

Latches: circuits that remember

Every gate so far forgets its inputs the moment they change. Today we close a loop, and two gates start to remember.

01 COLD OPEN 0-4 min

The lift button that stays lit

Press the call button of a lift and take your finger off. The light **stays on** until the lift arrives.

Every circuit we have built so far (DRL, DTL, TTL, CMOS gates) is **combinational**: its output depends only on its inputs *right now*. Remove your finger, and the output forgets.

Today's question: what is the smallest circuit that remembers one bit?

Feedback: its output must be one of its own inputs. Two inverters in a loop are enough to *hold* a bit; two NOR gates in a loop can hold it *and* be told what to hold. That loop is a **latch**.



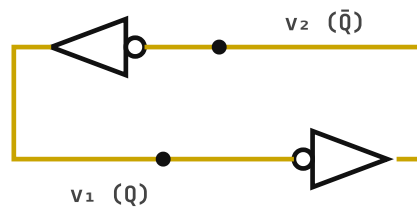
POP CULTURE

Every bit of cache memory in your phone's processor is a 6-transistor cell: two cross-coupled CMOS inverters plus two access switches. Billions of today's latch, on one chip.

02 CONCEPT 4-8 min

Feedback: the one idea behind all memory

- **Combinational:** output = $f(\text{present inputs})$. **Sequential:** output = $f(\text{present inputs}, \text{stored state})$.
- Two inverters in a ring: if v_1 is high, v_2 is low, which keeps v_1 high. Two self-consistent states, **$Q = 0$** and **$Q = 1$** : a **bistable** circuit.
- There is a third consistent point, $v_1 = v_2 = V_M$, but it is **unstable**: any imbalance is amplified round the loop (gain $A > 1$) until the circuit falls into one of the two stable states.
- A loop of inverters can hold a bit but has no way to be *told* a new one. Replace the inverters by NOR (or NAND) gates and the spare inputs become **Set** and **Reset**.



