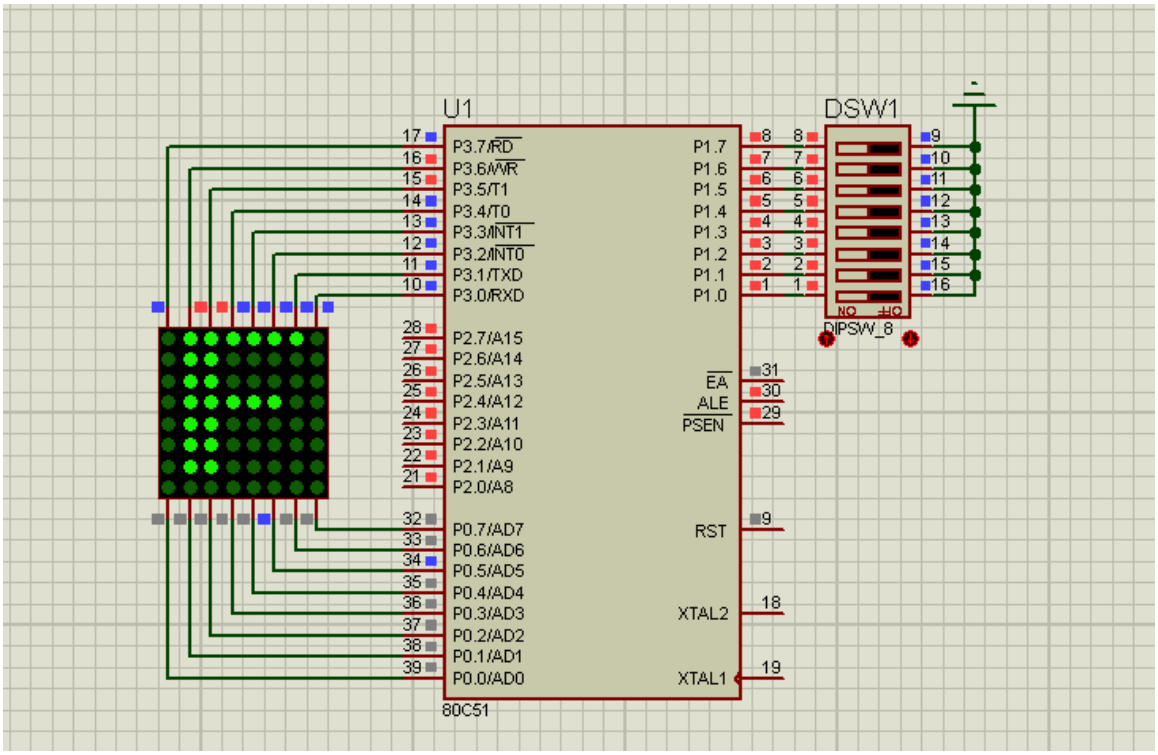
Problem 15



This project is very similar to problem 14, but with a LED dot-matrix instead of 4 seven segment display.

To display on this 8X8 LED matrix, we activate one column at a time from column 1 to column 8. With each active column, we send its corresponding LED status from the memory to the row pins of the dot-matrix.

This process must be repeated with a refresh rate greater then 20hz to deceive human eye.

Section 1 initialization

```
19  ROW_PORT EQU P3
20  COL_PORT EQU P0
21
22  ROW1 EQU 30H
23  ROW2 EQU 31H
24  ROW3 EQU 32H
25  ROW4 EQU 33H
26  ROW5 EQU 34H
27  ROW6 EQU 35H
28  ROW7 EQU 36H
29  ROW8 EQU 37H
30
31  SCAN EQU 38H
32
33  SW1 EQU P1.0
34  SW2 EQU P1.1
35  SW3 EQU P1.2
36  SW4 EQU P1.3
37  SW5 EQU P1.4
38  SW6 EQU P1.5
39  SW7 EQU P1.6
40  SW8 EQU P1.7
```

ROW_PORT → row pins of the dot-matrix

COL_PORT → column pins of the dot-matrix

ROW1-ROW8 → temporary storage of the current character to be displayed on the dot-matrix

SCAN → used to produce the value that will activate the columns of the dot-matrix one by one

SW1-SW8, the 8-bits of DIP switches

Section 2 Main code

```
LOOP:
   JNB SW1,NOT1
   MOV DPTR,#NUM1
   CALL DISP_NUM
NOT1:
   JNB SW2,NOT2
   MOV DPTR,#NUM2
   CALL DISP_NUM
NOT2:
   JNB SW3,NOT3
   MOV DPTR,#NUM3
   CALL DISP_NUM
NOT3:
   JNB SW4,NOT4
   MOV DPTR,#NUM4
   CALL DISP_NUM
NOT4:
   JNB SW5,NOT5
   MOV DPTR,#NUM5
   CALL DISP_NUM
NOT5:
   JNB SW6,NOT6
   MOV DPTR,#NUM6
   CALL DISP_NUM
NOT6:
   JNB SW7,NOT7
   MOV DPTR,#NUM7
   CALL DISP_NUM
NOT7:
   JNB SW8,NOT8
   MOV DPTR,#NUM8
   CALL DISP_NUM
NOT8:


   JMP LOOP
```

Again, it is the same as previous problem, but the DISP_NUM function will be different.

