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 on two **breadboards**[1]**,** which are non-permanent circuit prototyping tools that allow 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 most 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!

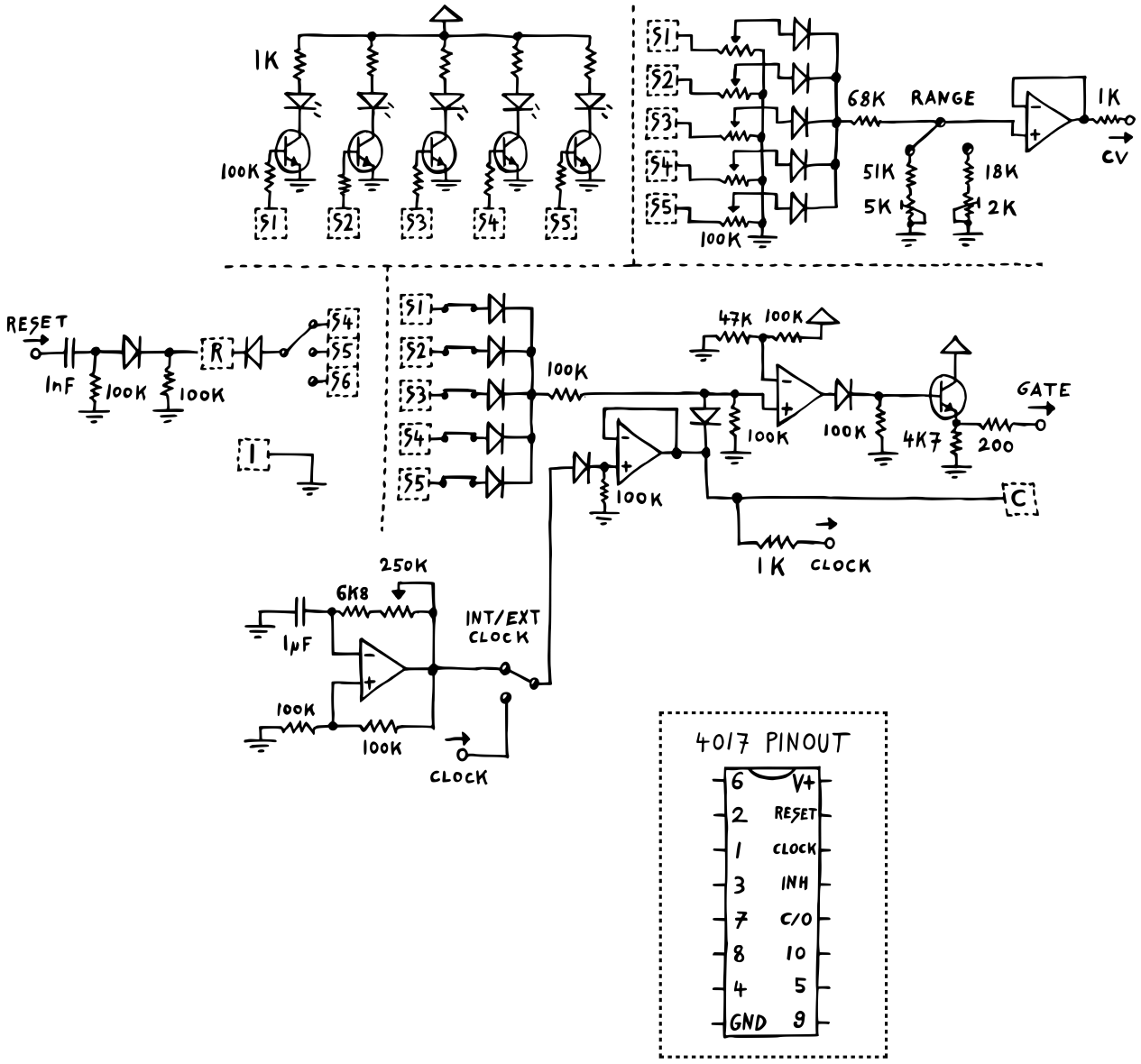
TABLE OF CONTENTS

CIRCUIT SCHEMATIC 2
BILL OF MATERIALS 3
POWERING YOUR BREADBOARDS 6
CIRCUIT DESIGN CLOSE-UP 7
COMPONENTS & CONCEPTS APPENDIX 34
TOOLS APPENDIX 47
MODULE ASSEMBLY APPENDIX 50
SOLDERING APPENDIX 63

[1] Note that there are no breadboards included in this kit! You will also need a pack of jumper wires and two 9 V batteries with clips. These things are cheap & easy to find in your local electronics shop.

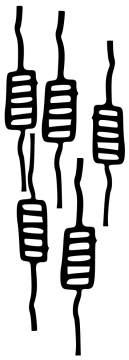
THE mki x es.edu SEQUENCER

If you're new to DIY synthesizer building, you might've run into this issue: you just set up your first VCO on a breadboard, spent a good long while troubleshooting the stubborn thing – and then you realize you have nothing to actually control it with. If this is you, no worries: here's a super simple five step sequencer I've designed that'll make your oscillator sing in no time.



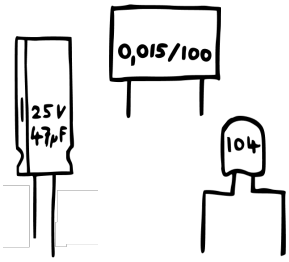
BILL OF MATERIALS

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:



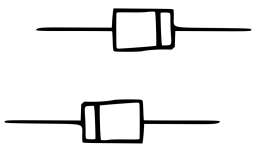
An array of resistors. The specific values (in ohms, which you should check for with a multimeter) are

100k x14
68k x1
51k x1
47k x1
18k x1
6k8 x1
4k7 x1
1k x7
200 x1
10 x2



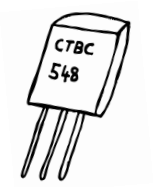
A bunch of capacitors. The specific values (which are printed onto their bodies) are

47 µF (electrolytic) x2
1 µf (1J/film) x1
100 nF (104/ceramic) x5
1 nF (102/ceramic) x1



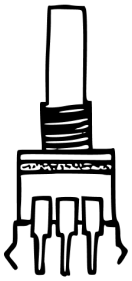
Lots of diodes. The specific model names (which are printed onto their bodies) are

SB140 (schottky) x2
1N4148 (signal) x15



A bunch of transistors. The specific model names (which are printed onto their bodies) are

2N3904 (NPN) x5
BC547 (NPN)[2] x1



A couple regular potentiometers. The specific values (which may be encoded & printed onto their bodies) are

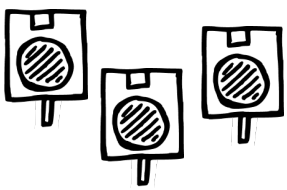
250k (B254) x1
100k (B104) x5

Some switches. The specific models (which you can identify by the number of connectors on their underside) are

Single pole, double throw x5
Single pole, triple throw x1

A few LEDs (light emitting diodes). The specific model (which you can identify by measuring their body's width) is

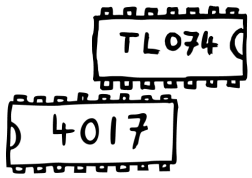
3mm (red) x5



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

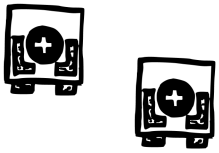
Switched mono (black) x5

[2] Please note that you might get a different, but functionally identical model (e.g. BC548) in your kit.



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

TL074 (quad op amp) x1
CD4017 (decade counter) x1



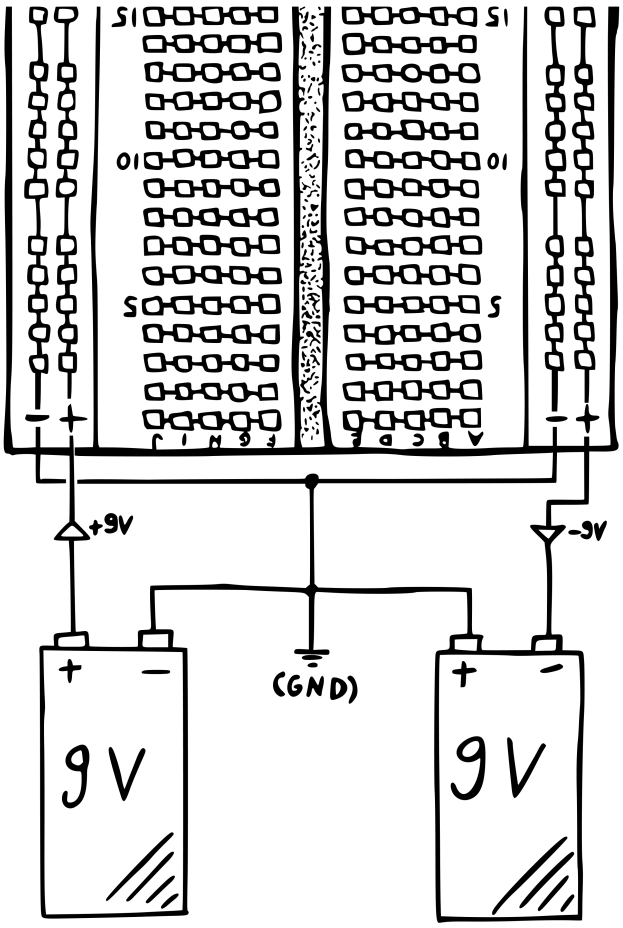
Some trimmer potentiometers. The specific values (which are printed onto their bodies) are

5k x1
2k x2

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

POWERING YOUR BREADBOARDS

Before we can start building, you'll need to find a way of providing your breadboards with power. Ideally, you'd use a dual power supply for this. 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 to one breadboard like shown here**. For this, you should use 9 V battery clips, which are cheap & widely available in every electronics shop.



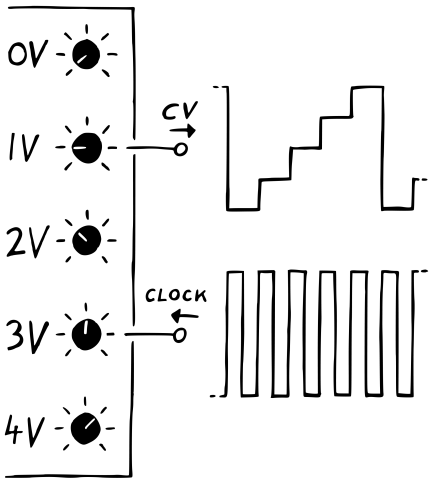
By connecting the batteries like this, the row on the left side labeled **+** becomes your positive rail, the row on the right side labeled **+** becomes your negative rail, and both rows labeled **–** become your ground rails.[3] Next, use jumper wires to route the rails to the second breadboard in the same configuration.

Please make sure you disconnect the batteries from your breadboards when you make changes to the circuit! Otherwise you run the risk of damaging components.

[3] This is a bit awkward because breadboards weren't really made with dual supply voltages in mind.

SEQUENCER BASICS

Before we dive into the circuit's design, let's first make sure we understand what a step sequencer actually does. For that, we'll have to separate two types of sequences: CV and gate. CV, if you don't know, stands for control voltage. A control voltage is a voltage that is used to control some parameter on a synthesizer module. An oscillator, for example, usually has multiple control voltage inputs. CV coming from a sequencer is often used to control an oscillator's pitch. The relation here is simple: the higher the control voltage, the higher the pitch.

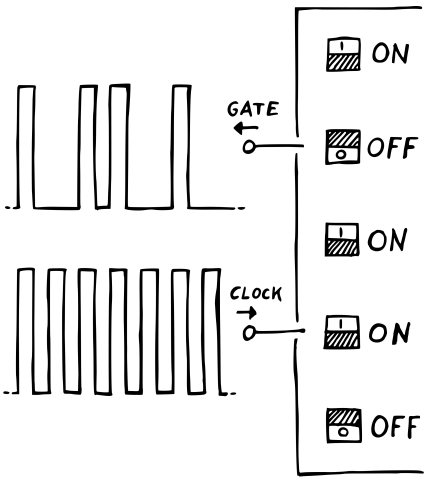


CV sequences are then really just strings of discrete voltage levels – a bit like sheet music for your synthesizer. Consider this example: here, we've got a five-step CV sequence, where the output voltage increases by 1 on each step.

A CV sequencer would then take this sequence of voltages and send them out one after the other via a single output. It's as if it was telling us the individual values in succession. (In data transmission theory, you'd call this serial communication.) Of course our sequencer still needs to know how fast it should step through the sequence. For that, we normally use something called a clock signal.

A clock signal is really just a square wave-oscillation, going on and off at a steady pace. Our sequencer would then proceed to the next step every time that clock signal goes from off to on (also called „low" to „high"). **And so the frequency of our clock signal determines the speed of our output sequence – i.e. how fast it switches from step to step.** After passing the last step, the sequencer would traditionally start over with the first value in the sequence, giving us an infinite five step loop. In our case, we'd get an output of „0 V, 1 V, 2 V, 3 V, 4 V" over and over.

So that's CV sequences. Gate sequences, on the other hand, are a bit simpler. A gate, if you don't know, is really just a voltage going from low to high and back low again. It's essentially a digital signal. To program a gate sequencer, you'd simply tell it wether you want a gate on a specific step – or not.

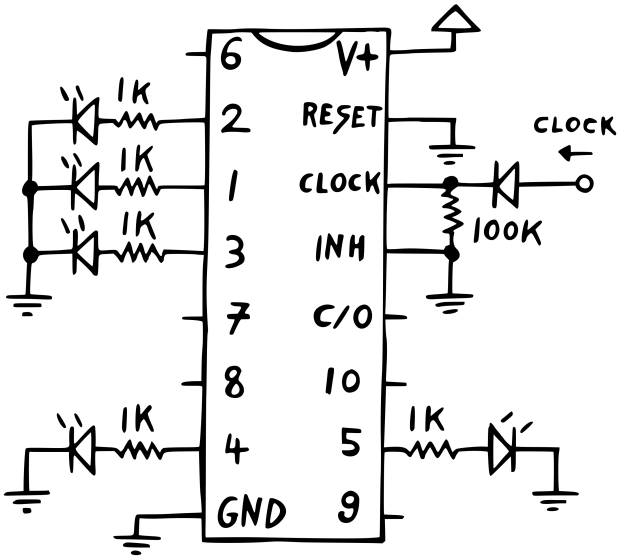


It will then go through these instructions, again advancing one step at a time, in sync with our clock signal. Every time it encounters a GATE ON, it'll send out a voltage pulse. Resulting in a gate sequence that looks something like this. **It's basically just a voltage that's pulsing rhythmically**.

In a typical patch, you would use this to drive an envelope generator, which in turn would drive a VCA and/or a filter. Allowing you to add rhythm and dynamics to your melodies. Now ideally, you'd want your sequencer to be able to send out both types of sequences at once – so that it's able to provide both melody and rhythm simultaneously. And even though that might sound somewhat complex, it's actually reasonably easy to achieve.

COUNTING WITH CHIPS

To do it, we'll use an old favorite from the synth DIY scene: the CD4017. It's a simple decade counter chip – which means that it's able to count from 1 to 10. For this, it has ten count state outputs and a clock input.



Here's how it works. When we fire the chip up, the first count state output will be on – meaning that it sends out a high level voltage, while all others will be off – meaning that they're sitting at ground level. This tells us that the current count is 1. Now whenever the signal we apply to the clock input goes from low to high, the currently active count state output will turn off – and the next one will turn on. Meaning that the count has increased by 1. **So the chip is basically just turning on its count state outputs one by one in sync with the clock signal**.

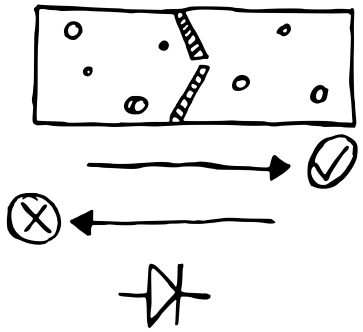
For this to work, there's one additional input we'll need to pay attention to, though – called the clock inhibit pin. Fortunately, all we have to do is tie it to ground permanently.[4] This is necessary because the chip will only listen for a clock signal if that pin is sitting at a low level voltage. If we give it a high level voltage, it'll tell the chip to ignore anything coming in through the clock input.

Speaking of the clock input, you might ask what the diode and resistor are doing here. Didn't we say that the chip is simply expecting a signal that goes on and off in regular intervals? If that's true, why can't we just feed it a square wave-oscillation from an oscillator? Simple: because oscillators usually send out a signal that is centered around the 0 V-line. Meaning that it swings between a positive and a negative voltage.

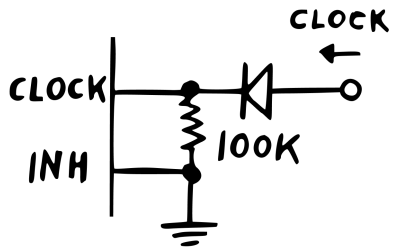
The problem with this is that many chips really don't like to receive any voltage levels that are outside of the ones we supply them with. In our case, we're powering our CD4017

[4] Along with the reset pin, just to keep it from floating. We'll talk about it in detail in a later chapter.

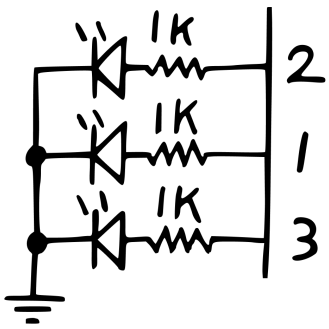
with +12 V and 0 V. So applying any voltage greater than 12 V or lower than 0 V to the chip's clock input is likely a bad idea.
In the worst case, it could even kill our chip completely. Which means we should protect it from those out-of-scope voltage levels. Thankfully, we don't really have to worry about voltages higher than 12 V – because that's the maximum voltage level in a typical eurorack system. So we'll only shield our chip from negative voltages.



To do that, we just need a bog standard diode. Diodes are basically like one way streets for electricity. They allow for current to flow in only one direction – which is indicated by the arrow in the diode symbol.[5] **If we set up the diode so that the arrow is pointing towards our clock input, only positive voltages will be allowed to pass through**. Negative voltages, on the other hand, will try to suck current out of the chip – which the diode will block.



Great! But then why the additional 100k resistor? Because without it, while the diode is blocking, the clock input pin would be floating, meaning that there'd be no defined voltage applied to it. This is a problem because floating inputs are prone to picking up noise and radio waves, which would cause the chip to behave in weird and unexpected ways. To avoid this, we use a pull-down resistor. Which is just a decently big resistor going to ground after our diode.
Here's how that works. Whenever the voltage coming from our clock signal is high, a small current is flowing through the diode and resistor and to ground. This is why the resistor should be relatively big: because we don't want to waste a lot of current here.
When the clock signal then swings low, the diode will block. But because the resistor is connecting the clock input to ground, it will be pulled down to ground level. **It's as if the resistor was setting a default voltage that gets overridden whenever the diode conducts**. So with this setup, we effectively chop off any part of our clock signal that might dip below the 0 V-line. Which should keep our chip safe and happy.

