Final Project

Final Project Schedule

Project Week 1: 4/1

- Submit a detailed project proposal. See below for a detailed example. Your TA will give you feedback by the end of the week. Note that this is the week of the Lab 9.2 demo. **This is worth 10 points.**

Project Week 2: 4/8

- Work on your final project, lab is optional this week.

Project Week 3: 4/15

- **(Required)** Mid-checkpoint with your TA. You should show tangible progress to your TA, as described in your proposal. **Not showing up will result in at least a 5 point loss of functionality points.**

Project Week 4: 4/22

- Work on your final project, lab is optional this week.

Project Week 5: 4/29

- Demo project. Return lab kit. No Extensions! Reports due at 11:59 PM the following week (week of 5/6) the day of your demo.

General Notes

- Your TA will give you feedback on your proposal. If it is too easy or too difficult, the proposal may need to be modified or entirely redone.
- Start working on your project as soon as your TA approves your proposal.
- Break the project down into milestones. Determine what features are critical, and what features can be cut if you fall behind.
- Create simulations while you work on the project, not afterwards. Without comprehensive simulations, it is unlikely that you will be able to debug your project.
- Get your project running on hardware as soon as possible. Running the code on hardware will allow you to catch problems that the simulations might not reveal.

Final Project Ideas

Term projects can be on any idea you want to pursue (provided they are approved by the instructor or the TA). The students are encouraged to pick the projects based on their interest. Please keep in mind that it is much better to have a working final project than a challenging proposal that doesn't work. Just to get you thinking about projects ideas, here's a partial list of projects. Your proposal should make clear what is software (C code) and what is hardware (SystemVerilog) in your design.

- TTL chip checker that checks the integrity of the chips
- Image/Video/Audio encoding and decoding (JPG, MP3, MJPEG, etc...)
- Encryption/Decryption for secure data transmission (e.g. Feistel, real time pipelined AES)
- Any video/arcade game which uses VGA screen and input devices
    - Arcade classics (Frogger, Space Invaders, Joust, Pacman, Missile Command)
    - Vertical or Horizontal Shooters
    - Tetris
    - DDR/Beatmania with sound
    - Snake - **not recommended, will have 0 difficulty points unless demo is impressive**
    - Breakout/Brickbreaker - **not recommended, will have 0 difficulty points unless demo is impressive**
    - Pong is **generally not allowed** due to similarity to Lab 8 unless it is a significantly unique take (e.g. 3D pong).
- Hardware implementations of classic CPUs or computers (e.g. NES on FPGA, C64 on FPGA)
- **SLC-3 extensions require special approval to ensure they are not identical to work done in other classes (e.g. 411 implements full LC-3, pipelining, cache, so these are not allowed as projects)**
- Audio or music DSP algorithms (speaker correction, reverberation, equalization, sound synthesis)
- Accelerated 2D or 3D graphics (e.g. 3D accelerator for Nios II based SoC)
- Artificial neural network applications (object identification, handwriting recognition, voice recognition)

Additional notes:

- You can also come up with your own project! (Discuss feasibility with your TA)

- Use of keyboards, VGA, RAM and the Nios II in any project is strongly recommended
- We now have in stock a few 1.6 Mega pixel digital camera boards and a few 3.6 inch LCD display boards. Both of these have connectors that directly connect to your DE2 board with ribbon cables. Ask a TA for these boards if you plan to use them in your project. The following two websites give information on our camera and LCD panel. Source for demo designs (in Verilog) are also available there.
  1.3 Mega Pixel Digital Camera Module
  3.6" Digital LCD Panel Kit
- A past student (Koushik Roy) has written a sound driver that you can use if you would like to incorporate sound into your project. The code and the documentation is included below.
  VHDL Audio Driver
  Audio Driver Documentation
- A Font file in the form of sprite table is provided. Please refer to the final project lecture slides on how sprite works.
  Font Sprite Tutorial (by Daniel Chen)
- PS/2 Keyboard Driver by Sai Ma and Marie Liu - supports n-key rollover
- DE2-115 Examples CD-ROM (several examples which may serve as a starting point for your project)
- **Rishi's ECE 385 Helper Tools (e.g. converting images to RAM)**

Assignment

- Design, implement, and debug your proposed final project circuit.
- Work on the final project report (JOINT report).
- Comment, zip and hand in your source files to your TA during the demo. Please include ALL of your .SV, .H, and .C files, including the provided ones and name the zip file such as ECE385_LabX_netID1_netID2.zip so it is distinguishable. Note that the submitted codes will count towards a big portion of your lab report score. You must submit the files EVEN if you did not complete a project, as we will need to look at your code to evaluate your level of understanding of the material.

