

```

;=====
;=====
;===== MIPS Domaci Jun 2006
;=====
;===== GLAVNI PROGRAM
;===== 8086_glavni_program.asm
;=====
;=====

RES_TAB SEGMENT AT 0FFFFH
    JMP Far ptr START
RES_TAB ENDS;

DAT SEGMENT AT 0000h
    IDIV0      dw 2 DUP(?)      ; deljenje nulom - ulaz 0
    ISS        dw 2 DUP(?)      ; single step - ulaz 1
    INMI       dw 2 DUP(?)      ; nemaskirajući prekid - ulaz 2
    IVT        dw 2*253 DUP(?)  ; ostalih 253 ulaza

;=====
; PROMENJIVE
;=====

    clock      dw 0              ;kad clock odbroji sa 1000 na 0 azuriramo sat
    vreme      db 00h,00h,80h,00h,80h,00h ; pocetno vreme - ( hh.mm.ss unazad )
    display     db 0              ;kada sa 2 padne na 0, osvezavamo cifru koja je na redu
    mode        db 0              ;mod rada ( 1-clock; 2-key_enter; 3-error )
    ktimeout    dw 0              ;preskaler za "key entering" timeout
    etimeout    dw 0              ;preskaler za "error" mod
    numkey      dw 0              ;broj unesenih tastera
    cif2s       db 0              ;broj cifre koja treba sledeca da se osvezi
    disp_mask   db 3Eh,3Dh,3Bh,37h,2Fh,1Fh ;paljenje s desna na levo
    keyboard    db '1','4','7','c','2','5','8','0','3','6','9','o'
    ID          db 0,0,0,0,0,0 ; trenutni uneseni tasteri
    adminID     db 'a','a','a','a','a','a' ; adminID
    admin       db 0              ; admin flag
    char2snd    db 0              ; karakter koji je na redu za slanje
    errornum    db 0              ; broj uzastopnih pogresnih logovanja

;=====
; adrese ulaza u IVT za prekidne rutine
;=====

    I_TIMER     EQU      (40h)*4 ; ulaz 64 u IVT
    I_USART     EQU      (41h)*4
    I_KEY       EQU      (42h)*4

;=====
; adrese na kojima se nalaze portovi
;=====

    A8251_Ctrl  EQU      0012h
    A8251_Data  EQU      0010h

    A8255_A     EQU      0050h
    A8255_B     EQU      0052h
    A8255_C     EQU      0054h
    A8255_Ctrl  EQU      0056h

    A8254_T0    EQU      0040h
    A8254_T1    EQU      0042h
    A8254_Ctrl  EQU      0046h

    A8259_A0    EQU      0030h
    A8259_A1    EQU      0032h

    DISP_OFF    EQU      FFh
    DERR        EQU      0Eh
    DKEY        EQU      0Ah
    EMPTY       EQU      0Fh
    GATEUP      EQU      00001111b ; portC.7 na 1
    GATEDOWN    EQU      00001110b ; portC.7 na 0
    mOK         EQU      FFh ; OK poruka od nadzornog racunara

