Architetture dei Sistemi di Elaborazione
02GOLOV

Laboratory 3

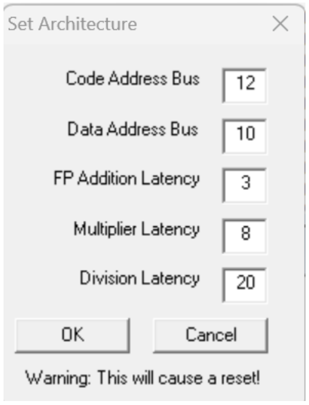
Delivery date:
9th November 2023

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): 3 stages
- Pipelined multiplier unit (latency): 8 stages
- divider unit (latency): not pipelined unit, 20 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*.



Code Address Bus	12
Data Address Bus	10
FP Addition Latency	3
Multiplier Latency	8
Division Latency	20

Warning: This will cause a reset!

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

```
int m=1 /* 64  bit */
double k,p
for (i = 0; i < 64; i++){
        if (i is even) {
           p= v1[i] * ((double)( m<< i)) /*logic shift */
           m = (int)p
        } else {
        /*  i is odd */
            p= v1[i] / ((double)m* i))
            k =  ((float)((int)v4[i]/ 2^i)
        }

        v5[i] = ((p * v2[i]) + v3[i])+v4[i];

        v6[i] = v5[i]/(k+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 2 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	6403	6403	6403	6467
By simulation	6599	5575	5576	5894

Collect the IPC (from the simulator) for different programs.

	program_1.s	**program_1_a.s**	**program_1_b.s**	**program_1_c.s**
IPC	0.37	0.39	0.40	0.37

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

Eventual explanation:

Eliminating hazards greatly improves the performance while the other techniques are probably more specific and in this instance do not improve performance, surprisingly the C is slower than the rest but that can be probably further optimized given that we don’t need to check the if clause anymore