5.6 ОПИС ПРОЦЕСІВ

Операційна система управляє подіями, які відбуваються в комп'ютерній системі. Вона планує і координує виконання процесів, виділяє їм ресурси і надає за запитом системних і призначених для користувача програм основні сервіси. Можна представити ОС як деякий механізм, який керує тим, як процеси використовують системні ресурси.

Ця концепція проілюстрована на рис. 5.7. Нехай в багатозадачному середовищі є декілька процесів (P1, P2,.,Pn), які вже створені і завантажені у віртуальну пам'ять. Кожному процесу для його функціонування потрібний доступ до певних ресурсів. У ситуації, зображеній на рис. 5.7, процес P1 знаходиться в стані виконання, тобто в основній пам'яті знаходиться принаймні частина цього процесу. Крім того, він здійснює управління двома пристроями введення-виведення. Процес P2 теж знаходиться в основній пам'яті, але він блокований, чекаючи, поки звільнитися пристрій введення-виведення, зайнятий процесом P1. Процес Pn вивантажений з основної пам'яті і, відповідно, призупинений.

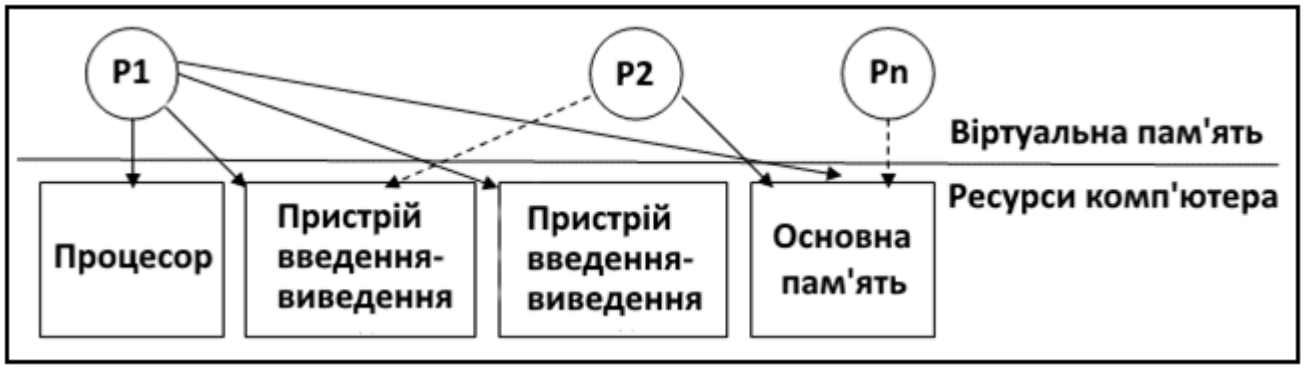

Рисунок 5.7 – Процеси і ресурси в певний момент часу

5.6.1 Управляючі структури ОС

Оскільки в задачі ОС входить управління процесами і ресурсами, вона повинна мати в розпорядженні інформацію про поточний стан кожного процесу і ресурсу. Універсальний підхід до надання такої інформації простий: ОС створює і підтримує таблиці з інформацією по кожному об'єкту управління. Хоча деталі таблиць в різних ОС можуть відрізнятися, по суті, усі ОС підтримують інформацію за чотирма категоріями. P1 P2

Таблиці пам'яті (Memory tables) використовуються для того, щоб стежити за основною (реальною) і вторинною (віртуальною) пам'яттю. Процеси, які знаходяться у вторинній пам'яті, використовують деякий різновид віртуальної пам'яті або простий механізм свопінгу. Таблиці включають таку інформацію:

- об'єм основної пам'яті, відведений процесу;
- об'єм вторинної пам'яті, відведений процесу;
- усі атрибути захисту блоків основної і віртуальної пам'яті;
- уся інформація, необхідна для управління віртуальною пам'яттю.

Детально ці інформаційні структури, які використовуються для управління пам'яттю, розглядатимемо пізніше в наступних розділах, а зараз коротко нагадаємо поняття «Віртуальної пам'яті».

Віртуальна пам'ять – це пристрій, що дозволяє програмістам розглядати пам'ять з логічної точки зору, не піклуючись про фізичну пам'ять достатнього об'єму. Принципи роботи з віртуальною пам'яттю були розроблені, щоб завдання декількох користувачів, виконуючись паралельно, могли одночасно бути присутніми в основній пам'яті. Із-за відмінностей в кількості пам'яті, що вимагається для різних процесів, при перемиканні процесора з одного процесу на інший важко компактно розмістити їх в основній пам'яті. Тому були розроблені системи із сторінковою організацією пам'яті, при якій пам'ять розбивається на блоки фіксованого розміру, що називаються сторінками. Звернення програми до слова пам'яті відбувається за віртуальною адресою, яка складається з номера сторінки і зміщення відносно її початку.

