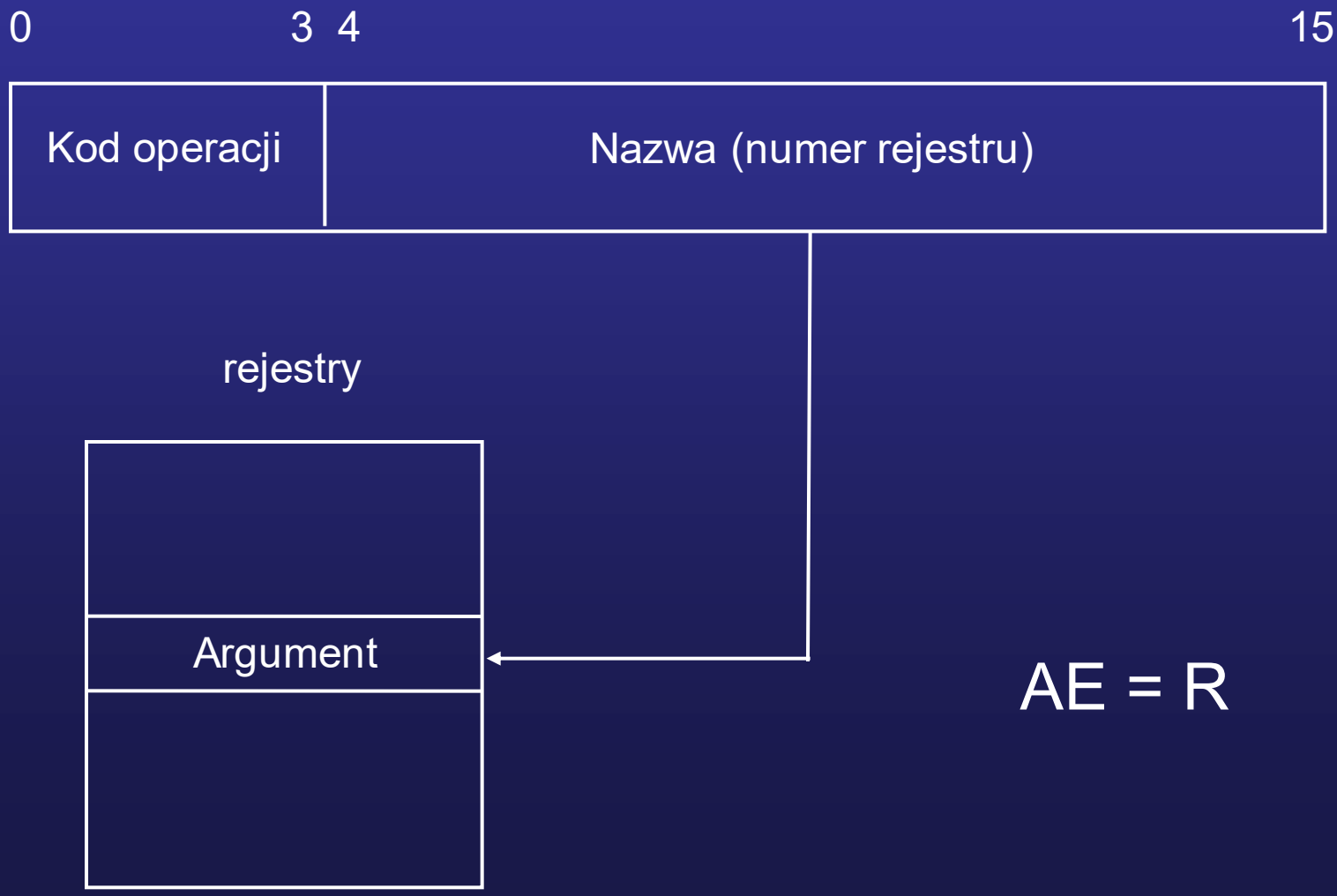
AE = A

Tryb pośredni

AE = (A)

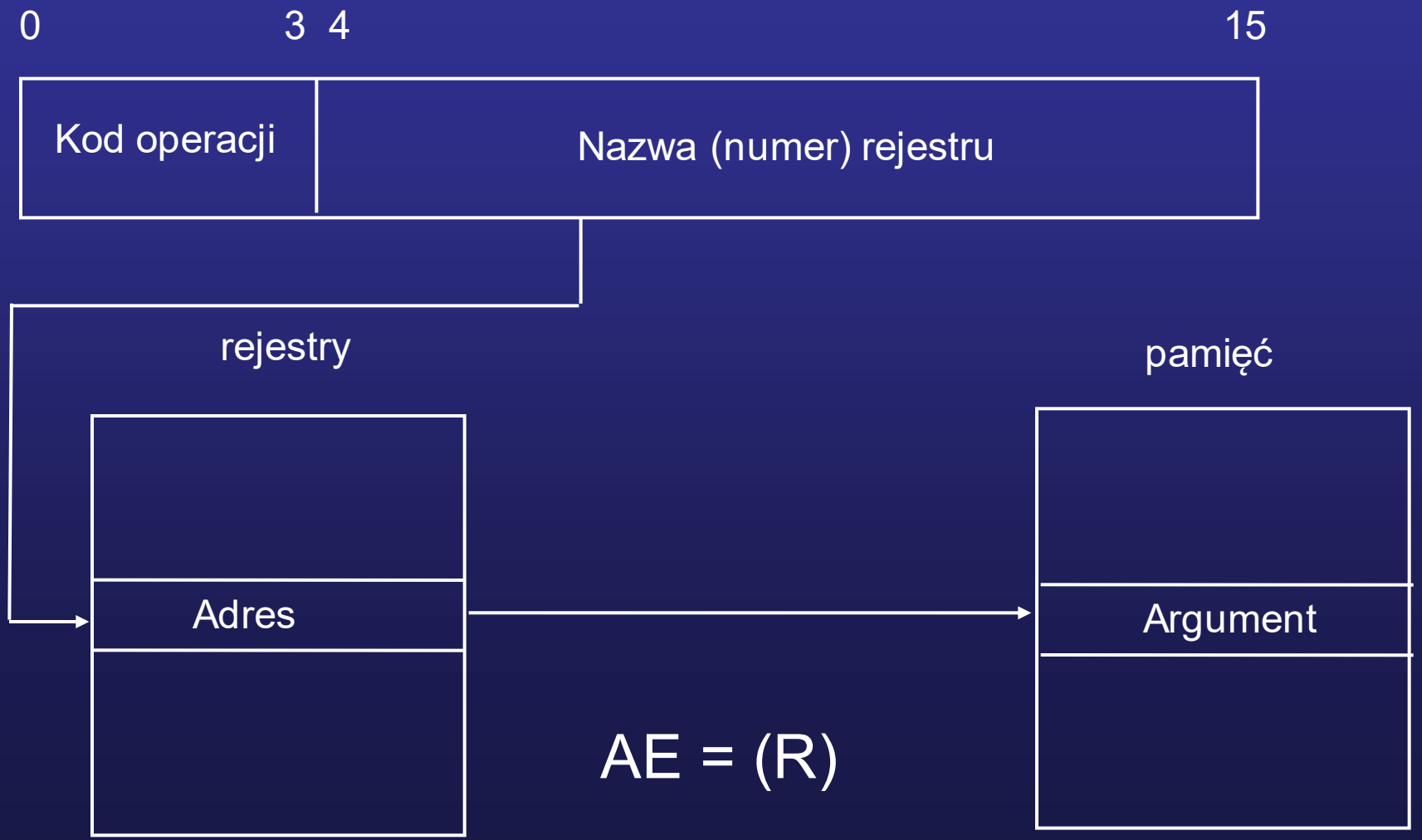
Możliwe zagnieżdżenia!

Tryb rejestrowy

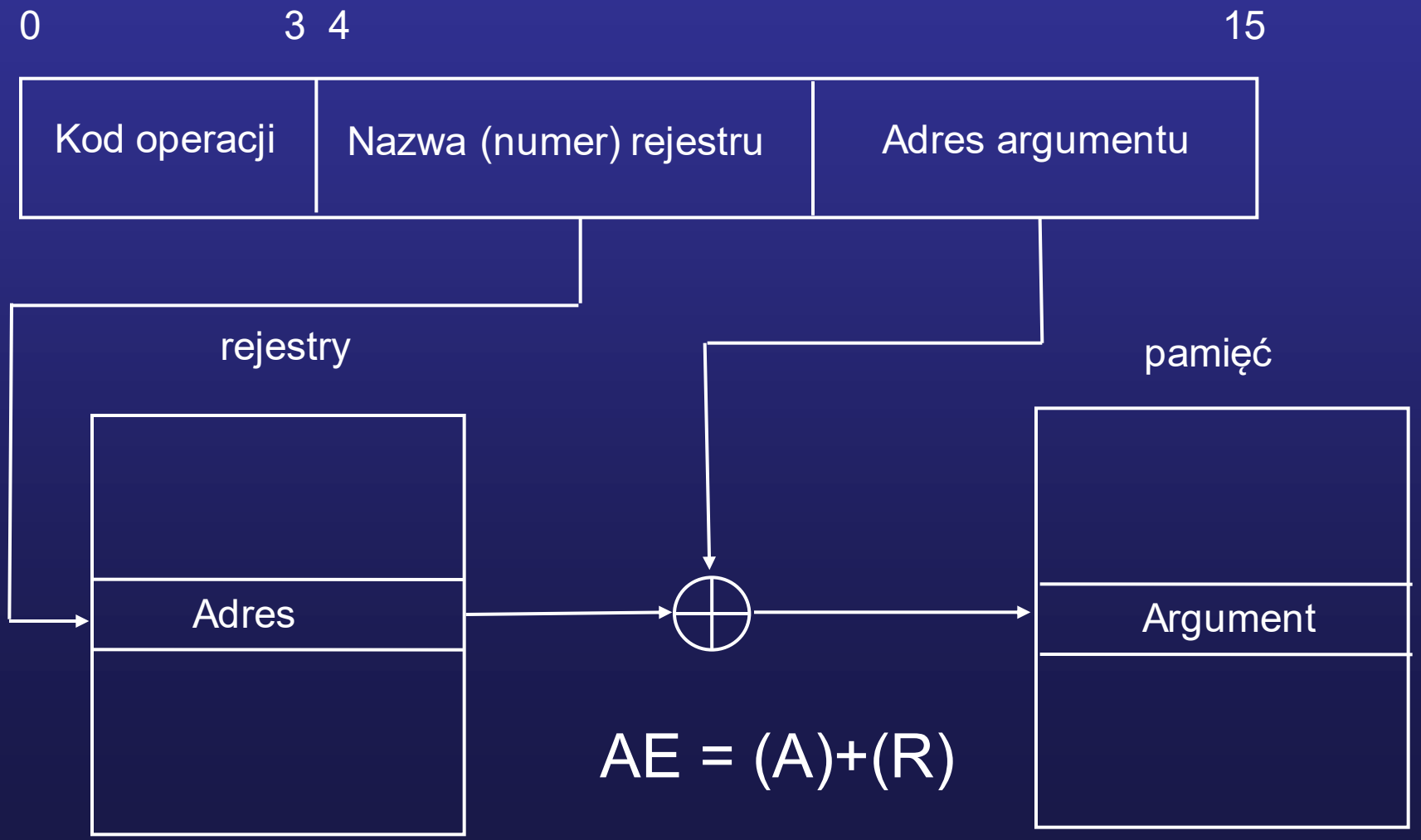


AE = R

Tryb rejestrowy pośredni



Tryb z przesunięciem

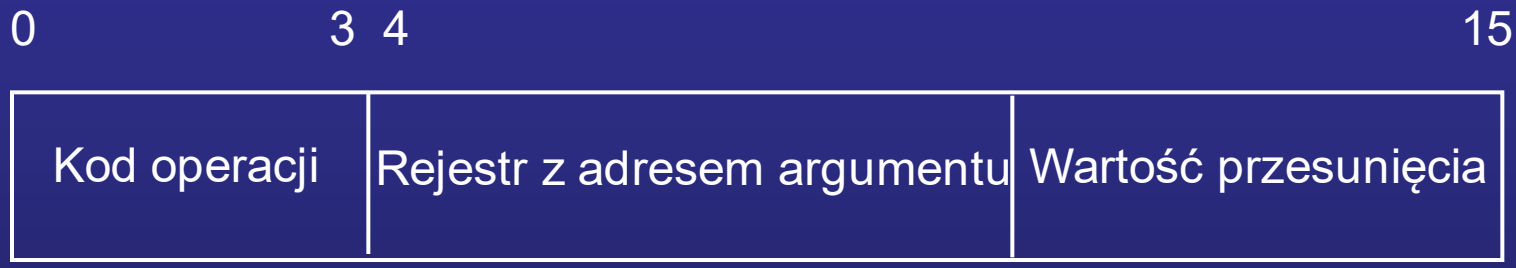


Adresowanie z przesunięciem c.d.

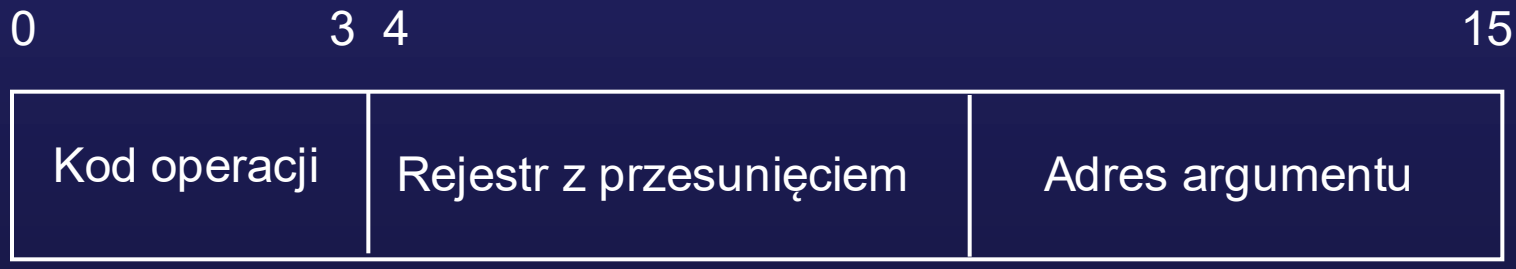
- Adresowanie względne (R=PC)
- Adresowanie z rejestrem podstawowym (wygodne dla pamięci segmentowanej) – AE=R+A
- Indeksowanie (interpretacja pola rejestru i pola adresowego odwrotna niż w przypadku adresowania z rejestrem podstawowym) – AE=A+R

Adresowanie z przesunięciem c.d.

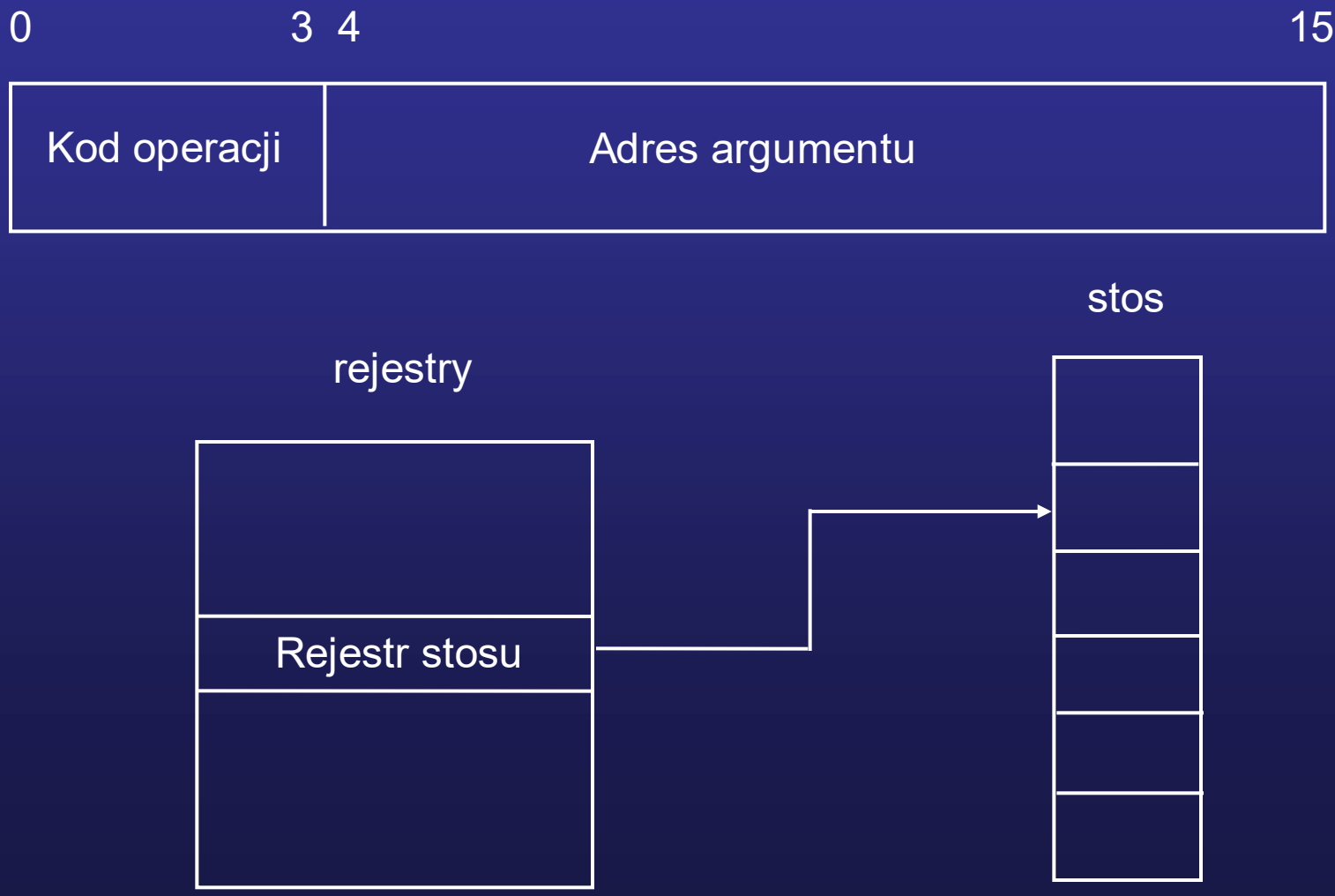
- Adresowanie z rejestrem podstawowym



- Indeksowanie (wykorzystuje dedykowany rejestr indeksowy)



Tryb stosowy



Typy adresów

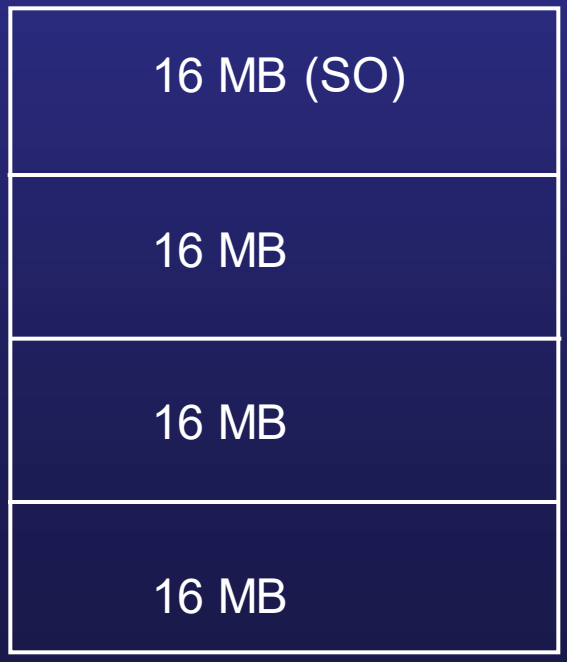
- Fizyczny – umiejscowienie adresu w pamięci
- Logiczny – umiejscowienie adresu względem początku programu
- Bazowy – adres początku programu

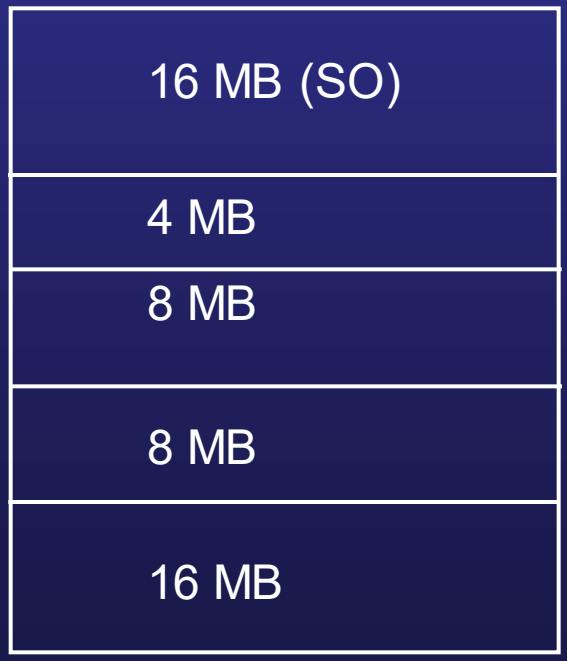
Organizacja pamięci z poziomu systemu operacyjnego

- Pamięć stronicowana (model niewidzialny dla programisty) – podzielona jest na małe fragmenty (ramki stron), które przydzielane są fragmentom (stronom) programów
- Pamięć segmentowana (model dostępny dla programisty) – zapewnia odrębną przestrzeń adresową dla każdego procesu

Partycjonowanie z ustalonym rozmiarem

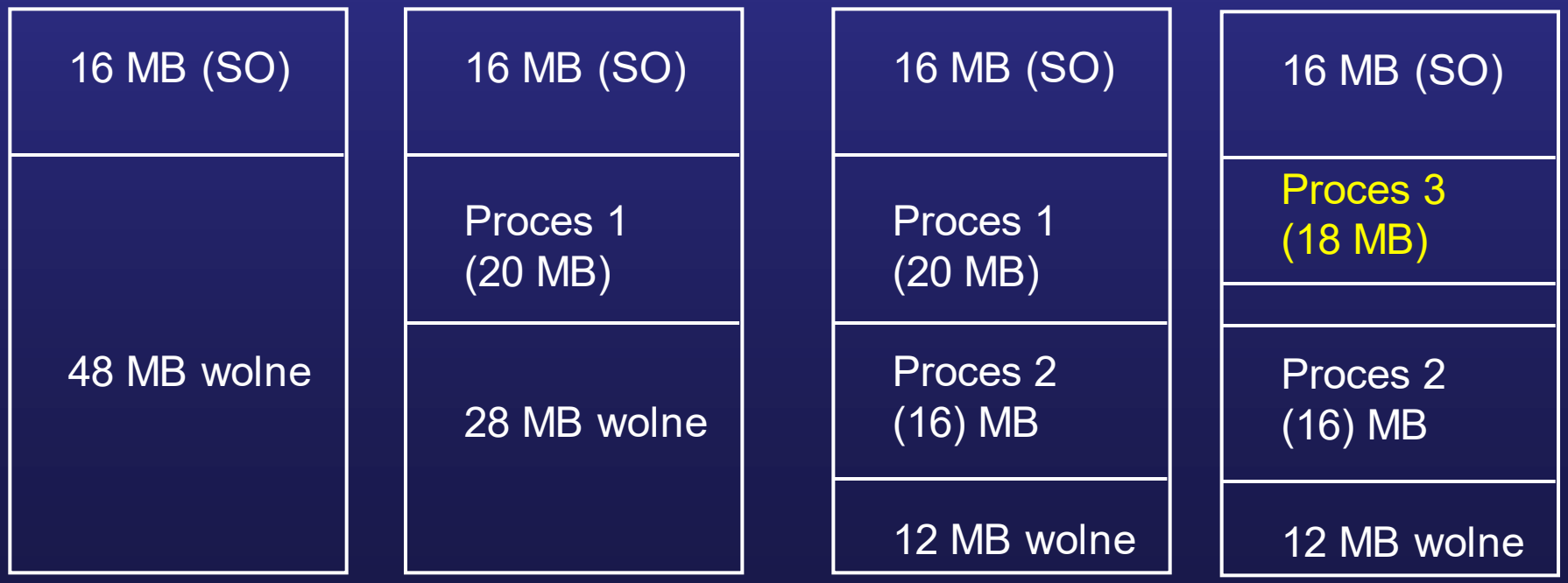
- Jest to podział pamięci na fragmenty o ustalonych rozmiarach (stałych lub zmiennych)





Partycjonowanie z rozmiarem dynamicznym

- Jest to podział pamięci na fragmenty o rozmiarach zależnych od zapotrzebowania procesów



Stronicowanie

Proces A:

Strona 0

Strona 1

Strona 2

Wolne ramki:

11

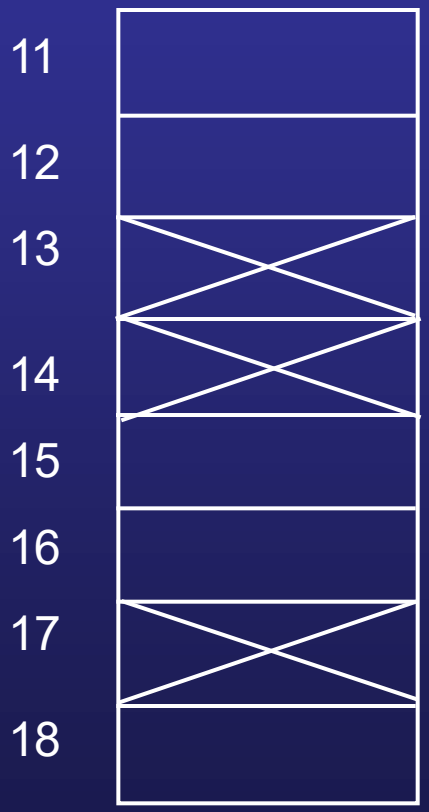
12

15

16

18

pamięć



Tablica stron procesu A:

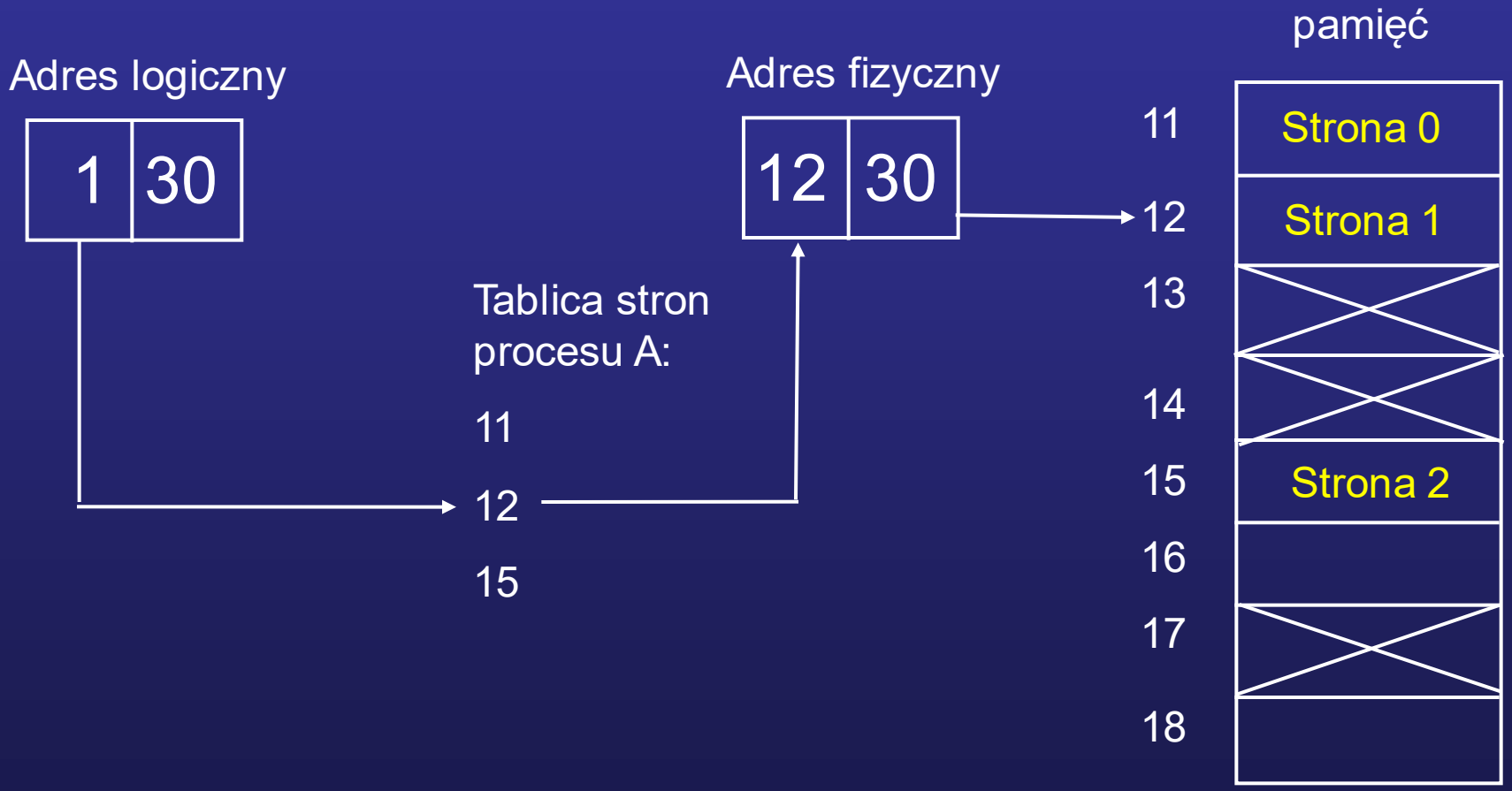
11

12

15

pamięć

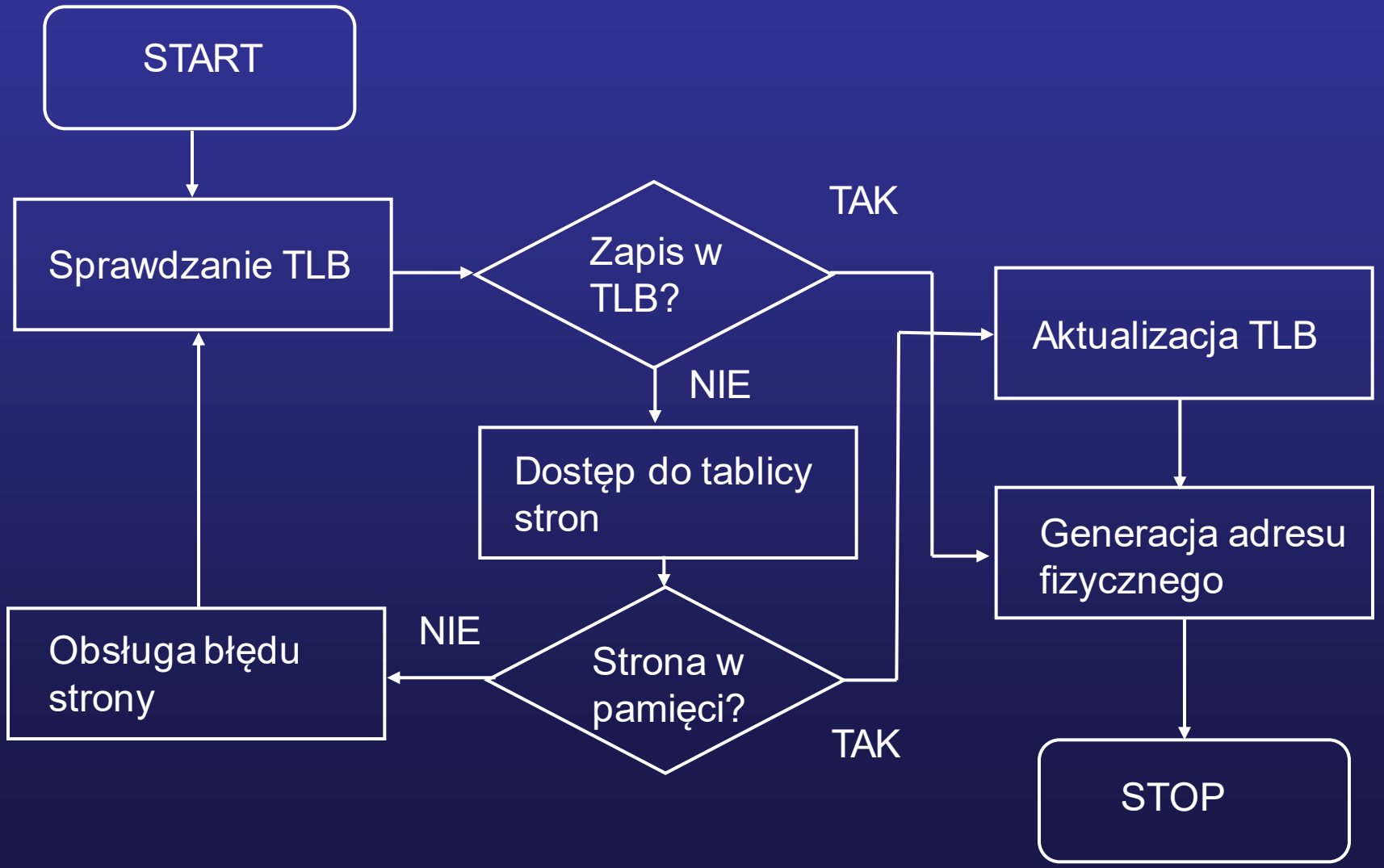
Adresy fizyczne i logiczne w strukturze stron



Translation Lookaside Buffer

- Pamięć podręczna dla ostatnich translacji adresów w pamięci
- Bardzo szybka i mała
- TLB współpracuje z „prawdziwą” pamięcią podręczną
- Niewłaściwy projekt prowadzi do problemów w systemie (patrz procesory Phenom)!

Bufor translacji adresów (TLB)



Segmentacja

- Prostota zarządzania strukturami danych o zmiennej wielkości
- Możliwość indywidualnego przypisywania praw procesom do poszczególnym segmentom
- Możliwość użytkowania segmentu przez wiele procesów
- Adresowanie: numer segmentu + adres względny

Przestrzenie adresowe Pentium

- Niesegmentowana pamięć niestronicowana (adres wirtualny = adres fizyczny) - sterowniki
- Niesegmentowana pamięć stronicowana (BSD)
- Segmentowana pamięć niestronicowana
- Segmentowana pamięć stronicowana (UNIX System V)

Segmentacja w Pentium II

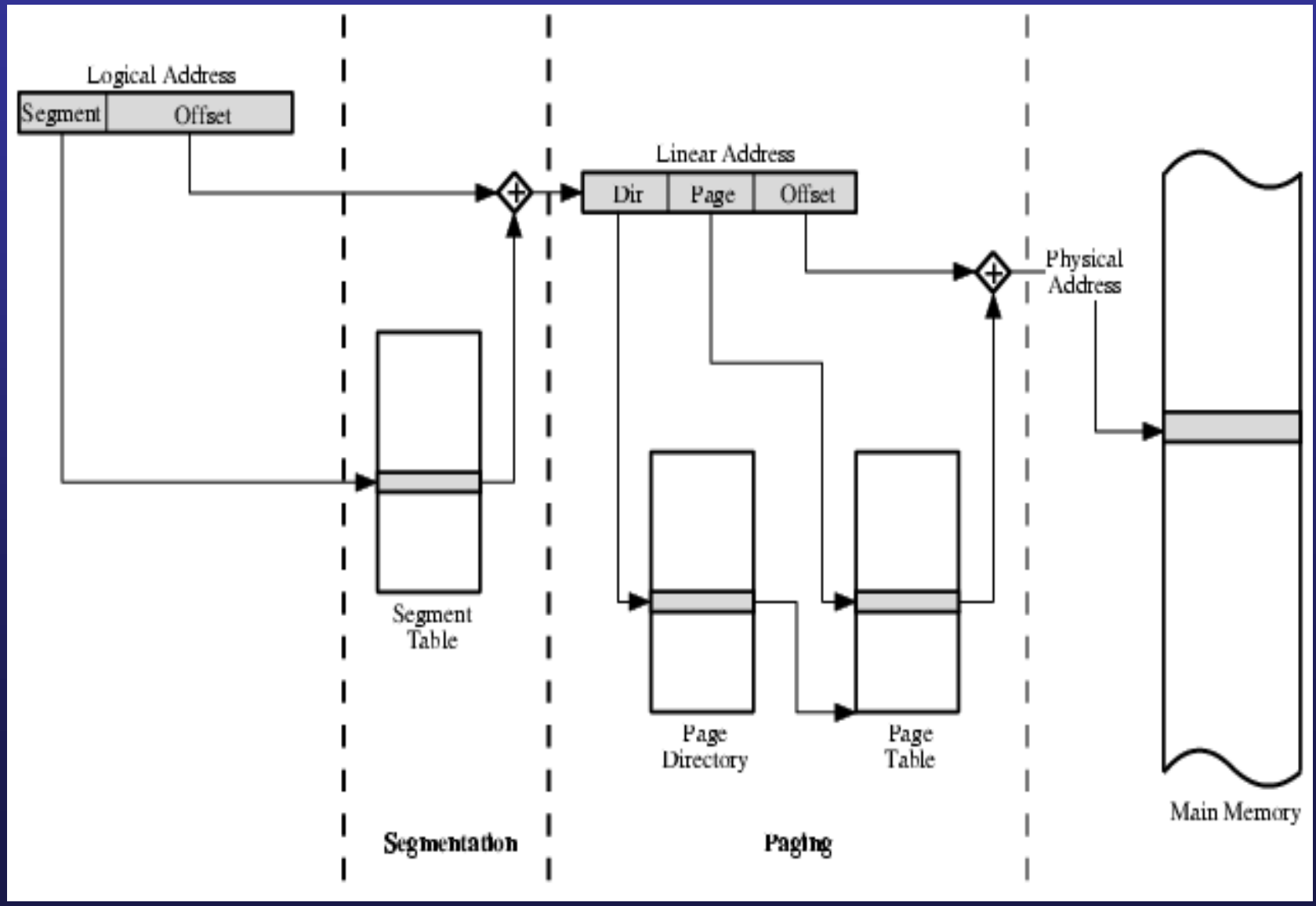
- 2 bity odniesienia do segmentu przeznaczone na mechanizm ochrony
- Adres względny pozwala adresować 2^{32} = 4 GB pamięci
- Całkowita ilość adresowanej pamięci wynosi 2^{16} x 2^{32} = 64 TB
- Dostęp procesu do segmentu zależny jest od poziomu dostępności (oznaczany cyfrą od 0 – najwyższy do 3 - najniższy)

Selektor i deskryptor segmentu

Stronicowanie w Pentium II

- Istnieje katalog stron o długości 1024 wpisów, dzielący pamięć na tyleż grup stron
- Każda grupa stron zawiera do 1024 wpisów
- Każda strona ma rozmiar 4 kB lub 4MB (w zależności od wartości bitu PSE)
- Tablice stron mogą być przechowywane w pamięci wirtualnej
- Bufor translacji adresów zawiera do 32 zapisów tablic stron dla określonego katalogu stron

Stronicowanie/segmentacja w Pentium II



Tryby adresowania Pentium

- Pentium posiada sześć rejestrów segmentowych (SR) do przechowywania adresów początkowych segmentów
- Z rejestrem segmentowym związany jest rejestr deskryptora segmentu (zawiera prawa dostępu do segmentu i jego długość)
- Dodatkowe rejestry (podstawowy i indeksowy) pomagają budować adres

Tryby adresowania Pentium c.d.

- Natychmiastowy (Arg = A)
- Rejestrowy (AL = R)
- Z przesunięciem (AL = (SR) + A)
- Z rejestrem podstawowym (AL = (SR) + (B))
- Z rejestrem podstawowym z przesunięciem (AL = (SR) + (B) + A)
- Skalowane indeksowanie z przesunięciem (AL = (SR) + (I) x S + A)
- Z rejestrem podstawowym z indeksem i przesunięciem (AL = (SR) + (B) + (I) + A)
- Z rejestrem podstawowym ze skalowanym indeksem i przesunięciem (AL = (SR) + (I) x S + (B) + A)
- Względny (AL = (PC) + A)

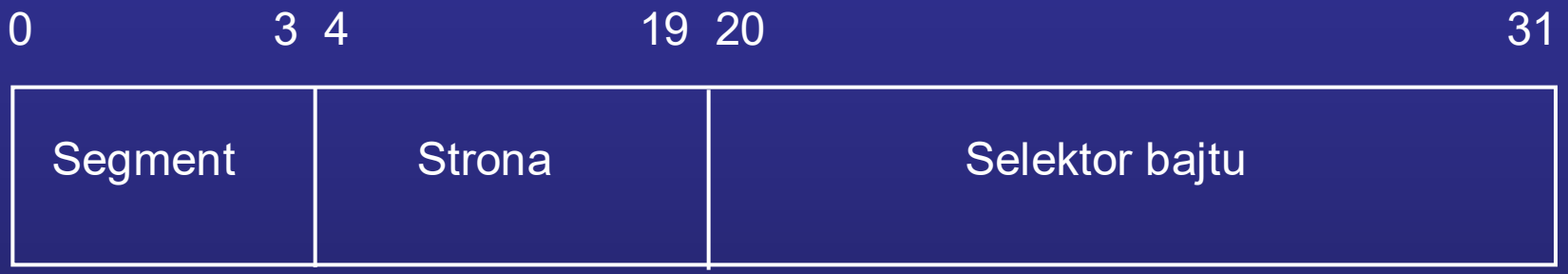
Ilustracja trybów adresowania Pentium

Zarządzanie pamięcią w PowerPC

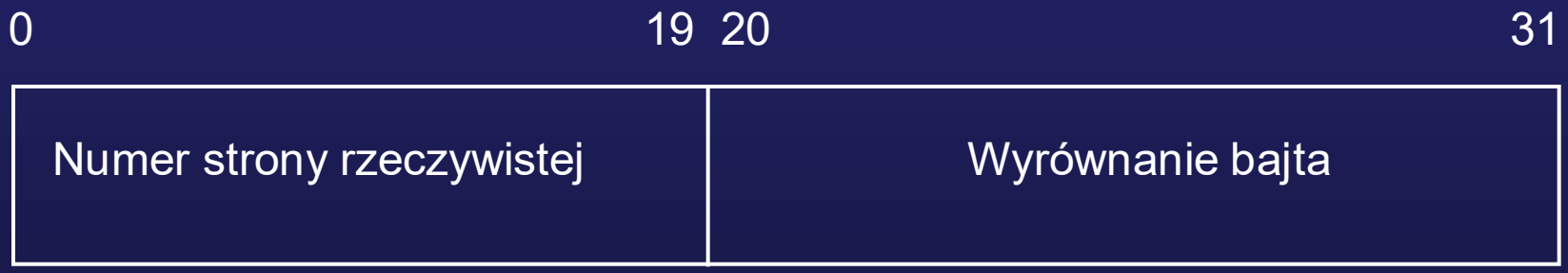
- Implementacja zależna jest od architektury (32- lub 64-bitowej – prostsza lub bardziej złożona segmentacja)
- Mechanizm translacji adresu bloku stosowany jest jako alternatywa dla mechanizmu stronicowania – pozwala ominąć stronicowanie dla dużych bloków pamięci
- Posiada 16 rejestrów segmentowych

Adres efektywny i rzeczywisty PowerPC

- Adres efektywny



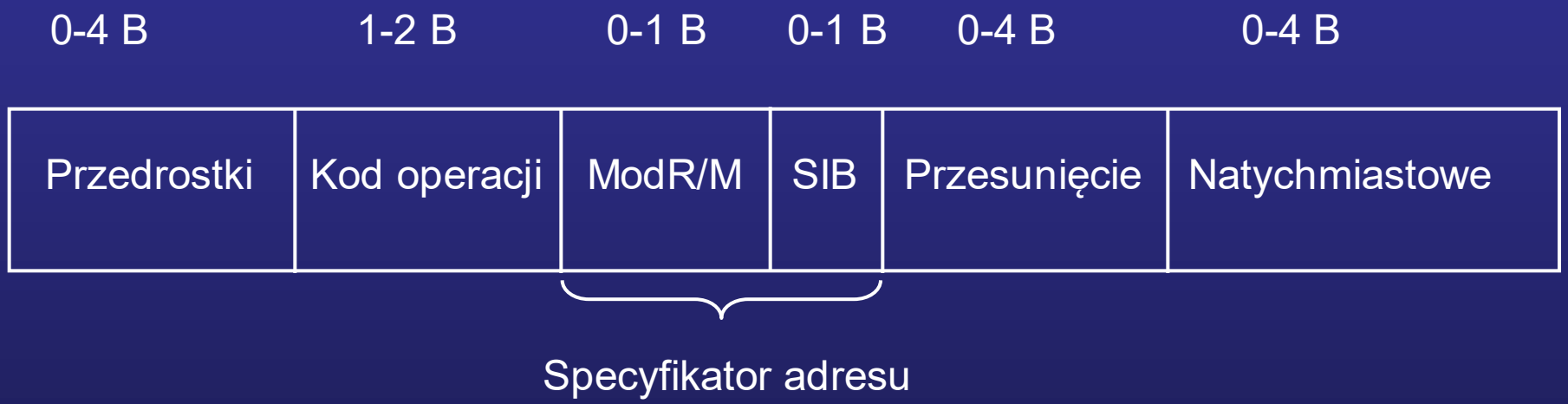
- Adres rzeczywisty



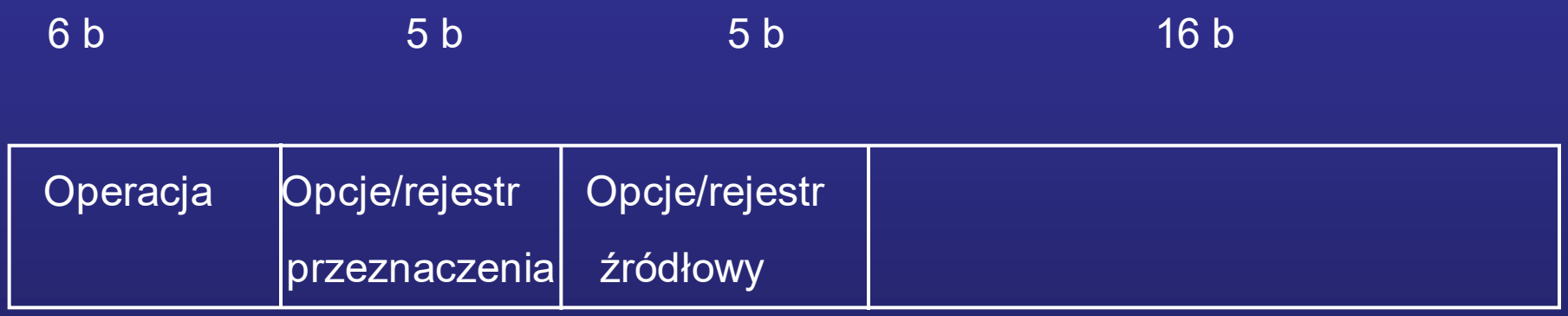
Tryby adresowania PowerPC

- Pośredni (AE = (BR) + D)
- Pośredni indeksowany (AE = (BR) +(IR))
- Adresowanie rozgałęzień
  - Bezwzględny (AE = I)
  - Względny (AE = (PC) +I)
  - Pośredni (AE = (LR/CR))
- Rejestrowy stałopozycyjny (AE = GPR)
- Natychmiastowy (Arg = I)
- Rejestrowy zmiennopozycyjny (AE = FPR)

Format rozkazów Pentium



Format rozkazów PowerPC



Wszystkie rozkazy mają długość 32 bitów!

