КЕРУВАННЯ ВВОДОМ/ВИВОДОМ

В комп'ютерах використовують такі чотири методи керування вводом/виводом:

1. Програмно керований ввод/вивод;
2. Кероване перериваннями введення-виведення ;
3. Прямий доступ до пам'яті;
4. Введення-виведення під керуванням периферійних процесорів.

1 **Програмно-керований ввод-вивод**

В комп'ютерах, що використовують програмно-кероване введення-виведення, в інтерфейсній схемі кожного пристрою введення-виведення є регістр команд і станів (РКС), рис. 1. В даному регістрі є розряд прапорця F, який при потребі обміну з боку пристрою введення-виведення встановлюється в стан логічної «1».

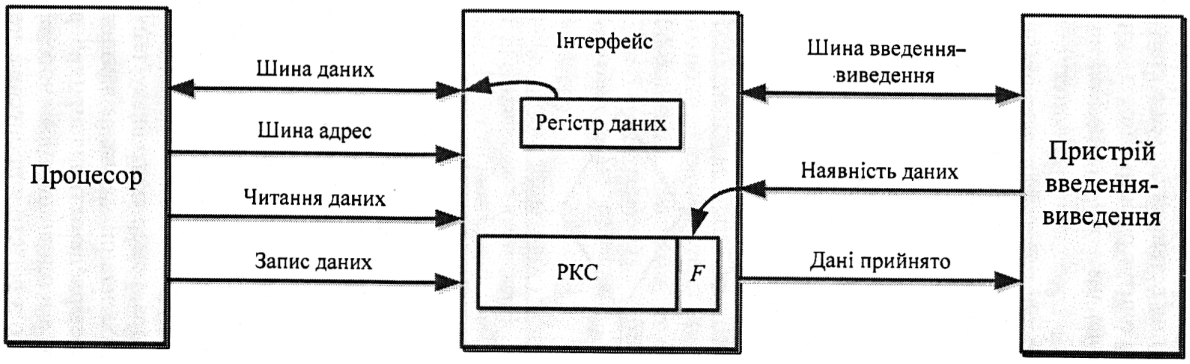

Рисунок 1 - Структура інтерфейсу для програмно-керованого вводу/виводу

Процесор безупинно опитує регістр команд і станів кожного пристрою введення-виведення, чекаючи на надходження даних рис. 2

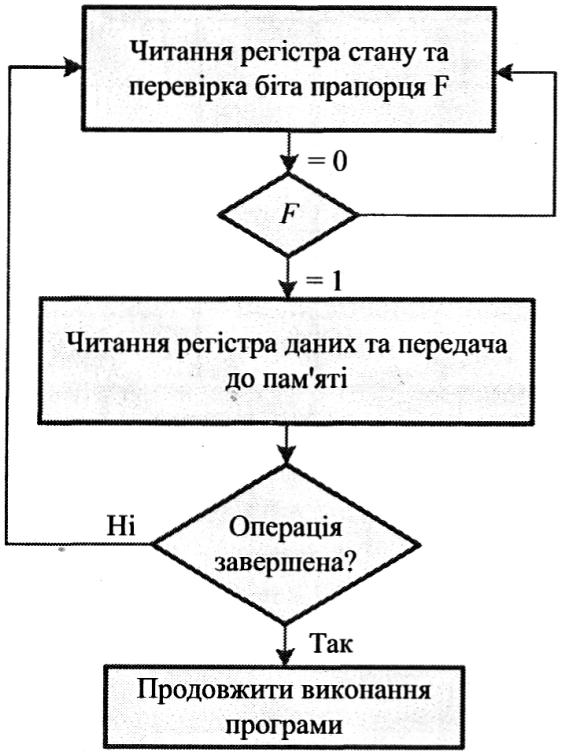

Рисунок 2 - Алгоритм програмно-керованого вводу/виводу

Як тільки процесор виявляє умову готовності даних, він діє згідно з вказівками команд з програми відповідного пристрою введення-виведення.

