Лекція 2.2. Мультипрограмування на основі переривань. Системні виклики

2.2.1. Призначення і типи переривань

Переривання є основною рушійною силою будь-якої ОС. Якщо відключити систему переривань – «життя» в ОС негайно зупиниться. Періодичні переривання від таймера змінюють процеси в мультипрограмній ОС, а переривання від пристроїв уведення-виведення керують потоками даних, якими обчислювальна система обмінюється із зовнішнім середовищем.

Залежно від джерела переривання поділяють на три великі класи:

- зовнішні;
- внутрішні;
- програмні.

Зовнішні переривання можуть спричиняти дії користувача чи оператора за терміналом, або сигнали, що надходять від апаратних пристроїв – сигнали завершення операцій введення-виведення, зовнішніх пристроїв комп'ютера, що виробляються контролерами, такими як принтер або нагромаджувач на жорстких дисках, або ж сигнали від датчиків керованих комп'ютером технічних об'єктів.

Зовнішні переривання називають також ***апаратними***, відображаючи той факт, що переривання виникає унаслідок подачі деякою апаратурою (наприклад, контролером принтера) електричного сигналу, який передається (можливо, проходячи через інші блоки комп'ютера, наприклад контролер переривань) на спеціальний вхід переривання процесора.

Цей клас переривань є асинхронним відносно потоку інструкцій програми, що переривається. Апаратура процесора працює так, що асинхронні переривання виникають між виконанням двох сусідніх інструкцій, при цьому система після оброблення переривання продовжує виконувати процес, вже починаючи з наступної інструкції.

Внутрішні переривання, названі також ***виключеннями (exeption)***, відбуваються синхронно до виконання програми в разі появи аварійної ситуації в ході виконання деякої інструкції програми. Прикладами виключень є ділення на нуль, помилки захисту пам'яті, звернення за неіснуючою адресою, спроба виконати привілейовану інструкцію в призначеному для користувача режимі й інші подібні виключення, що виникають безпосередньо в ході виконання тактів команди («усередині» виконання).

Програмні переривання відрізняються від попередніх двох класів тим, що вони за своєю суттю не є дійсними перериваннями. Програмне переривання відбувається під час виконання особливої команди процесора, виконання якої імітує переривання, тобто перехід на нову послідовність інструкцій.

Перериванням приписується пріоритет, за допомогою якого вони ранжуються за значущістю і терміновістю. Переривання, що мають однакове значення пріоритету, належать до одного рівня пріоритету переривань.

Процедури, що викликаються за перериваннями називають ***обробниками переривань, або процедурами обслуговування переривань (Interrupt Service Routine).*** Апаратні переривання обробляються драйверами відповідних зовнішніх пристроїв, виключення – спеціальними модулями ядра,

а програмні переривання – процедурами ОС, які обслуговують системні виклики. Окрім цих модулів в складі ОС може бути ***диспетчер переривань***, який координує роботу окремих обробників переривань.

2.2.2. Механізм переривань

Механізм переривань підтримується апаратними засобами комп'ютера і програмними засобами ОС.

Апаратна підтримка переривань має свої особливості, що залежать від типу процесора та інших апаратних компонентів, які передають сигнал запиту переривання від зовнішнього пристрою до процесора (контролер зовнішнього пристрою, шини підключення зовнішніх пристроїв, контролер переривань, що є посередником між сигналами шини та сигналами процесора). Особливості апаратної реалізації переривань впливають на засоби програмної підтримки переривань, що працюють у складі ОС.

