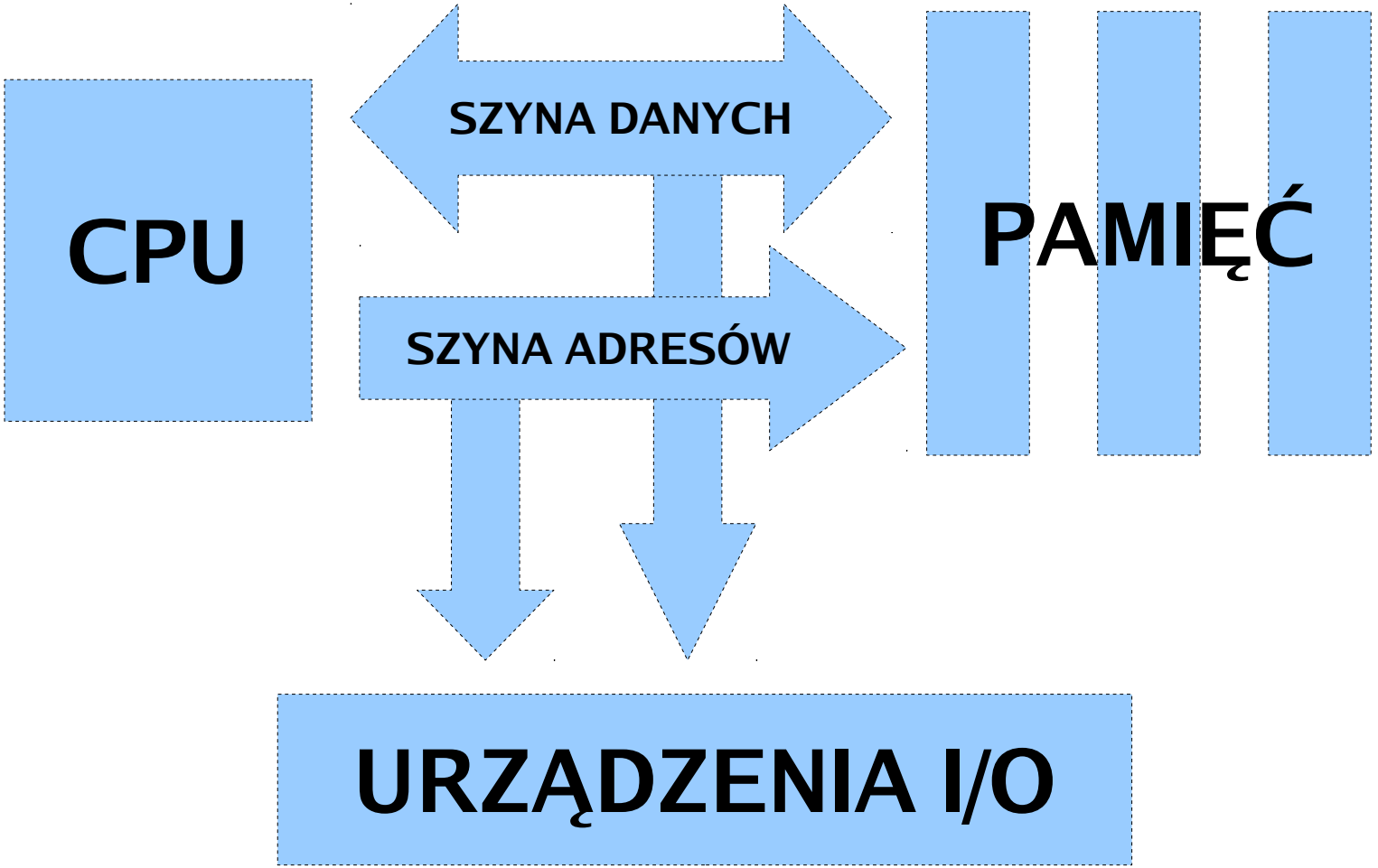
Systemy operacyjne III

WYKŁAD 4

Jan Kazimirski

Zarządzanie pamięcią operacyjna

Architektura komputera



Pamięć operacyjna

- Składa się z komórek o określonych adresach
- CPU posiada rozkazy operujące na komórkach pamięci
- Rozmiar fizycznej pamięci zależy od szerokości szyny adresów:

16 bitów	-	64KB
20 bitów	-	1MB
32 bity	-	4GB
36 bitów	-	64GB

Model pamięci

- System **jednoprogramowy** – pamięć podzielona na dwie części: pamięć systemu operacyjnego i pamięć programu
- System **wieloprogramowy** – konieczność zarządzania pamięcią poszczególnych programów

Zarządzanie pamięcią

- Model jednoprogramowy
  - Przydział całej wolnej pamięci do zadania
  - Przykład: DOS
- Model wieloprogramowy
  - Przydział poszczególnych fragmentów pamięci do poszczególnych programów (zadań)
  - Efektywne zarządzanie (jak najwięcej zadań w pamięci)
  - Przykłady: Unix/Linux, Windows

