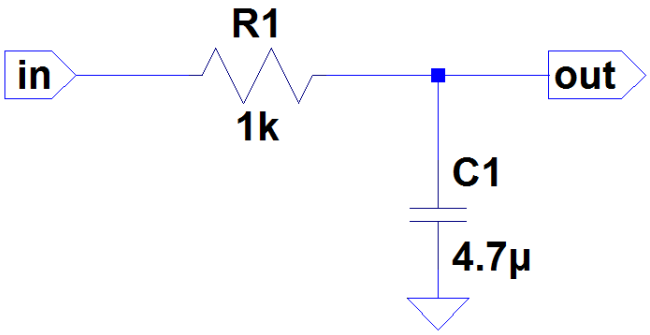
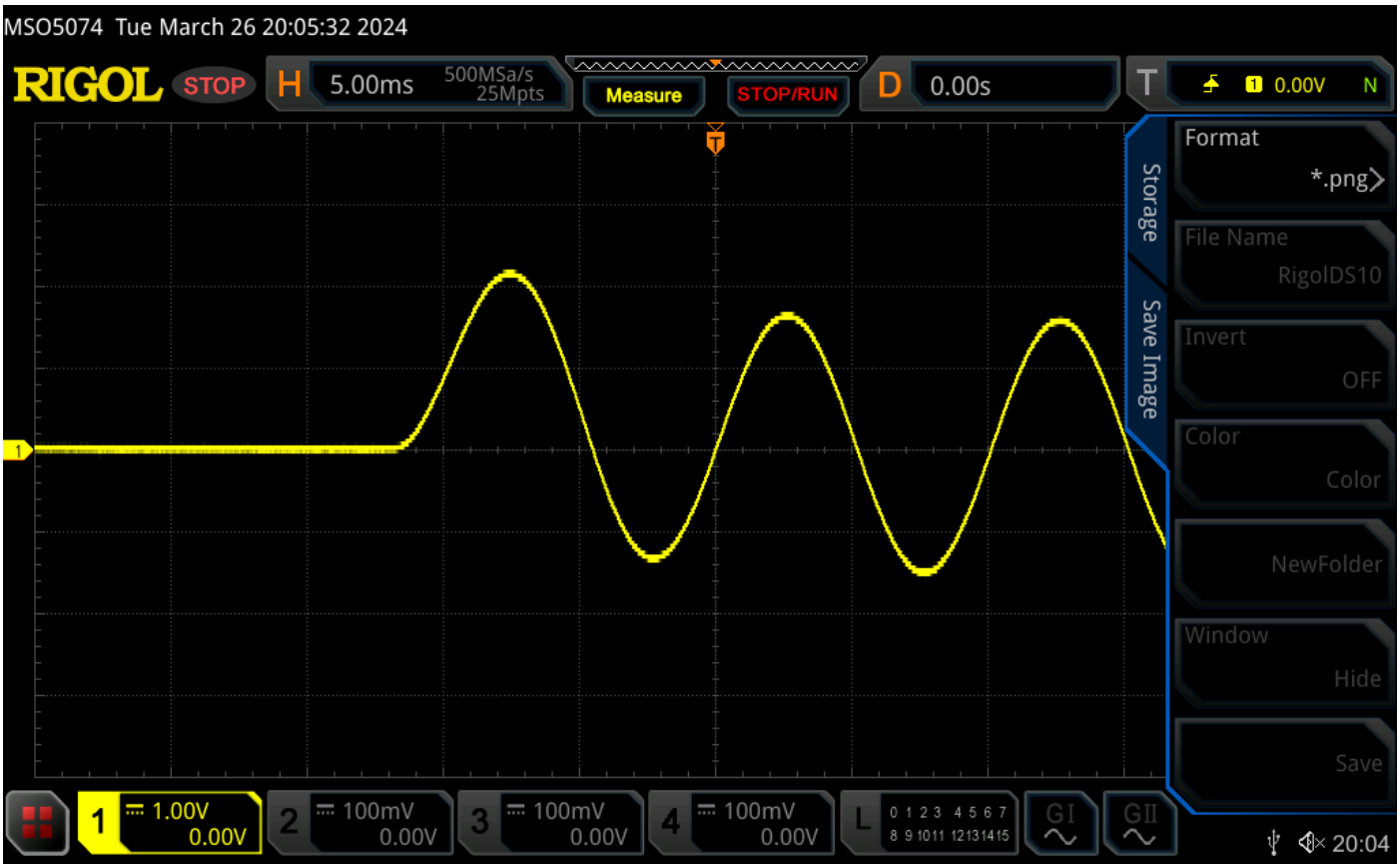
REPORT

Experiment 1: The Transient Response of RC Circuits



3. transient waveform



Question:

What did you find?

The transient response of an RC (Resistor-Capacitor) circuit refers to the behavior of the circuit when it is subjected to a sudden change in input.

In an RC circuit, the transient response is characterized by the following:

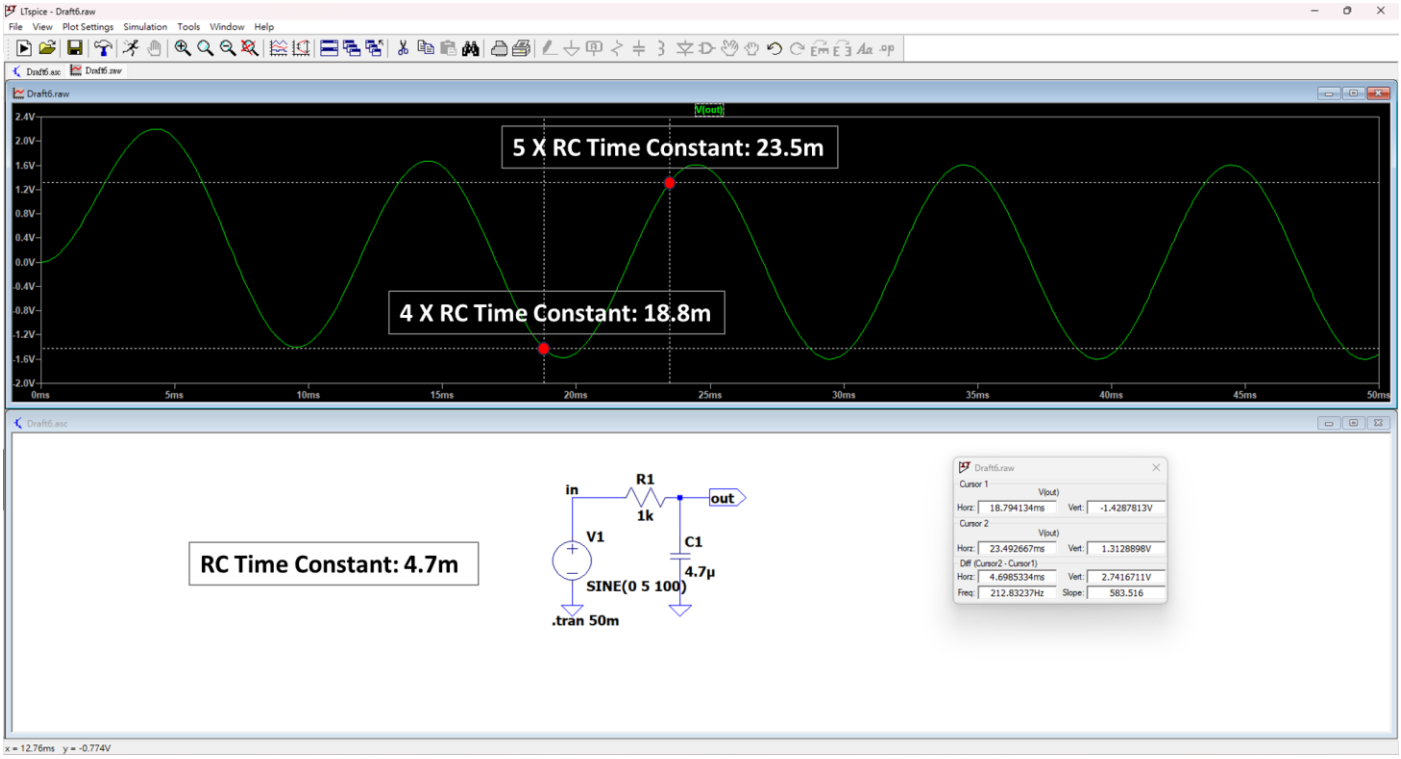
1. Charging/Discharging:
   When an input is applied to an RC circuit, the voltage across the capacitor will not instantly reach the final steady-state value. Instead, the voltage will oscillate for a certain period.

2. Time Constant (τ):
   The time constant, denoted as τ, is the time it takes for the capacitor voltage to reach approximately **63.2% of its final value** during charging or to drop to 36.8% of its initial value during discharging.
   The time constant is calculated as **τ = RC**, where R is the resistance and C is the capacitance of the circuit.

