

Fundamentals of Computer Systems

Sequential Logic

Stephen A. Edwards

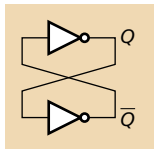
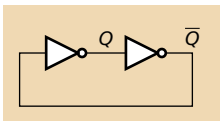
Columbia University

Spring 2012



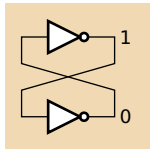
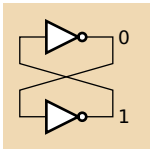
State-Holding Elements

Bistable Elements

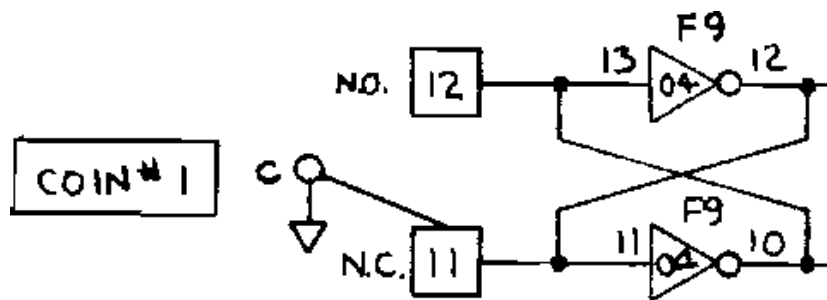


Equivalent circuits; right is more traditional.

Two stable states:



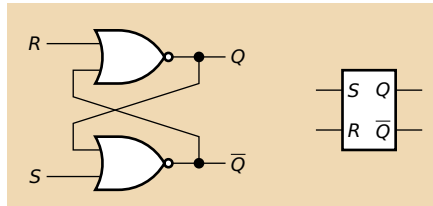
A Bistable in the Wild



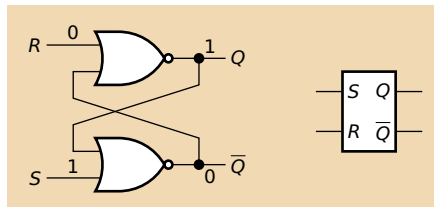
This "debounces" the coin switch.

Breakout, Atari 1976.

SR Latch

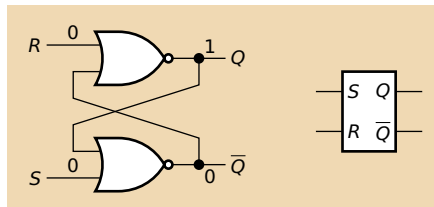


SR Latch



R —
 S —
 Q — Set
 \bar{Q} —

SR Latch



R _____

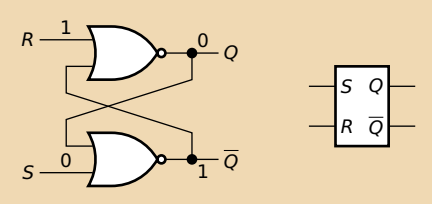
S _____

Q _____

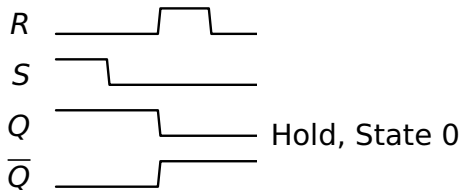
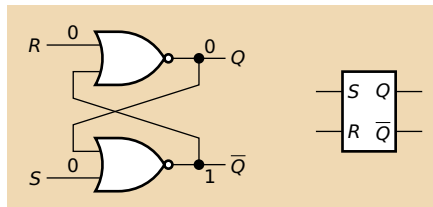
\bar{Q} _____

Hold, State 1

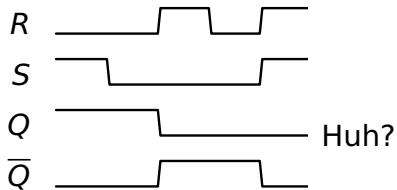
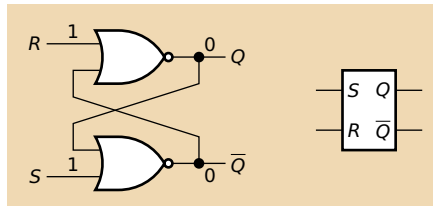
SR Latch



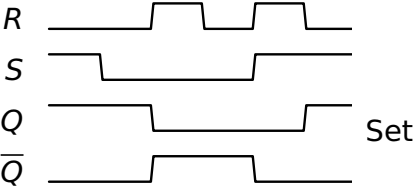
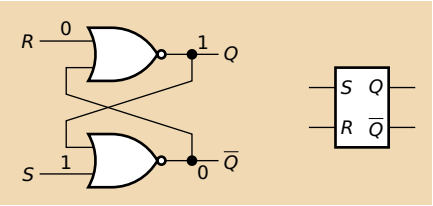
SR Latch



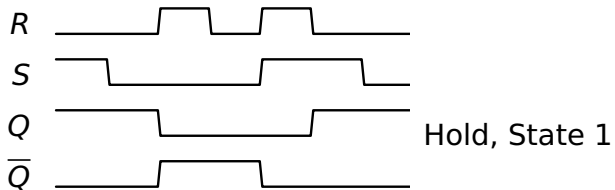
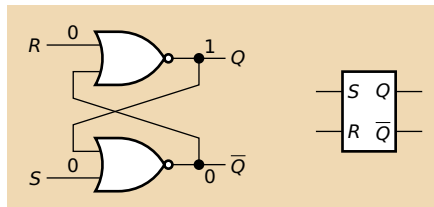
SR Latch



