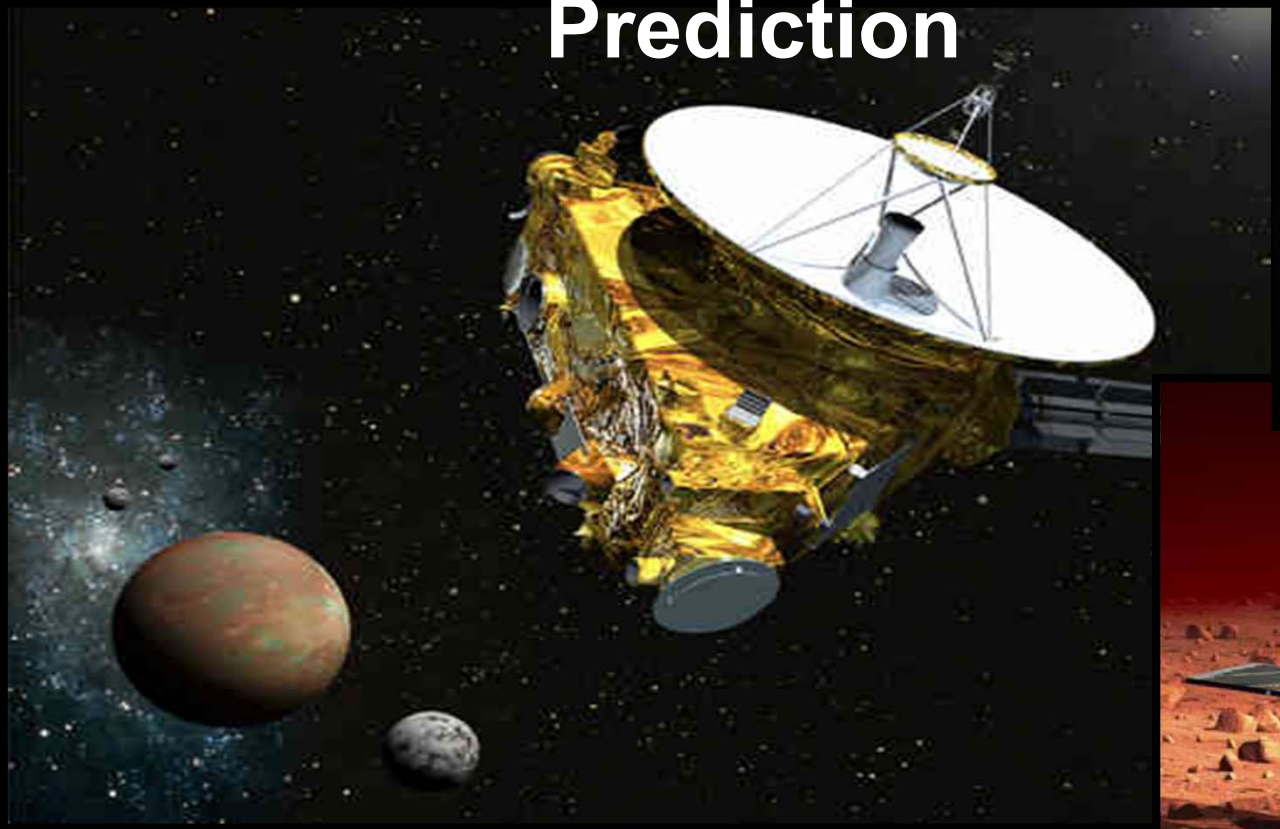
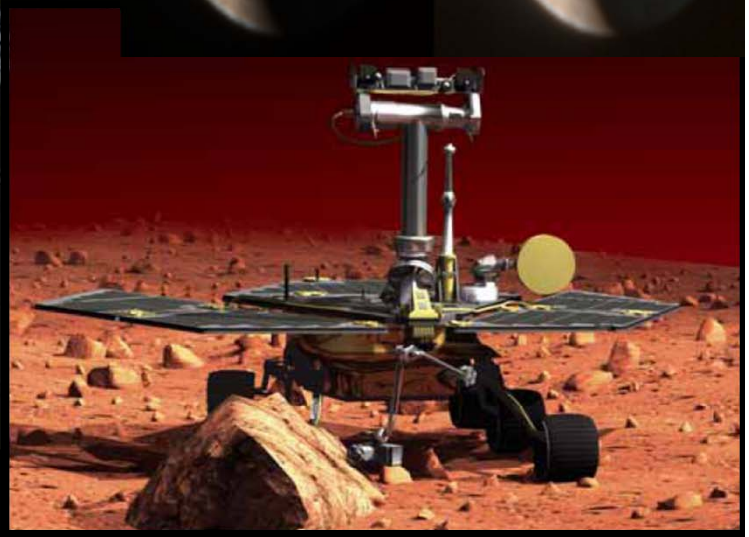
An Overview of the NASA Goddard Methodology for FPGA Radiation Testing and Soft Error Rate (SER) Prediction





Melanie Berg, MEI Technologies in support of NASA/GSFC

To be presented by Melanie Berg at the Single-Event Effects Symposium and Military and Aerospace Programmable Logic Devices (SEE-MAPLD),
La Jolla, CA, April 9-12, 2013, and published on http://nepp.nasa.gov/

Acknowledgements

- *Some of this work has been sponsored by the NASA Electronic Parts and Packaging (NEPP) Program and the Defense Threat Reduction Agency (DTRA)*
- *Thanks is given to the NASA Goddard Radiation Effects and Analysis Group (REAG) for their technical assistance and support. REAG is led by Kenneth LaBel and Jonathan Pellish*

Contact Information:

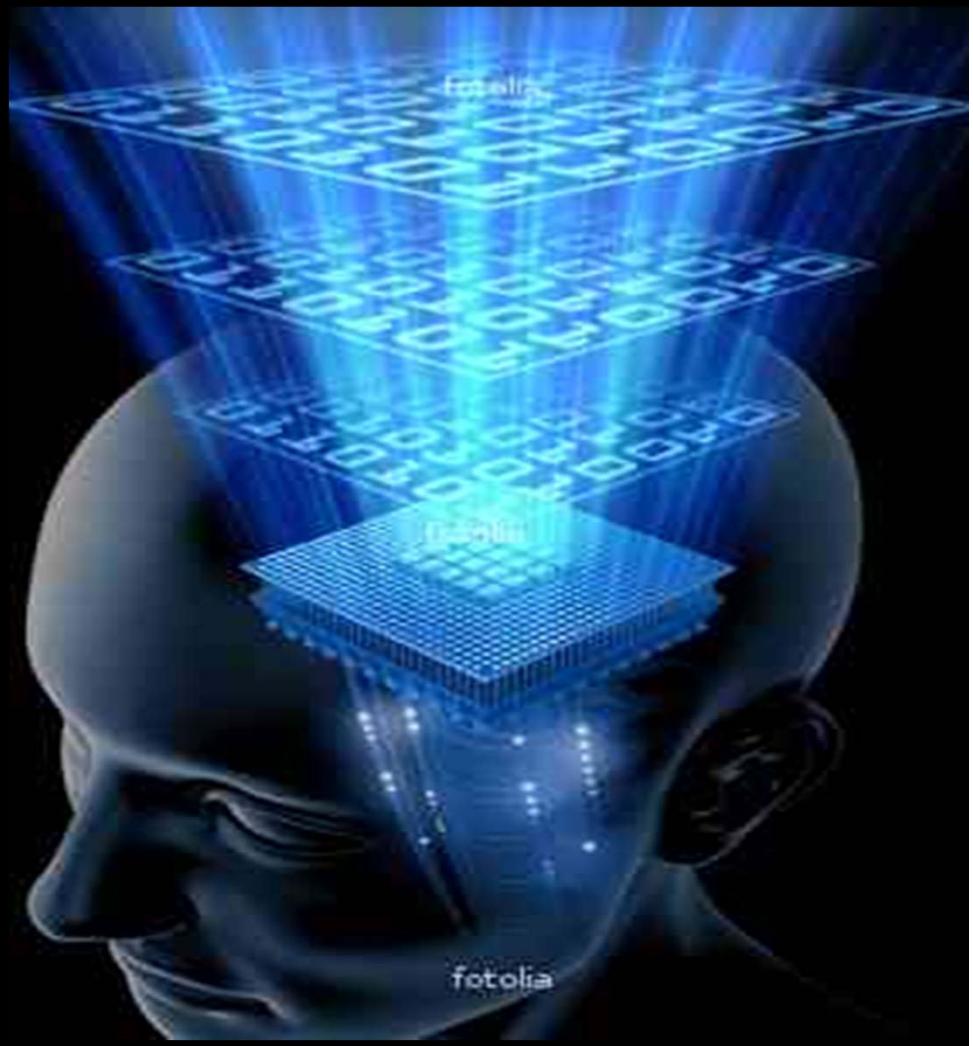
Melanie Berg: NASA Goddard REAG FPGA Principal Investigator:

Melanie.D.Berg@NASA.GOV

A Couple of Things to Think About before We Get Started

FPGAs are complex devices that require a complex process for evaluation

Our biggest challenge is understanding how SEU test data can be extrapolated to complex designs



Overview

- **Discuss Single Event Upset (SEU) accelerated testing considerations regarding Field Programmable Gate Array (FPGA) devices**
- **Terminology:**
  - **Single Event Transient (SET): current spike due to single particle ionization.**
  - **Single Event Upset (SEU): transient is caught by a memory element. Causes an incorrect state. SETs are categorized under SEUs**
  - **Single Event Functional Interrupt (SEFI): upset disrupts function**
  - **SEU cross-section (σ_{SEU}):** $\dfrac{number\ SEU\ upsets}{Particle\ Fluence}$

Summary of SEU Test Preparation and Test Implementation

Evaluate the Device Under Test (DUT) FPGA Fabric

- A comprehensive study of the FPGA's fabric must be performed prior to testing.
- The evaluation involves understanding the FPGA's elements and how designs are mapped into its elements.
  - Device speed
  - Device Combinatorial logic (CL) and flip-flops (DFFS)
  - Device global routing
  - Hardening circuits
  - Hidden logic
- From this information, specific radiation tests and test structures can be developed to target the DUT's various components.

Consider the Goal of SEU Testing Prior to Creating the Test Plan

- The intention of testing will drive the test structures implemented in the DUT.
- The following are examples of three goals that will require different test plans:
  - Evaluation of DFF susceptibility: Are the DFFs hardened?
  - Extrapolation of σ_{SEU} data to calculate error rates for real designs
  - Analysis of configuration memory susceptibility

Which elements are most susceptible to SEUs?

SEU Testing Goal Example(1): Evaluation of DFF susceptibility

- Visibility of DFF upsets is key; i.e., we do not want to mask or miss upsets
- In order to optimize visibility of FF susceptibility, test structures should be selected that have minimal data-path logic-masking
  - e.g., shift registers are optimal for DFF SEU testing
  - Complex circuits will use gates that can mask the propagation of upsets and can provide lower σ_{SEU}s
- Test vehicles must be fast enough to capture data output. Otherwise SEUs can be missed
- Test vehicles must be robust enough to capture bursts of errors (i.e., consecutive upsets). Otherwise error signatures cannot be differentiated.

SEU Testing Goal Example(2): Extrapolation of σ_{SEU} Data to Estimate Error Rates for real designs

- Characterizing SEU effects for real designs is a different process than studying individual elements such as FFs.
- σ_{SEU} Data is used to calculate upset rates for critical missions.
- The question is, what σ_{SEU} Data should be used?
  - Usually the mission's final design is not tested in the radiation beam.
  - Subsequently, test structures are developed and then radiation tested to evaluate trends.
  - The trends are then used to facilitate the extrapolation of σ_{SEU} data to calculate error rates

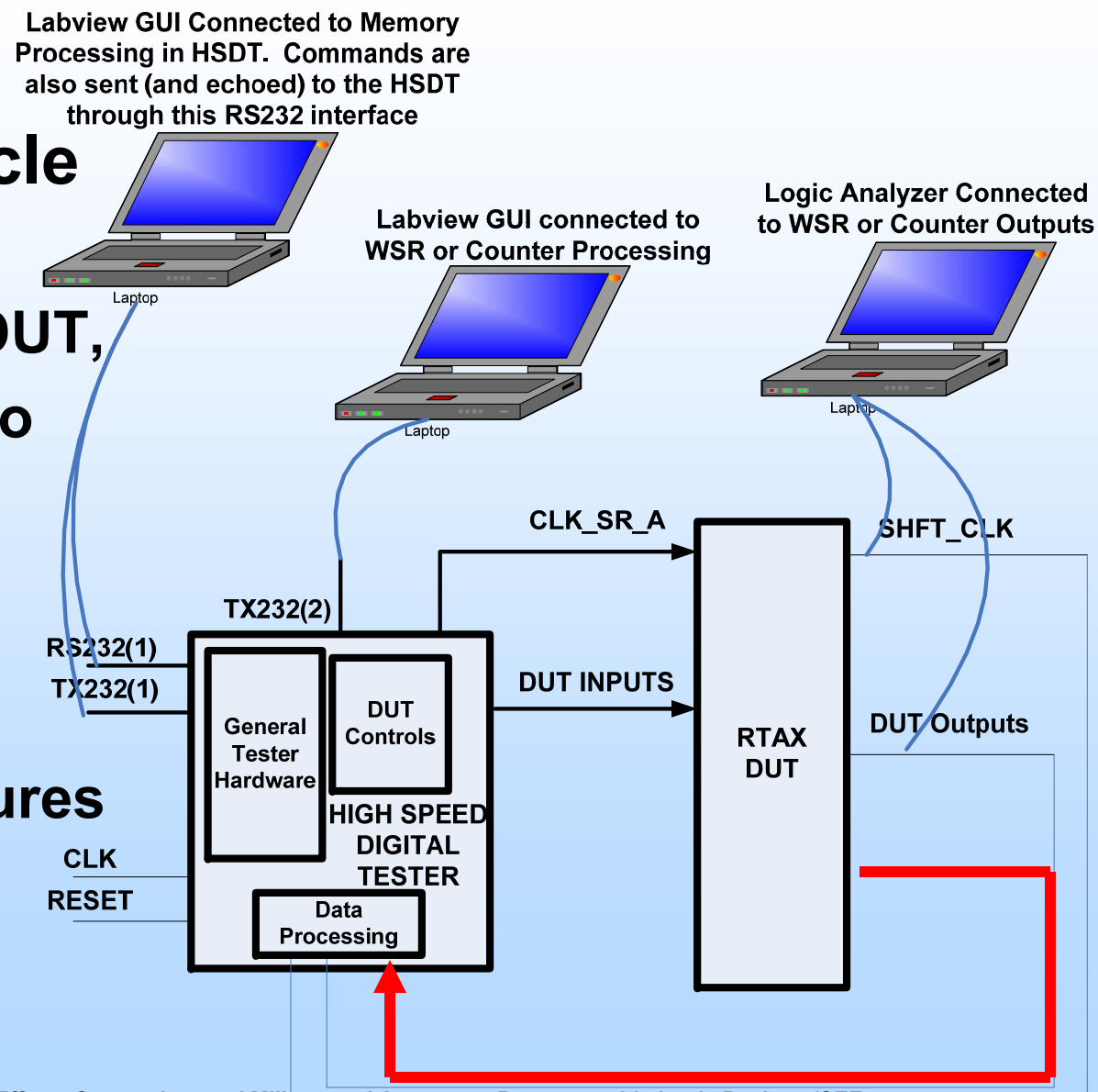
Creation of DUT test structures:

- **Based on the goal of testing, the following are considerations for FPGA test structures:**
  - **Utilize similar design topologies that utilize the same basic elements as real designs, i.e., synchronous design**
  - **Repetition of design structures to increase statistics**
  - **Functional visibility such that all upsets can be identified and recorded,**
  - **A state space that can be traversed within minutes; i.e., a traversable state-space.**

State Space Traversal Example: 32-bit counter running at 50MHz will take 86 seconds to traverse its entire state space… are you testing long enough?

