Maciej Byczko Bartosz Matysiak	Prowadzący: dr inż. Jacek Mazurkiewicz	Numer ćwiczenia 4
Cz 13:15 TN	Temat ćwiczenia: Klawiatura	Ocena:
Grupa: B	Data wykonania: 8 Kwietnia 2021	

1 Zadanie 1

1.1 Polecenie

Przygotować program umożliwiający wybór i przypisanie do klawiszy klawiatury - guzików od 0 do 9, A, B, C - jednego z trzech zestawów znaków, które będą wypisywane na wyświetlaczu LCD gdy klawisze klawiatury będą naciskane. Przełączenie zestawów: guzik * - małe litery od a do m, guzik # - duże litery do A do M, guzik D - cyfry od 0 do 9 oraz znaki A, B, C.

1.2 Rozwiązanie

```
1  ljmp start
2
3  P5 equ 0F8H
4  P7 equ 0DBH
5
6  LCDstatus  equ 0FF2EH        ; adres do odczytu gotowosci LCD
7  LCDcontrol equ 0FF2CH        ; adres do podania bajtu sterujacego LCD
8  LCDdataWR  equ 0FF2DH        ; adres do podania kodu ASCII na LCD
9
10 // bajty sterujace LCD, inne dostepne w opisie LCD na stronie WWW
11 #define  HOME     0x80      // put cursor to second line
12 #define  INITDISP 0x38      // LCD init (8-bit mode)
13 #define  HOM2     0xc0      // put cursor to second line
14 #define  LCDON    0x0e      // LCD nn, cursor off, blinking off
15 #define  CLEAR    0x01      // LCD display clear
16
17 // linie klawiatury - sterowanie na port P5
18 #define LINE_1    0x7f  // 0111 1111
19 #define LINE_2    0xbf  // 1011 1111
20 #define LINE_3    0xdf  // 1101 1111
21 #define LINE_4    0xef  // 1110 1111
22 #define ALL_LINES 0x0f  // 0000 1111
23
24 org 0100H
25
26 // macro do wprowadzenia bajtu sterujacego na LCD
27 LCDcntrlWR MACRO x           ; x - parametr wywolania macra - bajt
      sterujacy
28            LOCAL loop        ; LOCAL oznacza ze etykieta loop moze
                 sie powtórzyc w programie
```

```
29 loop: MOV  DPTR,#LCDstatus  ; DPTR zaladowany adresem statusu
30       MOVX A,@DPTR          ; pobranie bajtu z biezacym statusem LCD
31       JB   ACC.7,loop       ; testowanie najstarszego bitu
            akumulatora
32                             ; - wskazuje gotowosc LCD
33       MOV  DPTR,#LCDcontrol ; DPTR zaladowany adresem do podania
            bajtu sterujacego
34       MOV  A, x             ; do akumulatora trafia argument
            wywolania -macrabajt sterujacy
35       MOVX @DPTR,A          ; bajt sterujacy podany do LCD - zadana
            akcja widoczna na LCD
36       ENDM
37
38 // macro do wypisania znaku ASCII na LCD, znak ASCII przed
      wywolaniem macra ma byc w A
39 LCDcharWR MACRO
40       LOCAL tutu            ; LOCAL oznacza ze etykieta tutu moze
            sie powtórzyc w programie
41       PUSH ACC              ; odlozenie biezacej zawartosci
            akumulatora na stos
42 tutu: MOV  DPTR,#LCDstatus  ; DPTR zaladowany adresem statusu
43       MOVX A,@DPTR          ; pobranie bajtu z biezacym statusem LCD
44       JB   ACC.7,tutu       ; testowanie najstarszego bitu
            akumulatora
45                             ; - wskazuje gotowosc LCD
46       MOV  DPTR,#LCDdataWR  ; DPTR zaladowany adresem do podania
            bajtu sterujacego
47       POP  ACC              ; w akumulatorze ponownie kod ASCII
            znaku na LCD
48       MOVX @DPTR,A          ; kod ASCII podany do LCD - znak widoczny
             na LCD
49       ENDM
50
51 // macro do inicjalizacji wyswietlacza - bez parametrów
52 init_LCD MACRO
53          LCDcntrlWR #INITDISP ; wywolanie macra LCDcntrlWR -
               inicjalizacja LCD
54          LCDcntrlWR #CLEAR    ; wywolanie macra LCDcntrlWR -
               czyszczenie LCD
55          LCDcntrlWR #LCDON    ; wywolanie macra LCDcntrlWR -
               konfiguracja kursora
56          ENDM
57
58 // funkcja opóznienia
59
