

Architektura systemów komputerowych

Procesor – część I

1. ALU
2. Cykl rozkazowy
3. Schemat blokowy CPU
4. Architektura CISC i RISC

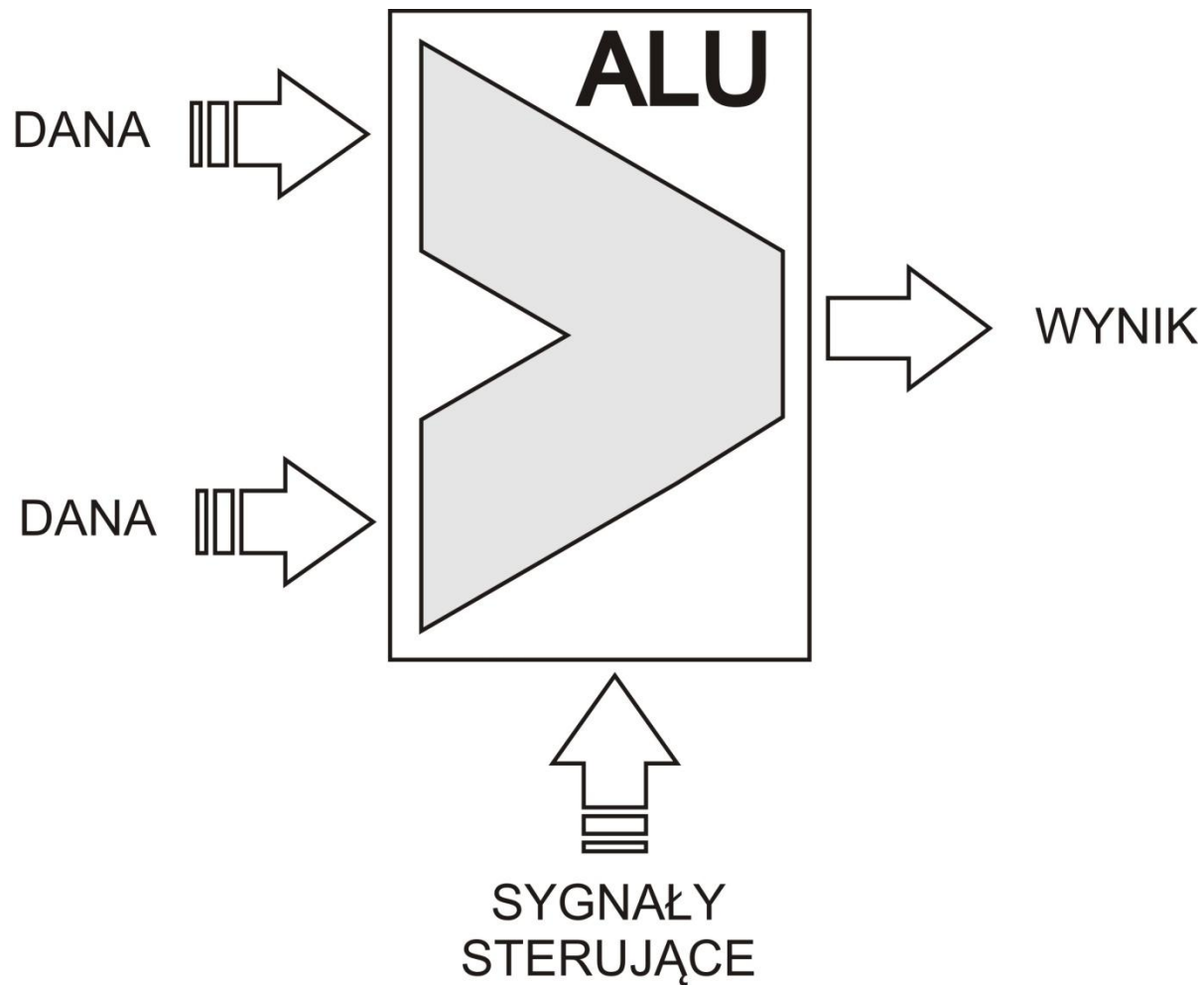


Schemat blokowy mikroprocesora (I)

Najważniejszym elementem procesora jest ALU – jednostka arytmetyczno-logiczna (ang. Arithmetic Logic Unit) nazywana niekiedy arytmetrem.

Jest to uniwersalny układ cyfrowy w którym wykonywane są operacje arytmetyczne (dodawanie, odejmowanie, dzielenie, mnożenie) oraz logiczne na dostarczanych do niego liczbach.

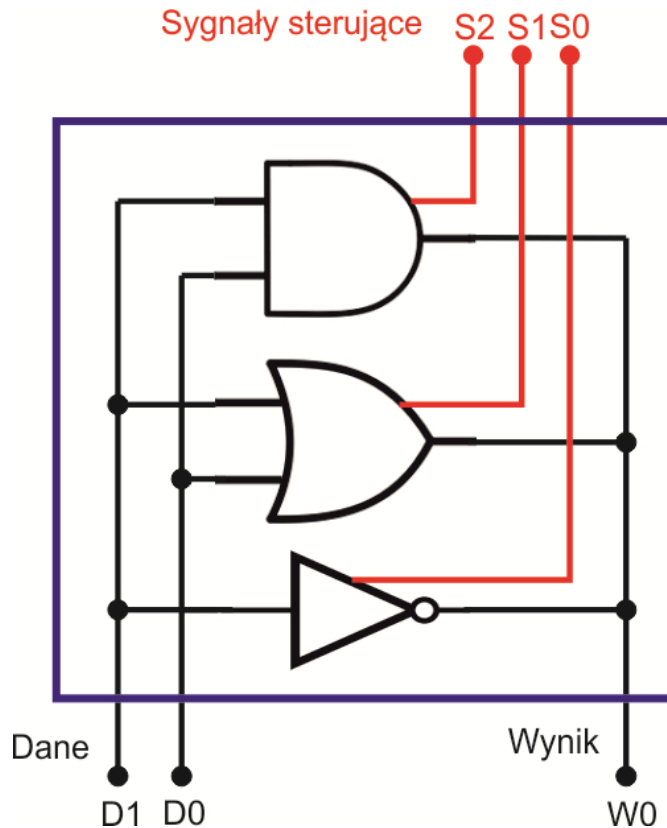
Dane pobierane są z pamięci operacyjnej lub rejestrów, a o tym, jaka operacja zostanie na nich wykonana decydują sygnały sterujące.



Zestaw operacji ALU powinien być **funkcjonalnie pełny**, tzn. taki za pomocą którego jesteśmy w stanie zrealizować dowolny algorytm przetwarzania informacji.

Każda lista rozkazów zawiera kilka grup działań występujących w różnych wersjach niemal w każdym komputerze są to:

- ✓ przesłania,
- ✓ działania arytmetyczne,
- ✓ działania logiczne,
- ✓ przesunięcia,
- ✓ sterowanie przebiegiem programu, przesłania wejścia-wyjścia, działania zmiennopozycyjne, działania na argumentach upakowanych.



Ten schemat prezentuje ideę działania ALU

Nasze ALU otrzymuje dwubitowe słowo na wejściu i wykonuje na nim jedną z trzech operacji logicznych (AND OR NOT) .

Za każdą operację odpowiada inny układ wewnątrz ALU.

