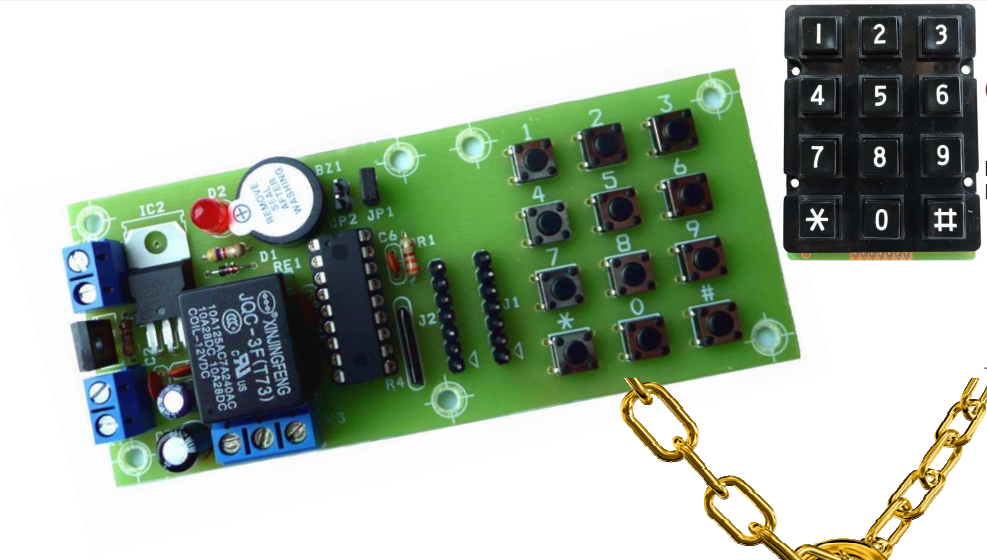
EPSICOM

Ready Prototyping

Colecția Home Automation

EP 0082



Cuprins

Prezentare Proiect
Fișa de Asamblare
1. Funcționare 2
2. Schema 2
3. PCB 3
4. Lista de componente 3
5. Soft 4 -12

Cod De Acces Cu PIC16F84

- Avantaj Pret/Calitate
- Livrare rapida
- Design Industrial
- Proiecte Modificabile
- Adaptabile cu alte module
- Module usor de asamblat
- Idei Interesante

Idei pentru afaceri

Hobby & Proiecte Educationale

www.epsicom.com/kits.php
a division of **EPSICO Manufacturing**

Din toate circuitelor de tip cifru electronic realizate cu PIC16F84-04 ce abundă pe diverse site-uri, acesta este aproape standard, cu posibilități de dezvoltare şi rapid de realizat.

Caracteristici:

- **Cod din 6 cifre**
- **Consum redus**
- **Posibilitate de expandare**

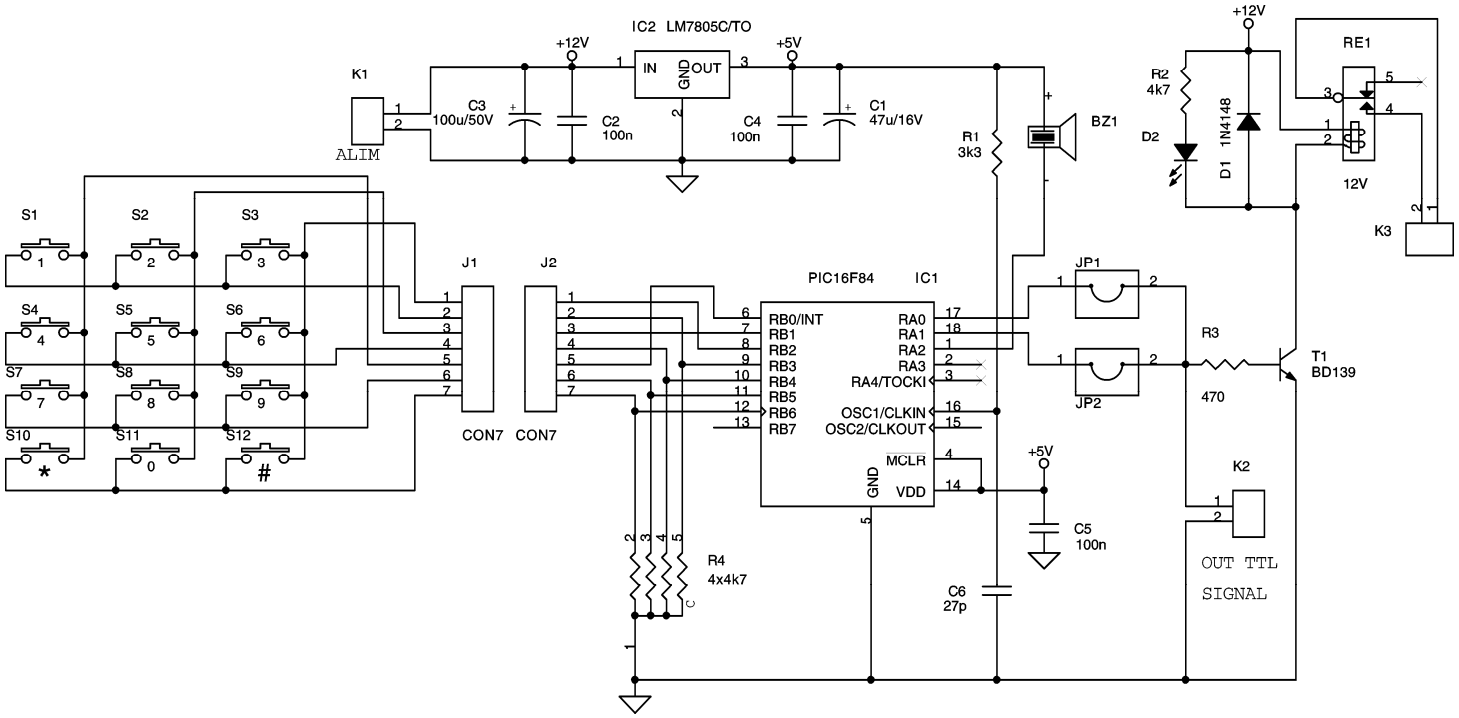
Funcționare

Circuitul conține µControllerul, o tastatură, un buzzer pentru confirmare apăsare, releul de acționare alarmă sau un electromagnet blocare-deblocare, sursa de 5V realizată cu LM7805 și un led pentru indicarea aclanșării releului. Rolul JP1 și JP2 este de a comanda tranzistorul continuu sau cu intermitențe (după o secundă de la introducerea codului).

