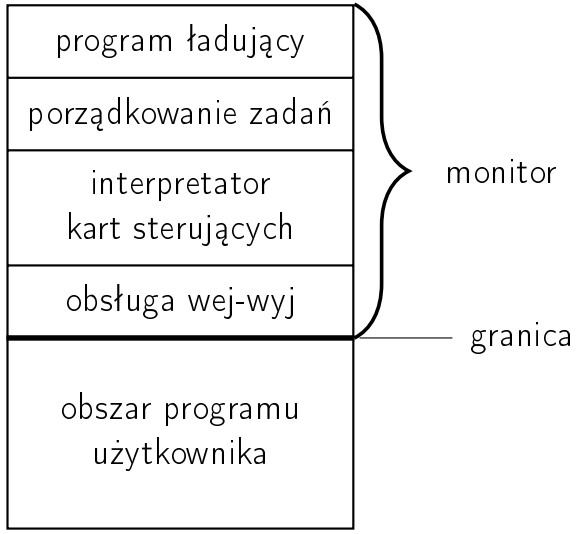
Jak system wykonuje zadania?

- Przetwarzanie wsadowe, monitor prosty, praca pośrednia
- Buforowanie, Spooling
- Wieloprogramowość
- Wielozadaniowość, wielodostępowość, systemy z podziałem czasu
- Wieloprocesorowość, systemy HPC
- Systemy rozproszone, klastrowe i chmurowe
- Systemy czasu rzeczywistego

Monitor prosty

- wczesne rozwiązania zarządzania zadaniami w systemach operacyjnych
- **przetwarzanie wsadowe** (*batch processing*) - zadania gromadzone w postaci wsadów są kolejno wykonywane, bez interakcji użytkownika
- **monitor rezydujący** – stale obecny w pamięci, automatyczne przekazuje sterowania od zadania do zadania, monitoruje i zarządza określonymi zasobami lub zdarzeniami
- automatyczne porządkowanie zadań - przydzielanie czasu procesora poszczególnym zadaniom w sposób sekwencyjny lub według prostych reguł
- każde zadanie musiało zakończyć się lub zrezygnować z procesora, zanim kolejne mogło zostać uruchomione
- w najprostszych monitorach brak mechanizmu przerwań sprzętowych - programista musiał jawnie wywoływać funkcje monitora, aby oddać sterowanie

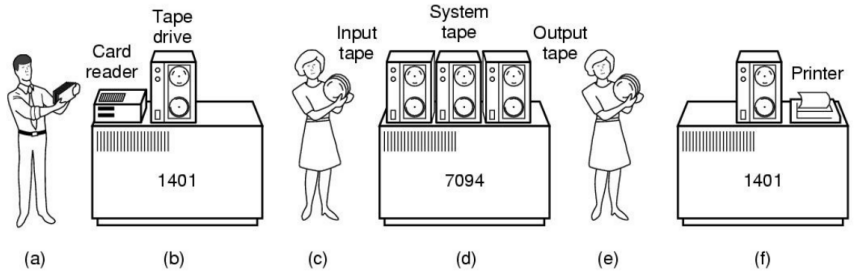
Monitor rezydujący w pamięci



- **ochrona pamięci** - program nie może ingerować w obszar pamięci monitora
- **uprzywilejowane instrukcje** - pewne instrukcje moga być wykonywane wyłącznie przez monitor (tryb jądra)

Praca pośrednia (off-line)

- **Praca pośrednia** - przetwarzanie danych, gdzie urządzenia wejścia/wyjścia (np. czytniki kart, drukarki, taśmy magnetyczne) nie są bezpośrednio sterowane przez procesor komputera, lecz przez odrębne urządzenia lub programy pośredniczące
- dane są najpierw zbierane lub przygotowywane przez mniejsze, specjalizowane urządzenia, a następnie przekazywane do komputera w blokach (np. na taśmie magnetycznej)
- procesor nie traci czasu na oczekiwanie na wolne urządzenia we/wy



