Chapter 1.3.3:
A Short Glimpse at the Turing Machine
And at Shift Registers Again

We already met the Turing Machine as well as shift registers in earlier parts of this book, and we will “meet” the Turing Machine in Chapter 5 in detail again, but here in the parts about sources of completely random processes both shall only be mentioned. The Turing machine generates completely random CV developments with its central main knob adjusted to the 12 o´clock position. Turning it down to the 5 o´clock position the generated sequences get more and more regular, loose more and more of their randomness, until we get a completely regular repeating pattern at the five o´clock position (and with that said I don´t have to mention the Turing Machine again when I start talking about sources with adjustable amounts of randomness a bit later in this chapter).

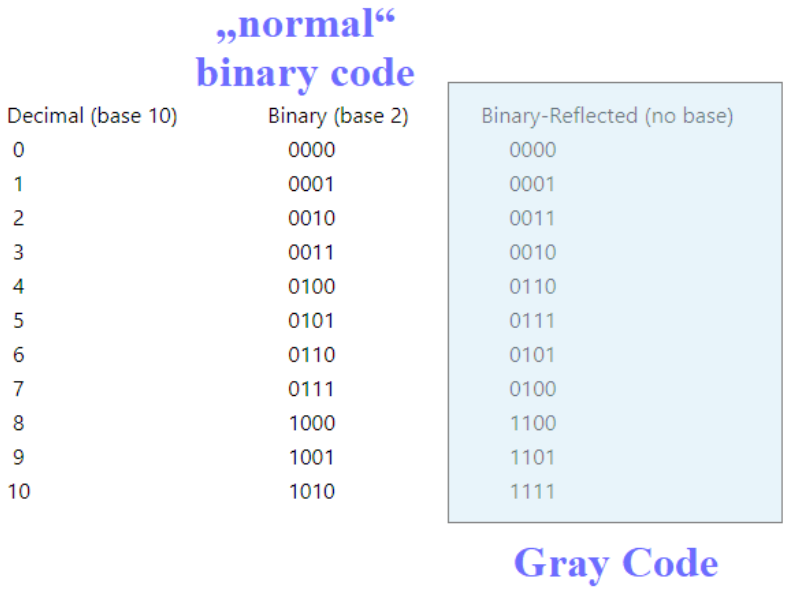
Shift registers with feedback (see chapter 1.2.3) are as random as the source they get their first register fed with, and as random as the gates, which start and stop feeding them (again: see chapter 1.2.3 for details). Therefore they can generate completely random sequences as well as completely regular repeating sequences as well as anything between.

Chapter 1.3.4:
Perfect Pseudo Randomness:
Gray Code Modules

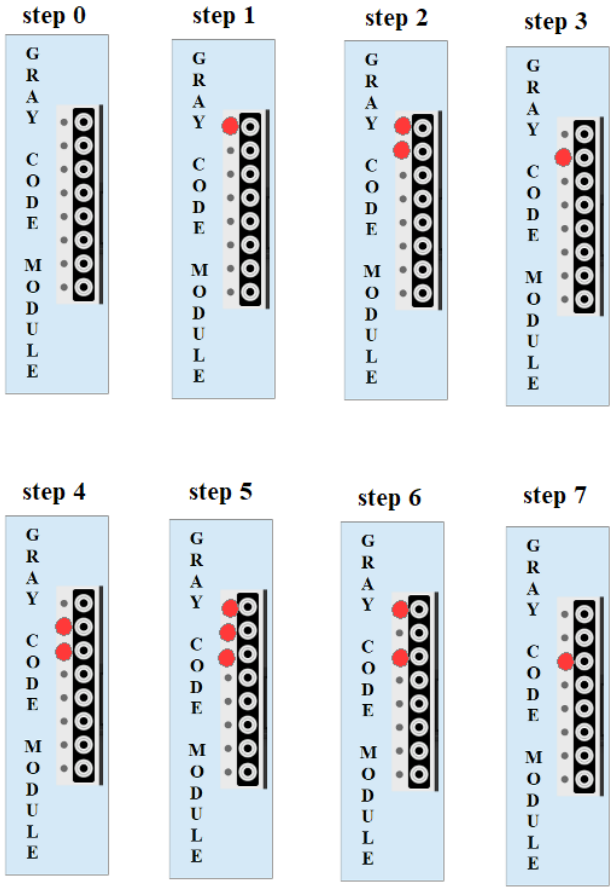
These Modules are named after the physicist Frank Gray, who invented a binary system, in which each two consecutive numbers differ only in one single bit. You will stand a tiny, tiny bit of mathematics here:

The binary correspondent to the decimal value of “1” is “0001” (in a 4-bit presentation), and the binary correspondent of “2” is “010”. So, two bits have to be changed. But in Gray Code the decimal number of “2” corresponds to “011” - only one bit has to be changed.

