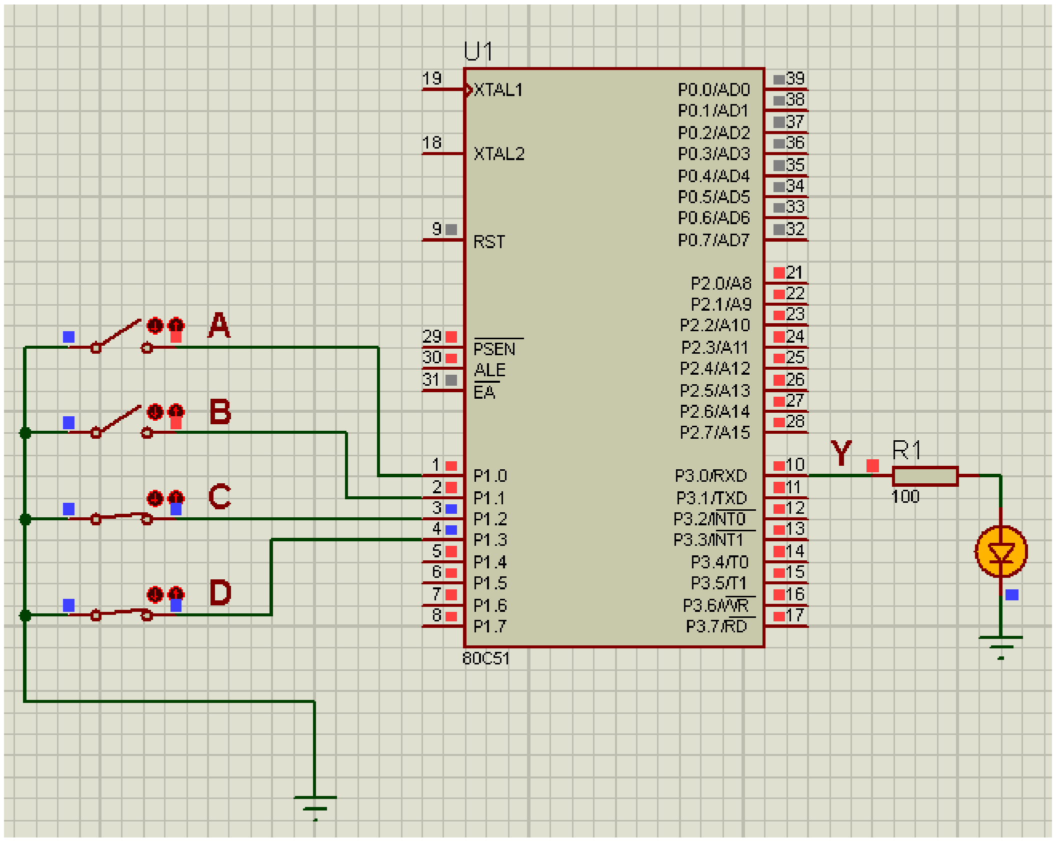
<u>Problem 1:</u>

We use the following logical function to be implemented

$$y = a.b + c.d$$

Here is the circuit connection



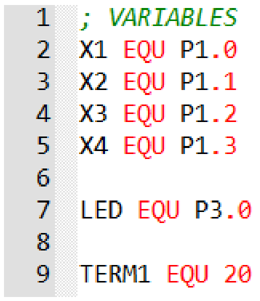
As shown, the switches are connected to P1.0, P1.1, P1.2, and P1.3.

Closed switch represents state "0", while open switch represents state '1'.

The output is connected to P3.0, with High output makes the LED ON.

Here is the code divided into sections:

<u>Section1 Variables definitions</u>



```
; VARIABLES
X1 EQU P1.0
X2 EQU P1.1
X3 EQU P1.2
X4 EQU P1.3

LED EQU P3.0

TERM1 EQU 20
```

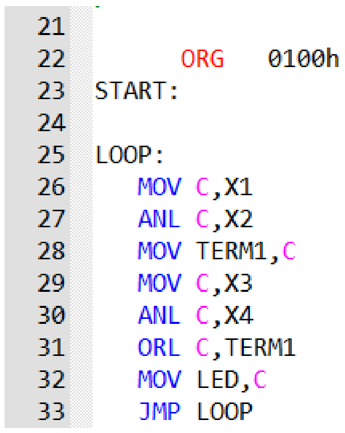
First, we define a name for each input and output to make code more readable.

X1, X2,X3,and X4 are the inputs (correspond to A, B, C, and D).

LED as output

Also, we add another variable TERM1 to facilitate the computation of logical expression, where after computing X1.X2, we will store this term in TERM1.