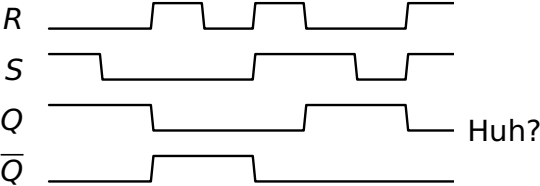
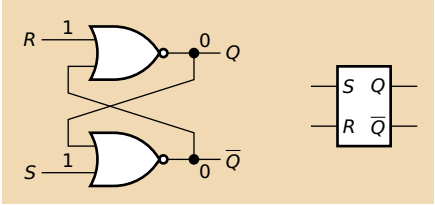
SR Latch



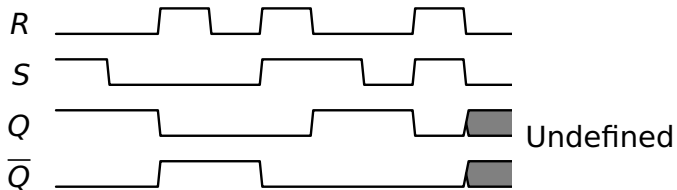
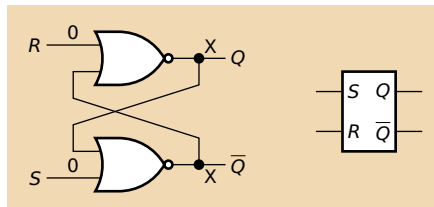
SR Latch



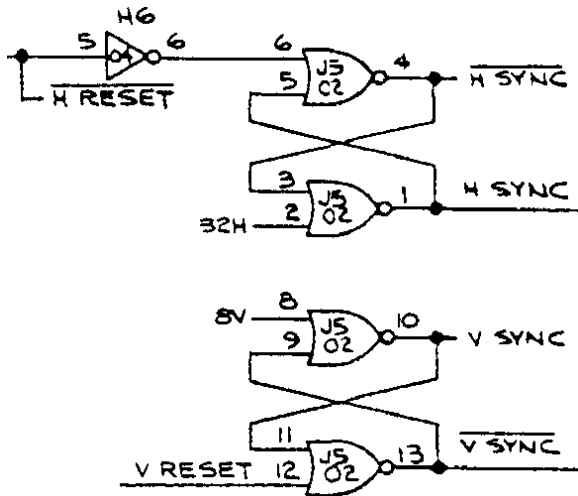
SR Latch



SR Latch



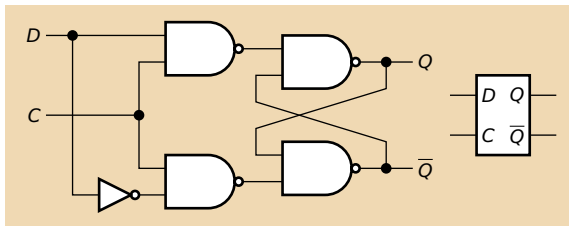
SR Latches in the Wild



Generates horizontal and vertical synchronization waveforms from counter bits.

Stunt Cycle, Atari 1976.

D Latch

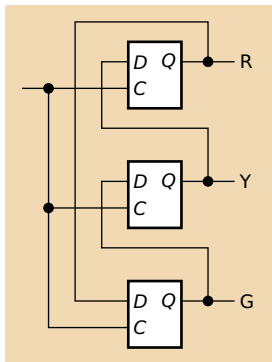


inputs		outputs	
C	D	Q	\bar{Q}
0	X	Q	\bar{Q}
1	0	0	1
1	1	1	0

A Challenge

A simple traffic light controller.

Want the lights to cycle green-yellow-red.



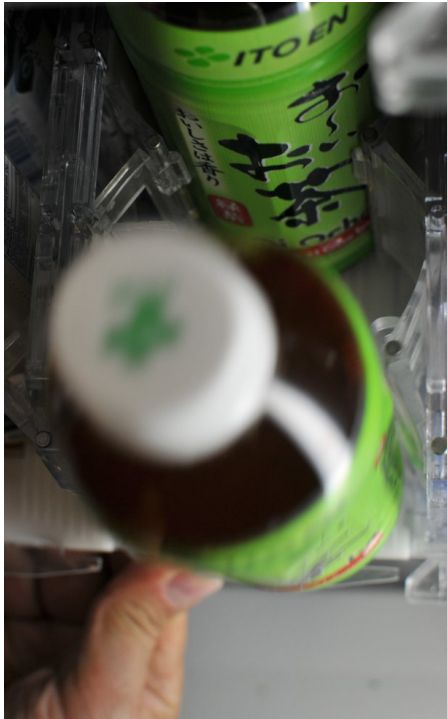
Does this work?





