**Lab5 实验报告**

*PB21071416 张郑飞扬*

*2022.12.31*

一、实验目的（Purpose）

利用基于LC-3的汇编语言，设计一个包含I/O中断功能的递归求汉诺塔最少步数的程序。

在未接受到输入时，要求循环输出“学号 学号 学号 ……”

要求对每一个输入及时响应，也就是说，汉诺塔层数N合法输入仅为0~9，对合法输入，输入要保存在x3FFF位置，并且要在显示屏上输出：“*<the input character> is a decimal digit*.”，并输出结果：“ *Tower of hanoi needs <output> moves.*”；对不合法输入，要在显示屏上输出*“<the input character> is not a decimal digit.”*

注：操作系统使能代码已给出，需要基于给出的代码，完成用户程序和键盘中断处理程序两部分内容。

二、实验原理

1. How do you work out the algorithm?

递归汉诺塔算法的实现较为简单，我们通过推导已经知道了汉诺塔的递推表达式：H(N) = 2H(N-1) + 1。那么，类似于书上Figure8.18对斐波那契数列的递归，我们同样可以设计对汉诺塔数列的递归，明确条件：递归出口为N=0，返回值为H(0)=0  
递归中间步骤为 N-1继续递归，返回值为 2H(N-1) + 1。

为了实现递归，需要开辟一个用户栈，本程序将栈底设置在x7000位置，每次递归压栈，当达到递归出口后弹栈。并且R0初始时R0应保存N值，最终R1输出结果。每次压栈三个数据：R7(保存返回链接)，R0（保存N），R2（子函数中使用）

2.How do you write the program?

