EECS 150 Spring 2012 Project: Cleanup, Optimizations, and Extra Credit

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

The final week of the term project does not have a required checkpoint. Use this week for code cleanup, optimizations, late checkpoints, and extra credit projects. All of this is optional.

2 Cleanup and Optimizations

A small portion of the final project grade will be based on resource usage (i.e. LUT count), codebase quality, and performance.

2.1 Resource usage

LUT counts should be in the range of 5,000 - 6,500. Groups with significantly lower counts may receive extra credit, and groups with significantly higher LUT counts will be penalized.

2.2 Performance

Your design must be able to run at 50 MHz.

To simplify the pixel feeder we did not require enabling the CPU and DVI to run in different clock domains. Adding this capability and scaling the CPU clock could be an extra credit project.

2.3 Quality

Your code should be modular, well documented, and consistently styled. Projects with incomprehensible code will upset the graders.

3 Extra Credit

Teams that have completed checkpoint 5 are eligible to receive extra credit worth up to 10% of the project grade. The following are suggested projects that are feasible in one week. The extra credit amounts given for each category are an upper bound; the actual amount will depend on the

difficulty of the project and the results.

Easy (up to 3% extra credit):

- **Program Profiling**: Add counters to track instruction categories (arithmetic, branch, load/store) as well as cache hit rates. Present this information for mmult and at least one other program.
- **Static Graphics**: Create a C program that displays an interesting image or graphic on the display.

Medium (up to 6% extra credit):

- **Benchmarking**: Port a popular benchmark to run on the MIPS150 processor and report results. A memory-intensive benchmark is preferable.
- **Program Optimization**: The provided `mmult` implementation is not efficient. Optimize the program for your processor and report performance gains.
- **Animated Graphics**: Create a C program that displays an interesting animation or moving image.

Difficult (up to 10% extra credit):

- **Interactive Graphics**: Create a graphics demonstration that reacts to inputs (either from serial or GPIO buttons). Games (e.g. helicopter, pong) will be well-received by the class and staff!
- **Double-buffered graphics**: Implement double buffering and create a simple demonstration showing tear-free graphics.
- **Graphic Terminal**: Show the bios150 shell on the display, using either hardware or software character drawing.
- **Vector image drawing**: Modify the bios to accept a stream of points, where every set of 4 specifies a line. Create a script (based on `~cs150/bin/coe_to_serial`) that sends a file and instructs the bios to draw the lines.
- **Audio output**: Add an audio interface to the CPU.
- **Triangle Shader**: Build a hardware triangle shader.

Detailed specifications for these suggestions will not be provided. As such, we recommend that you consult with a TA before starting the project to make sure you are on the right track.

Teams may also propose their own project; however, it must be approved by a TA to receive extra credit.

4 Final Project Checkoff

To ensure the project grade accurately reflect your work, the final project checkoff will consist of meeting with the TAs to discuss what works, what doesn't, testing methodology, test coverage, and any other relevant factors. Early checkoff may be done during regularly scheduled lab sections. **The final checkoff will be 4-5 PM, April 27.**