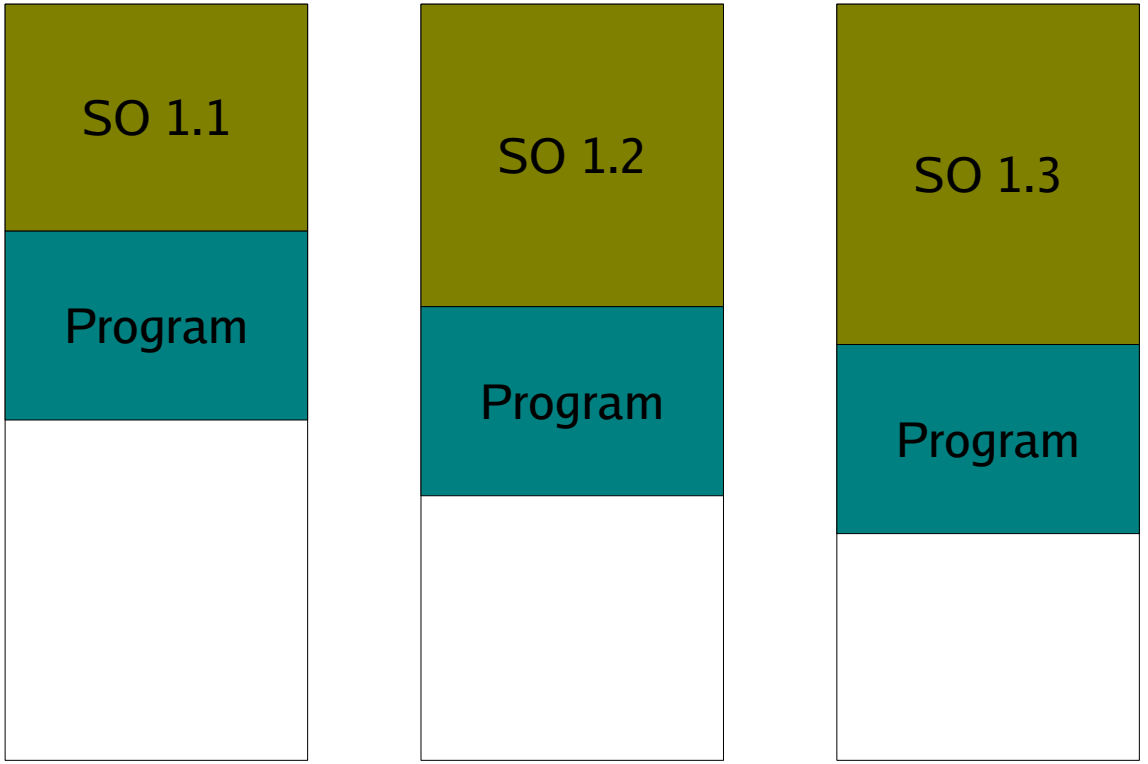
Zarządzanie pamięcią – wymagania ogólne

- Relokacja →
- Ochrona pamięci →
- Współdzielenie →
- Struktura logiczna →
- Struktura fizyczna →

Relokacja

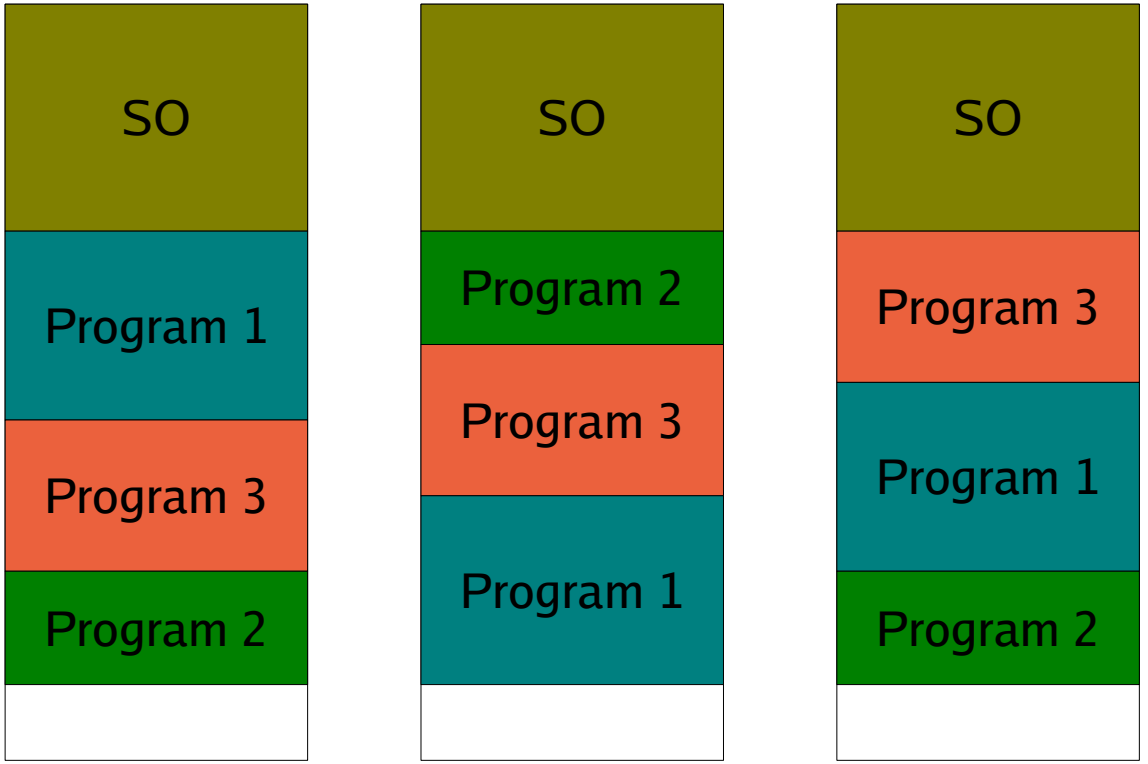
- Pamięć wykorzystywana jest przez wiele procesów
- Programista nie zna położenia swojego programu w czasie wykonania
- Położenie programu może się zmienić.
- Program zawiera odwołania do adresów, które muszą być konwertowane.