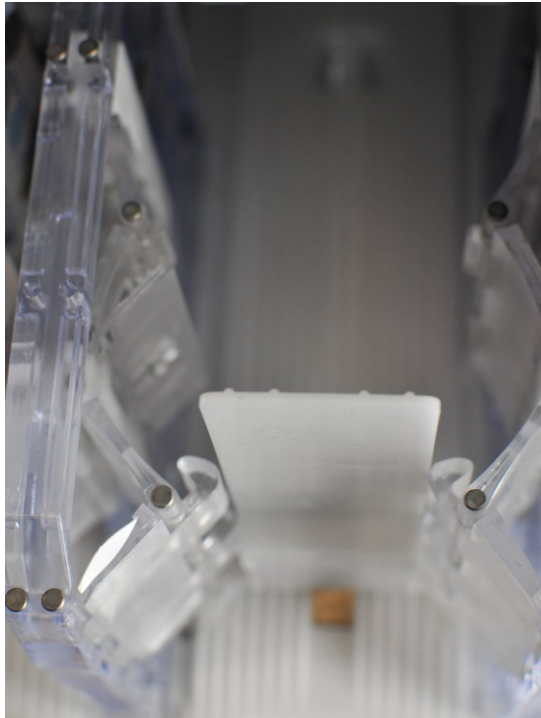


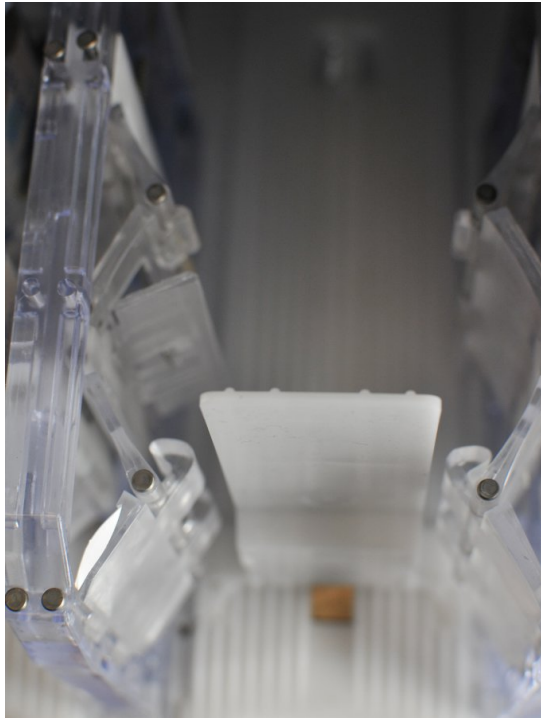


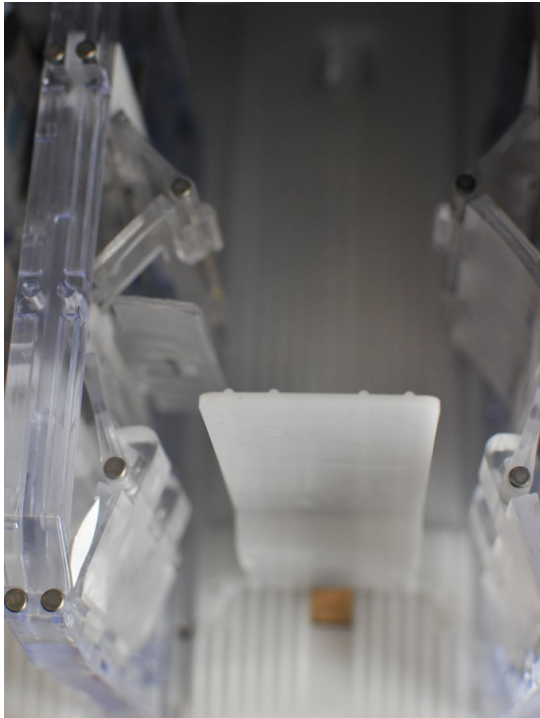


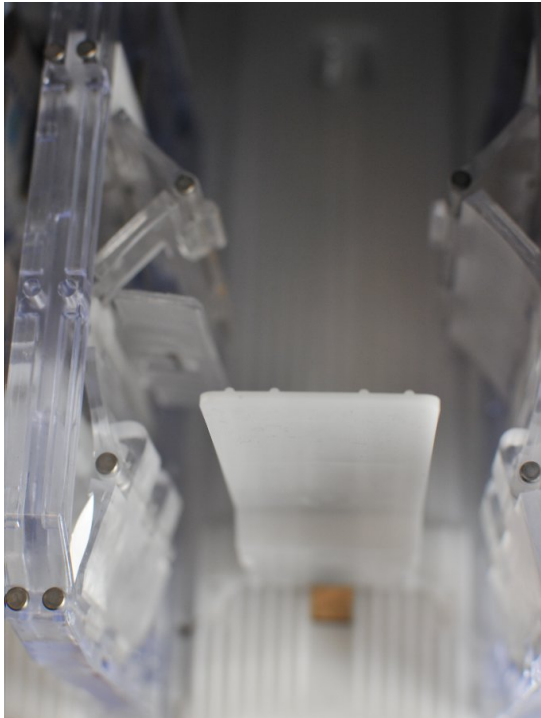


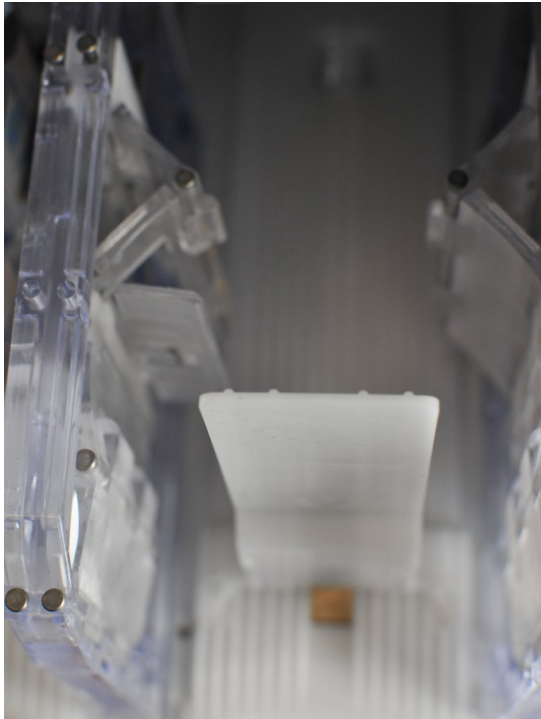




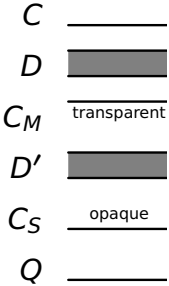
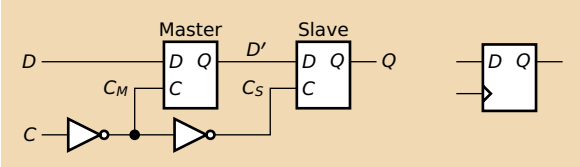




