Операционные системы

Лекция №5. Управление памятью в ОС.

Управление памятью

- Эффективное распределение памяти для размещения нескольких процессов
- Эффективное распределение памяти для обеспечения эффективного исполнения готовых процессов.

Требования к управлению памятью

- Перемещение
- Защита
- Совместное использование
- Логическая организация
- Физическая организация

Перемещение

- Программист не знает куда будет загружена программа при выполнении
- Во время выполнения программа может быть выгружена на диск и в дальнейшем загружена в память по другим адресам (перемещена)
- Ссылки на память должны быть переведены процессором и системой в реальные физические адреса.

Перемещение (Relocate)

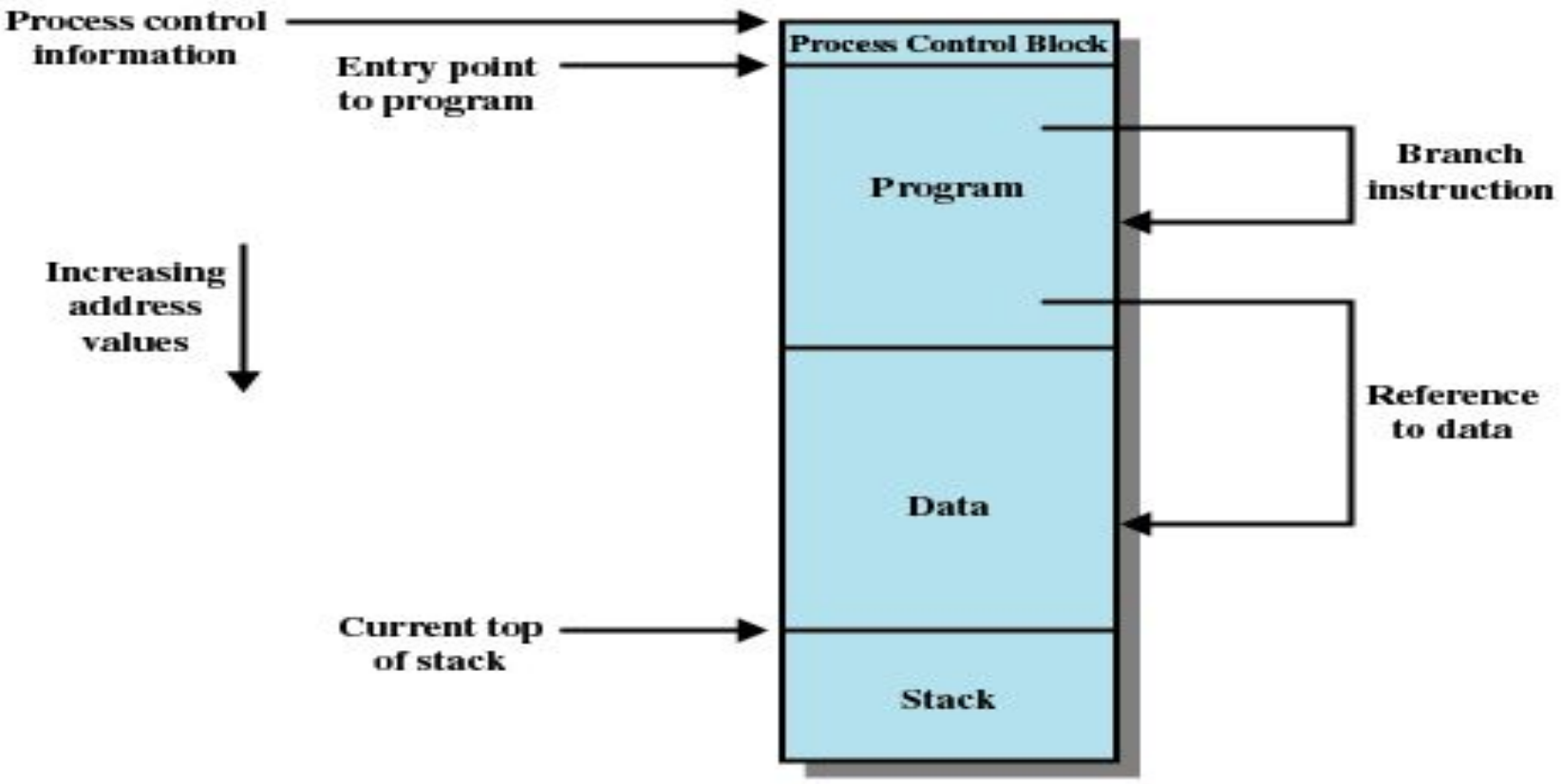


Figure 7.1 Addressing Requirements for a Process

Защита

- Процессы не должны без разрешения иметь возможности к обращениям к памяти других процессов
- Невозможно вычислить и проверить абсолютные адреса памяти при компиляции
- Защита памяти обеспечивается аппаратным обеспечением системы.
  - Операционная система не в состоянии предвидеть все обращения к памяти.
  - Использование механизма исключений (exception) для отслеживания нарушений доступа к памяти (0xEh в системах Intel)

Разделение

- Возможность нескольким процессам разделять общие адреса памяти
  - Эффективнее разделять общие блоки памяти несколькими процессами, чем создавать закрытые копии.

Логическая организация

- Программы имеют модульную структуру
- Модули могут создаваться и компилироваться независимо
- Можно установить различную степень защиты для модулей (только для чтения, только для выполнения и т.д.)
- Можно разделять модули между процессами

Физическая организация

- Двухуровневая архитектура памяти
  - Основная
  - Вторичная
- Основной задачей является эффективная организация потоков информации между основной и вторичной памятью

Распределение памяти

- Фиксированное распределение
- Динамическое распределение
- Система двойников
- Простая страничная организация
- Простая сегментная организация
- Страничная организация ВП
- Сегментная организация ВП

Фиксированное распределение

- Любой процесс, чей размер меньше или равен размеру раздела может быть загружен в свободный раздел
- Если все разделы заняты, то система может выгрузить простаивающий процесс во вторичную память
- Программы, которые не помещаются в раздел могут быть созданы с использованием оверлеев (overlays).

