Problem 11

In this problem we use 4X4 keypad with 4 rows, and 4 columns.

7 segment is common cathode

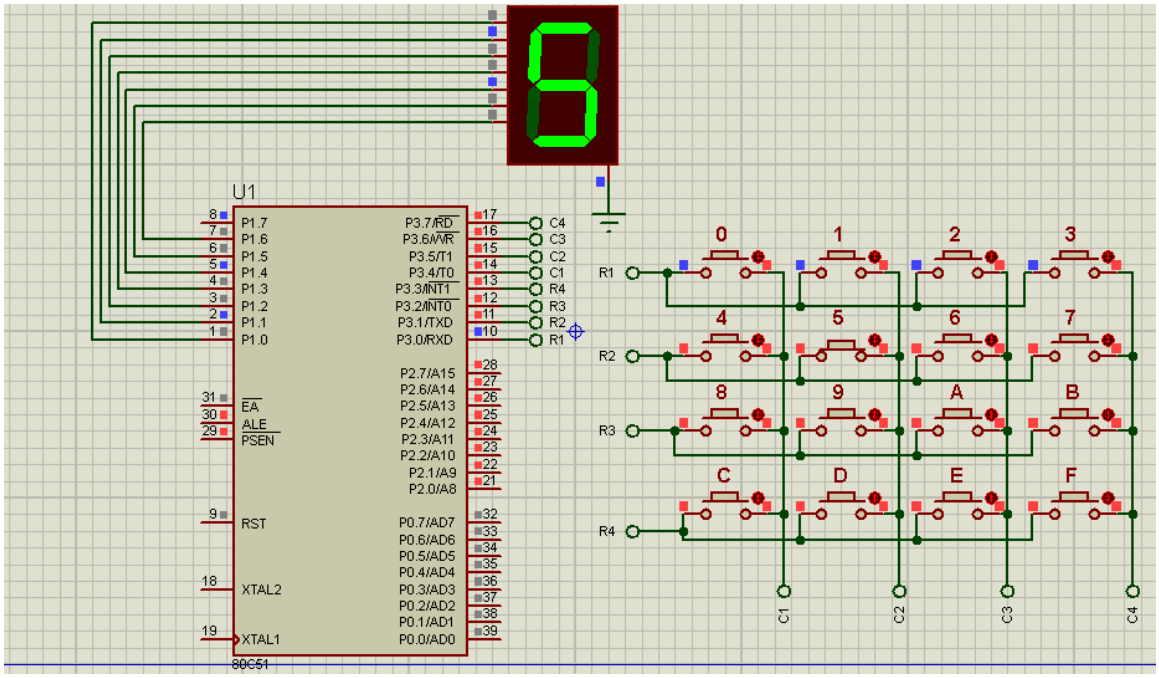
Reading from matrix keypad is done using the scan process.

Rows are used as output from the microcontroller, and columns are input

Columns pins have a default value of '1' due to internal pullups of the uC

The program will activate (put a zero) row by row starting from row 1 to row 4 then repeates.

When we activate row 1, we read the value of each column. If there is a "0" in a column, this means that the key connected between the activated rowvand this column was pressed



Section 1 initialization

```
8  ;===================================================================
9  ; VARIABLES
10 KEY EQU 30H
11 ROW1 EQU P3.0
12 ROW2 EQU P3.1
13 ROW3 EQU P3.2
14 ROW4 EQU P3.3
15 COL1 EQU P3.4
16 COL2 EQU P3.5
17 COL3 EQU P3.6
18 COL4 EQU P3.7
19
20 ;===================================================================
```

Definitions for the rows and columns pins

One variable KEY, used to store the key to be displayed

Section 2 Main code

```
33 Start:
34
35    MOV P1,#0            ;blank display
36 LOOP:
37    CALL READ_KEY        ;read if key is pressed
38    JNC Start            ;no carry means no key pressed, so go start and blank the display
39    MOV A,KEY            ;convert the key to 7-segment code
40    MOV DPTR,#DIGIT_CODE
41    MOVC A ,@A + DPTR
42    MOV P1,A
43    JMP LOOP             ;read again
44 ;====================================================================
```

