ICS Lab1

Yunfan Li

3200102555

Algorithm

I use `0000 0000 0000 0111` as the start value `i` and make it AND the number N, if there are 3 continuous 1s, `halt` and set `r2` to be 1. If not, move `i` left and compare it with N again. After looping 14 times, `i` becomes `1110 0000 0000 0000`, if it failures again, that is to say every 3 continuous bits in N is not `111` so `halt` and set `r2` to be 0.

```
r1:= 0000 0000 0000 0111
r0:= 13//as cnt
r4:= N
r5:= r1 AND r4
while r0>=0:
  if r5==r1:
    N is a F-word
  else:
    r1<<
    r0--
    r5=r1 AND r4
    continue
N is not a F-word
```

Judgement of `==`

To realize the judgement of `r5 == r1`, we compute `r5-r1==0`.

To compute `r5 - r1`, we first `NOT r3 r5` then `ADD r3 r3 #1` so `r3 = -r5`. Finally we compute `ADD r6 r3 r1`, if `r6 ==0` then `r5==r1`, that is to say, there are 3 continuous 1s in N.

Realization of `<<`

Just simply `ADD r1,r1,r1`, i.e. `r1*2`.

Code

```
0011 0000 0000 0000;  .ORIG x3000
0010 1000 1111 1111;    LD R4, x3100
0101 0000 0010 0000;    AND R0, R0, #0  ;initialization
```

```
0101 0010 0110 0000;    AND R1, R1, #0
0101 0100 1010 0000;    AND R2, R2, #0
0101 0110 1110 0000;  AND R3, R3, #0
0101 1011 0110 0000;    AND R5, R5, #0
0101 1101 1010 0000;    AND R6, R6, #0
0001 0000 0010 1101;    ADD R0, R0, #13 ;r0=13 for counting
0001 0010 0110 0111;    ADD R1, R1, #7 ;r1=0000 0000 0000 0111
0101 1011 0000 0001;    AND R5, R4, R1  ;r5=r4 AND r1
1001 0111 0111 1111;    NOT R3, R5          ;judge if r5==r1
0001 0110 1110 0001;  ADD R3, R3, #1
0001 1100 1100 0001;  ADD R6, R3, R1
0000 1010 0000 0001;  BRnp x300F      ;if r5!=r1->jump to loop
0000 0100 0000 0111;  BRz x3016       ;if r5==r1->jump to success
0001 0010 0100 0001;    ADD R1, R1, R1  ;r1<<
0001 0000 0011 1111;    ADD R0, R0, #-1 ;cnt--
0000 1000 0000 0010;    BRn x3014       ;if cnt<0 jump to failure
0101 1011 0000 0001;    AND R5, R4, R1  ;r5=r4 AND r1
0000 1111 1111 0110;    BRnzp x300A     ;jmup to judge
0001 0100 1010 0000;    ADD R2, R2, #0  ;failure,set r2=0
1111 0000 0010 0101;    HALT
0001 0100 1010 0001;    ADD R2, R2, #1  ;success,set r2=1
1111 0000 0010 0101;    HALT
```

Check questions

Q: How many times did your algorithm loop and Why?

A: 14 times, because from `0000 0000 0000 0111` to = `1110 0000 0000 0000` there exist 14 situations, so I need to loop 14 times.