Final Project Requirement Sheet 100 points

Project Goals

- Summarize all of your course knowledge in one significant coding project.
- A significant portion of your final course grade (15%) is the project component. For your project, you will work as a team or some cases (individually). A project team can have a maximum of 3 members.

Project Details

There are three tracks for the final project, namely, Track 1, Track 2, and Track 3. Of course, if a student would want to pursue a completely different track, this option is also available. Consult with and get the Professor's approval, if you like to develop an idea outside track 1 and 2.

It is required for all students to follow the honor code. Some important points from the class honor code are outlined below for your reference:

1. Students are not allowed to share code files and/or other implementation details outside their team. It is acceptable to have a healthy discussion with your peers. However, this discussion should be limited to sharing ideas only.
2. Submitting a copy of the other team's program(s) is strictly not allowed. Please note that all work done during this project will be an opportunity for team members to learn, practice, and master the materials taught in this course. By doing the work individually, and within their team, students maximize the learning objectives.

At any duration during and/or after the project, students are recommended to team up with the Professor and/or the Technical Leader(s) to clarify if there is any confusion related to the items in the project sheet and/or class materials.

Project Track 1 - Low-end Programming Computing Tool

- The course project in this track must solve one or more low-end computing problems, programmatically using either C programming language or MIPS or both. A few examples, of what we did so far in this direction, is to implement the shift, binary addition, and subtraction operators, converters such as binary to decimal and decimal to binary. In this track, it is important to focus on low-end computing tasks and simulate the execution of those tasks through one or more programs implemented in C and/or MIPS.
- A few examples of course projects that were done by the previous batch students:
    1. Implement a basic version of a scientific calculator, from scratch, with the ability to compute addition, subtraction, multiplication, division, logarithms, exponents (power operator), etc. in C and/or MIPS programming language. In this project, these operators should be developed from scratch. For example, multiply and divide operators should be implemented using the multiply and divide algorithms discussed in class. Addition and Subtraction operators should be implemented using the techniques discussed in our lessons. It is worth noting that the power operator is repeated multiplication, and the logarithm operator is repeated division. In this project, users should be presented with a command-line interface to easily perform any scientific calculations using the operators outlined above. This project assumes that the

built-in mathematical and the operators in math.h library does not exist. This is a very good practice to master the low-end technical concepts and to further refine your C, MIPS programming skills learned in this course.

2. Implement an advanced data converter, from scratch, that performs conversion using binary, decimal, and hexadecimal notations. That is, binary to decimal, decimal to binary, binary to hexadecimal, hexadecimal to binary, decimal to hexadecimal, and hexadecimal to decimal notation. The input and output data may be represented using an array for binary and hexadecimal notations.

Project Track 2 - Low-end Hardware Computing Tool

- The course project in this track is required to develop a one or more circuits to facilitate computation at the hardware level. One example is what we did with the usecase for the rainfall example. In the rainfall example, we developed a circuit based on a series of rules in a C program.
- An example of course projects that was done by the previous batch students:

  1. Grandma Ann, owner and founder of Grandma Ann's Crumbling Cookies, recognizes that to stay competitive in today's marketplace, she must integrate technology into her business in order to keep quality up and prices down. She designed a cookie watcher. The cookie watcher "watches" multiple batches of cookies baking simultaneously. It is precisely timed to an individual recipe's cooking time. The cookie watcher alerts an employee to when a batch is ready to be taken out of the oven, as well as when a batch is ruined (overcooked). It is easily scalable to watch multiple cookie batches at once. Thanks to the cookie watcher, fewer cookies are being burnt, and the cookies are consistently baked the proper amount of time. As a result, Grandma Ann's profits soared. She has since retired to a private island in the Antilles. In this project, you will recreate her cookie watcher circuit in logicly. Naturally, Grandma Ann first tested her design in Logisim and then eventually soldered her own physical version. One way to go about implementing this project, is to use one input to indicate whether that batch of cookies was handled. The other input is the clock signal. The inputs are fed into a counter. The counter's value is used to determine whether the cookies are done and whether they are ruined (overbaked). It is completely acceptable to think and discuss different ideas to develop this circuit.

