CSCI 2500 — Computer Organization
Team Project (document version 1.0)
Pipelined MIPS Simulator

Overview

- This project is due by 11:59:59 PM on Wednesday, December 11, 2019.
- This project is to be completed individually or in teams of no more than four students. Do not share your code with anyone else outside of your team.
- Use Submitty to define the teams and the discussion forum to advertise your skills and programming language preference. Be sure to use the `Project` category in the discussion forum. You will have until 11:59:59PM on Friday, November 22, 2019 to form your teams.
- After this deadline, anyone who didn't joint any team will be assumed to be working individually.
- Note that if you are working individually, you must define yourself as a team of one in Submitty!
- Each team member must submit the same code to receive a grade.
- If your team uses a late day, then each team member uses that late day; therefore, please coordinate accordingly.
- Hardcoding as a form of academic dishonesty is strictly forbidden and will be penalized with a project grade of 0 for the entire team. Hardcoding involves fabricating any part (except for literal text that defines the formatting of the output as shown in this PDF) of the "expected" output of the program and embedding it in the source code or in accompanying files.
- Academic integrity will be strictly enforced. Any submission which is similar to another submission (be that submitted by another team, any student/team from previous semesters, or code found online) will be penalized in accordance with the "Rensselaer Handbook of Student Rights and Responsibilities". In particular, penalties will equally apply to all teams who made similar submissions. The standard penalty for submitting work that is not your own, or for providing code that another student/team submits (perhaps after modification) will be 0 on the project and an additional overall 5% reduction on the semester grade for all members of all teams involved. Penalized students will also be prevented from dropping the course. More severe violations, such as stealing someone else's code, will lead to an automatic "F" in the course.

Programming Language Requirements

For this project, you can use C, C++, Python, Java, or MIPS. You must pick only one language. And your code **must** successfully execute on Submitty to obtain full credit.

Based on your language choice, you must use the following applicable naming conventions:

- If using C, please name your main file `p1.c`, with other supporting files named using `.c` and `.h` filename extensions as necessary. You must use `gcc` version 7.4.0. **And note that you can reuse your Homework 5 assignment only if working individually.**

- If using C++, please name your main file `p1.cpp`, with other supporting files named using `.cpp` and `.h` filename extensions as necessary. You must use `g++` version 7.4.0.
- If using Python, please name your main file `p1.py`, with other supporting files named using `.py` filename extensions as necessary. You must use Python 3.6.8.
- If using Java, please name your main public class `Project1` and place it in file `Project1.java`, with other supporting files named using `.java` filename extensions as necessary. Note that you must use Java version `javac 1.8.0_222`.
- If using MIPS, you are savage. There will be a 10% bonus (rounded to the nearest integer and capped at 100), to account for the fact that implementing this project in MIPS is arguably more time consuming than in high level programming languages. In other words, if you get 89 from the autograder, your grade becomes 98. If you get 92 from the autograder, your grade becomes 100. Please name your main file `p1.s`. Note that you must use SPIM version `8.0`. Good luck.

Homework Specifications

For this team-based project, you will implement a pipelined MIPS simulator, building on the pipelining concepts covered in Homework 5. As a reminder, there are five stages to the pipeline, i.e., `IF`, `ID`, `EX`, `MEM`, and `WB`.

For your simulation, you are required to support the `add`, `addi`, `and`, `andi`, `or`, `ori`, `slt`, `slti`, `beq`, and `bne` instructions; note that pseudo-instructions are not supported and that the `$zero` register may be used. More specifically, you must simulate the execution of a given set of instructions, showing the register contents as each instruction executes.

You must also show how the given sequence of instructions would be pipelined in a five-stage MIPS implementation. For this project, you must be able to support data hazards, forwarding, and control hazards.

You can assume that each given instruction will be syntactically correct. You can also assume that there is a single space character between the instruction and its parameters. Further, each parameter is delimited by a comma or parentheses. And you must support the usual set of `$t` and `$s` registers.

Since we are supporting branch instructions, labels must also be supported; you can assume that a label is all lowercase and is on a line by itself.

