CSC 258 Project Information - Fall 2016

1 Project Proposal Form

You will need to submit your project proposal online at `https://goo.gl/forms/OLj2TVYYS79CGVBb2` by

- **Thursday, November 10 at 11:59pm**, if you normally have labs on Wednesdays, or
- **Friday, November 11 at 11:59pm**, if you normally have labs on Thursdays.

There should be **only one project proposal submission per group**!

Print out the confirmation page (or save it), to prove you have submitted your proposal. Make sure that your submission goes through, because TAs will be providing feedback, via email, on your electronically-submitted proposal. The TAs will also be referring to these proposals, as well as the feedback provided to you on things you needed to change, when marking your weekly project progress.

Any proposal that is not submitted by the deadline, will get zero marks on the project proposal mark! As a reminder, your project proposal is worth 2%.

For your project, we will be marking you on whether you meet your personal goals for the project, as well as course-level goals. The course-level goals for the project are:

1. For you to demonstrate your understanding of digital logic, hardware, and Verilog.
2. For you to implement a creative application of these topics.

1.1 Project Scope

The project should include amount of work equivalent to three regular labs in this course. Note that Lab 6 was a below-average difficulty lab. For each week's milestone, your plan should include a reasonable amount of work. You should not plan to do everything in the last week! Doing so will result in feedback from TAs to adjust the milestones, and by the time you read the feedback you may already be behind on your week 1 milestone. If the scope of your project is not appropriate, every effort will be made to provide you with appropriate feedback during the initial project proposal review. In some cases, your TA may provide additional feedback and ask you to further adjust the scope of your project immediately **after** your Week 1 milestone. This may happen if your project proposal was not detailed enough to accurately evaluate the scope of the project. To avoid this from happening, please be specific when you submit your proposal.

1.2 Project Originality and How To Avoid Plagiarism

When thinking about the course project, you should aim to implement something original. One unoriginal project idea that tends to reoccur is the game of Pong. The game of pong is **not** an acceptable project, because it is a simple extension of part III of lab 7.

Do not try to copy ideas from other teams. **Copying another team's proposal would be considered a form of plagiarism, which would be a serious academic offence that would be dealt with very seriously. Similarly, "borrowing" a solution from previous years or other online sources is also plagiarism.** We have access to copies of lots of projects from previous years (i.e., Verilog code), and the TAs will ask you to submit your work electronically if there is any doubt in the origin of your work. Save yourself and everyone involved a lot of trouble and work on your own project. You will find that the experience is very rewarding.

Of course, in a large class like this one, there will inevitably be some project ideas that will be implemented by multiple teams independently. **The TAs will scrutinize project proposals for similarities and flag any proposals that implement similar ideas**. Such teams will be asked to submit their work (e.g., Verilog code and .do files) at the end of the project, and their work will be compared against work of other teams with similar ideas to ensure that all teams worked independently.

1.3 Proposal Content

Your proposal will be the completed google form (link listed earlier) and will contain:

1. Your and your project partner's names, student numbers and UofT emails.

2. The title of your project.

3. One paragraph description of the project.

   Please be specific in your description. This describtion should summarize the basic functionality of your hardware design and explicitly mention how the user will interact with it (e.g., using VGA, PS2 keyboard, switches).

   A **good description example**, if you were describing game of Pong (see above about Pong not being an acceptable project), may look something like this:

   > We will implement Pong, which is a two player game where each player controls a paddle that can move up and down on each vertical side of the screen. A ball bounces across the screen and each player's goal is to cause the ball to touch the opponent's side of the screen and prevent the ball from touching the player's own side of the screen by using the paddle to bounce the ball back into the field. If the ball touches one of the sides of the screen, the other player will score a point and the ball will reset. The game will keep a running score for each player on the HEX displays and will keep playing until one player reaches a predetermined score (e.g., 11). The players will control the movement of the paddles using a set of keys from the PS2 keyboard. If time permits, we will dynamically increase the speed of the ball as the game progresses.

   A **bad description example** (doesn't provide enough information about the implementation and the reader may not be familiar with this game) would be:

   > We will implement the Pong game with a bouncing ball and paddles.

4. What will you accomplish for the first milestone?
   Please be specific and indicate which partner will implement, or take the lead on, which part.

   A **good description example** may look something like this:

   > Implement the following parts of the datapath:
   > Delay and score counter, and collision detection FSM, including simulations and basic testing on the board (Tony)
   > X and Y counters, frame counter, and random number generator for ball direction, including simulations and basic testing on the board (Anne)

   A **bad description example** (not specific, no breakdown between partners):

   > We will implement as many parts of the datapath as we can.

5. What will you accomplish for the second milestone?
   Please be specific and indicate which partner will implement which part, just like for the first milestone.

6. What will you accomplish for the third milestone? (i.e. what your completed project will look like.)
   Don't just say "everything" just because this is the final milestone; describe the final components, indicating which partner will complete which part.

   A **good description example** may look something like this:

   > Combine FSM controlling the drawing with X and Y counters and simulate to ensure it works (Anne).
   > Combine Frame, and delay counter and simulate to ensure it works (Tony).
   > Figure out how to interface with the keyboard (Anne).
   > Simulate scoring counter to ensure it works (Tony).
   > Combine all parts and debug on the board (Anne and Tony).