<u>Section2 code</u>



```
21
22        ORG    0100h
23  START:
24
25  LOOP:
26     MOV C,X1
27     ANL C,X2
28     MOV TERM1,C
29     MOV C,X3
30     ANL C,X4
31     ORL C,TERM1
32     MOV LED,C
33     JMP LOOP
```

First, we start our code from location 100h to free up spaces used by 8051 interrupt handlers. Indeed in this code, we don't use interrupts but as common code style, we use this principle.

Now logic function calculation will be done with the help of the carry flag 'C' where 8051 has many instructions that can perform logical operation on the carry flag.

2nd (line 26), we put the value of X1 on 'C', then we make "AND" operation with X2 as in line 27.

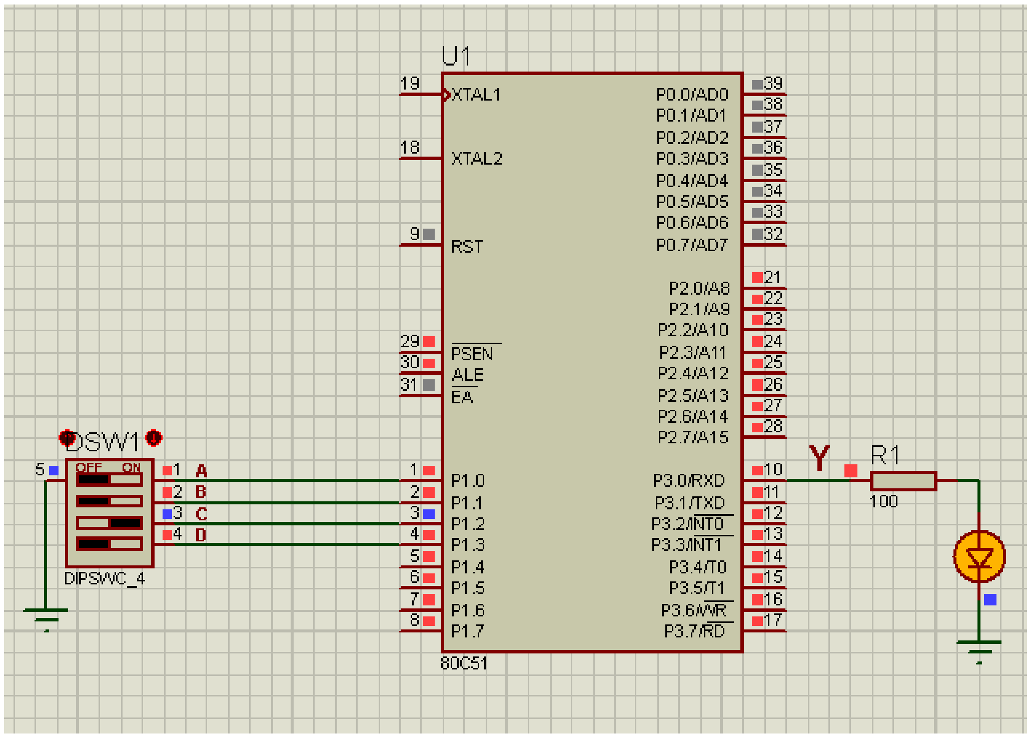
Now to make further calculation for X3.X4, we need to store the first calculation, and here comes the role of using the temp variable TERM1 that holds the value of the first term as in line 28.

Lines 29,30 performs the operation X3.X4, and line 31 will perform the logical OR between the two terms.

Finally we send this value to the LED.

Version 2

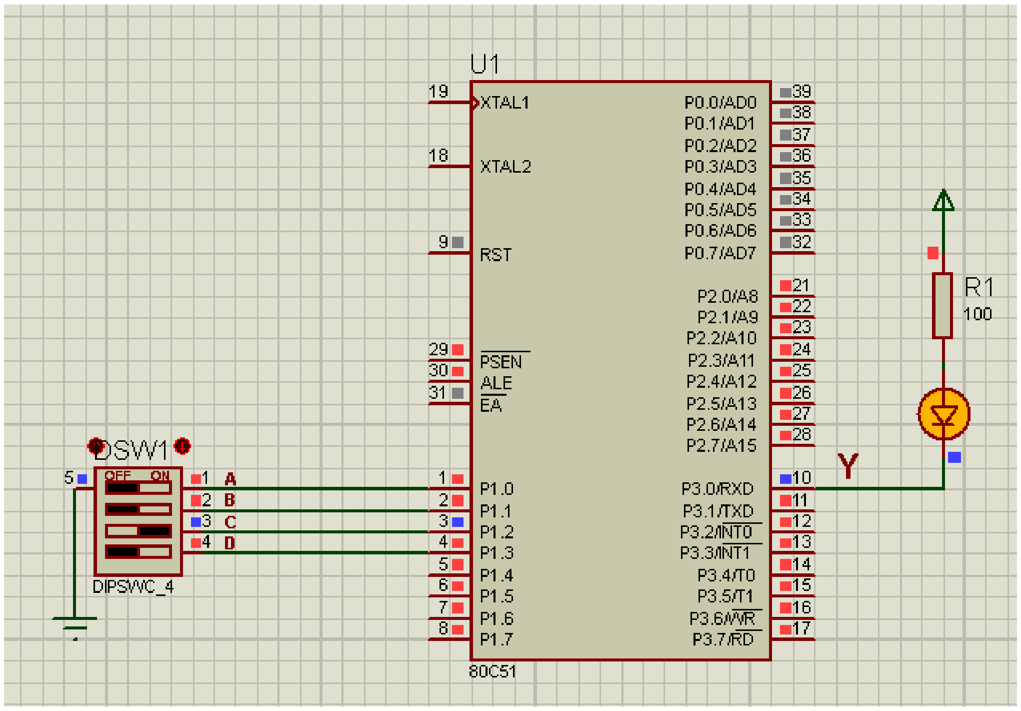
In this version, we use DIP switches instead of separate switches. DIP switches are consists of simple switches in a package of 4 or 8 switches.



