1. Podaj przykłady trzech różnych definicji systemu operacyjnego.
- dystrybutor zasobów (ang. resource allocator) – program, który przydziela zasoby poszczególnym zamawiającym i kontroluje dostęp do zasobów
- program sterujący (ang. control program) - nadzoruje wykonanie programów użytkowych oraz operacji wejścia/wyjścia, kontrola błędów
- jądro systemu (ang. kernel, ang. nucleus) - program działający w komputerze nieustannie (program rezydentny), często jest synonimem systemu operacyjnego
- system operacyjny jest programem, który działa, jako pośrednik między użytkownikiem komputera a sprzętem komputerowym

1.2)Wymień zadania systemu operacyjnego.
- tworzenie środowiska do wykonywania programów
- powodowanie, aby system komputerowy był wygodny w użyciu
- zadania: sterowanie programami, rozdzielanie zasobów, szeregowanie, sterowanie we/wy, zarządzanie danymi

1.3)Wymień rodzaje systemów operacyjnych.
- wczesne systemy - „goła maszyna” (lata 50-te)
- proste systemy wsadowe
- wieloprogramowane systemy wsadowe
- systemy z podziałem czasu
- Systemy na PC
- systemy rozproszone
- Systemy zgrupowane
- systemy czasu rzeczywistego
- Systemy kieszonkowe (PalmOS, Symbian, WinCE)

1.4) Omowa idee wieloprogramowania i podziału czasu

Wieloprogramowane systemy wsadowe

Wiele zadań jest przechowywanych w pamięci operacyjnej a procesor, w czasie oczekiwania jednego zadania na jakąś usługę np. operację wejścia/wyjścia jest przydzielany innemu zadaniu

-pierwszy przypadek, gdy program decyduje za użytkownikowi a w zasadzie za operatora systemu

Wielozadaniowość -wymagania systemu

-Wieloprogramowanie daje wielozadaniowość
-Obsługa we/wy przez system
-Zarządzanie przydziałem pamięci
-Planowanie przydziału procesora -wybór zadania do wykonania
-Przydział urządzeń zewnętrznych (dyski, taśmy)

Systemy z podziałem czasu

-Rozszerzenie wieloprogramowości
-Procesor jest przydzielony jedynie zadaniom w pamięci operacyjnej i wykonuje na przemian wiele zadań,
przy czym przełączenia następują tak często, że użytkownicy mogą współdziałać z programem podczas jego
wykonania
-Jedno z zadań (proces) w pamięci: dialog z użytkownikiem (sesja)
-Zadania usuwane z pamięci są zapisywane na dysk do zbioru wymiany (ang. swap)
-Bezpośrednia komunikacja użytkownika z systemem dzięki wielozadaniowości: zadanie = sesja lub proces

1.5) Co to jest *spooling* i *swapping*?

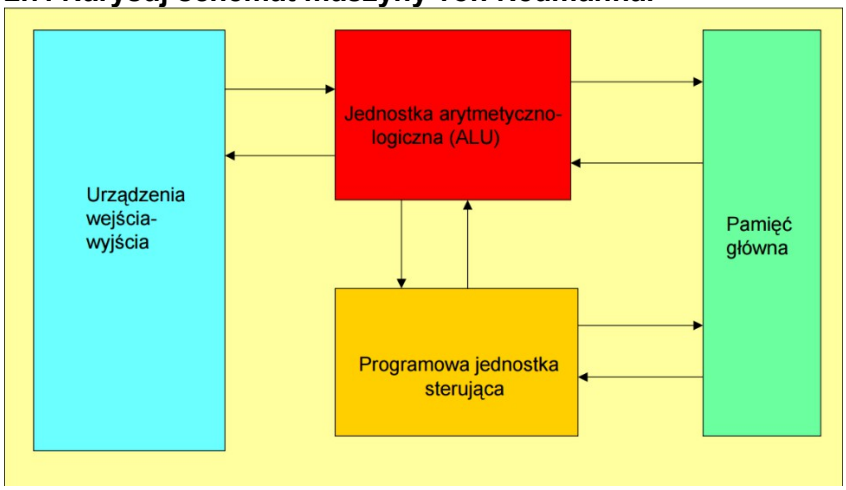
Spooling

-umożliwia wykonywanie zadań w czasie operacji
we/wy innych zadań
-wczytanie zadania z czytnika kart na dysk do
kolejki zadań
-wyprowadzenie wydruku zakończonego

zadania do kolejki na dysku a potem na drukarkę
-Pula zadań: zadania na dysku są wybierane do
wykonania w taki sposob, aby zwiększyć
wykorzystanie procesora
-Dysk jako wielki bufor
Wymiana -ang. swapping
-Problem: we/wy jest wolniejsze od procesora, więc nawet, wieloprogramowany system może powodować
przestoje w pracy procesora
-rozwiązania:
-zwiększenie pamięci: drogie i wzrasta apetyt
-wymiana
Co to jest wymiana?
-Zadanie zamiast przebywać bezczynnie w pamięci jest zapamiętane na dysku w zbiorze wymiany
-Zadanie (inne) zostaje wczytane ze zbioru wymiany do pamięci komputera i wykonywane
-Zadanie kończy się i jest usuwane z pamięci
1.6) Opisz generowanie systemu Linux (RedHat/Fedora).
Współczesne dystrybucje zawierają bazę danych nadzorującą instalowane pakiety i moduły sterujące. Użytkownik dokonuje wyboru jakie pakiety zainstalować a jakie pominąć, nie licząc pakietow podstawowych (jądro + interpreter). Umożliwia to, np. zainstalowanie systemu tylko z
trybem tekstowym.
2.1. Wymien składowe systemu komputerowego wraz z ich angielskimi odpowiednikami.
- sprzęt (ang. hardware) – zasoby o specyficznej architekturze oraz organizacji zarządzane przez system
operacyjny
- system operacyjny (operating system) - program, ktory nadzoruje i koordynuje dostęp programow do
zasobow
- programy użytkowe (software) – realizują, potrzeby użytkownikow systemu komputerowego (kompilatory, bazy danych, gry,....)
- użytkownicy (users) – ludzie, maszyny, komputery..
2.2. Wymien funkcje ogolne komputera.
- przetwarzanie danych - rożne formy i zakres wymagań
- przechowywanie danych: krotko i długotrwałe
- przemieszczanie danych - proces wejścia/wyjścia między komputerem a urządzeniem peryferyjnym
- sterowanie – wszystkimi powyższymi funkcjami
2.3. Podaj strukture komputera.
- jednostka centralna (CPU) – steruje działaniem komputera i realizuje funkcje przetwarzania danych
- pamięć głowna - przechowuje dane
- wejście/wyjście - przenosi dane między komputerem a światem zewnętrznym
- magistrala systemowa - mechanizmy zapewniające łączność między procesorem, pamięcią głowną a
wejściem/wyjściem
2.4. Podaj strukturę procesora.
- rejestry
- jednostka arytmetyczno – logiczna (ALU)
- jednostka sterująca
- połączenie wewnętrzne CPU
2.5. Podaj strukturę jednostki sterującej.
- rejestry i dekodery jedn. ster.

- układy logiczne szeregowania
- Pamięć sterująca

2.7. Narysuj schemat maszyny von Neumanna.



2.8. Opisz architekturę maszyny IAS : pamięć, rejestry, format danych i rozkazów.

- pamięć: 1000 x 40 bitowych słow
- reprezentacja binarna
- 2 x 20 bitowe rozkazy
- rejestry (pamiętane w CPU)
- rejestr buforowy pamięci
- rejestr adresowy pamięci
- rejestr rozkazów
- rejestr buforowy rozkazów
- licznik programu
- akumulator
- rejestr mnożenia-dzielenia

format danych i rozkazów:

rozkaz dzieli się na rozkaz lewy i
rozkaz prawy, każdy dwudziesto
bitowy

2.10. Scharakteryzuj kolejne generacje systemow komputerowych, podaj przedziały czasowe oraz implementacje.

I generacja 1949-1958
· ENIAC
· EDSAC
· IAS
· IBM 701,704,709
II generacja 1957-1964
· IBM/7000
· DEC PDP-1
III generacja 1964-1982
· IBM/360
· IBM/370
· IBM/4381
· DEC PDP-8
· Komputery oparte na mikroprocesorach Intela: 4004,8008,8080,8086
IV generacja 1982-do dziś
· Komputery oparte na mikroprocesorach Intela: 80286,386DX,486DX,Pentium,Pentium II Pentium III, Pentium IV, Celeron, Itanium, Penryn itd.

2.11. Opisz skale integracji mikroukładow.

-Mała skala integracji -od 1965 -do 100 elementow w mikroukładzie (ang. chip)
-Średniaskalaintegracji-do1971 -100-3,000 elementoww mikroukładzie

-Wielkaskalaintegracji-1971-1977 -3,000 -100,000 urządzeń w mikroukładzie
-Bardzo wielka skala integracji-1978 do dziś -100,000 -100,000,000 urządzeń w mikroukładzie
-Ultrawielka skala integracji -ponad100,000,000 urządzeń w chipie

16. Podaj wzory na MIPS i MFLOPS.

✦ MIPS - Millions of instructions per second - ile milionów instrukcji CPU wykonuje w ci ągu sekundy ✦ MIPS = $(T * 10^6)^{-1} * Ic = (CPI * 10^6)^{-1} * f$ - miara wydajności CPU

MFLOPS - Millions of floating point instructions per second
✦ ile milionów instrukcji zmiennoprzecinkowych CPU wykonuje w ciągu sekundy
✦ MFLOPS = (czas wykonania $* 10^6)^{-1} *$ (liczba wykonanych w programie instrukcji zmiennorzecinkowych)
✦ miara stosowana przy oszacowaniu wydajności gier i programów obliczeniowych

17. Podaj wzór Amdahla.

$$Speedup = \frac{\text{time to execute program on a single processor}}{\text{time to execute program on } N \text{ parallel processors}} = \frac{T(1-f)+Tf}{T(1-f)+\frac{Tf}{N}} = \frac{1}{(1-f)+\frac{f}{N}}$$

3.1. Wymien linie magistrali systemowej.
- linie danych - do przenoszenia danych np. szyna danych 8b a rozkaz 16b: 2 x transfer z pamięci
- linie adresowe - do określenia adresu danych
- linie sterowania - do sterowania liniami
- linie zasilania

3.3. Wymien funkcje jednostki centralnej.
-CPU = procesor = jednostka centralna
-Pobieranie rozkazow z pamięci
-Interpretowanie rozkazow
-Pobieranie danych (z pamięci lub we/wy)
-Przechowywanie danych w pamięci
-Przetwarzanie danych -wykonywanie rozkazow
-Zapisywanie wynikow (do pamięci lub na we/wy)

3.4. Opisz rejestry CPU oraz PSW.
- **rejestry procesora:**
- licznik programu (PC) - adres rozkazu do pobrania
- rejestr rozkazu (IR) - kod rozkazu
- rejestr adresowy pamięci (MAR) - adres lokacji
- rejestr buforowy pamięci (MBR) - dane do/z pamięci
- rejestry PSW - słowo stanu programu, informacje o stanie
- ***PSW* (bity, flagi)**
- znak - bit znaku ostatniej operacji
- zero - wynik operacji = zero
- przeniesienie - przeniesienie w wielokrotnej precyzji
- rowność - wynik porownania logicznego
- przepełnienie -
- zezwolenie/blokowanie przerwań
- nadzorca - tryb systemu lub tryb użytkownika

3.5. Opisz procesory RISC i CISC oraz wymien przykłady ich zastosowania.
- CISC - procesor o złożonej liście rozkazow
- złożona lista rozkazow = łatwiejsze programowanie

- trudne do zbudowania
- rozkazy maszynowe realizowane są przez mikrorozkazy
- IBM360/370, VAX, Pentium (Intel), C25xx (router Cisco): M68360 (Motorola), 20MHz
- RISC - procesor o zredukowanej liście rozkazow
- prosta lista rozkazow = trudniejsze programowanie
- łatwe do zbudowania ale programy są dłuższe
- SPARC, HP-PA, PowerPC, Athlon, C4500: MIPS R4600,100MHz

3.6. Jaki jest związek między rozkazami a mikrooperacjami?

Mikrooperacje

-Komputer wykonuje programy
-Cykl pobierania i wykonania
-Każdy cykl jest złoZony z mniejszych jednostek
✦dalsza dekompozycja rozkazu maszynowego
-Mikrooperacje
-kaZdy z podcykli (np. pobrania, adresowania pośredniego, wykonania, przerwania) obejmuje jedną
lub więcej mikrooperacji
-Mikrooperacje są elementarnymi operacjami (ang. atomic operation)wykonywanymi przez CPU
-Czas cyklu procesora –czas potrzebny do wykonania najkrotszej mikrooperacji
-Mikroprogramowanie -koncepcja zaproponowana przez M. V. Wilkes'a (1951)

3.7. Wymien kategorie urzadzen we/wy.

- przeznaczone do odczytu przez człowieka (np. monitor ekranowy, wydruk, dźwięk)
- przeznaczone do odczytu przez maszynę (np. dyski magnetyczne, taśmy, czujniki w robotach)
- komunikacyjne (np. modem, karta sieciowa)

3.8. Wymien funkcje oraz rodzaje modułow we/wy.

Funkcje

- sterowanie i taktowanie (zegar)
- komunikacja z procesorem
- komunikacja z urządzeniem
- buforowanie danych
- wykrywanie błędow

Rodzaje

-Sterownik wejścia/wyjścia (urządzenia) -prymitywny, wymaga szczegołowego sterowania
-Kanał (procesor) wejścia/wyjścia -przejmuje większość obciążenia we/wy CPU, wysoki priorytet dojścia
do procesora

3.9. Wymien i opisz sposoby realizacji we/wy.

- *programowane* – dane są wymieniane między procesorem a modułem we/wy, procesor czeka na
zakończenie operacji we/wy,
- *sterowane przerwaniami* – procesor wydaje operację we/wy i wykonuje dalsze rozkazy do momentu
zakończenia operacji we/wy (przerwanie we/wy),
- *bezpośredni dostęp do pamięci* (ang. direct memory access – DMA) – moduł we/wy wymienia dane
bezpośrednio z pamięcią bez udziału procesora

3.10. Wymien i opisz metody dostepu do pamieci.

- dostęp sekwencyjny

- dostęp liniowy blok po bloku w przod lub w tył,
- czas dostępu zależy od pozycji bloku względem pozycji bieżącej, przykład: taśmy,

- dostęp bezpośredni

- każdy blok na unikalny adres,

- czas dostępu realizowany przez skok do najbliższego otoczenia i sekwencyjne przeszukiwanie,
przykład: dysk magnetyczny,
- dostęp swobodny
- każda adresowalna lokacja w pamięci ma unikatowy, fizycznie wbudowany mechanizm adresowania,
- czas dostępu nie zależy od poprzednich operacji i jest stały, przykład: RAM,
- **dostęp skojarzeniowy**
- dane są lokalizowane raczej na podstawie porownania z ich zawartością niż na podstawie adresu,
- czas dostępu nie zależy od poprzednich operacji i jest stały, przykład: pamięć podręczna

3.11. Wymien rodzaje pamieci ze wzgledu na własnosci fizyczne.
- połprzewodnikowa (np. RAM),
- magnetyczna (dysk, taśma),
- magneto-optyczna (CD, DVD).

