INTRO | mki x es.edu

Hey there, thanks for buying this DIY kit! We – **Erica Synths** and **Moritz Klein** – have developed it with one specific goal in mind: teaching people with little to no prior experience how to design analog synthesizer circuits from scratch. So what you'll find in the box is not simply meant to be soldered together and then disappear in your rack.
Instead, we want to take you through the circuit design process step by step, explaining every choice we've made and how it impacts the finished module. For that, we strongly suggest you follow along using **LABOR[1],** which is a non-permanent circuit prototyping tool that allows you to experiment and play around with your components. To help you with this, we've included suggested breadboard layouts in select chapters.
In addition to this, you can also play around with some of the chapter's circuits in a **circuit simulator** called CircuitJS. CircuitJS runs in your browser. You'll find weblinks in the footnotes which will direct you to an instance that already has example circuits set up for you. We strongly encourage you to fiddle with the component values and general structure of those circuits to get a better understanding of the concepts we're laying out.
Generally, this manual is intended to be read and worked through front to back, but there were a few things we felt should go into a dedicated appendix. These are general vignettes on electronic components & concepts, tools, and the process of putting the module together once you're done experimenting. Don't hesitate to check in there whenever you think you're missing an important piece of information. Most importantly though: have fun!

TABLE OF CONTENTS

[1] You can also use a standard breadboard, but this will require you to get a little creative when adapting the suggested layouts. You'll also need to do some additional engineering to get the different supply voltages.

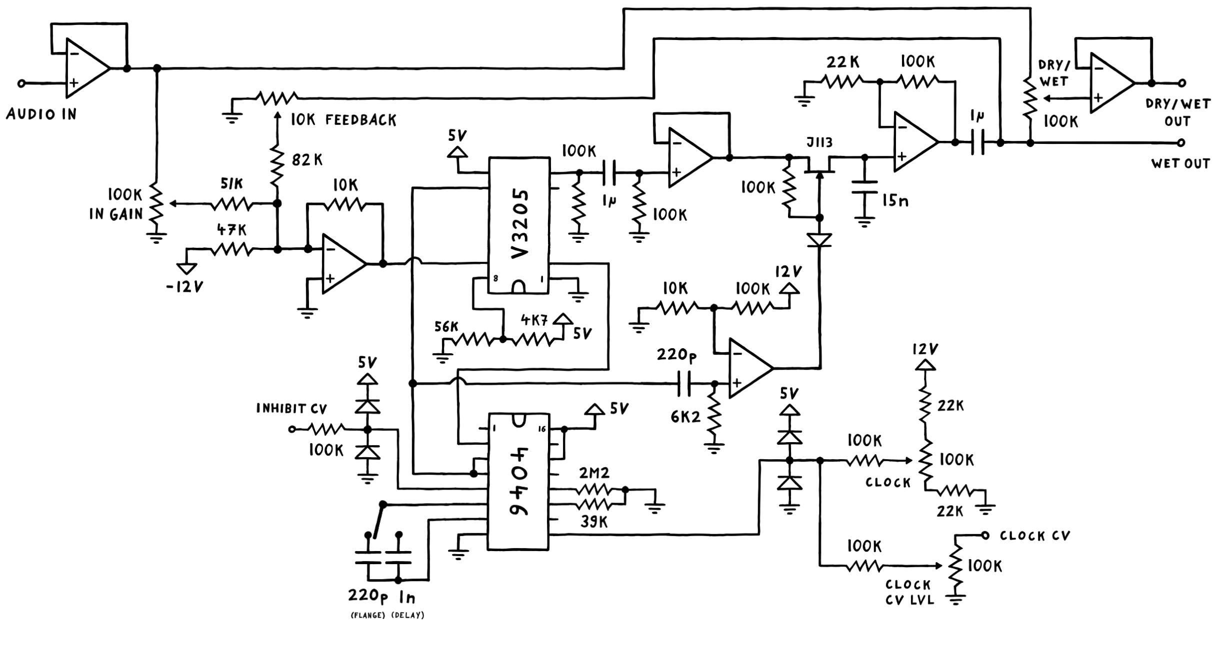
THE mki x es.edu BBD

Here's an interesting problem: how do you create a delayed duplicate of a sound without recording it to some sort of storage medium? Back in the days before digital signal processing and cheap, abundant memory, this was a prime engineering issue. That's because at that time, the only practical electronic implementation of an audio delay effect was large, bulky tape echo machines.

These machines worked by recording an input signal to electromagnetic tape and then playing it back with a time delta. Of course you could always decide to go no-tech and use a wide open space with a reflective surface in the distance to create an echo. This works because it takes time for sound to travel through a medium such as air.

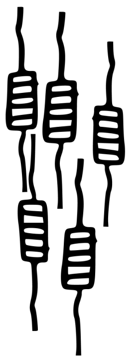
In the late sixties, two electrical engineers named Sangster and Teer managed to implement an idea that was essentially a unique hybrid of the two concepts I just mentioned: the bucket brigade delay. This circuit first splits an audio signal into analog samples, which are then routed sequentially through a chain of transistors and capacitors – similar to how sound travels through a medium.

At the output, the signal is reconstructed from those samples. **And because passing the samples along the transistor-capacitor chain takes time, the reconstructed signal will be delayed compared to the input, with the length of the chain determining the total delay time.** Over the course of this manual, we will reverse engineer the architecture of a classic bucket brigade delay, recreate a bare bones version on the breadboard – and then use a proper BBD chip to implement a simple audio delay effect.



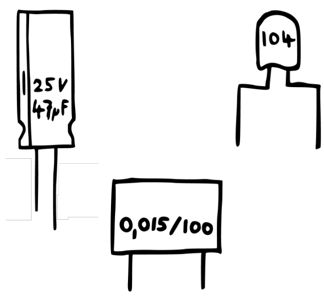
BILL OF MATERIALS

Before we start, please check if your kit contains all of the necessary components. In addition to a PCB, panel and power cable, your box should also contain:



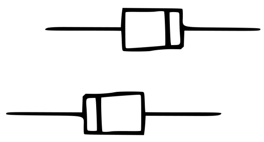
An array of resistors. The specific values (in ohms, which you should check for with a multimeter) are

2M2	x1
100k	x9
82k	x1
62k	x1
51k	x1
47k	x1
39k	x1
22k	x3
10k	x2
6k2	x1
4k7	x1
1k	x1
470	x2
10	x2



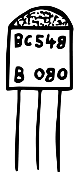
A bunch of capacitors. The specific values (which are printed onto their bodies) are

47 uF	x2
3.3 uF	x2
1 uF	x2
15 nF	x1
100 nF	x12
1 nF	x1
220 pF	x2



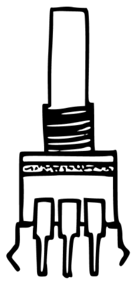
Some diodes. The specific model names (which are printed onto their bodies) are

1N4148 (signal) x5
1N5819 (schottky) x2



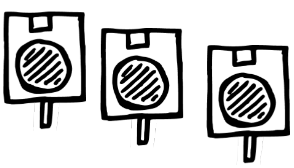
A couple transistors. The specific model names (which are printed onto their bodies) is

J113 (N-CH JFET) x1
78L05 (5V regulator) x1



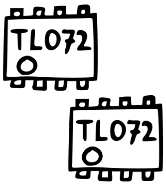
A handful of potentiometers. Their specific values (which may be encoded & printed onto their bodies) are

100k (A104) x1
100k (B104) x3
10k (B103) x1



A few jack sockets. The specific models (which you can identify by their color) are

Switched mono (black) x5



A couple chips. Their specific models (which are printed onto their bodies) are

TL072 (dual op amp)	x3
V3205SD (BBF 4096)	x1
CD4046BE (PLL)	x1

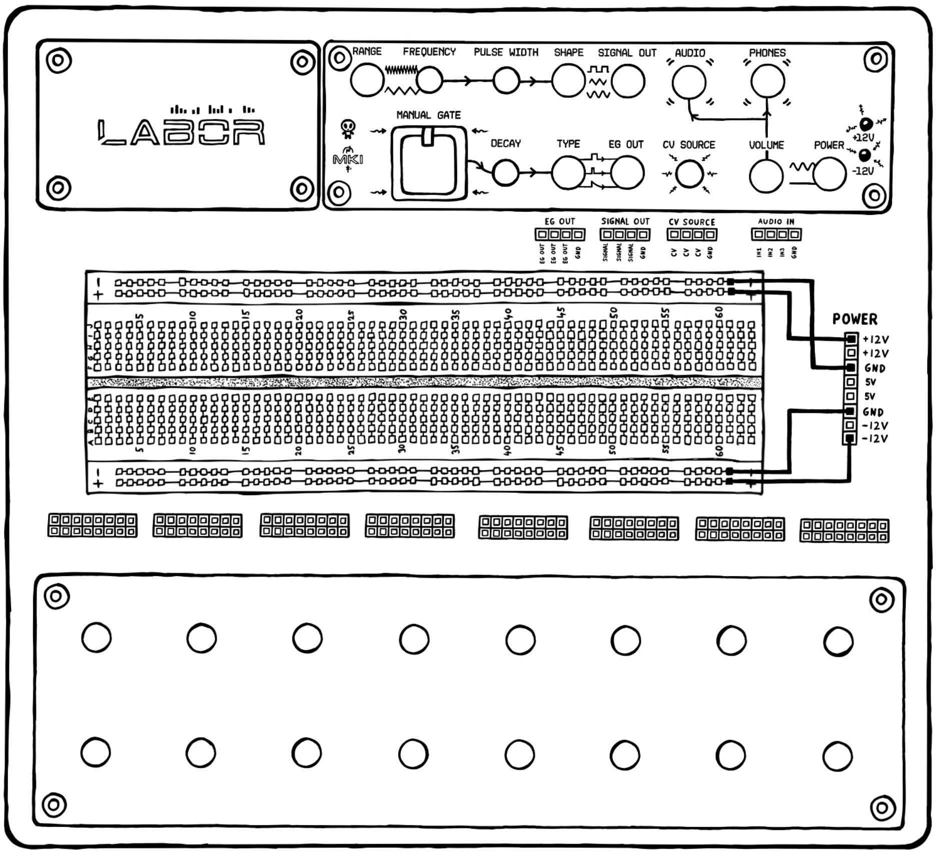
A switch. The specific model is

Single pole, double throw	x1

You will also find a few sockets that are only relevant when assembling the module in the end.

USAGE WITH MKI x ES LABOR

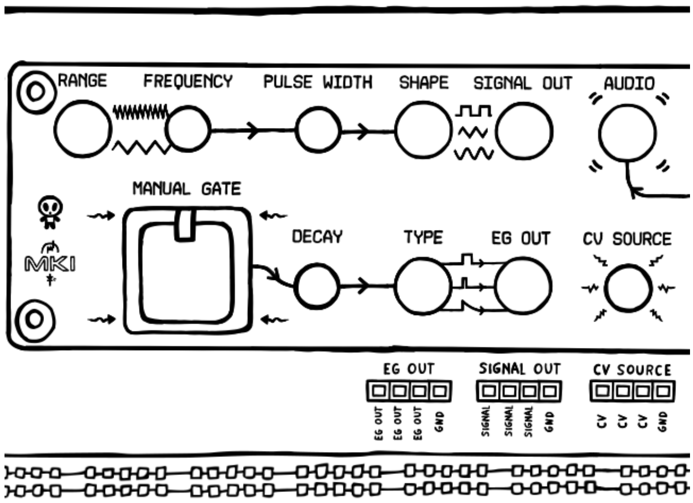
We recommend that you follow this guide using an **MKI x ES LABOR** prototyping board. **LABOR** comes equipped with everything you need for testing the circuits we lay out: a standard 830 tie point breadboard, an integrated dual power supply with over current protection, a manual gate/trigger/envelope generator, an LFO, a variable CV source, an output amplifier, and a modular interfacing section where you can insert all of your interfacing components like potentiometers, jack sockets, and switches.



Before you get started, connect the slots labeled **GND** on the power header to both breadboard rails labeled **–** (minus). Next, connect one slot labeled **+12 V** to the top breadboard rail labeled **+** (plus), and one slot labeled **–12 V** to the bottom breadboard rail labeled **+** (plus).

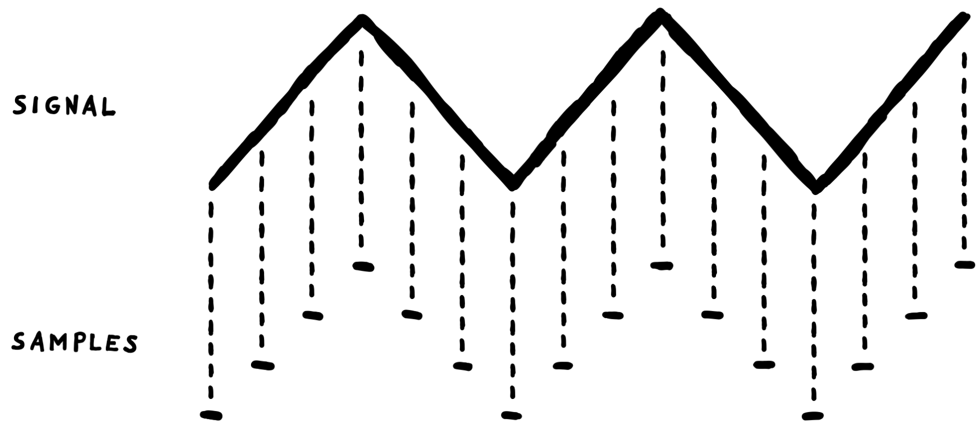
To listen to your circuit, you don't even need to set up an output jack socket. Instead, use the built-in output amplifier at the top of the device. Just plug your circuit's signal output into the header labeled **AUDIO IN**, and then connect your headphones to the **PHONES** output jack (or a line-level device like a standalone external speaker to the **AUDIO** output jack).

Sometimes, this guide will ask you to use external gear like sequencers or LFOs to send CV, audio signals, triggers or gates into your circuit. With **Labor**, there's no need for extra equipment – just use the built-in oscillator (audio/LFO), CV source or manual gate/trigger/envelope generator. You can grab all of those via the headers labeled **EG OUT**, **SIGNAL OUT**, and **CV SOURCE** and connect them to the designated points on the breadboard.

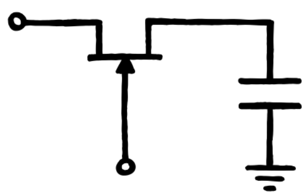


ANALOG SAMPLING

We'll start by investigating the first stage of a BBD: the analog sampling sub-circuit, which splits the input into transmissible chunks. Okay, but what does that mean, exactly? Well, in electronic circuits, an audio signal is represented as a swinging voltage – like this triangle wave.

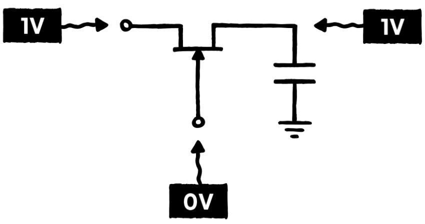


To extract a sample from this means that we probe the signal at any point and store whatever value we read. If we do this continuously and in regular intervals, we get a series of samples that resembles the input. **But unlike a swinging voltage, those samples are discrete and can be easily held in a capacitor, allowing us to control when and how they move through the rest of our circuit**.

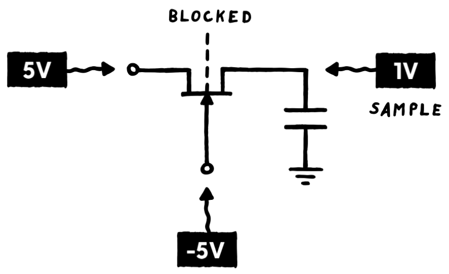


Alright, so how do we extract a sample from a signal? All we need are two components: a field effect transistor (FET for short), which will act as a probe, and a capacitor, which will act as a storage medium. The transistor has to be a FET, because unlike bipolar junction transistors, FETs allow for bidirectional switching.

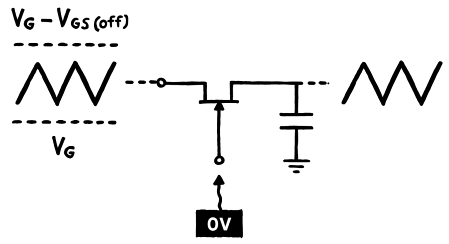
This is important because depending on the state of the input waveform when we probe it, current might need to flow into or out of our sampling capacitor. But while BBD chips normally use MOSFETs, we'll use a JFET as a substitute, since discrete, standalone MOSFETs are built in a way that disqualifies them for bidirectional switching.



To understand how the sampling process works, let's first assume that our input is a static 1 V signal. Then to sample that signal, we activate the JFET by bringing its gate voltage close to the source voltage. This allows current to flow into the capacitor until the voltage above it is 1 V as well.



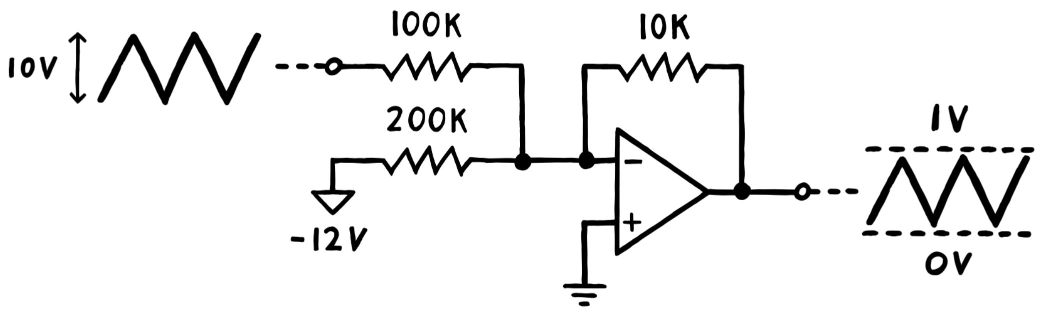
Next, we pull the gate voltage way below the source, disabling the JFET. This cuts the connection between signal and capacitor, which means that the charge is now trapped inside the cap. **This trapped charge is our sample**.



Okay, but what about a non-static signal? Luckily, it works just the same: we activate the transistor by raising the gate voltage, which causes the voltage above the capacitor to follow the input. Then, we pull the gate voltage low to trap a sample inside the cap. This works because while the transistor is open, charge can freely flow into and out of the capacitor as the input signal oscillates. At least if it stays within these limits – which are determined by the gate voltage we use to activate the transistor and its gate-to-source threshold.

If the input crosses the upper line, the transistor will close, since the gate is too negative compared to the source. And if it crosses the lower line, current will flow into the gate, potentially damaging the transistor. This is an issue because audio signals normally swing way outside these limits. So if we want to test our sampling circuit in practice, we'll need to scale and bias whatever input we want to feed it.

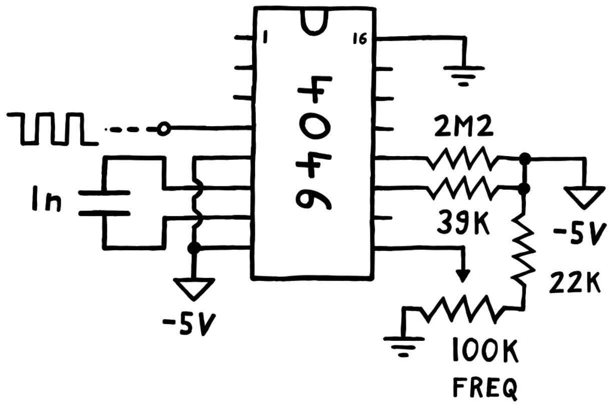
For that, we can use a simple inverting op amp, which we'll give both the input signal and an offset voltage. By balancing the input resistances against the feedback resistor, we can determine the exact amount of scaling and biasing we want to apply.



Since I plan to use audio from a eurorack synthesizer as our input, I assume that input to be a 10 V peak to peak signal that swings across the 0 V line. In that case, these resistor values will give us an approximately 1 V peak to peak result that is swinging just above the 0 V line.

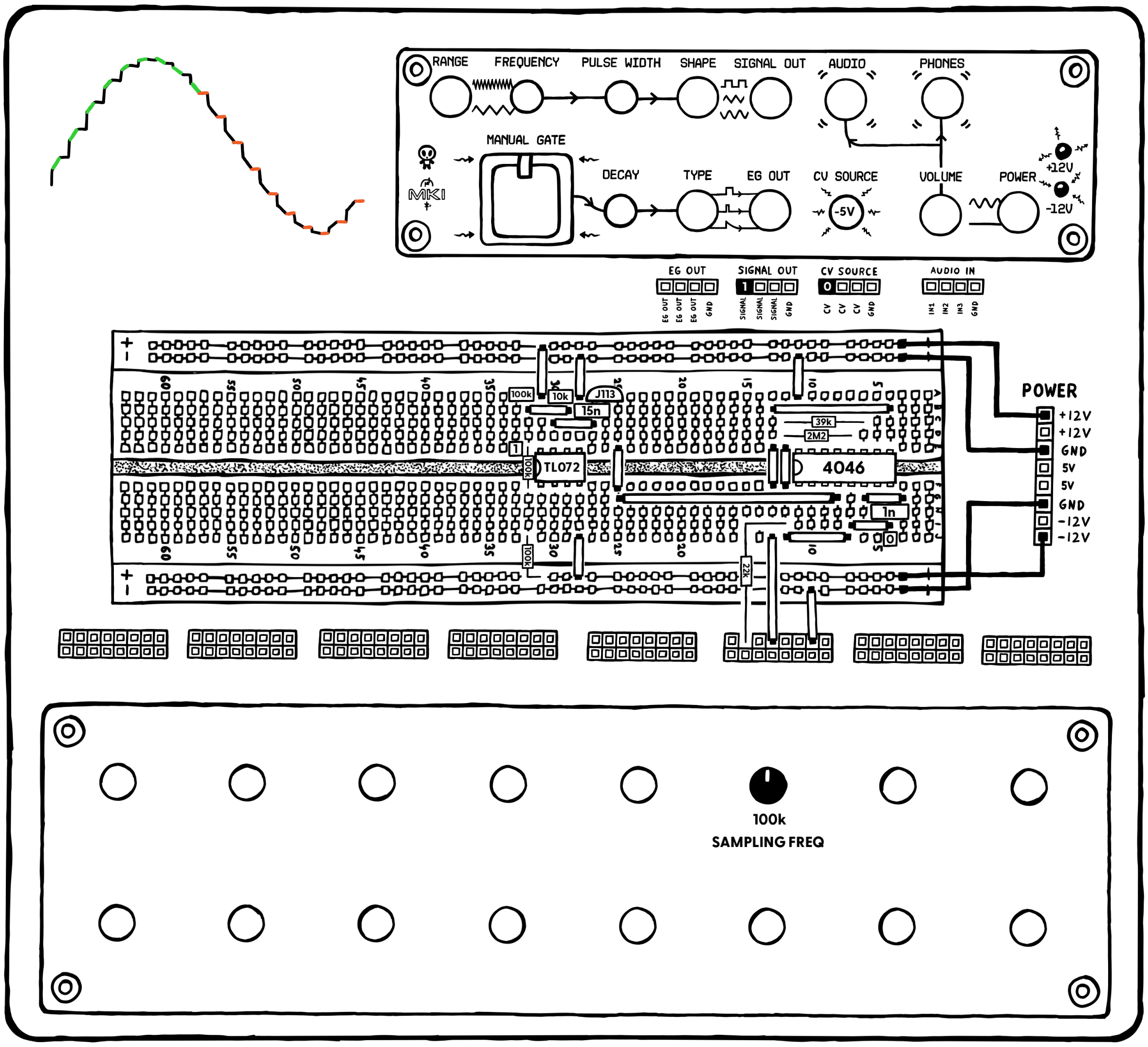
Okay, so with our input signal sorted, the next step is figuring out how to control when each sample is captured. As I said before, this needs to happen continuously and in regular intervals, so that all of the samples taken together resemble the input. Frequency is also a consideration here, since it will determine the sampling resolution. **And the higher the sampling resolution, the more accurately the samples will resemble the**

input. So we'll want to automate the sampling process with a fast clock signal – essentially just a high frequency square wave swinging between 0 and -5 V.
And while we could generate this clock with any basic square wave oscillator, I'm going to use a 4046 Phase Locked Loop chip instead. That's because the 4046 contains a full blown VCO, which is an oscillator that allows us to control its frequency with a voltage. (This will become important later.)



To use the VCO inside the chip, we first need to supply it with power. Since we want our clock to oscillate between 0 and -5 V, we'll give it 0 V as a high and -5 V as a low supply voltage. Next, we'll set the oscillator's frequency range. If we combine a 39k resistor to the negative supply at pin 11 with a 1 nF capacitor between pins 6 and 7, the VCO can produce clocks between 0 Hz and around 27 kHz.
Because it doesn't really make sense to have the frequency go down to 0, we'll add a frequency offset of around 800 Hz by connecting pin 12 to the negative rail via a big 2M2 resistor. Next, we'll need to apply a control voltage to pin 9. For now, we'll simply set up a 100k potentiometer as a variable voltage divider between -5 and 0 V.
Since the 4046 has a CV input dead zone of about 1 V above the lower supply, we'll also insert a 22k resistor between the pot and -5 V, limiting the divider's minimum output to around -4 V. Finally, we'll start the oscillator by tying pin 5 to the low level supply.[2]

[2] You can try this chapter's circuits in a simulator. I've already set them up for you right here. You can change all values by double clicking on components.