Below are a few example instructions and labels that you must support:

```
loop:
add $t0,$s2,$s3
addi $t1,$t3,73
or $s0,$zero,$t3
ori $s1,$zero,123
xyz:
slt $t3,$s1,$s2
beq $t2,$t4,loop
```

Required Command-Line Arguments

Your program must accept two command-line arguments as input. The first argument (i.e., `argv[1]`) is a single character, either `'F'` or `'N'`, that corresponds to supporting "Forwarding" mode or "Non-forwarding" mode.

The second argument (i.e., `argv[2]`) specifies the input file containing MIPS code to simulate. You may assume that no more than ten instructions are given in the input file. And note that each instruction will end with a newline character (i.e., `'\n'`).

If using MIPS for your Team Project implementation, instead of accepting command-line arguments, read input from the user by issuing appropriate `syscall` calls. First, take a single character for the forwarding mode, then input up to ten instructions. If the user enters a blank line (""), it means there are no more instructions to enter. To facilitate testing, you may put all user input in a text file and then redirect stdin of SPIM to read input from this text file, as shown in the example below:

```
spim -file p1.s < p1-test01-stdin.txt
```

Required Output

For your output, you must show *each cycle* of program execution, including the contents of the registers. To show the registers, use four columns, each with a fixed width of 20 characters. Display the `$s` registers first, then the `$t` registers. Be sure there are no trailing spaces at the end of each line of output.

And as with Homework 5, for the pipeline, each cycle will correspond to a column of output. Initially, each column is empty, indicated by a period (i.e., `'.'`). And note that all registers are assumed to be initialized to zero.

Unlike Homework 5, use fixed-width formatting (e.g., `printf()`) to delimit each column. More specifically, the first column must have a width of exactly 20 characters, while each subsequent column (corresponding to each clock cycle) must have a width of exactly four characters. Include a space between each column and left-justify each column. Further, be sure there are no trailing spaces on the end of each line of output.

Finally, show no more than 16 cycles in your simulation, meaning that if you reach cycle 16, display that last cycle and end your simulation.

Handling Data and Control Hazards

Recall that a *data hazard* describes a situation in which the next instruction cannot be executed in the next cycle until a previous instruction is complete. Your code should be able to detect when it is necessary to insert one or more "bubbles" (see Section 4.7 of the textbook and corresponding lecture notes for more details).

As you did on Homework 5, you will need to properly handle data hazards by adding `nop` instructions as necessary. Show these cases by indicating an asterisk (i.e., `'*'`) in the appropriate columns and adding the required number of `nop` instructions.

Data hazards are obstacles to pipelined execution, but in some instances, we can use *forwarding* to forward intermediate data as soon as it becomes available (i.e., after the `EX` stage) on to those components that need it. If the forwarding argument is set, then you should simulate forwarding (as illustrated in Figures 4.29 and 4.53 of our textbook).

Finally, for control hazards, you should use a prediction model that assumes the branch will *not* be taken. Recall that the decision to branch is decided at the `MEM` stage of the branch instruction. Therefore, if the branch is taken (i.e., we predicted incorrectly), the instructions (at most, three) that have been fetched and decoded must be discarded, with execution continuing at the branch target. Show this by indicating an asterisk (i.e., `'*'`) in the appropriate columns.

On the next few pages, we present a few example runs of your program that you should use to better understand how your program should work, how you can test your code, and what output formatting to use for Submitty.

The first example (i.e., `p1-input01.txt`) includes no data hazards (or forwarding).

```
bash$ cat p1-input01.txt
ori $s1,$zero,451
addi $t2,$s0,73
add $t4,$s3,$s7
bash$ ./a.out N p1-input01.txt
START OF SIMULATION (no forwarding)
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  .   .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .   .
add $t4,$s3,$s7     .   .   IF  .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .   .
add $t4,$s3,$s7     .   .   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
```

```
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .   .
add $t4,$s3,$s7     .   .   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
add $t4,$s3,$s7     .   .   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
add $t4,$s3,$s7     .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
END OF SIMULATION
```

