ECE137B LAB 3 Report

Anthony Guzman

Objective

The objective of this project is to develop a LCR measurement circuit.

It contains an oscillator capable of producing 4 frequencies at 100Hz, 1kHz, 10kHz, and 100kHz. The oscillator outputs 1 sine wave and 2 square waves. One square wave is in phase with the sine wave and the other has a 90 degree phase shift. The Oscillator requires a digital switch to control the frequency.

The LCR measurement circuit should be capable of measuring 1Ohm ~ 10MOhm, 1.76pF ~ 1.59mF, and 2.59uH ~ 1432H, as well as compound LCR circuits within range. The LCR circuit has a range switch, and we would use a mechanical switch in this case, as this is what it would be in a real experiment.

Oscillator - Design

Design Procedure

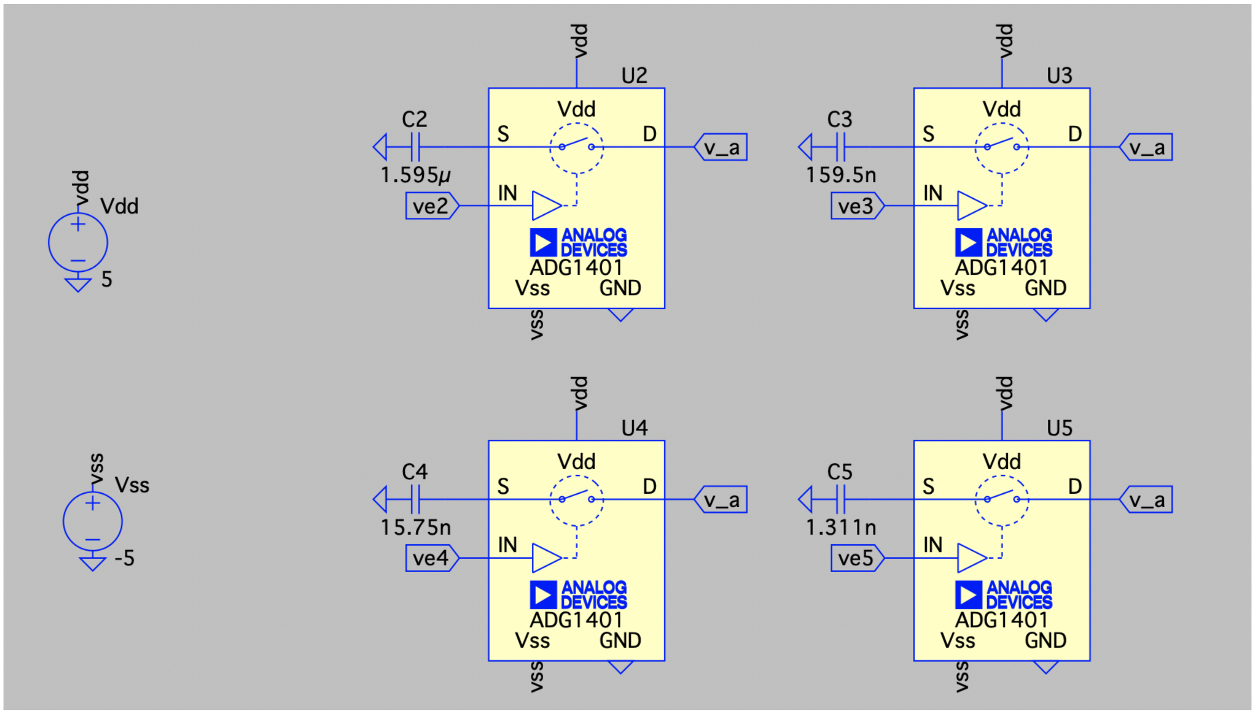
Our Oscillator design contains 4 op-amps connected in a loop, 3 of them are unity gain op-amps and the last one is inverting op-amp with a gain of 50. There is a RC filter between adjacent op-amps, so that the phase shift of the 4 RC circuits adds up to 180 degree, complementing the 180 degree phase shift of the inverting op-amp. There are also 2 comparators generating square waves.

We use R=1kOhm for simplicity, and the value of C depends on the frequency, and thus I use a digital switch. The switch has some parasitic resistance and capacitance, so I need to adjust the values of the capacitances to take those into consideration.

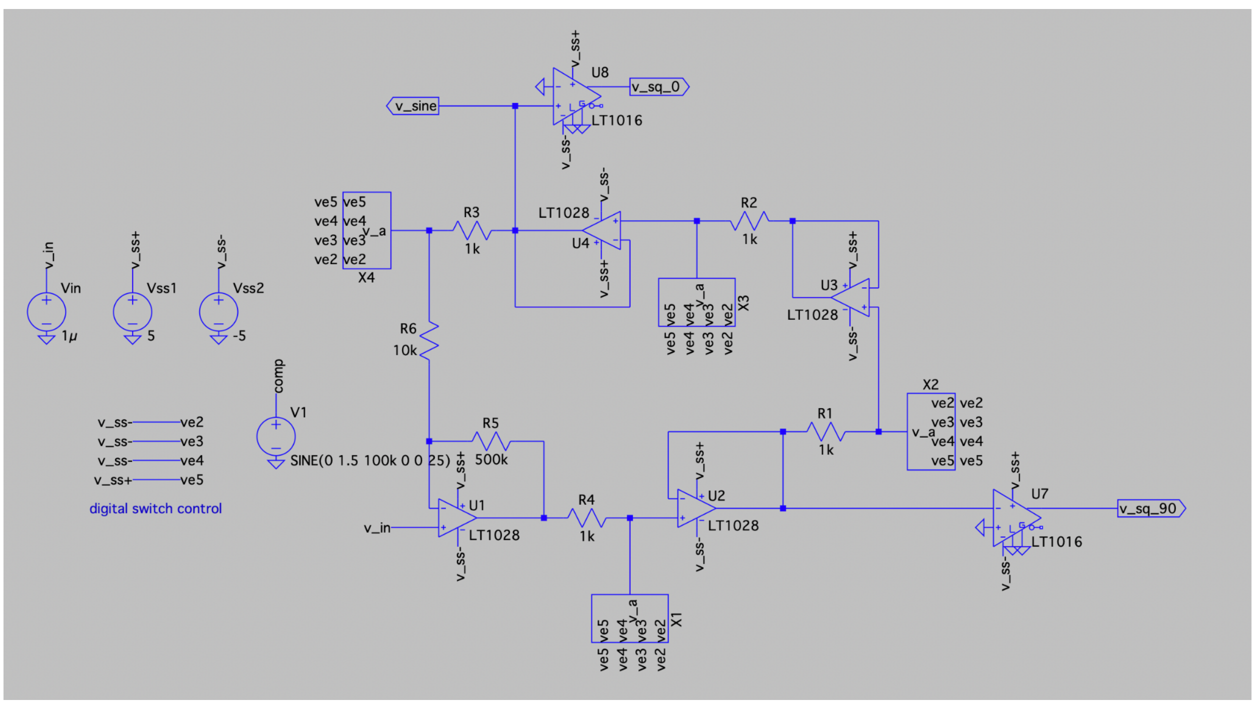
To make it oscillate in simulation, I provided it with a 1uV offset at the positive terminal of the first opamp. It hardly affects anything but starts the oscillator.