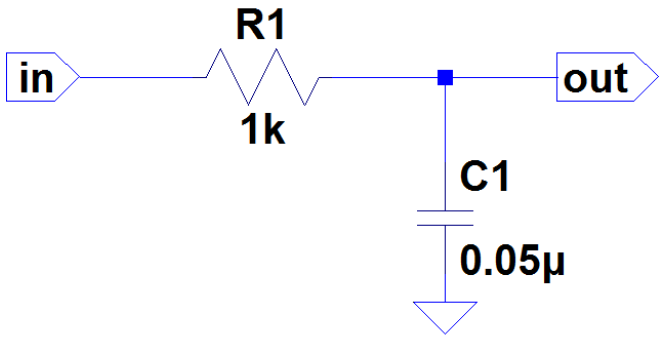
How much time did the system need to achieve stable state?

The time it takes for the capacitor voltage to effectively reach its final steady state value is called the transient time. The **transient time is approximately 4-5 times the time constant**. That means **with τ = 4.7m in this circuit, it takes the system 18.8ms ~ 23.5ms to reach stable state**.

LTspice Simulation:



Experiment 2: First-Order Low-pass filter and High-pass filter



2.

	f1	f2	f3	f4	f5	f6	f6	f8
Target Freq. (Hz)	50	100	500	1.60k	3.18k	5.5k	10.2k	31.7k
Vout,pp (V)	10.321	10.234	10.059	8.922	6.648	4.724	2.974	1.225
Vin,pp (V)	10.216	10.216	10.216	10.130	10.043	9.8705	9.7839	9.6973
gain (V/V)	1.010	1.002	0.985	0.881	0.662	0.479	0.304	0.126
*Δx (time)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
phase out->in (degree)	-0.757	-1.5877	-9.8946	-26.969	-45.629	-57.781	-67.354	-71.222

*optional: if your oscilloscope can measure phase difference directly, then ignore this item.

The **frequency response** of an RC circuit is given by:

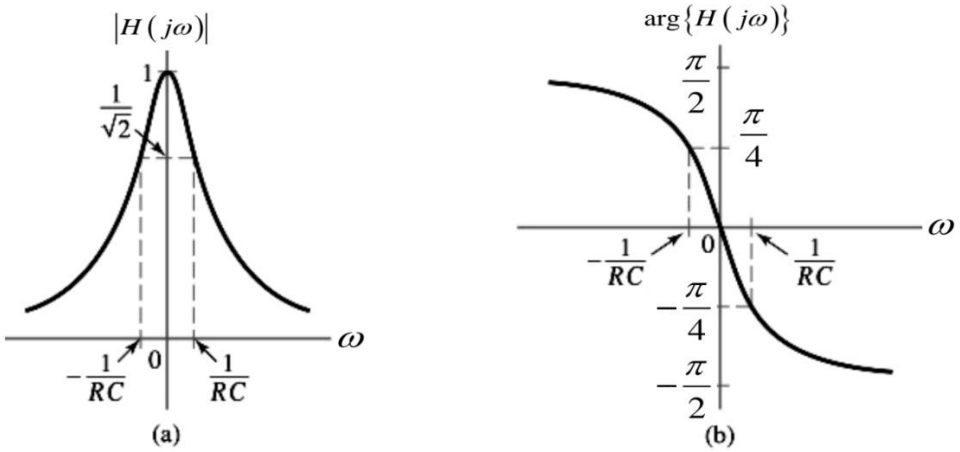
$$H(j\omega) = \frac{1}{RC}\int_{-\infty}^{\infty} e^{\frac{-\tau}{RC}} u(\tau) e^{-j\omega\tau} d\tau = \frac{1}{RC}\int_{0}^{\infty} e^{-(j\omega+\frac{1}{RC})\tau} d\tau = \frac{\frac{1}{RC}}{j\omega + \frac{1}{RC}}$$

Magnitude response:

$$|H(j\omega)| = \frac{\frac{1}{RC}}{\sqrt{\omega^2 + (\frac{1}{RC})^2}}$$

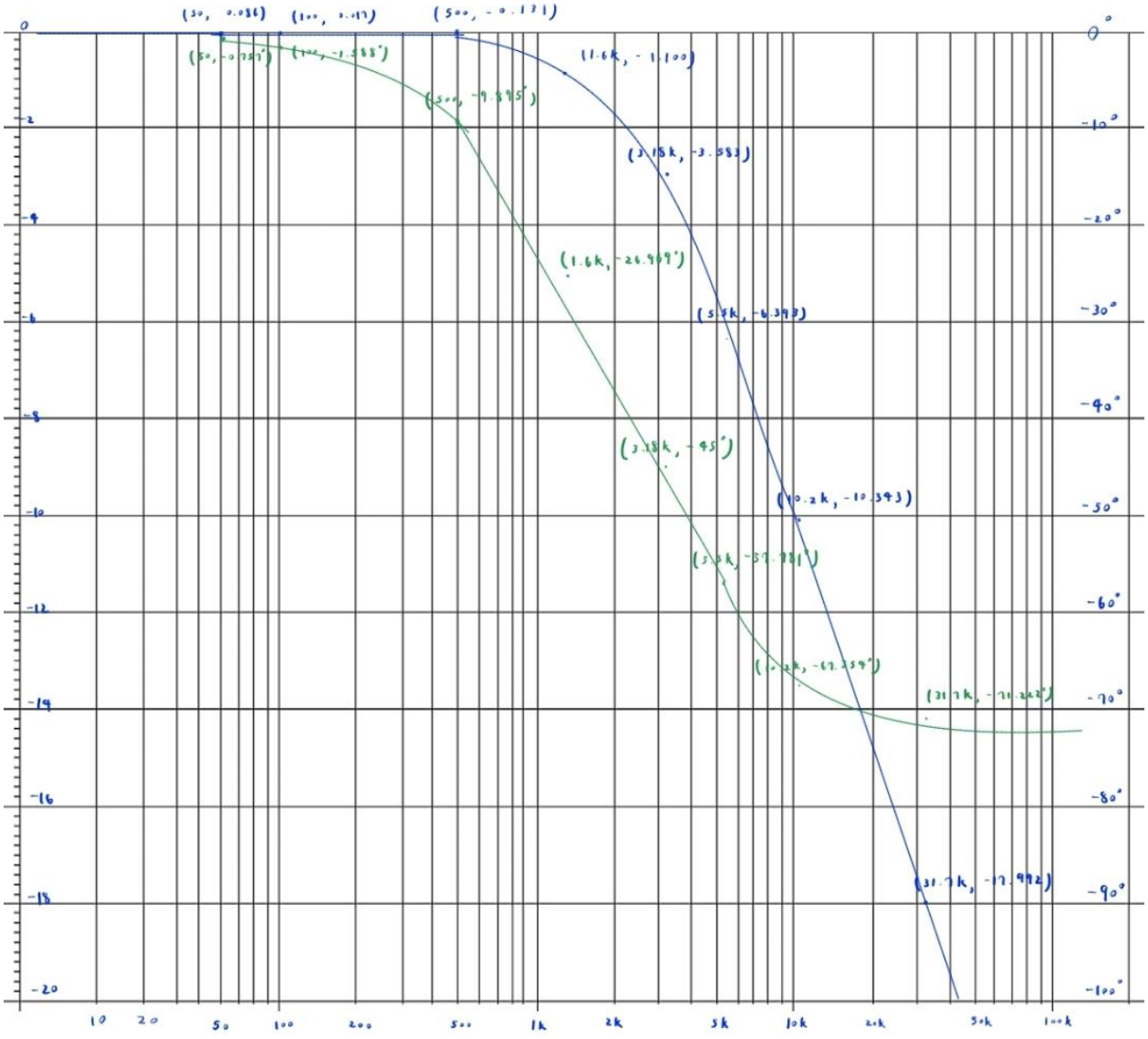
Phase response:

$$\arg\big(H(j\omega)\big) = -\arctan(\omega RC)$$

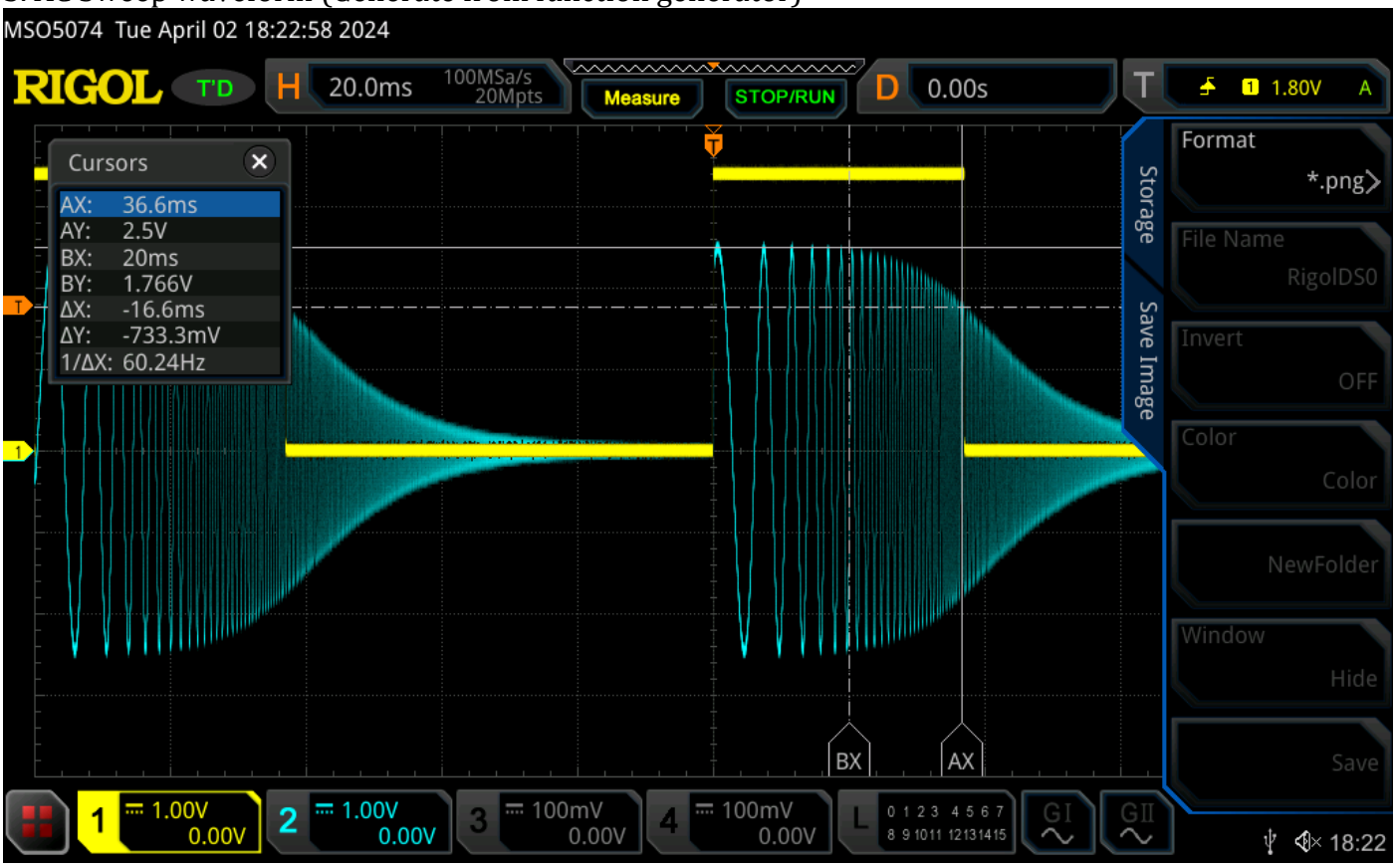


Magnitude & Phase Response Plot of RC Circuit (Source: *Signals and Systems*)

Draw Bode magnitude plot by hand (x-axis=freq., y-axis=20log(gain)dB)
(可以共用橫軸；頻率軸)
Draw Bode phase plot by hand (x-axis=freq., y-axis=phase difference (out to in) degree)
(可以共用橫軸；頻率軸)

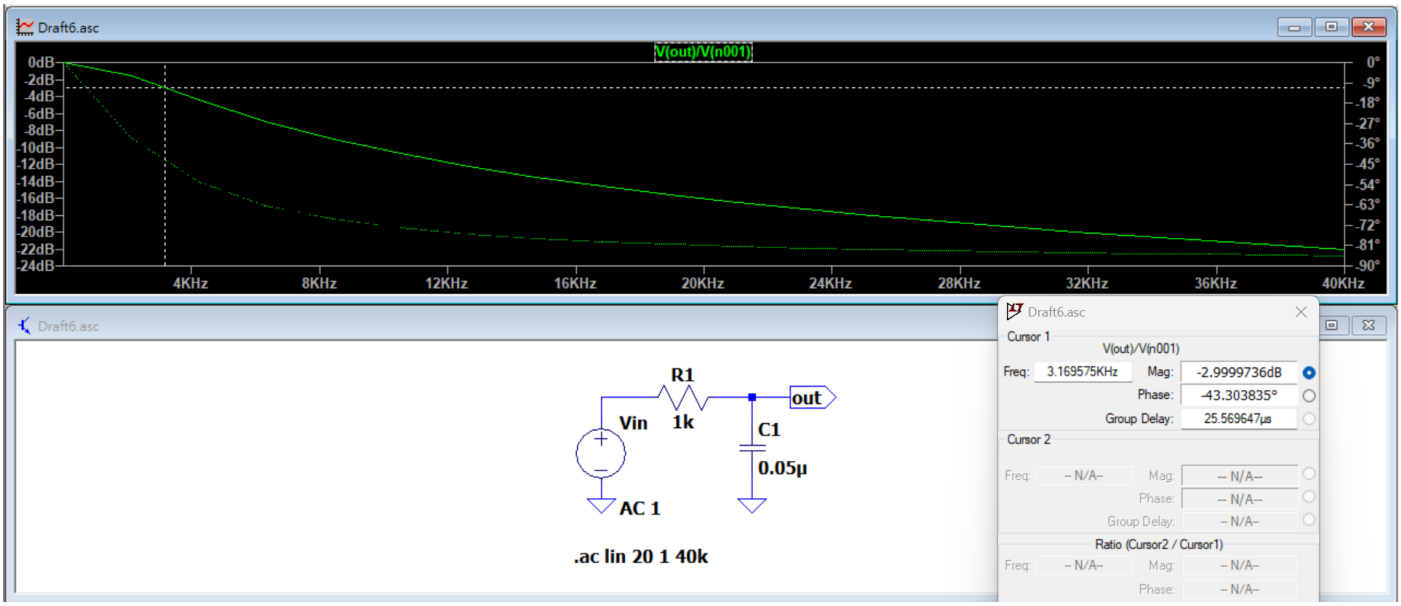


3. AC Sweep waveform (Generate from function generator)



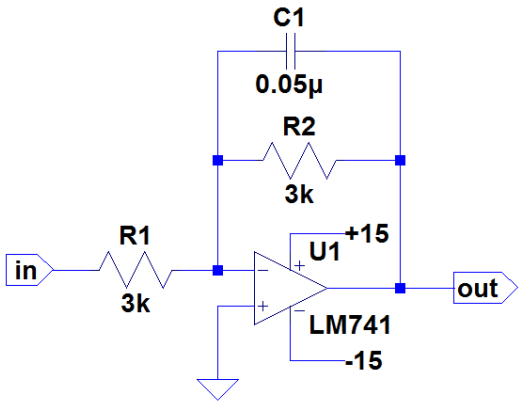
Approximate 3db frequency = 3k Hz from the AC Sweep waveform by applying maker.

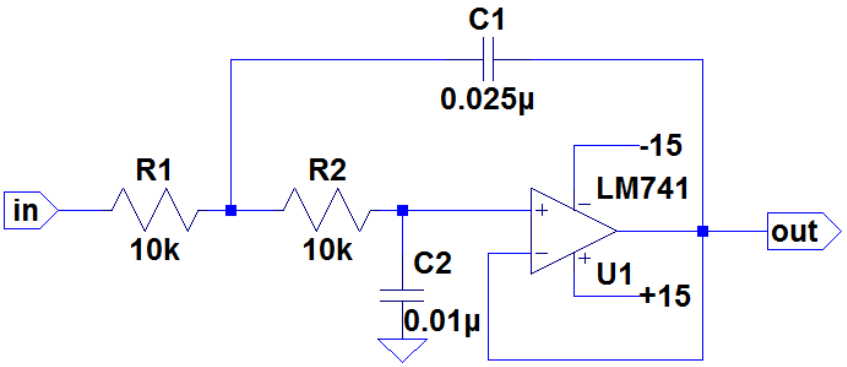
LTspice Simulation:



Simulation using LTspice confirms that the -3dB frequency is approximately 3k Hz indeed, more specifically 3.16k ~ 3.17k Hz.

