Bäschteler of Science Blog

bashtelor of science – a blog about my electronics and other creations

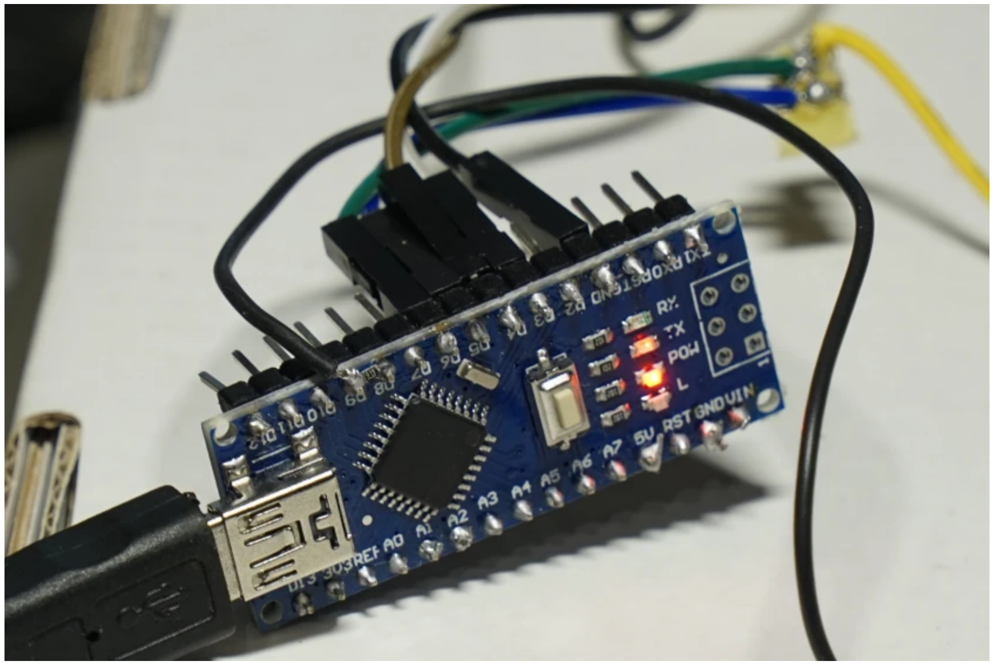
Advanced capacitive sensing with Arduino

…or any other microcontroller.

There are several methods to do capacitive sensing with a microcontroller. The easiest and most common way in the Arduino world is to charge the capacitive probe through a high value reseistor and measuring the time it takes to charge it. While this is indeed a simple method it is highly susceptable to interferance and noise as the wire acts as an antenna due to the high impedance. Also it is not very sensitive compared to other methods but good enough for a simple button.

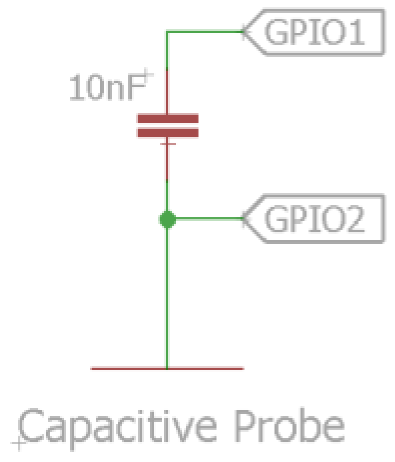
Here is a video comparing the resistor method to the two methods discussed in this blog post:

The Arduino code I wrote for this demo can be found at the bottom. Here is a close-up shot of the wiring: I soldered the 5MΩ resistor for the ‘resistor method’ directly between pins 8 and 9. The other two sensors are on separate prototyping boards and are connected with wires.



Charge transfer method

What is most commonly used in capacitive sensing ICs is the charge transfer method: the capacitor that is to be measured is charged repeatedly through a larger ‘sample’ or ‘reference’ capacitor that accumulates charge. Counting the number of cycles is a measure for the capacitance of the probe. This method can be implemented with two digital pins of a microcontroller and is way more sensitive and very stable. The method is described in detail here. Basically this is how it goes:



Both pins are set low to start with, the sample capacitor is discharged.