Relokacja – system jednozadaniowy



Problem relokacji może wystąpić również w systemie jednozadaniowym jeżeli program ma być uruchamiany np. na różnych wersjach systemu operacyjnego

Relokacja – system wielozadaniowy



Adresy programów będą się różnić zależnie od ilości, wielkości i kolejności ich uruchomienia

Mechanizm wymiany między pamięcią a urządzeniem wymiany jeszcze bardziej komplikuje sytuację (zmiana adresów w trakcie wykonania)

Ochrona pamięci

- Każdy proces powinien być chroniony przed innymi
- Poszczególne elementy programu mogą wymagać dodatkowej ochrony (dane, kod, dane stałe).
- Ze względu na relokację ochrona musi być realizowana w trakcie wykonywania programu
- Efektywne mechanizmy ochrony wymagają wsparcia sprzętu.

Współdzielenie

- Możliwość współdzielenia tego samego bloku pamięci przez kilka procesów
- Kontrolowane współdzielenie pamięci pozwala na komunikację między procesami.
- Współdzielenie fragmentów programu (np. biblioteki standardowe) oszczędza pamięć.

Struktura logiczna

- Programy w postaci modułów
- Niezależna kompilacja modułów
- Niezależne uprawnienia (tylko do odczytu, tylko do wykonania itp.)
- Współdzielenie modułów przez procesy.

Struktura fizyczna

- Fizyczna organizacja pamięci (rozmiar, sposób adresowania)
- Przepływ danych między pamięcią operacyjną a urządzeniem wymiany
- Reagowanie na sytuacje braku pamięci.

Metody zarządzania pamięcią

- Partycjonowanie (statyczne, dynamiczne) →
- Proste stronicowanie →
- Prosta segmentacja →
- Pamięć wirtualna i stronicowanie →→
- Pamięć wirtualna i segmentacja →→

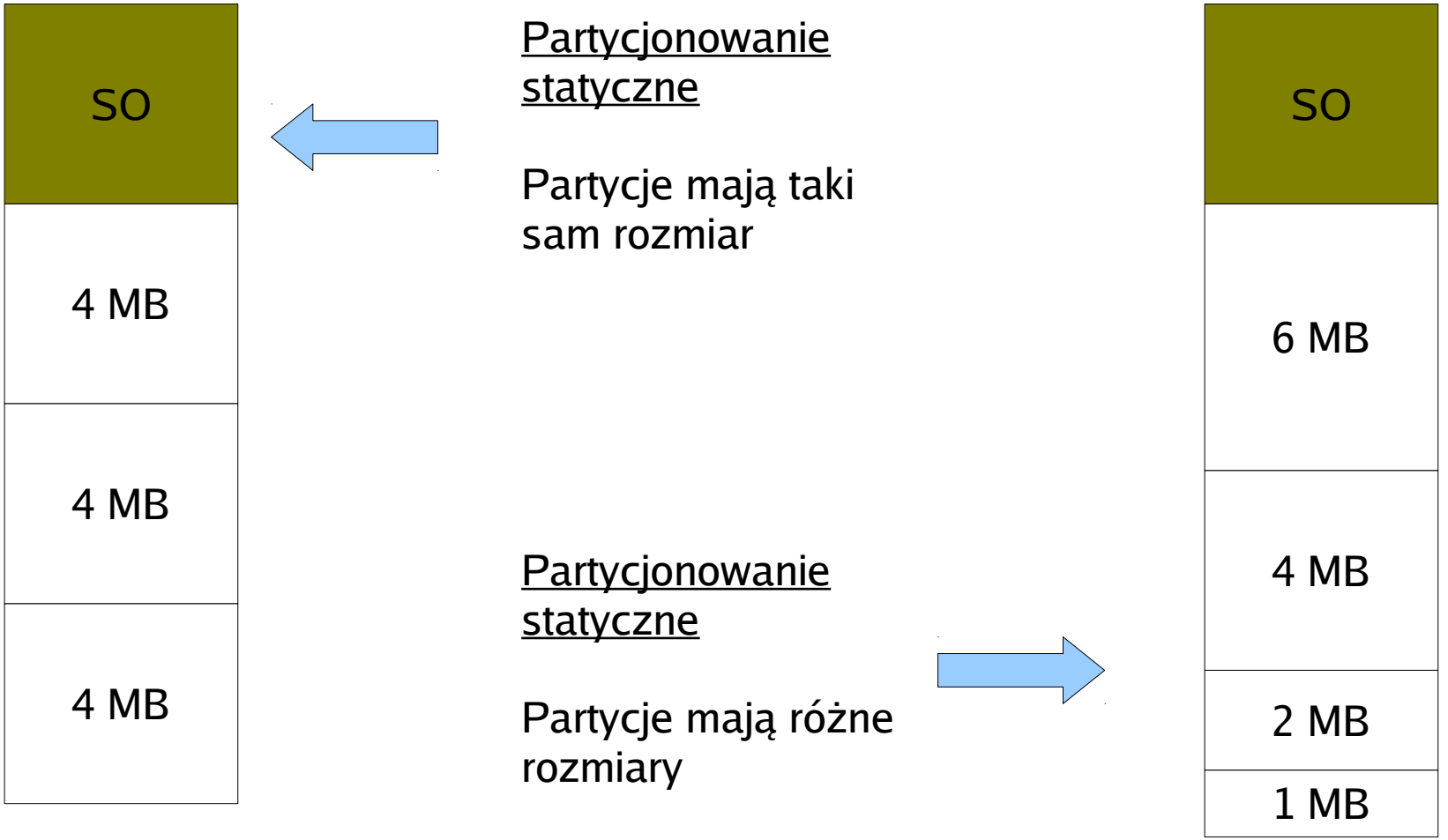
Partycjonowanie

- Prosta metoda zarządzania pamięcią – obecnie nie używana
- Podział dostępnej pamięci na obszary
  - partycjonowanie statyczne
  - partycjonowanie dynamiczne

Partycjonowanie statyczne

- Dostępna pamięć podzielona jest na obszary o stałych granicach.
- Proces o rozmiarze mniejszym lub równym wielkości partycji może być załadowany.
- Procesy większe muszą być nakładkowane (overlays).
- Gdy brakuje pamięci proces może być usunięty z partycji („wymieciony”).