We chose the LT1016 as the comparator because it does not have any overshoot and is very fast. The slope is barely noticeable in 100kHz waveform.

Circuit Diagram



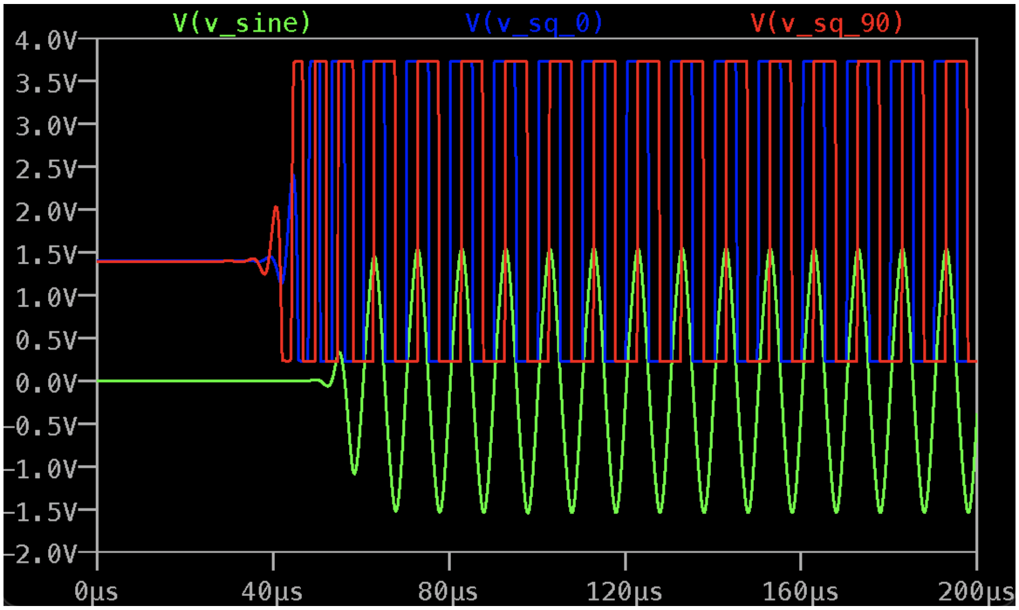
Capacitor Switches (for X1~X4 in the oscillator)