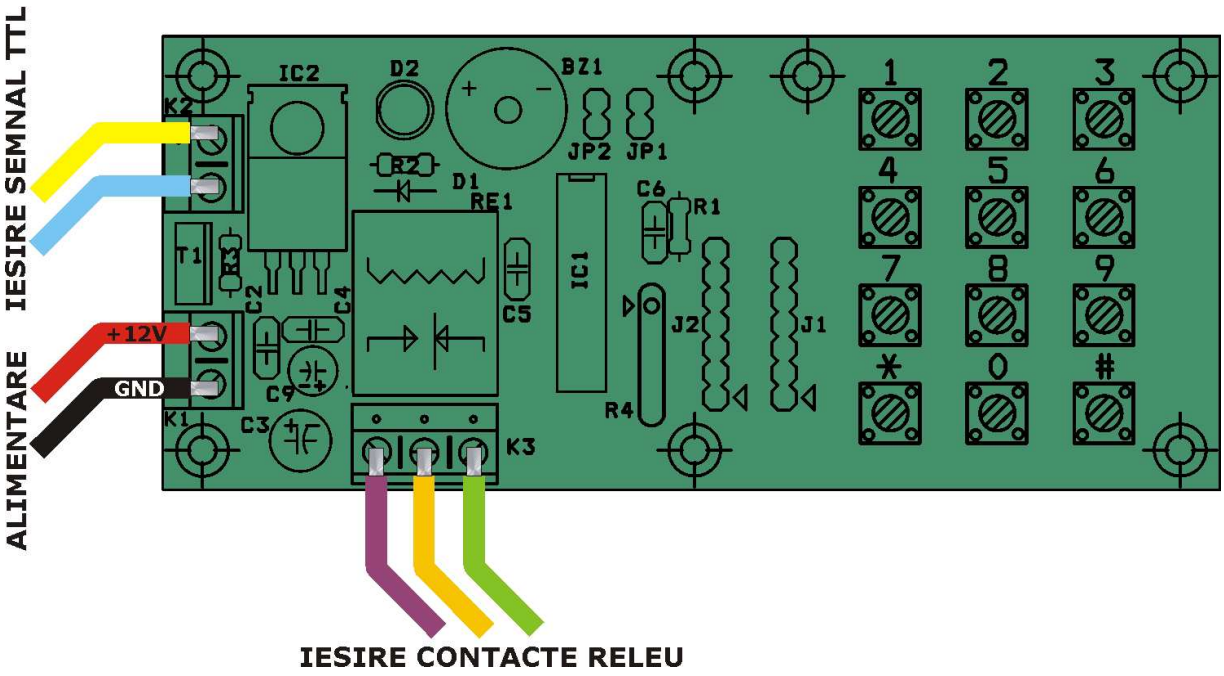
Ieșirea este inactivă la alimentare sau dacă circuitul a fost resetat. Codul se introduce prin apăsarea tastei * urmată de șase cifre. Dacă codul este corect buzzerul va confirma cu un beep. Dacă nu, va atenționa cu trei semnale. Dacă codul a fost introdus de 4 ori incorect ieșirile RA0 și RA1 vor trece în 5V iar alarma se va declanșa. Tastele devin inactive un minut. Codul inițial este de 123456. Schimbarea acestuia se face apăsând * urmat de # apoi codul original urmat de noul cod și încă odata noul cod de confirmare deci avem secvența:

*, # , Cod vechi (original) , Cod Nou, Cod Nou

Confirmarea de memorare a noului cod este dată printr-un beep, dacă a aparut o eroare suntem semnalizați cu 3 beepuri iar codul rămâne cel vechi. Prin apăsarea ** releul va fi deactivat. Ieșirea K2 TTL poate fi folosită ca și comandă într-un sistem mai complex. Alimentarea se face de la 12V însă poate varia între 8 și 15V. Consumul normal este de 12mA. Fișierul în .asm se livrează odata cu Kit-ul. Secțiunea tastaturii se poate detașa de modulul cu µcontroller iar conexiunile se fac prin circuit panglică.



Schema electrică



Amplasarea componentelor

Lista de componente

Nr.Crt.	Componenta	Denumire	Valoare	Cant
1	BZ1	Piezzo buzzer	Buzzer	1
2	C1	Condensator POL	47µF/16V	1
3	C2,C4,C5	Condensator NP	100nF	3
4	C3	Condensator POL	100µF/50V	1
5	C6	Condensator NP	27pF	1
6	D1	Diodă	1N4148	1
7	D2	Led	LED	1
8	IC1	C.I.	PIC 16F84	1
9	IC2	C.I.	LM7805	1
10	J1,J2	Conector	Con7	2
11	JP1,JP2	Jumper	JUMPER	2
12	K1,K2,K3	Conector	Con2	3
13	R1	Rezistență	3,3KΩ	1
14	R2	Rezistență	4,7KΩ	1
15	R3	Rezistență	470Ω	1
16	R4	Rezistență SIP	4x4,7KΩ	1
17	RE1	Releu	12V	1
18	S1…S12	Pushbutton	PUSH.	12
19	T1	Tranzistor	BD139	1

