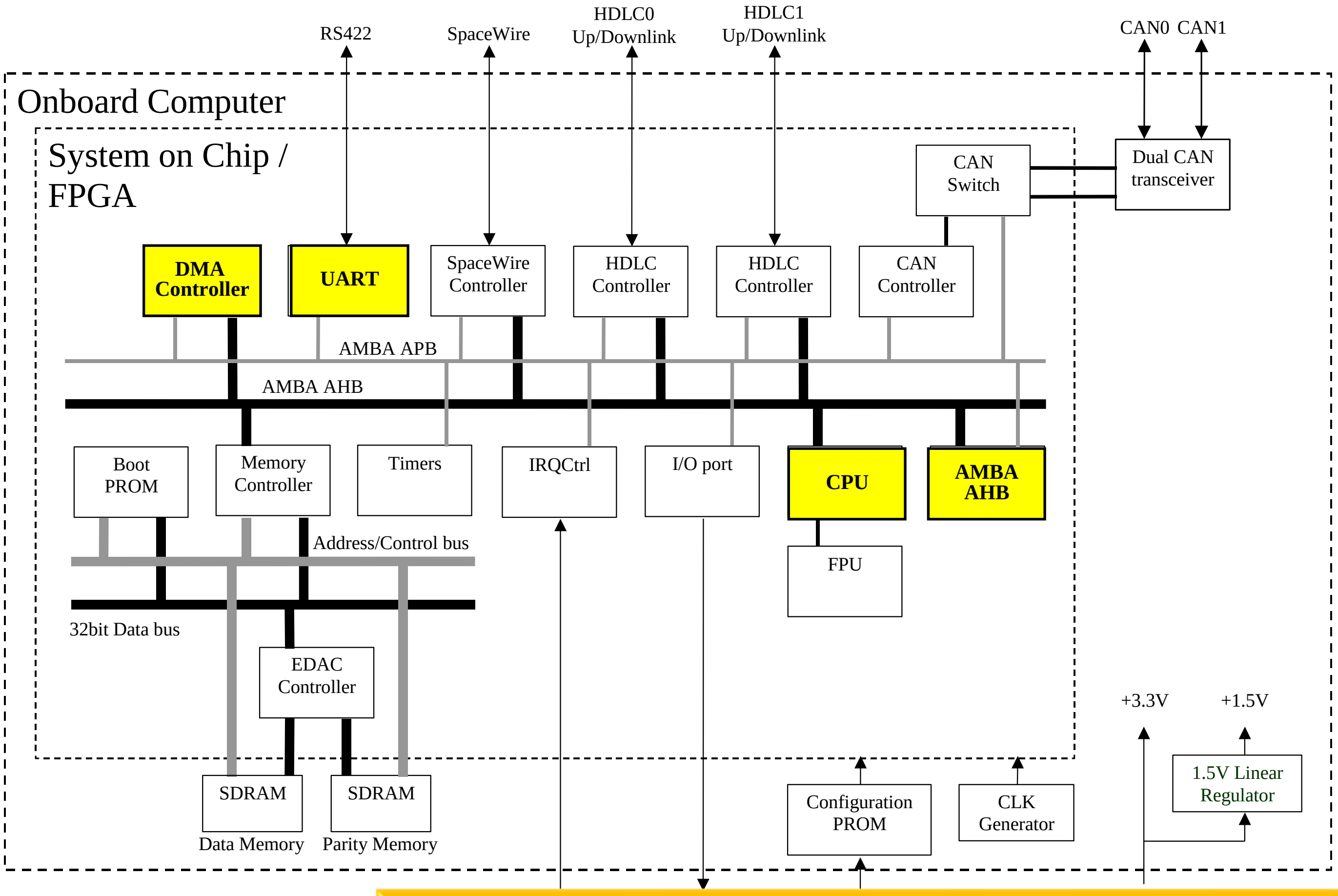
Avoiding Metastability in FPGA Devices

David Landoll
Applications Architect
Mentor Graphics Corp.

MAPLD 2009

Today’s FPGAs



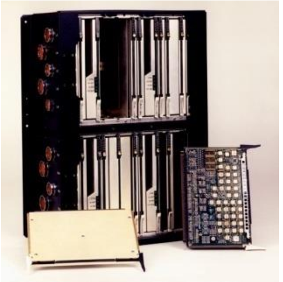
- Fabrication advances provide more available silicon area
- More functionality can weigh less and take up less space
- Integrating/reusing capabilities lowers cost

Integration Presents New Challenges

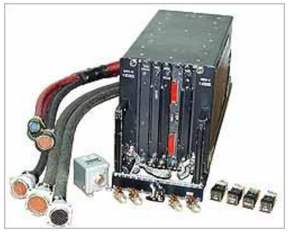
Weather Radar

Flight Management

Flight Control & Avoidance Systems



Integrated Avionics Processing



Maintenance Diagnostics

Image Processing

Boeing 787: Integration's Next Step

From its central processor to its common data network, surveillance system and navigation system, the theme of the Boeing 787 Dreamliner is integration.

James W. Ramsey

Communications

Such integration usually involves multiple independent clock domains, which leads to clock-domain crossings and metastability errors!

Clock Domain Crossing (CDC) Errors

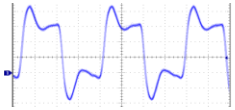
Unpredictable Loss of Data

- CDC problems
  - corrupt control and data signals
  - are subtle, intermittent, unpredictable
  - are the 2nd major cause of respins
  - are difficult to reproduce and debug
  - are temperature, voltage, and process sensitive
  - *will only* occur in hardware; often in the final design
- Traditional verification techniques *do not* work for CDC signals

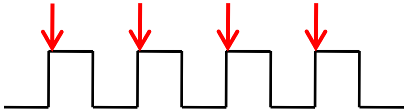
A CDC Verification methodology is needed to reduce the risk of CDC related data errors

Metastability
What the heck is it, anyway?

- **What is a clock?**



  - **Periodic pulsing signal**



Vcc, Vdd : **+5V, +3.3V**

Vee, Vss: GND: **0V**

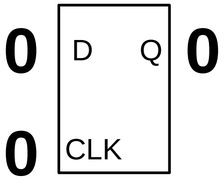
  - **Digital logic uniformly connected to this signal**
  - **Acts as the Symphony Conductor – keeps logic in sync**
  - **Action happens across the logic at one specific point**
    - **Typically the “rising edge”**

Metastability
What the heck is it, anyway?

