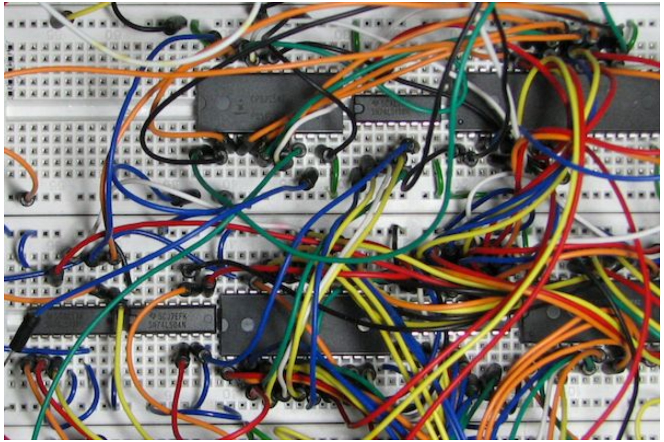
PHY405-L03

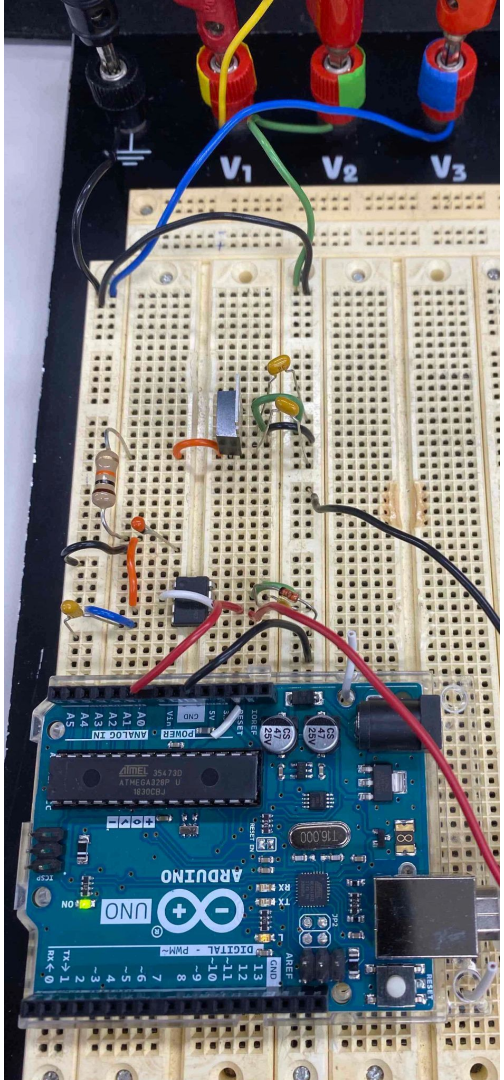
R, L, C, and Computer Control

Lab reminder

- Read lab manual **before** you come to the lab
- Read equipment manuals **before** you come to the lab
  - If you need something from equipment manuals, you need to know where to look it up…
- Design your circuit ahead of time
  - They are likely the ones you simulated last week
- Sign in
- Follow safety restrictions
- Bring your lab notebook
- If something doesn’t work, it’s expected. Debug and get help

Breadboard housekeeping



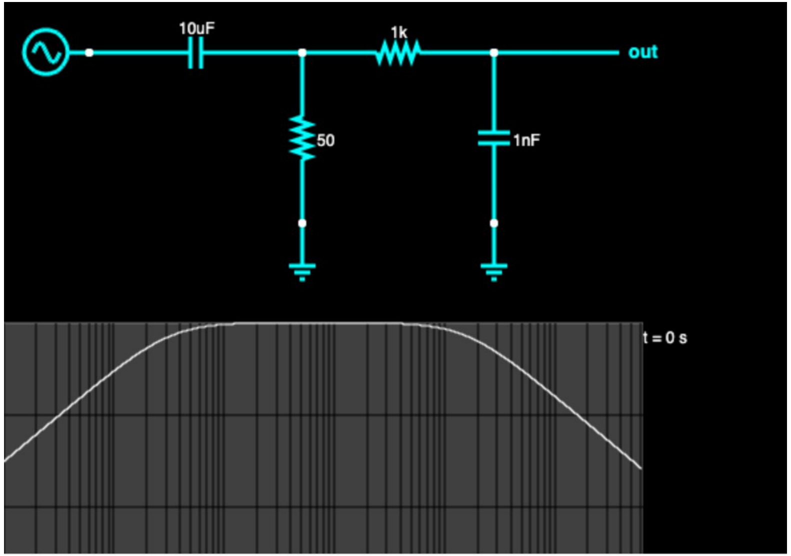


Lab 2 - R, L, C

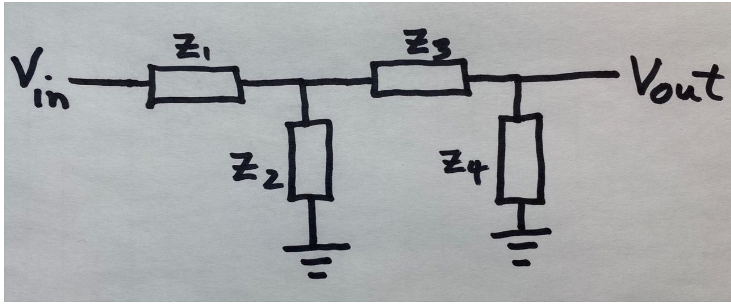
- Measurements of circuits with
  - Resistors
  - Capacitors
  - Inductors.
- Internal resistance
- RC and RL Circuits and Filters
- 3 ways to make Bode Plots
- Computer control of instruments

