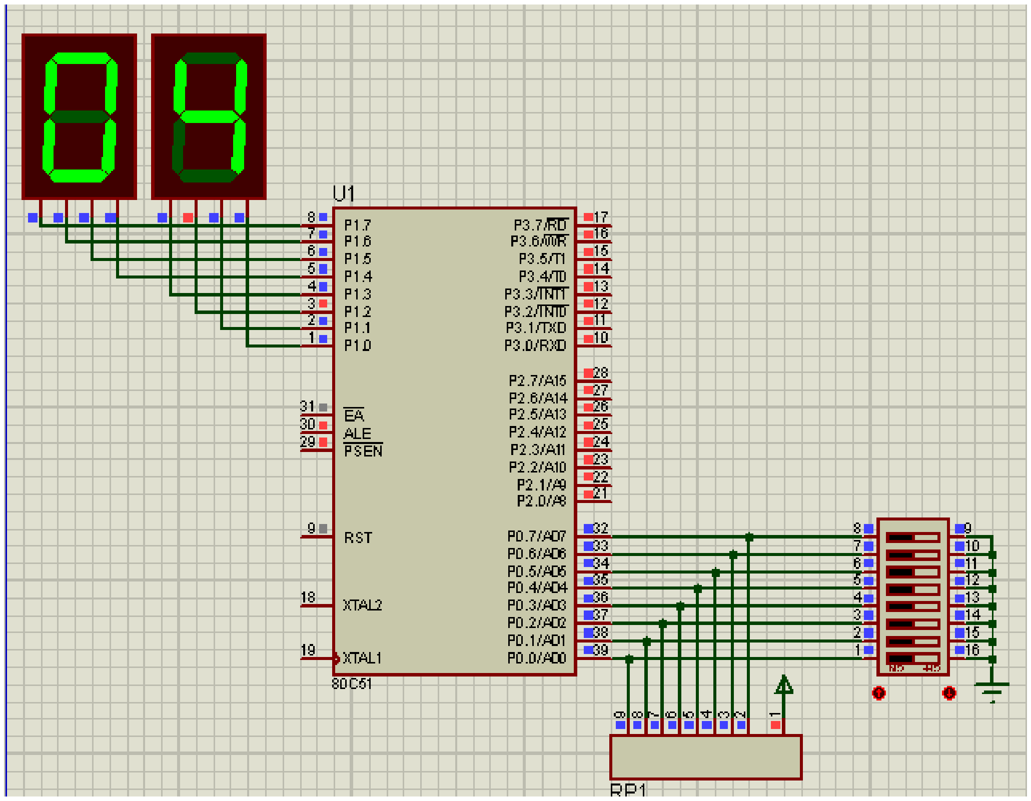
Problem 26



In this problem, we search for a given word in a list of words

Dip switches select the list to search within

P1 will display the index at which the word is found; if it is not found, it will display "EE"

Variables

```
1 INDEXW EQU 30H
2 INDEXL EQU 31H
3 INDEX EQU 32H
4
5 WSIZE EQU 34H
6 WSIZE2 EQU 35H
7 INDEXL2 EQU 37H
8
9 LIST_ADDRESS EQU 38H
```

1,2 → index for the word and the list

3→ the matched index in the list

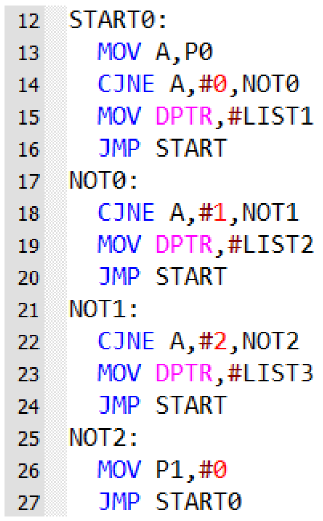
5→ word size

6→ wsize2 → used to store the size of the word in a list

7→ indexL2→ used as before in problem 25 to use the same match whole word function as before