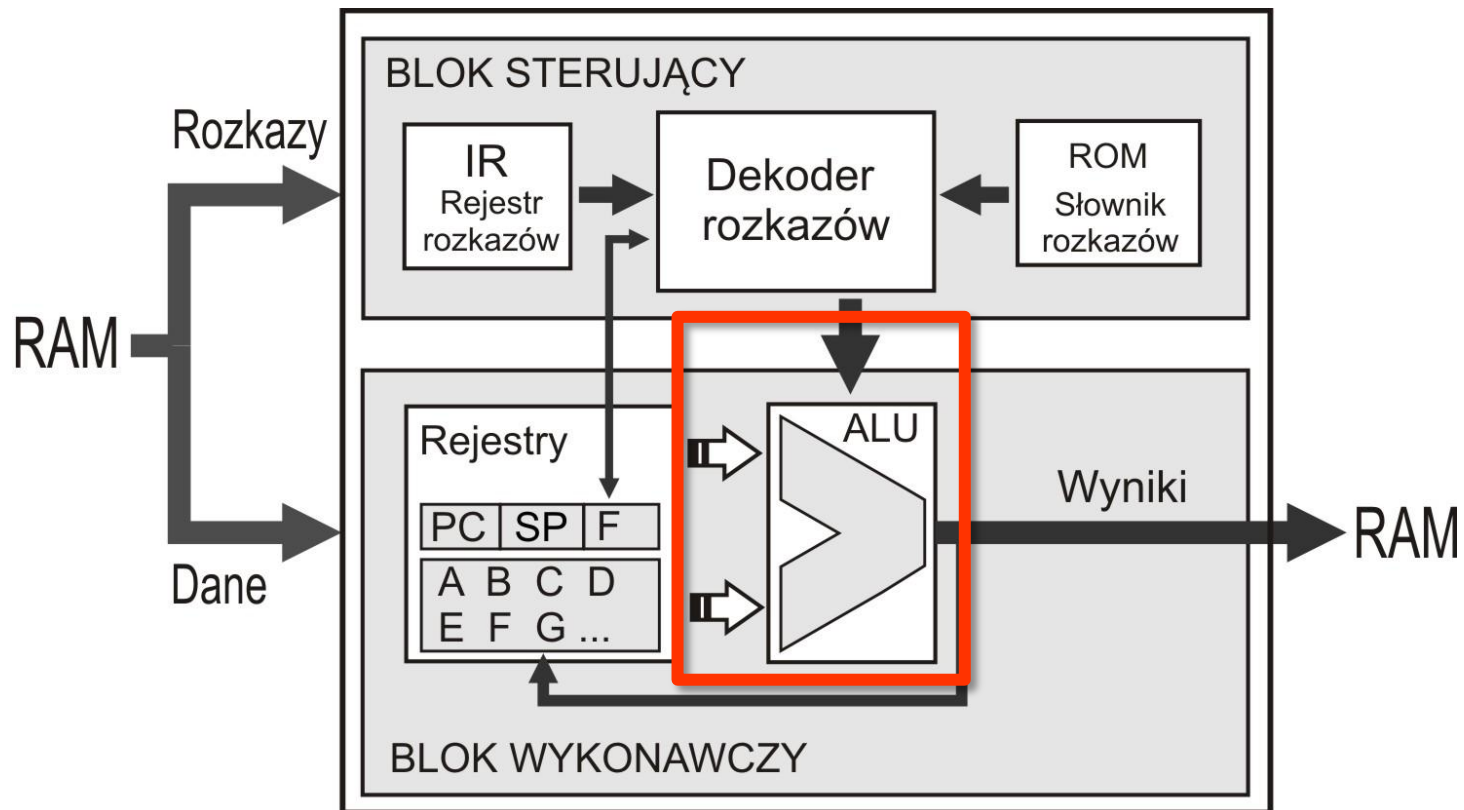
Sygnały sterujące wybierają odpowiedni układ.

Sygnały sterujące S2 S1 S0	Działanie
1 0 0	D1 AND D0
0 1 0	D1 OR D0
0 0 1	NOT D1

Oczywiści na takim ALU nie zbudujemy procesora. W rzeczywistości układy wewnątrz ALU nie są pojedynczymi bramkami, i jest ich znacznie więcej

Systemu mikroprocesorowy

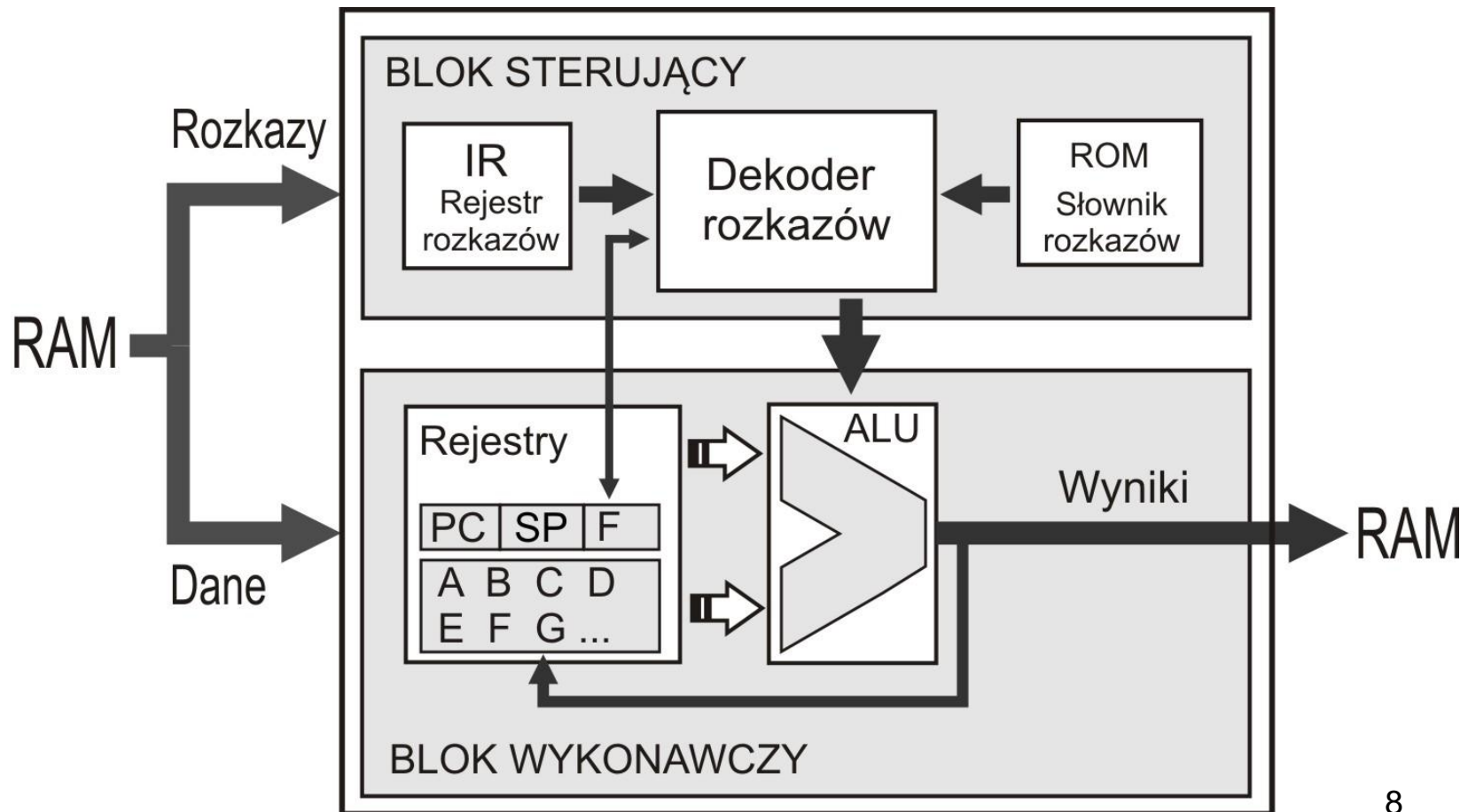
- ALU jest tylko częścią składową procesora. Nie posiada pamięci ani urządzeń umożliwiających współpracę z pamięcią RAM.
- ALU współpracuje z zestawem rejestrów.



Schemat blokowy mikroprocesora (I)

W procesorze wyróżnić możemy dwa główne bloki:

1. CU - BLOK STERUJĄCY (*ang. Control Unit*)
2. EU - BLOK WYKONAWCZY (*ang. Execution Unit*)



Jednostkę arytmetyczno-logiczną wraz z zespołem rejestrów nazywamy blokiem wykonawczym procesora – EU (*ang. Execution Unit*)

- ✓ ALU można wyobrazić sobie, jako zestaw wielu prostych układów elektroniki cyfrowej, z których każdy wykonuje pojedynczą operację arytmetyczną lub logiczną.
- ✓ Sygnały sterujące uaktywniają taką kombinację tych układów, która jest potrzebna w danej chwili, do wykonania aktualnie przetwarzanego rozkazu lub jego części.
- ✓ Zmiana sygnałów sterujących powoduje uaktywnienie nowej kombinacji układów i przełączenie się ALU inna operację.

Przetwarzaniem poleceń programu – rozkazów asemblera dla danego procesora – na wewnętrzne sygnały sterujące zajmuje się **blok sterujący – CU** (*ang. Control Unit*)

W jego skład wchodzi:

- ✓ **rejestr rozkazów (IR)**, w którym przechowywany jest kod aktualnie wykonywanego rozkazu;
- ✓ **dekoder rozkazów**, którego zadaniem jest rozpoznanie pobranego z pamięci operacyjnej rozkazu i wygenerowanie na jego podstawie sekwencji sygnałów sterujących dla ALU oraz pozostałych podzespołów procesora;
- ✓ **pamięć ROM zawierająca słownik rozkazów** (nie należy mylić z pamięcią ROM umieszczoną na płycie głównej).

