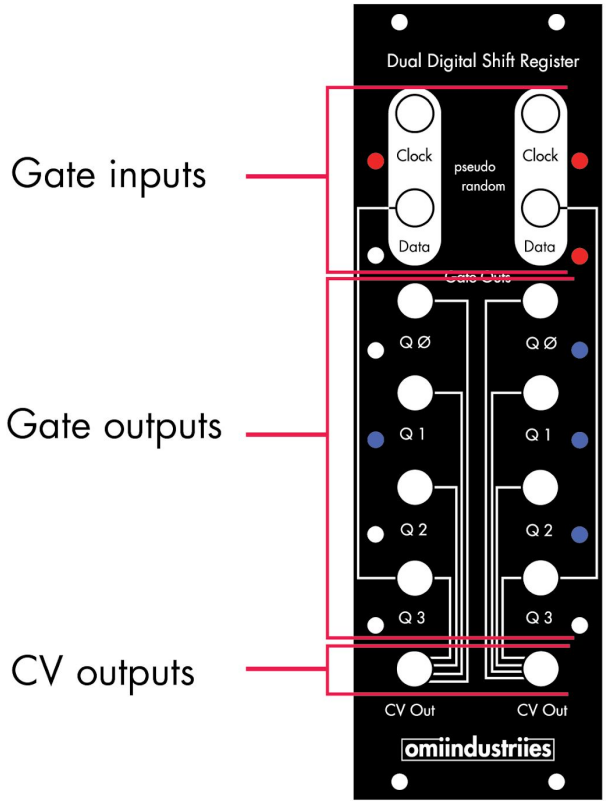
omiindustriies Dual Digital Shift Register

Dual Digital Shift Register



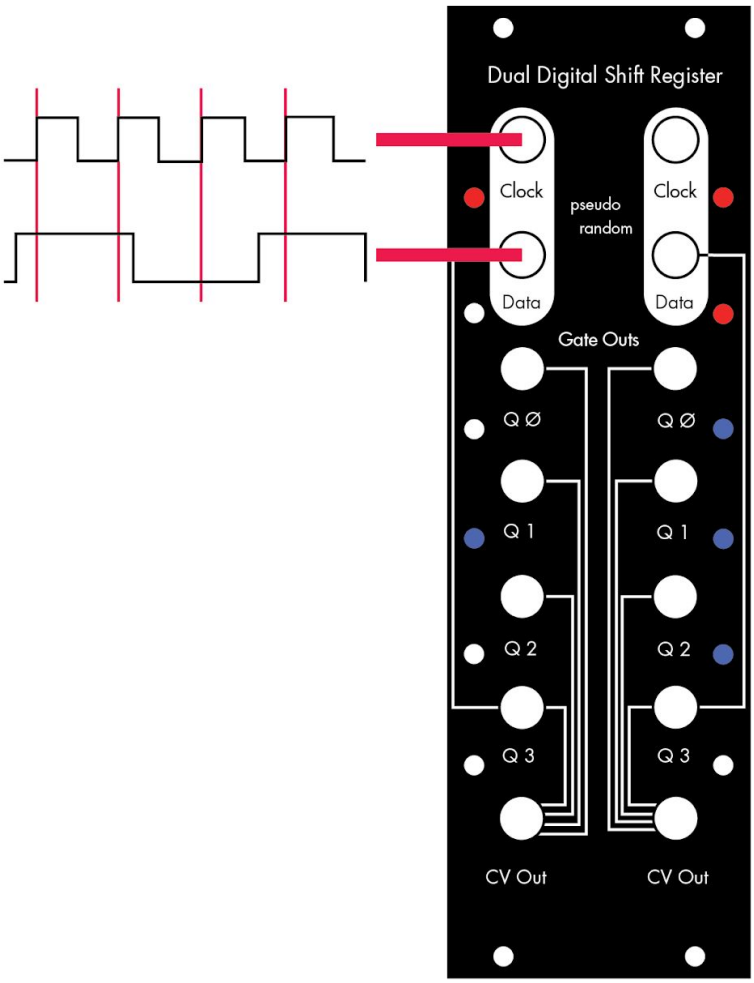
- The Dual Digital Shift Register (DDSR) is a shift register based pseudo-random cv and gate generator. It uses gate signals to create chance operations, generating aleatoric sequences.