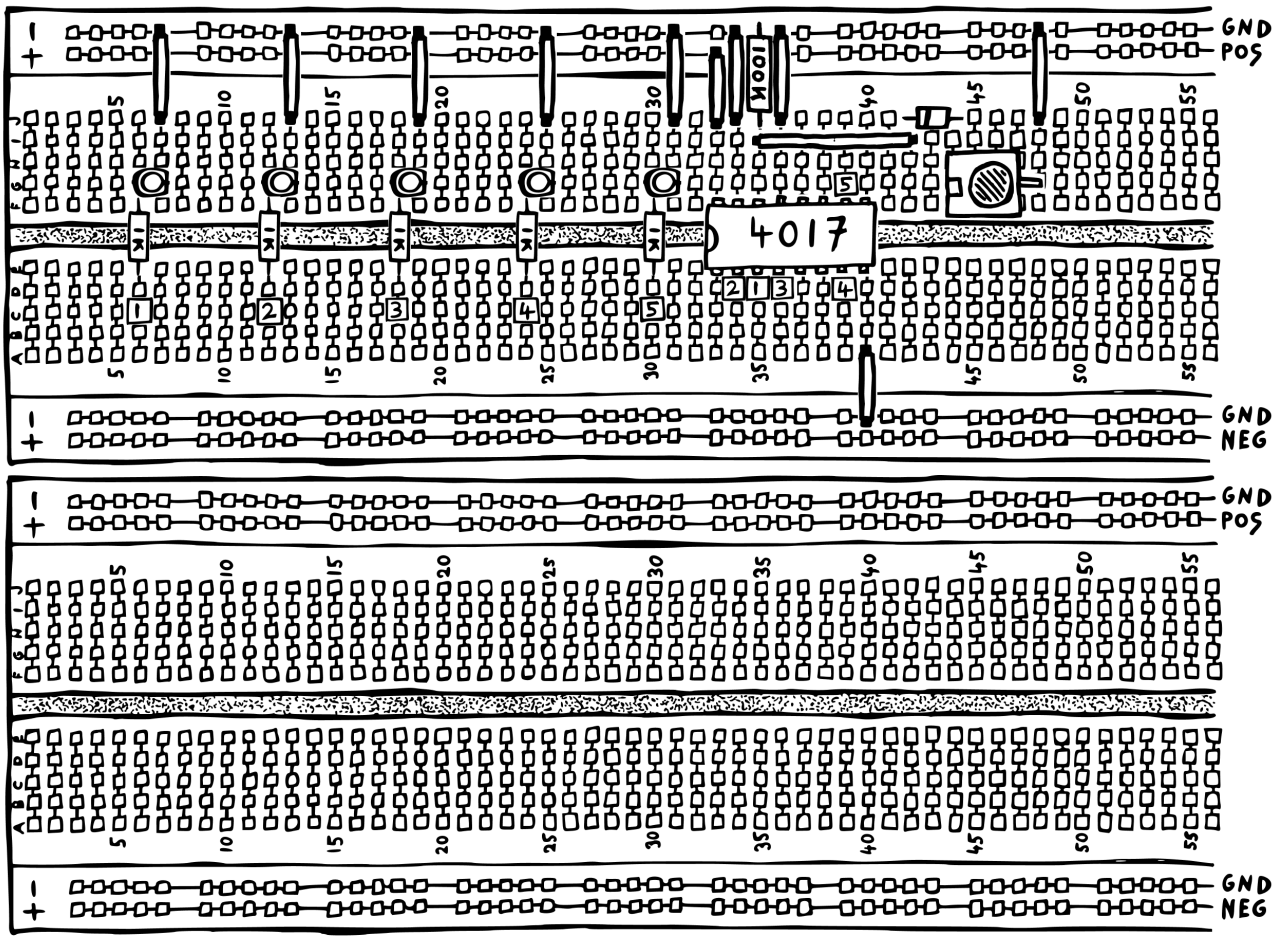


To try all this on a breadboard, we'll need to make the chip's internal state visible somehow. For this, we'll simply use a bunch of LEDs attached to the count state outputs through current limiting resistors. These are necessary because LEDs are super quick to burn out if we push too much current through them. Now, you might ask why we're setting up only five LEDs – when the chip has ten outputs in total?

[5] Read more about diodes in the components & concepts appendix (page 38).

Well, there's actually nothing stopping us from using all ten. **I simply made the decision to build a sequencer with five steps exactly – so I'm only interested in the first five outputs**. Assuming you're willing to go along with this, let's set up the circuit on a breadboard. You'll notice that a couple tie points are simply labeled with a number – this means that you should make the connection with long, flexible jumpers: 1 to 1, 2 to 2, 3 to 3 etc.

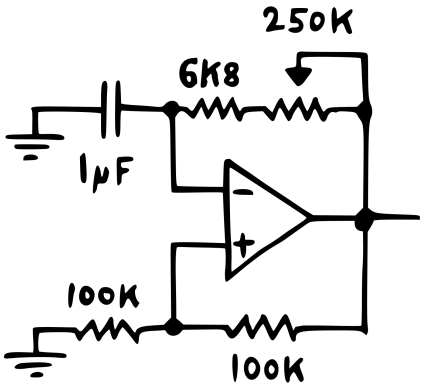
Also, mind the LED's orientation! They are essentially diodes, which means that they'll only let current flow through them in one direction: from anode to cathode. You can identify both by looking at the LED's legs – the longer leg is the anode, the shorter leg is the cathode. Additionally, the LED's body gives you further indication if you look at it from the top. One side is rounded (anode), the other looks almost cut-off (cathode).



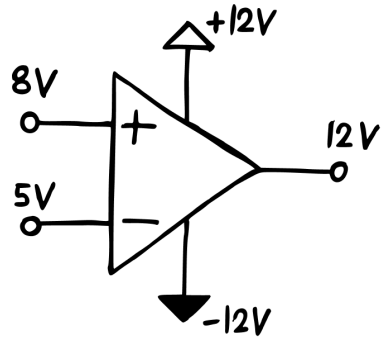
To test this, you'll need something like an LFO or a clock source. If you have one – great! Plug it into the clock input and see what happens. If you don't, no worries. Since our sequencer should have an internal clock anyways, we'll take slight detour and set one up right now.

THE CLOCK GENERATOR

To do that, we'll use just an op amp, a few resistors, a potentiometer and a capacitor. If we set these components up like this, we can pick up a square wave oscillation at the op amp's output.



Confused about how this works? If so, let's analyze this circuit step by step.[6] For that, we'll first have to understand what an op amp does. 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 amp's inputs – that's why we say their input impedance is very high**. Near infinite, actually. Okay, but why are there two of them?

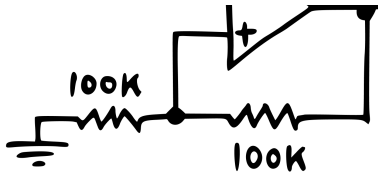


The key here is that op amps are essentially differential amplifiers. **This means that they only amplify the difference between their two inputs – not each of them individually**. If that sounds confusing, let's check out a quick example. So we'll imagine that one sensor – called the non-inverting input – 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.

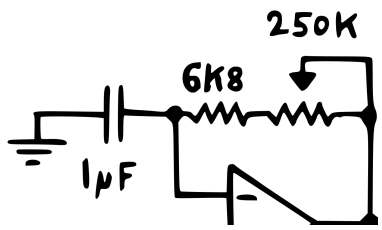
[6] You can try this chapter's circuit in a simulator. I've already set it up for you right here: https://tinyurl.com/2yk6obr5 – you can change all values by double clicking on components.

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.
So far, so simple. But in our clock generator, the relation between the two input voltages and the op amp's output voltage is quite complicated, because we feed the output voltage back to both inputs. To simplify things, we can isolate the two feedback paths for now.



Let's start with the lower one. Here, the two resistors between output, non-inverting input and ground form a 50% voltage divider.[7] It works like this. **By taking two resistances, connecting them in series, tying one end to a voltage and the other to ground, we can pick up a fraction of that voltage where they meet**. The relation between them will determine what fraction we get.

Since both resistors are of the same value here, the op amp's output voltage will be slashed in half before it reaches the non-inverting input. So we know that the voltage at that input is always exactly half of what the output voltage is. With this in mind, let's move on to the upper feedback path.



Here, things are a bit more messy, because instead of connecting output and input with a voltage divider, we use a combination of a variable resistance and a capacitor. **That resistance is made up of a fixed 6k8 resistor and a 250k potentiometer set up as a variable resistor**. We'll talk a bit more about potentiometers in a later chapter.[8] For now, all you have to know is that if we set one up like shown here, it will work as a resistor whose value you can control by turning a knob.

The added 6k8 resistor is then setting a minimum resistance – so that when the potentiometer is set to 0 Ω, there is still some resistance between the op amp's output and the capacitor. **This is important, because the whole idea is that we want to slowly fill and drain that capacitor by having a restricted amount of current flow from and to the op amp's output**.[9]
Why? Because this is what makes this circuit oscillate. Here's how. Imagine that we've just set this circuit up and connected our power supply. Then you might assume that the

[7] Read more about voltage dividers in the components & concepts appendix (page 39).

[8] You can also read more about potentiometers in the components & concepts appendix (page 40).

[9] Read more about capacitors in the components & concepts appendix (page 37).

voltages at the inverting and non-inverting inputs and the op amp's output must be 0 V. In theory, this is true – which means that nothing should happen, really, because no current is flowing on either path.

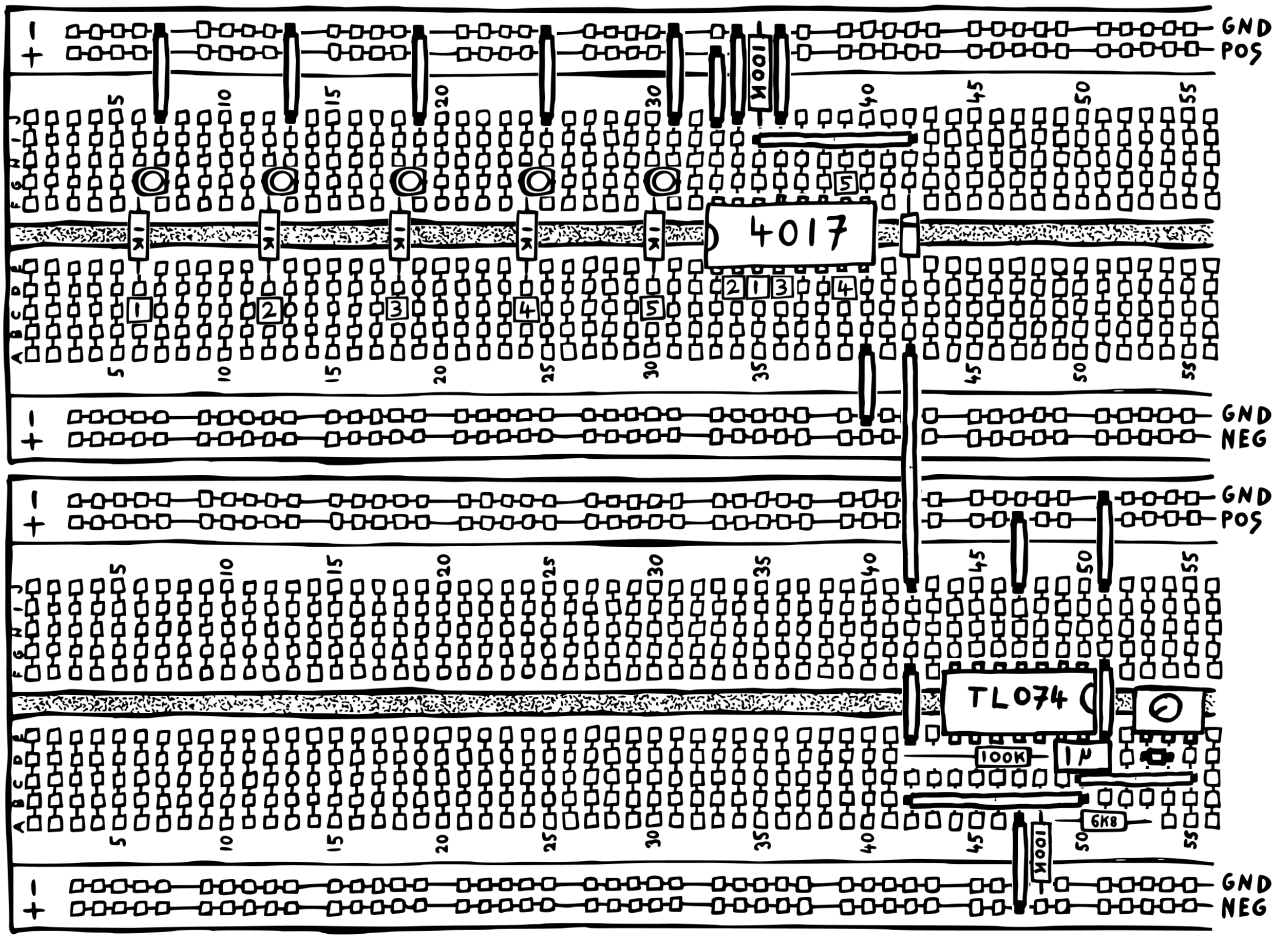
In reality, there will be subtle voltage differences though, because we're dealing with analog components in a non-ideal world. And because the op amp's gain is so strong, we can expect its output voltage to either crash down to –12, or jump up to + 12 V immediately. Let's assume it's the latter.

Then, the voltage divider on the lower feedback path will set the voltage at the non-inverting input to 6 V exactly. At the same time, we'll see a small current flow through the potentiometer and resistor on the upper feedback path and into the capacitor. As it fills up, the voltage at the inverting input rises slowly. Until it crosses the 6 V-line. **Then, because the result of the op amp's input subtraction is now negative, the output will crash down to –12 V**. Pulling the voltage at the non-inverting input down to –6 V with it.

At the inverting input, the voltage will only slowly start dropping down from 6 V, as the charge inside the capacitor is pulled into the op amp's output. As soon as it crosses the –6 V-line, everything resets: the output jumps to +12 V, pushing the non-inverting input's voltage up to 6 V, and the capacitor is slowly filling up again.

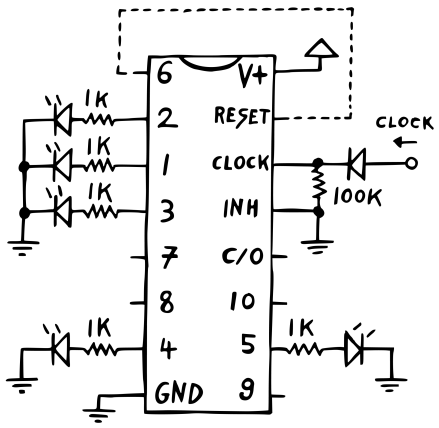
Since this process will repeat indefinitely, we get a constant square wave-oscillation at the op amp's output as a result. Better yet: we can adjust the frequency of that oscillation by turning the potentiometer's knob. **Because the lower the resistance on the top feedback path is, the quicker the capacitor is charged and discharged**. (Of course, if there's no resistance on that path, the mechanism falls apart – hence the 6k8 baseline resistor.)

So let's set this up on our breadboards and see how we fare. **Make sure that the TL074-chip (which houses four standard op amps) is set up exactly as shown in the layout – if you reverse the power connections, it will heat up and die!**



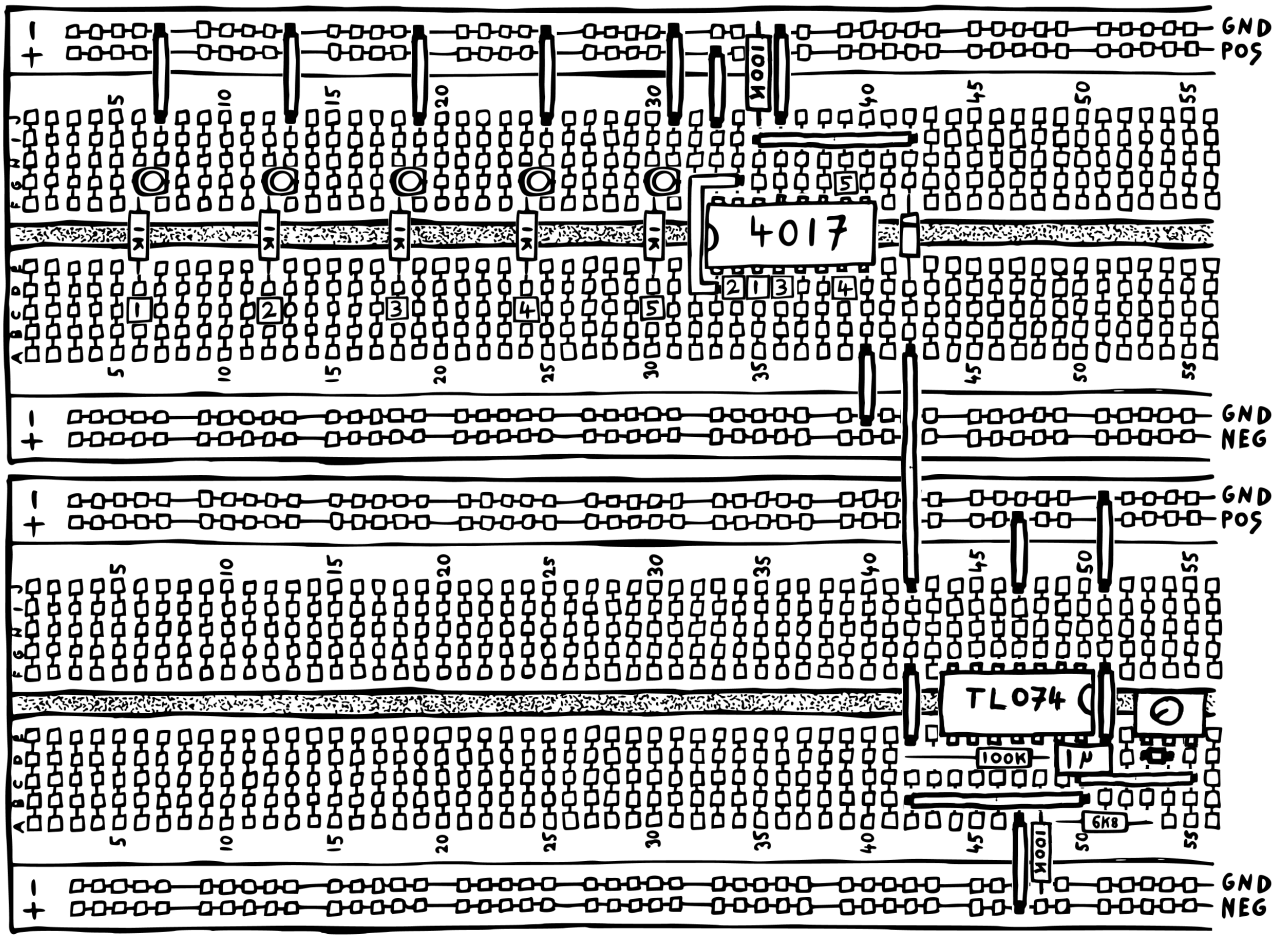
Since we've hooked our clock generator up to the 4017's clock input directly, you should now be able to see that all LEDs go dark for an extended period of time as the count state increases past 5. This is not really what we're looking for. As I said earlier, we'd expect a standard sequencer to loop back to the first step after it passes the last one. So how do we achieve this with our chip?