Шини виконують переривання за допомогою двох основних способів: ***векторним* (*vectored*) і *опитуваним* (*polled*)**. За обома способами процесору надається інформація про рівень пріоритету переривання на шині підключення зовнішніх пристроїв. У разі векторних переривань у процесор надходить також інформація про початкову адресу програми оброблення виниклого переривання – ***обробника переривань*.**

Пристроям, які використовують векторні переривання, призначається ***вектор переривань***. Це електричний сигнал, що подається на відповідні шини процесора і містить інформацію про визначений, закріплений за цим пристроєм номер, який ідентифікує відповідний обробник переривань. Цей вектор може бути ***фіксованим, конфігурованим*** (наприклад, з використанням перемикачів) або ***програмованим*.**

2.2.3. Програмне переривання

Програмне переривання реалізовує один із способів переходу на підпрограму за допомогою спеціальної інструкції процесора, такої як *INT* у процесорах **Intel Pentium**, ***trap*** у процесорах **Motorola**, ***syscall*** у процесорах **MIPS** або ***Ticc*** у процесорах SPARC. У ході виконання команди програмного переривання процесор відпрацьовує ту ж послідовність дій, що й у разі виникнення зовнішнього або внутрішнього переривання, тільки відбувається це в передбаченій точці програми – там, де програміст помістив цю програму.

Усі сучасні процесори мають в системі команд інструкції програмних переривань. Однією з причин появи інструкцій програмних переривань у системі команд процесорів є те, що їх використання часто приводить до компактнішого коду програм порівняно з використанням стандартних команд виконання

процедур. Це пояснюється тим, що розробники процесора зазвичай резервують для оброблення переривань невелику кількість можливих підпрограм, тому довжина операнда в команді програмного переривання, який вказує на потрібну підпрограму, менша, ніж в команді переходу на підпрограму..

2.2.4. Диспетчеризація і пріоритезація переривань в операційній системі

Операційна система повинна відігравати активну роль в організації оброблення переривань. Переривання виконують дуже корисну для обчислювальної системи функцію – вони дозволяють реагувати на асинхронні до обчислювального процесу події. Водночас переривання створюють додаткові труднощі для ОС в організації обчислювального процесу. Ці труднощі зумовлені з непередбаченими переходами керування від однієї процедури до іншої,

спричиненими перериваннями від контролерів зовнішніх пристроїв. Можуть також виникати в непередбачені моменти часу виключення, викликані помилками під час виконання інструкцій. Ускладнюють завдання планування обчислювальних робіт і запити на виконання системних функцій (системні виклики) від призначених для користувача застосувань, здійснювані за допомогою програмних переривань. Самі модулі ОС також часто викликають один одного за програмними перериваннями, ще більше заплутуючи обчислювальний процес.

Операційна система не може втрачати контроль над ходом виконання системних процедур, що викликаються за перериваннями. Вона повинна впорядковувати їх у часі так само, як планувальник упорядковує численні, призначені для користувача, потоки. Крім того, сам планувальник потоків є системною процедурою, що викликається за

перериваннями (апаратним – від таймера або контролера пристрою введення-виведення, або програмним – від додатка або модуля ОС). Тому правильне планування процедур, що викликаються за перериваннями, є необхідною умовою правильного планування призначених для користувача потоків. Інакше в системі можуть виникати, наприклад, такі ситуації, коли ОС тривалий час займатиметься завданням, що потребує миттєвої реакції на керування стримером, який архівує дані, в той час, коли високошвидкісний диск простоюватиме і гальмуватиме роботу численних застосувань, що обмінюються даними з цим диском. Приклад такої ситуації ілюструє рис. 2.10. Обробник переривань принтера блокує на тривалий час оброблення переривання від таймера, внаслідок чого системний час на деякий час «завмирає» і потік 2, критично важливий для користувача, не отримує керування в

установлений час. Гостроту проблеми пом'якшує та обставина, що у багатьох випадках оброблення переривання залежить від виконання декількох операцій введення-виведення і тому має дуже малу тривалість. Проте ОС завжди повинна контролювати ситуацію і виконувати критичну роботу вчасно, а не покладатися на випадок.

