

# Flip-flops: edge-triggered memory

A latch listens for as long as the clock is high. A flip-flop listens for one instant: the edge. That difference is what makes counters, registers and processors possible.

01 **COLD OPEN** 0-4 min

## The counter that counts wrong

Take last class's D latch, connect  $\bar{Q}$  **back to D**, and clock it. It should toggle once per clock: a divide-by-two, the first stage of every counter.

Instead, while the clock is high the new  $\bar{Q}$  races round the loop to D, through the transparent latch, back to  $\bar{Q}$ ... It toggles **again and again**, as many times as fit in the pulse.

### THINK DEEPER

How many times does the latch toggle in one clock pulse? What does it depend on?

About  $(\text{pulse width})/t_{pd}$  times: a 50 ns pulse and a 20 ns latch gives two or three toggles, so the final state is a coin toss. We need a memory that samples **once per clock**, at an instant.

### A STRONG ANSWER MENTIONS

- Each trip round the loop takes  $t_{pd}$  ( $D \rightarrow Q$ ).
- Number of toggles  $\approx \text{pulse width} \div t_{pd}$ : e.g.  $50 \text{ ns} \div 20 \text{ ns} = 2$  or  $3$ .
- It depends on temperature, supply voltage and the chip: the final state is unpredictable.

02 **LIVE LAB** 4-8 min

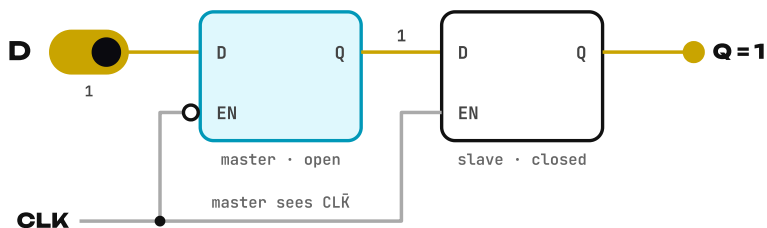
## Live lab: latch vs flip-



## Live lab: inside the master-slave D flip-flop

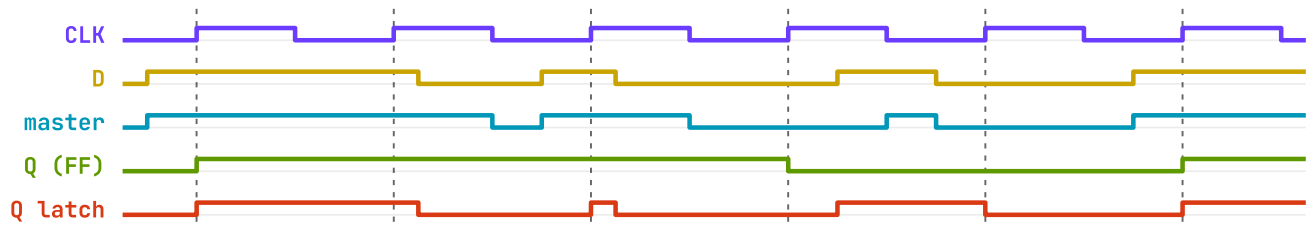
**Flip-flops: change only at the clock edge** The dashed lines mark rising clock edges. A latch follows its input for the whole time the clock is high; a flip-flop looks only at the edge.

☒ auto clock



$Q^+ = D$  (sampled at the rising edge)

CLK = 0: master follows D, slave holds.  
CLK = 1: master holds, slave shows it.  
Toggle D while CLK is high and compare the two Q rows.



