

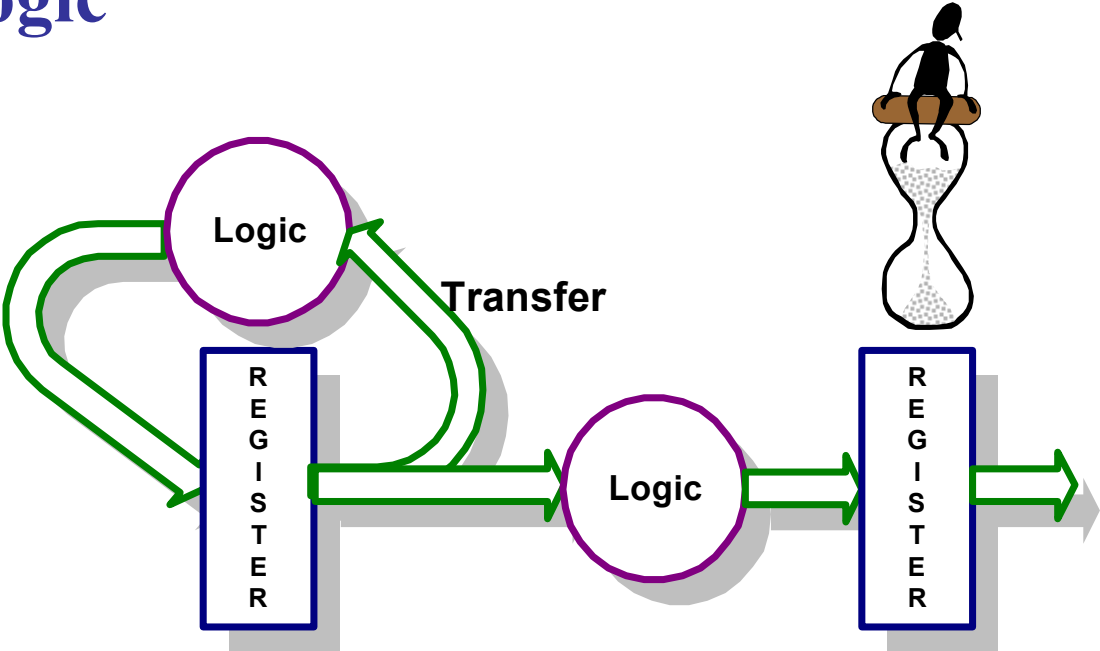
Clock Domain Crossing



Register

Transfer

Logic



RTL (cont)

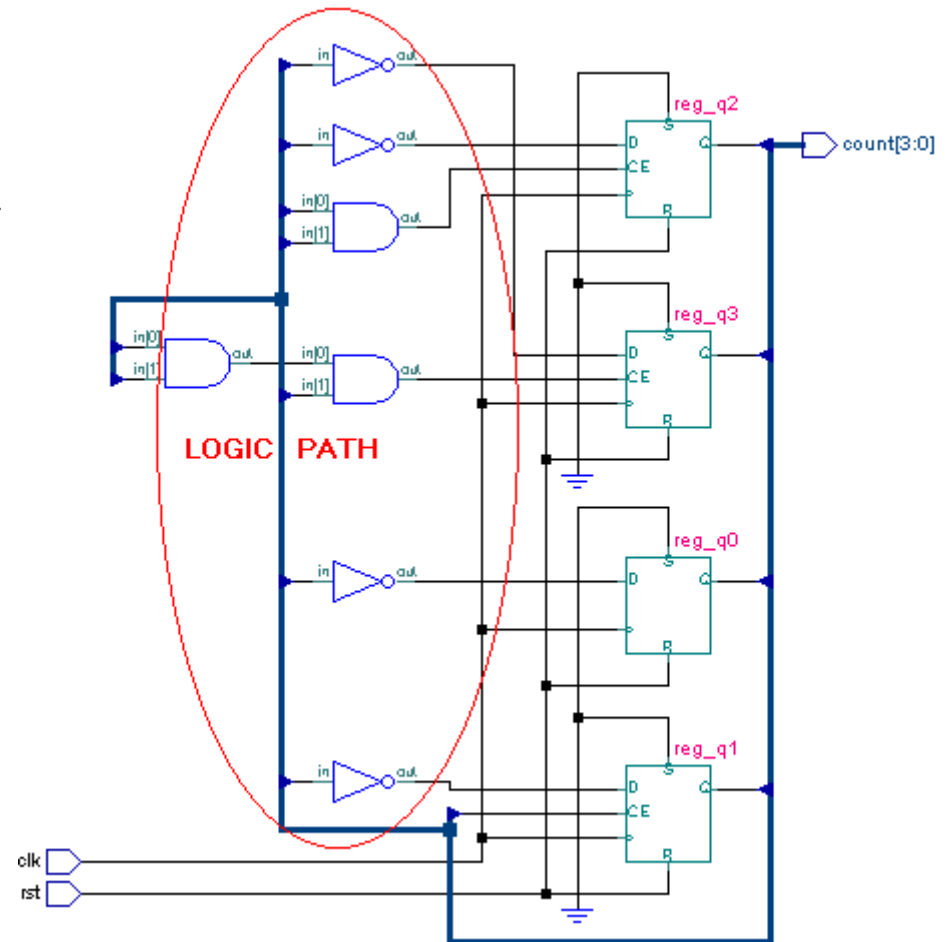
An RTL circuit is a digital circuit composed of:

- (R) registers of FFs responding to the edge of a clock signal
- (T) Transfer of wires are the connecting elements
- (L) Combinatorial Logic gates.

RTL (cont)

A **logic path** in RTL circuit is a set of transfer and logic elements connected to each other.

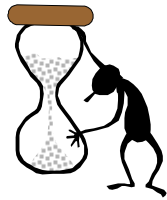
A **logic path** starts at the registers Q output and ends at the registers D input.



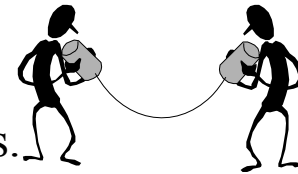
RTL (cont)

Clock Domain in a RTL circuit such that:

Registers =>R The clock signal is the same clock signal as for all other Registers.



Transfers =>T is part of a Logic path that starts and ends at Registers.

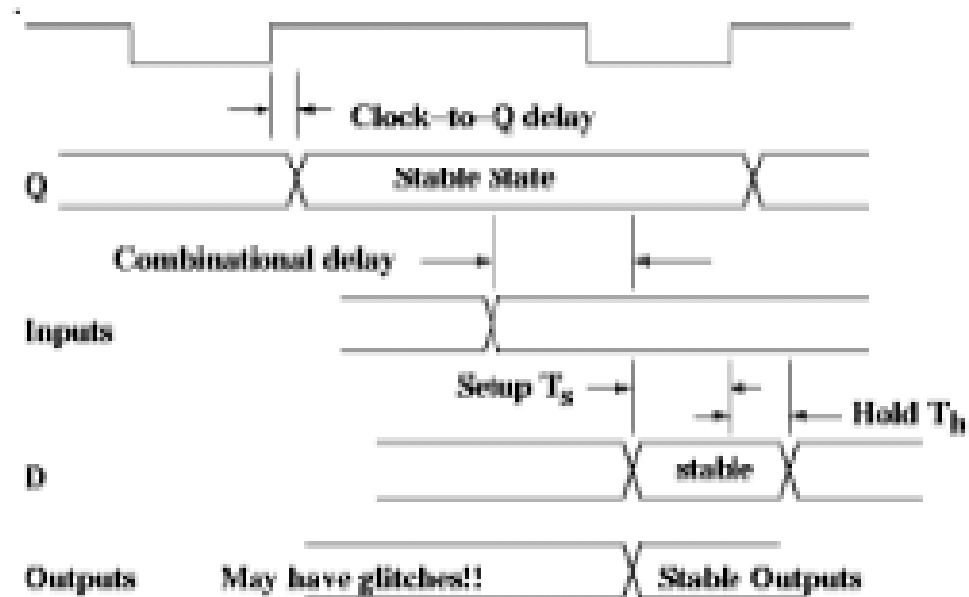
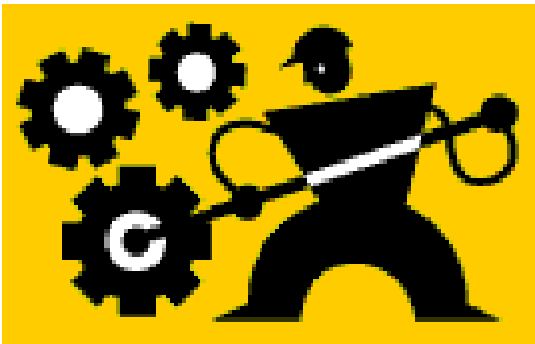


Logics => L is agate in a Logic path that starts and ends at RRs.



RTL (cont)

Clock Speed is limited by the flip-flop delay (clock to output), combinational delay, and setup time.

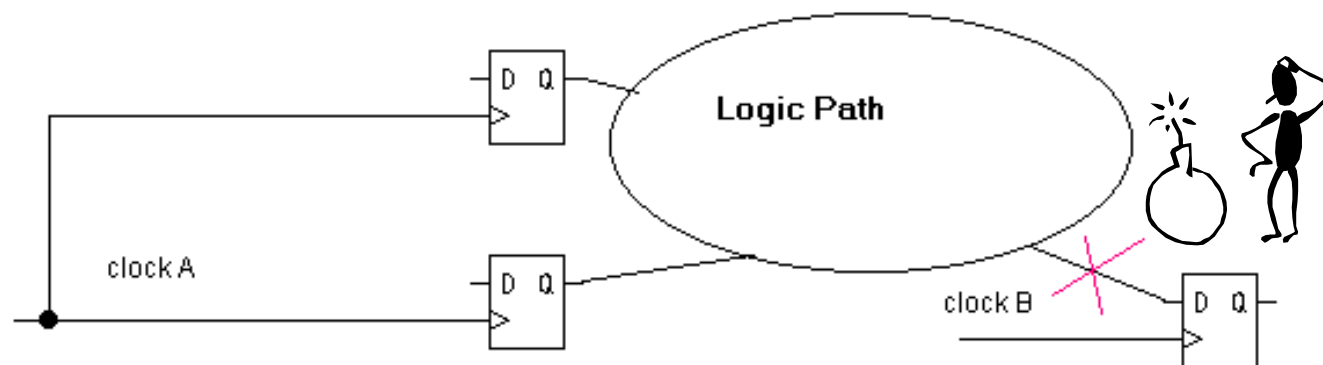


Bridge path

Bridge path : A bridge path in an RTL circuit is a logic path that starts at registers' **Q** outputs of one clock domain and ends at registers' **D** inputs of the second clock domain.

Bridge Path rule:

Bridge path *must be sampled* with a register clocked by the target domain's clock.



Clock Skew

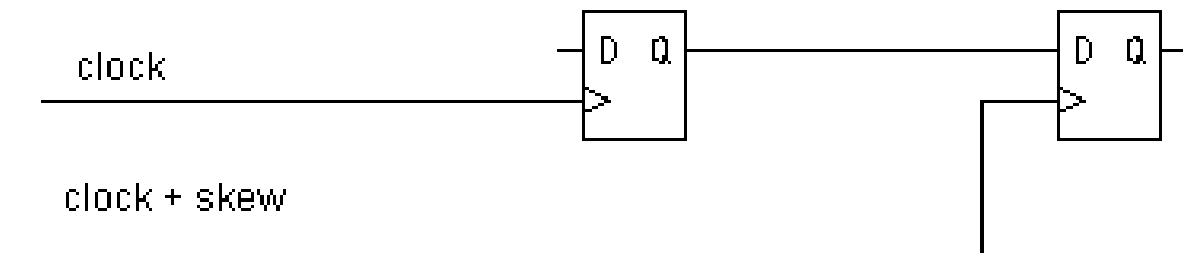
Clock Skew : is the *difference* measured in time between the clock edge (rising or falling edge) of two FFs.