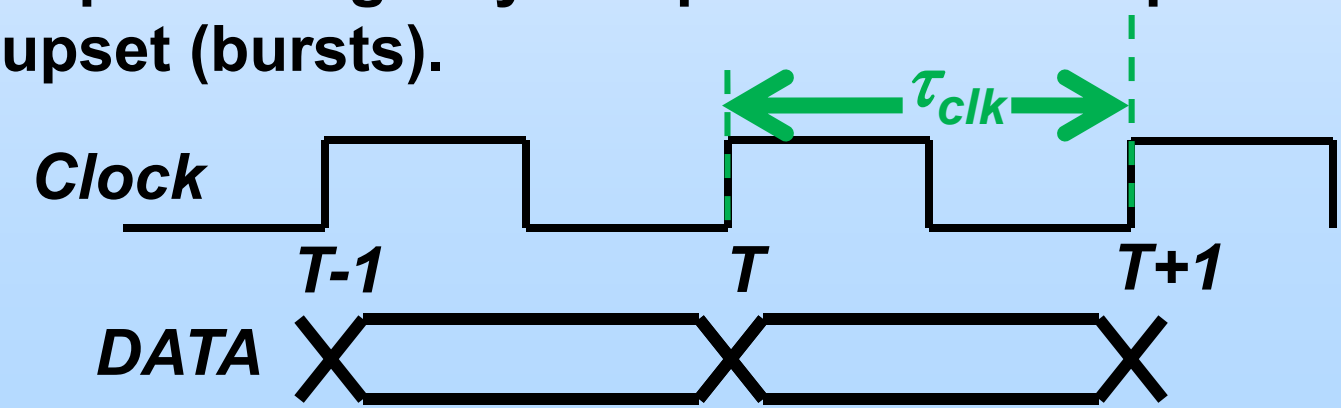
Test Vehicle Development(1):

- General test vehicle requirements:
  - Connects to the DUT,
  - Provides stimuli to the DUT,
  - Monitors the DUT during radiation testing, and
  - Records DUT failures during radiation testing.

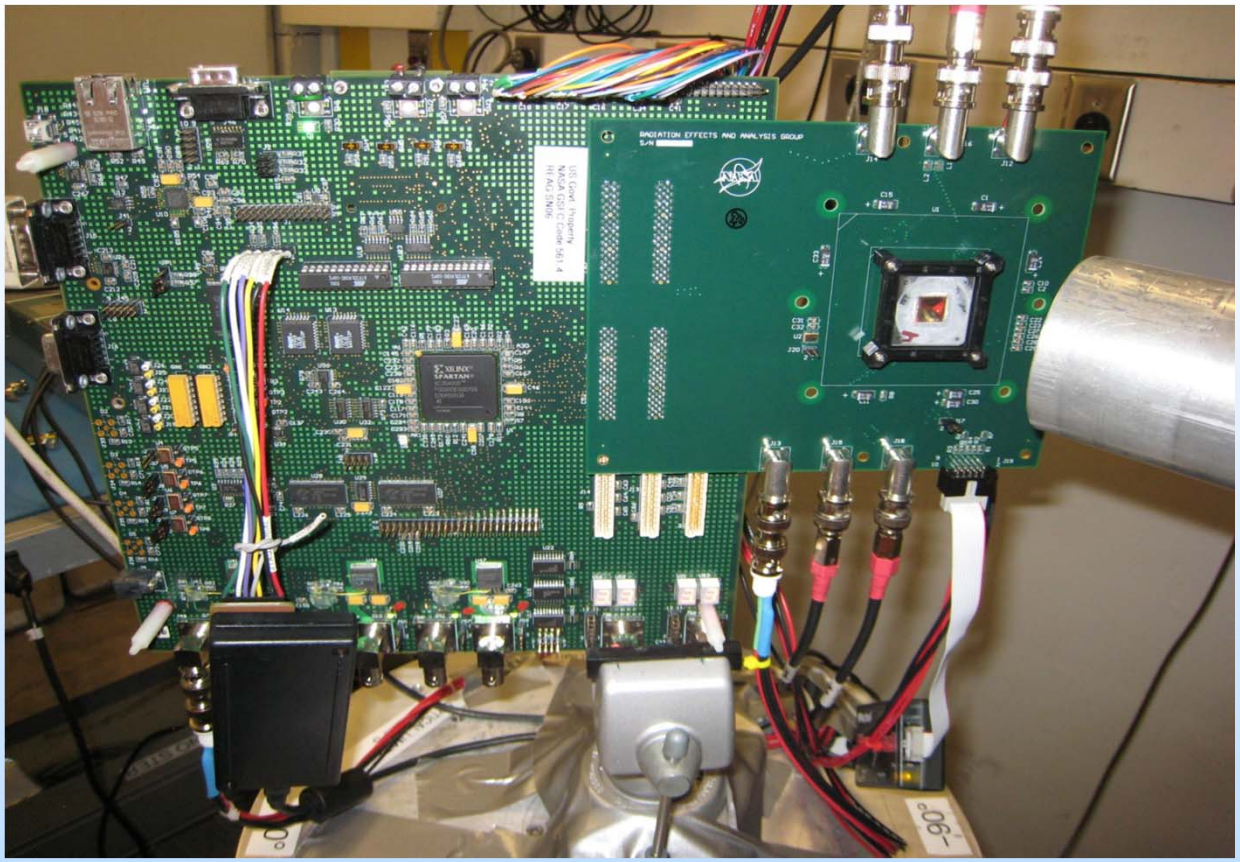


Test Vehicle Development(2):

- The test vehicle should be robust such that:
  - DUT stimuli (e.g., data patterns and operational frequency) can be varied.
  - It can reliably capture DUT data:
    - Meets electrical design requirements (e.g., Signal integrity, proper signal capture)
    - It is fast enough to handle DUT upset events in an accelerated radiation environment.
    - It can capture single cycle upsets and multiple cycle upset (bursts).



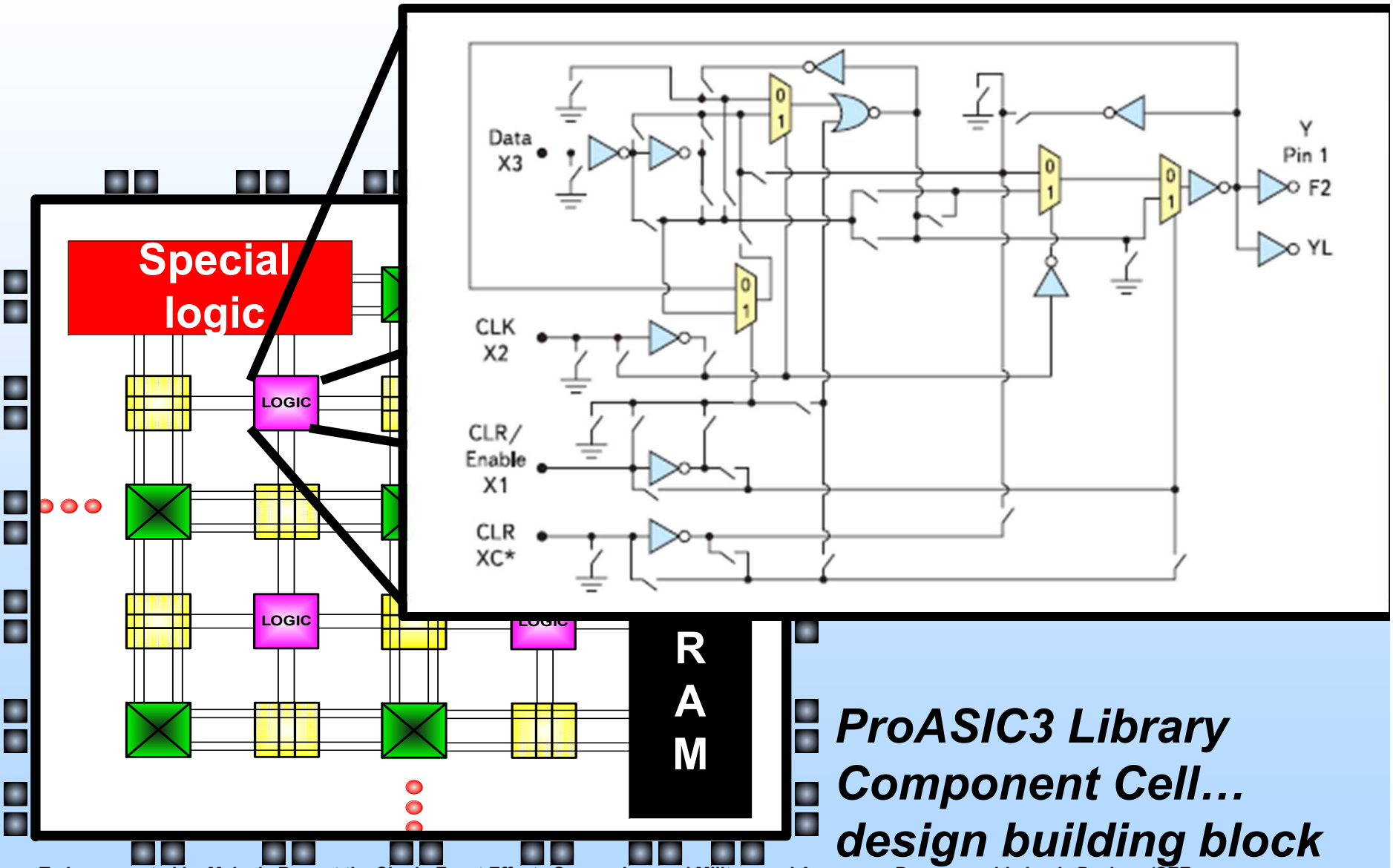
Test Vehicle Development(3)



- **Create your own test board: Expensive and challenging**
- **Buy off the shelf – issues: Flexibility, I/O, and physical interfaces**

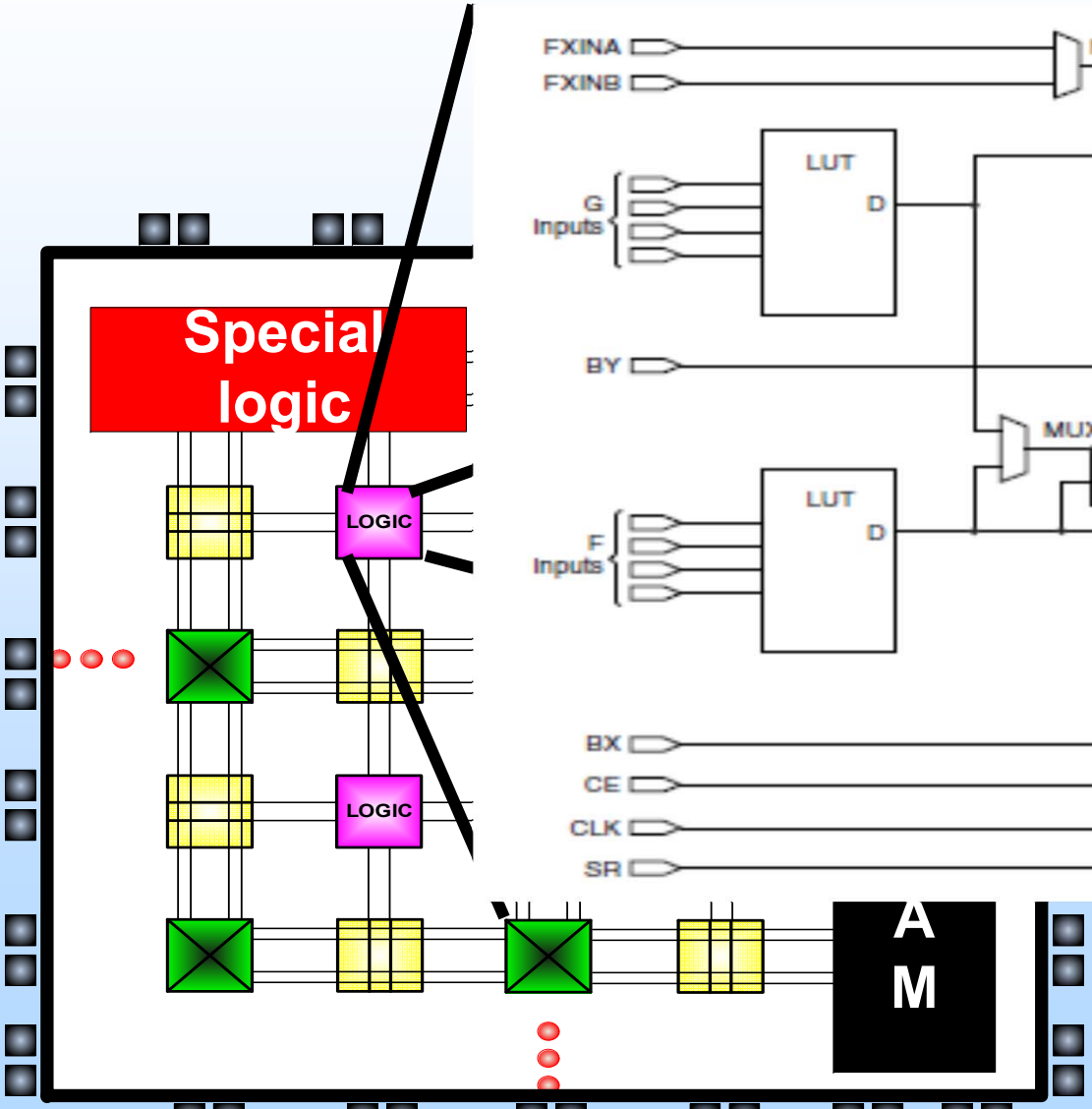
FPGA Fabric and General Synchronous Design Concepts: Beware – Different FPGAs will have different types of building blocks and will hence have varying susceptibility

