Introduction

In this project we will design and build an Arduino Uno based capacitor tester. The intention is to be able to test capacitors to a variety of Voltages and to be able to generate accurate measurements from a very wide range of possible values from small ceramic to large electrolytic. The circuit to test this may also be used to test the leakage values of the capacitor being tested, however this is not within the current scope of the project.

Background

The goal of this project is to design and build an Arduino based capacitor tester that is capable of testing at a wide range of voltages and sizes of capacitors. Inspiration for this project is from using a digital multimeter to test a supercapacitor and it not working correctly at all. A purpose built capacitor tester costs hundreds of dollars (*amazon.com*) and has an upper limit of testable capacitance of 200 mF. In this project we propose that we can create an accurate and versatile device to accomplish this task.

This project does not aim to produce a marketable project and is available on Github under an open source license. Anyone is free to use or modify the source code and/or schematics including for commercial works under CC BY-SA 4.0 (*Creative Commons*). It uses 100% common and commercially available parts in order to make it easy reproducable. It is my hope that someone may come across this project and it may help them in their own work or hobby. If not, that’s fine too.

How it works

There are several circuit blocks that are essential to the working of the project. It is helpful to break the over into these parts

Power Supply and overvoltage protection

Power is supplied by a 0-30v bench power supply. This is where you set the voltage that you want to test the capacitor at. It has a current limiting function built in to reduce the chance of circuit damage should something go wrong. The relays, transistors and arduino will not accept this variable voltage thus I use a 7805 linear regulator to power these. I used a dremel to cut down a heatsink from discarded computer parts and attached it to the regulator with a nut and bolt with thermal compound between the regulator and heatsink. I also added a capacitor to the 5v rail to reduce any noise on this line.

Zener Diodes

This project uses Zener diodes. These semiconductor devices operate just like a normal diode when they are forward biased. The interesting part is that when you apply a reverse voltage greater than the zener voltage they will act as a short and allow current to pass. This makes them useful in power regulation and circuit protection cases. In this project 5.1v Zener diodes are used to protect the Analog pins of the Arduino from any programming oopsies that may result in a high voltage given to the arduino. If this error voltage happens the current gets shunted by a zener through the reset coil of the relay tasked with power protection. This near immediately will disconnect the main power supply rail from the entire circuit.

USB Cable

To connect the arduino I cut a USB cable in two and used the front panel USB headers from an old PC case. I connected the D+ and D- pins to the male USB end to the header. I also connected the ground line but left the 5v line from the PC disconnected in order to have the device powered completely from the bench power supply. The USB header gets connected to the 5v rail on the breadboard. The arduino then gets connected with a standard USB cable to the header I put together for easy disconnection.

Voltage Divider

The analog inputs of the arduino have a maximum voltage, being 5v that they can accept without damage. In order to test capacitors at voltages higher than 5v a voltage divider network was necessary. To accomplish this I used several non latching DPDT relays that are controlled by digital outputs on the arduino. When a test first starts the arduino takes a voltage measurement from a voltage divider that will have a voltage range of 0 to 5v no matter what the 30v power supply is set at. It uses this reading to determine what voltage divider relay to enable. On one side of the relay the supply side is always connected to the main supply voltage, fed through a voltage divider and connected to an analog input pin on the arduino. The other side has an identical voltage divider on it, but it is connected to the capacitor testing side which gets disconnected by a relay once the testing begins. In this fashion we can sample the voltage without introducing power from the supply or having any voltage dividers connected for longer than necessary as any current path to ground from the capacitor side of the device will introduce error into the reading, especially for low value capacitors.

Resistor Network

The drain network has a set of relays that connect the charged capacitor to ground through a

variety of resistors covering several orders of magnitudes of resistor values. By doing a quick estimation of the capacitor value through a mid range resistor, a second test can be done through a resistor of appropriate value over a meaningful amount of time. The ballpark time I chose for this is 10 seconds as it is a time value that will completely overshadow the switching time of the relays (data sheets put switching time at approximately 2ms)

Build Process

The circuit was built using a sparkfun breadboard and components purchased from digikey, other than a few USB related parts. The design was sketched out roughly on paper and segments of the circuit tested on a prototyping board. Once the final design was settled on all parts were soldered onto the sparkfun board. I also got a few parts from JAGR (local electronics supply).