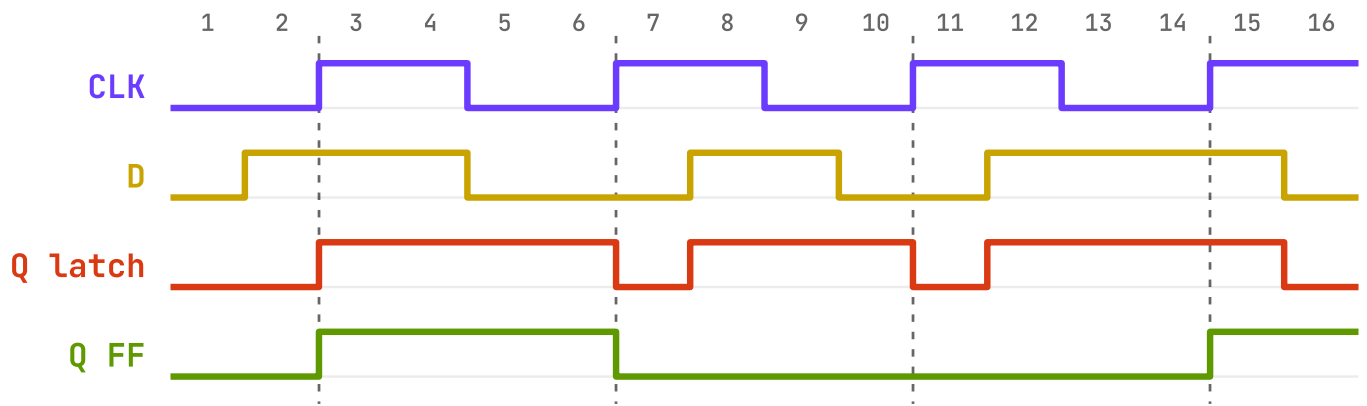
## Worked example: same D, latch vs flip-flop

**GIVEN**

CLK and D as shown. A D latch (enabled by CLK = 1) and a positive-edge D flip-flop both start with Q = 0. Delays neglected.

**FIND**

Q of the latch and Q of the flip-flop for all 16 slots.



Method: edges first, then the table (Mano: latch vs flip-flop)

### SOLUTION

1. Mark the rising edges first (dashed lines): slots 3, 7, 11 and 15. The flip-flop looks at D **only** there.
2. **Flip-flop:** edge at 3: D = 1 → Q = 1 until the next edge. Edge at 7: D = 0 → Q = 0. Edge at 11: D = 0 → still 0. Edge at 15: D = 1 → Q = 1.

$$Q_{FF} : 0011110000000011$$

3. **Latch** (transparent while CLK = 1, slots 3-4, 7-8, 11-12, 15-16): copy D inside those windows, hold outside.

$$Q_L : 0011110111011110$$

4. They disagree in slots 8-10, 12-14 and 16: each time **D changed while the clock was high**. The latch let the change through; the flip-flop waited for the next edge.

One sample per clock period, taken at a known instant: that is what makes synchronous design predictable.

### ANSWER

$Q_{FF} = 0011110000000011$ .  $Q_{latch} = 0011110111011110$ . They differ wherever D changed while CLK = 1.

06 CONCEPT 24-29 min

## SR, JK, D, T: four flip-flops, four equations

SR

|    |   |
|----|---|
| 00 | Q |
| 01 | 0 |
| 10 | 1 |
| 11 | ? |

$$Q^+ = S + \bar{R}Q$$

JK

|    |           |
|----|-----------|
| 00 | Q         |
| 01 | 0         |
| 10 | 1         |
| 11 | $\bar{Q}$ |

$$Q^+ = J\bar{Q} + \bar{K}Q$$

| D |   |   |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 1 |

$$Q^+ = D$$

| T |   |           |
|---|---|-----------|
| 0 | 0 | Q         |
| 1 | 1 | $\bar{Q}$ |

$$Q^+ = T \oplus Q$$

Characteristic tables (inputs → next state). The **JK** fixes SR's forbidden row by making it *toggle*; the **T** is a JK with  $J = K = T$ ; the **D** is the simplest and the one used inside almost every chip.



#### PRO TIP

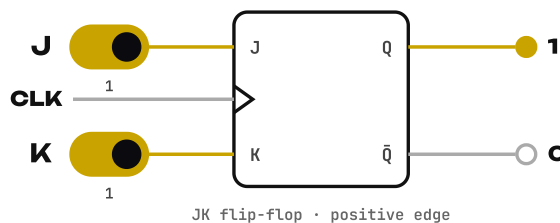
The JK toggle row is only safe because the flip-flop is edge-triggered. A JK *latch* with  $J = K = 1$  races around exactly like the opening demo.