RESETTING THE LOOP

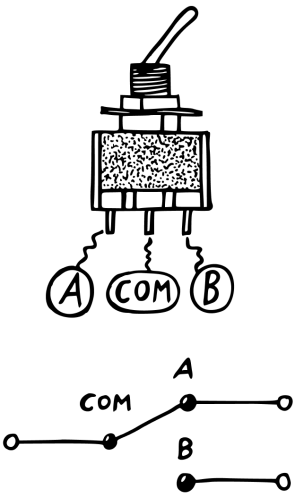


Simple: by using the chip's reset functionality. It works like this. Whenever we apply a high level voltage to the reset pin, the chip will clear its internal count state. Setting the count to one and making the first step LED light up, no matter where we were before. **If we want our counter to loop around after step five, we simply connect step six to the reset pin**.

Why step six and not step five? **Because the reset input is triggered the instant it detects a high level voltage**. So by connecting step six to the reset pin, we jump back to the first step as soon as we try to move past step five. Testing this on the breadboard is as easy as connecting step six and the reset pin with a jumper.

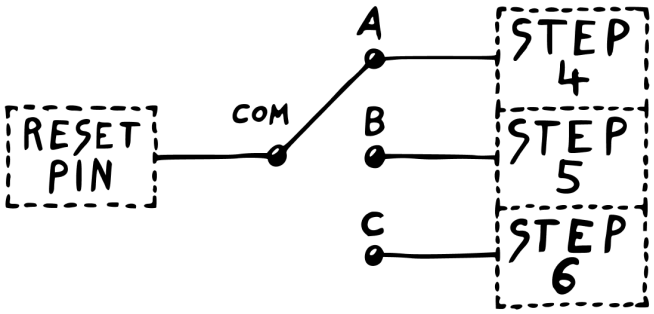


The chip should now be trapped in an endless five-step loop. Just as we wanted! Of course, there's nothing keeping us from shortening the loop to our heart's content. Go ahead and try it with four steps, three steps and two steps!



Doing this with a jumper is of course not the most user-friendly method, though. So in an actual eurorack module, you'd probably want to use a switch. With a regular single pole, double throw-switch, we can connect one point in a circuit to one of two others – depending on the switch's position. **It's the equivalent of keeping our jumper plugged into one spot, while moving the other end between two fixed points**.

This would allow us to connect our reset pin to two different steps, giving us the choice between two different loop lengths.[10] Since I like to be a bit more flexible with my options, though, we went for a single pole, triple throw-switch in the finalized design instead.

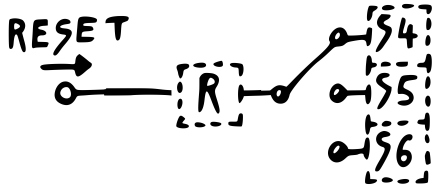


It's the same basic concept, but with three instead of just two signal destinations. Allowing us to choose between three distinct loop lengths: five steps, four steps and three steps.

[10] For the final module, we're using a similar idea to switch between the internal clock and our external clock input. But instead of a dedicated SPDT switch, we use something called a switched socket. It works like this: whenever nothing is plugged in, it will forward the internal clock signal. But once you do plug in an external signal, that signal takes precedence.

THE RESET INPUT

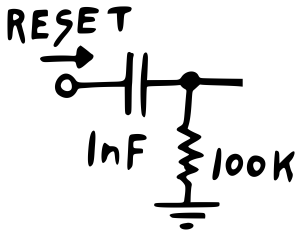
In some situations & patches, it might be useful if we're able to reset our sequence using an external trigger. For that, you might be tempted to simply set up a jack socket and connect it straight to the chip's reset pin. Unfortunately, it's not quite that easy. There are two problems with this approach.

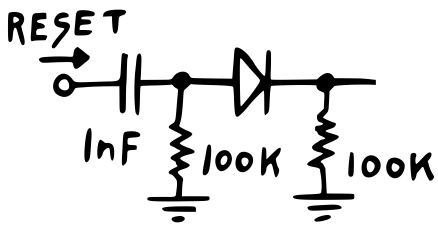


First, since the reset pin is always tied to one of our step outputs, there'd be a straight connection between that step output and the new reset input. **This is a problem because for one, the path between reset input and step output doesn't have any resistance.** It's basically a short circuit. So if our reset signal is able to provide a lot of current, we might actually see our chip go up in smoke. Also, since the step outputs are connected to our status LEDs, we'd see one of them flash every time we trigger the reset input externally.

The second problem is arguably more of a subjective thing, and it relates to way the 4017's reset pin works: **as long as there is a high level voltage present at that reset pin, the chip stops counting**. Meaning that it doesn't advance past the first step. It's essentially waiting for the reset pin voltage to fall back down to ground.

So if you'd use a standard square wave-signal to trigger a reset, the sequence would halt for a significant stretch of time. Simply because that square wave's high phase comprises half of its entire wavecycle. For me, this is not really desirable. I'd like my sequencer to keep on trucking along after each reset – even if the reset signal stays high for an extended period of time.



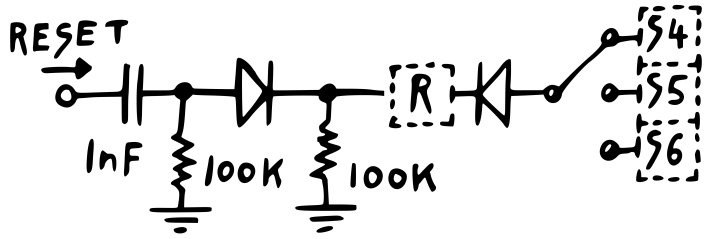


So how do we pull this off? Easy: with a basic, passive gate-to-trigger converter – which is really just a capacitor and a resistor, set up like this. **This circuit will turn a square wave-cycle into two short voltage spikes.**[11] First a positive one, when the input transitions from low to high – and then a negative one, when it drops from high to low.

Since we're not interested in the negative spike, and it could potentially harm our chip, we'll eliminate it using another diode plus pulldown resistor-combination. This way, only the positive spike can get through and trigger our chip. It's the same idea we've used for the clock input earlier.

[11] You can try some of this chapter's circuits in a simulator. I've already set them up for you right here: https://tinyurl.com/y8v7mrkr – you can change all values by double clicking on components.

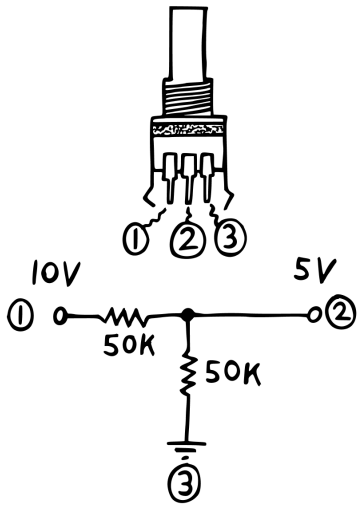
The 100k pulldown resistor ensures that there's always a defined voltage at the reset pin, no matter what happens at the reset input and step outputs. Speaking of the step outputs: to address the first problem and protect them from current coming in through the reset input, we'll simply use another diode that we'll put between them and the reset pin.



Cool! Unfortunately, our two breadboards will get pretty crowded as we move on – so I've decided to omit the reset input from the suggested layout. If you want, you can of course try to set this up anyways and test it out. If you do, experiment with different input signals – you could try LFOs, other sequencers, maybe even a sample & hold for some chaotic randomness.

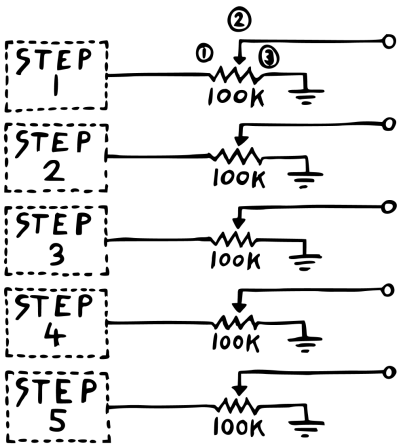
THE CV OUTPUT

As cool as watching our chip light up some LEDs is, it's still not sending out any type of sequence. So let's change that. We'll start with implementing a control voltage output. Right now, our individual steps are either sending out 0 V when they're off, or 12 V when they're on. And while we can't do much with 0 V, we can do a lot with 12 V.



Because if we take those 12 V and route them through a potentiometer set up as a variable voltage divider, we can scale them down to any voltage between 12 and 0 V. This is possible since conveniently, a potentiometer is really just a voltage divider shoved into a single component which allows us to change the resistance relation by turning a knob.[12]

So let's say we add in such a variable voltage divider after every one of our five step outputs. Then by turning the potentiometers' knobs, we can set a specific output voltage between 0 and 12 V for each step. Cool! Only problem is that now, we've got 5 individual outputs: one for each step. **That's not really what we were looking for – we wanted a single output that gives us each step's voltage in succession**.



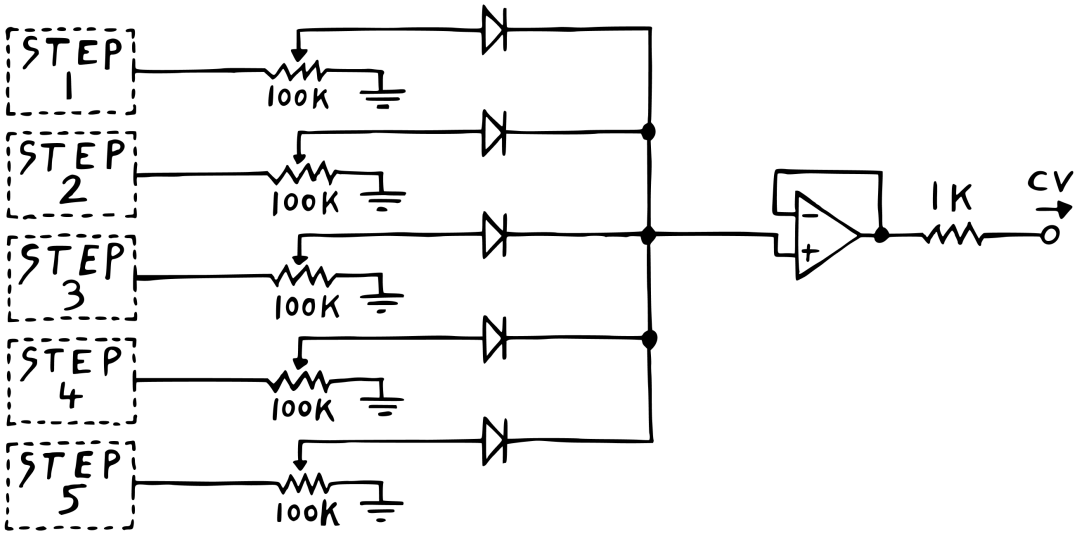
Now you might be tempted to simply connect all these individual outputs together. After all, the chip's steps only send out a voltage when they're active, right? And since no two steps are active simultaneously, we should only see the currently active step's voltage at our spliced-together output.

Unfortunately, this idea ignores the fact that these five outputs can't just source current – they can also sink it. Which means that when one of the outputs is active, current from that output would flow back into the other four. This is not only shoddy engineering – it would also mean that the output voltage is much lower than expected. So how do we fix that? Easy: with a bunch of diodes.

As we've discussed earlier, diodes only let current pass through in one direction. So if we place one after every individual step output and then connect all the diodes together, the problem should be fixed.

---

[12] You can try this chapter's circuits in a simulator. I've already set them up for you right here: https://tinyurl.com/yakndukq – you can change all values by double clicking on components.

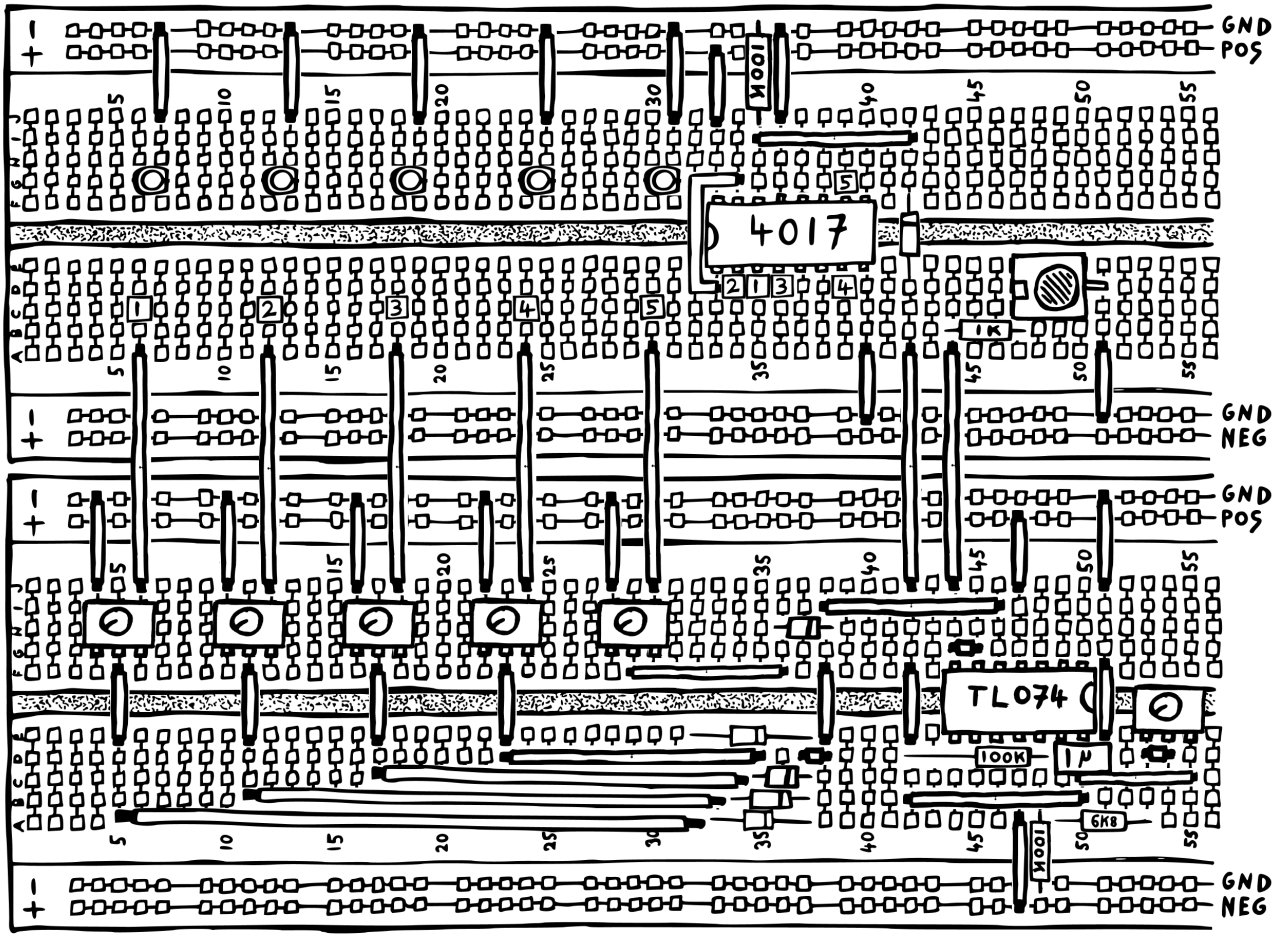


Now, no current can flow back into the outputs, since the diodes are blocking that. Great! There's just one small additional thing we need to do before we can give this a try. And that's buffering the output voltage. Because without doing that, the current coming from our chip has to squeeze through those potentiometers before it reaches the output.

This means that if the module we're driving with our output is even a bit power-hungry, it will make the output voltage drop noticeably. This is not ideal – we want the output voltage to be the same in any and all situations. To ensure that, we use an op amp-based buffer. Buffers are great in situations like these, because they allow us to make copies of voltages without pulling any current.[13]

So by placing a buffer between our diodes and the output socket, we can make sure the module we drive with it can pull plenty of current without the output voltage dropping. As a final touch, we'll put a 1k resistor after the buffer's output. This is just to limit the maximum amount of current flowing out of (or into) the op amp in case of a short circuit.

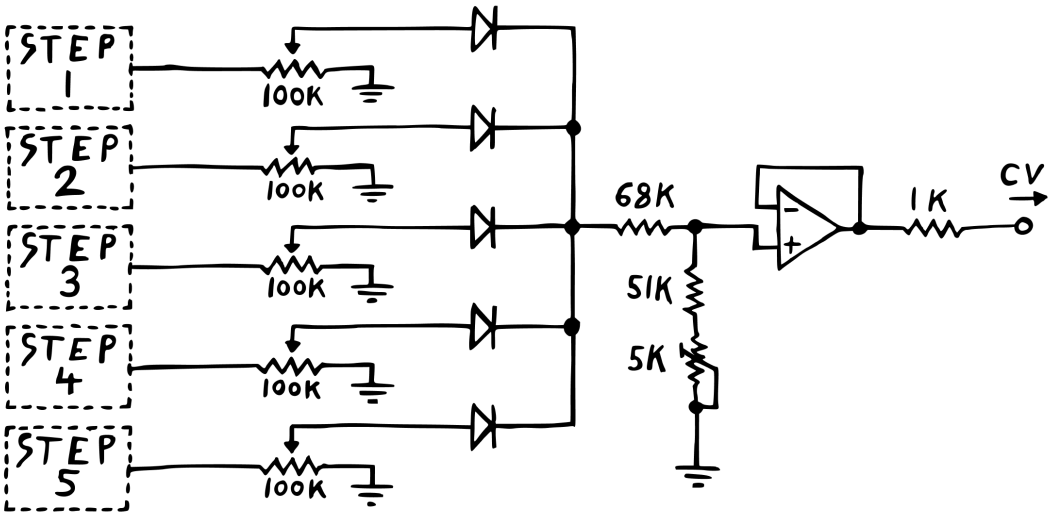
[13] Read more about op amps and buffers in the components & concepts appendix (page 42/43).



Let’s see if this works! If you play with the potentiometers, you should be able to get a melody going. Cool! Although it’s unfortunately pretty difficult to dial in notes with any kind of precision. This is because the CV range per step is just way too big – it’s going from 0 all the way up to 12 V!