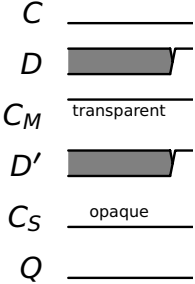
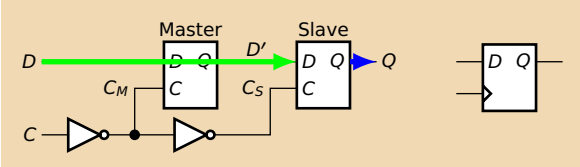




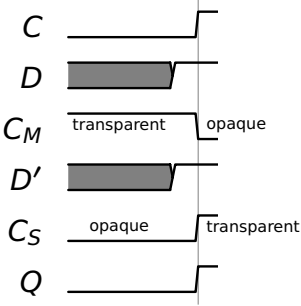
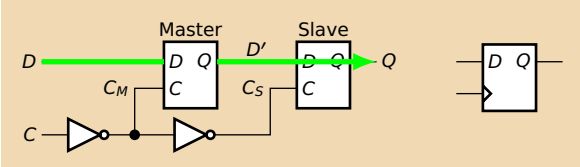
Positive-Edge-Triggered D Flip-Flop



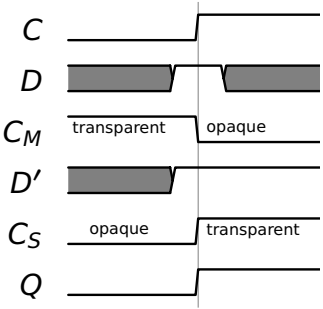
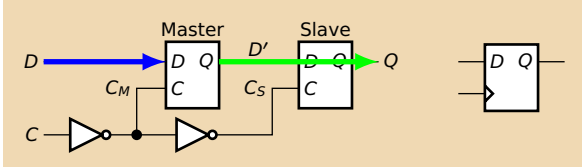
Positive-Edge-Triggered D Flip-Flop



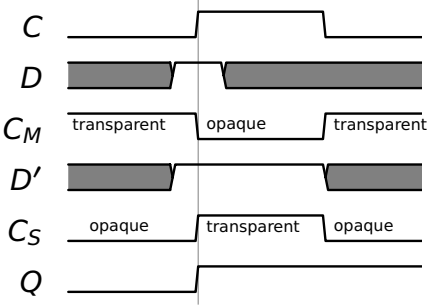
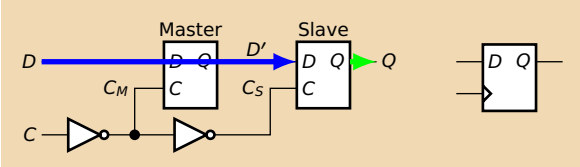
Positive-Edge-Triggered D Flip-Flop



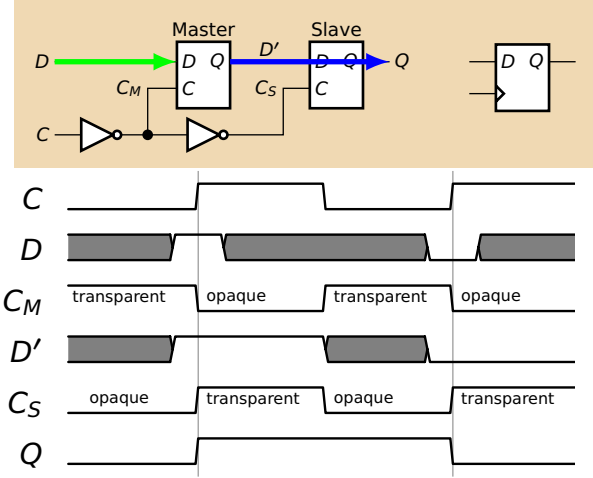
Positive-Edge-Triggered D Flip-Flop



Positive-Edge-Triggered D Flip-Flop

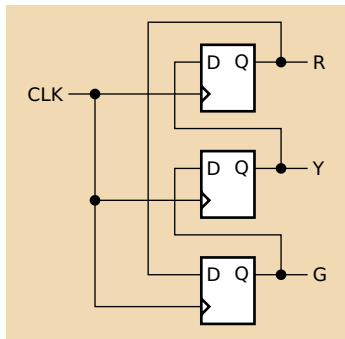


Positive-Edge-Triggered D Flip-Flop



The Traffic Light Controller: A second try

Let's try this again with D flip-flops.



CLK _____

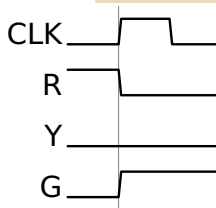
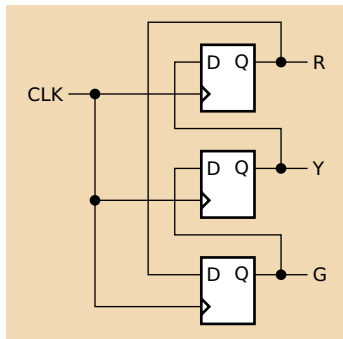
R _____

Y _____

G _____

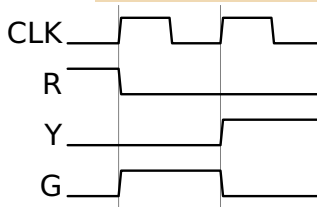
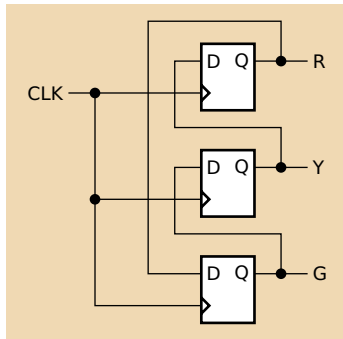
The Traffic Light Controller: A second try

Let's try this again with D flip-flops.