07 **LIVE LAB** 29–33 min

## Live lab: JK and T

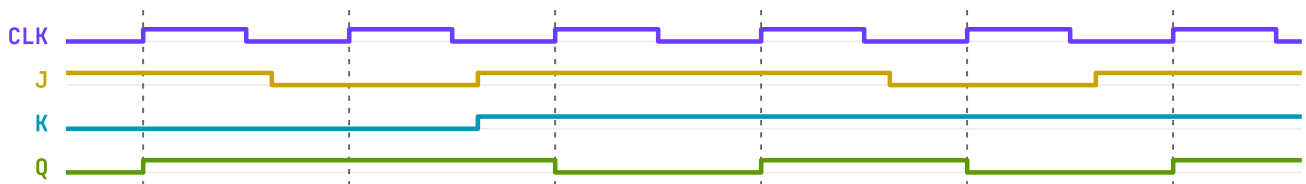
**Flip-flops: change only at the clock edge** The dashed lines mark rising clock edges. A latch follows its input for the whole time the clock is high; a flip-flop looks only at the edge.

☒ auto clock



| J | K | Q next    |        |
|---|---|-----------|--------|
| 0 | 0 | Q         | hold   |
| 0 | 1 | 0         | reset  |
| 1 | 0 | 1         | set    |
| 1 | 1 | $\bar{Q}$ | toggle |

$$Q^+ = J \cdot \bar{Q} + \bar{K} \cdot Q$$



08 **WORKED EXAMPLE** 33–38 min

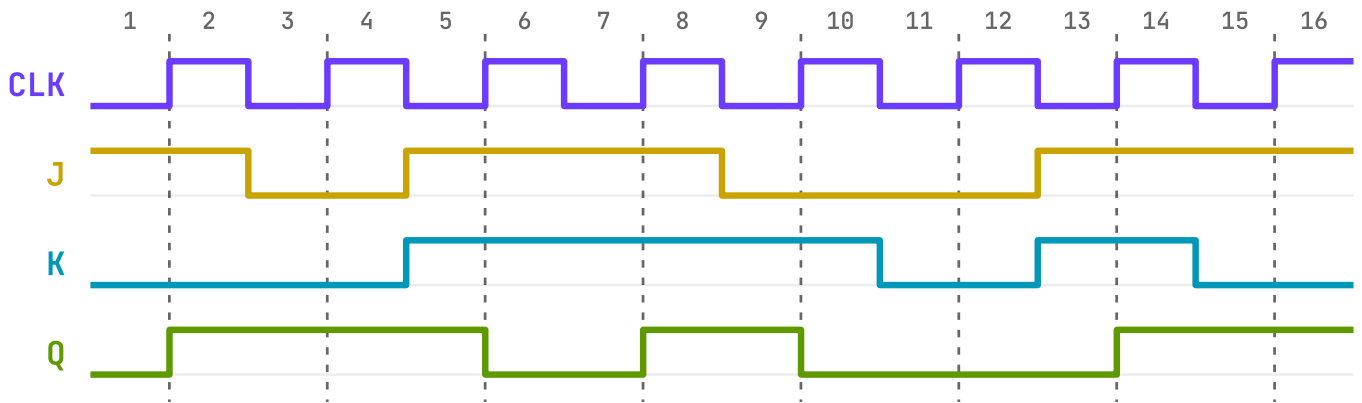
## Worked example: a JK timing diagram

GIVEN

A positive-edge JK flip-flop, initially  $Q = 0$ . J and K change only while CLK is low (well away from the edges).

FIND

Q for all 16 slots.



Method: read (J, K) at each rising edge only

#### SOLUTION

1. Rising edges at slots 2, 4, 6, ..., 16. At each edge read (J, K) and apply the table; between edges Q holds.
2. Edge 1 (1, 0): **set** → 1. Edge 2 (0, 0): hold 1. Edge 3 (1, 1): **toggle** → 0. Edge 4 (1, 1): toggle → 1.
3. Edge 5 (0, 1): **reset** → 0. Edge 6 (0, 0): hold 0. Edge 7 (1, 1): toggle → 1. Edge 8 (1, 0): set → 1 (already 1).

Q : 0111100110000111

4. Check one edge with the equation, e.g. edge 4 (Q = 0, J = K = 1):

$$Q^+ = J\bar{Q} + \bar{K}Q = 1 \cdot 1 + 0 \cdot 0 = 1 \checkmark$$

#### ANSWER

Q = 0111100110000111 (set, hold, toggle, toggle, reset, hold, toggle, set).