Step 1: GPIO2 is set as an input, GPIO1 is set to high, charging the the capacitive probe through the 10nF sampling capacitor. Since there is no connection to GND only a tiny amount of electrons flow into the capacitor. How many depends on the capacitance of the probe.

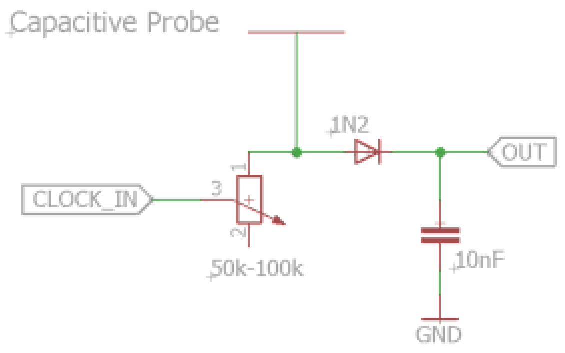
Step 2: GPIO1 is set as an input and GPIO2 is set to GND. This discharges the probe but does not discharging the sampling capacitor as GPIO1 is now floating and no current can flow.

Now repeat the two steps (set GPIO2 floating, set GPIO1 high, etc.) and with each cycle some more charge builds up on the sample capacitor. The voltage on GPIO1 is checked at the end of each cycle until eventually it reaches the 'high' trigger level. Counting the number of cycles gives an idea of the capacitance of the probe, it is by no means a good method to measure the exact capacitance as there are 'stray' capacitances all over the place influencing an exact measurement but for relative capacitance changes like in a proximity sensor or a capacitive slider switch this works very well.

The sample capacitor should be chosen somewhere between 1nF and 100nF depending on the size of the probe. Choosing a too large capacitor makes the measurement very slow and a too small capacitor will not give a good enough resolution.

The filter overflow method

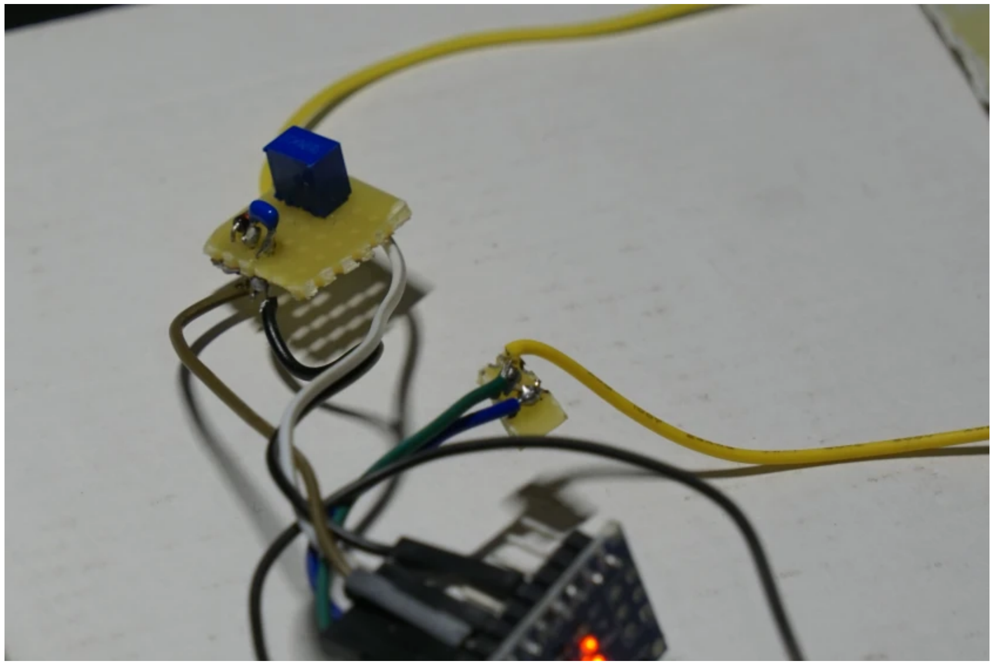
I invented this method a few years ago when looking at a design for a capacitive probe. When researching and looking at the schematics of some other methods I had the idea for this circuit:



In addition to the sample capacitor it requires a resistor/potentiometer and a diode with a low capacitance like the very common 1N4148. The input and the output are connected to digital GPIOs of the Arduino, I used D3 as the clock and D4 as the sensing input in the code below. The way it works is quite the opposite way of the charge transfer method. That is also why in the video the two signals go the opposite way. Here is how it works:

Step 1: apply a positive voltage to 'CLOCK_IN'. The probe is charged quickly through the resistor (the initial sensitivity is adjustable if potentiometer is used) and as soon as it is charged enough the diode will switch on and the current will 'overflow' to the bigger capacitor.