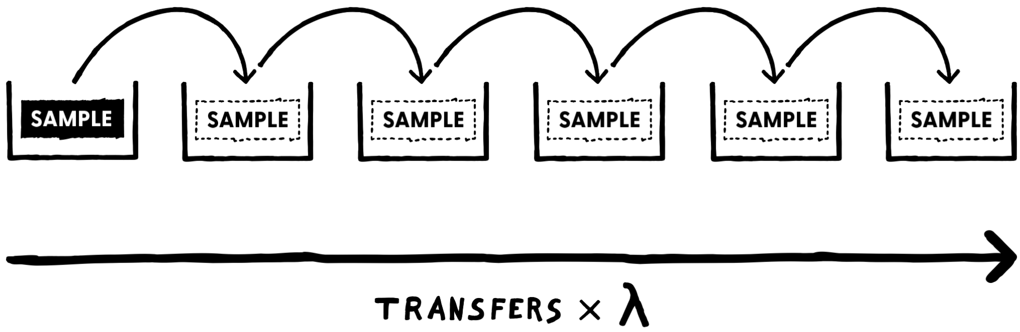


Okay. To start off, we're feeding LABOR's sine wave oscillator into our scaling and biasing sub-circuit (connection path **1**), before routing it to a J113 JFET followed by a 15 nF capacitor. Then, we connect the clock generator, which receives its -5 V supply from LABOR's variable voltage source (connection path **0**), to the JFET's gate. If you look at the output using an oscilloscope, you can see that the capacitor voltage looks like a sine wave – but with strange jagged edges. What's up with that?

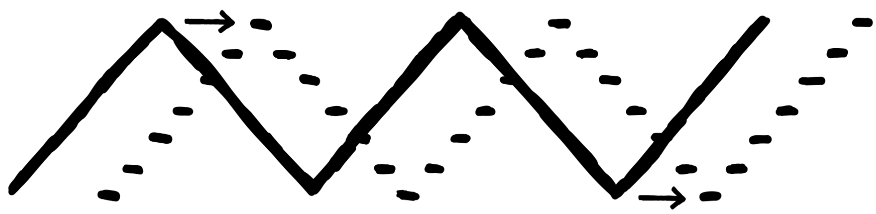
Simple: when the transistor is active, the cap voltage follows the input voltage – that's why those parts are rounded. **This is the sampling phase – marked green above.** And when the transistor is disabled, the cap voltage stays locked to the last value, resulting in visible stair steps. **These are our samples – marked orange above.** Now, if you increase the clock frequency, you can see how those jagged edges become smaller and smaller. That's because we're increasing the resolution of our sampling process, giving us an output that resembles the input sine wave much more closely.

SAMPLE TRANSFER

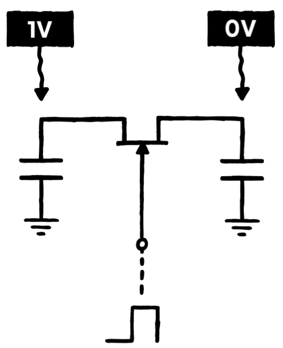
Great! Now that we've split our signal into samples, we can think about achieving the delay effect itself. The trick here is to take the individual samples and move them through a chain of containers, synchronized with the clock signal.



Since each transfer from one container to the next takes a small amount of time – one clock wavecycle, to be precise – the samples will reach the end of the chain with a time delay relative to the input. That time delay depends on the number of transfers and the frequency of the clock. If we then reconstruct the waveform from those delayed samples, we should get an output wave that's been shifted forwards on the x-axis (or in time).

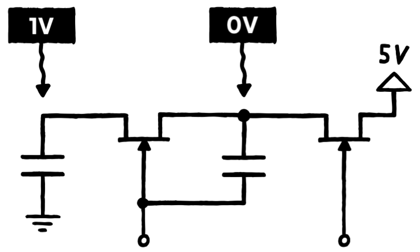


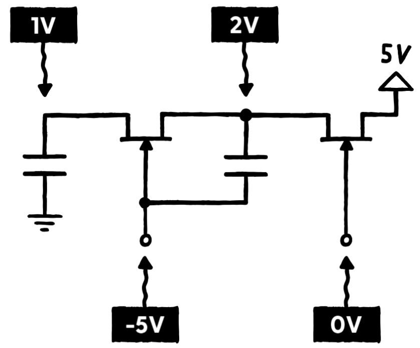
Alright, but how exactly do you transfer a sample? Well, in principle, if you have one capacitor that holds a sample on the left, and an identical (but empty) capacitor on the right, you just need to move the charge from cap one to cap two in response to a clock pulse.

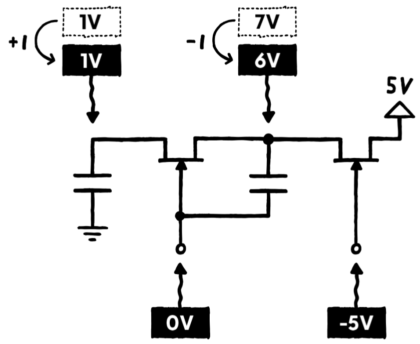


You might assume that we can do this with another JFET – plug it in between the two caps, connect our clock to the gate, and then the JFET should allow current to flow from cap 1 to cap 2 when the clock goes high. The problem with this idea is that we need all of the charge to move – but only half of it will. Because then, the voltages on both sides of the transistor will be equal, causing the current flow to stop. This is the exact issue that Sangster and Teer were faced with when they first developed the bucket brigade delay in the late 60s.

The solution that they came up with is simple – but not obvious at all. **Instead of transferring charge between capacitors, they decided to transfer the charge deficit.**





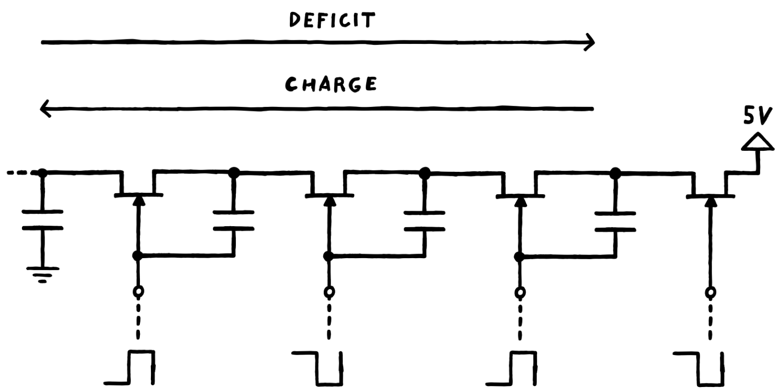


It works like this. We first add another JFET, which we connect to capacitor two on the left and a 5 V voltage source on the right. Then, we connect the other side of capacitor two to the gate of the first JFET (instead of connecting it to ground).

Next, we'll open JFET two by grounding its gate while keeping JFET one closed with a low gate voltage. This allows current to flow into capacitor two until we hit the JFET's gate-source threshold. Assuming that threshold is -2 V, the current flow will stop once we read 2 V above the cap. Keep in mind that this is relative to ground – if we measured the voltage across the cap, we'd see 7 V instead.

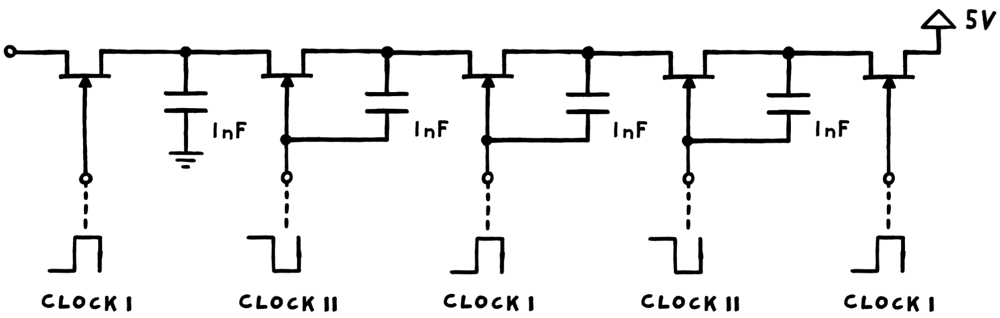
Next, we close JFET two by pulling its gate voltage low, and we open JFET one by pushing its gate voltage up. This also applies force to cap two, causing current to flow from right to left until cap one hits the gate-source threshold at 2 V. In this process, cap two loses exactly as much charge as cap one was missing relative to the 2 V maximum: 1 V in this specific case. **This is why Sangster and Teer call this technique a charge deficit transfer: the deficit has moved from cap one (which is now full) to cap two (which is now missing 1 V)**. And because we give cap two a strong push from below via the preceding JFET's gate voltage, this will work even if cap one starts out completely empty.

Of course this doesn't just work for a single transfer. We can add an arbitrary amount of JFETs and capacitors all set up the same way to increase the number of transfers. Charge will flow from right to left in just the same way, while the deficit travels from left to right until it reaches the end of the chain.



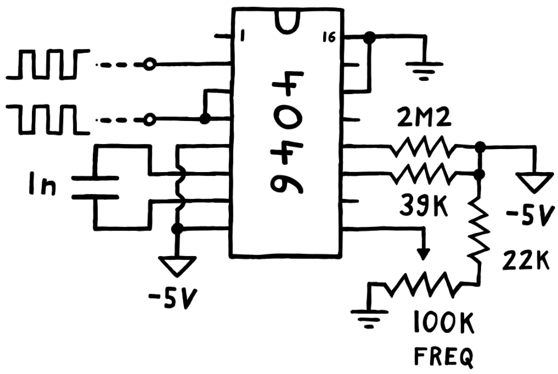
Note that to automate this circuit's operation, we'll need a second clock signal that is the exact inverse of our existing clock to drive every other JFET. Also note that each cap is only carrying information on every other clock pulse – it is either full (which means it's not carrying information) or at a charge deficit (which means that it is carrying information).

Next, we'll want to combine the charge deficit transfer idea with our previously established sampling stage. All we have to do there is add the sampling JFET and capacitor back in at the beginning of the chain and connect the JFET's gate to clock one.

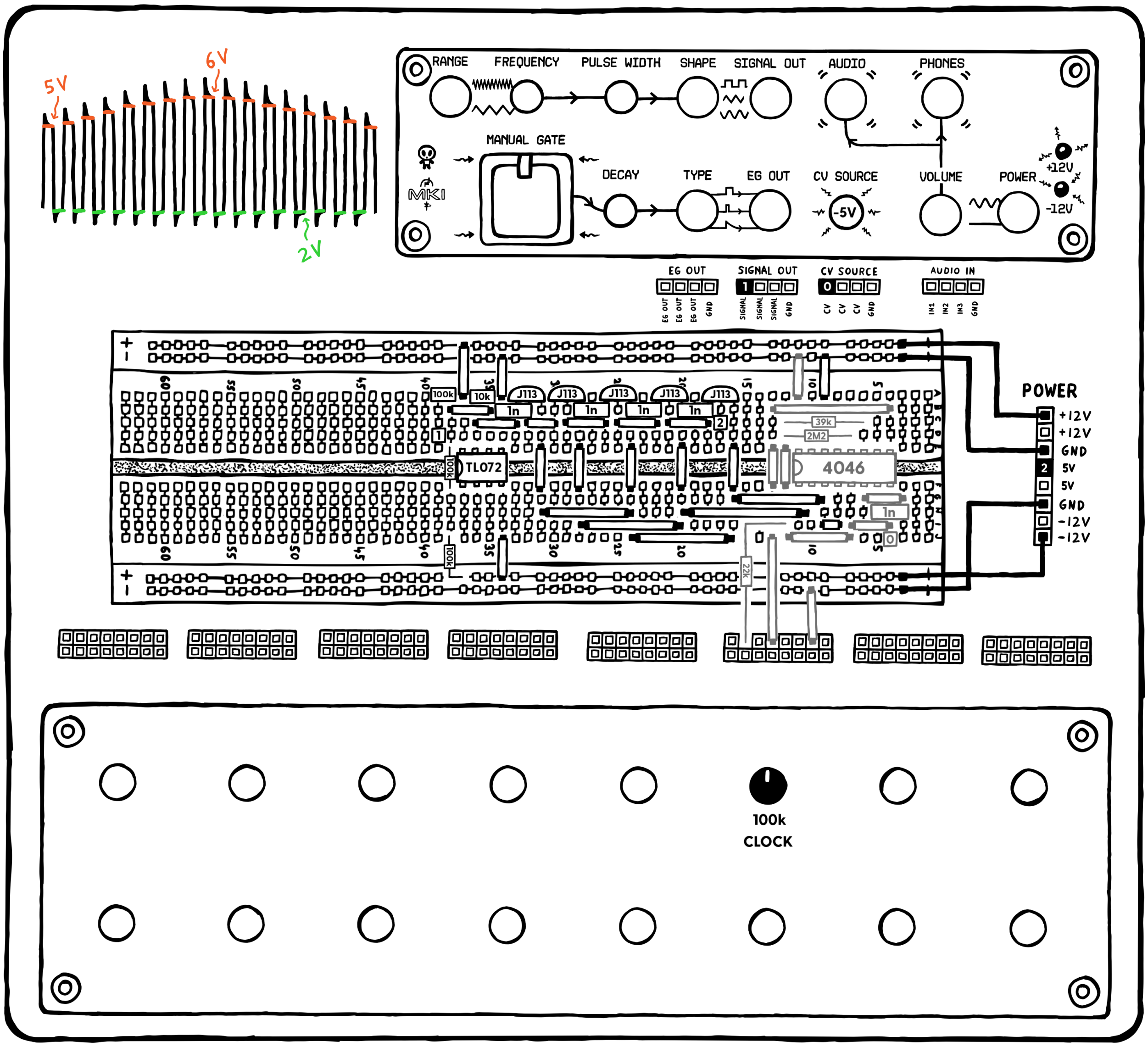


Then during the sampling phase (when clock one is high and clock two is low) JFET two will be closed, isolating the sampling stage and allowing it to operate just as we discussed earlier. And when the clocks flip, JFET one will be closed, kicking off the deficit transfer process.

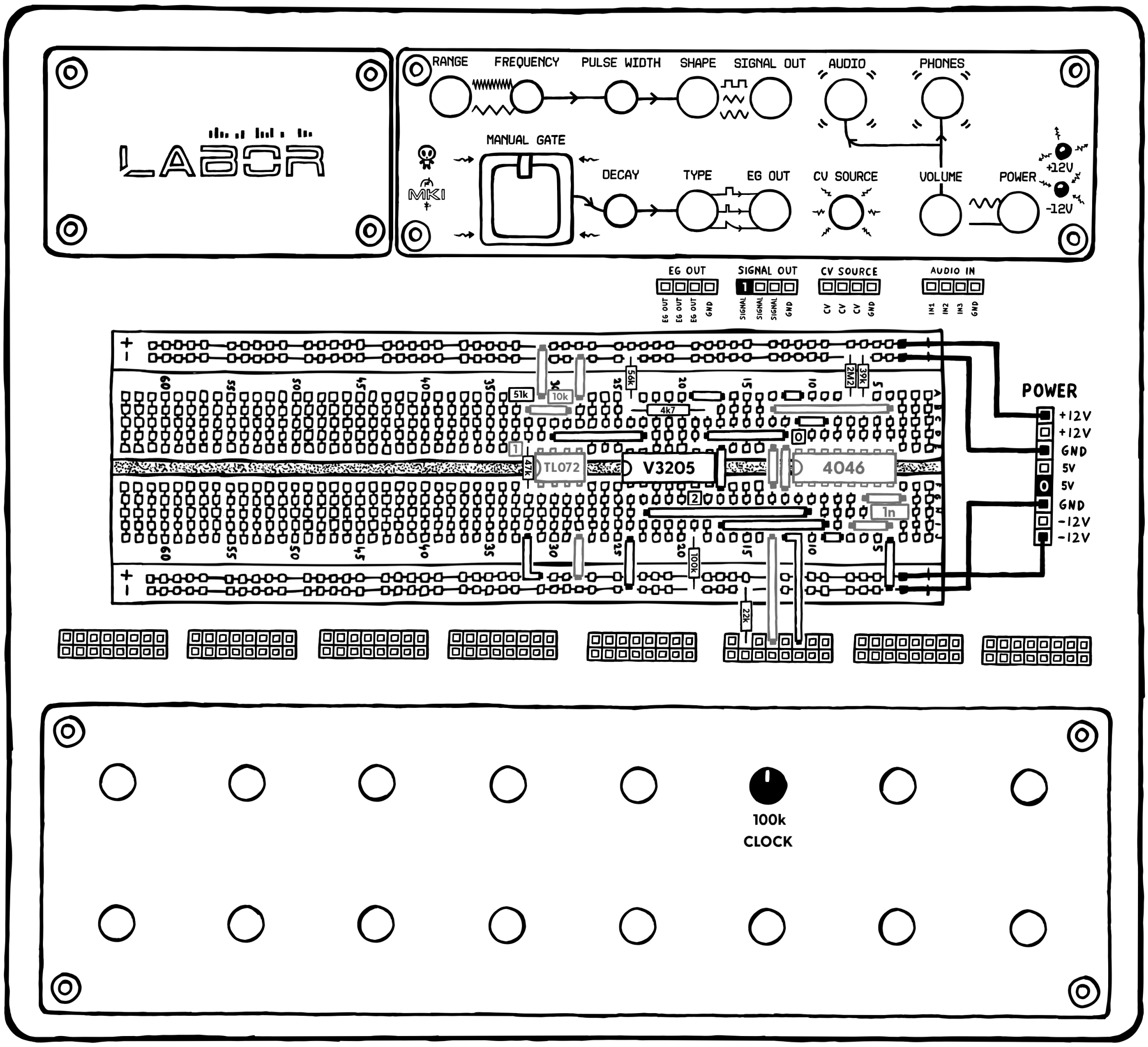
Before we can try this, we'll need to modify our clock circuit so that it generates an additional inverted clock. Thankfully, we don't have to bring in any extra components for this. **Because if we tie pin 14 on the 4046 chip to the high level supply, pins 2 and 3 act as a simple inverter, with pin 3 as the input and pin 2 as the output**. So if we connect the original clock to pin 3, we can pick up our inverted clock from pin 2.[3]



[3] You can try this chapter's circuit in a simulator. I've already set it up for you right here. You can change all values by double clicking on components.



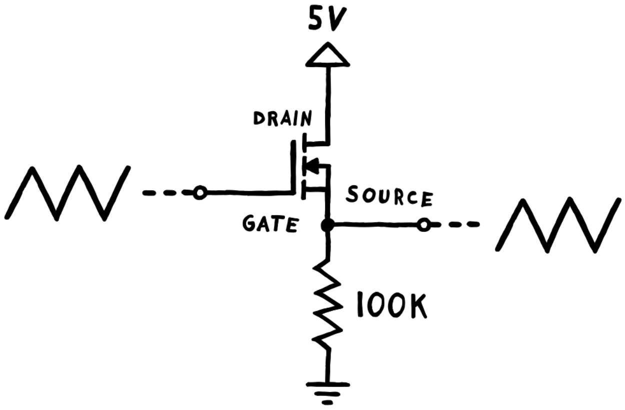
To try this, we'll add 3 BBD stages to our existing circuit. For the input, we'll use the same scaled and biased sine wave from before. **Please note that some of the components used in this layout are not included in your kit. If you don't have these extra components in your stash, build the alternate layout below that uses kit components exclusively.**



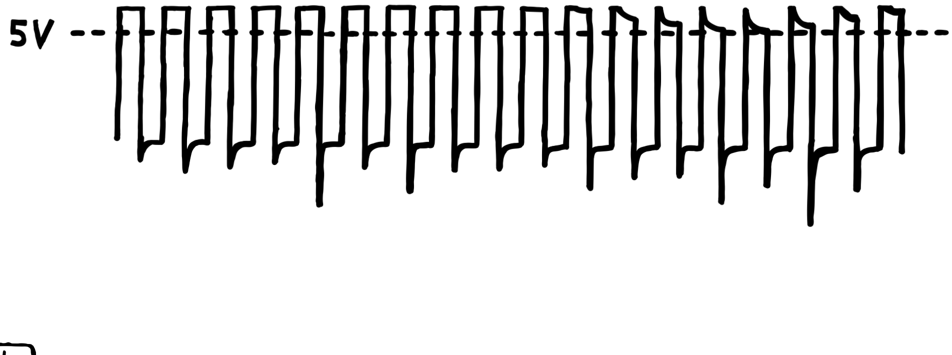
In this alternate layout, we use the dedicated BBD chip included in your kit (V3205). We'll look at its anatomy more in detail later, but this circuit will produce an output very similar to the above circuit. Check out the signal at connection point **2** with your oscilloscope. That voltage should alternate between the full state (2 V relative to ground), which carries no information, and the deficit state, which is moving up and down in a sinusoidal pattern. This is our sampled and delayed input.

MOSFET BUFFER

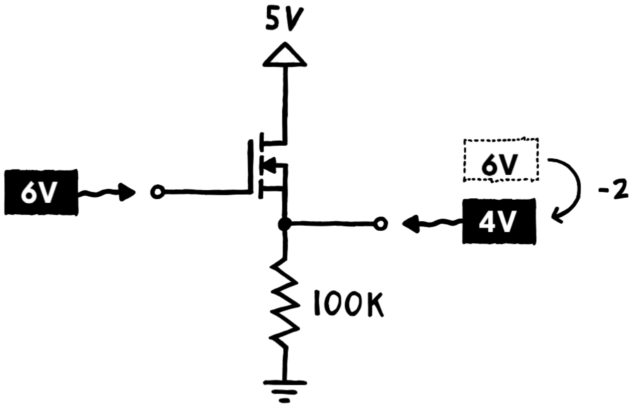
Now, in order to hear how it sounds, we need to add a voltage buffer to avoid drawing current from the BBD stage capacitor and distorting the signal. And while we could use an op amp here, BBD chips typically use a MOSFET for a smaller footprint. Lucky for us, we can actually do the same! **That's because voltage buffering is a unidirectional use case, which works just fine with a discrete MOSFET**.



The core mechanism is very much comparable to a regular BJT emitter follower. The signal we apply to the MOSFET's gate is reproduced above the source resistor due to negative feedback. There are two important differences, though. First, a MOSFET's gate terminal is isolated from the rest of the device – meaning that it doesn't draw constant current from the signal we're trying to replicate. And second, a MOSFET's activation threshold can be much higher than that of a standard BJT.
Both of these things are great in our case because one, it's important that we do not draw current from our BBD stage capacitor. **And two, a high activation threshold means that we can reproduce the output signal with the same 5 V supply voltage we use to fill up the capacitors**. The latter needs a little bit of an explanation. If we take a look at our current output signal, you'll notice that the deficit states are all above the 5 V line. If we tried to buffer this with a standard BJT emitter follower and a 5 V supply, everything above around 5.6 V would be cut off.

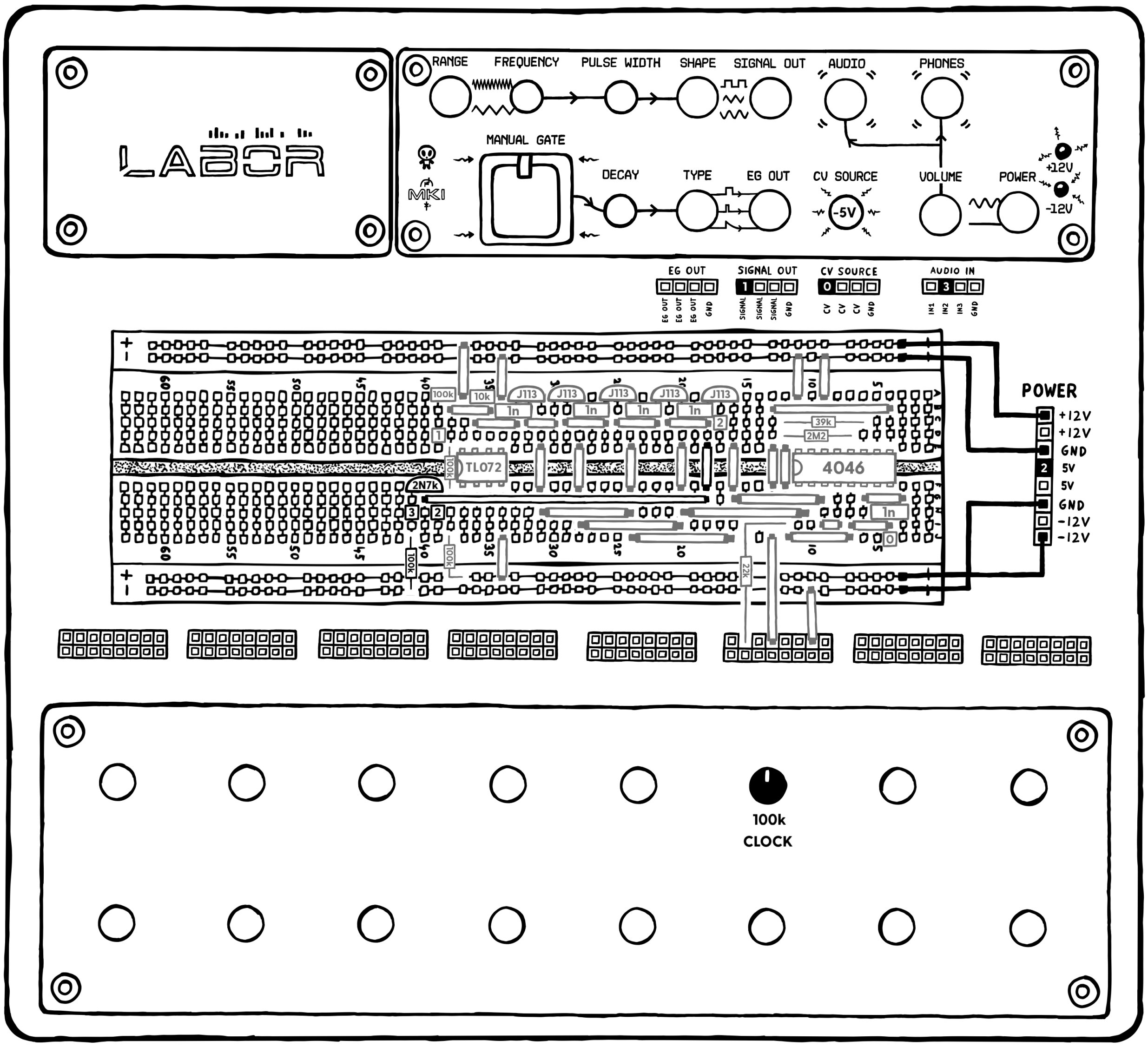


That's because the activation threshold for a BJT is around 600 mV – **which means that the voltage drop between base and emitter is 600 mV as well, shifting the output downwards by that amount and allowing us to buffer an input that's going slightly above our 5 V supply voltage**. With a MOSFET, that threshold can be considerably higher, depending on the model. And a higher threshold means that the output will be shifted downwards quite a bit more, giving us considerably more headroom with the same 5 V supply.