Data Delay : is a *difference* measured in time between the beginning and end of a logic path.

Clock domain Rule :

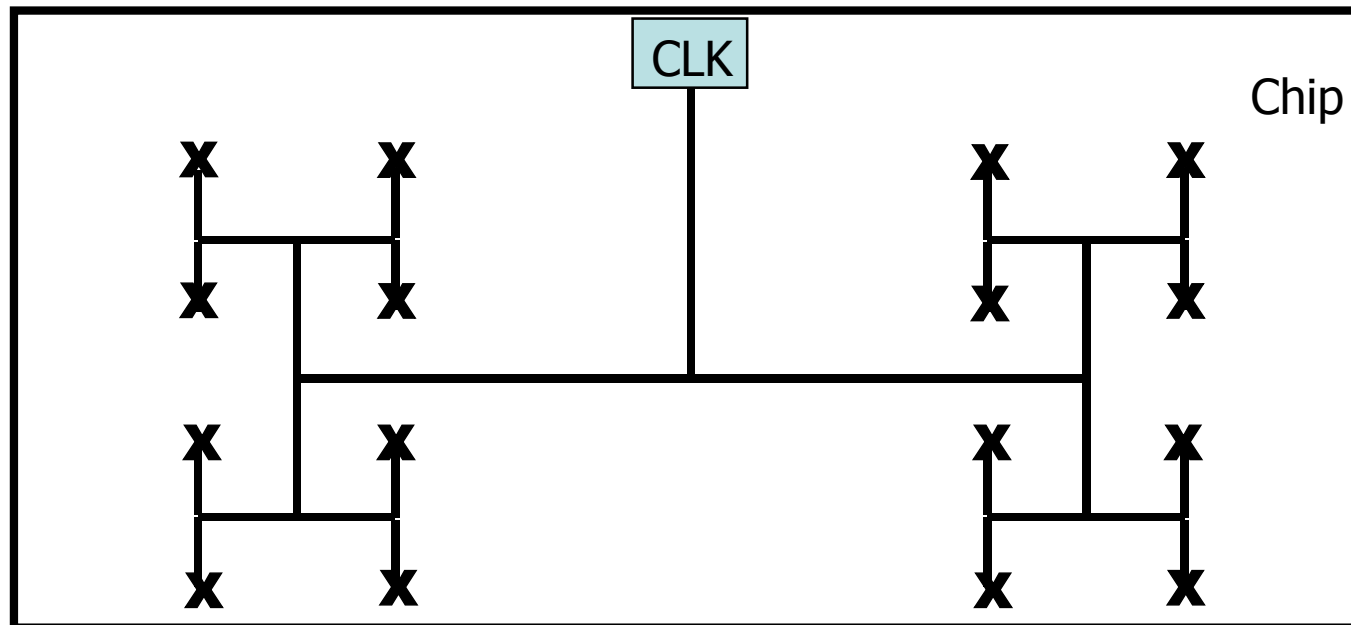
CLOCK SKEW \ll DATA DELAY

Critical in shift registers, counters and combinatorial systems.



Clock Tree Example – “H-Tree”

- Most of VLSI circuit signal propagation delay is caused by the wiring
 - Same distance from clock source to all X's

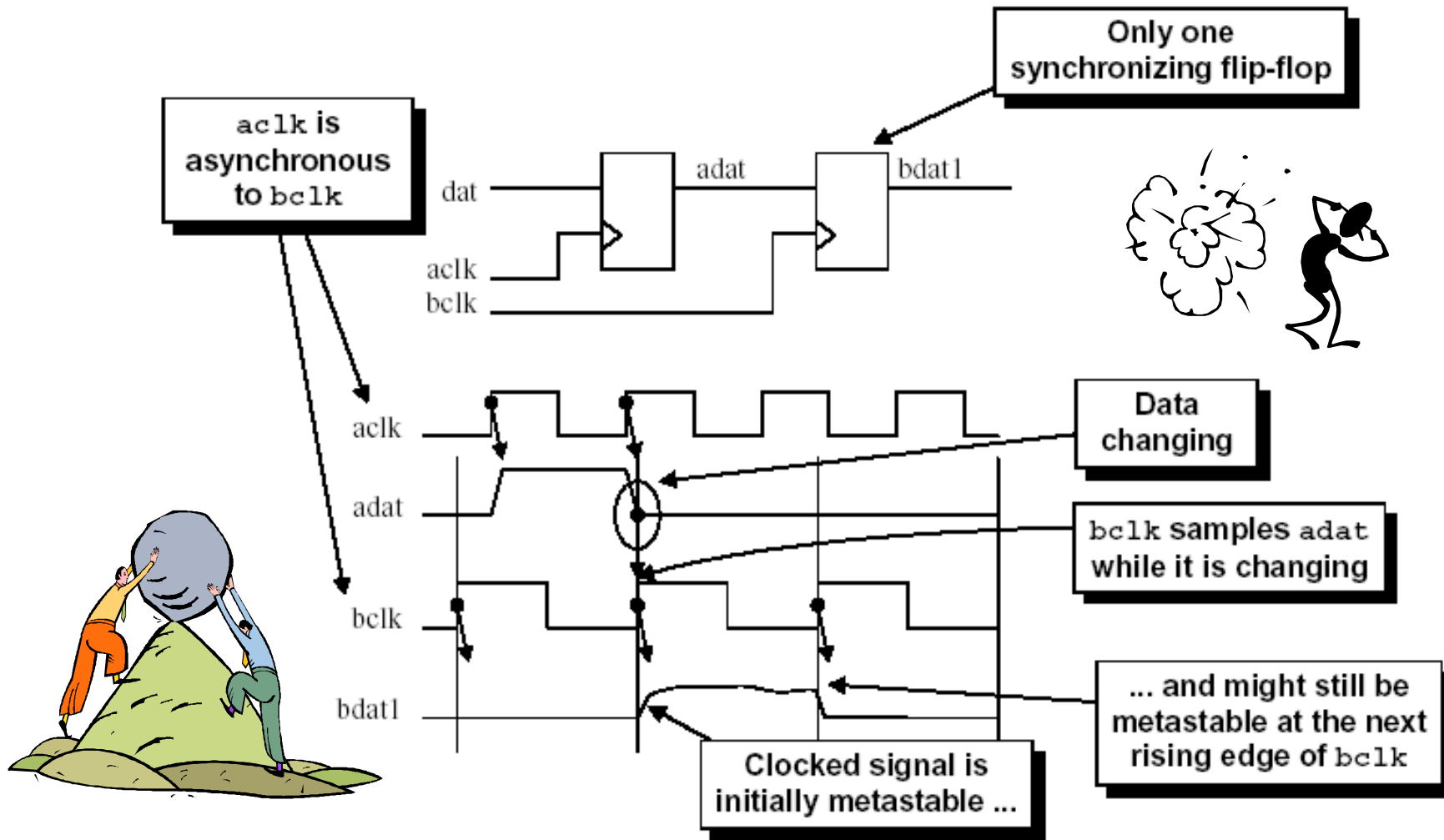


Meta-Stability

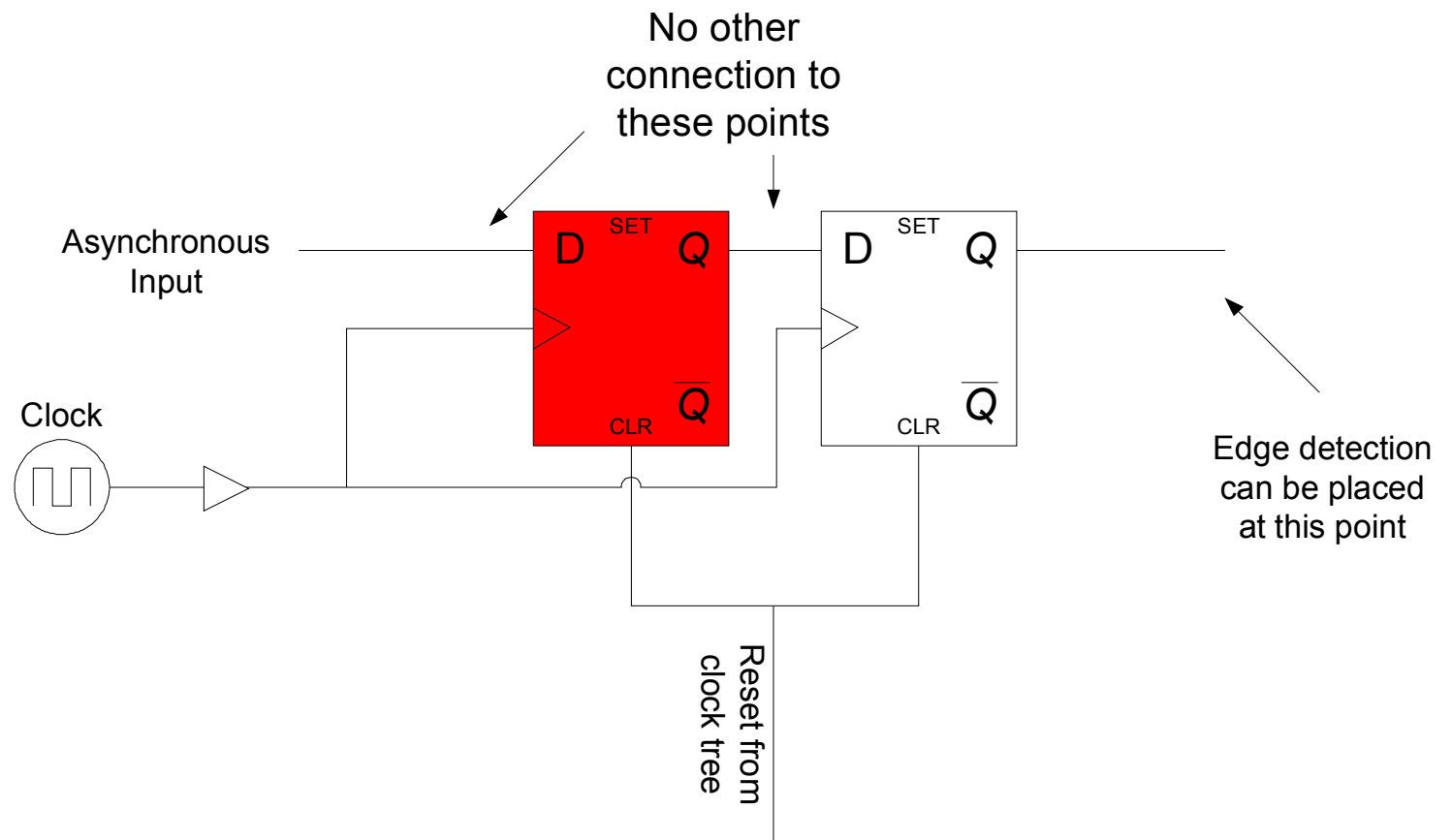
What are the cases in which meta-stability occurs?

