Операционные системы

Лекция №6. Виртуальная память.

Основные идеи

- Обращения логическим адресам памяти динамически транслируются в физические адреса во время исполнения
  - Процесс может быть выгружен на диск и вновь загружен в основную память таким образом он может находится в разных местах основной памяти
- Процесс может быть разбит на несколько блоков (страниц/сегментов), которые не обязательно занимают смежные адреса в основной памяти
- Нет необходимости загружать в основную память сразу все блоки

Выполнение программы

1. Операционная система помещает в основную память несколько блоков (страниц/сегментов) программы
   - Множество блоков, загруженных в основную память называется *резидентным множеством*
2. При обращении процесса к логическому адресу, отсутствующего в оперативной памяти блока, генерируется прерывание.
3. Операционная система переводит процесс в состояние блокировки.

Выполнение программы

4. Блок процесса, которому принадлежит логический адрес помещается (*погружается*) в основную память
   1. Операционная система инициирует операцию ввода-вывода с диска
   2. К процессору подключается другой процесс (поток) из очереди готовых.
   3. После завершения операции ввода-вывода генерируется прерывание.
   4. Система переводит блокированные процесс в состояние готовности.

Преимущества технологии виртуальной памяти

- Больше процессов может одновременно находится в основной памяти
  - Загружены только некоторые блоки процессов
  - При большом количестве процессов в основной памяти высока вероятность существования готовых к выполнению процессов в любой момент времени.
- Процесс может быть больше, чем вся основная память.

Типы памяти

- Основная память
  - Оперативная (реальная) память, где может выполняться процесс.
- Виртуальная память
  - Дисковая память
  - Обеспечивает эффективную многозадачность, снимая ограничения на объём основной памяти.
- Виртуальное адресное пространство (ВАП).
  - Диапазон ячеек памяти, доступных процессу
  - Не все адреса ВАП должны быть *спроецированы* на основную память при исполнении процесса.

Пробуксовка (Thrashing)

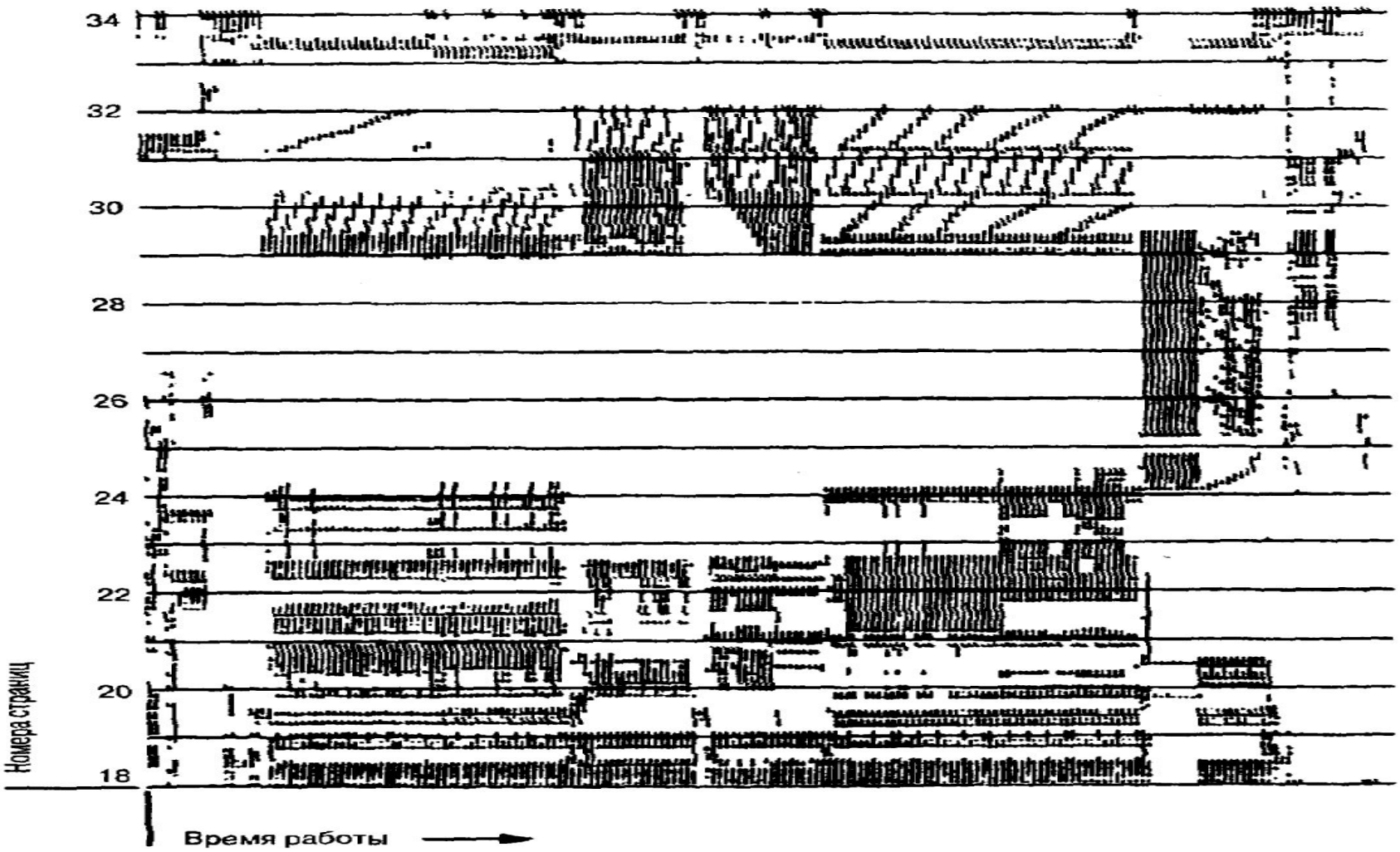
- Выгрузка из основной памяти активных блоков – блоки, которые требуются процессу для выполнения
- Система тратит больше времени на перемещение блоков между основной и вторичной памятью, чем на исполнение программ.

Принцип локализации

1. Обращения к коду и данным в операционной системе имеют тенденцию к кластеризации.
2. Только несколько блоков процесса требуются для исполнения в короткие промежутки времени.
3. Есть возможность делать реальные оценки необходимых в ближайшем будущем процессу блоков.

Принцип локализации даёт надежду на эффективность работы виртуальной памяти.

Кластеризация обращения к страницам процесса



Поддержка функционирования ВП

- Аппаратное обеспечение должно поддерживать сегментную, страничную или сегментно-страничную организацию
- Операционная система должна иметь возможность для переноса страниц и /или сегментов между основной и вторичной памятью.

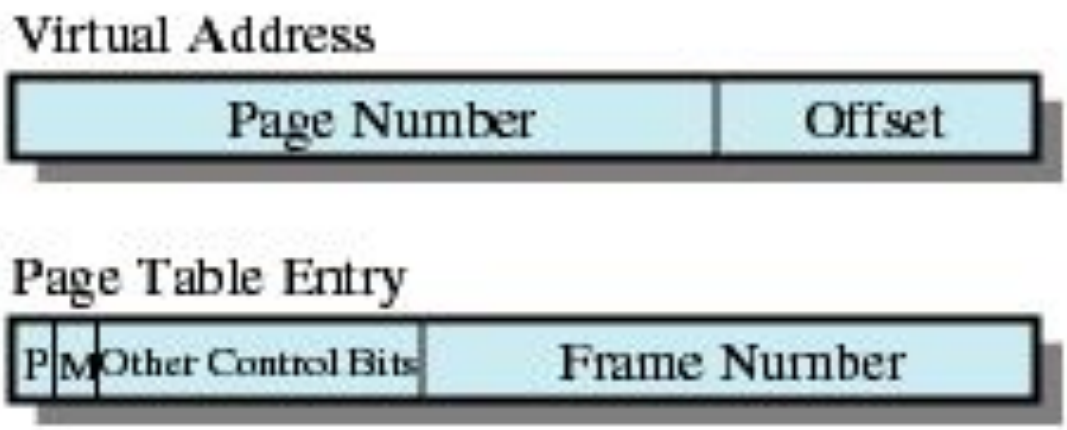
Страничная организация

- Каждый процесс имеет собственную таблицу страниц (page table).
- Каждая запись таблицы страниц (page table entry - PTE) содержит соответствующий странице номер кадра в основной памяти.
- Должен существовать бит присутствия, определяющий находится ли страница в основной памяти или нет.
- При обращении к отсутствующей в основной памяти странице генерируется прерывание *отказ страницы (Page Fault)*

