МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет автоматизации и информатики
Кафедра автоматизированных систем управления

Индивидуальное домашнее задание

по дисциплине «Архитектура вычислительных систем»

Изучение механизма страничной адресации

Студент АС-21-1 _____ Станиславчук С. М.

(подпись, дата)

Руководитель

Ст. преподаватель _____ Болдырихин О. В.

(подпись, дата)

Липецк 2023

Цель работы

Выполнение индивидуального домашнего задания заключается в изучении и исследовании определенной подсистемы, узла или механизма вычислительной системы или в разработке вычислительной системы на основе данного процессора.

Задание кафедры

Изучить материал по выбранной теме: техническую документацию, периодическую, учебную и другую литературу.

Разработать программу для экспериментального исследования по выбранной теме.

Выбрать варьируемые параметры и задать их диапазон.

Например, при исследовании видеосистемы одним из параметров может быть видеорежим, который, в свою очередь, включает более мелкие параметры: тип режима (текстовый или графический), разрешение и т.п.

Определить, на что влияет и как изменение варьируемых параметров.

Сделать выводы по результатам исследования.

Ход выполнения индивидуального домашнего задания

Теоретическая информация

Страничная память — способ организации виртуальной памяти, при котором виртуальные адреса отображаются на физические постранично.

Решаемые задачи:

1. Поддержка изоляции процессов и защиты памяти путём создания своего собственного виртуального адресного пространства для каждого процесса;

2. Поддержка изоляции области ядра от кода пользовательского режима;

3. Поддержка памяти «только для чтения» и неисполняемой памяти;

4. Поддержка отгрузки давно неиспользуемых страниц в область подкачки на диске (см. свопинг);

5. Поддержка отображённых в память файлов, в том числе загрузочных модулей;

6. Поддержка разделяемой между процессами памяти, в том числе с копированием по записи для экономии физических страниц;

7. Поддержка системного вызова fork() в ОС семейства UNIX.

Концепции

Адрес, используемый в машинном коде, то есть значение указателя, называется «виртуальный адрес».

Адрес, выставляемый процессором на шину, называется «линейный адрес» (который позже преобразуется в физический).

Запись таблицы страниц обычно содержит в себе следующую информацию:

1. Флаг «страница отображена»;

2. Физический адрес;

3. Флаг «страница доступна из режима пользователя». При неустановке данного флага страница доступна только из режима ядра;

4. Флаг «страница доступна только на чтение». В некоторых случаях используется только для режима пользователя, то есть в режиме ядра все страницы всегда доступны на запись;

5. Флаг «страница недоступна на исполнение»;

6. Режим использования кэша для страницы. Влияет на тип шинных транзакций, инициируемых процессором при обращении через данную запись. Особенно часто используется для видеопамяти (комбинированная запись) и для отображенных в память регистров устройств (полное отсутствие кэширования).

Число записей в одной таблице ограничено и зависит от размера записи и размера страницы. Используется многоуровневая организация таблиц, часто 2 или 3 уровня, иногда 4 уровня (для 64-разрядных архитектур).

В случае 2-х уровней используется «каталог» страниц, в котором хранятся записи, указывающие на физические адреса таблиц страниц. В таблицах содержатся записи, указывающие на страницы данных.

При использовании 3-уровневой организации добавляется «надкаталог», хранящий записи, указывающие на несколько каталогов.

Старшие биты виртуального адреса указывают на номер записи в каталоге, средние — номер записи в таблице, младшие (адрес внутри страницы) попадают в физический адрес без трансляции.

Формат записей таблиц, их размер, размер страницы и организация таблиц зависит от типа процессора, а иногда и от режима его работы.

Исторически x86 использует 32-битные PTE, 32-битные виртуальные адреса, 4-килобайтные страницы, 1024 записи в таблице, двухуровневые таблицы. Старшие 10 битов виртуального адреса — номер записи в каталоге, следующие 10 — номер записи в таблице, младшие 12 — адрес внутри страницы.

