Oprogramowanie systemowe

Dariusz Wawrzyniak


Politechnika Poznańska
Instytut Informatyki
ul. Piotrowo 2 (CW, pok. nr 5)
60-965 Poznań


Dariusz.Wawrzyniak@cs.put.poznan.pl


www.cs.put.poznan.pl/dwawrzyniak


+48 (61) 665 29 63

Program przedmiotu

1. Wprowadzenie — wielowarstwowa organizacja systemu komputerowego
2. Wielozadaniowość — koncepcja procesu, przełączanie kontekstu, szeregowanie zadań
3. Zarządzanie pamięcią — przydział i podział pamięci, transformacja adresów, stronicowanie i segmentacja, wymiana stron
4. Urządzenia wejścia-wyjścia — struktura mechanizmu wejścia-wyjścia, interakcja jednostki centralnej z urządzeniami wejścia-wyjścia, buforowanie i spooling
5. Zarządzanie informacją — organizacja systemu plików

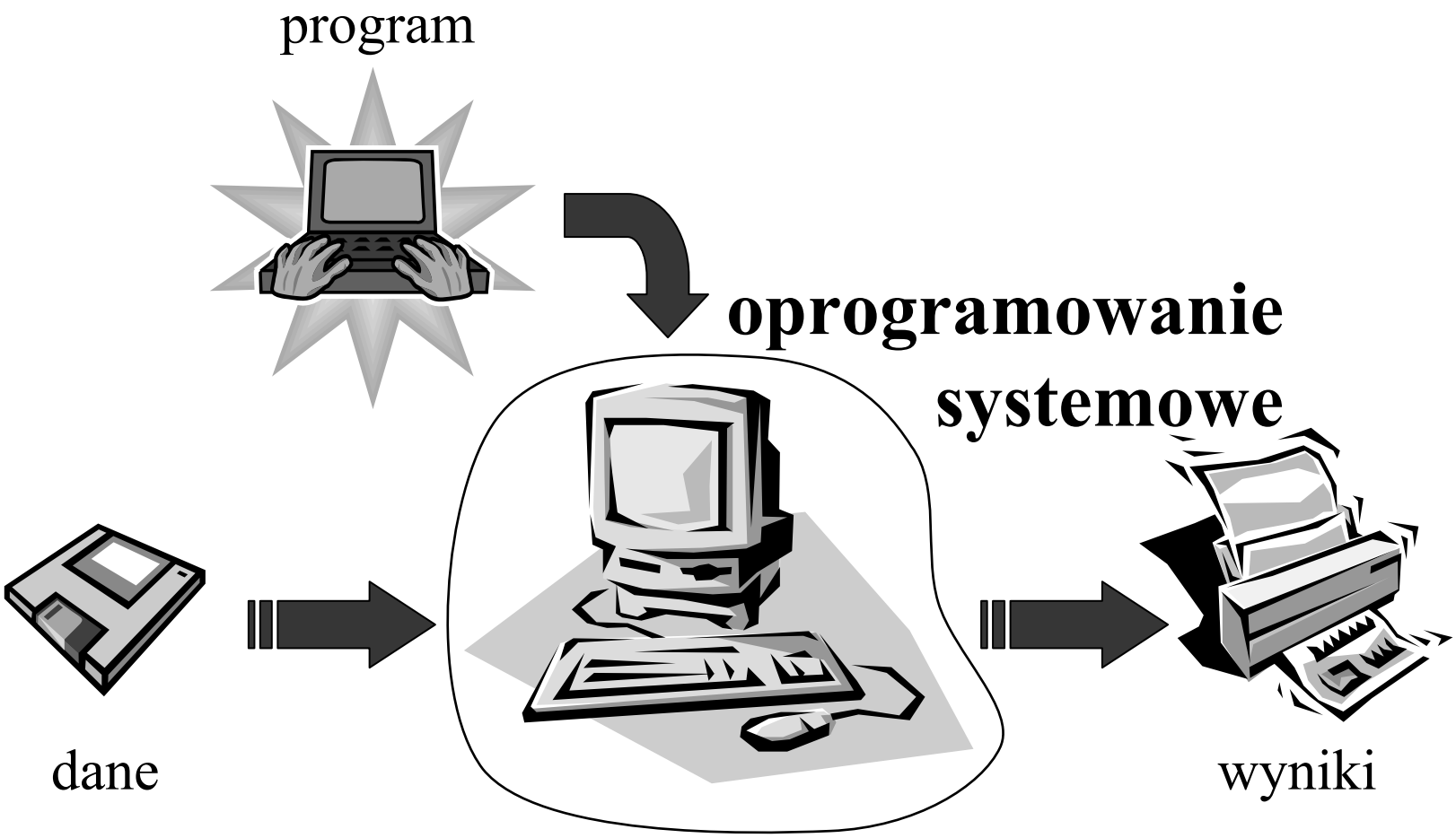
Literatura

1. A. Silberschatz, J.L. Peterson, G. Gagne: *Podstawy systemów operacyjnych*. WNT, W-wa, 2005.
2. W. Stallings: *Systemy operacyjne*, Wydawnictwo Robomatic, Wrocław, 2004.
3. A. S. Tanenbaum: *Structured computer organization*. wyd. 4, Prentice-Hall, Inc, 1999.
4. G. Nutt: *Operating Systems. A modern Perspective*, Addison Wesley Longman, Inc., 2000.
5. L. Null, J. Lobur: *Struktura organizacyjna i architektura systemów komputerowych*, Wydaw. HELION, 2004

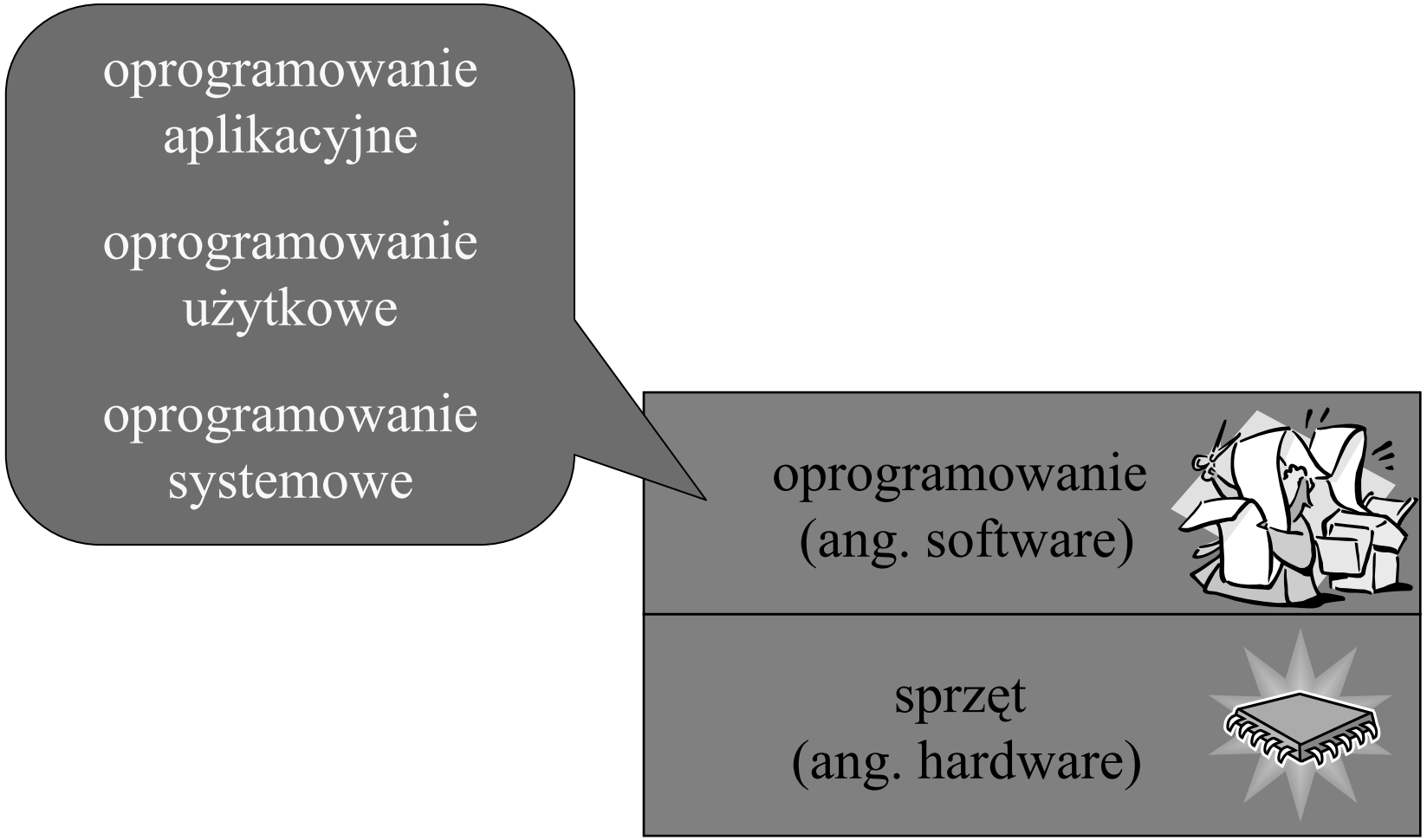
Wprowadzenie

1. Organizacja przetwarzania danych
2. Struktura systemu komputerowego
3. Oprogramowanie systemu komputerowego
4. Podejście wielowarstwowe
5. Translacja i interpretacja
6. Pojęcie organizacji i architektury systemu komp.
7. Wielowarstwowa struktura systemu komputerowego

Organizacja przetwarzania danych



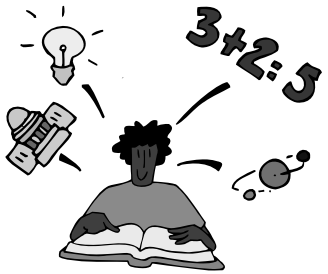
Struktura systemu komputerowego



Oprogramowanie

- aplikacyjne — zbiór programów do przetwarzania danych
- użytkowe — zbiór programów ułatwiających pracę i poruszanie się użytkownika w systemie komputerowym (edytory, eksploratory, kompilatory, debuggery, profilery itp.)
- oprogramowanie systemowe — zbiór narzędzi do automatycznego lub „ręcznego” zarządzania zasobami systemu komputerowego (np. system operacyjny)

Wielowarstwowe ujęcie struktury systemu komputerowego



program w języku „wygodnym”
dla człowieka

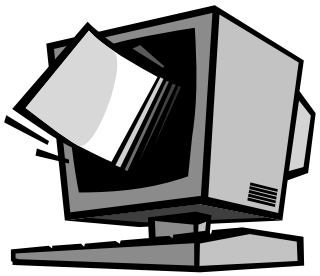
przekład na język L_n

przekład na język L_{n-1}

...

przekład na język L_1

program w języku „wygodnym”
dla komputera (L_0)



Maszyna wirtualna

- Maszyna wirtualna warstwy k — M_k — rozumiana jest jako logiczny komputer, którego językiem maszynowym jest język L_k, a realizacja programu w tym języku polega na zleceniu wykonania odpowiednich instrukcji w języku $L_{k\text{-}1}$ maszynie $M_{k\text{-}1}$.

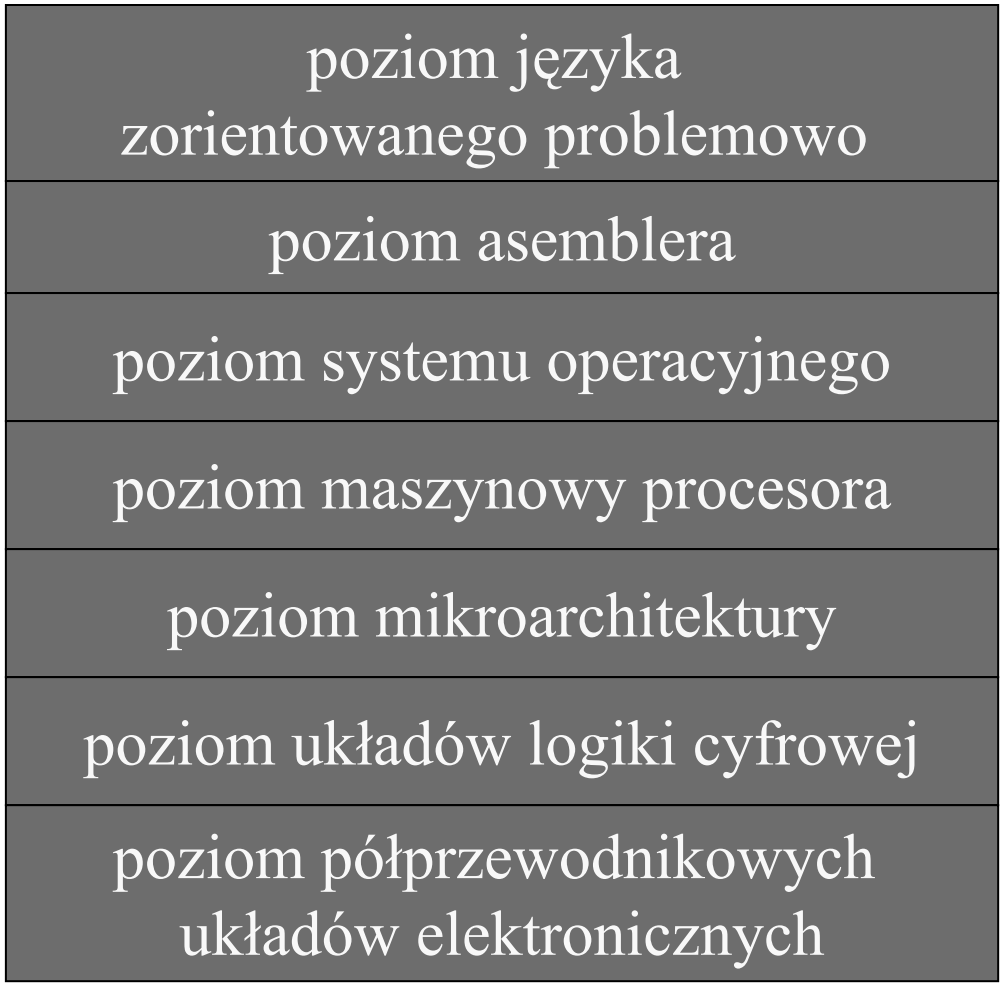
Translacja i interpretacja

- translacja — przełożenie całego ciągu instrukcji w języku warstwy wyższej — L_k — na równoważny ciąg instrukcji w języku warstwy niższej — $L_{k\text{-}1}$, a następnie realizacja uzyskanego programu w języku $L_{k\text{-}1}$ przez maszynę $M_{k\text{-}1}$.
- interpretacja — sukcesywne pobieranie instrukcji programu w języku warstwy wyższej — L_k, zamiana pobranej instrukcji na odpowiadający jej ciąg instrukcji w języku warstwy niższej — $L_{k\text{-}1}$, a następnie realizacja uzyskanego ciągu przez maszynę $M_{k\text{-}1}$.

Pojęcie organizacji i architektury systemu komputerowego

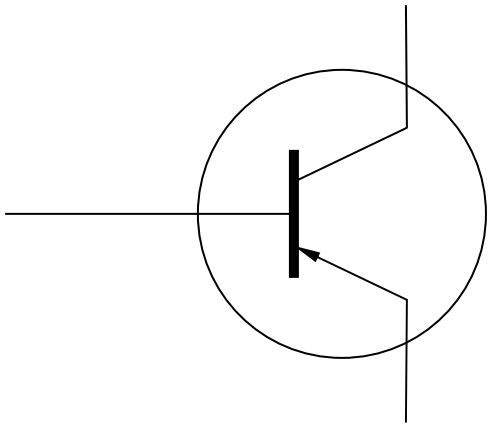
- **Architektura** systemu komputerowego obejmuje te jego elementy, które są istotne dla programisty (wpływają bezpośrednio na logiczne wykonywanie programu i tym samym sposób jego konstrukcji), np. lista instrukcji, typy danych, sposób adresowania pamięci itp.
- **Organizacja** systemu komputerowego obejmuje powiązania jednostek funkcjonalnych, które mogą wpływać na sposób wykonania instrukcji, ale nie mają bezpośredniego wpływu na logikę działania programu lub wynik jego działania, np. technologia wykonania, częstotliwość pracy procesora itp.

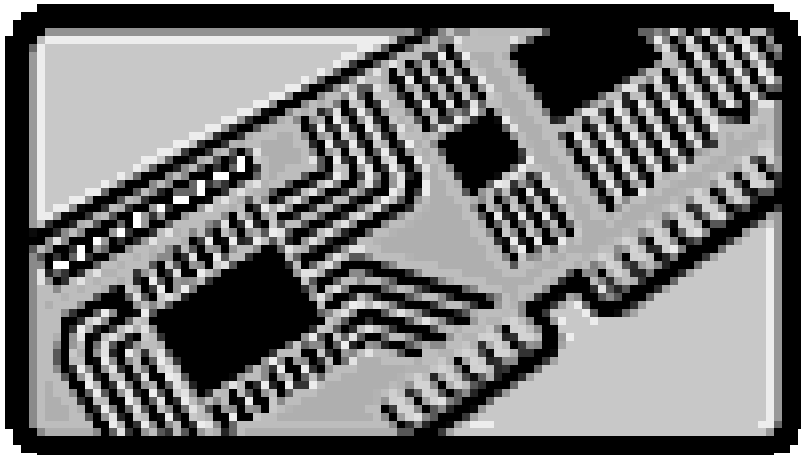
Wielowarstwowa struktura systemu komputerowego



Poziom półprzewodnikowych układów elektronicznych

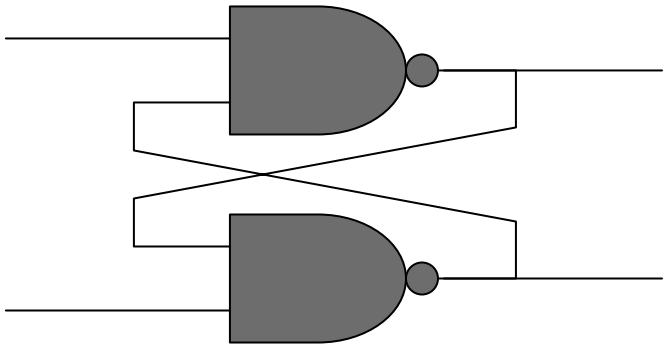
- Podstawowy element — tranzystor
- Informacja reprezentowana jest przez sygnał elektryczny





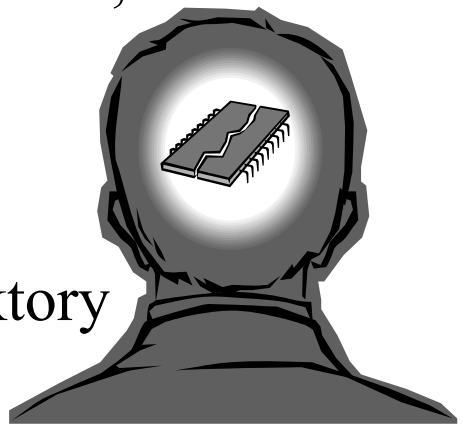
Poziom układów logiki cyfrowej

- Podstawowy element — bramka logiczna (AND, NAND, OR, NOR)
- Informacja reprezentowana jest przez stan logiczny wejść i wyjść układów cyfrowych



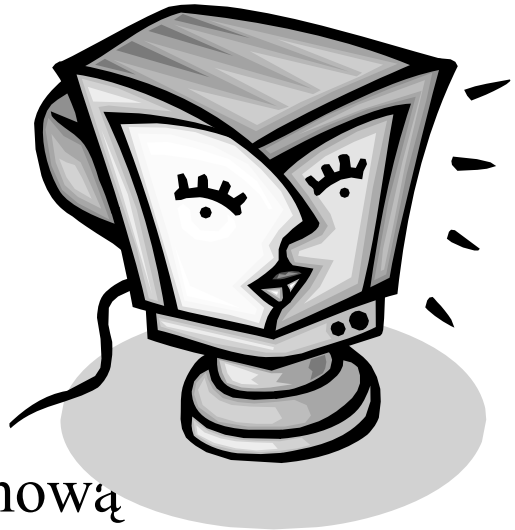
Poziom mikroarchitektury

- Podstawowe elementy — rejestry wewnętrzne, jednostka arytmetyczno-logiczna (ALU), jednostka zmiennoprzecinkowa (FPU), jednostka zarządzania pamięcią (MMU)
- Informacja reprezentowana jest przez wektory bitów (bajty, słowa)
- Poziom mikroarchitektury może być zrealizowany sprzętowo lub programowo (mikroprogram)



Poziom maszynowy procesora

- Podstawowe elementy — rejestry procesora, lista rozkazów, tryby adresowania pamięci
- Informacja reprezentowana jest przez niskopoziomowe struktury danych (liczby całkowite ze znakiem i bez znaku, liczby zmiennopozycyjne, liczby w kodzie BCD, adresy, łańcuchy znaków)
- Poziom maszynowy definiuje niskopoziomową architekturę systemu komputerowego



Poziom systemu operacyjnego

- Poziom hybrydowy — nie ukrywa poziomu maszynowego przed wyższymi warstwami, ale raczej uzupełnia go o dodatkowe mechanizmy
- Podstawowe elementy — elementy poziomu maszynowego procesora uzupełnione o organizację pamięci, interakcję z urządzeniami wejścia-wyjścia, system plików
- Informacja reprezentowana jest przez pliki

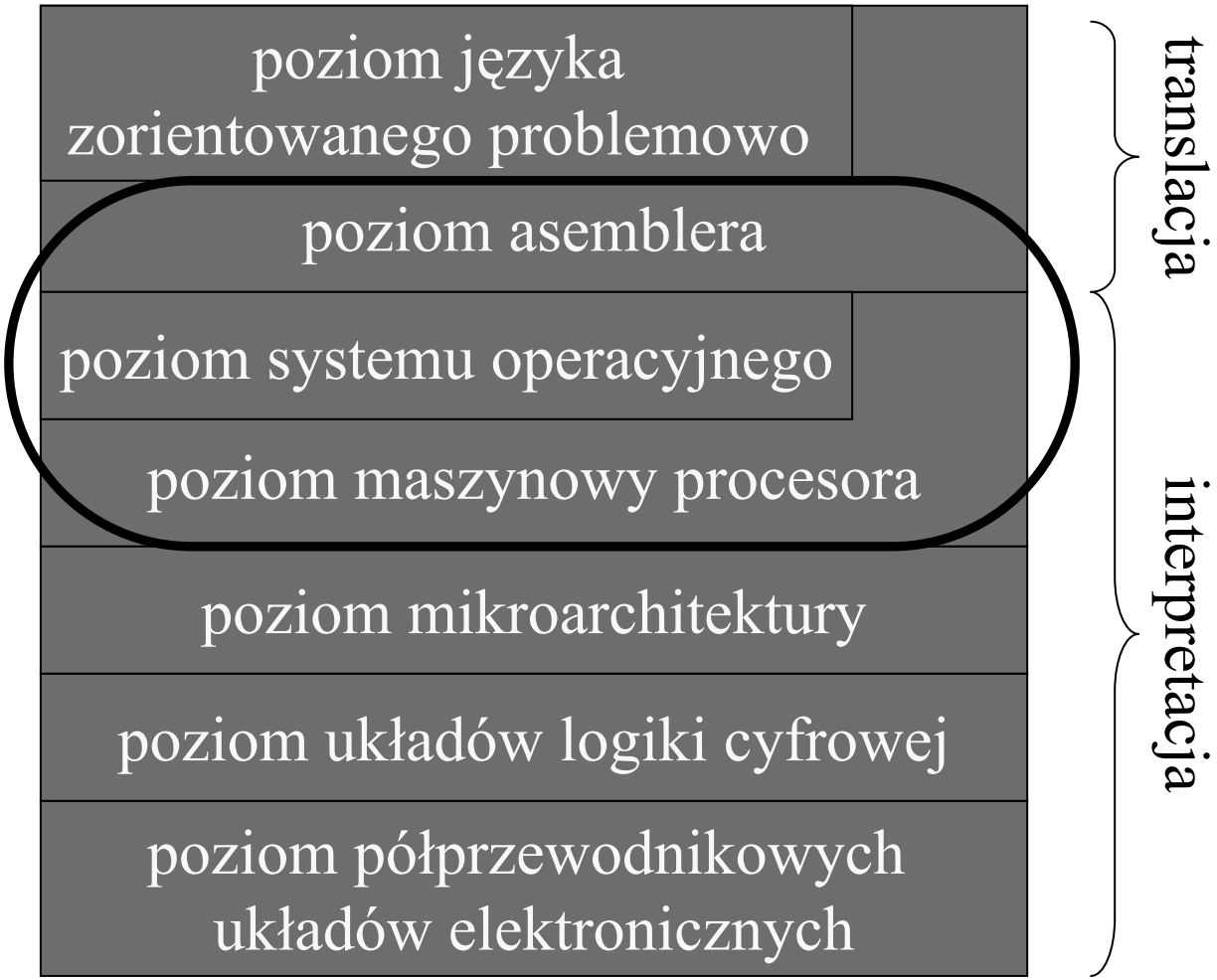
Poziom asemblera

- Udostępnia funkcjonalność poziomu maszynowego oraz poziomu systemu operacyjnego w formie symbolicznej, łatwiejszej do opanowania dla programisty (mnemoniki, etykiety)

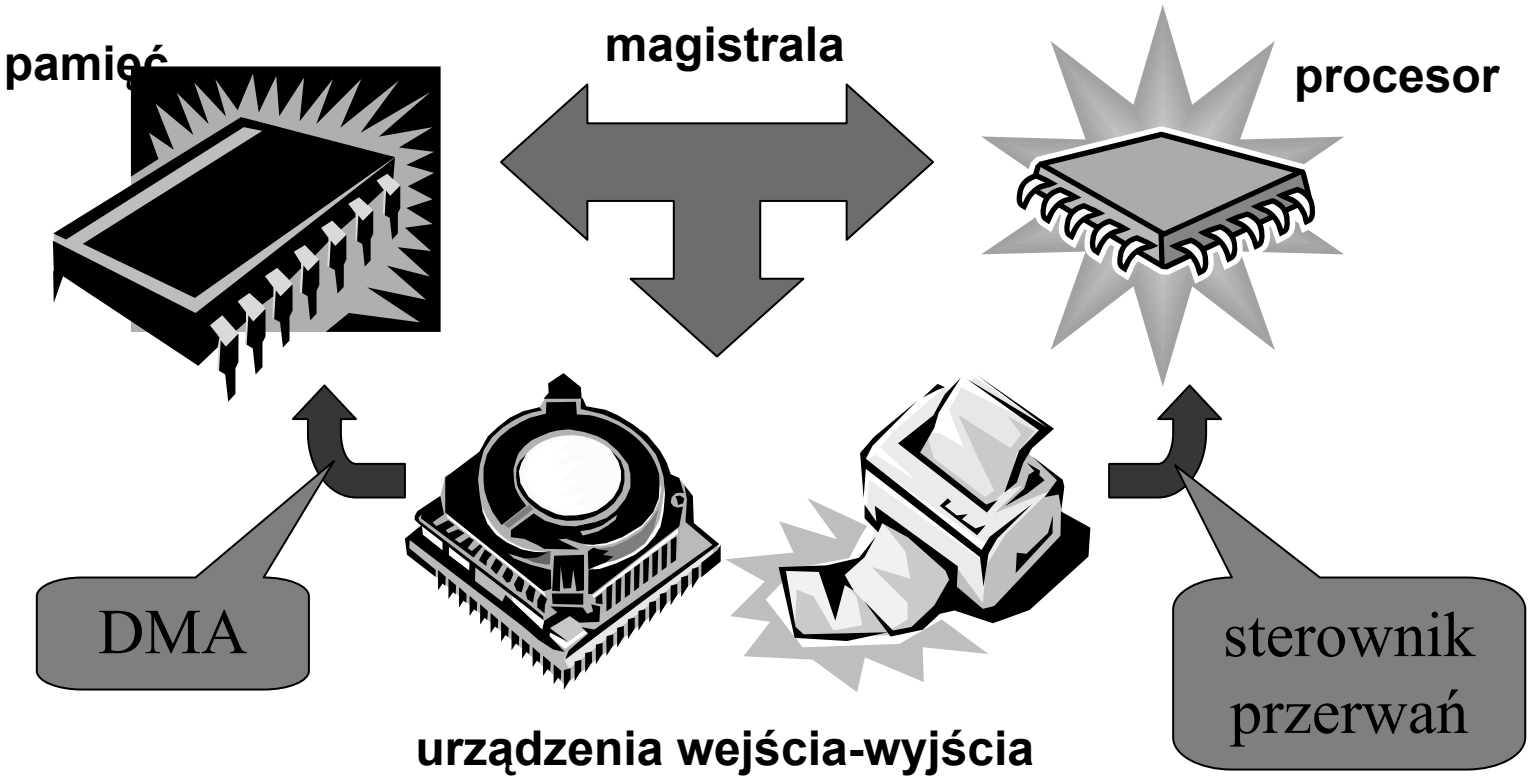
Poziom języka zorientowanego problemowo

- Ukrywa szczegóły architektury procesora oraz redefiniuje interfejs dostępu do niektórych usług systemu operacyjnego w celu uniezależnienia oprogramowania od konkretnego systemu komputerowego
- Udostępnia funkcjonalność (instrukcje, struktury danych) dostosowaną do konkretnych zastosowań (np. dostęp do baz danych, zastosowania numeryczne itp.)

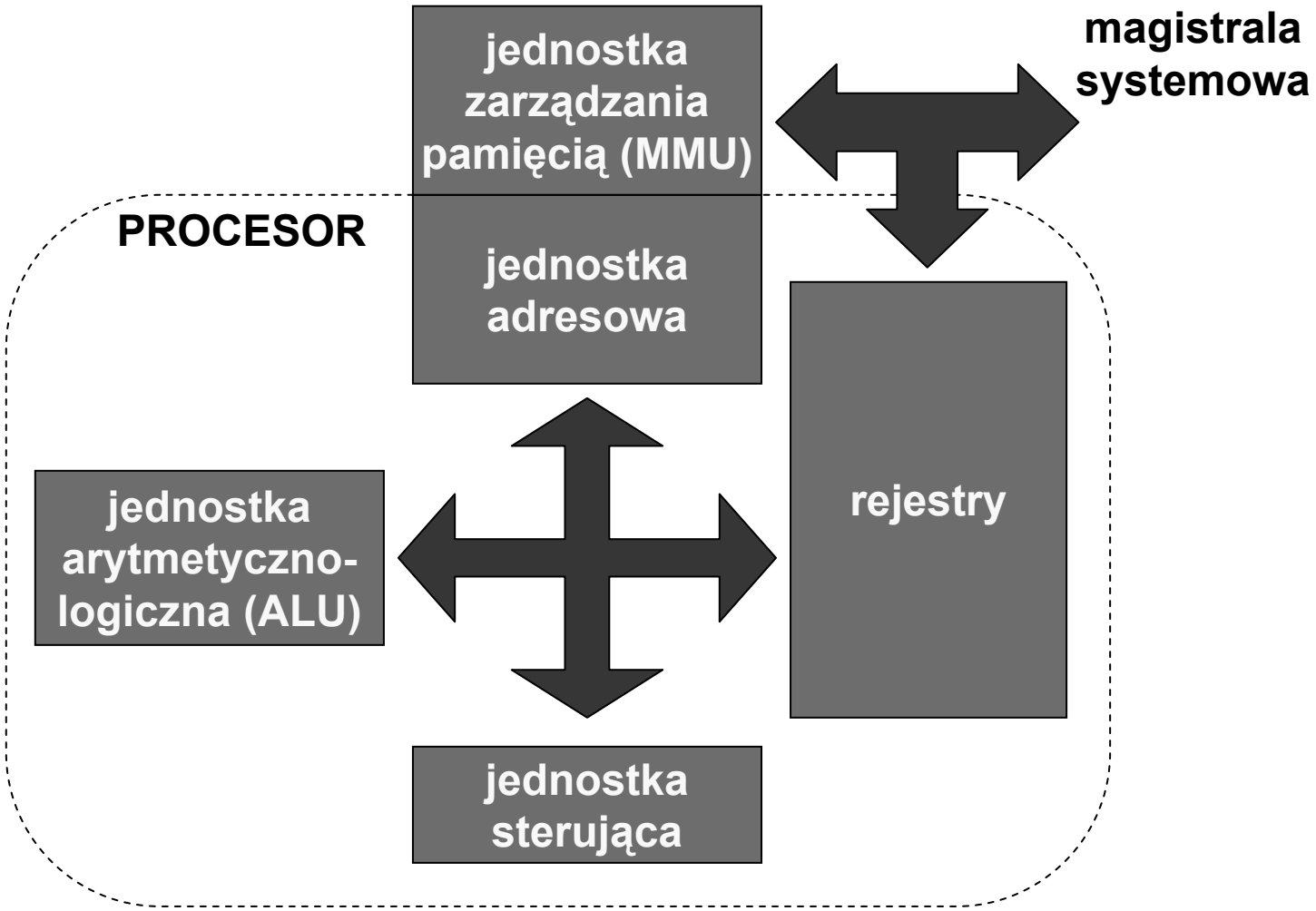
Wielowarstwowa struktura systemu komputerowego — podsumowanie



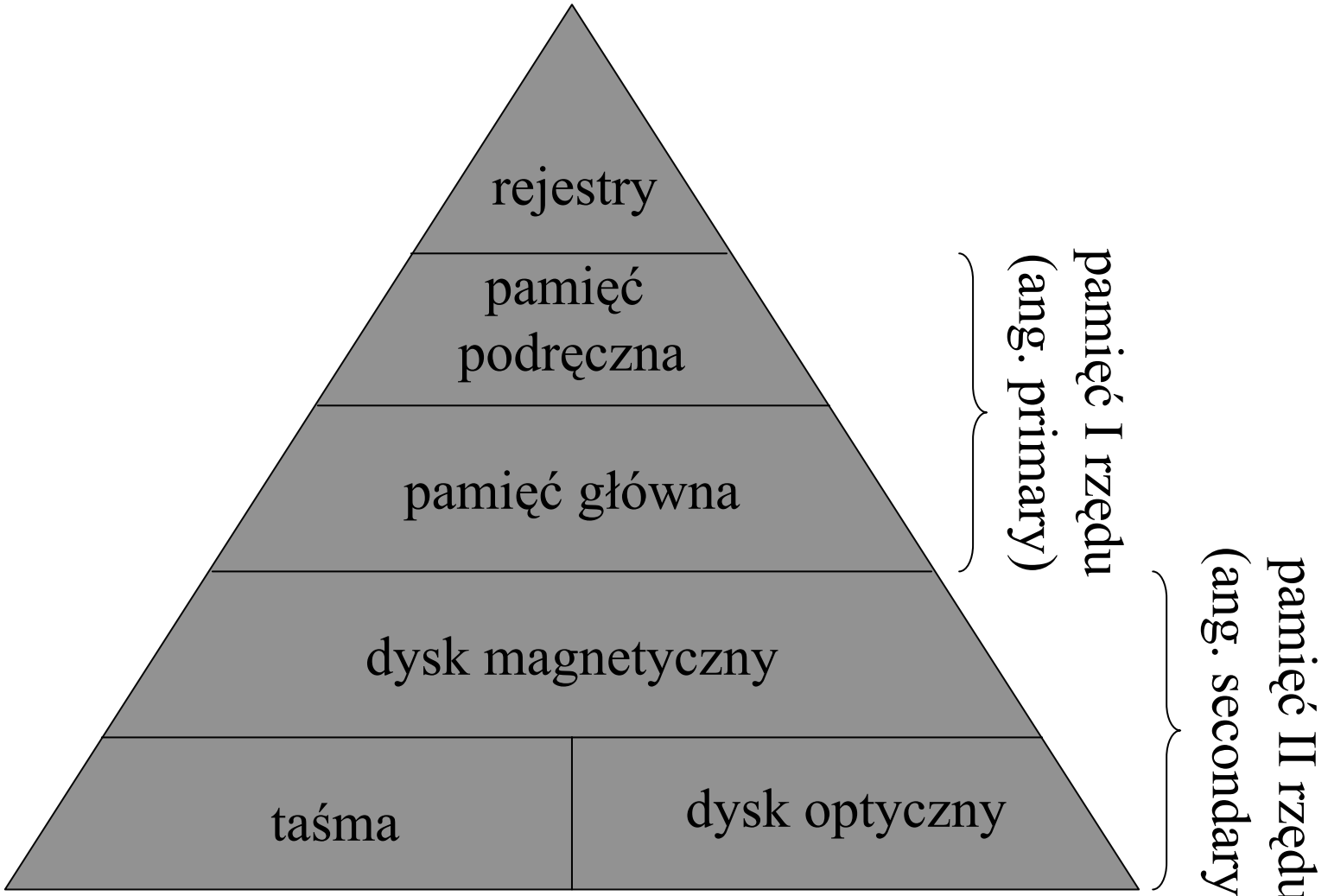
Architektura komputera z punktu widzenia poziomu maszynowego



Organizacja procesora



Hierarchia pamięci



System operacyjny

1. Definicja systemu operacyjnego
2. Klasyfikacja systemów operacyjnych
3. Zadania systemu operacyjnego
4. Zarządzanie zasobami systemu komputerowego
5. Zasoby zarządzane przez system operacyjny
6. Zasada działania systemu operacyjnego
7. Ogólna struktura systemu operacyjnego

Definicja systemu operacyjnego (1)

- System operacyjny jest zbiorem ręcznych i automatycznych procedur, które pozwalają grupie osób na efektywne współdzielenie urządzeń maszyny cyfrowej

  Per Brinch Hansen

- System operacyjny (nadzorczy, nadrzędny, sterujący) jest to zorganizowany zespół programów, które pośredniczą między sprzętem a użytkownikami, dostarczając użytkownikom zestawu środków ułatwiających projektowanie, kodowanie, uruchamianie i eksploatację programów oraz w tym samym czasie sterują przydziałem zasobów dla zapewnienia efektywnego działania

  Alen Shaw

Definicja systemu operacyjnego (2)

- System operacyjny jest programem, który działa jako pośrednik między użytkownikiem komputera a sprzętem komputerowym. Zadaniem systemu operacyjnego jest tworzenie środowiska, w którym użytkownik może wykonywać programy w sposób wygodny i wydajny.

  Abraham Silberschatz

- System operacyjny jest warstwą oprogramowania operującą bezpośrednio na sprzęcie, której celem jest zarządzanie zasobami systemu komputerowego i stworzenie użytkownikowi środowiska łatwiejszego do zrozumienia i wykorzystania.

