Audulus 3 Module Library Documentation

Mark Boyd

February 22, 2019

Contents

Chapter 1

Modules and How They Work

Modular synthesis is an exciting, freeing way to create music and design sounds. The freedom it offers can be intimidating at first, especially to a total beginner, but grasping the fundamentals is easier than it appears.

All Audulus modules are created using Audulus nodes. You can open up any module to see how it works. Some modules are very simple and may have only a few nodes inside. Others are a web of interconnected submodules, sometimes many layers deep.

The great thing about the Audulus module library is that you don't need to understand how each module is constructed to be able to use it. There are, however, a few things you need to know about the module library before getting started.

1.1 Signal Standards

Modules in the Audulus library operate with a set of standardized signals. The job of the modules is to generate and modify these signals in interesting ways.

There are four main signal types: Gate, Modulation, 1/Octave, and Audio. As a general rule, you will connect like with like, meaning connect a Gate output to a Gate input, or an Audio output to an Audio input.

There are no hard-and-fast distinctions made between these signals in Audulus. All signals are processed at audio rate and are interchangable insomuch as Audulus will allow you to connect any output to any input.

You can multiply a modulation signal by a gate signal, subtract a mod-

ulation signal from a 1/Oct signal, or add a 1/Octave signal to an audio signal. However, not all of these combinations will yield useful results, and for beginners, it is best to just connect like outputs to like inputs.

1.1.1 Gate

The Gate signal is used to drive sequencers, open and close envelopes, and syncronize tempo, among other things. Its signal range is 0 or 1: off or on.

Gates are generated mainly by clock modules and keyboard or MIDI input. You can modify gate signals with modules like clock dividers and multipliers, gate sequencers, and Bernoulli gates. They are used by modules like sequencers, envelopes, and LFOs.

Gate inputs and outputs are always marked with a green light which is also called the Light node. When a gate input or output is equal to 0, the color of the light is dark green, and when the output is equal to 1, the color of the light is a bright green.

The only exception to this rule is when a gate is entering an envelope. Here, gate height, or the upper value of the gate, will set the maximum value that the Attack period rises to. This allows you to play with dynamics, or making your sounds softer or louder.

Gate inputs and outputs may be marked Gate, Gte, Reset or Rset, PWM, or Clk. Sometimes they may be unmarked except for the light node inside each input or output. These labels will have contextual meanings for each specific module, which are explained in their individual entries in the manual below.

Most modules will respond to only the rising edge of a gate. The rising edge is the moment where a gate transitions from 0 to 1. For example: a step sequencer will only step forward at the rising edge, but will not step forward again on the falling edge.

Other modules, like an envelope, respond to both the rising and falling edge of the gate signal. The rising edge of the gate will initiate the attack stage of the envelope whereas the falling edge will initiate the release stage.

If you are familiar with modular synthesis, you might be wondering if there is a difference between a pulse and a gate in Audulus. A pulse is a very short on/off burst, usually generated by clock modules, whereas gate signals can be any length from very short to very long.

In Audulus, no distinction is made between a pulse and a gate. That said, several clock modules give you control over the pulse width of their gate output. Pulse width is the ratio of on to off time. A 10% pulse-width gate is on 10% of the time and off 90% of the time whereas a 90% pulse-width

gate is on 90% of the time and off 10% of the time.

1.1.2 Modulation

The Modulation signal is used to tweak parameters like filter cutoff, VCO shape, and VCA level. Its signal range is 0 to 1.

Modulation signals are genereated by modules like LFOs, envelopes, sequencers, and sample & holds. They can be used to modulate any knob on any module and often have their own separate inputs as well.

Modulation inputs and outputs are often marked with a red light. The light indicates the relative strength of the incoming modulation. If the light is black (not lit) its value is 0. If the light is fully red its value is 1.

