Lab1 Lucky 111

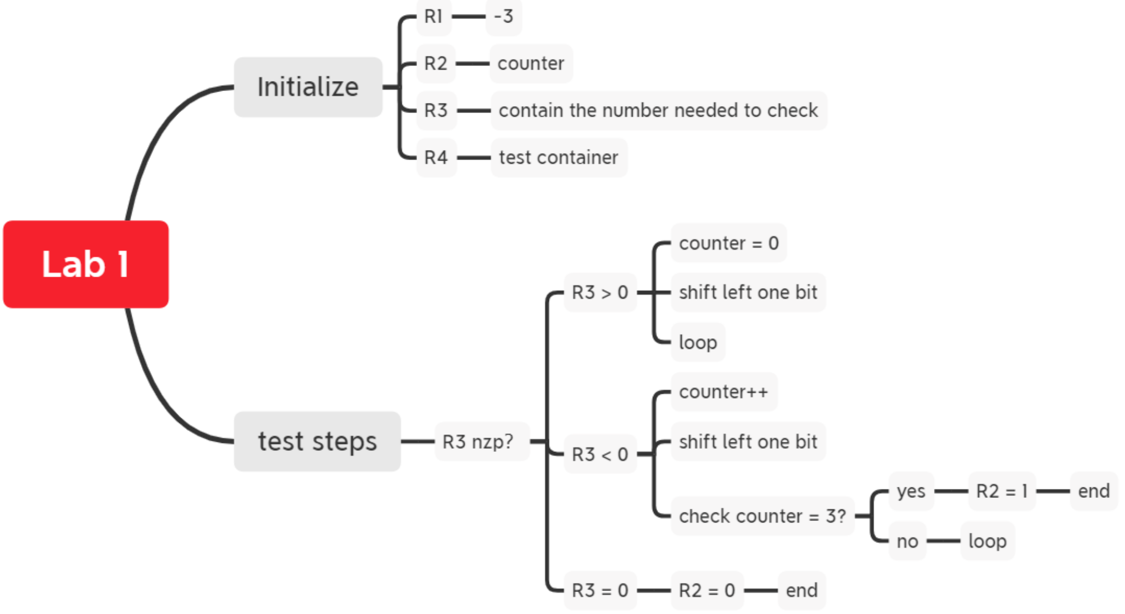

figure.1 flow chart

1. Initialization

R3 saves the number needed to be checked.
R2 saves the counter that counts the number of sequential 1. And it is set to 0 at first.
R1 is -3, which is used as a constance to check if R2 is 3.
R4 is a container to calculate R1 + R2.

2. algorithm

Check if R3 > 0 or R3 < 0, or R3 = 0.
(1)if R3 = 0, that means there is no 1 in R3, and then we can conclude R2 = 0 and end the program.
(2)if R3 > 0, that means the first figure of R3 is 0, so the counter should be set to 0.
(3)if R3 < 0, that means the first figure of R3 is 1, so the counter should be set to counter + 1. And we will check if counter is 3. If counter is 3, we know we get three sequential 1, in that case R2 = 1 and we can end the program. If counter is not 3, loop.
(4) For (2)(3) case, we check the first figure of R3, and then we need to check the next figure of R3. To get that, we can shift R3 left one bit and loop to check. And that can be achieved by let R3 = R3 + R3.

3. specific codes

```
    ;initialization
0011 0000 0000 0000;orig
0101 010 010 100000;R2<-0
0101 001 001 100000;R1<-0
0001 001 001 111101;R1<--3
0010 011 011111100;R3<-X3100
    ;check R3, check module
0001 011 011 100000;R3 = R3 + 0, to set nzp.
```

```
0000 010 000001001;if R3 is 0, jump to end and set R2 to 0.
0000 001 000000101;if R3 is positive,jump to R3 > 0.
    ;R3 < 0
0001 011 011 000011;R3 = R3 + R3, shift left one bit
0001 010 010 100001;R2<- R2 + 1
0001 100 001 000010;R4 = R2 + R1
0000 010 000000110;if R4 = 0,end and set R2 to 1
0000 111 111111000;loop
    ;R3 > 0
0101 010 010 100000;R2<-0
0001 011 011 000011;R3 = R3 + R3, shift left one bit
0000 111 111110101;loop
    ;set R2 to 0
0101 010 010 100000;R2<-0
1111 0000 00100101;halt
    ;set R2 to 1
0101 010 010 100000;R2<-0
0001 010 010 100001;R2<-1
1111 0000 00100101;halt
```