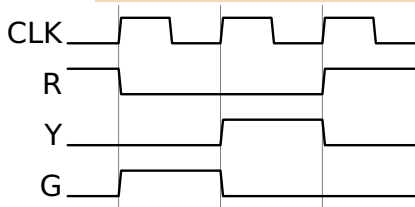
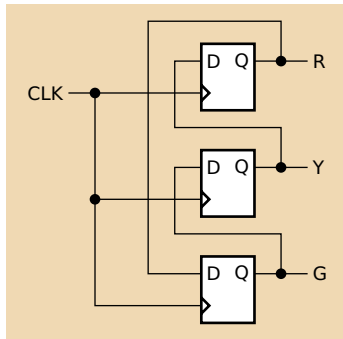
The Traffic Light Controller: A second try

Let's try this again with D flip-flops.



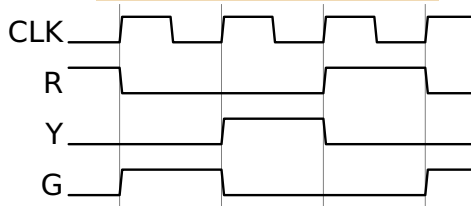
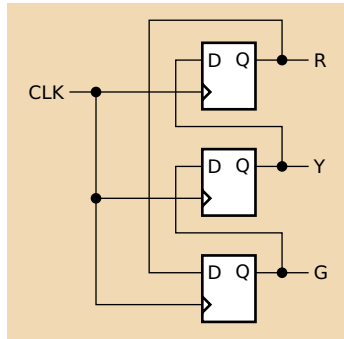
The Traffic Light Controller: A second try

Let's try this again with D flip-flops.

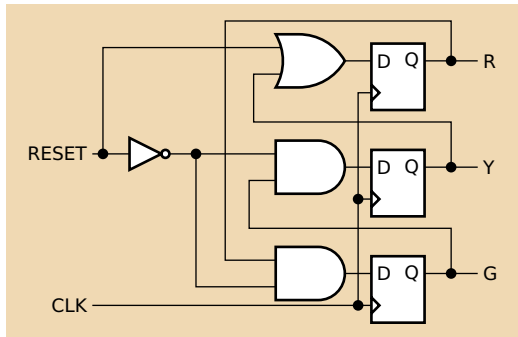


The Traffic Light Controller: A second try

Let's try this again with D flip-flops.



The Traffic Light Controller with Reset



CLK _____

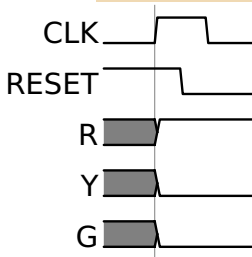
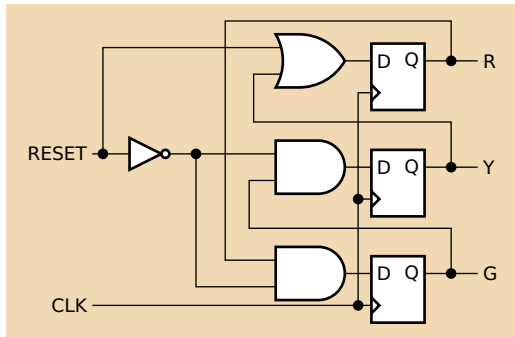
RESET _____

R

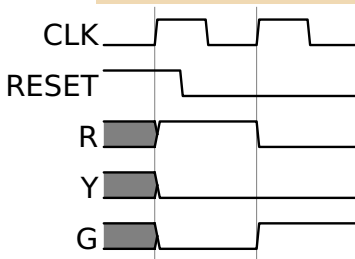
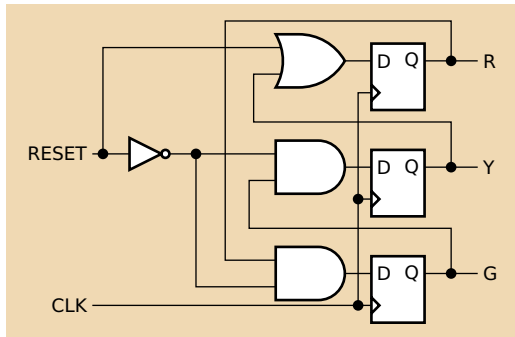
Y

G

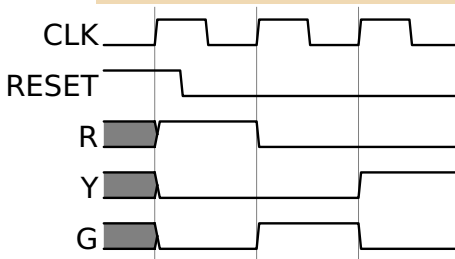
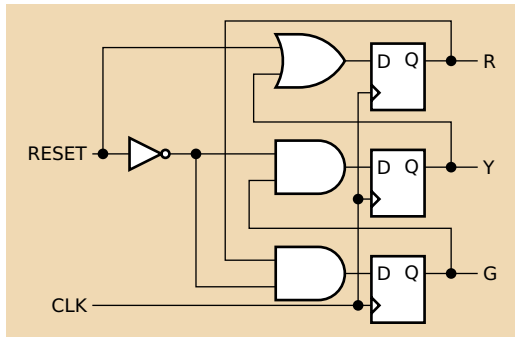
The Traffic Light Controller with Reset