3.12. Podaj hierarchie pamieci.
- rejestry,
- pamięć podręczna
- poziom 1,
- poziom 2,
- pamięć głowna,
- dyskowa pamięć podręczna,
- pamięć dyskowa,
- pamięć optyczna | taśmowa,
- malejący koszt – rosnący czas dostępu – rosnąca pojemność

3.13. Opisz zasade lokalnosci odniesien.
- *zasada lokalności odniesie*ń oznacza, że w czasie wykonania programu odwołania do danych i rozkazow
mają tendencję do gromadzenia się
- przyczyna: programy zwykle zawierają tablice deklaracji zmiennych oraz stałych i wiele pętli iteracyjnych
i podprogramow
- wykorzystanie zasady lokalności odniesień pozwala na zmniejszenie częstotliwości dostępu
- przykład: pamięć podręczna procesora

3.14. Opisz działanie pamieci podrecznej.
-Cache zawiera fragment pamięci głownej
-Procesor sprawdza czy aktualnie potrzebne do wykonania rozkazu słowo z pamięci jest w cache'u
-jeśli nie, to blok pamięci o ustalonej liczbie K słow zawierający potrzebne słowo jest ściągnięty do
pamięci podręcznej
-Cache zawiera znaczniki identyfikujące bloki pamięci głownej

3.15. Opisz sposoby mapowania dla pamieci podrecznej.
Mapowanie bezpośrednie
-Każdy blok w pamięci głownej jest odwzorowywany na tylko jeden możliwy wiersz (ang. line) pamięci
-tzn. jeśli blok jest w cache'u to tylko w ściśle określonym miejscu
-Adres jest dzielony na dwie części
-najmniej znaczących w-bitow identyfikuje jednoznacznie słowo(ang.word) lub bajt w pamięci
-najbardziej znaczących s-bitow określa jeden z 2^s blokow pamięci
-najbardziej znaczące bity są dzielone na pole wiersza złożone z r bitow oraz znacznik w postaci s-r
bitow (najbardziej znacząca część)
Mapowanie skojarzeniowe
-mapowanie skojarzeniowe

-Blok pamięci może zostać załadowany do dowolnego wiersza w cache'u
-Adres dzielimy na dwie części: znacznik i słowo
-znacznik jednoznacznie określa blok pamięci
-Aby stwierdzić czy blok jest w cache'u trzeba zbadać zgodność adresu ze znacznikiem każdego wiersza
-Kosztowana metoda zwłaszcza gdy rozmiar cache'a jest duży
-konieczność rownoległego badania znacznikow wszystkich wierszy w pamięci podręcznej

Mapowanie sekcyjno-skojarzeniowe

-k-drożne mapowanie sekcyjno-skojarzeniowe
-Cache jest dzielony na v sekcji
-Każda sekcja składa się z k wierszy
-Dany blok B może zostać odwzorowany na dowolny wiersz jakiejś sekcji i
-np.jeśli sekcja ma 2 wiersze
-2 sposoby mapowania (mapowanie dwudrożne)
-blok może zostać umieszczony w jednym z dwu wierszy w jednej sekcji
-np. jeśli adres sekcji jest 13 bitowy
-określa się numer bloku w pamięci modulo 213
-bloki 000000, 008000, 018000,...., FF8000są mapowane na tę samą sekcję0

3.16. Załozmy, ze mamy komputer z pamiecia głowna o *x* M adresowalnych bajtach oraz pamiec podreczna o pojemnosci *y* KB i blokach *z* B mapujaca bezposrednio, skojarzeniowo, *k*-droznie sekcyjnie-skojarzeniowo pamiec głowna:

Zadnie liczone na ćwiczeniach

3.17. Wymien zastosowania oraz cechy pamieci DRAM.

- bity zapamiętane jako ładunki w kondensatorach,
- rozładowywanie,
- okresowe „odświeżanie ładunku",
- prosta budowa,
- mała,
- tania,
- wolna,
- asynchroniczna,
- zastosowania: pamięć głowna.

3.18. Wymien zastosowania oraz cechy pamieci SRAM.

- bity przechowywane za pomocą przerzutnikow,
- nie wymaga odświeżania,
- bardziej złożona budowa,
- większa,
- droższa,
- szybsza,
- zastosowanie: pamięć podręczna.

3.19. Wymien zastosowania oraz rodzaje pamieci ROM.

-ROM -ang. read-only memory
-trwały wzor danych
-zastosowanie:
-mikroprogramowanie
-tablice funkcji
-programy systemowe (BIOS, niskopoziomowe we/wy)
✦-podprogramy biblioteczne dla często wywoływanych funkcji
- zapisywana w trakcie produkcji
- bardzo droga dla małych serii,
- pamięć programowalna (PROM)
- do zapisu (tylko raz) wymagane jest specjalne urządzenie,
- pamięć głownie do odczytu
- optycznie wymazywalna programowalna pamięć stała (EPROM)

- wymazywanie: naświetlanie promieniowaniem ultrafioletowym z układu znajdującego się w obudowie,
- elektrycznie wymazywalna programowalna pamięć stała (EEPROM))
- zapisywane są tylko bajty zaadresowane,
- zapisywanie trwa dłużej niż odczyt (kilka mikrosekund/B),
- mniej gęsto upakowana niż EPROM,
- pamięć błyskawiczna (ang. Flash memory)
- wymazywanie elektryczne,
- nie umożliwia wymazywania na poziomie bajtu,
- szybsza niż EPROM,
- tańsza niż EEPROM.
- zastosowanie: BIOS, Cisco IOS, PenDrive= USB flash memory drive (ok. 8GB)

4.1. Wymien tryby adresowania w instrukcjach maszynowych.

-natychmiastowy
-bezpośredni
-pośredni
-rejestrowy
-rejestrowy pośredni
-z przesunieciem (indeksowanie
-stosowy

4.2. Opisz adresowanie natychmiastowe.

-Argument (ang. operand) jest częścią rozkazu
-A –zawartość pola adresowego w rozkazie
-Operand= A
-np. ADD 5
-dodaj 5 do zawartości akumulatora
-5 jest argumentem
-Nie ma odniesienia do pamięci w celu pobrania argumentu
-Zaoszczędzony jeden cykl pamięci
-Wielkość operandu ograniczona przez rozmiar pola adresowego

4.3. Opisz adresowanie bezposrednie.

-Pole adresowe zawiera adres operandu
-EA-efektywny adres (ang. effective address) lokacji zawierający odniesiony argument
-EA =A
-np. ADD A
-dodaj zawartość komorki A do akumulatora
-pod adresem A znajduje się operand
-Jedno odniesienie do pamięci
-Nie są potrzebne dodatkowe obliczenia
-Zakres adresacji ograniczony przez wielkość pola adresowego (słowo -opcode)

4.4. Opisz adresowanie posrednie.

-Pole adresowe odnosi się do słowa w pamięci, ktore zawiera pełnej długości adres argumentu
-(X) –zawartość lokacji X (rejestr lub adres pamięci)
-EA= (A) -znajdź A, znajdź adres (A) pod ktorym jest operand
-np.ADD (A)
-dodaj zawarość komorki pamięci wyznaczonej przez zawartość pod adresem A do akumulatora
-Większa przestrzeń adresowa: 2n gdzie n = długość słowa
-Może być zagnieżdżone ✦np. EA = (((A)))
-Wiele odniesień do pamięci głownej w celu pobrania argumentu, (dlatego wolne)

4.5. Opisz adresowanie rejestrowe.

-Operand znajduje się w rejestrze określonym w polu adresowym
-EA= R
-Ograniczona liczba rejestrow (32)

-Małe pole adresowe (zwykle 3 do 5 bitow)
-krotsze rozkazy i czas pobrania z rejestru
-Ograniczona przestrzeń adresowa

4.6. Opisz adresowanie posrednie rejestrowe.

-Adres w rejestrze odnosi się do słowa w pamięci, ktore zawiera adres operandu
-EA = (R)
-Operand jest w komorce pamięci wskazanej przez adres zawarty w rejestrze R
-Duża przestrzeń adresowa(2n)
-O jedno odniesienie do pamięci mniej niż przy adresowaniu pośrednim

4.7. Wymien rodzaje adresowania z przesunieciem.

Opis

-EA = A + (R)
-Pole adresowe zawiera dwie wartości
-A= wartość bazowa
-R = rejestr zawierający przesunięcie
-lub odwrotnie (zależy od rodzaju)

Rodzaje

-adresowanie względne
-adresowanie z rejestrem podstawowym
-indeksowanie (adresowanie indeksowane)
-kombinacje
-indeksowanie wtorne: EA = (A) + (R)
-indeksowanie wstępne: EA= (A + (R))

4.8. Opisz adresowanie wzgledne.

-R= licznik programu, PC
-EA = A + (PC)
-zgodność z zasadą lokalności odniesień (np. w pamięci podręcznej)

4.9. Opisz adresowanie z rejestrem podstawowym.

-A zawiera przesunięcie a R adres bazowy
-R może zawierać adres bezpośredni lub pośredni
-np. rejestry segmentowew80x86
-zgodność z zasadą lokalności odniesień

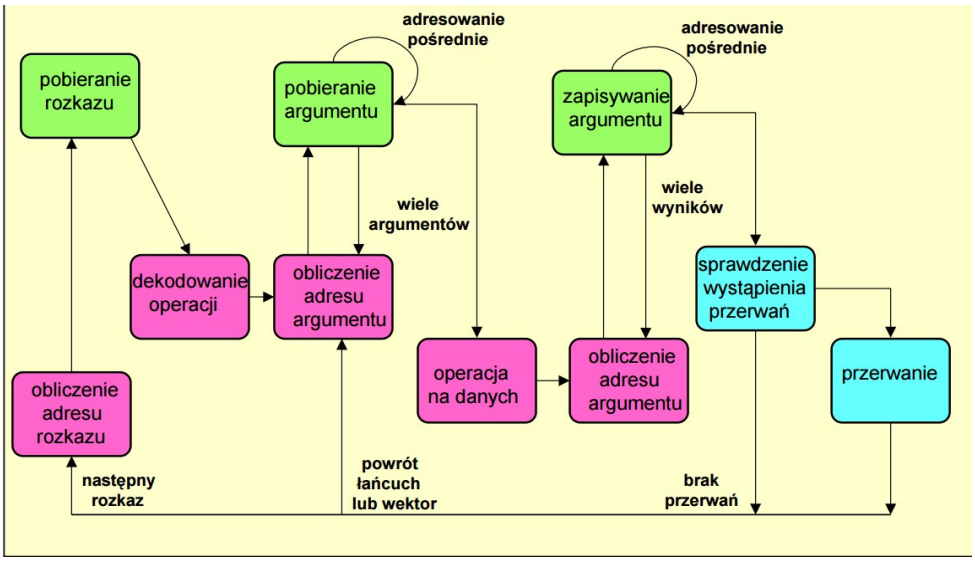
4.10. Opisz adresowanie indeksowane.

-A= adres bazowy, R= przesuniecie
-EA = A + R
-adresowanie dobre dla tablic: EA= A + R; R++
-autoindeksowanie: EA = A + (R); (R) <-(R) + 1

4.11. Opisz adresowanie stosowe.

-Operand znajduje się na wierzchołku stosu wskazanym przez rejestr
-np. ADD
-Usuń (POP) dwa wierzchołkowe elementy stosu i dodaj

4.17. Narysuj graf stanow cyklu rozkazu z przerwaniami.



4.18. Opisz potokowe przetwarzanie rozkazow.

✦Czas wykonywania rozkazu jest zwykle większy od czasu pobierania a więc musi być zwłoka wynikająca
z zajętości bufora
✦Rozkaz skoku warunkowego powoduje, że adres następnego rozkazu jest nieprzywidywalny
✦Gdy rozkaz skoku warunkowego przechodzi z etapu pobierania do wykonywania możemy pobrać rownież
rozkaz następny
✦jeśli jednak skok nastąpi to musi on zostać usunięty
✦Aby uzyskać większe przyspieszenie potok musi być przetwarzany w większej liczbie etapow

4.19. Opisz sprzetowa ochrone adresow.

-sprzęt jednostki centralnej porownuje każdy adresu wygenerowany w trybie użytkownika z zawartością
rejestrow bazowego i granicznego
-zwartości rejestru bazowego i granicznego mogą być załadowane jedynie w trybie monitora (load jest
instrukcją uprzywilejowaną)
-przerwanie protekcja pamięci

4.20. Opisz zegarowa ochrone CPU.

-Zegar jest ustawiany przez system operacyjny przed przekazaniem sterowania do programu użytkownika
-w trakcie wykonywania programu zegar jest zmniejszany
-jeśli zegar osiągnie 0 generowane jest przerwanie
-ładuj zegar jest rozkazem uprzywilejowanym
-zegar może być wykorzystany do realizacji podziału czasu -przerwanie zegarowe następuje po
wykorzystaniu kwantu czasu przez proces

4.21. Opisz dualny tryb operacji.

- system wielozadaniowy musi chronić system operacyjny oraz wykonywane programy przed każdym
niewłaściwie działającym programem
- należy wyposażyć sprzęt w środki pozwalające na rozrożnienie przynajmniej dwoch trybow pracy:
- tryb użytkownika - wykonanie na odpowiedzialność użytkownika

- tryb monitora = tryb nadzorcy = tryb systemu = tryb uprzywilejowany - wykonanie na odpowiedzialność
systemu operacyjnego
- bit trybu w sprzęcie komputerowym wskazujący na bieżący tryb pracy: system (0), użytkownik (1)
- wystąpienie błędu: przejście w tryb 0
- rozkazy uprzywilejowane- tryb systemu
52. Wymien składowe systemu operacyjnego.
✦ Zarządzanie procesami (ang. process management)
✦ Zarządzanie pamięcią operacyjną (ang. main memory management)
✦ Zarządzanie plikami (ang. file management)
✦ Zarządzanie systemem we/wy (ang. I/O system management)
✦ Zarządzanie pamięcią pomocniczą (ang. secondary-storage management)
✦ Praca sieciowa (ang. networking)
✦ System ochrony (ang. protection system)
✦ System interpretacji poleceń(ang. command-interpreter system)

5.1. Wymien usługi systemu operacyjnego.
- wykonanie programu - system powinien moc załadować program do pamięci i wykonać go,
- operacje we/wy – ponieważ program użytkowy nie wykonuje operacji we/wy bezpośrednio więc musi to
oferować system
- manipulowanie systemem plikow - program musi mieć możliwość (pod kontrolą) do czytania, pisania,
tworzenia i usuwania plikow
- komunikacja – wymiana informacji pomiędzy procesami wykonywanymi na tym samym lub zdalnym
komputerze np. za pomocą pamieci dzielonej lub przekazywania komunikatow
- wykrywanie błędow – zapewnienie prawidłowości działania komputera poprzez wykrywanie i obsługę
wszystkich błedow w jednostce centralnej, pamięci operacyjnej, urządzeniach we/wy (np. błąd sumy
kontrolnej) i w programie użytkownika (np. przekroczenie czasu)
- dodatkowe funkcje systemu nie są przeznaczone do pomagania użytkownikowi, lecz do optymalizacji
działania samego systemu:
- **przydzielanie zasobow** dla wielu użytkownikow i wielu zadań w tym samym czasie
- rozliczanie – przechowywanie danych o tym, ktorzy użytkownicy i w jakim stopniu korzystają z
poszczegolnych zasobow komputera (statystyka użytkowania)
- ochrona – zapewnienie aby cały dostęp do zasobow systemu odbywał się pod kontrolą np. dostęp
przez modem po podaniu hasła
5.2. Wymien etapy realizacji polecenia w systemie Unix.
- interpreter poleceń: powłoka
- system wielozadaniowy
- wprowadzenie polecenia to zapoczątkowanie procesu funkcją systemową fork lub exec
- powłoka oczekuje na zakończenie procesu
- jeśli polecenie wykonuje się w tle to powłoka może przyjmować nowe polecenia
- proces wykonuje funkcję systemową exit
- powłoka może realizować nowe polecenie
5.3. Co to sa wywołania (funkcje) systemowe?