When a module has several outputs like the Basic LFO, lights may be omitted to save CPU time. If a portion of a module does not terminate in a Meter node or audio output, then everything that precedes it will not be calculated. This means if you use only the Sine output of the Basic LFO, only the sine path will be calculated. If lights were present at each output they would force Audulus to calculate the unused paths, wasting CPU time.

Modulation inputs and outputs may be marked Mod, Env (Envelope), a specific waveshape like Sine or Saw, or correspond to a knob label like Hz Mod or Q. Sometimes they may be unmarked and have contextual meanings for each specific module, which are explained in their individual entries in the manual below.

1.1.3 1/Octave

The 1 per octave signal is a linearized pitch signal centered at 0 = A4 = 440Hz. To go an octave up, just add one: 1 = A5 = 880Hz. To go an octave down, subtract one: -1 = A3 = 220Hz. To go up or down in semitones, add or subtract in steps of 1/12th: 1/12 = A♯4 $\approx$ 466Hz; -1/12 = A♭4 $\approx$ 415Hz.

Hardware modular synthesizers typically use 1 volt per octave tracking where 0 volts is the lowest note, often C1. Audulus's 1/Octave signal is instead centered at the reference pitch, which is defaulted to A = 440Hz.

The Keyboard node only outputs Hz, so make sure you use the MIDI Input module if you wish to use Audulus modules with a keyboard or pipe in a MIDI sequence from a DAW.

All non-gate sequencers in the Audulus module library generate a modulation signal only. This makes it easy to use these sequencers to modulate parameters as well as pitch. Their modulation output can be translated

into a 1/Octave signal using a Modulation to 1/Oct utility module or with a quantizer module which all have this translation module built-in.

Translating a sequencer's output is simple. The Range parameter multiplies the 0 to 1 modulation output of the sequencer by 0 to 8. If the range is set to 2 then each knob of the sequencer covers 2 octaves. If the range is set to 0.5 then each knob covers 1/2 of an octave. The Shift parameter sets you lowest note. At 0, your lowest note will be A4. At -1, your lowest note will be A3. At 1, your lowest note will be A5.

1/Octave inputs may be marked 1/Oct or 1/O on thinner modules.

The default reference pitch is set to A = 440Hz, but you can change this in any VCO if you wish. Enter any VCO and look for the 1/Oct to Hz converter (it may be a layer or two down in the module). Look for the RefHz variable and alter it to whatever pitch you wish. This change will only affect that particular VCO, so if you want to globally change the reference pitch you will have to make sure you do it in every VCO you use.

1.1.4 Audio

The Audio signal carries what you hear to your headphones or speakers. Its signal range is -1 to 1.

Audio signals are generated by VCOs (voltage-controlled oscillators) or an external audio input like a guitar or an audio track. They are modified by modules like VCFs (voltage-controlled filters), VCAs (voltage-controlled amplifiers), and effects like delay, reverb, and distortion.

Audio signals can be used as modulation at FM (frequency modulation) inputs of VCOs or as AM (amplitude modulation) at modulation inputs of VCAs. You can also modulate knobs at audio rates, but the knob will clip the negative portion of the audio signal. This means when the audio signal goes below 0, the knob will stay at 0.

It is not recommended to plug an audio signal into a modulation input. Doing so may cause unpredictable and undesireable behavior from some modules. The only exception to this rule is the VCA modulation input, as mentioned above.

Audio inputs and outputs are unmarked by any lights. They may be labeled Audio, Aud, In/Out, Left/Right or L/R for stereo modules, or have context-specific labels like Sine or Saw on a VCO. Inputs labeled FM always expect an audio signal.

When audio signals are mixed together, their total output may exceed a -1 to 1 range. You can even boost your audio signals creatively to drive a distortion module harder. The important thing to remember is that audio

exiting Audulus to your speakers, headphones, or audio interface must be kept between -1 and 1 to prevent clipping distortion.

1.2 Categories of Modules

Broadly defined, there are 6 different categories of modules. If you understand what category a module is, you can start to understand how they are typically wired together. Of course once you understand conventional modular signal flow, you can subvert it and do things like stick a clock into a reverb and then into a modulation input of a VCA, but you'll be doing it because you're experimenting - not because you're clueless.

