Architektura komputerów

Wykład 3

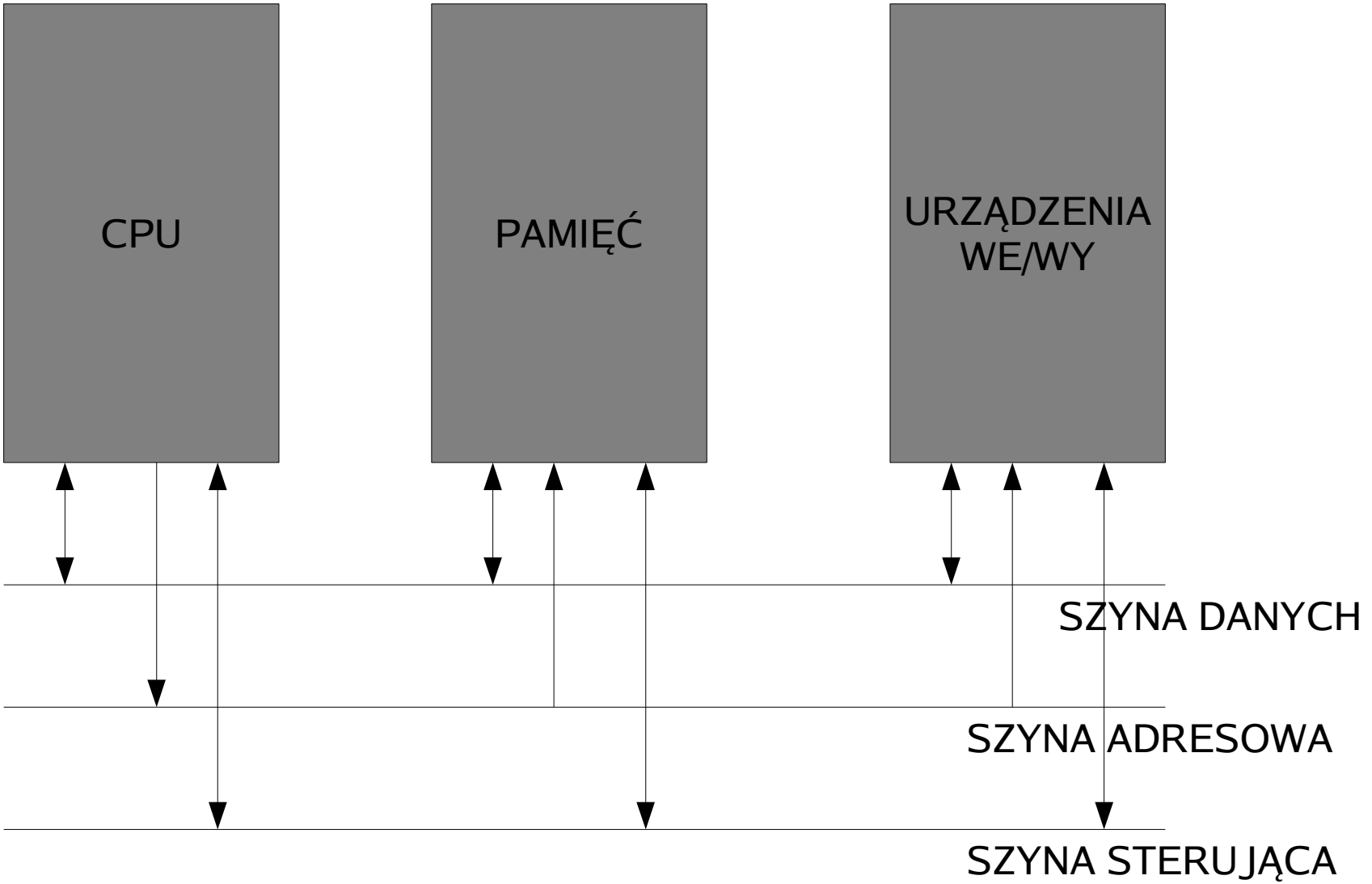
Jan Kazimirski

Podstawowe elementy komputera. Procesor (CPU)

Plan wykładu

- Podstawowe komponenty komputera
- Procesor CPU
- Cykl rozkazowy
- Typy instrukcji
- Stos
- Tryby adresowania

Podstawowe elementy komputera



CPU

- CPU – Central Processing Unit
- Wykonuje rozkazy programu komputerowego umieszczonego w pamięci komputera
- Elementy
  - Jednostka arytmetyczno-logiczna (ALU)
  - Rejestry sprzętowe
  - Układy sterujące

ALU

- Jednostka arytmetyczno-logiczna
- Wykonuje podstawowe operacje matematyczne i logiczne
- Operuje na binarnych liczbach całkowitych – bez znaku, lub w kodzie uzupełnienia do dwóch (czasami również inne kodowania np. BCD)
- Zakres liczb zależy od tzw. szerokości słowa maszynowego – 8,16,32,64,128 bitów

FPU

- FPU – Floating Point Unit – jednostka obliczeń zmiennoprzecinkowych
- Wykorzystuje standard IEEE 754 zapisu zmiennopozycyjnego
- Formaty 32 lub 64 bity (inne stosowane do wyników pośrednich)
- Dawniej dostępna jako osobny układ, obecnie najczęściej zintegrowana

Zbiór instrukcji procesora

- Pełny zestaw rozkazów zrozumiałych przez procesor
- Postać binarna – kod maszynowy
- Postać wygodniejsza dla człowieka – zapis symboliczny – assembler
- Język niskiego poziomu – pełna kontrola nad postacią maszynową programu

Wykonywanie instrukcji maszynowej

- Pobranie instrukcji
- Dekodowanie instrukcji
- Wyliczenie adresu operandu i pobranie operandu
- Wykonanie operacji
- Wyliczenie adresu rezultatu i zapisanie rezultatu