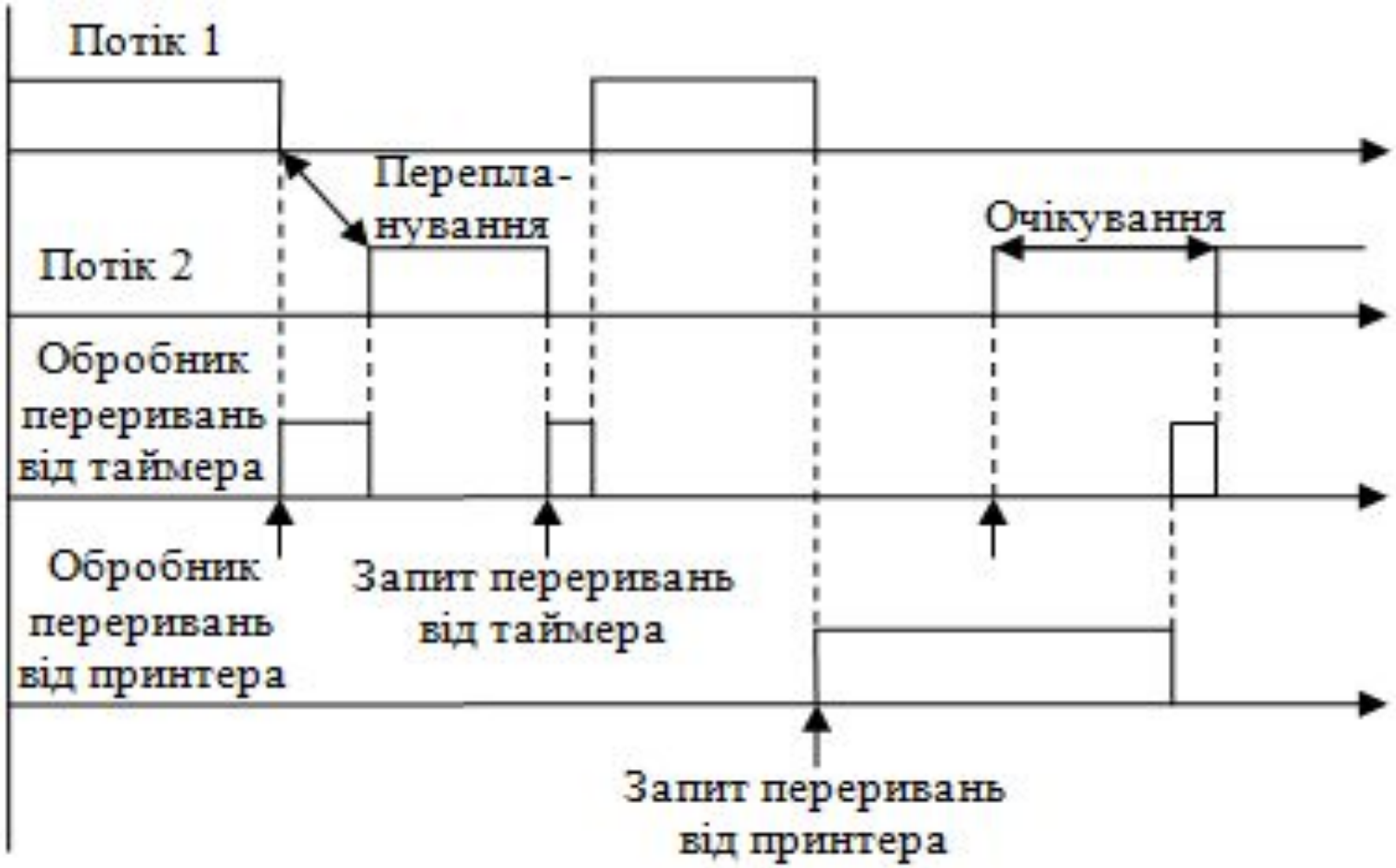


Рис. 2.10. Неврегульоване оброблення переривань

Для впорядкування роботи обробників переривань в ОС застосовують такий самий механізм, що і для впорядкування роботи призначених для користувача процесів – механізм пріоритетних черг. Усі джерела переривань зазвичай поділяють на декілька класів, причому кожному класу привласнюється пріоритет. В ОС виділяється програмний модуль, який займається диспетчеризацією обробників переривань. Цей модуль у різних ОС називають по-різному, але для визначеності його називатимемо диспетчером переривань..

