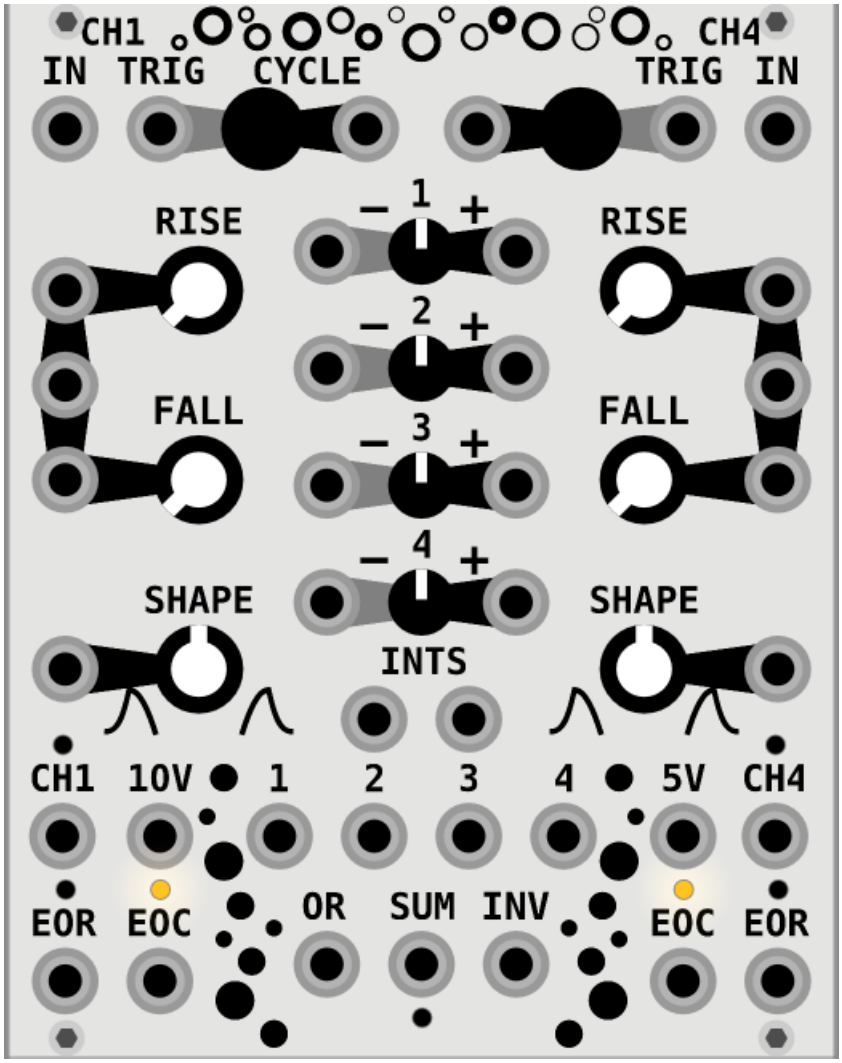
Documentation

Table of contents

01.-Module-Overview.md



Learning all of the ins and outs of Floats.

Floats is a module that can become any other module you can imagine with some clever patching. This can be intimidating, but we will start with the basics and go from there.

Before progressing, be sure to download the example patches! **Click Me!**

Using Channel 1 as an LFO

Now we will look at: **EX01BasicLFO**

The most basic use for Floats is as an LFO. This can be achieved by pressing the cycle button. When the Cycle button is on, it will glow red.

Now that you have turned on the cycle button, you will see that the scope is filled with blue. This is because the default value for the rise and fall time of Floats is set to its fastest rate. Raise the level of both the rise and fall times. You will now be able to see the shape of the oscillation. At this point, you should experiment with the rise, fall and shape knobs.

CV Controlling the Rise and Fall time

For this example, keep **EX01BasicLFO** open.

Each channel of Floats allows you to control the Rise and Fall time with external CV. These CV inputs are directly connected to their corresponding knob via thick black lines. We will now make our first Floats feedback patch! Take the **2** output located at the bottom middle of the module and run it to both the Rise and Fall CV inputs.