Experiment 3: Active First-Order and Second-Order Low-pass filter



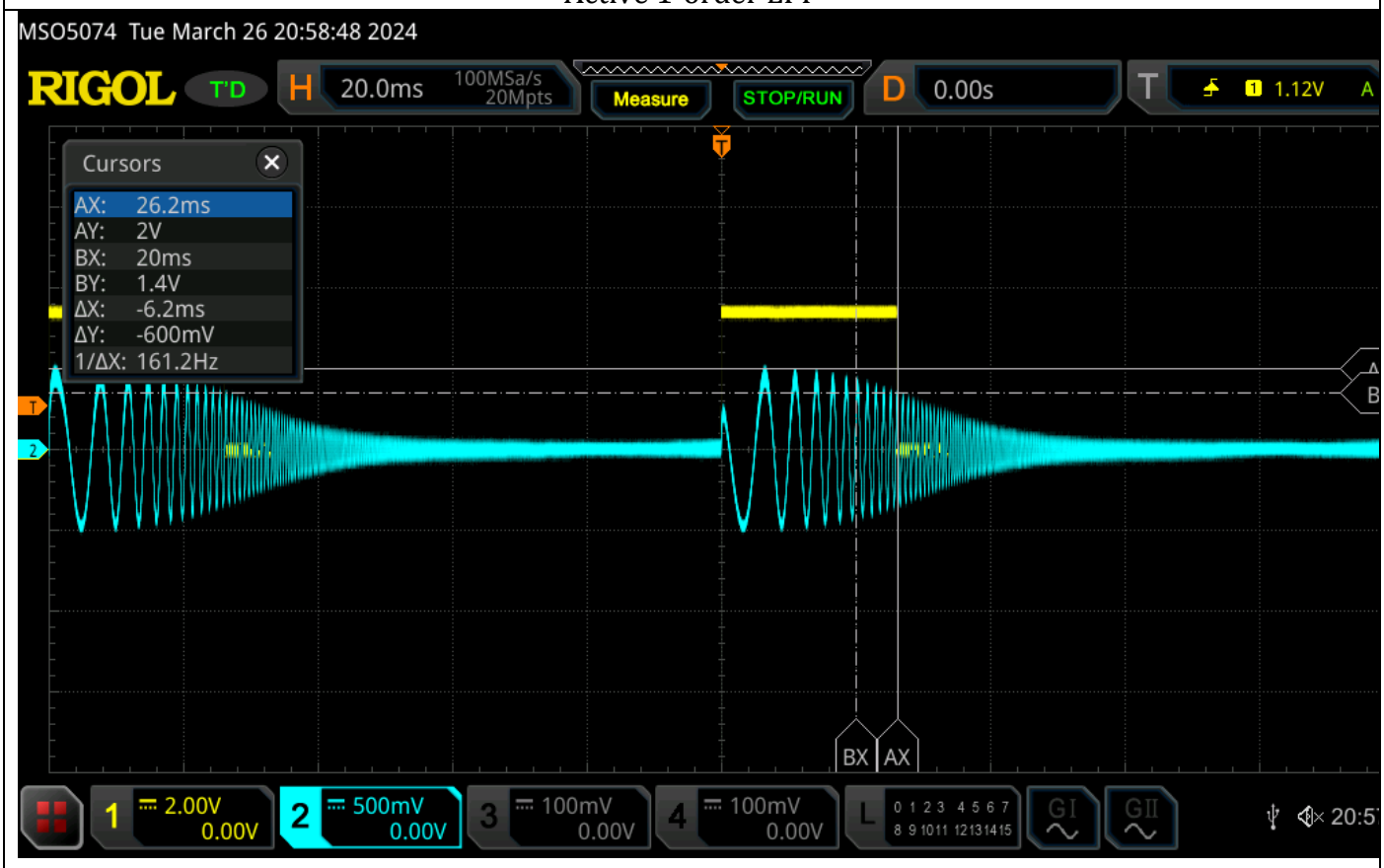


1.

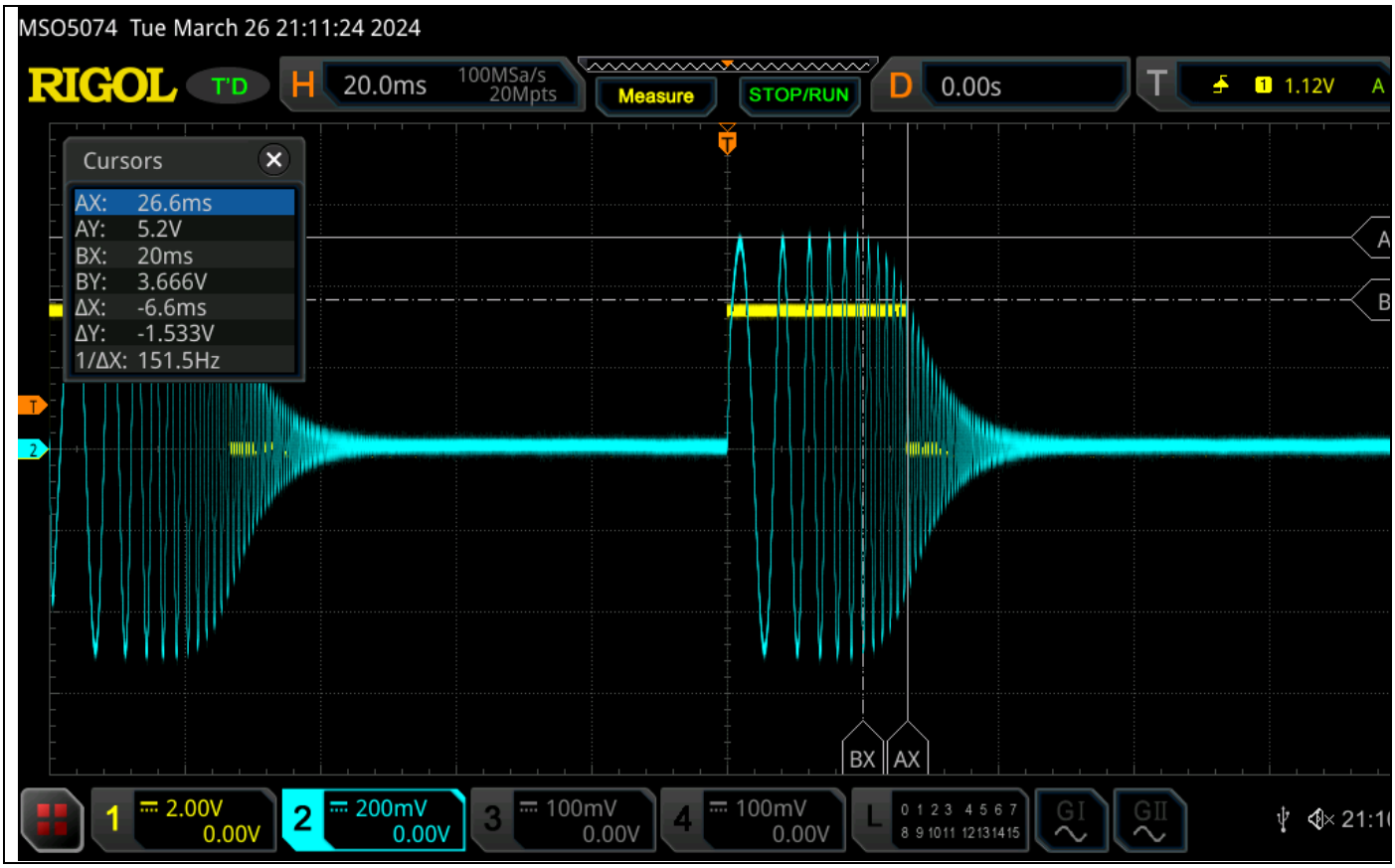
	f_{3dB}(measured) (Hz)
1st order LPF	1.1k
2nd order LPF	1.15k