```

```
mERROR      EQU      00h ; ERROR poruka od nadzornog racunara
mACK        EQU      0Fh ; ACK poruka za nadzorni racunar
mRESEND     EQU      F0h ; RESEND poruka za nadzorni racunar

EOI          EQU      20h ; specific EOI command

;vrh steka
ST_TOP      EQU      0400h
DAT ENDS
```

```
PROG SEGMENT AT 0FC00h
ASSUME CS: PROG, DS: DAT, ES:DAT, SS:DAT
```

```
;prekidna rutina za deljenje nulom
DIV0:
```

```
IRET
```

```
;prekidna rutina za NMI
```

```
NMI:
```

```
IRET
```

```
;=====
; program
;=====
```

```
START:
```

```
;inicijalizacija segmentnih registara i SP-a
```

```
XOR AX, AX
MOV DS, AX
MOV ES, AX
MOV SS, AX
MOV SP, ST_TOP
```

```
;=====
; inicijalizacija ulaza u IVT za prekidne rutine
;=====
```

```
; inicijalizacija za DIV0 prekidnu rutinu
```

```
MOV AX, offset DIV0
MOV IDIV0, AX
MOV AX, Seg DIV0
MOV IDIV0+2, AX
```

```
; inicijalizacija za NMI prekidnu rutinu
```

```
MOV AX, offset NMI
MOV INMI, AX
MOV AX, Seg NMI
MOV INMI+2, AX
```

```
; inicijalizacija za timer prekidnu rutinu
```

```
MOV AX, offset ir_tick
MOV DS:I_TIMER, AX
MOV AX, Seg ir_tick
MOV DS:I_TIMER+2, AX
```

```
; inicijalizacija za prekidnu rutinu za obradu prijema/slanja preko 8051
```

```
MOV AX, offset ir_usart
MOV DS:I_USART, AX
MOV AX, Seg ir_usart
MOV DS:I_USART+2, AX
```

```
; inicijalizacija za prekidnu rutinu za obradu pritiska tastera
```

```
MOV AX, offset ir_key
MOV DS:I_KEY, AX
MOV AX, Seg ir_key
MOV DS:I_KEY+2, AX
```

```
;=====
; inicijalizacija periferija
;=====
```

```

    CLI
    CALL Init8251
    CALL Init8254
    CALL Init8255
    CALL Init8259

; postavljanje pocetnih stanja

    MOV clock,3E8h
    MOV display,02h
    MOV mode,01h ; ukljuci mod za ispis trenutnog vremena
    MOV ktimeout,3A98H ; 3A98H = 15 000 => setovano na 15sekundi
    MOV etimeout,7530H ; 7530H = 30 000 => setovano na 30sekundi

    STI

MAIN:

    JMP MAIN

;=====
; Inicijalizacija za 8251
;=====

    Mode8251    EQU 01001110b    ; 1 stop, 8 bits, Baud Rate Faktor = 16x
    Comm8251    EQU 00110111b    ; RTS, err, RxEN, DTR, TxEN

Init8251 Proc Near
    MOV DX,A8251_Ctrl    ; worst-case inicijalizacija
    XOR AL,AL            ; da bi se osiguralo da je 8251
    OUT DX,AL            ; u command mod-u pre
    MOV CX,2            ; internog reseta,
delay1: LOOP delay1      ; prvo se tri puta šalje 0
    OUT DX,AL            ; zatim se šalje comanda za interni reset
    MOV CX,2
delay2: LOOP delay2      ; zbog Write-Recovery-Time,
    OUT DX,AL            ; prilikom slanja bilo koje komande
    MOV CX,2            ; treba sacekati 16 taktova signala CLK
delay3: LOOP delay3
    MOV AL,40h           ; reset naredba
    OUT DX,AL
    MOV CX,2
delay4: LOOP delay4
    MOV AL,Mode8251
    OUT DX,AL            ; salje se mode registar
    MOV CX,2
delay5: LOOP delay5
    MOV AL,Comm8251
    OUT DX,AL            ; salje se ctrl registar
    RET
Init8251 EndP

;=====
; Inicijalizacija za 8254
;=====

    T0CW        EQU 00010111b    ; T0, LSB only, mode=3, BCD
    T0DIVLSB     EQU 10h          ; 2457600 / (16*9600) = 16 = 10h

    T1CW        EQU 01110111b    ; T1, LSB then MSB, mode=3, BCD
    T1DIVLSB     EQU 58h          ; 2457600 / 1000 = 2458h
    T1DIVMSB     EQU 24h

Init8254 Proc Near
; Timer 0 se koristi kao generator takta za 8251
    MOV AL,T0CW
    OUT A8254_Ctrl,AL
    MOV AL,T0DIVLSB
    OUT A8254_T1,AL

; Timer 1 se koristi za merenje vremena (prekid na svakih 1ms)
    MOV AL,T0CW
    OUT A8254_Ctrl,AL