Granted, this is not super important on the breadboard – we have a 12 V supply available after all –, but for a BBD chip, it would be pretty inconvenient to require two separate supply voltages just so we can buffer the output.[4]

[4] You can try this chapter's circuit in a simulator. I've already set it up for you right here. You can change all values by double clicking on components.



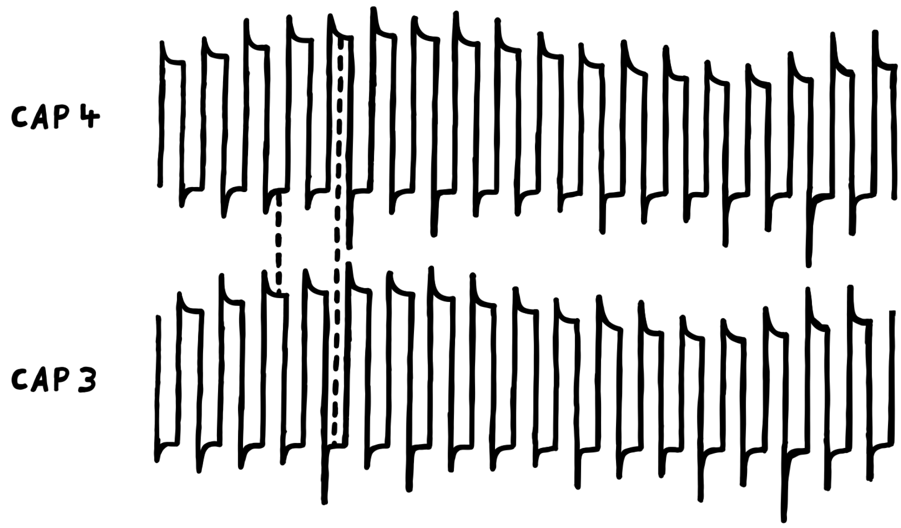
To test this, we'll set up a 2N7000 MOSFET as a voltage buffer. **Please note that some of the components used in this layout are not included in your kit. If you don't have these extra components in your stash, use the previous alternate layout and use connection point 2 as the circuit's output.**

If you check the output on your oscilloscope, it should be replicating our chain's output just fine. You can listen to it by routing it through LABOR's output amplifier. Interestingly, you should hear both the sine wave and the clock in the output – with the latter being significantly louder in the mix.

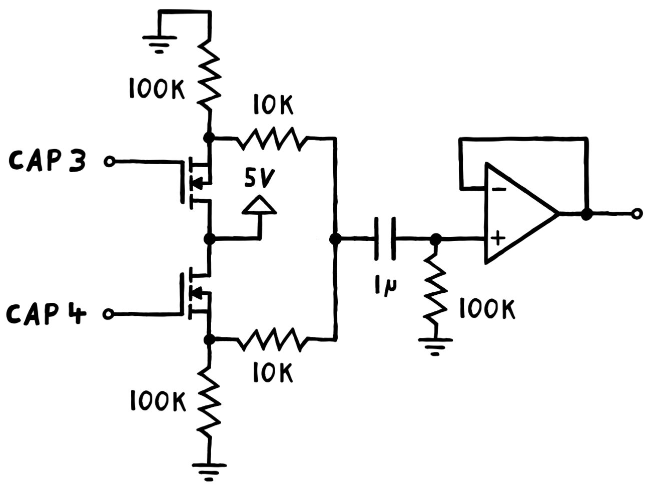
This makes sense, unfortunately: our signal is oscillating between the full state and the deficit state at the exact rate of the clock signal. Which is why that clock signal is so prominent in the output. Bummer – how do we fix this? Lucky for us, there is a classic BBD trick we can apply: dual tap signal reconstruction.

DUAL TAP RECONSTRUCTION

If we compare the signal at capacitor 4 to the signal at capacitor 3, you'll notice that they're complementary in the sense that whenever cap 4 carries no information, cap 3 does – and vice versa.

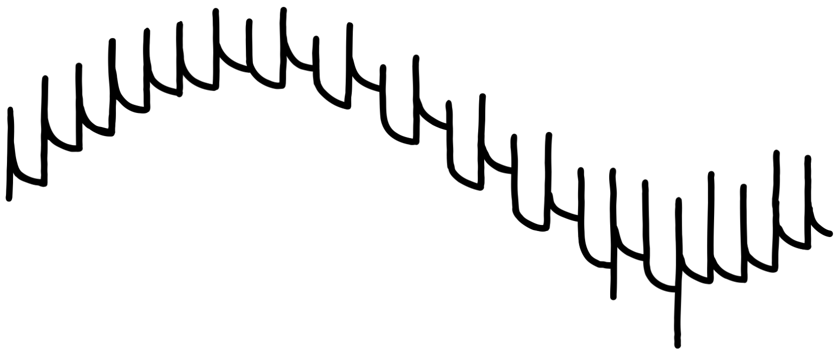


If we were to combine the two, we should be able to cancel out the clock oscillation, since the signal wouldn't jump between full and deficit state anymore. To test this, we'll simply set up another MOSFET buffer for cap 3 and then mix its output with the previous buffer using a simple passive mixer made of two 10k resistors.

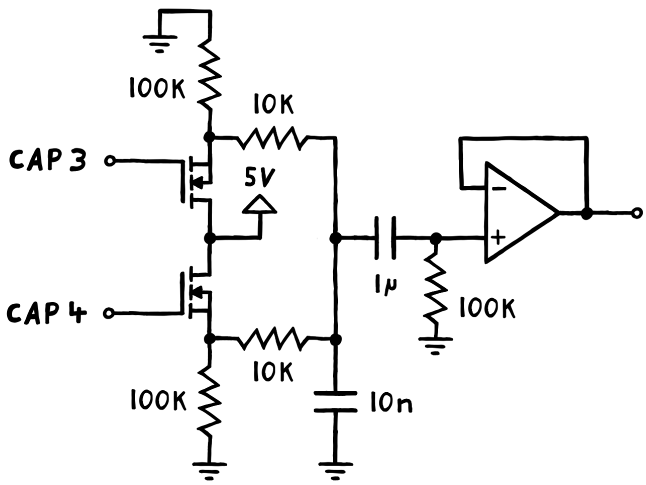


Since the reconstructed wave will have a significant DC offset, we'll then AC couple it using a 1 uF capacitor and a 100k resistor to ground. Finally, we'll add an op amp set up as a non-inverting buffer for a low output impedance.

If you build and listen to this, you'll notice that while the amount of clock noise is definitely reduced, the output does still have a lot of whine in it. This is because first, the two signals are not perfectly complementary, so there is no clean handoff between samples. Instead, we see these brief transitions from sample to full state and back.

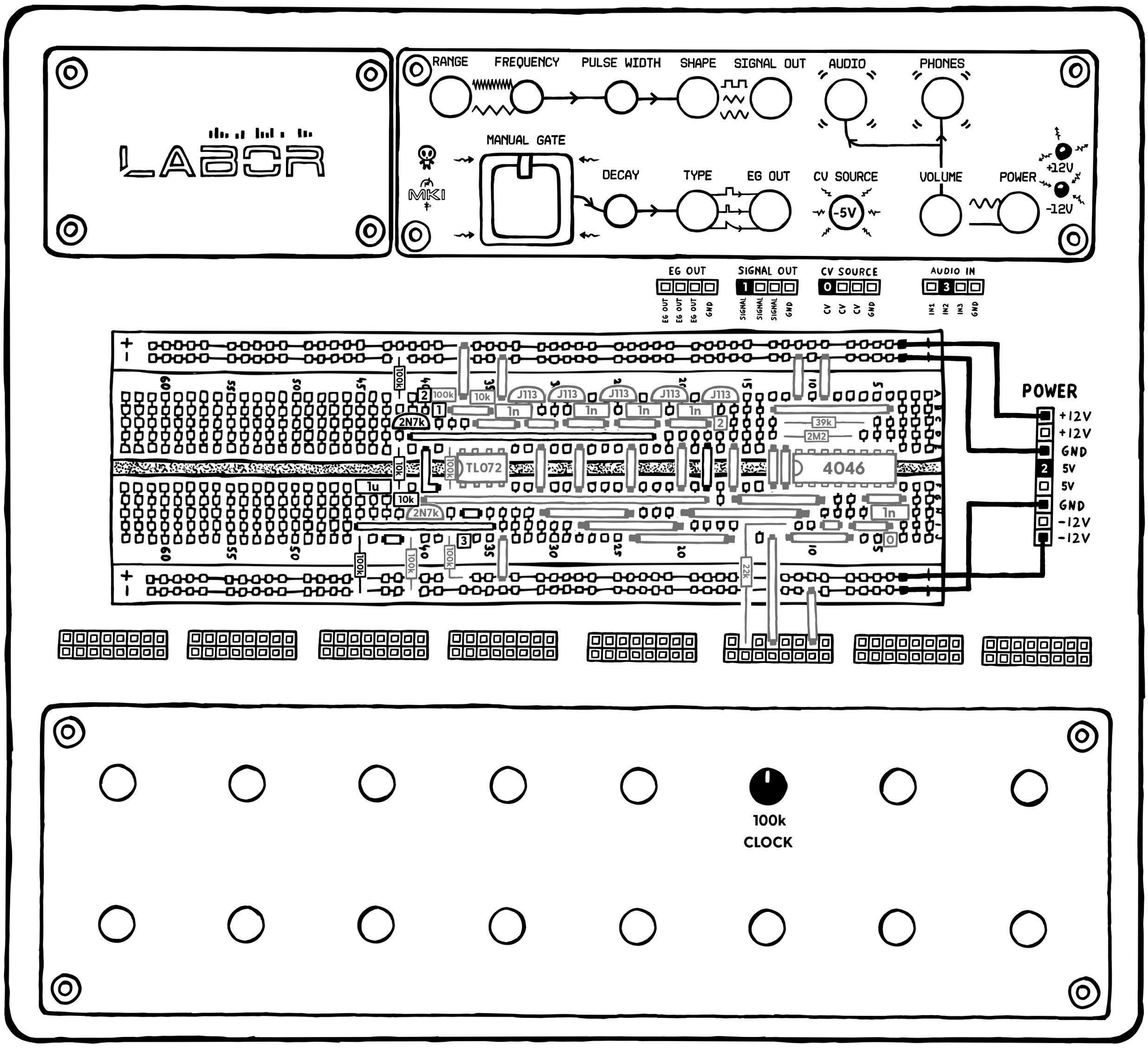


And second, the samples themselves also contain a clock spike, because the charge deficit transfer is not instantaneous. So what we're seeing here is one capacitor discharging into the previous one – until the deficit is fully transferred. **To try and deal with this, BBD circuits usually apply an aggressive low pass filter to the output signal.** To see how that would improve things, we can add a 10 nF capacitor to ground after our two resistors, creating a basic low pass filter.

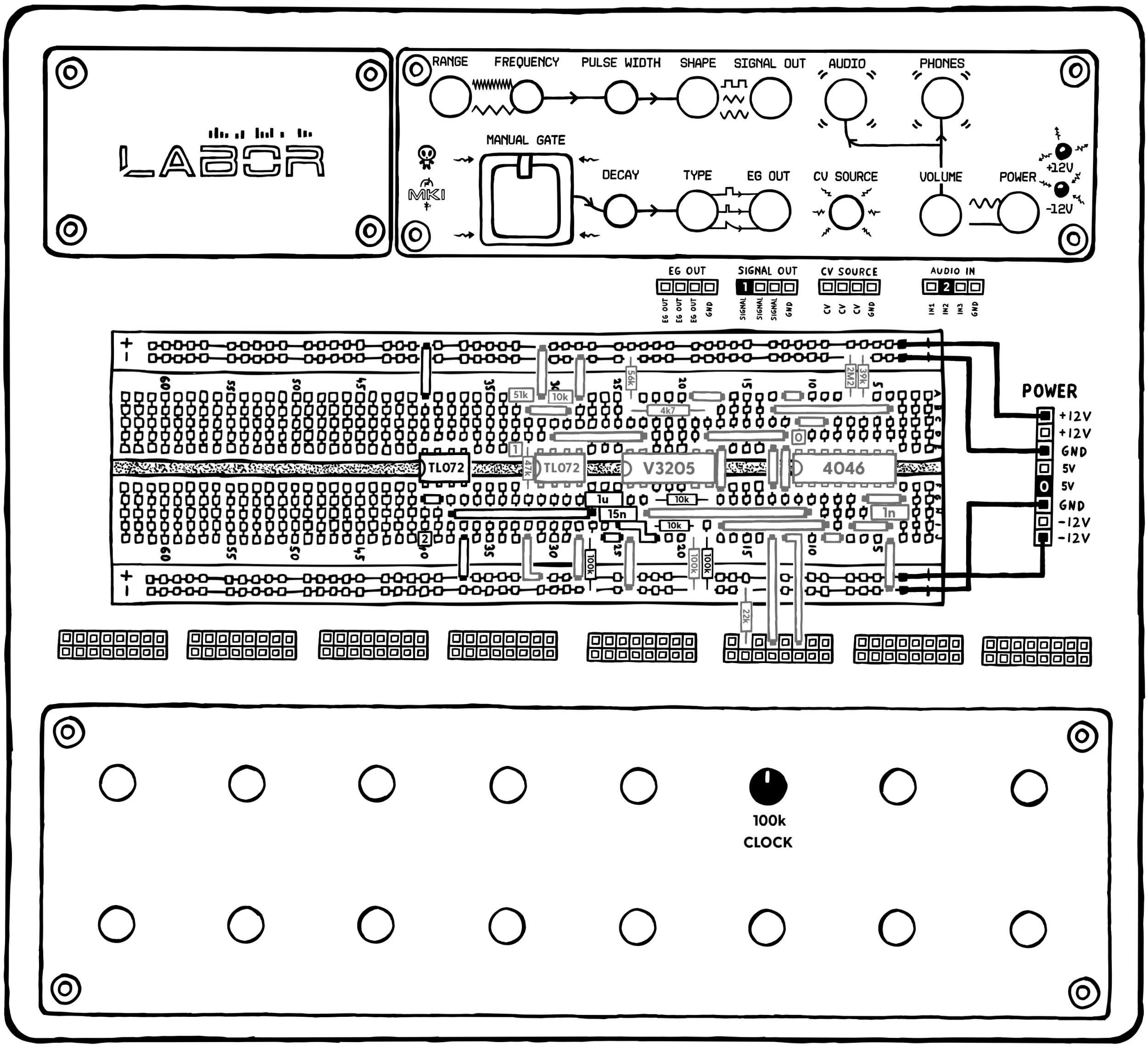


Of course this cuts away parts of the actual signal, but that's usually par for the course for a BBD – it's where they get their characteristic dark tone.[5]

[5] You can try this chapter's circuits in a simulator. I've already set them up for you right here. You can change all values by double clicking on components.



Please note that some of the components used in this layout are not included in your kit. If you don't have these extra components in your stash, build the alternate layout below that uses kit components exclusively.



If you check out the output on your oscilloscope, the waveform should look and sound a little cleaner. It's not exactly *good*, but we'll get back to this later. **Because at the moment, our circuit is still pretty useless as a delay, since three transfers at a clock frequency of 10 kHz gives us an effective delay time of just 300 microseconds**. So if you overlay the input and the delayed output on your oscilloscope, you'll see just a very small shift on the x-axis.[6]

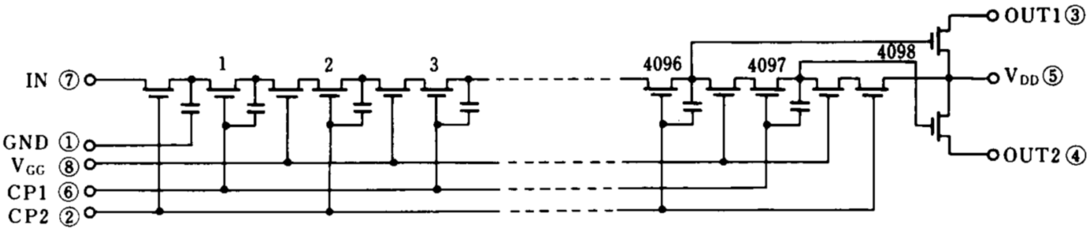
We can of course try to increase the delay time by reducing the clock speed. **But since that clock controls both the transfer process and the sampling process, the more we reduce the clock speed, the worse our sampling resolution gets**. Since we can only drop the clock speed so much before the output becomes completely unrecognisable, we should focus on a different strategy to increase the delay time instead: adding more stages to the chain. But because useful delay times at reasonable sampling rates can only be achieved with hundreds, if not thousands of stages, it's not feasible to set this up on our breadboard using discrete components.

---

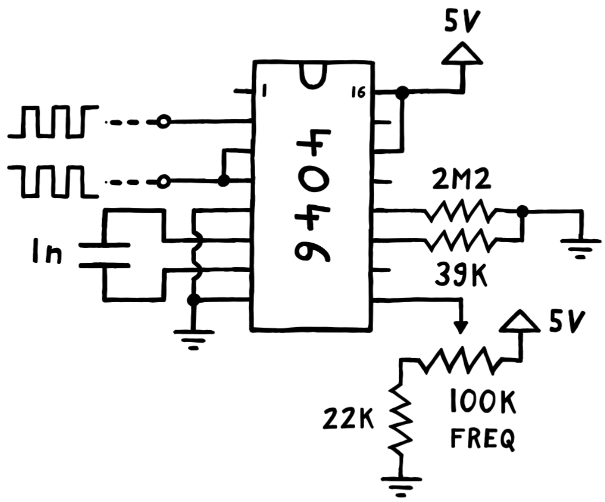
[6] This is not the case if you're already using the V3205.

MORE STAGES WITH THE V3205

Instead, we'll use a dedicated BBD chip: the V3205, which implements pretty much the same architecture we have just set up – but with 4096 stages instead of just three. There are two slight differences that'll require us to make changes to our existing circuit, though.

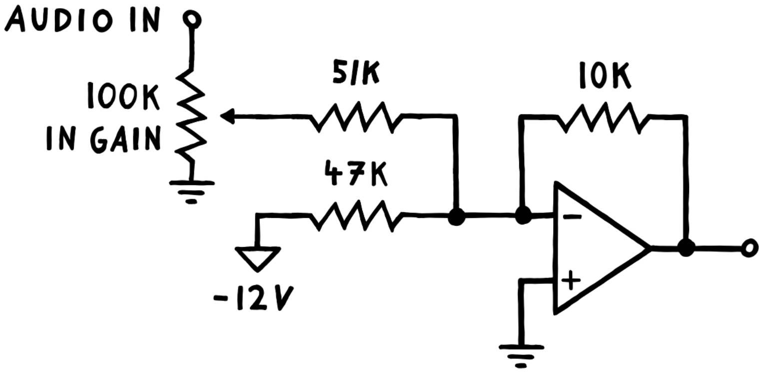


First, where we used JFETs, the chip exclusively uses MOSFETs. Because remember, non-discrete MOSFETs can actually be used for bidirectional switching. Since those MOSFETs are closed at 0 and open at 5 V gate voltage, we need to adjust our clocks to alternate between those values (instead of 0 and -5 V). To get there, all we need to do is feed our 4046 clock circuit 5 V as its high level and 0 V as its low level supply.



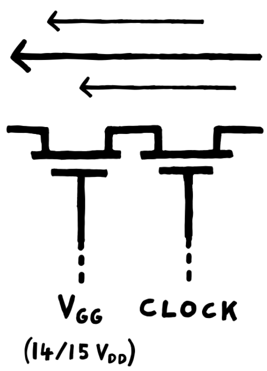
Additionally, we'll also need to adjust our input signal's bias & scale – since the MOSFETs in the V3205 operate with different gate to source thresholds than our JFETs. **In my experiments, the chip only responded well to signals swinging between around 1.9 and 3.2 V.** Anything above and below will be cut off.

To move our input into that range, we'll first replace the 200k offset resistor with a 47k. This will increase our offset by around 2 V.



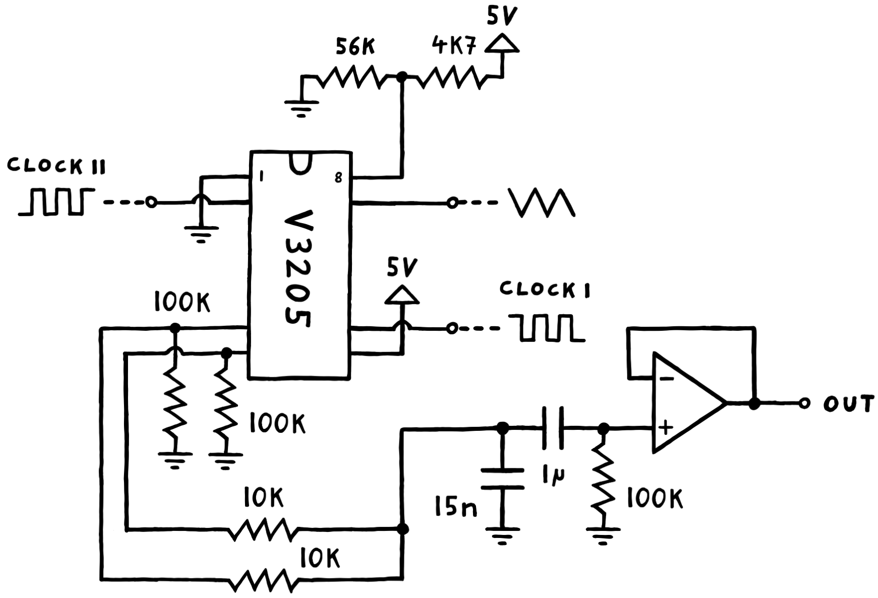
Next, we'll have to reduce the input resistance to increase the output's gain and make use of the available headroom. Because we might want to send in signals that are lower in volume than the eurorack-standard 10 V peak-to-peak, we'll drop that resistance down to 51k – and then insert a 100k variable voltage divider between it and the signal. **This way, we can dial in the appropriate input gain for a wide range of signals – instead of fixing it to a value that only works for a specific type**.

Okay. Now, the other difference between the chip and our JFET implementation is that the chip seemingly adds another transistor to each stage, whose gate is connected to a voltage labeled **VGG**. But this is actually just a (tricky) detail of how the individual delay stage MOSFETs are manufactured.

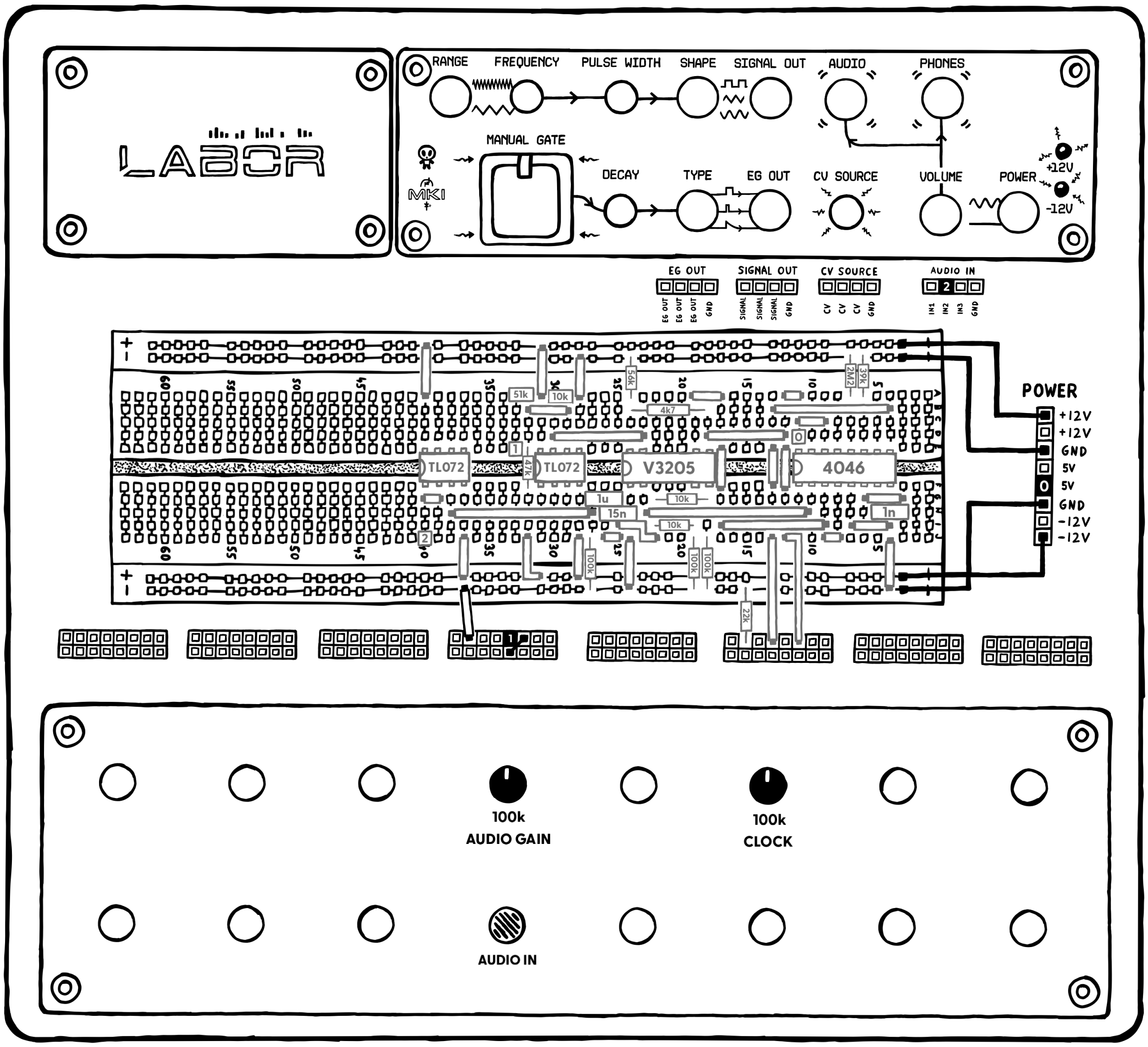


Sangster refers to this a „tetrode MOS structure": a single MOSFET with two gate terminals. By applying the clock to one of them (as we were) and providing the other with a high level voltage that is close to the supply (14/15 of the supply, to be precise), the MOSFET's ability to transfer charge is improved. I won't pretend I understand how this works in detail – but this refinement ensures that the charge deficit is completely transferred, with minimal error. Which ultimately means less distortion in our output.

