CSE 260M / ESE 260
Intro. To Digital Logic & Computer Design

Bill Siever
&
Jim Feher

This week

- Office hours posted! https://wustl-cse260m-fl24.github.io/
- Hw#2 Clarification

Review

- K-Maps: What & why?

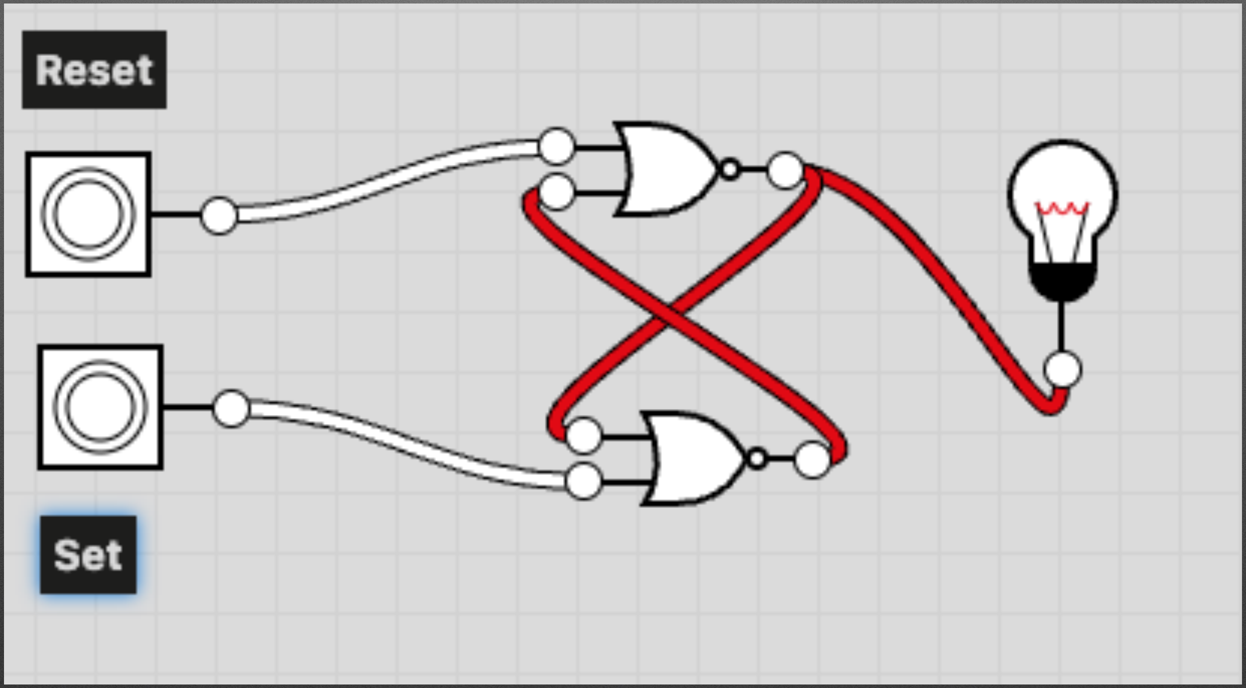
K-Maps

Goal: Store Data

Stable, Reinforcing Setup: SR Latch

- On-line Demo: https://logic.ly/
  - Bistable: Two stable configurations
- Goal: Met!
  - S=Set, R=Reset

SR Latch

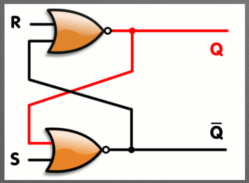


Goal: Store Data

- Set/Reset is *inconvenient*
  - We want something like, data=X, where x is 0 or 1 (store X, not "set or reset data based on X")
  - We want to store X in data *when we're ready to!*
- *Clock (Clk)*: Indicates *when* we want to change the data

D-Latch

- Start with SR Latch

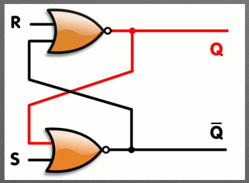


- Describe Desired Behavior (of output, Q)

CLOCK	DATA	Q
0	0	(Unchanged)
0	1	(Unchanged)
1	0	0
1	1	1

D-Latch

- Start with SR Latch

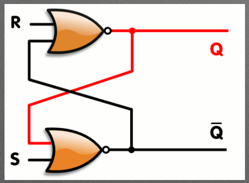


- Describe Desired Behavior (of output, Q)