Сторінки одного і того ж процесу можуть бути розкидані по усій основній пам'яті. Система розбиття на сторінки забезпечує динамічну відповідність між віртуальною адресою, що використовується програмою, і реальною або фізичною адресою основної пам'яті.

Наступним логічним кроком розвитку в цьому напрямі було виключення вимоги, щоб усі сторінки процесу одночасно знаходилися в основній пам'яті. Досить, щоб усі вони зберігалися на диску. Під час виконання процесу тільки деякі його сторінки знаходяться в основній пам'яті. Якщо програма звертається до сторінки, яка там відсутня, апаратне забезпечення, що управляє пам'яттю, виявить це і організовує завантаження відсутніх сторінок. Така схема називається віртуальною пам'яттю.

Таблиця введення-виведення (I/O tables) використовується ОС для управління пристроями введення-виведення і каналами комп'ютерної системи. У кожен момент часу пристрій введення-виведення може бути або вільним, або відданий в розпорядження якогось процесу. Якщо виконується операція введення-виведення, ОС повинна мати інформацію про її стан і про те, які адреси основної пам'яті задіяні в цій операції. Управління введенням-виведенням розглядатиметься пізніше.

Таблиці файлів. У цих таблицях знаходиться інформація про існуючі файли, їх розташування на магнітних носіях, поточний стан і інші атрибути. Велика частина цієї інформації, якщо не вся, може підтримуватися системою управління файлами. В цьому випадку ОС мало знає (чи зовсім нічого не знає) про файли. Ця тема також детально розглядатиметься в розділі «Управління файлами».

Таблиці процесів. Нарешті, ОС повинна підтримувати таблиці процесів (блоки управління процесами), щоб мати можливість управляти ними. Врешті- решт, управління пам'яттю, пристроями введення-виведення і файлами здійснюється для того, щоб могли виконуватися процеси, тому в таблицях процесів мають бути явні або неявні посилання на ці ресурси.

5.6.2 Структури управління процесами

Розглянемо питання про те, які відомості повинні бути в розпорядженні ОС, щоб вона могла управляти процесом. По-перше, вона повинна знати, де знаходитися процес, а по-друге, їй мають бути відомі необхідні для управління атрибути процесу (такі, як його ідентифікатор, стан і розміщення в пам'яті).

Упродовж існування процесу його виконання може бути багаторазово перерване і продовжене. Для того щоб відновити виконання процесу, необхідно відновити стан його операційного середовища. Стан операційного середовища відображається станом регістрів і програмного лічильника, режимом роботи процесора, покажчиками на відкриті файли, інформацією про незавершені операції введення-виведення тощо.

Блок управляючого процесу. Для того щоб операційна система могла виконувати операції над процесами, кожен процес представляється деякою структурою даних.

Розглянемо ще раз, в чому ж полягає фізичний прояв процесу. Як мінімум, у процес входить програма, яку треба виконати. З цією програмою пов'язаний набір елементів пам'яті, в яких зберігаються локальні і глобальні змінні. Таким чином,

процесу має бути виділений такий об'єм пам'яті, в якому помістилися б програма і дані. Крім того, при роботі програми використовується стек, за допомогою якого реалізуються виклики процедур і передача параметрів. Нарешті, з кожним процесом пов'язані декілька атрибутів, які використовуються ОС для управління цим процесом. Тому, ОС для реалізації планування процесів потрібно структура, що містить інформацію, специфічну для цього процесу:

- стан, в якому знаходиться процес;
- програмний лічильник процесу або, іншими словами, адреса команди, яка має бути виконана наступною;
- вміст регістрів процесора;
- дані, необхідні для планування використання процесора і управління пам'яттю (пріоритет процесу, розмір і розташування адресного простору тощо);
- облікові дані (ідентифікаційний номер процесу, загальний час використання процесора цим процесом тощо);
- інформацію про пристрої введення-виведення, пов'язані з процесом (наприклад, які пристрої закріплені за процесом, таблицю відкритих файлів).

Такий набір атрибутів називається управляючим блоком процесу (PCB – Process Control Block). Часто використовуються інші назви цієї структури даних – дескриптор процесу, блок управління задачою, дескриптор задачі.

Блок управління процесом є моделлю процесу для операційної системи. Будь-яка операція, виконана операційною системою над процесом, викликає певні зміни в PCB.

