Digital technology textbook (for Olga) (olga.zone)

CONTENT

Foreword

FIRST THE DEVICE NEEDS TO BE CONNECTED TO A VOLTAGE SOURCE. THERE IS A BOX OF AAA BATTERIES WITH A USB C CONNECTOR THAT NEEDS TO BE CONNECTED TO THE SOCKET OPPOSITE THE ROTARY ENCODER. IT IS POSSIBLE TO USE A NORMAL USB C CHARGER. THE DEVICE STARTS IMMEDIATELY BY CONNECTING TO THE VOLTAGE SUPPLY AND AFTER THE INTRO, THE USER IS PUT INTO THE MENU. THE DEVICE IS CONTROLLED ONLY BY A ROTARY ENCODER WITH A BUTTON, 4 SWITCHES ARE TO SIMULATE THE CIRCUIT.

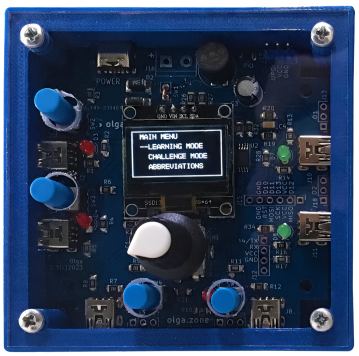
Every circuit part has its description, graphical representation and truth table.

We talk about circuit parts, because the first 16 “gates” are all possible combinational circuit schemes between 2 inputs and 1 output. These are not always gates, there are also completely or partially independent schemes, schemes missing connection to inputs. Our approach was chosen to cover situations in circuits, not just to repeat the 6 basic gates.

In sequential circuits (flipflops and latches), we’ve chosen a different approach, we explain 3 edge triggered flipflops and one level triggered latch (the D circuit) and we “build on top” of this separation. We show only one representative circuit from both T and JK flipflop/latch families. 4 T flipflops/latch exist. 16 JK flipflops/latches exist.

In Czech flipflops and latches are called just by one term - “klopný obvod”. This convention has its benefits.

“!” is the symbol for NOT in C family programming languages (we don’t do “~” because we’re always after resolution of expression that produces inputs and we handle multibit operations at the lowest level, using multi-input logic circuits rather than higher order logic, where ~ vs ! matters).



AND

2 inputs, 1 output
memory 0bit

AND gate is one of the most common gates. Its meaning is exactly the same as in natural language. The symbol in ASCII is “&” (in C family programming languages).

A	B	A AND B
0	0	0
1	0	0
0	1	0
1	1	1

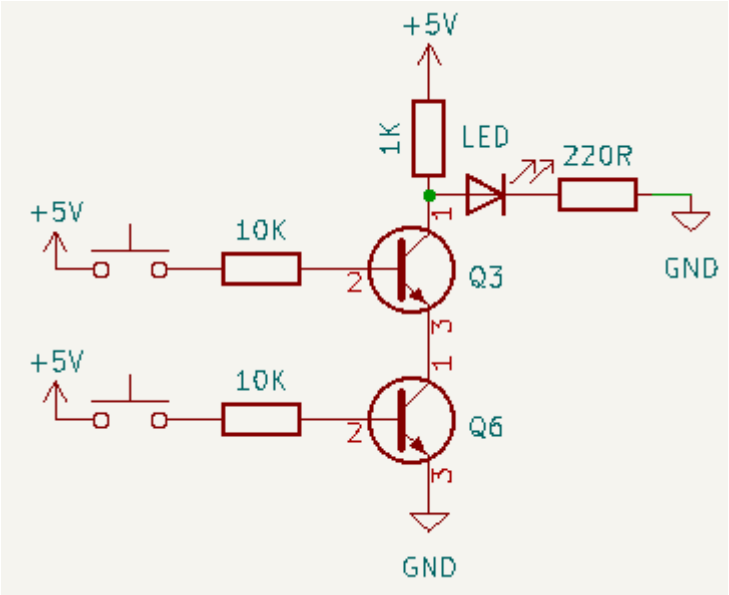
NAND

2 inputs, 1 output
memory 0bit

Most practical approach to memorize A NAND B is to think about “everything but A AND B”. It’s just negation of the output of the AND gate. This component is very widely used in microchip design for its planar simplicity.

A	B	A NAND B
0	0	1
1	0	1
0	1	1
1	1	0

Implementation of a NAND gate using two BJT NPN transistors. LED is an example of an output. Any other gates can be constructed using NAND and NOT gates. To keep the amount of data absorbable in this document, we refer to the usual search engines.



OR

2 inputs, 1 output
memory 0bit

OR gate is one of the most common gates. Its meaning is exactly the same as in natural language, but it's not exclusive, so both A and B matches OR. The symbol in ASCII is "|" (this is used in C family programming languages).

A	B	A OR B
0	0	0
1	0	1
0	1	1
1	1	1

NOR

2 inputs, 1 output
memory 0bit

NOR gate is a gate that’s active only if no input is active. Widely used gate, one of primary, because two transistors are just enough to create it. Not as popular as NAND gate due to non-planar transistor design (in some constructions).

A	B	A NOR B
0	0	1
1	0	0
0	1	0
1	1	0

XOR

2 inputs, 1 output
memory 0bit

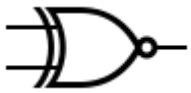
XOR gate is an “exclusive or” gate. If (ONLY A AND NOT B) OR (ONLY B AND NOT A) is active, the gate output is active. In other words if exactly one input is active, output is active. It’s widely used in cryptography and in some circuit designs it is used to negate one of inputs based on the state of the other one (conditional input negation).

A	B	A XOR B
0	0	0
1	0	1
0	1	1
1	1	0

XNOR

2 inputs, 1 output
memory 0bit

XNOR gate is a negation of an “exclusive or” gate. This gate reflects equivalence. The output is active if input A equals input B.



A	B	A XNOR B
0	0	1
1	0	0
0	1	0
1	1	1

CONTRADICTION

Any number of inputs, 1 output
memory 0bit

Contradiction is not a gate, it’s a situation, where any input produces logical 0 - false. It doesn’t have a widely popular symbol, in circuit it is represented as a ground.

