3.6 МУЛЬТИПРОГРАМУВАННЯ

Мультипрограмування, або багатозадачність (multitasking), – це спосіб організації обчислювального процесу, при якому на одному процесорі чередуючись виконуються відразу декілька програм. Ці програми спільно використовують не лише процесор, але і інші ресурси комп'ютера: оперативну і зовнішню пам'ять, пристрої введення-виведення, дані. Мультипрограмування підвищує ефективність використання обчислювальної системи, проте ефективність може розумітися по-різному. Найхарактернішими критеріями ефективності обчислювальних систем є:

- пропускна спроможність – кількість завдань, що виконуються обчислювальною системою за одиницю часу;
- зручність роботи користувачів, що полягає, зокрема, в тому, що вони мають можливість інтерактивно працювати одночасно з декількома додатками на одній машині;
- реактивність системи – здатність системи витримувати заздалегідь задані інтервали часу між запуском програми і отриманням результату.

Залежно від вибраного критерію ефективності ОС діляться на системи пакетної обробки, системи розподілу часу і системи реального часу. Кожен тип ОС має специфічні внутрішні механізми і особливі сфери застосування. Деякі ОС можуть підтримувати одночасно декілька режимів.

3.6.1 Мультипрограмування в системах пакетної обробки

При використанні мультипрограмування для підвищення пропускної спроможності комп'ютера головною метою є мінімізація простоїв усіх пристроїв комп'ютера, і, перш за все, центрального процесора. Такі простої можуть виникати із-за призупинення завдання з його внутрішніх причин, пов'язаних, наприклад, з очікуванням введення даних для обробки. Дані можуть зберігатися на диску або ж поступати від користувача, працюючого за терміналом, а також від вимірювальної апаратури, встановленої на зовнішніх технічних об'єктах. При виникненні такого роду блокування виконуваного завдання природним рішенням, що веде до підвищення ефективності використання процесора, є перемикання процесора на

виконання іншого завдання, в якого є дані для обробки. Така концепція мультипрограмування покладена в основу так званих пакетних систем.

Системи пакетної обробки призначалися для розв'язання задач в основному обчислювального характеру, що не вимагають швидкого отримання результатів. Головною їх метою і критерієм ефективності є максимальна пропускна спроможність, тобто розв'язання максимального числа завдань за одиницю часу.

Для досягнення цієї мети в системах пакетної обробки використовується така схема функціонування. На початку роботи формується пакет завдань, в якому кожне завдання містить вимогу до системних ресурсів. З цього пакету завдань формується мультипрограмна суміш, тобто певна кількість одночасно виконуваних завдань. Для одночасного виконання вибираються завдання, що пред'являють різні вимоги до ресурсів, так, щоб забезпечувалося збалансоване завантаження усіх пристроїв обчислювальної машини. Наприклад, в мультипрограмній суміші бажана одночасна присутність обчислювальних завдань і завдань з інтенсивним введенням-виведенням. Таким чином, вибір нового завдання з пакету завдань залежить від внутрішньої ситуації, що складається в системі, тобто вибирається «вигідне» завдання. Отже, в обчислювальних системах, працюючих під управлінням пакетних ОС, неможливо гарантувати виконання того або іншого завдання впродовж певного періоду часу.

Розглянемо детальніше поєднання в часі операцій введення-виведення і обчислень. Таке поєднання може досягатися різними способами. Один з них характерний, наприклад, для комп'ютерів, що мають спеціалізований процесор введення-виведення. У комп'ютерах класу мейнфреймів такі процесори називають каналами. У системі команд центрального процесора передбачається спеціальна інструкція, за допомогою якої каналу передаються параметри і вказівки на те, яку програму введення-виведення він повинен виконати. Починаючи з цього моменту центральний процесор і канал можуть працювати паралельно (рис. 3.14, а).

Інший спосіб поєднання обчислень з операціями введення-виведення реалізується в комп'ютерах, в яких зовнішні пристрої управляються не процесором введення-виведення, а контролерами. Кожен зовнішній пристрій (чи група зовнішніх пристроїв одного типу) має свій власний контролер, який автономно відпрацьовує команди, що поступають від центрального процесора. При цьому контролер і

центральний процесор працюють асинхронно. Оскільки багато зовнішніх пристроїв включають електромеханічні вузли, контролер виконує свої команди управління пристроями істотно повільніше, ніж центральний процесор – свої.