Acest produs se livrează în varianta circuit imprimat, circuit imprimat + componente sau în varianta asamblată în scopuri educaționale și va fi însoțit de documentația completă de asamblare pe CD.

```
; CODELOCK program

;==========================================
;    HARDWARE
;==========================================
; Keyboard connections
;
; RB0 = scanline 0
; RB1 = scanline 1
; RB2 = scanline 2
; RB3 = returnline 0
; RB4 = returnline 1
; RB5 = returnline 2
; RB6 = returnline 3
;(key = 3 x returnline (0,1,2,3) + scanline (0,1,2)
;
; RA2 = BEEPER active low
;
    ;RA3 = LED active low
;
; RA0 = level output
; RA1 = pulse output
;
;
;
;==========================================
; Configuration
;==========================================

RC      equ     b'11'           ;RC oscillator
XT      equ     b'01'           ;XT oscillator
WDTE    equ     b'100'          ;watchdog timer
PWRTE   equ     b'1000'         ;power up timer
CPOFF   equ     b'10000'        ;code protection

        PROCESSOR       16c84
        __CONFIG        RC+CPOFF+PWRTE

;==========================================
; Constants
;==========================================

;general

w       equ     0
f       equ     1
indf    equ     00h             ;indirect file
fsr     equ     04h             ;pointer to indirect file
ram     equ     0ch             ;rambottom (start)
status  equ     03h             ;status register file
porta   equ     05h             ;port A
portb   equ     06h             ;port B

intcon  equ     0bh             ;interrupt control register
rbif    equ     00h             ;rb has changed, interrupt event flag
rbie    equ     03h             ;rb interrupt enable flag
gie     equ     07h             ;global interrupt enable flag
inte    equ     04h             ;RB0/INT interrupt enable flag
intf    equ     01h             ;RB0/INT interrupt event flag
optreg  equ     81h             ;option register
intedg  equ     06h             ;INT edge

trisa   equ     85h             ;port A direction register
trisb   equ     86h             ;port B direction register

eecon1  equ     88h             ;eeprom control register
eecon2  equ     89h             ;eeprom data control register
eedata  equ     08h             ;eeprom data register
eeadr   equ     09h             ;eeprom adress register
eeif    equ     04h             ;
wrerr   equ     03h             ;
wren    equ     02h             ;
wr      equ     01h             ;
rd      equ     00h             ;

rp0     equ     05h             ;register bank
zero    equ     02h             ;zero bit position in status register
carry   equ     00h             ;carrybit position in status register
PC      equ     02h             ;program counter register

;==========================================
; macro definitions
;==========================================
jz      macro   lab1
```

```
        btfsc   status,zero
        goto    lab1
        endm

jnz     macro   lab2
        btfss   status,zero
        goto    lab2
        endm

jc      macro   lab3
        btfsc   status,carry
        goto    lab3
        endm

jnc     macro   lab4
        btfss   status,carry
        goto    lab4
        endm

;=========================================
; setbank
;=========================================
setbank macro   banknr
        if      banknr == 0
        bcf     status,rp0
        endif
        if      banknr == 1
        bsf     status,rp0
        endif
        endm

;=========================================
; portio
;
; sets tris of port
; usage: portio   porta,b'10010011'  ;example
; 0 is output, 1 is input
;=========================================
portio  macro   portnr,io
        clrf    portnr
        setbank 1
        movlw   io
        if      portnr == porta
        movwf   trisa
        endif
        if      portnr == portb
        movwf   trisb
        endif
        setbank 0
        endm


;=========================================
; program constants
;=========================================

nokey   equ     0ffh
hekje   equ     .12
sterretje       equ     .10

;=========================================
; variables
;=========================================
