California State University, Long Beach

CECS 447

Project 1

Rodrigo Becerril Ferreyra
Student ID 017584071

A project that uses the ARM Cortex-M4 microcontroller to play music.

15 September 2021

1 Introduction

The purpose of this project is to play music using a microcontroller. The project is split into two parts: first, using a digital output (square wave) to drive a speaker, and then using an analog output (sine wave) coupled to a passive digital-to-analog converter (DAC).

2 Operation

Since this project was split into two parts, I will describe the operation of both parts separately.

2.1 Part 1

The first part brings the task of driving a speaker by sending square waves to the speaker; that is, it uses one GPIO pin to drive the speaker, and it is turned on and off. The frequency of the note played is equal to double the frequency that the output flips its value; if the output switches at a frequency of 440 Hz, then the note played will be of a frequency of 880 Hz.

We are using the SysTick timer to control the frequency of each note played. Assuming a default system clock rate of 16 million cycles per second (16 MHz), the formula to find the reload value for the SysTick timer comes out to be

$$v_1 = \frac{16\,\text{MHz}/f_{\text{note}}}{2} - 1 \tag{1}$$

where v_1 is the reload value and f_{note} is the frequency of the desired note.

The main function of Part 1 is to play three songs: the Happy Birthday song, Marry Had a Little Lamb, and Twinkle Twinkle Little Star. The song that's currently playing loops indefinitely until one of the buttons are pressed: the left button starts or stops the song, while the right button changes the song.

Here is a video detailing the operation of Project 1 Part 1: https://youtu.be/ltoyG6DEJn0

2.2 Part 2

For Part 2, instead of driving the speaker with a square wave, the task at hand is to output a six-bit sine wave. The sine wave is to be outputted in 64 discrete intervals per 360° of sine wave. This necessitates a DAC, to convert a digital number from 0 to 63 into an analog voltage to drive the speaker.

Instead of using an IC DAC circuit that uses serial communication, I used an R-2R passive resistor ladder DAC circuit instead. This simplicity comes with the tradeoff of having to amplify the output before it is able to drive the speaker.

Since the output changes 64 times per sine cycle, the new reload value v_2 changes into

$$v_2 = \frac{16\,\text{MHz}/f_{\text{note}}}{64} - 1 \tag{2}$$

In addition to this, Part 2 has two modes: a music box mode similar to Part 1, and a piano mode. In the music box mode, the right button changes the song, and the left button switches to piano mode.

In piano mode, the system takes input from four buttons; these play the notes C, D, E, and F in the octave that is currently selected. The right button changes octave (there are three total), and the left button switches back to music box mode. Note that the songs played in music box mode will be played in the selected octave (the program remembers the octave it was in).

Here is a video detailing the operation of Project 1 Part 2 when in music box mode: https://youtu.be/4n18laB_mus

Here is a video detailing the operation of Project 1 Part 2 when in piano mode: https://youtu.be/qKPznZU5S7s

3 Theory

Since we are already using the SysTick timer to control the frequency of the note being played, we have to rely on a software loop to control the duration of the note (using a general-purpose timer would be overengineering, since the complexity it would add would supersede the benefits).

4 Hardware Design

The speaker I am using is rated at 4 Ω up to 3 W. The low impedance posed a serious problem in the second part, since I am rather unexperienced with amplifiers and amplification. It took me a lot of work to create an emitter follower with the small-signal transistors and low-power resistors I have at hand. Figure 1 is the emitter follower I have built. V_{in} represents the output of the DAC, which is a sine wave with 3.3 V peak-to-peak and a DC offset of 1.65 V at 440 Hz.

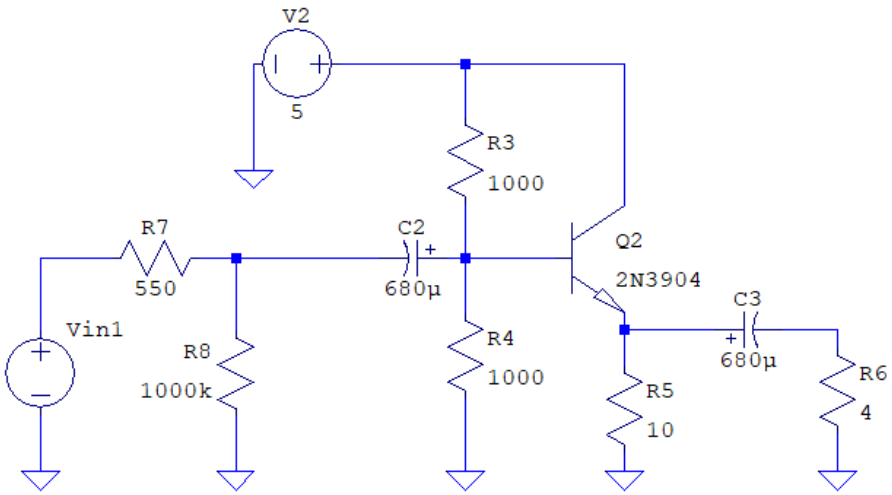


Figure 1: Low-power emitter follower.