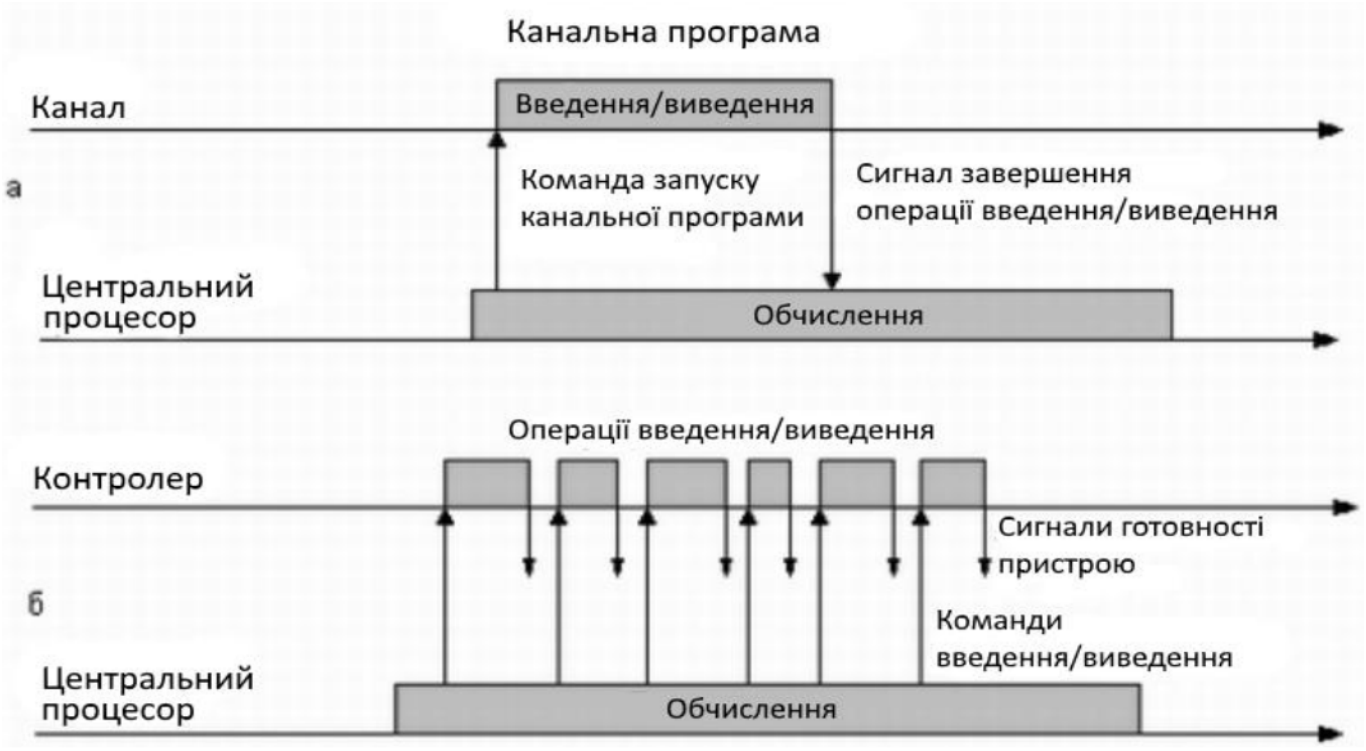

Рисунок 3.14 – Паралельне виконання обчислень і операцій введення-виведення

Ця обставина використовується для організації паралельного виконання обчислень і операцій введення-виведення: в проміжку часу між передачею команд, контролеру центральний процесор може виконувати обчислення (див. рис. 3.14, б). Контролер може повідомити центральний процесор про те, що він готовий прийняти наступну команду, сигналом переривання або центральний процесор дізнається про це, періодично опитуючи стан контролера.

Максимальний ефект прискорення досягається при найповнішому перекритті обчислень і введення-виведення. Розглянемо випадок, коли процесор виконує тільки одне завдання. У цій ситуації ступінь прискорення залежить від природи цього завдання і від того, наскільки ретельно був виявлений можливий паралелізм при її програмуванні. У завданнях, в яких переважають або обчислення, або введення-виведення, прискорення майже відсутнє. Паралелізм у рамках одного завдання неможливий також, коли для продовження обчислень потрібне повне завершення

операції введення-виведення. У таких випадках неминучі простої центрального процесора або каналу.

