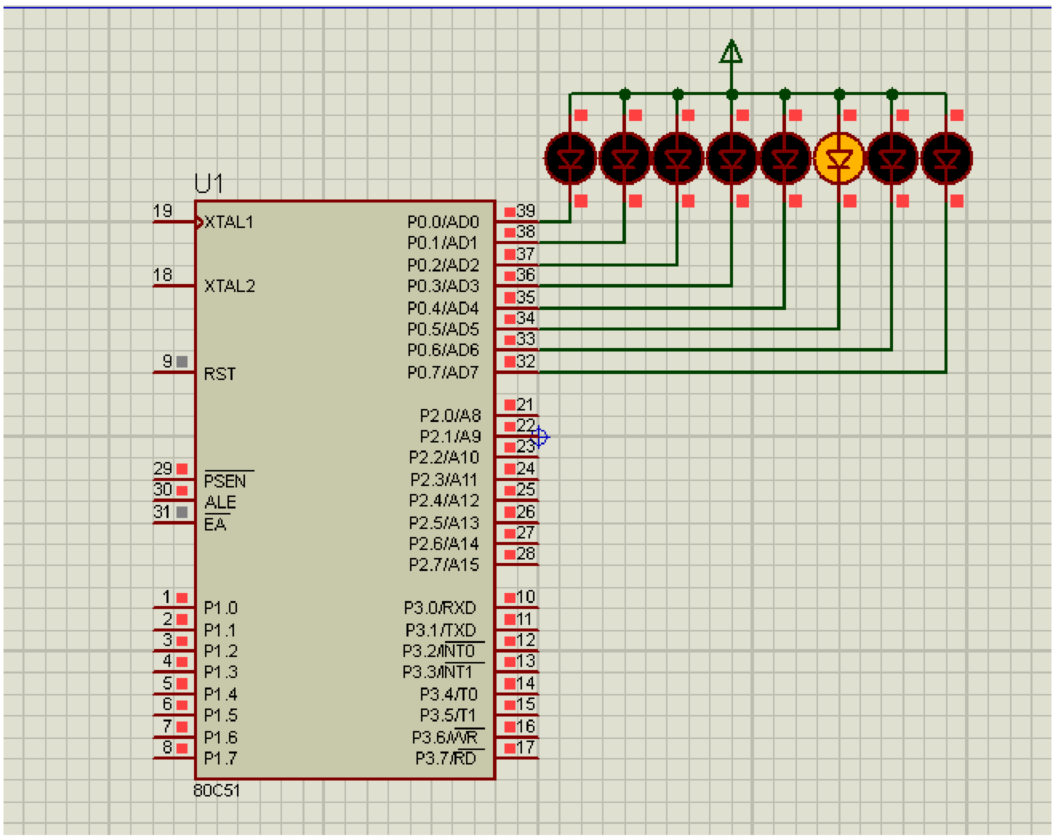
Problem 6

In this problem, we connect 8 LEDs to port 0, and will turn on one LED at a time from LED at P0.0 up to LED at P0.7.

Note that:

LEDs has a common anode connected to 5V, so it will be on when we output logic state '0'

The delay here will be = (the value of R4) * 10ms



Section 1 Variables

```
;===================================

LEDs EQU P0
COUNT EQU 30H
PATTERN EQU 31H
```

LEDS → port connected to LEDs

COUNT → a counter to scan all 8 LEDs

PATTERN → the start pattern for the LEDs

Section 2 Code

```
22 Loop:
23     MOV PATTERN,#00000001B
24     MOV COUNT,#8
25 ALL_BITS:
26     MOV A,PATTERN
27     CPL A
28     MOV LEDS,A
29
30     CALL DELAY_100MS
31
32     MOV A,PATTERN
33     CLR C
34     RLC A
35     MOV PATTERN,A
36
37     DJNZ COUNT,ALL_BITS
38
39     JMP LOOP
```

Lines 23, and 24 initialize both the PATTERN, and COUNT

COUNT is initialized by 8 to scan all 8 LEDs

Lines 26 to 28, read the pattern and complement it, then send it to Port0. We need to invert the pattern as the LEDs are "Active Low"

Line 30→ calls the DELAY_100MS subroutine.

Section 3 Subroutine

```
DELAY_100MS:
    MOV R6,#250
L2:
    MOV R7,#200
L1:
    DJNZ R7,L1
    DJNZ R6,L2
RET
```

It is basically like the delay function in Problem 5, but we will refine it more

The actual delay can be calculated as follows:

$$Delay = \Big(\big((200 * 2\mu s) + 3\mu s\big) * 250 + 3\mu s\Big) + 2\mu s = 100.755\ ms$$

Note: that with 12Mhz clock:

- DJNZ → takes $2\ \mu s$
- MOV R7,#200 → takes $3\mu s$

- RET → takes $2\mu s$

So, we have extra $755\mu s$ which affect the timing accuracy.

But, can refine the subroutine by taking into account these extra instructions that takes extra time.

The most inner loop takes $400\mu s$ and executed 250 times. If we decrease the outer loop by 2, the actual time delay will be;

$$Delay = \left(\left((200 * 2\mu s) + 3\mu s\right) * 248 + 3\mu s\right) + 2\mu s = 99.949\ ms$$

Which has an error of $51\mu s$ → more accurate!

We can improve the accuracy by adding extra instructions to consume the $51\mu s$

For example by executing another DJNZ instruction 24 times will give $48\mu s$ with $3\mu s$ consumed in the initialization, this will give a total of $51\mu s$

Here is the more refined delay subroutine:

```
DELAY_100MS:
   MOV R6,#248
L2:
   MOV R7,#200
L1:
   DJNZ R7,L1
   DJNZ R6,L2

   MOV R7,#24
L0:
   DJNZ R7,L0
RET
```