Бит модификации в таблице страниц

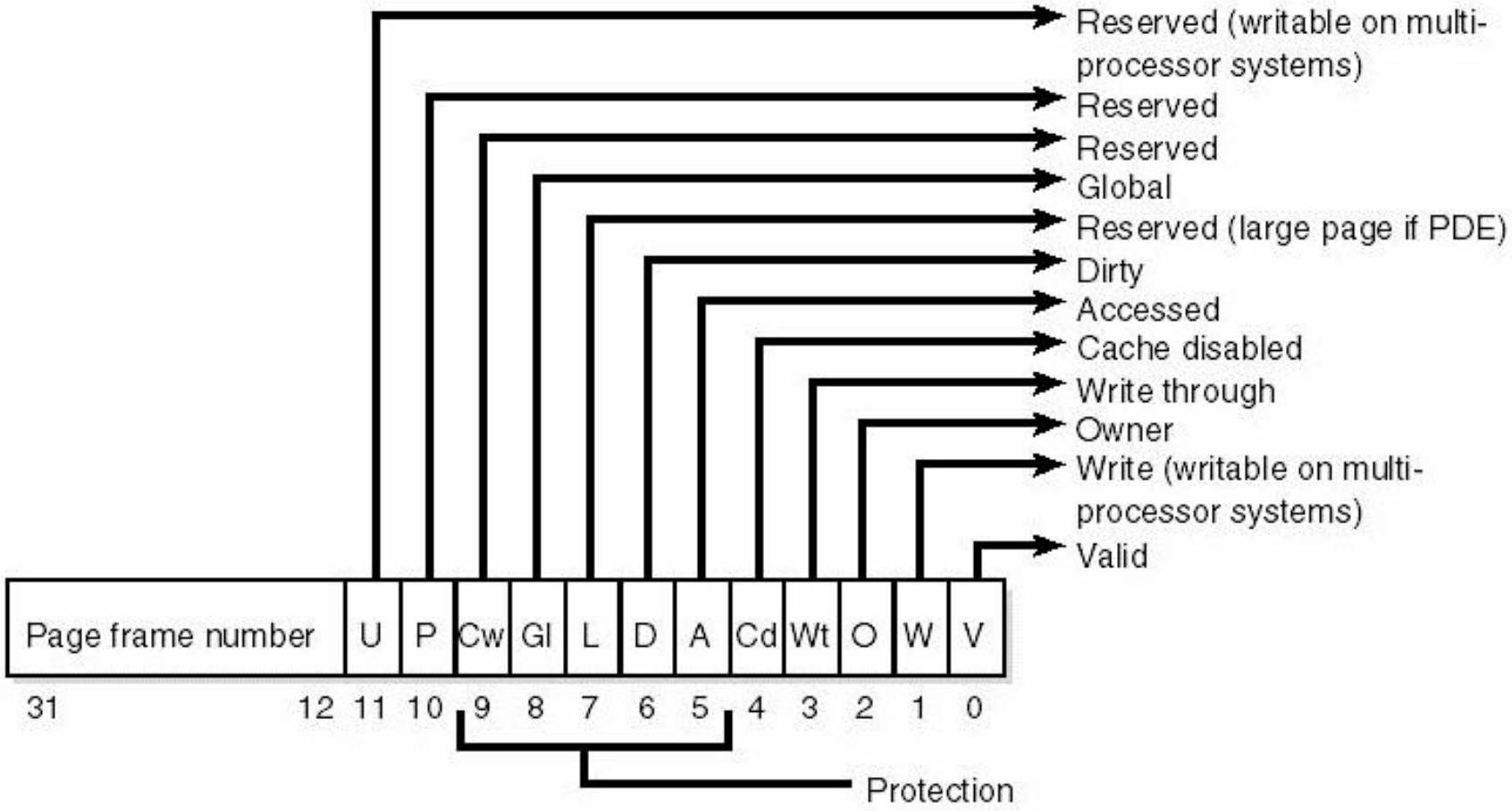
- Бит модификации, определяющий были ли модифицированы данные странице, с момента её загрузки в основную память.
- Если изменений не было, то страницу перед освобождением не требуется перемещать во вторичную память.

Страничная организация. PTE.



(a) Paging only

Запись таблицы страниц PTE в МП 80386



Адресация при страничной организации.

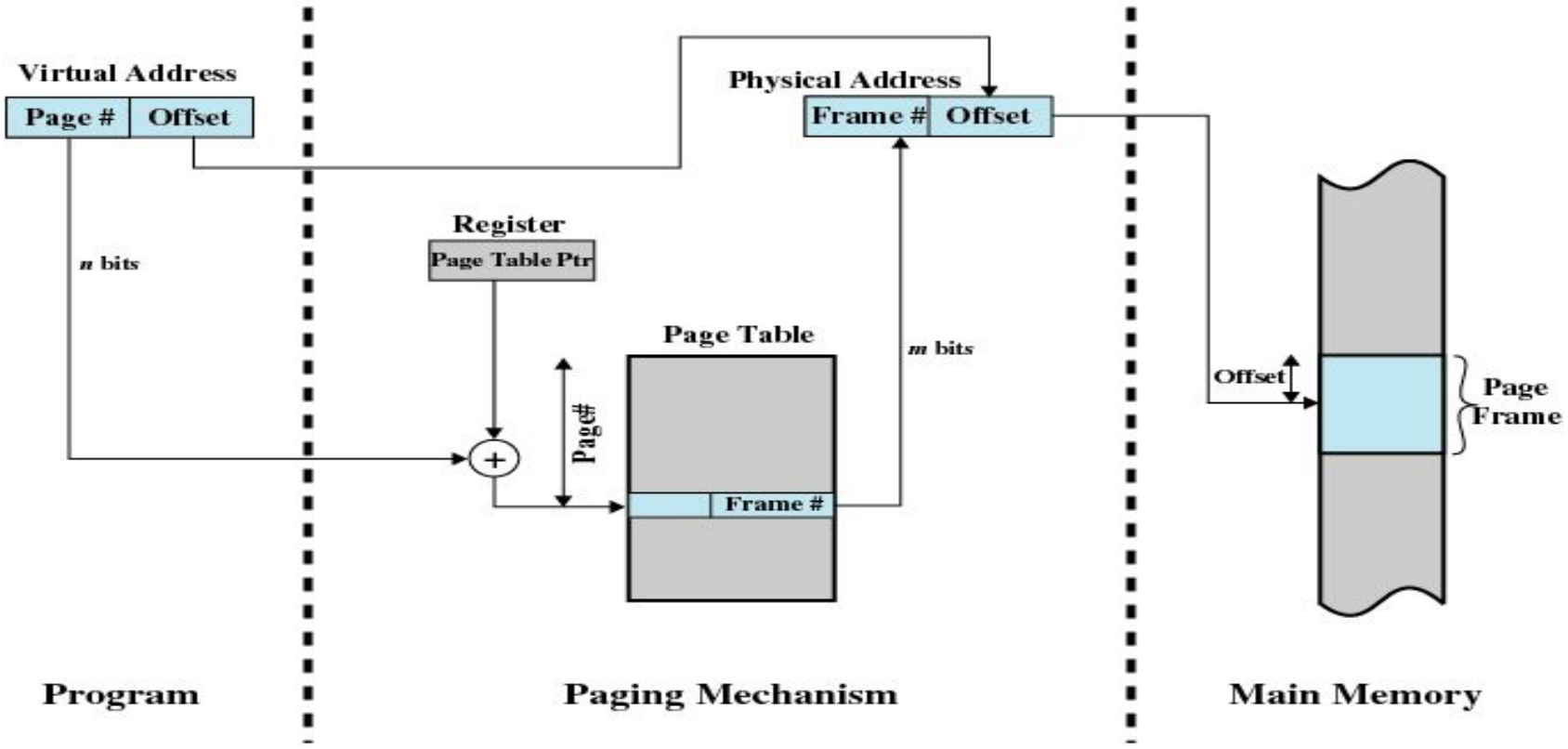


Figure 8.3 Address Translation in a Paging System

Двухуровневая адресация при страничной организации (32-разрядный адрес)

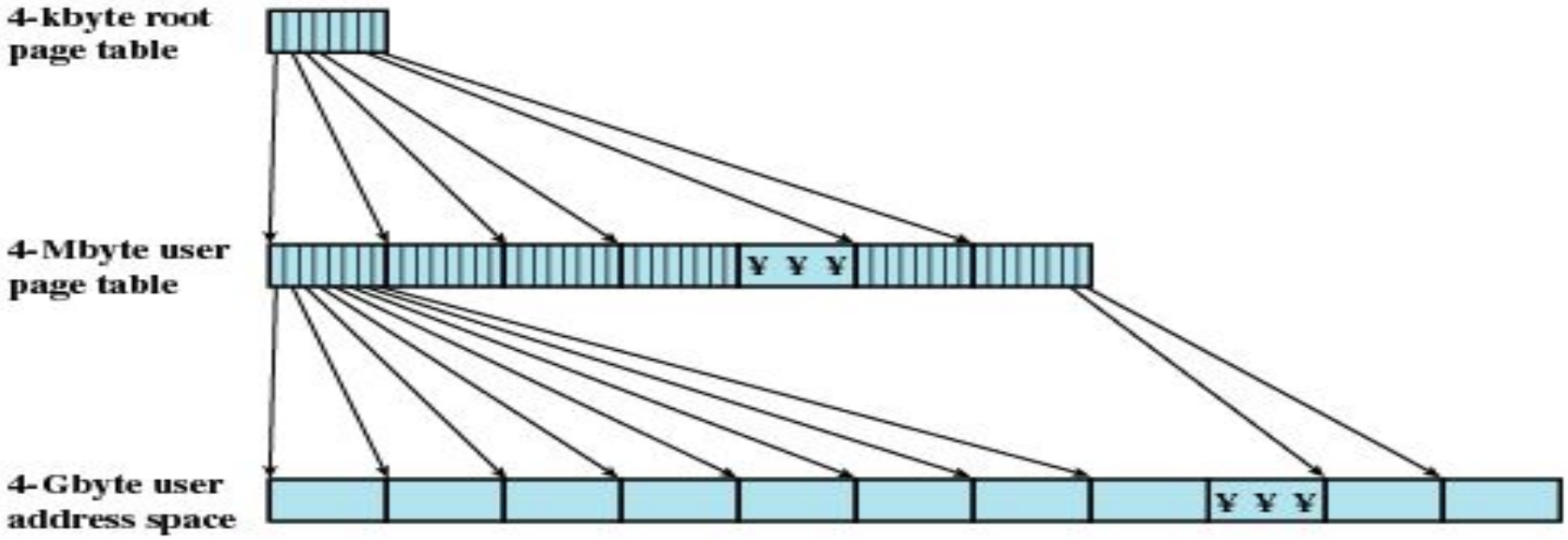
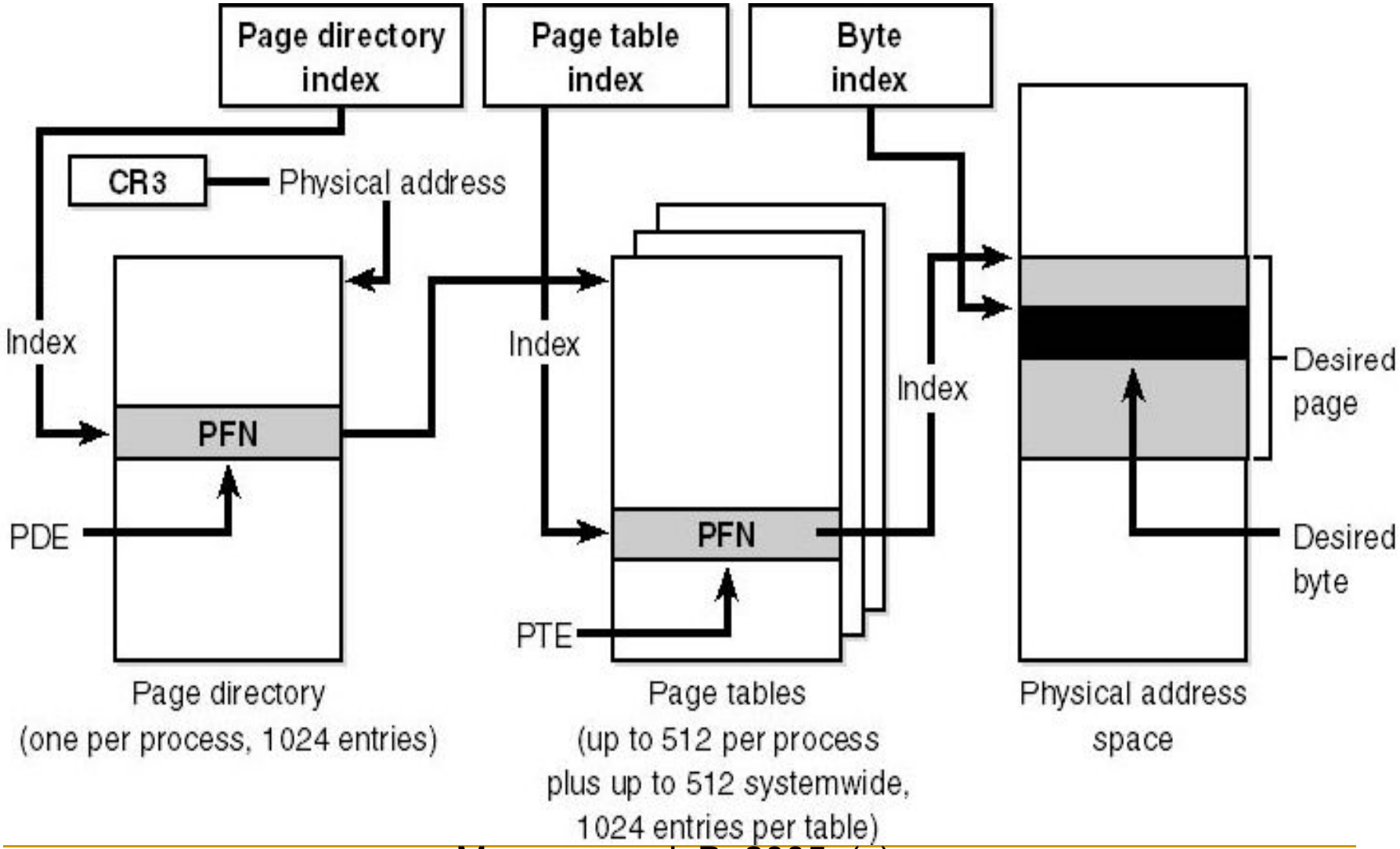