Bandpass filters

- Combine low pass and high pass filters, you can get a bandpass filter



Why bandpass order matters



$$\frac{V_{\text{out}}}{V_{\text{in}}} = \frac{Z_2 Z_4}{Z_1 Z_3 + Z_2 Z_4 + Z_1 Z_4 + Z_2 Z_3 + Z_1 Z_2}$$

Swap 1↔ 3, 2↔4

$$\frac{V_{\text{out}}}{V_{\text{in}}} = \frac{Z_2 Z_4}{Z_1 Z_3 + Z_2 Z_4 + Z_1 Z_4 + Z_2 Z_3 + Z_3 Z_4}$$

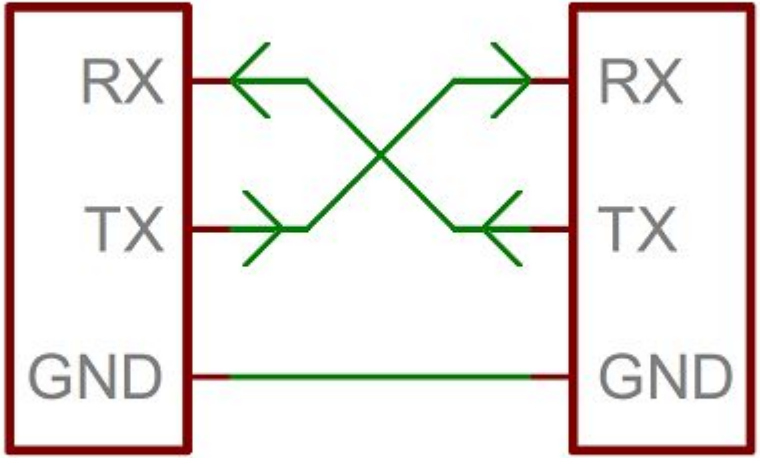
- Calculating the filter response is possible with these formulas, though might be easier with the simulation

Computer Control

- Computers and electronic instruments can be connected in various ways:
    - RS232 (Recommended Standard 232 for Serial port)
    - GPIB (General Purpose Interface Bus)
    - VXI , PXI, …
    - Ethernet, WiFi, Bluetooth, …
    - USB (Universal Serial Bus)
    - …