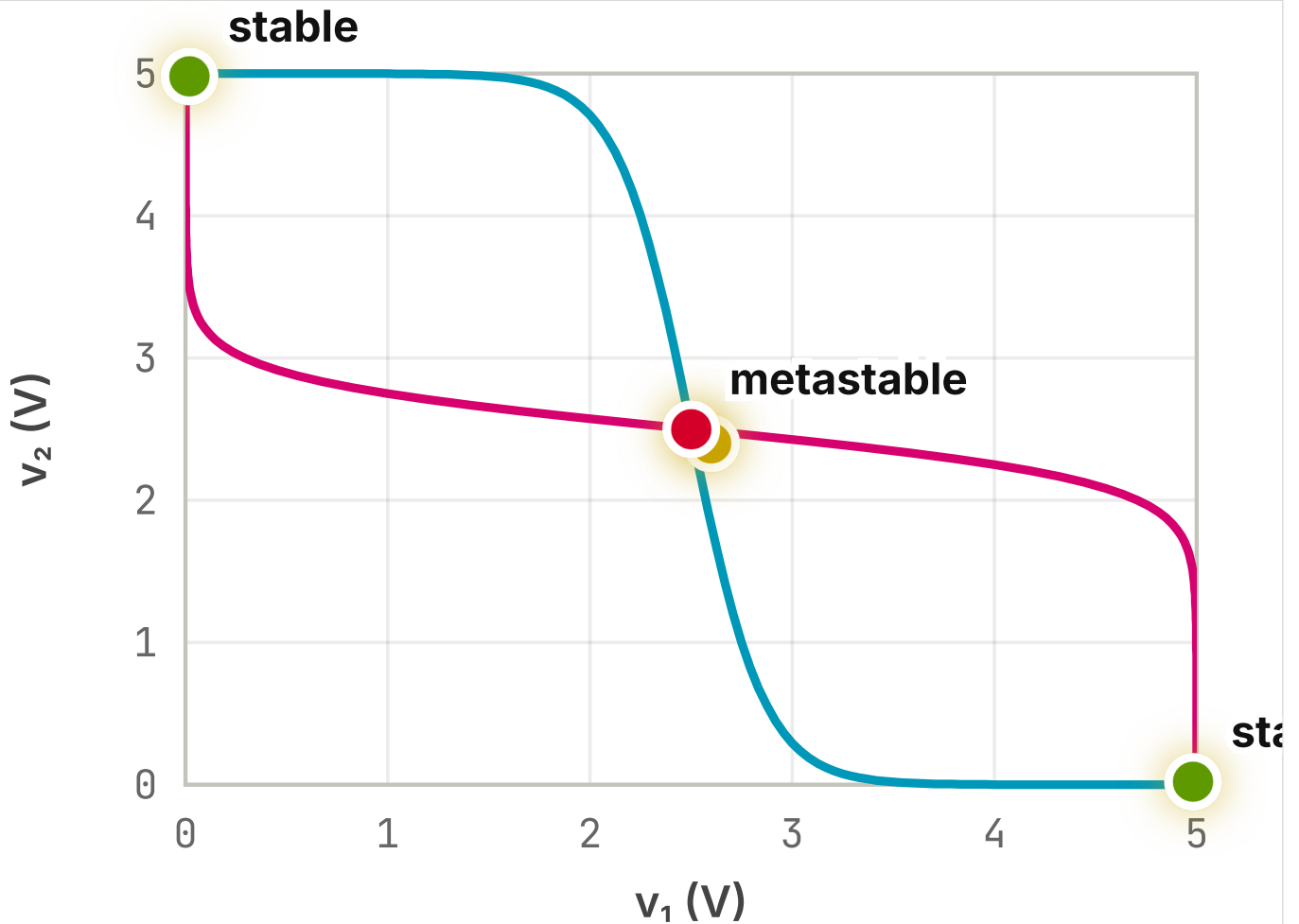
PRO TIP

Gain is the whole story: the loop is bistable only if the inverters amplify ($|\text{slope}| > 1$ near V_M).
Two resistors in a ring remember nothing.

Live lab: the hilltop and the two valleys

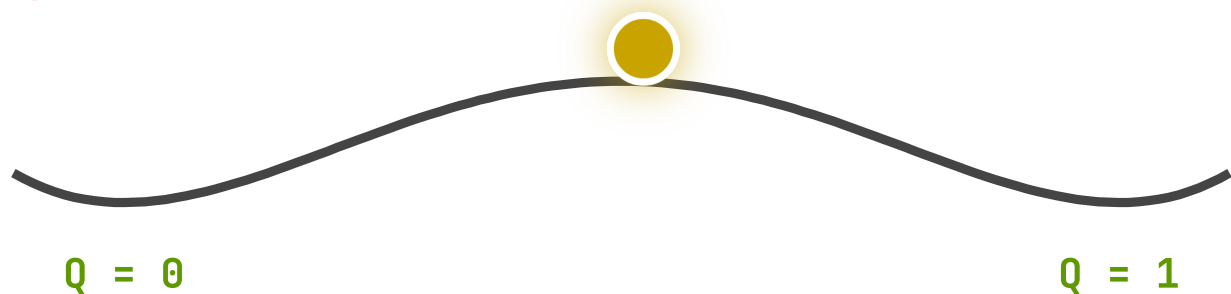
Two inverters in a loop: two valleys and a hilltop

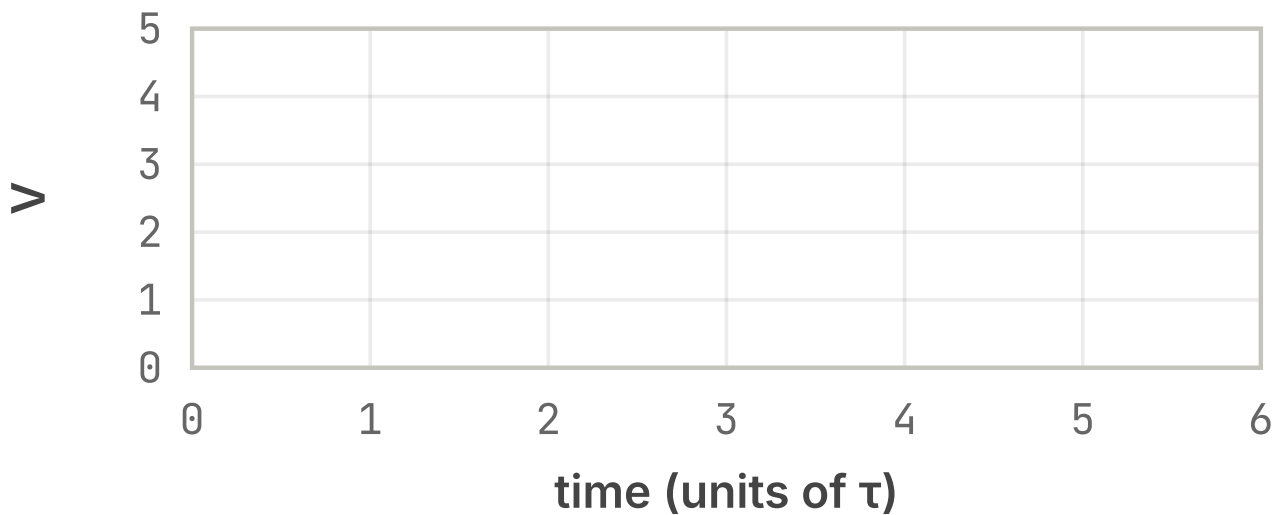
Release the loop slightly off balance. The smaller the imbalance, the longer it hesitates on the hilltop.