```

```

MOV AL, TODIVLSB
OUT A8254_T0, AL
MOV AL, TODIVLSB
OUT A8254_T0, AL

RET
Init8254 EndP

;=====
; Inicijalizacija za 8255
;=====

Ctrl8255 EQU 10000001b
; A - mod 0, A - output, C(upper) - output, B - mod 0, B - output, C(lower) - input

Init8255 Proc Near
MOV AL, Ctrl8255
OUT A8255_Ctrl,AL
MOV AL, 11111111b
OUT A8255_A, AL ; ugasi sve cifre
RET
Init8255 EndP

;=====
;Inicijalizacija za 8259
;=====

ICW1 EQU 00010011b ; edge triggered, single, ICW4
ICW2 EQU 01000000b ; interapti od ulaza 64 (40h)
ICW4 EQU 00000001b ; normal end-of-interrupt, 8086 režim
OCW1 EQU 11111000b ; demaskirani - clock, tastatura i USART
OCW2 EQU 00100000b ; nespecificna EOI komanda

Init8259 Proc Near
MOV AL, ICW1
OUT A8259_A0, AL ; slanje ICW1
MOV AL, ICW2
OUT A8259_A1, AL ; slanje ICW2
MOV AL, ICW4
OUT A8259_A1, AL ; slanje ICW4
MOV AL, OCW1
OUT A8259_A0, AL ; slanje OCW1
RET
Init8259 EndP

;=====
;=====
;=====
PROCEDURE
;=====
;=====

; Refresh - PROCEDURA ZA OSVEZAVANJE DISPLEJA
;
; ABSTRACT: Postoje 3 moda rada displeja - ispis trenutnog vremena, ispis znaka
; prilikom unosa tastera i ispis greske. Procedura ispituje u kojem je modu sistem i
; u zavisnosti od rezultata osvezava displej odgovarajucim podacima. Svake 2ms
; osvezava se sledeca cifra znaci jedna se osvezava svakih 12ms - tj. komotnih 83x /s
; Displej se osvezava standardnom softverskom tehnikom.
; REGISTERS: saves all registers and flags
;=====

Refresh Proc Near
PUSHF
PUSH AX
PUSH BX
PUSH DX
;ugasi displej
MOV DX, A8255_A
MOV AL, 11111111b

```

```

        OUT DX, AL
;u kojem modu je displej?
        MOV BL, cif2s
        MOV BH, 0
        MOV AL, mode
        CMP AL, 2
        JNZ RNO_KEY
; displej u KEY ENTER modu
        CMP BX, numkey
        JC RKEY2
; BX >= numkey - cifra prazna
        MOV DX, A8255_B
        MOV AL, 15 ; osvezi prazan displej
        OUT DX, AL
        MOV DX, A8255_A
        MOV AL, BYTE PTR disp_mask[BX]
        OUT DX, AL
        JMP ROVER
;BX < numkey - upisi izabrani znak u tu cifru
RKEY2:  MOV DX, A8255_B
        MOV AL, DKEY
        OUT DX, AL
        MOV DX, A8255_A
        MOV AL, BYTE PTR disp_mask[BX]
        OUT DX, AL
        JMP ROVER
RNO_KEY:
        CMP AL, 3
        JNZ RNO_ERROR
; displej je u ERROR modu
        MOV DX, A8255_B
        MOV AL, DERR
        OUT DX, AL
        MOV DX, A8255_A
        MOV AL, BYTE PTR disp_mask[BX]
        OUT DX, AL
        JMP ROVER
RNO_ERROR:
; displej je u CLOCK modu
; učitaj odgovarajući podatak
        MOV AL, BYTE PTR vreme[BX]
        MOV DX, A8255_B
        OUT DX, AL
; ukljuci displej
        MOV AL, BYTE PTR disp_mask[BX]
        MOV DX, A8255_A
        OUT DX, AL
; cifra osvezena
ROVER:  INC BX
        CMP BX, 5
        JNZ REXIT
        MOV BX, 0
REXIT:  POP DX
        POP BX
        POP AX
        POPF
        RET
Refresh EndP

