**Design Rationale**

**Overview**

To aid students in learning analogue electronics, it was crucial to include hardware associated with the ENEL270 labs such as OPAMPs, transistors, and sensors. In recent years, COVID has prevented certain labs from running. To resolve a part of this issue associated with the access to lab equipment, [the board] also features hardware that would mimic the ENEL270 lab equipment. The lab equipment would include a function generator and an oscilloscope. Fundamentally, the consideration for analogue electronics would only require choosing discrete components for each of the three components. However, the dispute rose as there were consideration for two pieces of hardware blocks that fulfilled the requirements of the function generation and data acquisition separately, or one piece of hardware block that could operate both for function generation and data acquisition. The different sections will cover the design rationale such that the requirements were met and resolve the dispute between utilising one or two pieces of hardware blocks for function generation and data acquisition.

**Function Generator (in the final requirements, change the sine wave and square wave to be hard requirements and the triangle wave to be soft requirements, also change the THD for sine wave to be less than 1%)**

In the ENEL270 labs, the function generators were used mostly to create sine waves. The occasional square wave was used for certain OPAMP and transistor circuits. Triangle waves could prove to be handy in certain situations but were not necessary to fulfil the requirements of the ENEL270 labs. Therefore, it was essential that [the board] can generator sine waves and square waves so that students can do the labs off campus. [The board]’s function generator should be able to generator frequencies between 10 Hz and 10 kHz to meet the common frequencies used in the ENEL270 labs. The last consideration for the function generation was that the “students can start the lab immediately after they have plugged the board into the computer, installed and opened the software. This means that the function generator must be easy to operate just as the lab function generators are.

Option 1

The first method of sine wave generation was fully analogue. A method that would bring many veteran electrical engineers to tears as they remember the nostalgia of the “good ol days”. This method includes generating a square wave and using a low-pass filter to create a sine wave.

Diagram

Description automatically generated

Figure 1: Oscilloscope FFT of a 10 kHz square wave. Each horizontal division is 10 kHz

Figure 1 shows that a square wave is made up of odd-order harmonics of the fundamental frequency. Those frequencies above the fundamental frequency can be filtered out with a low-pass filter to retrieve a sine wave. This is immediately an extremely viable option as this method provides both the square wave and sine wave. On a hardware level, there are many ways of producing the sine wave since it is a well-established foundation for analogue electronics. Analogue timers, astable multivibrators, and OPAMP self-oscillators are among the more common methods. The foundational operation of the analogue function generators depends on varying the time-constant of a resistor-capacitor network. Hence the most common way method of altering the frequency was to use fixed capacitors and vary the resistance with a potentiometer. The low-pass filter could be created in a variety of ways. The most common two methods were using a resistor-capacitor low-pass filter and an inductor-capacitor low-pass filter as they were the most inexpensive options. However, to generate a sine wave that was adjustable, it was more suitable to use an OPAMP 2nd order filter.

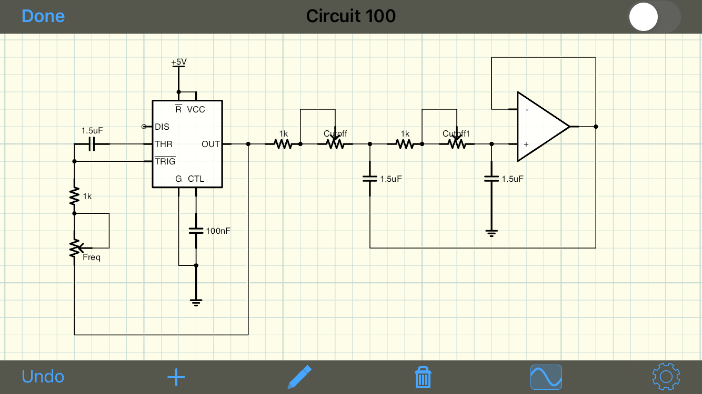


Figure 2: Schematic of a 555-timer based square wave and sine wave function generator.

However, to generate a sine wave that was adjustable, it was more suitable to use an OPAMP 2nd order filter as shown in Figure 2.

Table 1: Bill of Material for a 555-timer based square wave and sine wave function generator.

|  |  |  |
| --- | --- | --- |
| Item/Component | Quantity | Price (in bulks of 100) without GST |
| 555-timer | 1 | $5.40 |
| 15 nF capacitor | 3 | $0.6 |
| 150 nF capacitor | 3 | $0.66 |
| 1.5 uF capacitor | 3 | $0.78 |
| 1 kΩ resistor | 3 | $0.15 |
| 10 k potentiometer | 1 | $1.15 (bulk 230) |
| 10 k dual-gang potentiometer | 1 | $3.24 |
| TLV272 | 1 | $0.74 |
| 100 nF capacitor | 1 | $0.22 |
| Total |  | $13.04 |

Table 1 shows a list of all the components with decade selection using hot-swappable capacitors. Despite the fully analogue and fully discrete method theoretically being able to meet the frequency range and wave function requirements, this was not the final option. The square wave generation would pass the requirements. However, the sine wave generation requires excellent filtering. The OPAMP active circuit shown in Figure 2 forms a 2nd order low-pass filter that is theoretically able to attenuate the square wave harmonics at 100 kHz by 40 dB.