        org     ram

del0    equ     ram+0
del1    equ     ram+1
del10   equ     ram+2
scanln  equ     ram+3
retval  equ     ram+4
retln   equ     ram+5
key     equ     ram+6
scankey equ     ram+7
newkey  equ     ram+8
digit   equ     ram+9
match   equ     ram+0Ah

ncode0  equ     ram+0Bh
ncode1  equ     ram+0Ch
ncode2  equ     ram+0Dh
ncode3  equ     ram+0Eh
ncode4  equ     ram+0Fh
ncode5  equ     ram+10h

tries   equ     ram+11h
```

```
;last ram locatation = ram + 23h

;=========================================
; reset addres
;=========================================
	org	0
	goto	start

;=========================================
; identify interrupt source
;=========================================
	org	4
	goto	interrupt

;=========================================
; rdeeprom
; reads byte from eeprom
; entry EEADR, exit EEDATA
;=========================================
rdeeprom:
	bsf	status,rp0
	bsf	eecon1,rd
	bcf	status,rp0

	return


;=========================================
; wreeprom
;
; writes a byte of data to eeprom
;
; entry: eedata,eeadr
;=========================================
wreeprom:

	bcf	intcon,gie	;disable ints
	bsf	status,rp0	;bank 1
	bsf	eecon1,wren	;enable write

	;write sequence

	movlw	55h
	movwf	eecon2
	movlw	0aah
	movwf	eecon2
	bsf	eecon1,wr	;begin write

waitwr:
	btfsc	eecon1,wr
	goto	waitwr		;wait for writecycle complete

	bcf	status,rp0	;bank 0
	bsf	intcon,gie	;reenable ints

	return


;=========================================
; interrupt routines
;=========================================
interrupt:
	retfie

;=========================================
;
;=========================================
;=========================================
; Init
;=========================================
init:
	portio	porta,b'00000'
	portio	portb,b'11111000'

	bcf	porta,0		;clear level output
	bcf	porta,1		;clear pulse output

	bsf	porta,2		;beeper off

;	bsf	porta,3		;led off

	movlw	7h
	movwf	portb		;set keyboard scanlines
```

```
        movlw   1
        movwf   newkey

        clrf    tries

        ;test if signature is written in EEPROM
        ;if not then write signature and initial code
        ;else do nothing
        ;signature = AAh,55h,CCh, initial accescode = 1 2 3 4 5 6

        movlw   6
        movwf   eeadr           ;first signature adress
        call    rdeeprom
        movlw   0AAh
        subwf   eedata,w
        jnz     nosig

        incf    eeadr
        call    rdeeprom
        movlw   055h
        subwf   eedata,w
        jnz     nosig

        incf    eeadr
        call    rdeeprom
        movlw   0CCh
        subwf   eedata,w
        jnz     nosig

        ;signature match, do nothing
        return
nosig:
        ;no valid signature (first startup), write signature and accescode

        clrf    eeadr
        movlw   1
        movwf   eedata
wrcod:
        call    wreeprom
        incf    eedata
        incf    eeadr
        movlw   6
        subwf   eeadr,w
        jnz     wrcod

        ;code written, write signature

        movlw   0AAh
        movwf   eedata
        call    wreeprom
        incf    eeadr

        movlw   055h
        movwf   eedata
        call    wreeprom
        incf    eeadr

        movlw   0CCh
        movwf   eedata
        call    wreeprom

        return

;=========================================
; beep
; beeps 50mS
;=========================================
beep:
        bcf     porta,2
        movlw   .5
        movwf   del10
        call    delay10
        bsf     porta,2

        return

;=========================================
; delay10
; delays del10 x 10mS
;=========================================
delay10:
        ;delayloop 10 mS
        movlw   .40
        movwf   del1
dec250u:
```

```
        ;delayloop 250 uS
        movlw   .83
        movwf   del0
dec1u:
        decfsz  del0
        goto    dec1u

        decfsz  del1
        goto    dec250u

        decfsz  del10
        goto    delay10

        return
;=========================================
delay   macro   dly00
        movlw   dly00
        movwf   del10
        call    delay10
        endm

;=========================================