- funkcje systemowe tworzą interfejs między wykonywanym programem a systemem operacyjnym,
- dostępne na poziomie języka maszynowego (asemblera)
- pewne języki zastępują asembler w programowaniu systemowym i umożliwiają bezpośrednie
wykonywanie funkcji systemowych (np. C, C++, Bliss, PL/360, PERL)
- win32 API (Application Programmer Interface) - wielki zbior procedur dostarczanych przez Microsoft,
ktore umożliwiają realizację funkcji systemowych

5.4. Wymien podstawowe metody przekazywania parametrow miedzy procesem a systemem.

Są trzy metody przekazywania parametrow między wykonywanym programem a systemem operacyjnym
-umieszczenie parametrow w rejestrach jednostki centralnej
-zapamiętanie parametrow w tablicy w pamięci operacyjnej i przekazanie adresu tej tablicy jako
parametru w rejestrze
-składowanie przez program parametrow na stosie i zdejmowanie ze stosu przez system operacyjny

5.5. Wymien rodzaje wywołan (funkcji) systemowych.

- nadzorowanie procesow
- operacje na plikach
- operacje na urządzeniach
- utrzymywanie informacji
- komunikacja

5.9. Wymien kategorie programow systemowych.

- **manipulowanie plikami** - programy do tworzenia, usuwania, kopiowania, przemianowywania,
składowania i wyprowadzania zawartości plikow
- **informowanie o stanie systemu** - logi systemowe
- **tworzenie i zmienianie zawartości plikow** - np. edytory
- **translatory językow programowania** - C, Pascal, Basic, Lisp..
- **ładowanie i wykonywanie programow** - konsolidatory i programy ładujące
- **komunikacja** - np. remote login
- **programy aplikacyjne**

5.10. Wymien struktury systemow operacyjnych oraz przykłady ich realizacji.

- monolityczna - jądro jednoczęściowe
- OS/360 - 5000 programistow 1M kodu w ciągu 5 lat
- IBM/360 MVT/TSO - koszt 50M $
- AIX - Unix wersji IBM - jądro dwuczęściowe
- MS-DOS, Unix

- warstwowa
- struktura hierarchiczna - skutki małych zmian w jednej warstwie trudne do przewidzenia w innych
warstwach
- system THE (Technische Hogeschool w Eindhoven)

- mikrojądra
- jedynie bezwzględnie niezbędne funkcje systemowe w jądrze systemu (np. mikrojądro L4 ma 12kB
kodu, 7 funkcji systemowych)
- Mach – opracowany w Carnegie-Mellon University
- Tru64 UNIX – Digital UNIX
- Apple MacOS X Server
- QNX - przykład systemu czasu rzeczywistego

5.12. Wyjasnij zasade maszyny wirtualnej.
maszyna wirtualna (ang. virtual machine) jest logiczną konsekwencją podejścia warstwowego: jądro
systemu jest traktowane jako sprzęt
- maszyna wirtualna dostarcza identycznego interfejsu dla sprzętu
- system operacyjny tworzy wirtualne systemy komputerowe, każdy proces ma do dyspozycji własne
(wirtualne) jądro, dyski, pamięć, drukarki
- zasoby fizycznego komputera są dzielone w celu utworzenia maszyn wirtualnych
- planowanie przydziału procesora jest tak wykorzystane, że użytkownik ma wrażenie jakoby miał do
dyspozycji własny procesor
- spooling i system zarzadzania plikami jest wykorzystany tak, że powstaje wrażenie użytkowania drukarki,
czytnika na wyłączność
- zwykłe terminale użytkownika funkcjonują jak konsole operatorskie maszyny wirtualnej

5.13. Co to jest *mikrojadro* i jakie sa jego wady i zalety?
- *mikrojądra* (ang. microkernel, µ-kernel)
- jedynie bezwzględnie niezbędne funkcje systemowe w jądrze systemu
- jądro zredukowane do małego zbioru funkcji rdzeniowych tzw. mikrojądro
- większość operacji w przestrzeni użytkownika
- obsługa: urządzeń, plikow, pamięci wirtualnej, grafiki, ochrona przy pomocy komunikatow (np. IPC)
- zalety i wady mikrojąder
+ jednolity interfejs dla procesow
+ łatwość w kontrolowaniu kodu systemu operacyjnego
+ przenaszalność systemu (na inną architekturę)
+ niezawodność - małe jądro łatwiej przetestować (ok. 300 kB kodu, 140 funkcji systemowych)
-- wydajność - zbudowanie, zakolejkowanie, wysłanie, odebranie, potwierdzenie komunikatu
-- poprawianie wydajności prowadzi do rozbudowy mikrojądra
5.14. Omow trzy przykłady realizacji maszyny wirtualnej.
VM/370
Obserwacja: system podziału czasu daje użytkownikowi wieloprogramowanie poprzez stworzenie wrażenia
maszyny wirtualnej
-Wniosek: należy te dwie idee rozdzielić
-Monitor maszyny wirtualnej zarządza zasobami sprzętowymi wielu systemow operacyjnych CMS
-Przykład: program czyta plik z dysku wirtualnego
-(wirtualne) wywołanie systemowe CMS
-symulacja I/O przez system VM/370
JVM
-Program w Javie .java kompilowany programem javac tworzy kod wynikowy .class
-Kod wynikowy .class jest niezależny od platformy sprzętowej
-Do wykonania .class potrzebny jest jednak programjava(Java Virtual Machine (JVM)), ktory jest
implementacją maszyny wirtualnej
-Maszyna wirtualna Java (JVM)składa się z: -Classloader -Class verifier -Runtime interpreter
VPC2004
-aplikacja na system Windows
-deklaracja ilości pamięci RAM
-kreator wirtualnego hdd, fdd
-podłączanie rzeczywistych urządzeń : CD, FD, COM, LPT

-karta sieciowa: Microsoft Loopback
-prawy Alt do zmiany ekranu

63. Jaka jest róznica pomiedzy hyperwizorem typu 0, 1 i 2?
64. Omów rodzaje nieautoryzowanego wyjawienia.
✦ ujawnienie (ang. exposure) – np. przypadkowe
opublikowanie w sieci informacji poufnych
✦ przechwycenie (ang. interception) –np. pakietów w bezprzewodowej sieci LAN
✦ wnioskowanie (ang. inference) –np. napodstawie szczątkowych danych
✦ przejęcie (ang. intrusion)- np. dostępu dowrażliwych danych poprzez złamanie zabezpiecze ń systemu
65. Omów rodzaje podstepów komputerowych.
✦ maskarada (ang. masquerade)
- zalogowanie się na czyjeś konto
-uzyskanie nieautoryzowanego dostępu - koń trojański (ang.
Trojan horse)
✦ sfałszowanie (ang. falsification) - wprowadzenie fałszywch danych do pliku lub bazy danych
✦ odżegnanie (ang. repudiation) - zaprzeczanie wysyłania lub posiadaniu danych
66. Omów rodzaje przerwania działania systemu komputerowego.
✦ niezdolność do pracy (ang. incapacitation) – atak na dostępność
- zniszczenie sprzętu
- złośliwe oprogramowanie (malware - ang. malicious software): trojany, wirusy (ang. viruses), robaki (ang. worms)
✦ zmiana działania (ang. corruption) –atak naintegralność systemu
-złośliwe oprogramowanie powoduje niepożądane działanie systemu
- modyfikacja oprogramowania poprzez wstawienie tylnej furtki (ang.
backdoor)
✦ utrudnianie pracy (ang. obstruction)
- obciążanie systemu (odmowa usługi, DOS –ang. denial of service)
- zmiana działania łączy komunikacyjnych
67. Omów rodzaje uzurpacji komputerowej.
✦ niewłaściwe przywłaszczenie (ang.missappropriation)
-kradzież serwisu (np. CPU) : DDoS (ang. distributed DOS) – kradzież CPU i zasobów systemu operacyjnego porzez zmasowany atak z sieci
✦ niewłaściwe użycie (ang. misuse)
- uzyskanie dostępu do systemu (np. przez hakera (ang. hacker))
68. Omów kategorie intruzów komputerowych.
✦ podszywający
- masquerader (pol. przebieraniec)– ten kto penetruje system kontroli dostępu użytkowników w celu ich wykorzystania
✦ nadużywający
- misfeasor (pol. nadużywający władzy) –użytkownikwykorzystujący dostęp do danych do których nie mauprawnień
✦ skrywający
-clandestine (pol. tajny) user –ten kto przejmuje kontrolowanie (ang. audit) systemu aby uniknąć weryfikacji
69. Podaj klasyfikacje wirusów komputerowych.
✦cel infekcji (ang. target)
- wirus w sektorze startowym (ang. boot sektor infector)
- wirus w pliku systemowym (ang. file infector)
- makrowirus (ang. macro virus) – infekuje dokumenty, np. w MS Word
✦ strategia ukrycia (ang. concealment strategy)
- wirus zakodowany (ang. encrypted) –każdorazowe generowanie klucza idekodowanie wirusa, gdy wirus się replikuje koduje się za pomocą nowego klucza

- wirus utajniony (ang. stealth) –przed antywirusem np. skompresowany wirus
- wirus polimorficzny (ang. polimorphic) – mutujący się (zmieniający swój wzorzec bitowy)
- wirus metamorficzny(ang. metamorphic) –mutujący, zmieniający swoje działanie i wygląd

70. Omów boty oraz rootkity.

Boty

✦ ang. bot (robot), zombie, drone

✦ potajemne przejęcie kontroli nad innymi hostami (setki, tysiące) podłączonymi do Internetu w celu utworzenia sieci botów (ang. botnet)

✦ Przypuszczenie ataku poprzez mechanizm RCF (ang. remote control facility)

- trudno jest go przypisać konkretnemu twórcy bota

✦ Wykorzystanie botnetu

- DDoS (distributed-denial-of-service) – atak na serwisy sieciowe
- spamming – wysłanie e-maili
- sniffing – podsłuchiwanie ruchu sieciowego
- keylogging – podsłuchiwanie klawiatury
- rozpowszechnianie złośliwego oprogramowania
- instalowanie BHO (browser helper object) – dodatkowe reklamy
- ataki na sieci IRC, gry sieciowe, itp.

Rootkity

✦ ang. rootkits

✦ Zbiór programów zainstalowanych w systemie w celu uzyskania uprawnień administratora (root) oraz ukrycie swojego istnienia

- trwałe – aktywujące się w czasie ładowania systemu
- nietrwałe - w pamięci RAM
- w trybie jądra (np. funkcja systemowa) lub użytkownika (API programu użytkownika)

✦ Instalacja: wykorzystanie luki w systemie (np. otwarcie portu), uzyskanie uprawnień roota (cracking, malware, luka w systemie), upload rootkita, ukrycie rootkita podczas instalacji, otwarcie portu dla dalszych działań

✦ Tablica wywołań systemowych

- modyfikacja tablicy wywołań systemowych
- modyfikacja wywołań systemowych
- przekierowanie tablicy wywołań systemowych

6.1. Podaj przykłady uruchamiania procesow w systemie Unix.

-Proces to program działający we własnej przestrzeni adresowej
-proces wsadowy –polecenie batch, at, cron
-procesy interakcyjne
-procesy pierwszoplanowe –związane z terminalem (np. polecenie ls)
-procesy drugoplanowe –wykonywane w tle
-demony (ang. daemons) –wystartowane serwisy
-init, syslogd, sendmail, lpd, crond, getty, bdflush, pagedaemon, swapper, inetd, named, routed, dhcpd, portmap, nfsd, smbd, httpd, ntpd
-Limity zasobow procesow: polecenia limit, ulimit

6.2. Podaj komendy do sterowania procesami w systemie Unix.

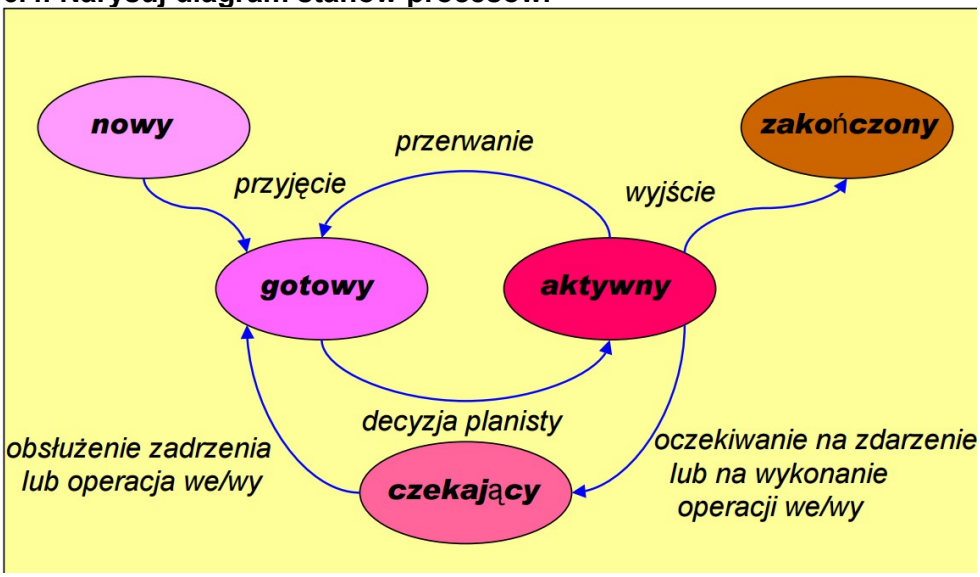
-Uruchamianie w tle: &
✦np. ls –R / &
-Zatrzymanie procesu pierwszoplanowego: ^Z
-Ponowne uruchamianie procesu w tle: bg
-Listowanie procesow w tle: jobs
-Odwołanie do procesu n w tle: %n
-Odwołanie do procesu xyz w tle: %?xyz

-Przenoszenie z tła na pierwszy plan: fg %

6.3. Wymien stany procesu wraz z ich angielskimi odpowiednikami.

- wykonujący się proces zmienia swoj stan (ang. state)
- nowy (ang. new): proces został utworzony
- aktywny (ang. running): są wykonywane instrukcje
- oczekiwanie (ang. waiting): proces czeka na zdarzenie (np. zakończenie we/wy)
- gotowy (ang. ready): proces czeka na przydział procesora
- zakończony (ang. terminated): proces zakończył działanie

