НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №1

з дисципліни **«**Периферійні пристрої**»**

Виконав:

студент 3 курсу

ФІОТ гр. ІО-21

Кузьменко Володимир

Перевірив:

Марковський О. П.

Київ – 2015 р.

Завдання

Встановити будильник реального часу на задане число секунд вперед по відношенню до поточного часу і по перериванню вивести своє прізвище.

Лістинг програми

.386

MODEL SMALL

linkDataWithRAM MACRO

MOV AX, DATA

MOV DS, AX

ENDM

outputString MACRO string

PUSH AX

PUSH DX

MOV AH, 09h

LEA DX, string

INT 21h

POP DX

POP AX

ENDM

readClockRegister MACRO reg

MOV AL, reg

OUT 70h, AL

IN AL, 71h

ENDM

writeClockRegister MACRO reg, op

MOV AL, reg

OUT 70h, AL

MOV AL, op

OUT 71h, AL

ENDM

outputSymbol MACRO sym

PUSH AX

PUSH DX

MOV AH, 02

MOV DL, sym

INT 21h

POP DX

POP AX

ENDM

outputBCDNumber MACRO reg

PUSH AX

PUSH DX

MOV BH, reg

MOV BL, reg

SHR BL, 4

MOV AH, 02

MOV DL, BL

ADD DL, '0'

INT 21h

AND BH, 00001111b

MOV AH, 02

MOV DL, BH

ADD DL, '0'

INT 21h

POP DX

POP AX

ENDM

outputCurrentTime MACRO

readClockRegister 4

outputBCDNumber AL

outputSymbol ':'

readClockRegister 2

outputBCDNumber AL

outputSymbol ":"

readClockRegister 0

outputBCDNumber AL

outputString eofl

ENDM

outputAlarmTime MACRO

readClockRegister 5

outputBCDNumber AL

outputSymbol ':'

readClockRegister 3

outputBCDNumber AL

outputSymbol ":"

readClockRegister 1

outputBCDNumber AL

outputString eofl

ENDM

; Read string to DX

inputString MACRO string

PUSH AX

PUSH DX

LEA EDX, string

MOV AH, 0Ah

INT 21h

outputString eofl

POP DX

POP AX

ENDM

; HEX number and result in [AL]

convert\_HEX\_to\_BCD MACRO

LOCAL l4, l6, l8

PUSH CX

PUSH BX

MOV CX, AX

XOR BX, BX

l4:

CMP AL, 100 ; if AX > 100

JB l6

SUB AL, 100

ADD BX, 100h ; add 100h to result

JMP l4

l6 :

CMP AL, 10 ; if AX > 10

JB L8

SUB AL, 10

ADD BX, 10h ; add 10h to result

JMP l6

l8 : XOR AH, AH

ADD BX, AX ; add remainder to result

MOV AX, BX

MOV AH, CH

POP BX

POP CX

ENDM

; read time in seconds to EAX

; max: 82800

inputTime MACRO

inputString buffer

; convert ASCII number to hex number

XOR DI, DI

XOR AX, AX

XOR CH, CH

XOR BX, BX

MOV CL, bufferLength

MOV SI, CX

MOV CL, 10

asc2hex:

MOV BL, BYTE PTR buf[DI]

CMP BL, 65h

JE finish

SUB BL, '0'

JB asc\_error

CMP BL, '9'

JA asc\_error

MUL ECX

XOR BH, BH

ADD EAX, EBX

INC DI

CMP DI, SI

JB asc2hex

ENDM

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

DATA SEGMENT use16

second DB 0

minute DB 0

hour DB 0

alarmSecond DB 0

alarmMinute DB 0

alarmHour DB 0

delayString DB 'Input length of interval in seconds', 0Dh, 0Ah, '$'

inputError DB 'Input error. Try again', 0Dh, 0Ah, '$'

currentTimeString DB 'Current time:', 0Dh, 0Ah, '$'

alarmTimeString DB 'Alarm''s time:', 0Dh, 0Ah, '$'

author DB 'Redko Alexander, IO-01', 0Dh, 0Ah, '$'

interval DD 0

buffer DB 50h

bufferLength DB 0

buf DB 254 dup (0)

eofl DB 0Dh, 0Ah, '$'

old\_handler DD 0

flag DB 0

DATA ENDS

CODE SEGMENT USE16

new\_4Ah PROC

outputString author

MOV CX, 15

bell: outputSymbol 07h

LOOP bell

MOV flag, 11b

; Restoring old alarm handler

LDS DX, old\_handler

MOV AX, 254Ah

INT 21h

IRET

new\_4Ah ENDP

begin: linkDataWithRAM

; Save old handler

MOV AX, 354Ah ; 4Ah – allarm’s interrupt

INT 21h

MOV WORD PTR old\_handler, BX

MOV WORD PTR old\_handler + 2, ES

linkDataWithRAM

; Set the new interrupt

MOV AX, 254Ah

MOV DX, seg new\_4Ah

MOV DS, DX

MOV DX, offset new\_4Ah

INT 21h

linkDataWithRAM

; Enable alarm's interrupt

readClockRegister 0Bh

MOV BL, AL

OR BL, 00100000b

writeClockRegister 0Bh, BL

outputString delayString

; Read time delay

input: inputTime

JMP saveInterval

asc\_error:

outputString inputError

JMP input

saveInterval: MOV interval, EAX

; Show and save current time

outputString currentTimeString

outputCurrentTime

readClockRegister 0

MOV second, AL

readClockRegister 2

MOV minute, AL

readClockRegister 4

MOV hour, AL

; parsing time

MOV EAX, interval

MOV ECX, DWORD PTR 3600

XOR EDX, EDX

DIV ECX

convert\_HEX\_to\_BCD

MOV alarmHour, AL

MOV AX, DX

MOV DL, WORD PTR 60

DIV DL

convert\_HEX\_to\_BCD

MOV alarmMinute, AL

MOV AL, AH

convert\_HEX\_to\_BCD

MOV alarmSecond, AL

; adding to current time entered time

MOV AL, second

ADD AL, alarmSecond

DAA

CMP AL, 60h

JB correction\_seconds

DEC SI

SUB AL, 60h

ADD alarmMinute, BYTE PTR 1h

correction\_seconds:

MOV alarmSecond, AL

MOV AL, minute

ADD AL, alarmMinute

DAA

CMP AL, 60h

JB correction\_minutes

SUB AL, 60h

ADD alarmHour, BYTE PTR 1h

correction\_minutes:

MOV alarmMinute, AL

MOV AL, hour

ADD AL, alarmHour

DAA

CMP AL, 24h

JB correction\_hours

SUB AL, 24h

correction\_hours:

MOV alarmHour, AL

MOV CH, alarmHour

writeClockRegister 5, CH

MOV CL, alarmMinute

writeClockRegister 3, CL

MOV DH, alarmSecond

writeClockRegister 1, DH

outputString alarmTimeString

outputAlarmTime

fin: MOV CL, 0

CMP CL, flag

JE fin

exit

CODE ENDS STACK 200h END begin