

# Informatyka 2

---

Politechnika Białostocka - Wydział Elektryczny  
Elektrotechnika, semestr III, studia stacjonarne I stopnia  
Rok akademicki 2018/2019

**Wykład nr 12 (18.12.2018)**

dr inż. Jarosław Forenc

## Plan wykładu nr 12

- Systemy plików
  - ext2
- Zarządzanie pamięcią operacyjną
  - partycjonowanie statyczne i dynamiczne
  - proste stronicowanie, prosta segmentacja
  - pamięć wirtualna
  - stronicowanie i segmentacja pamięci wirtualnej

## ext2

- pierwszy system plików w Linuxie: **Minix** (14-znakowe nazwy plików i maksymalny rozmiar wynoszący 64 MB)
- system Minix zastąpiono nowym systemem nazwanym rozszerzonym systemem plików - **ext** (ang. **extended file system**), a ten, w styczniu 1993 r., systemem **ext2** (ang. **second extended file system**)
- w systemie ext2 podstawowym elementem podziału dysku jest **blok**
- wielkość bloku jest stała w ramach całego systemu plików, określana na etapie jego tworzenia i może wynosić 1024, 2048 lub 4096 bajtów
- w celu zwiększenia bezpieczeństwa i optymalizacji zapisu na dysku posługujemy się nie pojedynczymi blokami, a **grupami bloków**

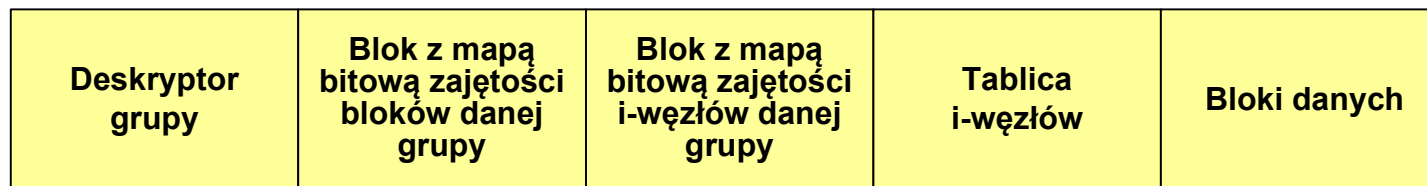


## ext2

- **Boot Sector** (blok startowy) przechowuje informacje wykorzystywane przez system operacyjny podczas jego uruchamiania

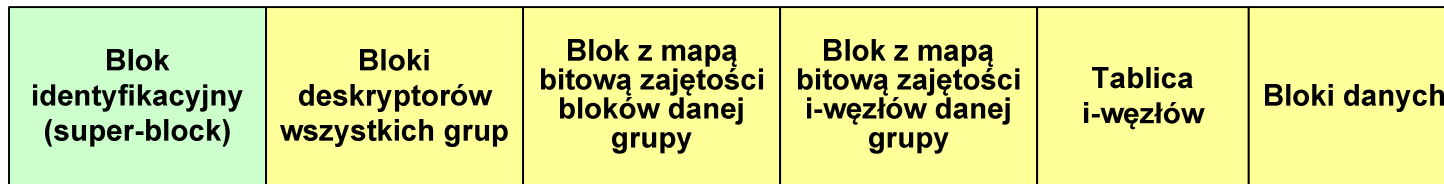


- na poziomie logicznym **grupę bloków** tworzą:
  - deskryptor grupy (32 bajty)
  - blok z mapą zajętości bloków danych (1 blok dyskowy)
  - blok z mapą zajętości i-węzłów (1 blok dyskowy)
  - bloki z tablicą i-węzłów
  - bloki danych



## ext2

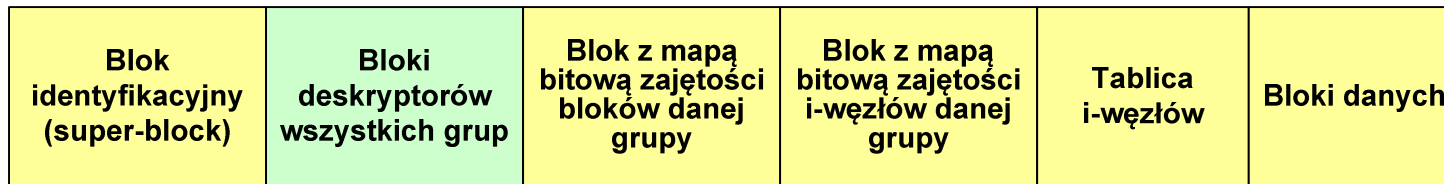
- każda **grupa fizyczna bloków** zawiera informacje o jednej grupie logicznej, a ponadto pewne informacje o całym systemie plików



- w każdej grupie fizycznej bloków znajduje się kopia tego samego bloku identyfikacyjnego oraz kopia bloków z deskryptorami wszystkich grup
- **blok identyfikacyjny** zawiera informacje na temat systemu plików:
  - numer urządzenia, na którym jest super-block
  - rodzaj systemu plików
  - rozmiar bloku
  - struktury do synchronizacji dostępu
  - czas dokonanej ostatnio zmiany
  - informacje specyficzne dla konkretnej implementacji

## ext2

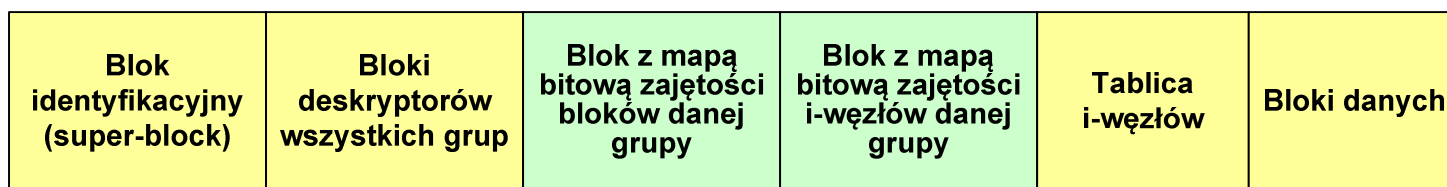
- każda **grupa fizyczna bloków** zawiera informacje o jednej grupie logicznej, a ponadto pewne informacje o całym systemie plików



- w **deskryptorach grupy** znajdują się informacje na temat grupy bloków:
  - numer bloku z bitmapą zajętości bloków grupy
  - numer bloku z bitmapą zajętości i-węzłów
  - numer pierwszego bloku z tablicą i-węzłów
  - liczba wolnych bloków
  - liczba wolnych i-węzłów w grupie
  - liczba katalogów w grupie