Once you have done that, you can now see that both the Rise and Fall times will increase at the same rate if you move the attenuverter labeled **2**. If you reset both Rise and Fall to their default rate, you can easily make a Triangle LFO. You can also get a square wave LFO by using the End of Rise or **EOR** output! You can get different pulse widths by independently changing the rise and fall time.

Exponential CV control

For this example, keep **EX01BasicLFO** open.

You may have noticed the jack between the Rise and Fall CV inputs. That is the exponential CV input. This input is more commonly known as the V/OCT CV input. Disconnect the two patch leads from the Rise and Fall CV inputs. Now run one patch lead from the **2** output to the exponential CV input. If you send a positive voltage, both the Rise and Fall rate will increase. If you send a negative voltage, both the Rise and Fall rate will decrease. Once you are done experimenting with this, we will make our second Floats feedback patch.

Channel 1 is routed to attenuverter **1**. This means that there is an attenuverted copy of Channel 1 on the **1** output. Disconnect **2** from the exponential CV input, and reset attenuverter **2** by right clicking on it. Now run the **1** output to the exponential CV in. Set the Rise and Fall time to a point where you can see the shape on the scope, and reset the Shape knob with a right click if you haven't already. We will now change the shape of the LFO by feeding the LFO back into itself. This is done by changing the value of attenuverter **1**. As you change the attenuverter, you will now see different shapes appear on the scope. You should also notice that changing the shape in this manner also changes the time it take for the LFO to cycle.

Changing the Range of Floats

For this example, keep **EX01BasicLFO** open.

By default, the rise and fall knobs cover a range of less than a millisecond to exactly 15 seconds. You can change this range by sending voltages to the exponential CV input. If you send negative voltages, the range will get wider. This is ideal for making very slow LFO cycles. The absolute longest time for a rising or falling segment is exactly 256 minutes! If both segments are set to the highest time, you will have a triangle LFO that takes just over 8.5 hours to repeat!

Of course Floats can also cover the full audio range too. This is done by tuning Floats to a reference oscillator then applying positive voltage to the exponential CV input. We will now quickly cover how to use Floats as an audio rate oscillator below before moving on.

Floats as an Audio Rate Oscillator

Now we will look at: **EX02BasicOSC**

As you have learned by now Floats has a spectacular number of shapes at its disposal. It would be an absolute waste to not use them as an audible oscillator!

The patch is already connected to be a simple synthesizer voice. You will just need to select your audio device on the Audio Driver module and your MIDI device on the MIDI to CV module. Once you have those set up, just have fun! If you do not have a MIDI device, turn on cycle for Channel 4.

There are two important things to note about this patch. The first is that we are using the **SUM** out. We need to shift Channel one down by 5 Volts to put it in the usual +-5 Volt range for oscillators. We do this by setting attenuverter **1** all the way to the right and by setting attenuverter **3** all the way to the left. This works because channel 3 is just a static value of 5 Volts, and by turning the attenuverter fully negative we subtract 5 Volts from the channel 1 output that cycles between 0 Volts and 10 Volts. The second important thing to note is that we are using the Channel 4 trigger or **TRIG** input. This is perfect for using Floats as an envelope or one shot LFO.

Make sure to experiment with all of the blue knobs. Now is a perfect time to remind you that you can CTRL+click on a knob to fine tune it. This is ideal for changing the tuning on Channel 1 of Floats.

Using the Trigger in and Cycle CV Input

Now we will look at: **EX03TriggeringAndCycling**

From here on out the patches will be a bit more complex. In this patch I have already set up a Square wave LFO to act as a clock that will retrigger Channel 1 over and over again. I have then taken the **EOR** out of Channel 1 and routed it to the Cycle CV input. If you look in the scope you will see that Channel 4 will cycle as Channel 1 falls. Change the fall time of Channel 1 to change the number of cycle on Channel 4. This setup is commonly called a burst generator.

What if we want our Channel 1 Fall time to remain the same, but only get a single cycle out of Channel 4? Simply move the **EOR** out from the Cycle CV input to the **TRIG** input on Channel 4. You can delay how long it takes for Channel 4 to trigger by changing the Channel 1 Rise time. This setup is commonly called a gate delay.

