

Sequential Circuits

So far we have assumed that the steady-state output of a network is dependent only upon the values assigned to the input switching variables – there is no memory of the previous output of the network nor of the values of the internal lines.

Definition: The output of a **sequential** circuit depends upon the **current input pattern and the current state of the circuit.**

The state of the circuit is represented by the values of certain internal variables that are maintained over time. It “remembers” information about the previous input pattern(s).

Two basic types of sequential circuits:

- Fundamental mode (asynchronous): inputs are assumed to be constant long enough for all variables (internal and output) to achieve steady state. State and output changes occur in response to the change in any input line.
- Pulse mode (synchronous): inputs and circuit function are synchronized with an external pulse signal. (Typically but not always a clock.)

Most sequential circuits you will encounter are pulse mode. (as are most digital systems at some level)

Pulse mode circuits depend upon basic memory components that are fundamental mode.

Additionally, fundamental mode is used when high performance is required. It is much more complicated to design fundamental mode than pulse mode.

We will consider first how fundamental mode circuits can be used to implement a memory device.

The internal variables are stored via the use of **combinational feedback**, i.e., somewhere in the network the output of a gate is connected to the input of an “upstream” gate thereby forming a cycle in the network.

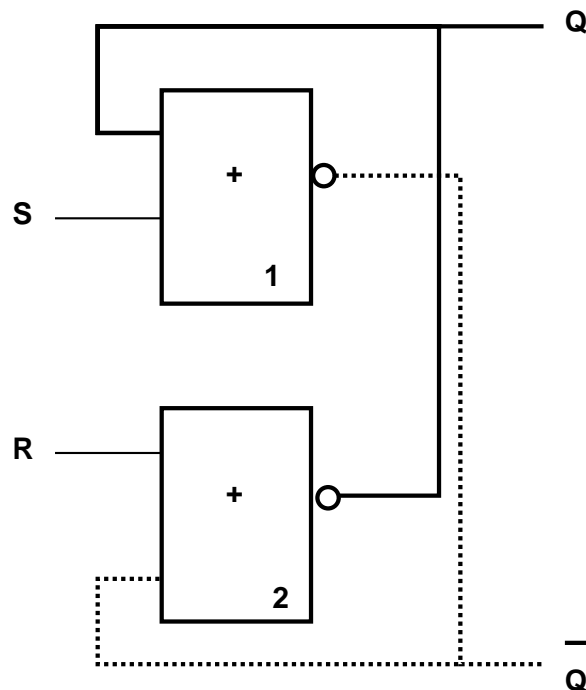
Latches

A **latch** is a simple fundamental mode sequential circuit that uses combinational feedback to store a signal value.

