

# Управление памятью в ОС Windows

Общие принципы управления виртуальной памятью в Win32

# Общие принципы управления виртуальной памятью в Win32

Менеджер виртуальной памяти и архитектура Win32 API

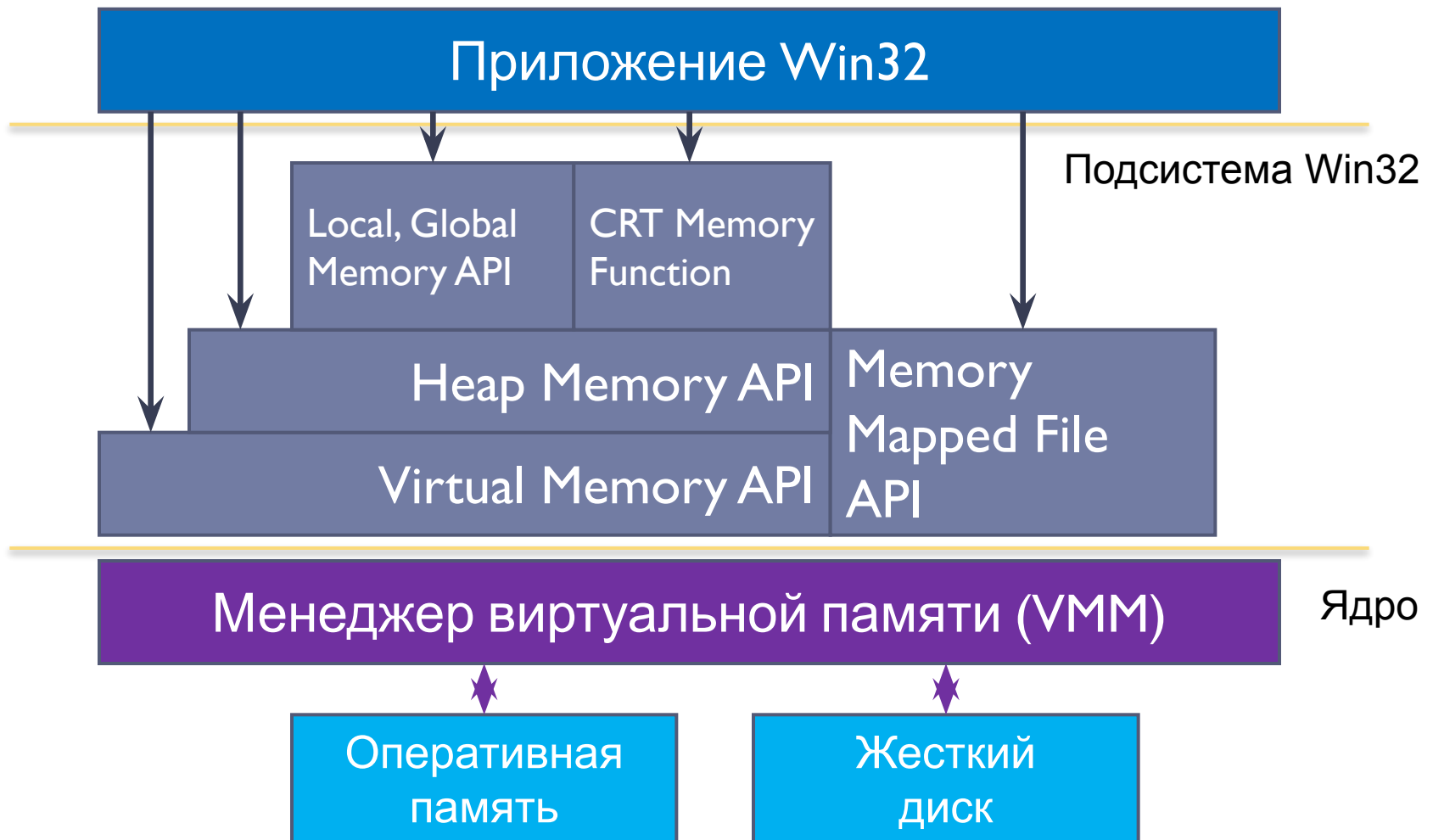
# Менеджер виртуальной памяти

---

- Менеджер виртуальной памяти (VMM) является составной частью ядра ОС. Приложения не могут получить к нему прямой доступ.
- Основные функции VMM:
  - управление виртуальными адресными пространствами процессов;
  - разделение памяти между процессами;
  - защита виртуальной памяти одного процесса от других процессов.



# Архитектура API управления 32-разрядной памятью



# Интерфейсы управления памятью

---

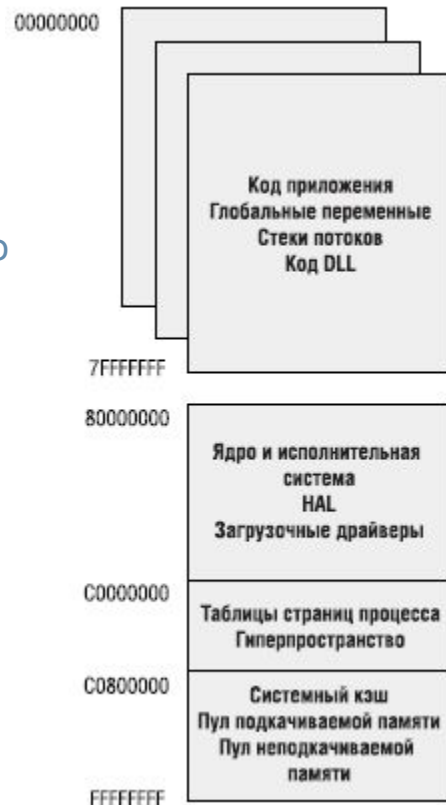
- **Virtual Memory API** – набор функций, позволяющих приложению работать с виртуальным адресным пространством: назначать физические страницы блоку адресов и освобождать их, устанавливать атрибуты защиты.
- **Memory Mapped File API** – набор функций, позволяющий работать с файлами, отображаемыми в память.
- **Heap Memory API** – набор функций, позволяющих работать с динамически распределяемыми областями памяти (кучами).
- **Local, Global Memory API** – набор функций работы с памятью, совместимых с 16-разрядной Windows. Следует избегать их использования.
- **CRT Memory API** – функции стандартной библиотеки языка Си периода исполнения (runtime).



# Виртуальное адресное пространство процесса

2 Гб  
для  
пользовательского  
пространства

2 Гб  
для нужд ОС



3 Гб  
для  
пользовательского  
пространства

1 Гб  
для нужд ОС



# Расширенное пользовательское пространство

---

- Чтобы включить в Windows Server 2003 и Windows 2000 расширенное пользовательское пространство, необходимо указать в файле Boot.ini ключ */3GB*.
- Windows XP и Windows Server 2003 поддерживают дополнительный ключ (*/USERVA*), который дает возможность задавать размер пользовательского адресного пространства между 2 и 3 Гб (значение указывается в мегабайтах).
- Приложение для адресации 3 Гб должно быть собрано с ключом */LARGEADDRESSAWARE:YES*.



# Страничное преобразование

---

- Виртуальная память в Windows 2000+ имеет страничную организацию.
- Каждому процессу Windows назначается свой каталог страниц. Именно поэтому адресное пространство каждого процесса изолировано, что очень хорошо с точки зрения защиты процессов друг от друга.
- Процессоры Intel начиная с Pentium Pro позволяют применять одно-, двух- и трехступенчатые схемы, также разрешается одновременное использование страниц различного размера.





# Размер страницы

---

Архитектура	«Малая» страница	«Большая» страница
x86	4 КБайта	2 Мбайта (режим PAE), 4 Мбайта (PSE-36)
x64	4 КБайта	2 МБайта
IA64	8 КБайт	16 МБайт



# Средства защиты памяти

---

- ❑ **Объектно-ориентированная защита памяти.** Каждый раз, когда процесс открывает указатель на блок адресов, монитор ссылок безопасности проверяет, разрешен ли доступ процесса к данному объекту.
- ❑ **Отдельное адресное пространство для каждого процесса.** Аппаратура запрещает процессу доступ к физическим адресам другого процесса.
- ❑ **Два режима работы:** режим ядра, в котором процессам разрешен доступ к системным данным, и пользовательский режим, в котором это запрещено.
- ❑ **Страничный механизм защиты.** Каждая виртуальная страница имеет набор признаков, который определяет разрешенные типы доступа в пользовательском режиме и в режиме ядра.
- ❑ **Принудительная очистка страниц,** освобождаемых процессами.

