Universidade de Aveiro

Mestrado em Engenharia de Computadores e Telemática

Arquitecturas de Alto Desempenho

DLX – Pipelining 2

Academic year 2021/2022 Adaptation of exercise guide by Nuno Lau/José Luís Azevedo

4. Write a program that orders the values of an integer array stored in memory from the largest value to the smallest one.

   4.1. Run the code in the DLX simulator with the *forwarding* option turned off and, through the interspersing of `nop` instructions to take care of the hazards, make it execute correctly. What is the number of clock cycles for the full execution? What is the speed up attained relatively to the original code being run in a non-pipelined processor which takes 4 clock cycles to execute *store* and *jump/branch* instructions and 5 clock cycles to execute all other instructions?

   4.2. Turn the *forwarding* option on and discard the `nop` instructions that are no longer required. What was the speed up now attained?

   4.3. Turn on the MIPS compatibility mode, in order to enable *pipeline interlocking*, and discard the remaining `nop` instructions. Check that your initial code now runs correctly. Go through the clock cycle diagram to understand why. Explain what has happened in your own words. What is the speed up now attained?

   4.4. Consider the *branch predictor* alternatives: *none*, *static – predict always not taken* and *static – predict always taken*. Which one produces the most efficient run? Why? What is the speed up now attained?