Figure 8.4 A Two-Level Hierarchical Page Table

Таблицы страниц при многоуровневой организации

- Таблицы страниц могут занимать слишком много места в оперативной памяти
- Таблицы страниц также располагаются в виртуальной памяти.
- Только часть таблицы страниц может располагаться в основной памяти при выполнении процесса.

Страничная адресация в 80386



Ассоциативный буфер трансляции (TLB – Translation Lookaside Buffer)

- Каждое обращение к виртуальному адресу может вызывать два обращения к физическому адресу
  - Для считывания таблицы страниц
  - Для чтения данных с кадра основной памяти
- Для решения этой проблемы существует высокоскоростной кэш для часто используемых PTE, который называется TLB.

Функционирование TLB

- При обращении к виртуальному адресу, процессор сначала обращается к TLB
- Если PTE присутствует в TLB (попадание в TLB), номер кадра извлекается из TLB и формируется адрес в основной памяти
- При промахе в TLB процессор использует номер страницы для обращения к PTE.

Функционирование TLB

- При обращении к виртуальному адресу проверяется наличие страницы в основной памяти
  - При отсутствии генерируется исключение
- Производится обновление TLB для включения нового PTE.

Функционирование TLB

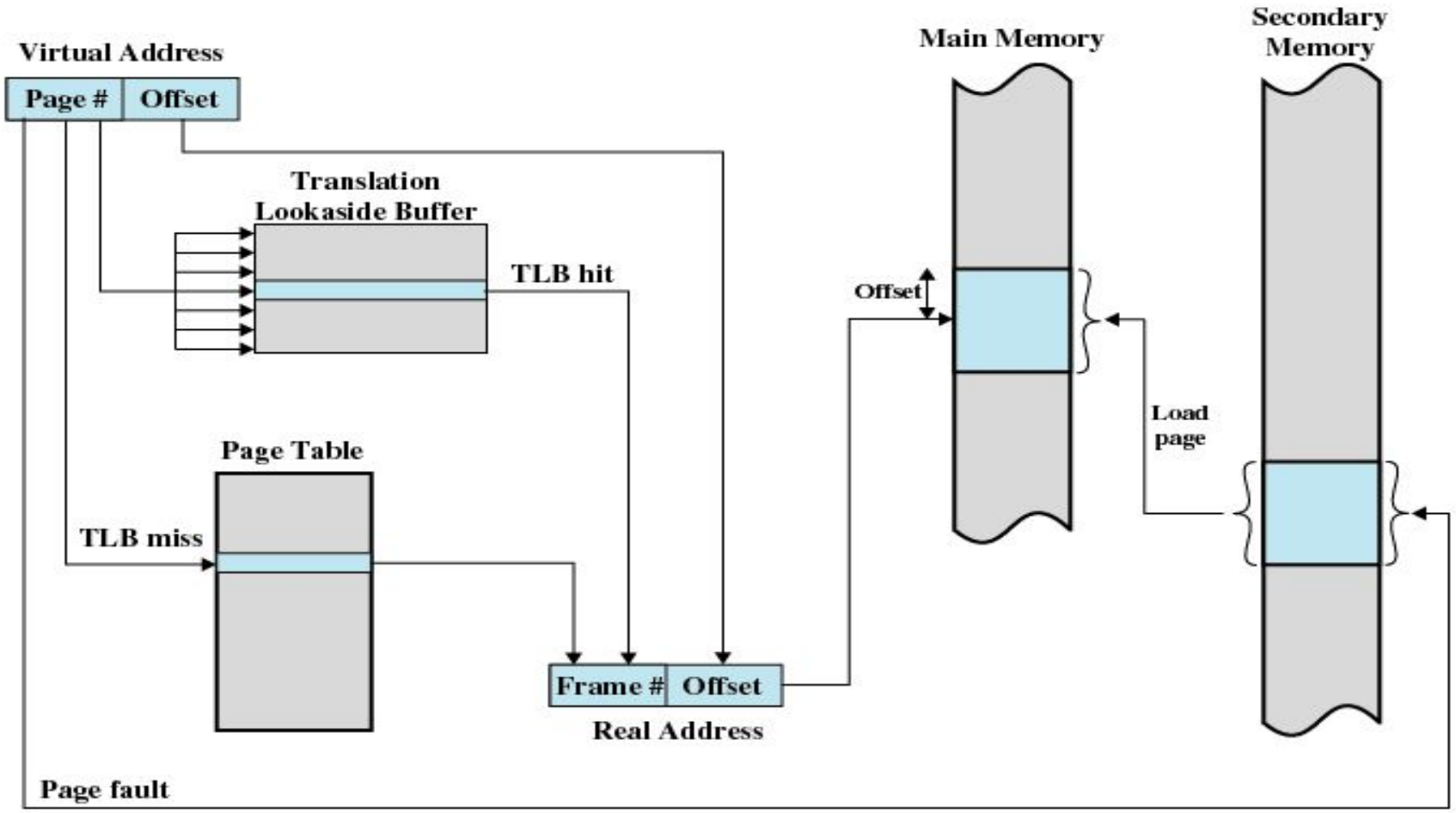
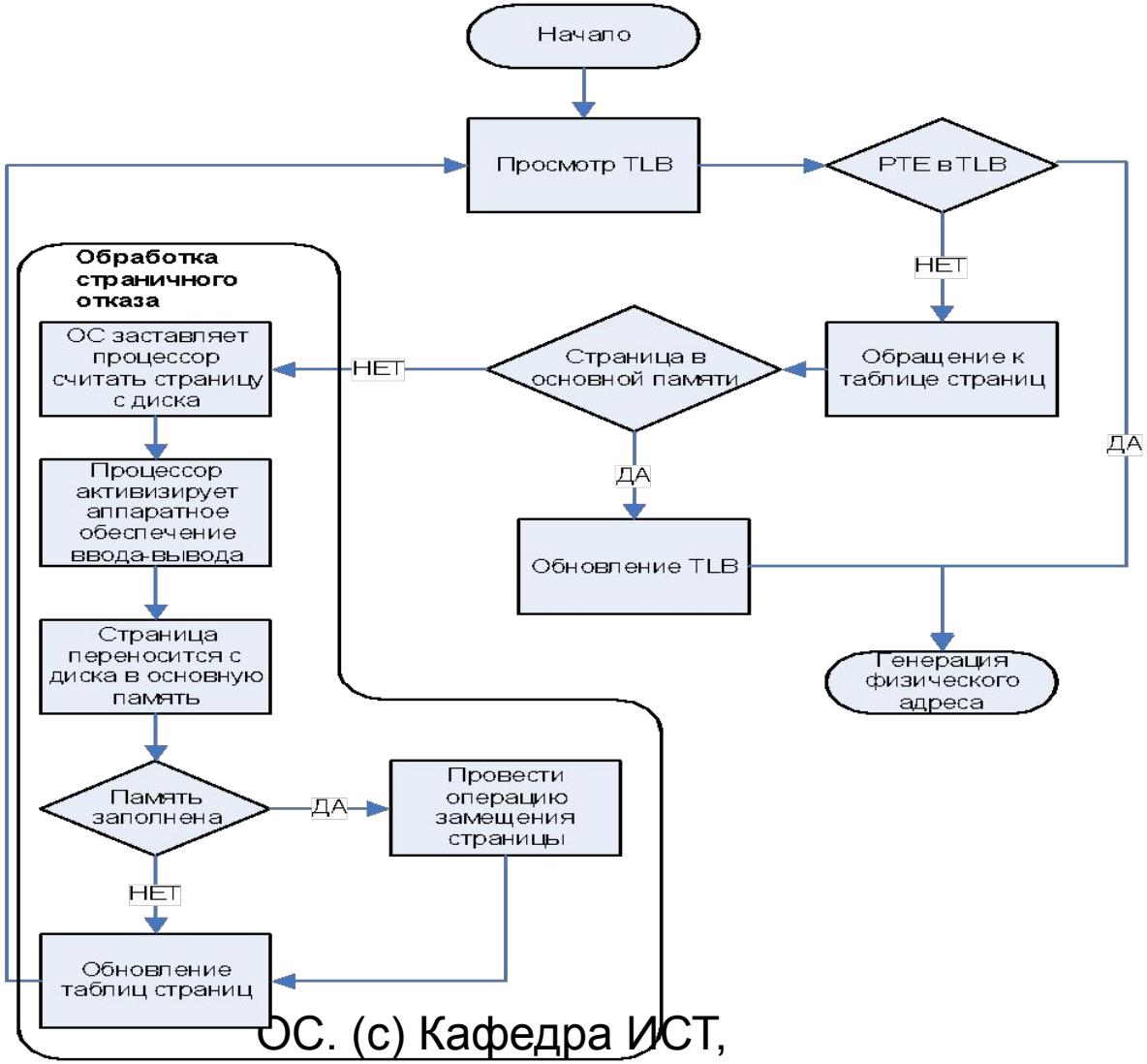