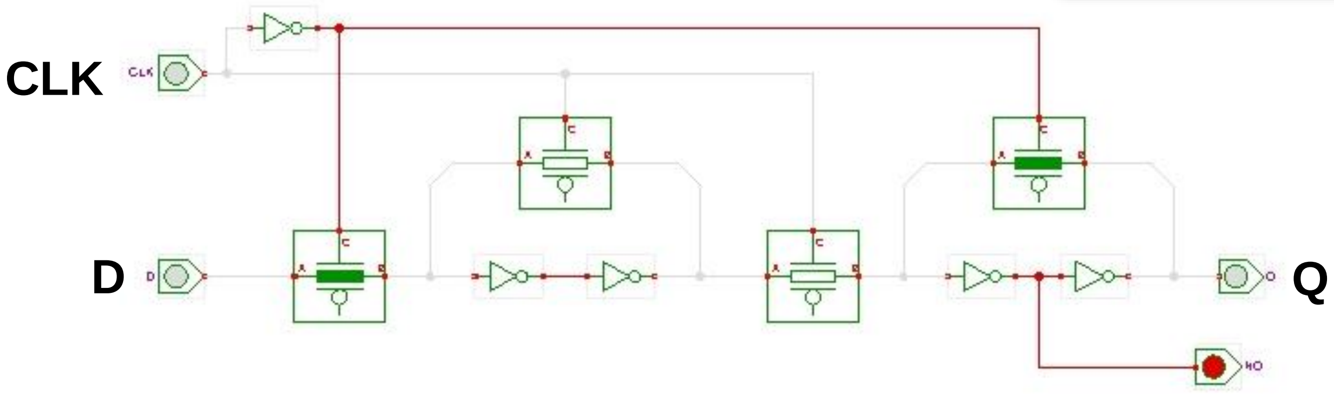
- **What's in a register?**
  - (Also known as a latch, flip-flop, etc)
  - **Contain transistors that "trap" the input value at the appropriate time**
    - E.g. rising edge of the clock
  - **How does this happen?**

Metastability
The Physics of a Register

- Let’s take a look at a register
  - CMOS D-type transmission gate flip-flop



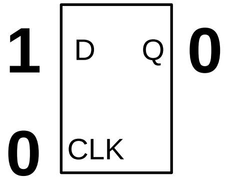
```
-- simple D-type flip-flop
process(CLK)
begin
  if rising_edge(CLK) then
    Q <= D;
  end if;
end process;
```



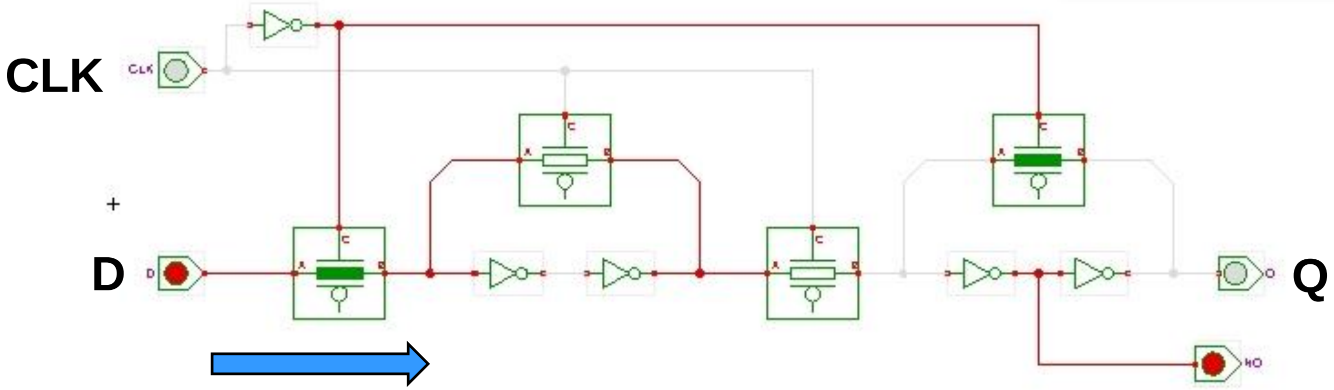
Transistor Model of a D Flip-Flop

Metastability
The Physics of a Register

- Let’s take a look at a register
  - CMOS D-type transmission gate flip-flop



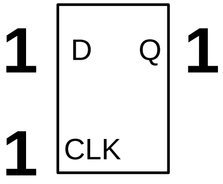
```
-- simple D-type flip-flop
process(CLK)
begin
  if rising_edge(CLK) then
    Q <= D;
  end if;
end process;
```



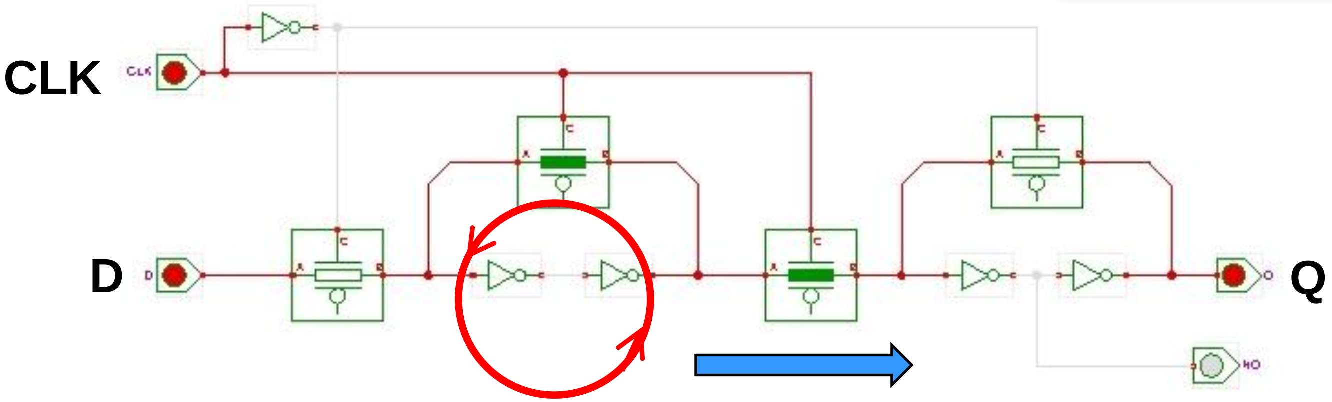
Transistor Model of a D Flip-Flop

Metastability
The Physics of a Register

- Let’s take a look at a register
  - CMOS D-type transmission gate flip-flop



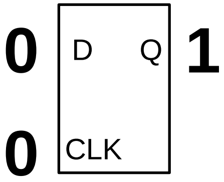
```
-- simple D-type flip-flop
process(CLK)
begin
  if rising_edge(CLK) then
    Q <= D;
  end if;
end process;
```