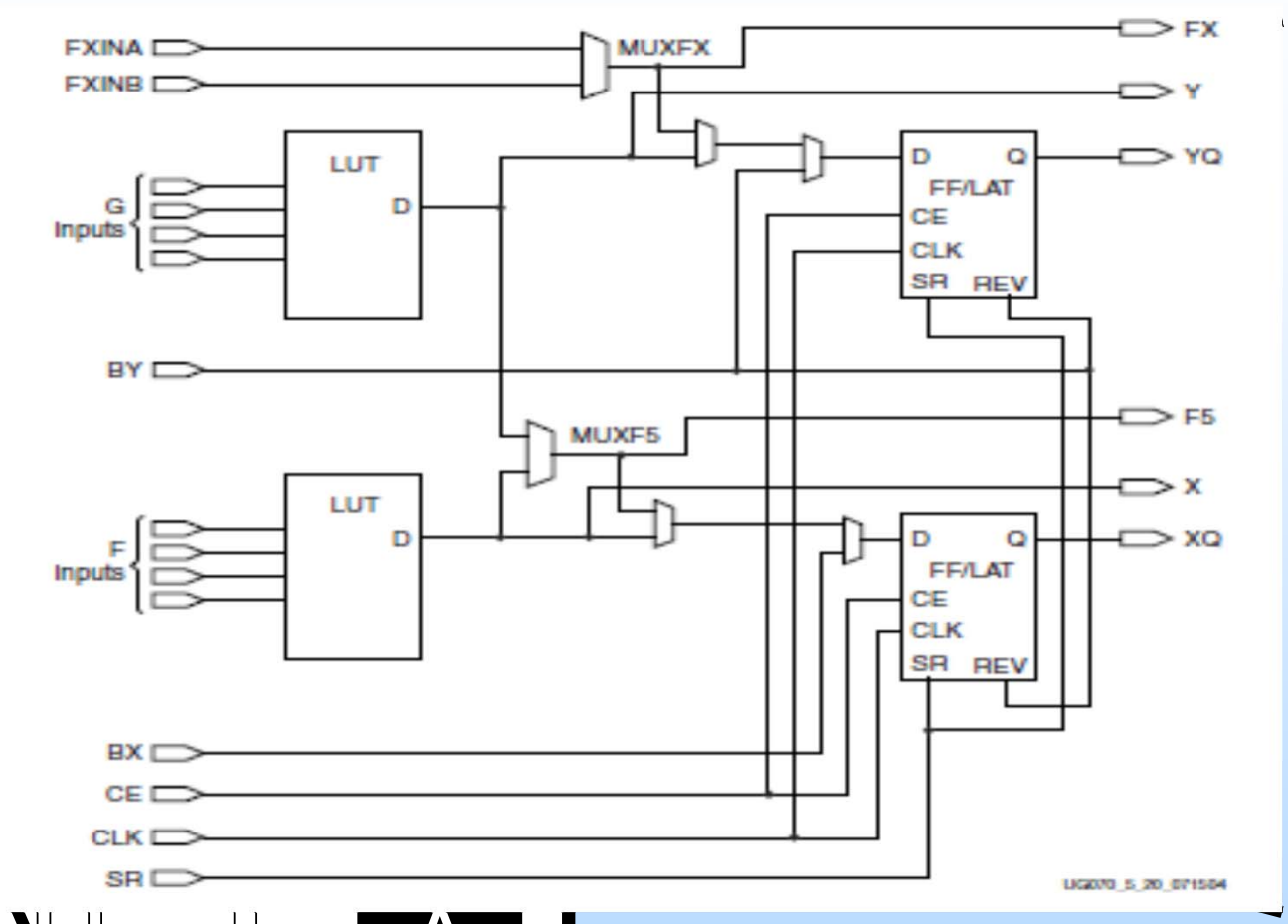
A Closer Look at an FPGA Logic Cell: Microsemi ProASIC3



ProASIC3 Library Component Cell… design building block

Background: Synchronous Design Building Blocks: FPGA Logic Cells: Xilinx Virtex-4





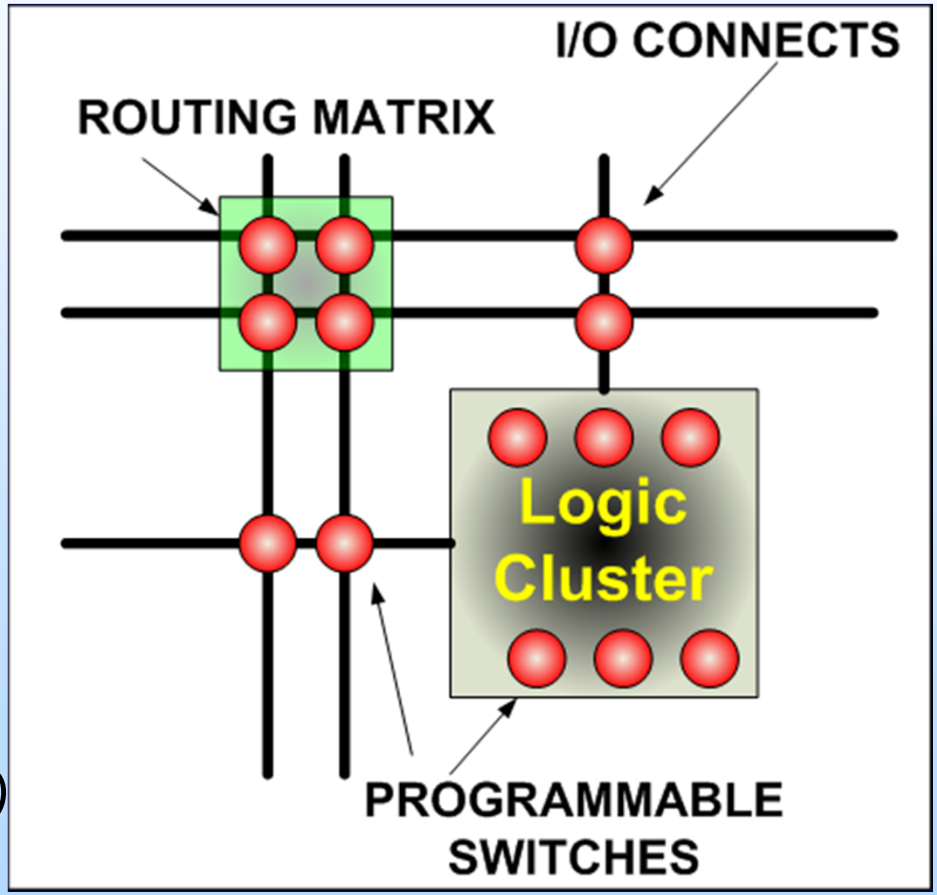
Xilinx Library Cell

Look-up-table (LUT)

FPGA Configuration

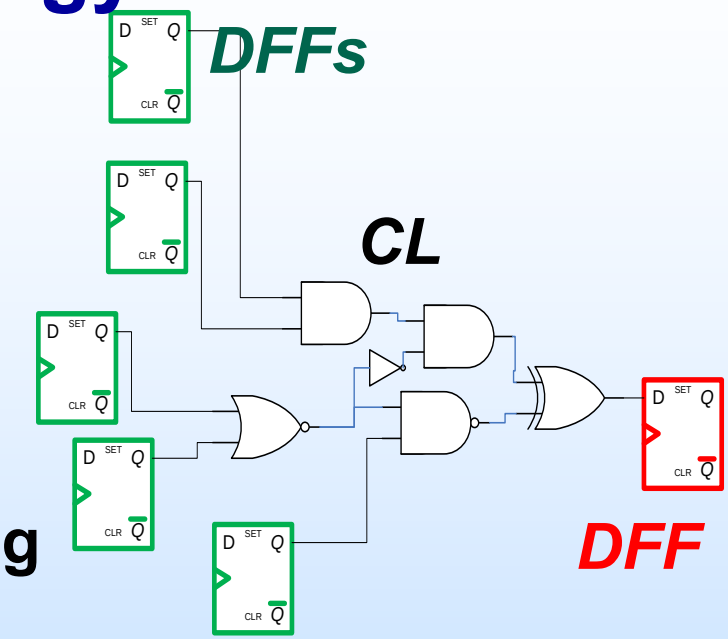
HDL → ***FPGA MAPPING*** → **Configuration**

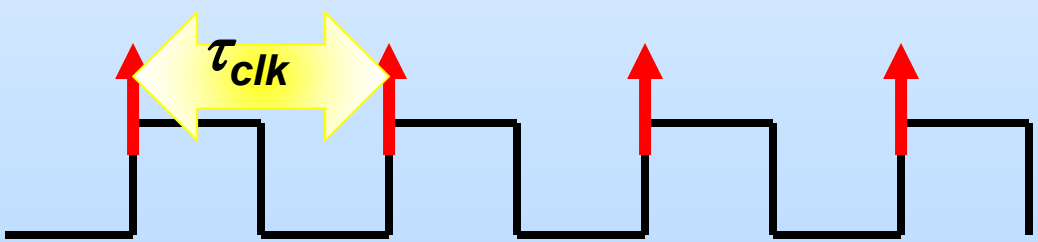
- ***Configuration Defines:*** *Arrangement of pre-existing logic via programmable switches*
  - *Functionality (logic cluster)*
  - *Connectivity (routes)*
- ***Programming Switch Types:***
  - ***Antifuse:*** *One time Programmable (OTP)*
  - ***SRAM:*** *Reprogrammable (RP)*
  - ***Flash:*** *Reprogrammable (RP)*



Most FPGA Designs Follow Synchronous Design Methodology

- **Synchronous design components:**
  - **Edge Triggered Flip-Flops (DFFs)**
  - **Clocks and resets (global routes)**
  - **Combinatorial Logic (CL)**
- **All DFFs are connected to a clock.**
- **DFFs sample their input at the rising edge of clock**





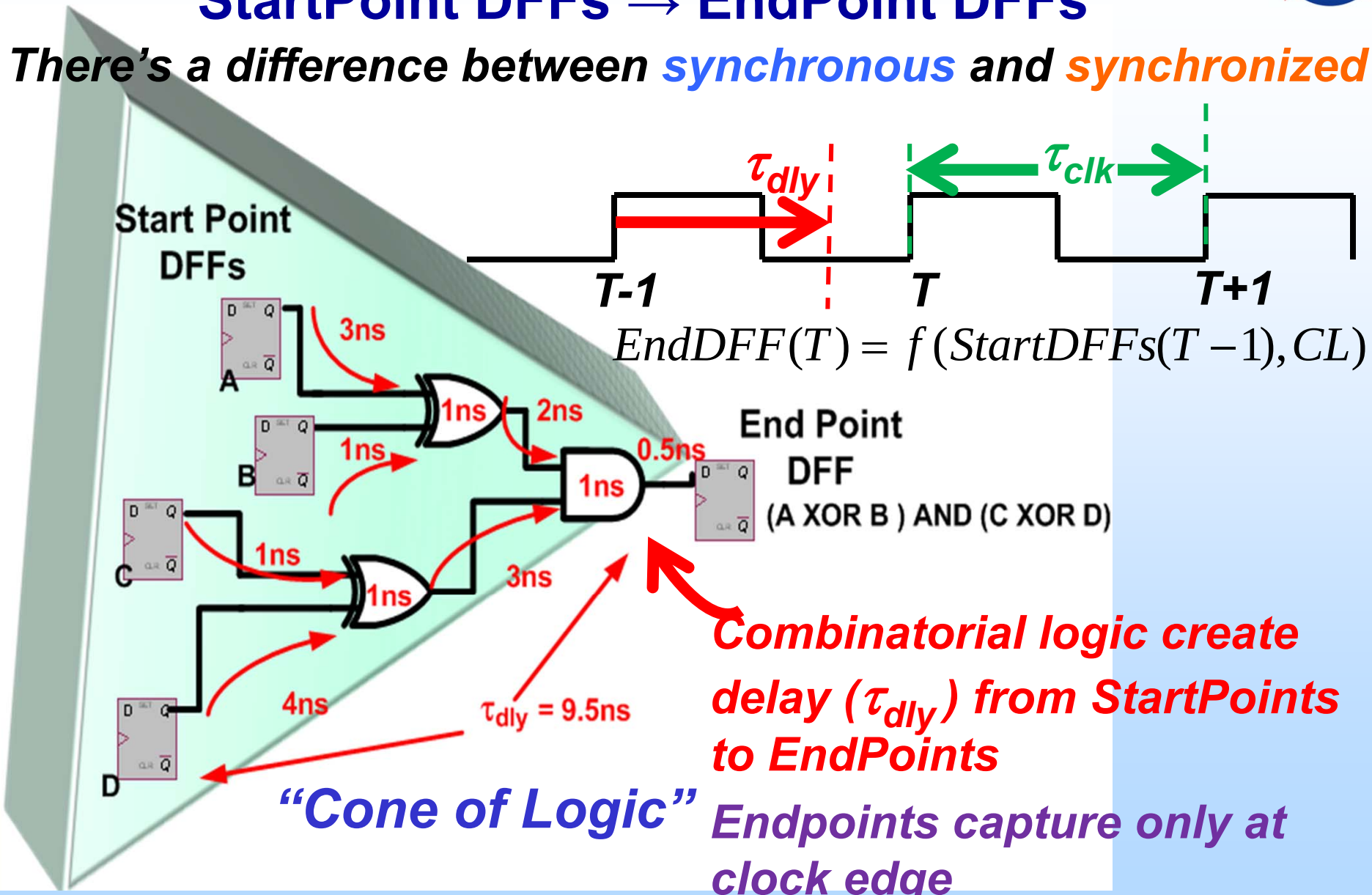
$$\text{Clock Period } \tau_{clk} = \frac{1}{f_S}$$

Frequency

- **CL compute between clock edges**

Synchronous System Data Paths: StartPoint DFFs → EndPoint DFFs

There's a difference between synchronous and synchronized



Combinatorial logic create delay (τ_{dly}) from StartPoints to EndPoints

Endpoints capture only at clock edge

SEUs in FPGA Devices

Categorization of FPGA SEUs as Defined by NASA Goddard REAG:

SEU Cross section: σ_{SEU} = #upsets/(#particles /cm^2)

$$P(fs)_{error} \propto P_{Configuration} + P(fs)_{functionalLogic} + P_{SEFI}$$

Design σ_{SEU}

Configuration σ_{SEU}

Functional logic σ_{SEU}

SEFI σ_{SEU}

Calculated per LET

Sequential and Combinatorial logic (CL) in data path

Global Routes and Hidden Logic

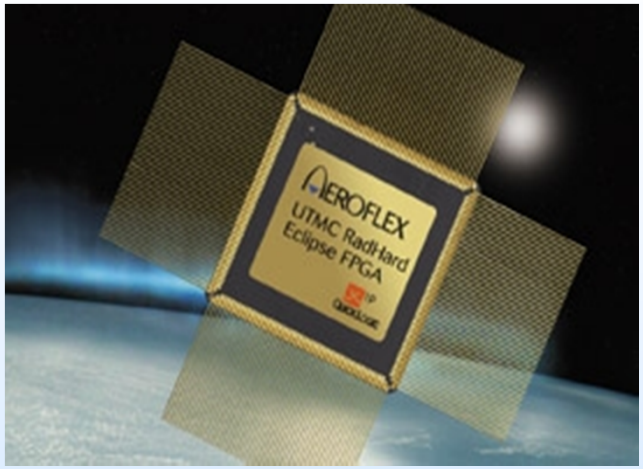
SEU Testing is required in order to characterize σ_{SEU}s for each of FPGA categories. We will focus on test structures used to analyze the functional logic data path