Начиная с Pentium Pro, процессор поддерживает страницы размером 4 Мб. Однако, чтобы система и программы, запущенные в ней, могли

использовать страницы такого размера, технология 4-Мб страниц (huge pages) должна быть соответствующим образом активирована, а приложение настроено на использование страниц такого размера.

Процессор x86 в режиме PAE (Physical Address Extension) и в режиме x86-64 (long mode) использует 64-битные PTE (из них реально задействованы не все биты физического адреса, от 36 в PAE до 48 в некоторых x86-64), 32-битные виртуальные адреса, 4-Кб страницы, 512 записей в таблице, трёхуровневые таблицы с четырьмя каталогами и четырьмя записями в «надкаталоге». Старшие 2 бита виртуального адреса — номер записи в «надкаталоге», следующие 9 — в каталоге, следующие 9 — в таблице. Физический адрес каталога или же «надкаталога» загружен в один из управляющих регистров процессора.

При использовании PAE вместо 4-Мб больших страниц используются двухмегабайтные. См. также PSE.

В архитектуре x86-64 возможно использовать страницы размером 4 килобайта (4096 байтов), 2 мегабайта, и (в некоторых AMD64) 1 гигабайт.

Если обращение к памяти не может быть оттранслировано через TLB, то микрокод процессора обращается к таблицам страниц и пытается загрузить PTE оттуда в TLB. Если и после такой попытки сохранились проблемы, то процессор исполняет специальное прерывание, называемое «отказ страницы» (page fault). Обработчик этого прерывания находится в подсистеме виртуальной памяти ядра ОС.

Некоторые процессоры (MIPS) не имеют обращающегося к таблице микрокода, и генерируют отказ страницы сразу после неудачи поиска в TLB, обращение к таблице и её интерпретация возлагаются уже на обработчик отказа страницы. Это лишает таблицы страниц требования соответствовать жёстко заданному на уровне аппаратуры формату.

Причины отказа страницы (page fault):

1. Не существует таблицы, отображающей данный регион;

2. PTE не имеет взведённого флага «страница отображена»;

3. Попытка обратиться из пользовательского режима к странице «только для ядра»;

4. Попытка записи в страницу «только для чтения»;

5. Попытка исполнения кода из страницы «исполнение запрещено».

Обработчик отказов в ядре может загрузить нужную страницу из файла или же из области подкачки (см. свопинг), может создать доступную на запись копию страницы «только для чтения», а может и возбудить исключительную ситуацию (в терминах UNIX — сигнал SIGSEGV) в данном процессе.

Каждый процесс имеет свой собственный набор таблиц страниц. Регистр «каталог страниц» перегружается при каждом переключении контекста процесса. Также необходимо сбросить ту часть TLB, которая относится к данному процессу.

В большинстве случаев ядро ОС помещается в то же адресное пространство, что и процессы, для него резервируются верхние 1—2 гигабайта 32-битного адресного пространства каждого процесса. Это делается с целью избежать переключения таблиц страниц при входе в ядро и выходе из него. Страницы ядра помечаются как недоступные для кода режима пользователя.

Память региона ядра часто совершенно одинакова для всех процессов, однако некоторые подрегионы региона ядра (например, регион Windows, где находится подсистема графики и драйвер видео) могут быть различными для разных групп процессов (сессий).

Так как память ядра одинакова у всех процессов, соответствующие ей TLB не нужно перегружать после переключения процесса. Для этой оптимизации x86 поддерживает флаг «глобальный» у PTE.

Направление исследования

В данном ИДЗ исследуется пример реализации механизма страничной памяти.