6.4. Narysuj diagram stanow procesow.



6.7. Opisz zawartosc PCB.

- każdy proces w systemie operacyjnym jest reprezentowany przez blok kontrolny procesu zawierający
- stan procesu – gotowy, nowy, aktywny, czekający, zatrzymany
- licznik rozkazow - adres następnego rozkazu do wykonania w procesie
- rejestry procesora - zależą od architektury komputera: akumulatory, rejestry (ogolne, bazowe, indeksowe)

wskaźniki stosu przechowywane aby proces mogł być kontynuowany po przerwaniu

- informacje o planowaniu przydziału procesora - priorytet procesu, wskaźniki do kolejek porządkujących

zamowienia

- informacje o zarządzaniu pamięcią - zawartości rejestrow granicznych, tablice stron, tablice segmentow w

zależności od systemu używanej pamięci

- informacje do rozliczeń - ilość zużytego czasu procesora i czasu rzeczywistego, ograniczenia czasowe,

numery kont, numery zadań

- informacje o stanie we/wy - lista zaalokowanych urządzeń, wykaz otwartych plikow

6.8. Omow przyczyny przełaczania procesu.

✦Przerwanie –zdarzenie zewnętrzne w stosunku do procesu

✦Pułapka –zdarzenie wewnątrz aktualnie przetwarzanego procesu, błąd lub warunek wyjątku

✦Polecenie administracyjne –wywołanie funkcji systemu operacyjnego

6.9. Omow rodzaje planistow i zadania przez nich realizowane.

- planista długoterminowylub planista zadań– wybiera procesy, ktore powinny być sprowadzone do pamięci

do kolejki procesow gotowych

- planista krotkoterminowy lub planista przydziału procesora - wybiera proces następny do wykonania z

kolejki procesow gotowych i przydziela mu procesor
- planista krotkoterminowy jest wołany bardzo często (milisekundy) dlatego musi być bardzo szybki
- planista długoterminowy jest wołany rzadko (sekundy, minuty) dlatego może nie być szybki
- planista długoterminowy nadzoruje stopień wieloprogramowości (liczbę procesow w pamięci)
- planista długoterminowy powinien dobrać mieszankę procesow zawierającą zarowno procesy ograniczone
przez we/wy jak i procesor
- planista średnioterminowy– swapping w celu uzyskania lepszego doboru procesow

6.10. Omow przełaczanie kontekstu.

- gdy procesor przełącza do innego procesu system musi zachować stan starego procesu i załadować
zachowany stan nowego procesu. Czynność tę nazywamy przełączaniem kontekstu
- przełączanie kontekstu jest ceną za wieloprogramowość; system operacyjny nie wykonuje wtedy żadnej
użytecznej pracy
- czas przełączenia kontekstu zależy od sprzętu(zwykle od 1 do 1000 milisekund)

6.13. Omow tworzenie procesu *zombie* i *orphan*.

Zombie-W przypadku, gdy proces potomny kończy się w czasie, gdy jego proces rodzicielski nie wykonuje
funkcji wait, wowczas kończący się proces jest umieszczany w stanie zawieszenia i staję się procesem
zombie. Proces zombie zajmuje pozycję w tabeli utrzymywanej w jądrze dla kontroli procesow, ale nie
używa żadnych innych zasobow jądra. Zostanie on zakończony, gdy jego proces rodzicielski zażąda
potomka za pomocą wykonania funkcji wait.

Orphan – to proces ktorego rodzic przestał istnieć (np. została wywołana funkcja exit). Przejmuje go
proces init (żeby nie mogł się stać zombie).

6.14. Omow komunikacje posrednia i bezposrednia miedzy procesami.

komunikacja bezpośrednia

- proces musi jawnie nazwać odbiorcę
- nadaj(P,komunikat) - nadaj komunikat do procesu P
- odbierz(Q,komunikat) - odbierz komunikat od procesu Q
- własności łącza
- łącze jest ustanawiane automatycznie, do komunikowania wystarczy znajomość identyfikatorow
- łącze dotyczy dokładnie dwoch procesow
- między każdą parą procesow istnieje dokładnie jedno łącze
- łącze jest zwykle dwukierunkowe, ale może być jednokierunkowe

komunikacja pośrednia

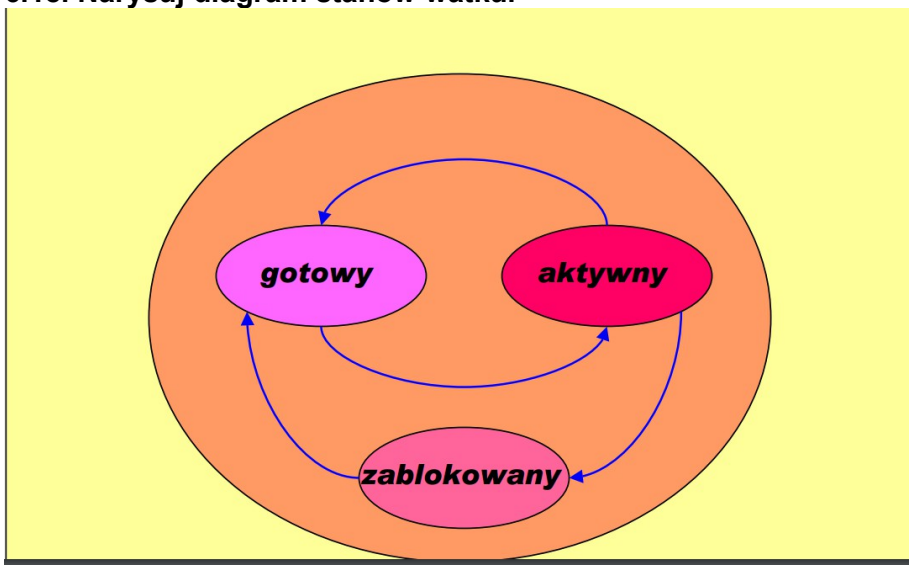
- komunikaty są nadawane i odbierane za pomocą skrzynek pocztowych nazywanych także portami
- każda skrzynka ma swoj unikalny identyfikator
- procesy komunikują się jeśli mają wspolną skrzynkę
- własności łącza
- łącze jest ustanawiane jedynie wtedy, gdy procesy dzielą skrzynkę
- łącze może być związane z więcej niż dwoma procesami
- każda para procesow może mieć kilka łączy, z ktorych każdy odpowiada jakiejś skrzynce
- łącze może być jedno- lub dwukierunkowe

6.16. Wymien zasoby uzywane przez watki.

- wątek nazywany niekiedy procesem lekkim jest podstawową jednostką wykorzystania procesora. W skład

tej jednostki wchodzą
- licznik rozkazow
- zbior rejestrow
- obszar stosu
- wątek współużytkuje z innymi rownorzędnymi wątkami
- sekcję kodu
- sekcję danych
- zasoby systemu (takie jak otwarte pliki i sygnały) zwane wspolnie zadaniem

6.18. Narysuj diagram stanow watku.



6.19. Wymien typy watkow.
- wątki obsługiwane przez jądro
- wątki tworzone na poziomie użytkownika za pomocą funkcji bibliotecznych
- zaleta: szybsze przełączanie, wada: planowanie wątkow
- hybrydowe podejście
- wątki zarządzane przez JVM
- zielone – watki w ramach maszyny wirtualnej
- natywne– watki w ramach systemu operacyjnego

6.20. Wymien sposoby odwzorowan watkow.
-Wiele do jednego
-Jeden do jednego
-Wiele do wielu
-Jeden do wielu

7.1. Wymien sytuacje w jakich planista przydziału procesora podejmuje decyzje o przydziale procesora.
- decyzje o przydziale procesora podejmowane są
- gdy proces przeszedł od stanu aktywności do czekania (np. z powodu we/wy)
- gdy proces przeszedł od stanu aktywności do gotowości (np. wskutek przerwania)
- gdy proces przeszedł od stanu czekania do gotowości (np. po zakończeniu we/wy)
- gdy proces kończy działanie

7.2. Wymien warunki planowania bez wywłaszczen.

- planowanie bez wywłaszczeń : proces, ktory otrzyma procesor, zachowuje go tak długo aż nie odda go z
powodu przejścia w stan oczekiwania lub zakończenia (nie wymaga zegara)
- planowanie w sytuacji 1 i 4 nazywamy niewywłaszczeniowym – z poprzedniego pytania

7.4. Co to jest *dispatcher* oraz *dispatch latency*?

- *ekspedytor* (ang. dispatcher) jest modułem, ktory faktycznie przekazuje procesor do dyspozycji procesu
wybranego przez planistę krotkoterminowego; obowiązki ekspedytora to
- przełączanie kontekstu
- przełączanie do trybu użytkownika
- wykonanie skoku do odpowiedniej komorki w programie użytkownika w celu wznowienia działania programu
- opoźnienie ekspedycji (ang. dispatch latency) to czas, ktory ekspedytor zużywa na wstrzymanie jednego
procesu i uaktywnienie innego

7.5. Co to jest przepustowosc oraz wykorzystanie CPU?

-Wykorzystanie procesora –procent czasu, przez ktory procesor pozostaje zajęty
-Przepustowość -liczba procesow kończących w jednostce czasu

7.6. Podaj definicje czasu (cyklu) przetwarzania, czasu oczekiwania, czasu odpowiedzi.

-Czas cyklu przetwarzania czas między nadejściem procesu do systemu a chwilą zakończenia procesu
-suma czasow czekania na wejście do pamięci, czekania w kolejce procesow gotowych, wykonywania procesu przez CPU i wykonywania operacji we/wy
-Czas oczekiwania suma okresow, w ktorych proces czeka w kolejce procesow gotowych do działania
-Czas odpowiedzi lub reakcji -ilość czasu między wysłaniem żądania a pojawieniem się odpowiedzi bez
uwzględnienia czasu potrzebnego na wyprowadzenie odpowiedzi (np. na ekran).
-czas odpowiedzi jest na ogoł uzależniony od szybkości działania urządzenia wyjściowego
-miara zastępująca miarę czasu cyklu przetwarzania w systemach interakcyjnych

7.7. Wymien kryteria optymalizacji algorytmu planowania.

- maksymalne wykorzystanie procesora
- maksymalna przepustowość
- minimalny cykl przetwarzania
- minimalny czas oczekiwania
- minimalny czas odpowiedzi
- minimalizowanie wariancji czasu odpowiedzi
- mało algorytmow minimalizujących wariancję
- pożądany system z sensownym i przewidywalnym czasem odpowiedzi

7.8. Opisz algorytm FCFS.

- pierwszy zgłoszony, pierwszy obsłużony (ang. first-come, first-served – FCFS)
- implementuje się łatwo za pomocą kolejek FIFO - blok kontrolny procesu dołączany na koniec kolejki,
procesor dostaje PCB z czoła kolejki
- algorytm FCFS jest niewywłaszczający
- proces utrzymuje procesor do czasu aż zwolni go wskutek zakończenia lub zamowi operację we/wy -
algorytm FCFS jest kłopotliwy w systemach z podziałem czasu, bowiem w takich systemach ważne jest
uzyskiwanie procesora w regularnych odstępach czasu

7.9. Opisz algorytm SJF wywłaszczajacy.

- algorytm najpierw najkrotsze zadanie (ang. shortest-job-first - SJF) wiąże z każdym procesem długość
jego najbliższej z przyszłych faz procesora. Gdy procesor staje się dostępny wowczas zostaje przydzielony

procesowi o najkrotszej następnej fazie (gdy fazy są rowne to mamy FCFS)
- wywłaszczający – SJF usunie proces jeśli nowy proces w kolejce procesow gotowych ma krotszą następną
fazę procesora od czasu do zakończenia procesu; metoda najpierw najkrotszy pozostały czas
(ang. shortest-remaining-time-first - SRTF)

7.10. Opisz algorytm HRRN.

-Def. Stosunkiem reaktywności nazywamy liczbę R = 1+ w/t, gdzie w oznacza czas oczekiwania na
procesor zaś t -fazę procesora
-Największy stosunek reaktywności jako następny
-Faworyzuje krotkie zadania, lecz po pewnym czasie oczekiwania dłuższy proces uzyska CPU
-Podobnie jak SJF i SRTF rownież algorytm HRRN wymaga oszacowaniadla następnej fazy procesora
-Faworyzuje krotkie zadania jednak oczekiwanie dłuższych zadań zmienia ich współczynnik i w
konsekwencji pozwala im uzyskać dostęp do CPU
-Ma dobry czas odpowiedzi
-Proces zawsze dostanie się do CPU (po pewnym czasie) tj. nie ma groźby zagłodzenia procesow

7.11. Podaj wzor na oszacowanie nastepnej fazy procesora.

-Nie ma sposobu na poznanie długości następnej fazy, możemy ją jedynie oszacować
-Można tego dokonać wyliczając średnią wykładniczą poprzednich faz procesora
-t(n) = długość n-tej fazy procesora
-a -liczba z przedziału [0,1], zwykle 0.5
-Definiujemy średnią wykładniczą jako*: s(n+1) = a*t(n) + (1-a)*s(n)* gdzie s(n+1) = przewidywana
długość następnej fazy a s(n) przechowuje dane z minionej historii; s(0) –stała (np. średnia wzięta z
całego systemu)
-a=0
-s(n+1) = s(n)
-niedawna historia nie ma wpływu
-a=1
-s(n+1) = t(n)
-jedynie najnowsze notowanie długości fazy ma wpływ
-a*(1-a)≠0i rozwiniemy wzor to:
s(n+1) = a*t(n) + (1-a)*a*t(n-1) + ... (1-a)^j*a*t(n-j) + ...+ (1-a)^(n+1)*s(0)
-Ponieważ a i (1-a) są mniejsze od 1 to starsze składniki (przeszłość) mają coraz mniejszą wagę

7.12. Opisz planowanie priorytetowe.

-SJF jest przykładem planowania priorytetowego, w ktorym każdemu procesowi przypisuje się priorytet
-Priorytety należą do pewnego skończonego podzbioru liczb naturalnych np. [0..7], [0,4095]
-Procesor przydziela się procesowi o najwyższym priorytecie (jeśli priorytety są rowne to FCFS)
-planowanie priorytetowe wywłaszczające
-planowanie priorytetowe niewywłaszczające
-SJF -priorytet jest odwrotnością następnej fazy

7.13. Opisz algorytm RR.

- planowanie rotacyjne zaprojektowano dla systemow z podziałem czasu
- każdy proces otrzymuje małą jednostkę czasu, nazywaną kwantem czasu. Gdy ten czas minie proces jest
wywłaszczany i umieszczany na końcu kolejki zadań gotowych (FCFS z wywłaszczeniami)

- jeśli jest n procesow w kolejce procesow gotowych a kwant czasu wynosi q to każdy proces otrzymuje 1/n
czasu procesora porcjami wielkości co najwyżej q jednostek czasu.
- każdy proces czeka nie dłużej niż (n-1)*q jednostek czasu
- wydajność algorytmu
- gdy q duże to RR przechodzi w FCFS
- gdy q małe to mamy dzielenie procesora ale wtedy q musi być duże w stosunku do przełączania kontekstu
- mniejszy kwant czasu zwiększa przełączanie kontekstu
- czas cyklu przetwarzania zależy od kwantu czasu

7.14. Rozwazmy procesy: *P*1, *P*2, *P*3, *P*4, *P*5 o nastepujacych czasach trwania fazy (ms): *f*1, *f*2, *f*3, *f*4, *f*5, ktore przybyły o czasie *t*1, *t*2, *t*3, *t*4, *t*5 i priorytetach: *p*1, *p*2, *p*3, *p*4, *p*5 przy czym najwyzszy priorytet ma *pi*. Narysuj diagramy Gantta dla algorytmow FCFS, HRRN, SJF, SRTF, priorytetowy z i bez wywłaszczen, RR (kwant=q) oraz oblicz ich srednie czasy cyklu przetwarzania i oczekiwania. Uwaga: jesli nie sa podane *t*1, *t*2, *t*3, *t*4, *t*5 to przyjmujemy, ze *t*1 = *t*2 = *t*3 = *t*4 = *t*5 = 0 oraz ze procesy przybyły w porzadku *P*1, *P*2, *P*3, *P*4, *P*5 jesli zas nie sa podane *p*1, *p*2, *p*3, *p*4, *p*5 to przyjmujemy *p*1 = *p*2 = *p*3 = *p*4 = *p*5.