AC Sweep waveform

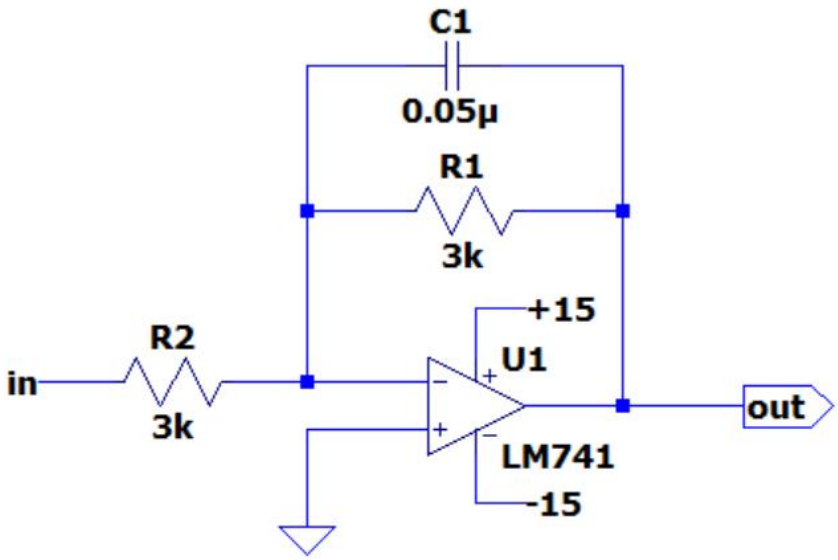
Active 1-order LPF

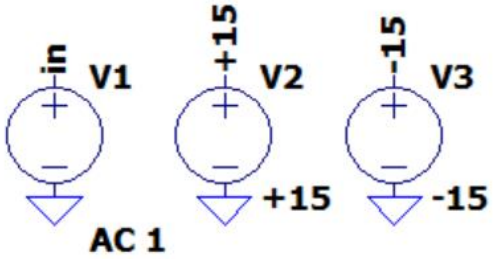


Active 2-order LPF

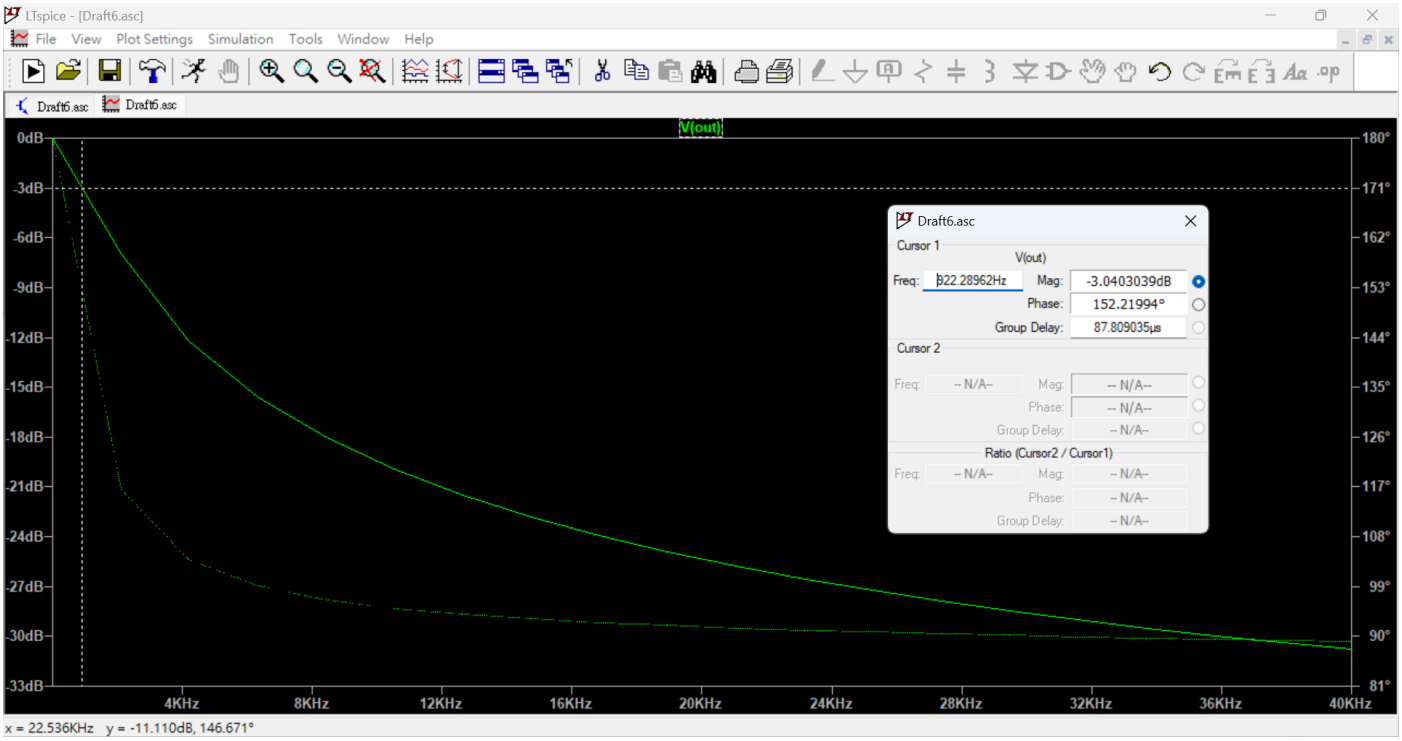


LTspice Simulation (1st order LPF):