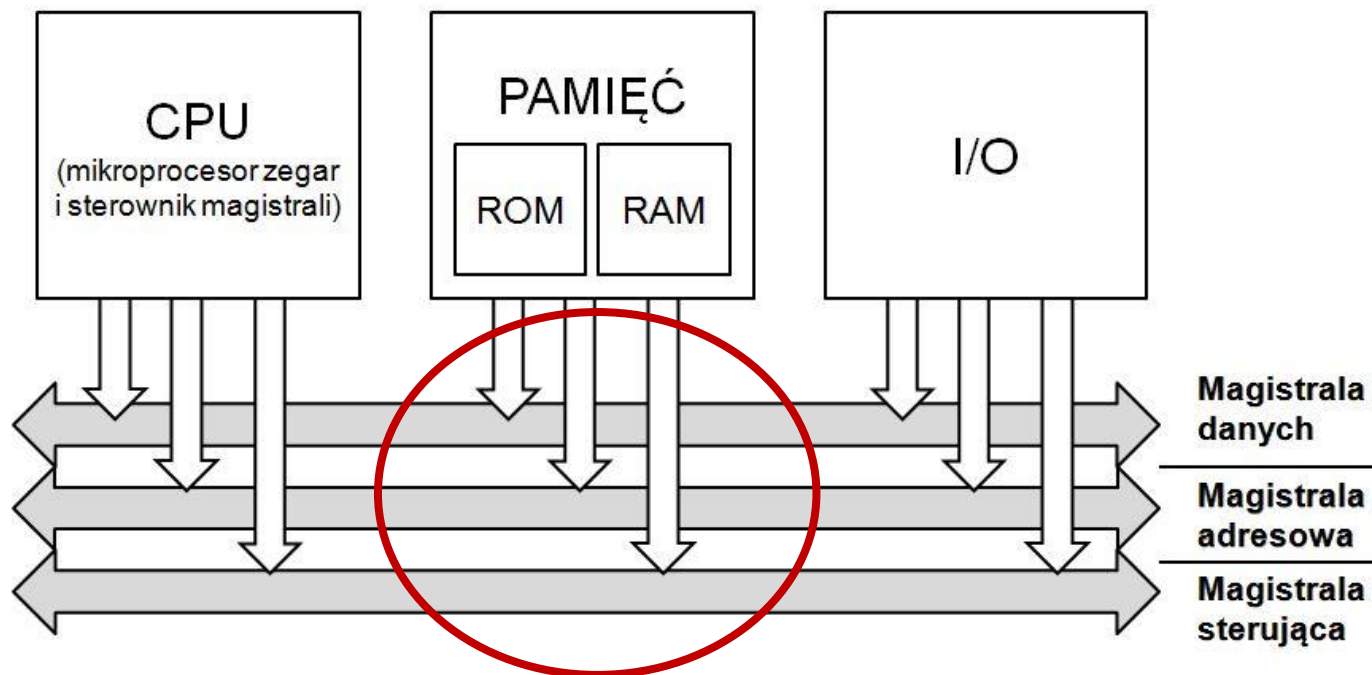


Schemat blokowy mikroprocesora (II)

- ✓ Zadaniem części sterującej jest pobieranie rozkazów z pamięci, dekodowanie ich, przygotowanie argumentów i generowanie sygnałów sterujących mikrooperacjami w fazie wykonania.
- ✓ Układ sterujący może być zrealizowany na dwa sposoby - jako sterowanie mikroukładowe (*hardwired control*) lub sterowanie mikroprogramowane (*microprogrammed control*).

Zadaniem CPU oprócz przetwarzania informacji jest także sterowanie pracą pozostałych układów systemu.

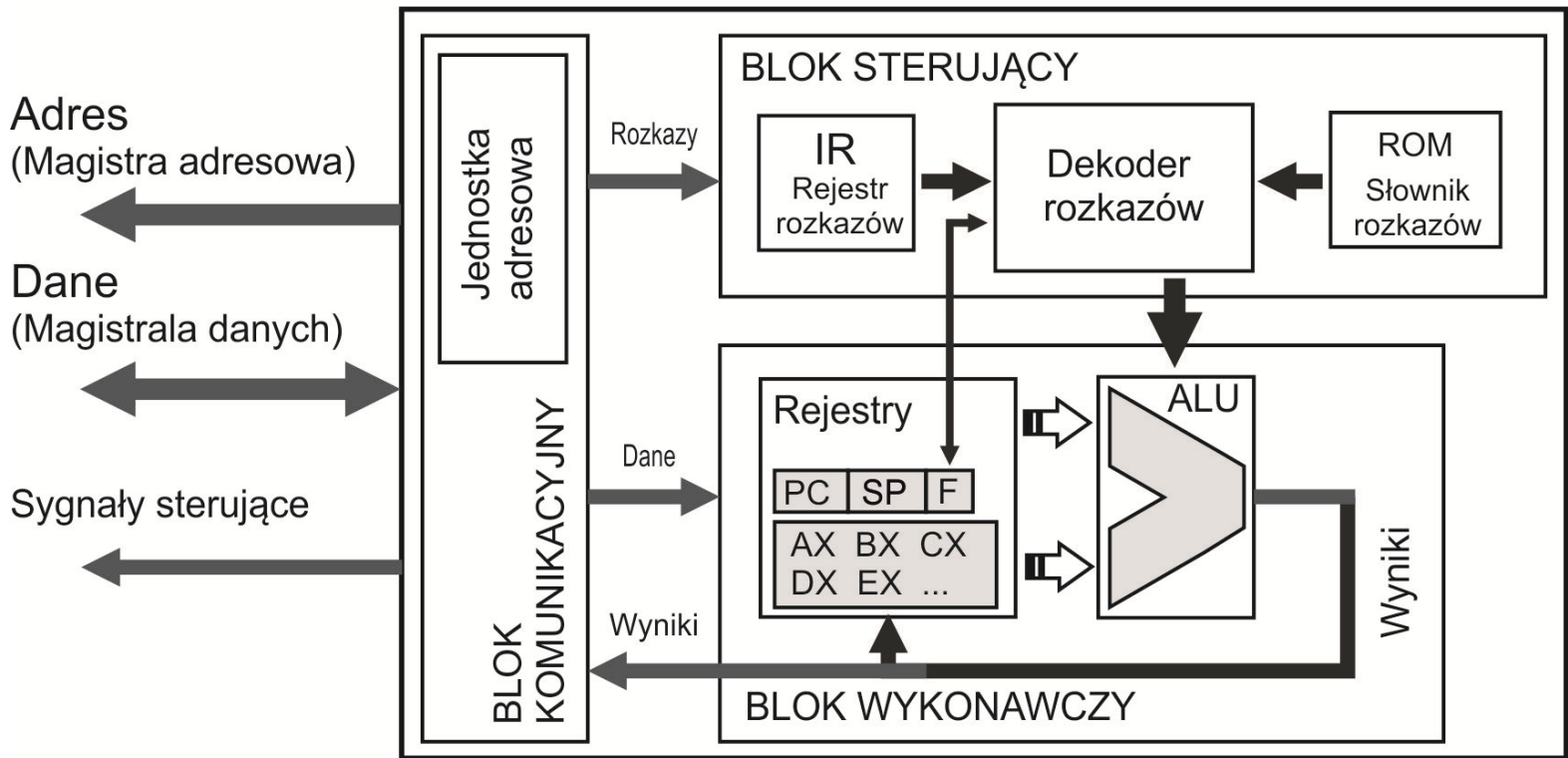
- ✓ **Wszystkie działania i operacje w systemie są sterowane lub zainicjowane przez procesor.** Rodzaj tych działań uzależniony jest od ciągu instrukcji dostarczonych do mikroprocesora nazywanych programem.
- ✓ Tak więc, każde działanie wykonane przez system jest wynikiem działania programu lub jego fragmentu.
- ✓ Program musi być przechowywany w pamięci o krótkim czasie dostępu i dostępie swobodnym (pamięć półprzewodnikowa). Pamięci masowa nie nadają się - mają zbyt długi czas dostępu i dostęp sekwencyjny.



Sięgając do modelu Von Neumana (z 1 wykładu) widzimy, że procesor komunikuje się z resztą systemu za pomocą 3 magistral:

1. Magistrali pamięci,
2. Magistrali danych
3. Magistrali sterującej

Schemat blokowy mikroprocesora (I)



- Schemat z wcześniejszego slajdu rozszerzony został o dostęp do magistral.
- Obsługę magistral realizuje blok komunikacyjny. Najważniejszą jego częścią jest jednostka adresowania przeliczająca względne adresy w programie na fizyczną adresację pamięci RAM

Definicja:

Rozkazem nazywamy najprostszą operację – w programowaniu, taką której wykonania program może zażądać od procesora

Definicja:

Lista rozkazów to pełny zestaw instrukcji maszynowych jakie może wykonać procesor

- ✓ Każdy rozkaz składa się z kilku pól.
- ✓ Jedno z nich występuje zawsze i nosi nawet pola kodu operacji. Kod ten definiuje funkcję rozkazu, czyli czynności jakie należy wykonać.
- ✓ Pozostałe pola zawierają argumenty operacji.
- ✓ Liczba tych pól zależy od rodzaju operacji, jakiej odpowiada rozkaz.
- ✓ W rozkazach bez argumentów pola dodatkowe nie występują.

Definicja:

Sposób rozmieszczenia wymienionych elementów w słowie lub słowach komputera nazywamy **formatem rozkazu**.