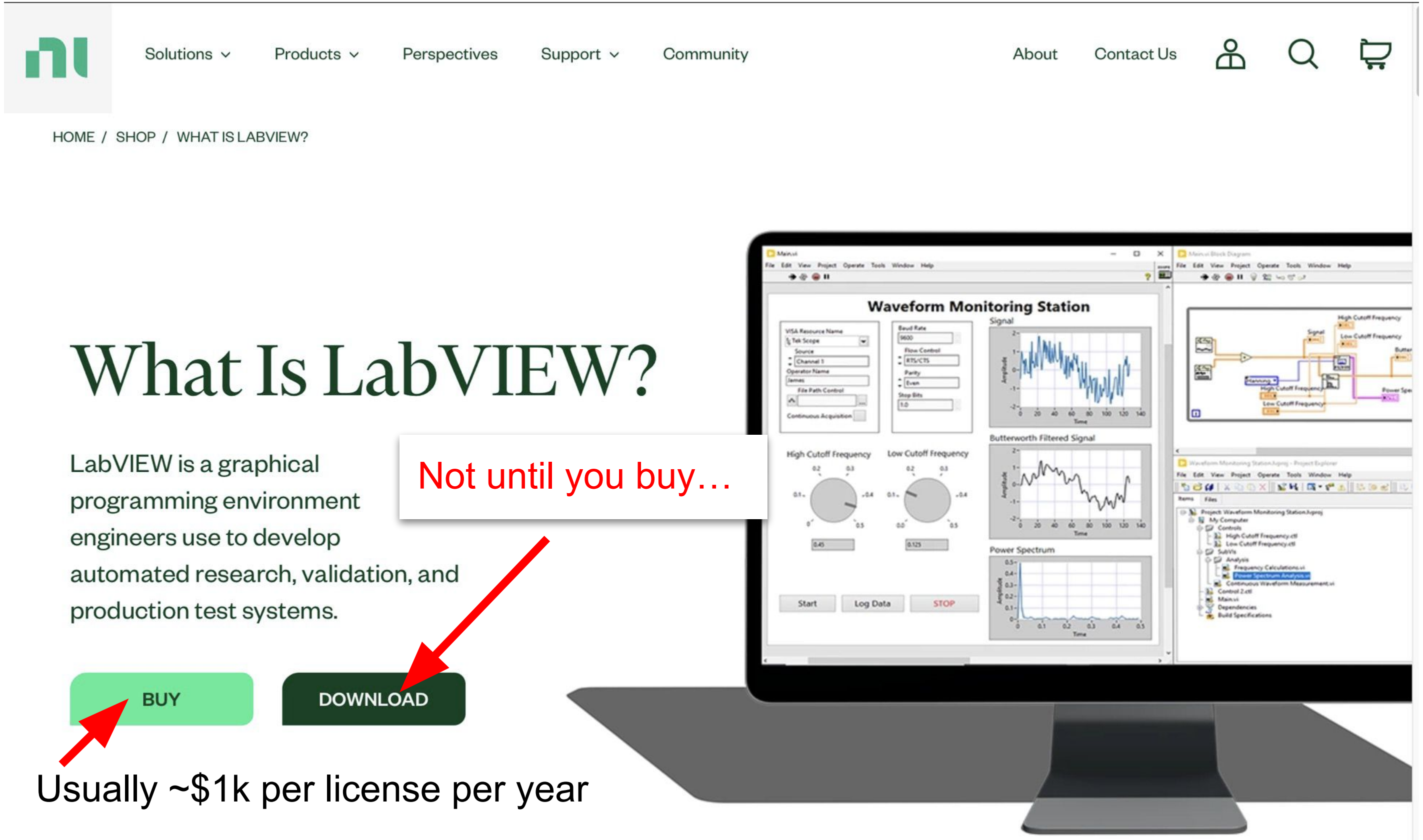
Serial / RS-232

- Serial port (often RS-232) is the most common communication protocol for experiment control
- 3 lines:
  - TX: Transmitter
  - RX: Receiver
  - GND
- Digital signals transmitted
  - +3.3 V or +5 V
- Often there's a cross-over
  - Causes a lot of headaches



https://learn.sparkfun.com/tutorials/serial-communication/wiring-and-hardware

National Instruments LabVIEW



Not until you buy…

Usually ~$1k per license per year

PyVisa

PyVISA is a Python package that enables you to control all kinds of measurement devices independently of the interface (e.g. GPIB, RS232, USB, Ethernet). As an example, reading self-identification from a Keithley Multimeter with GPIB number 12 is as easy as three lines of Python code:

```
>>> import pyvisa
>>> rm = pyvisa.ResourceManager()
>>> rm.list_resources()
('ASRL1::INSTR', 'ASRL2::INSTR', 'GPIB0::12::INSTR')
>>> inst = rm.open_resource('GPIB0::12::INSTR')
>>> print(inst.query("*IDN?"))
```

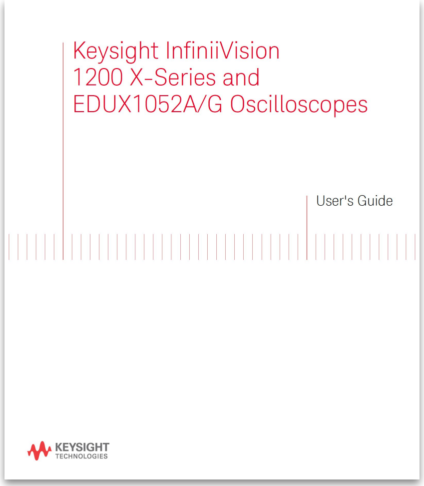
VISA: Virtual Interface Software Architecture

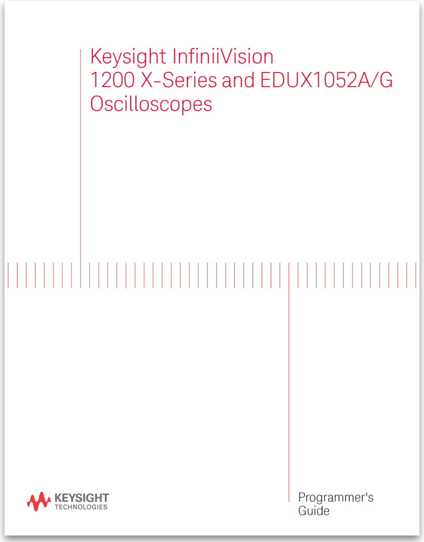
SCPI

- SCPI: Standard Commands for Programmable Instruments
- Eg. scope.query(":WGEN?")
  scope.write(":WGEN:OUTPUT ON")
  scope.write(":FRAN:DATA?")
  scope.read()
- Sending commands too fast can cause trouble
  scope.query("*ESR?")
- SCPI Introduction via this link
- query == write + read
- SCPI control → Pressing buttons with your computer

Device manuals and programmer’s guide

- Device manuals usually tell you how to manually use the equipment
    - Including the physics, technology, connections, specifications, limits, etc.
- Programmer’s guides tell you the computer control methods
    - Eg. SCPI commands
- Skimming through both ahead of time often speeds up experiments
- You need to know where to look up information when you need it





SCPI command example

- scope.write(":WGEN:OUTPUT ON")

:WGEN:OUTPut

N (see page 818)

Command Syntax :WGEN:OUTPut <on_off>

<on_off> ::= {{1 | ON} | {0 | OFF}

The :WGEN:OUTPut command specifies whether the waveform generator signal output is ON (1) or OFF (0).

Query Syntax :WGEN:OUTPut?

The :WGEN:OUTPut? query returns the current state of the waveform generator output setting.

Return Format <on_off><NL>

<on_off> ::= {1 | 0}

See Also · "Introduction to :WGEN Commands" on page 691

- scope.query(":WGEN:OUTPUT?")

