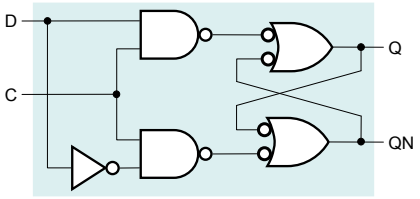
14:332:231
DIGITAL LOGIC DESIGN

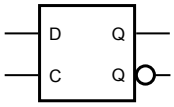
Ivan Marsic, Rutgers University
Electrical & Computer Engineering
Fall 2013

Lecture #16: D Latch ; Flip-Flops

D Latch

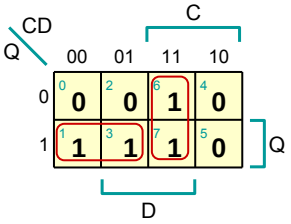
- "Data" latch: single input "D", plus enable input "C"
- The D Latch eliminates the S=R=1 problem of the SR latch
- However, violations of setup and hold time still cause metastability …





Inputs		Outputs	
C	D	Q	QN
1	0	0	1
1	1	1	0
0	x	last Q	last QN

Q*:

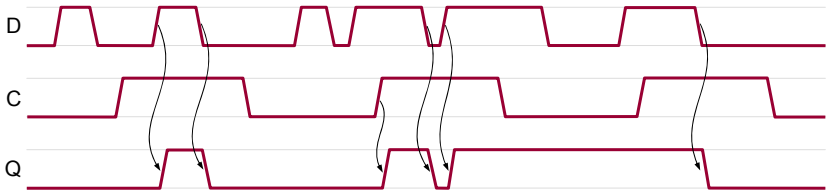


Characteristic equation:

$Q^* = C' \cdot Q + C \cdot D$

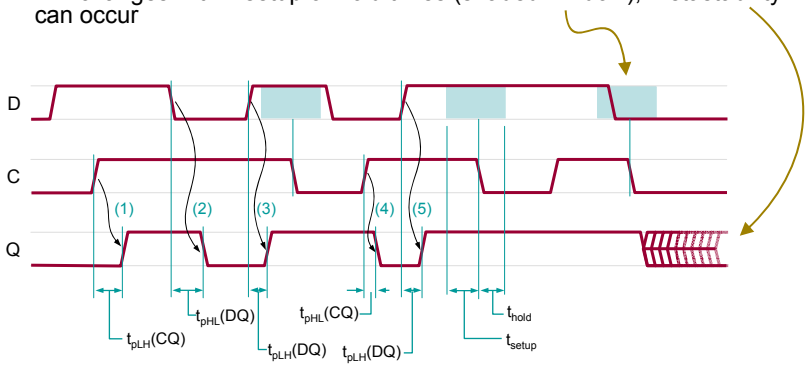
D Latch Operation

- Functional behavior of a D latch for various inputs:
  - When the enable input C is asserted, the Q output follows the D input
  - The latch is said to be "open" and the path from D input to Q output is "transparent" → *transparent latch*
  - When enable (C) is asserted, the latch acts like a combinational circuit!



D Latch Timing Parameters

- Propagation delay (from C or D to output Q)
- Setup time (D before C falling/rising edge)
- Hold time (D after C falling/rising edge)
  - If D changes within setup or hold times (shaded window), metastability can occur



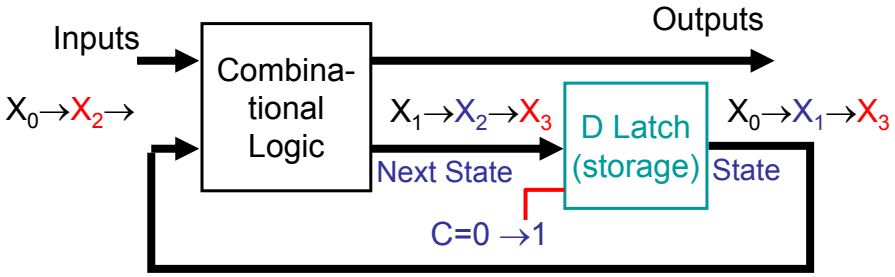
(1) & (4) — Latch initially closed and when C asserted, Q changes
(2), (3) & (5) — Latch already opened, so Q transparently follows D