```

```

;=====
; SetTime - PROCEDURA ZA HARDVERSKO PODESAVANJE TACNOG VREMENA
;
; ABSTRACT: postoje tri tastera za podesavanje sati, minuta i sekundi
; U zavisnosti koji od njih je pritisnut, procedura inkrementira odgovarajucu
; vrednost i ocuvava integritet
;
; REGISTERS: AL - kod tastera koji se pritisnuo
;             AL = 0Eh - podesavanje sati
;             AL = 0Dh - podesavanje minuta
;             AL = 0Ch - podesavanje sekundi
;
;=====

```

```

SetTime Proc Near
;iskljuci interapte - semaphore ( da ne smetaju dok se podesava vreme )
    CLI
    CMP AL, 0Eh
    JNZ SNEXT1
;podesavanje sati
    MOV AL, BYTE PTR vreme[5]
    CMP AL, 2
    JNZ SSAT1
    MOV AL, BYTE PTR vreme[4]
    INC AL
    CMP AL, 84h
    JNZ SSAT2
    MOV vreme[4], 80h
    MOV vreme[5], 0
    JMP SEXIT
SSAT2: MOV vreme[4], AL
    JMP SEXIT
SSAT1: MOV AL, BYTE PTR vreme[4]
    INC AL
    CMP AL, 84h
    JNZ SSAT3
    MOV vreme[4], 0
    INC vreme[5]
    JMP SEXIT
SSAT3: MOV vreme[4], AL
    JMP SEXIT
SNEXT1: CMP AL, 0Dh
    JNZ SNEXT2
;podesavanje minuta
    MOV AL, BYTE PTR vreme[2]
    INC AL
    CMP AL, 8Ah
    JNZ SMIN1
    MOV vreme[2], 0
    MOV AL, BYTE PTR vreme[3]
    INC AL
    CMP AL, 06h
    JNZ SMIN2
    MOV vreme[3], 0
    JMP SEXIT
SMIN1: MOV vreme[2], AL
    JMP SEXIT
SMIN2: MOV vreme[3], AL
    JMP SEXIT
SNEXT2:
;podesavanje sekundi
    MOV AL, BYTE PTR vreme[0]
    INC AL
    CMP AL, 0Ah
    JNZ SSEK1
    MOV vreme[0], 0
    MOV AL, BYTE PTR vreme[1]
    INC AL
    CMP AL, 06h
    JNZ SSEK2
    MOV vreme[1], 0
    JMP SEXIT
SSEK1: MOV vreme[0], AL
    JMP SEXIT
SSEK2: MOV vreme[1], AL
SEXIT: STI
    RET
SetTime EndP

;=====
; Time - PROCEDURA ZA ODRZAVANJE TACNOG VREMENA
;
; ABSTRACT: azurira tacno vreme obradjujuci svih 6 cifara sekvencijalno
;           pocevsi od cifre jedinica za sekunde
; REGISTERS: saves all registers and flags
;=====

Time Proc Near

```

```

        PUSHF          ; save registers and flags
        PUSH AX
        PUSH BX
        MOV BX,00H      ; 1. cifra sekundi
        MOV AL, BYTE PTR vreme[BX]
        INC AL
        CMP AL, 0AH
        JNZ CUPD
        MOV vreme[BX], 00H ; 2. cifra sekundi
        INC BX
        MOV AL, BYTE PTR vreme[BX]
        INC AL
        CMP AL,06H
        JNZ CUPD
        MOV vreme[BX],00H ; 1. cifra minuta
        INC BX
        MOV AL, BYTE PTR vreme[BX]
        INC AL
        CMP AL,8AH
        JNZ CUPD
        MOV vreme[BX],80H ; 2. cifra minuta
        INC BX
        MOV AL, BYTE PTR vreme[BX]
        INC AL
        CMP AL,06H
        JNZ CUPD
        MOV vreme[BX],00H ; 1. i 2. cifra sati
        MOV AL, vreme[BX+2] ; 2. cifra sati
        INC BX
        CMP AL, 02H
        JNZ CSATI1
        MOV AL, vreme[BX]
        INC AL
        CMP AL,84H
        JNZ CUPD          ; 24.00.00 -> 00.00.00
        MOV vreme[BX],80H
        MOV vreme[BX+1],00H
        JMP CEXIT
CSATI1: MOV AL, vreme[BX]
        INC AL
        CMP AL,8AH
        JNZ CUPD
        MOV vreme[BX], 80H
        INC vreme[BX+1]
        JMP CEXIT
CUPD:   MOV vreme[BX], AL
CEXIT:  POP BX
        POP AX
        POPF
        RET
Time EndP