Фиксированное распределение

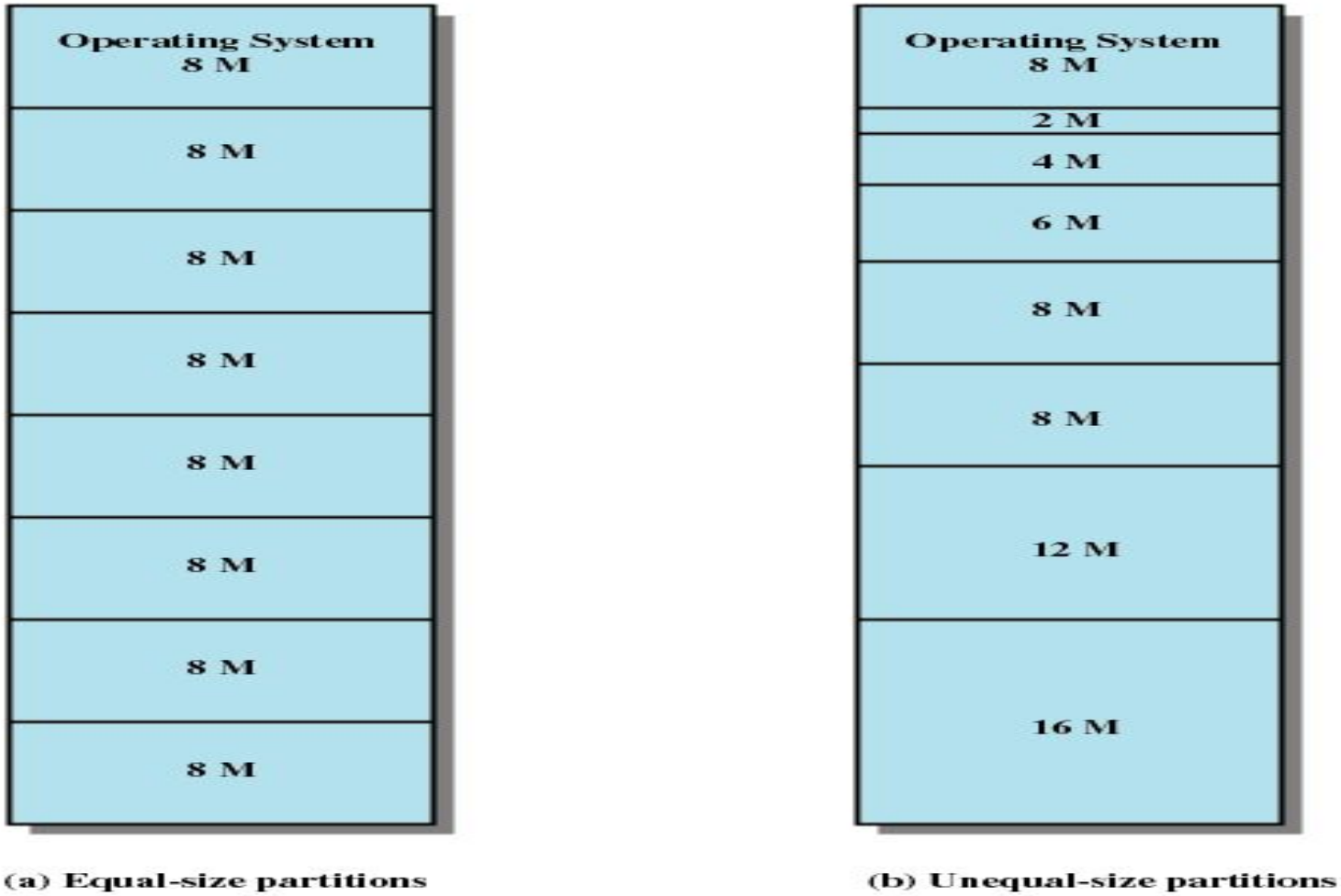


Figure 7.2 Example of Fixed Partitioning of a 64-Mbyte Memory

Алгоритм размещения при фиксированном распределении

- Разделы одинакового размера
  - Не имеет значения какой свободный раздел выбрать для использования.
- Разделы разного размера
  - Использовать наименьший раздел, способный вместить процесс
  - Очередь процессов для каждого раздела