A	B	CONTRADICTION
0	0	0
1	0	0
0	1	0
1	1	0

TAUTOLOGY

Any number of inputs, 1 output
memory 0bit

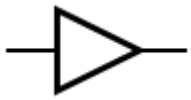
Tautology is not a gate, it’s a situation, where any input produces logical 1 - true. It doesn’t have a widely popular symbol, in circuit it is represented as Vcc (positive voltage).

A	B	TAUTOLOGY
0	0	1
1	0	1
0	1	1
1	1	1

BUFA

2 inputs, 1 output
memory 0bit

Buffer A is not a gate, it’s a situation where input A is sent directly to output. It doesn’t necessarily have a symbol (below), in circuit it is represented as a buffer or just a connection between A and output, completely ignoring B.



A	B	BUFA
0	0	0
1	0	1
0	1	0
1	1	1

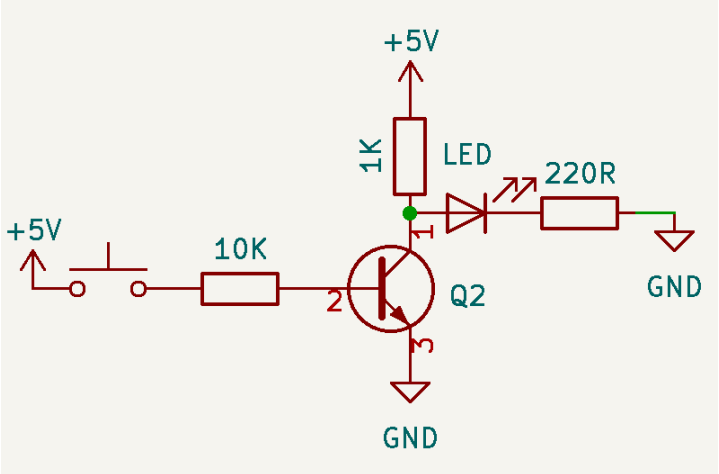
NOTA

2 inputs, 1 output
memory 0bit

“Not A” is not a gate, it’s a situation where input A is sent to output via NOT gate. Negation (NOT gate) has its symbol below, but NOTA connection represents a situation in a circuit where A is connected to output via NOT gate, completely ignoring B (despite B exists).

A	B	NOTA
0	0	1
1	0	0
0	1	1
1	1	0

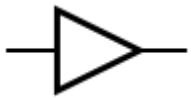
Implementation of a NOT gate using one BJT NPN transistor. LED is an example of an output. Any other gates can be constructed using NAND and NOT gates. To keep the amount of data absorbable in this document, we refer to the usual search engines.



BUFB

2 inputs, 1 output
memory 0bit

Buffer B is not a gate, it’s a situation where input B is sent directly to output. It doesn’t necessarily have a symbol (below), in circuit it is represented as a buffer or just a connection between B and output, completely ignoring A.



A	B	BUFB
0	0	0
1	0	0
0	1	1
1	1	1

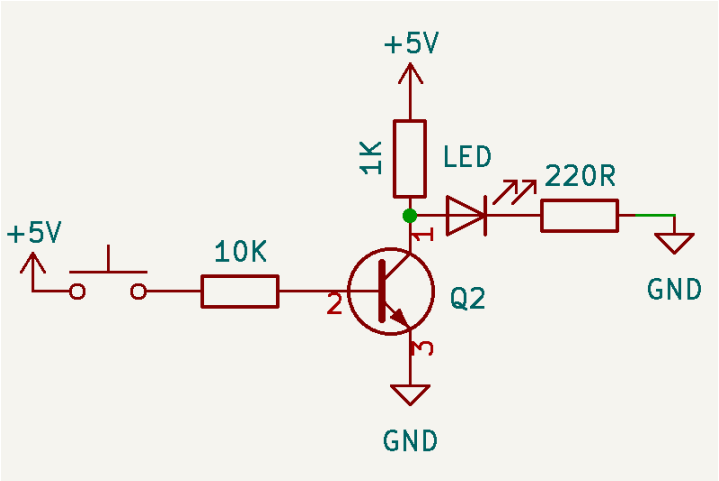
NOTB

2 inputs, 1 output
memory 0bit

“Not B” is not a gate, it’s a situation where input B is sent to output via NOT gate. Negation (NOT gate) has its symbol below, but NOTB connection represents a situation in a circuit where B is connected to output via NOT gate, completely ignoring A (despite A exists).

A	B	NOTB
0	0	1
1	0	1
0	1	0
1	1	0

Implementation of a NOT gate using one BJT NPN transistor. LED is an example of an output. Any other gates can be constructed using NAND and NOT gates. To keep the amount of data absorbable in this document, we refer to the usual search engines.



AIB

2 inputs, 1 output
memory 0bit

A implies B is a rarely used “gate”, that is typically constructed from 2 other gates. It doesn’t have a commonly used symbol.

A	B	AIB
0	0	1
1	0	1
0	1	0
1	1	1

ANB

2 inputs, 1 output
memory 0bit

“Anything but A implies B” is a rarely used “gate” that is typically constructed from 2 other gates. It doesn’t have a commonly used symbol.

A	B	ANB
0	0	0
1	0	0
0	1	1
1	1	0

BIA

2 inputs, 1 output
memory 0bit

B implies A is a rarely used “gate”, that is typically constructed from 2 other gates. It doesn’t have a commonly used symbol.