У разі виникнення переривання диспетчер переривань викликається першим. Він забороняє ненадовго всі переривання, а потім з'ясовує причину переривання. Після цього диспетчер порівнює призначений цьому джерелу переривання пріоритет з поточним пріоритетом потоку команд, які виконував процесор. У цей момент часу процесор вже може виконувати інструкції іншого обробника переривань, що також має деякий пріоритет. Якщо пріоритет нового

запиту вищий від поточного, то виконання інструкції поточного обробника припиняється і він поміщається у відповідну чергу обробників переривань. Інакше в чергу поміщається обробник нового запиту.

Функції централізованого диспетчера переривань на прикладі Windows NT. Деякі процесори або контролери переривань комп'ютера на апаратному рівні підтримують пріоритезацію запитів на переривання. Наприклад, у процесорах MIPS є декілька рівнів апаратних запитів на переривання і декілька рівнів програмних запитів. Процесор має внутрішню змінну, названу рівнем переривань процесора. Переривання відбувається лише у тому випадку, коли рівень запиту на переривання вищий від поточного рівня переривань процесора. Є також привілейована інструкція, за допомогою якої код ядра ОС може змінити рівень переривань процесора.

Для усунення залежності від апаратної платформи в деякі ОС упроваджують власну програмну систему пріоритетів переривань. Прикладом такої ОС є Windows NT.

У Windows NT нижчий рівень IRQL відповідає звичайним потокам, призначуваним на виконання диспетчером потоків (рис. 2.11). Це є деяким допущенням, оскільки код потоків починає виконуватися процесором не в результаті запиту на переривання, але це допущення справедливе, оскільки дозволяє будь-якому «справжньому» запиту переривати код звичайного потоку.

Рис. 2.11. Диспетчеризація переривань у Windows NT

2.2.5. Процедури оброблення переривань і поточний процес

Важливою особливістю процедур, що виконуються за запитами переривань, є те, що вони виконують роботу, найчастіше ніяк не пов'язану з поточним процесом. Наприклад, драйвер диска може отримати керування після того, як контролер диска записав у відповідні сектори інформацію, отриману від процесу *A*, але цей момент часу, імовірно, не збігатиметься з періодом чергової ітерації виконання процесу *A* або його потоку. У найбільш типовому випадку процес *A* перебуватиме в стані очікування завершення операції введення-виведення (за синхронного режиму виконання цієї операції) і драйвер диска перерве який-небудь інший процес, наприклад процес *B*. У Windows NT процедури, що викликаються як DPC, також можуть працювати в

контексті процесу, що відрізняється від того, для якого вони виконують свої функції. У деяких випадках взагалі важко однозначно визначити, для якого процесу виконує роботу той чи інший програмний модуль ОС, наприклад планувальник потоків. Тому для такого роду процедур вводяться обмеження – вони не мають права використовувати ресурси (пам'ять, відкриті файли і т. ін.), з якими працює поточний процес, або ж від імені цього процесу запрошувати виділення додаткових ресурсів. Процедури оброблення переривань працюють з ресурсами, які були виділені ним під час ініціалізації відповідного драйвера або ініціалізації самої ОС. Ці ресурси належать ОС, а не конкретному процесу. Зокрема, пам'ять виділяється драйверам із системної ділянки, тобто тієї ділянки, на яку відображуються сегменти із загальної частини

віртуального адресного простору всіх процесів. Тому зазвичай процедури оброблення переривань працюють поза контекстом процесу. Оскільки всі подібні процедури є частиною ОС за дотримання цих обмежень відповідає системний програміст. Змусити свої модулі виконувати ці обмеження ОС не може.

