

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

ОТЧЕТ
по лабораторной работе №5
по дисциплине «Организация ЭВМ и систем»
Тема: Написание собственного прерывания.

Студент гр. 1381

Мелькумянц Д.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Ознакомиться с особенностями прерываний на языке Ассемблера, написать собственное прерывание.

Постановка задачи.

Написать прерывание `int 23h`, прерывание пользователя - должно генерироваться при нажатии `CTRL+C`. Печать сообщения на экране. Под стек отвести не менее 1Кб.

Выполнение работы.

В сегменте данных `DATA` содержится две переменных для хранения старого прерывания, содержавшегося по смещению `23h` - . Также в этом сегменте содержится `message` – сообщение, которое будет выводиться во время работы прерывания, `finaly` – сообщение, которое будет выведено после завершения работы прерывания.

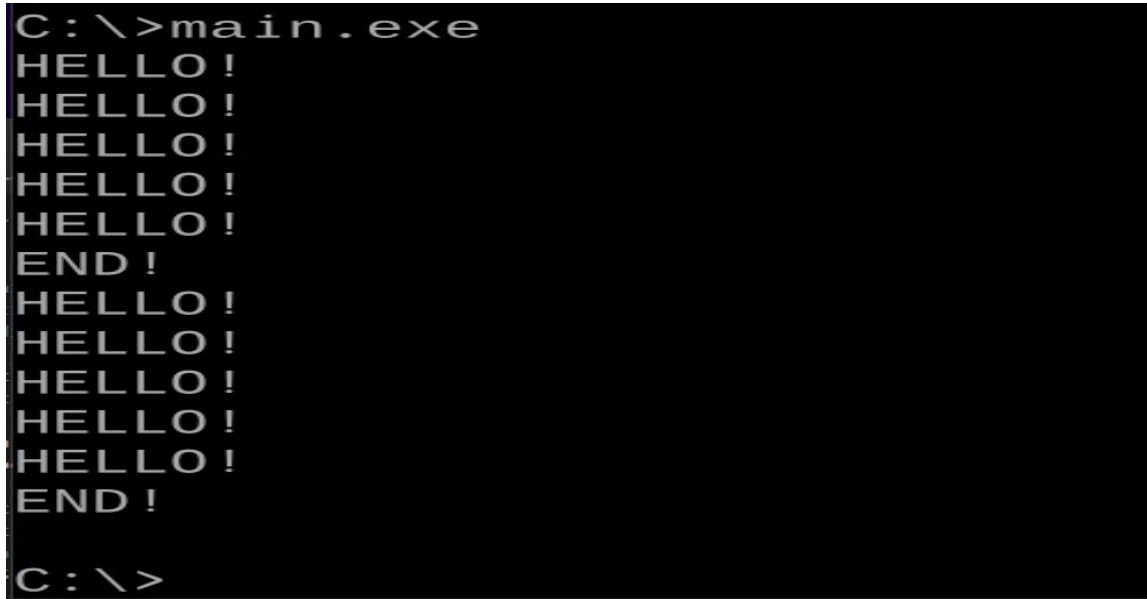
В сегменте `Astack`, как и требуется по заданию, выделяется 1Кбайт памяти, то есть `dw 512`.

В сегменте кода сначала определяем процедуру пользовательского прерывания `FUNC`. Сначала на стеке сохраняются значения регистров до входа в прерывания. С помощью метки `lp` строка из `dx` выводится заданное в `sx` количество раз. Далее реализована задержка после вывода строк с помощью прерывания `15h`. После чего выводится сообщение о завершение. Вызов прерывания происходит в процедуре `Main`. Для этого сначала с помощью прерывания, хранящееся по смещению `23h`. В переменных, указанных в сегменте данных, сохраняется старое прерывание. Новое прерывание `FUNC` записывается по смещению `23h`, также с помощью прерывания `21h`.

После вызова нового прерывания происходит восстановление старого прерывания и выход из программы.

Тестирование.

Работа программы с заданными условиями представлена на рисунке 1.



```
C:\>main.exe
HELLO!
HELLO!
HELLO!
HELLO!
HELLO!
END!
HELLO!
HELLO!
HELLO!
HELLO!
HELLO!
END!
C:\>
```

Рисунок 1 - Работа программы

Вывод.

В ходе выполнения лабораторной работы были изучены виды прерываний и работа с ними. В соответствии с заданием было создано собственное прерывание. Была написана программа, выводящая одно сообщение определенное количество раз, а другое - один раз с определенной задержкой при нажатии CTRL+C.

Приложение А. Код программы main.asm

AStack SEGMENT STACK

DB 512 DUP(?)

AStack ENDS

DATA SEGMENT

KEEP_CS DW 0

KEEP_IP DW 0

MESSAGE DB 'HELLO!', 0dh, 0ah, '\$'

FINALLY DB 'END!', 0dh, 0ah, '\$'

