# Assignment Report 1

## 1. Assignment

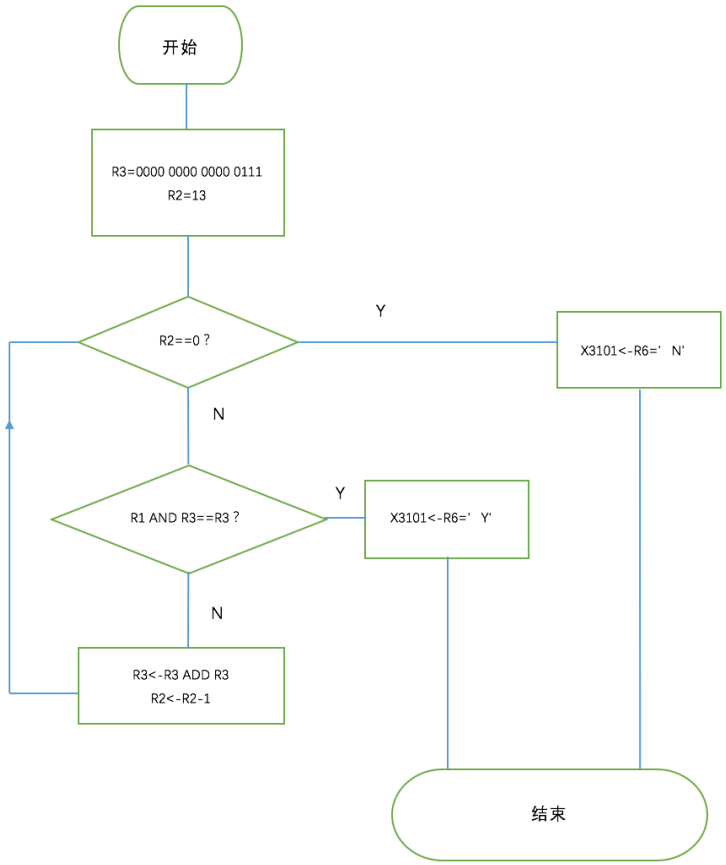
*A 16-bit value is stored in memory location x3100 of the LC-3. Your task is to figure out whether or not the 16 bits contain at least three consecutive 1's.*
*If so, you are to store the ASCII code for Y in memory location x3101, otherwise, put the code for N at the same location.*
*For example, if x3100 contains: 0110000110001101, you should store the ASCII code for N in x3101.*
*If the memory content is 0000001101111000, ASCII code for Y should be put in x3101.*
*Please write your program in machine code, and submit your program in binary text.*

## 2. Solution and Flow Chart

First of all, I read from location x3100 into Reg1.Then get the ASCII number 'N' from x3015, which is set behind the HALT. Now it' s clear that R1 has the data that needs to be examined whether it has at least three consecutive 1' s and R6 will have the result of 'Y' or 'N' . Now if the following instructions found that the data is satisfied with the request, R6 would change from 'N' into ' Y' . Then I set R2 to R5 clear as 0. Because if I need to run the program again in the simulator, the R2-R5 would not be 0. So it' s necessary to set it 0.

Then it' s the key instructions: First, I set R2 as a count down that examine 14 times of the 16-bit-long binary data. Second, I set R3 as 0000 0000 0000 0111, then, R3 and R1 do the AND instruction and store the result in R4, if R1 has 111 in the last three positions, which satisfies the assignment' s request, then R4 would be the same as R3, so I store R3 AND R4 into R5. Then I use BRz instruction to take a branch that store R6 as 'Y' in x3101, then program halts. If R1 doesn' t have 111 at the last three positions, then R2<-R2-1, R3 doubled into 0000 0000 0000 1110, and the loop goes on, examine the bit[3:0] , and the loop goes on until it satisfies the request or R2 is count down to 0 and the program

ends and the x3101 is still ‘N’ .




## 3. Source Code

```
0011 0000 0000 0000 ;.ORIG x3000
1110 001 011111111 ;LEA R1,x3100
0010 110 000 010100 ;LD R6, ASCII
0110 001 001 000000 ;LDR R1,R1,#0
0011 110 011111101 ;ST R6,x3101
0001 110 110 1 01011 ;ADD R6,R6,#11   set R6 as the ASCII code of 'Y'
0101 010 010 1 00000 ;AND R2,R2,#0
0101 011 011 1 00000 ;AND R3,R3,#0
0101 100 100 1 00000 ;AND R4,R4,#0
0101 101 101 1 00000 ;AND R5,R5,#0   clear the registers
0001 011 011 1 00111 ;ADD R3,R3,#7   set the compare number R3
0001 010 010 1 01110 ;ADD R2,R2,#14   set the countdown number R2
0000 010 000001001 ;BRz   the loop of countdown
0101 100 001 000 011 ;AND R4,R1,R3
1001 100 100 111111 ;NOT R4,R4
0101 101 011 000 100 ;AND R5,R3,R4   if R3 AND R4 =0 then take a branch of write 'Y' in x3101
0000 001 000000010 ; BRp
0011 110 011110000 ;ST R6,x3101
1111 0000 00100101 ;HALT
0001 011 011 000 011 ;ADD R3,R3,R3  R3 doubled up
0001 010 010 1 11111 ; ADD R2,R2,#-1 R2 countdown
0000 111 111110110 ;BRnzp
1111 0000 00100101 ;HALT
0000 0000 0100 1110 ;.ASCII x4E
```