Step 2: set the input low and check the output voltage. If the foltage at the output is not yet 'high' then repeat.



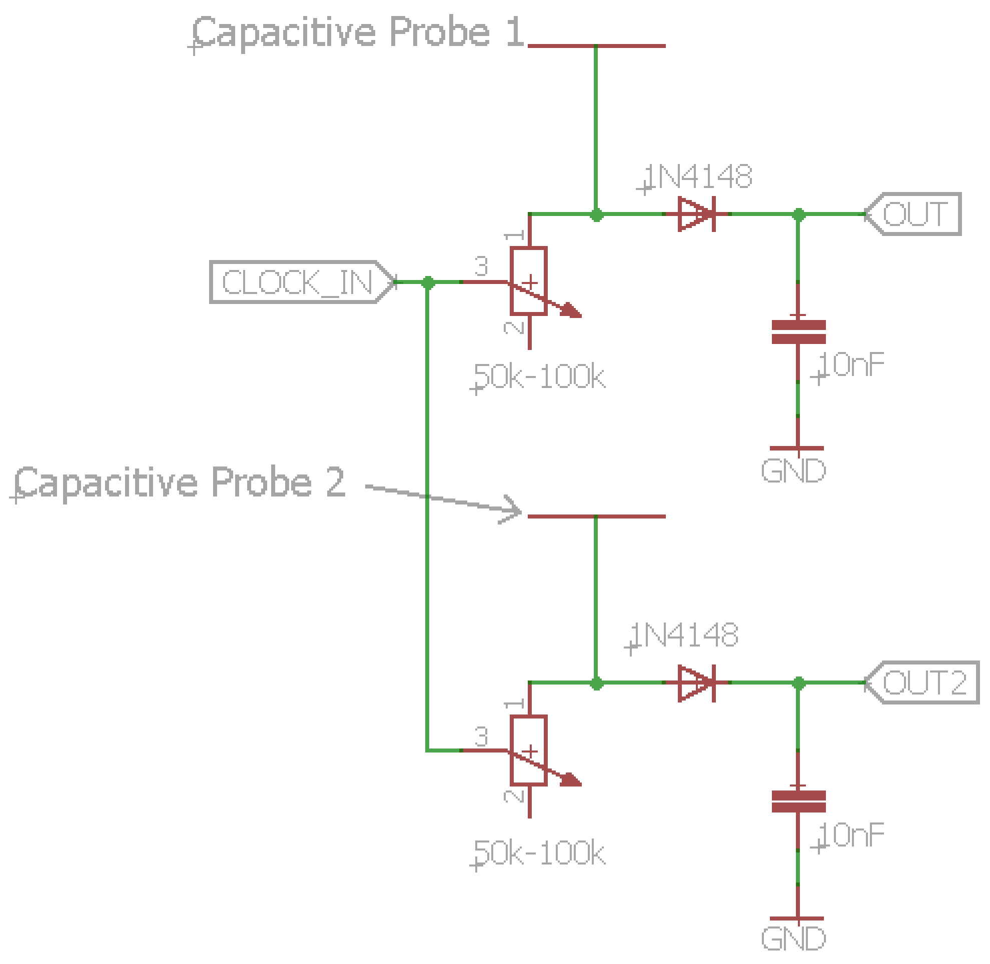
Filter overflow circuit (back) and charge transfer method using a barely visible SMD cap (front)

For a large capacitance on the probe it takes more cycles than for a small one: the probe acts like a low-pass filter on the signal. The trick is to make the clock fast: the faster it is the more sensitive the probe can be made. If the clock is chosen too fast for a large probe then the diode will stop switching on after a few cycles and the output may never reach 'high'. At that point the on-time should be increased. This is done in the code by switching to a longer on-time after counting a few hundred (or thousand) clock cycles so there is no infinite loop. In the code below I use the assembler command ‘nop’ which waits exactly one clock cycle or 62.5ns at 16MHz. More ‘nop’ operations can be added to decrease the clock speed.