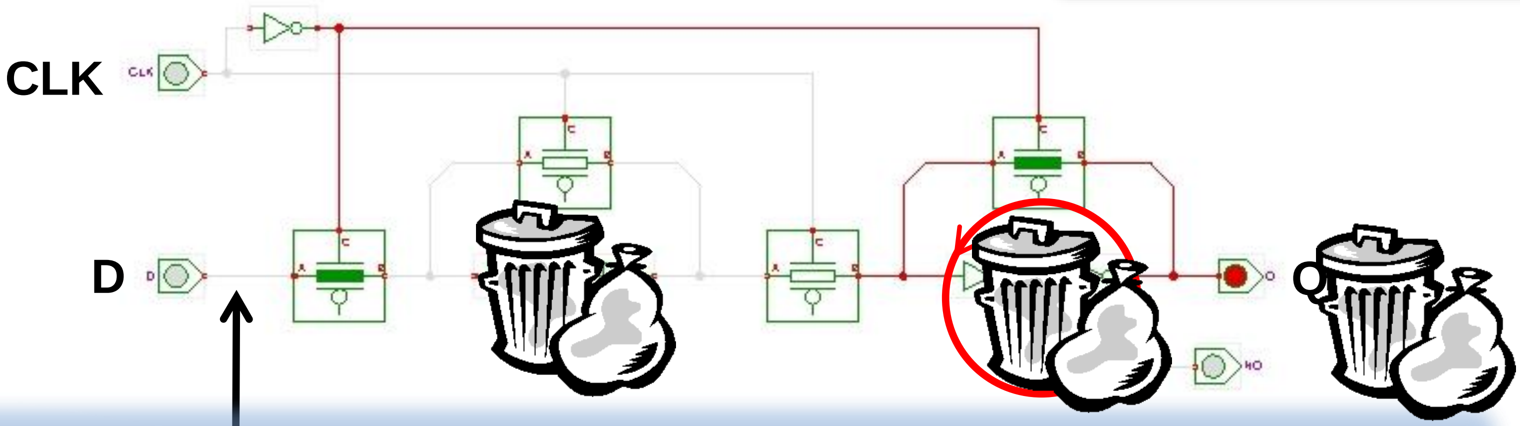
Transistor Model of a D Flip-Flop

Metastability
The Physics of a Register

- Let’s take a look at a register
  - CMOS D-type transmission gate flip-flop



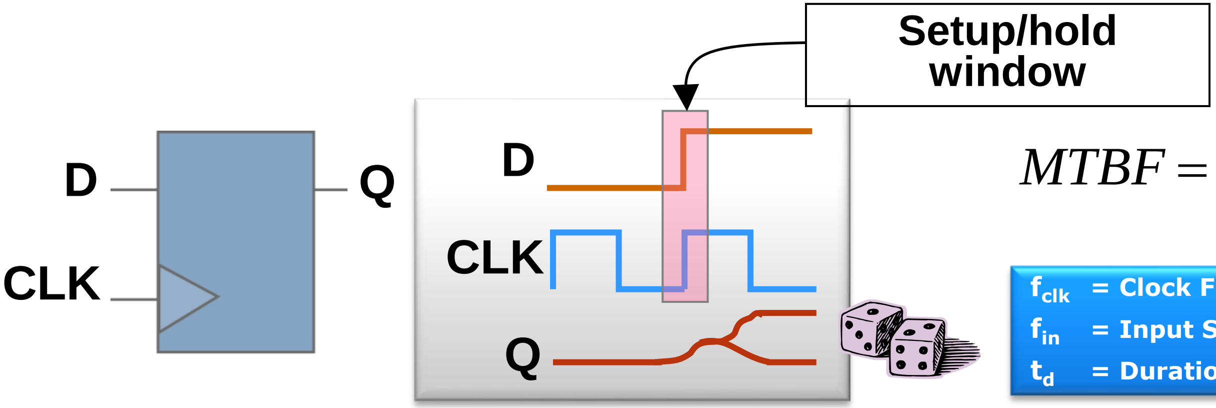
```
-- simple D-type flip-flop
process(CLK)
begin
  if rising_edge(CLK) then
    Q <= D;
  end if;
end process;
```



Only works if D has a “good value” at the rising edge of the clock (no Set-up/hold time violations)

Metastability
The Physics of a Register

- When setup/hold conditions are violated, the output of a storage element becomes unpredictable



$$MTBF = \frac{1}{f_{clk} \times f_{in} \times t_d}$$

f_{clk} = Clock Frequency

f_{in} = Input Signal Frequency

t_d = Duration of critical time window

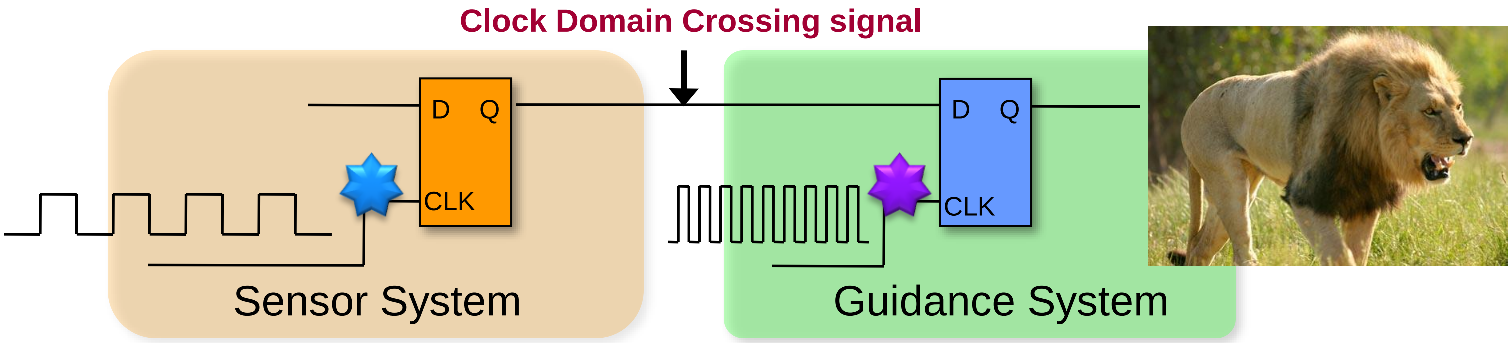
- This effect is called **metastability**
- **If not contained, metastability can propagate…**