Якщо ж в системі виконуються одночасно декілька завдань, з'являється можливість поєднання обчислень одного завдання з введенням-виведенням іншого. Поки одне завдання чекає якої-небудь події, процесор не простоює, як це відбувається при послідовному виконанні програм, а виконує інше завдання. Відмітимо, що такою подією в мультипрограмній системі може бути не лише завершення введення-виведення, але і, наприклад, настання певного моменту часу, розблокування файлу або завантаження сторінки з диска.

Загальний час виконання суміші завдань часто виявляється меншим, ніж їх сумарний час послідовного виконання (рис. 3.15, а). Проте виконання окремого завдання в мультипрограмному режимі може зайняти більше часу, ніж при монопольному виділенні процесора цьому завданню.

Дійсно, при спільному використанні процесора в системі можуть виникати ситуації, коли завдання готове виконуватися, але процесор зайнятий виконанням іншого завдання. У таких випадках завдання, що завершило введення-виведення, готове виконуватися, але змушене чекати звільнення процесора, і це подовжує термін його виконання.

Так, з рис. 3.15 видно, що в однопрограмному режимі завдання А виконується за 6 одиниць часу, а в мультипрограмному – за 7. Завдання В також замість 5 одиниць часу виконується за 6. Та зате час виконання обох завдань в мультипрограмному режимі складає всього 8 одиниць, що на 3 одиниці менше, ніж при послідовному виконанні.

У системах пакетної обробки перемикання процесора з виконання одного завдання на виконання іншого відбувається за ініціативою найактивнішого завдання, наприклад, коли завдання відмовляється від процесора із-за необхідності виконати операцію введення-виведення. Тому існує висока ймовірність того, що одне завдання може надовго зайняти процесор і виконання інтерактивних завдань стане неможливим.