The ability to choose the sensitivity of the circuit just by changing the on-time is another advantage over the charge transfer method which is fixed in sensitivity without changing the sample capacitor.

If the resistor/capacitor/clock combination is tuned just right the probe can be made very sensitive: I was able to sense my hand as far as 30cm away from the elctrode by using a 51k resistor, a 1nF sampling capacitor and five ‘nop’ in the code with a 30cm long cable as the probe. The noise picked up at this sensitivity is horrible though.

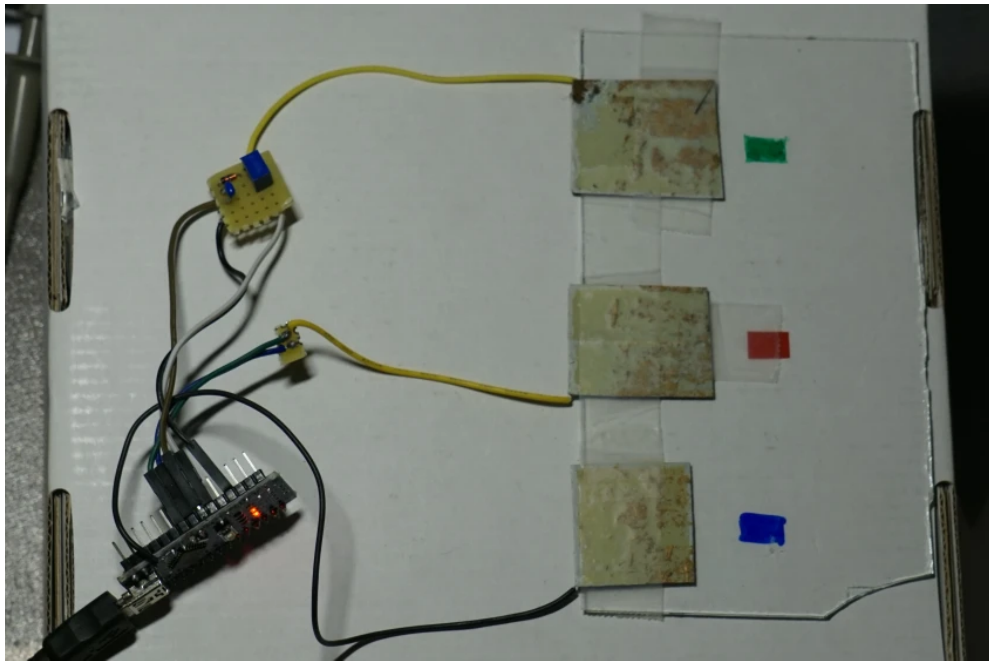
If you want to use multiple probes on a single clock here is how you would wire this up:



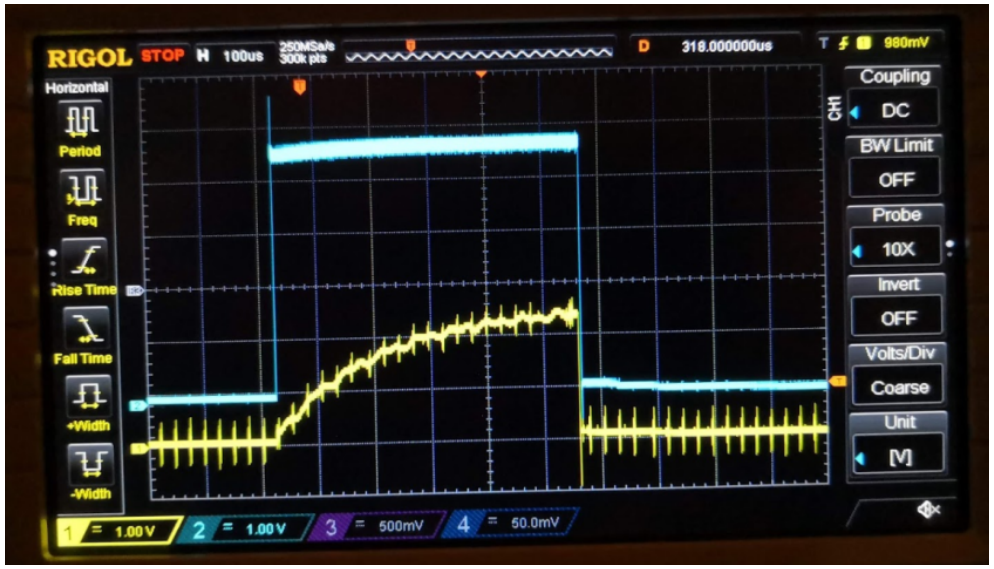
In the code both outputs have to be checked after the clock line is set low of course. This cascading can be continued with more probes. If the circuit is done with SMD components it can still be made very small on a prototyping board.