A	B	BIA
0	0	1
1	0	0
0	1	1
1	1	1

BNA

2 inputs, 1 output
memory 0bit

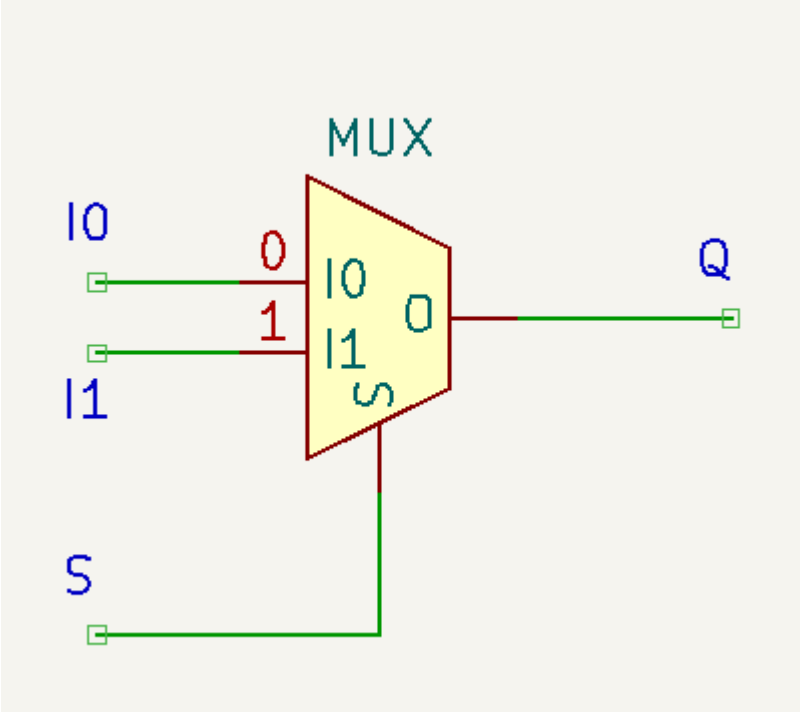
“Anything but B implies A” is a rarely used “gate” that is typically constructed from 2 other gates. It doesn’t have a commonly used symbol.

A	B	BNA
0	0	0
1	0	1
0	1	0
1	1	0

2:1 MUX

3 inputs, 1 output
memory 0bit

2:1 multiplexer (mux) is a circuit that transmits input I0 to Q if S is low and input I1 to Q if S is high. Higher order multiplexers exist.

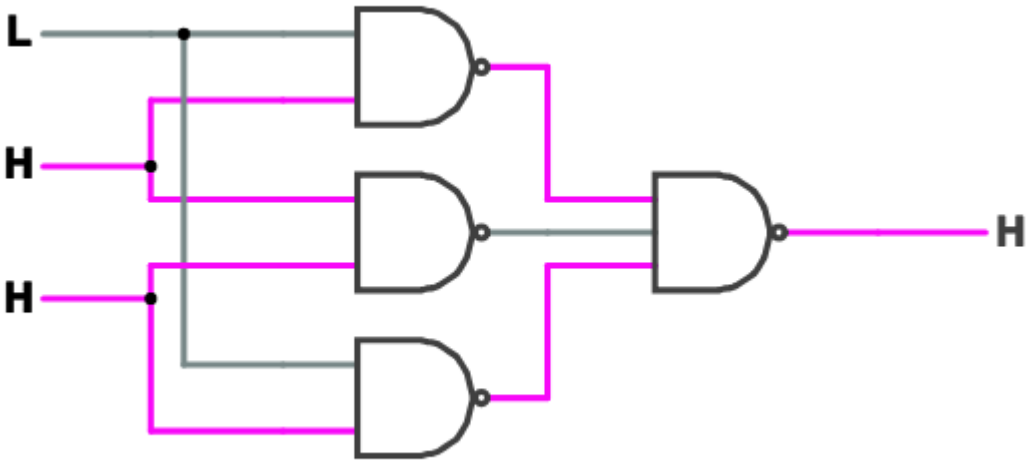


I0	I1	S	Q
0	X	0	0
1	X	0	1
X	0	1	0
X	1	1	1

MAJORITY

3 inputs, 1 output
memory 0bit

Majority circuit outputs what's majority in 3 inputs.

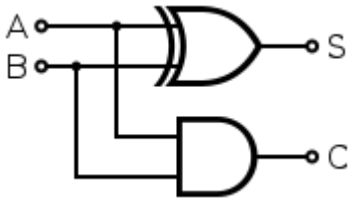


I1	I2	I3	Q
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

HALF ADDER

2 inputs, 2 outputs
memory 0bit

The half adder adds two single binary digits A and B. It has two outputs, sum (S) and carry (C). The carry signal represents an overflow into the next digit of a multi-digit addition.

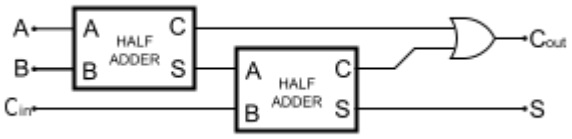


A	B	S	C
0	0	0	0
1	0	1	0
0	1	1	0
1	1	1	1

FULL ADDER

3 inputs, 2 outputs
memory 0bit

Full adder processes two inputs and a carry input to produce a sum and a carry output. The OR gate processes both possible overflows.

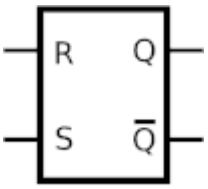


A	B	Cin	S	Cout
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	1	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

NOR RS LATCH

2 inputs, 1 output (+negation)
memory 1bit

While the R and S inputs are both low, feedback maintains the Q and !Q outputs in a constant state, with !Q the complement of Q. If S (Set) is pulsed high while R (Reset) is held low, then the Q output is forced high, and stays high when S returns to low; similarly, if R is pulsed high while S is held low, then the Q output is forced low, and stays low when R returns to low. “!” is the symbol for NOT in C family programming languages.
Clocked latch of this kind exists, E - enable input via 2 AND gates with R and S, enabled input allows changes on input changes. It may be confused with JK latch that just doesn’t have invalid state.



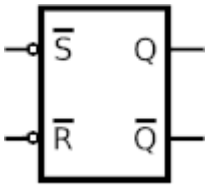
state transition table			
S	R	Qnext	Action
0	0	Q	Hold State
0	1	0	Reset
1	0	1	Set
1	1	X	Not Allowed

NAND RS LATCH

2 inputs, 1 output (+negation)
memory 1bit

While the R and S inputs are both high, feedback maintains the Q and !Q outputs in a constant state, with !Q the complement of Q. If S (Set) is pulsed low while R (Reset) is held high, then the Q output is forced high, and stays high when S returns to high; similarly, if R is pulsed low while S is held high, then the Q output is forced low, and stays low when R returns to high. “!” is the symbol for NOT in C family programming languages.