; scankeyboard
; scans the keyboard
; returns scankey = 1..12, 0ffh = nokey
; scankey is nokey if no key is pressed
; scanlines = RB0..RB2
; returnlines = RB3..RB6
;=========================================
scankeyboard:

        ;find scanline

        movlw   0
        movwf   portb
        clrf    scanln
        bsf     portb,0
        nop
        movf    portb,w
        andlw   b'01111000'
        jnz     retl
        incf    scanln
        bsf     portb,1
        nop
        movf    portb,w
        andlw   b'01111000'
        jnz     retl
        incf    scanln
        bsf     portb,2
        nop
        movf    portb,w
        andlw   b'01111000'
        jnz     retl
        goto    nokeyp          ;no keypress

        ;scanline found, find returnline
retl:
        clrf    retln
        movf    portb,w
        movwf   retval
        rrf     retval
        rrf     retval
        rrf     retval
nxtrl:
        bcf     status,carry
        rrf     retval
        jc      keyfound
        incf    retln
        movlw   4
        subwf   retln,w
        jnz     nxtrl

        ;no (new) key pressed, exit
nokeyp:
        movlw   nokey
        movwf   scankey
        return

keyfound:

        ;key found, calculate keynumber = 3x RL + SL + 1
        movf    retln,w
        bcf     status,carry
        rlf     retln           ;2x
        addwf   retln,f         ;3x
```

```
        movf    scanln,w
        addwf   retln,w
        movwf   scankey

        incf    scankey

        return

;=========================================
; getkey
; exits key:1..12,nokey
; debounced filtered output
;=========================================
getkey:
        call    scankeyboard
        movf    scankey,w
        movwf   key
        movlw   .2
        movwf   del10
        call    delay           ;delay 20mS
        call    scankeyboard
        movf    scankey,w
        subwf   key,w
        jnz     getkey

        ;debounce complete

        ;test if the key is a new keypress, if not return nokey=0ffh

        movlw   nokey
        subwf   key,w
        jnz     exitkey
        movlw   1
        movwf   newkey          ;next key is new
        goto    pkey

exitkey:
        movlw   1
        subwf   newkey,w
        jnz     pkey
        clrf    newkey
        return

pkey:   ;exit nokey
        movlw   nokey
        movwf   key
        return
;=========================================
; readkey
; reads keyboard until key pressed
; exits key = 1..12
;=========================================
readkey:
        call    getkey
        movlw   nokey
        subwf   key,w
        jz      readkey
        call    beep

        return

;=========================================
; main program
;=========================================
start:
        call    init
        call    beep

rdkey:
        call    readkey         ;wait for key

        ;key pressed

        movlw   sterretje
        subwf   key,w
        jnz     rdkey

        ; '*' pressed

nxkey0:
        call    readkey

        ;find out if second key is a number, '*' or '#'

        movlw   sterretje
```

```
        subwf   key,w
        jz      deactivate      ;'**' pressed, deactivate ouput, restart input

        movlw   hekje
        subwf   key,w
        jz      getcode         ;'*#' goto new code input

        ;'*number' pressed, read code, key holds first digit

        movlw   6
        movwf   digit           ;digit count, 6
        movlw   1
        movwf   match           ;code match to input flag
        clrf    eeadr           ;start with eeprom adress 0
rdee:
        call    rdeeprom
        movf    eedata,w
        subwf   key,w           ;compare stored codedigit to key
        jz      eq00
        clrf    match
eq00:
        decfsz  digit
        goto    nxtdg00

        ;6 digits compared
        btfss   match,0
        goto    nomatch0        ;no match, do nothing, return to input start

        ;codes match, set outputs
        clrf    tries
        bsf     porta,0         ;set level
;       bcf     porta,3         ;LED on


        ;pulse 1 sec
        ;beep 1 sec
        bcf     porta,2