Pobieranie i zapis danych

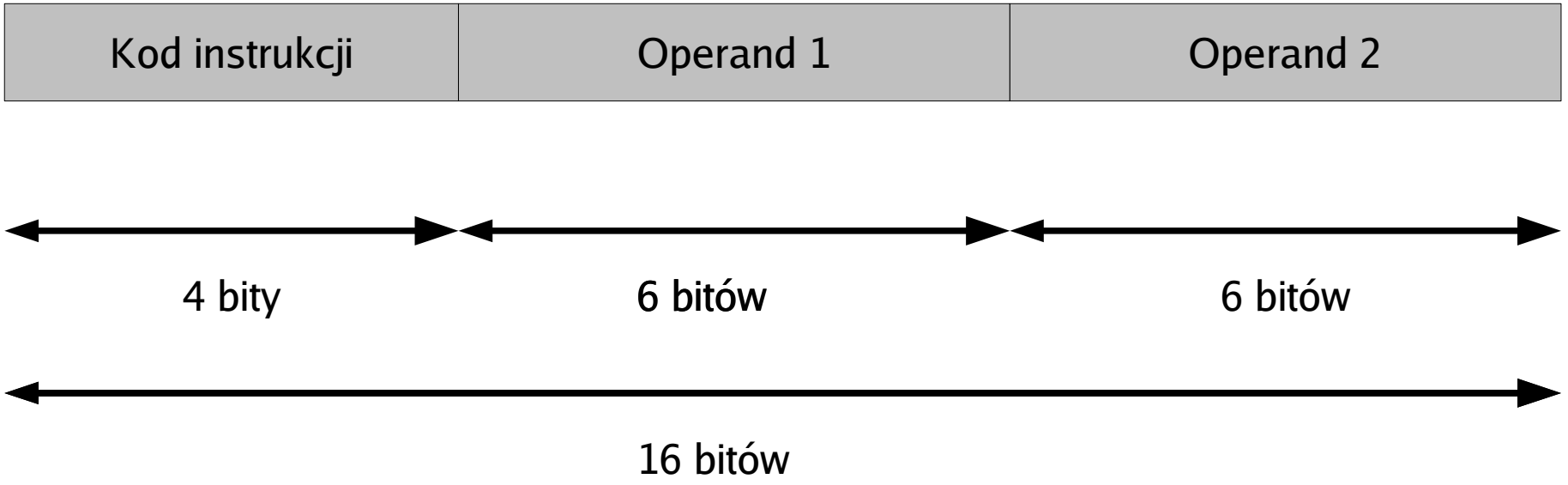
- Dane mogą być zapisywane i pobierane z:
  - Pamięci operacyjnej
  - Jednego z rejestrów procesora
  - Urządzenia zewnętrznego

Format instrukcji

- Każda instrukcja ma unikalny wzorzec bitowy
- Wzorce bitowe instrukcji są dekodowane po załadowaniu instrukcji do rejestru instrukcji (IR)
- Zapis symboliczny instrukcji: np. ADD, MOV, DIV, JMP itd.
- Zaawansowane assemblery pozwalają na symboliczny zapis położenia danych np.

  ADD R, Y

Przykład formatu instrukcji



Typy instrukcji

- Przetwarzanie danych (instrukcje arytmetyczne, logiczne, konwersji)
- Transfer danych (pamięć – rejestr, rejestr – rejestr, rejestr – stos)
- Instrukcje wejścia/wyjścia
- Instrukcje sterujące

Liczba operandów

- 0 operandów – wszystkie operandy w domyślnych lokacjach (np. na stosie)
- 1 operand – dla operacji dwuargumentowych drugi argument w domyślnej lokacji (np. Akumulator)
- 2 operandy – jeden z operandów jest jednocześnie lokacją rezultatu
- 3 i więcej operandów – złożone instrukcje CISC

Typy operandów

- Adresy (pamięć, rejestry)
- Liczby (całkowite, rzeczywiste, BCD)
- Znaki
- Dane logiczne (bity, flagi)

Instrukcje – przetwarzanie danych

- Instrukcje arytmetyczne – dodawanie, odejmowanie, mnożenie, dzielenie
- Operacje przesunięcia i rotacji (przesuwanie bitów)
- Operacje logiczne i bitowe (AND,OR,NOT,XOR)
- Operacje konwersji (np. Dwójkowa – dziesiętna, dwójkowa - BCD)

Instrukcje – transfer danych

- Ładowanie danych z pamięci do rejestrów i z rejestrów do pamięci
- Ładowanie danych bezpośrednich do rejestrów i pamięci.
- Przenoszenie danych w pamięci
- Operacje na stosie

Instrukcje – wejście/wyjście

- Realizowane za pomocą rozkazów maszynowych (IN, OUT)
- Realizowane za pomocą zwykłych przesłań (przestrzeń I/O mapowana w pamięci)
- Stosowanie kontrolerów DMA

Instrukcje - sterowanie

- Instrukcje skoków warunkowych i bezwarunkowych
- Instrukcje wywołania i powrotu z podprogramu
- Obsługa przerwań
- Instrukcje zarezerwowane dla systemu operacyjnego