Алгоритм размещения при фиксированном распределении

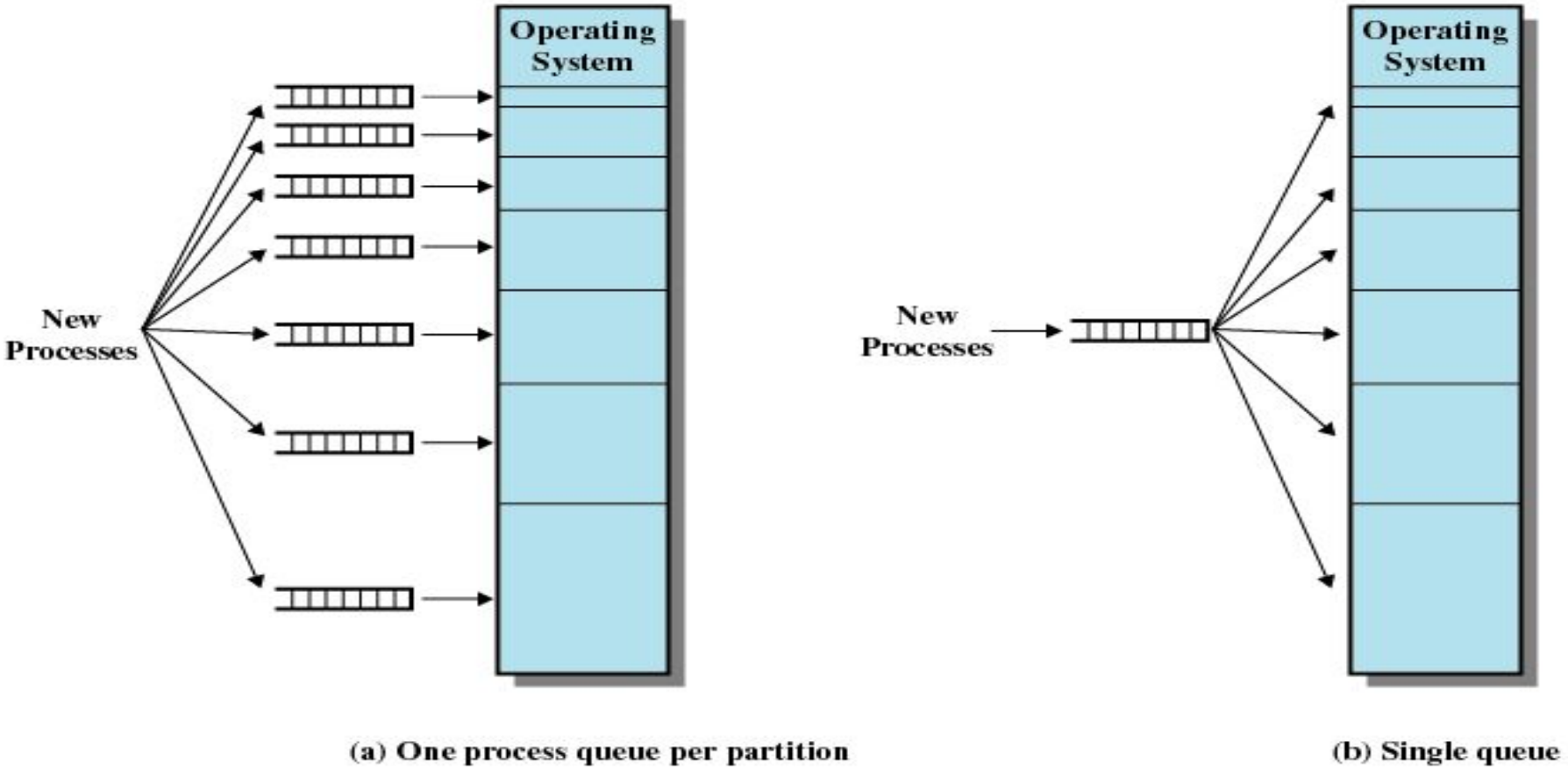


Figure 7.3 Memory Assignment for Fixed Partitioning

Динамическое распределение

- Разделы имеют переменную длину
- Процесс распределяет памяти ровно столько, сколько требуется
- Возможно появления «дыр» в памяти – внешняя фрагментация памяти.
- Для преодоления фрагментации используется метод уплотнения для перемещения процессов в смежные области памяти; свободная память собирается в один блок.

Динамическое распределение

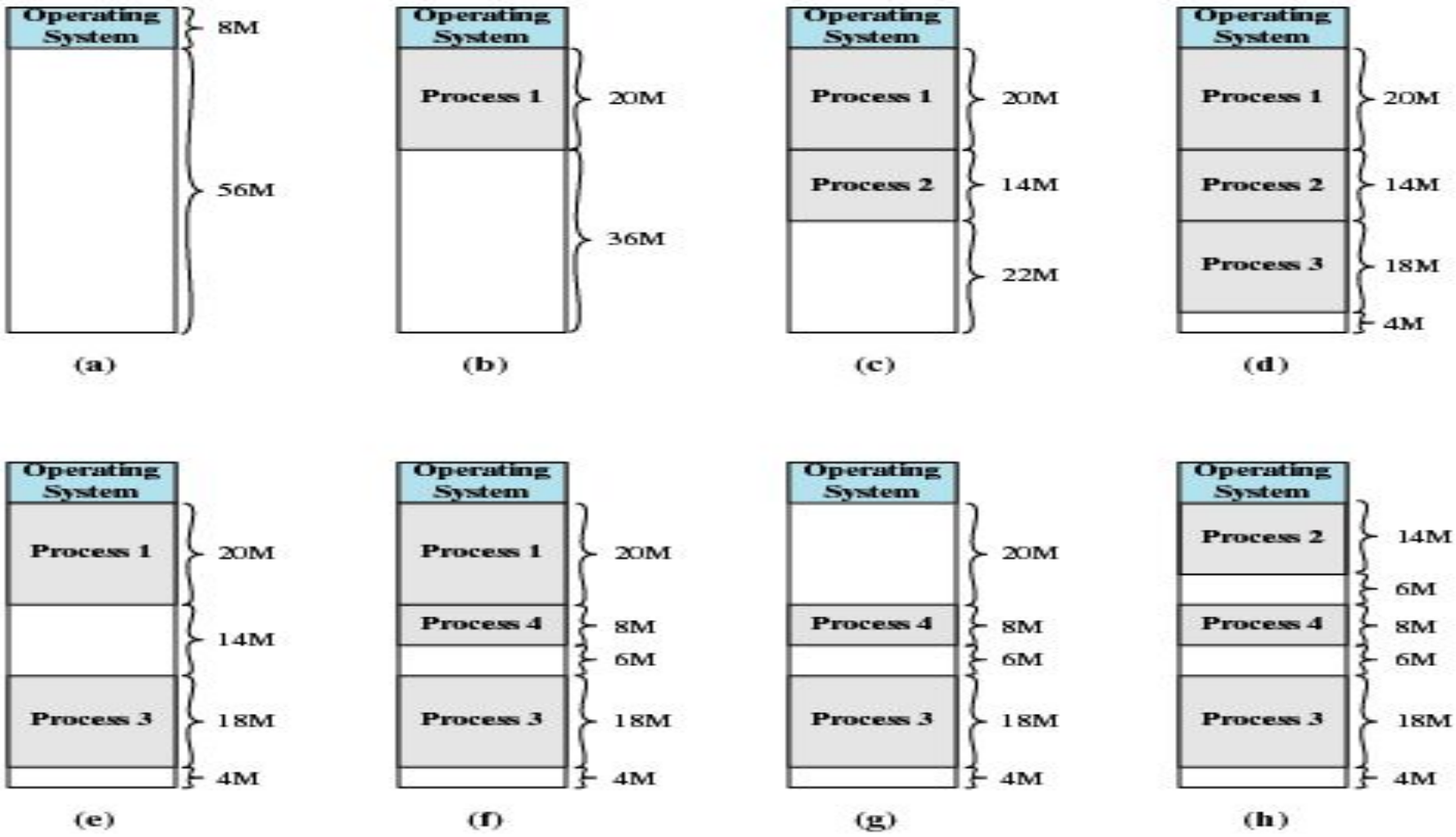


Figure 7.4 The Effect of Dynamic Partitioning

Динамическое распределение. Алгоритм размещения.

- Первый подходящий блок памяти
- Наилучший подходящий

Динамическое распределение. Алгоритм размещения.