60   delay:  mov r1, #0FFH
61   dwa:    mov r2, #0FFH
62     trzy: djnz r2, trzy
63       djnz r1, dwa
64       ret
65
```

```
66  // funkcja wypisania znaku
67  putcharLCD: LCDcharWR
68          ret
69
70  // tablica przekodowania klawisze - ASCII w XRAM
71
72  keyascii:
73          ;znaki dla nacisniecia *
74          ; Uklad klawiatury:
75          /*
76              a b c d
77              e f g h
78              i j k l
79               m
80          */
81
82          mov dptr, #80EBH
83          mov a, #"m"
84          movx @dptr, a
85
86          mov dptr, #8077H
87          mov a, #"a"
88          movx @dptr, a
89
90          mov dptr, #807BH
91          mov a, #"b"
92          movx @dptr, a
93
94          mov dptr, #807DH
95          mov a, #"c"
96          movx @dptr, a
97
98          mov dptr, #80B7H
99          mov a, #"e"
100         movx @dptr, a
101
102         mov dptr, #80BBH
103         mov a, #"f"
104         movx @dptr, a
105
106         mov dptr, #80BDH
107         mov a, #"g"
108         movx @dptr, a
109
110         mov dptr, #80D7H
111         mov a, #"i"
112         movx @dptr, a
113
114         mov dptr, #80DBH
115         mov a, #"j"
116         movx @dptr, a
```

```
117
118     mov dptr, #80DDH
119     mov a, #"k"
120     movx @dptr, a
121
122     mov dptr, #807EH
123     mov a, #"d"
124     movx @dptr, a
125
126     mov dptr, #80BEH
127     mov a, #"h"
128     movx @dptr, a
129
130     mov dptr, #80DEH
131     mov a, #"l"
132     movx @dptr, a
133
134     /*
135     mov dptr, #80EEH
136     mov a, #"D"
137     movx @dptr, a
138
139     mov dptr, #80E7H
140     mov a, #"*"
141     movx @dptr, a
142
143     mov dptr, #80EDH
144     mov a, #"#"
145     movx @dptr, a
146     */
147
148     ; znaki dla nacisniecia #
149     ; Uklad klawiatury:
150     /*
151         A B C D
152         E F G H
153         I J K L
154           M
155     */
156     mov dptr, #81EBH
157     mov a, #"M"
158     movx @dptr, a
159
160     mov dptr, #8177H
161     mov a, #"A"
162     movx @dptr, a
163
164     mov dptr, #817BH
165     mov a, #"B"
166     movx @dptr, a
167
```

```
168         mov dptr, #817DH
169         mov a, #"C"
170         movx @dptr, a
171 
172         mov dptr, #81B7H
173         mov a, #"E"
174         movx @dptr, a
175 
176         mov dptr, #81BBH
177         mov a, #"F"
178         movx @dptr, a
179 
180         mov dptr, #81BDH
181         mov a, #"G"
182         movx @dptr, a
183 
184         mov dptr, #81D7H
185         mov a, #"I"
186         movx @dptr, a
187 
188         mov dptr, #81DBH
189         mov a, #"J"
190         movx @dptr, a
191 
192         mov dptr, #81DDH
193         mov a, #"K"
194         movx @dptr, a
195 
196         mov dptr, #817EH
197         mov a, #"D"
198         movx @dptr, a
199 
200         mov dptr, #81BEH
201         mov a, #"H"
202         movx @dptr, a
203 
204         mov dptr, #81DEH
205         mov a, #"L"
206         movx @dptr, a
207 
208         ; znaki dla nacisniecia D
209         ; Uklad klawiatury:
210         /*
211              1 2 3 A
212              4 5 6 B
213              7 8 9 C
214                0
215         */
216 
217         mov dptr, #82EBH
218         mov a, #"0"
```

```
219         movx @dptr, a
220 
221         mov dptr, #8277H
222         mov a, #"1"
223         movx @dptr, a
224 
225         mov dptr, #827BH
226         mov a, #"2"
227         movx @dptr, a
228 
229         mov dptr, #827DH
230         mov a, #"3"
231         movx @dptr, a
232 
233         mov dptr, #82B7H
234         mov a, #"4"
235         movx @dptr, a
236 
237         mov dptr, #82BBH
238         mov a, #"5"
239         movx @dptr, a
