01.1. Організація та способи управління пам’яттю

- фізична організація пристроїв пам’яті → ієрархія
- рівні управління пам’яттю
- багатозадачність → захист пам’яті
- логічні vs. фізичні адреси → перетворення адрес
- свопінг (swapping)
- сегментна організація пам’яті
- сторінкова огранізація пам’яті
- віртуальна пам’ять

01.2. Організація та способи управління пам’яттю

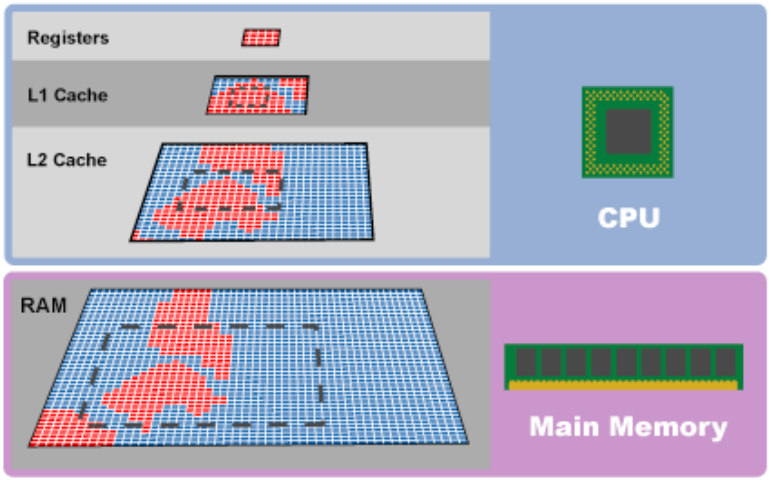
1. Назва: оперативна (основна) пам’ять (Main Memory, MM).

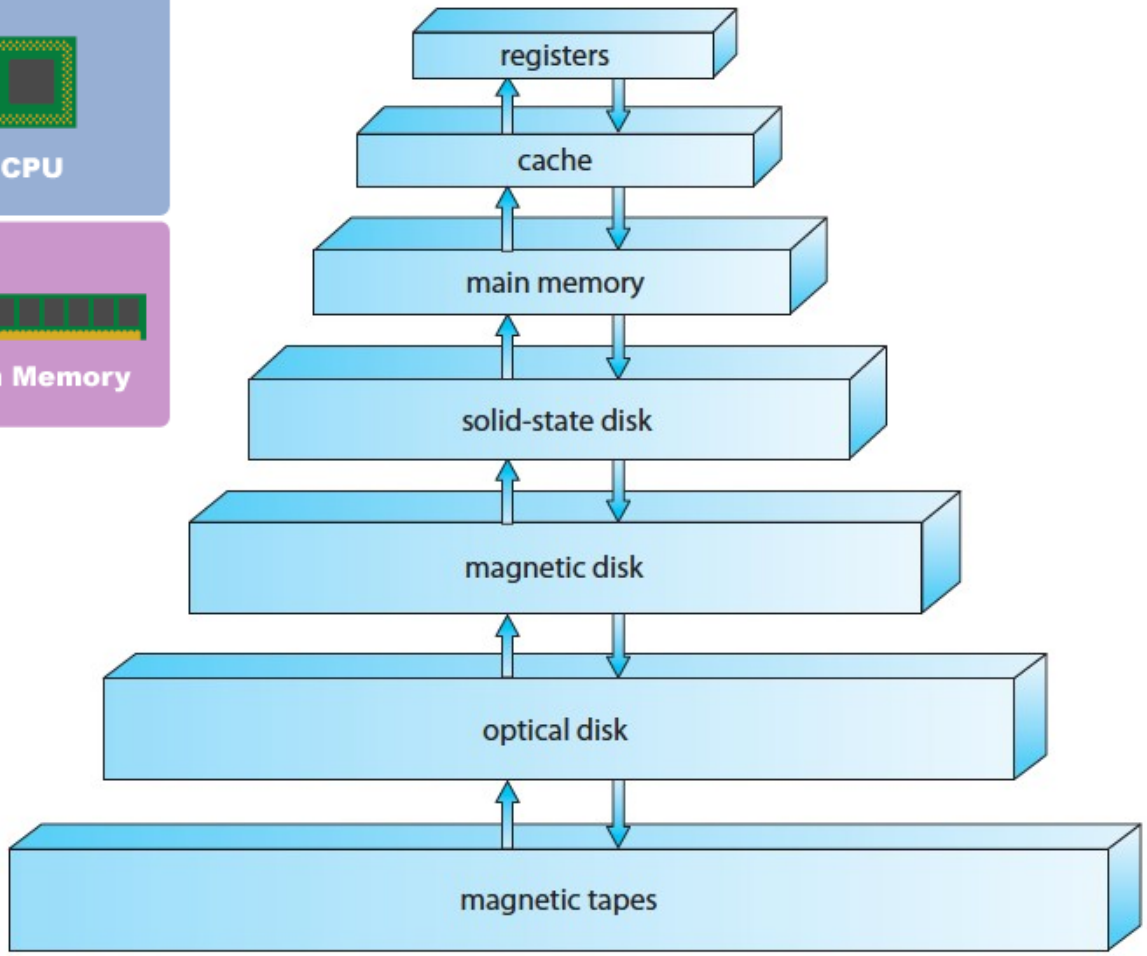
2. В ахітектурі фон Неймана пам’ять - це лінійний масив комірок (індекс=адреса) загального призначення (в них розміщується як код, так і дані).

3. Фізична організація пристроїв пам’яті: принцип ієрархії (↓швидкість доступу+ціна, ↑ємність пристрою пам’яті).

4. Мета: забезпечити одночасно швидкий доступ до пам’яті та її великий об’єм.

01.3. Організація та способи управління пам’яттю





01.4. Організація та способи управління пам’яттю

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

Порівняння характеристик пристроїв пам’яті

01.5. Організація та способи управління пам’яттю

Основні задачі організації пристроїв пам’яті у єдину систему:

1. Забезпечення максимально можливої швидкості доступу до даних.

2. Забезпечення «прозорості» ієрархії пам’яті (приховування від обчислювальних процесів факту її існування).

01.6. Організація та способи управління пам’яттю

- Специфічний клас оптимізаційних задач: виявлення даних, які використовуються найчастіше або будуть використані в найближчий час, і розміщення їх в більш швидких пристроях пам’яті.

- Ці оптимізаційні задачі вирішуються на основі принципу локальності звертань до пам’яті (principle of locality).

01.7. Організація та способи управління пам’яттю

1. Управління пам’яттю може здійснюватись 1) апаратно та 2) програмно. В сучасних комп’ютерних системах переважно застосовується апаратне управління.

2. «Взаємодія» між найбільш швидкими пристроями пам’яті (регістри процесора ↔ кеш 1,2,3 рівня ↔ Main Memory) знаходиться під управлінням спеціальних апаратних засобів.

3. «Взаємодія» оперативної пам’яті та зовнішньої пам’яті знаходиться під управлінням ОС.