Основные сведения о регистрах и флагах

Страничные записи

Каждый процесс обладает своим собственным набором страничных отображений. Таким образом, пространства виртуальной памяти каждого процесса независимы. В архитектуре x86 размер страниц фиксирован. Каждая страница имеет дескриптор, который содержит информацию о кадре, на который она отображается (размер дескриптора - 32 бита). Так как страницы и кадры должны быть выровнены по границе страницы (4 Кб = 4 * 0x1000 Б), последние 12 бит дескриптора всегда равны нулю. В эти биты помещается служебная информация. Дескриптор выглядит так:

31 12 11 9 8 7 6 5 4 3 2 1 0

|Frame address|AVAIL|RSVD|D|A|RSVD|U/S|R/W|P|

Поля, представленные на рисунке:

P установлен, если страница представлена в памяти;

R/W Если установлен, то страница доступна для записи. Игнорируется, если код исполняется в режиме ядра;

U/S Если установлен, то это страница уровня пользователя. В противном случае - уровня ядра. В режиме пользователя нельзя читать или записывать в страницы уровня ядра;

Reserved Зарезервировано;

A Установлен, если к странице уже обращались;

D Установлен, если к странице обращались для записи;

AVAIL Эти три бита не используются и доступны для использования ядром ОС;

Page frame address Старшие 20 бит адреса кадра в физической памяти.

Базовый физический адрес каталога страниц хранится в регистре CR3. И таблицы, и каталог выровнены на размер страницы.

Подготовим таблицы страниц и загрузим адрес каталога в CR3 в реальном режиме. Переход в защищённый режим мы выполним одновременно с включением страничной адресации (бит 31 регистра CR0).

Необходимо создать две таблицы страниц - первую (чтобы примонтировать первый мегабайт адрес-в-адрес) и последнюю (для ядра). Помимо проекции загруженного файла ядра, в этой таблице страниц будет ещё стек и самое главное - сама таблица.

Приложение А

Код программы с примером реализации механизма страничной адресации:

```
code segment
	assume cs:code
begin:
	; Очистим таблицы страниц
	xor ax, ax
	mov cx, 3 * 4096 / 2
	mov di, 0x1000
	rep stosw
	; Заполним каталог страниц
	mov word[0x1000], 0x2000 + 111b
	mov word[0x1FFC], 0x3000 + 111b
	; Заполним первую таблицу страниц
	mov eax, 11b
	mov cx, 0x100000 / 4096
	mov di, 0x2000
@b:
	stosd
	add eax, 0x1000
	loop @b
	; Заполним последнюю таблицу страниц
	mov di, 0x3000
	mov eax, dword[0x6000]
	or eax, 11b
	mov ecx, dword[0x6008]
	shr ecx, 12
@b1:
	stosd
	add eax, 0x1000
	loop @b1
	mov word[0x3FF4], 0x4000 + 11b ; Kernel stack
	mov word[0x3FF8], 0x3000 + 11b ; Kernel page table
	; Загрузим значение в CR3
	mov eax, 0x1000
	mov cr3, eax
	; Загрузим значение в GDTR
	lgdt [gdtr32]
	; Запретим прерывания
```

```
        cli
        ; Перейдём в защищённый режим
        mov eax, cr0
        or eax, 0x80000001
        mov cr0, eax
        ; Перейдём на 32-битный код
        jmp 8:start32
        ; Таблица дескрипторов сегментов для 32-битного ядра
        align 16
gdt32:
        dq 0 ; NULL - 0
        dq 0x00CF9A000000FFFF ; CODE - 8
        dq 0x00CF92000000FFFF ; DATA - 16
gdtr32:
        dw $ - gdt32 - 1
        dd gdt32
        ; 32-битный код
        use32
        start32:
        ; Настроим сегментные регистры и стек
        mov eax, 16
        mov ds, ax
        mov es, ax
        mov fs, ax
        mov gs, ax
        mov ss, ax
        mov esp, 0xFFFFDFFC
        ; Выводим символы на экран
        mov byte[0xB8000 + (25 * 80 - 1) * 2], "K"
        mov dword[0xFFFFEFFC], 0xB8000 + 11b
        mov byte[0xFFFFF000 + (25 * 80 - 2) * 2], "O"
code ends
end begin
```

Вывод

В ходе выполнения индивидуального домашнего задания была разработана программа для изучения и исследования механизма страничной памяти.