Figure 8.7 Use of a Translation Lookaside Buffer

Функционирование TLB



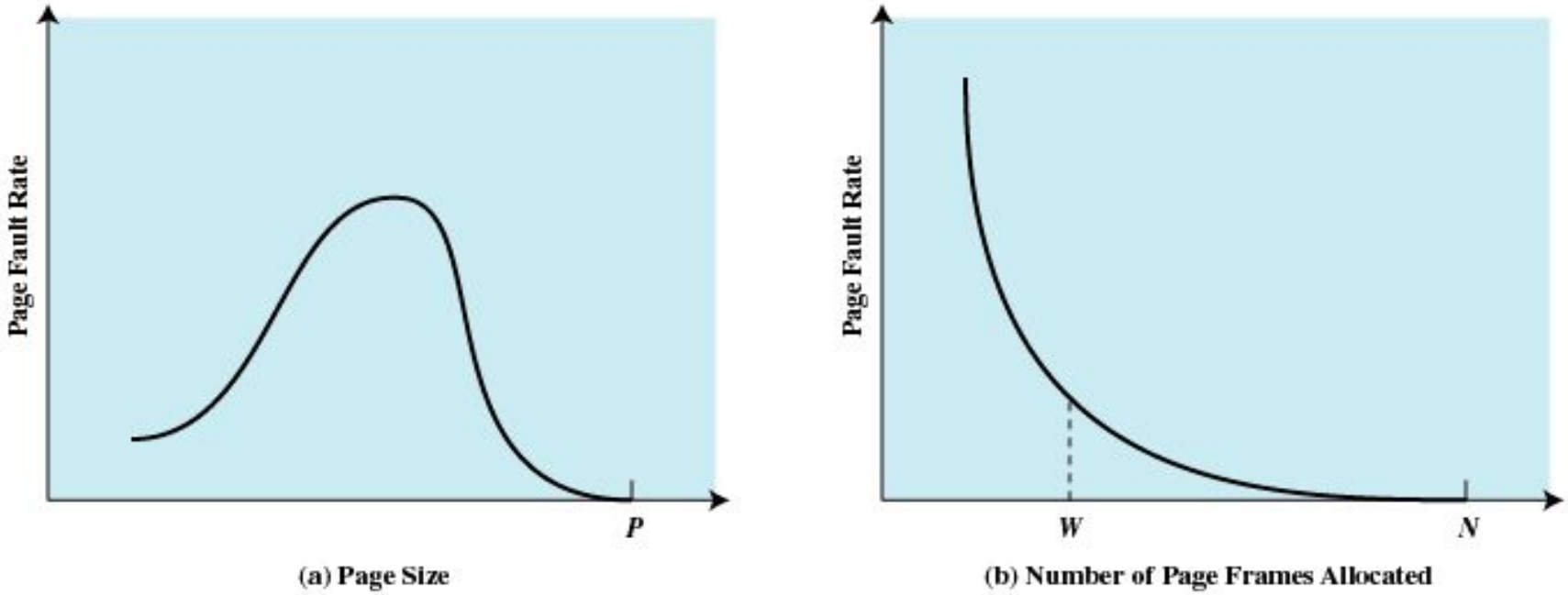
Размер страницы

- Меньше размер страницы – меньше внутренняя фрагментация
- Меньше размер страницы – больше страниц требуется процессу
- Больше страниц процесса – больше памяти отводится под таблицы страниц
- Вторичная память эффективнее работает с большими блоками данных – эффективней использовать большие страницы.

Размер страницы

- Малый размер страницы – большее количество страниц можно держать в оперативной памяти
- После некоторого времени исполнения процесса, страницы памяти будут содержать все необходимые процессу данные (принцип локализации). Частота страничных отказов будет низкой.
- Увеличение размера страницы приводит к тому, что страницы содержат данные, которые располагаются вдалеке от последних обращений к памяти. Частота страничных отказов возрастает.

Размер страницы



P = size of entire process
W = working set size
N = total number of pages in process

Figure 8.11 Typical Paging Behavior of a Program

Размеры страниц различных архитектур МП.

Table 8.2 Example Page Sizes

Computer	Page Size
Atlas	512 48-bit words
Honeywell-Multics	1024 36-bit word
IBM 370/XA and 370/ESA	4 Kbytes
VAX family	512 bytes
IBM AS/400	512 bytes
DEC Alpha	8 Kbytes
MIPS	4 kbyes to 16 Mbytes
UltraSPARC	8 Kbytes to 4 Mbytes
Pentium	4 Kbytes or 4 Mbytes
PowerPc	4 Kbytes
Itanium	4 Kbytes to 256 Mbytes

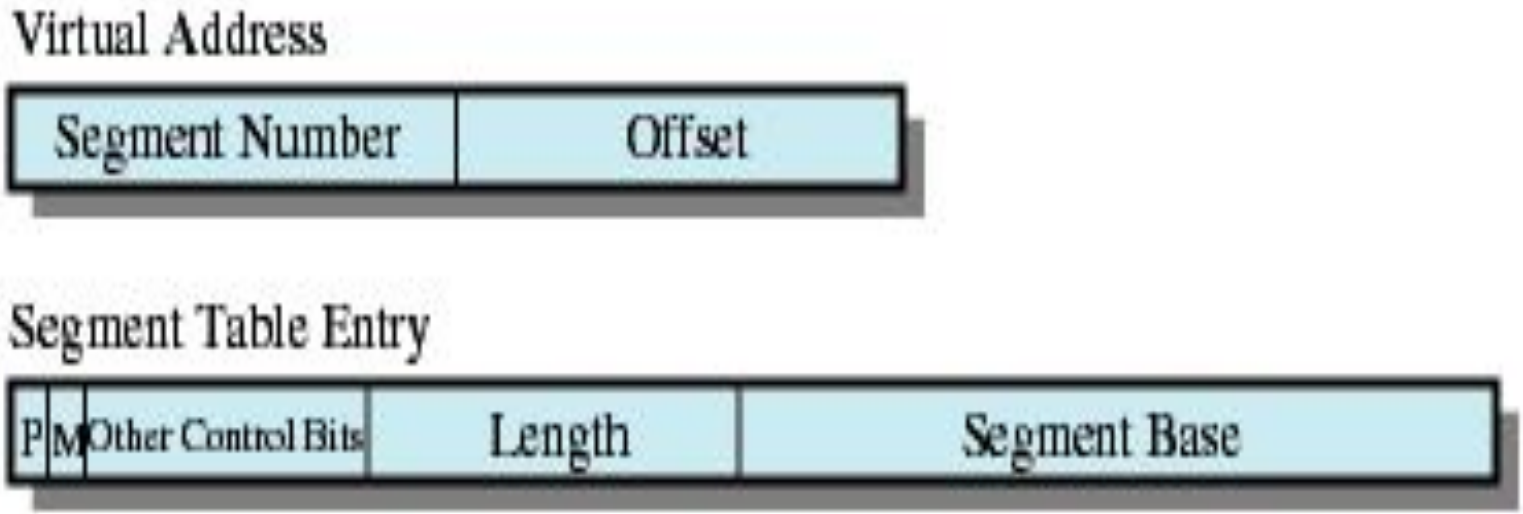
Сегментация

- Сегменты могут иметь разные, динамически изменяемые размеры
- Упрощается обработка структур данных изменяемого размера
- Упрощается совместное использование кода и данных разными процессами
- Улучшается защита.

Таблицы сегментов

- Располагается в основной памяти
- Содержит начальный адрес сегмента и его длину
- Существует бит присутствия сегмента в основной памяти
- Дополнительные биты: модификации, доступа.