240 
241         mov dptr, #82BDH
242         mov a, #"6"
243         movx @dptr, a
244 
245         mov dptr, #82D7H
246         mov a, #"7"
247         movx @dptr, a
248 
249         mov dptr, #82DBH
250         mov a, #"8"
251         movx @dptr, a
252 
253         mov dptr, #82DDH
254         mov a, #"9"
255         movx @dptr, a
256 
257         mov dptr, #827EH
258         mov a, #"A"
259         movx @dptr, a
260 
261         mov dptr, #82BEH
262         mov a, #"B"
263         movx @dptr, a
264 
265         mov dptr, #82DEH
266         mov a, #"C"
267         movx @dptr, a
268         ret
269 
```

```
// program główny
    start:  init_LCD

      acall keyascii

      mov r3, #80h          ; zakladamy, ze na poczatku dzialamy w
         trybie *

 key_1:  mov r0, #LINE_1
      mov a, r0
      mov P5, a
      mov a, P7
      anl a, r0
      mov r2, a
      clr c
      subb a, r0
      jz key_2
      mov a, r2
      mov dph, r3
      mov dpl, a
      movx a,@dptr
      mov P1, a
      acall putcharLCD
      acall delay

 key_2:  mov r0, #LINE_2
      mov a, r0
      mov P5, a
      mov a, P7
      anl a, r0
      mov r2, a
      clr c
      subb a, r0
      jz key_3
      mov a, r2
      mov dph, r3
      mov dpl, a
      movx a,@dptr
      mov P1, a
      acall putcharLCD
      acall delay

 key_3:  mov r0, #LINE_3
      mov a, r0
      mov P5, a
      mov a, P7
      anl a, r0
      mov r2, a
      clr c
      subb a, r0
      jz key_4
```

```
320         mov a, r2
321         mov dph, r3
322         mov dpl, a
323         movx a,@dptr
324         mov P1, a
325         acall putcharLCD
326         acall delay
327
328     key_4:  mov r0, #LINE_4
329         mov a, r0
330         mov P5, a
331         mov a, P7
332         anl a, r0
333         mov r2, a
334         clr c
335         subb a, r0
336         jz key_1
337         mov a, r2
338         ;mamy teraz backup w r2
339         ;sprawdzenie *
340         clr c
341         subb a, #0E7H ;kod skaningowy *
342         jz modeStar
343         ;jesli nie, to odtworz a i sprawdz #
344         mov a, r2
345         clr c
346         subb a, #0EDH ;kod skaningowy #
347         jz modeHash
348         ;jesli nie, to odtworz i sprawdz D
349         mov a, r2
350         clr c
351         subb a, #0EEH ;kod skaningowy D
352         jz modeD
353         ;jesli nic sie nie spelnilo, to znaczy ze mamy naciniety
              klawisz "0", i mozemy bezpiecznie nawiazac do zainicjowanej
              wczesniej komorki pamieci
354         mov a, r2
355         mov dph, r3
356         mov dpl, a
357         movx a,@dptr
358         mov P1, a
359         acall putcharLCD
360         acall delay
361
362       zapetl:
363         jmp key_1
364
365         ;minifunkcje ustawiajace odpowiedni uklad klawiatury
366         modeStar:
367           mov r3, #080h
368           sjmp zapetl
```

```
369
370         modeHash:
371           mov r3, #081h
372           sjmp zapetl
373
374         modeD:
375           mov r3, #082h
376           sjmp zapetl
377
378       nop
379       nop
380       nop
381       jmp $
382       end start
```

2 Zadanie 2

2.1 Polecenie

Przygotować program wyświetlający na LCD znaki uzyskiwane przez naciśnięcia klawiszy klawiatury. Uzyskiwany z klawiatury tekst jest wyświetlany tak by w pierwszej linii LCD pokazanych zostało 16 znaków, po czym następuje automatyczne przejście do drugiej linii, gdzie wyświetlamy kolejne 16 znaków. Jeśli z klawiatury generujemy kolejne znaki, to kasujemy bieżącą zawartość wyświetlacza i znów w pierwszej linii wyświetlamy następne 16 znaków tekstu, w drugiej - kolejne 16 znaków tekstu, itd.