The resulting output has a voltage of about 282.84 mV and 70.71 mA, which is a power output of approximately 20 mW. This is audible to the ear in a standard environment, but this is much quieter than other students' pre-built amplifiers, and it is hard to hear through Zoom (so much that I had to put the microphone directly on the speaker so it would pick up the speaker's vibrations). I could make this amplifier much louder if I had a power BJT and some high-power resistors at hand.

Figures 2 and 3 detail the hardware schematics and embedded system pictures of Parts 1 and 2, respectively.

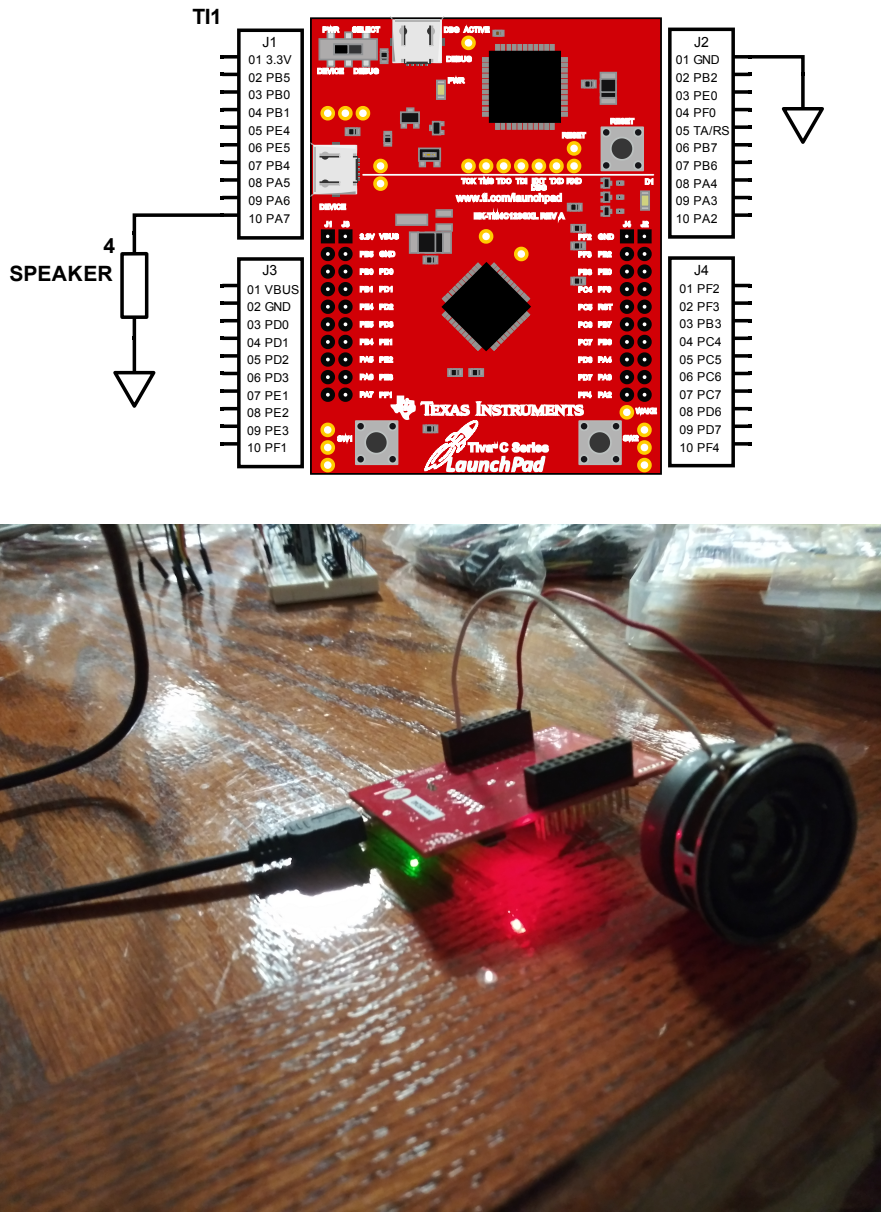
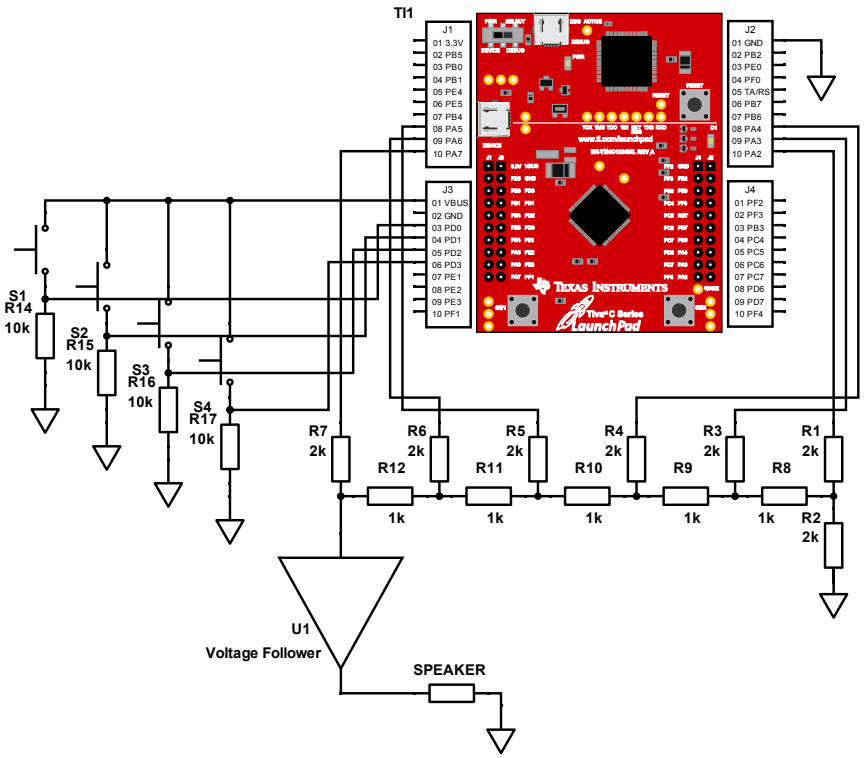