01.8. Організація та способи управління пам’яттю

Рівні управління пам’ятю:

1) Управління пам’яттю на апаратному рівні (memory management hardware: RAM, MMU, DMA).

2) Управління пам’яттю на рівні ОС (OS memory management: memory protection, virtual memory, shared memory, etc.).

3) Управління пам’яттю на рівні прикладних програм (application memory management: allocation, deallocation, garbage collection).

01.9. Організація та способи управління пам’яттю: Захист пам’яті

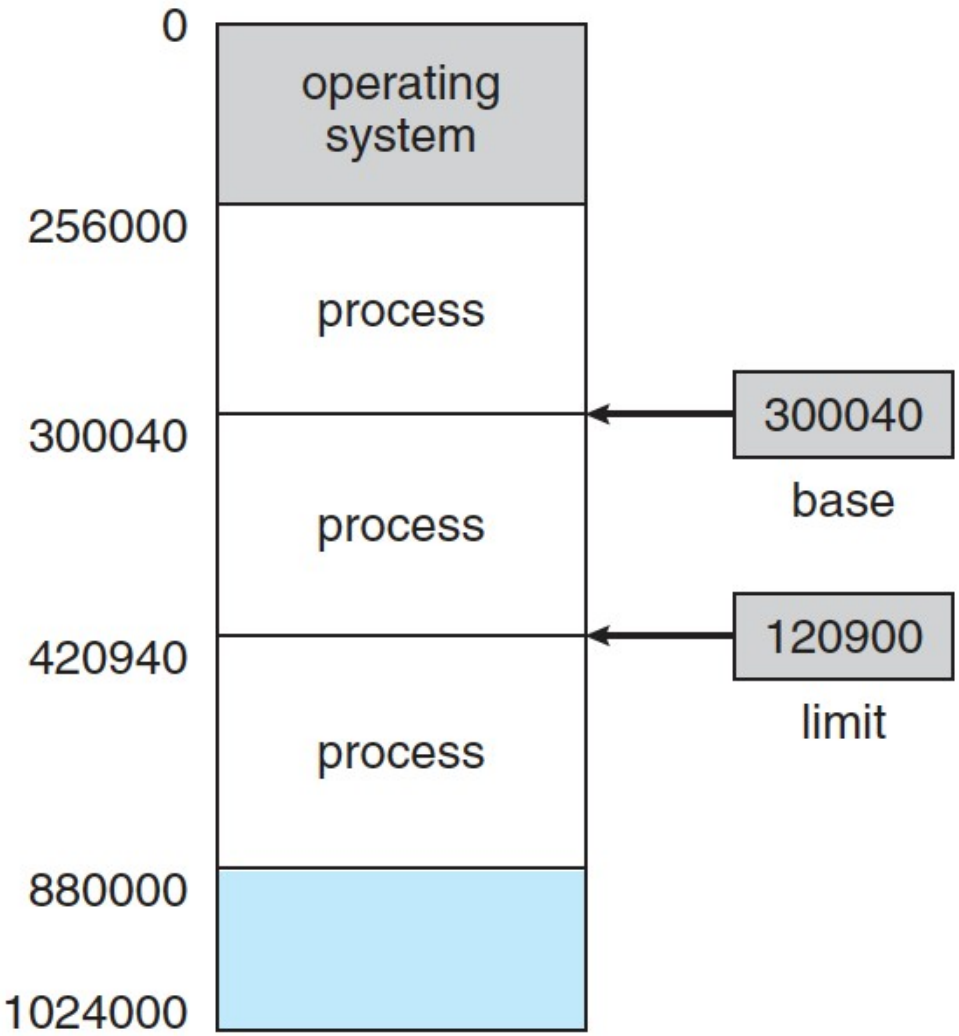
1. Внаслідок багатозадачності (розпаралелення обчислень) в оперативній пам’яті одночасно розміщується виконавчий код та дані багатьох обчислювальних процесів.

2. Проблема: окремий процес може помилково або «навмисно» пошкодити код або дані інших процесів (в тому числі код або дані ядра ОС і системних процесів).

3. Рішення: захист пам’яті (memory protection) — захист ділянок пам’яті одних процесів від несанкціонованого доступу інших процесів.

4. Приклад реалізації: апаратний захист пам’яті з використанням базового регістра (base register) та регістра обмеження (limit register).