Modules in the library are organized into more specific folders than these seven categories. This makes it easier to find the ones you want more quickly. Again, it might be overwhelming at first, but as you get accustomed to building, you'll find navigating the menus becomes second nature.

There are sometimes two or more versions of a particular module. The module that has a u- or μ-prefix such as μLFO, μClock, and μVCO are "micro" modules. These are small, CPU-efficient versions of larger modules that have just the bare essential functions. If you are completely new to modular synthesis, focus on using these modules first so you are not overwhelmed by the number of inputs, buttons, and controls of the other modules.

1.2.1 Tempo Modules

Tempo modules set the speed of your patch. There is really only one type of tempo module: the clock module. The clock module outputs gates.

A clock provides a steady pulse that can tick forward a sequencer and be used to open and close an envelope. The Master Clock module has many different subdivisions to play around with, but if you're new to modular synthesis, you might want to first stick with the μClock module.

Unless you are making experimental music that is not tied to a particular unified tempo, you should only need one clock module per patch. This makes it easy to adjust the overall tempo of your entire patch with one knob instead of having to turn several knobs at once.

1.2.2 Rhythm Modules

Rhythm modules take a clock signal and turn it into something more than a steady beat. Rhythm modules have gate inputs and gate outputs. The most recognizeable to you might be the 16 Step Gate Sequencer. If you plug

in a 1/16th note clock, each button pressed will represent a gate you can send to a kick, snare, clap, or even a synth instrument.

More exotic modules like the Euclidean Gate Sequencer take a clock signal and, based on an internal algorithm with several knob controls, can produce a regular, repeatable rhythm. You can even use two clock modules tuned to irregular intervals plugged into a μLogic module to create rhythms.

You can also create interesting movement in a patch by rhythmically modulating parameters like filter cutoff.

1.2.3 Modulation Modules

Modulation modules add expressiveness and movement to your patch. Envelopes, LFOs, sequencers, and sample & hold modules are all sources of modulation. Modulation modules will sometimes have modulation or gate inputs with modulation outputs.

The most basic and necessary modulation module is an envelope module. Without an envelope controlling a VCA, all of your oscillators would be constantly running. Envelopes can define the beginning and end of your sound.

Low-frequency oscillators or LFOs are another key element of modular synthesis. LFOs can change parameters of your patch over short and long periods of time, acting like extra hands turning knobs on your synth. A classic application is using an LFO in conjunction with an envelope to vary the cutoff of a filter. The envelope sweeps in a small range while the LFO slowly offsets the envelope higher and then lower in the cutoff range.

Modulation modules don't always need to be used on an audio-generating or modifying module. You can use an LFO to change clock speed, or a sample & hold module to add randomness to particular sequencer steps. Any knob you see on any module in Audulus is something you can modulate with any modulation module.

Modulation modules also include utilities like the Attenuate-Offset module. This module doesn't do much by itself, but when used in conjunction with an LFO, can help you adjust the range and limit of your modulation signal. Another modulation utility is a modulation mixer, which allows you to add together several modulation signals.

1.2.4 Pitch Modules

Pitch modules in Audulus are usually a combination of a module that creates modulation, like a sequencer, and a quantizer module, which outputs a 1/oct

signal. Without quantizer modules, you would have to manually tune each note of every step. This is how some old-school sequencers worked, and it could be a real drag when you just wanted to get a sequence running quickly.

Quantizers snap the smooth modulation transition from 0 to 1 into musical notes. A chromatic quantizer will equally divide the space between 0 and 1 into steps of 1/12 for each semi-tone. Other quantizers, like the Quick Quantizer, will snap to a user-defined scale.

Typically, you would use a clock to drive a sequencer, then send the output of the sequencer into a quantizer and then on into your VCO. However, you can use any modulation source as input to a quantizer like a sample & hold module, an LFO, or even an envelope.