CV SCALING

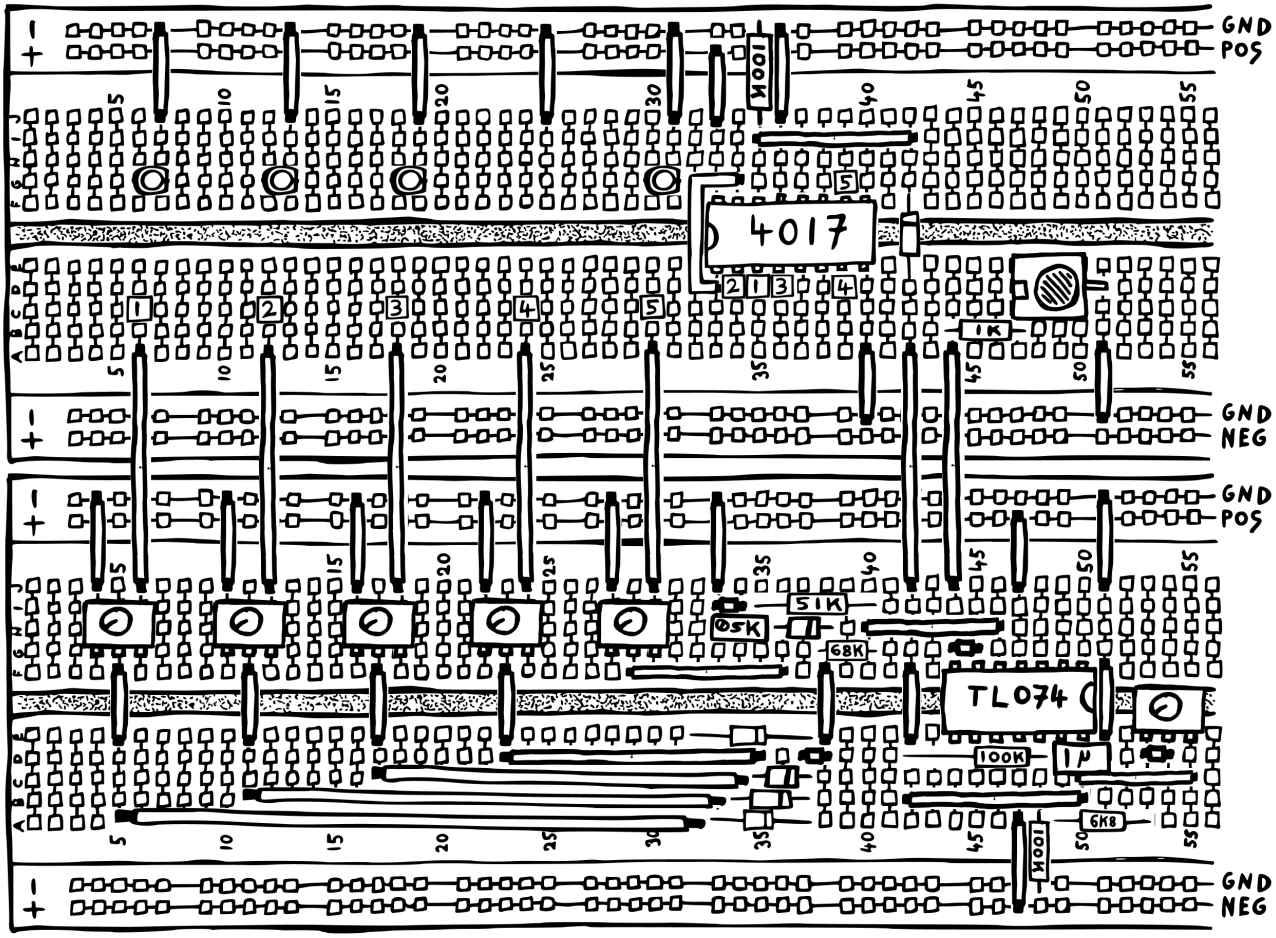
Thankfully, adjusting this is pretty easy. All we have to do is scale our CV down using a standard, fixed voltage divider. If we want our CV to stay within a range of 0 to 5 V, for example, we'll have to choose two resistors that'll give us a divide-down factor of about 2.4. (Simply because 12 V divided by 2.4 gives us 5 V.) Unfortunately, that ratio is hard to hit exactly. Partly because of component tolerances, partly because finding the right value resistors can be tricky. **So we'll have to give ourselves some wiggle-room by including a small-value trimmer potentiometer in our voltage divider**.[14]



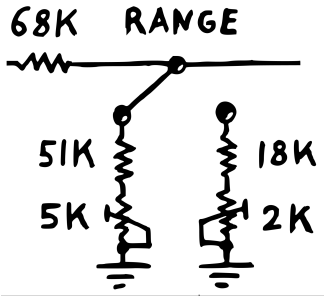
And even though we now technically have three resistors in our voltage divider, we can simply act as if the trimmer and the 51k resistor form one single, bigger resistor. So by turning the trimmer's knob, we can fine tune the voltage divider's ratio.[15]

[14] You can try this chapter's circuit in a simulator. I've already set it up for you right here: https://tinyurl.com/ycvhha6h – you can change all values by double clicking on components.

[15] Please be aware that the trimmer included in your kit might have a different footprint than the one shown in the suggested breadboard layout below. If that's the case, you'll have to slightly adapt the layout.



Now, to make sure that our sequencer's maximum CV level is close to 5 V, you should connect a multimeter to the CV output, while setting all potentiometers to maximum blast. Then, you'll have to fiddle with the trimmer until you hit those 5 V. Once you get there, connect your oscillator again and see if dialing in a melody has gotten any easier.

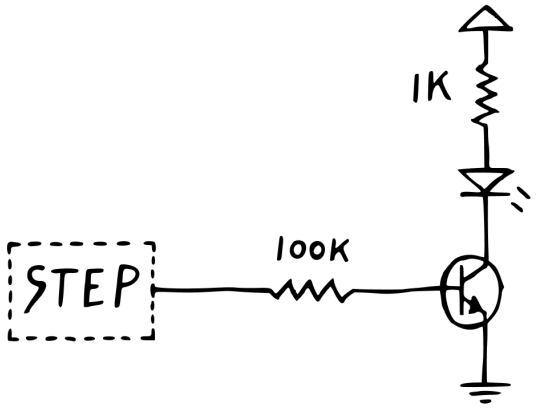


Of course you could reduce the range even more if you want. All you'd have to do is adjust the voltage divider's ratio. **For the production design, we've decided to implement two different ranges – 2.5 V and 5 V – that are selectable via an SPDT switch**. Both paths include a dedicated trimmer that you'll need to adjust when you're calibrating the finished module.

STATUS LEDS

Now, before we move on to the gate output, I'd like to take a short detour and add our LEDs back in – simply because that would make it much easier to know which step is currently active. You might be tempted to simply drive them the same way as we did before.

But that would actually be a bad long-term decision. **Because while the 4017 is able to provide enough current for a single LED, this already pushes it to its absolute limits**. Which means that if we have it drive LEDs for an extended period of time, we're risking damage or even component failure. So we'll have to find a way to offload it. For that, we can use an NPN transistor.[16] If we set it up like this, a very small current coming from the chip will be enough to drive the LED at full throttle.[17]



Here's how it works. As long as the chip's output is sitting at 0 V, no current will flow through the 100k resistor and into the transistor's base – which means that the transistor blocks any current from flowing between its collector and emitter. So the LED will be turned off as well.

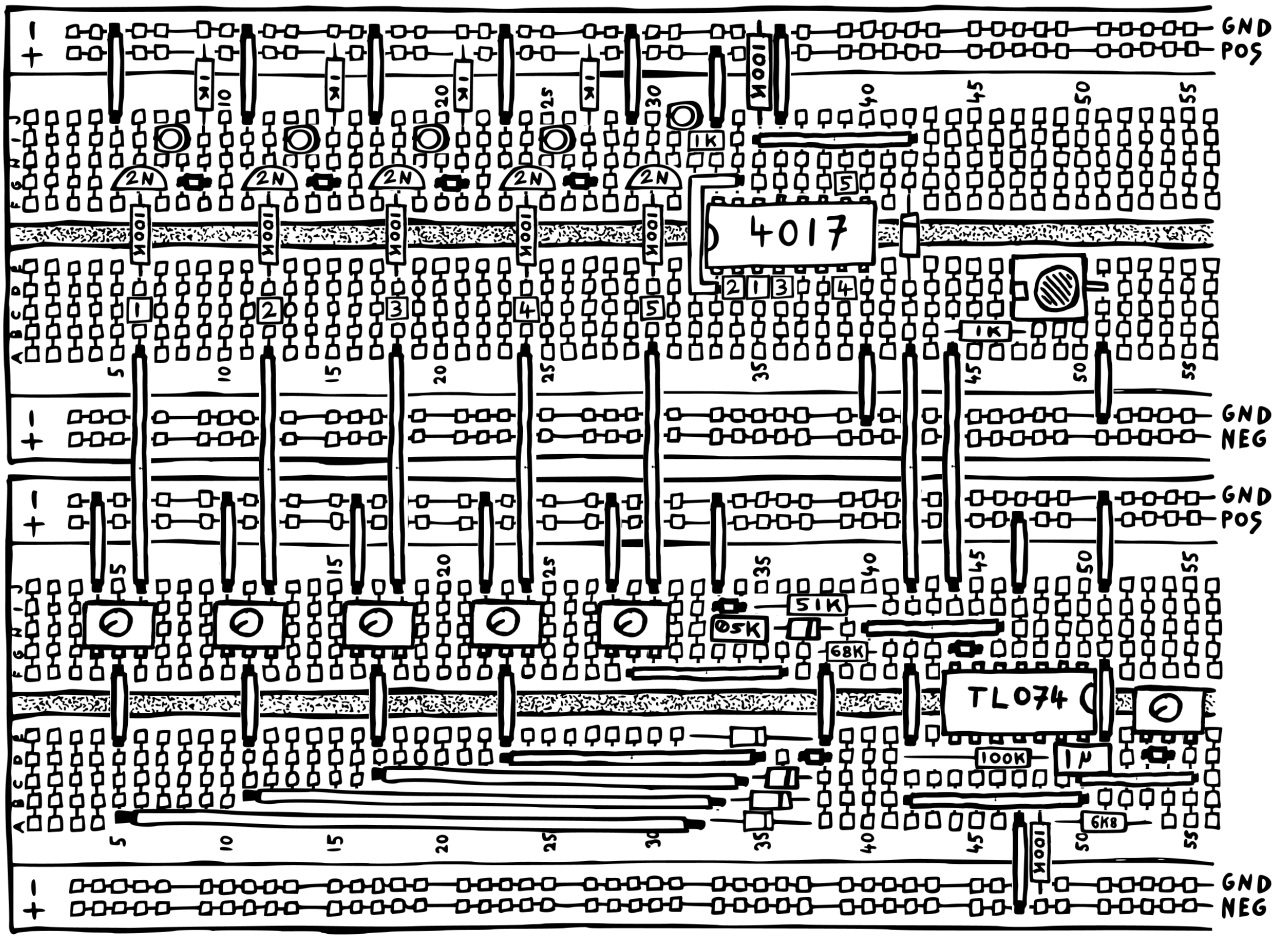
But as soon as the output goes high, a very small current is squeezed through the 100k resistor and into the transistor's base. **This is enough to make the transistor open up wide, allowing for lots of current to flow between collector and emitter**.

---

[16] Read more about NPN transistors in the components & concepts appendix (page 45).

[17] You can try this chapter's circuits in a simulator. I've already set them up for you right here: https://tinyurl.com/ybvstvjp – you can change all values by double clicking on components.

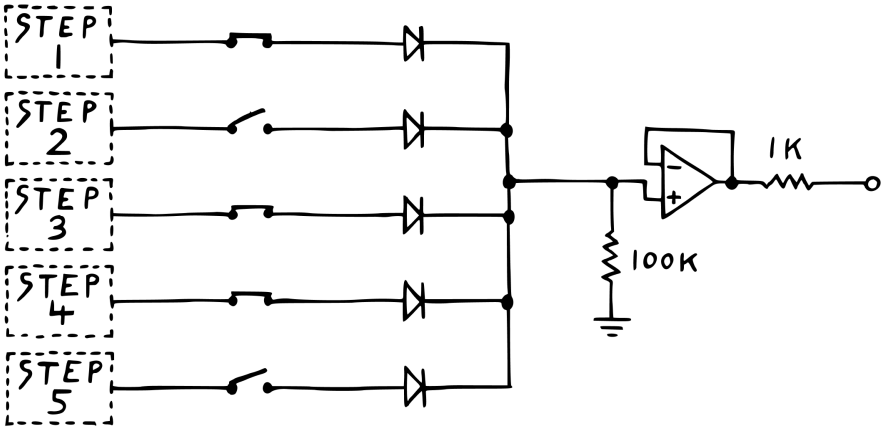
That current might actually be strong enough to wreck our LED – so to be safe, we'll put a 1k resistor between the LED and our power rail. Now, whenever the output is high, our LED will safely light up without stressing the chip at all. Great! So let's set this up on the breadboard. **Make sure you use 2N3904-transistors here!** Their pinout differs from the BC547 also included in your kit.



If you now plug your batteries back in, you should immediately see the five LEDs light up the same way they did before. They do? Then it's time to move on to the gate output.

THE GATE OUTPUT

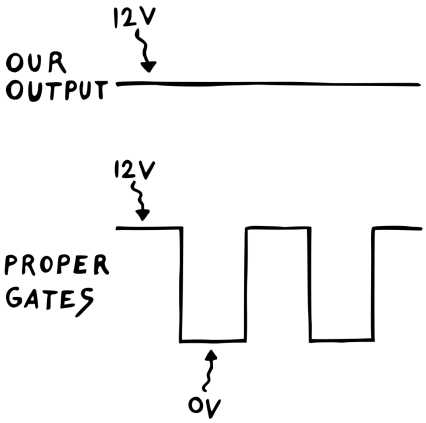
Since we said earlier that a gate is really just a voltage going from low to high and back low again, you might assume that our chip is already providing gates for us. After all, it simply turns its outputs on and off in succession – so we're essentially getting a gate per step, right? If that were true, we could apply what we've learned from the CV output and set up something like this.



Each step gets routed through its own switch, followed by a diode. Then, we'd simply connect them all together and buffer the resulting voltage with another op amp. Now, by turning these switches on or off, we should be able to decide if we want to get a gate on a specific step or not. To make sure our buffer's input isn't left floating when a step's gate is turned off, we also need to add in a 100k pulldown resistor here.

Unfortunately, if you'd try this by connecting the output to an envelope generator, you'd encounter something odd. **With all step switches turned on, that envelope won't get triggered at all**. Why is that?

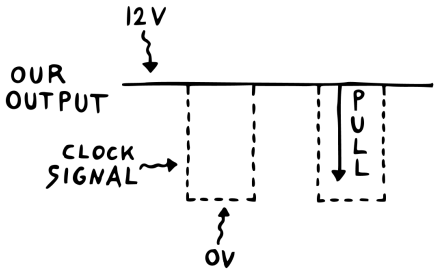
With the sequencer running, take a closer look at our step indicator LEDs. You can see that one of them is lit up at all times. There's never a moment where all LEDs go dark. Since we are combining all of the steps together for our gate output, that gate output will be stuck at a high level voltage permanently.



Simply because if one of the step outputs is high, the gate output will be high as well. This is why the envelope wouldn't be triggered properly – the gate output has to momentarily go low for that. Bummer! So how can we fix this?

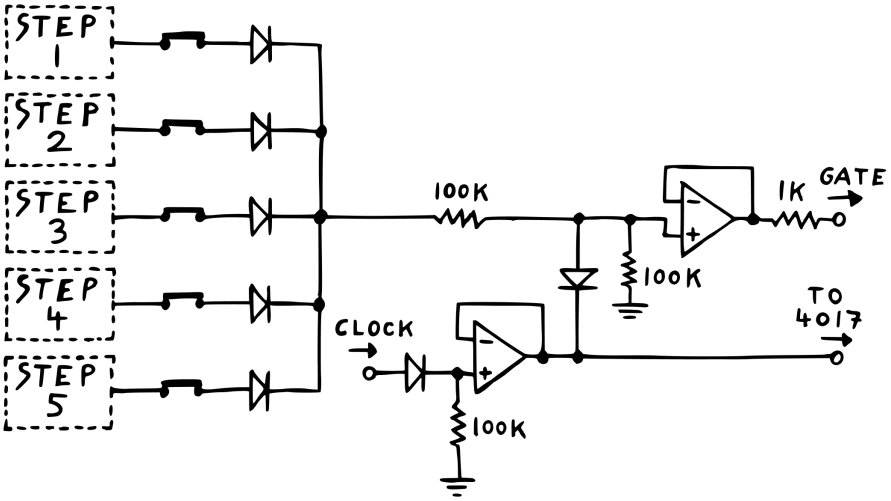
Take a look at these two graphs. The one up top is what we're currently getting out of our gate output. The one below shows us what a proper gate sequence with all gates active should look like.

Because a gate, as we said in the beginning, is a voltage going from low to high and back low again. So how do we convert our output signal into a proper sequence of gates? This is where it gets a bit tricky. We basically have to make sure that our output always goes low before we reach the next step. Thankfully, we've got something that can help us out with this.



That something is the clock signal. Because if we overlay our current output and that clock signal, we see something interesting. Where the output stays high during each entire step, the clock actually goes low for the second half.

So if we were somehow able to pull the output low whenever the clock is low, our problem would be solved! And though this might sound complicated, it's actually quite easy to implement.[18]



All we need are three components: an op amp, a 100k resistor and a diode. Here's how it works. Whenever the clock input is low, the clock buffer's output will also be low. This will allow for current to flow through the diode and into the clock buffer. Now, without the 100k resistor in the center, this would basically result in a short circuit and burning chips.

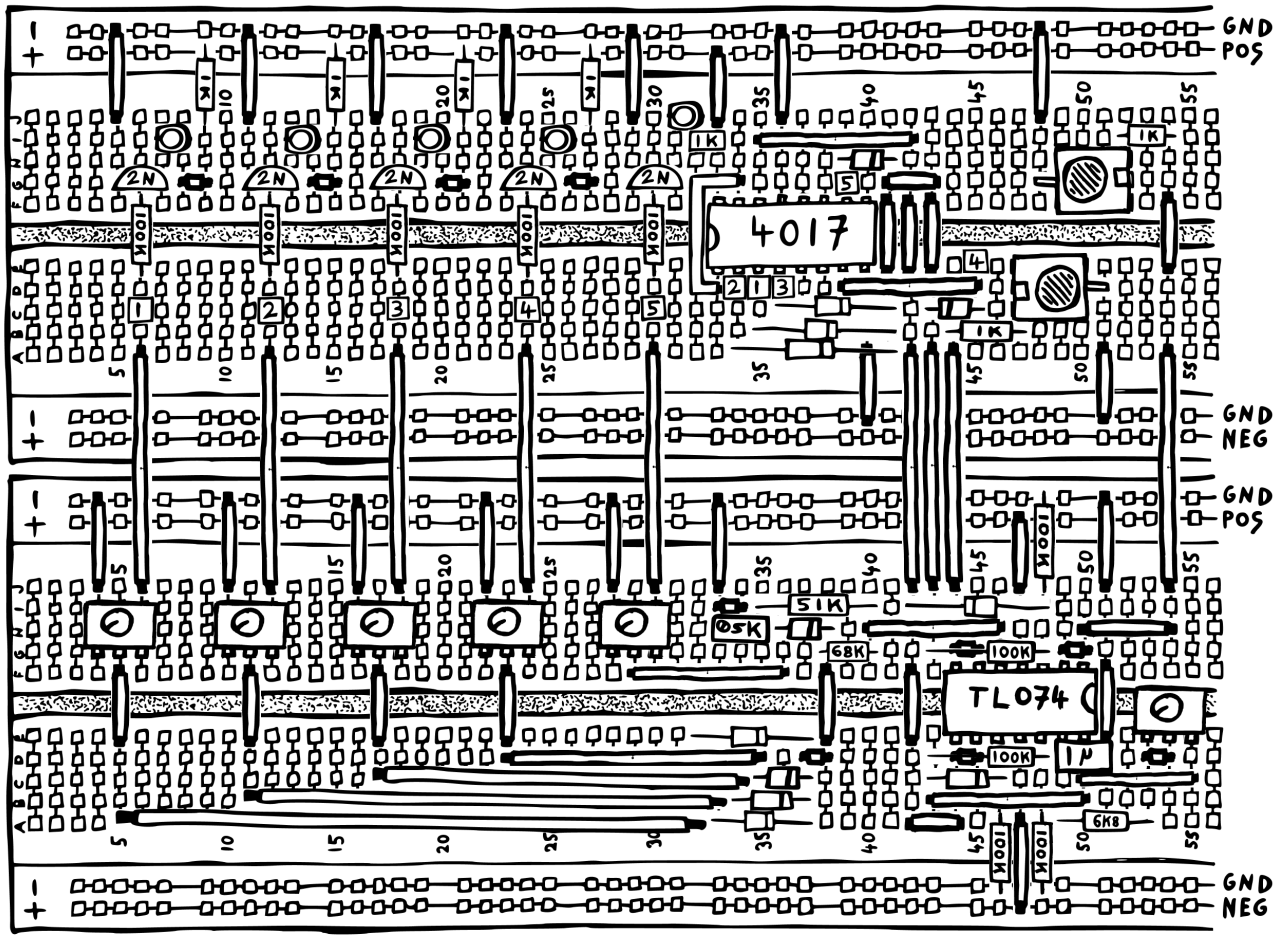
But with it, we'll simply see the voltage at the gate buffer's input drop close to ground level, as the clock buffer is comfortably eating up what's squeezed through the 100k resistor. This is why we need the clock buffer, by the way: to ensure that we can sink all the current coming down through the diode.[19]

---

[18] You can try some of this chapter's circuits in a simulator. I've already set them up for you right here: https://tinyurl.com/yatgsglm – you can change all values by double clicking on components.