01.10. Організація та способи управління пам’яттю: Захист пам’яті



01.11. Організація та способи управління пам’яттю: Захист пам’яті

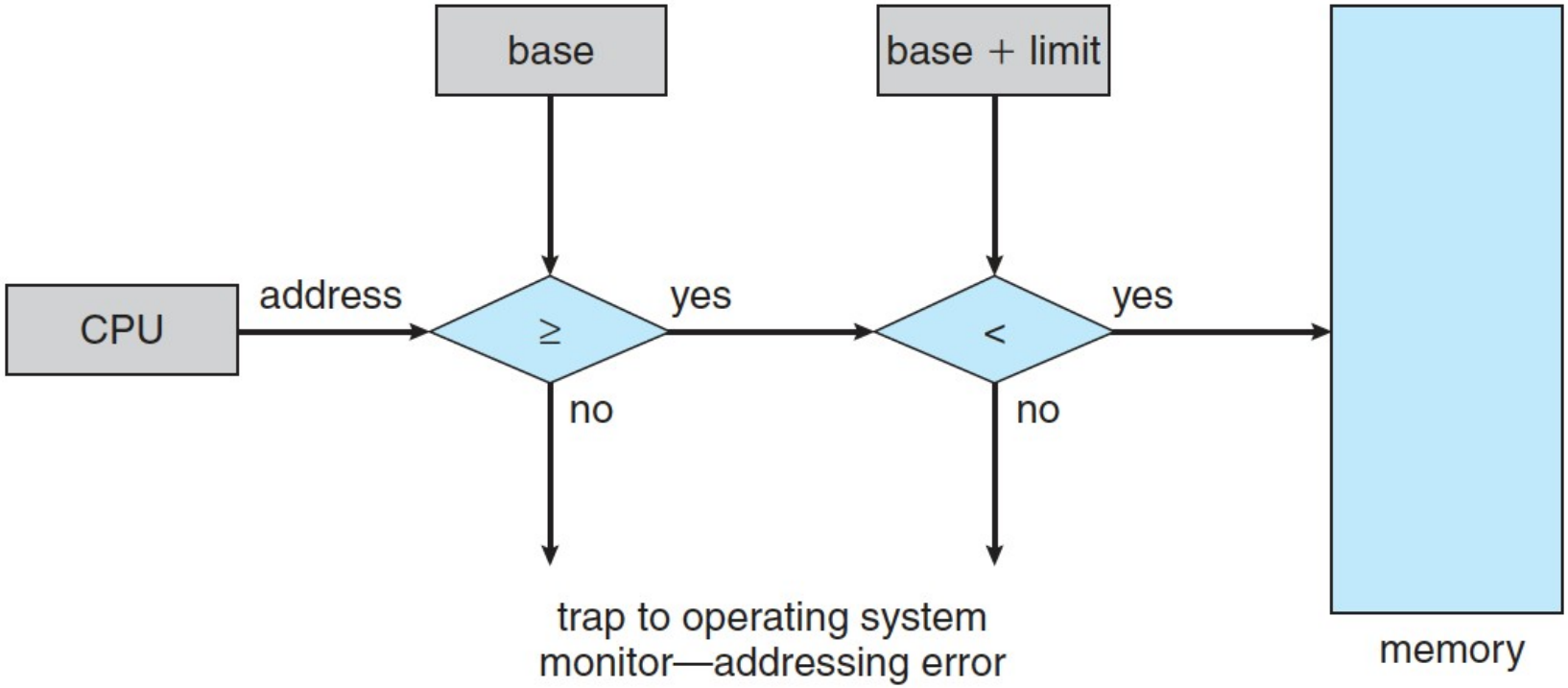


Схема захисту пам’яті з використанням base register та limit register

01.12. Організація та способи управління пам’яттю: Захист пам’яті

5. Апаратне забезпечення CPU аналізує кожну адресу комірки пам’яті, згенеровану в режимі користувача.

6. Кожна спроба користувацького процесу звернутись до області пам’яті ядра ОС або іншого користувацького процесу призводить до фатальної помилки (зупинки процеса).

7. Значення базового регістра та регістра обмеження можуть бути встановлені тільки ядром ОС за допомогою відповідної привілейованої інструкції (яка може бути виконана тільки в режимі ядра).

8. Ядро ОС має необмежений доступ до усіх областей оперативної пам’яті. Користувацький процес має доступ тільки до власної області пам’яті.

01.13. Організація та способи управління пам’яттю: перетворення адрес

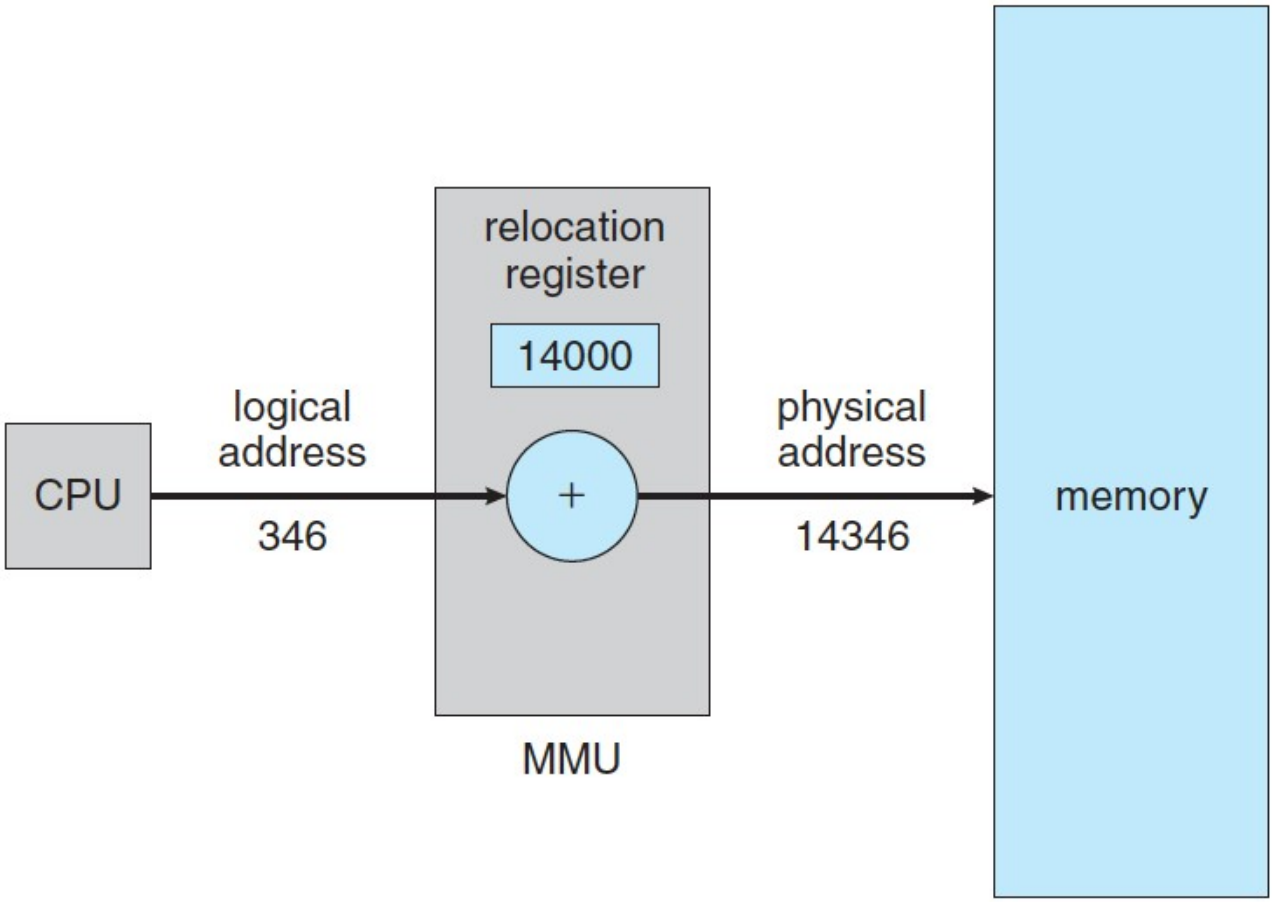
В сучасних КС та ОС розрізняють:
1) логічний адресний простір (Logical Address Space) та
2) фізичний адресний простір (Physical Address Space).

Принцип: CPU оперує логічними (віртуальними) адресами, а в Main Memory використовуються фізичні («справжні») адреси.

01.14. Організація та способи управління пам’яттю: перетворення адрес

- Такий підхід робить можливим
  1) виконувати прив’язку адрес (address binding) в режимі виконання (execution time),
  2) виконувати динамічне завантаження коду програм (dynamic loading),
  3) динамічну прив’язку до системних бібліотек (dynamic linking).
- Перетворення логічної адреси у фізичну виконує блок управління пам’яттю (memory-management unit, MMU).
- Приклад: простий варіант перетворення адрес з використанням регістра перерозміщення (relocation register).