CLOCK	DATA	Q
0	0	Q_{prev}
0	1	Q_{prev}
1	0	0
1	1	1

D-Latch

- Start with SR Latch

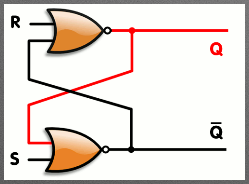


- Describe Desired Behavior (of output, Q)

CLOCK	DATA	Q
0	0	Q_{prev}
0	1	Q_{prev}
1	0	RESET
1	1	SET

D-Latch

- Start with SR Latch

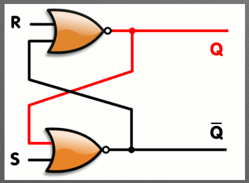


- Describe Desired Behavior (of output, Q)
- Just combinational logic

CLOCK	DATA	Q
0	0	Q_{prev}
0	1	Q_{prev}
1	0	RESET
1	1	SET

D-Latch

- Start with SR Latch

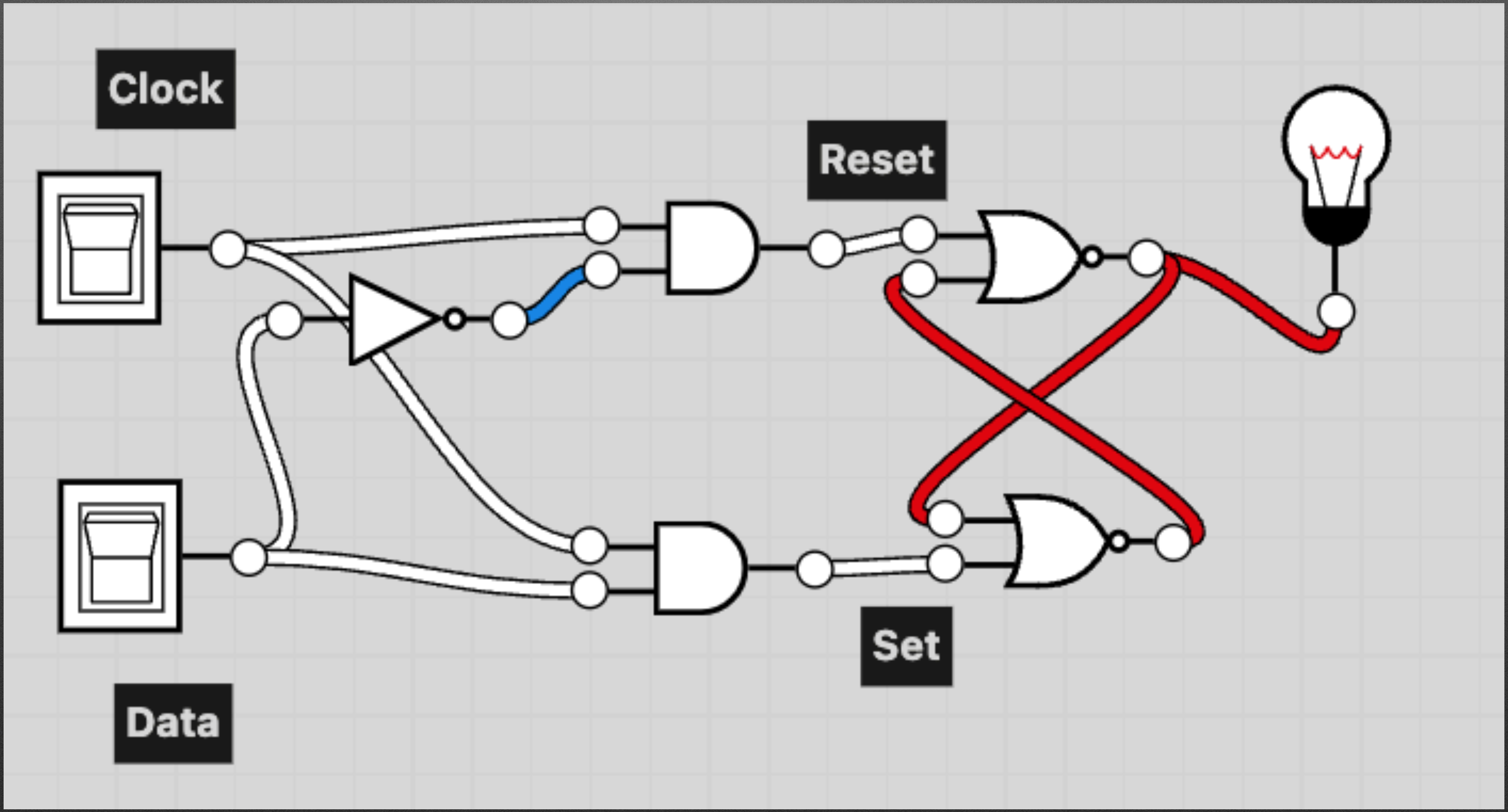


- Describe Desired Behavior (of output, Q)
- Just combinational logic
- Reset = Clock * /Data
  Set = Clock * Data

CLOCK	DATA	Q
0	0	Q_{prev}
0	1	Q_{prev}
1	0	RESET
1	1	SET

Updates: https://logic.ly/

D-Latch



D-Latch