2.2 Rozwiązanie

```
1  ljmp start
2
3  P5 equ 0F8H
4  P7 equ 0DBH
5
6  LCDstatus  equ 0FF2EH        ; adres do odczytu gotowosci LCD
7  LCDcontrol equ 0FF2CH        ; adres do podania bajtu sterujacego LCD
8  LCDdataWR  equ 0FF2DH        ; adres do podania kodu ASCII na LCD
9
10 // bajty sterujace LCD, inne dostepne w opisie LCD na stronie WWW
11 #define  HOME     0x80      // put cursor to second line
12 #define  INITDISP 0x38      // LCD init (8-bit mode)
13 #define  HOM2     0xc0      // put cursor to second line
14 #define  LCDON    0x0e      // LCD nn, cursor off, blinking off
15 #define  CLEAR    0x01      // LCD display clear
16
17 // linie klawiatury - sterowanie na port P5
18 #define LINE_1    0x7f  // 0111 1111
19 #define LINE_2    0xbf  // 1011 1111
20 #define LINE_3    0xdf  // 1101 1111
21 #define LINE_4    0xef  // 1110 1111
22 #define ALL_LINES 0x0f  // 0000 1111
23
```

```
24 org 0100H
25
26 // macro do wprowadzenia bajtu sterujacego na LCD
27 LCDcntrlWR MACRO x          ; x - parametr wywolania macra - bajt
      sterujacy
28          LOCAL loop         ; LOCAL oznacza ze etykieta loop moze
               sie powtórzyc w programie
29 loop: MOV  DPTR,#LCDstatus  ; DPTR zaladowany adresem statusu
30       MOVX A,@DPTR          ; pobranie bajtu z biezacym statusem LCD
31       JB   ACC.7,loop       ; testowanie najstarszego bitu
            akumulatora
32                             ; - wskazuje gotowosc LCD
33       MOV  DPTR,#LCDcontrol ; DPTR zaladowany adresem do podania
            bajtu sterujacego
34       MOV  A, x             ; do akumulatora trafia argument
            wywolania -macrabajt sterujacy
35       MOVX @DPTR,A          ; bajt sterujacy podany do LCD - zadana
            akcja widoczna na LCD
36       ENDM
37
38 // macro do wypisania znaku ASCII na LCD, znak ASCII przed
      wywolaniem macra ma byc w A
39 LCDcharWR MACRO
40       LOCAL tutu            ; LOCAL oznacza ze etykieta tutu moze
            sie powtórzyc w programie
41       PUSH ACC              ; odlozenie biezacej zawartosci
            akumulatora na stos
42 tutu: MOV  DPTR,#LCDstatus  ; DPTR zaladowany adresem statusu
43       MOVX A,@DPTR          ; pobranie bajtu z biezacym statusem LCD
44       JB   ACC.7,tutu       ; testowanie najstarszego bitu
            akumulatora
45                             ; - wskazuje gotowosc LCD
46       MOV  DPTR,#LCDdataWR  ; DPTR zaladowany adresem do podania
            bajtu sterujacego
47       POP  ACC              ; w akumulatorze ponownie kod ASCII
            znaku na LCD
48       MOVX @DPTR,A          ; kod ASCII podany do LCD - znak widoczny
             na LCD
49       ENDM
50
51 // macro do inicjalizacji wyswietlacza - bez parametrów
52 init_LCD MACRO
53          LCDcntrlWR #INITDISP ; wywolanie macra LCDcntrlWR -
               inicjalizacja LCD
54          LCDcntrlWR #CLEAR    ; wywolanie macra LCDcntrlWR -
               czyszczenie LCD
55          LCDcntrlWR #LCDON    ; wywolanie macra LCDcntrlWR -
               konfiguracja kursora
56          ENDM
57
58 // funkcja opóznienia
```

```
59
60    delay:   mov r1, #0FFH
61    dwa:   mov r2, #0FFH
62       trzy: djnz r2, trzy
63          djnz r1, dwa
64          ret
65
66  // funkcja wypisania znaku
67  putcharLCD: LCDcharWR
68          ret
69
70  // tablica przekodowania klawisze - ASCII w XRAM
71
72  keyascii:
73          mov dptr, #80EBH
74          mov a, #"0"
75          movx @dptr, a
76
77          mov dptr, #8077H
78          mov a, #"1"
79          movx @dptr, a
80
81          mov dptr, #807BH
82          mov a, #"2"
83          movx @dptr, a
84
85          mov dptr, #807DH
86          mov a, #"3"
87          movx @dptr, a
88
89          mov dptr, #80B7H
90          mov a, #"4"
91          movx @dptr, a
92
93          mov dptr, #80BBH