Configuration

$$P(fs)_{error} \propto P_{Configuration} + P_{functionalLogic} + P_{SEFI}$$

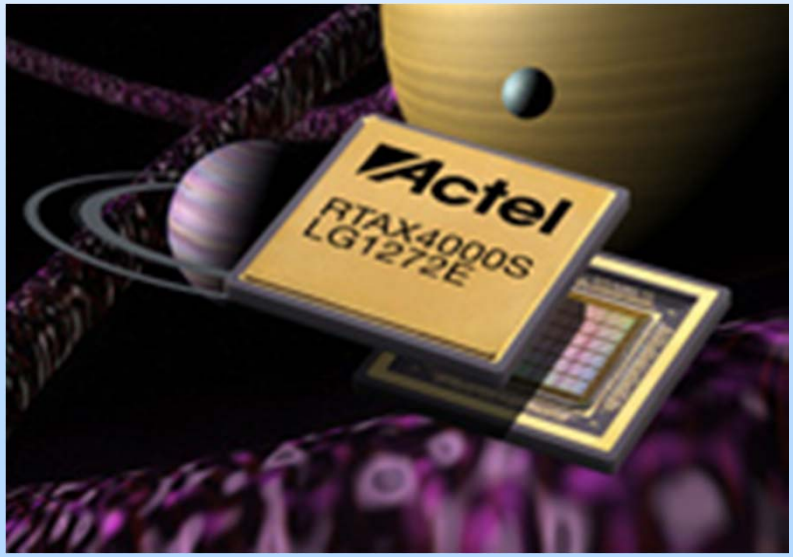
FPGA Categorized by Configuration Types

Antifuse



SRAM

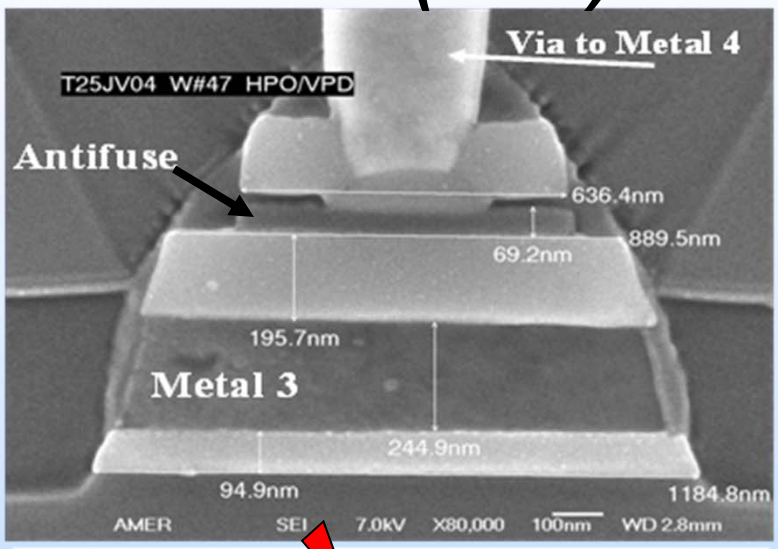
Antifuse



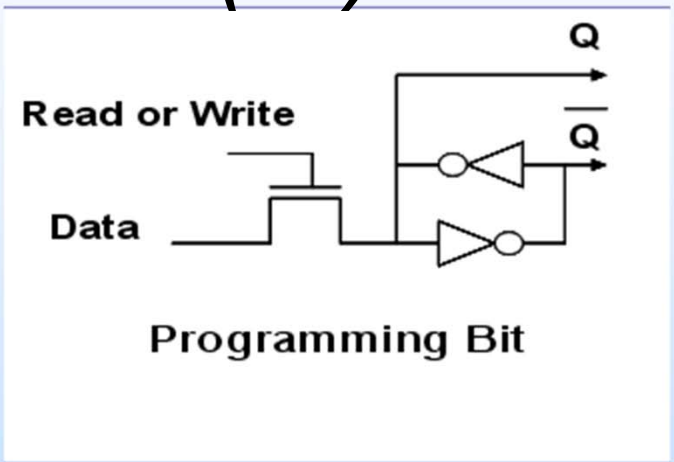
Flash

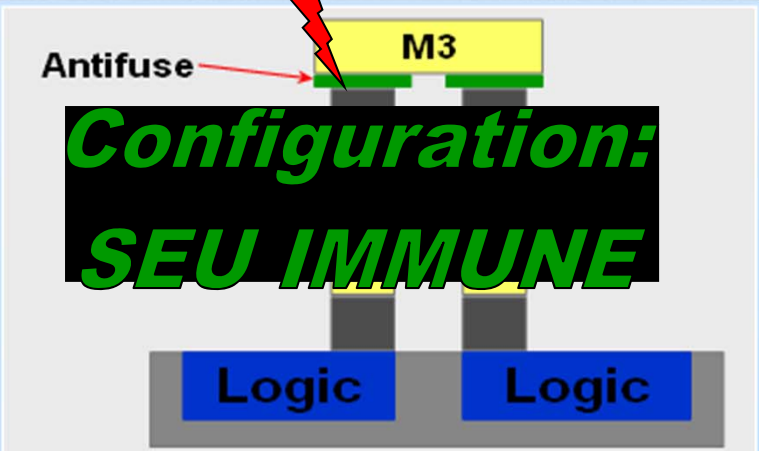
Programmable Switch Implementation and SEU Susceptibility

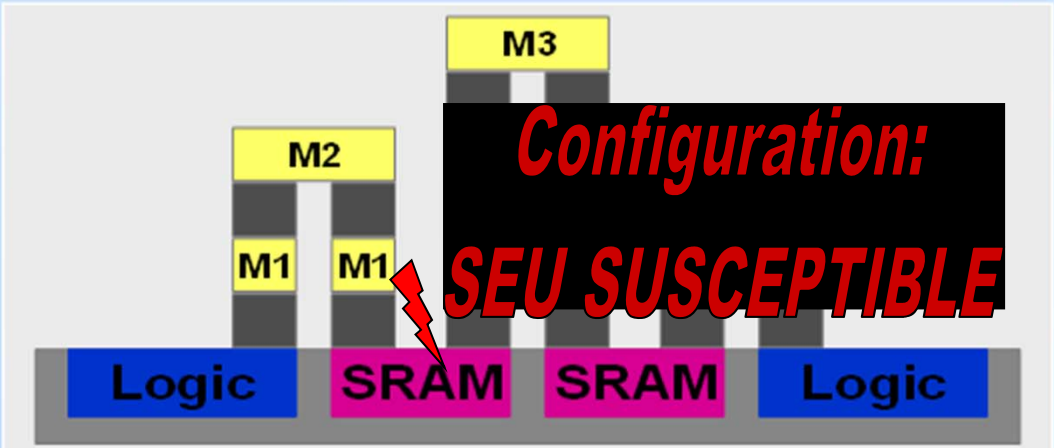
ANTIFUSE (OTP)



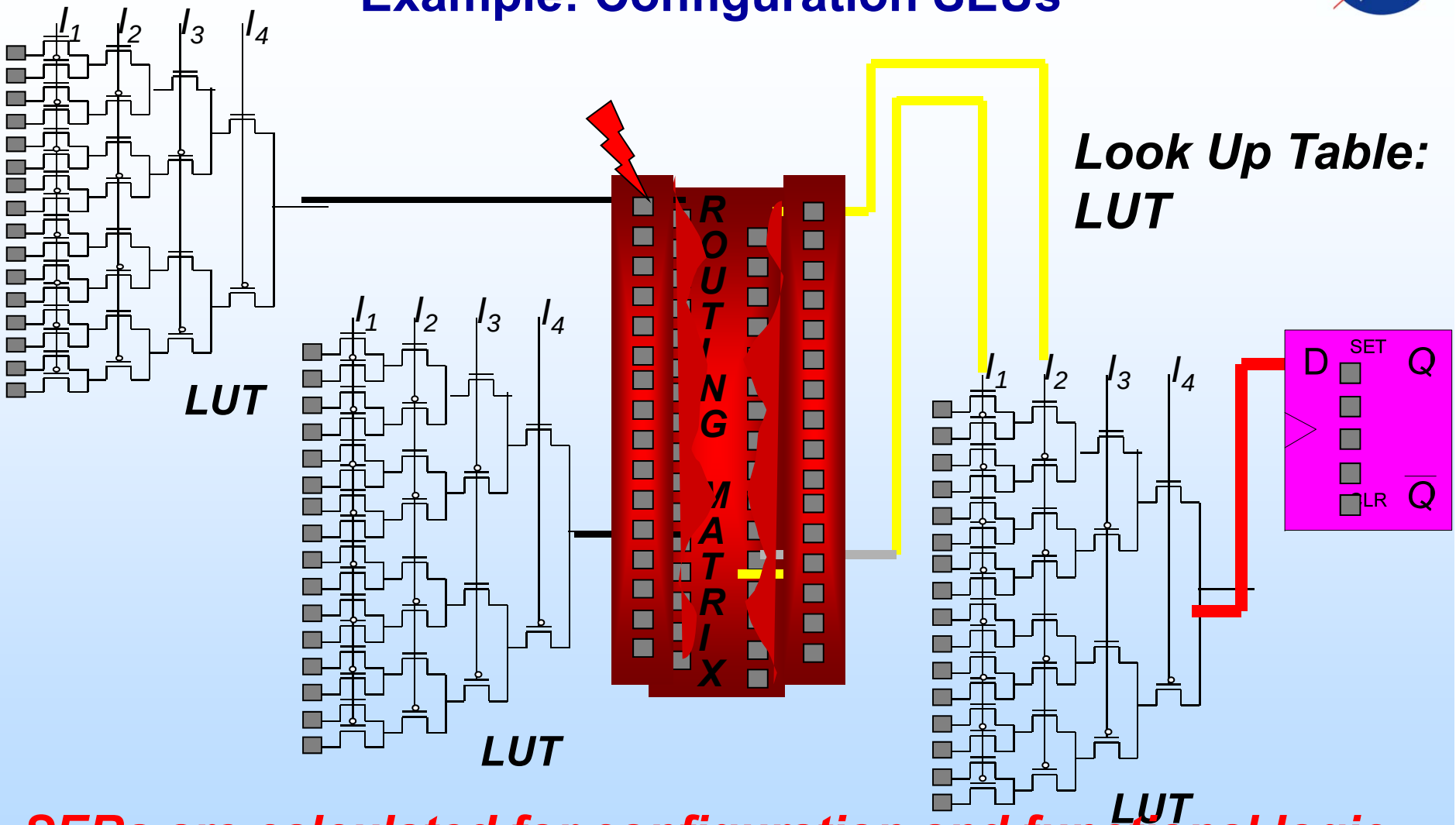
SRAM (RP)







Example: Configuration SEUs



SERs are calculated for configuration and functional logic. This is the case for memory based configuration. Antifuse does not have the same susceptibility issues

To be presented by Melanie Berg at the Single-Event Effects Symposium and Military and Aerospace Programmable Logic Devices (SEE-MAPLD),
La Jolla, CA, April 9-13, 2013, and published on http://nepp.nasa.gov/

How Do We Test Configuration

- **Antifuse or Flash Configuration:**
  - Normal testing of functional logic data paths
  - Check for stuck-at-faults
- **SRAM-based Configuration:**
  - Static Testing
  - Load configuration memory
  - Place the device in beam-line
  - Turn beam off
  - Read back the configuration and count the number of upsets

Configuration SEU Test Results and the REAG FPGA SEU Model

$$P(fs)_{error} \propto P_{Configuration} + P(fs)_{functional\ Logic} + P_{SEFI}$$

Configuration	REAG Model
Antifuse	$P(fs)_{error} \propto P(fs)_{functionalLogic} + P_{SEFI}$
SRAM **(non-mitigated)**	$P(fs)_{error} \propto P_{Configuration}$
Flash	$P(fs)_{error} \propto P(fs)_{functionalLogic} + P_{SEFI}$
Hardened SRAM	$P(fs)_{error} \propto \cancel{P_{Configuration}} + P(fs)_{functionalLogic} + P_{SEFI}$

Functional Data Path SEU Modeling

It's important to understand the factors that exist in the functional data path that can:

Mask upsets or

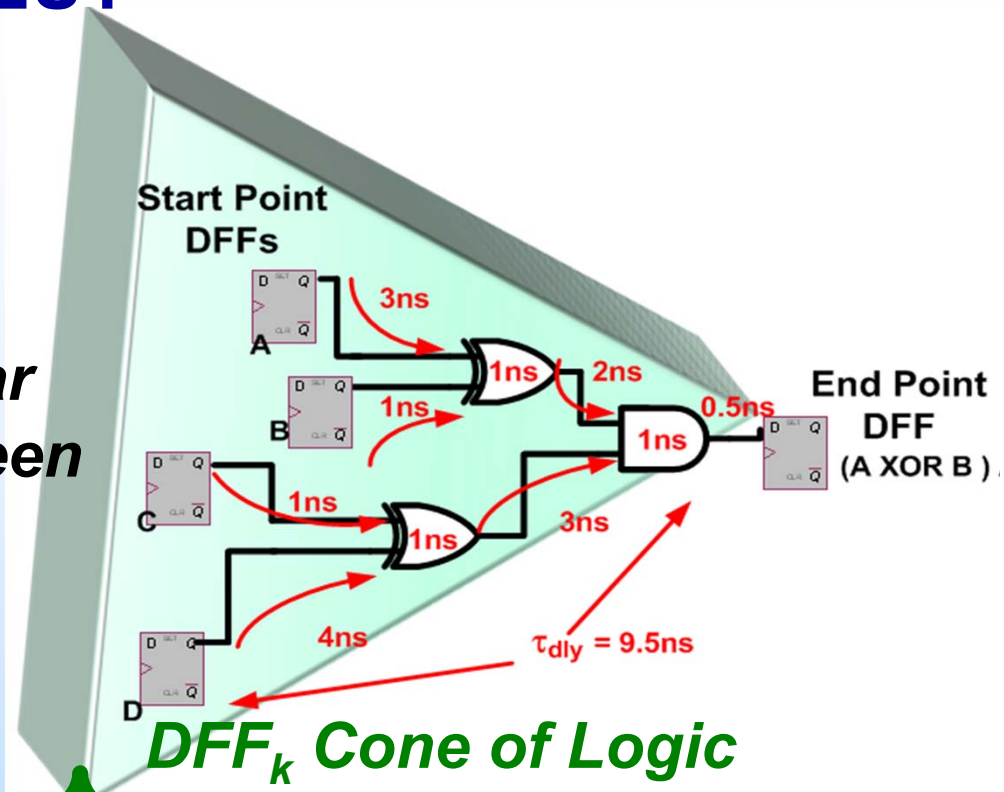
Cause additional upsets (domino effect, signal integrity)

$$P(fs)_{error} \propto P_{Configuration} + P(fs)_{functionalLogic} + P_{SEFI}$$

How can a DFF Contain an Incorrect State from an SEU?