Another example of a pitch module is a slew limiter. This module will add glide or portamento between your note transitions. Adding a just a little slew can make a sequence feel more natural - adding a lot of slew can really accenuate the transitions between notes. You can even use a sequencer to adjust the amount of slew to affect momentary note ties.

1.2.5 Audio-Generating Modules

Audio-generating modules create sound. VCOs are audio-generating modules. These modules typically have 1/Oct, Modulation, and sometimes Gate inputs, and have one or more audio outputs.

There are several different techniques of audio generation on display in the Audulus module library. Some output basic waveforms like sine, triangle, square and saw. Others, like the Chebyshev Polynomial VCO, allow you to build your sound wave with various parameters, and still others like the Karplus-Strong VCO build a model of a real string using noise, filters, and feedback loops.

1.2.6 Effect Modules

Effect modules transform audio. Delays add ghostly repeating echos to a sound, reverb adds a sense of space, distortion mangles a VCO's waveshape to add harmonics - to name just a few.

VCFs and VCAs are also essentially effect modules. They sculpt the sound of the raw oscillator into something more interesting and less static.

1.2.7 Audio Mixing and Output Modules

Mixing and output modules are usually the final modules in your patch. Mixers combine audio and send the audio to an output module.

It is best to have only one audio output module per patch, even if you have several mixer modules. Just like you want to have one master clock so you can control the overall tempo of your patch, you want to have one audio output module so you can control the overall loudness of your patch.

1.3 Wiring Modules Together

To make any sound with Audulus you will need to know how to wire modules together. This might be intimidating at first, but it is really easier than it looks.

In the following section we'll go through several types of basic patches you can make. For each example, we'll make a simple version and a more advanced version, explaining each module and how it is wired as we go.

After reading this section and building as you go, you will be able to use all of the modules in the Audulus module library.

1.3.1 Sequencer-Driven Subtractive Synthesizer

Subtractive synthesis is a method of creating sounds where you begin with a harmonic-rich wave like a square or saw and subtract frequencies using a filter. The typical signal flow of a subtractive synthesizer is VCO → VCF → VCA.

A step sequencer is a module that helps you play a synthesizer automatically. They usually have a set number of steps. For example, an 8 step sequencer can play back a sequence of 8 notes. Each step has a knob that can set the pitch of the note you want it to play.

Simple Sequencer-Driven Patch

Any sequencer-driven patch must begin with a clock. We'll use the μClock module because it can both drive a sequencer and open and close an envelope.

Next we'll attach the gate output of the μClock to the Gate input of the 8 Step Sequencer. Once connected, you should immediately see lights moving on the sequencer. The moving blue light indicates the current step of the sequencer. The green light represents the first step of the sequence, and the red light represents the last step of the sequencer. Once the sequence has reached the last step, it starts back again at the first step.

Since the sequencer only outputs a modulation signal, we'll need a quantizer to translate that modulation signal into an 1/octave signal and snap

them to a scale. We'll use the Chromatic Quantizer module and wire up the modulation output of the sequencer to the modulation input of the quantizer.

Now that we're generating a pitch signal, we'll attach a μVCO module to the quantizer. You won't be able to hear anything just yet because we haven't added an audio output module. Before we do that, we'll want to create a few more modules to help shape the sound.

The next module we'll add is an envelope. An envelope is a module that shapes the contours of each note. In this patch we'll use one envelope to control both the cutoff of the VCF and the loudness of the VCA. Attach the Gate output of the μClock to the Gate input of the μADSR module. You should see the output of the envelope flashing red in time with the incoming gate signal.

Once you have wired up the envelope, create a μVCF module. Attach the Square output of the μVCO module to the input of the μVCF, and then wire the Envelope output of the μADSR to the Modulation input of the μVCF.

Next we'll create a VCA, and wire it similarly to the VCF module. Add a μVCA module and attach the audio output of the μVCF to the input of the μVCA, and then wire the Envelope output of the μADSR to the Modulation input of the μVCA.

