Beyond classical circuit design lecture 9

Alternative Design Styles

Further Reading

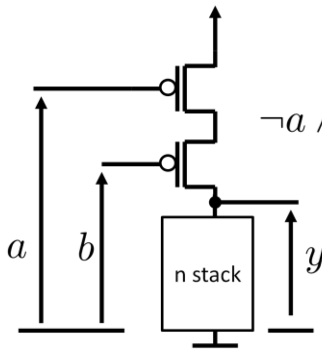
Jan M. Rabaey, Anantha Chandrakasan, Borivoje Nikolic: *Digital Integrated Circuits. A Design Perspective. 2nd edition.* Prentice Hall, 2003.

Remember: CMOS Design

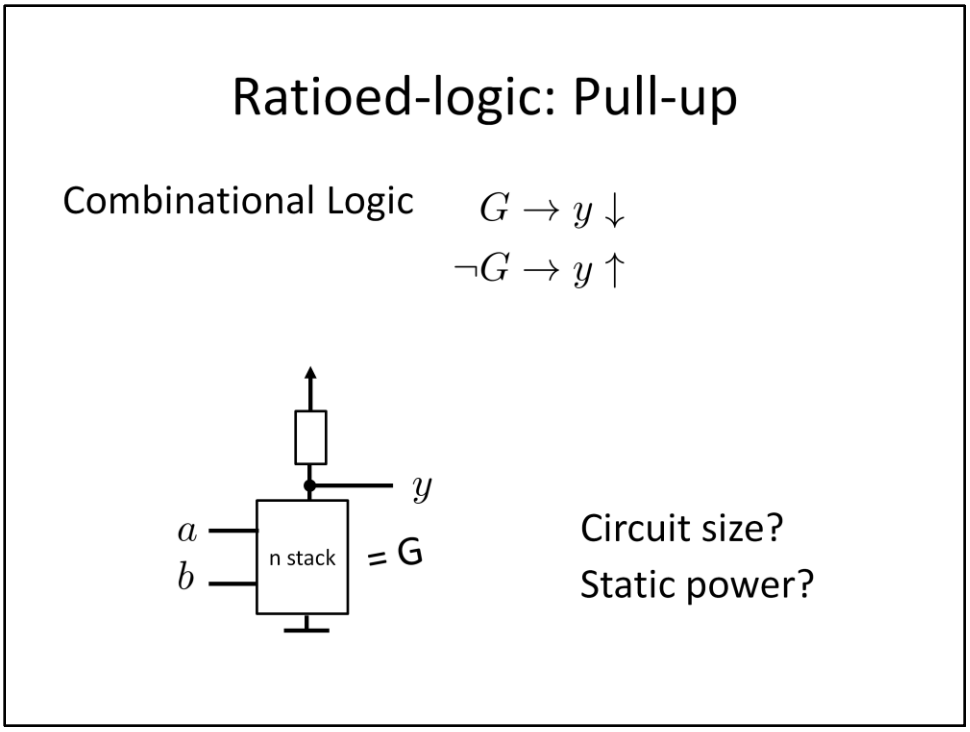
Combinational Logic:

n-stack: down transition $G \rightarrow y \downarrow$

p-stack: up transition $\neg G \rightarrow y \uparrow$



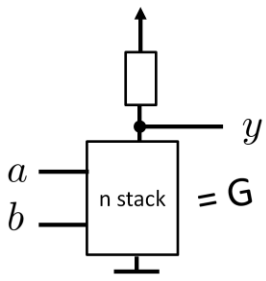
$\neg a \wedge \neg b \rightarrow y \uparrow$



Ratioed-logic: Pull-up

Combinational Logic $G \to y\downarrow$

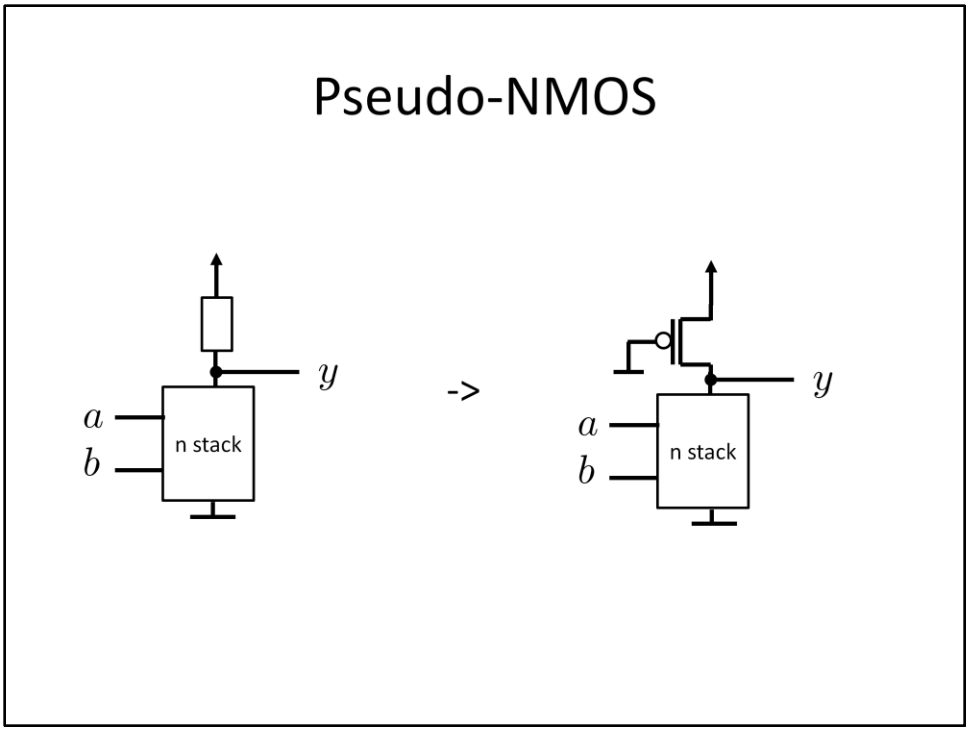
$\neg G \to y\uparrow$



Circuit size?

