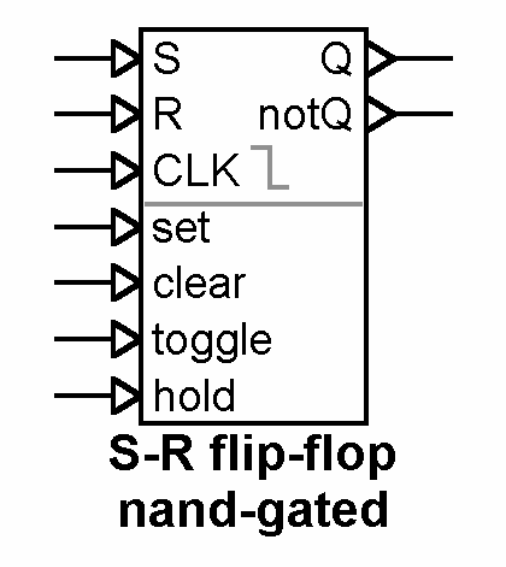
Flip-flop: S-R nand-gated falling-clock full-override



1 Description

This device is an implementation of a nand-gated S-R flip-flop with falling-edge clock and full override controls. For a simple version without the override controls, use the device "S-R nand-gated falling-clock".

1.1 Pins

This device has nine pins:

pin	*type*	*description*
S	input	S input
R	input	R input
CLK	input	falling-edge clock
set	input	set override
clear	input	clear override
toggle	input	toggle override
hold	input	hold override
Q	output	Q output
notQ	output	notQ output

1.2 Parameters

The initial value of *Q* must be defined if the device is possibly holding or toggling at *t=0*. When the device operates in clearing or setting mode at *t=0*, the initial value is ignored.

The value of the *stepped_mode* flag determines whether the device operates in *stepped* or *ramped* mode.
In *stepped* mode (the default for ideal logical signals), the outputs are represented as stepped signals, where changes in value are observed as vertical steps at the time they occur. In *ramped* mode, the value transitions of the outputs are seen as ramps between *t-Δt* and *t*.

parameter	*description*
Q_ini	initial value of Q if holding or toggling at t=0
stepped_mode	=1 to indicate stepped mode (default) =0 to indicate ramped mode

1.3 Input

The input pins may be connected to any control signals.

Numerical input values are automatically interpreted as logical values by this device, as follows:

input	*converted logical value*	*logical value representation*
value > 0	true	1
value ≤ 0	false	0

1.4 Output

The outputs are *Q* and its logical inverse *notQ*. Their representation as *stepped* or *ramped* signals is determined by the value given to the parameter *stepped_mode*.

The numerical representation of the output logical values is:

output logical value	*output numerical value*
true	1
false	0

1.5 Representation

The implementation of the model can be inspected by opening the device's subcircuit.

The model applies the following logic for determining its state:

rule sequence	*action*	*output*
if set>0	setting	Q(t) = 1
else if clear>0	clearing	Q(t) = 0
else if toggle>0	toggling	Q(t) = not Q(t-Δt)
else if hold>0	holding	Q(t) = Q(t-Δt)
else if not triggering	holding	Q(t) = Q(t-Δt)
else if S>0 and R>0	setting	Q(t) = 1
else if S<=0 and R<=0	holding	Q(t) = Q(t-Δt)
else if S>0	setting	Q(t) = 1
else if S<=0	clearing	Q(t) = 0
endif		

if holding at t=0	use Q_ini	Q(0) = Q_ini
if toggling at t=0	use Q_ini	Q(0) = Q_ini

where triggering occurs on a falling edge on the clock signal.