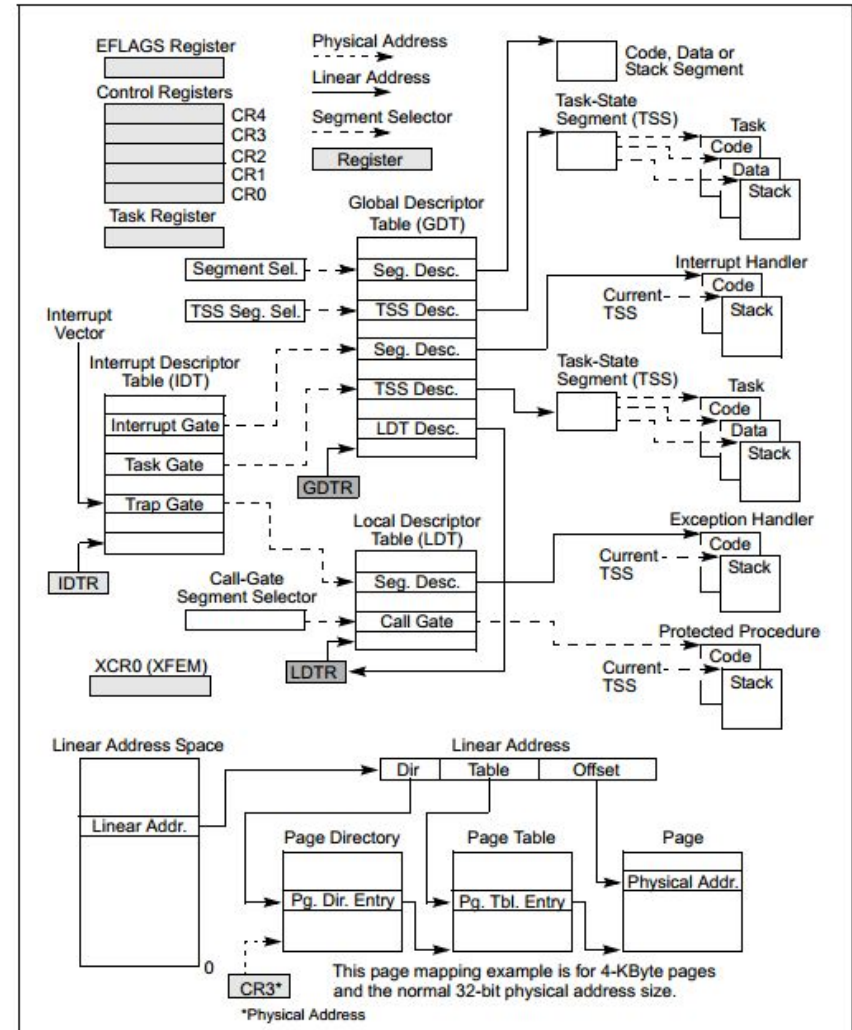


# Общие принципы управления виртуальной памятью в Win32

Управлению памятью на архитектуре IA-32

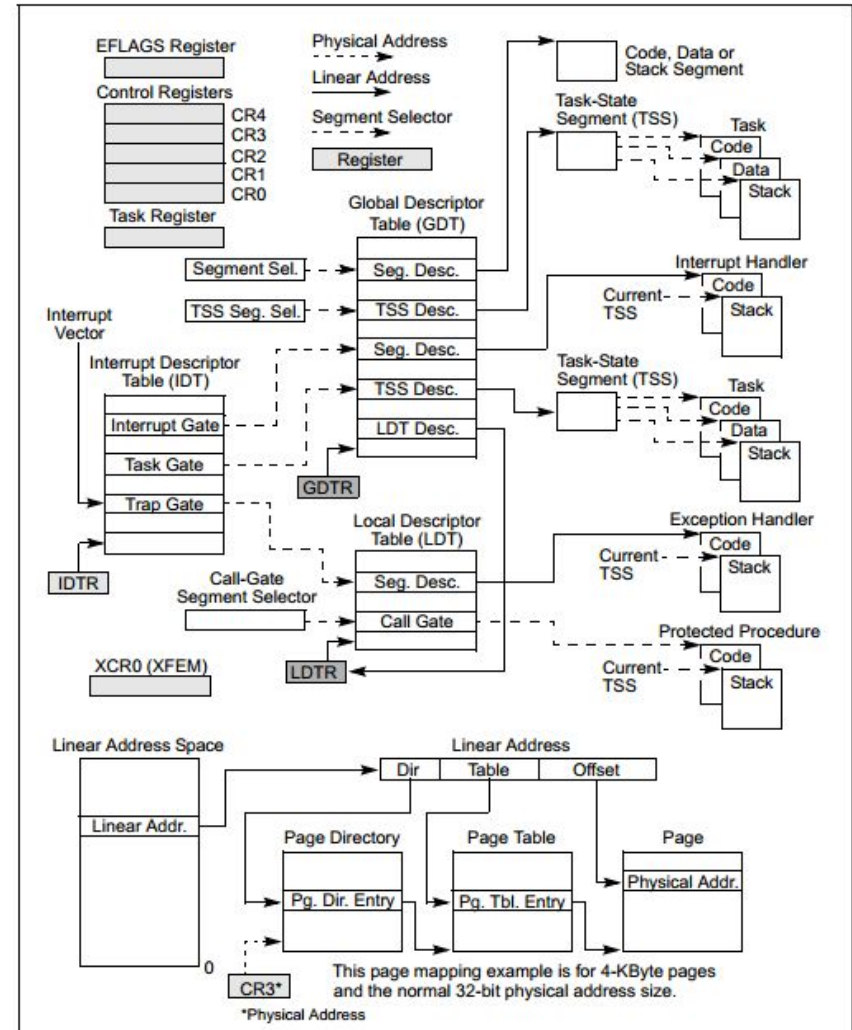
# Архитектура процессора IA-32 (1)

- Все 32-разрядные процессоры, начиная с i386, имеют набор системных **регистров**, предназначенных для использования в защищённом режиме, среди них есть **регистры управления CR0, CR1, CR3 и CR4** (был введен в процессоре Pentium).
- Регистры управления, в основном, состоят из флагов. Назначение и использование каждого флага требует отдельного рассмотрения.



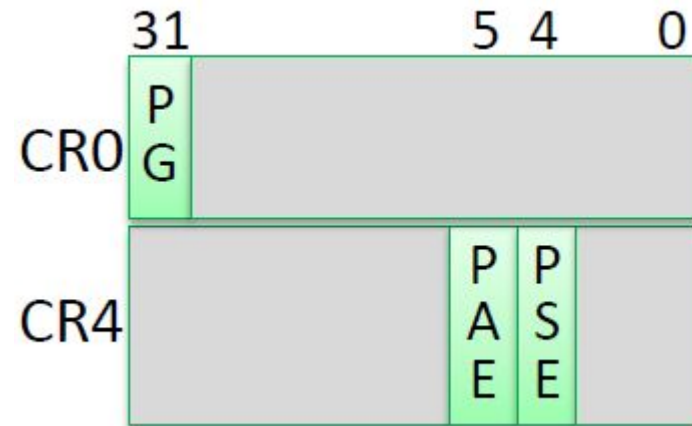
# Архитектура процессора IA-32 (2)

- Например, младший бит из регистра **CR0** называется PE (Protection Enable). Если установить этот бит в 1, процессор перейдёт в защищённый режим, если сбросить – то в режим реальных адресов.



# Регистры управления памятью на процессоре IA-32

- Отдельные биты регистров **CR0** и **CR4** отвечают за управление памятью:
  - **PG (Paging)** включает использование страничного преобразования;
  - **PSE (Page Size Extension)** управляет размером страницы;
  - **PAE (Page Address Extension)** включает режим расширения физического адресного пространства (36 бит).
- Кроме того, 17-ый бит регистра EDX процессора (CPUID.01H:EDX[bit17]) говорит о поддержке специального режима PSE-36, который позволяет использовать 36-битную адресацию физической памяти при размере страницы равном 4 МБайт.



# Режим PAE

---

- Расширение физических адресов (Physical Address Extension – PAE) – режим работы встроенного блока управления памятью процессоров с архитектурой IA-32, в котором используются 64-битные элементы таблиц страниц (из которых для адресации используются только 36 бит).
- PAE делает возможной адресацию процессором 64 Гбайт физической памяти, хотя каждый процесс всё равно может адресовать максимум до 4 Гбайт адресов виртуальной памяти.
- Кроме того, режим PAE позволяет использовать «большие» страницы размером 2 Мбайта.



# Поддержка PAE в различных операционных системах

---

- В 32-разрядных Microsoft Windows (начиная с Windows XP SP2) использование 36-битного PAE включается ключом /PAE в файле boot.ini.
- Одним из пунктов минимальных системных требований Windows 8 является обязательная поддержка процессором PAE.
- Linux начиная с версии 2.3.23.
- FreeBSD поддерживает PAE: в линейке 4.x версий – начиная с 4.9, в линейке 5.x версий – начиная с 5.1, все 6.x и более поздние.
- Solaris поддерживает PAE, начиная с версии 7.
- В Mac OS X режим PAE включён по умолчанию при использовании 32-разрядного ядра.