Comparing the waveforms

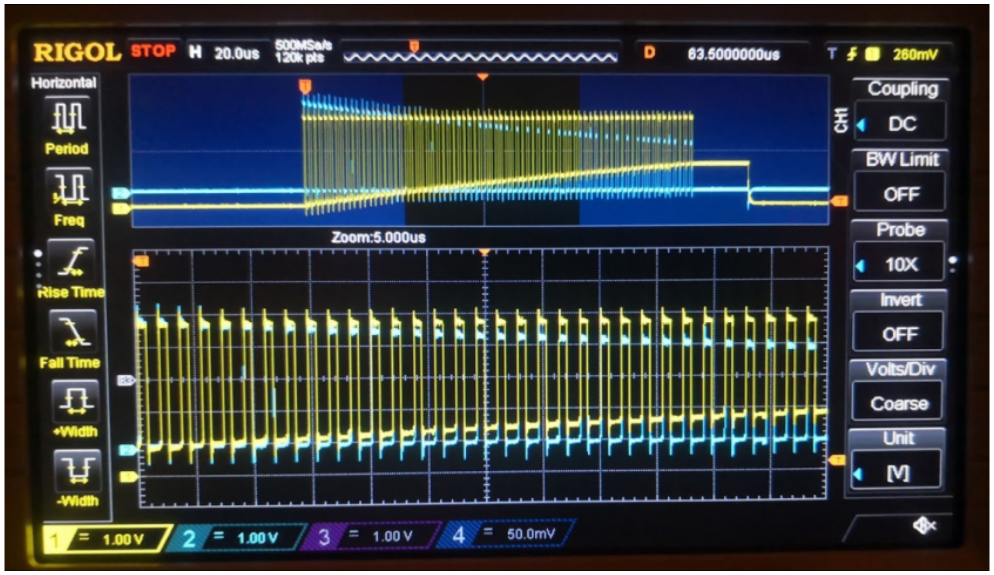
The stuff above explains how these methods work in theory but let’s look at what the voltages actually look like. It may help to understand what is going on in the world of electrons. The photo shows a ‘behind the scenes’ shot of the setup used in the demo video: the copper clads used to be taped to a wooden surface, hence the dirty look on the bottom side. This setup was also used to take the shots below.



First off the resistor method. The blue signal shows the voltage being set to high, then the capacitive probe starts charging through the resistor. Once the pin is charged up, the software takes the time and sets both pins low. The video shows this in action: when I touch the glass plate with my finger the charging time increases as the capacitance of the probe gets bigger but also there is a lot of jitter picked up as my body acts as an antenna bringing the noise close to the circuit. So why is only this circuit so susceptible to this noise? Because of the high resistor value it takes very little AC-currents to cause large voltages on the circuit. The other circuits do not use such large resistors so they can just ‘eat up’ that current without causing much noise.

