jArchitetture dei Sistemi di Elaborazione
02GOLOV

Laboratory 0x03

Computer Architectures
02LSEYG

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
If(i is multiple	rem x13, x7, x14 bnez x13,NotMul3	Data - RAW	ID	X13 non confrontabile con x0

of 3				fino a che non e' stato calcolato da rem
A = v1[i]*((float)m*i);	fcvt.s.w f12,x12 fmul.s f10, f1, f12	Data - RAW	EX	Non posso moltiplicare v1[i] per m fino a che non ho finito il cast

a=v1[i]*((float) m*i);
m=(int)a;
	fmul.s f10, f1, f12
	fcvt.w.s x11, f10
		Structural hazard	ID	L'operazione di cast attende di poter fare il decode mentre la precedente e' in stallo

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	2149	2117	1784
Clock cycles by simulation	2164	2132	1799

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	2.24714	2.21391	1.99004

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

Eventual explanation:
Come e' facilmente notare il calcolo manuale differisce sempre di 15 cc da quello del simulatore, questo dovuto ai colpi di clock persi per il riscaldamento delle cache. Si puo' notare inoltre come l'unroll aggiunge grandi quantita di istruzioni a disposizione per il rescheduling e riduce fortemente il numero di jump aumentando cosi drasticamente le performance.