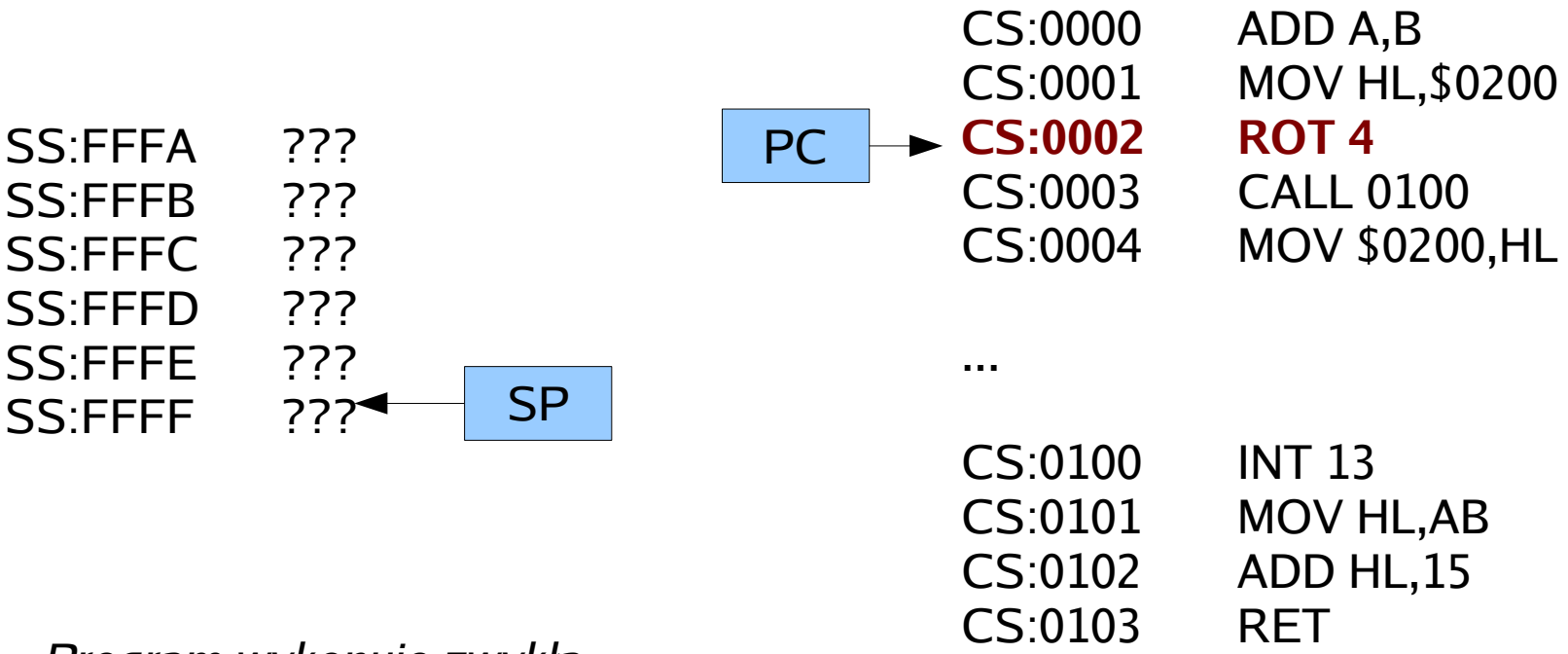
Stos

- Stosowany do realizacji wywołań podprogramów oraz tymczasowego przechowywania danych
- Realizacja sprzętowa i programowa:
  - Stos sprzętowy – osobne układy logiczne
  - Stos programowy – wydzielony obszar pamięci – segment stosu, wierzchołek stosu przechowywany w rejestrze.

Stos c.d.

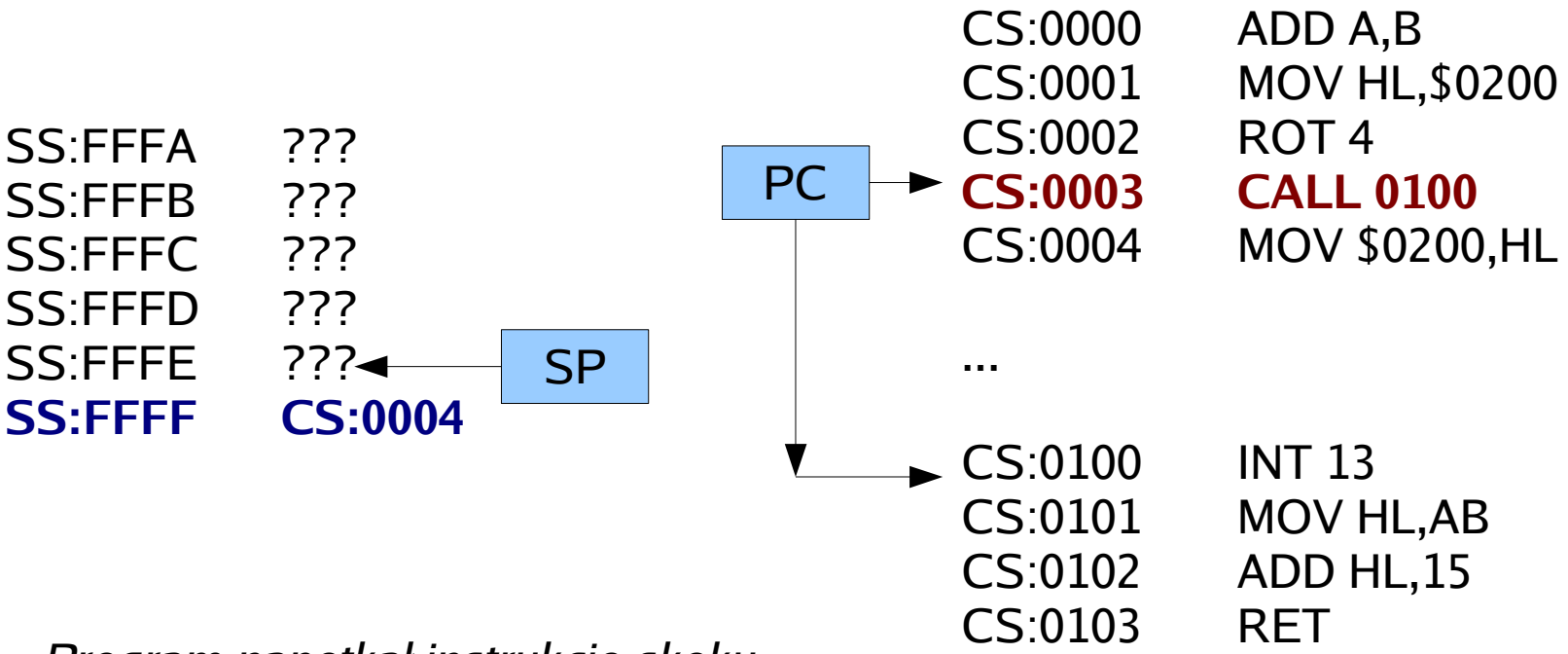
- Stos pozwala na realizację zagnieżdżonych skoków do podprogramów.
- Wykorzystywany jest również do przekazywania danych z i do podprogramów
- Na stosie można np. zachować zawartość rejestrów aby nie uległa zamazaniu w trakcie wykonywania podprogramu lub procedury obsługi przerwania.