Static power?

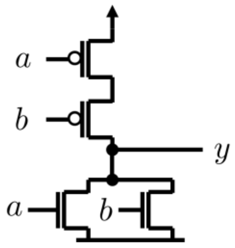
pull-up instead of p-stack



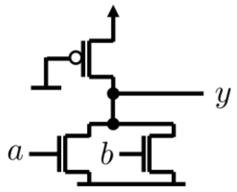
Pseudo-NMOS

implementation with MOSFETs
thus the name “ratioed”: the pMOS acting as a pull-up resistor must not be too strong compared to the pull-down nMOS
that is: the ratio W/L of pMOS and W/L of nMOS must fulfill a certain ratio for the circuit to work correctly.

Pseudo-NMOS NOR

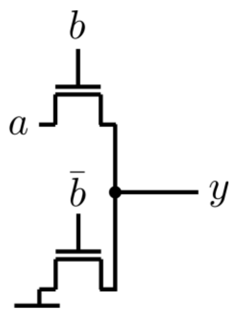


versus



Pass-transistor Logic

AND gate

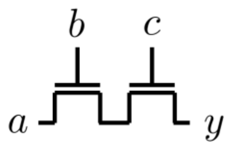


Mind: reduced voltage swing!

Pass-transistor Logic

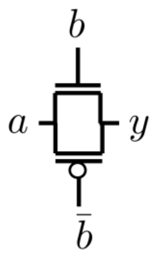
In general:

... what about this wrt. voltage swing?

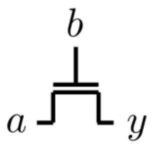


Pass-transistor Logic

Complete pass gate



versus



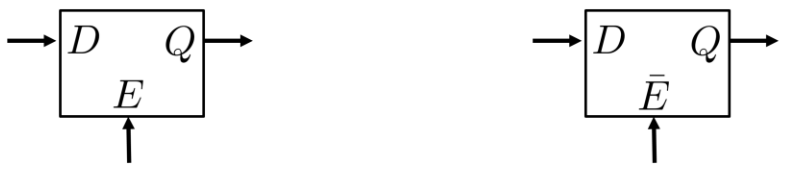
[Hw]

Beyond classical circuit design lecture 9.5

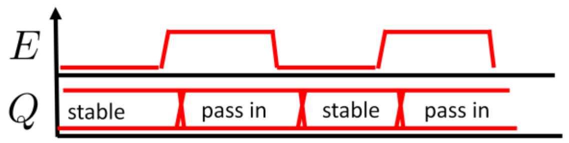
Clocked Design Styles

Latch

Positive latch Negative latch



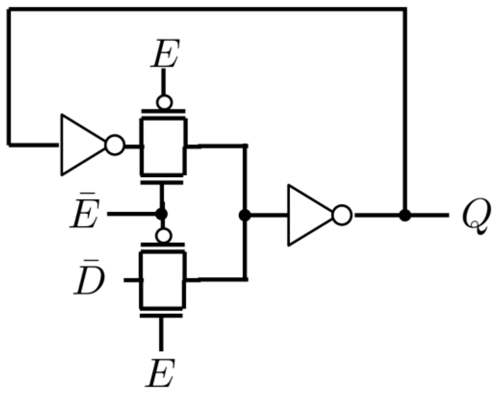
Positive latch behavior



negative latch specified analogously with inverted enable signal E_

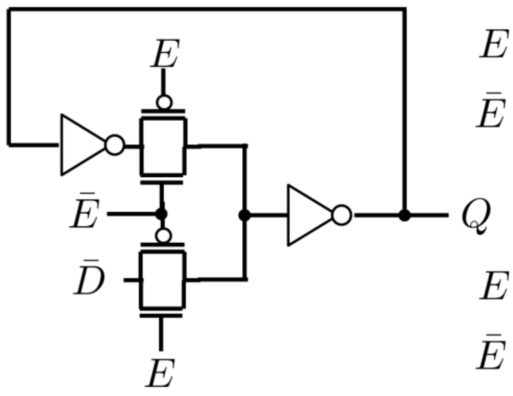
Positive Latch Implementation

by transmission gates

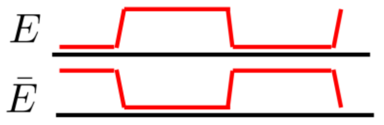


Positive Latch Implementation

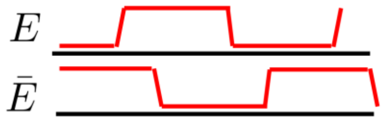
by transmission gates



But mind:



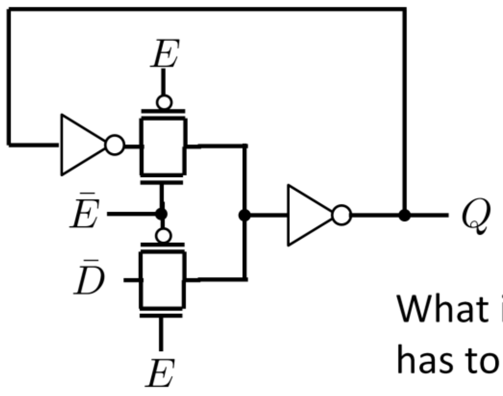
versus



Assumes a perfectly inverted signal E_ = not E.