To test all this, here's how we'll set up the V3205. First, it needs to be supplied with 5 V at pin 5 and ground at pin 1. Next, we'll give it the VGG voltage at pin 8 via a 4k7/56k voltage divider between 5 V and ground.



Then, we'll send clock 1 into pin 6, clock 2 into pin 2, and our scaled and biased input into pin 7. Finally, we'll drive our output reconstruction circuit from pins 3 and 4.

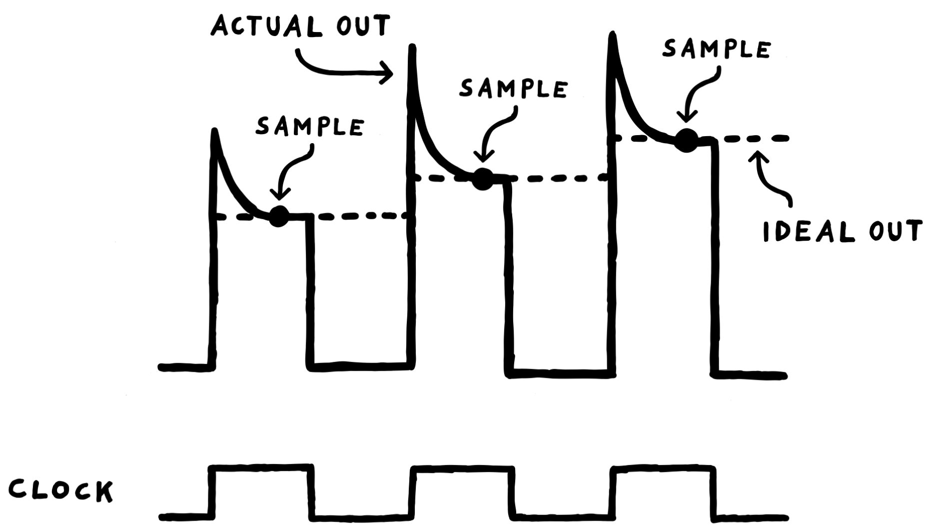


If you have previously built the discrete component version of our BBD using JFETs and 1 nF capacitors, please start with the alternate layout in the Sample Transfer chapter and work your way down to this one to set up the V3205.

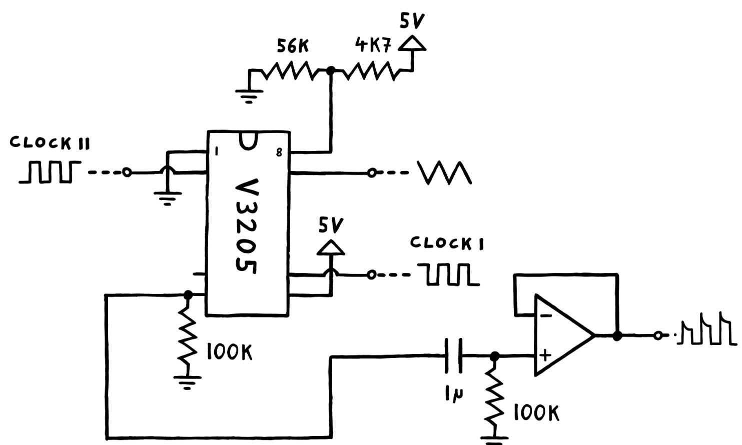
Once you've built this, try turning down the clock frequency. The added stages should allow you to significantly push the output back on the x-axis. Great! Though I'm really not happy with the amount of clock noise clock noise bleeding through, even with the low pass filter.

RECONSTRUCTION SAMPLING

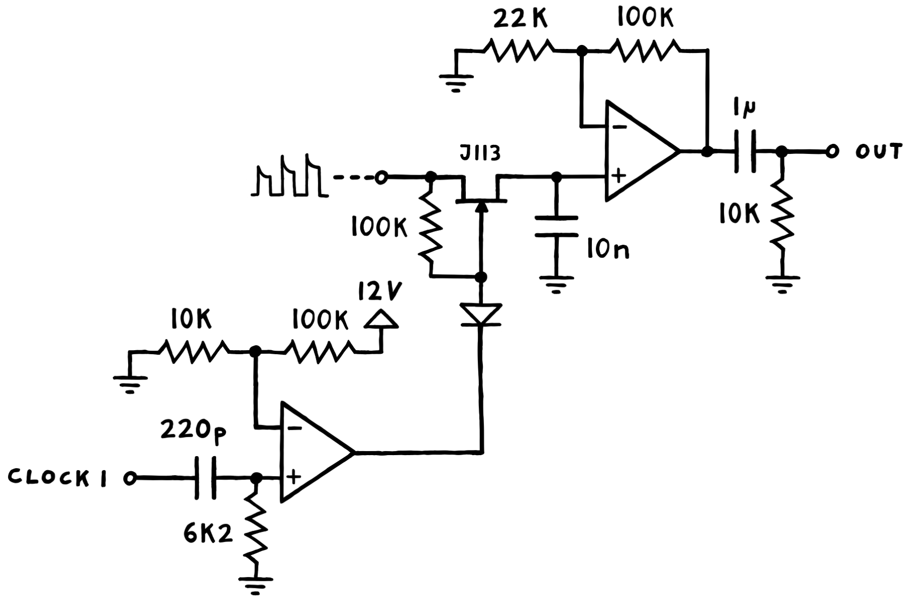
Traditionally, the way to fix the clock noise bleed problem has been to use a low pass with a much steeper slope than ours. This way, more of the annoying clock whine is cut out – but also more of the input signal's upper frequency range. And depending on how low we want our clock frequency to go, this would be a significant chunk. To avoid this, I want to go with a different approach that does not involve filtering.
To get there, let's go back to the raw, unmixed and unfiltered output from our BBD chain and compare it to the ideal, clean output that we're aiming for. You'll notice that where our actual output jumps between different voltage levels, the ideal output sticks to the same voltage for an entire clock cycle.



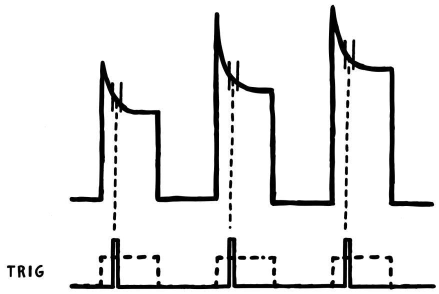
That voltage is the sample we took at the BBD's input. And we get that voltage in our actual output between the initial spike and the mid-clock cycle drop. So if we were to grab a small sample somewhere in that area and then hold that value until we grab the next one, we should be able to extract our ideal signal from the noisy BBD output.
To try and implement this idea, we'll first need to undo our mixing and filtering approach.



This means removing one output from the equation and killing the basic low-pass filter. However, we'll keep both the AC coupling and the buffer to ensure that we don't run into any headroom issues later on.
Next, we'll set up a general purpose sample & hold circuit. You can check our sample & hold kit's manual for an in-depth explanation, but there are three main differences compared to the sampling circuit we set up in the beginning of this manual.

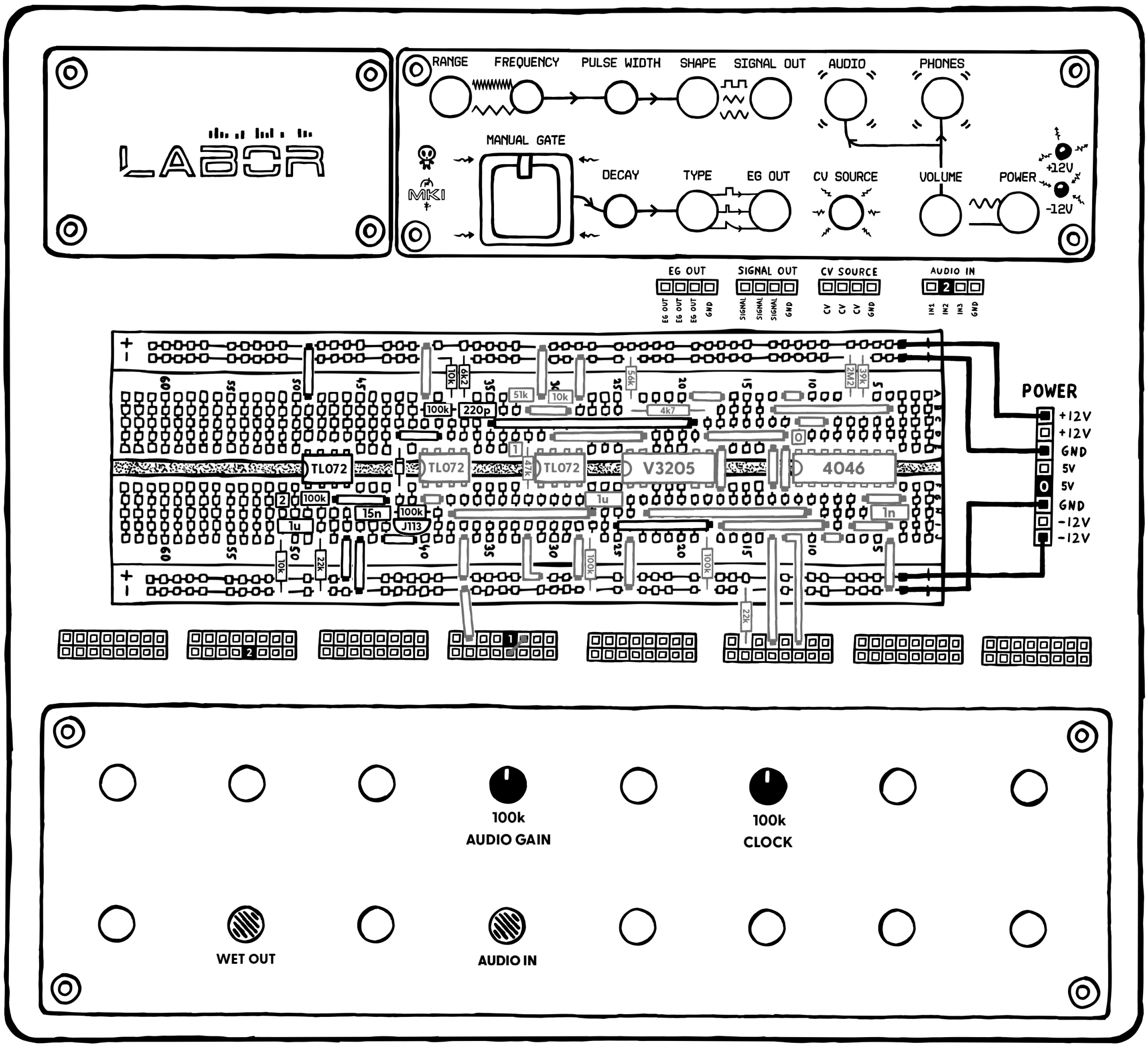


First, there is an added 100k resistor between the input and the JFET's gate terminal plus a diode between the gate and the sampling trigger. **This removes the need for biasing and scaling the input by ensuring that the gate is always locked to the source voltage during the sampling phase**. Which is very useful since we don't have to worry about what the signal coming out of the BBD chip looks like. We'll be able to sample it without re-biasing or re-scaling. (At the expense of a more complex circuit.)
Second, there is a gate-to-trigger converter between the clock and the diode at the JFET's gate. This turns the square wave clock into a super narrow pulse, giving us a very short sampling window. But not only that. Additionally, that gate-to-trigger converter will provide us with something called a propagation delay. **This means that there is always a very small time difference between the clock going high at the input and a trigger being generated at the output**.



This shifts the trigger pulse to the right ever so slightly, allowing us to dodge the worst part of the clock spike. Okay, but our sampling window still contains a sizeable chunk of that spike. Wouldn't this show up in the output? It would if charging and discharging the sampling capacitor was instantaneous. But depending on the size of that capacitor, charging and discharging it takes a considerable amount of time.

So by picking a relatively big capacitor in our sample & hold, we'll get more of a rough average voltage than a 1:1 reproduction of this downward slope.
Third and finally, there is a non-inverting amplifier at the output with a gain of around 5, restoring the signal to standard eurorack levels – plus another round of AC coupling to remove any remaining DC offset from the reconstructed signal.



To see how all this works in practice, I recommend that you first compare the gate-to-trigger converter's output to the clock signal on your oscilloscope. You should be able to see a clear gap between the clock's rising edge and the generated trigger. If you then swap the clock signal for the BBD chip's output, the trigger should be dodging the worst part of the spike as expected.
In combination with the big 15 nF sampling capacitor, this will produce an output that is surprisingly clean. To hear how it sounds, send a drum beat, a synth sequence or any other audio signal into the **audio in** socket. As you reduce the clock speed, you should hear the effect of the lower sampling rate, giving the sound a crunchy, digital feel. You'll

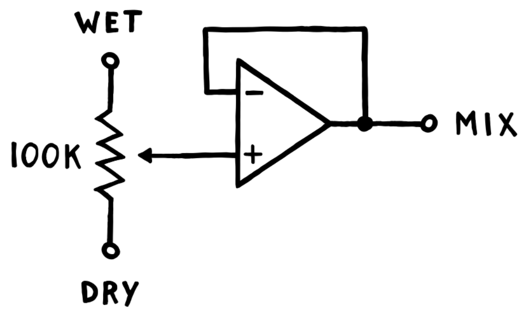
also notice that you're getting that typical re-pitching effect as you turn the clock frequency knob.

This happens because changing the clock speed alters the playback speed for the samples that are already inside the chip. **While those samples are being played back, the pitch changes according to the new playback speed**. And once that string of samples has reached the end of the transistor-capacitor-chain, the pitch returns to normal, as the sampling and playback speeds realign.

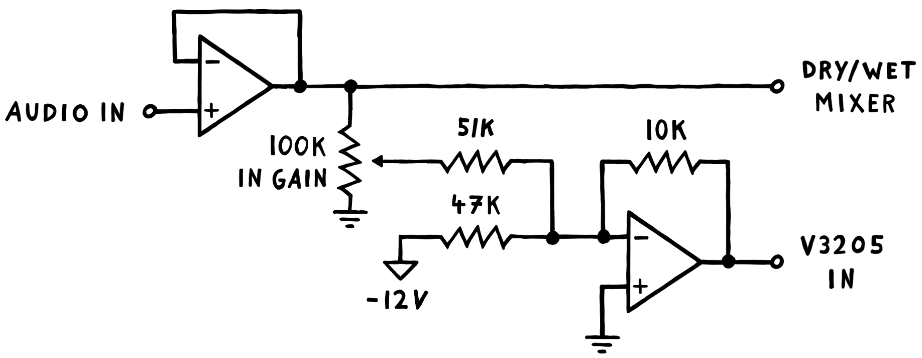
If you compare our current output to what we got previously, you'll notice that the reconstruction sampling approach gets rid of the clock noise almost entirely, while the traditional approach still has lots of noise plus a much more muffled sound. Which is pretty cool.

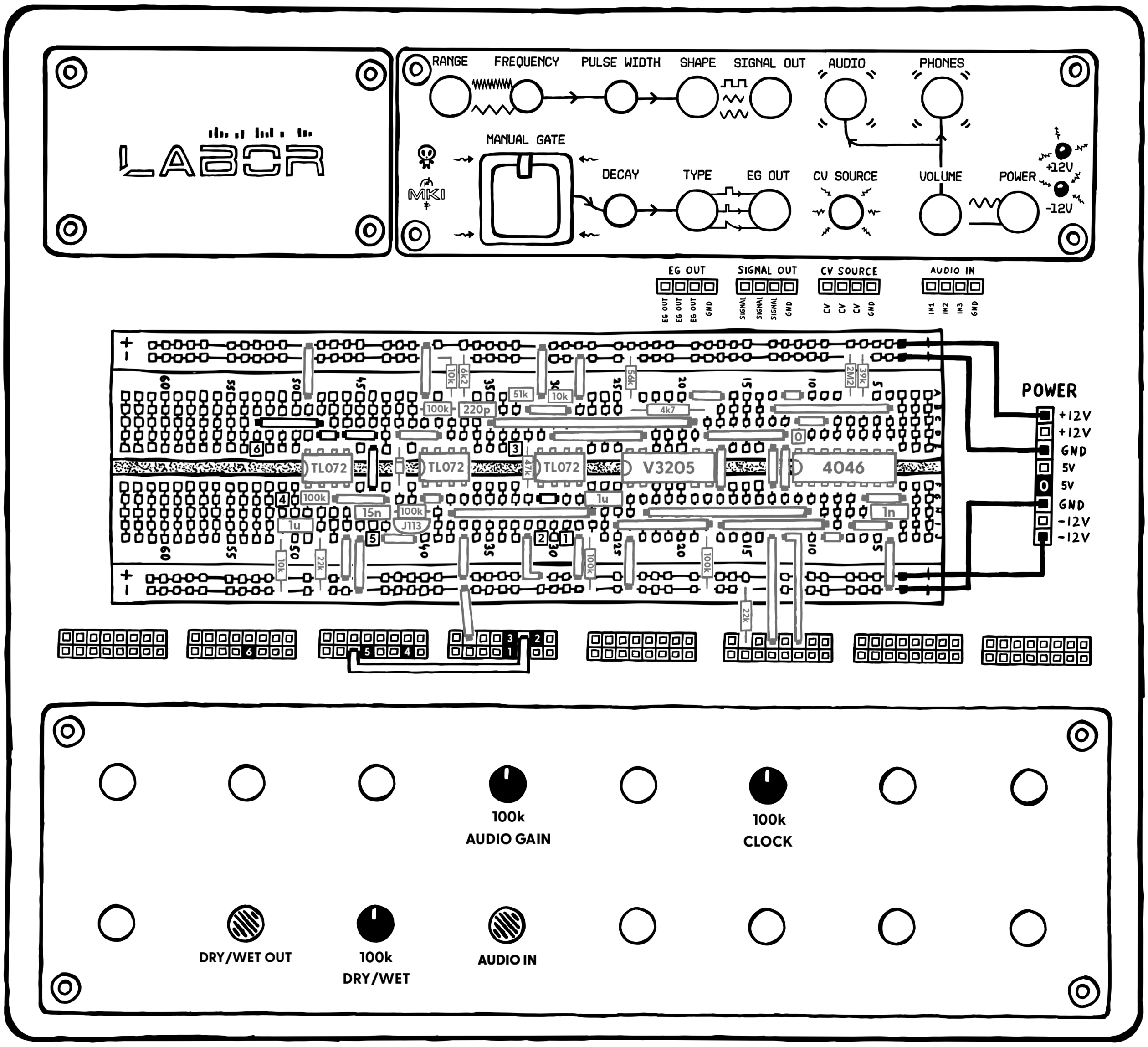
DRY/WET MIXING

Great! But so far, we're only getting what you would call a fully wet signal from our circuit, meaning that we only hear the delayed sound, not the original one. To change this, we'll add a simple dry/wet mixing stage. We'll start by sending the wet (delayed) and dry (unprocessed) signals to opposite sides of a 100k potentiometer.



Then, we'll buffer the signal at the pot's wiper with a simple op amp buffer. **This way, we've created an active mixer that allows us to apply different weights to both inputs simultaneously by turning a knob**. Because we now draw current from our input in two separate places, I'm going to also buffer it – just to make sure we don't load the signal down and potentially distort it.

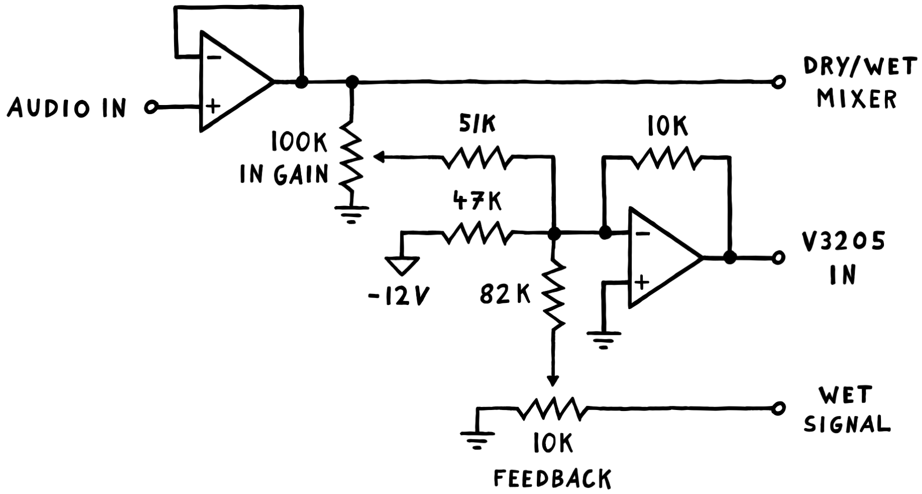




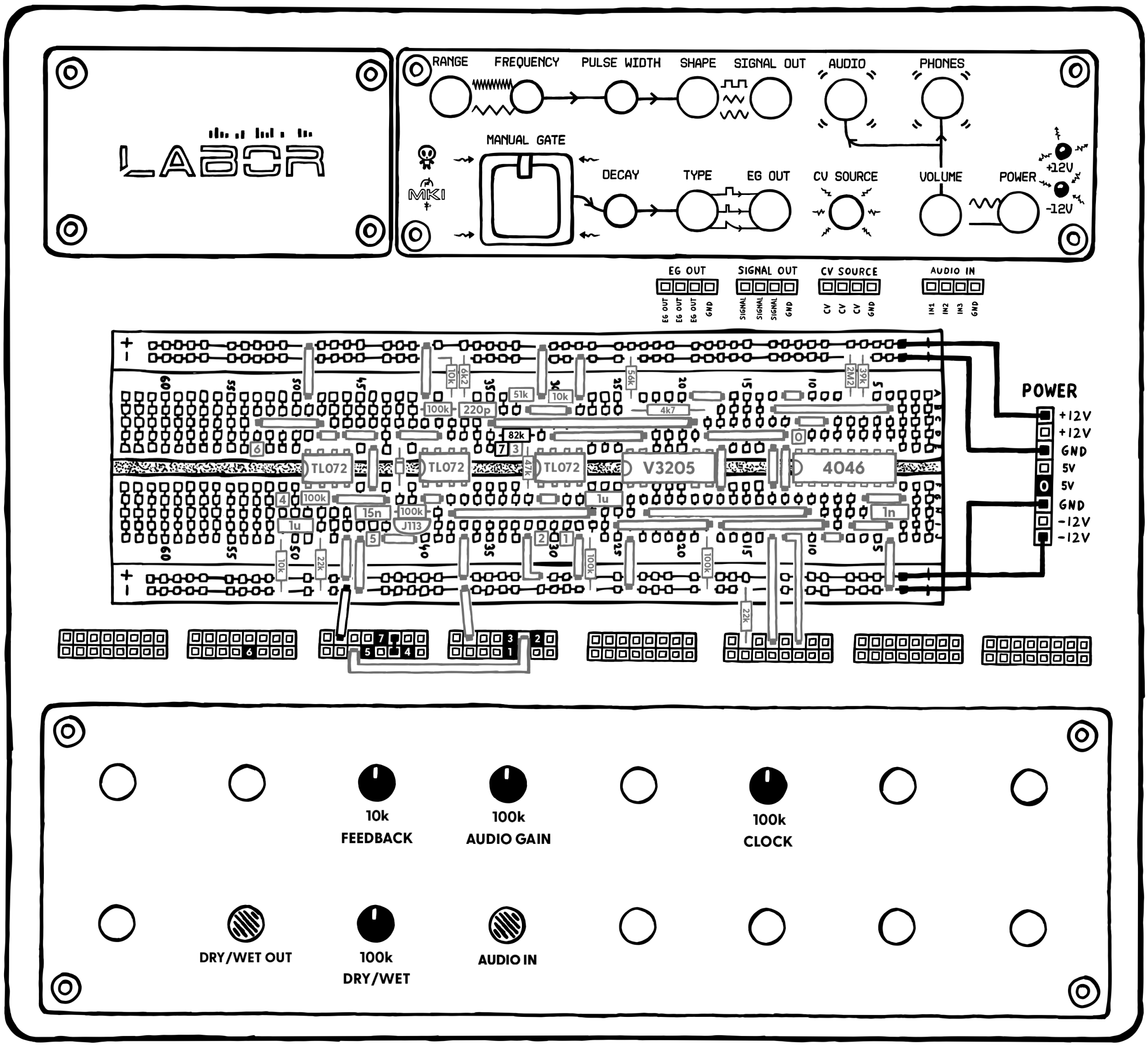
If you turn the **dry/wet** potentiometer all the way clockwise, you should only hear the wet signal. And if you move it in the other direction, more and more of the dry signal should come in until the wet version completely disappears. Great!

