Synchronizers and Arbiters

David Kinniment
University of Newcastle

Outline

- What's the problem? Why does it matter?
- Synchronizer and arbiter circuits
- Noise, and its effects
- Latency, and how to overcome it

9:00 – 10:30

- Metastability measurement 1
  - Simple measurements

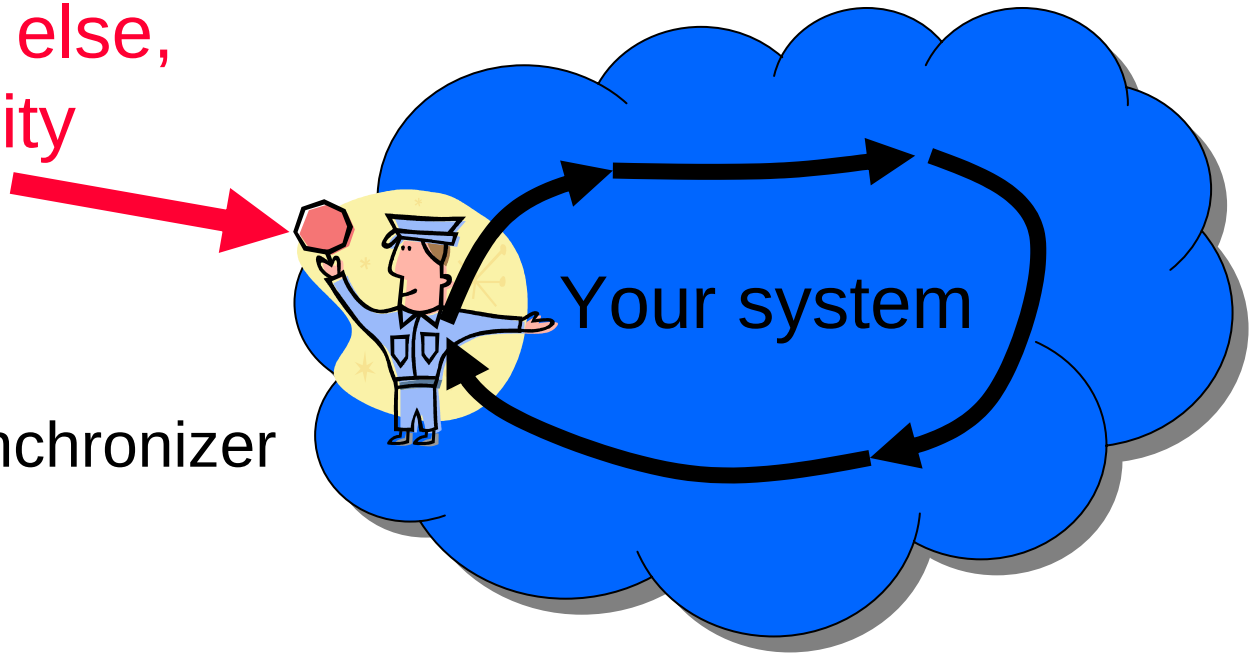
- Arbitration

11:00 – 12:00

- Metastability measurement 2
  - Second order effects (Which may matter)

What's the problem: The digital world and the real world

Everything else, or Reality

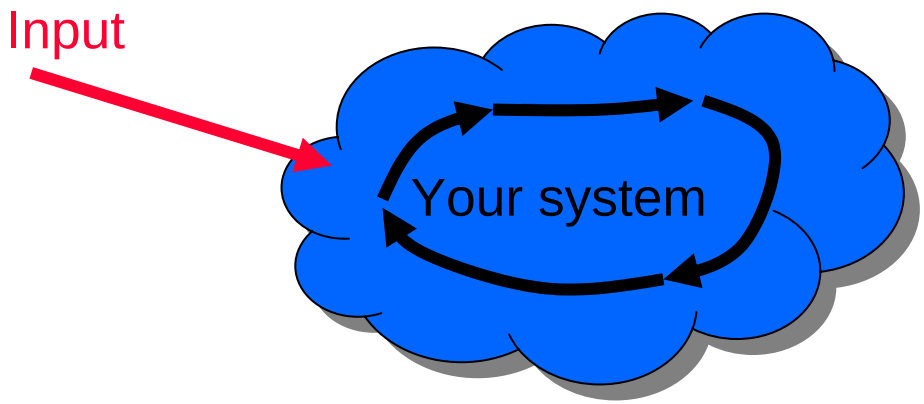


The synchronizer

Synchronizers and arbiters

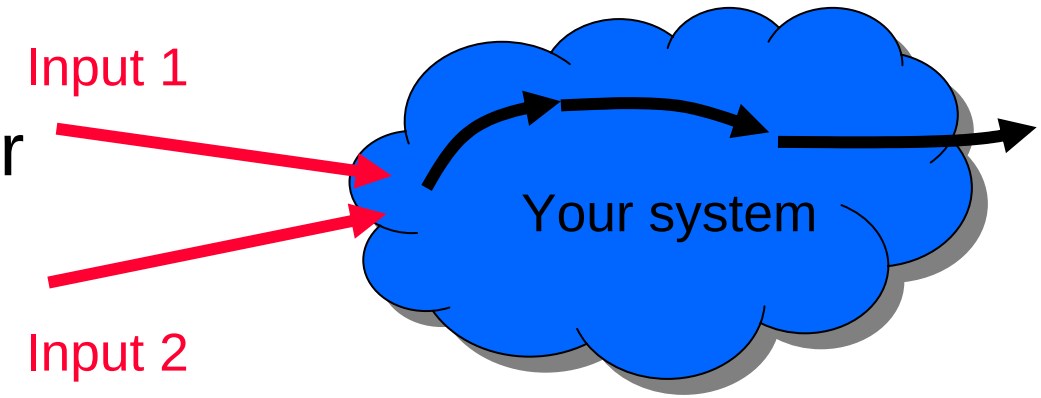
- Synchronizer

  Decides which clock cycle to use for input



- Asynchronous arbiter

  Decides which input to take first



Time Comparison Hardware

- Digital comparison hardware (which compares integers) is easy
  - Fast
  - Bounded time
- Analog comparison hardware (which compares reals like time) is hard
  - Normally fast, but takes longer as the difference becomes smaller
  - Can take forever
- Synchronization and arbitration involve comparison of time

History & Philosophy

- Abu Hamid Ibn Muhammad Ibn Muhammad al-Tusi al-Shafi'i al-Ghazali ~1100
  - "Suppose two similar dates in front of a man who has a strong desire for them, but who is unable to take them both. Surely he will take one of them through a quality in him, the nature of which is to differentiate between two similar things"
  - He felt that this demonstrated free will
- Jehan Buridan, Rector of Paris University ~1340
  - Buridan's Ass (A dog with two bowls?)
  - "Should two courses be judged equal, then the will cannot break the deadlock, all it can do is to suspend judgment until the circumstances change, and the right course of action is clear"
  - He's not so sure

Digital Computers

- Voltages have a finite number of values in a computer, 1 and 0
- Time has a discrete number of instants in a synchronous system

BUT

- Computers have to talk to other computers and to people who are not synchronous
- Known to early computer designers:
  - Lubkin 1952, Catt 1966
  - Chaney and Littlefield 1966/72

State of the art in 1980

'Well, Dammit, That's Not What This Printout Says!'

Your options

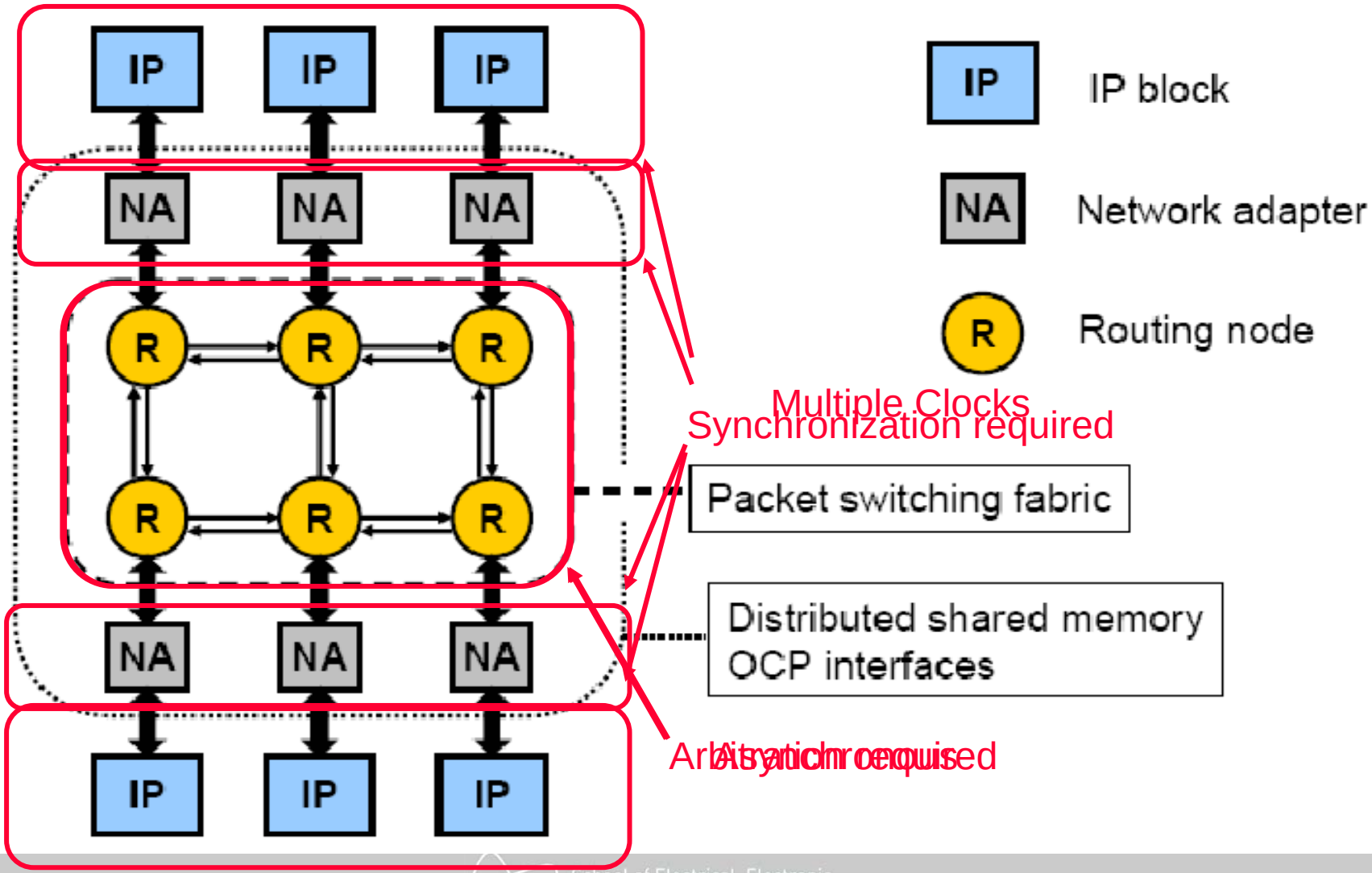
- Synchronizing a clocked system
  - You have a limited time to synchronize
  - Synchronizer circuits may fail to work in that time
  - System sometimes fails
  - You fly into a mountain
- Arbitrating requests for an asynchronous system
  - Can take forever (with decreasing probability)
  - You fly into a mountain

Why does it matter?

- Systems are Globally Asynchronous
  - 4 x increase in global asynchronous signalling by 2012
  - 8 x by 2020 [ITRS 2005]
  - Communication time is an increasing part in system performance

- And Locally Synchronous
  - Many different clocks
  - Many synchronizers
  - Need to know the reliability of the synchronizer.
  - Synchronisation adds latency to communication time

A Network on Chip (Sparsø 2005)



Outline

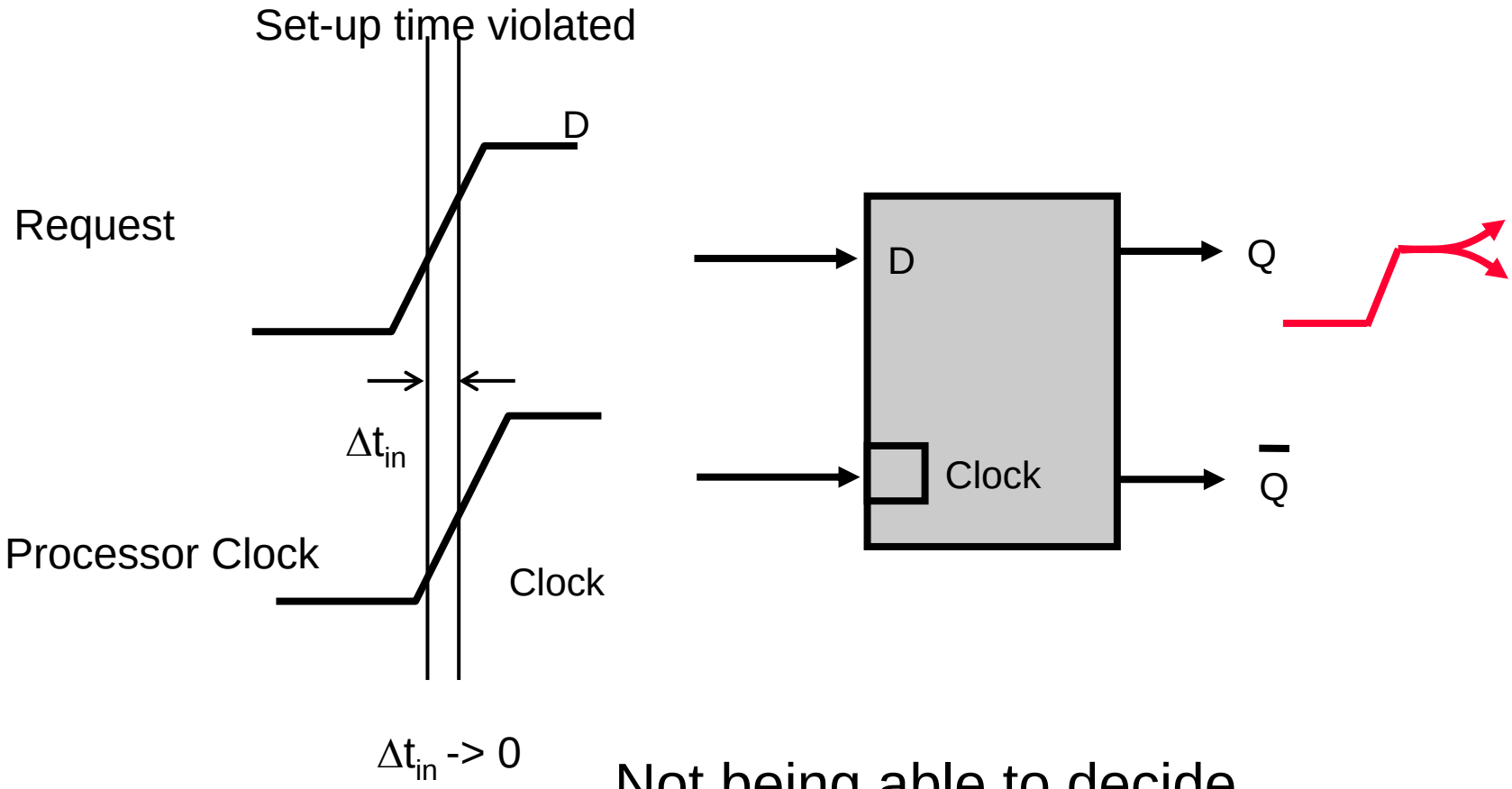
- What's the problem? Why does it matter?
- Synchronizer and arbiter circuits
- Noise, and its effects
- Latency, and how to overcome it

- Metastability measurement 1
  - Simple measurements

- Arbitration

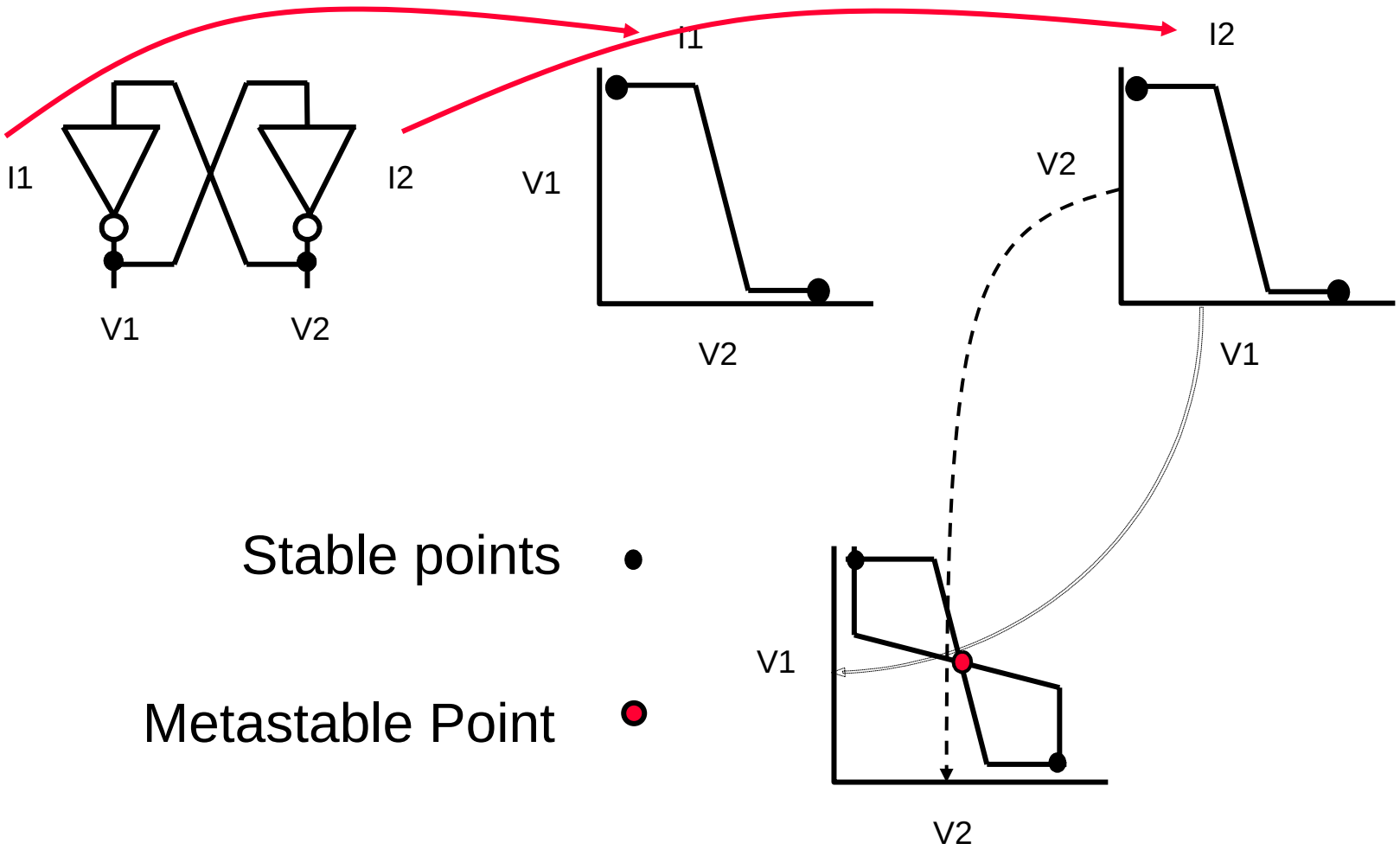
- Metastability measurement 2
  - Second order effects (Which may matter)

Metastability is....



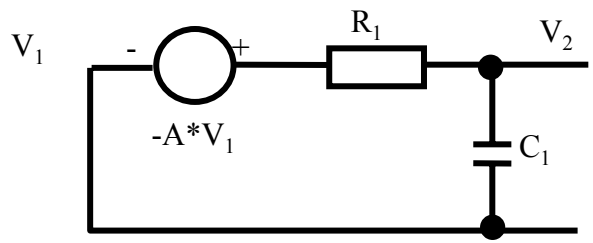
Not being able to decide…

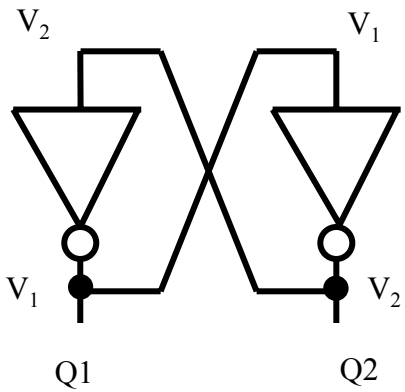
Metastability in a Latch



Linear Model

- Simple linear model leads to two exponentials
- τ_a is convergent, τ_b is divergent





$$\tau_1 = \frac{C_1 . R_1}{A}, \tau_2 = \frac{C_2 . R_2}{A} \qquad g_m = \frac{A}{R}$$

$$0 = \tau_1 . \tau_2 . \frac{d^2 V_1}{dt^2} + \frac{(\tau_1 + \tau_2)}{A} . \frac{dV_1}{dt} + (\frac{1}{A^2} - 1).V_1$$

$$V_1 = K_a . e^{\frac{-t}{\tau_a}} + K_b . e^{\frac{t}{\tau_b}}$$

How often does it fail?

- The output trajectory is an exponential that depends on the starting condition *K*, *K* depends on Δt_{in}
- Suppose the clock frequency is f_c, the data rate f_d, and $K_a = 0$
- In *M* seconds we have M.f_c clocks.
- The probability of a data change within Δt_{in} of any clock is Δt_{in}. f_d, so there will be one within M seconds if
- The time taken to resolve this event is *t* (T_w is the metastability window)

$$V = K.e^{\frac{t}{\tau}}$$

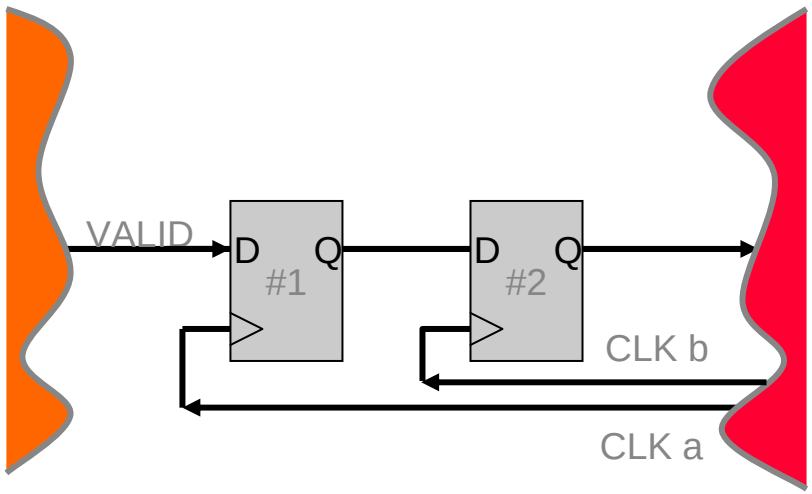
$$\Delta t_{in} = \frac{1}{M.f_c.f_d}$$

$$\frac{V}{K} = \frac{T_w}{\Delta t_{in}} = e^{\frac{t}{\tau}}$$

Synchronizer

- *t* is time between clock a and clock b
- τ, and T_w depend on circuit

$$MTBF = \frac{1}{\Delta t_{in} f_c f_d}$$

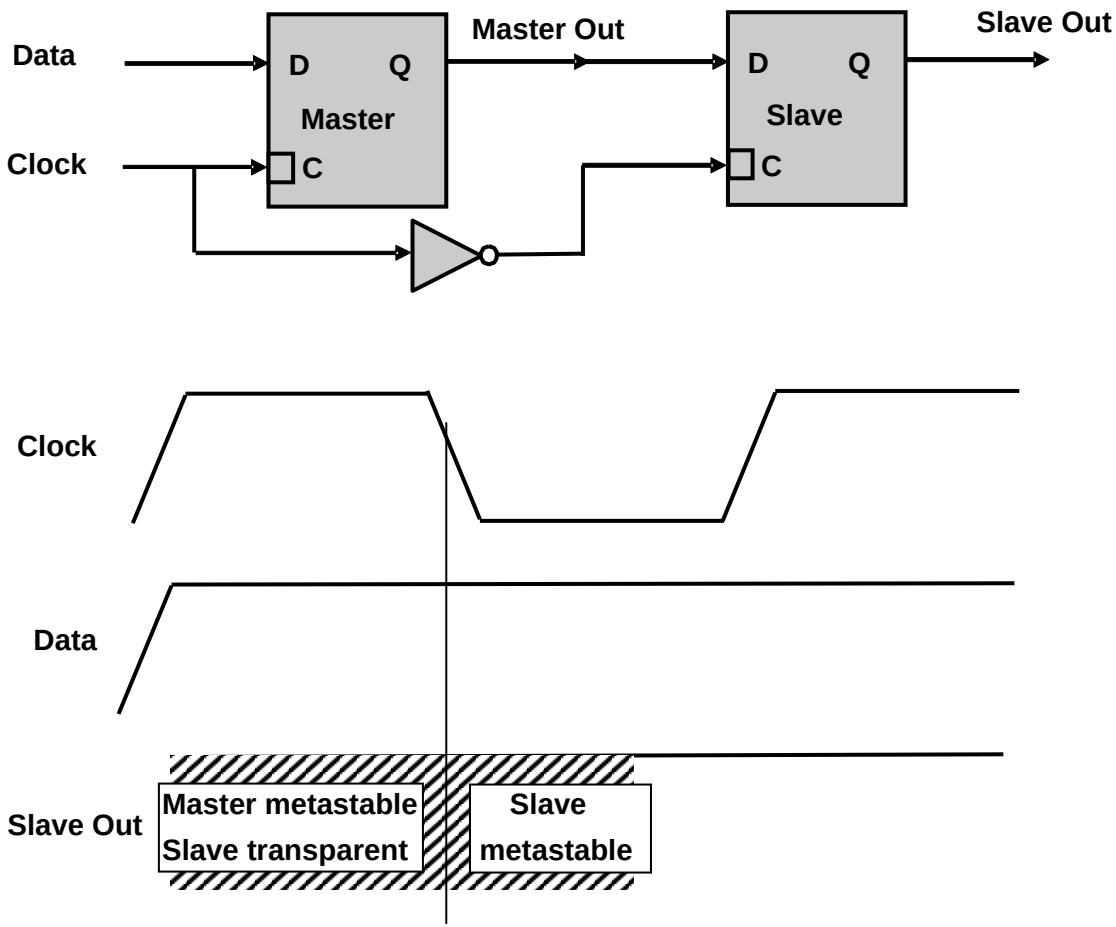


OR

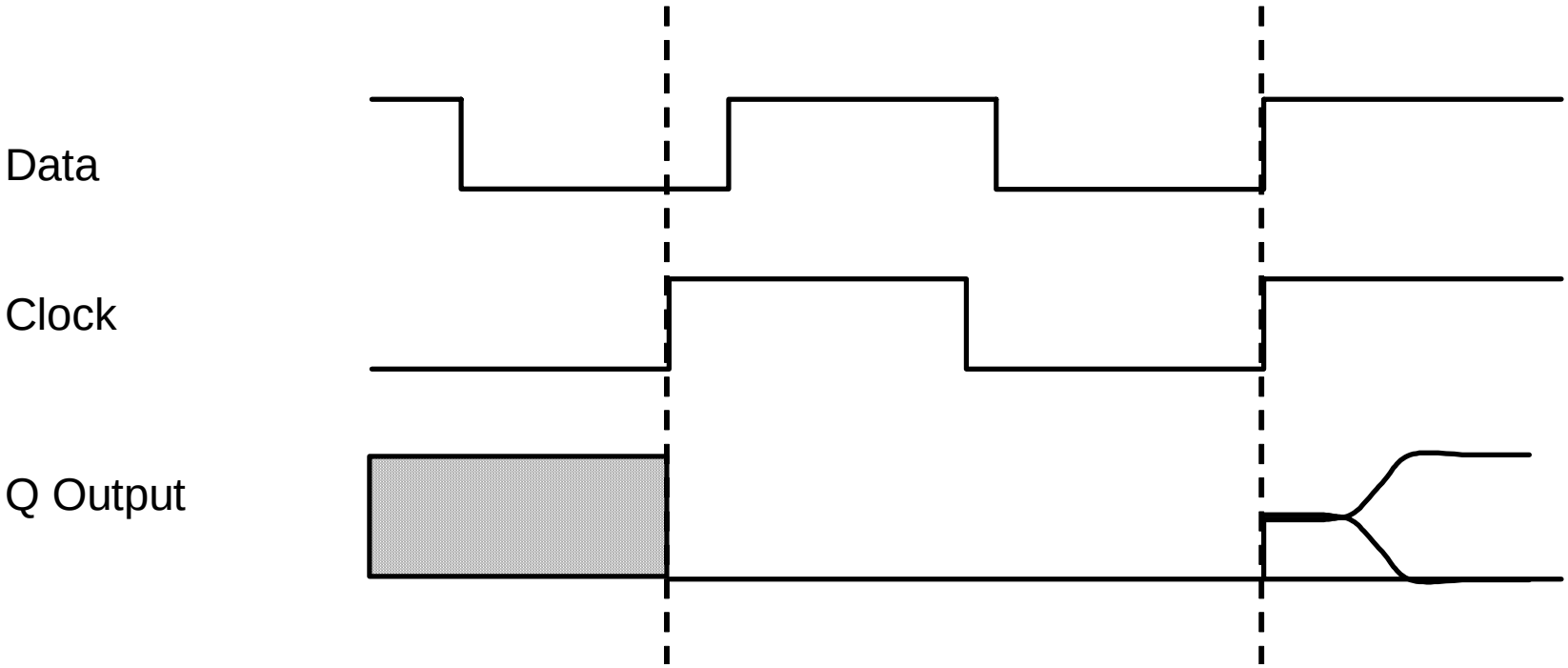
$$MTBF = \frac{e^{t/\tau}}{T_w . f_c . f_d}$$

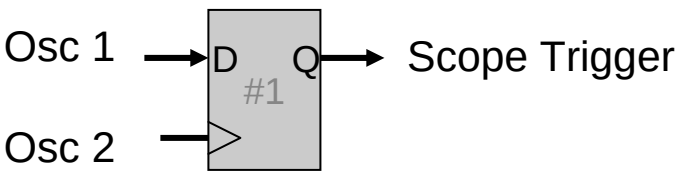
Edge Triggered FF

- Synchronizer fails if time too long
- Failures proportional to
  - Clock frequency
  - Data change frequency