SCPI command example

- scope.query("*ESR?")

Return Format <status><NL>

<status> ::= 0,..,255; an integer in NR1 format.

NOTE Reading the Standard Event Status Register clears it. High or 1 indicates the bit is true.

*ESR (Standard Event Status Register)

C (see page 818)

Query Syntax *ESR?

The *ESR? query returns the contents of the Standard Event Status Register. When you read the Event Status Register, the value returned is the total bit weights of all of the bits that are high at the time you read the byte. Reading the register clears the Event Status Register.

The following table shows bit weight, name, and condition for each bit.

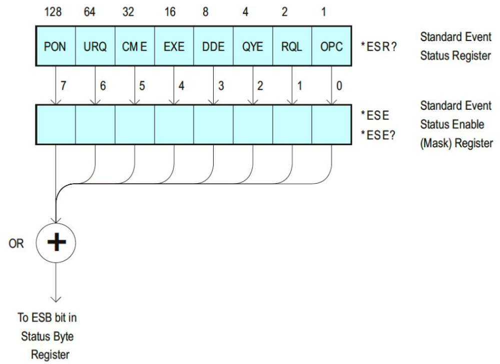


Table 42 Standard Event Status Register (ESR)

Bit	Name	Description	When Set (1 = High = True), Indicates:
7	PON	Power On	An OFF to ON transition has occurred.
6	URQ	User Request	A front-panel key has been pressed.
5	CME	Command Error	A command error has been detected.
4	EXE	Execution Error	An execution error has been detected.
3	DDE	Device Dependent Error	A device-dependent error has been detected.
2	QYE	Query Error	A query error has been detected.
1	RQL	Request Control	The device is requesting control. (Not used.)
0	OPC	Operation Complete	Operation is complete.

Example codes

- [PHY405_Test_USB_Control.py]

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
@summary:       Tests USB Control of standard PHY405 instruments
@filename:      PHY405_Test_USB_Control.py
@author:        David Bailey
@organization:  University of Toronto
@date:          6 October 2021
@contact:       dbailey@physics.utoronto.ca
@license:       MIT (https://opensource.org/licenses/MIT)
@Copyright:     2021 University of Toronto
```

- [PHY405_Bode_Plot_Analyze.py]

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
@summary:       Controls DSOX1204G Scope to create Bode Plot for a circuit.
@filename:      PHY405_Bode_Plot_Analyze.py
@author:        David Bailey
@organization:  University of Toronto
@date:          2 January 2022
@contact:       dbailey@physics.utoronto.ca
@license:       MIT (https://opensource.org/licenses/MIT)
@Copyright:     2022 University of Toronto
```

Note: You are expected to understand the example codes and modify them to suit your needs

PHY405_Bode_Plot_Analyze.py

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
@summary:       Controls DSOX1204G Scope to create Bode Plot for a circuit.
@filename:      PHY405_Bode_Plot_Analyze.py
@author:        David Bailey
@organization:  University of Toronto
@date:          2 January 2022
@contact:       dbailey@physics.utoronto.ca
@license:       MIT (https://opensource.org/licenses/MIT)
@Copyright:     2022 University of Toronto

@note: Scope commands can by found in the Keysight InfiniiVision 1200 X-Series
     and EDUX1052A/G Oscilloscopes Programmers Guide
     https://www.keysight.com/ca/en/assets/9018-07747/programming-guides/9018-07747.pdf
"""
import pyvisa as visa
from matplotlib import pyplot
from datetime import datetime
import time
import numpy

rm = visa.ResourceManager('')  # use default backend

# Print list of instruments detected on USB bus
print("*** USB Instruments Detected ***")
for resource in rm.list_resources():
    try :
        instrument=rm.open_resource(resource)
        ID = instrument.query("*IDN?")
    except :
        print("    ", "Resource does not respond to IDN query")
    print("\n  ", ID[:-1])  # the "-1 is to eliminate unneeded white space

# Some instruments are invisible to subsequent instrument.query, so reset
#   resource manager and then reopen it.
rm.close()
rm = visa.ResourceManager('')  # use default backend
```

PHY405_Bode_Plot_Analyze.py --- 2

- Find and initialize scope

```
print("\n*** Instrument Control Started ***")

try :
#   Search for scope on bus and, if found, open it for control
for resource in rm.list_resources():
    # Check for Keysight DSOX1204G scope
    if resource[6:20] == "0x2A8D::0x0396" :
        scope = rm.open_resource(resource)
        print("\n  ", scope.query("*IDN?"))
except :
    print ("Keysight DSOX1204G Oscilloscope either not detected or an error",
           "occurred.")