## ext2

- każda **grupa fizyczna bloków** zawiera informacje o jednej grupie logicznej, a ponadto pewne informacje o całym systemie plików



- **blok z mapą bitową zajętości bloków danej grupy** jest tablicą bitów o rozmiarze jednego bloku
  - jeśli blok ma rozmiar 1 kB to pojedynczą mapą można opisać fizyczną grupę 8096 bloków czyli 8 MB danych
  - jeśli natomiast blok ma rozmiar 4 kB, to fizyczna grupa bloków zajmuje 128 MB danych
- przed tablicą i-węzłów znajduje się **blok z mapą bitową zajętości i-węzłów danej grupy** - jest to tablica bitów, z których każdy zawiera informację czy dany i-węzeł jest wolny czy zajęty

## ext2 - i-węzeł

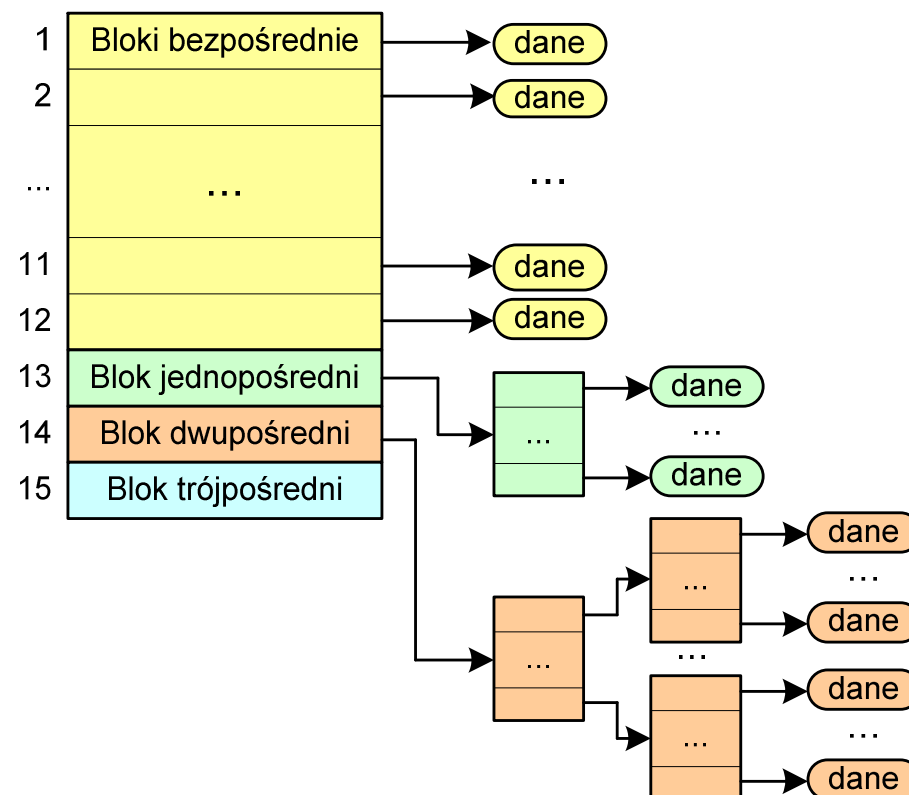
- pliki na dysku reprezentowane są przez **i-węzły** (ang. **i-node**)
- każdemu plikowi odpowiada dokładnie jeden i-węzeł, który jest strukturą zawierającą m.in. następujące pola:
  - numer i-węzła w dyskowej tablicy i-węzłów
  - typ pliku: zwykły, katalog, łącze nazwane, specjalny, znakowy
  - prawa dostępu do pliku: dla wszystkich, grupy, użytkownika
  - liczba dowiązań do pliku
  - identyfikator właściciela pliku
  - identyfikator grupy właściciela pliku
  - rozmiar pliku w bajtach (max. 4 GB)
  - czas utworzenia pliku
  - czas ostatniego dostępu do pliku
  - czas ostatniej modyfikacji pliku
  - liczba bloków dyskowych zajmowanych przez plik



## ext2 - i-węzeł

□ położenie pliku na dysku określają w i-węźle pola:

- 12 adresów bloków zawierających dane (w systemie Unix jest ich 10)
  - **bloki bezpośrednie**
- 1 adres bloku zawierającego adresy bloków zawierających dane - **blok jednopięśredni** (ang. single indirect block)
- 1 adres bloku zawierającego adresy bloków jednopięśrednich - **blok dwupięśredni** (ang. double indirect block)
- 1 adres bloku zawierającego adresy bloków dwupięśrednich - **blok trójpięśredni** (ang. triple indirect block)



## ext2

- **nazwy plików** przechowywane są w **katalogach**, które w systemie Linux są plikami, ale o specjalnej strukturze
- katalogi składają się z ciągu tzw. **pozycji katalogowych** o nieustalonej z góry długości
- każda pozycja opisuje dowiązanie do jednego pliku i zawiera:
  - numer i-węzła (4 bajty)
  - rozmiar pozycji katalogowej (2 bajty)
  - długość nazwy (2 bajty)
  - nazwa (od 1 do 255 znaków)

