Lab 4 Report

1 Introduction

In this lab, we will write a program that implements the bird in the game *Flappy Bird*, which consists the system booting code, the interrupt service routine and the user program.

2 Solution

In the system booting code, we will store the location of interrupt service routine x2000 into the start address of the program that handles keyboard interrupts x180.

In the interrupt service routine, we will store the letter into the location that stores the letter or we will renew the height the bird flies.

In the user program, we will loop through the air "." and the bird an infinite number of times.

2.1 Algorithm

The algorithm of looping through the air "." and the bird an infinite number of times is as followed.

```
procedure Flappy_Bird(*cha, *heig)
b := *heig - 1// the number of "." below the bird
if b < 0
a := 17 - b// the number of "." above the bird
*heig := b
for i := b to 1 do
    output "."// the air below the bird
for i := 3 to 1 do
    output *cha// the bird
for i := a to 1 do
    output "."// the air above the bird
output 10// go to the next line
Flappy_Bird(*cha, *heig)
```

The procedure is in a infinite loop. When a character is inputted, the program interrupt and modify the value of *cha or *heig.

2.2 Essential part of the code

The first part is in the system booting code which modifies the value at the x0180 address to the start address of our own keyboard interrupter. Here, we set the start address to x2000.

```
        LD R0, INKB; x0180
        LD R1, KBI; the start address of keyboard interrupt program is x2000
        STR R1, R0, #0
INKB    .FILL x0180
KBI     .FILL x2000
```

The second part is the interrupt service routine. In this part, we first check whether the input character is a letter or a number. Then we jump to the corresponding part. We subtract 58 to the ASCII code of the input. If it's a number the outcome would be negative, or it's a letter.

```
        .ORIG x2000
        ST R0, SAVE0; save the original value of R0
        ST R1, SAVE1; save the original value of R1
        LDI R0, KBDR; R0 get the input ASCII code
        LD R1, CHE
        ADD R1, R1, R0; check whether the input character is a number or
letter
        BRn NNUM; if it's a number
        STI R0, LET; LET contains the letter to become
        BR ENDI; done
NNUM    LD R1, NUM
        ADD R0, R0, R1; R0 contains the number input
        LDI R1, HEIG
        ADD R0, R1, R0; R0 contains the height plus 1 after the number's
input
        LD R1, MAX
        ADD R1, R1, R0; check whether it's too high
        BRp LIM; if it's too high
        STI R0, HEIG; if if's not, store the outcome
ENDI    LD R0, SAVE0
        LD R1, SAVE1
        RTI
LIM     LD R0, MAXI
        STI R0, HEIG; set the HEIG to the maximum of height
        BR ENDI
KBDR    .FILL xFE02
CHE     .FILL #-58
NUM     .FILL #-47
MAX     .FILL #-18
MAXI    .FILL #18
LET     .FILL x5000
HEIG    .FILL x5001
SAVE0   .FILL #0
SAVE1   .FILL #0
        .END
```

3 Q & A

> Q: How can you know the priority of the program that handles keyboard interrupts?

A: We set a breakpoint at x2000. When the program stops, we will see the value of PSR in the chart Resisters. Since the priority stores in the bit[10, 8] in the PSR, we only need to check the value of this three bits and then we know the priority of it.

> Q: What if an illegal character is inputted?

A: If the ASCII code of the character is larger than 58, the shape of the bird will become the character inputted. Here we represent the ASCII code with c. If it's smaller than 58, c must be smaller than 47, which is the ASCII code of 0. The height of the bird will be subtracted 47-c. If the height after the operation is negative, we reset it to 0.