Zadnie liczone na ćwiczeniach

7.15. Opisz algorytm wielopoziomowych kolejek ze sprzezeniem zwrotnym.

- kolejki wielopoziomowe z sprzężeniem zwrotnym umożliwiają przesuwanie procesow między kolejkami
- proces, ktory zużywa za dużo procesora można zdymisjonować poprzez przesunięcie go do kolejki o
niższym priorytecie i dzięki temu zapobiec zagłodzeniu innych procesow
- postępowanie takie prowadzi do pozostawienia procesow ograniczonych przez we/wy oraz interakcyjnych
w kolejkach o wyższych priorytetach
- trzy kolejki:
- Q0 - kwant czasu 8 milisekund
- Q1 - kwant czasu 16 milisekund
- Q2 - FCFS
- planowanie
-nowe zadanie wchodzi do kolejki Q0 dostaje 8 milisekund i jeśli się nie zmieści w tym czasie zostaje przeniesione na koniec kolejki Q1
-gdy kolejka Q0 się oprożni wowczas proces z czoła kolejki Q1 dostanie kwant czasu 16 milisekund
-jeśli ponownie nie zmieści się w 16ms zostaje wywłaszony do kolejki Q2
-procesy z kolejki Q2są obsługiwane algorytmem FCFS i tylko wtedy, gdy puste są kolejki Q0 i Q1
- algorytm ten daje najwyższy priorytet procesom o fazie nie większej niż 8ms, procesy o fazie między 8ms
i 24ms są także szybko obsługiwane; długie procesy wchodzą do kolejki 2 i są obsługiwane (FCFS) w
cyklach pracy procesora nie wykorzystanych przez procesy z kolejek 0 i 1
- planowanie ze sprzężeniem zwrotnym jest najogolniejszym i najbardziej złożonym algorytmem
planowania przydziału procesora

7.16. Opisz planowanie watkow.

-PCS –planowanie ULT na bazie LWP (struktura pośrednia jądra) zwykle priorytetowe
-modele M:1 (many-to-one) i M:M (many-to-many)
-nie ma podziału czasu między wątkami o rownym priorytecie
-SCS –planowanie wszystkich wątkow KLT w systemie
-model 1:1 (one-to-one) (XP, Solaris, Linux)

-✦POSIX Pthread API –planowanie wątkow
-✦Klasy: SCHED_FIFO, SCHED_RR , SCHED_OTHER
-Polityki: PTHREAD_SCOPE_PROCESS –planowanie PCS, PTHREAD_SCOPE_SYSTEM –
planowanie SCS (na systemach M:M tworzy LWP dla każdego ULT t.j. efektywnie mamy 1:1)
-Pthread API pozwala programiście na zmianę priorytetu wątku (librt)

7.17. Wymien metody oceny algorytmow planowania.

- modelowanie deterministyczne - przyjmuje się konkretne, z gory określone obciążenie robocze
systemu i definiuje zachowanie algorytmu w warunkach tego obciążenia. Jest to odmiana oceny
analitycznej
- modele obsługi kolejek – analiza obsługi kolejek w sieciach
- symulacja sterowana rozkładami - ma ograniczoną dokładność

7.18. Opisz planowanie w systemie Solaris.

-4 klasy: real time, system, time sharing, interactive
-Priorytet globalny i priorytety w obrębie klas
-Proces potomny dziedziczy klasę i priorytet
-Klasa domyślna: time sharing, dynamicznie zmieniane priorytety
-Klasa interactive: wyższy priorytet dla aplikacji X-ow
-Klasa system -procesy jądra (np. planista, pagedaemon)
-Klasa real time ma najwyższy priorytet
-Planista wylicza priorytet globalny dla wątkow z danych klas i CPU wykonuje wątek aż
-wątek się zablokuje
-wątek zużyje swoj kwant czasu
-wątek zostanie wywyłaszczony przez wątek o wyższym priorytetcie
-jeśli dwa wątkie mają taki sam priorytet to planista stosuje RR

7.19. Wymien polecenia planowania procesow w systemie Solaris.

-vmstat przedział[licznik]
-dispadmin
-priocntl -d -i all

7.20. Opisz planowanie w systemie Windows 2000.

-Dyspozytor: planowanie priorytetowe z wywłaszczeniami [0..31]
-Wątek jest wykonywany aż zostanie wywłaszczony przez proces o wyższym priorytecie, zakończy, zużyje
kwant czasu, wykona blokujące wywołanie systemowe (np. we/wy)
-klasa czasu rzeczywistego [16..31] i klasa zmienna [0..15]
-Priorytety podstawowe (normatywne)
-czas rzeczywisty –24
-wysoki -13
-nadnormatywny –10
-normatywny –8 (Domyślny)
-podnormatywny –6
-idle (postojowy) -4
-Wątek w klasie zmiennej po wyczerpaniu czasu
-ma obniżany priorytet (jest degradowany) nie niżej niż priorytet podstawowy
-gdy zakończy czekanie (we/wy) dyspozytor podwyższa priorytet (wątek jest awansowany)
-zapewnia to dobry czas odpowiedzi dla aplikacji okienkowych oraz nie nadużycie czasu procesora
przez procesy ograniczone przez CPU
-Procesy pierwszo(drugo)planowe
-Proces wybrany na ekranie staje się pierwszoplanowy i ma zwiększany trzy razy kwant czasu, co
zapewnia mu dłuższe działanie

7.21. Opisz planowanie w systemie UNIX 4.3 BSD.

-Algorytm priorytetowy wielopoziomowych kolejek ze sprzężeniem zwrotnym wraz RR(q=1s)
-dobry czas reakcji dla użytkownikow interakcyjnych
-zadania o niskim priorytetecie nie są głodzone
-Priorytet każdego procesu jest obliczany raz na sekundę
-Grupy procesow (wymienione według malejącego poziomu uprzywilejowania)
-program wymiany (swapper)
-sterowanie urządzeniami blokowymi we/wy
-zarządzanie plikami
-sterowanie urządzeniami znakowymi we/wy
-procesy użytkownika
-Priorytet podstawowy dzieli wszystkie procesy na grupy o określonym priorytetcie

8.1. Wymień i scharakteryzuj kategorie ziarnistości.

Rozmiar ziarna	Opis	Częstotliwość synchronizacji (instrukcje)
Drobne	Paralelizm w jednym strumieniu instrukcji	<20
Średnie	Przetwarzanie równoległe albo wielozadaniowość w ramach jednej aplikacji	20-200
Grube	Wieloprzetwarzanie współbieżnych procesów w środowisku wieloprogramowym	200-2000
Bardzo grube	Przetwarzanie rozproszone w wielu węzłach sieci tworzących jedno środowisko obliczeniowe	2000-1M
Niezależne	Wiele niezwiązanych ze sobą procesów	Nie dotyczy

8.2. Wymień i scharakteryzuj techniki szeregowania wątkow na wielu procesorach.

✦Współdzielenie obciążenia
✦procesy nie są przydzielane konkretnemu procesorowi
✦Szeregowanie grupowe lub zespołowe (szeregowanie zbiorow)
✦zbior powiązanych wątkow jest jednocześnie kierowany do wykonania przez zbior wielu procesorow, na zasadzie jeden do jednego
✦Rezerwacja procesorow (dedykowane przydzielenie procesora)–przeciwieństwo współdzielenia
✦wątkom są przydzielane procesory, na zasadzie jeden do jednego
✦Szeregowanie dynamiczne
✦liczba wątkow w procesie może się zmieniać w czasie wykonania

8.3. Wymien i scharakteryzuj klasy zadan czasu rzeczywistego.

✦twarde zadanie czasu rzeczywistego musi dotrzymać wyznaczonego terminu
✦miękkie zadanie czasu rzeczywistego może nie dotrzymać wyznaczonego terminu
✦zadanie nieokresowe ma określony termin rozpoczęcia {lub | i} zakończenia
✦zadanie okresowe ma określony termin rozpoczęcia {lub | i } zakończenia „dokładnie co T jednostek” lub
„raz na okres T”

8.4. Scharateryzuj sytuacje odwrocenia priorytetow.

✦sytuacja, gdy proces wysokopriorytetowy czeka na zakończenie procesu o niższym priorytecie nosi nazwę
odwrocenia priorytetow
✦może mieć miejsce przy szeregowaniu priorytetowym z wywłaszczaniem
✦jeśli czas trwania odwrocenia priorytetow zależy od nieprzewidywalnych działań innych procesow
to mamy nieograniczone odwrocenie priorytetow

8.6. Scharaketryzuj pojecia odwrocenia i dziedziczenia priorytetow .
✦Unikanie nieograniczonego odwrocenia priorytetow
✦protokoł dziedziczenia priorytetow -w czasie użycia zasobow proces dziedziczy wysoki priorytet
(po użyciu stary priorytet jest przywracany)
✦pułap priorytetow–przydzielenie zasobu skutkuje zmianą na priorytet o jeden wyższy od najwyższego priorytetu w systemie; zwolnienie zasobu przywraca stary priorytet
8.8. Wymień i scharakteryzuj algorytmy szeregowania czasu rzeczywistego
✦Statyczne algorytmy tabelaryczne
✦na podstawie statycznej analizy wykonalnych planow szeregowania buduje się harmonogram
określający, kiedy zadanie ma się wykonywać
✦stosuje się do zadań okresowych
✦najpierw najwcześniejszy termin
✦Statyczne algorytmy priorytetowe z wywłaszczaniem
✦analiza statyczna wykonalnych planow na podstawie, ktorej przydziela się zadaniom priorytety i
stosuje się tradycyjnego planistę priorytetowego z wywłaszczaniem
✦algorytm częstotliwościowo monotoniczny
✦Dynamiczne algorytmy z planowaniem
✦wykonalność ustala się w czasie działania
✦przybywające zadanie jest akceptowane tylko wtedy gdy da się dotrzymać terminu jego wykonania
✦Dynamiczne algorytmy „rob co w twej mocy”
✦nie przeprowadza się analizy wykonalności (zadanie zwykle nieokresowe)
✦system stara się dotrzymać wszystkich terminow i przerywa każdy proces, ktory przekroczył
termin
8.10. Podaj warunek konieczny na dotrzymanie twardych terminow dla zadan okresowych.
✦aby możliwe było dotrzymanie twardych terminow musi być spełniona nierowność:
C1 /T1 + C2/T2 + ... + Cn/Tn ≤1
8.11. Opisz algorytm RMS i podaj gorne oszacowanie na wykorzystanie CPU.
Szeregowanie częstotliwościowe monotoniczne (ang. **R**ate **M**onotonic **S**cheduling) -zadanie o
krotszym okresie ma wyższy priorytet.
Wykorzystanie procesora dla RMS: **C1 /T1 + C2/T2 + ... + Cn/Tn ≤n(21/n -1) _ln 2 ≈ 0,6931**
9.1. Opisz szkodliwa rywalizacje (ang. race condition).
✦Sytuacja, w ktorej kilka procesow współbieżnie wykonuje działania na tych samych danych i w
konsekwencji wynik tych działań zależy od porządku, w jakim one następowały nazywa się szkodliwą
rywalizacją (lub sytuacją wyścigow)

9.2. Opisz problem sekcji krytycznej.
✦W systemie działają jednocześnie procesy: P0 ,P1 ,P2, . . ., Pn-1
✦Każdy proces ma segment kodu, zwany sekcją krytyczną, w ktorym może zmieniać wspolne zmienne
wykorzystywane przez P0,P1, P2, . . ., Pn-1
✦Gdy proces Pi, i=0..n-1, wykonuje się w swojej sekcji krytycznej to proces Pk, gdzie k≠i, nie może
wykonywać się w swojej sekcji krytycznej
✦Wykonywanie sekcji musi podlegać wzajemnemu wykluczaniu w czasie
9.3. Wymien i scharakteryzuj warunki sekcji krytycznej.

✦ (E): Wzajemne wykluczanie: jeśli proces Pi działa w swojej sekcji krytycznej, to żaden inny proces nie
działa w swojej sekcji krytycznej
✦(P): Postęp: jeśli żaden proces nie działa w swojej sekcji krytycznej i jakieś procesy chcą wejść do swoich
sekcji krytycznych wowczas kandydatem do wejścia do swojej sekcji krytycznej może być jedynie proces,
ktory nie wykonuje swojej reszty i wybor ten nie może być odwlekany w nieskończoność
✦(W): Ograniczone czekanie: jeśli dany proces zgłosił chęć wejścia do swojej sekcji krytycznej to musi
istnieć graniczna liczba wejść innych procesow do swoich sekcji krytycznych po ktorej dany proces uzyska
pozwolenie na wejście do swojej sekcji krytycznej

9.6. Scharaketryzuj sprzetowe srodki synchronizacji.

✦Wiele komputerow posiada sprzętowe wsparcie do rozwiązywania problemow sekcji krytycznej
✦W środowisku jednoprocesorowym można zakazać (zamaskować) przerwania w trakcie modyfikowania
zmiennej dzielonej
✦gwarantuje to brak wywłaszczeń tzn. gwarantuje sekwencyjność wykonywania się kodu
✦rozwiązanie to jest nieprzydatne w środowsku wieloprocesorowym ze względu na konieczność
przekazywania komunikatow do pozostałych procesorow
✦Wspołczesne komputery posiadają rozkazy maszynowe wykonywane w sposob niepodzielny
(nieprzerywalny, ang. atomically)
✦testuj pamięć i ustaw (TestAndSet)
✦zamień zawartość dwoch słow w pamięci (Swap)

9.7. Opisz rozwiazanie problemu sekcji krytycznej przy pomocy TestAndSet.

✦Struktury o wartościach początkowych false :boolean lock; boolean waiting[n];
✦Warunki (E),(P)
✦proces Pi wchodzi do swojej sekcji krytycznej ✦(waiting[i] = = false albo key = = false)
✦zmienna key może przyjąć wartość false tylko wskutek wykonania rozkazu TestAndSet
✦pierwszy proces wykonujący rozkaz TestAndSet zastanie key = = false
✦zmienna waiting[i] będzie miała wartość false tylko wtedy, jeśli inny proces opuści swoją sekcję
krytyczną i zostanie ona nadana tylko jednej zmiennej waiting[i]
✦proces wychodzący z sekcji krytycznej przypisuje wartość false albo zmiennej lock albo zmiennej
waiting[j] dzięki czemu proces czekając na wejście do swojej sekcji krytycznej może kontynuować
pracę (warunek (P))
✦Warunek (W)
✦gdy proces Pi opuszcza swoją sekcję krytyczną, wowczas cyklicznie przegląda tablicę waiting w
porządku (i+1,...n-1,0,..i-1)
✦zgodę na wejście do swojej sekcji krytycznej uzyska pierwszy proces przebywający w sekcji
wejściowej (waiting[j] = = true) a więc po nie więcej niż n-1 jednostkach oczekiwania