These two basic setups are important to know for building more complex CV patterns. I would strongly encourage you to experiment with both setups to build a wide number rhythmic curves and shapes! As one last piece of advice, use the Channel 1 and 4 attenuverters to mix the two

channels together in interesting ways. You can use the **SUM** output to see the combined waveshapes. You should also experiment with the **OR** output that will only output the maximum voltage.

Before moving on, can you find a way to cycle Channel 4 without using the Cycle button or Cycle CV input? I'll give you a hint. **EOC** means end of cycle.

Clocking and Slewing

Now we will look at: **EX04ClockingAndSlewing**

If you have been following along with the manual until this point, you will know that we have covered every part of Floats except the two inputs labeled **IN**.

In this patch we are using Channel 1 of Floats to clock the sample and hold of the Audible Instruments Utilities. We will then process the sample and hold output with Channel 4 of Floats by running it to the **IN** of Channel 4. Now you can change the Rise, Fall, and Shape of Channel 4 to change how the sample and hold is processed. You can even feedback the attenuverter **4** output back to the exponential in on Channel 4 to get more shapes. The time between random samples can be changed with the Channel 1 Rise knob. Once you are done experimenting with the sample and hold, we will learn how to use the **IN** as an attack sustain release envelope or ASR envelope for short.

Making an ASR envelope is one of the most important uses of the two **IN** jacks on Floats when it comes to modulation. To make an ASR envelope, simply run a gate signal to an **IN** then adjust the Rise, Fall, and Shape however you wish. This will commonly be done with the MIDI to CV gate source, but you can also use a square wave LFO or even the **EOR** output on Channel 1.

Moving On

If you have followed along with all of the above examples, you are ready to move on to more advanced patches! However, we will now do a quick recap of the most important aspects of Floats.

- Floats has a default Rise and Fall time that ranges from less than a millisecond to 15 seconds.
    - This range can be extended by running a negative voltage to the exponential CV input.
- Floats can act as an oscillator by turning on the Cycle button.
    - The exponential CV input gives you control of the Rise and Fall times over a 20 octave range.
    - Tuning Floats to an external audio rate oscillator lets you use it as a normal oscillator.
- The **TRIG** inputs will only trigger the oscillators to cycle one time.
- The Cycle CV input will cause the oscillators to cycle as long as the voltage at the input is over 1 Volt.
- The two **IN** inputs are used when you want to directly process a signal with the Floats Rise and Fall time.
    - This is useful for making simple ASR envelopes.

To be clear, gaining full mastery of Floats mostly comes down to knowing what range to set your Rise and Fall times to for a given application. This is why we spent so long with the basic LFO in Example 01. Then after that, you will want to master using the four attenuverters to mix all four channels together in new and interesting ways. The next page of this manual will cover building more advanced modulations.

02.-Advanced-Modulation.md

Advanced Modulation

Floats has the ability to create many common forms of modulation with only one Channel, but we will now explore patches that use both Channel 1 and 4 of Floats to create even more complex patches. From this point on, I will try to keep explanations to a minimum as full explanations of each part for these patches can get very verbose.

Making an ADSR Envelope

Now we will look at: **EX05MakeAnADSR**

As you are already aware, Floats can make both AD envelopes and ASR envelopes. If we combine both an AD and ASR envelope we will get an ADSR envelope!

The attack time is controlled by attenuverter **2**. The decay time is set by Channel 4 Fall. The Sustain level is controlled by attenuverter **1**. The Release time is set by Channel 1 Fall. Open the patch and experiment with these four knobs to see if you can figure out why they work together to build an ADSR.

There are five important things to know about this patch.

- The gate source is run to the **IN** of Channel 1 and the **TRIG** of Channel 4.
- The ASR is generated by Channel 1 and the AD is generated by Channel 4.
- The attack time of the envelope is controlled by both the Channel 1 and 4 Rise time.
    - This is why we use attenuverter **2** to control both Rise times at the same time!
- Attenuverter **4** is always set all the way to the right. This is because the attack and decay portions of an ADSR envelope always reach 10 Volts.
- We use the **OR** out as it will only output the maximum voltage at any time.
    - This is how we combine Channel 1 and 4 into a single ADSR.