  Andrew Tanenbaum

Klasyfikacja systemów operacyjnych ze względu na sposób przetwarzania

- Systemy przetwarzania bezpośredniego (ang. on-line processing systems) — systemy interakcyjne
  - występuje bezpośrednia interakcja pomiędzy użytkownikiem a systemem
  - wykonywanie zadania użytkownika rozpoczyna się zaraz po przedłożeniu
- Systemy przetwarzania pośredniego (ang. off-line processing systems) — systemy wsadowe
  - występuje istotna, nieznana zwłoka czasowa między przedłożeniem zadania a rozpoczęciem jego wykonywania
  - niemożliwa jest ingerencja użytkownika w wykonywanie zadania

Klasyfikacja systemów operacyjnych ze względu na dopuszczalną liczbę wykonywanych programów

- Systemy jednoprogramowe — niedopuszczalne jest rozpoczęcie wykonywania następnego zadania użytkownika przed zakończeniem poprzedniego
- Systemy wieloprogramowe — dopuszczalne jest istnienie jednocześnie wielu zadań (procesów), którym kolejno przydzielany jest procesor. Zwolnienie procesora następuje w wyniku
  - żądania przydziału dodatkowego zasobu
  - zainicjowaniu operacji wejścia-wyjścia
  - przekroczenia ustalonego limitu czasu (kwantu czasu) — systemy z podziałem czasu (ang. time-sharing systems)

Klasyfikacja systemów operacyjnych ze względu na liczbę użytkowników

- Systemy dla jednego użytkownika — zasoby systemu przeznaczone są dla jednego użytkownika (np. w przypadku komputerów osobistych), nie ma mechanizmów autoryzacji dostępu, a mechanizmy ochrony informacji są ograniczone
- Systemy wielodostępne — wielu użytkowników może jednocześnie korzystać ze współdzielonych zasobów systemu w taki sposób, że żaden z nich nie musi być świadomy istnienia innych użytkowników, a system synchronizuje dostęp do zasobów i gwarantuje ochronę informacji przed niepowołaną ingerencją

Inne rodzaje systemów operacyjnych

- Systemy czasu rzeczywistego (ang. real-time systems) — umożliwiają wyspecyfikowanie czasu zakończenie przetwarzania zadania, tzw. linii krytycznej (ang. deadline)
- Systemy sieciowe i rozproszone (ang. network and distributed systems) — umożliwiają zarządzanie zbiorem rozproszonych jednostek przetwarzających, czyli zbiorem jednostek (komputerów), które są zintegrowane siecią komputerową i nie współdzielą fizycznie zasobów

Zadania systemu operacyjnego

- Definicja interfejsu użytkownika
- Udostępnianie systemu plików
- Udostępnianie środowiska do wykonywania programów użytkownika
  - mechanizm ładowania i uruchamiania programów
  - mechanizmy synchronizacji i komunikacji procesów
- Sterowanie urządzeniami wejścia-wyjścia
- Obsługa podstawowej klasy błędów
- Zarządzanie zasobami (sprzęt i oprogramowanie) systemu komputerowego

Uruchamianie zadań — podstawowe pojęcia

- Użytkownik (ang. user) — jednostka zlecająca wykonywanie zadań
- Praca (ang. job) — zbiór akcji niezbędnych do realizacji określonego przetwarzania, np. sekwencja: kompilacja, załadowanie (uruchomienie) programu i wykonanie programu
- Proces (zadanie, ang. process, task) — najmniejsza jednostka aktywności zarządzana przez system operacyjny, która może ubiegać się o zasoby systemu komputerowego

Rodzaje użytkowników systemu operacyjnego

- Użytkownik końcowy — korzysta z poleceń i programów użytkowych
- Programista — w swoich programach korzysta z usług jądra (jako użytkownik końcowy głównie z edytorów, kompilatorów i debuggerów)
- Administrator — poprzez system praw ustala zasady dostępności zasobów dla użytkowników oraz rozwiązuje bieżące problemy związane w funkcjonowaniem systemu
- Programista systemowy — modyfikuje program jądra stosownie do aktualnych potrzeb użytkowników lub urządzeń

Zarządzanie zasobami systemu komputerowego

- Przydział zasobów
- Synchronizacja dostępu do zasobów
- Ochrona i autoryzacja dostępu do zasobów
- Odzyskiwanie zasobów
- Rozliczanie — gromadzenie danych o wykorzystaniu zasobów

Zasoby zarządzane przez system operacyjny

- Procesor — przydział czasu procesora
- Pamięć
  - alokacja przestrzeni adresowej dla procesów
  - transformacja adresów
- Urządzenia zewnętrzne
  - udostępnianie i sterowanie urządzeniami pamięci masowej
  - alokacja przestrzeni dyskowej
  - udostępnianie i sterownie drukarkami, skanerami itp.
- Informacja (system plików)
  - organizacja i udostępnianie informacji
  - ochrona i autoryzacja dostępu do informacji

Zasada działania systemu operacyjnego

- Odwołania do jądra systemu przez system przerwań lub specjalne instrukcje (przerwanie programowe)
- Dualny tryb pracy — tryb użytkownika (ang. user mode) i tryb systemowy (ang. system mode)
- Wyróżnienie instrukcji uprzywilejowanych, wykonywanych tylko w trybie systemowym
- Sprzętowa ochrona pamięci
- Uprzywilejowanie instrukcji wejścia-wyjścia
- Przerwanie zegarowe

Przerwania w systemie komputerowym

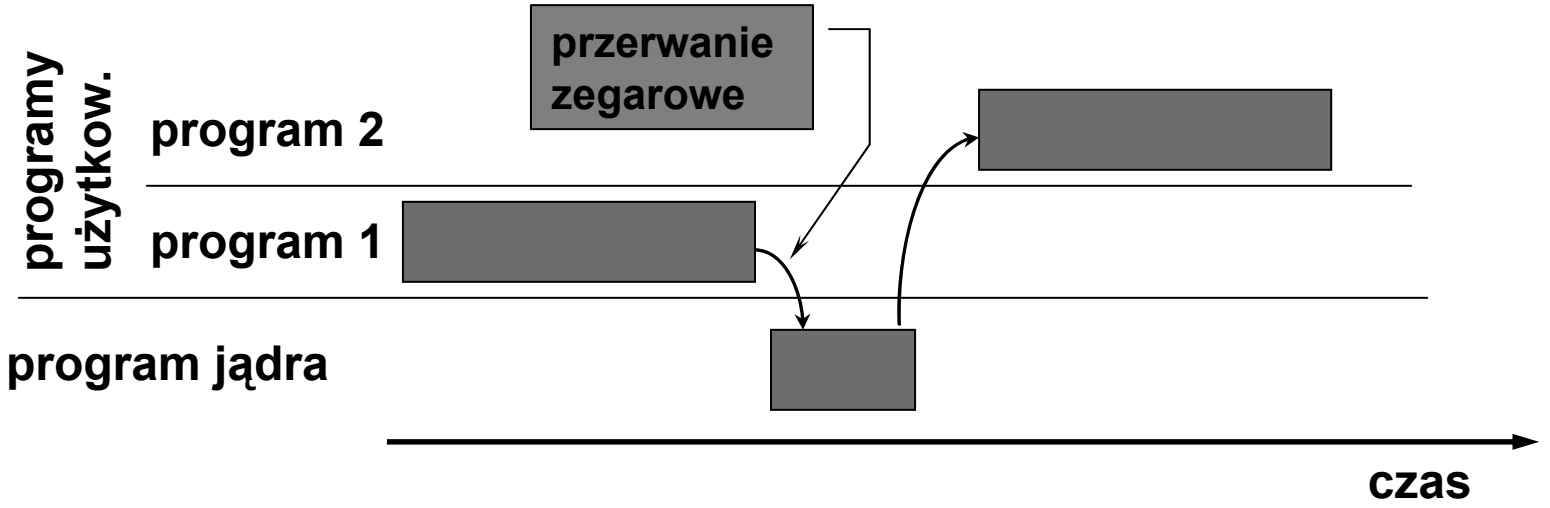
Przerwanie jest reakcją na asynchroniczne zdarzenie, polegającą na automatycznym zapamiętaniu bieżącego stanu procesora w celu późniejszego odtworzenia oraz przekazaniu sterowania do ustalonej procedury obsługi przerwania.

Podział przerwań ze względu na źródło:

- przerwania zewnętrzne — od urządzeń zewnętrznych
- przerwania programowe — wykonanie specjalnej instrukcji
- przerwania diagnostyczne — pułapki, błędy programowe i sprzętowe

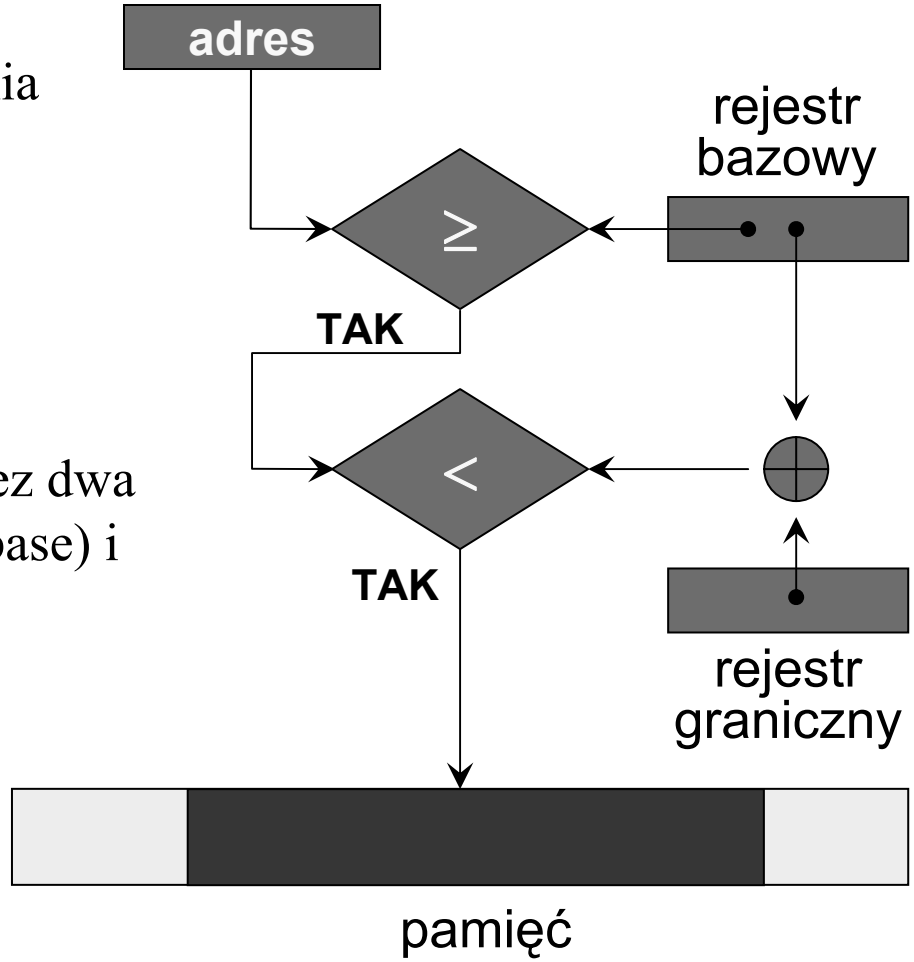
Przerwanie zegarowe

- Przerwanie zegarowe generowane jest przez czasomierz (ang. timer) po wyznaczonym okresie czasu.
- Obsługa przerwania zegarowego oznacza przekazanie sterowania do jądra systemu operacyjnego, umożliwiając w ten sposób wykonanie pewnych zdań okresowych oraz uniemożliwiając zawłaszczenie procesora przez program użytkownika.

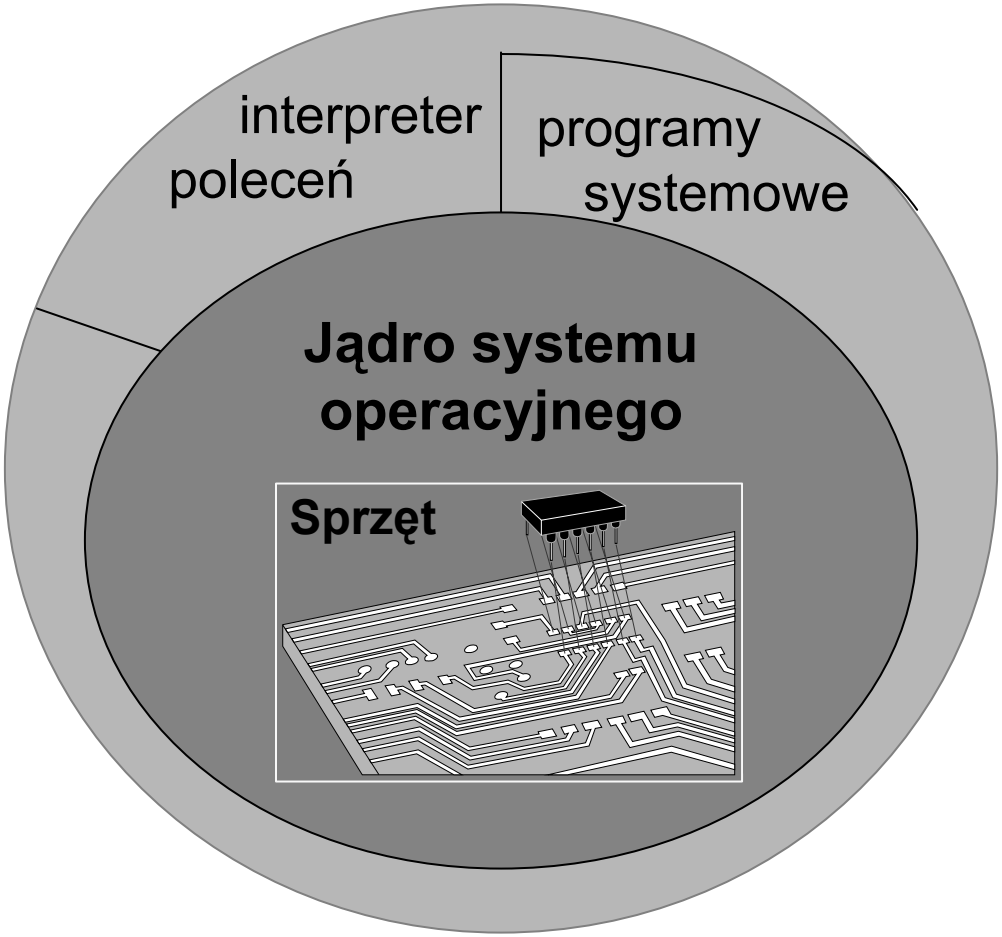


Zasady ochrony pamięci

- W wyniku wykonywania programu następuje odwołanie do komórek pamięci o określonych adresach
- Zaalokowane obszary pamięci opisane są przez dwa parametry: bazę (ang. base) i granicę (ang. limit)



Ogólna struktura systemu operacyjnego



Wielozadaniowość

1. Koncepcja procesu
2. Blok kontrolny procesu i przełączanie kontekstu
3. Kolejki procesów
4. Szeregowanie procesów (planowanie przydziału procesora
   - Planiści
   - Kryteria planowania
   - Algorytmy planowania

Koncepcja procesu

- Proces jest elementarną jednostką pracy (aktywności) zarządzaną przez system operacyjny, która może ubiegać się o zasoby systemu komputerowego
- Proces = wykonujący się program
- Zasoby potrzebne do wykonania procesu
  - czas procesora
  - pamięć operacyjna
  - pliki
  - urządzenia wejścia-wyjścia
- Proces wykonuje kod programu użytkownika (proces użytkownika) lub kod systemowy (proces systemowy)

Stany procesu

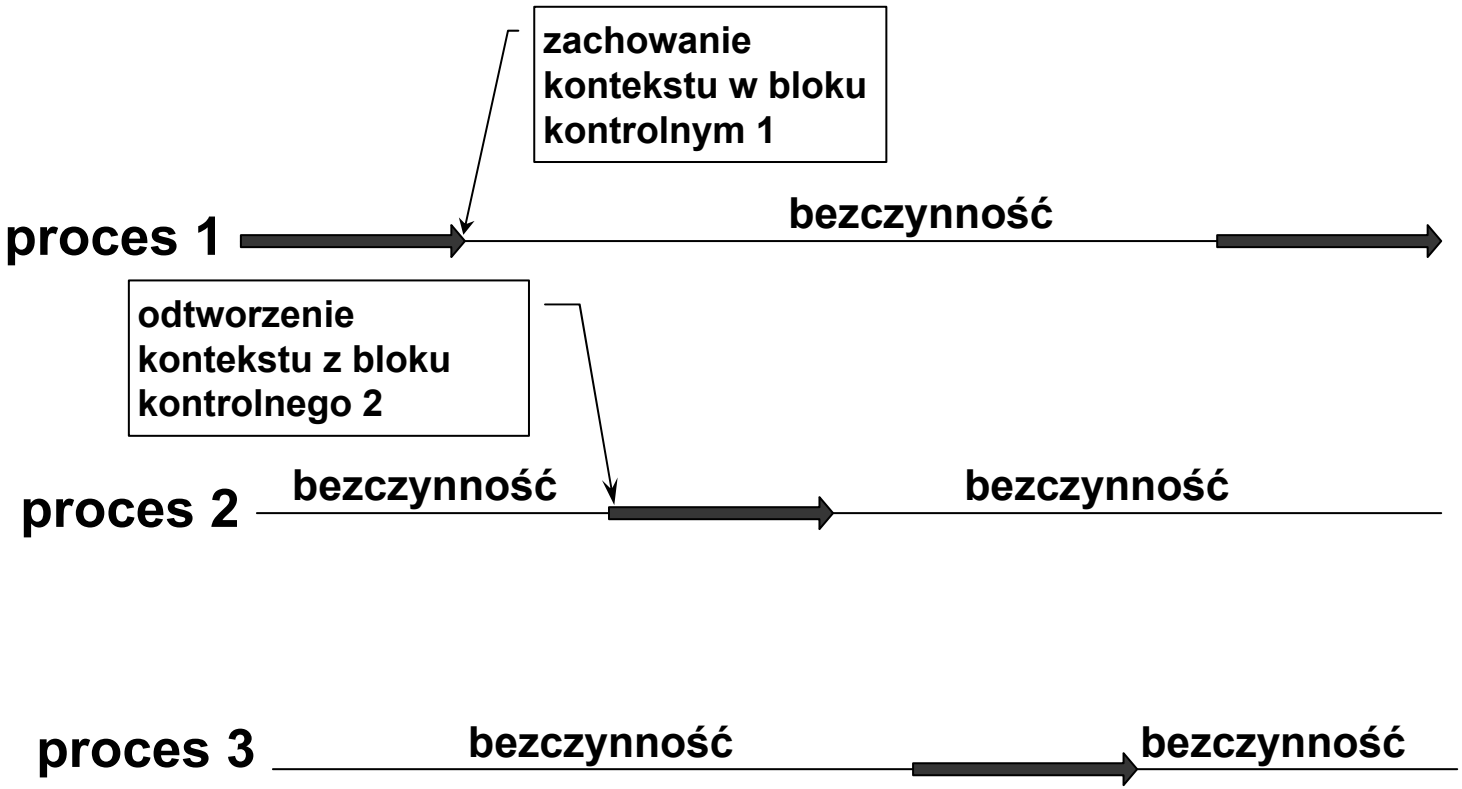
- Nowy (ang. new) — proces został utworzony
- Wykonywany (ang. active) — wykonywane są instrukcje programu
- Zawieszony (oczekujący, ang. suspanded, waiting) — proces oczekuje na jakieś zdarzenie, np. na zakończenie operacji wejścia-wyjścia, na przydział dodatkowego zasobu, synchronizuje się z innymi procesami
- Gotowy (ang. ready) — proces czeka na przydział procesora
- Zakończony (ang. terminated) — proces zakończył działanie i zwalnia zasoby

Cykl zmian stanów procesu

Blok kontrolny procesu (ang. process control block)

- Stan procesu (nowy, gotowy, aktywny, oczekujący, itd.)
- Licznik rozkazów — adres następnego rozkazu
- Rejestry procesora (akumulatory, rejestry indeksowe, wskaźniki stosu, rejestry ogólnego przeznaczenia, rejestry flagowe, rejestry warunków)
- Informacje na potrzeby planowaniu przydziału procesora (priorytet, wskaźniki do kolejek porządkujących itp.)
- Informacje o zarządzaniu pamięcią (elementy tablic stron i segmentów, zawartości rejestrów granicznych)
- Informacje o stanie wejścia-wyjścia
- Informacje do rozliczeń

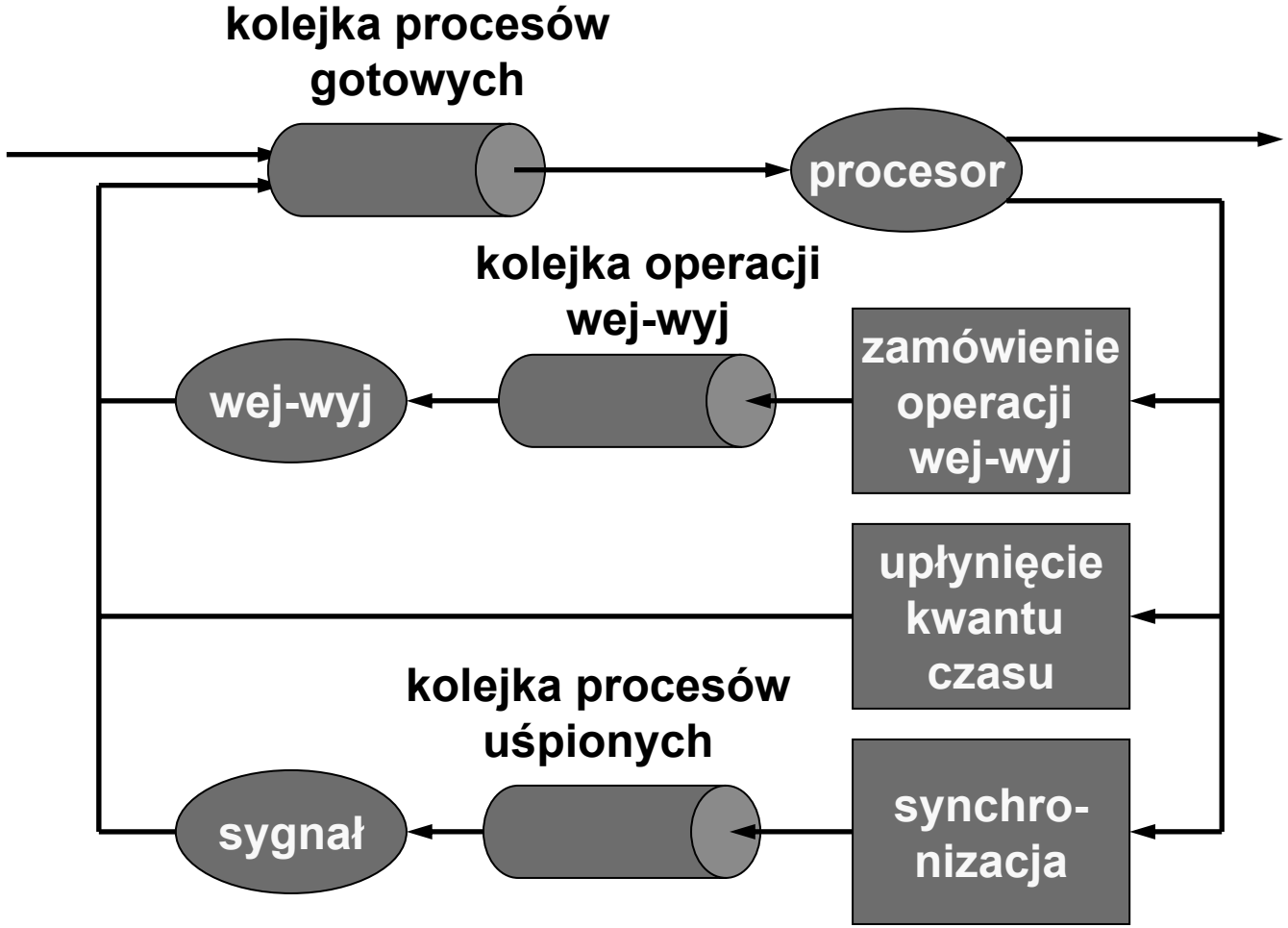
Przełączanie kontekstu (ang. context switch)



Kolejki procesów

- Kolejka zadań (ang. job queue) — wszystkie procesy systemu
- Kolejka procesów gotowych (ang. ready queue) — procesy gotowe do działania, przebywające w pamięci głównej
- Kolejka do urządzenia (ang. device queue) — procesy czekające na zakończenie operacji wejścia-wyjścia
- Kolejka procesów oczekujących w wyniku synchronizacji z innymi procesami (np. kolejka procesów na semaforze)

Diagram kolejek w planowaniu procesora



Planista (ang. scheduler)

- Planista długoterminowy, planista zadań (ang. long-term scheduler, job scheduler) — zajmuje się ładowaniem nowych programów do pamięci, kontroluje stopień wieloprogramowości, dąży do zrównoważenia procesora.
- Planista krótkoterminowy, planista przydziału procesora (ang. CPU scheduler) — zajmuje się przydziałem procesora do procesów gotowych.
- Planista średnioterminowy (ang. medium-term scheduler) — zajmuje się wymianą procesów pomiędzy pamięcią główną a pamięcią zewnętrzną (np. dyskiem).

Kryteria planowania przydziału procesora

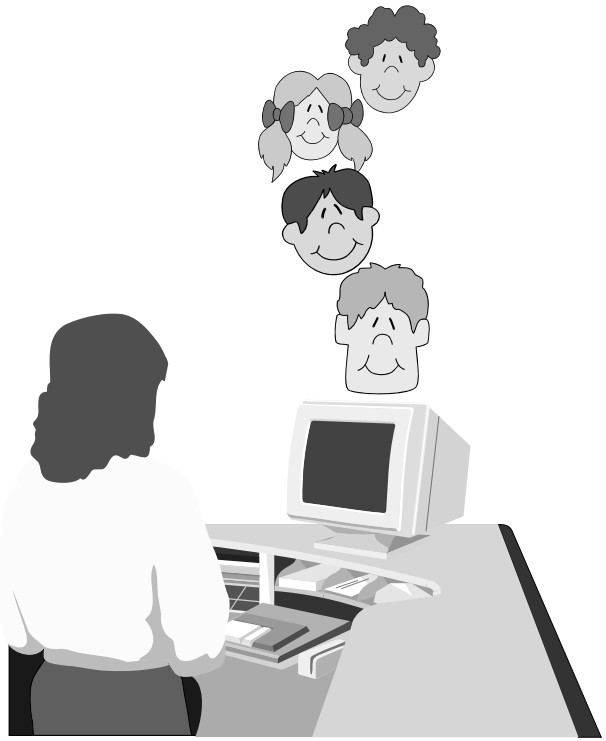
- Wykorzystanie procesora — procent czasu, przez który procesor jest zajęty pracą
- Przepustowość — liczba procesów kończonych w jednostce czasu
- Czas cyklu przetwarzania — czas pomiędzy przedłożeniem zadania, a zakończeniem jego wykonywania
- Czas oczekiwania — łączny czas spędzony przez proces w kolejce procesów gotowych
- Czas odpowiedzi — czas pomiędzy przedłożeniem zadania, a uzyskaniem pierwszej odpowiedzi

Algorytmy planowania przydziału procesora

- FCFS procesy wybierane są z kolejki procesów gotowych w kolejności, w której wchodziły do tej kolejki.
- SJF z kolejki procesów gotowych wybierany jest ten proces, który ma najkrótszą następną fazę procesora.
- Planowanie priorytetowe z kolejki procesów gotowych wybierany jest proces o najwyższym priorytecie.
- Planowanie rotacyjne po ustalonym kwancie czasu proces aktywny jest przerywany i trafia do kolejki procesów gotowych.
- Planowanie wielokolejkowe w systemie jest wiele kolejek procesów gotowych i każda z kolejek może być inaczej obsługiwana.

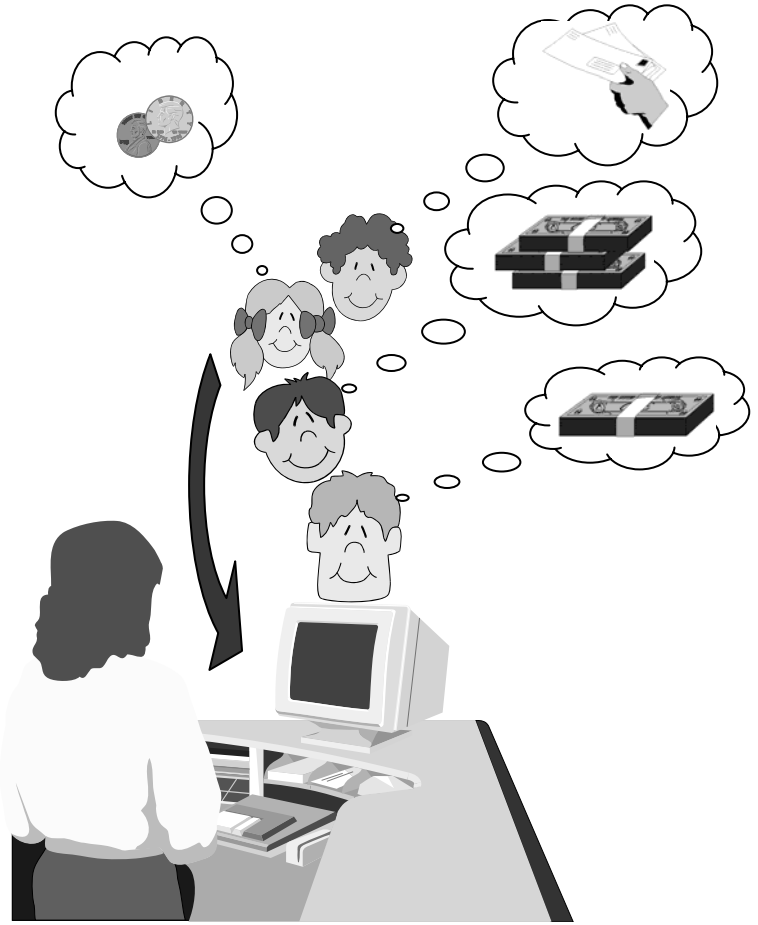
Pierwszy zgłoszony pierwszy obsłużony (ang. First Come First Served — FCFS)

- Wykonywanie procesów w kolejności zgłaszania się do systemu
- Algorytm **niewywłaszczający** (proces oddaje procesor dopiero po zażądaniu operacji wejścia-wyjścia)
- Duży rozrzut czasu oczekiwania



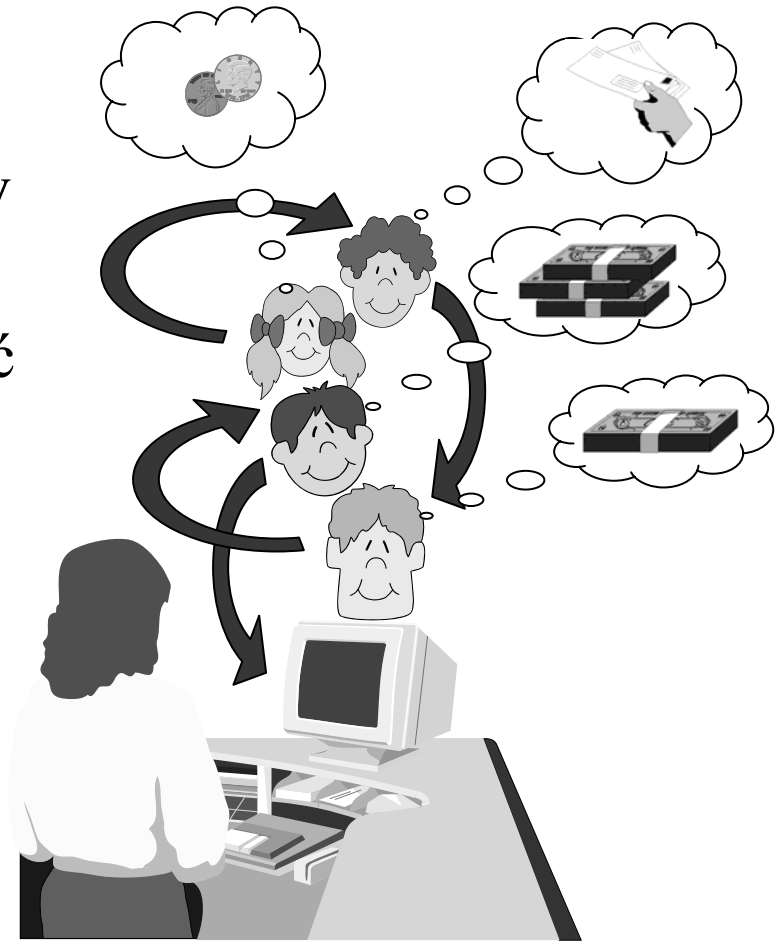
Najpierw najkrótsze zadanie (ang. Shortest Job First — SJF)

- Wybierany jest proces, który ma najkrótszą następną fazę procesora
- Daje minimalny średni czas oczekiwania dla każdego procesu



Planowanie priorytetowe (ang. priority scheduling)

- Wybierany jest proces, który ma największy priorytet
- Priorytet proces może ulegać zmianie w czasie jego cyklu życia w systemie



Planowanie rotacyjne (ang. Round Robin — RR)

- Ustalany jest kwant czasu, po upływie którego proces jest **wywłaszczany** i trafia na koniec kolejki procesów gotowych (chyba że wcześniej zażąda operacji wejścia-wyjścia)
- Preferencja dla zadań krótkich (wydłuża się czas oczekiwania i czas cyklu przetwarzania dla zadań długich)
- Przełączanie kontekstu pochłania pewien czas!!!

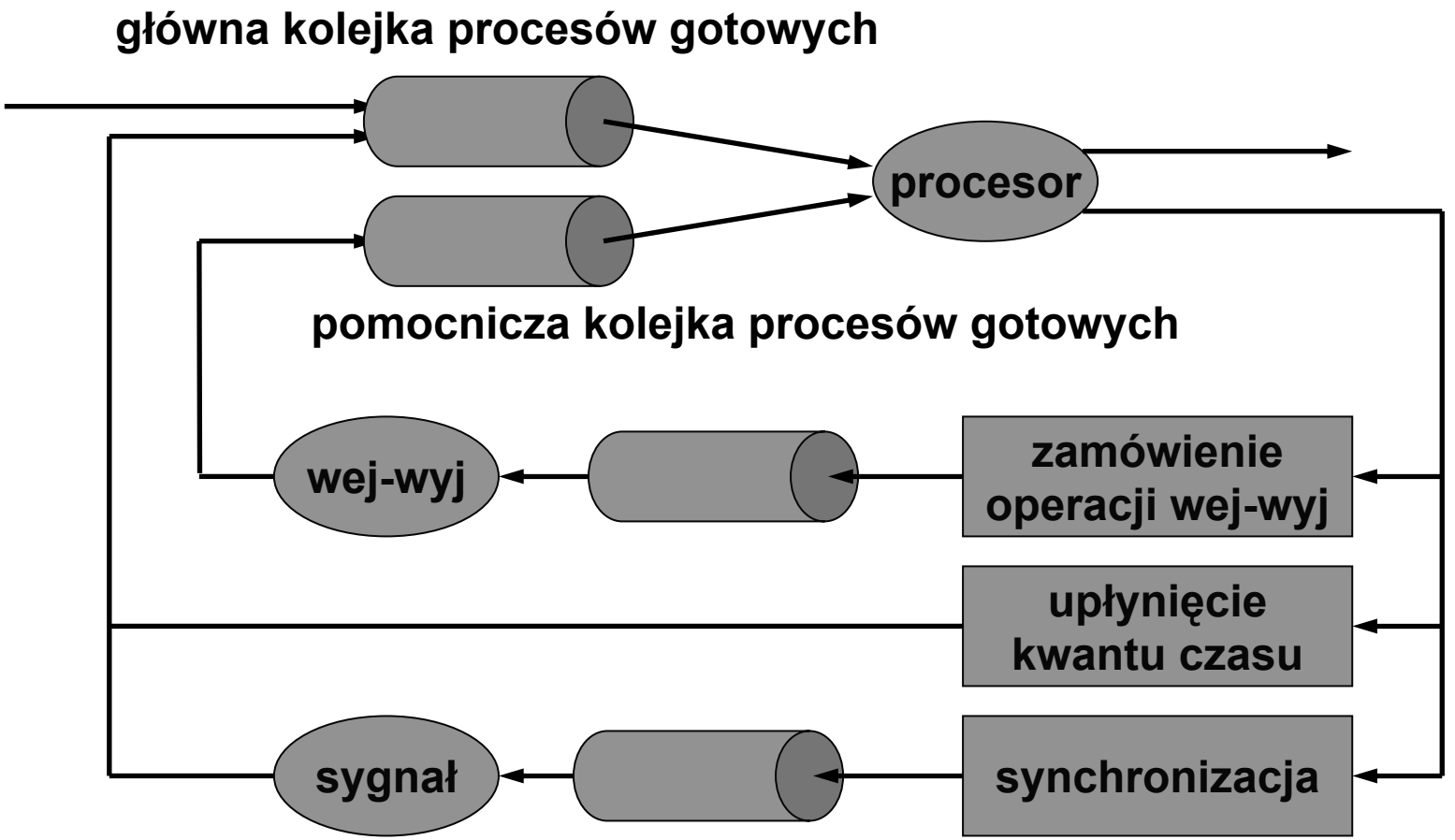
Planowanie wielokolejkowe (ang. multiple queues)

- Podział procesów na grupy (np. procesy interaktywne i procesy wsadowe) i wynikający z tego przydział do różnych kolejek procesów gotowych
- Możliwość przydziału różnych priorytetów oraz różnych algorytmów szeregowania do poszczególnych kolejek

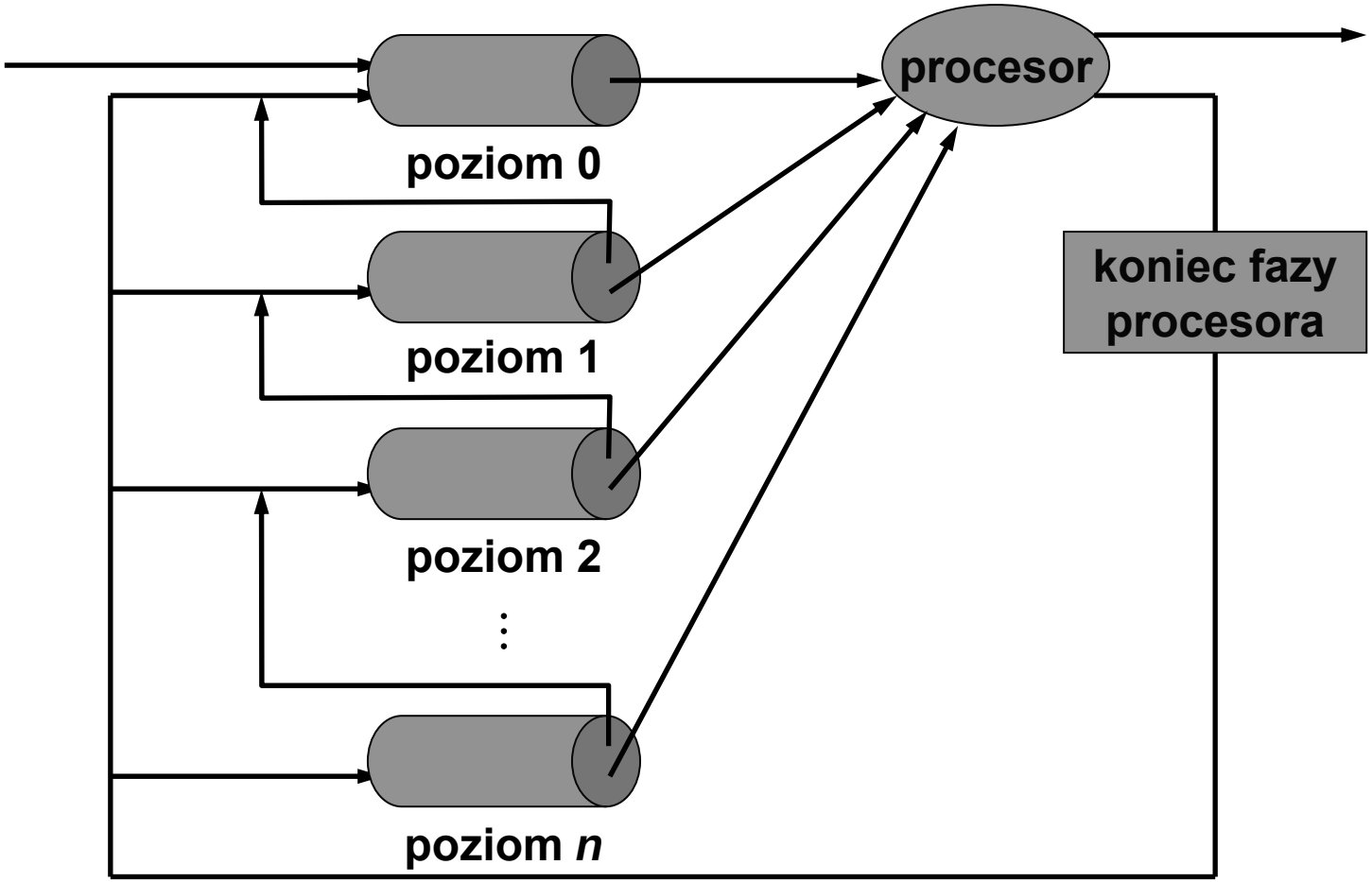
Szeregowanie procesów, ograniczonych wejściem-wyjściem

- Procesy ograniczone wejściem-wyjściem potrzebują niewiele czasu procesora, większość czasu w systemie spędzając na oczekiwaniu na urządzenia zewnętrzne.
- Opóźnianie przydziału procesora dla tego typu procesów powoduje zmniejszenie wykorzystania urządzeń zewnętrznych, a przydział — ze względu na nie długą fazę procesora — nie powoduje istotnego zwiększenia czasu oczekiwania innych procesów.
- Właściwym algorytmem byłby SJF lub SRT.
- Bezwzględna preferencja dla procesów oczekujących na gotowość urządzeń może spowodować głodzenie procesów ograniczonych procesorem.