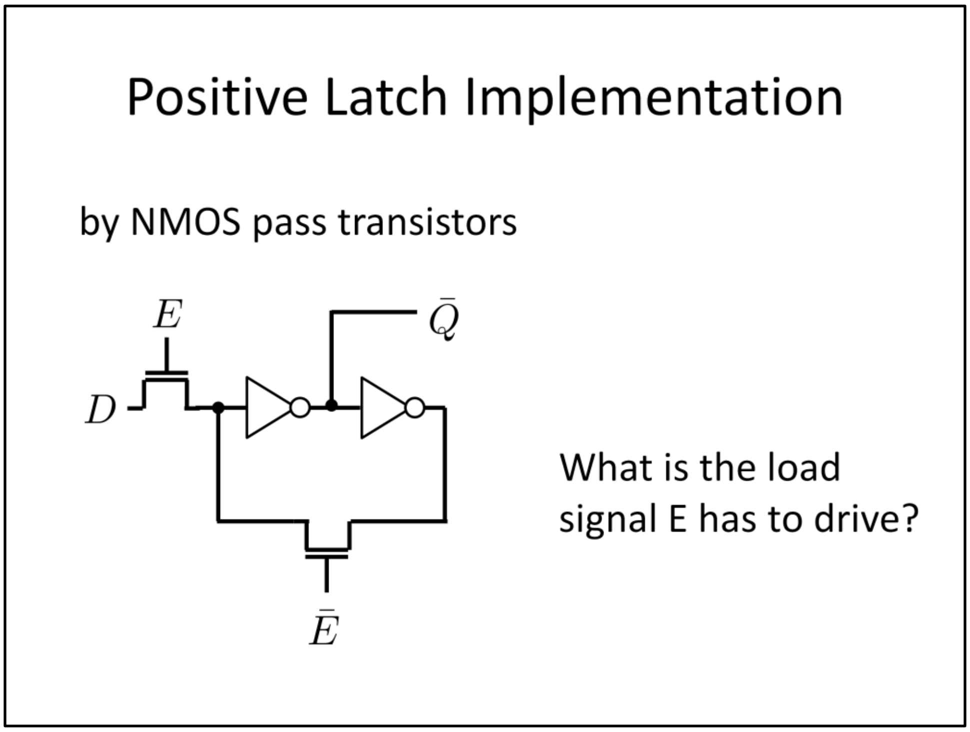
Positive Latch Implementation

by transmission gates



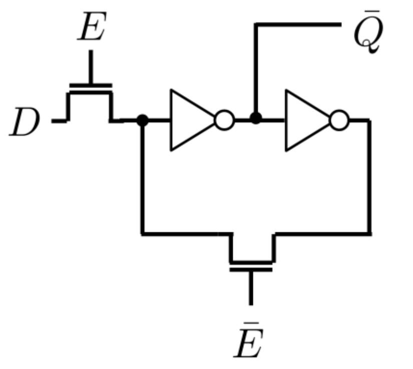
What is the load signal E has to drive?

E and E_ drive 4 transistors.



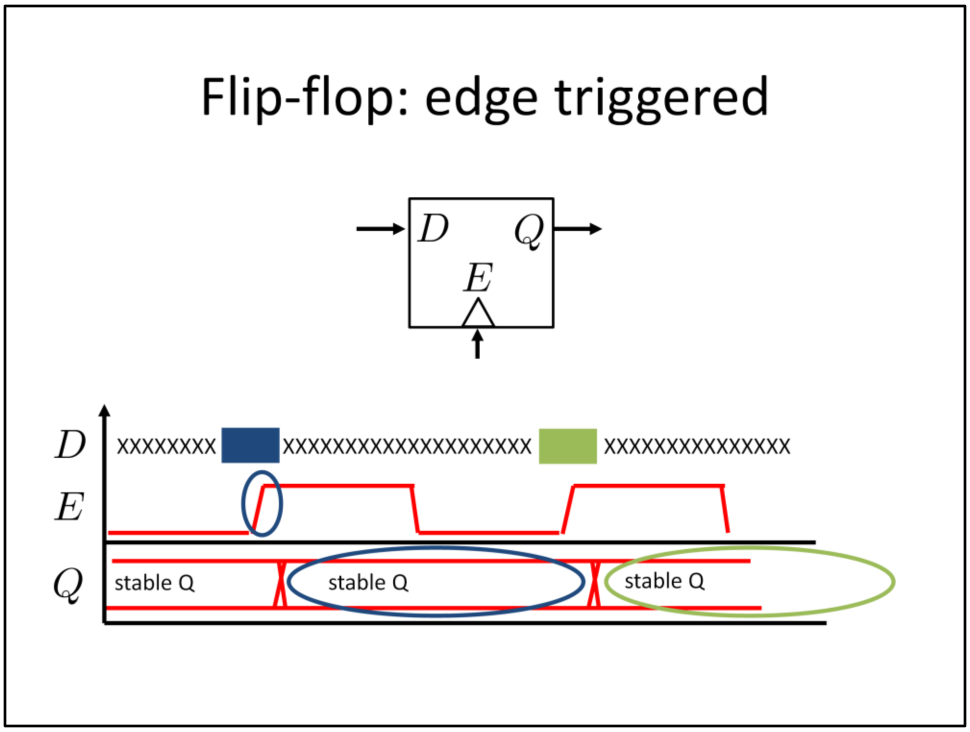
Positive Latch Implementation

by NMOS pass transistors



What is the load signal E has to drive?