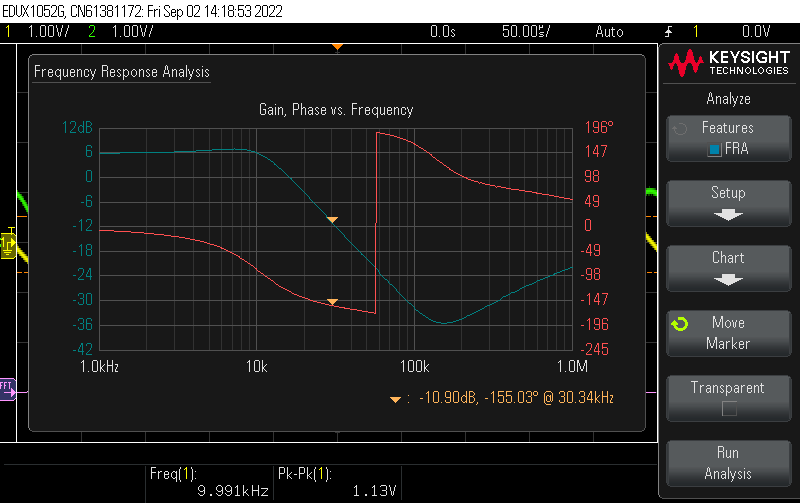


Figure 3: Bode plot of a 2nd order filter with a cut-off frequency of approximately 10.6 kHz.

Figure 3 proves that the 2nd order low-pass filter sufficiently attenuated 40 dB at 100 kHz. The bode plot shows that the first odd-order harmonics at 30 kHz –which is present in a 10 kHz square wave– is attenuated by approximately 17 dB from passband. It is evident that more filter stages were needed for a less distorted sine wave.

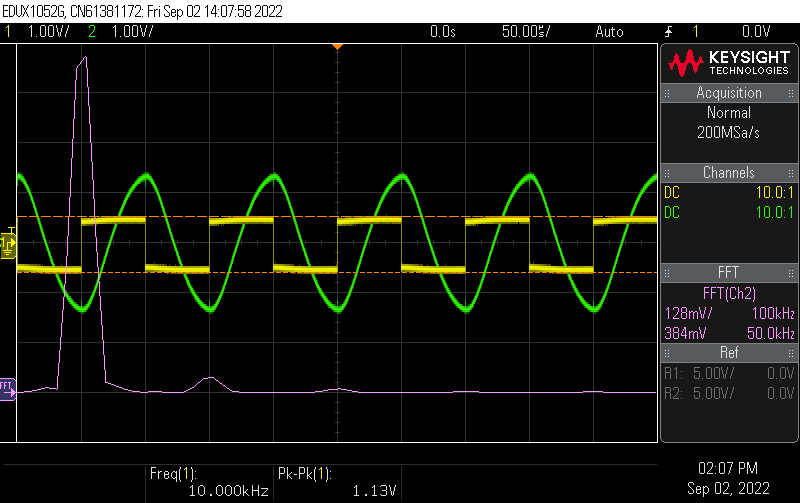


Figure 4: Sine wave (green) generated from a square wave (yellow) filtered by a 2nd order low-pass filter. The purple line is the oscilloscope’s FFT of the sine wave.

As a result, Figure 4 shows the remnant of the square wave’s first odd-order harmonic which is causing the sine wave to be distorted. The hard requirement for the sine wave was to ensure that the total harmonic distortion was less than 1% and should not have crossover distortion. The sine wave shown in Figure 4 does not have crossover distortion. Using the same waveform in Figure 4, the total harmonic distortion was calculated using the following equation.

Though the total harmonic distortion is less than 1%, other options were explored as this would not meet our requirement that students can start the lab immediately upon connecting the board to a computer. Students would have to adjust the frequency of the square wave, adjust the cut-off frequency of the filter, and change capacitors. Although turning two knobs and moving jumper wires is not an unreasonable expectation, setting the cut-off frequency to match the square wave frequency manually cause inconsistencies in the sine waves produced, unnecessarily prolonging the main lab. Designing the function generator for ENEL270 lab would necessitate that any observable error on the data acquisition side would be from the students incorrectly setting up their circuits and not from the function generator.