Grading and Point Allocation (60 points total)

Functionality (20 points)

Functionality points are allocated towards ***completeness and the correct operation*** *of your* ***proposed*** ***design.***

- 5 points are allocated for the mid-checkpoint. This is largely graded on the basis of attendance to the mid-checkpoint and satisfactory progress. Satisfactory progress means that you have largely finished your research phase and have some code to demonstrate.
- 15 points are allocated for the final demo. If your circuit meet the fundamental requirements of your proposed circuit (discuss with TA), you will most likely receive close to full credit. If your circuit meet most of the fundamental requirements but is lacking some minor details or if the circuit is glitchy/buggy, you will most likely receive more than half of the credits. If your circuit is lacking fundamental requirements or if little physical demo is shown other than the written codes, you will most likely receive less than half of the credits. **Note that if you demo a project significantly different than your proposal, you may receive fewer functionality points if what you demo was significantly easier than what you proposed.**

Difficulty (10 points)

- 10 points are allocated towards the ***intrinsic difficulty*** of your proposed design. That is, the complexity of your design/logic/state machine/algorithm inherent to the choice of your project. Note that this may include both technical difficulty and usability difficulty (e.g. points may be deducted for a game which has poor responsiveness or poor framerate). Also, keep in mind that some approaches to the similar functionality may have different difficult levels (e.g. score keeping on the HEX displays is easier than score keeping on the VGA display using font drawing). Ideas to add difficulty can include:
  - Addition of sound/speech
  - Score keeping in game/font drawing/high score table
  - Multiplayer in game
  - AI
  - External hardware
  - Live video
  - Sophisticated graphics drawing

Proposal and Final Report (30 points)

- 10 points are allocated towards your project proposal. This is an example of what a good proposal should include (.PDF link). A good proposal showing sufficient feasibility research is the first step to a good final project. Uploaded to Compass during Project Week 0, same due time as lab report.
- 20 points are allocated towards the final report. The proposal may be a good outline to start the final report from, but the final report should also include all the sections as in the previous 8 lab reports. Uploaded to compass 11:59 PM the day of your demo.

Hints & FAQ

- **"Is there a specific format expected for the Final Project Proposal?"**
  - There is a proposal description provided in a pdf above. The document provides details about what should go into the proposal.

- **"Will we be provided SystemVerilog code for a PS/2 keyboard interface for the final project if we did not get it functioning correctly for Lab 8?"**

    - Yes. See above for the updated SV code for the PS/2 keyboard.

- **"Are we allowed use any existing code online?"**
    - Just like the keyboard entity, it is allowed for you to work on any existing codes and projects (even if it's not yours) as the foundation of your final project, PROVIDED that you inform your TA before using them. There are a lot of good FPGA codes /demoes/projects online that are very interesting, and can be further enhanced to make them even fancier.
      However, being your final project, you will be graded solely on your own contribution and your own effort in the coming five weeks, rather than the fanciness of the base code/demoes/projects. So this is why it is VERY IMPORTANT for you to notify your TA before using ANY CODE that is not yours to avoid being identified as "submitting other's work as your own" (aka plagiarizing)!!!

- **"Is there any tutorial given to us on how to use sprites?**
    - The lectures following the end of Lab 9 will talk extensively about various ways to draw graphics on the screen.

- **"What should I include in my final project report?"**
    - There's no a set guideline for what to be included in the final report since everyone's project is different. However, you have done 9 lab reports for the semester so you should have a pretty good idea about what you should put down in your lab report. Simulation waveform will be necessary unless your project is absolutely not capable of debugging using simulation (say 100% graphically or memory based and no algorithm or control available at all, which is not likely the case). Which means, although you don't have to simulate the graphical and memory interface, you should simulate individual modules if possible. A clear and easily understood block diagram is also necessary, as well as state diagrams of essential state machines. If you used the NIOS II, then code documentation and a description of the hardware/software communication protocol is necessary.

- **"How do I write a ROM file?"**
    - Take a look at the helper tools found here **Rishi's ECE 385 Helper Tools.**

USB Resources:

USB Mouse Interface:

For those of you who wish to implement a mouse interface for your project, this code is provided by Altera for a NIOS mouse and VGA interface.

USB Mouse NIOS interface from Altera

Documentation