# Сводная информация по управлению памятью в IA-32

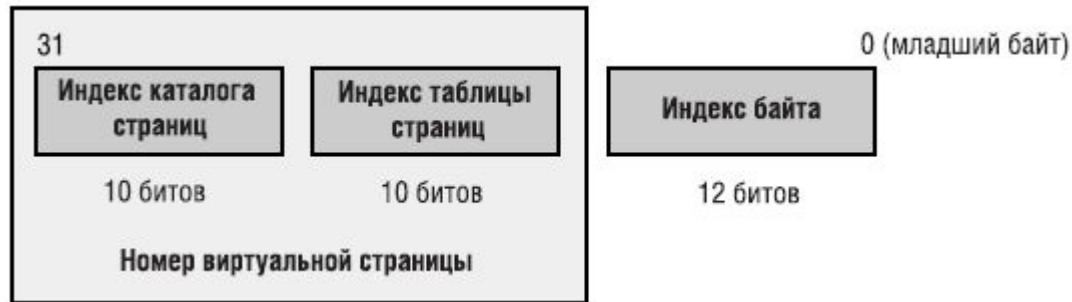
- Таким образом, архитектура IA-32 поддерживает страницы нескольких размеров (4 КБайт, 2 МБайт и 4 МБайт), а также возможность адресации до 64 ГБайт физического адресного пространства.

Размер страницы	Разрядность физического адреса	Размер физической памяти	PSE-36	PSE	PAE
4 KB	32 bits	4 GB		0	0
4 MB	32 bits	4 GB	No	1	0
4 MB	36 bits	64 GB	Yes	1	0
4 KB	36 bits	64 GB		0	1
2 MB	36 bits	64 GB		1	1

# Общие принципы управления виртуальной памятью в Win32

Реализация страничного преобразования

# Формат 32-разрядного виртуального адреса в системах x86 (страница 4 КБайт)



- Старшие 10 разрядов адреса определяют номер одного из 1024 элементов в каталоге страниц, адрес которого находится в регистре процессора **CR3**. Этот элемент содержит физический адрес таблицы страниц.
- Следующие 10 разрядов линейного адреса определяют номер элемента таблицы. Элемент, в свою очередь, содержит физический адрес страницы виртуальной памяти.
- Размер страницы – 4 Кбайт, и младших 12 разрядов линейного адреса как раз хватает ( $2^{12} = 4096$ ), чтобы определить точный физический номер адресуемой ячейки памяти внутри этой страницы.

# Адресация больших страниц для x86-архитектуры

## □ PSE-36



## □ PAE



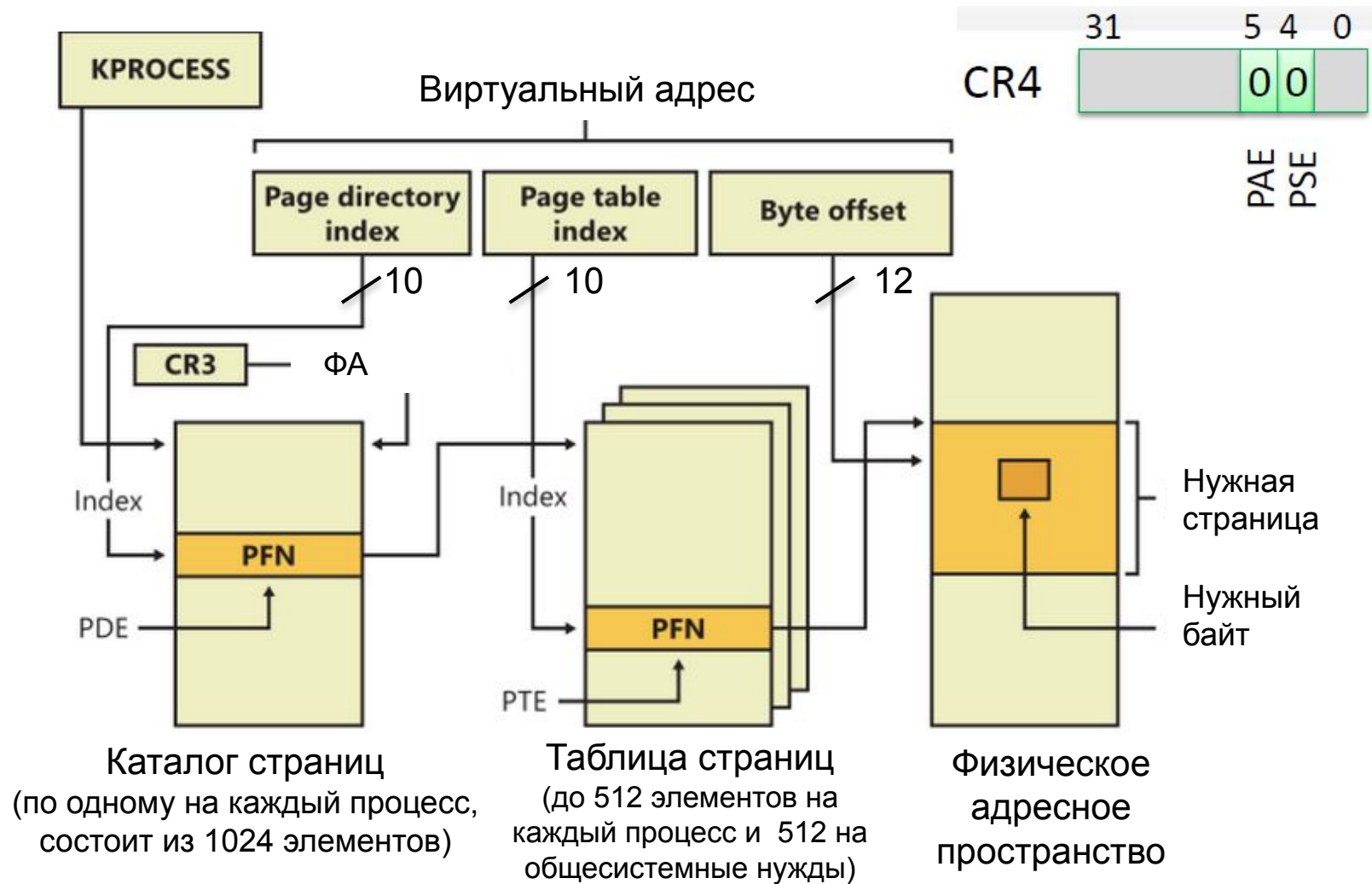
# Вопрос

---

- Какие проблемы Вы видите при использовании драйверами операционной системы «больших» страниц?



# Трансляция виртуального адреса в системах x86 (страница 4 КБайт)



# Формат PTE для страницы размером 4 Кбайта



- Таблица страниц – это массив записей, состоящий из  $2^{10}$  4-байтовых элементов PTE (Page Table Entries).
- Каждый элемент PTE определяет состояние отдельной страницы размером 4 Кбайта.
- Если страница находится в оперативной памяти (бит «P» = 1), то PTE указывает адрес соответствующей страницы физической памяти.

# БИТЫ PTE

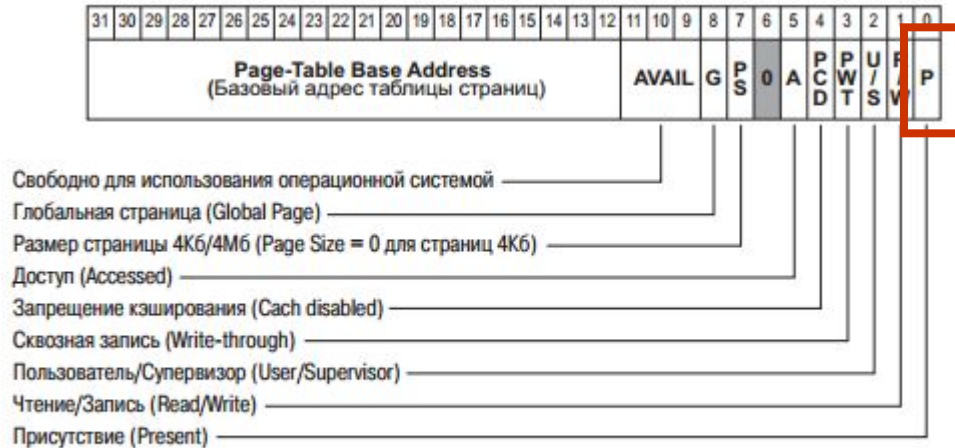
---

- Global (**G**) – страница относится ко всем процессам
  - Page Table Attribute Index (**PAT**) – используется процессором (начиная с Pentium Pro) вместе с битами **PCD** и **PWT** для определения по специальной внутренней программируемой *PAT-таблице* режима кэширования соответствующей страницы
  - Dirty (**D**) – страница была изменена (была произведена запись)
  - Accessed (**A**) – к странице был осуществлен доступ
  - Cache disabled (**PCD**) – кэширование данной страницы отключено
  - Write through (**PWT**) – включает режим сквозной записи при кэшировании
  - User/Supervisor (**U/S**) – доступна ли страница для пользовательского кода
  - Read/Write (**R/W**) – в однопроцессорных системах указывает разрешение на запись в страницу (страница для чтения и записи = 1 или только для чтения = 0)
- 





