Single Event Induced Multiple Bit Errors and the Effects of Logic Masking

Presented by: Melanie Berg, MEI Technologies in support of NASA/GSFC Melanie.D.Berg@NASA.gov

Ken LaBel, Jonathan Pellish: NASA Goddard Space Flight Center

Hak Kim Christina Siedlick: MEI Technologies in support of NASA Goddard Space Flight Center

Presented by Melanie Berg at the Nuclear Science Radiation Effects Conference (NSREC), San Francisco, CA, July 7-12, 2013.

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*
- *Thanks is given to Microsemi for supplying field programmable gate array (FPGA) test devices*

Acronym Table

Acronym	Definition	Acronym	Definition
AND	And logic gate	CL	Combinatorial Logic
FPGA	Field Programmable Gate Array	f_s	Operational frequency (equal to the inverse of clock period)
DFF	Flip-flop	EndPoint	Ending flip-flop for a data path
INV	Inverter logic gate	LET	Linear Energy Transfer
P_{logic}	Logic masking Probability	SET	Single Event Transient
SEU	Single Event Upset	StartPoint	Starting flip-flop for a data path
σ_{SEU}	Single Event Upset Cross Section	τ_{clk}	operational clock period (equal to the inverse of frequency)
τ_{dly}	time delay from Start Point flip-flop to End Point flip-flop	XOR	exclusive-or logic gate