The second example (i.e., `p1-input02.txt`) includes a dependency on register `$s1`, but no forwarding.

```
bash$ cat p1-input02.txt
ori $s1,$zero,451
addi $t2,$s0,73
add $t4,$s1,$s7
bash$ ./a.out N p1-input02.txt
START OF SIMULATION (no forwarding)
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  .   .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
```

```
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .   .
nop                 .   .   IF  ID  *   .   .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  ID  .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
nop                 .   .   IF  ID  *   *   .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  ID  EX  .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
nop                 .   .   IF  ID  *   *   *   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  ID  EX  MEM .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
nop                 .   .   IF  ID  *   *   *   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  ID  EX  MEM WB  .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 451           $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
END OF SIMULATION
```

The third example (i.e., again `p1-input02.txt`) includes a dependency on register `$s1`, this time with forwarding.

```
bash$ cat p1-input02.txt
ori $s1,$zero,451
addi $t2,$s0,73
add $t4,$s1,$s7
bash$ ./a.out F p1-input02.txt
START OF SIMULATION (forwarding)
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  .   .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
```

```
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$s0,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
add $t4,$s1,$s7     .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 451           $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
END OF SIMULATION
```

The fourth example (i.e., `p1-input03.txt`) illustrates a control hazard, with forwarding.

```
bash$ cat p1-input03.txt
ori $s1,$zero,451
loop:
addi $t2,$t2,73
slti $t4,$s1,453
addi $s1,$s1,1
bne $t4,$zero,loop
ori $s6,$t6,77
add $s7,$s0,$s0
andi $s2,$t5,255
bash$ ./a.out F p1-input03.txt
START OF SIMULATION (forwarding)
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  .   .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  .   .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
```

```
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  .   .   .   .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  .   .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 0             $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  .   .   .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  .   .   .   .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  .   .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 0             $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  .   .   .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  .   .   .   .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  .   .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 0             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
```

```
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM .   .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  .   .   .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  .   .   .   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  .   .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 451           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM .   .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  .   .   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  .   .   .   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  .   .   .   .   .   .   .   .

$s0 = 0             $s1 = 452           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  *   .   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  *   .   .   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  *   .   .   .   .   .   .   .
addi $t2,$t2,73     .   .   .   .   .   .   .   .   IF  .   .   .   .   .   .   .

$s0 = 0             $s1 = 452           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
```

```
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  *   *   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  *   *   .   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  *   *   .   .   .   .   .   .
addi $t2,$t2,73     .   .   .   .   .   .   .   .   IF  ID  .   .   .   .   .   .
slti $t4,$s1,453    .   .   .   .   .   .   .   .   .   IF  .   .   .   .   .   .

$s0 = 0             $s1 = 452           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  *   *   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  *   *   *   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  *   *   *   .   .   .   .   .
addi $t2,$t2,73     .   .   .   .   .   .   .   .   IF  ID  EX  .   .   .   .   .
slti $t4,$s1,453    .   .   .   .   .   .   .   .   .   IF  ID  .   .   .   .   .
addi $s1,$s1,1      .   .   .   .   .   .   .   .   .   .   IF  .   .   .   .   .

$s0 = 0             $s1 = 452           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
```

```
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  *   *   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  *   *   *   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  *   *   *   *   .   .   .   .
addi $t2,$t2,73     .   .   .   .   .   .   .   .   IF  ID  EX  MEM .   .   .   .
slti $t4,$s1,453    .   .   .   .   .   .   .   .   .   IF  ID  EX  .   .   .   .