Now we will look at a fine detail of that last point. How can we get such a clean ADSR signal when we are also sending a static voltage from attenuverter **2** to the **OR** output? This is actually very simple. When you connect a patch lead to outputs **1** through **4** you remove them from the **OR**, **SUM**, and **INV** output calculations. You can prove this by changing the level of attenuverter **3** then sending its output to somewhere else in the patch. I would recommend connecting it to the Channel **3** input that is under the S of Floats.

Before moving on, why not plug in a MIDI keyboard and have some fun with the patch?

Quadrature LFO

Now we will look at: **EX06QuadratureLFO**

This patch is actually very simple, but we will use it to discuss **EOR**, **EOC**, and how retriggering the Channels works. We use attenuverter **2** to control the Rise and Fall times on both Channel 1 and 4 to get two even triangle waves. The **EOR** out of Channel 1 is routed to the **TRIG** in of Channel 4. In return, the **EOC** out is routed back to the **TRIG** in of Channel 1. You will need to momentarily engage the Cycle button momentarily to start the cycling.

We will now examine when the **EOR** and **EOC** outputs go high. The **EOR** output goes high when Channel 1 ***RISES*** to it's final destination. **EOR** will also stay high as the signal is falling. The **EOC** output will go high when Channel 4 ***FALLS*** to it's final destination and stays at it. **EOC** will not go high when Channel 4 is cycling as it never stays at the final destination. This is also why **EOC** is not high when the module first starts, Channel 4 needs to have tried to cycle at least one time to reach the end of its cycle.

Before moving on, experement by setting the rise and fall times to be different from each other. It is possible to build many unique LFO patterns by mixing Channel 1 and 4 and using the SUM output. I would recommend starting by making Channel 4 negative.

True Quadrature LFO

Now we will look at: **EX07TrueQuadratureLFO**

The LFO above can not be a true quadrature LFO as it only has two LFOs and not four. This example uses four Floats hooked together to

make a true quadrature LFO. You will need to momentarily engage the Cycle button Channel 1 of the upper left Floats. You can set the cycle time of all LFOs with attenuverter **2** on the upper left Floats.

This patch is great for making very complex and rhythmic shapes. You can randomly turn the cycle on then off for any of the channels and you will generate a new pattern. You can use the spare channels on Floats to modulate the exponential CV input or even ping the Cycle CV input.

Checkout **EX07TrueQuadratureLFOCrazy** to see how crazy four Floats in a feedback patch can get! Before moving on, it is also possible to build a series of LFOs that are chained together via the **EOC** out. The crazy example has an example of this type of linking. This means that you will also need to momentarily engage the Channel 4 Cycle button on the upper left Floats.

Burst Generator

Now we will look at: **EX08BurstGenerator**

The burst generator was already briefly looked at in the previous section of the manual, but we will quickly recap it here. The burst generator is a great way to make an event repeat multiple times when a gate signal is received. A gate signal triggers Channel 1. While Channel 1 is falling, the **EOR** is high and is sent to the Cycle CV input of Channel 4. This means that Channel 4 will cycle for as long as Channel 1 is falling.

Next, we will look at a more interesting evolution of the burst generator.

Bouncing Ball

Now we will look at: **EX09BouncingBall**

This setup is exactly the same as the Burst Generator. However, we are also using the curve of Channel 1 to control the rate and amplitude of Channel 4. We use the linear input of the VCA module so that we can set the amplitude fall off from Channel 1 using its shape knob. We also route the **1** output to the exponential CV input of Channel 4. This will make it so that the rate of Channel 4 increases as its amplitude decreases. This is exactly the same as if you had dropped a bouncing ball as it will bounce more often as it gets closer to the ground.

You can play around with all of the parameters to create complex LFO shapes and even reverse the effects of gravity by changing the polarity of attenuverter **1**.