Таблица сегментов



(b) Segmentation only

Трансляция адреса при сегментной организации

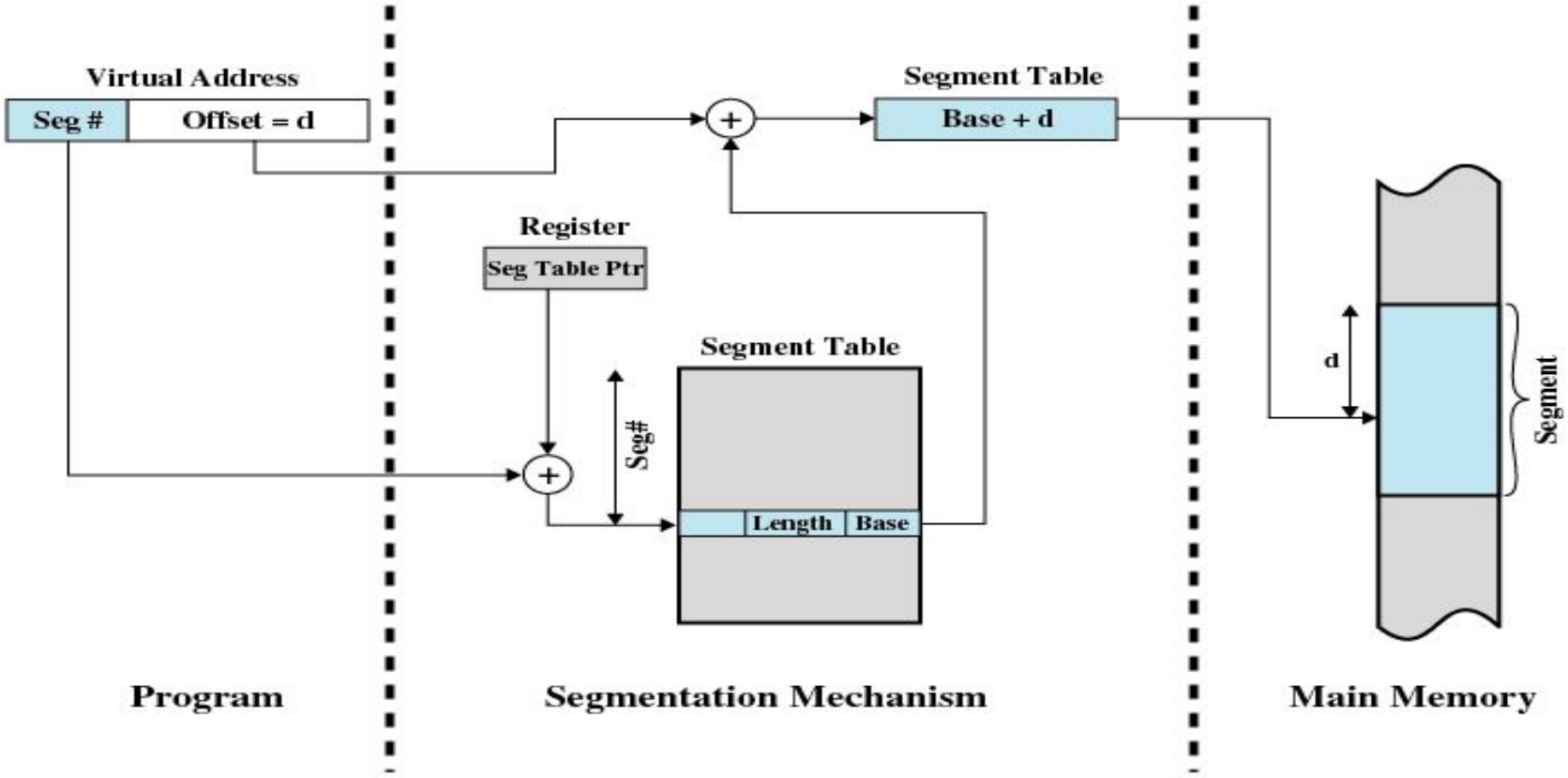
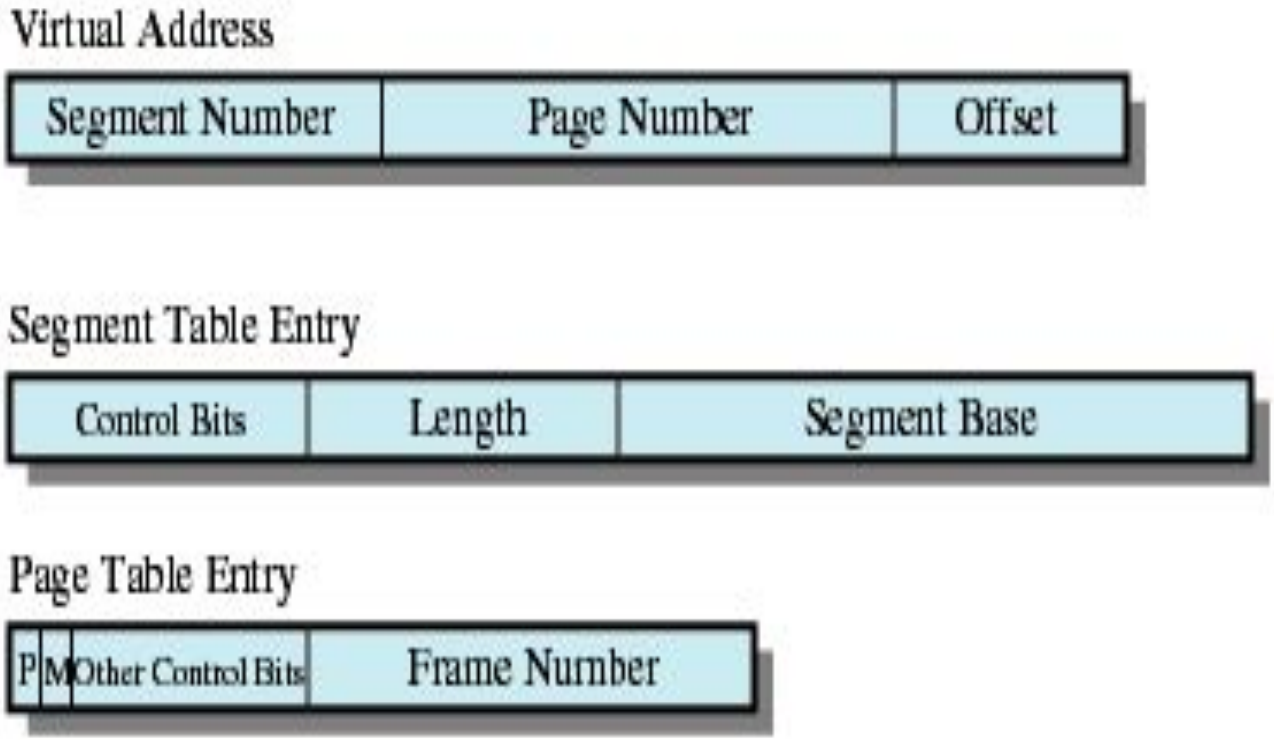


Figure 8.12 Address Translation in a Segmentation System

Сегментно-страничная организация

- Страничная организация
  - Прозрачна для программиста
  - Устраняет внешнюю фрагментацию
- Сегментная организация
  - Непрозрачна для программиста
  - Модульность
  - Возможность обработки структур данных переменной длины
  - Защита
  - Совместное использование памяти

Сегментно-страничная организация



P= present bit
M = Modified bit

(c) Combined segmentation and paging

Сегментно-страничная организация

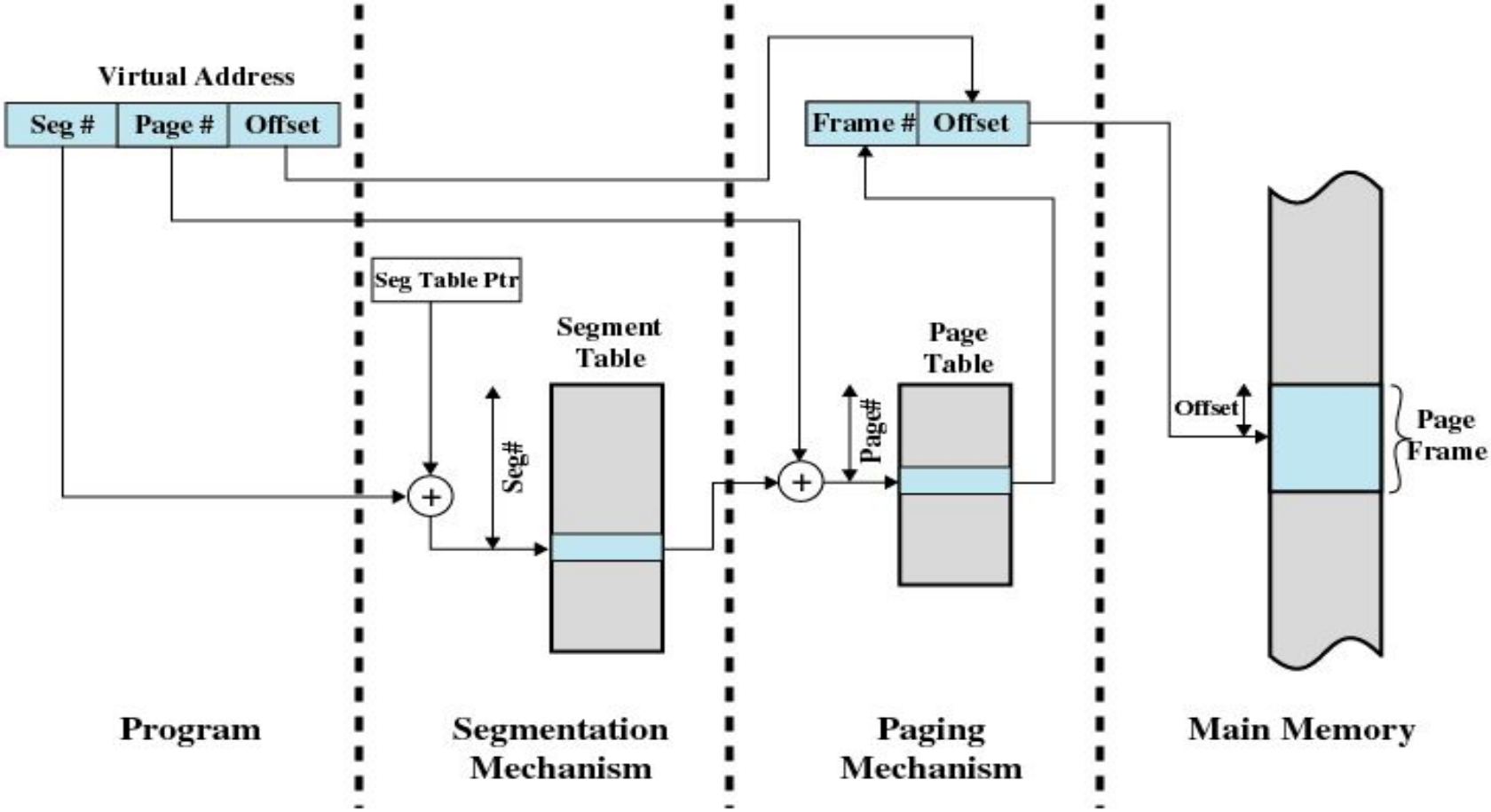


Figure 8.13 Address Translation in a Segmentation/Paging System

Задачи управления виртуальной памятью

Задача:

- Выборки
- Размещения
- Замещения
- Проецирования

Стратегия выборки (Fetch Policy)

- Определяет момент времени, когда страница должна быть помещена в основную память
  - По требованию
  - Предварительная выборка
    - Кластеризация (Windows 2000)
    - Интеллектуальная выборка (Windows XP)

Стратегия размещения

- Определяет место в основной памяти, где должен быть расположен блок
- Не имеет значения для страничной или сегментно-страничной организации.
- Приобретает первостепенную важность при построении систем с неоднородной памятью (например, на базе NUMA)

Стратегия замещения

- Определяет какую страницу основной памяти требуется заместить.
- Вероятность обращения к замещаемой странице в ближайшем будущем должна быть минимальной.
- Все стратегии замещения основаны на предсказании будущего использования страницы, основываясь на данных из прошлого.

