Laboration report 2

Group 93
Gustav Larsson(Guslap-5)
Carl Fristenstam(Carfri-5)

We've learned how pipelining works in mips. We have learned how to avoid hazards by not using the same variable many times in a row after certain instructions. We also learned to use the extended SyncSim mode and use the interrupt window. I guess we will use that more in the coming labs.

The most difficult part of the lab was trying to grasp when the hazards were occurring and when they could be prevented by restructuring the code.

Yes we used SyncSim when writing and debugging. At first we placed a lot of nop instructions throughout the code, then we tried removing a few and rerunning the program in SyncSim to see what changed.

36073 cycles to decode the string "coded" in lab 1b.

42227 cycles in lab2 without the interrupt.

4 * 36073 / 42227 = 3.417

The reason for why we didn't get 4 (the ideal speedup) is probably because our pipeline code isn't optimized perfectly. For example there are 2 places in the code where we stall, we do this because there is no way to solve it without grand changes to the code. If we rewrote the code with pipelining in mind we would probably get a much better result, we could then for example avoid storing the result directly after a call.

42709 clock cycles with the timer interrupts. The extra clock cycles can be explained by the extra instructions that had to be run within the interrupt routine.

By typing in the window we extended the program by around 1000 extra cycles. When there is an input in the I/O window the program will be interrupted for a few cycles because some specific interrupt instructions will be executed. This explains why the more you input into the I/O window the more cycles it will take to finish the code.