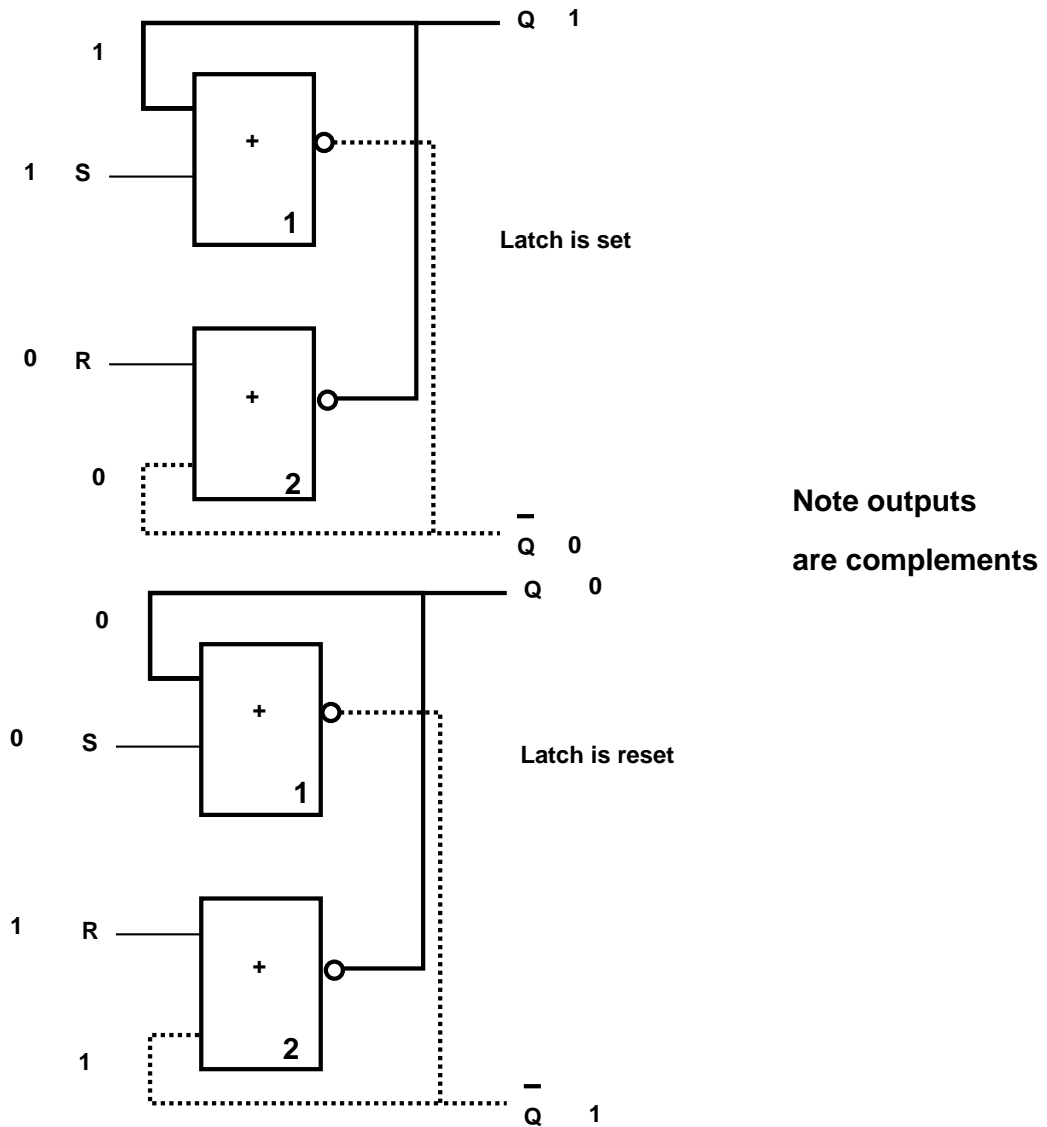
Once the latch properties are understood and an effective design produced, they can be used as basic memory device and combined with other latches and combinational logic in a larger fundamental mode or pulse mode circuit.

We will consider an **SR-latch** or a set-reset latch.

NOR gate implementation of an SR-latch



Two stable states are of most interest



For either steady-state situation above,
 if the input that was 1 is changed to 0 the latch
 remains set or reset, i.e., the state (Q , \bar{Q}) does not change
 since gate 1 and gate 2 are "opaque" respectively.

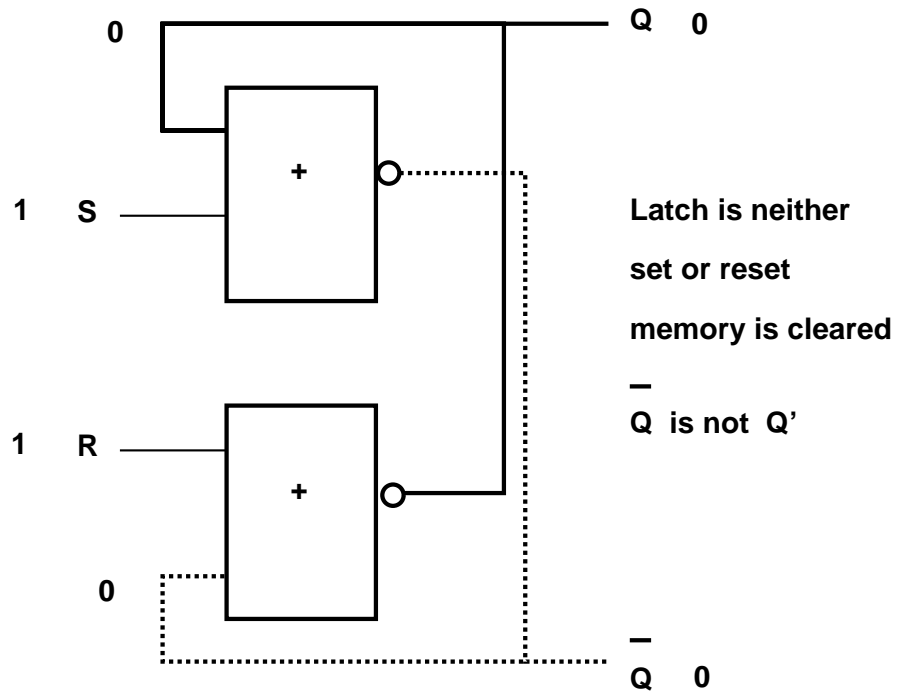
- **SAVE:** When $(S, R) \rightarrow (0, 0)$, the latch acts as a memory device to remember which of the two lines S or R was set to 1 last (with the other set to 0).
- **SET/RESET:** The state of the latch – the value of (Q, \bar{Q}) – before setting $(S, R) \rightarrow (1, 0)$ or $(S, R) \rightarrow (0, 1)$ does not matter. The value of (Q, \bar{Q}) must settle to $(1, 0)$ or $(0, 1)$ respectively.