94          mov a, #"5"
95          movx @dptr, a
96
97          mov dptr, #80BDH
98          mov a, #"6"
99          movx @dptr, a
100
101         mov dptr, #80D7H
102         mov a, #"7"
103         movx @dptr, a
104
105         mov dptr, #80DBH
106         mov a, #"8"
107         movx @dptr, a
108
109         mov dptr, #80DDH
```

```
110       mov a, #"9"
111       movx @dptr, a
112
113       mov dptr, #807EH
114       mov a, #"A"
115       movx @dptr, a
116
117       mov dptr, #80BEH
118       mov a, #"B"
119       movx @dptr, a
120
121       mov dptr, #80DEH
122       mov a, #"C"
123       movx @dptr, a
124
125       mov dptr, #80EEH
126       mov a, #"D"
127       movx @dptr, a
128
129       mov dptr, #80E7H
130       mov a, #"*"
131       movx @dptr, a
132
133       mov dptr, #80EDH
134       mov a, #"#"
135       movx @dptr, a
136
137       ret
138
139 ; znak do wyswietlenia w akumulatorze, ktory jest uzywany -
      koniecznosc uzycia stosu lub innego rejestru
140 putkbdCharsin2Lines:
141     ;sprawdzenie, czy r3==#20H, wtedy przenosimy kursor do pierwszej
          linii
142     mov r4, a
143     mov a, r3
144     clr c
145     subb a, #20H
146     jnz nieUstawiajPoczatku1Linii ;
147
148     LCDcntrlWR #HOME
149
150     nieUstawiajPoczatku1Linii:
151     mov a, r3
152     clr c
153     subb a, #10H
154     jnz nieUstawiajPoczatku2Linii
155
156     LCDcntrlWR #HOM2
157
158     nieUstawiajPoczatku2Linii:
```

```
159     mov a, r3
160     clr c
161     subb a, #00H
162     jnz nieClear
163 
164     LCDcntrlWR #CLEAR
165     LCDcntrlWR #HOME
166     mov r3, #20H
167 
168     nieClear:
169     mov a, r4 ; a - wartosc znaku do wpisania na wyswietlacz
170     acall putCharLCD
171     dec r3
172 
173  koniec: ret
174 
175 // program główny
176     start:  init_LCD
177 
178       acall keyascii
179 
180       mov r3, #20H ;licznik pomocniczy
181 
182   key_1:  mov r0, #LINE_1
183       mov a, r0
184       mov P5, a
185       mov a, P7
186       anl a, r0
187       mov r2, a
188       clr c
189       subb a, r0
190       jz key_2
191       mov a, r2
192       mov dph, #80h
193       mov dpl, a
194       movx a,@dptr
195       mov P1, a
196       acall putkbdCharsin2Lines
197       acall delay
198 
199   key_2:  mov r0, #LINE_2
200       mov a, r0
201       mov P5, a
202       mov a, P7
203       anl a, r0
204       mov r2, a
205       clr c
206       subb a, r0
207       jz key_3
208       mov a, r2
209       mov dph, #80h
```

```
210         mov dpl, a
211         movx a,@dptr
212         mov P1, a
213         acall putkbdCharsin2Lines
214         acall delay
215
216     key_3:  mov r0, #LINE_3
217         mov a, r0
218         mov P5, a
219         mov a, P7
220         anl a, r0
221         mov r2, a
222         clr c
223         subb a, r0
224         jz key_4
225         mov a, r2
226         mov dph, #80h
227         mov dpl, a
228         movx a,@dptr
229         mov P1, a
230         acall putkbdCharsin2Lines
231         acall delay
232
233     key_4:  mov r0, #LINE_4
234         mov a, r0
235         mov P5, a
236         mov a, P7
237         anl a, r0
238         mov r2, a
239         clr c
240         subb a, r0
241         jz key_1
242         mov a, r2
243         mov dph, #80h
244         mov dpl, a
245         movx a,@dptr
246         mov P1, a
247         acall putkbdCharsin2Lines
248         acall delay
249
250     zapetl:
251         jmp key_1
252
253
254
255
256
257       nop
258       nop
259       nop
260       jmp $
```

```
261     end start
```

3 Zadanie 3

3.1 Polecenie

Zadanie nadobowiązkowe. Przygotować program realizujący wyłączenie repetycji naciśniętego klawisza - jego kod skaningowy jest poddawany dalszym operacjom dopiero po naciśnięciu i następującym potem zwolnieniu klawisza.