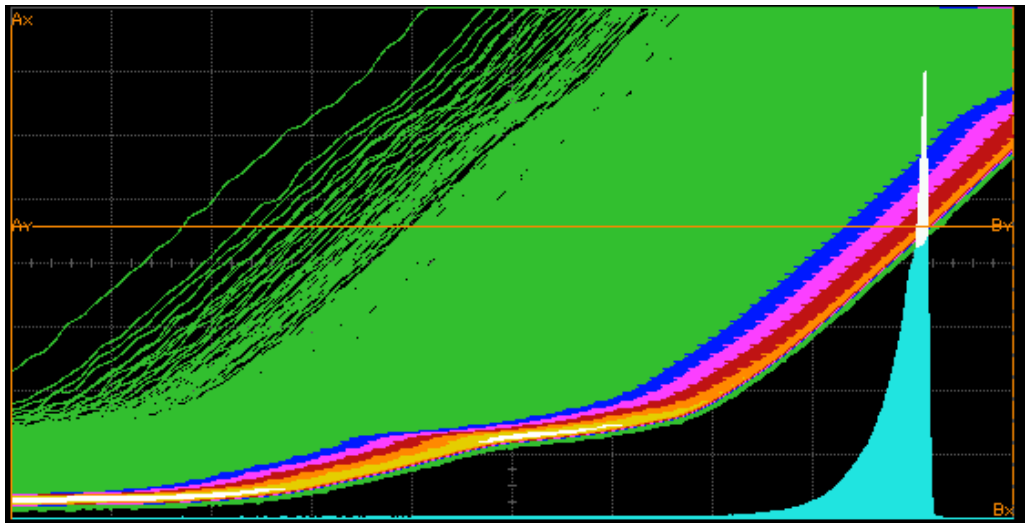
Synchronizer responses





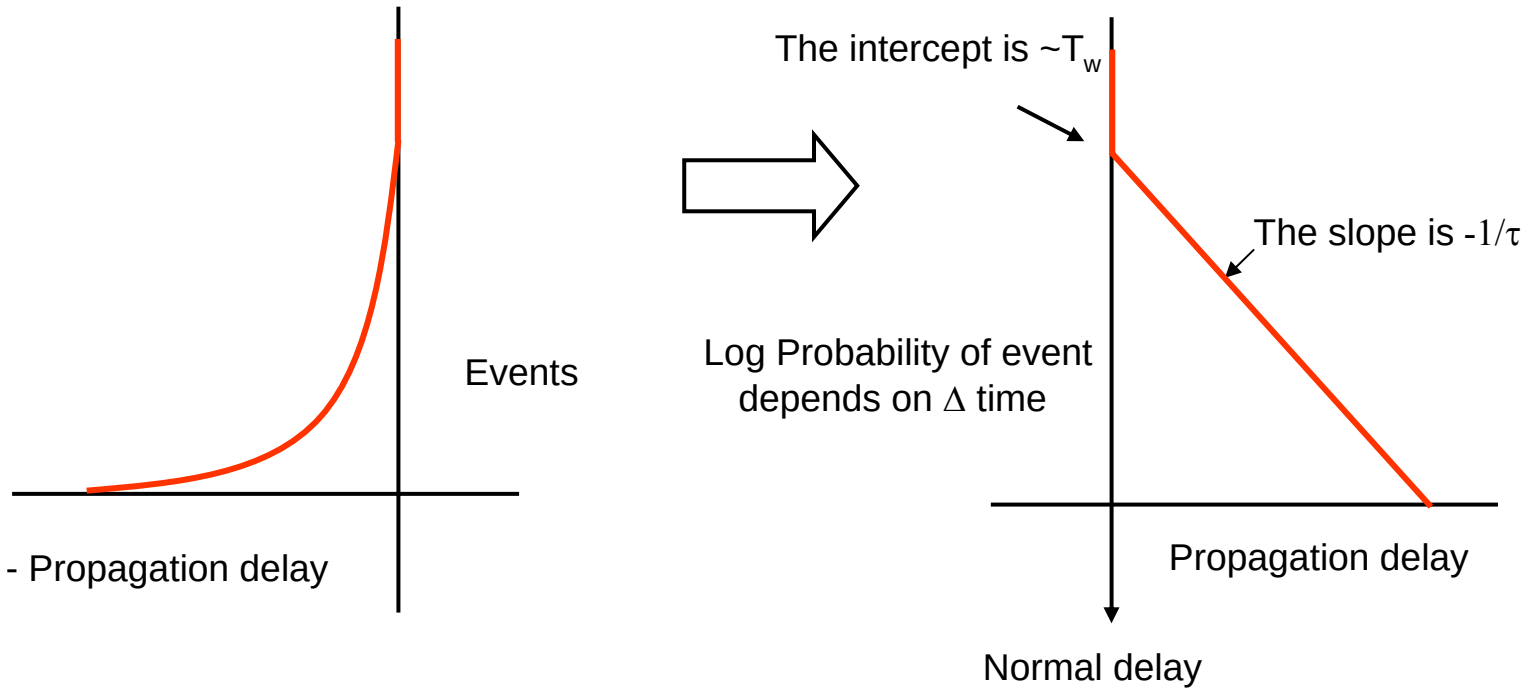
- **Data and Clock are asynchronous**
- **Trigger from Q and observe time between clock and Q**

Typical responses



- All starting points are equally possible
- Most are a long way from the “balance point”
- A few are very close and take a long time to resolve

Event Histogram

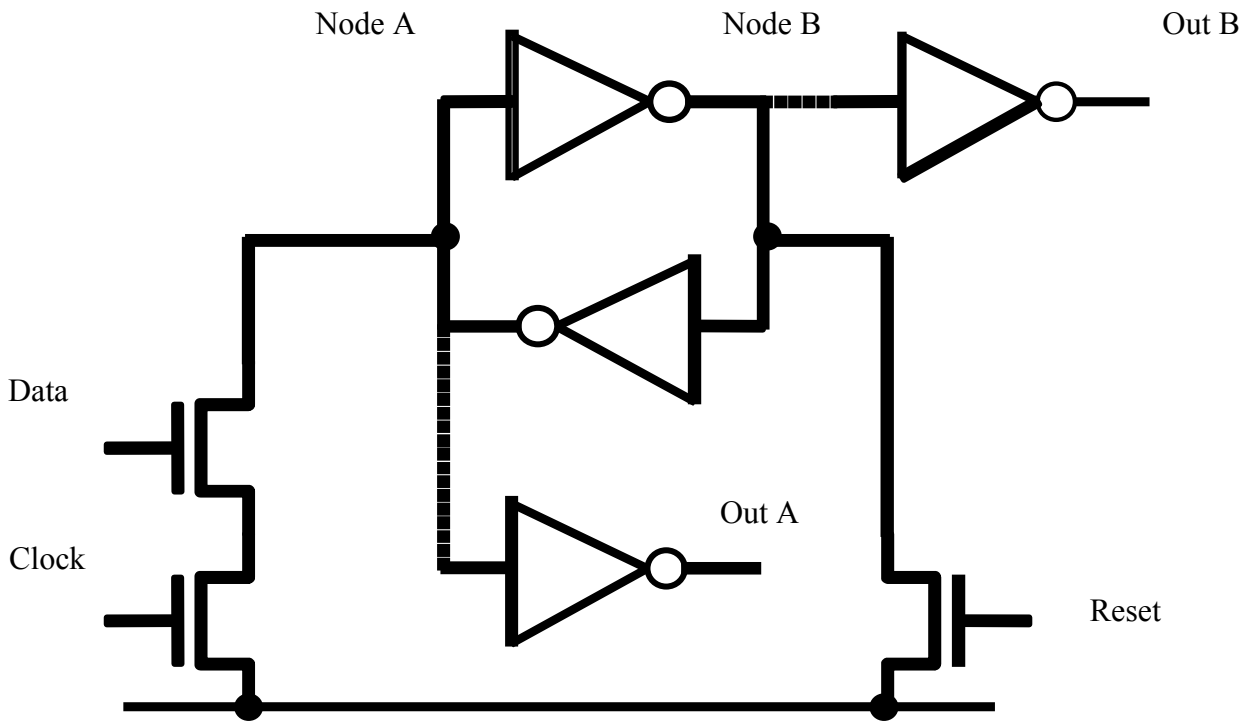


State of the art

- You require about 35 τ s in order to get the MTBF out to about 1 century. (That's for 1 synchronizer)
- Each typical static gate delay is equivalent to about 5 τ s in a properly designed synchronizing flop. (You can increase V_{dd} on the flop to get it faster)
- You should assume a 'malicious' input to the synchronizer. This adds about 5 -10 τs to the delay (depending on how cautious you are).

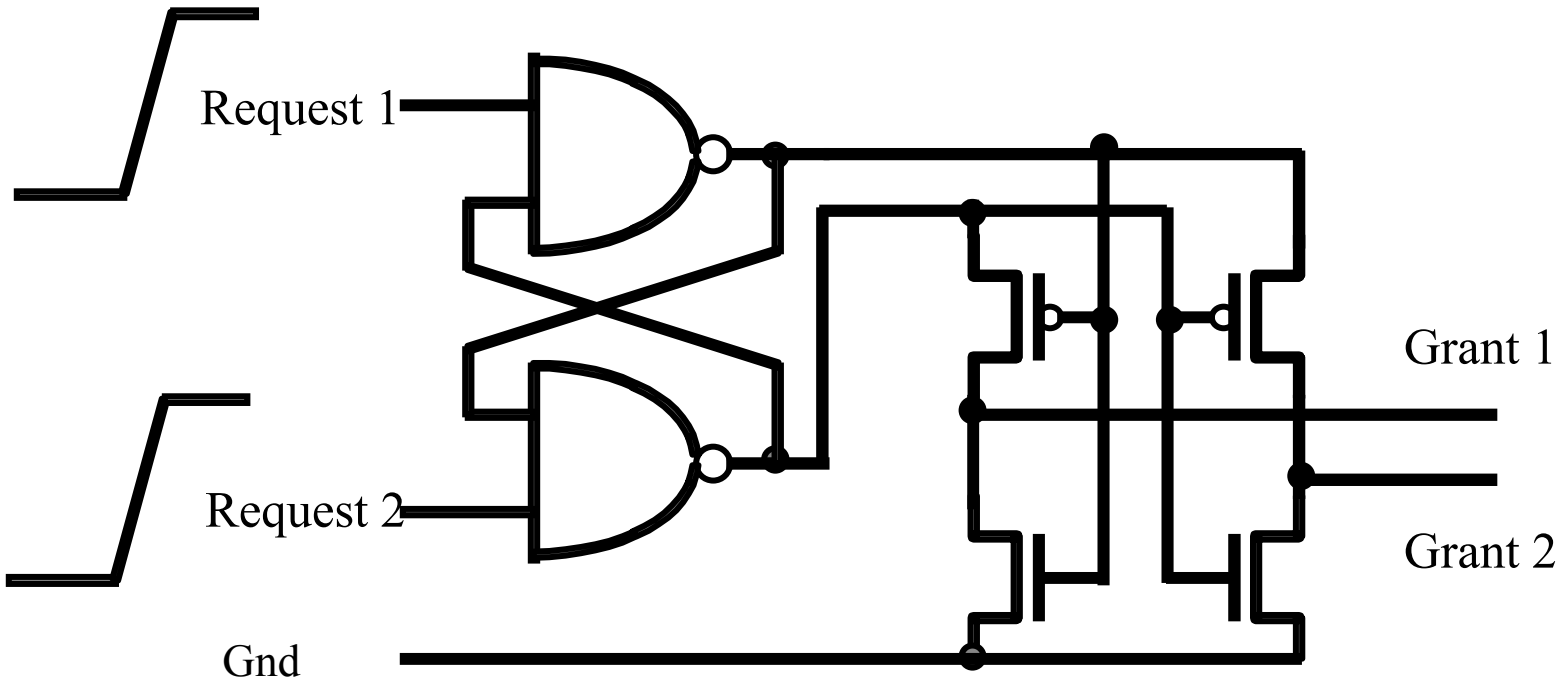
Jamb latch synchronizer

- Fast and simple



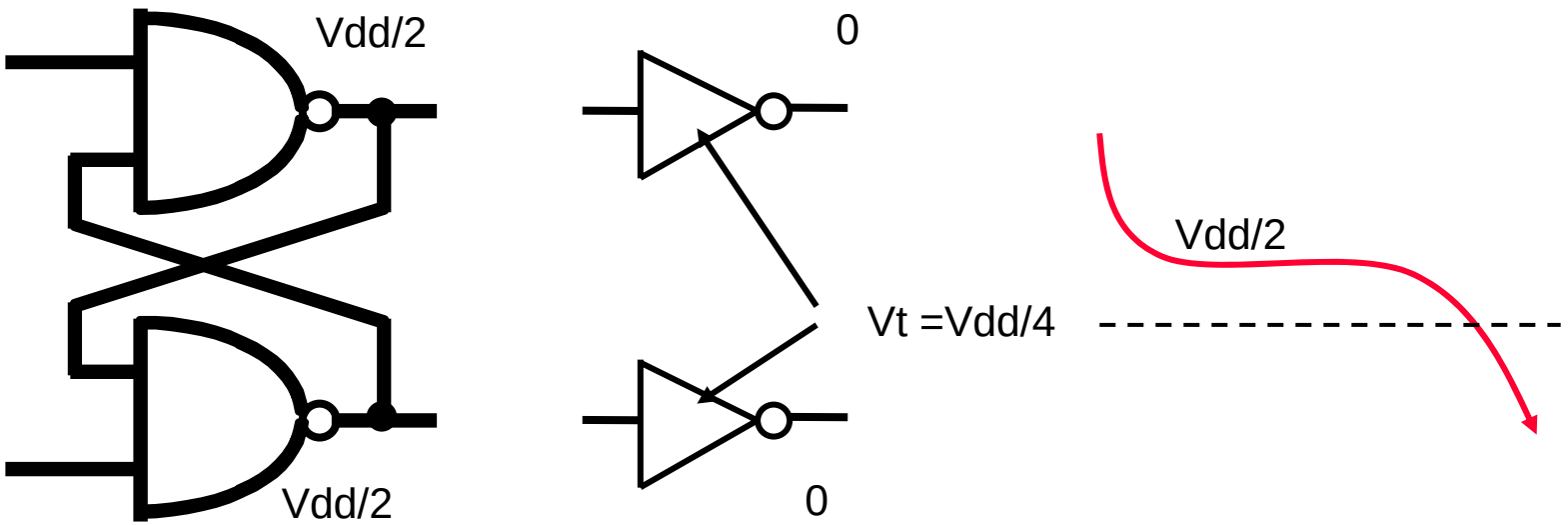
The arbiter (MUTEX)

- Asynchronous arbitration
- No time bound



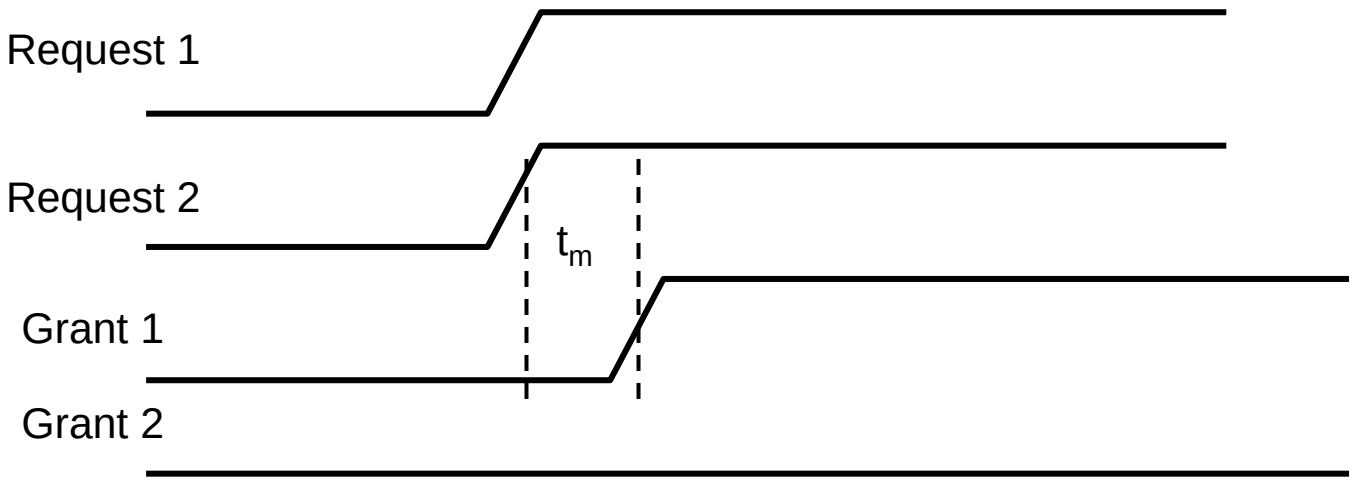
Metastability filters

- Half levels due to metastability need to be removed
  - Low (or high) threshold inverters
  - Measure divergence
- Filters define the time to reach a stable state



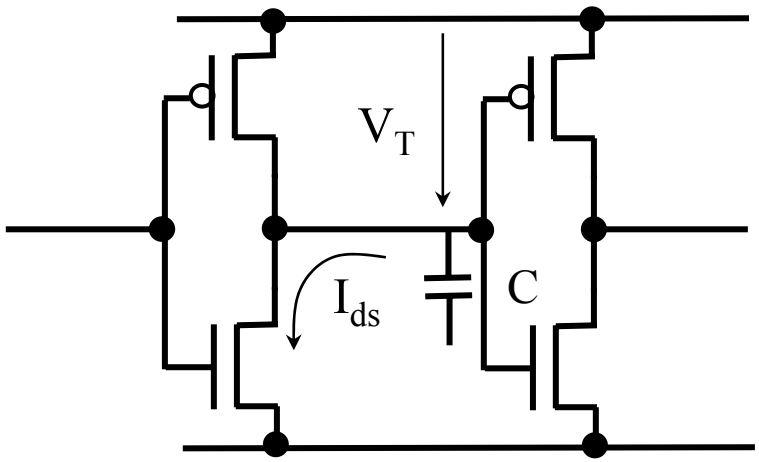
Arbitration time

- Unlike a synchronizer, an arbiter may take for ever.
- It usually doesn't, long responses are rare.
- On average the time is only τ longer than the normal response.
- Outputs are always monotonic



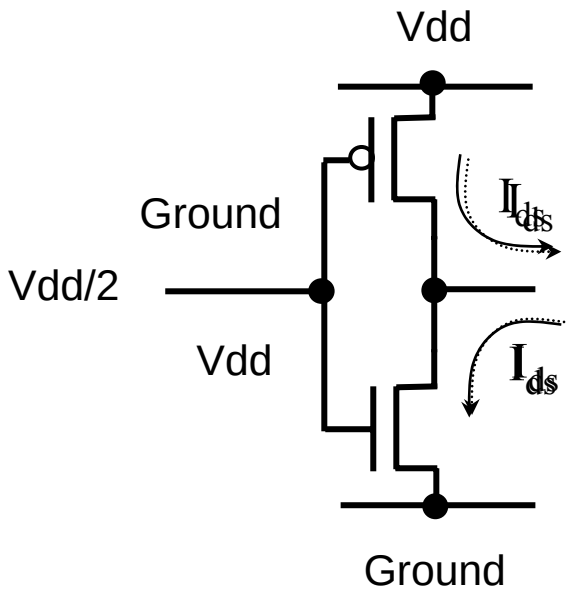
Future synchronizers

- Synchronizers don't work well in nanometre technologies
- Worse that gates! Why?
- Gate delays depend on large signal issues:
  - $C.V_T/I_{ds}$ determines how long does it take to discharge C to V_T before the next gate changes state
  - I_{ds} large when transistor is hard on



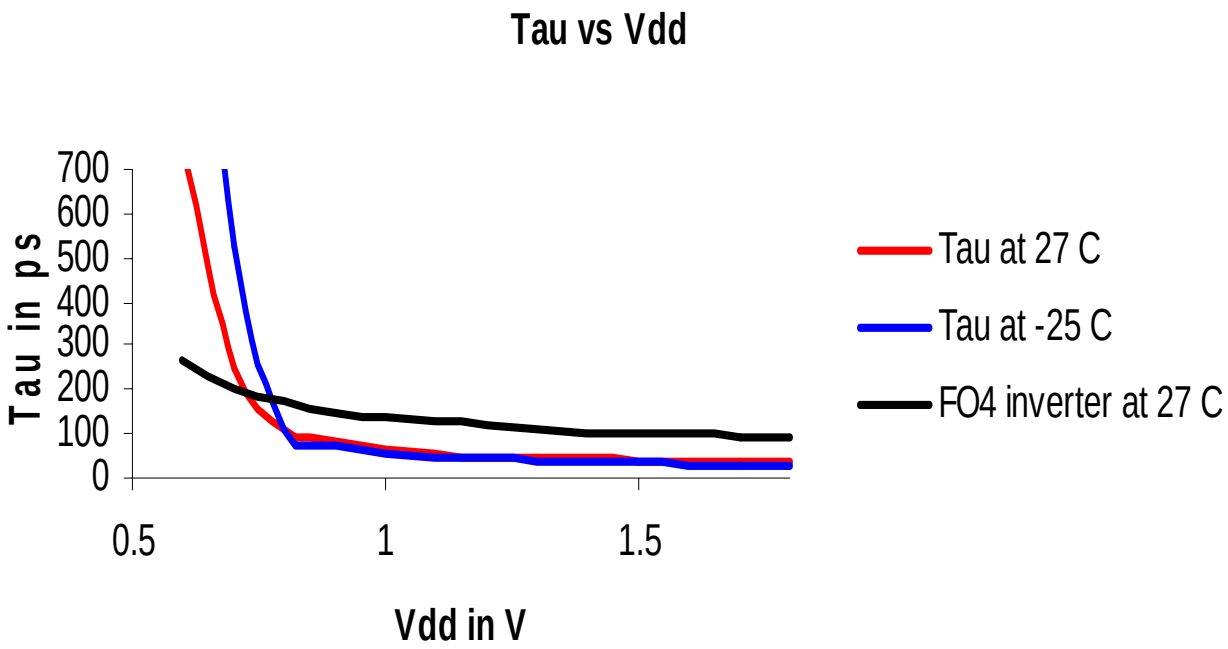
No gain at Vdd/2

- As Vdd decreases with process shrink
  - Gate threshold does not decrease to minimise leakage
- A gate input is either HIGH
  - Output pulled down
- Or Low
  - Output pulled up
- A metastable gate is neither
  - Both transistors can be off
  - g_m very low
- Synchronization time constant $\tau = C/g_m$



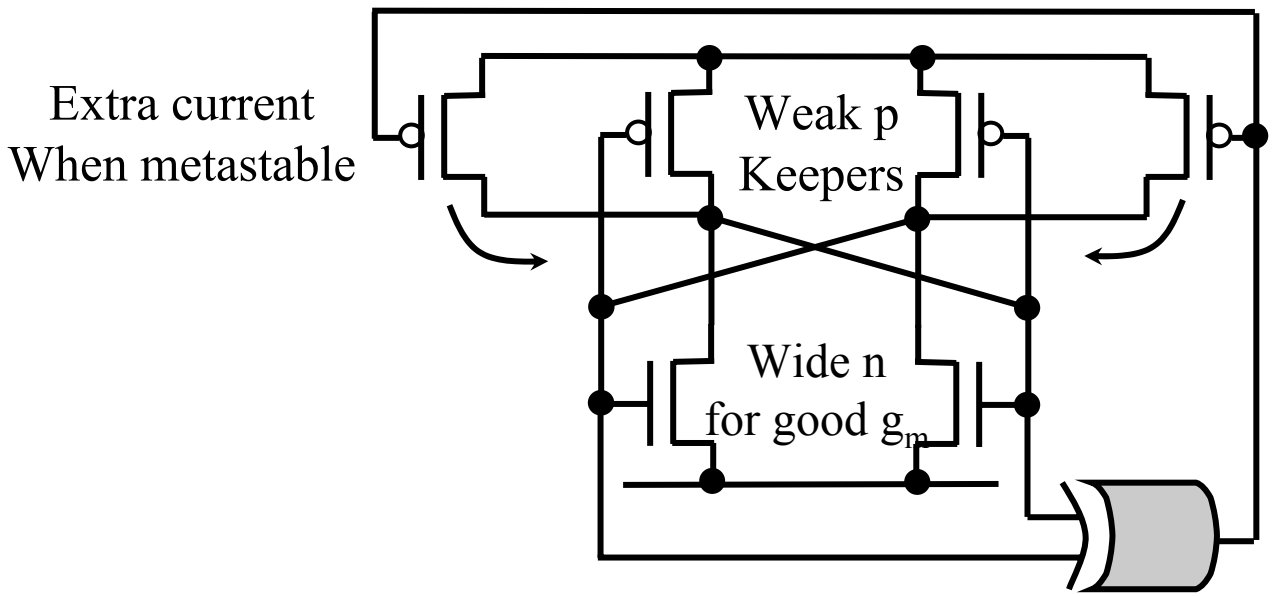
Low Vdd, low temperature

- Both transistors off, $g_m \rightarrow 0$, $\tau \rightarrow \infty$ at Vdd < 0.6V
- Low temperature gives higher threshold so even worse
- Does not track logic



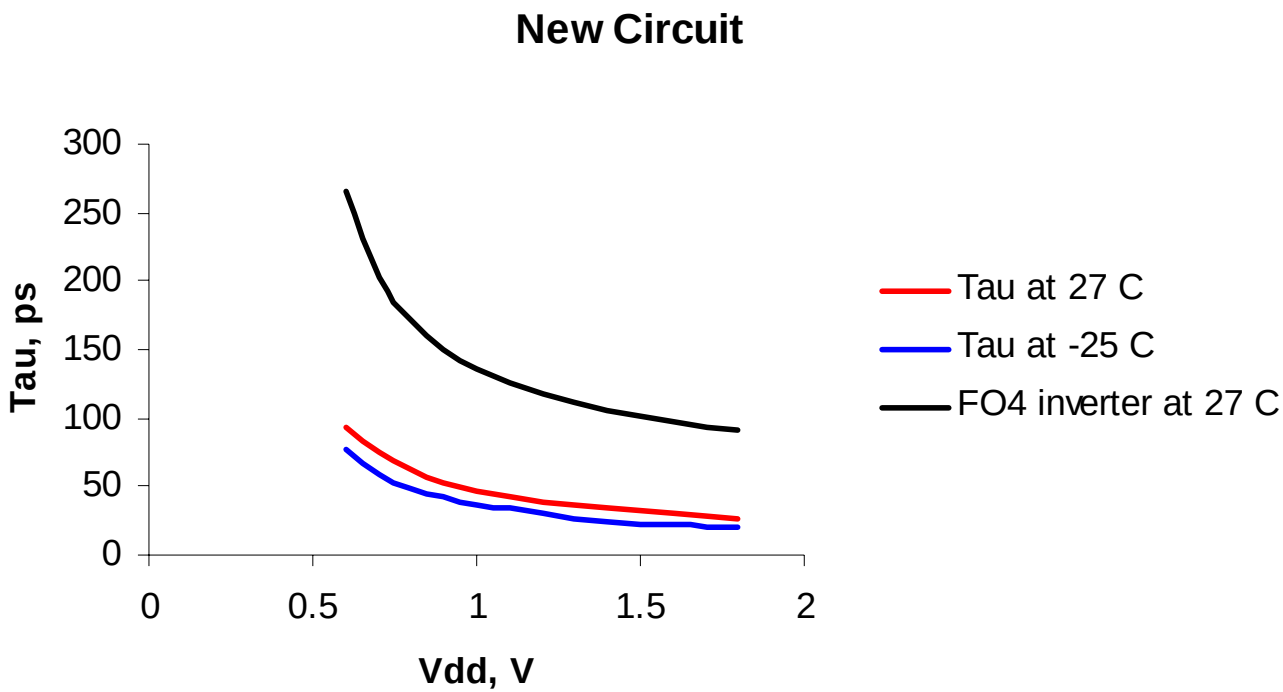
Vdd tolerant circuit

- Turn on p-types when latch is metastable
  - Extra current gives high g_m in n-types
  - Normally low power
- g_m depends mainly on n-types
  - fast



Effect of extra current

- Tau at 0.6V down from >700ps to < 100ps
- Tracks logic, so does not limit performance



Outline

- What's the problem? Why does it matter?
- Synchronizer and arbiter circuits
- Noise, and its effects
- Latency, and how to overcome it

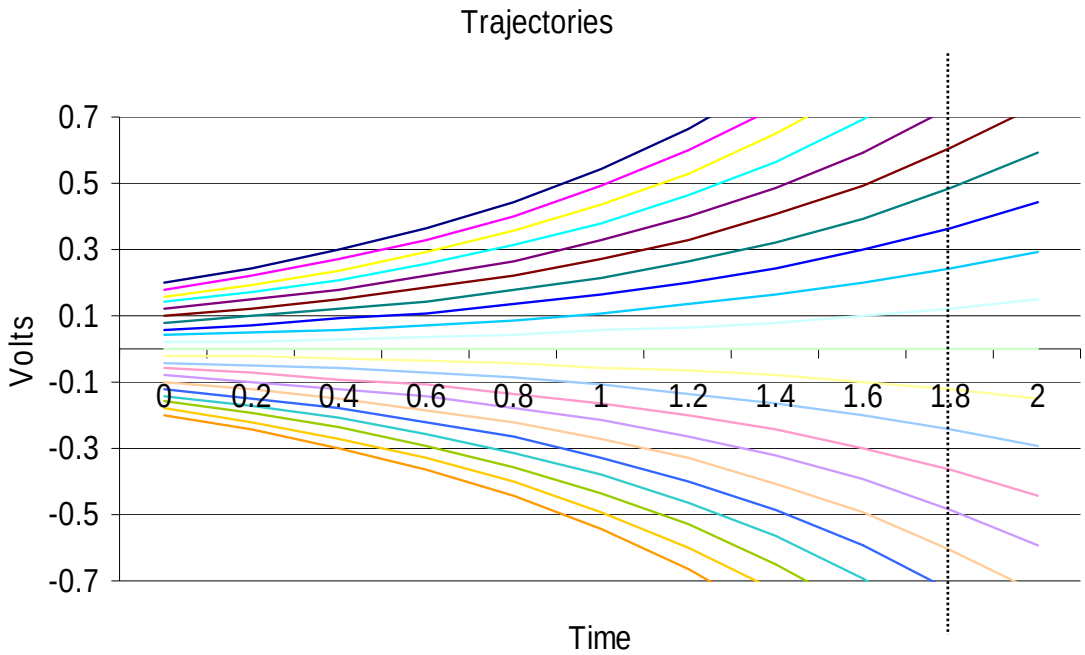
- Metastability measurement 1
  - Simple measurements

- Arbitration

- Metastability measurement 2
  - Second order effects (Which may matter)

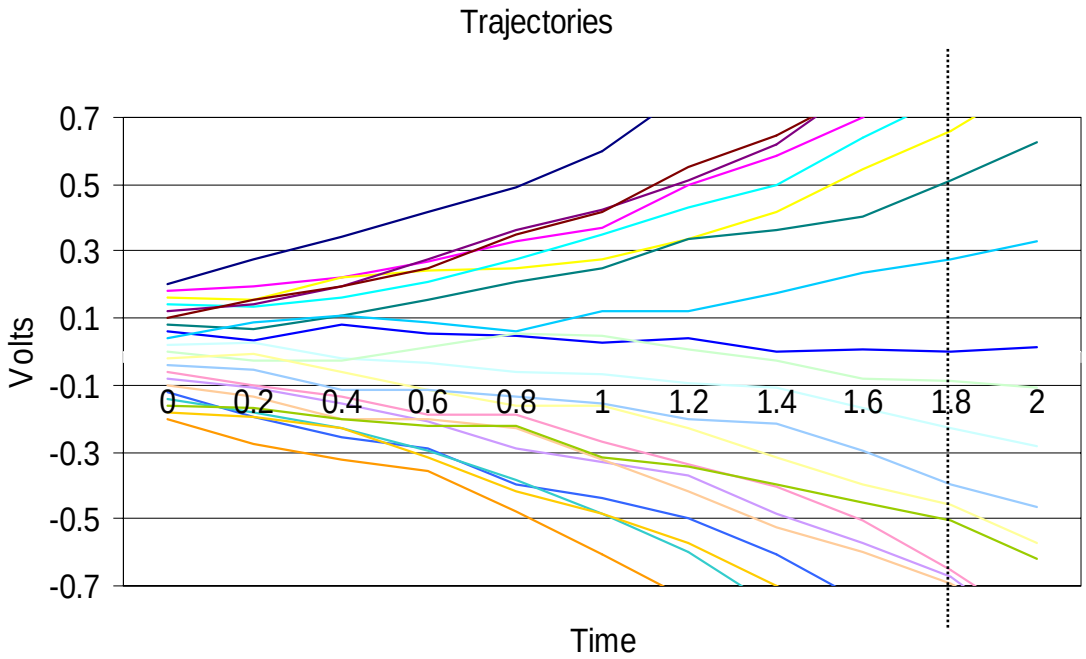
Does noise affect τ?