Set a stored reset state		
SR	$Q\bar{Q}$	
00	01	steady state
10	00	gate 1 drops transient $\bar{Q} \neq Q'$
10	10	gate 2 rises
10	10	feedback stopped since gate 1 is “opaque” due to $S = 1$

Reset a stored set state		
SR	$Q\bar{Q}$	
00	10	steady state
01	00	gate 2 drops transient $\bar{Q} \neq Q'$
01	01	gate 1 rises
01	01	feedback stopped since gate 2 is “opaque” due to $R = 1$

01 \leftrightarrow 10 through 11		
SR	$Q\bar{Q}$	
10	10	steady state
11	00	gate 2 drops transient $\bar{Q} \neq Q'$ holds until S drops
01	01	S drops, gate 1 rises
01	01	feedback stopped since gate 2 is "opaque" due to $R = 1$

$S = R = 1$ is OK as a transient state. What about steady state?



If (S, R) is changed to (0, 0) the cleared state is not maintained. The latch must pass through a state of (1, 0) or (0, 1) depending on gate delays. Therefore, it will settle to a set or reset state indeterminately when starting from the cleared memory state.

- $S = R = 1$ and $Q = \bar{Q} = 0$ is a steady state condition NOT a transient $\bar{Q} \neq Q'$ situation as before (so we cannot necessarily protect against its effect with downstream hazard-free design).
- Even if we design under the assumption that $\bar{Q} \neq Q'$, the state $Q = \bar{Q} = 0$ CANNOT be saved reliably without strict control of the timing of the input to the latch:
 - if S or R changes first then set or reset is new steady state
 - if S and R change to 0 simultaneously (whatever that may mean given the response time of the NOR gates) an oscillation results. Both Q and \bar{Q} transition $0 \rightarrow 1 \rightarrow 0$ etc. until one of the feedback signals makes the change appear not to be simultaneous.
 - Essentially you have two gates trying to force the change of state of the machine. The two changes are racing each other around the feedback loop without an opaque gate to stop the cycle. (We will revisit this issue later when we discuss hazards in sequential machines.)
- Usually, $S = R = 1$ is not allowed as an input pattern in this form of SR latch.

The behavior of the simple SR latch with $S = R = 1$ forbidden can be summed up in a simple formula which relates the new state (Q, \bar{Q}) , the old state (q, \bar{q}) and (S, R) :

$$Q = S + R'q$$

$$\bar{Q} = \bar{Q}'$$

S	R	
0	0	$Q = q$
1	0	$Q = 1$
0	1	$Q = 0$

This is called the **characteristic function** of the SR latch.

The behavior of the latch described thus far relies on the settling behavior of the network of NOR gates just as if it was a combinational network, i.e., any delay in the new output appearing is caused only by gate delay.