Наприклад, при пересиланні даних з клавіатури послідовність операцій має бути наступною. Процесор постійно перевіряє вміст прапорця F регістра РКС, аж поки він не встановиться в одиницю, після чого відбувається пересилання даного з регістра даних РД в один з регістрів процесора, номер якого вказується в команді.

Аналогічно при пересиланні даних із процесора на дисплей, процесор постійно перевіряє вміст прапорця F регістра РКС, аж поки він не встановиться в одиницю, після чого відбувається пересилання даного з одного з регістрів процесора, номер якого вказується в команді, до регістра даних РД. Вигода від використання цього підходу полягає в тому, що забезпечується програмний контроль над поведінкою кожного пристрою, але недоліком його є те, що він не виконує інших функцій, поки є процес введення — виведення. Тобто продуктивність процесора знижується до рівня продуктивності пристроїв введення — виведення. Внаслідок цих обмежень, програмно-кероване введення виведення найкраще підходить для спеціальних систем, де вимоги до продуктивності процесора невисокі.

2 Кероване перериваннями введення-виведення. Функції системи переривання програм

Роботу комп’ютера можна представити як послідовність подій, частина яких є програмно визначеною, а інша частина подій є програмно незалежною, тобто моменти виникнення подій наперед невідомі. До програмно незалежних подій можна віднести:

1. Зупинення виконання програми, пов’язане з перевищенням виділеного часу для її виконання;
2. Запити оператора, який вирішив внести зміни до програми під час її виконання;
3. Запити периферійних пристроїв по завершення операцій введення-виведення або потреби проведення додаткових операцій по їх обслуговуванню;
4. Переповнення розрядної сітки;
5. Ділення на нуль;
6. Вихід за дозволені границі пам’яті.

Останні три названі вище та подібні події, які ініційовані апаратними засобами та фіксуються програмою, виділяють в окремий клас подій зважаючи на їх важливість та те, що вони не є передбачуваними. Програми взаємодіють з операційною системою комп’ютера через ці події.

Комп’ютер реагує на програмно визначені події відповідно до вказівок програми. Для реакції на програмно незалежні події в комп’ютер введено спеціальні апаратно-програмні засоби, які дістали назву системи переривання програм (СПП). Ці засоби є невід’ємною частиною сучасних комп’ютерів. Без них поява будь-якої програмно незалежної події приведе до необхідності повторного запуску програми.

Перериванням програми називають властивість комп'ютера тимчасово переривати виконання поточної програми на час виконання деяких подій і передавати керування програмі, яка спеціально передбачена для даної події.

Запитами переривання називають сигнали, які сповіщають про появу програмно незалежних подій.

Перивальними програмами називають програми, на виконання яких є запити, тобто такими, що переривають інші програми.

Перериваними програмами називають програми, які виконувались до появи запитів, тобто такими, що перериваються.

Часова діаграма процесу переривання наведена на рис. 3, де t_p — час реакції системи переривання на запит переривання; t_3 — час запам'ятовування стану перериваної програми, $t_{впп}$ — час виконання перивальної програми, $t_в$ — час виходу з перивальної програми та повернення до перериваної програми (відновлення її стану).