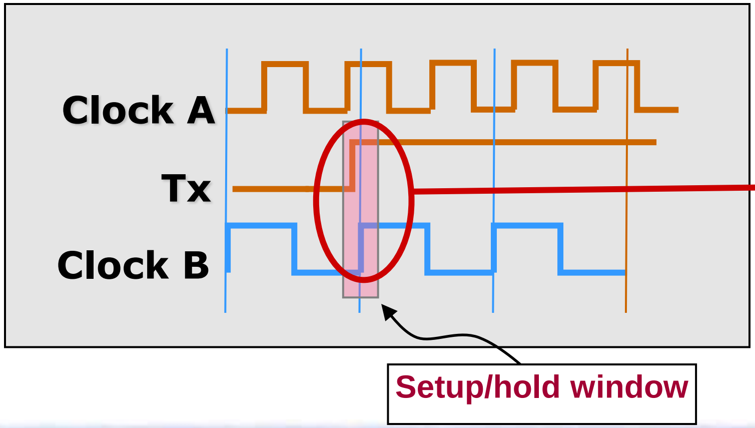
Metastability is UNAVOIDABLE in designs with multiple asynchronous clocks

Clock Domain Crossings
Guaranteed to Cause Metastability

When 2 or more designs run on disparate clocks:

- The clocks will continually skew, guaranteeing setup/hold violations
- Signals from one design to another are "Clock Domain Crossings" (CDCs)





Signals that cross asynchronous clock domains (CDC signals) WILL violate setup and hold conditions

Mitigating Clock Domain Crossing Issues

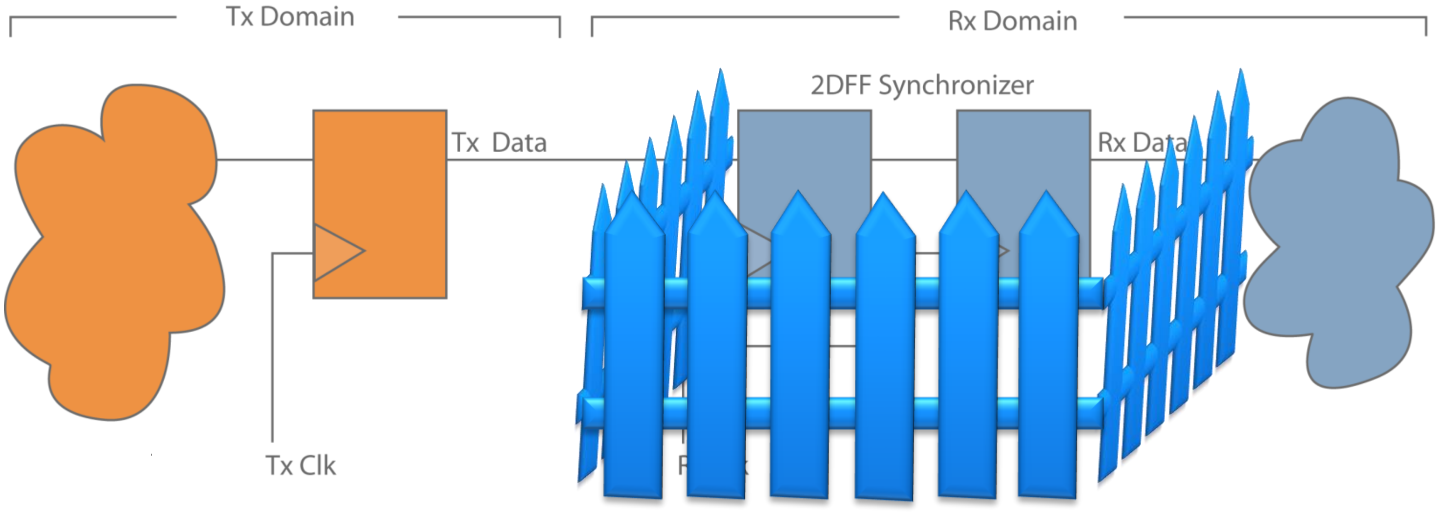
Problem:

- Signals crossing a clock domain will violate set-up/hold
- **Impact:** Control/data signals will be dropped/corrupted
  - *Loss of Data*

Approaches:

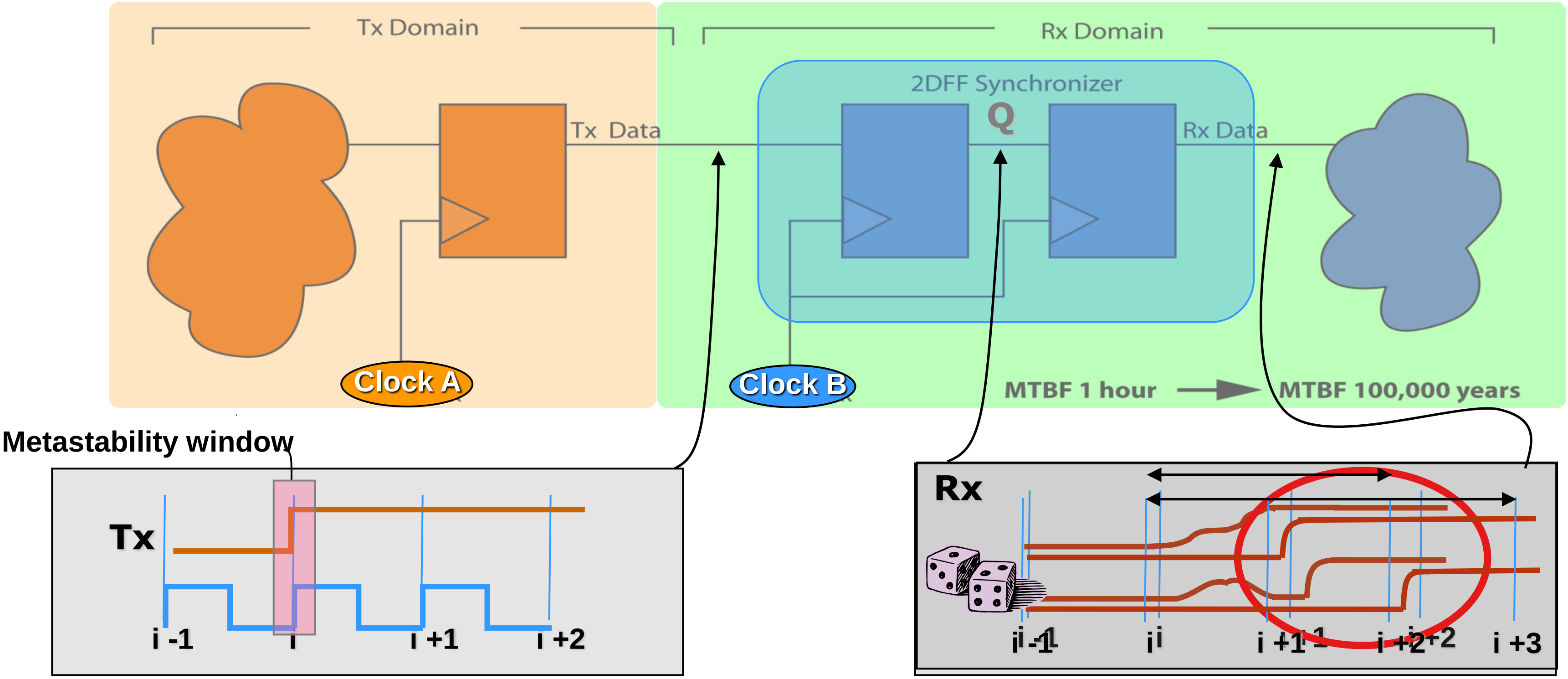
- **Avoid having systems that have multiple clocks**
  - Although sensible, it's becoming impossible
- **Design around the problem**
  - Designer can add "synchronizers" to the design
  - Metastability still happens, but nobody else sees it
    - E.g. 2DFF, FIFO, etc.
    - **"Fences in" metastability**

Isolate Metastability: Synchronizers



- **Designers add synchronizers to reduce the probability of metastable signals**
- **Synchronizers are sub-circuits that can prevent metastable values from being sampled across clock domains**
  - Take unpredictable metastable signals and create predictable behavior

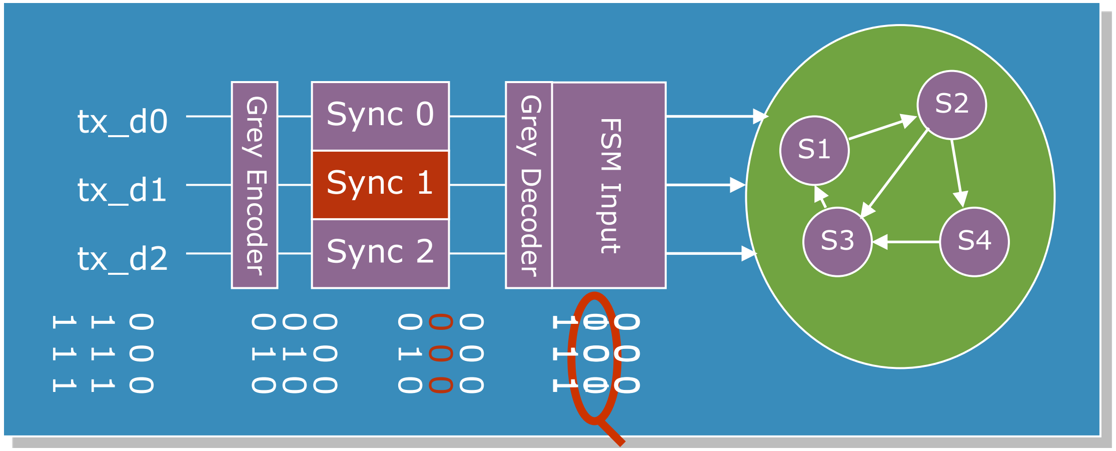
Mitigating Clock Domain Crossing Issues
Isolate Metastability: Synchronizers



When metastability occurs, the delay through a synchronizer becomes unpredictable

Synchronizer Delays Can Reconverge *with unexpected results*

- **CDC signals cross with an assumed relationship**
- **Can be combinational, sequential, or deeply sequential**
- **Unpredictable delays on CDC paths lead to reconvergence errors**
  - Designs need logic to correctly handle reconvergence
  - Can occur on single-bit or multiple-bit signals

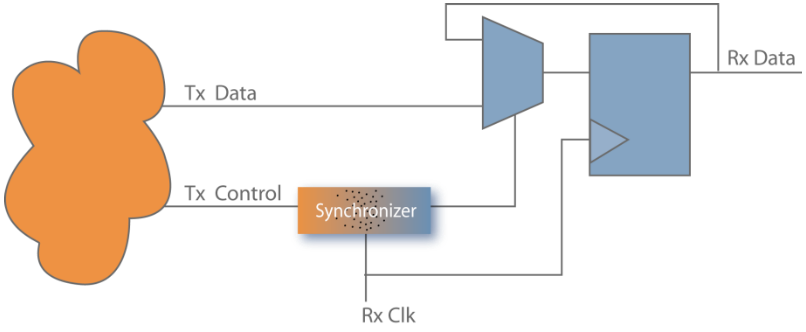


Invalid Command

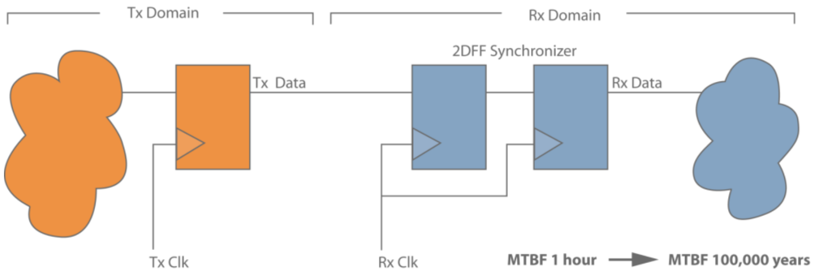
Valid Command – but delayed

And, Synchronizers Fail if Misused

- Synchronization between clock domains *requires* a transfer protocol
  - Ensures data is *predictably* transferred between domains
- These protocols *must* be verified
- When protocol is violated
  - Data is lost
  - Simulation *may not* show a failure
  - Silicon *will* eventually show a functional error