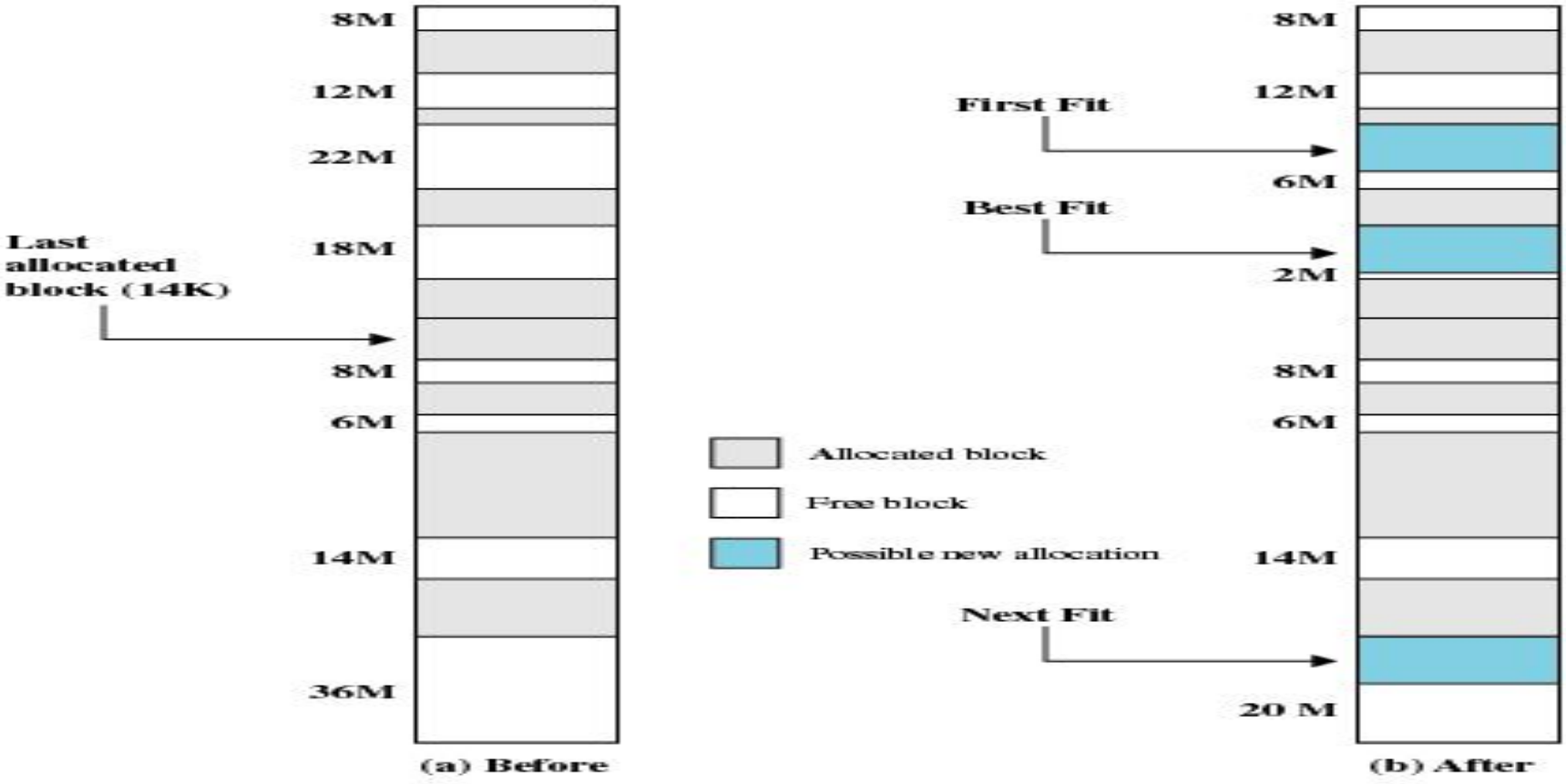


Figure 7.5 Example Memory Configuration Before and After Allocation of 16 Mbyte Block

Динамическое распределение. Система двойников.

- Всё доступное пространство считается единым блоком размером 2^U
- При запросе размером *s*, таким, что $2^{U-1} < s <= 2^U$, выделяется весь блок
- В противном случае блок разделяется на два эквивалентных двойника размерами 2^{U-1}
- Если $2^{U-2} < s <= 2^{U-1}$, то по запросу выделяется один из двойников.

Динамическое распределение. Система двойников.

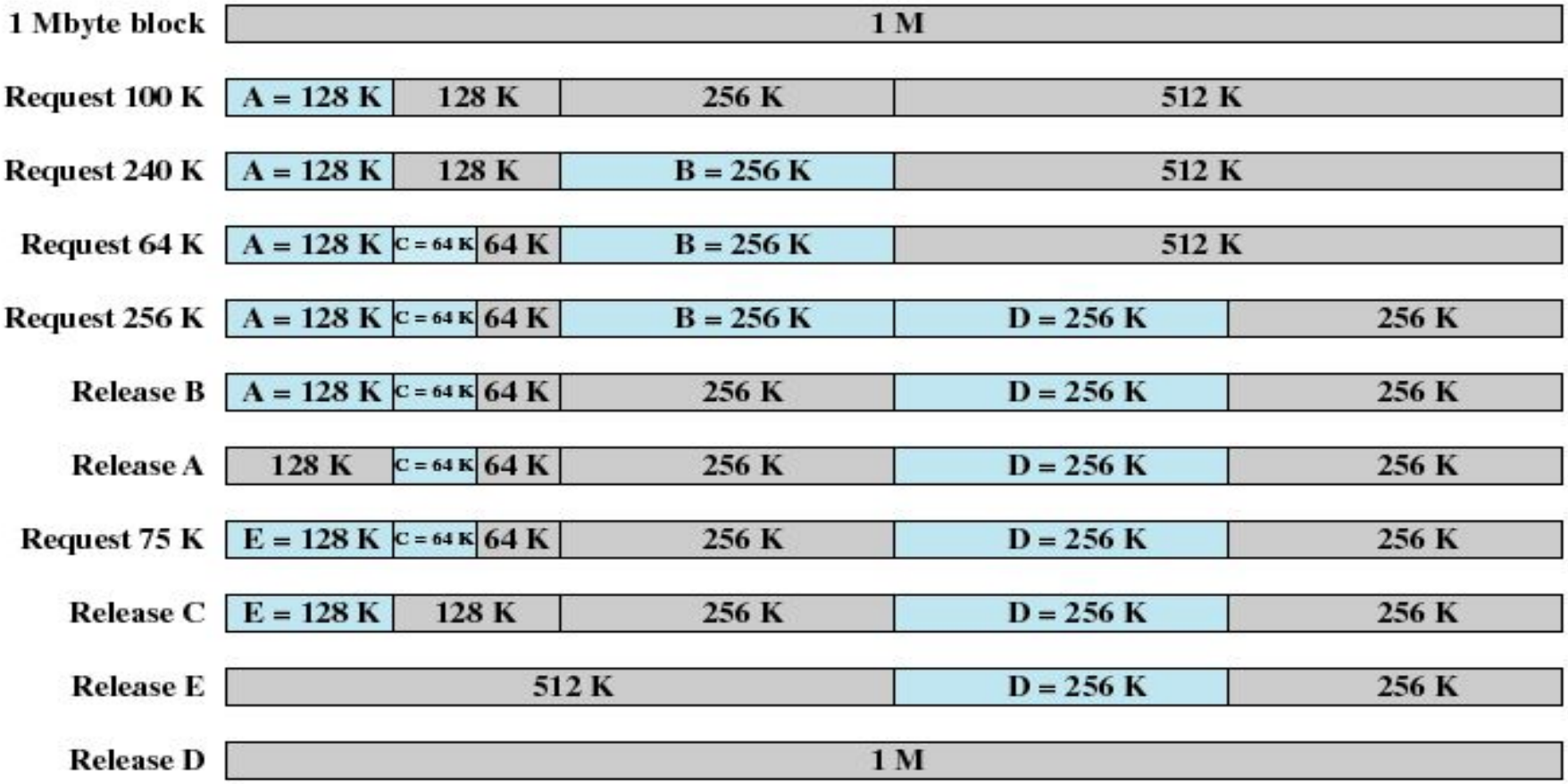


Figure 7.6 Example of Buddy System

Динамическое распределение. Система двойников.

