Intro to Digital Logic, Lab 6
Final Project

Lab Objectives

Now that you are an expert logic designer, it's time to prove yourself. You have until about the end of the quarter to do something cool with the DE1 board and your digital design skills.

Note that this lab is MUCH more complex than previous labs, and will take a LOT of time. Read the entire lab carefully, and START EARLY. You will need the time!

Note also that there are a bunch of tutorials on the class website for various inputs and outputs you can use on the DE1-SoC board, including video, audio, mouse, and others. Don't use such an interface unless you need it for your project, since it'll be more complex, but they are all tested and usable as needed to come up with something cool. Also, work on getting any interfaces running ASAP, so that if you have problems you can (1) get help from the TA, or (2) switch to another project quickly!

Ground Rules

You will design a significant project on the DE1 board. In some cases you may want to use the breadboard as well – note that all of the pins at the bottom of the breadboard are labeled with the pin they talk to on the FPGA, and thus are usable. This will be most useful to people who want to connect up to more interesting inputs and/or outputs for their design, since all of the digital logic should be done inside the FPGA. You can access these pins by having a bus connection "inout logic [35:0] GPIO_0" to your top-level module (like you used KEY, SW, LEDR, etc. in other labs). GPIO_0[0] is the leftmost connection (AC18), and GPIO_0[35] is the rightmost connection (AJ21).

You may use the clock_divider circuit if you wish. However, your ENTIRE circuit should be based off of EXACTLY ONE clock. If you need two clock speeds, use the clock divider to generate the faster clock. Then, use a counter as a timer to generate an enable signal at the slower rate – all slower elements will still use the fast clock, but will only change state when the slower counter signal occurs. Take a look at the traffic light controller and the timer for an example.

For some projects you may want to add an LED array, or work with peripherals like a monitor, keyboard, speakers, etc. Check the class website for documentation on how to use some of these peripherals.

Grading

Because this lab is significantly more complex than others, it will be weighted much more significantly than a normal weekly lab.

You will be graded 100 points on correctness, style, testing, etc. Your bonus goals is cool/interesting features. For each of these projects, there are lots of obvious ways to make it more useful, more efficient, and more fun. TAs will give up to 20 bonus points for any of these. However, you will get MUCH more credit for a working, simple design than a not-working, awesome system. **The best plan is to get the basic system working, then add any frills if/when you have extra time.**

Extra Credit - Early Finish

All labs are due by 5pm on the first day of finals week. Anything turned in after that time will be considered late, based upon the standard class late penalty.

To encourage people to get started early, and leave time to fix things when the inevitable problems come in, all students should attempt to get their projects finished a week earlier. There is a 15% extra credit bonus for finishing your lab on this date, decreasing by 3% per school day after that. Thus, turning in the lab on the due date gets you 0% extra credit, 1 schoolday before that you get 3%, etc. Demos are done first-come-first-served in the TA's lab hours – if you show up right as lab hours are ending for the day there is no guarantee the TA will be able to demo it that day, so show up early when possible.

Projects

To keep things interesting, we've listed a large number of different possible designs. You can pick any of these you prefer, and in fact can pick the "Venture Capitalist" design that lets you make up your own project. All of these projects are meant to be approximately the same difficulty to complete, though there are no guarantees that we have guessed correctly on this.

Frogger

The urban horticulture program on campus thinks we can eliminate road-kill accidents by training the local wildlife to avoid cars. Your job is to develop a high-tech traffic simulator so they can learn how to safely cross the road (see Wikipedia under "frogger").

You'll need to have several rows of red LEDs, with moving lights to show where the cars are (they'll likely be some basic pattern, with the cars in a row slowly moving to the right or left). You'll also need a green LED for each location as well, to show the position of your frog. Then, the user should have left/right/back/forward buttons to move their frog around. The goal is to get the frog from one side of the board to the other without being in the same square as a car (squish!). They win, they play again. They lose, they become road-kill-souffle'.

You'll need a reasonable sized board (16x16 or so), and the frog should be able to move faster than the cars (otherwise there's no hope).

Conway's Game of Life

I'm sick of liberal arts majors saying we should get a life – we've got Conway's Game of Life: <http://en.wikipedia.org/wiki/Conway's_Game_of_Life>. This is a simple computation that can give rise to very interesting behaviors over time. You need to implement at least an 8x8 array of the life board, and have ways for a player to input a new pattern and then start it running.