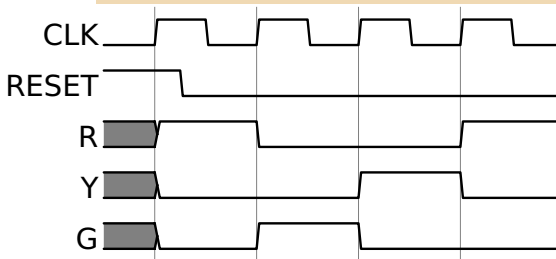
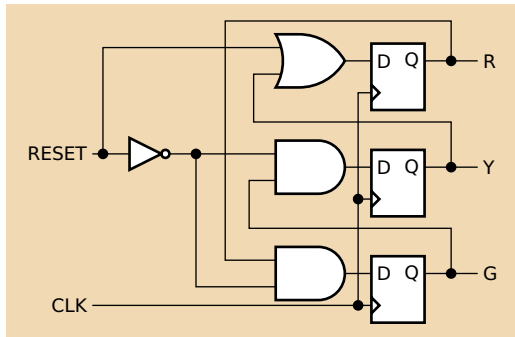
The Traffic Light Controller with Reset



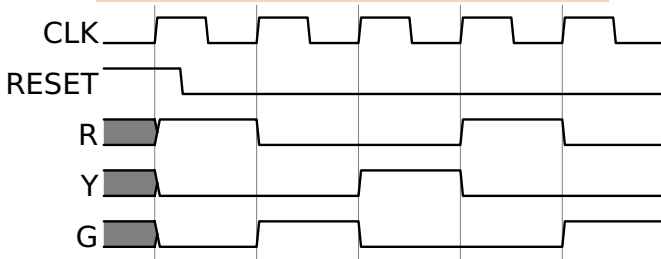
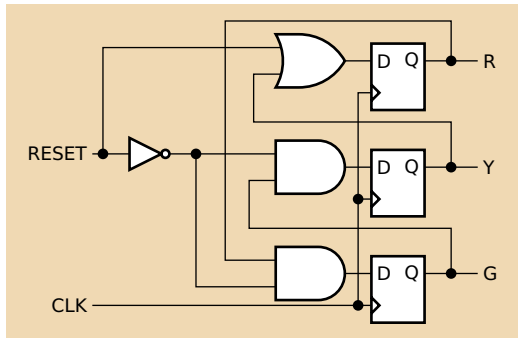
The Traffic Light Controller with Reset



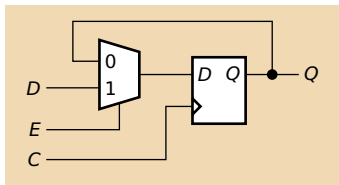
The Traffic Light Controller with Reset



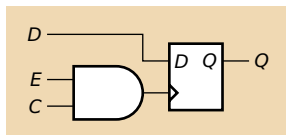
The Traffic Light Controller with Reset



D Flip-Flop with Enable

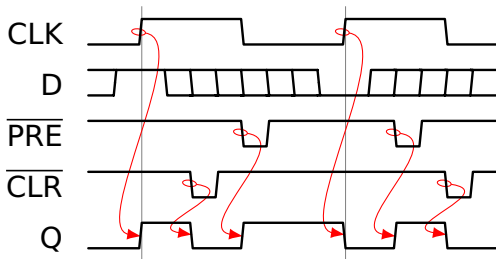
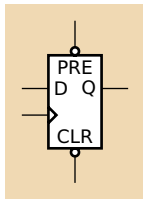


C	E	D	Q
\uparrow	0	X	Q
\uparrow	1	0	0
\uparrow	1	1	1
0	X	X	Q
1	X	X	Q

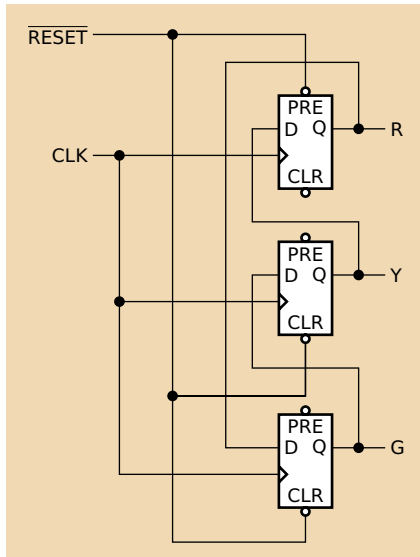


What's wrong with this solution?