Partycjonowanie statyczne c.d.



Partycjonowanie statyczne c.d.

- Problem – proces zawsze zajmuje cała partycję - „fragmentacja wewnętrzna”
- Narzucona maksymalna liczba aktywnych procesów
- Zaleta – prosta implementacja, niewielkie obciążenie systemu

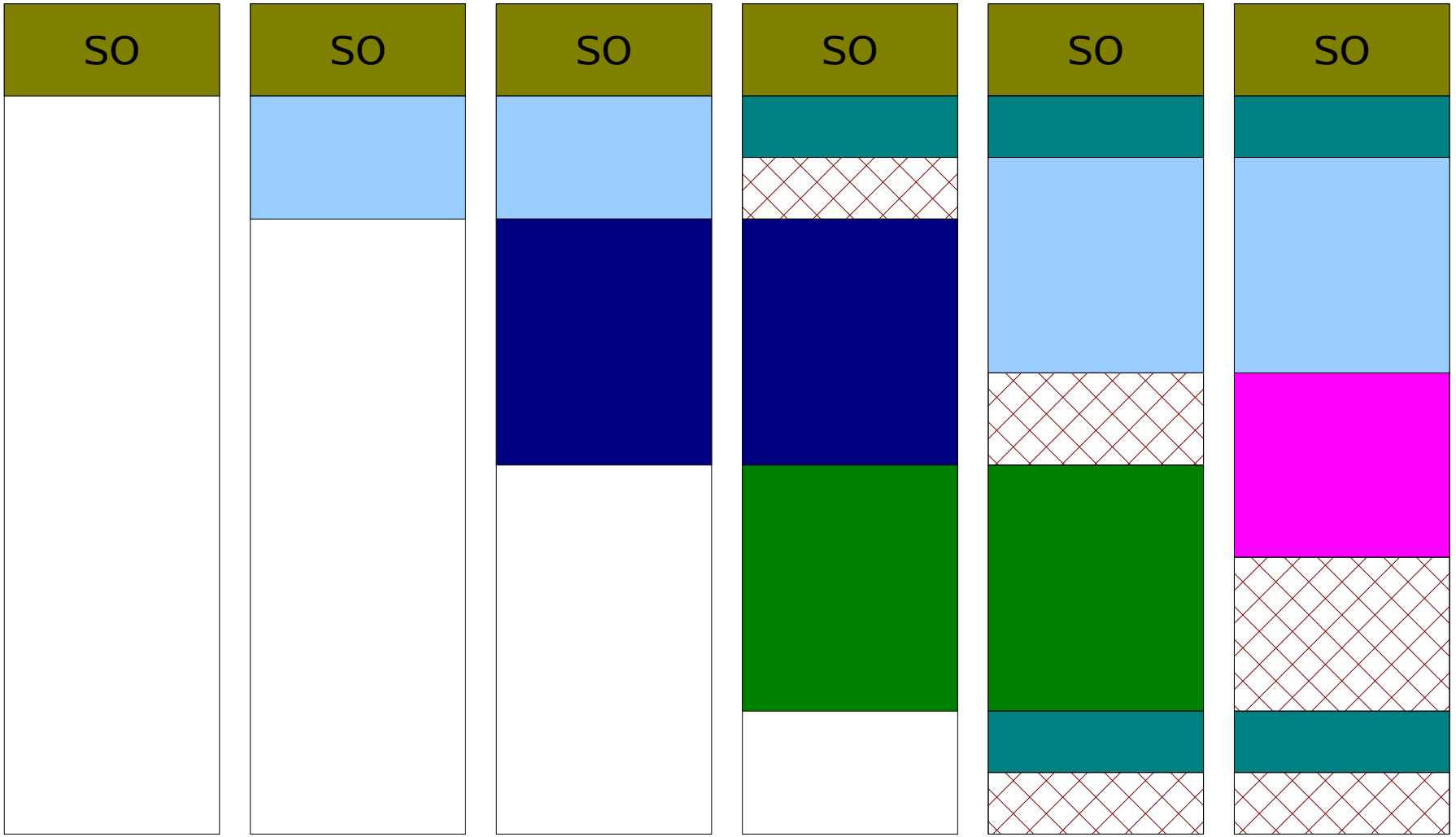
Algorytm rozmieszczania

- Partycje o jednakowych rozmiarach – nie ma znaczenia która będzie użyta
- Partycje o różnych rozmiarach:
  - Najmniejsza pasująca
  - Minimalizuje fragmentację
  - Rozwiązanie z osobną kolejką dla każdej partycji lub z globalną kolejką procesów

Partycjonowanie dynamiczne

- Partycje o różnej wielkości (dopasowanej do procesu)
- Różna liczba partycji.
- Eksploatacja prowadzi do powstawania „dziur” („fragmentacja zewnętrzna”)
- Konieczność okresowego porządkowania pamięci.