- Probability of escape from metastability does not change with gaussian noise (Couranz and Wann 1975)

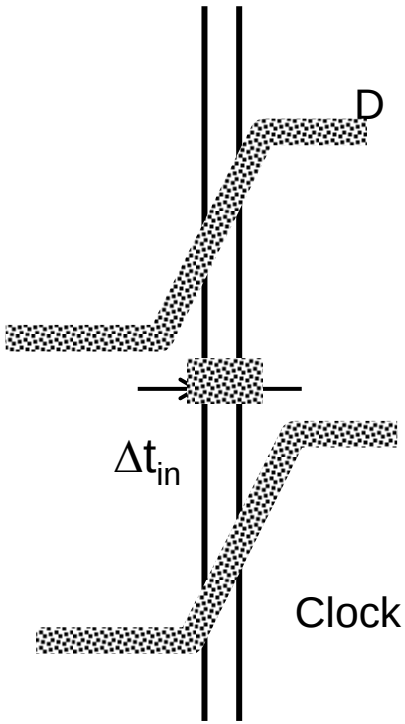


Does noise affect τ?

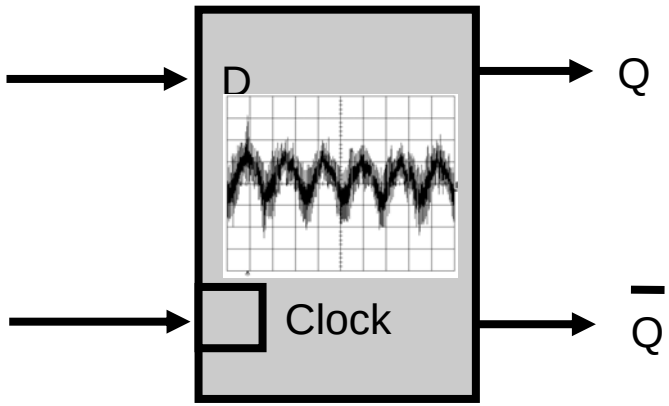
- Probability of escape from metastability does not change with gaussian noise (Couranz and Wann 1975)



Noise can change the input time



Δt_{in} -> 0



Or maybe not.....

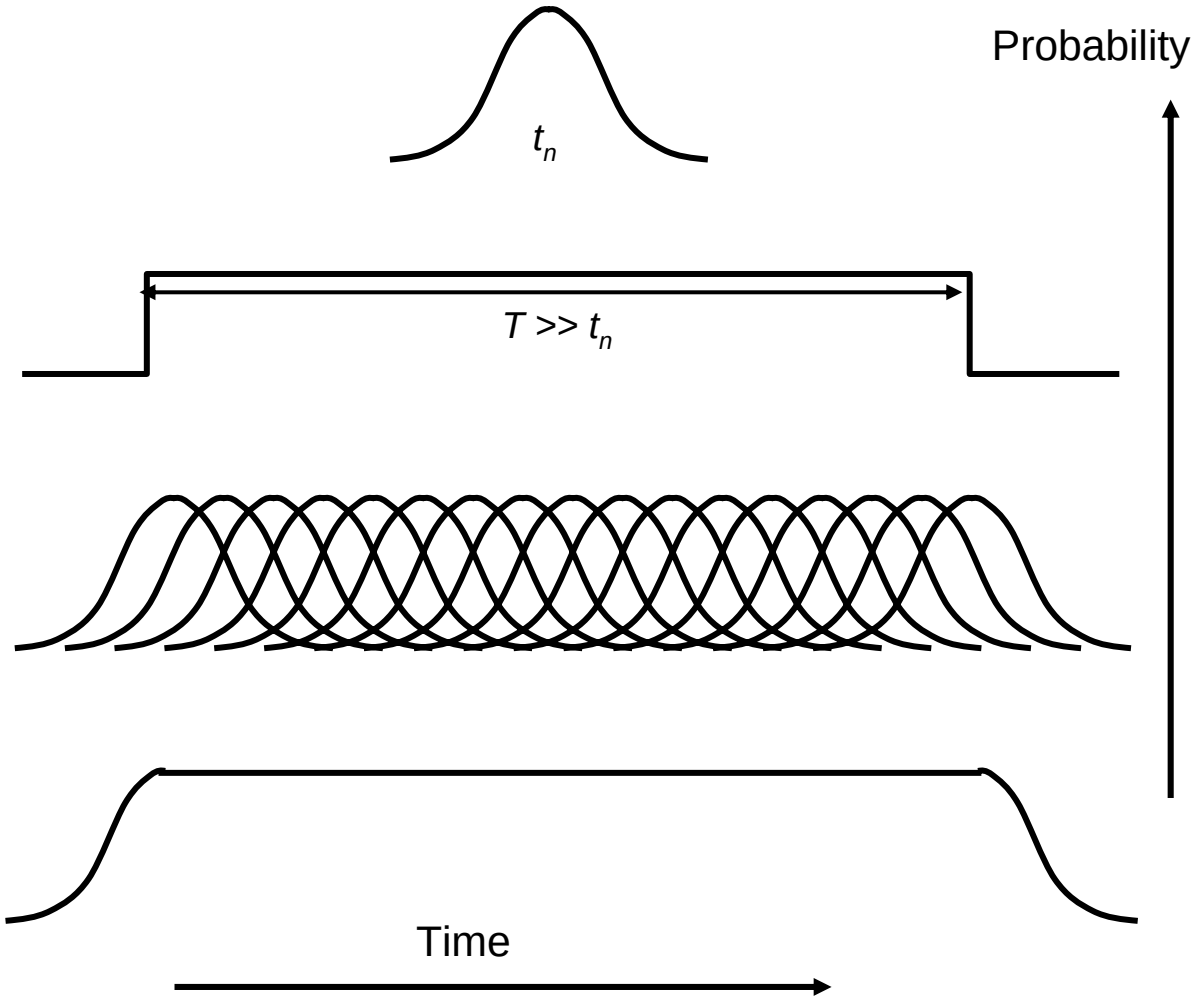
The normal case

Probability of initial difference due to noise component $P_1(v)$

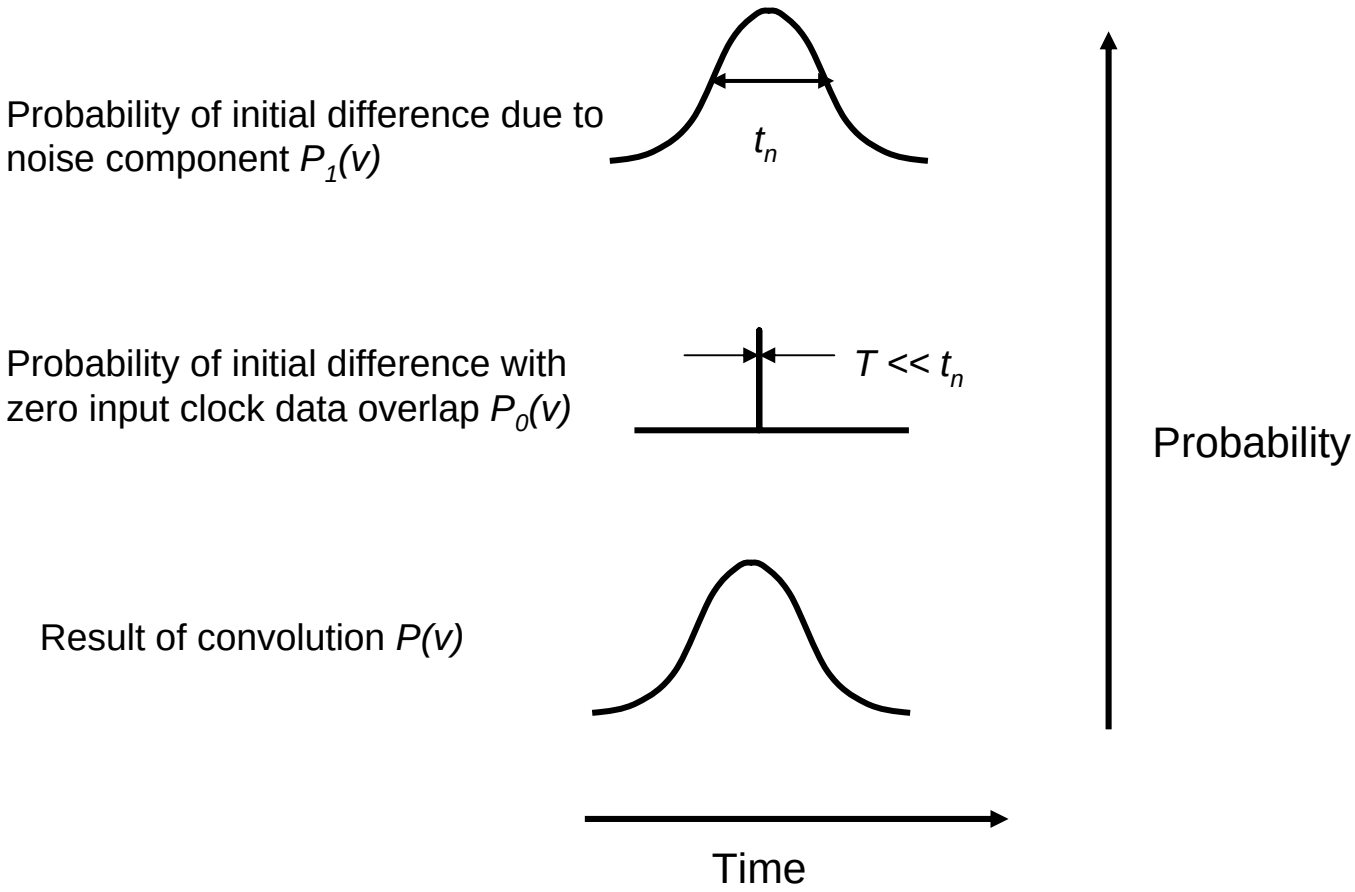
Probability of initial difference due to input clock data overlap $P_0(v)$

Convolution

Result of convolution $P(v)$



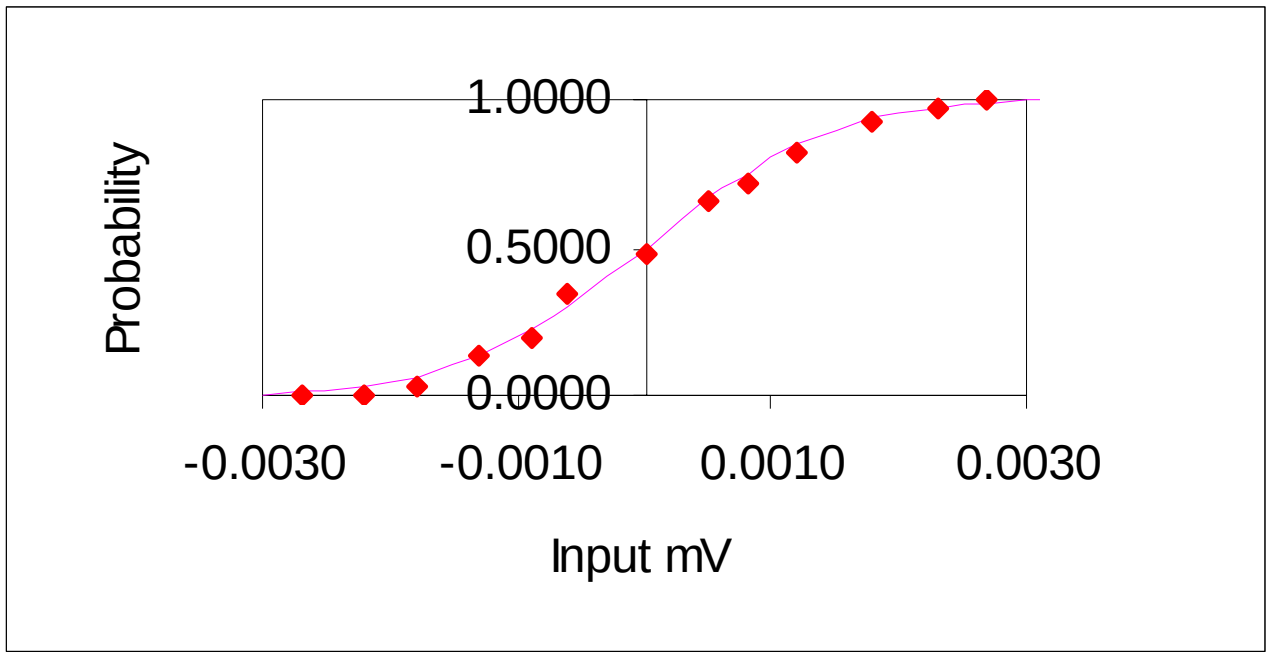
The malicious input



Noise measurement

Probability of an output 1 as a function of input voltage difference

A measurement of approximately 1.7mV RMS at the input corresponds to about 0.6mV total between latch nodes



$$\sqrt{\frac{4kT}{C}} \approx 0.7mV$$

This is equivalent to about 0.1ps
Typically this leads to a synchronization time of about 11τ longer than the simple case for a malicious input.

Outline

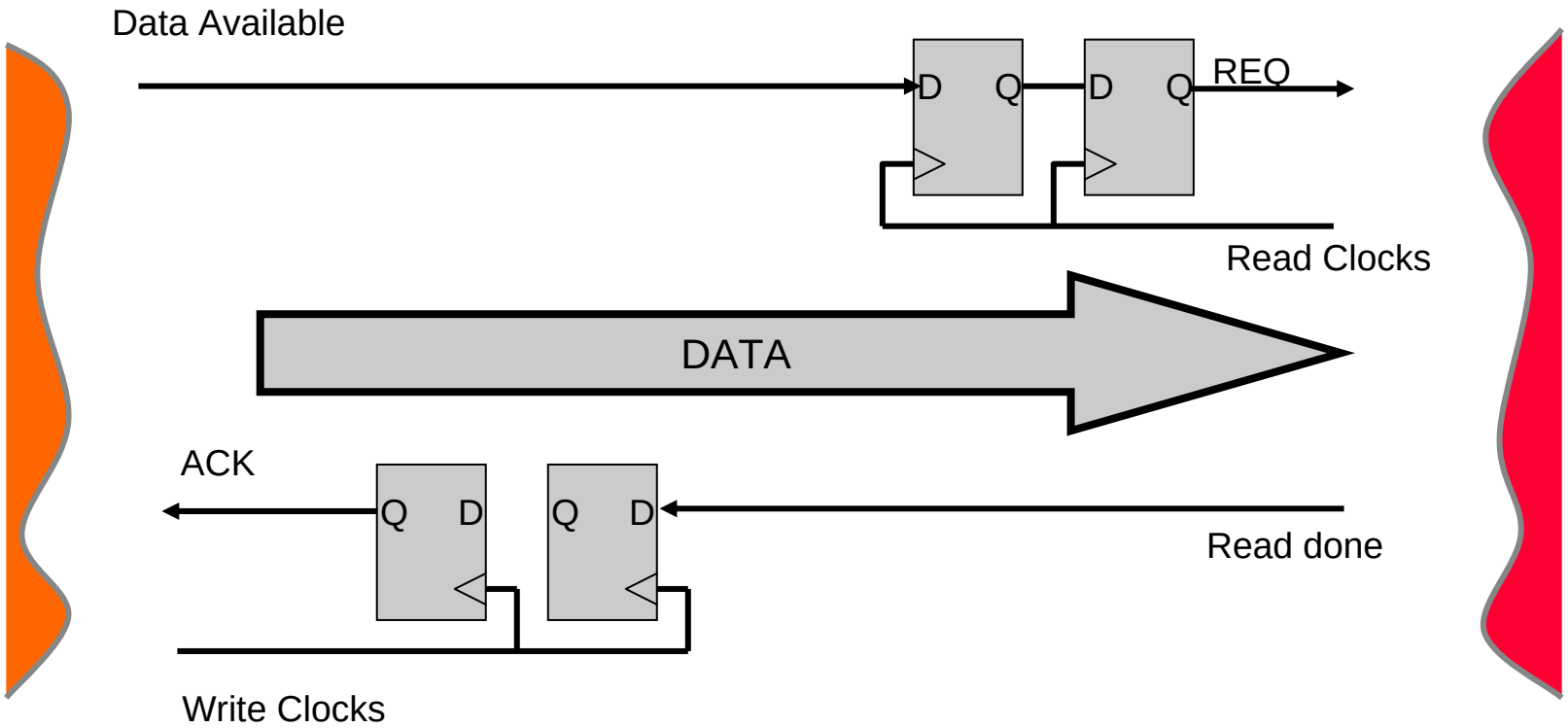
- What's the problem? Why does it matter?
- Synchronizer and arbiter circuits
- Noise, and its effects
- Latency, and how to overcome it

- Metastability measurement 1
  - Simple measurements

- Arbitration

- Metastability measurement 2
  - Second order effects (Which may matter)

Request and Acknowledge



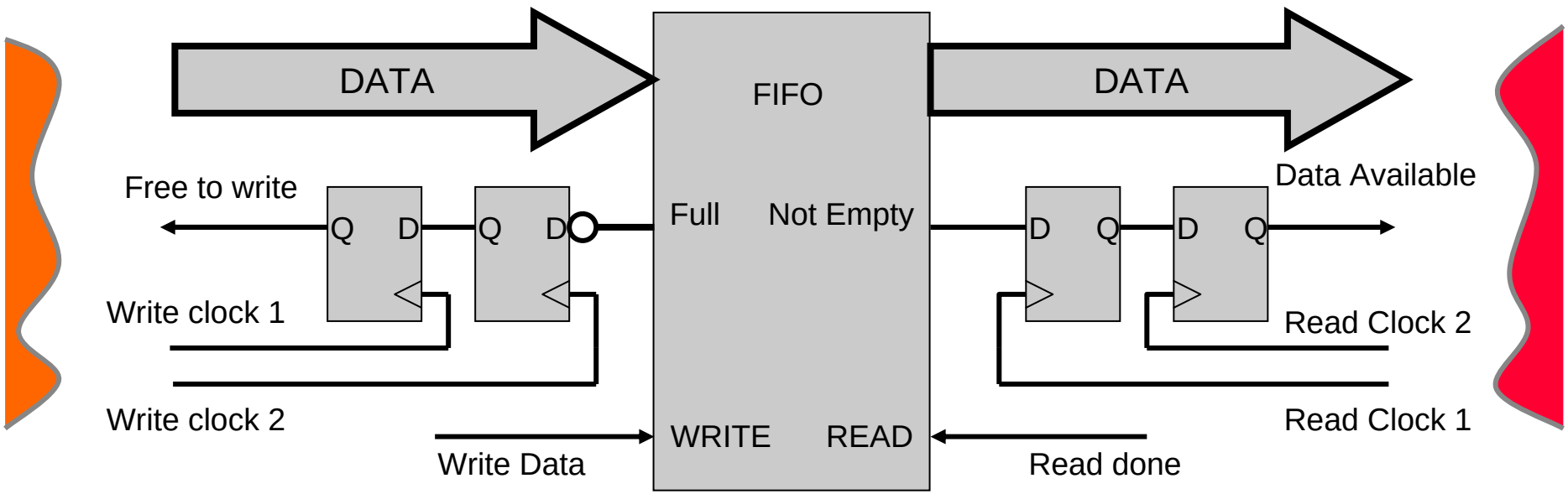
Latency

- It takes one - two receive clocks to synchronise the request
- Then one – two write clocks to acknowledge it

- Significant latency (1-3 clocks)
- Poor data rate (2 – 6 Clocks)

FIFO

- Can improve data rate by using a FIFO
- But not latency (which gets worse)
- FIFO is asynchronous (usually RAM + read and write pointers)

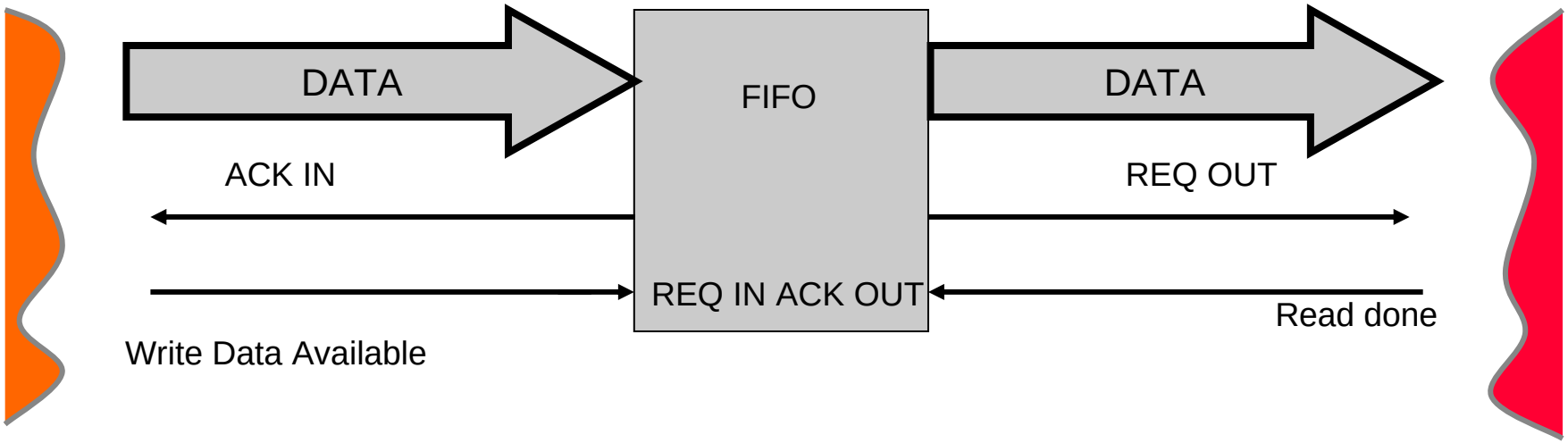


Timing regions can have predictable relationships

- Locked
  - Two clocks are not the same but phase linked, The relationship is known as *mesochronous*.
  - Two clocks from same source
  - Linked by PLL
  - One produced by dividing the other
  - Some asynchronous systems
  - Some GALS
- Not locked together
  - Phase difference can drift in an unbounded manner. This relationship is called *plesiochronous*
  - Two clocks same frequency, but different oscillators.
  - As above, same frequency ratio

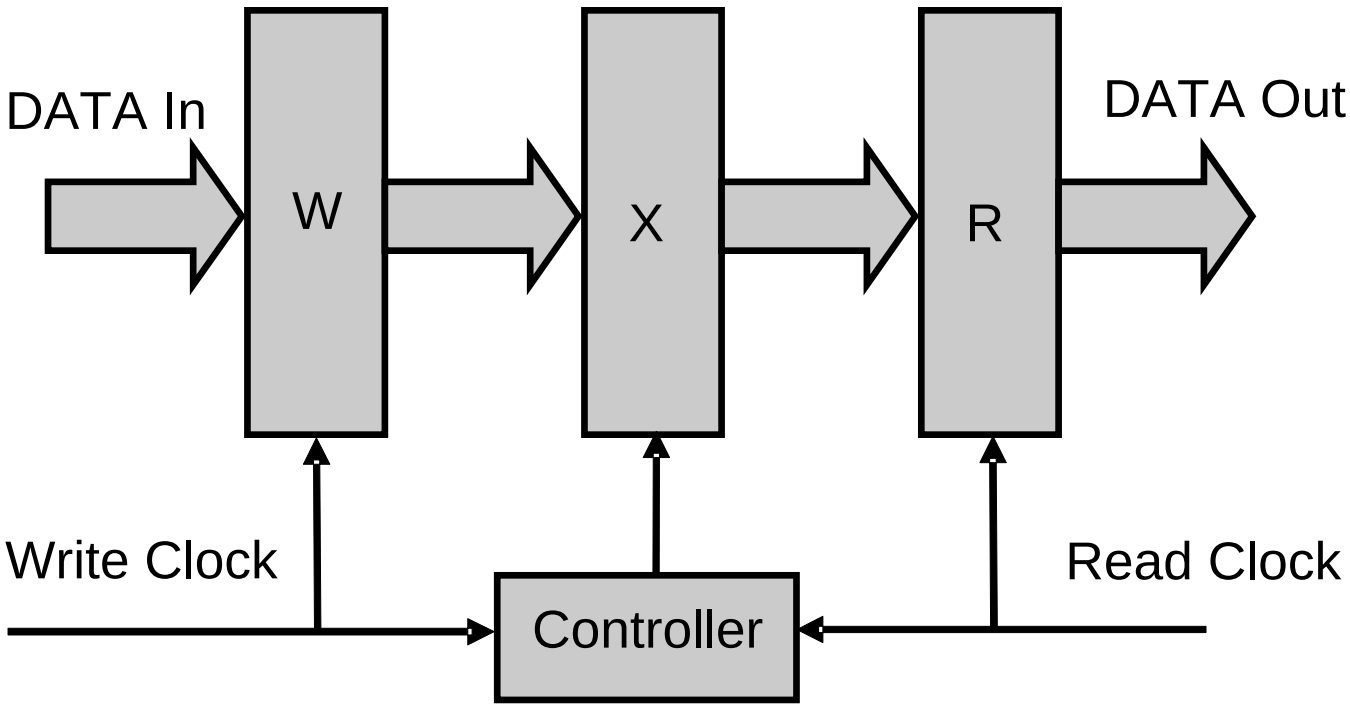
Don’t synchronise when you don’t need to

- If the two clocks are locked together, you don’t need a synchroniser, just an asynchronous FIFO big enough to accommodate any jitter/skew
- FIFO must never overflow, so there is latency



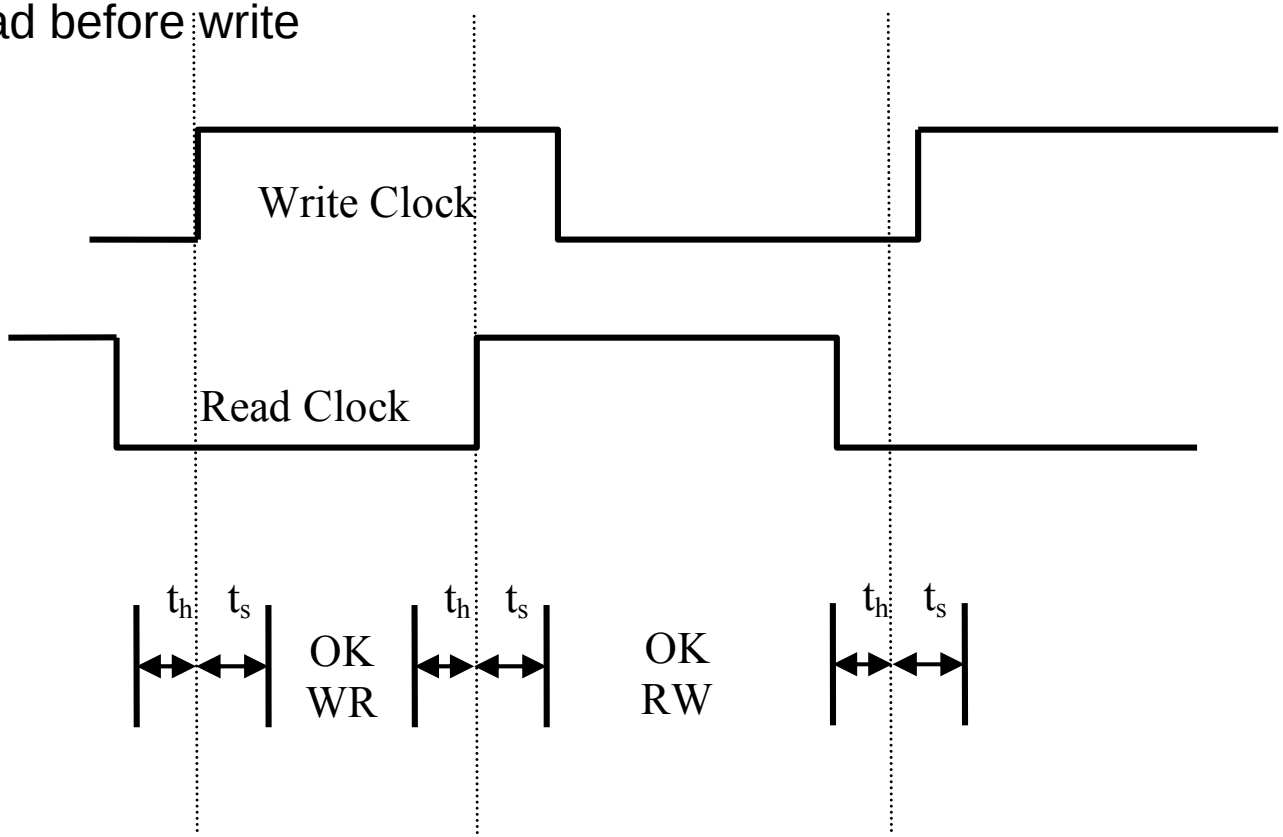
Mesochronous data exchange

- Intermediate X register used to retime data
- Need to find a place where write data is stable, and read register available
  - Greenstreet 2004



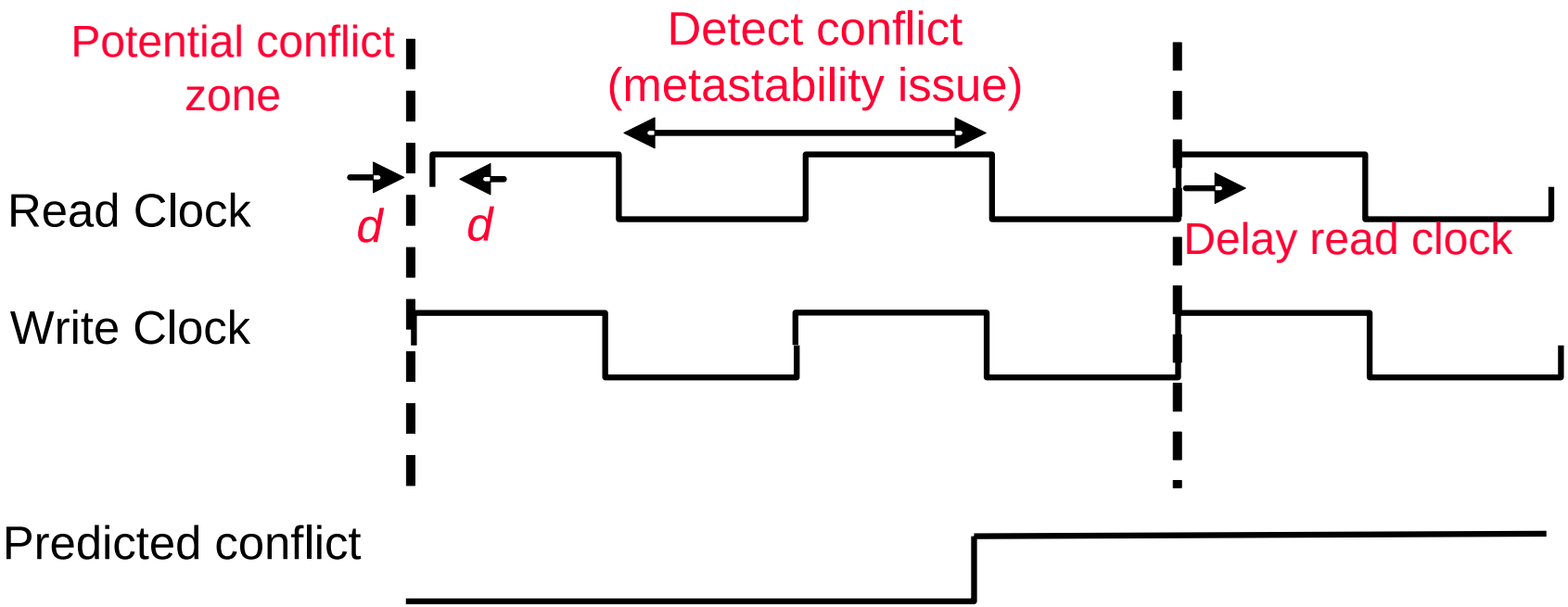
Finding the place to clock X

- provided that tc > 2(th + ts) at least one place is always available for data transfer, but we lose one cycle.
  - Write before read, or
  - Read before write