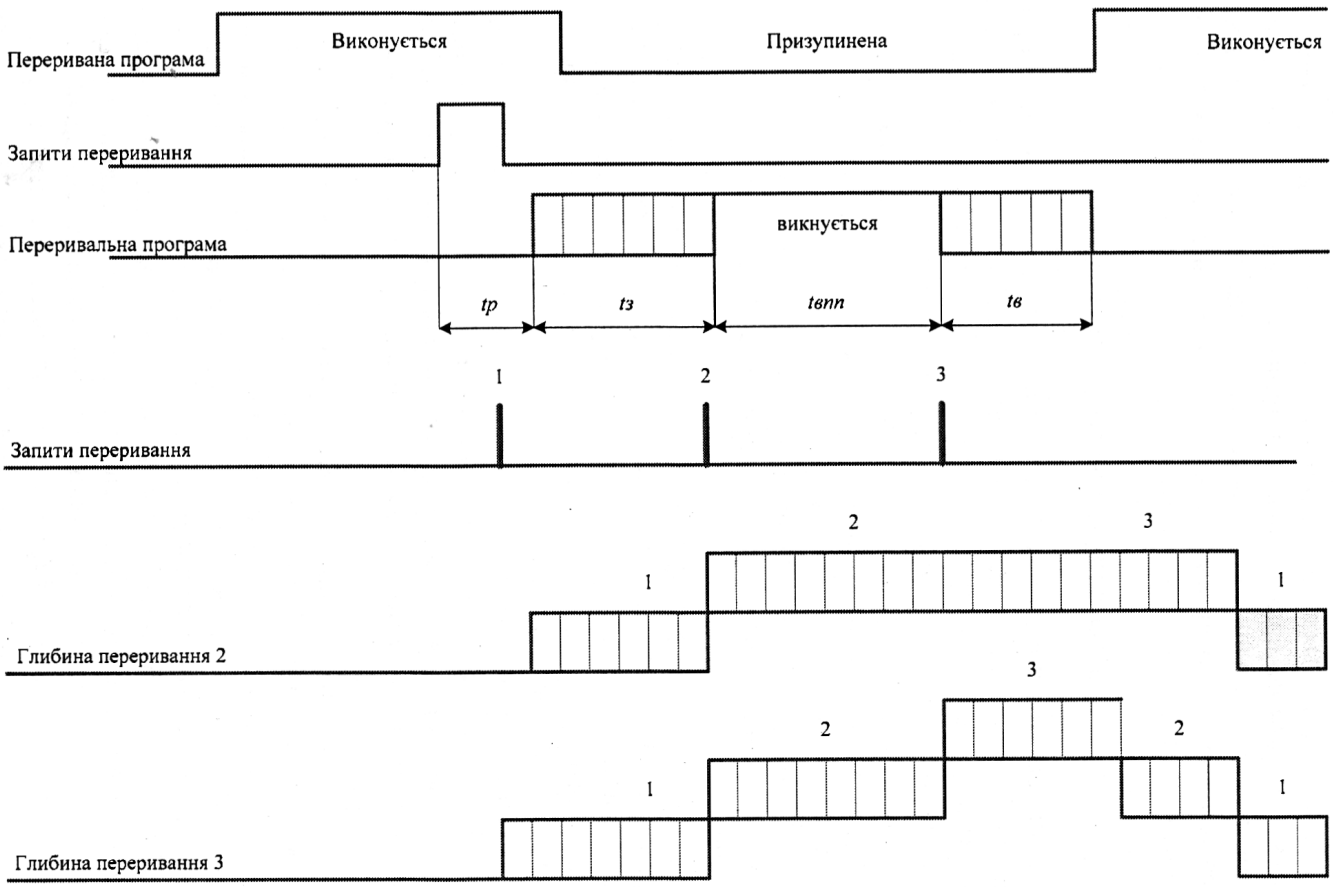

Рисунок 3 - Часова діаграма процесу переривання

Характеристики системи переривання програм

До основних характеристик системи переривання програм належать наступні:

1. Час реакції t_p, який показує як швидко система реагує на запит переривання;

2. Час обслуговування переривання — затрати часу на вхід в переривальну програму та вихід з переривальної програми $t_0 = t_з + t_в$;

3. Глибина переривання — максимальне число програм, які можуть переривати одна одну;

4. Кількість входів приймання переривань.

На рис. 3. показано переривання двох програм запитами, причому перша програма має глибину переривання 2, а друга - 3.

В першому випадку, коли поступив запит переривання 2, виконання першої програми переривається і виконується друга програма, яка виконується до завершення не дивлячись на те, що під час її виконання поступив запит 3. Це пояснюється тим, що система переривання програм не дозволяє виконання наступного переривання. Після закінчення виконання програми 2 виконується програма 3 (звичайно, якщо вона має вищий пріоритет, ніж програма 1), а тоді відбувається повернення до виконання програми 1.