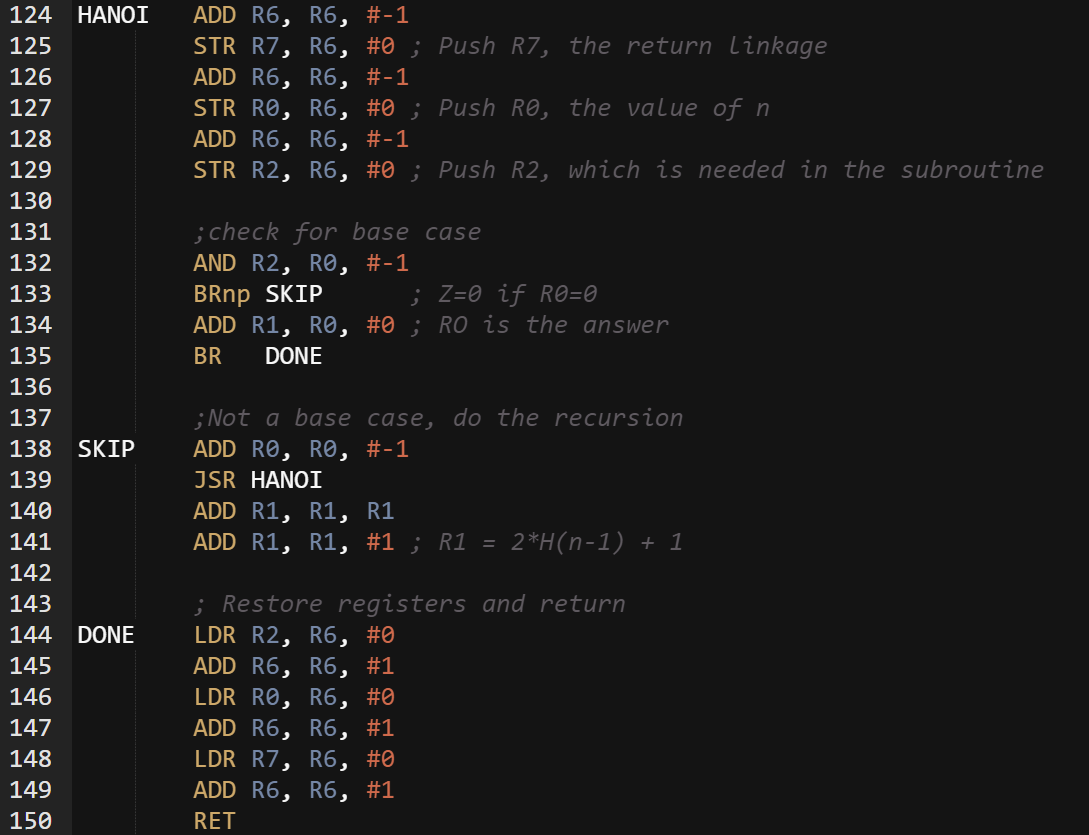
该程序分为两部分，用户程序和键盘中断处理程序。

先来看用户程序：

1）用户程序首先需要在未接受键盘输入时循环输出学号，这比较简单，用.STRINGZ设置好字符串，用PUTS指令循环输出即可（详见程序中的AGAIN部分）。需要注意，本程序设置了一个FLAG，用于标识输入是否合法，当检测到合法输入，就需要跳出默认循环，转到HANOI子函数中去计算了。

2）为了便于人的肉眼观察，需要控制输出速度，这就需要设置一个DELAY函数，其原理是内置一个计数器，当计数器达到一定数后再输出。如本程序中COUNT为5000，在输出前JSR到子函数DELAY，就可以实现经过5000个周期后输出一次，控制了输出速度。

3）当检测到合法输入（已被存储到x3FFF中），就需要调用HANOI子函数来递归计算得到结果。HANOI函数的原理在上面已经阐明。下面给出代码：



4）输出结果也需要一定处理。HANOI函数调用完成后，结果被储存在R1中，然而，R1中的值并不能直接输出。输出时，需要一位一位地用ASCII码输出。因此，我们需要把R1中的值，按十进制的百位、十位、个位依次提取，并加上十进制数48，就可以得到各自的ASCII码，从而输出。

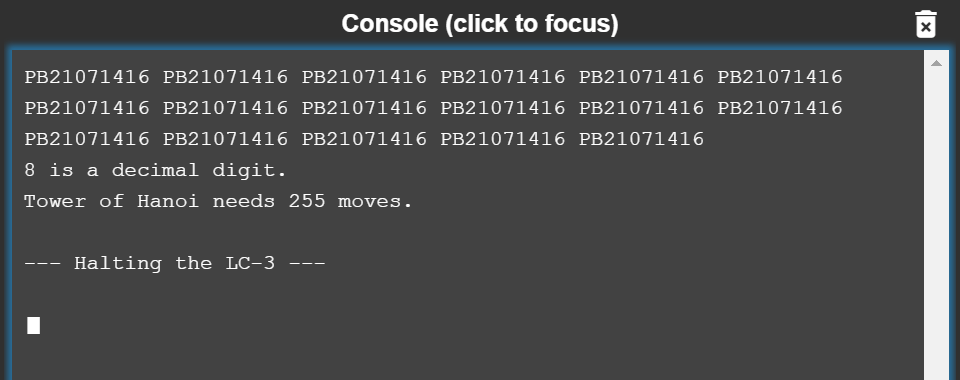
提取各数位的方法为:以百位为例，将结果循环减100，如果为非负数，计数器就加1，如果为负数，停止计数。此时，计数器中的数据就是百位的数字，再加上48得到ASCII码，就可以输出了。十位、个位的方法相同，不再赘述。

然后是键盘中断处理程序：

从键盘得到输入数据后，需要判断其是否为0~9之中的一个数据（即是否合法），0的ASCII码为48，9的ASCII码为57，于是我们只需要把输入和48,57比较即可：

若 48≤input≤57，则合法，FLAG置为1，表示可以停止学号显示，进行结果输出。否则输入不合法，FLAG仍为0，继续循环显示学号，等待下一个输入。

3. How do you design your own test cases to ensure the program works fine?

测试程序是否能正常运行比较简单，我们将0~9都输入一遍，看输出是否正确。结果表明都正确，下面给出输入为8时程序运行截图：

另外，我们还需要测试几个非法数据，例如我们输入 “!”, “(”, “a”,

“ ”都显示不合法：