        bsf     porta,1         ;set pulse
        delay   .100
        bcf     porta,1
        bsf     porta,2

        goto    rdkey

deactivate:
        bcf     porta,0         ;deactivate output
;       bsf     porta,3         ;LED off
        goto    nxkey0

nxtdg00:
        call    readkey         ;wait for key

        ;key pressed
        movlw   sterretje
        subwf   key,w
        jz      nxkey0          ;restart input

        movlw   hekje
        subwf   key,w
        jz      rdkey           ;restart input

        ;digit pressed
        incf    eeadr
        goto    rdee

nomatch0:
        ;wrong accescode entry
        incf    tries
        movlw   4
        subwf   tries,w
        jnz     nomatch

        ;four times mismatch in a row
        ;alarm for 1 minute
        movlw   .60
        movwf   digit

alarm:
        call    beep            ;50mS
        delay   .15             ;150ms
        call    beep            ;50
        delay   .15             ;150
        call    beep            ;50
        delay   .55             ;550mS
```

```
        decfsz  digit
        goto    alarm
        goto    rdkey

nomatch:
        ;beep 3x
        delay   .30
        call    beep
        delay   .10
        call    beep
        delay   .10
        call    beep

        goto    rdkey

getcode:
        ;'*#' pressed, get new accescode
        ;format = '*#ooooonnnnnmmmmm'
        ;o = oldcode,n = newcode, m = newcode verification

        ;read and verify the old code
        movlw   1
        movwf   match
        movlw   6
        movwf   digit
        clrf    eeadr

        ;read digit and verify
gtdg00:
        call    readkey         ;wait for key
        movlw   sterretje
        subwf   key,w
        jz      nxkey0          ;'*' pressed, restart input

        movlw   hekje
        subwf   key,w
        jz      nomatch         ;error in input

        ;digit pressed, compare to eeprom

        call    rdeeprom
        movf    key,w
        subwf   eedata
        jz      match00
        clrf    match           ;mismatch in input
match00:
        incf    eeadr
        decfsz  digit
        goto    gtdg00          ;read next


        ;read and save new accescode

        movlw   6
        movwf   digit
        movlw   ncode0
        movwf   fsr             ;point to first digit

rdnc00:
        call    readkey         ;wait for key
        movlw   sterretje
        subwf   key,w
        jz      nxkey0          ;'*' pressed, restart input

        movlw   hekje
        subwf   key,w
        jz      nomatch         ;error in input

        movf    key,w
        movwf   indf

        incf    fsr
        decfsz  digit
        goto    rdnc00

        ;new code saved
        ;read verification of newcode and compare

        movlw   6
        movwf   digit
        movlw   ncode0
        movwf   fsr             ;point to saved code

rdvc00:
        call    readkey         ;wait for key
```

```
        movlw   sterretje
        subwf   key,w
        jz      nxkey0          ;'*' pressed, restart input

        movlw   hekje
        subwf   key,w
        jz      nomatch         ;error in input

        movf    indf,w
        subwf   key,w
        jz      mok
        clrf    match           ;key and saved digit differ, no match

mok:
        incf    fsr
        decfsz  digit
        goto    rdvc00

        ;new code verified, test if matched

        btfss   match,0
        goto    nomatch         ;input mismatch, error signal, restart input

        ;codes match, store new code in eeprom

        ;long beep
        bcf     porta,2
        delay   .100
        bsf     porta,2

        movlw   6
        movwf   digit
        movlw   ncode0
        movwf   fsr
        clrf    eeadr

store00:
        movf    indf,w
        movwf   eedata
        call    wreeprom

        incf    fsr
        incf    eeadr
        decfsz  digit
        goto    store00

        goto    rdkey           ;new code stored in eeprom, restart input

;=========================================
; eeprom definition
;=========================================
        org     2100h

        de      0,0,0,0,0,0,0,0



        end
```