  A few examples of open source logisim projects that can take a look are:

  1. Tic Tac Toe Game
     `https://github.com/Ishikashah2510/Tic-Tac-Toe-Logisim-`
  2. Paper Scissors Rock
     `https://github.com/woodylouis/PaperScissorsRock-Logisim/blob/master/LogicOverview.png`
  3. Rapid Roll Game:
     `https://github.com/Juandavid716/Rapid-roll-game-in-Logisim`
  4. You can find more open source projects in github by googling the following keywords: **logisim game github**

Project Track 3 - Student-Designed Project

Students will develop an idea for their own project that focuses on one or more real-world topics in the field of low end computing. In this track, if you had chose to implement in C, you are expected to use Dynamic and Heterogeneous Data Stores such as Pointers, and Structs in your implementation. After receiving the course instructor's approval for your idea, you will complete the project and report on your results. For example, an extension of the songs and FBI program?

If you're completely stumped in coming up with a project idea, you can certainly talk to me and we will set up a brainstorming session. Be creative and choose something interesting to you!

- The course project must have a significant implementation part where you will develop and expand on the course topics covered this semester.
- The course project must be extensive enough to qualify as a project (think of work for at least 3 to 4 one-week lab assignments), but not too extensive so that you cannot finish it in the remainder of the semester.

Timeline and Deliverables

The timeline and deliverable details are provided below for your reference:

1. Proposal – Start developing an idea for your final project. Write a 1-page technical report (single or double spaced) of what you propose to do in your final project and submit a PDF copy of your proposal through the GitHub link shared. I don't expect the proposal to be very detailed at this point. But, it should summarize what project you are going to pursue, what you want to do (the real low-end problem you will tackle, how you plan to solve this low-end computing problem, and at least a couple of references to indicate that you have done some research about the problem. **Deadline:** November 18th, 2021, 5:00 PM.
2. Progress Report – Start developing a 5-page technical report (single/double spaced) to document the progress done by the team. By this point, you should have made a good amount of progress towards implementing your project. Were there any unexpected challenges? Did you have to change your initial model/framework or the project skeleton code? Include everything you have done so far in your progress report, even if it is incomplete. No need to include the actual code (unless you want my help with it), just describe what progress you have made with it. Submit a PDF copy of your progress report using the GitHub link shared. **Deadline:** December 3rd, 2021, 5:00 PM.
3. Presentation – Teams can present their course project by recording a 10 minutes video to virtually present their course project. It is expected that students use Slides during their presentation. The deadline for this part is December 8th at 5:00 PM EST. The link below may be used to do the recordings. OBS is another option to record videos.

   https://www.apowersoft.com/free-online-screen-recorder

   By the presentation session, you should have finished implementation, run some basic testing, and done some code or circuit overview. In the presentation, you should describe the motivation, problem definition, challenges, approaches, and results and analysis. Use diagrams and a few bullet points rather than long sentences and equations. The goal of the presentation is to convey the important high-level ideas and give intuition rather than be a formal specification of everything you did. Design at least 6 to 10 slides, including a slide with the title of your project and your name. Also, it is required to show a short demo of your tool at the end of the presentation. Team members should contribute equally to the presentation. You may upload one video with an edited version of the different team members presentation or upload separate videos. The presentation slides and video should be uploaded to the following google driver folder.

   `https://drive.google.com/drive/folders/17j6WAHRL2Zd4i74GFxZWmQy6L4yMJjG3?usp=sharing`
4. Final Report – Start developing a 8-page technical report (single/double spaced) to summarize the technical details of the tool developed by the team. The final report should be clear and well written, which includes no typos or grammatical errors. The report should be written professionally and technically. The deadline for this part is December 8th at 5:00 PM EST. The report should include the following:
   - The motivation for your project. Why is the problem you decided to solve important or useful?
   - Background of the proposed problem. What have others done for it already? Include references.
   - Detailed project overview, a summary of the proposed implementation, and the methodology used. Include pseudocode, diagrams, and examples if appropriate. If you are extending existing work, briefly describe previous work and include references to it.

- Conclusion. Give a short overview of your project and its results. Describe what you learned, what were the biggest challenges, and the biggest rewards.
- For each deliverable, you need to submit a PDF with your report (or presentation slides). For your final report, you need to submit any supplementary material (code, data, a README file documenting what everything is, and how to run your program) to the git repository using the GitHub link.

Grading Rubric

1. Proposal – 10 points
2. Progress report – 20 points
3. Presentation – 35 points
4. Final report and implementation – 35 points
5. Please make sure to include the honor code statement in all submission files.
6. If a student needs any clarification on their project credits, it is strongly recommended to talk to the Professor. The project credits may be changed if deemed appropriate.