# ECE 484W

## Final Project
## Transmitting Overlay & Image Processing to On-Board Camera Video Feed

**Presentation (Team): Due by date listed on Canvas. Each team needs to submit one video (between 5 and 6 minutes) where two people take turn to present slides. You can use zoom (One is the host and allows both to have privilege to share screen. Then you can record the presentation). Late submission is not accepted.**

**Demonstration (Team): Due by date listed on Canvas. One video (less than 3 minutes) to demonstrate your results related to each requirement. Late submission is not accepted.**

**Report (Individual): Turn in your report via Canvas (each individual needs to do it; Late submission is not accepted.)**

**If you submit your report/presentation/demonstration multiple times, I will grade the last attempt and your submission time will be the time for your last attempt. For presentation and demonstration, only one of your team members can submit videos for the presentation and demonstration but must indicate who your teammate is in a write-up.**

**Requirements:**

Using the on-board camera and LCD, your design must allow you to pass camera data through the NIOS processor on the on-board monitor. Use your GUI and Wi-Fi communication from previous assignments to allow a user to:

1. Add an overlay image to every frame of the video stream from the on-board camera. The overlay image will be transmitted from the GUI on the computer.
2. Adjust brightness and contrast of the on-board video stream from the GUI. The brightness and contrast values will be transmitted from the GUI on the computer.
3. Show the result on the on-board LCD screen.

Note: No open-source code.

As with the previous assignments: Put a copy of your code in the Appendix with detailed comments. Follow the report guidelines in the syllabus as there is a lot of emphasis on “why” and “explaining the results”. In the alternative design section: state the problem, and how the alt. design overcomes it with experimental results as proof as to how much improvement it makes.

# <u>Grading Rubric:</u>

**Demonstration and Presentation: 15% of Final Grade (Group)**

***Presentation Format:***

1. Each group is allotted 5 minutes, to be divided equally among the group members. Each group should give a coherent presentation, which means unless it is a medical or family emergency, each person must present a portion. If anyone does not show up, the remaining team members must cover his/her portion. Your grade will be based on the group performance. If you do not show up, you will receive 0 points, unless it is a medical or family emergency.

2. Format:
   a. Introduction
      - Scope of this project
   b. Design Methodology
      - How to accomplish each requirement (3 requirements in total).
   c. Alternative Design
      - What item you want to optimize? How you optimize your design? Proof of improvement. (Experimental results to compare it to the original design).
   d. Conclusion

3. Flow + timing
   **a.** <u>**You have 1 minute flexibility (a minimum of 5 minutes). Any contents after 6 minutes does not count.**</u>
   b. Each person needs to take a part of the presentation.

***Demonstration Format:***

- Similar to previous assignments:
  - Record a short video showing your design fulfills all of the above 3 requirements

**Report: 20% of Final Grade *(Individual; Late submission is not accepted)***

Format is as the same as the lab assignment. However, you need to emphasize your contribution and have adequate write-up to address each requirement for the final project. It is an individual grade.

As with the previous assignments: Utilize the “ifconfig” command to view the assigned IP address of the board to modify your GUI/Python Receiver program with each time!

**The Grading Rubric for the Presentation and Demonstration is Shown Below:**

| Concept/Category | Unacceptable (55%) | Marginal (70%) | Acceptable (85%) | Exceptional (100%) | Points |
|---|---|---|---|---|---|
| Design Methodology<br>(30 points) | Lacks understanding of the design problem. Partially finish any one requirement | Shows understanding of the design problem. Completely finish any one requirement.<br>Partially finish any one requirement. | Understands the design problem and objective. Completely finish any two requirements. | Shows good understanding of the design method. Completely accomplish three requirements | |
| Alternative Design (30 points) | Lack of alternative design in the design | Shows understanding of technical problem; Marginal use of alternative design without proof. | Adequate use of alternative design which are reflected in the final design with marginal proof. | Design takes into account all problems and the final design is optimized with proof | |
| Demonstration<br>(25 points) | Partially finish any one requirement | Completely finish any one requirement.<br>Partially finish any one requirement. | Completely finish any two requirements. | Completely accomplish three requirements. | |
| Presentation Format<br>(15 points) | Poor organization, timing and formatting. Mechanical errors | Some degree of organization. Proper formatting, timing and relatively error free. | Well organized and properly formatted and error free. Good timing. | Communicates the key concepts effectively. Well organized and properly formatted and error free. Good timing. | |

**<u>Important Suggestions:</u>**

1. This assignment requires running two programs simultaneously on-board alongside the GUI program on your PC. (This requires opening 2 separate LXTerminals)
    1. Python Receiver slightly modified from previous assignment
    2. Modified pre-existing Camera Executable

2. Study and then modify the pre-existing example camera program on-board as a foundation for your assignment (functionality of accessing the camera is already provided here)
    1. All modifications should be made to the camera_in.cpp file, **each time it is altered, the "make" command should be entered to recompile the executable.**
    2. Cv2 image processing library is highly recommended.
    3. This can be found in VEEK-MT2S_v.1.0.3_SystemCD/Demonstration/SoC_Advanced/OpenCV/camera_in
        I. A more complex example can be studied in ./face_detection which draws images from XML files onto a face that is detected with an algorithm.

**<u>Notes:</u>**

- First, as a sanity check, run the camera application or control panel from the VEEK-MT2S LCD screen to ensure the camera is functioning correctly.

- As with the last assignment, it is suggested to run the programs on the board directly with a USB keyboard rather than through a PuTTY terminal; however, it may be possible to establish a SSH connection with the PuTTY terminal (rather than the usual serial) and run the programs in this manner.

- Depending on your design, it may be important to make note of file paths and where files such as pictures or text files are being stored from the Python receiver and if the camera executable can properly access this data.

- Additionally, depending on how your GUI is implemented, no modifications may be necessary. The only necessary modifications may be to your Python Receiver program and the on-board Camera .cpp file.

- A highly recommended function to study for this implemented is "addWeighted()" to perform image manipulations directly on-board.