[19] Also, the buffer allows us to set up a clock output/clock through by simply connecting a jack socket to the buffer's output through a 1k resistor. If you check the initial, complete circuit schematic, you'll see that that's exactly what we've done for the final module.

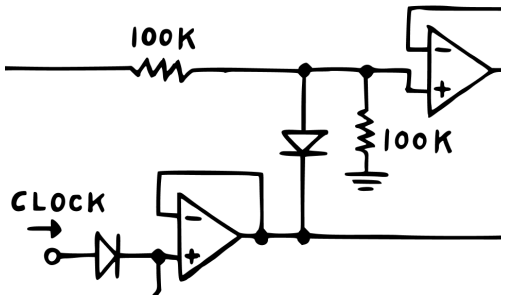
On the other hand, whenever the clock signal is high, the clock buffer's output will also be high. This means that the pull-down diode will now be pushed shut from below. Because of this, current is blocked from flowing into the clock buffer. Leaving whatever voltage we currently have at the gate buffer's input undisturbed. **So no matter if that voltage is high or low, it will stay that way whenever the clock is high**. Cool! Let's set this up on the breadboard and see how we fare.



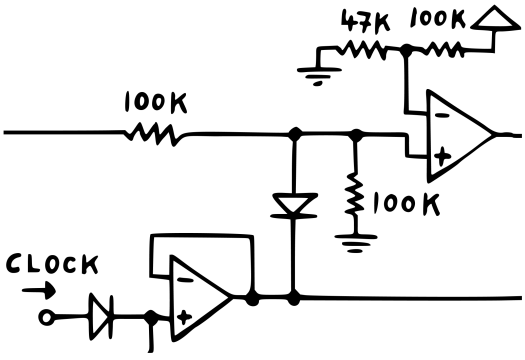
Now, depending on the envelope you're trying to drive with our new gate output, it might or might not get triggered. If it does – great! But in case it doesn't, we'll have to retrace our steps for a bit.

THE FAILSAFE GATE OUTPUT

There is one small thing that you might be concerned about already if you're really observant. When we added in this extra 100k resistor after our five diodes, we only wanted to prevent creating a short circuit.



Unfortunately, this has an unintended side-effect. **Together with our pulldown resistor, the new current limiting resistor forms a voltage divider**. This means that even if the clock input is high and the pulldown diode is blocked, the voltage at the gate buffer's input will be cut down. By 50%, to be precise.
So instead of the full 12 V coming from our counter chip, we'll only get around 6 V at the gate buffer's output. Now apparently, this is still a strong enough signal to trigger some envelopes. But different envelopes have different gate signal requirements. Some might only trigger at 8 or even 10 V. **So in order to make our sequencer compatible with any envelope, we should try and boost its gate output up to 12 V**. To do that, we'll first turn our gate output buffer into a comparator.[20]



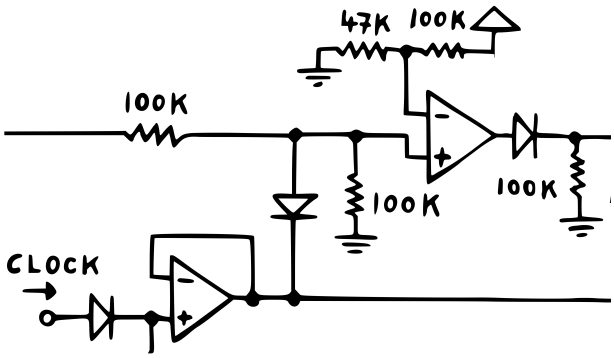
Comparators, if you don't know, simply compare an input voltage to a reference voltage. When the input voltage is above the threshold, the comparator's output will jump to the positive supply voltage. Conversely, when the input is below the threshold, the output will jump to the negative supply voltage.

---

[20] You can try this chapter's circuits in a simulator. I've already set them up for you right here: https://tinyurl.com/y6vasvt7 – you can change all values by double clicking on components.

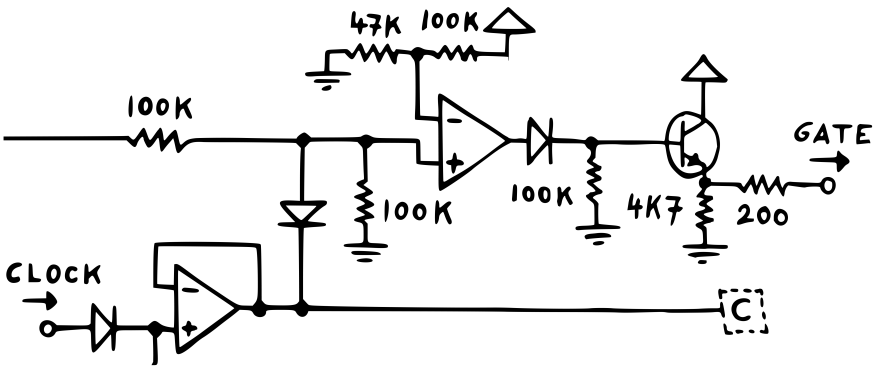
In our setup here, I've set the comparator's threshold to about 3.8 V with a 100k/47k voltage divider. The exact value is actually not that important here – as long as its significantly above 0 V and below 6 V. Because as we know, the voltage at the comparator's input will be swinging between slightly above 0 and around 6 V. So the threshold needs to be somewhere in the middle between those two values.

Then, the comparator will push out super beefy gates jumping between + and – 12 V. Unfortunately, that's overshooting the mark considerably. We said we want 12 V gates – not 24 V gates! To fix this, we use a trick that should be very familiar by now: routing our signal through a diode, followed by a pulldown resistor.



This will cut off our gates' lower halves, making them swing between just +12 and 0 V. There's just one problem with this. **Whenever the comparator's output is low and the diode blocks, the circuit's output impedance is really high – simply because there's a big 100k resistor between the output and ground**. Meaning that it can't sink a lot of current. This can be a problem if the module we're driving with our gate output relies on it having a low output impedance both in the high- and low phases.

Okay, so why don't we just buffer our output with another op amp? Unfortunately, we've already used up all four op amps in our TL074. And while we could set up an additional single op amp-chip, the problem is that we couldn't really fit it onto the module's PCB. So we'll have to find a more compact solution. Thankfully, we can build a decent, if somewhat power-hungry buffer with a single NPN transistor and two resistors.

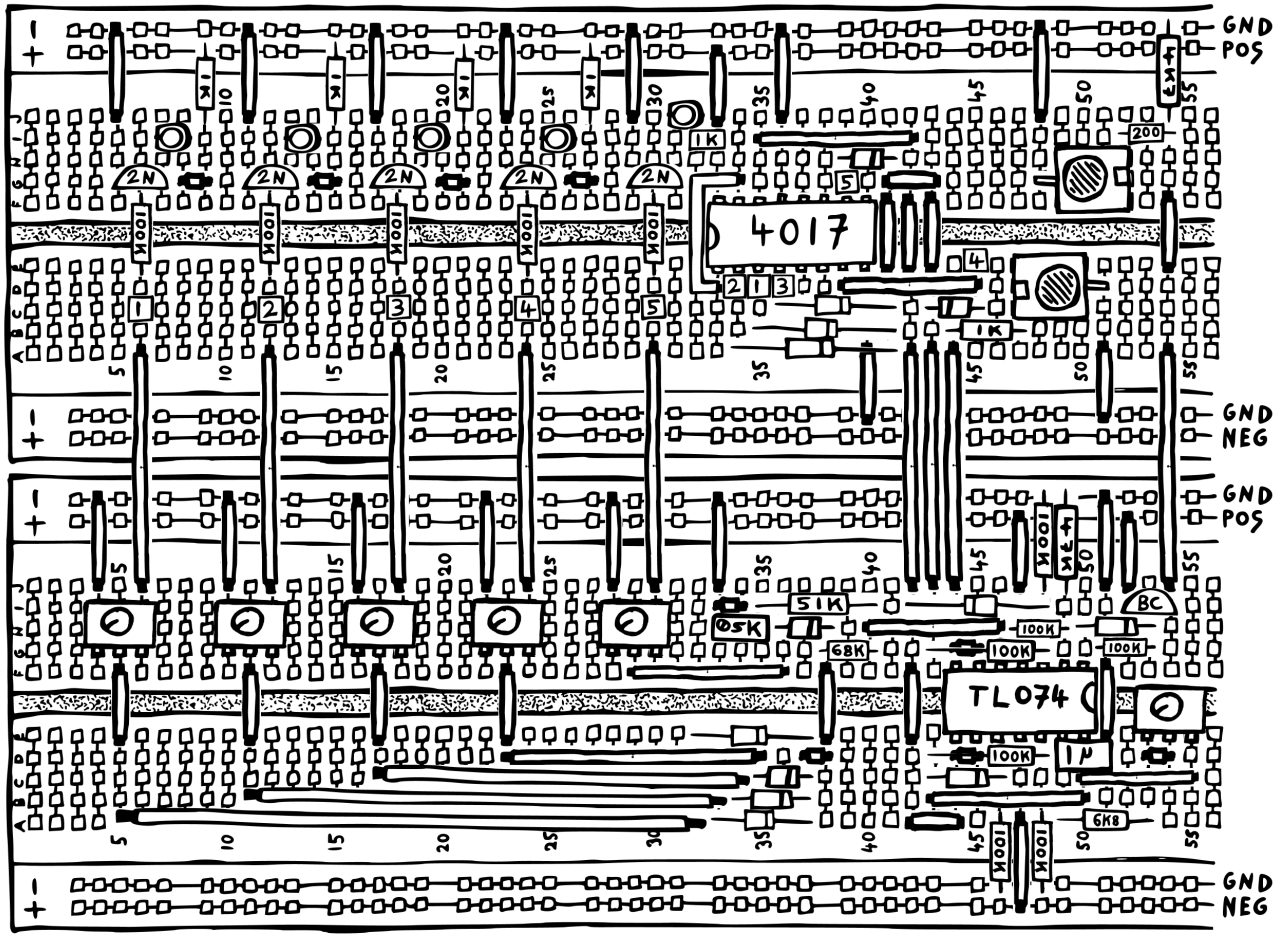


Here's how it works. **We set the transistor up in what's known as the emitter follower configuration**. For that, we simply connect its collector to the power rail, its base to the signal we want to buffer, and its emitter to ground through a rather small-value resistor. Then, we can pick up a buffered version of our gate signal directly at the transistor's emitter (which we'll hook up to an output socket through a current limiting 200 Ω resistor). To understand why, let's analyze the two different states this circuit can be in.

The first one would be if the comparator's output is currently low. Then, the transistor will be closed, because there is no current flowing into its base. **As a result, our gate output is simply connected to ground through the 200 Ω + 4k7 resistor combination**. This is still a considerably higher output impedance than the 1k that we've got on all other outputs. But compared to the 100k from the diode plus pulldown resistor setup, it's a marked improvement. (This is why we use a 200 Ω output resistor here, by the way – to not push up the output impedance any further.)

The circuit enters its second state once our comparator's output goes high. Then, a small current will start flowing through the diode and into the transistor's base, opening the connection between collector and emitter. **This results in a much larger current flowing directly from the power rail through the 4k7 resistor and to ground**. And because that 4k7 resistor restricts the maximum amount of current that can flow, a voltage will build up above it – so right at the transistor's emitter.

Now conveniently, that voltage will be just slightly lower than the voltage our comparator is applying to the transistor's base. Because if it was much lower, then more current would flow into the base, opening the transistor up even wider, which in turn would raise the voltage at the emitter. Until they are pretty close to each other. **That's why it's called an emitter follower – because the emitter voltage follows the base voltage**! And so our gates should be properly buffered as a result.



If your envelope wasn't triggered using the previous setup, it should work just fine now. But there's one thing you maybe haven't tested yet: deactivating some of the steps' gates. If so, go ahead and try it by removing some of the gate's diodes. **If that works: congratulations, you've built a fully functional five step sequencer!**

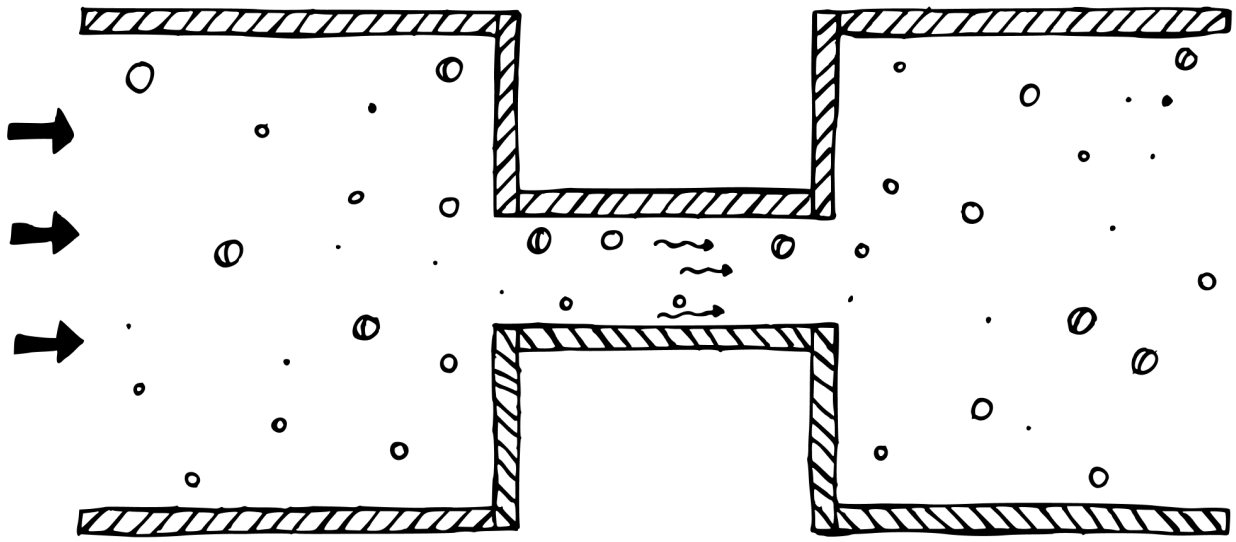
If you now want to make your creation permanent, dig out the panel and PCB from the kit, heat up your soldering iron and get to building! You can find more information on how to populate the board & how to solder in the enclosed appendix.

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

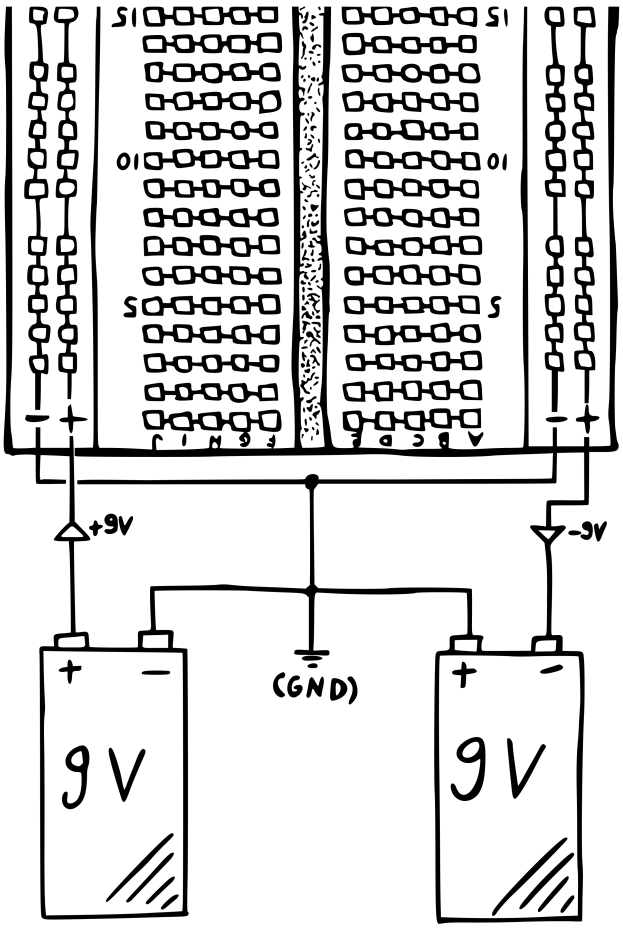
There are three main properties we're interested in when talking about electronic circuits: resistance, voltage and current. To make these less abstract, we can use a common beginner's metaphor and compare the flow of electrons to the flow of water through a pipe.



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.

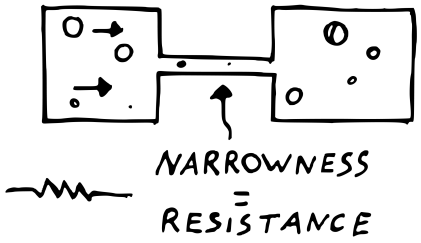


By connecting the batteries like this, the positive terminal of the left battery becomes your +9 V, while the negative terminal of the right is now your –9 V, and the other two combine to become your new ground.[21] **Please make sure you disconnect the batteries from your breadboard when you make changes to the circuit!** Otherwise you run the risk of damaging components.