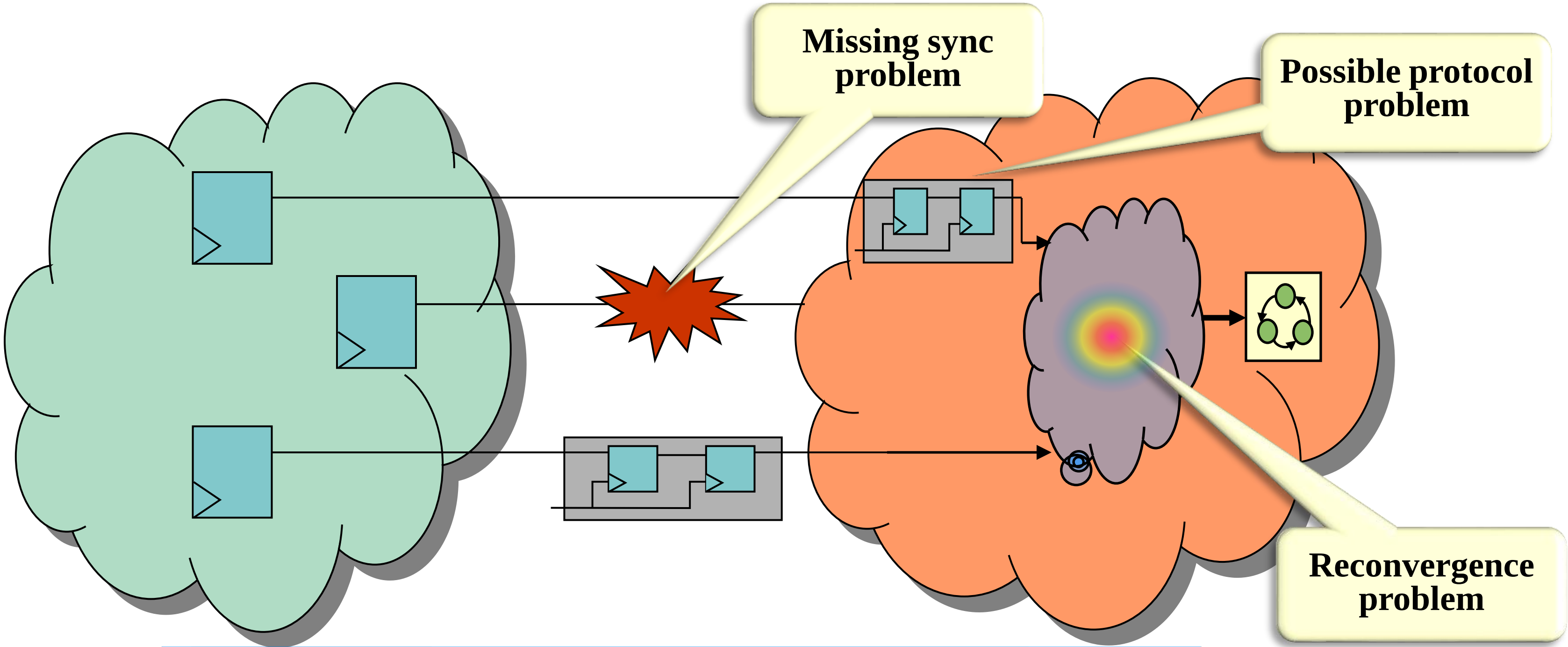


Data-Mux (DMUX) Synchronizer



Synchronizer won't function properly if the required Transfer Protocol is violated

Verification Must Cover All Three CDC Problems



Clock domain crossings need:

- Structured synchronization
- Transfer protocols
- Global reconvergence checking

Mitigating Clock Domain Crossing Issues

- **Problem:**
  - Signals crossing a clock domain will violate set-up/hold
  - **Impact:** Control/data signals will be dropped/corrupted
- **Approaches:**
  - Avoid having systems that have multiple clocks
  - Designer can add "synchronizers" to the design
  - **Designer-added synchronizers + full CDC verification**
    - **Assures synchronizers are present and used correctly**

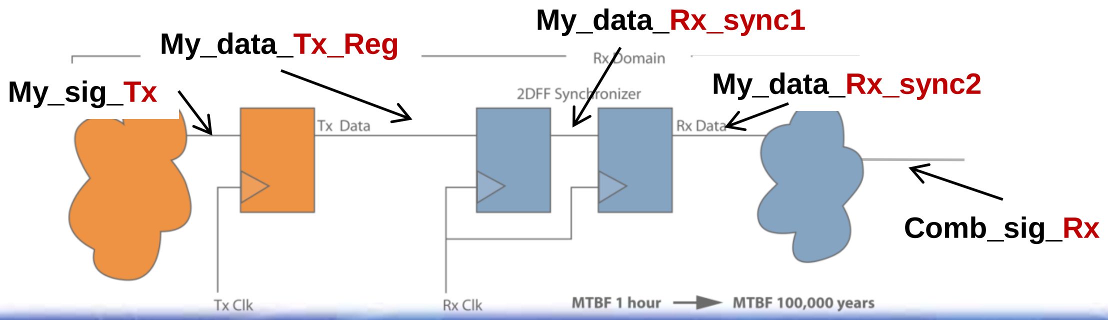
Recommendations

During design planning

1. Create systems/designs using 1 clk, 1 edge when possible
2. If multiple clocks are required, try to use 1 designer for both clock domains, and use coding guidelines
    1. Use signal naming conventions
    2. Many clock domain errors come from design changes, not the initial design

For Example:

- Append "_A_reg" to signals leaving A-clk register, "_A" for A-clk combo signals
- Leverage during code reviews - help identify missing synchronizers
- Make sure ONLY _A_reg signals go to synchronizers (no combo logic)