# Каталог страниц и формат PDE для страниц размером 4 Кбайта

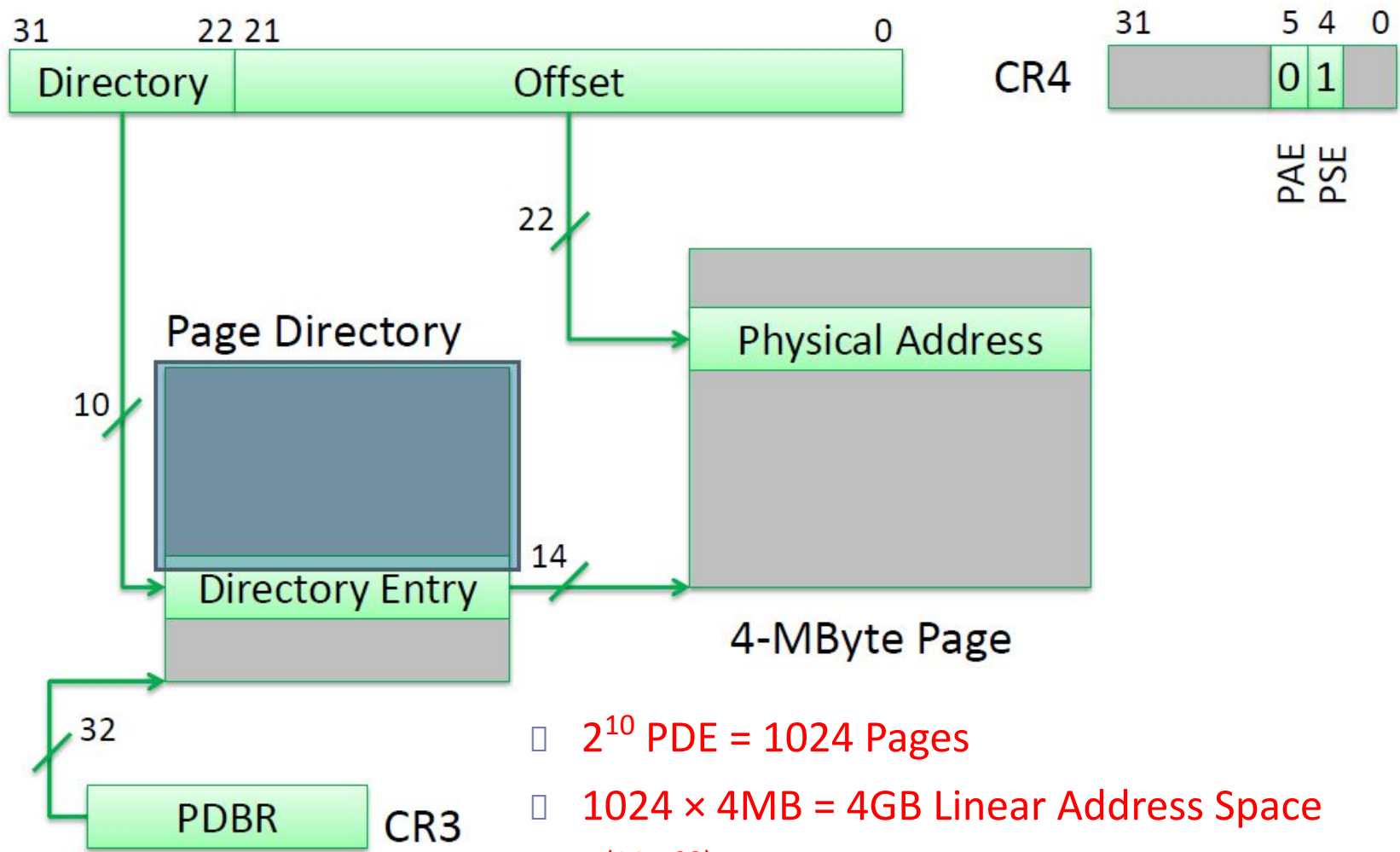


«P» – бит присутствия таблицы в ОП

■ Резервированные биты. Должны устанавливаться только в указанные значения.

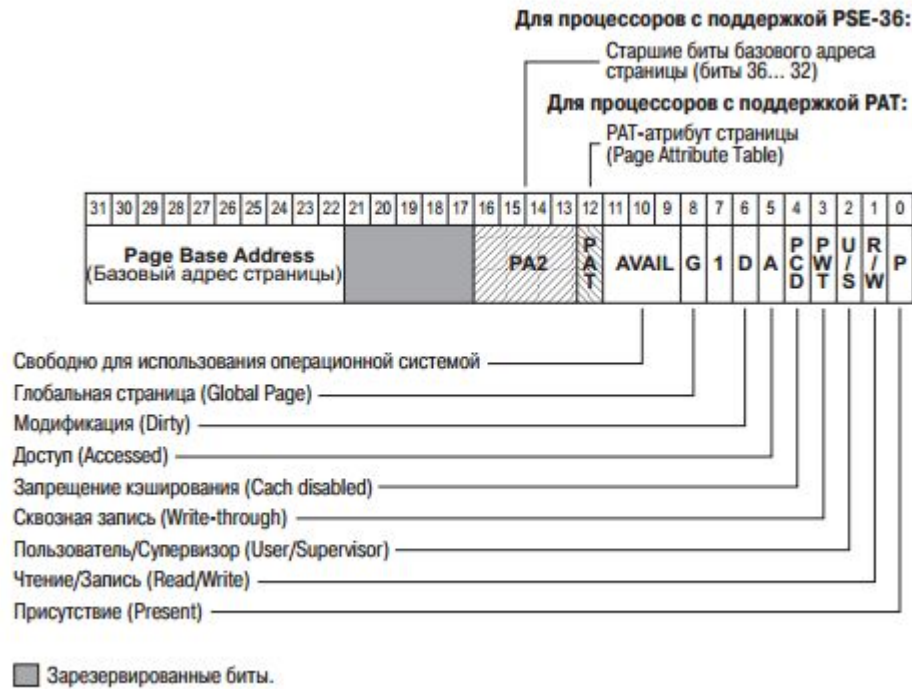
- Каталог страниц – это массив, состоящий из  $2^{10}$  4-байтовых элементов PDE (Page Directory Entries).
- Каждый элемент PDE определяет положение таблицы страниц, указывая адрес физической страницы памяти (бит «P» = 1).
- По формату PDE почти совпадает с PTE.

# Трансляция виртуального адреса в системах x86 (страница 4 МБайт)



- $2^{10}$  PDE = 1024 Pages
- $1024 \times 4\text{MB} = 4\text{GB}$  Linear Address Space
- $2^{(14 + 22)} = 64\text{GB}$  Physical Address Space

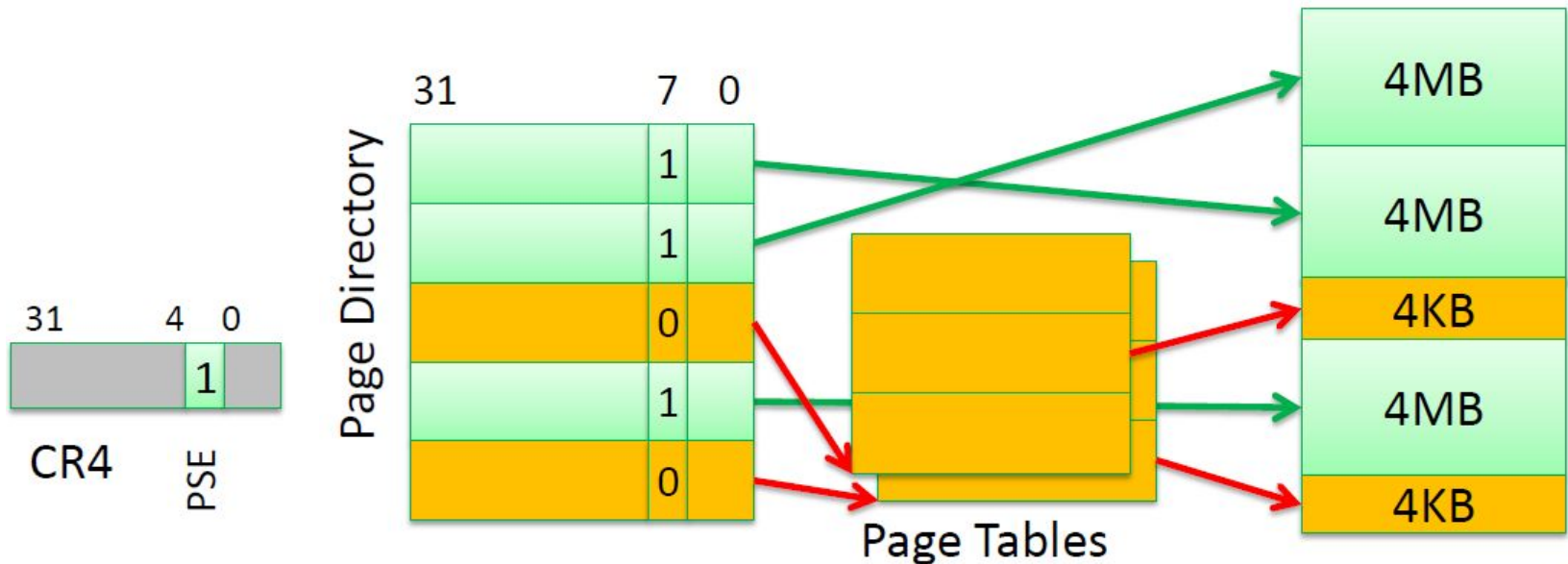
# Каталог страниц и формат PDE для страниц размером 4 МБайта



- Каталог страниц – это массив, состоящий из  $2^{10}$  4-байтовых элементов PDE (Page Directory Entries).
- Каждый элемент PDE определяет положение **страницы размером 4 Мбайта**, указывая адрес физической страницы памяти (бит «P» = 1).