3.2 Rozwiązanie

```
1  ljmp start
2
3  P5 equ 0F8H
4  P7 equ 0DBH
5
6  LCDstatus  equ 0FF2EH        ; adres do odczytu gotowosci LCD
7  LCDcontrol equ 0FF2CH        ; adres do podania bajtu sterujacego LCD
8  LCDdataWR  equ 0FF2DH        ; adres do podania kodu ASCII na LCD
9
10 // bajty sterujace LCD, inne dostepne w opisie LCD na stronie WWW
11 #define  HOME     0x80      // put cursor to second line
12 #define  INITDISP 0x38      // LCD init (8-bit mode)
13 #define  HOM2     0xc0      // put cursor to second line
14 #define  LCDON    0x0e      // LCD nn, cursor off, blinking off
15 #define  CLEAR    0x01      // LCD display clear
16
17 // linie klawiatury - sterowanie na port P5
18 #define LINE_1    0x7f  // 0111 1111
19 #define LINE_2    0xbf  // 1011 1111
20 #define LINE_3    0xdf  // 1101 1111
21 #define LINE_4    0xef  // 1110 1111
22 #define ALL_LINES 0x0f  // 0000 1111
23
24 org 0100H
25
26 // macro do wprowadzenia bajtu sterujacego na LCD
27 LCDcntrlWR MACRO x           ; x - parametr wywolania macra - bajt
      sterujacy
28           LOCAL loop         ; LOCAL oznacza ze etykieta loop moze
                sie powtórzyc w programie
29 loop: MOV  DPTR,#LCDstatus   ; DPTR zaladowany adresem statusu
30       MOVX A,@DPTR           ; pobranie bajtu z biezacym statusem LCD
31       JB   ACC.7,loop        ; testowanie najstarszego bitu
            akumulatora
32                              ; - wskazuje gotowosc LCD
33       MOV  DPTR,#LCDcontrol  ; DPTR zaladowany adresem do podania
            bajtu sterujacego
34       MOV  A, x              ; do akumulatora trafia argument
            wywolania -macrabajt sterujacy
```

```
35        MOVX @DPTR,A            ; bajt sterujacy podany do LCD - zadana
              akcja widoczna na LCD
36        ENDM
37
38  // macro do wypisania znaku ASCII na LCD, znak ASCII przed
        wywolaniem macra ma byc w A
39  LCDcharWR MACRO
40        LOCAL tutu              ; LOCAL oznacza ze etykieta tutu moze
              sie powtórzyc w programie
41        PUSH ACC                ; odlozenie biezacej zawartosci
              akumulatora na stos
42  tutu: MOV  DPTR,#LCDstatus    ; DPTR zaladowany adresem statusu
43        MOVX A,@DPTR            ; pobranie bajtu z biezacym statusem LCD
44        JB   ACC.7,tutu         ; testowanie najstarszego bitu
              akumulatora
45                                ; - wskazuje gotowosc LCD
46        MOV  DPTR,#LCDdataWR    ; DPTR zaladowany adresem do podania
              bajtu sterujacego
47        POP  ACC                ; w akumulatorze ponownie kod ASCII
              znaku na LCD
48        MOVX @DPTR,A            ; kod ASCII podany do LCD - znak widoczny
               na LCD
49        ENDM
50
51  // macro do inicjalizacji wyswietlacza - bez parametrów
52  init_LCD MACRO
53            LCDcntrlWR #INITDISP ; wywolanie macra LCDcntrlWR -
                  inicjalizacja LCD
54            LCDcntrlWR #CLEAR    ; wywolanie macra LCDcntrlWR -
                  czyszczenie LCD
55            LCDcntrlWR #LCDON    ; wywolanie macra LCDcntrlWR -
                  konfiguracja kursora
56            ENDM
57
58  // funkcja opóznienia
59    delay:  mov r0, #15H
60    one:    mov r1, #0FFH