Wirtualne planowanie rotacyjne (VRR)



Wielopoziomowe kolejki ze sprzężeniem zwrotnym



Pamięć

Przestrzeń adresowa

1. Alokacja
2. Fragmentacja
3. Odwzorowanie adresu
   a) liniowość
   b) stronicowanie
   c) segmentacja
4. Wiązanie adresu

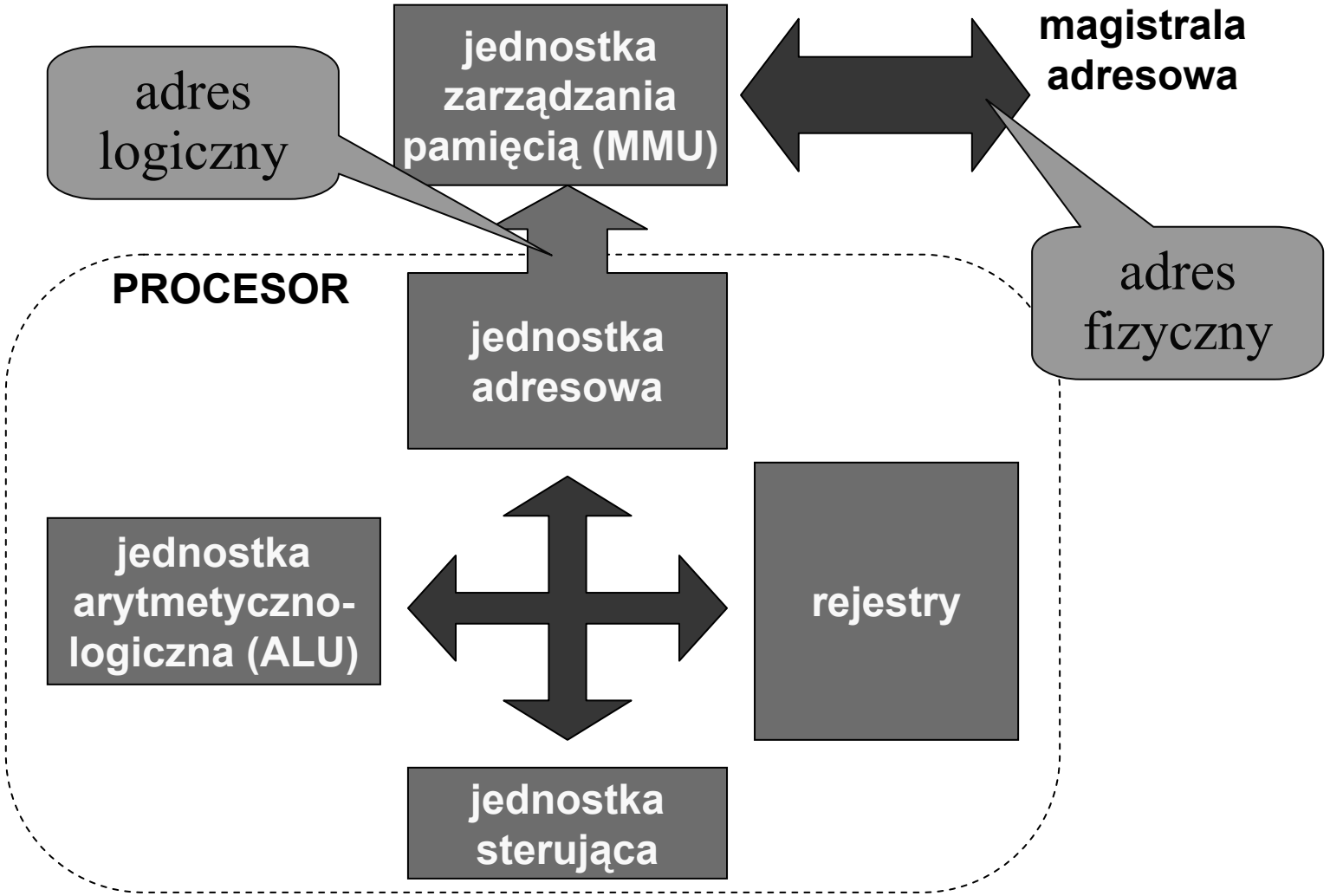
Hierarchia pamięci

1. Pamięć główna
2. Pamięć pomocnicza
   a) wymiana
   b) wirtualizacja
3. Pamięć podręczna
   a) organizacja
   b) koherencja

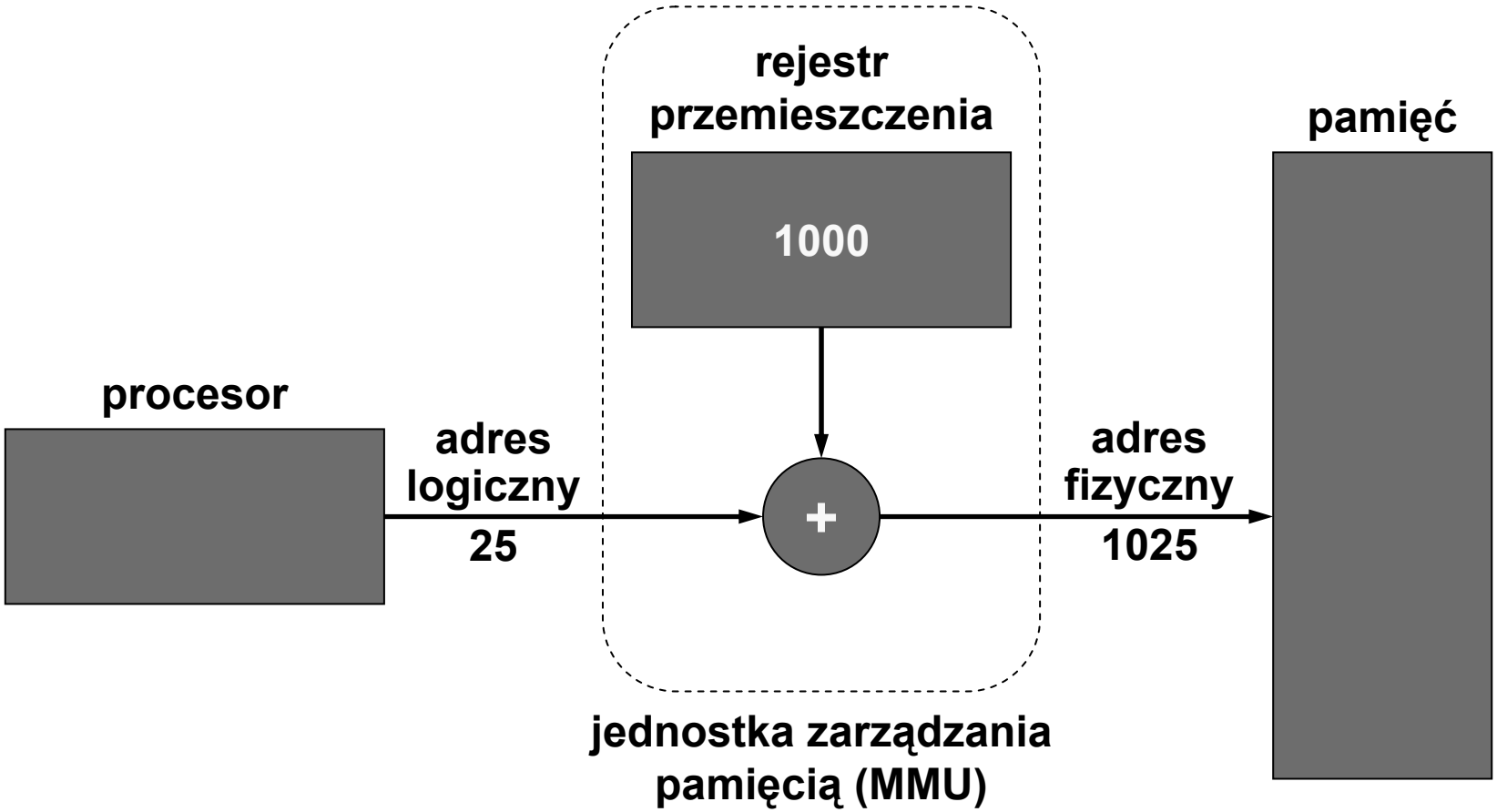
Podstawowe pojęcia

- Pamięć — zasób systemu komputerowego, służący do przechowywania danych
  - pamięć operacyjna (wewnętrzna) — adresowana przez procesor, umożliwia bezpośredni dostęp do danych
  - pamięć masowa (zewnętrzna) — dostępna poprzez odpowiedni układ wejścia-wyjścia
- Przestrzeń adresowa — zbiór wszystkich dopuszczalnych adresów pamięci operacyjnej
  - przestrzeń fizyczna — zbiór adresów przekazywanych do układów pamięci
  - przestrzeń logiczna — zbiór adresów generowanych przez procesor

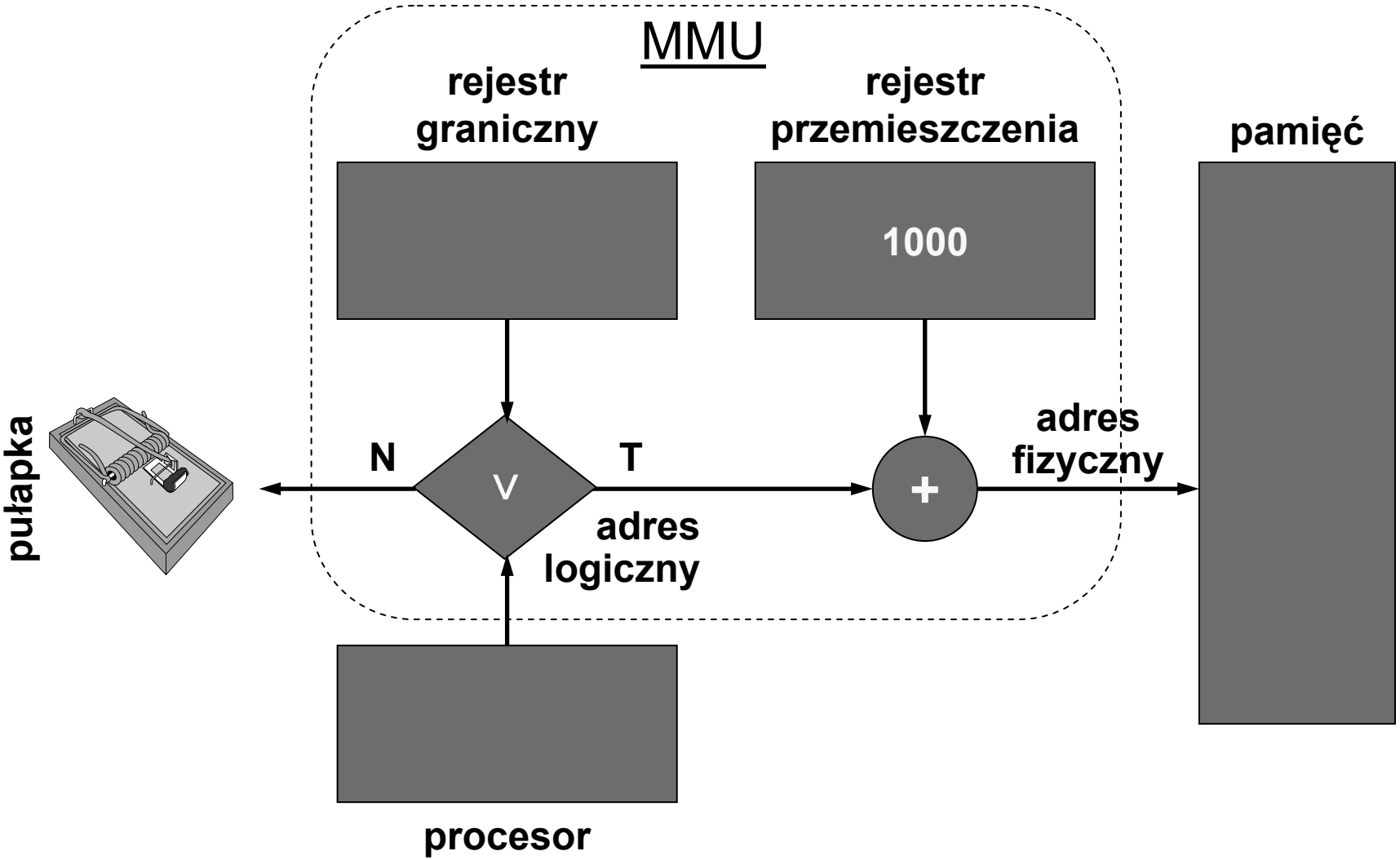
Adres logiczny i fizyczny



Przykład odwzorowania adresu logicznego na fizyczny



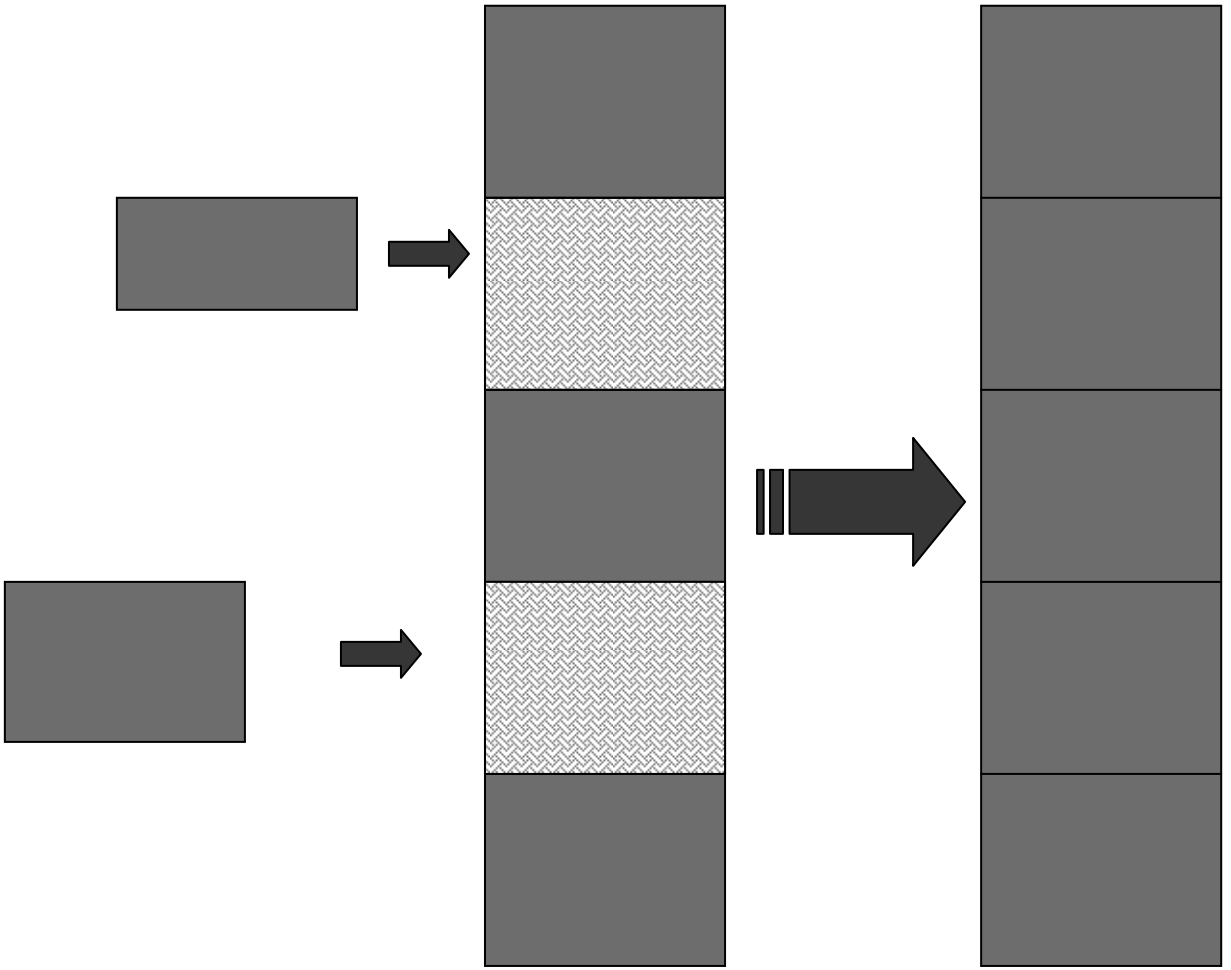
Przykład ochrony obszaru pamięci



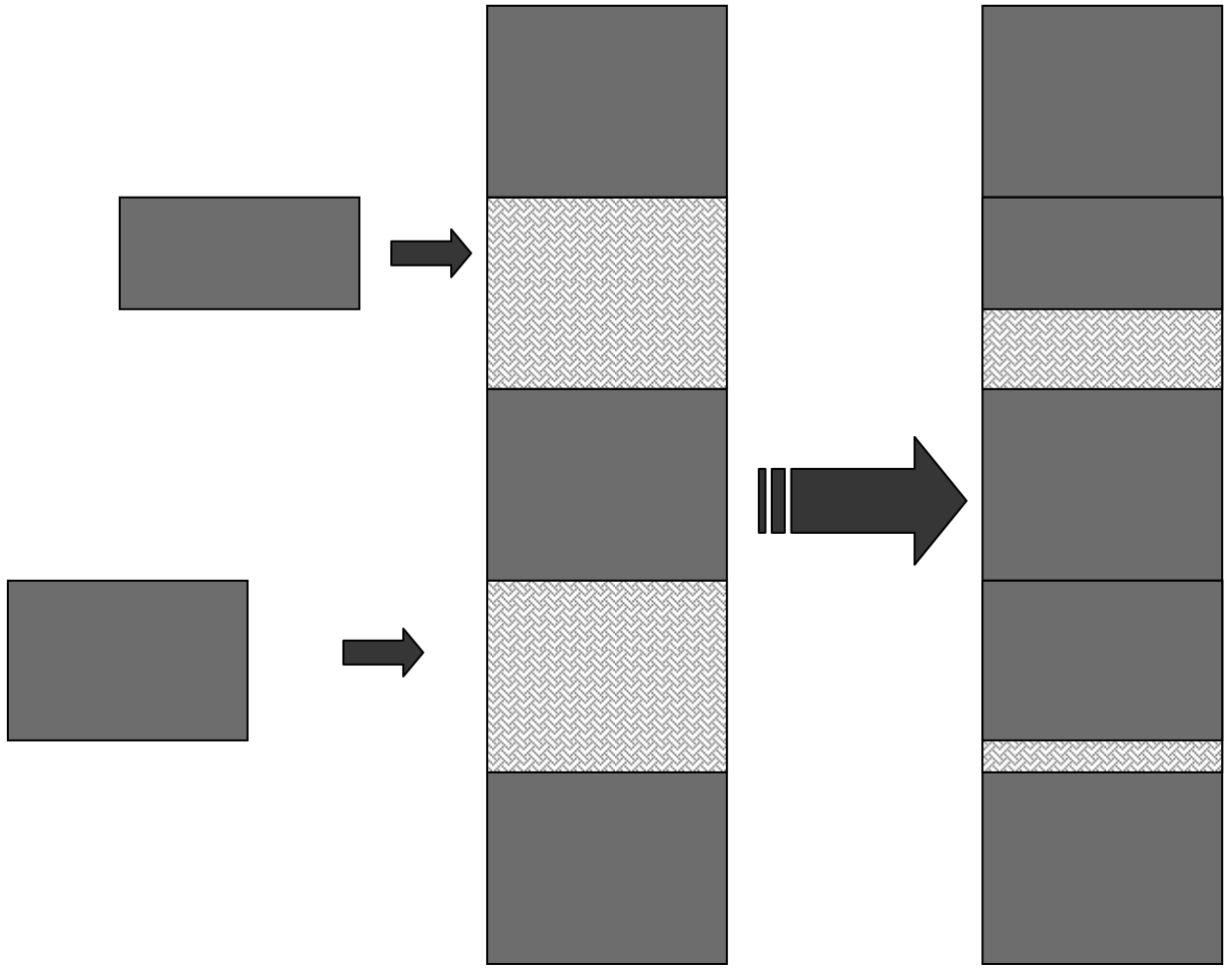
Przydział pamięci (partycjonowanie)

- Systemy ze stałym podziałem pamięci — pamięć dla programów użytkowych podzielona jest na stałe obszary (strefy, partycje). Procesom przydzielane są całe obszary wynikające z podziału.
- Systemy ze zmiennym podziałem pamięci — rozmiary przydzielanych obszarów są zmienne i dąży się do tego by odpowiadały dokładnie rozmiarom pamięci wymaganym przez zadania. Żądającemu procesowi przydziela się tyle pamięci, ile potrzebuje — reszta pozostaje wolna.

Systemy z podziałem stałym



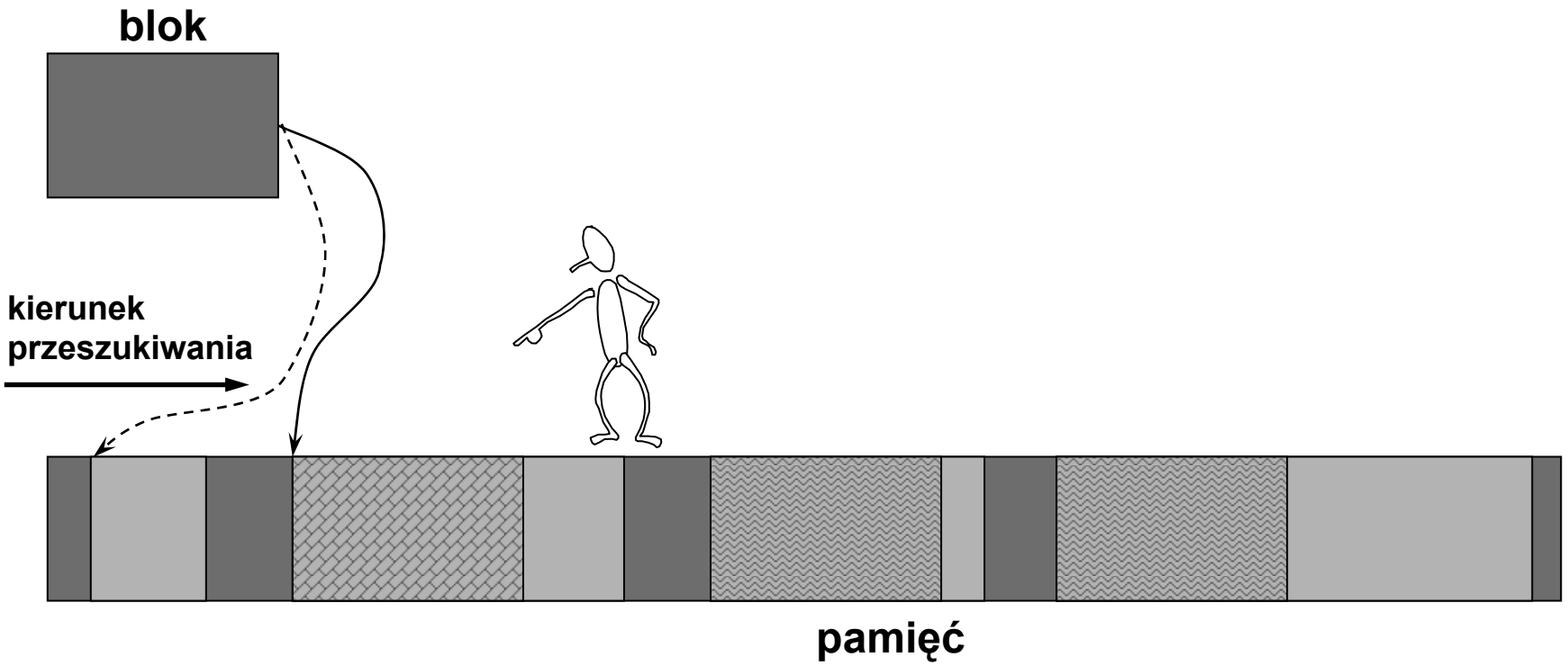
Systemy ze podziałem zmiennym



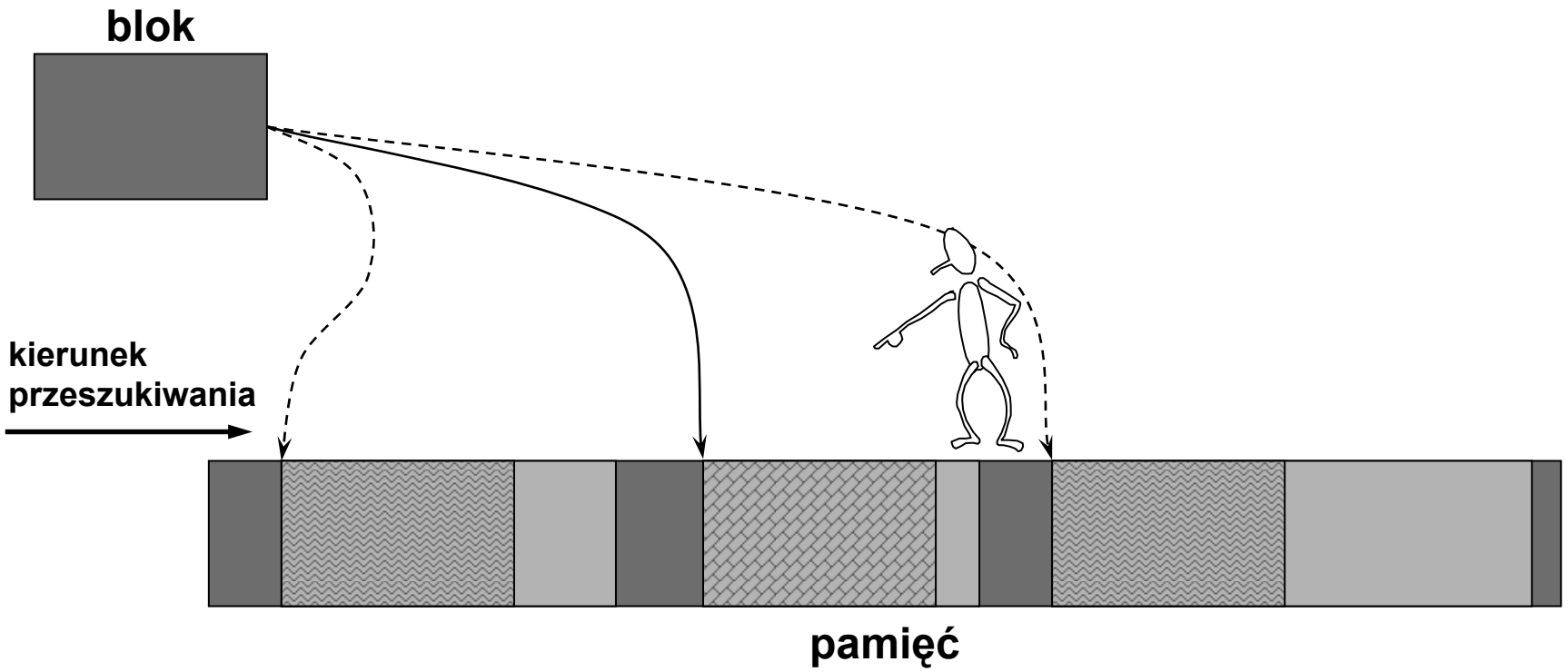
Systemy z podziałem zmiennym — problem wyboru bloku

- Pierwsze dopasowanie — przydziela się **pierwszy wolny** obszar (tzw. dziurę) o wystarczającej wielkości. Poszukiwanie kończy się po znalezieniu takiego obszaru.
- Najlepsze dopasowanie — przydziela się **najmniejszy dostatecznie duży** wolny obszar pamięci. Konieczne jest przeszukanie wszystkich dziur.
- Najgorsze dopasowanie — przydziela się **największy** wolny obszar pamięci. Konieczne jest przeszukanie wszystkich dziur.

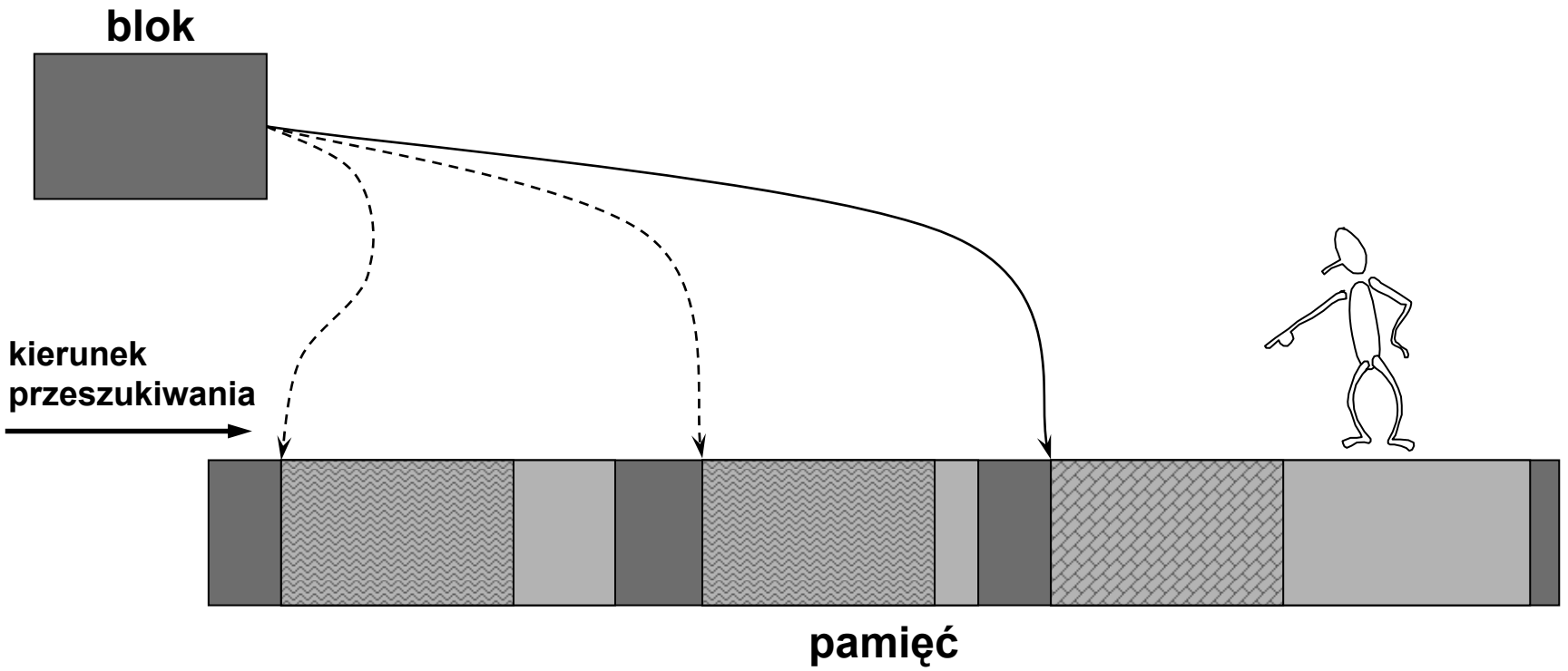
Przydział ciągłego obszaru pamięci — pierwsze dopasowanie



Przydział ciągłego obszaru pamięci — najlepsze dopasowanie



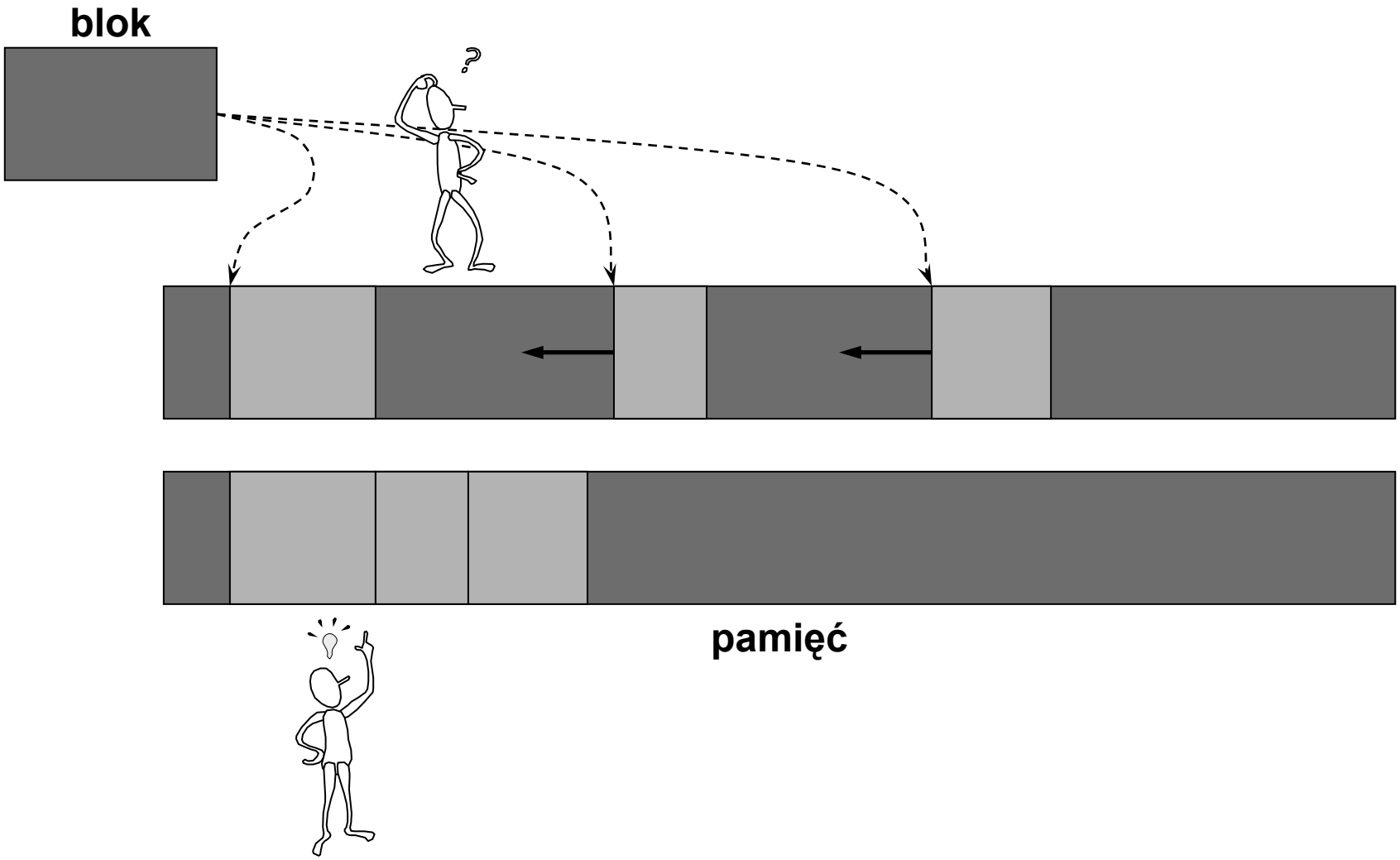
Przydział ciągłego obszaru pamięci — najgorsze dopasowanie



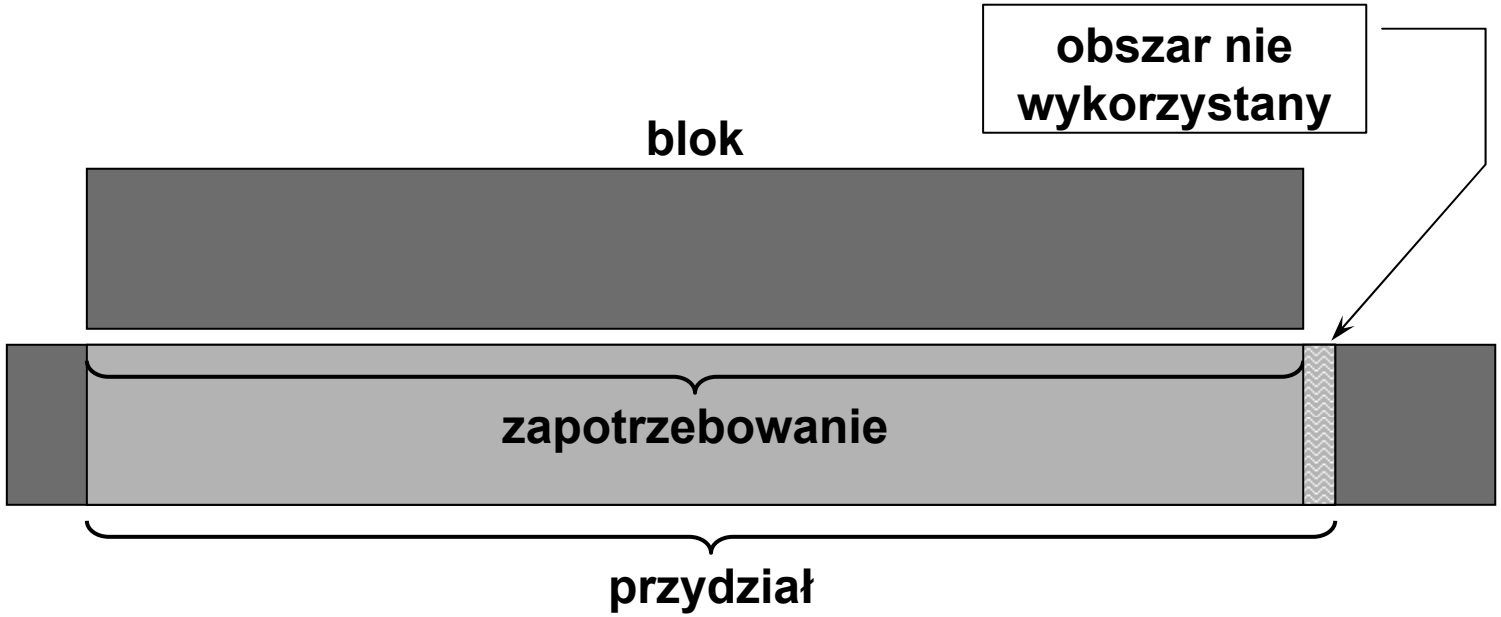
Fragmentacja

- Fragmentacja zewnętrzna — podział obszaru pamięci na rozłączne fragmenty, które nie stanowią ciągłości w przestrzeni adresowej (może to dotyczyć zarówno obszaru wolnego, jak i zajętego)
- Fragmentacja wewnętrzna — pozostawienie niewykorzystywanego fragmentu przestrzeni adresowej wewnątrz przydzielonego obszaru (formalnie fragment jest zajęty, w rzeczywistości nie jest wykorzystany)

Fragmentacja zewnętrzna



Fragmentacja wewnętrzna



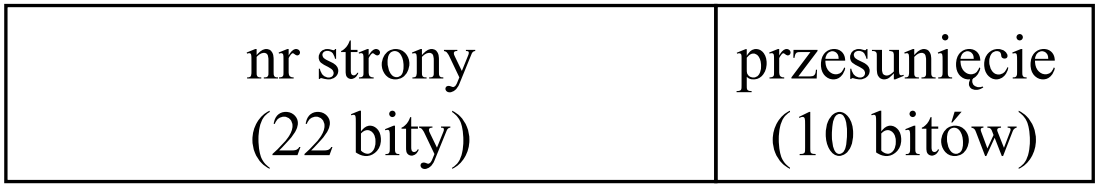
Przydział dokładnie tylu bajtów, ile wynosi zapotrzebowanie, powoduje, że koszt utrzymania bardzo małego obszaru wolnego jest niewspółmiernie duży, np. dane o obszarze zajmują więcej bajtów, niż rozmiar tego obszaru. Dlatego wolny obszar przydzielany jest w całości, ale nie jest w pełni wykorzystany.

Odwzorowanie logicznej przestrzeni adresowej w fizyczną

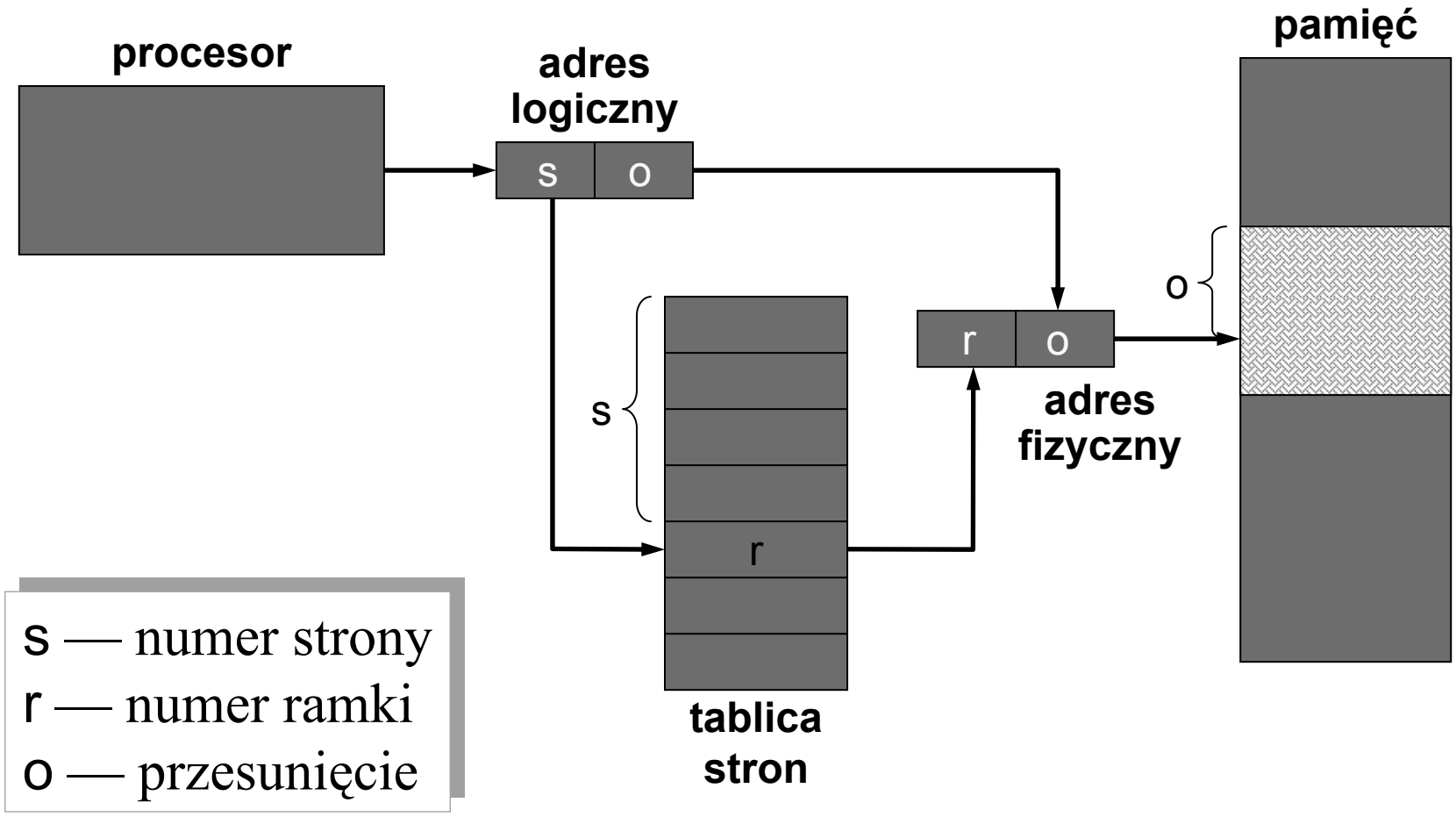
- **Odwzorowanie bezpośrednie** — pamięć jest tablicą słów (bajtów). Adres jest ciągiem bitów, interpretowanych jako indeks słowa (bajtu) w pamięci.
- **Stronicowanie** — pamięć jest podzielona na strony, czyli obszary o ustalonym rozmiarze. Bardziej znacząca część bitów adresu interpretowana jest jako numer strony, a pozostała, mniej znacząca część bitów jest przesunięciem na stronie.
- **Segmentacja** — pamięć jest podzielona na segmenty, czyli obszary zdefiniowane wskazanie początku w pamięci i rozmiar. Adres logiczny obejmuje identyfikator (numer) segmentu oraz przesunięcie wewnątrz segmentu.

Stronicowanie (ang. paging)

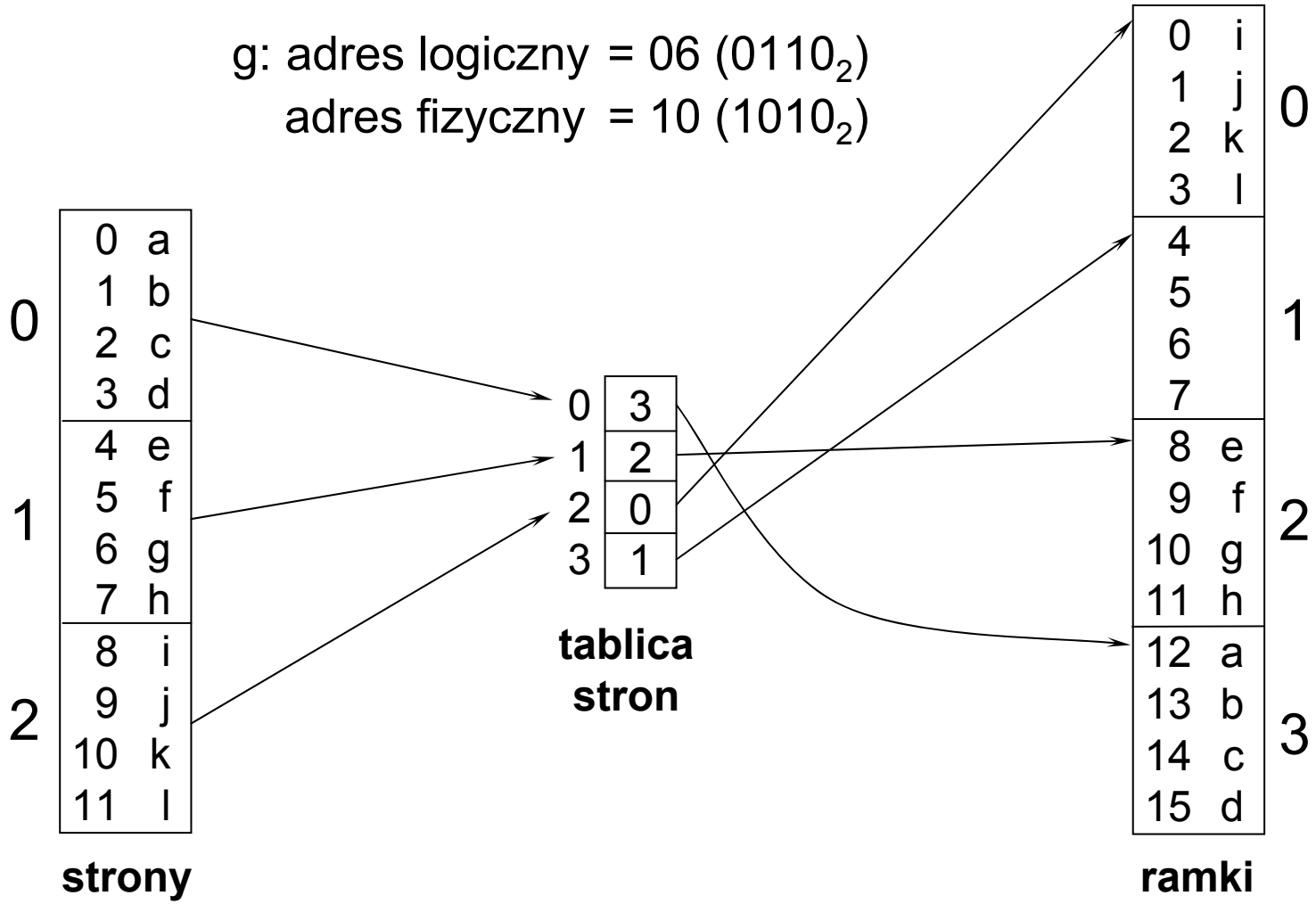
- Podział fizycznej przestrzeni adresowej na bloki stałej wielkości, zwane **ramkami** (ang. frames).
- Podział logicznej przestrzeni adresowej na bloki stałej (takiej samej jak ramki) wielkości, zwane **stronami** (ang. pages).
- Odwzorowanie pomiędzy numerem strony a numerem ramki za pomocą **tablicy stron** (ang. page table).
- Adres zawiera numer strony i przesunięcia na stronie (ang. offset).