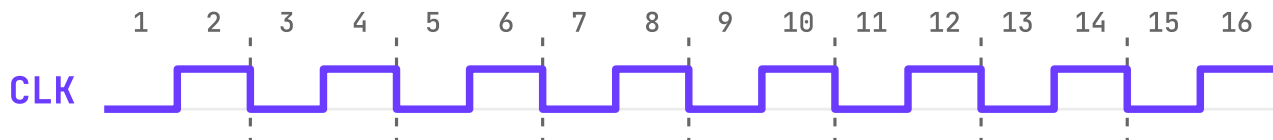
09 YOUR TURN 38-43 min

## Your turn • 1: two T flip-

# flops make a counter

## 1 EXAM CQ

Two **negative-edge** T flip-flops, both with  $T = 1$ , start at 0. CLK (10 MHz) drives the first; the first's output  $Q_0$  is the clock of the second.



Draw  $Q_0$  and  $Q_1$ . What do  $Q_1Q_0$  do as a 2-bit number? What are their frequencies?

### SOLUTION

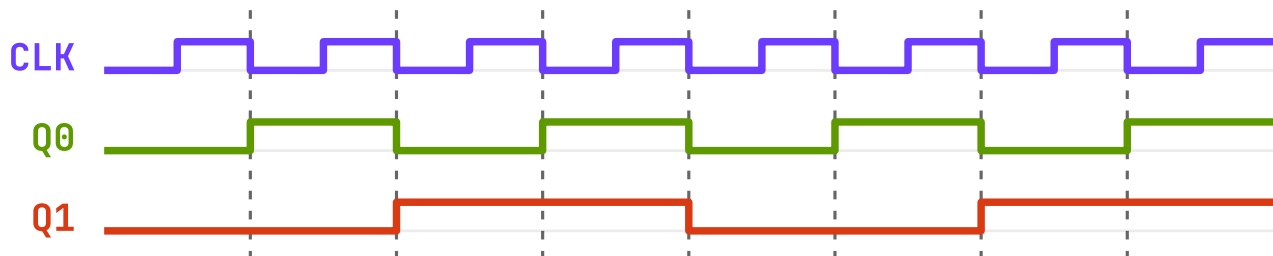
1. CLK falls at the start of slots 3, 5, 7, ..., 15.  $Q_0$  toggles there:

$$Q_0 : 0011001100110011$$

2.  $Q_0$  falls ( $1 \rightarrow 0$ ) at slots 5, 9 and 13.  $Q_1$  toggles there:

$$Q_1 : 0000111100001111$$

3. Read  $Q_1Q_0$  as a binary number: 00, 01, 10, 11, 00, 01, 10, 11: it **counts up** modulo 4.
4.  $f(Q_0) = 10/2 = 5 \text{ MHz}$ ,  $f(Q_1) = 10/4 = 2.5 \text{ MHz}$ , both with 50 % duty. Three stages would give 1.25 MHz.



### ANSWER

$Q_0 = 0011001100110011$  (5 MHz),  $Q_1 = 0000111100001111$  (2.5 MHz).  $Q_1Q_0$  counts 0, 1, 2, 3, 0, ...: a 2-bit ripple up-counter.

## 10 WORKED EXAMPLE 43-48 min

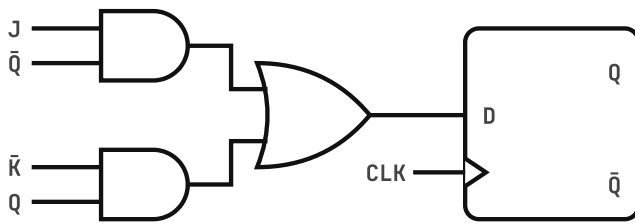
### Worked example: a JK flip-flop from a D flip-flop

#### GIVEN

A positive-edge D flip-flop ( $Q^+ = D$ ) and any gates you need.

#### FIND

Logic that makes it behave as a JK flip-flop.



Method: feed D with the target's characteristic equation (Mano, flip-flop conversion)

#### SOLUTION

1. A D flip-flop does  $Q^+ = D$ . So whatever we feed into D *becomes* the next state: D must equal the JK characteristic equation.

$$D = Q_{JK}^+ = J\bar{Q} + \bar{K}Q$$

2. Check with the excitation idea, row by row:  $Q = 0, J = 1 \rightarrow D = 1$  (set ✓);  $Q = 1, K = 1 \rightarrow D = 0$  (reset ✓);  $J = K = 1 \rightarrow D = \bar{Q}$  (toggle ✓);  $J = K = 0 \rightarrow D = Q$  (hold ✓).

