# Project Proposal: Pipeline Mechanics

## Abstract

In order to determine the effect of forwarding on the standard MIPS64 pipeline, the EduMIPS64 simulator will be altered to perform partial and full forwarding. The modified pipelines can then be evaluated on their performance using a set of standardized benchmarks. Finally, these results will be compared, and the cost and performance trade-offs of each modification discussed.

## Methodology

### Modifications

Many modern architectures leverage forwarding in order to reduce the number of stalls generated by data dependencies and other hazards. Forwarding can be implemented in a variety of ways, each having different levels of cost. Three cases will be implemented

1. The standard MIPS64 pipeline without forwarding
2. Modified MIPS64 pipeline with forwarding from the execution stage only
3. Modified MIPS64 pipeline with forwarding from both the execution and memory stages

Implementing the pipeline modifications would normally consist of adding hazard detection in the ID stage and then adding some virtual muxes to take care of the additional instruction sources. EduMIPs seems to work in a way such that Hazards aren’t detected in the ID stage but are detected dynamically through the pipeline as needed. Although this wouldn’t be very feasible in hardware because of increased complexity, it has the same functional behavior in terms of how instructions move through the pipeline. To deal with this, we’ll stay with the structure of the program and just add some buffers for the EX and MEM stages such that their values can be read when it comes time to compute the EX stage of the pipeline.

### Test Cases

The EduMIPS 64 simulation environment provides a platform that will be modified to implement the changes described above. However, a suitable test set will be composed to test each modification and examine the number of stalls in the pipeline. Consequently, the test bench should be made up of a variety of assembly programs that are structured with various data hazards. The following hazards will be tested.

* Read after write
* Write after read
* Write after write

### Evaluation

The impact of each modification will be evaluated by the improvement (or speedup) resulting from each modification. Because of how the pipeline is managed by EduMIPs, our modifications won’t be easily mapped to hardware (EduMIPs strays far from the hardware implementation, possibly for a performance gain). For this reason, it’ll be hard to compare the added complexities of our implementations.