Schemat transformacji adresu w systemie pamięci stronicowanej



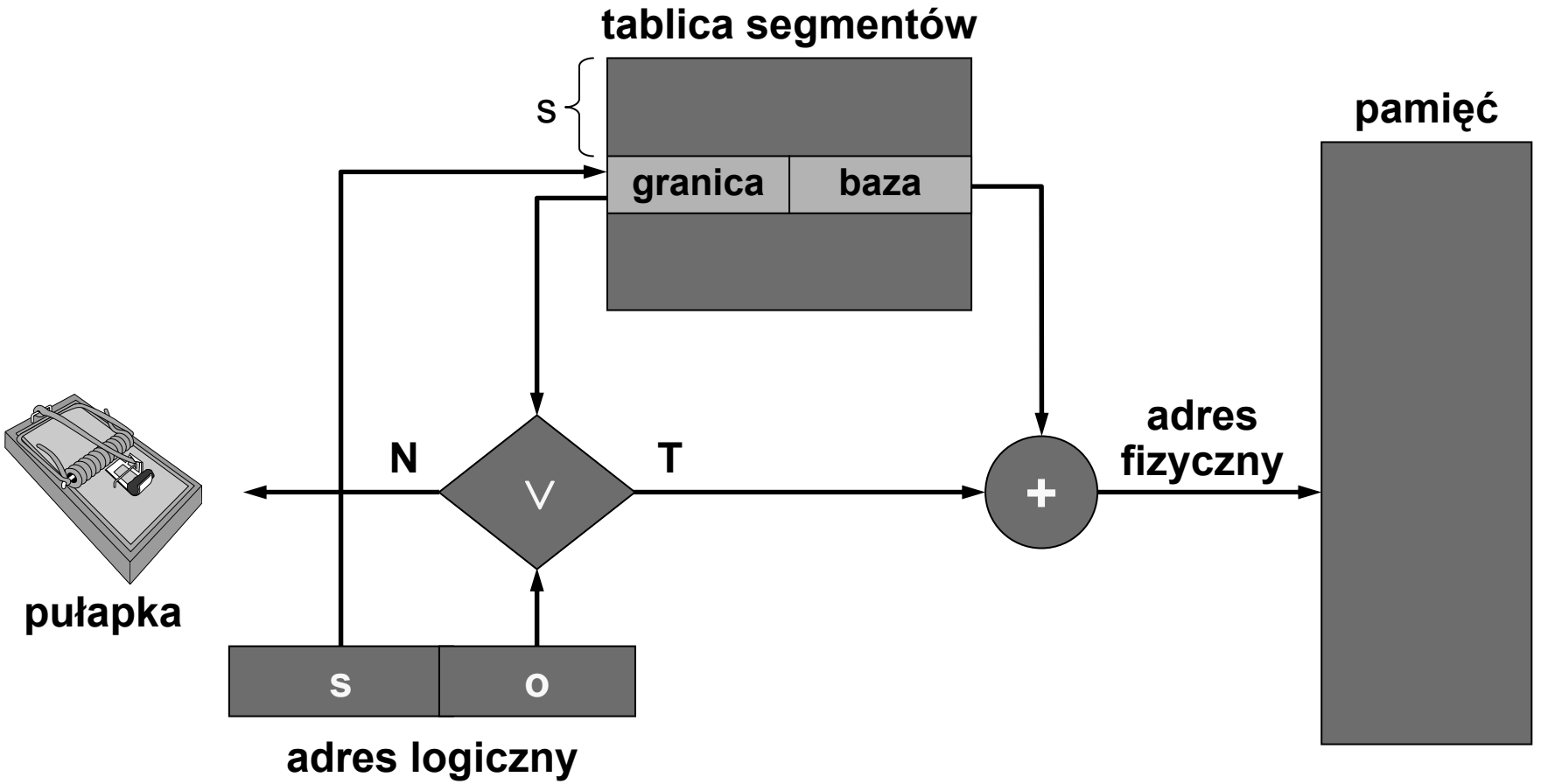
Przykład odwzorowania stron w ramki



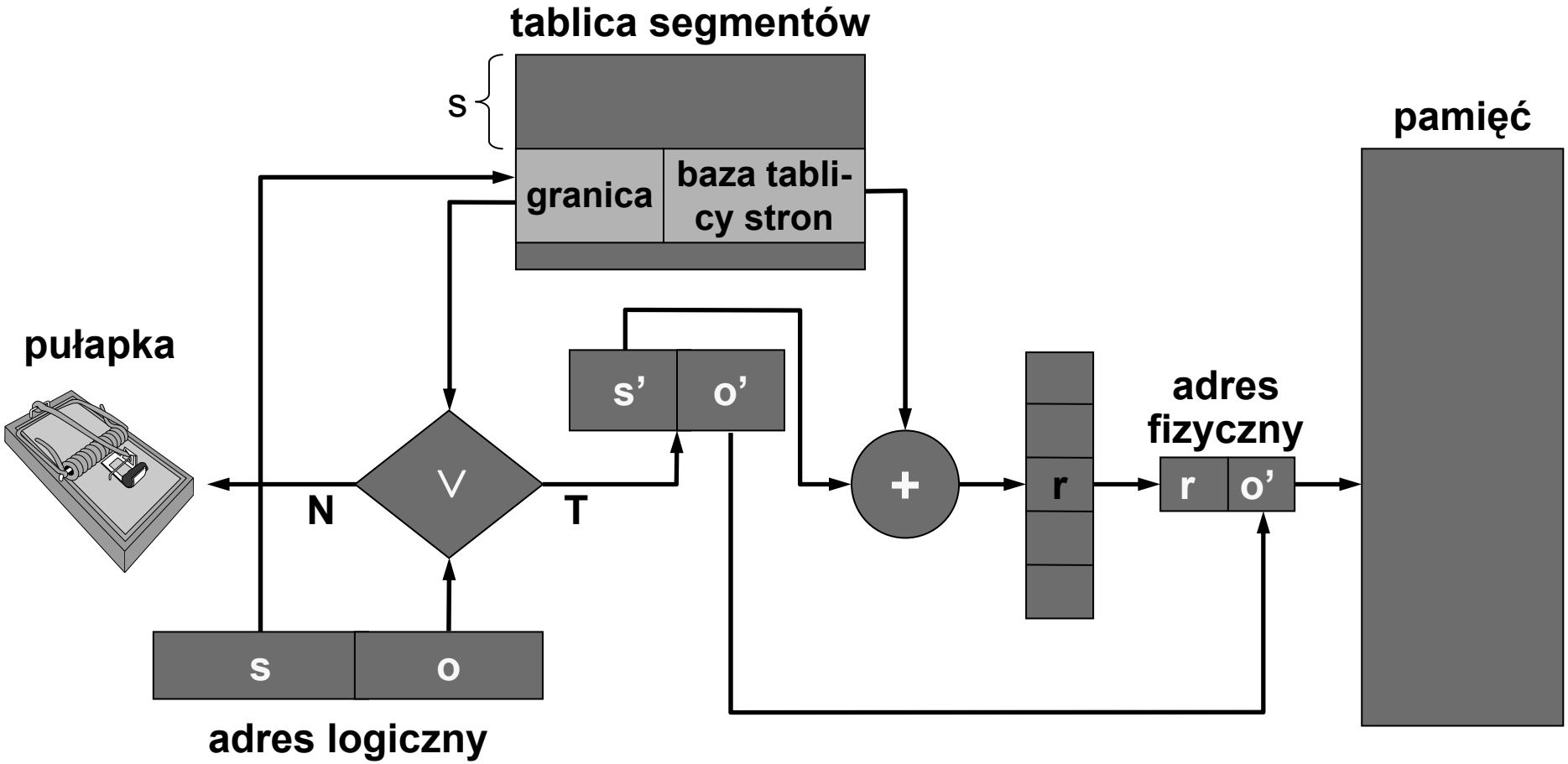
Segmentacja (ang. segmentation)

- Przestrzeń adresów logicznych jest postrzegana jako zbiór segmentów.
- Podstawowe atrybuty segmentu:
  - nazwa (identyfikator, numer), która jest indeksem w tablicy segmentów
  - adres bazowy — wskazanie na początek segmentu
  - rozmiar — długość segmentu w ustalonych jednostkach (np. w bajtach, paragrafach).
- Adres logiczny składa się z numeru segmentu i przesunięcia wewnątrz segmentu.
- Odwzorowanie adresów logicznych w fizyczne zapewnia tablica segmentów.

Schemat adresowania z segmentacją



Schemat adresowania z segmentacją i stronicowaniem



Pamięć wirtualna

1. Stronicowanie na żądanie (ang. demand paging)
2. Błąd (braku) strony (ang. page-fault)
3. Zastępowanie stron (ang. page replacement)
4. Problemy ze stronicowaniem (wybór ofiary, szamotanie stron, problem wznawiania wykonania rozkazów)
5. Algorytmy zastępowanie (ang. page-replacement alg.)
   a) Algorytm wymiany na żądanie
   b) Algorytm ze sprowadzaniem na żądanie
   c) Algorytm ze wstępnym stronicowaniem

Uzasadnienie stosowania pamięci wirtualnej

- Pewne fragmenty kodu programu wykonywane są bardzo rzadko lub nigdy nie są wykonywane, np.:
  - procedury obsługi bardzo mało prawdopodobnych wyjątków lub błędów,
  - moduły programu, które w typowych warunkach eksploatacji okazują się zbędne.
- Pewne fragmenty kodu programu nie muszą być dostępne jednocześnie.
- Rozmiary pewnych struktur danych (np. tablic) ustalane są z nadmiarem w stosunku do typowych potrzeb.

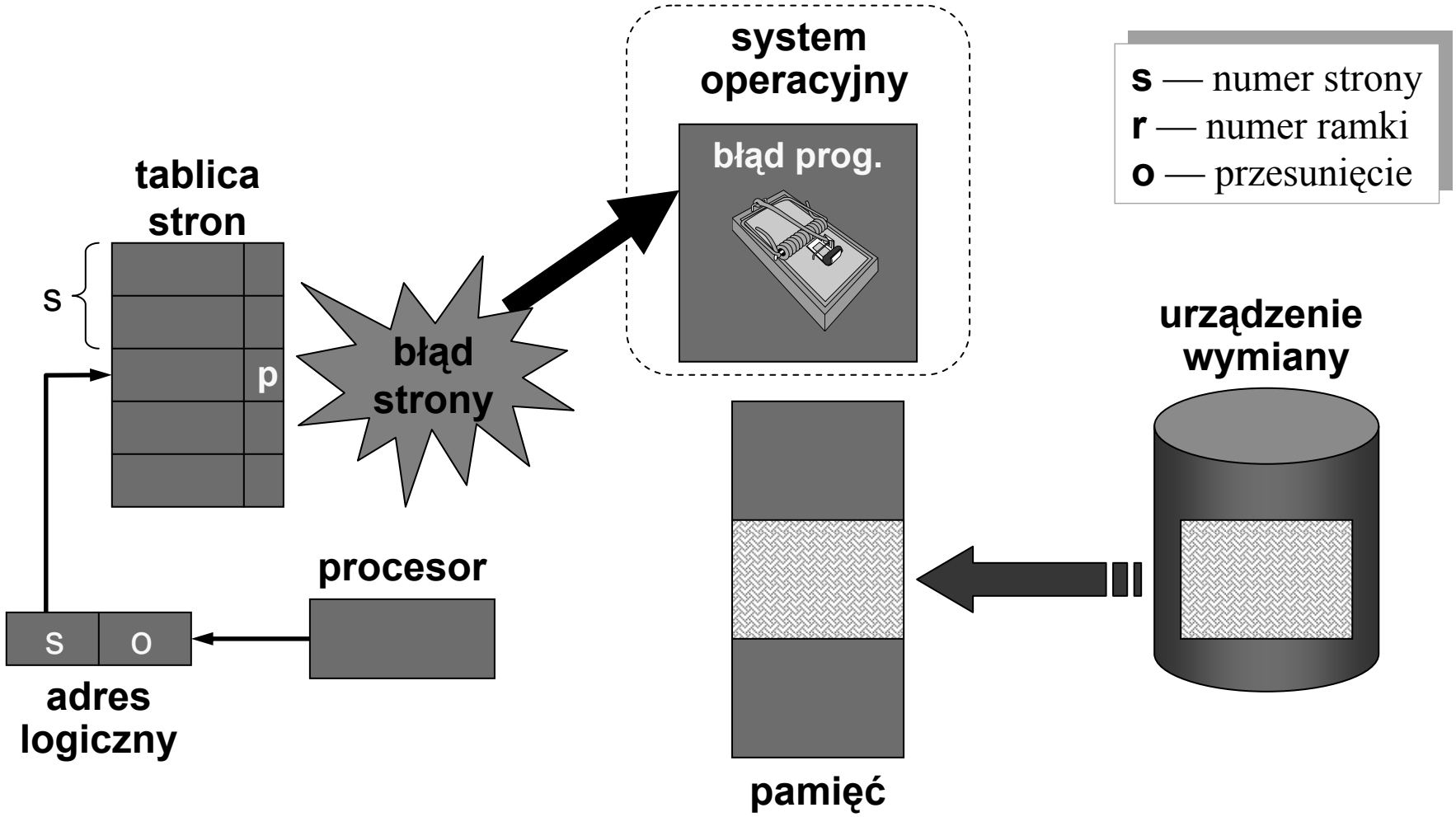
Zalety stosowania pamięci wirtualnej

- Możliwość zwiększenia rozmiarów programów ponad rozmiar pamięci fizycznej
- Możliwość zwiększenia liczby procesów (poprawa wykorzystania procesora i przepustowości systemu bez zwiększania czasu odpowiedzi i czasu cyklu przetwarzania) dzięki zredukowaniu zapotrzebowania na pamięć fizyczną przez poszczególne procesy
- Zmniejszenie liczby operacji wejścia-wyjścia i tym samym zredukowanie czasu ładowanie programów przy ich uruchamianiu, gdyż nie jest konieczne ładowanie programów w całości

Stronicowanie na żądanie

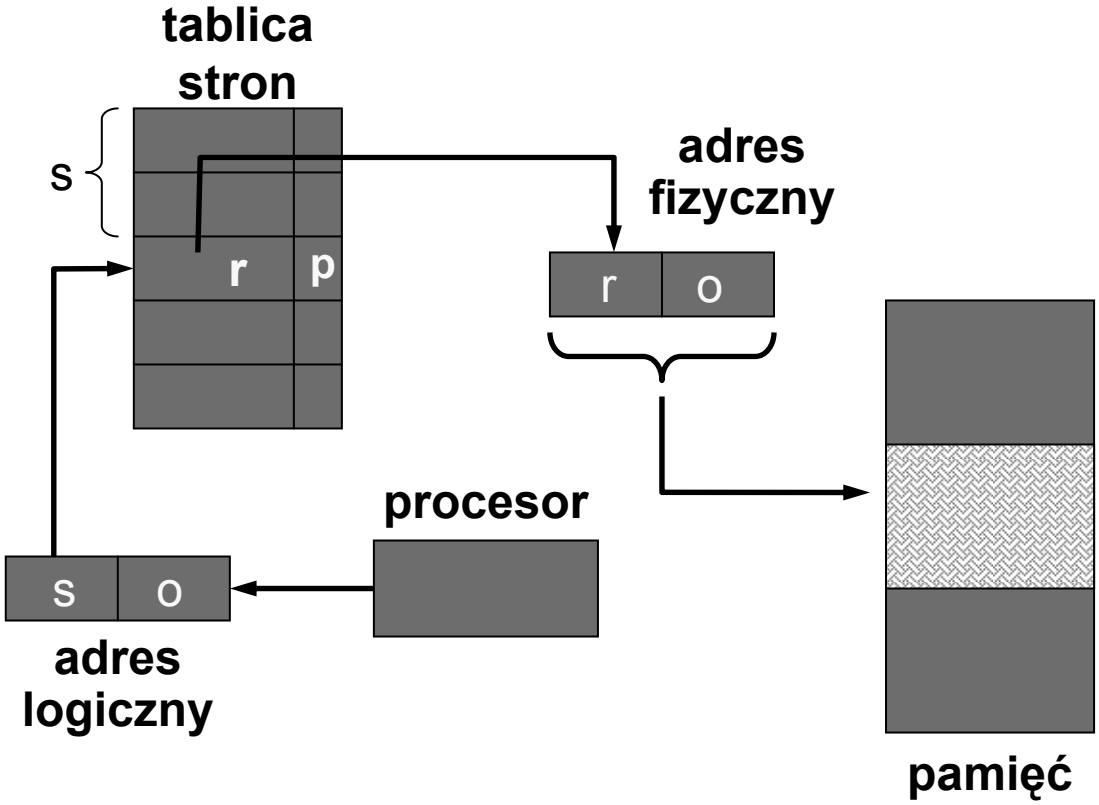
- Stronicowanie na żądanie (ang. demand paging) a stronicowanie w przypadku wymiany (ang. swapping)
  - wymiana — pomiędzy pamięcią fizyczną a pomocniczą przesyłane są całe procesy,
  - stronicowanie na żądanie — proces traktowany jest jako ciąg stron, które są sprowadzane do pamięci tylko wówczas, gdy jest to konieczne (leniwa wymiana, ang. lazy swapping).
- Wymaganie sprzętowe związane ze stronicowaniem na żądanie
  - tablica stron z bitem poprawności (ang. valid-invalid bit) dla każdej pozycji,
  - urządzenie wymiany (ang. swap device) — pamięć pomocnicza.

Obsługa błędu strony



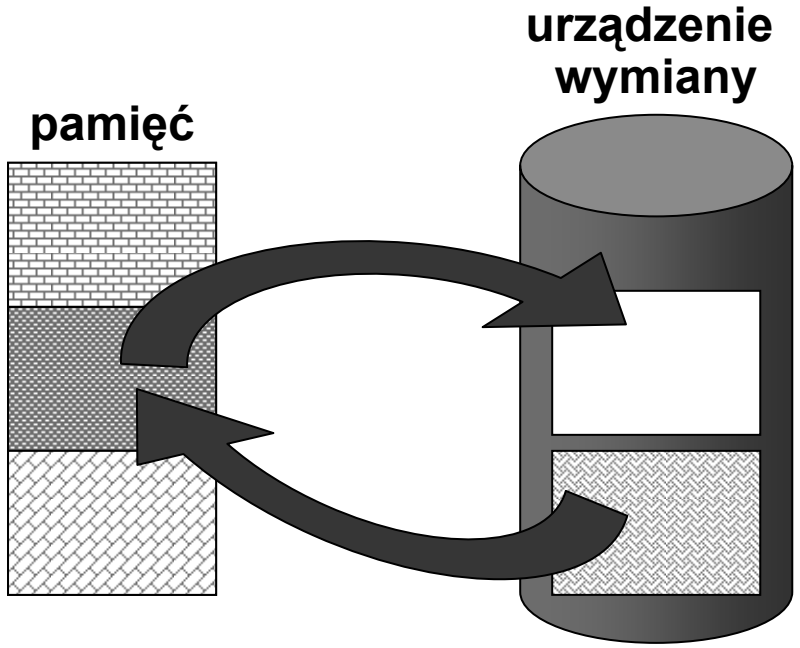
Powtórne wykonania rozkazu

s — numer strony
r — numer ramki
o — przesunięcie



Zastępowanie stron (ang. page replacement)

- Problem zastępowania (wymiany) stron pojawia się w, gdy w pamięci fizycznej brakuje wolnych ramek i konieczne jest zwolnienie jakieś ramki poprzez usunięcie z niej strony.
- Jeśli strona była modyfikowana w pamięci, konieczne jest zapisanie jej na dysku $\Rightarrow$ konieczność wprowadzenie bitu modyfikacji (ang. modify bit), zwanego też bitem zabrudzenia (ang. dirty bit).



Problemy zastępowania stron

- Problem wyboru ofiary — niewłaściwy wybór ramki-ofiary może prowadzić do zjawiska migotania, w którym często dochodzi do wystąpienia odniesienia do właśnie usuniętej strony, co konsekwencji wymaga ponownego sprowadzenie jej do pamięci. Dalszą konsekwencją takiego zjawiska może być drastyczny spadek efektywności działania systemu komputerowego.
- Problem wznawiania rozkazów — w przypadku wielokrotnego odniesienia do pamięci w jednym cyklu rozkazowym należy zapewnić, że wszystkie adresowane strony są jednocześnie dostępne w ramkach w pamięci fizycznej.

Problem wyboru ofiary

- Zakładając, że przyszły ciąg odniesień do pamięci nie jest znany, na podstawie historii odniesień należy wybrać taką ramkę, do której prawdopodobieństwo odniesienia w przyszłości jest małe.
- Podstawowa własność programów, na podstawie której można szacować takie prawdopodobieństwo nazywana jest *lokalnością*.

Własność lokalności

- Lokalność jest określona jako tendencja procesów do generowania w stosunkowo długich przedziałach czasu odniesień do niewielkiego podzbioru stron wirtualnych zwanego *zbiorem stron aktywnych*.
- Rodzaje własność lokalności:
  - lokalność czasowa — tendencja procesu do generowania z dużym prawdopodobieństwem w przedziale czasu $(t, t + \tau)$ odniesień do stron adresowanych w przedziale czasu $(t - \tau, t)$;
  - lokalność przestrzenna — tendencja procesu do generowania z dużym prawdopodobieństwem odniesień do stron o zbliżonych numerach (stron sąsiednich).

Problem efektywności systemu z pamięcią wirtualną

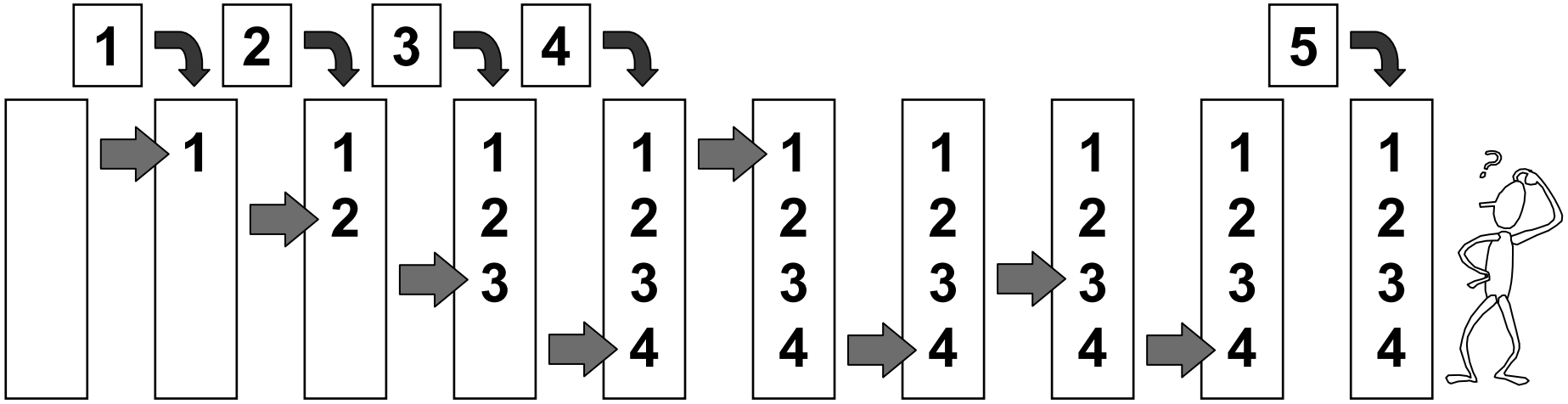
- Efektywność działania systemu pamięci wirtualnej zależy od precyzji identyfikacji zbioru stron aktywnych i możliwości utrzymania ich w pamięci fizycznej.
- Wobec braku a priori pełnego ciągu odniesień do stron wirtualnych identyfikacja takiego zbioru może wynikać z różnych przesłanek, czego skutkiem jest duża różnorodność algorytmów wymiany.

Przykłady alg. zastępowania stron (ang. page-replacement algorithms)

- Algorytm FIFO (ang. First In First Out) — zastępowana jest strona najstarsza (najwcześniej wprowadzona do pamięci)
- Algorytm OPT (MIN) — zastępowana jest strona, która najdłużej nie będzie używana
- Algorytm LRU (ang. Least Recently Used) — zastępowana jest najdawniej używana strona (najdłużej nie używana)
- LFU (ang. Least Frequently Used) — zastępowana jest najrzadziej używana strona
- MFU (ang. Most Frequently Used) — zastępowana jest najczęściej używana strona

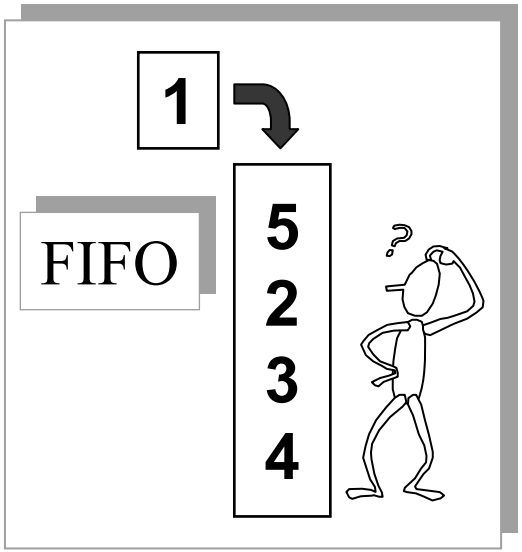
Przykład działania algorytmu wymiany stron (1)

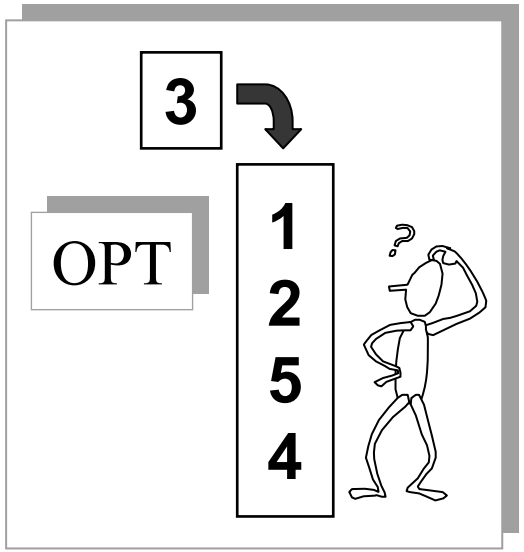
- W systemie pamięci wirtualnej są 4 ramki.
- Wszystkie ramki są początkowo puste
- W systemie pojawiają się następujące odniesień (odwołań) do stron: 1, 2, 3, 4, 1, 4, 3, 4, **5**, 2, 1, 4, 3, 4

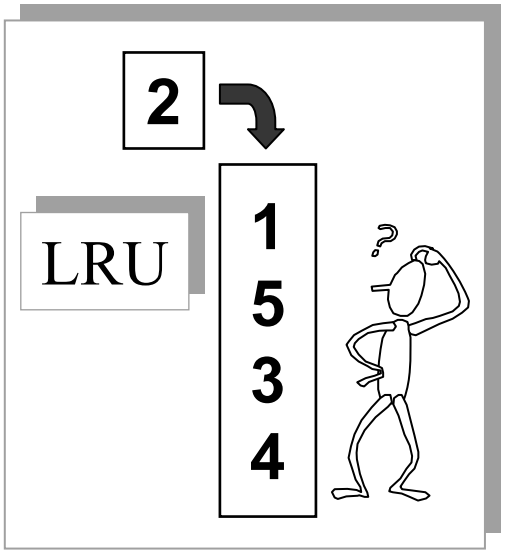


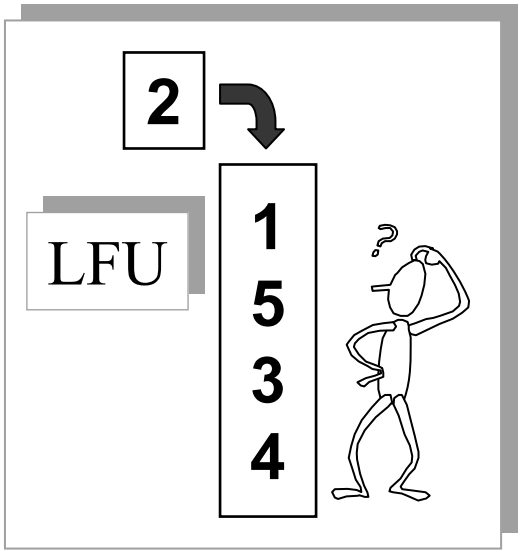
Przykład działania algorytmu wymiany stron (2)

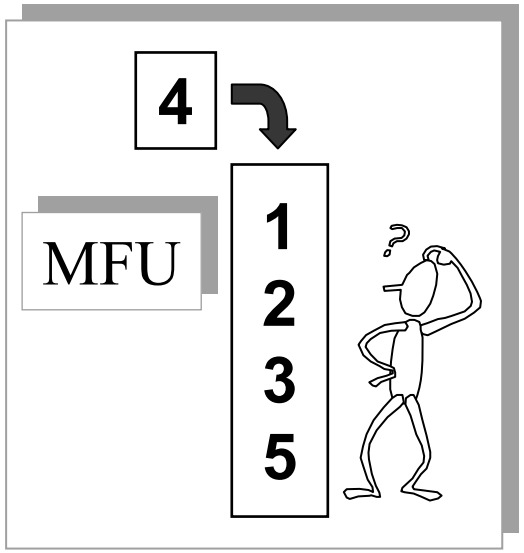
Dalszy ciąg odniesień:
2, 1, 4, 3, 4











Zagadnienia implementacyjne

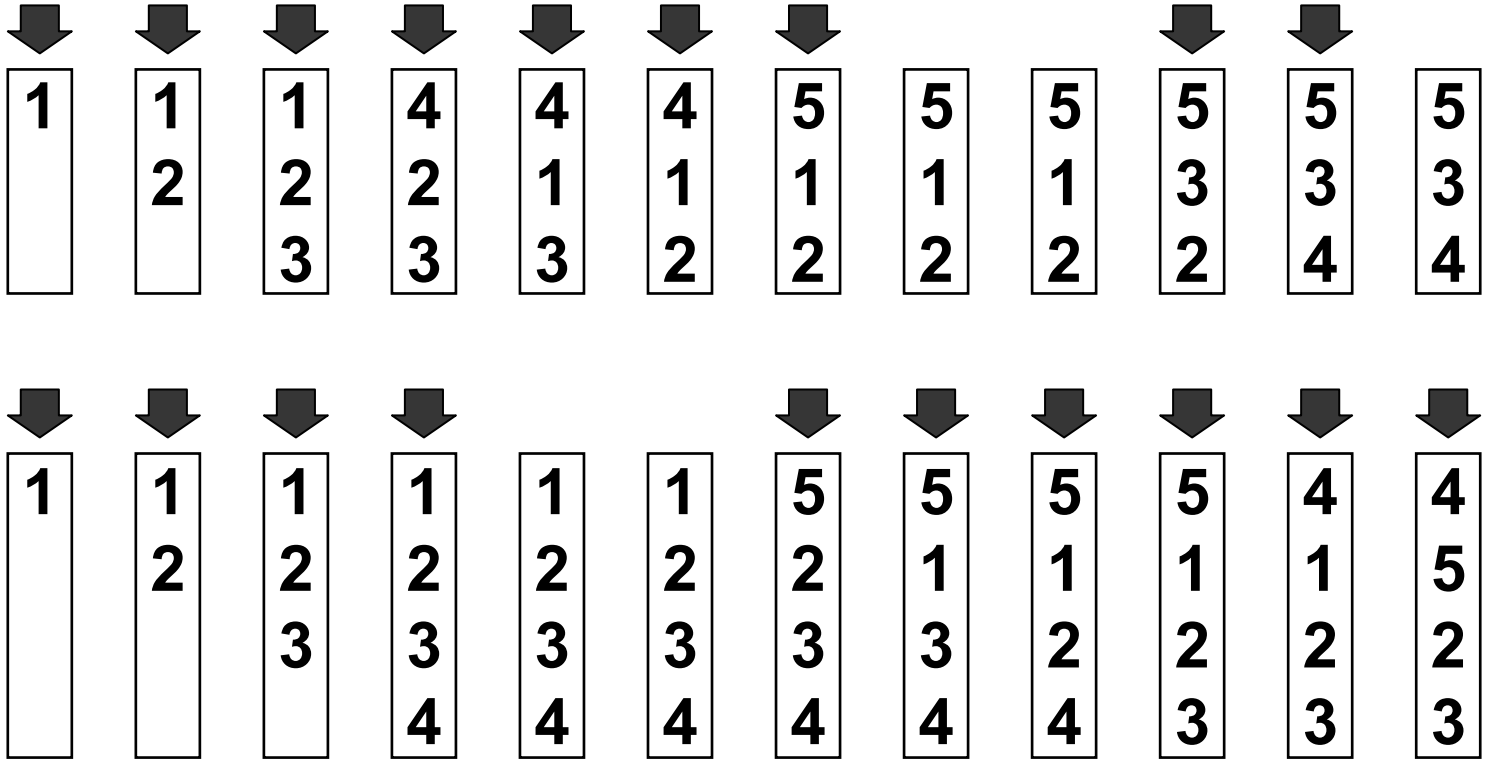
- Implementacja algorytmu FIFO
- Implementacja algorytmu LRU
- Algorytmy przybliżające metodę LRU
  - algorytm dodatkowych bitów odwołań
  - algorytm drugiej szansy (FINUFO)
  - ulepszony algorytm drugiej szansy
- Implementacja algorytmów licznikowych

Implementacja algorytmu FIFO

- Utrzymywanie listy numerów stron w kolejności ich sprowadzania do pamięci
- Umieszczanie numeru sprowadzanej strony na końcu listy
- Usuwanie z pamięci (i z listy) strony, której numer znajduje się na początku listy
- Możliwość wystąpienia anomalii Belady'ego (przy zwiększeniu liczby ramek następuje przypadkowe zwiększenie liczby błędów strony)

Przykład anomalii Belady'ego

- Ciąg odniesień: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
- Przy zastosowaniu algorytmu FIFO dla 3 ramek mamy 9 błędów strony, a dla 4 ramek mamy 10 błędów strony



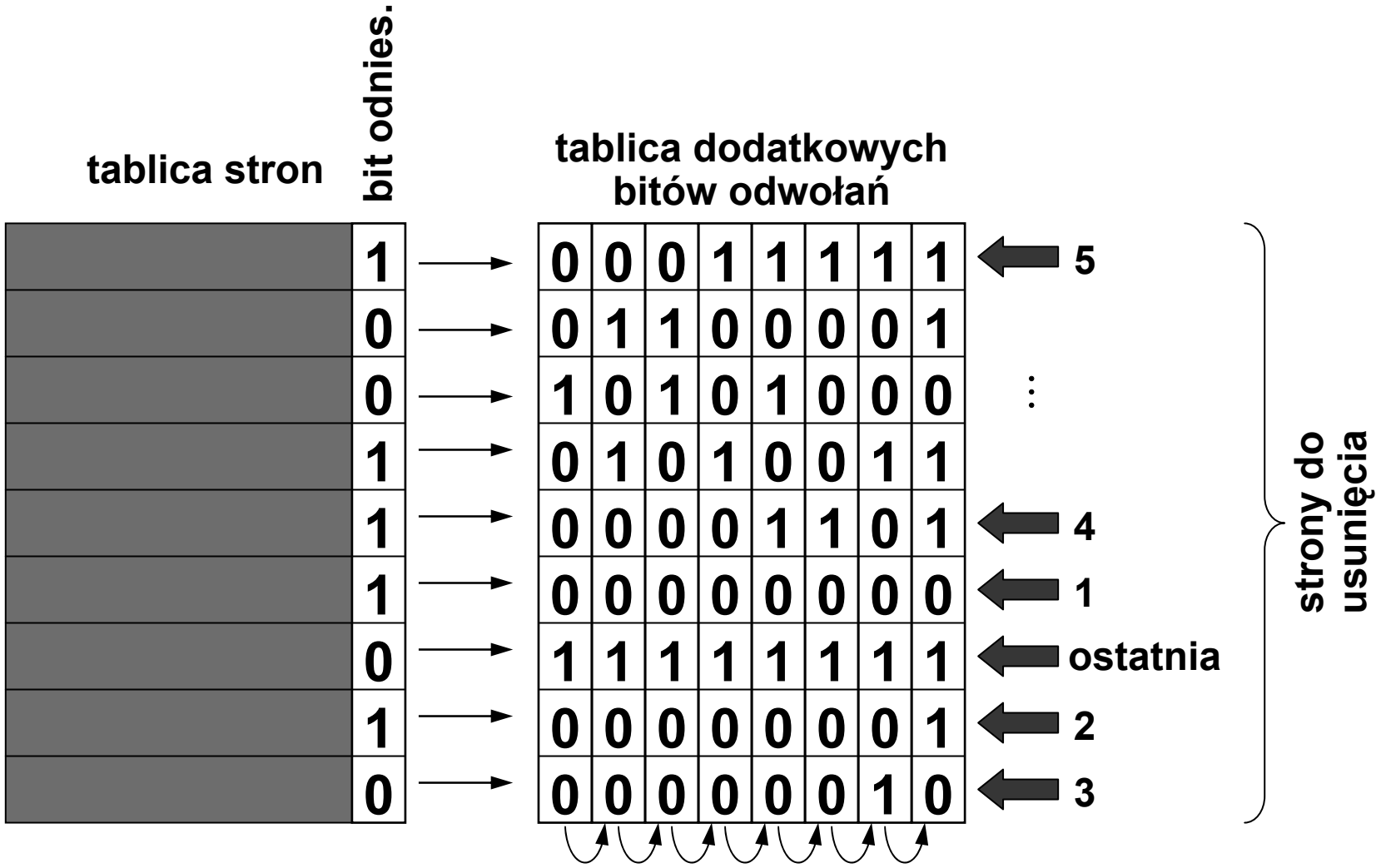
Implementacja algorytmu LRU

- Licznik — przed każdym odwołaniem do pamięci zwiększana jest wartość pewnego licznika i wpisywana do odpowiedniej pozycji opisującej stronę w tablicy stron (lub w innej specjalnej strukturze systemu operacyjnego). Z pamięci usuwana jest wówczas strona z najmniejszą wartością tego licznika, co wymaga przejrzenia całej tablicy stron.
- Stos — numery stron, do których następuje odwołanie, odkładane są na szczycie stosu. Przed odłożeniem na szczycie numer strony musi być wydobyty ze środka stosu, czyli z miejsca, gdzie był ostatnio odłożony. W tej implementacji pamięci usuwana jest strona z dna stosu.

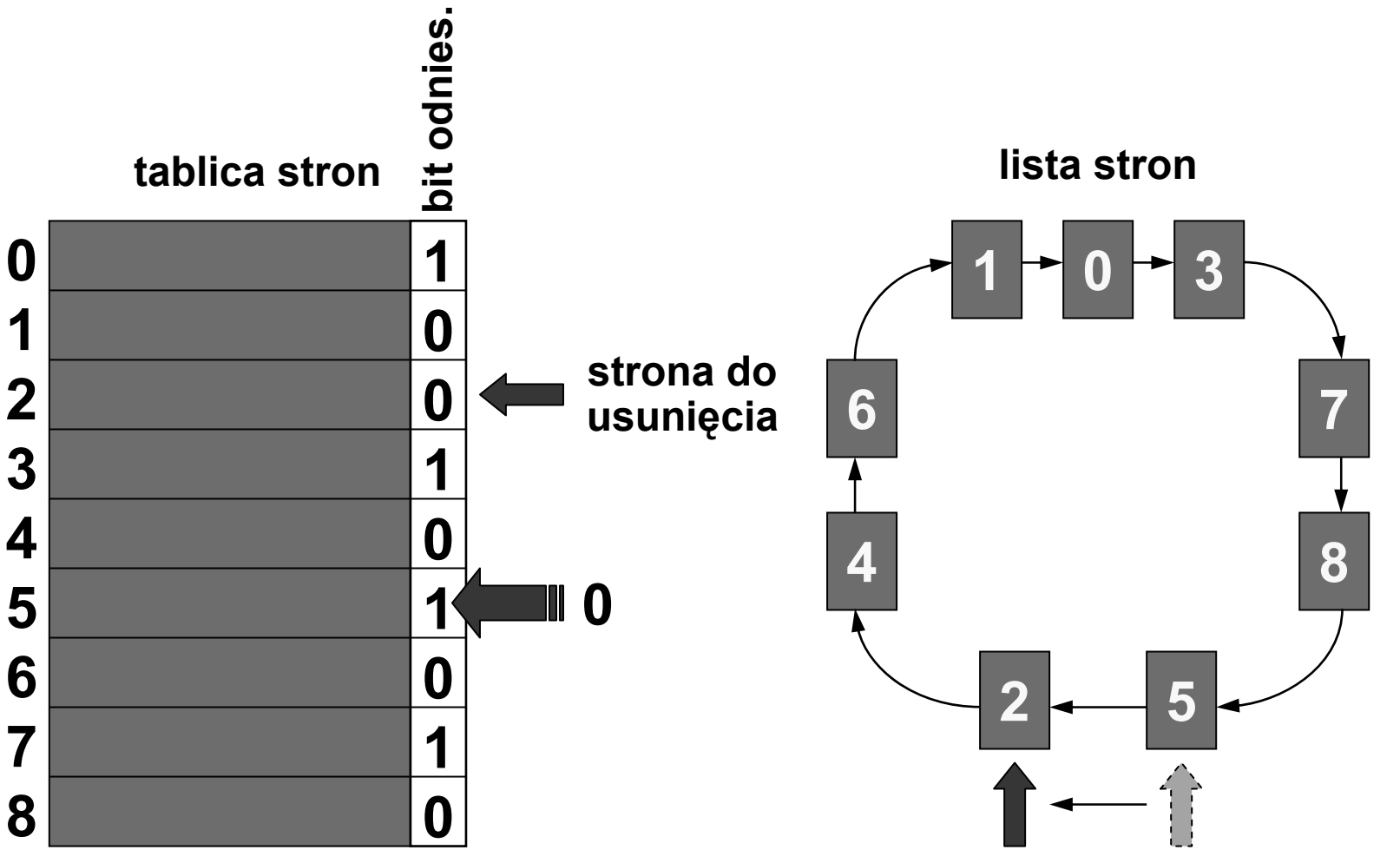
Algorytmy przybliżające metodę LRU

- Niezbędne wspomaganie sprzętowe:
  - bit odniesienia (ang. reference bit) — ustawiany dla danej strony przez sprzęt zawsze, gdy następuje **zapis lub odczyt** na tej stronie,
  - bit modyfikacji (ang. modify bit) — ustawiany dla danej strony przez sprzęt zawsze, gdy następuje **zapis** na tej stronie.
- Algorytmy korzystające ze wspomagania sprzętowego:
  - algorytm dodatkowych bitów odwołań — wykorzystuje bit odniesienia,
  - algorytm drugiej szansy — wykorzystuje bit odniesienia,
  - ulepszony algorytm drugiej szansy — wykorzystuje bit odniesienia i bit modyfikacji.