- When the input signal is an **asynchronous** signal.
- When the clock **skew/slew** is too much (rise and fall time are more than the tolerable values).
- When interfacing **two domains** operating at **two different** frequencies or at the same frequency but with different phase.
- When the **combinational delay** is such that flip-flop data input changes in the critical window (setup + hold window)
- Digital components recover from meta-stable states quickly but the end value is indeterminate

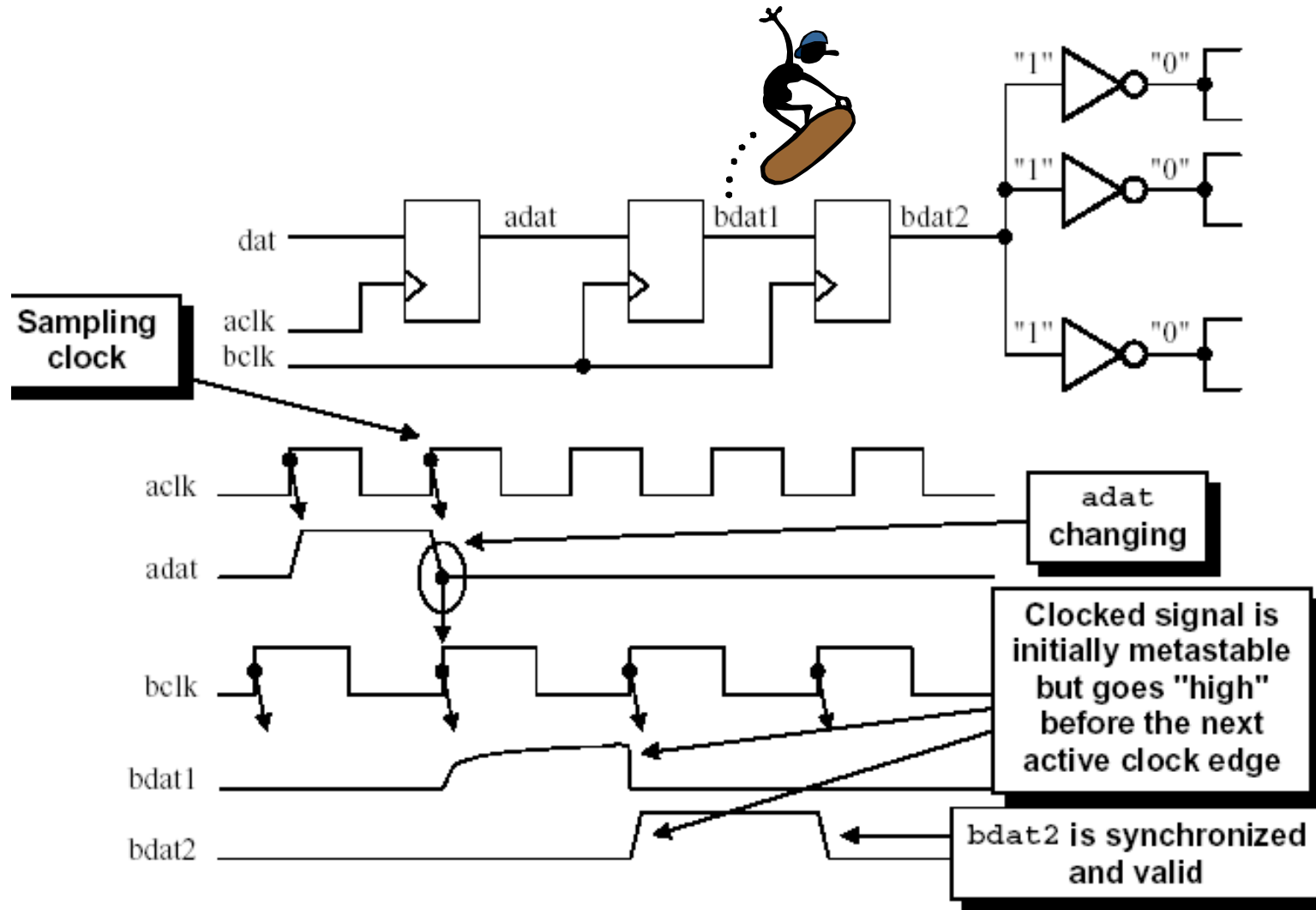
Meta-Stability



Meta-Stability



Meta-Stability



Meta-Stability

MTBF – Mean Time Between Failures

$$MTBF = \frac{e^{C_2 t_{MET}}}{f_{clk} \cdot f_{dat} \cdot C_1}$$

f_{clk} = Clock frequency

f_{dat} = The average frequency of asynchronous data changes

t_{MET} = Time allowed for the FF to settle in a stable state

C_1 = probability of meta-stability catching setup/hold time window

C_2 = Technology dependent. Describes the speed with which the meta-stable condition is being resolved

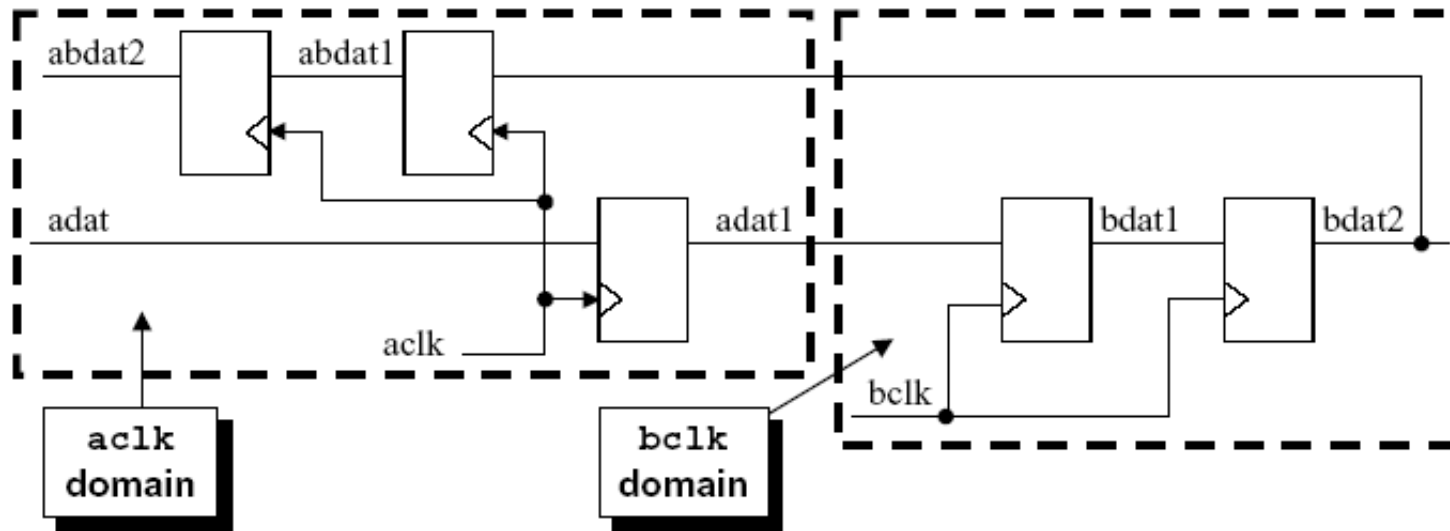
CDC failure

Difficulties causes when interfacing two or more asynchronous clock domains

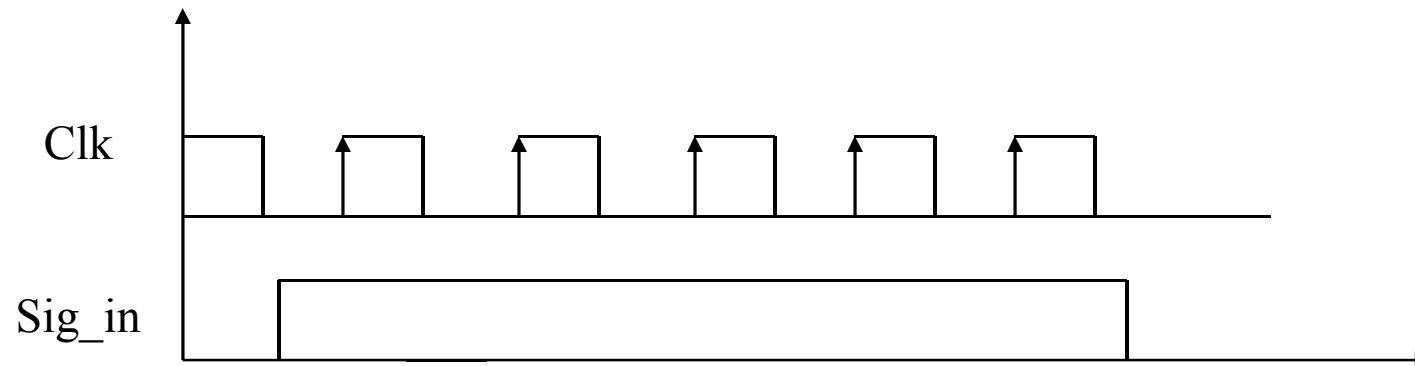
- Synchronization failure
- Un-reliable data transfers
 - ☠ Transaction never happen
 - ☠ Transaction Happen too many times
 - ☠ Transaction delivered wrong data

CDC Feedback

Although synchronizing a feedback signal is a very safe technique to acknowledge that the first control signal was recognized and sampled into the new clock domain, there is considerable delay associated with synchronizing control signals in both directions before releasing the control signal.



CDC - Edge detection

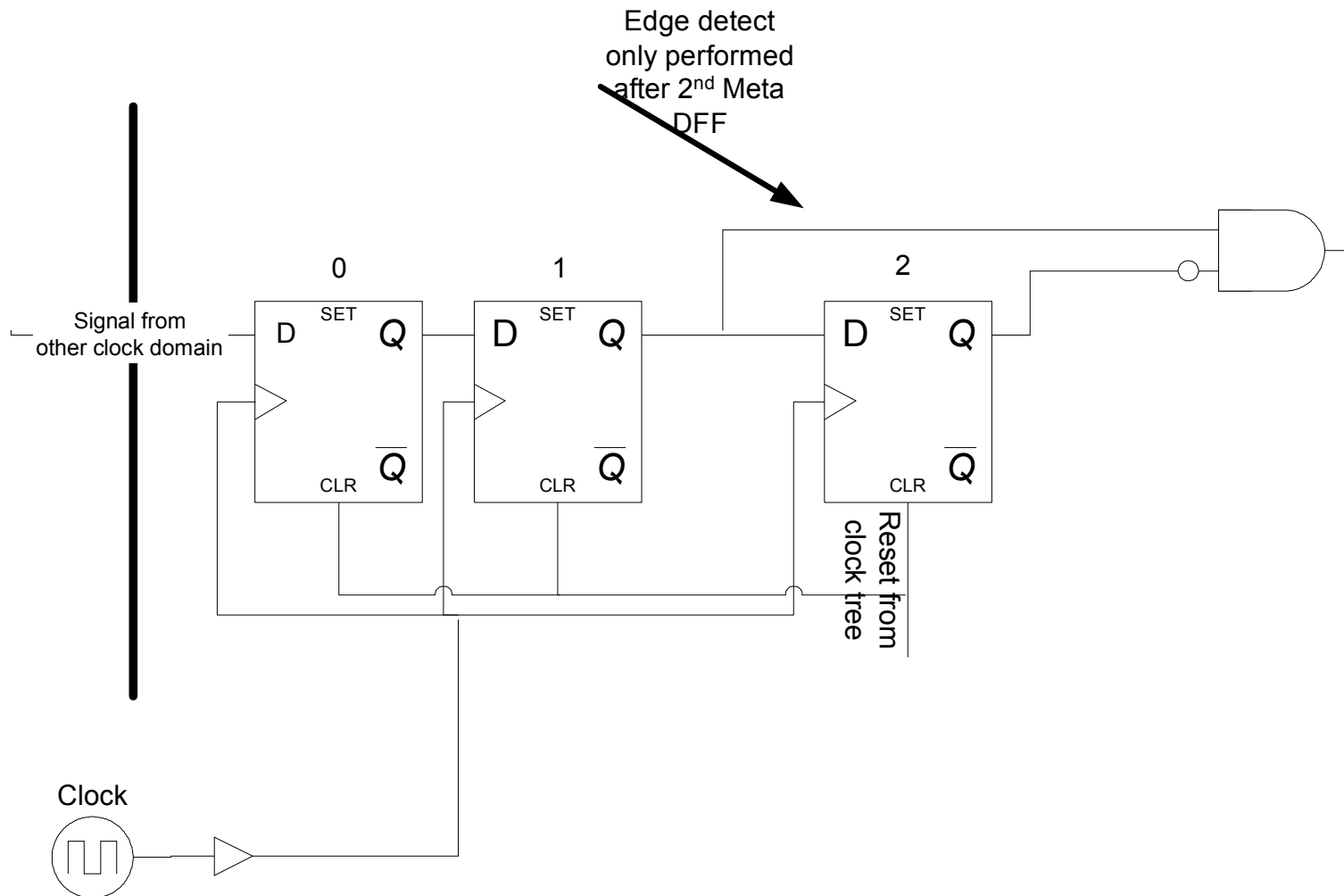


Sig_in events changes for a long time period.