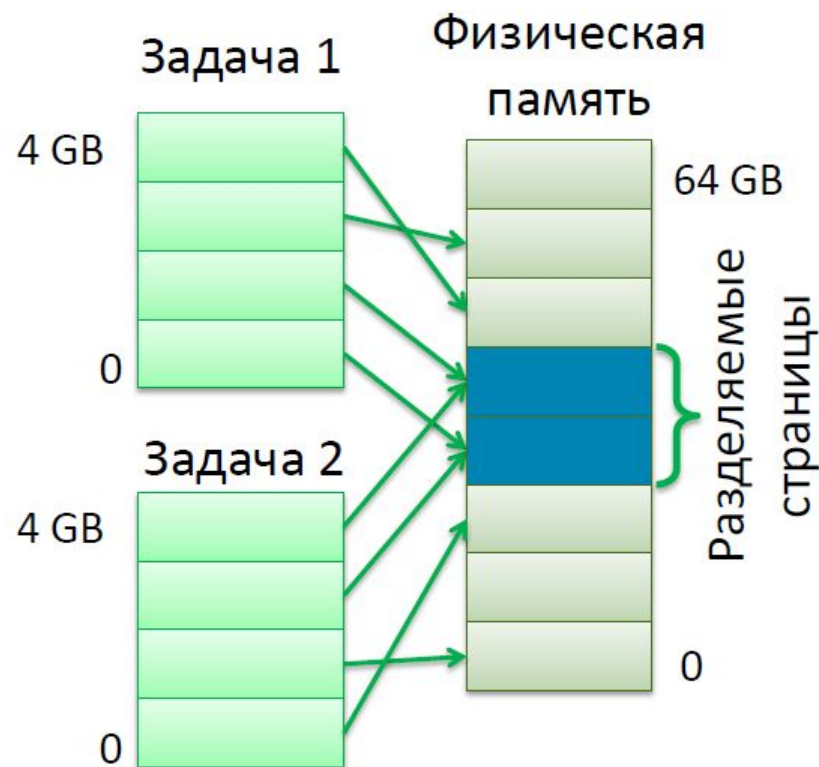
# Совместное использование страниц разного размера

- При установленном бите PSE регистра **CR4** возможно одновременное использование 4KB и 4MB страниц.
- Управление размером страницы осуществляется битом Page Size (PS, бит 7) элемента каталога страниц PDE:
  - PS=1 – страница 4MB и PDE указывает на страницу;
  - PS=0 – страница 4KB и PDE указывает на таблицу страниц.



# Расширение физического адресного пространства

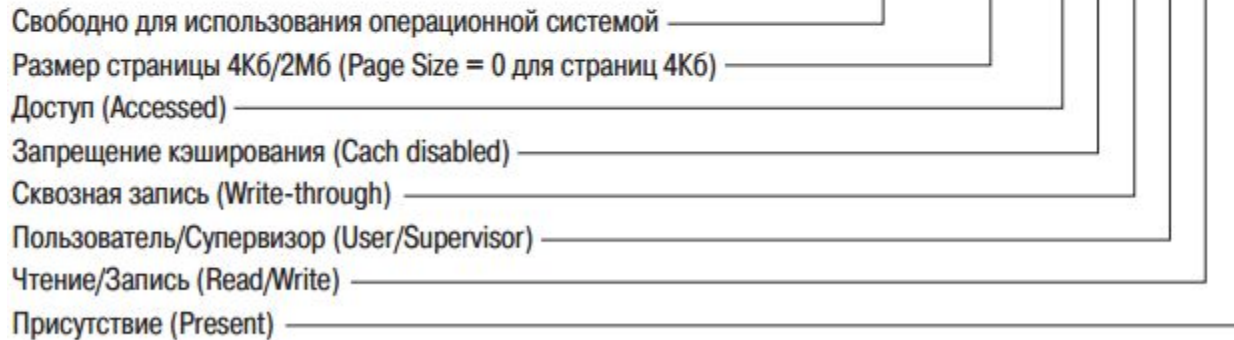
- Благодаря поддержке процессором механизма расширения физического адресного пространства (PAE - Physical Address Extension), операционная система может использовать 36-разрядное пространство для организации многозадачности.
- При этом процессам, по-прежнему, остается доступным только 32-разрядное пространство.
- Кроме того процессор допускает множественные ссылки на страницу (Memory Aliasing).



# Формат PDE в режиме PAE

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32					
Зарезервировано (все 0)																																		Page Table Base Address		

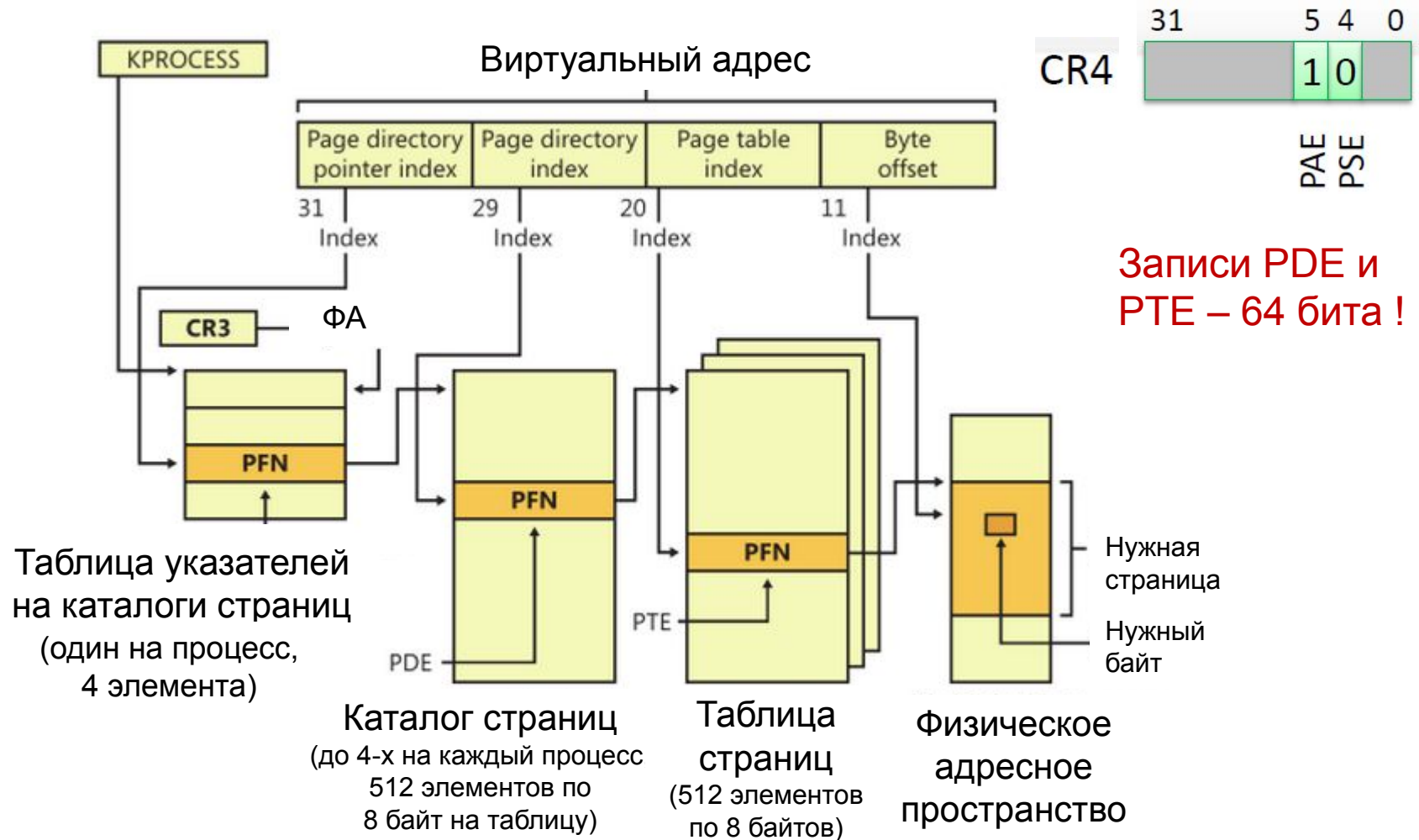
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Page-Table Base Address (Базовый адрес таблицы страниц)																					AVAIL	0	0	0	A	P C D	P W T	U / S	R / W	P	



Зарезервированные биты. Должны устанавливаться только в указанные значения.