В другому випадку з поступленням кожного запиту переривання він обслуговується без затримки. Чим більша глибина переривання, тим краще враховуються пріоритети запитів переривання. .Якщо tp або t0 більше за час надходження запиту переривання від того ж джерела, то виникає так зване насичення системи переривання, чого не можна допускати.

Вхід в переривальну програму

При поступленні запиту переривання системи переривання програм повинна спочатку визначити допустимий момент переривання і початкову адресу переривальної програми. При цьому, для забезпечення повернення до перериваної програми, вміст елементів пам’яті комп’ютера, в першу чергу лічильника команд та регістра станів, в момент її переривання повинен бути збережений з тим, щоб після завершення виконання переривальної команди відновити цей вміст та продовжити виконання перериваної програми.

Існують наступні способи визначення допустимого моменту переривання:

1. Включення в кожну команду програми розряду дозволу переривання. В цьому випадку програміст сам слідкує за тим, щоб переривання не зашкодило виконанню перериваної програми. Недолік цього способу — великий час реакції t_p, а також створення додаткових клопотів програмісту;

2. Переривання дозволено після закінчення будь-якої команди. Недоліки цього способу: необхідність запам’ятовувати стани програмно доступних регістрів процесора та комірок пам’яті, включаючи і кеш-пам’ять, а також малий час реакції t_p, який є рівним часу виконання команди;

3. Переривання дозволено після кожного такту команди. Тут час реакції t_p є мінімальним. Недолік цього способу: необхідність збереження станів не тільки програмно доступних регістрів та комірок пам’яті, включаючи і кеш-пам’ять, а й недоступних програм і вузлів, зокрема вмісту лічильника тактів.

Другий і третій способи використовуються в комп'ютерах найчастіше. При цьому потрібно зазначити, що для їх реалізації в процесор необхідно ввести спеціальні апаратні та програмні засоби для проведення операцій із збереження та відновлення після завершення переривальної програми вмісту програмно доступних регістрів та комірок основної пам'яті, в яких зберігалася програма, включаючи і кеш-пам'ять. Як ми вже наголошували раніше, для обробки переривань в пам'яті мікрокоманд пристрою керування для цього записані спеціальні мікропрограми.

Після визначення допустимого моменту переривання потрібно визначити початкову адресу переривальної програми з тим, щоб приступити до її виконання. Зрозуміло, що джерел переривання може бути декілька. Це, наприклад, запити від пристроїв введення-виведення, від джерела живлення, від засобів захисту пам'яті, від АЛП. Тому система переривання програм повинна вміти розпізнати причину переривання та вказати на початкову адресу програми, яка обробляє це переривання. Існують наступні способи визначення початкової адреси переривальної програми:

1. Програмне розпізнавання причин переривання. Запит переривання спочатку фіксується, а потім переривальна програма проводить опитування джерел переривання;

2. Апаратне розпізнавання причин переривання. Тут кожному джерелу переривання, або групі джерел переривання, ставиться у відповідність своя адреса початку переривальної програми. Кількість початкових адрес рівна кількості рівнів системи переривання. Для реалізації цього способу потрібні додаткові апаратні засоби.

Зазвичай використовують комбінацію обох описаних способів, тобто апаратно-програмне розпізнавання причин переривання.

Пріоритетне обслуговування переривання

Джерел переривання в комп'ютері є багато, причому запити від них можуть поступати одночасно, а тому потрібно встановити пріоритетність їх обслуговування, яка має два типи в системі переривання:

1. Пріоритет між запитами переривань;
2. Пріоритет між переривальними програмами.

Перший пріоритет потрібний головним чином для селекції: вибирається один запит з багатьох, оскільки комп'ютер не завжди може відразу прийняти декілька запитів. Він може бути реалізований методом послідовного пошуку, або шляхом паралельного дешифрування всіх запитів, та вибору з них одного за якимось правилом, наприклад, з найменшим номером серед прийнятих запитів.