Skok ze śladem



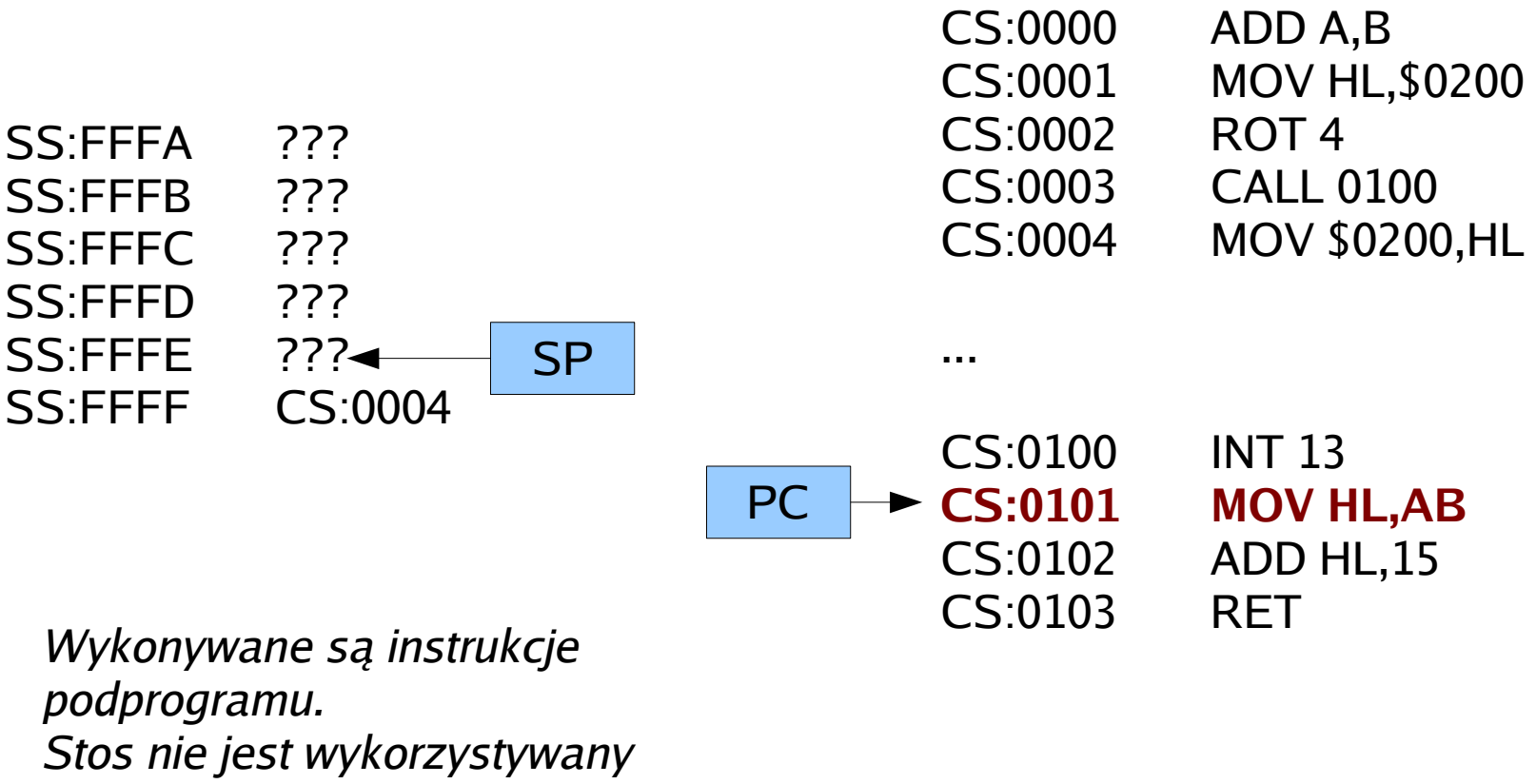
Program wykonuje zwykłą instrukcję, nie korzysta ze stosu

Skok ze śladem



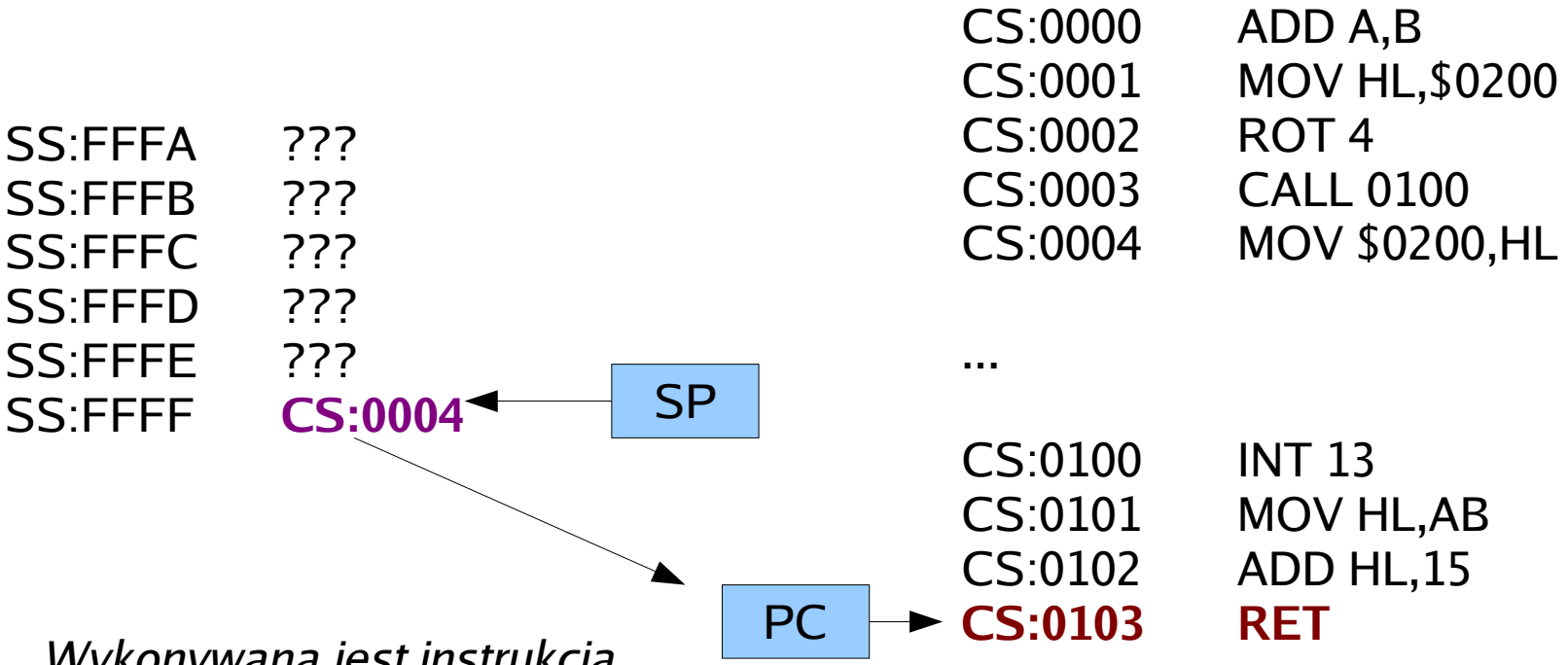
Program napotkał instrukcję skoku
Adres kolejnej instrukcji
umieszczany jest na stosie
Licznik programu ustawiany jest na
instrukcję podprogramu