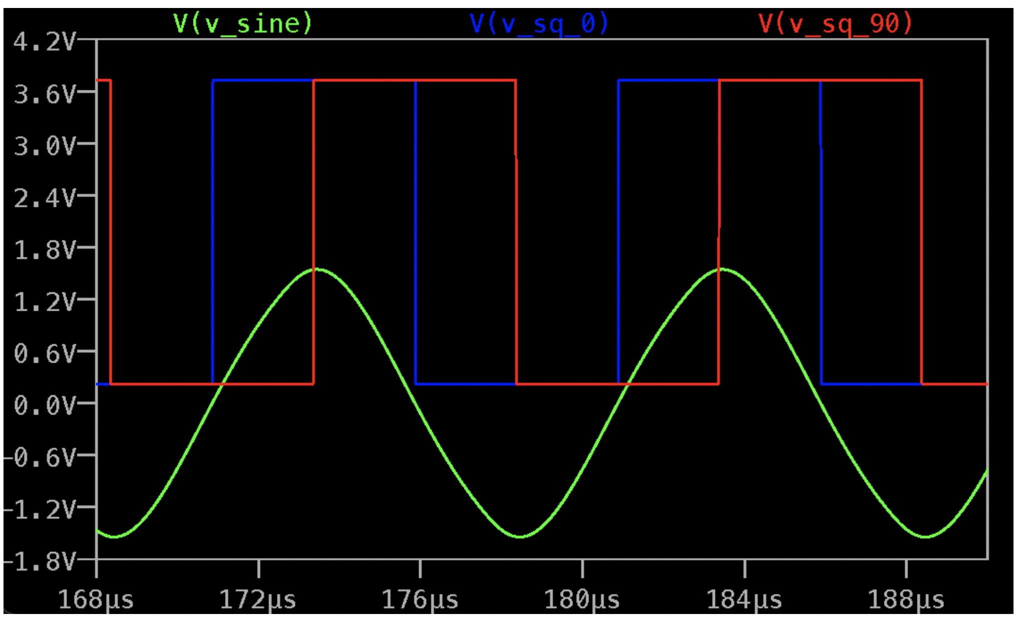


Oscillator Circuit

Oscillator - Simulation

At 100kHz

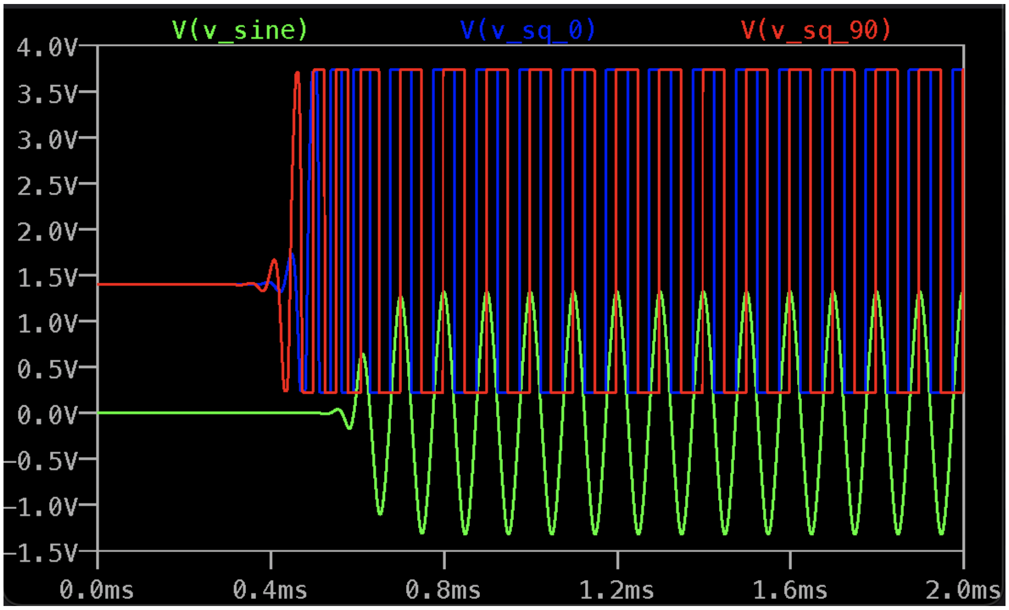


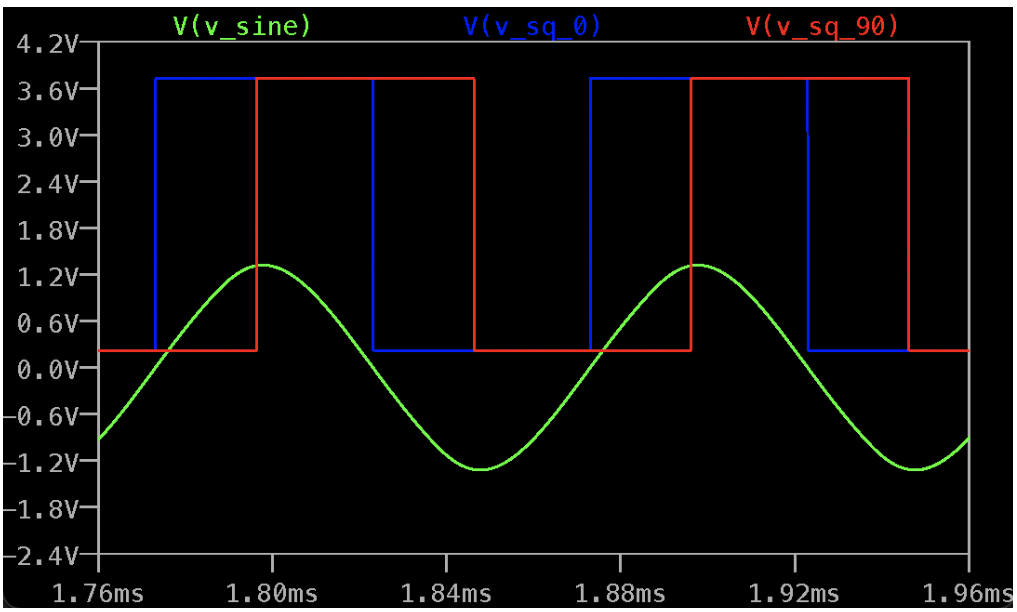


C = 1.311nF, setup time = 60us.

The period is 10.01us, 0.1% longer than desired, but more precise control requires finer value of capacitance, which is impractical in a real lab. Thus, I will deal with this with softwave.