— $v_2 = f(v_1)$ — $v_1 = f(v_2)$

top: $v_1 = v_2 = 2.5$ V





initial imbalance $v_1 - v_2 = 0.200 \text{ V}$

☐ thermal noise

LOOP GAIN A
6.9

SETTLES IN
—

THEORY
0.50 τ

04 WORKED EXAMPLE 13-20 min

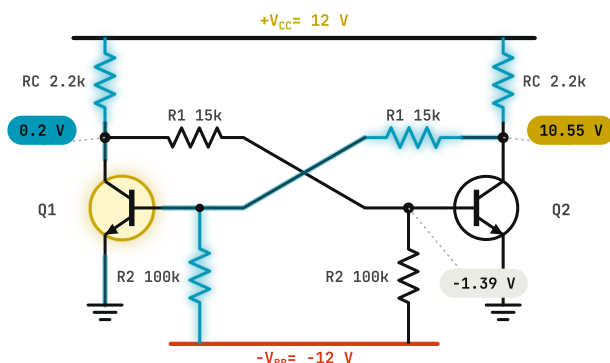
Worked example: a transistor latch

GIVEN

The fixed-bias binary in the figure. $V_{CC} = 12 \text{ V}$, $V_{BB} = 12 \text{ V}$, $R_C = 2.2 \text{ k}\Omega$, $R_1 = 15 \text{ k}\Omega$ (collector to the other base), $R_2 = 100 \text{ k}\Omega$ (base to $-V_{BB}$), $h_{FE(\min)} = 30$, $V_{BE(\text{sat})} = 0.7 \text{ V}$, $V_{CE(\text{sat})} = 0.2 \text{ V}$.

FIND

Show that the state Q1 ON, Q2 OFF is self-consistent. Find V_{B2} , V_{C2} , I_{B1} , I_{C1} and check that Q1 saturates.



Method: assume a state, then verify every transistor (Millman & Taub, the fixed-bias binary)

SOLUTION

1. **Assume Q1 ON (saturated), Q2 OFF** and check both. Q2's base sits on a divider from $V_{C1} = 0.2$ V to $-V_{BB}$:

$$V_{B2} = 0.2 + (-12 - 0.2) \frac{15}{15 + 100} = 0.2 - 1.59 = -1.39 \text{ V}$$

$V_{BE2} = -1.39$ V: reverse biased, Q2 is firmly OFF ✓.

2. Q2 is off, so R_C on its side feeds only R_1 into Q1's base ($V_{B1} = 0.7$ V):

$$V_{C2} = \frac{V_{CC}R_1 + V_{BE}R_C}{R_1 + R_C} = \frac{12(15) + 0.7(2.2)}{17.2} = 10.55 \text{ V}$$

3. Base current of Q1 = current in through R_1 minus current out through R_2 :

$$I_{B1} = \frac{10.55 - 0.7}{15} - \frac{0.7 + 12}{100} = 0.657 - 0.127 = 0.530 \text{ mA}$$

4. Collector current of Q1 = R_C current minus what leaks into Q2's divider:

$$I_{C1} = \frac{12 - 0.2}{2.2} - \frac{0.2 + 12}{115} = 5.364 - 0.106 = 5.26 \text{ mA}$$

5. **Saturation check:**

$$I_{B(min)} = \frac{I_{C1}}{h_{FE}} = \frac{5.26}{30} = 0.175 \text{ mA} < 0.530 \text{ mA} \checkmark$$

Overdrive $\approx 3\times$: Q1 stays saturated even with weak transistors. By symmetry, the other state (Q1 OFF, Q2 ON) is just as stable.

ANSWER

$V_{B2} = -1.39$ V (Q2 off), $V_{C2} = 10.55$ V, $I_{B1} = 0.53$ mA, $I_{C1} = 5.26$ mA, $I_{B(min)} = 0.175$ mA: Q1 saturated. Outputs: $V_{C1} = 0.2$ V (0), $V_{C2} = 10.55$ V (1).

05 **CONCEPT** 20-25 min

The SR latch: give the loop two handles

S	R	Q NEXT	$\bar{Q}$ NEXT	
0	0	Q	$\bar{Q}$	hold (memory)
0	1	0	1	reset
1	0	1	0	set
1	1	0	0	forbidden

$$Q^+ = S + \bar{R}Q \quad (SR = 0)$$

Two NOR gates, each output feeding the other's spare input. A 1 on any NOR input forces its output to 0: that is how S and R take control.

ACTIVE-LOW TWIN: THE NAND $\bar{S}\bar{R}$ LATCH

A 0 on a NAND input forces its output to 1. So the inputs are active-low: $\bar{S} = 0$ sets, $\bar{R} = 0$ resets, $\bar{S} = \bar{R} = 1$ holds, and $\bar{S} = \bar{R} = 0$ is forbidden ($Q = \bar{Q} = 1$).