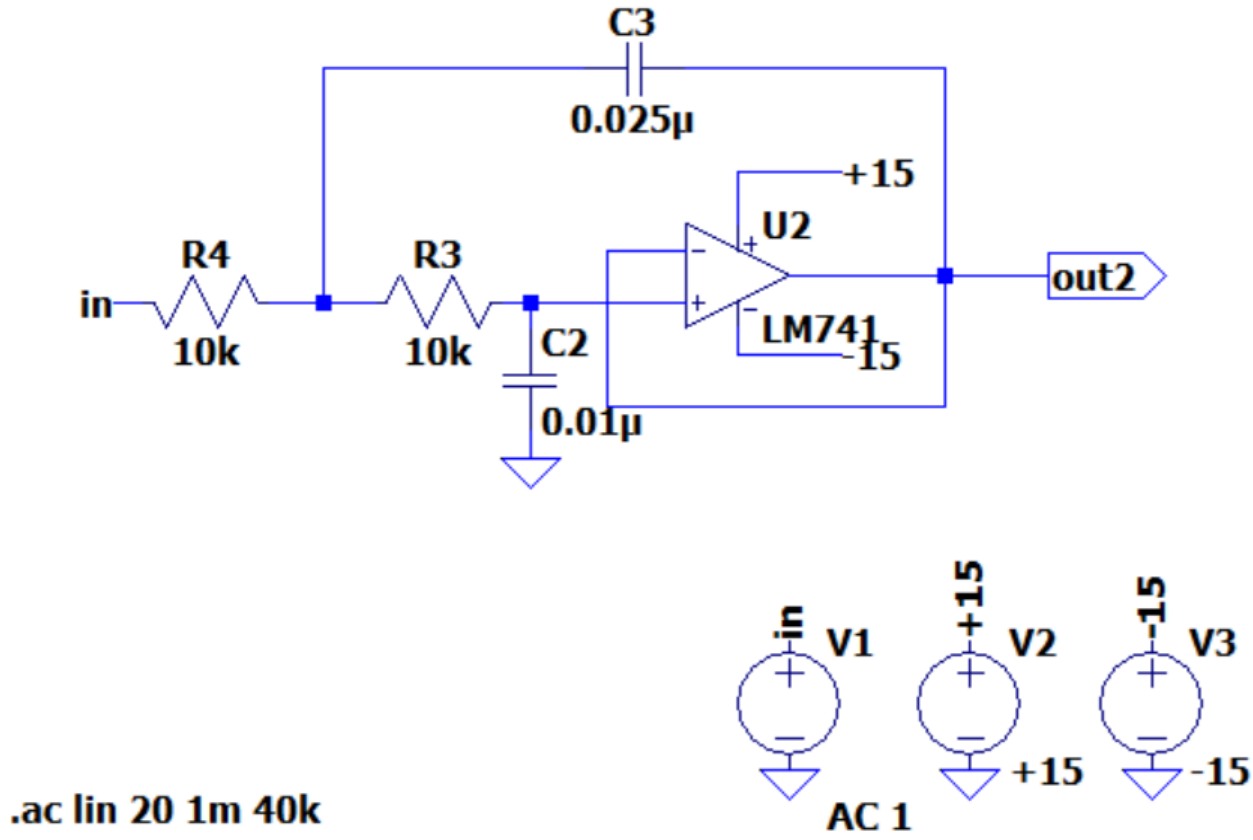


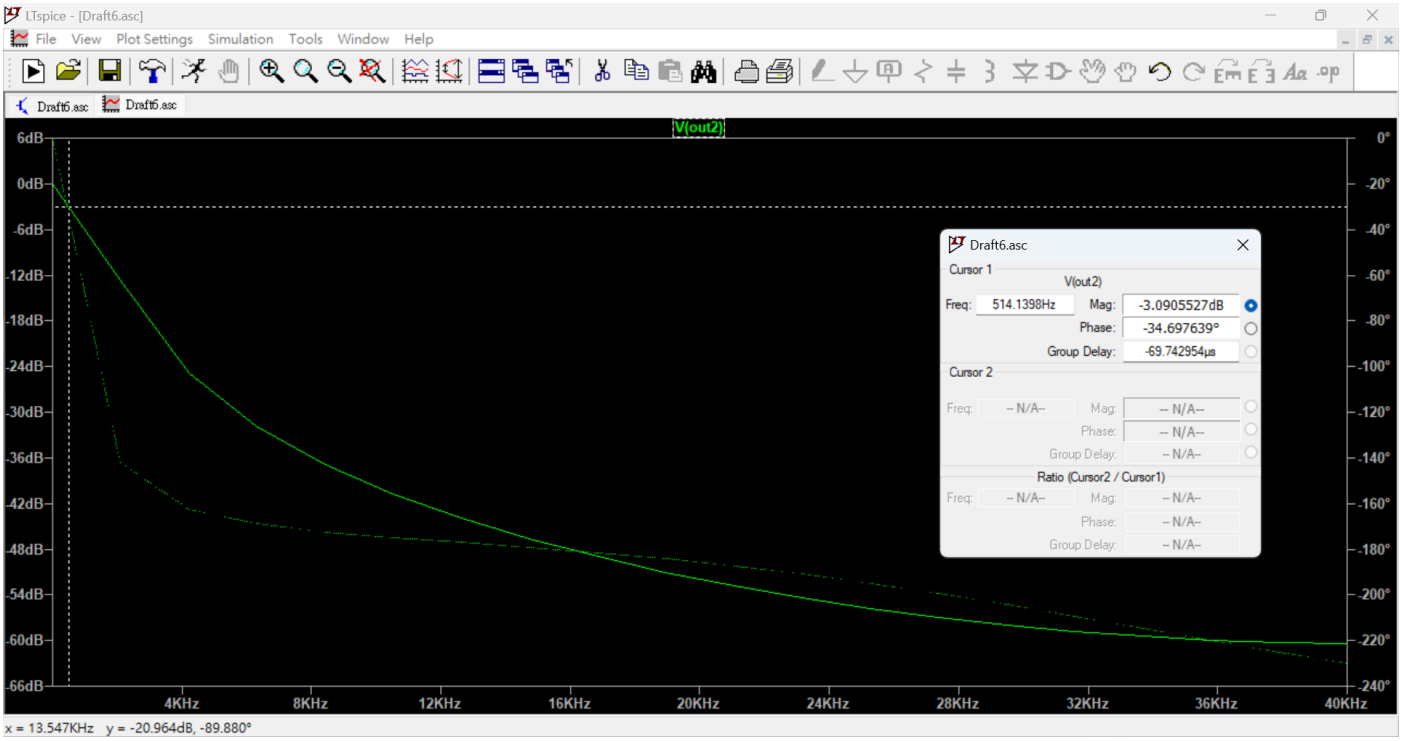


.ac lin 20 1m 1Meg



LTspice Simulation (2nd order LPF):



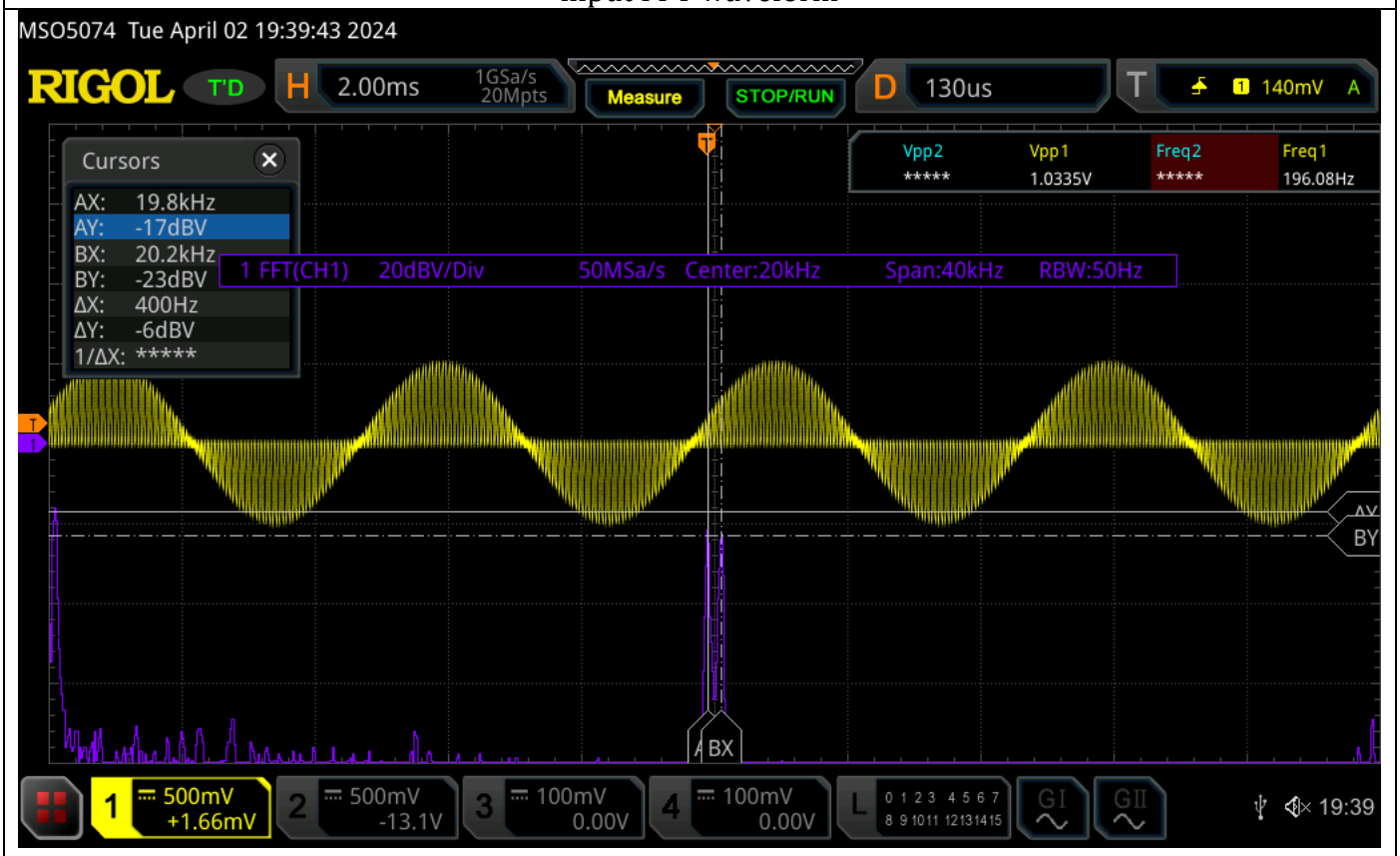


2.

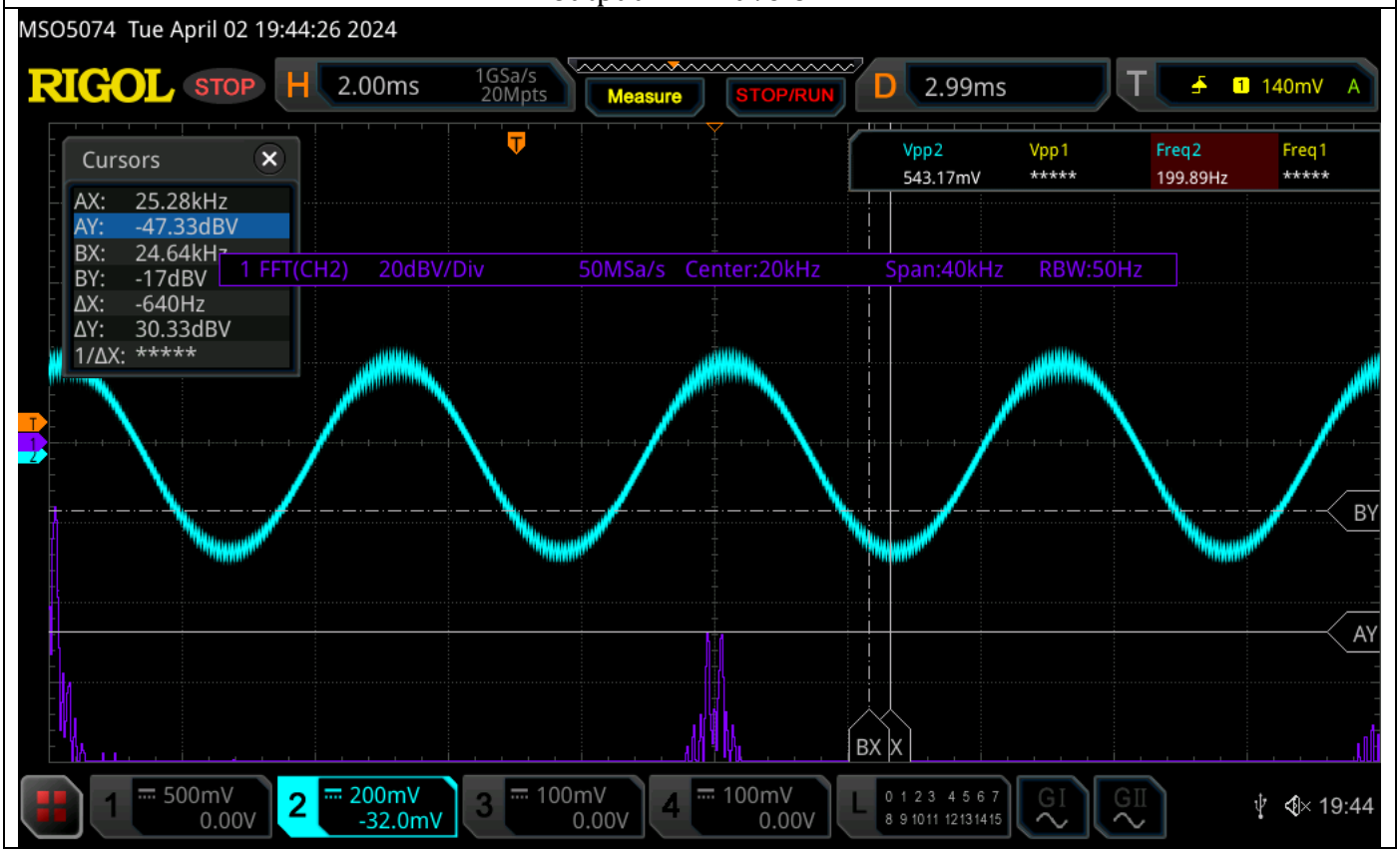
Frequency (Hz)	Carrier	Lower sideband	Upper sideband
	200	19.8k	20.2k
Before filter (dB)	-17	-23	
After 1st order filter (dB)	-17	-47.33	
After 2nd order filter (dB)	-17	-72	