Skok ze śladem



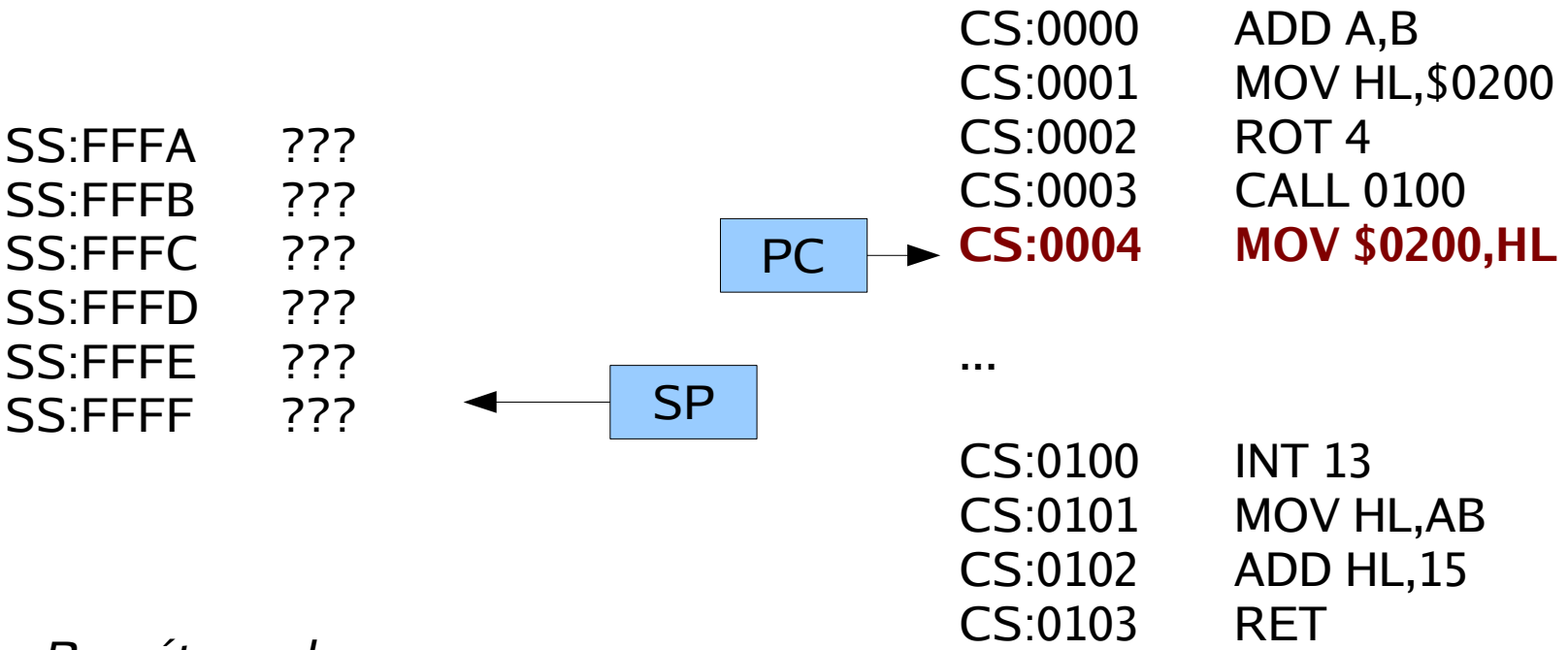
Wykonywane są instrukcje podprogramu.
Stos nie jest wykorzystywany

Skok ze śladem



Wykonywana jest instrukcja powrotu z podprogramu. Adres z wierzchołka stosu umieszczany jest w rejestrze PC Rejestr SP jest przesuwany (usunięcie adresu ze stosu)

Skok ze śladem



Powrót z podprogramu.
Wykonywana jest kolejna instrukcja programu głównego (następna za instrukcją wywołującą podprogram)

Adresowanie

- Sposoby wyznaczania położenia operandu
  - Natychmiastowe
  - Bezpośrednie
  - Pośrednie
  - Rejestrowe
  - Rejestrowe pośrednie
  - Indeksowe
  - Wykorzystujące stos

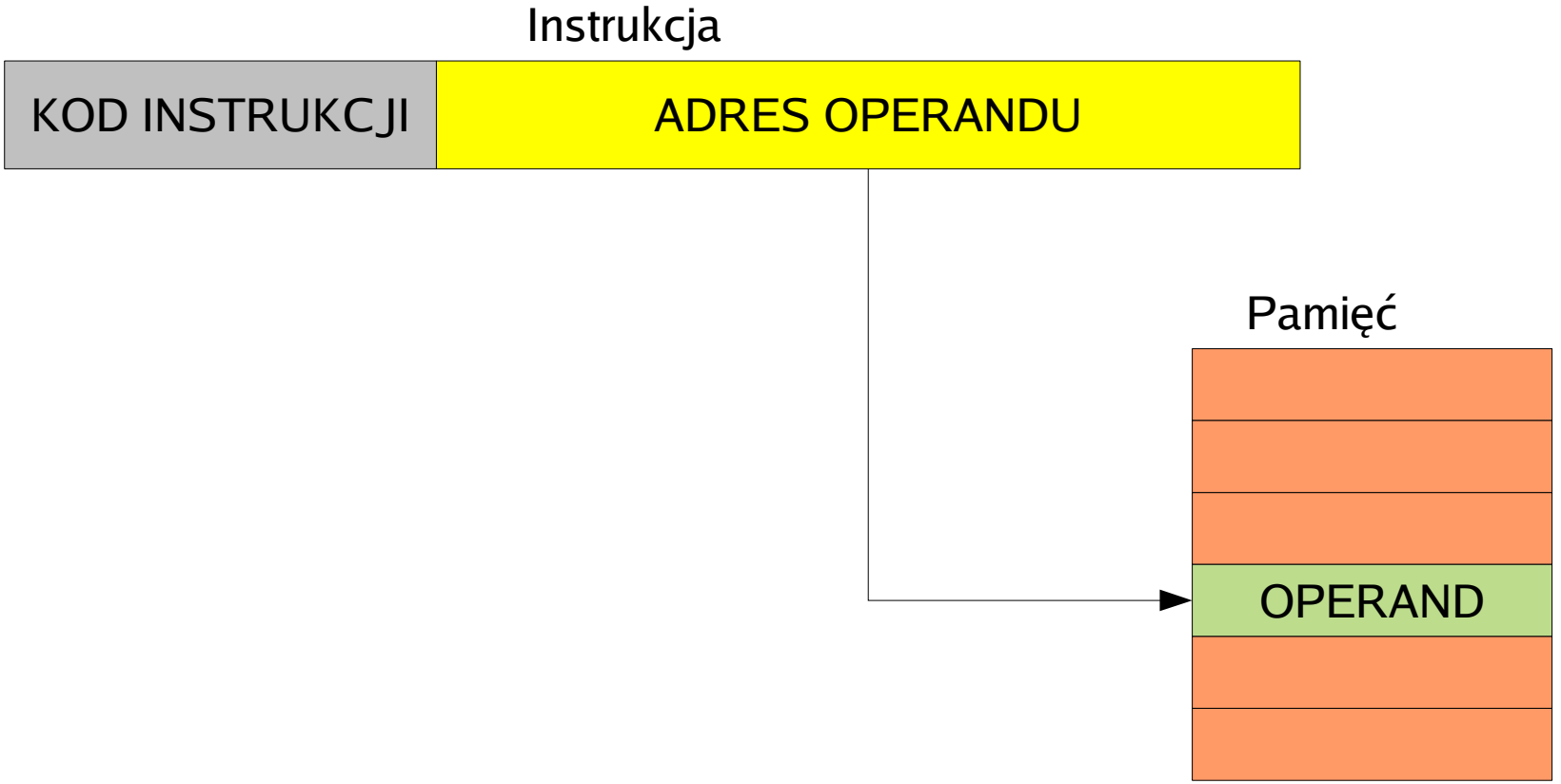
Adresowanie natychmiastowe

- Operand jest częścią instrukcji (np. ADD 5 – dodaj liczbę 5 do akumulatora)
- Brak dodatkowego dostępu do pamięci (wydajność)
- Ograniczony zakres i liczba operandów (długość instrukcji)
- Schemat instrukcji:

Adresowanie bezpośrednie

- Instrukcja zawiera adres pamięci z operandem
- Wymaga dodatkowego dostępu do pamięci
- Nie wymaga dodatkowych wyliczeń efektywnego adresu
- Ograniczony zakres adresu (rozmiar instrukcji)
- Przykład: ADD [100] – dodaj do akumulatora zawartość komórki o adresie 100

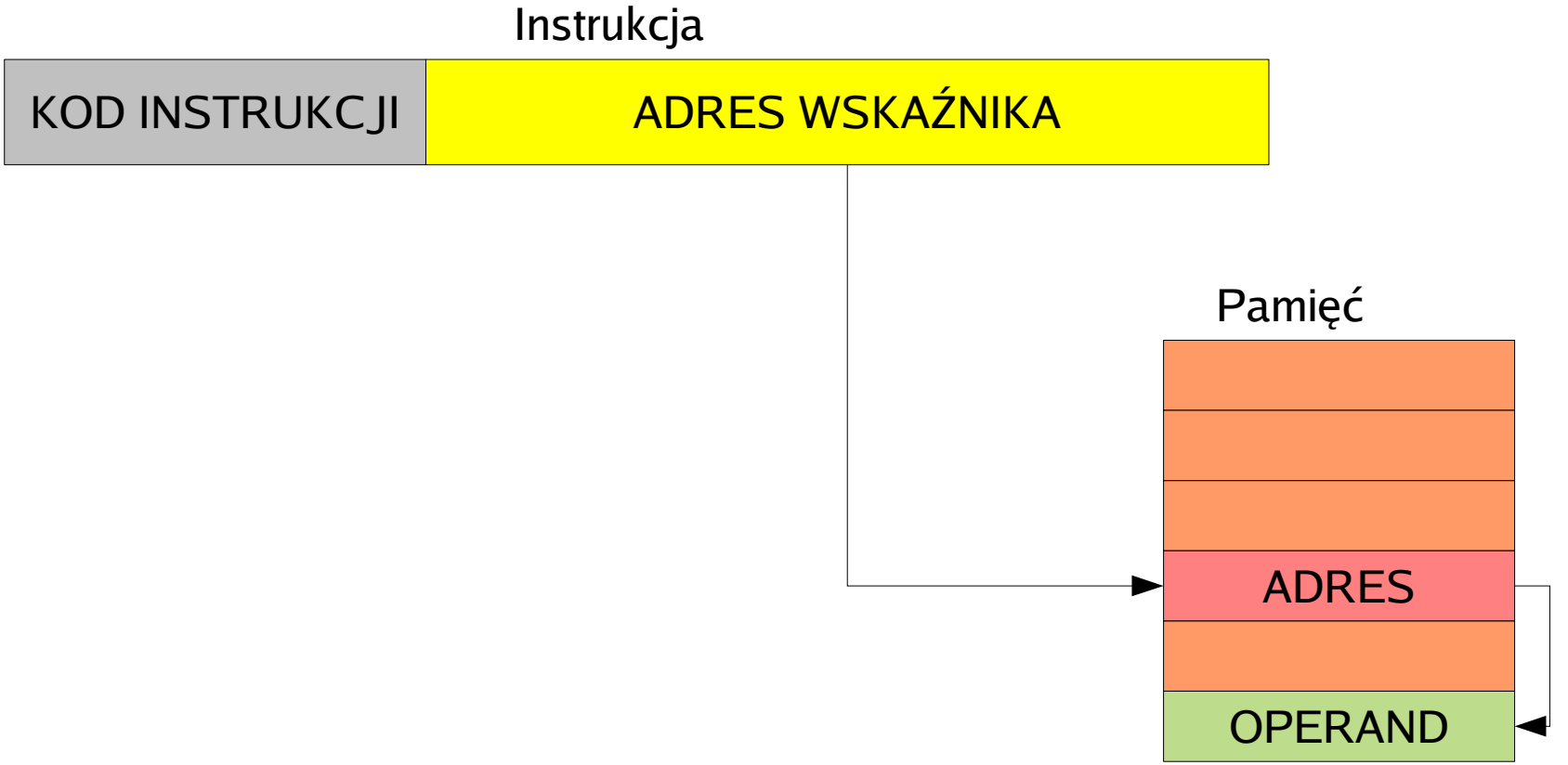
Adresowanie bezpośrednie - format instrukcji



Adresowanie pośrednie

- Instrukcja zawiera adres komórki pamięci. Ta komórka (i kolejne) zawiera adres operandu.
- Przykład ADD (100) – dodaj do akumulatora wartość z pamięci o adresie zapisanym w komórce 100
- Większa przestrzeń adresowa
- Wymaga dodatkowego dostępu do pamięci