All soldering was done using a Weller Portasol butane soldering iron. I highly recommend this iron as it heats quick, doesn't need an AC power source and it's exhaust is useful for heatshrink tubing. The only downside is that you need to refill it's butane fairly regularly, but cans of butane are cheap and available at any hardware store.

I built the circuit in what I consider to be the logical order. First I put together the power supply and protection components and tested to see if the regulator and Zener diodes operated as I expected. The overvoltage protection did not operate quite as expected, it cuts the circuit off from power at 8.5 volts instead of 5.1v. I used 5.1v Zener diodes to trip the circuit protection relay. This is something that may get revisited in the future.

After the power supply was tested and working I put together the voltage divider network. This is the area that gave the most headaches as sorting out transistor biasing issues and flyback voltages

from the relay coils caused the early death of quite a few PN2222A transistors. I ended up switching the design to using TIP120 darlington transistors as they have higher power handling and have a build in protection diode.

Finally the resistor network was the last to be built, this was relatively simple after the voltage divider network but it did require the use of a second breadboard in order to fit in all the compenents I needed.

Programming

The program written to operate the arduino is interupt driven. It will sit in the main loop and output the current reading from the calibration voltage divider connected to A0. To tell the arduino to make a reading it waits for an interupt on D2 (falling edge to detect button release). This calls the function that contains the control logic of taking a reading.

The input and output pins have been wrapped in a class in order to associate certain data to an individual hardware pin.

` The output class has a boolean value to track whether the pin is currently HIGH or LOW (HIGH==TRUE and LOW==FALSE in the source). This is not necessary but may serve as a useful function in the future

The input class keeps track of the last reading made on that pin, in certain cases it is desirable to re-use a reading instead of going to hardware for it. The analog input pins have a documented capacitance of 16 picoFarads, and I make the assumption that a non zero amount of energy is taken from the measured circuit by the arduino when a reading is taken. I also know for fact that when a voltage reading is taken from this circuit some amount of power gets dropped to ground via the voltage

divider so it is imperative that the voltage divider relays stay open as much as possible in order to minimize the error introduced to our measurements. The class retains the last value read from hardware and makes it available through get(). If we want to refresh the measurement we pass a boolean true to the get() function and it will return this fresh value. This class also records the time of the recorded measurement for later comparisons.

The first thing that we do when the user presses the button is to determine the voltage range of the power supply connected to the system. This allows us to make safe choices on which voltage dividers to use. the voltage divider operates linearly so the value read on A0 effectively maps a range of 0 to 30v to an int reading of 0 to 1023. From this reading we can easily determine which set of voltage divider relays to use for the remainder of the current measurement.

Now that the voltage range is known, we can attempt a quick approximation of the value of the capacitor under test. We do this by charging the capacitor, then discharging it briefly through a 1kohm resistor. We then do some math and determine which discharge resistor would be closest to giving a time constant of 10 seconds (seconds = ohm farads). The value of 10 seconds is nearly completely arbitrary other than seperating our experiment time from the relay switching time by several orders of magnitude (The relays specified here have an operate time of 5ms and a release time of 3 ms in their datasheet).

Now that we have picked a suitable value resistor to discharge though, we can recharge the capacitor to supply voltage. We then disconnect the charge relay and then immediately take a voltage reading across the capacitor. After 10 seconds have passed we reconnect the voltage divider and take our second reading. We now have all the information necessary to produce a capacitance value using formula 5 in Table 1.

After our measurement is produced we can send it to our serial output, or to an LCD display if you have one available. At this point it is prudent to tell all our relays to disconnect in case the user decides to change the supply voltage on the external power supply. Currently there is no provision for detecting a change in supply voltage during the test so it is dangerous to the Arduino to increase supply voltage during this phase.