Gated (or clocked) latch of this kind exists, E - enable input via 2 NAND gates on R and S input enable output changes on input changes. It may be confused with JK latch that just doesn’t have invalid state.



state transition table			
S	R	Qnext	Action
0	0	X	Not Allowed
0	1	0	Set
1	0	1	Reset
1	1	Q	Hold State

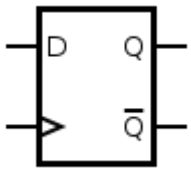
D LATCH

2 inputs, 1 output (+negation)
memory 1bit

The D latch (D for "data") or transparent latch is a simple extension of the gated SR latch that removes the possibility of invalid input states. We take input D, connect it directly to set and via NOT (negation) to Reset. The D latch outputs the D input whenever the Clock input is high, otherwise the output is whatever the D input was when the Clock input was last high. This is why it is also known as a transparent latch - when Clock is active, the latch is said to be "transparent" - signals propagate directly through it as if it isn't there.

If the clock is high, the D latch responds to input immediately. During one clock pulse there may be for example 5 input changes, that change the output Q immediately. We say D latch is level triggered (by clock).

The symbol is typically not distinguished for D latch and D flipflops. The state transition table is also the same.



state transition table			
CLK	D	Qnext	Action
0	0	Q	Hold State
0	1	Q	Hold State
1	0	0	Reset
1	1	1	Set

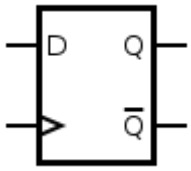
RISING EDGE D FLIPFLOP

2 inputs, 1 output (+negation)
memory 1bit

Rising edge D flipflop is a flipflop, that changes output using the same rules as D latch, but only at the moment of rising edge of clock.

If the clock is becoming high, the D flipflop responds to input. During one clock pulse there is always only one possible output change. We say D latch is edge triggered (by clock).

The symbol is typically not distinguished for D latch and D flipflops. The state transition table is also the same.



state transition table			
CLK	D	Qnext	Action
0	0	Q	Hold State
0	1	Q	Hold State
1	0	0	Reset
1	1	1	Set

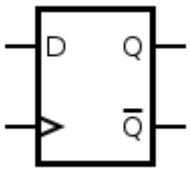
FALLING EDGE D FLIPFLOP

2 inputs, 1 output (+negation)
memory 1bit

Falling edge D flipflop is a flipflop, that changes output using the same rules as D latch, but only at the moment of falling edge of clock.

If the clock is becoming low, the D flipflop responds to input. During one clock pulse there is always only one possible output change. We say D flipflop is falling edge triggered (by clock).

The symbol is typically not distinguished for D latch and D flipflops. The state transition table is also the same.



state transition table			
CLK	D	Qnext	Action
0	0	Q	Hold State
0	1	Q	Hold State
1	0	0	Reset
1	1	1	Set

DUAL EDGE D FLIPFLOP

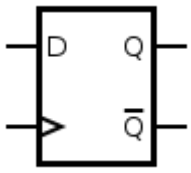
2 inputs, 1 output (+negation)
memory 1bit

Dual edge D flipflop is a flipflop, that changes output using the same rules as D latch, but only at the moment of rising edge and falling edge of clock.

If the clock is becoming low or high, the D flipflop responds to input. During one clock pulse there are always up to two possible output changes. We say D flipflop is dual edge triggered (by clock).

The symbol is typically not distinguished for D latch and D flipflops. The state transition table is also the same.

We construct dual edge D flipflop using both rising and falling edge flipflops and muxing them 2:1 together by clock.

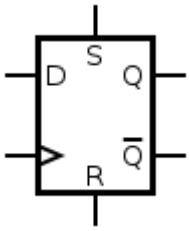


state transition table			
CLK	D	Qnext	Action
0	0	Q	Hold State
0	1	Q	Hold State
1	0	0	Reset
1	1	1	Set

RISING EDGE SET-RESET D FLIPFLOP

4 inputs, 1 output (+negation)
memory 1bit

Set-Reset D flipflop (any one) is a flipflop with additional inputs S and R, which may force output to be high or low. Other properties are the same as in the flipflop that S and R enhance. Flipflops with just S, or just R also exist.

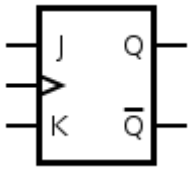


state transition table					
CLK	D	S	R	Qnext	Action
0	0	0	0	Q	Hold State
0	1	0	0	Q	Hold State
1	0	0	0	0	Reset
1	1	0	0	1	Set
X	X	1	0	1	Force Set
X	X	0	1	0	Force Reset
X	X	1	1	X	Not Allowed

FALLING EDGE JK FLIPFLOP

3 inputs, 1 output (+negation)
memory 1bit

Many JK flipflops-latches exist. The inputs may behave as one of the four D flipflops/latch. There are two inputs, therefore there are 16 combinations, including a pure JK latch, level triggered J and edge triggered K, level triggered K and edge triggered J, or edge triggered J and edge triggered K, where edge triggered means falling edge, rising edge or dual edge triggered behavior. They may also toggle in a different way.
The JK latch is an SR latch that is made to toggle its output (oscillate between 0 and 1) when passed the input combination of 11. Unlike the JK flip-flop, the 11 input combination for the JK latch is not very useful because there is no clock that directs toggling.
If we make the J and K react on the edge of the clock, we receive for example a falling edge JK flipflop. This circuit resolves both the J and the K situation on the falling edge of the clock. The JK flipflop augments the behavior of the SR flip-flop (J: Set, K: Reset) by interpreting the J = K = 1 condition as a "flip" or toggle command. Specifically, the combination J = 1, K = 0 is a command to set the flipflop; the combination J = 0, K = 1 is a command to reset the flipflop; and the combination J = K = 1 is a command to toggle the flipflop, i.e., change its output to the logical complement of its current value. Setting J = K = 0 maintains the current state. To synthesize a D flip-flop, simply set K equal to the complement of J (input J will act as input D). Similarly, to synthesize a T flipflop, set K equal to J. The JK flip-flop is therefore a universal flip-flop, because it can be configured to work as an SR flip-flop, a D flip-flop, or a T flip-flop.
“!” is the symbol for NOT in C family programming languages.