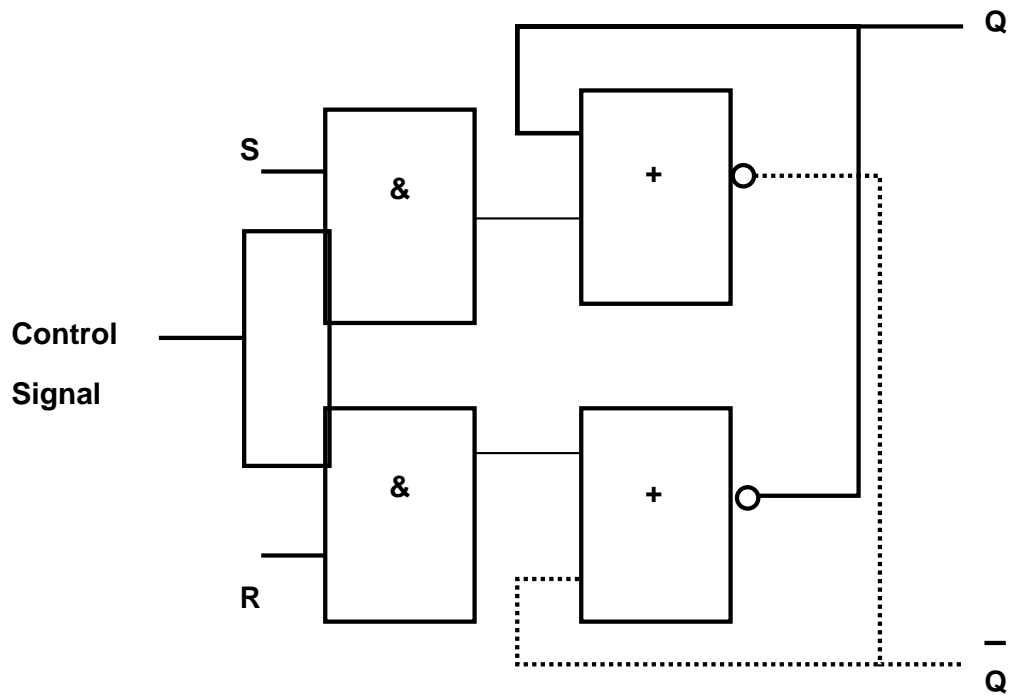
Such latches are called **transparent latches**.

This behavior is characteristic of fundamental mode devices. State and output change in direct response to input changes when they occur (typically only one input may change at a time).

It is also possible to have latches with some **synchronous** inputs. For synchronous S and R inputs, each input signal is passed through an AND gate along with a control signal. Therefore, the S or R input to the latch is only sent when the control signal is 1. (See the figure on the next page.) Such latches are called **gated latches**.

This is not the same as a pulse mode or synchronous sequential circuit. The latch is transparent as long as the control signal is high. However, gated latches are an important part of pulse mode design when they are combined with careful timing analysis of the control signal.

A gated SR-latch



As with combinational circuits we need to follow a three step process for sequential machines:

1. consider an idealized algebraic model of the network's operation to get techniques for the analysis of a given sequential machine: *machine* \rightarrow *function*
2. consider the deviation of the model from physical reality and devise ways of detecting when the model is inaccurate in disastrous way, i.e., model the deviation (Example: timing and combinational hazards)
3. reverse the analysis process to get a practical synthesis strategy: *function* \rightarrow *machine*

We will consider

- fundamental mode analysis applied to latches
- pulse mode analysis
- synthesis of pulse mode circuits

Analysis of SR Latch

In order to derive the characteristic function of the SR latch we must develop an analysis strategy for fundamental mode circuits whose memory is contained in combinational feedback.

To analyze we simply break the feedback loops until we get a combinational circuit. The smallest number of feedback loops that you must break to accomplish this is called the **inherent memory** of the circuit.

The points at which the loops are broken will have an old value on the upstream side and a new value on the downstream side.

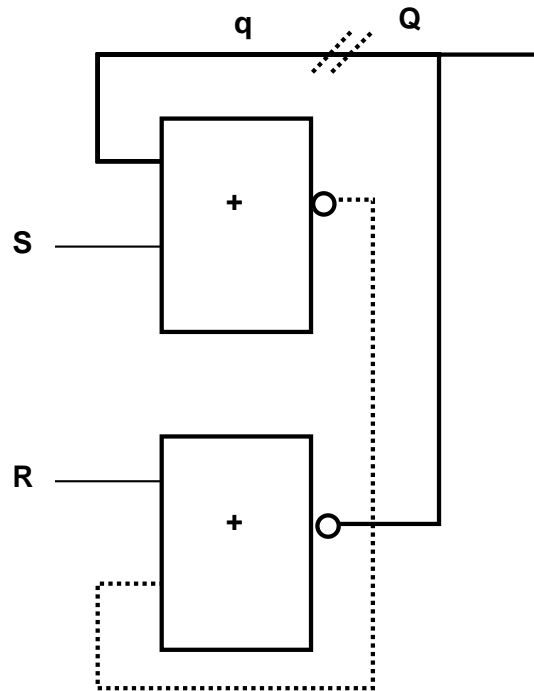
This can be used to define an excitation function of the new value in terms of the old. An excitation table can be defined and then using a characteristic function the transition table can be developed.

Notice that we have just broken the loop for analytical purposes the characteristic function is the identity, i.e.,

$$Y = E$$

where E is the excitation function and Y is the state variable.