Partycjonowanie dynamiczne



Algorytmy rozmieszczania

- Najlepsze dopasowanie →
- Pierwsze dopasowanie →
- Kolejne dopasowanie →
- System bliźniaczy („Buddy system”) →

Najlepsze dopasowanie

- Szukany jest blok najlepiej pasujący rozmiarem do procesu.
- Niewielka fragmentacja
- „Zaśmiecanie” pamięci przez małe bloki
- Duży narzut (poszukiwanie pasującego bloku)

Pierwsze dopasowanie

- Szukanie pierwszego wolnego bloku pamięci w którym zmieści się proces
- Duża szybkość
- „Zaśmieca” początkową część pamięci operacyjnej
- Konieczność przeglądania wielu bloków na początku pamięci

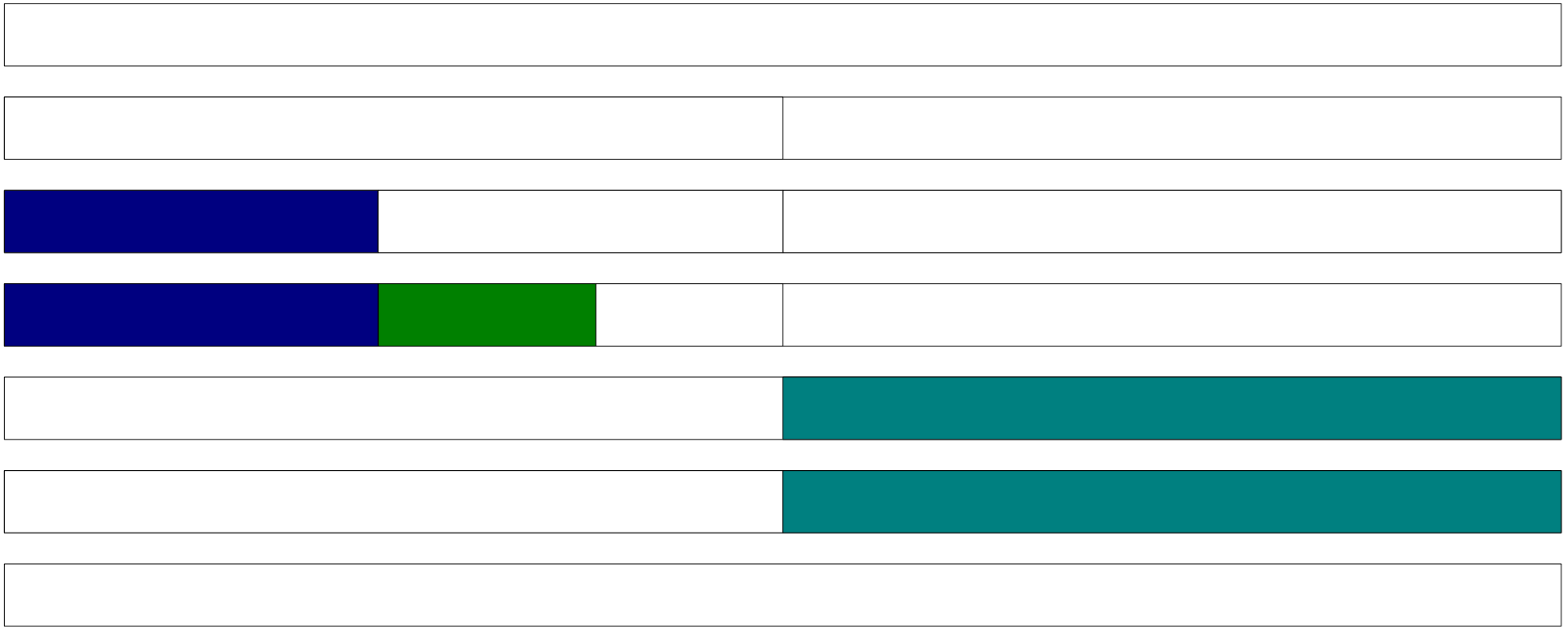
Kolejne dopasowanie

- Pamięć przeglądana jest od bloku do którego ostatnio wstawiono proces
- Tendencja do wykorzystywania końcowej części pamięci
- Wydajny, ale może wymagać częstego porządkowania.

System bliźniaczy

- Początkowo pamięć traktowana jest jako jeden blok o rozmiarze 2^u.
- Jeżeli rozmiar procesu jest większy niż połowa bloku to cały blok jest przydzielany
- W przeciwnym wypadku blok jest dzielony na połowę
- Podziału dokonuje się dopóki nie zostanie wygenerowany najmniejszy blok do którego proces pasuje
- System przechowuje listę „dziur”.
- Dziury mogą być wykorzystane do rozmieszczania procesów
- Sąsiednie „dziury” są łączone w większe bloki i umieszczane w puli.

System bliźniaczy c.d.