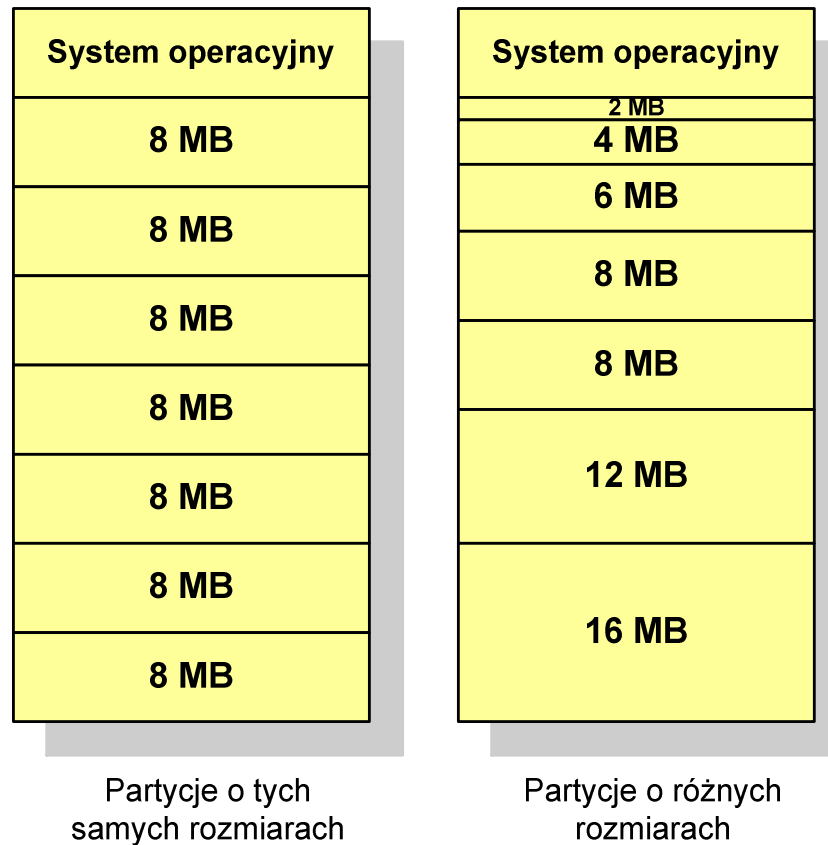
```
struct ext2_dir_entry
{
    _u32  inode           /* numer i-wezla           */
    _u16  rec_len        /* dlugosc pozycji katalogowej */
    _u16  name_len       /* dlugosc nazwy           */
    char  name[EXT2_NAME_LEN] /* nazwa                   */
}
```

## Zarządzanie pamięcią

- zarządzanie pamięcią polega na wydajnym przenoszeniu programów i danych do i z pamięci operacyjnej
- w nowoczesnych wieloprogramowych systemach operacyjnych zarządzanie pamięcią opiera się na **pamięci wirtualnej**
- pamięć wirtualna bazuje na wykorzystaniu **segmentacji i stronicowania**
- z historycznego punktu widzenia w systemach komputerowych stosowane były/są następujące metody zarządzania pamięcią:
  - partycjonowanie statyczne, partycjonowanie dynamiczne
  - proste stronicowanie, prosta segmentacja
  - stronicowanie pamięci wirtualnej, segmentacja pamięci wirtualnej
  - **stronicowanie i segmentacja pamięci wirtualnej**

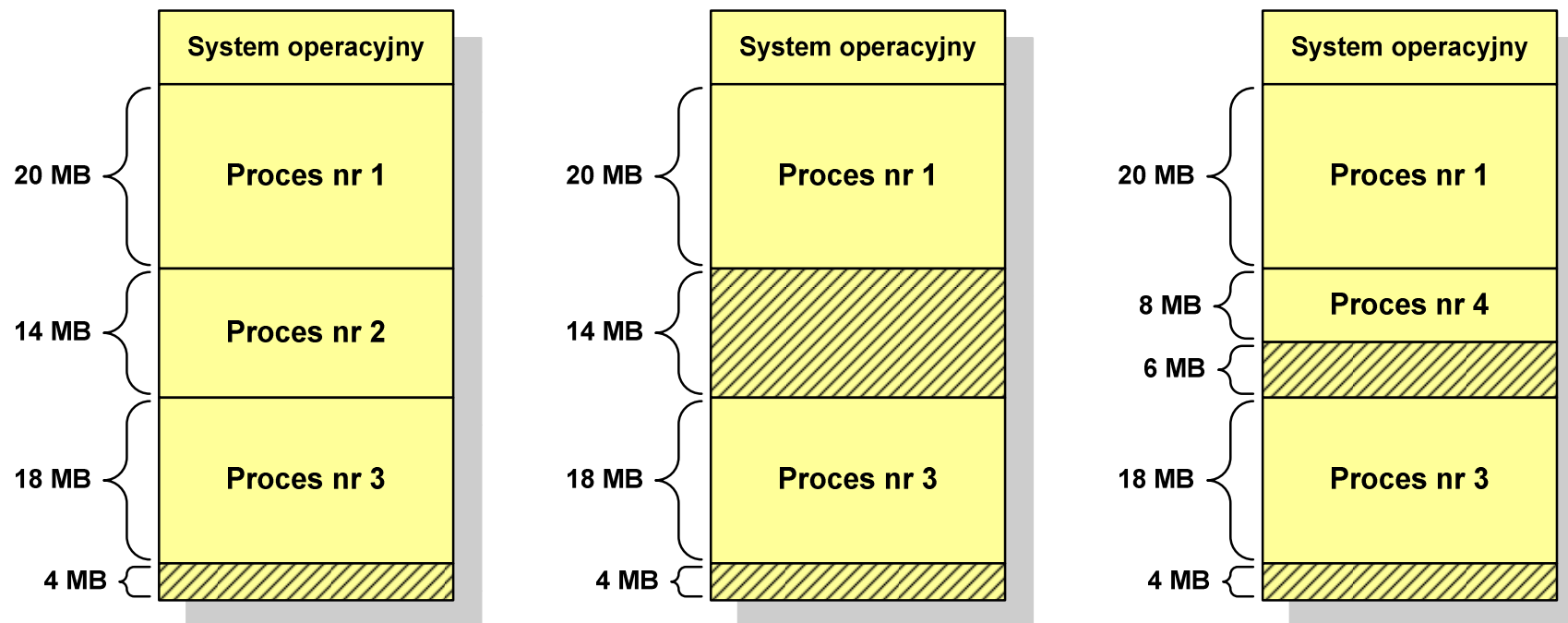
## Partycjonowanie statyczne

- podział pamięci operacyjnej na obszary o takim samym lub różnym rozmiarze, ustalonym podczas generowania systemu



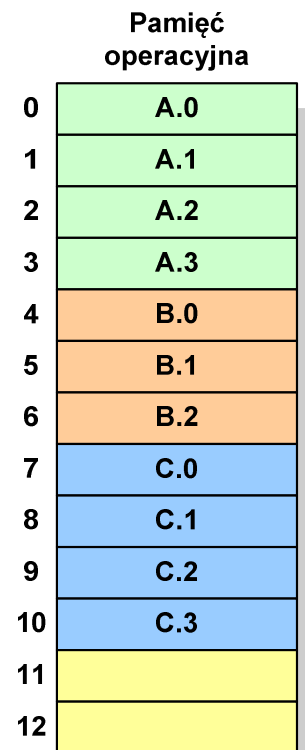
## Partycjonowanie dynamiczne

- partycje są tworzone dynamicznie w ten sposób, że każdy proces jest ładowany do partycji o rozmiarze równym rozmiarowi procesu
- partycje mają różną długość, może zmieniać się także ich liczba
- przykład - w systemie działa 5 procesów: 20 MB, 14 MB, 18 MB, 8 MB, 8 MB