Organizacja i Architektura Komputerów

Wykład nr 6: Struktura procesora

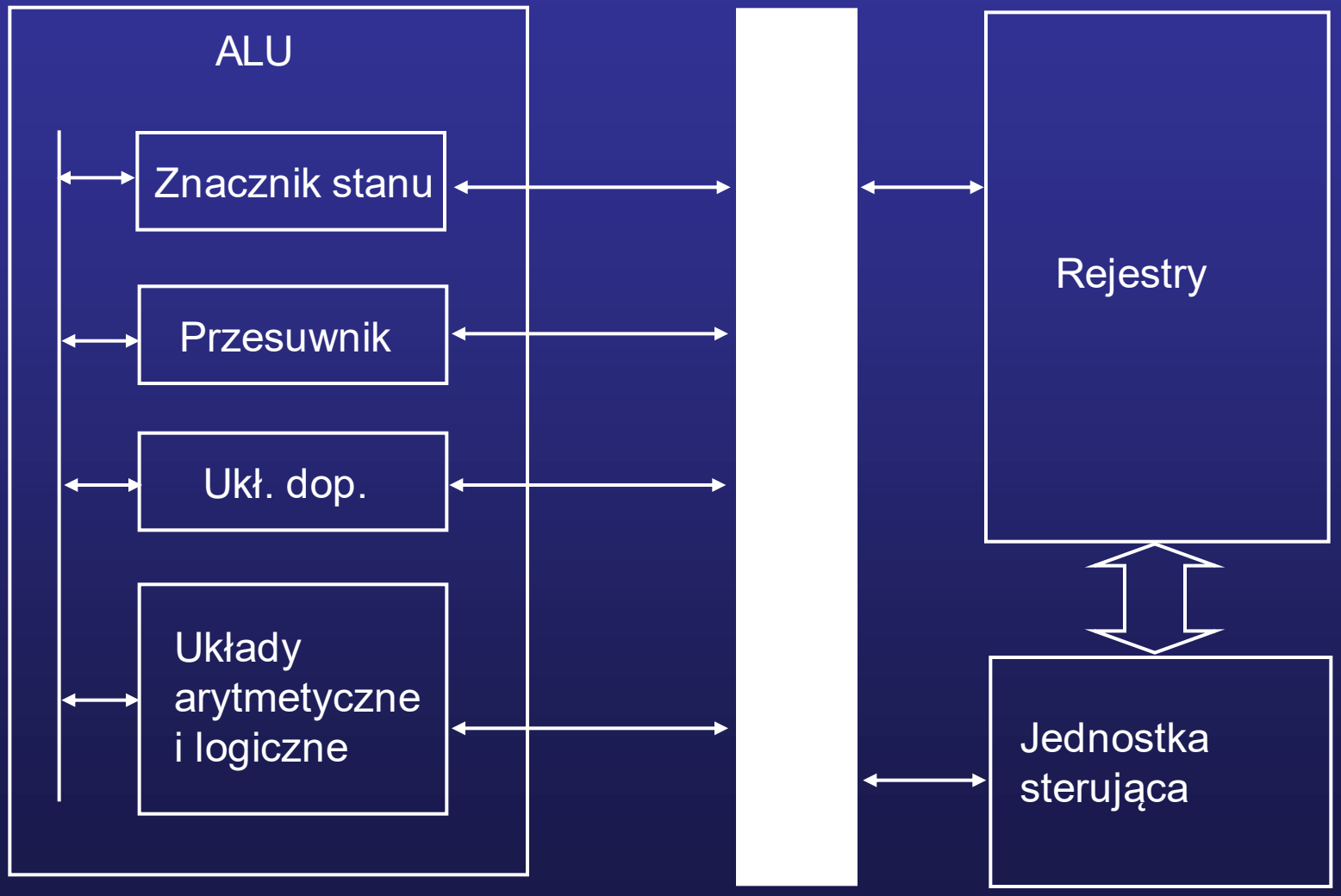
Piotr Bilski

Zadania procesora:

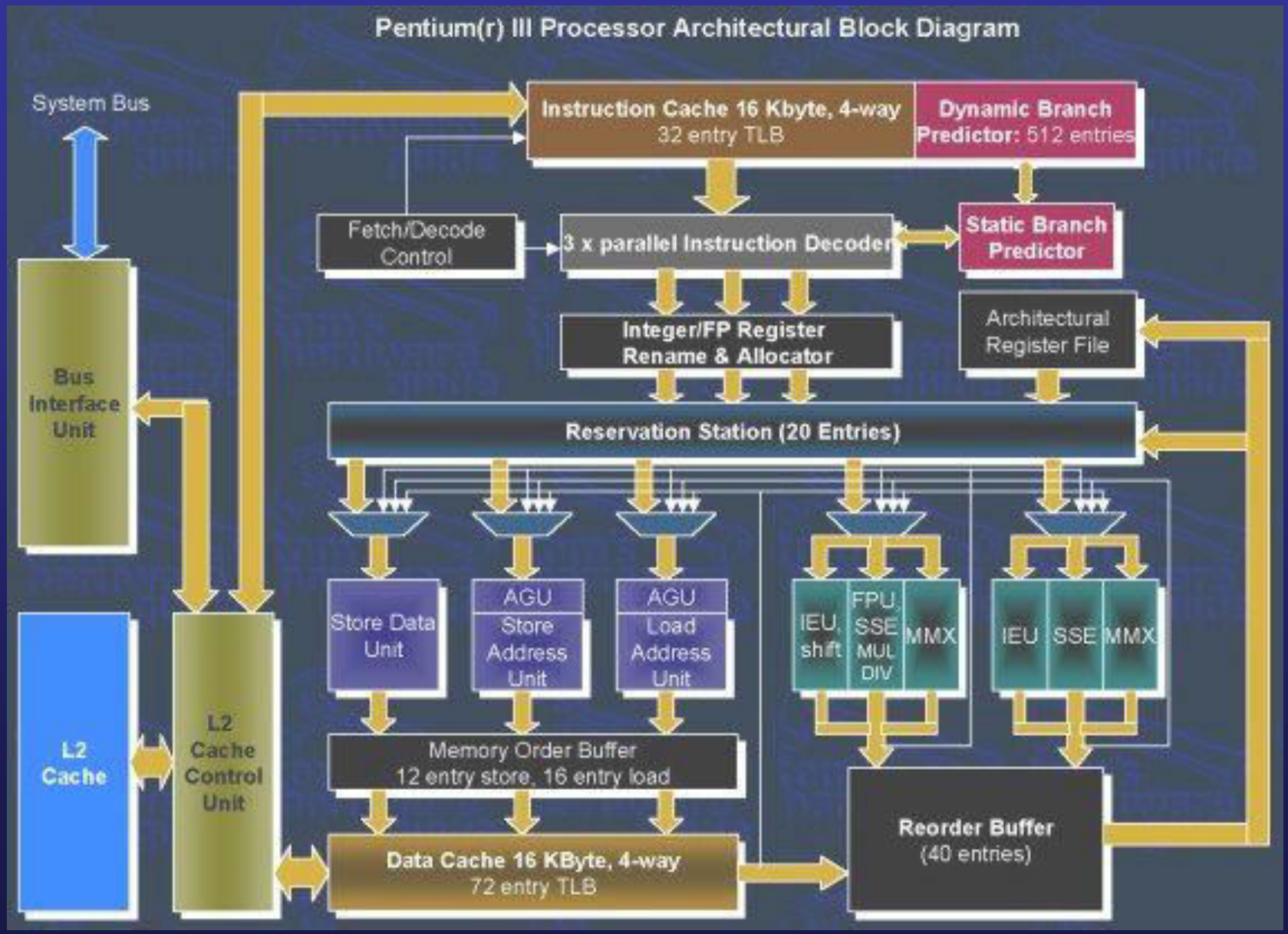
- Pobieranie rozkazów
- Interpretacja rozkazów
- Pobieranie danych
- Przetwarzanie danych
- Zapis danych

Stąd wymaganie posiadania rejestrów

Wewnętrzna struktura procesora

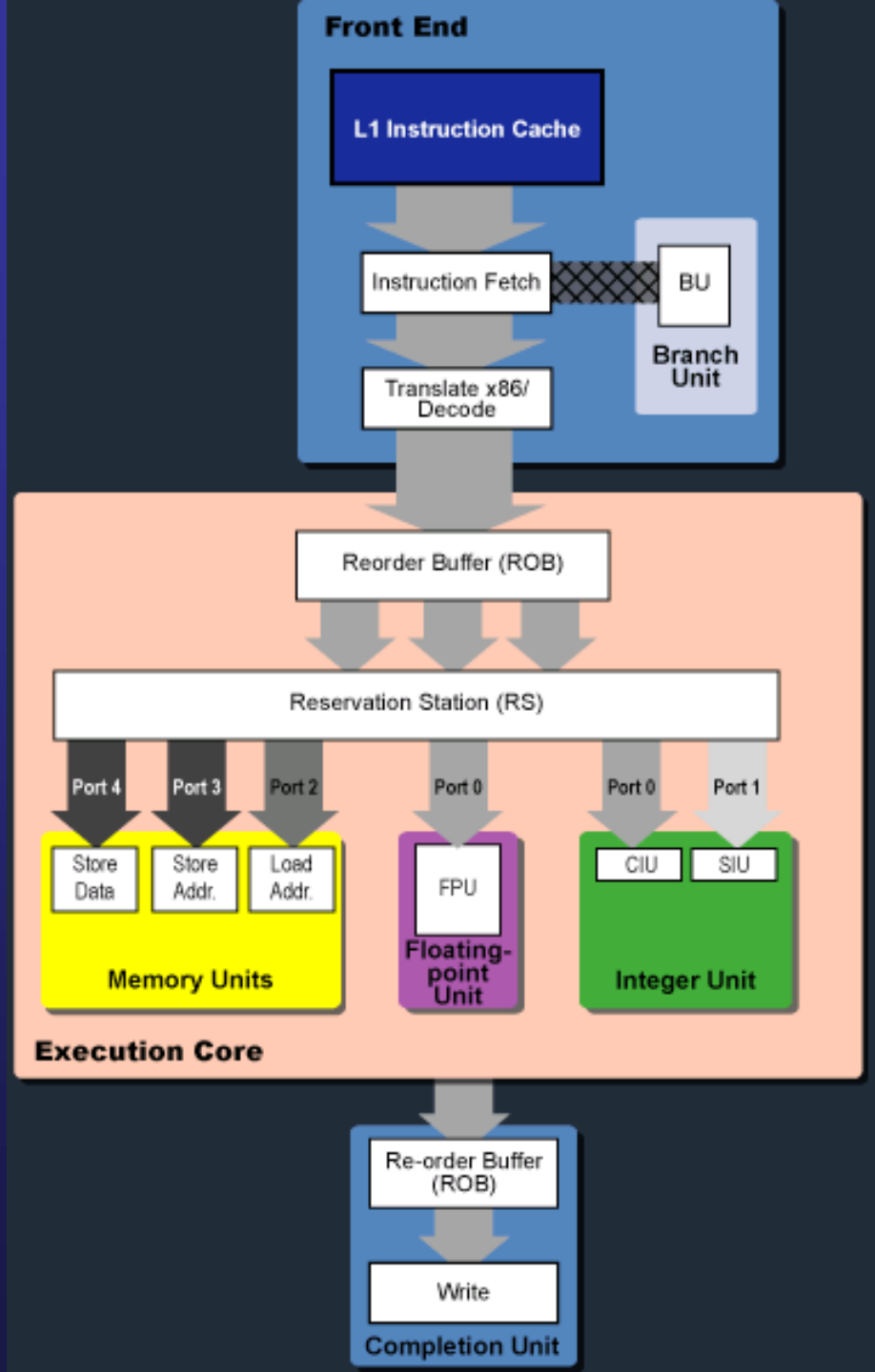


Schemat blokowy procesora Pentium 3



Schemat blokowy rdzenia P6 (Pentium Pro) – 1995 r.

- „przód” procesora
- rdzeń
- układ kończenia rozkazu



Rodzaje rejestrów

- Widzialne dla użytkownika (adresowe, danych itp.)
- Niewidzialne dla użytkownika (sterowania, stanu)

- Rozgraniczenie takie jest umowne (patrz licznik rozkazów)!

Rejestry widzialne dla użytkownika

- Ogólnego przeznaczenia – robocze (GPR – general purpose registers)
- Danych
- Adresów (wskaźnik segmentu, stosu, rejestr indeksowy)
- Kodów warunkowych (znaczniki stanu, flagi) – tylko do odczytu!

Rejestry sterowania i stanu

- Podstawowe:
  - Licznik programu (PC)
  - Rejestr rozkazu (IR)
  - Rejestr adresowy pamięci (MAR)
  - Rejestr buforowy pamięci (MBR)
- Słowo stanu programu (PSW)
- Rejestr wektora przerwań
- Wskaźnik tablicy stron

Słowo stanu programu

S – bit znaku

Z – bit dla wyniku operacji równego zero

P – bit przeniesienia

R – bit wyniku porównania logicznego

O – bit przepełnienia

I – zezwolenie na wykonanie przerwania

N – tryb nadzorcy

Organizacja rejestrów w procesorze Motorola MC68000

- Rejestry danych i adresowe 32-bitowe
- Specjalizacja: 8 rejestrów danych (D0-D7) i 9 adresowych (dwa używane wymiennie w trybie użytkownika i nadzorcy)
- Magistrala sterująca 24-bitowa, magistrala danych 16-bitowa
- Wskaźnik stosu (SP) jest rejestrem A7
- Rejestr stanu (SR)16-bitowy (inaczej CCR)
- Licznik rozkazów (PC) 32-bitowy
- Rozkazy przechowywane pod adresami parzystymi

Organizacja rejestrów w procesorze Intel 8086

- Rejestry danych i adresowe 16-bitowe
- Rejestry danych/ogólnego przeznaczenia (AX, BX, CX, DX)
- Rejestry wskaźników i indeksowe (SP, BP, SI, DI)
- Rejestry segmentowe (CS, DS, SS, ES)
- Wskaźnik rozkazu
- Wskaźnik stanu

Organizacja rejestrów w procesorze Intel 8086 (c.d.)

AX	Akumulator	SP	Wskaźnik stosu
BX	Bazowy	BP	Wskaźnik bazy
CX	Zliczający	SI	Indeks źródła
DX	Danych	DI	Indeks m. przes

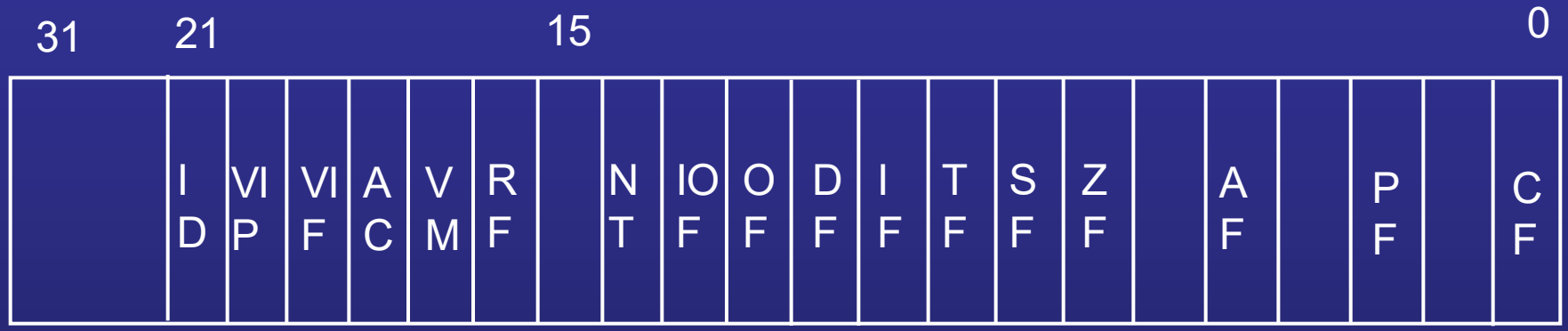
Organizacja rejestrów w procesorze Intel 386 - Pentium

- Rejestry danych i adresowe 32-bitowe
- Osiem rejestrów ogólnego przeznaczenia (EAX, EBX, ECX, EDX, ESP, EBP, ESI, EDI)
- Dla zachowania zgodności wstecznej mniejsza połowa rejestrów pełni rolę rejestrów 16-bitowych
- Rejestr znaczników 32-bitowy
- Wskaźnik rozkazu 32-bitowy

Rejestry zmiennopozycyjne procesora Pentium

- Osiem rejestrów numerycznych (80-bitowych)
- 16-bitowy rejestr sterowania
- 16-bitowy rejestr stanu
- 16-bitowe słowo wyróżników
- 48-bitowy wskaźnik rozkazu
- 48-bitowy wskaźnik danych

Rejestr EFLAGS



- TF – znacznik pułapki
- IF – znacznik zezwolenia przerwania
- DF – znacznik kierunku
- IOPL – znacznik uprzywilejowania wejścia/wyjścia
- RF – znacznik wznowienia
- AC – kontrola wyrównania
- ID – znacznik identyfikacji

Organizacja rejestrów w procesorze Athlon 64

- Zgodność z architekturą x86-64 (40-bitowa fizyczna przestrzeń adresowa, 48-bitowa wirtualna przestrzeń adresowa)
- Rejestr danych i adresowe 64-bitowe
- 8 rejestrów ogólnego przeznaczenia (RAX, RBX, RCX, RDX, RBP, RSI, RDI, RSP), działają w trybie zgodności 32-bitowej
- Opteron posiada dodatkowych 8 rejestrów ogólnego przeznaczenia (R8-R15)
- 16 rejestrów SSE (XMM0-XMM15)
- 8 rejestrów zmiennoprzecinkowych x87 80-bitowych

Organizacja rejestrów w procesorze PowerPC

- 32 rejestry ogólnego przeznaczenia (64-bitowe) + rejestr wyjątku (XER)
- 32 rejestry dla jednostki zmiennopozycyjnej (64-bitowe) + rejestr stanu i sterowania (FPSCR)
- Rejestry jednostki przetwarzania rozgałęzień: 32-bitowy rejestr warunku, 64-bitowe rejestry powiązania i zliczania

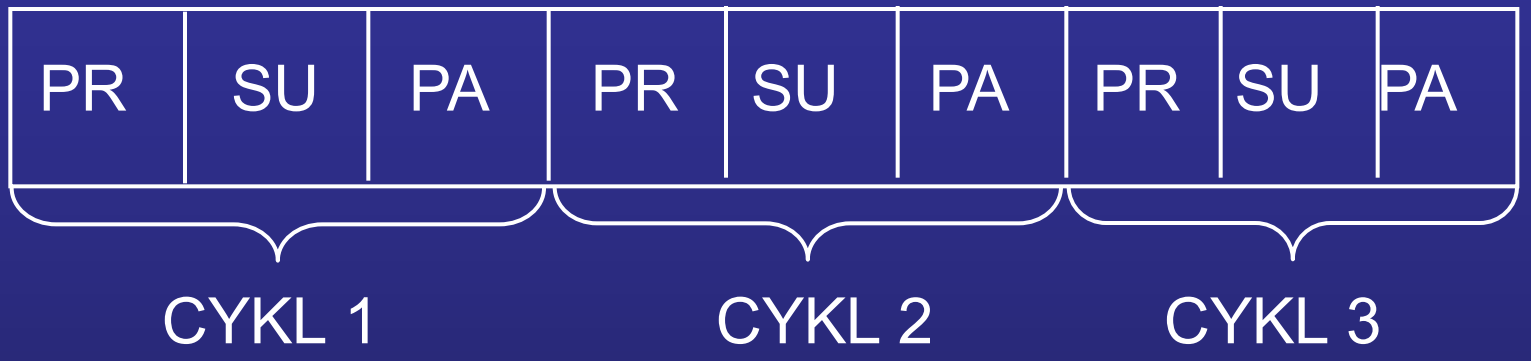
Cykl rozkazu

Potok

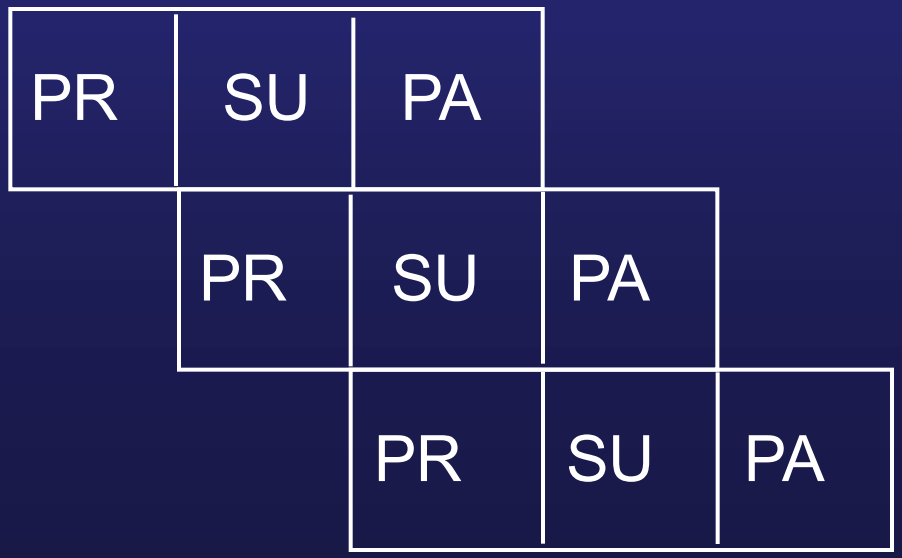
- Problem: przy wykonywaniu cyklu rozkazowego przetwarzany jest tylko jeden rozkaz
- Rozwiązanie: podział cyklu na mniejsze fragmenty
- Warunek: konieczne są momenty, gdy nie ma odwołania do pamięci głównej!

Przykład potoku - pralnia

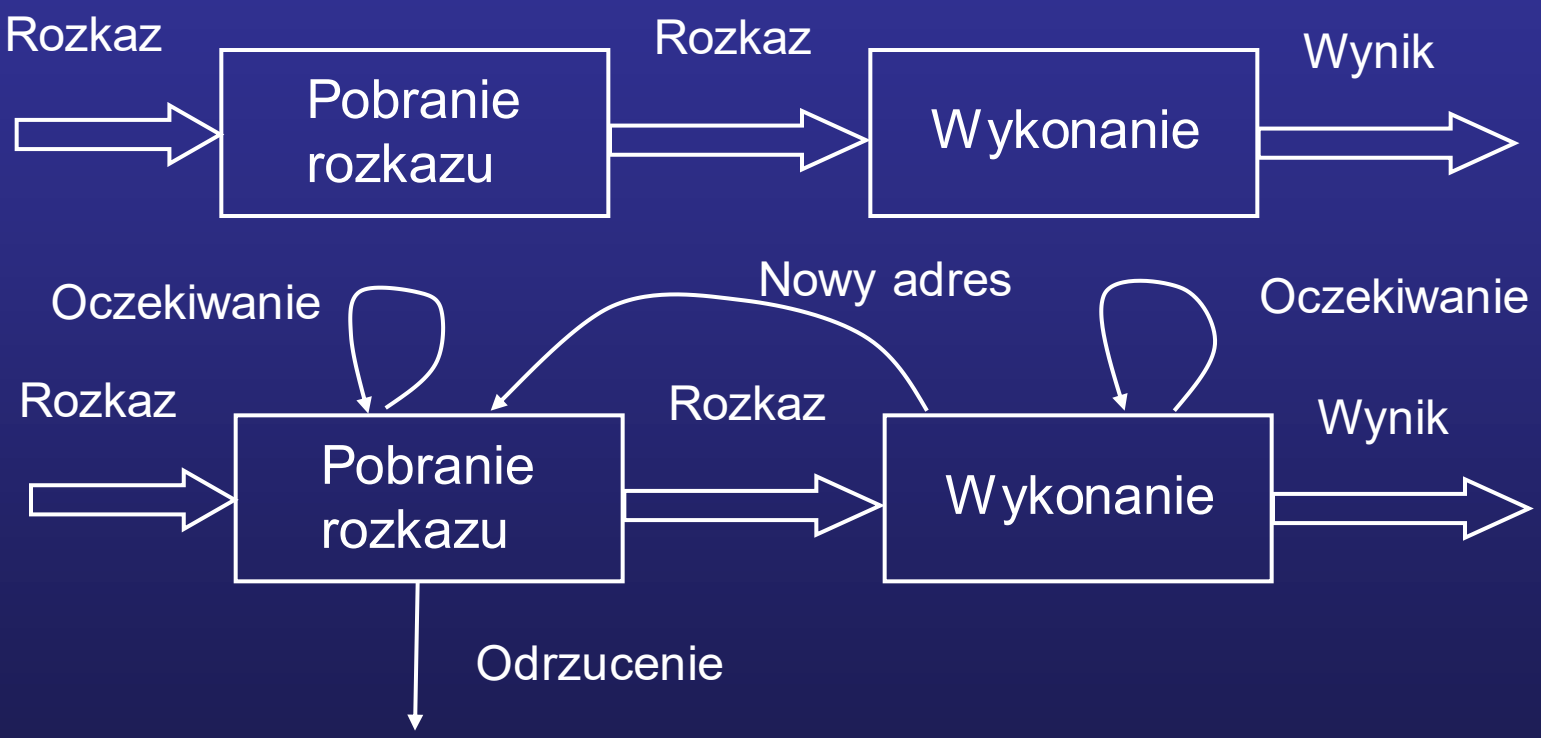
3 godziny / cykl – 9 godzin całość



3 godziny / cykl – 5 godzin całość !!



Pobieranie z wyprzedzeniem

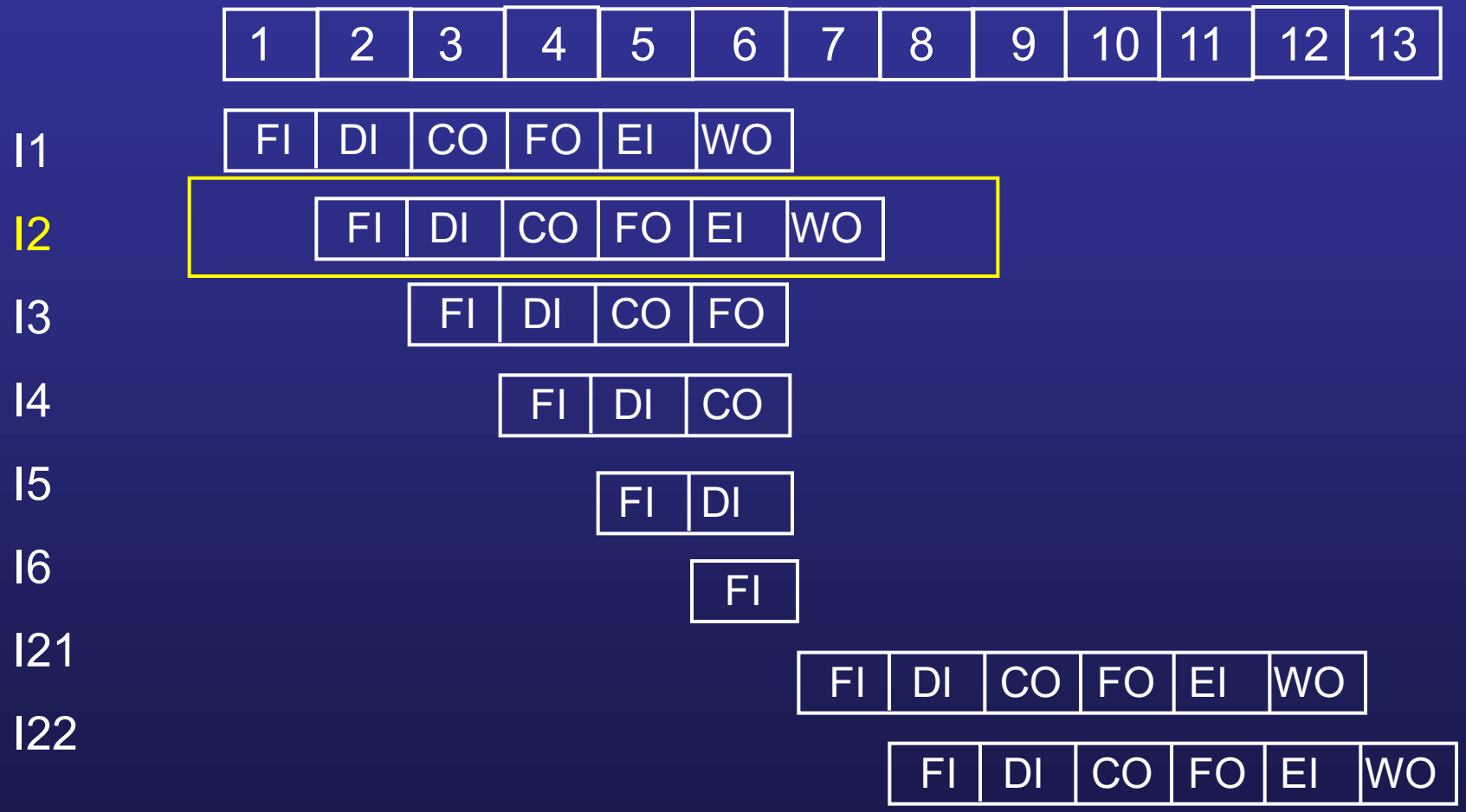


- UWAGA: przyspieszenie nie jest dwukrotne, bo dostęp do pamięci trwa dłużej, niż wykonanie rozkazu

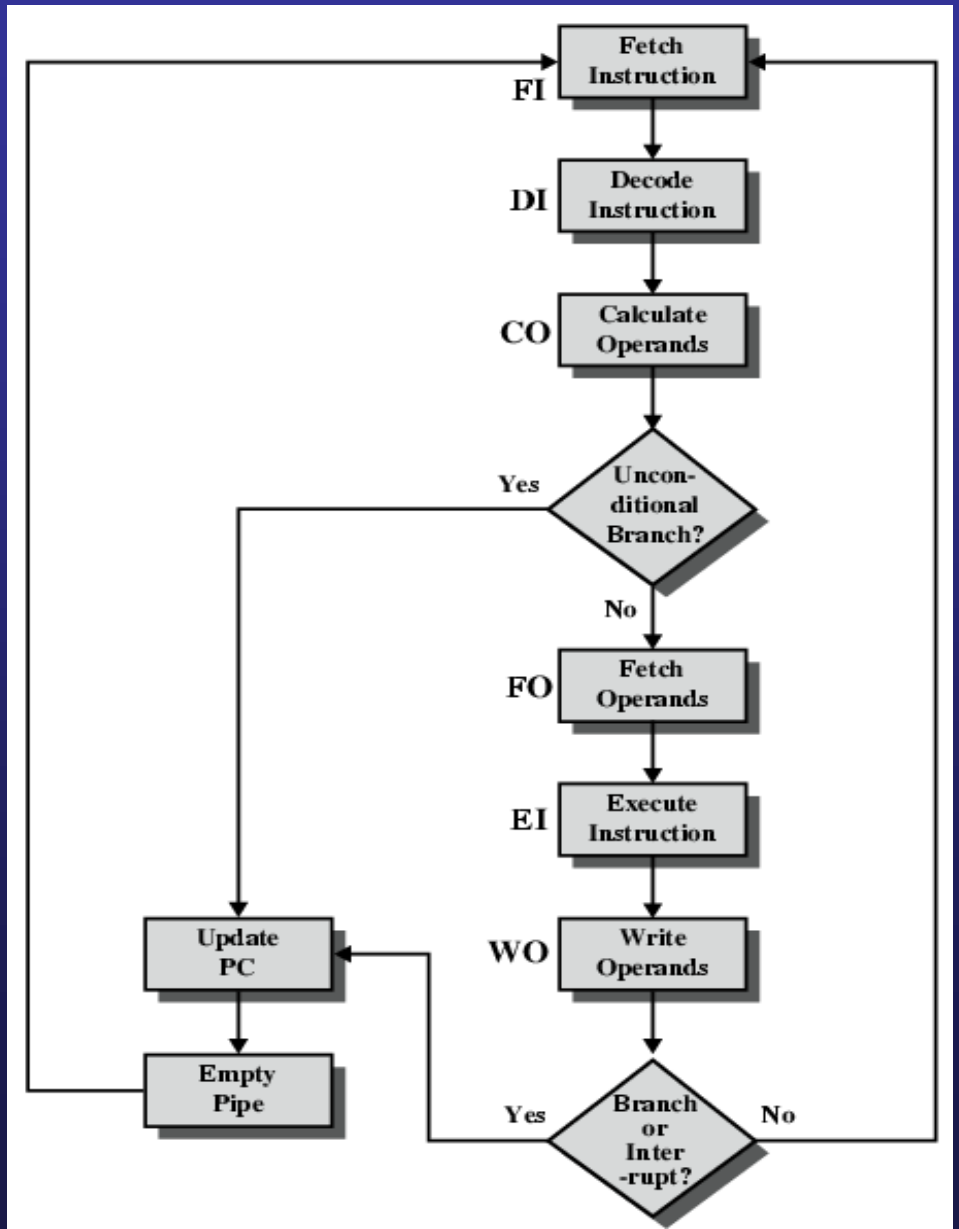
Podstawowy podział cyklu rozkazu:

- Pobranie rozkazu (FI)
- Dekodowanie rozkazu (DI)
- Obliczanie argumentów (CO)
- Pobranie argumentów (FO)
- Wykonanie rozkazu (EI)
- Zapisanie argumentu (WO)

Rozgałęzienie a potoki



Algorytm realizacji potoku



Problemy z przetwarzaniem potokowym

- Różne etapy potoku nie zajmują tej samej ilości czasu
- Przenoszenie danych między buforami może znacząco wydłużyć trwanie potoku
- Zależność rejestrów i pamięci od optymalizacji potoku może być minimalizowana dużym nakładem kosztów

Wydajność przetwarzania potokowego

Czas trwania cyklu:

$$\tau = \max[\tau_i] + d = \tau_m + d, \quad 1 \le i \le k$$

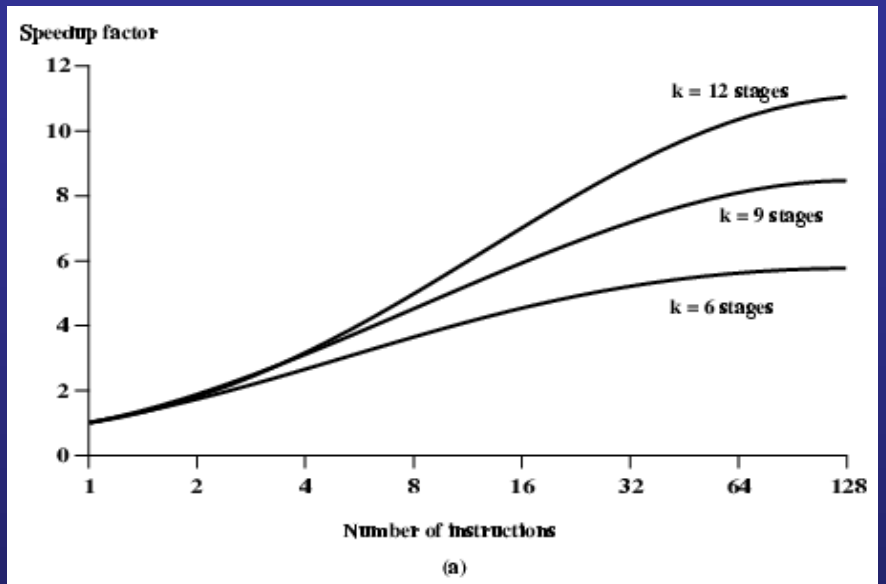
Czas potrzebny na wykonanie wszystkich rozkazów:

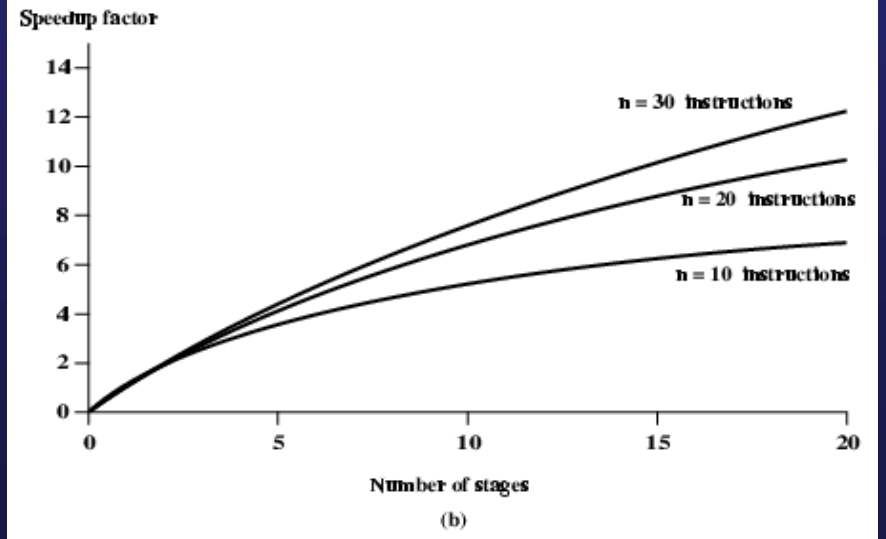
$$T_k = [k + (n-1)] \cdot \tau$$

Współczynnik przyspieszenia potoku rozkazów:

$$S_k = \frac{T_1}{T_k} = \frac{nk}{k + (n-1)}$$

Ilustracja wydajności przetwarzania potokowego





Potoki współczesnych procesorów

- Pentium 3 – 10 etapów
- Athlon – 10 etapów dla ALU, 15 etapów dla FPU
- Pentium M – 12 etapów
- Athlon 64/ 64 X2 – 12 etapów dla ALU, 17 etapów dla FPU
- Pentium 4 Northwood – 20 etapów (hiperpotok!!)
- Pentium 4 Prescott – 31 etapów
- Core2Duo – 14 etapów

Obsługa rozgałęzień

- Zwielokrotnienie strumienia
- Pobieranie docelowego rozkazu z wyprzedzeniem (prefetch)
- Bufor pętli
- Przewidywanie rozgałęzienia
- Opóźnione rozgałęzienie

Zwielokrotnione strumienie

- Oba rozkazy, które mogą być przetwarzane w wyniku wykonania rozgałęzienia są ładowane do dwóch strumieni
- Problemem jest dostęp do pamięci dla obu instrukcji

Pobieranie docelowego rozkazu z wyprzedzeniem

- Gdy rozpoznana zostanie instrukcja rozgałęzienia, następuje pobranie rozkazu, który jest rozkazem docelowym. Jest on przechowywany do czasu wykonania rozgałęzienia

Bufor pętli

- Tworzony jest bufor w pamięci do przechowywania następujących po sobie rozkazów
- Jest on szczególnie przydatny w instrukcjach rozgałęzień warunkowych i pętli

Przewidywanie rozgałęzień warunkowych

- Statyczne
  - Przewidywanie nigdy nie następującego rozgałęzienia (Sun SPARC, MIPS)
  - Przewidywanie zawsze następującego rozgałęzienia
  - Przewidywanie na podstawie kodu operacji
- Dynamiczne
  - Przełącznik nastąpiło/nie nastąpiło
  - Tablica historii rozgałęzień

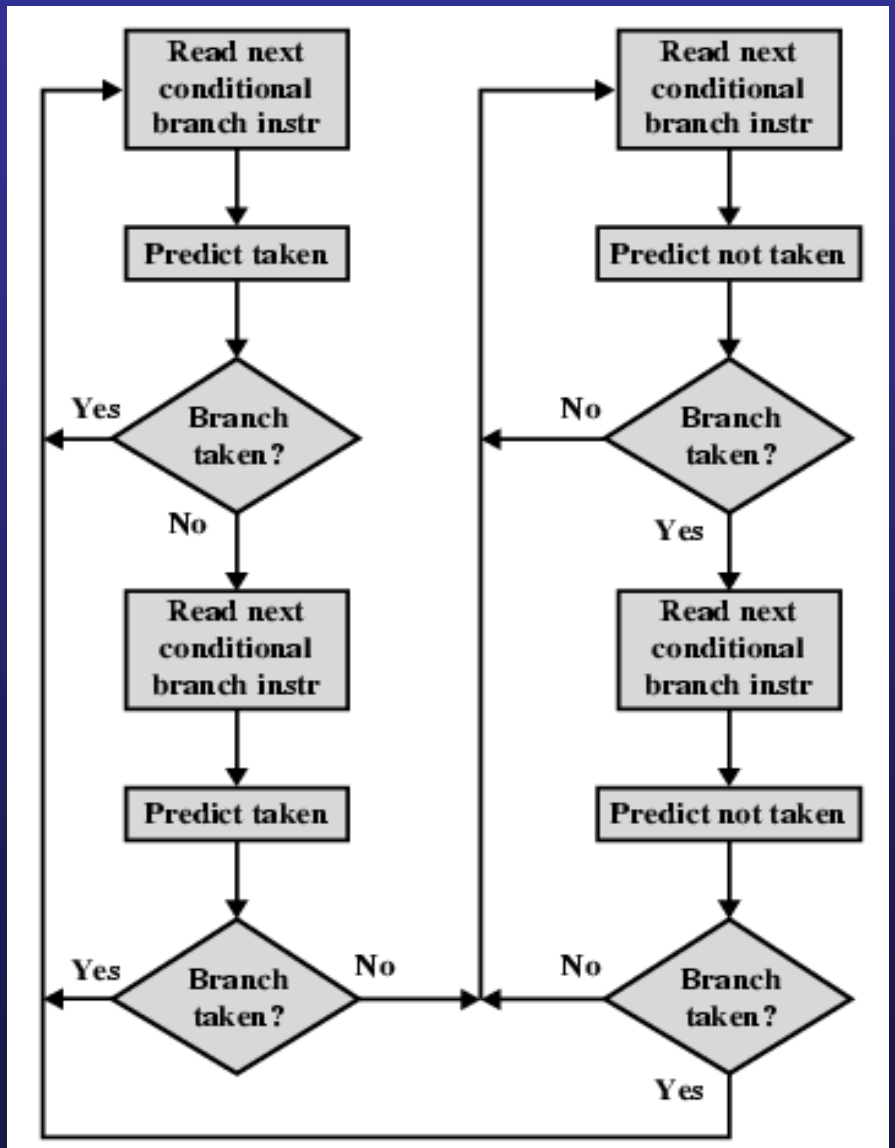
Przewidywanie statyczne

- Najprostsze, używane jako metoda ostatniej szansy (fallback method), np. w procesorze Motorola MPC7450
- Pentium 4 umożliwiał wstawienie do kodu sugestii, czy przewidywanie statyczne powinno wskazywać na skok, czy nie (tzw. prediction hint)

Rozwiązania dynamiczne przewidywania rozgałęzień warunkowych

- Przechowuje się historię rozkazów rozgałęzienia warunkowego
- Jest ona reprezentowana przez bity przechowywane w pamięci podręcznej
- Każdy rozkaz otrzymuje własne bity historii
- Drugie rozwiązanie to tablica przechowująca informacje o wyniku rozkazu rozgałęzienia warunkowego

Przewidywanie przy pomocy bitów historii



Tablica historii rozgałęzień

Adres rozkazu rozgałęzienia	Bity historii	Rozkaz docelowy

Lokalna predykcja rozgałęzień

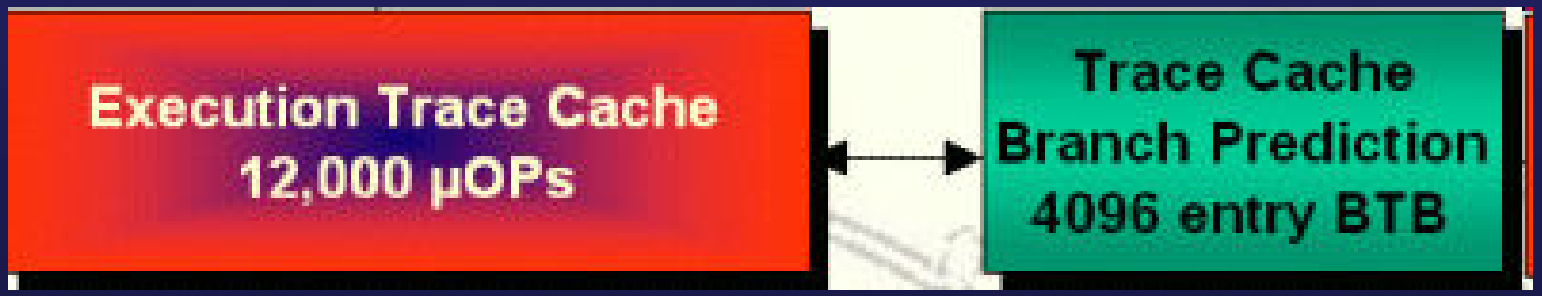
- Wymaga osobnego bufora historii dla każdej instrukcji, chociaż tablica historii może być wspólna dla poszczególnych typów rozkazów
- Procesory Pentium MMX, Pentium 2 i 3 mają układy lokalnego przewidywania z 4 bitami historii oraz tablicą historii z 16 pozycjami dla każdego typu rozkazu
- Skuteczność lokalnej predykcji zmierzono na poziomie 97 %