```

```

;=====
; ir_tick - PROCEDURA ZA OBRADU lms PREKIDA
;
; ABSTRACT - svakih 1s poziva proceduru za azuriranje vremena, svakih 2ms poziva
;   proceduru za osvezavanje display-a; ako je sistem u "key enter" modu duze od
;   15s - vraca sistem u "clock" mod.
; REGISTERS - saves all flags; uses no registers
;=====

```

```

ir_tick Proc far
        PUSHF
        DEC clock
        JNZ NOTIME
;1s prosla -> azuriraj vreme
        MOV clock, 3E8h
        CALL Time
NOTIME: DEC display
        JNZ NODISP
;2ms prosle -> osvezi sledecu cifru
        MOV display, 2
        CALL Refresh
;da li je u "clock" modu?

```

```

NODISP: CMP mode,1
        JZ TEXTIT
;da li je u "key enter" modu?
        CMP mode,2
        JZ TCLOCK
;sistem je u error modu
        DEC etimeout
        JNZ TEXTIT
;error timeouted, prebaci sistem u "clock" mod
        MOV mode, 1
        MOV etimeout, 7530H
        JMP TEXTIT
;sistem je u key enter modu
TCLOCK: DEC ktimeout
        JNZ TEXTIT
;key enter mode has timeouted, vrati sistem u prvobitni rezim
;iskljuci admin mod, char2snd = 0, ponisti key bafer
;i prebaci sistem u CLOCK mod.
;ovaj timeout predstavlja i oporavak sistema od greske
;( npr. nemogucnost komuniciranja sa nadzornim racunarom)
        MOV numkey, 0
        MOV admin, 0
        MOV char2snd, 0
        MOV mode, 1
        MOV ktimeout, 3A98H
;end of interrupt
TEXTIT: MOV DX, A8259_A0
        MOV AL, EOI
        OUT DX, AL
        POPF
        IRET
ir_tick EndP

```

```

;=====
; ir_usart - PROCEDURA ZA OBRADU SERIJSKOG PRIJEMA/SLANJA
;
; ABSTRACT:
;   Ako je primljen podatak
;   - u slucaju da je doslo do greske pri slanju ili da sistem
;     prepozna komandu - salji nadzornom sistemu "resend" komandu
;   - ako sistem prepozna OK komandu - otvara vrata i salje racunaru "ACK"
;   - ako sistem prepozna ERROR komandu - u slucaju da je broj uzastopnih gresaka
;     veci od 3, prelazi u ERROR mod, inace prelazi u CLOCK mod
;   Ako nije primljen podatak, znaci da je TxRDY ( zbog samog protokola ne mogu istovremeno
;     biti postavljeni i TxRDY i RxRDY ) i ako postoji podatak za slanje, salje ga
;     nadzornom racunaru
;
; REGISTERS: saves all registers and flags
;
;=====

ir_usart Proc far
; save flags and registers
        PUSHF
        PUSH AX
        PUSH BX
        PUSH DX
; proveriti status
        MOV DX, A8251_Ctrl
        IN AL, DX
        AND AL, 00000010b
        JNZ UTMP
; ===== stigao podatak =====
        IN AL, DX
        AND AL, 01110000b
        JNZ UERROR
; citaj podatak
        MOV DX, A8251_Data
        IN AL, DX
        CMP AL, 15 ; mOK message
        JNZ UCMP2
; ===== stigla "OK" poruka od nadzornog racunara =====