Envelope Follower

Now we will look at: **EX10EnvelopeFollower**

Any bidirectional slew can be used as an envelope follower. The secret is setting the fall time longer than the rise time. This preserves the snappy peaks of drums while smoothing out the input oscillation.

This patch also uses Channels 2 and 3 to make a full wave rectifier to more accurately represent the envelope. A full wave rectifier is made by running the same signal to Channels 2 and 3 then setting attenuverter **2** fully to the right while setting attenuverter **3** fully to the left. You then use the **OR** output. You can make a half wave rectifier by only using one channel then taking the **OR** output.

Change the envelope on Channel 4 of Floats and watch on the Scope how the blue envelope is pulled from the purple oscillation.

Logic Comparator

Now we will look at: **EX11LogicComparator**

In this patch we use the way the **EOR** and **EOC** functions work to pull logic signals from square wave LFOs. The **EOR** will copy the logic state of the square wave LFO as shown on the left most Scope. The **EOC** will be opposite the logic state of the square wave LFO as shown on the right most Scope. We then run the **EOR** to Channel 2 and the **EOC** to Channel 3. The **OR** output and the **INV** are output to the middle scope. You can change the way the logic comparator works by changing the polarity of attenuverters **2** and **3**. This can be used to build more complex trigger sequences.

All logic signals generated in this manner will always have a low state of 0 Volts and a high state of 10 Volts no matter what the external voltage input is oscillating between. You can test this by setting the LFOs to be bipolar instead of unipolar. You can also experiment by attenuating them as well.

Gate Delay

Now we will look at: **EX12GateDelay**

We momentarily covered this in the previous section of the manual, but we will now examine a very important use of the gate delay. If you watch the Scope, you can see that I have made it so that the phase of the gate delay is exactly 180 degrees off from the LFO input.

Now increase the rate of the LFO. By watching the Scope, you can see that the space between the **EOR** is still the same as when the LFO was slower. This turns the gate delay into a clock divider. However, you can get clock divisions of any length including fractions of beats instead of just integers. This is very useful for adding swing and variance to drum sequences.

- Remember, the cycle can not be reset until it reaches the falling state. This is why it acts as a clock divider instead of simply resetting and never reaching the end of rise state.

If you slow down the LFO again, you will once again get a gate delay.

Moving On

You should now be familiar with how combining the simple elements of Float can build complex and specific modulation sources. I would once again like to raise the importance of modulation rates when using Floats. As in the Gate delay example, both the rise and fall time of Floats were vital to the use of the gate delay, but the external LFO set whether it acted as a gate delay or a clock divider.

The next section of the manual will cover advanced audio rate uses for Floats. These include using it as an audio rate oscillator as well as a low pass gate.

03.-Advanced-Oscillators.md

Advanced Oscillators

Floats is a fantastic oscillator. If you need a fresh source of waveshapes and timbres, your search ends here. Floats also makes a wonderful amplifier and waveshaper as you will discover in our first patch.

A Full Synthesizer Voice

Now we will look at: **EX13FullSynthVoice**

All sound is generated and shaped by only two Floats modules. The Floats on the left is a dual oscillator and the Floats on the right is a low pass gate with a dual stage distortion and built in envelope. The example is great for making plucked basses and unique drums.

First we will look at the Floats to the left that is setup to be a dual oscillator. The main thing to master when using Floats as an oscillator is to understand that changing the timbre with the rise and fall also affects the oscillator's pitch. This means that Floats is more suited as a complex oscillator, but using it as a more typical East Coast oscillator is nice as well. Both Channel 1 and 4 are set up as rising saw oscillators. This means that the tuning of the oscillator is controlled by only the rise parameter. We use the **SUM** out to add the two oscillators together. We also use attenuverter **2** to add -10 Volts to the overall oscillations. We need to add -10 Volts as the range of Channel 1 plus Channel 4 is 0 Volts to 20 Volts, and we want it to oscillate from -10 Volts to 10 Volts.