# Clear Scope Status and ReSeT scope to default setting
scope.write("*CLS;*RST")
# Wait a couple of seconds for reset to complete
time.sleep(2)
```

PHY405_Bode_Plot_Analyze.py --- 3

- Do frequency analysis

```
# Do Frequency Analysis
scope.write(":FRAN:ENAB ON")               # Turn Fequency Analysis on
scope.write(":FRAN:FREQ:STAR 10Hz")        # Set start frequency
scope.write(":FRAN:FREQ:STOP 100kHz")      # Set stop frequency
scope.write(":FRAN:SWE:POIN 32")           # Set number of point to sweep
scope.write(":FRAN:RUN")                   # Start analysis
# Loop waiting for analsys to finish
for i in range(100):
    time.sleep(1)
    try :
        # Check the OPeration Complete status, which is bit 1 in the
        #   standard Event Status Register
        OPC=int(bytes(scope.query("*ESR?"), 'utf-8'))&1
        # If analysis operation complete, wait another second then exit loop
        if OPC == 1:
            time.sleep(1)
            break
    except :
        continue
    print("Still taking data after ", i, "seconds.")
# End frequency analysis
```

PHY405_Bode_Plot_Analyze.py --- 4

- Read out data

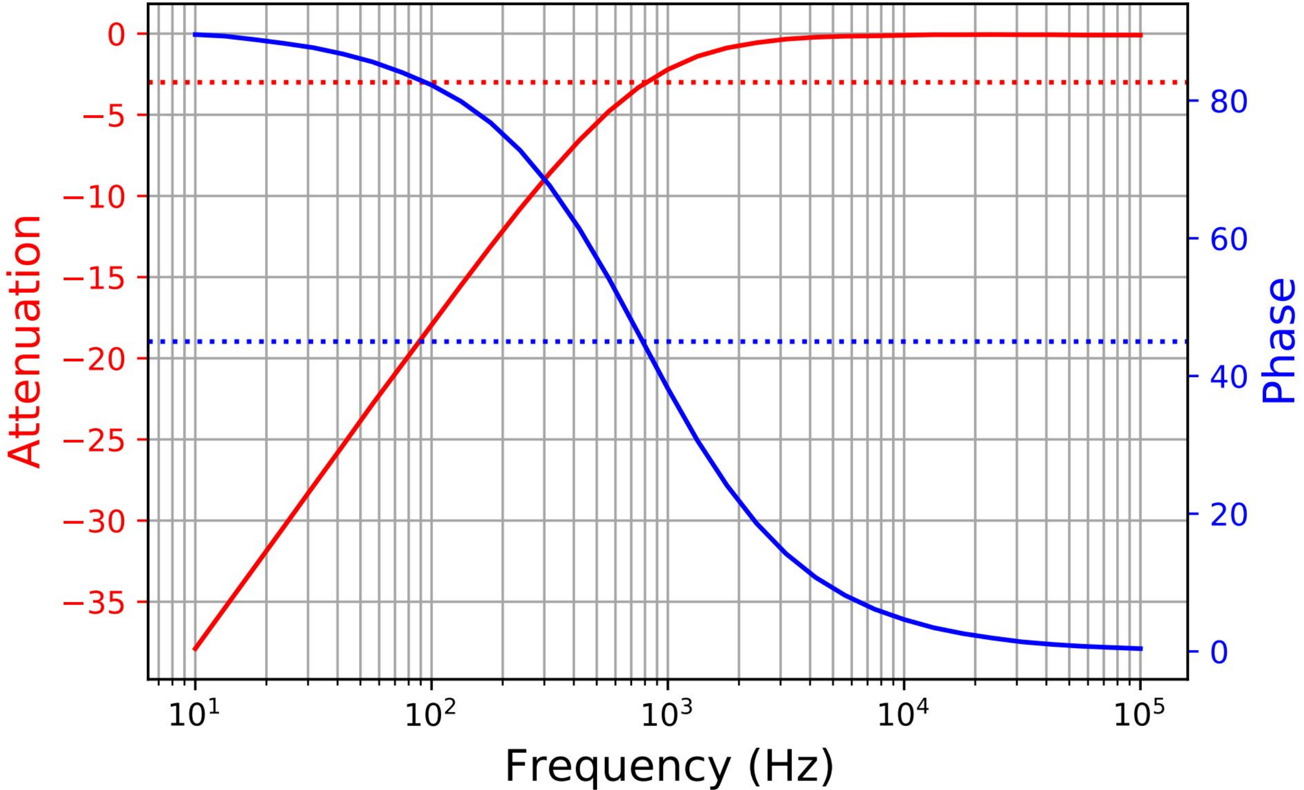
```
freq, db, phase = [],[],[]
# Read out Bode plot from scope
scope.write(":FRAN:DATA?")
data_raw = scope.read_raw()
# The scope output has a header that should be ignored and a specific encoding
data = data_raw[13:].decode("iso-8859-1")
print(data)
# Data is a string, so needs to be split and parsed
lines = data.splitlines()
for i in range(1,len(lines)-1) :
    measurements = [float(j) for j in lines[i].split(",")]
    freq.append(measurements[1])
    db.append(measurements[3])
    phase.append(measurements[4])