Globalna predykcja rozgałęzień

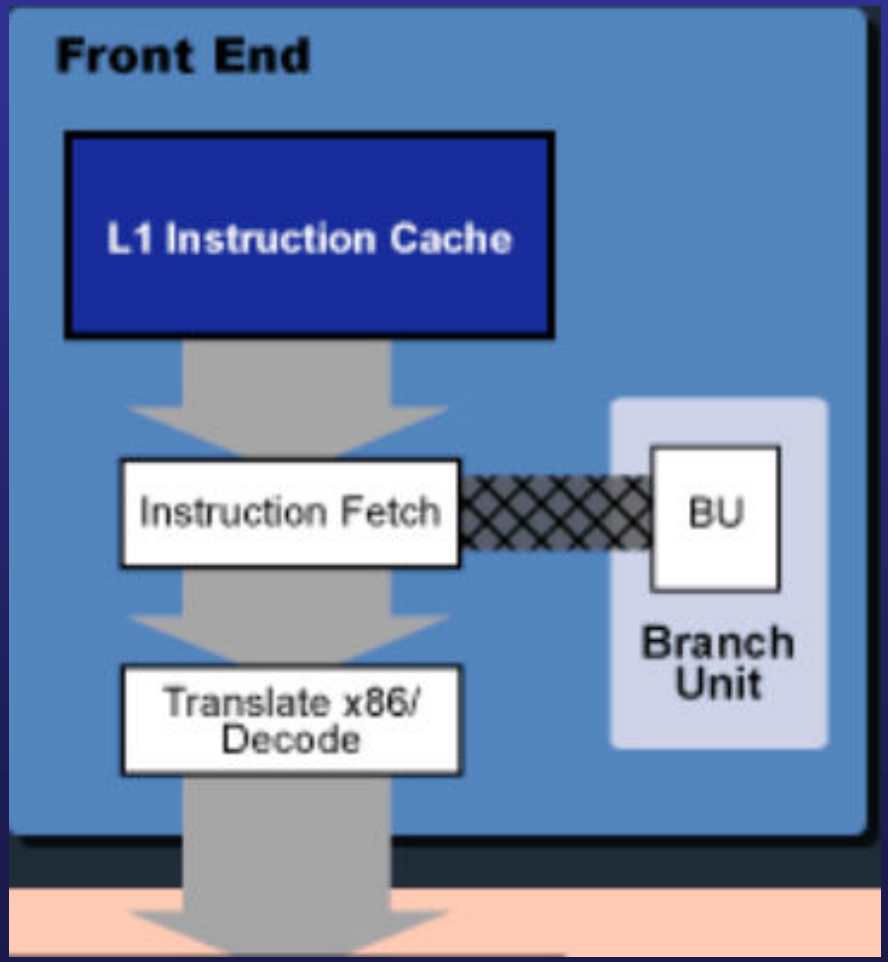
- Przechowywana jest współdzielona historia wszystkich rozkazów rozgałęzień
- Pozwala wziąć pod uwagę zależności między różnymi rozkazami rozgałęzień
- Rozwiązanie rzadko tak dobre, jak predykcja lokalna
- Rozwiązanie hybrydowe: współdzielony układ predykcji globalnej i tablica historii (procesory AMD, Pentium M, Core, Core 2)

Jednostka przewidywania rozgałęzień

- Układ w procesorze odpowiedzialny za przewidywanie wszystkich zaburzeń w sekwencyjnym wykonaniu programu
- Powiązana jest często z pamięcią podręczną mikrooperacji
- W Pentium 4 bufor dla przewidywania rozgałęzień ma 4096 pozycji, w Pentium 3 – tylko 512. Dzięki temu miał mieć o 33 procent lepszy współczynnik trafień



Lokalizacja jednostki przewidywania rozgałęzień



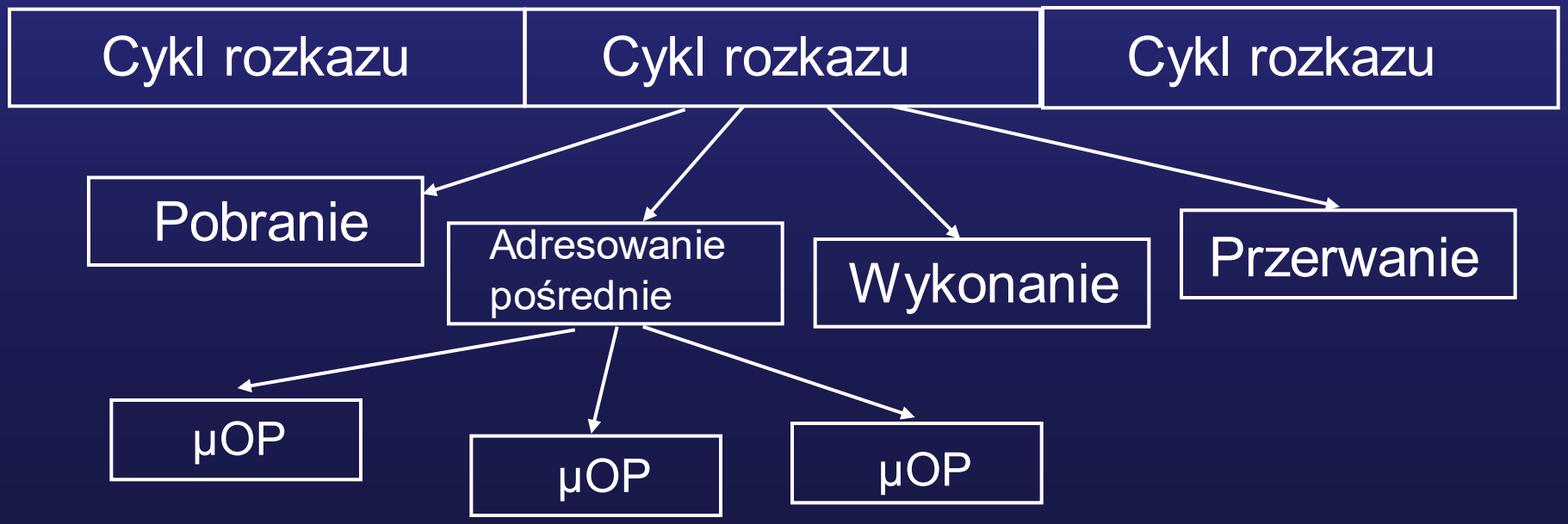
Organizacja i Architektura Komputerów

Wykład nr 7: Układ sterujący. Mikrooperacje

Piotr Bilski

Zadania układu sterującego

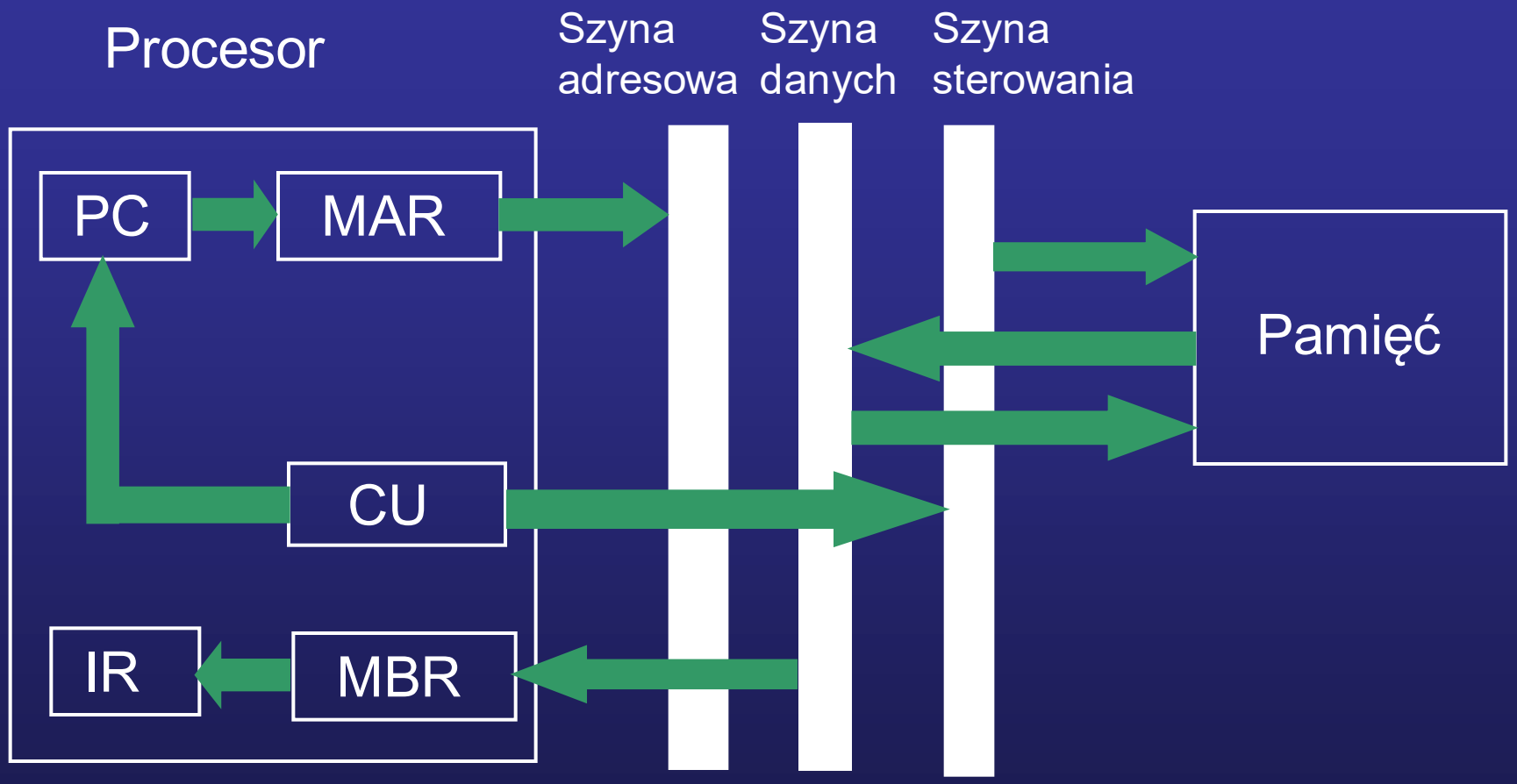
- Monitorowanie stanu procesora
- Sterowanie pracą procesora
- Szeregowanie mikrooperacji



Mikrooperacje

- Elementarne kroki w ramach cyklu rozkazu
- Każdy krok jest prosty, lecz jego efekt jest również niewielki
- Każda mikrooperacja zajmuje jedną jednostkę czasu
- Występują we wszystkich fazach cyklu rozkazowego: pobrania, adresowania pośredniego, wykonywania, przerwania

Cykl pobrania rozkazu



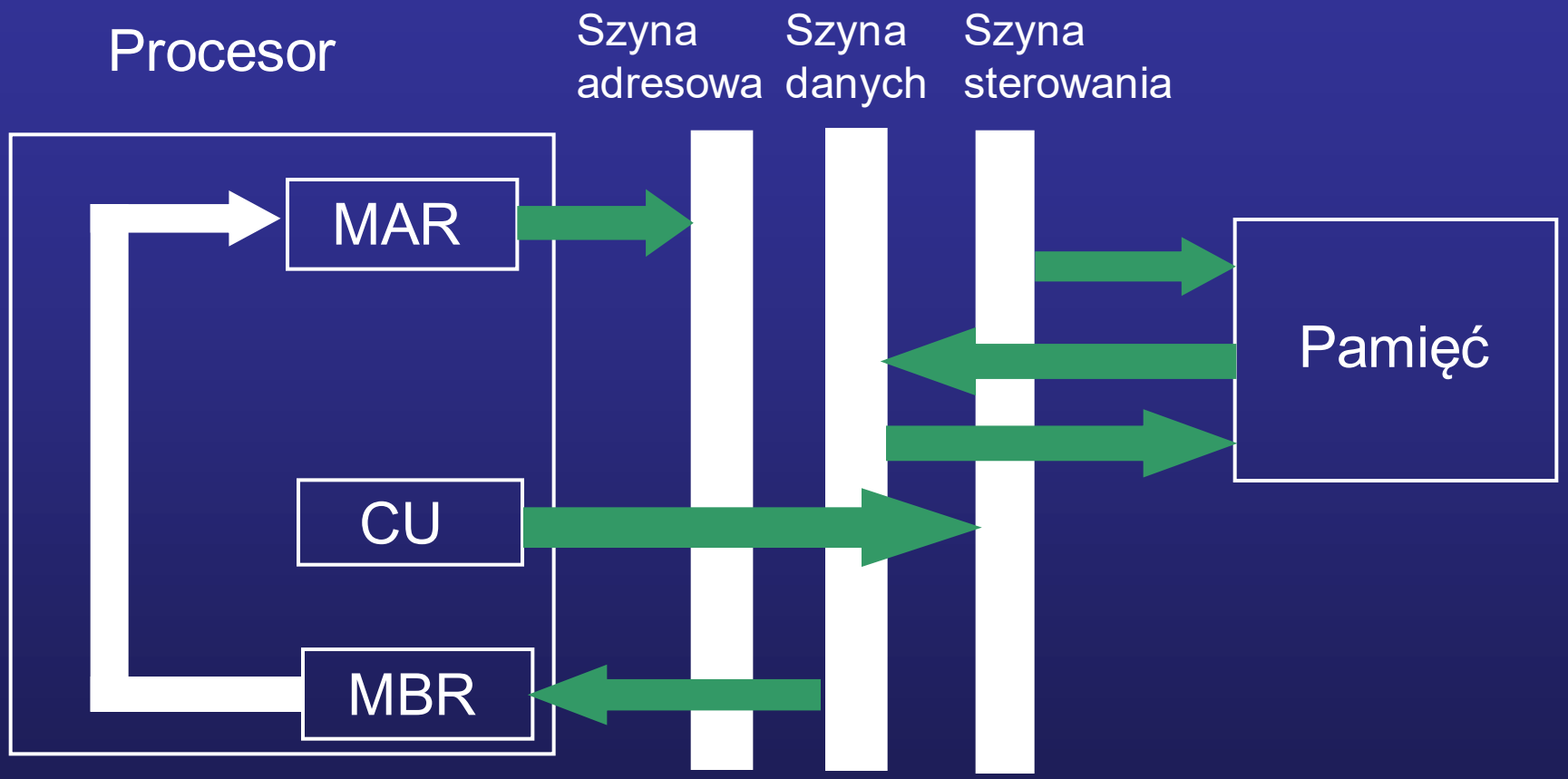
t_1: MAR ← (PC)

t_2: MBR ← M(MAR)

PC ← (PC) + 1

t_3: IR ← (MBR)

Cykl adresowania pośredniego

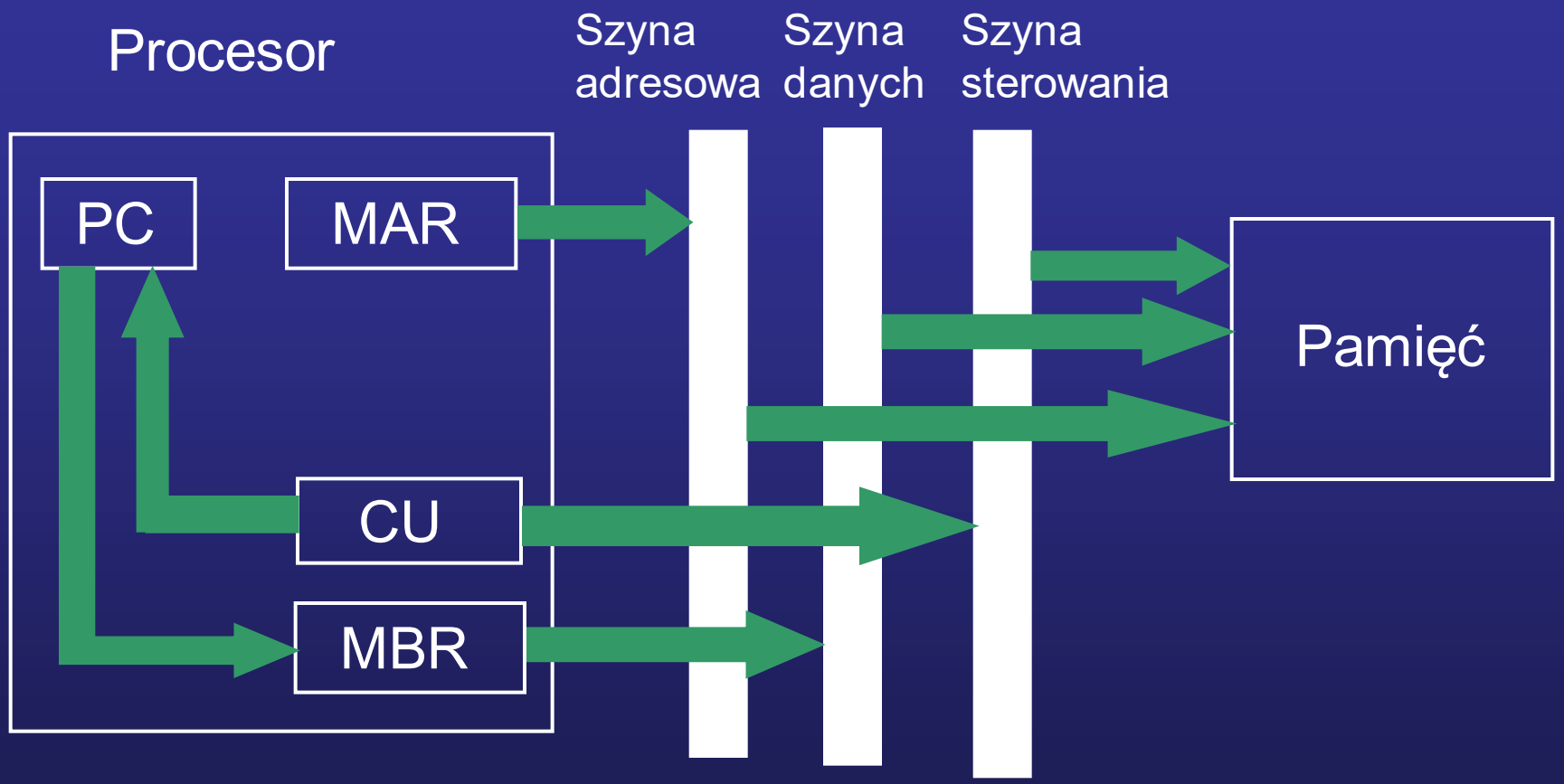


t_1: MAR ← (IR(Adres))

t_2: MBR ← M(MAR)

t_3: IR(Adres) ← (MBR(Adres))

Cykle przerwania



t_1: MBR ← (PC)

t_2: MAR ← Adres

PC ← Adres POP

t_3: M(MAR) ← (MBR)

Cykl wykonania

- Najbardziej skomplikowany cykl ze względu na rozgałęzienia
- Przykład:

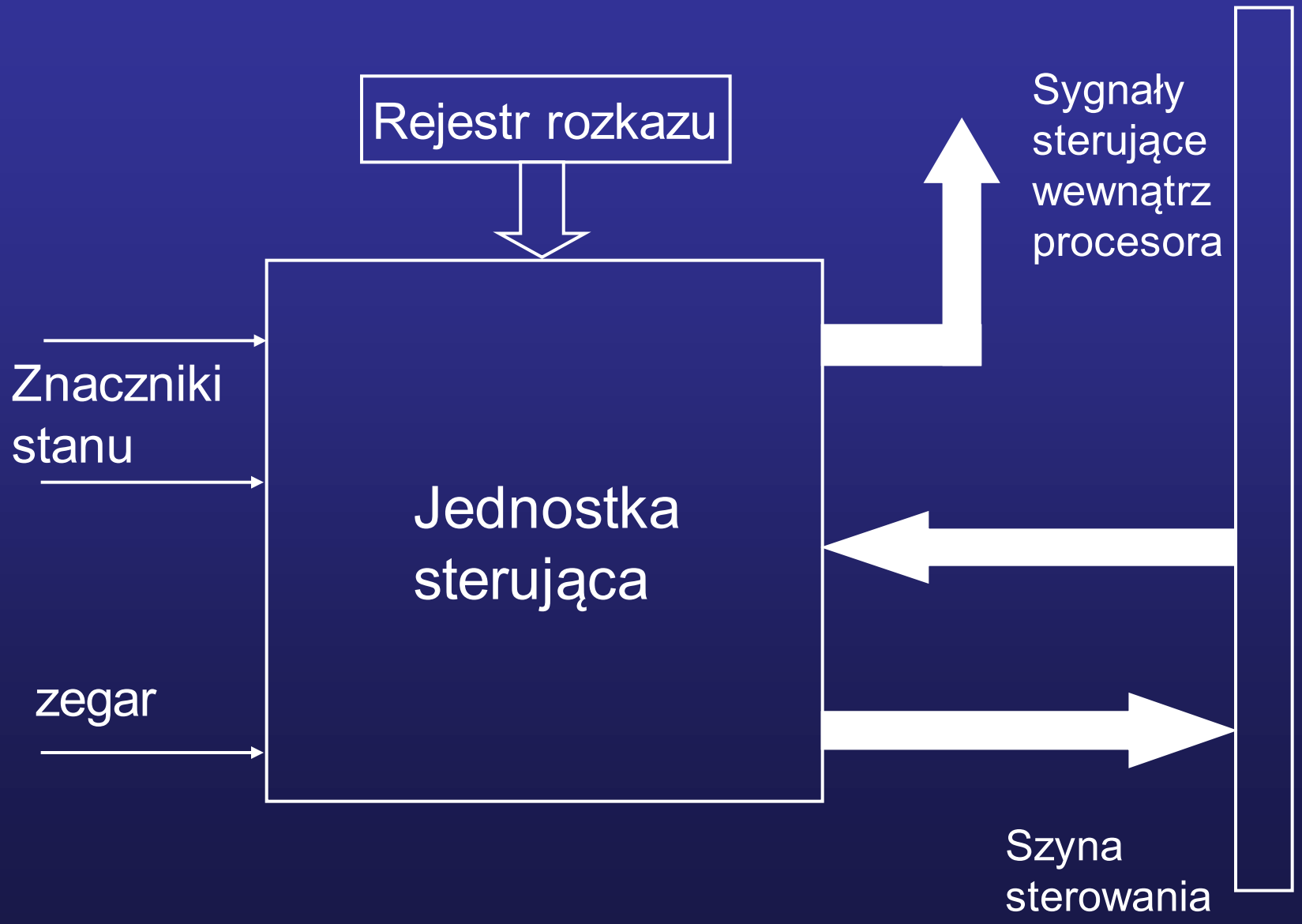
BSA X

t_1: MAR $\leftarrow$ (IR(Adres))
MBR $\leftarrow$ (PC)
t_2: PC $\leftarrow$ (IR(Adres))
M() $\leftarrow$ (MBR)
t_3: PC $\leftarrow$ (PC) + 1

Rejestr cyklu rozkazu (ICC)

- rejestr przechowujący kod cyklu rozkazu, w którym znajduje się procesor, np.:
- 00 – pobranie
- 01 – adresowanie pośrednie
- 10 – wykonanie
- 11 – przerwanie

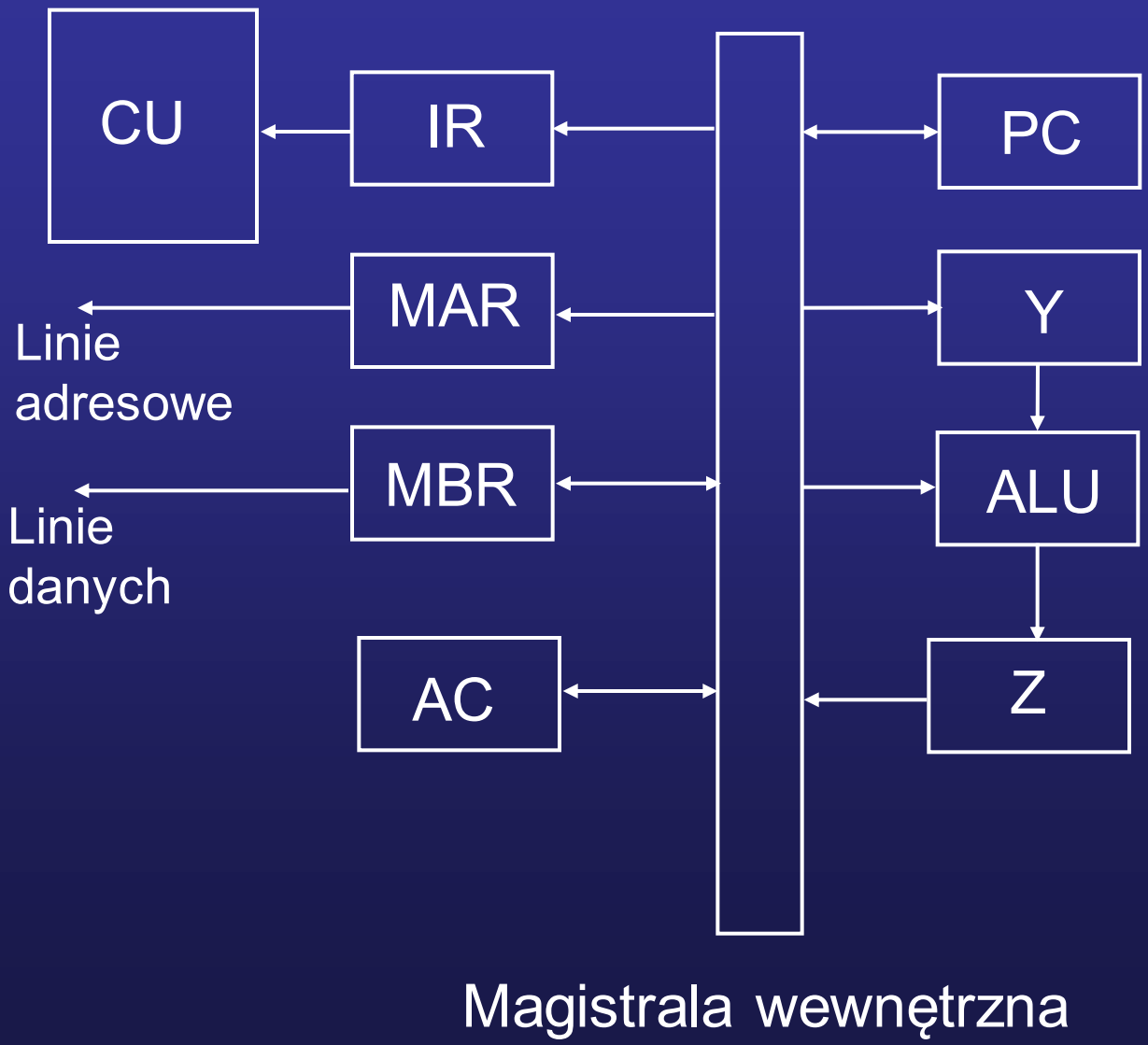
Model jednostki sterującej



Rodzaje sygnałów sterujących

- Aktywujące funkcje ALU
- Aktywujące ścieżkę danych
- Związane z magistralą systemową
- Sterowanie przy pomocy tych sygnałów odbywa się poprzez otwieranie bramek pomiędzy rejestrami a pamięcią
- Konieczna jest do tego znajomość ICC

Wewnętrzna organizacja procesora



t_1: MAR ← (IR(Adr))

t_2: MBR ← M(MAR)

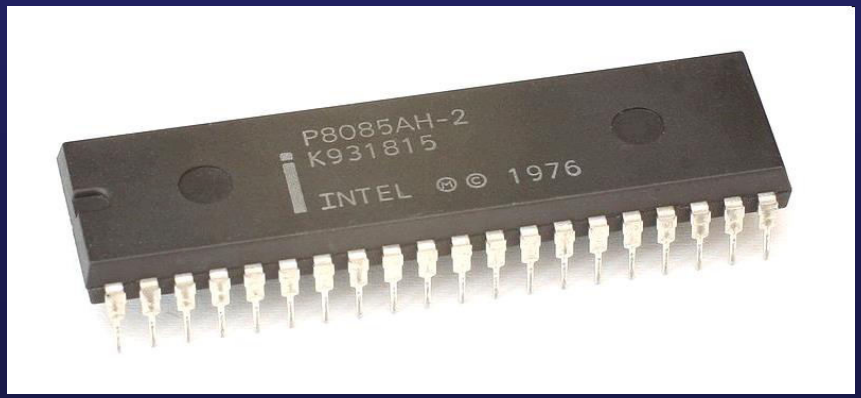
t_3: Y ← (MBR)

t_4: Z ← (AC) + (Y)

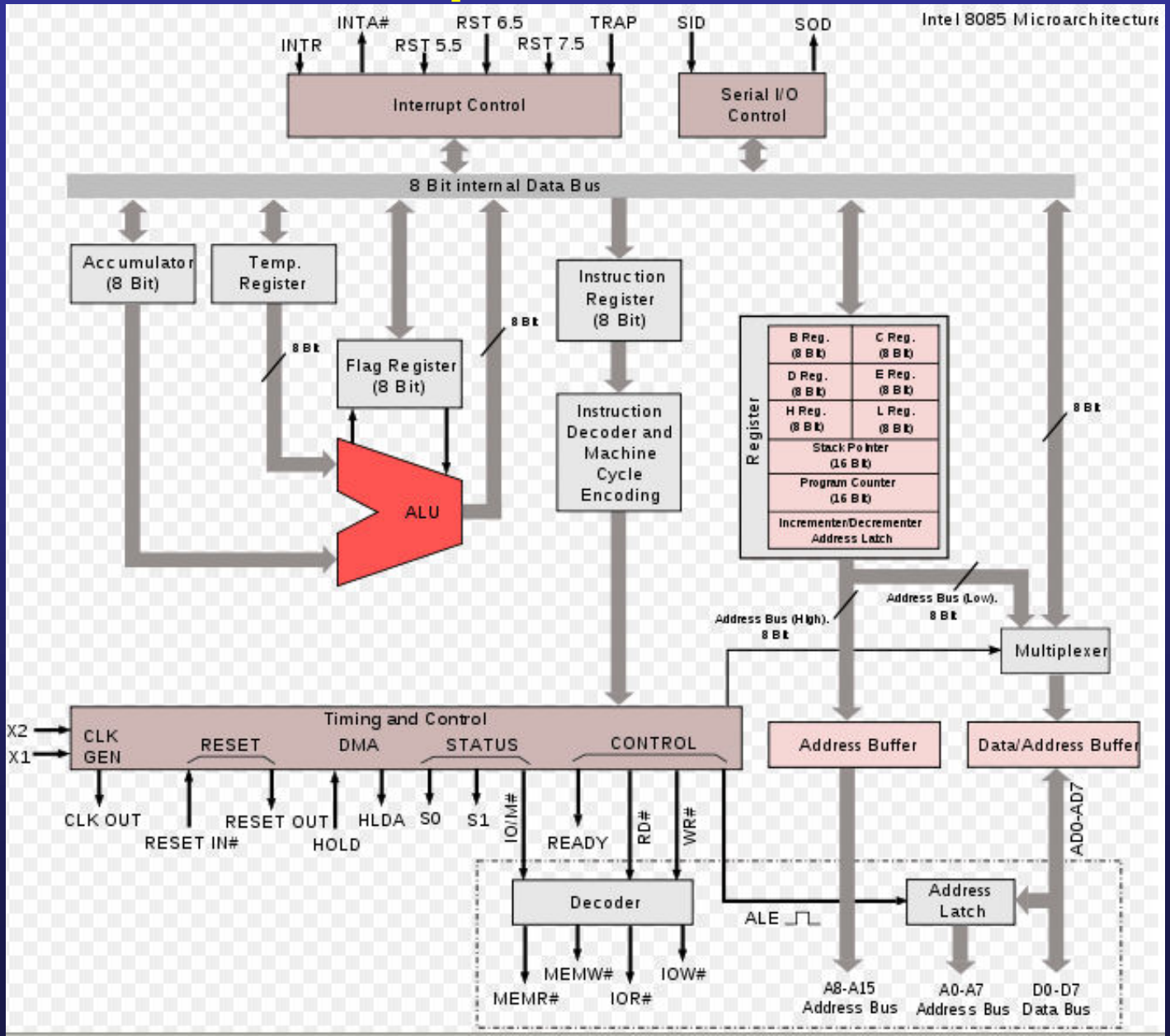
t_5: AC ← (Z)

Procesor Intel 8085 (1977)

- Zgodny z procesorem 8080
- Zasilany napięciem +5V
- Struktura: CU, PC, A, RA, ALU, RF, RR
- Licznik rozkazów – 16 bitów, rejestr adresów – 16 bitów, rejestr rozkazów – 8 bitów
- Stos zewnętrzny w pamięci RAM
- ALU – przetwarza maksymalnie 2 argumenty



Schemat procesora 8085



Przykładowe sygnały zewnętrzne (Intel 8085)

- Sygnały adresów i danych
  - Adresy/dane (A_0-A_7, A_8-A_{15})
  - Szeregowe dane wejściowe/wyjściowe (SID, SOD)
- Sygnały taktujące i sterujące
  - CLK, X_1, X_2
  - Zezwolenie zatrzasku adresu (ALE)
  - IO/M
  - Sterowanie odczytu/zapisu (RD, WR)
- Sygnały inicjowane przez pamięć i we-wy
- Sygnały przerwań
- Inicjowanie procesora (RESET)
- Zasilanie

Rodzaje implementacji jednostki sterującej

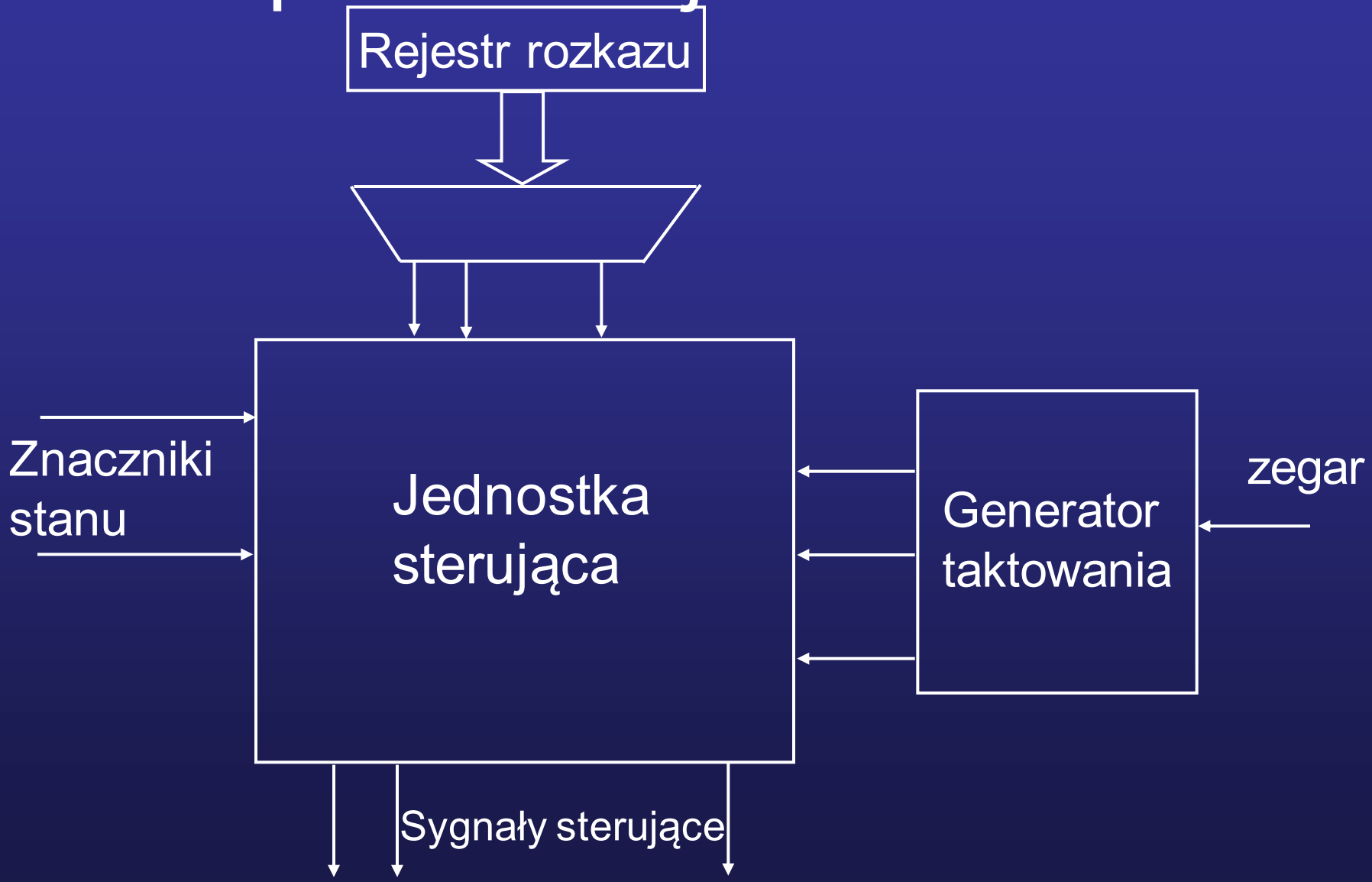
Implementacja układowa

- Szybka (układy logiczne)
- Skomplikowana
- Droga
- Realizacja: układ kombinacyjny
- Używana w maszynach RISC

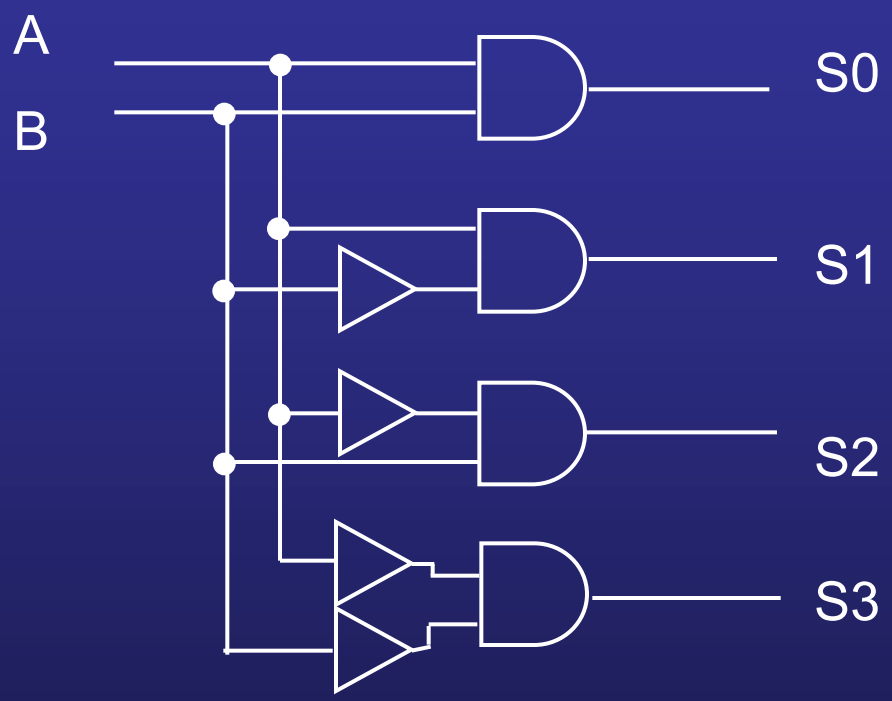
Implementacja mikroprogramowalna

- Wolniejsza
- Elastyczna
- Tania
- Realizacja: mikroprogram
- Używana w maszynach CISC

Implementacja układowa



Dekoder 2 na 4



A	B	S0	S1	S2	S3
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Implementacja mikroprogramowalna jednostki sterującej

- Logika działania jednostki sterującej jest określana przez mikroprogram
- Mikroprogram składa się z mikrorozkazów przechowywanych w pamięci sterującej
- Technika pierwszy raz zastosowana w komputerze System/360 (IBM)

Koncepcja jednostki sterującej (Wilkes – 1951 r.)