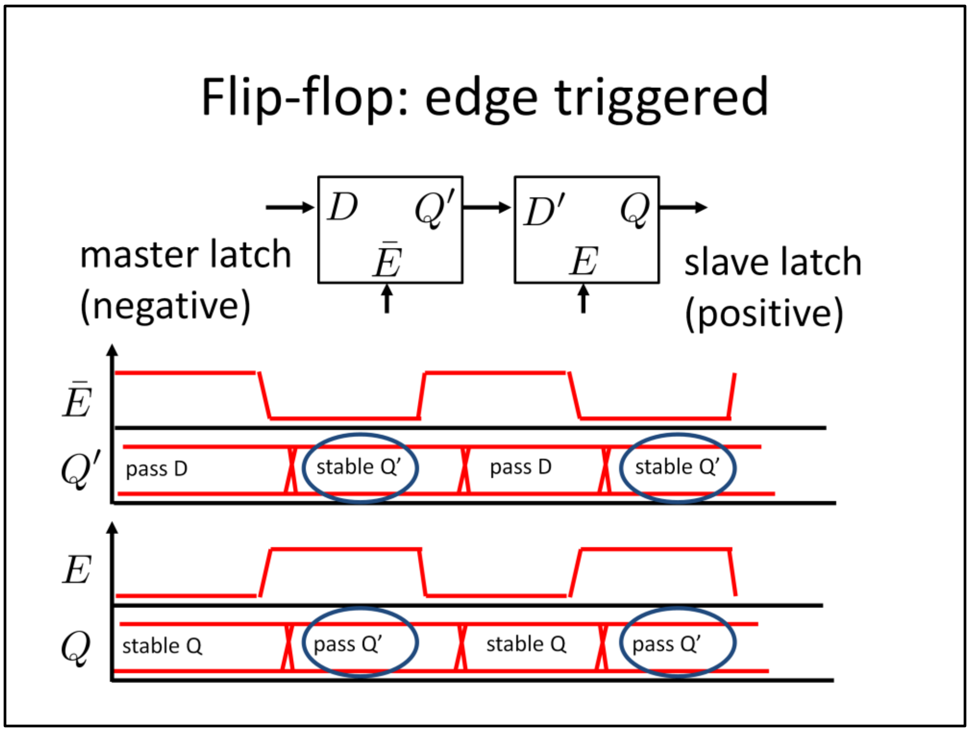
Using Q instead of Q_ increases the load for the pass gate. Decoupling it by an INV is often a better choice to get a fast storage loop.
E and E_ drive 2 transistors.



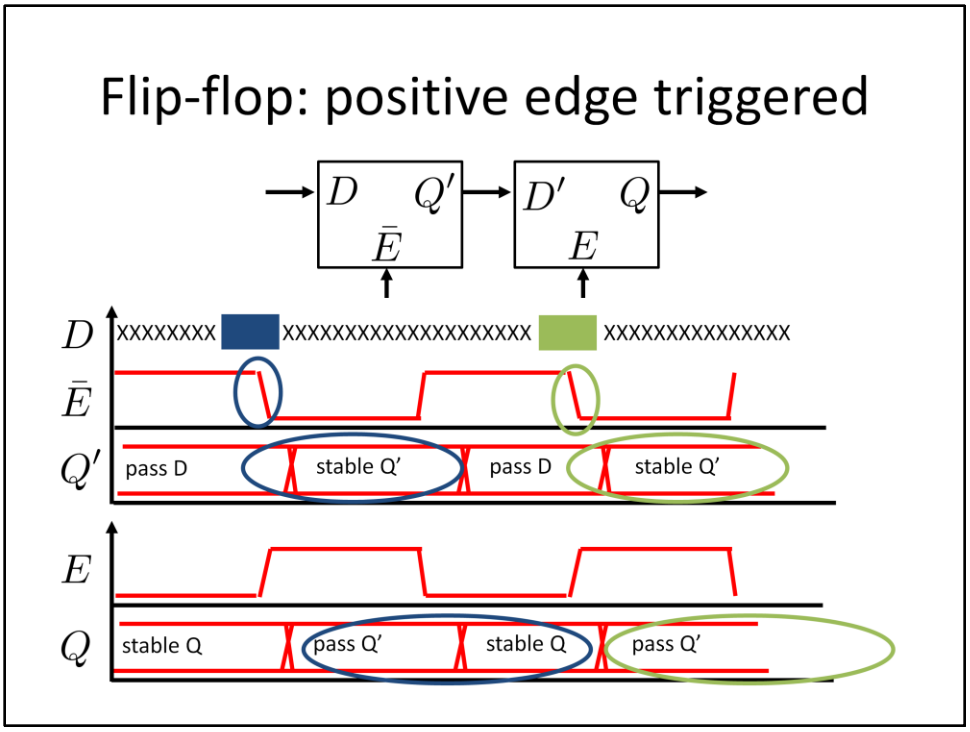
Flip-flop: edge triggered

positive edge triggered flip-flop:
on positive transition of E -> copy stable D to output Q and hold until next change.

assumption: D is table around positive edge of E (before edge: setup time, after edge: hold time).
If not: either copy arbitrary value to output, or even become metastable (discussed later on)

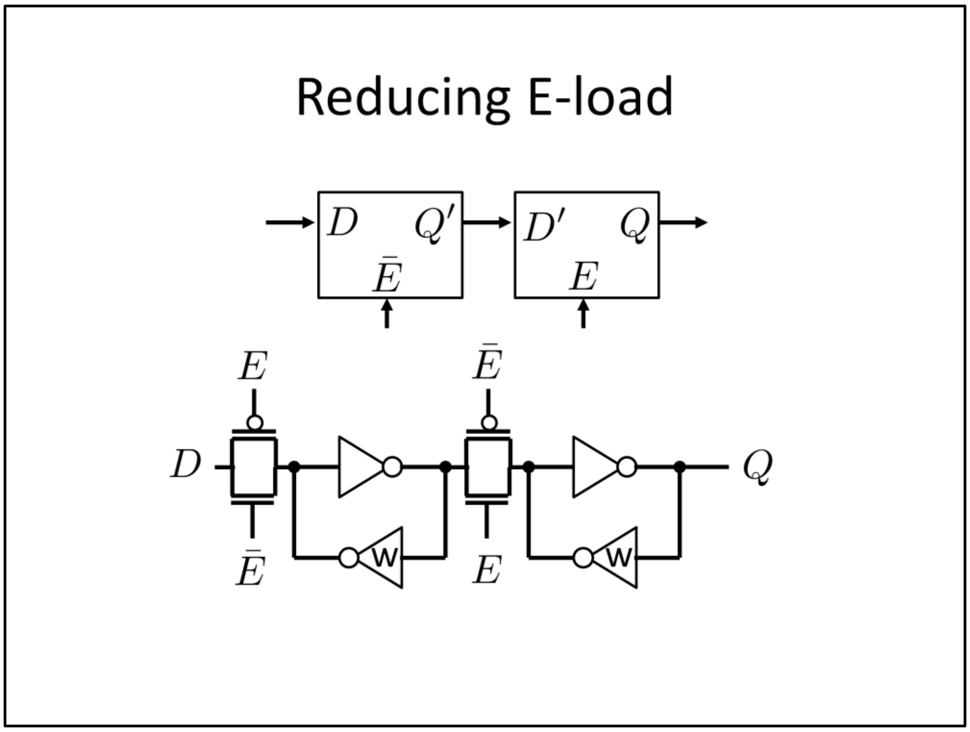


whenever slave passes, master is stable.