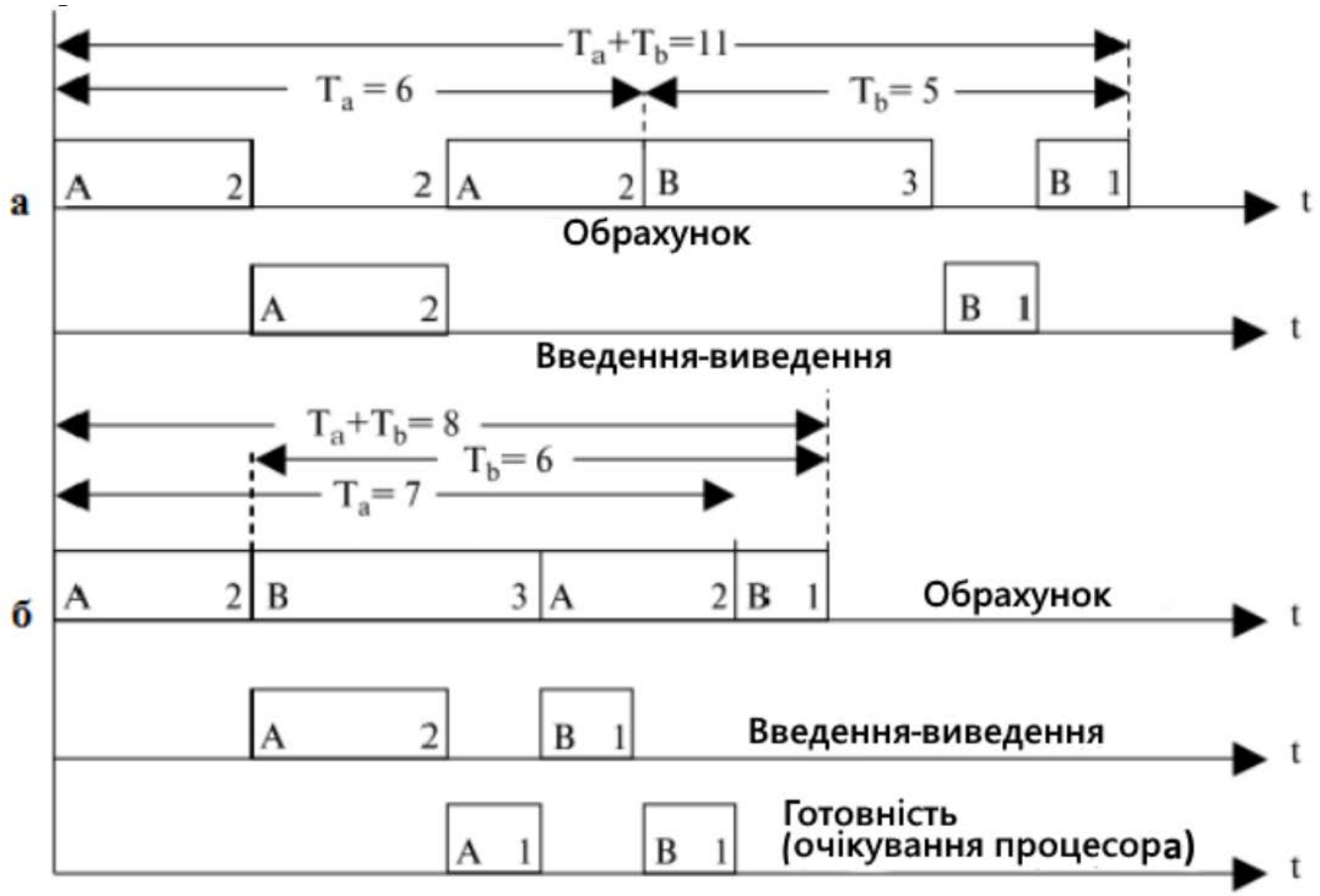

Рисунок 3.15 – Час виконання двох завдань: в однопрограмній (а), та мультипрограмній (б) системах

Взаємодія користувача з обчислювальною машиною з системою пакетної обробки зводиться до того, що він приносить завдання, віддає його диспетчерові-операторові, а в кінці дня після виконання усього пакету завдань отримує результат. Очевидно, що такий порядок підвищує ефективність функціонування апаратури, але знижує ефективність роботи користувача.

3.6.2 Мультипрограмування в системах розподілу часу

Підвищення зручності і ефективності роботи користувача є метою іншого способу мультипрограмування – розподілу часу. У системах розподілу часу користувачам (чи одному користувачеві) надається можливість інтерактивної роботи відразу з декількома додатками. Для цього кожен додаток повинен регулярно одержувати можливість «спілкування» з користувачем. Зрозуміло, що в пакетних системах можливості діалогу користувача з додатком дуже обмежені. У системах розподілу часу ця проблема вирішується за рахунок того, що

ОС примусово періодично призупиняє додатки, не чекаючи, коли вони добровільно звільнять процесор. Усім додаткам поперемінно виділяється квант процесорного часу, таким чином користувачі, що запустили програми на виконання, одержують можливість підтримувати з ними діалог.

Системи розподілу часу покликані виправити основний недолік систем пакетної обробки – ізоляцію користувача-програміста від процесу виконання його завдань. Кожному користувачеві в цьому випадку надається термінал, з якого він може вести діалог зі своєю програмою (рис. 3.16).