- Mikrooperacja jest wierszem matrycy, połączonej z magistralą sterującą i rejestrem następnej mikrooperacji
- Część mikrorozkazu to sygnały sterujące, a część wskazuje następny mikrorozkaz
- Rejestr następnej mikrooperacji jest dwupoziomowy
- Dekoder adresu jest sterowany zegarem
- Mikrorozkaz ma postać jawną

Mikrorozkazy poziome i pionowe

Wewnętrzne sygnały sterujące procesorem

Sygnały sterujące magistrali systemowej

Adres mikrorozkazu

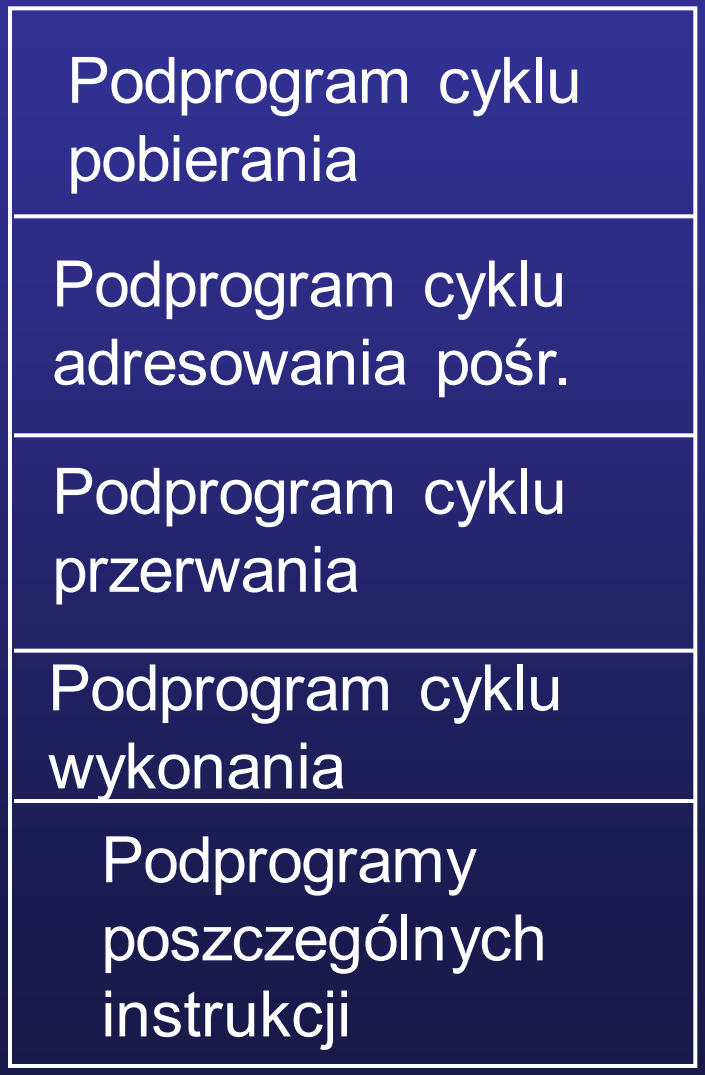
Kody funkcji

Adres mikrorozkazu

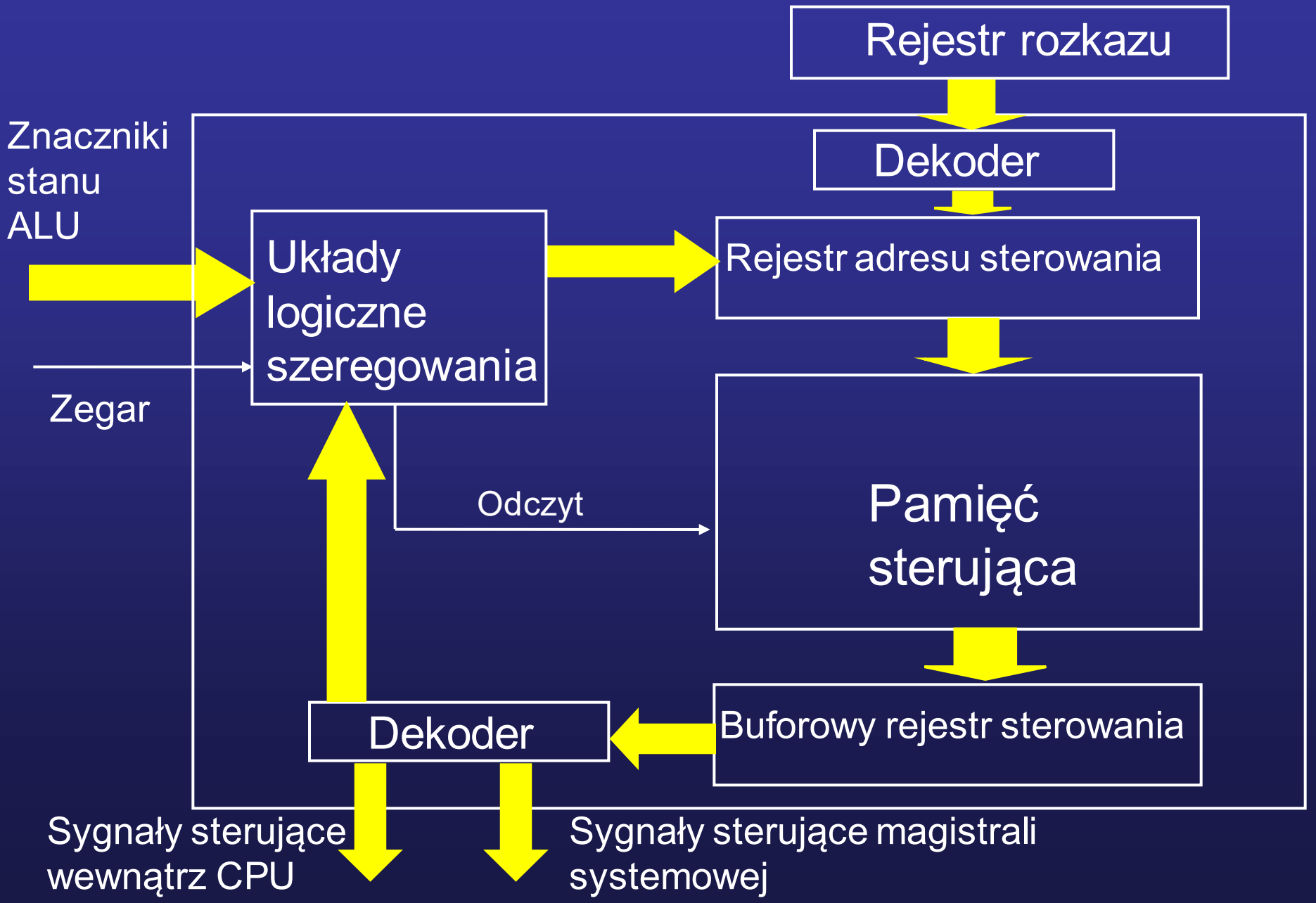
Warunek skoku

Kody zamiast linii sterujących – potrzebne dekodery

Organizacja pamięci sterującej



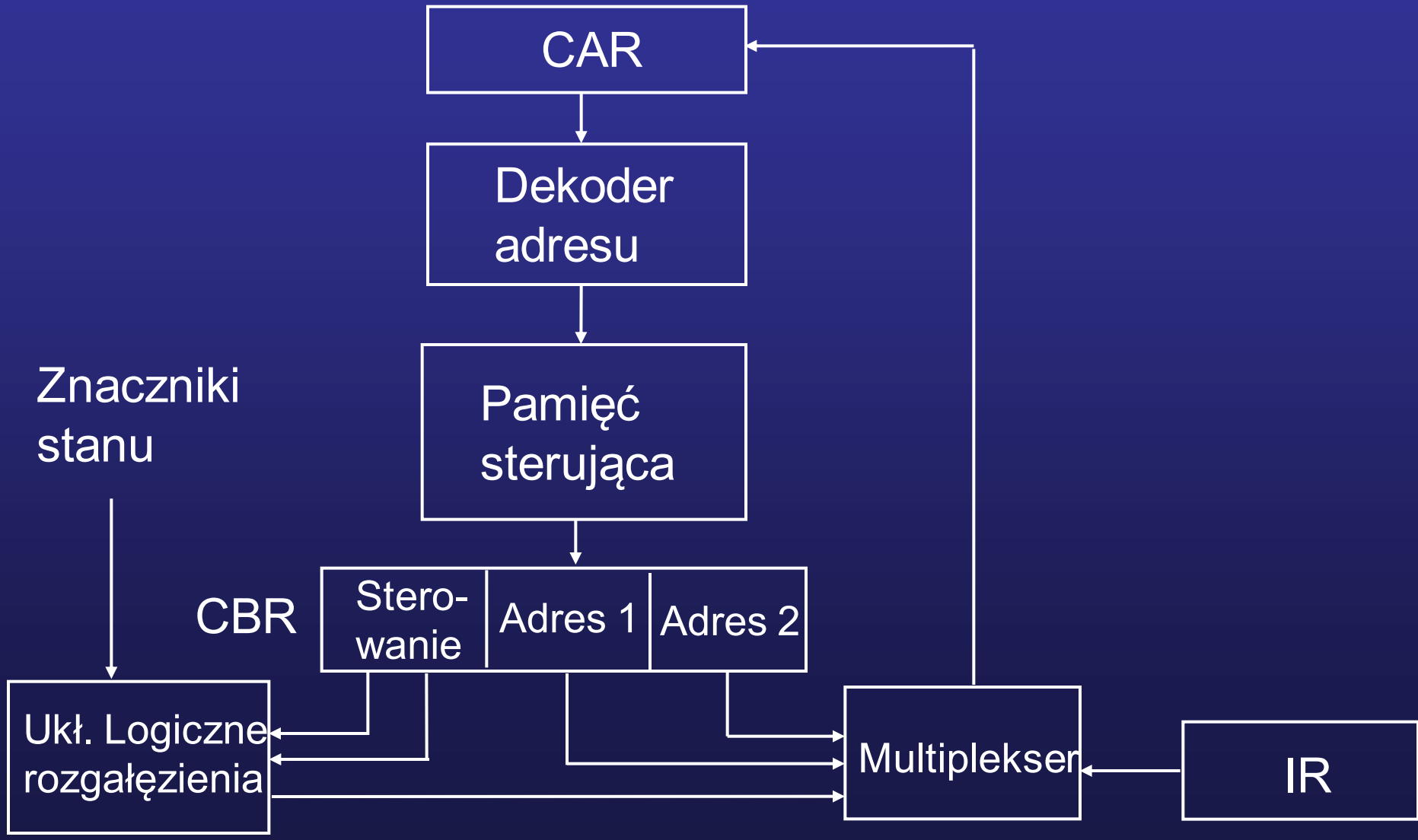
Mikroprogramowalna jednostka sterująca



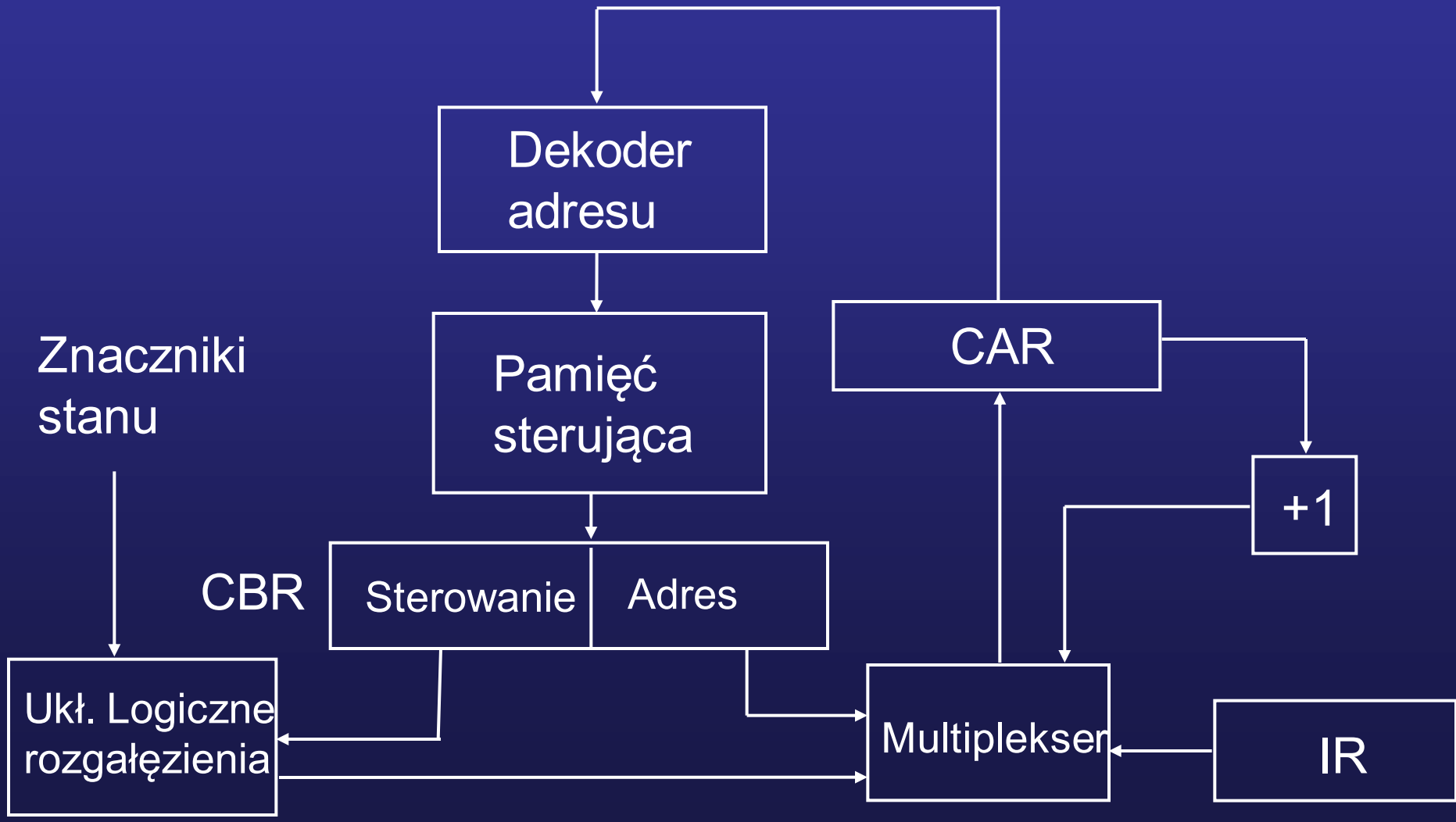
Szeregowanie mikrorozkazów

- Rozmiar rozkazu a czas generowania adresu
- Adres następnego mikrorozkazu jest:
  - W rejestrze rozkazu (na początku cyklu rozkazu)
  - W rozgałęzieniu
  - Kolejny w sekwencji (najczęściej występujący)
- Metody szeregowania:
  - Dwa pola adresowe
  - Jedno pole adresowe
  - Format zmienny

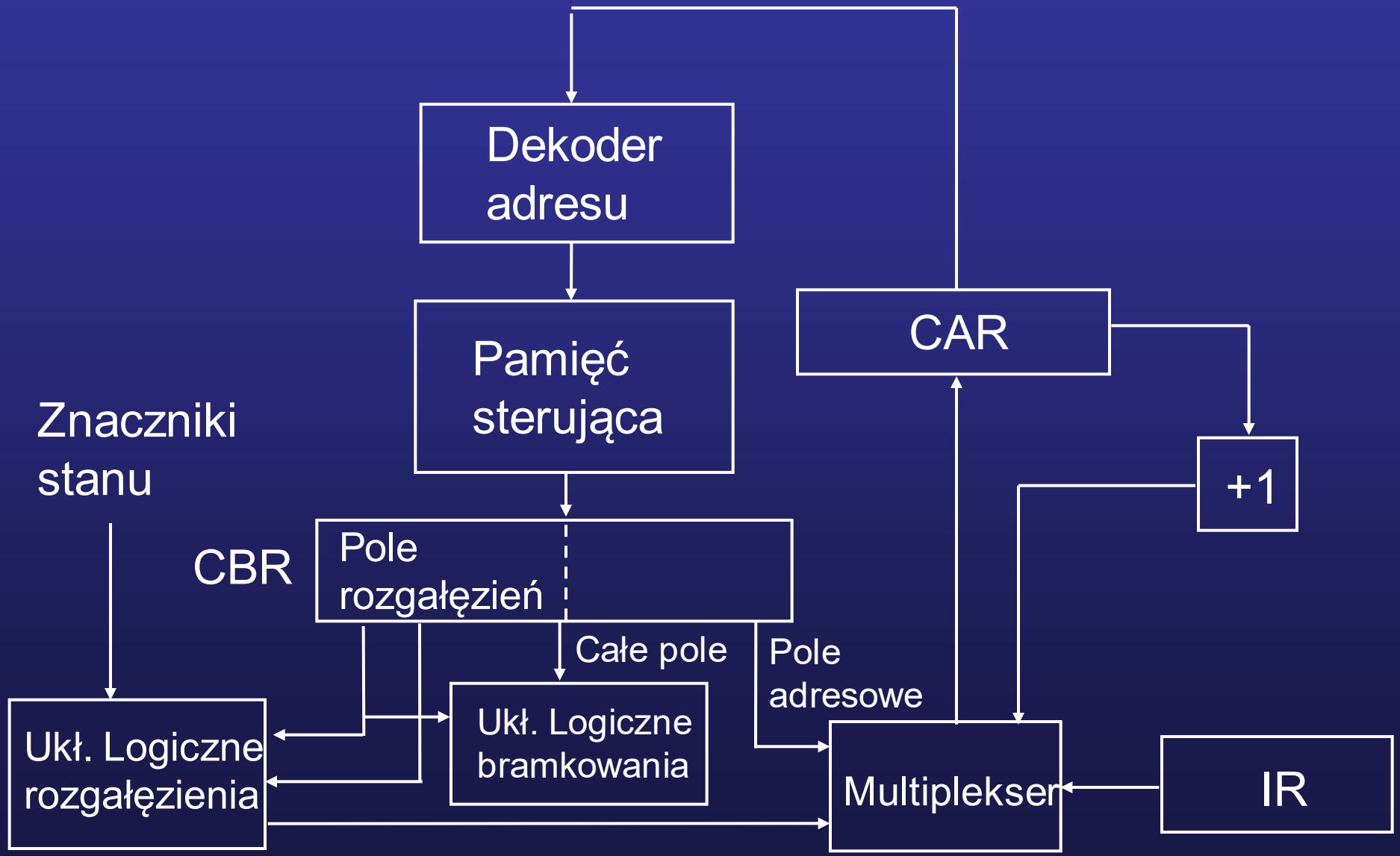
Szeregowanie z dwoma polami adresowymi



Szeregowanie z jednym polem adresowym



Szeregowanie z formatem zmiennym



Generowanie adresów

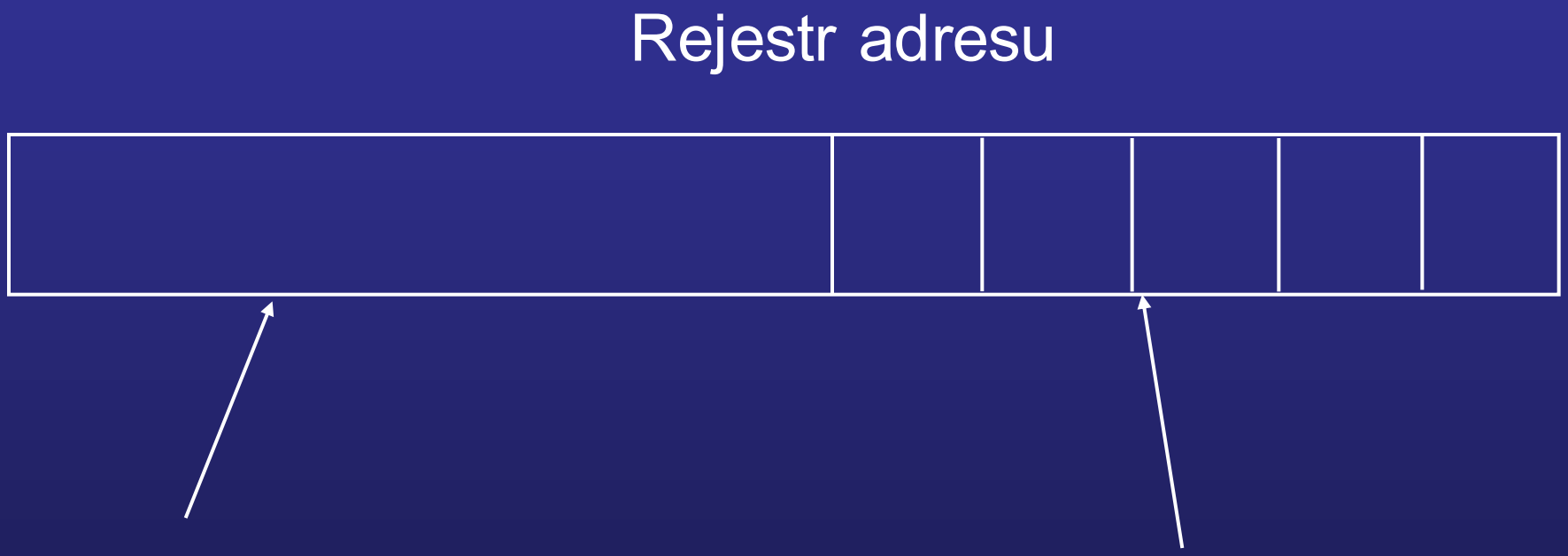
Jawne

- Dwa pola
- Rozgałęzienie warunkowe
- Rozgałęzienie bezwarunkowe

Niejawne

- Odwzorowanie
- Dodawanie
- Sterowanie szczątkowe

Odwzorowanie niejawne - dodawanie



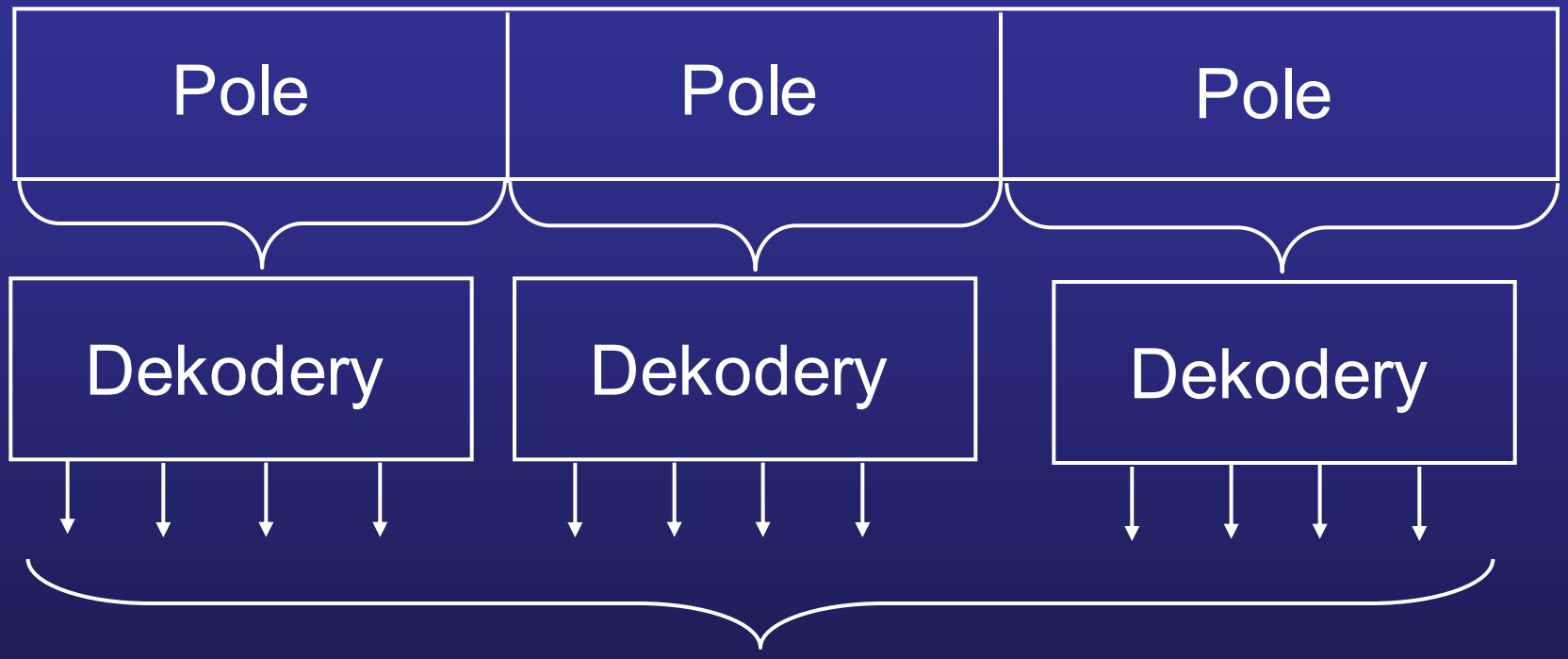
Szeregowanie mikrorozkazów w LSI-11

- Procesor składa się z trzech układów: danych, sterowania i pamięć sterowania
- 22-bitowy mikrorozkaz
- Ok. 4 KB pamięci sterowania
- Metody wyznaczania następnego adresu:
  - Następny w kolejności
  - Odwzorowanie kodu operacji
  - Podprogram standardowy (jednopoziomowy)
  - Testowanie przerwania
  - Rozgałęzienie

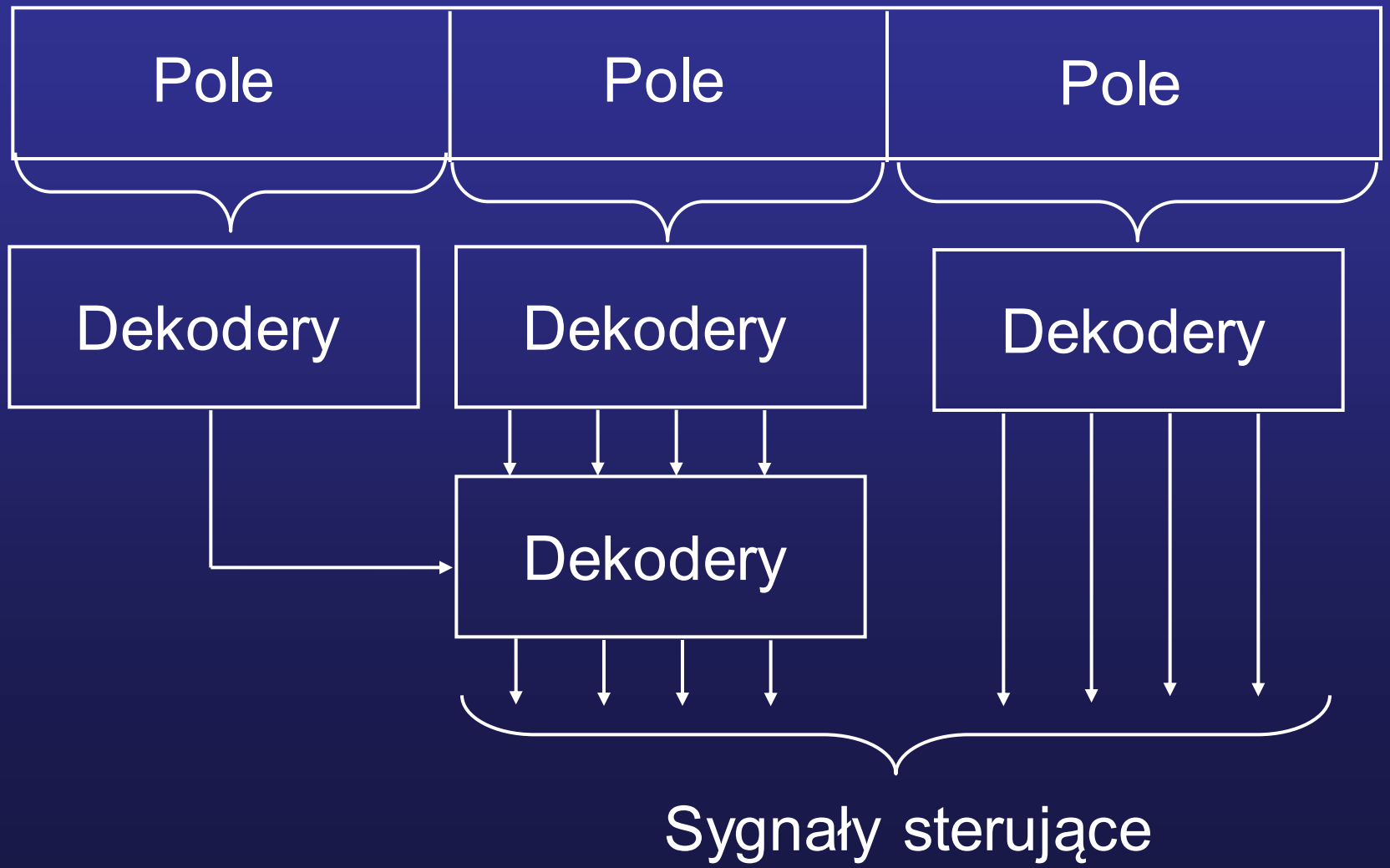
Wykonywanie mikrorozkazów

- Cykl mikrorozkazu składa się z fazy pobrania i wykonania
- Wykonanie powoduje umieszczenie na magistrali wewnętrznej i systemowej sygnałów sterujących oraz określenie adresu następnego rozkazu
- Kombinacja bitów sterujących związana z mikrorozkazem jest kodowana

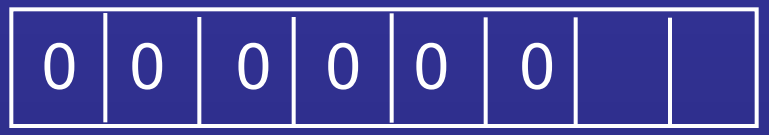
Bezpośrednie kodowanie mikrorozkazu



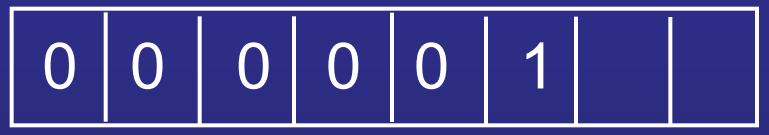
Pośrednie kodowanie mikrorozkazu



Przykłady rozkazów pionowych



MBR ← Rejestr



Rejestr ← MBR

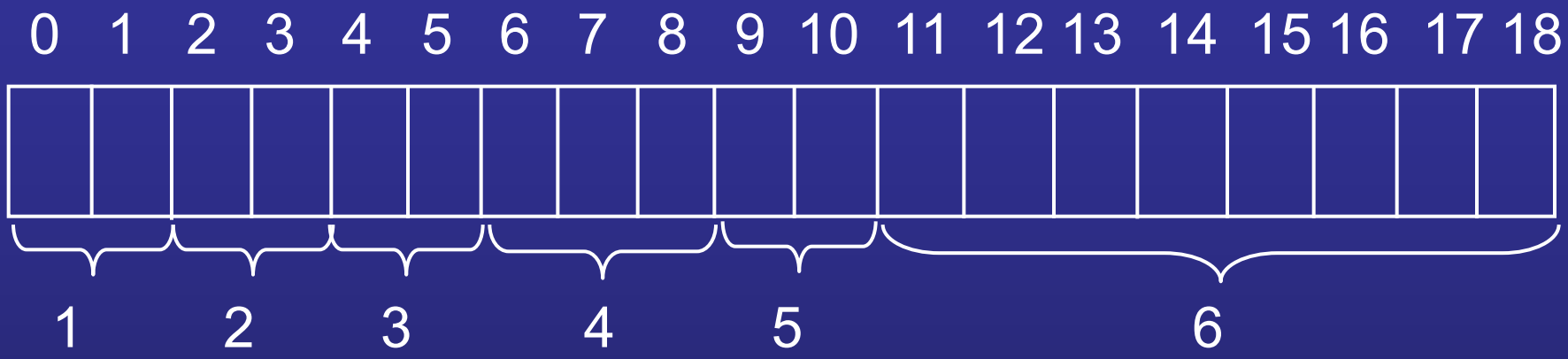
MAR ← Rejestr

Odczyt z pamięci

Zapis do pamięci

AC ← AC + Rejestr

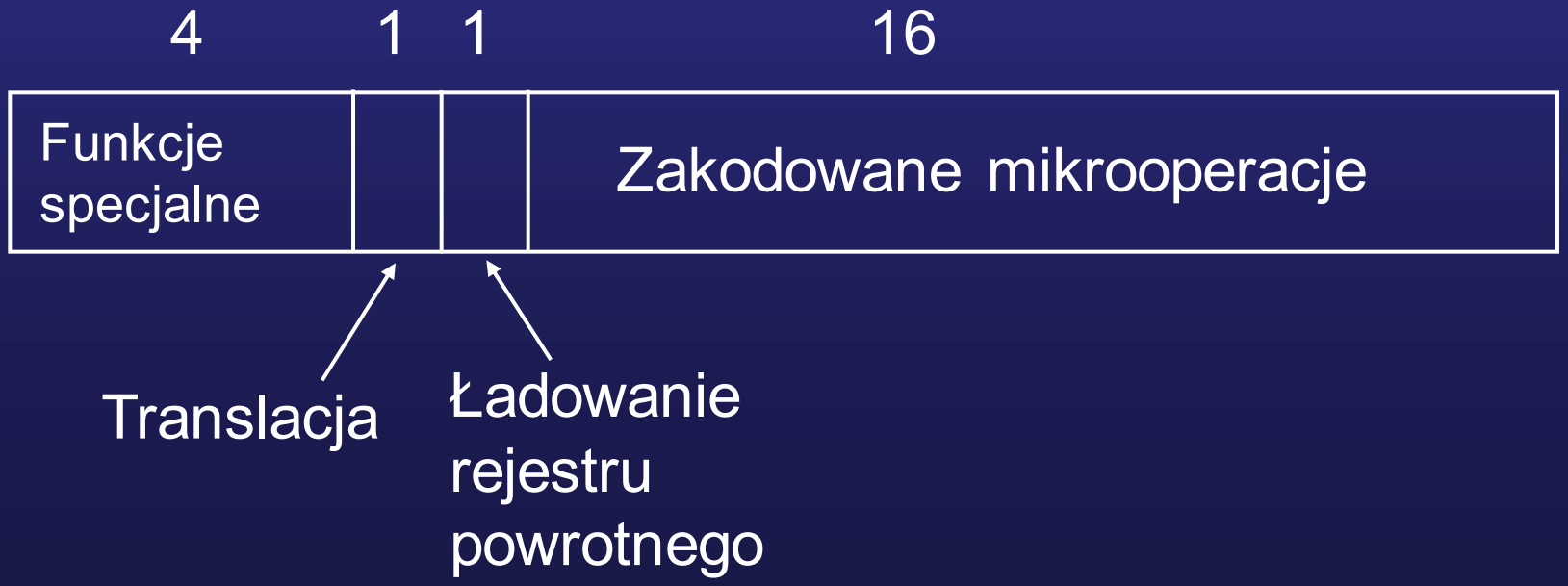
Przykłady rozkazów poziomych



1. transfer rejestrowy
2. operacja pamięci
3. Operacja szeregowania
4. Operacja ALU
5. Wybór rejestru
6. stała

Format mikrorozkazu LSI-11

- Szerokość rozkazu: 22 bity
- Lista mikrorozkazów komplementarna z listą rozkazów maszynowych



Format mikrorozkazu IBM 3033

- Pamięć sterowania – 4 KB słów
- Rozkazy z przedziału 0000-07FF są 108-bitowe
- Rozkazy z przedziału 0800-0FFF są 126-bitowe
- Rozkazy są poziome, ale z kodowaniem

Organizacja i Architektura Komputerów

Wykład nr 8: Hierarchia pamięci. Pamięć podręczna

Piotr Bilski

Własności systemów pamięciowych

- Położenie
- Pojemność
- Jednostka transferu
- Sposób dostępu
- Wydajność
- Rodzaj fizyczny
- Własności fizyczne
- Organizacja

Położenie pamięci

- Procesor (rejestry, pamięć podręczna L1, L2)
- Pamięć wewnętrzna (główna - RAM)
- Pamięć zewnętrzna (pomocnicza – np. dyskowa)

Pojemność pamięci

- Rozmiar słowa
- Liczba słów

- Pojemność pamięci jest wyrażana w bajtach i ich wielokrotnościach, przy czym:

1 B = 8 b

1 KB = 1024 B, 1 MB = 1024 KB itd.

Jednostka transferu

- Jest to liczba linii danych doprowadzonych do modułu pamięci (często równa długości słowa), przy czym:
  - Słowo to najbardziej podstawowa jednostka organizacji pamięci
  - Jednostka adresowalna służy do bezpośredniego adresowania pamięci (słowo lub bajt)
  - Jednostka transferu może, lecz nie musi być równa słowu ani jednostce adresowalnej

Sposób dostępu do pamięci

- Dostęp sekwencyjny (np. pamięci taśmowe)
- Dostęp bezpośredni (pamięci dyskowe)
- Dostęp swobodny (pamięć główna)
- Dostęp skojarzeniowy (pamięci podręczne)

Wydajność pamięci

- Czas dostępu – czas pomiędzy doprowadzeniem adresu na szynę adresową a pojawieniem się informacji na szynie danych
- Czas cyklu – czas dostępu powiększony o czas przerwy pomiędzy następnym dostępem
- Szybkość przesyłu – dla pamięci RAM: 1 / czas cyklu

Fizyczna budowa pamięci

- Półprzewodnikowa (RAM, ROM)
- Magnetyczna (dyski twarde, dyskietki, streamery)
- Optyczna (CD-ROM, DVD-ROM)
- Magnetooptyczna (WORM)

Własności fizyczne

- Ulotność informacji
  - Pamięć ulotna (RAM)
  - Pamięć nieulotna (ROM, RMM)
- Zmiana zawartości
  - Wymazywalne (np. RAM, EPROM, dyski magnetyczne)
  - Niewymazywalne (ROM, część dysków optycznych)

Organizacja pamięci

- Jednopoziomowe („płaskie”)
- Wielopoziomowe (np. cache)

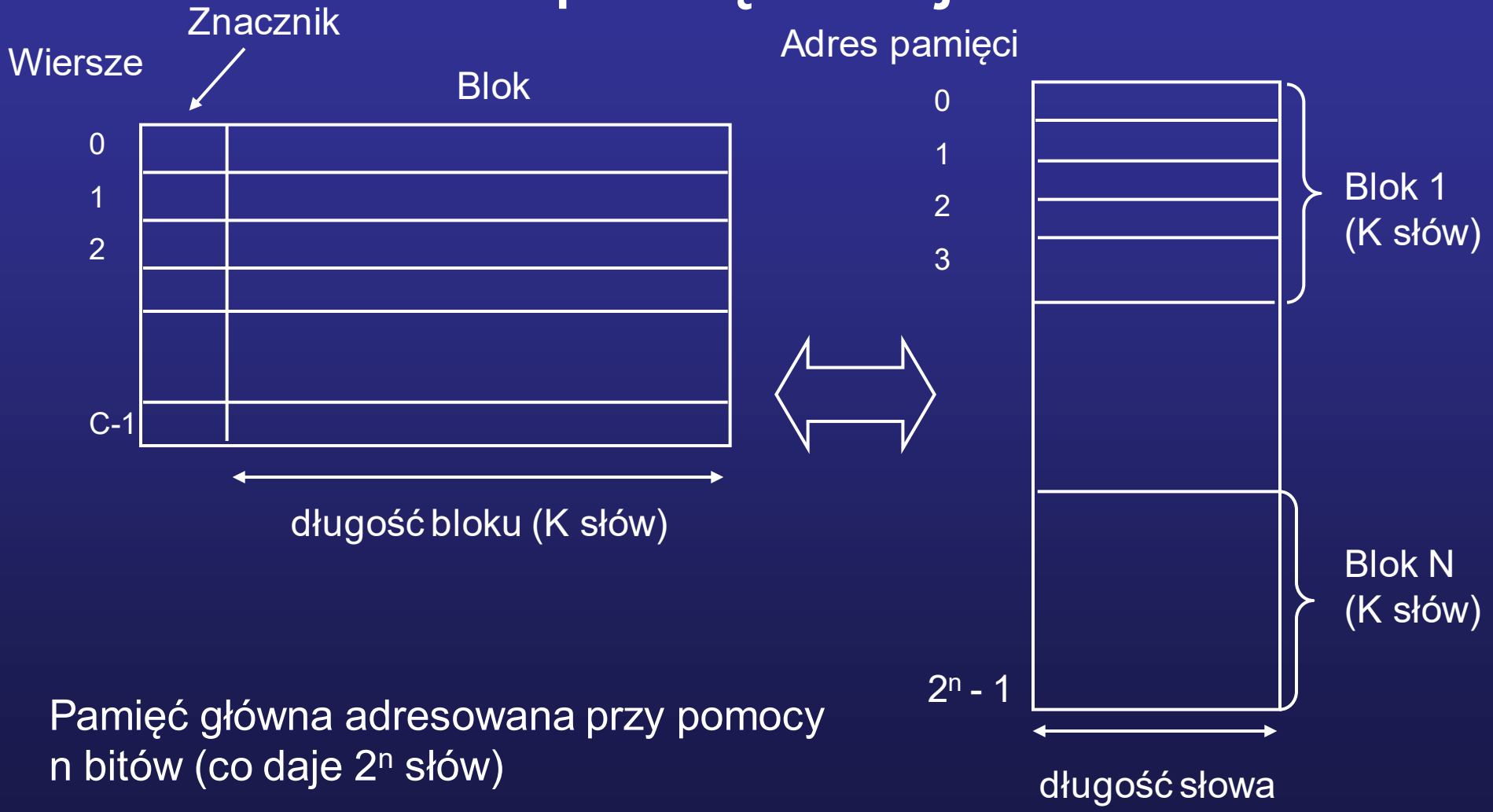
Hierarchia pamięci

Uzasadnienie konieczności pamięci podręcznej

- Zasada lokalności odniesień – wykonywany program składa się z fragmentów znajdujących się obok siebie i wykonywanych jeden po drugim

- Lokalność czasowa
- Lokalność przestrzenna

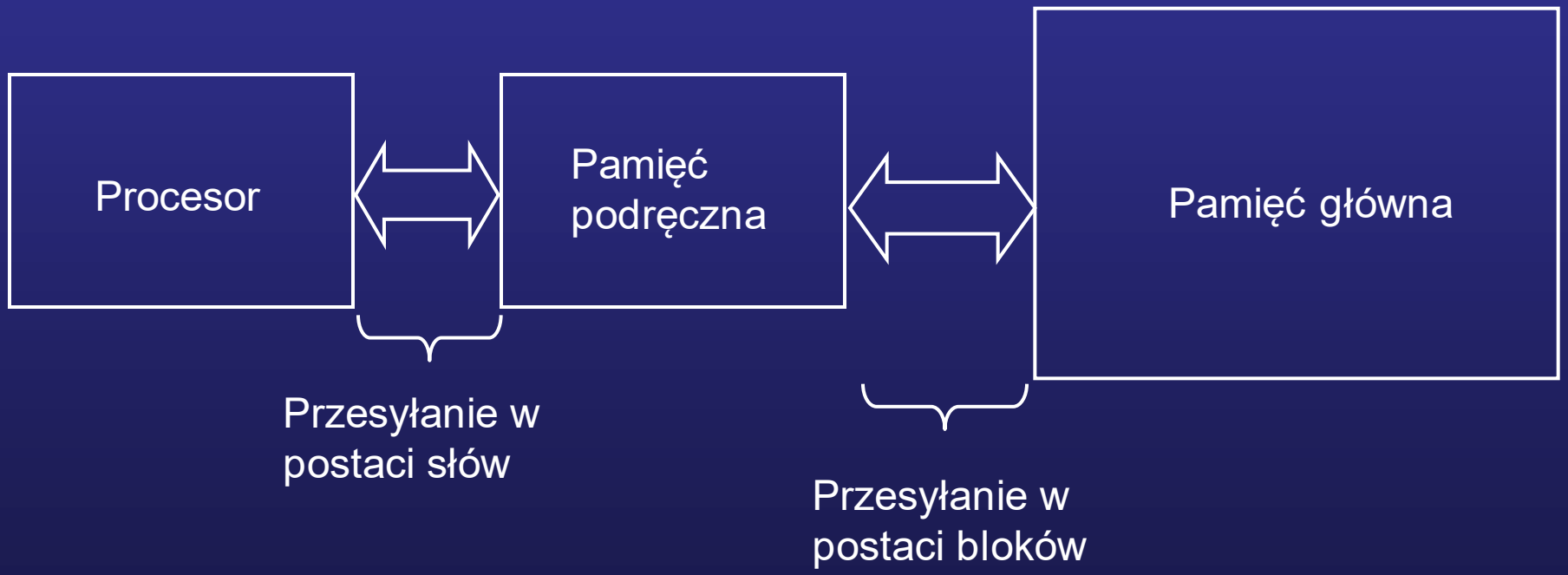
Zasada działania pamięci podręcznej



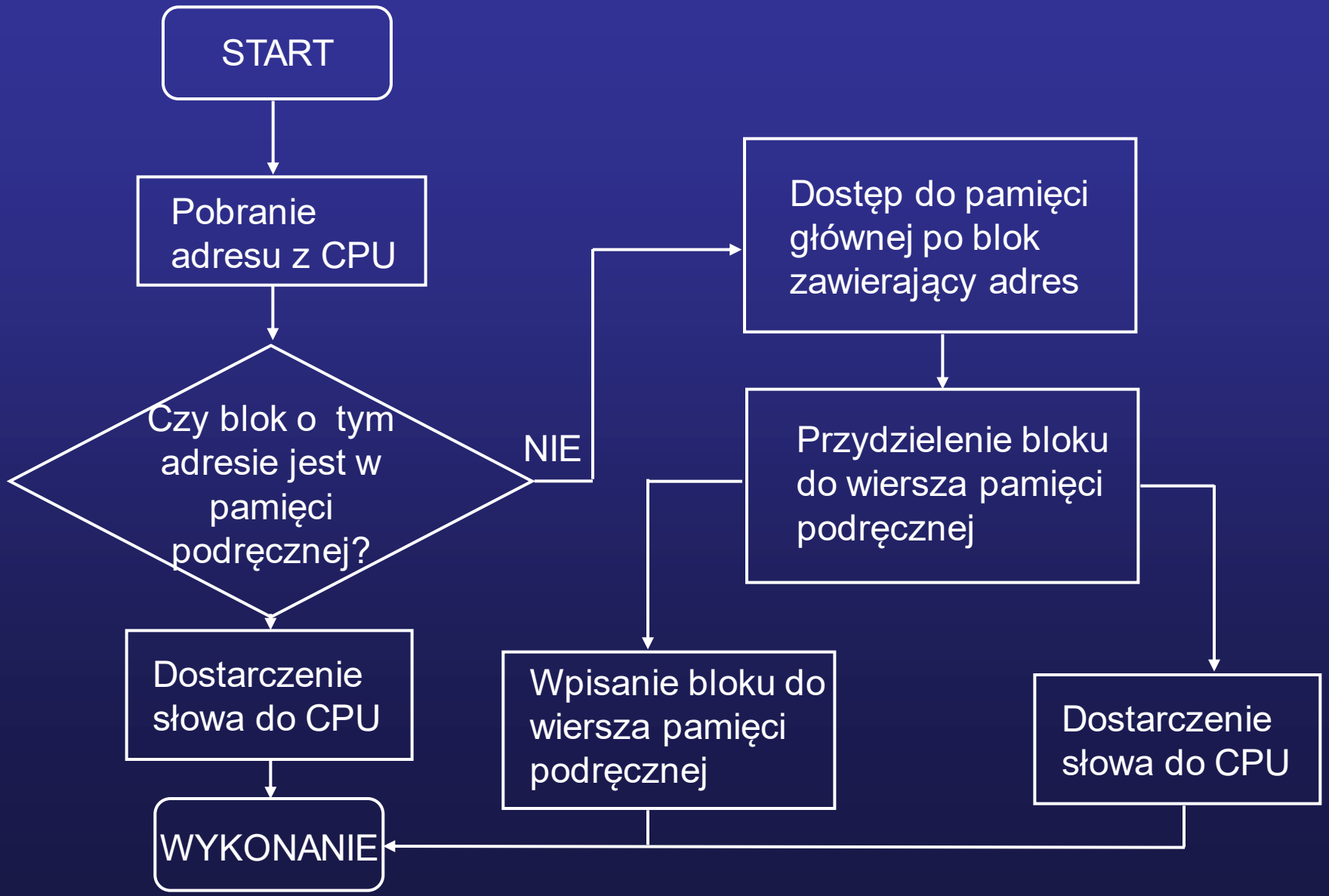
Pamięć główna adresowana przy pomocy n bitów (co daje 2^n słów)

Pamięć podręczna ma C wierszy

Zasada działania pamięci podręcznej (c.d.)