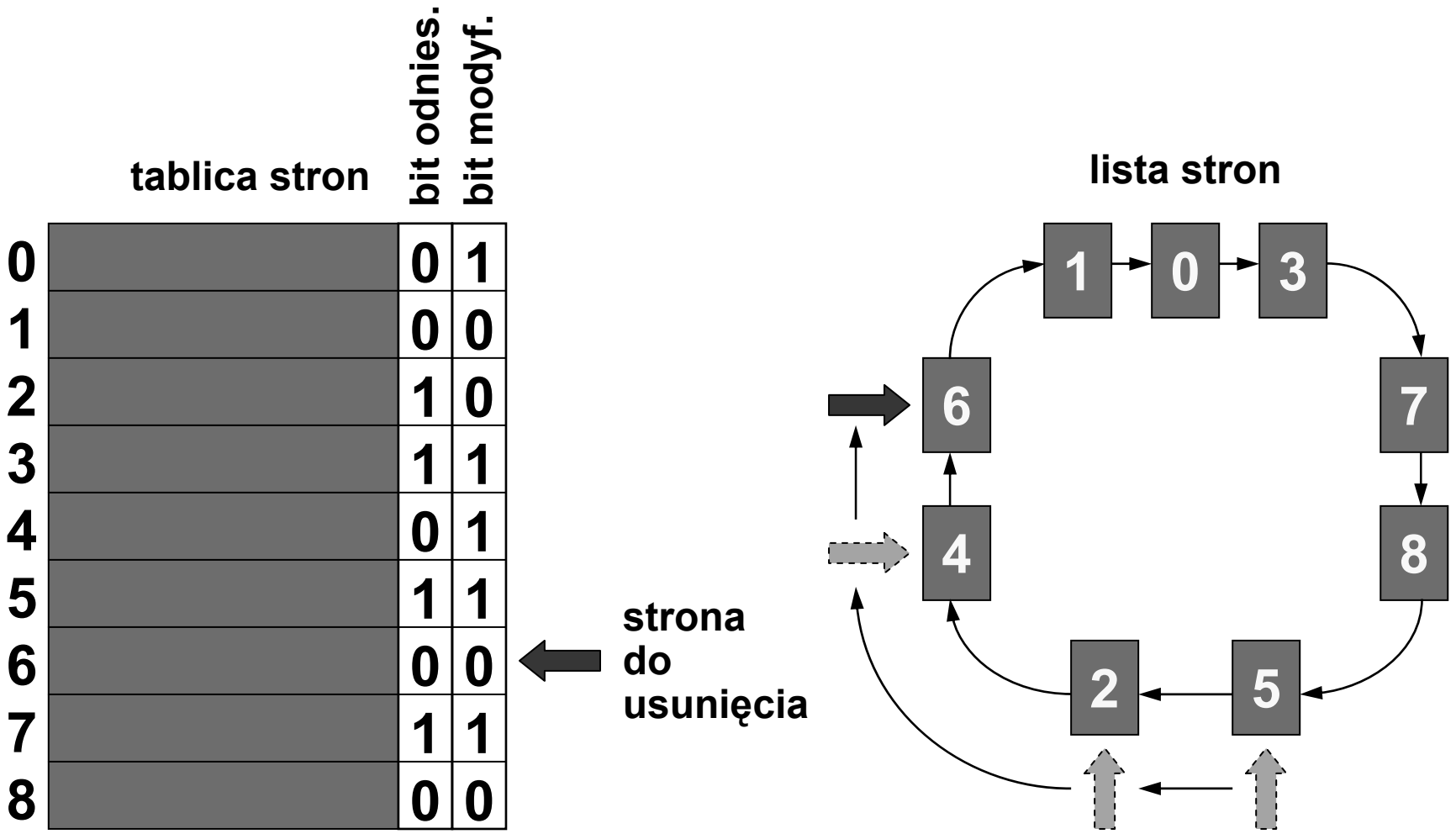
Algorytm dodatkowych bitów odwołań



Algorytm drugiej szansy



Ulepszony algorytm drugiej szansy



Implementacja algorytmów LFU i MFU

- Utrzymywanie liczników odniesień do każdej ze stron i ich inkrementacja przy każdym odwołaniu do odpowiedniej strony
- Usuwanie strony z najmniejszą (LFU) lub największą (MFU) wartością licznika
- Zerowanie licznika po usunięciu strony z pamięci

Urządzenia wej.-wyj.

1. Rodzaje urządzeń wejścia-wyjścia
2. Struktura mechanizmu wejścia-wyjścia
   a) sterownik urządzenia
   b) moduł sterujący
   c) podsystem wejścia-wyjścia
3. Miejsce urządzeń wejścia-wyjścia w architekturze systemu komputerowego
   a) odwzorowanie w przestrzeni adresowej wejścia-wyjścia
   b) odwzorowanie w przestrzeni adresowej pamięci
4. Interakcja z urządzeniem wejścia-wyjścia
   a) odpytywanie (ang. polling)
   b) przerwania (ang. interrupts)
   c) bezpośredni dostęp do pamięci (ang. direct memory access)
5. Buforowanie i spooling

Rodzaje urządzeń wejścia-wyjścia (ang. IO devices)

- Urządzenia składowania danych (dyski, dyskietki, taśmy, CD ROM, DVD itp.)
- Urządzenia transmisji danych (karty sieciowe, modemy)
- Urządzenia do komunikacji z człowiekiem (monitory, projektory, klawiatury, myszy, drukarki, skanery, kamery itp.)
- Urządzenia specjalizowane
  - układy sterowania (np. elektrownią, samolotem, systemem obrony antyrakietowej itd.)
  - kasy i drukarki fiskalne itp.
  - urządzenia medyczne

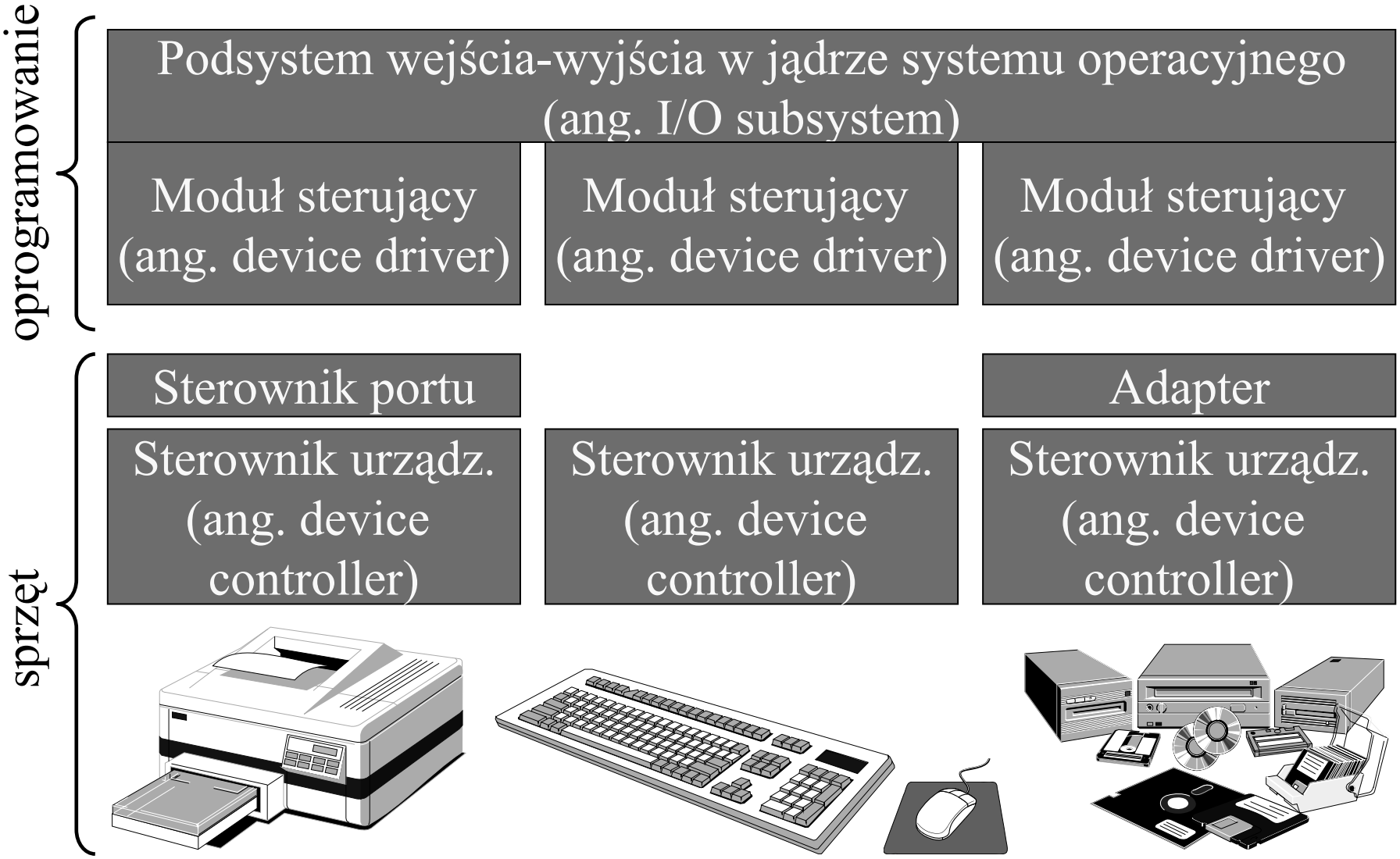
Właściwości urządzeń wejścia-wyjścia (1)

- Tryb transmisji danych:
  - znakowy — urządzenie przysyła dane bajt po bajcie (ang. chracter-stream device)
  - blokowy — dane przysyłane są w blokach (np. po 512 bajtów)
- Sposób dostępu do danych:
  - sekwencyjny — urządzenie przesyła dane w określonym porządku, zależnym od samego urządzenie (np. karta sieciowa)
  - swobodny — możliwy jest wpływ na wybór danych do przesyłania przez urządzenie (np. dysk)
- Tryb pracy urządzenia:
  - synchroniczny — dane zostaną przekazane w znanym z góry (przewidywalnym) czasie (np. dysk)
  - asynchroniczny — dane mogą zostać przesłane w dowolnym, trudnym do przewidzenia, momencie (np. klawiatura, karta sieciowa)

Właściwości urządzeń wejścia-wyjścia (2)

- Tryb współdzielenia:
  - wyłączny — używanie urządzenia przez jeden proces uniemożliwia jego używanie przez inny proces (np. drukarka)
  - współdzielony — dopuszczalne jest współbieżne używanie urządzenia przez wiele procesów (np. dysk)
- Szybkość działania (transmisji)
  - od bardzo wolnych (np. drukarka)
  - do bardzo szybkich (np. dysk)
- Kierunek przekazywania danych
  - urządzenia wejścia i wyjścia — możliwość zarówno zapisu jak i odczytu (np. dysk, karta sieciowa)
  - urządzenia wejścia — tylko możliwość odczytu z urządzenia (np. klawiatura)
  - urządzenia wyjścia — tylko możliwość zapisu (np. drukarka)

Struktura mechanizmu wejścia-wyjścia

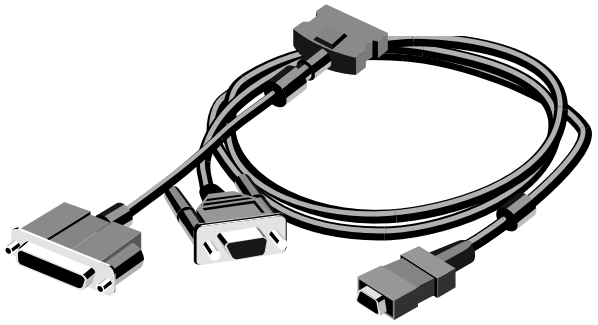


Podsystem wejścia-wyjścia (ang. I/O subsystem)

- Dostarczanie aplikacji interfejsu funkcji (API) umożliwiających wykonywanie operacji wejścia-wyjścia w sposób jednolity, niezależny od urządzenia lub grupy, do której należy urządzenie.
- Typowy interfejs obejmuje funkcje:
  - **`read`** — odczyt z urządzenia (pobieranie danych)
  - **`write`** — zapis do urządzenia (wysyłanie danych)

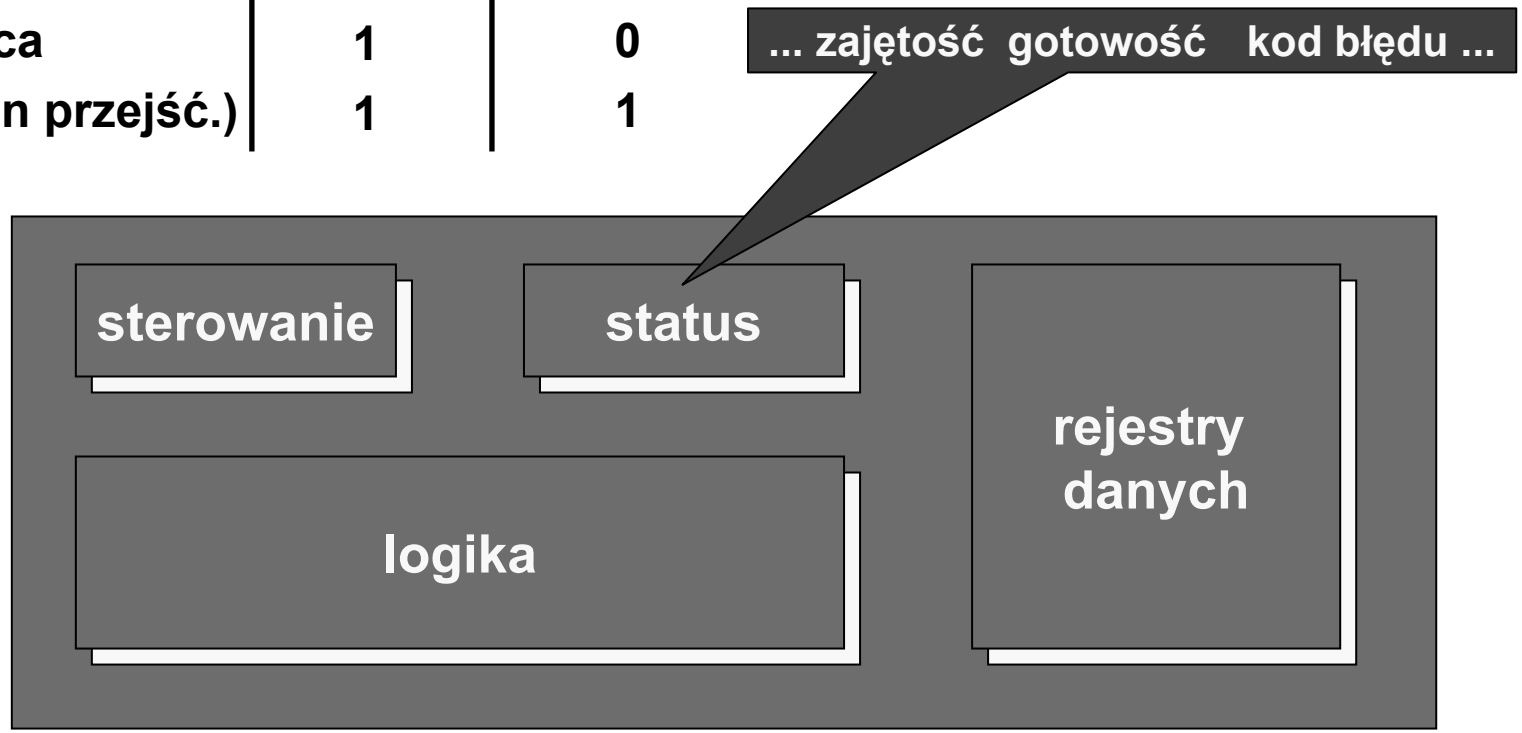
Moduł sterujący (ang. device driver)

- Dostarczanie jednolitego (wspólnego dla pewnej grupy urządzeń) interfejsu dostępu, czyli pewnego standardowego zbioru operacji na danym urządzeniu.
- Ukrywanie sprzętowych szczegółów realizacji danego urządzenia przed podsystemem wejścia-wyjścia.
- Moduły dostarczane są dla typowych systemów operacyjnych (Windows 95/98, Windows NT, Solaris, Linux) przez wytwórców urządzeń zewnętrznych.



Sterownik portu (adapter)

	zajętość	gotowość
bezczynność	0	0
zakończenie	0	1
praca	1	0
(stan przejść.)	1	1



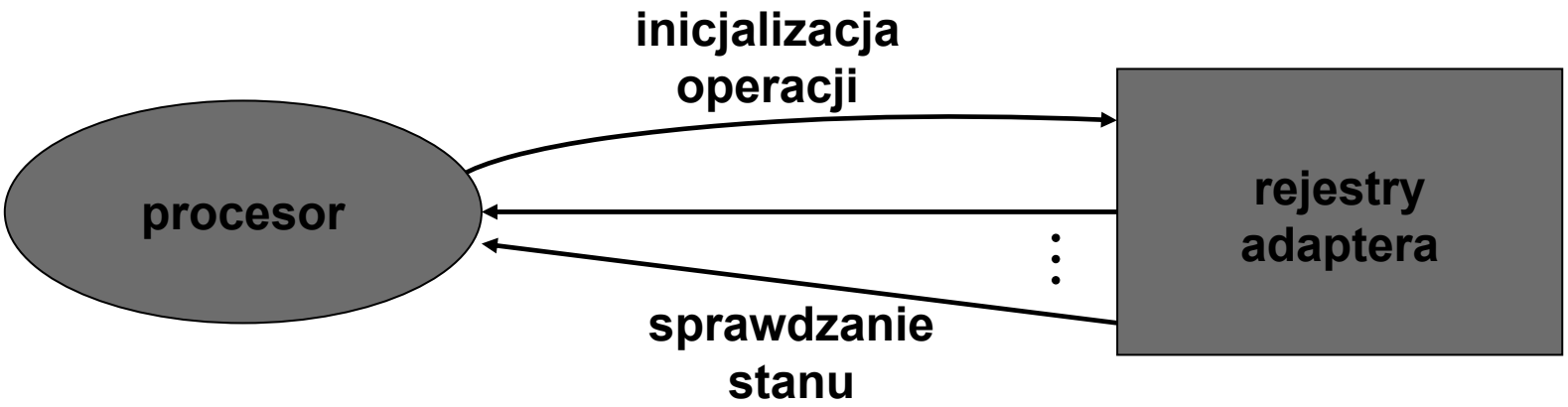
Typowe rejestry sterownika portu (adaptera)

- rejestr stanu (ang. status register) — zawiera bity wskazujące na stan portu (np. zakończenie polecenia, dostępność bajtu, błąd urządzenia itp.), może być czytany przez procesor
- rejestr sterowania (ang. control register, command register) — zawiera bity definiujące tryb pracy urządzenia lub umożliwiające rozpoczęcie realizacji polecenia, jest najczęściej zapisywany przez procesor
- rejestr danych wejściowych (ang. data-in register) — jest czytany przez procesor w celu odbioru danych z urządzenia
- rejestr danych wyjściowych (ang. data-out register) — jest zapisywany przez procesor w celu wysłania danych do urządz.

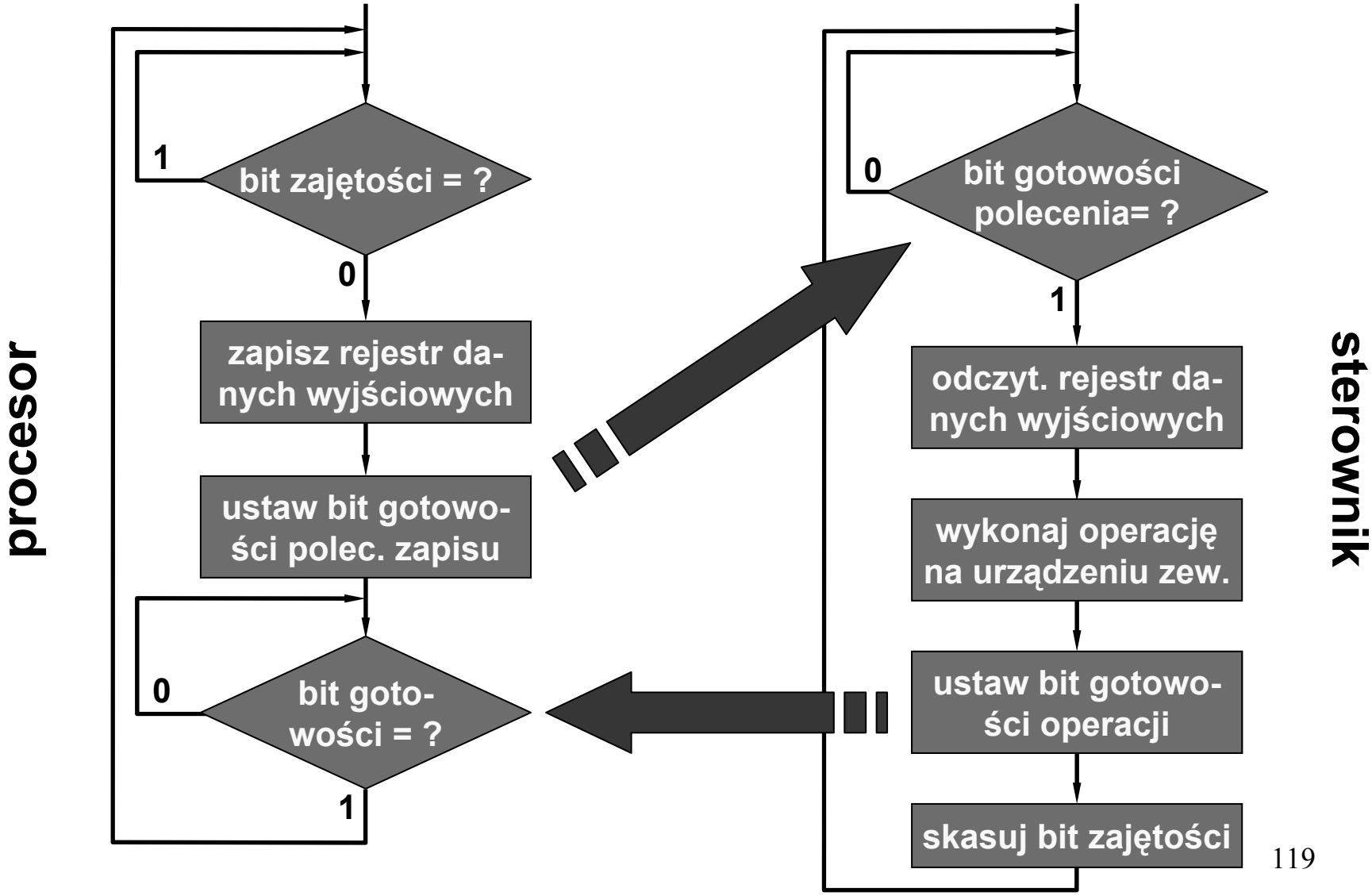
Interakcja jednostki centralnej ze sterownikiem urządzenia wejścia-wyjścia

- Odpytywanie (ang. polling) — procesor co jakiś czas (w szczególności bez przerwy) wykonuje rozkaz odczytu odpowiedniego rejestru sterownika, sprawdzając jego stan.
- Sterowanie przerwaniami (ang. interrupt-driven I/O) — procesor inicjalizuje pracę sterownika a o jej zakończeniu lub zaistnieniu określonego stanu informowany jest przez przerwanie, które zgłasza sterownik.
- Bezpośredni dostęp do pamięci (ang. direct memory access) — zadanie przekazywania danych pomiędzy sterownikiem a pamięcią spada na specjalizowany układ (DMA), który wykonuje swoje zadanie bez angażowania procesora.

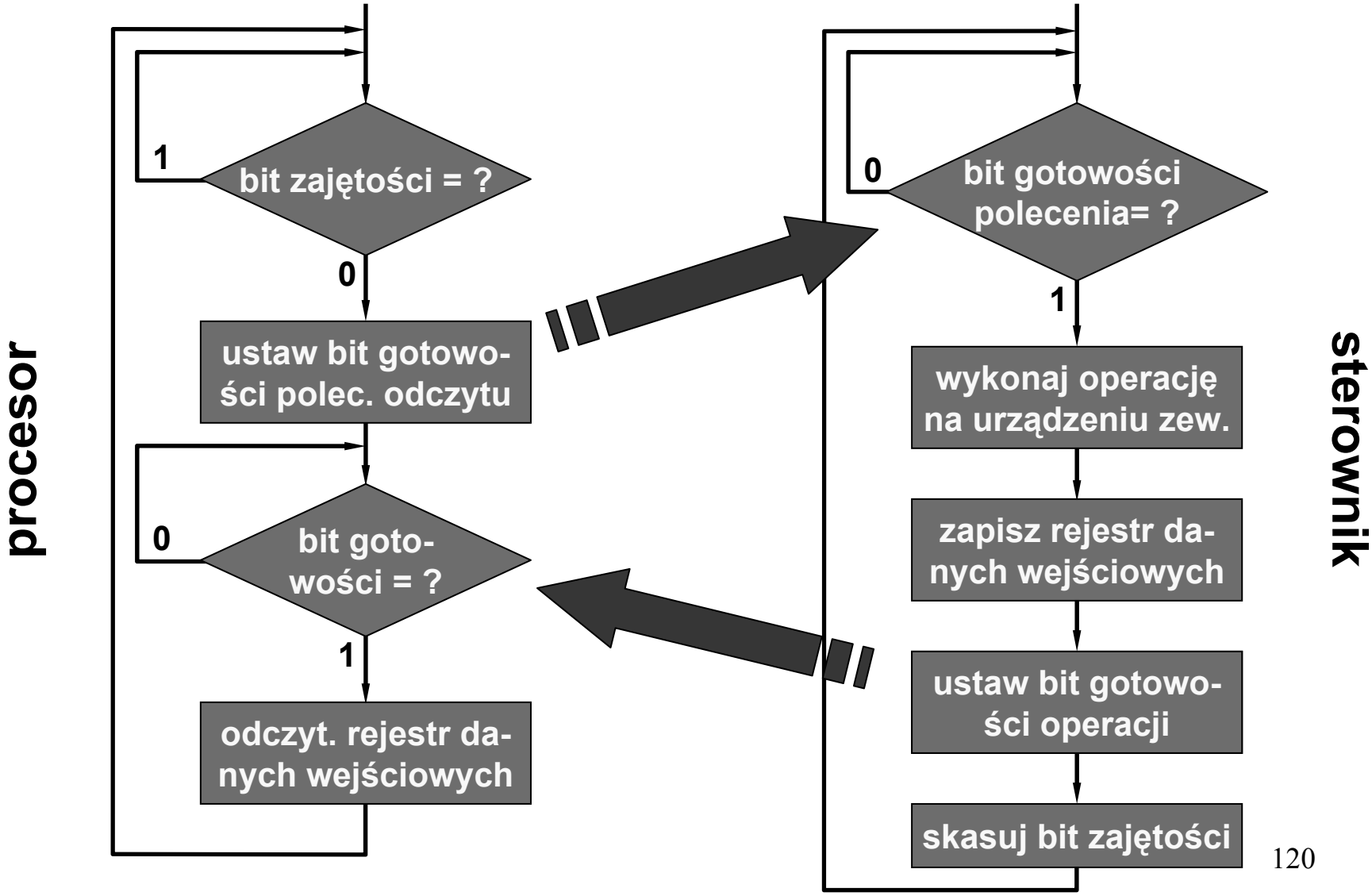
Odpytywanie



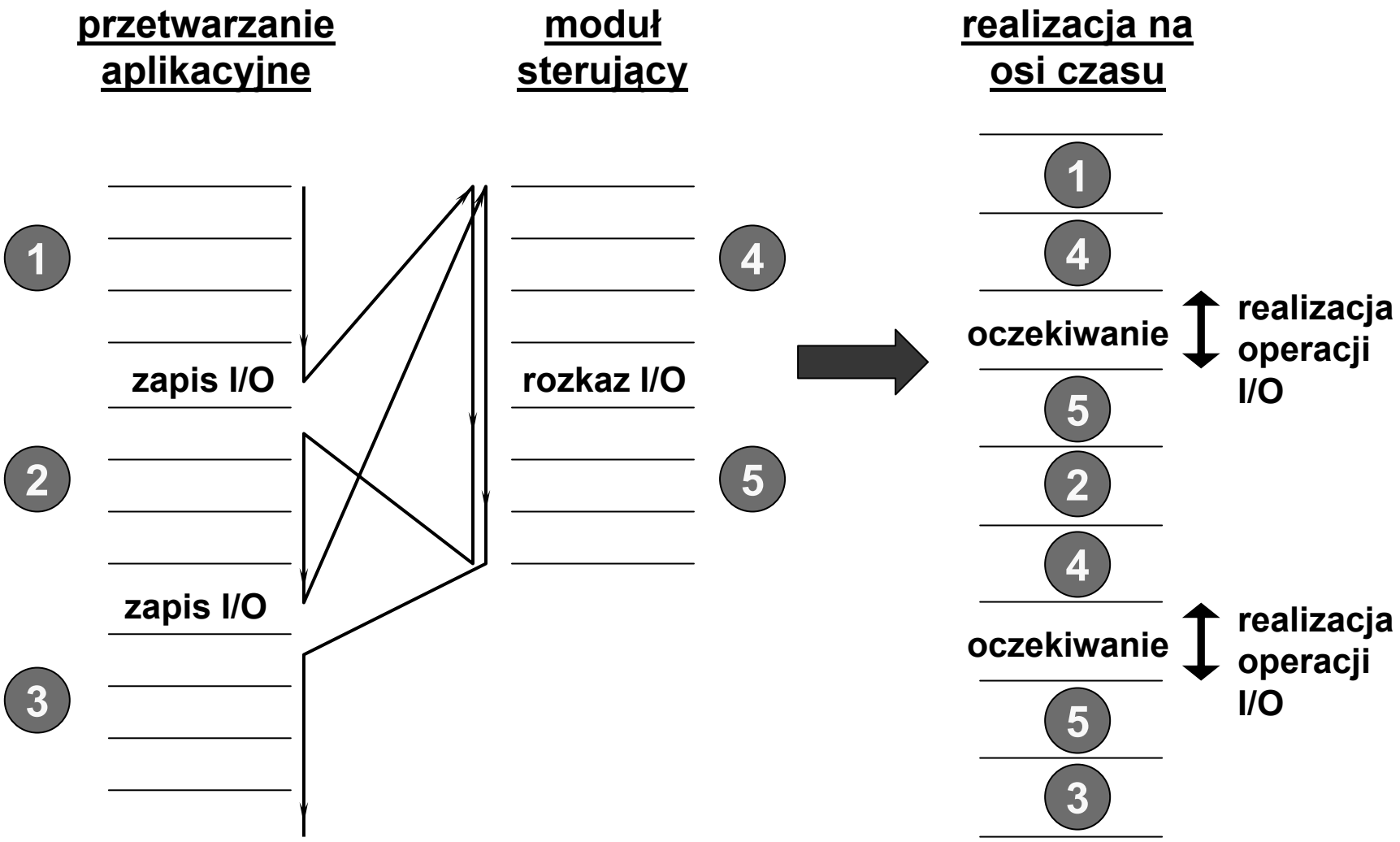
Interakcja procesor — sterownik w oper. wyjścia w trybie odpytywania



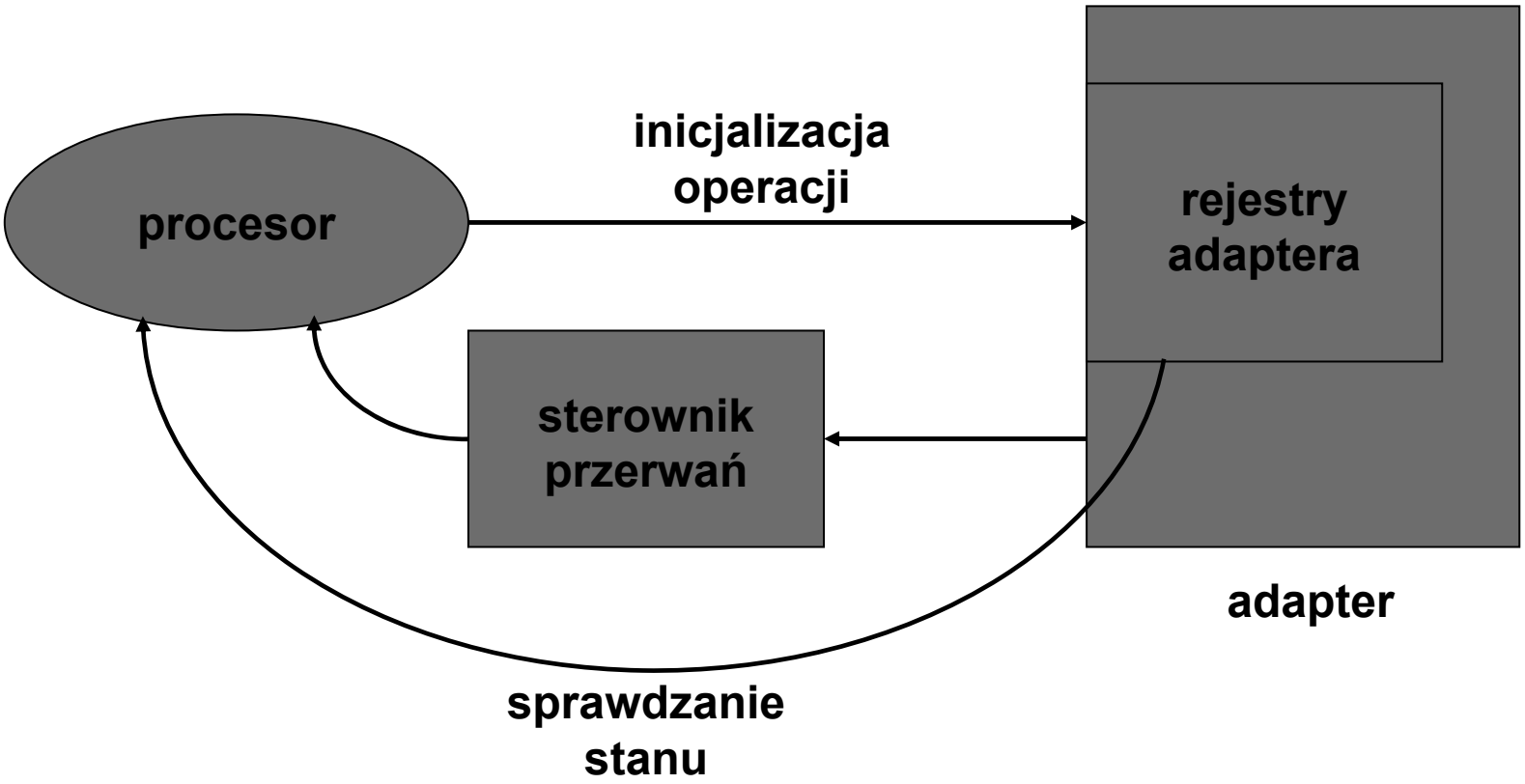
Interakcja procesor — sterownik w oper. wejścia w trybie odpytywania



