Lab 1 Report

This lab requires to write a LC-3 program in machine language to detect whether a 16-bit value has at least three consecutive '1's. If the value has at least three consecutive '1's, store value 1 in R2, otherwise store 0. To achieve the detection, we need to implement a sequence of AND instruction.

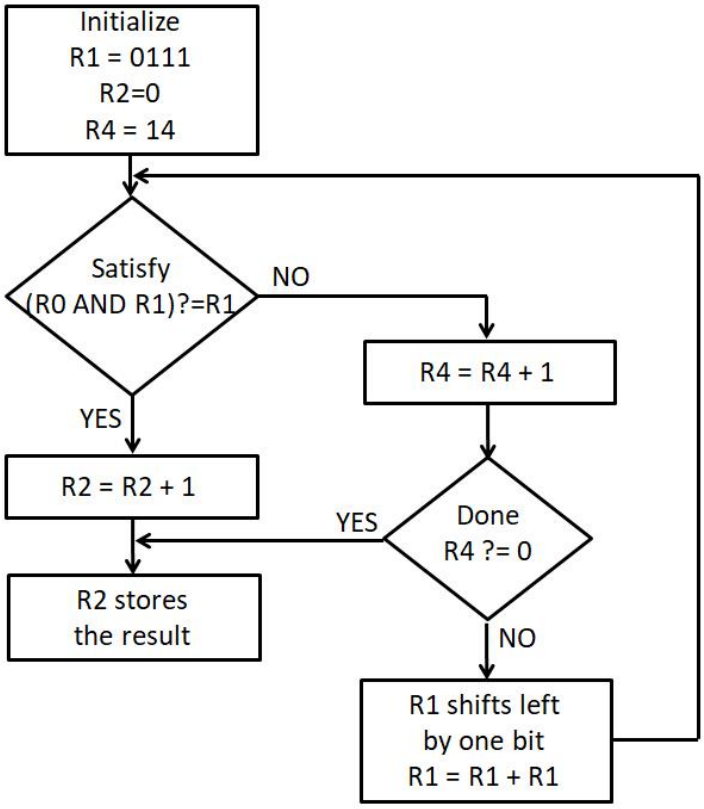
Algorithm

Since there are 14 cases in total of a 16-bit value having three consecutive '1's, we need to perform the AND instruction 14 times at most to finish the detection. So the algorithm is:

1. Get the value from memory;
2. Detect all the cases in a loop;
3. Check if all the cases have been detected;
4. Store the result in R2;

In my program, R0 stores the value to be detected. R1 signals the cases of having three consecutive '1's. R2 stores the result. R3 stores the results of R0 AND R1, and subsequently the results of R1 minus R3. R4 acts like a counter counting the times, which is used to check if all the cases have been detected.

The loop structure is shown as follow:



Code

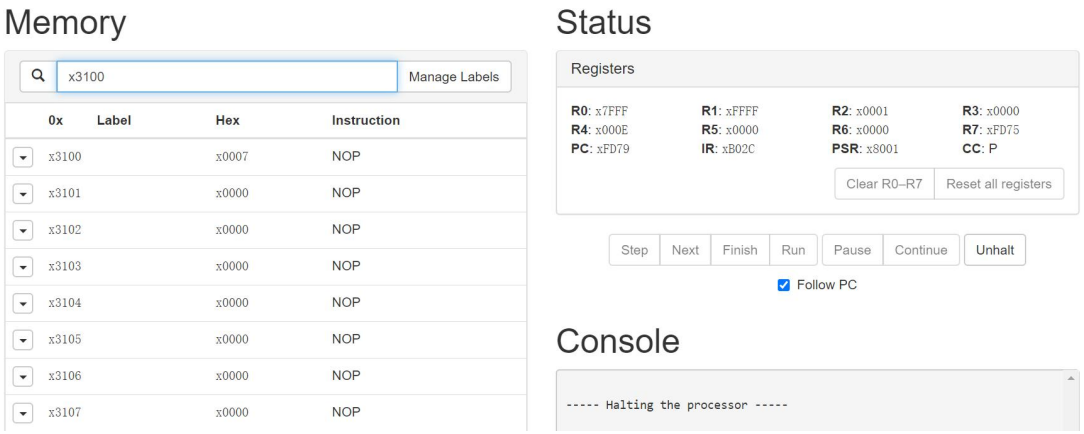
```
0011 0000 0000 0000     ; Not an instruction.
                        ; This line tells the simulator
                        ; to start the program from x3000.

0101 001 001 1 00000    ; clear R1
0101 010 010 1 00000    ; clear R2
0101 100 100 1 00000    ; clear R4
0010 000 011111100      ; load the value from x3100 into R0
0001 001 001 1 00111    ; set R1 to 0111
0001 100 100 1 01110    ; set R4 to 14

0101 011 000 000 001    ; check if the value has three consecutive '1's
1001 011 011 1 11111    ; compute the opposite number of the result
0001 011 011 1 00001    ; compute the opposite number of the result
0001 011 001 000 011    ; check if R1 and R3 are the same
0000 010 000000100      ; if the same, end the loop
0001 100 100 1 11111    ; decrease R4 by 1
0000 010 000000011      ; if 0, jump to the end
0001 001 001 000 001    ; add R1 to itself
0000 111 111110111      ; jump back to the loop

0001 010 010 1 00001    ; add 1 to R2
1111 0000 0010 0101     ; halt
```

Test Results



Memory

x3100 Manage Labels

0x	Label	Hex	Instruction
x3100		x0000	NOP
x3101		x0000	NOP
x3102		x0000	NOP
x3103		x0000	NOP
x3104		x0000	NOP
x3105		x0000	NOP
x3106		x0000	NOP
x3107		x0000	NOP

Status

Registers

R0: x7FFF	**R1**: xFFFF	**R2**: x0000	**R3**: xE000
R4: x0000	**R5**: x0000	**R6**: x0000	**R7**: xFD75
PC: xFD79	**IR**: xB02C	**PSR**: x8001	**CC**: P

Clear R0–R7 | Reset all registers

Step | Next | Finish | Run | Pause | Continue | Unhalt

Follow PC

Console

```
----- Halting the processor -----
```