Now that we have all the elements of our sequencer-driven subtractive synthesizer wired together, we just need to create an audio output. Add an Audio Output module and wire the output of the μVCA to the left and right inputs of the Audio Output module.

The first thing to try is to change the notes of the sequencer. Turn each knob until you have a sequence going that you like, then try adjusting the knobs on the μADSR module, and notice how the sound changes. If you change the PWM value of the μClock module, you can get longer or shorter note durations which will interact with the settings of your μADSR. Another thing to try is adjusting the Hz (cutoff) and Env knobs of the μVCF. The Hz knob sets the base frequency of the filter and the Env knob adjusts how much envelope modulation is applied to the cutoff control. Explore all of the controls of every module in this patch and you'll become familiar with what they do.

Advanced Sequencer-Driven Patch

There are two elements missing from the basic sequencer-driven patch above: rhythm and modulation. We can build on the basic patch by adding some

modules and subbing out others.

First, let's exchange the μClock for the Master Clock module. The Master Clock offers many different subdivisions or clock speeds that are all rhythmically related. We'll use these different clock outputs to drive our sequencer and open and close our envelope to create rhythm.

The next module we need to add is the 8 Step Sequential Switch. A sequential switch is a special sequencer that doesn't switch between knobs but instead switches between inputs. Only one input at a time is sent to the sequential switch's output, and we use the knobs to set which input is selected. At the bottom of the module is an indicator that shows which input is selected at each step.

Now we can wire 8 different subdivisions into the sequential switch and choose a new subdivision for each step of the sequencer. You don't have to set the sequencer to switch between the inputs in order like this:

$$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8$$

You can easily do something more interesting like this:

$$5 \rightarrow 3 \rightarrow 8 \rightarrow 1 \rightarrow 1 \rightarrow 2 \rightarrow 5 \rightarrow 7$$

To make the sequential switch work correctly for this application, we have to wire the output of the sequential switch back to its own Gate input. We also have to make sure all of the inputs are connected to a different clock division. If one of the inputs is left open, the sequencer will stop if you select that input.

Once you have wired up your sequential switch, attach the output of the sequential switch to the Gate input of the 8 Step Sequencer and the Gate input of the μADSR module. You should immediately hear that your sequence now has a rhythmic element.

You might notice that the sequencers seem out of sync with one another. The sequential switch might be on step 1 when the sequencer is on step 5. Since you have two different modules controlling two different parameters of individual notes, it will be easier to program them if step 1 of the sequential switch equals step 1 of the sequencer.

You can sync these modules in two different ways. The first and sometimes fastest is to simply close and reopen the patch. When you reopen the patch, everything will be reset and automatically be synchronized. Another option is to create a Trigger node and attach its output the Reset inputs of each module and press the button once.

The next thing we can add to create a more interesting patch is modulation. We'll use two different types of modulation modules in a few places in our patch.

The first module we'll use is a μLFO. It has two outputs: Sine and Triangle. Attach the Triangle output to the Shape input of the μVCO. Modulation at the Shape input will change the pulse width of the square wave. The pulse width is the ratio of high-to-low time. If you want to really see how it's affecting the wave, copy and paste another μVCO to the side, attach a value of -9 to the 1/Octave input, and connect the Square output to the input of a Waveform node.

If you set the LFO speed low, you can get a nice, slow, evolving sound that breaks up the monotony of the patch. If you set the LFO speed high, you can get almost a vibrato effect from it. You can also experiment with the Atten (attenuate) and Offst (offset) controls to get the modulation working in the range you want it to. The LFO will move only as much as you set the attenuate control, and will bottom out at the point you set with the offset control. If the value of the attenuate and offset controls are greater than 1 (set attenuate all the wave up and offset half way up) the wave will be clipped at a maximum of 1 so that only the bottom portion of the wave will come through.