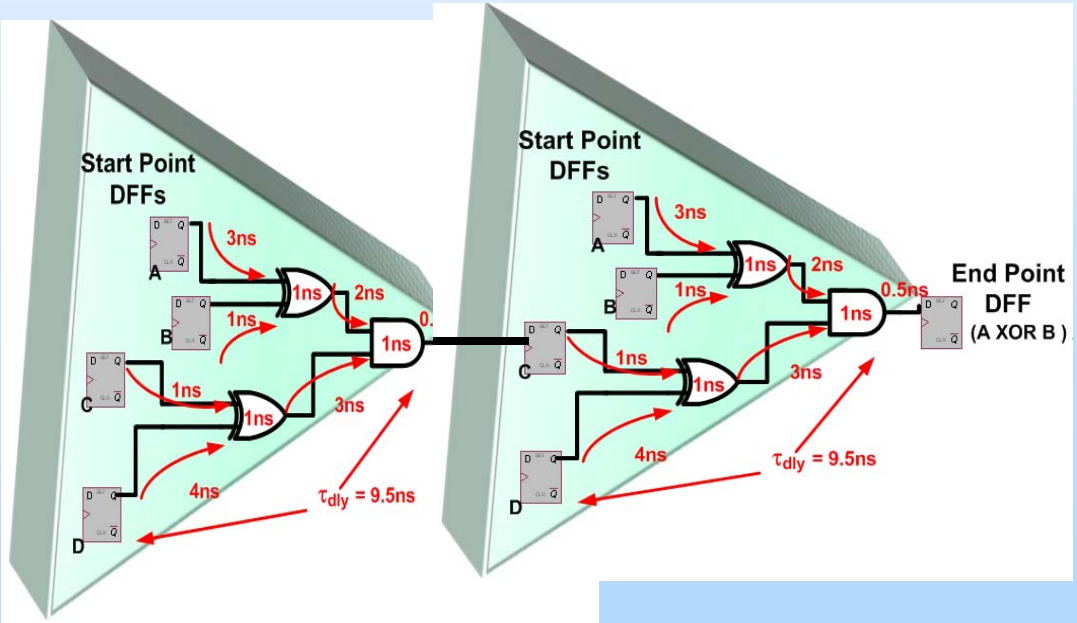
Abstract

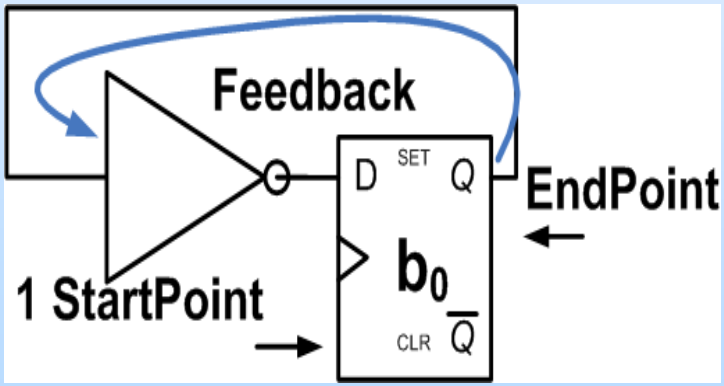
- We investigate how an SEU (occurring mid-clock-cycle) can induce multiple bit errors in the next clock-cycle
  - We refer to this as a domino-effect
  - We leverage our investigation from the NASA Goddard REAG FPGA SEU model
  - We investigate how logic masking (P_{logic}) can reduce the domino-effect
- Proposed Process:
  - Establish data-paths (StartPoint→CL→EndPoint) that feed into each EndPoint (cone-of-logic)
  - Calculate the probability of reaching erroneous states using the model and σ_{SEU} information

We want to extrapolate σ_{SEU} data to complex designs so we can predict next state behavior under SEU conditions