Другий пріоритет визначає фактичний порядок, в якому переривальні програми будуть виконуватись.

Вибір пріоритету програм в багаторівневій системі переривання програм є досить складною задачею. Часто пріоритетність програм потрібно поміняти. Тому встановлення пріоритетів має бути програмно керованим.

В сучасних комп’ютерах використовується метод встановлення програмно-керованого пріоритету, який забезпечує присвоєння пріоритету програмістом, який називають маскою переривань. Маска переривань — це двійкове число, кожний розряд якого ставиться у відповідність одному із рівнів переривання і дозволяє (якщо його значення рівне «1»), або забороняє (якщо його значення рівне «0») переривання від запитів, які належать до даного рівня.

Для кожної переривальної програми може бути встановлена своя маска. Маска для всіх програм зберігається в основній пам’яті та засилається в регістр маски, якщо викликається до виконання відповідна програма.

Якщо запит переривання заборонений маскою, він або ігнорується, або запам’ятовується відповідною апаратурою, про що може бути відповідна вказівка в масці переривання.

Організація повернення до перериваної програми

Повернення до перериваної програми полягає у відновленні стану цієї програми, в якому вона була в момент переривання і який був збережений в момент входу в переривальну програму.

Інформація про поточний стан перериваної програми ділиться на дві частини:

1. Основну, яка повинна запам’ятовуватись завжди при вході в переривальну програму;
2. Додаткову інформацію, необхідність запам’ятовування якої залежить від змісту переривальної програми.

До складу першої частини інформації про поточний стан перериваної програми належать:

- вміст лічильника команд;
- стан тригера роботи комп’ютера (робота/зупинка);
- маска переривання;
- код переривання — двійковий номер джерела переривання;
- вміст регістрів захисту пам’яті.

Перераховані дані компонуються в так званий вектор переривання, який займає декілька комірок основної пам’яті. В деяких комп’ютерах, наприклад в системі IBM 370, для збереження вектора переривання використовується регістр слова стану програми. В ньому зберігаються код переривання, маска переривання та інша важлива інформація. До складу другої частини інформації про бі-жучий стан перериваної програми належать вміст індексних та інших програмно доступних регістрів. В більшості випадків сам програміст визначає, що тут необхідно зберігати.

Введення-виведення за перериваннями

При програмно-керованому введенні-виведенні процесор чекає, коли зовнішній пристрій буде готовий до обміну. В цей час він не зайнятий роботою. Інший підхід — процесор постійно працює, а при необхідності введення-виведення з пристрою введення-виведення до нього по спеціальній лінії поступає запит переривання. Це і є введення-виведення за перериваннями, реалізація якого покладена на систему переривання програм, описану вище. Тому організація введення-виведення за перериваннями не відрізняється від виконання інших програм, які виконуються в комп'ютері з за діянням системи переривання програм.

Коли поступає переривання від зовнішнього пристрою, процесор припиняє виконання поточної перериваної програми і переходить до програми обслуговування цього переривання, повідомивши про це відповідний пристрій введення-виведення, який після цього знімає запит.

Введення-виведення за перериваннями можна представити як інверсію програмованого введення-виведення. Замість того, щоб процесор безупинно опитував приєднані до нього пристрої чи вони хочуть вводити інформацію, самі ці пристрої говорять процесору, коли вони мають дані для посилання. Процесор обслуговує інші задачі поки пристрій введення-виведення не виставить переривання. Про наявність переривання повідомляє відповідний прапорець в регістрі стану процесора, який називається прапорцем переривання.

Як тільки прапорець переривання встановлений, операційна система перериває виконання поточної програми, зберігаючи стан програми і її змінну інформацію. Система переривання вибирає адресний вектор, який указує на адресу службової програми введення-виведення. Після того, як процесор завершив обслуговування введення-виведення, він відновлює інформацію, яка була збережена від перерваної програми, і виконання програми поновлюється.