- “Latches on” to last data value when clock goes low
    - Is sensitive to the *level* of the clock
    - Is “transparent” when the clock is high

Goal: Store Data

- D-Latch is still a bit *inconvenient*
  - We'd like something like a (simple) camera
  - Instant shutter is "pressed" we capture data

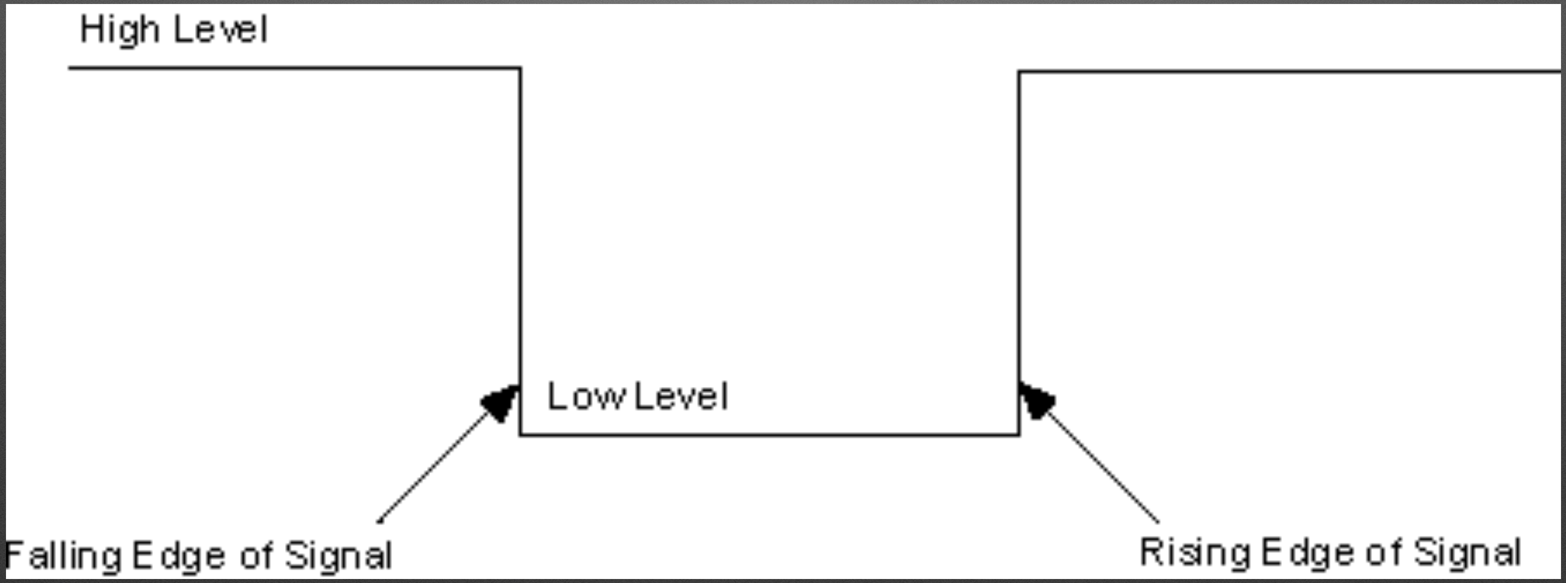
Flip-Flop

https://openclipart.org/detail/288726/flip-flops-4

D Flip-Flop

- Two D-Latches with clocks in opposite states (via an inverter)
  - First stage: Transparent when clock is Low
  - Second stage: Transparent when clock is High
  - Effect: Capture D at precise instant clock goes from low to high
    - I.e. the clock EDGE
    - Edge triggered. Specifically, Rising Edge Triggered