61    dwa:    mov r2, #0FFH
62      trzy: djnz r2, trzy
63          djnz r1, dwa
64          djnz r0, one
65          ret
66
67  // funkcja wypisania znaku
68  putcharLCD: LCDcharWR
69          ret
70
71  // tablica przekodowania klawisze - ASCII w XRAM
72
73  keyascii: mov dptr, #80EBH
74          mov a, #"0"
```

```
75      movx @dptr, a
76
77      mov dptr, #8077H
78      mov a, #"1"
79      movx @dptr, a
80
81      mov dptr, #807BH
82      mov a, #"2"
83      movx @dptr, a
84
85      mov dptr, #807DH
86      mov a, #"3"
87      movx @dptr, a
88
89      mov dptr, #80B7H
90      mov a, #"4"
91      movx @dptr, a
92
93      mov dptr, #80BBH
94      mov a, #"5"
95      movx @dptr, a
96
97      mov dptr, #80BDH
98      mov a, #"6"
99      movx @dptr, a
100
101     mov dptr, #80D7H
102     mov a, #"7"
103     movx @dptr, a
104
105     mov dptr, #80DBH
106     mov a, #"8"
107     movx @dptr, a
108
109     mov dptr, #80DDH
110     mov a, #"9"
111     movx @dptr, a
112
113     mov dptr, #807EH
114     mov a, #"A"
115     movx @dptr, a
116
117     mov dptr, #80BEH
118     mov a, #"B"
119     movx @dptr, a
120
121     mov dptr, #80DEH
122     mov a, #"C"
123     movx @dptr, a
124
125     mov dptr, #80EEH
```

```
126         mov a, #"D"
127         movx @dptr, a
128 
129         mov dptr, #80E7H
130         mov a, #"*"
131         movx @dptr, a
132 
133         mov dptr, #80EDH
134         mov a, #"#"
135         movx @dptr, a
136 
137         ret
138 
139 // program główny
140     start:  init_LCD
141 
142         acall keyascii
143         ; w nawiasach co sie dzieje gdy przycisk spelniajacy warunek
              jest wcisniety
144   key_1:  mov r0, #LINE_1; wczytanie linii pierwszej [r0 = 0111
            1111]
145         mov a, r0; wpisanie r0 do akumulatora [ a = 0111 1111]
146         mov P5, a; aktywacja portu P5 [ P5 = 0111 1111]
147         mov a, P7; wczytanie informacji o wcisnietym przycisku [P7 =
              1111 0111 -> a = 1111 0111]
148         anl a, r0; AND - utworzenie maski wiersza [a = 0111 0111]
149         mov r2, a; zapisanie akumulatora na potem [r2 = 0111 0111]
150         clr c; wyczyszczenie c aby nie kolidowal w subb
151         subb a, r0; sprawdzenie czy jest wcisniety przycisk [0111 0111
              - 0111 1111 =~ 0000 1000] (=~ niprawdziwy wynik ale
              pokazane ze rozne od zera)
152         jz key_2; jezeli nie wcisniety skocz do nastepnego
153         ; jezeli przycisk jest wcisniety to wykonaj wyswietlenie:
154         mov a, r2; wczytaj co jest wcisniete [a = 0111 0111]
155         mov dph, #80h ; wpisz wartosc 80 do starszej czesci dptr
156         mov dpl, a; wpisz akumulator do mlodszej czesci dptr
157         movx a,@dptr; ladowanie znaku ascii do akumulatora
158         mov P1, a; podaj znak na port P1 - Diody
159         acall putcharLCD; wypisz znak
160 
161 
162 wcisniety_1:
163         ; nawiasy z lewej = wcisniete to samo, z prawej = klawisz
              puszczony
164         mov a, P7; wczytanie informacji o wcisetej kolumnie [P7 = 1111
              0111 -> a = 1111 0111] [P7 = 1111 1111 -> a = 1111 1111]
165         anl a, r0; [a = 0111 0111] [a = 0111 1111]
166         clr c; wyczyszczenie aby nie przeszkadzal