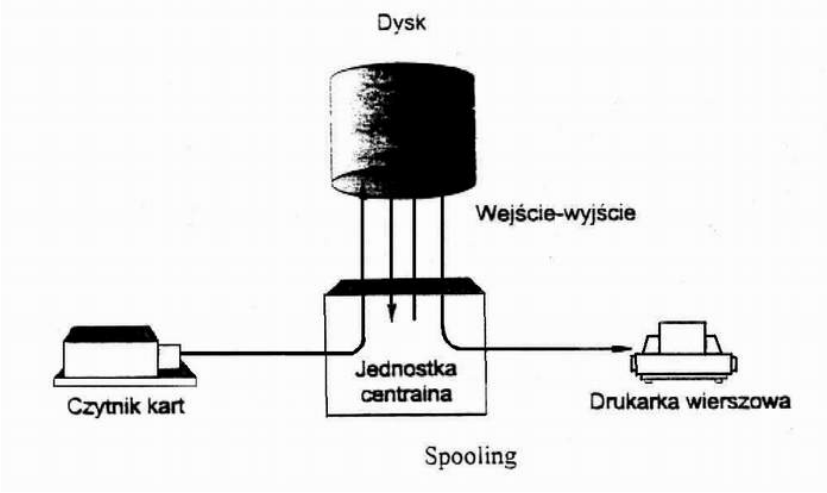
Buforowanie

Buforowanie jest metodą usprawniającą wykonywanie obliczeń i operacji wej-wyj dla danego zadania.

- polega na tymczasowym przechowywaniu danych w pamięci (buforze) przed ich dalszym przetwarzaniem lub przesyłaniem
- buforowanie zmniejsza liczbę operacji I/O, przyspiesza dostęp do danych
- procesor może pracować nad innymi zadaniami, podczas gdy dane są ładowane do bufora. To pozwala na lepsze wykorzystanie zasobów procesora i zmniejsza czas oczekiwania na dane
- przykłady: pamięć podręczna procesora, bufor dyskowy, buforowanie w systemach plików, buforowanie w sieciach komputerowych (DNS cache)

Spooling

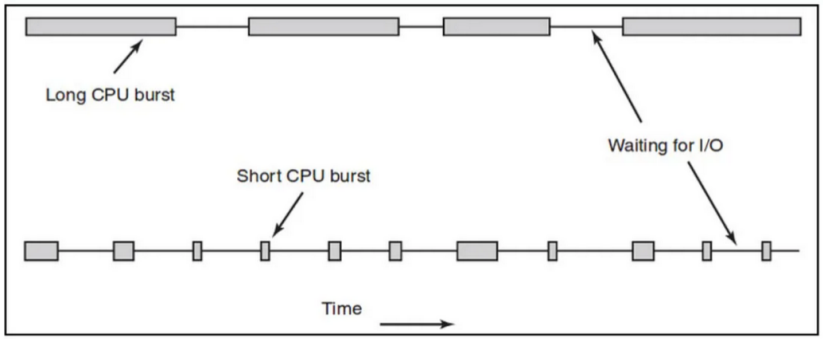
- **spooling** (*Simultaneous Peripheral Operation On-Line*) jednoczesna bezpośrednia praca urządzeń
- dane są przechowywane w plikach na dysku w celu ich późniejszego przetwarzania przez urządzenia peryferyjne (np. drukarki)
- spooling umożliwia równoczesne wykonywanie operacji wej-wyj jednego zadania i obliczeń dla innych zadań znajdujących się we wsadzie
- spooling wytwarza pulę zadań do wykonania i umożliwia planowanie zadań i wieloprogramowość



Spooling

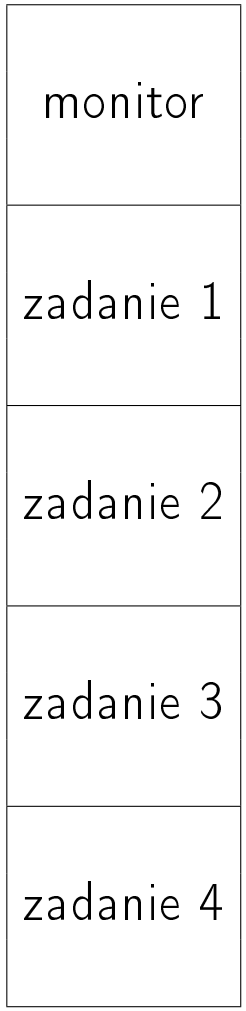
Zadania ograniczone CPU i ograniczone I/O

- Zadania ograniczone operacjami wej-wyj (*I/O bound*)
  - prędkość przetwarzania zależna od szybkości urządzeń wej-wyj.
  - buforowanie może zwiększyć przepustowość poprzez grupowanie wielu operacji I/O
- Zadania ograniczone procesorem (*CPU bound*)
  - prędkość przetwarzania ograniczona przez szybkość CPU (bufor wejściowy zawsze pełny, bufor wyjściowy zawsze pusty)
  - buforowanie zwiększa wydajność gromadząc często używane dane i zmniejszając czas dostępu