9.8. Scharakteryzuj semafory.

Semafor zliczający (definicja)

✦Semafor zliczający (ang. counting) można inicjalizować tylko za pomocą liczby dodatniej
✦Operacja wait powoduje zmniejszenie wartości semafora
✦jeśli wartość semafora stanie się wartością ujemną to dojdzie do zablokowania operacji wait

✦w przeciwnym razie wykonywanie procesu jest kontynuowane
✦Operacja signal powoduje zwiększenie wartości semafora
✦jeśli wartość semafora jest nie większa od zera to proces zablokowany przez operację wait zostaje
odblokowany

Semafor binarny (definicja)

✦Semafor binarny można inicjalizować za pomocą 0 lub 1
✦Operacja wait sprawdza wartość semafora
✦jeśli wynosi ona 0 to proces wykonujący wait zostaje zablokowany
✦jeśli wynosi ona 1 to wartość semafora zmieniana jest na 0 i proces kontynuuje wykonanie
✦Operacja signal sprawdza czy jakieś procesy nie czekają pod semaforem
✦jeśli tak to proces zablokowany przez wait zostaje odblokowany
✦jeśli żaden proces nie jest zablokowany to wartość semafora zostaje ustawiona na 1

9.13. Scharakteryzuj monitory.

✦Typ monitor składa się z deklaracji zmiennych określających stan obiektu oraz treści procedur
realizujacych działania na tym obiekcie
✦W danej chwili we wnętrzu monitora może być aktywny tylko jeden proces
✦Procedura wewnątrz monitora może korzystać tylko ze zmiennych zadeklarowanych w nim lokalnie
✦Programista nie musi jawnie kodować barier synchronizacyjnych

9.14. Scharakeryzuj zmienne warunkowe.

✦Aby monitor mogł synchronizować potrzebna jest konstrukcja pod nazwą warunek
✦conditionx, y;
✦Operacje na zmiennej typu condition:
✦x.wait () –proces wywołujący tę operację zostaje zawieszony do czasu, aż inny proces wywoła
operację x.signal();
✦x.signal () –wznawia dokładnie jeden z zwieszonych procesow (ktory wywołał x.wait ()) a jeśli nie
ma takiego procesu to operacja ta nie ma żadnych skutkow

10.1. Podaj przykład kodu programu z impasem i bez impasu.

Przykład kodu

```
typedef int semaphore;
 sempahore resource_1;
 sempahore resource_2;
void process_A(void) {
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_2);
   up(&resource_1);
}
void process_B(void) {
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_2);
   up(&resource_1);
}
```

kod bez impasu

```
typedef int semaphore;
 sempahore resource_1;
 sempahore resource_2;
void process_A(void) {
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_2);
   up(&resource_1);
}
void process_B(void) {
   down(&resource_2);
   down(&resource_1);
   use_both_resources();
   up(&resource_1);
   up(&resource_2);
}
```

kod z impasem

10.2. Podaj definicje impasu.

Def. Zbior procesow jest w stanie impasu, gdy każdy proces z tego zbioru czeka na zdarzenie, ktore może
być spowodowane tylko przez inny proces z tego samego zbioru

10.3. Kiedy moze wystapic impas w systemie?

✦Do impasow może dochodzić wtedy, kiedy w systemie zachodzą jednocześnie cztery warunki
✦wzajemne wykluczanie
✦przetrzymywanie i oczekiwanie
✦brak wywłaszczeń
✦czekanie cykliczne

10.4. Podaj przykład grafu przydziału zasobow z cyklem i bez impasu.

✦P= {P1, P2, …, Pn}, zbior wszystkich procesow systemu
✦R= {R1, R2, …, Rm}, zbior wszystkich typow zasobow w systemie

10.5. Opisz sposob zapobiegania impasom.

✦Zapewnić, aby nie mogł wystąpić przynajmniej jeden z czterech warunkow koniecznych:
✦Wzajemne wykluczanie -dotyczy jedynie zasobow niepodzielnych (np. drukarek); nie można zaprzeczyć
temu warunkowi, bowiem niektore zasoby są z natury niepodzielne
✦Przetrzymywanie i oczekiwanie -aby zapewnić, że warunek ten nigdy nie wystąpi w systemie trzeba
zagwarantować, że gdy proces zamawia zasob to nie powinien trzymać innych zasobow
✦każdy proces zamawia i otrzymuje wszystkie swoje zasoby przed rozpoczęciem działania
✦proces może zamowić zasob o ile nie ma żadnych zasobow
✦zanim proces zamowi zasob musi wpierw wszystkie oddać
✦Brak wywłaszczeń -trzeba zapewnić, że zasoby nie ulegają wywłaszczeniu jedynie z inicjatywy
przetrzymującego je procesu

✦Czekanie cykliczne –sposobem, aby warunek ten nigdy nie wystąpił w systemie jest przyporządkowanie
wszystkim zasobom danego typu kolejnych liczb i żądanie, aby
✦każdy proces zamawiał zasoby we wzrastającym porządku numeracji
✦alternatywnie można wymagać, aby proces zamawiający zasob miał zawsze zwolnione zasoby o
numeracji wyższej od zamawianego

10.6. Opisz algorytm unikania impasu dla zasobow reprezentowanych jednokrotnie.

- wymaga dodatkowej informacji o tym jak będzie następowało zamawianie zasobow
- w najprostszym i najbardziej użytecznym modelu zakłada się, że system zadeklaruje maksymalną liczbę
zasobow każdego typu, ktorą mogłby potrzebować
- algorytm unikania impasu (ang. deadlockavoidance) sprawdza dynamicznie stan przydziału zasobow aby
zapewnić, że nigdy nie dojdzie do spełnienia warunku czekania cyklicznego
- stan przydziału zasobow (ang. resourceallocation state) jest określony przez liczbę dostępnych (ang.
available) i przydzielonych (ang. allocated) zasobow oraz przez maksymalne zapotrzebowanie procesow

10.7. Podaj definicje stanu bezpiecznego.

- kiedy proces żąda wolnego zasobu system musi zadecydować czy przydzielenie zasobu pozostawi system
w stanie bezpiecznym
- system jest w stanie bezpiecznym, jeśli istnieje taki porządek przydzielania zasobow każdemu procesowi,
ktory pozwala na uniknięcie impasu
- system jest w stanie bezpiecznym, jeśli istnieje bezpieczny ciąg procesow
- system jest w stanie zagrożenia jeśli nie jest w stanie bezpiecznym
- ciąg procesow <P1, P2, …, Pn> jest bezpieczny jeśli dla każdego procesu Pi , jego potencjalne
zapotrzebowanie na zasoby może być zaspokojone przez bieżąco dostępne zasoby + zasoby użytkowane
przez wszystkie procesy Pj , przy czym j<i
- jeśli Pi żąda zasobow, ktore nie są natychmiast dostępne, to Pi może poczekać aż skończą się wszystkie
procesy Pj
- jeśli Pj skończą, Pi może otrzymać wszystkie potrzebne zasoby, dokończyć pracę, zwolnić zasoby i
zakończyć
- jeśli Pi zakończy, Pi+1 może otrzymać niezbędne zasoby itd.

10.8. Opisz algorytm bankiera.

✦n -liczba procesow w systemie; m-liczba typow zasobow
✦Dostępne (ang. Available) : wektor długości m. Jeśli Available[j] = k, to jest k egzemplarzy zasobu typu
Rj dostępnych w systemie
✦Maksymalne (Max) : macierz n x m. Jeśli Max[i,j] = k, to proces Pi może zażądać co najwyżej k
egzemplarzy zasobu typu Rj.
✦Przydzielone (ang. Allocation) : macierz n x m. Jeśli Allocated[i,j] = kto proces Pi ma przydzielonych k
egzemplarzy zasobu o typie Rj.
✦Potrzebne (ang. Need): macierz n x m. Jeśli Need[i,j] =k, to proces Pi może jeszcze potrzebować k
dodatkowych egzemplarzy zasobu typu Rj aby zakończyć
Need[i,j]= Max[i,j] –Allocation[i,j].

10.9. Opisz algorytm zamawiania zasobow.
Requesti - wektor żądań Pi. Jeśli Requesti[j] == k to Pi chce k egzemplarzy zasobu typu Rj
1.Jeśli Requesti ≤ Needi goto 2. W przeciwnym razie błąd
2.Jeśli Requesti ≤ Available, goto 3. W przeciwnym razie Pi czeka
3.System probuje zaalokować zasoby Pi modyfikując stan
Available= Available –Requesti ;
Allocationi= Allocationi+ Requesti ;
Needi=Needi–Requesti ;
•Jeśli stan safe ✦alokuj zasoby do Pi .
•Jeśli stan unsafe ✦Pi musi czekać na realizację i poprzedni stan przydziału zasobow jest odtwarzany
10.10. Posługujac sie algorytmem bankiera okresl czy system jest w stanie bezpiecznym.
Jesli proces *P*? złozy zamowienie (?, ?, ?, ?) to czy jest mozliwe jego spełnienie?
Ćwiczenia
10.11. Opisz algorytm wykrywania impasu dla zasobow reprezentowanych jednokrotnie.
✦Tworzymy graf oczekiwania
✦węzły grafu są procesami
✦Pi→Pj gdy Pi czeka na Pj.
✦Okresowo wykonujemy algorytm szukający cyklu w grafie oczekiwania
✦Rząd algorytmu wykrywania cykli w grafie oczekiwania wynosi n2, przy czym n jest liczbą wierzchołkow grafu
10.12. Opisz algorytm wykrywania impasu dla zasobow reprezentowanych wielokrotnie.
✦Metoda grafu oczekiwania jest bezużyteczna do wykrywania impasu, gdy każdy typ zasobu ma wiele egzemplarzy
✦Algorytm wykrywania impasu bada czy istnieje ciąg bezpieczny dla procesow, ktore trzeba dokończyć
✦korzysta ze struktur danych algorytmu bankiera
✦Available: wektor długości m oznacza liczbę dostępnych zasobow każdego typu
✦Allocation: macierz n x m definiuje liczbę zasobow każdego typu aktualnie zaalokowanych do każdego z procesow
✦Request: macierz n x m określa bieżące zamowienie każdego procesu. Jeśli Request [i,j] = k, to proces Pi zamawia dodatkowo k egzemplarzy zasobu typu Rj.
10.13. Opisz sposoby likwidowania impasu.
✦Zaniechanie (abort) wszystkich zakleszczonych procesow
✦Usuwanie procesow (wywłaszczanie) pojedyńczo, aż do wyeliminowania cyklu impasu
✦Usuwanie procesow (wywłaszczanie)
✦Wycofanie (ang. rollback) -wycofanie procesu do bezpiecznego stanu, od ktorego można go będzie wznowić
✦Głodzenie -ten sam proces może być zawsze ofiarą, podobnie jak i ten sam proces może być ciągle wycofywany. Trzeba zadbać, aby proces mogł być delegowany do roli ofiary tylko skończoną liczbę razy
✦Podejście mieszane
11.1. Wymień etapy przetwarzania programu uzytkownika.
- Czas kompilacji - Czas ładowania - Czas wykonania
11.2. Opisz działanie MMU.

✦Urządzenie sprzętowe dokonujące odwzorowania adresow wirtualnych na fizyczne nazywa się MMU
✦Istnieją wiele sposobow odwzorowywania adresow
✦Proste MMU
✦do każdego adresu wytwarzanego przez proces użytkownika dodawana jest wartość rejestru
przemieszczenia w chwili odwoływania się do pamięci
✦MSDOS używa czterech rejestrow przemieszczenia (architektura Intel 80x86)
✦program użytkownika nigdy nie ma do czynienia z rzeczywistym adresem; program ten działa na adresach
logicznych

11.3. Opisz rożnice miedzy adresacją logiczna i fizyczna.

- logiczny adres – wygenerowany przez CPU; jeśli odwzorowany na adres fizyczny podczas wykonywania
programu wtedy jest to wirtualny adres
- fizyczny adres - adres widziany przez sterownik pamięci
- adres logiczny i fizyczny jest taki sam podczas kompilacji i ładowania; logiczny (wirtulany) i fizyczny
adres rożnią się podczas wykonania

11.4. Opisz zasadę konsolidacji dynamicznej.

- konsolidację opoźnia się do czasu wykonania
- w obrazie binarnym, w miejscu odwołania bibliotecznego znajduje się tylko namiastka (ang. stub)
procedury będąca małym fragmentem kodu, wskazującym jak odnaleźć odpowiedni, rezydujący w pamięci
podprogram biblioteczny lub jak załadować bibliotekę, jeśli podprogramu nie ma w pamięci
- namiastka wprowadza na swoje miejsce adres podprogramu i go wykonuje
- system operacyjny sprawdza podprogram czy jest w pamięci a jeśli nie ma to go sprowadza
- biblioteki dzielone wymagają wspomagania ze strony systemu operacyjnego, niektore systemy realizują
jedynie konsolidację statyczną

11.6. Wymień sposoby jak na podstawie listy wolnych dziur spełnić zamowienie procesu na pamięć.

- pierwsze dopasowanie: (ang. first-fit) – wybierz pierwszy wolny wystarczająco duży blok
- najlepsze dopasowanie: (ang. best-fit) – wybierz najmniejszy wystarczająco duży blok
- najgorsze dopasowanie: (ang. worst-fit) – wybierz największy dostępny blok pamięci

11.7. Wyjaśnij rożnice miedzy fragmentacja zewnetrzna i wewnetrzna.

- fragmentacja zewnętrzna - suma wolnych obszarow w pamięci wystarcza na spełnienie zamowienia, ale
nie tworzą one ciągłego obszaru – opisuje stan przed przydzieleniem pamięci
- fragmentacja wewnętrzna - zaalokowana pamięć jest nieznacznie większa od żądania alokacji pamięci;
rożnica ta stanowi bezużyteczny kawałek pamięci wewnątrz przydzielonego obszaru – opisuje stan po
przydzieleniu

11.8. Opisz *paging*.

- logiczna przestrzeń adresowa procesu może być nieciągła tj. procesowi można przydzielać dowolne
dostępne miejsca w pamięci fizycznej
- pamięć fizyczną dzieli się na bloki stałej długości zwane ramkami
(rozmiar jest potęgą 2, między 512B a 16MB)
- pamięć logiczną dzieli się na bloki tego samego rozmiaru zwane stronami
- pamiętana jest lista wolnych ramek
- wykonanie procesu o rozmiarze n stron wymaga znalezienia n wolnych ramek i załadowanie w nie procesu

- utworzenie tablicy stron do odwzorowywania adresow logicznych na fizyczne
- eliminuje się fragmentację zewnętrzna, ale może powstać fragmentacja wewnętrzna

11.9. Podaj schemat translacji adresu przy stronicowaniu.

- stronicowanie wymaga wsparcia sprzętowego
- adres wygenerowany przez CPU jest dzielony na dwie części:
- numer strony (p) – używany jako indeks w tablicy stron zawierającej adresy bazowe wszystkich stron w
pamięci fizycznej
- odległość na stronie (d) – w połączeniu z adresem bazowym definiuje fizyczny adres pamięci posyłany do
jednostki pamięci