3 Прямий доступ до пам'яті

Прямий доступ до пам'яті від пристроїв введення-виведення здійснюється через контролер прямого доступу до пам'яті, який керує взаємодією пам'яті та пристроїв введення-виведення без участі процесора. Прямий доступ до пам'яті (ПДП) — створення прямого тракту передачі даних від зовнішніх пристроїв до пам'яті або від пам'яті до зовнішніх пристроїв. У звичайному обміні передачі між ОП і пам'яттю вимагають спочатку прийняти дані від джерела в процесор, а потім видати їх з процесора приймачеві, тобто реалізуються за два командних цикли. При ПДП дані не проходять через процесор, і передача слова проводиться за один цикл. Для реалізації ПДП розроблено спеціальні контролери прямого доступу до пам'яті (КПДП), здатні завдяки програмуванню обслуговувати ПДП з урахуванням конкретних вимог різних систем.

Взаємодію блоків процесорної системи при ПДП показано на рис. 4 процесор виконує операцію програмування КПДП, налаштовуючи його на певний режим роботи, і може читати стан контролера. Відповідні зв'язки показані штриховою лінією.

Для здійснення ПДП процесор відключено, а контролер виробляє сигнали керування обміном для ОП і ОЗП. Тракт передачі даних пов'язує ОП з ОЗП безпосередньо. Можливі два види ПДП — з блочними або поодинокими передачами.

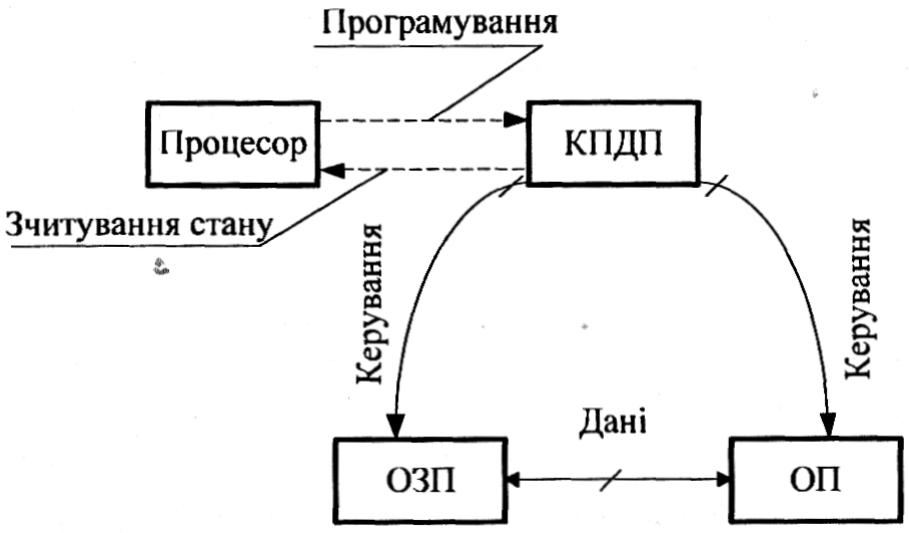

Рисунок 4 - Взаємодія блоків процесорної системи при ПДП

У першому з них робота процесора зупиняється на весь час передачі блоку даних, у другому — передачі слів у режимі ПДП перемежовуються з виконанням програми, і для передач ПДП виділяються окремі такти машинних циклів, у яких процесор не використовує системні шини. Кожен командний цикл починається з машинного циклу — вибірки команди. У цьому машинному циклі є такт декодування прийнятої процесором команди, в якому системні шини не використовуються. На цей час системні шини можна віддати для ПДП і передати одне слово.

Продуктивність комп'ютера може зрости через паралелізм процесів обміну і оброблення даних, завдяки тому, що ПДП буде для процесора «невидимим». Сам обмін з ПДП буде не швидким, темп обміну нерегулярний, оскільки тривалість циклів різних команд різна, і, крім того, ПДП може навіть сповільнити виконання програми, якщо цикл ПДП не вкладеться в інтервал, відповідний такту процесора.