We start by displaying a blank 7-segment (line 35)

Call the function that scans the rows and read the columns (37). The function return the key pressed in the "KEY" variable

Then we convert it into 7-segment code (40,41)

The conversion process depend on the following table in the code

```
134 DIGIT_CODE:
135 DB 3FH ; digit drive pattern for 0
136 DB 06H ; digit drive pattern for 1
137 DB 5BH ; digit drive pattern for 2
138 DB 4FH ; digit drive pattern for 3
139 DB 66H ; digit drive pattern for 4
140 DB 6DH ; digit drive pattern for 5
141 DB 7DH ; digit drive pattern for 6
142 DB 07H ; digit drive pattern for 7
143 DB 7FH ; digit drive pattern for 8
144 DB 6FH ; digit drive pattern for 9
145 DB 1110111B ; digit drive pattern for A
146 DB 1111100B ; digit drive pattern for B
147 DB 0111001B ; digit drive pattern for C
148 DB 1011110B ; digit drive pattern for D
149 DB 1111001B ; digit drive pattern for E
150 DB 1110001B ; digit drive pattern for F
151 DB 40H ; -
```

Same table and conversion as in problem 10, with the addition of the characters "ABCDEF" and the "-" that indicates multiple key pressed

Subroutine READ_KEY

```
45 READ_KEY:
46    CLR A                ;ACC will be used to indicate number of keys pressed
47    CLR ROW1             ;Activate row1
48    JB COL1,NOT0         ;if COL1 = 1 --> no key pressed in the colou
49    MOV B,#0
50    INC A
51 NOT0:
52    JB COL2,NOT1
53    MOV B,#1
54    INC A
55 NOT1:
56    JB COL3,NOT2
57    MOV B,#2
58    INC A
59 NOT2:
60    JB COL4,NOT3
61    MOV B,#3
62    INC A
63 NOT3:
```

A: is used to count the number of keys pressed

B: is the key pressed or the last key pressed in multiple key case

First, we activate row1 (47) then we test all columns from column 1 to column 4 (48, 52, 56, 60).

If a column is read as zero → we store its corresponding key value in "B"

Note that we also increment "A" each time we found a key press, so A will have the number of key pressed

We repeat the process for every row

```
NOT3:
   SETB ROW1
   CLR ROW2
   JB COL1,NOT4
   MOV B,#4
   INC A
NOT4:
   JB COL2,NOT5
   MOV B,#5
   INC A
NOT5:
   JB COL3,NOT6
   MOV B,#6
   INC A
NOT6:
   JB COL4,NOT7
   MOV B,#7
   INC A
NOT7:
```

```
NOT7:
   SETB ROW2
   CLR ROW3
   JB COL1,NOT8
   MOV B,#8
   INC A
NOT8:
   JB COL2,NOT9
   MOV B,#9
   INC A
NOT9:
   JB COL3,NOT10
   MOV B,#10
   INC A
NOT10:
   JB COL4,NOT11
   MOV B,#11
   INC A
NOT11:
```

```
99  NOT11:
100    SETB ROW3
101    CLR ROW4
102    JB COL1,NOT12
103    MOV B,#12
104    INC A
105 NOT12:
106    JB COL2,NOT13
107    MOV B,#13
108    INC A
109 NOT13:
110    JB COL3,NOT14
111    MOV B,#14
112    INC A
113 NOT14:
114    JB COL4,NOT15
115    MOV B,#15
116    INC A
117 NOT15:
```

Finally, we determine the following cases

No key press → A = 0 (line 121, we clear the carry)

Only one key press → A = 1(lines 125-127, we store key value and set the carry)

Multiple key press → A> 1 (line 129, we store 16 in key position → this is the code for the minus sign)

```
120    CJNE A,#0,TEST_MULTIPLE_KEYS
121    CLR C
122    RET
123 TEST_MULTIPLE_KEYS:
124    CJNE A,#1,MULTIPLE_KEYS
125    MOV KEY,B
126    SETB C
127    RET
128 MULTIPLE_KEYS:
129    MOV KEY,#16
130    SETB C
131 RET
```