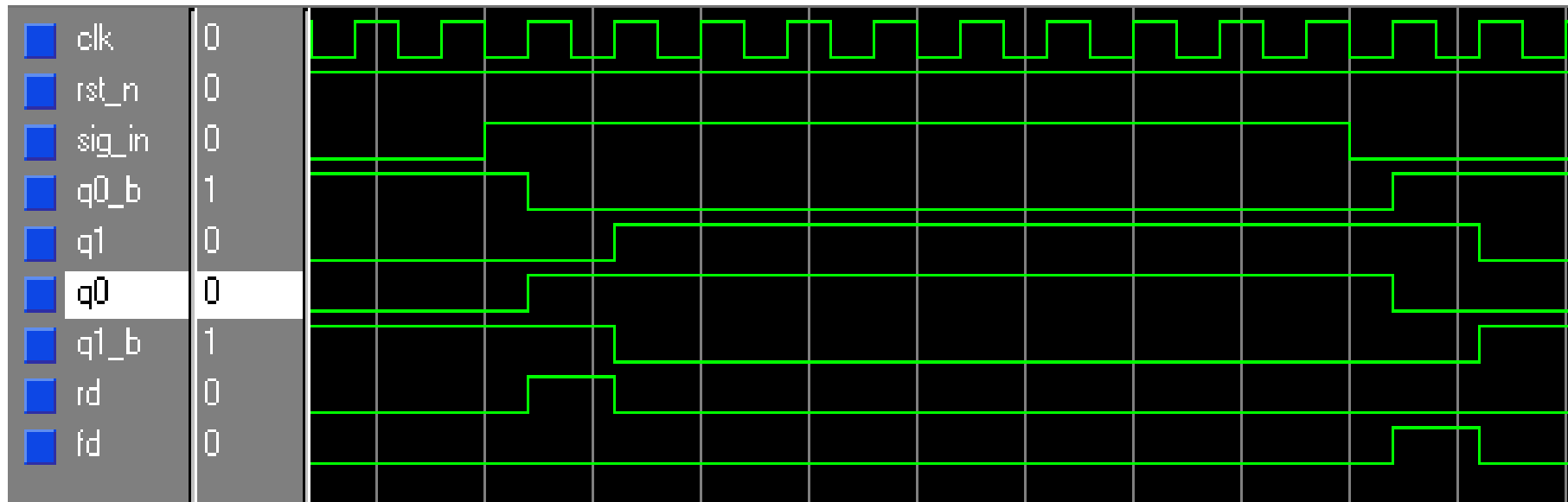
Our aim is to detect moment of change (either rise-detect or fall-detect) and to create “one-clock” length signal that will advice about sig_in changes.

- Use this one-cycle edge detection output to control data capture and other necessary functions – avoids multiple samples or counts per clock/data input pair

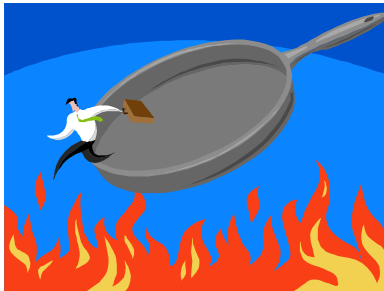
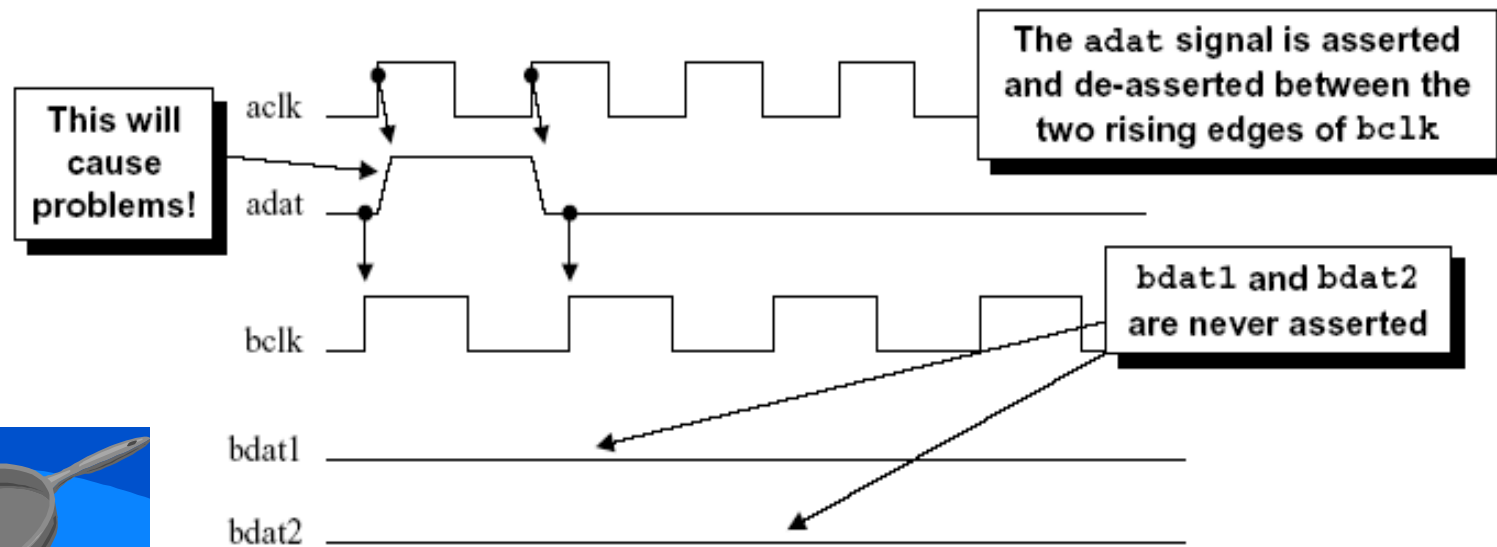
CDC - Edge detection (cont)



CDC - Edge detection (cont)



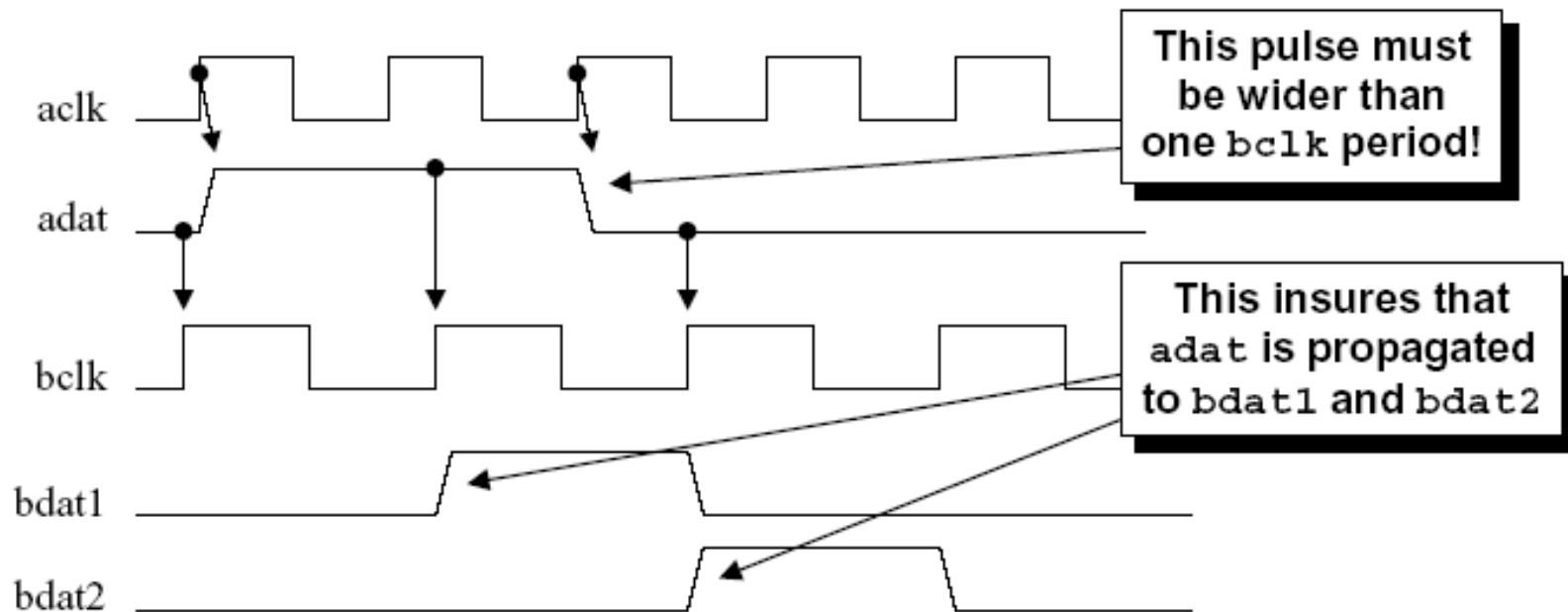
Synchronizing Fast Signals Into Slow Clock Domains (Fast 2 Slow)



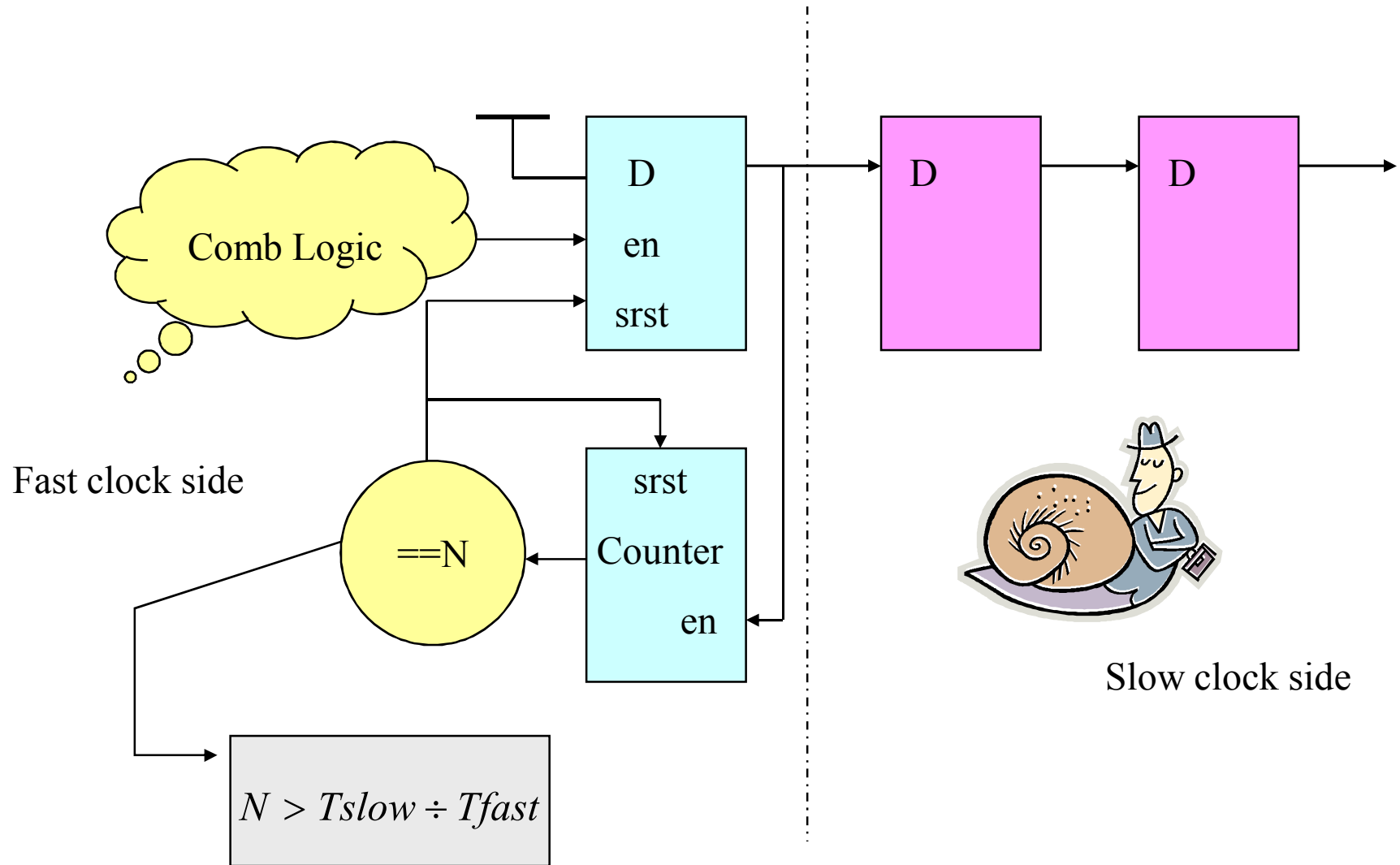
Fast 2 Slow

One potential solution to this problem is to assert control signals for a period of time that exceeds the cycle time of the sampling clock.