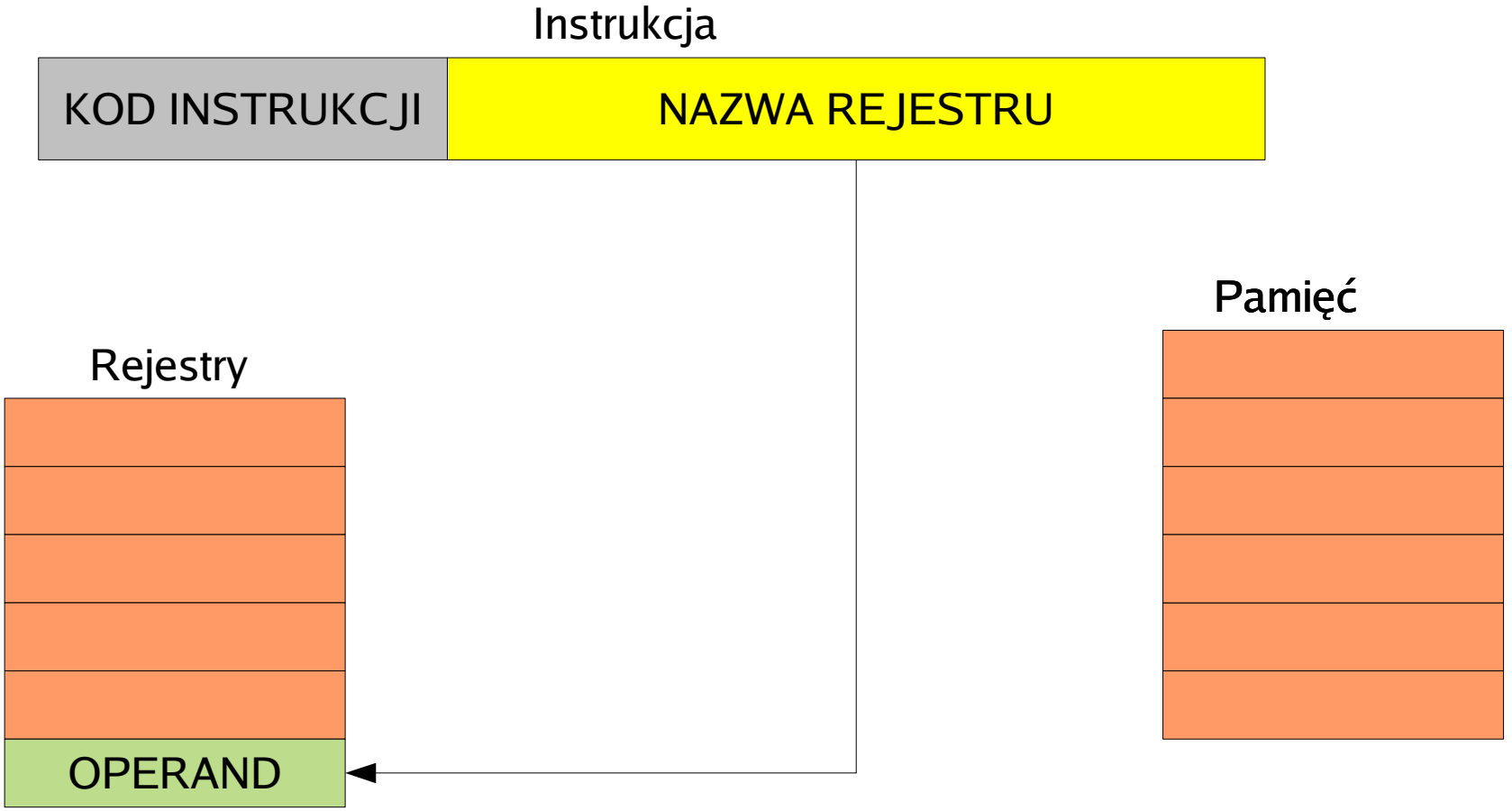
Adresowanie pośrednie - format instrukcji



Adresowanie rejestrowe

- Wartość operandu przechowywana jest w rejestrze
- ADD AX,BX – dodaj do rejestru AX zawartość rejestru BX
- Niewielki rozmiar instrukcji
- Brak odwołania do pamięci – wydajność!
- Niewielka liczba dostępnych rejestrów

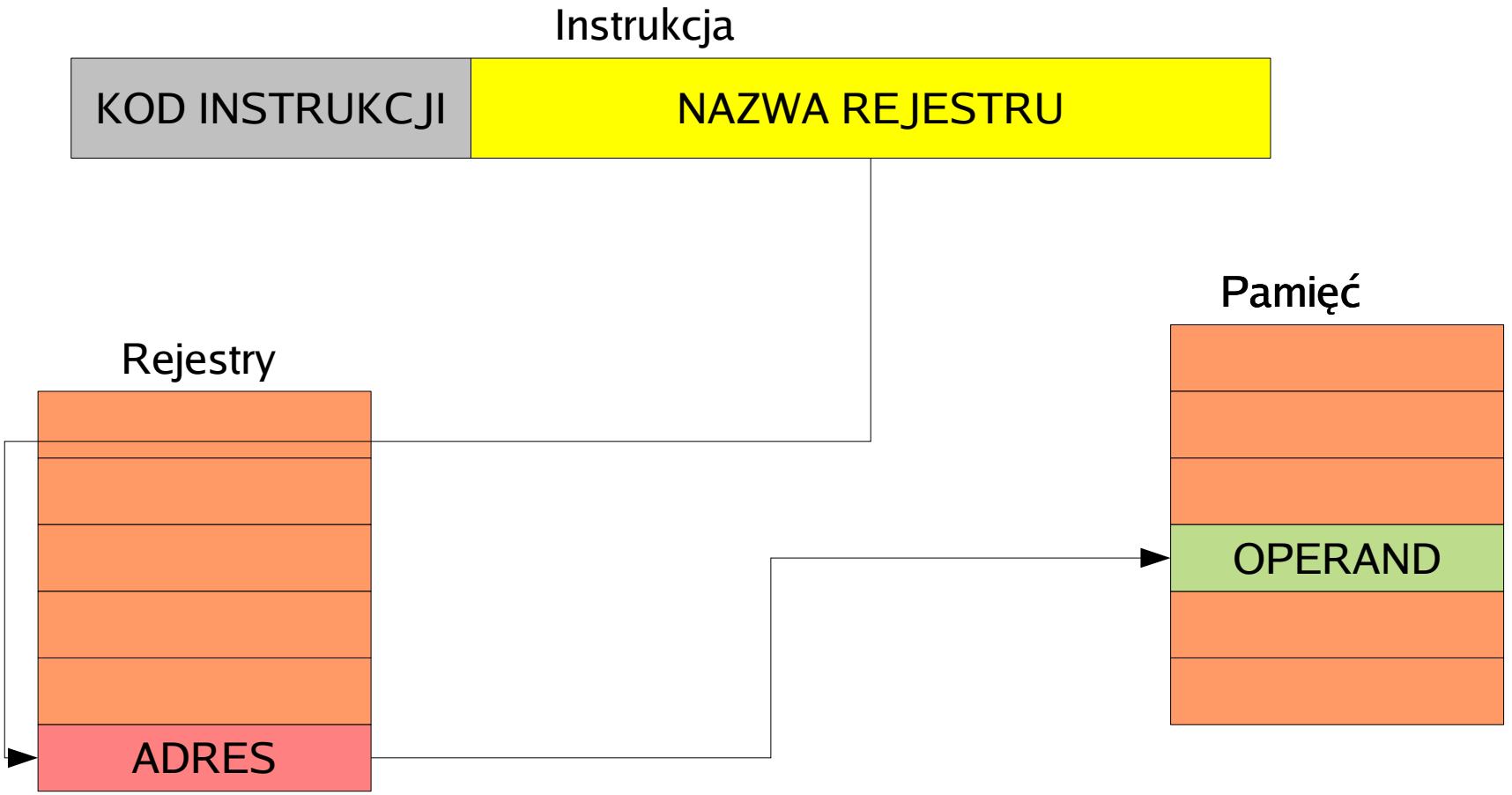
Adresowanie rejestrowe - format instrukcji