In the introduction it is mentioned that this circuit could be used to measure leakage of a capacitor – do accomplish this would be a matter of writing code that would charge the capacitor up and then periodically poll for a current voltage, this would however be mostly just practical for detecting faulty caps as a high quality cap may take an extremely long time to drop a signifigant voltage.

Conclusion

In the process of designing and building this I learned and dipped my toe into several new things. I learned a great deal about the ins and outs of getting transistors and relays operating as expected, several things about transistors that I thought I knew and took for granted weren't entirely correct (BJT's being biased by current, not necessarly voltage, using an NPN transistor in the proper location in the circuit). LaTeX usage is now less scary to me, up until now I knew of it but had never used it. It is now installed on all my PC's as well as plugins added to Libreoffice in order to produce spiffy looking formulas within digital documents.

I cannot say that I met all the milestones that I would have liked to with this project. At some point in the future I would still like to design a PCB layout for a project like this, or perhaps for this project yet and do some DIY board etching. It is a technique that I have known of for a very long time but the preperation required, the PCB layout, dangerous chemicals and actual need to do it have kept it

on the “to do” list.

I would like to extend my appreciation for the opportunity to take this class, it’s not a requirement for my major however without it I may have never picked up a microcontroller and have been exposed to the suprising capabilities of these devices.

Math

1) $V_c = V_s e^{-t/\mathcal{T}}$ Capacitor discharge formula	V_s = Supply voltage
2) $\mathcal{T} = RC$	V_c = Capacitor voltage
3) $\frac{V_c}{V_s} = e^{-t/RC}$	t = elapsed time (seconds)
4) $ln(\frac{V_c}{V_s}) = \frac{-t}{RC}$	R = Discharge resistor (ohms)
5) $C = \frac{-t}{ln(\frac{V_c}{V_s})}$ Derived formula for capacitance	C = Capacitance (farads)
	$\mathcal{T}$ = Time constant

Table 1

Datasheets

TIP120 Darlington transistor http://www.onsemi.com/pub/Collateral/TIP120-D.PDF

DPDT non latching relay http://www.te.com/usa-en/product-1462000-1.html

DPDT Latching relay http://www.te.com/usa-en/product-3-1462034-0.html5.1v Zener diode https://www.fairchildsemi.com/datasheets/1N/1N5231C.pdf5v Linear Regulator http://www.st.com/content/ccc/resource/technical/document/datasheet/41/4f/b3/b0/12/d4/47/88/CD00000444.pdf/files/CD00000444.pdf/jcr:content/translations/en.CD00000444.pdfSolderable breadboard https://github.com/sparkfun/Solderable_Breadboard_Large