At first no output is high (high voltage level, gate open, trigger send etc.) representing the number “0”.
The next step makes output 1 high (= “1”). The next step (= “2”) makes output 2 high leaving output 1 high as well.
The next step (=”3”) makes output 1 low, but leaves output 2 high. The next step (=”4”) makes output 3 high leaving output 2 high and output 1 low. And so on according to the Gray Code binary system. The pulses or gates coming out of the 8 outputs seem perfectly random, but indeed they strictly follow the Gray Code formula.



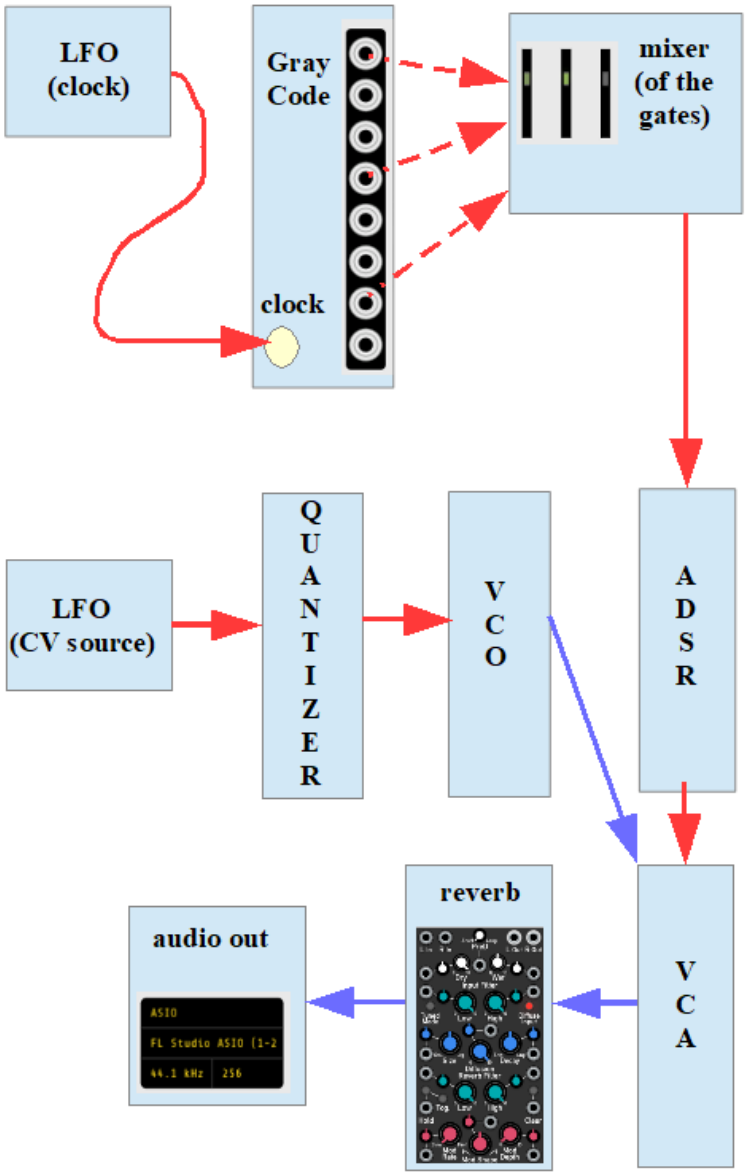
We can use each of these outputs (or some of them at the same time, or even all together) to trigger events, open ADSRs etc. They cannot be used to generate sequences of different pitches directly, because there are only two values (low and high), but the video behind the following link shows how to use them to generate random pitch sequences nevertheless.



Please look at the last picture (steps 0- 7). The first (upper) output switches 2 times to a high during these 7 steps. Output 2 switches only once from low to high, but stays high for 4 steps etc.

