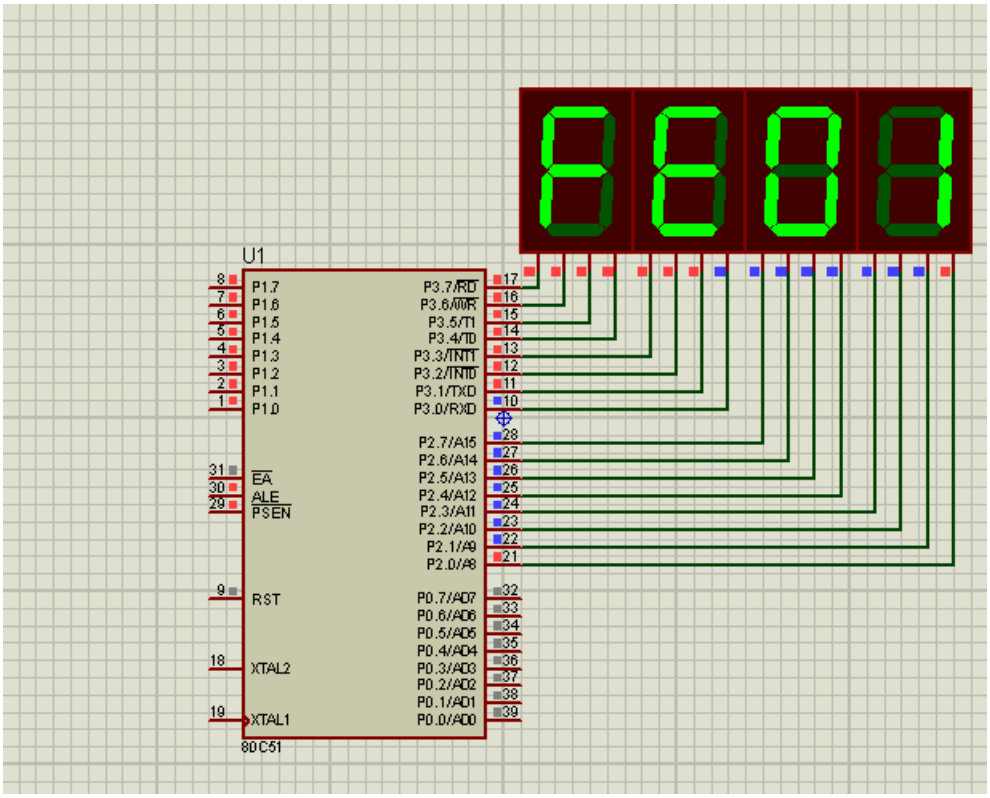
Problem 29_TWO_HEX



In this problem we will multiply two numbers, each number is represented by two hex digits/byte with a length of 2 byte

The algorithm used for the multiply process is as follows

If we have two digit numbers multiplications

```
        Z2 Z1 → FIRST NUMBER
        Y2 Y1 → 2ND NUMBER
  ------------------------------
        X2 X1 → Y1*Z1
     X4 X3    → Y1*Z2
     X6 X5    → Y2*Z1
  X8 X7       → Y2*Z2
--------------------------------
  R4 R3 R2 R1 → RESTUL
```

R1 = X1

R2 = X2+X3+X5

R3 = X4+X6+X7

R4 = X8

So, we will make the four multiplications and save them in variable x1 to x8

Then we add as indicated above to get R1 to R4

Variables

```
1  LEN EQU 2
2
3  TEMP1 EQU 30H
4  TEMP2 EQU 34H
5
6  RESULT EQU 38H
7  TEMP EQU 40H
8  X1 EQU 41H
9  X2 EQU 42H
10 X3 EQU 43H
11 X4 EQU 44H
12 X5 EQU 45H
13 X6 EQU 46H
14 X7 EQU 47H
15 X8 EQU 48H
```

Temp1, temp2 are the numbers to be multiplied

Result saves the result of multiplications

X1 to X8 are the temp storage for partial products

Main code

```
19 START:
20    CALL LOAD_NUMBERS
21    CALL MULTIPLY_TWO_HEX
22    JMP START
```

The main loop start by loading numbers to the ram (temp1,temp2), then call the multiply subroutine

Functions

1-multiply_two_hex

```
24  MULTIPLY_TWO_HEX:
25     MOV A,TEMP1
26     MOV B,TEMP2
27     MUL AB
28     MOV X1,A
29     MOV X2,B
30
31     MOV A,TEMP1
32     MOV B,TEMP2+1
33     MUL AB
34     MOV X3,A
35     MOV X4,B
36
37     MOV A,TEMP1+1
38     MOV B,TEMP2
39     MUL AB
40     MOV X5,A
41     MOV X6,B
42
43     MOV A,TEMP1+1
44     MOV B,TEMP2+1
45     MUL AB
46     MOV X7,A
47     MOV X8,B
```

25-29 multiply Y1*Z1 and store the result in X1,X2

31-35 multiply Y1*Z2 and store the result in X3,X4

37-41 multiply Y2*Z1 and store the result in X5,X6

43-47 multiply Y2*Z2 and store the result in X7,X8

```
49     MOV RESULT,X1
50     MOV A,X2
51     ADD A,X3
52     MOV CR,C
53     ADD A,X5
54     ORL C,CR
55     MOV RESULT+1,A
56
57     MOV A,X4
58     ADDC A,X6
59     MOV CR,C
60     ADD A,X7
61     ORL C,CR
62     MOV RESULT+2,A
63     CLR A
64     ADDC A,X8
65     MOV RESULT+3,A
66
67     CALL DISP_RESULT
68  RET
```

Now we make the addition to get R1 to R4

R1 is just X1 (49)

R2 = X2+X3+X5 (52-55)

R3 = X4+X6+X7 (57-65) + carry from previous addition

R4 = X8 + carry from previous addition

Note that any carry generated during the addition process are saved in CR and restored into C before the next addition