FEEDBACK

So far, so good. What we have now is what's commonly known as a slapback delay: a single delayed repeat. But what if we want multiple repeats instead of just one? Thankfully, getting there is really easy. **All we have to do is feed the wet signal back into the input via the scaling & biasing op amp.** Because then, every repeat will go through the BBD chip again and again.



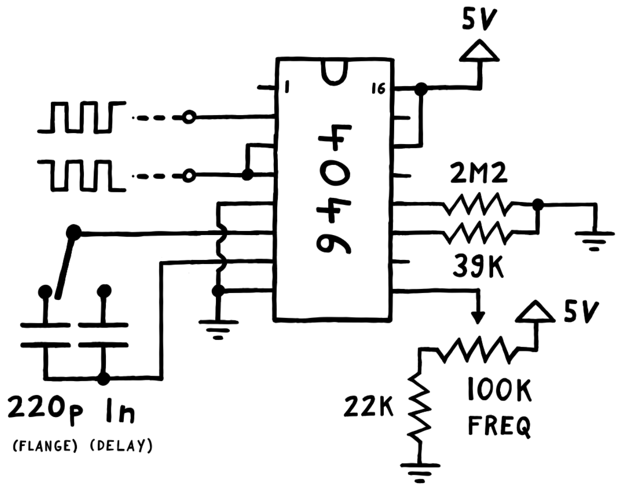
Using an 82k input resistor here will give us a feedback gain of about 1. Which would result in an infinite number of repeats. The circuit basically goes into self-oscillation. To be able to dial this back, we'll put a 10k potentiometer as a variable voltage divider between the wet output and the 82k resistor.
Small side note, but since the feedback potentiometer is providing a path to ground, we can get rid of the 10k AC coupling resistor at the wet output. It's simply redundant now.



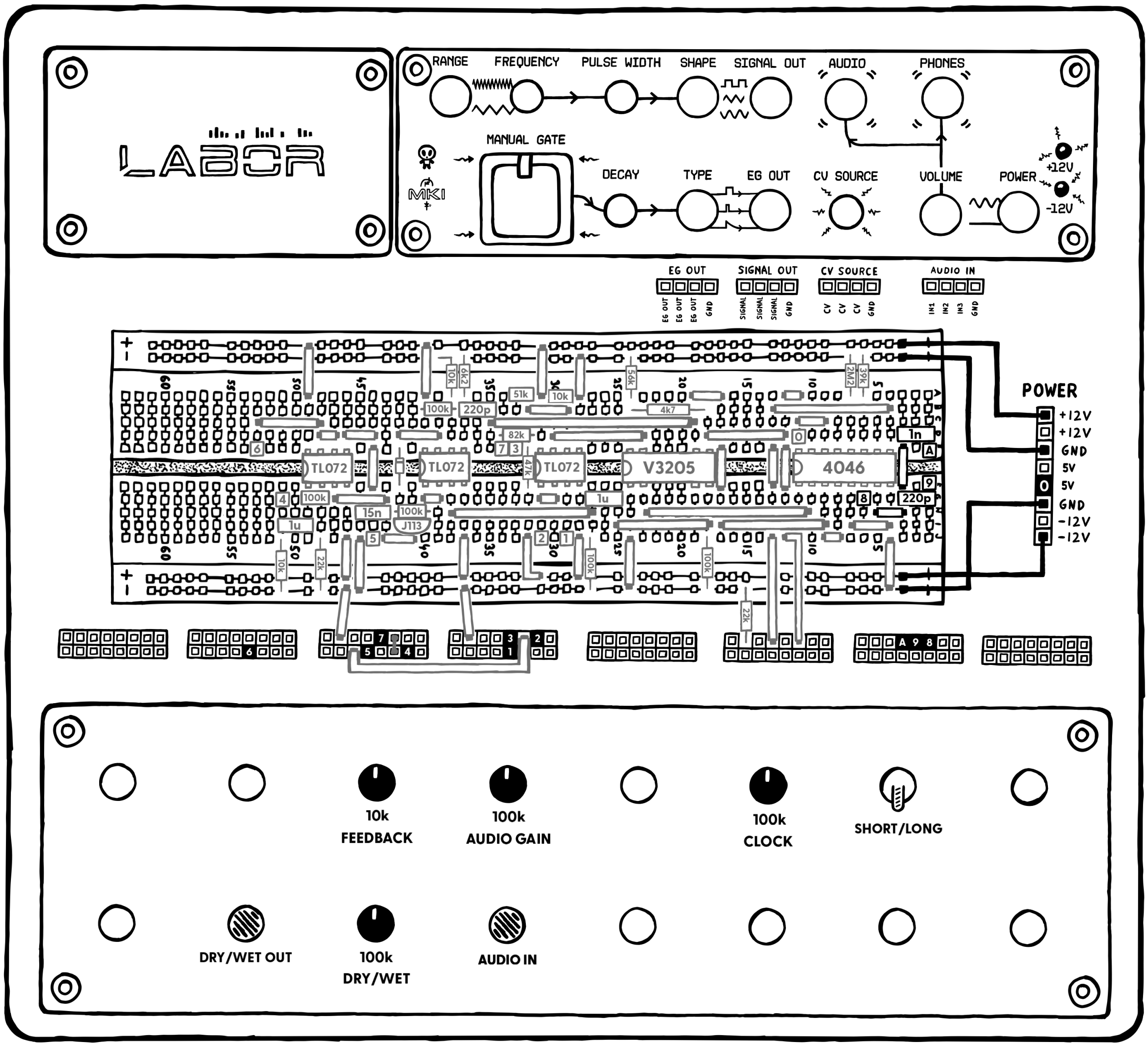
Using the new **feedback** potentiometer, you should be able to vary the number of repeats from just 1 to so many that the chip goes into wild self oscillation mode. Cool!

FLANGER MODE

But while our current delay time range is working great for, well, delay, I'd like to be able to push it up into flanger territory. A flanger, if you don't know, is basically just a delay with a very short delay time. **Since the delay time depends on the speed of our clock, we can shorten it by increasing the clock frequency**. For that, all we have to do is swap the 1 nF capacitor for a smaller one.



A 220 pF replacement gives us a frequency range of around 4 kHz to 112 kHz. Which should result in delay times ranging from around 1s to just 35 ms. And because I'd like to jump between delay and flanger mode on the fly, I'll set up both capacitor options behind a simple SPDT switch.

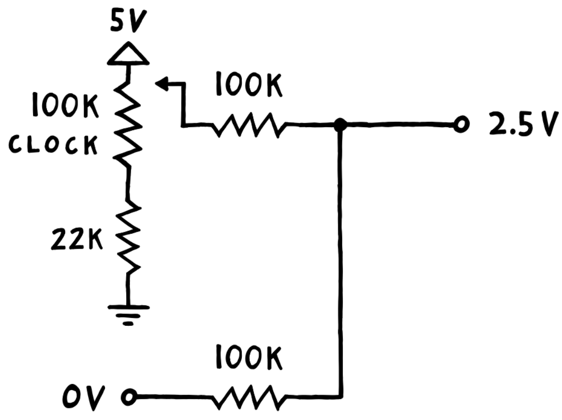


If you set the new switch to **short**, you should be able to dial in much shorter delay times, which, with a bit of feedback, start to sound really trippy. Great! But you might ask: why not use an even smaller capacitor for even shorter delay times? To answer that, take another look at the triggers driving our reconstruction sample & hold circuit compared to the BBD chip's output. Dial the clock to the maximum frequency.

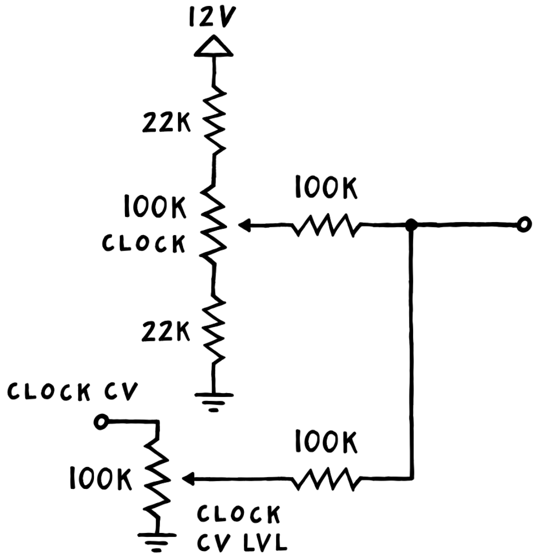
You should see that the parts of the output signal that actually carry the sample get smaller and smaller. Until they're almost smaller than the sampling window itself. **If they were to become smaller than the sampling window, we'd end up with severe distortion in the output**. So the 220 pF capacitor is actually the perfect choice here!

DELAY TIME MODULATION

Now, an actual flanger is not just a delay with a very short delay time – it also modulates that delay time using an LFO or similar. This modulation gives it that characteristic jet plane sound. To implement this in our circuit, we'll need to add a control voltage input for the clock frequency. **And this is why I went for a 4046-based clock generator in the first place: we're already controlling it with a voltage – we just need to find a way to mix that voltage with an additional external voltage**.
The naive approach here would be to use two 100k resistors as a simple passive mixer. Which would work in principle. But in practice, this would create a rather annoying usability issue.

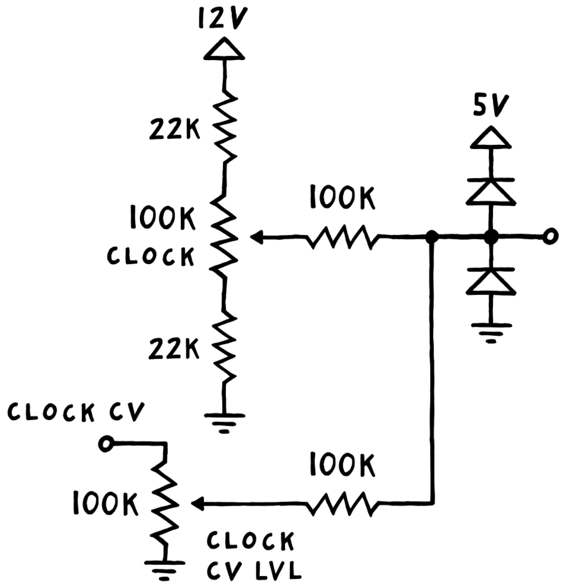


That's because a passive mixer doesn't produce the sum but rather the average of its input signals. Imagine we've got our delay time potentiometer dialed all the way up, causing it to send out a 5 V signal. If we now add an external signal that's currently idling at 0 V, you'd expect the delay time to stay the same – 0 V should mean that no modulation is applied. **But since our passive mixer is producing the average of our two signals, the resulting control voltage would drop down to 2.5 V**. Causing the delay time to drop down as well.
Bummer – how do we fix this? Well normally, you'd have to go through the trouble of adding a full blown summing mixer, involving two op amps and a couple additional resistors. But in our case, there is a clever trick we can apply. **If we modify the variable voltage divider to double its maximum output voltage, adding in the 0 V external voltage actually scales the knob's range back down to the original 1 V – 5 V**. We're essentially first multiplying and then dividing by 2.

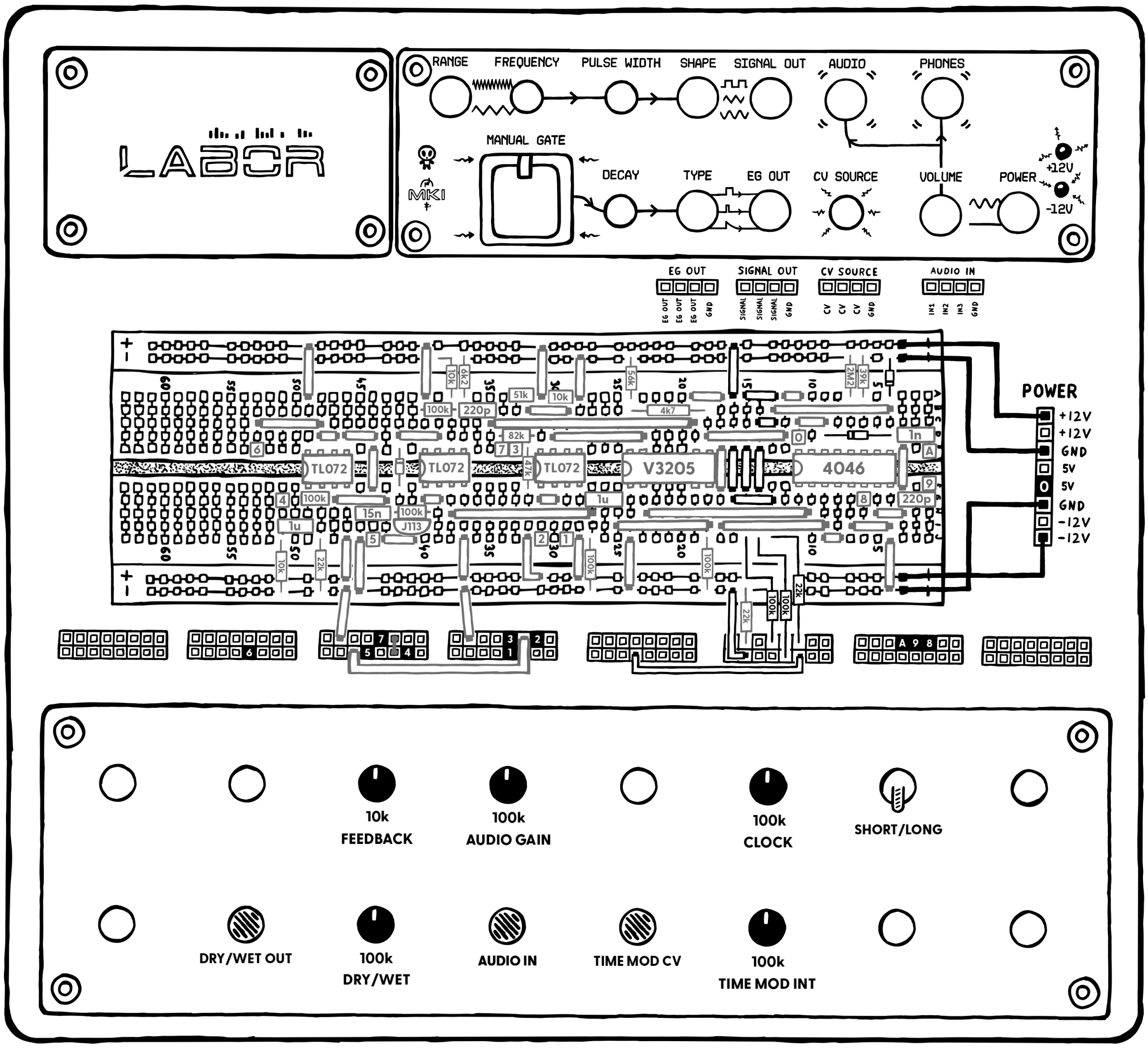


Okay, but what if there is no external signal plugged in at all? In that case, we would blast the 4046's CV input with 10 V – which it is not able to handle. To fix this, we simply add another variable voltage divider before the external CV input. This solves two problems at once. **First, it provides a path to ground even if there's no external signal plugged in. And second, it allows us to change the intensity with which the CV affects the delay time.**

There's one small issue left. What if the external CV is going above 5 or below 0 V? In that case, we might again apply voltages to the chip that it is not able to handle. To prevent that, we'll make use of diode-based limiting.



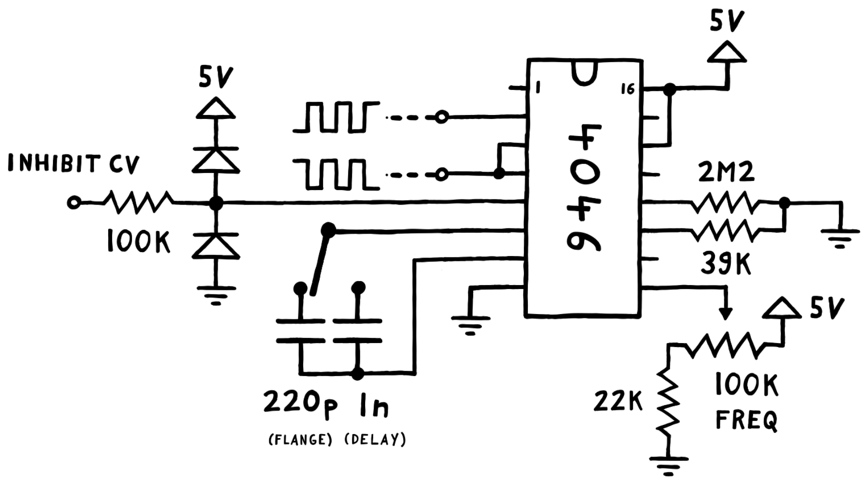
If we simply add in two diodes after our mixer – one pointing up to the 5 V rail and one pointing up from ground – then the voltage at the chip's input can never go significantly above 5 or below 0 V. That's because if it tries to, the diodes conduct and thereby neutralize the unwanted in- or decrease.



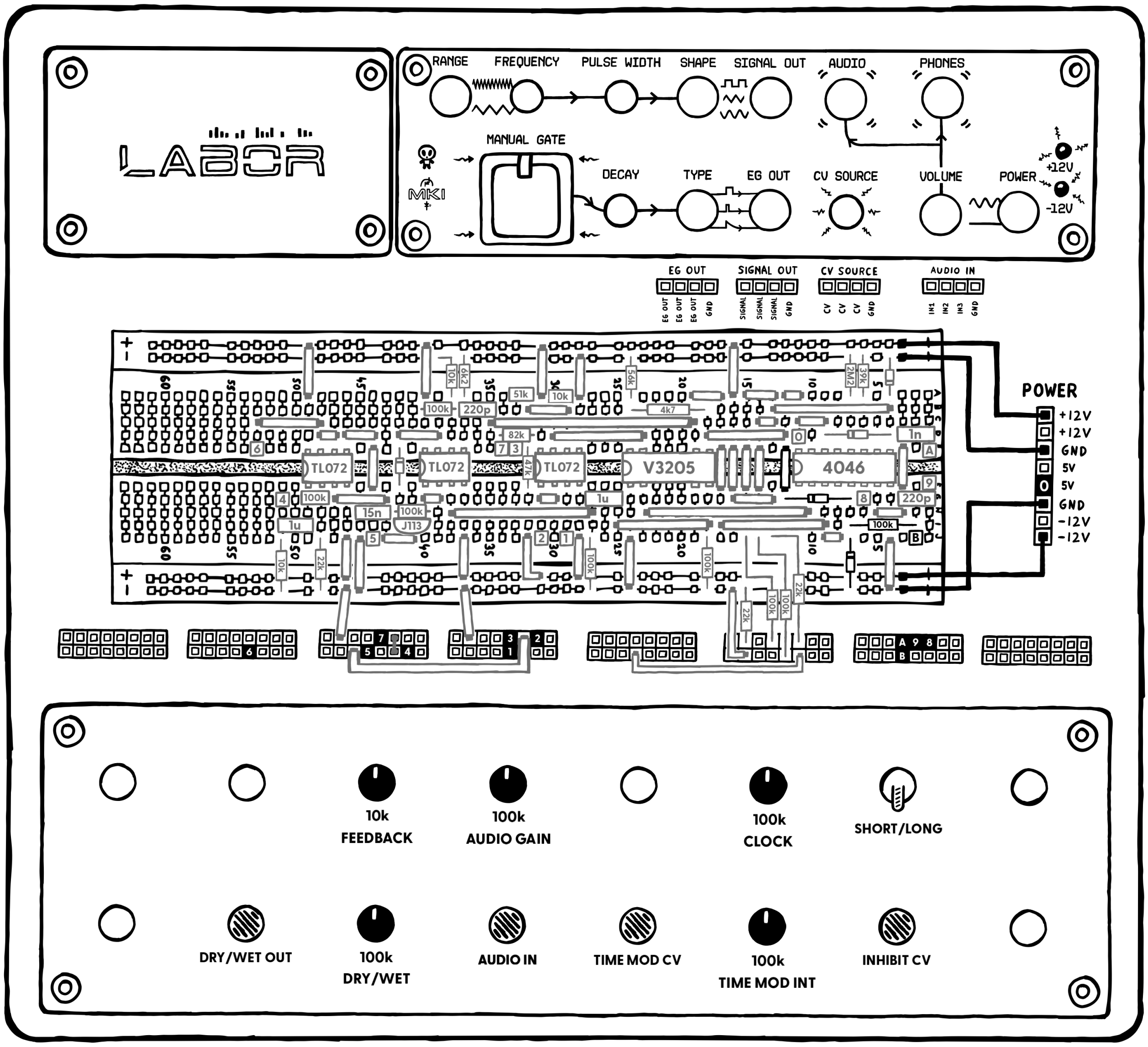
To test this, first try sweeping the delay time knob's range without any external CV applied. The range should stay roughly the same. Next, connect LABOR's oscillator to the clock CV input (while using something else for the audio input, of course). You should now be able to dial in different amounts of modulation going from subtle to extreme. Great!

INHIBIT CV

And while our circuit is now pretty much feature complete, there is a pin on the 4046 chip that caught my attention: the INHIBIT pin, which we've decided to tie straight to ground previously. As a reminder, the voltage applied to that pin decides whether the chip's oscillator is running or not. **So if we were to expose that pin via a jack socket, we should be able to start and stop our delay using a control voltage**.



To protect the pin from voltages below 0 or above 5 V, we can repurpose the diode-based limiting approach that we just came up with.



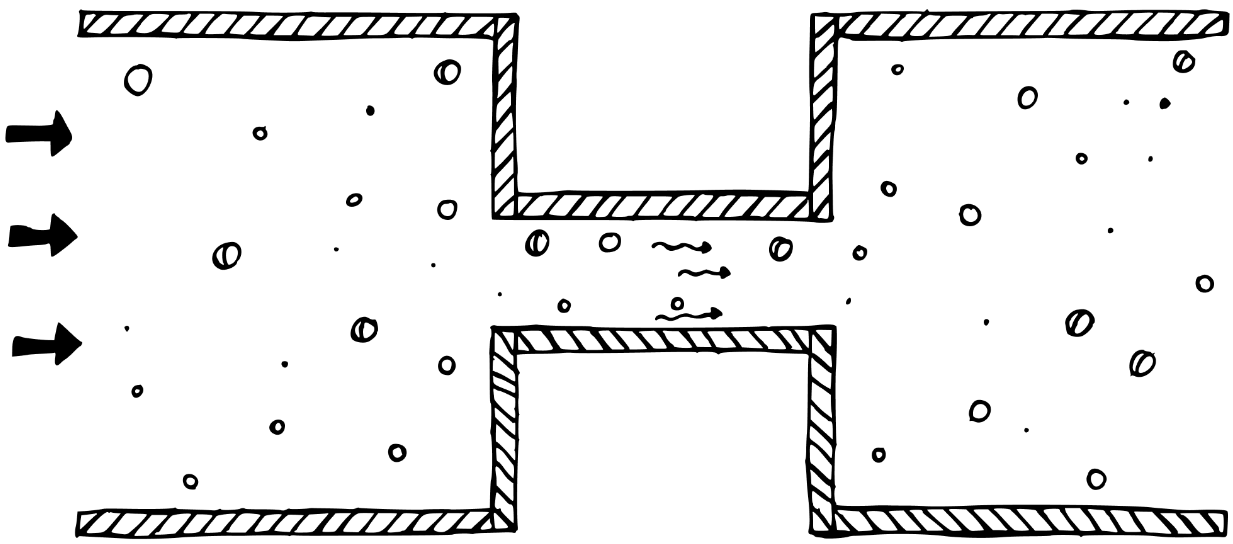
If you now apply a gate sequence to the new **inhibit cv** input, you should be able to create a fun rhythmic stuttering effect. Great! And with this, our bucket brigade delay is complete. Once you're done experimenting, dig out the panel and PCB from the kit, heat up your soldering iron and get to building. You can find more information on how to populate the board & how to solder in the enclosed appendix.

COMPONENTS & CONCEPTS APPENDIX

In this section, we'll take a closer look at the components and elemental circuit design concepts we're using to build our module. Check these whenever the main manual moves a bit too fast for you!

THE BASICS: RESISTANCE, VOLTAGE, CURRENT

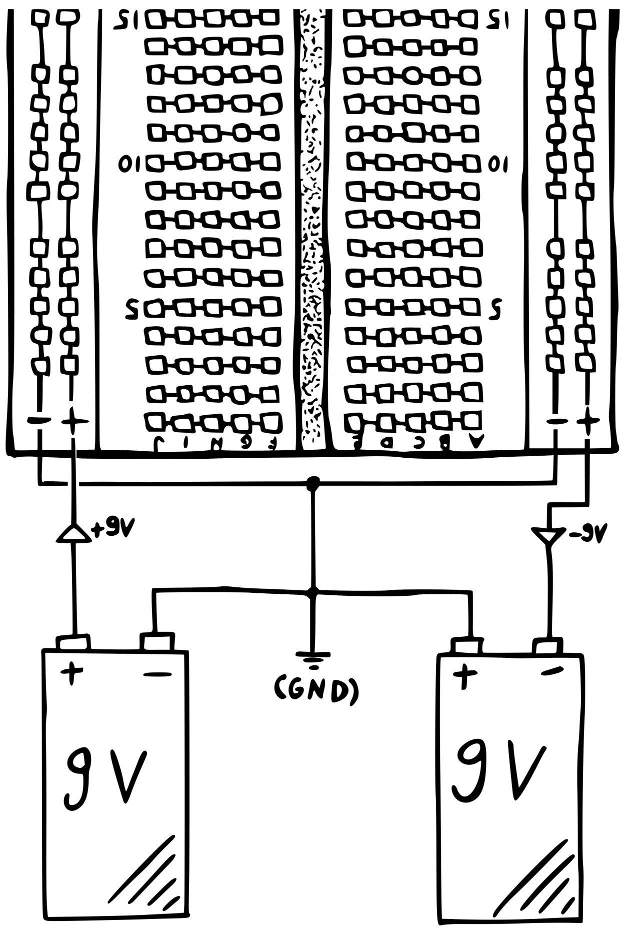
There are three main properties we're interested in when talking about electronic circuits: resistance, voltage and current. To make these less abstract, we can use a common beginner's metaphor and compare the flow of electrons to the flow of water through a pipe.