The assumption is that the control signal will be sampled at least once and possibly twice by the receiver clock.

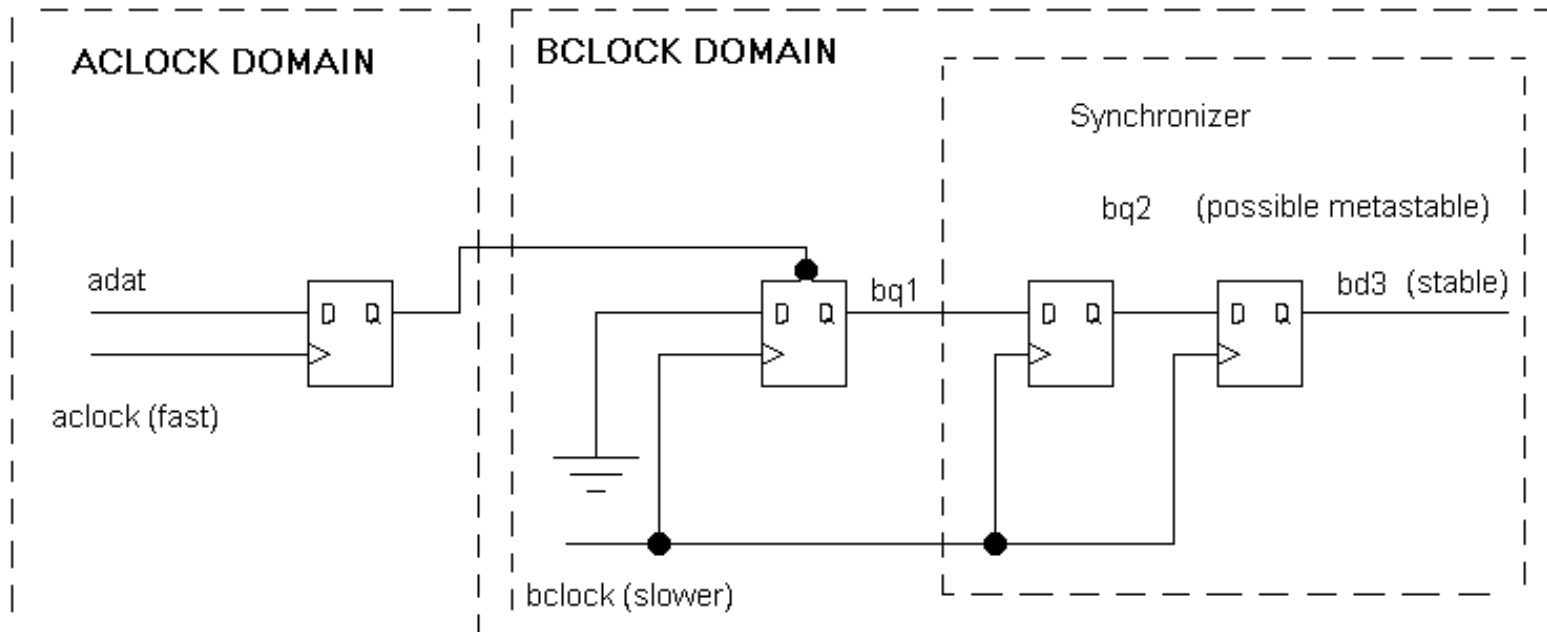


Fast 2 Slow



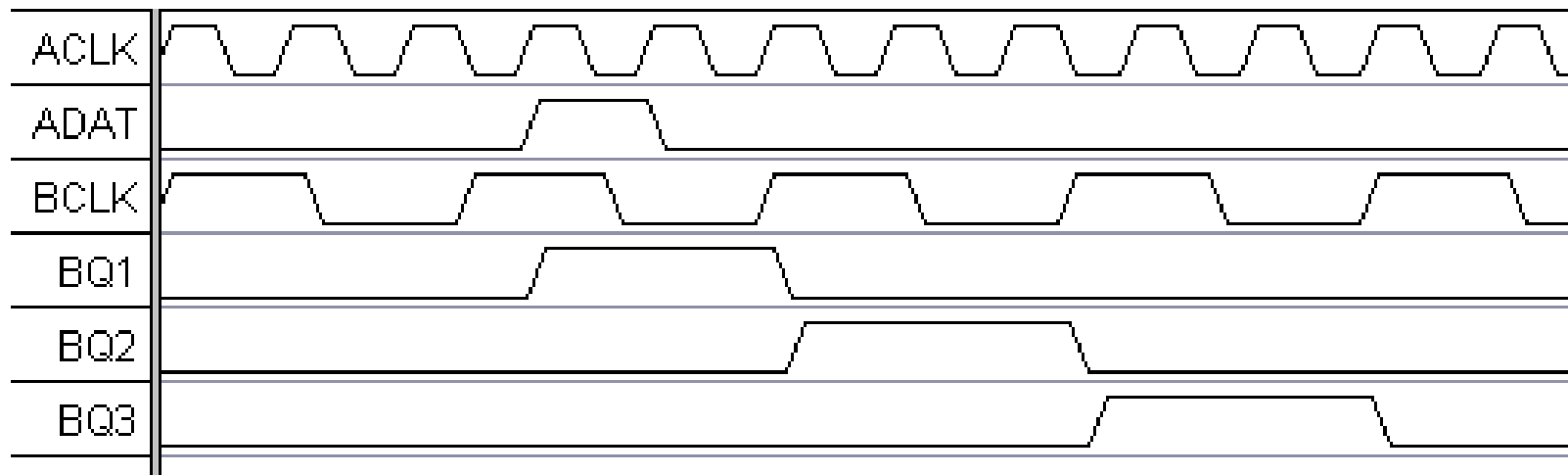
Fast 2 Slow

A second possible solution to this problem is pulse extracting (stretch)