The second module we'll add is a Basic Sample & Hold. Sample & Hold modules output random stepped modulation signals. If we apply this modulation signal to the Hz knob of our μVCF, we can get a classic beep-boop robot sounding filter. This works best with quick modulation, so turn up the Hz control.

The Basic Sample & Hold module also has some extra controls that are worth trying out. The first one is the Distr (Distribution) knob. When the Distribution control is set to 0.5, there is an equal chance for any random number from 0 to 1 to be chosen. If you set the Distribution knob to 0, this biases the random numbers to be closer to 0. If you set the Distribution knob at 1, the numbers will be closer to 1. Generally on a filter, the modulation will sound better with a Distribution value of 0 to 0.5. This is because you still have an envelope input, and if the combined amount of the Sample & Hold modulation and the incoming envelope exceeds 1, it just holds the filter wide open and you don't hear any change in the sound.

The other cool thing we can do is sync the Sample & Hold to our Master Clock. Press the Ext (External) button next to the Gate input and attach any subdivision you want from the Master Clock module. The random modulations will now be in time with your patch.

Finally, try experimenting with the Slew knob which smooths the tran-

sitions between the random values.

As a bonus, we'll also replace the Chromatic Quantizer with the Quick Quantizer module. The Quick Quantizer allows you to dial in a root note and scale with just two knobs. This makes creating melodies much easier. We can then also attach a second clock-synced Sample & Hold module to modulate the steps of the sequencer. Connect the output of the 8 Step Sequential Switch module to the external Gate input of the Sample & Hold module and attach its output to any or all of the 8 Step Sequencer's knobs.

Also try creating another μLFO and attaching a Triangle wave to the Offset control on the Quick Quantizer module. Set the speed to slow and use the attenuate and offset controls to get it modulating around the middle of the control. Now not only will all of some of your sequence sound random, but it will also be shifting up and down in octaves smoothly.

1.3.2 Keyboard-Controlled FM Synthesizer

1.3.3 Drum Patterns and Mixing

1.3.4 Stereo Effects

1.3.5 Generative Sequencing

1.3.6 Automation

1.3.7 DAW Integration

1.3.8 Eurorack Integration

1.4 Audulus Module Library Manual Organization

The modules presented in this manual are presented in the order they appear in the module menu. Each chapter represents a category of modules.

In the in-app menu, modules may have reversed names like Clock Master and Gate Bernoulli when they are refered to here as Master Clock and Bernoulli Gate. Organizing them this way in the menu makes them easier to quickly find, especially in the right-click menu on Mac. Doing this also cuts down on the need for sub-folders, which are difficult to navigate past 2 levels.

Modules in this manual are presented first with a screenshot and a table that outlines what each input, output, knob, and button does. Then, after a brief description of the module, a few example patches are presented.