Relokacja

- Po załadowaniu procesu do pamięci system musi wyliczyć bezwzględne odwołania do pamięci.
- Mechanizm wymiatania może wymagać zmiany absolutnego położenia procesu w pamięci
- Porządkowanie (defragmentacja) pamięci również zaburza bezwzględne adresy pamięci

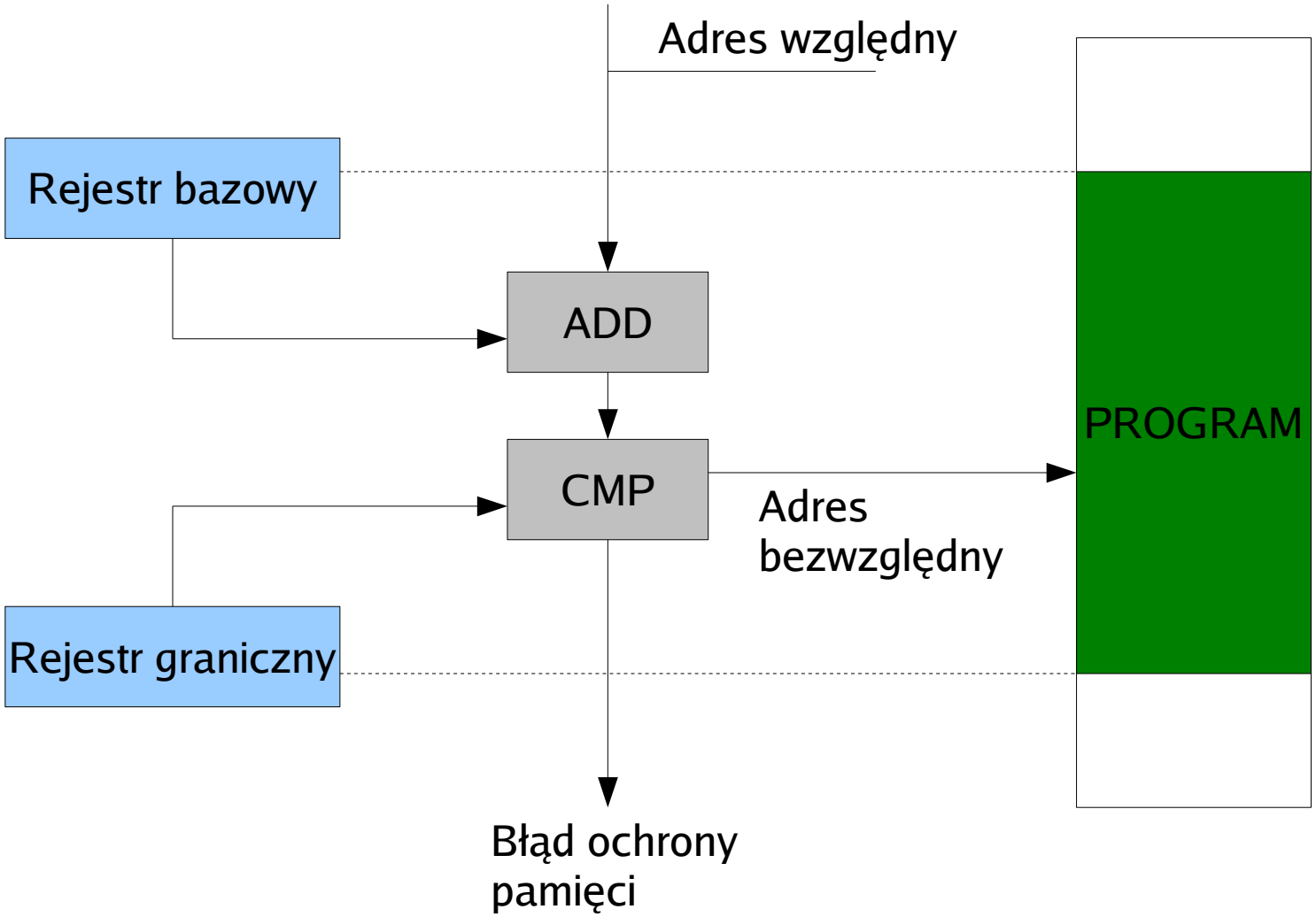
Rodzaje adresów

- Adresy logiczne – odniesienie do pozycji w pamięci niezależnej od bieżącej alokacji danych – muszą być konwertowane przed uruchomieniem.
- Adresy względne – lokalizacja względem jakiegoś znanego punktu
- Adresy fizyczne – rzeczywista lokalizacja w pamięci operacyjnej

Tłumaczenie adresów

- Rejestr bazowy – określa początkową lokalizację programu
- Rejestr graniczny – określa końcową lokalizację programu
- Każde odwołanie w programie wyliczane jest względem rejestru bazowego i porównywane z rejestrem granicznym (izolowanie przestrzeni adresowej procesu)