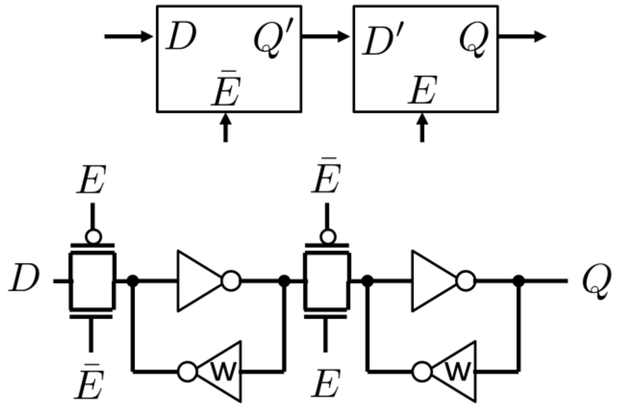


Flip-flop: positive edge triggered

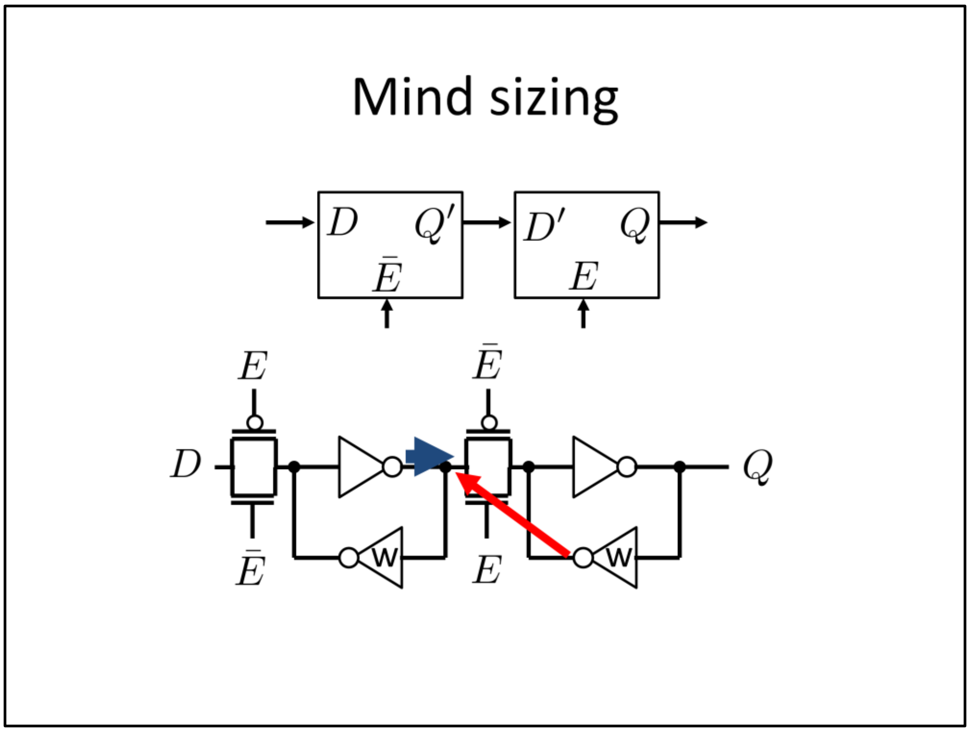
blue & green: data waves that are stable values.

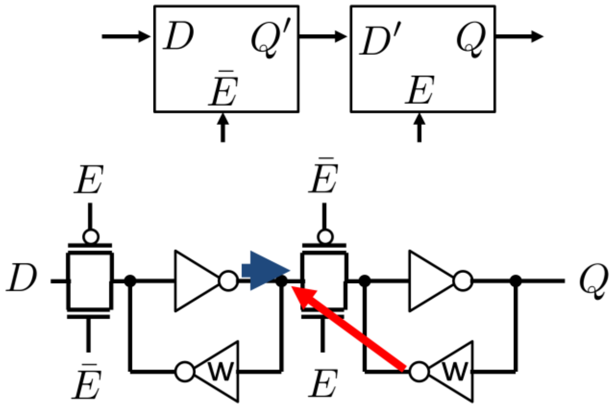


Reducing E-load



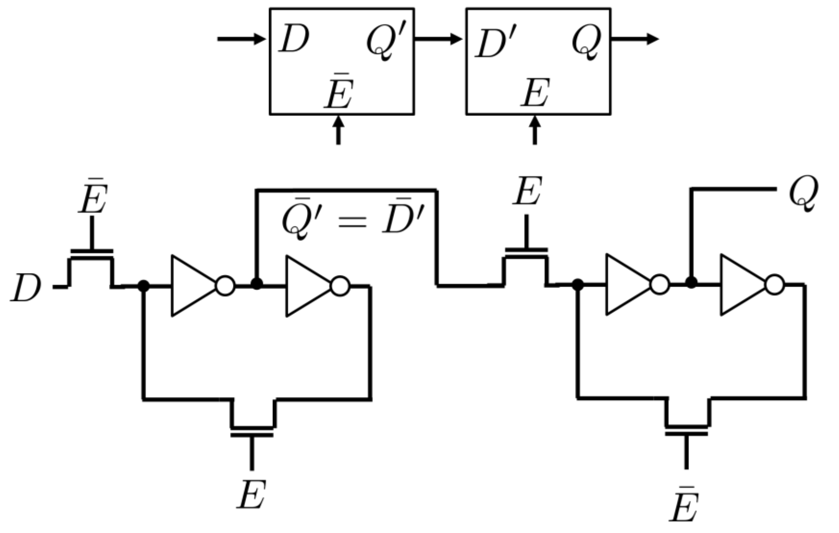
by removing the feedback-loop pass gates, we reduce size and load signals E and E_ have to drive