Asynchronous Preset/Clear



The Traffic Light Controller w/ Async. Reset

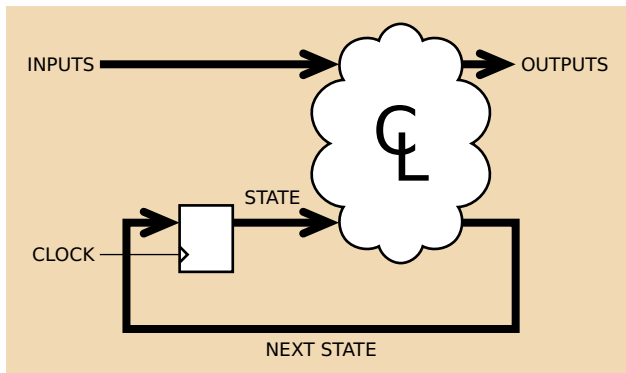


The Synchronous Digital Logic Paradigm

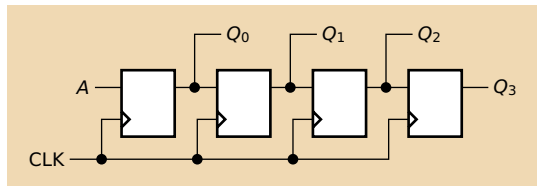
Gates and D
flip-flops only

Each flip-flop
driven by the
same clock

Every cyclic
path contains
at least one
flip-flop

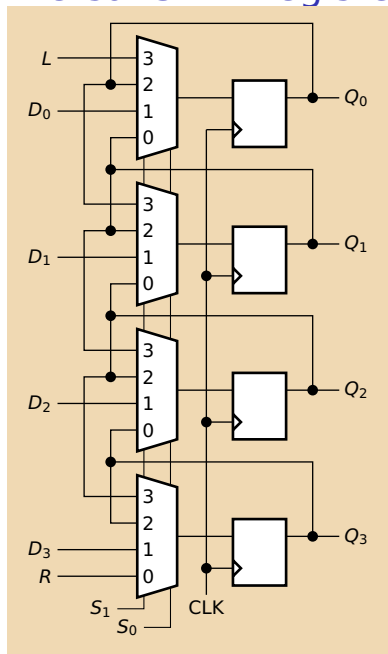


Cool Sequential Circuits: Shift Registers



A	Q_0	Q_1	Q_2	Q_3
0	X	X	X	X
1	0	X	X	X
1	1	0	X	X
0	1	1	0	X
1	0	1	1	0
0	1	0	1	1
0	0	1	0	1
0	0	0	1	0
1	0	0	0	1
0	1	0	0	0

Universal Shift Register

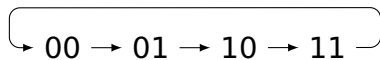


S_1	S_0	Q_3	Q_2	Q_1	Q_0
0	0	R	Q_3	Q_2	Q_1
0	1	D_3	D_2	D_1	D_0
1	0	Q_3	Q_2	Q_1	Q_0
1	1	Q_2	Q_1	Q_0	L

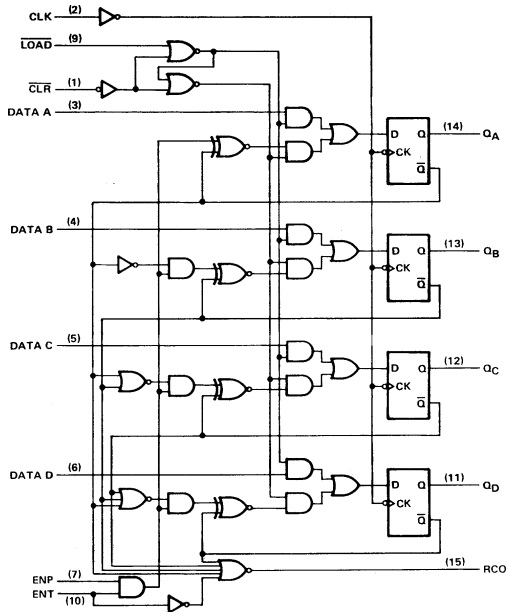
S_1	S_0	Operation
0	0	Shift right
0	1	Load
1	0	Hold
1	1	Shift left

Cool Sequential Circuits: Counters

Cycle through sequences of numbers, e.g.,

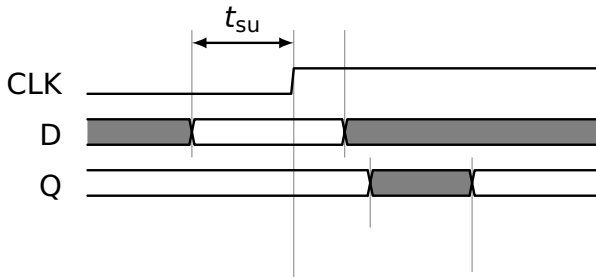


The 74LS163 Synchronous Binary Counter



Flip-Flop Timing

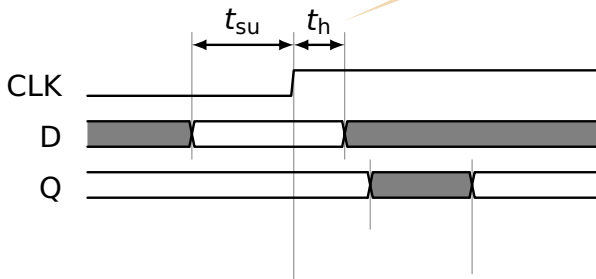
Setup Time: Time before the clock edge after which the data may not change



Flip-Flop Timing

Setup Time: Time before the clock edge after which the data may not change

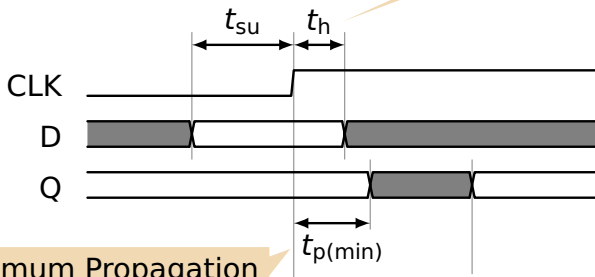
Hold Time: Time after the clock edge after which the data may change



