Final Project: **Connect Four**

Due Dates:
All due dates will be on the indicated date and the specific time. Late work will result in a zero for that milestone/assignment.

- Project scope, timeline, and testing strategy - 4/16 @ 11:59 PM
- Project Check-In #1 - 4/24 (During Lab)
- Code - 5/1 @ **Noon**
- Demo / Interview Grading - 5/1 (During lab)
    - The demo and interview grading cannot be made up. Plan accordingly.
- Lab Write-Up and Documentation - 5/4 @ 11:59 PM

Prerequisites:

- Solid understanding of C programming
- Solid understanding of Embedded Systems Programming
- Knowledge of Git for version control
- Strong ability to read code and documentation of code you did not write
- Strong ability to integrate and leverage code you did not write
- The ability to consider design constraints and benefits
- The ability to ask for help when needed
- The ability to problem-solve
- The ability to make complex decisions

Background information

As you know, there is no final exam for this class, but we need to assess your knowledge of the concepts and topics presented in this course. This project will do just that while evaluating your ability to integrate and formally document your code.

This Lab will also have written portions. At the end of the project, you will turn in your code, documentation, and a lab write-up. This project should be done independently, but collaborating with others is encouraged.

We will provide only a small amount of optional code for basic usage of the LCD with touch functionality. You may have to explore the documentation and HAL to use the LCD sufficiently.

Project requirements

- You will need to use the RNG peripheral. If you are creating your own project, you can swap out a peripheral for a new one.
- The LCD screen must be used for touch and display
- Scheduling and/or interrupts must be used
- The project and code should be well-documented

- GitHub must be used. Helpful links are below if you haven't used git before. GitHub desktop is okay to use, but ensure you know what basic commands do.
  - https://www.geeksforgeeks.org/working-on-git-bash/
  - https://github.com/git-guides/install-git

Important milestones and descriptions

Late work will result in a zero for that milestone/assignment.

Project scope, timeline, and testing strategy:

This will be what will be shown and delivered on the due date of this final project. Project scope also should explain which peripheral(s) you plan to use and for what purpose. The timeline should include significant events, excluding project check-ins and any other milestone listed on this document. The testing and verification strategy should also be explained. How are you going to ensure your project works as intended? This should be 2-3 paragraphs.

Project Check-In:

This will be a project check-in, during which a member of the instructional staff will check in on your progress and verify that you are on track. It will also be used to see if we need to expand the deliverables or, in some extreme cases, reduce what should be delivered. These should be easy points as long as you've made significant progress.

Code:

The code should be archived and turned into Canvas, similar to how you turned in other Labs.

LATE SUBMISSIONS WILL BE IGNORED. Please turn in your project by this deadline to avoid a zero for your code implementation. **A GitHub link should also be submitted with it!**

Demo / Interview Grading:

The instructional staff will ask you a series of questions. These questions may include, but are not limited to, your implementation of things and technical questions regarding your project. The instructional staff may also ask to see your code physically. You should have that code ready and pulled up in your favorite IDE or coding environment, preferably one with indexing that will allow us to jump to different functions easily.

Final Lab Write-Up:

This will include:

- Project description (high level)
- Project scope and timeline
- Testing strategy
- Project documentation
  - This can be a Doxygen link or additional file(s). If you use a link, make sure it is accessible. We will subtract points if we cannot access it.
  - Coding documentation should consist of a document(s) for your code

        - The document for your code should describe essential functions and their purpose, your coding hierarchy will also be a part of this
    - The instructional staff should be the target audience for both documentation pieces. In other words, you can use technical terms without explaining what those terms mean.
- Struggles and Obstacles and how you overcame them
- What you learned about this project
- How could your project be improved?

Most of these can be brief. Part of being an engineer is conveying information as quickly and efficiently as possible.

Valuable tips and tricks

- Make your git repository FIRST, then make the STM project in the cloned repository
- Start early, and do not procrastinate.
- Utilize version control, you would be surprised by how easy it is to regress and worsen your code with a seemingly innocuous change.
- Have clear expectations for yourself and your code. Try to avoid lazy coding implementations.
    - This is why modular coding is essential and very useful.
    - Code integration and testing can be seen as its own development step when rolling out a new project or coding "feature" (as it is referred to in the industry)
- Never code and do intense debugging in the same sitting.
    - This sounds weird, but you need to realize that when programming, you are the murderer, the victim, and the detective. You need to debug your code as if it were someone else's and assume they are doing everything wrong.
- If you are stuck, ask for help. Sometimes, interesting things happen "behind the scenes" of embedded programming that some people only know about because they have dealt with it before.

Acceptance Criteria

A. The first screen:
    - It should have at least two buttons shown on the touchscreen:
        - One-Player Button: This will be pushed to play the game in one-player mode!
        - Two-Player Button: This will be pushed to play the game in two-player mode!