- It has 2 channels, each with 2 inputs and 5 outputs. The inputs are clock and data.

- The outputs are 4 gate outputs, labeled QØ, Q1, Q2, & Q3 and a cv output.

- A shift register is used to sequentially pass information from a data input through a certain number of stages. When the data reaches the end of the stages it is shifted out of the register. It has a clock input, that tells it when to look for new data and to move the data through the register. An analog shift register is basically a series of sample and holds. The voltage level present at the data input is the data that is passed from one stage to the next. In a digital shift register, the data is binary, meaning it only has two stages, on or off, in modular synthesis terms, this would be a gate signal. Two famous examples of modular synthesizers that use shift register are the Turing Machine from Music Thing Modular and the Rungler circuit from Rob Hordijk's Benjolin.

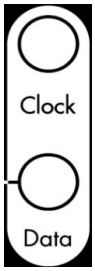
- It can be predictable if given closely related signals, which can repeat or subtly morph between different rhythmic expressions. The DDSR settles into patterns and then moves away from them, there are periods when it will grab sequences and loop them. The sequences it creates are based around loops instead of random uncorrelated voltages and because of that the results are musical and familiar.



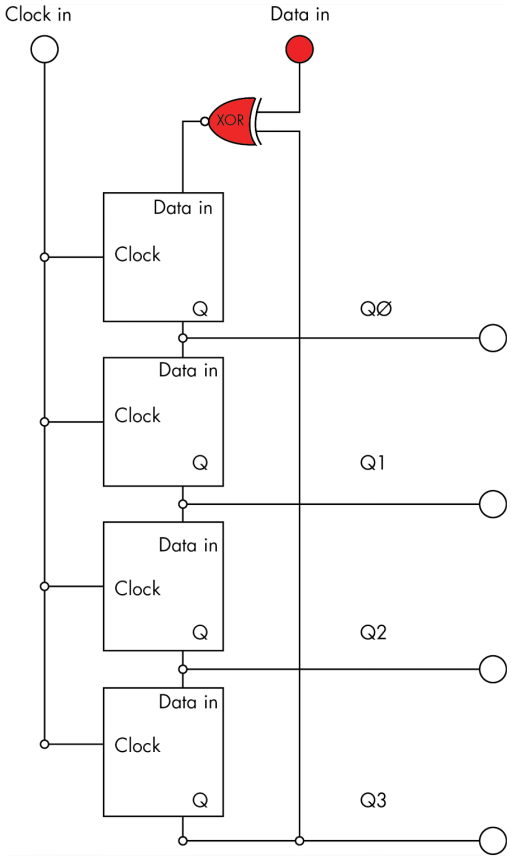
Your First Patch

➔ To begin, connect a signal to the clock and data inputs on whichever side you want to use. Depending on the state on start up; some, none, or all of the LEDs may be illuminated. The pulse width of the clock signal does not matter as it only pays attention to the rising portion of the clock signal, but the time at which the data input is high does matter. Once you have a sequence going, it will repeat until new data is added to it. The red lines indicate the rising portion of the clock pulse, when new data is added to the shift register.

How It Works



➔ The DDSR has two inputs that are labeled clock and data. These expect gate signals, which can be thought of as binary signals. A binary signal has two states on and off, which is often written as 0 or 1. I will be referring to those two states as high or low. The clock and data inputs will register as high when a signal over .5v is received, indicated by a illuminated red LED.



➔ When the DDSR gets a clock signal, the shift register looks at the data input. If the data is high, then it moves a high signal into the first stage of the shift register, QØ. If the data is low, then it moves a low signal into the QØ. On the next clock pulse, the shift register looks at the data input and moves that state into Q0. The signal in Q0 is shifted to the next stage of the shift register, Q1, Q1 is shifted to Q2, and Q2 to Q3 When a bit reaches the final stage, Q3, it is fed back into the data input and XOR'd with the external data input. When two signals are XOR'd it means that the output will be high if one of the inputs is high, and will be low if neither of them is high or they both are. With no input at the data input, the 4 step pattern is recycled.