3. Gates: two AND, one OR, one inverter on K ( $\bar{Q}$  is already available from the flip-flop).

The general recipe: feed the D input with the target flip-flop's characteristic equation. For other target/source pairs, use the source's excitation table.

#### ANSWER

**$D = J\bar{Q} + \bar{K}Q$ : two AND gates, one OR, one inverter on K.**

## Your turn • 2: convert

### 2 EXAM CQ

Using any gates, convert: (a) a **D** flip-flop into a **T** flip-flop; (b) a **JK** into a **D**; (c) a **JK** into a **T**.

#### SOLUTION

1. **(a)** T from D: D must equal T's next state.

$$D = T \oplus Q \quad (\text{one XOR gate})$$

2. **(b)** D from JK: D = 1 must set (J = 1, K = 0), D = 0 must reset (J = 0, K = 1):

$$J = D, \quad K = \bar{D}$$

3. **(c)** T from JK: hold when T = 0 (J = K = 0), toggle when T = 1 (J = K = 1):

$$J = K = T$$

#### ANSWER

(a)  $D = T \oplus Q$ . (b)  $J = D, K = \bar{D}$ . (c)  $J = K = T$ .

## Timing rules: setup, hold, clock-to-Q

- **Setup time  $t_{su}$** : D must be stable this long *before* the edge.
- **Hold time  $t_h$** : D must stay stable this long *after* the edge.
- **Clock-to-Q  $t_{pcq}$  (max) /  $t_{ccq}$  (min)**: how long after the edge Q changes.

$$\text{setup: } T_{clk} \geq t_{pcq} + t_{pd, logic} + t_{su} \quad \text{hold: } t_{ccq} + t_{cd, logic} \geq t_h$$

#### 74LS74-CLASS D FF (APPROX.)

|                      |                     |
|----------------------|---------------------|
| $t_{su}$             | 20 ns               |
| $t_h$                | 5 ns                |
| $t_{pcq}$ (max)      | 25-40 ns            |
| $f_{max}$ (toggling) | $\approx$ 25-33 MHz |

### ! GOTCHA

Break setup or hold and the flip-flop is back on the hilltop of last class: **metastable**.

## Worked example: how fast can it run?

### GIVEN

Flip-flop A drives two gates in series, which drive flip-flop B (same clock). Flip-flops:  $t_{pcq} = 25$  ns,  $t_{ccq} = 3$  ns,  $t_{su} = 20$  ns,  $t_h = 5$  ns. Gates:  $t_{pd} = 15$  ns max,  $t_{cd} = 3$  ns min; the shortest path has one gate.

### FIND

The maximum clock frequency, and whether the hold time is met.

Method: setup and hold constraints (two-flip-flop path)

### SOLUTION

1. The data launched at one edge must arrive, and settle for  $t_{su}$ , before the next edge:

$$T_{clk} \geq t_{pcq} + t_{pd, logic} + t_{su}$$

2. Longest path: 2 gates of 15 ns each (worst case):

$$T_{min} = 25 + 2(15) + 20 = 75 \text{ ns} \Rightarrow f_{max} = \frac{1}{75 \text{ ns}} = 13.3 \text{ MHz}$$

3. **Hold check** (shortest path, minimum delays): the new data must not arrive before the old has been held for  $t_h$ :

$$t_{ccq} + t_{cd, logic} = 3 + 3 = 6 \text{ ns} \geq t_h = 5 \text{ ns} \checkmark$$

4. Hold does *not* depend on the clock period. A hold violation cannot be fixed by slowing the clock: only by adding delay to the short path.

A setup violation is fixed by a slower clock (or faster logic). A hold violation is a design bug.