Background (1) : Data-Paths…StartPoints→CL→EndPoints

- Based off of defining data-paths as cones of logic, 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

Background(2) : The NASA REAG SEU FPGA Model; The Probability an EndPoint will have an Erroneous Next State

One sum-term per next state of one EndPoint

EndPoint

StartPoints

Combinatorial Logic

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

Probabilities:

- $P_{logic(k)}$ ➔ ***EndPoint Logic Masking***
- $P_{logic(j)}$ ➔ ***StartPoint Logic Masking***
- $P_{logic(i)}$ ➔ ***Combinatorial Logic Masking***

$P(fs)_{DFFSEU}$ ➔ ***DFF Upset***

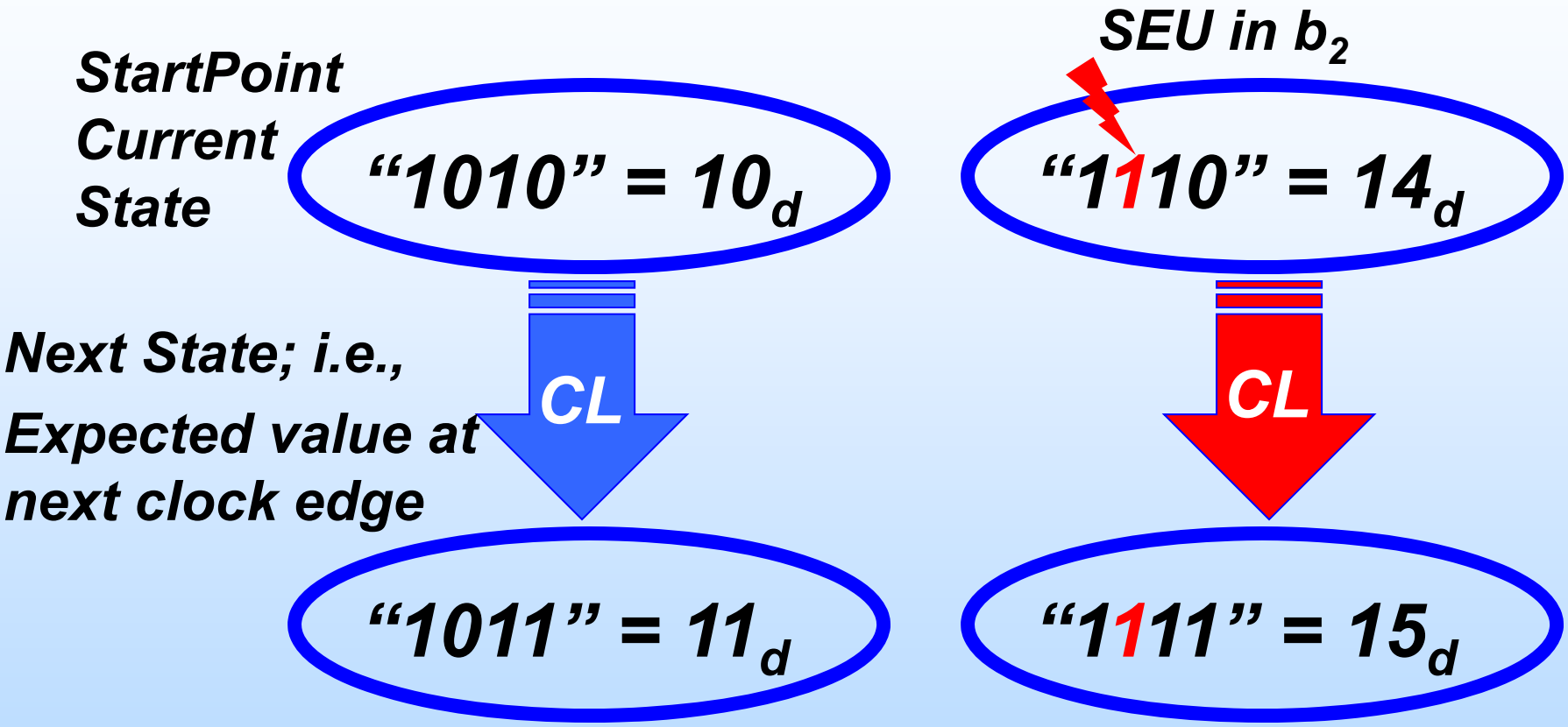
$P_{prop(i)}$ ➔ ***Electrical Masking***