01.15. Організація та способи управління пам’яттю: перетворення адрес



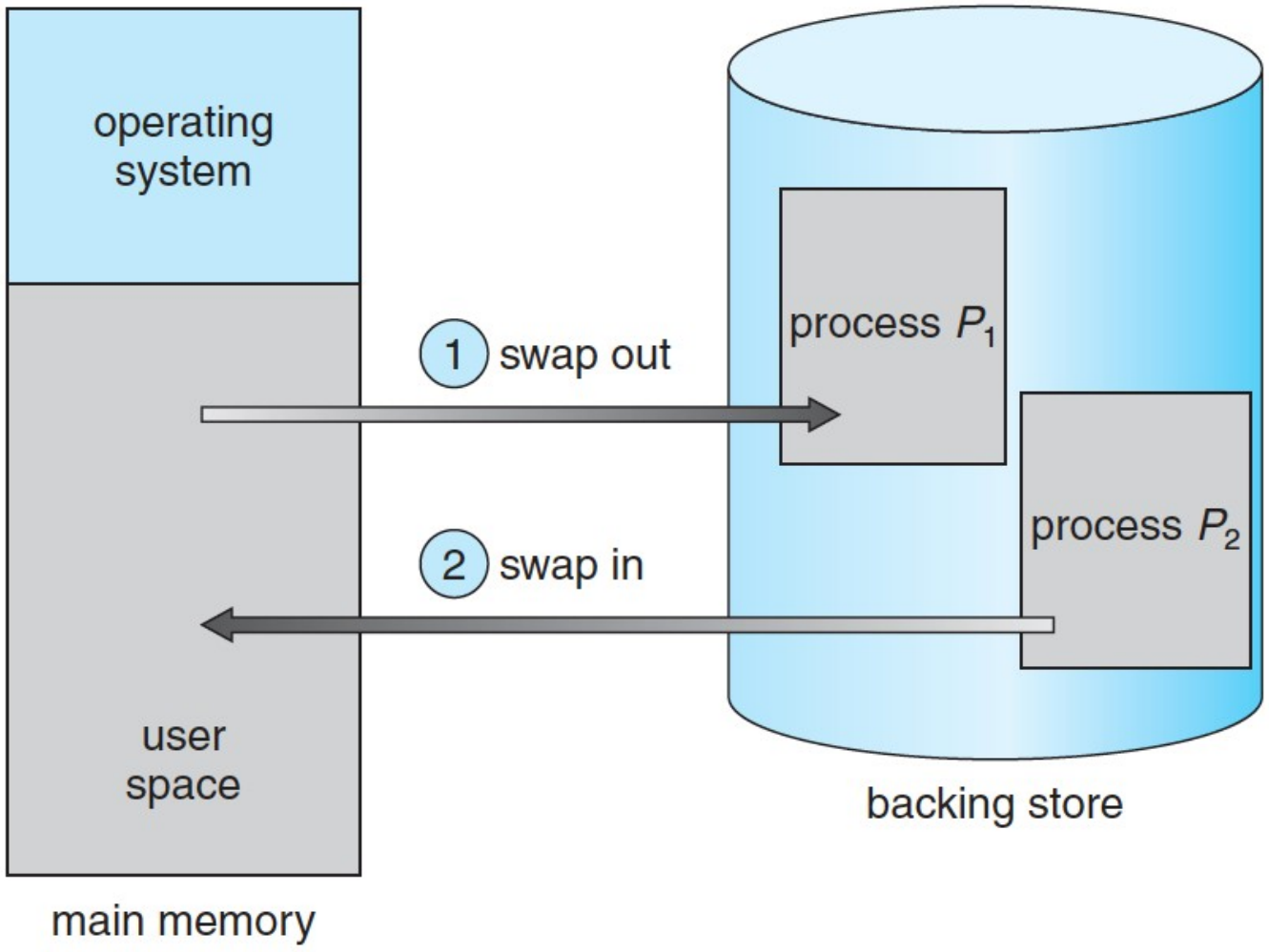
01.16. Організація та способи управління пам’яттю: Свопінг (swapping)

1. Свопінг (swapping) — це переміщення області пам’яті процеса (код, дані, стек) між оперативною та зовнішньою пам’яттю.

2. Операції: swap out, swap in.

3. Мета: забезпечити можливість одночасного запуску багатьох процесів, сумарний об’єм пам’яті яких міг би бути більшим за об’єм оперативної пам’яті.

01.17. Організація та способи управління пам'яттю: Свопінг (swapping)



01.18. Організація та способи управління пам’яттю: Свопінг (swapping)

4. Класичний свопінг (старі версії UNIX): переноситься вся область пам’яті процеса.

5. В сучасних ОС використовується варіант свопінгу на основі сторінкової організації пам’яті: підкачка сторінок пам’яті (paging) → у зовнішню пам’ять вивантажується лише деяка частина пам’яті процесу.

6. Зі зростанням об’єму оперативної пам’яті механізм свопінгу застосовується все рідше.

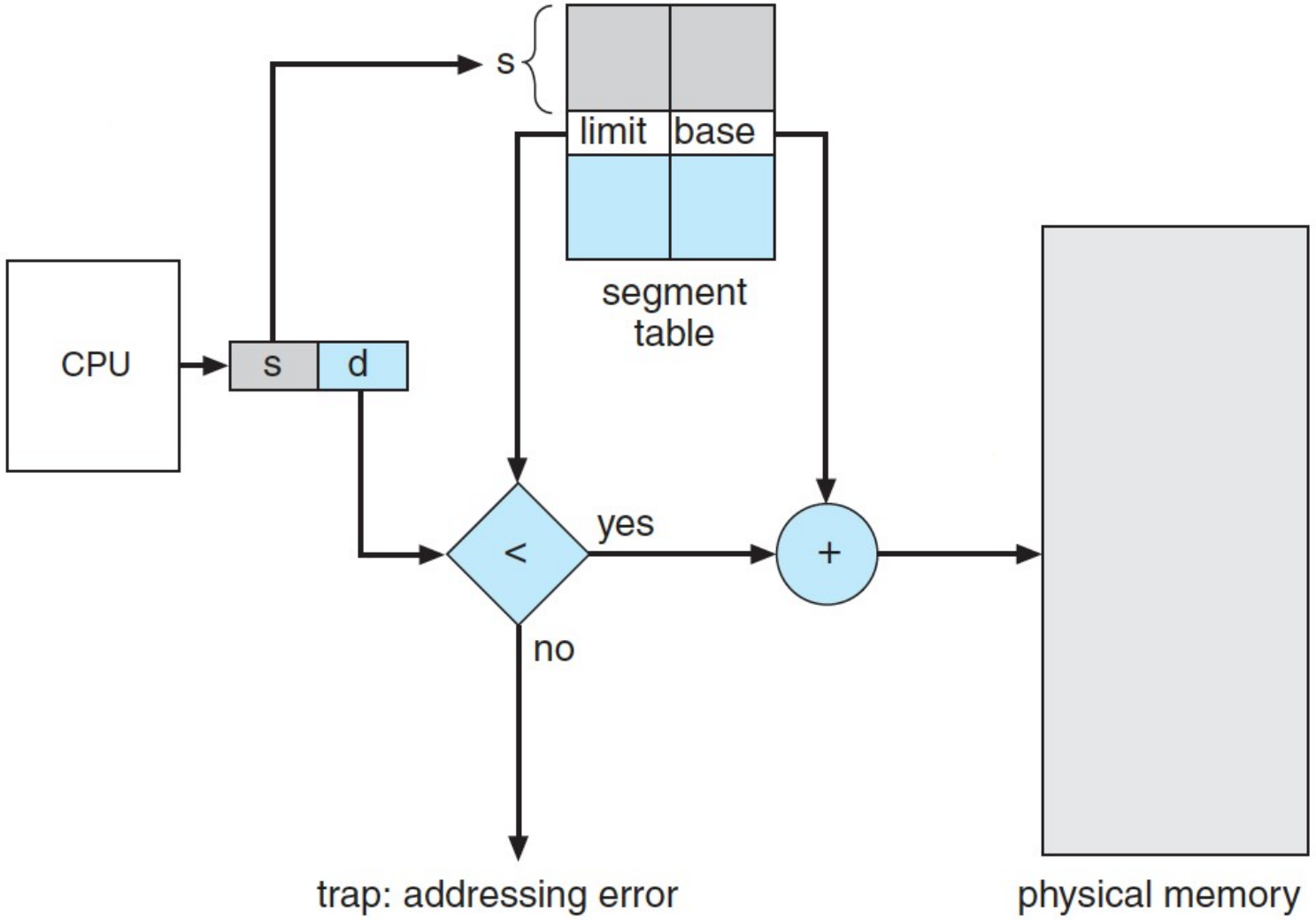
01.19. Організація та способи управління пам’яттю: Сегментна організація пам’яті