Алгоритмы замещения

- Оптимальный
- Дольше всех не используемая (Least Recently Used – LRU)
- FIFO
- Часовой алгоритм (Clock Policy)

Оптимальный алгоритм

- Выбирает на замещение страницу к которой дольше всего **не будет** обращений.
- Реализовать не возможно, т.к. нельзя знать все будущие события в системе.

LRU

- Выбирает на замещение страницу которая дольше всех не использовалась
- В соответствие с принципом локализации считается, что в ближайшем будущем к этой странице не будет обращений.
- С каждой страницей должна быть сопоставлена информация о времени последнего использования.

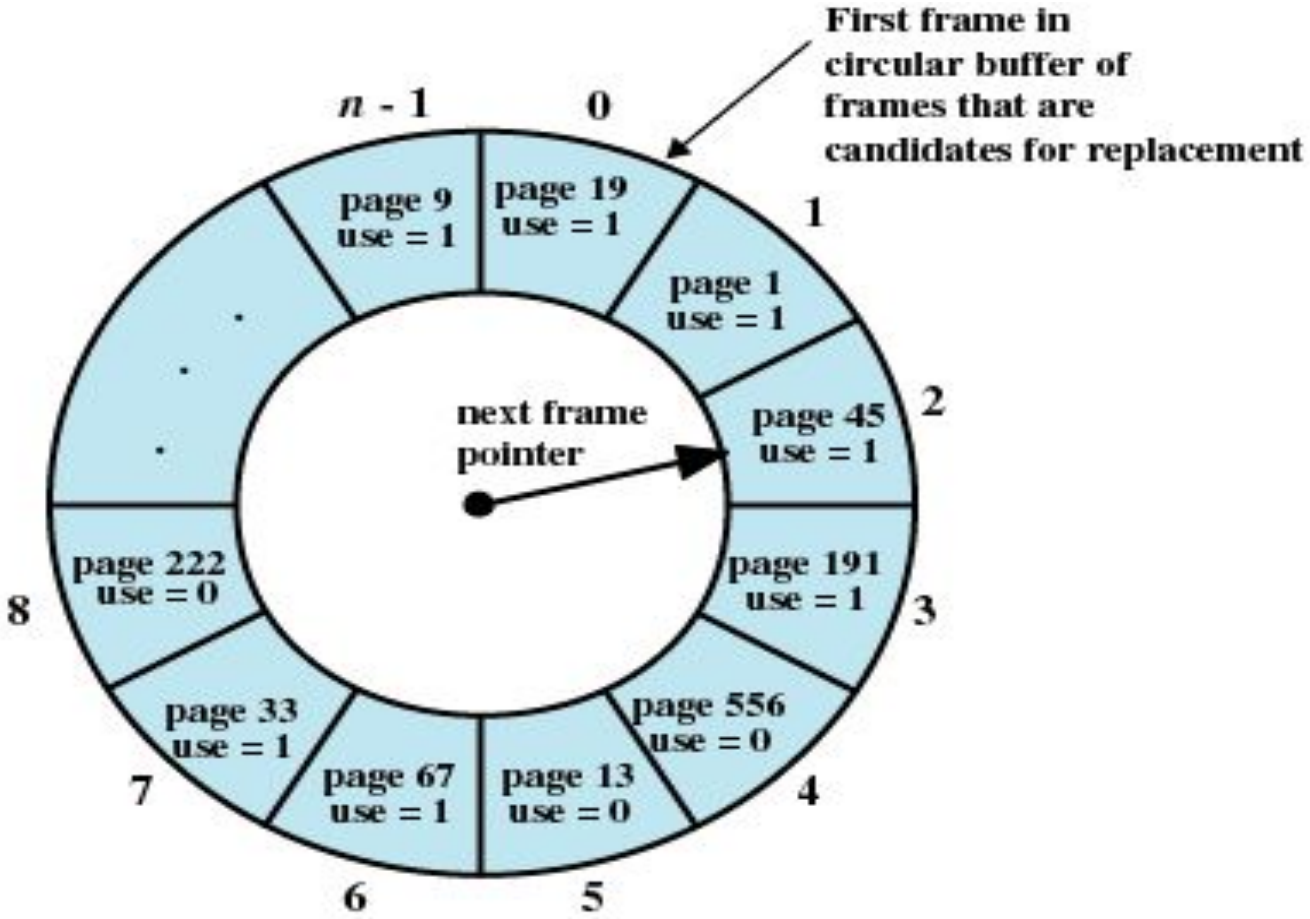
FIFO

- Рассматривает все страницы в памяти как круговой буфер.
- Страницы удаляются из буфера по принципу карусели (Round-robin).
- Самый простой алгоритм в реализации.
- Удаляется страница памяти, которая находится там дольше всего.
- Эта страница может снова понадобится в ближайшем будущем.

Clock Policy (версия 1)

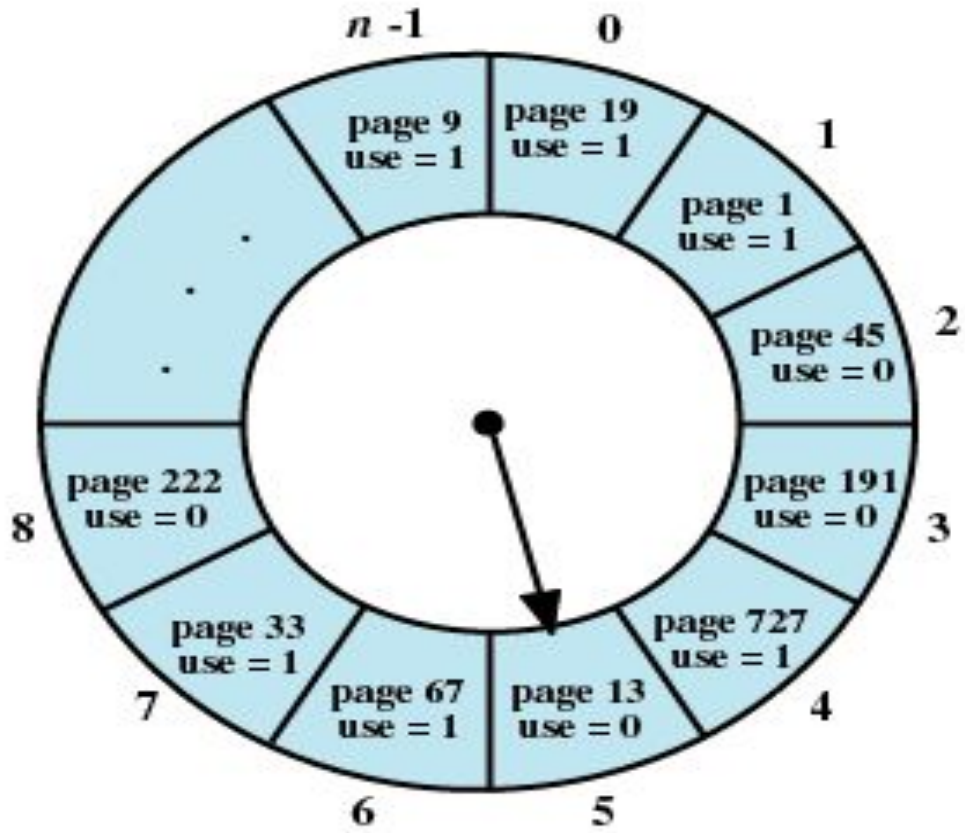
- Все загруженные страницы организованы в виде кругового буфера
- Существует бит использования (Use bit)
- При загрузке страницы в основную память бит устанавливается в 1
- При обращении к странице бит устанавливается в 1
- Замещению подвергается первая встреченная страница со сброшенным битом.
- При операции замещения все встреченные установленные биты сбрасываются.

Clock Policy (версия 1)



(a) State of buffer just prior to a page replacement

Clock Policy (версия 1)



(b) State of buffer just after the next page replacement

Figure 8.16 Example of Clock Policy Operation

Clock Policy. (версия 2)

- Кадры страниц разделены на категории
  - Не использован, не модифицирован (u=0, m=0)
  - Использован, не модифицирован (u=1, m=0)
  - Не использован, модифицирован (u=0, m=1)
  - Использован, модифицирован (u=1, m=1)

Clock Policy. (версия 2)

1. Сканируем буфер кадров, начиная с текущего положения. Бит u не изменяем. Первый же кадр (u=0, m=0) замещается.
2. Если шаг №1 не дал результатов, то ищем кадр с (u=0, m=1), во время сканирования сбрасываем биты использования. Перед замещением содержимое кадра сбрасывается на диск.
3. Если шаг №2 не дал результатов, то переходим к шагу №1.