Дескриптори окремих процесів об'єднані в список, що утворює таблицю процесів. Пам'ять для таблиці процесів відводиться динамічно в області ядра. На підставі інформації, що міститься в таблиці процесів, операційна система здійснює планування і синхронізацію процесів. У дескрипторі прямо або побічно (через покажчики, на пов'язані з процесом структури) міститься інформація про стан процесу, про розташування образу процесу в оперативній пам'яті і на диску, про значення окремих складових пріоритету. Також у таблиці процесів знаходиться інформація про його підсумкові значення – глобальний пріоритет, ідентифікатор користувача процесу, споріднені процеси, події, здійснення яких чекає цей процес, тощо.

Конкретний склад РСВ і будова залежать, звичайно, від конкретної операційної системи. У багатьох операційних системах інформація, що характеризує процес, зберігається не в одній, а в декількох пов'язаних структурах даних. Ці структури можуть мати різні найменування, містити додаткову інформацію або, навпаки, лише частину описаної інформації. Для нас це не має значення. Для нас важливо лише те, що для будь-якого процесу, що знаходиться в обчислювальній системі, вся інформація, необхідна для здійснення операцій над ним, доступна операційній системі.

Контекст процесу. Інформацію, для зберігання якої призначений блок управління процесом, зручно для подальшого викладу розділити на дві частини. Вміст усіх регістрів процесора (включаючи значення програмного лічильника) називатимемо регістровим контекстом процесу, а усе інше – системним контекстом процесу. Знання регістрового і системного контекстів процесу вистачає для того, щоб управляти його поведінкою в операційній системі, здійснюючи над ним операції. Проте цього недостатньо, щоб повністю характеризувати процес.

Операційну систему не цікавить, якими саме обчисленнями займається процес, тобто який код і які дані знаходяться в його адресному просторі. З точки зору користувача, навпаки, найбільший інтерес представляє вміст адресного простору процесу, можливо разом з регістровим контекстом, що визначає послідовність перетворення даних і отримані результати. Код і дані, що знаходяться в адресному просторі процесу, називатимемо його контекстом користувача. Сукупність регістрового, системного і призначеного для користувача контекстів процесу скорочено прийнято називати просто контекстом процесу. У будь-який момент часу процес повністю характеризується своїм контекстом.

Множина, в яку входять програма, дані, стек і атрибути (управляючий блок процесу), називається образом процесу (process image). Місцезнаходження образу процесу залежить від використовуваної схеми управління пам'яттю. У простому випадку образ процесу має вигляд безперервного блоку пам'яті, який розташований у вторинній пам'яті, зазвичай на диску. Щоб ОС могла управляти процесом, принаймні, невелика частина його образу повинна знаходитися в основній пам'яті. Щоб запустити процес, його образ необхідно повністю завантажити в основну пам'ять.

5.6.3 Атрибути процесів

Багатозадачна система повинна мати в розпорядженні відомості про кожен процес. Різні ОС організовують цю інформацію по-різному. Тому розглянемо питання про те, яка інформація (типові елементи управляючого блоку процесу) може знадобитися ОС, не зупиняючись на схемі організації цієї інформації.

Ідентифікація процесів.

Ідентифікатори. Числові ідентифікатори, які можуть зберігатися в управляючому блоці процесу:

1. Ідентифікатор процесу. Зазвичай це числовий ідентифікатор. При створенні дочірніх процесів ідентифікатори вказують батьківський і дочірні процеси.
2. Ідентифікатор батьківського процесу.
3. Ідентифікатор користувача.
4. Інформація про стан процесу.

Регістри, доступні користувачеві. Це регістри, до яких можна звернутися за допомогою машинних команд. Зазвичай їх від 8 до 32, хоча в деяких реалізаціях RISC процесорів (з обмеженим набором команд) їх понад 100.

Управляючі регістри і регістри стану. У процесорі є декілька різновидів регістрів, які використовуються для управління роботою процесора. Перериваючи процес, усю інформацію, що міститься в регістрах, необхідно зберегти, щоб потім відновити при відновленні виконання процесу. До них належать:

1. Лічильник команд. У цьому регістрі зберігається адреса чергової команди.
2. Коди умови. Відбиває результат виконання арифметичної або логічної операції (наприклад, знак, рівність, переповнювання).
3. Інформація про стан. Сюди входять прапори дозволу переривань і інформація про режим виконання, слово стану процесора PSW (Processor Status Word). Цей регістр містить біти коду станів, які задаються командами порівняння, пріоритетом центрального процесора, режимом користувача/ядра, та іншу інформацію.
4. Покажчики на стек. З кожним процесом пов'язані один або декілька системних стеків. У стеку зберігаються параметри і адреси викликів процедур і

системних служб. Покажчик стека вказує на його вершину. Управляча інформація процесу.

