Lab 5 - Audio

Goals

- Interact with audio on the Microbit
- Play arbitrary tones using PWM
- Play analog waveforms using PWM

Equipment

- Computer with build environment
- Micro:bit and USB cable

Documentation

- nRF52833 datasheet: https://infocenter.nordicsemi.com/pdf/nRF52833_PS_v1.3.pdf
- Microbit schematic: https://github.com/microbit-foundation/microbit-v2-hardware/blob/main/V2/MicroBit_V2.0.0_S_schematic.PDF
- Lecture slides are posted to the Canvas homepage

Lab Steps

1. Update your local repository

- `cd` into the base of your repo
- `git pull https://github.com/nu-ce346/nu-microbit-base.git`
- `git submodule update --init --recursive`
    - There may be no changes

2. Play tones over speaker with PWM

- cd into `software/apps/pwm_tone/` You'll be editing `main.c` here

- Initialize the PWM peripheral

  Use the `nrfx_pwm_init()` within the `pwm_init()` function

  - Documentation for the nrfx_pwm driver

  - You will need to make an `nrfx_pwm_config_t` local variable to pass into the initialize function
    - `output_pins` is an array of four GPIO pin numbers, use SPEAKER_OUT for the speaker pin and NRFX_PWM_PIN_NOT_USED for unused pins
    - `base_clock` should be 500 kHz
    - `count_mode` should be Up
    - `top_value` is the default value for `COUNTERTOP` which we will overwrite later
    - `load_mode` should be Common
    - `step_mode` should be Auto

  - We don't care about callback events for the PWM, so passing in `NULL` for the callback is fine

- Play a tone using the PWM peripheral

  To do so, you'll need to fill in the code for `play_tone()` and call it from `main()`

  - To stop the PWM, use `nrfx_pwm_stop()`

  - Use `NRF_PWM0->COUNTERTOP` to access the `COUNTERTOP` register

  - The PWM sequence variables have already been created for you globally. You will need to edit the value in order to set a 25% duty cycle
    - Note: you don't have to think about left or right alignment here, just set a value so that the toggle occurs at 25% of the way counting up to `COUNTERTOP`

  - To start the PWM, use `nrfx_pwm_simple_playback()`
    - The instance is defined for you at the top of the file

  - Here is a list of frequencies for the tones of a piano

- Play multiple tones to form music
  - You can play whatever you want as long as it consists of at least four distinct notes, so feel free to play a short jingle or song
  - If you don't have any particular music in mind, the comments in main right now guide you through the A major arpeggio scale (A_4, $C\#_5$, E_5, A_5), which is sufficient
  - You almost certainly want to stop the PWM after the last note so it doesn't continue playing it forever
- **Checkoff**: demonstrate your code and app to course staff
  - If it is too quiet to hear, you might have to bump duty cycle back up to 50% for the demonstration

3. Record audio and play it back over the speaker

- cd into `software/apps/record_and_play/` You'll be editing `main.c` here

- The ADC side of the application has been provided for you

  Look through the code already in the file. You shouldn't have to change any of this, but you'll definitely have to interface with it, so understanding what it is doing will be helpful.

- Initialize the PWM peripheral

  Edit the function `pwm_init()` to do so

  - This will be almost the same as the initialization from the previous part of the lab except that the PWM clock should be set to 16 MHz

  - Also you will need to actually set a COUNTERTOP value here

- Play audio samples over the speaker using PWM

  Edit play_audio_samples_looped() to do so

  - First, modify each sample in place so it represents a PWM duty cycle rather than ADC counts. There is a `#define` for the maximum ADC value at the top of the file
    - Again, you don't have to worry about alignment here. Just set each value so that the toggle occurs at a duty cycle proportional to how large of an analog value you read

  - Next, create the PWM sequence. You can copy some of this from the previous part of the lab.
    - You almost certainly want a non-zero repeat count. If repeat is zero, each analog sample is represented by exactly one PWM period. Instead, we want each sample to be represented by a few periods so that there is more than a single waveform to average energy across
    - Increasing the repeat count will require modifying `COUNTERTOP` however. Otherwise your playback will be at an incorrect frequency
    - Note that the value is the number of repetitions after the first play (so the total number of times a sample is played will be 1+`repeats`)

  - Finally, start the playback. You should loop the playback forever with the proper flag

- Test the record and play application

  That should be everything needed to make the application work. You'll need to speak relatively loudly and pretty close to the Microbit in order for the output to be audible.

  - If the output is loud and the correct speed, but distorted, you may be clipping the signal (it may have reached max). Try speaking a little softer

  - If the output is slow, you likely miscalculated the `COUNTERTOP` value, leading to the output playing at a slower frequency than it was recorded

  - You might want to make it so that pressing a button stops the PWM from playing so that it doesn't keep repeating you forever

- **Checkoff**: demonstrate your code and app to course staff

Lab Checkoffs

You must be checked off by course staff to receive credit for this lab. This can be the instructor, TA, or PM during a Friday lab session or during office hours.

- Demonstrate the pwm_tone code and application
- Demonstrate the record_and_play code and application

How to demonstrate audio over zoom

- Zoom likes to filter out background noise, which our audio output very much looks like. To make the audio easier to hear:
  - Click the up arrow next to the microphone mute button in zoom
  - Choose "audio settings…"
  - Below the "Suppress Background Noise" option, there is an advanced option you should click
  - There should now be an option that should be worded something like Show in-meeting option to "Enable original sound from microphone". Check that option. (no need to check any of the options beneath it)
  - Close the audio settings window
  - In the main zoom window, the top left should now say "Turn on Original Sound" (or something similar) Click that
  - Your lab should now be demonstrable over zoom!

Also, don't forget to answer the lab questions assignment on Canvas.