Clock Policy (версия 2)

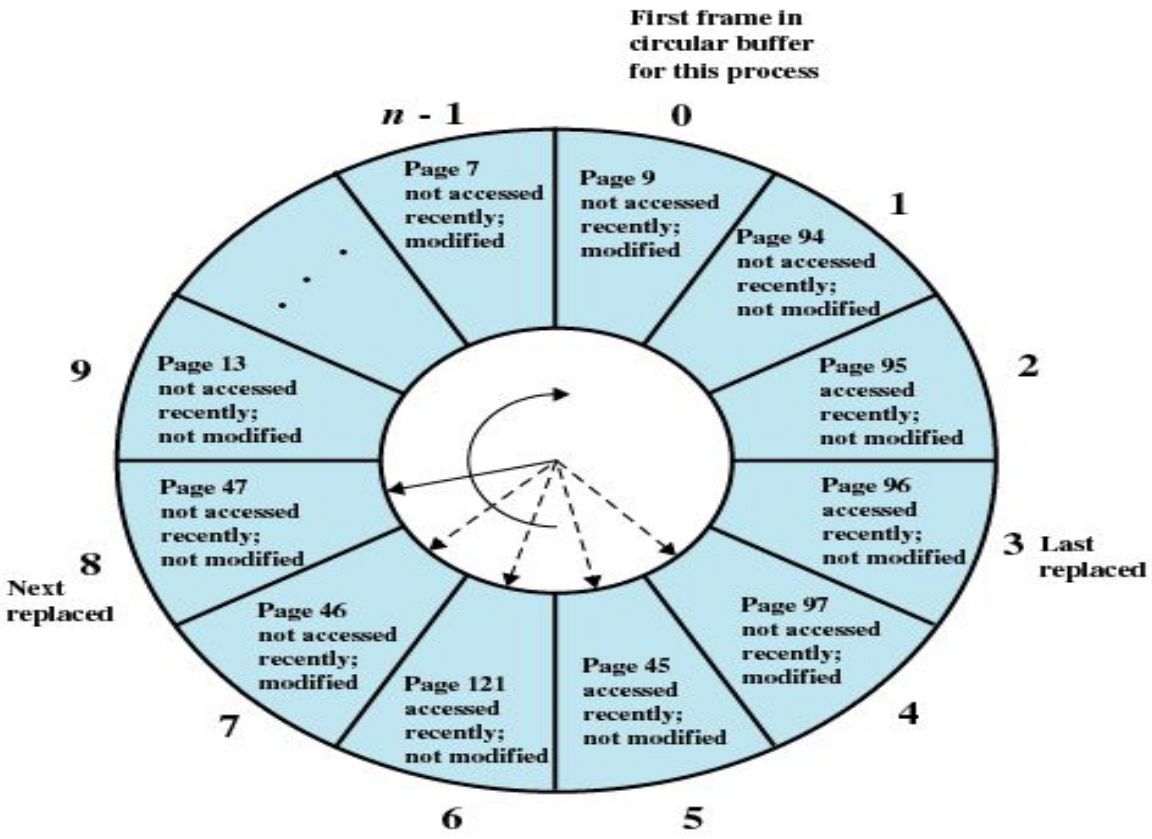


Figure 8.18 The Clock Page-Replacement Algorithm [GOLD89]

Сравнение алгоритмов замещения

Page address stream	2	3	2	1	5	2	4	5	3	2	5	2
OPT	2	2	2	2	2	2	4	4	4	2	2	2
		3	3	3	3	3	3	3	3	3	3	3
				1	5	5	5	5	5	5	5	5
					F		F			F		
LRU	2	2	2	2	2	2	2	2	3	3	3	3
		3	3	3	5	5	5	5	5	5	5	5
				1	1	1	4	4	4	2	2	2
					F		F		F	F		
FIFO	2	2	2	2	5	5	5	5	3	3	3	3
		3	3	3	3	2	2	2	2	2	5	5
				1	1	1	4	4	4	4	4	2
					F	F	F		F		F	F
CLOCK	2*	2*	2*	2*	5*	5*	5*	5*	3*	3*	3*	3*
		3*	3*	3*	3	2*	2*	2*	2	2*	2	2*
				1*	1	1	4*	4*	4	4	5*	5*
					F	F	F		F		F	

F = page fault occurring after the frame allocation is initially filled

Figure 8.15 Behavior of Four Page-Replacement Algorithms

Сравнение алгоритмов замещения

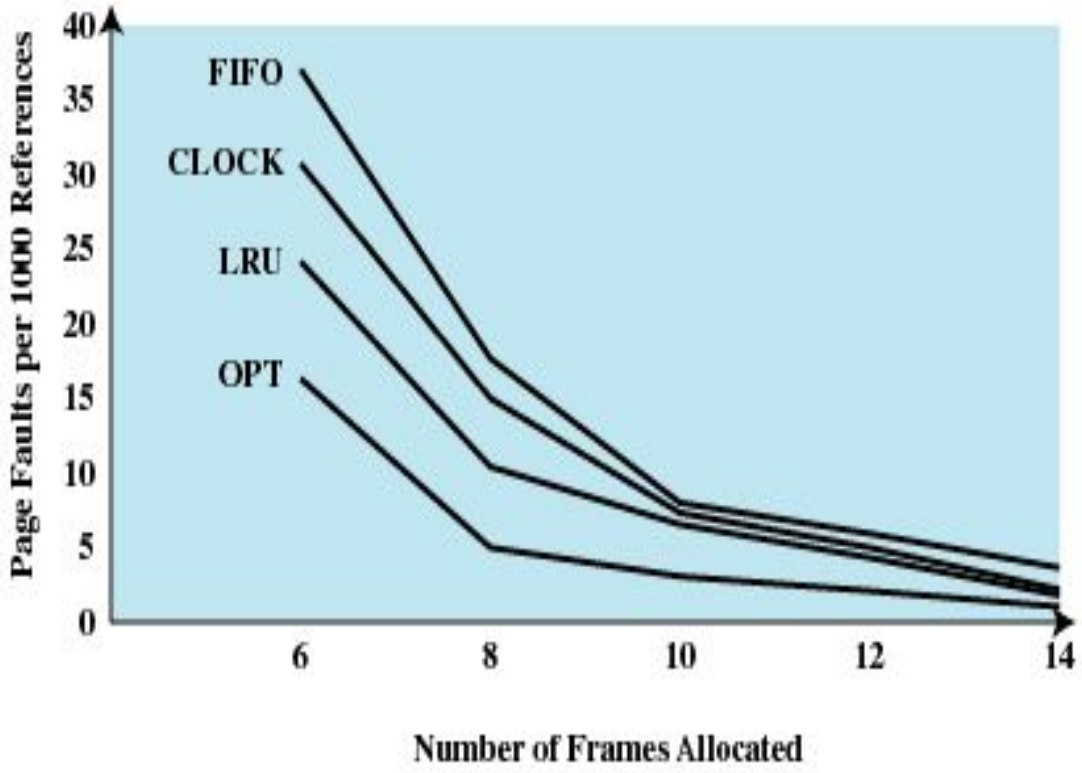


Figure 8.17 Comparison of Fixed-Allocation, Local Page Replacement Algorithms

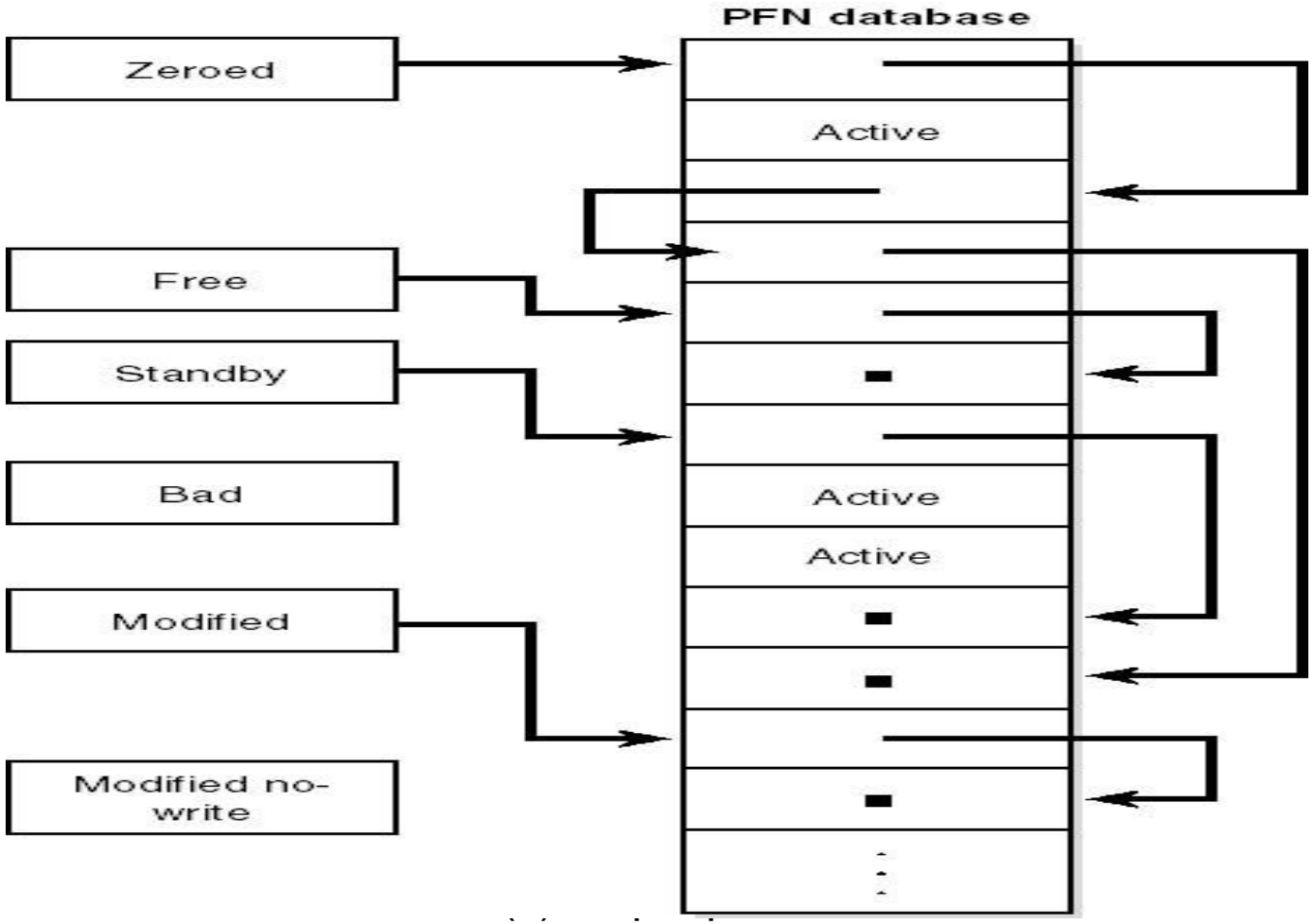
Буферизация страниц

- Система выбирает страницу на замещение по принципу FIFO
- Замещаемые страницы не удаляются из основной памяти, а попадают один из списков:
  - Список модифицированных страниц
  - Список свободных станиц

