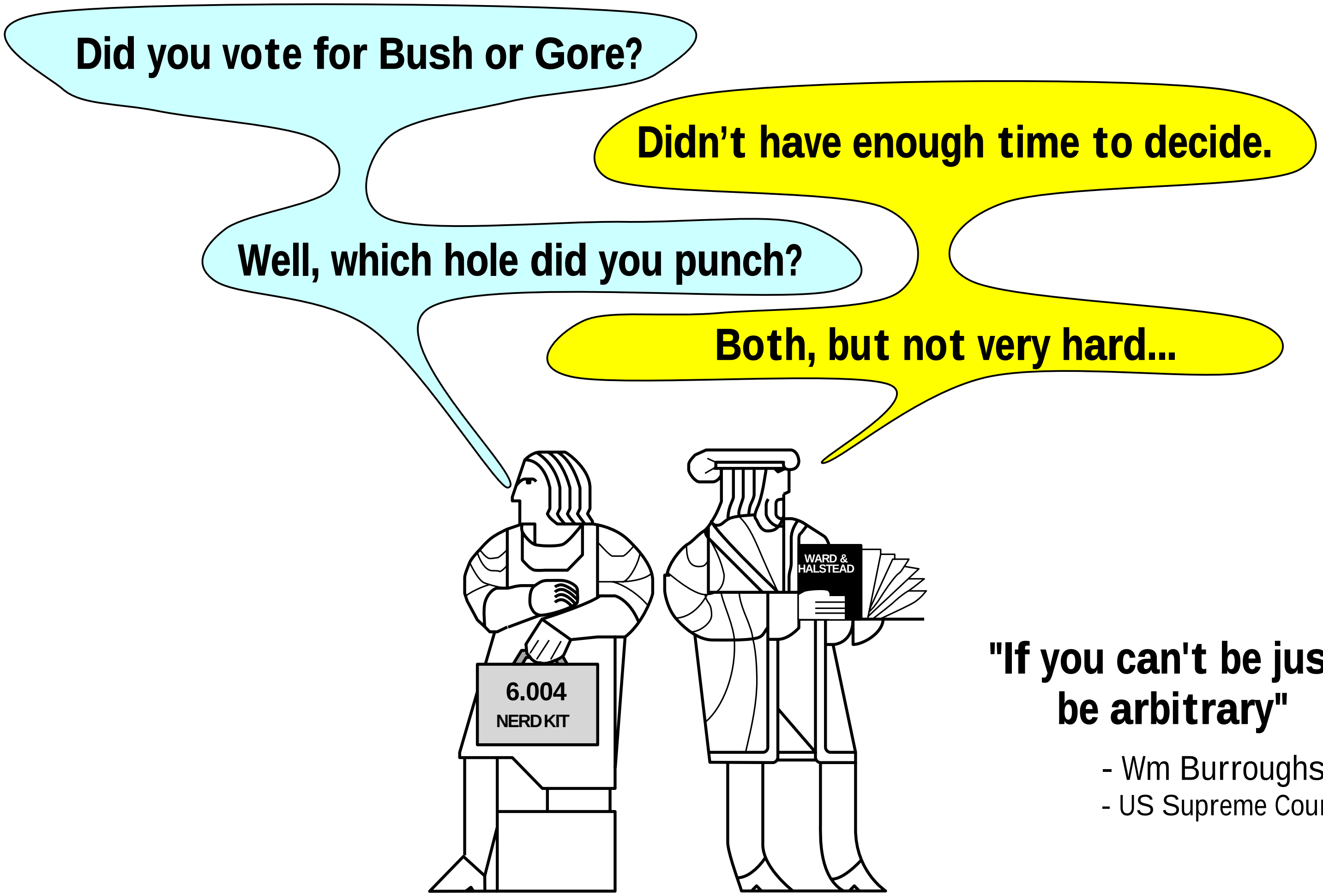
Synchronization, Metastability and Arbitration


Did you vote for Bush or Gore?
Didn't have enough time to decide.
Well, which hole did you punch?
Both, but not very hard...
WARD & HALSTEAD
6.004 NERD KIT

"If you can't be just,
be arbitrary"

- Wm Burroughs, *Naked Lunch*
- US Supreme Court 12/00

Handouts: Lecture Slides

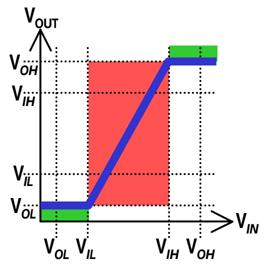
The Importance of being Discrete

We avoid possible errors by disciplines that avoid asking the tough questions – using a *forbidden zone* in both voltage and time dimensions:

Digital Values:

Problem: Distinguishing voltages representing “1” from “0”

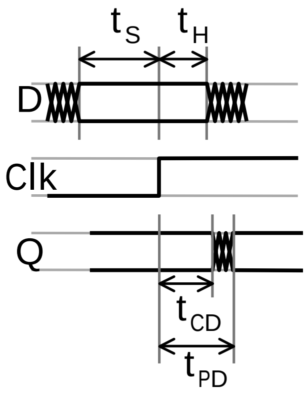
Solution: *Forbidden Zone:* avoid using similar voltages for “1” and “0”



Digital Time:

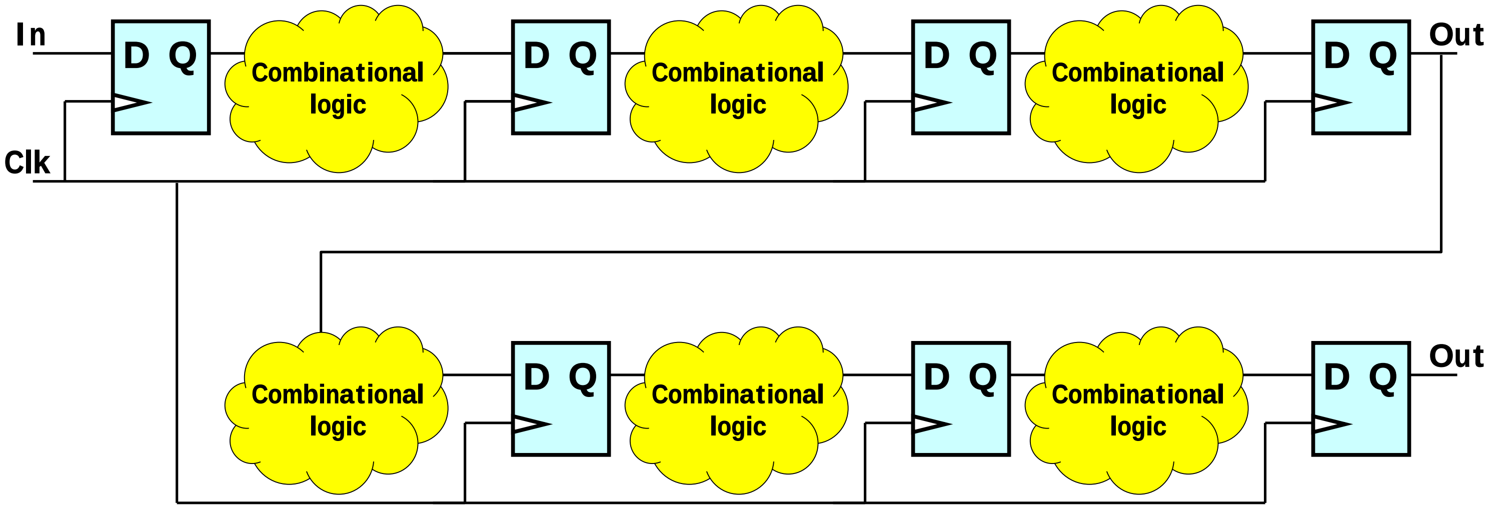
Problem: “Which transition happened first?” questions

Solution: Dynamic Discipline: avoid asking such questions in close races



If we follow these simple rules...