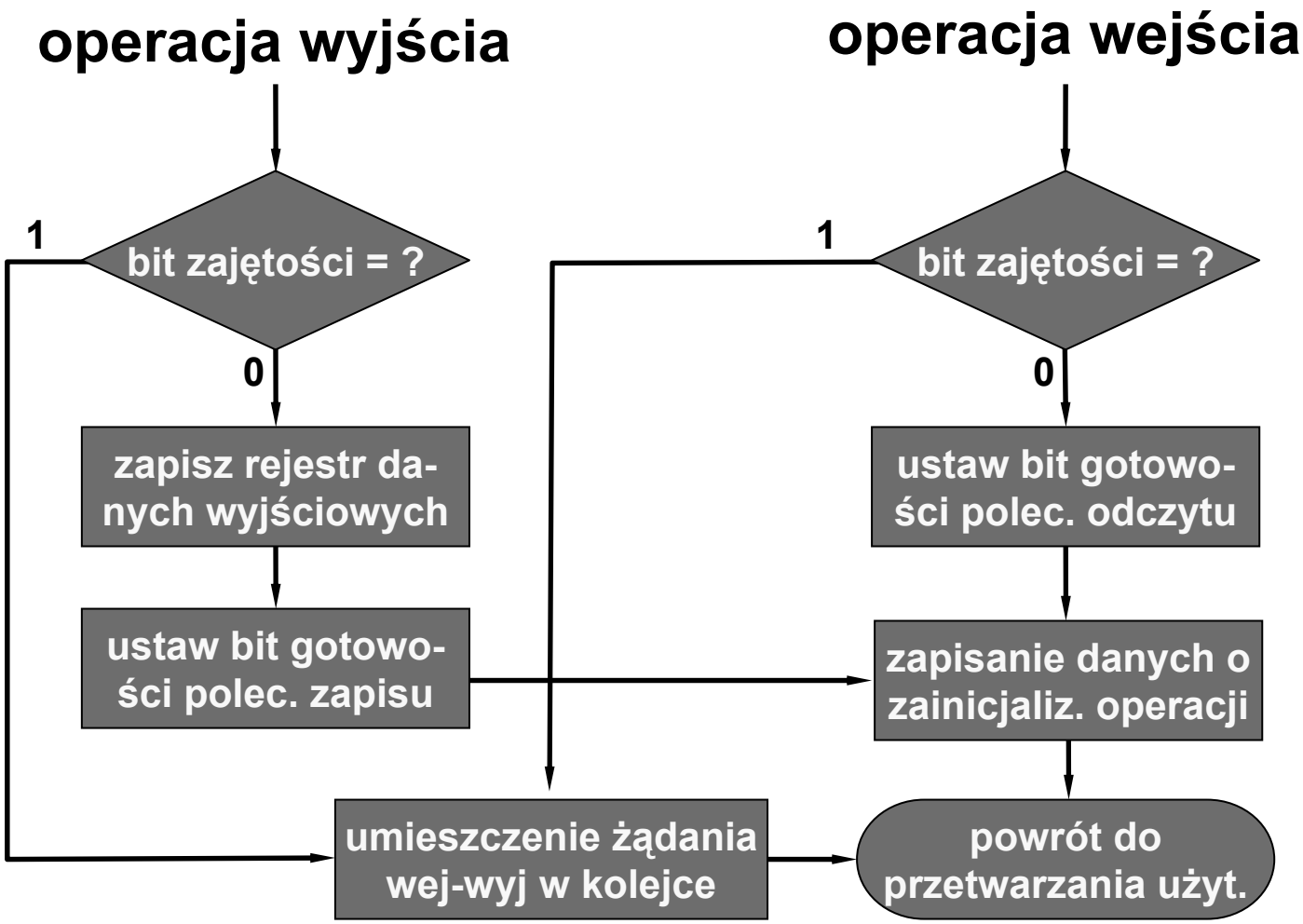
Odpytywanie — przepływ sterowania



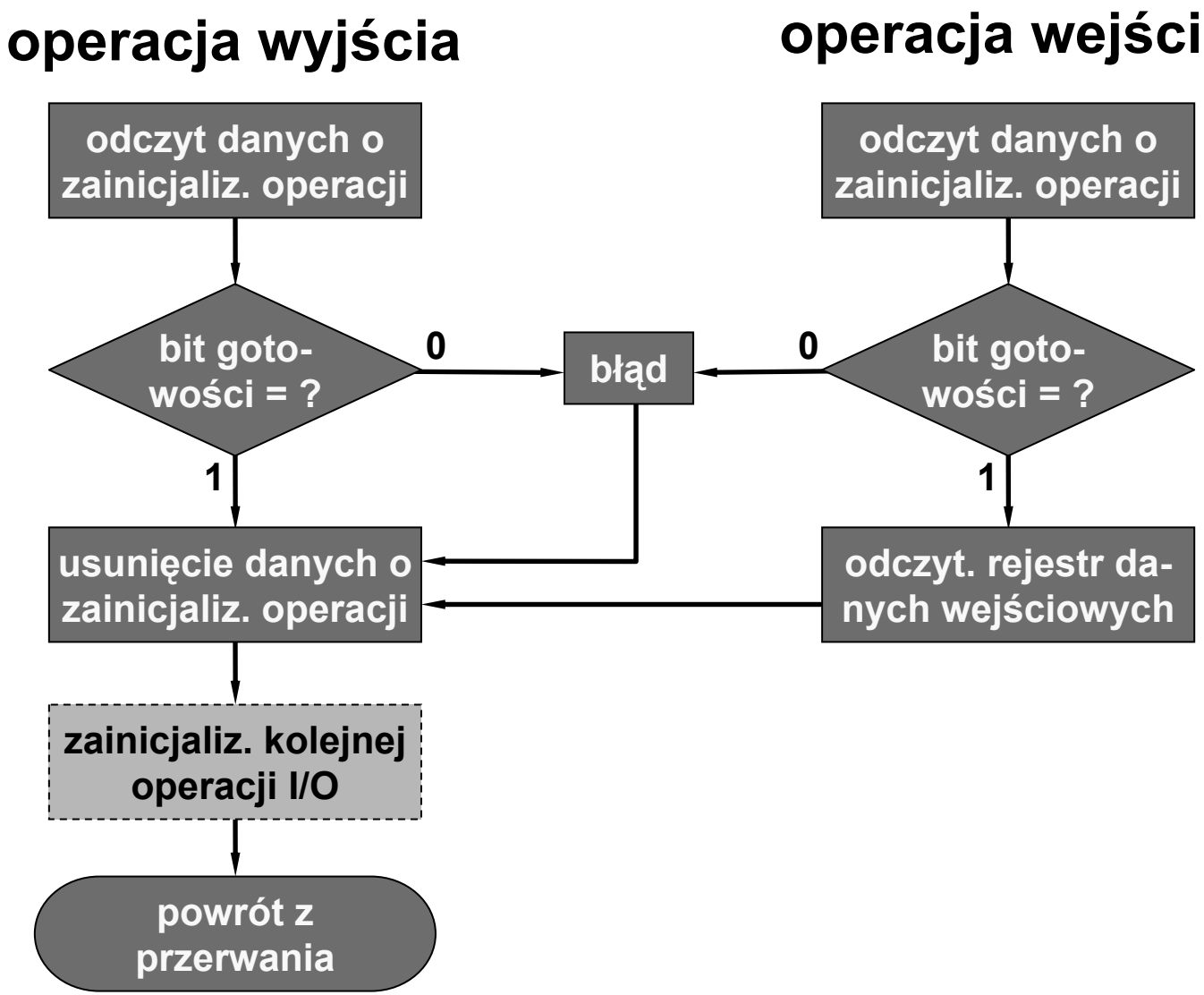
Sterowanie przerwaniami



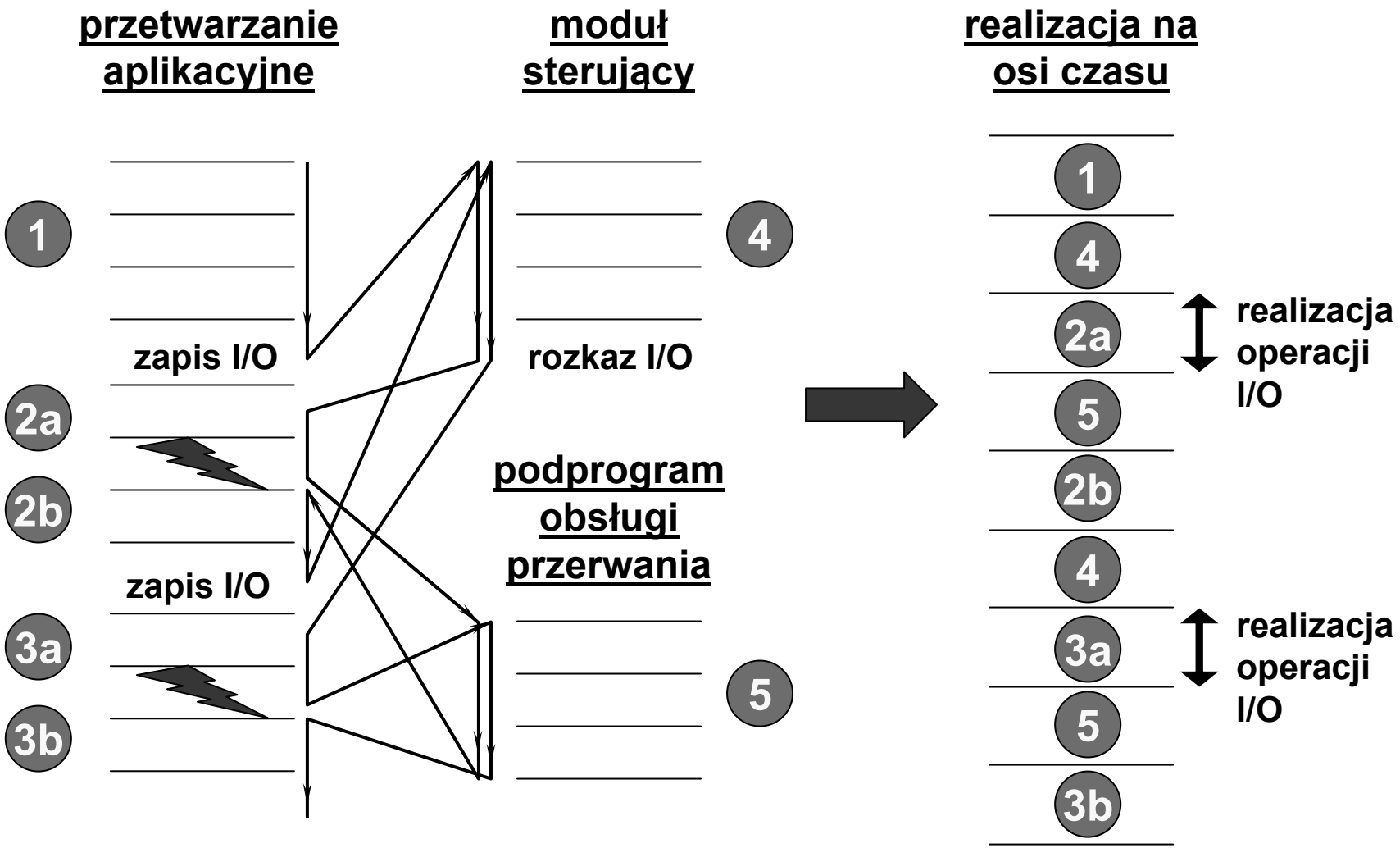
Sterowanie przerwaniami — inicjalizacja operacji



Sterowanie przerwaniami — obsługa przerwania



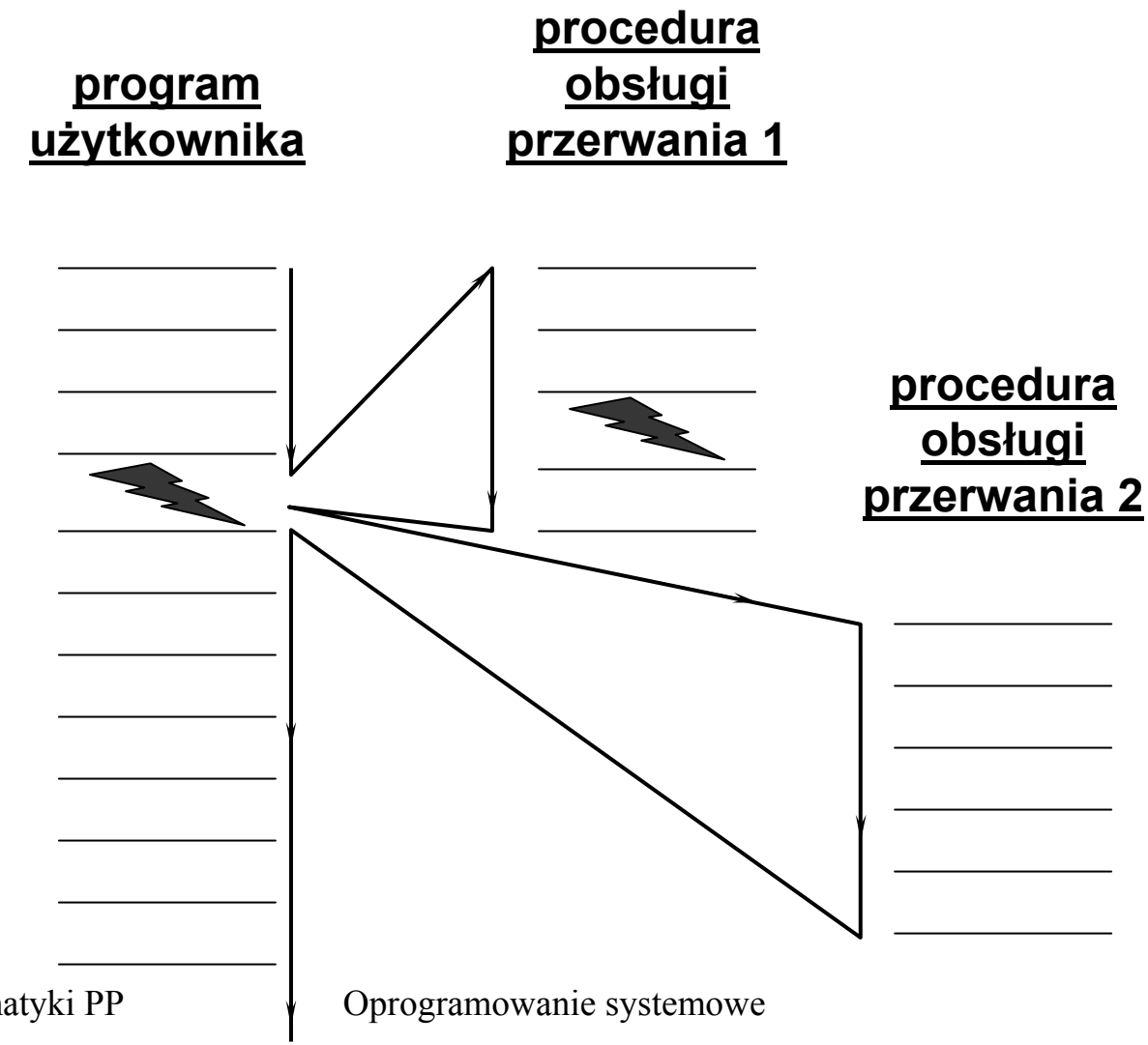
Sterowanie przerwaniami — przepływ sterowania



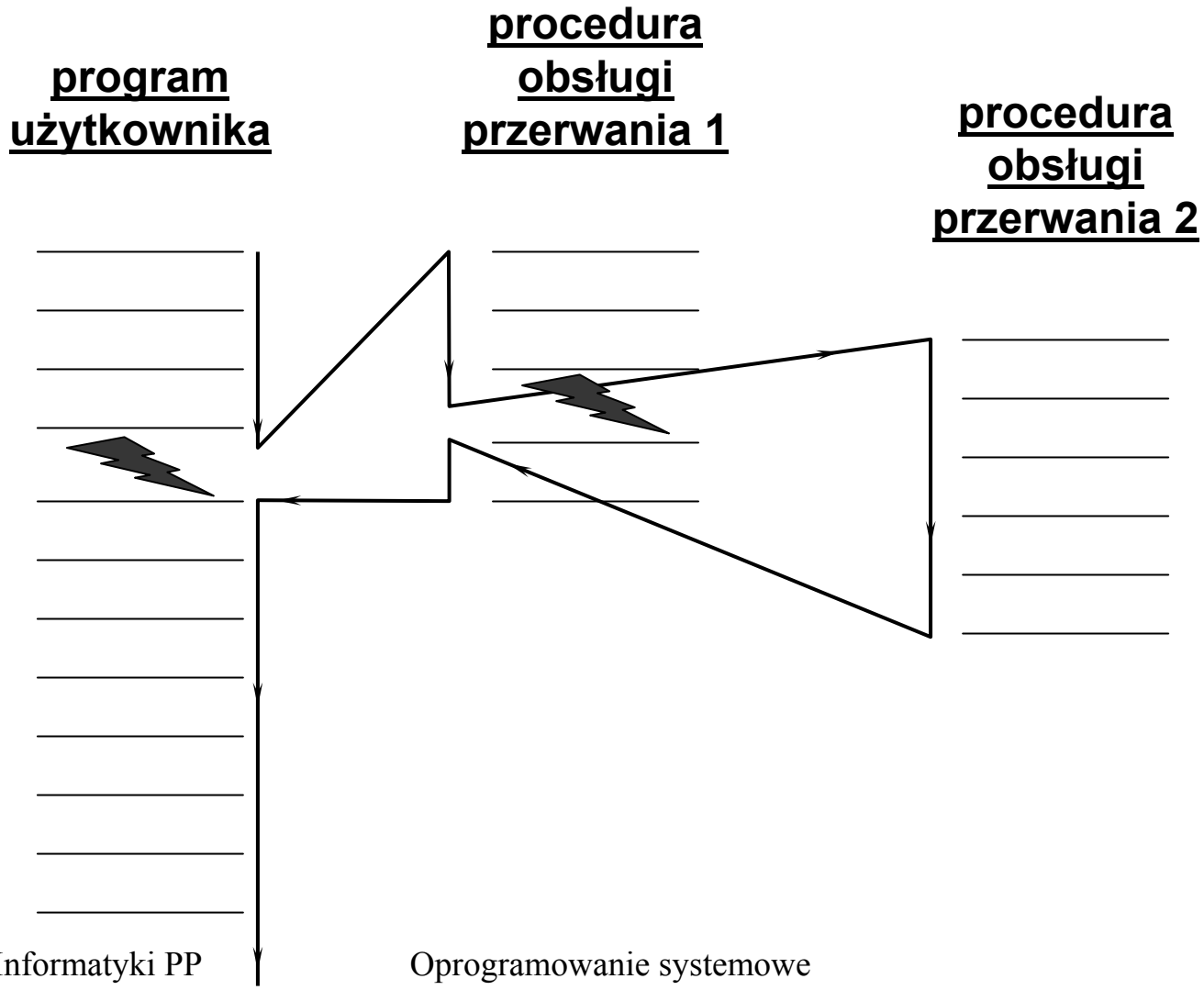
Obsługa przerwań wielokrotnych

- Problem przerwań wielokrotnych polega na zgłoszeniu kolejnego przerwania w czasie obsługi innego przerwania.
- Podejścia do obsługi przerwań wielokrotnych:
  - obsługa sekwencyjna — kolejne przerwanie (zgłoszone podczas obsługi) obsługiwane jest po zakończeniu obsługi bieżącego przerwania),
  - obsługa zagnieżdżona — po zgłoszeniu nowego przerwania obsługa bieżącego przerwania jest zawieszana i kontynuowana po obsłużeniu przerwania nowo zgłoszonego,
  - obsługa priorytetowa — zawieszenie obsługi bieżącego przerwania następuje tylko wówczas, gdy nowo zgłoszone przerwanie ma wyższy priorytet, w przeciwnym razie obsługa następuje po obsłużeniu wszystkich zgłoszonych przerwań o wyższym priorytecie

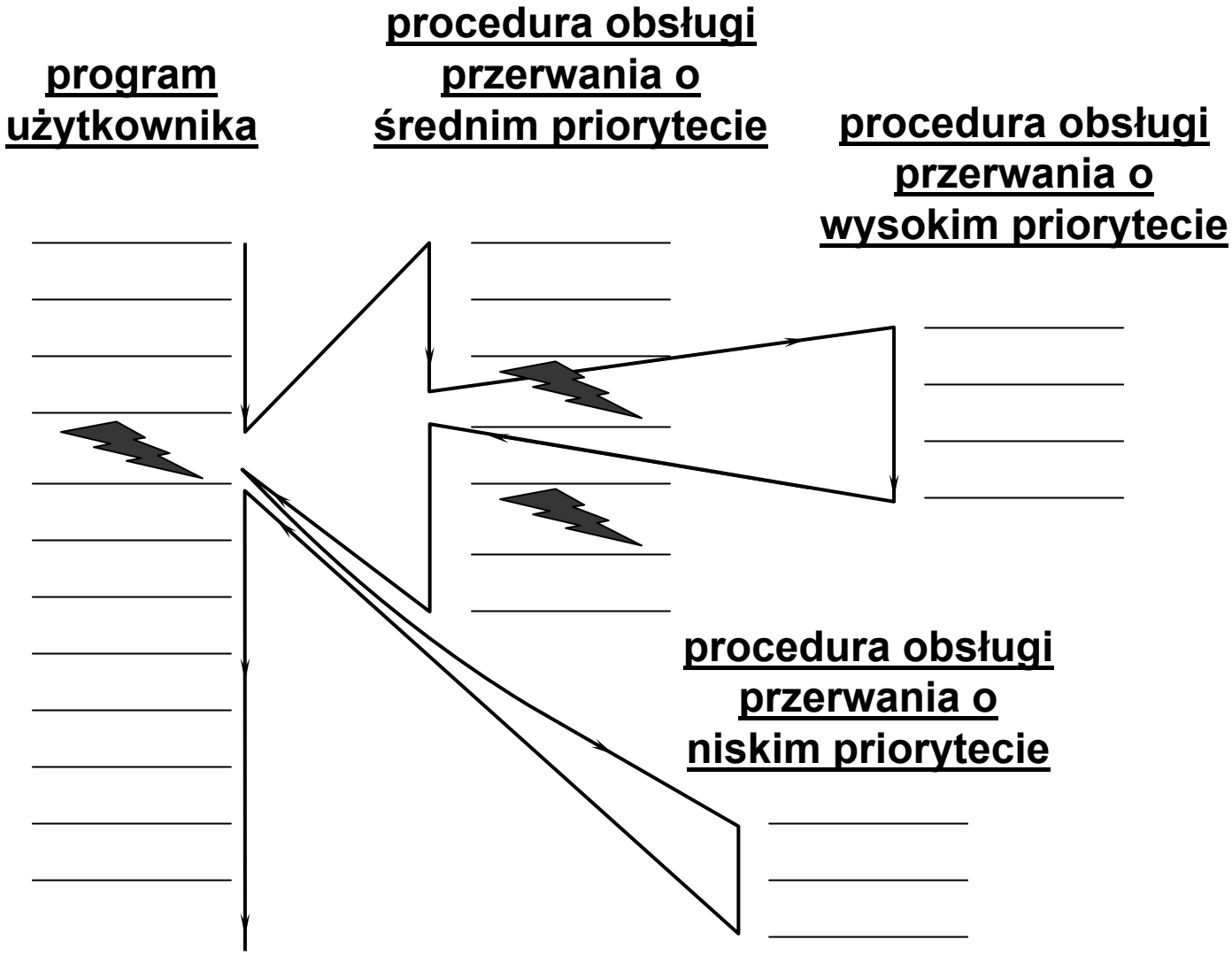
Sekwencyjna obsługa przerwań — przepływ sterowania



Zagnieżdżona obsługa przerwań — przepływ sterowania



Priorytetowa obsługa przerwań — przepływ sterowania



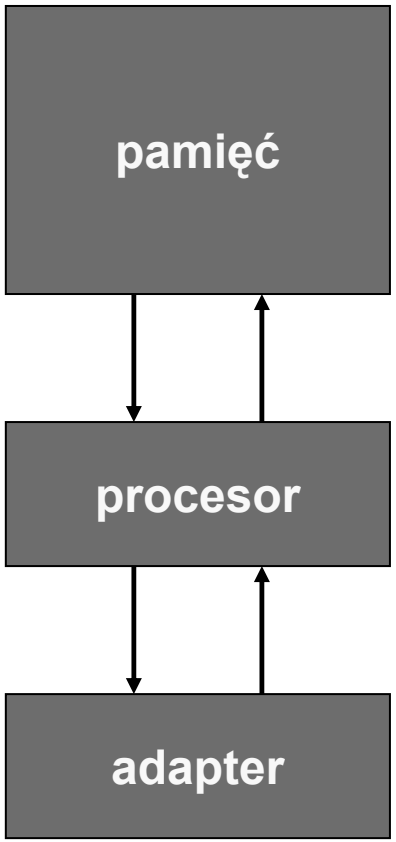
Problemy współbieżnej obsługi wielu urządzeń

- Problem identyfikacji źródła przerwania — zidentyfikowanie urządzenia, które poprzez zgłoszenie przerwania wymusiło przekazanie sterowania do procedury obsługi przerwania.
- Problem priorytetów — zagwarantowanie określonej kolejności wyboru urządzeń w przypadku deklaracji gotowości kilku z nich w tym samym czasie.

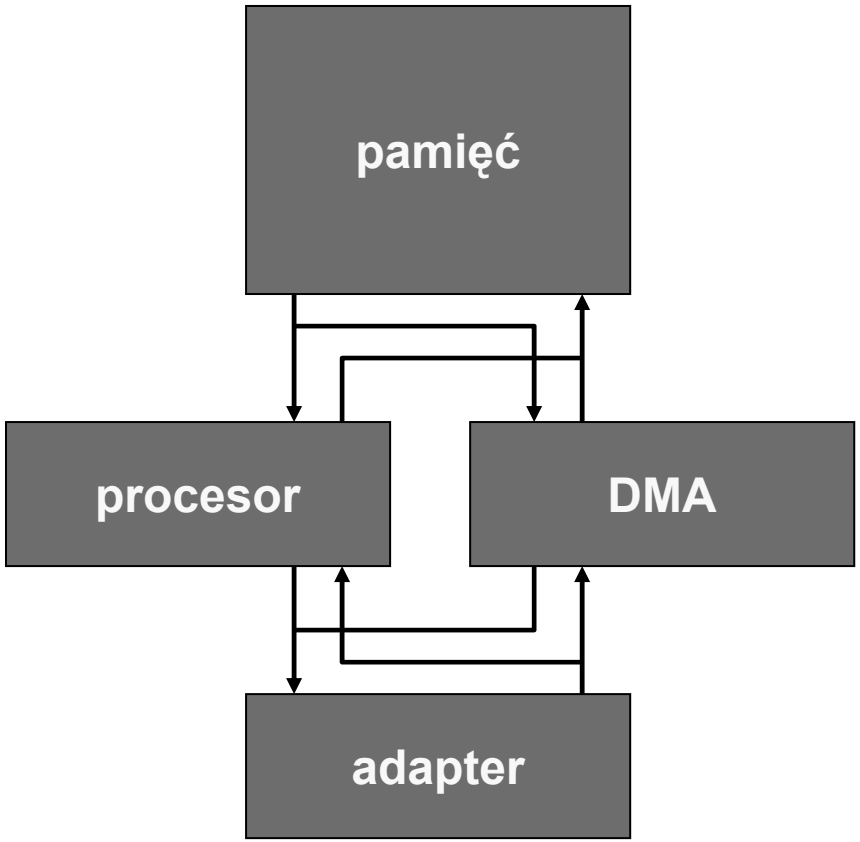
Sposoby identyfikacji źródła przerwania

- Wiele linii przerwań — doprowadzenie do procesora osobnej linii przerwania dla każdego urządzenia i przygotowanie osobnej procedury obsługi przerwania dla każdej linii.
- Odpytywanie programowe — odczyt rejestru stanu i sprawdzanie bitu gotowości każdego urządzenia, które mogło potencjalnie zgłosić przerwanie.
- Odpytywanie sprzętowe — wysłanie sygnału potwierdzającego otrzymanie sygnału przerwania propagowanego łańcuchowo przez urządzenia aż do napotkania tego, które zgłosiło przerwanie i które w odpowiedzi wystawi odpowiedni wektor na magistralę.
- Arbitraż na magistrali — uzyskanie przed zgłoszeniem przerwania wyłączności dostępu do magistrali i wystawienie odpowiedniego wektora.

Bezpośredni dostęp do pamięci



tradycyjne I/O

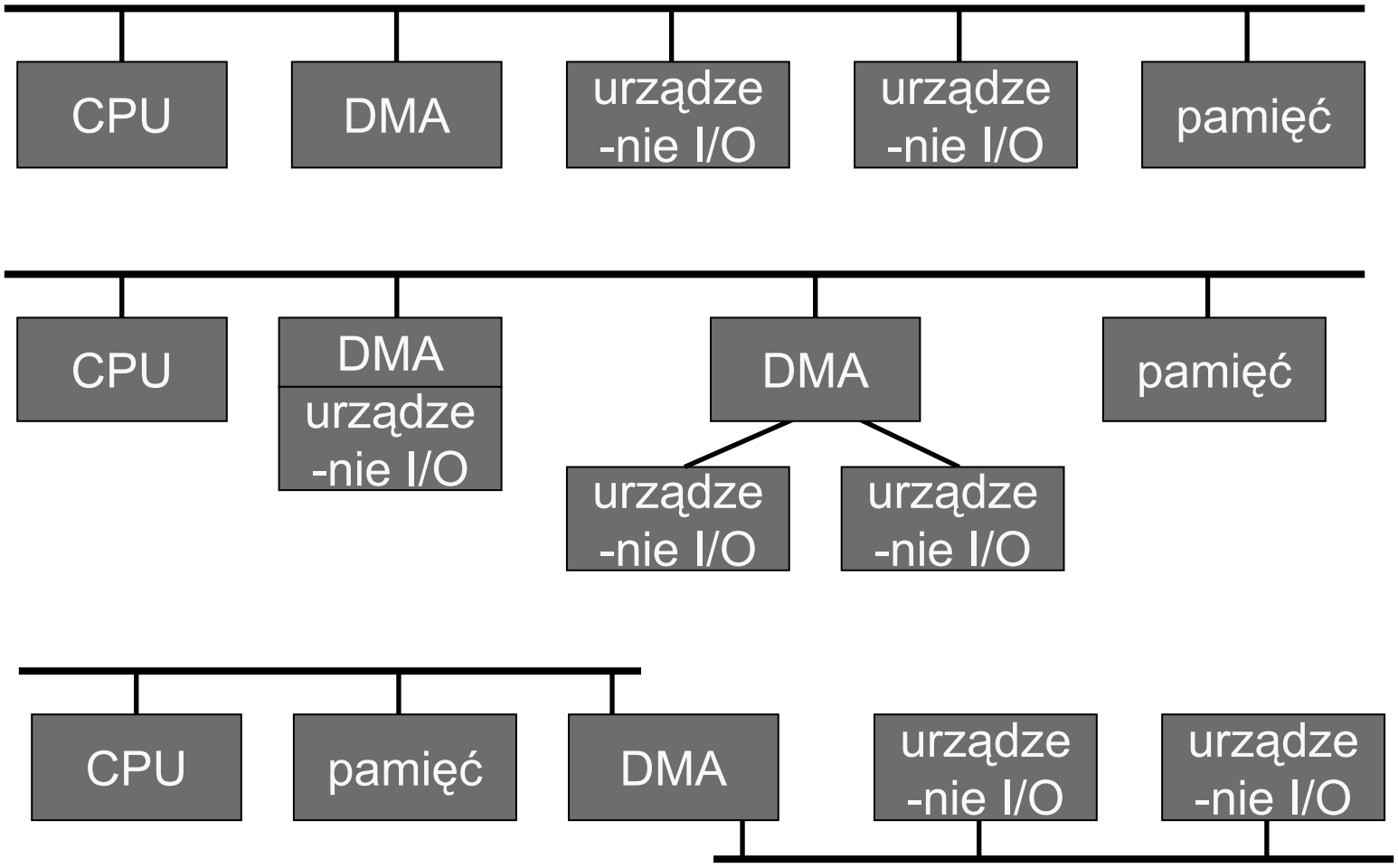


I/O z DMA

Działanie układu DMA

- Układ DMA ma „kompetencje” procesora w zakresie dostępu do pamięci, w związku z czym może rywalizować z procesorem o dostęp do magistrali systemowej w celu przejęcia sterowania systemem komputerowym.
- Procesor zleca układowi DMA transmisję danych przekazując następujące dane:
  - rodzaj operacji (zapis lub odczyt bloku w pamięci),
  - adres urządzenia wejścia-wyjścia
  - początkowy adres bloku w pamięci do zapisu lub odczytu
  - rozmiar zapisywanego lub odczytywanego bloku w bajtach lub słowach
- W celu realizacji zlecenia układ DMA przejmuje kontrolę nad magistralą, gdy nie jest ona potrzebna procesorowi lub „wykrada” cykl magistrali procesorowi, realizuje przesłanie słowa.
- Fakt zakończenia transmisji układ DMA sygnalizuje procesorowi zgłaszając przerwanie.

Organizacja wejścia-wyjścia z układem DMA

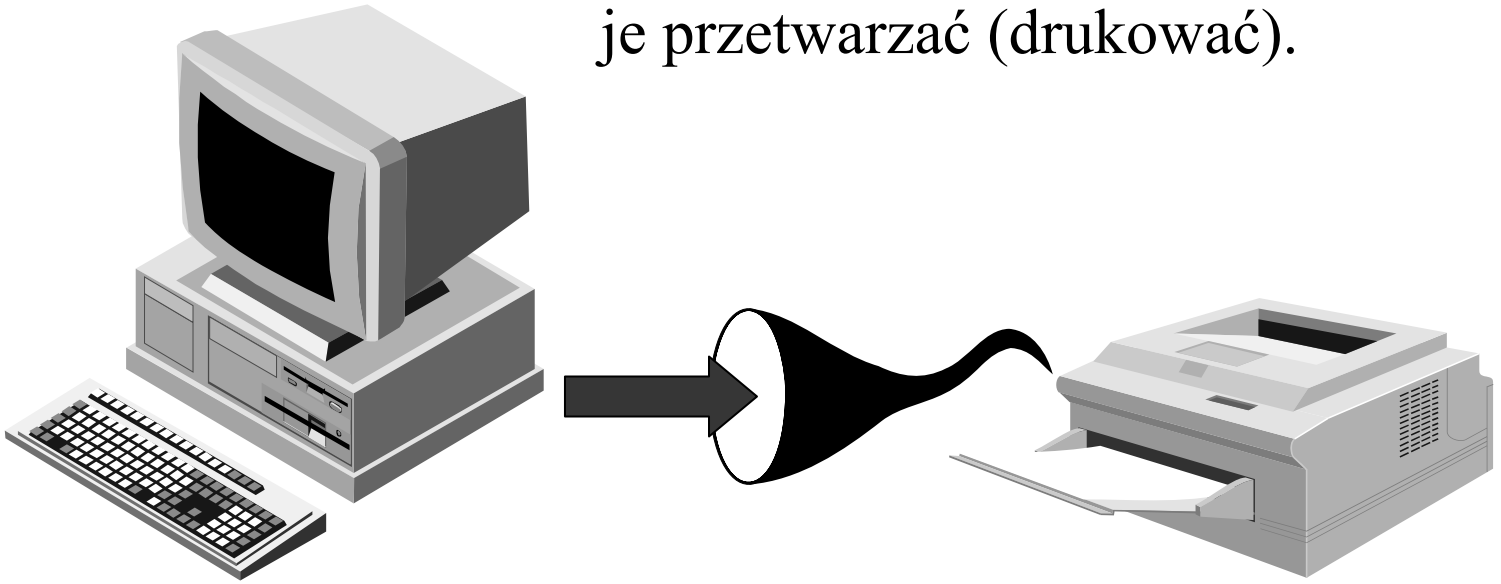


Buforowanie

- Dopasowanie urządzeń różniących się szybkością przekazywania danych — dopasowanie znacznie szybszego producenta danych do możliwości konsumenta.
- Dopasowanie urządzeń różniących się podstawową jednostką transmisji danych — dopasowanie w celu efektywnego przekazywania danych urządzeń przesyłających mniejsze jednostki danych do urządzeń wymagających większych jednostek lub odwrotnie (fragmentowanie).
- Semantyka kopii — zagwarantowanie niezmienności danych w czasie wykonywania operacji wejścia-wyjścia.

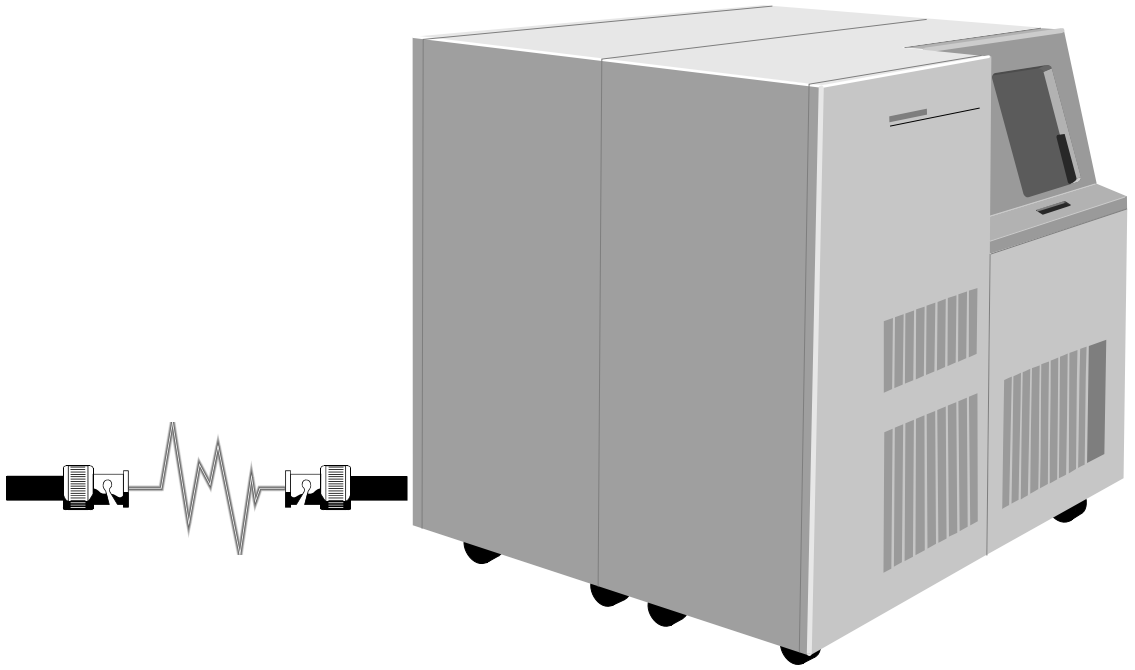
Dopasowanie różnic szybkości

Przykład: komputer potrafi wysyłać dane znacznie szybciej niż drukarka je przetwarzać (drukować).

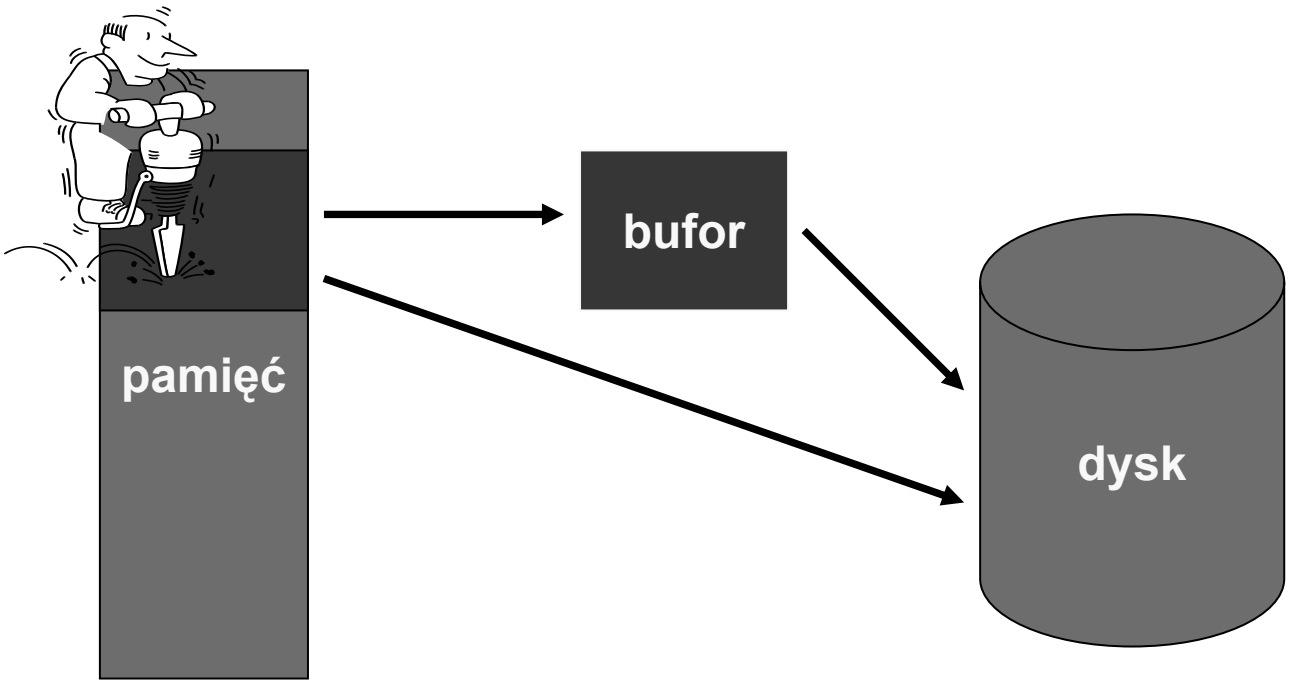


Dopasowanie jednostek transmisji

Przykład: zapis na dysku danych odbieranych z sieci.

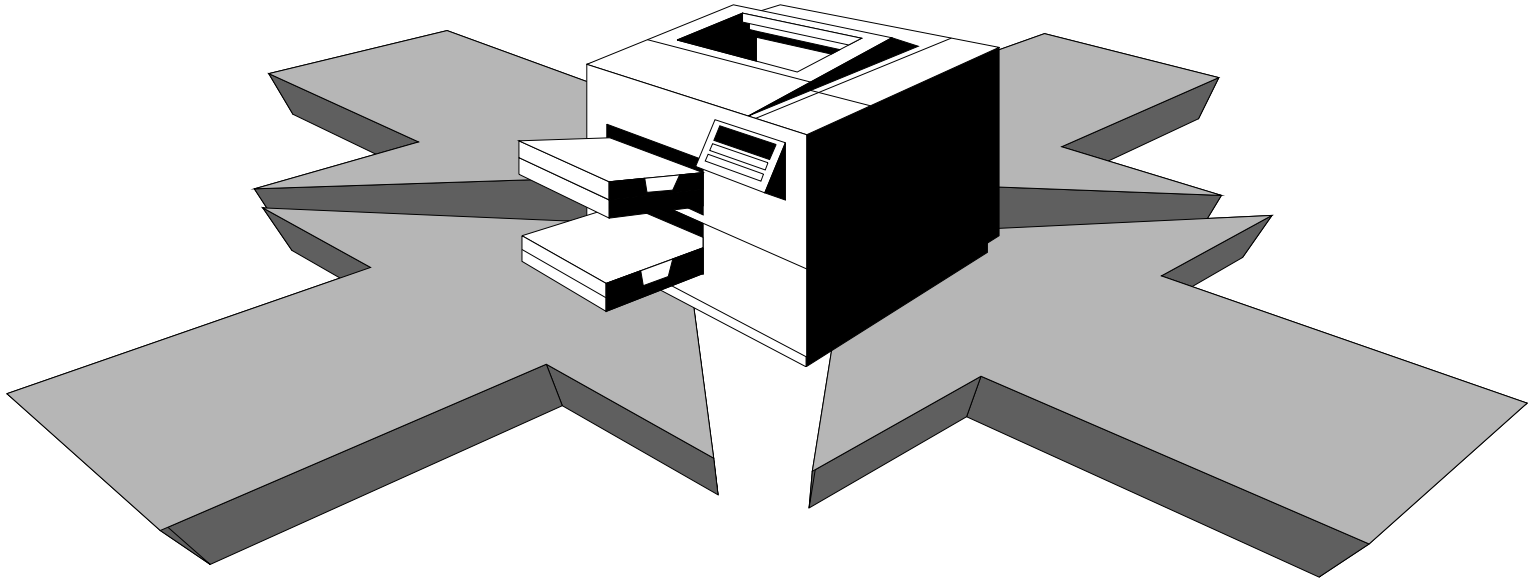


Semantyka kopii



Spooling

Buforowanie strumienia danych przekazywanych do urządzenia zewnętrznego, które nie mogą być przeplatane z danymi pochodzącymi z innych strumieni.



System plików

1. Pojęcie pliku
2. Typy i struktury plików
3. Metody dostępu do plików
4. Katalogi
5. Budowa systemu plików
6. Buforowanie
7. Integralność systemu plików
8. Semantyka spójności

Pojęcie pliku (ang. file)

- Plik jest abstrakcyjnym obrazem informacji gromadzonej i udostępnianej przez system komputerowy.
- Plik jest podstawową jednostką logiczną magazynowania informacji w systemie komputerowym, widoczną dla użytkownika.
- Plik jest nazwanym zbiorem powiązanych ze sobą informacji, zapisanym w pamięci pomocniczej (najczęściej nieulotnej, czyli trwałej).
- Zadaniem systemu operacyjnego w odniesieniu do plików jest zapewnienie odwzorowania pomiędzy abstrakcyjnym obrazem pliku a urządzeniami fizycznymi.