Прикладом того, що не буває правил без винятку, є ОС Windows NT. У ній є процедури оброблення переривань, які виконуються завжди в контексті певного процесу. Це процедури, що викликаються за допомогою програмного переривання APC (Asynchronous Procedure Call – виклик асинхронної процедури). Для них у диспетчері переривань передбачено власний рівень пріоритету IRQL, вищий за рівень для звичайного коду, але нижчий за рівень DPC. Ці процедури можуть перервати поточний код і виконуватися у разі дотримання двох умов:

поточний код має нижчий рівень пріоритету (тобто виконується звичайний код), поточним процесом є певний процес, описувач якого задається в запиті на переривання для певної процедури APC. Процедури APC можуть користуватися ресурсами поточного процесу, і, власне, для цього вони і були впроваджені. Основне призначення APC-процедур – переміщення даних, отриманих драйвером від якого-небудь пристрою введення-виведення, з пам'яті системної ділянки пам'яті, куди вони поміщаються після зчитування з регістрів контролера цього пристрою, в індивідуальну частину адресного простору процесу, що запитував операцію введення-виведення. Така дія постійно виконується системою введення-виведення, і для її реалізації були введені такі специфічні процедури оброблення переривань, як APC.

Диспетчеризація переривань є важливою функцією ОС, і ця функція реалізується майже у всіх мульти-програмних ОС. У загальному випадку в ОС реалізується дворівневий механізм планування робіт. Верхній рівень планування виконується диспетчером переривань, який розподіляє процесорний час між потоком запитів, що надходять на переривання різних типів, – зовнішніх, внутрішніх і програмних. Процесорний час, що залишився, розподіляється іншим диспетчером – диспетчером потоків – на підставі дисциплін квантування й інших дисциплін.

2.2.6. Системні виклики

Системний виклик дозволяє додатку звернутися до ОС з тим, щоб вона виконала ту або іншу дію, оформлену як процедуру (або набір процедур) кодового сегменту ОС.

Для прикладного програміста ОС виглядає як деяка бібліотека, що надає деякий набір корисних функцій, за допомогою яких можна спростити прикладну програму або виконати дії, заборонені в користувацькому режимі, наприклад обмін даними з пристроєм уведення-виведення.

Реалізація системних викликів має задовольняти такі вимоги:

- забезпечувати перемикання в привілейований режим;
- мати високу швидкість виклику процедур ОС;
- забезпечувати по можливості одноманітне звернення до системних викликів для всіх апаратних платформ, на яких працює ОС;
- допускати незначне розширення набору системних викликів;

- забезпечувати контроль з боку ОС за коректним використанням системних викликів.