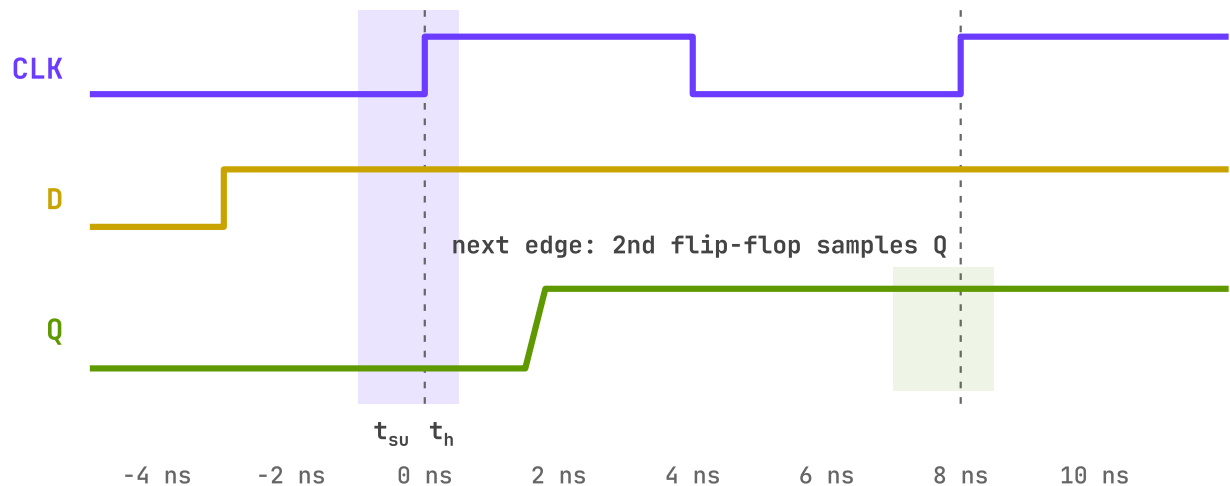
### ANSWER

$T_{min} = 75 \text{ ns}$ ,  $f_{max} = 13.3 \text{ MHz}$ . Hold:  $6 \text{ ns} \geq 5 \text{ ns}$ , met (by only 1 ns).

## Live lab: enter the forbidden window

### Setup and hold: the forbidden window

Slide the moment D changes. Outside the shaded window the flip-flop is predictable. Inside it, the output hangs halfway, for longer the closer you get to the balance point.



D changes at **-3.00 ns** relative to the clock edge

FLIP-FLOP 1

captured the **NEW** value (1)

2ND FLIP-FLOP (SYNCHRONISER)

clean

## Critical thinking • the button that

# crashes one board in a million



## CRITICAL THINKING

A button, a sensor or another chip's signal is **asynchronous**: it can change at any moment, including inside the setup–hold window.

Take  $\tau = 0.2$  ns,  $T_0 = 0.1$  ns,  $f_{\text{clk}} = 100$  MHz and an input that changes  $10^6$  times a second. Estimate the mean time between failures if the logic reads the flip-flop (a) 1 ns after the edge, (b) after a second flip-flop,  $\approx 8$  ns later.

## SOLUTION

1. One flip-flop, logic reads it 1 ns later:

$$\text{MTBF} = \frac{e^{1/0.2}}{(10^{-10})(10^8)(10^6)} = \frac{148}{10^4} \approx 0.015 \text{ s}$$

| About 70 failures per second.

2. Add a second flip-flop: the first has almost a whole period ( $\approx 8$  ns of the 10 ns) to settle:

$$\text{MTBF} = \frac{e^{8/0.2}}{10^4} = \frac{2.35 \times 10^{17}}{10^4} \approx 2.4 \times 10^{13} \text{ s} \approx 750,000 \text{ years}$$

3. Metastability is not removed, only made improbable. Each extra nanosecond of settling multiplies the MTBF by  $e^5 \approx 150$ .

## ANSWER

**Single flip-flop: MTBF  $\approx 15$  ms. Two-flip-flop synchroniser:  $\approx 750,000$  years. Always synchronise asynchronous inputs with (at least) two flip-flops.**

16 REAL WORLD 68–72 min

## Real world: counting heartbeats

A heart-rate meter: an R-peak detector (comparator on the ECG) produces one pulse per beat,  $\approx 80$  ms wide. A counter of flip-flops counts the pulses during a gate window; the processor reads the count.

- **Edge-triggered**, so each beat counts once however long the pulse lasts. A latch-based counter would race through several states during each 80 ms pulse.
- **Synchronised**: the heartbeat has nothing to do with the 1 MHz clock, so the pulse first passes a two-flip-flop synchroniser, then an edge detector ( $Q_1 \cdot \bar{Q}_2$ ) makes a one-clock-wide count enable.



## BME CORNER

Bedside monitors, Holter recorders and smartwatches all do this, in hardware or firmware. A miscounted beat is not a cosmetic bug: arrhythmia detection depends on exact R–R intervals.