Odczyt z pamięci podręcznej



Szczegóły związane z pamięcią podręczną

- Rozmiar
- Odwzorowywanie
- Algorytm zastępowania
- Metoda zapisu
- Rozmiar wiersza
- Liczba pamięci podręcznych

Rozmiar pamięci podręcznej

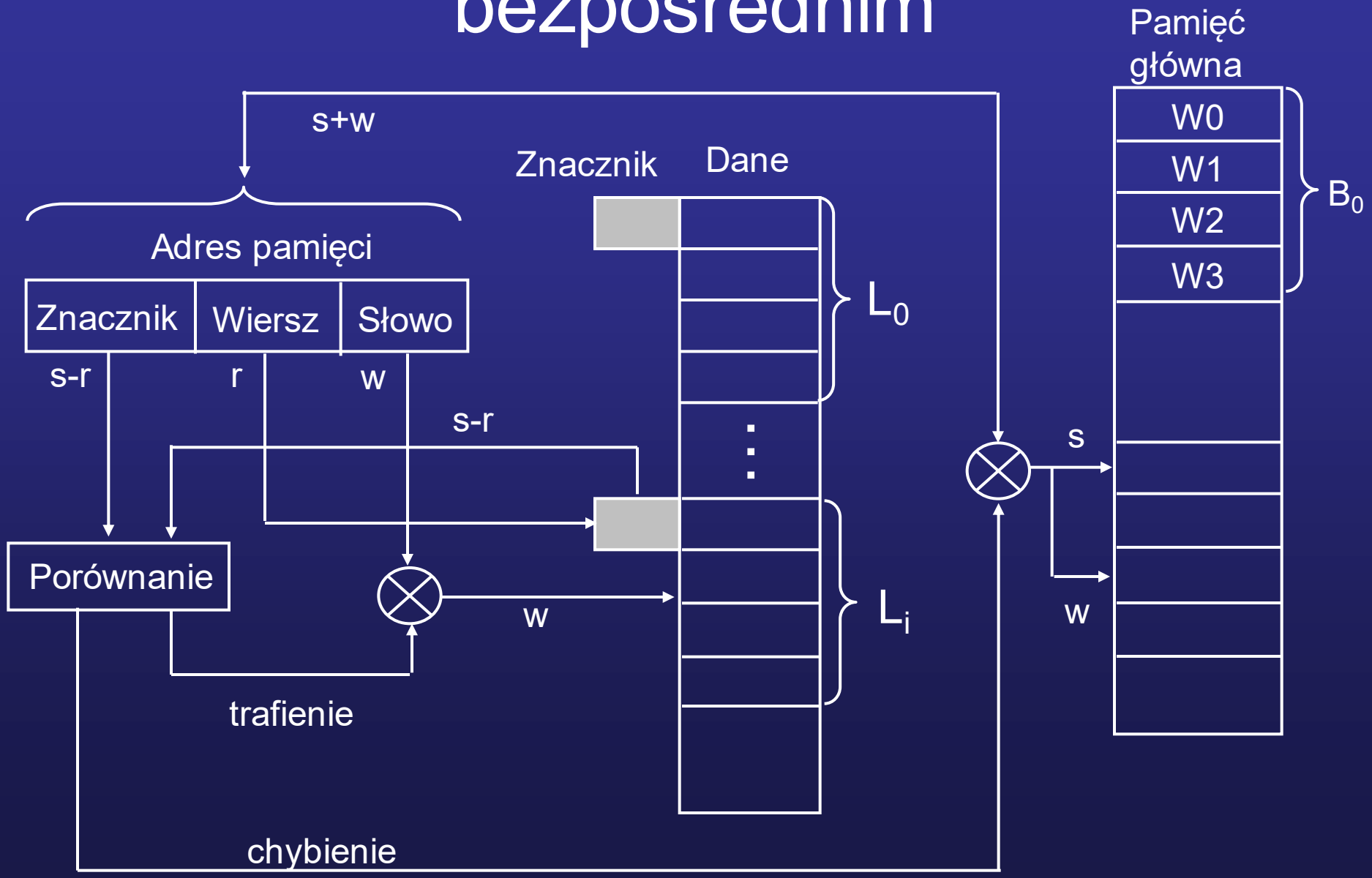
- Minimalizacja kosztu pamięci
- Maksymalizacja szybkości procesora

Procesor	Rodzaj	Rok prod.	Pamięć L1 rozkazy	Pamięć L1 dane	Pamięć L2	Pamięć L3
IBM 360	komputer	1968	16-32 KB		Brak	Brak
IBM 3033	komputer	1978	64 KB		Brak	Brak
80486	PC	1989	8 KB		Brak	Brak
Pentium	PC	1993	8 KB	8 KB	256/512	Brak
PowerPC G4	PC/serv.	1999	32 KB	32 KB	256/1 MB	2 MB
Pentium 4	PC/serv.	2000	8 KB	8 KB	256 KB	Brak
Itanium	PC/serv.	2001	16 KB	16 KB	96 KB	4 MB
Athlon Xp	PC/serv.	1999	64 KB	64 KB	512 KB	Brak
Athlon 64	PC/serv.	2002	64 KB	64 KB	1 MB	Brak

Funkcja odwzorowywania

- Wierszy w pamięci podręcznej jest mniej niż bloków w pamięci głównej
- Istnieją trzy metody:
  - Bezpośrednia
  - Skojarzeniowa (asocjacyjna)
  - Sekcyjno-skojarzeniowa

Pamięć podręczna o odwzorowaniu bezpośrednim



Odwzorowanie bezpośrednie c.d.

- i – numer wiersza pamięci podręcznej
- j – numer bloku z pamięci głównej
- m – liczba wierszy w pamięci podręcznej

$$i = j \bmod m$$

Długość adresu: s+w bitów
Liczba adresowalnych jednostek: 2^{s+w} słów
Rozmiar bloku = rozmiar wiersza: 2^{w} słów
Liczba bloków w pamięci głównej: 2^{s}
Liczba wierszy w pamięci podręcznej: 2^{r}

Efekt odwzorowania bezpośredniego

Wiersz pamięci podręcznej	Przypisane bloki pamięci głównej
0	0, m, 2m, ... , 2^s – m
1	1, m+1, 2m+1, ..., 2^s – m +1
...	...
m-1	m-1, 2m-1, 3m-1, ... , 2^s – 1

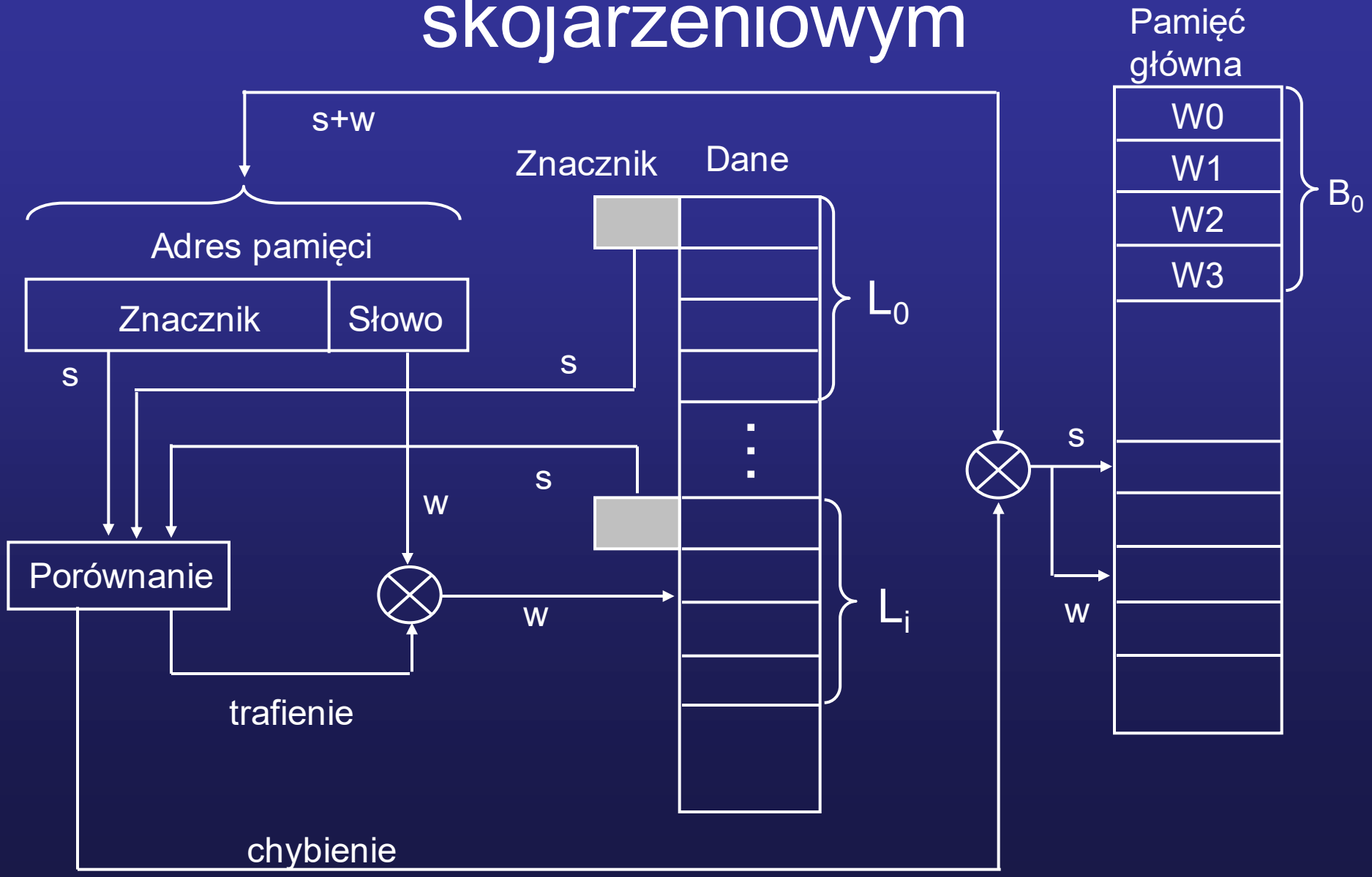
Przykład odwzorowania bezpośredniego

- dla pamięci podręcznej liczącej 2^{14} wierszy (po 4 B każdy) i pamięci głównej o pojemności 16 MB:

Wiersz pamięci podręcznej	Przypisane bloki pamięci głównej
0	000000, 010000, ... , FF0000
1	000004, 010004, ..., FF0004
...	...
2^{14}-1	00FFFC, 01FFFC, ... , FFFFFC

Szerokość wiersza: 8 b znacznika, 32 b na dane

Pamięć podręczna o odwzorowaniu skojarzeniowym



Odwzorowanie skojarzeniowe c.d.

Długość adresu: s+w bitów

Liczba adresowalnych jednostek: 2^{s+w} słów

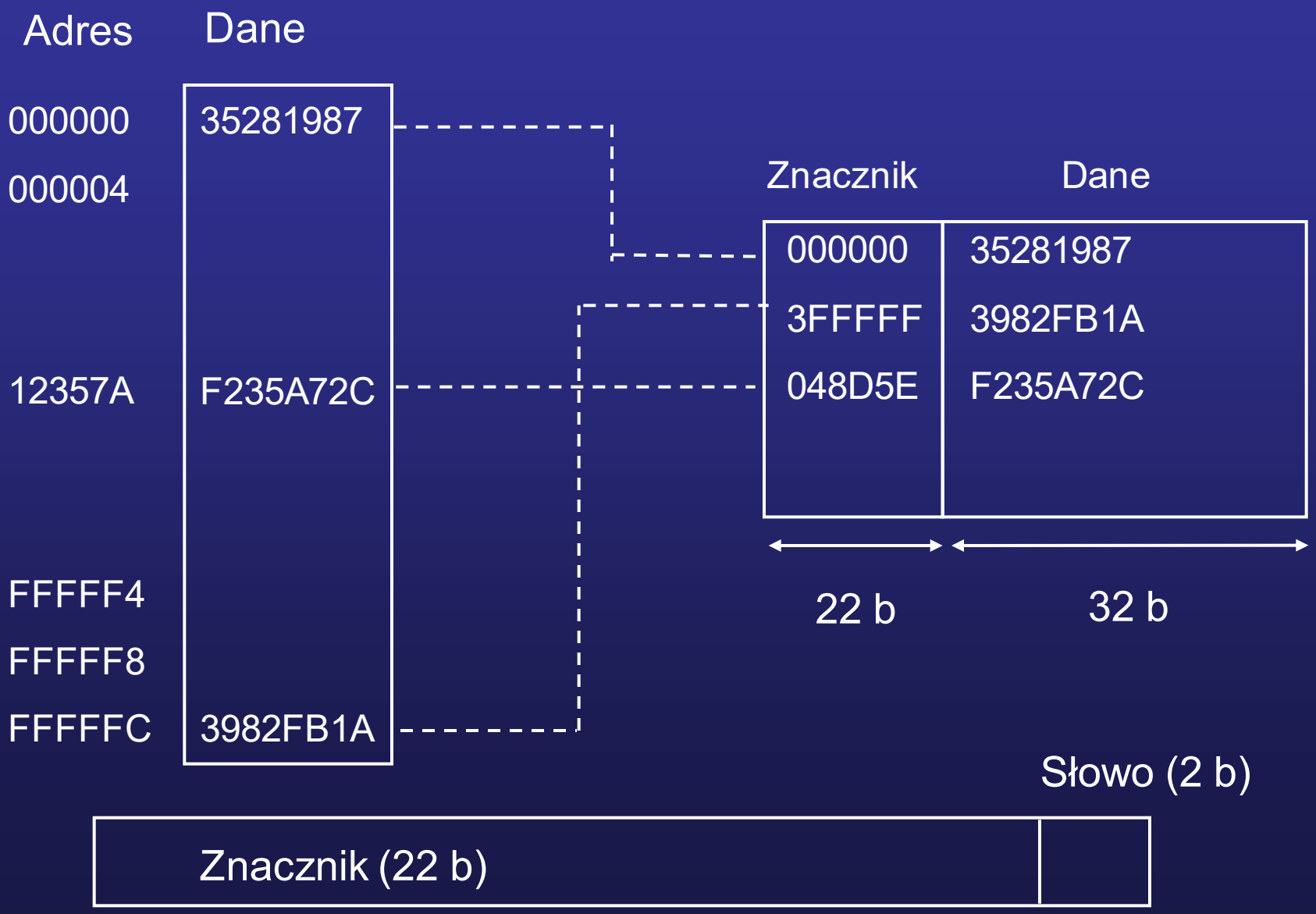
Rozmiar bloku = rozmiar wiersza: 2^{w} słów

Liczba bloków w pamięci głównej: 2^{s}

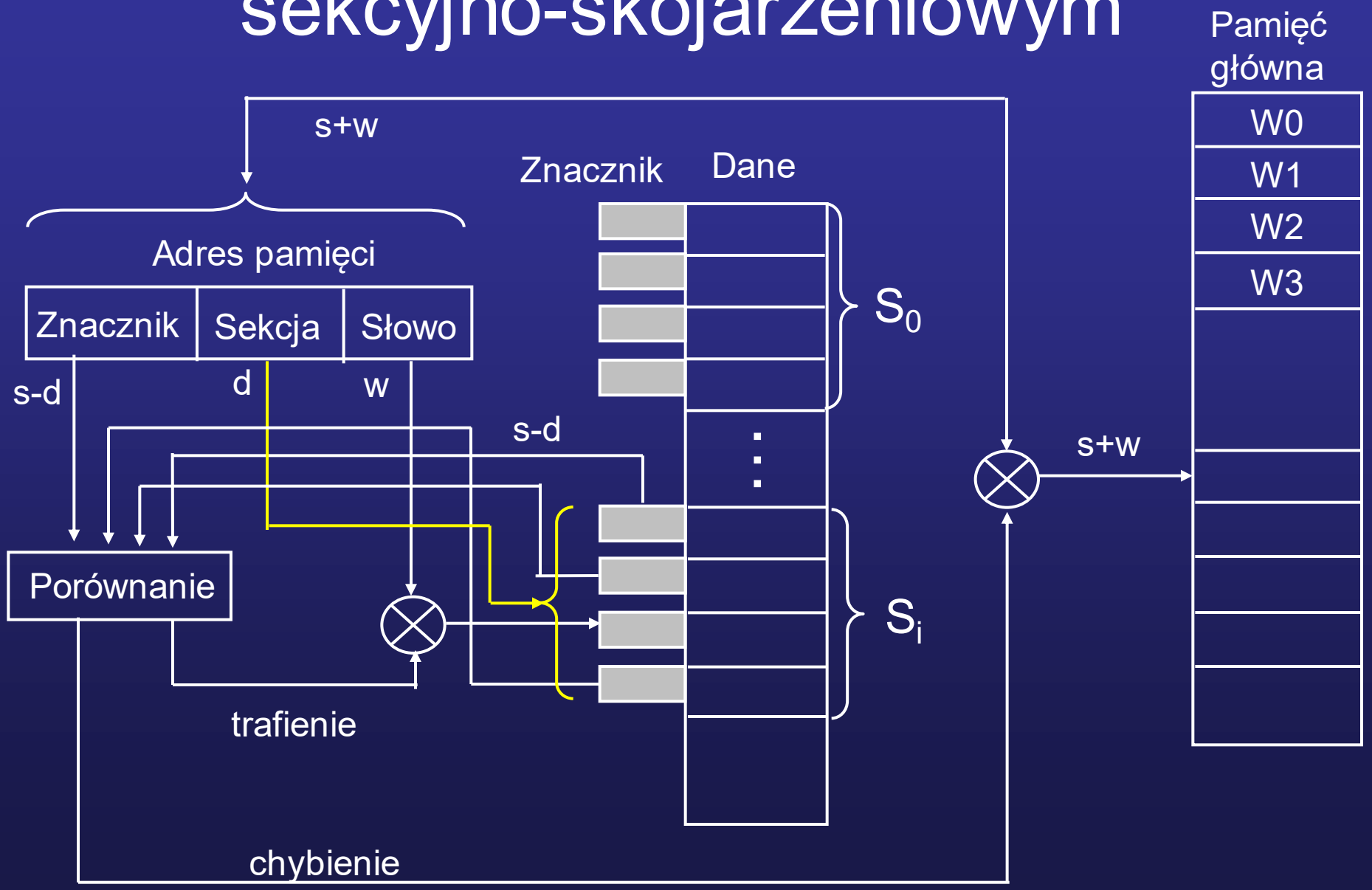
Liczba wierszy w pamięci podręcznej: dowolna

Rozmiar znacznika: s bitów

Przykład odwzorowania skojarzeniowego



Pamięć podręczna o odwzorowaniu sekcyjno-skojarzeniowym



Odwzorowanie sekcyjno-skojarzeniowe c.d.

- i – numer wiersza pamięci podręcznej
- j – numer bloku z pamięci głównej
- m – liczba wierszy w pamięci podręcznej

$$m = v \times k$$
$$i = j \bmod v$$

Długość adresu: s+w bitów

Liczba adresowalnych jednostek: 2^{s+w} słów

Rozmiar bloku = rozmiar wiersza: 2^{w} słów

Liczba bloków w pamięci głównej: 2^{s}

Odwzorowanie sekcyjno-skojarzeniowe c.d.

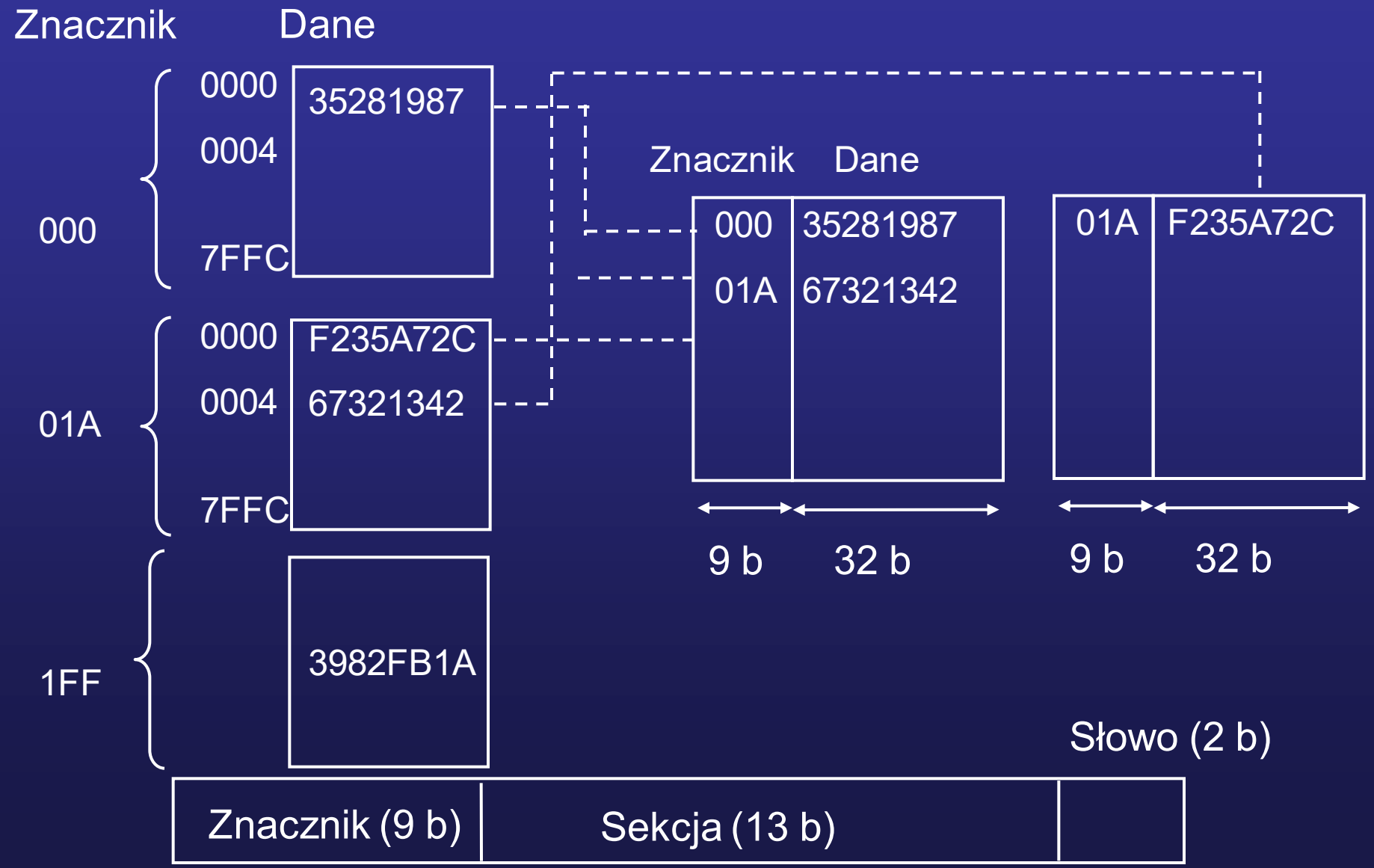
Liczba wierszy w sekcji: k

Liczba sekcji $v = 2^d$

Liczba wierszy w pamięci podręcznej $kv = k \times 2^d$

Rozmiar znacznika: (s-d) bitów

Przykład odwzorowania sekcyjno-skojarzeniowego



Algorytmy zastępowania zawartości pamięci podręcznej

- najdawniej używany (LRU)
- pierwszy wchodzi-pierwszy wychodzi (FIFO)
- najrzadziej używany (LFU)
- wybór losowy

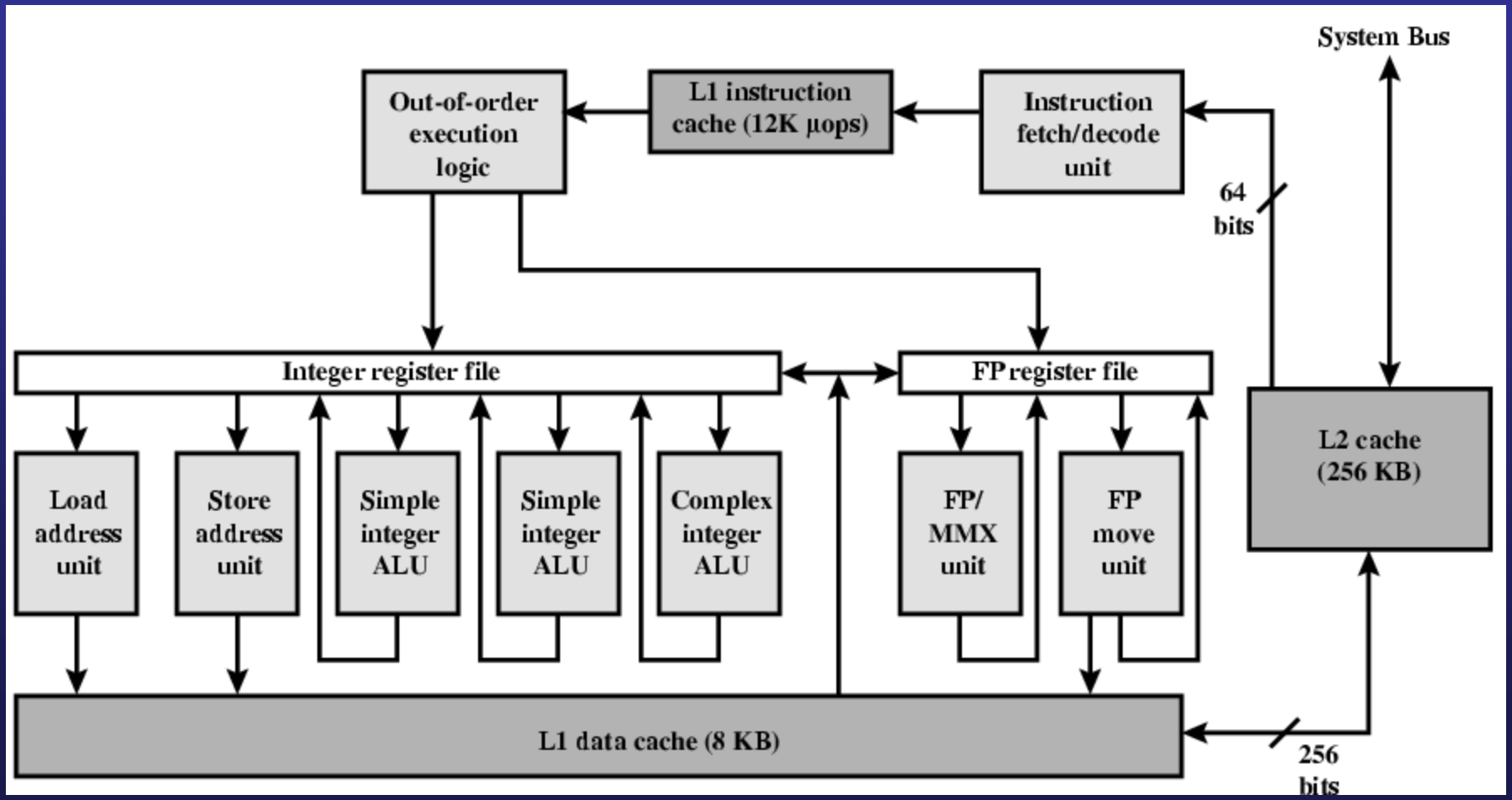
Algorytmy zapisu do pamięci podręcznej

- zapis jednoczesny (write through)
- zapis opóźniony (write back)
- system zapewnienia spójności (wiele procesorów wyposażonych w cache)
  - obserwacja magistrali z zapisem jednoczesnym
  - przezroczystość sprzętowa
  - pamięć nieodwzorowywana przez pamięć podręczną

Pozostałe problemy

- rozmiar wiersza a wielkość bloku
- liczba pamięci podręcznych
  - pamięć wyższego rzędu jest w jednym układzie z procesorem, pracuje z jego częstotliwością
  - pamięć niższego rzędu pracuje z częstotliwością magistrali (bo jest blisko płyty głównej)

Pamięć podręczna w P4



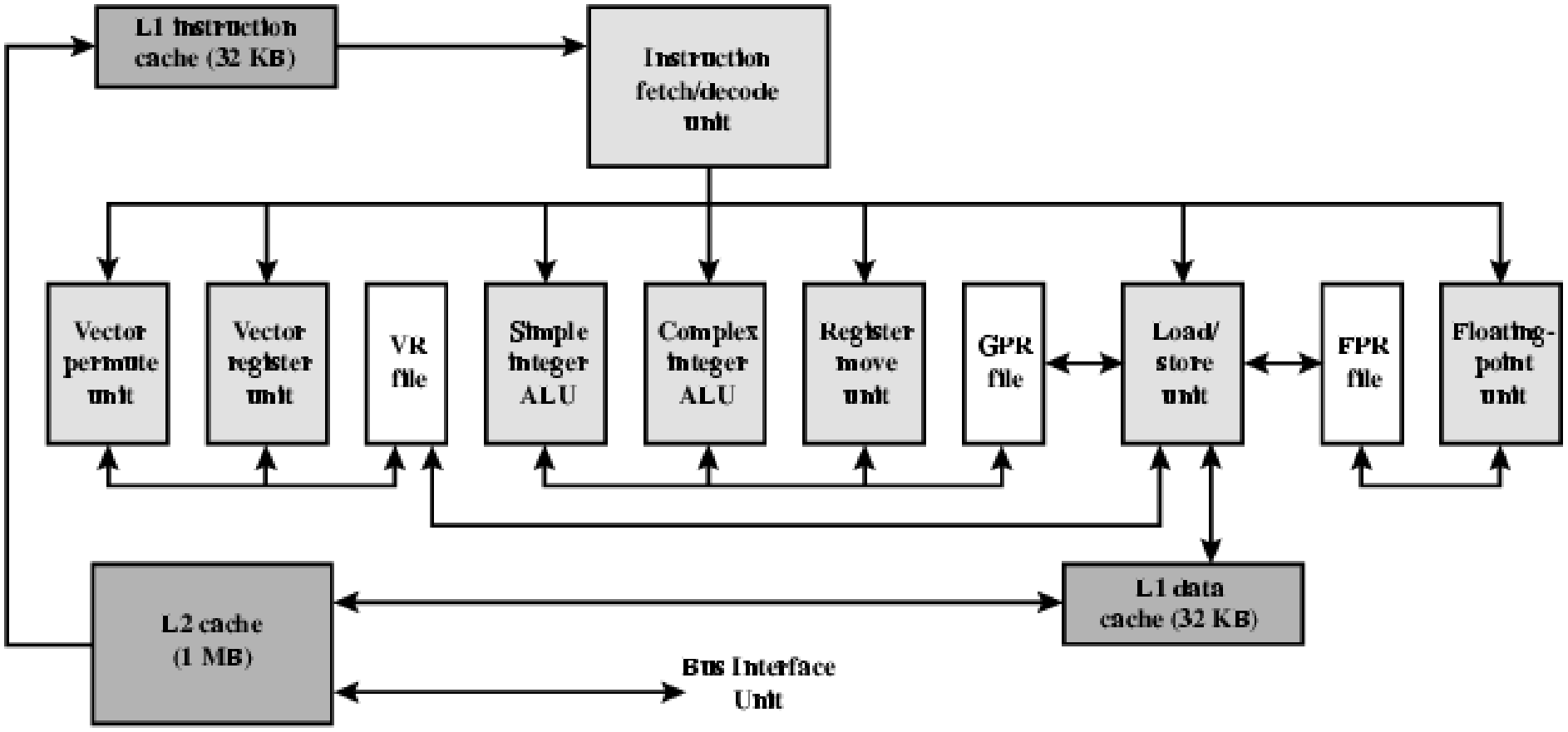
Rdzeń procesora P4

- Jednostka pobierania / dekodowania rozkazów
  - pobiera instrukcje z pamięci L2
  - dekoduje je na mikrooperacje
  - przesyła mikrooperacje do pamięci L1
- Układ wykonywania rozkazów poza kolejnością
  - Kolejkuje mikrooperacje
- Jednostki wykonawcze
  - wykonują mikrooperacje
  - dane pobierają z pamięci L1
  - wyniki zapisują do rejestrów
- Podsystem pamięci
  - komunikuje się z magistralą systemową i pamięcią L2

Pamięć podręczna w PowerPC

Procesor	Rozmiar	B / wiersz	Organizacja
PowerPC 601	1 x 32 KB	32	8-drożna
PowerPC 603	2 x 8 KB	32	2-drożna
PowerPC 604	2 x 16 KB	32	4-drożna
PowerPC 620	2 x 32 KB	64	8-drożna
PowerPC G3	2 x 32 KB	64	8-drożna
PowerPC G4	2 x 32 KB	32	8-drożna

Pamięć podręczna w PowerPC c.d.



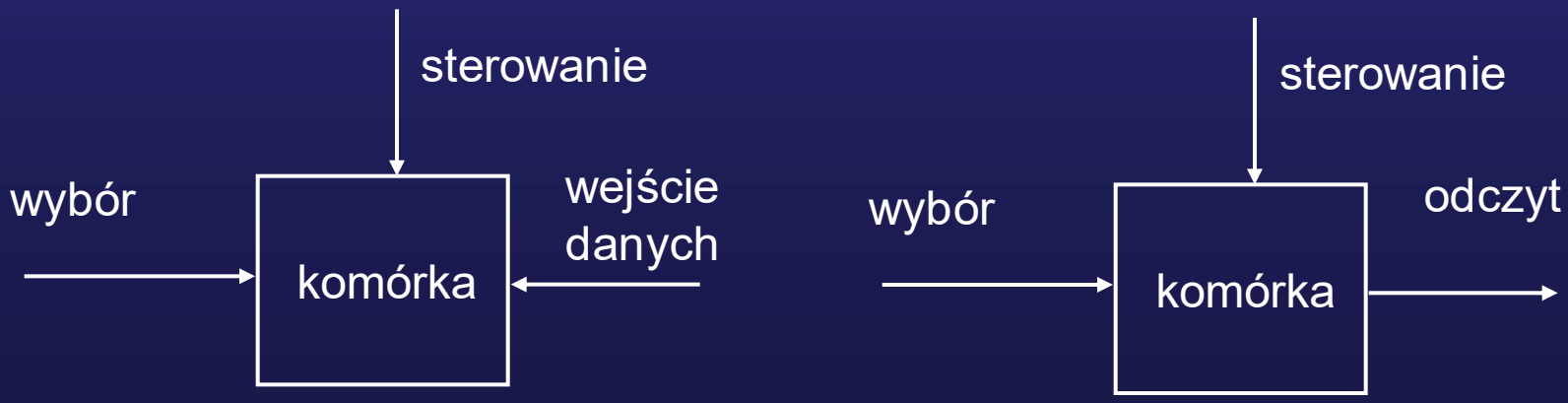
Organizacja i Architektura Komputerów

Wykład nr 9: Pamięć główna i zewnętrzna

Piotr Bilski

Współczesna pamięć główna

- Podstawowy element – komórka pamięci
- Komórki pogrupowane są w słowa
- Komórki mają dwa stany służące do przechowywania cyfr binarnych
- Możliwy jest zapis i/lub odczyt



Rodzaje pamięci półprzewodnikowych

- Pamięć o dostępie swobodnym (RAM)
  - Dynamiczna (DRAM)
  - Statyczna (SRAM)
- Pamięć tylko do odczytu (ROM)
  - Programowalna (PROM)
  - Wymazywalna (EPROM)
  - Błyskawiczna (flash ROM)
  - Elektrycznie wymazywalna (EEPROM)

Cechy pamięci

Rodzaj	Kategoria	Wymazywanie	Zapis	Ulotność
RAM	odczyt/ zapis	elektryczne	elektr.	tak
ROM	odczyt	brak	maska	nie
PROM	odczyt	brak	elektr.	nie
EPROM	gł. odczyt	UV	elektr.	nie
Flash	gł. odczyt	elektryczne	elektr.	nie
EEPROM	gł. odczyt	elektryczne	elektr.	nie

Organizacja logiczna pamięci

- Ile komórek jest adresowane tym samym adresem?
- Najczęściej ich liczba wyznaczona jest przez organizację logiczną jednostki danych komputera – długość słowa

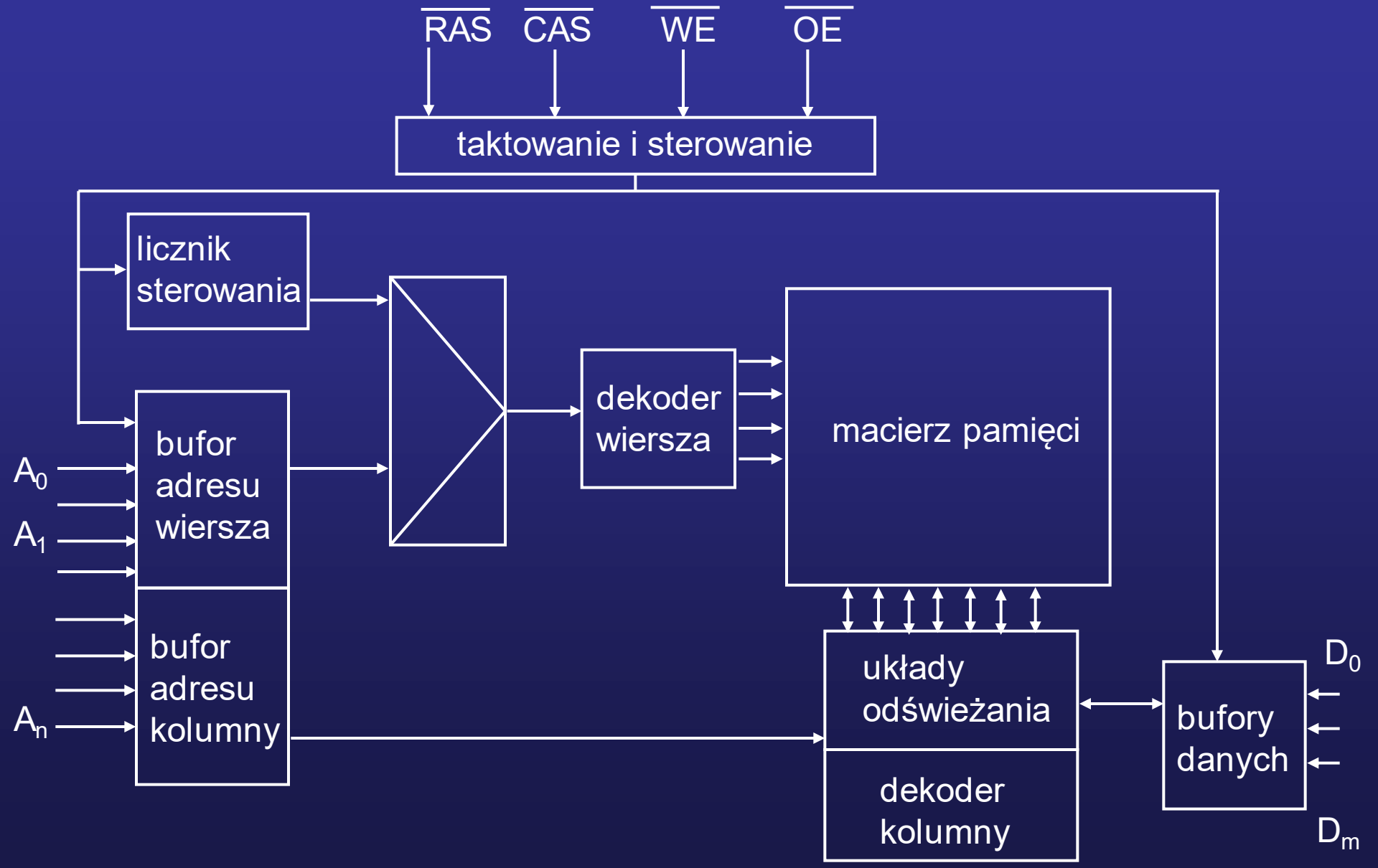
- Np.: 16 Mb $\rightarrow$ 1M $\times$ 16 b

  $\rightarrow$ 0,5 M $\times$ 32 b

Pamięć DRAM

- Ładunek przechowywany w kondensatorze oznacza logiczną jedynkę, jego brak – zero
- odczyt powoduje wymazanie zawartości komórki
- kondensator rozładowuje się z czasem, jego zawartość trzeba cyklicznie odświeżać

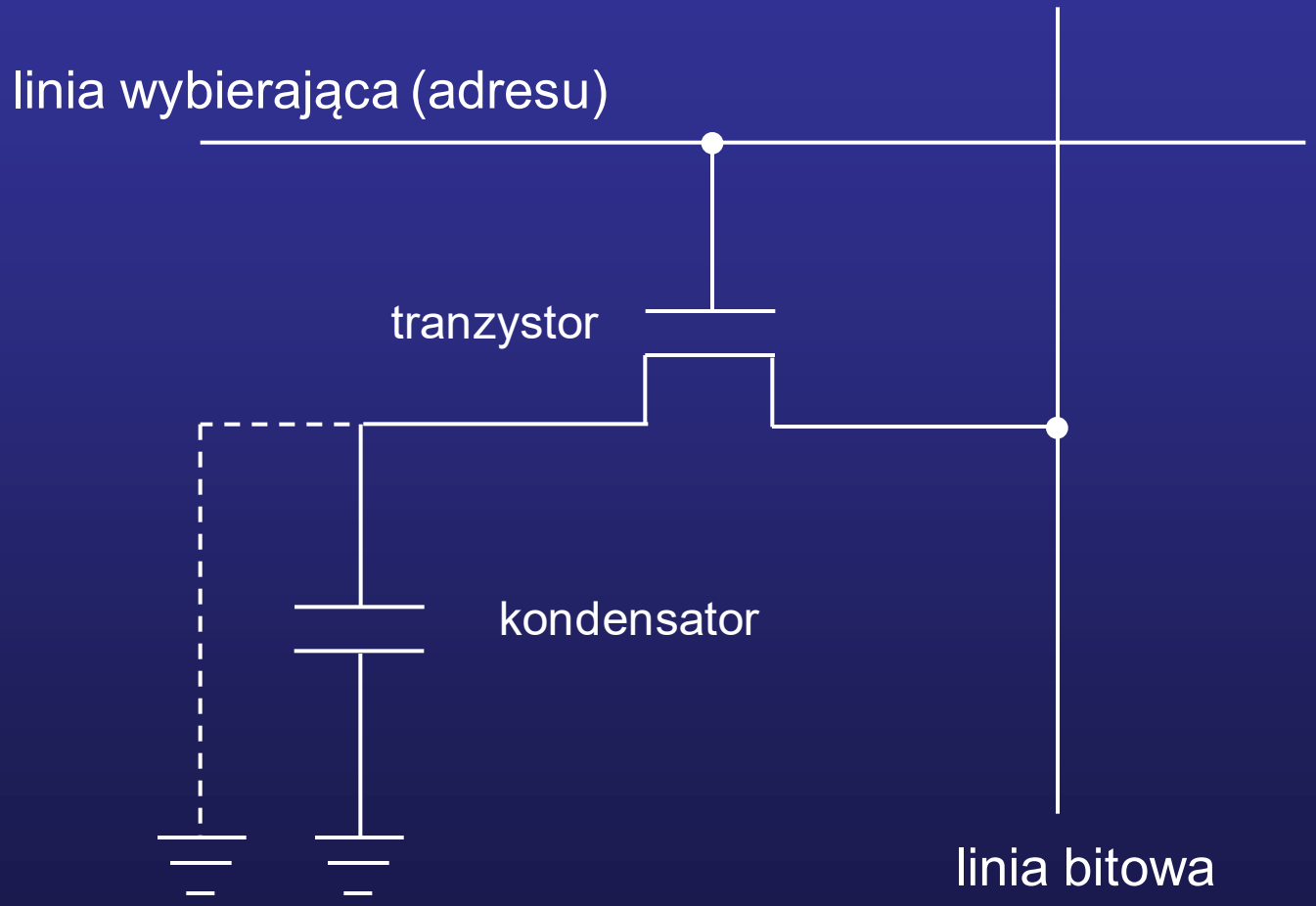
Organizacja układu DRAM



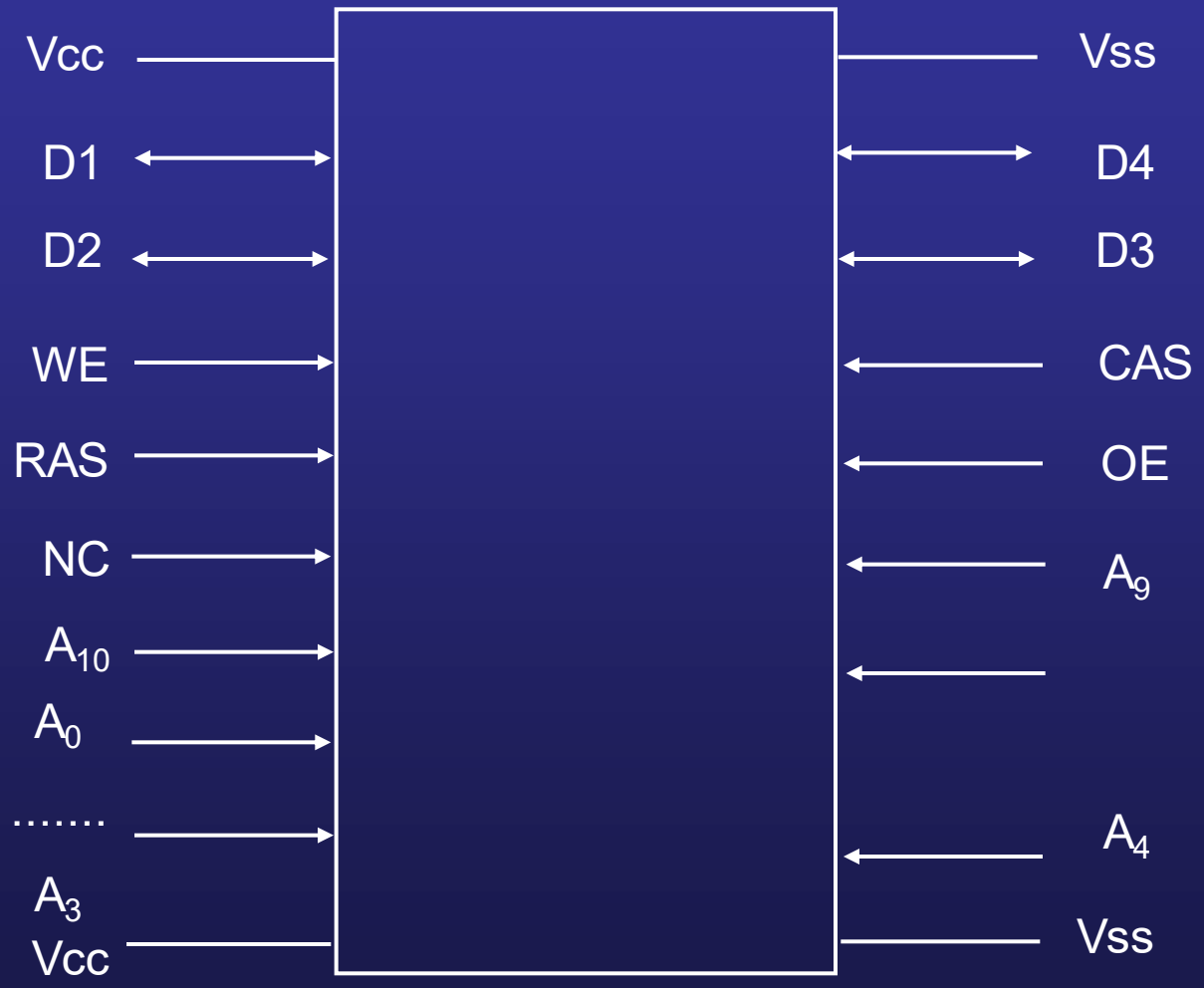
Przykład organizacji układu DRAM

- Pamięć 16 Mb
- organizacja logiczna: 4M $\times$ 4b (4 układy o rozmiarach 2048 $\times$ 2048)
- do zaadresowania każdego wiersza w układzie potrzeba 11 linii, podobnie jak dla kolumny