1. Segmentation = сегментна організація пам’яті: пам’ять процеса ділиться на сегменти різного функціонального призначення (коду, даних, стеку і т.п.), які можуть мати різний розмір.

2. Таблиця сегментів (segment table) містить пари значень: початок сегменту (segment base) та його розмір (segment limit).

3. Логічна адреса складається з двох частин:
s = номер сегмента, d = зміщення в сегменті.

01.20. Організація та способи управління пам’яттю: Сегментна організація пам’яті



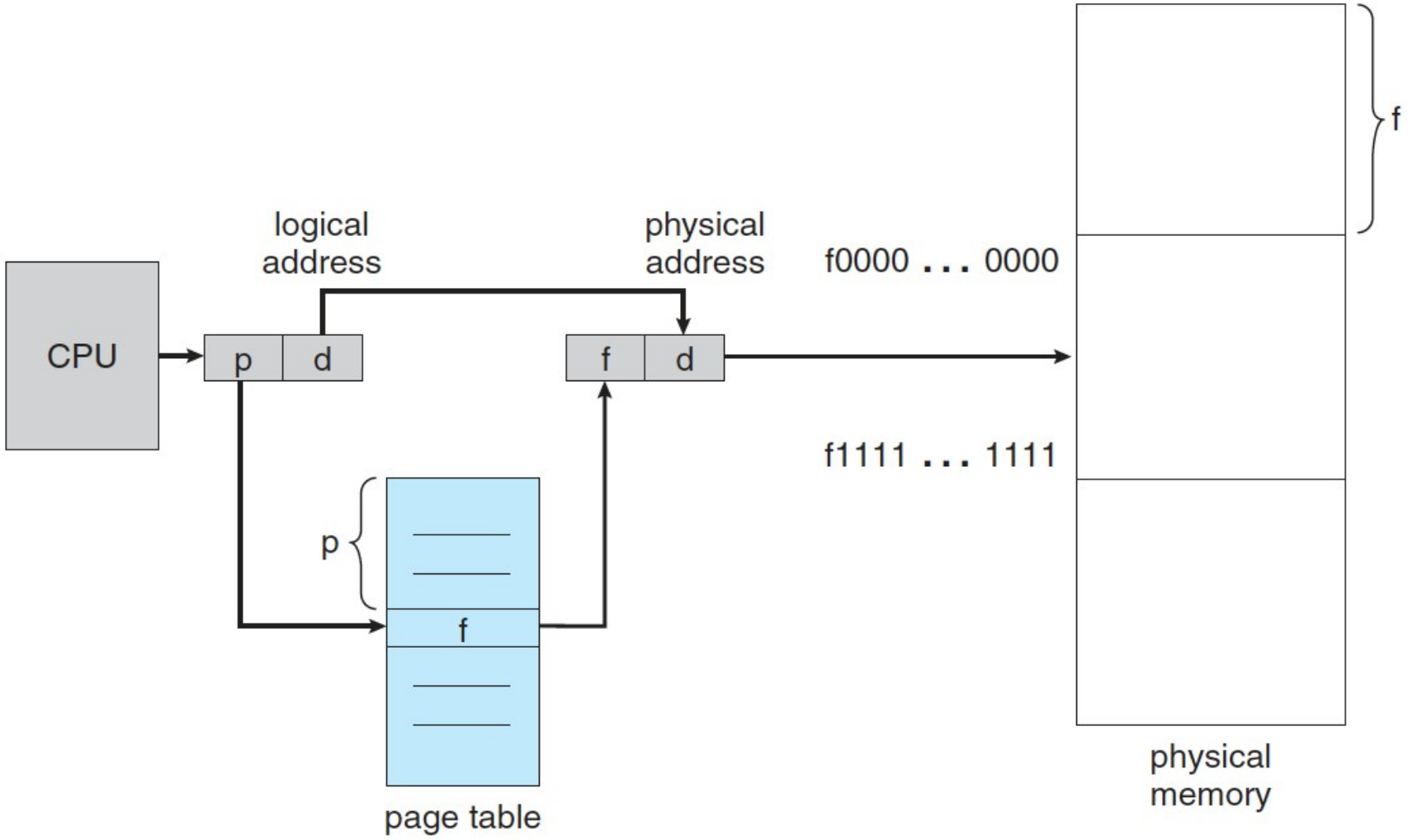
01.21. Організація та способи управління пам’яттю: Сторінкова організація пам’яті

1. Paging = сторінкова організація пам’яті: вся пам’ять ділиться на блоки однакового розміру і «універсального» призначення.

2. Блок логічної пам’яті (пам’яті процеса) називається сторінка (page). Блок фізичної пам’яті називається фрейм (frame). Розмір сторінки та фрейма однаковий. Типовий розмір сторінки: 4 kB.

3. Логічна адреса складається з двох частин:
p = номер сторінки, d = зміщення в сторінці.

01.22. Організація та способи управління пам’яттю: Сторінкова організація пам’яті



01.23. Організація та способи управління пам’яттю: Сторінкова організація пам’яті

4. Таблиця сторінок (page table) містить пари значень: індекс (=номер сторінки), початок (base address) відповідного фрейму у фізичній пам’яті.

5. Для контролю за станом оперативної пам’яті ОС використовує таблицю фреймів (frame table), в якій для кожного фрейма вказано його стан: вільний чи зайнятий (+ якою сторінкою якого процеса).

6. Сторінкова організація пам’яті вирішує багато проблем (фрагментація оперативної пам’яті, фрагментація ділянки свопінгу, розділення логічного та фізичного адресного простору та ін.).

02.01. Віртуальна пам’ять (virtual memory)

1. Мета: забезпечити «необмежений» логічний адресний простір (щоб логіка роботи програми не залежала від розміру оперативної пам’яті).