You can (that is a ripple counter, next class), but then the count changes at a moment unrelated to the processor's clock. If the processor reads it mid-change it can read a value that never existed (e.g. 0111 → 1000 read as 1111). Synchronising first makes every change happen on a known edge.

17 POP QUIZ 72-76 min

## Pop quiz

### 1. A positive-edge-triggered D flip-flop changes its output...

- A. whenever D changes
- B. while CLK = 1
- C. only at the rising edge of CLK ✓**
- D. only at the falling edge of CLK

It samples D at the rising edge and holds that value for the whole period. A D latch would follow D during CLK = 1.

### 2. A JK flip-flop has J = K = 1 and receives 6 clock edges, starting from Q = 0. Q ends at...

- A. 0 ✓**
- B. 1
- C. undefined
- D. it depends on the pulse width

J = K = 1 toggles at every edge: 6 toggles bring it back to 0. (For a latch it would depend on the pulse width; for an edge-triggered FF it does not.)

### 3. A T flip-flop with T = 1 is clocked at 8 MHz. Its output frequency is...

- A. 16 MHz
- B. 8 MHz
- C. 4 MHz ✓**
- D. 2 MHz

It toggles once per edge, so two clock periods make one output period: divide by 2. A 50 % duty cycle comes for free.

### 4. Which violation can be fixed by lowering the clock frequency?

- A. hold
- B. setup ✓**
- C. both
- D. neither

Setup:  $T_{\text{clk}} \geq t_{\text{pcq}} + t_{\text{logic}} + t_{\text{su}}$ , so a longer period helps. Hold has no  $T_{\text{clk}}$  in it.

### 5. Why do we pass an asynchronous button signal through two flip-flops in series?

- A. to double its frequency
- B. to give a metastable first flip-flop a whole clock period to settle ✓**
- C. to invert it twice
- D. to debounce it

The first flip-flop may go metastable. The second samples it one period later, by which time it has (almost certainly) resolved. MTBF grows like  $e^{t/\tau}$ .

## Three things to remember

### O1

A **flip-flop** samples at the clock **edge** (master-slave: two latches never open together). A latch is transparent for the whole level.

### O2

**Q\***:  $SR = S + \bar{R}Q$ ,  $JK = J\bar{Q} + \bar{K}Q$ ,  $D = D$ ,  $T = T \oplus Q$ . Convert by feeding the source's inputs from the target's equation.

### O3

**Timing**:  $T_{clk} \geq t_{pcq} + t_{logic} + t_{su}$ ;  $t_{ccq} + t_{cd} \geq t_h$ . Asynchronous inputs: two-flip-flop synchroniser.

### HW

BEFORE NEXT CLASS

Flip-flops:  $t_{pcq} = 12 \text{ ns}$ ,  $t_{ccq} = 4 \text{ ns}$ ,  $t_{su} = 6 \text{ ns}$ ,  $t_h = 3 \text{ ns}$ . Between two of them: a longest path of 3 gates (8 ns each, max) and a shortest path of 1 gate (2 ns min).

(a)  $f_{max}$ ? (b) Is hold met? (c) Is hold met for a direct flip-flop-to-flip-flop wire (a shift register)?

**Class Test 1: next class (Week 5, Class 1), 20 minutes, Weeks 1-4.** Assignment 1 is due at the start of the same class.

### SOLUTION

1. (a)

$$T_{min} = 12 + 3(8) + 6 = 42 \text{ ns} \Rightarrow f_{max} = 23.8 \text{ MHz}$$

2. (b) Shortest path:  $4 + 2 = 6 \text{ ns} \geq 3 \text{ ns}$ : hold met.

3. (c) Direct connection:  $t_{ccq} = 4 \text{ ns} \geq 3 \text{ ns}$ : still met. (With  $t_h = 5 \text{ ns}$  it would fail, at any clock speed.)

### ANSWER

$f_{max} = 23.8 \text{ MHz}$ ; hold met with logic ( $6 \text{ ns} \geq 3 \text{ ns}$ ) and without ( $4 \text{ ns} \geq 3 \text{ ns}$ ).

**S. M. Sakeef Sani**

Adjunct Lecturer, CSE · Northern University Bangladesh

smsakeefsani.hbitcorp.com