$P_{gen(i)}$ ➔ ***SET generation***

τ_{width} = ***average width of SET***

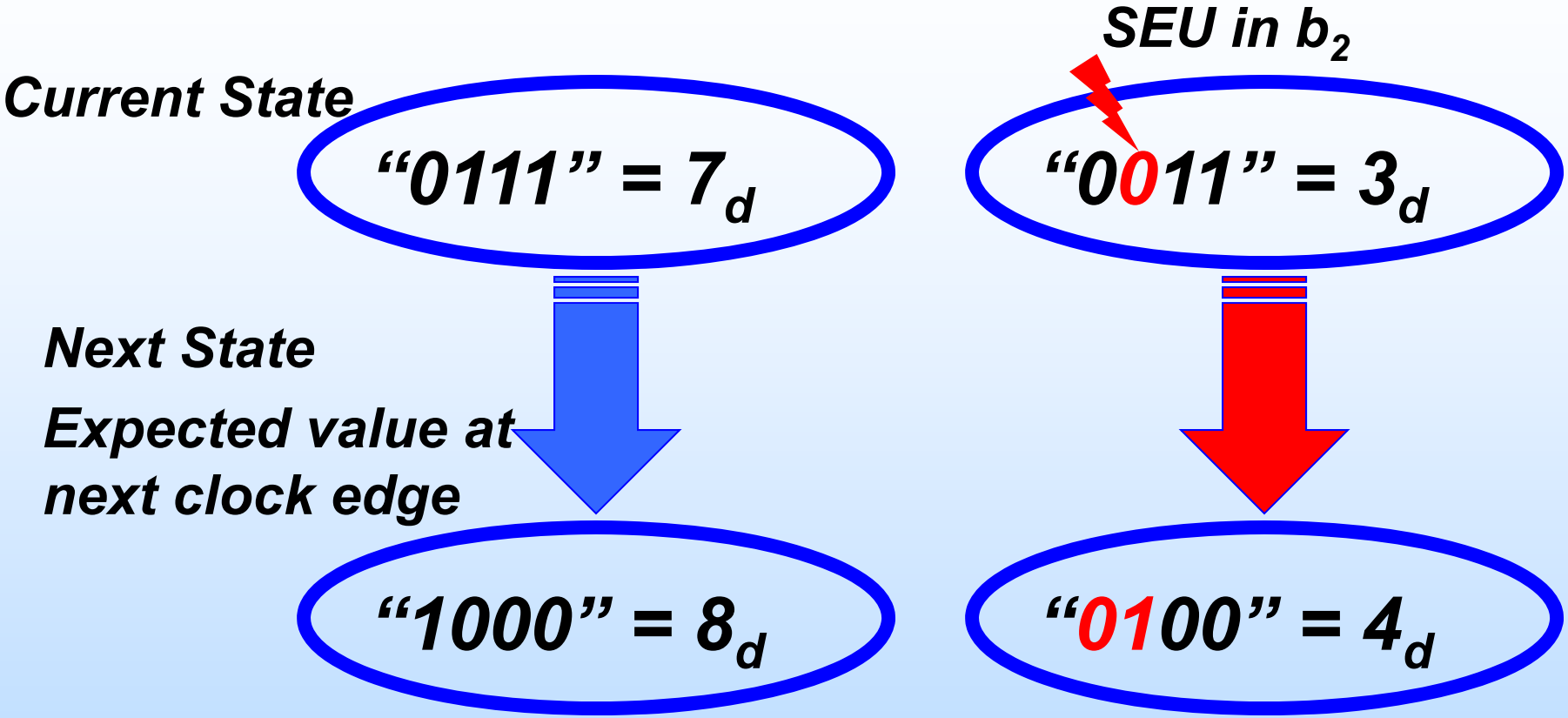
- **For designs with non-mitigated DFFs the dominant upsets are StartPoints (i.e., SETs and EndPoint upsets are insignificant)**