print("\nFrequencies:", freq)
print("\nAttenuation:", db)
print("\nPhase      :", phase)
```

PHY405_Bode_Plot_Analyze.py --- 5

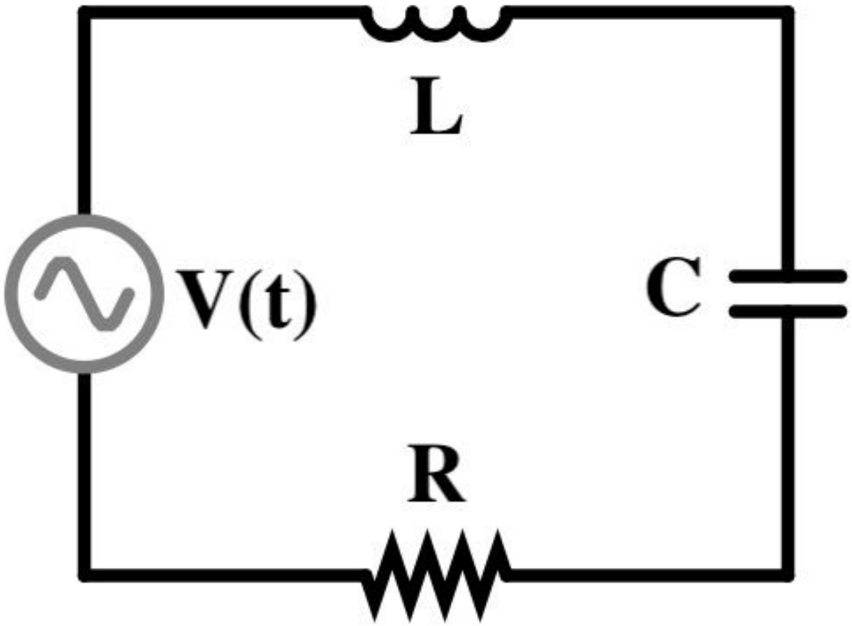
- Make plots