Adresowanie rejestrowe pośrednie

- Operand znajduje się w komórce pamięci o adresie przechowywanym we wskazanym rejestrze.
- Przykład: ADD [BX] – dodaj do akumulatora zawartość komórki pamięci o adresie z rejestru BX
- Duża przestrzeń adresowa
- Nieduży rozmiar instrukcji

Adresowanie rejestrowe pośrednie – format instrukcji



Adresowanie bazowo-indeksowe

- Adres efektywny jest sumą zawartości rejestru oraz przesunięcia.
- Różne odmiany:
  - Adresowanie względne
  - Adresowanie bazowe
  - Adresowanie indeksowe
  - Adresowanie indeksowe ze skalowaniem

Adresowanie względne

- Adres operandu podawany jest jako przesunięcie względem adresu aktualnie wykonywanej instrukcji (rejestr PC).
- Krótka instrukcja (zwykle 1-bajtowe przesunięcie)
- Mała przestrzeń adresowa
- Wykorzystanie zasady lokalności

Adresowanie bazowe

- Adres efektywny wyliczany względem pewnej wartości bazowej
- Rejestry segmentowe (bazowe)
- Segmentacja – podział programu na bloki (kodu, danych, stosu)

Adresowanie indeksowe

- Adres efektywny wyliczany na podstawie zawartości rejestru indeksowego i podanego przesunięcia
- Wygodne operacje na blokach danych
- CPU często ma automatyczne instrukcje wykorzystujące indeksowanie
- Adresowanie indeksowe może być połączone z bazowym

Adresowanie indeksowe ze skalowaniem

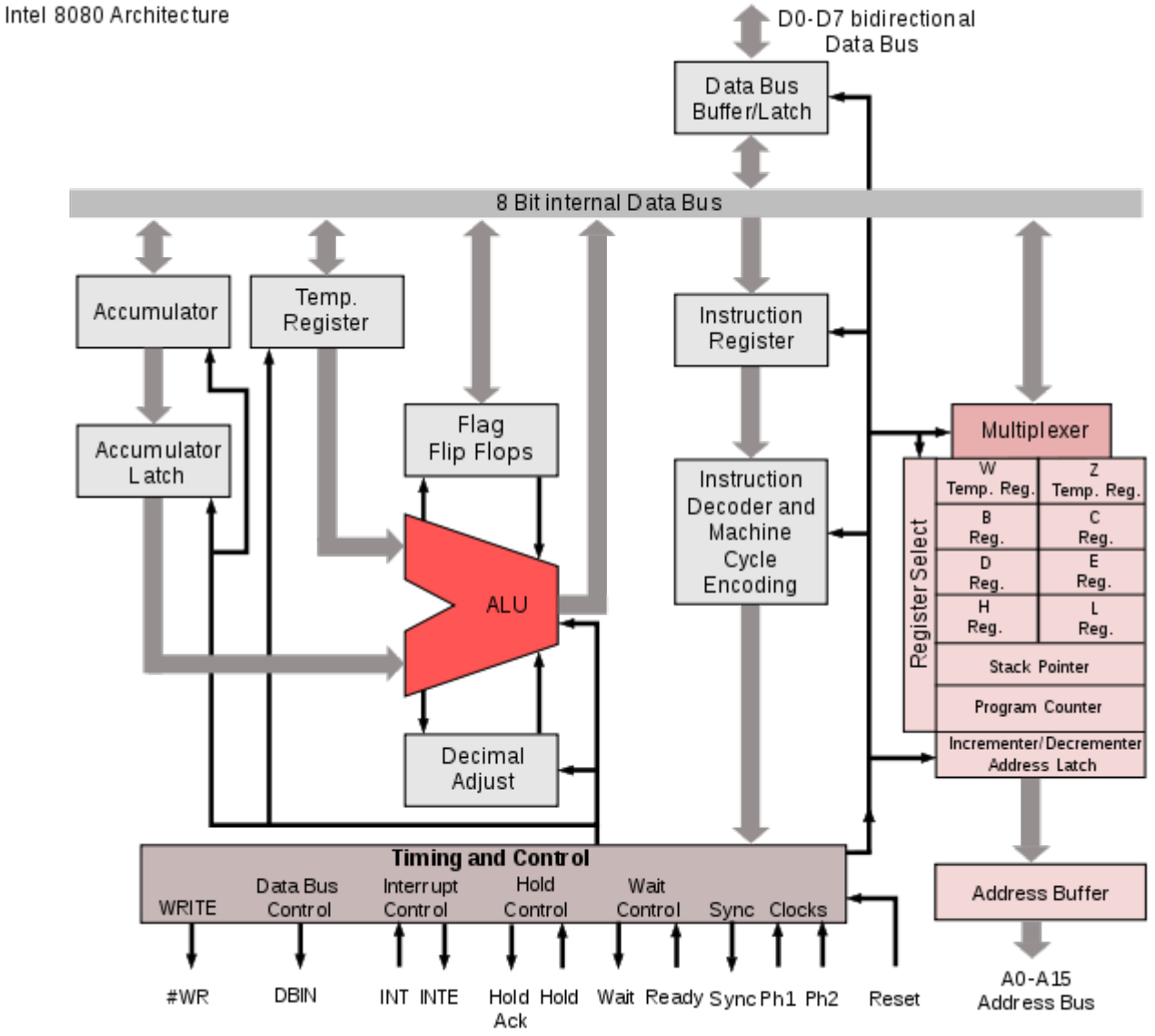
- Wprowadzony dodatkowy czynnik skalujący
- Pozwala na łatwiejszy dostęp do tablic
- Łatwiejszy dostęp do wielobajtowych typów danych
- Typowe czynniki skalujące 1,2,4,8

Adresowanie korzystające ze stosu

- Operacje wykonywane na stosie (podobni jak w RPN)
- Operandy pobierane są z wierzchołka stosu, wynik umieszczany jest na stosie.
- Przykład:

PUSH a - *umieść a na stosie*
PUSH b - *umieść b na stosie*
ADD - *dodaj a i b*
POP - *pobierz wynik*

Architektura CPU - Intel 8080



Wikipedia

Podsumowanie

- Podstawowe elementy komputera.
- Procesor (CPU). Rozkazy procesora
- Tryby adresowania
- Architektura procesora – ALU, FPU, rejestry, układy sterujące