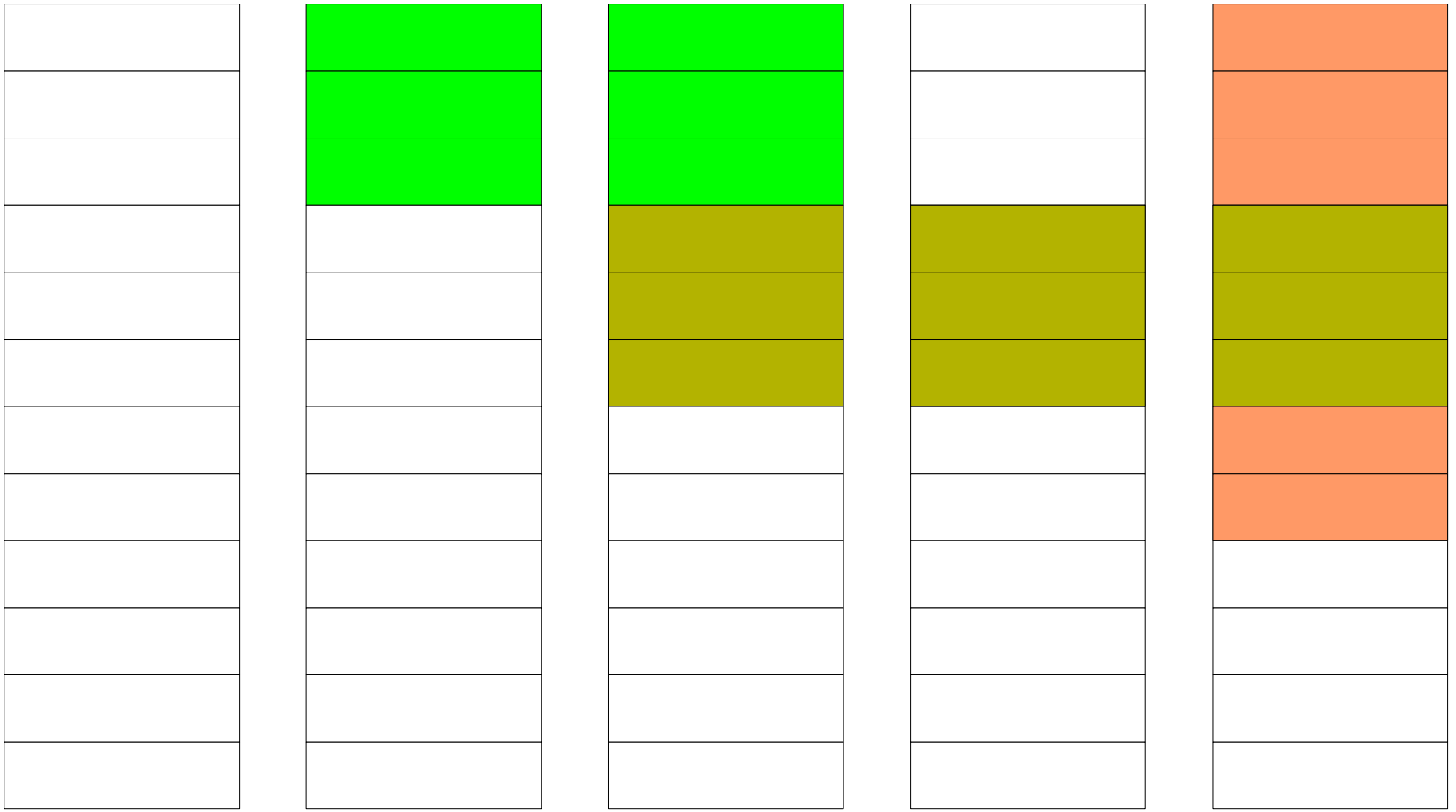
Tłumaczenie adresów c.d.



Stronicowanie

- Pamięć podzielona na niewielkie bloki
- Bloki procesu to **strony** a bloki fizyczne to **ramki**
- System operacyjny przechowuje dla każdego procesu osobną tablicę stron
- Tablica stron odwzorowuje logiczne strony na fizyczne ramki

Stronicowanie c.d.



