Министерство науки и высшего образования Российской Федерации
Мытищинский филиал
Федерального государственного автономного образовательного учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Космический

КАФЕДРА К3

отчет

ПО ЛАБОРАТОРНОЙ РАБОТЕ

№ 5

по ДИСЦИПЛИНЕ

«ОПЕРАЦИОННЫЕ СИСТЕМЫ»

Студент К3-66Б — В. Д. Чернов
(Группа) (И.О.Фамилия)

Преподаватель — А. В. Чернышов
(И.О.Фамилия)

2025 г.

Задание к лабораторной работе

Написать на ассемблере IBM PC программу простейшего планировщика, обеспечивающего параллельное выполнение нескольких процессов. В качестве процессов использовать циклически работающие процедуры, включённые в текст программы планировщика. Для демонстрации своей работы процессы должны выполнять какие-либо преобразования в видеопамяти с использованием прямого доступа к ней (не используя сервис прерываний). Конкретный вид преобразований студенты должны подобрать самостоятельно.

РЕШЕНИЯ ЗАДАНИЯ

```
.186
code segment
assume cs:code, ds:data
org 100h
start:
mov screen_pos, 0 ; начальная позиция экрана = 0
  mov ax, data
  mov ds, ax   ; ds = data
  ; Сохраняем и заменяем int 08h на наше
  cli ; clear interrupts – отключаем прерывания(сброс флага if)
mov ax, 0
  mov es, ax ; переходим в начало es
  mov ax, es:[20h]     ; offset для int 08h
  mov cs:[old_08h], ax  ; сохраняем оригинальный offset
mov ax, es:[22h] ; segment для int 08h
  mov cs:[old_08h+2], ax ; сохраняем оригинальный segment
  lea ax, handler ; ax = адрес метки handler
mov es:[20h], ax      ; теперь offset int 08h = handler
  mov ax, cs
  mov es:[22h], ax                ; теперь segment int 08h = cs
  sti         ; sti – set interrupt – включаем прерывания (установка флага if)

Pause:
mov bx, count ; задержка, чтобы прерывание сработало
  cmp bx, 04h
  jl Pause

  cli ; отключаем прерывания и восстанаваливаем оригинальное 08h
```

```
   mov ax, 0
   mov es, ax
   mov ax, cs:[old_08h]
mov es:[20h], ax
   mov ax, cs:[old_08h+2]
   mov es:[22h], ax
   sti
   mov ah, 4ch
   int 21h ; завершаем программу

handler  proc
push ax
    push bx

    push si
    push di
    push ds
    push es
   mov ax, cs:count
   cmp ax, 0
   jne Reset  ; if ax != 0 -> reset
   mov si, 0
   mov cs:[SSs], ss ; stack segment сохраняем
   mov cs:[SPs], sp ; stack pointer(offset) сохраняем
   mov ax, ds
   mov ss, ax ; ss = ds для взаимодействия с текущими данными
sti ; включаем прерывания

   mov sp, offset stack1+100h ; sp = верхушка стека stack1 (100h=256)
   pushf ; сохраняем текущее состояние флагов
```

```
  push cs ; сохраняем сегмент кода
  push offset proc1 ; адерс начала процедуры proc1
  mov cs:[SPs+2], sp ; сохраняем состояние sp для stack1
mov cs:[SSs+2], ss ; сохраняем состояние ss для stack1

  mov sp, offset stack2+100h
  pushf
  push cs
  push offset proc2
  mov cs:[SPs+4], sp
mov cs:[SSs+4], ss

mov sp, offset stack3+100h
  pushf
  push cs
  push offset proc3
  mov cs:[SPs+6], sp
mov cs:[SSs+6], ss
Reset:
  add si, 2
  cmp si, 8
  jne Switch ; if si != 8
  mov si, 2
Switch:
mov ax, cs:[SPs+si]
  mov sp, ax
mov ax, cs:[SSs+si]
mov ss, ax
mov ax, count ; счетчик вывода
inc ax
```

```
mov count, ax ; увеличиваем счетчик
cmp ax, 04h ; кол-во строк
  jl Exit
  mov ax, cs:[SSs]
  mov ss, ax
  mov ax, cs:[SPs]
  mov sp, ax
cli
Exit:

  jmp dword ptr cs:old_08h ; переход на оригинал

handler endp

proc1 proc
  mov bx, 0b800h
  mov es, bx
  mov di, screen_pos
  add di, 160*20 + 30 * 2
  mov al, '1'
  mov es:[di], al
  add screen_pos, 2
  cmp screen_pos, 20  ; кол-во символов в строке
  jl no_reset1
  mov screen_pos, 0
no_reset1:
  jmp proc1
proc1 endp

proc2 proc
```

```
    mov bx, 0b800h
    mov es, bx
    mov di, screen_pos
    add di, 160*21 + 30 * 2
    mov al, '2'
    mov es:[di], al
    add screen_pos, 2
    cmp screen_pos, 20
    jl no_reset2
    mov screen_pos, 0
no_reset2:
    jmp proc2
proc2 endp

proc3 proc
    mov bx, 0b800h
    mov es, bx
    mov di, screen_pos
    add di, 160*22 + 30 * 2
    mov al, '3'
    mov es:[di], al
    add screen_pos, 2
    cmp screen_pos, 20
    jl no_reset3
    mov screen_pos, 0
no_reset3:
    jmp proc3
proc3 endp

code ends
```

```
data segment
   count dw 0
screen_pos dw 0
   stack1 db 100h dup(0)
   stack2 db 100h dup(0)
stack3 db 100h dup(0)
   SSs dw 8h dup(0)
   SPs dw 8h dup(0)
   old_08h dw 0
data ends
end start
```

Результат работы