Can we guarantee that our system will always work?

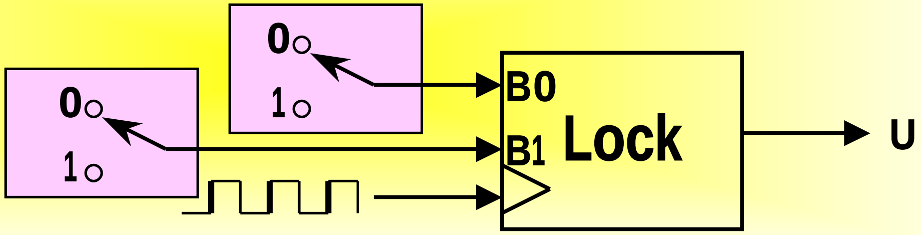


With careful design we can make sure that the dynamic discipline is obeyed everywhere*...

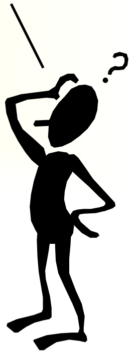
* well, *almost* everywhere...

The world doesn’t run on our clock!

What if each button input is an asynchronous 0/1 level?

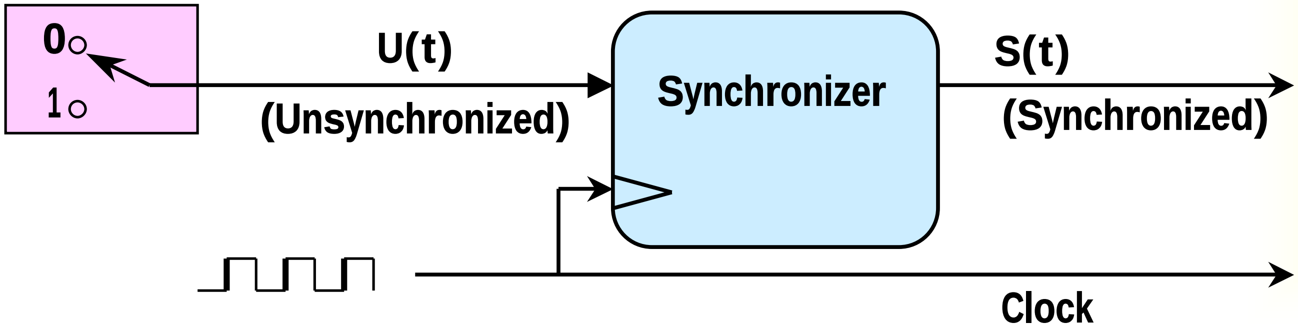


But what
About the
Dynamic
Discipline?



To build a system with asynchronous inputs, we have to break the rules: *we cannot guarantee that setup and hold time requirements are met at the inputs!*

So, lets use a “synchronizer” at each input:

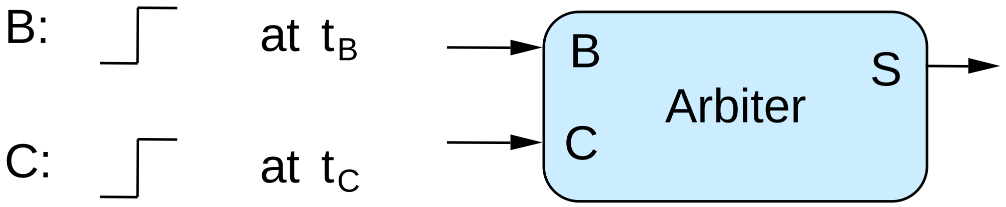


Valid except
for brief periods
following active
clock edges

The Asynchronous Arbiter:

a classic problem

UNSOLVABLE



For NO finite value of t_E and t_D is this spec realizable, even with reliable components!

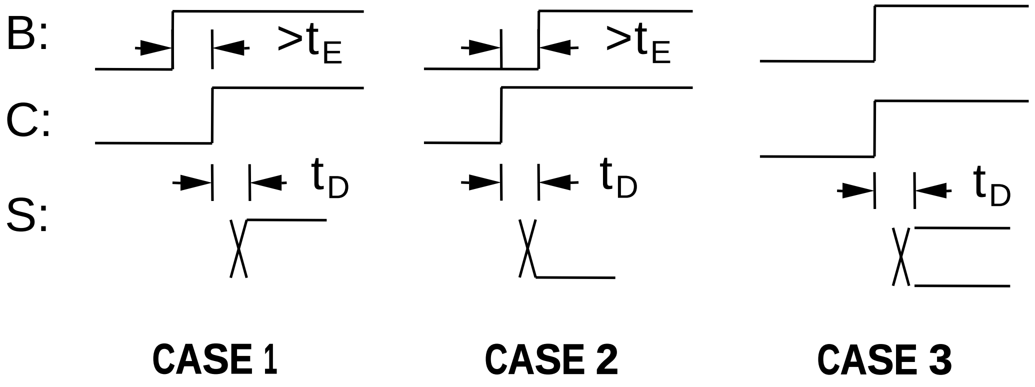
Arbiter specifications:

- finite t_D (decision time)
- finite t_E (allowable error)
- value of S at time t_C+t_D:

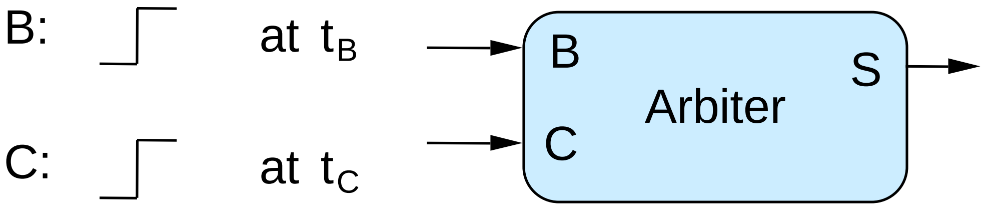
  1 if $t_B < t_C - t_E$

  0 if $t_B > t_C + t_E$

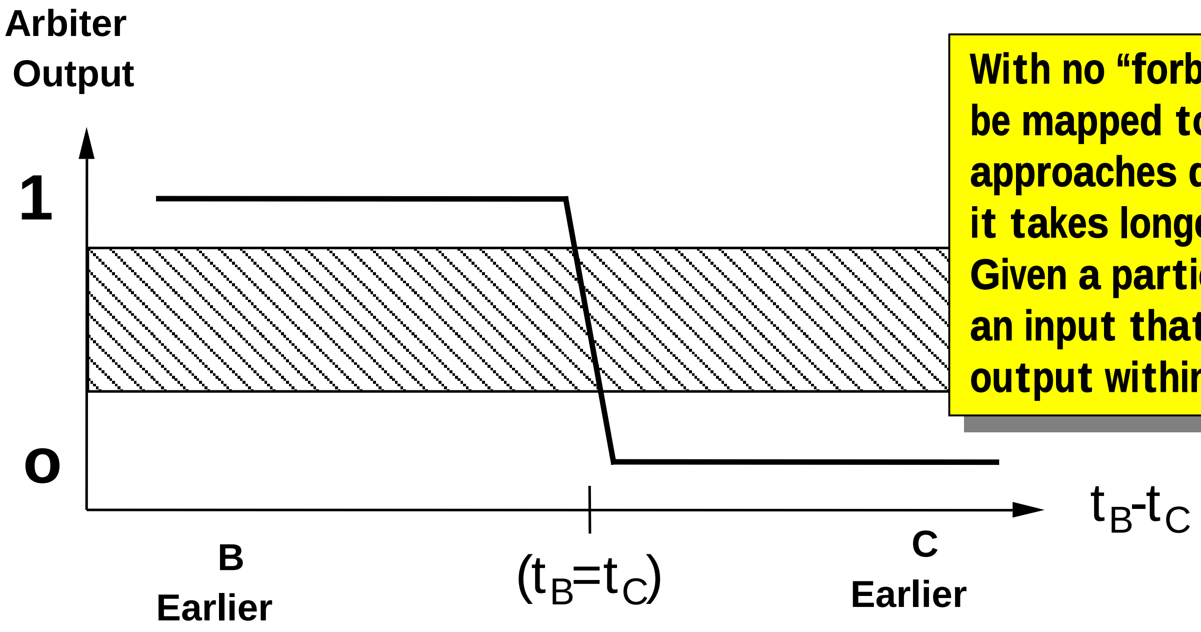
  0, 1 otherwise



Violating the Forbidden Zone



Issue: Mapping the *continuous* variable ($t_B - t_C$) onto the *discrete* variable S <u>in bounded time</u>.

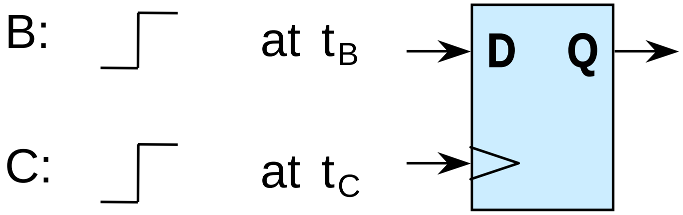


With no "forbidden zone," all inputs have to be mapped to a valid output. As the input approaches discontinuities in the mapping, it takes longer to determine the answer. Given a particular time bound, you can find an input that won't be mapped to a valid output within the allotted time.

Unsolvable?

that <u>can't</u> be true...

Lets just use a D Flip Flop:



We're lured by the digital abstraction into assuming that Q must be either 1 or 0. But lets look at the input latch in the flip flop whe B and C change at about the same time...

DECISION TIME **is T_{PD} of flop.**

ALLOWABLE ERROR **is max(t_{SETUP}, t_{HOLD})**

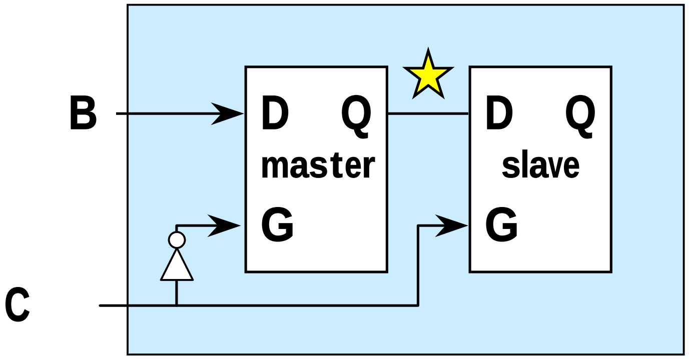
Our logic:

T_{PD} after T_C, we'll have

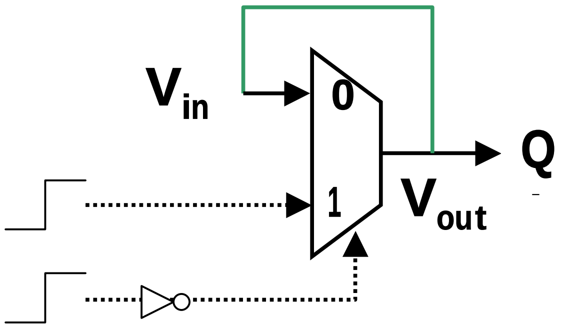
Q=0 iff $t_B + t_{SETUP} < t_C$

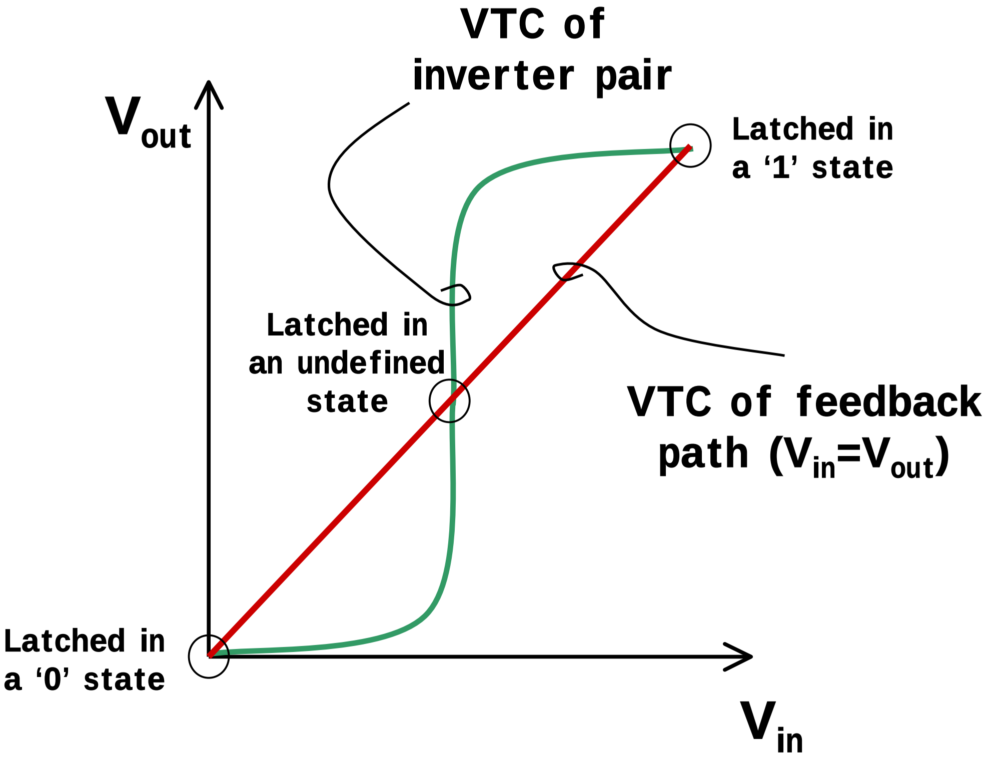
Q=1 iff $t_C + t_{HOLD} < t_B$

Q=0 or 1 otherwise.



The Mysterious Metastable State





VTC of inverter pair

Latched in a '1' state

Latched in an undefined state

VTC of feedback path (V_{in}=V_{out})

Latched in a '0' state

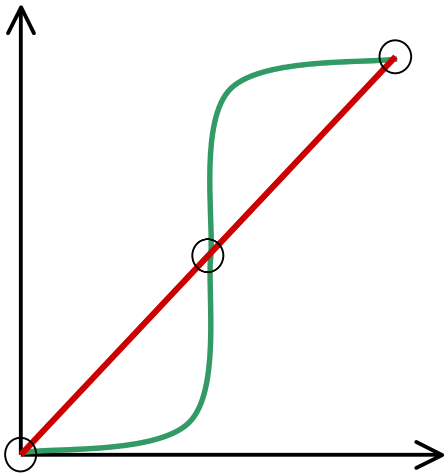
Recall that the latch output is the solution to two simultaneous constraints:

1. The VTC of 2 cascaded gates; and
2. $V_{in} = V_{out}$

In addition to our expected stable solutions, we find an unstable equilibrium in the forbidden zone called the "Metastable State"

Metastable State: Properties

1. It corresponds to an *invalid* logic level – the switching threshold of the device.
2. Its an *unstable* equilibrium; a small perturbation will cause it to accelerate toward a stable 0 or 1.
3. It will settle to a valid 0 or 1… eventually.
4. BUT – depending on how close it is to the $V_{in}=V_{out}$ "fixed point" of the device – it may take arbitrarily long to settle out.
5. EVERY bistable system exhibits at least one metastable state!



EVERY bistable system?

Yep, every last one.

Coin flip??

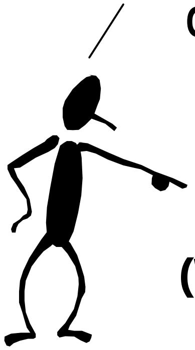
Could land on edge.

Horse race??

Photo finish.

Presidential Election??

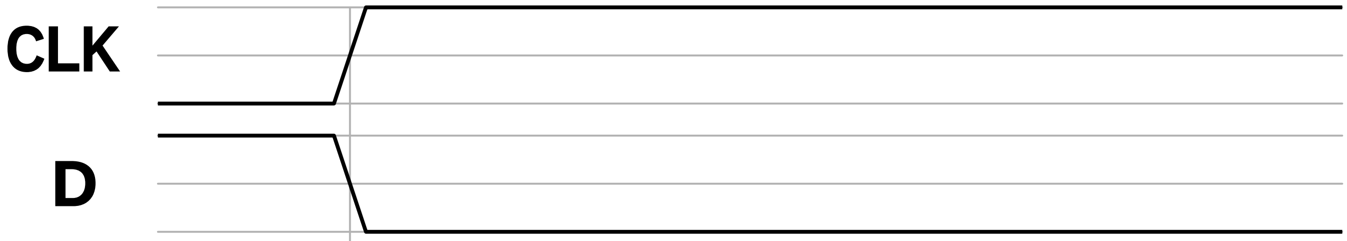
(Where's this twit been hiding???)



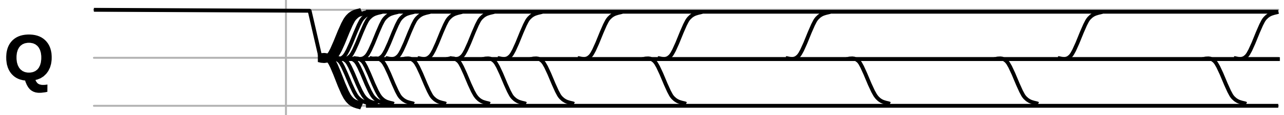
Observed Behavior:

typical metastable symptoms

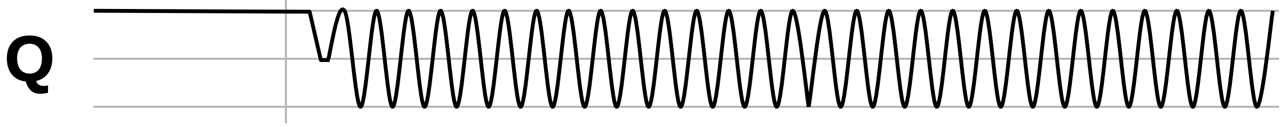
Following a clock edge on an asynchronous input:

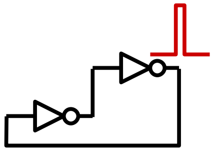


We may see exponentially-distributed metastable intervals:

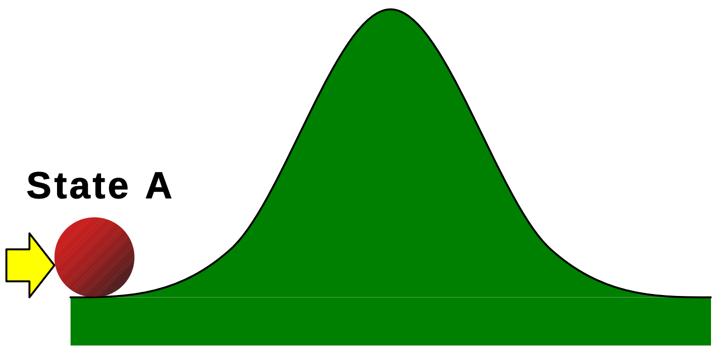


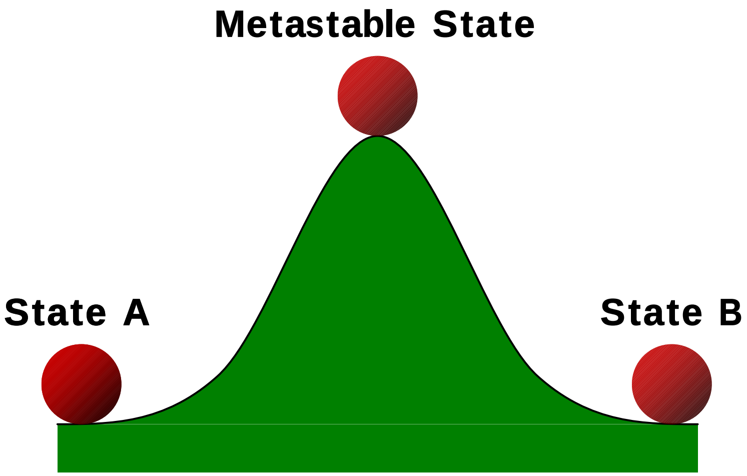
Or periods of high-frequency oscillation (if the feedback path is long):