Kod rozkazu

Kod rozkazu	Argument
-------------	----------

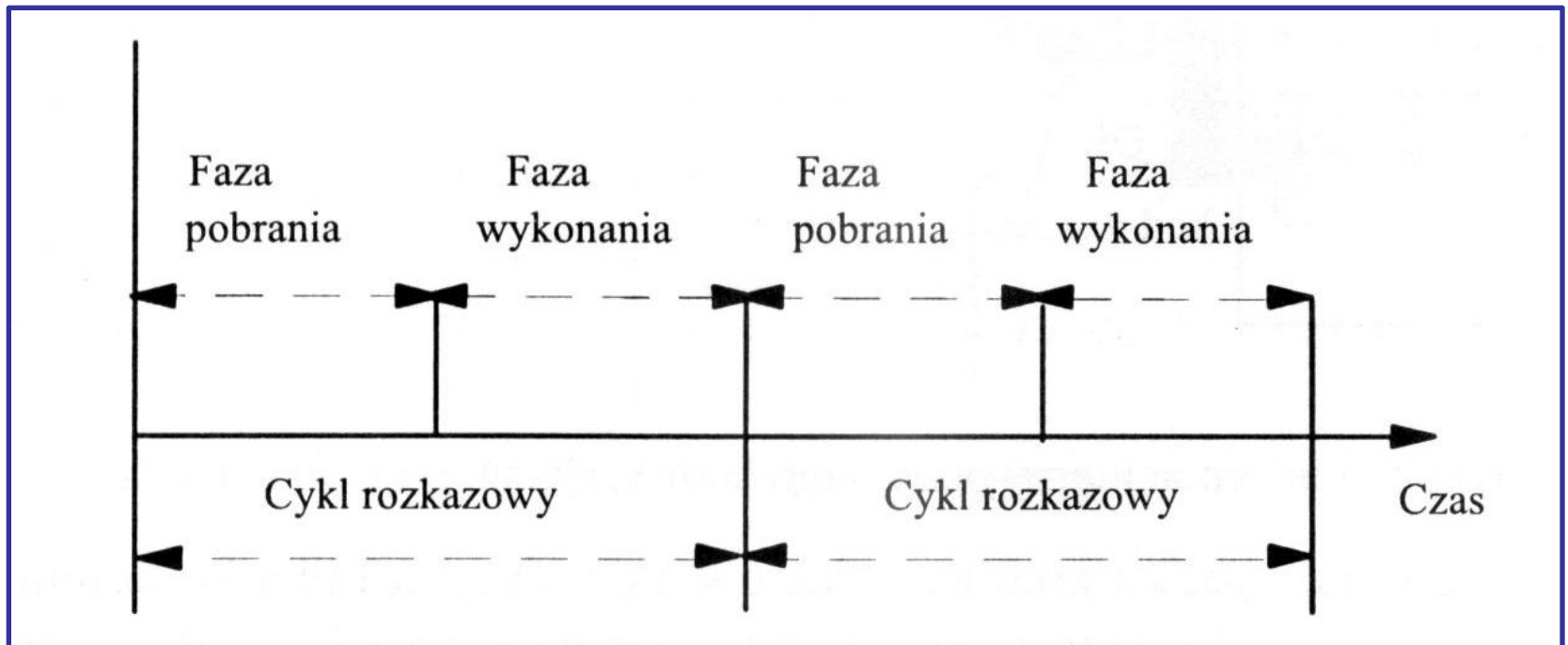
Kod rozkazu	Argument 1	Argument 2
-------------	------------	------------

Kod rozkazu	Argument 1	...	Argument n
-------------	------------	-----	------------

W polach oznaczonych jako “argument” mogą znajdować się:

- wartości argumentów (adresowanie natychmiastowe),
- nazwy rejestrów (adresowanie rejestrowe),
- informacje potrzebne do wyznaczenia adresu efektywnego.

Cykl rozkazowy procesora obejmuje dwie powtarzające się czynności:



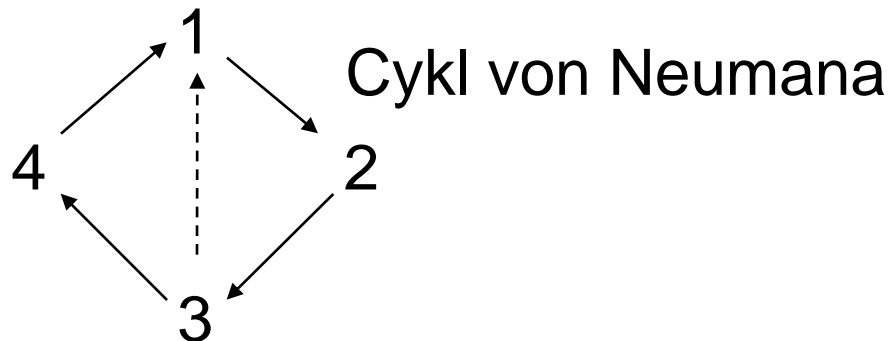
Kolejne etapy fazy pobierania i wykonywania:

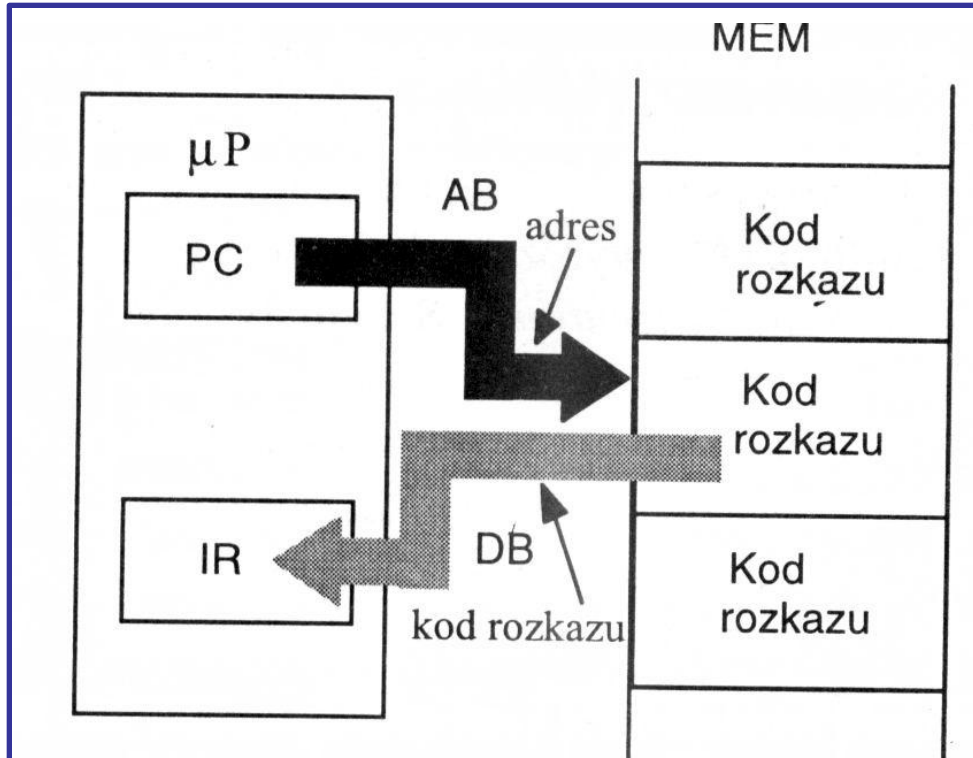
Faza pobrania:

1. Podanie zawartości licznika rozkazów PC na magistralę adresową AB
2. Wczytanie zawartości zaadresowanej komórki do rejestru rozkazów (IR)
3. Zwiększenie zawartości licznika rozkazów

Faza wykonania:

4. Zdekodowanie kodu rozkazu i wytworzenie sygnałów sterujących realizujących rozkaz





Źródło: Wojtuszkiewicz Krzysztof - *Urządzenia techniki komputerowej*, MIKOM

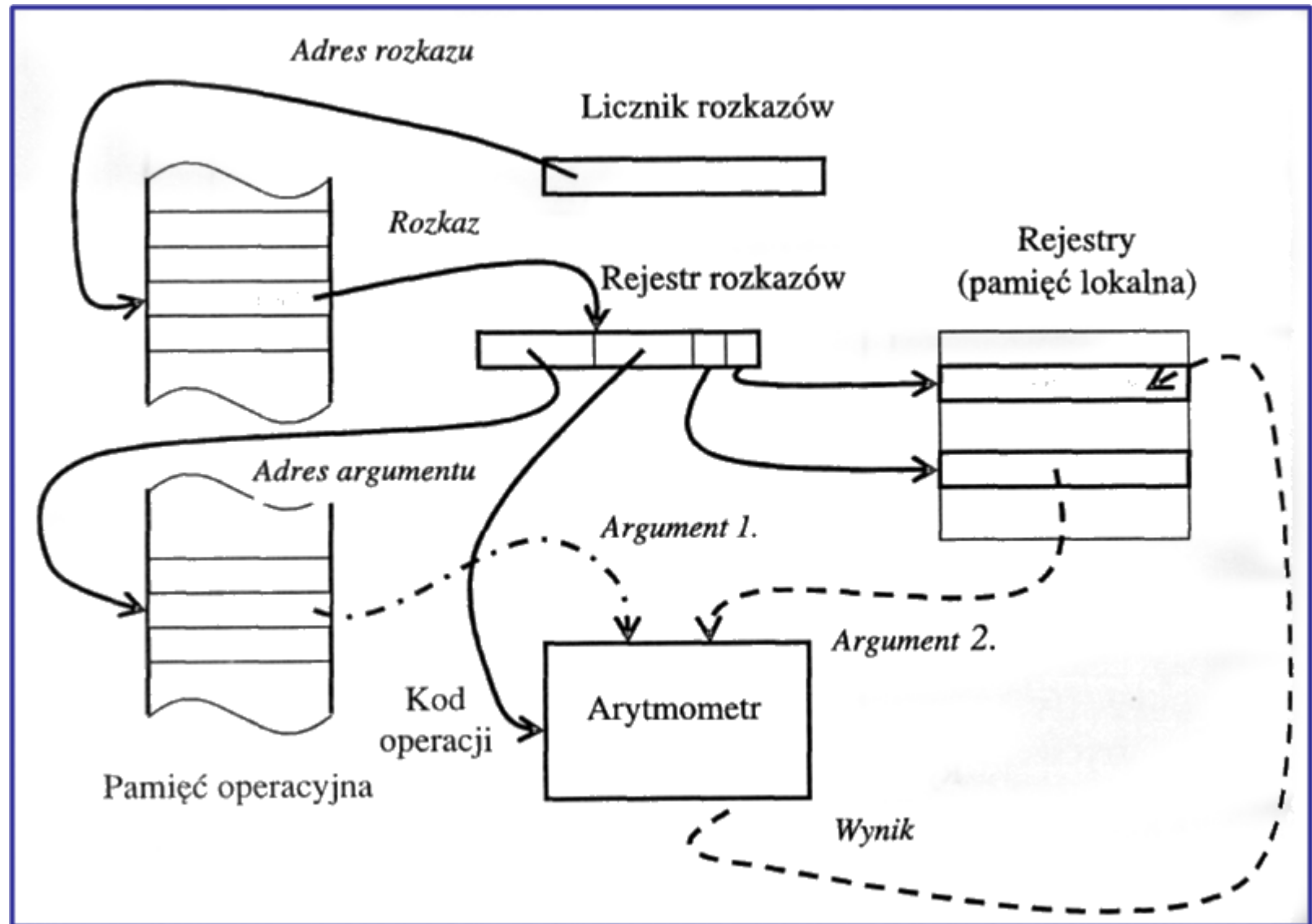
Faza pobrania:

1. Podanie zawartości licznika rozkazów PC na magistralę adresową AB
2. Wczytanie zawartości zaadresowanej komórki do rejestru rozkazów (IR)
3. Zwiększenie zawartości licznika rozkazów

Faza wykonania:

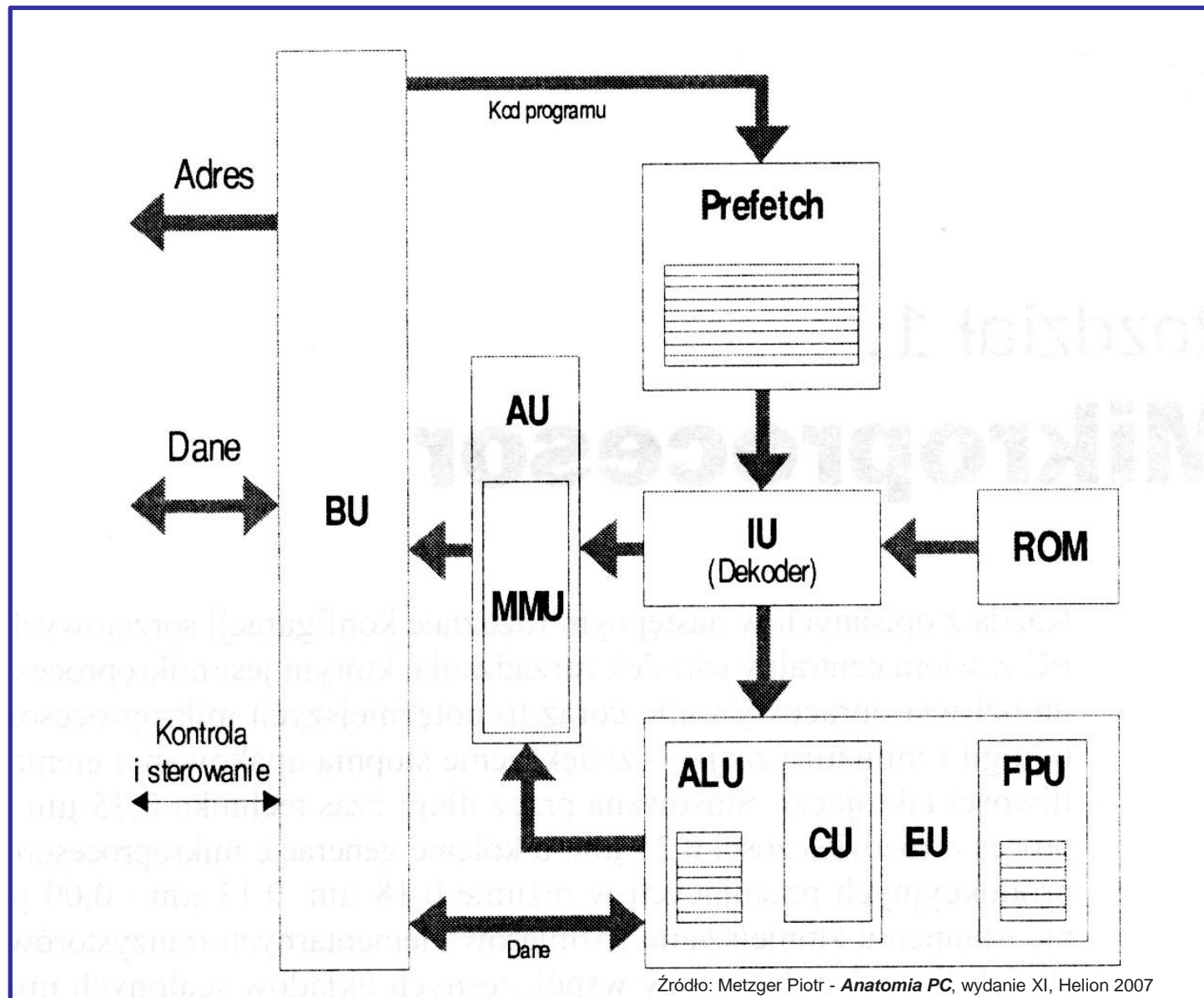
4. Zdekodowanie kodu rozkazu i wytworzenie sygnałów sterujących realizujących rozkaz

1. Rozkaz jest odczytywany z pamięci na podstawie adresu w tzw. **liczniku rozkazów** (PC, *program counter* lub IP, *instruction pointer*)
2. **Rozkaz wpisywany do rejestru rozkazów**, w którym przebywa aż do zakończenia cyklu rozkazowego. Poszczególne części rejestru rozkazów, odpowiadające polom słowa rozkazowego, są dekodowane określając dalszy przebieg cyklu.
3. **Wykrywana jest długość pobranego rozkazu** i o te wielkość zwiększa się licznik rozkazów, wskazując od tej pory adres kolejnego rozkazu. Ten mechanizm zapewnia sekwencyjne działanie komputera - rozkazy są pobierane i wykonywane w takiej kolejności, w jakiej są umieszczone w pamięci, czyli tak jak były zapisane w programie.
4. Jedynie rozkazy sterujące (skoki) mogą w czasie wykonania, a więc przed pobraniem następnego rozkazu, zmienić zawartość PC.