Counter Current State and Expected Next State : Single bit SEU Occurring between Clock Edges– Single bit SEU in Next State



- An SEU that occurs between clock edges in b_2 causes b_2 to be incorrect in the next clock cycle

Counter Current State and Expected Next State: The Domino Effect



- An SEU in b_2 causes b_2 and b_3 to be erroneous in the next clock cycle.
- The number of EndPoints that will be incorrect will depend on the current state, CL, τ_{dly}, and P_{logic}.

CL between StartPoints and Endoint Define Logic Masking (P_{logic})

Next state for EndPoint at clock cycle (T):

$$EndPoint\,DFF(T) = f(StartPoint\,DFFs(T-1), CL)$$

- P_{logic} =1 :
  - No logic masking
  - Inverter, buffer, XOR
- 0<P_{logic} <1
  - Variable logic masking
  - AND, OR, MUX
- P_{logic} =0: full logic masking
  - Complete logic masking 100% of the time
  - Global Triple Modular Redundancy (GTMR) with voter

Probability that an EndPoint will have an erroneous next state:

$$\sum_{i=1}^{\#StartPoints} \beta P(fs)_{DFFSEU(j)} (1-\tau_{dly} fs) * P_{logic(j)}$$

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

Once we figure out the equation for each EndPoint, we can determine the impact of P_{logic} and potential Domino-Effects

Counter: EndPoint b_0 Data-Paths

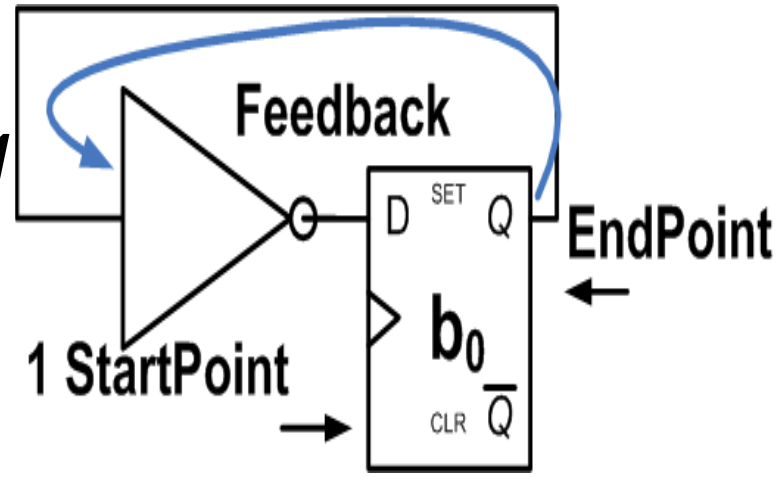
function b_0= INV b_0

4-Bit Counter: States 0..15

b3	b2	b1	b0	State
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

$$EndPoint\,DFF(T) = f(StartPoint\,DFFs(T-1), CL)$$

P_{logic} = 1



EndPoint=b_0

- b_0: next state = inverse of current state
- Only one data path: b_0 → INV → b_0