GOTCHA

"Forbidden" is not about damage. With $S = R = 1$, Q and $\bar{Q}$ stop being complements, and if both inputs are released together the next state is a race nobody can predict.



PRO TIP

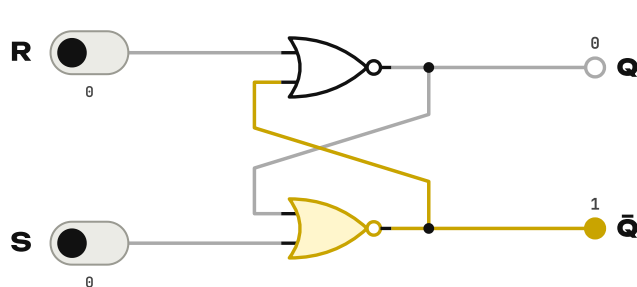
A set or reset pulse must last at least $2 t_{pd}$: one gate delay for the first gate to change, one for the second to answer and close the loop.

06 **LIVE LAB** 25-31 min

Live lab: set, reset, hold, and the race

Latches: two gates that remember

Flip the inputs and watch the change travel round the loop, one gate delay at a time. Then try the forbidden input and release it.



S	R	Q next	
0	0	Q	hold
0	1	0	reset
1	0	1	set
1	1	0*	forbidden

OUTPUTS

Q = 0, $\bar{Q}$ = 1



07 **WORKED EXAMPLE** 31-36 min

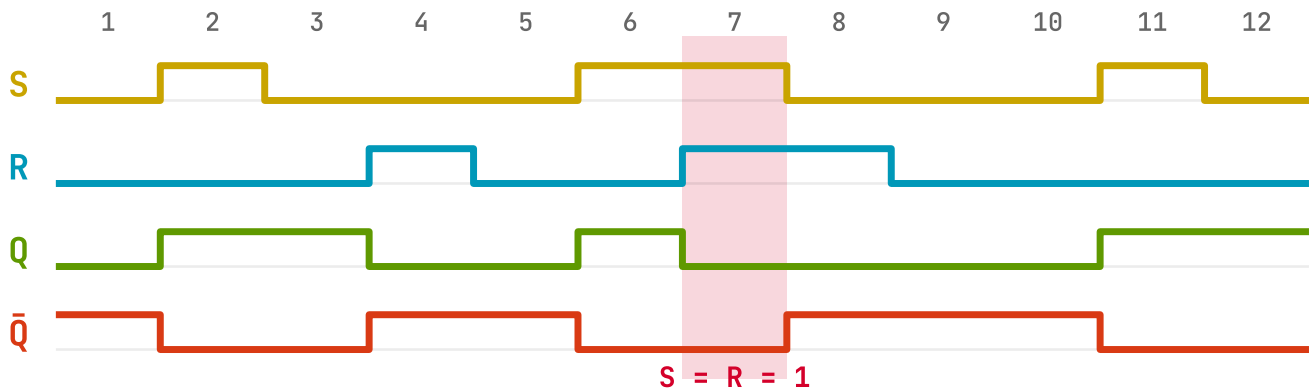
Worked example: reading a NOR latch timing diagram

GIVEN

A NOR SR latch, initially $Q = 0$. Gate delays are neglected (each slot is long compared with t_{pd}). The S and R waveforms are in the figure.

FIND

Q and $\bar{Q}$ for all 12 slots. Mark any slot where the latch is not in a valid state.



Method: hold / set / reset / forbidden, one slot at a time (Mano, SR latch)

SOLUTION

1. Start from $Q = 0$. Slot 1: $S = R = 0 \rightarrow$ **hold** 0. Slot 2: $S = 1 \rightarrow$ **set**, $Q = 1$. Slot 3: hold 1.
2. Slot 4: $R = 1 \rightarrow$ **reset**, $Q = 0$. Slot 5: hold 0. Slot 6: $S = 1 \rightarrow$ set, $Q = 1$.
3. Slot 7: **$S = R = 1$** : both NOR outputs are forced to 0, so $Q = \bar{Q} = 0$. Not a valid memory state.
| Q and $\bar{Q}$ are no longer complements: anything reading $\bar{Q}$ gets the wrong answer.
4. Slot 8: S drops first, R stays 1 $\rightarrow$ a clean **reset**: $Q = 0$, $\bar{Q} = 1$. (Had both dropped together, the result would be unpredictable.)
5. Slots 9–10: hold 0. Slot 11: set, $Q = 1$. Slot 12: hold 1.