Latch Issues

- Latches can cause serious timing problems (races) in sequential circuits
  - Due to the fact that a latch is "transparent": D = Q while the clock C = 1
    - *Advantage*: input changes can be seen immediately at output
    - *Disadvantage*: output changes every time input changes (while C = 1)
  - Desired behavior for a memory device: stored value changes only at discrete time instants, determined by rising or falling edges of *clock* signal
- The timing problems can be avoided by using "Flip-Flops"

The Latch Timing Problem (cont'd)

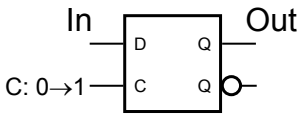
- Similar timing problems in the sequential circuits:
  - Recall that an enabled latch acts like a combinational circuit
  - Hence, we possibly have two cascaded combinational circuits feeding each other, generating oscillations and unstable transient behavior



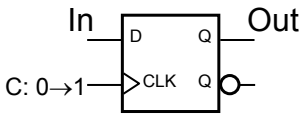
- Need for circuits that change state **only once per clock cycle**:
  - C=1:
    - Now the current state becomes X_1 and a new state is generated by the combinational logic circuit: X_2.
    - However, if C=1, the new "next state" X_2 will create a new current state X_2!, etc…

Use Flip-flops to Solve the Timing Problem

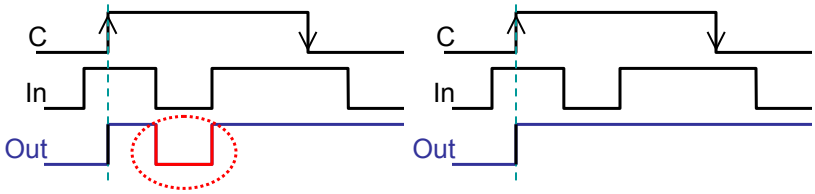
- A solution to the latch timing problem is to break the closed path from In to Out within the storage element



D latch



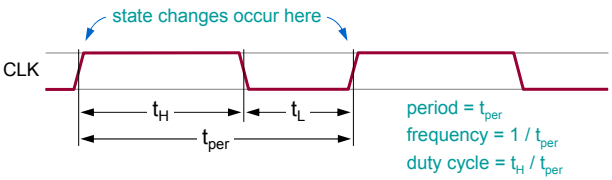
D flip-flop



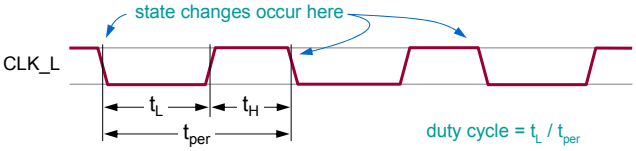
Clock Signals

- Clocks are regular periodic signals used to specify state changes
- Very important for sequential circuits
  - State variables change state at the clock edge

Active high:

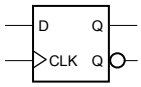


Active low:



Flip-flops

- Flip-flops change state only on the edge of a clock (CLK) pulse (either rising or falling edge)
  - Symbolized by the *dynamic input indicator* |>
- Clock is *active high* → then the state changes on the *rising* (positive) edge
- Clock is *active low* → then the state changes on the *falling* (negative) edge