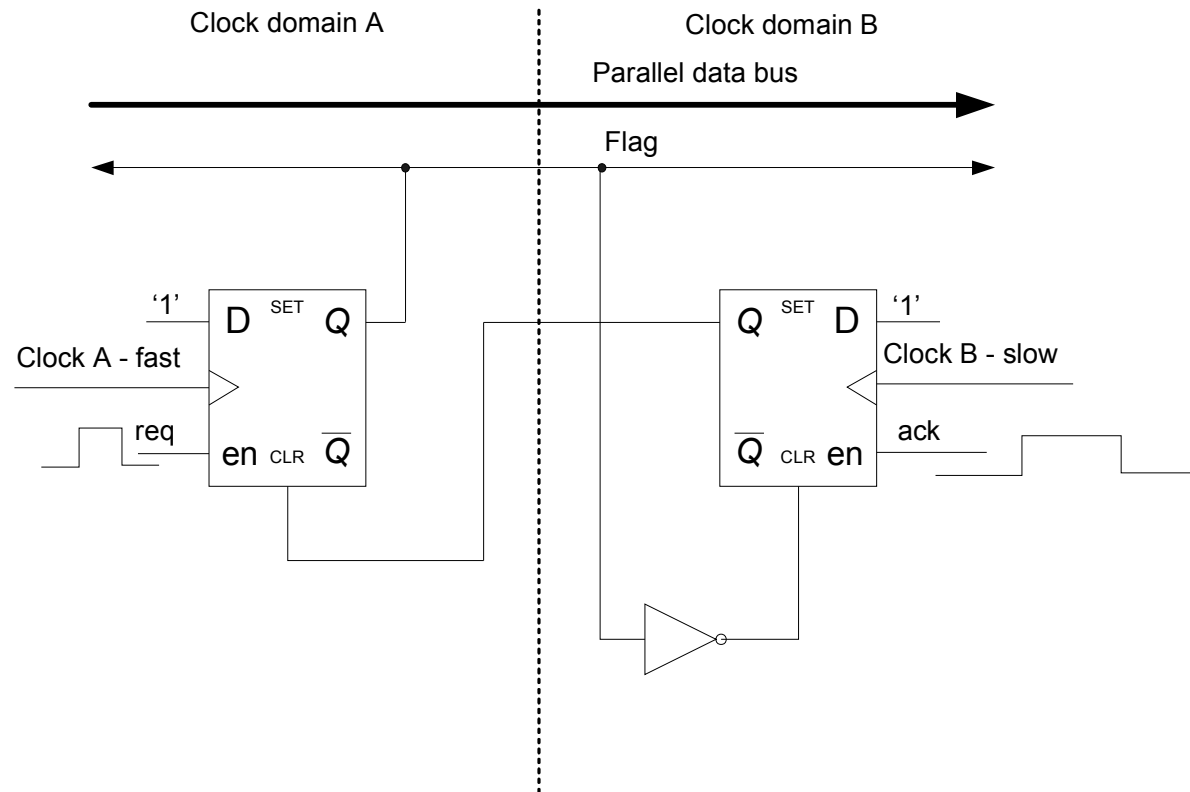
Fast 2 Slow

Lengthened pulse to guarantee that the control signal will be sampled



Fast 2 Slow

Using acknowledge – feedback flag

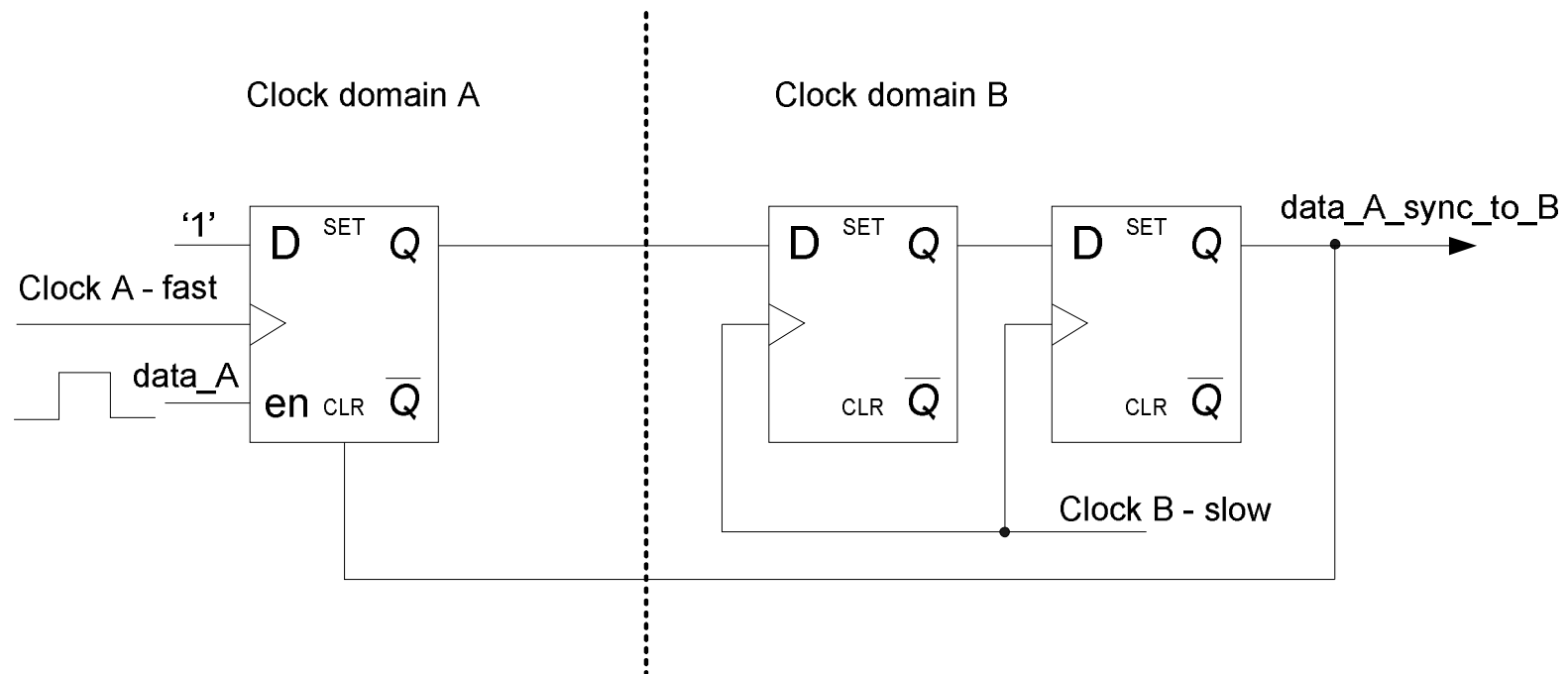


Disadvantages:

- ✓ Global reset is not used
- ✓ Asynchronous solution
- ✓ Uncertainty in timing constrain boundaries

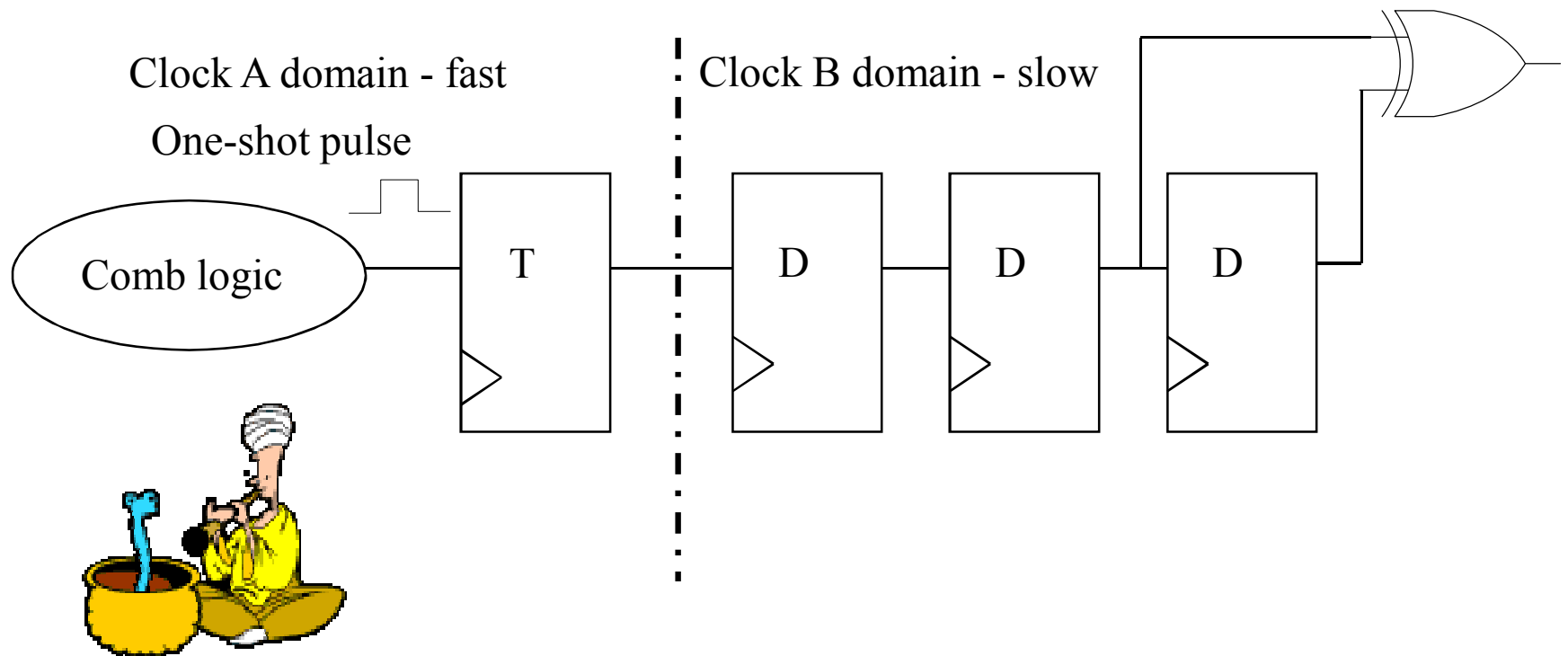
Fast 2 Slow

Better than previous solution, but still asynchronous



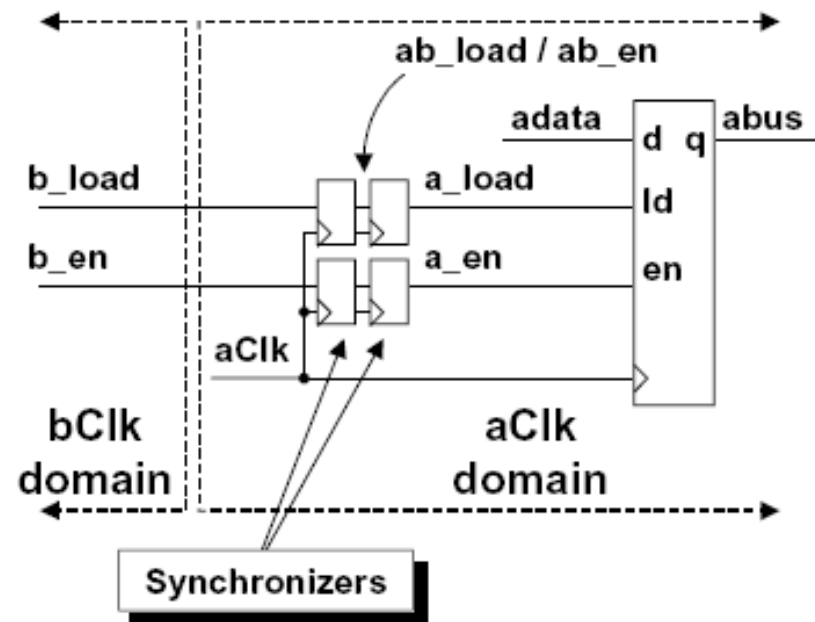
Fast 2 Slow

The synchronous decision of a transfers of single pulses between two clock domains problem applicable as well as in FPGA and in ASIC is below resulted completely:

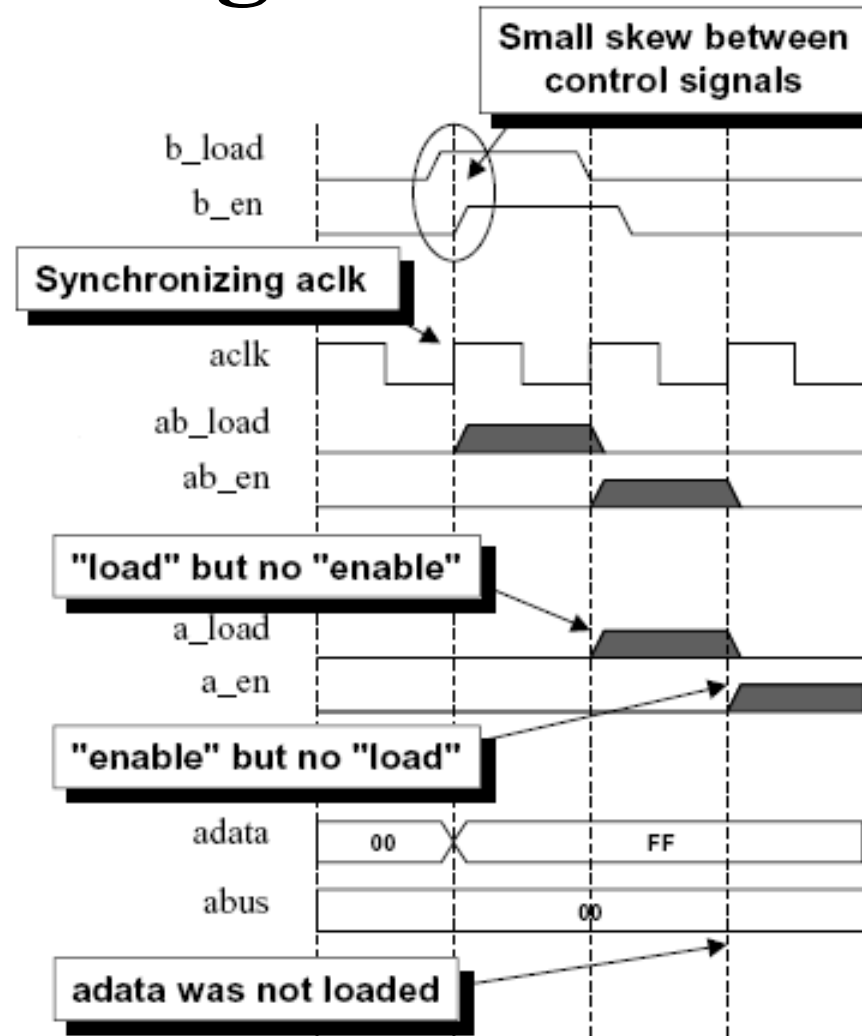
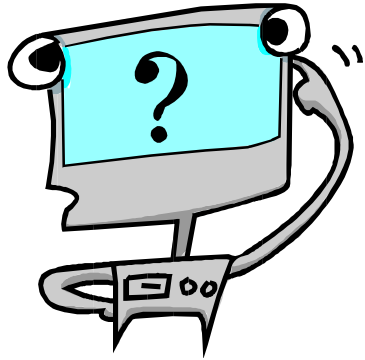


Two simultaneously required control signals

A register in the new clock domain requires both a load signal and an enable signal in order to load a data value into the register. If both the load and enable signals are being sent from one clock domain, there is a chance that a small skew between the control signals could cause the two signals to be synchronized into different clock cycles within the new clock domain.