[21] If you're struggling with setting this up, you can watch me do it here: https://youtu.be/XpMZoR3fgd0?t=742

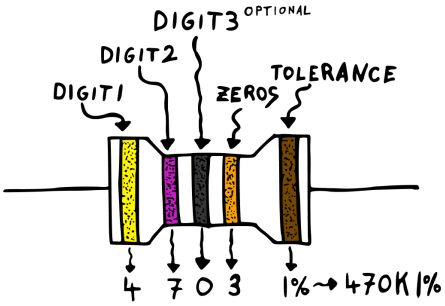
RESISTORS

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[22] – 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.



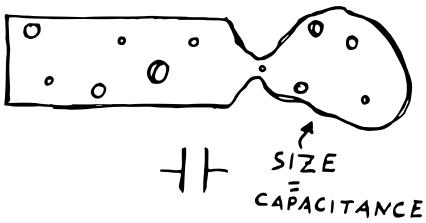
For the resistors in this kit, a yellow body tells you that the actual resistance value might be ±5 % off. A dark blue body indicates ±1 % tolerance. Some kits will also contain light blue ± 0.1% resistors to avoid the need for manual resistor matching.

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.

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

CAPACITORS

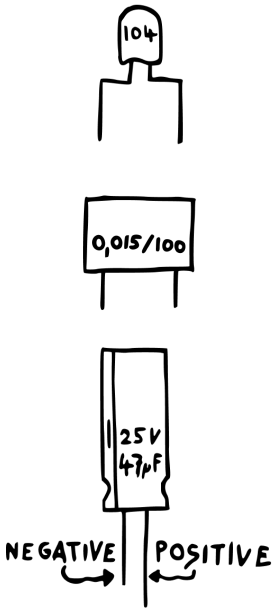
A capacitor is a bit like a balloon that you can attach to the open end of a pipe. If there's some pressure in the pipe, the balloon will fill up with water until the pressure equalizes. (Since the balloon needs some space to expand into, both of the capacitor's legs need to be connected to points in 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.[23] 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.[24]

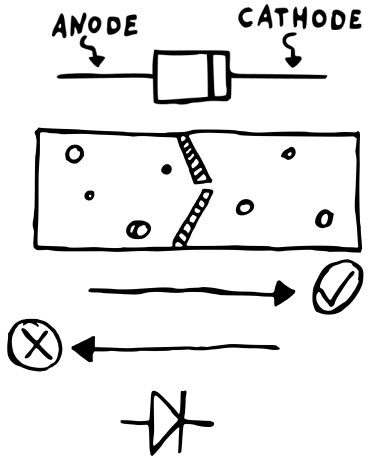
Electrolytic capacitors can be identified by their cylinder shape and silver top, and they usually directly indicate their capacitance value and their voltage rating. They are polarized – so make sure you put them into your circuit in the correct orientation.

---

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

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

DIODES

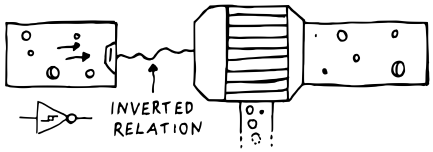


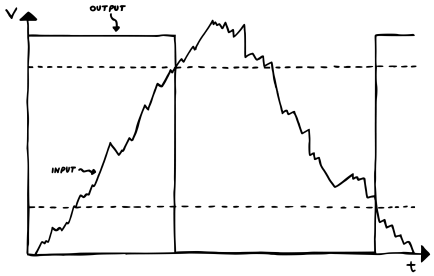
Diodes are basically like one-way valves. **Current can only pass through in one direction – from anode to cathode**. That direction is indicated by the arrow in the diode symbol and by a black stripe on the diode's casing. So any current trying to move in the opposite direction is blocked from flowing.

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

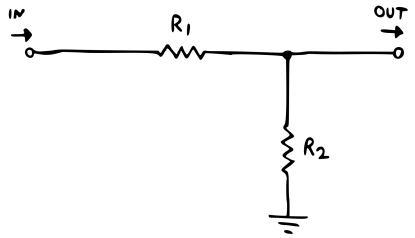




You can think of a Schmitt trigger inverter as two separate things. **On the left, there's a sensor that measures the pressure inside an attached pipe**. **On the right, there is a water pump**. This pump's operation is controlled by the sensor. Whenever the pressure probed by this sensor is below a certain threshold, the pump will be working. If the pressure is above a second threshold, the pump won't be working.

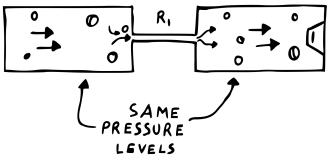
Here's a quick graph to visualize that. The squiggly line represents the voltage at the input, while the dotted line shows the voltage at the output. So every time we cross the upper threshold on our way up, and the lower one on our way down, the output changes its state. One thing that's very important to keep in mind: no current flows into the sensor! It's really just sensing the voltage without affecting it.

VOLTAGE DIVIDERS

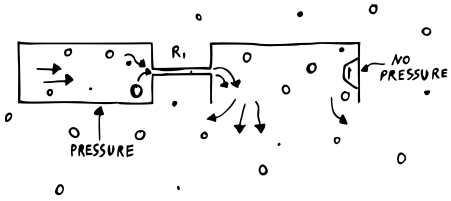


A voltage divider is really just two resistors set up like this: input on the left, output on the right. If R1 and R2 are of the same value, the output voltage will be half of what the input voltage is. How does it work?

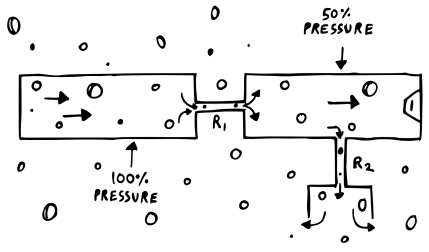
Let's use our analogy again: so we have a pipe on the left, where water is being pushed to the right with a specific amount of force. Attached to it is a narrow pipe, representing R1, followed by another wide pipe. Then at the bottom, there's another narrow pipe, representing R2, where water can exit 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.



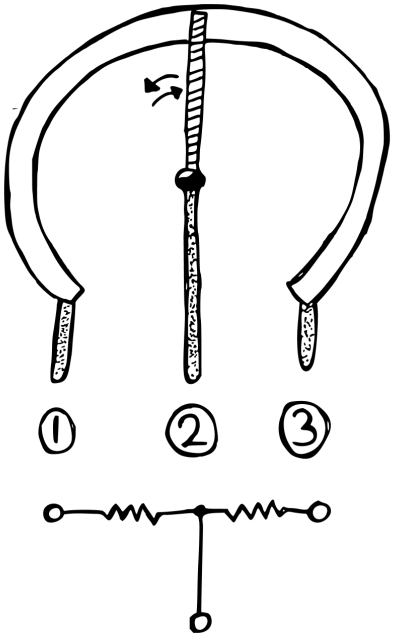
On the other hand, imagine R2 would just be a wide opening. Then **the pressure on the right would be 0**, because it'd all escape through that opening. But what happens if R2 is neither completely closed off nor wide open? Then the pressure would be retained to varying degrees, depending on the narrowness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

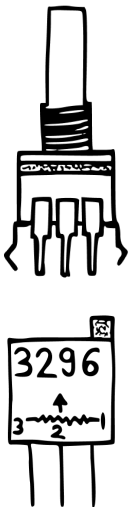
Potentiometers can be used as variable resistors that you control by turning a knob. **But, and that's the handy part, they can also be set up as variable voltage dividers**. To see how that works, let's imagine we open one up.



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.



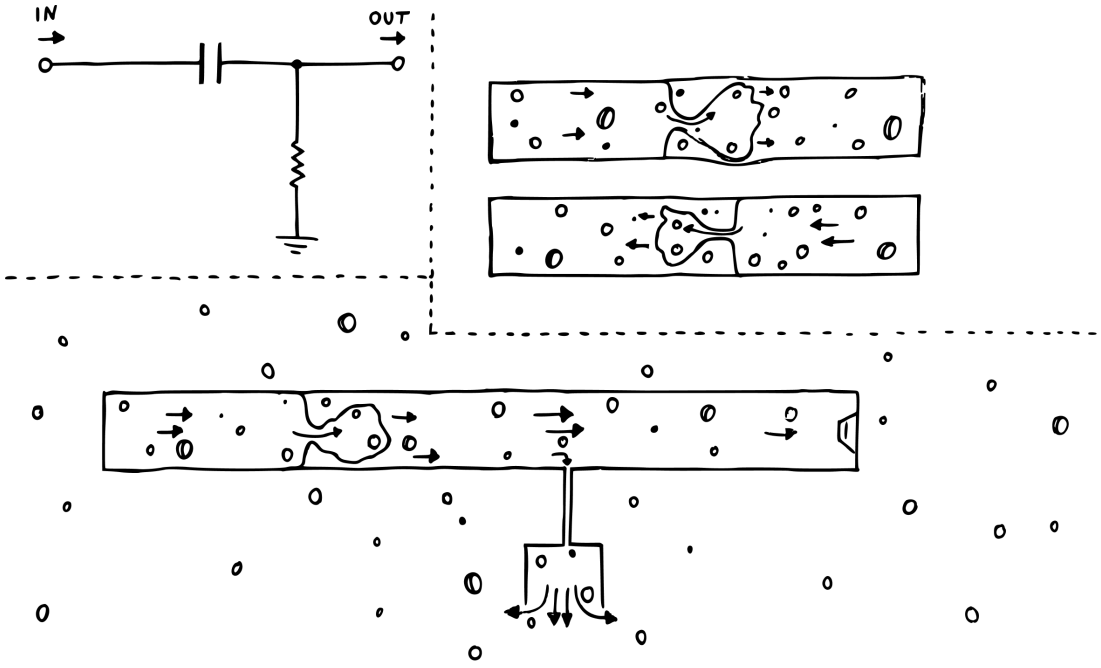
In these kits, you will encounter different types of potentiometers. First, there's the regular, full-size variant with a long shaft on top. These are used to implement user-facing controls on the module's panel and they usually – but not always – indicate their value directly on their casing. Sometimes, they'll use a similar encoding strategy as capacitors, though.[25]

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.

[25] 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**.

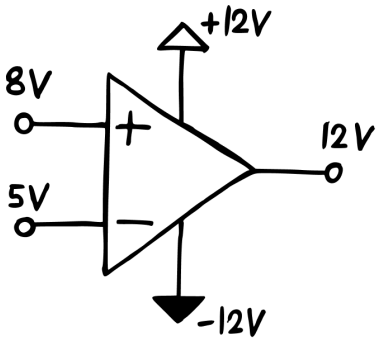


Next, we'll bring in a resistor after the coupling point, going straight to ground. **This acts like a kind of equalizing valve**. Now imagine we apply a steady 5 V from one side. Then on the other side, we'll read 0 V after a short amount of time. Why? Because we're pushing water into the balloon with 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?



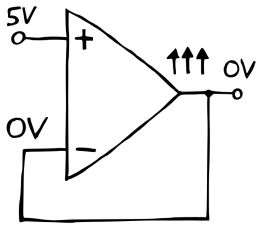
The key here is that op amps are essentially differential amplifiers. This means that they only amplify the difference between their two inputs – not each of them individually. If that sounds confusing, let's check out a quick example. So we'll imagine that one sensor – called the non-inverting input – 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.

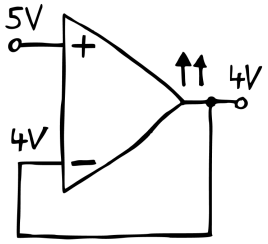
The handy thing though about op amp outputs is that they draw their power directly from the power source. This means that they can supply lots of current while keeping the voltage stable. **That's why we say an op amp has a very low output impedance**.

OP AMP BUFFERS/AMPLIFIERS

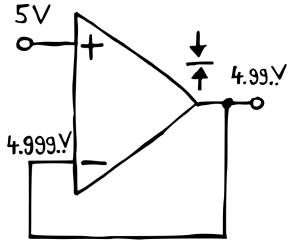
Buffering, in the world of electronics, means that we provide a perfect copy of a voltage without interfering with that voltage in the process. With an op amp-based buffer, the buffering process itself works like this. We use the non-inverting input to probe a voltage, while the inverting input connects 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.



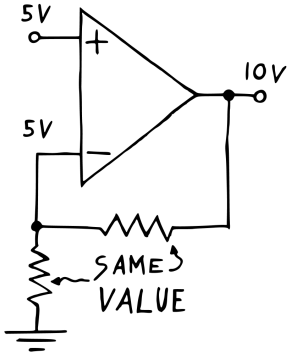
Before the op amp starts processing the voltages at its inputs, the output will be switched off. This means that **output and inverting input sit at 0 V at first**. So then, the op amp will subtract 0 from 5 and multiply the result by its gain. Finally, it will try and increase its output voltage to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



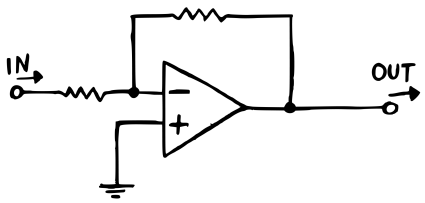
But as it's pushing up that output voltage, the **voltage at the inverting input will be raised simultaneously**. So the difference between the two inputs is shrinking down. Initially, this doesn't matter much because the gain is so large. As the voltage at the inverting input gets closer to 5 V though,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

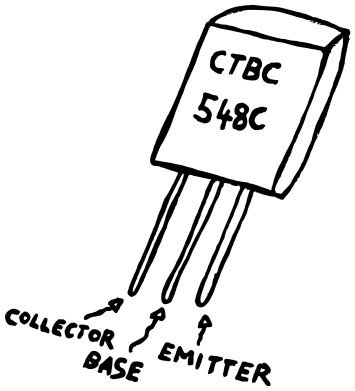
How does this work? Well, let's assume that we're applying a steady voltage of 5 V on the left. Then, as we already know, the op amp will subtract the inverting input's voltage from the non-inverting input's voltage, leaving us with a result of –5 V. Multiply that by the huge internal gain, and the op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

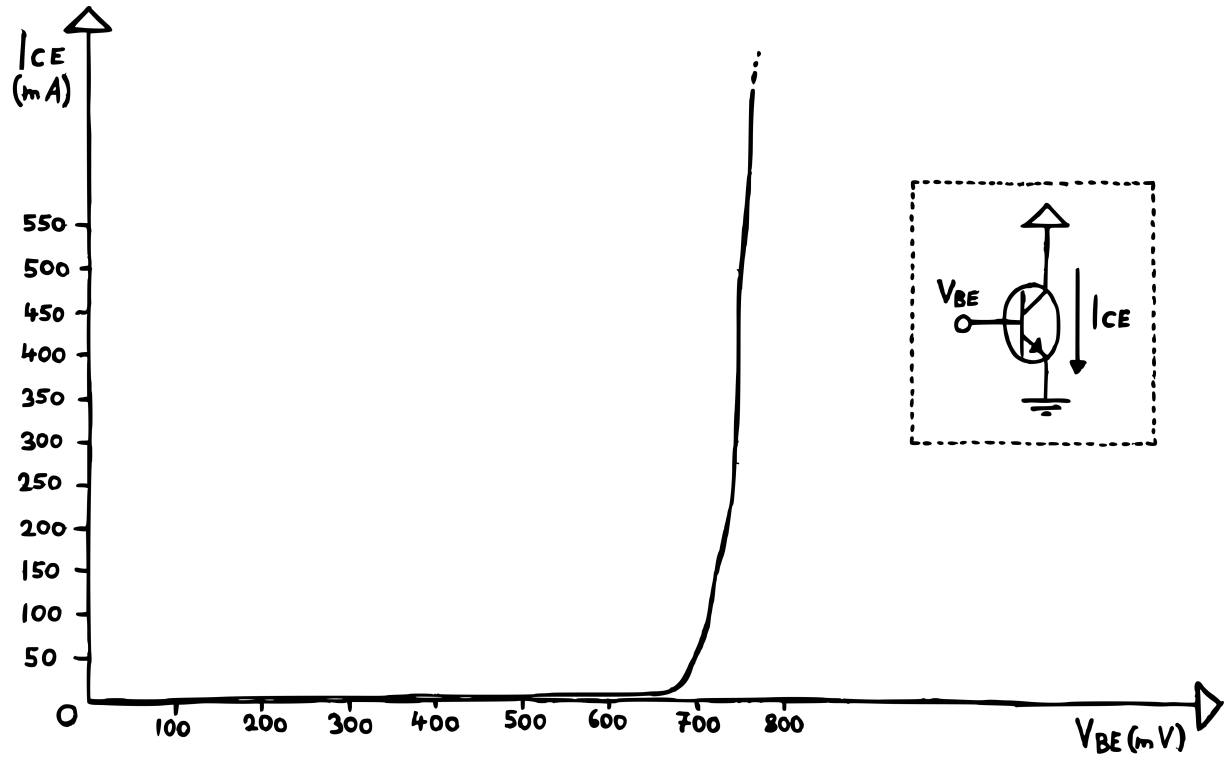
And that means that the voltage at the inverting input will be lowered to about 0 V, allowing our op-amp to settle on the current output voltage level. So while we read 5 V on the left, we'll now read a stable –5 V at the op amp's output. Congrats – we've built an inverting buffer! **If we want to turn 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**.[26] 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[27] 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.

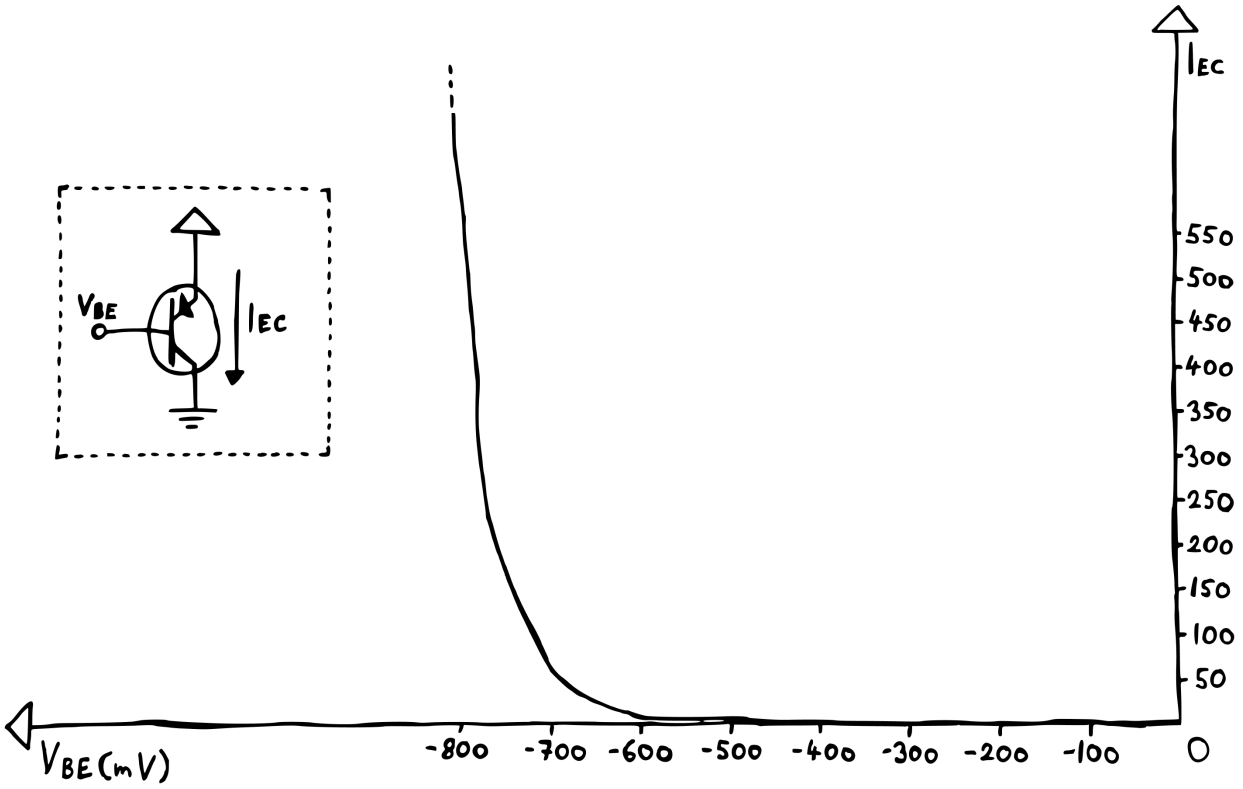


First, the relation between base voltage and collector current is exponential. Second, unlike a resistor, a BJT is not symmetrical – so we can't really reverse the direction of the

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

[27] 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**[28]. 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!