# Реализация механизма PAE для страниц размером 4 Кбайта

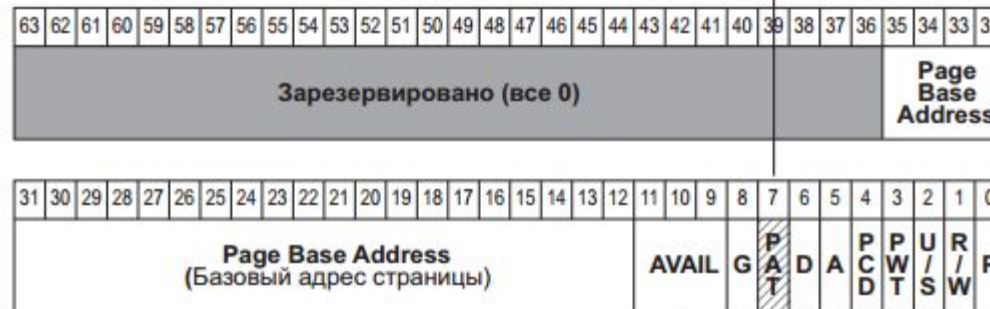




# Формат PTE для страницы 4 КБайта

Для процессоров с поддержкой PAT:

PAT-атрибут страницы (Page Attribute Table)



- Свободно для использования операционной системой
- Глобальная страница (Global Page)
- Модификация (Dirty)
- Доступ (Accessed)
- Запрещение кэширования (Cach disabled)
- Сквозная запись (Write-through)
- Пользователь/Супервизор (User/Supervisor)
- Чтение/Запись (Read/Write)
- Присутствие (Present)

Зарезервированные биты. Должны устанавливаться только в указанные значения.





# Формат PDE для страницы 2 МБайта

Для процессоров с поддержкой PAT:

PAT-атрибут страницы (Page Attribute Table)



- Свободно для использования операционной системой
- Глобальная страница (Global Page)
- Размер страницы 4Кб/2Мб (Page Size = 1 для страниц 2Мб)
- Модификация (Dirty)
- Доступ (Accessed)
- Запрещение кэширования (Cach disabled)
- Сквозная запись (Write-through)
- Пользователь/Супервизор (User/Supervisor)
- Чтение/Запись (Read/Write)
- Присутствие (Present)

■ Зарезервированные биты. Должны устанавливаться только в указанные значения.



# Практическое использование «больших» страниц

---

- Для пользовательского приложения – выделение виртуальной памяти с помощью вызова функции *VirtualAlloc ()* с флагом **MEM\_LARGE\_PAGE**.
- Для драйверов операционной системы – задать список драйверов в реестре (параметр HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\LargePageDrivers).



# Общие принципы управления виртуальной памятью в Win32

Ускорение страничных преобразований

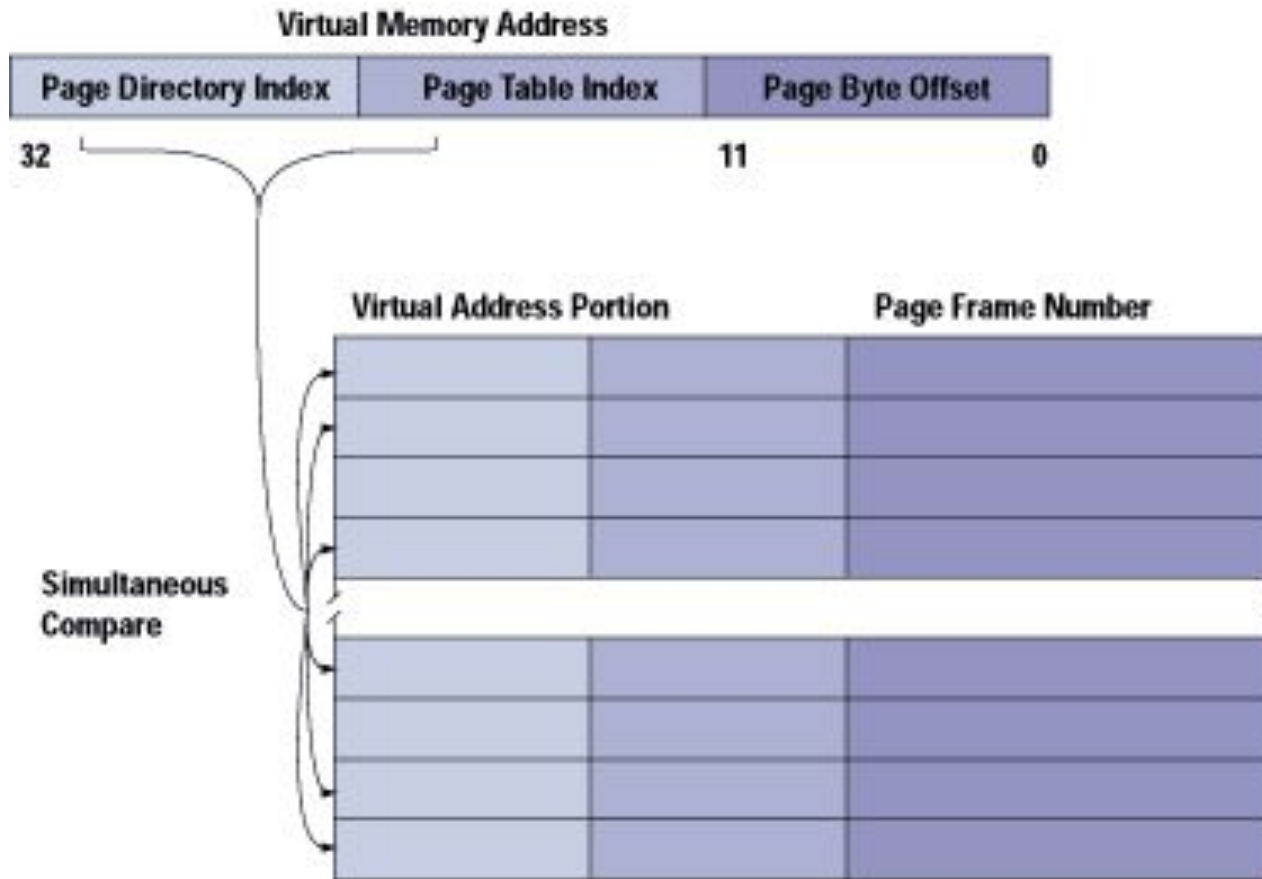
# Реализация TLB-кэша в Windows

---

- Часто используемым страницам (точнее их PTE) соответствуют элементы в TLB (Translation Lookaside Buffer), который обеспечивает быструю трансляцию виртуальных адресов в физические, а в результате и быстрый доступ к памяти.
- Если процесс обращается к странице, для которой нет записи в TLB, то для этой страницы создается элемент TLB.
- Если страница оказалась в страничном файле (бит «P» = 0) или если диспетчер памяти изменил его PTE, диспетчер памяти должен явно объявить соответствующий элемент TLB недействительным.



# Схема реализации TLB



# Эффективность использования больших страниц для TLB-кэша

---

- Недостатком «маленьких» страниц является неэффективное использование TLB, так для страниц размером 4 КБайт, механизм TLB содержит всего 32 записи в L1 кэше и 512 записей в L2 кэше. Так как каждая запись ссылается на 4 КБайт, то в сумме все записи «покрывают» чуть более 2 МБайт виртуальной памяти.
- В случае использования «больших» страниц TLB используется более эффективно. Для больших страниц TLB содержит восемь записей, так как каждая страница отображает 2 МБайт, то TLB может «покрывать» 16 МБайт виртуальной памяти.
- Использование «больших» страниц приводит к значительному увеличению производительности TLB-кэша и страничного преобразования в целом.



# Проблемы использования TLB

---

- При переключении процессов нужно добиться того, чтобы новый процесс не видел в ассоциативной памяти информацию, относящуюся к предыдущему процессу, например, выполнять ее очистку. Для очистки TLB отдельной страницы предназначена команда **INVTLB**.
  - В Windows очищаются все записи PTE, кроме тех, у которых установлен флаг Global. Для того чтобы объявить такую запись PTE недействительной, необходимо выполнить команду **INVLPG**.
  - TLB-кэши многопроцессорной системы аппаратно не синхронизируются, ядро операционной системы должно само выполнять действия по синхронизации их содержимого.
-

# Общие принципы управления виртуальной памятью в Win32

Стратегия управления виртуальной памятью и свопинг



# Стратегия управления виртуальной памятью

---

- ✓ Стратегия выборки (*fetch policy*)
- ✓ Стратегия размещения (*placement policy*)
- ✓ Стратегия замещения (*replacement policy*)



# Стратегия выборки

---

## ✓ Стратегия выборки (*fetch policy*):

- ✓ Выборка определяет, в какой момент необходимо переписать страницу с диска в ОП.
- ✓ В Windows используется классическая схема выборки с упреждением: система переписывает в память не только выбранную страницу, но и несколько следующих по принципу пространственной локальности, гласящему: наиболее вероятным является обращение к тем ячейкам памяти, которые находятся в непосредственной близости от ячейки, к которой производится обращение в настоящий момент. Поэтому вероятность того, что будут востребованы последовательные страницы, достаточно высока. Их упреждающая подкачка позволяет снизить накладные расходы, связанные с обработкой прерываний.