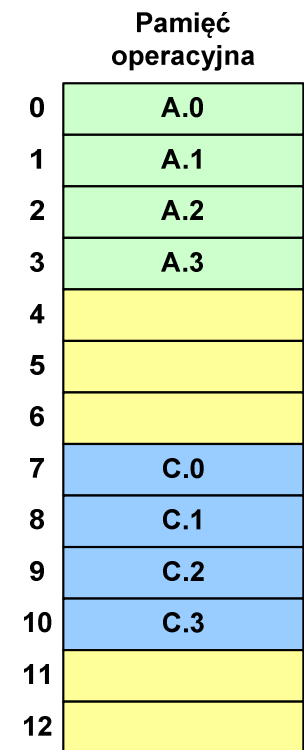


## Proste stronicowanie

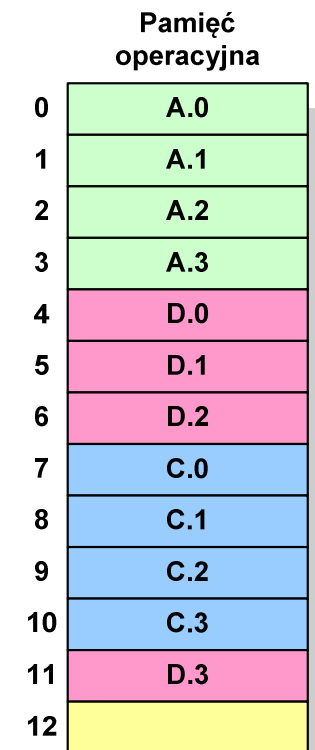
- pamięć operacyjna podzielona jest na jednakowe bloki o stałym niewielkim rozmiarze nazywane **ramkami** lub **ramkami stron** (page frames)
- do tych ramek wstawiane są fragmenty procesu zwane **stronami** (pages)
- aby proces mógł zostać uruchomiony wszystkie jego strony muszą znajdować się w pamięci operacyjnej