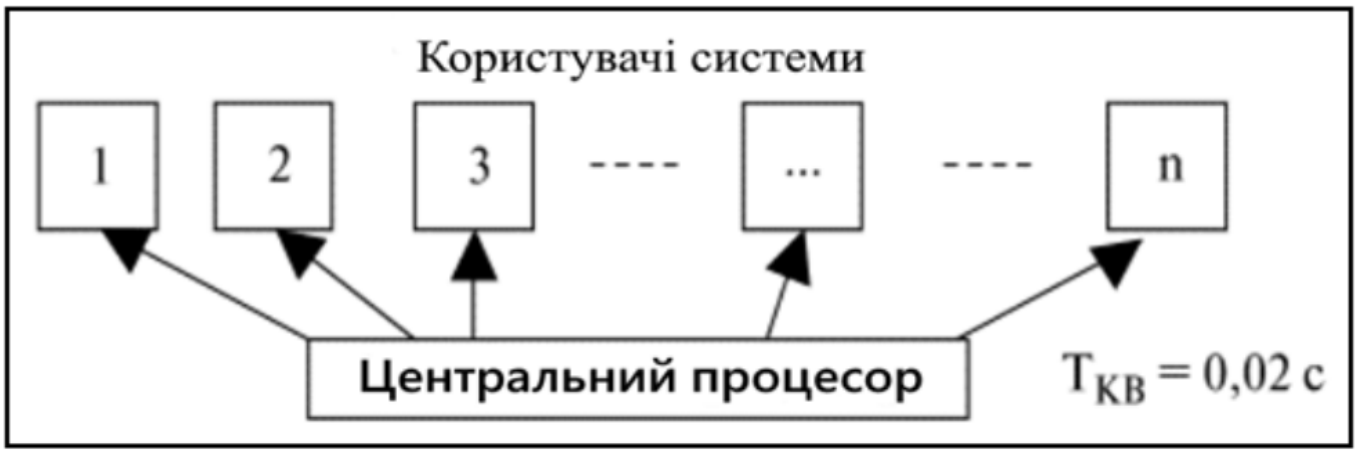

Рисунок 3.16 – Система розподілу часу

Оскільки в системах розподілу часу кожному завданню виділяється тільки квант процесорного часу, жодне завдання не займає процесор надовго і час відповіді виявляється прийнятним. Якщо квант вибраний досить невеликим, то в усіх користувачів, одночасно працюючих на одній і тій же машині, складається враження, що кожен з них одноосібно використовує машину.

Ясно, що системи розподілу часу мають меншу пропускну спроможність, чим системи пакетної обробки, оскільки на виконання приймається кожне запущене користувачем завдання, а не те, яке «вигідне» системі.

Крім того, продуктивність системи знижується із-за збільшених накладних витрат обчислювальної потужності на частіше перемикання процесора із завдання на завдання. Це цілком відповідає тому, що критерієм ефективності систем розподілу часу є не максимальна пропускна спроможність, а зручність і ефективність роботи користувача. В той же час мультипрограмне виконання інтерактивних додатків підвищує і пропускну спроможність комп'ютера (хай і не в такому ступені, як пакетні

системи). Апаратура завантажується краще, оскільки в той час, поки один додаток чекає повідомлення користувача, інші додатки можуть оброблятися процесором.

3.6.3 Мультипрограмування в системах реального часу

Ще один різновид мультипрограмування використовується в системах реального часу, призначених для управління від комп'ютера різними технічними об'єктами (наприклад, верстатом, супутником, науковою експериментальною установкою тощо) або технологічними процесами (наприклад, гальванічною лінією, доменним процесом тощо).

В усіх цих випадках існує гранично допустимий час, впродовж якого має бути виконана та або інша програма, що управляє об'єктом. Інакше може статися аварія: супутник вийде із зони видимості, експериментальні дані, що поступають з датчиків, будуть втрачені, товщина гальванічного покриття не відповідатиме нормі.

Таким чином, критерієм ефективності тут є здатність витримувати заздалегідь задані інтервали часу між запуском програми і отриманням результату (дії, що управляє). Цей час називається часом реакції системи, а відповідна властивість системи – реактивністю. Вимоги до часу реакції залежать від специфіки керованого процесу. Контролер робота може вимагати від вбудованого комп'ютера відповідь протягом 1 мс, тоді як при моделюванні польоту може бути прийнятна відповідь в 40 мс.

У системах реального часу мультипрограмна суміш є фіксованим набором заздалегідь розроблених програм, а вибір програми на виконання здійснюється за перериваннями (виходячи з поточного стану об'єкту) або відповідно до розкладу планових робіт.

Здатність апаратури комп'ютера і ОС до швидкої відповіді залежить в основному від швидкості перемикання з одного завдання на інше і, зокрема, від швидкості обробки сигналів переривання. Якщо при виникненні переривання процесор повинен опитати сотні потенційних джерел переривання, то реакція системи буде занадто повільною. Час обробки переривання в системах реального часу часто визначає вимоги до класу процесора навіть при невеликому його завантаженні.

У системах реального часу не прагнуть максимально завантажувати всі пристрої, навпаки, при проектуванні програмного комплексу, що управляє,

закладається деякий «запас» обчислювальної потужності на випадок пікового навантаження. Статистичні аргументи про низьку ймовірність виникнення пікового навантаження, грунтуються на тому, що ймовірність одночасного виникнення великої кількості незалежних подій дуже мала, і не застосована до багатьох ситуацій в системах управління.