Option 2

The second option explored all the analogue function generation circuitry in one package. On paper, these chips can produce all the waveforms needed for ENEL270 and has a wide frequency range. They are much simpler in operation compared to option 1, only requiring the user to change the capacitor for decade selection and a potentiometer for fine tuning the specific frequency. The XR2206 was a popular monolithic IC that was used in some function generators in the past. Although these monolithic analogue function generator ICs looked promising, they were either entirely obsolete or sold at unreasonable prices. JayCar sold the XR2206 for $24.90, and major electronics vendors such as DigiKey, Mouser and Element14 did not return any results. Subsequently, this option was quickly abandoned.

Option 3

The third option explored used a dedicated filter. Since the first option used a square wave generator, using an IC to do the filtering to get the sine wave was a feasible consideration. As shown in Figure 4, the higher frequencies can be filtered from the fundamental frequency to obtain the sine wave. Using a dedicated filter can provided higher order filters that have a stepper roll-off. The 2nd order filter shown in Figure 3 attenuates the first odd-order harmonics by 17 dB from passband. One of our colleagues had a MAX7403, an 8th order low-pass elliptic filter.

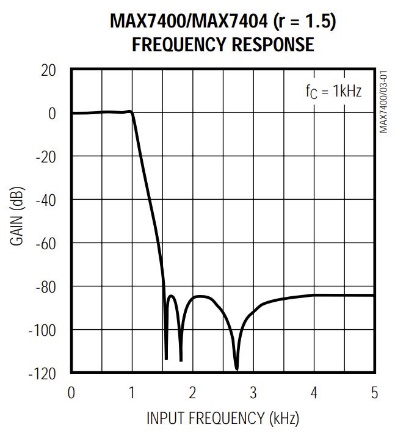


Figure 5: Frequency response of the MAX7403 according to the datasheet. Source: https://datasheets.maximintegrated.com/en/ds/MAX7400-MAX7407.pdf

Figure 5 shows that with a fundamental frequency of 1 kHz, this 8th order elliptic filter can attenuate the first odd-order harmonic (3 kHz) by over 80 dB. The external component count is rather simple, only requiring two external capacitors. However, the elliptic filter uses a switched-capacitor topology. This means that the MAX7403 requires an external clock source to switch the capacitors that form the filter network. For normal operation, the elliptic filter requires a square wave at the desired frequency of the sine wave, and particularly for the MAX7403, a clock frequency 100 times the desired frequency of the sine wave. Meaning that this elliptic filter requires two separate square waves at different frequencies.

Scope shot but then why we didn’t pick it at the end.

Option 4

The fourth option explored function generation using microcontrollers featuring internal DACs or had the capability of using an external DAC. Both options were explored to see what kind of microcontrollers and DACs were available on the market. It was important that for final production, the microcontroller and DAC (with the rest of the board) fit in the budget constraint and was easy to programme so that [the board] can be seamlessly rolled out ready to be used by students.

The first fully digital option involved a microcontroller with an external DAC. This was the initial reaction because the team was very familiar with the well-established Arduino platform. Popular microcontrollers such as the ATMEGA328P, ATMEGA32U4, and ATTINY85 were considered for function generation. The DACs were a harder decision. To the team, there were two common methods of DACs, one was using a resistive string DAC and the second was using a direct digital synthesis IC. The first kind of DAC uses a string of resistors, and the programme calls the DAC to pass the voltage from in-between each resistor in the string. This kind of DAC can output different voltages as the entire resistor string mimics a resistor divider. The MCP4822 is an example of a resistive string DAC that one of our colleagues had available.

Show oscilloscope shots of said wave.

Then microcontrollers with built-in DACs.

The second fully digital option involved microcontrollers with internal DACs. Despite having an abundance more experience on the Arduino platform collectively, one of our colleagues had limited experience with the STM32 line-up of microcontrollers. These microcontrollers had plenty of hardware on tap. With many ADC channels, many timers, DMAs, DACs, and internal crystals, the STM32 on paper was a convincing choice for both function generation and data acquisition.

Final Decision

**OPAMP circuit**

The OPAMP featured on [the board] must be able to complete all the exercises in the ENEL270 labs that included OPAMPs. They were labs one to four. An OPAMP is an extremely versatile device that can be used for many purposes such as an amplifier, filter, comparator, and many more. The OPAMP circuits contained in labs one to four are non-inverting amplifier, non-inverting comparator, Schmitt trigger, summing amplifier, active low-pass filter, active high-pass filter, active band-pass filter, transconductance amplifier, square/triangular wave generator, and peak detector. To extend the versatility of [the board], the OPAMP circuit will also include a voltage follower, inverting amplifier, integrator, differentiator, and differential amplifier. The OPAMP currently used in the ENEL270 lab is the LMC6462.

**Transistor Circuit**

**Sensor Circuit**

PN168

**Data Acquisition**

The data acquisition design rationale came naturally with the selection of the function generator.

**Software**

TO DO

USB to Text

Triangle Wave 100 kHz

Elliptic Filter

Arduino and DAC