B. The Second Screen:
    - This will be the "Gameplay" Screen; this will be where the game is actually played
        - The Board Size must be 7 columns x 6 rows (this is the standard size)
        - The coins' colors can be any scheme you want, but they MUST be clearly opposite colors: red/yellow, Red/Green, or black/White. The board can

also be any color of your choosing, but it must be obvious where the holes are.

    - The coin should be over the board, dictating whose turn it is . You will use the touch screen to move the coin left and right.
        1. Touching anywhere on the left part of the screen will move the coin to the left.
        2. Touching anywhere on the right part of the screen will move the coin to the right.
    - The button will drop the coin to the bottom of the selected column. You **do not** have to show it dropping in the respective column; it can just “teleport” to the specific spot.
    - A user or AI should not be able to place a coin in a column already filled to the top with coins. It should just not let them set a coin there. You do not need any special message or something to tell the user they cannot place a coin there.
  - For 1-Player Mode:
    - Using the RNG (Random Number Generator), simulate the computer's movement.
        1. If you are doing the AI extra credit, you do not have to do this.
    - The user should then be able to place their block. It doesn’t matter who goes first or who is which color. You technically can “randomize” that as well if you choose.
    - The game concludes when someone gets “Connect 4” or if it is a tie.
  - For 2-Player Mode:
    - It doesn’t matter which player goes first; the game will alternate players until the game is complete, i.e., Connect 4 happens, or it is a tie.

C. The Final Screen:
  - This should display data calculated during and across plays (Not across physical device resets). There should also be an option to start a new round.
    - This should say the running score between red and yellow (or whatever color you choose). You do not have to include ties if you do not want to.
    - This should also show the total time in **seconds** it took to play that round.
    - There should also be a button to restart the round.

D. The GitHub was created and accurate
  - You should just upload your code. There is no need to upload the HAL.

E. Specified documentation is created, updated, and accessible.

F. Extra Credit Opportunities:
  - (30 points) Utilizing the Gyro to move the coin before placing
    - Tilting the board to the left would move the coin to the left
    - Titling the board to the right would move the coin to the right
  - (20 points) Having an “intelligent” AI for 1-player mode:
    - This AI should actively play offense and defense. Hint: There *may* be a mathematical approach that could be attempted for this…

It is recommended that you get the base project working and THEN try to implement it and take a shot at any/all of the extra credit. Partial credit will not be awarded for

Cheating or Academic Dishonesty will result in you getting a zero for your code submission AND your interview grading.

Grading rubric:
This lab will be worth 250 points with the potential of 50 extra credit points. The breakdown of grading will be given below.

1. Code functionality (150 points)
    a. Acceptance Criteria A (25 points):
        - All the acceptance criteria for A are met. This is all or nothing.

    b. Acceptance Criteria B (85 points):
        - Gameplay functionality (30 points)
            1. Touching the screen moves the coin in its respective direction
            2. Pushing the button drops the coin to the bottom of the appropriate column
            3. All of these must be present to get the points.
        - 1 Player Gameplay (40 points)
            1. The game concludes if there is a tie (all holes are filled without a Connect 4) or if there is a Connect Four.
            2. The RNG (Or AI if doing Extra Credit) is correctly used to generate AI's move.
            3. A coin is set for each turn and does not overlap/replace an already-placed coin.
            4. All of these must be present to get the points.
        - 2 Player Gameplay (15 points)
            1. The game concludes if there is a tie (all holes are filled without a Connect 4) or if there is a Connect Four.
            2. A coin is set for each turn and does not overlap/replace an already-placed coin
            3. All of these must be present to get the points

    c. Acceptance Criteria C (40 points):
        - Time is accurately displayed (35 points)
        - Accurate Scoreboard (15 points)
2. Project Writeups (100 points)
    a. Scope, Timeline, and Testing Strategy (10 points)
    b. Project Check-in #1 (15 points)
    c. Lab Write-Up (45 points)
    d. GitHub Link (15 points):
    e. Code Documentation(15 points)
3. Interview Grading (50 points)

    a. Even if your project does not work well or at all, the instructional staff will ask questions regarding the course and your project. Failure to attend this lab period dedicated to interview grading will result in losing all 50 points.
    b. These questions/answers should be more open-ended, so as long as you did the project, these should be easy points.
    c. **During Interview Grading, inform your interviewer if you did any extra credit or not!** This will make sure we grade appropriately.
4. Extra Credit (Up to 50 points)
    a. Utilizing the Gyro to move the coin before placing (30 points)
    b. Having an "intelligent" AI for 1 player mode (20 points)
        - You will automatically be awarded points for using the RNG