Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Архитектура вычислительных систем»

ОТЧЕТ
к лабораторной работе №2
на тему:
«ЗАЩИЩЕННЫЙ РЕЖИМ 32-РАЗРЯДНЫХ ПРОЦЕССОРОВ»
БГУИР 6-05-0612-02 67

Выполнил студент группы 353503
КОХАН Артём Игоревич

_____
(дата, подпись студента)

Проверил ассистент каф. информатики
КАЛИНОВСКАЯ Анастасия Александровна

_____
(дата, подпись преподавателя)

Минск 2025

1 ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Изучение особенностей защищенного режима 32-разрядных процессоров и разработка практических навыков по программированию процедур переключения процессора из реального режима в защищенный и обратно.

Написать программу, переключающую процессор в защищенный режим, выводящую на экране сообщение и затем возвращающую процессор в реальный режим (8 часов).

2 ВЫПОЛНЕНИЕ РАБОТЫ

Программа представляет собой загрузочный сектор (boot sector), который переключает процессор из реального режима (16-битного) в защищенный режим (32-битный) и демонстрирует основные принципы работы с защищенным режимом.

```
[BITS 16]
[ORG 0x7C00]

start:
    cli
    xor ax, ax
    mov ds, ax
    mov es, ax
    mov ss, ax
    mov sp, 0x7C00

    mov si, msg_real_mode
    call print_string_real_mode

    call wait_for_keypress

    ; Load GDT
    lgdt [gdt_descriptor]

    ; Set PE=1 in CR0
    mov eax, cr0
    or eax, 1
    mov cr0, eax

    jmp CODE_SEL:protected_mode_entry


; ------------------------------
; Protected Mode (32-bit)
; ------------------------------
[BITS 32]
protected_mode_entry:
    mov ax, DATA_SEL
    mov ds, ax
    mov es, ax
    mov fs, ax
    mov gs, ax
    mov ss, ax
    mov esp, 0x7C00

    ; clear screen
    mov edi, 0xB8000
    mov ax, 0x0720
    mov ecx, 80*25
```

```
    rep stosw

    ; Input str "Protected Mode"
    mov edi, 0xB8000
    mov esi, msg_protected_mode
    call print_string

    ; Check PE-bit in CR0
    mov eax, cr0
    test eax, 1
    jz no_pe
    mov esi, msg_pe
    call print_string
no_pe:

hang:
    hlt
    jmp hang


; ----------------------------
; real mode
; ----------------------------
[BITS 16]
print_string_real_mode:
    lodsb
    or al, al
    jz .done_real
    mov ah, 0x0E
    int 0x10
    jmp print_string_real_mode
.done_real:
    ret

wait_for_keypress:
    xor ax, ax
    int 0x16
    ret


; ----------------------------
; protected mode
; ----------------------------
[BITS 32]
print_string:
    lodsb
    test al, al
    jz .done_pm
    mov ah, 0x07
    stosw
    jmp print_string
.done_pm:
    ret
```

```
        ; ------------------------------
        ; message
        ; ------------------------------
        msg_real_mode db 'Real Mode: Press any key to enter Protected Mode...',
0
        msg_protected_mode db 'Protected Mode: Hello World! ', 0
        msg_pe db 'PE=1 (Protected Mode Active)', 0


        ; ------------------------------
        ; GDT
        ; ------------------------------
        align 8
        gdt:
            dq 0x0000000000000000
            dq 0x00CF9A000000FFFF
            dq 0x00CF92000000FFFF

        gdt_descriptor:
            dw gdt_end - gdt - 1
            dd gdt
        gdt_end:

        CODE_SEL equ 0x08
        DATA_SEL equ 0x10

        ; ------------------------------
        ; Boot sector padding
        ; ------------------------------
        times 510 - ($ - $$) db 0
        dw 0xAA55

        ; nasm -f bin -o main.bin main.asm
        ; qemu-system-x86_64 -drive format=raw,file=main.bin
```

При старте программы, процессор находится в реальном режиме. Это мы можем наблюдать на рисунке 1.

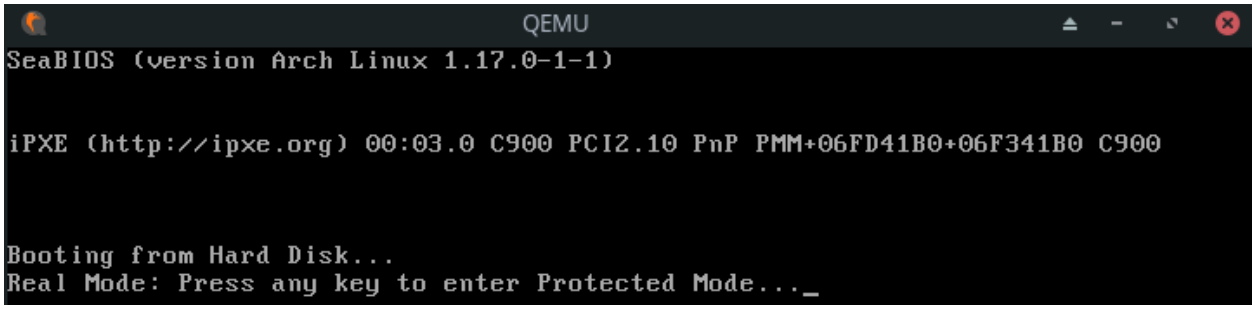

Рисунок 1 – Запуск программы и отображение реального режима

После нажатия на любую клавишу, увидим на рисунке 2 установку бита PE (Protection Enable) в регистре CR0 для активации защищенного режима.

Рисунок 2 – Отображение перехода в защищенный режим

ВЫВОД

В ходе выполнения работы был успешно реализован и протестирован механизм переключения процессора из реального режима в защищенный режим. Программа демонстрирует корректную инициализацию в реальном режиме с установкой сегментных регистров и стека, загрузку Global Descriptor Table (GDT) – фундаментального элемента защищенного режима, установку бита PE (Protection Enable) в регистре CR0 для активации защищенного режима с выводом советующего сообщения на экран, работу с сегментными селекторами в защищенном режиме, прямой доступ к видеопамяти (0xB8000) для вывода информации.