11.10. Opisz wykorzystanie TLB w stronicowaniu.

✦Pamięć skojarzeniowa: (klucz, wartość)
Translacja adresu(A´, A´´)
✦jeśli klucz (znacznik) obiektu (numer strony) A´ jest w TLB to weź
odpowiadającą mu wartość A´´(numer ramki)
✦w przeciwnym razie weź numer ramki z tablicy stron
✦Zasady zastępowania –zastąp najdawniej używany element (LRU, niezbędne wsparcie sprzętowe)
✦Niektore z pozycji TLB mogą być nieusuwalne i wtedy nazywamy je przypiętymi
✦ASID –identyfikatory w TLB jednoznacznie określające związany z daną pozycją proces
✦Jeśli bufor TLB nie udostępnia odrębnych identyfikatorow ASID to przy wyborze nowej tablicy stron (np.
przełączenie kontekstu) bufor TLB musi zostać wykasowany

11.11. Podaj wzor na efektywny czas dostepu do pamieci.

- przeglądnięcie rejestrow asocjacyjnych = ε jednostek czasu
- niech cykl pamięci wynosi 1 mikrosekundę
- współczynnik trafień - procent numerow stron odnajdowanych w rejestrach asocjacyjnych; współczynnik
zależy od liczby rejestrow asocjacyjnych. Współczynnik trafień = α
- Effective Access Time (EAT)
- EAT = (1 + ε) α + (2 + ε)(1 - α) = 2 + ε - α

11.12. Opisz sposob ochrony pamieci przy stronicowaniu.

✦Ochrona pamięci jest zaimplementowana za pomocą bitow ochrony przypisanych każdej ramce
✦Bit poprawności -każdy wpis w tablicy stron zostaje uzupełniony o dodatkowy bit:
✦“poprawny” oznacza, że strona, z ktorą jest on związany, znajduje się w logicznej przestrzeni
adresowej procesu, a więc jest ona dozwolona
✦“niepoprawny” oznacza, że strona nie należy do logicznej przestrzenia adresowej procesu tzn. nie
poprawne odwołanie do strony

11.13. Podaj schemat translacji adresu przy stronicowaniu dwupoziomowym.

- adres logiczny = <p1,p2,d>
- p1 – indeks do zewnetrznej tablicy stron
- p2 – przesuniecie na stronie zewnetrznej tablicy stron
- d - odleglosc na stronie
- adres fizyczny = ((zewnetrzna tablica stron[p1])[p2])+d
- oczywiście przy każdym odniesieniu należy sprawdzić czy nie przekroczone zostały granice.

11.14. Podaj schemat translacji adresu przy haszowanej tablicy stron.

- przestrzeń adresowa > 32 bitow
- numer strony pamięci wirtualnej jest odwzorowany (ang. hashed) przy pomocy funkcji haszującej na
pozycje w tablicy (ang. hashed page table)

- wszystkie strony wirtualne ktorym odpowiada ta sama pozycja w tablicy (kolizja) zostają umieszczone na
jednej liście (metoda łańcuchowa)
- element listy: nr strony wirtualnej (p), nr strony pamięci (r), wskaźnik do następnego elementu listy

11.15. Opisz haszowanie liniowe i łancuchowe.

haszowane tablice stron

- przestrzeń adresowa > 32 bitow
- numer strony pamięci wirtualnej jest odwzorowany przy pomocy funkcji haszującej na pozycje w tablicy
- wszystkie strony wirtualne, ktorym odpowiada ta sama pozycja w tablicy (kolizja) zostają umieszczone na
jednej liście (metoda łańcuchowa)
- element listy: nr str. wirtualnej (p), nr str. pamięci (r), wskaźnik do następnego elementu listy

11.16. Opisz algorytm wyszukiwania w haszowanej tablicy stron.

✦Numer strony wirtualnej jest odwzorowany przy pomocy funkcji mieszającej na element listy powiązanej
w tablicy stron: (q, s, *), (p, r, *),....
✦Numer strony jest porownywany z pierwszym elementem (q) w tablicy
✦jeśli numer strony jest zgodny to używa się odpowiadającej mu ramki (s)
✦w przeciwnym razie wybieramy następny (*) adres, t.j. (p, r, *) ✦itd.

11.17. Podaj schemat translacji adresu przy odwroconej tablicy stron.

- odwrocona tablica stron ma po jednej pozycji dla każdej rzeczywistej strony pamięci (ramki)
- każda pozycja zawiera adres wirtualny strony przechowywanej w ramce rzeczywistej pamięci oraz
informacje o procesie posiadającym stronę
- zmniejsza się rozmiar pamięci potrzebnej do pamiętania wszystkich tablic stron, jednak zwiększa się czas
potrzebny do przeszukania tablicy przy odwołaniu do strony
- tablica haszowania – ogranicza szukanie do jednego lub najwyżej kilku wpisow w tablicy stron

11.18. Opisz schemat stron dzielonych.

- strony dzielone - dzielenie kodu
- jedna kopia kodu nie modyfikującego samego siebie tj. wznawialnego jest dzielona pomiędzy procesy
- kod dzielony musi być widziany pod tą samą lokacją w logicznej przestrzeni adresowej dla wszystkich
procesow
- kod prywatny i dane
- każdy proces ma własną kopie kodu i danych
- strony dla prywatnego kodu i danych mogą się pojawić w dowolnym miejscu logicznej przestrzeni
adresowej

11.19. Podaj schemat translacji adresu przy segmentacji.

✦ Adres logiczny składa się z dwu części : <numer-segmentu, odległość w segmencie>,
✦Tablica segmentow–jest wykazem par:
✦ bazy – zawiera początkowy fizyczny adres segmentu w pamięci
✦ granica – oznacza długość segmentu
✦ Rejestr bazowy tablicy segmentow (STBR) wskazuje na tablicę segmentow w pamięci
✦Rejestr długości tablicy segmentow (STLR) oznacza liczbę segmentow przypadających na program;
numer segmentu s jest poprawny jeśli s< STLR.

11.20. Opisz segmentacje z dwupoziomowym schematem stronicowania dla procesora *i386*.

✦Intel386 (i poźniejsze) stosuje segmentację ze stronicowaniem do zarządzania pamięcią z

dwupoziomowym schematem stronicowania
✦CPU generuje adreslogiczny
✦Maksymalna liczba segmentow w procesie 16K
✦Każdy segment mniejszy niż 4GB
✦Rozmiar strony 4KB
✦Segmentowanie jest opcjonalne
✦Przestrzeń adresowa ma dwie strefy zawierające po co najwyżej 8K segmentow
✦prywatne segmenty procesu przechowywane w tablicy lokalnych deskryptorow (64b)
✦wspolne segmenty procesow przechowywane w globalnej tablicy deskryptorow (64b)
✦Każdy adres logiczny (wirtualny) jest parą (selektor, odległość)
✦selektor: 16b liczba, odległość 32b liczba
✦Procesor ma 6 rejestrow segmentowych (16b) zawierających selektory do zaadresowania: segmentu
rozkazu (CS), segmentu stosu (SS) i 4 segmentow danych (DS, ES, FS, GS)
✦Procesor ma także pamięć podręczną w postaci 6 rejestrow mikroprogramowych (8B) do przechowywania
odpowiednich deskryptorow z tablicy LDT lub GDT
✦Adres fizyczny ma 32b
✦Każdy segment w i386 jest stronicowany
✦W procesorze 386 przyjęto stronicowanie dwupoziomowe
✦Tablice stron można odsyłać na dysk

11.22. Opisz polecenia do ustalania, monitorowania wielkosci pamieci i strony w systemie Solaris i Linux.

Ustalanie wielkości pamięci
✦ Linux
- free
✦ Solaris
- dmesg | grep mem

Monitorowanie wykorzystania pamięci
✦ ps –ayl (Solaris)
-używana pamięć fizyczna (wirtualna) w kolumnie RSS (SZ)
- udział w zużyciu pamięci w kolumnie %MEM
✦ ps vx (Linux)
- MAJFL – liczba braków stron

Wyświetlanie obszarów
stronicowania
✦ Linux
- cat /proc/swaps
- swapon -s ; free -m -o
✦ Solaris
- swap –l

12.1. Wymien zalety pamieci wirtualnej.

- brak ograniczeń na pamięć
- więcej programow – lepsze wykorzystanie procesora
- można jedynie część programu załadować do pamięci w celu wykonania (reszta nieużywana
(procedury obsługi rzadkich błędow) albo nadmiarowa (np. nadmiarowe tablice) jest w pamięci wirtualnej)
- logiczna przestrzeń adresowa może być większa niż fizyczna
- odseparowanie pamięci logicznej użytkownika od pamięci fizycznej,
- kopiowanie przy zapisie przy tworzeniu procesu

- odwzorowanie plikow do pamięci przy tworzeniu procesu

12. 2 Wymien sposoby implementacji pamieci wirtualnej.

- stronicowanie na żądanie – demand paging
- segmentacja na żądanie – demand segmentation

12.3. Opisz stronicowanie na zadanie.

●System stronicowania na żądanie jest podobny do stronicowania z wymianą
●Procedura leniwej wymiany
●nigdy nie dokonuje się wymiany strony w pamięci jeśli nie jest to konieczne
●mniej operacji we/wy
●mniej pamięci
●szybsza reakcja
●więcej użytkownikow
●Jeśli strona jest potrzebna odwołaj się
●niepoprawne odwołanie_abort
●brak strony w pamięci (sprowadź stronę do pamięci
●Zgodność z Zasadą Lokalności Odniesień

12.4. Opisz procedure obsługi braku strony.

- system operacyjny sprawdza wewnętrzną tablicę oraz decyduje że:
- jeśli odwołanie niedozwolone - kończy proces
- jeśli odwołanie dozwolone tylko zabrakło strony w pamięci to sprowadza tę stronę
- system znajduje wolną ramkę na liście wolnych ramek
- gdy nie ma wolnej ramki to szukamy strony w pamięci ktora nie jest używana i zapisujemy ją na dysk
- system wczytuje stronę z dysku do wolnej ramki
- system wstawia bit 1 w tablicy stron
- system wykonuje przerwany rozkaz

12.5. Podaj wzor na obliczenie sprawnosci stronicowania.

- EAT – efektywny czas dostępu
- p – prawdopodobieństwo braku strony (0 – brak braku stron , 1 - każde odwołanie generuje brak strony)
- cd - czas dostępu do pamięci
- cz - czas obsługi strony (obsługa przerawnia wywołanego brakiem strony, czytanie strony, wznowienie procesu)
- EAT = (1-p)*cd+p*cz

12.9. Opisz procedure zastepowania stron.

●1. Zlokalizowanie potrzebnej strony na dysku
●2. Odnalezienie wolnej ramki
●jeśli ramka istnieje -zostaje użyta
●w przeciwnym razie typowanie ramki ofiary
●ramka ofiara zapisana na dysk; zmień tablicę stron i tablicę ramek
●3. Wczytanie potrzebnej strony; zmień tablice stron i ramek
●4. Wznowienie procesu
●Gdy nie ma wolnych ramek -potrzebne jest dwukrotne przesyłanie stron a w konsekwencji wydłużenie efektywnego czasu dostępu

12.10. Opisz algorytm zastepowania stron FIFO.

- algorytm FIFO stowarzysza z każdą ze stron czas kiedy została ona sprowadzona do pamięci
- jeśli trzeba zastąpić stronę to zastępowana jest najstarsza ze stron
- implementuje się za pomocą kolejek FIFO

12.11. Skonstruuj przykład ilustrujacy anomalie Belady'ego.

- anomalia Belady'ego odzwierciedla fakt, że w niektorych algorytmach zastępowania stron współczynnik brakow stron może wzrastać ze wzrostem wolnych ramek

- przykład dla algorytmu FIFO:
- ciąg odniesień: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
- 3 ramki (3 strony mogą być w pamięci w tym samym czasie) - 9 brakow stron
- 4 ramki (4 strony mogą być w pamięci w tym samym czasie) – 10 brakow stron

12.12. Opisz algorytm zastepowania stron LRU.

- zastąp tę stronę, ktora najdawniej była użyta
- nie jest dotknięty anomalią Belady'ego
- odwracalność
- dwie implementacje
- liczniki – do każdej pozycji w tablicy stron dołączamy rejestr czasu użycia, do procesora zaś dodajemy
zegar logiczny lub licznik. Wskazania zegara są zwiększane wraz z każdym odniesieniem do pamięci.
Ilekroć występuje odniesienie do pamięci, tylekroć zawartość rejestru zegara jest kopiowana do rejestru
czasu użycia należącego do danej strony w tablicy stron
- stos – przy każdym odwołaniu do strony jej numer wyjmuje się ze stosu i umieszcza na szczycie -
najlepsza implementacja to dwukierunkowa lista ze wskaźnikami do czoła i do końca - najwyżej 6 zmian
wskaźnikow - nie jest potrzebne przeszukiwanie listy

12.13. Opisz sposoby implementacji algorytmu LRU.

●Liczniki -do każdej pozycji w tablicy stron dołączamy rejestr czasu użycia, do procesora zaś dodajemy
zegar logiczny lub licznik. Wskazania 'zegara' są zwiększane wraz z każdym odniesieniem do pamięci.
Ilekroć występuje odniesienie do pamięci, tylekroć zawartość rejestru zegara jest kopiowana do pola czasu
użycia należącego do danej strony w tablicy stron
●Stos -przy każdym odwołaniu do strony jej numer wyjmuje się ze stosu i umieszcza na szczycie -najlepsza
implementacja to dwukierunkowa lista ze wskaźnikami do czoła i do końca
●najwyżej 6 zmian wskaźnikow ●nie jest potrzebne przeszukiwanie listy

12.14. Opisz algorytm zastepowania stron OPT.

- zastąp tę stronę, ktora najdłużej nie będzie używana; nazywany OPT lub MIN
- nie ma anomalii Belady'ego
- bardzo trudny do realizacji, bo wymaga wiedzy o przyszłej postaci ciągu odniesień (podobnie jak przy
planowaniu procesora metodą SJF)
- używany głownie w studiach porownawczych
- wiedza o tym, że jakiś algorytm odbiega od optymalnego o 12,3% a średnio jest od niego gorszy o 4,7%
może okazać się cenną

12.15. Scharakteryzuj algorytmy stosowe.

●Def. Algorytm stosowy to taki algorytm dla ktorego zbior stron w pamięci w przypadku n ramek jest
podzbiorem zbioru stron w pamięci w przypadku n+1 ramek
●Przykład: LRU
●Własność: klasa algorytmow stosowych nie jest dotknięta anomalią Belady'ego
●Implementacja LRU wymaga wsparcia sprzętowego: uaktualnianie pol zegara lub stosu musi być
dokonywane przy każdym odniesieniu do pamięci
●Możemy nie mieć takiego sprzętu

12.16. Opisz algorytm dodatkowych bitow odniesienia.

●każda strona ma 8 bitowy rejestr w pamięci głownej

●cyklicznie (co 100ms) przerwanie zegarowe powoduje wywołanie procedury ktora powoduje wprowadzenie bitu odniesienia na najbardziej znaczącą pozycję rejestru oraz przesunięcie pozostałych w
prawo o 1 bit
●00000000 –strona nieużywana przez osiem cykli
●11111111 -strona używana co najmniej jeden raz w każdym cyklu
●interpretujemy rejestry jako liczby bez znaku, tzn. strona najdawniej użyta ma najmniejszą zawartość
rejestru

12.17. Opisz zegarowy algorytm drugiej szansy (CLOCK).

- bit odniesienia
- z każdą stroną stowarzyszamy na początku bit 0
- czytanie lub pisanie na stronie ustawia bit na 1
- zastąp stronę jeśli ma bit 0
- nie można poznać porządku użycia stron
- algorytm drugiej szansy
- algorytm FIFO (wymaga zegara)
- gdy strona (FIFO) ma bit odniesienia = 1 to strona dostaje drugą szansę na pobyt pamięci:
- bit odniesienia = 0 i czas przybycia = bieżący
- zostawia się stronę w pamięci
- zastępuje się następną w porządku FIFO stronę według powyższych zasad
- ulepszony algorytm drugiej szansy
- (x,y) - x - bit odniesienia, y- bit modyfikacji
- 4 klasy (od najniższej)
- (0,0) - nie używana ostatnio i nie zmieniana: najlepsza ofiara
- (0,1) - nie używana ostatnio ale zmieniona: gorsza ofiara bo wymaga zapisu na dysk
- (1,0) - używana ostatnio i czysta: może być wkrotce użyta
- (1,1) - używana ostatnio i zmieniana - najgorsza ofiara, prawdopodobnie będzie zaraz użyta
- algorytm drugiej szansy ale zastępujemy pierwszą napotkaną stronę z najniższej niepustej klasy (x,y)

12.18. Rozwazmy nastepujacy ciag odniesien:
?, ?
Ile brakow stron wystapi gdy mamy x, y lub z ramek dla algorytmow FIFO,
OPT, LRU, CLOCK ?

