# Project Final Report: Pipeline Mechanics

## Abstract

## Problem Statement

In order to improve the performance of a modern processor, designers attempt to leverage instruction level parallelism to increase the number of instructions executed in a given clock cycle or reduce the amount of time between each instruction. Pipelining is a possible solution to this problem, where instructions are broken down into multiple stages. A basic MIPS64 pipeline is shown below, where registers are added to create the various pipeline stages.



Figure : Basic MIPS RISC Pipeline [1]

Using pipelining, more than one instruction can be fed into the pipeline allowing at most one instruction to be completed at the end of each clock cycle once the pipeline is filled. However, various structural and data hazards keep this maximum performance from being reached. For instance, consider a set of instructions that suffer from a read after write dependency, where the following instruction depends on an output generated by the previous instruction. The following code snippet illustrates this problem.

DADD R1, R2, R3 ;Put something in R1

DADD R4, R1, R5 ;Dependency on R1

In the standard non-optimized MIPS pipeline, a stall would be needed such that the first instruction can complete before the second can be resumed. To improve the efficieny of the pipeline, a technique known as forwarding was developed. By passing the result of the first instruction to the

## Methodology

### Modifications

### Test Cases

### Evaluation