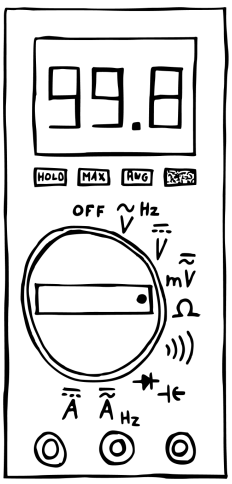


[28] Again, the voltage is measured between base and emitter.

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.

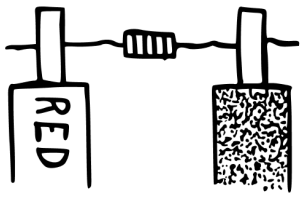
MULTIMETERS



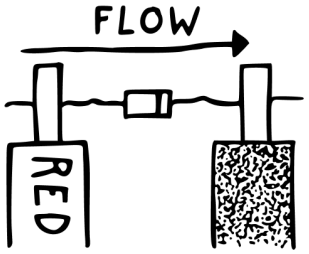
Multimeters come in different shapes and sizes, but the most common type is probably the hand-held, battery powered variant. It can measure a bunch of different things: voltage, current, resistance, continuity. Some have additional capabilities, allowing you to check capacitance, oscillation frequency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



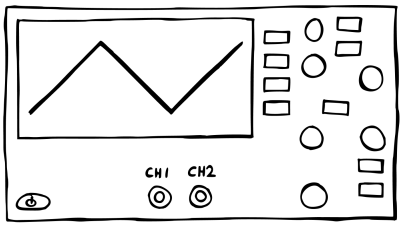
In some cases, it doesn't matter which probe you connect to which component leg or point in your circuit. This is true for testing resistors, non-polarized capacitors (foil/film, ceramic, teflon, glass etc.), continuity[29] or AC voltage.

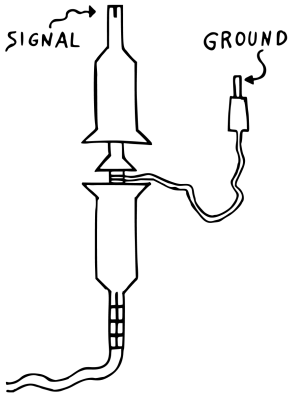


In others, you'll have to be careful about which probe you connect where. For testing the forward voltage drop of a diode, for example, **the multimeter tries to push a current from the red to the black probe**. Here, you'll have to make sure the diode is oriented correctly, so that it doesn't block 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.

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

OSCILLOSCOPES



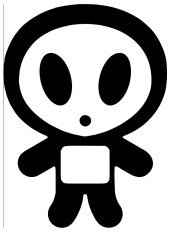


While multimeters are fairly cheap and compact, oscilloscopes are usually somewhat pricey and bulky. **If you're willing to make the investment, they are a huge help with the troubleshooting process, though**. Using one is, again, surprisingly straightforward – if you manage to work your way through 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 Sequencer** 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 **Clock input** jack socket, **XS2** is the **Clock output** jack socket; it basically duplicates clock input, so you can use it to synchronize other modules to the clock of the Sequencer. **XS3** is the **Reset input** jack socket, **XS4** is the **Gate output** jack socket and **XS5** is the **CV output** jack socket. In our designs, we use eurorack standard 3,5mm jack sockets (part number WQP-PJ301M-12).

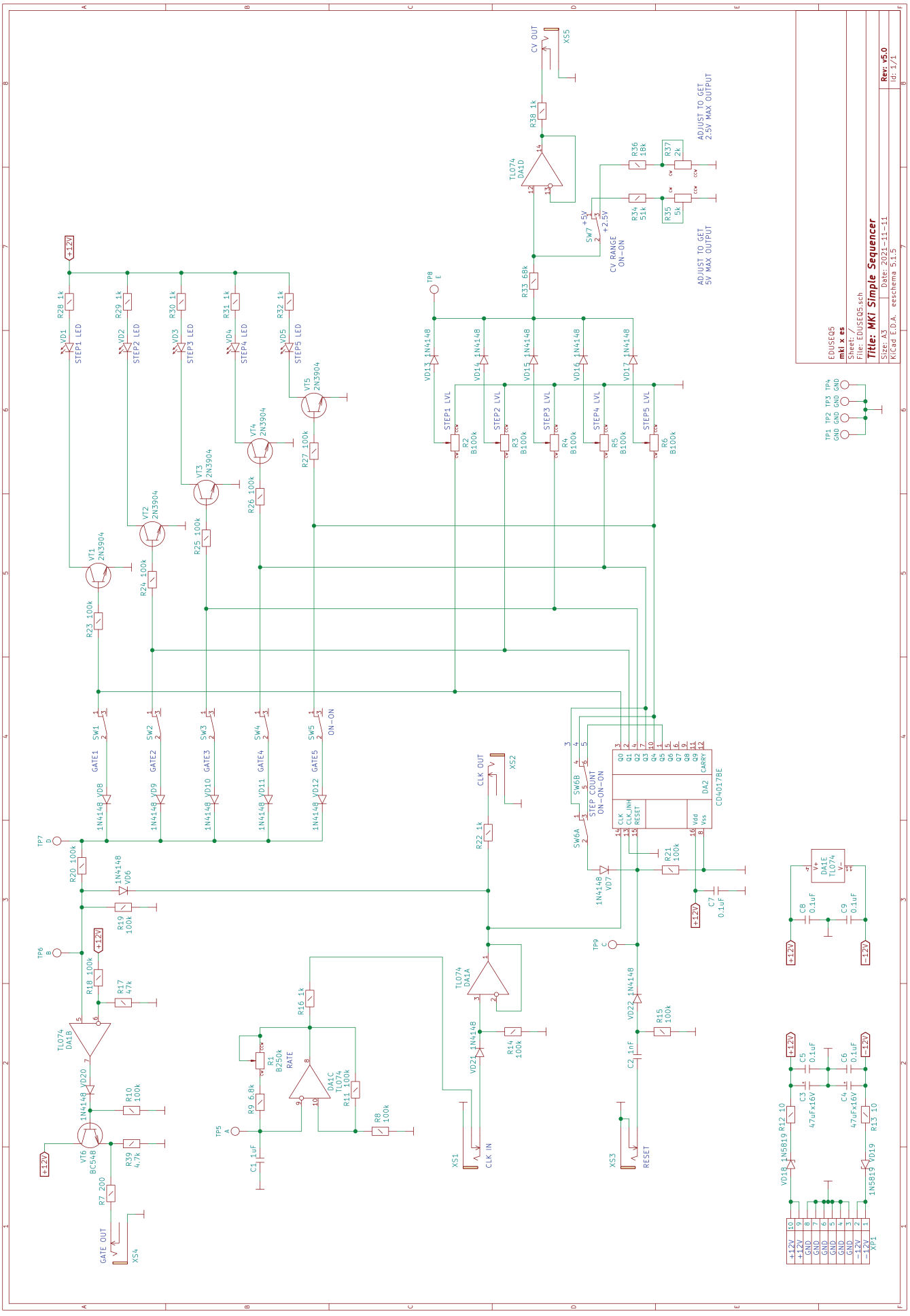
XP1 is a standard eurorack **power connector**. It's a 2x5 male pin header with a key (the black plastic shroud around the pins) to prevent accidental reverse polarity power supply connection. This is necessary because connecting the power incorrectly will permanently damage the module.

VD18 and **VD19** are **schottky diodes** that double-secure the reverse polarity power supply protection. Diodes pass current only in one direction. Because the anode of VD18 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 VD18, it closes, and no current passes through. The same goes for VD19, 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.

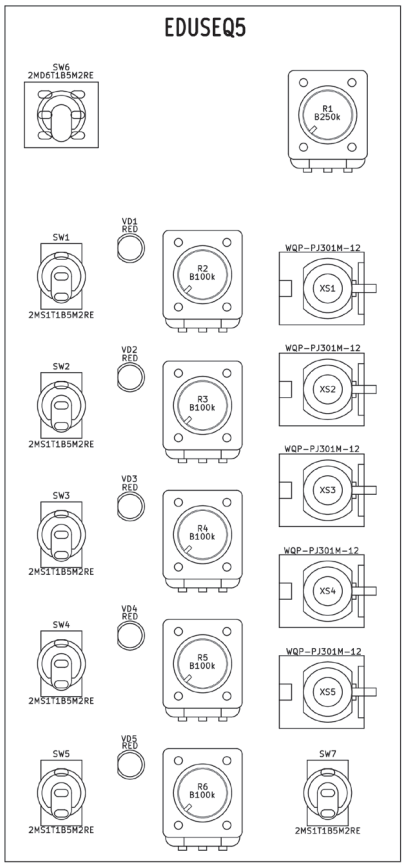
Next, we have two **10 Ohm resistors** (**R12** and **R13**) on the + and – 12 V rails, with **decoupling** (or **bypass**-) capacitors **C3** – **C6**. These capacitors serve as energy reservoirs that keep the module's internal supply voltages stable in case there are any fluctuations in the power supply of the entire modular system. In combination with R12 and R13, the large 47 microfarad pair (C3 and C4) compensates for low frequency fluctuations, while C5 and C6 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.

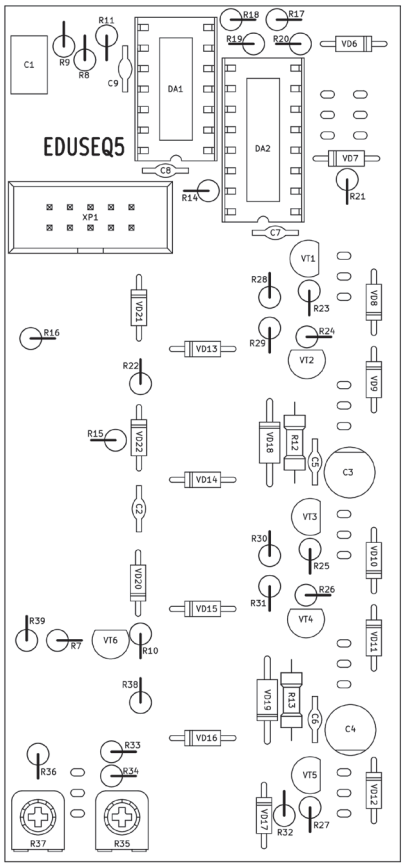
Another advantage of 10 Ohm resistors is that they will act like **slow** "**fuses**" in case there's an accidental short circuit somewhere on the PCB, or an integrated circuit (IC) is inserted backwards into a DIP socket. The resistor will get hot, begin smoking and finally break the connection. Even 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**.

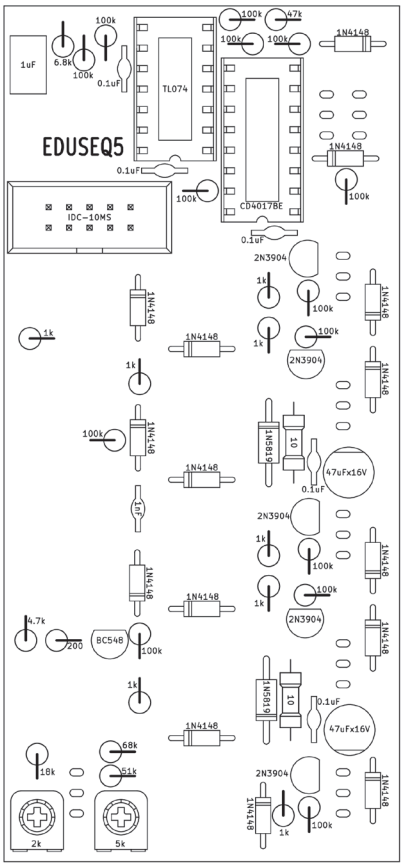
Capacitors **C8** and **C9** are additional decoupling capacitors. If you inspect the PCB, you'll see that these are placed as close to the power supply pins of the ICs as possible. For well-designed, larger PCBs you will find decoupling capacitors next to each IC. Like the others, their job is to simply 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.

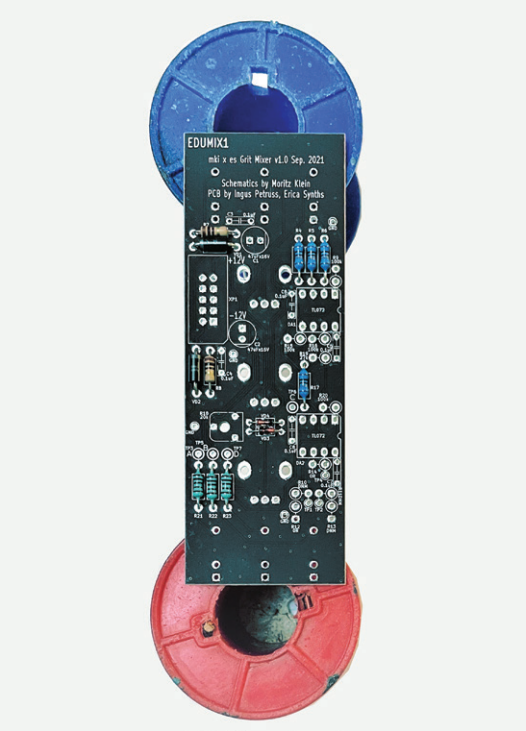

Title: MKi Simple Sequencer
Rev: v5.0
Size: A3
Date: 2021-11-11
KiCad E.D.A. eeschema 5.1.5
Id: 1/1
EDUSEQ5
mki x es
Sheet: /
File: EDUSEQ5.sch
CV OUT
CLK OUT
CLK IN
RESET
GATE OUT
ADJUST TO GET 2.5V MAX OUTPUT
ADJUST TO GET 5V MAX OUTPUT
CV RANGE ON-ON
STEP COUNT ON-ON-ON
ON-ON
+12V
-12V
+5V
+2.5V
STEP1 LED
STEP2 LED
STEP3 LED
STEP4 LED
STEP5 LED
STEP1 LVL
STEP2 LVL
STEP3 LVL
STEP4 LVL
STEP5 LVL
GATE1
GATE2
GATE3
GATE4
GATE5
RATE
2N3904
1N4148
1N5819
TL074
CD4017BE
BC548

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.

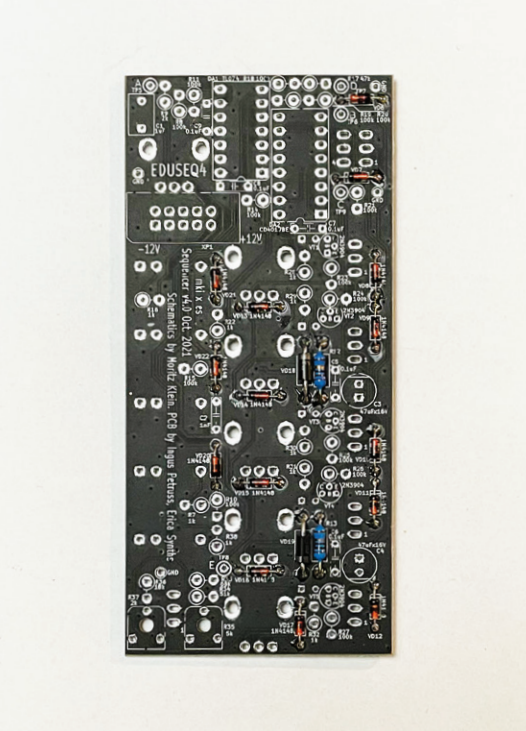




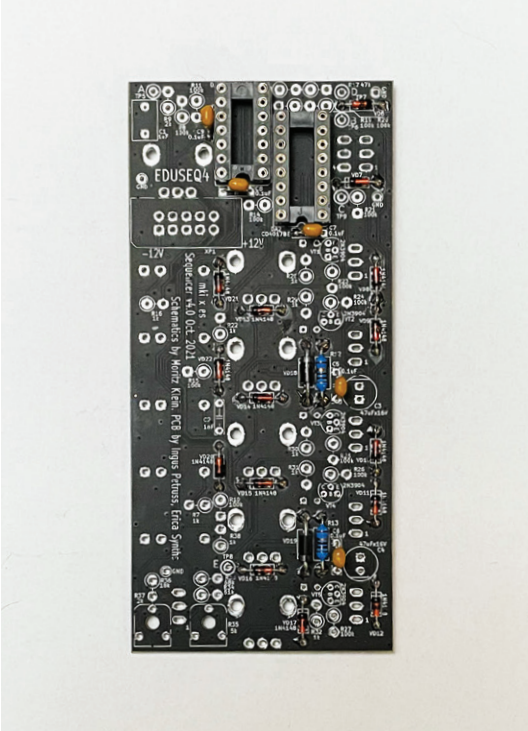




Place the Sequencer PCB in a PCB holder for soldering or simply on top of some spacers (I use two empty solder wire coils here).

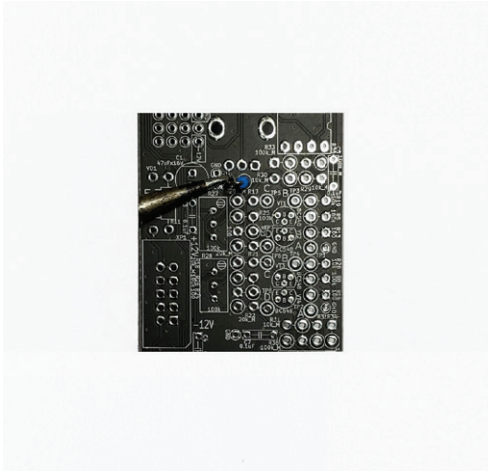


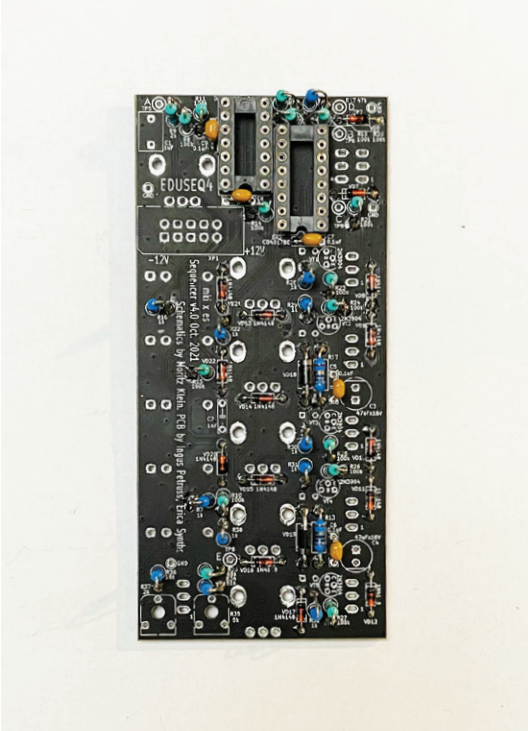
I usually start populating PCBs with lower, horizontally placed components. In this case, these are **switching diodes**, **10 ohm resistors** and the **power protection diodes**. Bend the resistor leads and insert them in the relevant places according to the part placement diagram above. All components 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.



