Architetture dei Sistemi di Elaborazione	Delivery date: October 29th 2021
Laboratory 3	Expected delivery of lab_03.zip must include: - `program_2_a.s, program_2_b.s` and `program_2_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): 4 stages
- Pipelined multiplier unit (latency): 8 stages
- divider unit (latency): not pipelined unit, 12 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.*

1) Starting from the assembly program you created in the previous lab called **`program_2.s`**,:

```
for (i = 0; i < 40; i++){
        v5[i] = v1[i]+(v2[i] * v3[i]);
        v6[i] = v5[i]*v4[i];
        v7[i] = v6[i]/v2[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 much hazards as possible. Compute manually the number of clock cycles the new program (**`program_2_a.s`**) requires to execute, and compare the obtained results with the ones obtained by the simulator.

   c. Starting from **`program_2_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_2_b.s`**) requires to execute, and compare the obtained results with the ones obtained by the simulator.

   d. Unroll 4 times the program (**`program_2_b.s`**), if necessary re-schedule some instructions and renaming the used registers. Compute manually the number of clock cycles the new program (**`program_2_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_2.s`	`program_2_a.s`	`program_2_b.s`	`program_2_c.s`
`By hand`	1645	1015	937	792
`By simulation`	1650	1018	939	796

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

Eventual explanation:

Da _2.s a _2_a.s: rimosso un jump finale, dovuto al modo di ciclare che usavo. Ho usato dei registri per fare le store delle operazioni diversi dal registro indice che incremento ad ogni ciclo. In questo modo ho potuto spostare le store prima delle operazioni (quindi la store memorizza il valore delle operazioni fatte dal ciclo precedente).

Da _2_a.s a _2_b.s: ho spostato delle istruzioni per distanziare delle operazioni dipendenti tra di loro. Essendoci il delay slot attivo ho rimosso la nop dopo il salto e l'ho rimpiazzata cona una istruzione utile al programma, in questo caso la div.

Da _2_b.s a _2_c.s: ho eseguito l'unroll 4 volte e raggruppato le operazioni simili che non avevano dipendenza da registri (lavorando su dati differenti). Le div sono le uniche operazioni che non sono riuscito a distanziare causa mancanza di istruzioni. Anche qui faccio le store dei risultati al ciclo successivo.