Now we will look at the Floats to the right that is our low pass gate and envelope. Channel 4 is set up as a basic AD envelope. However, we need to set its range to be from -10 Volts to 0 Volts by setting attenuverter **2** fully to the left and attenuverter **4** fully to the right. This is why we use the **SUM** output and run that to the exponential CV in of Channel 1. Why does the range need to be set this way? The exponential CV in of Floats will increase the time it takes for the slew to meet its target voltage as more negative voltage is introduced. This means that our low pass gate is fully closed when we have sent -10 Volts to the exponential CV input, and it is fully open when we send 0 Volts to it.

Now that we understand how opening and closing the low pass gate works, let's look at the dual feedback drive that is setup. As you will recall, we are sending the **SUM** out to the exponential CV input. This means that we can feedback Channel 1 and Channel 3 into our low pass gate! Channel 3 is just a copy of Channel 1 in this instance. Set attenuverters **1** and **3** to different levels to get more distorted timbres. I personally like adding a bit of negative feedback on Channel 1. Definitely try this out on the bass notes!

This patch starts simple, but it can become incredibly complex with the addition of another Floats or two for extra modulation. I would also encourage you to break from the usual oscillator setup that we are currently at and experiment with the waveshapes you have been discovering throughout this manual!

PWM Oscillator

Now we will look at: **EX14PWM**

While not the most exciting waveshape in the world, some bloke named Nick has deemed it a necessity for every oscillator!

This patch is set up to be a demo of how to make a PWM oscillator. As it involves a bit of balancing to get good results, there is not a lot going on in the patch. You can add a MIDI to CV module into the patch if you want to play with it.

Channel 4 is our main oscillator, and its fall time sets the pitch of the square wave. We then modulate the rise time on Channel 1 to change the pulse width. As you know by now, the **EOR** out is low while Channel 1 is rising. This is why we are able to set our pulse width from the channel 1 rise time.

You should be able to follow the patch leads and figure out why the patch is set up the way it is by now. It is good practice for eventually building on top of this patch to achieve more exotic timbres.

- A small rant about PWM of other wave shapes.
    - The PWM shaping on the triangle and saw waves of the AX60/AX80 is just normal square wave PWM mixed on top of a saw or triangle wave. There is nothing magic about it. I always see this on the internet sighted as some magic oscillator that has special shaping for triangle and saw waves. Neither of them do. It is just a square wave on top of a saw or triangle. Some oscillators do let you change the balance of rise and fall time of a triangle wave to morph it into a rising saw or falling saw shape. This manual is all about an oscillator that will let you do just that!

Complex Oscillator

Now we will look at: **EX15ComplexOscillator**

At long last we demonstrate the most basic complex oscillator patch that can be made with a single Floats module. This is a patch that needs to be played with more than it needs to be explained. When it starts, there is no feedback modulation present on the output, so it just sounds like a standard triangle bass wave. However, by changing the levels of attenuverter **1** and **4** you will create an FM feedback loop! This will have drastic and unpredictable impact on the timber of the output. You will then want to mess around with all of the blue knobs. Even small adjustments to these knobs will completely change the way the FM feeback loop behaves.

As with the previous PWM patch, I highly encourage you to pull it apart yourself and maybe even add another Floats to the mix! You can use the current setup as a complex modulator for another oscillator or build a massive feedback matrix for cloud and drone oscillators. You can load up many instances of Floats, so there is basically no limitation to how massive these things can get.

In Conclusion

If you have followed the manual and experimented with each example patch, you should now understand the basics of building many types of patches with Floats. This manual is by no means an exhaustive list of all that can be done with Floats. There are plenty of articles and videos on the internet that will expand upon what you can do with a dual slope generator. These things have been around since the 70's, and people are still finding new and exciting uses for them. As you now have a nearly unlimited number of them, you can pioneer new kinds of patches that would be practically impossible to make with hardware.

There is nothing wrong with sticking to the basics, but those who push beyond standard convention will find Floats a lifelong source of inspiration.

04.-Community-Patches.md

Community Patches

Right now the community patches section is empty, but you can help by adding your patches! If you discover something you would like to share with the community, it would be wise to add it to the manual. Just make an issue on this manual, and we will get it added to this page.

Home.md

Welcome to the FloatsManual wiki!