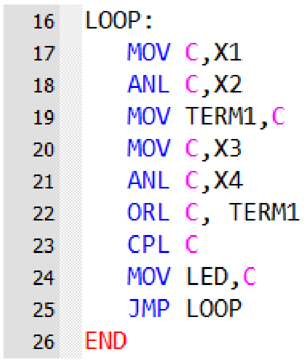
All switches are connected to a common pin "5", so if we connect this pin to GND, it will be the same as of problem 1 and hence the code will be the same.

<u>Version 3</u>

In this version, we use a LED with its anode connected to 5V and its cathode to P3.0 as shown



Now to turn this LED ON, we must output zero volt on pin P3.0. So the code will invert the output to make the LED on when the input expression is true.

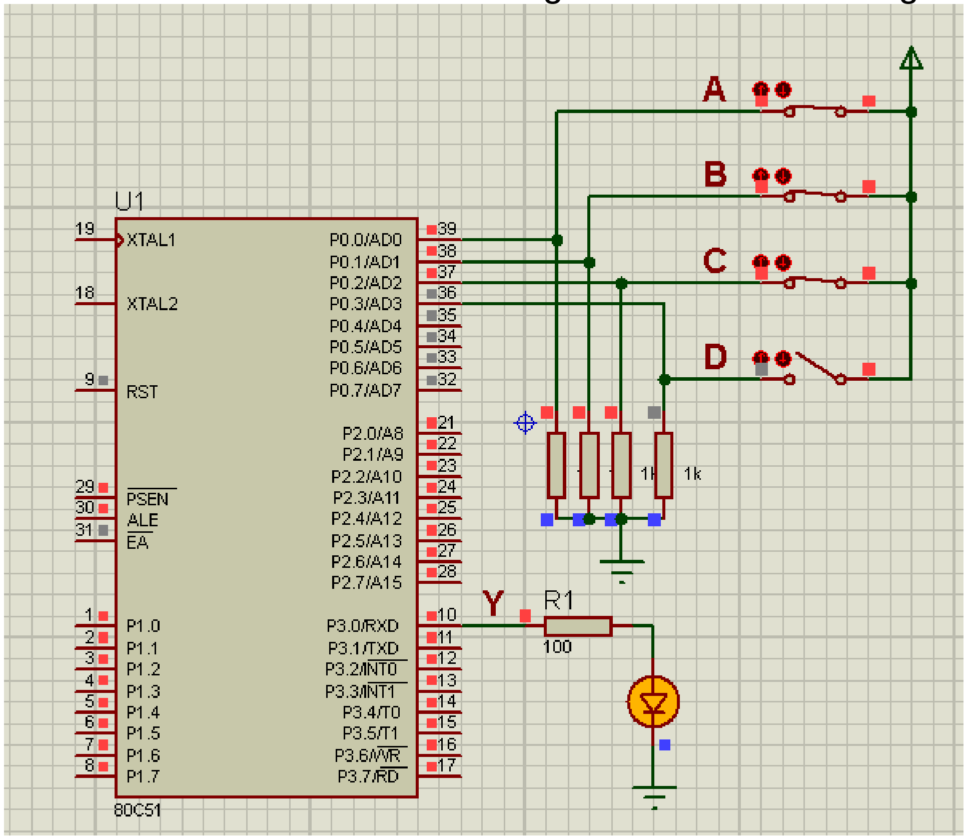


```
16 LOOP:
17     MOV C,X1
18     ANL C,X2
19     MOV TERM1,C
20     MOV C,X3
21     ANL C,X4
22     ORL C, TERM1
23     CPL C
24     MOV LED,C
25     JMP LOOP
26 END
```

We add line 23 that complements the carry before sending it to P3.0

Version 4

In this version we used "Active High" switches → closing a switch gives high state.



Also in this version we use P0 (which has neither pull-up resistors nor pull-down). So we add pull-down resistors to give state '0' when the switch is open.

The code will be the same as problem 1