Q : 011001000011 $\bar{Q}$: 100110011100

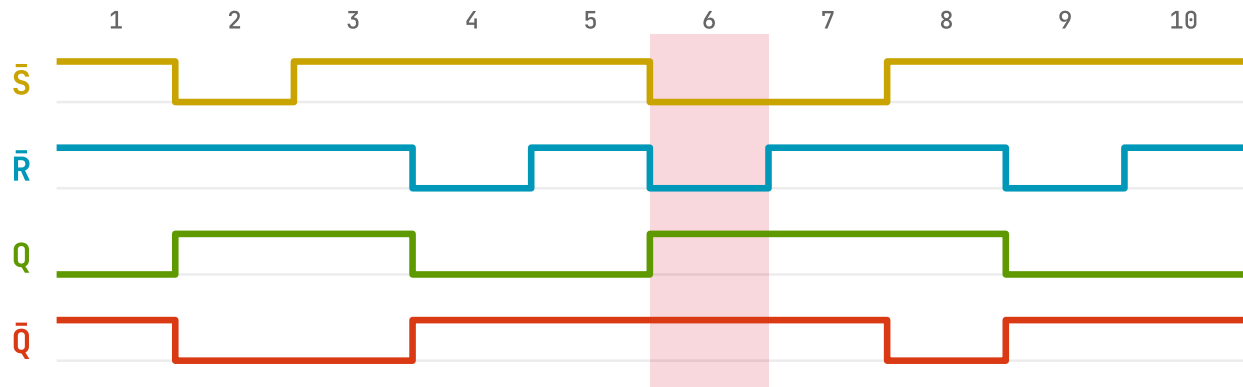
ANSWER

$Q = 011001000011$, $\bar{Q} = 100110011100$. Slot 7 ($S = R = 1$) is invalid: $Q = \bar{Q} = 0$.

Your turn • 1: the active-low latch

1 EXAM CQ

A NAND $\bar{S}\bar{R}$ latch starts with $Q = 0$. Draw Q and $\bar{Q}$.



SOLUTION

- Slot 1 hold 0. Slot 2 $\bar{S} = 0$: set, $Q = 1$. Slot 3 hold. Slot 4 $\bar{R} = 0$: reset, $Q = 0$. Slot 5 hold.
- Slot 6: $\bar{S} = \bar{R} = 0 \rightarrow Q = \bar{Q} = 1$ (invalid).
- Slot 7: $\bar{R}$ returns to 1 while $\bar{S}$ stays 0 $\rightarrow$ clean **set**, $Q = 1$. Slot 8 hold. Slot 9 reset. Slot 10 hold.

Q : 0110011100 $\bar{Q}$: 1001110011

ANSWER

$Q = 0110011100$; $\bar{Q} = 1001110011$. Slot 6 is the forbidden state (both 1); slot 7 is a clean set because $\bar{S}$ stays 0.

Critical thinking • why is

S = R = 1 really forbidden?

THINK DEEPER

S = R = 1 just gives $Q = \bar{Q} = 0$, which is a perfectly definite output. So what exactly goes wrong?

The state itself is harmless; **leaving it** is the problem. Released together, the two gates race each other, the latch may oscillate for a while and end up in either state: **the next state is not determined by the inputs**, which is the one thing a logic circuit must guarantee.

A STRONG ANSWER MENTIONS

- $\bar{Q}$ is supposed to be the complement of Q: a circuit using both outputs now sees contradictory information.
- The real problem is the exit: if S and R fall together, both NORs see 0, 0 and both outputs rise together, then both fall...
- The loop is balanced, like the hilltop in the first lab. Tiny gate mismatches or noise decide, at an unpredictable time.

THINK DEEPER

In the lab the race always ended. Could a real latch stay undecided for, say, a whole microsecond?

Yes, but with vanishing probability. A balanced latch leaves the hilltop at a rate set by its loop gain; the chance of still being undecided after time t falls like $e^{-t/\tau}$. Designers cannot remove metastability, only make it rare enough.

A STRONG ANSWER MENTIONS

- Yes, in principle: the balanced point is an equilibrium, just an unstable one.
- The probability of still being undecided falls exponentially with time: $P \propto e^{-t/\tau}$, with τ a few tens of picoseconds in modern CMOS.
- So "a microsecond" is astronomically unlikely, but never impossible. Next class this becomes the flip-flop's MTBF.

10 **CONCEPT** 43–47 min

Gated latches: listen only when enabled

- Put two NAND gates in front of an $\bar{S}\bar{R}$ latch, both driven by an **enable** C (often called the clock). C = 0: both front gates output 1, the latch **holds**. C = 1: S and R pass through.
- The **gated SR** latch still has a forbidden input (C = 1, S = R = 1).
- The **D latch**: feed S = D and R = $\bar{D}$ through an inverter. S and R can never both be 1, so the forbidden input **disappears**.

$$\text{D latch: } Q^+ = D \ (C = 1), \quad Q^+ = Q \ (C = 0)$$

TRANSPARENT

While C = 1 the D latch is a wire: Q follows D (one or two gate delays later). When C falls, Q **freezes** at the last value of D. That is what "latching" means.