Mechanical Metastability





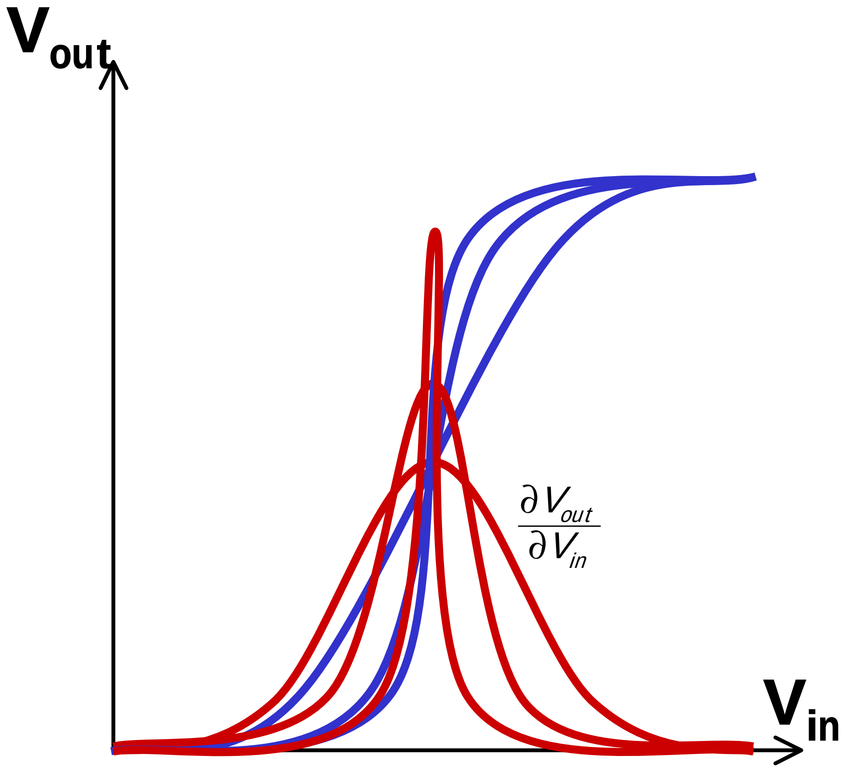
If we launch a ball up a hill we expect one of 3 possible outcomes:

a) Goes over
b) Rolls back
c) Stalls at the apex

That last outcome is not stable.

- a gust of wind
- Brownian motion
- it doesn't take much

How do balls relate to digital logic?



Our hill is simply the derivative of the VTC (Voltage Transfer Curve).

Notice that the higher the gain thru the transition region, the steeper the peak of the hill... making it harder to get into a metastable state.

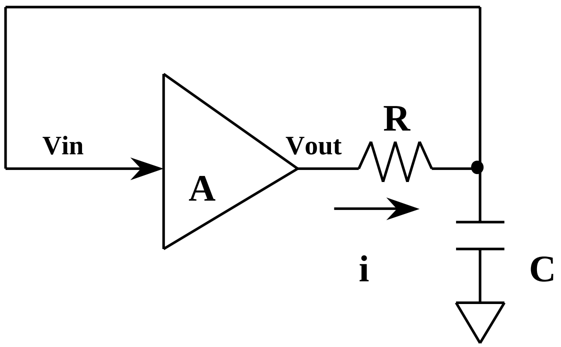
We can decrease the probability of getting into the metastable state, but we can't eliminate it…

The Metastable State:

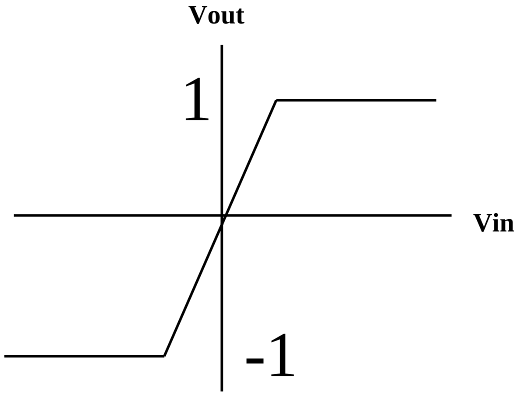
Why is it an inevitable risk of synchronization?

- Our active devices always have a fixed-point voltage, V_M, such that $V_{IN}=V_M$ implies $V_{OUT} = V_M$
- Violation of dynamic discipline puts our feedback loop at some voltage V_0 near V_M
- The rate at which V progresses toward a stable "0" or "1" value is proportional to $(V - V_M)$
- The time to settle to a stable value depends on $(V_0 - V_M)$; its theoretically infinite for $V_0 = V_M$
- Since there's no lower bound on $(V_0 - V_M)$, there's no upper bound on the settling time.
- Noise, uncertainty complicate analysis (but don't help).

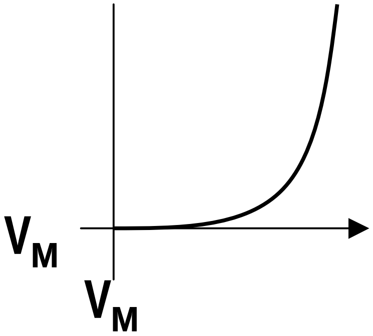
Sketch of analysis…



We can model our cascaded gates as an amplifier with gain A and saturation at +1, -1