Trzy procesy  
w pamięci: A, B, C



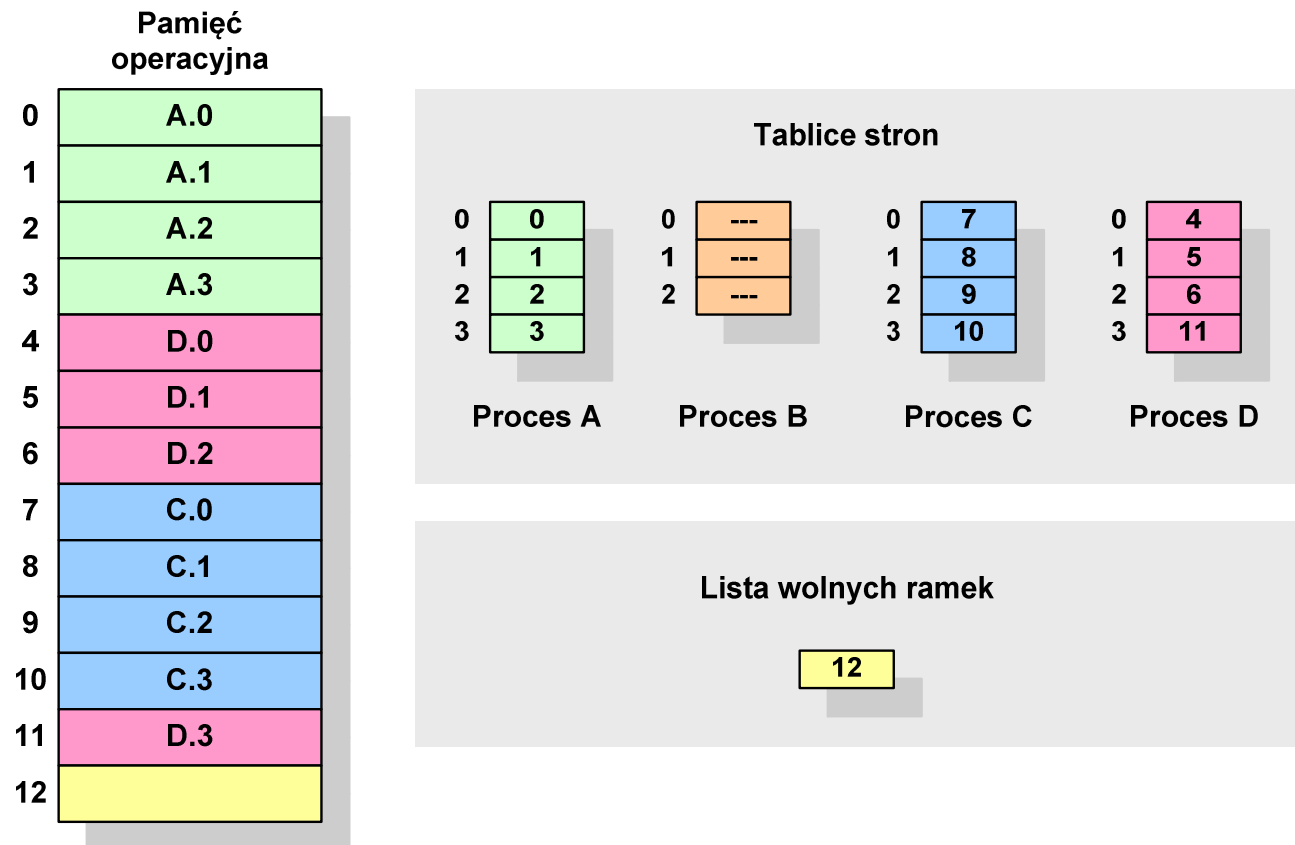
Usunięcie procesu  
B z pamięci



Dodanie procesu D

## Proste stronicowanie

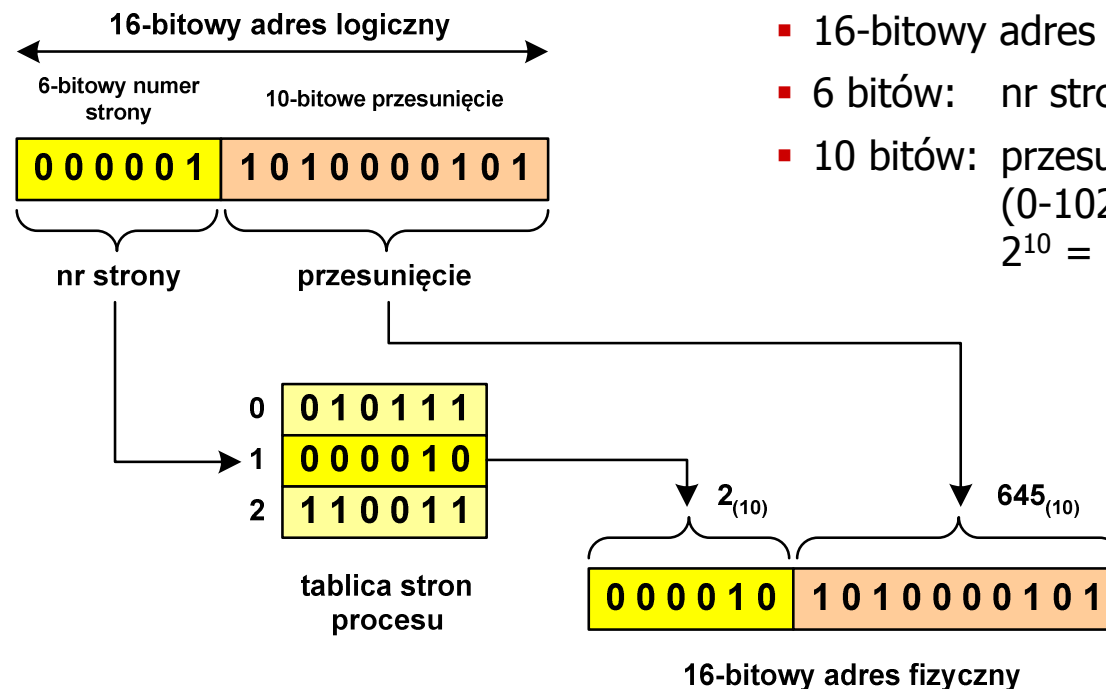
- dla każdego procesu przechowywana jest **tablica strony** (page table) zawierająca lokalizację ramki dla każdej strony procesu



## Proste stronicowanie

- aby mechanizm stronicowania był wygodny ustala się, że rozmiar strony jest liczbą podniesioną do potęgi drugiej - dzięki temu adres względny oraz adres logiczny (numer strony + jej przesunięcie) są takie same

### Przykład:

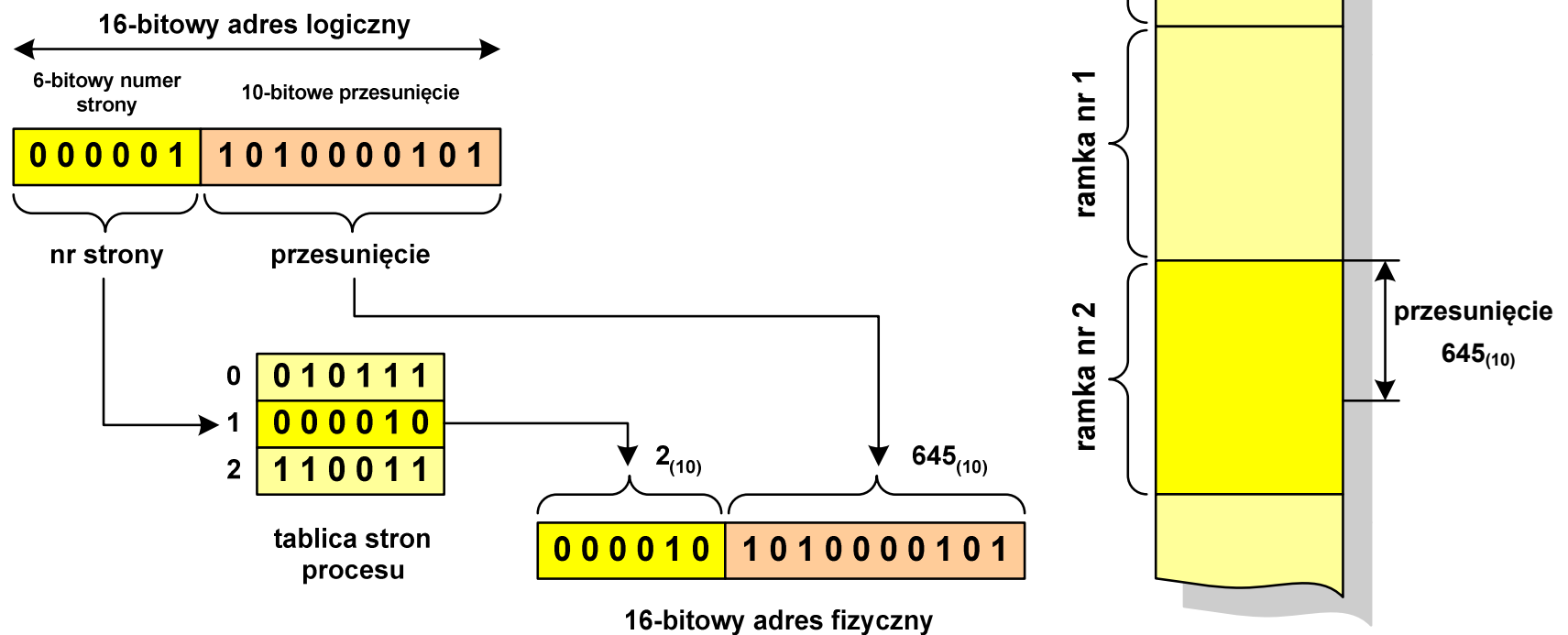


- 16-bitowy adres logiczny
- 6 bitów: nr strony (0-63), max.  $2^6 = 64$  strony
- 10 bitów: przesunięcie w ramach strony (0-1023), rozmiar strony wynosi:  $2^{10} = 1024$  bajty = 1 kB



## Proste stronicowanie

- **zalety:** brak fragmentacji zewnętrznej, stronicowanie nie jest widoczne dla programisty
- **wady:** niewielki stopień fragmentacji wewnętrznej