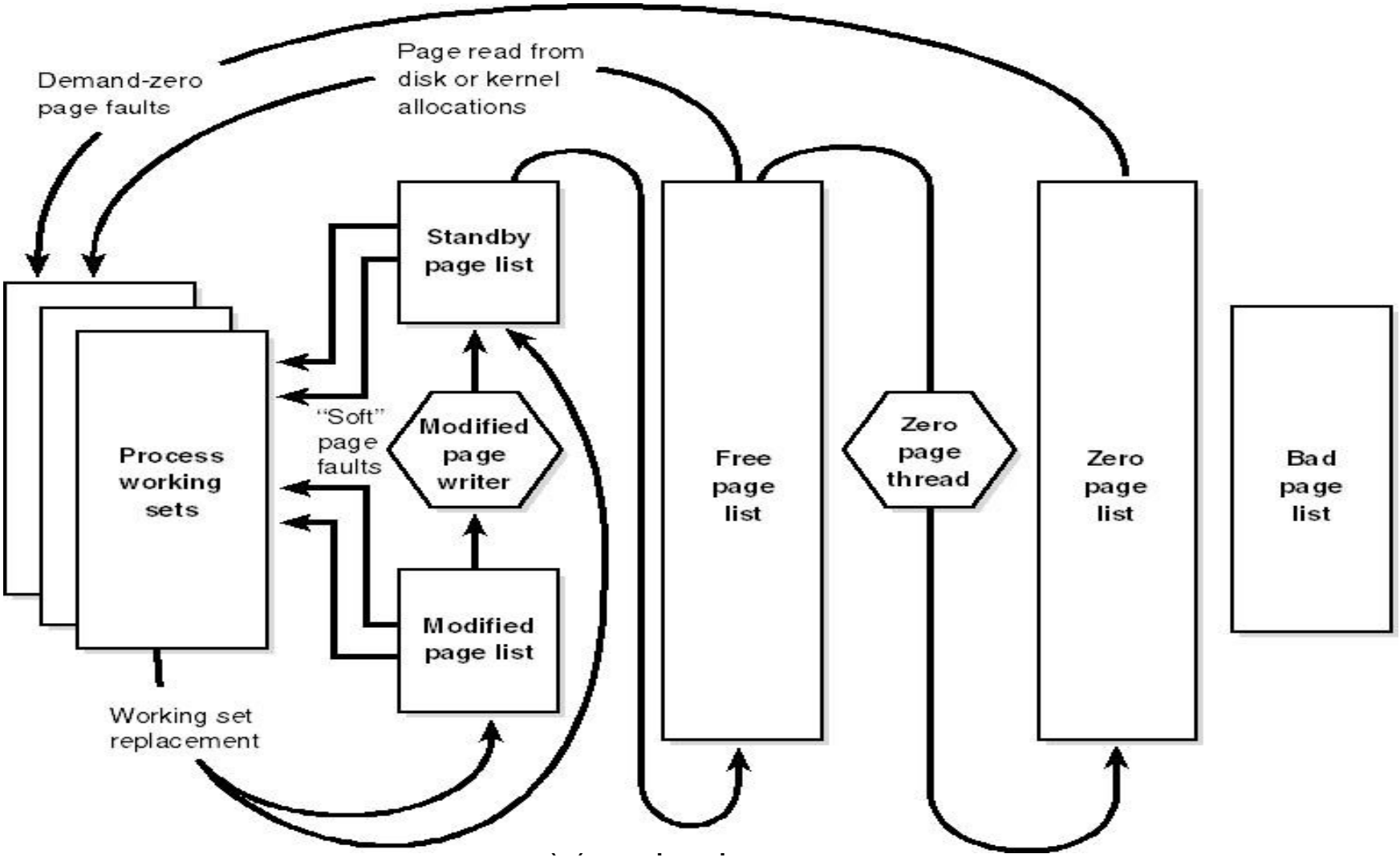
Использование буферизации страниц в Windows 2000

- Состояния фреймов страниц
  - Активная (Active/ Valid)
  - Переходная (Transition)
  - Простаивающая (Standby)
  - Модифицированная (Modified)
  - Модифицированная, но не записываемая (Modified no-write)
  - Свободная (Free)
  - Обнулённая (Zeroed)
  - Аварийная (Bad)

Организация списков страниц в базе данных PFN



Динамика списков страниц в Windows 2000



Управление резидентным множеством

Основной вопрос: сколько основной памяти требуется выделить процессу?

- Чем меньше памяти выделяется процессу, тем большее количество процессов может находится в основной памяти и тем выше вероятность существования готовых к исполнению процессов.
- При небольшом количестве страниц процесса в основной памяти частота страничных отказов достаточно высока.
- После определённого предела дополнительное выделение основной памяти процессу в соответствии с принципом локализации не будет приводить к значительному снижению частоты страничных отказов.

Размер резидентного множества

- Фиксированное распределение
  - Процессу выделяется фиксированное число страниц основной памяти для исполнения
  - При обработке страничного отказа система замещает страницы процесса
- Переменное распределение
  - Число страниц основной памяти, выделяемых процессу изменяется в течение жизни процесса.

Фиксированное распределение, локальное замещение.

- Необходимо заранее решить вопрос о количестве кадров, выделяемых процессу
- При выделении слишком малого количества памяти получаем высокую частоту страничных отказов
- При слишком большом количестве кадров получаем малое количество процессов в основной памяти.
- При исчерпании резерва резидентного множества страница замещается другой страницей того же процесса.

Переменное распределение, глобальная область видимости

- Операционная система держит список свободных страниц.
- При наступлении страничного отказа, операционная система включает страницу в резидентное множество процесса.
- При нехватке свободных страниц система производит операцию замещения страницы.

Переменное распределение, локальная область видимости

- При загрузке процесса ему в качестве резидентного множества выделяется некоторое количество страниц, исходя из типа приложения, запроса программы или других критериев. Для заполнения рабочего множества используется стратегия выборки по требованию или предварительная выборка.
- При возникновении страничного отказа и заполнении резидентного множества страница для замещения выбирается среди резидентного множества процесса, сгенерировавшего отказ.
- Система периодически проводит переоценку распределения памяти процессам.

Стратегия рабочего множества (working set strategy)

$W(t, \Delta)$

множество страниц к которым

обращался процесс в течение времени Δ,

начиная со времени t

$W(t, \Delta + 1) \supseteq W(t, \Delta)$

рабочее множество есть

неубывающая функция от размера окна

$1 \leq | W(t, \Delta) \leq \min(\Delta, N)$

рабочее множество

процесса может расти с той же скоростью,

что и количество страниц процесса N,

если происходят обращения к различным страницам

и если позволяет размер окна.

Стратегия рабочего множества

Последовательность обращений к страницам	Размер окна Δ			
	2	3	4	5
24	24	24	24	24
15	24 15	24 15	24 15	24 15
18	15 18	24 15 18	24 15 18	24 15 18
23	18 23	15 18 23	24 15 18 23	24 15 18 23
24	23 24	18 23 24	•	•
17	24 17	23 24 17	18 23 24 17	15 18 23 24 17
18	17 18	24 17 18	•	18 23 24 17
24	18 24	•	24 17 18	•
18	•	18 24	•	24 17 18
17	18 17	24 18 17	•	•
17	17	18 17	•	•
15	17 15	17 15	18 17 15	24 18 17 15
24	18 24	17 15 24	17 15 24	•
17	24 17	•	•	17 15 24
24	•	24 17	•	•
18	24 18	17 24 18	17 24 18	15 17 24 18

Управление рабочими множествами (наборами) в Windows 2000

Диспетчер рабочих наборов
Диспетчер настройки баланса

- Все процессы начинают жизненный цикл с одинаковыми максимальными и минимальными размерами рабочего набора (345 и 50 страниц соответственно для систем с большим объёмом памяти)
- При возникновении страничного отказа система, система проверяет лимиты рабочего набора процесса и объём свободной памяти.
- Если условия позволяют диспетчер памяти разрешает процессу увеличить размер рабочего множества до максимума и даже превысить его, если достаточно свободных страниц.
- При нехватке памяти система заменяет страницы в рабочем наборе.

Управление рабочими наборами в Windows 2000

- При слишком частой генерации модифицированных страниц вызывается диспетчер рабочих наборов, который инициирует автоматическое усечение рабочего набора для увеличения объёма свободной памяти.
  - Подсчитывает, сколько страниц при необходимости можно изъять из рабочего набора процессов, если их рабочий набор превышает минимальный.
  - Оптимально упорядочивает список процессов – кандидатов на усечение рабочего набора.

Управление рабочими наборами в Windows 2000.

- Если с момента усечения рабочего набора процесс вызовет определённое количество страничных отказов, он исключается из числа кандидатов на усечение до следующего цикла усечения (через 6 секунд)
- Для усечения рабочего набора используется часовой алгоритм для определения исключаемых страниц (начиная с Windows XP/2003 используется единый алгоритм для много и однопроцессорных систем)

Список литературы

1. Столлингс, Вильям. **Операционные системы**, 4-е издание. «Вильямс», 2002.
2. Соломон Д., Руссинович М. **Внутреннее устройство Microsoft Windows 2000**. СПб.: Питер, «Русская Редакция», 2001.
3. Рихтер Дж. **Windows для профессионалов: программирование для Windows 95 и Windows NT 4 на базе Win32 API**. «Русская Редакция», 1997.