Wieloprogramowość

- **wieloprogramowość** - w pamięci rezyduje więcej niż jeden program użytkownika, ale tylko jeden jest wykonywany w danym momencie przez procesor
- w momencie zablokowania (np. operacja I/O) procesor jest przełaczany do innego zadania
- celem jest maksymalizacja wykorzystania CPU



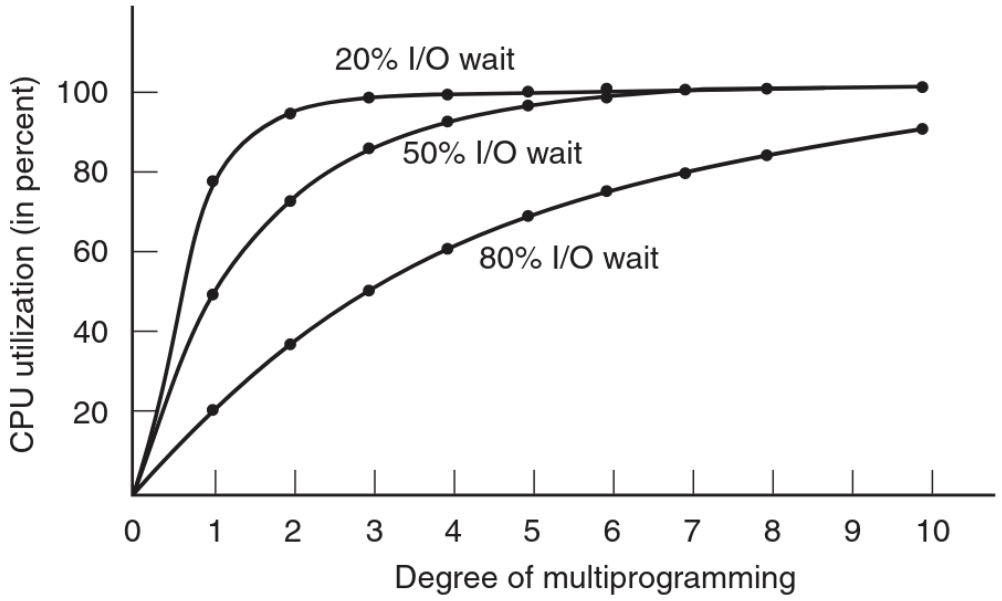
Realizacja wieloprogramowości wymaga skomplikowanego systemu operacyjnego:

- ochrona pamięci
- planowanie przydziału procesora, kolejka zadań
- zarzadzanie przydziałem pamięci
- ochrona zasobów

Wieloprogramowość: wykorzystanie CPU

Prosty model wykorzystania CPU:

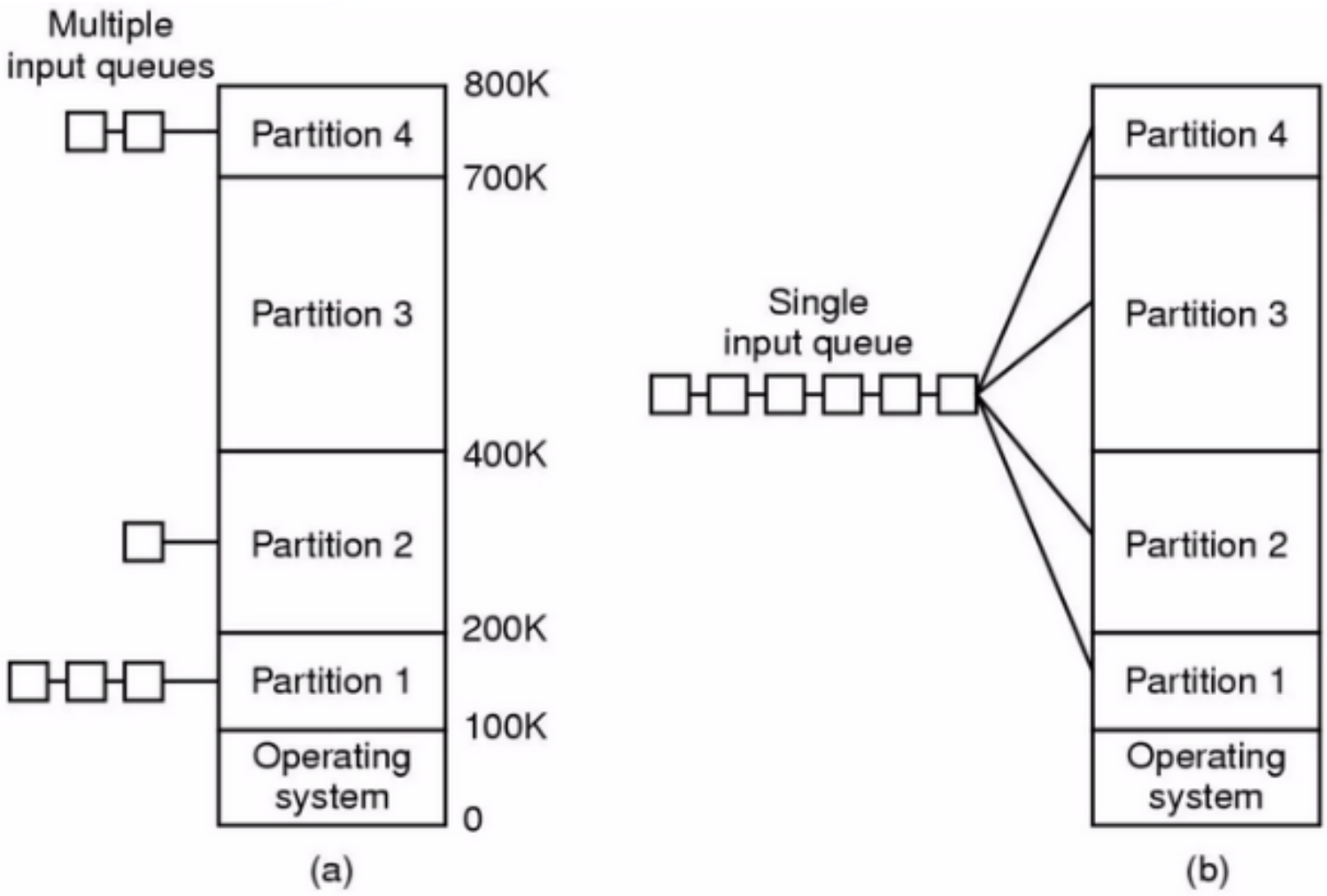
- n-procesów, każdy spędza część swego czasu w stanie I/O
- p - prawdopodobieństwo, że pojedynczy proces w danym przedziale czasowym nie korzysta z procesora (oczekuje na I/O)
- p^n – prawdopodobieństwo, że n-procesów będzie w stanie I/O
- wykorzystanie CPU $= 1 - p^n$



Wieloprogramowość: wykorzystanie pamięci

- podział pamięci:
  - partycje o **stałym** rozmiarze (jednakowej lub różnej wielkości)
  - **dynamiczny** - obszary pamięci dopasowane do rozmiaru zadania
- kolejka zadań oczekujących na przydział pamięci:
  - jedna wspólna kolejka dla wszystkich partycji
  - osobne kolejki dla każdej partycji
- algorytm wyboru zadania z kolejki, np. FIFO
- problem fragmentacji pamięci:
  - **wewnętrzna** - niewykorzystana pamięć wewnątrz przydzielonego bloku, gdy blok pamięci większy od rozmiaru zadania
  - **zewnętrzna** - niewykorzystane obszary pomiędzy przydzielonymi blokami, wolna pamięć jest rozproszona w wielu małych, nieciągłych obszarach

Wieloprogramowość: wykorzystanie pamięci



Wieloprogramowość ze stałym podziałem pamięci

a.) Osobne kolejki zadań do każdej partycji

- zadanie trafia do kolejki do najmniejszej pasującej partycji
- duże partycje moga pozostawać puste, nawet gdy są zadania w innych kolejkach, które mogłyby się zmieścić
- małe zadania muszą czekać

b.) Wspólna kolejka do wszystkich segmentów:

- zadanie z początku kolejki trafia do pierwszej wolnej partycji
- może powodować marnowanie pamięci na małe zadania
- potrzebny alg. wyszukiwania zadan w kolejce pasujących do wolnego obszaru - ale to dyskryminuje małe zadania