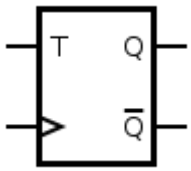
state transition table			
J	K	Qnext	Action
0	0	Q	Hold State
1	0	1	Set
0	1	0	Reset
1	1	!Q	Toggle

FALLING EDGE T FLIPFLOP

2 inputs, 1 output (+negation)
memory 2bit

The T flip-flop can be built using a JK flip-flop (J and K pins are connected together and act as T) or a D flip-flop (T input XOR Qprevious drives the D input). “!” is the symbol for NOT in C family programming languages.

When T is active, the T flipflop toggles by the clock. Four types of T flipflop/latch exist: 3 edge triggered and one level triggered. Level triggered - T latch is not widely used, because it toggles immediately and not by the clock.

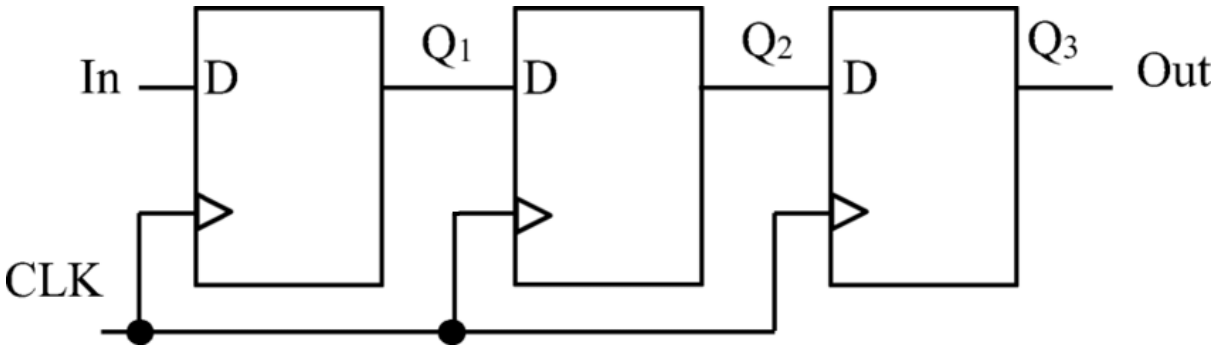


state transition table			
CLK	T	Qnext	Action
0	0	Q	Hold State
1	0	Q	Hold State
0	1	!Q	Toggle
1	1	!Q	Toggle

RISING EDGE 3BIT SISO SHIFT REGISTER

2 inputs, 1 output
memory 3bit

The shift register, which allows serial input (one bit after the other through a single data line) and produces a serial output is known as a Serial-In Serial-Out shift register. Since there is only one output, the data leaves the shift register one bit at a time in a serial pattern, thus the name Serial-In Serial-Out Shift Register. The logic circuit given below shows a serial-in serial-out shift register. The circuit consists of three D flip-flops which are connected in a serial manner. All these flip-flops are synchronous with each other since the same clock signal is applied to each flip-flop.

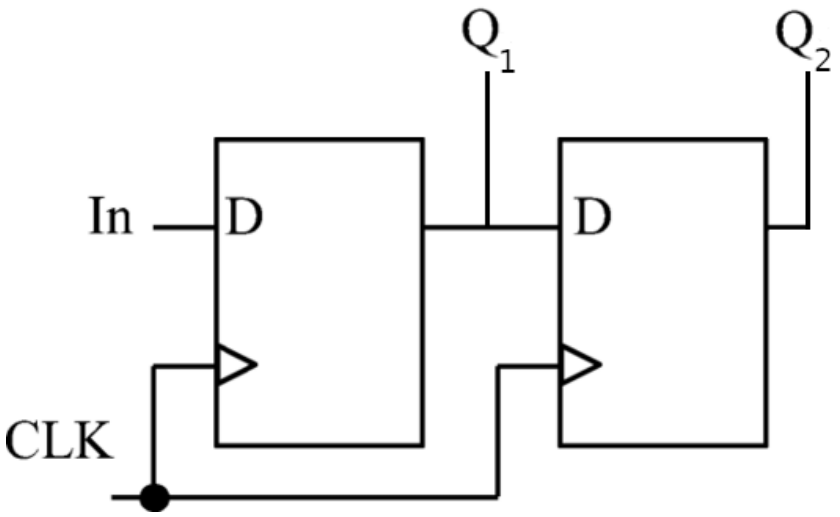


state transition table				
CLK #	In	(Q1)	(Q2)	Q3 = Out
0	0	0	0	0
1	0	0	0	0
2	1	0	0	0
3	0	1	0	0
4	1	0	1	0
5	0	1	0	1
6	0	0	1	0
7	0	0	0	1
8	0	0	0	0

RISING EDGE 2BIT SIPO SHIFT REGISTER

2 inputs, 2 outputs
memory 2bit

In digital circuits, a shift register is a cascade of flip flops, sharing the same clock, in which the output of each flip-flop is connected to the "data" input of the next flip-flop in the chain. Resulting circuit shifts by one position the "bit array" stored in it, shifting in the data present at its input and shifting out the last bit in the array, at each transition of the clock input.
What is SIPO Shift Register? The shift register which allows serial input parallel output is known as the SIPO shift register. In the SIPO register, the term SIPO stands for serial input parallel output.