At 10kHz

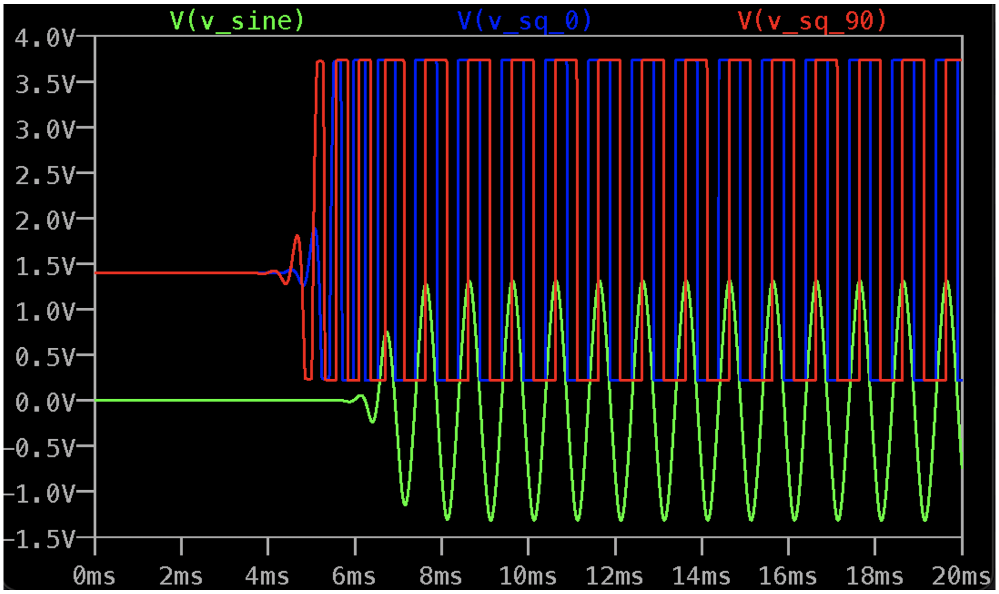


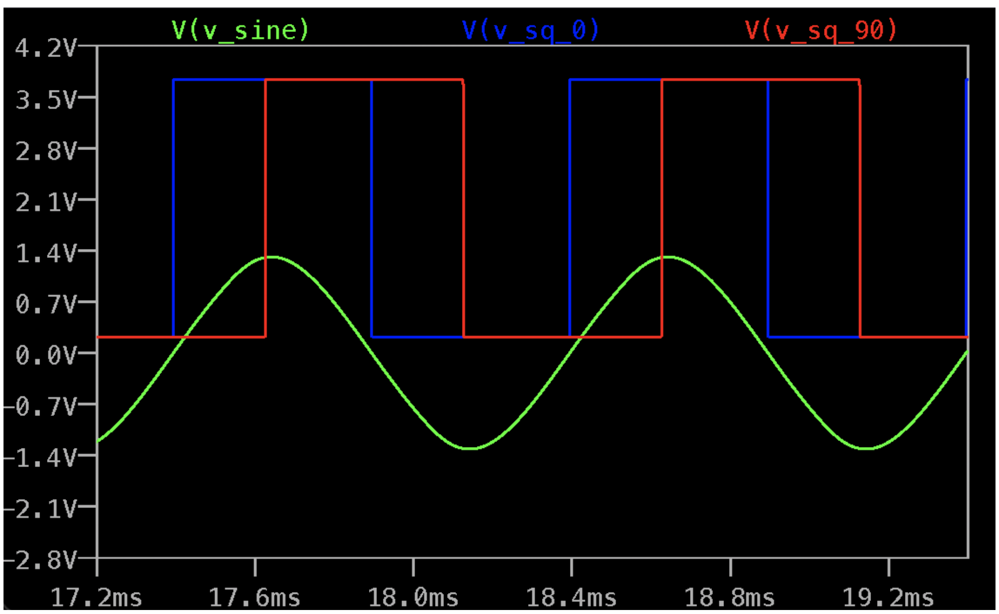


C = 15.75nF, setup time = 0.8ms

The period is 100.02us, 0.02% longer than desired

At 1kHz

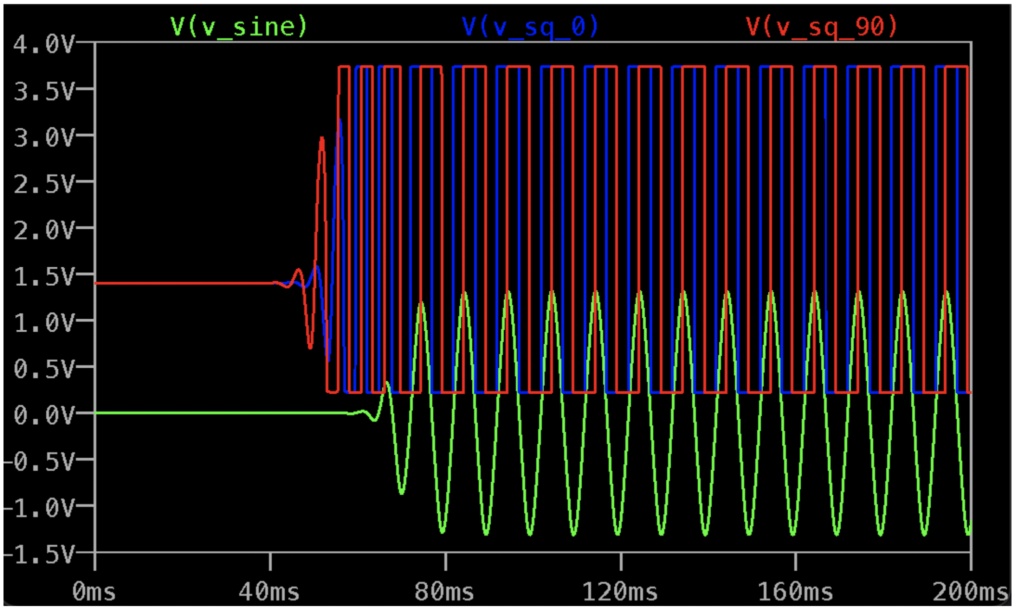


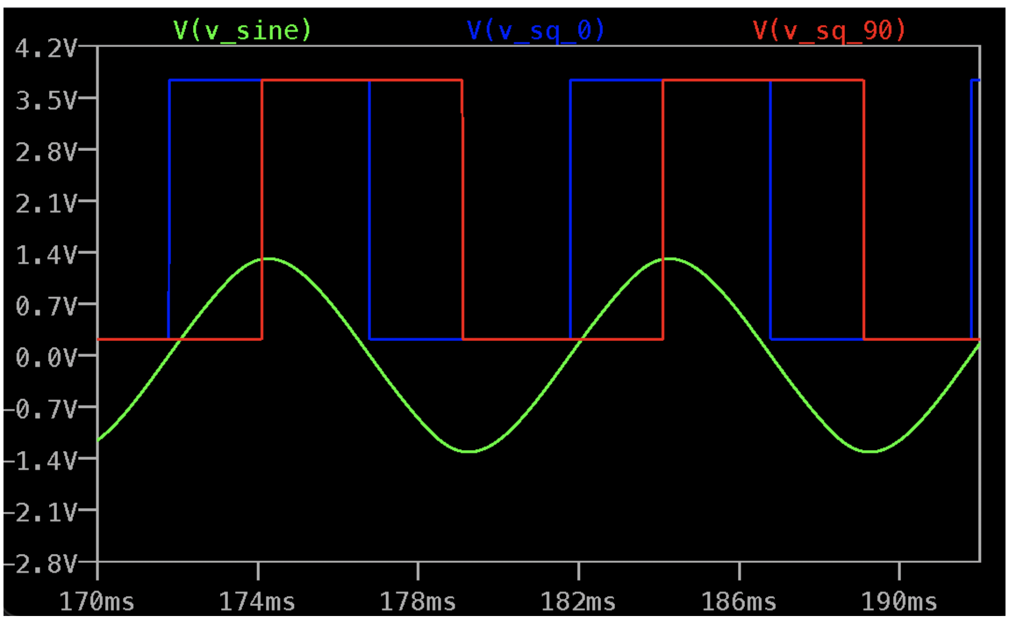


C = 159.5nF, setup time = 8ms

The period is 1.001ms, 0.1% longer than desired.

At 100Hz





C = 1.595uF, setup time = 80ms

The period is 9.995ms, 0.05% less than desired.

LCR Measurement Circuit - Design

Design Procedure

Our LCR Measurement design contains the oscillator that we built, 2 op amps, and a physical switch control for three different settings. The op amp we chose, LT1122, is a high speed high input resistance op amp. It has an input capacitance of Cin=4pF. Thus, we need to use a feedback capacitor to eliminate the pole it might cause. The datasheet suggests RfCf = (Rz+Zt)Cin, but to be safe, I would choose RfCf = 5us so that it would work with any Zt.

The effect of feedback capacitance and input capacitance will be analyzed later and addressed in code. The 3 settings are for Zt < 1kOhm, 1kOhm < Zt < 100kOhm, and 100kOhm < Zt. We choose Rf = Rz so that the gain will not vary too much.