Pre synchronizing

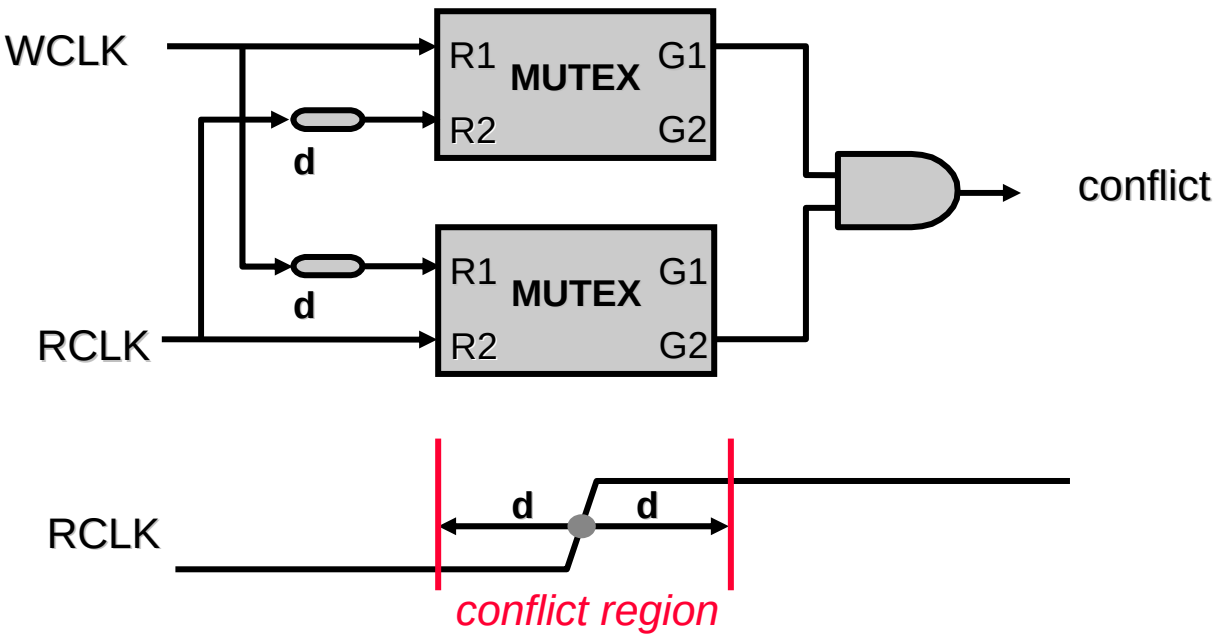
If the phase can vary with time (Plesiochronous), synchronization still need not cause large latencies



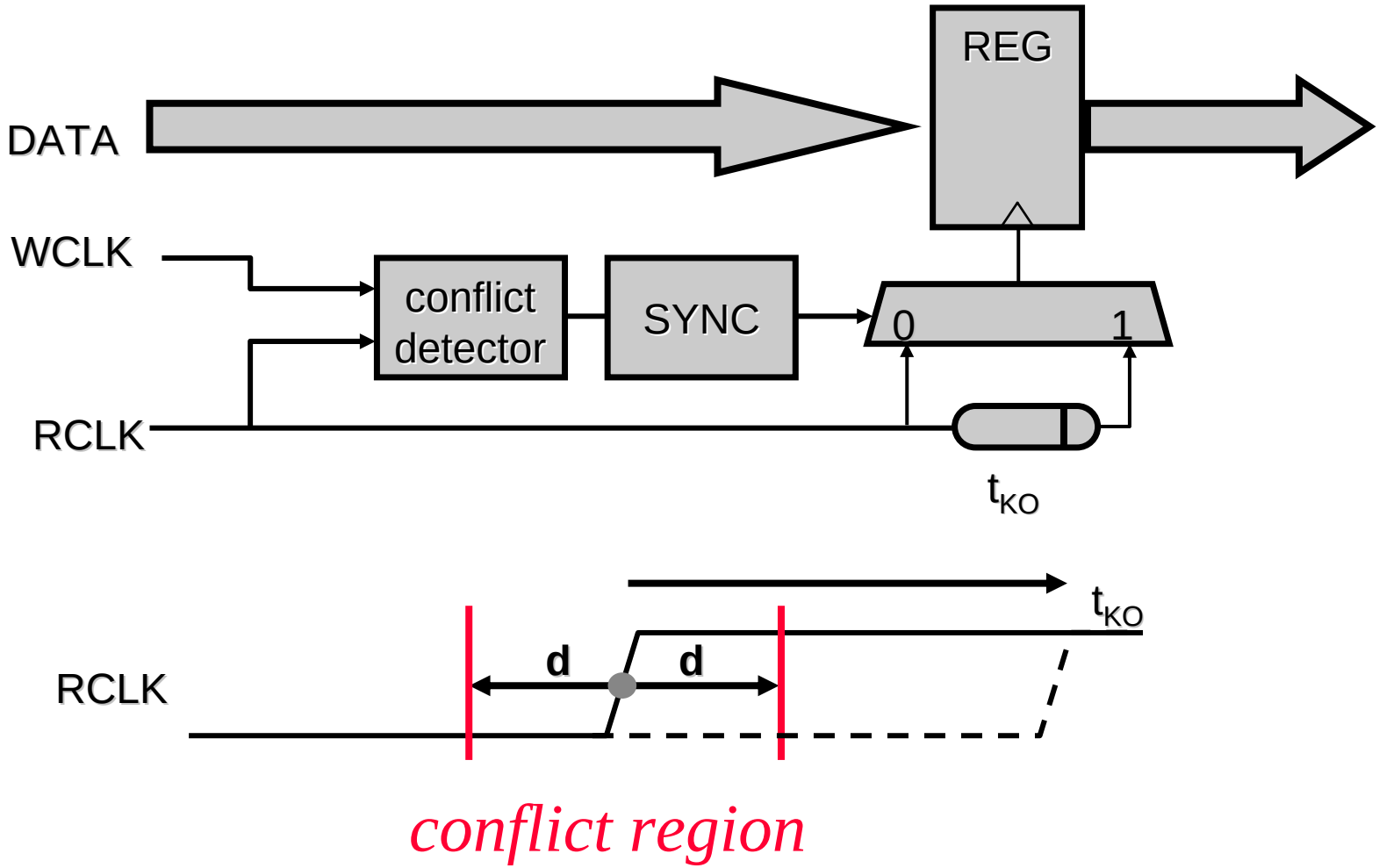
Synchronization problem known in advance

Conflict prediction

- Predict when clocks are going to conflict and delay synchronization
- Dike's conflict detector



Clock delay synchronizer (Ginosar 2004)



Pre synchronizer latency

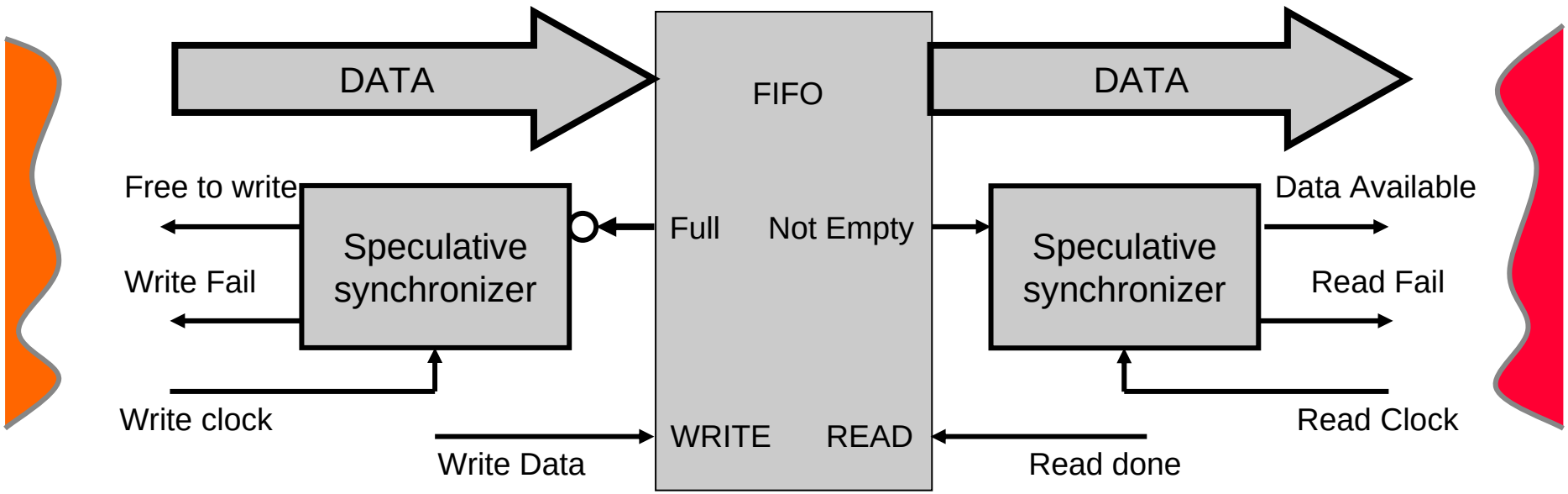
- Nominally 0 – 1 clock cycle
- Relies on accurately predicting conflicts
- Clocks must remain stable over synchronisation time.
- Always lose t_{ko} of next computation stage
- Alternative: shift all conflicts to next read cycle
  - On average this loses 2d
  - 2d must be big enough to cover any clock drift/jitter over synchronization time

Speculation

- Mostly, the synchronizer does not need 30τ to settle
- Only e^{-13} (0.00023%) need more than 13τ
- Why not go ahead anyway, and try again if more time was needed

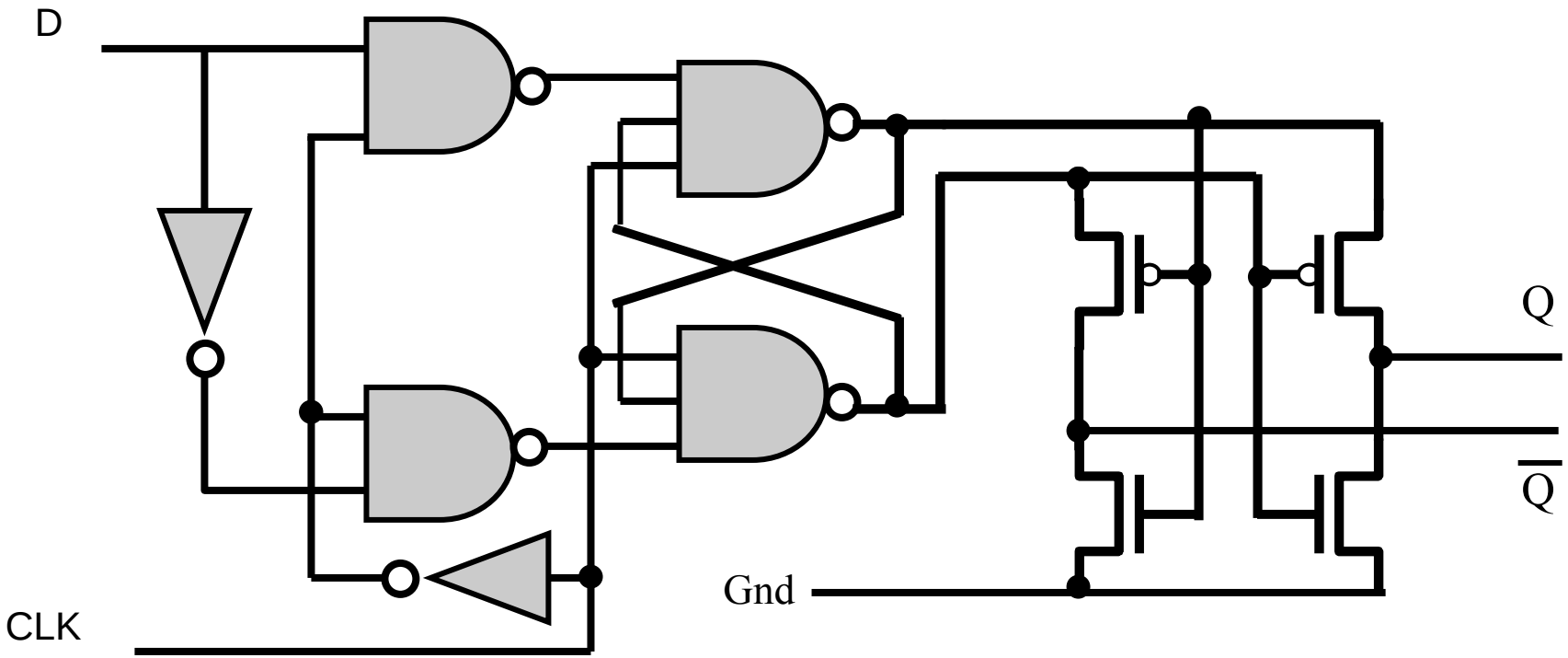
Low latency synchronization

- Data Available, or Free to write are produced early.
- If they prove to be in error, synchronization failed.
- Read Fail or Write Fail flag is then raised and the action can be repeated.



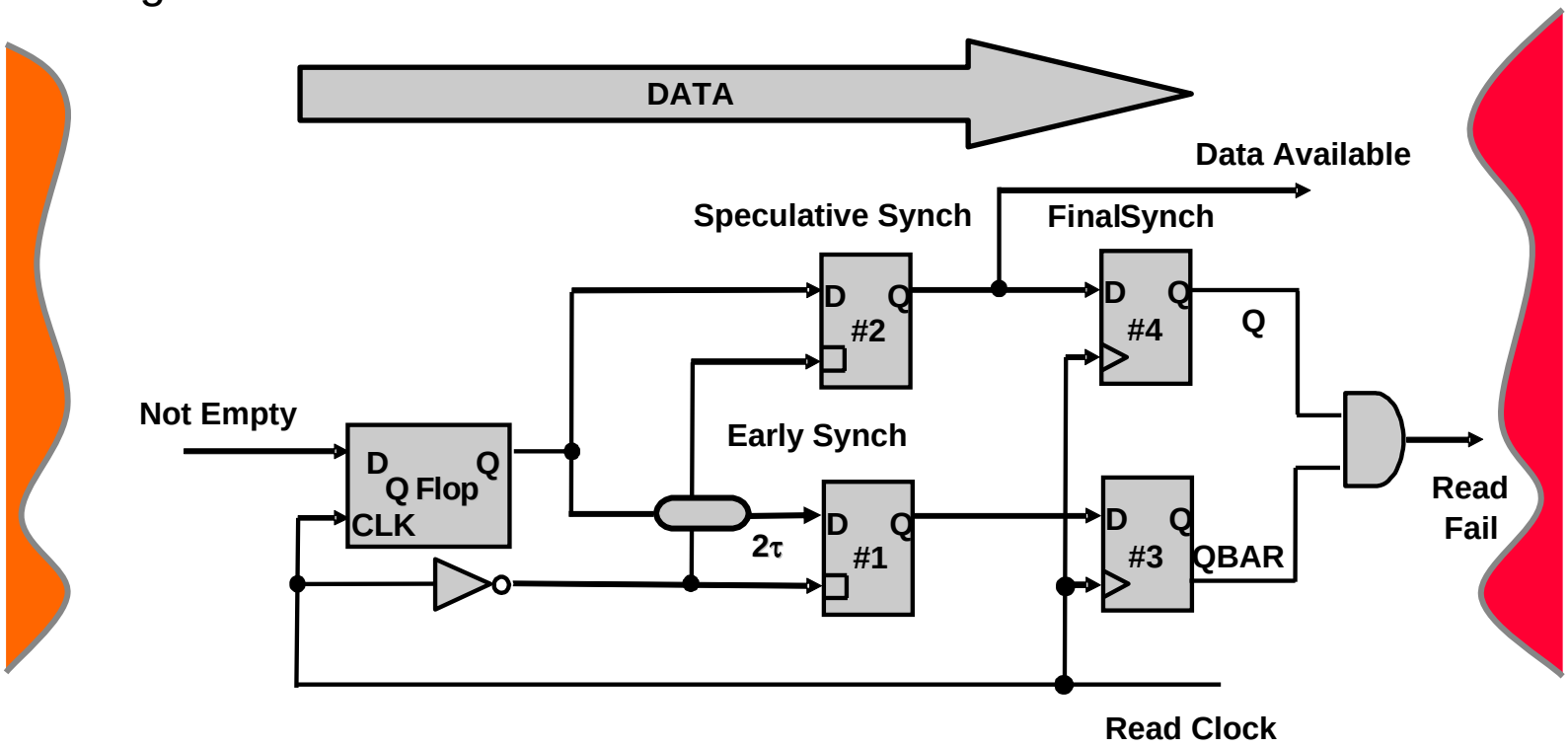
Q Flop

- With CLK low, both outputs are low
- With CLK high, Q becomes equal to D only after metastability
- Q and Qbar are both low until metastability resolved
- We can detect events that take longer than a half cycle



Was it OK?

- FF#1 is set after a half cycle - 2τ, FF#2 after a half cycle, FF#3 at a full cycle
- Latency is normally half a cycle = 15 τ, but synchroniser fails often
- By the time we look at the Read Fail signal (a full cycle = 30τ) all signals are stable



When to recover

Early FF1 Half Cycle – 2/13τ	Speculative FF2 Half Cycle/15τ	Fail FF3, 4 End of Cycle/30τ	Comment
?	?	metastable?	Unrecoverable error, Probability low.
0	0	0	No data was available
0	1	1	Stable at the end of the cycle, but the speculative output may have been metastable. Return to original state
1	1	0	Normal data Transfer

Speculative Synchronisation latency

- Recovery means restoring any corrupted registers, and may take some time,

BUT

- Probability of recovery operation is e^{-13}, so little time lost on average.

- Can reduce average synchronization latency from one cycle to a half cycle

Comments

- Synchronization/arbitration requires special circuit elements
- They’re not digital!
- If there’s a real choice, and bounded time you will have failures.
- The MTBF can be made longer than the life of the universe
- Design gets more difficult with small dimensions
- Latency is a problem, but not insuperable.
- Synchronizers are not deterministic.

Outline

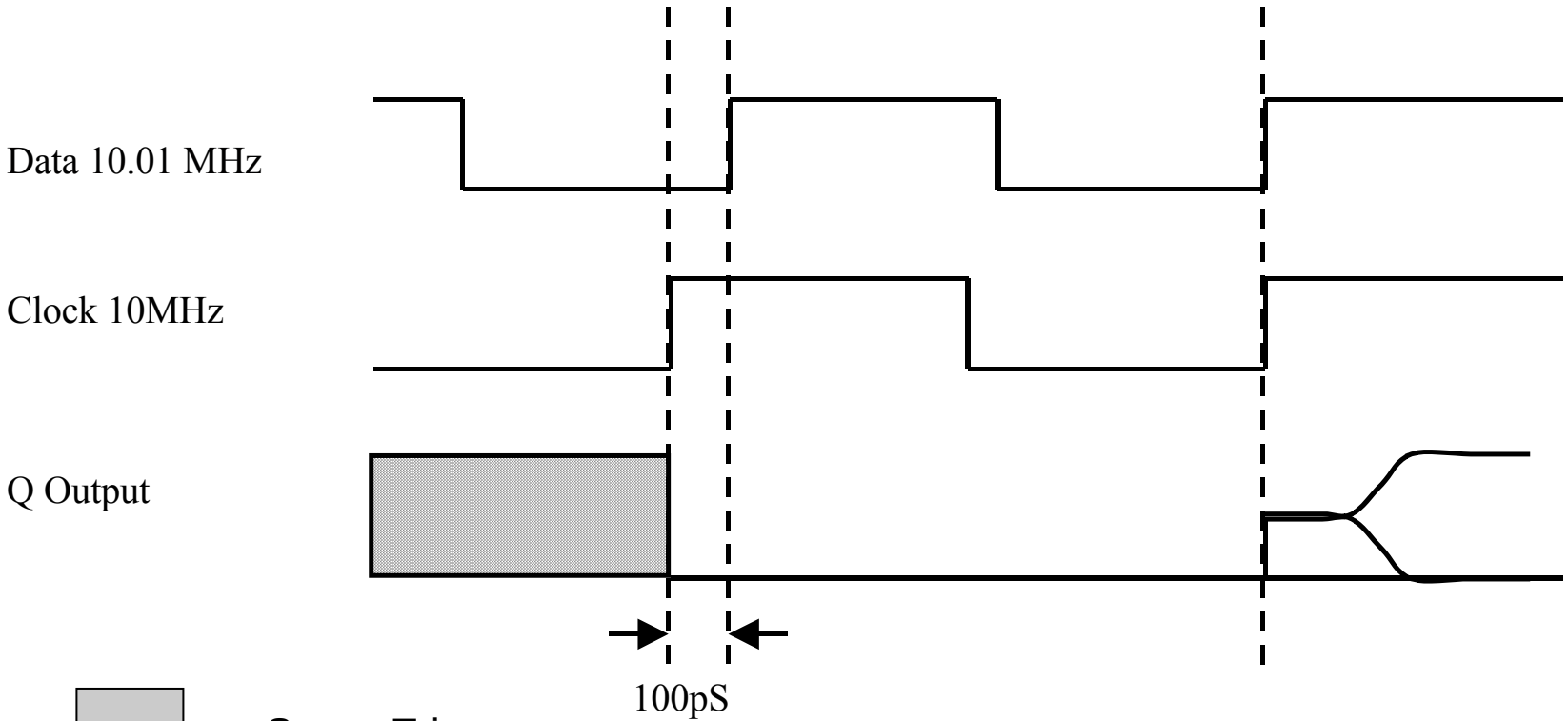
- What's the problem? Why does it matter?
- Synchronizer and arbiter circuits
- Noise, and its effects
- Latency, and how to overcome it

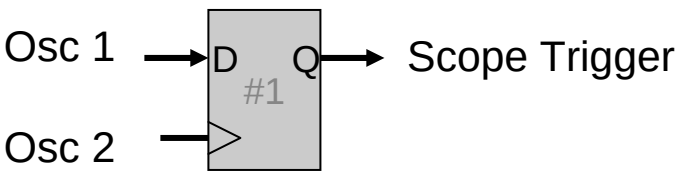
- Metastability measurement 1
  - Simple measurements

- Arbitration

- Metastability measurement 2
  - Second order effects (Which may matter)

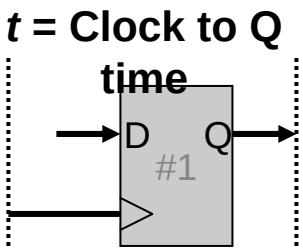
Testing synchronizers

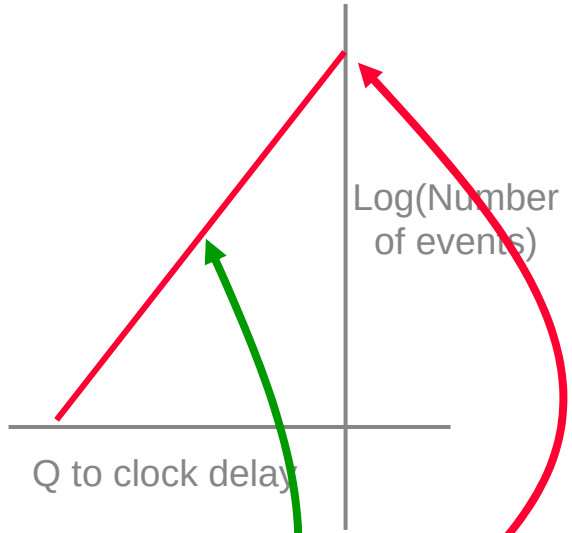




- **Data and Clock are asynchronous**
- **Q only changes if Data and clock edges are within 100ps (1 in 1000)**

Event histogram

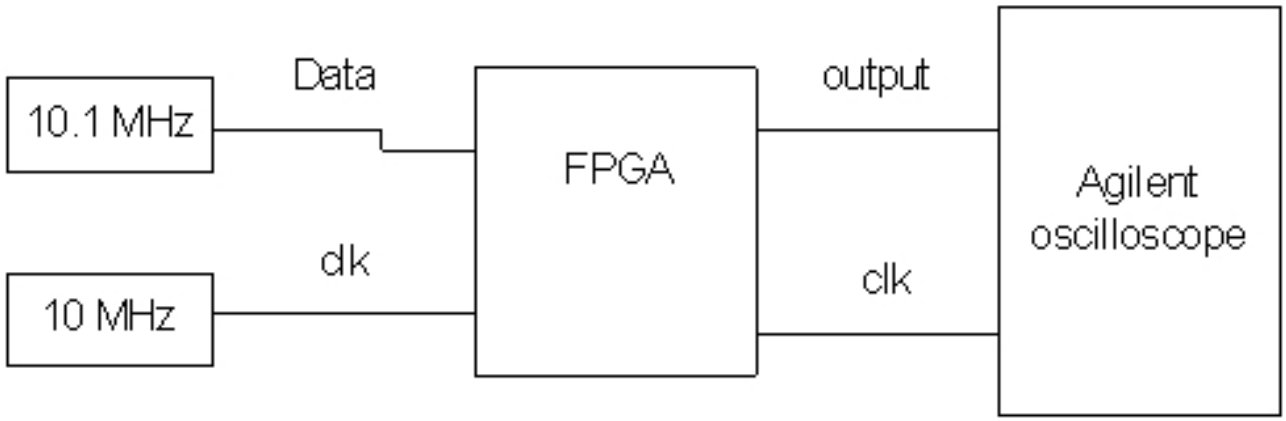




- Trigger from Q going high
- Observe clock, so scale is negative
- Log scale of events because

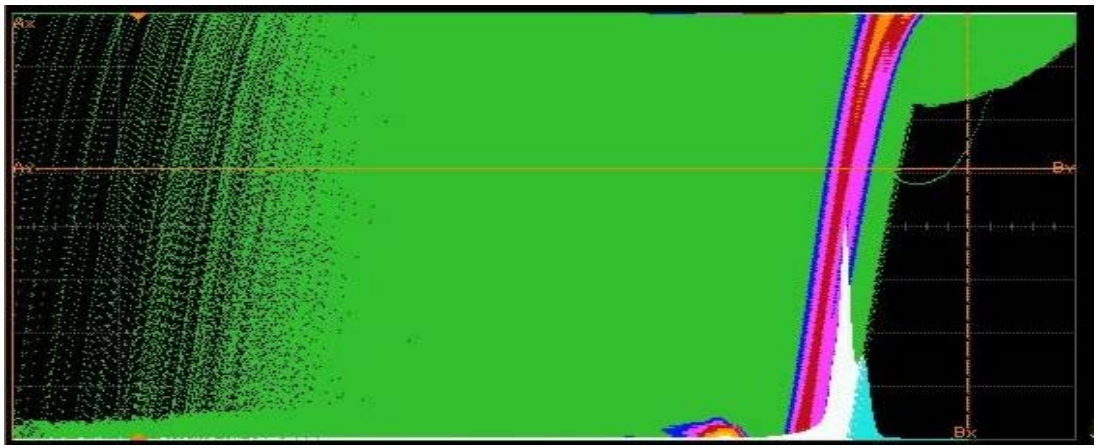
$$Events = \frac{T_{Elapsed}}{MTBF} = T_{Elapsed}.T_w f_c f_d e^{-t/\tau}$$

Experimental measurement set-up



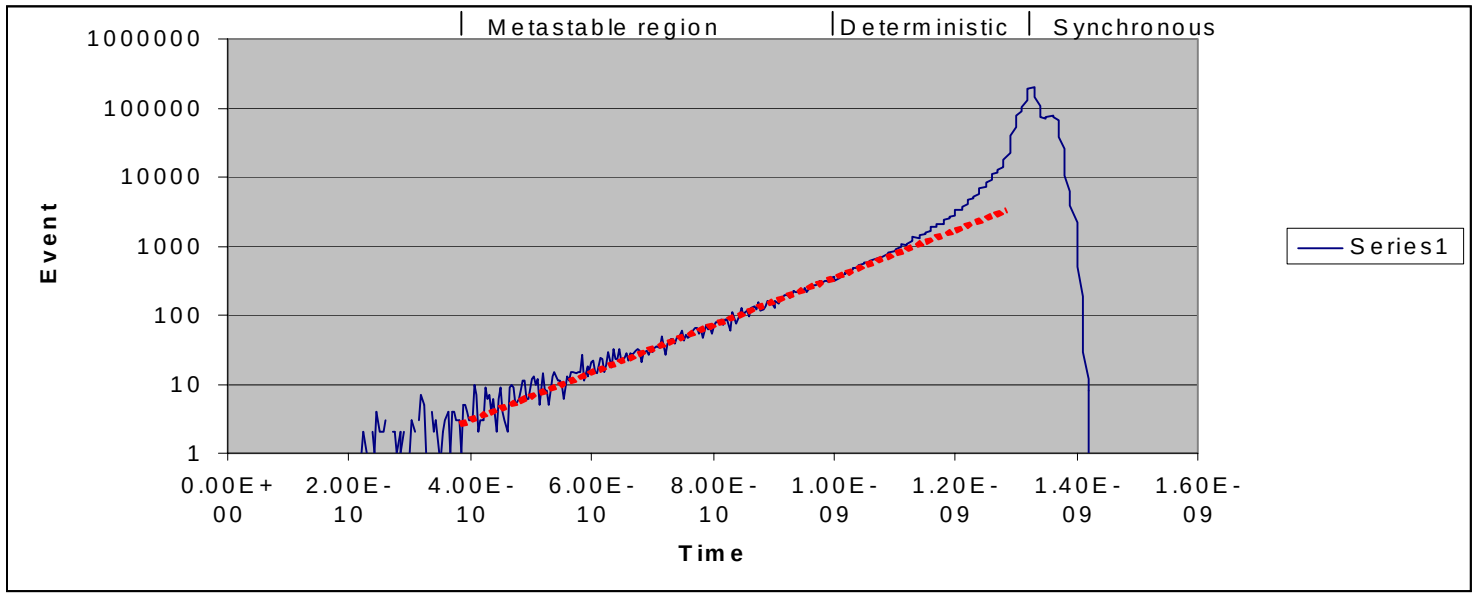
- Two asynchronous oscillators are used to drive the data and the clock inputs of a D-type edge triggered Flip-Flop.
- With the slight difference in the frequency of the two oscillators, the clock rising edge may or may not produce a change in the Q output.
- Oscillators should produce constant probability of Data – Clock change with time (But may not: Cantoni 2007)

Altera FPGA measurements



- An Altera FLEX10K70 used here, manufactured in a 0.45μm CMOS process.
- The events collected over a period of 4 hours.
- To calculate the value of τ (resolution time constant), the histogram of the trace density can be plotted in semi-log scale.

Altera FPGA plot



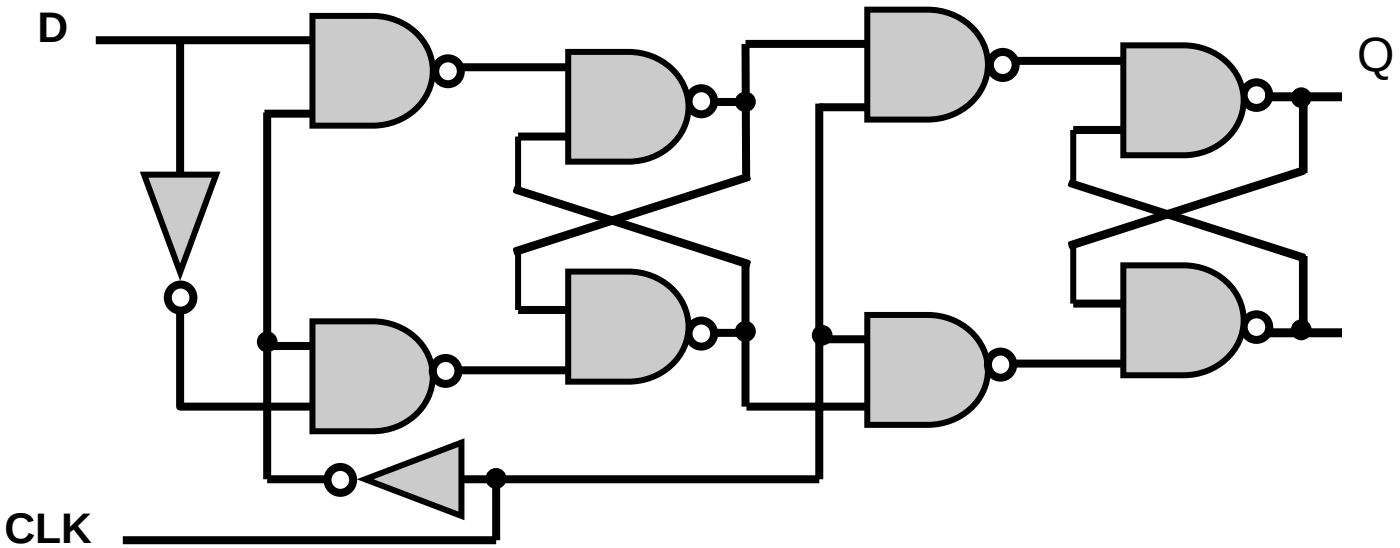
- The X-axis represent time from a triggering Q output back to the clock edge. Therefore increasing metastability time is shown from right to left.