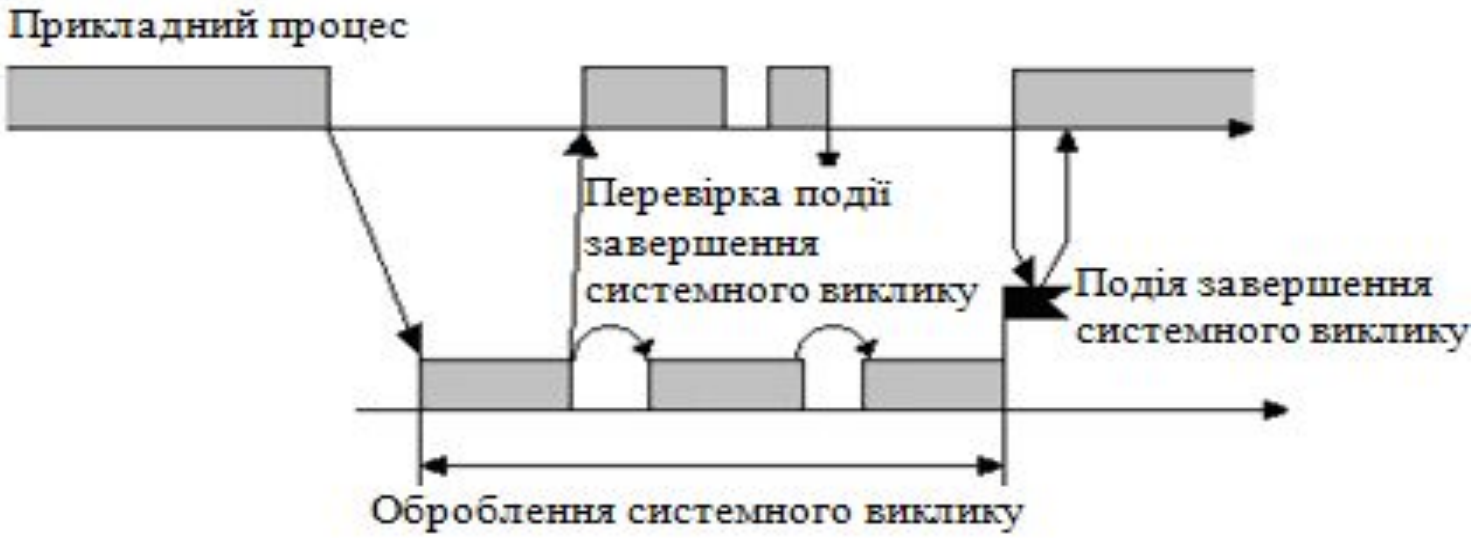
Перша вимога для більшості апаратних платформ може бути виконаною лише за допомогою механізму програмних переривань. Тому вважатимемо, що останні вимоги потрібно забезпечити саме для такої реалізації системних викликів. Як це зазвичай буває, деякі з цих вимог взаємно суперечливі.

Для забезпечення високої швидкості було б корисно використовувати векторні властивості системи програм-них переривань, притаманні багатьом процесорам, тобто закріпити за кожним системним викликом певне значення вектора. Додаток за такого способу виклику безпосередньо вказує в аргументі запиту значення вектора, після чого керування негайно передається

необхідній процедурі ОС. Проте цей децентралізований спосіб передавання керування залежить від особливостей апаратної платформи, а також не дозволяє ОС легко модифікувати набір системних викликів і контролювати їх використання. Наприклад, у процесорі *Pentium* кількість системних викликів визначається кількістю векторів переривань, виділених для цієї мети із загального пулу з 256 елементів (частина яких використовується під апаратні переривання і оброблення виключень). Додавання нового системного виклику вимагає від системного програміста ретельного пошуку вільного елемента в таблиці переривань, якого на якомусь етапі розвитку ОС може і не виявитися.

Операційна система може виконувати системні виклики в ***синхронному*** або ***асинхронному режимі***. ***Синхронний системний виклик*** означає, що процес, який зробив такий виклик, припиняється (переводиться планувальником ОС у стан очікування) доти, доки системний виклик не виконає всю потрібну від нього роботу (рис. 2.12, *а*). Після цього планувальник переводить процес у стан готовності і під час чергового виконання процес гарантовано може скористатися результатами завершеного до цього часу системного виклику. Синхронні виклики називаються також ***блокувальними***, оскільки процес, що викликав системну дію, блокується до його завершення.

а



б

Рис. 2.12. Синхронні та асинхронні системні виклики

Асинхронний системний виклик не приводить до переведення процесу в режим очікування після виконання деяких початкових системних дій, наприклад запуску операції введення-виведення, керування повертається прикладному процесу (рис. 2.12, б).

Більшість системних викликів в ОС є синхронними, оскільки цей режим звільняє додаток від роботи зі з'ясування моменту появи результату виклику. Водночас у нових версіях ОС кількість асинхронних системних викликів поступово збільшується, що надає більше свободи розробникам складних застосувань. Особливо потрібні асинхронні системні виклики в ОС на основі мікроядерного підходу, оскільки в призначеному для користувача режимі працює декілька ОС, яким необхідно мати повну свободу в організації своєї роботи, а таку свободу дає лише асинхронний режим обслуговування викликів мікроядром.

Дякую за увагу!!!