На відміну від процесів переривання при ПДП, обмін виконується без участі програми, тому вміст робочих регістрів комп'ютера не порушується і на входження в режим ПДП не потрібно витрат часу (немає передачі в стек на зберігання вмісту робочих регістрів процесора). ПДП надається по завершенні поточного машинного циклу.

Прикладом КПДП може служити ВІС Intel 8237A (К580ВТ57).

Дії, що виконуються КПДП при блокових передачах, полягають у наступному:

1. Приймання відомостей про область пам'яті, відведеної для блоку даних, що належать до передачі (початкова адреса і розмір блоку);

2. Трансляція запиту на ПДП, що виходить від ОП, в запит ПДП для процесора з урахуванням маскування та пріоритетності запитів, які надходять на КПДП. Приймання сигналу підтвердження ПДП, що свідчить про те, що процесор відключився від системних шин;

3. Генерація адрес для ЗП і сигналів керування для ЗП і ОП;

4.Фіксація завершеності ПДП;

5. Зняття запиту ПДП з відповідного входу процесора і повернення керування основній програмі.

Можливості КПДП дозволяють організувати обмін типу «пам’ять-пам’ять», тобто вирішувати задачу переміщення блоку даних в адресному просторі системи.

4 Вводу-виводу під керуванням периферійних процесорів

Принципи введення-виведення

Програмно-кероване введення-виведення передбачає передачу одного байта за один раз. Кероване перериваннями введення-виведення також може маніпулювати даними по одному байту за один раз або малими блоками, залежно від виду пристрою, що бере участь у введенні-виведенні. Повільніші пристрої, як наприклад клавіатура, генерують більшу кількість переривань відносно кількості переданих байтів, ніж принтери. Методи прямого доступу до пам’яті є блочно-орієнтованими і вимагають меншої участі процесора, ніж при керованому перериваннями введенні-виведенні. Такі підходи до організації введення-виведення допустимі для малих систем, орієнтованих на одного користувача, проте, вони не є ефективними для великих систем, орієнтованих на багатьох користувачів, як наприклад, великі універсальні комп’ютери — мейнфрейми. Більшість мейнфреймів використовують інтелектуальний вид інтерфейсу прямого доступу до пам’яті, відомого як **канальне введення-виведення**.

При використанні **канального введення-виведення** один або більше процесорів введення-виведення керують різними трактами введення-виведення. Тракти для «повільних» пристроїв, як, наприклад, термінали і принтери, можуть комбінуватися, дозволяючи керування деякою кількістю цих пристроїв тільки одним процесором введення-виведення. Процесори введення-виведення оптимізують для організації виконання операцій введення-виведення. На відміну від схем прямого доступу до пам’яті, процесори введення-виведення мають здатність виконувати програми, які включають арифметично-логічні команди та команди переходу. На рис. 5 наведена спрощена конфігурація канального введення-виведення.