- Here τ = 120ps

Points to consider

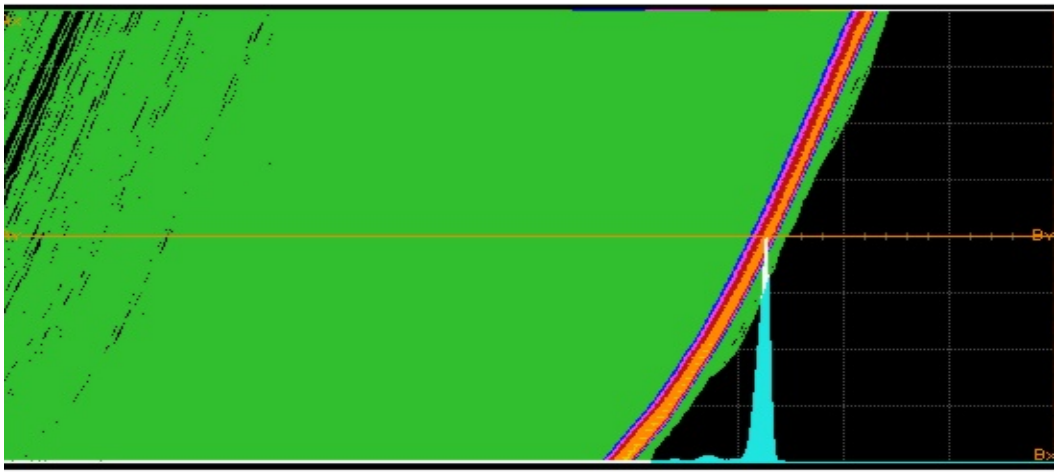
- This type of measurement depends on
  - Uniform distribution of clock data overlaps
  - Often not true because the oscillators affect each other (Cantoni 2007)
- Uses an expensive oscilloscope to do the histograms
  - You don’t HAVE to use one. Counters and delays will do
- The theory only applies to simple FFs
  - FFs need to be predesigned, or laid out in a small area

Measurements in a bistable element



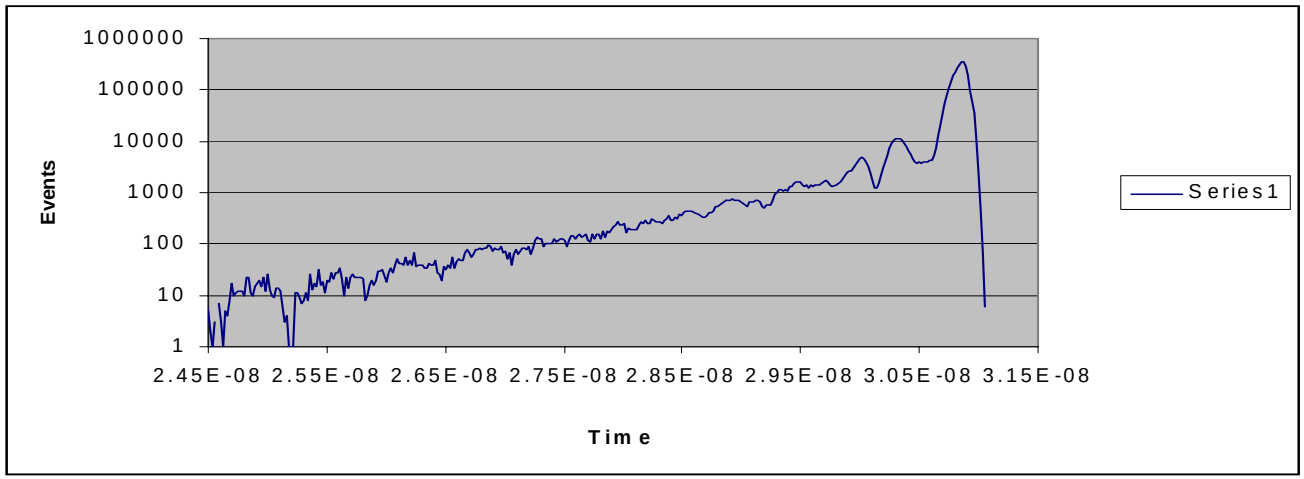
- A D-type edge triggered Flip-Flop constructed using NAND gates on the Altera FPGA.
- The master and the slave were placed very close to each other.

Components are close, but not in same cell



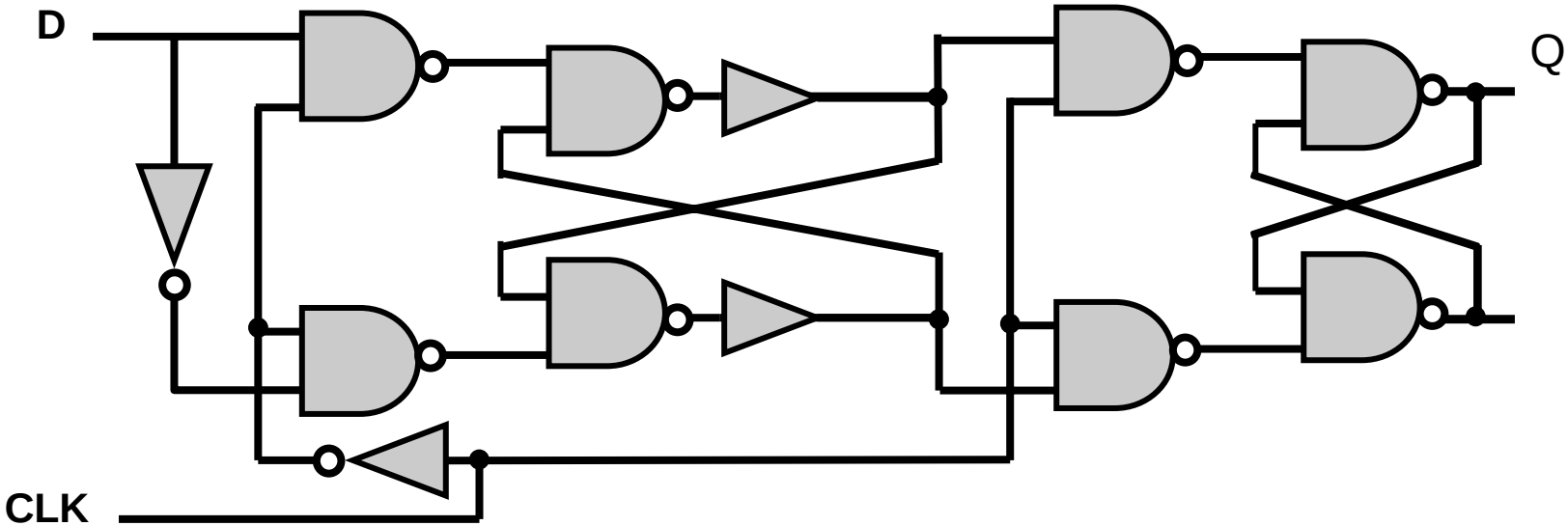
- Routing delays play significant role in this experiment.
- Long metastability times due to the feedback loop delays .

Measurements in a bistable element (cont.)



- From the histogram, a damped oscillation in the deterministic region can be observed.
- The value of τ is in the order of 5 nanoseconds, making this particular design unsuitable for any application.
- Circuits with feedback loops passing through LUTs can exhibit oscillation.

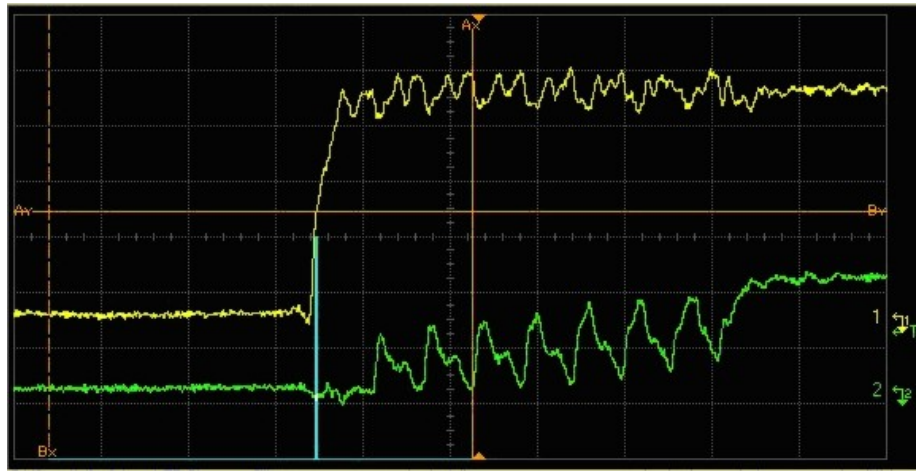
Too much delay

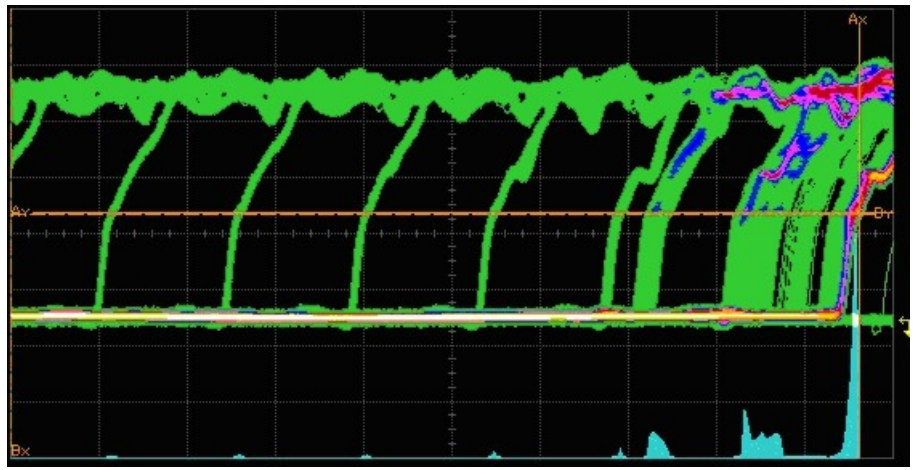


- It may not be easy to place elements close to each other
- Extra delay can cause the loop to become unstable

Complex response

- We put an extra gate in the feedback loop of the master FF here
- So the output oscillates, and causes ripples in the histogram
- Time between cycles is about 3ns, so you get lots of outputs at more than 20ns
- Demo by Nikolaos Minas





Outline

- What's the problem? Why does it matter?
- Synchronizer and arbiter circuits
- Noise, and its effects
- Latency, and how to overcome it

- Metastability measurement 1
  - Simple measurements

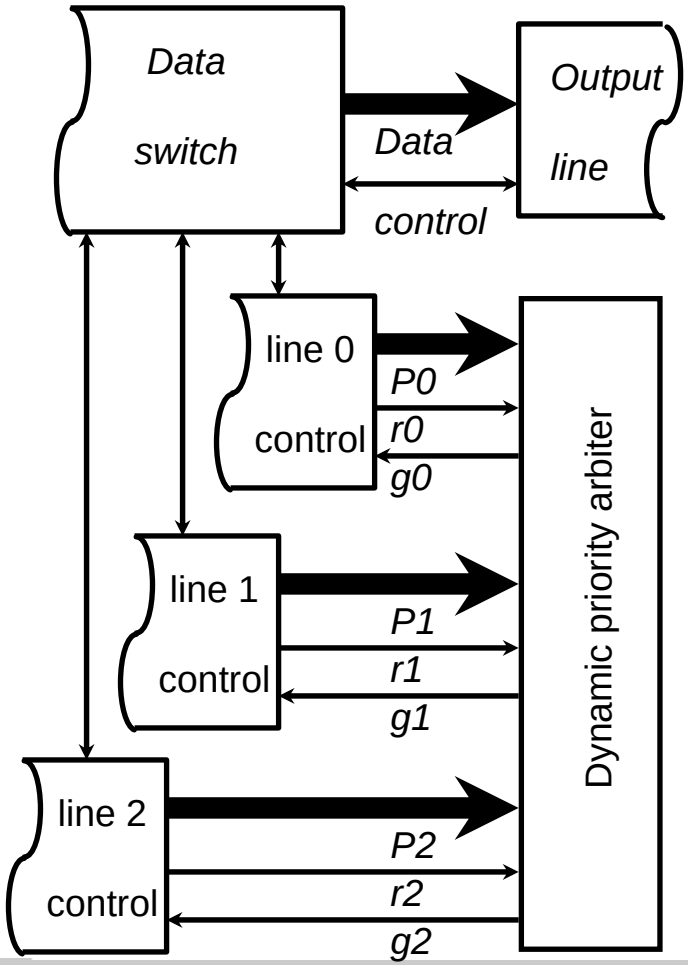
- Arbitration

- Metastability measurement 2
  - Second order effects (Which may matter)

11:00 – 12:00

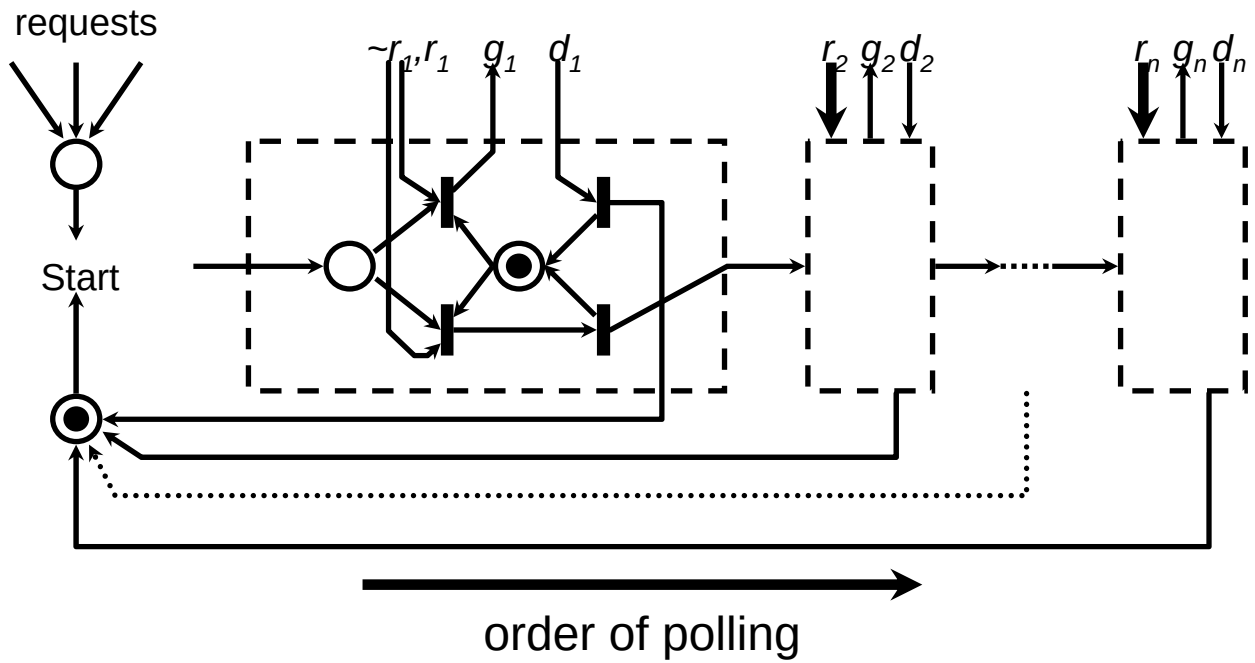
Arbitration

- Complex systems may require that some requests overtake others
- Here three input channels require access to a single output port
- Each request may have a different priority
- Priority can be topologically fixed, or determined by a function



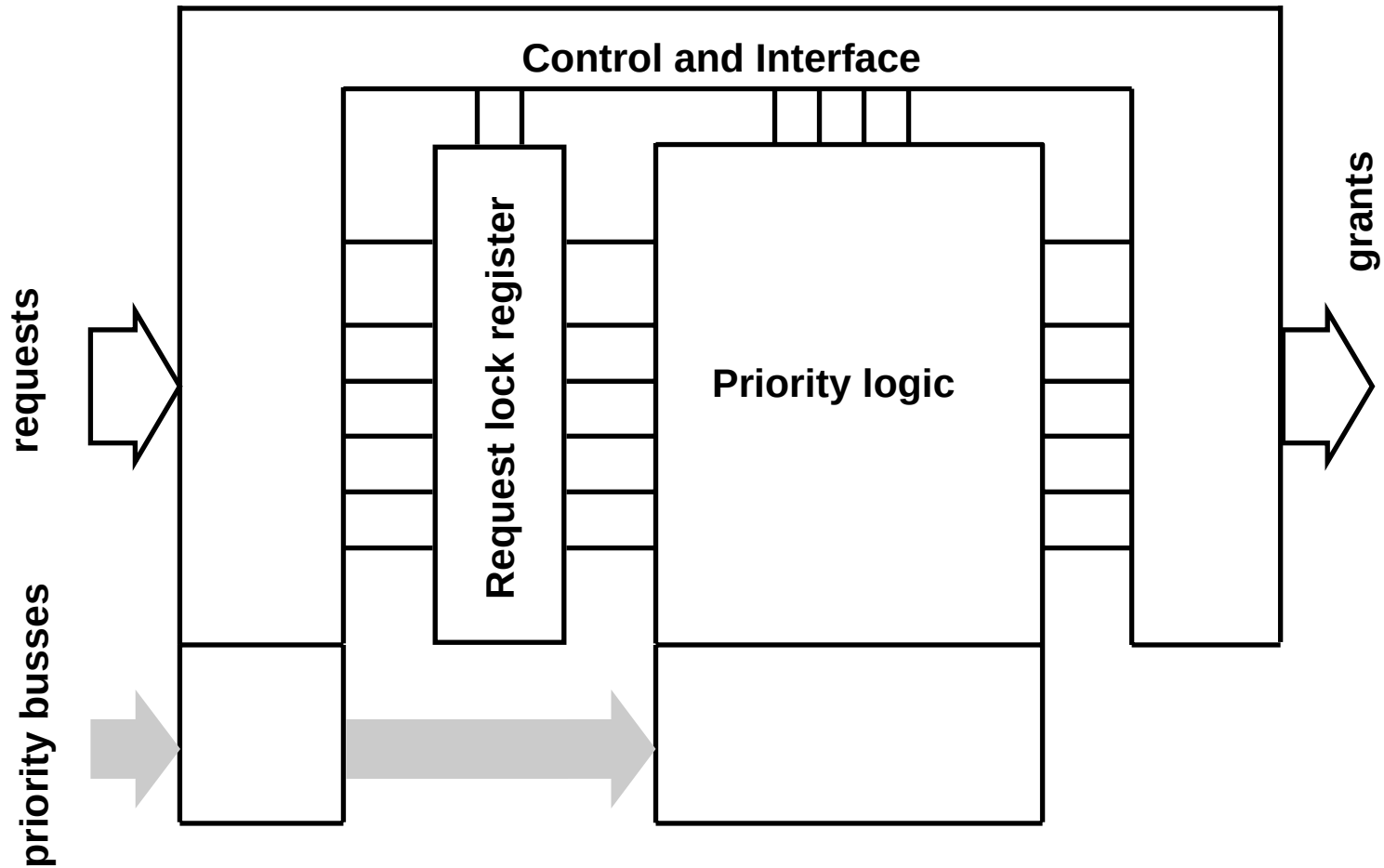
Types of arbiter

- Topologically fixed
  - priorities determined by structure, e.g. daisy-chain



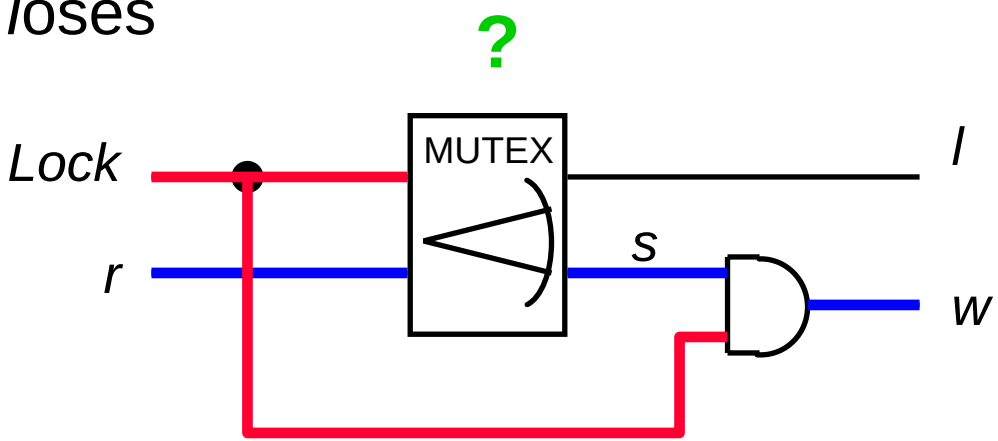
- Static or dynamic priority
  - determined by fixed hardware, or priority data supplied

Static or dynamic priority



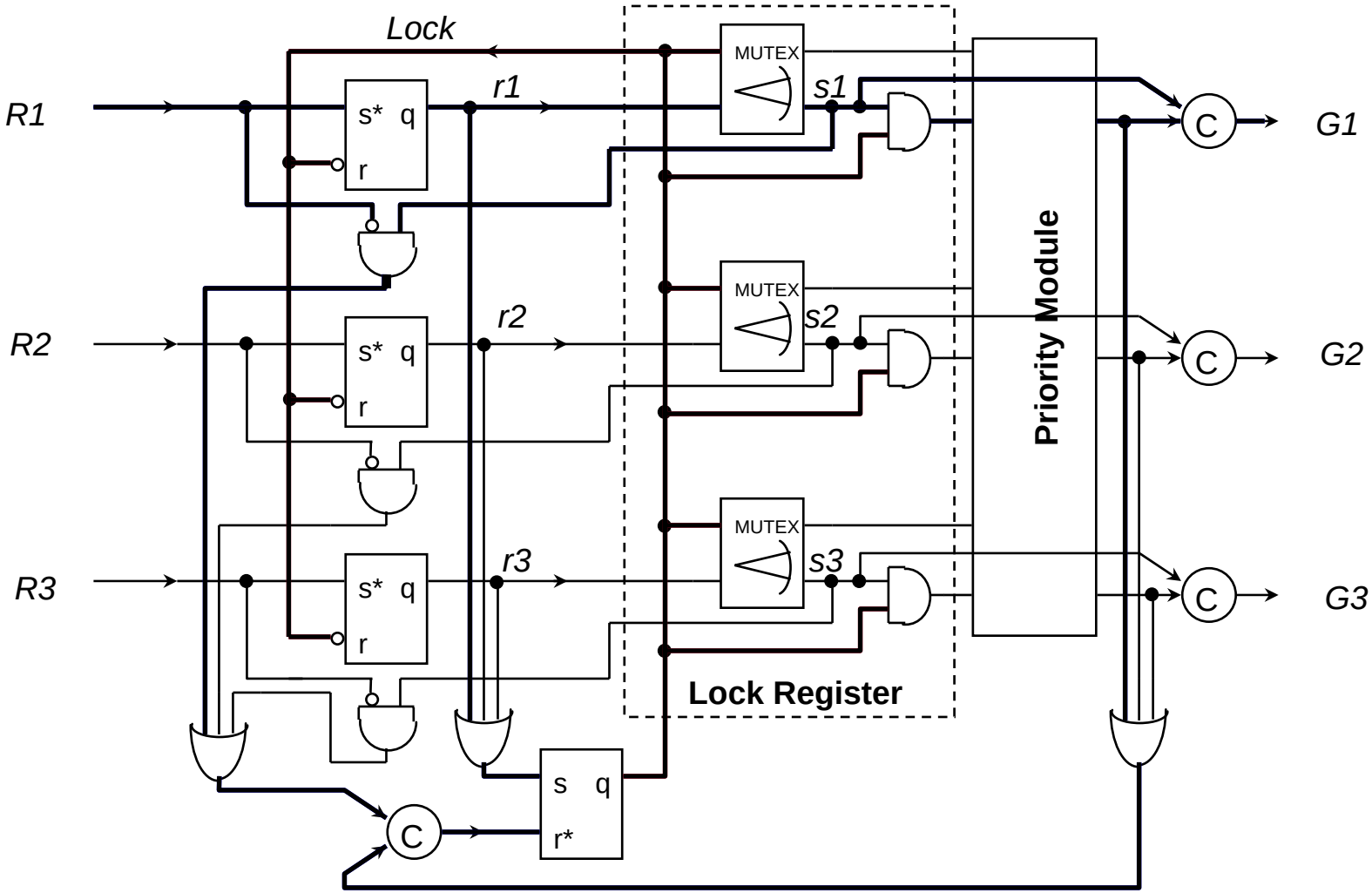
Metastability and priority

- Lock the request pattern
  - incoming requests cause *Lock* to go high
  - following MUTEX ensures that request *w*ins or *l*oses



- Evaluate priorities with a fixed request pattern

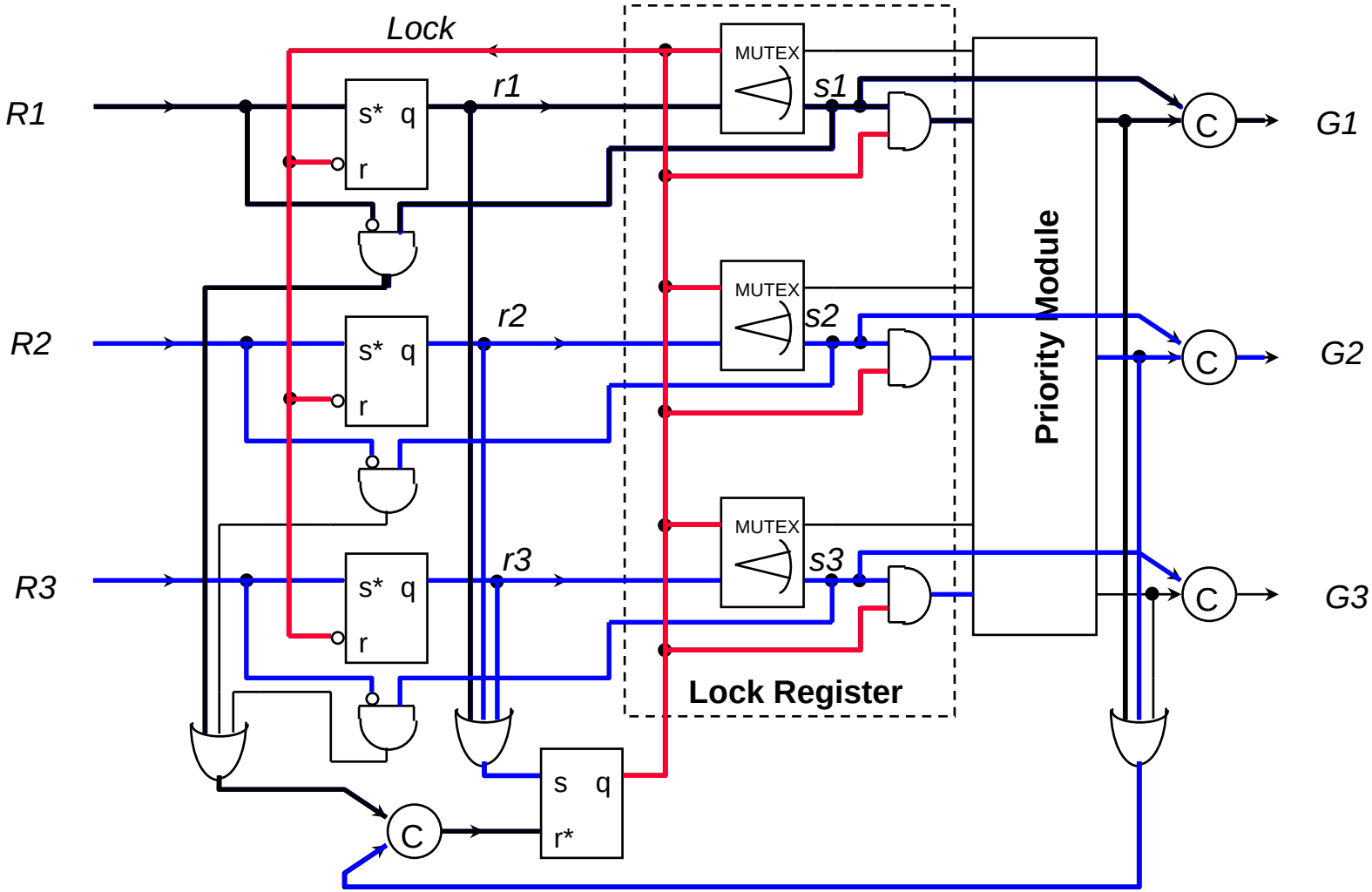
Static priority arbiter



More than one request

- Priority needed if requests are competing
- Shared resource free
  - resolution required only if second request arrives before the lock signal due to first request
- Shared resource busy
  - Further requests may accumulate, and one may be higher priority

Two more requests



Outline

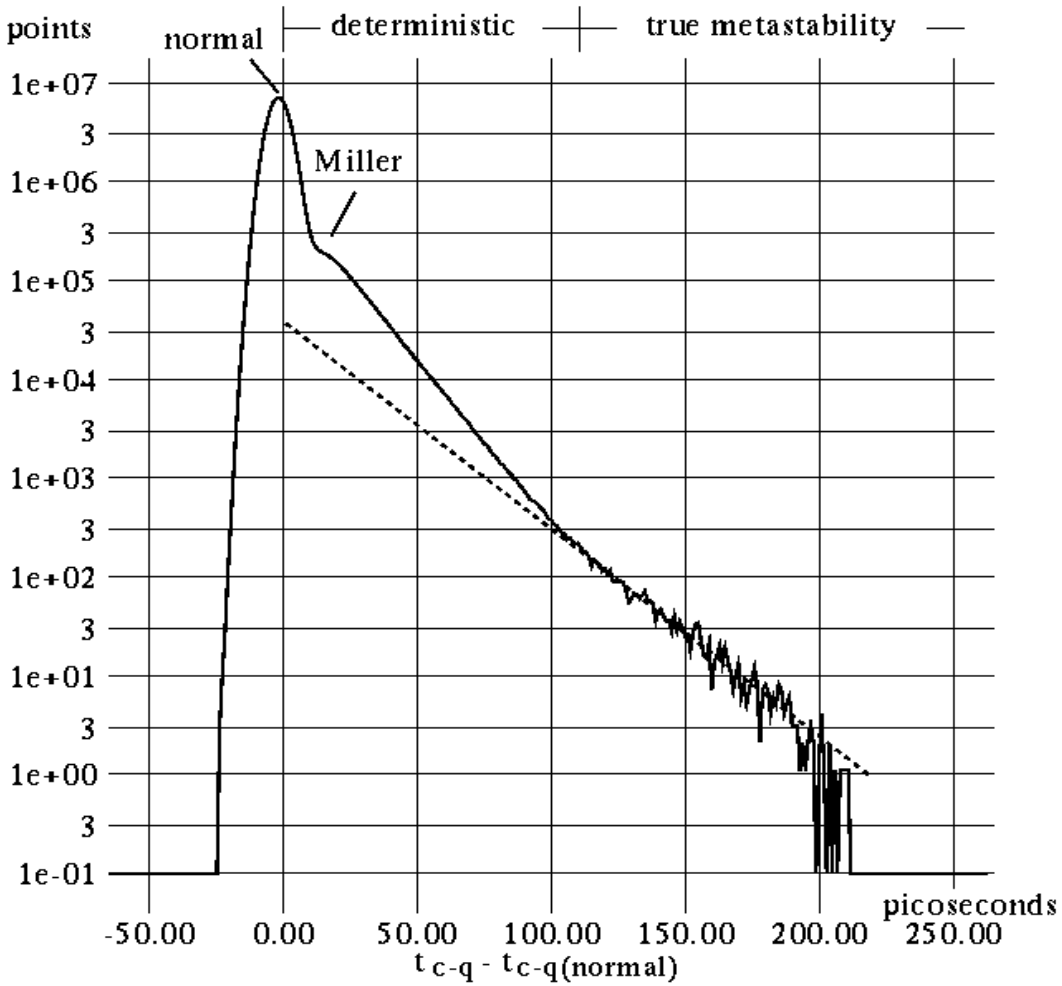
- What's the problem? Why does it matter?
- Synchronizer and arbiter circuits
- Noise, and its effects
- Latency, and how to overcome it