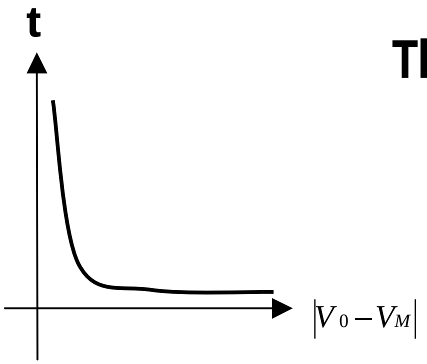
Evolution of voltage is an exponential whose time constant is determined by RC and A



$$V(t) = (V_0 - V_M)e^{\frac{(A-1)}{RC}t}$$

(V_M is the fixed-point voltage; V_0 is the initial voltage)

$$|V_0 - V_M| = \frac{1}{A}e^{-\frac{t}{\tau}}$$



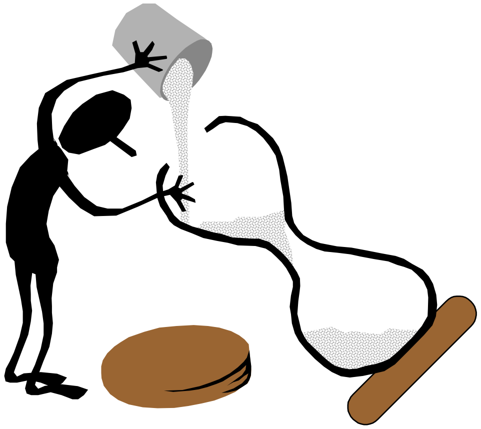
The magnitude of (V_0 – V_M) that results in a settling time of t is a negative exponential in t

Failure Probabilities vs Delay

Making conservative assumptions about the distribution of V_0 and system time constants, and assuming a 100 MHz clock frequency, we get results like the following:

Delay	P(Metastable)	Average time between failures
31 ns	$3x10^{-16}$	1 year
33.2 ns	$3x10^{-17}$	10 years
100 ns	10^{-45}	10^{30} years!

[For comparision:
Age of oldest hominid fossil: $5x10^{6}$ years
Age of earth: $5x10^{9}$ years]



Lesson: Allowing a bit of settling time is an easy way to avoid metastable states in practice!

The Metastable State:

a brief history

Antiquity: Early recognition

Buriden's Ass, and other fables…

Denial: Early 70s

Widespread disbelief. Early analyses documenting inevitability of problem rejected by skeptical journal editors.

Folk Cures: 70s-80s

Popular pastime: Concoct a "Cure" for the problem of "synchronization failure". Commercial synchronizer products.

Reconciliation: 80s-90s

Acceptance of the reality: synchronization takes time. Interesting special case solutions.

Ancient Metastability

Metastability is the occurrence of a persistent invalid output… an unstable equilibria.

The idea of Metastability is not new:

The Paradox of Buridan's Ass

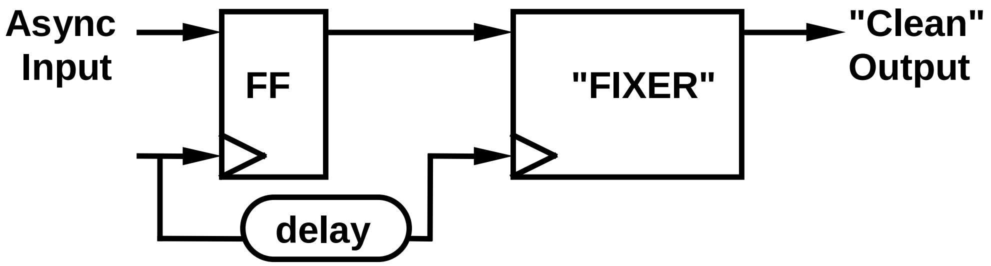
Buridan, Jean (1300-58), French Scholastic philosopher, who held a theory of determinism, contending that the will must choose the greater good. Born in Bethune, he was educated at the University of Paris, where he studied with the English Scholastic philosopher William of Ockham (whom you might recall from his razor business). After his studies were completed, he was appointed professor of philosophy, and later rector, at the same university. Buridan is traditionally, but probably incorrectly, associated with a philosophical dilemma of moral choice called "Buridan's ass."

In the problem an ass starves to death between two alluring bundles of hay because it does not have the will to decide which one to eat.

Folk Cures

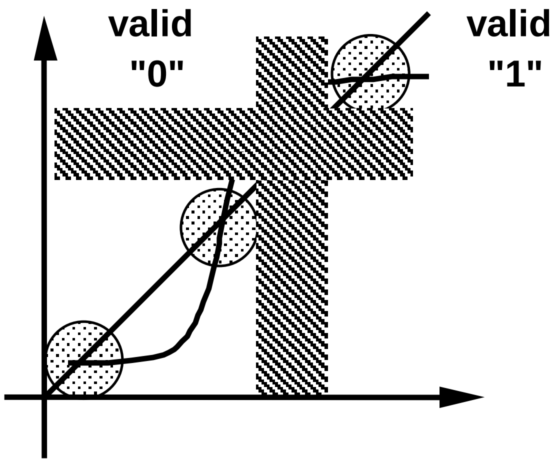
the "perpetual motion machine" of digital logic

Bad Idea # 1: Detect metastable state & Fix



Bug: detecting metastability is itself subject to metastable states, i.e., the "fixer" will fail to resolve the problem in bounded time.

Bad Idea #2: Define the problem away by making metastable point a valid output



Bug: the memory element will flip some valid "0" inputs to "1" after a while.

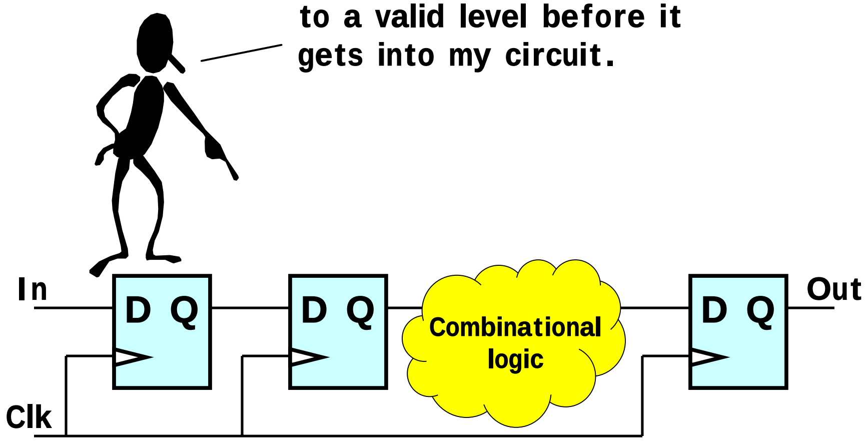
Many other bad ideas – involving noise injection, strange analog circuitry, … have been proposed.