One problem with this LCR design is that if Zt is an inductor, the transfer function of this opamp will have rising gain at certain frequencies. It means that the harmonic distortion of the oscillator, which is not visible through the bare eye, will be magnified and cause a significant impact on the output waveform and thus affect the accuracy of this LCR measurement.

Thus, the software part has 2 methods of analysis, and one of them can address this issue.

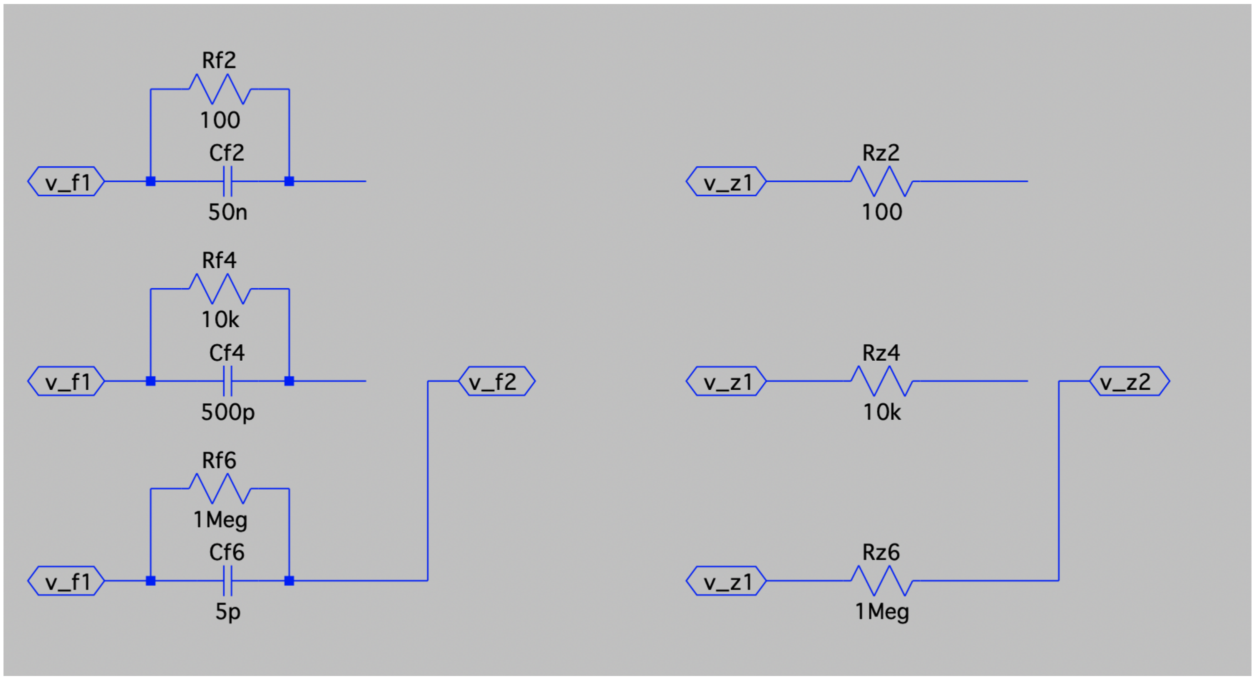
Analysis

Since we know Vx, Vy, and Vout, we can derive that

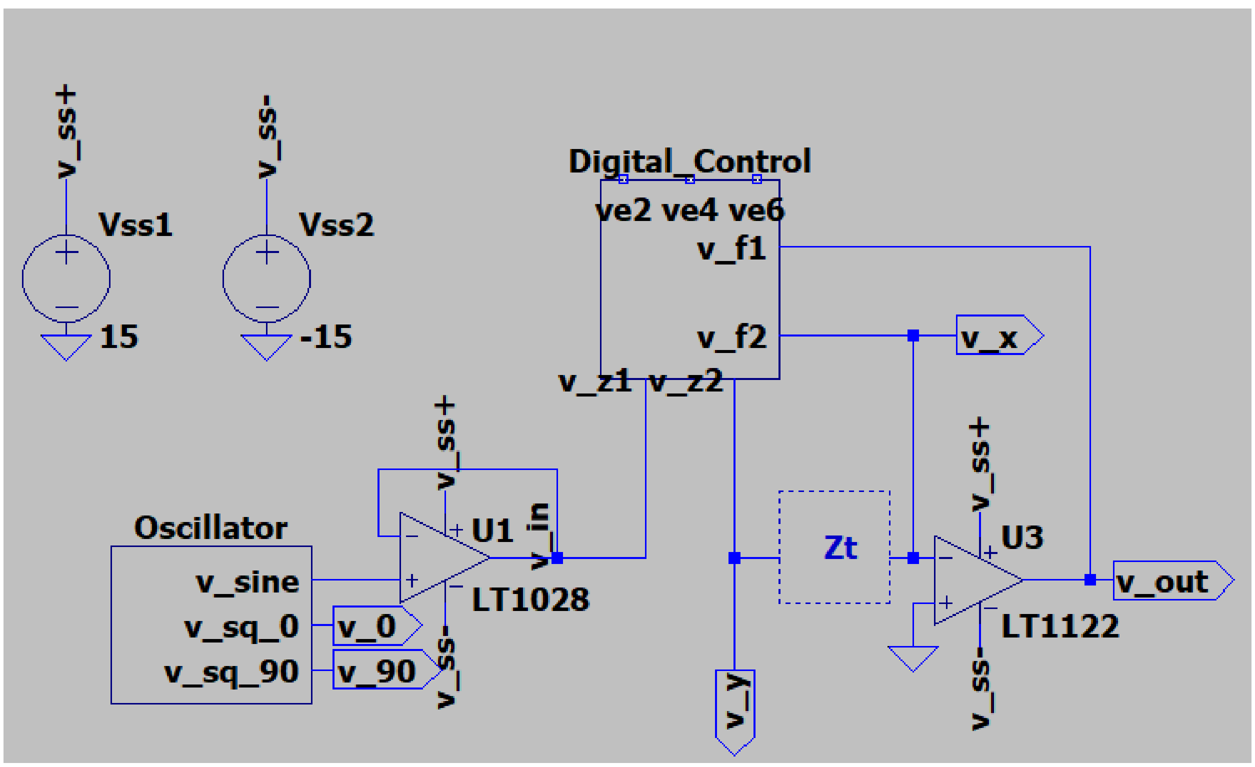
$\frac{V_x - V_y}{Z_T} + \frac{V_x}{Z_s} + \frac{V_x - V_{out}}{Z_f} = 0$, where Zs = Rin||1/Cin, Zf = Rf||1/Cf.