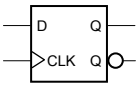


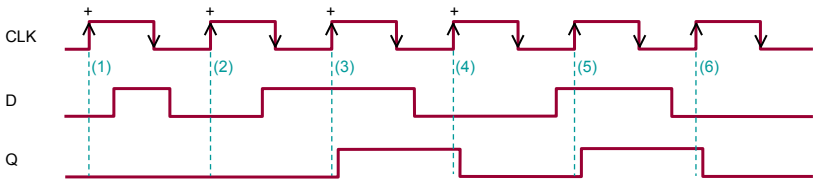
The D Flip-flop: The Principle

- Positive-edge triggered D flip-flop
- Inputs sampled on rising edge; outputs change after rising edge

Inputs		*Outputs*	
D	**CLK**	**Q**	**QN**
0	↑	0	1
1	↑	1	0
x	0	last Q	last QN
x	1	last Q	last QN

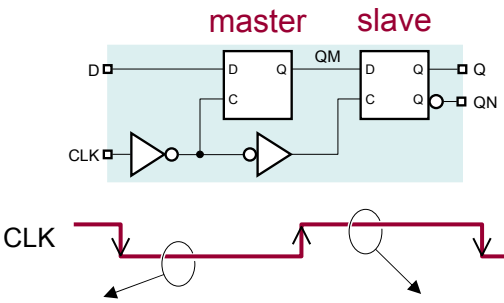
Next-state equation: $Q^* = D$





The D Flip-flop: Implementation

- Consists of two clocked D latches in series with the clock on the second latch inverted

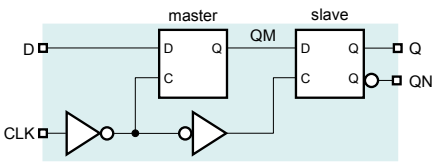


- Master latch responds to input (QM changes)
- Slave latch is inactive
- Output Q unchanged

- Master latch is inactive
- Slave latch responds to input QM
- Output Q changes

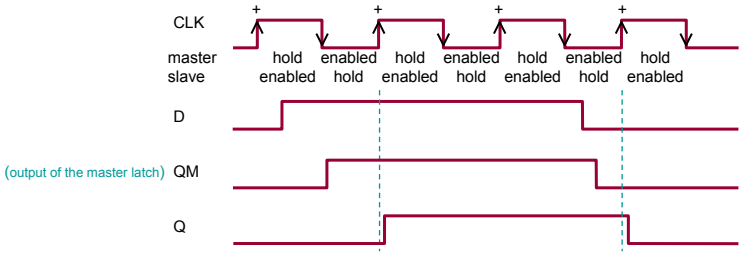
The D Flip-flop: Implementation

- Positive-edge triggered D flip-flop:
  - Output changes at positive clock edge



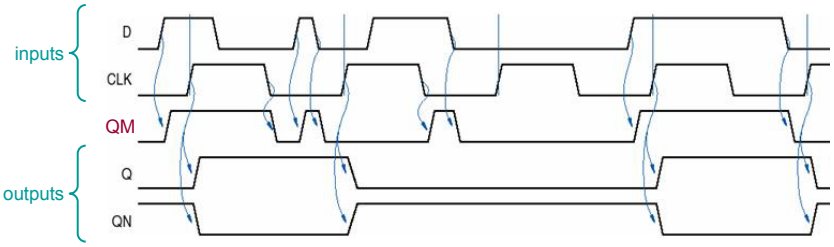
Recall D latch table:

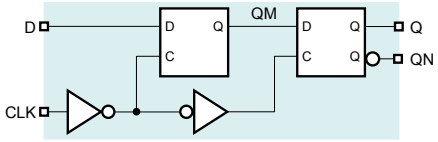
	Inputs		Outputs	
	C	D	Q	QN
(enabled)	1	0	0	1
	1	1	1	0
(hold)	0	x	last Q	last QN



Edge-triggered D Flip-flop Behavior

- Master latch output QM changes only when CLK is "0"
- When CLK → "1", the current QM value is transferred to Q





Inputs		Outputs	
D	CLK	Q	QN
0	↑	0	1
1	↑	1	0
x	0	last Q	last QN
x	1	last Q	last QN