Figure 2: Schematic diagram and picture of Part 1 implementation.



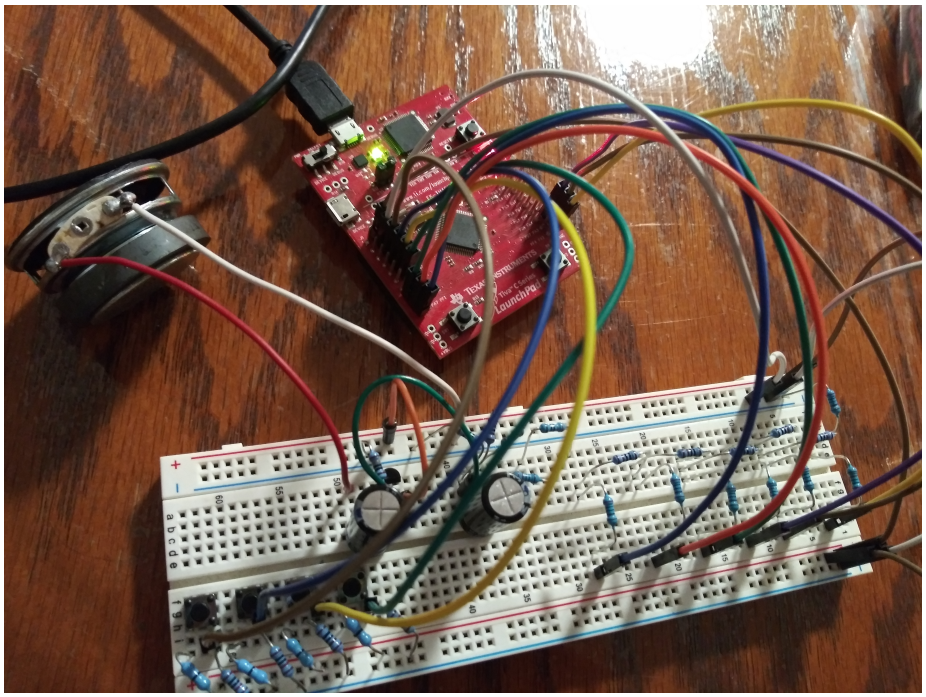


Figure 3: Schematic diagram and picture of Part 2 implementation.

5 Software Design

I used structs to define a `Note`, which have a reload value and an amount of time to hold. Each song is an array of `Note`s; to play the song, a function loops through this array, sets the SysTick timer's reload register to whatever the `Note` requires, and turns the timer on. The timer stays on until the software loop finishes (i.e. until the correct time as defined by `Note` has elapsed). There is a special "note" which signifies the end of song, in which case the same song is repeated.

The on-board buttons trigger interrupts to perform different tasks, depending on the Part: they can force the song to stop playing, change song, or change octave. The on-board LEDs are signals that signify different things: in Part 1, the red LED turns on when the output is high and turns off when the output is low; in piano mode, the LED is blue when the program is in piano mode, and red when the program is in music box mode.

6 Conclusion

This first project is more like two, and it brings many old and new concepts together. I had much trouble with the voltage follower. I also had trouble with my notes being out of pitch during the live demo, though they sound better in the video. Overall, this was a challenging but fair project.