Інформація про планування і стан. Зазвичай включає таке:

1. Стан процесу. Тобто, виконується, готовий до виконання, очікуючий якоїсь події або призупинений.

2. Пріоритет. У деяких ОС їх може бути декілька (за замовчуванням, поточний, максимально можливий).

3. Інформація, що пов'язана з плануванням. Залежить від використаного алгоритму планування.

4. Інформація про події. Ідентифікація події, настання якої дозволить продовжити виконання процесу, що знаходиться в очікуванні.

Структуризація даних. Процес може бути пов'язаний з іншими процесами за допомогою черги, кільця або іншої структури. Наприклад, усі процеси в стані очікування, що мають один і той же пріоритет, можуть знаходитися в одній черзі. Процеси можуть мати родинні відношення (бути батьківськими або дочірніми стосовно один до одного), тому можуть бути покажчики на процеси.

Обмін інформацією між процесами. Різні прапори, сигнали і повідомлення можуть мати стосунки до обміну інформацією між двома незалежними процесами.

Привілеї процесів. Процеси можуть мати привілеї прав доступу до певних областей пам'яті, виконувати деякі команди або використати різні системні утиліти і служби.

Управління пам'яттю. Цей розділ може містити покажчик на програмний сегмент, покажчик на сегмент даних, покажчики на таблиці сегментів і/або сторінок, в яких описується розподіл процесу у віртуальній пам'яті.

Управління файлами. Розділ може містити кореневий каталог, робочий каталог, дескриптори файлу, ідентифікатор користувача.

Володіння ресурсами і їх використання. Вказуються ресурси, якими управляє процес (наприклад, перелік відкритих файлів).

Особливо слід зауважити, що в інформації про стан процесора є опис регістра або набору регістрів, відомих під назвою «слово стану програми» (Program Status Word – PSW), в яких міститься інформація про стан і коди умов.

5.6.4 Управління процесами

Перш ніж обговорити метод, який ОС використовує для управління процесами, розглянемо спочатку розбіжності між режимами роботи процесора при виконанні коду ОС, і при виконанні кодів програм користувача. Більшість процесорів підтримують два режими роботи. Певні команди виконуються тільки в більше привілейованому режимі. До них належать команди читання або внесення змін до управляючих регістрів (LDTR, GDTR та ін.), команди введення-виведення, а також команди, які пов'язані з управлінням пам'яттю. Крім того, доступ до деяких ділянок пам'яті може бути дозволений тільки в привілейованому режимі.

Режим з меншими привілеями називають режимом користувача, оскільки в ньому виконуються програми користувача. Режим з вищими привілеями називається системним режимом (system mode), режимом управління (control mode) або режимом ядра (kernel mode). Ядро – це частина ОС, яка виконує найважливіші її функції, серед яких виділимо наведені нижче.

1. Управління процесами:
   - створення і завершення процесів;
   - планування і диспетчеризація процесів;
   - перемикання процесів;
   - синхронізація і підтримка обміну інформацією між процесами;
   - організація управляючих блоків процесів.
2. Управління пам'яттю:
   - виділення адресного простору процесам;
   - свопінг;
   - управління сторінками і сегментами;
   - управління введенням-виведенням;
   - управління буферами;
   - виділення процесам каналів і пристроїв введення-виведення.
3. Функції підтримки:
   - обробка переривань;
   - облік використання ресурсів;
   - поточний контроль системи.

З наведеного легко зрозуміти, для чого потрібні два вище згаданих режими. Це необхідно для захисту ОС і її основних структур, таких як управляючі блоки процесів, від можливого впливу програм користувачів. Програми, які працюють в режимі ядра, мають повний контроль над процесором і всіма його командами і регістрами, а також мають доступ до всіх елементів пам'яті. Такий рівень привілеїв для програм користувачів не потрібний, тому, виходячи з міркувань безпеки, його роблять недоступним для програм користувача.

У зв'язку з цим виникає два питання: яким чином процесор може визначити, в якому режимі може виконуватися ця програма, і як здійснюється перемикання з одного режиму в інший? Відносно першого питання, то в PSW програми є спеціальний біт (VM), де вказаний режим виконання. При деяких подіях цей біт змінюється. Наприклад, якщо програма викличе сервіс ОС, встановлюється режим ядра. Це виникає в результаті виконання команд зміни режиму. Якщо програма користувача спробує виконати таку команду, то це призведе до передачі управління ОС, а якщо така зміна режиму не дозволена, то виникне помилка виконання.