Every DFF has a unique cone of logic

We make a clear distinction between DFF SEUs



DFF_k Cone of Logic

$\forall_{DFF}$

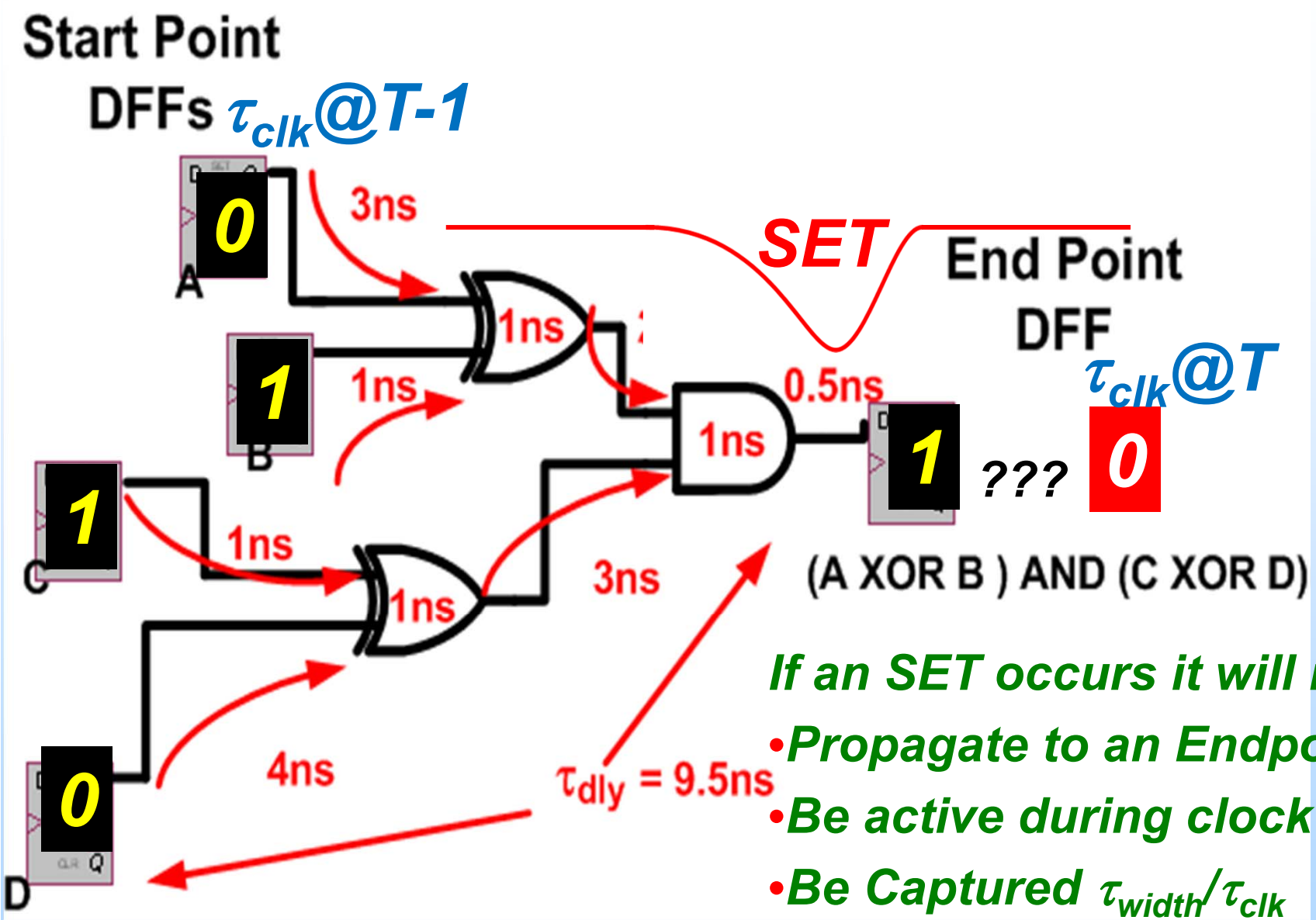
EndPoint DFF SEUs + StartPoint DFF SEUs + CL SETs

DFF upsets that occur at the clock edge

DFF upsets that occur between clock edges and are captured by EndPoints

Single Event Transients captured by EndPoints

Synchronous System: CL SET Capture

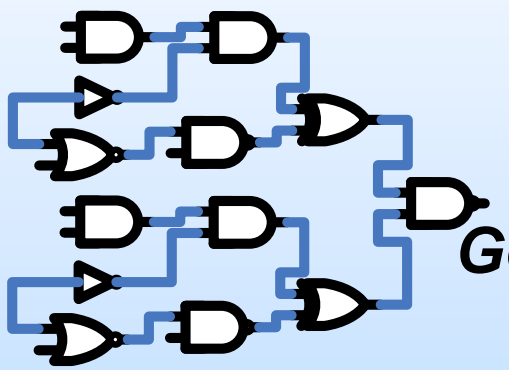


If an SET occurs it will need to

- *Propagate to an Endpoint*
- *Be active during clock edge*
- *Be Captured τ_{width}/τ_{clk}*

Details of Combinatorial Logic SET Capture

$$\exists_{DFF}\left(\sum_{i=1}^{\#Combinatorialcells}(P_{gen(i)}P_{prop(i)}P_{logic}\tau_{width(i)}fs)\right)$$



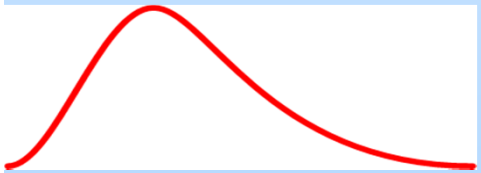
CL

Generation

Propagation

Logic Masking

Capture

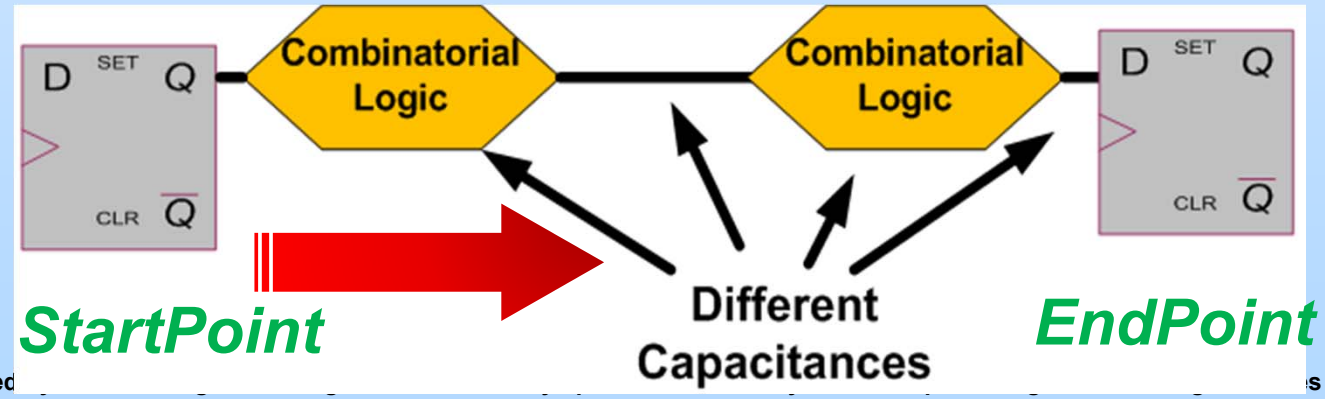


Double Sided

- SET Generation (P_{gen}) occurs between clock edges
- SET Capture occurs at a clock edge

SET Propagation to an EndPoint DFF: P_{prop}

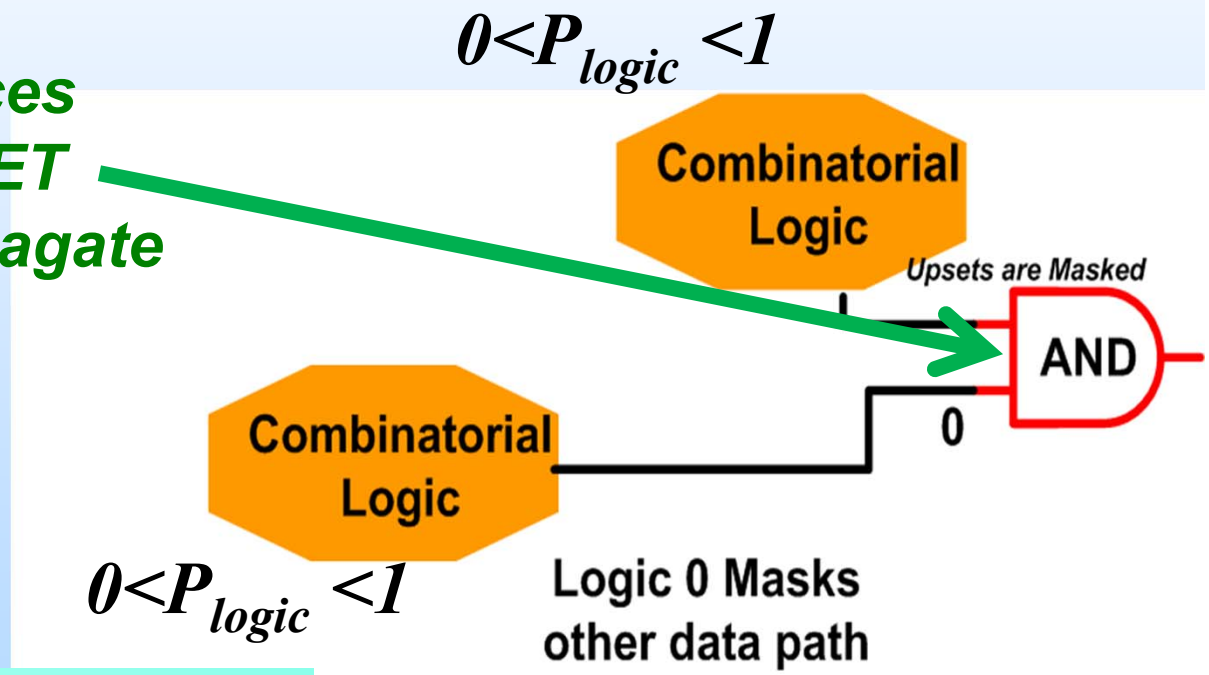
- In order for the data path SET to become an upset, it must propagate and be captured by its Endpoint DFF
- P_{prop} only pertains to electrical medium (capacitance of path… combinatorial logic and routing)
  - Capacitive SET amplitude reshaping
  - Capacitive SET width reshaping
- Small SETs or paths with high capacitance have low P_{prop}
- P_{prop} contributes to the non-linearity of $P(fs)_{SET \rightarrow SEU}$ because of the variation in path capacitance



SET Logic Masking: P_{logic}

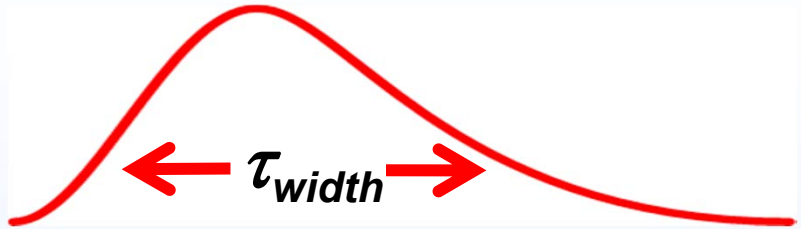
- ***P_{logic}: Probability that a SET can logically propagate through a cone of logic. Based on state of the combinatorial logic gates and their potential masking.***

"AND" gate reduces probability that SET will logically propagate



Determining P_{logic} for a complex system can be very difficult

SET Capture at Destination DFF



The transient width (τ_{width}) will be a fraction of the clock period (τ_{clk}) for a synchronous design in a CMOS process.

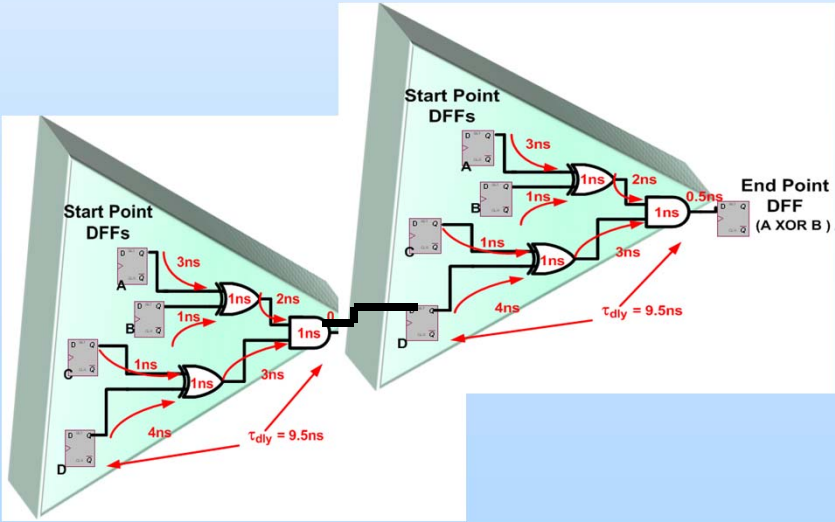
$$P(\tau_{clk})_{SET \rightarrow SEU} \propto \frac{\tau_{width}}{\tau_{clk}}$$

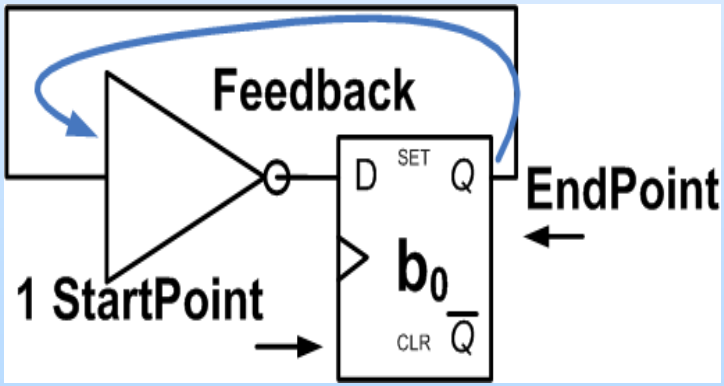
$$P(fs)_{SET \rightarrow SEU} \propto \tau_{width} fs$$