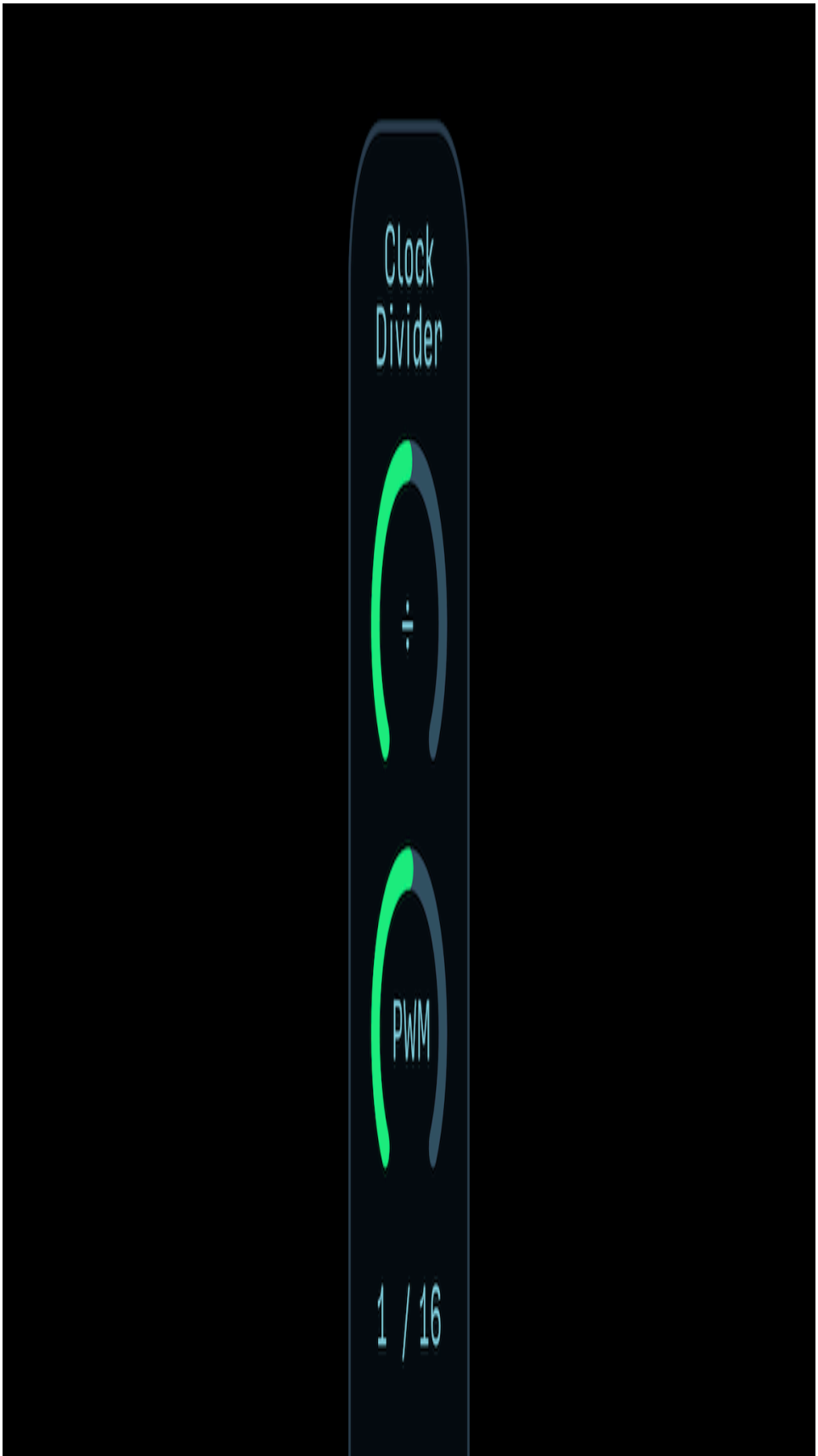
Chapter 2

Clock Modules

Clock modules output gate signals. They keep time, trigger events, and open and close envelopes. In hardware modular synths, clocks typically output a very short gate signal called a pulse. In Audulus, however, clock signals can be long or short gates.

This category of modules includes clocks as well as clock modifiers. Clock modifiers take an incoming clock signal and change it in some way. For example, the Clock Divider will take a fast incoming clock signal and make it slower, while the Bernoulli Gate module will take an incoming clock signal and send it to one destination or another based on chance.

2.1 Clock Divider



Knob	
÷	*Sets clock speed divide-by factor from 1/1 to 1/64.*
PWM	*Sets the pulse width of the divided clock at the PWM output. Does not effect the Gate output.*
Input	
Rset	*Gate this input to reset the divide-by counter. Useful when attempting to sync multiple Clock Divider modules.*
Gate	*Connect the clock signal you want to divide to this input.*
Output	
Gate	*The divided gate signal output. This gate output will preserve the pulse width of the incoming gate signal.*
PWM	*The divided gate signal output. This gate output will have a pulse width set by the PWM knob.*

The Clock Divider module takes an incoming clock signal and slows it down. It does this by only letting 1 out of every x number of pulses through to its output. For example, if an incoming clock comes once per second and the Clock Divider divides the speed by 2, then the outgoing clock will come once every two seconds.