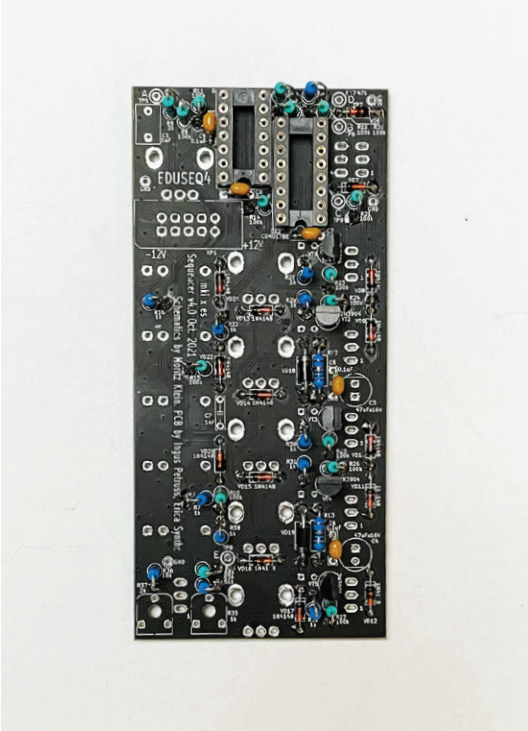
Next, insert the first DIP socket, hold it in place and solder one of the pins. Continue with the **next DIP socket**. Make sure the DIP sockets **are oriented correctly** – the notch on the socket should match the notch on the PCB's silkscreen. Now, turn the PCB around and solder all remaining pins of the DIP sockets. Then proceed with **the ceramic capacitors**. 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 Sequencer, have **vertically placed resistors**. The next step is to place & solder those. 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.



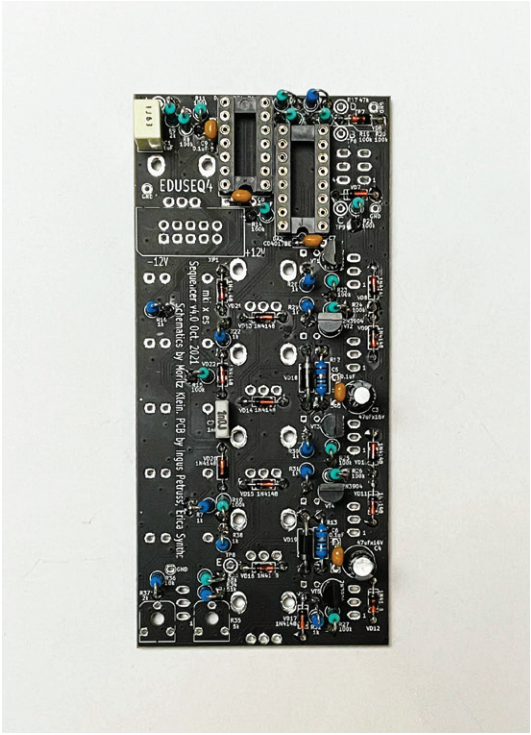


Once you are done with soldering all resistors, your PCB should look like this:

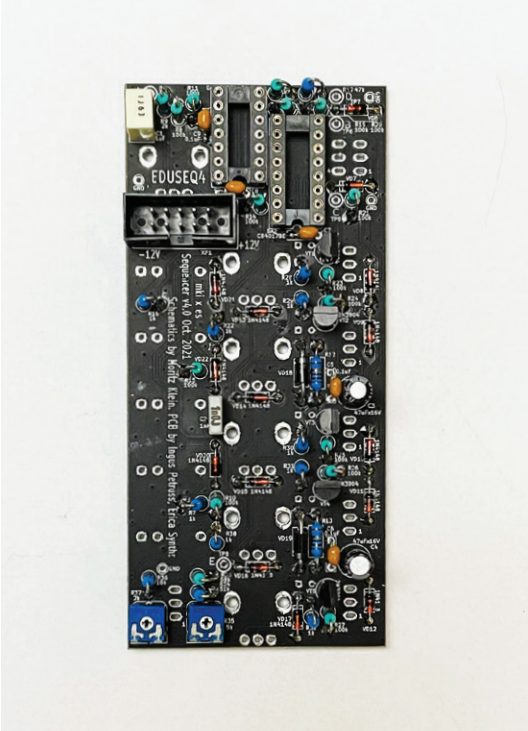


Next up: inserting & soldering transistors. Make sure you align the transistor with the marked outline on the silkscreen – **orientation is critically important here**.

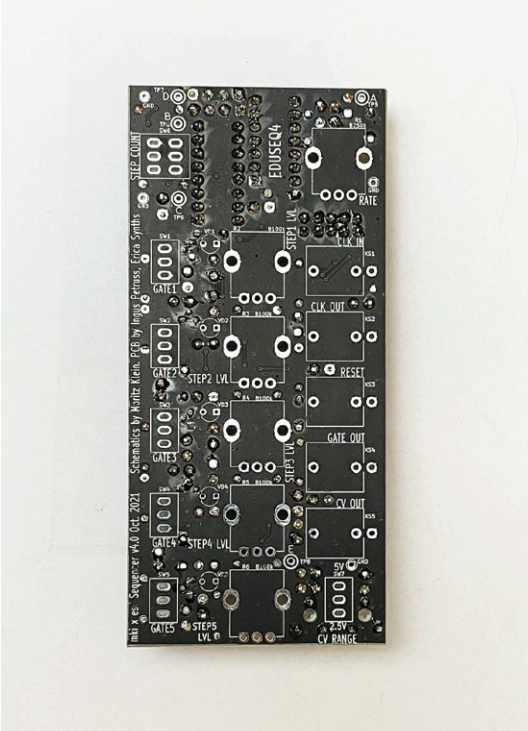
Now, insert **film capacitors** and solder them. Then, insert & solder **the electrolytic capacitors**. Electrolytic capacitors are bipolar, and you need to mind their orientation. The positive lead of each electrolytic capacitor is longer, and there is a minus stripe on the side of the capacitor's body 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.



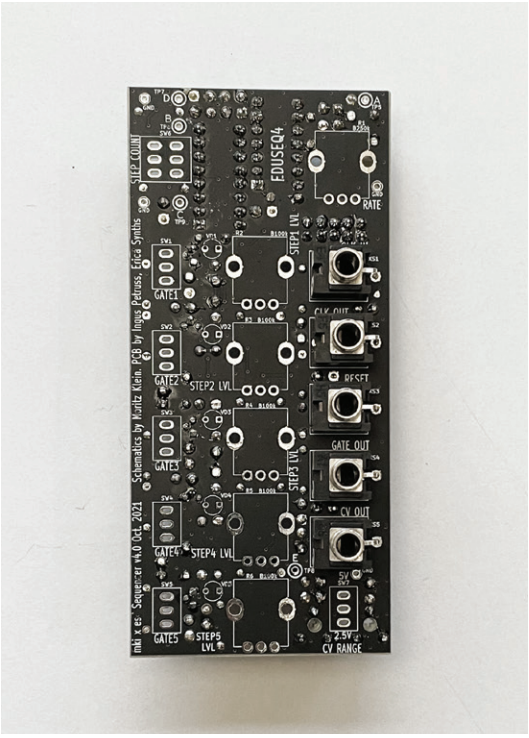
Now, your PCB should look like this.



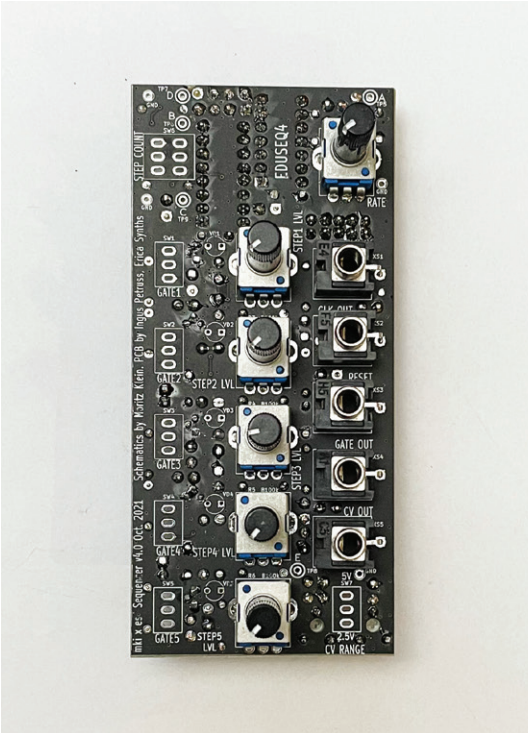
Then complete the component side of the Envelope PCB by soldering the **PSU socket** and **trimmer potentiometers**. Make sure the orientation of the socket is as shown in the picture below – the arrow pointing to the first pin is aligned with a notch on the silkscreen. **The key** on the socket will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



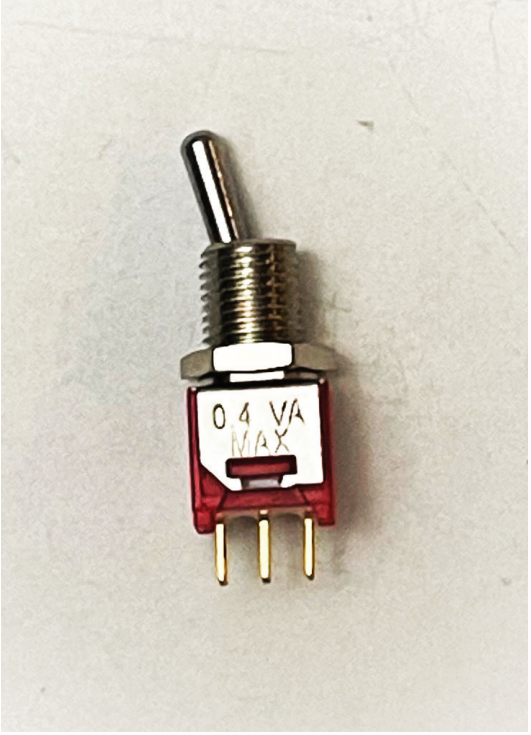
Now, turn the PCB around and inspect your solder joints. **Make sure all components are soldered properly and there are no cold solder joints or accidental shorts.** Clean the PCB to remove extra flux, if necessary.



Insert the jack sockets and solder them.

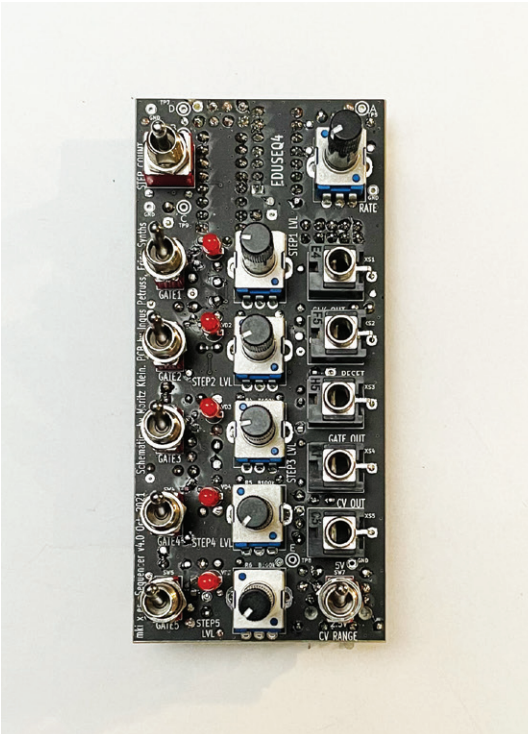


Insert **the potentiometers** (make sure, you have right values! The RATE potentiometer is 250 kOhm, step potentiometers are 100 kOhm), **but don't solder them yet**! Fit the front panel (you may want to fix it with few nuts on the jack sockets) and make sure that the potentiometer shafts are aligned with the holes in the panel – and that they're able to rotate freely. Now, go ahead and solder the potentiometers.



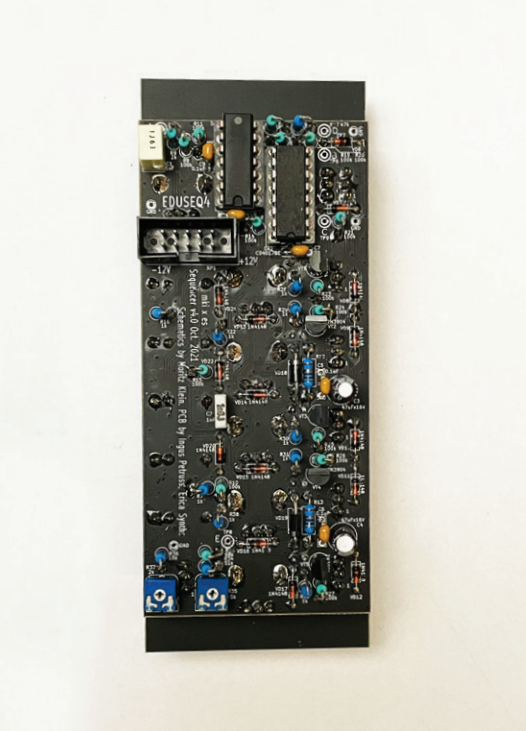
The **switches** require special attention. There are two nuts for the switch (they look identical to the jack socket nuts, but the thread is different). Screw on one of the nuts until it fixes itself on the bottom of the thread.

Now, insert **the switches** in the relevant places on the PCB, place the front panel, fix it with few nuts on the jack sockets and **solder the switches**.



Insert the LEDs in the relevant places on the PCB, but do not solder them, yet! Orientation of the LED is important – check the silkscreen! A notch on the silkscreen indicates the **cathode of the LED** (a shorter lead next to a notch on the LED) and the longer lead – the **anode of the LED** – has to go into a hole with square-like polygon on the PCB.

Fit the front panel again and **fix it** with some nuts on jack sockets and one nut on the switch. Now, solder the **LEDs**. We are almost done!



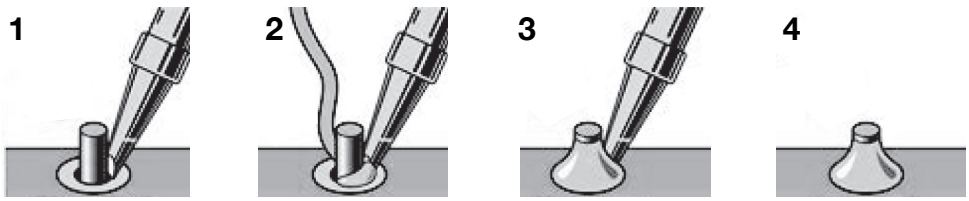
Now, **insert the ICs into their respective DIP sockets**. Mind the orientation of the ICs – match the notch on each IC with the one on its socket.

Congratulations! **You have completed the assembly of the mki x es.edu Sequencer module!** 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. The **step LEDs** have to light on sequentially; the **speed** has to change depending on **RATE potentiometer** setting. Check if the **STEP COUNT** switch **changes the length** of the sequence. If all above works, proceed with calibration as described in the build manual above. Adjust the **trimpots R35** and **R37** in order to achieve desired voltage on the **CV output**. We already have several modules that can be controlled by the Sequencer. It's time to patch them up and **enjoy some musical results**!

SOLDERING APPENDIX

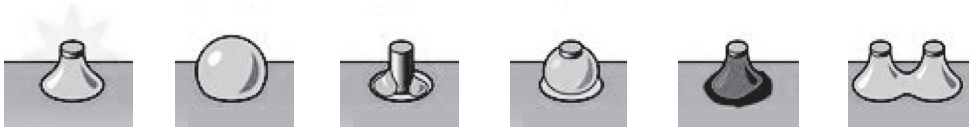
If you've never soldered before – or if your skills have become rusty – it's probably wise to check out some **THT** (through-hole technology) **soldering tutorials on YouTube**. The main thing you have to remember while soldering is that melted solder will flow towards higher temperature areas. So you 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:



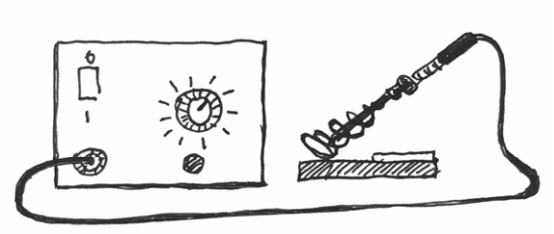
Heat part and pad 2 - 3 sec | Add solder | Continue heating 1 -2 sec. | Let cool

After you have completed soldering, inspect the solder joint:

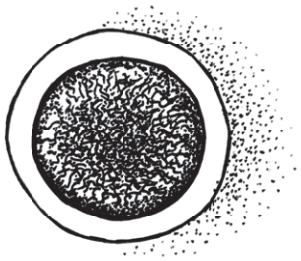


Perfect | Too much solder | Not enough solder | Cold joint | Too much heat | Short

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:

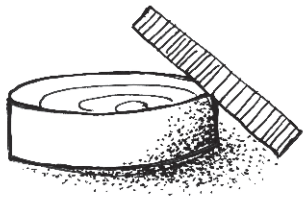


A decent soldering station. Top-of-the-line soldering stations (brands like Weller) will cost 200€ and above, but cheaper alternatives around 50€ are often good enough. Make sure your soldering station of choice comes with multiple differently-sized soldering iron tips. The most useful ones for DIY electronics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



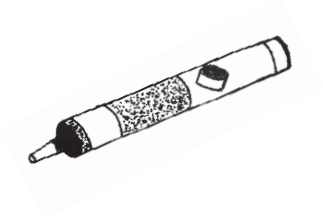
When heated up, the tips of soldering irons tend to oxidize. As a result, solder won't stick to them, so you'll need to clean your tip frequently. Most soldering stations come with a **damp sponge for cleaning the iron tips** – but there are also professional solder tip cleaners with **golden curls** (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.

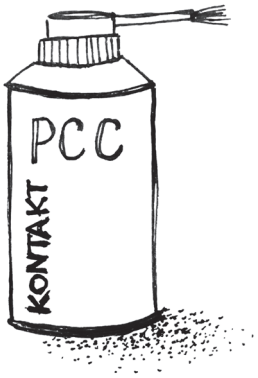


Some **soldering flux** paste or pen will be useful as well.

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.

"Satisfaction of one's curiosity is one of the greatest sources of happiness in life."

– Linus Pauling