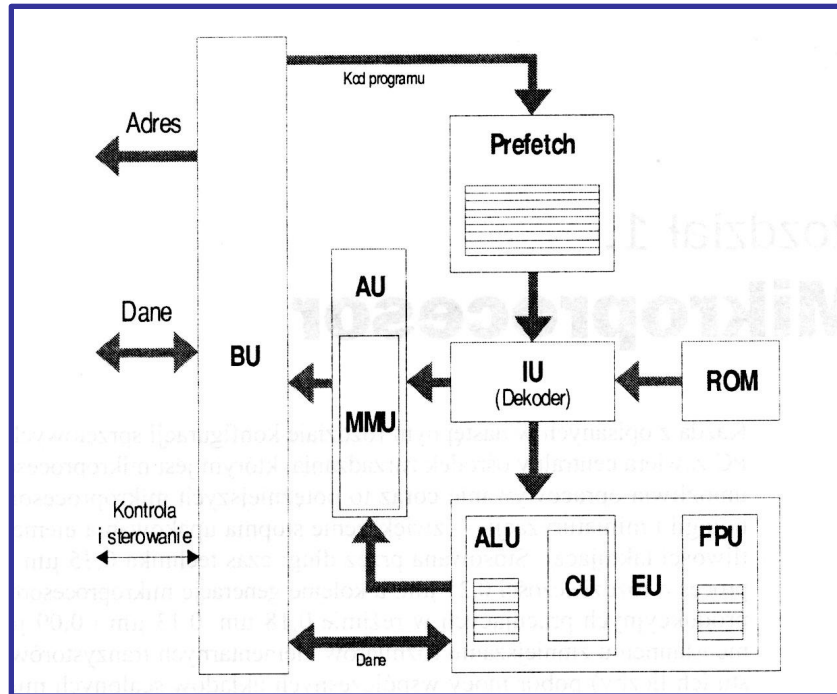


1. Adres zawarty w liczniku rozkazów podawany jest do pamięci.
2. Zawartość zaadresowanej komórki przepisywana jest do rejestru rozkazów.
3. Przykładowy rozkaz (z rys. na poprzednim slajdzie) zawiera:
 - kod rozkazu.
 - adres pierwszego z argumentów (adres w pamięci RAM),
 - adres (nazwę) rejestru w, którym znajduje się drugi argument,
 - adres (nazwę) rejestru do którego ma być zapisany wynik.
4. Argumenty pobrane z RAM-u i rejestru trafiają do ALU (arytmometru)
5. Wynik zapisywany jest w rejestrze

Schemat blokowy mikroprocesora (III)



Schemat blokowy mikroprocesora (III)



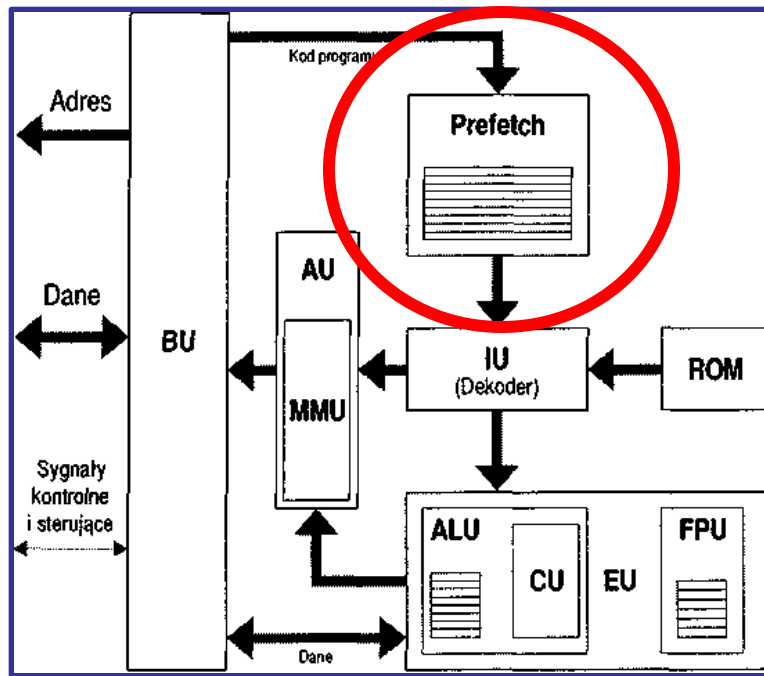
BU (Bus Unit) - wyizolowany blok komunikacyjny odpowiadający za współpracę z pamięcią

Prefeth – kolejka oczekujących na wykonanie danych

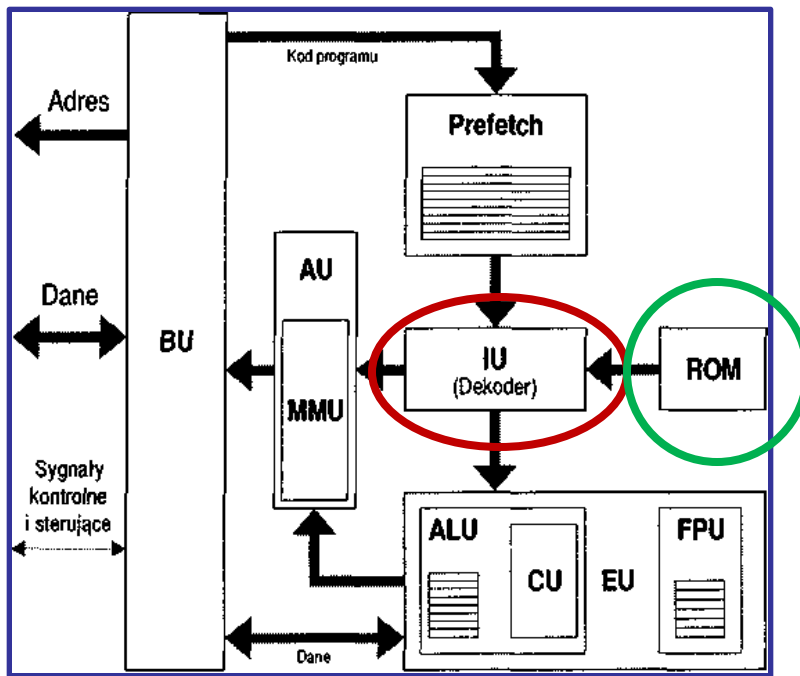
IU (instruction Unit) – blok dekodera – korzysta on z pamięci ROM w której znajduje się słownik tłumaczący przyjmowane kody rozkazów na sekwencje instrukcji wewnętrznych mikroprocesora

EU (Execution Unit) – układ wykonawczy realizujący operacje określone przez kod rozkazu.

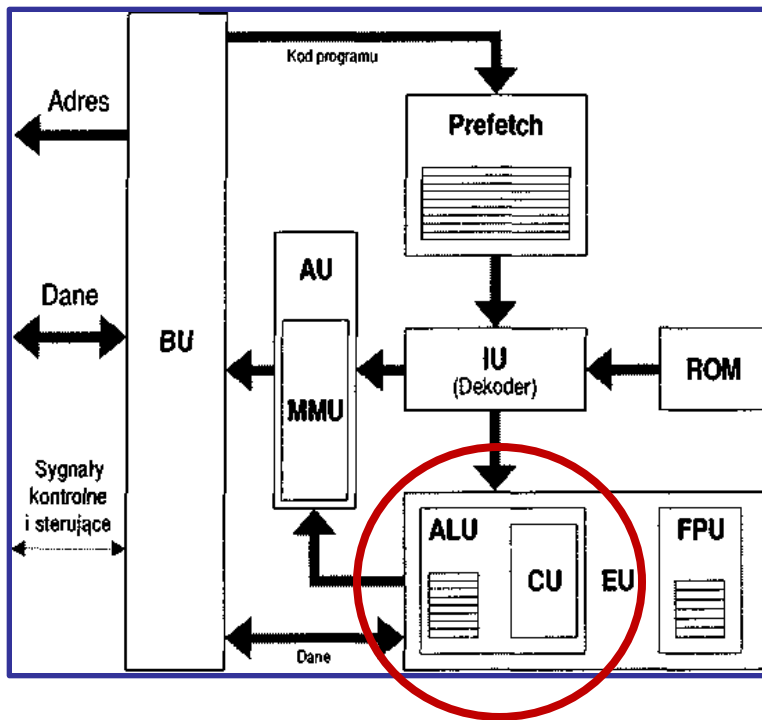
Znaczna część programu podlega obróbce w module **ALU (Arithmetic-Logic Unit)** sterowanym z bloku **CU (Control Unit)**. Dla przyspieszenia pracy procesora operacje zmiennoprzecinkowe przekazywane są do wyspecjalizowanej jednostki **FPU (Floating Point Unit)**



Konieczność zapewnienia płynnego funkcjonowania procesora wymaga, by dane do wykonania (kod programowy) pobierane były w większych porcjach i gromadzone w **kolejce Prefetch**, gdzie oczekują na wykonanie.