0011 0000 0000 0000 ;.ORIG x3100

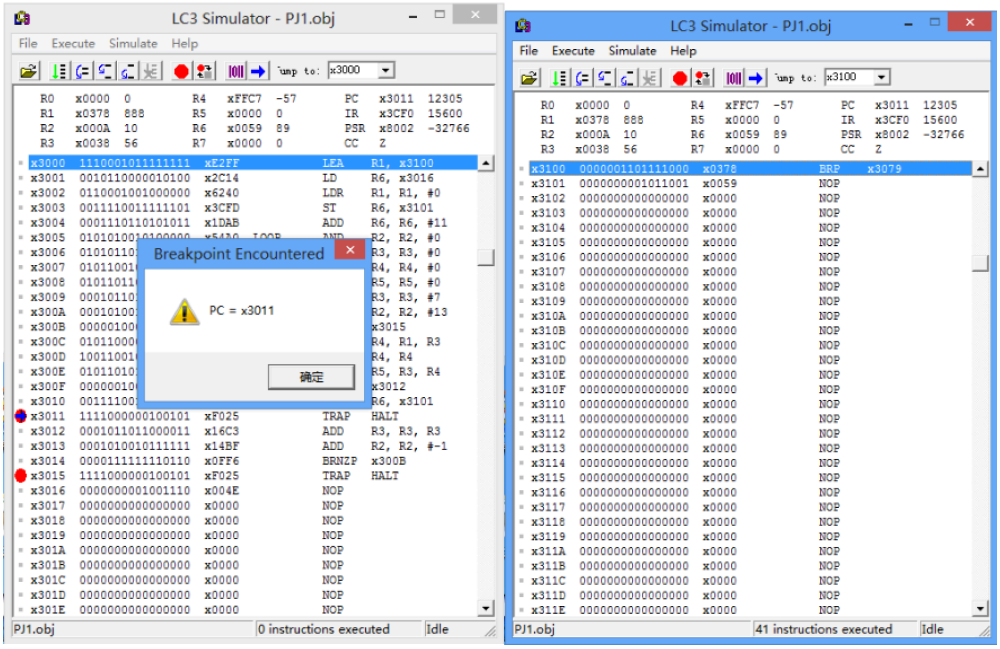
1110 001 011111111 ;LEA R1,x3100

0010 110 000 010100 ;LD R6, ASCLL

```
0110 001 001 000000 ;LDR R1,R1,#0

0011 110 011111101 ;ST R6,x3101

0001 110 110 1 01011 ;ADD R6,R6,#11      set R6 as the ASCII code of 'Y'

0101 010 010 1 00000 ;AND R2,R2,#0

0101 011 011 1 00000 ;AND R3,R3,#0

0101 100 100 1 00000 ;AND R4,R4,#0

0101 101 101 1 00000 ;AND R5,R5,#0      clear the registers

0001 011 011 1 00111 ;ADD R3,R3,#7      set the compare number R3

0001 010 010 1 01110 ;ADD R2,R2,#14      set the countdown number R2

0000 010 000001001 ;BRz      the loop of countdown

0101 100 001 000 011 ;AND R4,R1,R3

1001 100 100 111111 ;NOT R4,R4

0101 101 011 000 100 ;AND R5,R3,R4      if R3 AND R4 =0 then take a branch of

write 'Y' in x3101

0000 001 000000010 ; BRp

0011 110 011110000 ;ST R6,x3101

1111 0000 00100101 ;HALT

0001 011 011 000 011 ;ADD R3,R3,R3    R3 doubled up

0001 010 010 1 11111 ; ADD R2,R2,#-1 R2 countdown

0000 111 111110110 ;BRnzp

1111 0000 00100101 ;HALT

0000 0000 0100 1110 ;.ASCII x4E
```

## 4. Snapshots

### Example1：test number：0000 0011 0111 1000 result： ‘Y’




### Example2：test number：0110 0001 1000 1101 result： ‘N’