Thus, $Z_T = \left[\frac{V_y}{V_y - V_x}\frac{1}{Z_s} + \frac{V_x - V_{out}}{V_y - V_x}\frac{1}{Z_f}\right]^{-1}$

Circuit Diagram



Physical Switch Control



LCR Measurement Circuit

Analysis Code

The central part will be to convert the waveform into complex numbers. There are 2 methods available: inner product with square wave or FFT. Each addresses some problems and leaves the others unresolved.

When performing measurements, we will first see what is the crude structure of the device, and decide which method to use for precise measurements. The first method is better for capacitor-like devices, and the second method is better for inductor-like devices.

Method 1 - Square wave inner product

Here I address 4 problems:

- imperfection of the square wave shape (negligible)
- imperfection of the oscillator frequency (small but noticeable)
- phase offset of the square waves (significant)
- DC offset of the measured voltages (significant)

There are 2 problems left unresolved:

- the harmonic distortion of the input sine wave (significant for inductors)
- the duty cycle error of the square waves (negligible)

These 2 problems are not possible to solve with this approach, and their effects are unclear.

For square wave phase error, the first inner product would produce $V_0 = cos(\alpha)$, and the second inner product would produce $V_{90} =- sin(\alpha)$ while actually it produces $V_b = cos(\alpha + \beta)$, and $\beta \approx \pi/2$

$$cos(\alpha + \beta) = cos(\alpha)cos(\beta) - sin(\alpha)sin(\beta)$$

$$V_{90} =- sin(\alpha) = (cos(\alpha + \beta) - cos(\alpha)cos(\beta))/sin(\beta) = (V_b - V_0 cos(\beta))/sin(\beta)$$

This corrects the square wave phase offset.

Method 1 Code

```
% use square wave to compute
function [ape,pe,val] = calct(table,rf,cf,zt_actu,phase_actu,cs,rs)
    g.table=table; % waveform data

    %% this section convert the square wave into perfect unity magnitude square waves
    g.v0_t0 = find(g.table.("V(v_0)")>2,1); % starting index for high V0
    g.v0_t1 = find(g.table.("V(v_0)")(g.v0_t0:end)<2,1)+g.v0_t0-1; % ending index for high V0
    g.v0_t2 = find(g.table.("V(v_0)")(g.v0_t1+1:end)>2,1)+g.v0_t1; % starting index for high V0
    g.v90_t0 = find(g.table.("V(v_90)")>2,1); % starting index for high V0
    g.v90_t1 = find(g.table.("V(v_90)")(g.v90_t0:end)<2,1)+g.v90_t0-1; % ending index for high V0
    g.v90_t2 = find(g.table.("V(v_90)")(g.v90_t1+1:end)>2,1)+g.v90_t1; % starting index for high V0

    % this function calculates inner product with perfect square wave
    g.func = @(var,time,t0,t1)(sum(var(t0:t1-1).*(time(t0+1:t1)-time(t0:t1-1))/(time(t1)-time(t0))));

    %% this section address the problem of frequency and square wave phase error
    % calculate frequency from square wave, offsetting error from oscillator
    f=1/(g.table.time(g.v0_t2)-g.table.time(g.v0_t0));
    % calculate phase shift from square wave, offsetting error from oscillator
    dth=(g.table.time(g.v90_t0)-g.table.time(g.v0_t0))*f*2*pi;

    zs=rs/(1+2*pi*1i*f*cs*rs); % calculate parasitic input impedance Zs
    zf=rf/(1+2*pi*1i*f*cf*rf); % calculate feedback impedance Zf

    %% this section calculates the complex form of Vy-Vx, Vx-Vout, and Vx, see below for calcz
    vyvx = calcz(g,g.table.("V(v_y)")-g.table.("V(v_x)"),dth); %Vy-Vx
    vxvout = calcz(g,g.table.("V(v_x)")-g.table.("V(v_out)"),dth); %Vx-Vout
    vx = calcz(g,g.table.("V(v_x)"),dth);% Vx

    zt.val=1/(vx.val/vyvx.val/zs+vxvout.val/vyvx.val/zf);
    zt.amp=abs(zt.val);
    zt.rad=angle(zt.val);
    zt.deg=zt.rad/pi*180;

    pe=zt.deg-phase_actu;
    ape=(zt.amp/zt_actu-1)*100;
    val=zt;
end

% calculate complex exponentials through square wave
function z = calcz(g,var,dth)
    z.dc = g.func(var,g.table.time,g.v0_t0,g.v0_t2);% calculate DC offset
    z.v0 = g.func(var,g.table.time,g.v0_t0,g.v0_t1)/2-z.dc/2; % calculate 0 phase product
    z.v90 = g.func(var,g.table.time,g.v90_t0,g.v90_t1)/2-z.dc/2; % calculate dth phase product
    z.v90 = (z.v90-z.v0*cos(dth))/sin(dth); % corrects the square wave phase error
    z.amp = sqrt((z.v0^2+z.v90^2))*pi; % calculate amplitude
    z.rad = atan2(-z.v90,z.v0);
    z.deg = z.rad/pi*180;
    z.val = z.amp*exp(1j*z.rad);
end
```

Method 2 - Fourier Transform

The previous method does not address harmonic distortion, so we think of another approach to deal with it. We would take 3 periods of the waveform of Vx, Vy, and Vout, resample it, and compute the Fourier transform. Since the sample contains 3 periods, the first value in the spectral series will be DC offset, the 4th value in the spectral series will be the value of Vx, Vy, and Vout in complex form, and the (1+3n)th value will be the harmonic distortions.

This approach addresses the problems of:

- Harmonic distortion
- DC offset
- Any square wave errors as it does not use the square wave

This approach left these problems unresolved:

- Oscillator frequency error

The error in frequency will make the sample contain extra or less data points than 3 complete cycles. The effect is unknown and may vary with the type of device to be measured, as well as the relative magnitude between Rz and ZT.