13 of 23

Flip-flop Timing Constraints

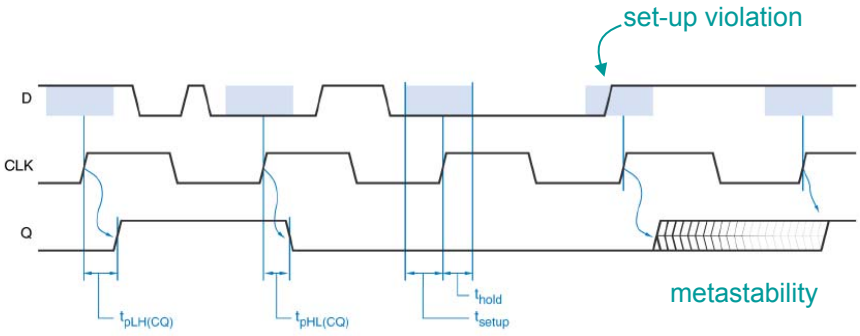
Proper operation requires strict timing rules:

- Minimum clock pulse width: t_{pw} (t_{pwH}, t_{pwL})
- **Set-up time** (t_{setup}):
  minimum time that the input signal must be present prior to a clock edge
- **Hold time** (t_{hold}): minimum time the input must be kept after the clock edge

14 of 23

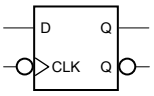
D Flip-flop Timing Parameters

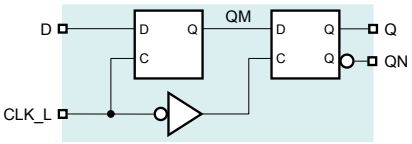
- Propagation delay (from CLK)
- Setup time (D before CLK)
- Hold time (D after CLK)
- Shaded window duration is $t_{setup} + t_{hold}$ —— values depend on the technology and gate delays



Other D Flip-flop Variations

- Negative-edge triggered D flip-flop
  - Simply inverts the clock input — active low
  - Action takes place on the falling edge of CLK_L

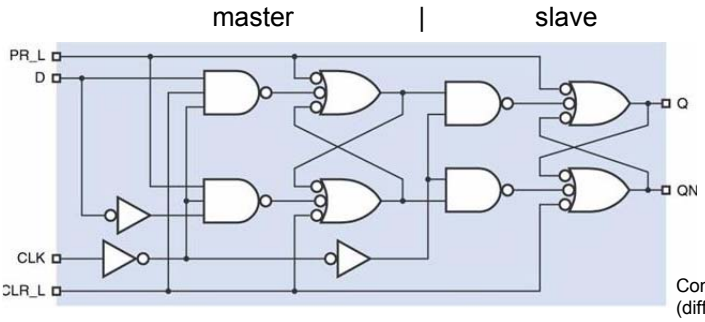


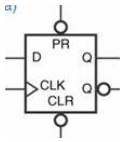


Inputs		*Outputs*	
D	**CLK_L**	**Q**	**QN**
0	↓	0	1
1	↓	1	0
x	0	last Q	last QN
x	1	last Q	last QN

D Flip-flop w/ Asynchronous Inputs

- with **Preset** (PR) and **Clear** (CLR), similar to set/reset of S-R latch
  - Used to force the flip-flop to a particular state independent of CLK & D
  - Suitable for initialization and testing purposes, e.g., to initialize a counter
- Positive-edge-triggered





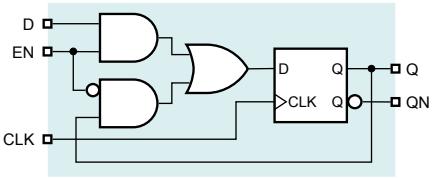
Commercial version is 74x74 (different from this diagram)

- Placing a "0" on CLR_L forces the state Q = 0
- If CLR_L = 1, no effect on NAND gates
- Similarly, PR_L = 0 forces the state Q = 1, and PR_L = 1 has no effect