Probability of capture is proportional to the width of the transient as seen from the destination DFF

DFF Analysis from the Perspective of StartPoints and EndPoints

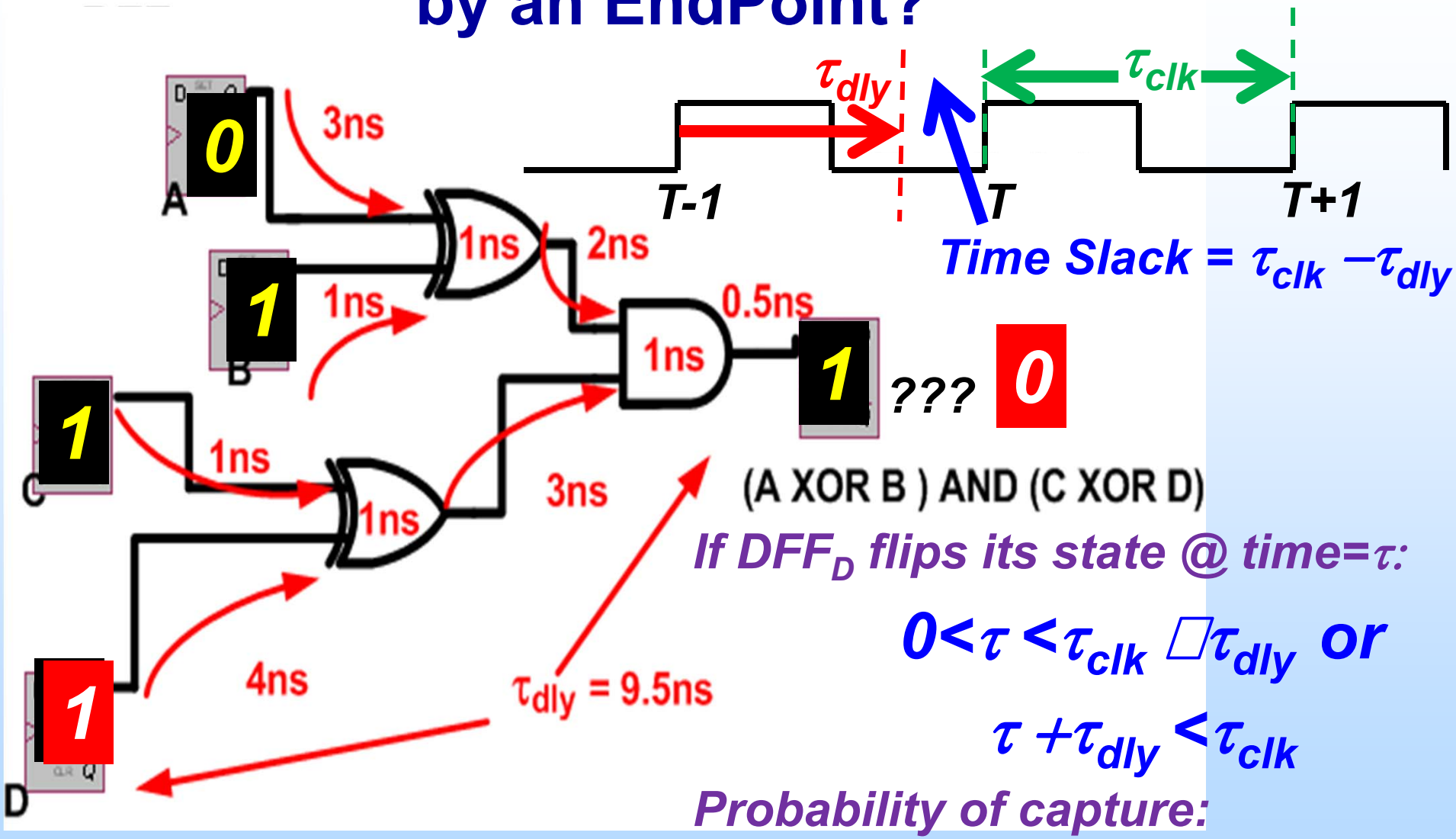
- All DFFs are analyzed as both StartPoints and EndPoints
- DFF is only analyzed as an EndPoint at the instance of a clock edge – i.e.; during capture
- DFF is analyzed as a StartPoint at all other moments; i.e., in between clock edges





EndPoint = b_0

How Does a StartPoint SEU get Captured by an EndPoint?



If DFF_D flips its state @ time=τ:

$0 < \tau < \tau_{clk} - \tau_{dly}$ or

$\tau + \tau_{dly} < \tau_{clk}$

Probability of capture:

$1 - (\tau_{dly}/\tau_{clk}) = 1 - \tau_{dly} fs$

StartPoint SEUs and Their Capture

- **StartPoint is upset in between clock edges ($\beta P(fs)_{DFFSEU}$)**
- **The upset must have enough time to propagate to the EndPoint ($\tau < \tau_{clk} - \tau_{dly}$)**
- **The upset cannot get logically masked (P_{logic})**

Probability that a StartPoint will upset

$$\underbrace{\sum_{j=1}^{\#StartPoint\ DFFs} \left(\beta P(fs)_{DFFSEU(j)} (1 - \tau_{dly(j)} fs)) * P_{logic(j)} \right)}_{\textit{Probability that any of the StartPoint SEUs will get caught by an EndPoint}}$$

Synchronous Design: NASA REAG FPGA Data Path Susceptibility Model… per LET

$$\sum_{k=1}^{\#EndPoint\ DFFs} P_{logic(k)} * \left(\alpha P(fs)_{DFFSEU(k)} + \sum_{j=1}^{\#StartPoint\ DFFs} \left(\beta P(fs)_{DFFSEU(J)} (1 - \tau_{dly(j)} fs)) * P_{logic(j)} \right) + \sum_{i=1}^{\#CL} (P_{gen_{(i)}} * P_{prop_{(i)}} * P_{logic_{(i)}} * \tau_{width_{(i)}} fs) \right)$$

EndPoint Logic Masking

EndPoint

StartPoints

Combinatorial Logic

$P_{logic(k)}$ ➔ ***EndPoint Logic Masking***

$P_{logic(j)}$ ➔ ***StartPoint Logic Masking***

$P_{logic(i)}$ ➔ ***Combinatorial Logic Masking***

$\tau_{width(i)}$ ➔ ***SET width***

$P_{gen(i)}$ ➔ ***SET generation***

$P_{prop(i)}$ ➔ ***SET propagation***

$P_{logic(i)}$ ➔ ***SET logic masking***

Trends and the NASA REAG FPGA Data Path Susceptibility Model

$$\sum_{k=1}^{\#EndPoint\ DFFs} P_{logic(k)} * \left(\alpha P(fs)_{DFFSEU(k)} + \sum_{j=1}^{\#StartPoint\ DFFs} \left(\beta P(fs)_{DFFSEU(J)} (1 - \tau_{dly(j)} fs)) * P_{logic(j)} \right) + \sum_{i=1}^{\#CL} (P_{gen_{(i)}} * P_{prop_{(i)}} * P_{logic_{(i)}} * \tau_{width_{(i)}} fs) \right)$$

EndPoint Logic Masking

EndPoint

StartPoints

CL

StartPoints and combinatorial logic (CL) need to be captured thus have data path de-rating factors

Component Contribution to σ_{SEU} across Frequency and Gate Count

	Frequency	# of Gates in Path
$P(fs)_{DFFSEU}$	Directly Proportional	N/A
StartPoint	Inversely Proportional	Inversely Proportional
CL	Directly Proportional	Directly Proportional

What about MBUs?

- MBU: multiple DFFs are affected by one SEU strike.
  - Particle passes through 2 sensitive transistor nodes
  - Charge sharing between two nodes due to energy transfer.
- FPGA data path elements can be categorized as follows:
  - Manufacturer circuits: Inside the FPGA logic blocks… manufacturer must follow strict placement design rules to avoid charge sharing… they do
  - Designer/User circuits: A system of building blocks. Automated placement tool follows strict design rules.
- Separate DFFs are placed too far apart and do not suffer from MBUs (a DFF that is Dual Interlocked Cell (DICE) mitigated does not apply). Proven from many years of FPGA SEU testing

SEU Test Structures and Their Error Trends

Ideal Test Structures for Functional Logic Path SEU Testing

- Goal is to estimate error rates for a target design/operation
- Must consider:
  - Error rates of components
  - Error rates of components when connected as a system (design)
  - Error rates of components in a system during operation versus static error rates

Ideal Test Structures for Functional Logic Path SEU Testing

- Test structures should be constructed to expose the sensitivity of various FPGA components
  - Visibility is key
  - State space traversal – simple designs
  - Error differentiation
  - Complexity management – want enough complexity but not too much such that errors are masked, confused, or never obtainable
- Think probability – think statistics – think repeatability

What Design is Best for Single Event Testing?

Simple Architecture

- No/minimal functional Masking
- Easy to base-line across FPGAs
- increases state space coverage
- Differentiation of errors in simpler
- Data may not map into real design

Complex Architecture

- Functional Masking
- Reduction in state space coverage
- Data may be a better fit for real designs
- Error differentiation becomes more complex
-

Logic Complexity Issues and SEU Testing

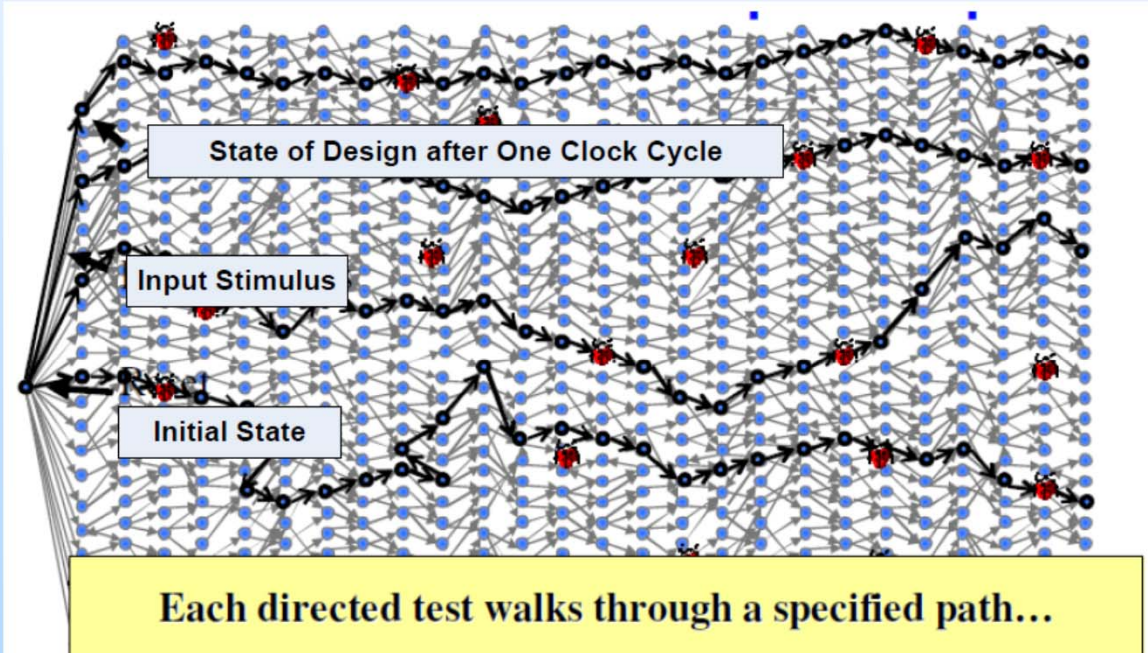
- **Complete flight designs are generally not used as test structures during accelerated SEU testing because complete designs are too complex:**
  - **Error may not be visible during a test run (MASKING)**
  - **Error may shut down the system during a test run and is unable to be identified. (QUICK SEFIs – No Identification)**
  - **Design isn't ready**

Minimal visibility

Minimal state-space coverage

Bad Statistics

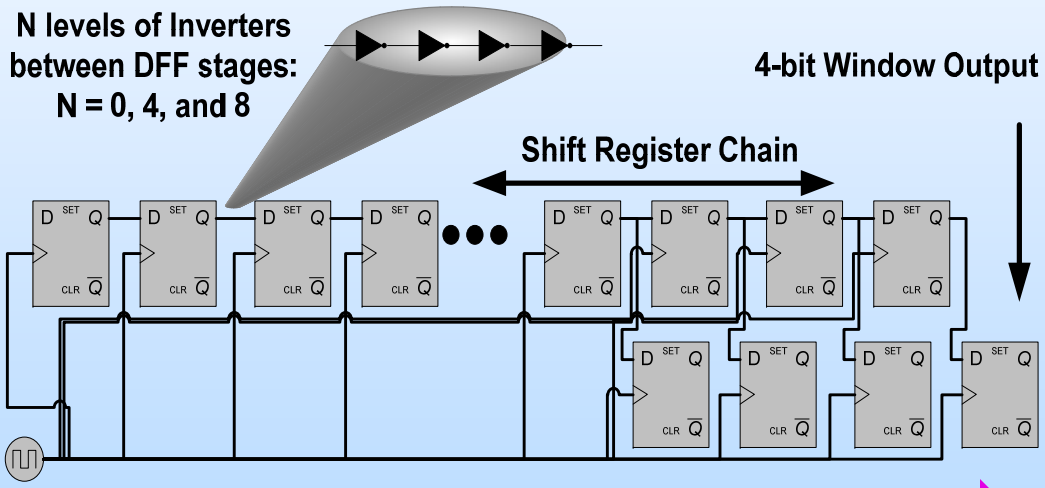
What are you getting out the test?

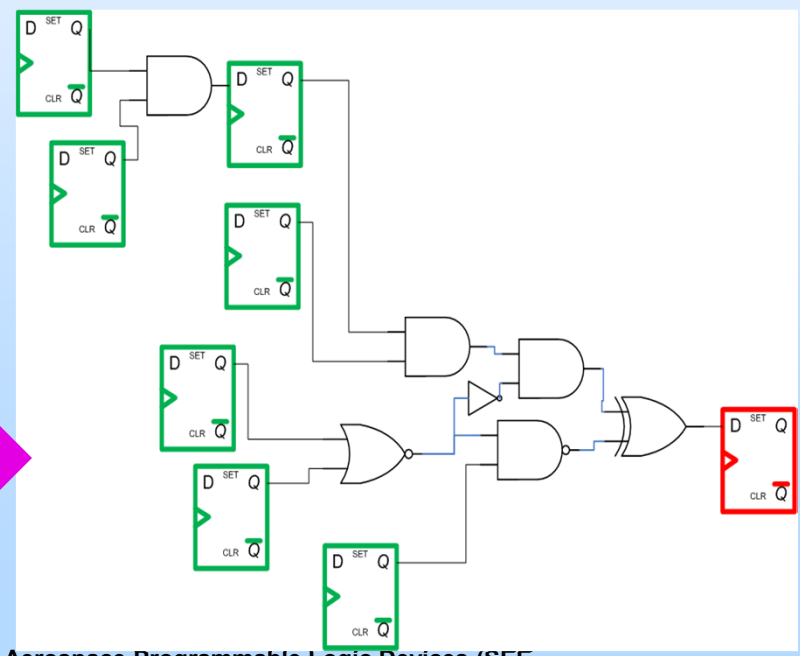


How long does it take to walk through a path?

