

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ
по лабораторной работе №4
по дисциплине «Организация ЭВМ и систем»
Тема: Представление и обработка символьной информации с
использованием строковых команд.

Студент гр. 0383

Куликов А. В.

Преподаватель

Ефремов М. А.

Санкт-Петербург

2021

Цель работы.

Написание собственного прерывания.

Задание.

Код задания 2D:

2 - 60h - прерывание пользователя - должно генерироваться в программе;

D - Выполнить чтение и вывод на экран отсчета системных часов (в тиках, где 1 тик = 55 мсек).

Выполнение работы.

Для хранения сегмента и смещения прерывания были созданы переменные KEEP_CS и KEEP_IP. Функция 35h прерывания 21h возвращает текущее значение вектора прерывания (в варианте лабораторной работы - 60h), и его смещение и сегмент заносятся в переменные KEEP_CS и KEEP_IP для дальнейшего восстановления. После этого с помощью функции 25h прерывания 21h устанавливается свое прерывание (процедура SUBR_INT) путем помещения смещения в DX, сегмента в DS. Потом с помощью функции 25h прерывания 21h восстанавливается старое прерывание.

Табл.1 Проверка работы программы

Входные данные	Результат работы программы	Примечание
—	244841	Верно
— (Запуск программы меньше чем через секунду)	244862	Верно
— (Запуск программы через, примерно, 2 секунды)	244951	Верно

Выводы.

В ходе выполнения данной лабораторной работы был изучен механизм написания собственного прерывания.

ПРИЛОЖЕНИЕ А

Тексты исходных файлов программ

lab5.asm

```
ASSUME CS:CODE, DS:DATA, SS:STACK
```

```
STACK  SEGMENT STACK
        DW 1024 DUP(?)
STACK  ENDS
```

```
DATA SEGMENT
    KEEP_CS DW 0
    KEEP_IP DW 0
    NUM DW 0
    MESSAGE DB 2 DUP(?)
```

```
DATA ENDS
```

```
CODE  SEGMENT
```

```
getInt PROC
```

```
    push dx
    push cx
```

```
    xor cx, cx
    mov bx, 10
```

```
L1:
    xor dx, dx
    div bx
    push dx
    inc cx
    test ax, ax
    jnz L1
```

```
    mov ah, 02h
```

```
L2:
    pop dx
    add dl, '0'
    int 21h
    loop L2
```

```
    pop cx
    pop dx
```

```
    ret
getInt ENDP
```

SUBR_INT PROC FAR

jmp start_proc

save_SP DW 0000h

save_SS DW 0000h

INT_STACK DB 40 DUP(0)

start_proc:

MOV save_SP, SP

MOV save_SS, SS

MOV SP, SEG INT_STACK

MOV SS, SP

MOV SP, offset start_proc

PUSH AX

PUSH CX

PUSH DX

mov AH, 00H

int 1AH

mov ax, cx

call getInt

mov ax, dx

call getInt

pop dx

pop cx

pop ax

mov ss, save_SS

mov sp, save_SP

mov al, 20H

out 20H, al

iret

SUBR_INT ENDP

Main PROC FAR

push DS

sub AX,AX

push AX

mov AX,DATA

mov DS,AX

MOV AH, 35H

MOV AL, 60H

INT 21H

MOV KEEP_IP, BX

MOV KEEP_CS, ES

```
PUSH DS
MOV DX, OFFSET SUBR_INT
MOV AX, SEG SUBR_INT
MOV DS, AX
MOV AH, 25H
MOV AL, 60H
INT 21H
POP DS
```

```
int 60H
```

```
CLI
PUSH DS
MOV DX, KEEP_IP
MOV AX, KEEP_CS
MOV DS, AX
MOV AH, 25H
MOV AL, 60H
INT 21H
POP DS
STI
```

```
RET
Main ENDP
CODE ENDS
END Main
```

ПРИЛОЖЕНИЕ Б

Тексты файлов диагностических сообщений программ

lab5.lst

Microsoft (R) Macro Assembler Version 5.10
Page 1-1

12/23/21 02:38:4

```
                                ASSUME CS:CODE, DS:DATA, SS:STACK

0000                                STACK  SEGMENT STACK
0000 0400[                        DW 1024 DUP(?)
    ????
    ]

0800                                STACK  ENDS

0000                                DATA SEGMENT
0000 0000                        KEEP_CS DW 0
0002 0000                        KEEP_IP DW 0
0004 0000                        NUM DW 0
0006 0002[                      MESSAGE DB 2 DUP(?)
    ??
    ]

0008                                DATA ENDS

0000                                CODE   SEGMENT

0000                                getInt PROC
0000 52                          push dx
0001 51                          push cx

0002 33 C9                      xor cx, cx
0004 BB 000A                    mov bx, 10

0007                                L1:
0007 33 D2                      xor dx, dx
0009 F7 F3                      div bx
000B 52                          push dx
000C 41                          inc cx
000D 85 C0                      test ax, ax
000F 75 F6                      jnz L1

0011 B4 02                      mov ah, 02h

0013                                L2:
0013 5A                          pop dx
```