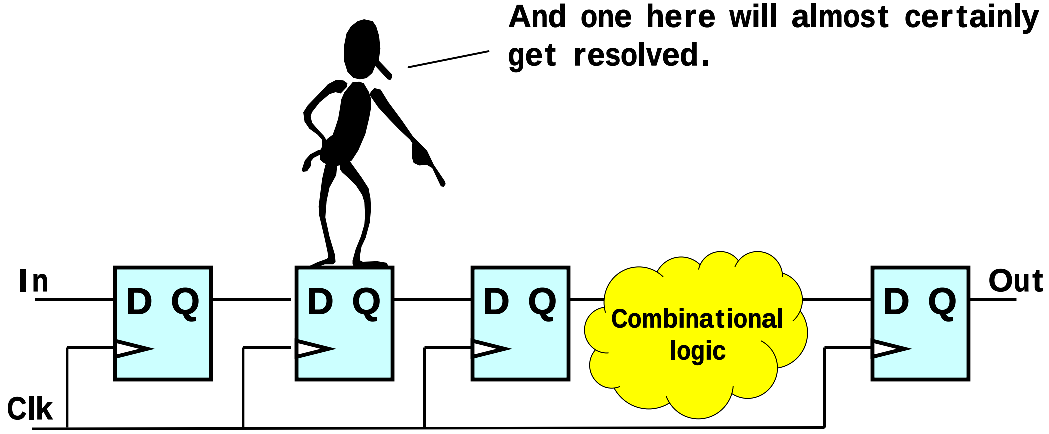
Modern Reconciliation:

delay buys reliability

Synchronizers, extra flip flops between the asynchronous input and your logic, are the best insurance against metastable states.

The higher the clock rate, the more synchronizers should be considered.


A metastable state here will probably resolve itself to a valid level before it gets into my circuit.


And one here will almost certainly get resolved.

There’s no easy solution

… so, embrace the confusion.

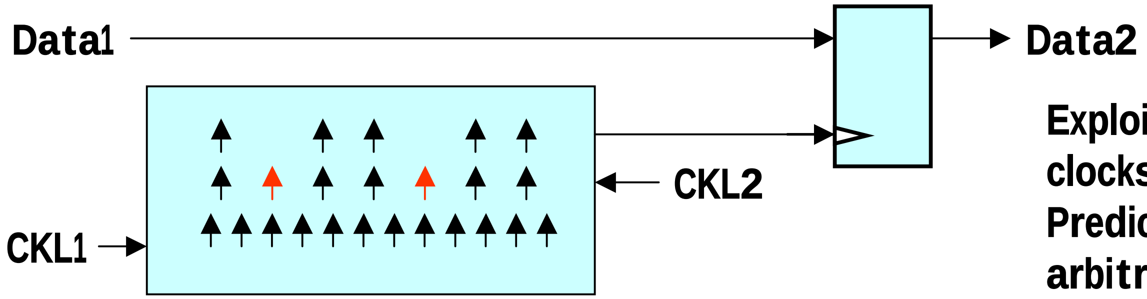
"Metastable States":

- <u>Inescapable consequence</u> of bistable systems
- Eventually a metastable state will resolve itself to valid binary level.
- However, the recovery time is UNBOUNDED ... but influenced by parameters (gain, noise, etc)
- Probability of a metastable state falls off EXPONENTIALLY with time -- modest delay after state change can make it very unlikely.

Our STRATEGY; since we can’t eliminate metastability, we will do the best we can to keep it from contaminating our designs

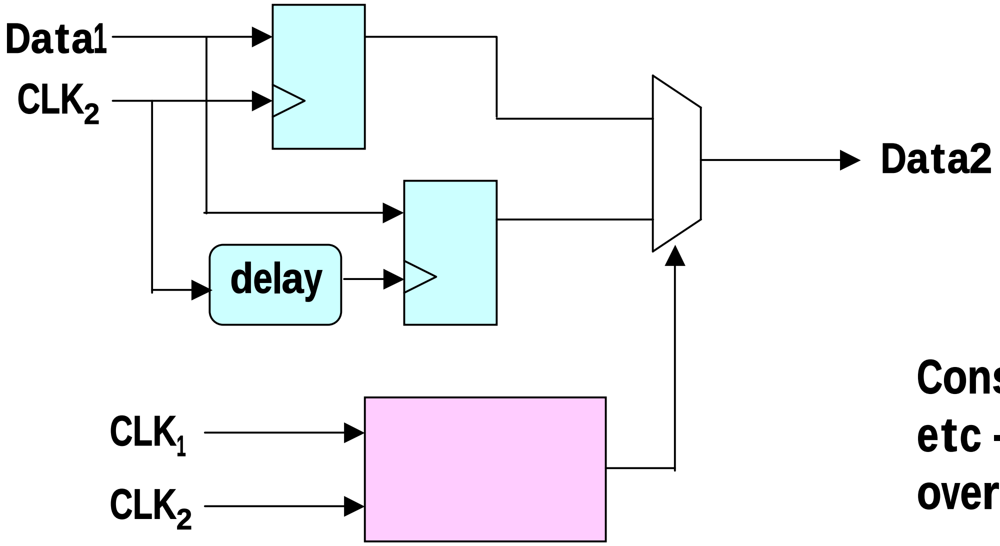
Interesting Special Case Hacks

Predictive periodic synchronization:



Exploits fact that, given 2 periodic clocks, “close calls” are predictable. Predicts, and solves in advance, arbitration problems (thus eliminating cost of delay)

Mesochronous communication:

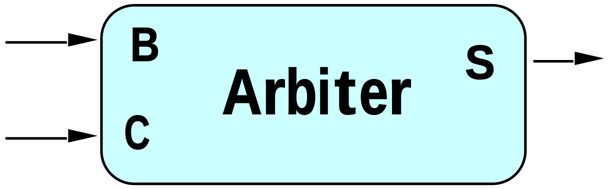


For systems with unsychronized clocks of same nominal frequency. Data goes to two flops clocked a half period apart; one output is bound to be “clean”. An observer circuit monitors the slowly-varying phase relationship between the clocks, and selects the clean output via a lenient MUX.

Constraints on clock timing – periodicity, etc – can often be used to “hide” time overhead associated with synchronization.

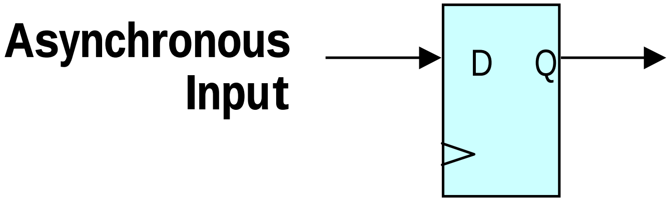
Things we CAN'T build

1. Bounded-time Asynchronous Arbiter:



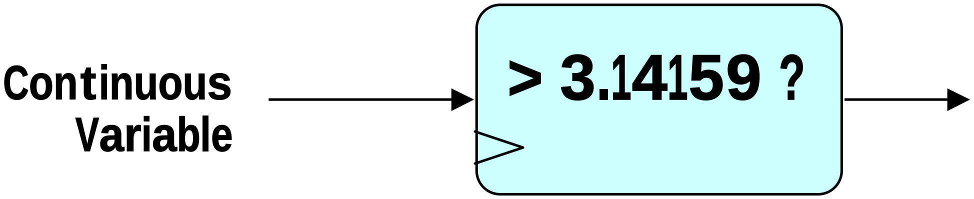
S=0 iff B edge first, 1 iff C edge first,
1 or 0 if nearly coincident
S valid **after t_{pd}** following (either) edge