2. Визначення №1: віртуальна пам’ять — це механізм, що дозволяє виконувати обчислювальні процеси, які не знаходяться повністю у оперативній пам’яті [Silberschatz].

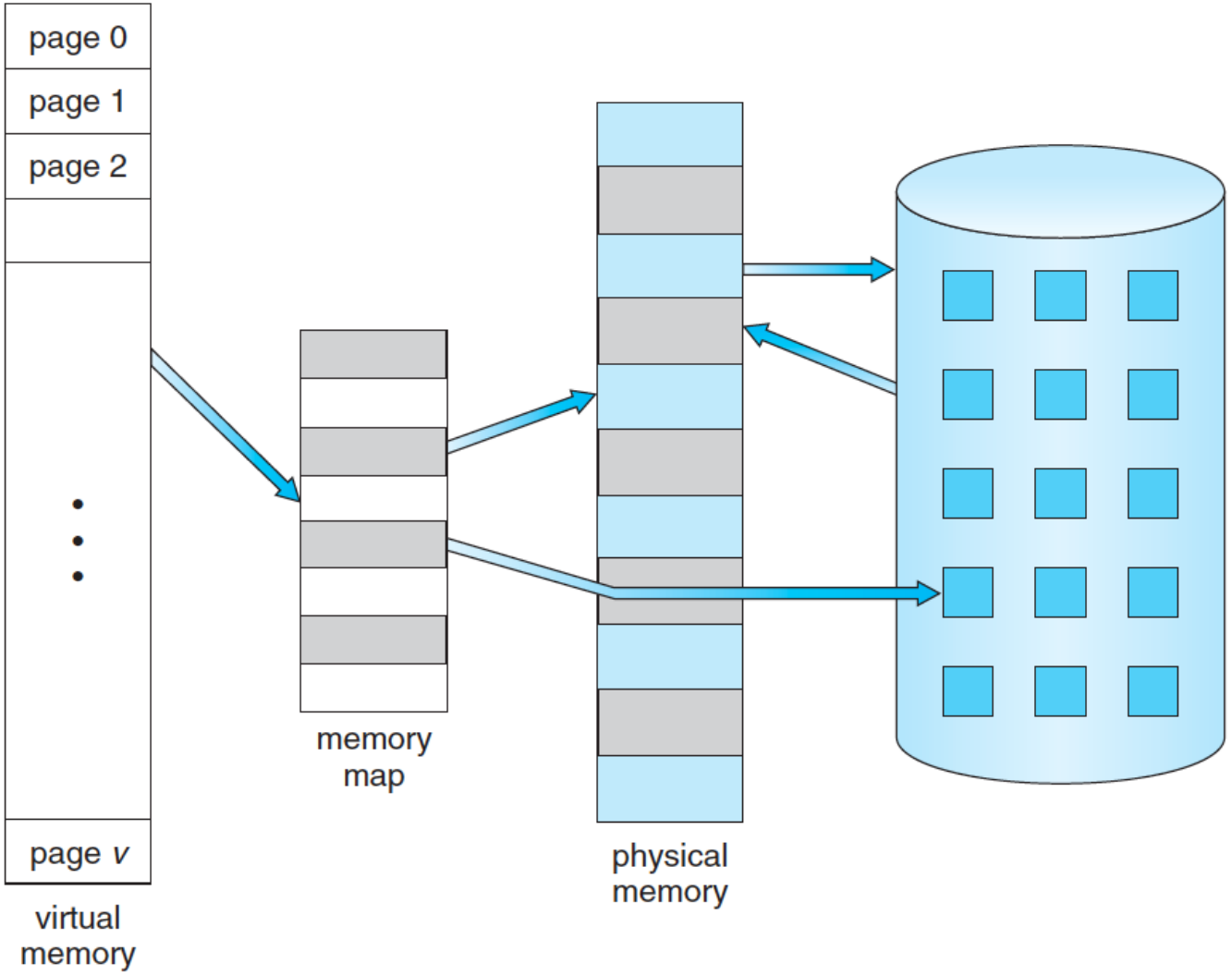
02.02. Віртуальна пам’ять (virtual memory)

3. Наприклад, механізм віртуальної пам’яті дозволяє виконувати програму, розмір якої більше об’єму оперативної пам’яті.

4. Визначення №2: віртуальна пам’ять — це поєднання ідей

1) захисту пам’яті,
2) розділення логічного та фізичного адресного простору та
3) свопінгу.

02.03. Віртуальна пам’ять (virtual memory)



02.04. Віртуальна пам’ять (virtual memory)

5. В сучасних ОС механізм віртуальної пам’яті реалізується на основі сторінкової організації пам’яті та завантаження сторінок за потребою (demand paging).

6. Demand paging — це варіант свопінгу, коли з пристрою зовнішньої пам’яті у оперативну пам’ять завантажуються лише ті сторінки, які потрібні для виконання даного процеса.

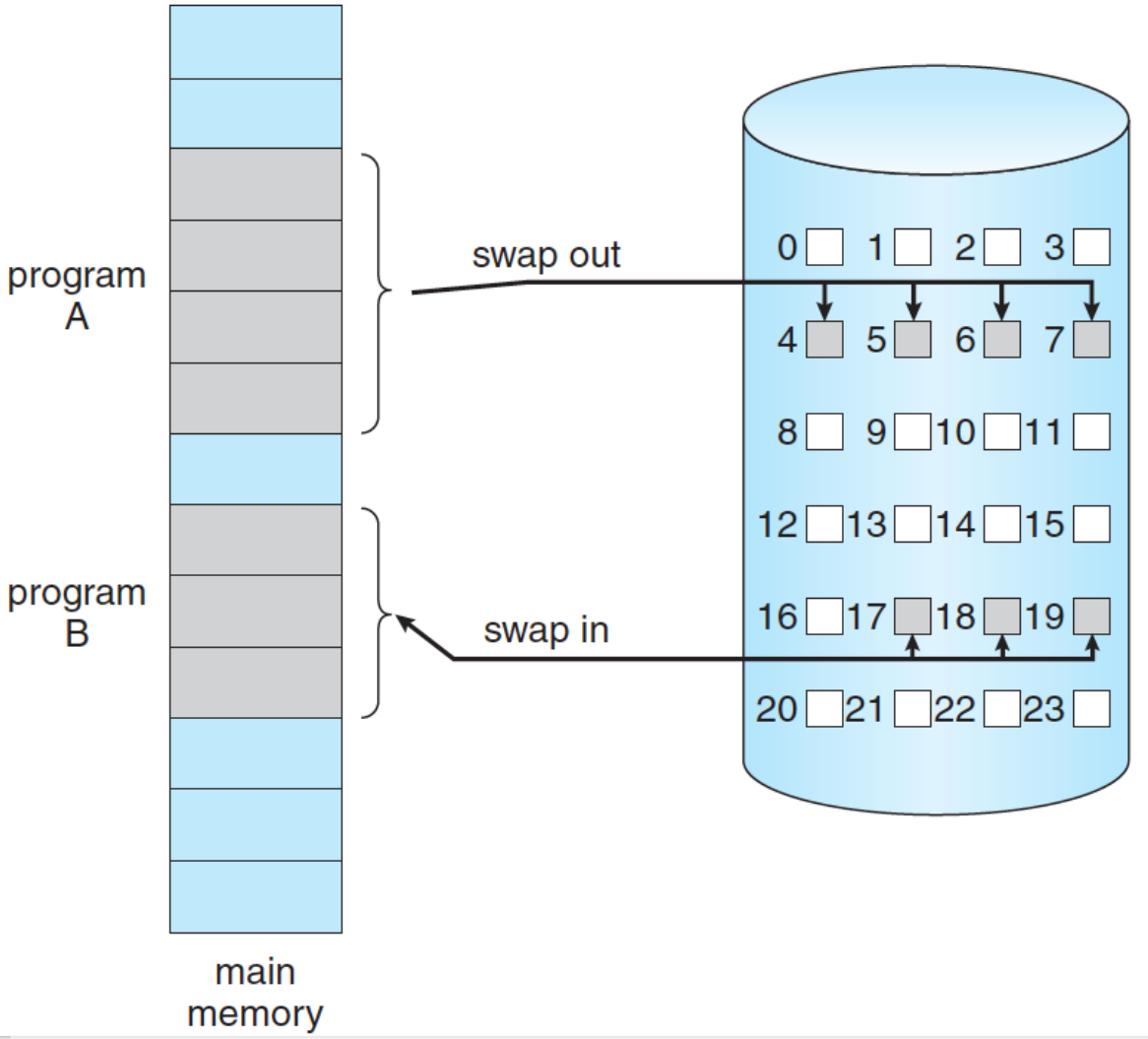
02.05. Віртуальна пам’ять (virtual memory)