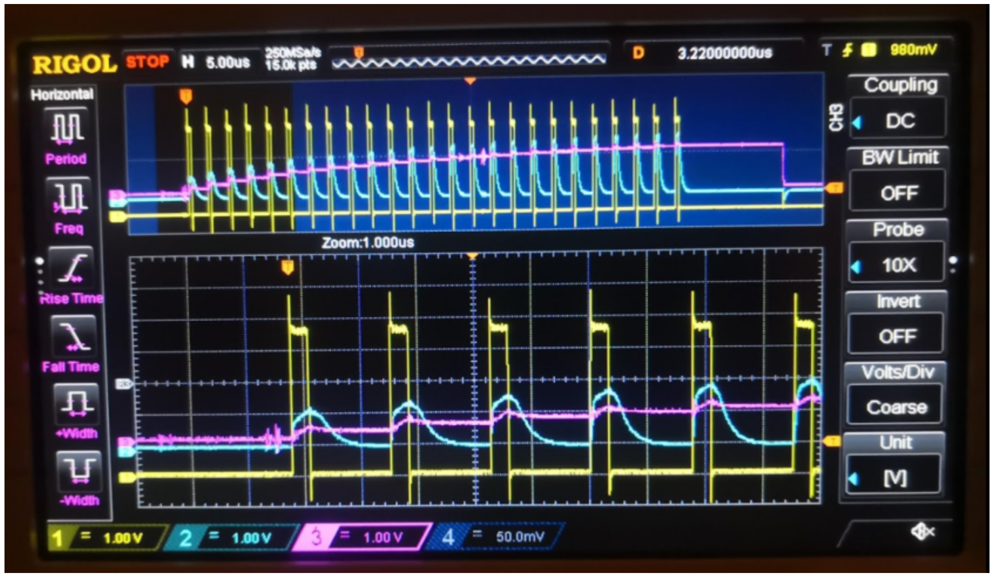

RIGOL
STOP
H 100us
250MSa/s
300k pts
D 318.000000us
T 980mV
Horizontal
Period
Freq
Rise Time
Fall Time
+Width
-Width
CH1
Coupling
DC
BW Limit
OFF
Probe
10X
Invert
OFF
Volts/Div
Coarse
Unit
[V]
1 = 1.00 V
2 = 1.00 V
3 = 500mV
4 = 50.0mV

The charge transfer method has a completely different approach: it uses a clocked circuit and is more ‘digital’ and hence also very stable. The screenshot shows the charging up of the capacitor. Channel1 (yellow) shows the voltage pattern on the GPIO1 pin: the lower end increases with every cycle until the Arduino pin triggers as ‘high’. Channel2 (blue) on the other hand shows the voltage at the probe which is zero when set to low and shows the difference between 5V and the sample capacitor voltage when left floating, so it looks like a mirror image of channel1. Each time channel 1 is set to high the current flows through the sample capacitor into the probe, charging the sample capacitor up by a little bit.



Again, the video shows how it behaves when I touch the glass plate: as the capacitance of the copper plate increases the number of cycles it takes to charge up the sample capacitor decreases.

The third method is probably a bit more complicated to understand. The yellow line shows the clock input. The current through the resistor (or potentiometer) charges up the capacitive probe *very* quickly: you can barely see the blue curve rising more slowly than the yellow line. Once it reaches a voltage higher than the sampling capacitors voltage (channel 3 in purple) the diode starts conducting and charging the sampling capacitor up some more. See how the blue and purple lines start rising in parallel. When the clock is set low the probe is discharged. Also note how the sampling capacitor loses just a tiny bit of charge as well. This is due to leakage currents. If the clock is too fast for the size of the capacitive probe you can imagine that this game can reach an equilibrium state before the sampling capacitor is charged enough to trigger the ‘high’ level at about 2.5V.



In contrast to the ‘charge transfer method’ you can see that the number of clocks it takes when coming close to the glass plate increases. So this method could be viewed as a combination of the resistor method and the charge transfer method as it takes the best from both circuits: it uses only one pin per additional channel (like the resistor method) but has the stability and sensitivity of the charge transfer method.

Arduino Code

I put the example I used to create the video at the top up on git: Arduino Code

I hope you enjoyed this little (or rather lengthy?) lesson about capacitive sensing. If you have any suggestions or questions please leave a comment.

bashtelorofscience / February 8, 2019

One thought on "Advanced capacitive sensing with Arduino"

Thomas

March 10, 2020 at 1:40 am

Wow, very nice!
Thanks a lot. I'm going to try that next on my musical instrument project
https://github.com/ttempe/Pocket_Organ

★ Like