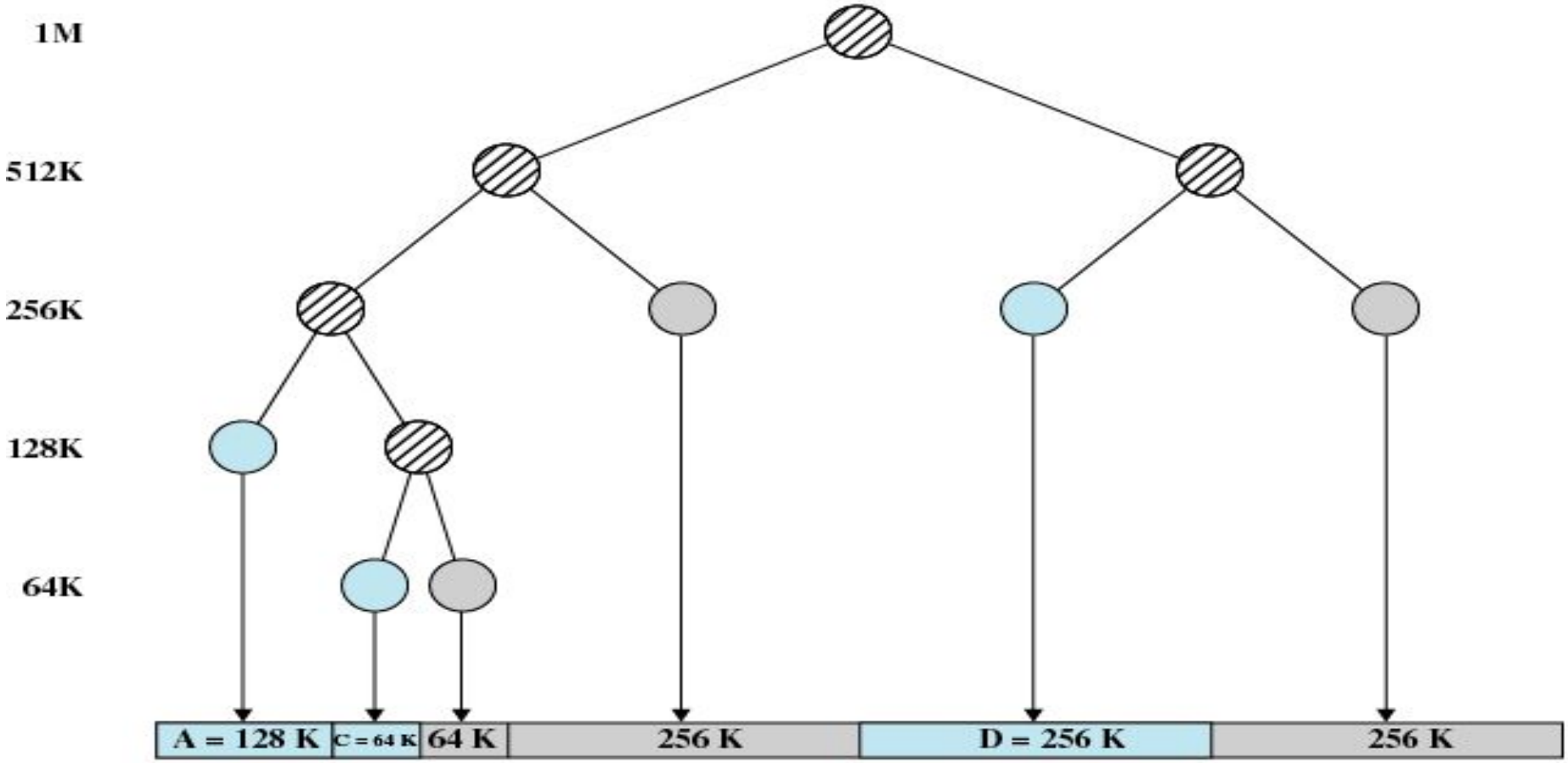


Figure 7.7 Tree Representation of Buddy System

Перемещение. Типы адресов.

- **Логический адрес (Logical)**. Ссылка на ячейку памяти, не зависящая от текущего расположения данных в памяти
- **Относительный адрес (Relative)**. Частный случай логического адреса, когда адрес рассчитывается относительно известной точки в программе.
- **Физический адрес (абсолютный, physical)**. Действительное расположение ячейки в основной памяти.

Перемещение.

- Абсолютные (физические) адреса памяти известны только при загрузке программы в память.
- Процесс может перемещаться между различными разделами в результате операций свопинга
- Уплотнение также вызывает перемещение процесса в основной памяти.