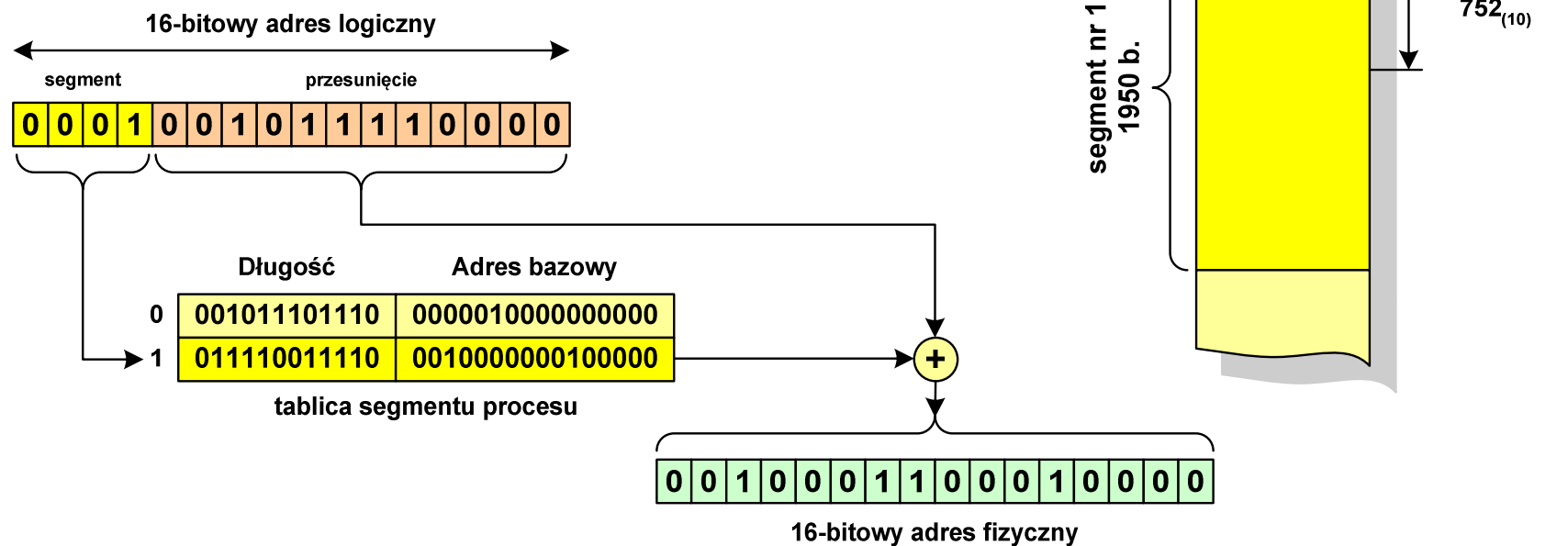
## Prosta segmentacja

- polega na podzieleniu programu i skojarzonych z nim danych na odpowiednią liczbę **segmentów** o **różnej długości**
- ładowanie procesu do pamięci polega na wczytaniu wszystkich jego segmentów do partycji dynamicznych (nie muszą być ciągłe)
- segmentacja jest widoczna dla programisty i ma na celu wygodniejszą organizację programów i danych
- **adres logiczny** wykorzystujący segmentację składa się z dwóch części:
  - numeru segmentu
  - przesunięcia
- dla każdego procesu określana jest **tablica segmentu procesu** zawierająca:
  - długość danego segmentu
  - adres początkowy danego segmentu w pamięci operacyjnej

# Prosta segmentacja

## Przykład:

- 16-bitowy adres logiczny
- 4 bity: nr segmentu (0-15), max.  $2^4 = 16$  segmentów
- 12 bitów: przesunięcie w ramach segmentu (0-4095), rozmiar segmentu wynosi:  $2^{12} = 4096$  bajtów = 4 kB



## Pamięć wirtualna

- **pamięć wirtualna** umożliwia przechowywanie stron/segmentów wykonywanego procesu w pamięci dodatkowej (na dysku twardym)

Co się dzieje, gdy procesor chce odczytać stronę z pamięci dodatkowej?

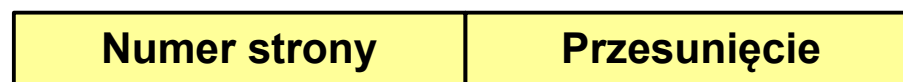
- generowanie przerwania sygnalizującego błąd w dostępie do pamięci
- zmiana stan procesu na zablokowany
- wstawienie do pamięci operacyjnej fragment procesu zawierający adres logiczny, który był przyczyną błędu
- zmiana stanu procesu na uruchomiony

Dzięki zastosowaniu pamięci wirtualnej:

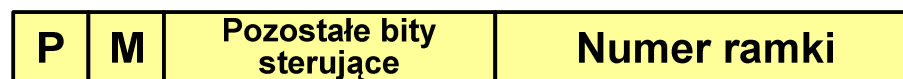
- w pamięci operacyjnej może być przechowywanych więcej procesów
- proces może być większy od całej pamięci operacyjnej

## Stronicowanie pamięci wirtualnej

- przy zastosowaniu stronicowania, **adres wirtualny** (logiczny) ma postać:



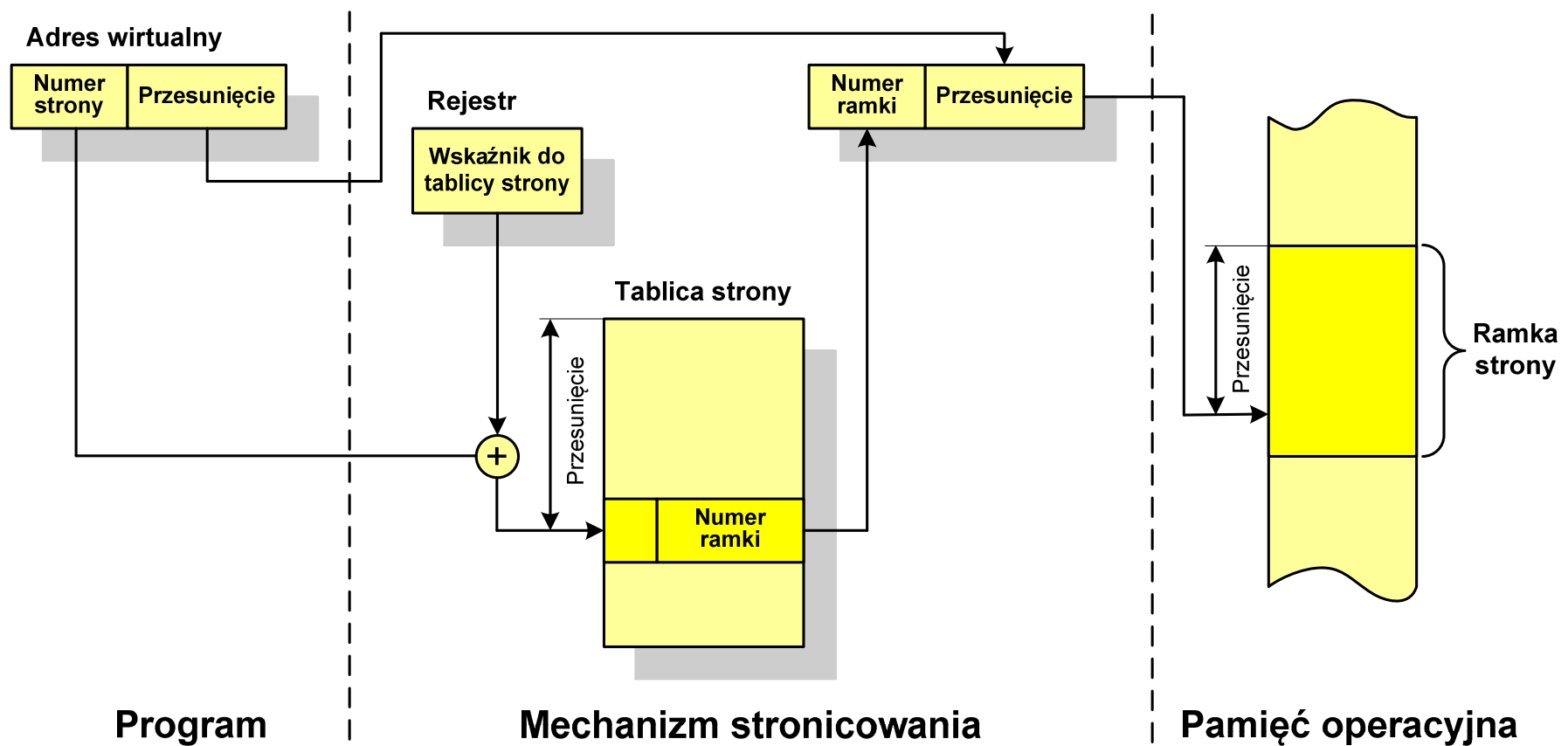
- mechanizm pamięci wirtualnej bazującej na stronicowaniu wymaga również tablicy stron



- **P** - bit określający, czy strona znajduje się w pamięci operacyjnej, jeśli tak, to zapis zawiera numer ramki tej strony
- **M** - bit określający, czy zawartość strony skojarzonej z tą tablicą została zmodyfikowana od ostatniego załadowania tej strony do pamięci - jeśli nie, to nie trzeba tej strony zapisywać, gdy ma być ona przeniesiona do pamięci pomocniczej

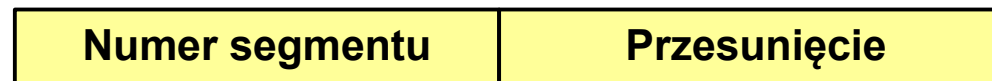
## Stronicowanie pamięci wirtualnej

- odczytanie strony wymaga translacji adresu wirtualnego na fizyczny

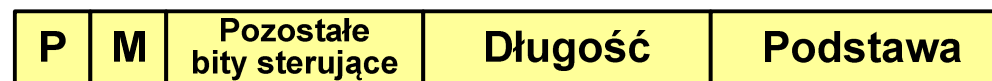


## Segmentacja pamięci wirtualnej

- w przypadku segmentacji, **adres wirtualny** ma postać:



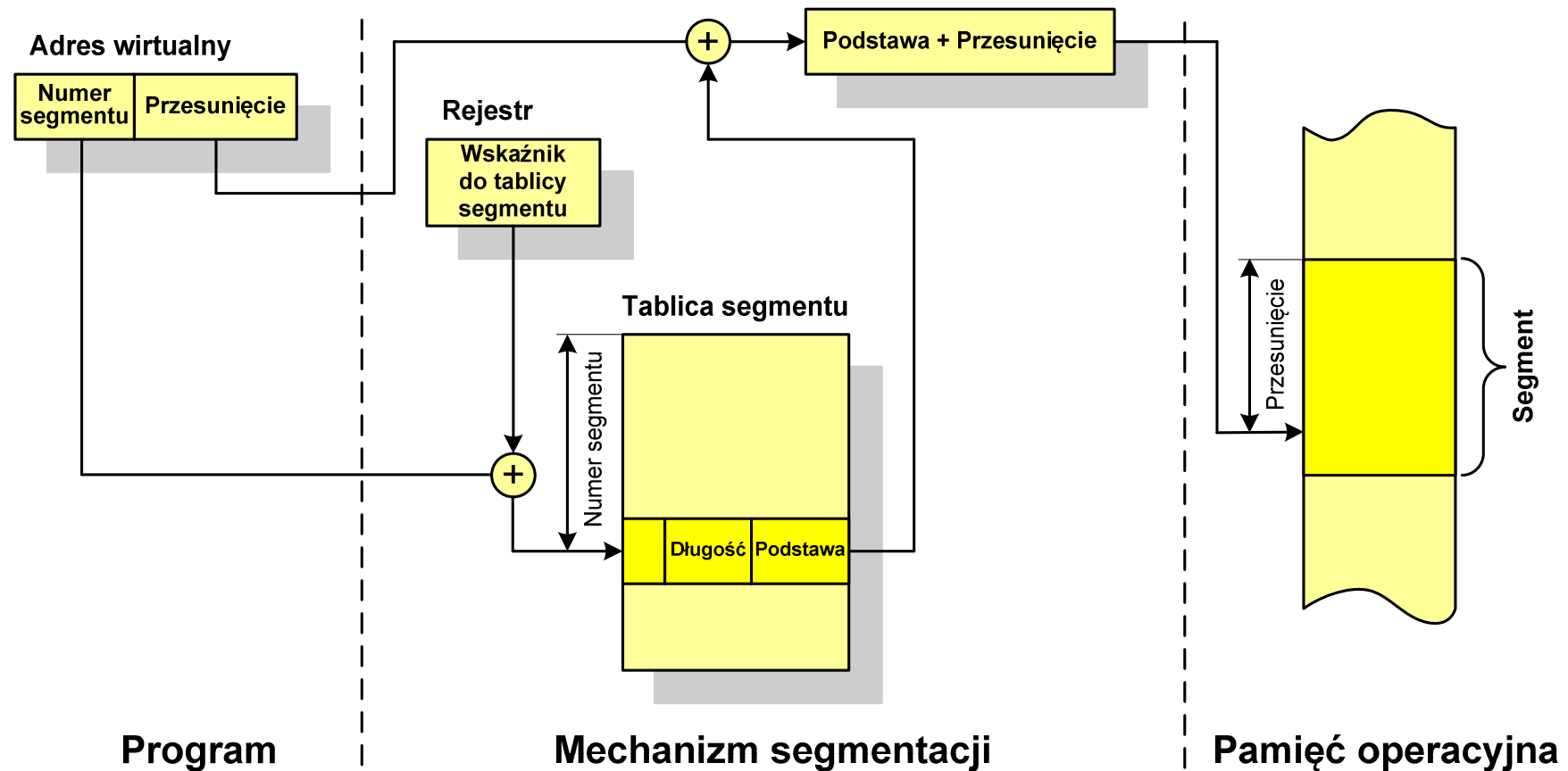
- mechanizm pamięci wirtualnej wykorzystujący segmentację wymaga **tablicy segmentu** zawierającej więcej pól