My_data_Tx_Reg

My_data_Rx_sync1

My_sig_Tx

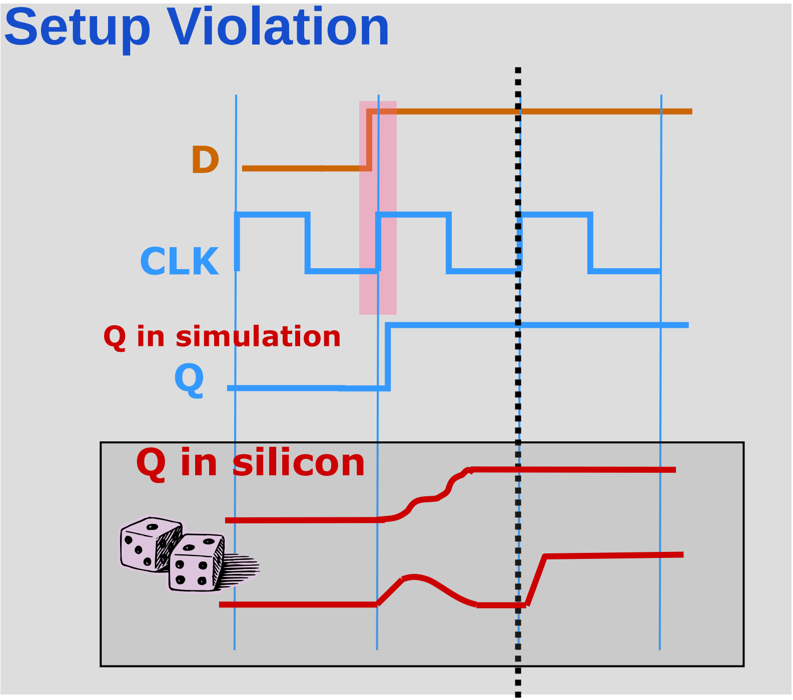
My_data_Rx_sync2

Comb_sig_Rx

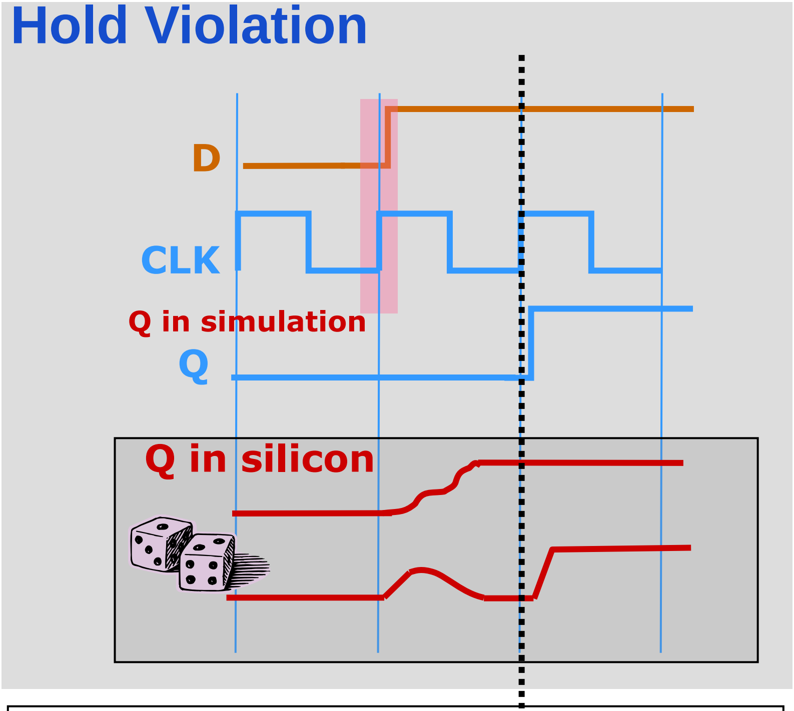
Verifying CDC Synchronization

- **Problem:**
  - Missing synchronizers will create metastability
  - Correctly placed but misused synchronizers won’t work
  - Reconvergence of synchronized signals can create unexpected behavior
- **Approaches:**
  - **Simulation**
    - **Digital logic simulators do NOT model transistor behavior**
    - **Do not model “metastability”**

For example …



Simulation captures a ‘1’ while silicon produces either a ‘1’ or ‘0’



Simulation captures a ‘0’ while silicon produces either a ‘1’ or ‘0’