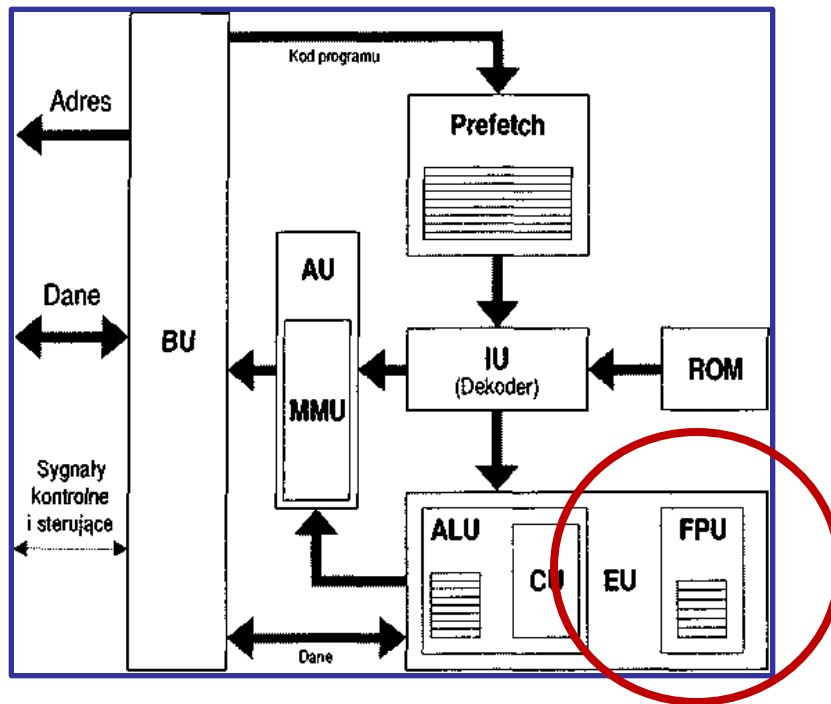


Ich odtwarzania odbywa się w **bloku dekodera (IU - Instruction Unit)**. Praca tego układu wspomagana jest często przez obszerną **podręczną pamięć stałą (ROM)**, w której zawarty jest słownik tłumaczący przyjmowane kody rozkazowe na sekwencje ukrywających się pod nimi operacji.

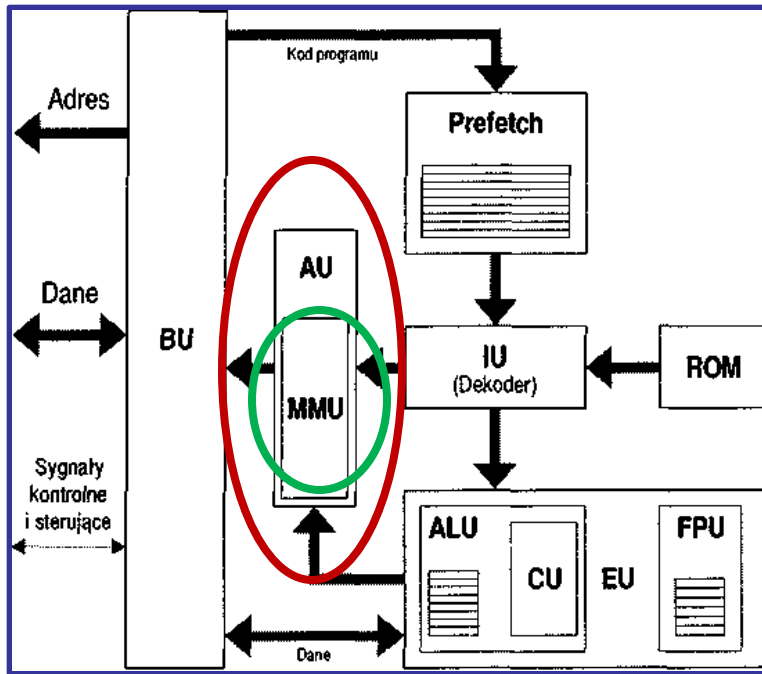


Rozkodowane instrukcje przekazywane są do właściwego **układu wykonawczego (EU - Execution Unit)**, gdzie realizowana jest operacja określona danym kodem rozkazowym. Znaczna część powszechnie używanego kodu pracuje na liczbach stałoprzecinkowych (*Integer*) i podlega obróbce w module ALU (*Arithmetic-Logic Unii*) sterowanego z bloku CU (*Control Unit*).

Schemat blokowy mikroprocesora



Jeśli jednak rozkaz dotyczył obiektów zmiennoprzecinkowych przekazuje się go do wyspecjalizowanej **jednostki zmiennoprzecinkowej (FPU - Floating Point Unit)**.



Rozkazy posługują się zwykle pewnymi argumentami (parametry funkcji, na przykład składniki przy dodawaniu), które również trzeba pobrać z pamięci operacyjnej.

Często wymaga się, by wynik operacji przesłać pod określony adres. Obsługę przesyłania bierze na siebie **jednostka adresowania (AU — Addressing Unit)**. Względy natury technicznej (omówione na poprzednim wykładzie) powodują, iż dostęp do pamięci operacyjnej wymaga pewnych dodatkowych nakładów, których realizacji poświęca się jednostkę **zarządzania pamięcią (MMU - Memory Management Unit)**.

- ✓ Początkowo listy rozkazów były niewielkie i zawierały operacje niezbędne do wykonywania działań arytmetycznych i do tworzenia elementarnych struktur progra-mowych (skoki, pętle, rozgałęzienia warunkowe, podprogramy). Później dołączono rozkazy działające na znakach, a ostatnio rozkazy ukierunkowane na zastosowania w obsłudze operacji multimedialnych.
- ✓ Po wprowadzeniu języków symbolicznych (assemblerów), które stały się głównym narzędziem programowania systemowego (systemy operacyjne, kompilatory), nowe listy rozkazów wyposażano w coraz bardziej złożone instrukcje ułatwiające programowanie. Celem stało się zniwelowanie luki semantycznej między assemblerami, a więc i językami maszynowymi, a językami algorytmicznymi wysokiego poziomu.



CISC

- ✓ Architektura CISC bazuje na sterowaniu mikroprogramowym.
- ✓ Pojedynczy, złożony rozkaz rozbijany był wewnątrz procesora na kilka prostszych rozkazów, a jego wykonanie trwa kilka lub nawet kilkanaście taktów zegara.
- ✓ Sterowanie mikroprogramowe wymaga wyposażenia procesora w skomplikowaną jednostkę sterującą.

CISC

Procesory ze złożoną listą instrukcji są określane jako CISC (ang. *Complex Instruction Set Computer*). Charakteryzują się:

- dużą liczbą rozkazów o różnych długościach;
- dużą liczbą trybów adresowania;

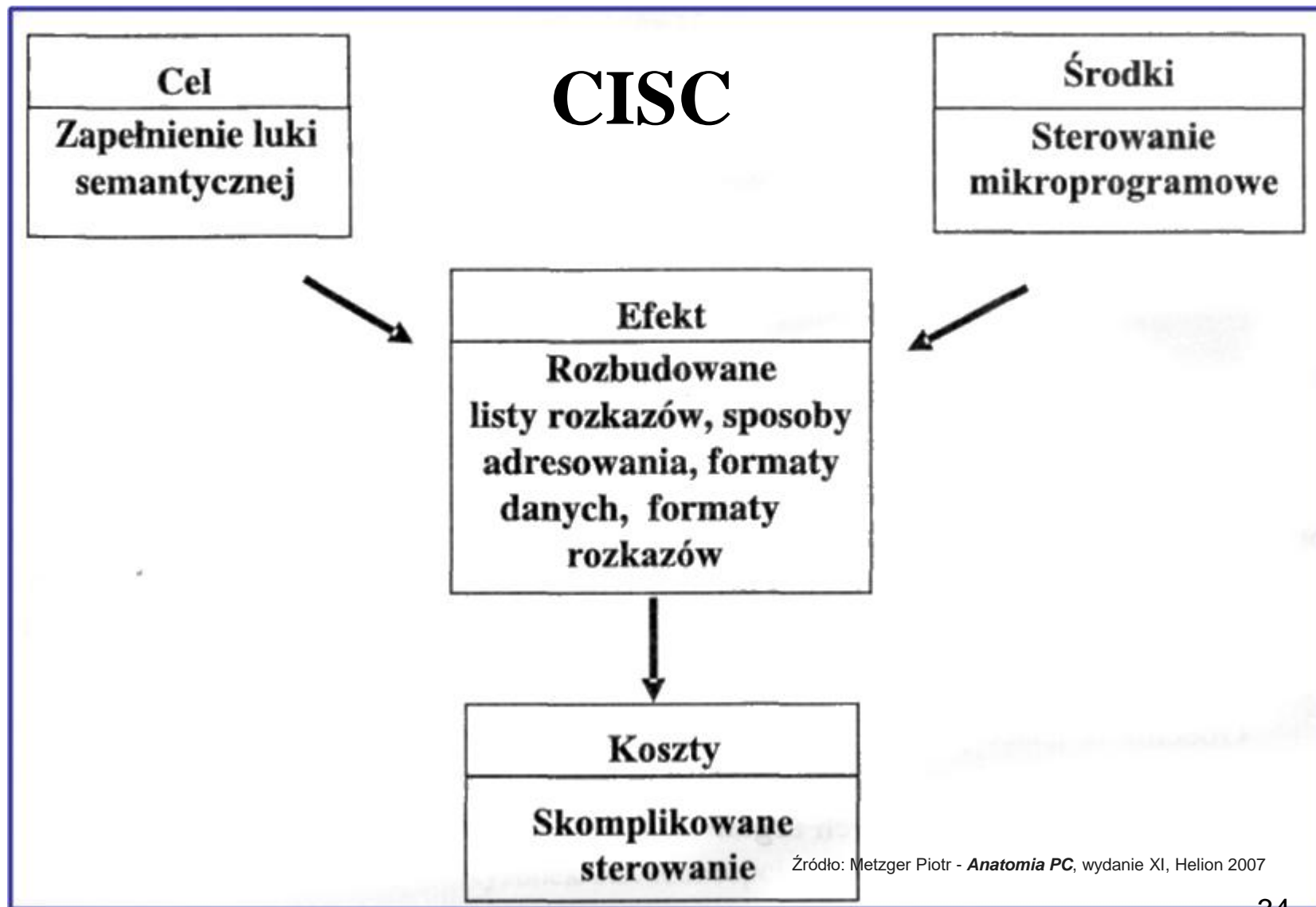
Zalety:

- ✓ łatwością tworzenia oprogramowania

Wady:

- ✓ bogactwo rozkazów maszynowych prowadziło do tego, że rozkazy były długie i różnej długości, co komplikowało strukturę i działanie układu sterującego

AB Koncepcja CISC



RISC

Procesory ze zredukowaną (uproszczoną) listą rozkazów – RISC (ang. *Reduced Instruction Set Computer*) charakteryzują się:

- niewielką liczbą rozkazów
- małą liczbą trybów adresowania
- prostą i szybką jednostką sterującą

Zalety:

- ✓ prosta (a więc tania) jednostka sterująca;
- ✓ możliwość zwiększania taktowania procesora;
- ✓ przetwarzanie potokowe;

Wady

- ✓ trudnością w tworzeniu oprogramowania
- ✓ duże obciążenie magistral pamięciowych

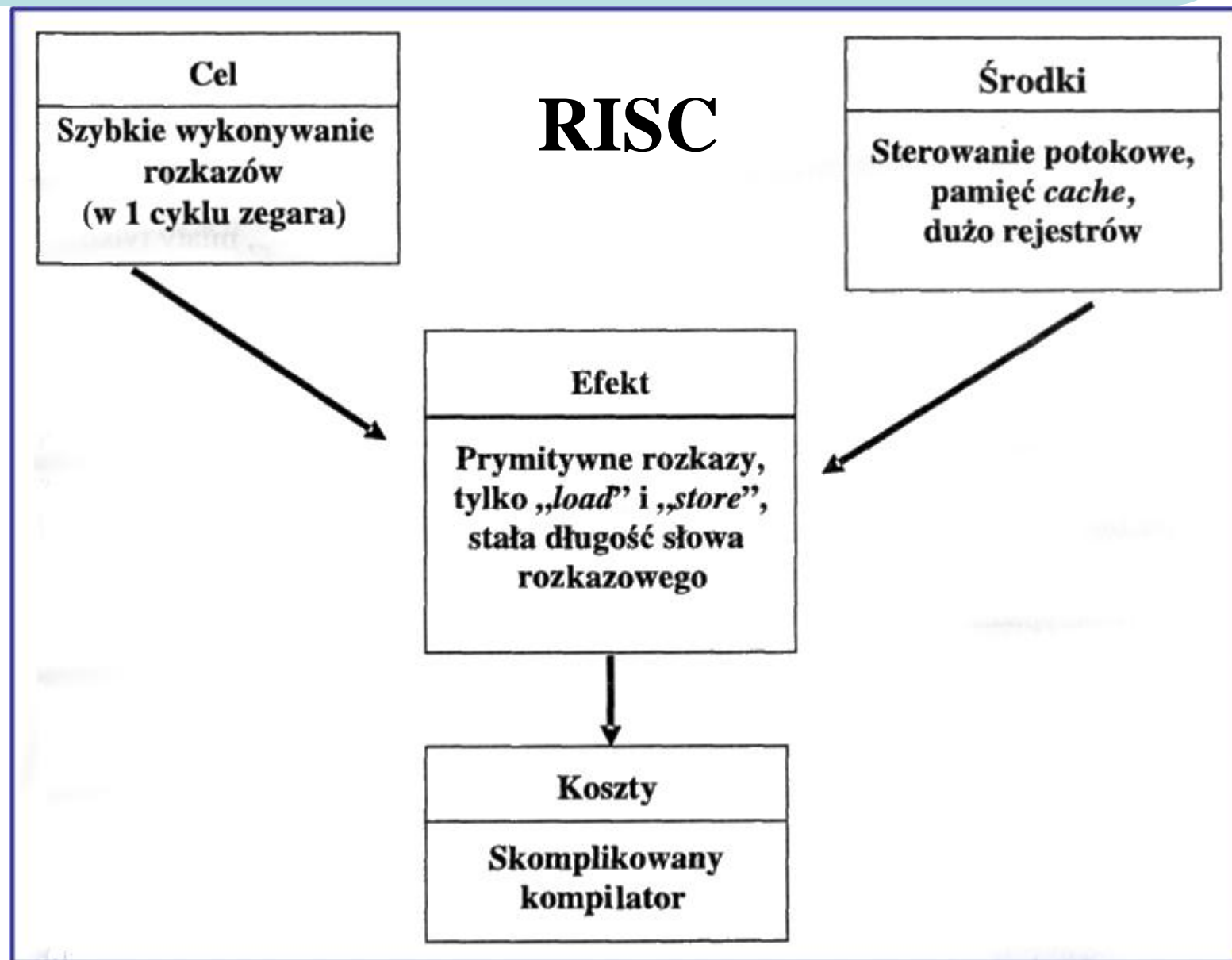
RISC

- ✓ Szybkość przetwarzania instrukcji w technologii RISC osiągnięto poprzez sterowanie mikroukładowe.
- ✓ W procesorze dysponującym niewielką liczbą prostych rozkazów nie zachodzi, w przeciwieństwie do architektury CISC, konieczność rozbijania ich na prostsze operacje składowe.
- ✓ Realizacją każdego rozkazu może zajmować się wyspecjalizowany układ logiczny bloku wykonawczego.

RISC

- ✓ Procesor zbudowany w technologii RISC wykonuje w krótkim czasie bardzo dużo prostych rozkazów. Rozkazy te (wraz z ich argumentami) muszą być dostarczane w odpowiednim tempie. Wymaga to dużej przepustowości magistrali łączącej procesor z pamięcią operacyjną.
- ✓ Problem ten nie występował w technologii CISC, gdzie pojedynczy rozkaz był wczytany i wykonywany przez kilkanaście taktów zegara. Procesory RISC są natomiast w stanie pobrać więcej niż jeden rozkaz w pojedynczym takcie zegara (tak, to możliwe – superskalarność).
- ✓ Dla zapewnienia odpowiednio szybkiej komunikacji z pamięciami RAM współczesne procesory wyposażone są w dwa lub trzy poziomy pamięci podręcznej CACHE (L1, L2 i L3).

AB Koncepcja RISC





RISC czy CISC?

- ✓ Rozwój rynku mikroprocesorów w ciągu ostatnich 20 lat pokazał, że koncepcja RISC daje lepsze efekty mierzone stosunkiem wydajności procesora do jego kosztu.
- ✓ Współczesne procesory rodziny x86 produkowane przez firmy Intel oraz AMD mają listy rozkazów dużo dłuższe, niż starsze procesory wykonane w technologii CISC. Jednocześnie procesory te posiadają większość kluczowych cech technologii RISC (mała liczba trybów adresowania, podział pamięci Cache na blok danych i blok instrukcji, możliwość pracy potokowej i superskalarnej).
- ✓ **Procesory te łączą cechy obu typów architektury.** Długa lista rozkazów zapewnia zgodność ze starszym oprogramowaniem, a szybkie, oparte na koncepcji RISC, przetwarzanie programu wewnątrz procesora zwiększa jego moc obliczeniową.



Koncepcja RISC

	Architektura	
	CISC	RISC
Liczba rozkazów:	duża	mała
Format (długość) rozkazów:	zmienna	stała
Rodzaj sterowania jednostką wykonawczą:	mikroprogramowe	mikroukładowe
Budowa jednostki sterującej:	skomplikowana	prosta
Konieczność stosowania wysokowydajnych magistral pamięci:	NIE	TAK
Konieczność stosowania pamięci Cache:	NIE	TAK
Możliwość przetwarzania potokowego:	NIE	TAK
Możliwość przetwarzania superskalarnego:	NIE	TAK



Literatura:

1. Metzger Piotr - **Anatomia PC**, wydanie XI, Helion 2007
2. Wojtuszkiewicz Krzysztof - **Urządzenia techniki komputerowej, część I: Jak działa komputer**, MIKOM, Warszawa 2000
3. Wojtuszkiewicz Krzysztof - **Urządzenia techniki komputerowej, część II: Urządzenia peryferyjne i interfejsy**, MIKOM, Warszawa 2000
4. Komorowski Witold - **Krótki kurs architektury i organizacji komputerów**, MIKOM Warszawa 2004
5. Gook Michael - **Interfejsy sprzętowe komputerów PC**, Helion, 2005