NOR gate implementation of an SR-latch



There is a single feedback loop. The value of \bar{q} is always derived from the current values of q and S via the identity, $\bar{q} = q'S'$.

After breaking the feedback loop we have the following combinational relationship between the control inputs, the previous state and the next state:

$$\begin{aligned} Q &= [(q + S)' + R]' \\ &= (q + S)R' \\ &= qR' + SR' \end{aligned}$$

This yields the transition table below. All single input bit transitions are as we saw in our informal analysis of the latch. As with K-maps and hazards for combinational circuits we need to do some further work to alert ourselves to the cycling problem for a $(1,1) \rightarrow (0,0)$ transition.

The states and the associated values of q and \bar{q} are:

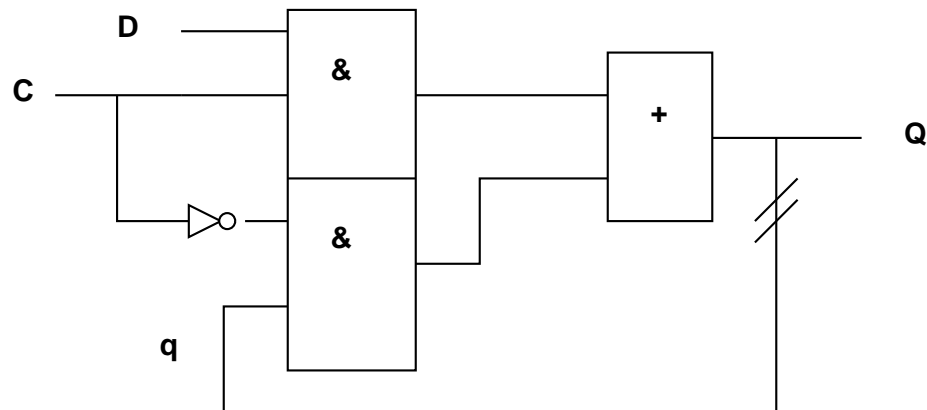
cell	condition	(q, \bar{q})
0	hold of a reset	$(0, 1)$
1	hold of a set	$(1, 0)$
2	reset	$(0, 1)$
5	set	$(1, 0)$
6	cleared	$(0, 0)$

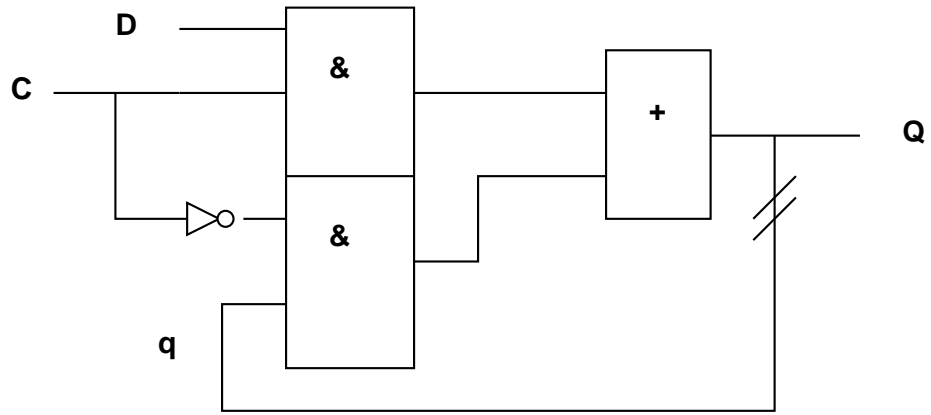
		SR			
		00	01	11	10
q	0	0 ⁰	0 ²	0 ⁶	1 ⁴
	1	1 ¹	0 ³	0 ⁷	1 ⁵

Hazards and latches

What about hazards in fundamental mode circuits which rely on combinational feedback to store information, e.g., latches.

Hazards inside the latches must be controlled since they can cause incorrect operation. Consider the gated D latch (the basic building block of many fundamental mode devices including the D flip flop).





$$Q = DC + C'q$$

		CD				Transition Table
		00	01	11	10	
q	0	0	0	1	0	
	1	1	1	1	0	

		CD				Stable 1-sets
		00	01	11	10	
q	0	0	0	1	0	
	1	1	1	1	0	

$C'q$
 DC

$(C+C')$ is the only unstable 0-set which corresponds to the static 1-hazard seen above

		CD				Stable 0-sets
		00	01	11	10	
q	0	0	0	1	0	
	1	1	1	1	0	

$C+q$
 $D+q$
 $C'+D$
20

No static 0-hazards

There are three cases of interest when the latch is transitioning between the two minterms shown.