Mind sizing



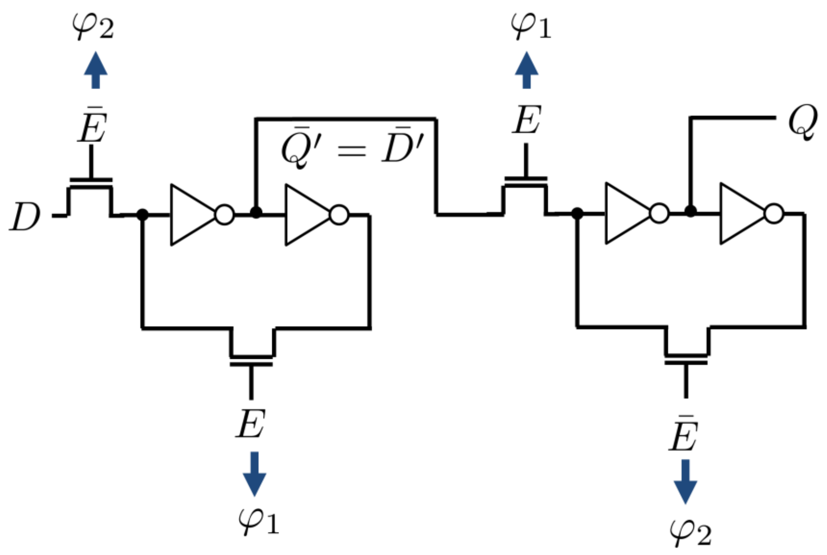
removing the feedback -> attention has to be paid that the slave could influence the master.
However, weak INV drives against INV.

Preventing this...



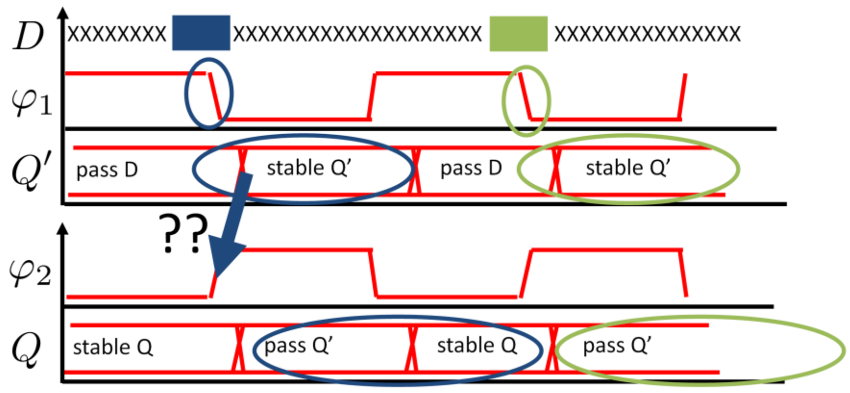
We would like to prevent this (driving against each other). But, do not want to increase load -> use the nMOS pass transistor implementation from before.

Mind phases



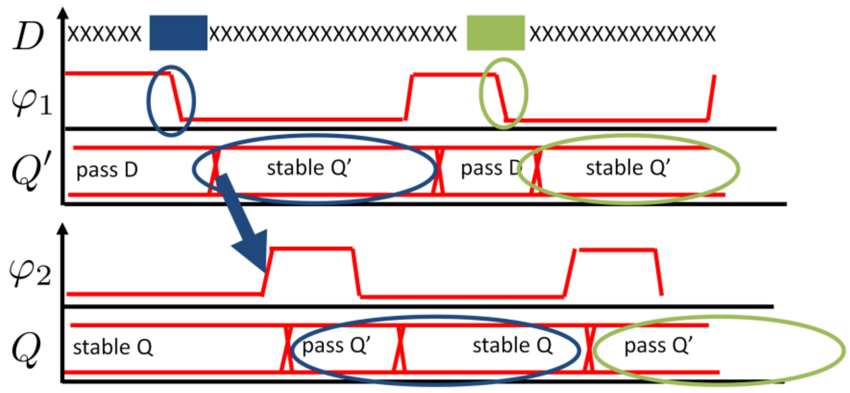
instead of E and E_ use two aligned phases phi1 and phi2.
Here: just renaming and decoupling from E.

Mind phases



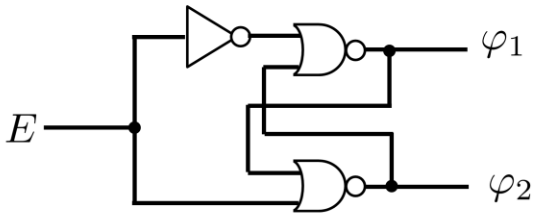
possible timing violation

Mind phases



clearly separated phase signals phi1 and phi2.

Generate phases (locally)

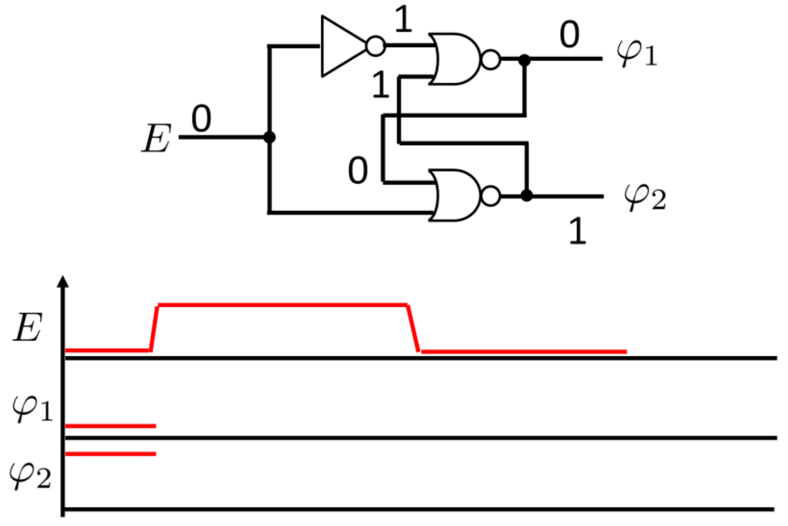


locally since otherwise:

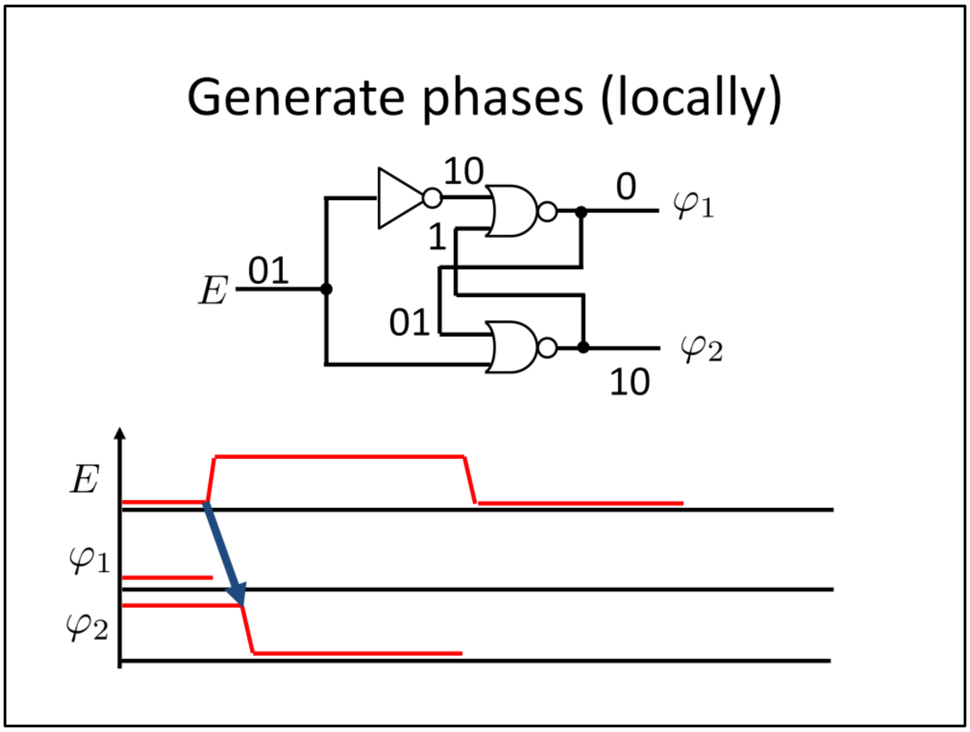
- two signals to distribute
- non-overlap problem, again

from Rabaey et al. Digital Integrated Circuits (2nd ed) p339

Generate phases (locally)