Active 1-order LPF
input and output waveform (time domain)

input FFT waveform

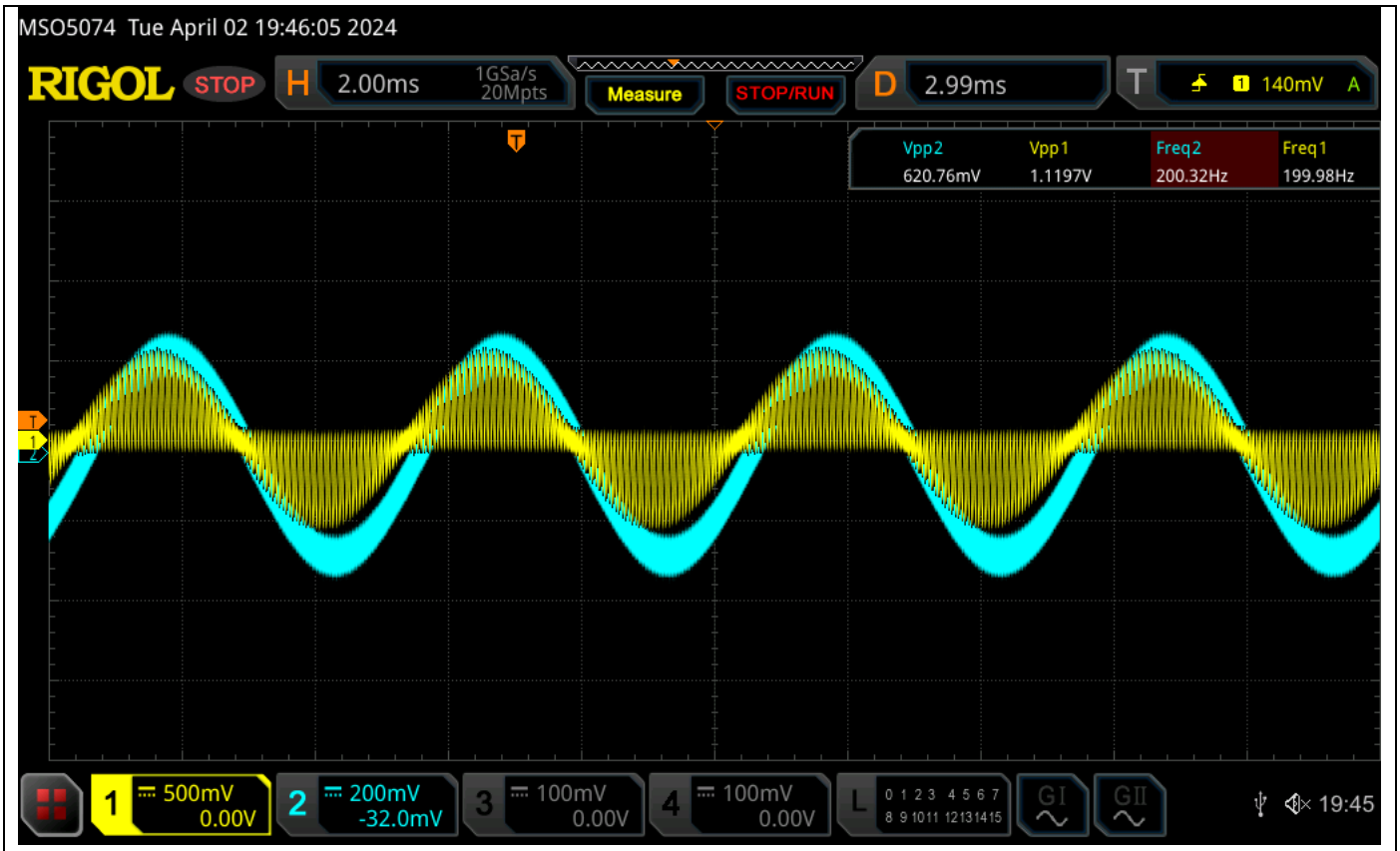


output FFT waveform

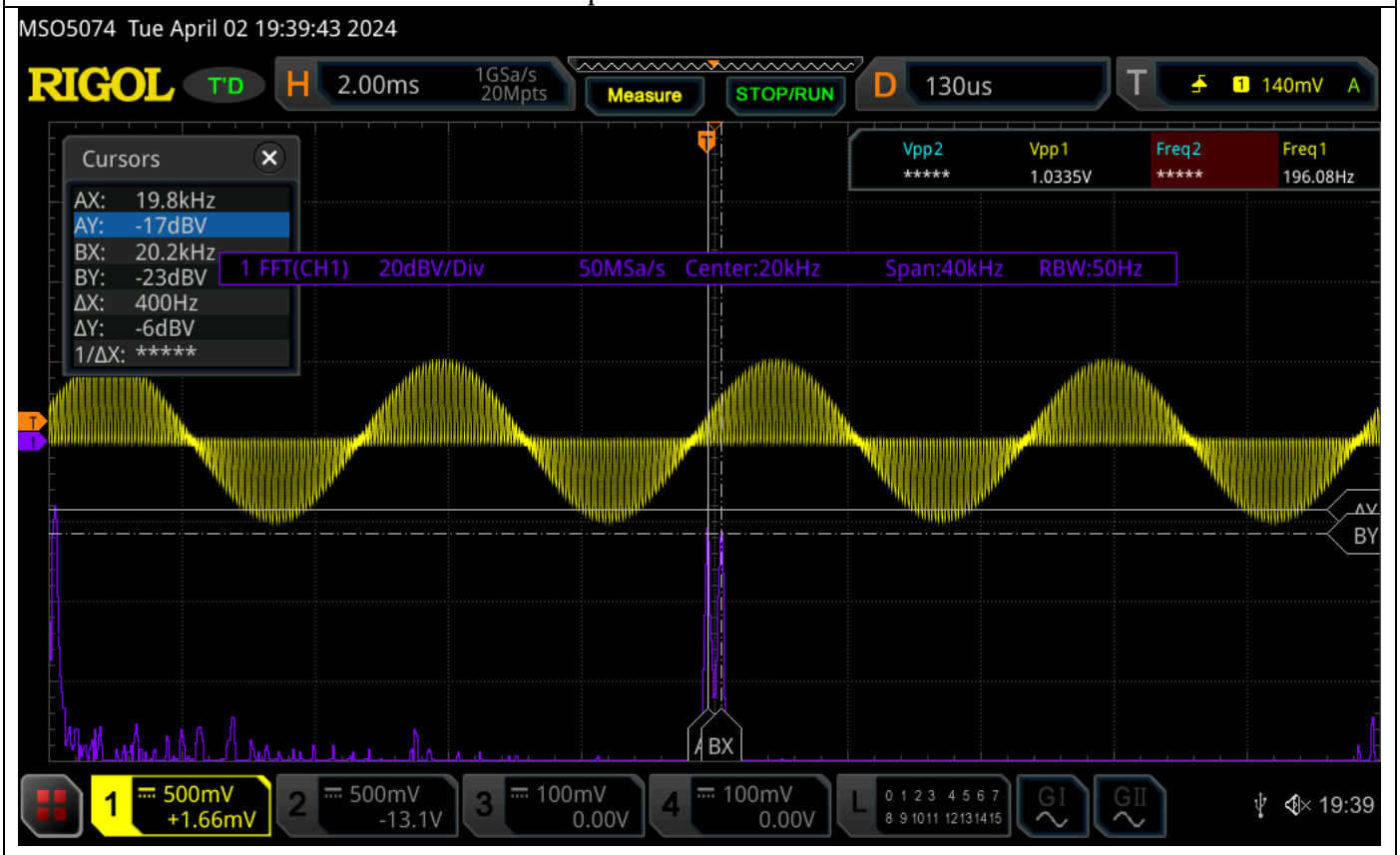


Active 2-order LPF

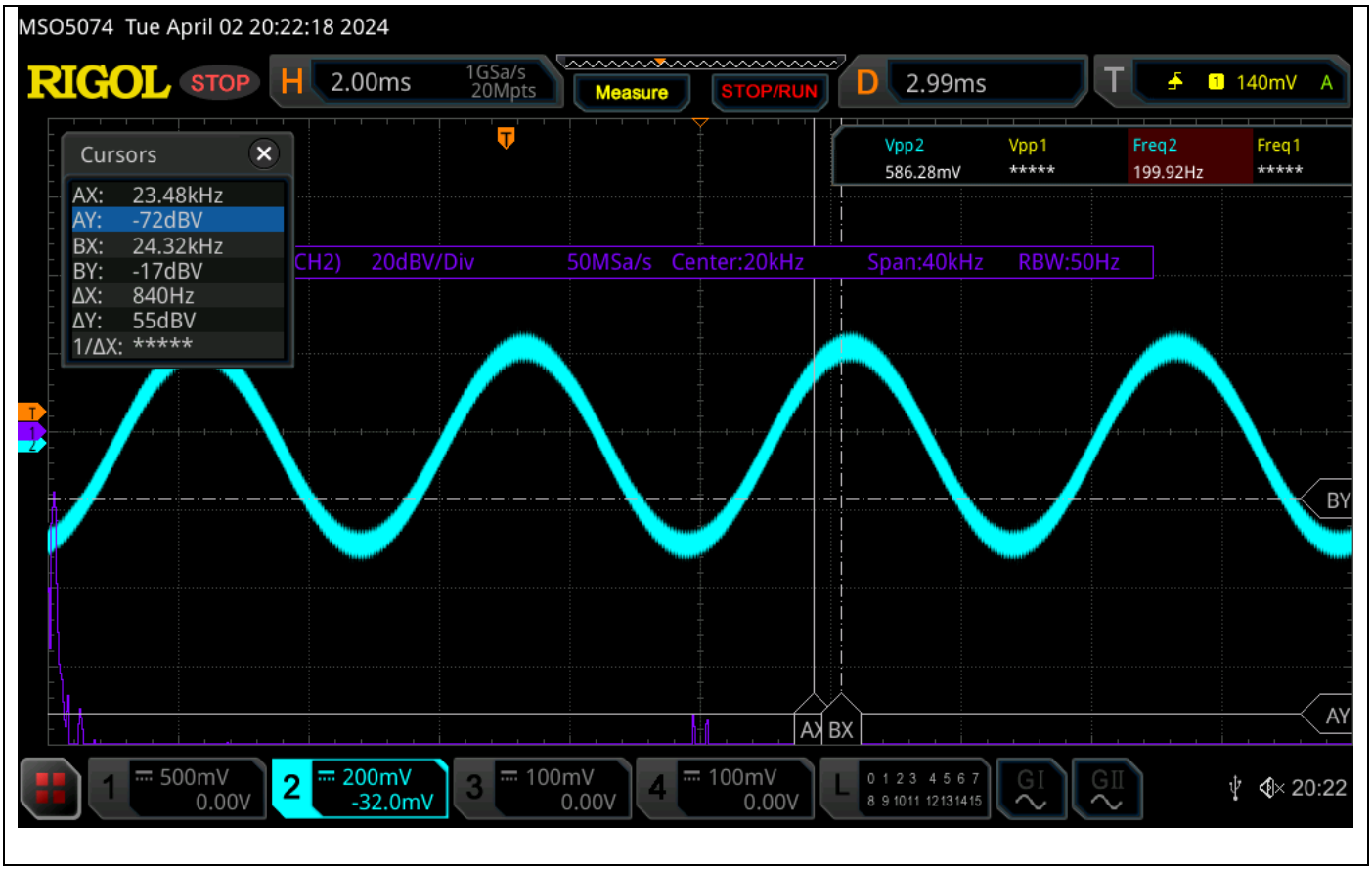
input and output waveform (time domain)



input FFT waveform



output FFT waveform



What is the difference between a first-order filter and a second-order filter?

The key difference between a first-order filter and a second-order filter lies in **the number of poles (or roots)** in their transfer functions.

First-Order Filter:

- ✧ A first-order filter has a **single pole** in its transfer function.
- ✧ The transfer function of a first-order filter has the form:

$$H(j\omega) = \frac{1}{1 + \frac{j\omega}{\omega_0}}$$

where ω_0 is the cutoff frequency of the filter.

- ✧ First-order filters exhibit a single-pole **roll-off of 20 dB/decade** in the frequency domain.
- ✧ They have a simple exponential response in the time domain, with a **single time constant**.

Second-Order Filter:

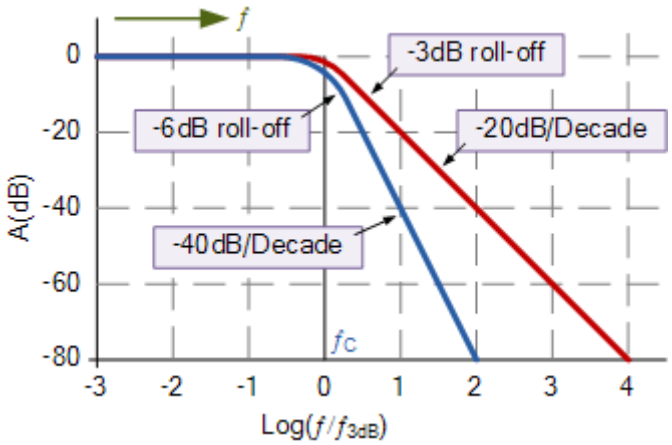
- ✧ A second-order filter has **two poles** in its transfer function.
- ✧ The transfer function of a second-order filter has the form:

$$H(j\omega) = \frac{{\omega_0}^2}{j\omega^2 + 2\zeta\omega_0 j\omega + {\omega_0}^2}$$

where ω_0 is the natural frequency, and ζ is the damping ratio.

- ✧ Second-order filters exhibit a **double-pole roll-off of 40 dB/decade** in the frequency domain.
- ✧ They have a more complex response in the time domain, with the potential for underdamped, critically damped, or overdamped behavior, depending on the value of the damping ratio.

The **main advantages** of second-order filters over first-order filters are their ability to achieve **sharper cutoff characteristics and more flexible frequency response** shaping.



Frequency Response Comparison Between 1st Order LPF & 2nd Order LPF