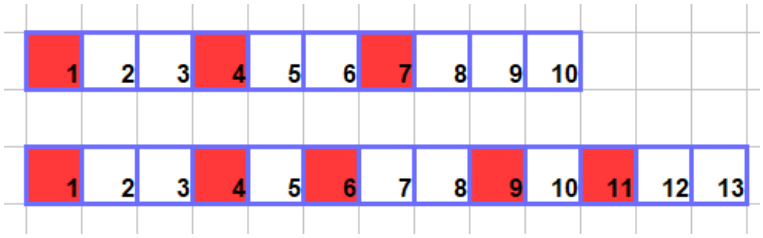
The block diagram of the patch looks as follows:
the upper three modules generate gate signals in seemingly random order. Always when one of the outputs (of the Gray Code module), that is patched into the upper mixer is at “high”, a gate signal is sent to the ADSR. This gate lasts as long as the corresponding output of the Gray Code module is at “high”. In the block diagram there are 3 of the outputs patched into the upper mixer, so that there are 3 different gate levels, which occur at the mixer´s output, but each of these levels will open the ADSR – the three different **levels** are not of any importance therefore. As long as any gate level (except for zero-level - “no gate” of course) comes out of the mixer, parts of the sequence of pitches, which are generated by the lower LFO-Quantizer combination are audible. The preset “graycode.vcv” (presets are only availyble in the book. See https://dev.rofilm-media.net) and the video behind the following link may make things even clearer to you, and shall serve as a “base camp” for

experiments of your own.
https://youtu.be/U3htiOV_oBs



Chapter 1.3.5:
Imperfect Pseudo Randomness:
Euclidean Sequencers

The idea is simple: it´s all about spreading a certain number of events (e.g. triggers or gates) as evenly as possible across a given area (e.g. the length of a sequence).
Let´s take a sequence of 10 steps length. The most even way to spread 3 triggers is at step 1, at step 4 and at step 7.
Or spreading 5 triggers across a sequence of 13 steps following this principle would lead to triggers at steps 1, 4, 6, 9 and 11.



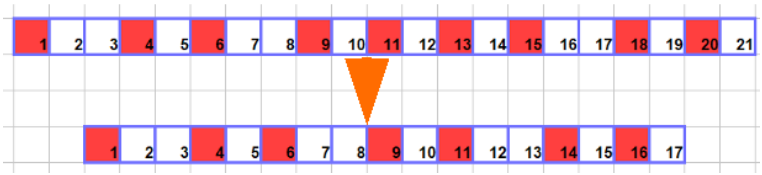
The mathematical algorithm to calculate these positions goes back to the Greek mathematician Euclid. Therefore the name. There is absolutely nothing random about it – but there wasn´t any randomness in Gray Code modules either, and – nevertheless - they "sounded" VERY random, didn´t they?

Well with Euclidean sequencers it´s a bit different. Their output doesn´t even sound random – irregular and a bit eccentric, but not random. When we use Euclidean sequencers for creating rhythms with percussion instruments we are in the world of native African rhythms quite soon. So, why are these modules of any importance for the matter of this book?

The answer reads: because we can modulate the length of the sequence (= the area, where we have to spread out our events) as well as the number of events to spread across this sequence. And here it starts to sound random – not that perfectly as with Gray Code modules, but random enough for our purpose.

When I wrote "number of events" I rather should have called it "percentage of steps, which initiate an event", because increasing the length of a sequence with an Euclidean sequencer increases the number of events accordingly. And reducing the length of the sequence reduces the number of events accordingly too. Otherwise modulating the length of the sequence AND the number of events would lead to impossible situations (e.g. more events than available steps etc.)

Let´s say I have a sequence of 21 steps and place 9 events evenly (they will be at steps 1, 4, 6, 9, 11, 13, 15, 18 and 20). Then I reduce the length of the sequence to let´s say 17. The sequencer automatically reduces the number of events from 9 to 7 then (at steps 1, 4, 6, 9, 11, 14 and 16).



Things get quite interesting with a larger number of events (e.g. 7 events in a sequence of 11) and a slow clock. But watch the video behind the following link, and use the preset “euclid.vcv” (presets are only available in the book. See https://dev.rofilm-media.net) to experiment.
https://youtu.be/Q-ibEGvAQsc

… to be continued