2.2 Master Clock

Knob	
BPM	*Sets the beats per minute (BPM) of the clock from 60 to 220 in steps of 1BPM.*
PWM	*Sets the pulse width of all outputs simultaneously.*
Input	
Rset	*Gate this input to reset the divide-by counter. Useful when attempting to sync multiple Clock Divider modules.*
Gate	*Connect the clock signal you want to divide to this input.*
Output	
Gate	*The divided gate signal output. This gate output will preserve the pulse width of the incoming gate signal.*
PWM	*The divided gate signal output. This gate output will have a pulse width set by the PWM knob.*

The Clock Divider module takes an incoming clock signal and slows it down. It does this by only letting 1 out of every x number of pulses through to its output. For example, if an incoming clock comes once per second and

the Clock Divider divides the speed by 2, then the outgoing clock will come once every two seconds.

2.3 Bernoulli Gate

2.4 Chance Select Gate

2.5 Pachinko Machine

2.6 Random Bursts

2.7 uClock

Chapter 3

Drum Modules

3.1 Percussion

3.2 μHat

3.3 μKick

3.4 μSnare

Chapter 4

Effect Modules

4.1 Delay

4.1.1 Stereo Delay

4.1.2 Looper Sync

4.1.3 Tempo Sync

4.1.4 uDelay

4.2 Distortion

4.2.1 Asymmetrical Drive

4.2.2 Fold Processor

4.3 Reverb

4.3.1 Really Humungous Reverb

4.3.2 uVerb

Chapter 5

Envelope Modules

5.1 Analog Envelope

5.2 EOC Max ADSR

5.3 μADSR

Chapter 6

Input-Output Modules

6.1 Audio Input

6.2 Audio Output

6.3 Expert Sleepers ES-3

6.4 Expert Sleepers ES-6

6.5 Expert Sleepers ES-8

6.6 MIDI Input

6.7 VPO Converter

Chapter 7

LFO Modules

7.1 Basic LFO

7.2 Octature Sine LFO

7.3 TZFM LFO

7.4 μLFO

Chapter 8

Mixer Modules

8.1 8 Channel Mixer

8.2 CV Mixer

8.3 μMix Audio

8.4 μMix CV

Chapter 9

Quantizer Modules

9.1 Chromatic Quantizer

9.2 Church Modes Quantizer

9.3 Gateable Quantizer

9.4 Modulation Quantizer

9.5 Neo-Reimannian Triad Transformer

9.6 Quick Quantizer

Chapter 10

Sequencer Modules

10.1 8 Step Sequencer

10.2 16 Step Gate Sequencer

10.3 Binary Gate Sequencer

10.4 Euclidean Gate Sequencer

10.5 Matrix Gate Sequencer

10.6 Shape Gate Sequencer

Chapter 11

Utility Modules

11.1 Attenuate-Offset

11.2 Attenuverter

11.3 Audio Attenuverter-Offset

11.4 Automation Lane

11.5 Chaos Generator

11.6 Modulation to 1/Oct

11.7 Poly to Mono

11.8 Sample & Hold

11.9 Slew

11.10 Switch Sequential 8 Step

11.11 VC Switch

11.12 μLogic

11.13 Unison

11.14 μSample & Hold

11.15 Window Comparator

Chapter 12

VCA Modules

12.1 Digital VCA

12.2 Diode VCA

12.3 μVCA

Chapter 13

VCF Modules

13.1 303 VCF

13.2 Basic EQ

13.3 K35 VCF

13.4 Ladder VCF

13.5 LPG

13.6 SEM VCF

13.7 μVCF

Chapter 14

VCO Modules

14.1 Basic VCO

14.2 Chebyshev Additive VCO

14.3 Karplus-Strong VCO

14.4 Morphing VCO

14.5 Noise

14.6 Skew VCO

14.7 Supersaw VCO

14.8 TZFM VCO

14.9 μNoise

14.10 μVCO