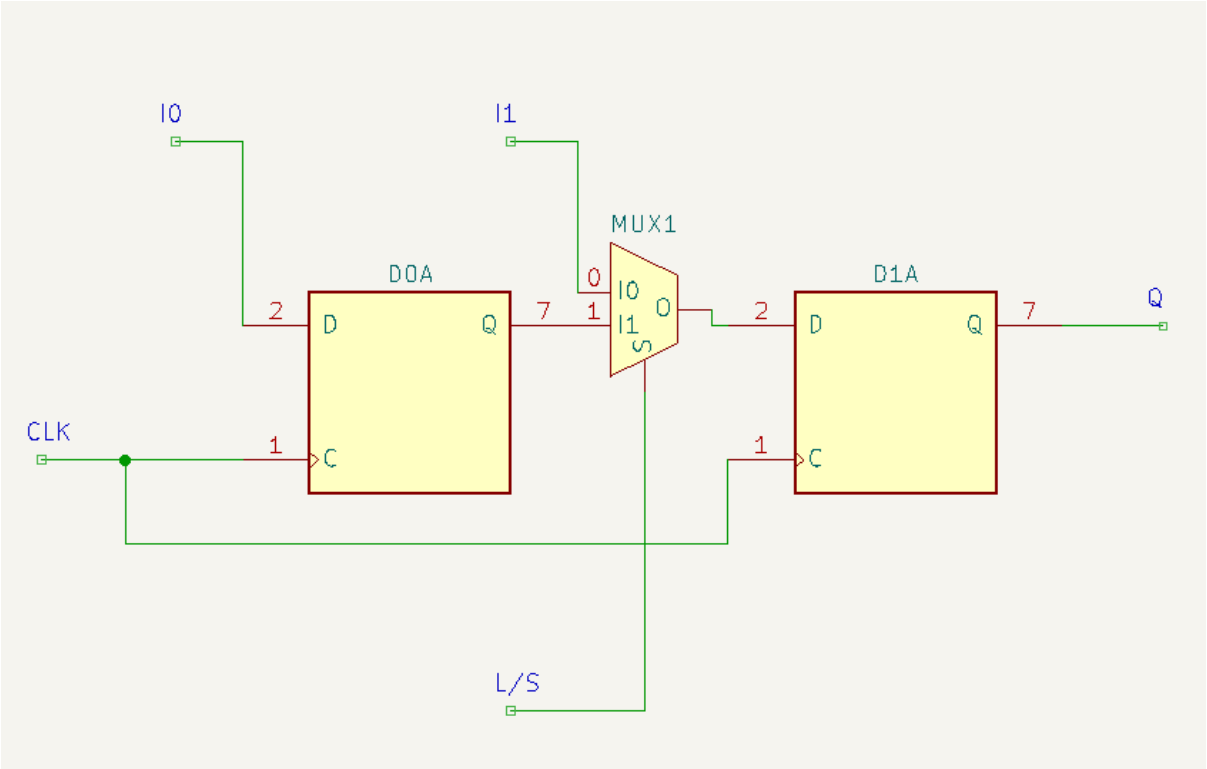


state transition table			
CLK #	In	Q1	Q2
1	0	0	0
2	1	0	0
3	0	1	0
4	1	0	1
5	0	1	0
6	0	0	1

RISING EDGE 2BIT PISO SHIFT REGISTER

4 inputs, 1 output
memory 2bit

PISO (Parallel In Serial Out) shift register is a device that loads parallel inputs to the serial line. Its work is determined by L/S input (Load/Shift), which determines if the circuit works like SISO or loads data from inputs. If Shift is selected, the circuit moves data to the next D flipflop. Each couple of D flipflops is separated by mux between input and previous D flipflop based on L/S. The circuit shares a common clock in all D flipflops.

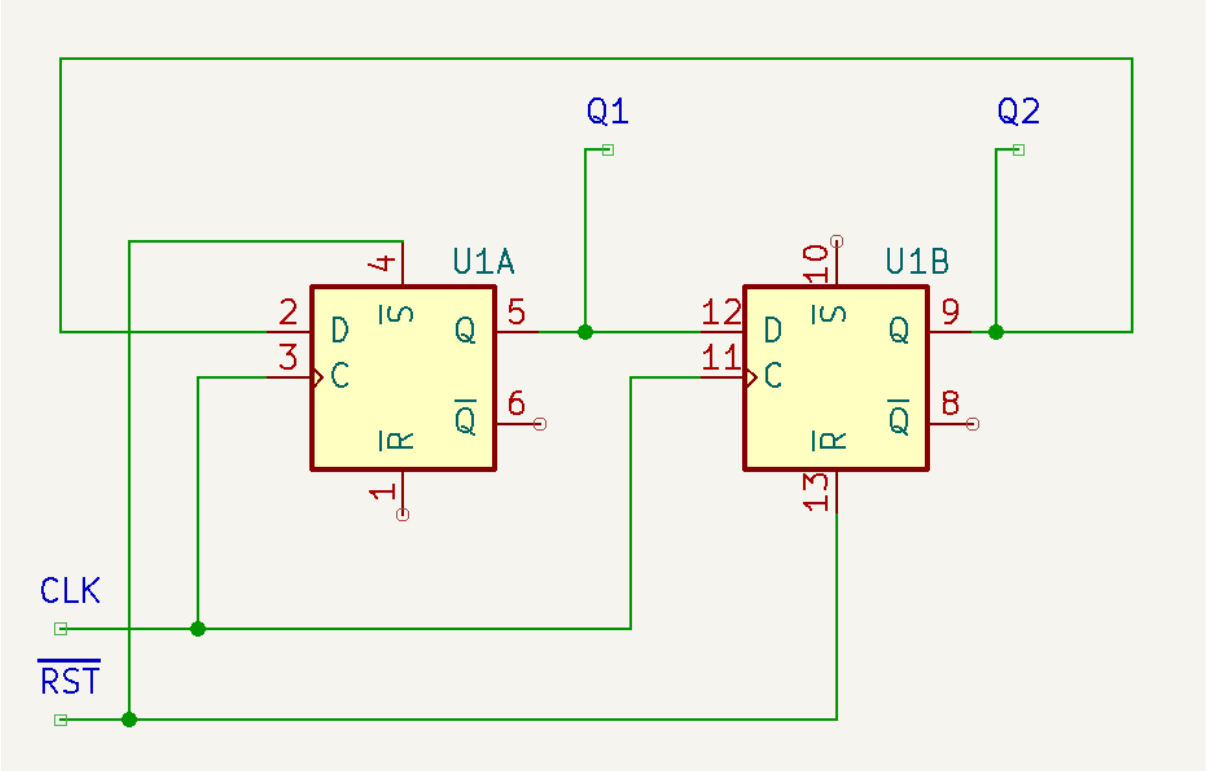


example state transition table (input 11, shift, shift)						
CLK #	L/S	I0	D0	I1	D1	Q
1	0	0	0	0	0	0
2	0	1	0	1	0	0
3	1	1	1	1	1	1
4	1	0	0	0	1	1
5	0	0	0	0	0	0

STRAIGHT RING COUNTER

2 inputs, 2 outputs
memory 2bit

A straight ring counter, also known as a one-hot counter, connects the output of the last shift register to the first shift register input and circulates a single one (or zero) bit around the ring.

