Lab4 Report

1. algorithm explanation

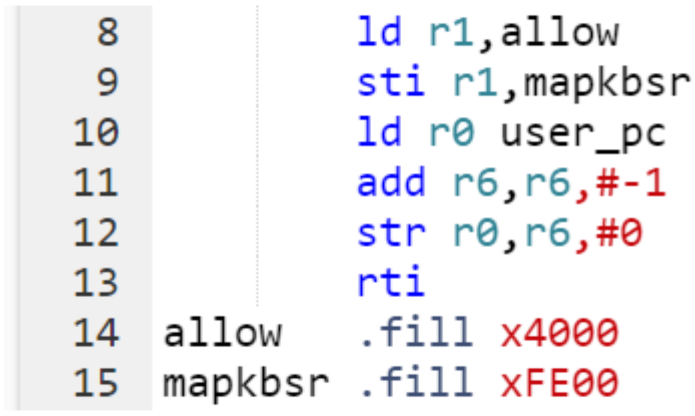
First, we need to set the 14th bit of KBSR, which we can only do with privilege, so we plan to do this in os segment x0200, where originally the turning up codes are stored. We copy the codes and insert the code to set the 14th bit of KBSR in it.

Second, after the 14th bit of KBSR is set, we need to input the first line of input by getc, but we need to stop the interrupt to happen because if it happens here, it will disturb our input. So we set R7 to #723 when we do the input, and if the interrupt test the R7 and find it is #723, it won’ t do anything. We use R1 to count how many chars we have read ,for we will read 20 chars in total. And R4 will store the last index where the char we read is a letter but not a dot. R5 will store the last letter we read ,so it is the body of the bird temporarily. After the input, we will subtract R4 with 2, so it is the first index of the body of the bird.

Third, we will delay for a while and move the bird forward by one index and out put it in a new line. To realize the delay, we will do x10000 subtractions. To move the bird forward for one index we will subtract R4 with 1, and if it is less than 1, we will set it to 1.In the output procedure, we will print 17 dots and 3 letters, if the index of the out put is equal to r4, we will output three letters.

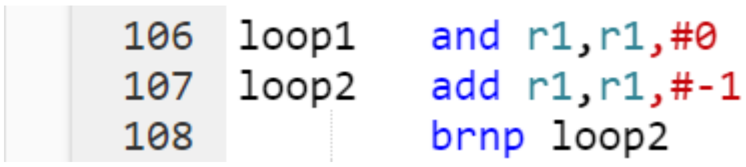
Fourth, if during the delay, we hit the character with 1-9 or a-z, we will interrupt the program and go to x2000 to execute our interrupt, if it is a number we will add it to r4, and set x4000, which means we have increase r4, so we will skip the decrease r4 part once. If after the increasing, r4 will larger than 18, we will set it to 18. If it is a letter, then we will cover r5 with the letter.

2. essential codes



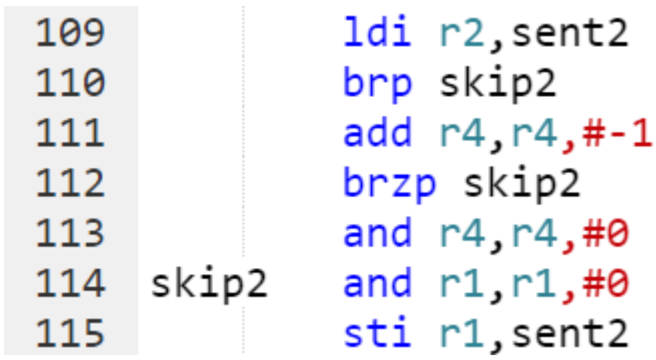
```
        ld r1,allow
        sti r1,mapkbsr
        ld r0 user_pc
        add r6,r6,#-1
        str r0,r6,#0
        rti
allow   .fill x4000
mapkbsr .fill xFE00
```

Here we make the 14th bit of KBSR 1.



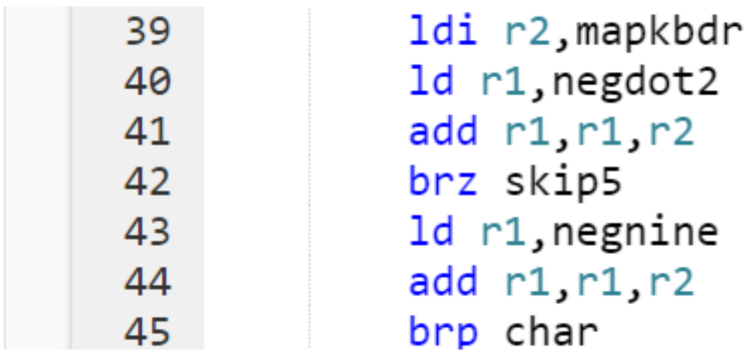
```
loop1   and r1,r1,#0
loop2   add r1,r1,#-1
        brnp loop2
```

Here is the delay loop, where we will do x10000 subtractions.



```
        ldi r2,sent2
        brp skip2
        add r4,r4,#-1
        brzp skip2
        and r4,r4,#0
skip2   and r1,r1,#0
        sti r1,sent2
```

Here we first load x4000 in r2 to test if an interrupt has happened. If an interrupt has happened, we will not decrease r4. And we will clear x4000 after the skip.



```
ldi r2,mapkbdr
ld r1,negdot2
add r1,r1,r2
brz skip5
ld r1,negnine
add r1,r1,r2
brp char
```

Here inside the interrupt, we are judging whether the char is a dot, if it is a dot, we won’ t do anything. Then we need to check if is larger than ’ 9’ , which will determine whether we will change r5 or change r4.

3. TA questions

Q: what if I hit the keyboard fast enough so that the second hit happens when the first interrupt is executing?

A: An interrupt happens if and only if the three conditions are satisfied at the same time, so when the second hit happens, the 15th bit of KBSR will be set, and the 14th bit of KBSR has been set long ago, but if we are executing another interrupt of keyboard simultaneously, the priority of both is the same. So the second interrupt will not happen immediately. But after the first interrupt is over. Control is handed back to our program. During the fetch phase, all the three conditions is satisfied and the INT is set, so in that phase the control will be handed to the next interrupt and go on executing,

Q: What if the hit happens during the code cycle of an instruction? For example, an add instruction.

A: The interrupt will not happen immediately. Actually, it won’ t happen until the fetch phase of the next instruction. Where we will check the INT and find that there is an interrupt waited for us to handle.