How Many Square Feet are There?

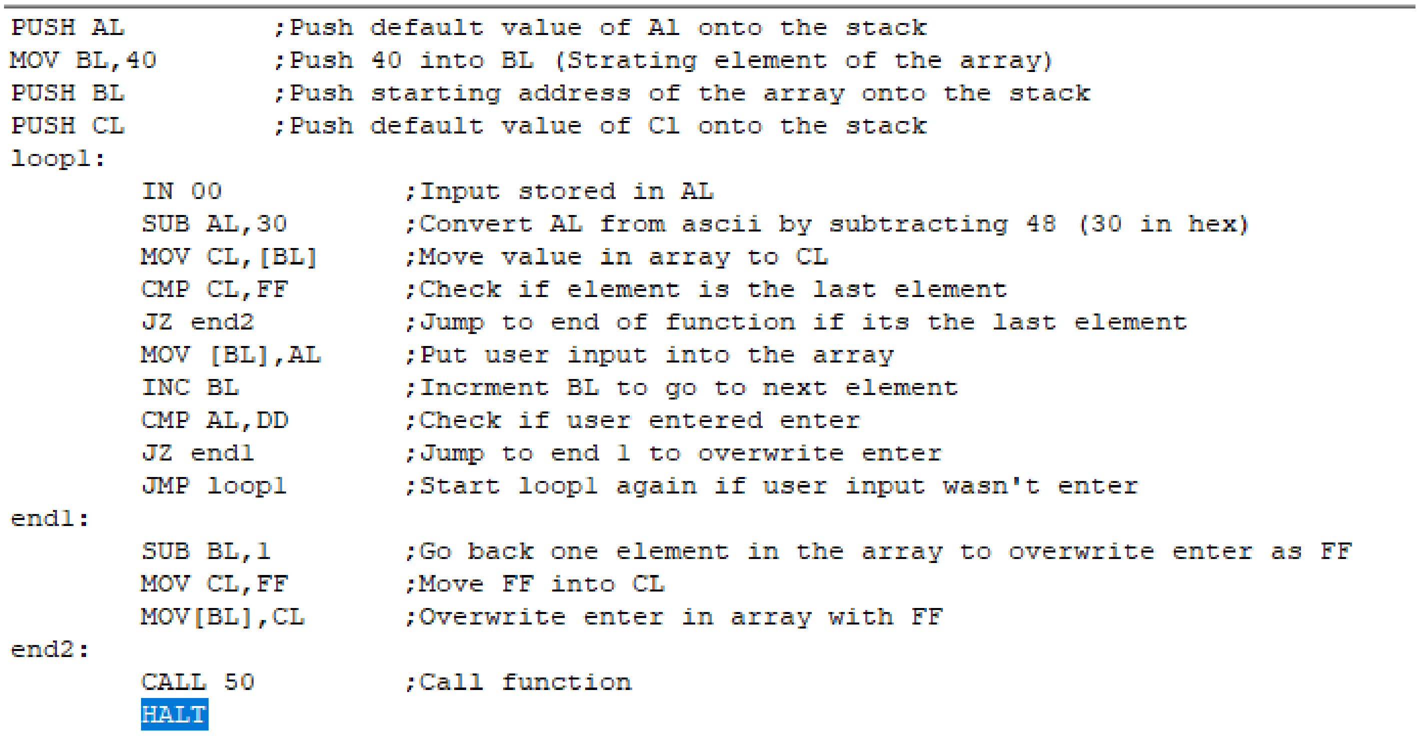
Changes:

- I added a loop to the main program to get input from the user for what the elements in the array are. The loop ends when the user presses enter or when the end of the array is reached.
- Because the registers are only 8 bits, the elements in the array (the room lengths in the problem) must be much smaller because the total number of square feet must be less than 255, which is the highest number that an 8-bit register can hold.
- Since the side lengths of the rooms in the problem have to be small and all of the rooms are square, this problem would make much more sense if the area being calculated was for cubicles in an office. Perhaps a new department is moving in and need to know how much space the required cubicles will take up in the office.

Explanation:

- Comments in the code explain its execution.

