

实验一

一、源代码：

```
1: 0011000000000000
2: 0101101001100000;x3000 R5=0 (flag~)
3: 0010001011111110;x3001 R1= n
4: 0101010001100001;x3002 R2=R1&1 odd or even(IF even ,R1=0)
5: 0000010000001100;x3003 if even 跳 (18)
6: 0001011010100000;x3004 R3=R2=1
7: 0101100011000001;x3005 R4 = R1&R3
8: 0000101000000001;x3006 if R3&R1 >0 (就是这次判断的位置是 1) goto (10)
9: 0001101101100001;x3007 ADD R5=R5+1 (不是 1 就是零喽, 是零就 flag++)
10: 0001011011000011;x3008 ADD R3=2*R3 (R3*2,判断是否是 1 的位置左移一位)
11: 0001010010100001;x3009 ADD R2=R2+1 (计数器, 一共判断 16 次)
12: 0001110010110000;x300A ADD R6=R2-16
13: 0000110111111001;x300B if R6<=0 跳 (7)
14: 0010001011110100;x300C R1=id
15: 0001001001000101;x300D R1=R1+R5
16: 0011001011110011;x300E 存入目标位置
17: 1111000000100101;HALT
18: 1001001001111111;NOT R1 (取反)
19: 0001001001100001;ADD R1=R1+1 (取反加一得到负补码)
20: 0101010010100000;AND R2=0
21: 0001010010100001;ADD R2=R2+1
22: 0000111111011111;跳转 (6)-17 10001 01111X3004: 0000 010 0 0000 1110
```

二、实验结果

The image shows two screenshots of the LC3Tools emulator interface. The top screenshot shows the initial state of the registers and memory. The bottom screenshot shows the state after some instructions have been executed.

Registers			Memory		
R0	x0000	0	x3100	x0005	5
R1	x7FFF	32767	x3101	x0008	8
R2	x0011	17	x3102	x0016	22
R3	x0000	0	x3103	x0000	0
R4	x0000	0	x3104	x0000	0
R5	x000E	14	x3105	x0000	0
R6	x2FFE	12286	x3106	x0000	0
R7	x0000	0	x3107	x0000	0

Registers			Memory		
R0	x0000	0	x3100	x0064	100
R1	x7FFF	32767	x3101	x0008	8
R2	x0011	17	x3102	x000C	12
R3	x0000	0	x3103	x0000	0
R4	x8000	32768	x3104	x0000	0
R5	x0004	4	x3105	x0000	0
R6	x2FFE	12286	x3106	x0000	0
R7	x0000	0	x3107	x0000	0