```



```

; resetuj broj gresaka, otvori vrata, posalji "ACK", prebaci se u CLOCK mod
    MOV errornum, 0
    MOV DX, A8255_Ctrl
    MOV AL, GATEDOWN
    OUT DX, AL
    MOV AL, GATEUP
    OUT DX, AL
    MOV numkey, 0
UWAIT_TX:
; ako se desio timeout, izađi iz petlje
    CMP mode, 1
    JZ UWAIT_TX2 ; UEXIT bi trebalo
; da li je Tx spreman?
    MOV DX, A8251_Ctrl
    IN AL, DX
    AND AL, 10000001b
    JZ UWAIT_TX
; ready to send - send ACK
    MOV DX, A8251_Data
    MOV AL, mACK
    OUT DX, AL
    MOV mode, 1
    JMP UEXIT
UCMP2:  CMP AL, mERROR
        JNZ UERROR
; ===== stigla je ERROR poruka od nadzornog racunara =====
; posalji "ACK", povecaj broj uzastopnih pogresnih logovanja i
; ako je on veci od 3, ukljuci ERROR mod
    INC errornum
UWAIT_TX2:
; ako se desio timeout, izađi iz petlje
    CMP mode, 1
    JZ UEXIT
; da li je Tx spreman?
    MOV DX, A8251_Ctrl
    IN AL, DX
    AND AL, 10000001b
    JZ UWAIT_TX2
; ready to send - send ACK
    MOV DX, A8251_Data
    MOV AL, mACK
    OUT DX, AL
    CMP errornum, 4
    JC UERMD2
; errornum manji od 4
    MOV mode, 1
    JMP UEXIT
UERMD2:
; errornum veci od 3, ukljuci ERROR mod
    MOV mode, 3
    JMP UEXIT
UTMP:   JMP URDY2SND
UERROR:
; doslo je do greske prilikom prenosa ili je poruka ispravno primljena
; ali sistem nije prepoznao komandu - salji nadzornom racunaru RESEND poruku
UWAIT_TX3:
; ako se desio timeout, izađi iz petlje
    CMP mode, 1
    JZ UEXIT
; da li je Tx spreman?
    MOV DX, A8251_Ctrl
    IN AL, DX
    AND AL, 10000001b
    JZ UWAIT_TX3
; ready to send - send RESEND
    MOV DX, A8251_Data
    MOV AL, 11110000b ; resend komanda
    OUT DX, AL
    JMP UEXIT
URDY2SND:
; ===== INT je nastao zbog TxRDY uzlazne ivice =====
; da li ima nesto za slanje?
    CMP char2snd, 1

```

```

        JC UEXIT
; postoji podatak za slanje
        DEC char2snd
        MOV DX, A8251_Data
; salji id[5-char2snd]
        MOV BX, 5
        SUB BL, char2snd
        MOV AL, BYTE PTR id[BX]
        OUT DX, AL
        CMP char2snd, 0
        JNZ UEXIT
; ako je poslan poslednji karakter, isprazni bafer
        MOV numkey, 0
UEXIT:
; specific EOI
        MOV DX, A8259_A0
        MOV AL, EOI
        OUT DX, AL
; restore registers and flags
        POP DX
        POP BX
        POP AX
        POPF
        IRET
ir_usart EndP

;=====
; ir_key - PROCEDURA ZA OBRADU PRITISKA TASTERA
;
; ABSTRACT: Tastatura ima 16 tastera - 0..9, "C", "OK", 3 tastera za setovanje internog
; sata i taster za unos administratorskog IDa koji se ne salje nadzornom racunaru
; na proveru ( moze se koristiti u slucaju da je veza prekinuta ).
; Procedura ocita i dekoduje taster a zatim radi odgovarajucu radnju za svaki.
; "c" brise poslednji znak
; "ok" - ako je uneto 6 znakova - ako se nalazimo u admin rezimu, postavlja uneseni
; ID za adminID, ako ne - otvara vrata ako je ID jednak adminID-u u suprotnom salje
; ID na proveru nadzornom racunaru
; "(0..9)" - ako ima manje od 6 tastera, dodaje se izabrani taster u bafer
; "s" - povecava broj sekundi za 1
; "m" - povecava broj minuta za 1
; "h" - povecava broj sati za 1
; "p" - ukljucuje admin mod za unos adminID-a
;
; REGISTERS: Key Procedura je najnizeg prioriteta - ne moze da prekine niti jednu drugu
; zato nema potrebe da se registri cuvaju na steku
;
;=====

ir_key Proc far

; ako je u toku ERROR mod, sistem ne reaguje na tastere
        CMP mode, 3
        JZ KTMP1 ; redirekcija do KEXIT
; reset timeout logic
        MOV ktimeout, 3A98H
; read the key
        MOV DX, A8255_C
        IN AL, DX
        AND AL, 00001111b
        CMP AL, 0Ch
        JC KNO_SET
; ===== pritisnut je neki od PREFERENCES tastera =====
        CMP AL, 0Fh
        JNZ KSET_TIME
; pritisnut je taster za unos adminID-a
; prebaci se u admin rezim i ukljuci
; KEY ENTER mod
        MOV admin, 01
        MOV mode, 02
        JMP KEXIT
KSET_TIME:
; Podesi vreme
        CALL SetTime