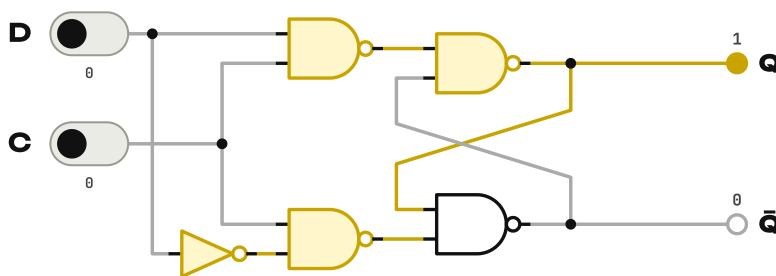
PRO TIP

The 74LS75 holds four D latches; two enables each control a pair. Processors used them to hold an address steady on the bus while the data moved.

11 **LIVE LAB** 47–52 min

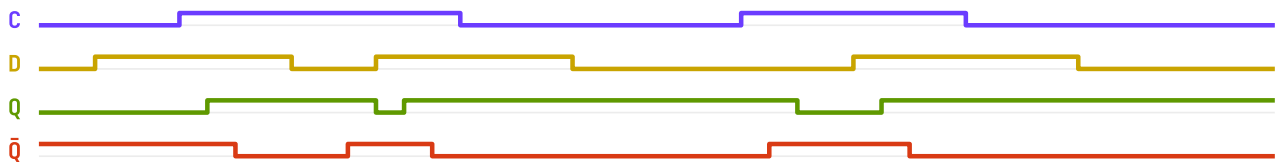
Live lab: the transparent D latch

Latches: two gates that remember Flip the inputs and watch the change travel round the loop, one gate delay at a time. Then try the forbidden input and release it.



C	D	Q next	
0	x	Q	hold (latched)
1	0	0	transparent
1	1	1	transparent

OUTPUTS
Q = 1, $\bar{Q}$ = 0



12 **WORKED EXAMPLE** 52–57 min

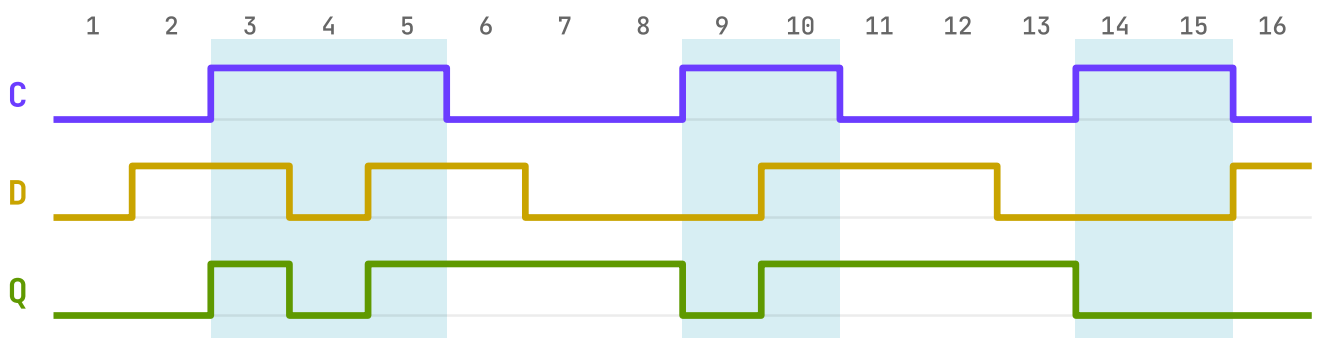
Worked example: D latch timing

GIVEN

A gated D latch, initially $Q = 0$. C and D as shown (gate delays neglected).

FIND

Q for all 16 slots.



Method: $C = 1 \rightarrow$ copy D; $C = 0 \rightarrow$ hold

SOLUTION

1. **C = 0** (slots 1–2): latched. Q holds its initial 0 whatever D does.
2. **C = 1** (slots 3–5): transparent. Q copies D: 1, 0, 1. The short 0 on D at slot 4 goes straight through.
This is the weakness for next class: while C = 1 every change on D, glitches included, reaches Q.
3. C falls after slot 5: Q **freezes** at the value D had at that moment (1) and holds it through slots 6–8.
4. Slots 9–10 (C = 1): Q = 0, then 1. Slots 11–13 (C = 0): holds 1. Slots 14–15 (C = 1): Q = 0, 0. Slot 16 (C = 0): holds 0.

Q : 0010111101111000

ANSWER

Q = 0010111101111000. Q follows D while C = 1 and holds the last value while C = 0.