Method 2 Code

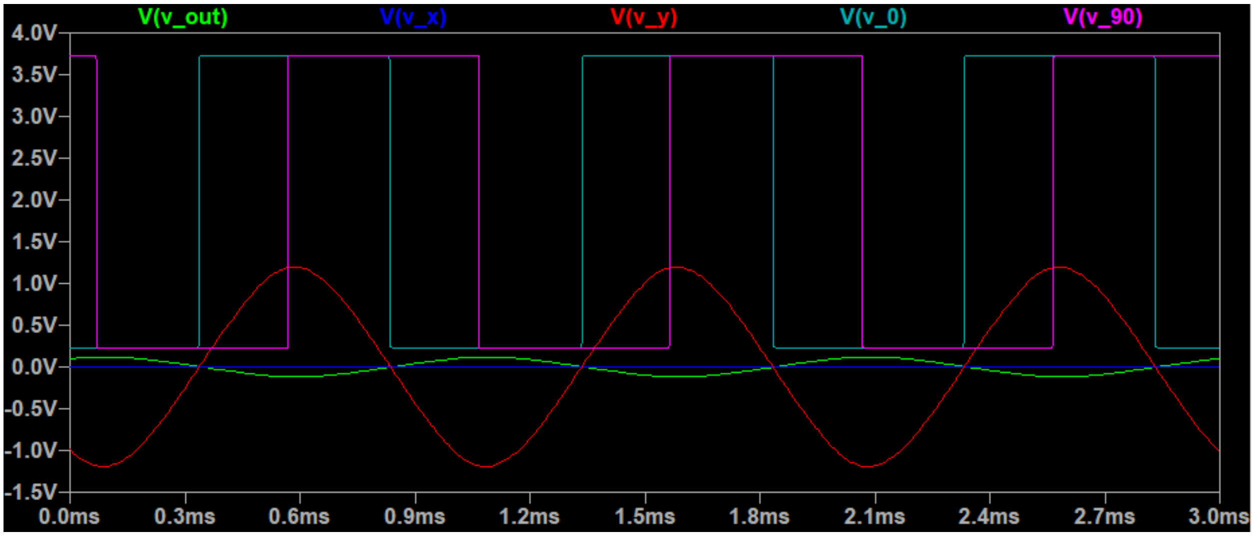
```
% use FFT to compute
function [ape,pe,val] = calca(table,rf,cf,zt_actu,phase_actu,cs,rs,f)
    peak=4; % sample for 3 periods, so the first harmonic appears at 4th value
    vy.raw=table.("V(v_y)"); % raw time domain v_y
    vx.raw=table.("V(v_x)"); % raw time domain v_x
    vo.raw=table.("V(v_out)"); % raw time domain v_out
    time=table.time; % time vector
    tn = size(time,1); % size of time vector
    n = 16384; % sample size
    vy.rs = zeros(n,1); % resampled v_y
    vx.rs = zeros(n,1); % resampled v_x
    vo.rs = zeros(n,1); % resampled v_out
    j=1;
    for i = 1:n
        t0 = i/n*time(end); % current timestamp
        while(j<tn && time(j)<t0)
            j = j+1;
        end
        vy.rs(i) = vy.raw(j);
        vx.rs(i) = vx.raw(j);
        vo.rs(i) = vo.raw(j);
    end
    vy.f=fft(vy.rs); % frequency domain of v_y
    vy.val=vy.f(peak); % first harmonic of v_y
    vx.f=fft(vx.rs); % frequency domain of v_x
    vx.val=vx.f(peak); % first harmonic of v_x
    vo.f=fft(vo.rs); % frequency domain of v_out
    vo.val=vo.f(peak); % first harmonic of v_out

    zs=rs/(1+2*pi*1i*f*cs*rs); % calculate input impedance Zs
    zf=rf/(1+2*pi*1i*f*cf*rf); % calculate feedback impedance Zf

    zt=1/(vx.val/(vy.val-vx.val)/zs+(vx.val-vo.val)/(vy.val-vx.val)/zf);

    amp=abs(zt);
    rad=angle(zt);
    deg=rad/pi*180;

    pe=deg-phase_actu;
    ape=(amp/zt_actu-1)*100;
    val=zt;
end
```

LCR Measurement Circuit - Simulation

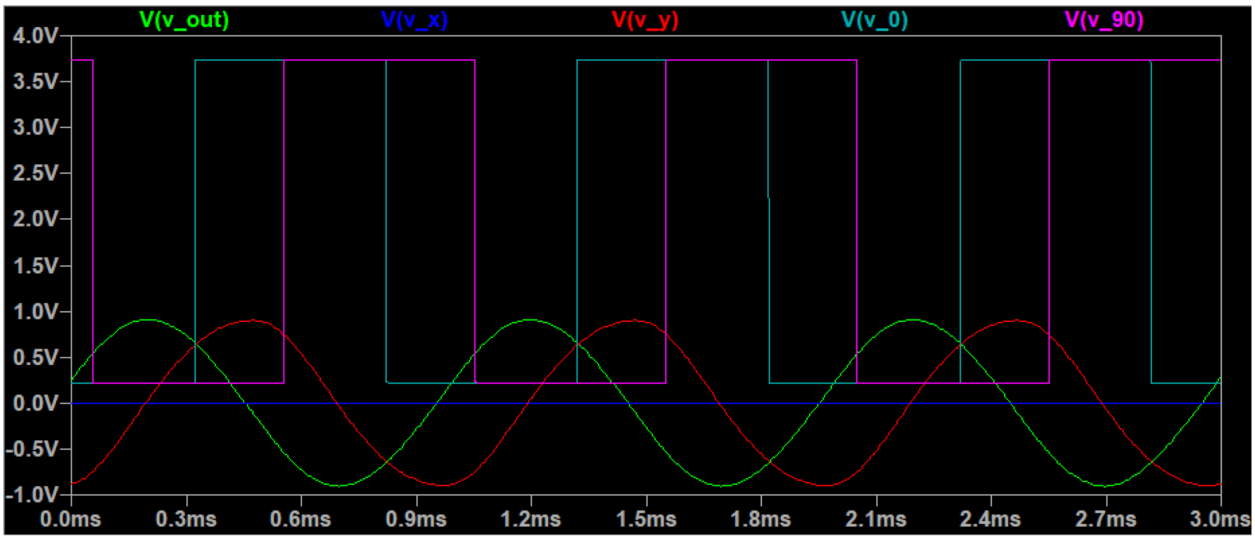
Single Element Test
Amplitude errors are shown in percent
phase errors are shown in degree
Both up to 2 digits of significance
All impedance values equal or equivalent to 1Ω ~ 10MΩ are shown in the table.
This table spans and exceeds the range of resistance, capacitance, and inductance required.
Errors larger than 0.1% or 0.1° will be marked with yellow.