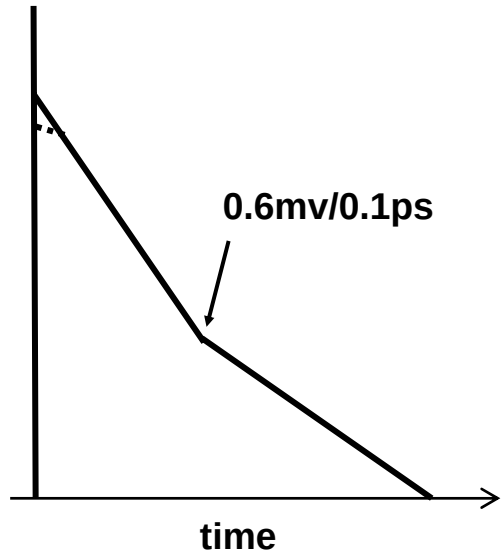
- Metastability measurement 1
  - Simple measurements

- Arbitration

- Metastability measurement 2
  - Second order effects (Which may matter)

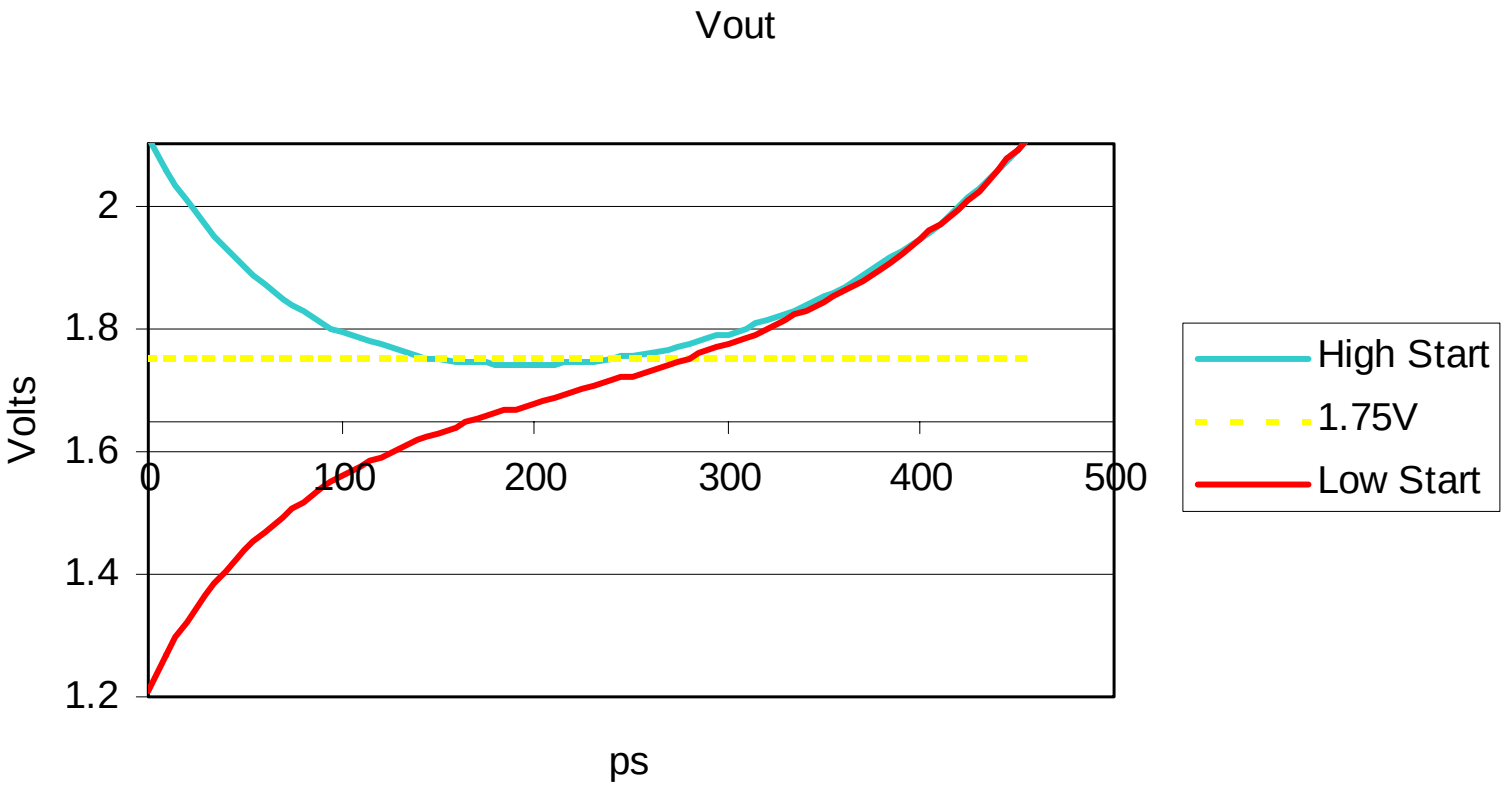
Measured Histogram





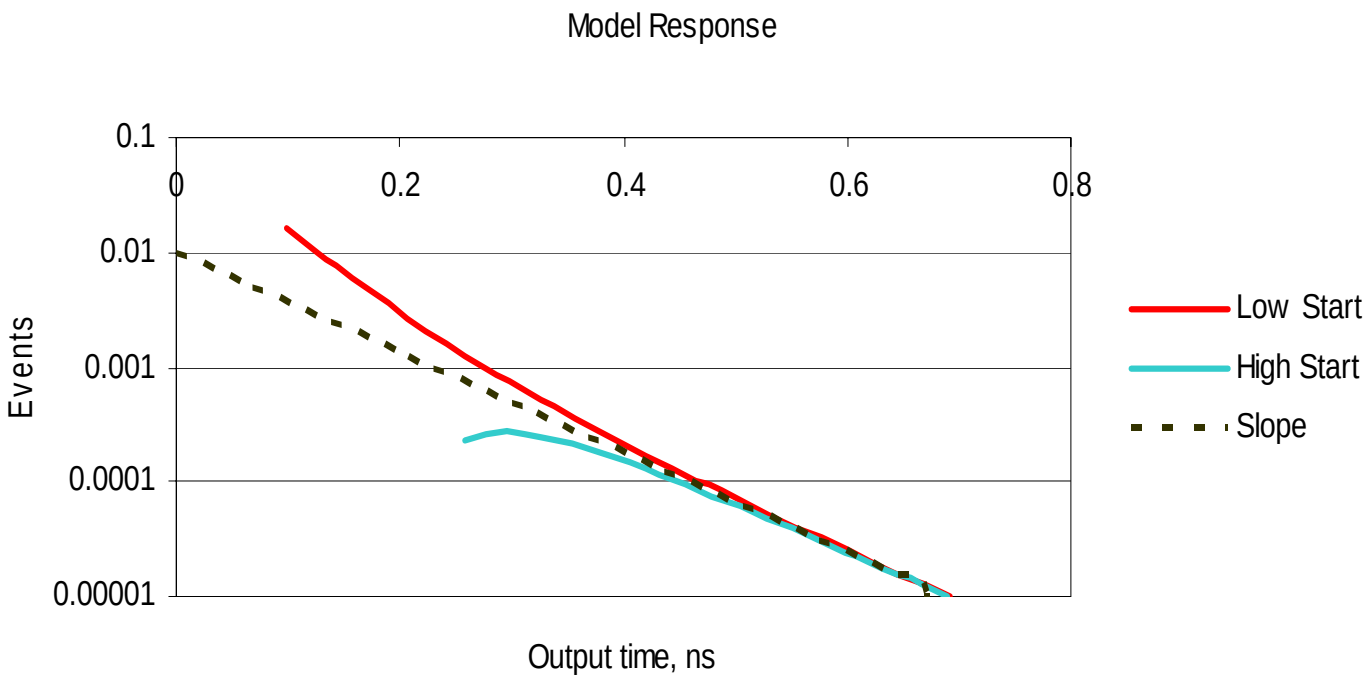
0.6mV is about the level of thermal noise on a node in 0.18μ

Why isn't it straight?



- The starting point makes a difference
- Early events are more affected than late ones

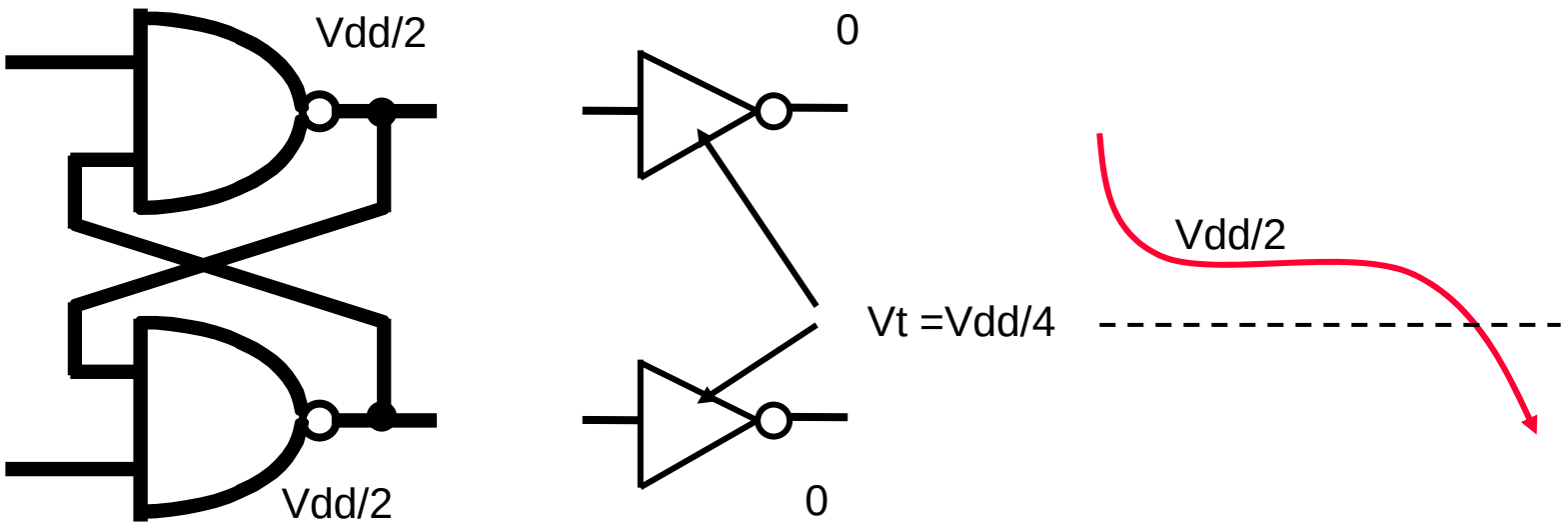
Histogram of events



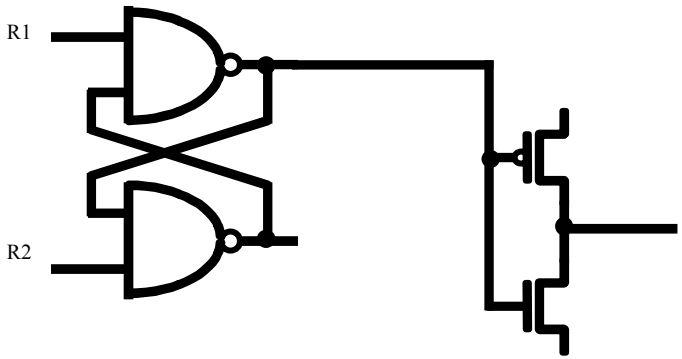
- Probability of an event occurring within 10ps of a particular output time

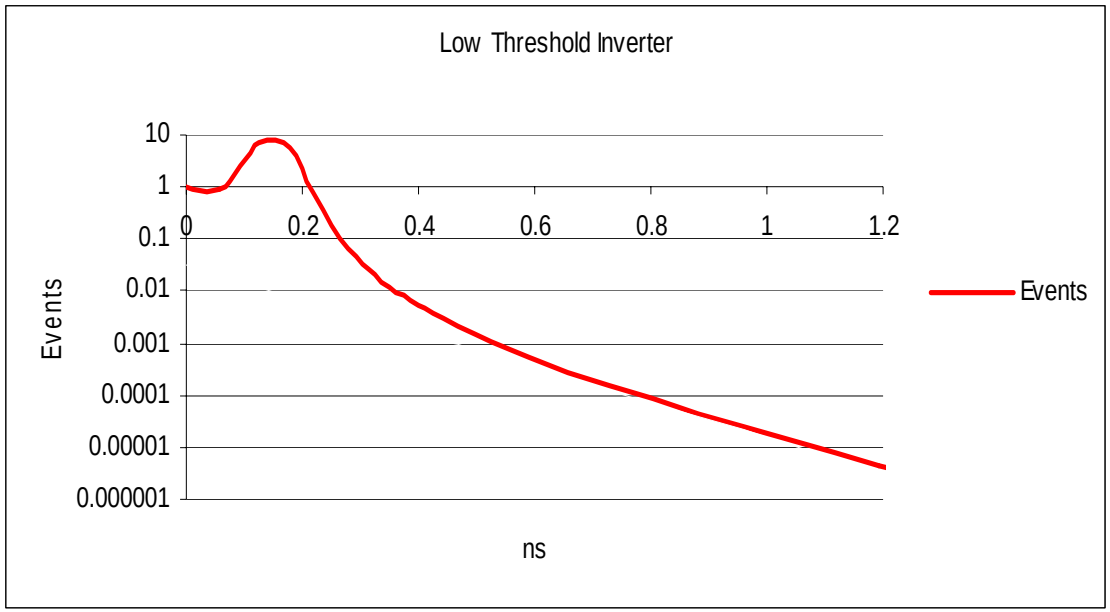
Metastability filters

- Affect response
- Inverters usually have a threshold close to the metastability level



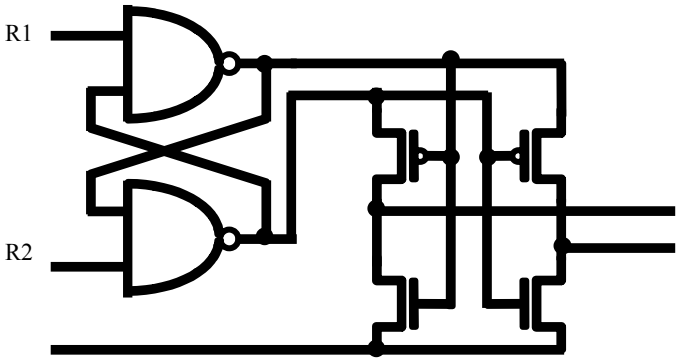
MUTEX with low threshold output

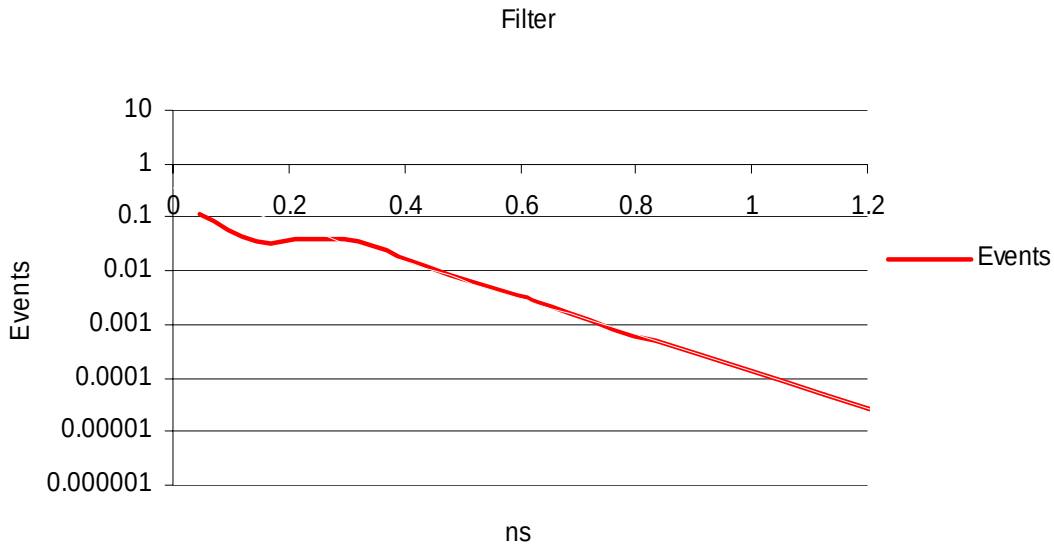




- Starts high, needs to go low to give output
- Threshold about 100 mV low

MUTEX with filter



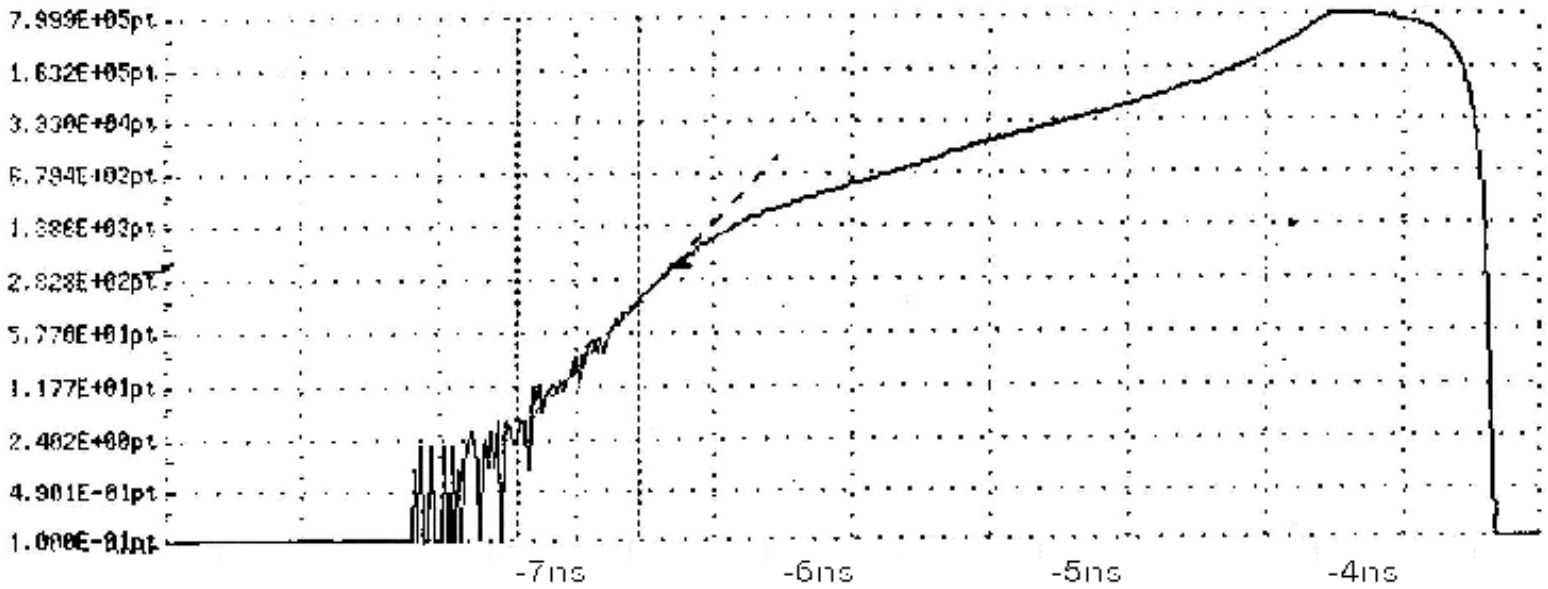


- Needs more than 1V difference to give output
- Slower, but slope more constant

What we know

- Things we know
  - Synchronizers are unreliable, the more there are the more unreliable the system
  - How to measure reliability up to a few hours
- Things we know we don’t know
  - What reliability is at 3 years
  - How to measure it
  - Complex circuits give complex results, the simple MTBF formula may not apply
- Things we don’t know we don’t know
  - What happens on the back edge of the clock

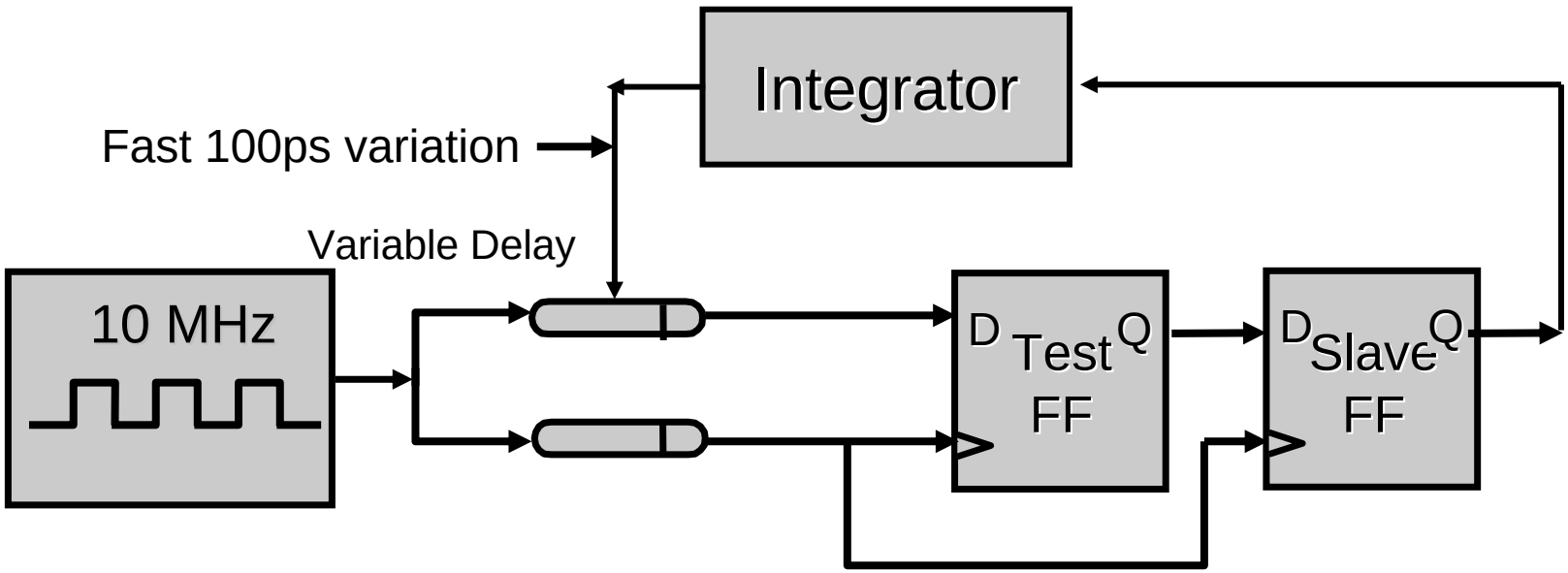
74F5074 Histogram



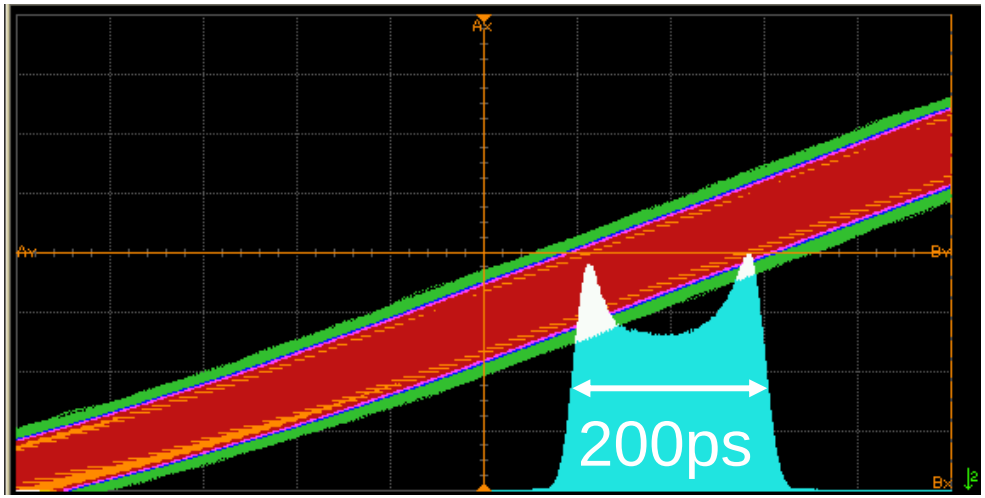
- Slope, τ, is about 120ps (in fast region)
- Typical delay time (most events) is 4ns
- 99.9% of clock cycles do not cause useful events
- To get 1 event at 7ns requires hours

Increasing the number of events

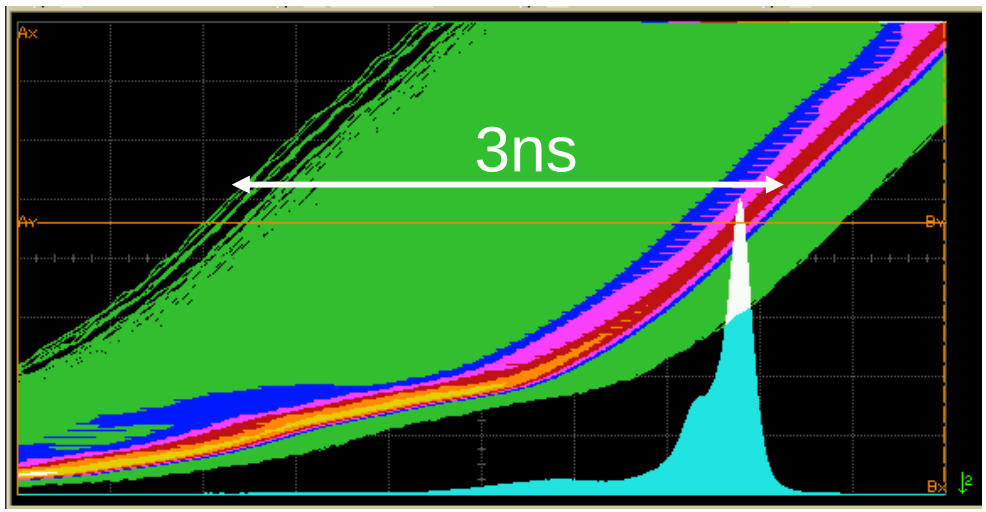
- Test FF is driven to metastability
- Every clock produces a metastable response
- Integrator ensures half outputs high, half low



What you get

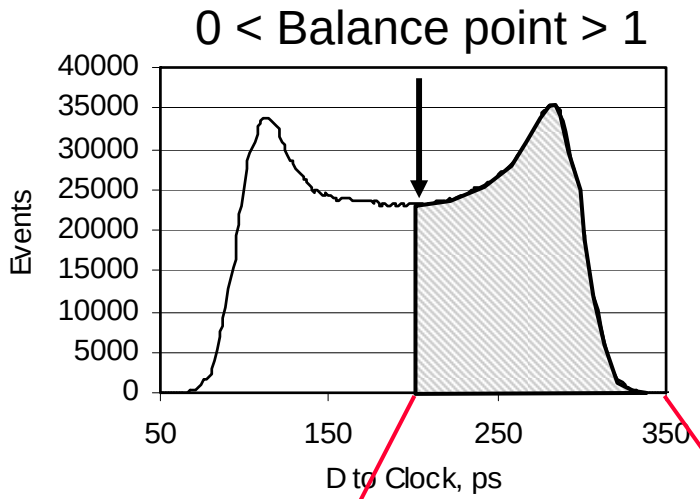


- Clock to D (Input) histogram



- Q to Clock (Output) histogram

Interpreting results

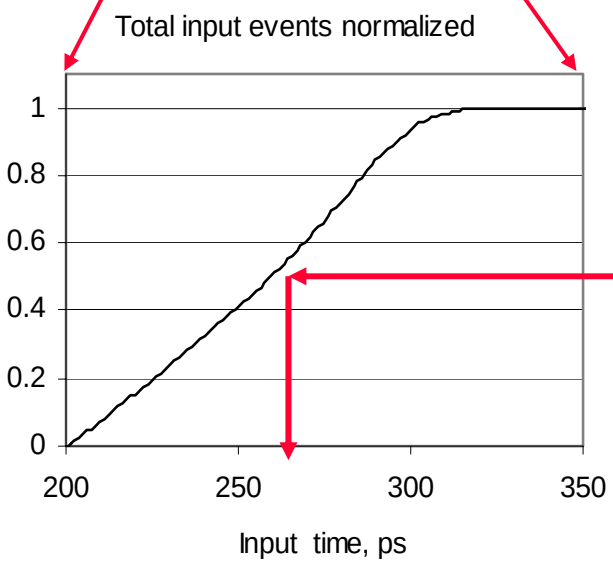


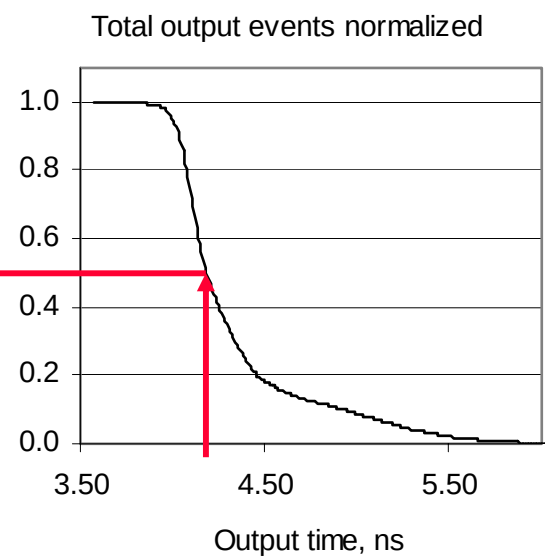
Input time distribution is not flat

Proportion of total inputs causing events vs input time

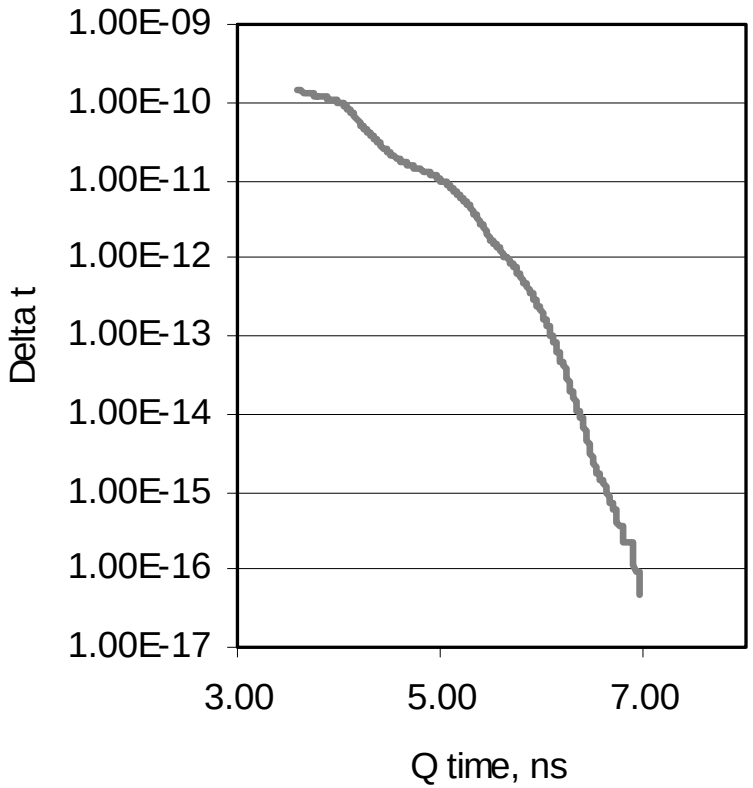
Proportion of total output events vs output time