- **P** - bit określający, czy segment znajduje się w pamięci operacyjnej
- **M** - bit określający, czy zawartość segmentu skojarzonego z tablicą została zmodyfikowana od ostatniego załadowania tego segmentu do pamięci

## Segmentacja pamięci wirtualnej

- mechanizm odczytania słowa z pamięci obejmuje translację adresu wirtualnego na fizyczny za pomocą tablicy segmentu

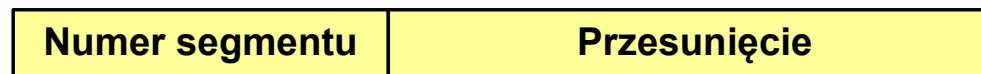




## Stronicowanie i segmentacja pamięci wirtualnej

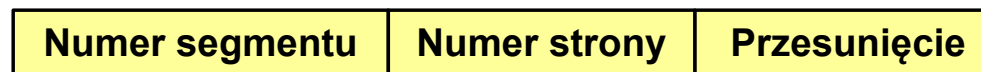
- przestrzeń adresowa użytkownika jest dzielona na dowolną liczbę **segmentów** według uznania programisty
- każdy segment jest dzielony na dowolną liczbę **stron** o stałym rozmiarze równym długości ramki pamięci operacyjnej
- z punktu widzenia programisty adres logiczny składa się z numeru segmentu oraz jego przesunięcia

Adres wirtualny



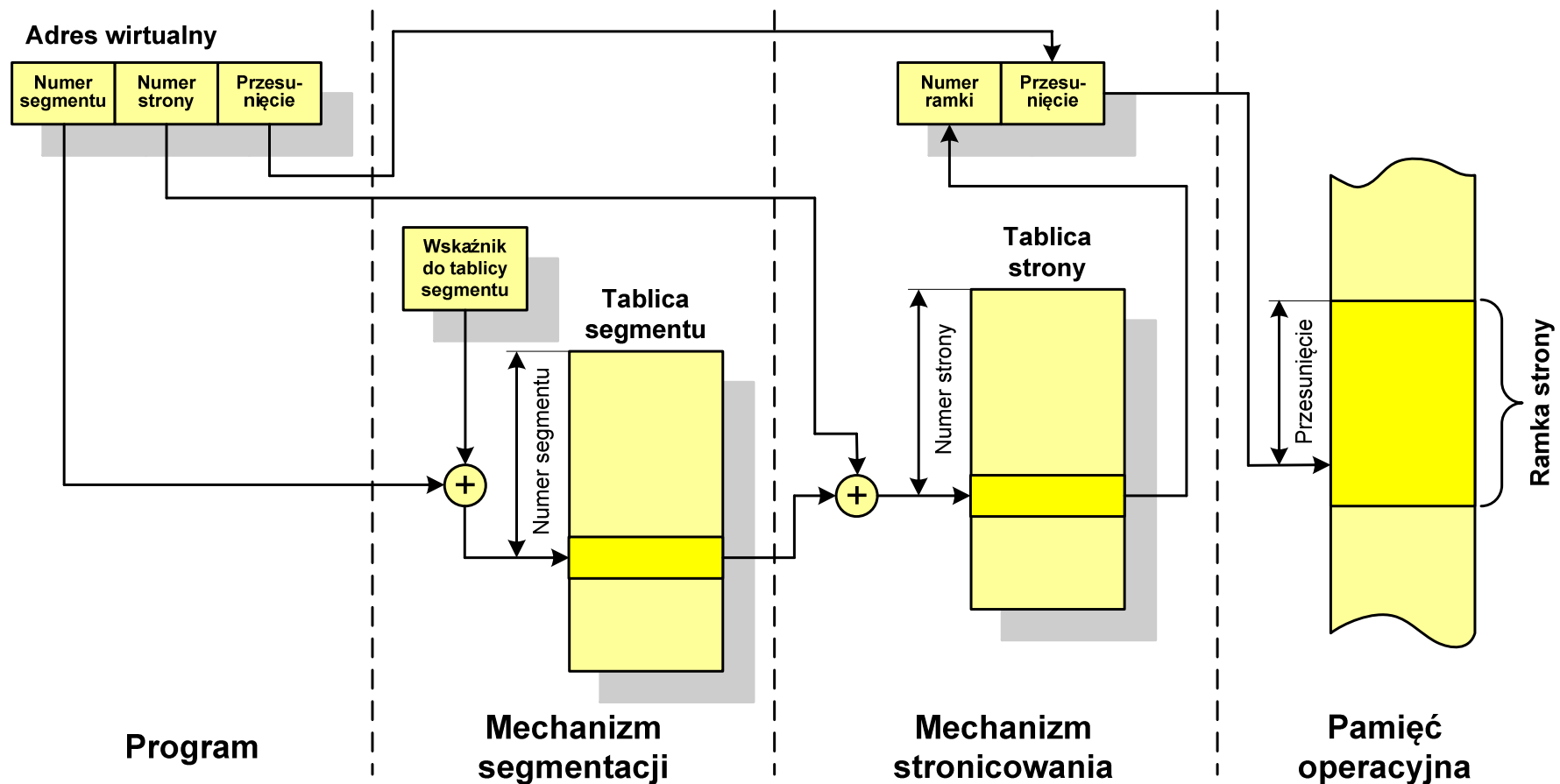
- z punktu widzenia systemu, przesunięcie segmentu jest postrzegane jako numer strony oraz przesunięcie strony dla strony wewnątrz określonego segmentu

Adres wirtualny



# Stronicowanie i segmentacja pamięci wirtualnej

- tłumaczenie adresu wirtualnego na adres fizyczny:



Koniec wykładu nr 12

Dziękuję za uwagę!