Wieloprogramowość z dynamicznym podziałem pamięci

- obszary tworzone dynamicznie dopasowane do rozmiaru zadania
- brak fragmentacji wewnętrznej - efektywne wykorzystanie pamięci RAM
- konieczność wykonywania upakowań (defragmentacji) w celu uniknięcia fragmentacji zewnętrznej (niewykorzystane obszary pomiędzy zaalokowanymi blokami), co zwiększa użycie CPU

Obecnie stosowane rozwiązanie: pamięć wirtualna, segmentacja o stałym rozmiarze i mechanizm stronicowania

Wielozadaniowość

- praca wsadowa a praca interakcyjna
- **wielozadaniowość** (*multitasking*) - wykonywanie wielu procesów w określonym przedziale czasu. Użytkownik ma wrażenie, że programy działają jednocześnie:
  - **współbieżna** na jednym procesorze przez szybkie przełączanie zadań (podział czasu)
  - **równoległa** na wielu procesorach
- **wielodostępowość** - wielu użytkowników korzysta z systemu jednocześnie (np. terminale podłączone do jednego komputera)

Wielozadaniowość w systemach z podziałem czasu

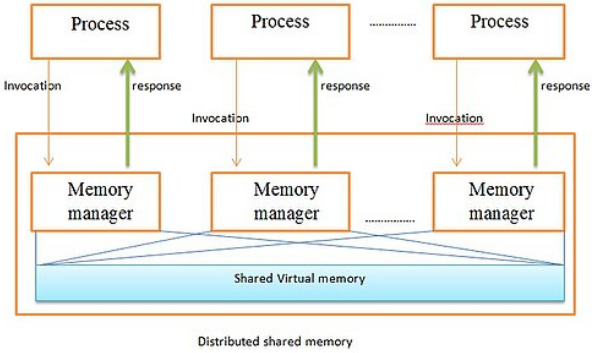
- **system z podziałem czasu** – system wielozadaniowy, w którym procesor jest przydzielany procesom w krotkich, równych porcjach czasu (kwantach), gdy kwant się kończy to CPU przedzielane innemu procesowi
- celem jest **interaktywność** - każdy użytkownik otrzymuje „wrażenie" ciągłej pracy systemu

Systemy z podziałem czasu są skomplikowane, gdyż wymagają:

- realizacji mechanizmów działań współbieżnych
- zarządzania pamięcią
- ochrony pamięci
- planowania przydziału CPU
- administrowania pamięcią dyskową
- systemu plików dostępnych bezpośrednio

Systemy HPC (*High Performance Computing*)

- **wieloprocesowość** (*multiprocessing*) - systemy posiadające więcej niż jeden procesor pozwalające na równoczesne przetwarzanie zadań
- **SMP** (*Symmetric MultiProcessing*)
  - współdzielona pamięć, I/O
  - wspólny system operacyjny
  - jednostki CPU są równoważne
- **DSM** (*Distributed Shared Memory*)
  - rozdzielone fizycznie pamięci są adresowane wspólną przestrzenią adresową
  - tańszy niż SMP i dobrze się skaluje
  - pozwala na obsługe dużych baz danych
  - wolniejsza komunikacja między węzłami niż SMP



Systemy HPC (*High Performance Computing*)

- **MPP** (*Massively Parallel Processor*)
  - setki/tysiące procesorów, rozproszona pamięć
  - kazdy CPU posiada własny system operacyjny, własną pamięć
  - podsystemy komunikują się łączami o dużej szybkości
- **klastry obliczeniowe**
  - zbiór węzłów pracujących wspólnie widocznych jako jedna jednostka obliczeniowa
  - połączenie przez sieć lokalną lub szybkie łącza
- **przetwarzanie sieciowe (grid)**
  - zbiór geograficznie rozproszonych zasobów komputerowych

Rygorystyczne systemy czasu rzeczywistego

- sterowniki urządzeń o ścisłe określonym zastosowaniu: nadzorowanie procesów produkcyjnych, eksperymentów naukowych, kierowanie sygnalizacją świetlną, autopilot, itp.
- działanie podlega ostrym rygorom czasowym, nawet kosztem efektywnego wykorzystania systemu

Łagodne systemy czasu rzeczywistego

- działanie podlega złagodzonym rygorom czasowym
- niespełnienie wymagań czasowych nie jest niebezpieczne ale powoduje pogorszenie jakości usług świadczonych przez system
- zastosowania: techniki multimedialne, tworzenie wirtualnej rzeczywistości, urządzenia zdolne do samodzielnej eksploracji