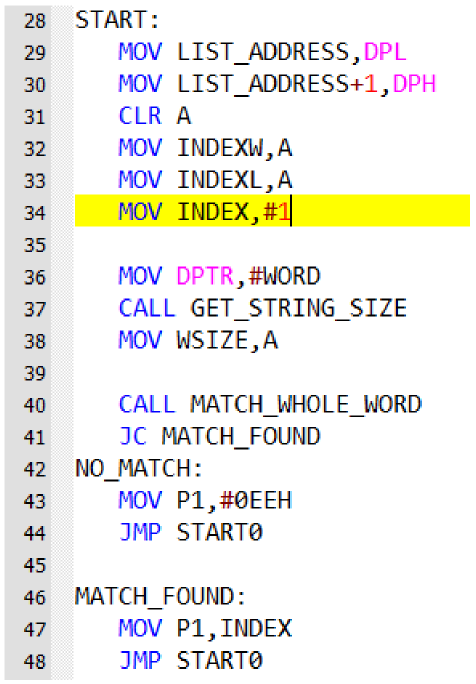
9→ list_address → point to the selected list

Main code



```
12 START0:
13   MOV A,P0
14   CJNE A,#0,NOT0
15   MOV DPTR,#LIST1
16   JMP START
17 NOT0:
18   CJNE A,#1,NOT1
19   MOV DPTR,#LIST2
20   JMP START
21 NOT1:
22   CJNE A,#2,NOT2
23   MOV DPTR,#LIST3
24   JMP START
25 NOT2:
26   MOV P1,#0
27   JMP START0
```

First, we read P0 to determine the required list to search within(12), then we make a comparisons with 0,1,2 as in lines (14,18,22) and load DPTR with the start address of the list (15,19,23), then we go to start



```
28 START:
29    MOV LIST_ADDRESS,DPL
30    MOV LIST_ADDRESS+1,DPH
31    CLR A
32    MOV INDEXW,A
33    MOV INDEXL,A
34    MOV INDEX,#1
35
36    MOV DPTR,#WORD
37    CALL GET_STRING_SIZE
38    MOV WSIZE,A
39
40    CALL MATCH_WHOLE_WORD
41    JC MATCH_FOUND
42 NO_MATCH:
43    MOV P1,#0EEH
44    JMP START0
45
46 MATCH_FOUND:
47    MOV P1,INDEX
48    JMP START0
```

Here, we load list_address with the 16 bit starting address of the list (29-30)

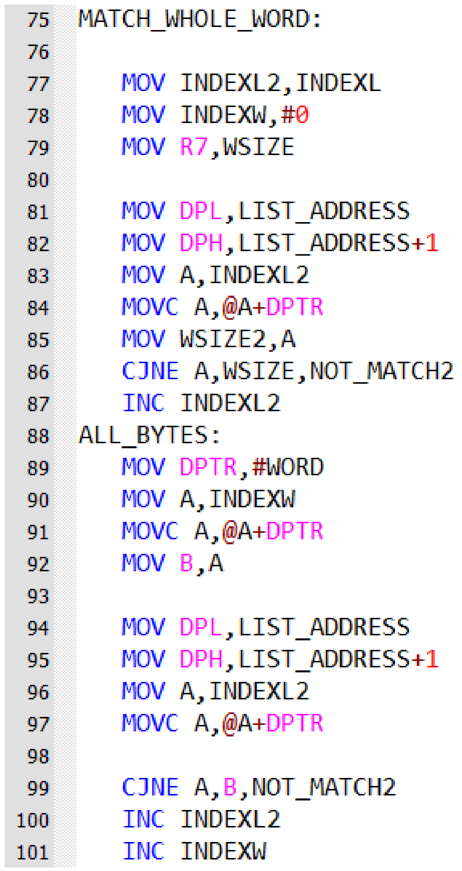
Clear word index, list index, but set the match index to 1 (32-34)