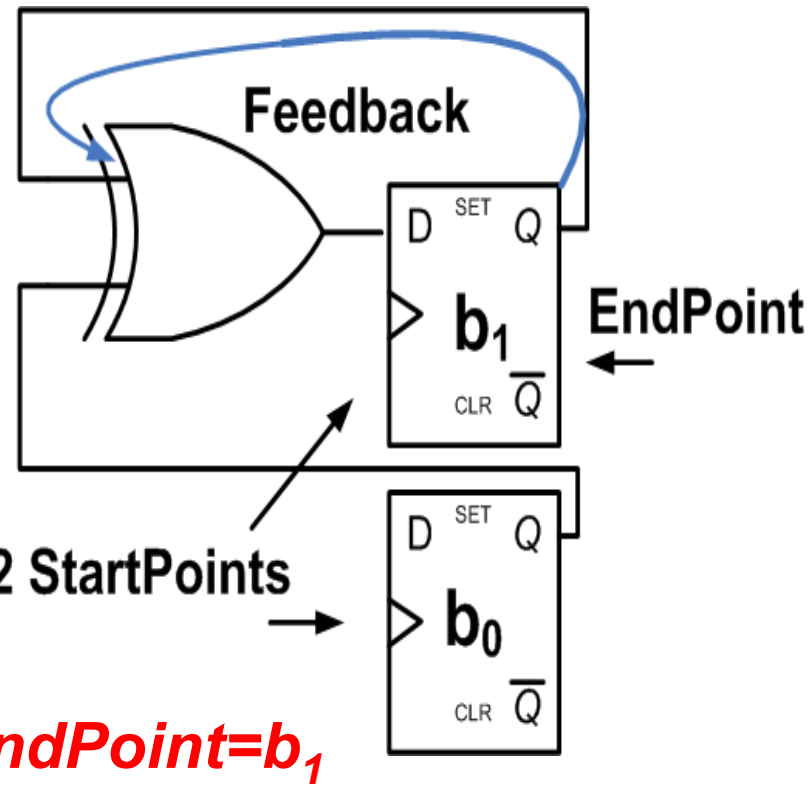
Counter: EndPoint b_1 Data-Paths:

function: $b_1 = b_1$ XOR b_0

4-Bit Counter: States 0..15

b3	b2	b1	b0	State
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

$P_{logic} = 1$



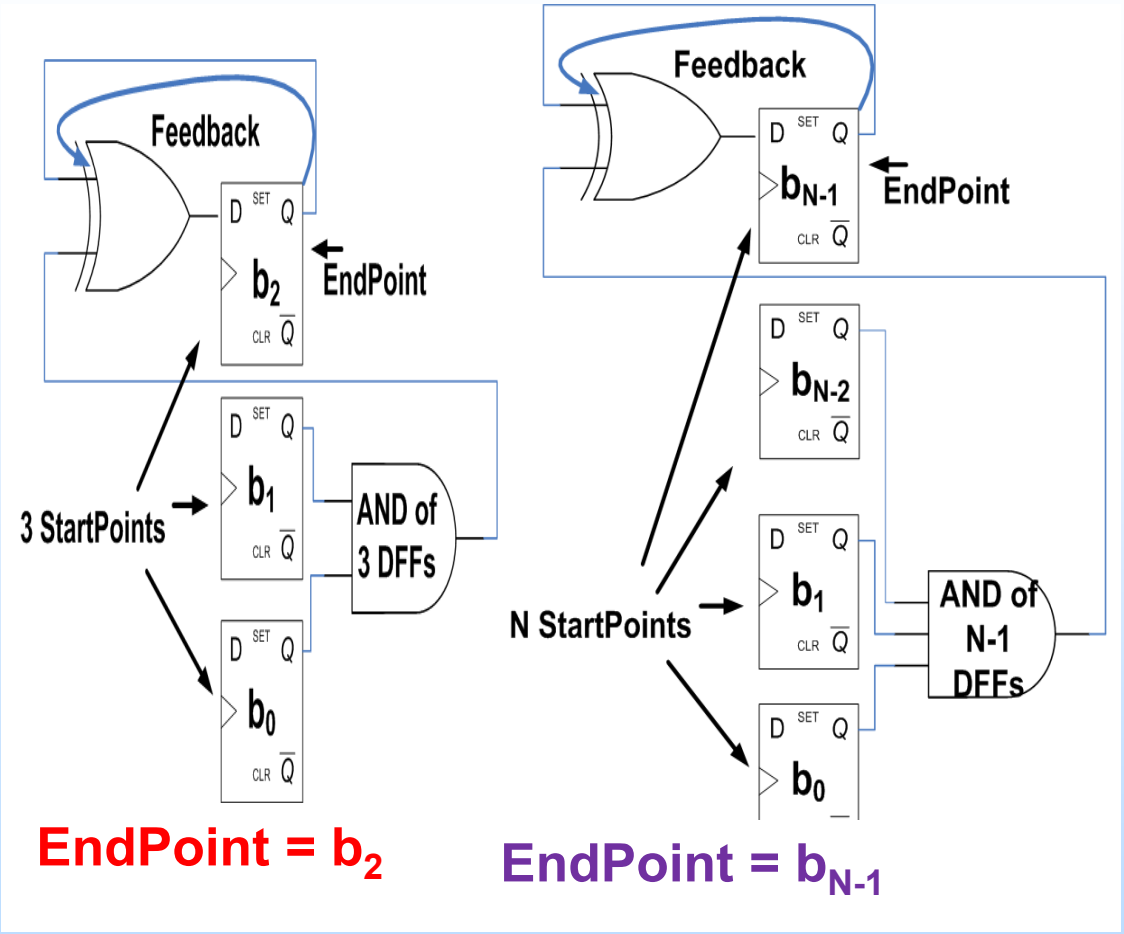
- b_1: next state = XOR of current state for b_0 and b_1
- Two variables= two data paths:

b_0 → XOR → b_1 and b_1 → XOR → b_1

Counter: EndPoints b_k Data-Paths

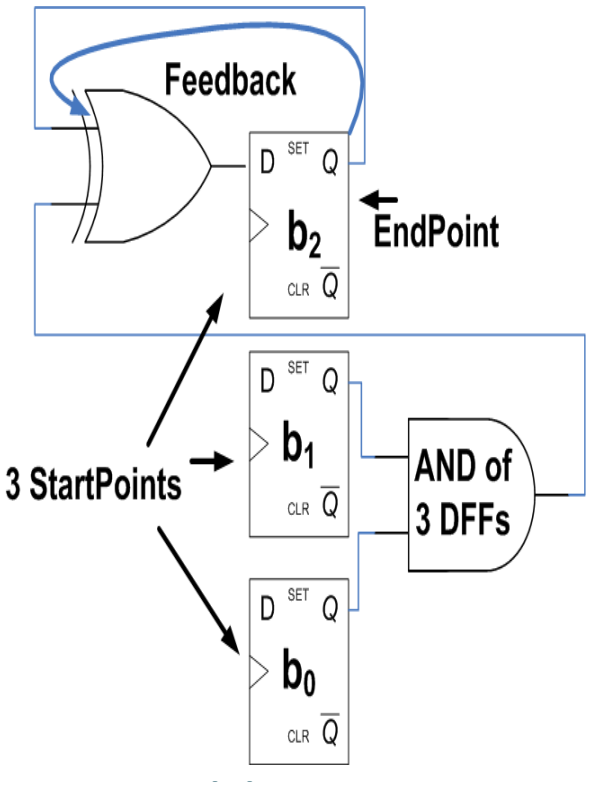
4-Bit Counter: States 0..15

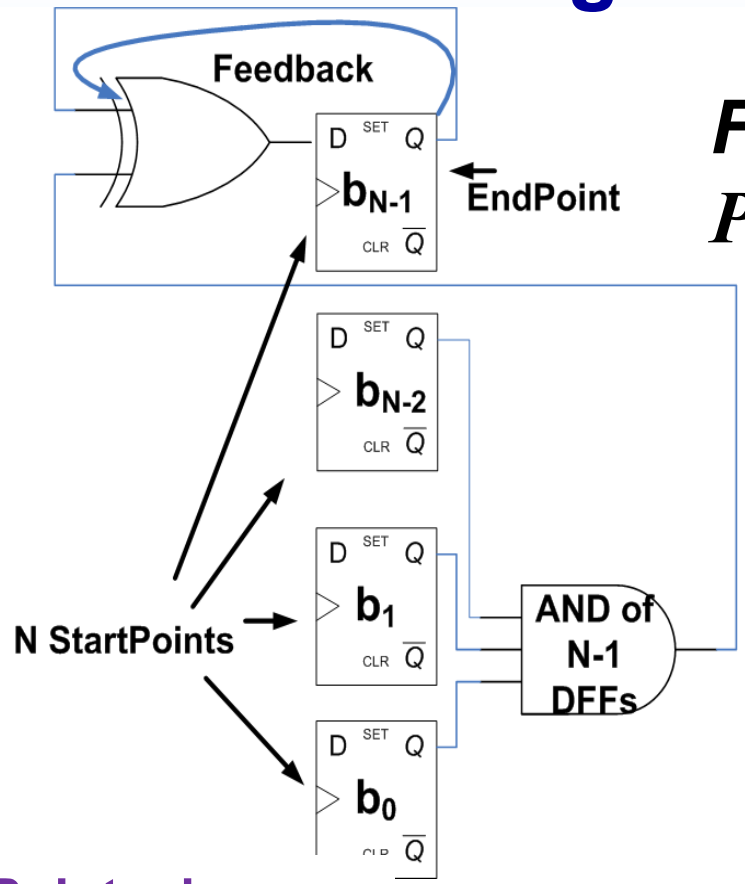
b3	b2	b1	b0	State
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15



- **b_k next state = inverse of current state if all lower bits are equal to a logic '1'**