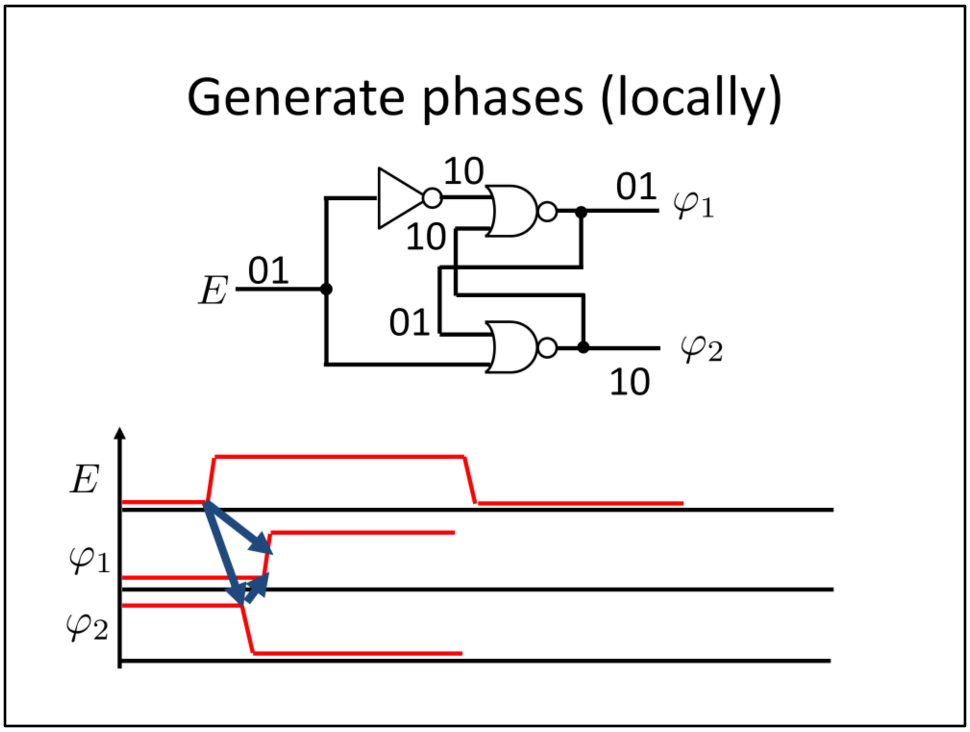


initial steady state

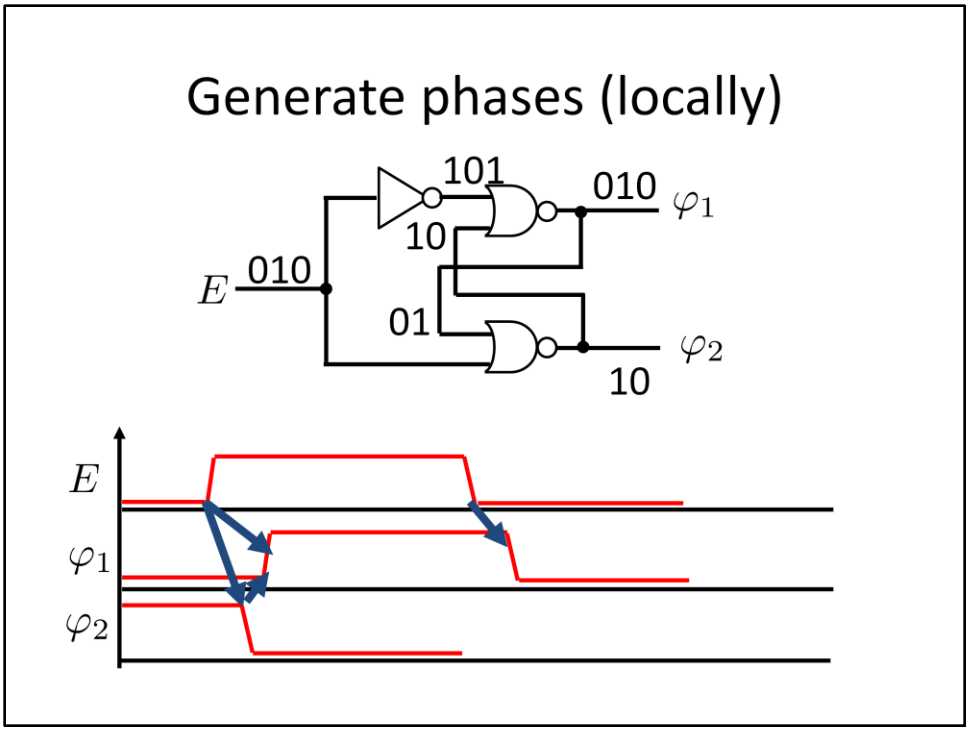


Generate phases (locally)

... still unstable

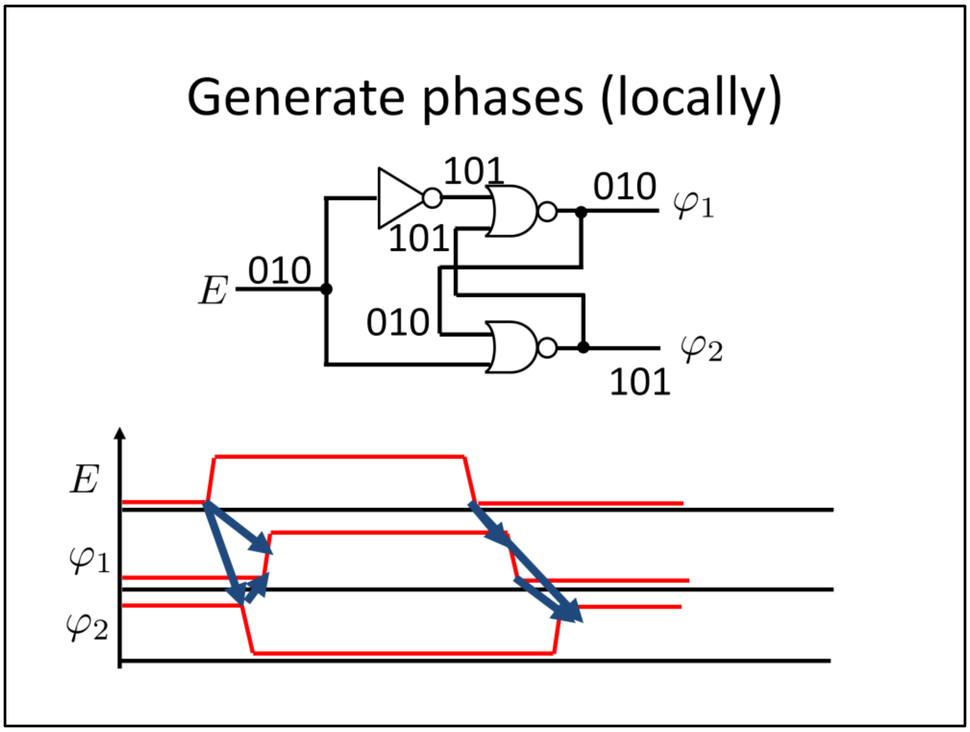

Generate phases (locally)

... stable state after E had a positive transition.
Note the “happened before” constraints for both E-up -> phi_1 up and phi_2 down -> phi_1 up.
This guarantees well-separating between phi_1 and phi_2.



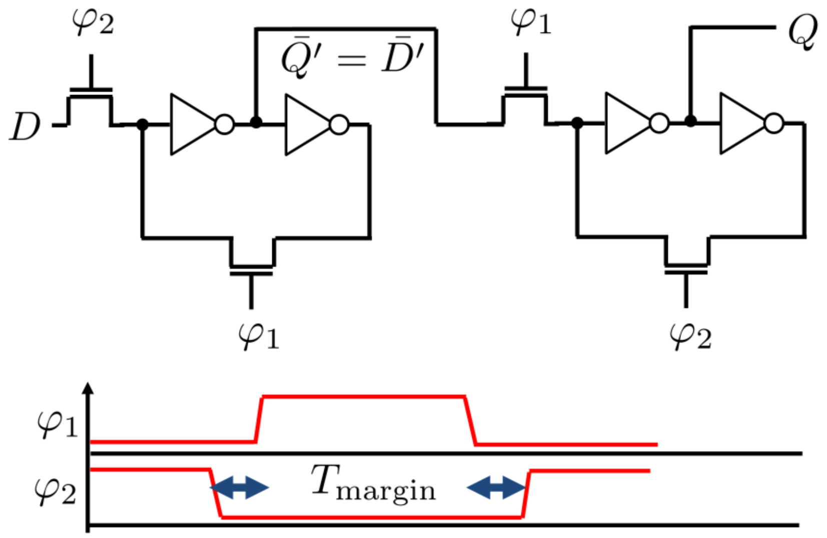
Generate phases (locally)

... still not stable


Generate phases (locally)

stable again. From here on cyclic behavior.
Mind the well-separated phases.

Is this static?



higher T_margin -> better separation between phases and thus less chance of overlap due to e.g. delay variations
However: comes at a price (not only performance): implicit (wire/load) capacitance has to hold charge in the meantime -> decharging might be harmful if T_margin is too long.