In that metaphor, resistance would be the width of a pipe. The wider it is, the more water can travel through it at once, and the easier it is to push a set amount from one end to the other. Current would then describe the flow, while voltage would describe the pressure pushing the water through the pipe. You can probably see how all three properties are interlinked: **more voltage increases the current, while more resistance to that voltage in turn decreases the current**.

USING TWO 9 V BATTERIES AS A DUAL POWER SUPPLY

Dual power supplies are great – and if you want to get serious about synth design, you should invest in one at some point. But what if you're just starting out, and you'd like to use batteries instead? Thankfully that's totally doable. **You just need to connect two 9 V batteries like shown here.** For this, you should use 9 V battery clips, which are cheap & widely available in every electronics shop.

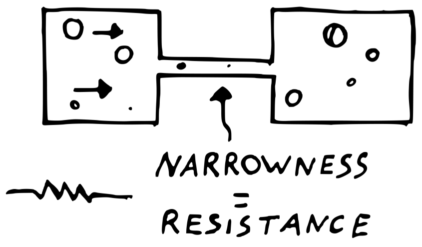


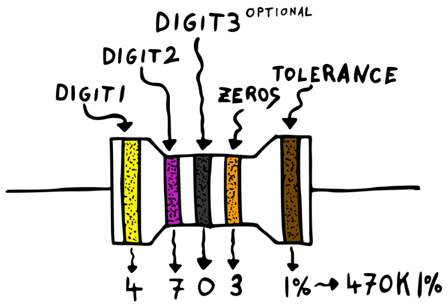
By connecting the batteries like this, the positive terminal of the left battery becomes your +9 V, while the negative terminal of the right is now your –9 V, and the other two combine to become your new ground.[7] **Please make sure you disconnect the batteries from your breadboard when you make changes to the circuit!** Otherwise you run the risk of damaging components.

[7] If you're struggling with setting this up, you can watch me do it here.

RESISTORS

While a conductive wire is like a very big pipe where lots of water can pass through, **a resistor is like a narrow pipe that restricts the amount of water that can flow**. The narrowness of that pipe is equivalent to the resistance value, measured in ohms (Ω). The higher that value, the tighter the pipe.



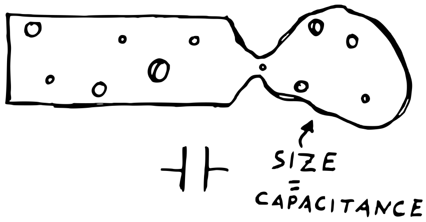


Resistors have two distinctive properties: linearity and symmetry. Linearity, in this context, means that for a doubling in voltage, the current flowing will double as well. Symmetry means that the direction of flow doesn't matter – resistors work the same either way.
On a real-life resistor, you'll notice that its value is not printed on the outside – like it is with other components. Instead, it is indicated by colored stripes[8] – along with the resistor's tolerance rating. In addition to that, the resistor itself is also colored. Sometimes, depending on who made the resistor, this will be an additional tolerance indicator.
For the resistors in this kit, a yellow body tells you that the actual resistance value might be ±5 % off. A dark blue body indicates ±1 % tolerance. Some kits will also contain light blue ± 0.1% resistors to avoid the need for manual resistor matching.
While in the long run, learning all these color codes will be quite helpful, you can also simply use a multimeter to determine a resistor's value.

[8] For a detailed breakdown, look up resistor color coding. There are also calculation tools available.

CAPACITORS

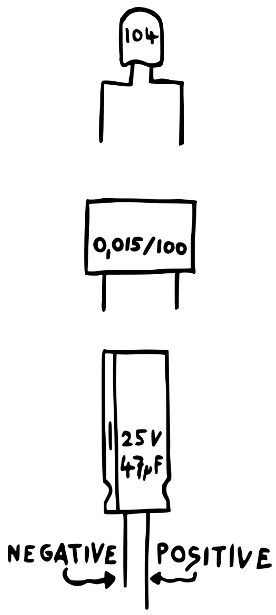
A capacitor is a bit like a balloon that you can attach to the open end of a pipe. If there's some pressure in the pipe, the balloon will fill up with water until the pressure equalizes. (Since the balloon needs some space to expand into, both of the capacitor's legs need to be connected to points in your circuit.)



Then, should the pressure in the pipe drop, the balloon releases the water it stored into the pipe. The maximum size of the balloon is determined by the capacitor's capacitance, which we measure in farad (F). There are quite a few different types of capacitors: electrolytic, foil, ceramic, tantalum etc. They all have their unique properties and ideal usage scenarios – but the most important distinction is if they are polarized or not.

You shouldn't use polarized capacitors against their polarization (applying a negative voltage to their positive terminal and vice versa) – so they're out for most audio-related uses like AC coupling, high- & low-pass filters etc.

Unlike resistors, capacitors have their capacitance value printed onto their casing, sometimes together with a maximum operating voltage. **Be extra careful here!** That voltage rating is important. Your capacitors can actually explode if you exceed it! So they should be able to withstand the maximum voltage used in your circuit. If they're rated higher – even better, since it will increase their lifespan. No worries though: the capacitors in this kit are carefully chosen to work properly in this circuit.



Ceramic capacitors usually come in disk- or pillow-like cases, are non-polarized and typically encode their capacitance value.[9] Annoyingly, they rarely indicate their voltage rating – so you'll have to note it down when buying them.

Film capacitors come in rectangular, boxy cases, are non-polarized and sometimes, but not always, directly indicate their capacitance value and their voltage rating without any form of encoding.[10]

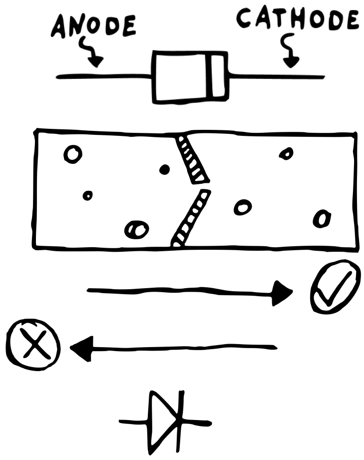
Electrolytic capacitors can be identified by their cylinder shape and silver top, and they usually directly indicate their capacitance value and their voltage rating. They are polarized – so make sure you put them into your circuit in the correct orientation.

---

[9] For a detailed breakdown, look up ceramic capacitor value code. There are also calculation tools available.

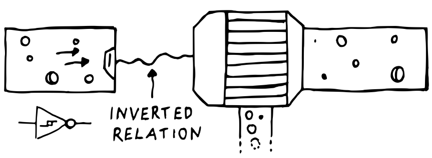
[10] If yours do encode their values, same idea applies here – look up film capacitor value code.

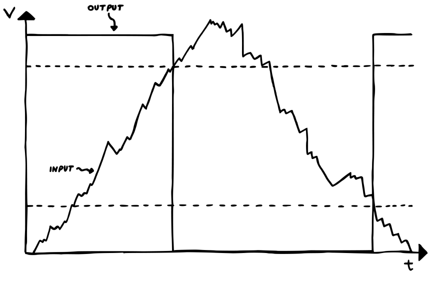
DIODES



Diodes are basically like one-way valves. Current can only pass through in one direction – from anode to cathode. That direction is indicated by the arrow in the diode symbol and by a black stripe on the diode's casing. So any current trying to move in the opposite direction is blocked from flowing.
There are a few quirks here, though. For one, the diode will only open up if the pushing force is strong enough. Generally, people say that's 0.7 V**,** but in reality, it's usually a bit lower. Also, diodes don't open up abruptly – they start conducting even at much lower voltages, although just slightly.
There are a lot of different diode types: Zener, Schottky, rectifier, small signal etc. They all have their unique properties and ideal usage scenarios – but usually, a generic 1N4148 small signal diode will get the job done.

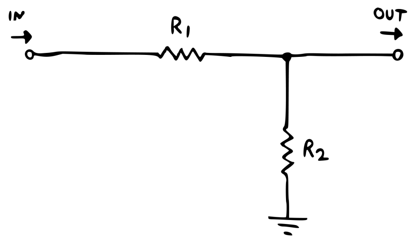
SCHMITT TRIGGER INVERTERS



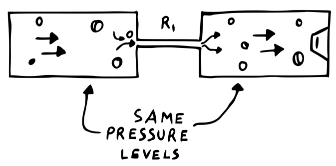


You can think of a Schmitt trigger inverter as two separate things. On the left, there's a sensor that measures the pressure inside an attached pipe. On the right, there is a water pump. This pump's operation is controlled by the sensor. Whenever the pressure probed by this sensor is below a certain threshold, the pump will be working. If the pressure is above a second threshold, the pump won't be working.
Here's a quick graph to visualize that. The squiggly line represents the voltage at the input, while the dotted line shows the voltage at the output. So every time we cross the upper threshold on our way up, and the lower one on our way down, the output changes its state. One thing that's very important to keep in mind: no current flows into the sensor! It's really just sensing the voltage without affecting it.

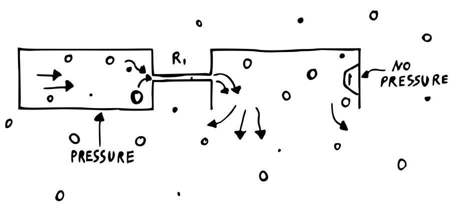
VOLTAGE DIVIDERS



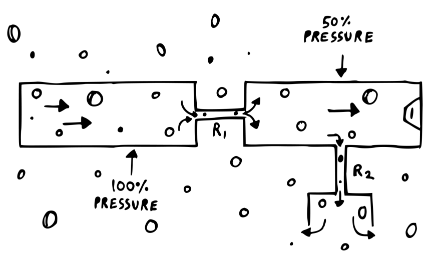
A voltage divider is really just two resistors set up like this: input on the left, output on the right. If R1 and R2 are of the same value, the output voltage will be half of what the input voltage is. How does it work?
Let's use our analogy again: so we have a pipe on the left, where water is being pushed to the right with a specific amount of force. Attached to it is a narrow pipe, representing R1, followed by another wide pipe. Then at the bottom, there's another narrow pipe, representing R2, where water can exit the pipe system. Finally, imagine we've set up a sensor measuring the voltage in the right hand pipe.



First, think about what would happen if R2 was completely sealed off. Our sensor would tell us that **the pressure on the right side is exactly the same as the pressure on the left**. Because the pushing force has nowhere else to go.



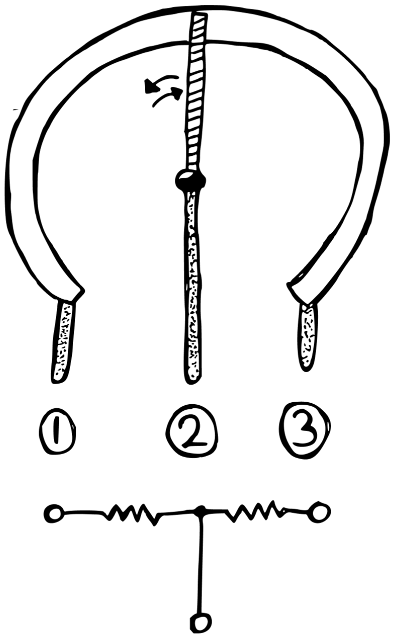
On the other hand, imagine R2 would just be a wide opening. Then **the pressure on the right would be 0**, because it'd all escape through that opening. But what happens if R2 is neither completely closed off nor wide open? Then the pressure would be retained to varying degrees, depending on the narrowness of the two resistor paths.



If pipe R1 is wide and pipe R2 is narrow, most of the pressure will be retained. But if it's the reverse, the pressure level will be only a tiny fraction. And if R1 and R2 are identical, **the pressure will be exactly half of what we send in**.

POTENTIOMETERS

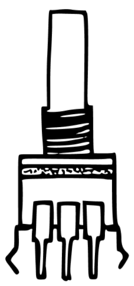
Potentiometers can be used as variable resistors that you control by turning a knob. But, and that's the handy part, they can also be set up as variable voltage dividers. To see how that works, let's imagine we open one up.

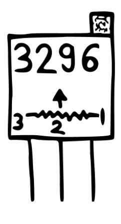


Inside, we would find two things: a round track of resistive material with connectors on both ends plus what's called a wiper. This wiper makes contact with the track and also has a connector. It can be moved to any position on the track. Now, the resistance value between the two track connectors is always going to stay exactly the same. That's why it's used to identify a potentiometer: as a 10k, 20k, 100k etc. But if you look at the resistance between either of those connectors and the wiper connector, you'll find that this is completely dependent on the wiper's position.

The logic here is really simple: **the closer the wiper is to a track connector, the lower the resistance is going to be between the two**. So if the wiper is dead in the middle, you'll have 50 % of the total resistance between each track connector and the wiper.

From here, you can move it in either direction and thereby shift the ratio between the two resistances to be whatever you want it to be. By now, you might be able to see how that relates to our voltage divider. If we send our input signal to connector 1 while grounding connector 3, we can pick up our output signal from the wiper. Then by turning the potentiometer's knob, we can adjust the voltage level from 0 to the input voltage – and anything in between.





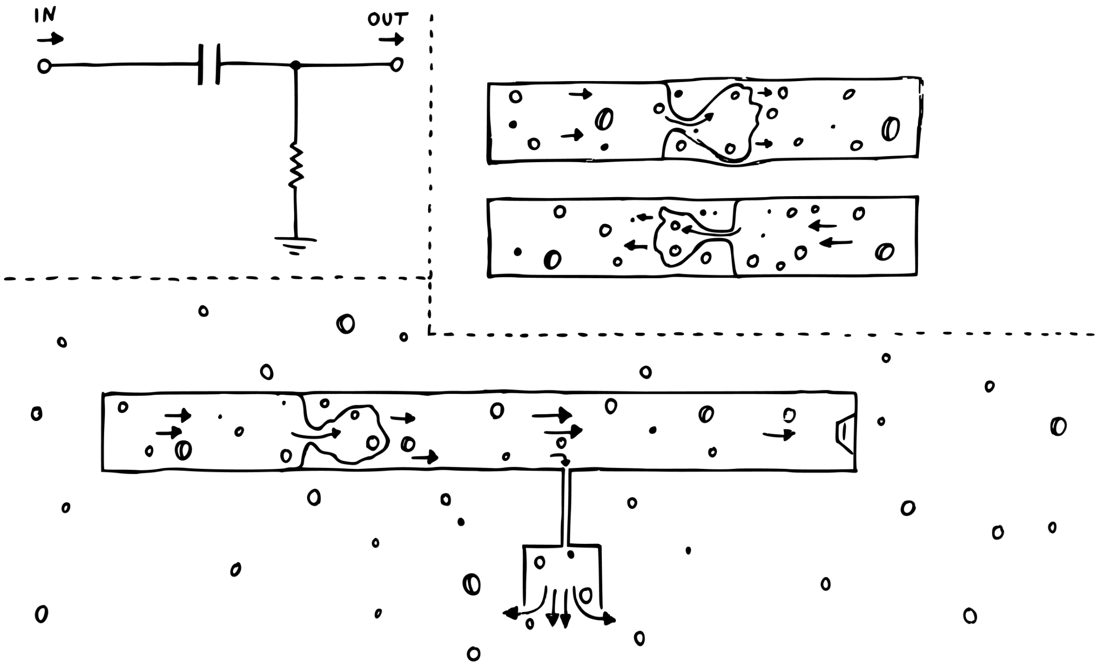
In these kits, you will encounter different types of potentiometers. First, there's the regular, full-size variant with a long shaft on top. These are used to implement user-facing controls on the module's panel and they usually – but not always – indicate their value directly on their casing. Sometimes, they'll use a similar encoding strategy as capacitors, though.[11]

Second, we've got the trimmer potentiometer, which is usually much smaller and doesn't sport a shaft on top. Instead, these have a small screw head which is supposed to be used for one-time set-and-forget calibrations. Trimmers usually encode their value.

[11] Look up potentiometer value code for a detailed breakdown.

AC COUPLING

What is AC coupling – and how does it work? Imagine two adjacent pipes with a balloon between them. Now, no water can get from one pipe into the other, since it's blocked by the balloon. But, and that's the kicker, **water from one side can still push into the other by bending and stretching the balloon, causing a flow by displacement**.

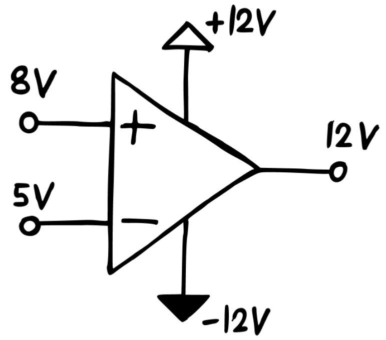


Next, we'll bring in a resistor after the coupling point, going straight to ground. **This acts like a kind of equalizing valve**. Now imagine we apply a steady 5 V from one side. Then on the other side, we'll read 0 V after a short amount of time. Why? Because we're pushing water into the balloon with a constant force, causing it to stretch into the other side, displacing some water. If we didn't have the equalizing valve there, we'd simply raise the pressure. But since we do have it, the excess water can drain out of the system. Until the pressure is neutralized, and no water is actively flowing anymore.

Okay, so now imagine that the voltage on the left hand side starts oscillating, let's say between 4 V and 6 V. When we start to go below 5 V, the balloon will begin contracting, basically pulling the water to the left. This will create a negative voltage level in the right hand pipe – like as if you're sucking on a straw, making the voltage there drop below 0 V. Then, once the pressure on the other side rises above 5 V, the balloon will inflate and stretch out again, pushing water to the right. And the pressure in the right hand pipe will go positive, making the voltage rise above 0 V. **We've re-centered our oscillation around the 0 V line.** Okay, but what about the resistor? If current can escape through it, doesn't that mess with our oscillation? Well, technically yes, but practically, we're choosing a narrow enough pipe to make the effect on quick pressure changes negligible!

OP AMPS

Op amps might seem intimidating at first, but they're actually quite easy to understand and use. The basic concept is this: every op amp has two inputs and one output. Think of those inputs like voltage sensors. You can attach them to any point in your circuit and they will detect the voltage there without interfering. **No current flows into the op amps inputs – that's why we say their input impedance is very high**. Near infinite, actually. Okay, but why are there two of them?



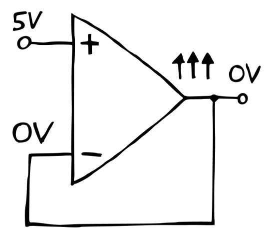
The key here is that op amps are essentially differential amplifiers. This means that they only amplify the difference between their two inputs – not each of them individually. If that sounds confusing, let's check out a quick example. So we'll imagine that one sensor – called the non-inverting input – is reading 8 V from somewhere. The other sensor – called the inverting input – reads 5 V. Then, as a first step, the op amp will subtract the inverting input's value from the non-inverting input's value. Leaving us with a result of 3. (Because 8 minus 5 is 3.) **This result then gets multiplied by a very large number – called the op amp's gain**. Finally, the op amp will try to push out a voltage that corresponds to that multiplication's result.

But of course, the op amp is limited here by the voltages that we supply it with. If we give it –12 V as a minimum and +12 V as a maximum, the highest it can go will be +12 V. So in our example, even though the result of that multiplication would be huge, the op amp will simply push out 12 V here and call it a day.

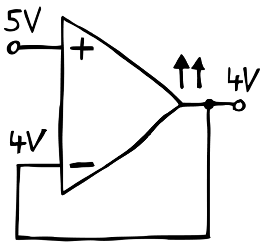
The handy thing though about op amp outputs is that they draw their power directly from the power source. This means that they can supply lots of current while keeping the voltage stable. **That's why we say an op amp has a very low output impedance**.

OP AMP BUFFERS/AMPLIFIERS

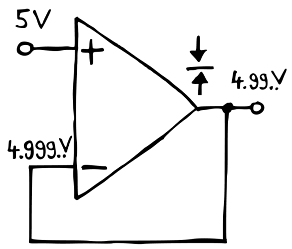
Buffering, in the world of electronics, means that we provide a perfect copy of a voltage without interfering with that voltage in the process. With an op amp-based buffer, the buffering process itself works like this. We use the non-inverting input to probe a voltage, while the inverting input connects straight to the op amp's output. **This creates what we call a negative feedback loop**. Think of it this way. We apply a specific voltage level to the non-inverting input – let's say 5 V.



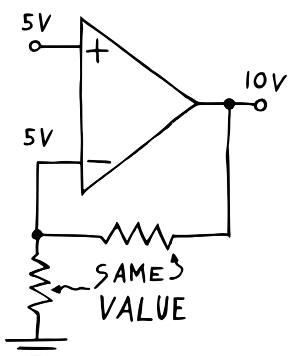
Before the op amp starts processing the voltages at its inputs, the output will be switched off. This means that **output and inverting input sit at 0 V at first**. So then, the op amp will subtract 0 from 5 and multiply the result by its gain. Finally, it will try and increase its output voltage to match the calculation's outcome.



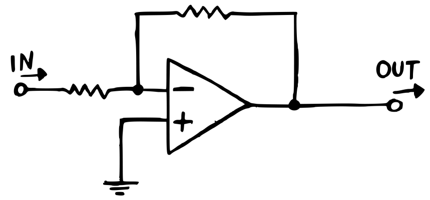
But as it's pushing up that output voltage, the **voltage at the inverting input will be raised simultaneously**. So the difference between the two inputs is shrinking down. Initially, this doesn't matter much because the gain is so large. As the voltage at the inverting input gets closer to 5 V though, the difference will shrink so much that in relation, the gain suddenly isn't so large anymore.



Then, the output will **stabilize at a voltage level that is a tiny bit below 5 V**, so that the difference between the two inputs multiplied by the huge gain gives us exactly that voltage slightly below 5 V. And this process simply loops forever, keeping everything stable through negative feedback. Now if the voltage at the non-inverting input changes, that feedback loop would ensure that the output voltage is always following. So that's why this configuration works as a buffer: the **output is simply following the input**.



How about amplifying a signal though? To do that, we'll have to turn our buffer into a proper non-inverting amplifier. We can do that by replacing the straight connection between inverting input and output with a voltage divider, forcing the op amp to work harder. Here's how that works. Say we feed our non-inverting input a voltage of 5 V. Now, **the output needs to push out 10 V in order to get the voltage at the inverting input up to 5 V**. We call this setup a non-inverting



amplifier because the output signal is in phase with the input.

For an inverting buffer/amplifier, the input signal is no longer applied to the non-inverting input. Instead, that input is tied directly to ground. So it'll just sit at 0 V the entire time. The real action, then, is happening at the inverting input. Here, we first send in our waveform through a resistor. Then, the inverting input is connected to the op amp's output through another resistor of the same value.

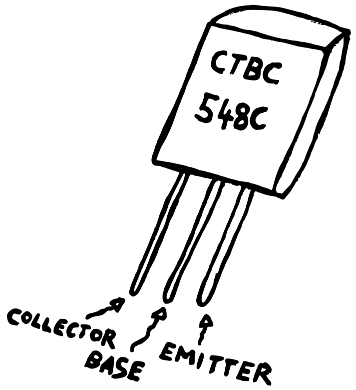
How does this work? Well, let's assume that we're applying a steady voltage of 5 V on the left. Then, as we already know, the op amp will subtract the inverting input's voltage from the non-inverting input's voltage, leaving us with a result of –5 V. Multiply that by the huge internal gain, and the op amp will try to massively decrease the voltage at its output.

But as it's doing that, an increasingly larger current will flow through both resistors and into the output. Now, as long as the pushing voltage on the left is stronger than the pulling voltage on the right, some potential (e.g. a non-zero voltage) will remain at the inverting input. Once the output reaches about –5 V though, we'll enter a state of balance. Since both resistors are of the same value, the pushing force on the left is fighting the exact same resistance as the pulling force on the right. **So all of the current being pushed through one resistor is instantly being pulled through the other**.