Аппаратная поддержка перемещения

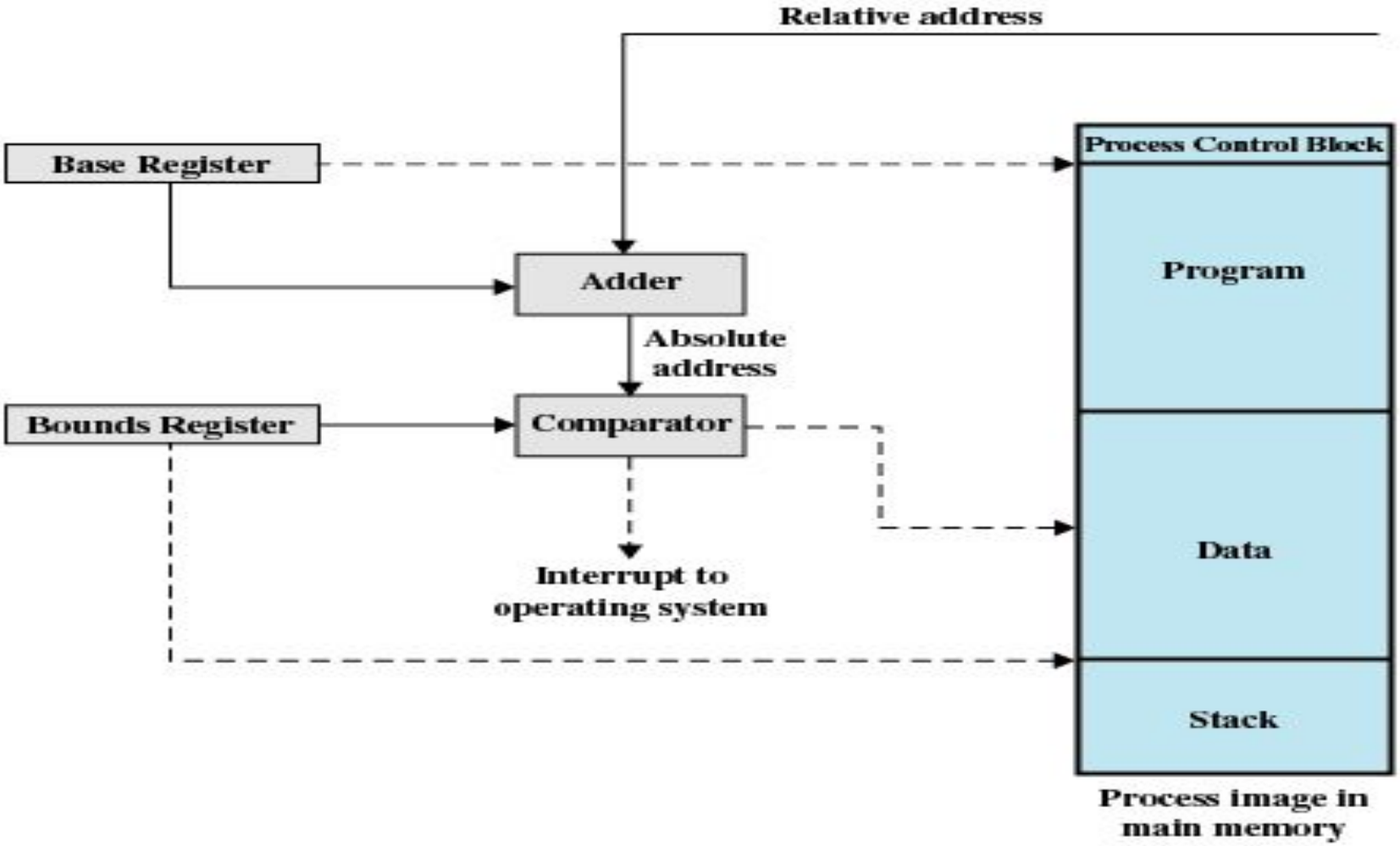
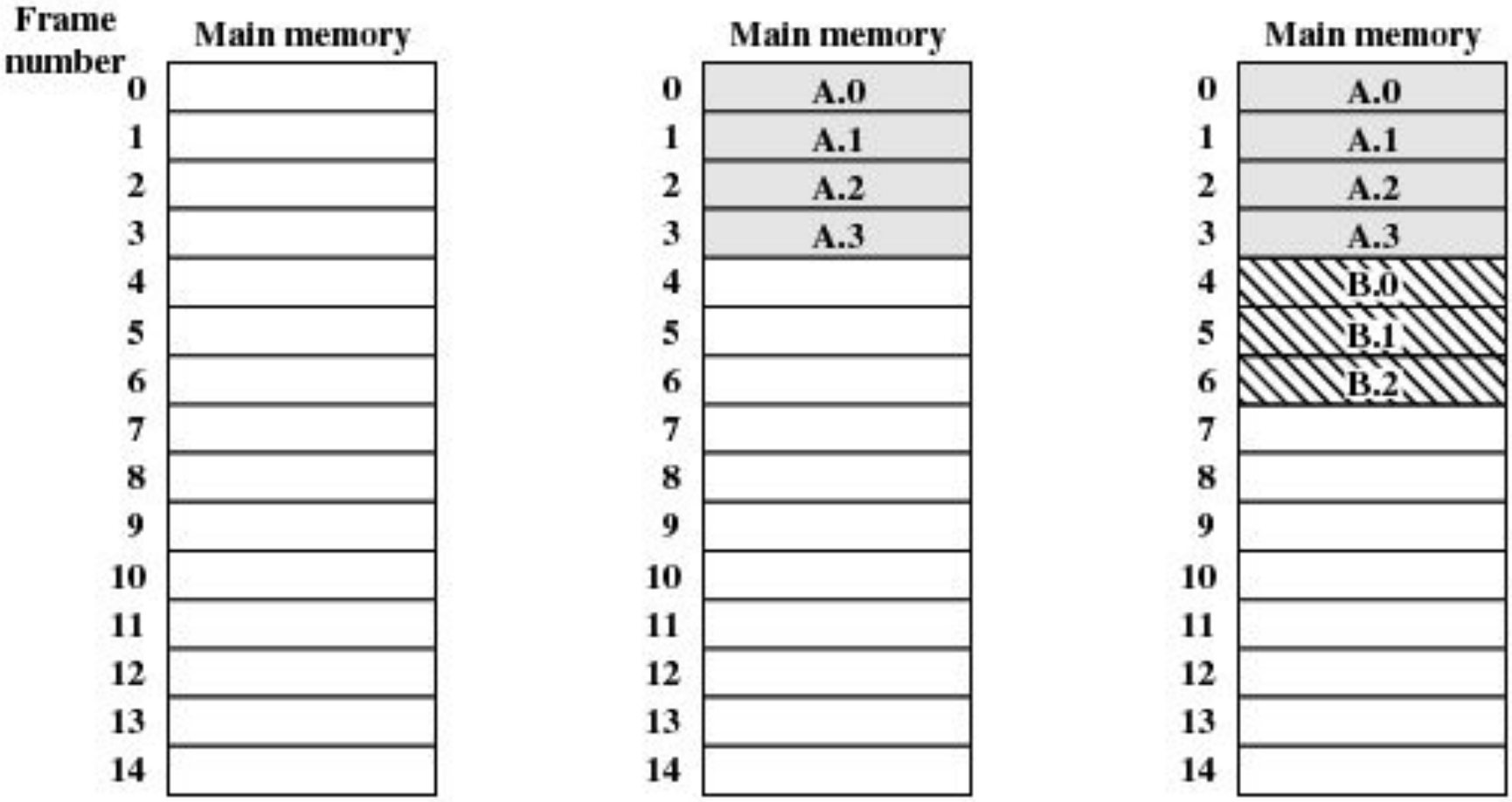