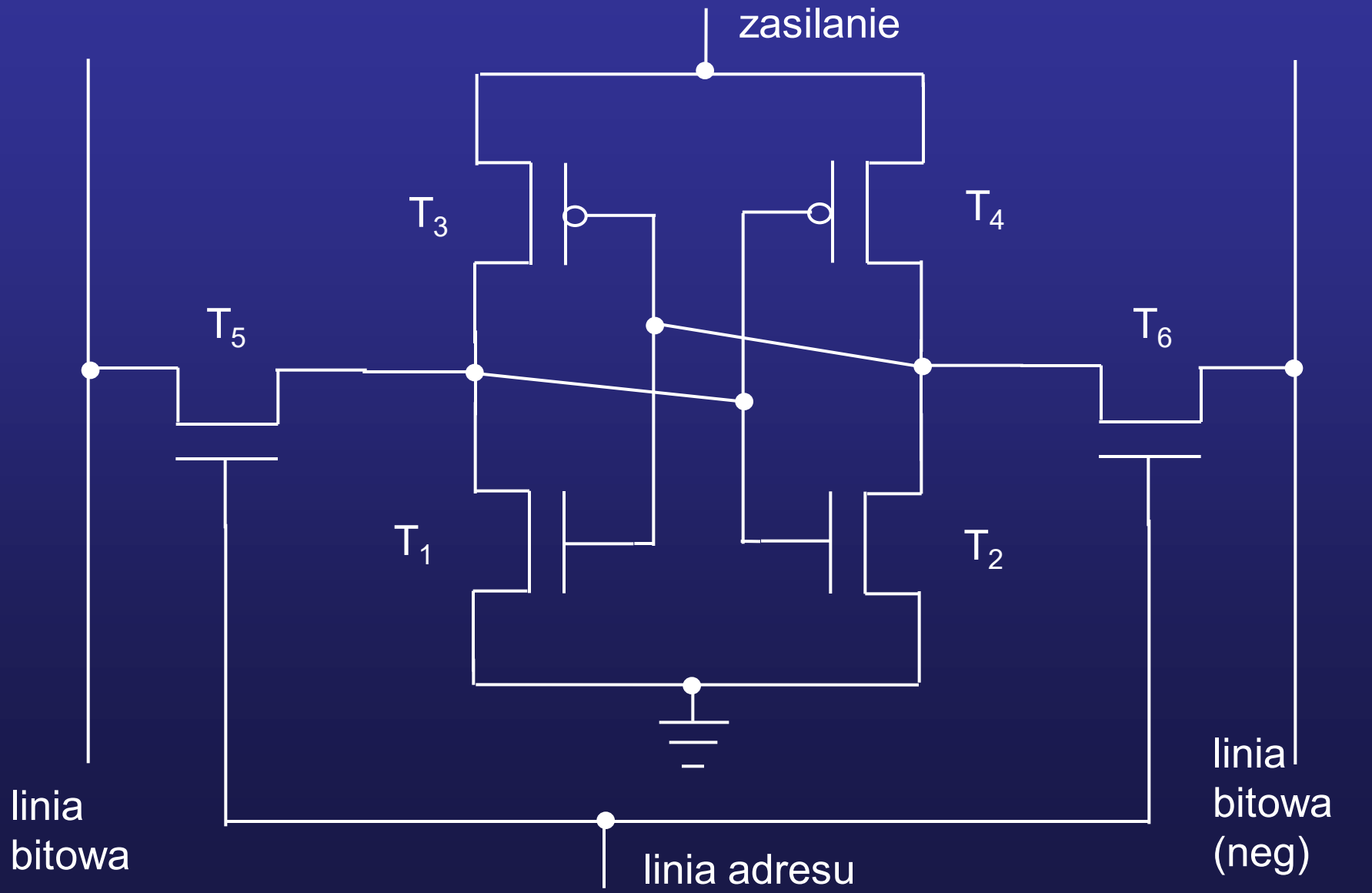
Schemat komórki DRAM



Przykład obudowy DRAM



Schemat komórki SRAM



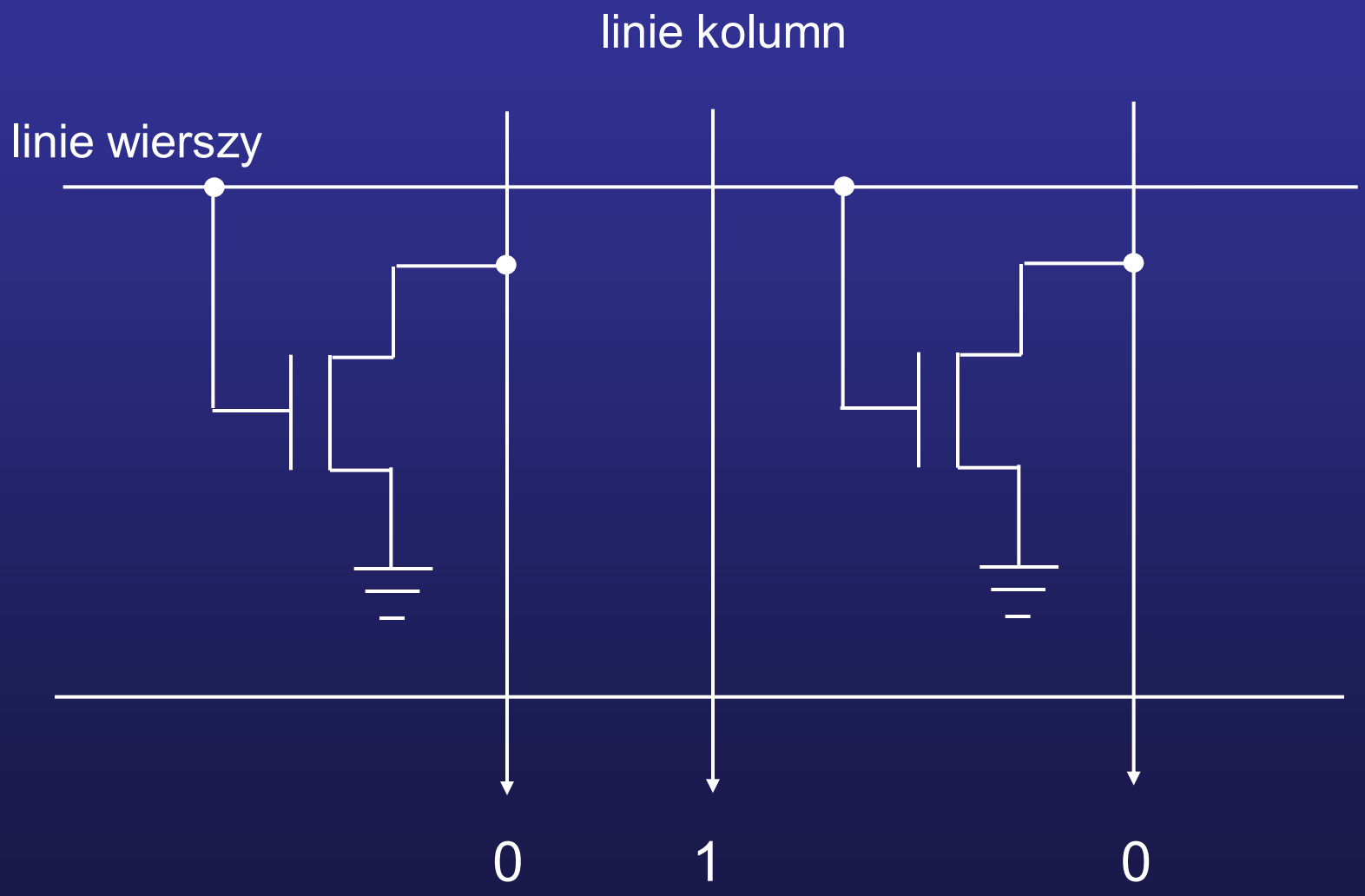
Pamięci ROM

- Kosztowne wprowadzanie programu do pamięci, produkcja opłacalna tylko w dużych ilościach
- Zastosowania: programy systemowe (np. BIOS), tablice funkcji, podprogramy biblioteczne

Programowanie pamięci ROM

- jednokrotne, po wyprodukowaniu układu (PROM)
- pamięci głównie do odczytu (read-mostly memory):
  - EPROM – przed zapisem cała zawartość jest kasowana
  - EEPROM – przed zapisem wymazywane tylko żądane bajty
  - flash – przed zapisem wymazywane bloki bajtów

Schemat komórki ROM



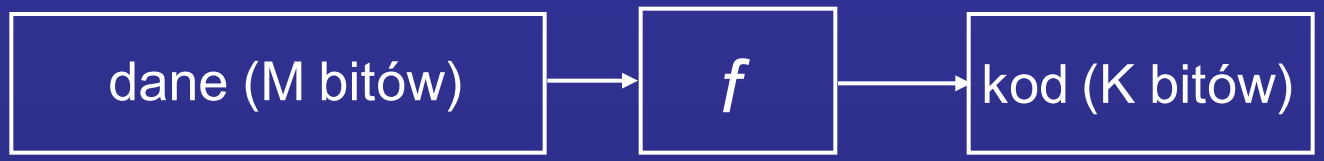
Pamięć typu flash

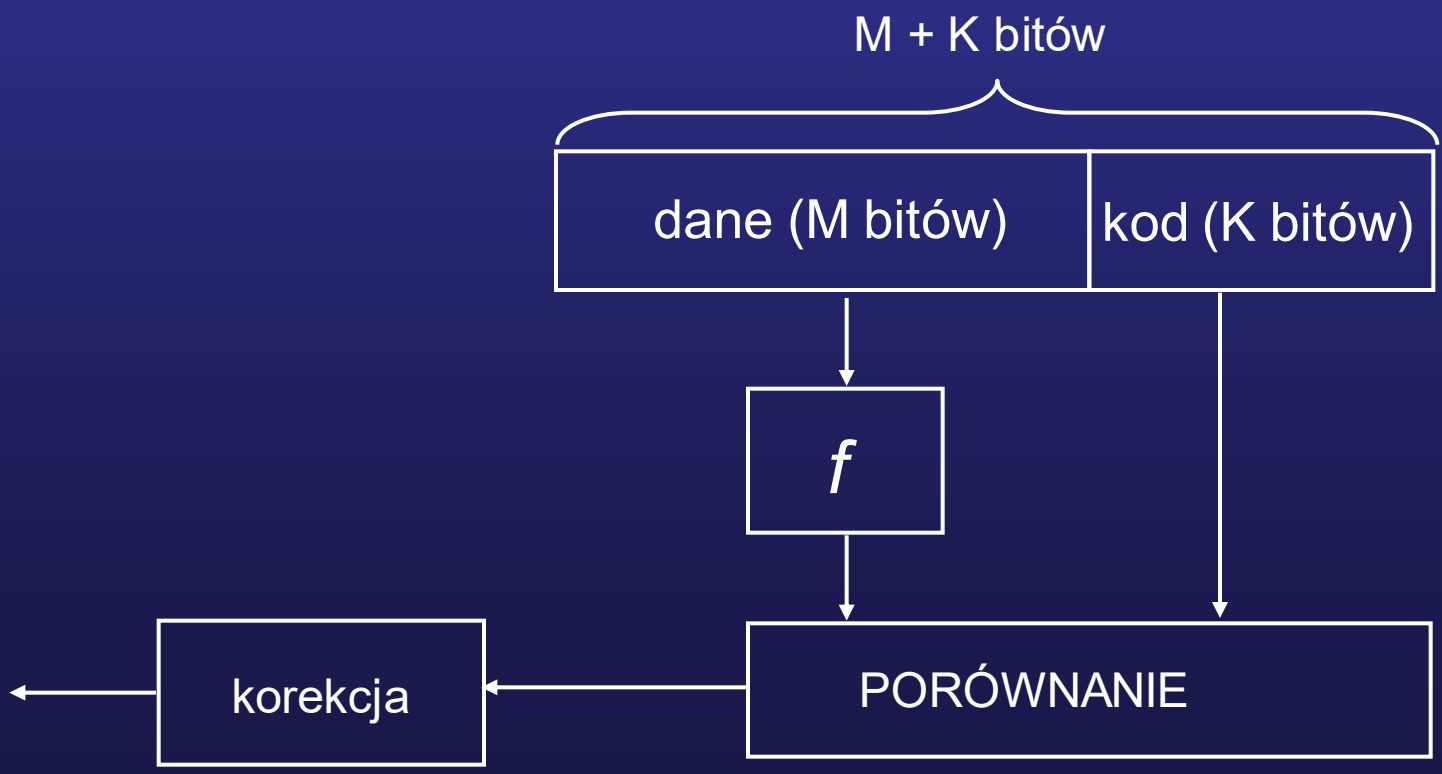
- Jest to odmiana pamięci EEPROM, możliwe jest zapisywanie i wymazywanie wielu komórek jednocześnie
- Mniejszy czas dostępu od dysków twardych (rzędu od 10 ms do 70 ns)
- Niskie napięcie zasilania (nawet 1,8 V)
- Pamięć ulega zużyciu! (AMD gwarantuje wytrzymałość miliona cykli zapisu na sektor, inni producenci nawet 10 mln)

Korekcja błędów

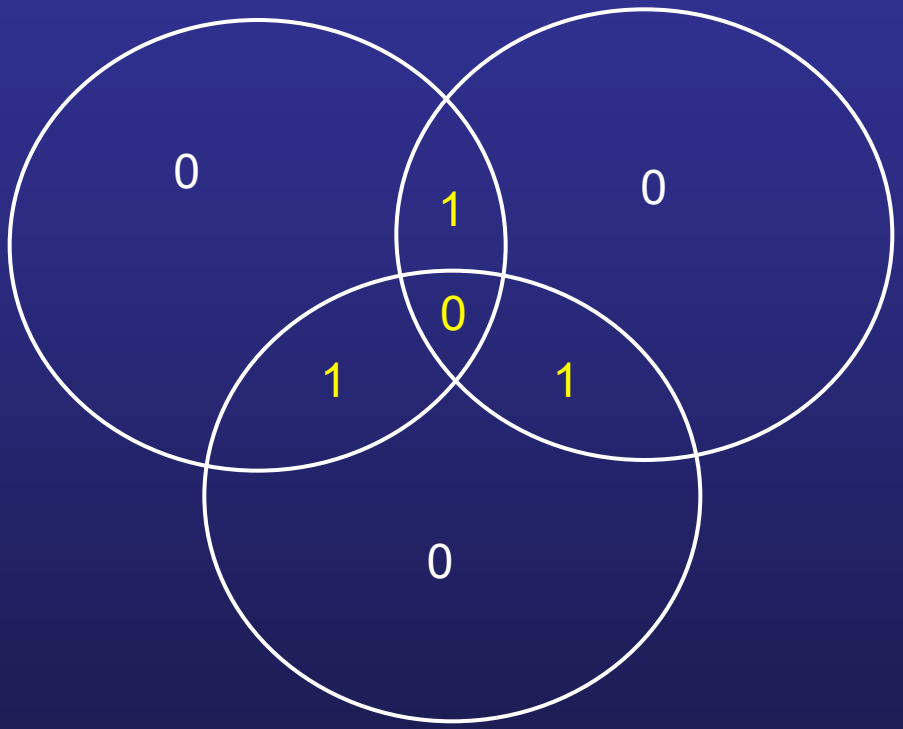
- Podczas pracy pamięci mogą wystąpić błędy:
  - stałe – uszkodzenie sprzętowe komórki
  - przypadkowe (losowe)
- Konieczne jest stworzenie kodu na podstawie zawartości słowa, którego wartość wskazywałaby wystąpienie błędu

Działanie korekcji błędów





Kod korekcyjny Hamminga



bity parzystości

bity słowa

słowo 4-bitowe

Przykład kodu korekcyjnego – słowo 8-bitowe

- Wynik porównania kodów nazywany jest słowem-syndromem
- Zero w syndromie oznacza brak błędu na tej pozycji
- Liczba bitów kodu określona jest jako:

$$2^k - 1 \geq M + K$$

Przykład kodu korekcyjnego – słowo 8-bitowe (c.d.)

Pozycja bitowa	12	11	10	9	8	7	6	5	4	3	2	1
Numer pozycji	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Bit danych	8	7	6	5		4	3	2		1		
Bit kontrolny					4				3		2	1

$C1 = D1 \oplus D2 \oplus D4 \oplus D5 \oplus D7$

$C2 = D1 \oplus D3 \oplus D4 \oplus D6 \oplus D7$

$C3 = D2 \oplus D3 \oplus D4 \oplus D8$

$C4 = D5 \oplus D6 \oplus D7 \oplus D8$

Kody korekcyjne a długość słowa

Długość słowa	Korekcja 1 błędu	Wykrycie 2 błędów
8	4	5
16	5	6
32	6	7
64	7	8
128	8	9
256	9	10

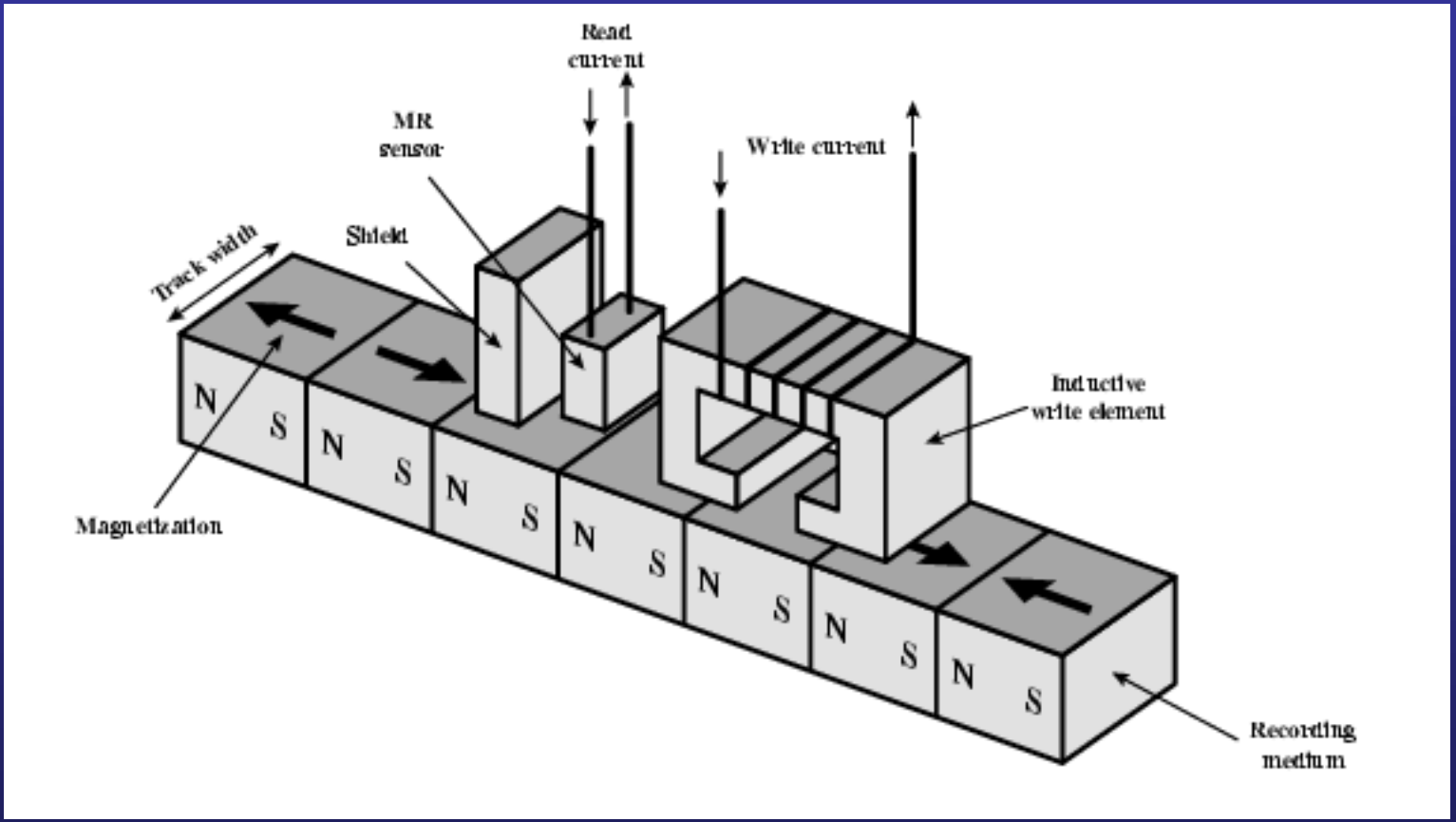
Przykłady pamięci RAM

- FP RAM (Fast Page RAM)
- EDO RAM (Extended Data Output RAM)
- SDRAM (Synchronous DRAM)
- DDR DRAM (Double Data Rate DRAM)
- DDR2 DRAM
- DDR3 DRAM
- DDR4 DRAM
- DDR5 DRAM
- RDRAM (Rambus DRAM)
- CDRAM (Cache DRAM)

Rodzaje pamięci zewnętrznych

- dyski magnetyczne – dyski twarde, dyskietki
  - IDE (PATA)
  - SATA (Serial ATA)
- dyski magnetyczne – zapis taśmowy
- napędy półprzewodnikowe
  - Pamięci Flash
  - Dyski SSD
- dyski optyczne
  - CD-ROM
  - DVD-ROM
  - CD-R, CD-RW
  - DVD-R, DVD-RW
  - Blu Ray

Dysk magnetyczny



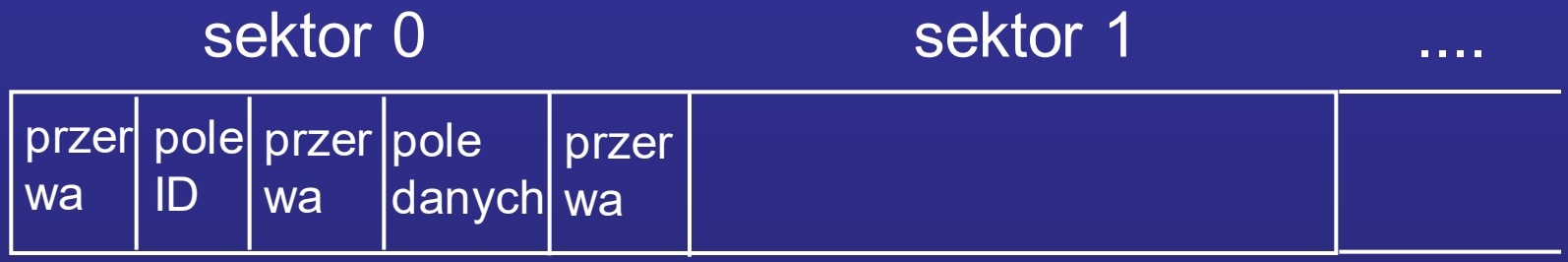
- głowica porusza się (jeśli w ogóle) wzdłuż promienia dysku
- Zapis zerojedynkowy możliwy jest dzięki zmianie kierunku prądu płynącego przez cewkę
- Odczyt wykorzystuje zjawisko indukcji

Rozkład danych na dysku

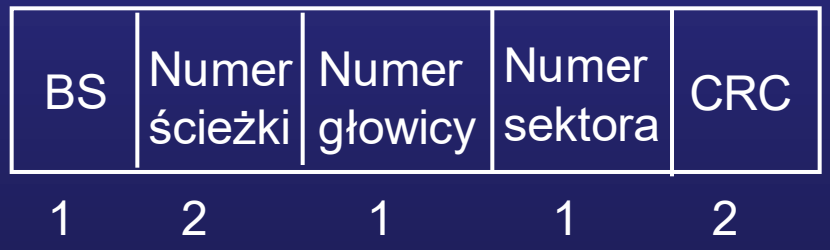
Gęstość zapisu danych na dysku

- stała prędkość kątowa (CAV)
  - sektor znajdujący się najbliżej środka zawiera tyle samo danych, co sektor najbardziej odległy
  - łatwy dostęp do poszczególnych fragmentów dysku
- zapis wielostrefowy
  - dysk podzielony jest na strefy
  - w obrębie strefy gęstość zapisu jest stała
  - trudniejsze adresowanie

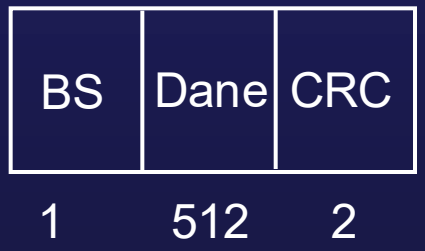
Format ścieżki dysku twardego



Pole ID:

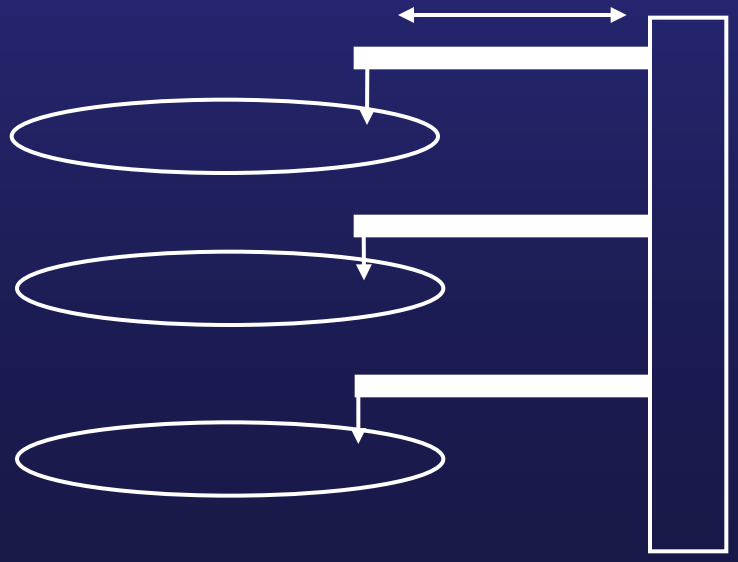


Pole danych:



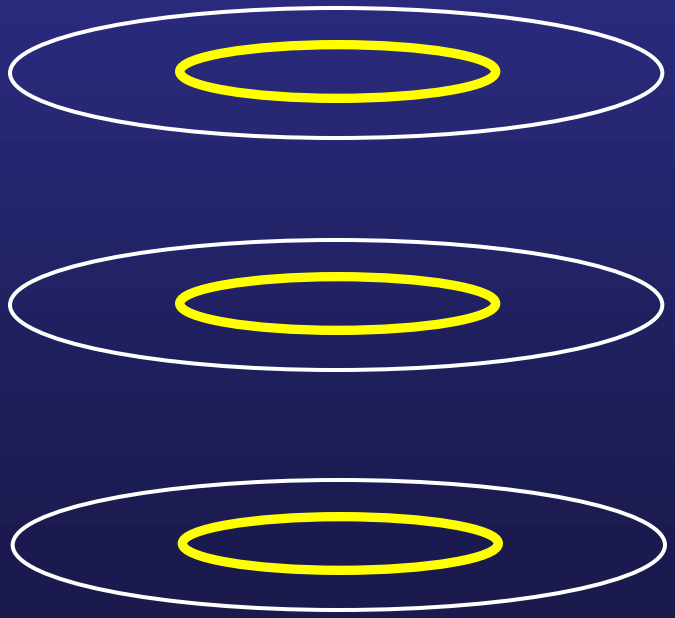
Dyski wielopłytowe

- istnieje wiele płyt (talerzy) umieszczonych na współosiowym walcu
- nad każdym talerzem znajduje się głowica
- wszystkie głowice poruszają się w ten sam sposób



Cylindry

- Są to wszystkie ścieżki znajdujące się w tym samym miejscu na każdej płycie



Parametry dysków twardych

Parametr	Maxtor Diamond Max 10	WD Caviar WD7500 AADS	Seagate Cheetah X15-36LP	Toshiba HDD1242	IBM Micro drive
Pojemność [GB]	200	750	36,7	5	1
Prędkość obr/min	7200	7200	15000	4200	3600
śr. czas przeszukiwania	9,4 ms	8,9 ms	3,6 ms	15 ms	12 ms
śr. opóźnienie obrotowe [ms]	4,2	4,2	2	7,14	8,33
transfer [MB/s]	133	300	520-710	66	13,3
bajtów na sektor	512	512	512	512	512

- czas przeszukiwania – czas pozycjonowania głowicy
- opóźnienie obrotowe – czas dotarcia do żądanego sektora
- czas dostępu – suma powyższych
- czas transferu – czas potrzebny na przesłanie odczytanych danych

Czas przeszukiwania

Czas przeszukiwania

- składa się z czasu rozruchu oraz czasu przejścia przez ścieżki pośrednie
- zależy od rozmiarów nośnika (obecnie typowa średnica wynosi 9 cm)

Opóźnienie obrotowe

- zależy od liczby obrotów na minutę (od 3600 do 15000 obr/min dla HDD i 600 obr/min dla FDD)

Czas transferu

$$T_a = T_s + \frac{1}{2r} + \frac{b}{rN}$$

- T_s – średni czas przeszukiwania
- r – prędkość obrotowa
- b – liczba przesyłanych bajtów
- N – liczba bajtów na ścieżce

Problem rozmieszczenia danych na dysku

Sekwencyjna organizacja danych

T_s = 4 ms

T_r = 4 ms

T_o = 8 ms (500 sec.)

Razem:

16 ms

T_t = 16 ms + 4 × 12 ms = 64 ms

Rozmieszczenie przypadkowe

T_s = 4 ms

T_r = 4 ms

T_o = 0,016 ms (1 sec.)

Razem:

8,016 ms

T_t = 2500 × 8,016 ms = 20,04 s

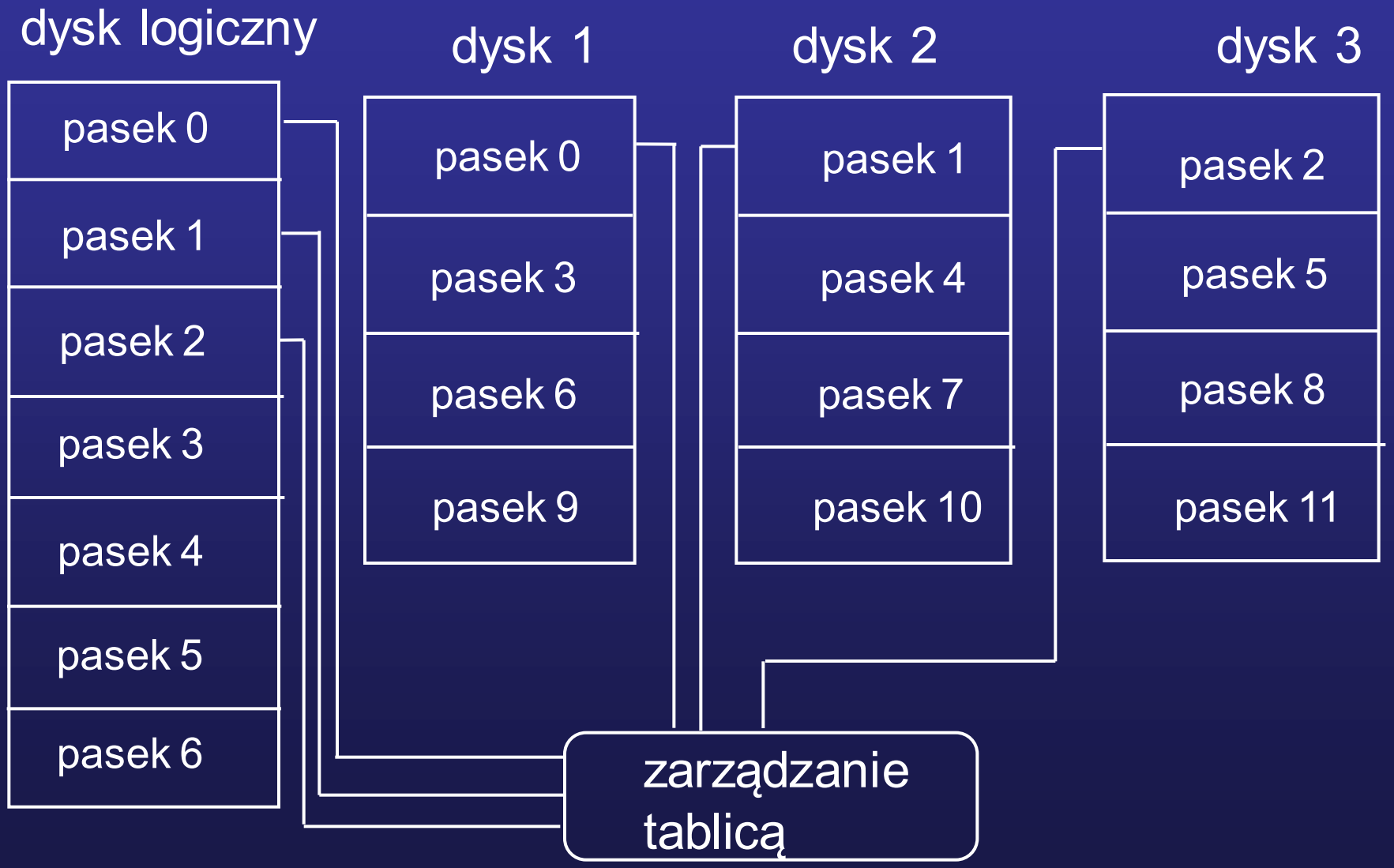
RAID

- technika wykorzystania wielu dysków jednocześnie do odczytu równoległego w celu poprawy efektywności pamięci zewnętrznej
- zaproponowano siedem poziomów, różniących się sposobem wykorzystania dysków

RAID 0

- brak redundancji danych
- szybkość transferu optymalna dla małych pasków
- zastosowanie: systemy o wysokiej wydajności i niekrytycznych danych

Schemat RAID 0



RAID 1

- redundancja polega na lustrzanym odwzorowaniu danych pomiędzy dyskami
- szybkość transferu danych zależy od najwolniejszego napędu
- zastosowanie: serwery, przechowywanie i przetwarzanie krytycznych plików
- niskie ryzyko utraty danych
- główna wada: wysoki koszt

Schemat RAID 1

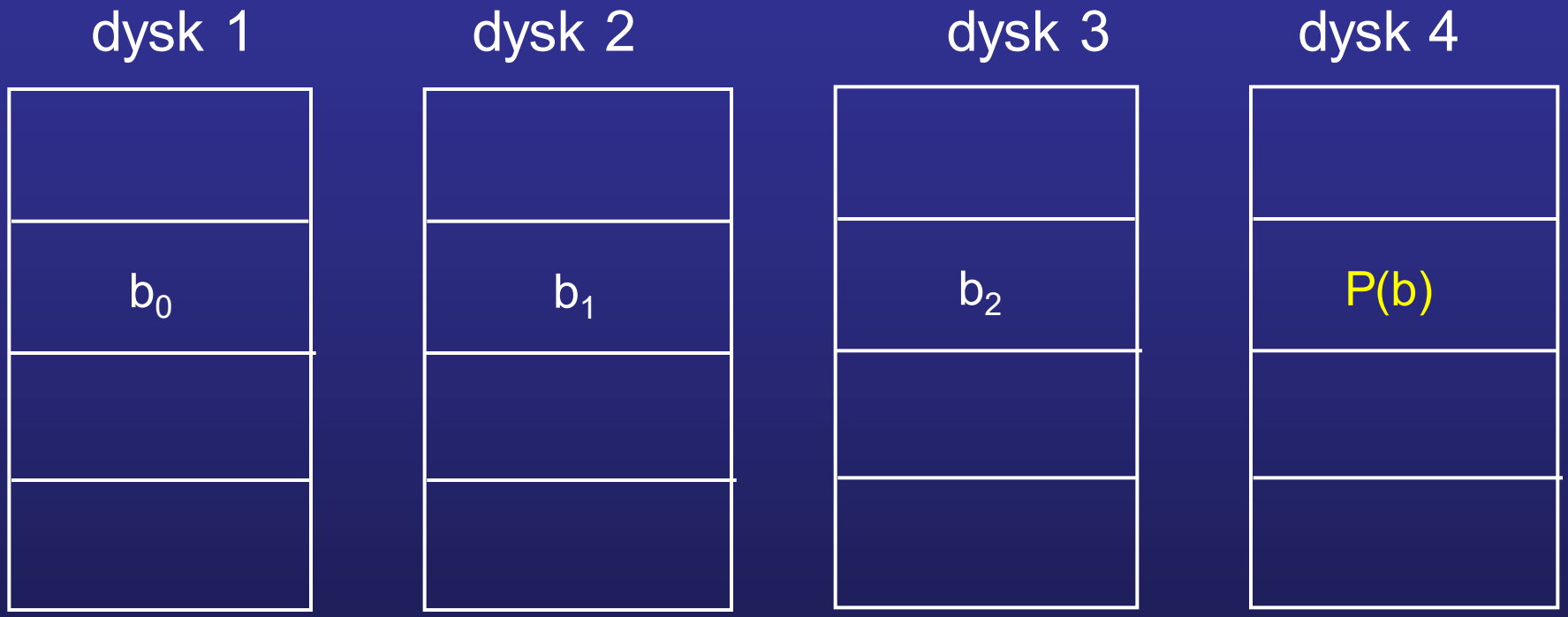
RAID 2 i 3

- obie techniki wykorzystują metodę dostępu równoległego (synchronizacja głowic)
- paski mają bardzo mały rozmiar (wielkości bajtu lub słowa)
- w RAID 2 liczba dysków nadmiarowych = log(liczba dysków danych)
- RAID 2 nadaje się tylko wtedy, gdy występuje dużo błędów dyskowych

RAID 3

- tylko jeden dysk nadmiarowy
- korekcja błędów wykorzystuje bit parzystości dla grupy bitów zajmujących tą samą pozycję na wszystkich dyskach
- po wystąpieniu uszkodzenia informacja odtwarzana jest z pozostałych dysków
- duże szybkości transferu

Schemat RAID 3



$$P(b) = b_0 \oplus b_1 \oplus b_2$$

RAID 4, 5 i 6

- wykorzystywana jest tu metoda dostępu niezależnego (każdy dysk działa oddzielnie)
- paski są dużych rozmiarów
- informacje korekcji błędów są obliczane na blokach danych
- w RAID 4 istnieje dysk nadmiarowy przechowujący informacje korekcji błędów

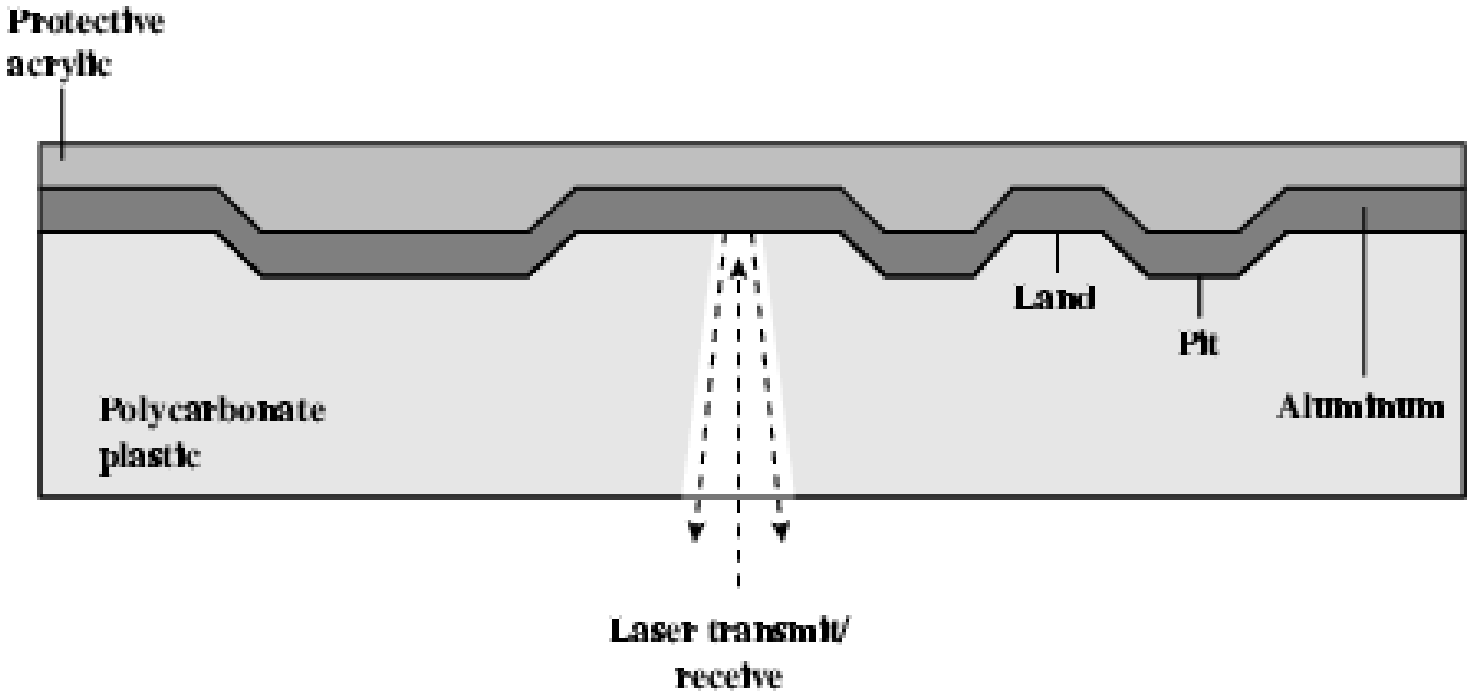
Pamięć optyczna

- CD – pierwotnie przeznaczony do zastosowań audio
- CD-ROM – fizycznie odpowiadający dyskom CD, przeznaczony jako nośnik danych komputerowych
- CD-R – dysk kompaktowy do zapisu jednorazowego
- CD-RW – dysk kompaktowy wielokrotnego zapisu
- DVD – dysk optyczny przeznaczony głównie do zastosowań audio i video (jedno- lub dwustronny)
- DVD-R – dysk DVD zapisywalny jednorazowo
- DVD-RW – dysk DVD wielokrotnego zapisu
- Blu Ray (zwycięzca rywalizacji z HD-DVD)

Dyski kompaktowe

- różnica pomiędzy CD a CD-ROM leży w korekcji błędów
- tłoczenie dysku CD wymaga użycia lasera o dużej mocy, który tworzy zagłębienia w powierzchni odbijającej światło
- zapisana informacja jest przykrywana warstwą srebra lub złota oraz lakierem
- dane na dysku kompaktowym są ułożone na jednej spiralnej ścieżce idącej od środka do brzegu płyty

Schemat dysku CD



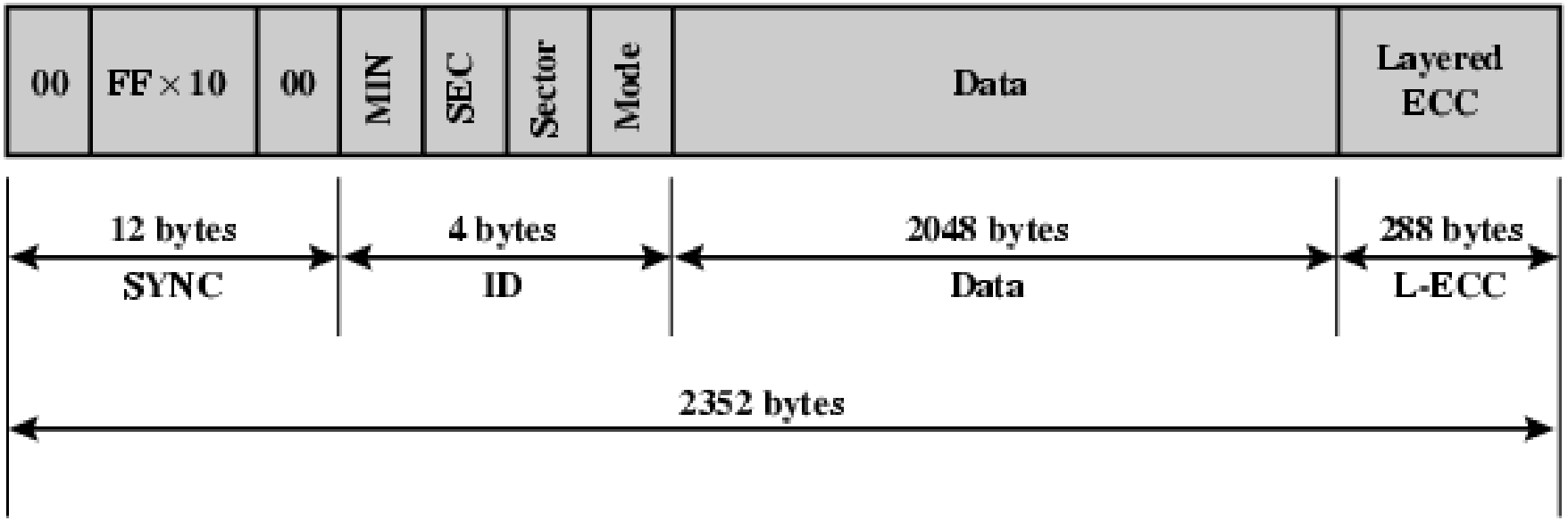
Dyski kompaktowe (c.d.)

- Odczyt danych z dysku możliwy jest z użyciem lasera średniej mocy
- prędkość odczytu wzrosła z pierwotnych 150 KB/s (standard audio) do 8,4 MB/s (odtwarzacze x56)
- odstęp między zwojami spirali wynosi 1,6 μm
- minimalna odległość między wgłębieniami na spirali wynosi 0,824 μm

Dyski kompaktowe (c.d.)

- Dysk CD Audio jest odtwarzany z prędkością x1
  - odczyt lasera odbywa się ze stałą prędkością liniową
  - Spirala ścieżki ma długość ok. 5.3 km
  - Maksymalny czas trwania utworu to 4391 s = 73.2 min
- Dane na CD-ROMie zorganizowane są w postaci bloków
- Czas dostępu duży – nawet do 0,5 s!