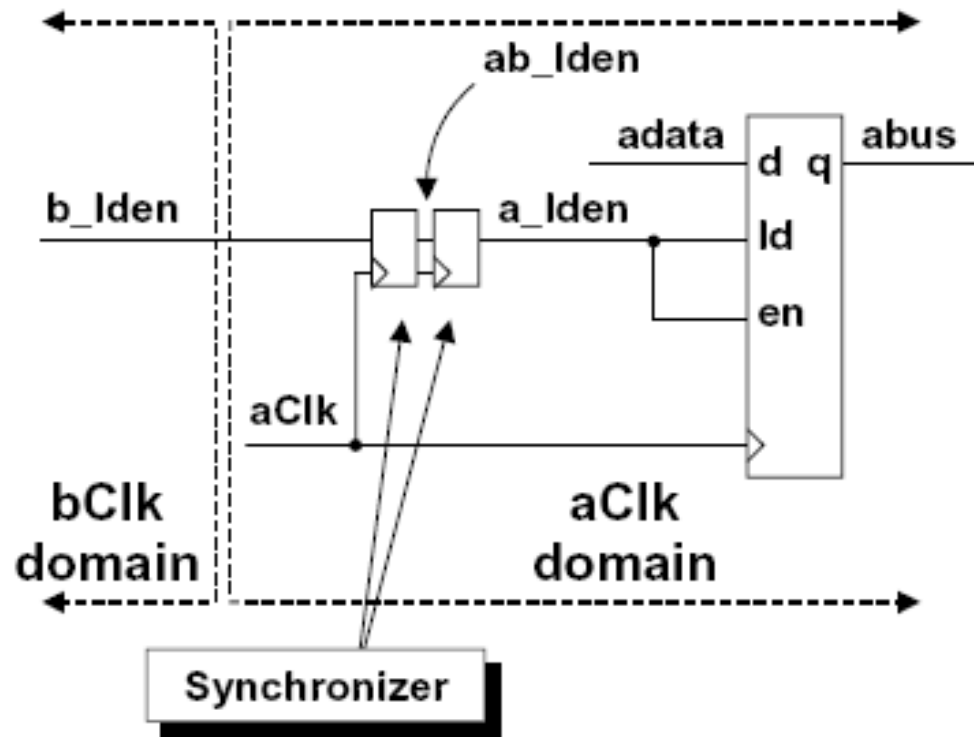
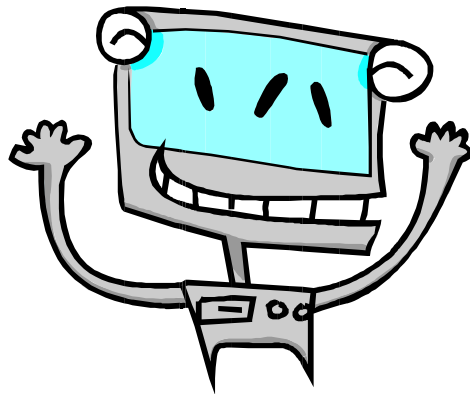


Two simultaneously required control signals

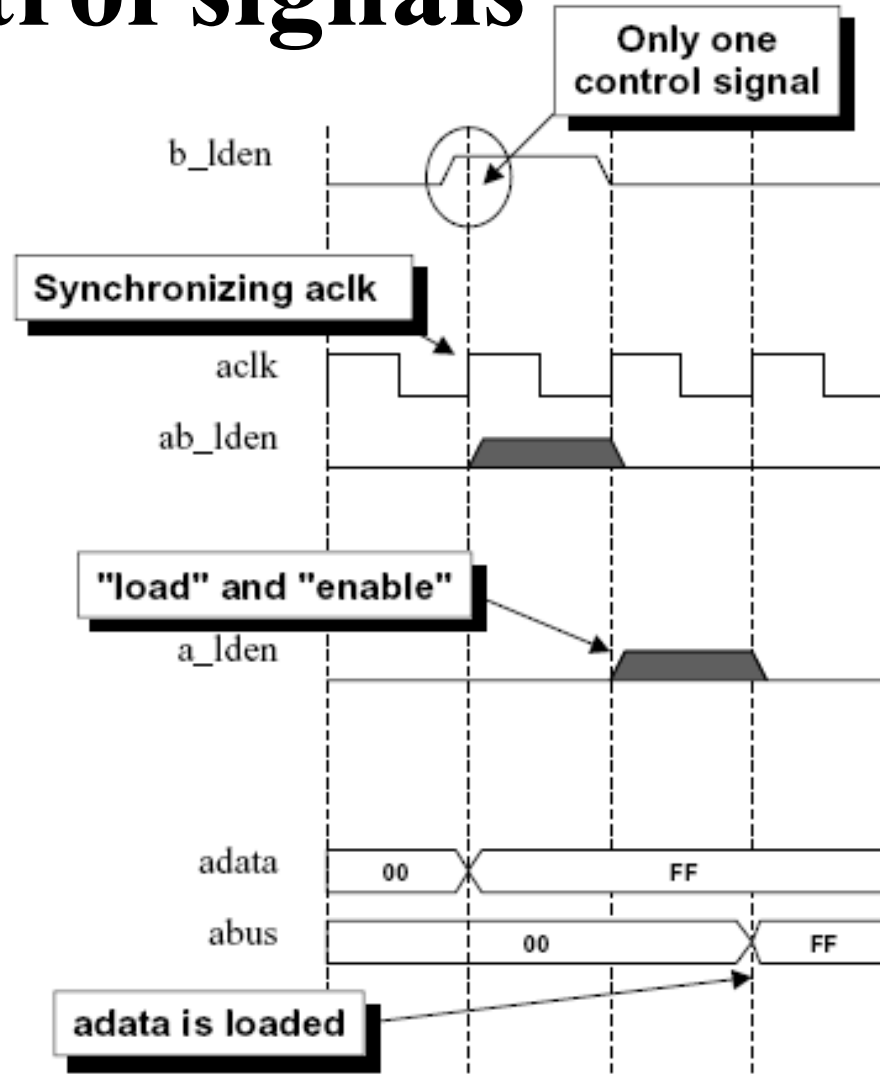
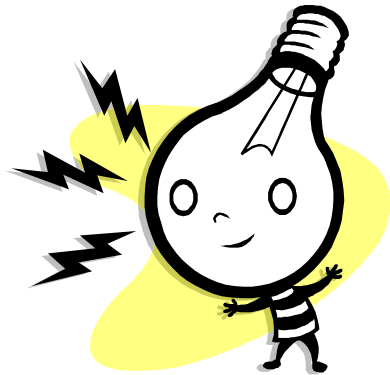


Two simultaneously required control signals

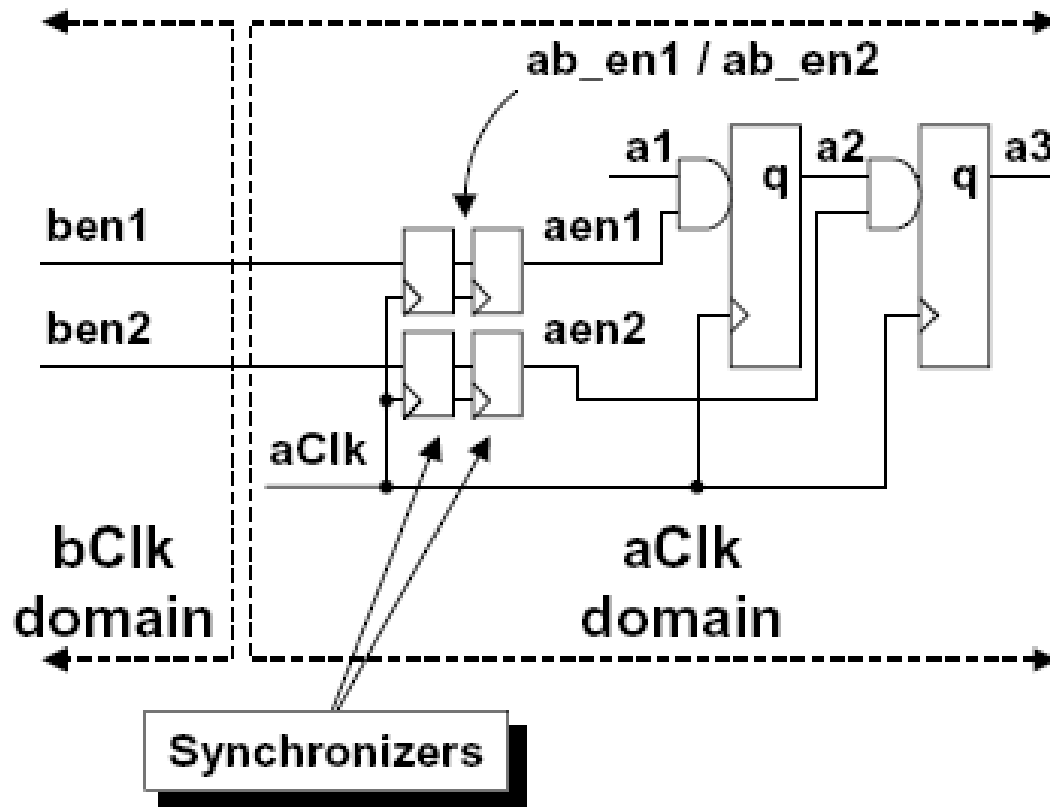
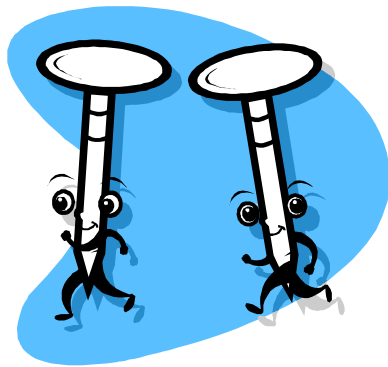
Solution - Consolidating control signals before passing them between clock domains



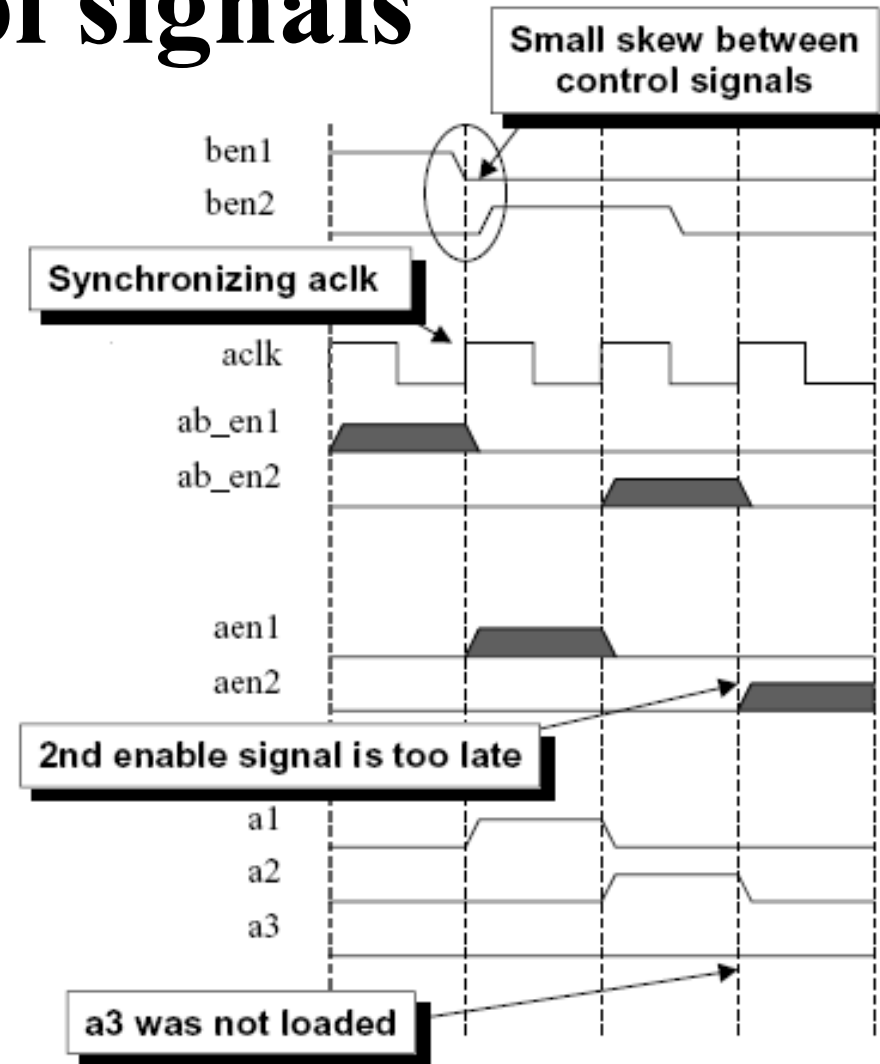
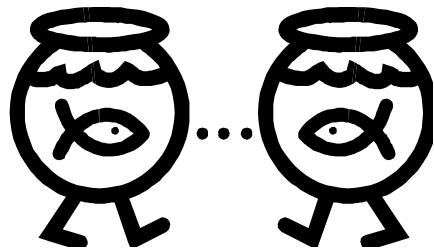
Two simultaneously required control signals