Mapping output times to input times





100ps variation

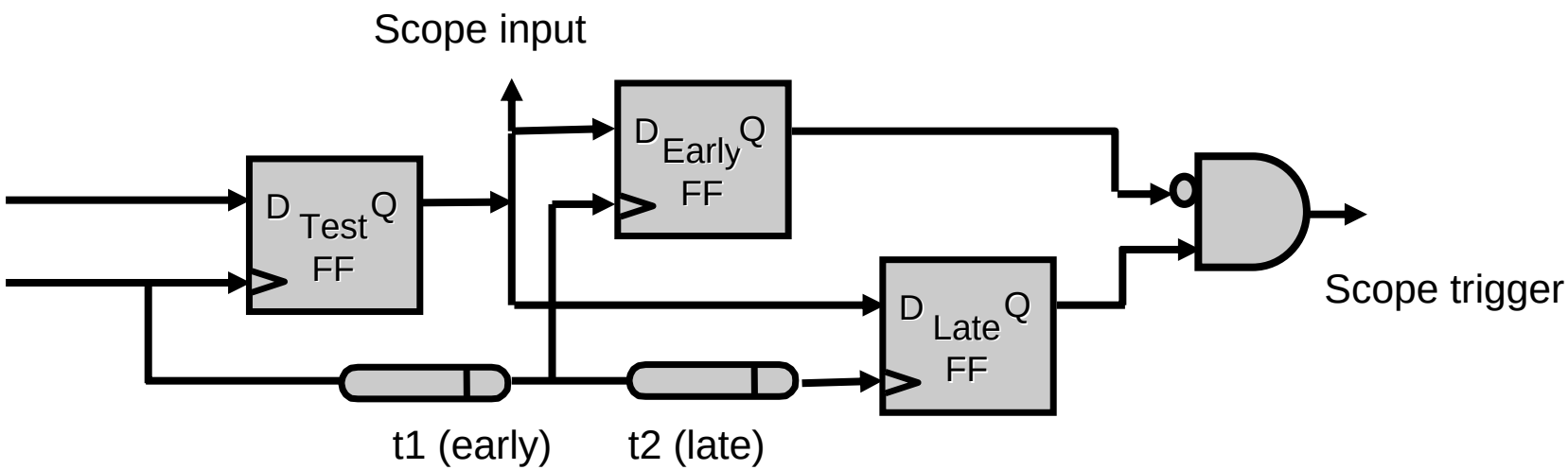


- Δt is the time from the "balance point" of ~200ps
- Similar to original graph BUT Δt not events
- Much quicker to gather data
- Reliability results days not minutes
- Δt does not depend on f_c and f_d or measurement time. Events do

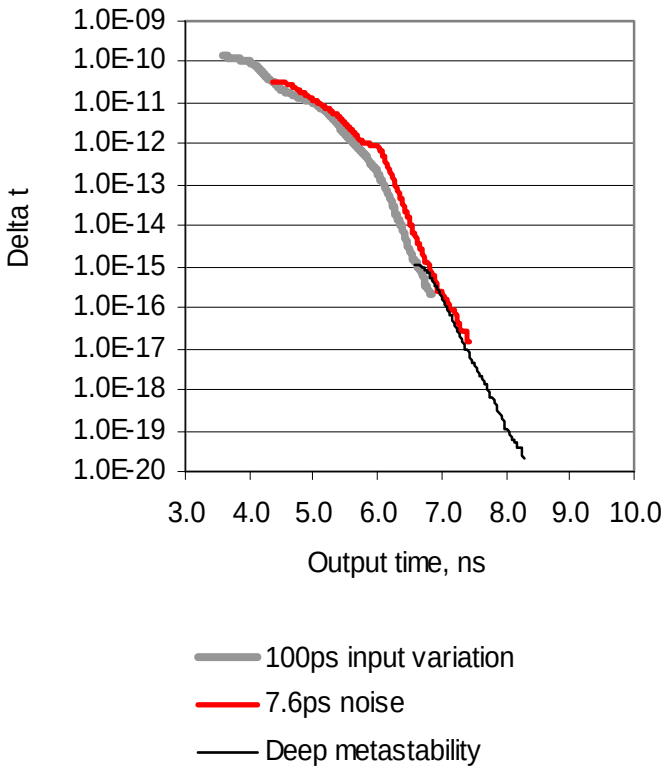
$$MTBF = \frac{1}{\Delta_t f_c f_d}$$

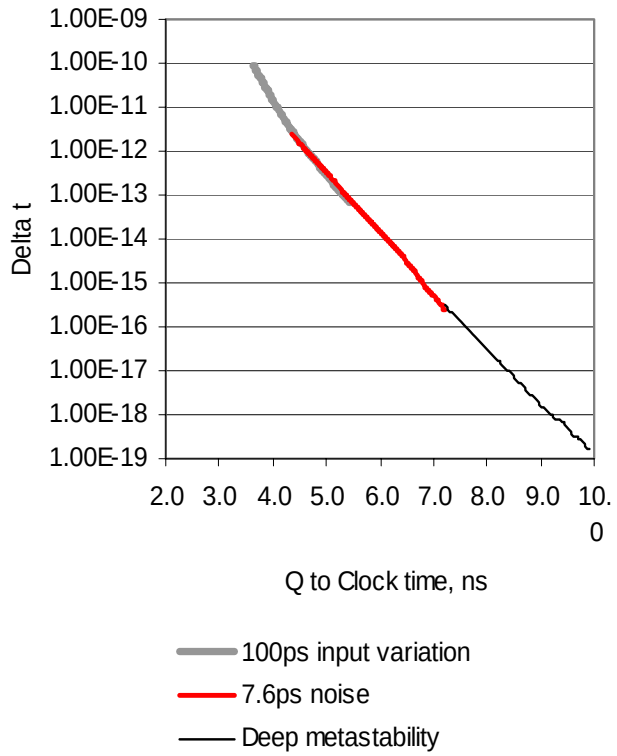
Deep metastability

- Minimum deviation is 7.6ps
- 100/7.6 = 13 times as many events with small input times (weeks not days)
- They occur every 100ns, too fast for the scope
- Only 1 in 1000 captured
- Most events still produce early output times
- Filter them out so that the event rate is much slower
- Results years not weeks



Results of all methods





74F5074 Schottky bipolar

74ACT74 CMOS

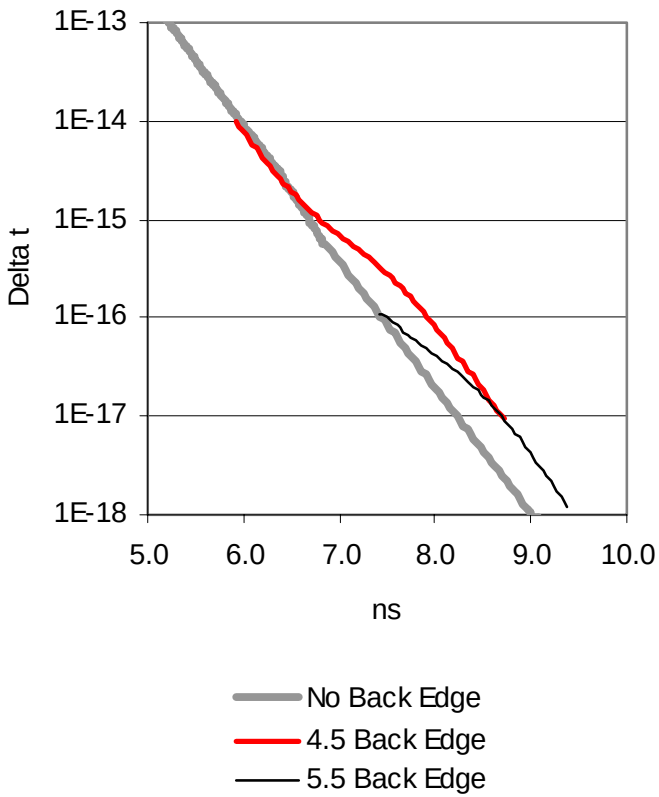
- Reliability measurements to 10^{-20} seconds (MTBF ~ 11days)
- Done in 3 minutes

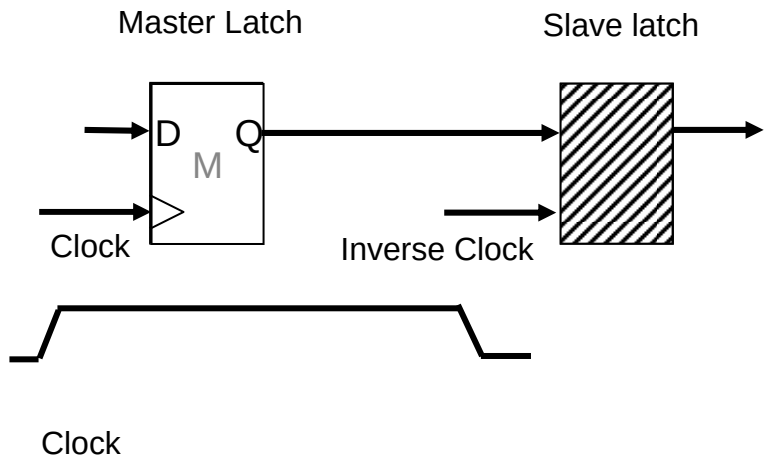
Results

$$MTBF = \frac{1}{\Delta_t f_c f_d} = \frac{1}{10^{-20}.10^7.10^7} = 11 days$$

- We can measure reliabilities of weeks not hours in a few minutes
- To get to 3 years reliability (10^{-22} seconds input overlap?) the experiment is run for 5 hours
  - picoseconds 10^{-12}, femtoseconds 10^{-15} , attoseconds 10^{-18} , zeptoseconds 10^{-21}, yoctoseconds 10^{-24}
- More than two slopes on one sample, 350ps, 120ps and 140ps
- We can see output events at up to 10 ns

When the clock goes low

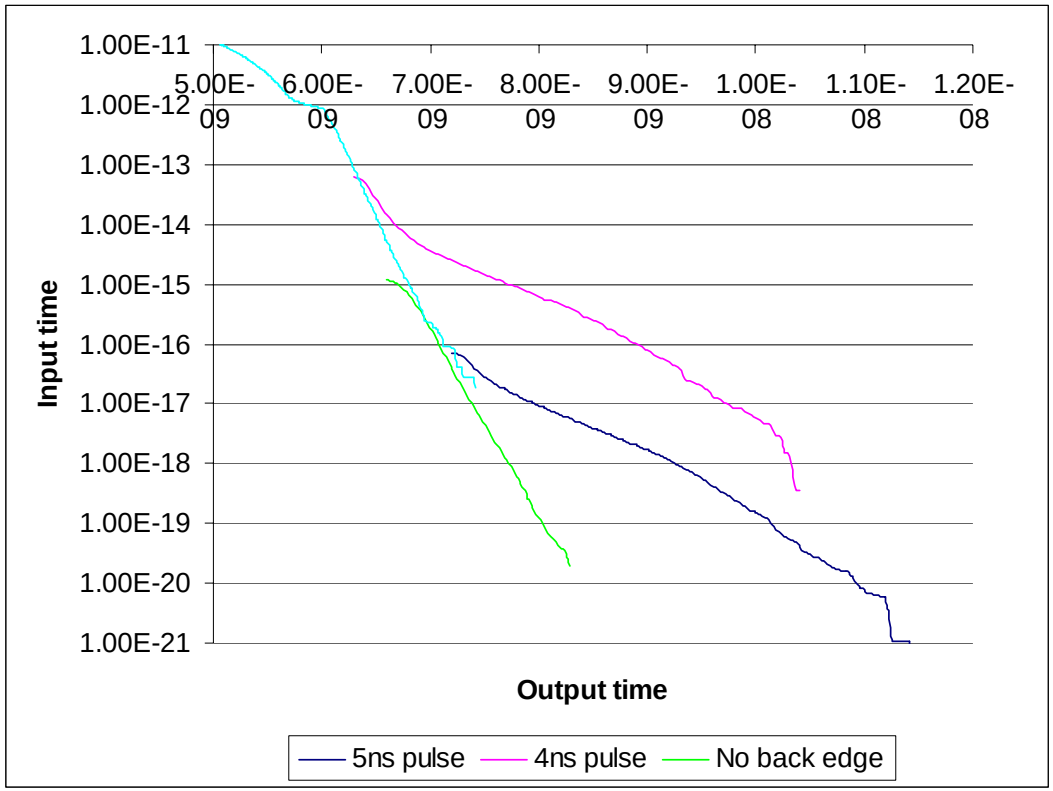


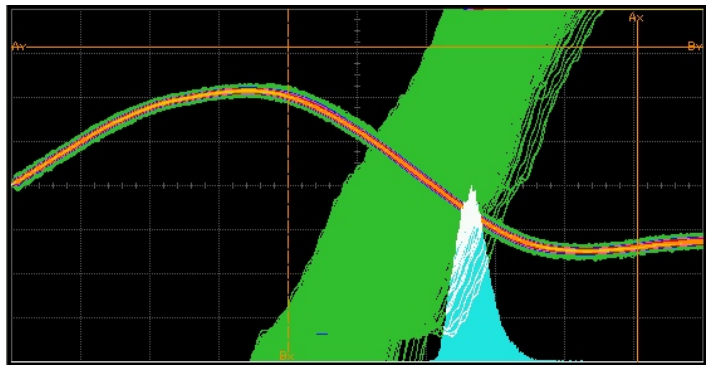


- Clock goes high, master goes metastable
- Master output arrives at slave
  - Before slave clock high: transparent gate delay t_d
  - As slave clock goes high: metastable, slightly longer delay

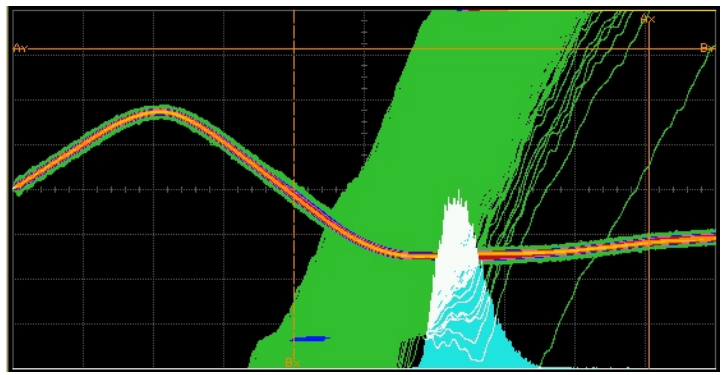
Back edge of clock causes increased delay

Effect of clock low on 74F5074





6 ns pulse

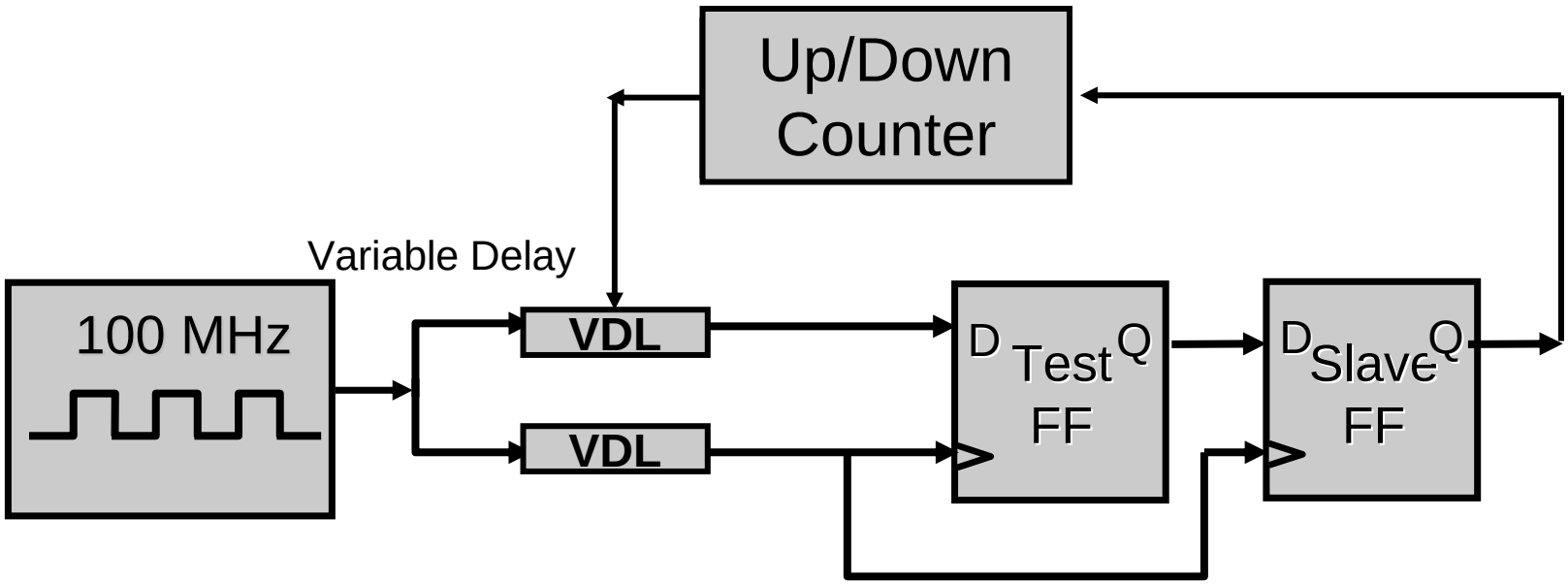


4 ns pulse

- 1 – 3 ns additional delay

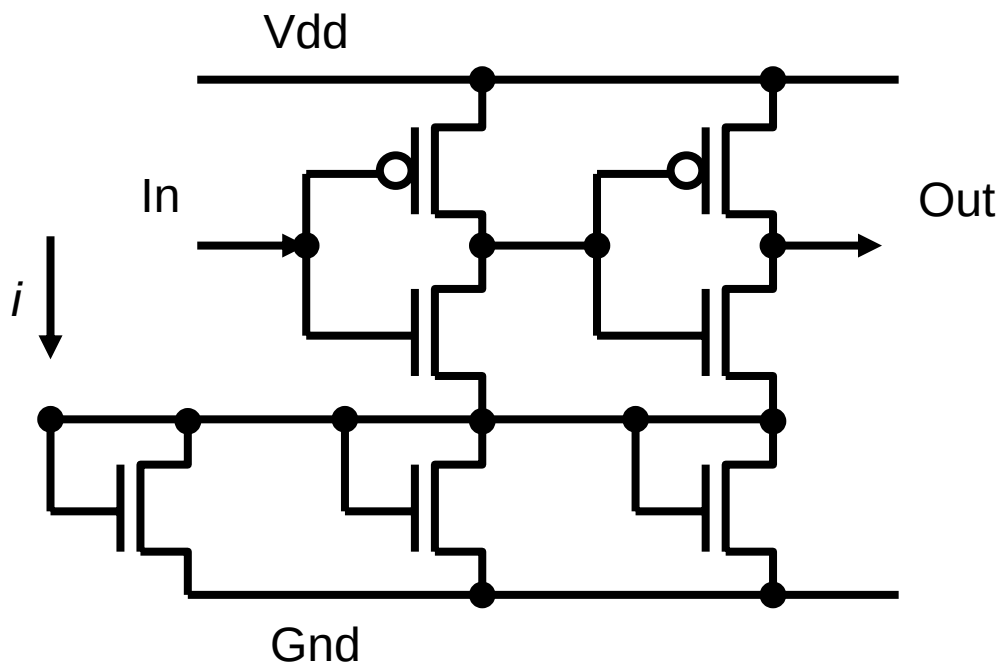
On-chip metastability measurement

- Analog delay replaced by digital delay (VDL)
- Analog integrator replaced by counter

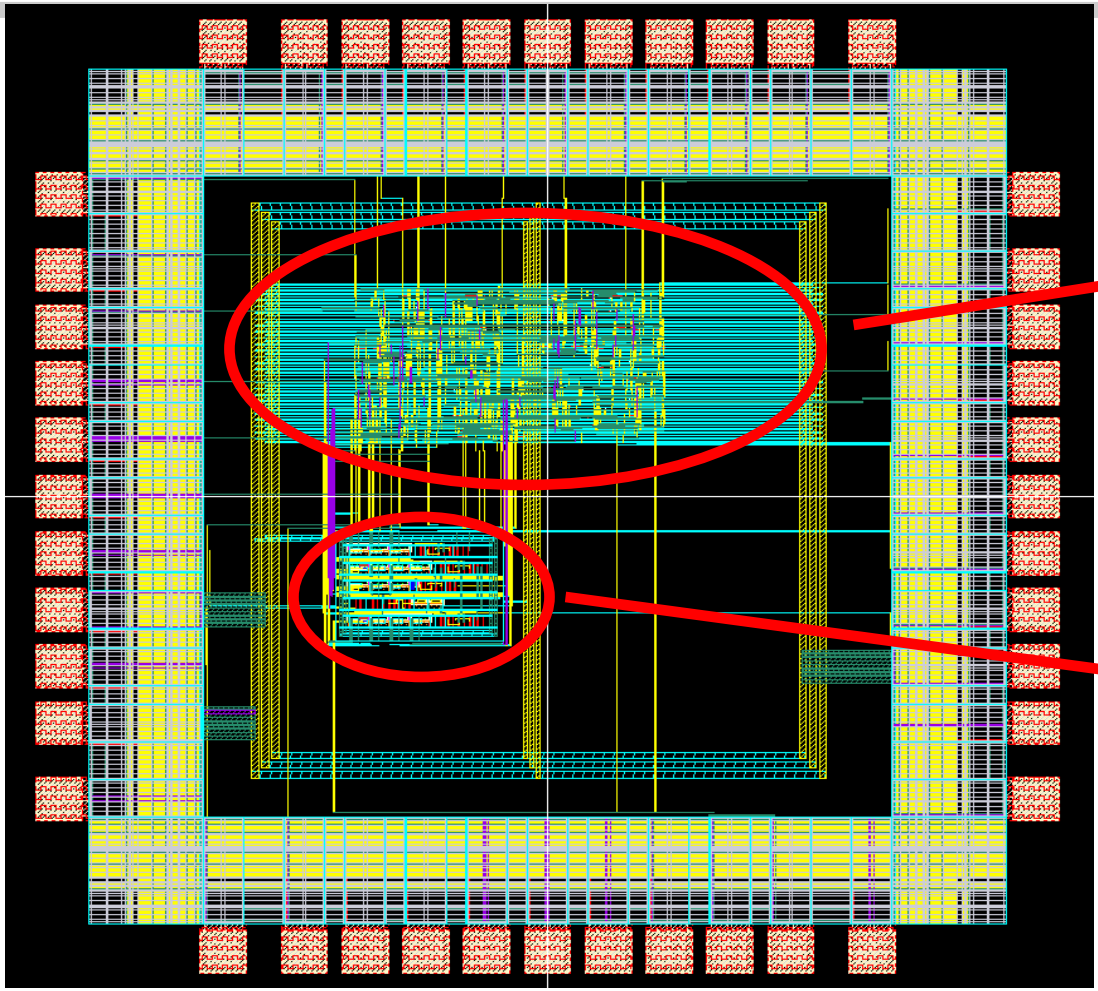


Variable delay stage

- Pair of current starved inverters
- Source current *i* variable in steps
- Delay changes can be as low as 0.1ps