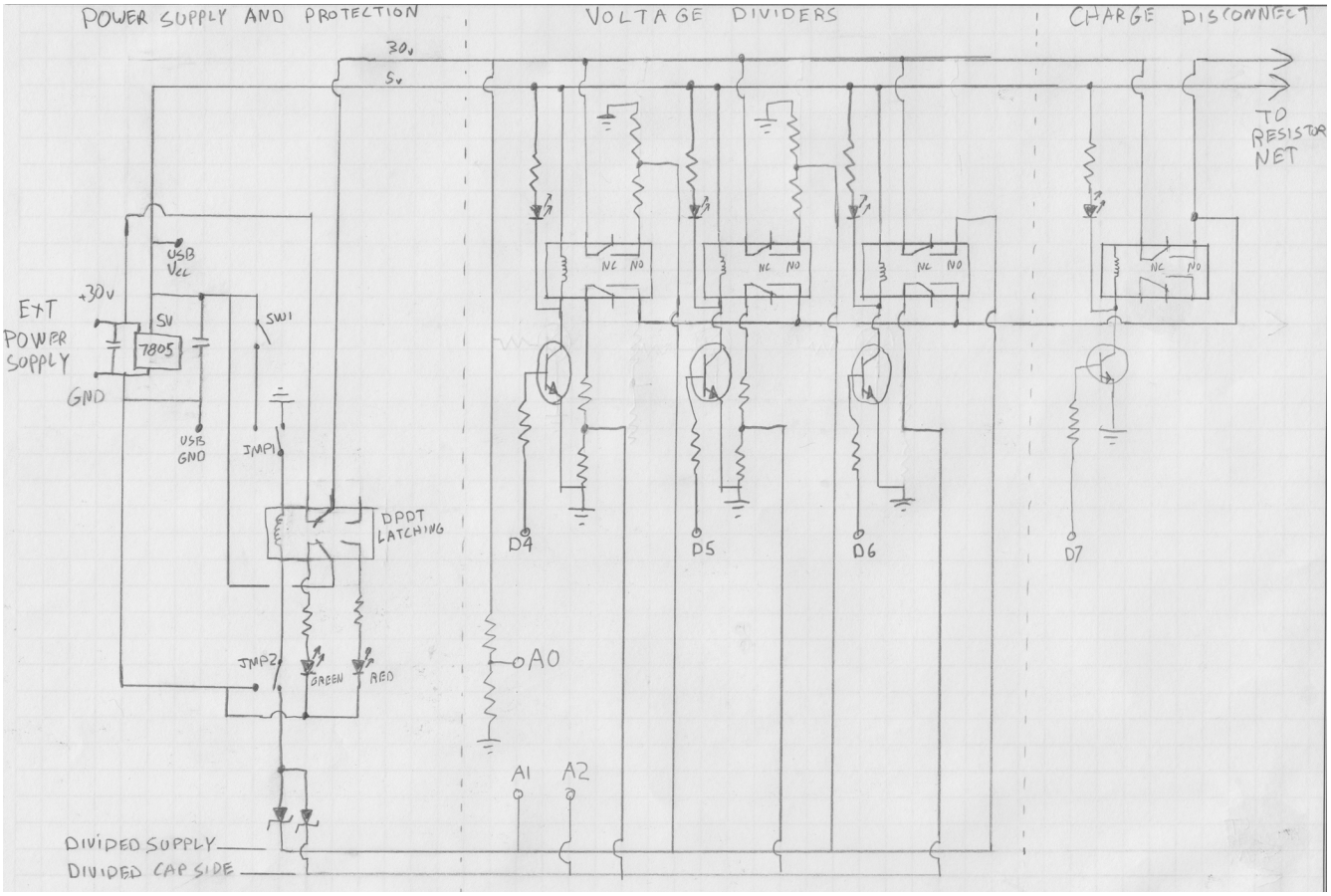
References

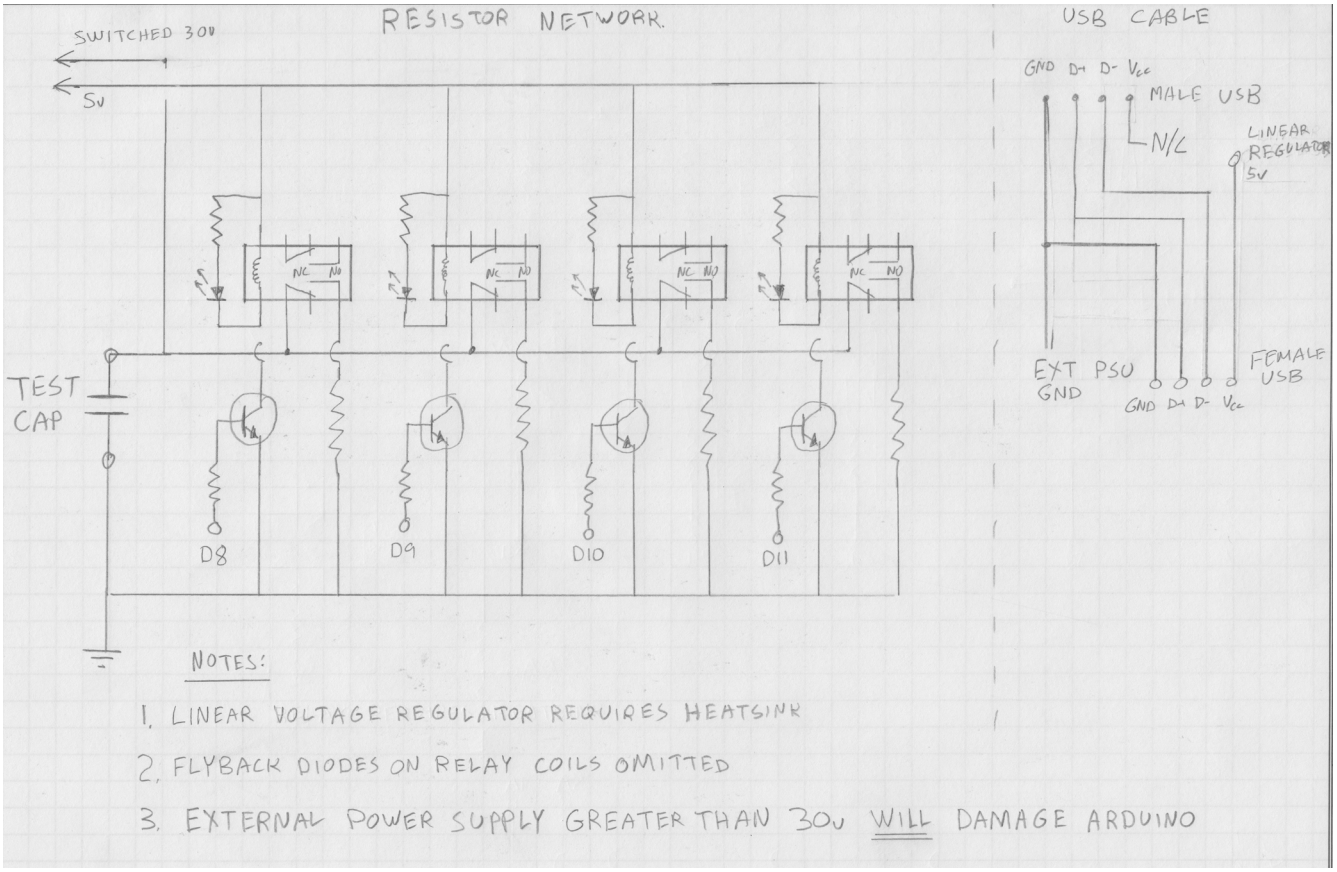
Amazon.com https://www.amazon.com/Precision-830C-Display-Handheld-Capacitance/dp/B0053ZCFVO/ref=lp_5011682011_1_15?s=industrial&ie=UTF8&qid=1523851785&sr=1-15

Creative Commons https://creativecommons.org/licenses/by-sa/4.0/

TexMaths - https://extensions.libreoffice.org/extensions/texmaths-1

TexLive - http://www.tug.org/texlive/


POWER SUPPLY AND PROTECTION
VOLTAGE DIVIDERS
CHARGE DISCONNECT
30v
5v
TO RESISTOR NET
USB Vcc
EXT POWER SUPPLY
+30v
5V
7805
GND
SW1
USB GND
JMP1
DPDT LATCHING
JMP2
GREEN
RED
D4
D5
D6
D7
A0
A1
A2
DIVIDED SUPPLY
DIVIDED CAP SIDE


RESISTOR NETWORK
SWITCHED 30v
5v
TEST CAP
D8
D9
D10
D11
USB CABLE
GND D+ D- Vcc
MALE USB
N/C
LINEAR REGULATOR 5v
EXT PSU GND
FEMALE USB
NOTES:
1. LINEAR VOLTAGE REGULATOR REQUIRES HEATSINK
2. FLYBACK DIODES ON RELAY COILS OMITTED
3. EXTERNAL POWER SUPPLY GREATER THAN 30v WILL DAMAGE ARDUINO