Figure 7.8 Hardware Support for Relocation

Страничная организация

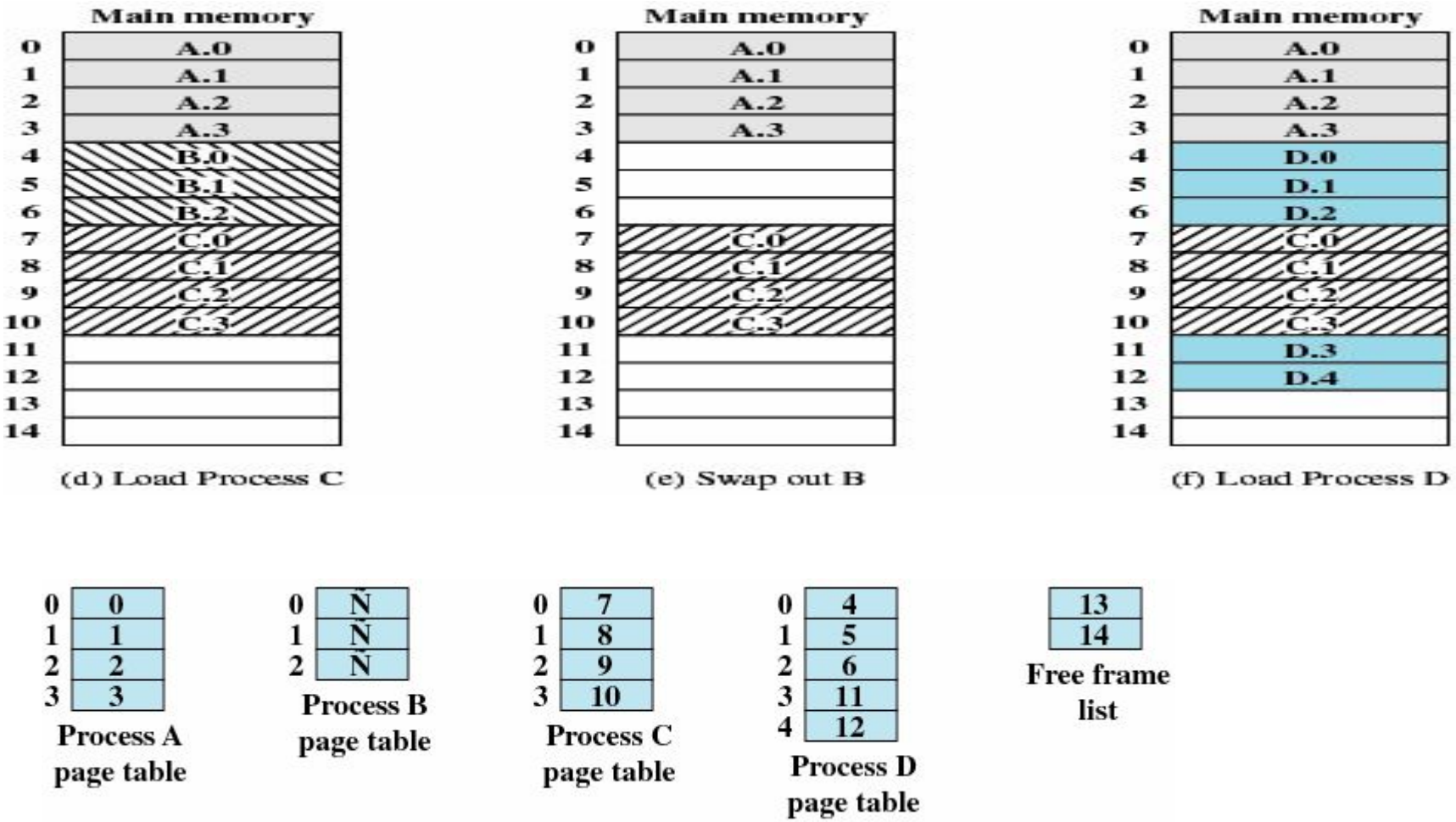
- Разделение физической памяти на блоки равной длины и разделение процесса на блоки этой же длины
- Блоки процесса называются страницы (pages). Блоки основной памяти – фреймы (frames).
- Операционная система поддерживает таблицу страниц для каждого процесса
  - Содержит расположение кадра для каждой страницы процесса
  - Адрес памяти представлен адресом страницы и смещением внутри страницы.

Страничная организация. Распределение памяти.



(a) Fifteen Available Frames (b) Load Process A (c) Load Process B

Страничная организация. Распределение памяти.



Сегментация

- Сегменты программы могут иметь разный размер
- Существует максимальный размер сегмента
- Адрес состоит из двух частей: номера сегмента и смещения внутри сегмента
- Так как сегменты имеют переменный размер, сегментация похожа на динамическое распределение.

Получение логического адреса в различных организациях

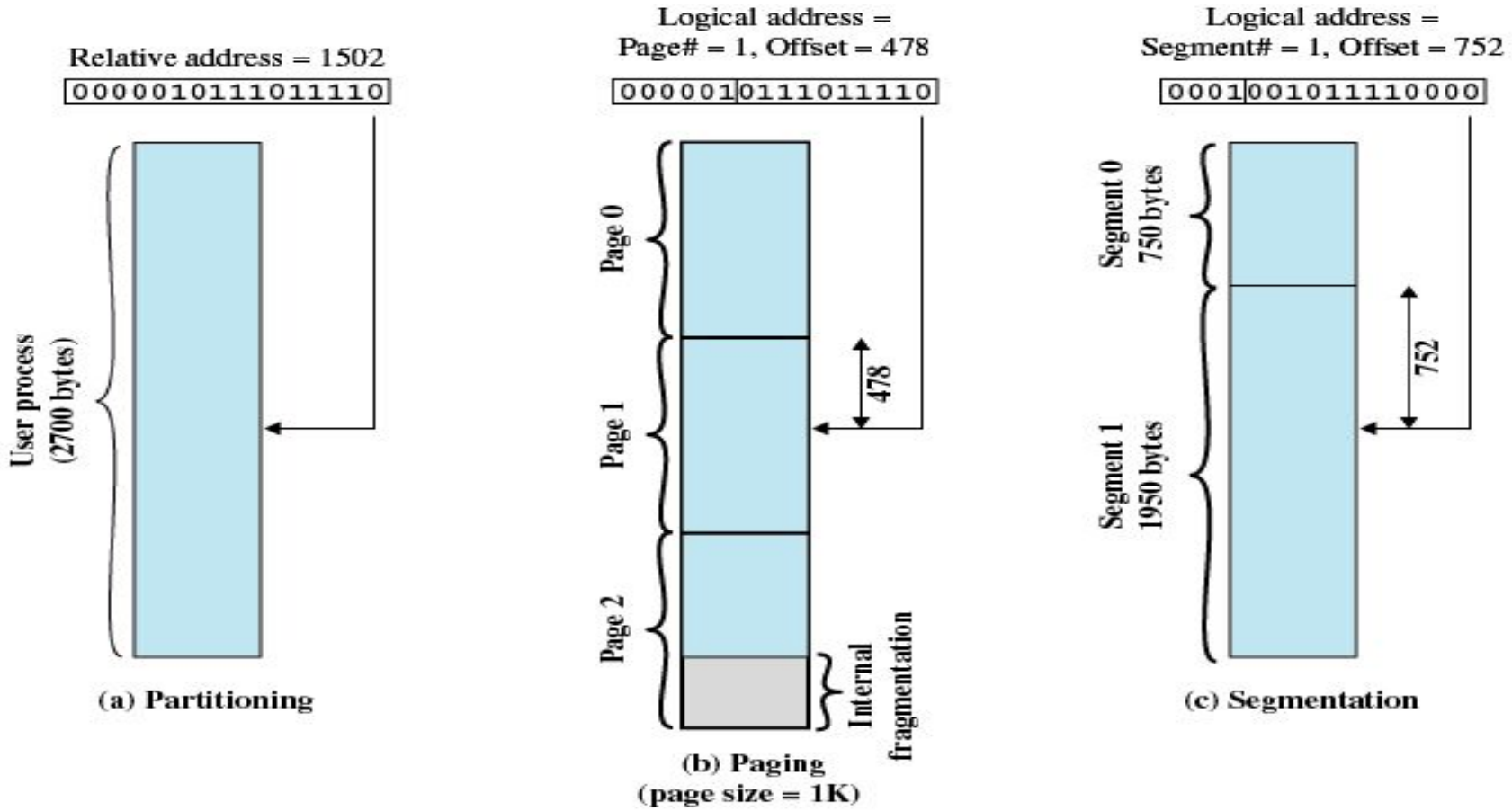
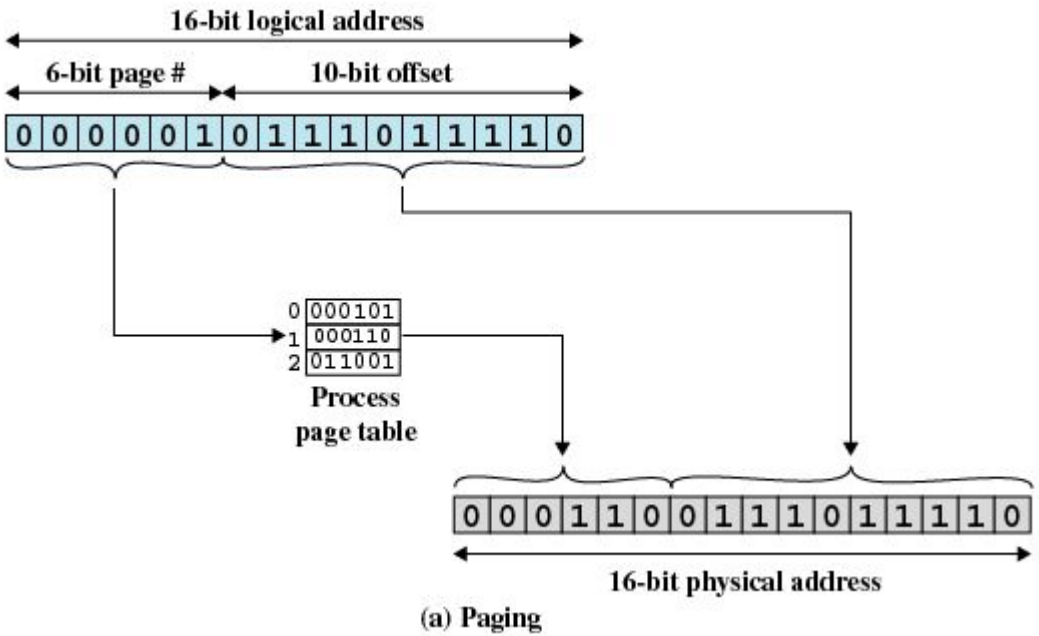