7. Дві основні задачі:

1) Управління кількістю сторінок виділених процесу (скількі сторінок в даний момент часу буде мати процес X в оперативній пам’яті?).

2) Вибір стратегії (алгоритма) заміни сторінок окремого процеса (свопінгу сторінок процеса між оперативною та зовнішньою пам’ятю).

02.06. Віртуальна пам’ять (virtual memory)



03.1. Принцип локальності звертань до пам’яті

Визначення №1: Згідно принципу локальності (principle of locality) в сучасних КС з архітектурою фон Неймана в процесі виконання деякої програми доступ до одних даних (окремих або пов’язаних між собою) в пам’яті відбувається більш часто ніж до інших.

Визначення №2: Принцип локальності полягає в тому, що набори даних, до яких відбувається звертання в порівняно короткий проміжок часу, розміщуються в пам’яті достатньо добре прогнозованими кластерами.

03.2. Принцип локальності звертань до пам’яті

2. Принцип локальності є різновидом передбачуваної «поведінки» (predictable behavior), яка виникає в КС.

3. Якщо система демонструє сильну локальність звертань (locality of reference), то перспективним є застосування відповідних механізмів оптимізації роботи КС (стратегії заміни даних в кеші, попередня вибірка інструкцій (instruction prefetch), передбачення розгалуджень (branch prediction), конвеєризація та ін.).

03.3. Принцип локальності звертань до пам’яті

4. «Джерела» локальності звертань:
1) послідовна організація програми (відсутність умовних та безумовних перехоів);
2) впорядковані структури даних (масиви, списки і т.п.).

5. Два основних завдання:
1) підвищення локальності звертань при створенні програми або на етапі її компіляції;
2) дослідження наявної локальності звертань в ході виконання програми та використання результатів дослідження для вирішення задач оптимізації.

04.1. Основні види локальності звертань до пам’яті

1. Локальність у часі (temporal locality),

2. Локальність у просторі (spatial locality),

3. Еквідистантна локальність (equidistant locality),

4. Локальність розгалужень (branch locality).

04.2. Основні види локальності звертань до пам’яті

1. Локальність у часі (temporal locality):

- Якщо в даний момент часу відбулось звертання до даної ділянки пам’яті (memory location), то з великою ймовірністю в найближчий час відбудеться наступне звертання до неї.
- Тобто звертання до одної і тої ж ділянки пам’яті, як правило, близькі у часі.
- При цьому можуть розглядатись різні масштаби близькості у часі (temporal proximity).

04.3. Основні види локальності звертань до пам’яті

2. Локальність у просторі (spatial locality):

- Якщо в даний момент часу відбулось звертання до даної ділянки пам’яті, то з великою ймовірністю в найближчий час відбудеться звертання до сусідніх ділянок пам’яті.

- Тобто ділянки пам’яті, до яких відбувається звертання на короткому проміжку часу, близькі у просторі пам’яті.

04.4. Основні види локальності звертань до пам’яті

- Додатково може розглядатись задача прогнозування масштабу сусідства у просторі (spatial proximity), тобто розміру області пам’яті, яка містить сусідні ділянки.

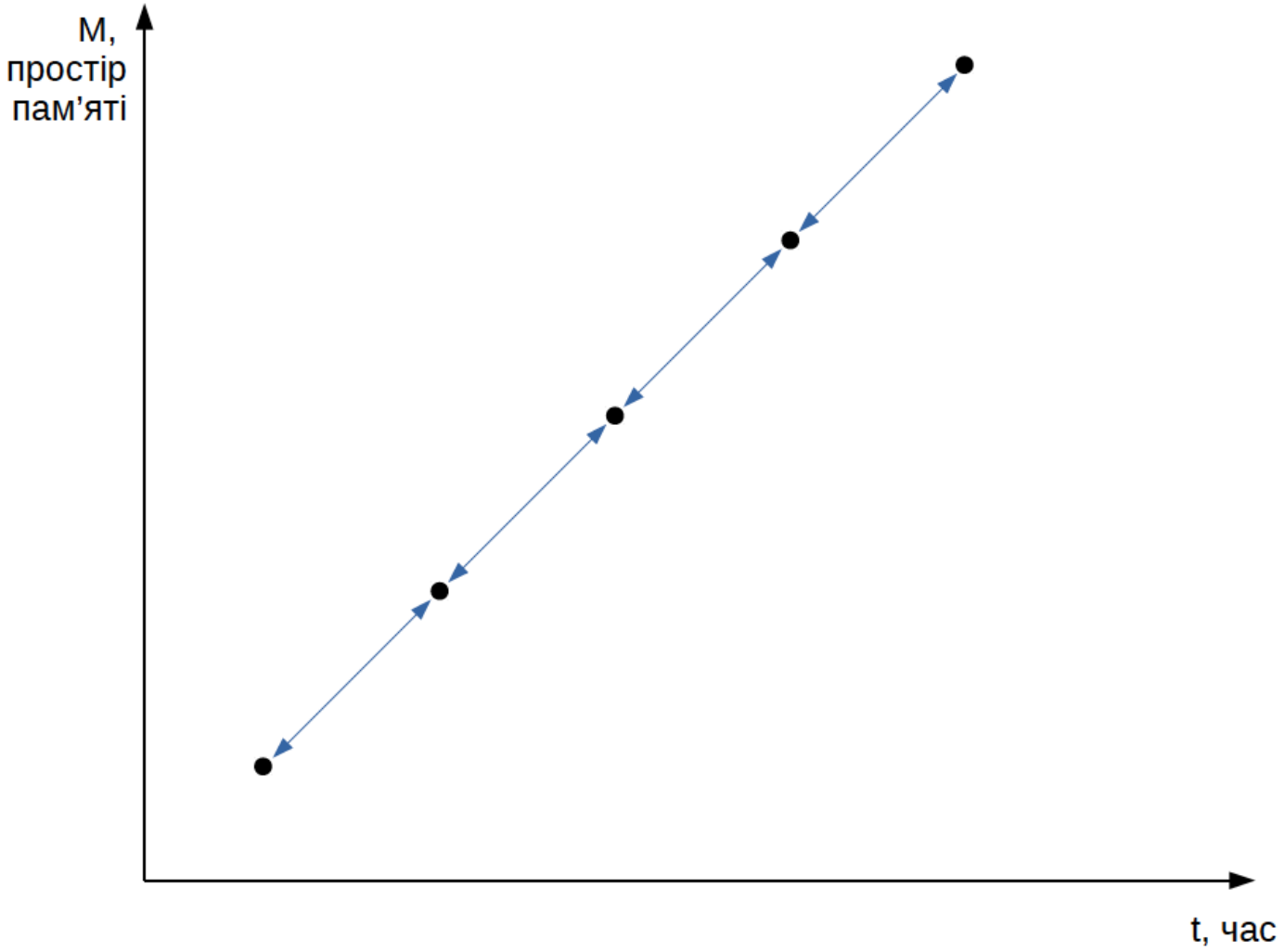
- Спеціальним випадком локальності у просторі є послідовна локальність (sequential locality), коли ділянки розміщені у пам’яті послідовно одна за одною (приклад: одновимірний масив даних).