17 of 23

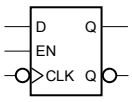
Clock Enable Positive-edge-triggered D Flip-flop

- Flip-flops sometimes include additional logic:
  Clock enable (EN or CE)
- At clock edge, holds the last value stored, rather than load a new value
- A 2-input multiplexer controls the input value of the internal D flip-flop



Inputs			*Outputs*	
D	**EN**	**CLK**	**Q**	**QN**
0	1	↑	0	1
1	1	↑	1	0
x	0	↑	last Q	last QN
x	x	0	last Q	last QN
x	x	1	last Q	last QN

- If EN is asserted, the external input D is selected
- If EN is negated, the flip-flop's current output is used



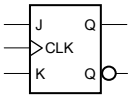
18 of 23

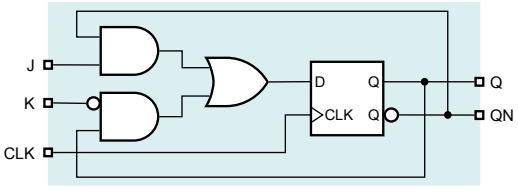
Sections not covered & not required…

- Wakerly, Section 7.2.7
  "Scan Flip-flop"
- Wakerly, Section 7.2.8
  "Master/slave S-R Flip-flop"
- Wakerly, Section 7.2.9
  "Master/slave J-K Flip-flop"

Edge-triggered J-K Flip-flop

- Not used much anymore, except in the lab …





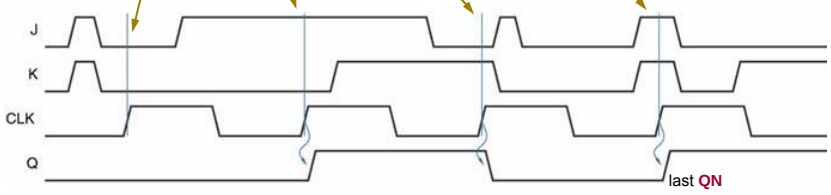
Inputs			Outputs	
J	K	CLK	Q	QN
x	x	0	last Q	last QN
x	x	1	last Q	last QN
0	0	↑	last Q	last QN
0	1	↑	0	1
1	0	↑	1	0
1	1	↑	last QN	last Q

Next-state equation: $Q^* = J \cdot Q' + K' \cdot Q$

Functional Behavior of a Positive-edge-triggered J-K Flip-flop

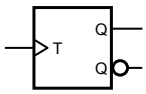
- The commercial version is a TTL positive-edge-triggered J-K flip-flop 74x109

Inputs			Outputs	
J	K	CLK	Q	QN
x	x	0	last Q	last QN
x	x	1	last Q	last QN
0	0	↓	last Q	last QN
0	1	↓	0	1
1	0	↓	1	0
1	1	↓	last **QN**	last Q

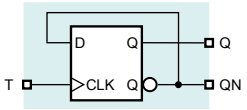


T (toggle) Flip-flop

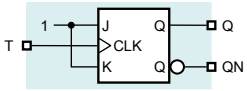
- Changes state at every clock tick
  - Signal on the Q output has frequency = ½ T
- Used in counters and frequency dividers
- Can be constructed from D or J-K flip-flops



D flip-flop:



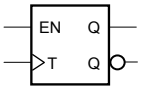
J-K flip-flop:



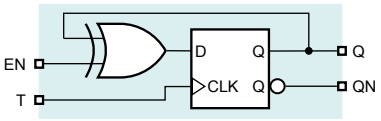
Next-state equation: $Q^* = Q'$

T (toggle) Flip-flop with Enable

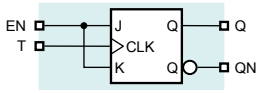
- Flip-flop toggled only when enabled
  - Important for counters
- t_{setup} and t_{hold} must be satisfied w.r.t. T and EN



T flip-flop with enable: D



J-K flip-flop



Next-state equation: $Q^* = Q \oplus EN$