Figure 7.11 Logical Addresses

Трансляция логических адресов. Страничная организация.



(a) Paging

Трансляция логических адресов. Сегментная организация.

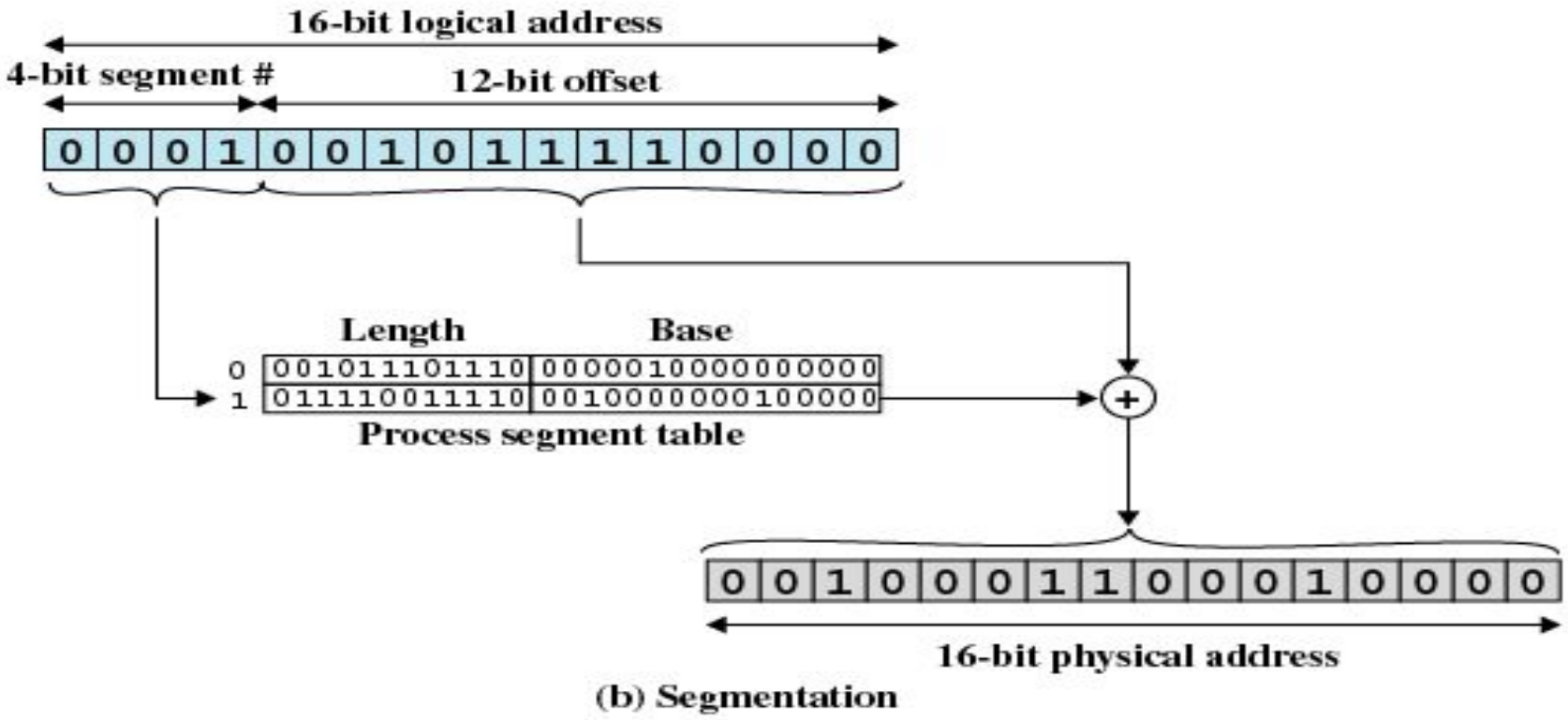


Figure 7.12 Examples of Logical-to-Physical Address Translation

Список литературы

1. Столлингс, Вильям. **Операционные системы**, 4-е издание. «Вильямс», 2002.