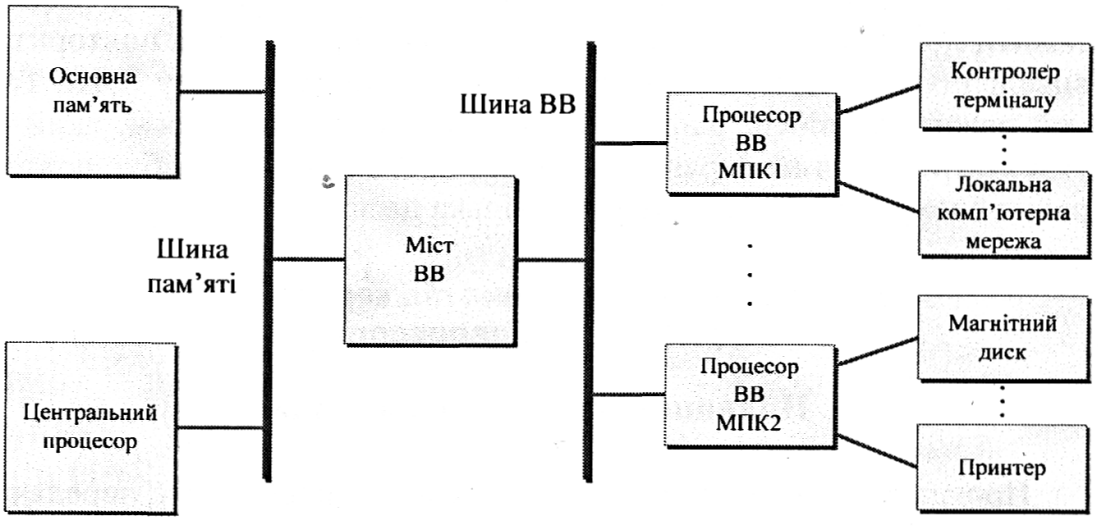

Рисунок 5 - Структура канального вводу-виводу

Процесори введення-виведення виконують програми, які розміщені центральним процесором в основній пам’яті. Ці програми, складаючись з серій слів команд каналу, включають не тільки фактичні команди пересилання, але і команди, які керують пристроями введення-виведення. Як тільки програма введення-виведення буде розміщена в пам’яті, процесор комп’ютера видає команду старту підканалу, інформуючи процесор введення-виведення про місце розташування програми в пам’яті. Після того, як процесор введення-виведення завершить свою роботу, він розміщує інформацію про це в пам’яті та відправляє переривання центральному процесору комп’ютера. Центральний процесор отримує інформацію про завершення і вживає заходи по виходу з переривання.

Головна відмінність між автономним прямим доступом до пам’яті і канальним введенням-виведенням полягає в більш широких можливостях процесора введення-виведення. Процесор введення-виведення забезпечує дотримання протоколів, видає команди пристрою, транслює коди зовнішньої пам’яті до кодів основної пам’яті, і може перенести повні файли або групи файлів незалежно від центрального процесора. Центральному процесору комп’ютера належить лише створити команди програми для операцій введення-виведення і вказати процесору введення-виведення, де їх знайти.

Подібно до автономного прямого доступу до пам’яті, процесор введення-виведення повинен забрати від центрального процесора цикли звернення до пам’яті. На відміну від автономного прямого доступу до пам’яті, системи введення-виведення обладнані окремими шинами введення-виведення, які допомагають звільнити центральний процесор від виконання введення-виведення. Копіюючи файл, наприклад, з магнітного диску на принтер, процесор введення-виведення використовує системну шину пам’яті тільки для вибору його команд з основної пам’яті. Остання частина передачі проводиться використовуючи лише шину введення-виведення. Завдяки широким функціональним можливостям та шинній ізоляції, канальне введення-виведення використовується в середовищах діалогової високопродуктивної обробки запитів, де його вартість і складність можуть бути виправдані.

Причини застосування каналів введення-виведення

- Створення комп’ютерних систем із змінним складом обладнання;
- Забезпечення паралельної роботи пристроїв введення-виведення як досамих себе, так і по відношенню до процесора;
- Уніфікація програмування введення-виведення;
- Забезпечення реакції обчислювальних засобів на багатоманітність ситуацій, які виникають в пристроях введення-виведення ;
- Узгодження швидкості роботи процесора і пристроїв введення-виведення;
- Розвантаження процесора від виконання функцій керування обміном.

Виходячи із вище викладеного під каналом розуміють спеціальний уніфікований пристрій керування вводом-виводом, який забезпечує обмін інформацією між ядром комп’ютера (процесором і основною пам’яттю), та периферійними пристроями з здійсненням паралельної роботи змінного набору периферійних пристроїв різних типів. При цьому в комп’ютері повинні бути уніфіковані шини, схеми підключення, сигнали та алгоритми керування обміном, формат і множина команд процесора.