Domino-Effect Occurs through And-Path yet P_{logic} Reduces its Significance





Feedback Path:
$P_{logic} = 1$

AND Paths:
$P_{logic} = 2^{-k}$

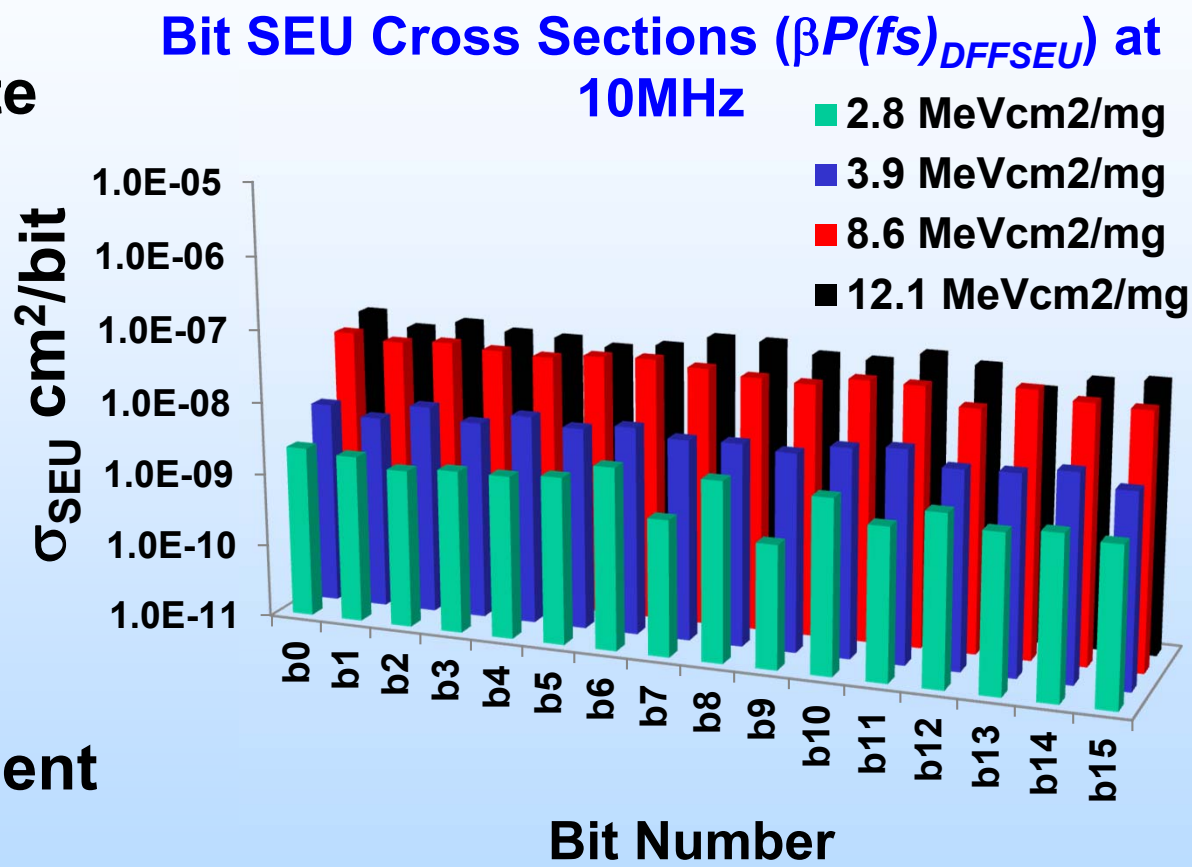
EndPoint = b_2

EndPoint = b_{N-1}

- **b_k → XOR → b_k feedback data: 1 path**
- **b_j → AND → XOR → b_k k paths: $0 \le j \le (k-1)$**

ProASIC3 FPGA σ_{SEU}s for StartPoint DFFs obtained at Texas A&M Cyclotron Facility

- Data does not represent Next State domino effects
- …they represent StartPoint DFFs $\beta P(f_s)_{DFFSEU(j)}$
- Each DFF bit has statistically equivalent σ_{SEU}s



How can we apply this data to predict next state response?

Extrapolation of SEU Data…Finding the Probability the Next State of b_2 will not Match its Expected Value

$$for\ \tau_{dly(j)} \ll \tau_{clk} \qquad \sum_{j=1}^{\#StartPoint\ DFFs} \left(\beta P_{DFFSEU(J)} * P_{logic(j)} \right);$$

Feedback:

$P_{logic} = 1$

And-Path:

$P_{logic} = 2^{-2} = 0.25$

And-Path:

$P_{logic} = 2^{-2} = 0.25$

$$\beta P(f_s)_{DFFSEU(b2)} + 0.25\beta P(f_s)_{DFFSEU(b1)} + 0.25\beta P(f_s)_{DFFSEU(b0)}$$

From the σ_{SEU} graph at LET = 12 MeV-cm²/mg

$$\beta P(f_s)_{DFFSEU(b2)} \approx \beta P(f_s)_{DFFSEU(b1)} \approx \beta P(f_s)_{DFFSEU(b0)} \approx 4.0 \times 10^{-7}\ bit/cm^2$$

σ_{SEU} for the next state of b_2 not equal to its expected value: $\approx 6.0 \times 10^{-7}$ bit/cm²

Each EndPoint DFF (b_0 to b_n) undergoes same analysis

Conclusion(1)

- **We use a counter as the design under analysis to illustrate the domino-effect – the potential for an SEU to induce multiple bit errors.**
- **The following are real design examples of domino-effect circuit responses and considerations:**
  - **Do not assume that an SEU will only perturb one bit within one clock cycle; e.g.:**

    Will the SEU response of a state machine cause the next state to be uncontrollable because the "safe" state implemention cannot protect against this situation?
  - **Proliferation of control logic to various portions of a circuit must be taken into account; e.g.:**

    Will one SEU fan-out to all reachable paths or just some? Can the error correction or protection handle this event?

Conclusion(2)

- **The domino effect can be reduced by several factors, however we focus on the effects of logic masking (P_{logic})**
- **This study is part of a process under development targeted to extrapolate σ_{SEU} data for complex designs**
- **We show how we can use the NASA Goddard REAG FPGA SEU Model combined with SEU data to predict a design's next state under SEU conditions.**
- **Future work will incorporate τ_{dly} and illustrate how this process can be implemented as an automated tool for complex circuit application**

Raw bit σ_{SEU}s may not be good enough for error rate prediction. Design topology must be taken into consideration