Atrybuty pliku

- **Nazwa** — ciąg znaków służących użytkownikowi do identyfikacji pliku
- **Typ** — informacja służąca do rozróżnienie typów plików
- **Lokalizacja** — informacja służąca do odnalezienia pliku w systemie komputerowym (urządzenie i położenie pliku w tym urządzeniu)
- **Rozmiar** — bieżący rozmiar pliku w ustalonych jednostkach (bajtach, słowach, blokach itp.)
- **Ochrona** — informacje umożliwiające kontrolę dostępu
- **Czasy dostępów** — daty i czasy wykonywania pewnych operacji na pliku, typu odczyt, modyfikacja, utworzenie

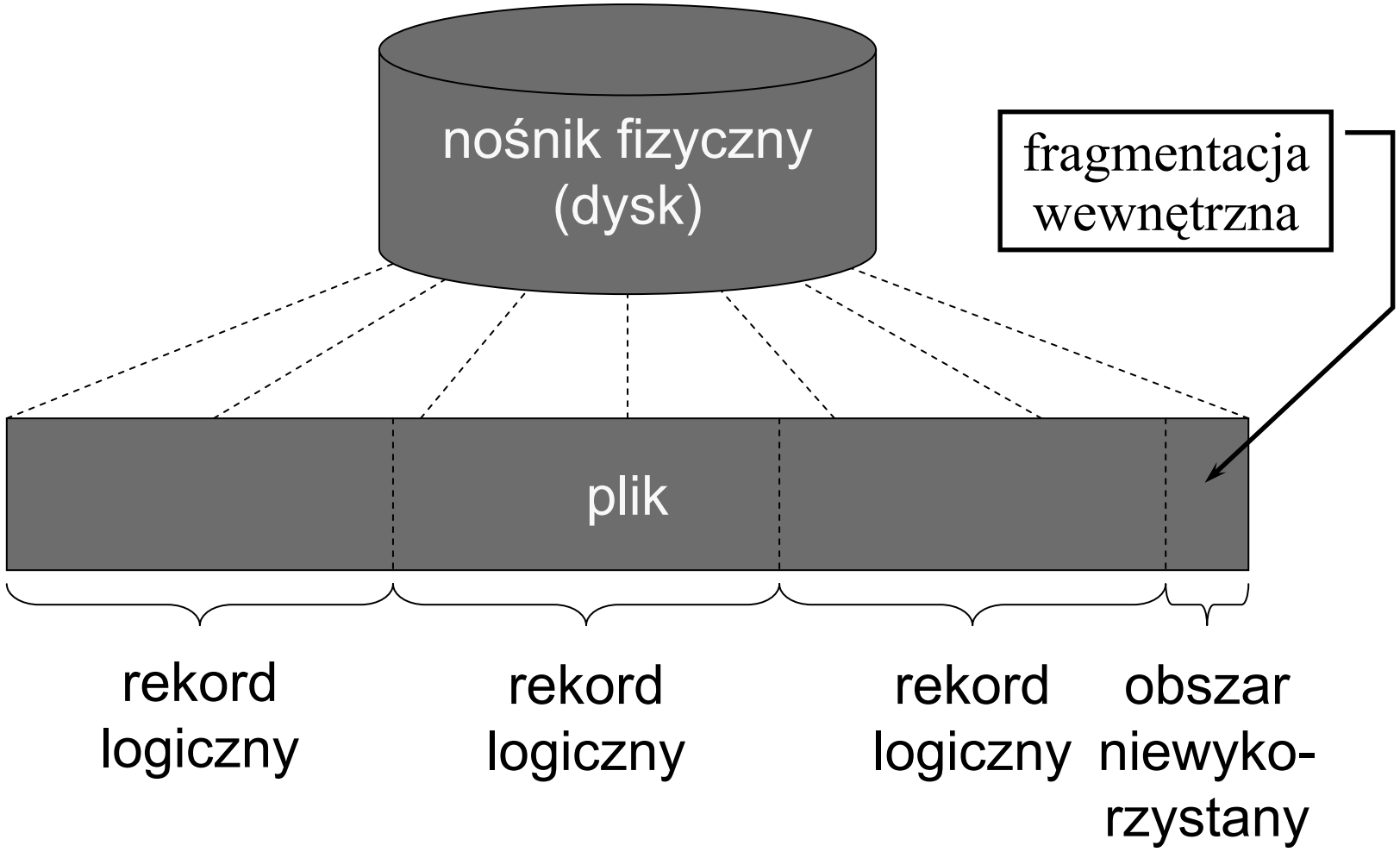
Typy plików

- Typ pliku określa rodzaj informacji przechowywanej w pliku i tym samym sposób interpretacji jego zawartości, np. program binarny, wynik kompilacji, kod źródłowy, makrodefinicja (plik wsadowy, skrypt powłoki itp.), tekst, biblioteka programisty, grafika, dane aplikacji.
- Informacja o typie pliku może być przechowywana w strukturach wewnętrznych systemu plików, w zawartości samego pliku, w katalogach lub w nazwie pliku.
- Typ pliku może być rozpoznawany przez system operacyjny, ale może to być również tylko informacja interpretowana przez użytkownika lub aplikację.

Struktura pliku

- Struktura pliku określa jego wewnętrzną organizację.
- Struktura może być definiowana i rozpoznawana przez system operacyjny lub może być rozpoznawana na poziomie aplikacji korzystającej z tego pliku.
- Definiowanie różnych struktur plików na poziomie systemu operacyjnego może być pomocne dla użytkownika, ale w systemie musi być wówczas zawarty kod do obsługi każdej z tych struktur.
- Wyróżniana jest też wewnętrzna (fizyczna) struktura plików, narzucana przez urządzanie, które ten plik przechowuje

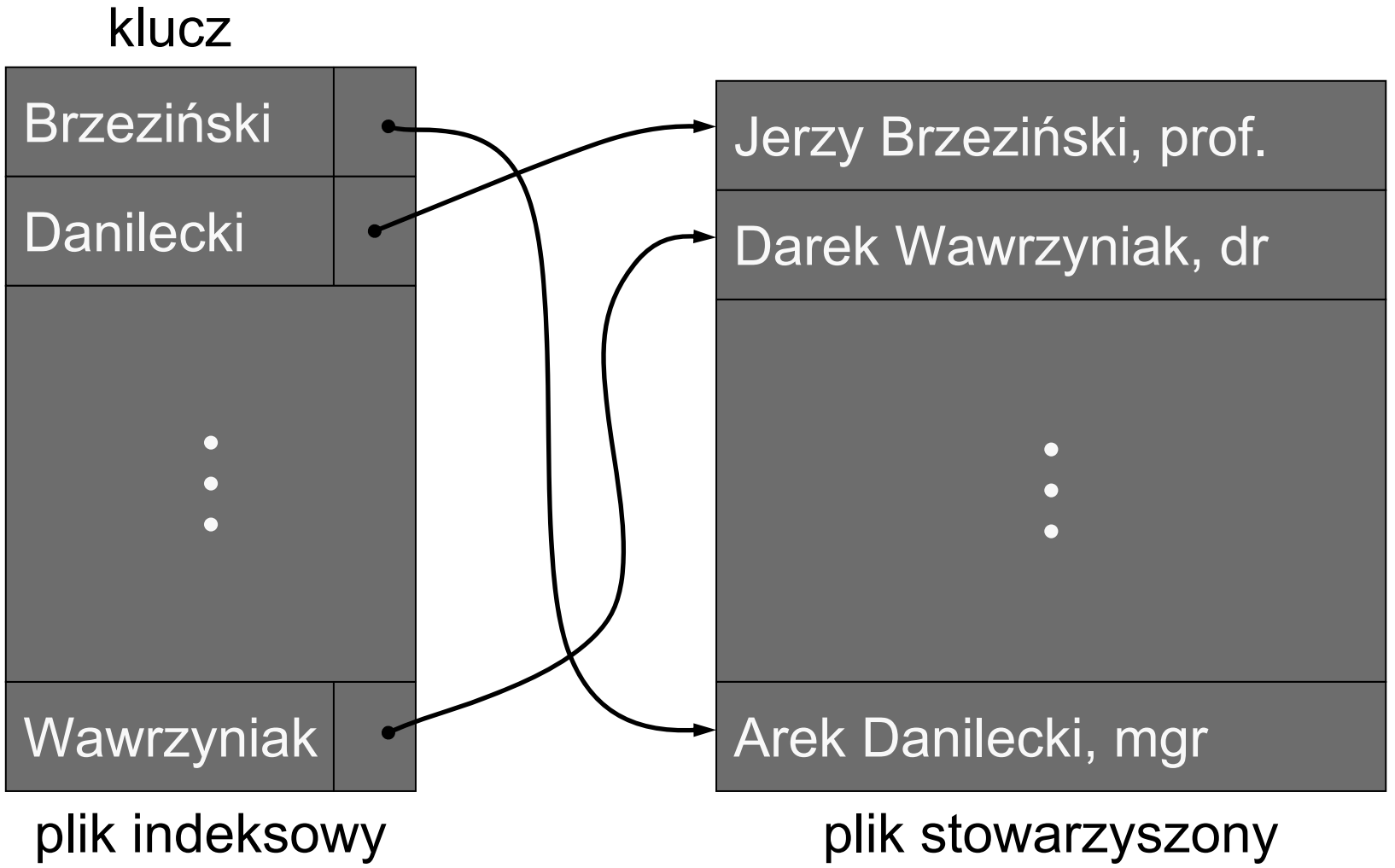
Odwzorowanie struktury logicznej na fizyczną



Metody dostępu do plików

- Metody dostępu do plików określają sposób wykonywania operacji na plikach w celu udostępnienia znajdującej się w nich informacji.
- Wyróżnia się następujące metody dostępu:
  - dostęp sekwencyjny (ang. sequential access) — informacje w pliku przetwarzane są rekord po rekordzie, tzn. po wykonaniu operacji na określonym rekordzie system przygotowuje się do wykonania operacji na kolejnym rekordzie w pliku,
  - dostęp bezpośredni (ang. direct access) — lokalizacja rekordu do przetwarzania podawana jest jako parametr odpowiedniej operacji,
  - dostęp indeksowy — rekord, na którym ma być wykonana operacja identyfikowany jest przez klucz, odwzorowywany na konkretny rekord w pliku stowarzyszonym poprzez plik indeksowy.

Przykład pliku indeksowego



Podstawowe operacje na plikach udostępniane przez system operacyjny (1)

- Tworzenie pliku — konieczne jest określenie podstawowych atrybutów pliku, znalezienie miejsca na ten plik w systemie komputerowym oraz jego zaewidencjonowanie (utworzenie wpisu katalogowego)
- Zapis do pliku — konieczne jest określenie, co ma być zapisane i gdzie ma być zapisane (w którym pliku i w jakim miejscu tego plik, zależnie od sposobu dostępu)
- Odczyt z pliku — konieczne jest określenie, co ma być odczytane (z którego pliku i z jakiego miejsca tego plik, zależnie od sposobu dostępu) i gdzie mają być umieszczone odczytane dane

Podstawowe operacje na plikach udostępniane przez system operacyjny (2)

- Usuwanie informacji z pliku — należy określić jaki fragment pliku (i którego pliku) ma być usunięty. Najczęściej możliwe jest tylko skracanie pliku, czyli usuwanie jego końcowej zawartości lub całej jego zawartości.
- Usuwanie pliku — należy określić plik do usunięcia. Usuwana jest zawartość oraz wpis ewidencyjny pliku.
- Dodatkowe operacje na plikach, wykonywane w celu uzyskania dostępu do zawartości pliku:
  - otwieranie,
  - zamykanie,
  - przesuwanie wskaźnika bieżącej pozycji.

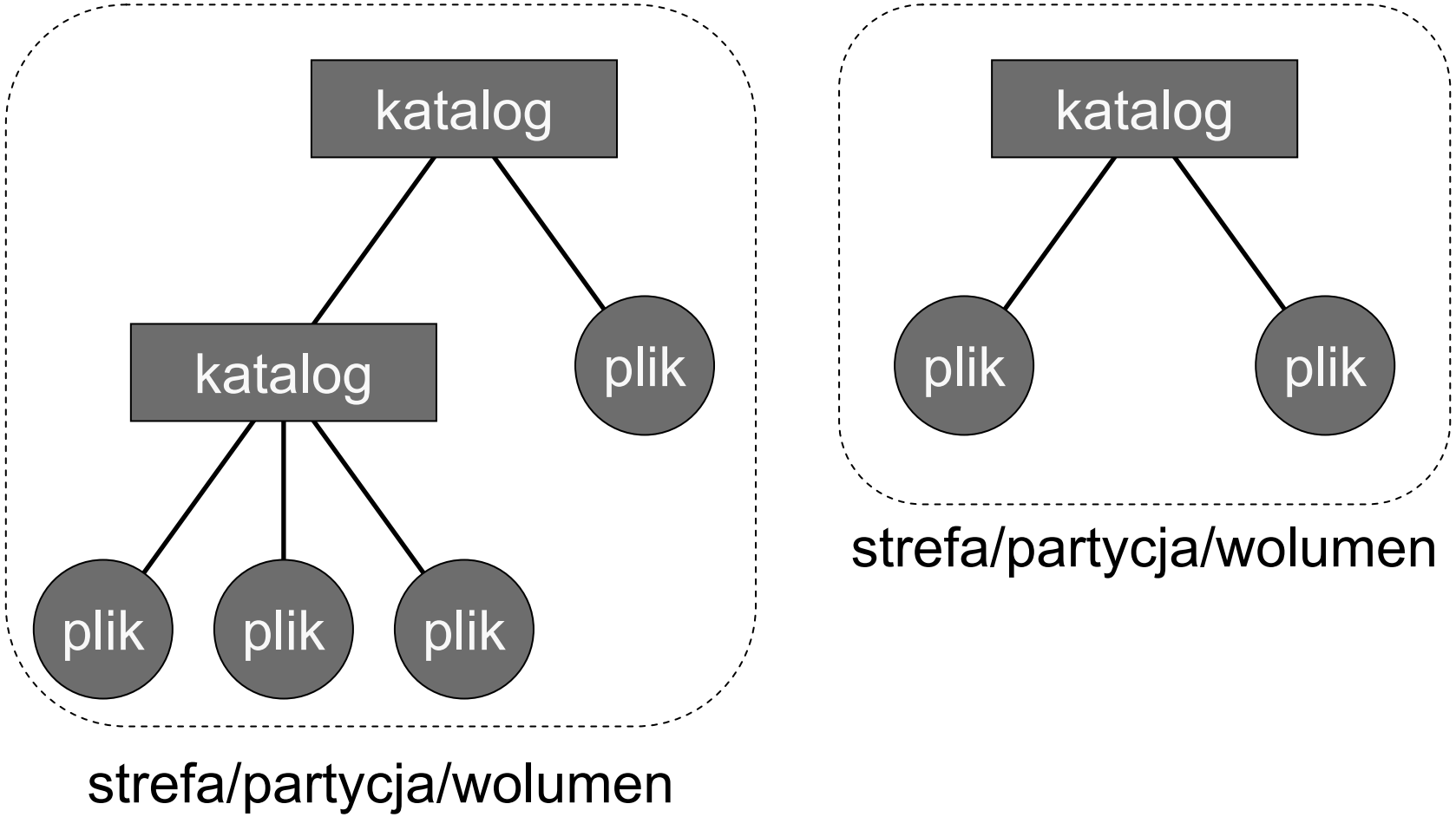
Interfejs dostępu do pliku w systemie UNIX

- Tworzenie pliku — funkcja systemowa **`creat`**
- Otwieranie pliku — funkcja systemowa **`open`**
- Odczyt z pliku — funkcja systemowa **`read`**
- Zapis do pliku — funkcja systemowa **`write`**
- Przesunięcie wskaźnika bieżącej pozycji — funkcja systemowa **`lseek`**
- Zamykanie otwartego pliku — funkcja systemowa **`close`**
- Usuwanie dowiązania do pliku — funkcja systemowa **`unlink`**

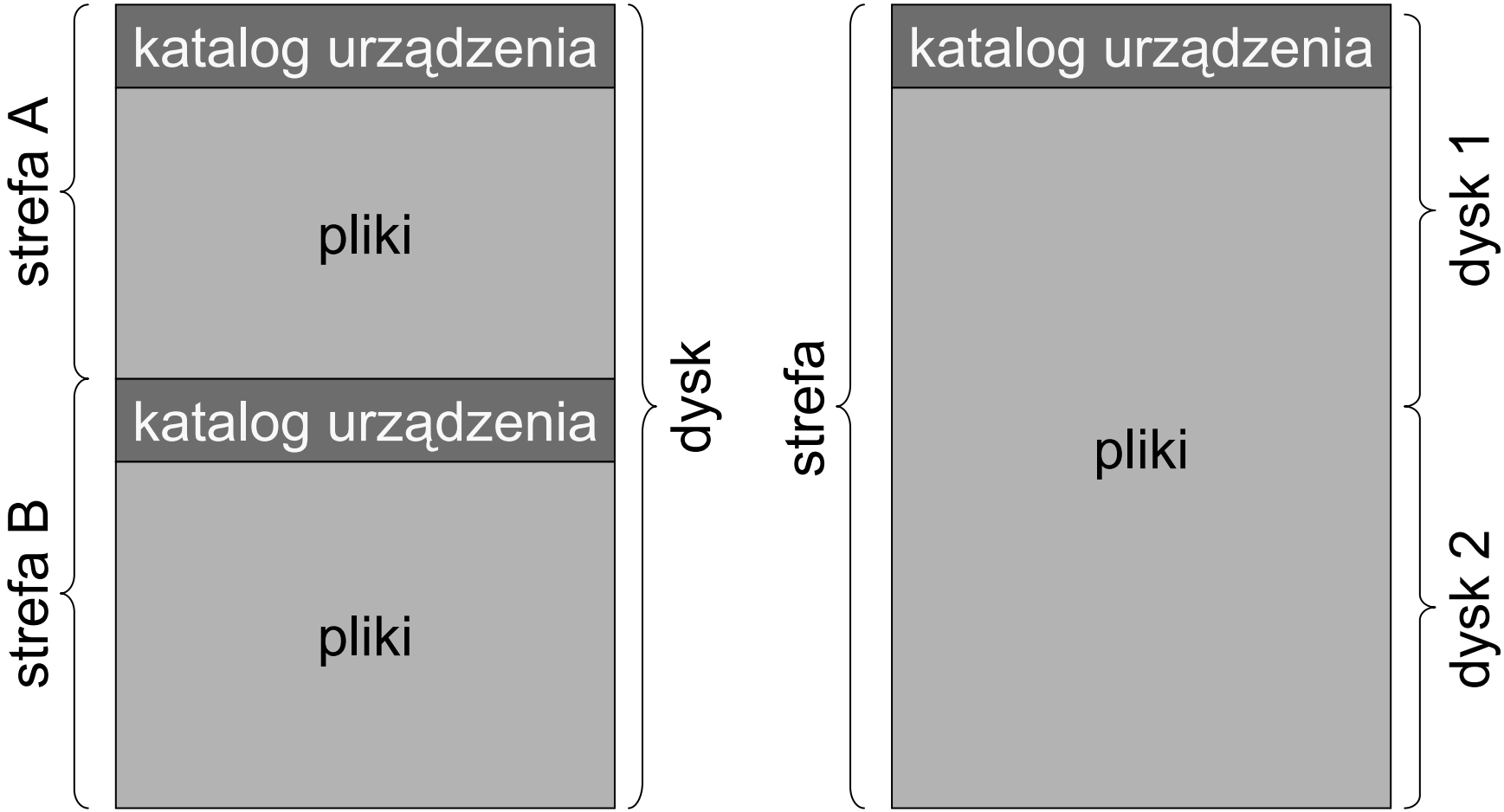
Organizacja systemu plików (1)

- Podział na strefy (wolumeny, woluminy, tomy, partycje)
  - strefa obejmuje część dysku, jeden dysk lub kilka dysków
  - strefa zawiera pliki i katalogi
- Organizacja katalogów:
  - katalog jest tablicą kojarzącą nazwy plików z wpisami katalogowymi, obejmującymi inne atrybuty plików,
  - katalogi mogą być jednopoziomowe lub wielopoziomowe ,
  - katalogi wielopoziomowe zorganizowane mogą być w różne struktury logiczne (drzewo, graf acykliczny, dowolny graf),
- Pliki identyfikowane są przez nazwy, znajdujące się w katalogach.

Organizacja systemu plików (2)



Podział na strefy



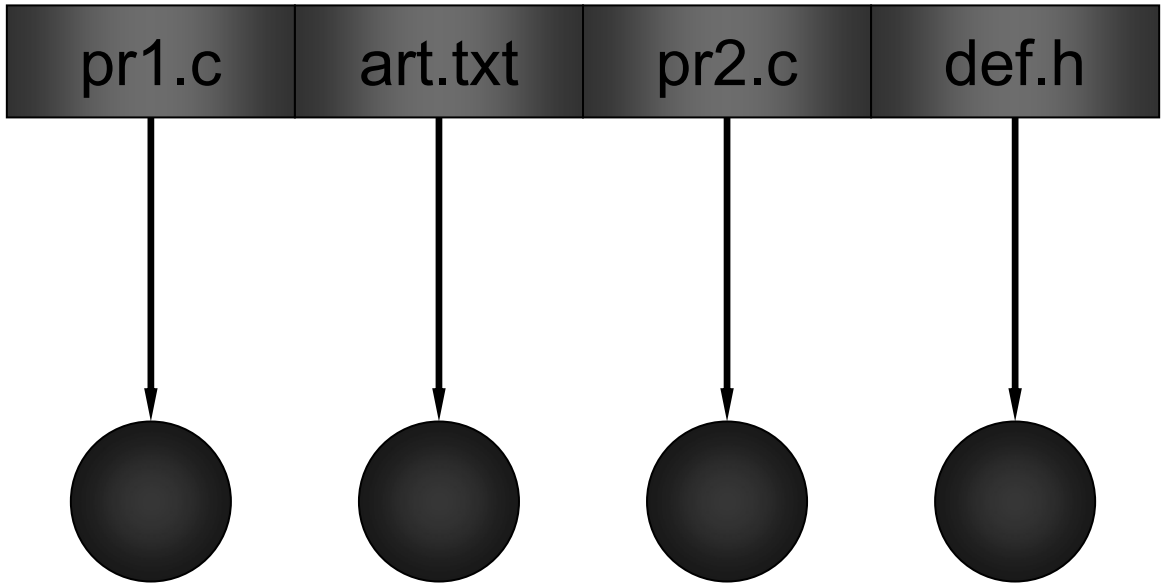
Operacje na katalogu

- Tworzenie wpisu katalogowego, gdy tworzony jest plik lub alternatywnej jego nazwy
- Usuwanie wpisu katalogowego
- Odnajdowanie wpisu katalogowego
- Tworzenie wykazu wpisów katalogowych (listing zawartości)
- Przemianowanie pliku (zmiana nazwy)

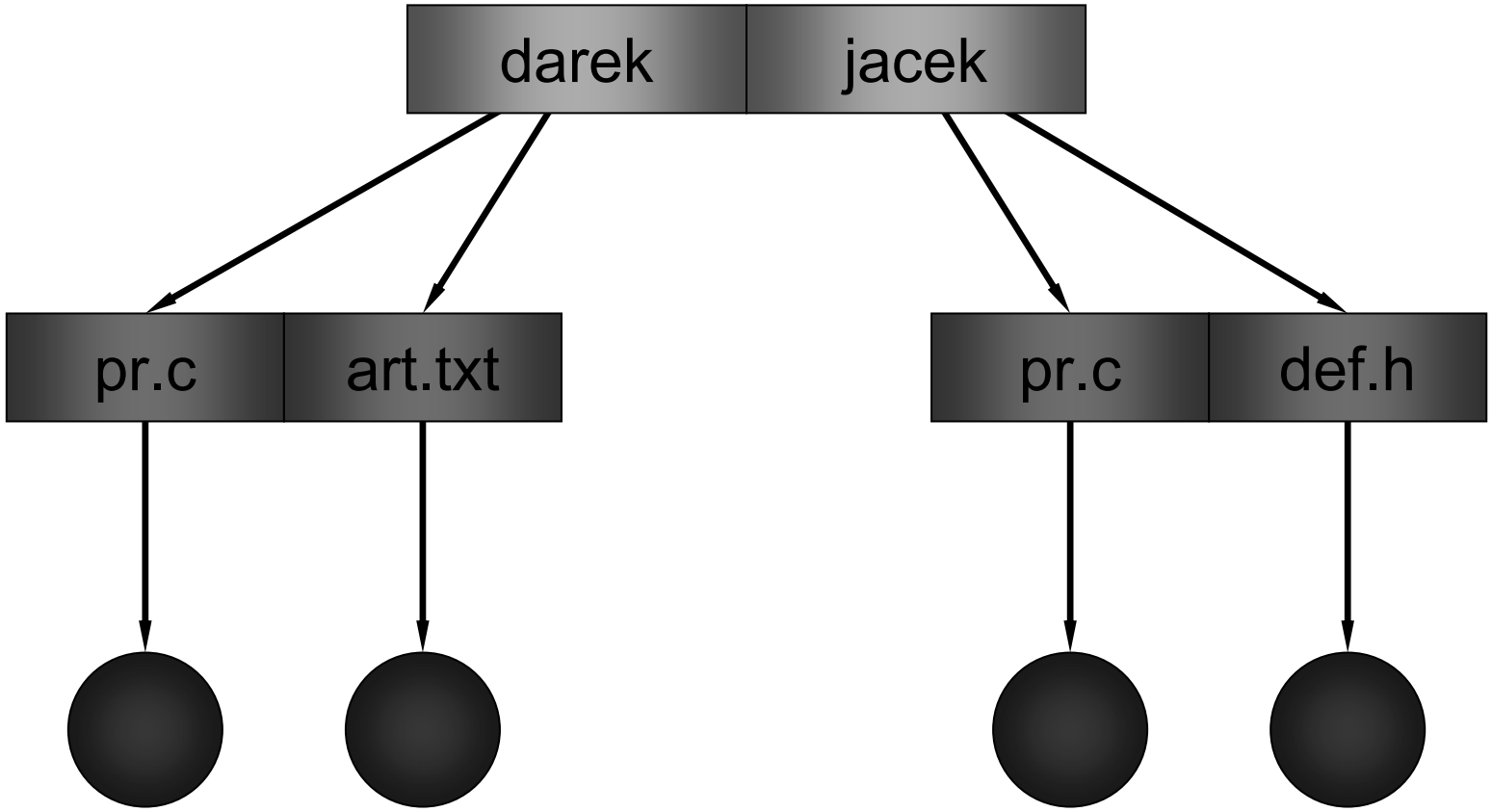
Struktura logiczna katalogów

- Struktura jednopoziomowa — wpisy katalogowe poszczególnych plików znajdują się w tym samym katalogu (na tym samym poziomie).
- Struktura dwupoziomowa — wpisy katalogowe plików znajdują się w różnych katalogach, ale katalogi nie mogą zawierać innych katalogów.
- Struktura drzewiasta — w katalogach można tworzyć podkatalogi (z dowolnym poziomem zagnieżdżenia) oraz pliki.
- Graf acykliczny — podkatalog (lub plik) może być umieszczony w wielu katalogach.
- Graf ogólny — dopuszcza się cykl w powiązaniach pomiędzy katalogami

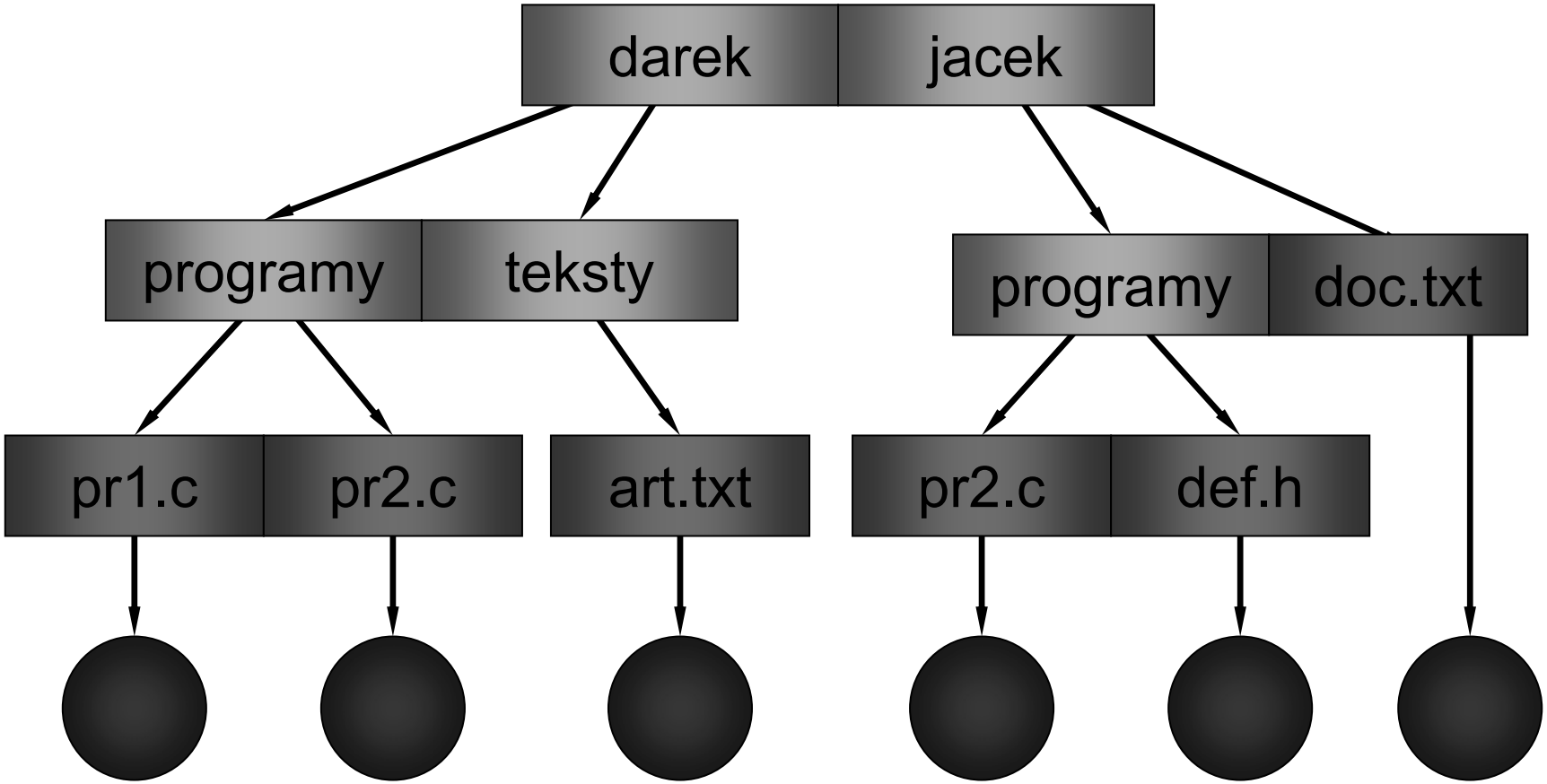
Struktura jednopoziomowa



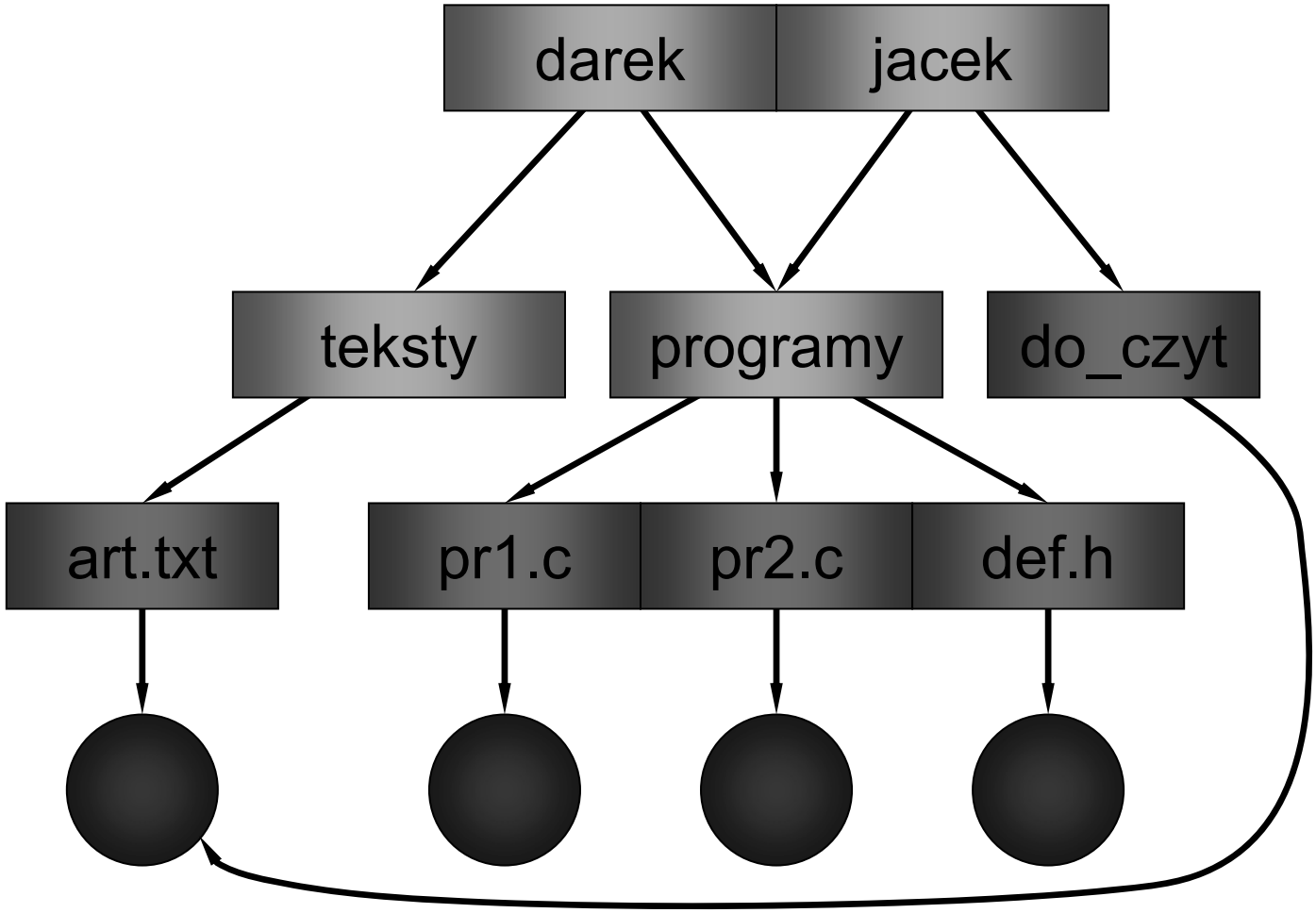
Struktura dwupoziomowa



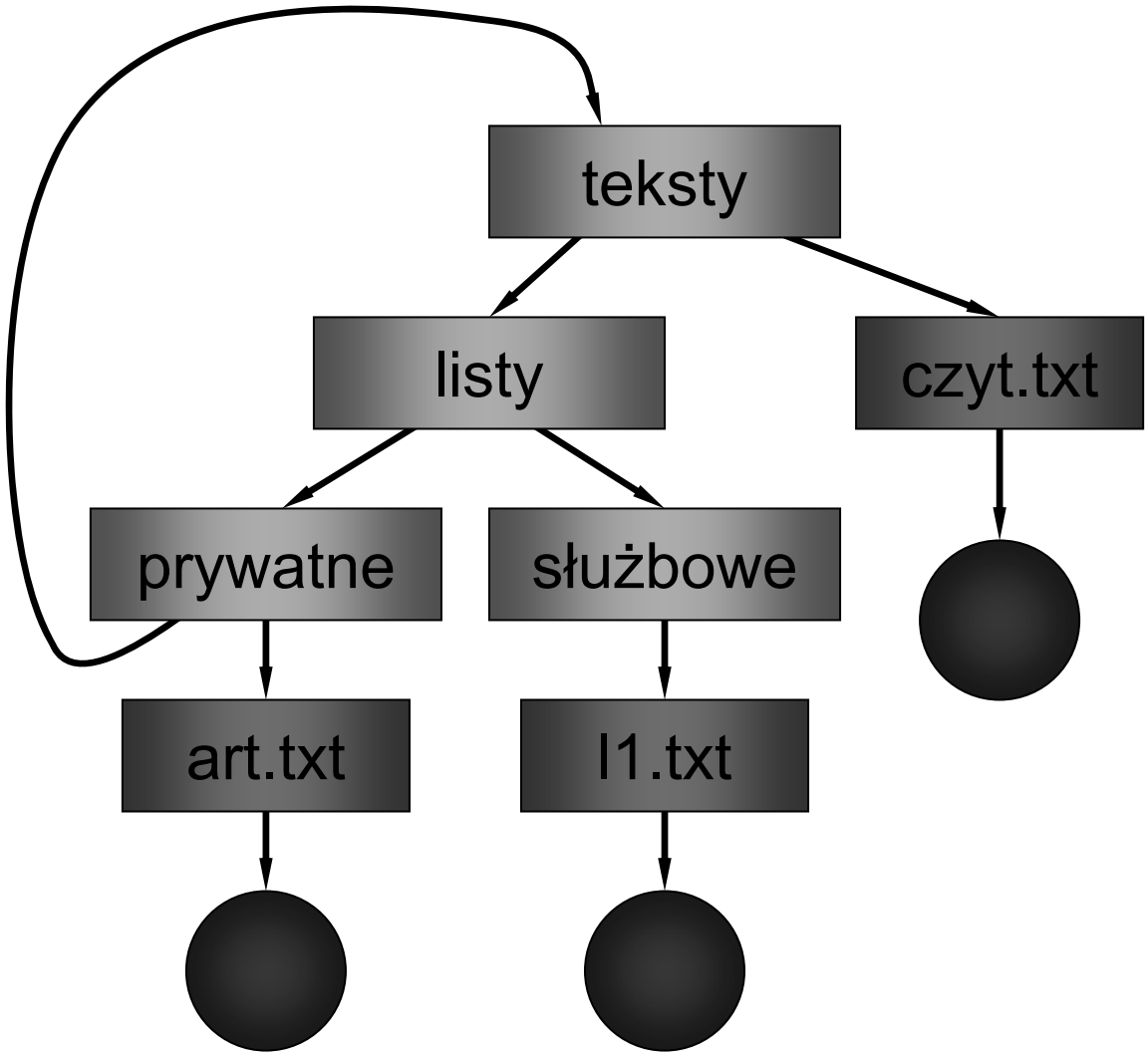
Struktura drzewiasta



Graf acykliczny



Graf ogólny



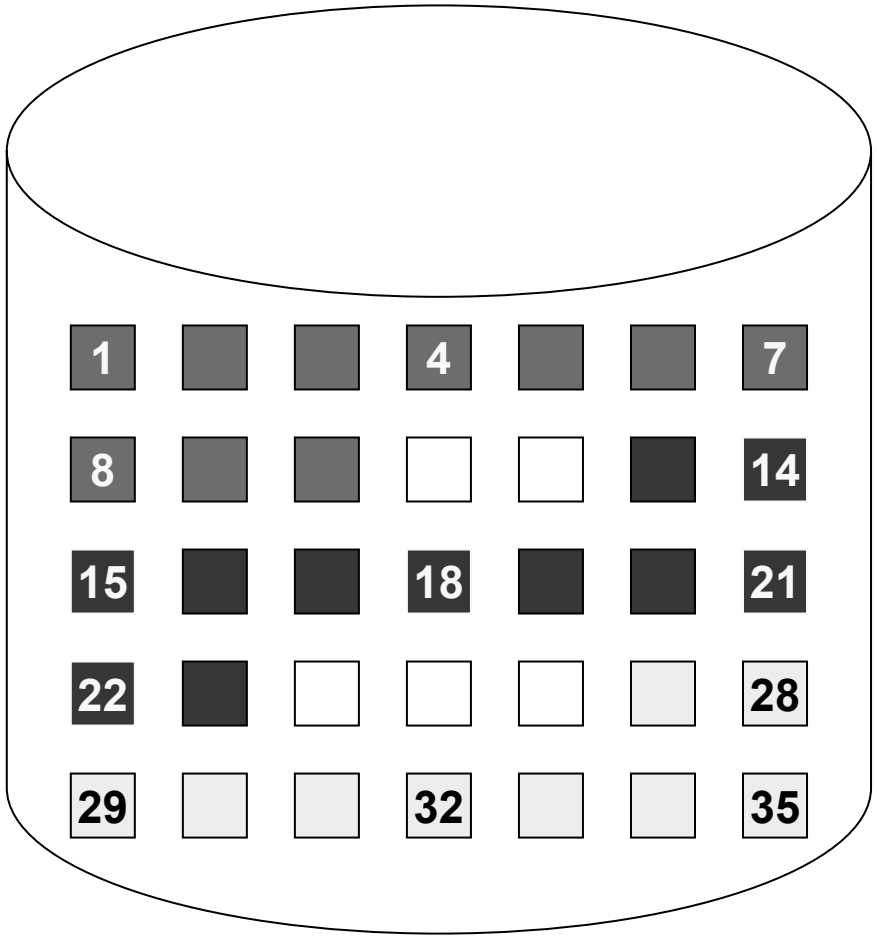
Implementacja systemu plików

- Przydział miejsca na dysku
  - przydział ciągły
  - przydział listowy
  - przydział indeksowy
- Zarządzanie wolną przestrzenią
  - wektor bitowy
  - lista powiązana
  - grupowanie
  - zliczanie
- Implementacja katalogu
  - lista liniowa
  - tablica haszowa

Przydział miejsca na dysku

- Przydział ciągły (ang. contiguous allocation) — cały plik zajmuje ciąg kolejnych bloków (jednostek alokacji)
- Przydział listowy (ang. linked allocation) — plik jest listą powiązanych bloków, dowolnie rozproszonych w dostępnej przestrzeni dyskowej
- Przydział indeksowy (ang. indexed allocation) — wskaźniki do rozproszonych bloków dyskowych (indeksy) skupione są w jednym miejscu, w tzw. bloku indeksowym
  - schemat listowy bloku indeksowego
  - indeks wielopoziomowy
  - schemat kombinowany

Przydział ciągły



Katalog:

blok początkowy: 1
rozmiar: 10 bloków

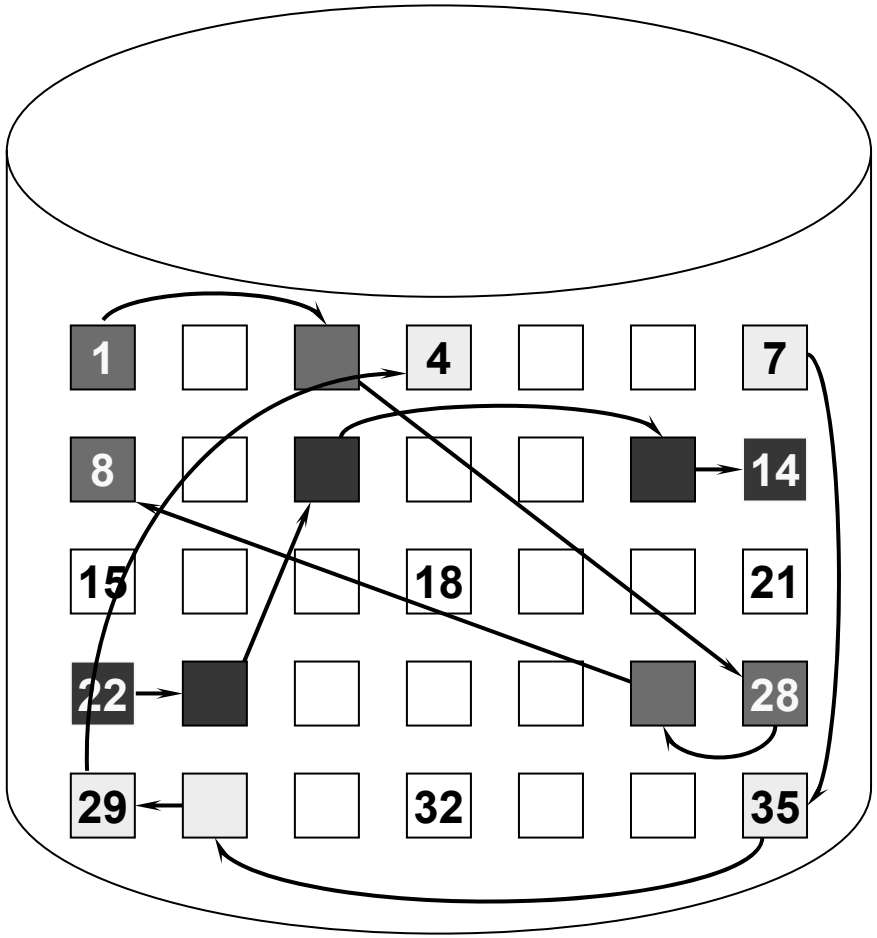
blok początkowy: 13
rozmiar: 11 bloków

blok początkowy: 27
rozmiar: 9 bloków

Przydział ciągły — właściwości

- Efektywność dostępu (niewielkie ruchy głowic dysk.)
- Łatwa lokalizacja bloków pliku zarówno przy dostępie sekwencyjnym jak i swobodnym
- Problem fragmentacji zewnętrznej — po usuniętych plikach pozostają dziury, które trudno połączyć w jeden większy blok
- Problem rozszerzania pliku
  - pliku nie da się rozszerzyć,
  - będzie go trzeba przenieść w nowe miejsce (znaleźć większą dziurę)
  - będzie trzeba z góry zarezerwować więcej miejsca w przestrzeni dyskowej

