# Analytical study in the pipelined processor from [2]

We lose 1 cycle in every iteration of the loop due to the RAW hazard between the second lw instruction and the sub instruction. The processor must insert a bubble in between both instructions and then forward the data read by the lw. We also lose 1 cycle in every iteration due to the bne instruction being taken. The processor will flush the sequentially fetched instruction (last addi instruction).

**Detailed analysis:**

* 100 iterations are performed.
* The first instruction is fetched in cycle 1.
* The first lw is fetched in cycle 4 in the first iteration, and it is fetched in cycle 13 in the second iteration. Thus, each iteration from the 1st to the 99th takes 9 cycles.
* 100th iteration takes 8 cycles, as the bne instruction is not taken.
* The last addi instruction takes 5 cycles.

Taking all this into account we have:

* Number of cycles = 3 + 99\*9 + 8 + 5 = 907 cycles
* Number of instructions = 100\*7 + 4 = 704 instructions
* **CPI** = 907/704 **≈** **1.29**

We can easily reorder the program by moving instruction (ADDI $t1,$t1,-1) after the second lw instruction. That way, a bubble is not necessary after the lw. In this case we´d have:

* Number of cycles = 3 + 99\*8 + 7 + 5 = 807 cycles
* Number of instructions = 100\*7 + 4 = 704 instructions
* **CPI** = 807/704 **≈ 1.15**

# Analytical study in MIPSfpga

When the baseline program is executed in MIPSfpga, performance is the same as in the processor from [2], as there are two stalls per iteration (1 for the RAW hazard between the lw and the sub, and 1 for the nop instruction after the bne). Thus, **CPI** = 908/704 **≈ 1.29**.

If we now reorder the program manually, we can move instruction (ADDI $t1,$t1,-1) after the second lw instruction and fill the delay-slot with instruction (addi $t6,$t6,4). That way, we will experiment no stalls in the loop, thus:

* Number of cycles = 3 + 100\*7 + 5 = 708 cycles
* Number of instructions = 100\*7 + 4 = 704 instructions
* **CPI** = 708/704 **≈ 1**

# Empirical study in MIPSfpga

## Original program (no optimizations)

The results provided by the performance counters are the following:

* **Number of cycles = 923**
* **Number of instructions =** 812 – 100 (1 nop per iteration) **= 712**

We need to correct the number of instructions provided by the performance counters, as nop instructions should not be included.

* **CPI =** 923 / 712 **≈ 1.3**

Note that this result is not perfectly accurate, as there are some extra instructions included due to the Performance Counter accounting process. You can look for “<main>:” in file *program.dis* to inspect the assembly code generated by the compiler.

## Original program (-O3 option)

The results provided by the performance counters are the following:

* **Number of cycles = 819**
* **Number of instructions = 709**
* **CPI =** 819 / 709 **≈ 1.15**

## Reordered program

The results provided by the performance counters are the following:

* **Number of cycles = 719**
* **Number of instructions = 709**
* **CPI =** 719 / 709 **≈ 1**