13 YOUR TURN 57–63 min

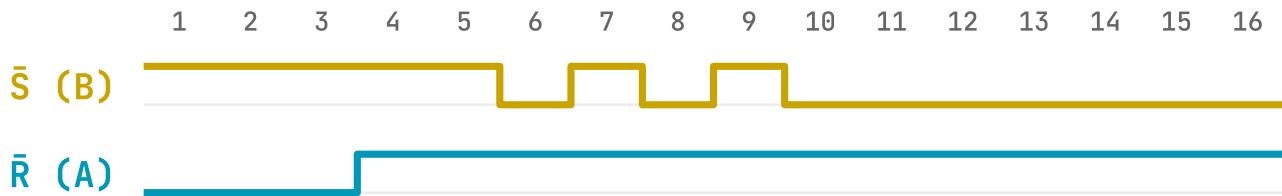
Your turn • 2: debounce

a switch with a latch

2

DESIGN

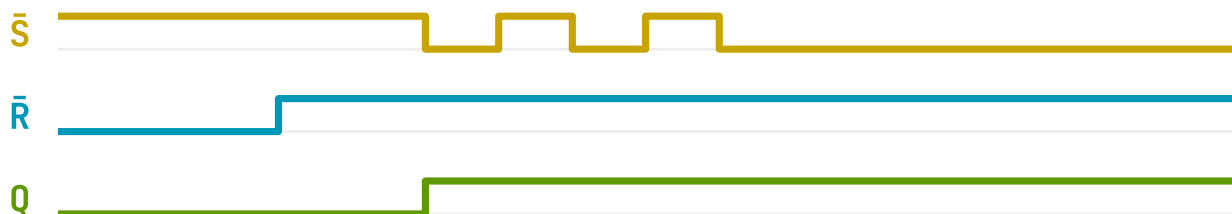
A push button is an SPDT switch: contact A (normally closed) drives $\bar{R}$, contact B drives $\bar{S}$, both lines pulled up to 5 V. When pressed, the arm leaves A and **bounces** on B for a few ms.



(a) How many presses would a counter driven by the B line count? (b) Draw Q of an $\bar{S}\bar{R}$ latch. (c) Why does the latch remove the bounce? (d) Would it work with a two-terminal (SPST) button?

SOLUTION

- (a)** The B line alone (a pull-up to 5 V, pulled to 0 by the contact) falls three times: at slots 6, 8 and 10. A counter clocked by it would count **3 presses**.
- (b)** Slots 1–3: $\bar{R} = 0 \rightarrow$ reset, $Q = 0$. Slots 4–5: both 1 $\rightarrow$ hold 0. Slot 6: $\bar{S} = 0 \rightarrow$ set, $Q = 1$. Slots 7–16: $\bar{S}$ is 0 or 1, $\bar{R}$ is 1: set or hold, never reset $\rightarrow$ **Q stays 1**.
- (c)** A bouncing contact touches B or nothing, *never* A: it can only ever set again or hold. One clean edge per press.
- (d)** With an SPST switch there is no second contact to produce the reset, so the latch could not tell "released" from "bouncing": use an RC filter and a Schmitt trigger instead (Week 12).



ANSWER

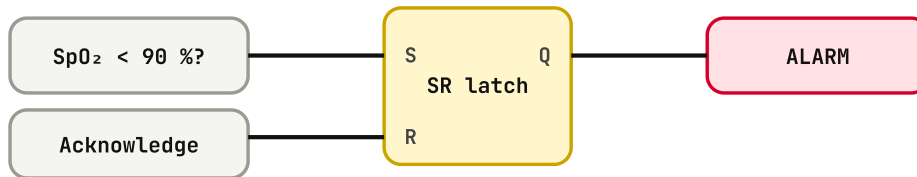
Raw line: 3 falling edges (3 counts). Latch output: $Q = 0000011111111111$, a single clean rising edge at slot 6. Works because the bouncing arm can never reach A.

14 REAL WORLD 63–68 min

Real world: alarms that must not forget

A pulse oximeter on a sleeping patient sees SpO_2 fall to 82 % for 20 seconds at 3 AM, then recover. The nurse was in another ward.

Medical alarm standards (IEC 60601-1-8) define a **latching alarm**: once triggered it keeps sounding *after the cause has gone*, until a person acknowledges it. In hardware, that is an SR latch: the comparator output drives S, the **Acknowledge** button drives R.



BME CORNER

The same pattern appears in defibrillators ("shock advised" stays lit), infusion pumps (occlusion alarms) and ventilators. Memory turns a brief event into something a human can notice.

THINK DEEPER

What should happen if the SpO_2 is still low when the nurse presses Acknowledge?

Both inputs are active at once. A plain SR latch has no defined answer, so the designer adds priority logic. For safety, **set wins**: the alarm cannot be silenced while the condition persists.

A STRONG ANSWER MENTIONS

- S and R are both active: the forbidden input of an SR latch.
- Here the designer must choose: acknowledge wins (R priority) or alarm wins (S priority).
- Safety says the alarm wins: $Q^* = S + \bar{R}Q$ is "set-priority", exactly what the NOR latch's forbidden case is replaced by in a safe design.