```
# Make Bode Plot for attenuation and phase
fig,ax=pyplot.subplots()
ax.plot(freq, db,color="red")
ax.set_xlabel("Frequency (Hz)",fontsize=14)
ax_colour = "red"
ax.set_ylabel("Attenuation",color=ax_colour,fontsize=14)
ax.tick_params(axis='y', colors=ax_colour)
# Create different (phase angle) right hand axis plot
ax2=ax.twinx()
ax2_colour="blue"
ax2.plot(freq, phase,color=ax2_colour)
ax2.set_ylabel("Phase",color=ax2_colour,fontsize=14)
ax2.tick_params(axis='y', colors=ax2_colour)
# Draw lines indicating 3db and ±45° lines
#   Whether +45° or -45° depends on the sign of the average phase
ax.axhline(y=-3, color=ax_colour, linestyle=':')
ax2.axhline(y=45*numpy.sign(numpy.average(phase)),
        color=ax2_colour, linestyle=':')
pyplot.xscale("log")
ax.grid(True, axis='both', which='both', ls="-", color='0.65')
```

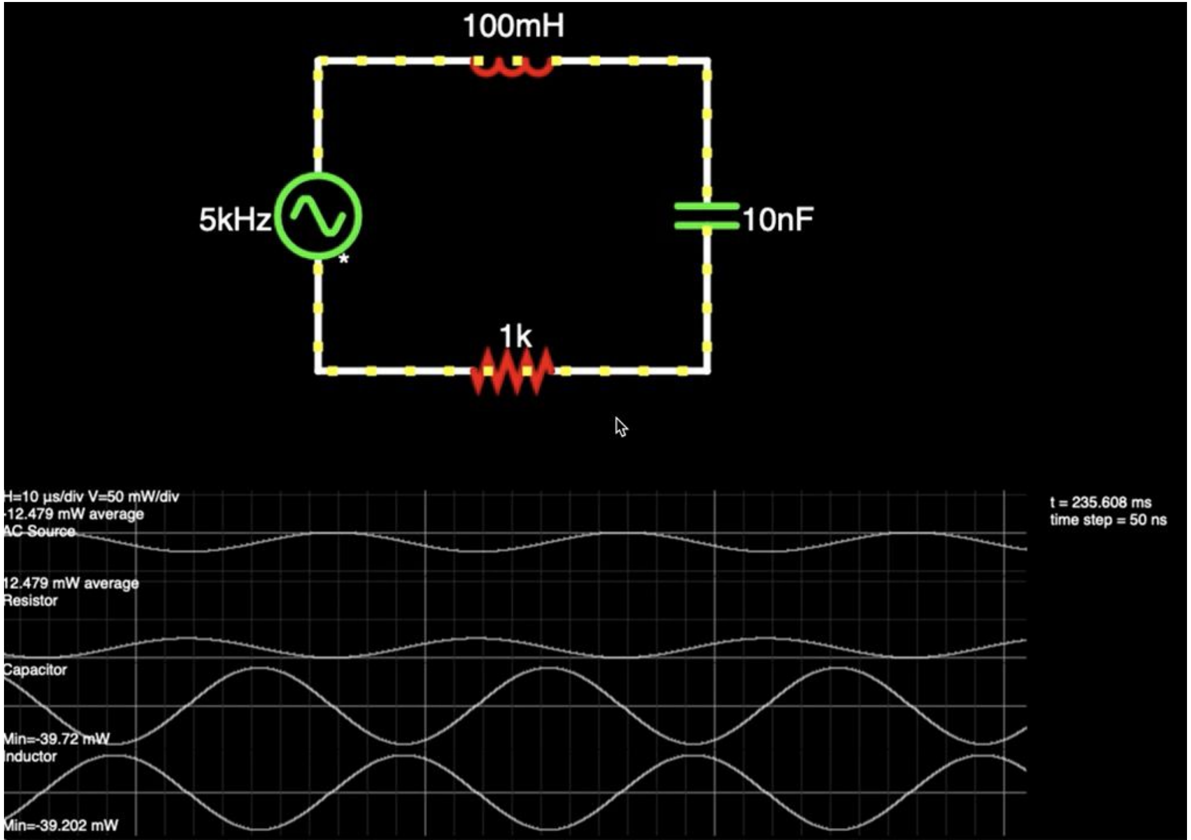
Example Bode plot



LCR Circuits

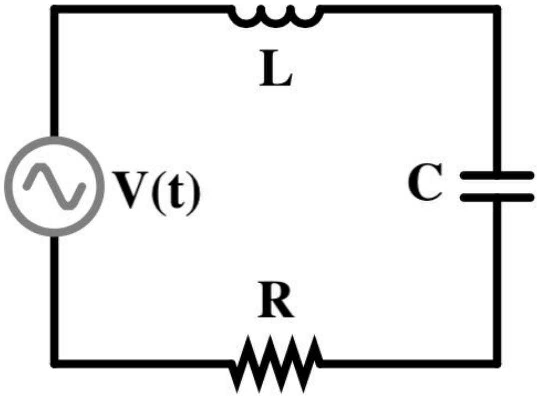


Series LCR Falstad Simulation



Impedance of LCR in series

$$
\begin{aligned}
Z_{RLC} &= Z_R + Z_L + Z_C \\
&= R + j\omega L + \frac{1}{j\omega C} \\
&= R + j\left(\omega L - \frac{1}{\omega C}\right) = |Z_{RLC}|e^{j\phi} \\
&= R + j\frac{L}{\omega}\left(\omega^2 - \frac{1}{LC}\right) \\
\therefore |Z_{RLC}| &= R\sqrt{1 + \frac{L^2}{R^2\omega^2}\left(\omega^2 - \frac{1}{LC}\right)^2} \\
\phi &= \tan^{-1}\frac{L\left(\omega^2 - \frac{1}{LC}\right)}{\omega R}
\end{aligned}
$$



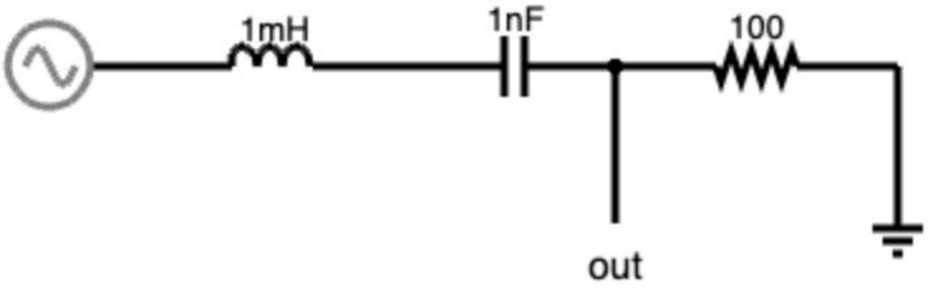
LCR Resonance

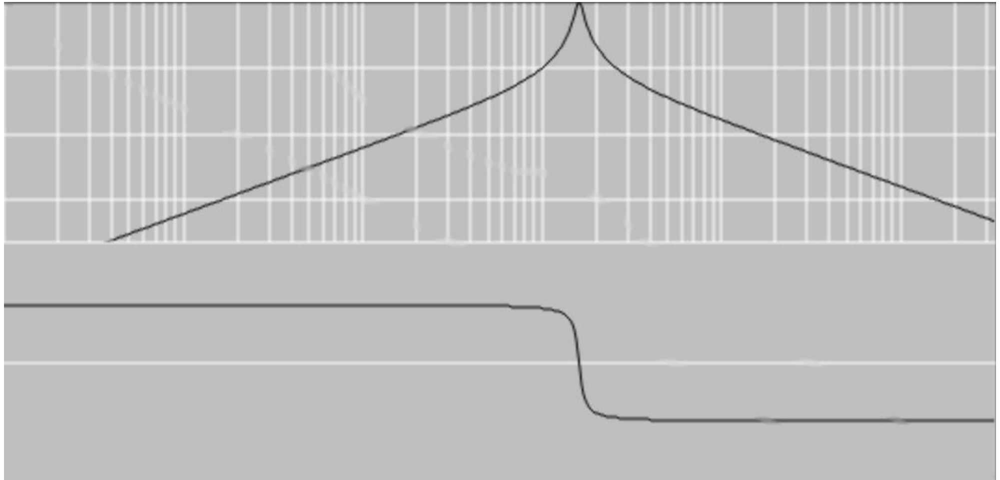
- Because the impedances of inductors and capacitors are 180° out-of-phase, they can cancel each other out at the resonant frequency where their magnitudes are the same:

$$f_0 = \frac{1}{2\pi}\omega_0 = \frac{1}{2\pi}\frac{1}{\sqrt{LC}}$$

- The impedance has it minimum value, $|Z_0| = R$, at resonance, so the current (and hence voltage drop across the resistor) is a maximum.

Simple LCR Resonant circuit





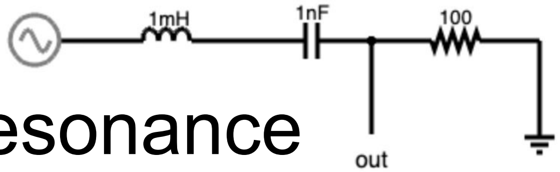
[Falstad Filter example]()

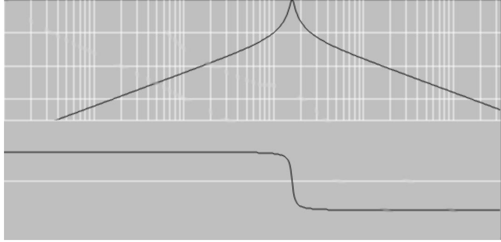
Attenuation

$$A(f) = \frac{|V_{\text{output}}\, f|}{|V_{\text{input}}\,(f)|}$$

Phase

Phase and Width of Resonance



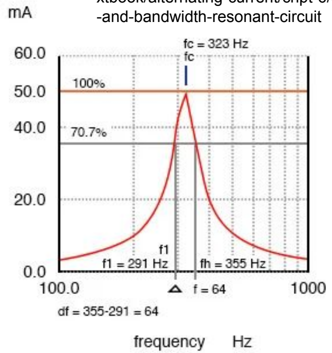


- At high frequencies the inductor dominates the impedance so the voltage V leads the current I, while at low frequencies the capacitor dominates so I leads V. At resonance, ϕ_0 = 0.
- The width of the resonance is parameterized by the half-power points (ω±) on either side of the resonant frequency.
    - The circuit current is $I = V_{AC}/Z_{LRC}$, so the resistor power dissipation $P = I^2 R = V_{AC}^2 R/Z_{LRC}^2$ is halved when circuit impedance Z_{LRC} increases by $\sqrt{2}$
- This happens when $\left(\omega L - \frac{1}{\omega C}\right)^2 = R^2$