addi $s1,$s1,1      .   .   .   .   .   .   .   .   .   .   IF  ID  .   .   .   .
bne $t4,$zero,loop  .   .   .   .   .   .   .   .   .   .   .   IF  .   .   .   .

$s0 = 0             $s1 = 452           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 73            $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  *   *   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  *   *   *   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  *   *   *   *   .   .   .   .
addi $t2,$t2,73     .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .   .   .
slti $t4,$s1,453    .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM .   .   .
addi $s1,$s1,1      .   .   .   .   .   .   .   .   .   .   IF  ID  EX  .   .   .
bne $t4,$zero,loop  .   .   .   .   .   .   .   .   .   .   .   IF  ID  .   .   .
ori $s6,$t6,77      .   .   .   .   .   .   .   .   .   .   .   .   IF  .   .   .

$s0 = 0             $s1 = 452           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 146           $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
```

```
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  *   *   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  *   *   *   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  *   *   *   *   .   .   .   .
addi $t2,$t2,73     .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .   .   .
slti $t4,$s1,453    .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .   .
addi $s1,$s1,1      .   .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM .   .
bne $t4,$zero,loop  .   .   .   .   .   .   .   .   .   .   .   IF  ID  EX  .   .
ori $s6,$t6,77      .   .   .   .   .   .   .   .   .   .   .   .   IF  ID  .   .
add $s7,$s0,$s0     .   .   .   .   .   .   .   .   .   .   .   .   .   IF  .   .

$s0 = 0             $s1 = 452           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 146           $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  *   *   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  *   *   *   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  *   *   *   *   .   .   .   .
addi $t2,$t2,73     .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .   .   .
slti $t4,$s1,453    .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .   .
addi $s1,$s1,1      .   .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .
bne $t4,$zero,loop  .   .   .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM .
ori $s6,$t6,77      .   .   .   .   .   .   .   .   .   .   .   .   IF  ID  EX  .
add $s7,$s0,$s0     .   .   .   .   .   .   .   .   .   .   .   .   .   IF  ID  .
andi $s2,$t5,255    .   .   .   .   .   .   .   .   .   .   .   .   .   .   IF  .

$s0 = 0             $s1 = 453           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 146           $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
```

```
CPU Cycles ===>     1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
ori $s1,$zero,451   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .   .
addi $t2,$t2,73     .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .   .
slti $t4,$s1,453    .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .   .
addi $s1,$s1,1      .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .   .
bne $t4,$zero,loop  .   .   .   .   IF  ID  EX  MEM WB  .   .   .   .   .   .   .
ori $s6,$t6,77      .   .   .   .   .   IF  ID  EX  *   *   .   .   .   .   .   .
add $s7,$s0,$s0     .   .   .   .   .   .   IF  ID  *   *   *   .   .   .   .   .
andi $s2,$t5,255    .   .   .   .   .   .   .   IF  *   *   *   *   .   .   .   .
addi $t2,$t2,73     .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .   .   .
slti $t4,$s1,453    .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .   .
addi $s1,$s1,1      .   .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB  .
bne $t4,$zero,loop  .   .   .   .   .   .   .   .   .   .   .   IF  ID  EX  MEM WB
ori $s6,$t6,77      .   .   .   .   .   .   .   .   .   .   .   .   IF  ID  EX  *
add $s7,$s0,$s0     .   .   .   .   .   .   .   .   .   .   .   .   .   IF  ID  *
andi $s2,$t5,255    .   .   .   .   .   .   .   .   .   .   .   .   .   .   IF  *
addi $t2,$t2,73     .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   IF

$s0 = 0             $s1 = 453           $s2 = 0             $s3 = 0
$s4 = 0             $s5 = 0             $s6 = 0             $s7 = 0
$t0 = 0             $t1 = 0             $t2 = 146           $t3 = 0
$t4 = 1             $t5 = 0             $t6 = 0             $t7 = 0
$t8 = 0             $t9 = 0
--------------------------------------------------------------------------------
END OF SIMULATION
```

Assumptions

Given the complexity of this project, you can make the following assumptions:

- Assume all input files are valid.
- Assume the length of `argv[1]` is exactly 1 character.
- Assume the length of `argv[2]` is at most 128 characters.

Submission Instructions

Before you submit your code, be sure that you have clearly commented your code (this should not be an after-thought). Further, your code should have a clear and logical organization.

To submit your project (and also perform final testing of your code), please use Submitty. Note that the test cases for this team project will be available on Submitty a minimum of three days before the due date and will include hidden test cases.

Each team member must submit the same code to receive a grade.

Also as a reminder, your code **must** successfully execute on Submitty to obtain credit for this project.