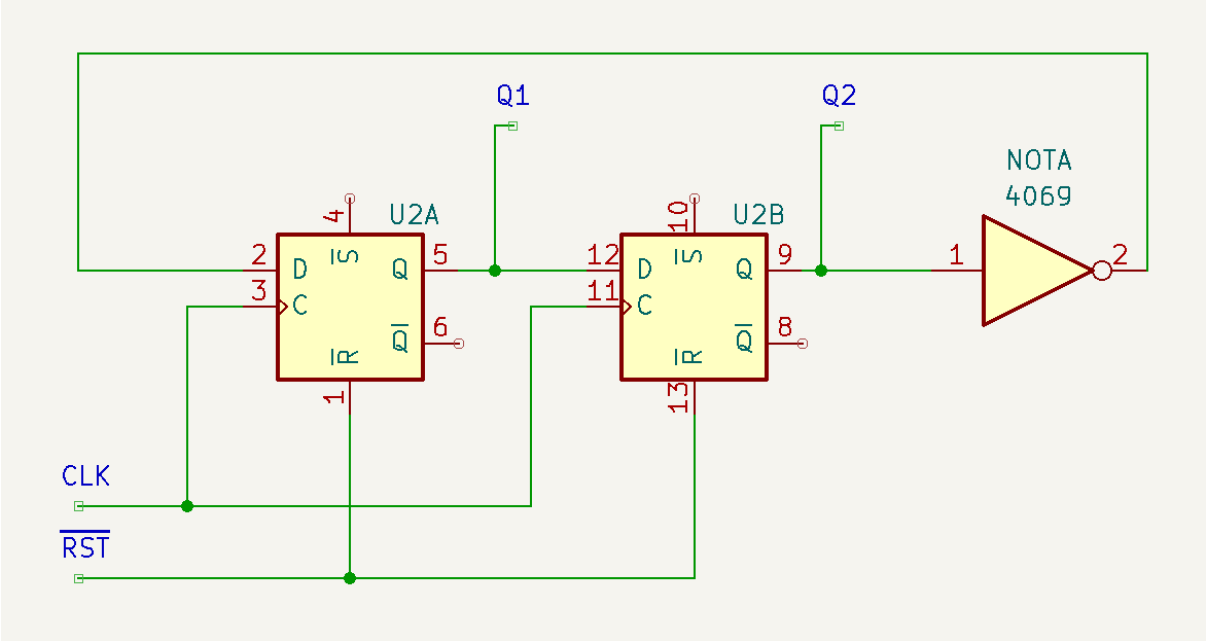


state transition table			
CLK #	STATE	Q1	Q2
1	0	0	0
2	1	1	0
3	2	0	1
4	0	0	0

JOHNSON RING COUNTER

2 inputs, 2 outputs
memory 2bit

A twisted ring counter, also called switch-tail ring counter, walking ring counter, Johnson counter, or Möbius counter, connects the complement of the output of the last D flipflop to the input of the first D flipflop and circulates a stream of ones followed by zeros around the ring.

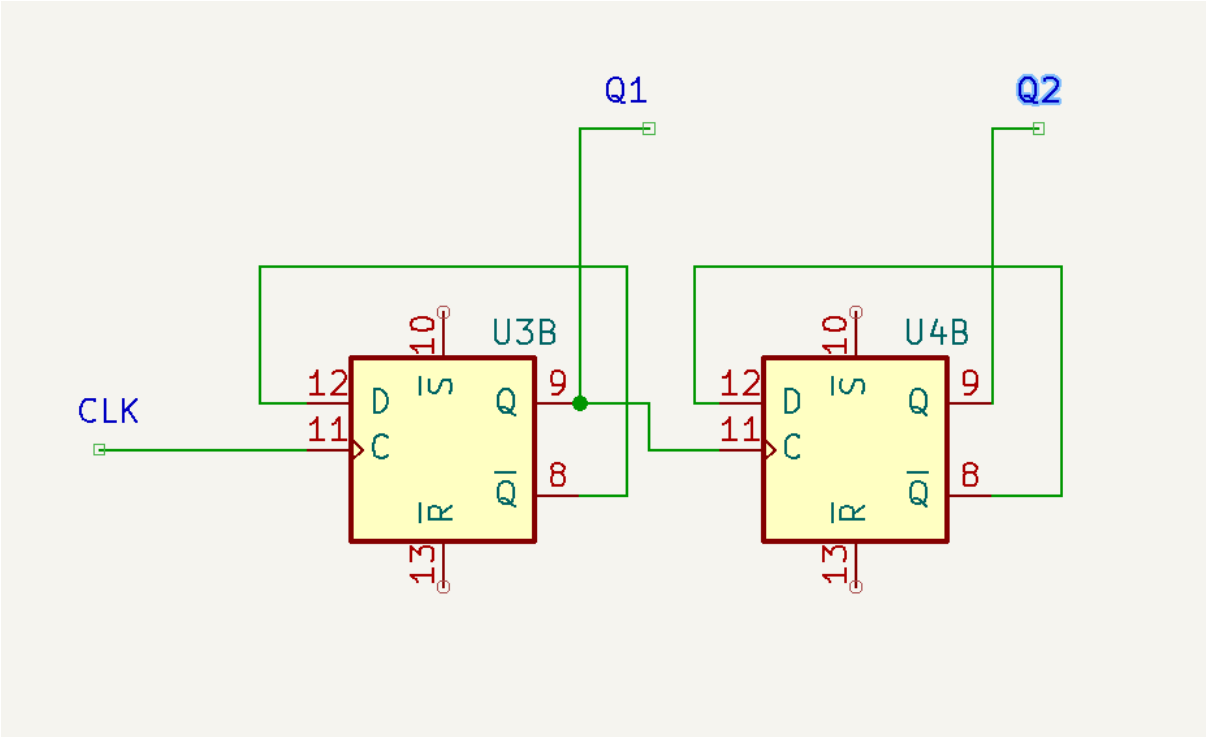


state transition table			
CLK #	STATE	Q1	Q2
1	0	0	0
2	1	1	0
3	2	1	1
4	3	0	1
5	0	0	0