0014 80 C2 30	add dl, '0'
0017 CD 21	int 21h
0019 E2 F8	loop L2
001B 59	pop cx
001C 5A	pop dx
001D C3	ret
001E	getInt ENDP

001E	SUBR_INT PROC FAR	
	Microsoft (R) Macro Assembler Version 5.10	12/23/21 02:38:4
	Page 1-2	

001E EB 2D 90	jmp start_proc
0021 0000	save_SP DW 0000h
0023 0000	save_SS DW 0000h
0025 0028[INT_STACK DB 40 DUP(0)
00	

]

004D	start_proc:
004D 2E: 89 26 0021 R	MOV save_SP, SP
0052 2E: 8C 16 0023 R	MOV save_SS, SS
0057 BC ---- R	MOV SP, SEG INT_STACK
005A 8E D4	MOV SS, SP
005C BC 004D R	MOV SP, offset start_proc
005F 50	PUSH AX
0060 51	PUSH CX
0061 52	PUSH DX

0062 B4 00	mov AH, 00H
0064 CD 1A	int 1AH

0066 8B C1	mov ax, cx
0068 E8 0000 R	call getInt
006B 8B C2	mov ax, dx
006D E8 0000 R	call getInt

0070 5A	pop dx
0071 59	pop cx
0072 58	pop ax
0073 2E: 8E 16 0023 R	mov ss, save_SS
0078 2E: 8B 26 0021 R	mov sp, save_SP
007D B0 20	mov al, 20H

007F E6 20	out 20H, al
------------	-------------

0081	CF	iret
0082		SUBR_INT ENDP
0082		Main PROC FAR
0082	1E	push DS
0083	2B C0	sub AX,AX
0085	50	push AX
0086	B8 ---- R	mov AX,DATA
0089	8E D8	mov DS,AX
008B	B4 35	MOV AH, 35H
008D	B0 60	MOV AL, 60H
008F	CD 21	INT 21H
0091	89 1E 0002 R	MOV KEEP_IP, BX
0095	8C 06 0000 R	MOV KEEP_CS, ES

Microsoft (R) Macro Assembler Version 5.10
Page 1-3

12/23/21 02:38:4

0099	1E	PUSH DS
009A	BA 001E R	MOV DX, OFFSET SUBR_INT
009D	B8 ---- R	MOV AX, SEG SUBR_INT
00A0	8E D8	MOV DS, AX
00A2	B4 25	MOV AH, 25H
00A4	B0 60	MOV AL, 60H
00A6	CD 21	INT 21H
00A8	1F	POP DS
00A9	CD 60	int 60H
00AB	FA	CLI
00AC	1E	PUSH DS
00AD	8B 16 0002 R	MOV DX, KEEP_IP
00B1	A1 0000 R	MOV AX, KEEP_CS
00B4	8E D8	MOV DS, AX
00B6	B4 25	MOV AH, 25H
00B8	B0 60	MOV AL, 60H
00BA	CD 21	INT 21H
00BC	1F	POP DS
00BD	FB	STI

00BE	CB	RET
00BF		Main ENDP
00BF		CODE ENDS
		END Main

Microsoft (R) Macro Assembler Version 5.10
Symbols-1

12/23/21 02:38:4

Segments and Groups:

N a m e	Length	Align	Combine	Class
CODE	00BF		PARA	NONE
DATA	0008		PARA	NONE
STACK	0800		PARA	STACK

Symbols:

N a m e	Type	Value	Attr
GETINT	N PROC	0000	CODE Length = 001E
INT_STACK	L BYTE	0025	CODE Length = 0028
KEEP_CS	L WORD	0000	DATA
KEEP_IP	L WORD	0002	DATA
L1	L NEAR	0007	CODE
L2	L NEAR	0013	CODE
MAIN	F PROC	0082	CODE Length = 003D
MESSAGE	L BYTE	0006	DATA Length = 0002
NUM	L WORD	0004	DATA
SAVE_SP	L WORD	0021	CODE
SAVE_SS	L WORD	0023	CODE
START_PROC	L NEAR	004D	CODE
SUBR_INT	F PROC	001E	CODE Length = 0064
@CPU	TEXT	0101h	
@FILENAME	TEXT	lab5	
@VERSION	TEXT	510	

125 Source Lines
125 Total Lines
21 Symbols

48020 + 461287 Bytes symbol space free

0 Warning Errors
0 Severe Errors