Two phase-shifted sequencing control signals

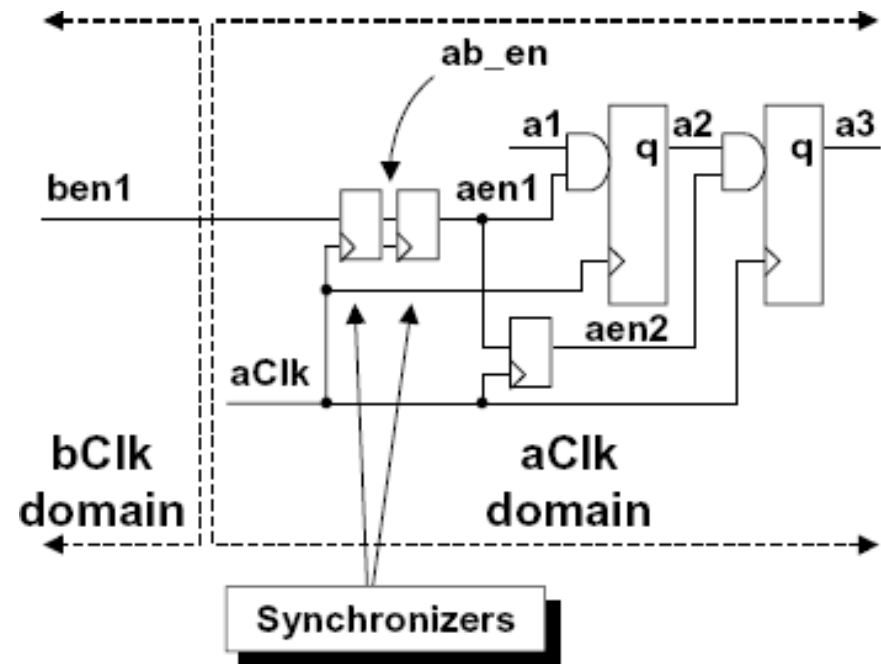
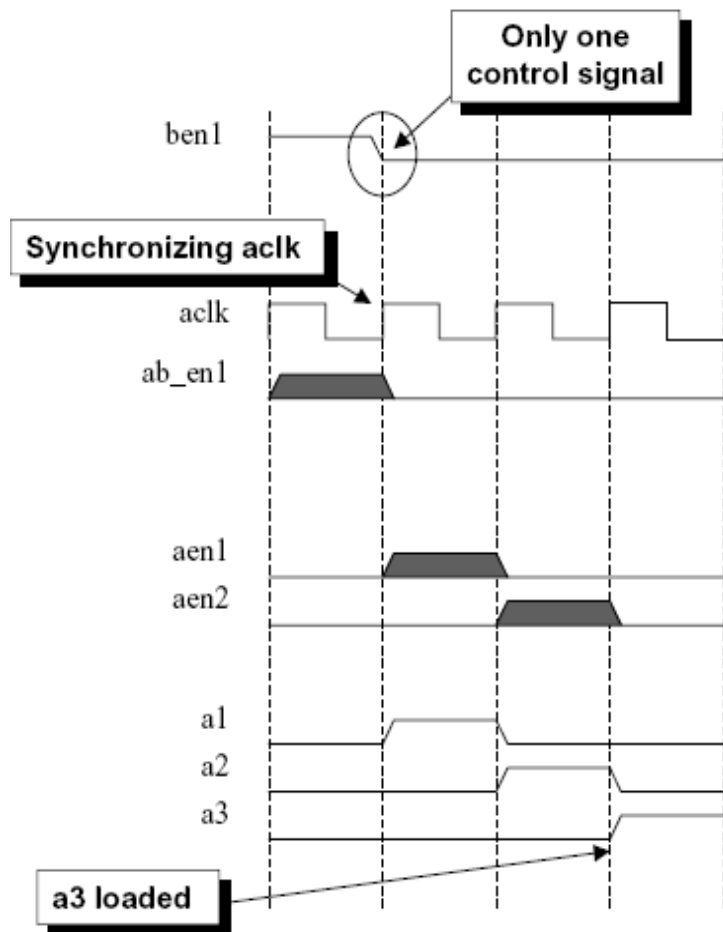


Two phase-shifted sequencing control signals

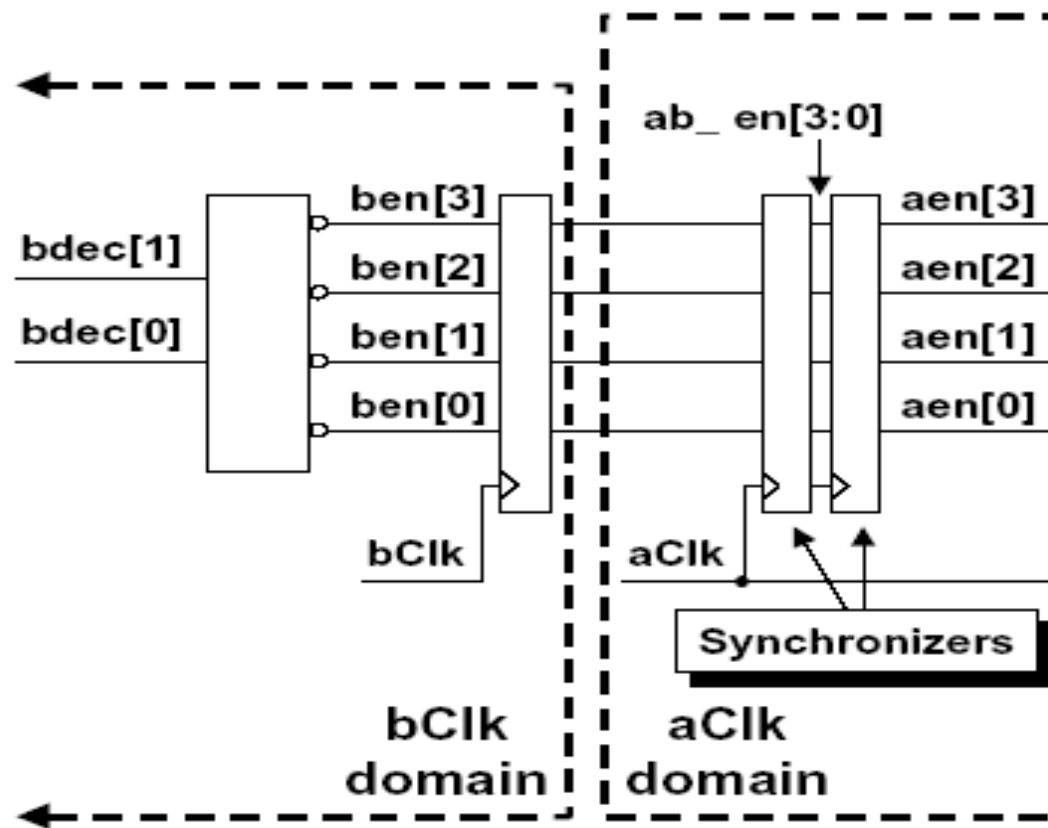
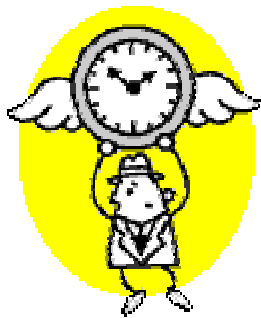


Two phase-shifted sequencing control signals

The solution to the problem is to send only one control signal into the new clock domain and generate the second phase-shifted sequential control signal within the new clock domain.



Control Buses synchronizing between clock domains



One-Hot Checker

Q: How to check the one-hot code ?

A: Use dual-rail code!

The dual-rail code for 4 bit example:

Y_1, Y_2, Y_3, Y_4 - the outputs from second register of receive clock domain, in which coded by one-hot code.

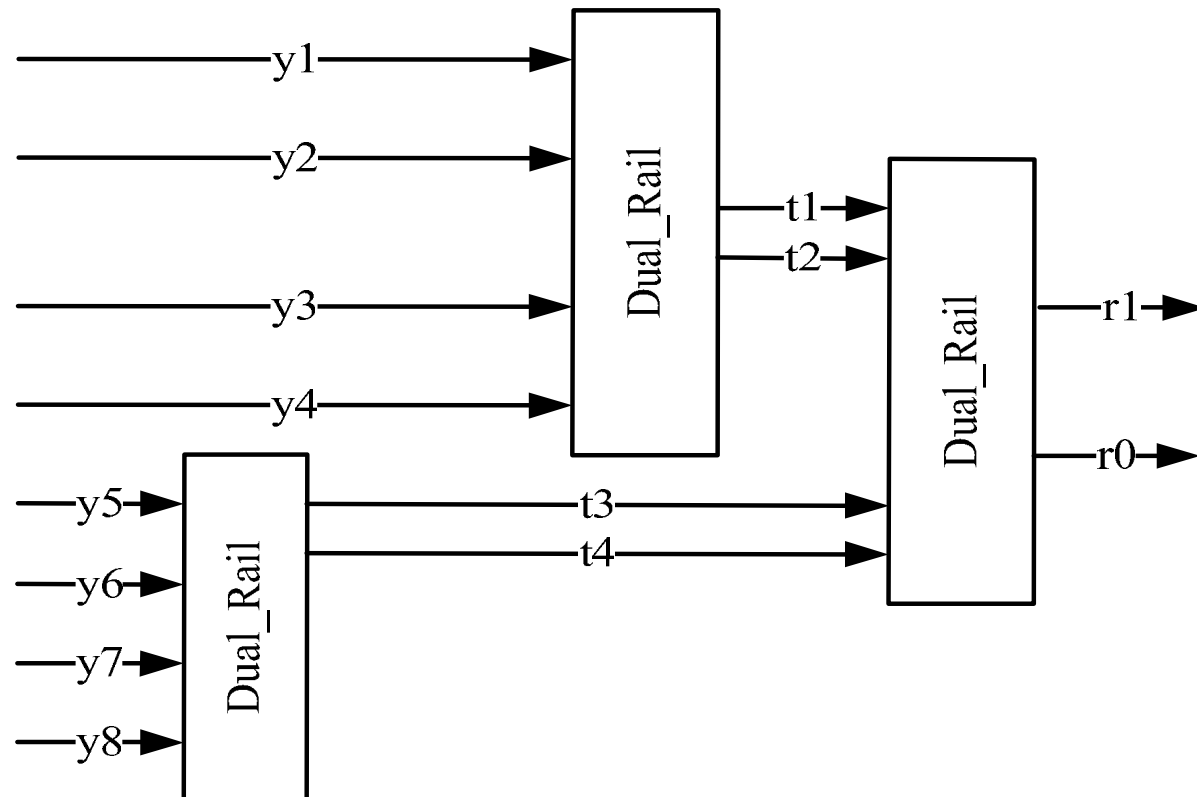
$$r_0 = y_1 + y_2 + y_3y_4 \quad r_1 = y_3 + y_4 + y_1y_2$$

The result of checking – **XOR** between outputs of these symmetric functions.

The dual-rail checker should be simply cascaded for any width of checked word.

One-Hot Checker

8 bit One-hot code checker architecture



Control Buses synchronizing between clock domains

Clock domain crossing with data XOR de-correlator/correlator

