Architetture dei Sistemi di Elaborazione 02GOLOV [A-L]	Delivery date: Wednesday, November 6, 2019
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 < 30; i++){
        v5[i] = (v1[i]*v2[i]) + v3[i];
        v6[i] =(v3[i]*v4[i])/v5[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 again the code in order to positively exploit the branch delay slot, or add NOP operations that avoid the code to lost its functionalities. 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 3 times the program (**`program_2_b.s`**), after unrolling the code, reschedule again the code in order to improve the program performance. 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:

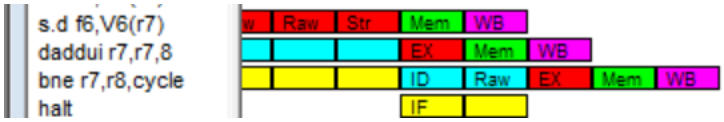
Program / Clock cycle computation	`program_2.s`	`program_2_a.s`	`program_2_b.s`	`program_2_c.s`
`By hand`	936	936	878	548
`By simulation`	966	966	908	548

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

Eventual explanation:

I c.c tra il **`program_2.s`** e **`program_2_a.s`** sono identici in quanto si era cercato di ottimizzare il programma già nello scorso laboratorio, quindi nulla è stato cambiato. Attivando il delay slot nel programma b si riescono a risparmiare 30 cicli di clock (nel calcolo a mano) in quanto si elimina lo stallo dovuto alla branch anticipando l'esecuzione della prima istruzione del ciclo nel dealy slot.

NOTA: il calcolo a mano e il calcolo per simulazione differiscono di 30 cicli a causa di un comportamento apparentemente anomalo dell'emulatore, che va in stallo nell'istruzione di branch, nonostante il forwarding sia abilitato (è stato verificato che non ci siano dipendenze ne con r7 ne con r8)



Questo fa si che per ogni ciclo del programma si perdono due clock cycle, (al posto di uno) per un totale di 60 nei primi due casi e di 30 nel caso del terzo (uno dei due stalli viene risolto dal delay slot)

L'ultimo caso è il più efficente in quanto,utiizzando dei registry aggiuntivi, si possono parallelizzare 3 cicli alla volta riducendo di molto il numero di stalli