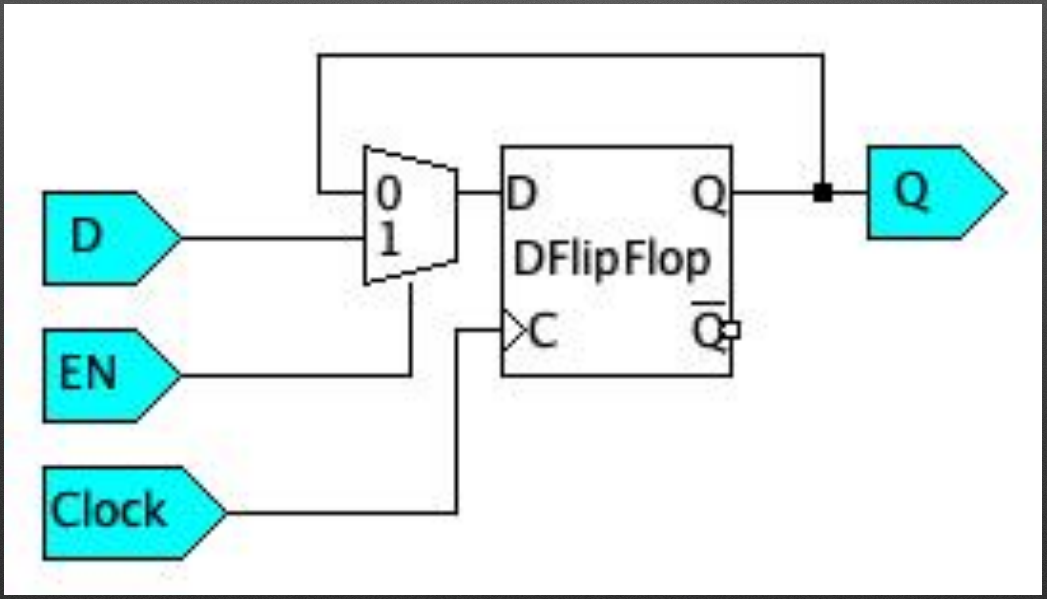
Signal Edges



https://www.ni.com/docs/en-US/bundle/ni-hsdio/page/hsdio/fedge_trigger.html

"Enable"

- We may want to have two things control timing: the clock and an enable
    - Ex: X[0] = 1 (in a program) . We only want to modify X *when that line runs.*



Combinational Logic

- (Purely) combines current inputs to produce output
  - Doesn't depend on past inputs
  - Can be represented with a simple table
  - One-way: Doesn't have any feedback paths from output back to inputs

Sequential & Synchronous Logic

- Need to know sequence of inputs
    - Can’t be represented with a *simple* table of just inputs and outputs (Possibly a complex table of history of inputs and outputs)

Synchronous Sequential Circuits

- Are *synchronized* by a common clock
- Uses registers
- Mix of registers and combinational logic
- Cycles include at least one register
- Goal: Impose predictable behavior!

Finite State Machines

- State: A condition of being
- Finite: Er. Finite
  - Real machine has real-world limitations: $k \times$ D-latches
  - k D-Latches means $\leq 2^k$ states (finite)

Questions

- [Favorite vacation spot?]
- [Exams: Like Hw?]: Yep.
- [State Machine Encoding Choices?]
  Memory vs. Logic: They can impact complexity of combinational logic.
  (Typically One-hot: more memory, but simpler logic))
- [FSMs? Mealy? Moore?]: More (Moore?) next time
- [I don’t get/understand Latches/Edges/Levels/etc.]
- [Which memory things are improtant]: (Rising) Edge Triggered D-latch.
  (Most others were just part of the journey to it)
- [Starbursts Dissolve in Hair Cream]? Whahhh????

Next Time

- Studio
- Homework 2 due Thursday night!