36-38 -> get word size

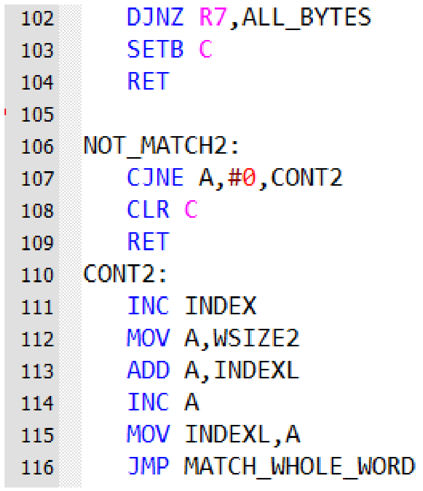
40→ call the function that compare all list entries with the word for a match

41→ if carry set → match found then we display the index on P1 (47) else

43→ display "EE"



```
 75 MATCH_WHOLE_WORD:
 76
 77     MOV INDEXL2,INDEXL
 78     MOV INDEXW,#0
 79     MOV R7,WSIZE
 80
 81     MOV DPL,LIST_ADDRESS
 82     MOV DPH,LIST_ADDRESS+1
 83     MOV A,INDEXL2
 84     MOVC A,@A+DPTR
 85     MOV WSIZE2,A
 86     CJNE A,WSIZE,NOT_MATCH2
 87     INC INDEXL2
 88 ALL_BYTES:
 89     MOV DPTR,#WORD
 90     MOV A,INDEXW
 91     MOVC A,@A+DPTR
 92     MOV B,A
 93
 94     MOV DPL,LIST_ADDRESS
 95     MOV DPH,LIST_ADDRESS+1
 96     MOV A,INDEXL2
 97     MOVC A,@A+DPTR
 98
 99     CJNE A,B,NOT_MATCH2
100     INC INDEXL2
101     INC INDEXW
```



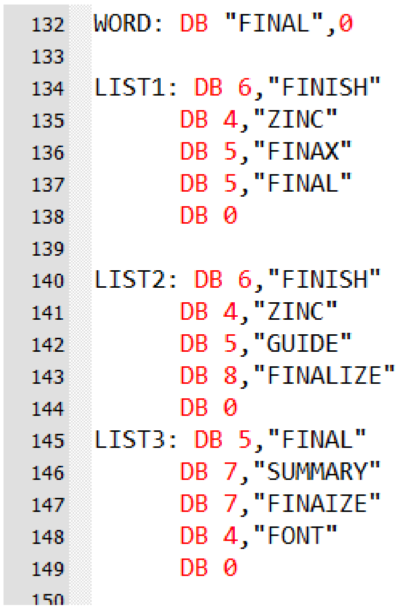
```
102     DJNZ R7,ALL_BYTES
103     SETB C
104     RET
105
106 NOT_MATCH2:
107     CJNE A,#0,CONT2
108     CLR C
109     RET
110 CONT2:
111     INC INDEX
112     MOV A,WSIZE2
113     ADD A,INDEXL
114     INC A
115     MOV INDEXL,A
116     JMP MATCH_WHOLE_WORD
```

We start by initializing all indexes (77-78) and R7 with word size (79)

81-86 → compare the our word size with the word size in list(indexL2); if they are not equal → no match →is this the last entry of the list (107)→ clear carry and return (108-109);else point to next list entry address by adding the current word size to indexL+1 and repeat matching process(111-116)

Note that the last word in the list has a length of '0' and this end the search (107)

Every entry starts by entry length as shown



```
132 WORD: DB "FINAL",0
133
134 LIST1: DB 6,"FINISH"
135       DB 4,"ZINC"
136       DB 5,"FINAX"
137       DB 5,"FINAL"
138       DB 0
139
140 LIST2: DB 6,"FINISH"
141       DB 4,"ZINC"
142       DB 5,"GUIDE"
143       DB 8,"FINALIZE"
144       DB 0
145 LIST3: DB 5,"FINAL"
146       DB 7,"SUMMARY"
147       DB 7,"FINAIZE"
148       DB 4,"FONT"
149       DB 0
150
```

List 1 has 4 entries; each entry start by its length (6,4,5,5). A '0' at line 138 indicates the end of the list.