## ✓ Стратегия размещения (*placement policy*)

## ✓ Стратегия замещения (*replacement policy*)

---

# Стратегия размещения

---

- ✓ *Стратегия выборки (fetch policy)*
- ✓ *Стратегия размещения (placement policy):*
  - ✓ Размещение определяет, в какое место оперативной памяти необходимо поместить подгружаемую страницу.
  - ✓ Для систем со страничной организацией данная стратегия практически не имеет никакого значения, и поэтому Windows выбирает первую попавшуюся свободную страницу.
- ✓ *Стратегия замещения (replacement policy)*



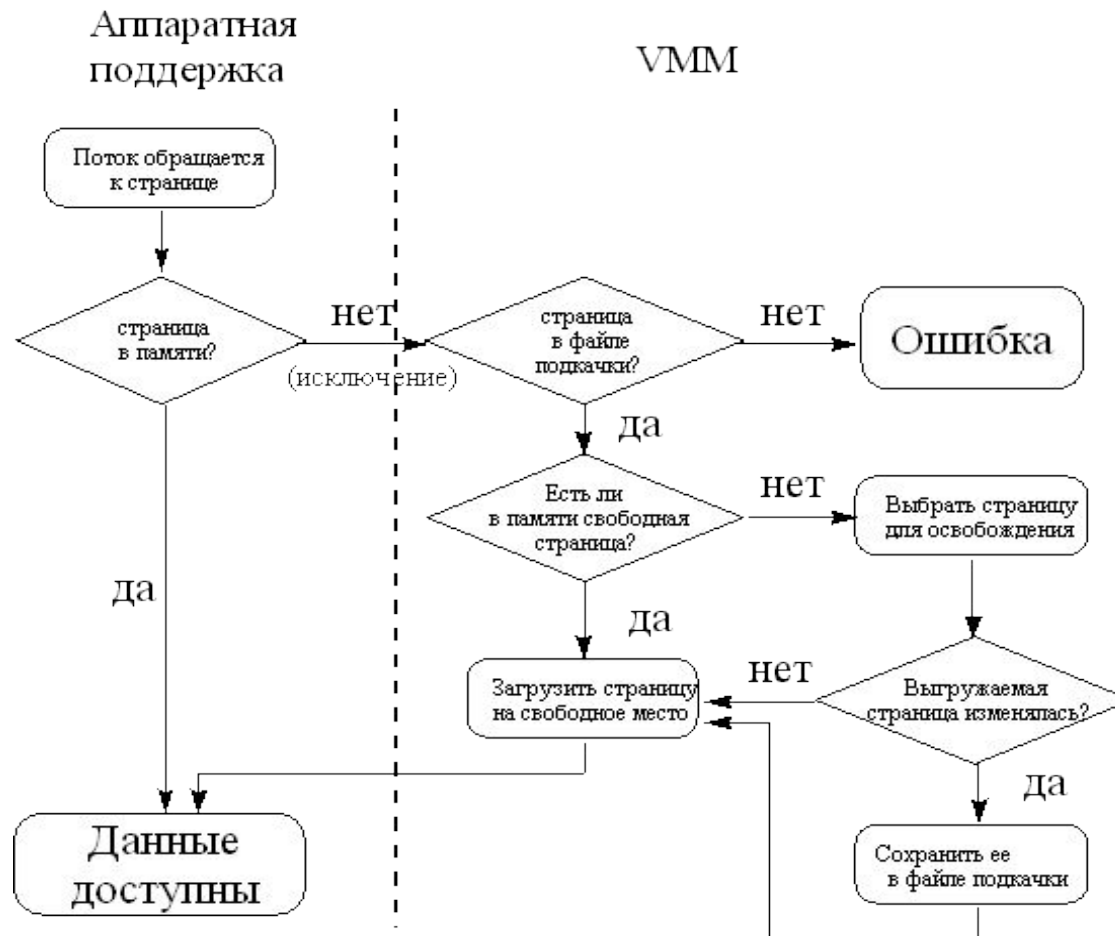
# Стратегия замещения

---

- ✓ *Стратегия выборки (fetch policy)*
- ✓ *Стратегия размещения (placement policy)*
- ✓ *Стратегия замещения (replacement policy):*
  - ✓ Замещение начинает действовать с того момента, когда в оперативной памяти компьютера не остается свободного места для размещения подгружаемой страницы. В этом случае необходимо решить, какую страницу вытеснить из физической памяти в файл подкачки (свопинг).



# Реализация стратегии управления виртуальной памятью



# СВОПИНГ

---

- Для того, чтобы обеспечить все линейное адресное пространство процесса физическими ячейками памяти, Windows применяет свопинг (замещение страниц).
- Организацией свопинга занимается менеджер виртуальной памяти.
- При генерации системы на диске образуется специальный файл свопинга (файл подкачки), куда записываются те страницы, которым не находится места в физической памяти.
- Менеджер виртуальной памяти использует программную реализацию локального алгоритма **LRU** (Least Recently Used) – замещение дольше всех неиспользовавшихся страниц.
- Локальный алгоритм **LRU** используется для предотвращения трэшинга.
- Программная реализация алгоритма **LRU** предполагает, что каждая из страниц виртуальной памяти в каждый момент времени может иметь одно из нескольких состояний, на основании информации о состоянии страниц менеджер

# Состояния страниц

---

- **Valid** или **Active** – страница используется процессом. Она реально существует в ОП и помечена в PTE как присутствующая в рабочем множестве процесса ( $P(V)=1, D=0,1$ ).
- **Standby** – содержимое страницы не изменялось ( $D=0$ ). В PTE страница помечена как отсутствующая ( $P(V)=0$ ) и переходная ( $T=1$ ).
- **Modified** – содержимое страницы было изменено ( $D=1$ ). В PTE страница помечена как отсутствующая ( $P(V)=0$ ) и переходная ( $T=1$ ).
- **Free** – страница, на которую не ссылается ни один PTE. Страница свободна, но подлежит обнулению, прежде чем будет использована.
- **Zeroed** – свободная и обнуленная страница, пригодная к непосредственному использованию любым процессом.
- **Bad** – страница, которая вызывает аппаратные ошибки и не может быть использована ни одним процессом.



# Формат PTE для страницы в оперативной памяти

Для процессоров с поддержкой PAT:

PAT-атрибут страницы (Page Attribute Table)



«P» – бит присутствия страницы в ОП

Свободно для использования операционной системой

Глобальная страница (Global Page)

Модификация (Dirty)

Доступ (Accessed)

Запрещение кэширования (Cach disabled)

Сквозная запись (Write-through)

Пользователь/Супервизор (User/Supervisor)

Чтение/Запись (Read/Write)

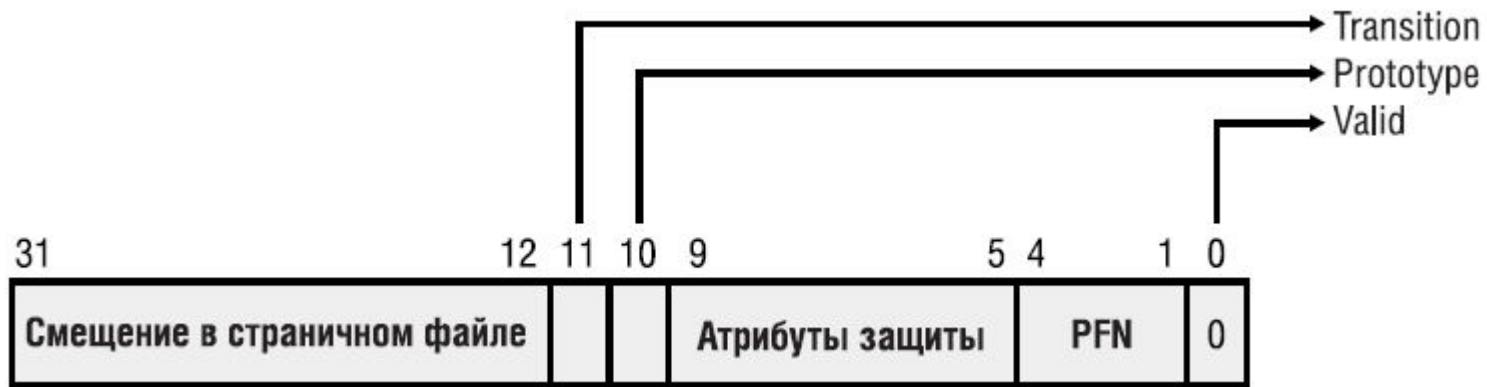
Присутствие (Present)

■ Резервированные биты. Должны устанавливаться только в указанные значения.