Format bloku CD-ROM



- SYNC – synchronizacja (identyfikacja początku bloku)
- ID – nagłówek
- ECC – kod korekcyjny

Zapisywalne dyski kompaktowe

- zapis na dysku CD-R jest możliwy przy użyciu lasera średniej mocy, który naprowadzany jest na warstwę barwnika
- Rozszerzenia standardu CD-R pozwoliły uzyskać większą pojemność – od 650 MB do 870 MB
- zapis na dysku CD-RW jest możliwy dzięki zjawisku przemiany fazowej

Kolorowe księgi

- czerwona księga – właściwości fizyczne płyty CD-DA, sposób cyfrowego kodowania dźwięku
- żółta księga – właściwości płyt CD-ROM
- CD-ROM XA – rozszerzenie żółtej księgi, opisujące m.in. format VideoCD i Playstation
- zielona księga - opisuje format CD-I
- pomarańczowa księga – zawiera specyfikacje zapisywalnych płyt CD (CD-R, CD-MO, CD-RW)
- biała księga – opisuje specyfikację VideoCD z rozszerzeniami (Karaoke CD, VCD, SVCD)
- niebieska księga – definiuje format Enhanced Music CD (CD Extra)
- CD Graphics – rozszerzenie czerwonej księgi opisujące obsługę danych graficznych i tekstu

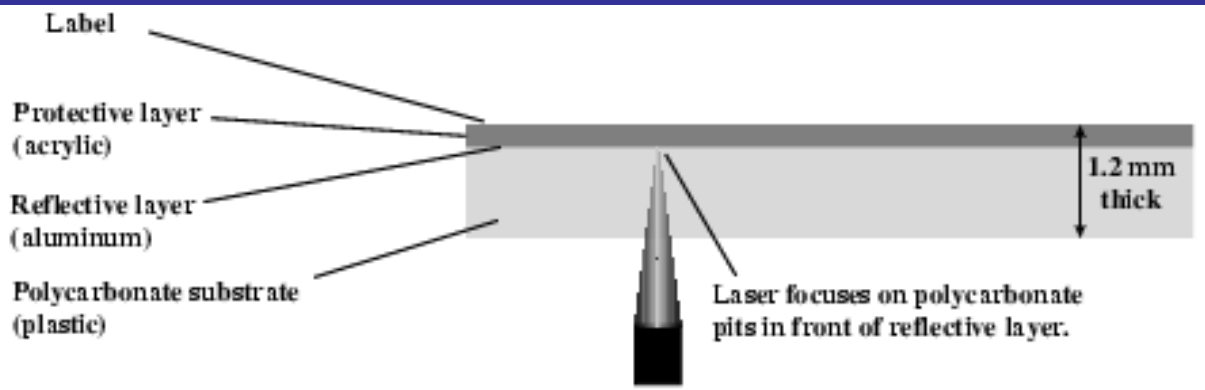
Dyski DVD

- DVD – Digital Versatile Disc
- odczyt przy pomocy czerwonego lasera (długość fali 650 nm)
- maksymalna pojemność dysku dwustronnego i dwuwarstwowego wynosi 17 GB
- w stosunku do standardu CD-ROM, DVD oferuje większe upakowanie danych – odstęp między zwojami spirali wynosi 0,74 μm; odstęp między wgłębieniami 0,4 μm
- płyty DVD mogą mieć drugą warstwę odbijającą światło pod pierwszą, co pozwala niemal podwoić pojemność płyty

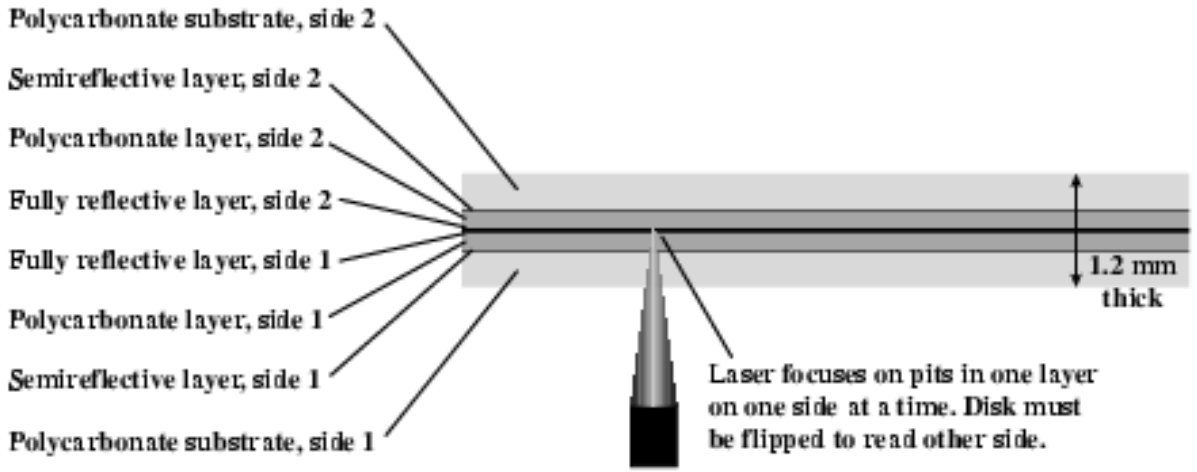
Zapisywalne dyski DVD

- DVD-R – jednokrotnego zapisu, jednostronne i jednowarstwowe, pojemność 4,7 GB
- DVD-RW – wielokrotnego zapisu, również jednostronne i jednowarstwowe
- Wiele standardów zapisu (R+/-, RW+/-, DVD-RAM), problem z uzyskaniem jednorodności standardów

Odczyt z płyty DVD



(a) CD-ROM - Capacity 682 MB



(b) DVD-ROM, double-sided, dual-layer - Capacity 17 GB

Blu Ray

- nowy standard firmy Sony w technologiach audio-video (jako BD)
- użycie błękitnego lasera (długość fali 405 nm) pozwala na płycie o standardowych rozmiarach zapisać 50 GB danych
- zastosowanie: filmy w wysokiej rozdzielczości (HD)

Organizacja i Architektura Komputerów

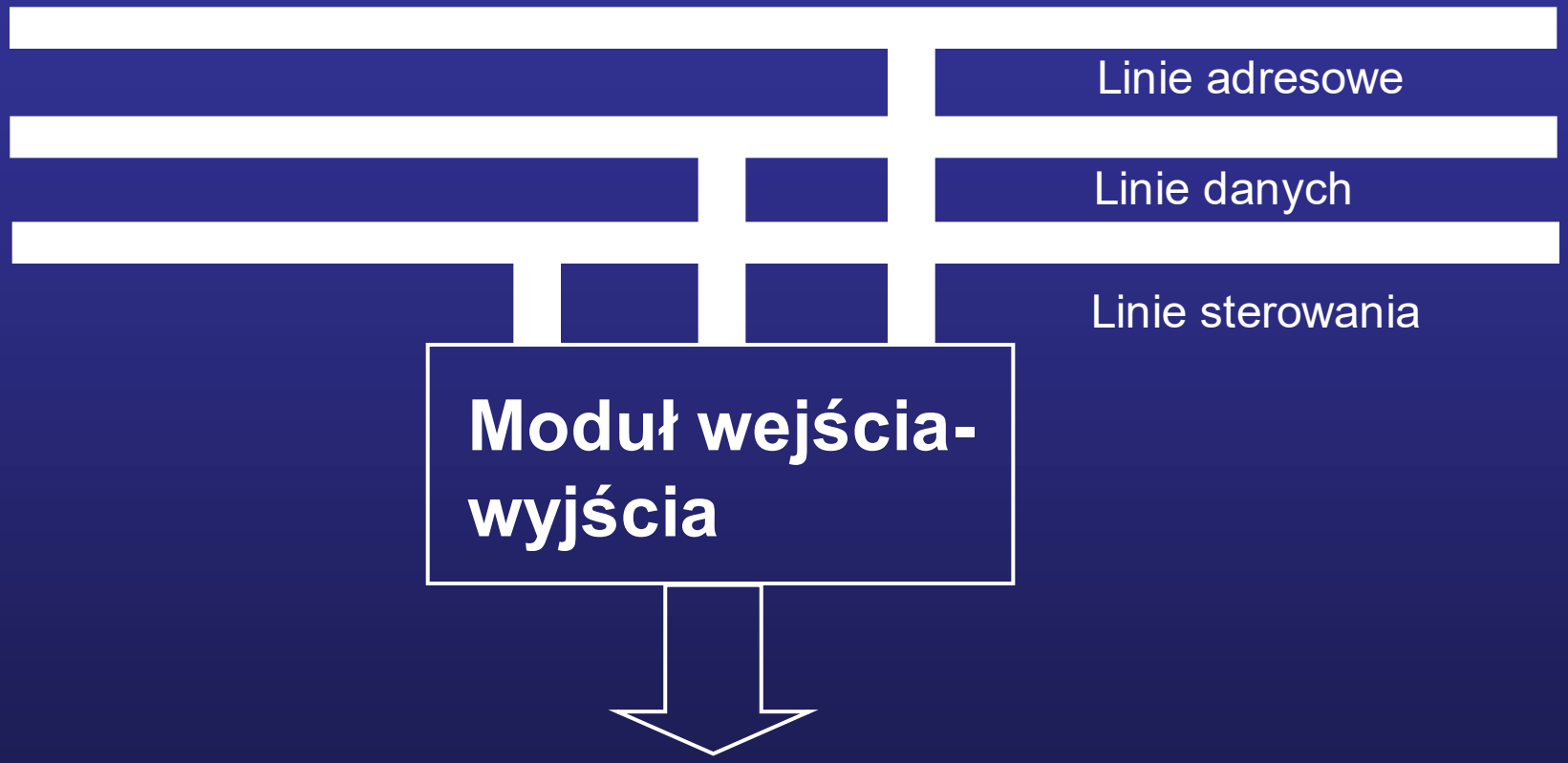
Wykład nr 10: Urządzenia wejścia/wyjścia

Piotr Bilski

Komunikacja z urządzeniami we/wy

- Urządzenie połączone jest z magistrala systemową poprzez moduł we/wy
- Prędkości urządzeń i magistrali znacząco się różnią – pośrednik jest potrzebny!
- Formaty danych obsługiwane przez komputer są różne od formatów wspieranych przez urządzenie

Schemat modułu we/wy

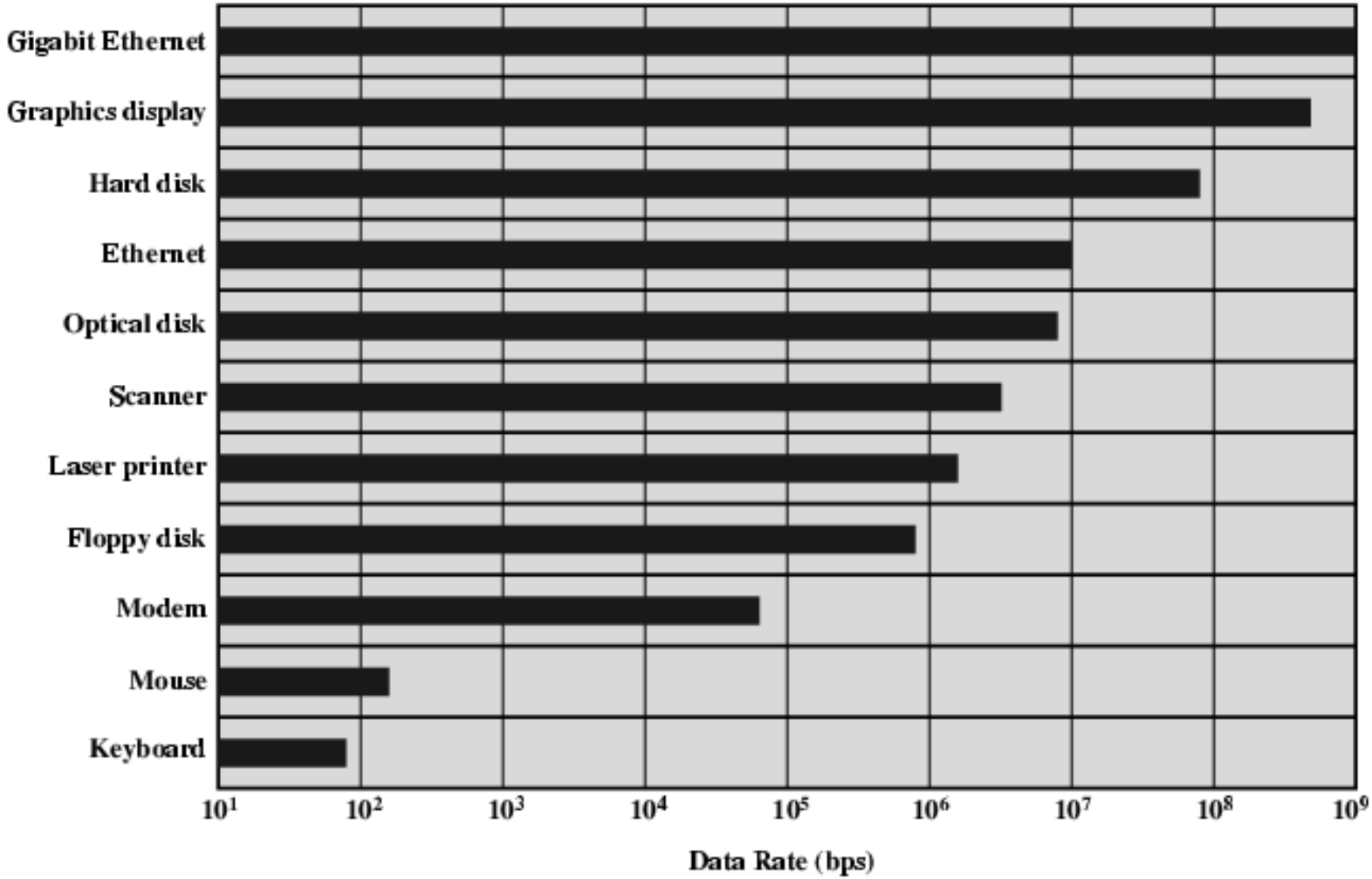


Urządzenia zewnętrzne

- do komunikacji z człowiekiem (np. monitor, drukarka)
- do komunikacji z maszyną (np. napęd CD)
- do komunikacji zdalnej (np. modem, karta sieciowa)

Schemat urządzenia peryferyjnego

Prędkości urządzeń peryferyjnych



Moduł wejścia-wyjścia

Realizowane funkcje:

- sterowanie i taktowanie
- komunikacja z procesorem
- komunikacja z urządzeniem
- buforowanie danych
- wykrywanie błędów

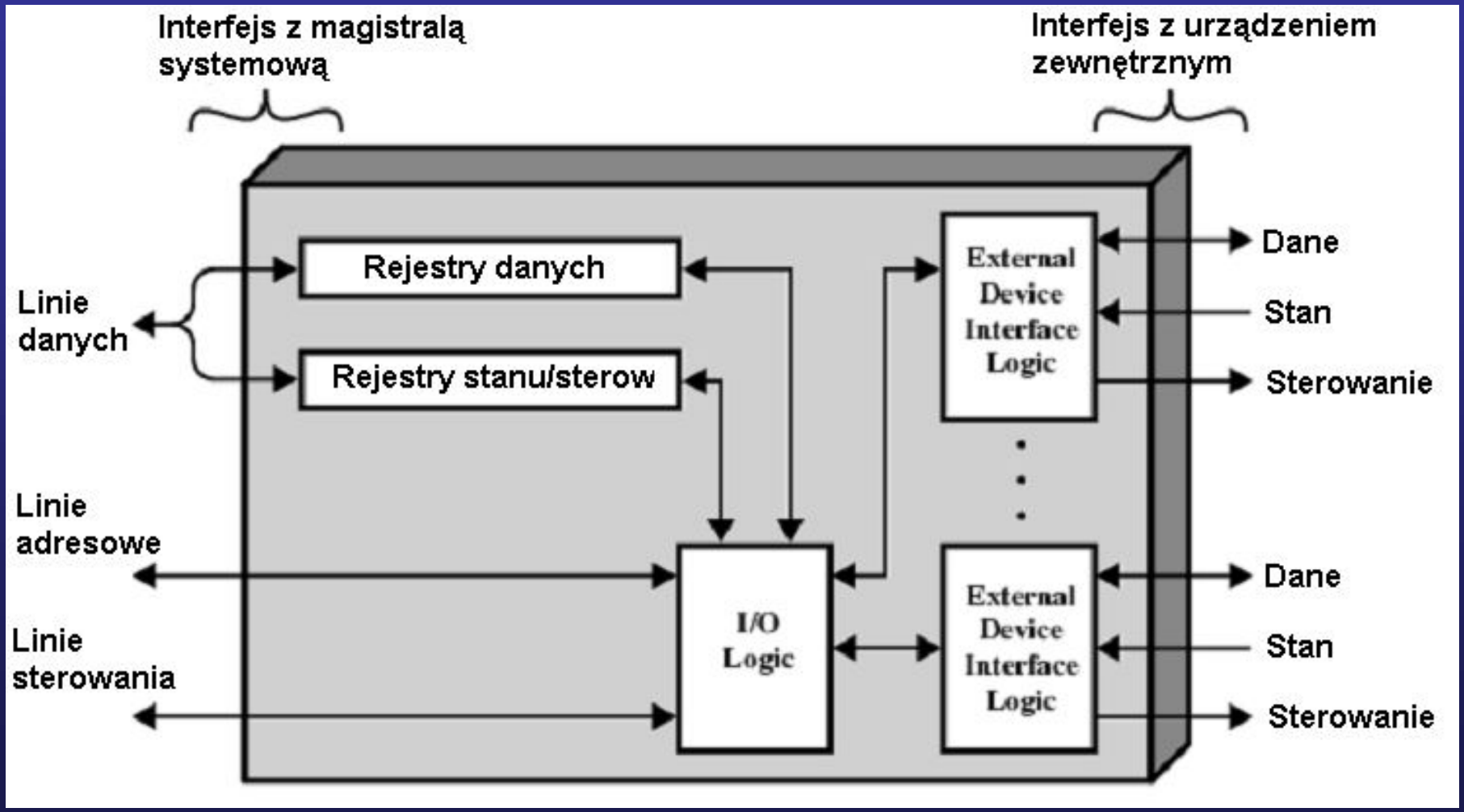
Komunikacja modułu z procesorem

- dekodowanie rozkazu
- przesyłanie danych
- przesyłanie informacji o stanie
- rozpoznawanie adresu

Przykład współpracy procesora i urządzenia peryferyjnego

- żądanie ze strony procesora stanu urządzenia
- przesłanie informacji o stanie urządzenia przez moduł we/wy
- jeśli urządzenie działa, procesor zgłasza żądanie przesłania danych
- moduł otrzymuje jednostkę danych z urządzenia zewnętrznego i przesyła do procesora

Schemat modułu wejścia-wyjścia



Rodzaje modułów we/wy

- kanał wejścia-wyjścia – bardziej złożony, wykonuje większość pracy związanej z komunikacją i obsługą urządzenia (komputery mainframe)
- sterownik urządzenia – prosty, większość funkcji musi wykonywać procesor (komputery osobiste)

Techniki wejścia-wyjścia

- programowane wejście-wyjście – duże zaangażowanie procesora
- wejście-wyjście sterowane przerwaniami – procesor odciążony
- bezpośredni dostęp do pamięci – procesor niepotrzebny

Programowane wejście-wyjście

- Procesor przesyła rozkazy wejścia-wyjścia do modułu. Sam musi pilnować, kiedy dane są dostępne
- Instrukcje pobierane z pamięci mają swoje odpowiedniki w rozkazach wysyłanych modułowi
- Wada: procesor bezczynny przez większość czasu

Rozkazy wejścia-wyjścia

- sterowania (charakterystyczne dla poszczególnych urządzeń)
- testowania (stan urządzenia i wynik ostatniej operacji)
- odczytu
- zapisu

Schemat programowanego wejścia-wyjścia

Instrukcje wejścia-wyjścia

- Problem adresowania urządzeń we/wy:
  a) odwzorowanie w pamięci
  b) izolowane wejście-wyjście

200 Load AC	1
201 Store AC	517
202 Load AC	517
203 Branch if St=0	202
204 Load AC	516

(a)

200 Start I/O	5
201 Test I/O	5
203 Branch if St=0	201
204 In	5

(b)

Wejście-wyjście sterowane przerwaniami

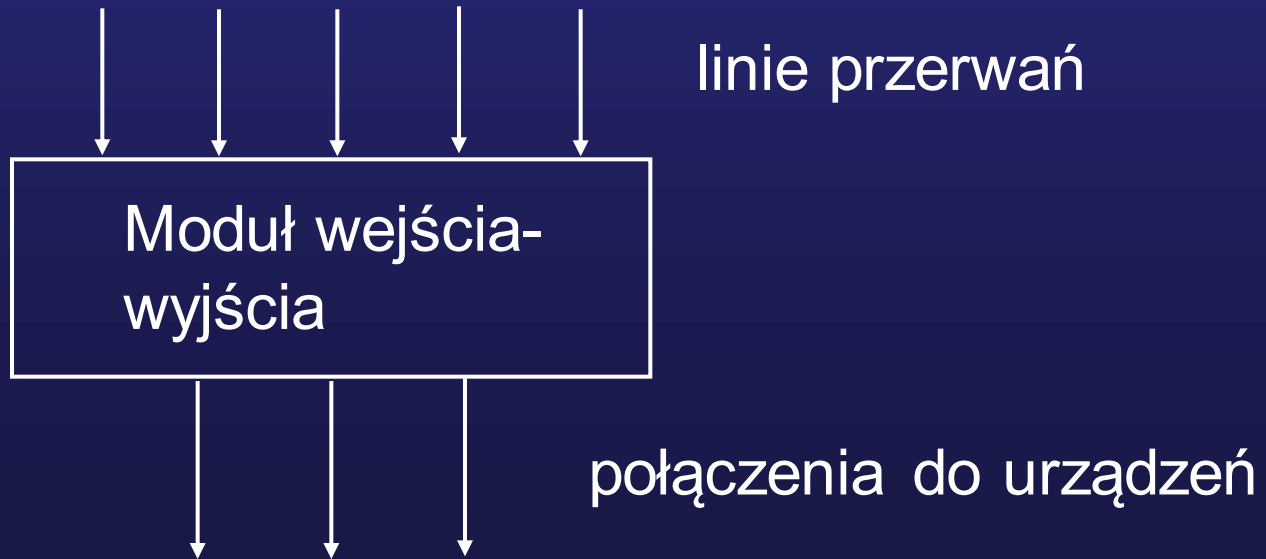
- procesor wysyła żądania do modułu I/O, ale nie czeka na odpowiedź
- gdy moduł jest gotowy do komunikacji z procesorem, zgłasza sygnał przerwania
- procesor w każdym cyklu rozkazowym sprawdza, czy zgłoszono przerwanie
- wada: procesor pośredniczy w komunikacji między peryferiami i pamięcią

Schemat obsługi przerwań

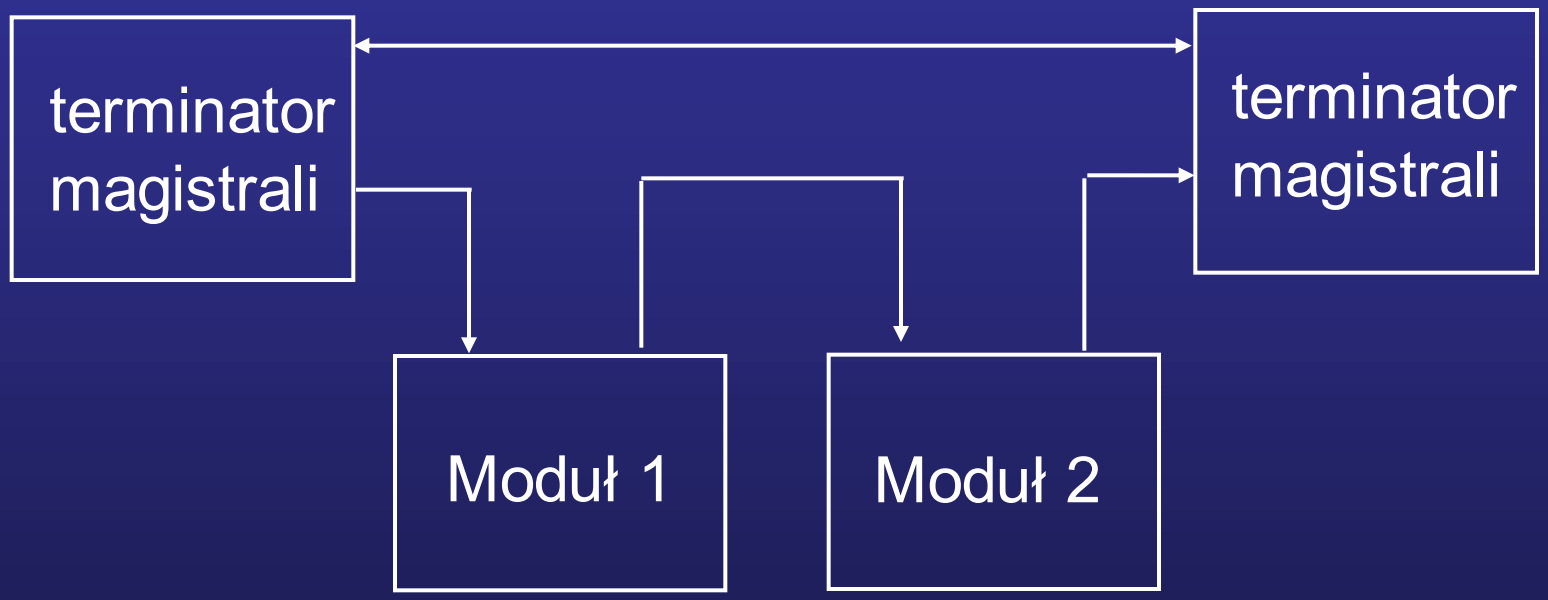
Szczegóły przetwarzania przerwania

Problem obsługi wielu urządzeń

- wiele linii przerwań
- odpytywanie za pomocą oprogramowania
- odpytywanie za pomocą sprzętu
- arbitraż za pomocą magistrali (PCI, SCSI)

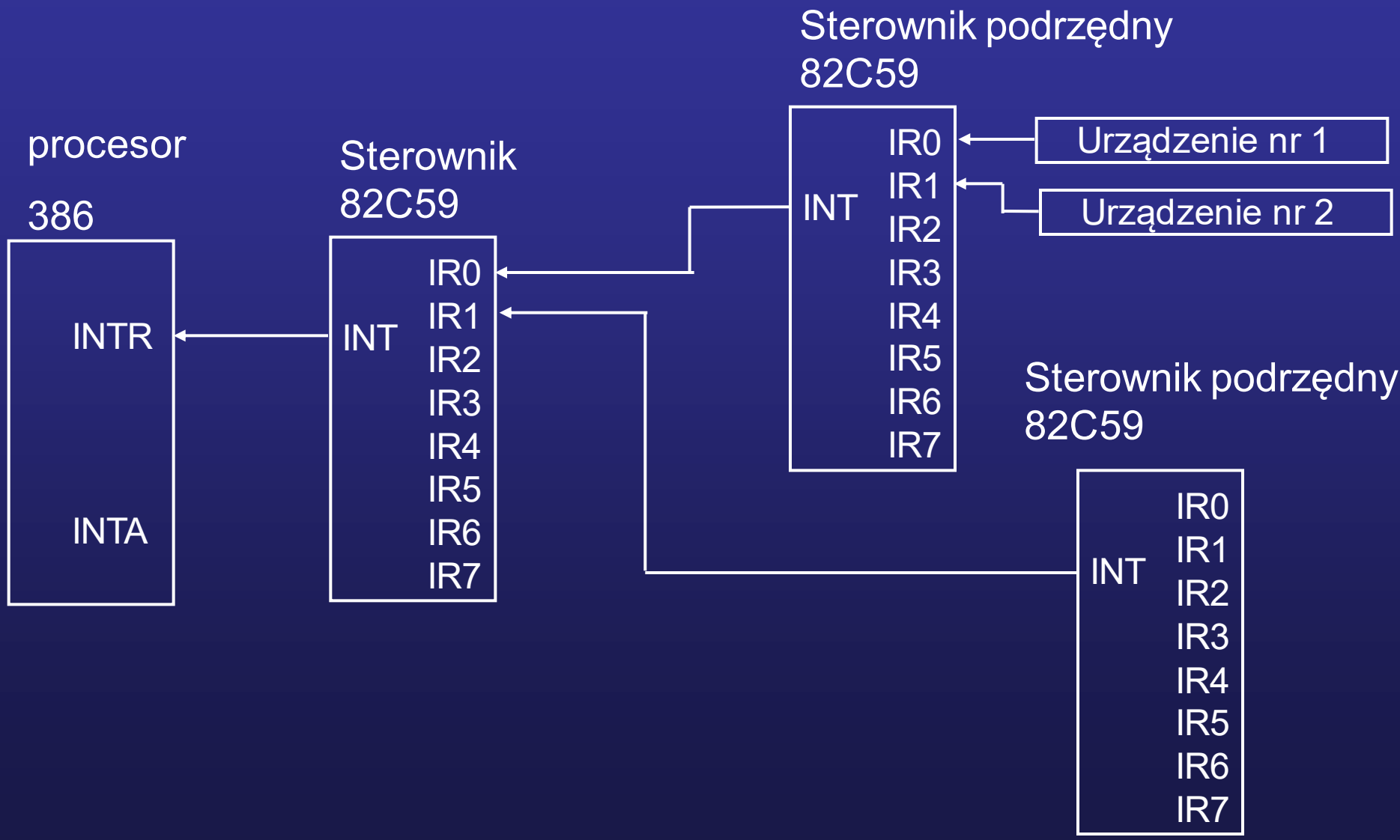


Odpytywanie łańcuchowe



- moduł sygnalizuje zgłoszenie przerwania wystawiając na szynie danych wektor

Sterownik przerwań Intel 82C59A



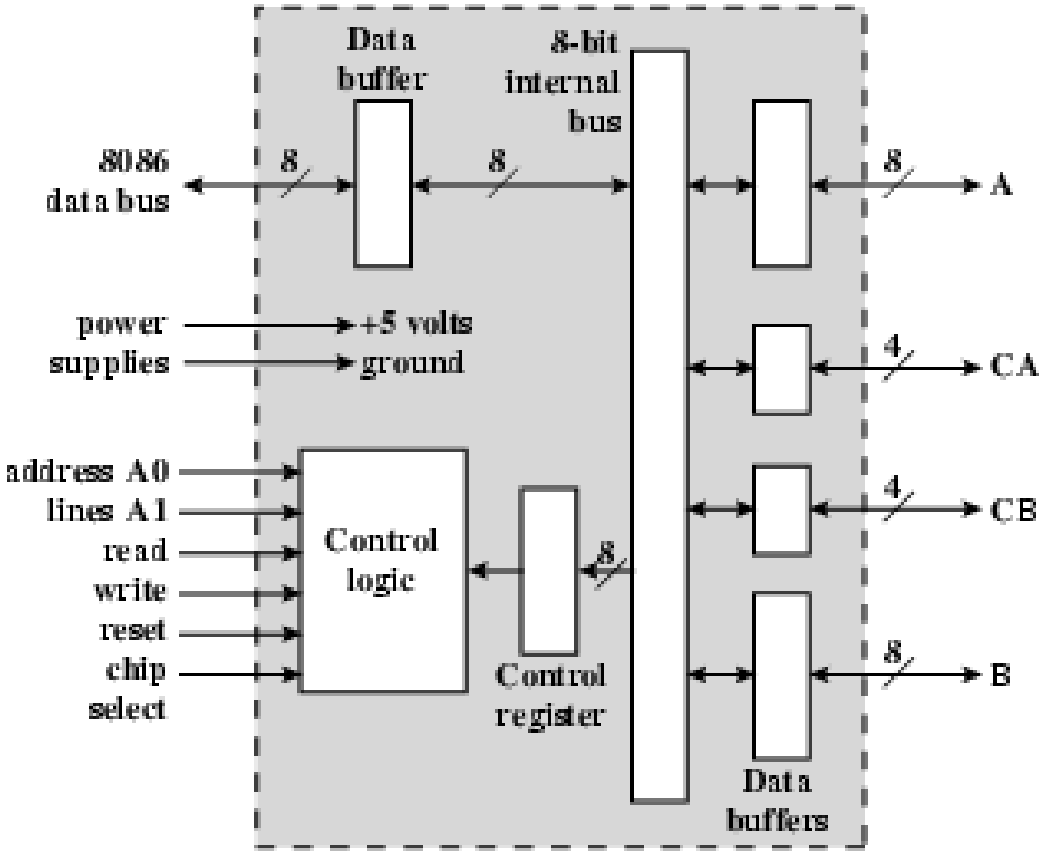
System przerwań magistrali ISA

- dwa układy 8259 w łańcuchu
- połączenie z układami odbywa się poprzez kanał IRQ 2
- Kanał IRQ 9 służy do przekierowywania żądań na IRQ 2
- 15 linii przerwań do łańcucha

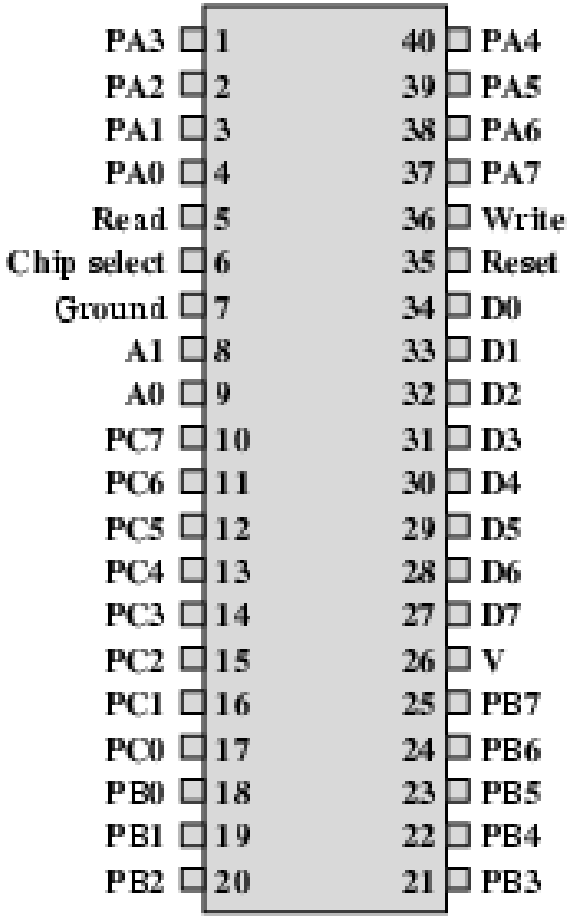
Programowalny interfejs urządzeń peryferyjnych Intel 82C55A

- moduł we/wy używany przez procesor 80386
- układ jest programowalny za pomocą rejestru sterowania (uniwersalność)
- 40 linii sygnałowych:
  - 24 linie wejścia-wyjścia
  - 8 linii (bitów) danych
  - 2 linie adresowe

Schemat układu 82C55A



(a) Block diagram



(b) Pin layout

Bezpośredni dostęp do pamięci

- Direct Memory Access (DMA)
- pozwala przesyłać szybko duże ilości danych z/do pamięci bez ingerencji procesora
- metoda szybsza od sterowania przerwaniami

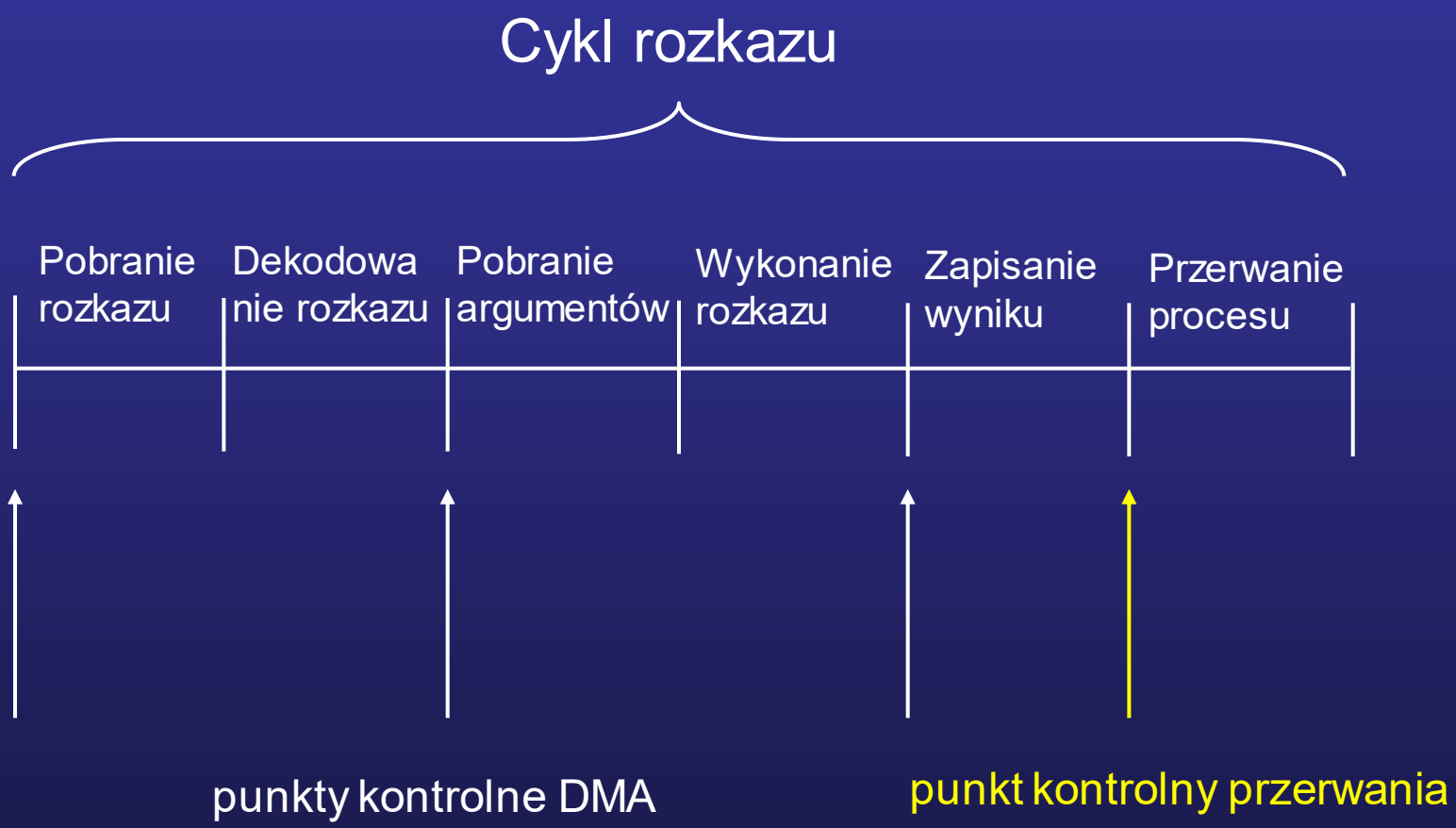
Schemat modułu DMA

- moduł pamięci „udaje” procesor (kradzież cyklu)

Działanie DMA

- informacje żądane przez procesor od kontrolera DMA:
  - zapis, czy odczyt
  - adres urządzenia peryferyjnego
  - adres początkowej komórki pamięci do odczytu/zapisu
  - liczba słów do odczytu/zapisu
- procesor wykonuje kolejne rozkazy programu
- kontroler DMA steruje przepływem informacji pomiędzy peryferiami a pamięcią
- gdy zadanie jest zakończone, kontroler zgłasza przerwanie

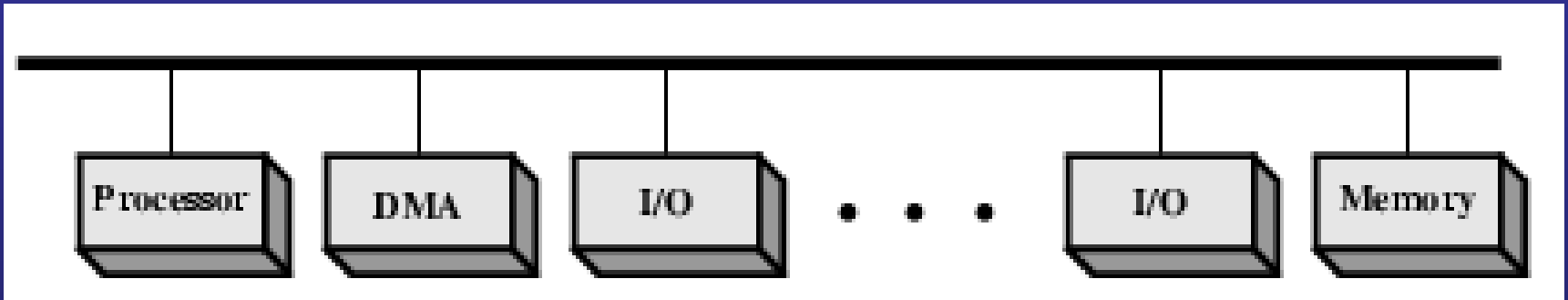
Punkty kontrolne DMA



Konfiguracje DMA

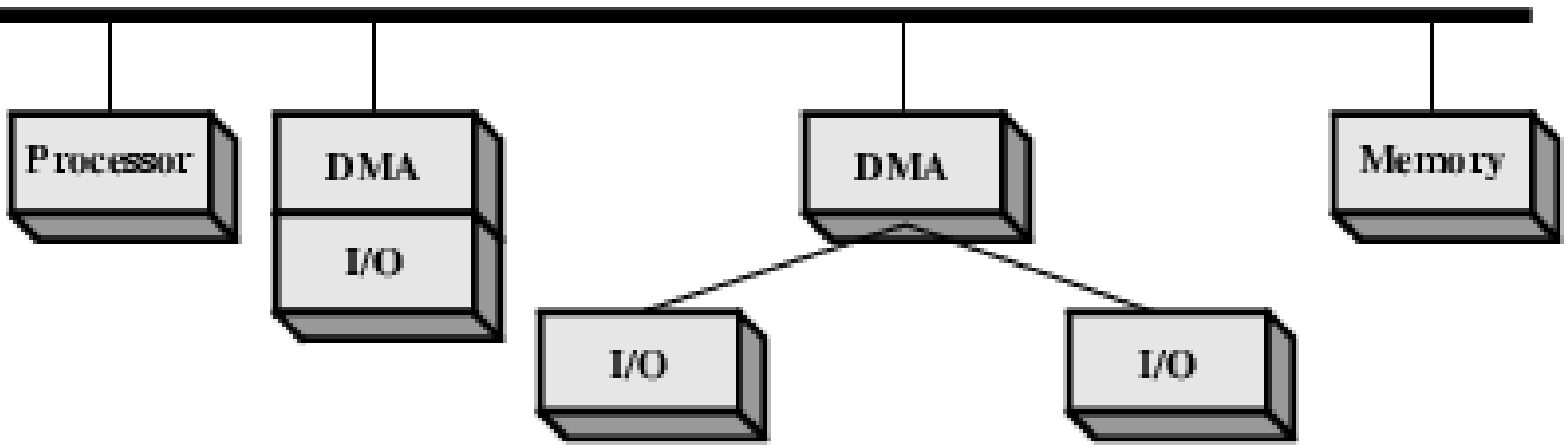
- pojedyncza magistrala dla wszystkich modułów, sterowanie programowe wejściem-wyjściem
- pojedyncza magistrala, zintegrowane DMA
- magistrala wejścia-wyjścia – osobna, tylko dla peryferiów, moduł DMA pełni rolę mostka

Pojedyncza magistrala, odłączane DMA



- zawieszenie procesora następuje dwukrotnie w czasie komunikacji z urządzeniem peryferyjnym
- magistrala również użyta dwukrotnie

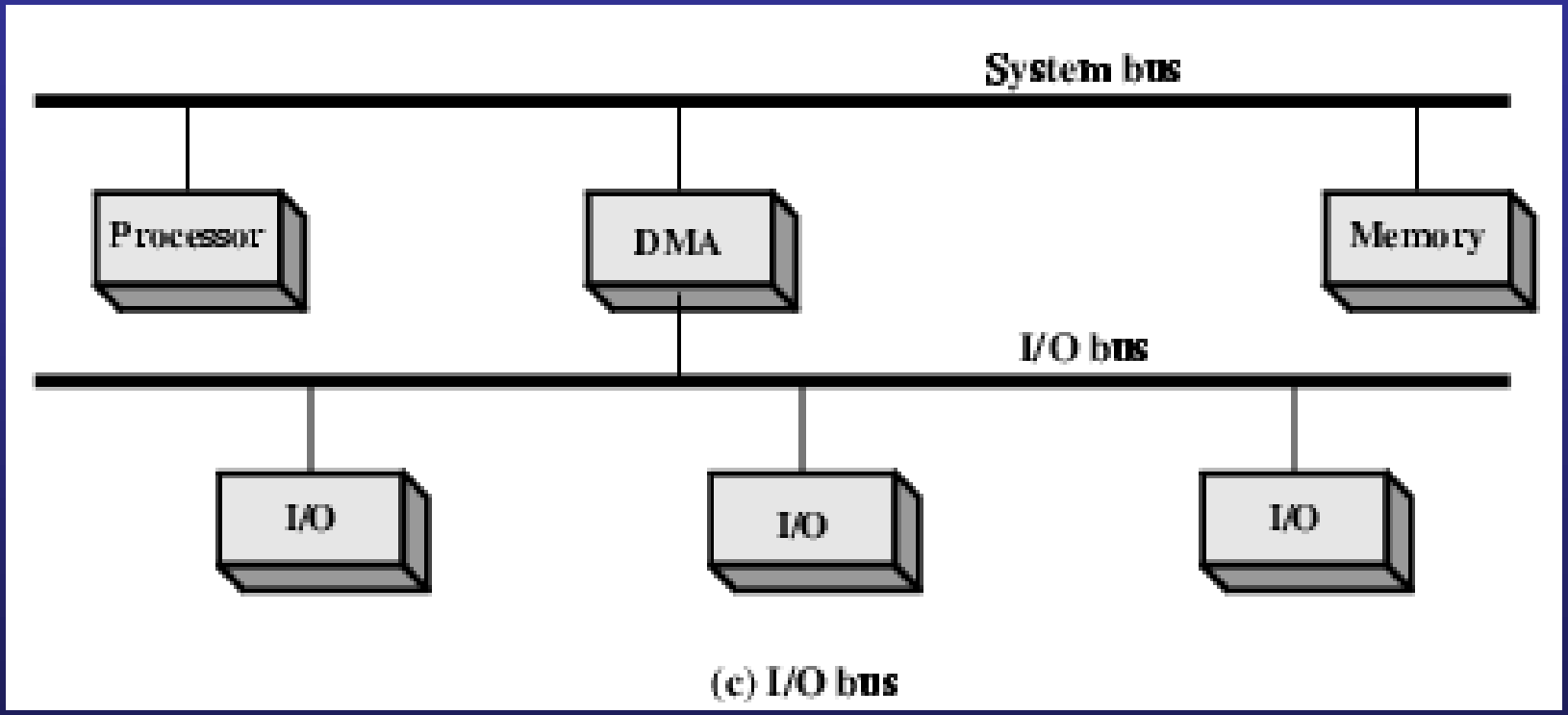
Pojedyncza magistrala, zintegrowane DMA



(b) Single-bus, Integrated DMA-I/O

- kontroler DMA może wspierać więcej, niż jedno urządzenie
- transfer wymaga jednokrotnego dostępu do magistrali, podobnie jest z zawieszeniem procesora