Dancing with your Thumbs

Implement a simplified version of "Dance-Dance Revolution". Your machine will have four banks of lights, each bank with at least 5 lights. The 2nd to top light in each bank is a different color. Each bank of lights also has a button. The system will randomly turn on a light at the bottom of a bank, which will quickly move up that bank (one light on at a time, going from bottom to top). The goal is for the user to press the button on a bank of lights right when the light hits the 2nd to top position. If the user times it right, they get 2 points. If they get it in a position next to that goal position, they get 1 point. Pressing at any other point, or letting the light go off the top, you lose 2 points. Your system should keep track of

the score over time, to see how well the player is doing. To make it interesting, multiple lights can go at once, and speed can be adjusted by speeding up/slowing down the machine (manually by the user).

Flappy Bird

The iPhone game Flappy Bird ruined the author's life – why shouldn't it do the same to yours? Flappy bird has our hero the Caped Cardinal (otherwise known as a red dot) flapping through a maze of "definitely NOT a Mario Brothers Ripoff" pipes (otherwise known as green vertical lines). Press the button and the CC goes up, release the button and CC goes down. Avoid the pipes by flying into the holes as you fly sideways through the pipe maze. You need to also keep track of score, which will be a decimal value of up to at least 999, shown on the hex displays of the board.

D.O.T.lite

The traffic lights are wearing out by U-Village and somebody better fix 'em fast! At the corner of Pend Oreille and 25th Ave (NorthEast corner of campus) there's a 4-way intersection. You'll need to be able to handle traffic on both 25th and Pend Orielle, including the turn lane for 25th. BTW, just in case you thought it was too easy, the turn lights on 25th are different in the two directions, and activated by a sensor in the road.

Note that your Professor drives through that intersection daily, so no 4-way greens!

Snakes & Apples

The experts at WSU have a great idea for dealing with Washington State's bumper crop of apples – feed 'em to snakes! (They thought about making them into tea sets, but everyone knows Cougs can't get their hands on an Apple Cup). Let's see what happens.

Develop an implementation of the snake game (see Wikipedia under "Snake (video game)"). The snake (a green dot head plus 2 body segments) moves around the board looking for apples (red dots). When the head hits the apple, the snake scores a point, and grows longer. If the snake ever runs over its own tail, it dies.

Your system should keep score, and should allow the snake to grow long enough for the game to become challenging. Note that the easiest way to build this game is likely to build it like Tug of War – design a cell for each board position, and have it know how long to keep the light on once the snake head passes over that location.

Connect-4

What's better than tic-tac-toe? Tic-tac-toe with gravity! Connect-4 is a four-in-a-row game with checkers sliding down columns. Check it out on Wikipedia.

The game will have two human players with an intuitive method for specifying the column to add a piece, animation of the piece falling down the column, and a winner detection circuit.

Jailbreak

Instead of slamming your head against a wall, how about slamming a small white ball against it? Implement the old classic game Breakout (or Arkanoid without the moving bad guys). See Wikipedia under "Breakout (video game)".

Pro tip: You can define for yourself what happens if, going up, you hit a block on a high row, and then come back down and hit a lower row. Don't know what I'm talking about? Start building this and you'll see…

Drum Sequencer

Beat the curve with your own personal drum sequencer! Using the audio outputs of the DE-1 board (plus the audio driver code on the course website), create a system that can produce a repeating sequence of beats. It must have at least 2 sounds, at least 8 timesteps (times at which either sound could play), and the beat pattern must be programmable. The sequencer will be able to play that pattern repeatedly, until your roommate finally decides to smash it to pieces…

Venture Capitalist

This is a project of your choice. It must be approved by the venture capitalist guys... that would be your Professor... before you can start. Set up a time to meet with your Professor to discuss your plans well in advance. Note: For the Venture Capitalist if you do not have a written project spec with the Professor's signature on it, in advance, you will not get credit.

Lab Demonstration/Turn-In Requirements

A TA needs to "Check You Off" for each of the tasks listed below.

- Turn in a block diagram of your entire system, and a detailed description of how it should operate (i.e. a User's Manual)
- Turn in all design files for your design, including testbenches of the submodules and the system as a whole.
- Demonstrate your complete system to the TA.
- Tell the TA how many hours (estimated) it took to complete this lab, including reading, planning, design, coding, debugging, testing, etc. Everything related to the lab (in total).