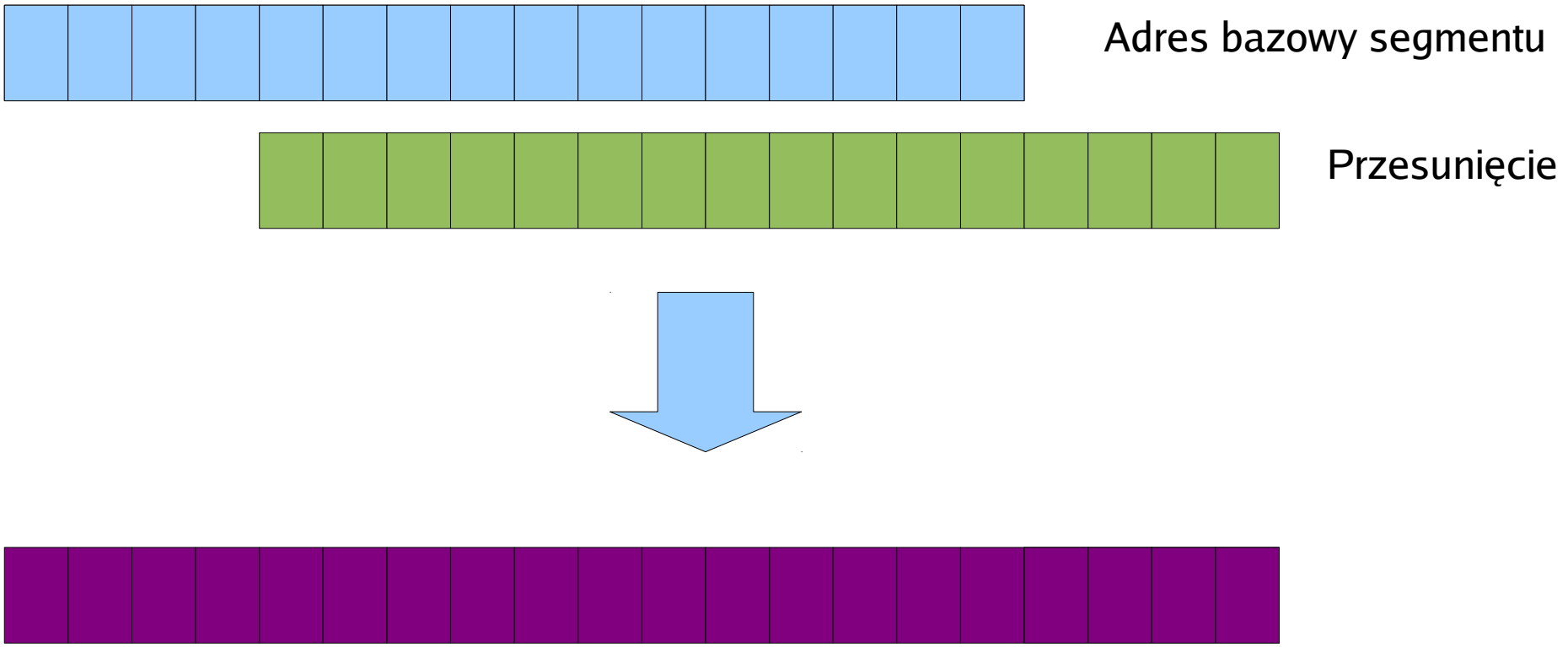
Segmentacja

- Poszczególne segmenty procesu nie muszą być takiej samej wielkości.
- Funkcjonalny podział programu: kod, dane, dane tylko do odczytu, dane współdzielone itd.
- Każdy proces posiada tablicę segmentów
- Adresowanie: adres segmentu + adres przesunięcia w segmencie.

Segmentacja – x86 – tryb rzeczywisty

- Przestrzeń adresowa – 1 MB
- Rejestry segmentowe 16 bitowe:
  - CS – segment kodu
  - DS,ES – segmenty danych
  - SS – segment stosu
- Adres efektywny – adres bazowy (rejestr segmentowy)*16 + przesunięcie.
- Przesunięcie – 16 bitów. Rozmiar segmentu – 64 KB.

Tryb rzeczywisty x86



Tryb rzeczywisty x86 c.d.

- Różne kombinacje adresu bazowego i przesunięcia generowały ten sam adres efektywny.
- Możliwość umieszczania początku segmentu pod dowolnym podzielnym przez 16 adresem.
- Ułatwienie ładowania programu na różne wersje systemu, nakładki rezydentne itp.

Program COM

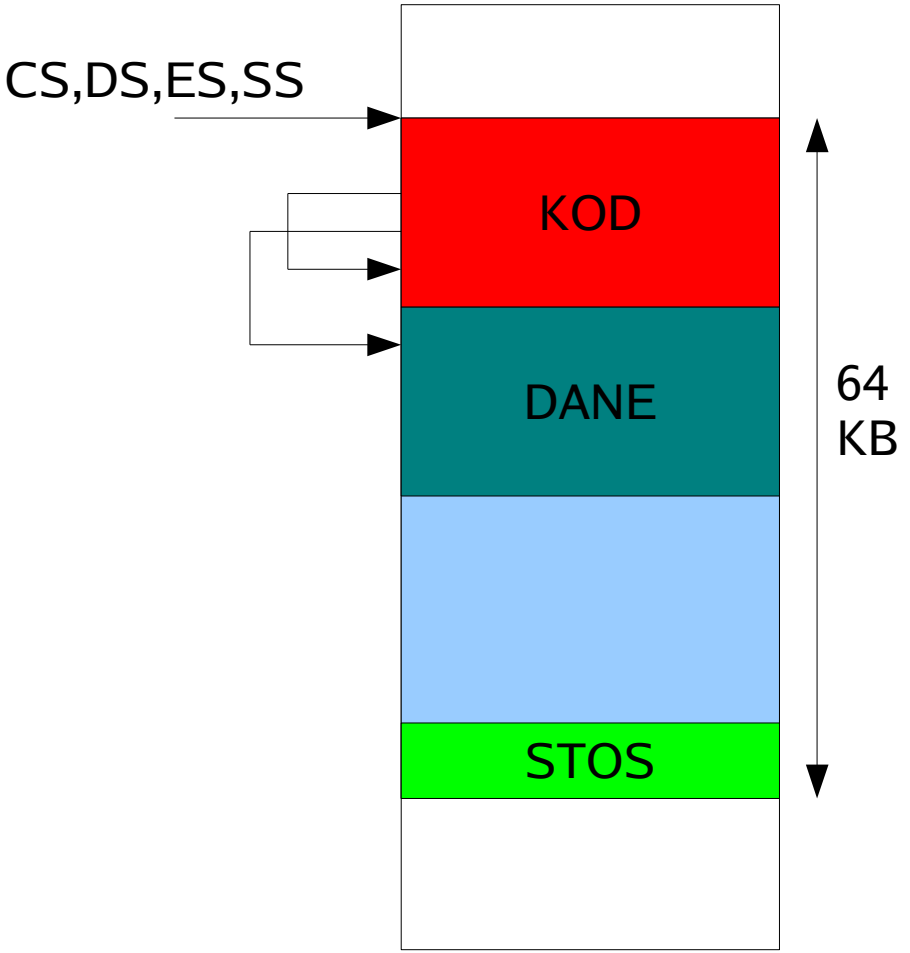
Program COM.

Cały program nie mógł przekroczyć 64 KB (kod,dane,stos)

Ładowany pod dowolny adres podzielny przez 16 (początek segmentu).

Wszystkie rejestry segmentowe ładowane adresem początku programu (segmenty nakładają się na siebie).

Wszystkie skoki w programie realizowane jako skoki „bliskie” - w ramach jednego segmentu.



Program EXE

Program EXE.

Program mógł składać się z kilku segmentów (rozmiar programu powyżej 64 KB)

Ładowanie programu wymagało określenia absolutnych adresów segmentów.

Skoki tzw. „dalekie” (między-segmentowe) wymagały przeliczenia adresów w czasie ładowania programu do pamięci.

Program EXE wymagał specjalnego nagłówka pozwalającego systemowi operacyjnemu dokonać odpowiednich wyliczeń adresów efektywnych