➔ If all the stages are low, all the outputs will remain at 0 until a new signal is present at the external data input. If all stages are high, all the outputs will be at their maximum value. *If either of these two conditions happen, there will be no changes in any modulation. Add new data in order to start the sequence again.*

➔ Unsynced signals can cause this more easily than synced signals. Additionally, gate signals are suggested for data inputs over triggers as there is more chance for the clock and data to overlap as the clock looks for new data on the rising edge of the clock signal. Pulse width is not taken into account for the clock.

Aleatoric and Pseudo-Random

- ➔ Aleatory comes from the Latin word “alea” meaning dice. It describes phenomena where some element is left to chance. These chance operations often have a limited number of outcomes. The sequences are pseudo-random.

- ➔ Pseudo random in this case differentiates itself from “true” random by being deterministic, meaning the output values are based on the relationship of the inputs and being periodic, meaning it has the tendency to repeat. Given the same initial state, which is referred to as the seed, and the same clock and data inputs, the sequences can be recreated. Additionally there are only a certain number of possible output values. In a 4 bit shift register the possible number of outputs, including 0, is 16. By only having a limited number of states, discernible patterns emerge. These patterns often loop for periods of time and then settle into new patterns which will repeat for a different length of time. With the input of new data, the pattern may change before it has a chance to become noticeable as a looping pattern. Using inputs that are closely related to each other causes the patterns to be more noticeable such as even divisions from a master clock.

Gate outputs

- ➔ Each of the stages outputs a +5v gate signal, indicated by a blue LED. The signals are continuous, so the gate output will remain high until a low signal is shifted into it. The 4 stages act as 4 pseudo-random gate delays, whose outputs are shifted one clock pulse from each other and whose pulse width is also pseudo-random, but influenced by the gate being delayed. Gates can be used as timing events, such as triggering envelopes or advancing a sequencers. They can also be used as modulation sources as phase shifted pulse wave oscillator whose frequency can go as low as you can clock it (including stopped) up to well above human hearing.

- ➔ The gate outputs from one channel can be patched into either input on the other. This allows for fewer signals to be dedicated to the DDSR. Although you only need 1 external clock signal to make the shift register work, but it’s best to use at least two. If you don’t have a way to generate multiple clocks, you can patch the same clock into both channels and patch a gate signal from each side to the data input on the other. Additionally you can patch signals of different rates into the two different clock inputs for more complex and less directly related output patterns.

CV

- ➔ The Dual Digital Shift Register uses a digital to analog converter to turn the 4 digital binary signals into an analog voltage. The voltages are added together to determine the output voltage. The stages are weighted differently, meaning the stages affect the output voltage to different degrees. The state of QØ, which is known as the most significant bit, causes the most change, while Q3. the least significant bit, causes the least change. The output range is different for each channel. Channel A has an approximate range of 0–+3.1V and channel B has an approximate range of 0–+4.3v, both with 16 different values spread out through that range.

Sensitive to Slew

- ➔ The Dual Digital Shift Register is sensitive. As we've established, the inputs on the Dual Digital Shift Register expect gate signals, but what happens when they receive signals that don't have a hard edge? Well, the chip doesn't like clock signals with slew or noise on them. It freaks out the circuit. Instead of moving the data through the register linearly, it skips around according to the shape of the input signal. Use this to your advantage. One thing you can do is run a gate signal through a voltage controllable slew generator and when you modulate the rise and fall times, errors occur in the sequence as the signal becomes more slewed. It will track normally when the signal isn't slewed. Try different inputs signals with different shapes to see the results.

Tips

- ➔ If the DDSR is not receiving new data, you may have to adjust the rate or pulse width of the data signal. Narrow pulses and/or pulses that are faster than the clock signal may not overlap properly. If both clock and data go high at exactly the same time the shift register won't recognize the data as being high. The circuit only looks at the data input at the moment the clock pulse goes from low to high. If both happen at the same time then the data signal isn't high, it's in the process of going high. However, eurorack exists in the real world and not software, so there may be a slight drift in timing signals. This may lead to the data signal occasionally overlapping with the rising edge of the clock signal.

- ➔ The DDSR works as a noise source when clocked at audio rate. Changing the rate of either input signal will change the both the pitch of the noise and the harmonic content of it. Using slewed signals will output a different kind of noise than noise derived from gates.