167         subb a, r0; odejmij, jezeli jest to samo co wyzej to beda
              wszystkie 0 [ 0111 0111 - 0111 1111 =~ 0000 1000] [ 0111
              1111 - 0111 1111 = 0000 0000]
```

```
168         jnz wcisniety_1; to nie przechodz dalej
169
170     key_2:  mov r0, #LINE_2
171         mov a, r0
172         mov P5, a
173         mov a, P7
174         anl a, r0
175         mov r2, a
176         clr c
177         subb a, r0
178         jz key_3
179         mov a, r2
180         mov dph, #80h
181         mov dpl, a
182         movx a,@dptr
183         mov P1, a
184         acall putcharLCD
185 wcisniety_2:
186         ; nawiasy z lewej = wcisniete to samo, z prawej = klawisz
              puszczony
187         mov a, P7; wczytanie informacji o wcisetej kolumnie [P7 = 1111
              0111 -> a = 1111 0111] [P7 = 1111 1111 -> a = 1111 1111]
188         anl a, r0; [a = 0111 0111] [a = 0111 1111]
189         clr c; wyczyszczenie aby nie przeszkadzal
190         subb a, r0; odejmij, jezeli jest to samo co wyzej to beda
              wszystkie 0 [ 0111 0111 - 0111 1111 =~ 0000 1000] [ 0111
              1111 - 0111 1111 = 0000 0000]
191         jnz wcisniety_2; to nie przechodz dalej
192
193     key_3:  mov r0, #LINE_3
194         mov a, r0
195         mov P5, a
196         mov a, P7
197         anl a, r0
198         mov r2, a
199         clr c
200         subb a, r0
201         jz key_4
202         mov a, r2
203         mov dph, #80h
204         mov dpl, a
205         movx a,@dptr
206         mov P1, a
207         acall putcharLCD
208 wcisniety_3:
209         ; nawiasy z lewej = wcisniete to samo, z prawej = klawisz
              puszczony
210         mov a, P7; wczytanie informacji o wcisetej kolumnie [P7 = 1111
              0111 -> a = 1111 0111] [P7 = 1111 1111 -> a = 1111 1111]
211         anl a, r0; [a = 0111 0111] [a = 0111 1111]
212         clr c; wyczyszczenie aby nie przeszkadzal
```

```
213       subb a, r0; odejmij, jezeli jest to samo co wyzej to beda
              wszystkie 0 [ 0111 0111 - 0111 1111 =~ 0000 1000] [ 0111
              1111 - 0111 1111 = 0000 0000]
214       jnz wcisniety_3; to nie przechodz dalej
215
216   key_4:  mov r0, #LINE_4
217       mov a, r0
218       mov P5, a
219       mov a, P7
220       anl a, r0
221       mov r2, a
222       clr c
223       subb a, r0
224       jz key_1
225       mov a, r2
226       mov dph, #80h
227       mov dpl, a
228       movx a,@dptr
229       mov P1, a
230       acall putcharLCD
231 wcisniety_4:
232       ; nawiasy z lewej = wcisniete to samo, z prawej = klawisz
              puszczony
233       mov a, P7; wczytanie informacji o wcisetej kolumnie [P7 = 1111
              0111 -> a = 1111 0111] [P7 = 1111 1111 -> a = 1111 1111]
234       anl a, r0; [a = 0111 0111] [a = 0111 1111]
235       clr c; wyczyszczenie aby nie przeszkadzal
236       subb a, r0; odejmij, jezeli jest to samo co wyzej to beda
              wszystkie 0 [ 0111 0111 - 0111 1111 =~ 0000 1000] [ 0111
              1111 - 0111 1111 = 0000 0000]
237       jnz wcisniety_4; to nie przechodz dalej
238
239       jmp key_1
240
241
242
243     nop
244     nop
245     nop
246     jmp $
247     end start
```