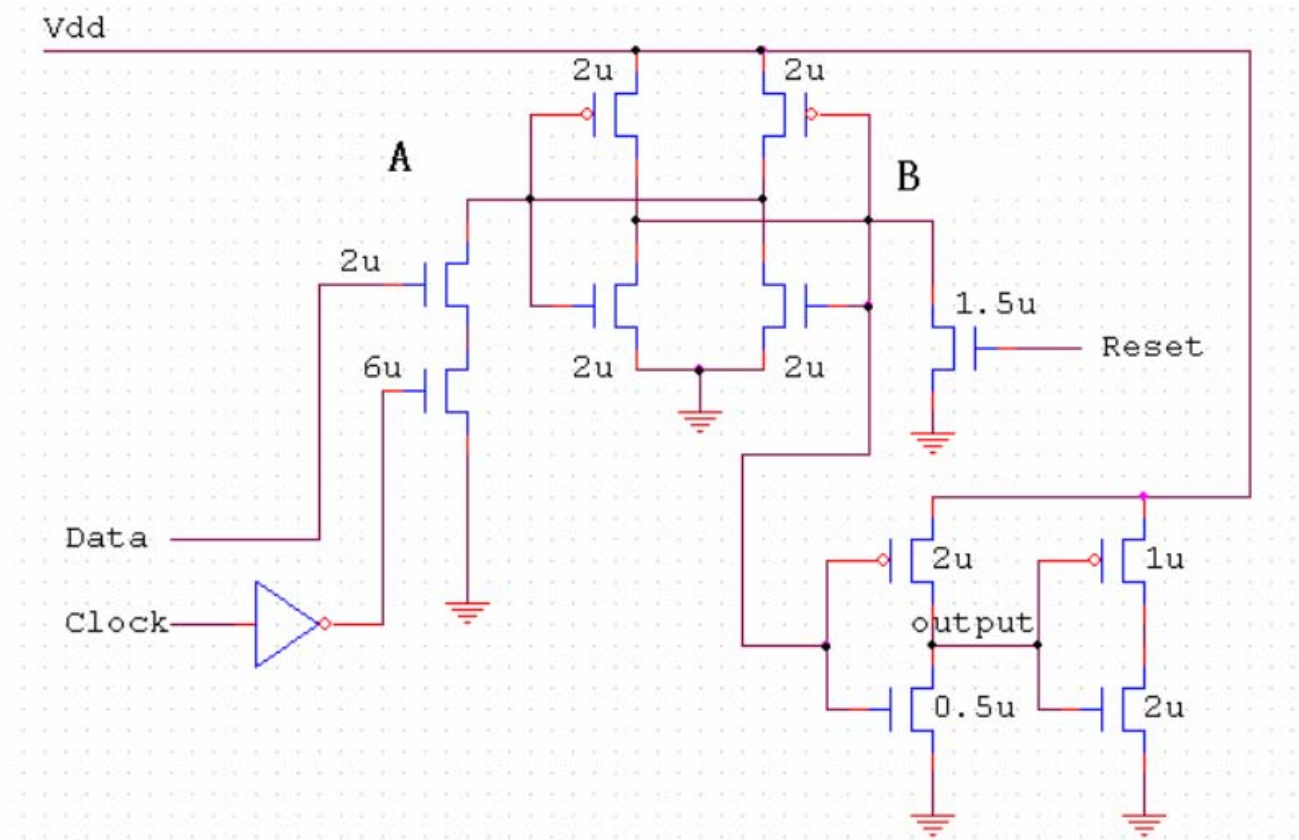


On-chip Implementation



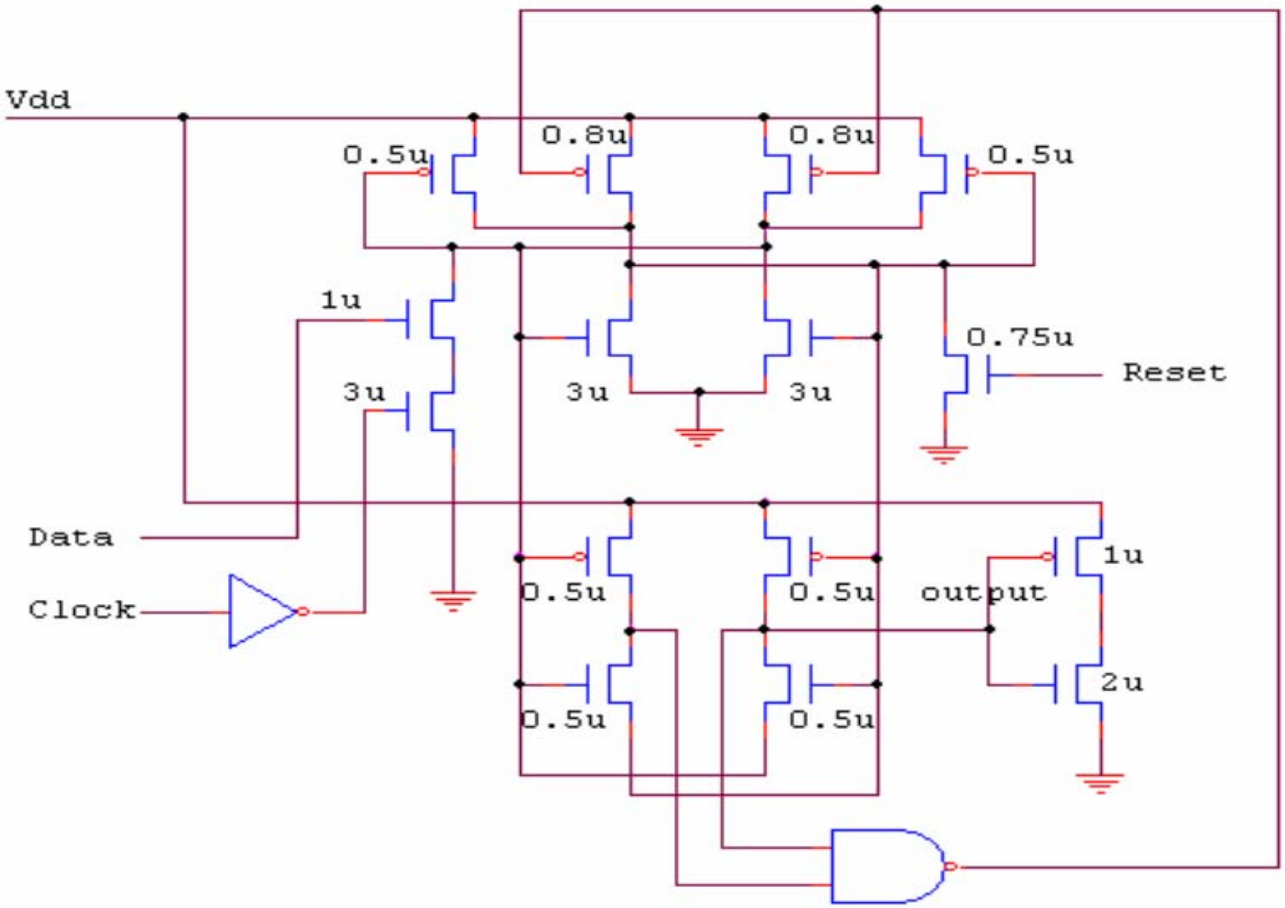
Layout of on-chip measurement circuit

Devices Under Test



Jamb Latch

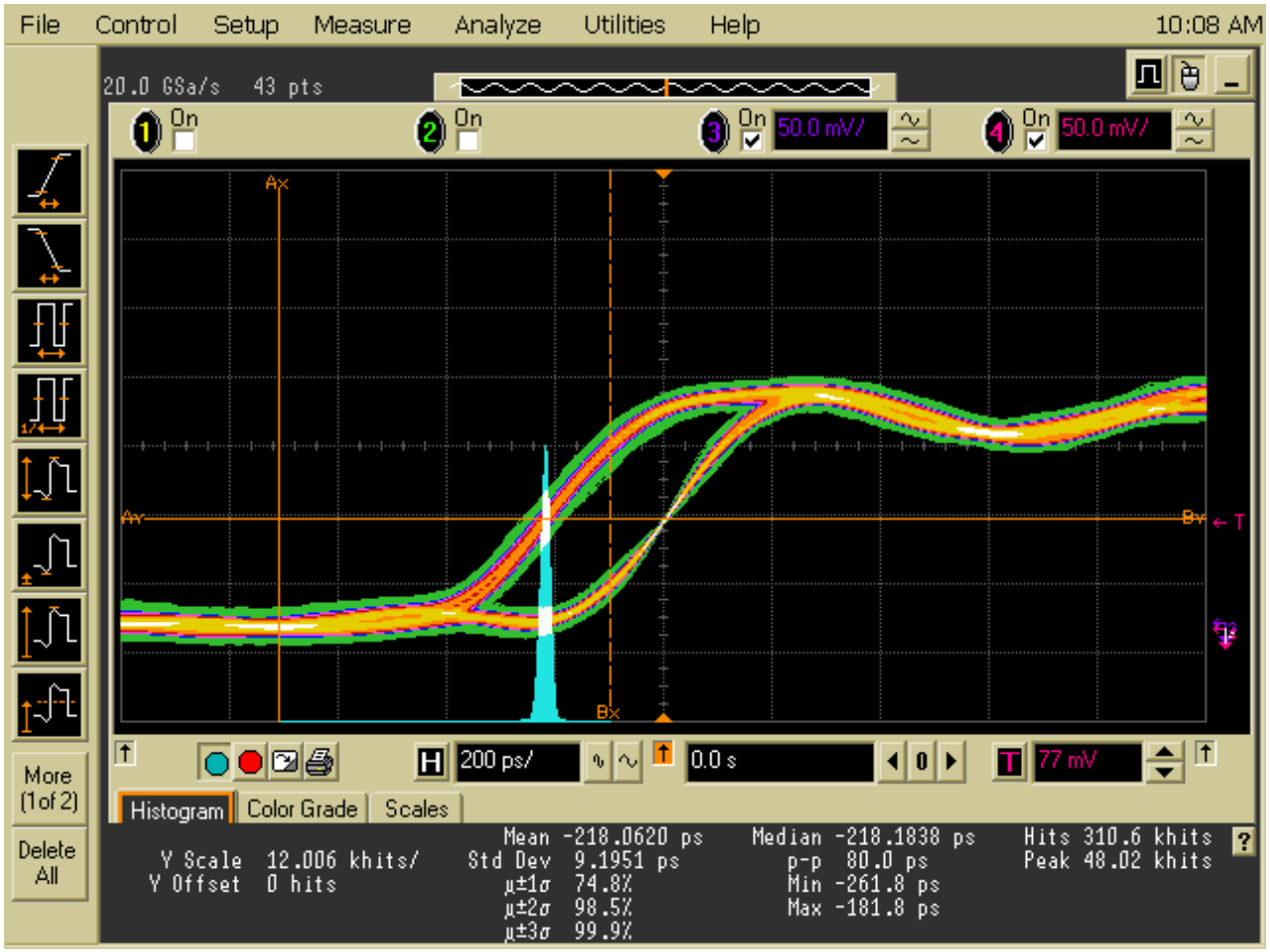
Devices Under Test



Robust Synchronizer

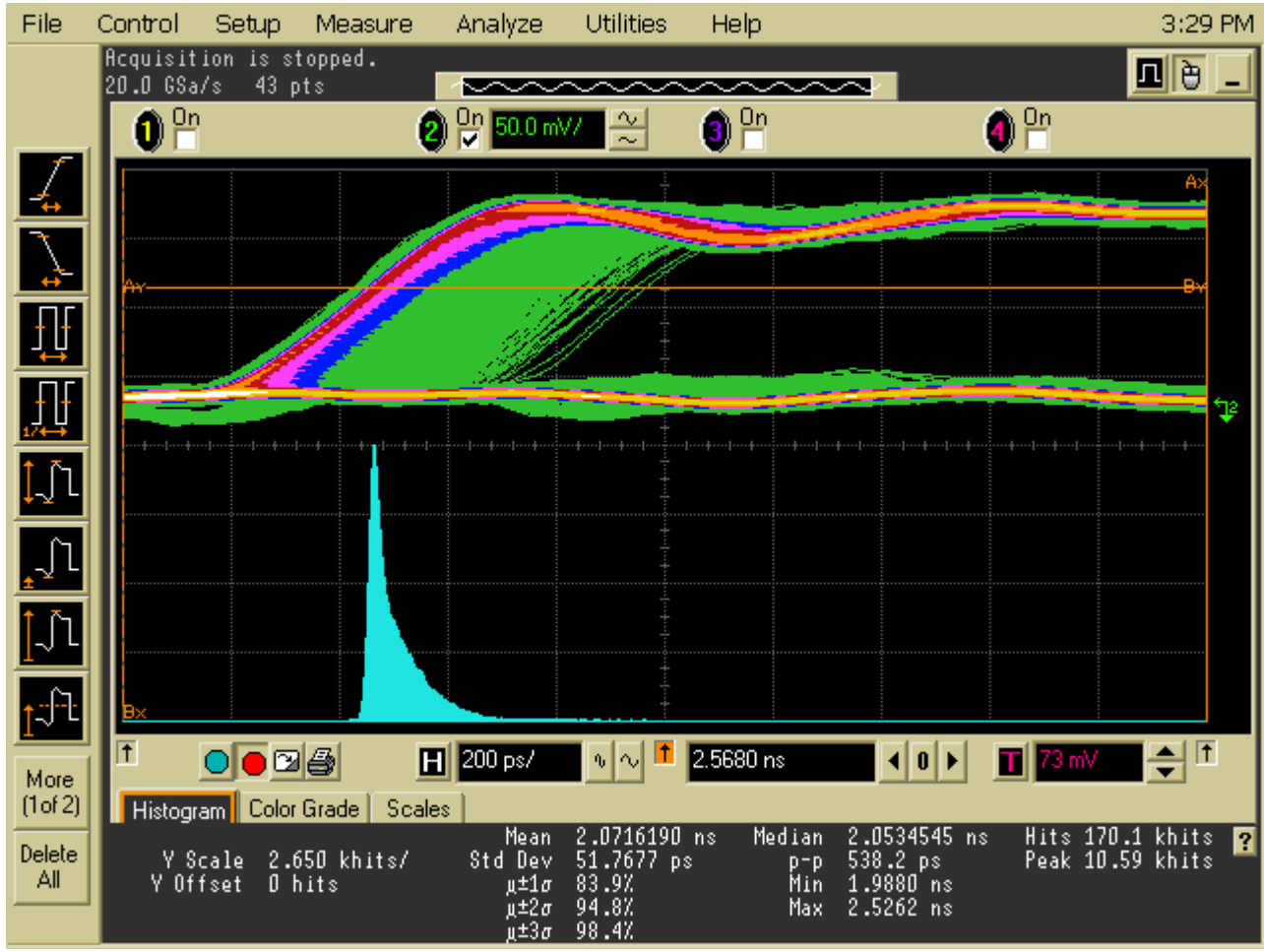
Input

- Deviation of clock 0ps at trigger
- Around 8-9ps elsewhere
- Data deviation around 9.2ps



Output

- τ around 30ps
- Does not rely on oscillators being independent



Results

	Measurement Results (ps)			
Vdd(v)	**Jamb Latch B**		**Robust Synchronizer**	
	$>10^{-14}$ s	$<10^{-14}$ s	$>10^{-14}$ s	$<10^{-14}$ s
1.8	19.44	35.55	15.27	34.92
1.7	21.75	37.29	16.53	35.76
1.6	25.64	40.93	19.38	38.25
1.5	28.77	52.36	20.29	43.07
1.4	36.22	66.17	23.75	50.36
1.35	45.43	75.35	28.51	58.19

τ (metastability time constant) vs Vdd

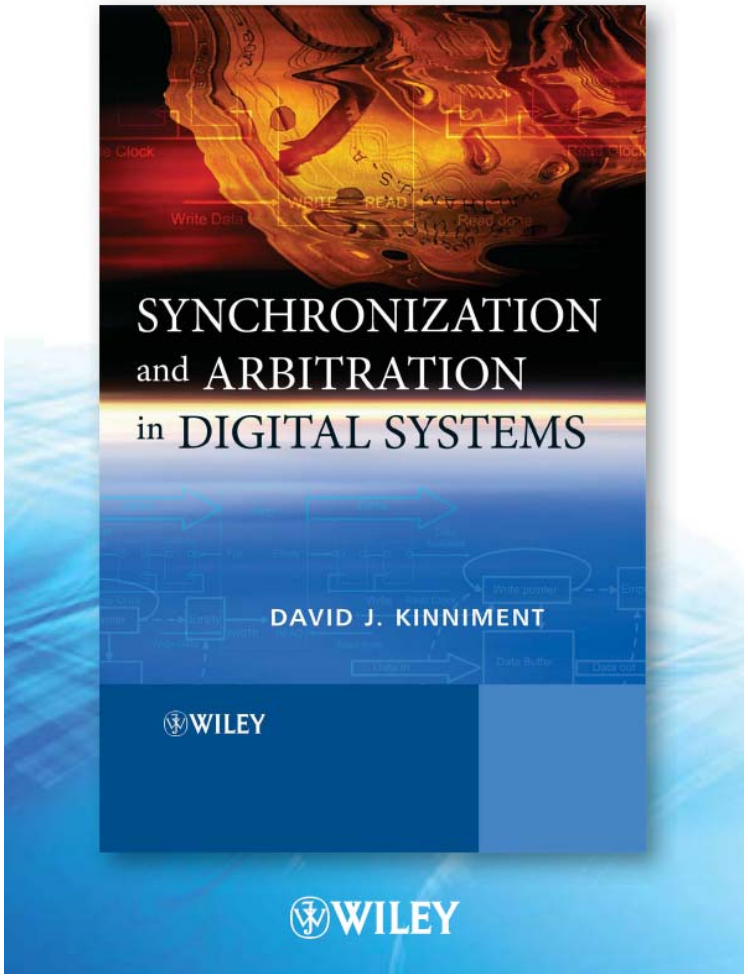
Measurement results

- Reliability measurements extended from
  - 10^{-15} s or MTBF = 16 min at 10MHz, to
  - 10^{-22} s or MTBF = 3 years
- We can see variations in τ not previously seen
- Measurement is statistical, not affected by noise
- Not affected by oscillator linking
- Back edge of clock pulse is seen to be an important effect, can be 0 – 15τ
- Demo by Jun Zhou

To learn more

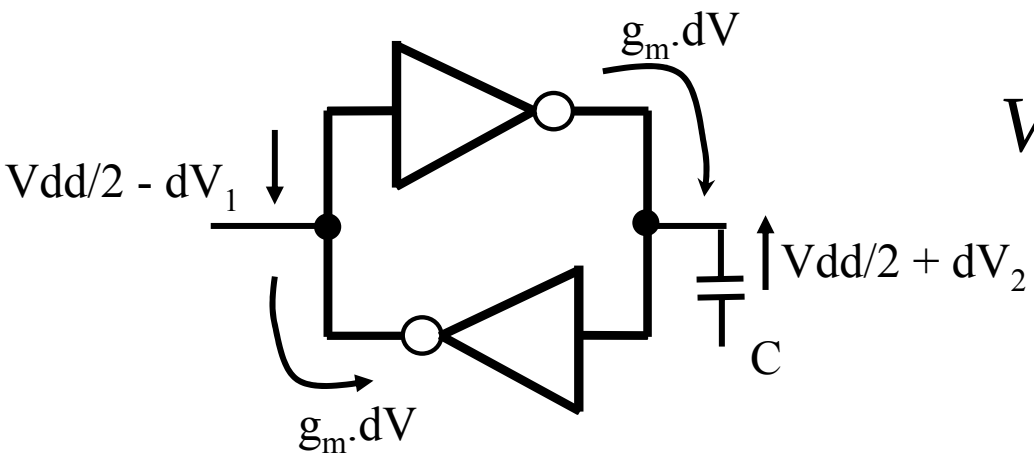
Bibliography:
http://www.iangclark.net/metastability.html

Book:
Synchronization and arbitration in digital systems
David J Kinniment Wiley 2007



Synchronizers and arbiters are part analog

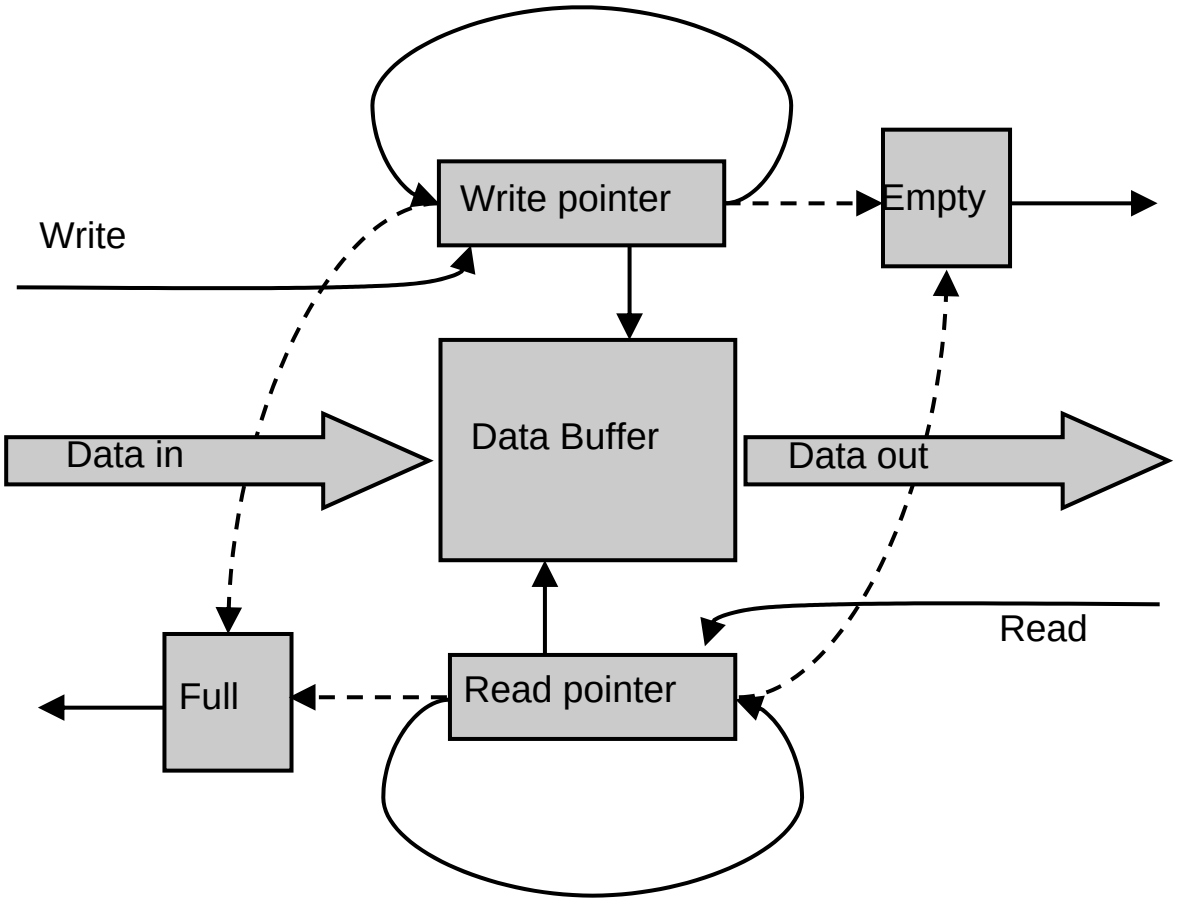
- Synchronizers depend on small signal parameters
- Synchronization time constant τ
  - 1/gain bandwidth product = $\tau = C/g_m$
  - $dV_2/dt = dV_1 * g_m/C$



$$V_1 \sim K.e^{\frac{t}{\tau}}$$

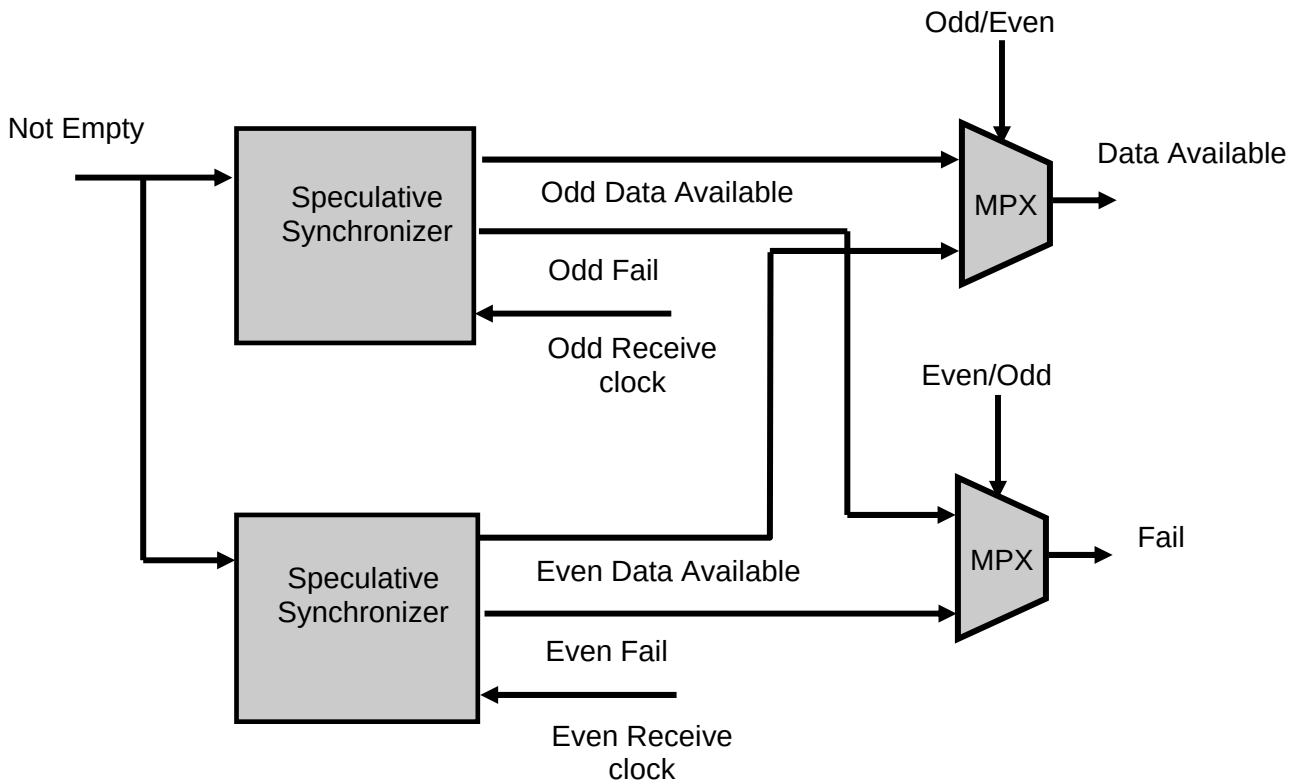
FIFO control

- A write advances the Write pointer
- A read advances the read pointer
- The write pointer is kept *n* accesses ahead of the read pointer, other wise empty is indicated
- There must be *m* locations free behind the read pointer, otherwise full is indicated



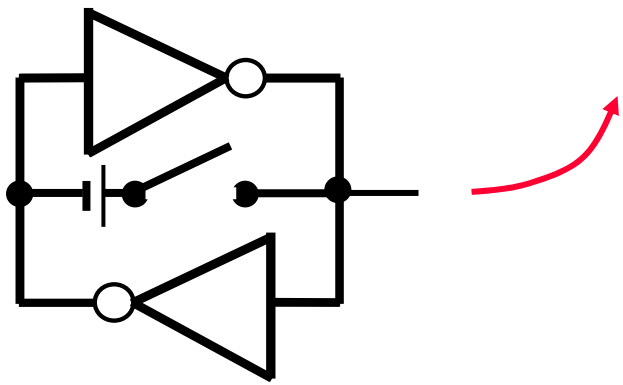
Overlapping two synchronizers

- Full cycle needed before fail status is known
- BUT
- Synchronizers can be overlapped to maintain throughput



Measuring τ by Simulation

- Short FF nodes together with small offset voltage, then open switch and measure time constant
- Fairly accurate for long term τ
- Not practical on some library devices, FPGAs

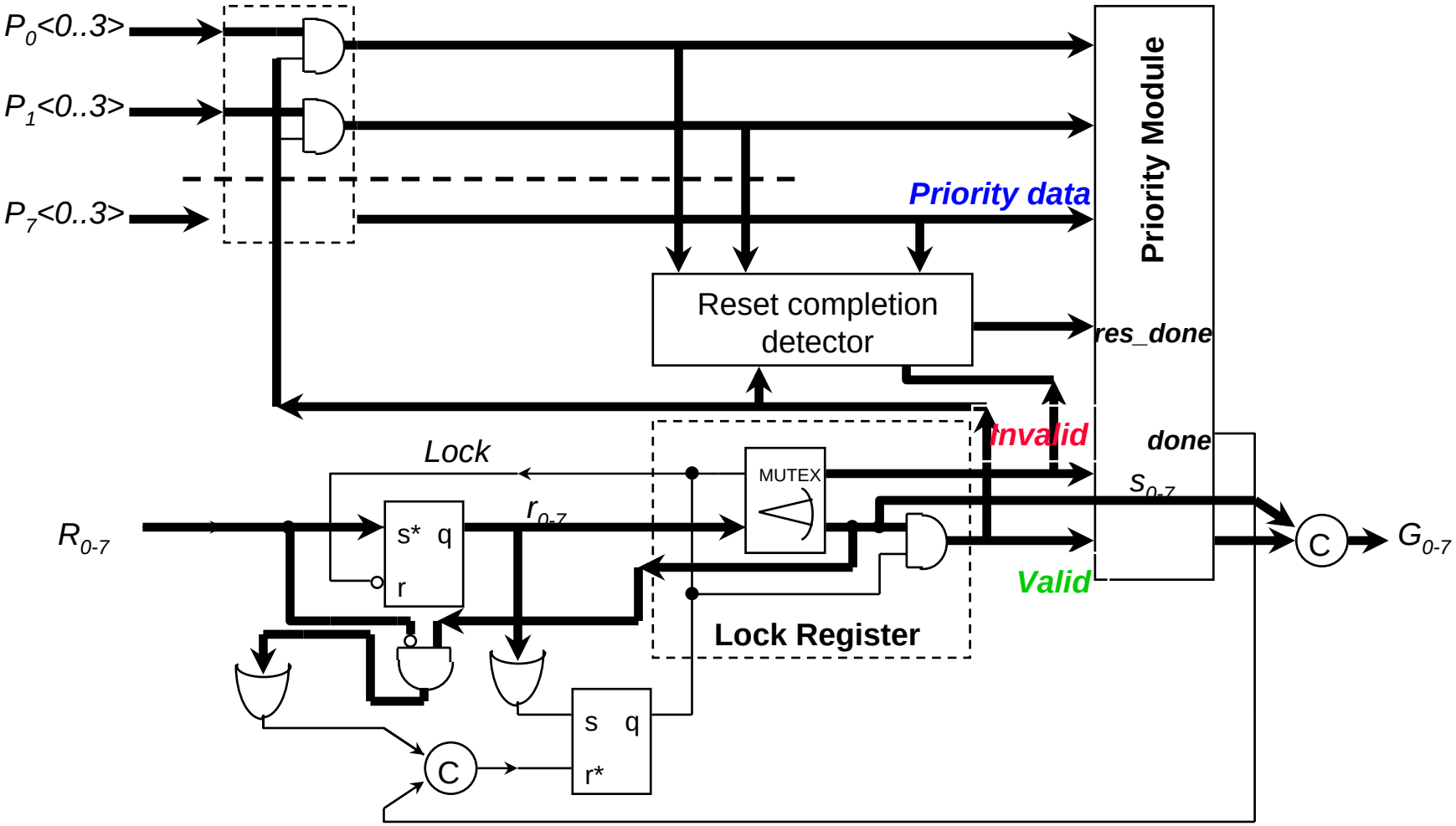


Quasi speed independent

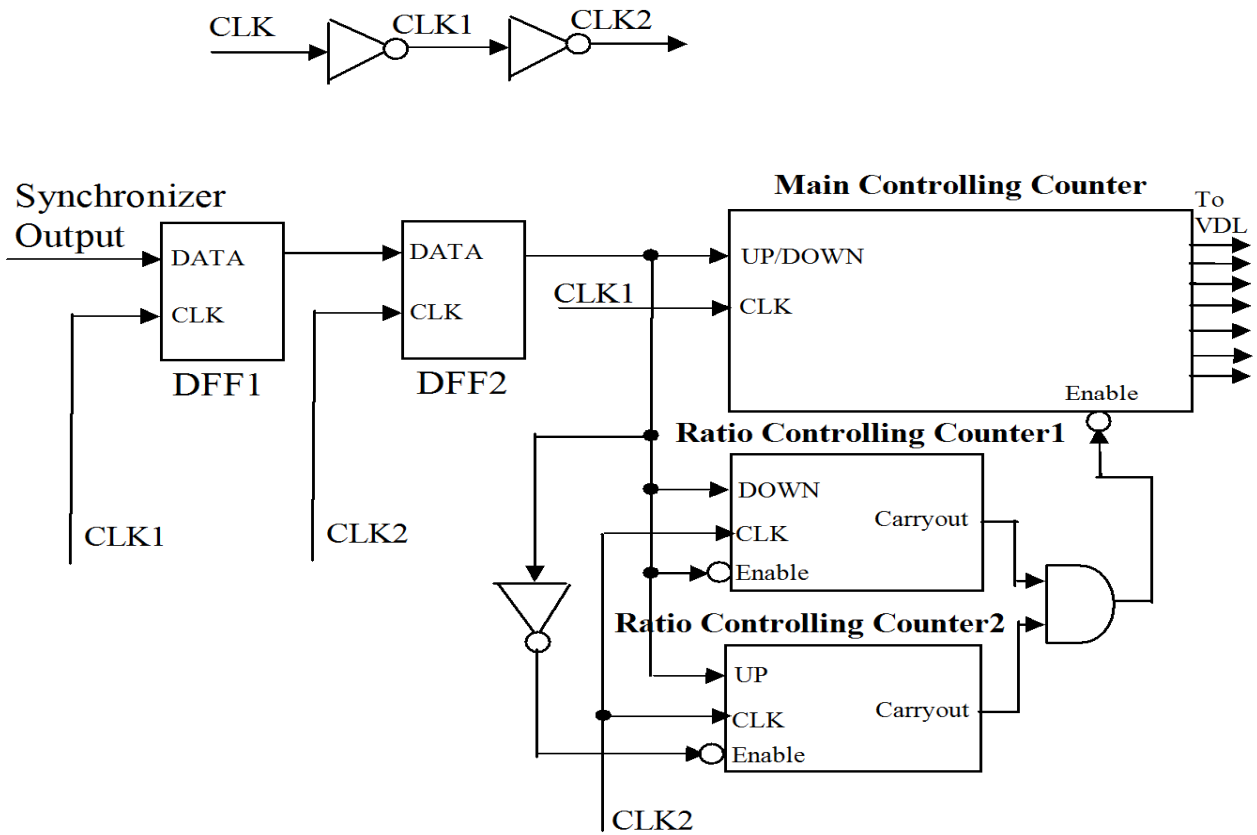
Assumptions

- *s*+ must occur before *Lock*+
  - The physics of the MUTEX are such that if *r*+ is before *Lock*+, *s*+ must be asserted
- The three inputs to the Lock bistable are implemented as a single complex gate set.
  - A faster non speed independent implementation in which the gate is separate is possible

Dynamic priority

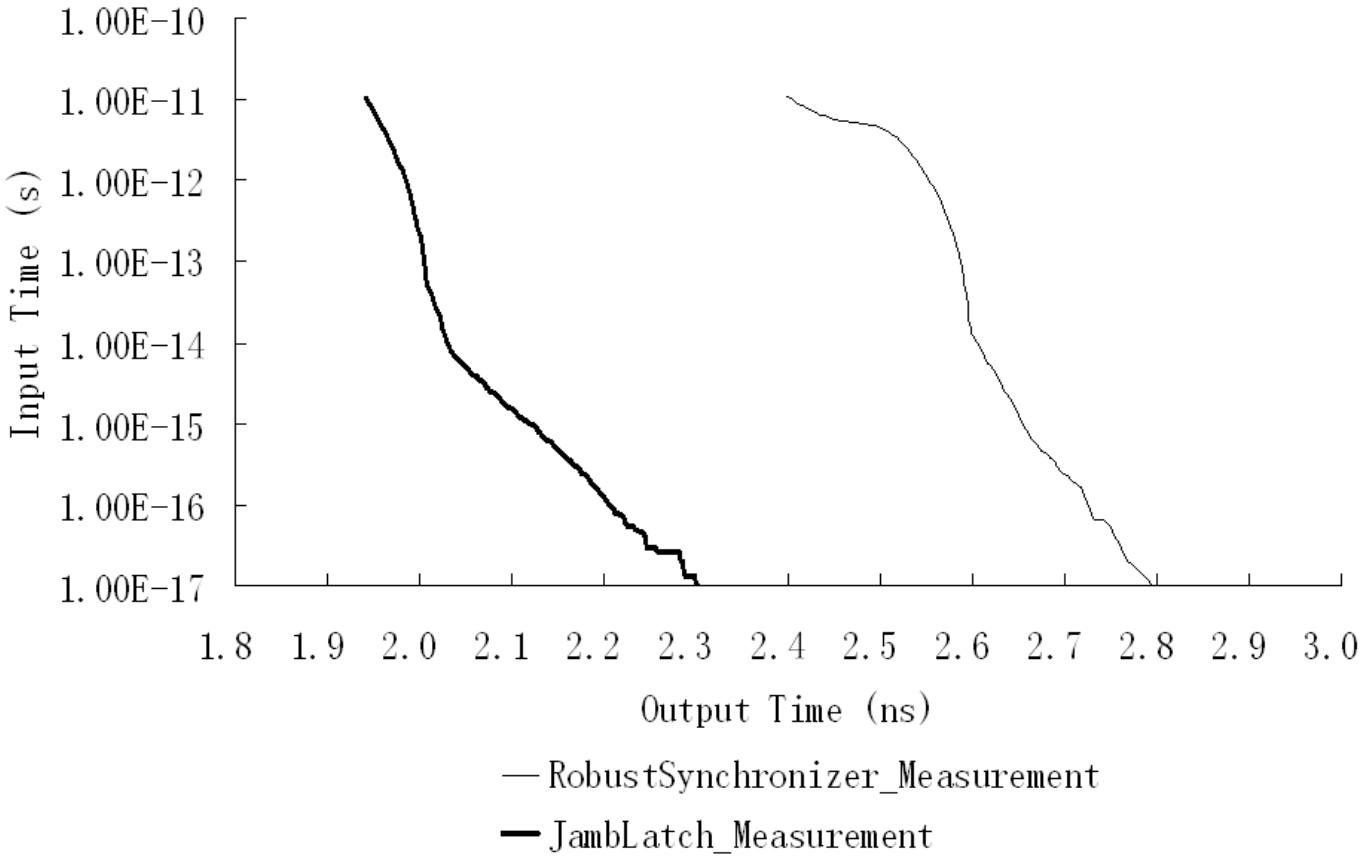


On-chip Implementation



Controlling Counters

Results



Input Time vs Output Time