Architetture dei Sistemi di Elaborazione 02GOLOV	Computer Architectures 02LSEYG
Laboratory 0x03	Expected delivery of lab_03.zip must include: - **program_1.s**, **program_1_a.s**, and **program_1_b.s** - This file, filled with information and possibly compiled in a **PDF** format. Delivery date: October 30th 2025

This lab will explore some of the concepts seen during the lessons, such as hazards, rescheduling, and loop unrolling. The first thing to do is to configure the GEM5 simulator with the ***Initial Configuration*** provided below:

```
INTEGER_ALU_LATENCY = 1
INTEGER_MUL_LATENCY = 1
INTEGER_DIV_LATENCY = 1
FLOAT_ALU_LATENCY = 4
FLOAT_MUL_LATENCY = 6
FLOAT_DIV_LATENCY = 12
```

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

```
int m = 1;
float a, b;
for (i = 31; i >= 0; i--) {
    if (i is a multiple of 3) {
        a = v1[i] / ((float) m << i); /*logic shift */
        m = (int) a;
    } else {
        a = v1[i] * ((float) m * i);
        m = (int) a;
    }
    v4[i] = a * v1[i] - v2[i];
    v5[i] = v4[i]/v3[i] - b;
    v6[i] = (v4[i]-v1[i]) * v5[i];
}
```

a. Manually detect the different data, structural, and control hazards that cause a pipeline stall. Report at least 3 hazards in the code (or the pipeline) and fill the following table:

C Line	Corresponding RISC-V Instruction(s)	Type of Hazard	Pipeline Stage	Cause of Stall
`a = v1[i] / ((float) m << 1)`	`fcvt.s.w fs7, x28` `fdiv.s fs8, fs1, fs7`	Data – RAW	EX	`fs7` non pronto: il convertitore FP scrive in WB, la divisione lo legge in EX.
`v4[i] = a * v1[i] - v2[i]`	`fmul.s fs4, fs8, fs1` `fsub.s fs4, fs4, fs2`	Data – RAW	EX	`fs4` prodotto dal moltiplicatore non ancora disponibile per la sottrazione.

`v5[i] = v4[i]/v3[i] - b`	`fdiv.s fs5, fs4, fs3` `fsub.s fs5, fs5, fs2`	Data – RAW	EX	`fs5` disponibile solo al termine della divisione; la sottrazione deve attendere.
`v6[i] = (v4[i] - v1[i]) * v5[i]`	`fsub.s fs11, fs4, fs1` `fmul.s fs6, fs11, fs5`	Data – RAW	EX	`fs11` non pronto per l'operando della moltiplicazione.
Ramo `if/else` e fine `loop`	`bnez x27, else_branch` `j loop`	Control	IF/ID	Direzione/target ignoti fino alla risoluzione: flush delle istruzioni in fetch.
Strutturale (divisione)	`fdiv.s fs8, fs1, fs7` `fdiv.s fs5, fs4, fs3`	Structural	ID/issue	Unità divisione non pipelinata: la seconda divisione attende che la prima liberi l'unità.

b. Optimize the program by re-scheduling instructions to eliminate as many hazards as possible. Manually calculate the number of clock cycles for the new program (**`program_1_a.s`**) to execute and compare the results with those obtained by the simulator.

c. Unroll the program (**`program_1_a.s`**) two times; If necessary, re-schedule instructions and increase the number of registers used. Manually calculate the number of clock cycles to execute the new program (**`program_1_b.s`**) and compare the results obtained with those obtained by the simulator.

Complete the following table with the obtained results:

Program	**program_1.s**	**program_1_a.s**	**program_1_b.s**
Clock cycles by hand	2567	2620	2343
Clock cycles by simulation	2302	2239	1678

2) Collect the Cycles Per Instruction (CPI) from the simulator for different programs

	program_1.s	**program_1_a.s**	**program_1_b.s**
CPI	2.091	1.885	1.844

Compare the results obtained in 1) and provide some explanation if the results are different.

Eventual explanation: La differenza nasce perché il conteggio "by hand" somma le latenze come se bloccassero sempre la pipeline, mentre in simulazione le istruzioni indipendenti si sovrappongono grazie a ILP, forwarding e unità FP pipelinate o con initiation interval che consente nuovi issue mentre una `fdiv/fmul/fsup` è in esecuzione; inoltre branch e loop costano meno (risoluzione/predizione precoce), le store non fermano EX e le load hanno solo la vera penalità di load-use. Per questo `program_1.s` mostra un gap moderato, `_a.s` aumenta il divario perché lo scheduling riduce le dipendenze visibili, e `_b.s` ha il gap maggiore grazie all'unrolling a due elementi che crea catene FP parallele e copre quasi tutta la latenza delle unità lunghe.