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA, SS:AStack

WriteMsg PROC NEAR

mov AH, 9

int 21h

ret

WriteMsg ENDP

FUNC PROC FAR

push ax

push cx

push dx

mov dx, OFFSET MESSAGE

mov cx, 5

lp:

call WriteMsg

loop lp

mov cx, 0033h

mov dx, 00FFh

mov ah, 86h

int 15h

mov dx, OFFSET FINALLY

call WriteMsg

pop dx

pop cx

pop ax

mov al, 20h

out 20h, al

iret

FUNC ENDP

MAIN PROC FAR

push ds

sub ax, ax

push ax

mov ax, DATA

mov ds, ax

mov ah, 35h

mov al, 23h

int 21h

mov KEEP_IP, bx

```
mov KEEP_CS, es
```

```
push ds
```

```
mov dx, OFFSET FUNC
```

```
mov ax, SEG FUNC
```

```
mov ds, ax
```

```
mov ah, 25h
```

```
mov al, 23h
```

```
int 21h
```

```
pop ds
```

```
begin:
```

```
    mov ah, 0
```

```
    int 16h
```

```
    cmp al, 3
```

```
    jnz begin
```

```
    int 23h
```

```
cli
```

```
push ds
```

```
mov dx, KEEP_IP
```

```
mov ax, KEEP_CS
```

```
mov ds, ax
```

```
mov ah, 25h
```

```
mov al, 23h
```

```
int 21h
```

```
pop ds
```

```
sti
```

```
mov ah, 4ch
```

```
int 21h
```

MAIN ENDP

CODE ENDS

END MAIN

Приложение Б. Листинг программы main.lst

#Microsoft (R) Macro Assembler Version 5.10

11/14/22 14:45:1

Page 1-1

0000 AStack SEGMENT STACK

0000 0200[DB 512 DUP(?)

??

]

0200 AStack ENDS

0000 DATA SEGMENT

0000 0000 KEEP_CS DW 0

0002 0000 KEEP_IP DW 0

0004 48 45 4C 4C 4F 21 MESSAGE DB 'HELLO!', 0dh,

0ah, '\$'

0D 0A 24

000D 45 4E 44 21 0D 0A FINALLY DB 'END!', 0dh, 0ah,

'\$'

24

0014 DATA ENDS

0000 CODE SEGMENT

ASSUME CS:CODE, DS:DATA,

SS:AStack

0000 WriteMsg PROC NEAR

0000 B4 09 mov AH, 9

0002	CD 21	int 21h
0004	C3	ret
0005		WriteMsg ENDP
0005		FUNC PROC FAR
0005	50	push ax
0006	51	push cx
0007	52	push dx
0008	BA 0004 R	mov dx, OFFSET
MESSAGE		
000B	B9 0005	mov cx, 5
000E		lp:
000E	E8 0000 R	call WriteMsg
0011	E2 FB	loop lp
0013	B9 0033	mov cx, 0033h
0016	BA 00FF	mov dx,
00FFh		
0019	B4 86	mov ah, 86h
001B	CD 15	int 15h
001D	BA 000D R	mov dx, OFFSET
FINALLY		
0020	E8 0000 R	call WriteMsg
0023	5A	pop dx
0024	59	pop cx
0025	58	pop ax
0026	B0 20	mov al, 20h

0028	E6 20	out 20h, al
002A	CF	iret
002B		FUNC ENDP

002B		MAIN PROC FAR
002B	1E	push ds

#Microsoft (R) Macro Assembler Version 5.10	11/14/22 14:45:1
---	------------------

Page 1-2

002C	2B C0	sub ax, ax
002E	50	push ax
002F	B8 ---- R	mov ax, DATA
0032	8E D8	mov ds, ax

0034	B4 35	mov ah, 35h
0036	B0 23	mov al, 23h
0038	CD 21	int 21h
003A	89 1E 0002 R	mov KEEP_IP, bx
003E	8C 06 0000 R	mov KEEP_CS, es

0042	1E	push ds
0043	BA 0005 R	mov dx, OFFSET FUNC
0046	B8 ---- R	mov ax, SEG FUNC
0049	8E D8	mov ds, ax
004B	B4 25	mov ah, 25h
004D	B0 23	mov al, 23h
004F	CD 21	int 21h
0051	1F	pop ds

0052	begin:
0052 B4 00	mov ah, 0
0054 CD 16	int 16h
0056 3C 03	cmp al, 3
0058 75 F8	jnz begin
005A CD 23	int 23h
005C FA	cli
005D 1E	push ds
005E 8B 16 0002 R	mov dx, KEEP_IP
0062 A1 0000 R	mov ax, KEEP_CS
0065 8E D8	mov ds, ax
0067 B4 25	mov ah, 25h
0069 B0 23	mov al, 23h
006B CD 21	int 21h
006D 1F	pop ds
006E FB	sti
006F B4 4C	mov ah, 4ch
0071 CD 21	int 21h
0073	MAIN ENDP
0073	CODE ENDS
END MAIN	

#Microsoft (R) Macro Assembler Version 5.10

11/14/22 14:45:1

Symbols-1

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK		0200	PARA	STACK
CODE		0073	PARA	NONE
DATA		0014	PARA	NONE

Symbols:

N a m e	Type	Value	Attr	
BEGIN		L NEAR 0052	CODE	
FINALLY		L BYTE 000D	DATA	
FUNC		F PROC 0005	CODE	Length = 0026
KEEP_CS		L WORD 0000	DATA	
KEEP_IP		L WORD 0002	DATA	
LP	L NEAR	000E	CODE	
MAIN	F PROC	002B	CODE	Length = 0048
MESSAGE	L BYTE	0004	DATA	
WRITEMSG	N PROC	0000	CODE	Length = 0005
@CPU	TEXT	0101h		
@FILENAME	TEXT	main		
@VERSION	TEXT	510		

91 Source Lines

91 Total Lines

17 Symbols

47932 + 434719 Bytes symbol space free

0 Warning Errors

0 Severe Errors