```

```

        JMP KEXIT

KNO_SET:
; non-preferences tasteri
; predji u KEY ENTER mod
        MOV mode, 2
; dekoduj taster ( u ASCII kod ) i stavi ga u AL
        MOV BL, AL
        MOV AL, BYTE PTR keyboard[BX]
        CMP BL, 'c'
        JNZ KNO_C
; ===== pritisnut je taster 'c' =====
; ako je numkey = 0, ne radi nista, inace numkey--
        CMP numkey, 1
        JC KNO_C
; numkey > 0
        DEC numkey
KNO_C:  CMP AL, 'o'
        JNZ KNO_O
; ===== pritisnut je taster "ok" =====
; ako nema 6 unesenih cifara, "OK" ne radi nista
        CMP numkey, 6
        JNZ KNO_O
        CMP admin, 01
        JNZ KNO_PR
; sistem je u admin rezimu i uneseno je 6 cifara
; napravi novi adminID od unesenih cifara
        MOV CX, 6
        CLD
        LEA SI, ID
        LEA DI, adminID
        REP MOVSB
; ugasi admin, resetuj bafer i prebaci se u CLOCK mod
        MOV admin, 0
        MOV numkey, 0
        MOV mode, 1
KTMP1:  JMP KEXIT
KNO_PR:
; sistem nije u admin rezimu
; da li je uneseni ID = admin ID?
        MOV SI, OFFSET ID
        MOV DI, OFFSET adminID
        CLD
        MOV CX, 6
        REPE CMPSB
        JNZ KNO_ADM
; unesen je admin ID, otvori vrata odmah
; vrata se otvaraju uzlaznom ivicom
        MOV DX, A8255_Ctrl
        MOV AL, GATEDOWN
        OUT DX, AL
        MOV AL, GATEUP
        OUT DX, AL
        MOV numkey, 0
        MOV errornum, 0
        MOV mode, 1
        JMP KEXIT
KNO_ADM:
; salji ID nadzornom racunaru
KWAIT_TX:
; ako se desio timeout, izađi iz petlje
        CMP mode, 1
        JZ KEXIT
; da li je Tx spreman?
        MOV DX, A8251_Ctrl
        IN AL, DX
        AND AL, 10000001b
        JZ KWAIT_TX

; ready to send
        MOV DX, A8251_Data
; dovoljno je poslati samo 1. cifru, 8051 int rutina
; ce pokupiti ostalih 5
        MOV AL, BYTE PTR id[0]

```

```
        MOV char2snd, 05
        OUT DX, AL
        JMP KEXIT

KNO_O:
; ===== pritisnut je taster 0..9 =====
        CMP numkey, 6
        JNC KEXIT
        MOV BX, numkey
        MOV BYTE PTR id[BX], AL
        INC numkey

KEXIT:
; specific end of interupt
        MOV DX, A8259_A0
        MOV AL, EOI
        OUT DX, AL
        IRET
ir_key EndP
```

```
PROG ENDS
End START
```