Test Structures and Data Extrapolation

- **The idea is to extrapolate test structure SEU data to estimate the target design SER.**
- **One test structure may not be sufficient to characterize the target design susceptibility**





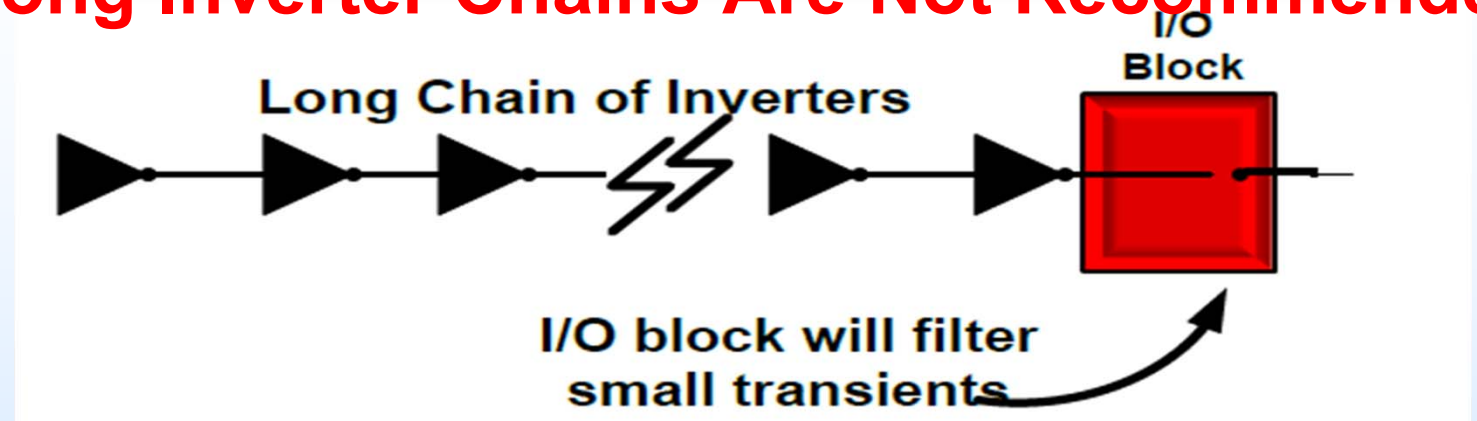
Does the σ_{SEU} data map?

Test Structures and Data Extrapolation (2)

- We create test structures to analyze various components and study their SEU error response under various design topologies. It's all about trends:
  - What happens when we add combinatorial logic
  - What happens as we increase frequency
  - What happens when we add logic to our clock trees
- DFFs are our capture components and are key… how does a DFF end up in an incorrect state
  - DFFs have an upset rate
  - DFFs can capture SETs in their data path
  - DFFs clocks and resets that can cause upsets

Implementing Traditional Test Structures In FPGA Devices.

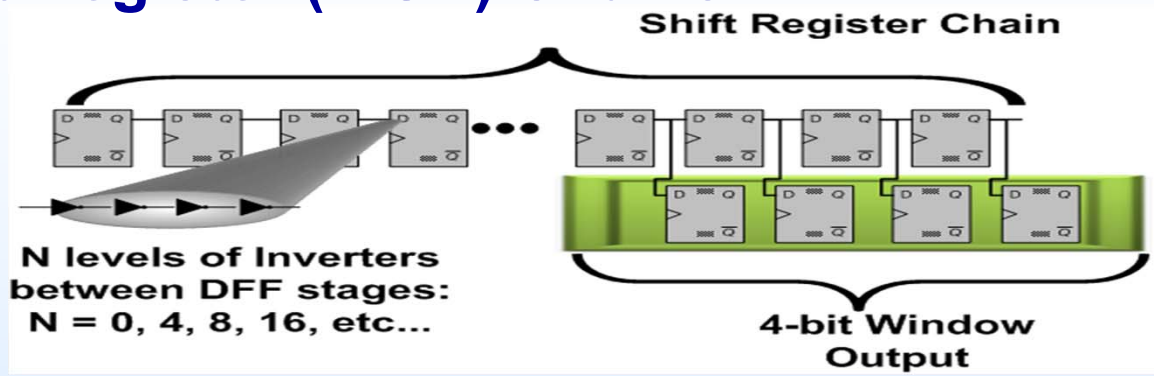
Long Inverter Chains Are Not Recommended



- **Goal is to calculate susceptibility of inverters? Or to evaluate General combinatorial logic gates? Issues:**
  - **Assumes cascaded combinatorial logic has linear SEU effects – However this is not true (capacitive effects such as attenuation)**
  - **Does not take into account inverters are not inverters in FPGA devices. The formation of an inverter requires additional circuitry**
  - **Does not take into account complex routing due to the length of the chain**
  - **Inverter chains in FPGA devices have a significant amount of noise. Most FPGAs are made to implement synchronous designs**

Implementing Traditional Test Structures In FPGA Devices (1).

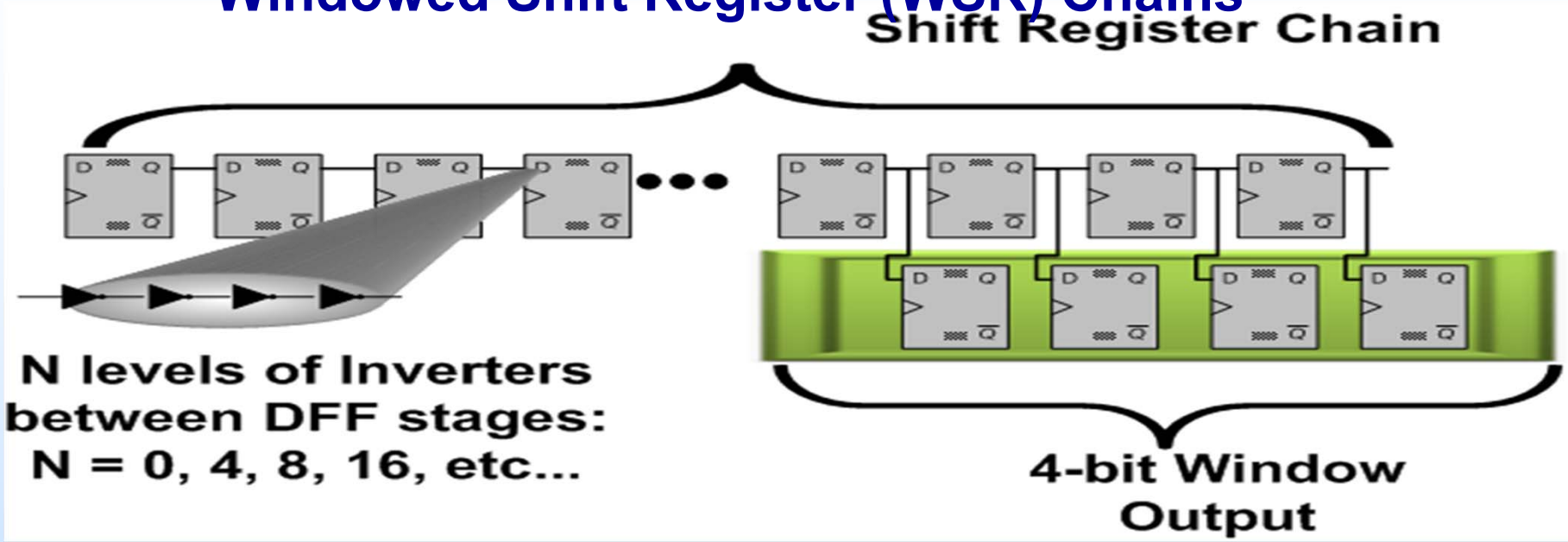
Windowed Shift Register (WSR) Chains



- Goal is to calculate susceptibility of DFFs and CL gates.
- Benefits
  - Simple test structure that has no logic masking
  - Use of combinatorial logic and DFFs helps to study trends. What happens as the amount of combinatorial logic is increased? Does frequency matter?
  - Can easily traverse the entire state space several times over during accelerated testing

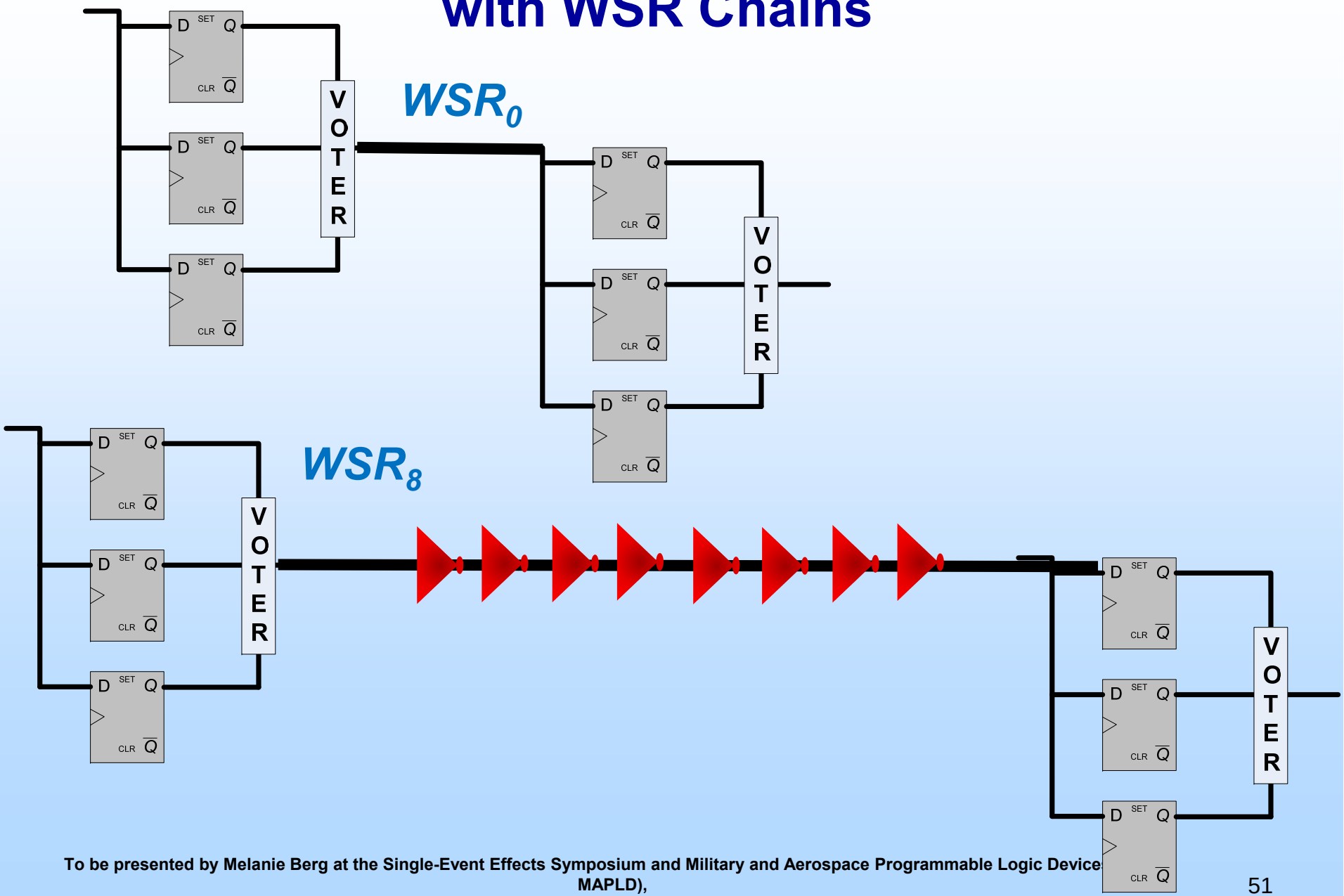
Implementing Traditional Test Structures In FPGA Devices (2).

Windowed Shift Register (WSR) Chains

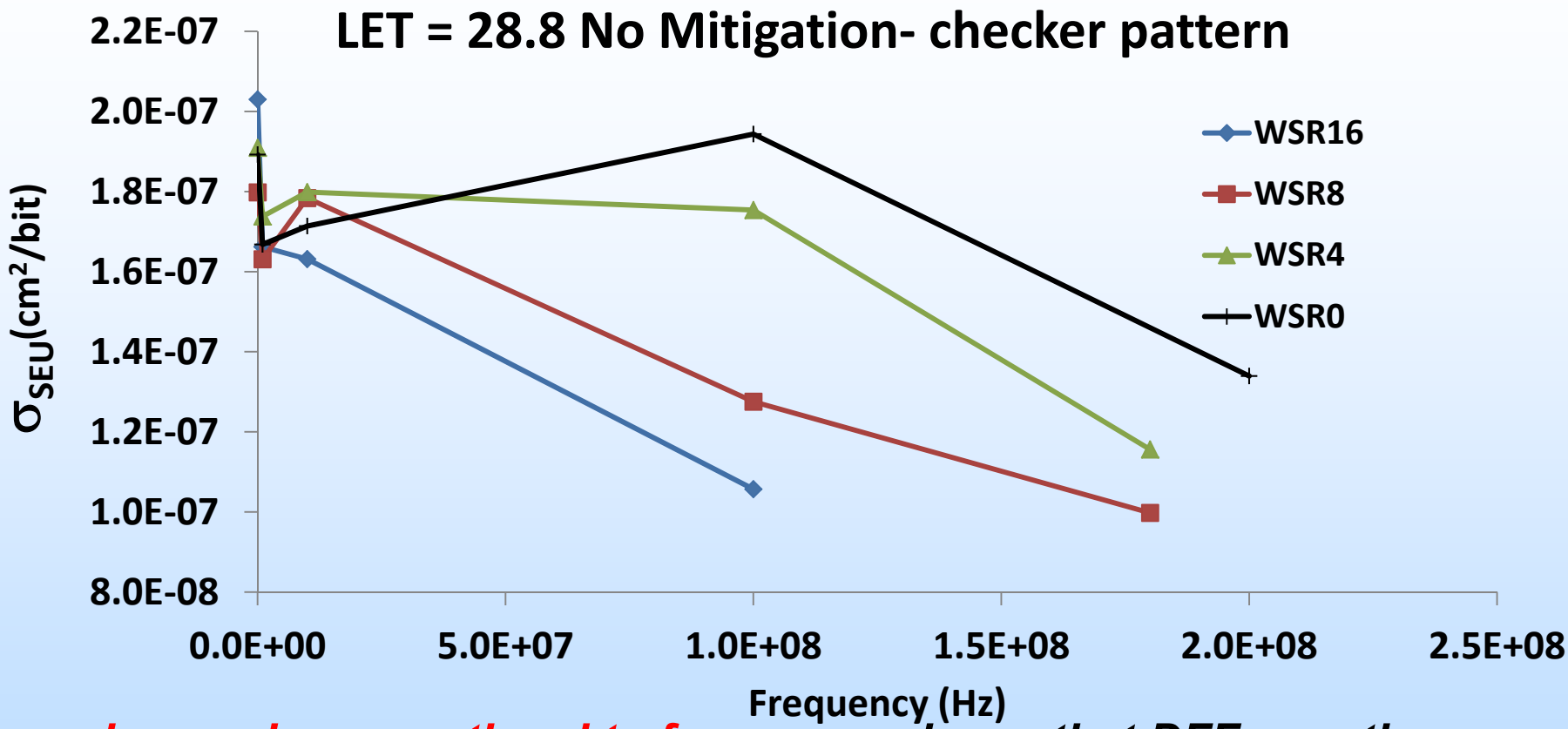


- **Disadvantages - Does not represent a complex design well:**
  - **No logic masking**
  - **Linear path of combinatorial logic**
  - **Capacitive loading is minimized**