7. How this project relates to material covered in CSC258. It is important to spell this out. The bulk of the work in your project should be related to course material (digital logic, Verilog, ...).

   A **good description example** may look something like this:

   > Our project uses counters and registers for data storage. It also uses adders to compute coordinates. These components are assembled into two independent datapaths. Several complex FSMs are used to control the datapath and facilitate communication with the keyboard and the VGA.

8. Why this project is cool (for both CSC258 students and non-CSC258 students) and why the idea of working on this appeals to you personally.
   Show off your creativity, and try to have fun with the project. If you pick a project you are excited about, the work will be less tedious and you will get a better sense of achievement!

2 Project Milestones

For your project milestones, we will be marking you as follows.

- Week 1 (worth 3% of your final grade)
- Week 2 (worth 3% of your final grade)
- Week 3 (worth 6% of your final grade)

For each of the milestones, the TAs will evaluate how much of the planned work you completed. In weeks 1 and 2, your work should include simulations showing that components you built operate as expected. It is not necessary to print the simulations, but you should be ready to show them to your TA.

For weeks 1 and 2, TAs will have some freedom to allow you to slightly deviate from the plan. For example, if you found that it made more sense to complete a component from week 2 in week 1, and you pushed another component from week 1 into week 2 to compensate, the TAs may still award you high marks for week 1, **provided that the amount of work for the two swapped components roughly balances out**. If you do significantly less work in one week than your milestone specifies, the TAs will not award full marks for that week's milestone.

For the final milestone, please prepare a brief presentation (4-6 slides) showing the high level block diagram of the circuit you implemented, screenshots of simulations you performed, any difficulties you encountered and how you tackled them, and what you learned. Finally, include a **summary table indicating which partner completed which part of the project**.

Note that for the final milestone, you will be graded against the project proposal, and TAs will generally not allow deviations from the final project plan in the last week. You will be graded based on completeness of your project, but also based on how much testing you have done. You will not be told your week 3 mark immediately (although you will get a general idea). This is because all TAs will meet with the instructors to ensure that week 3 marks are awarded consistently across all teams in the course.

Finally, note that **if TAs find that one lab partner did significantly more work than the other partners, they will award marks proportionately**. This applies to marking of all three milestones.

3 Advice

- Using hierarchical design for your code will be invaluable. Make sure you draw a high-level schematic which specifies the various components and how they are interconnected. It might help if you write down a paragraph about the expected functionality of each of these components (i.e., its specification). Repeat this process iteratively until you have a clear idea of what each module should implement.
- Do thorough testing of each module before you integrate multiple modules in your final design. Having separate simulations of each module using multiple 'do' files will help you with testing.
- If your design includes multiple independent datapath components, consider implementing multiple FSMs to control them. This may make your solution simpler than creating one giant FSM. For example, in the game of Pong, one might choose to control each paddle by a separate FSM, and have another FSM that controls the movement of the ball. Also keep in mind that outputs of one FSM may be inputs to another FSM (e.g. one FSM could produce a signal when it is done with one part of processing, which could be used as a 'Go' signal for another FSM to start its work). Different approaches will be suitable for different projects, so think through different options before committing to a specific implementation, and consult with your TA.
- Coordinating work with your partner is key. Split work. Communicate frequently!
- TAs are here to help you, so make sure to take advantage of that. Ask them for advice, but don't expect them to implement the project for you.
- Do not spend time on components unrelated to CSC258 material as that work will not contribute to your final project mark. In general, avoid using Arduino or Raspberry Pi or complex physical hardware circuits you build by hand.
- If you anticipate a specific component to be challenging, then have a back-up plan. For example, if you want to use the keyboard to provide inputs to your hardware design and this does not work, you can provide input via the switches and/or keys on the DE1-SoC board instead. If possible, implement and test the simple implementation using switches/keys first, then deal with keyboard input after the simple version already works. As another example, if you are implementing a game that will accelerate as the user plays the game, implement the game at fixed speed first, then deal with acceleration after the basic version already works.
- Backup plans are also good for internal features or modules within your system!
- Don't forget to back-up your code frequently. It is also a good idea to make a copy of a file that has working code, before you start modifying it to add new functionality, so you do not accidentally lose your work. Or use version control software.

4 Past CSC258 Projects (Winter 2015)

Title	Collaborators	Video Link
Cyclone Racer	Yaroslav Taben, Qiao Song	https://www.youtube.com/watch?v=5AfWiH-4YLQ
TicTacToe (with graphics)	Roman Polyanovsky, Umut Akkaya	https://www.youtube.com/watch?v=z5ulaAOu7QU
Catching Banana	Bryan (Man Yeung) Li, KwangJern Lee	https://www.youtube.com/watch?v=iP1Z-7yjimM
Keypad Puzzle	Katie Datsenko, Chloe Duan	https://www.youtube.com/watch?v=C1hH2x2ouWI
The Dream Keyboard	Adam Kasztenny, Doga Ister	https://www.youtube.com/watch?v=h8ZHRP-FADo
Digimon Virtual Pet	Claudia Chen, Jerry Wang	https://www.youtube.com/watch?v=QzPu2EivWvg
Rock Band	Mack Heller, Gavin Barill	https://www.youtube.com/watch?v=OoUmot7Mn3A