(Source: https://www.electronics-tutorials.ws/filter/second-order-filters.html)

What is the FFT?

The Fourier Transform is a mathematical technique that allows you to **decompose a complex waveform or signal into its constituent frequencies.** The Fast Fourier Transform (FFT) is an optimized algorithm that can compute the Discrete Fourier Transform much more efficiently than the straightforward DFT computation, especially for large datasets.

Some key properties of the FFT algorithm include:

- ✧ Significantly reduced computational complexity compared to the DFT.
- ✧ Ability to compute the DFT of N points using only **O(*N log N*)** operations.
  (Otherwise directly computing DFT requires $O(N^2)$)

- ✧ Supports both real-valued and complex-valued input data.

What is the purpose of using different window functions?

The purpose of window functions is to **mitigate the spectral leakage** that can occur when analyzing finite-length segments of a signal using the Fourier transform. **Spectral leakage happens when the finite-length signal being analyzed does not contain an integer number of periods of the underlying waveform.** This can lead to artifacts and distortions in the frequency domain representation of the signal.
Window functions help address this issue by:

1. **Tapering the signal** at the beginning and end of the analysis window:
   This reduces the discontinuities at the window edges, which helps minimize spectral leakage.

2. **Emphasizing the central portion** of the analysis window:
   This ensures the most representative part of the signal is weighted more heavily in the Fourier analysis.

Reference:

1. Signals and Systems 2nd Edition by Simon Haykin & Barry Van Veen
2. Tektronix: https://www.tek.com/en/video/what-is-fast-fourier-transform
3. Fundamentals of Electric Circuits by Charles K. Alexander & Matthew Sadiku
4. ElectronicsTutorial: https://www.electronics-tutorials.ws/filter/second-order-filters.html