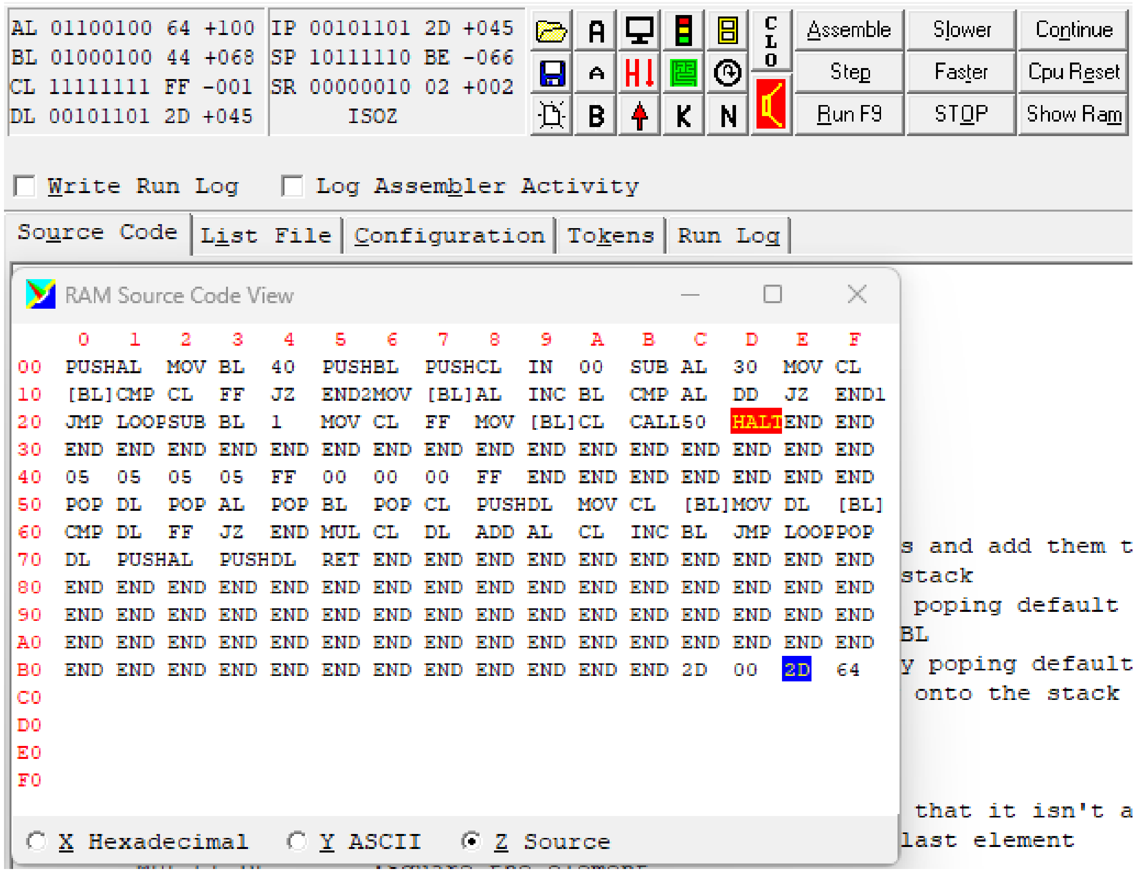
Screenshots:



```
PUSH AL         ;Push default value of Al onto the stack
MOV BL,40       ;Push 40 into BL (Strating element of the array)
PUSH BL         ;Push starting address of the array onto the stack
PUSH CL         ;Push default value of Cl onto the stack
loop1:
        IN 00           ;Input stored in AL
        SUB AL,30       ;Convert AL from ascii by subtracting 48 (30 in hex)
        MOV CL,[BL]     ;Move value in array to CL
        CMP CL,FF       ;Check if element is the last element
        JZ end2         ;Jump to end of function if its the last element
        MOV [BL],AL     ;Put user input into the array
        INC BL          ;Incrment BL to go to next element
        CMP AL,DD       ;Check if user entered enter
        JZ end1         ;Jump to end 1 to overwrite enter
        JMP loop1       ;Start loop1 again if user input wasn't enter
end1:
        SUB BL,1        ;Go back one element in the array to overwrite enter as FF
        MOV CL,FF       ;Move FF into CL
        MOV[BL],CL      ;Overwrite enter in array with FF
end2:
        CALL 50         ;Call function
        HALT
```

```
ORG 40                  ;Create array and initialize all elements with 0 and a max of 8 elements
DB 00
DB 00
DB 00
DB 00
DB 00
DB 00
DB 00
DB 00
DB FF                   ;Set FF as the end of the array
```

```
ORG 50                  ;Create a function to square elements and add them together
POP DL                  ;POP function return value from the stack
POP AL                  ;Reset AL for use in the function by poping default value from the stack
POP BL                  ;POP starting element of array into BL
POP CL                  ;Reset CL for use in this function by poping default value from the stack
PUSH DL                 ;Push the function return value back onto the stack so that it can return to the main program
loop2:
        MOV CL,[BL]     ;Move elemnt of array into CL
        MOV DL,[BL]     ;Move element of array into DL
        CMP DL,FF       ;Check if element is last element so that it isn't added to AL
        JZ end          ;Jump to end of function if its the last element
        MUL CL,DL       ;Square the element
        ADD AL,CL       ;Add the squared number to AL
        INC BL          ;Increment BL to go to next element
        JMP loop2       ;Restart loop2
end:
        POP DL          ;POP function return address into DL
        PUSH AL         ;PUSH AL (The total square feet) onto the stack
        PUSH DL         ;POP function return value from the stack
        RET             ;Return to main program
END
```



```
AL 01100100 64 +100   IP 00101101 2D +045
BL 01000100 44 +068   SP 10111110 BE -066
CL 11111111 FF -001   SR 00000010 02 +002
DL 00101101 2D +045           ISOZ
```

```
    0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F
00  PUSHAL  MOV BL  40  PUSHBL  PUSHCL  IN  00  SUB AL  30  MOV CL
10  [BL]CMP CL  FF  JZ  END2MOV [BL]AL  INC BL  CMP AL  DD  JZ  END1
20  JMP LOOPSUB BL  1   MOV CL  FF  MOV [BL]CL  CALL50  HALTEND END
30  END END END END END END END END END END END END END END END END
40  05  05  05  05  FF  00  00  00  FF  END END END END END END END
50  POP DL  POP AL  POP BL  POP CL  PUSHDL  MOV CL  [BL]MOV DL  [BL]
60  CMP DL  FF  JZ  END MUL CL  DL  ADD AL  CL  INC BL  JMP LOOPPOP
70  DL  PUSHAL  PUSHDL  RET END END END END END END END END END END
80  END END END END END END END END END END END END END END END END
90  END END END END END END END END END END END END END END END END
A0  END END END END END END END END END END END END END END END END
B0  END END END END END END END END END END END END 2D  00  2D  64
C0
D0
E0
F0
```