04.5. Основні види локальності звертань до пам’яті

3. Еквідистантна локальність (equidistant locality):

- Ділянки пам’яті, до яких відбувається звертання, утворюють регулярну (і відтак передбачувану) структуру у просторово-часовому координатному просторі (spatial-temporal coordinate space).
- Наприклад, на основі аналізу декількох ітерацій циклу ініціалізації масиву, можна за допомогою лінійної функції передбачити (екстраполювати) до якої ділянки пам’яті відбудеться звертнання в найближчий час.

04.6. Основні види локальності звертань до пам’яті

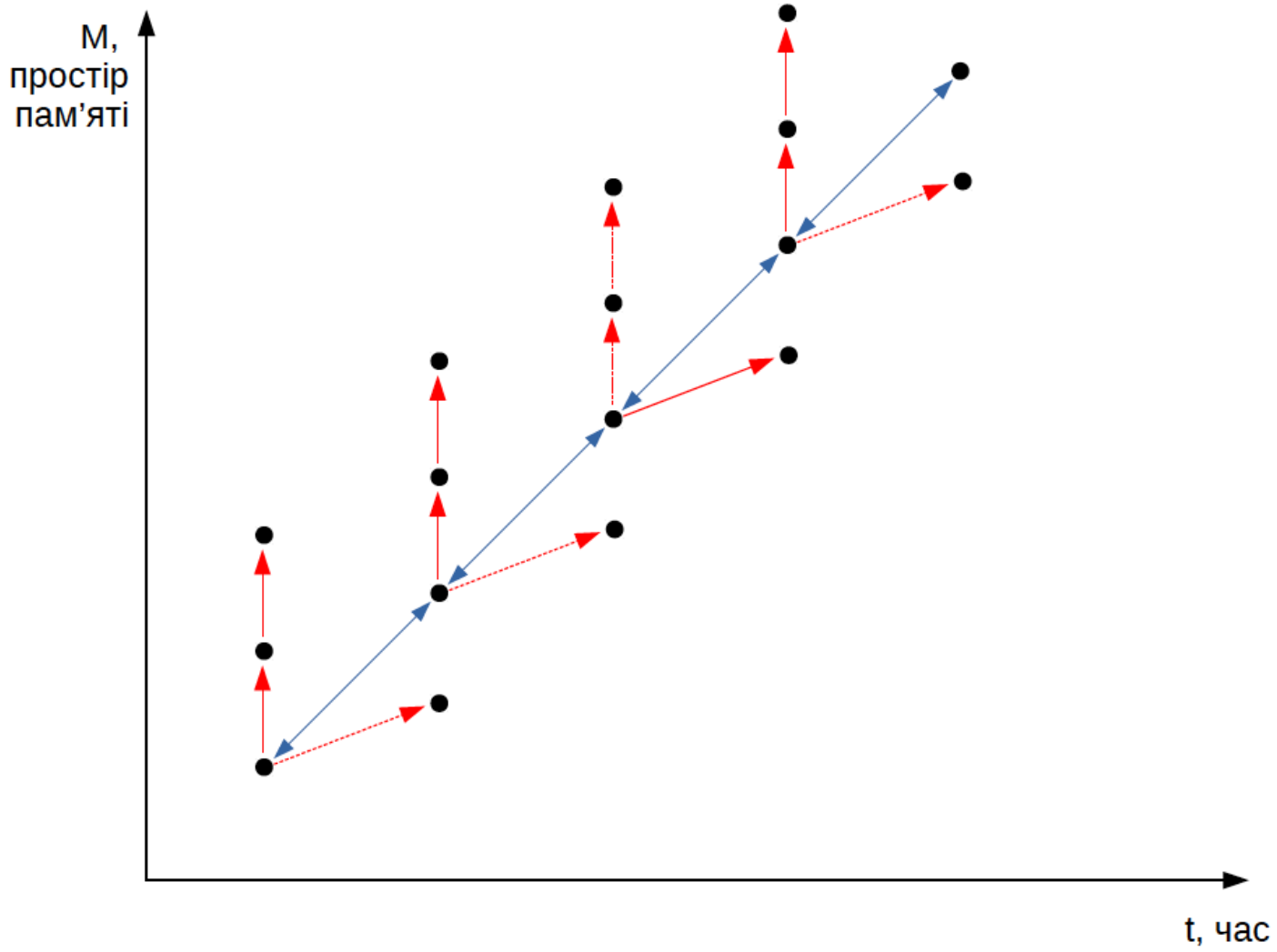


04.7. Основні види локальності звертань до пам’яті

4. Локальність розгалужень (branch locality):

- Альтернативні шляхи виконання програми формують регулярні розгалужені структури ділянок пам’яті у просторово-часовому координатному просторі.
- У випадку локальності розгалужень, як правило, береться до уваги порівняно мала множина альтернатив (наприклад, одна або дві умови в тілі циклу).
- На апаратному рівні принцип локальності розгалужень покладено у основу роботи модуля передбачення переходів (branch predictor).

04.8. Основні види локальності звертань до пам’яті



04.9. Основні види локальності звертань до пам’яті