Magistrala wejścia-wyjścia

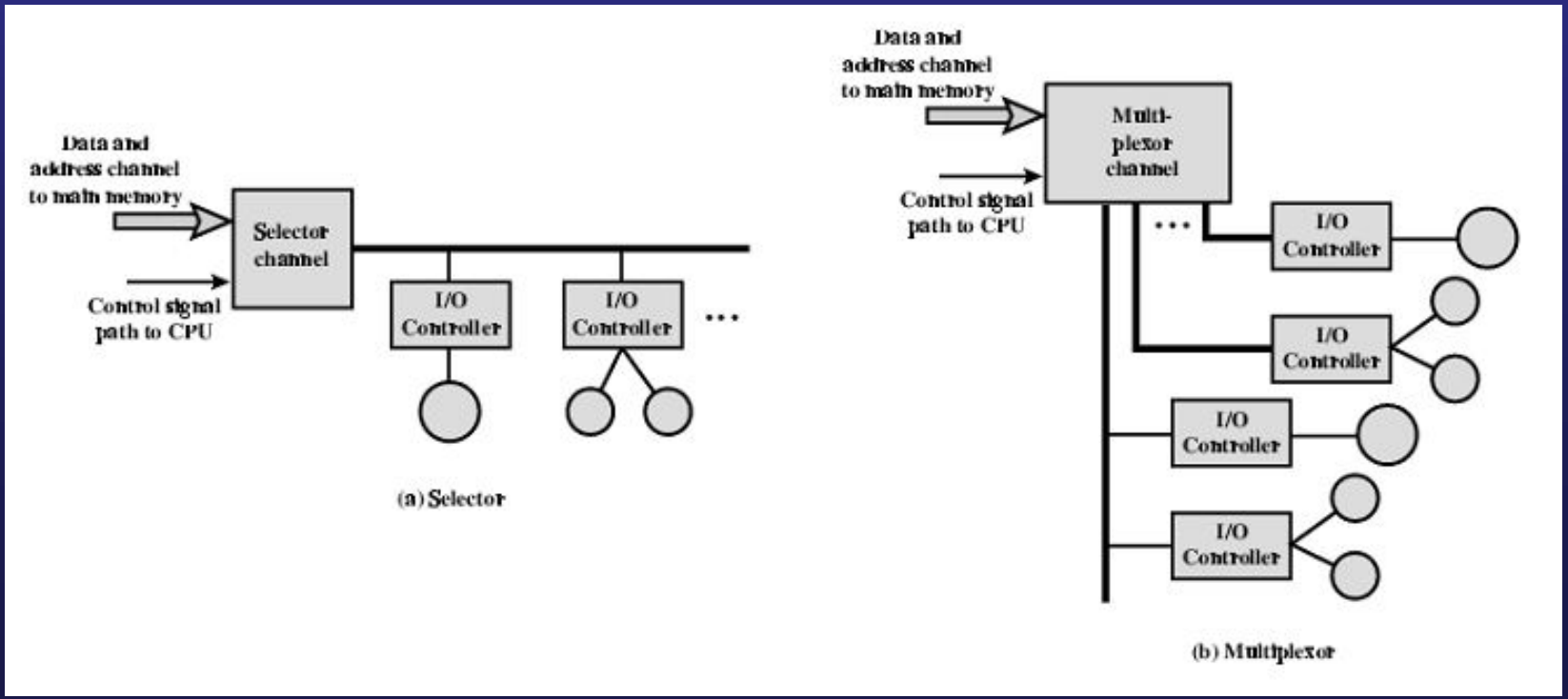


(c) I/O bus

- jednokrotny dostęp do magistrali (DMA do pamięci)
- procesor zawieszany tylko raz

Kanały wejścia-wyjścia

- są to wyspecjalizowane moduły I/O, zdolne do wykonywania programów z pamięci bez udziału CPU
- dwa rodzaje kanałów:
  a) wybiórczy
  b) multiplekserowy



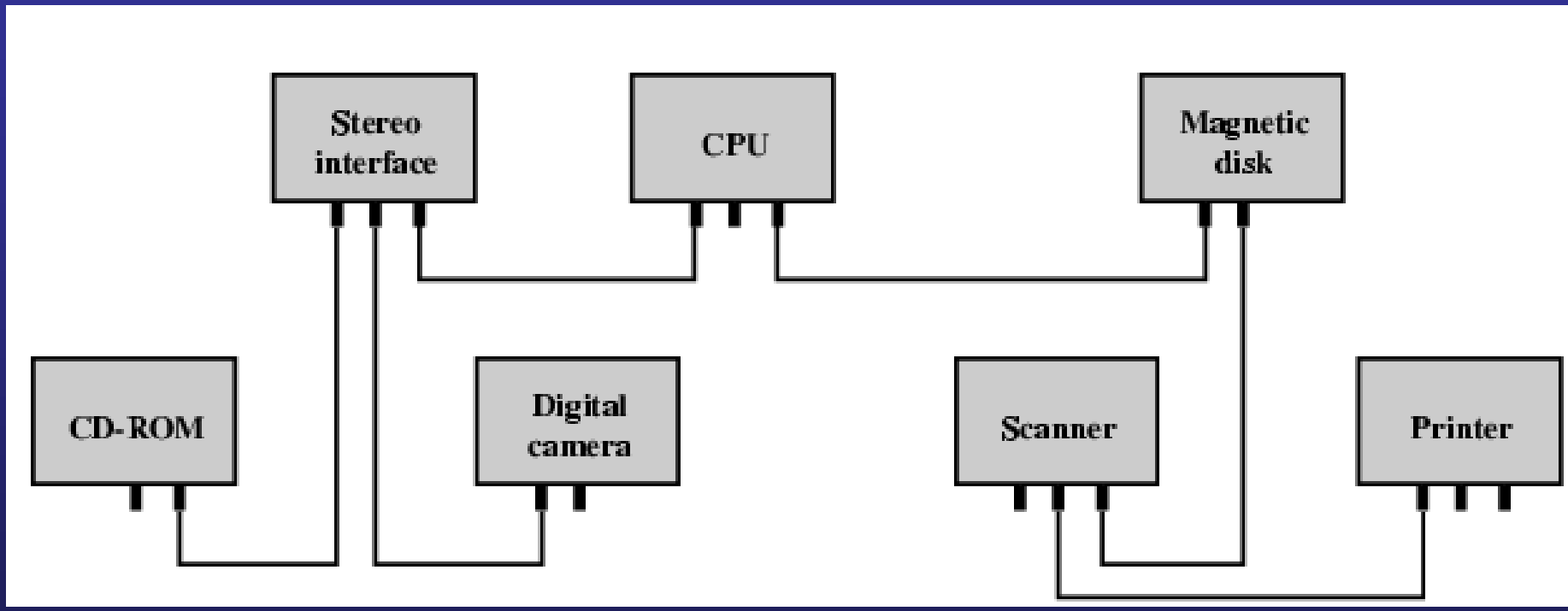
Rodzaje interfejsów zewnętrznych

- interfejs równoległy – wiele linii, szybkie urządzenia (np. dyski twarde)
- interfejs szeregowy – jedna linia, wolne urządzenia (np. drukarki)
- łącze dwupunktowe
- łącze wielopunktowe
- moduł I/O musi mieć bufor do przechowywania danych z/do urządzenia peryferyjnego

Magistrala szeregowa FireWire

- standard IEEE-1394
- pojedyncza linia danych nie wymaga ekranowania
- urządzenia ułożone w konfiguracji łańcuchowej, lub drzewiastej, do jednego portu można dołączyć 63 urządzenia
- aktywne podłączanie urządzeń (hot plugging)

Konfiguracja FireWire



- brak terminatorów!!

Warstwy protokołów FireWire

- warstwa fizyczna
  - definiuje arbitraż, ponowną synchronizację danych, kodowanie-dekodowanie, stan połączenia, złącza, poziom sygnału
- warstwa łącza
  - transmisja danych (nadawanie i odbieranie pakietów, sterowanie cyklem)
- warstwa transakcyjna
  - odczyt i zapis, blokowanie

Organizacja i Architektura Komputerów

Wykład nr 11: Komputery o zredukowanej liście rozkazów

Piotr Bilski

Charakterystyka architektury o zredukowanej liście rozkazów

- Procesor zawiera dużą liczbę rejestrów roboczych
- Kompilatory RISC stosują procedury optymalizacji wykorzystania rejestrów
- Liczba rozkazów jest niewielka w stosunku do CISC
- Rozkazy maja prostą postać
- Optymalizacja potoku rozkazów

Porównanie wybranych procesorów

Parametr	IBM 370	VAX 11	80486	SPARC	MIPS R4000	PowerPC	Ultra Sparc
Rok prod.	1973	1978	1989	1987	1991	1993	1996
Liczba rozkazów	208	303	235	69	94	225	
Rozmiar rozkazu	2-6 B	2-57 B	1-11 B	4 B	4 B	4 B	4 B
Tryby adr.	4	22	11	1	1	2	1
Liczba rejestrów	16	16	8	40-520	32	32	40-520
Rozmiar cache	64 kB	64 kB	8 kB	32 kB	128 kB	16/32 kB	32 kB

RISC a języki programowania

- Języki wysokiego poziomu – ułatwienie dla programisty, wiele szczegółów ukrytych
- Konieczność tłumaczenia języka wysokiego poziomu na język maszynowy
- Małe programy wysokiego poziomu $\rightarrow$ duże programy maszynowe
- Zwiększenie efektywności wymaga konstrukcji złożonych kompilatorów

Właściwości wykonywania rozkazów

- Rodzaje rozkazów – rodzaje operacji wykonywanych przez procesor
- Rodzaje i liczba argumentów w rozkazach
- Szeregowanie rozkazów – przetwarzanie potokowe

Analiza częstotliwości wykonywania rozkazów w programie

Język	Pascal	Fortran	Pascal	C	SAL
Zadanie	Naukowe	Studenckie	systemowe	systemowe	systemowe
Przypisanie	73	67	45	38	42
Pętle	4	3	5	3	4
Wywołanie	1	3	15	12	12
If	20	11	29	43	36
Goto	2	9	0	3	0
inne	0	7	6	1	6

Analiza częstotliwości wykonywania rozkazów w programie (c.d.)

	Występowanie dynamiczne		Ważona wg rozkazów maszynowych		Ważona wg odniesień do pamięci	
Język	Pascal	C	Pascal	C	Pascal	C
ASSIGN	45%	38%	13%	13%	14%	15%
LOOP	5%	3%	42%	32%	33%	26%
CALL	15%	12%	31%	33%	44%	45%
IF	29%	43%	11%	21%	7%	13%
GOTO	0	3%	0	0	0	0
inne	6%	1%	3%	1%	2%	1%

Rodzaje danych przetwarzane przez rozkazy

	Pascal	C	Średnia
Stałe całkowite	16 %	23 %	20 %
Zmienne skalarne	58 %	53 %	55 %
Tablice/ struktury	26 %	24 %	25 %

Wykorzystanie argumentów w wywołaniach procedur

Udział programów zawierających:	Kompilator, interpreter	Małe programy nienumeryczne
3-4 argumenty	0-7 %	0-5 %
5-7 argumentów	0-3 %	0 %
8-11 argumentów	1-20 %	0-6 %
12 i więcej argumentów	1-6 %	0-3 %

Tablice rejestrów

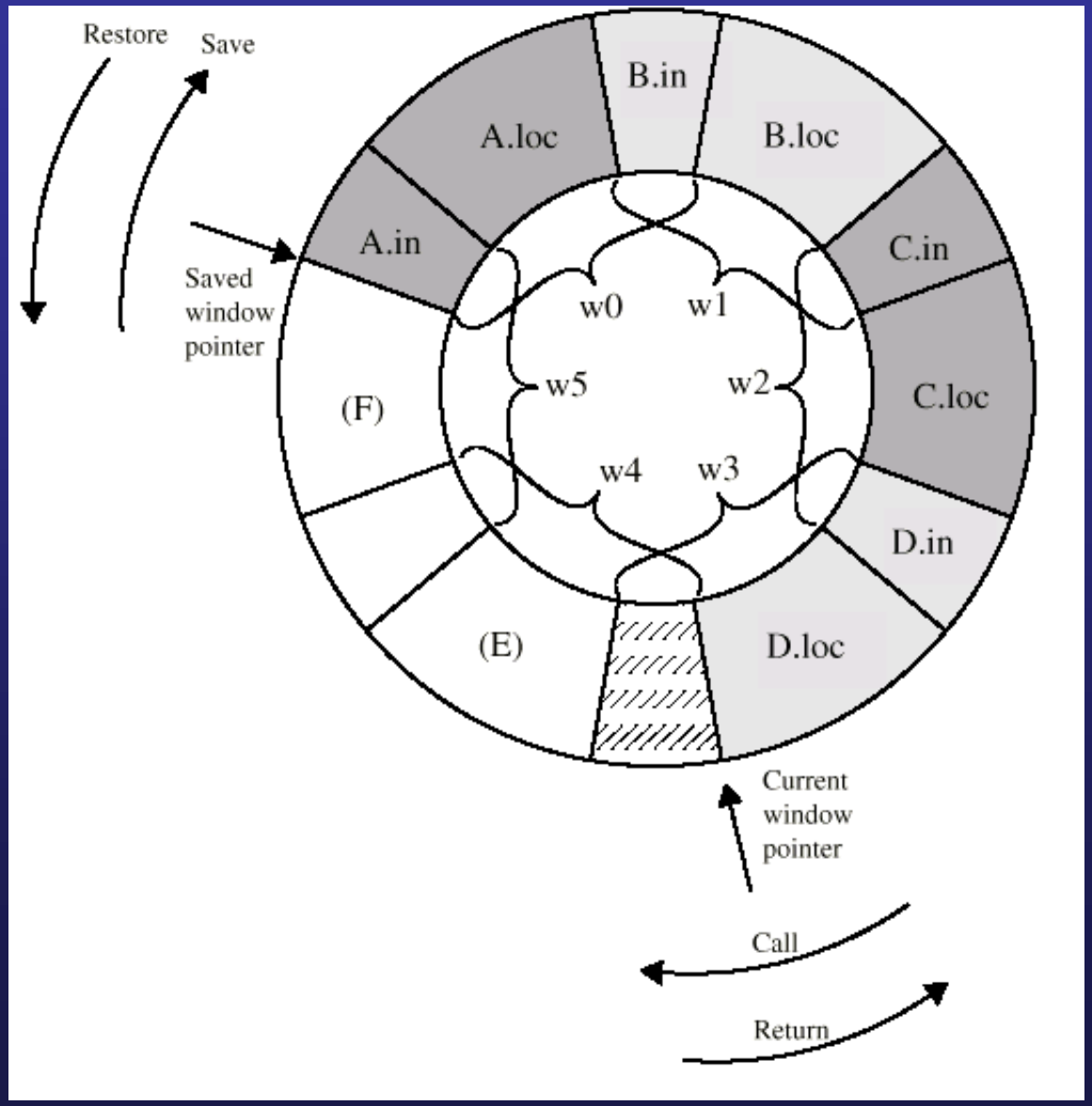
- konieczność optymalizacji licznych operacji przypisania – rejestry
- zmienne lokalne powinny być przechowywane w jednym bloku rejestrów
- problem przekazywania argumentów do funkcji i zwracania wartości
- zagnieżdżone wywoływanie funkcji a liczba rejestrów

Okna rejestrów

- ponieważ funkcje wykorzystują zwykle małą liczbę zmiennych lokalnych, każda funkcja potrzebuje małej liczby rejestrów
- każdej funkcji przypisany jest odrębny zestaw rejestrów (tzw. okno)
- sąsiednie okna nakładają się, co umożliwia przekazywanie argumentów między funkcjami

Okna rejestrów - organizacja

Okna rejestrów - przykład



- wskaźnik obecnego okna (CWP)
- wskaźnik zapisanego okna (SWP)

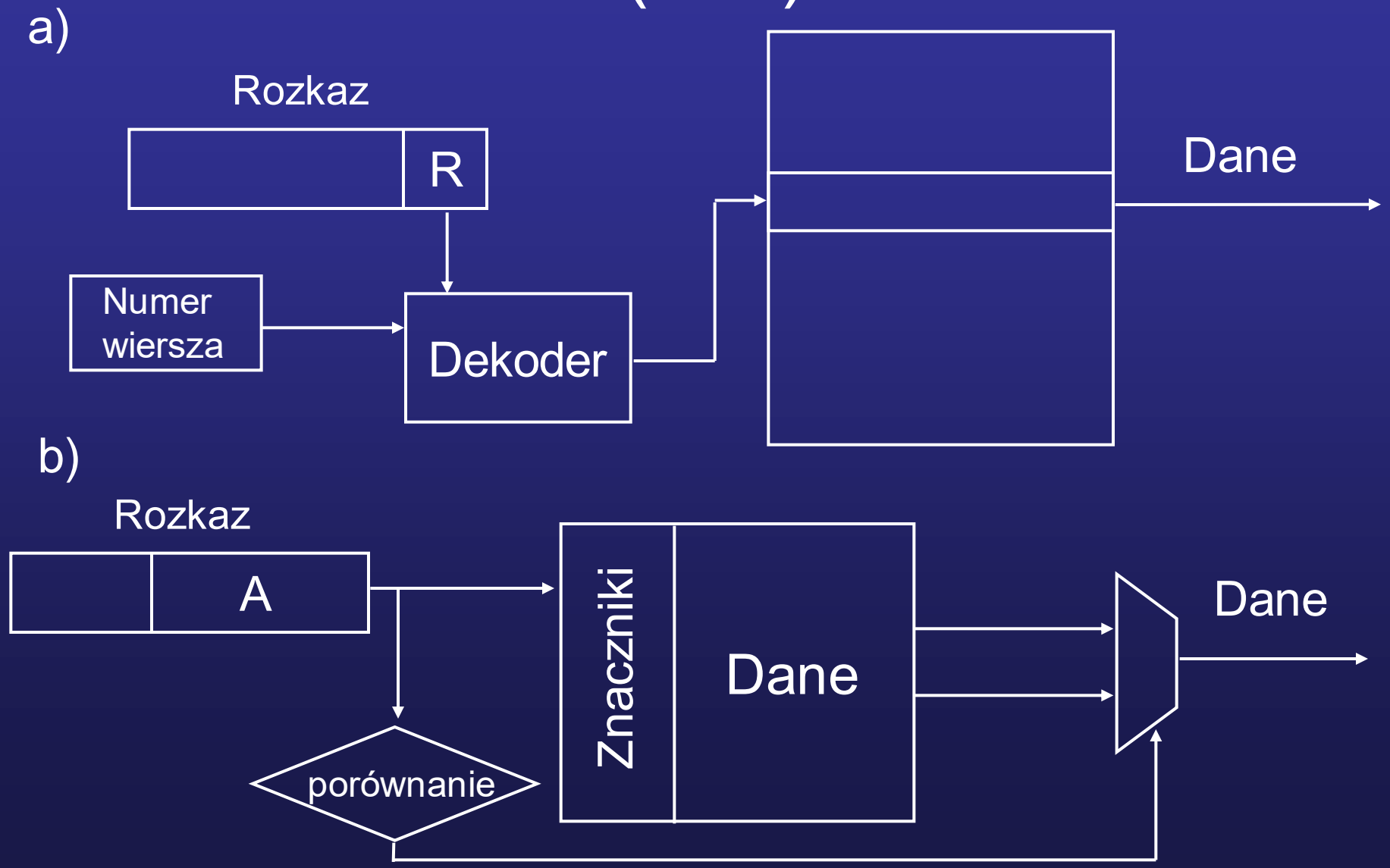
Zmienne globalne

- obsługiwane przez zestaw rejestrów globalnych
- problem adresowania rejestrów globalnych i okna – jednolity sposób adresowania
- decyzja o przypisaniu zmiennych do rejestru zależy od kompilatora

Tablica rejestrów a pamięć podręczna

Duża tablica rejestrów	Pamięć podręczna
wszystkie skalary lokalne	ostatnio używane skalary lokalne
pojedyncze zmienne	bloki pamięci
zmienne globalne wskazane przez kompilator	ostatnio używane zmienne globalne
zmienne przechowywane w zależności od zagnieżdżenia funkcji	zmienne przechowywane w zależności od algorytmu wymiany
adresowanie rejestrów	adresowanie pamięci

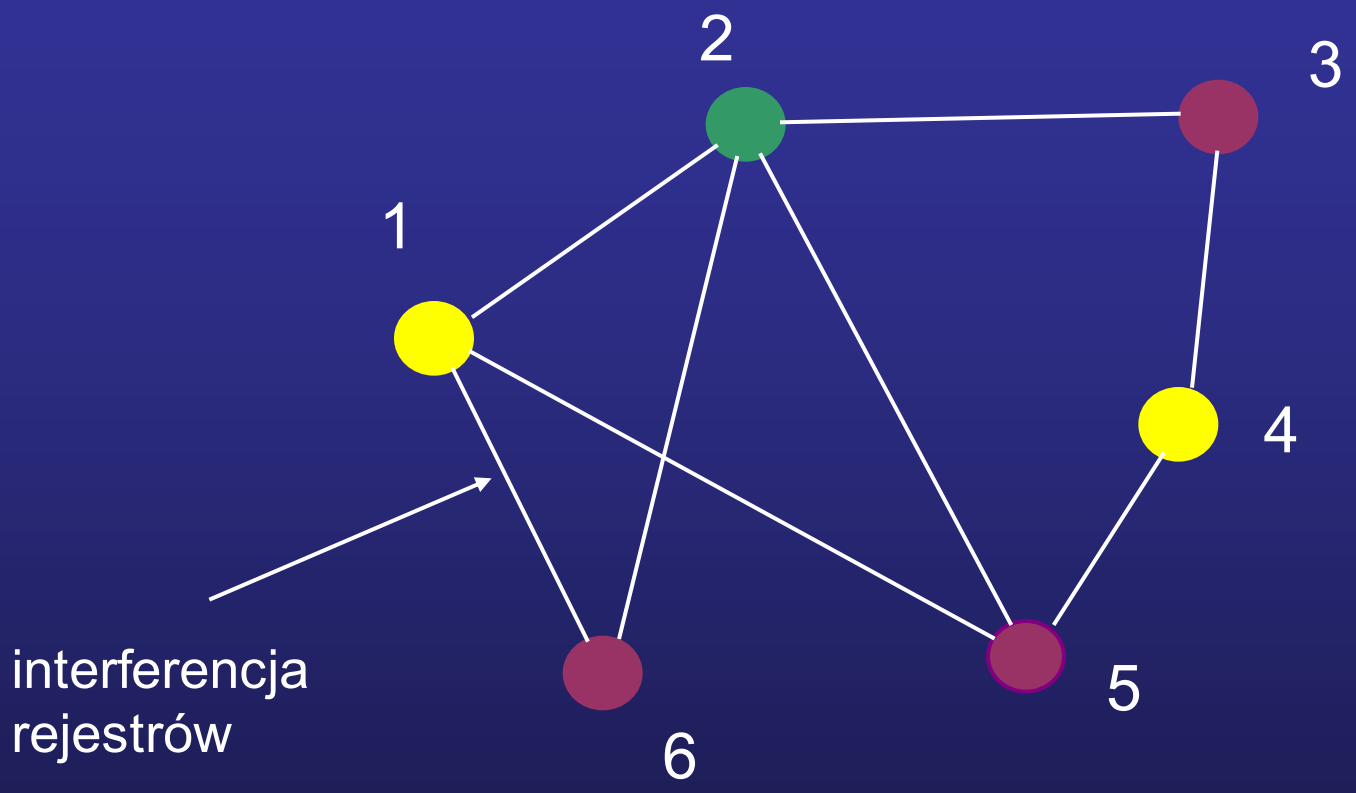
Tablica rejestrów a pamięć podręczna (c.d.)



Optymalizacja rejestrów za pomocą kompilatora

- gdy liczba rejestrów jest za mała, potrzebna jest decyzja, które zmienne do nich wpisać
- tworzona jest nieograniczona liczba rejestrów symbolicznych, które odwzorowywane są na rejestry rzeczywiste
- przypisanie to jest problemem kolorowania grafu

Przykład kolorowania grafu



- wierzchołki to rejestry symboliczne
- kolory to rejestry rzeczywiste

Problemy projektowania kompilatora

- czy krótszy program przekłada się na krótszy program maszynowy?
- czy programy zawierające bardziej złożone procedury są szybsze?

	proc. 1	proc. 2	proc. 3
RISC I	1,0	1,0	1,0
VAX 11	0,8	0,67	
M68000	0,9		0,9

Własności rozkazów RISC

- jeden rozkaz wykonywany w ciągu cyklu
- operacje przenoszenia danych typu „z rejestru do rejestru”
- wyłącznie proste tryby adresowania
- proste formaty rozkazów

Rozkaz maszynowy RISC

- jeden rozkaz w czasie jednego cyklu maszynowego
- cykl maszynowy to czas wymagany do pobrania dwóch argumentów z rejestru, wykonania operacji przez ALU i zapisania wyniku w rejestrze
- brak mikrokodu, wykonanie rozkazu sprzętowe!

Przesyłanie danych rejestr-rejestr

- wszystkie dane powinny być w rejestrach
- wyjątek to nieliczne odwołania do pamięci (LOAD, STORE)
- lista rozkazów krótsza, jednostka sterująca prostsza

8	16	16	16
Dodaj	B	C	A
Dodaj	A	C	B
Odejmij	B	D	D

Rozmiar rozkazów = 168 B

8	4	4	4
Dodaj	rA	rB	rC
Dodaj	rB	rA	rC
Odejmij	rD	rD	rB

Rozmiar rozkazów = 60 B

Proste formaty rozkazu

- długość wszystkich rozkazów jest stała!
- proste rozkazy łatwiej optymalizować na etapie kompilacji
- ALU dla prostszych rozkazów jest prostsze i szybsze
- przetwarzanie potokowe jest bardziej efektywne
- łatwiejsza obsługa przerwań
- najczęściej wykonywane rozkazy mogą być implementowane sprzętowo

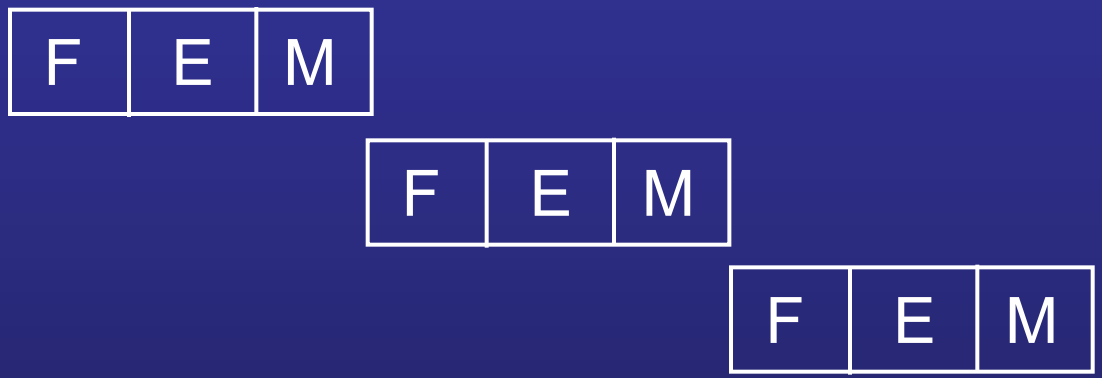
CISC a RISC - porównanie

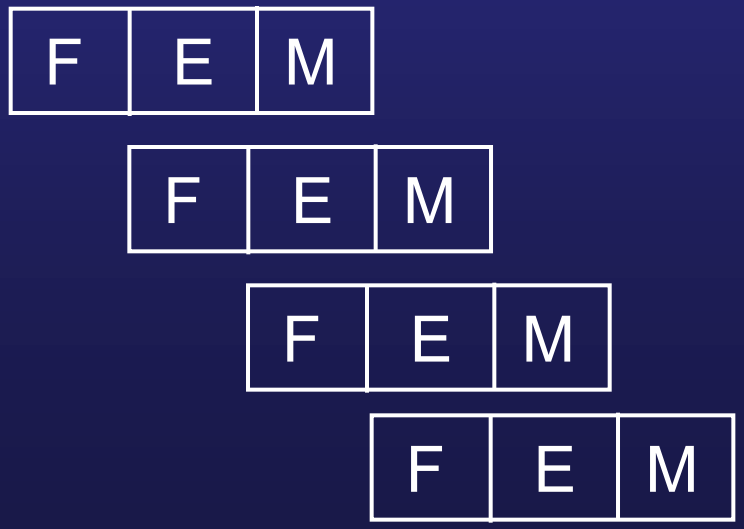
Procesor	Rodzaj	Rozmiar rozkazu	Adresowanie pośrednie	Liczba argumentów
R2000	RISC	4 B	nie	1
SPARC	RISC	4 B	nie	1
IBM RS/6000	RISC	4 B	nie	1
Intel i860	RISC	4 B	nie	1
80486	CISC	12 B	nie	2
M68040	CISC	22 B	tak	2
VAX	CISC	56 B	tak	6

Przetwarzanie potokowe

- przetwarzanie rejestr-rejestr
  - pobranie rozkazu (F)
  - wykonanie rozkazu (E)
- ładowanie z pamięci i zapis do niej
  - pobranie rozkazu (F)
  - wykonanie rozkazu (E)
  - operacja na pamięci (M)

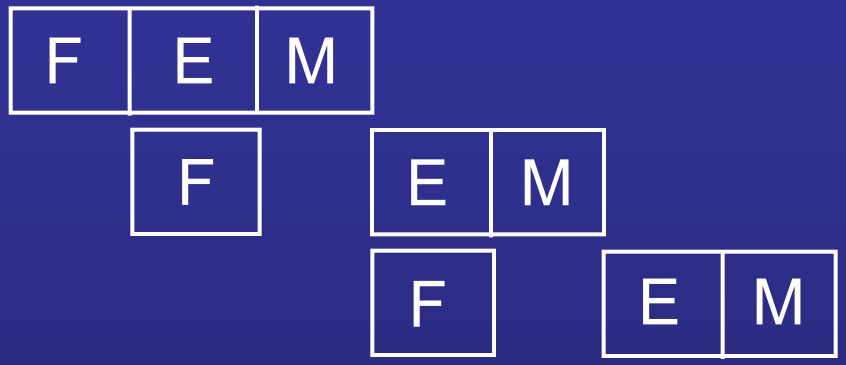
Wykonywanie sekwencyjne a potok



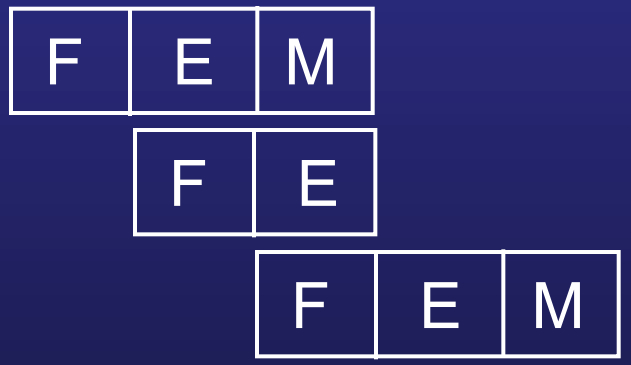


Przetwarzanie potokowe

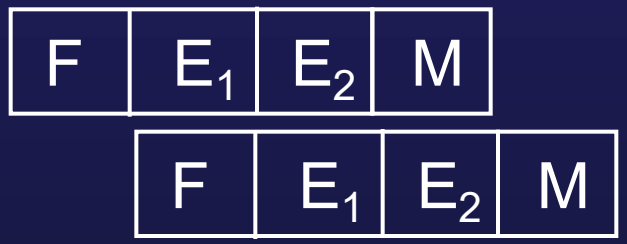
- 2-etapowe – jeden dostęp do pamięci



- 3-etapowe – dwa dostępy do pamięci



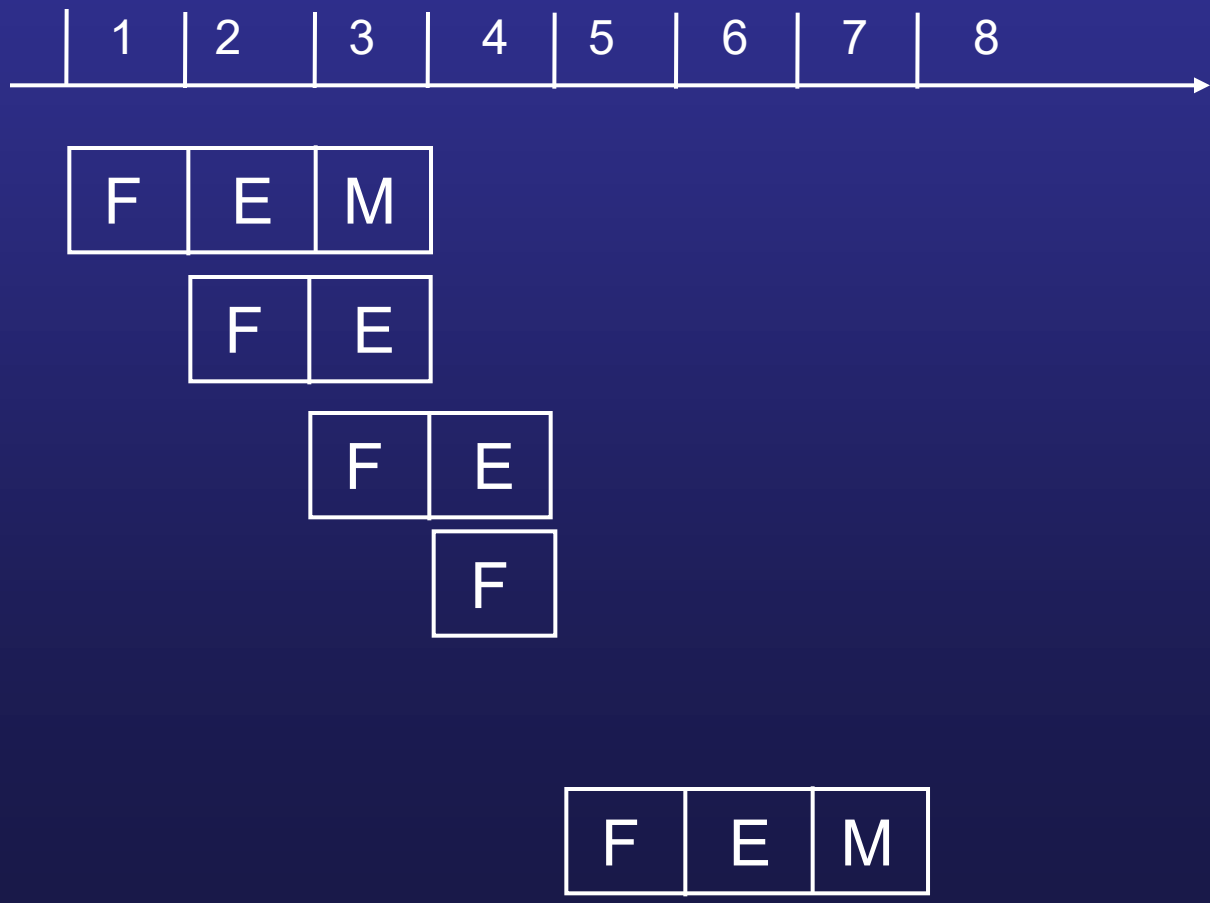
- 4-etapowe



Przetwarzanie potokowe ze skokiem

Program:

```
100 LOAD   X,A
101 ADD     1,A
102 JUMP   105
103 ADD     A,B
104 SUB     C,B
105 STORE A,Z
```



Wstawienie rozkazu pustego

Program:

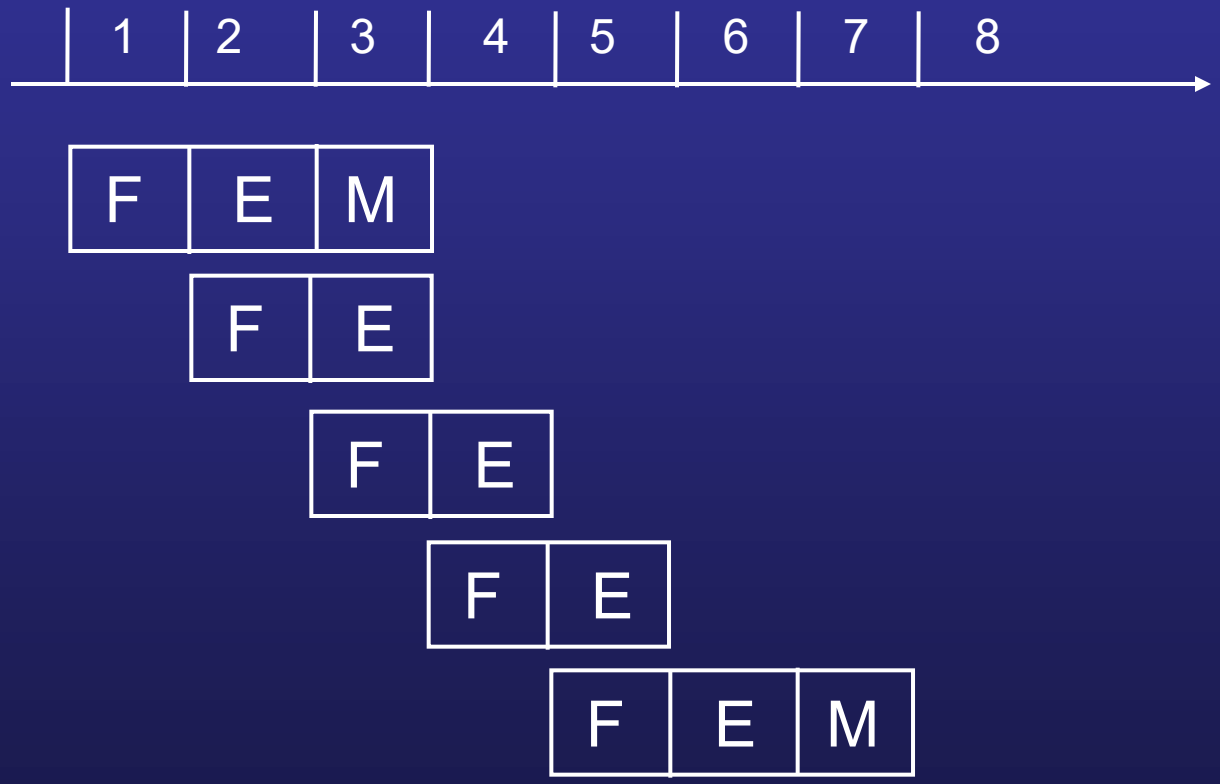
100 LOAD X,A

101 ADD 1,A

102 JUMP 106

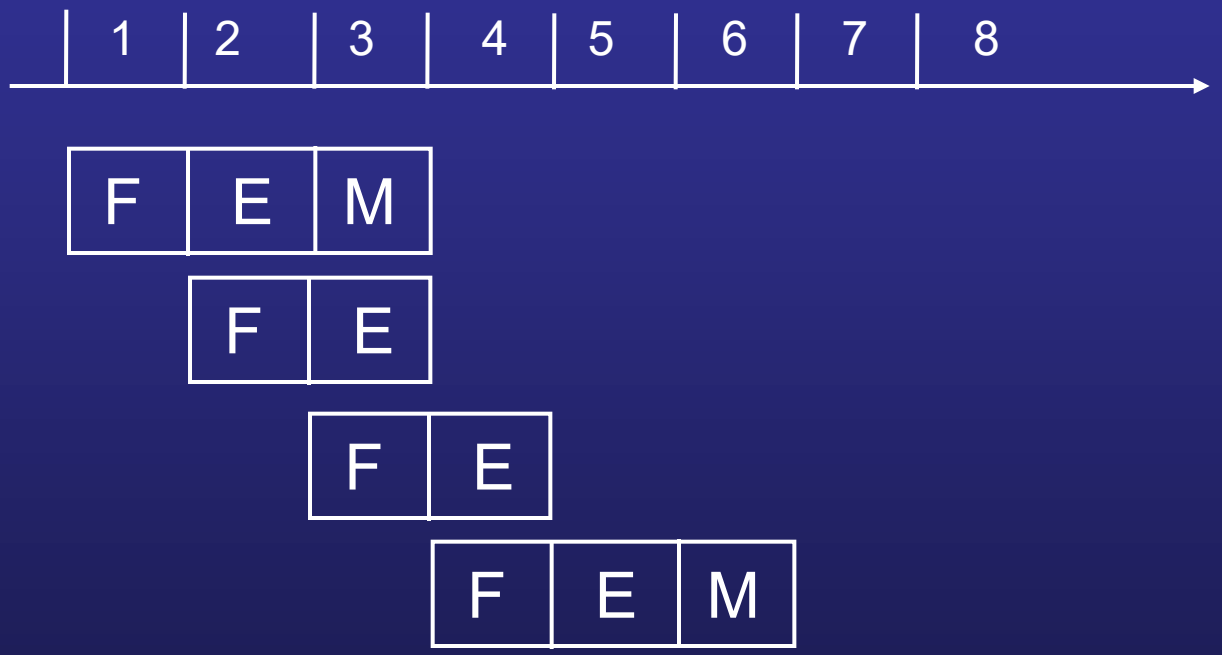
103 NOOP

106 STORE A,Z



Odwrócona kolejność rozkazów

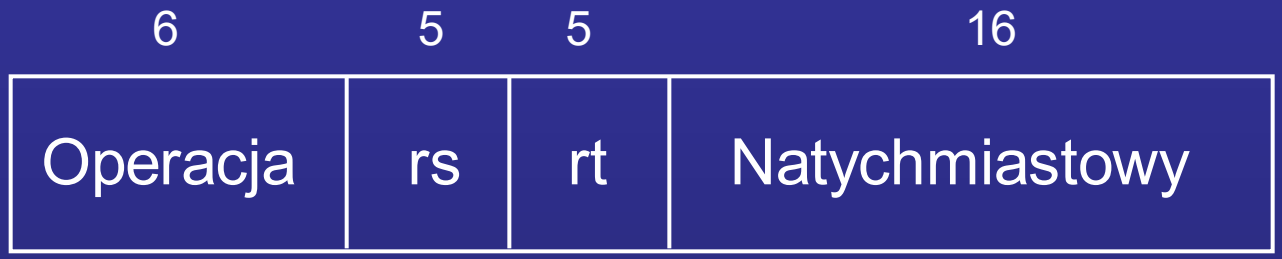
Program:



Przykład RISC – MIPS R4000

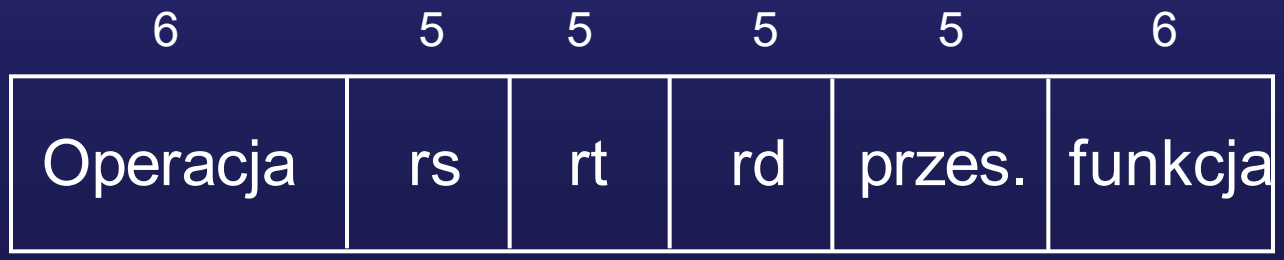
- procesor o architekturze 64-bitowej (taka długość rejestrów i magistrali ALU)
- w układzie R4000 znajduje się procesor oraz jednostka zarządzania pamięcią – MMU
- 32 rejestry robocze, do 128 kB pamięci podręcznej (po połowie na rozkazy i dane)
- stały format rozkazów – 4 bajty
- brak kodów warunkowych
- trzy formaty rozkazów

Formaty rozkazów MIPS R4000



akcja na argumencie natychmiastowym

akcja skoku



adresowanie rejestrowe

Przykład RISC – Sun SPARC

- procesor wykorzystuje okna rejestrów (od 2 do 32 okien po 24 rejestry)
- osiem rejestrów globalnych (0-7)
- rejestry wyjściowe (wywoływane wraz z procedurą wywoływaną, 8-15)
- rejestry wejściowe (używane z procedurą wywołującą, 24-31)
- rejestry lokalne, o numerach 16-23
- wszystkie rozkazy 32-bitowe

Formaty rozkazów SPARC