( ) X Hexadecimal   ( ) Y ASCII   [✓ Z Source]

```
s and add them t
stack
 poping default
BL
y poping default
 onto the stack

 that it isn't a
last element
```

This is the resulting RAM window and registers after running the program with a scenario of adding 4 5ftx5ft cubicles. The user inputs the number 5 4 times and presses enter the next time. The resulting area in square feet (100) is stored in AL and is pushed onto the stack inside the function.

<u>Code:</u>

```
PUSH AL                 ;Push default value of Al onto the stack
MOV BL,40     ;Push 40 into BL (Strating element of the array)
PUSH BL                 ;Push starting address of the array onto the stack
PUSH CL                 ;Push default value of Cl onto the stack
loop1:
          IN 00         ;Input stored in AL
          SUB AL,30     ;Convert AL from ascii by subtracting 48 (30 in hex)
          MOV CL,[BL]   ;Move value in array to CL
          CMP CL,FF     ;Check if element is the last element
          JZ end2                 ;Jump to end of function if its the last element
          MOV [BL],AL   ;Put user input into the array
          INC BL        ;Incrment BL to go to next element
          CMP AL,DD     ;Check if user entered enter
          JZ end1                 ;Jump to end 1 to overwrite enter
          JMP loop1     ;Start loop1 again if user input wasn't enter
end1:
          SUB BL,1      ;Go back one element in the array to overwrite enter as FF
          MOV CL,FF     ;Move FF into CL
          MOV[BL],CL    ;Overwrite enter in array with FF
end2:
          CALL 50                 ;Call function
          HALT
```

```
ORG 40                          ;Create array and initialize all elements with 0 and a max of 8
elements

DB 00

DB 00

DB 00

DB 00

DB 00

DB 00

DB 00

DB 00

DB FF                   ;Set FF as the end of the array


ORG 50                          ;Create a function to square elements and add them together

POP DL                          ;POP function return value from the stack

POP AL                          ;Reset AL for use in the function by poping default value from
the stack

POP BL                          ;POP starting element of array into BL

POP CL                          ;Reset CL for use in this function by poping default value from
the stack

PUSH DL                         ;Push the function return value back onto the stack so that it
can return to the main program

loop2:

        MOV CL,[BL]     ;Move elemnt of array into CL

        MOV DL,[BL]     ;Move element of array into DL

        CMP DL,FF       ;Check if element is last element so that it isn't added to AL

        JZ end          ;Jump to end of function if its the last element

        MUL CL,DL       ;Square the element
        ADD AL,CL       ;Add the squared number to AL
        INC BL          ;Increment BL to go to next element
        JMP loop2       ;Restart loop2
end:
        POP DL                  ;POP function return address into DL
        PUSH AL                 ;PUSH AL (The total square feet) onto the stack
        PUSH DL                 ;POP function return value from the stack
        RET             ;Return to main program
END
```