Flip-Flop Timing

Setup Time: Time before the clock edge after which the data may not change

Hold Time: Time after the clock edge after which the data may change

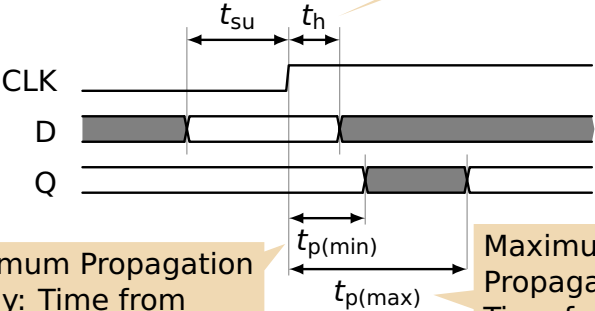


Minimum Propagation Delay: Time from clock edge to when Q might start changing

Flip-Flop Timing

Setup Time: Time before the clock edge after which the data may not change

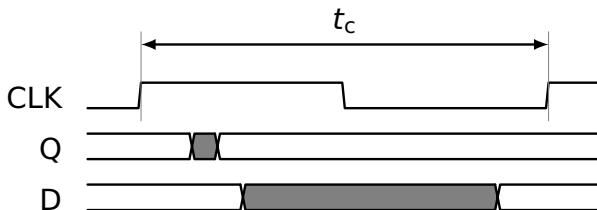
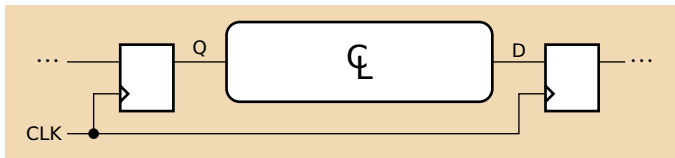
Hold Time: Time after the clock edge after which the data may change



Minimum Propagation Delay: Time from clock edge to when Q might start changing

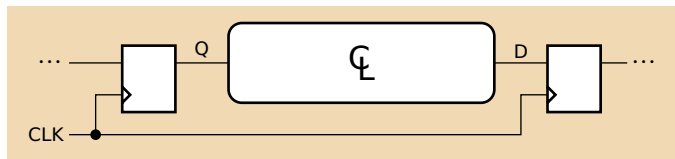
Maximum Propagation Delay: Time from clock edge to when Q guaranteed stable

Timing in Synchronous Circuits

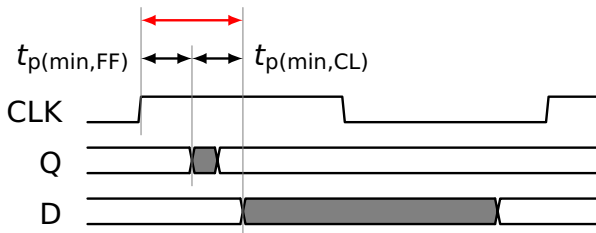


t_c : Clock period. E.g., 10 ns for a 100 MHz clock

Timing in Synchronous Circuits

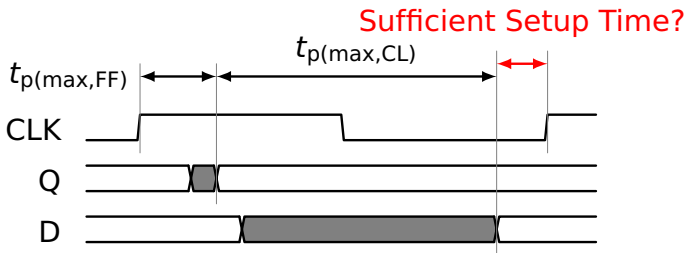
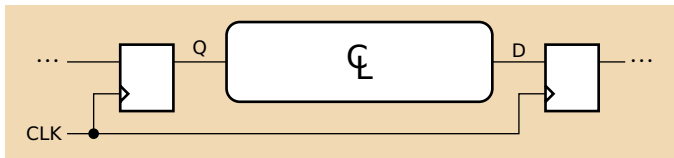


Sufficient Hold Time?



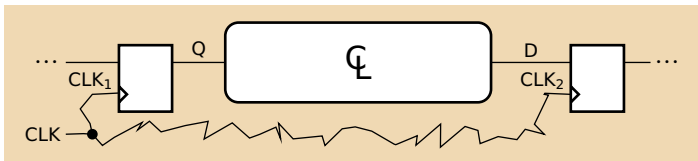
Hold time constraint: how soon after the clock edge can D start changing? Min. FF delay + min. logic delay

Timing in Synchronous Circuits

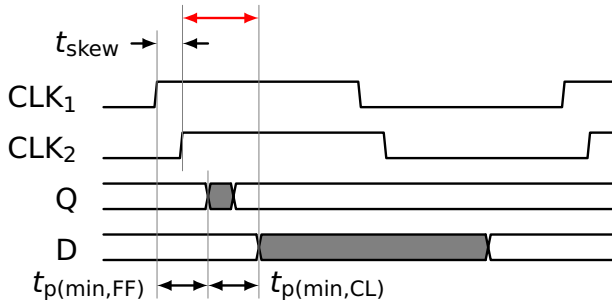


Setup time constraint: when before the clock edge is D guaranteed stable? Max. FF delay + max. logic delay

Clock Skew: What Really Happens

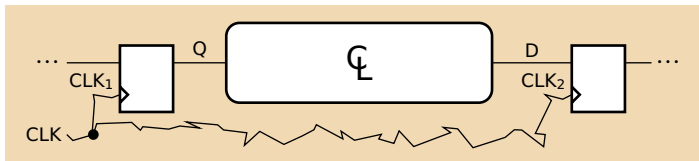


Sufficient Hold Time?

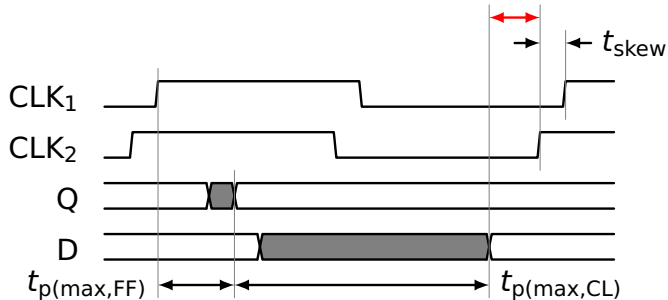


CLK_2 arrives late: clock skew reduces hold time

Clock Skew: What Really Happens



Sufficient Setup Time?



CLK_1 arrives early: clock skew reduces setup time