Przydział listowy



Katalog:

blok początkowy: 1
blok końcowy: 8

blok początkowy: 22
blok końcowy: 14

blok początkowy: 7
blok końcowy: 4

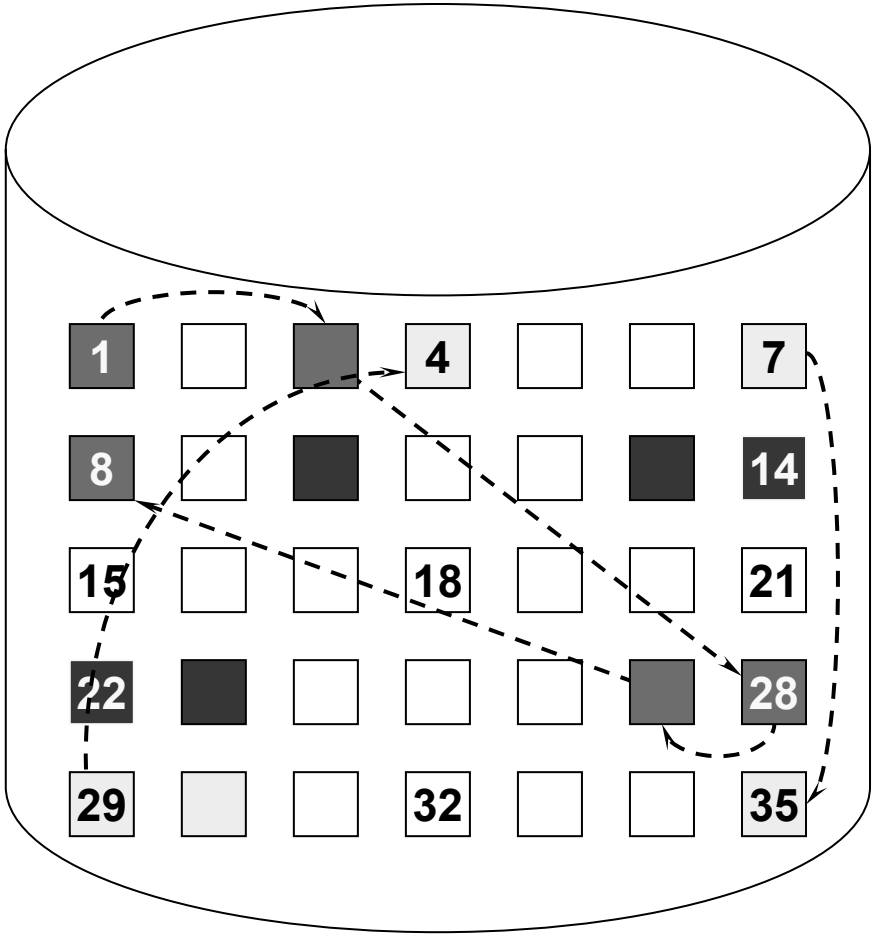
Przydział listowy — właściwości

- Nie ma problemu fragmentacji zewnętrznej i wynikających z niej wad metody przydziału
- Łatwa realizacja dostępu sekwencyjnego
- Problem realizacji dostępu swobodnego (bezpośredniego) — przejście do poprzedniego bloku wymaga rozpoczęcia przeglądania listy bloków od początku
- Konieczność pamiętania wewnątrz bloku wskaźnika do bloku następnego
- Niezawodność — utrata jednego bloku pociąga za sobą stratę wszystkich następnych

Tablica alokacji plików — FAT (File Allocation Table)

- FAT jest dodatkową strukturą (tablicą) umieszczoną w odpowiednim obszarze na dysku
- Każdy element tablicy FAT odpowiada dokładnie jednej jednostce alokacji (blokowi) z przestrzeni bloków plikowych i indeksowany jest numerem bloku
- Element tablicy FAT zawiera indeks następnego bloku przydzielonego danemu plikowi lub pewną wartość specjalną oznaczającą wolną pozycję lub ostatnią pozycję danego pliku

Struktury tablicy alokacji plików



1	3
2	
3	28
4	#
5	4
6	4
7	35
8	#
	...
27	8
28	27

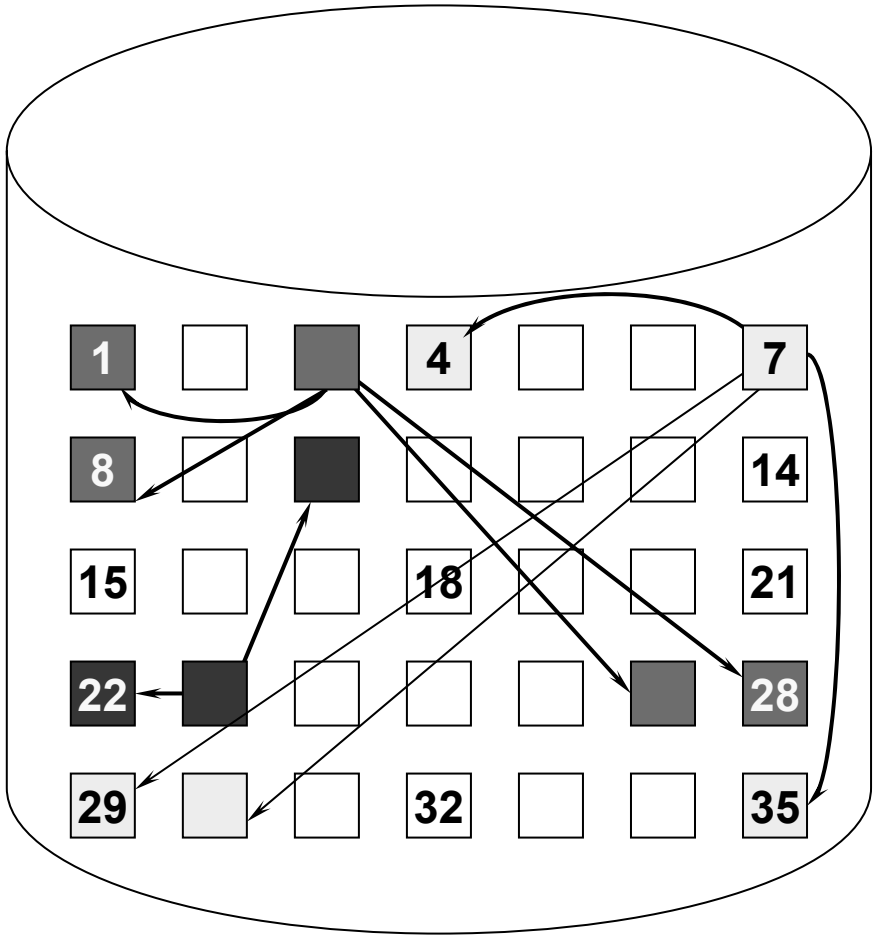
Katalog:

blok początkowy: 1
blok końcowy: 8

blok początkowy: 22
blok końcowy: 14

blok początkowy: 7
blok końcowy: 4

Przydział indeksowy



Katalog:

blok indeksowy: 3
rozmiar: 4 bloki

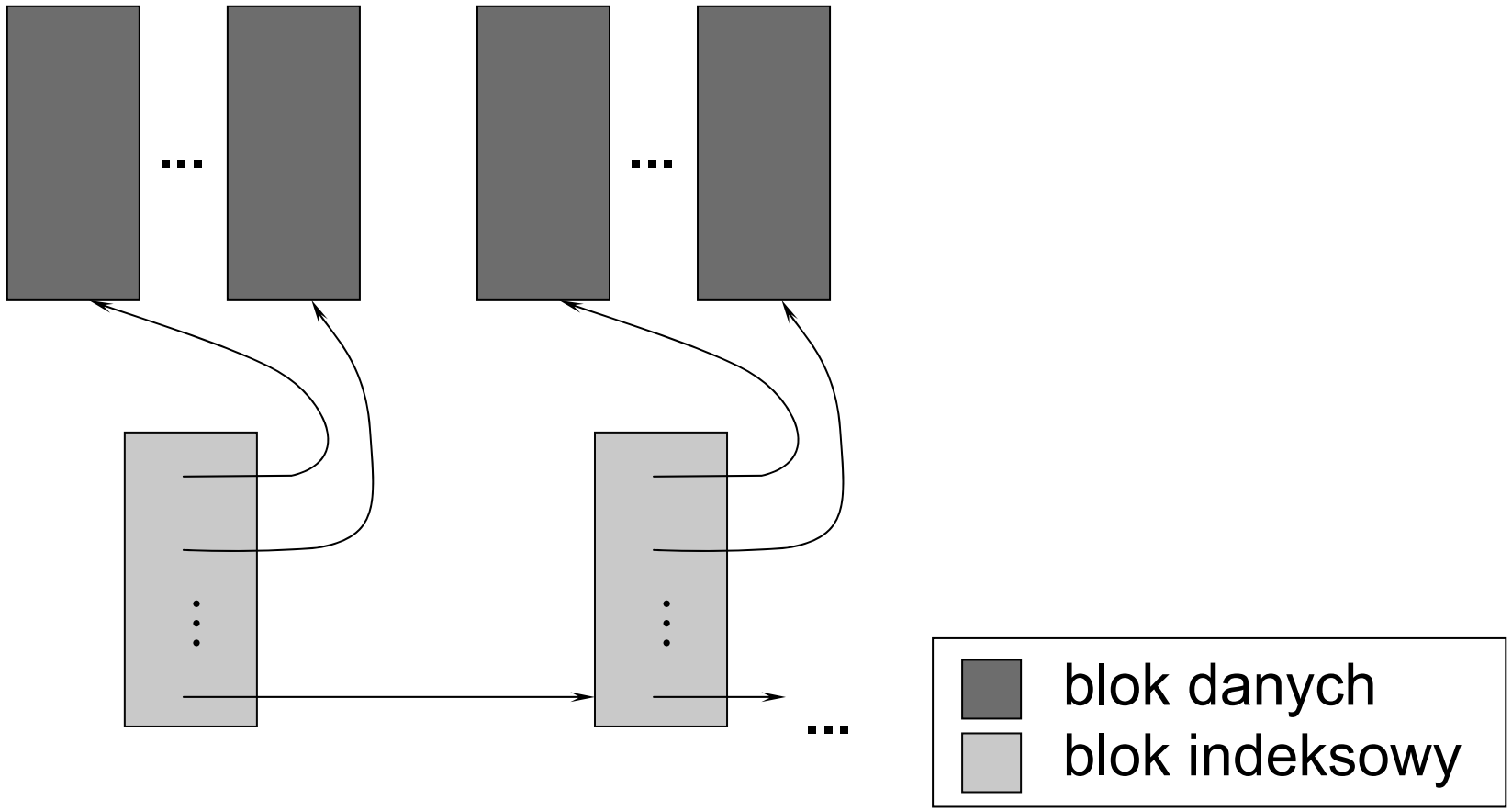
blok indeksowy: 23
rozmiar: 2 bloki

blok indeksowy: 7
rozmiar: 4 bloki

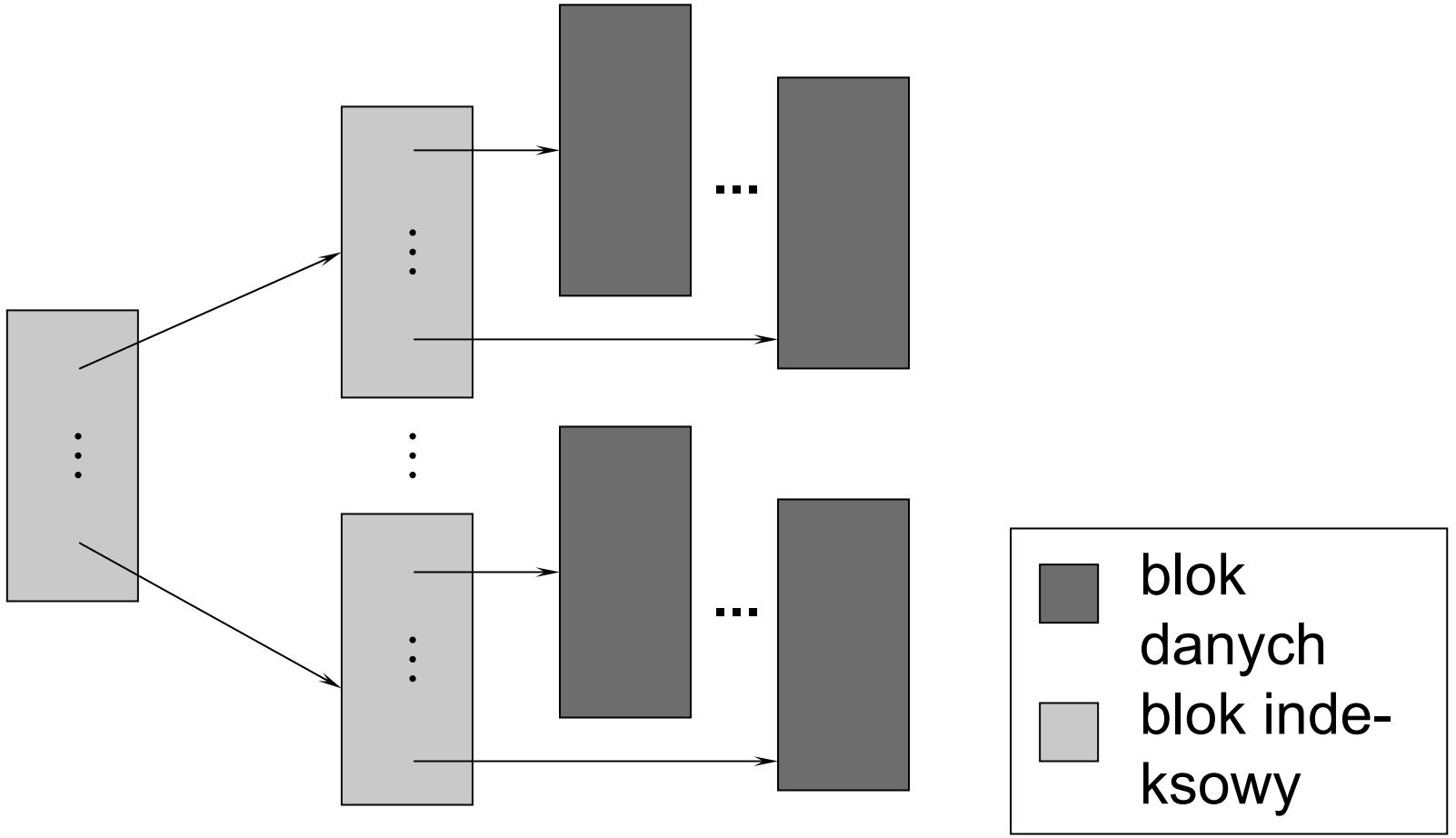
Struktura bloku indeksowego

- Schemat listowy — w ostatnim elemencie bloku indeksowego znajduje się wskaźnik do następnego bloku indeksowego tego pliku.
- Indeks wielopoziomowy — blok indeksowy pierwszego poziomu zawiera wskaźnik do bloków drugiego poziomu itd.
- Schemat kombinowany — zastosowanie do pewnej liczby bloków indeksu pierwszego poziomu, dla następnych bloków indeksu dwupoziomowego itp.

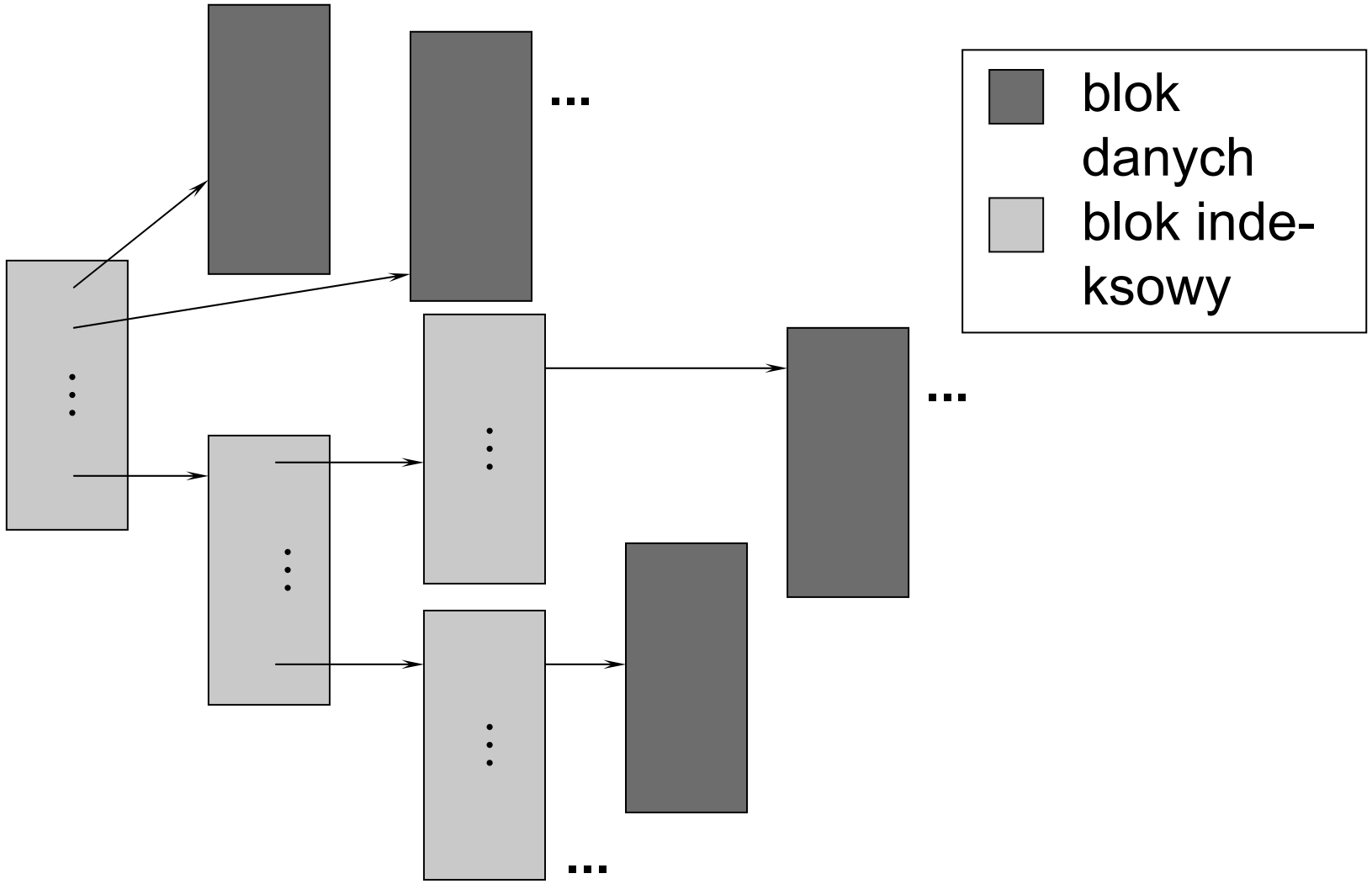
Struktura bloku indeksowego — schemat listowy



Struktura bloku indeksowego — indeks wielopoziomowy



Struktura bloku indeksowego — schemat kombinowany



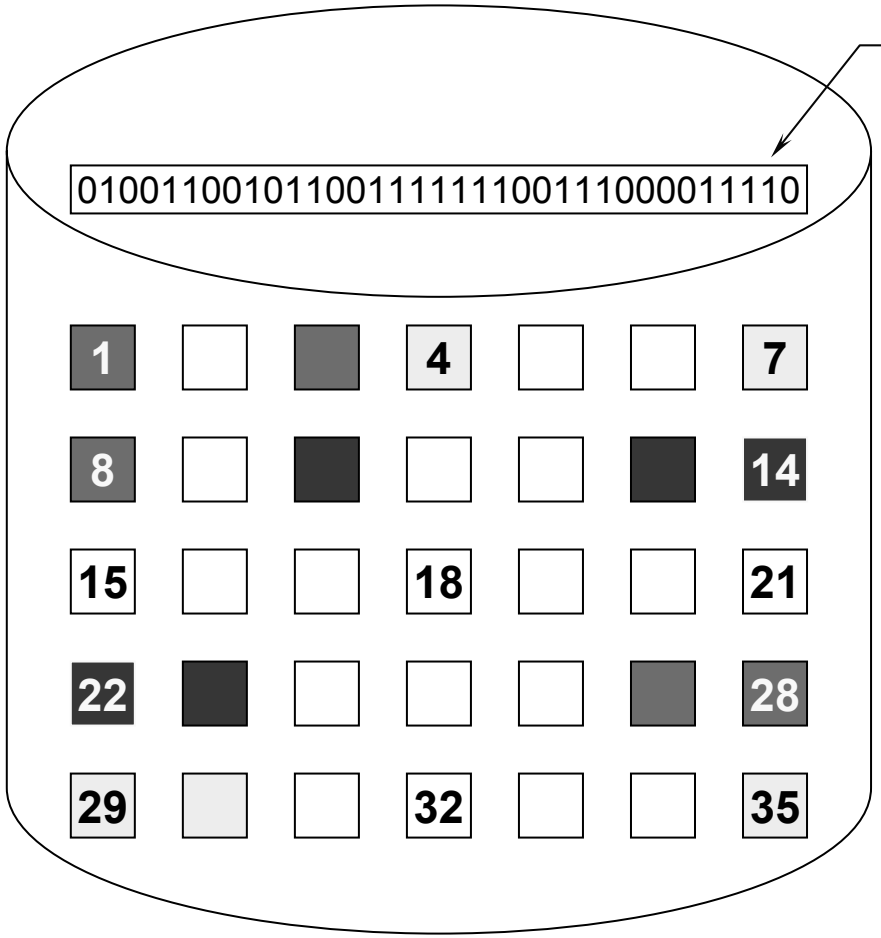
Przydział indeksowy — właściwości

- Łatwa lokalizacja bloków pliku zarówno przy dostępie sekwencyjnym jak i swobodnym
- Łatwa realizacja dostępu swobodnego (bezpośredniego)
- Brak problemu fragmentacji zewnętrznej
- Konieczność przeznaczenie pewnej części przestrzeni dyskowej na bloki indeksowe

Zarządzanie wolną przestrzenią

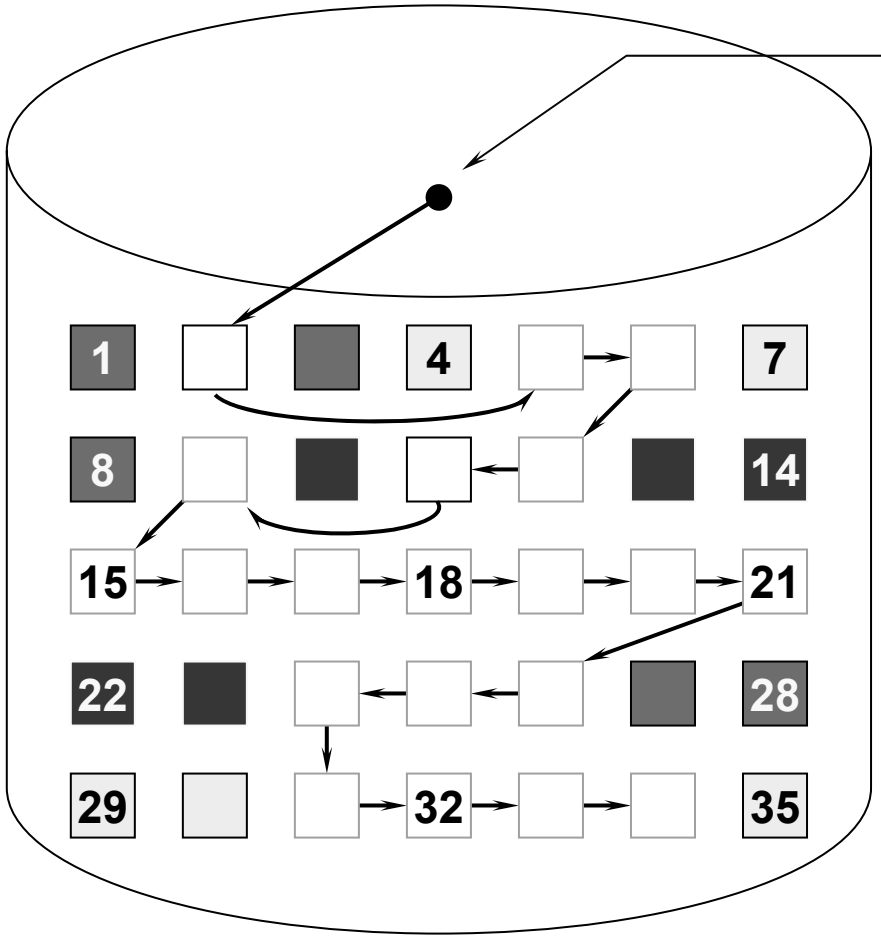
- Wektor bitowy — każdy bit odpowiada jednemu blokowi, wartość 1 oznacza wolny blok.
- Lista powiązana — każdy wolny blok zawiera indeks następnego wolnego bloku.
- Grupowanie — niektóre wolne bloki zapełnione są w całości indeksami innych wolnych bloków, ostatni indeks wskazuje na kolejny blok zapełniony w całości indeksami.
- Zliczanie — wykaz wolnych bloków obejmuje indeks pierwszego wolnego bloku oraz liczbę wolnych bloków znajdujących się za nim, stanowiących ciągły obszar.

Zarządzanie wolną przestrzenią — wektor bitowy



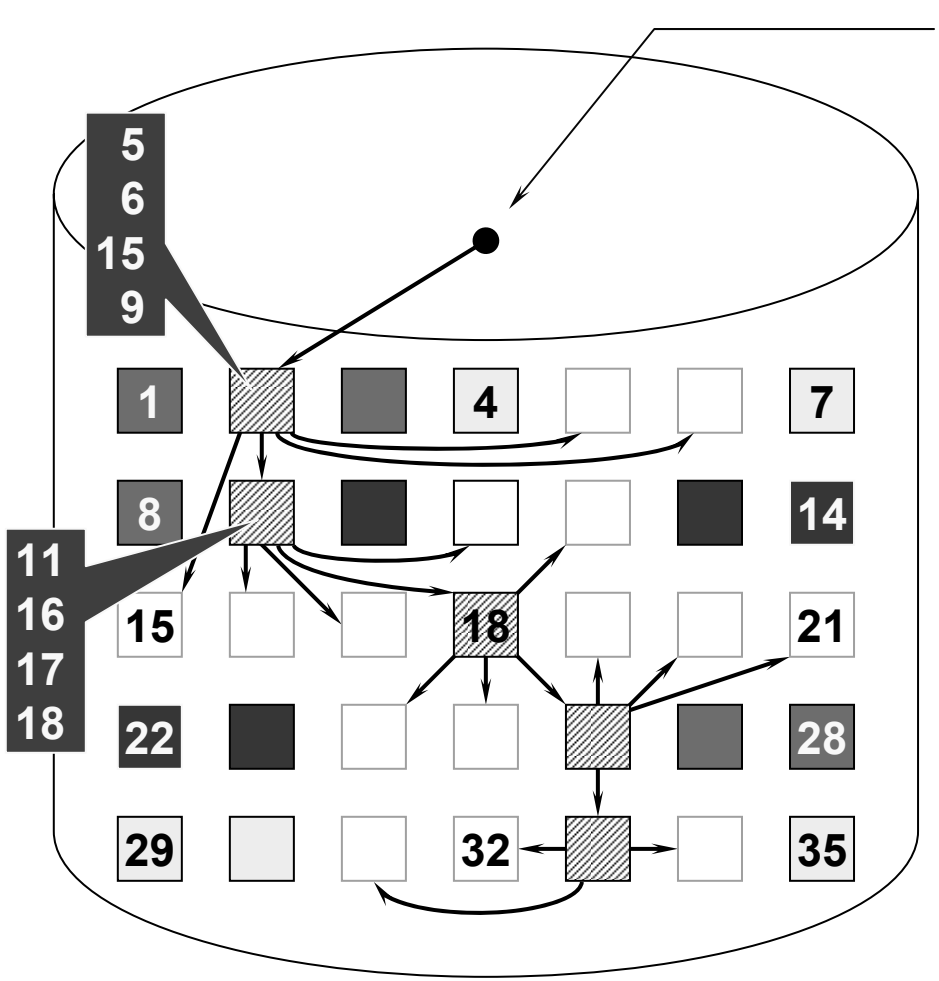
Każdy blok dyskowy (jednostka alokacji) reprezentowany jest przez jeden bit w wektorze. Wartość 1 tego bitu oznacza, że dany plik jest wolny.

Zarządzanie wolną przestrzenią — lista powiązana



Powiązanie wszystkich wolnych bloków w ten sposób, że w bloku poprzednim znajduje się indeks bloku następnego, a indeks pierwszego bloku znajduje się w specjalnym miejscu w systemie plików

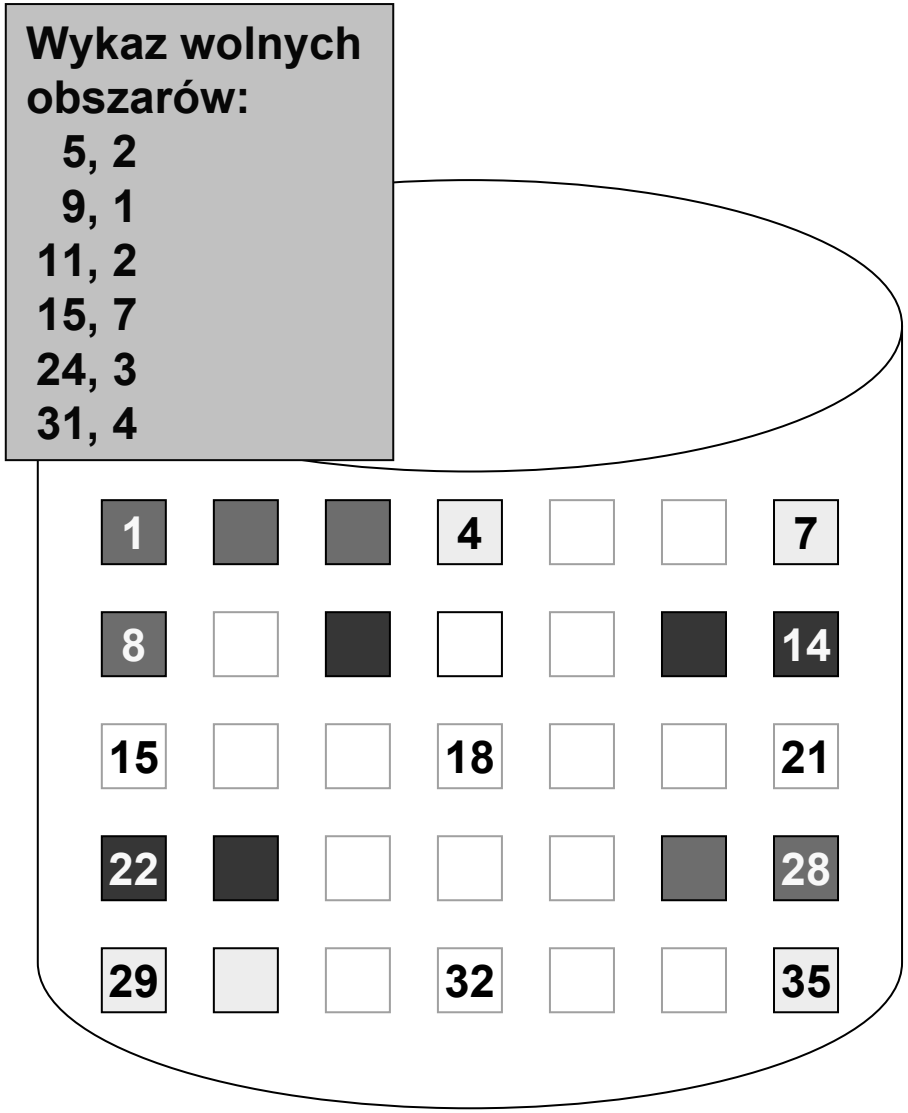
Zarządzanie wolną przestrzenią — grupowanie



indeks bloku pierwszej grupy wolnych bloków

Pierwszy wolny blok zawiera indeksy *n* innych wolnych bloków, z których *n*-1 dotyczy bloków do alokacji przez pliki, a *n*-ty blok zawiera znowu *n*-1 indeksów kolejnych wolnych bloków

Zarządzanie wolną przestrzenią — zliczanie



W przypadku kilku kolejnych (przylegających do siebie) wolnych bloków pamiętany jest tylko indeks pierwszego z nich oraz liczba wolnych bloków znajdujących się bezpośrednio za nim. Wykaz wolnych obszarów jest ciągiem wpisów, składających się z indeksu bloku oraz licznika.

Implementacja katalogu

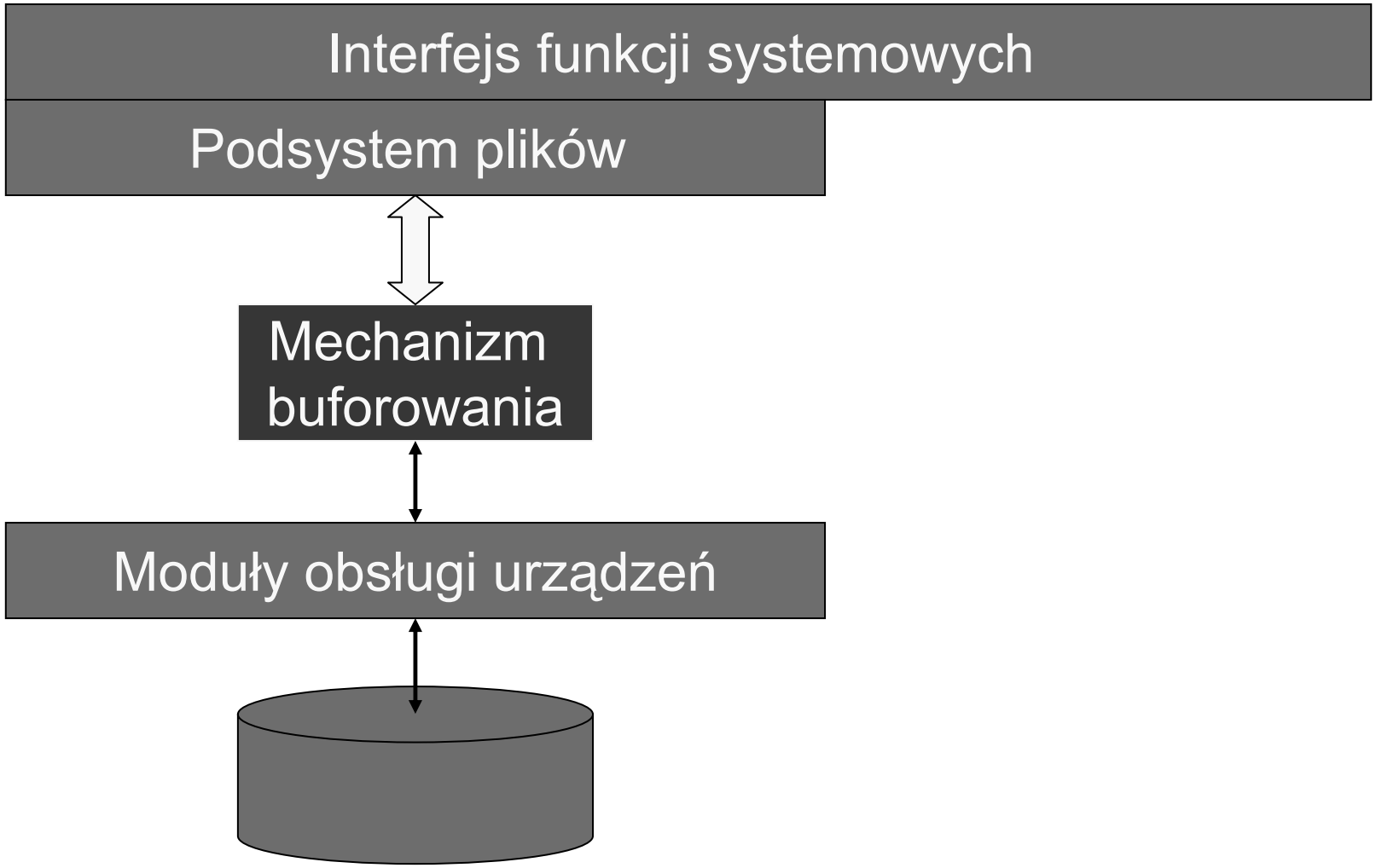
- Lista liniowa — katalog składa się z ciągu wpisów katalogowych ogólnej postaci:

  - lokalizacja wpisu polega na przeszukiwaniu liniowym (sprawdzane są kolejne pozycje, począwszy od pierwszej)
- Tablica haszowa — do lokalizacji wpisu wykorzystywana jest funkcja haszująca, która odwzorowuje nazwę pliku na wartość z określonego przedziału, traktowaną jako indeks wpisu. W katalogu mogą być potrzebne dodatkowe struktury pomocne przy usuwaniu konfliktów.

Buforowanie w systemie plików

- Operacja dostępu do danych w pliku wymaga ich sprowadzenia do pamięci operacyjnej, gdzie można je testować, zmieniać, po czym zażądać ponownego ich zapisania do systemu plików.
- Czytanie i pisanie bezpośrednio z / na dysk podczas wszystkich operacji dostępu do plików, jest nieefektywne ze względu na czas reakcji dysku oraz relatywnie małą szybkość transmisji dyskowych.
- Minimalizacja dostępów do dysku możliwa jest przez utrzymywanie puli wewnętrznych buforów, zwanych podręczną pamięcią buforową (ang. buffer cache), które zawierają dane z ostatnio używanych bloków dyskowych.

Miejsce mechanizmu buforowania w strukturze systemu plików



Ogólne zasady buforowania

- Bloki dyskowe aktualnie wykorzystywane są utrzymywane w pamięci *buffer cache* — dane w jednym buforze odpowiadają danym z jednego bloku dyskowego.
- Obsługa żądania odczytu bloku polega najpierw na sprawdzeniu czy dany blok znajduje się w pamięci *buffer cache*.
- Jeśli blok nie znajduje się w pamięci *buffer cache* jest czytany z dysku do tej pamięci, a następnie kopiowany w odpowiednie miejsce w przestrzeni adresowej procesu.
- Dane zapisywane na dysk są również zapamiętywane w pamięci buforowej, by były tam dostępne dla ewentualnych kolejnych operacji odczytu.

Buforowanie w realizacji operacji odczytu

1. Znajdź adres bloku dyskowego, zawierającego fragment pliku, którego odczytu zażądano.
2. Przekopiuj zawartość tego bloku do bufora systemu plików w pamięci głównej (jeśli ten blok się tam jeszcze nie znajduje).
3. Przekopiuj żądany fragment z bufora do przestrzeni adresowej procesu.

Buforowanie w realizacji operacji zapisu

1. Znajdź adres bloku dyskowego, zawierającego fragment pliku, którego zapisu zażądano.
2. Przekopiuj zawartość tego bloku do bufora w pamięci głównej (jeśli ten blok się tam jeszcze nie znajduje);
3. Przekopiuj żądany fragment z przestrzeni adresowej procesu do bufora.
4. Zapisz na dysk uaktualniony blok z bufora (albo w trybie natychmiastowym albo z opóźnieniem)

Integralność systemu plików

- W wyniku awarii systemu (np. na skutek zaniku napięcia w sieci elektrycznej) zawartość podręcznej pamięci buforowej może nie zostać zapisana na dysku lub może zostać zapisana tylko częściowo.
- Skutkiem w/w awarii może być pozostawienie systemu plików w stanie niespójnym.
- Większość systemów operacyjnych dostarcza odpowiednie narzędzie do sprawdzania spójności systemu plików, uruchamiane w ramach restartu systemu po awarii.

Przejawy niespójności systemu plików

- Brak bloku zarówno w wykazie bloków zaalokowanych jak i bloków wolnych
- Obecność bloku zarówno w wykazie bloków zaalokowanych jak i bloków wolnych
- Wielokrotne powtórzenie się bloku w wykazie bloków wolnych (duplikacja wolnego bloku)
- Wielokrotne powtórzenie się bloku w wykazie bloków zaalokowanych (duplikacja zaalokowanego bloku)
- Niespójność informacji we wpisach katalogowych (np. niezgodność licznika dowiązań w systemie UNIX).

Semantyka spójności

- Semantyka spójności określa sposób postrzegania zmian zawartości pliku, dokonywanych przez współbieżnie działające (i korzystającego z danego pliku) procesy.
- Przykłady semantyki spójności:
  - semantyka spójności systemu UNIX — wynik operacji zapisu jest natychmiast widoczny dla innych procesów, które otwarły dany plik,
  - semantyka sesji — zmiany w pliku stają się widoczne tylko dla procesów, które ten plik otworzą po jego zamknięciu przez proces zapisujący,
  - semantyka stałych plików dzielonych — plik dzielony nie może podlegać modyfikacjom, czyli może być tylko czytany.

Synchronizacja dostępu do plików

- Współbieżny dostęp do zawartości pliku można synchronizować na poziomie całego pliku lub poszczególnych jego fragmentów (zajmowanie rekordów).
- Najczęściej dopuszcza się dwa rodzaje blokad
  - blokada współdzielona (shared lock, read lock)
  - blokada wyłączna (exclusive lock, write lock)

	shared	exclusive
shared	zgodne	wyklucz.
exclusive	wyklucz.	wyklucz.