EECS 150 Spring 2012 Checkpoint 2: Stall Implementation

Prof. John Wawrzynek
TAs: James Parker, Daiwei Li, Shaoyi Cheng
Department of Electrical Engineering and Computer Sciences
College of Engineering, University of California, Berkeley

Revision 1

1 Introduction

In the next checkpoint, we will move to a memory hierarchy that provides significantly more space at the cost of variable, multi-cycle latency. In preparation for this, checkpoint 2 requires that your processor can be stalled arbitrarily and still correctly execute programs.

2 Stalling the Processor

The `MIPS150` module now has a `stall` input. When this signal is asserted, instructions should not advance in the pipeline and processor state should not be altered. This means:

- Pipeline registers should retain the same value.
- No state elements should change (for example, input registers of the UART).
- Block RAMs should be disabled (using the `ena` port) to prevent loading a new address.

3 Checkoff

The stall signal is toggled every clock cycle in `ml505top.v`. For checkoff, the processor must correctly run `bios150v3` on the board and you must demonstrate in simulation that instructions do not advance when stall is asserted. **Checkpoint 2 is due in lab no later than 2:00 PM, Wednesday, March 21.**