Lab3:Modulo 7

1 Overview

This lab helps you to understand the details of **input** and **output** in LC-3. Meanwhile, you will learn how to do a modulo operation with simply addition and branch instructions. The input will be given by **keyboard** and the output must be shown on **screen**.

2 Problem Description

On starting, the program wait for an input composed of **0** and **1**, which describe a number in **Binary**. After typing the number,type **y** submit. We name the input number **x**. Then the program calculate **x mod 7** and show the result on screen.

Noticed that **x** is a **positive** number not larger than **2^32**.

2.1 Examples

Input(by keyboard)	Output(on screen)
1	1
10111001110111	2
100110000100110100011101	4

2.2 Requirements

1. Start with **.ORIG x3000** and end with **.END**
2. Last instruction should be **TRAP x25(HALT)**
3. Use **capitalized** keywords and labels(e.g. “ADD” rather than “add”)
4. Maintain **spaces** after **commas**(ADD R0, R0, #1 rather than ADD R0,R0,#1)
5. **Decimal** constants start with **#**,**hexadecimal** with lowercase **x**
6. Write **comments** when necessary

3 Reference Approaches

3.1 Input and Output

Actually, there are TRAP instructions to help us (e.g., TRAP x23 for input and

TRAP x21 for output). But we encourage you to explore how it works under the TRAP instructions.

So try not to use TRAP instructions.

You can do input like this

```
01     START  LDI     R1, A          ; Test for
02            BRzp    START          ; character input
03            LDI     R0, B
04            BRnzp   NEXT_TASK      ; Go to the next task
05     A      .FILL   xFE00          ; Address of KBSR
06     B      .FILL   xFE02          ; Address of KBDR
```

You can do output like this

```
01     START   LDI     R1, A         ; Test to see if
02             BRzp    START         ; output register is ready
03             STI     R0, B
04             BRnzp   NEXT_TASK
05     A       .FILL   xFE04         ; Address of DSR
06     B       .FILL   xFE06         ; Address of DDR
```

TIPS

1.In LC-3, the input and output are all in **ASCII** form. which means if you type '1', the actual input is 0011 0001(or 49 in Decimal). If you want to show '2' on screen, the actual output should be 0011 0010(or 50 in Decimal).

2.**ECHO** means a program show your input on the screen. You may program an ECHO to check your input.

3.2 Modulo Calculation

Below are some possible approaches to calculate modulo 7. You can use one of them or propose your own solution.

3.2.1 Approach A

main idea

Use a loop to calculate x = x - 7 when x > 7. Once the program break from the loop, x is the result.

TIPS

1.Since the input x can be larger than 2^16, you are not able to store it in just one register, and you need a high accuracy algorithm to do the minus operation.

2.When x is a huge number, this approach can take a long time. So try this loop: x = x - b when x > b (b is another huge number satisfy b mod 7 = 0). Use a series of b to quickly reduce x.

3.2.2 Approach B

main idea

Use FSM (Finite State Machine). The FSM have 7 states represent the seven results of mod 7. Suppose the current x mod 7 = 2(state 2), and next we type 1, because next x = 2x + 1 and 2x + 1 mod 7 = 5, the next state is state 5,which means next x mod 7 = 5.The state will trans each time we type 0 or 1, and after we type y, the state is the result.

TIPS

1.Draw the FSM to help you program.

2.Use LABEL to define your states,and a register to store the current state.Use condition branch to simulate state trans.

3.2.3 Approach C

main idea

In second grade of primary school, we have learned how to quickly calculate x mod 3.Add up each of its digits, the sum mod 3 shares the same result with x mod 3 (x is in Decimal). If x is in Octal, it has a same property when calculate x mod 7.

For example, 10111001110111 is 11895 in Decimal and is 27167 in Octal. Since 2+7+1+6+7 = 27 and 27 mod 7 = 2 (in Octal), we have 11895 mod 7 = 2.

And Binary can easily turn into Octal by put 3-bits in a group.For example, 10 111 001 110 111, then 10 + 111 + 001 + 110 + 111 = 10 111, which is 27 in Octal.

With all above, you can get the result.

TIPS

1.Add some 0 before the input x to turn it into Octal.

2.Use Approach A to calculate the sum mod 7, since the sum must be a very small number.

4 Submission

submit this lab as follows:

PB********_Name_lab3.zip
|– – PB********_Name_report.pdf
└– – lab3.asm

4.1 Requirement for Report

1.Your name and student ID
2.Description of your solution
    For the above approach, here are some advice:
    For approach A, answer how to choose a series of b.
    For approach B, show FSM and how to draw it.
    For approach C, consider why adding up digits works.
3.Take some screen-shots include input and output.
4.(Optional) Any challenges you encountered and how you addressed them.