- I.e. the half-frequencies are $f_\pm = \frac{\omega_\pm}{2\pi} = \frac{\pm R + \sqrt{R^2 + 4L/C}}{4\pi L}$

The phase at half-power points is 45°

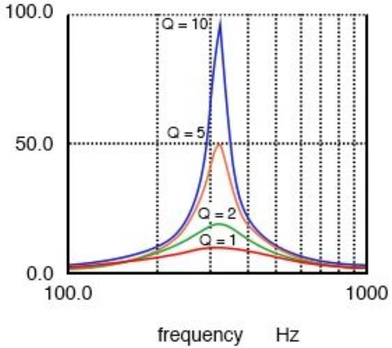
Bandwidth and Q value of resonance

https://www.allaboutcircuits.com/textbook/alternating-current/chpt-6/q-and-bandwidth-resonant-circuit



- The bandwidth of a resonant circuit is the Full Width at Half Maximum (FWHM), i.e. the difference between the half-power points

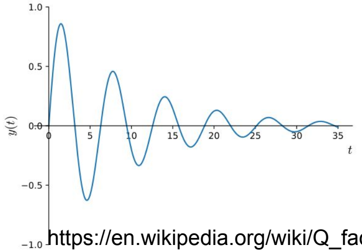
$$BW = f_{+} - f_{-} = \frac{R}{2\pi L} = \frac{f_0}{Q}$$



- where Q is the quality factor:

$$Q = \frac{1}{R}\sqrt{\frac{L}{C}}$$

- Roughly speaking, is the decay time of energy stored in the circuit, measured in units of the period (i.e. inverse of frequency)



https://en.wikipedia.org/wiki/Q_factor

Bandwidth (BW)

- The analog bandwidth of an oscilloscope (and other systems) is conventionally defined as the frequency range over which the output amplitude of an input sine wave is reduced by no more than 3 db (~ 30%), and hence the power is reduced by no more than a factor of two.
- The rise time (t_{rise}), of an oscilloscope, (the time it takes for the oscilloscope to trace out an instantaneous change in its input), is related to the bandwidth. Typically $BW \times t_{\text{rise}}$ ~ 0.35 − 0.55
  - Note: Decibels are defined in terms of power ratios $G = 10\log_{10}(P_{\text{out}}/P_{\text{in}}) = 20\log_{10}(V_{\text{out}}/V_{\text{in}})$ (since $P \propto V^2$).

Lab 2

https://www.physics.utoronto.ca/apl/405/Lab_02.html

PHY405 Electronics, Lab 2

Last updated: Jan. 8, 2023

Adapted from Prof. David Bailey's website in 2022

Overview

Measurements of circuits with resistors, capacitors, and inductors.

- Internal resistance
- RC and RL Circuits and Filters
- Computer control of instruments

Read over the whole lab write-up before starting, since knowing what comes later sometimes helps earlier.

Background preparation before starting the lab

Read through all these Lab instructions before Lecture.

- You will be building a real circuit that you previously simulated, so be sure you have access to the link to your circuit so you can build it and revise it if necessary.
- If you want to control your instruments from your own laptop, be sure you have pyVisa installed.
- You may not be able to finish in time if you only read these instructions after arriving in the lab.
- Ask questions at the Lecture or look things up *before* the lab.
- If asked to design a circuit or watch a video or look up some instrument specifications. Do this *before* the lab.

Remember

- By plotting data as you take it, you can avoid taking more data than you need.
- Note any anomalies.

Clean up after you are done

Yes, this is important enough to deserve it’s own slide

Questions?