Architetture dei Sistemi di Elaborazione
02GOLOV [GRB-ZZZ]

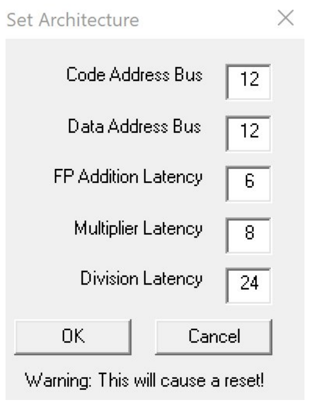
Delivery date:
1st November 2022

Laboratory 3

Expected delivery of lab_03.zip must include:
- `program_1_a.s`, `program_1_b.s` and `program_1_c.s`
- this file compiled and if possible in pdf format.

Please, configure the winMIPS64 simulator with the *Base Configuration* provided in the following:

- Code address bus: 12
- Data address bus: 12
- Pipelined FP arithmetic unit (latency): 6 stages
- Pipelined multiplier unit (latency): 8 stages
- Divider unit (latency): not pipelined unit, 24 clock cycles
- Forwarding is enabled
- Branch prediction is disabled
- Branch delay slot is disabled
- *Integer ALU: 1 clock cycle*
- *Data memory: 1 clock cycle*
- *Branch delay slot: 1 clock cycle.*


Set Architecture

Code Address Bus	12
Data Address Bus	12
FP Addition Latency	6
Multiplier Latency	8
Division Latency	24

Warning: This will cause a reset!

1) Starting from the assembly program you created in the previous lab called **`program_1.s`**:

```
for (i = 0; i < 60; i++){
        v5[i] = ((v1[i]+v2[i]) * v3[i])+v4[i];
        v6[i] = v5[i]/(v4[i]*v1[i]);
        v7[i] = v6[i]*(v2[i]+v3[i]);
}
```

   a. Detect manually the different data, structural and control hazards that provoke a pipeline stall

   b. Optimize the program by re-scheduling the program instructions in order to eliminate as many hazards as possible. Compute manually the number of clock cycles the new program (**`program_1_a.s`**) requires to execute, and compare the obtained results with the ones obtained by the simulator.

   c. Starting from **`program_1_a.s`**, enable the *branch delay slot* and re-schedule some instructions in order to improve the previous program execution time. Compute manually the number of clock cycles the new program (**`program_1_b.s`**) requires to execute, and compare the obtained results with the ones obtained by the simulator.

   d. Unroll 3 times the program (**`program_1_b.s`**), if necessary re-schedule some instructions and increase the number of used registers. Compute manually the number of clock cycles the new program (**`program_1_c.s`**) requires to execute, and compare the obtained results with the ones obtained by the simulator.

Complete the following table with the obtained results:

Program / **Clock cycle computation**	**`program_1.s`**	**`program_1_a.s`**	**`program_1_b.s`**	**`program_1_c.s`**
`By hand`	3905	3360	3360	2645
`By simulation`	4029	3726	3607	2747

Compare the results obtained in point 1, and provide some explanation in the case the results are different.

Eventual explanation:

The main difference between the computation by hand and by simulation is that during the simulation on winmips, when a stall occurs, the folowing instruction will stall in a different part of the pipeline. In particular if a floating point instruciton doesn't have the required operands, the instruciton enter in the execution stage and then stalls.

Otherwise, when we simulate the execution by hand, usualy stall the instruction in the previously stage of the pipeline.