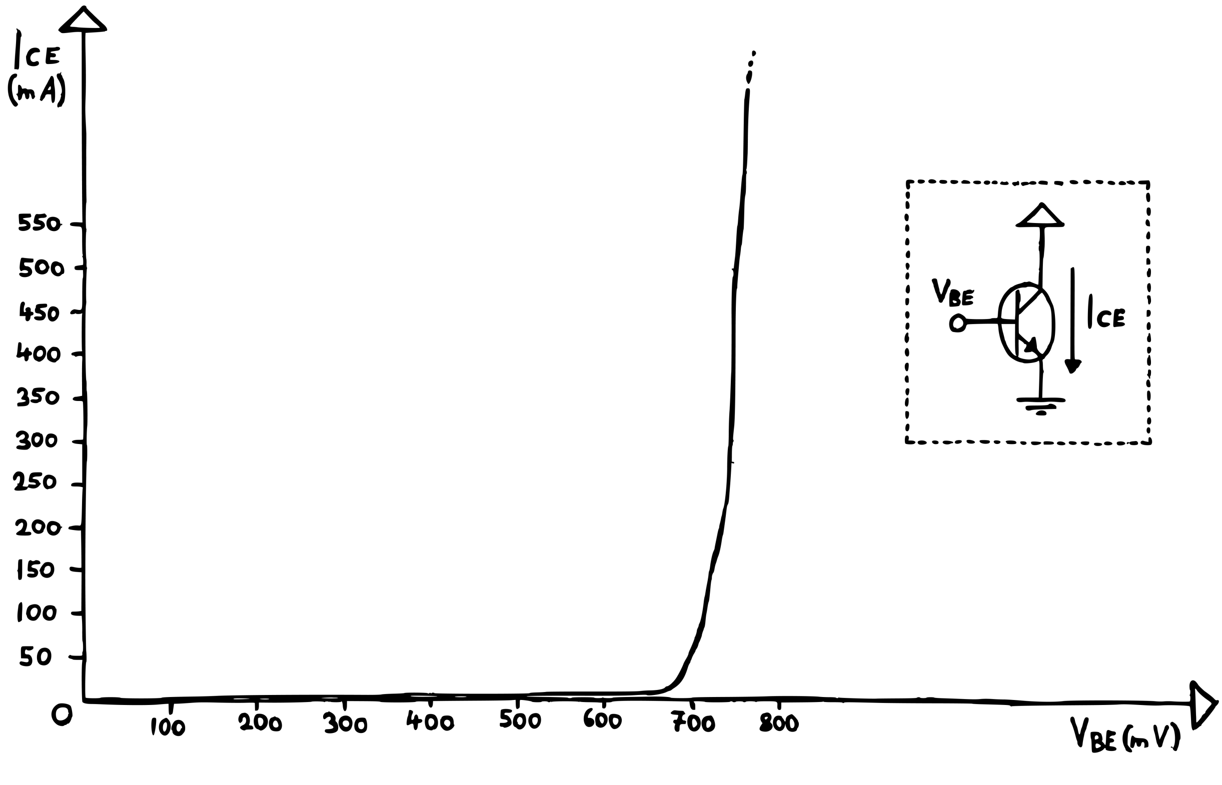
And that means that the voltage at the inverting input will be lowered to about 0 V, allowing our op-amp to settle on the current output voltage level. So while we read 5 V on the left, we'll now read a stable –5 V at the op amp's output. Congrats – we've built an inverting buffer! **If we want to turn it into a proper amplifier, we'll simply have to change the relation between the two resistances**. By doing this, we can either increase (if you increase the right-hand resistor's value) or reduce (if you increase the left-hand resistor's value) the gain to our heart's content.

BIPOLAR JUNCTION TRANSISTORS

Bipolar junction transistors (or BJTs for short) come in two flavors: NPN and PNP. This refers to how the device is built internally and how it'll behave in a circuit. Apart from that, they look pretty much identical: a small black half-cylinder with three legs.



Let's take a look at the more commonly used NPN variant first. Here's how we distinguish between its three legs. **There's a collector, a base and an emitter.**[12] All three serve a specific purpose, and the basic idea is that you control the current flow between collector and emitter by applying a small voltage[13] to the base. The relation is simple: **more base voltage equals more collector current**. Drop it down to 0 V and the transistor will be completely closed off. Sounds simple – but there are four important quirks to this.



First, the relation between base voltage and collector current is exponential. Second, unlike a resistor, a BJT is not symmetrical – so we can't really reverse the direction of the

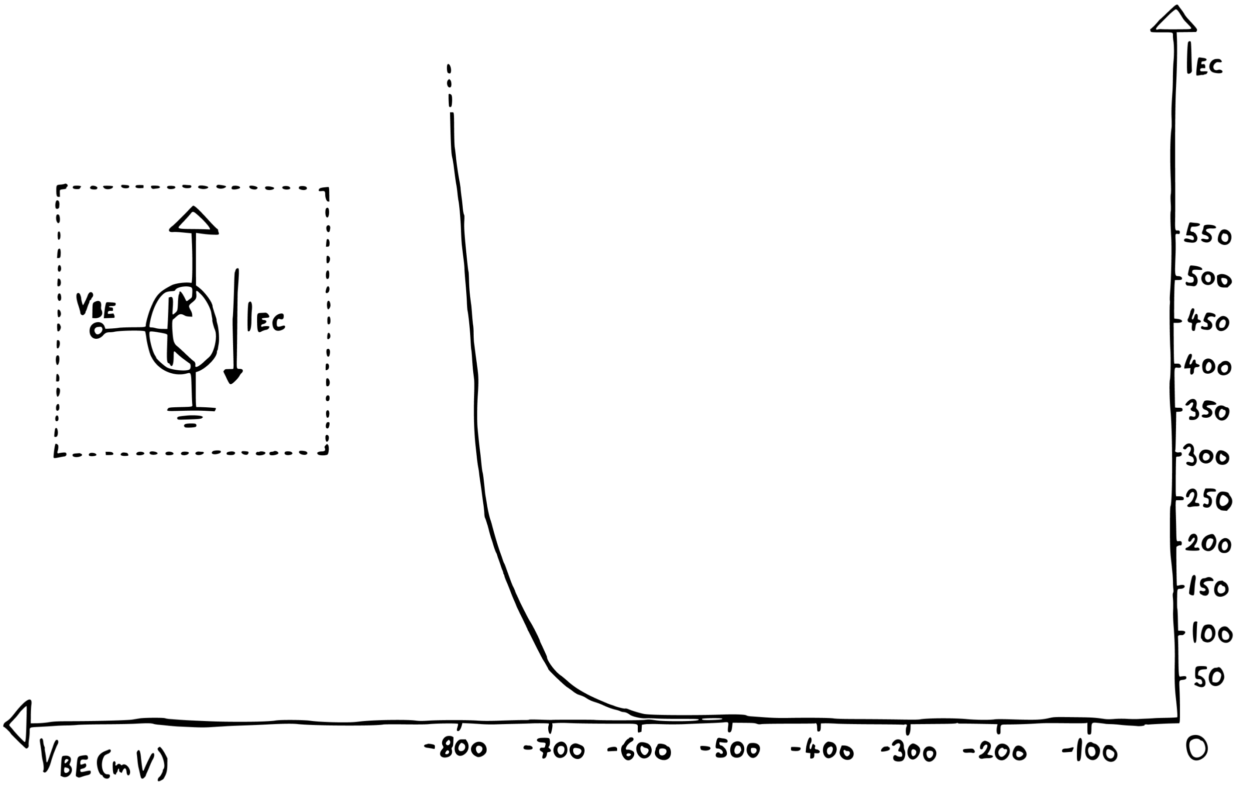
---

12 Please note that the pinout shown here only applies for the BC series of transistors. Others, like the 2N series, allocate their pins differently.

13 The voltage is measured between base and emitter. So „a small voltage“ effectively means a small voltage **difference** between base and emitter!

collector current. (At least not without some unwanted side effects.) Third, also unlike a resistor, a BJT is not a linear device. Meaning that a change in collector voltage will not affect the collector current. And fourth, the collector current is affected by the transistor's temperature! The more it heats up, the more current will flow.

Now, for the PNP transistor, all of the above applies, too – except for two little details. Unlike with the NPN, **the PNP transistor decreases its collector current when the voltage at its base increases**[14]. So you have to bring the base voltage below the emitter to open the transistor up. Also, that collector current flows out of, not into the collector!

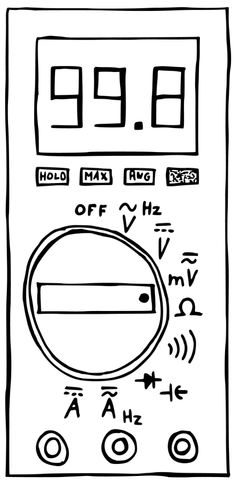


[14] Again, the voltage is measured between base and emitter.

TOOLS APPENDIX

There are two types of tools that will help you tremendously while designing a circuit: multimeters and oscilloscopes. In this appendix, we'll take a quick look at each of these and explore how to use them.

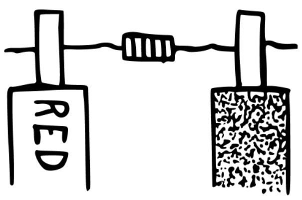
MULTIMETERS

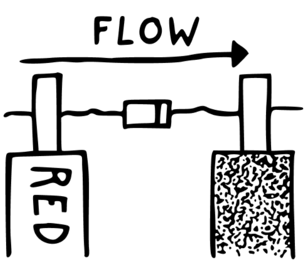


Multimeters come in different shapes and sizes, but the most common type is probably the hand-held, battery powered variant. It can measure a bunch of different things: voltage, current, resistance, continuity. Some have additional capabilities, allowing you to check capacitance, oscillation frequency or the forward voltage drop of a diode.

When shopping for one, you'll probably notice that there are really expensive models boasting about being TRUE RMS multimeters. For our purposes, this is really kind of irrelevant, so don't feel bad about going for a cheap model!

Using a multimeter is actually really straightforward. Simply attach two probes to your device – the one with a black cable traditionally plugs into the middle, while the red one goes into the right connector. Next, find whatever you want to measure and select the corresponding mode setting.



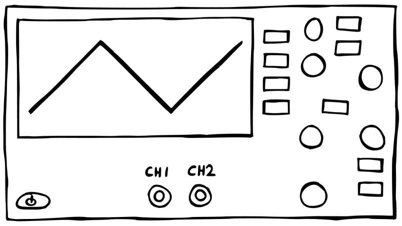


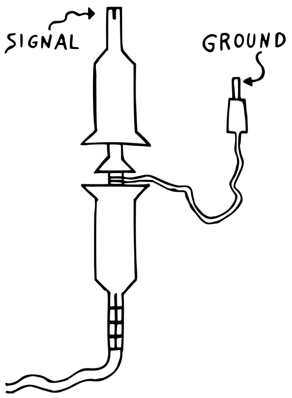
In some cases, it doesn't matter which probe you connect to which component leg or point in your circuit. This is true for testing resistors, non-polarized capacitors (foil/film, ceramic, teflon, glass etc.), continuity[15] or AC voltage.

In others, you'll have to be careful about which probe you connect where. For testing the forward voltage drop of a diode, for example, **the multimeter tries to push a current from the red to the black probe**. Here, you'll have to make sure the diode is oriented correctly, so that it doesn't block that current from flowing. For testing a DC voltage, you want to make sure the black probe is connected to ground, while you use the red one to actually take your measurement.

[15] Just a fancy word for saying that two points are electrically connected.

OSCILLOSCOPES



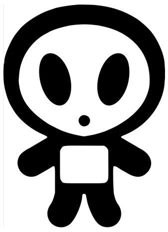


While multimeters are fairly cheap and compact, oscilloscopes are usually somewhat pricey and bulky. **If you're willing to make the investment, they are a huge help with the troubleshooting process, though**. Using one is, again, surprisingly straightforward – if you manage to work your way through the sometimes quite convoluted UI, especially on digital models.

To start using your scope, simply attach a probe to one of the channel inputs. These probes usually have two connectors on the other end: a big one that you operate by pulling the top part back – and a smaller one, which is usually a standard alligator clip. The latter needs to be connected to your circuit's ground rail, while you probe your oscillation with the former. Now what the oscilloscope will do is **monitor the voltage between the two connectors over time and draw it onto the screen as a graph**. Here, the x-axis is showing time, while the y-axis is showing voltage. You can use the device's scaling controls to zoom in on a specific part of your waveform.

Usually, digital oscilloscopes will also tell you a couple useful things about the signal you're currently viewing: minimum/maximum voltage level, oscillation frequency, signal offset. Some even offer a spectrum analyzer, which can be useful to check the frequencies contained in your signal.

BUILD GUIDE



MODULE ASSEMBLY APPENDIX

Before we start building, let's take a look at the complete **mki x es.edu BBD** schematics (see next page) that were used for the final module's design and PCB fabrication. Most components on the production schematics have denominations (a name – like R1, C1, VT1, VD1, etc.) and values next to them. Denominations help identify each component on the PCB, which is particularly useful during **calibration, modification** or **troubleshooting**.

XS1 is the **TIME CV input** jack socket, **XS2** is the unique feature – **INHIBIT CV** jack socket and **XS3** is the **AUDIO INPUT** jack socket, **XS4** is the **WET OUTPUT** and **XS5** is the **MIX OUTPUT** jack socket – these are the very same we've already been using on the breadboard for interfacing with other devices. In our designs, we use eurorack standard 3,5mm jack sockets (part number WQP-PJ301M-12).

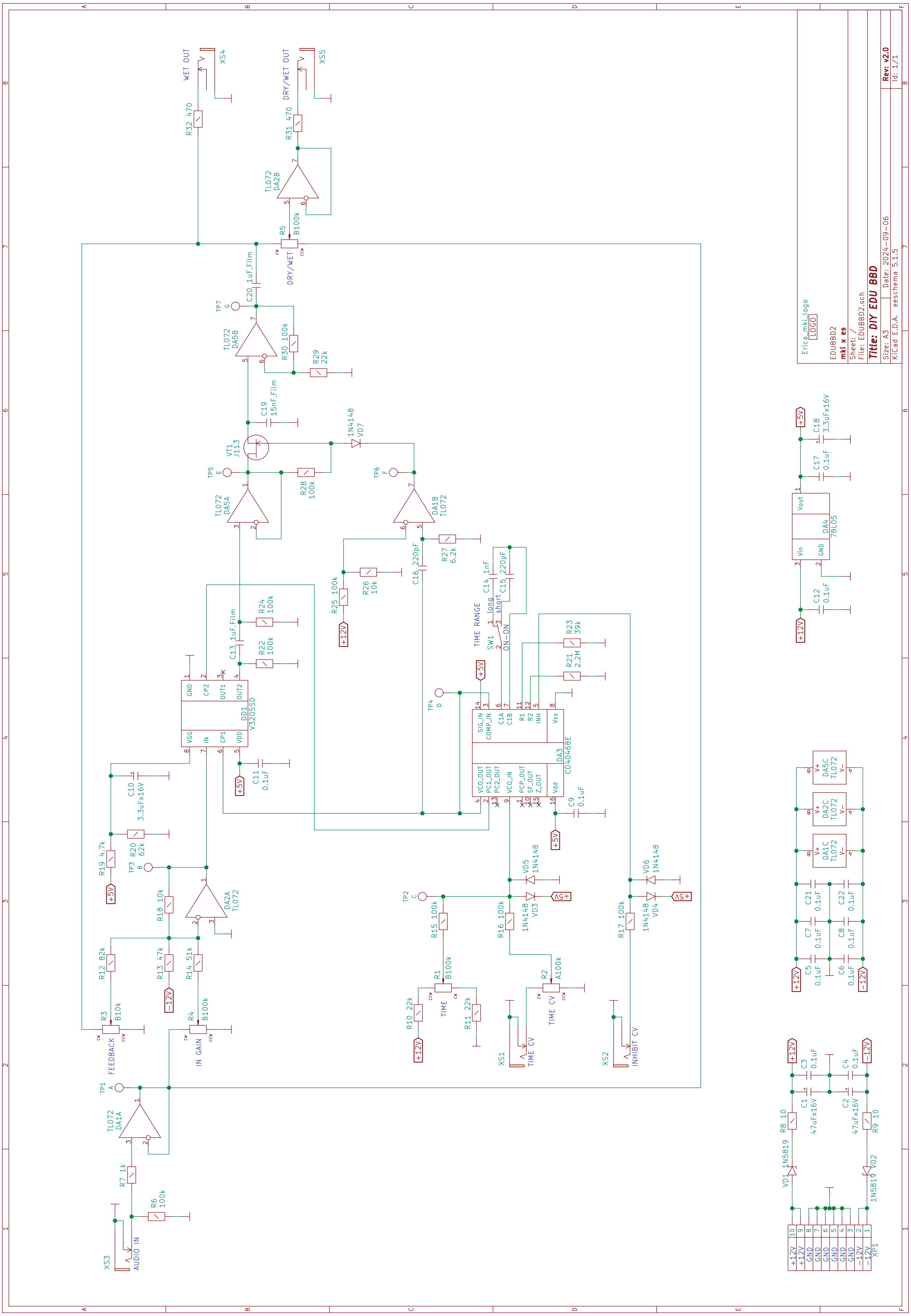
XP1 is a standard eurorack **power connector**. It's a 2x5 male pin header with a key (the black plastic shroud around the pins) to prevent accidental reverse polarity power supply connection. This is necessary because connecting the power incorrectly will permanently damage the module.

VD1 and **VD2** are **schottky diodes** that double-secure the reverse polarity power supply protection. Diodes pass current only in one direction. Because the anode of VD2 is connected to +12 V on our power header, it'll only conduct if the connector is plugged in correctly. If a negative voltage is accidentally applied to the anode of VD1, it closes, and no current passes through. The same goes for VD2, which is connected to -12 V. Because schottky diodes have a low forward voltage drop, they are the most efficient choice for applications like this.

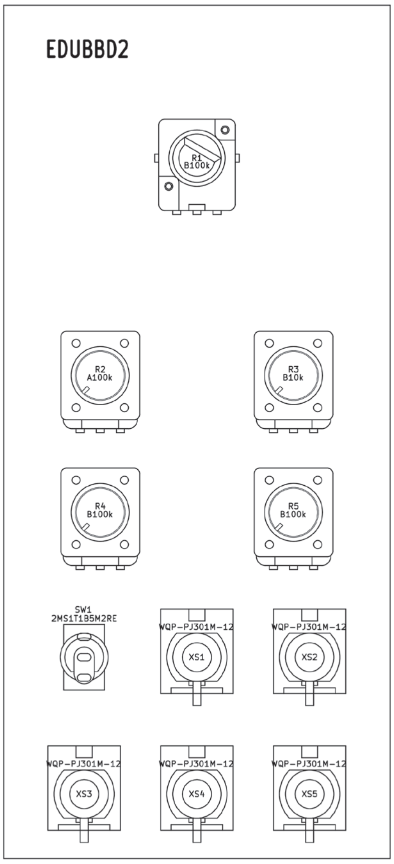
Next, we have two **10 Ohm resistors (R8** and **R9)** on the + and – 12 V rails, with **decoupling** (or **bypass-**) capacitors **C3** and **C4**. These capacitors serve as energy reservoirs that keep the module's internal supply voltages stable in case there are any fluctuations in the power supply of the entire modular system. In combination with R5 and R6, the large 47 microfarad pair (**C1** and **C2**) compensates for low frequency fluctuations, while C3 and C4 filter out radio frequencies, high frequency spikes from switching power supplies and quick spikes created by other modules. Often another component – a **ferrite bead** – is used instead of a 10 Ohm resistor and there's no clear consensus among electronic designers which works best, but generally for analogue modules that work mostly in the audio frequency range (as opposed to digital ones that use microcontrollers running at 8 MHz frequencies and above), resistors are considered to be superior.

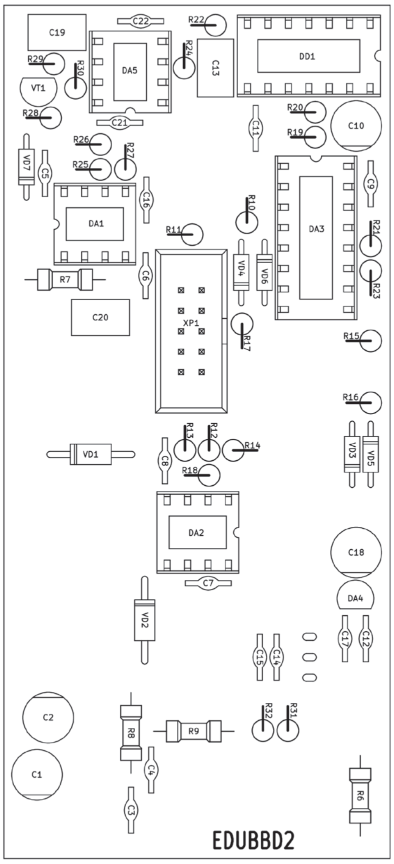
Another advantage of 10 Ohm resistors is that they will act like **slow "fuses**" in case there's an accidental short circuit somewhere on the PCB, or an integrated circuit (IC) is inserted backwards into a DIP socket. The resistor will get hot, begin smoking and finally break the connection. Even though they aren't really fuses, just having them there as fuse substitutes is pretty useful - **you'd rather lose a cent on a destroyed resistor than a few euros on destroyed ICs.**

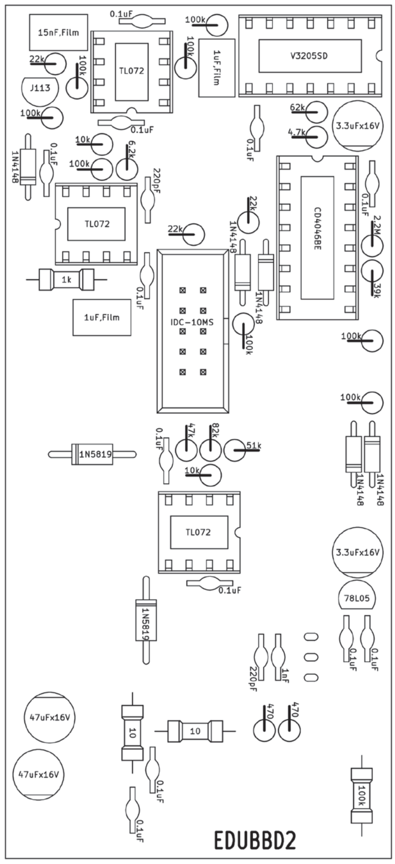
Capacitors **C5 – C8** and **C21, C22** are additional decoupling capacitors. If you inspect the PCB, you'll see that these are placed as close to the power supply pins of the ICs as possible. For well-designed, larger PCBs you will find decoupling capacitors next to each IC. Like the others, their job is to simply compensate for any unwanted noise in the supply rails. If the input voltage drops, then these capacitors will be able to bridge the gap to keep the voltage at the IC stable. And vice-versa - if the voltage increases, then they'll be able to absorb the excess energy trying to flow through to the IC, which again keeps the voltage stable. Typically, 0.1 uF capacitors are used for this purpose.

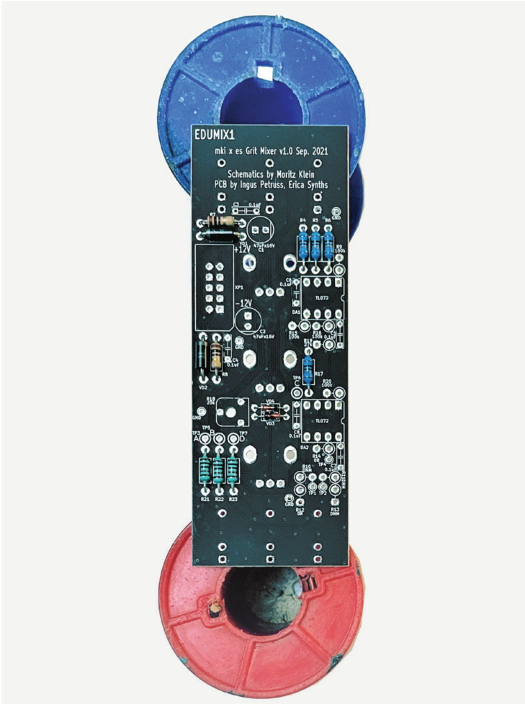

Title: DIY EDU BBD
Rev: v2.0
Size: A3
Date: 2024-09-06
KiCad E.D.A. eeschema 5.1.5
Id: 1/1
File: EDUBBD2.sch
Sheet: /
mki x es
EDUBBD2

Before you start soldering, we highly recommend printing out the following part placement diagrams with designators and values. Because some of our PCBs are rather densely populated, this will help you to avoid mistakes in the build process.

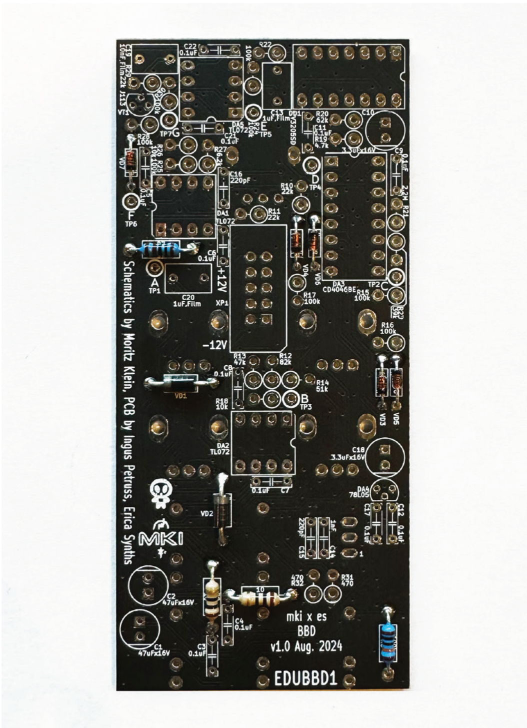




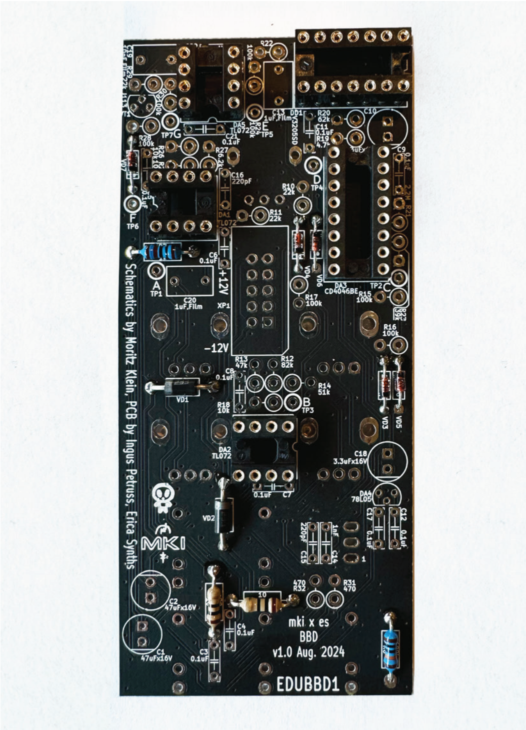




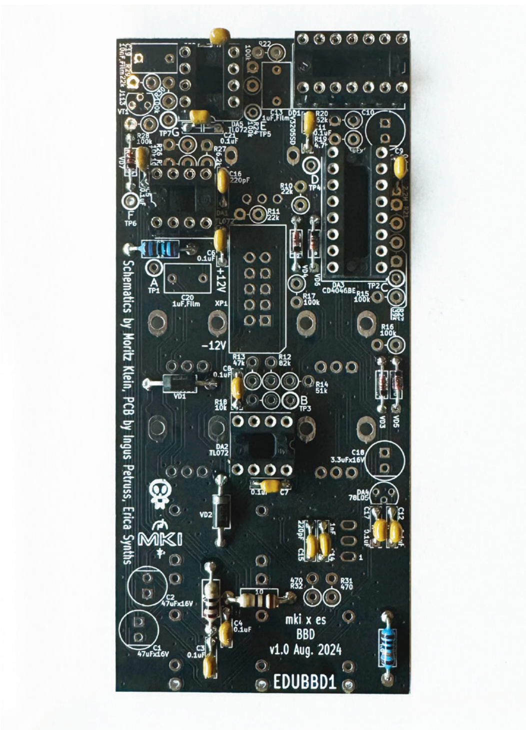
Place the BBD PCB in a PCB holder for soldering or simply on top of some spacers (I use two empty solder wire coils here).