$C : 0 \rightarrow 1$

$$\begin{aligned} D \bullet C + C' \bullet q &= Q \\ 1 \bullet 0 + 1 \bullet 1 &= 1 \\ 1 \bullet 0 + 0 \bullet 1 &= 0 \\ 1 \bullet 1 + 0 \bullet 0 &= 1 \\ 1 \bullet 1 + 0 \bullet 1 &= 1 \end{aligned}$$

Sequence of changes: C' drops; C rises and Q fed back; Q second feedback. The feedback is stopped due to $C' = 0$ and $D = 1$ holding. Static 1-hazard, settles to correct state. Incorrect information however, has been sent out from the latch.

$C : 1 \rightarrow 0$

$$\begin{aligned} D \bullet C + C' \bullet q &= Q \\ 1 \bullet 1 + 0 \bullet 1 &= 1 \\ 1 \bullet 0 + 0 \bullet 1 &= 0 \\ 1 \bullet 0 + 0 \bullet 0 &= 0 \\ 1 \bullet 0 + 1 \bullet 0 &= 0 \end{aligned}$$

Sequence of changes: C drops; Q fed back before C' affects AND gate; C' rises.

The affect of C' is not felt due to the fast feedback of the $Q = 0$ which makes the AND gate opaque. No hazard seen but, settles to incorrect 0 state and holds there.

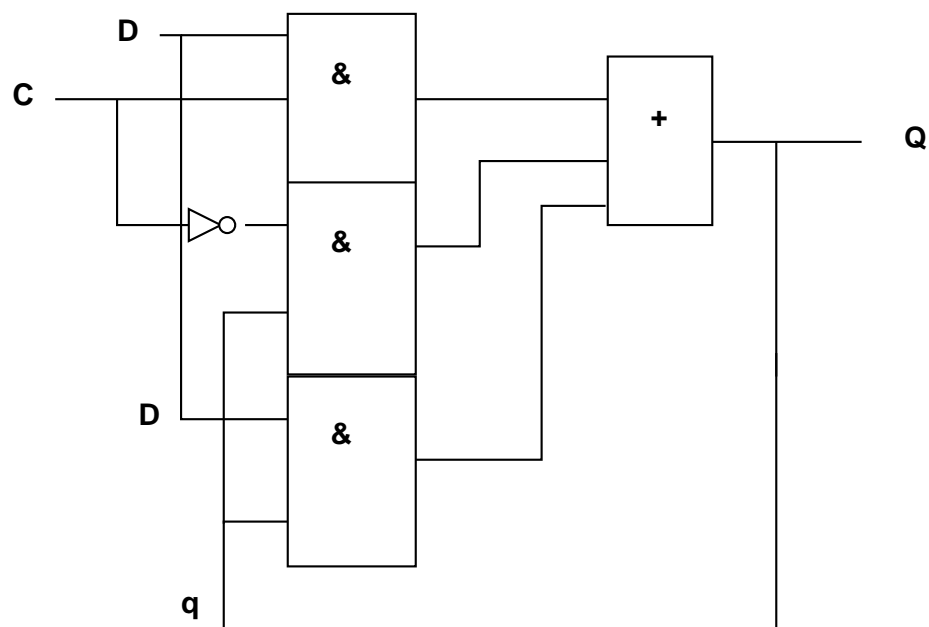
$C : 1 \rightarrow 0$

$$\begin{aligned} D \bullet C + C' \bullet q &= Q \\ 1 \bullet 1 + 0 \bullet 1 &= 1 \\ 1 \bullet 0 + 0 \bullet 1 &= 0 \\ 1 \bullet 0 + 1 \bullet 1 &= 1 \\ 1 \bullet 0 + 1 \bullet 0 &= 0 \\ 1 \bullet 0 + 1 \bullet 1 &= 1 \end{aligned}$$

Sequence of changes: C drops; C' rises before feedback of $Q = 0$ AND gate; $Q = 0$ feedback; $Q = 1$ feedback etc.

Static 1-hazard on a transparent feed back loop due to $C' = 1$ and $C = 0$. Oscillation occurs which in practice will settle nondeterministically.

The oscillation of the SR latch can also be explained as a static hazard on a transparent feedback loop.



Add another stable 1-set $D q$

This removes the static 1-hazard