# Audio Processing Project

EELE 468
SoC FPGAs II: Application Specific Computing

---

## Grading policy

- Completing the base requirements: 90%
- Each additional proposal: 10%
- Multiple additional proposals can be made, resulting in a grade that can exceed 100%

## Base Requirements

- Each group member needs to create a Simulink model that will be put in the FPGA fabric.
- Each sound effect model needs
  - to be controlled from Linux using a device driver.
  - to have an *enable* functionality than enables/disables the effect. This can be done in software or hardware (e.g. you could control an enable register from software, or you could hook up the switches on the audio mini as your enable switches.)
  - at least one control parameter (register) beyond the *enable* control.
- Each sound effect needs software that demonstrates the sound effect, e.g. a script that changes the effect's control parameters.
- The sound effects' device drivers need to load on boot using a systemd service.

The feedforward comb filter from lab 5 can be used as one of your group's Simulink models.

If your group only has one Simulink model, you will lose 10% of the "base requirement" grade. However, that 10% can be made up with an additional proposal.

## Proposals

In addition to the base requirements, each team needs to propose at least one additional feature. Multiple groups can have the same proposals.

Submit your proposal(s) to the "Additional functionality proposals" assignment folder on Brightspace.

Proposals are due by **May 3, 11:59 pm**.

Ideas include

- Additional hardware: pots, buttons, etc.
- Using device tree overlays to individually load each sound effect on your board.
- A terminal user interface (TUI) for your sound effects.

  - Example Python library: https://github.com/Textualize/textual
  - Example Python library: https://docs.python.org/3.10/library/curses.html
  - Example node.js library: https://github.com/chjj/blessed
  - Example Rust library: https://github.com/fdehau/tui-rs
  - Hackernews thread with some other libraries: https://news.ycombinator.com/item?id=24412687
- A simple web application to control your sound effects. For example:
  - Use the Flask or Django libraries in Python
  - Use a javascript framework to create a single page application (SPA)
- Put the signal energy monitor (lab 1) in the FPGA fabric
- Adjust the volume of your output audio and display the (relative) volume on some LEDs.
- Display some sort of status or register information on LEDs.
- Design and 3D print an enclosure for the DE10 Nano.
- Utilize the fact that you have a full embedded Linux computer, e.g. write some sort of software in a language that isn't C.

## Teamwork Logistics

You can split the work up however you like. For example, if one person loves Linux device drivers and the other person hates Linux, the Linux person could write all the device drivers.

However, the work should be split fairly equally. If team member contributions are significantly unequal, separate grades will be given for each team member. If you think your team's contributions deserve separate grades, please contact me.

## Project Demonstration

Demos will be done during Finals week (or earlier if you're done early). Demos need to be completed by the end of **Wednesday May 11**.

Schedule a 30 minute time slot for you demo: https://www.montana.edu/scheduler/login/student/?fac=851

### Demo steps

1. Show your Simulink model and briefly explain your sound effect.
2. Show your Platform Designer system and Avalon wrapper VHDL code.
3. Show your device tree and device drivers.
4. Demonstrate your sound effect on your board:
   (a) List the contents of `/sys/class/misc/<sound-effect-name>`.
   (b) Enable/disable the sound effect.

(c) Change your sound effect control parameters.

5. Show and describe your extra functionality.

## Submission

Submit your project files using your GitHub repo:

1. Make sure all relevant files are committed to your `main` branch.
2. Document your project using READMEs! Modify this README to give an overview of your project and the repository. Add other READMEs as needed or desired. The goal is for people to be able to understand and use your project by reading the documentation; your documentation doesn't have to be super thorough.
3. Create a GitHub release your submission. @mention me in your release notes.