	Resistors			Capacitors			Inductors		
Freq	R	Amp	Δphase	C	Amp	Δphase	L	Amp	Δphase
100Hz	1Ω	0%	0°	160pF	-0.06%	0.08°	1.6mH	-0.05%	-0.09°
	1kΩ	0%	0°	500nF	-0.02%	0.01°	4.8H	0.03%	-0.04°
	10MΩ	0%	0°	1.6mF	0.02%	0.03°	1.6kH	-0.02%	-0.02°
1kHz	1Ω	0%	0°	16pF	-0.14%	0.05°	160uH	0.18%	-0.10°
	1kΩ	0%	0.01°	50nF	0.14%	0.01°	480mH	0.04%	-0.03°
	10MΩ	0%	0.01°	160uF	0.13%	-0.02°	160H	-0.24%	-0.03°
10kHz	1Ω	-0.02%	0.03°	1.6pF	-0.01%	0.05°	16uH	0.28%	-0.08°
	1kΩ	-0.04%	0.05°	5nF	-0.11%	-0.05°	48mH	0.17%	0.02°
	10MΩ	-0.07%	0.01°	16uF	0.02%	0.14°	16H	0.19%	-0.03°
100kHz	1Ω	-0.16%	0.01°	0.16pF	0.06%	0.01°	1.6uH	-0.07%	-0.04°
	1kΩ	-0.15%	0°	500pF	0.04%	0°	4.8mH	-0.10%	-0.04°
	10MΩ	-0.13%	-0.02°	1.6uF	-0.08%	0.06°	1.6H	-0.07%	-0.04°

Typical Simulation Waveform



1kHz, Rz = Rf = 1MΩ, Cf = 5pF, Unknown R = 10MΩ



1kHz, Rz = Rf = 1MΩ, Cf = 5pF, Unknown L = 160H

Series RLC Test
R = 10kOhm, L = 1.6H, C = 16nF

Impedance Measurement Table
Valid choice of Rf is marked green.

	100Hz	1kHz	10kHz	100kHz
Actual ‖ZT‖	98.973 kΩ	9.9994 kΩ	100.4 kΩ	1.0053 MΩ
Actual ∠ZT	-84.20°	0.61°	84.26°	89.43°
Rf=100Ω ‖ZT‖	99.023 kΩ	10.001 kΩ	99.53 kΩ	1.0087 MΩ
Rf=100Ω ∠ZT	-84.00°	0.50°	84.21°	89.46°
Rf=10kΩ ‖ZT‖	98.988 kΩ	10.001 kΩ	100.13 kΩ	1.0089 MΩ
Rf=10kΩ ∠ZT	-84.11°	0.50°	84.41°	89.46°
Rf=1MΩ ‖ZT‖	98.616 kΩ	10.004 kΩ	100.81 kΩ	1.0056 MΩ
Rf=1MΩ ∠ZT	-84.21°	0.50°	84.17°	89.43°

The measured impedance shows that the amplitude of the impedance is falling and then rising. Thus, it means that it has one pole and 2 zeros. Thus, it must be R||C+L (pole not at origin) or R+L+C (pole at origin). The pole is likely to be at origin, thus this device is likely to be R+L+C circuit.

It demonstrates that our measurement circuit can determine the series RLC circuit precisely.

Parallel RLC Test
R = 10kOhm, L = 1.6H, C = 16nF

Impedance Measurement Table
Valid choice of Rf is marked green.

	100Hz	1kHz	10kHz	100kHz
Actual ‖ZT‖	1010Ω	9999.4Ω	999.6Ω	99.48Ω
Actual ∠ZT	84.20°	-0.61°	-84.26°	-89.43°
Rf=100Ω ‖ZT‖	1011Ω	10001Ω	998.4Ω	99.48Ω
Rf=100Ω ∠ZT	84.26°	-0.52°	-84.36°	-89.44°
Rf=10kΩ ‖ZT‖	1013Ω	9998Ω	1002.4Ω	99.46Ω
Rf=10kΩ ∠ZT	84.11°	-0.50°	-84.19°	-89.27°
Rf=1MΩ ‖ZT‖	1011Ω	10000Ω	1001.2Ω	99.51Ω
Rf=1MΩ ∠ZT	83.93°	-0.50°	-84.24°	-89.31°

The measured impedance shows that the amplitude of the impedance is rising and then falling. Thus, it means that it has one zero and 2 poles. Thus, it must be (R+L)||C (zero not at origin) or R||L||C (zero at origin). The zero is likely to be at origin, thus this device is likely to be R||L||C circuit.

It demonstrates that our measurement circuit can determine the parallel RLC circuit precisely.

Discussion and Conclusion

Oscillator Design

In a real lab, calibrating a capacitor is not as easy as in a simulator, where we can just alter the values anyway we want. Thus we might not be able to control the frequency just right, and the frequency error will be larger. However, in a real lab we can also collect more data easily, while in simulation it might take far longer to collect more data. This makes the accuracy of the Fourier transform method more accurate in the real world, as we can increase the size of FFT and thus be able to correct for the frequency error.

Effect of Real World ADC Accuracy

In the real world, ADC might not be as fast and as accurate. However, this effect can be mitigated by taking more data and finding the average, which would take a very long time in simulation. Thus, both methods can either maintain the current accuracy or achieve higher accuracy in a real lab.

Determining Unknown Device

Given that the device is at most second order, we can use the 4 data points to roughly guess the shape of both the amplitude and the phase bode plot. Then, based on the bode plots, we can determine the structure. After that, formulate an expression for the amplitude and phase of the impedance as a function of frequency and containing unknown values of R, L, and C. Solve the equation with the measured data and finally find the R,L,C values. We have 8 data points and 3 unknown variables, so this should give a very accurate result with predictable error margin using statistical tools.