Function DISP_NUM:

```
DISP_NUM:
   MOV R0,#ROW1
   MOV R1,#8
 ALL_ROWS:
   CLR A
   MOVC A,@A + DPTR
   INC DPTR
   MOV @R0,A
   INC R0
   DJNZ R1,ALL_ROWS
```

First, we load the bit-pattern of character to be displayed into internal RAM of the 8051.

R0 points to the first ROW (97)

Read one byte from ROM (DPTR points to the required character data) "lines 100-102", then store it in the RAM (103).

Repeat this process for all 8 bytes.

Next, we go through the scan process here;

```
107     MOV R4,#50
108  REPEAT_1000MS:
109     MOV R0,#ROW1
110     MOV R1,#8
111     MOV SCAN,#11111110B
112     MOV COL_PORT,#0FFH
113  ALL_BITS2:
114     MOV A,@R0
115     INC R0
116     MOV ROW_PORT,A
117     MOV COL_PORT,SCAN
118     CALL DELAY
119     MOV COL_PORT,#0FFH
120     MOV A,SCAN
121     SETB C
122     RLC A
123     MOV SCAN,A
124     DJNZ R1, ALL_BITS2
125
126     DJNZ R4,REPEAT_1000MS
127  RET
```

Similar to the scan procedure in problem 14 but with 8 bytes

Bytes are read from the RAM starting from ROW1 "pointed by R0" (114), and then sent to the ROW_PORT (116).

Now to activate the column, we send the "SCAN" variable to COL_PORT (117). Note that this variable starts with all bits '1' except bit0 = '0' (active low). So at first we activate column 1. To activate the next column we make a left rotation with carry (122) after setting the carry to 1 (121) → the next scan value will be 11111101.

For each column we make it active for a 2.5ms by calling the delay function (118).

After that we need to deactivate all columns by sending "11111111" or "0FFH" (119) before sending next row value.

We repeat this for all 8 columns (124).

Then we repeat all for 50 times (126).

So each character will be displayed for a time of = 2.5ms*8*50=1000ms

The delay function in this problem was adjusted for 2.5ms only instead of 5ms in the previous project

```
129 DELAY:
130     MOV R6,#5
131 L1:
132     MOV R7,#250
133 L0:
134     DJNZ R7,L0
135     DJNZ R6,L1
136 RET
```

Finally, here is the characters definitions (one indicates its corresponding LED is ON)

```
140 NUM1:
141     DB 00011100B
142     DB 00100010B
143     DB 00100010B
144     DB 00100010B
145     DB 00111110B
146     DB 00100010B
147     DB 00100010B
148     DB 00000000B
149 NUM2:
150     DB 00111100B
151     DB 00100010B
152     DB 00100010B
153     DB 00111100B
154     DB 00100010B
155     DB 00100010B
156     DB 00111100B
157     DB 00000000B
158 NUM3:
159     DB 00111100B
160     DB 01100110B
161     DB 01100000B
162     DB 01100000B
163     DB 01100000B
164     DB 01100110B
165     DB 00111100B
166     DB 00000000B
```

```
NUM4:
    DB 01111110B
    DB 01100000B
    DB 01100000B
    DB 01111100B
    DB 01100000B
    DB 01100000B
    DB 01111110B
    DB 00000000B
NUM5:
    DB 01111110B
    DB 01100000B
    DB 01100000B
    DB 01111100B
    DB 01100000B
    DB 01100000B
    DB 01100000B
    DB 00000000B
NUM6:
    DB 00111100B
    DB 01100110B
    DB 01100000B
    DB 01100000B
    DB 01101110B
    DB 01100110B
    DB 00111110B
    DB 00000000B
```

```
NUM7:
    DB 01100110B
    DB 01100110B
    DB 01100110B
    DB 01111110B
    DB 01100110B
    DB 01100110B
    DB 01100110B
    DB 00000000B
NUM8:
    DB 00111100B
    DB 00011000B
    DB 00011000B
    DB 00011000B
    DB 00011000B
    DB 00011000B
    DB 00111100B
    DB 00000000B
```