Localized Triple Modular Redundancy (LTMR) with WSR Chains



Frequency Effects and WSRs in the Microsemi ProASIC3 FPGA

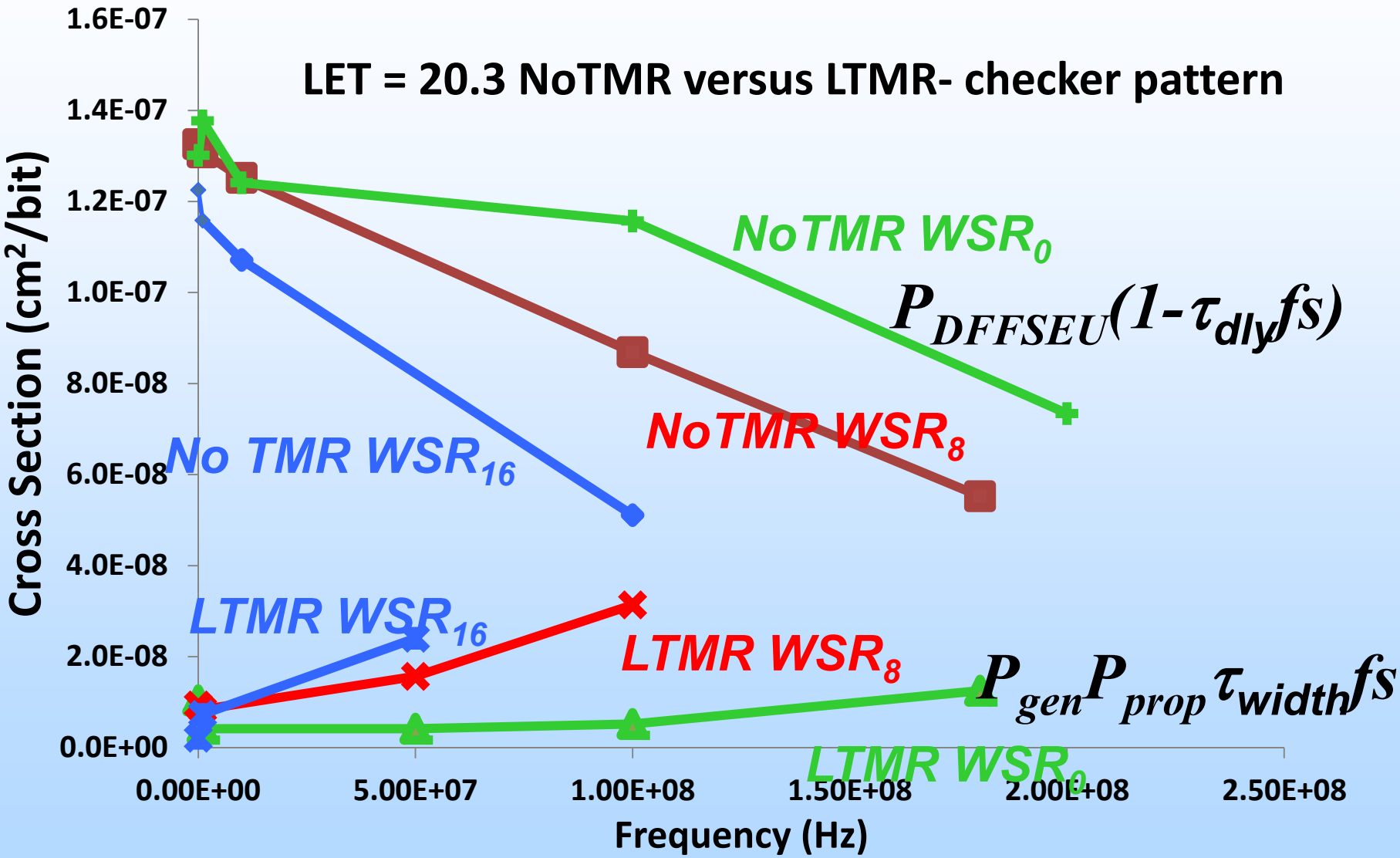


σ_{SEU} inversely proportional to frequency shows that DFFs are the dominant source of error

σ_{SEU} inversely proportional to amount of combinatorial logic between DFFs shows that DFFs are the dominant source of error

Testing across frequency is important

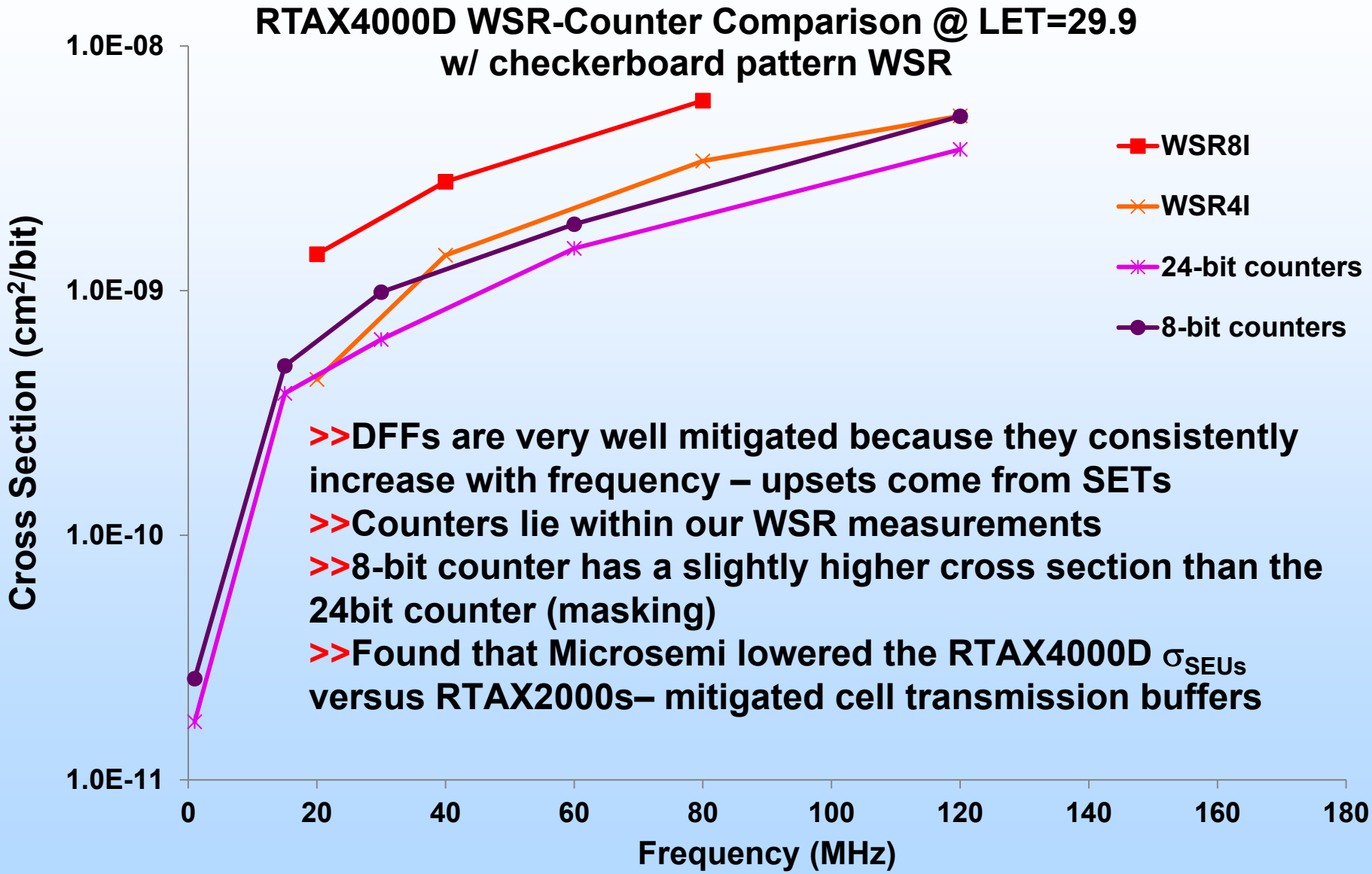
Another Look at No-TMR versus LTMR with the ProASIC3… Regard the Frequency Trends



Increasing Test Structure Complexity: Shift Registers versus ?

- There are benefits to increasing design complexity. However, limitations must be taken into account:
  - State space traversal during testing
  - Amount of logic masking
  - Visibility of upsets
- NASA REAG uses counters and digital signal processing units (e.g. multipliers and accumulators) as test structures
- Interface (I/O) management can be difficult for high-speed circuits or designs with a large number of I/O.
  - Built-in-Self-Test (BIST) can be a solution
  - Caution: BIST circuits have limited visibility –
    - Error differentiation can become extremely difficult
    - Determining if the test is operating correctly can become difficult

Comparison of WSRs and Counters in the Microsemi-RTAXs. Embedded LTMR



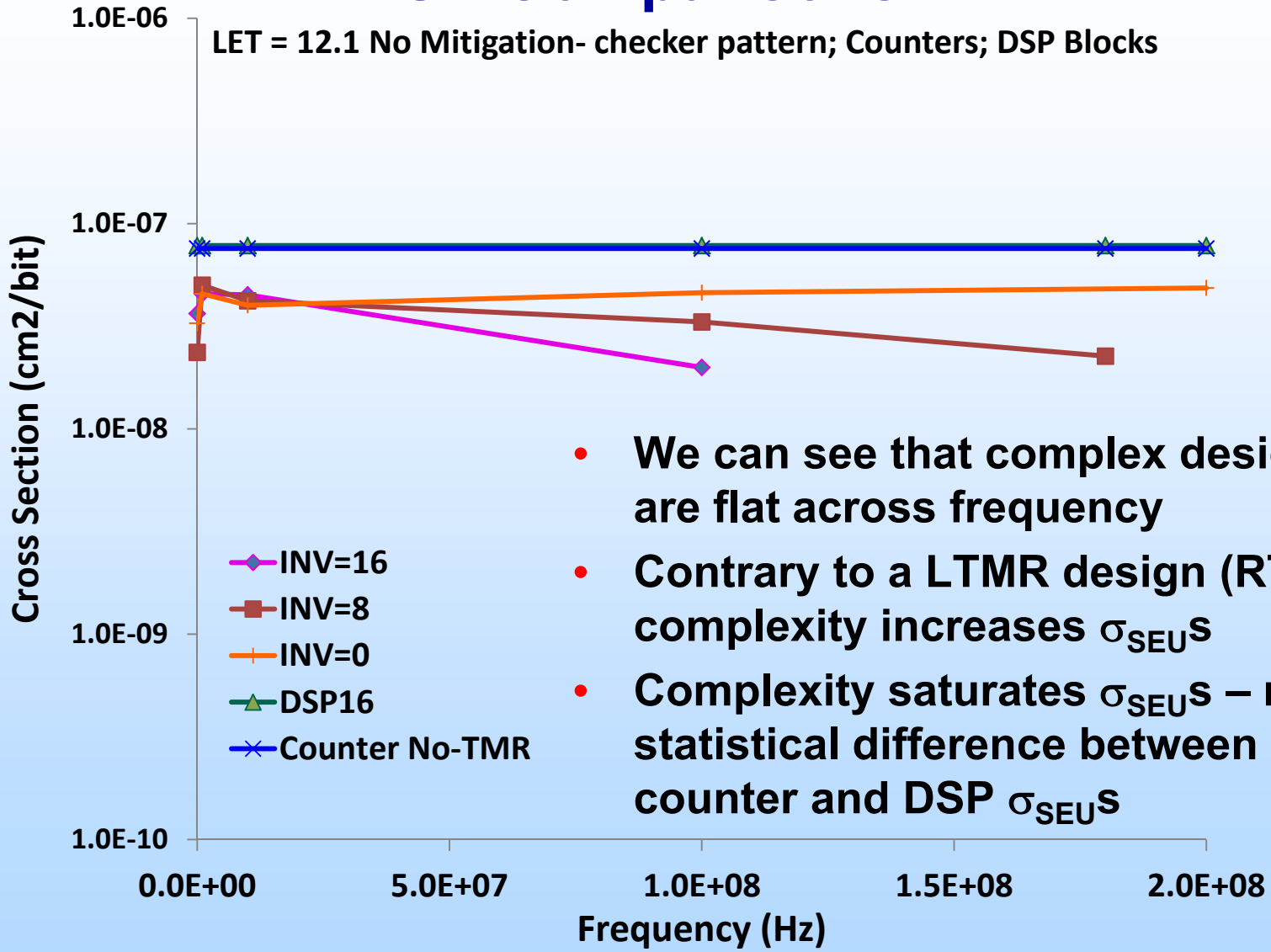
>>DFFs are very well mitigated because they consistently increase with frequency – upsets come from SETs

>>Counters lie within our WSR measurements

>>8-bit counter has a slightly higher cross section than the 24bit counter (masking)

>>Found that Microsemi lowered the RTAX4000D σ_{SEUs} versus RTAX2000s– mitigated cell transmission buffers

ProASIC3 Non-Mitigated WSR, Counter and DSP Comparisons



- We can see that complex designs are flat across frequency
- Contrary to a LTMR design (RTAXs), complexity increases σ_{SEU}s
- Complexity saturates σ_{SEU}s – no statistical difference between counter and DSP σ_{SEU}s

Conclusion

- This presentation covered a small portion of SEU characterization for FPGA designs: test structure selection
- Appropriate test structure selection is key to accurate SEU characterization
  - Simple test structure error responses may not represent complex designs. Hence mapping of radiation data may not be accurate
  - Complex test structures will limit visibility of errors and state space traversal
  - It is best to study a variety of test structures and analyze radiation data trends:
    - Amount of combinatorial logic
    - Frequency
    - Input data pattern
- Testing across frequency is essential