## tales from the core


a transmediale '05 workshop
by Niklas Roy \& Uwe Schüler
Part II :

Uwe Schüler featuring
"Zuse on acid"
building multimedia hardware in relay technique

Our 1st D.I.Y.-relay-machine will be a drum-machine or more detailed: a 4 track/8 note -hardware sequencer made of 1 clock generator board and 4 drum-loop-boards featuring real-time note input. The 4 outputs trigger electromagnets which hammer against something making noise.
Before we start some hints:

- read Niklas'Part I handout first ; you must understand how a relay works
- we will use 48 V for relay power supply. This is still called "low voltage", and safe to touch with your fingers - but keep your tongue away... and switch off power before soldering - it's also safer for the circuit ;-)
- the relays we use are compact modern types: they have 2 pins for the solenoid coil ( $\mathrm{Sa}, \mathrm{Sb}$ ) controlling 2 switches each with 2 positions. That's why this type of relay is called "double pole double throw" (dpdt)
relay bottom view
(solder pins)
relay schematics
(logical wiring)



NO : Normally Open
NC: Normally Closed
C : Common contact
"Normally" means : no power applied, OFF-state
to make schematic drawings more readable we will use 2 symbols to represent connections to the +48 V power supply and OV (ground ,minus)

$$
\begin{aligned}
& \Phi=48 \mathrm{~V} \\
& \Gamma=\varnothing \mathrm{V}, \text { Gid }
\end{aligned}
$$

to make your circuit more "readable", please do all wiring with the right colors : red, brown for +48 V , blue, black and blank wire for OV . white ( also with color rings), green, yellow.... for signals
beside the relays we use resistors and capacitors : resistors have a color code printed showing their value - don't mix them up! Our capacitors are electrolyte types; they must be used in the right direction, otherways they die or even might explode! Polarization (,+- ) is printed on the insulation sleeve. Ask if you are not sure!
Now let's see how some basic digital logic gates can be implemented with our relays. If you don't remember this AND, OR, NOT, NAND... stuff, please read about it again ( many tutorials on the web...)

## basic logic gates with relays:


the XOR is a bit tricky : the coil is only energized (and thus moving the switch output from OV to +48 V ) when there is a voltage BETWEEN $A$ and $B$. Polarity plays no role - $A, B$ could be exchanged. This means : if BOTH inputs have either OV or BOTH have +48 V , there is no voltage BETWEEN $A$ and $B$, so the relay will not switch and the output will be " 0 ", which is exactly the XOR function.

## basic memory functions with relays:

a simple 1 bit memory can be done with a single relay :

when the "Set" switch is pushed, the coil is connected to +48 V and the output $Q$ switches to +48 V . A resistor $R$ feeds this voltage back to the coil. When the "Set"switch is released R continues supplying the coil, so the relay keeps the ON state. When the "Res" button is pushed, the coil is shorted to OV, so the relay is reset to OFF-state. That means with a "1" pulse on the input, this memory cell ( or Flip-Flop) can be set, and with a "O" pulse on the input, it can be cleared. This is called a LOCK RELAY.


D - (Data) Flip-Flops store the state of the $D$-input when the clock input CLK makes a transition from " 0 " to " 1 ".
This circuit also uses the lock relay from above. Set/Reset pulses come from a 47uF capacitor, that is charged with 48 V or OV, depending on input D. A 2nd relay transfers the charge of the 47uf Cap. to the lock relay when CLK becomes "1". There are 2 outputs : Q (noninverted) and QNOT (inverted)

Such a D-FlipFlop is called "edge-triggered ", because data is stored only upon a "0->1" transition of the clock pulse. The reason why we use such a tricky FF instead of a simple lock relay is not obvious yet, but we'll soon see why we can' $\dagger$ live without it - let's connect a couple of D-FF like this :


> this is called a "serial shift register ": data at the serial input SI on the left side is shifted from one D-FF to the next D-FF in this chain, each time the clock pulse goes from "0" to "1"

Example: after power up all outputs $Q$ are " 0 ". Now we apply a " 1 " to SI. What happens to the Qs ? Right, nothing: they all stay "O" until the 1st clock pulse stores the "1" from SI, so $Q$ of the left D-FF becomes a "1". The D-FF in the middle got the clock pulse at exactly the same time, so it stored the " 0 " from left $Q$ that was present at the middle $D$ input. A few milliseconds later the left $Q$ goes to "1", but this is too late to be stored in the middle D-FF because the clock transition from "0" to "1" happened already. That's the reason why we need an "edge triggered" D-FF! If Data would have been stored during the whole duration of the clock pulse this would happen : the "1" from SI occurs on $Q$ of the left FF, which is D of the middle FF, so $Q$ of the middle FF would also go "1" and a short time later $Q$ of the right FF would also go "1". So you can see our 3 bit serial shift register needs edge-triggered D-FFs and after 3 clock pulses, data from SI appears at the serial output SO. OK, you might agree, but why should I build a serial shift register? Let's add another shift-stage ( 4 is a nice number, but for our drummachine we will need 8 or 16 of them) and connect the serial output back to the serial input with a litte circuit like this:

there are 2 push-buttons: SET and CLEAR. If none of the 2 is pushed, Data from SO flows through the 2 relays, as shown, back to SI. That means when clock pulses were applied, Data stored in the D-FFs will CIRCULATE in this so called "ring buffer" : after 4 clock beats the state of our shift-register will be the same, because all data leaving SO is fed back to SI. SET and CLEAR buttons allow us to input a "1" or a "0" in this ever circulating datastream: this is one track of a 4 note drummachine!
example: if we apply clock pulses and never touch the buttons, " 0 " $s$ will circulate in our ring buffer forever. Now we hit SET and after the next rising edge of the clock pulse a "1" enters the shiftregister. If we had a trigger-input of an electronic basedrum connected to $Q$ of the left D-FF, we would now hear a "BOOM". We immediately release the SET button and watch the next clock pulse: a "O" from SO is now connected to SI and will be stored in the left D-FF. The "1" that caused the drum-trigger before, is now shifted to the 2nd D-FF. So the state after the 1st clock pulse of our ring buffer was " 1000 "; after the 2nd clock it is "0 100 "; after the 3rd clock "O 010 " and after the 4th clock "0001". serial output SO is now a "1" and with the 5th clock pulse it is stored again in the leftmost D-FF and the next "BOOM" can be heard.
Voila, a simple "4 on the floor" base-line-loop is programmed!
We can SET now drum-triggers in "real-time" while our ring-buffer loops over and over. Of course we must also be able to delete a note when we don't want to hear it anymore : hitting CLEAR enters a " 0 " with the next clock pulse into the loop.
A final small modification to the output must be done to complete our design: if you create a pattern like e.g. "1 100 " you want to hear something like "BOOM BOOM - - ". But because there is no "gap" between the two "1"s the output looks like a single drum-trigger with double length and your electronic drum would generate a single "BOOOOM - - " .

with the help of an additional relay connected to the output and driven by the clock signal we can insert " 0 "- gaps to separate consecutive "1" - pulses : when clock is " 0 ", this relay switches to OV so OUT = "O", when clock becomes "1", OUT becomes the output of the ringbuffer.
Beside adding more D-FFs to enlarge the drum-pattern from 4 to 8 or even 16 notes and building more of these 1-track machines to get at least a pratically useful 4-track device, we should take a closer look at clock generation now ...

## basic clock generators with relays :

there are many ways to make relays oscillate. For our drum-machine we need a design that can easily be adjusted in frequency, let's say from 30bpm up to 200bmp. And it must be simple, because after soldering together all these DFlipFlops there will be almost no time left to construct the clock generator ;-)


The practical circuit has a small ( 3300 hm ) resistor in series with the k 1 discharge switch to avoid burning the contacts by high discharge currents. A variable resistor (potentiometer) in parallel to the coil makes the chargecurrent and thus charge-time controllable:

to convert the needle-pulses to nice 50/50 square and extend the range towards lower frequencies, we finally add a frequency-divider to the output..

## basic frequency divider with relays :


the most simple binary frequency divider is a so called T-FF (Toggle FlipFlop). It is made from our well known D-FF by connecting the inverted output QNOT back to the data input. That means with each positive clock edge the output $Q$ becomes inverted.
This generates symmetrical pulses with half the clock frequency
function: $Q=0$ means $Q N O T=D=1$. with the next clock pulse this $D=1$ will be stored, so $Q=1$ and that means $Q N O T=D=0$. with the next clock pulse this $D=0$ will be stored, so $Q=0$ and that means $Q N O T=D=1$......

OK, the final circuit of our drum-machine is a "kick-base" circuit. We use it to supply an electromagnet that hammers on a base-drum ( tin-pan or whatever...) with a very narrow, kicking pulse like the hammering mechanics used in a piano :

remember: the output of our ring- buffer has a width of the positive clock pulse.
This input signal goes directly to the upper coil pin of a relay, and delayed by a RCcircuit to the lower coil pin. When the input pulse goes "1", the Vcap. stays low for a while until cap. is loaded. This means the relay switches OUT=IN= "1" for a short time. When the capacitor is charged, there is no voltage-diverence over the coil and the relay switches back ( see XOR gate!). The same delay happens while discharging the cap.: when IN switches from "1" to "0". ,the relay also switches on for a short time, but OUT stays" 0 " because IN is already " 0 ". That why we get a narrow OUT pulse for the rising edge of IN.

