ICS Lab Report #4

StuID: **Name:**

Problem Setting

Here we are using the LC3 assembly language to implement a quite simplified "flappy bird" game. A bird flying in the air is represented by 3 consequent characters in some points. Player can increase the height of the bird by pressing number 0-9, or change the representation of the bird by pressing a-z.

Algorithm Specification

The program is split into 3 parts, however knowing the interrupt part and user program would be enough to understand.

Interrupt part is run by pressing any key on keyboard. Once called, height and appearance of the bird would be adjusted.

```
R0 <- KBDR
if R0 is alphabet
  print <- R0
else if R0 is number
  loc <- loc + R0 - '0'
  if loc > 18
    loc = 18
```

Program part is used to print on screen.

```
while(true)
  for i = 0 to 19 step 1
    if i == loc
      output(print) * 3
      i <- i + 2
    else
      output('.')
  wait()
```

LC3 Implementation

System booting code segment should set keyboard interrupt enabled, and replace the keyboard input trap vector with our trap routine address.

```
LD R6, OS_SP
LD R0, USER_PSR
ADD R6, R6, #-1
STR R0, R6, #0
LD R0, USER_PC
ADD R6, R6, #-1
STR R0, R6, #0 ;save SP and PSR
LD R0, SETBIT
LDI R1, KBSR
NOT R1, R1
NOT R0, R0
AND R0, R0, R1
NOT R0, R0 ;set enable
STI R0, KBSR
LD R0, INTADDR
LD R1, ALTADDR ;replace trap vector
STR R1, R0, #0
RTI
```

Trap routine deal with keyboard interrupt. Save and restore stack is omitted.

```
LDI R0, KBDR
LD R1, N_BORDER
ADD R1, R0, R1
BRN NUMBER
STI R0, INTPRINT ;if alphabet
BRNZP RETINT
NUMBER
LD R1, N_48 ;if number
ADD R0, R0, R1
LDI R1, INTLOC
ADD R1, R1, R0
LD R0, N_18
ADD R0, R0, R1
BRN NONSHRINK
```

```
AND R1, R1, #0 ;if greater than 18
ADD R1, R1, #9
ADD R1, R1, #8 ;shrink to 18
NONSHRINK
ADD R1, R1, #1 ;for next main loop to decrease
STI R1, INTLOC
```

Main program is easy to implement using the pseudocode, so here we ignore it.

Check Problem

Q: How do the keyboard interrupt signal produced?

A: When pressing a key, the input logic will set KBDR[16]. The signal is produced by BITAND KBDR[16] and KBDR[15].