- Valid (Present) = 1 – страница присутствует в ОП

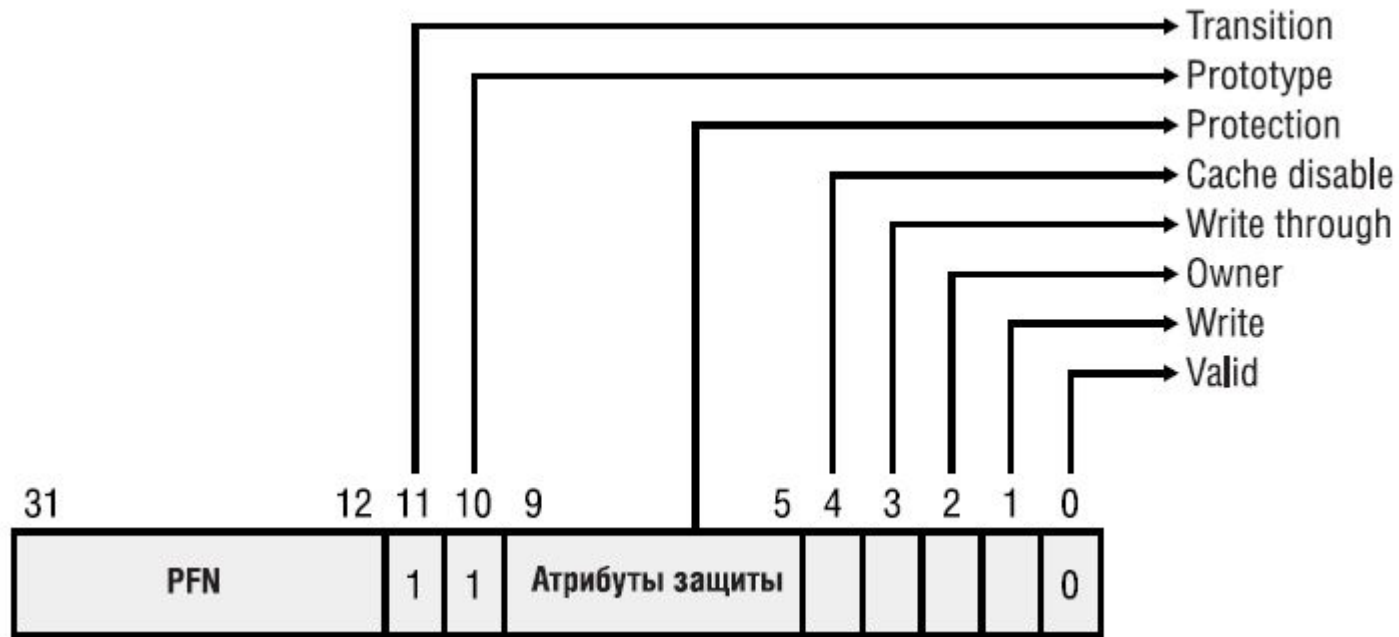


# Формат PTE для страницы в файле подкачки



- Valid (Present) = 0 – страница отсутствует в ОП
- Transition = 0

# Формат PTE для «похищенной» страницы



- Valid (Present) = 0 – страница отсутствует в ОП
- Transition = 1 – страница «похищена» менеджером виртуальной памяти

# Таблица состояний страниц

Transition	Dirty	Valid (Present)	Состояние страницы
-	0	1	Valid page
-	1	1	Valid dirty page
1	0	0	Standby
1	1	0	Modified
0	-	0	Free

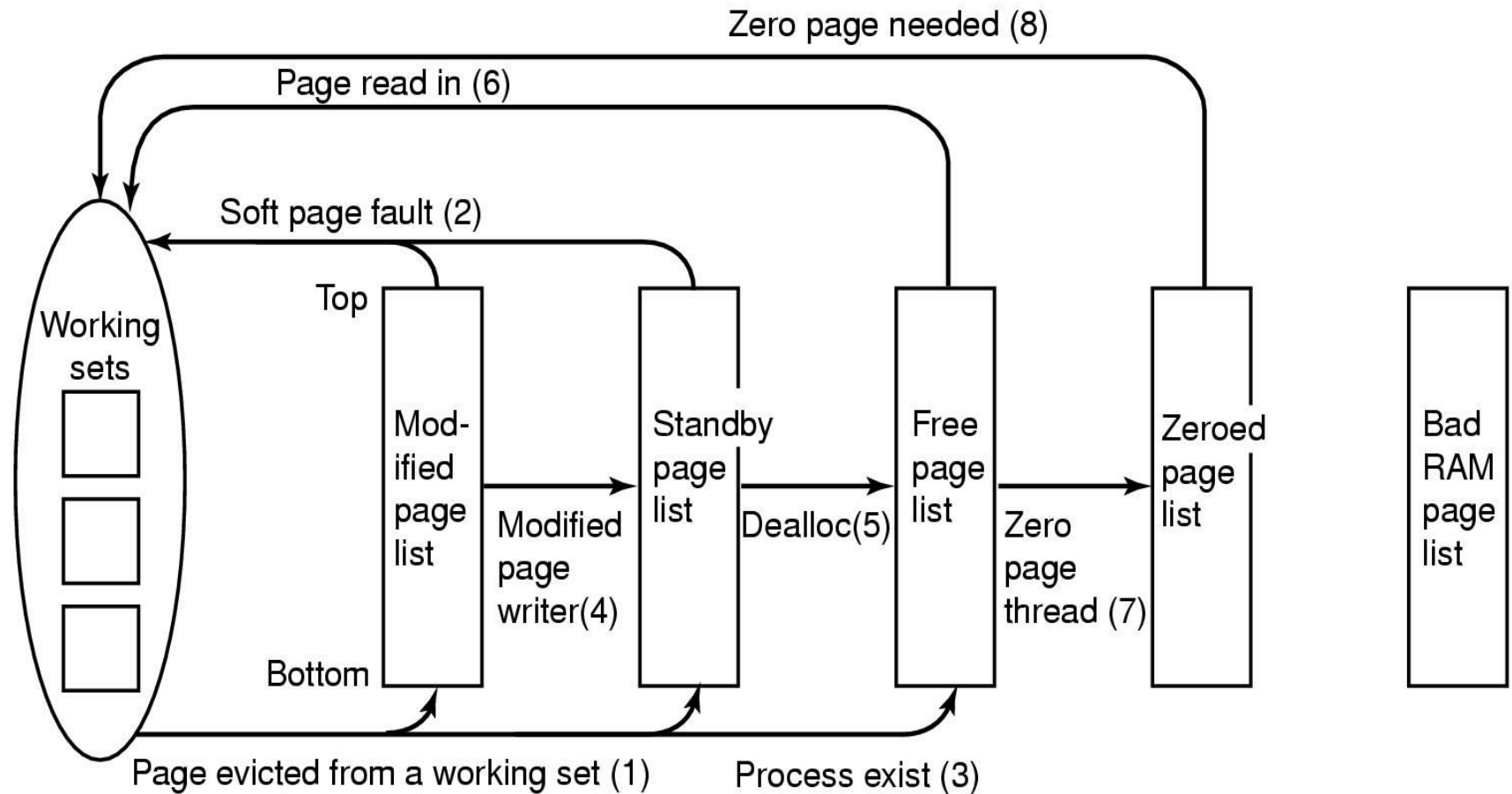


# Windows-реализация алгоритма замещения LRU

---

- ❑ VMM периодически просматривает список страниц с установленным флагом Present (Valid) и пытается похитить их у процесса **(1)**. Он помечает их как отсутствующие ( $P=0$ ), но на самом деле оставляет их в оперативной памяти, только переводит в список Modified или Standby в зависимости от значения бита D из PTE.
  - ❑ Если содержимое страницы была изменено в ОП ( $D=1$ ), то VMM выполнит запись страницы на диск **(4)**.
  - ❑ Если похищенная страница принадлежит рабочему множеству, то к ней в ближайшее время произойдет обращение. Так как страница помечена как отсутствующая, то обращение к ней вызовет страничное прерывание («soft» page fault). Но VMM очень быстро сделает эту страницу вновь доступной процессу, поскольку она находится в оперативной памяти **(2)**.
  - ❑ Далее если к странице не будет обращений (страница вне рабочего множества), то она со временем перейдет в состояние Free **(5)** и станет доступна для замещения страниц в рамках данного процесса **(6)**.
  - ❑ Затем системный поток обнуляет страницу – Zeroed **(7)**, и она станет доступна другим процессам системы **(8)**.
-

# Граф состояний страниц (1)



# Граф состояний страниц (2)