2. Bounded-time Synchronizer:



Output = D at active clock edge, either 1 or 0
iff D invalid near clock edge
Q valid **after t_{pd}** following active clock edge

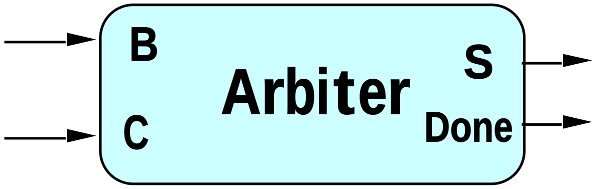
3. Bounded-time Analog Comparator:



0 or 1,
finite t_{pd}

Some things we CAN build

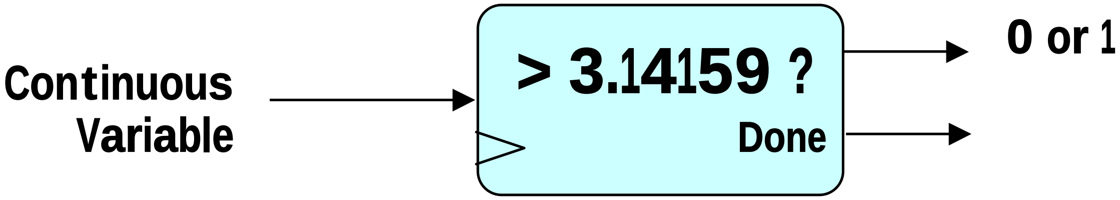
1. Unbounded-time Asynchronous Arbiter:



S valid when Done=1; unbounded time.

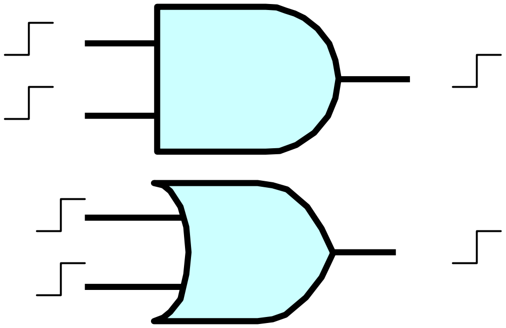
S=0 iff B edge first, 1 iff C edge first,
1 or 0 if nearly coincident

2. Unbounded-time Analog Comparator:



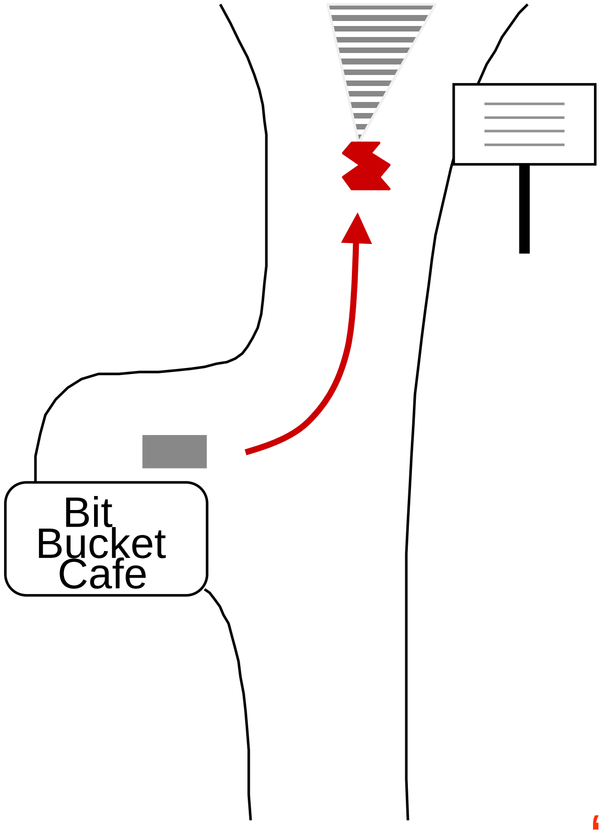
After arbitrary interval, decides whether input at time of last active clock edge was above/below threshold.

3. Bounded-time combinational logic:



Produce an output transition within a fixed propagation delay of first (or second) transition on the input.

Every-day Metastability - I



Ben Bitdiddle tries the famous "6.004 defense":

Ben leaves the Bit Bucket Café and approaches fork in the road. He hits the barrier in the middle of the fork, later explaining "I can't be expected to decide which fork to take in bounded time!".

Is the accident Ben's fault?

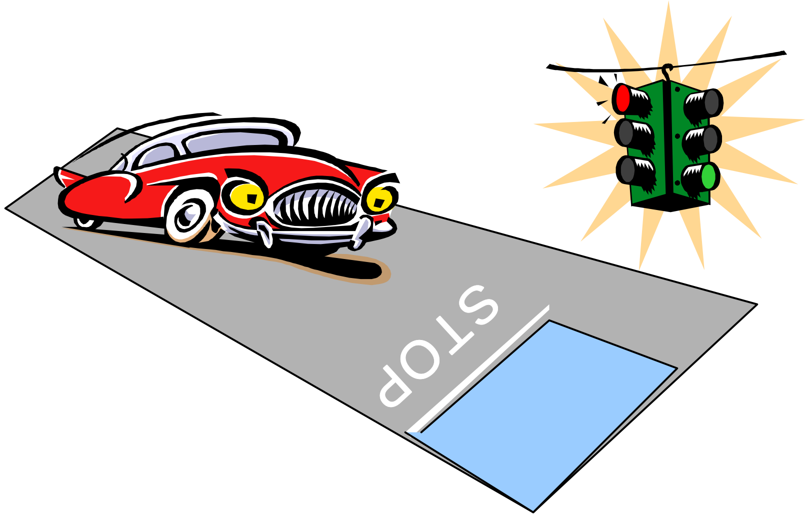
"Yes; he should have stopped until his decision was made."

Judge R. B. Trator, MIT '86

Every-day Metastability - II

<u>GIVEN</u>:

- Normal traffic light:
- GREEN, YELLOW, RED sequence
- 55 MPH Speed Limit
- Sufficiently long YELLOW, GREEN periods
- Analog POSITION input
- digital RED, YELLOW, GREEN inputs
- digital GO output



Can one reliably obey....

- LAW #1: DON'T CROSS LINE while light is RED.

  GO = GREEN

- LAW #2: DON'T BE IN INTERSETION while light is RED.

<u>PLAUSIBLE STRATEGIES</u>:

A. Move at 55. At calculated distance D from light, sample color (using an unbounded-time synchronizer). GO ONLY WHEN stable GREEN.

B. Stop 1 foot before intersection. On GREEN, gun it.

Summary

The most difficult decisions
are those that matter the least.

As a system designer…

Avoid the problem altogether, where possible

- Use single clock, obey dynamic discipline
- Avoid state. Combinational logic has no metastable states!

Delay after sampling asynchronous inputs: a fundamental cost of synchronization