Zadanie liczone na cwiczeniach

12.19. Opisz ulepszony algorytm drugiej szansy.

●(x,y) -x -bit odniesienia, y-bit modyfikacji przypisany jest (w CPU) do każdej ramki
●4 klasy (od najniższej)
●(0,0) -nie używana ostatnio i nie zmieniana : najlepsza ofiara
●(0,1) -nie używana ostatnio ale zmieniona: gorsza ofiara bo wymaga zapisu na dysk choć zgodnie z
zasadą lokalności pewnie nie zostanie użyta
●(1,0) -używana ostatnio i czysta: jeszcze gorsza ofiara bo zgodnie z zasadą lokalności może być
wkrotce użyta
●(1,1) -używana ostatnio i zmieniana -najgorsza ofiara, prawdopodobnie (zgodnie z zasadą lokalności) będzie zaraz użyta a ponadto wymaga zapisu na dysku
●1. Skanowanie bufora ramek od pozycji wskaźnika
●bity nie są zmieniane; do wymiany pierwsza ramka (0,0)
●2. Jeśli w kroku 1 nie znaleziono ofiary to następuje skanowanie bufora w celu znalezienia ramki (0,1)
●do wymiany pierwsza znaleziona; bity odniesienia są w trakcie zerowane
●3. Jeśli w kroku 2 nie znaleziono ofiary to wskaźnik znajduje się w położeniu początkowym i wszystkie
bity odniesienia są wyzerowane

●powtarzamy krok 1 a w przypadku nie znalezienia ofiary krok 2
●znajdujemy ramkę ofiarę; wymieniamy stronę; ustawiamy wskaźnik na następną ramkę w buforze
●Ulepszony algorytm drugiej szansy w pierwszej kolejności zastępuje stronę, ktora nie była zmodyfikowana

12.21. Opisz schematy przydziału ramek.

●Przydział rowny -np. jeśli mamy 100 ramek i 5 procesow, to każdy proces może dostać 20 ramek
●Przydział proporcjonalny -każdemu procesowi przydziela się dostępną pamięć proporcjonalnie do jego
rozmiaru

14.11. Opisz parametry do mierzenia wydajnosci we/wy dla dysku.

- czytanie lub pisanie wymaga pozycjonowania głowicy dysku nad określoną ścieżką oraz na początku
określonego sektora
- czas przeszukiwania -czas potrzeby na ustawienie głowicy nad scieżką
- opoźnienie obrotowe - czas potrzebny głowicy na osiągnięcie sektora
- czas dostępu
- suma czasu przeszukiwania i opoźnienia obrotowego
- czas osiągnięcia stanu umożliwiającego zapis/odczyt
- odczyt/Zapis jest dokonywany w miarę przemieszczania się sektora pod głowicą

14.12. Podaj wzor na sredni czas dostepu dla dyskow.

- T_a = T_s + 1/2r + T
-T_s – średni czas przeszukiwania (5 do 10ms)
- r – prędkość w obrotach na min (10000 rpm)
- T - czas transferu
- T=b/r*N
- b - ilość transferowanych bajtow; N - ilość bajtow na ścieżkę
- przykład: 2650 sektorow = 512B*320S*8T = 1.3MB
- pierwsza ścieżka: 10ms+ 6/2ms + 6ms = 19ms
- następne ścieżki: 3ms + 6ms =9ms; T= 19ms+7*9ms=0.082s
- niesekwencyjnie: 2560*(10ms + 3ms + 6*512B/512B*320) = 2560*13.01875ms= 33.328s

14.14. Wymien algorytmy planowania we/wy dla dyskow.

- FIFO (ang First-in, First-out),
- PRI (ang. priority),
- LIFO (ang Last-in, First-out),
- SSTF (ang. Shortest service time first),
- SCAN,
- C-SCAN,
- N-step-SCAN,
- FSCAN

14.15. Opisz algorytm SSTF oraz jego własnosci.

✦realizuj to żądanie we/wy, ktore wymaga najmiejszego ruchu głowicy licząc od pozycji bieżącej
✦zawsze minimalny czas przeszukiwania
✦lepszy niż FIFO
✦możliwe głodzenie

14.16. Opisz algorytm SCAN oraz jego własnosci.

- głowice przemieszczają się w jednym kierunku do ostatniej ścieżki lub do ostatniego żądania w danym
kierunku
- kierunek zostaje następnie odwrocony
- neguje zasadę lokalności odniesień
- nie ma możliwości głodzenia
- SCAN faworyzuje zadania o żądaniach we/wy na skrajnych ścieżkach

176. Załózmy, ze głowica dysku znajduje sie w chwili *t*0 nad sciezka *s*0 oraz ze zarzadca we/wy otrzymał zadania dostepu do sciezek: *s*1, *s*2, . . . , *sn*. Dla danego ciagu zadan do sciezek porównaj metoda deterministyczna (tzn. oblicz srednia liczbe ruchów głowica) algorytmy: FIFO, SSTF, SCAN, CSCAN.

177. Jaka jest róznica miedzy polem, rekordem, plikiem a baza danych?

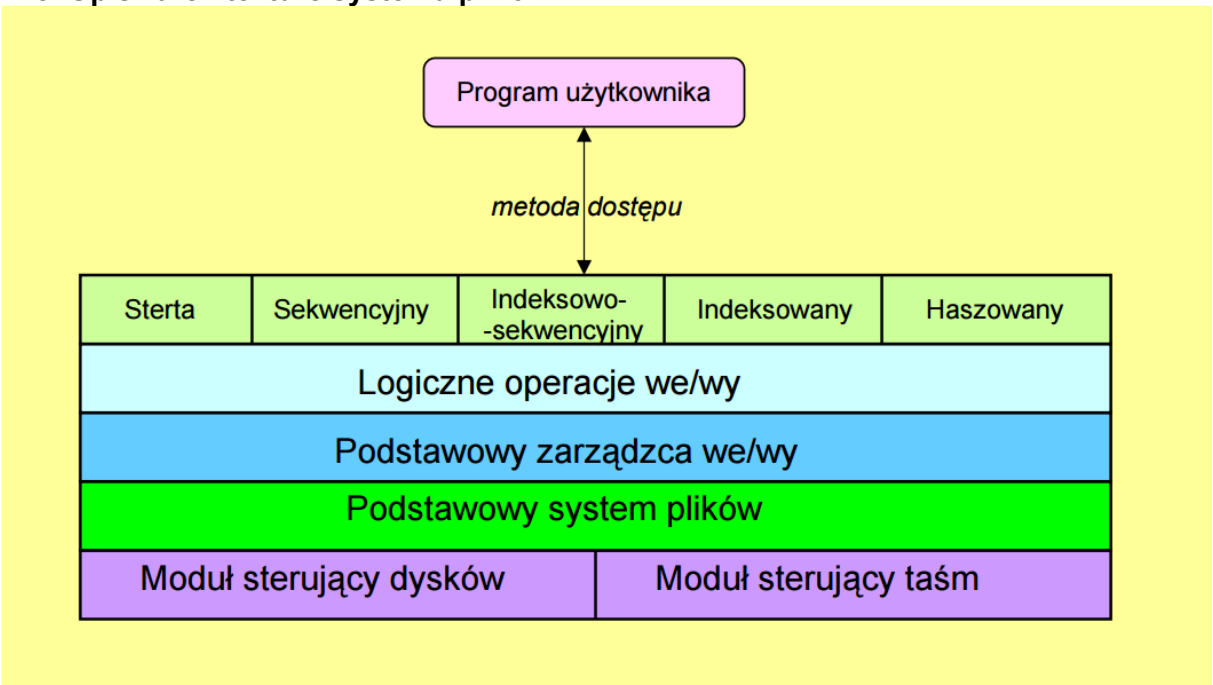
●Pole
- ●podstawowy element danych
- ●pojedyńcze pole zawiera jedną wartość (np. nazwisko)
- ●charakteryzuje się długością i typem danych (np. znaki ASCII)

●Rekord
- ●zbiór powiązanych pól
- ●traktowany przez aplikacje jako całość
- ●rekord pracownik zawiera pola: nazwisko, numer ubezpieczenia, datę zatrudnienia

●Plik
- ●zbiór podobnych rekordów
- ●traktowany przez użytkowników jako całość
- ●unikalna nazwa, może być tworzony i kasowany
- ●kontrola dostępu na poziomie pliku

●Baza danych
- ●zbiór bezpośrednio powiązanych ze sobą danych
- ●niezależny mechanizm do zarządzania

178. Opisz architekture systemu plików.



179. Scharakteryzuj organizacje sterty.

●dane są zapisywane w kolejności napływania
●proste kumulowanie danych oraz ich zapis
●rekordy mogą mieć różne pola lub podobne pola w różnej kolejności
●każde pole musi zawierać swój opis (nazwę, długość)
●brak struktury
●długie wyszukiwanie: jeśli chcemy znaleźć wszystkie rekordy, które zawierają pole z konkretną
zawartością to musimy przejrzeć wszystkie pliki
●taka organizacja pliku jest nieodpowiednia w większości przypadków

180. Scharakteryzuj organizacje sekwencyjna.
●rekordy o stałym formacie
●jednakowa długość rekordów
●wszystkie pola mają tą samą długość i kolejność
●nazwa każdego pola i jego długość są atrybutami pliku
●szczególne ważne pole to pole klucza
    ●jednoznacznie identyfikuje rekord
    ●rekordy są gromadzone zgodnie z porządkiem klucza
●nowe rekordy są zapamiętywane w pliku sterty, nazywanym plikiem dziennika lub transakcyjnym
●w trybie wsadowym nowe rekordy w dzienniku są okresowo złączane z plikiem głównym
●niewydajny w zastosowaniach interaktywnych związanych z kwerendami lub aktualizacjami rekordów

181. Scharakteryzuj organizacje indeksowo-sekwencyjna.
●indeks daje możliwość przeglądania, co pozwala na szybkie wyznaczanie otoczenia z poszukiwanym rekordem
    ●indeks jest plikiem sekwencyjnym, którego każdy rekord zawiera dwa pola –pole klucza oraz
wskaźnik do pliku głównego
●wyszukiwanie rekordu
    ●indeks jest przeszukiwany aż do znalezienia największej wartości klucza
    ●przeszukiwanie jest kontynuowane w pliku głównym od miejsca wyznaczonego przez wskaźnik
●każdy rekord w pliku głównym ma pole będące wskaźnikiem do pliku nadmiaru
●nowe rekordy są dodawane do pliku nadmiaru
●rekord w pliku głównym, który bezpośrednio poprzedza nowy rekord w sekwencji logicznej, aktualizowany jest tak, aby zawierał wskaźnik do nowego rekordu w pliku nadmiaru
●plik nadmiaru jest scalany w trybie wsadowym z plikiem indeksowo-sekwencyjnym
●dla zwiększenia efektywności dostępu stosuje się indeksowanie wielopoziomowe
●skraca znacznie czas potrzebny do dostępu do pojedynczego rekordu
182. Scharakteryzuj organizacje indeksowana.
●w przypadku wyszukiwania opartego na innej zasadzie niż klucz, metody sekwencyjna i indeksowo sekwencyjna są niewystarczające
●w pliku indeksowanym brak jest sekwencyjności i klucza: dostęp do rekordów poprzez indeksy wielokrotne
●po jednym indeksie dla każdego wyszukiwanego pola
●można stosować rekordy o zmiennej długości
●dwa typy indeksów
    ●pełny –po jednym zapisie dla każdego rekordu w pliku głównym
    ●częściowy –obejmuje tylko zapisy tych rekordów pliku głównego, które zawierają odpowiednie pole
●wprowadzenie nowego rekordu wymaga aktualizacji wszystkich indeksów
●stosowane w aplikacjach, gdy ważna jest szybkość

183. Wymień i scharakteryzuj techniki grupowania rekordów w bloki.
●Wykonywanie operacji we/wy wymusza grupowanie rekordów w blokach
●Problemy
    ●długość bloków: stała, zmienna
    ●wielkość bloków
●Rozwiązania
    ●bloki stałej długości
    ●bloki zmiennej długości z podziałem rekordu

●bloki zmiennej długości bez podziału rekordu

184. Wymień i scharakteryzuj metody alokacji plików.

Alokacja ciągła

●w chwili tworzenia pliku przydzielany jest jednorazowo pojedynczy, ciągły zestaw bloków

●FAT zawiera tylko jeden zapis wskazujący blok początkowy i długość pliku

●Problem fragmentacji zewnętrznej

●sporadyczne wykonanie algorytmu upakowania

Alokacja łańcuchowa

●realizowana na bazie pojedynczych bloków

●każdy blok zawiera wskaźnik do następnego bloku w łańcuchu

●FAT zawiera pojedynczy zapis

●blok początkowy i długość pliku

●Eliminacja fragmentacji zewnętrznej

●Najlepiej nadaje się do plików sekwencyjnych

●Neguje zasadę lokalności odniesień - okresowa konsolidacja

Alokacja indeksowana

●FAT zawiera oddzielny indeks jednopoziomowy dla każdego pliku

●indeks ma po jednym zapisie dla każdej porcji przydzielanej plikowi

●indeks jest rejestrowany w oddzielnym bloku

●FAT zawiera wskaźnik do numeru bloku indeksu

●Alokacja może by realizowana na bazie bloków stałej oraz zmiennej długości

●alokacja na podstawie bloku eliminuje fragmentację zewnętrzną

●alokacja na podstawie fragmentów o zmiennych rozmiarach zwiększa zastosowanie zasady

Lokalności odniesień