Simulation Does NOT Reflect Silicon Behavior

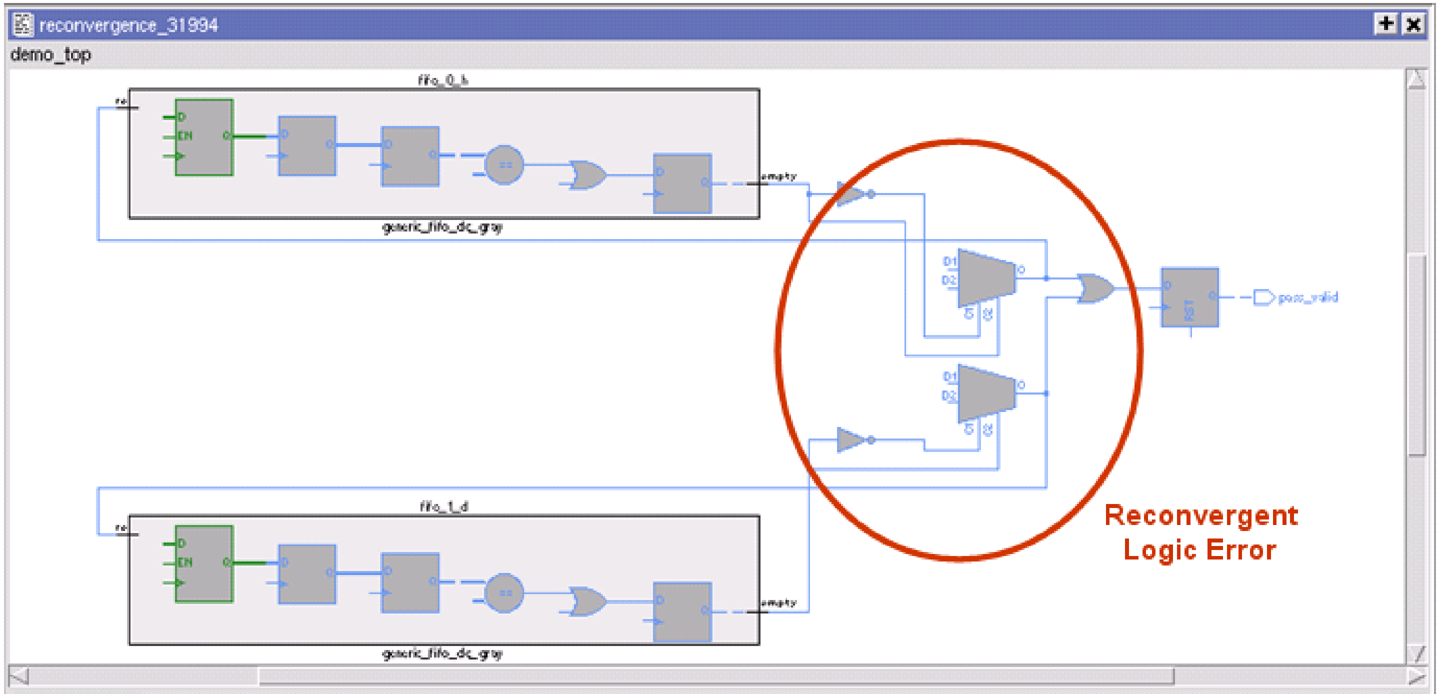
Verifying CDC Synchronization

- **Problem:**
  - Missing synchronizers will create metastability
  - Correctly placed but misused synchronizers won't work
  - Reconvergence of synchronized → Control logic bugs
- **Approaches:**
  - **Simulation**
    - Won't model CDC's correctly to detect errors
  - **Static Timing Analysis**
    - Can be used to identify signals that cross domains
    - Can be used as input for a manual review
    - But…Won't detect missing or incorrectly used synchronizers, or reconvergence

Verifying CDC Synchronization

- **Problem:**
  - Missing synchronizers will create metastability
  - Correctly placed but misused synchronizers won’t work
  - Reconvergence of synchronized ➔Control logic bugs
- **Approaches:**
  - **Simulation**
    - Won’t model CDC’s correctly to detect errors
  - **Static Timing Analysis**
    - Identifies signals for manual review, but otherwise useless
  - **Manual Design Reviews**
    - Error prone (and very time consuming)
    - Typically only identifies synchronizer structures, misses reconvergence and invalid sync protocol usage
    - Evidence suggests at least some synchronizers will be missed

For Example…
Trivial Reconvergence Error



Reconvergent Logic Error

- **Reconverging synchronized CDC signals - timing is unpredictable.**
- **Need to verify the downstream logic can handle variations**
  - Manually identifying the reconvergence is very hard
  - Manually identifying all possible behaviors is harder
  - Manually assuring logic will behave correctly – typically intractable

Verifying CDC Synchronization

- **Problem:**
  - Missing synchronizers will create metastability
  - Correctly placed but misused synchronizers won't work
  - Reconvergence of synchronized ➔Control logic bugs
- **Approaches:**
  - **Simulation -** ***Won't model CDC's correctly to detect errors***
  - **Timing Analysis -** ***Identifies signals for review, but otherwise useless***
  - **Manual Design Reviews -** ***error prone, incomplete***
  - **Lab Verification?**
    - Problem is intermittent, debug is impossible
  - **Spice simulation? – It *does* model transistors, but…**
    - Where will you get the "Spice deck"? (transistor level model)
    - Would be far too slow on a large FPGA

Verifying CDC Synchronization

- **Problem:**
  - Missing synchronizers will create metastability
  - Correctly placed but misused synchronizers won’t work
  - Reconvergence of synchronized ➔Control logic bugs
- **Approaches:**
  - **So - we need a new method that reliably:**
    - Identifies ALL CDC signals, structures, reconvergence
    - Assures ALL connected, functioning correctly
    - Creates reports for manual reviews

    ➔ **The EDA industry has responded**
    - 6 commercial tools now available…and counting
    - **But…most won’t identify all 3 of our CDC issues**

Mentor's CDC Verification Technology

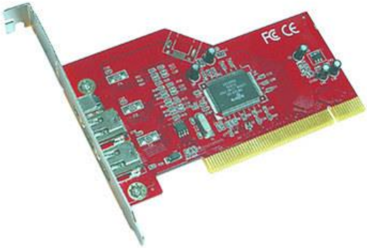
Who's using our technology?

- Mil-Aero
  - Honeywell, Inc.
  - L-3 Communications
  - Lockheed Martin Co
  - Ministry of Aerospace & Aeronautics
  - Northrop Grumman Corp
  - Raytheon
  - Rockwell Collins Inc.
  - SAAB Group
  - Thales
- Commercial
  - Widely used in commercial space

- *The market leader in CDC verification*

Example Value from One Customer

- **Design**
  - IEEE standard serial communications core
  - Used in 50-60 other COMMERCIAL ASIC products
  - *Widely* deployed (millions in use daily)
- **Placed core in a sensor guidance system**
  - Found issues in the lab
  - Debugged FPGA for weeks
  - Suspected a CDC issue, but not sure…
- **Deployed Mentor's CDC solution**
  - Results same day
  - ***Found 199 serious CDC bugs!***
    - **45 Missing Synchronizers**
    - **83 Incorrect Synchronizers**
    - **76 Reconverging Signals**
    - **11 other problems**
  - ***Most resulting from "more stressful" usage***
- ***In production:***
  - ***Commercial ASIC : Customer issue – device is erratic, locks up***
  - ***Avionics: Could result in an Airworthiness Directive***



Summary
Recommendations

✓**During design planning**

1. Create systems/designs using 1 clk, 1 edge when possible
2. If multiple clocks are required, try to use 1 designer for all clock domains
3. When multi-clock design is required, plan for proper verification

✓**During verification**

1. Watch for multiple clocks in designs ***(Tip – Count PLLs)***
2. Ask how CDC issues are mitigated (remember there are 3)

✓**Utilize commercial tools designed for detecting these problems**

1. Verify all 3 classes of CDC problems
   1. Structural Verification
   2. Protocols Verification
   3. Reconvergance Verification
2. Use reports to aid manual reviews
3. Use CDC tools to support ROBUSTNESS

In Conclusion …

- **Every multi-clock design is subject to metastability**
- **Traditional verification methodologies CANNOT assure robustness**

- **To properly mitigate the dangers of CDC, we strongly recommend a solution that… :**
  - Supports Manual Reviews
  - *Automatically* reports all sources of CDC problems
  - Has a proven CDC verification methodology & customer success

Mentor Graphics®

www.mentor.com