Performance improvement using *pipeline*

A task has been created in the web page of the course. Through this task you can download the program source.s written in assembly language for the MIPS64 architecture. You must follow these steps:

- Download the aforementioned source file using the referred task in the Campus Virtual. You need to provide your Spanish ID since the file is customized for each student.

- Set up the next options in the simulator before loading the program:

  - Floating point operations:
    - Addition latency 2 clock cycles
    - Multiplier latency 5 clock cycles
    - Division latency 10 clock cycles

  - If enabled, disable the option “Enable forwarding”. This is really important, so please double check that forwarding is disabled before simulating the program.

The exercise will be marked automatically so the next rules must be followed carefully. If the deliverable does not match the following rules, it will be marked with 0. On the other hand, if the simulator is not properly set up according to the previous settings, a penalty of 5 points is given.

Create a text file, answers.txt, with UTF-8 encoding. The two first lines are your Spanish ID and your full name. At the end of this document there is an example of the answers.txt file.

1. Write the segmentation faults that stall the pipeline while executing the program: You must point out the clock cycle, the type of the segmentation fault, the duration in clock cycles, and a brief description indicating the source of the problem. You just need to provide the first three segmentation faults of each type (if there are more than three). You must follow the next syntax to provide the answer for this question:

   1. **Cycle**;**Type**;**Duration;Description**

   An example of an answer for this question is shown below.

   ```
   1. 3;Control;1;A jump is executed
      19;Data;2;Instruction X requires the value …
      24;Structural;1;Item Y stored in Z is required
   ```

   Please note that “1.” identifies the answer for question 1. Furthermore, each segmentation fault is provided in a new line and each field is separated with “;”.

2. How many clock cycles does the program require to be executed? Use the format shown below to provide your answer, where X is a natural number.

   2. **X**

3. What is the average CPI in the program execution? Use the format shown below to provide your answer, where X is a natural number.

   3. **X**

4. What is the speedup obtained in the program compared with a sequential version of the CPU where the five stages of the pipeline run in a sequential mode for all the instructions? Use the format shown below to provide your answer, where X is a natural number.

   4. **X**

Take a screenshot of the simulator after executing 63 clock cycles of the program. The settings of the simulator must be also shown in this screenshot (using the dialog box Set Architecture from Configure → Architecture). Save this screenshot as capture.png.

You must deliver the following files zipped in a .zip file called Results.zip:

- Source code: source.s
- Screenshot: capture.png
- Answers file: answers.txt

A checking format tool is provided in the same task in the Campus Virtual to validate your results file. You must check that the format of your files is correct before delivering them.

Example of Answers.txt

```
11222333
Name Foo Var
1. 3;Control;1;A jump is executed
   19;Data;2;Instruction X requires the value …
   24;Structural;1;Item Y stored in Z is required    …
2. 83
3. 4.6
4. 6.78
```