15 **POP QUIZ** 68-74 min

Pop quiz

1. **A NOR SR latch has $S = 0$ and $R = 0$. Its output is...**

- A. always 0
- B. always 1
- C. whatever it was before ✓**
- D. undefined

With both inputs 0, each NOR acts as an inverter of the other output: the loop holds its state. That is the memory.

2. **For a NAND $\bar{S}\bar{R}$ latch, the input that must be avoided is...**

- A. $\bar{S} = \bar{R} = 1$
- B. $\bar{S} = \bar{R} = 0$ ✓**
- C. $\bar{S} = 0, \bar{R} = 1$
- D. $\bar{S} = 1, \bar{R} = 0$

A 0 on any NAND input forces its output to 1, so $\bar{S} = \bar{R} = 0$ gives $Q = \bar{Q} = 1$. Releasing both together is a race.

3. **A gated D latch has $C = 1$. Its output Q ...**

- A. holds
- B. follows D ✓**
- C. toggles
- D. is forced to 0

$C = 1$ makes it transparent: $Q = D$ (after a short delay). When C goes to 0, Q keeps the last value of D .

4. **Why does the D latch have no forbidden input?**

- A. It uses NOR gates
- B. Its internal S and R are D and $\bar{D}$, never both 1 ✓**
- C. Its clock is faster
- D. It has no feedback

The inverter makes $S = D$ and $R = \bar{D}$: they can never be active together.

5. **Two identical cross-coupled inverters start exactly at $v_1 = v_2 = V_M$. They...**

- A. go to $Q = 1$ immediately
- B. go to $Q = 0$ immediately
- C. can stay balanced for an unbounded time (metastable) ✓**
- D. oscillate forever at a fixed frequency

V_M is an equilibrium, but an unstable one: any tiny imbalance grows as $e^{(A-1)t/\tau}$, but with zero imbalance nothing pushes it. In practice noise decides, at a random time.

16 **RECAP** 74-80 min

Three things to remember

O1

Feedback with gain makes a bistable: two stable states and one metastable balance point. A transistor latch: assume one ON, one OFF, then verify.

O2

SR latch (NOR, active-high) and **$\bar{S}\bar{R}$** (NAND, active-low): hold, set, reset, forbidden. $Q^+ = S + \bar{R}Q$, $SR = 0$.

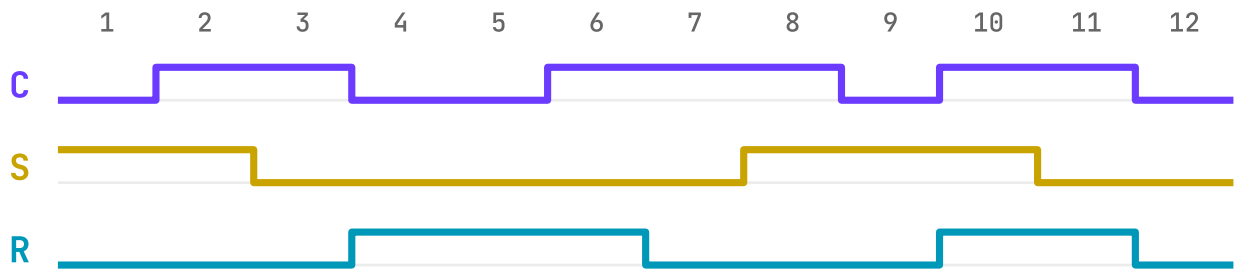
O3

Gated D latch: $C = 1$ transparent (Q follows D), $C = 0$ holds. No forbidden input, but every glitch on D gets through while $C = 1$.

HW

BEFORE NEXT CLASS

A gated SR latch (built from NAND gates) starts with $Q = 0$. Draw Q .



Class Test 1 is in **Week 5, Class 1** (Weeks 1–4, including next class's flip-flops). Assignment 1 is due the same day.

SOLUTION

- Slot 1 hold 0. Slot 2 set $\rightarrow$ 1. Slot 3 ($C = 1, S = R = 0$) hold 1. Slots 4–5 ($C = 0$) hold 1.
- Slot 6 reset $\rightarrow$ 0. Slot 7 hold 0. Slot 8 set $\rightarrow$ 1. Slot 9 ($C = 0$) hold 1.
- Slot 10: $C = 1, S = R = 1$: **forbidden**, $Q = \bar{Q} = 1$ (NAND-based). Slot 11: R stays 1 $\rightarrow$ reset, $Q = 0$. Slot 12 hold 0.

ANSWER

$Q = 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ * \ 0\ 0$ (slot 10 invalid: $Q = \bar{Q} = 1$).

S. M. Sakeef Sani

Adjunct Lecturer, CSE · Northern University Bangladesh

smsakeefsani.hbitcorp.com