MOD-4 COUNTER

1 input, 2 outputs
memory 2bit

MOD-4 counter is a modulus counter that increases its state on clock pulse. When the state overflows, the counter returns to state 0 (modulus behavior). Q1 represents MSB (most significant bit), Q2 represents LSB (least significant bit). Circuits of this family are constructed connecting less significant bit D flipflop output to clock of more significant bit D flipflop, connecting its output's complement to its D input, making inverting feedback.

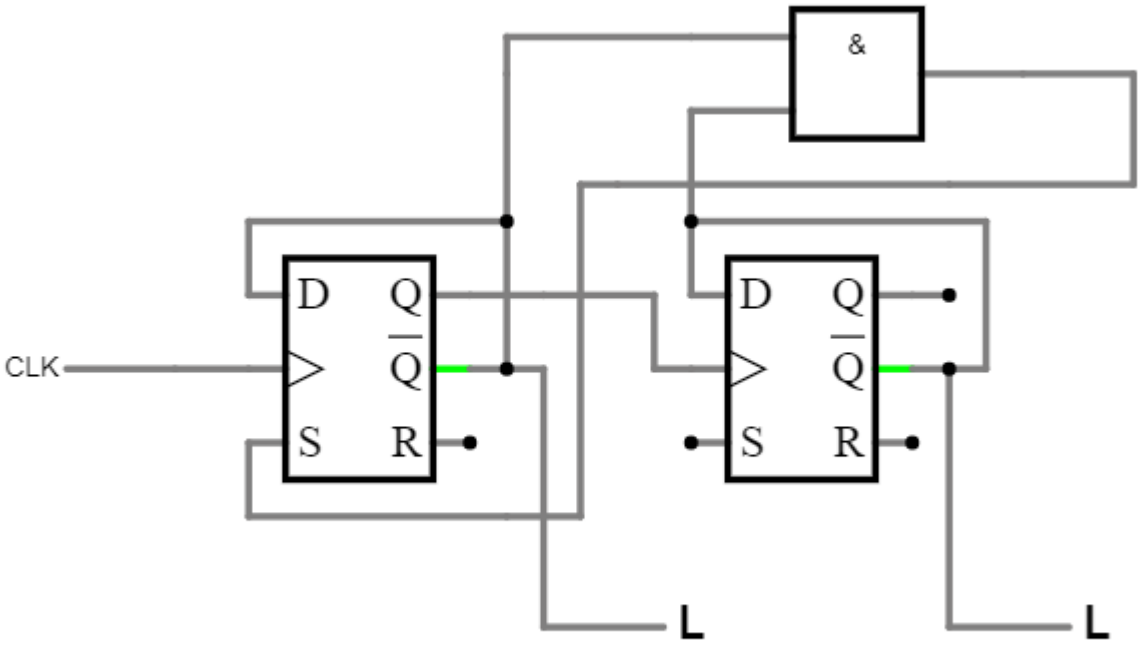


state transition table			
CLK #	STATE	Q1 (MSB)	Q2 (LSB)
1	0	0	0
2	1	0	1
3	2	1	0
4	3	1	1
5	0	0	0

MOD-3 COUNTER

1 input, 2 outputs
memory 2bit

MOD-3 counter is equal to MOD-4 counter, but it contains a decoding circuit that sets the counter to its 00 state on 11 (invalid) state. In this circuit active low Set on the first D flipflop forces NAND to be used as such a decoding circuit. Using similar logic we can create MOD ANYTHING counters.

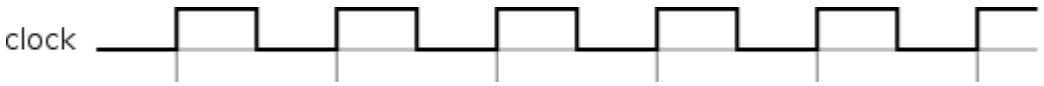


state transition table			
CLK #	STATE	Q1	Q2
1	0	0	0
2	1	0	1
3	2	1	0
4	0	0	0

CLOCK

0 input, 1 output
memory 0bit

Clock is a signal that toggles between 1 and 0 forming rectangle shaped pulses. It’s characterized by the voltage and frequency. OLGA.ZONE has a 1Hz frequency, therefore one period lasts 1 second.



RESOURCES

https://www.falstad.com/circuit/
https://commons.wikimedia.org/wiki/Logic_gates_unified_symbols
https://en.wikipedia.org/wiki/Flip-flop_(electronics)#Flip-flop_types
https://en.wikibooks.org/wiki/Digital_Circuits/Latches
https://en.wikipedia.org/wiki/Majority_function
https://en.wikipedia.org/wiki/Adder_(electronics)
https://cs.wikipedia.org/wiki/Bin%C3%A1rn%C3%AD_s%C4%8D%C3%ADta%C4%8Dka
https://www.geeksforgeeks.org/shift-registers-in-digital-logic/
https://www.researchgate.net/figure/Block-diagram-of-3-bit-SISO-shift-register_fig2_335640645
https://en.wikipedia.org/wiki/Ring_counter

PODĚKOVÁNÍ

I thank Jakub Bösser for his generous help with the translation into Czech.
Olga hardware would be impossible without Vladimir Chromy and Martin Bereza.