I usually start populating PCBs with lower, horizontally placed components. In this case, these are **few resistors, switching diodes** and the **power protection diodes.** Bend the resistor leads and insert them in the relevant places according to the part placement diagram above. All components on the PCB have both their value and denomination printed onto the silkscreen. If you are not sure about a resistor's value, use a multimeter to double-check. Next, insert the diodes. Remember – **when inserting the diodes, orientation is critical!** A thick white stripe on the PCB indicates the cathode of a diode – match it with the stripe on the component. Flip the PCB over and solder all components. Then, use pliers to cut off the excess leads.

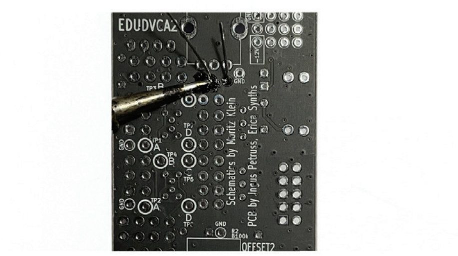


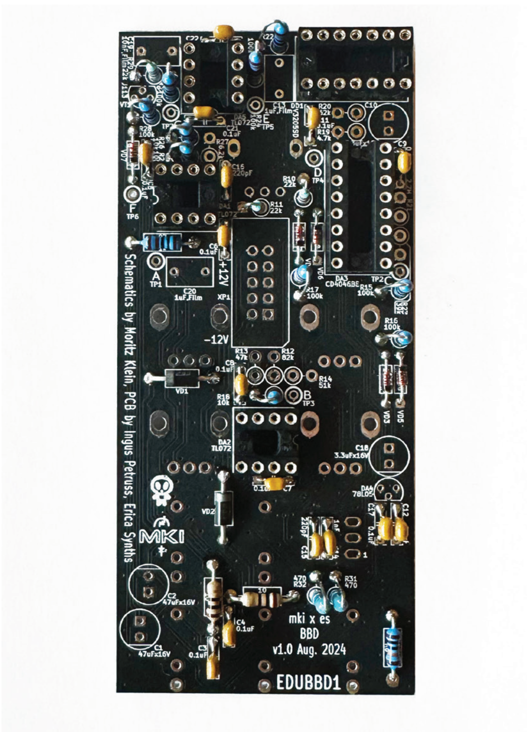
Next, insert the first DIP socket, hold it in place and solder one of the pins. Continue with the **next DIP socket.** Make sure the DIP sockets **are oriented correctly** – the notch on the socket should match the notch on the PCB's silkscreen. Now, turn the PCB around and solder all remaining pins of the DIP sockets.



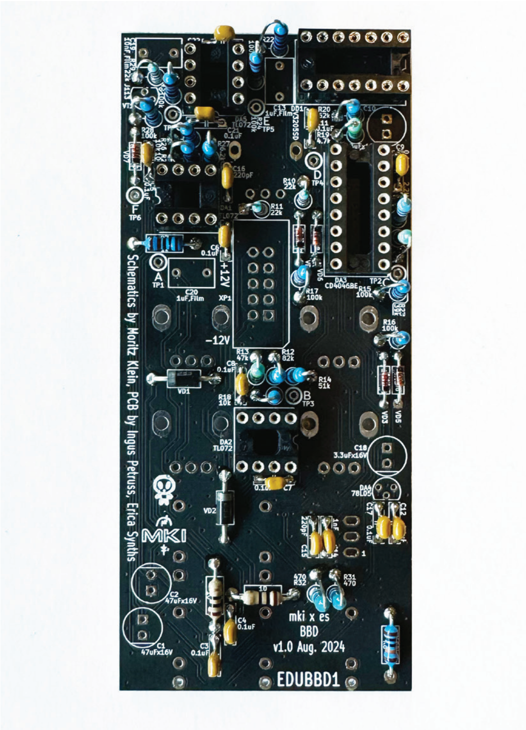
Then proceed with the **ceramic capacitors.** In order to avoid mistakes, first sort the capacitors by their values. Then place the PCB in your PCB holder or on spacers, insert the capacitors and solder them like you did with the resistors & diodes before. Now your PCB should look like this:

In order to save space on the PCB, some of our projects, including the BBD, have **vertically placed resistors.** The next step is to place & solder those. Let’s start with 100k, 22k, 10k and 470ohm resistors. Bend a resistor’s legs so that its body is aligned with both legs and insert it in its designated spot. Then solder the longer lead from the top side of the PCB to secure it in place, turn the PCB around and solder the other lead from the bottom. You can insert several resistors at once. Once done with soldering, use pliers to cut off excess leads.



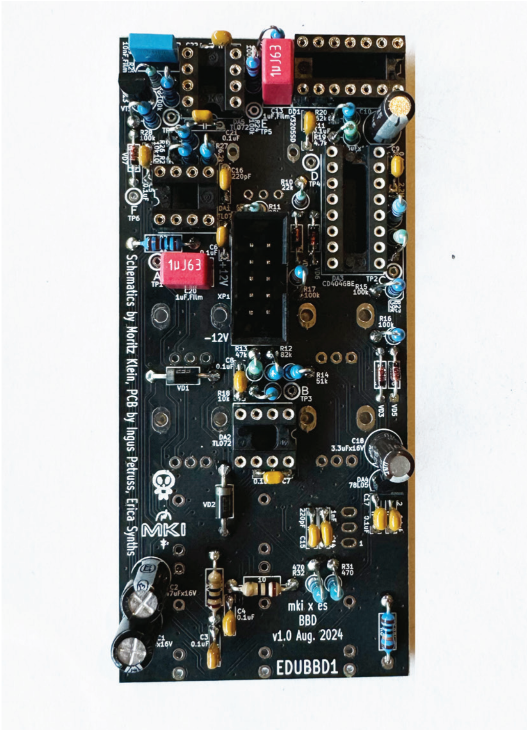


Once you are done with soldering the first batch of resistors, your PCB should look like this:

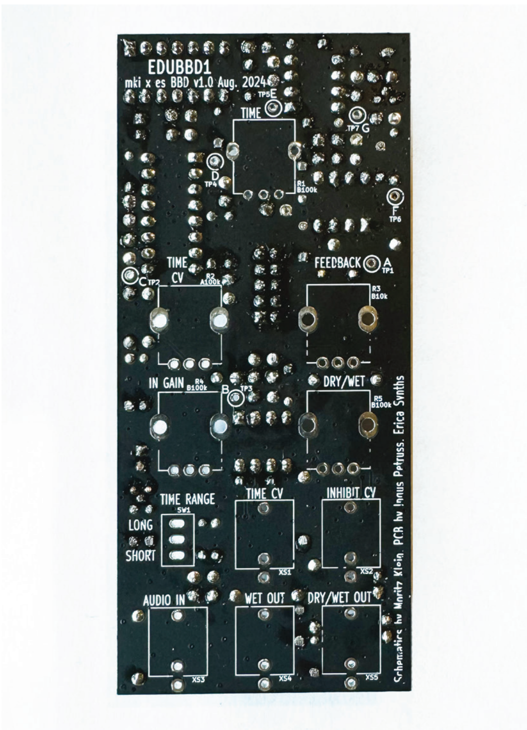


Now, proceed with remaining resistors.

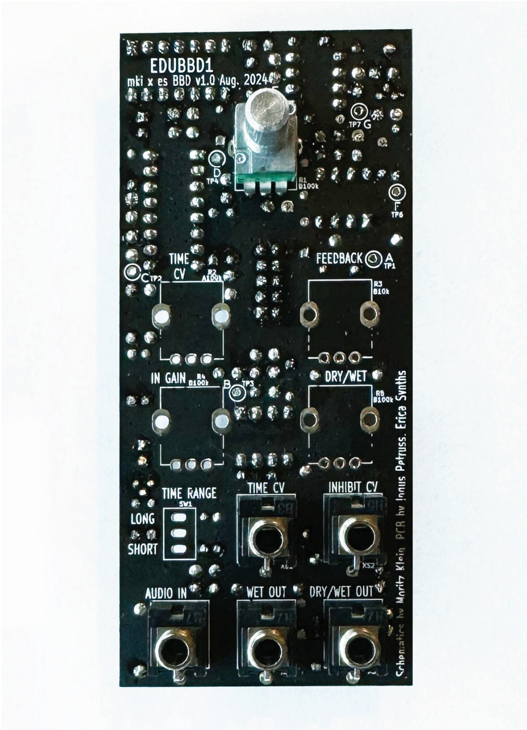
Next up: inserting & soldering **the 5V voltage regulator the transistor.** Make sure you align those components in TO-92 case with the marked outline on the silkscreen – **orientation is critically important here.** Also, insert **film capacitors** and solder them.
Next, insert & solder **the electrolytic capacitors.** Electrolytic capacitors are bipolar, and you need to mind their orientation. The positive lead of each electrolytic capacitor is longer, and there is a minus stripe on the side of the capacitor's body to indicate the negative lead. On our PCBs, the positive pad for the capacitor has a square shape, and the negative lead should go into the pad next to the notch on the silkscreen.



Then complete the component side of the BBD PCB by soldering the **PSU socket.** Make sure the orientation of the socket is as shown in the picture below – the arrow pointing to the first pin is aligned with a notch on the silkscreen. The key on the socket will be facing inwards towards the PCB. Now your PCB should look like this:

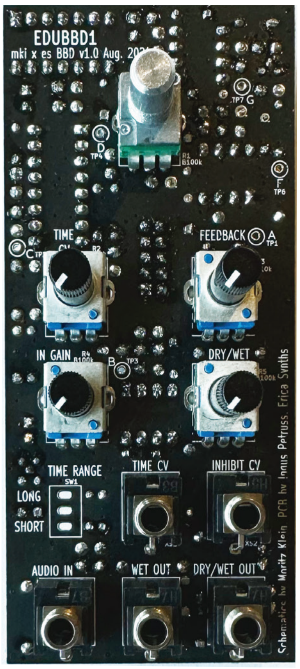


Now, turn the PCB around and inspect your solder joints. **Make sure all components are soldered properly and there are no cold solder joints or accidental shorts.** Clean the PCB to remove extra flux, if necessary.

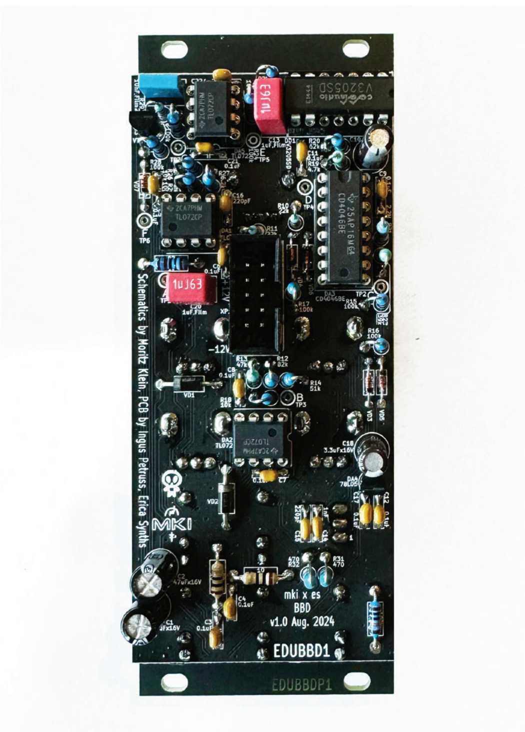


Insert the **jack sockets and the green B100k potentiometer** and solder them.

Insert other potentiometers, but don't solder them yet! Fit the front panel and make sure that the potentiometer shafts are aligned with the holes in the panel – and that they're able to rotate freely. Now, go ahead and solder the potentiometers.



The **LONG/SHORT switch** requires special attention. Insert the switch in the relevant place on the PCB, place the front panel, fix it with few nuts on the potentiometer and jack sockets, then fix the switch with its nut (do not overtighten it) and then **solder the switch.**



Now, **insert the ICs into their respective DIP sockets.** Mind the orientation of the ICs – match the notch on each IC with the one on its socket.

To complete the module install all nuts on the jack sockets, tighten them and install the knob on the potentiometer.

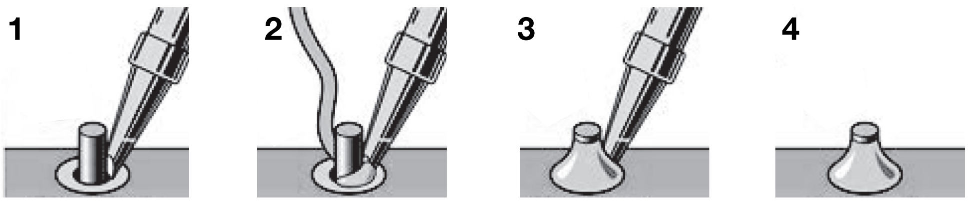
Congratulations! **You have completed the assembly of the mki x es.edu BBD module!** It does not need any calibration and, if assembly is correct, it should work straight away. Connect it to your eurorack power supply and switch it on. If there's no "magic smoke", it's a good sign that your build was successful. To test it, apply some pulsating audio (our DIY Snare Drum will work great) to the audio in and tweak the TIME and FEEDBACK knobs to appreciate the delay effect.

Enjoy!

SOLDERING APPENDIX

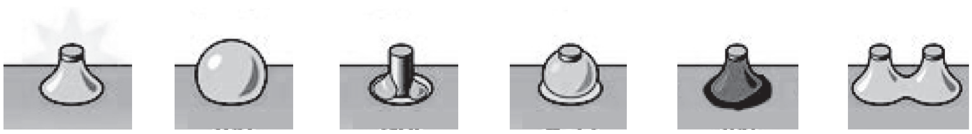
If you've never soldered before – or if your skills have become rusty – it's probably wise to check out some **THT** (through-hole technology) **soldering tutorials on YouTube**. The main thing you have to remember while soldering is that melted solder will flow towards higher temperature areas. So you need to make sure you apply equal heat to the component you are soldering and the solder pad on the PCB. The pad will typically absorb more heat (especially ground-connected pads which have more thermal mass), so keep your soldering iron closer to the pad on the PCB. It's critically important to dial in the right temperature on your soldering station. I found that about 320 °C is the optimal temperature for most of parts, while for larger elements like potentiometers and sockets, you may want to increase that temperature to **370 °C.**

Here's the recommended soldering sequence:



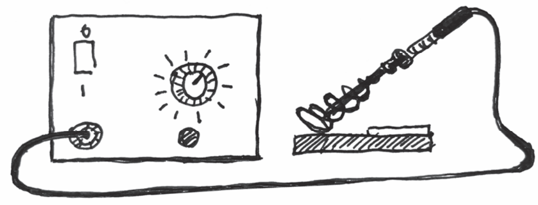
Heat part and pad 2 - 3 sec | Add solder | Continue heating 1 -2 sec. | Let cool

After you have completed soldering, inspect the solder joint:

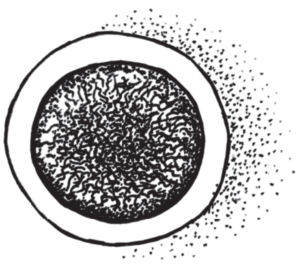


Perfect | Too much solder | Not enough solder | Cold joint | Too much heat | Short

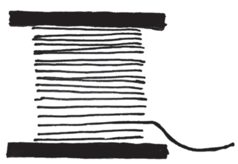
DIY electronics is a great (and quite addictive) hobby, therefore we highly recommend you invest in good tools. In order to really enjoy soldering, you'll need:



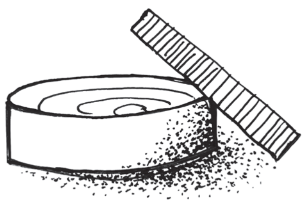
A decent soldering station. Top-of-the-line soldering stations (brands like Weller) will cost 200€ and above, but cheaper alternatives around 50€ are often good enough. Make sure your soldering station of choice comes with multiple differently-sized soldering iron tips. The most useful ones for DIY electronics are flat, 2mm wide tips.



When heated up, the tips of soldering irons tend to oxidize. As a result, solder won't stick to them, so you'll need to clean your tip frequently. Most soldering stations come with a **damp sponge for cleaning the iron tips** – but there are also professional solder tip cleaners with **golden curls** (not really gold, so not as expensive as it sounds). These work much better because they do not cool down the iron.

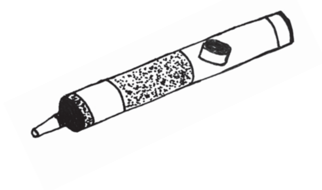


Solder wire with flux. I find 0,7mm solder wire works best for DIY projects.

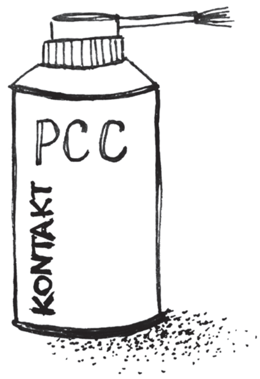


Some **soldering flux** paste or pen will be useful as well.

Cutting pliers. Use them to cut off excess component leads after soldering.



A solder suction pump. No matter how refined your soldering skills are, you will make mistakes. So when you'll inevitably need to de-solder components, you will also need to remove any remaining solder from the solder pads in order to insert new components.



Once you have finished soldering your PCB, it's recommended to remove excess flux from the solder joints. **A PCB cleaner** is the best way to go.

All of these tools can be found on major electronic components retailer websites, like Mouser, Farnell and at your local electronics shops. As you work your way towards more and more advanced projects, you'll need to expand your skillset and your tool belt – but the gratification will be much greater.

"We keep moving forward, opening new doors, and doing new things, because we're curious and curiosity keeps leading us down new paths."

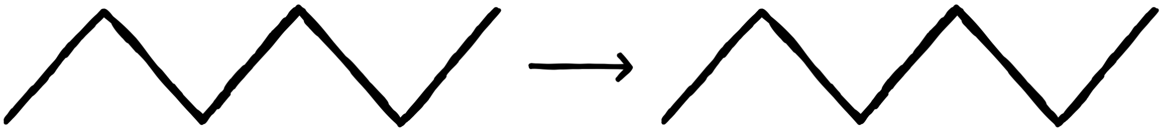
– Walt Disney

TROUBLSHOOTING

Your assembled module might not immediately work as expected. This could be due to many factors: cold solder joints, faulty, missing or misplaced components – or you might've even missed some solder joints entirely. **To diagnose this, we've exposed multiple test points on the PCB**. Checking these with your oscilloscope can give you some indication as to what went wrong with your build.

TP1 (A)

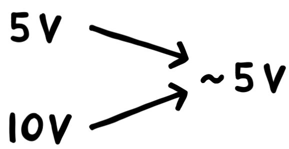
The signal at this test point should be identical to the signal you feed into the BBD's input. It is only routed through an op amp buffer.



Any signal differences between input and the test point would indicate that either the audio socket **XS3**, the resistor **R7**, or the op amp **DA1A** are faulty or not properly soldered to the PCB.

TP2 (C)

The voltage at this test point should be the combination of internal and the external time CV. It should never go above 5V or below 0V, no matter what external voltage you apply. If it does, the diodes **VD3** or **VD5** might be faulty or not properly soldered to the PCB.



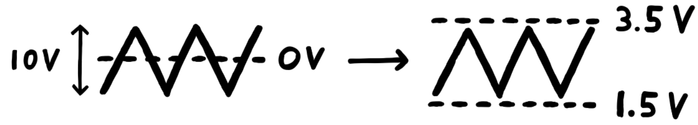
Turning the TIME potentiometer should result in a changed voltage at this test point. If it doesn't, the potentiometer **R1** or the resistors **R10** and **R15** might be faulty or not properly soldered to the PCB.

If the test point voltage is fixed at ~5V regardless of what you do with the TIME potentiometer, resistor **R11** might be faulty or not properly soldered to the PCB.

Applying an external time CV should result in a changed voltage at this test point. If it doesn't, the potentiometer **R2** or the resistor **R16** might be faulty or not properly soldered to the PCB.

TP3 (B)

If the FEEDBACK pot is dialed all the way down and the IN GAIN pot is set to a non-zero value, the signal at this test point should be identical to the input signal, but scaled down and biased above the 0V line.



If you don't see any signal, the potentiometer **R4**, the resistors **R14** and **R18**, or the op amp **DA2A** might be faulty or not properly soldered to the PCB.

If the signal is not biased above the 0V line, resistor **R13** might be faulty or not properly soldered to the PCB.

You should be able to change the signal's gain by turning the IN GAIN potentiometer.

Turning the FEEDBACK potentiometer should alter the signal noticeably. If nothing happens, the potentiometer **R3** or resistor **R12** might be faulty or not properly soldered to the PCB. More things might be going wrong as well, though. Check the other test points, too.

TP4 (D)

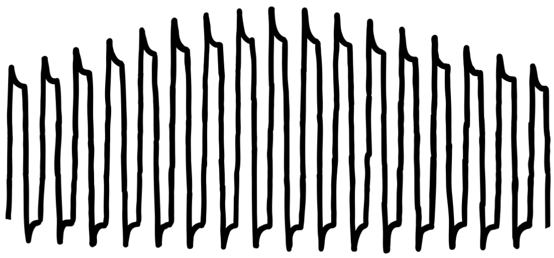
You should see a high frequency clock signal (square wave) at this point. Its frequency should change if the tune CV (TP2) changes.

If you don't get any signal, chip **DA3** might be faulty or not properly soldered to the PCB. Also, resistors **R21** and **R23** could cause trouble – so check these, too.

Flipping the TIME RANGE switch should significantly change the signal's frequency. If it doesn't, switch **SW1** or capacitors **C14** and **C15** might be faulty or not properly soldered to the PCB.

TP5 (E)

The voltage at this test point should constantly jump from a low to a high voltage, in sync with the clock signal (TP4). The low voltage should always be the same, while the high voltage should fluctuate a bit.



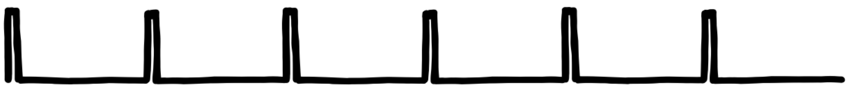
If you don't get any signal at all, the op amp **DA5A**, capacitor **C13**, resistor **R19**, or chip **DD1** might be faulty or not properly soldered to the PCB. Also, check the clock signal at TP4.

If you get the jumping voltage, but the high voltage is not changing at all, make sure you get the correctly biased and scaled input at TP3.

The signal should be centered around the 0V line. If it has a DC offset, resistor **R24** might be faulty or not properly soldered to the PCB.

TP6 (F)

The signal at this test point should be a very narrow pulse wave. The frequency should be identical to that of the clock signal at TP4.

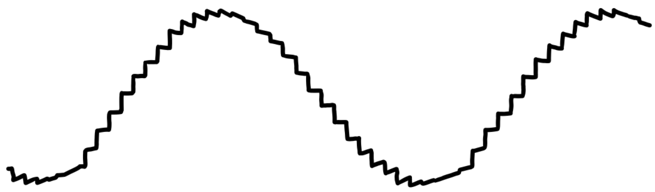


If you don't get any signal, the op amp **DA1B**, or the capacitor **C16** might be faulty or not properly soldered to the PCB.

If the pulse wave looks malformed or more like a square wave, resistors **R25**, **R26** and **R27** might be faulty or not properly soldered to the PCB.

TP7 (G)

If the FEEDBACK pot is dialed all the way down, the signal at this test point should look very similar to the input signal – though it should be stair stepped. (This should become more noticeable as you turn down the clock speed.)



If you don't get any signal at all (even though TP5 and TP6 look fine), the op amp **DA5B**, transistor **VT1** or diode **VD7** might be faulty or not properly soldered to the PCB.

If the signal does not really resemble the input or is very spiky, the capacitor **C19** might be faulty or not properly soldered to the PCB.

If the signal is very quiet, resistor **R29** might be faulty or not properly soldered to the PCB.

If the signal is extremely overdriven, resistor **R30** might be faulty or not properly soldered to the PCB.