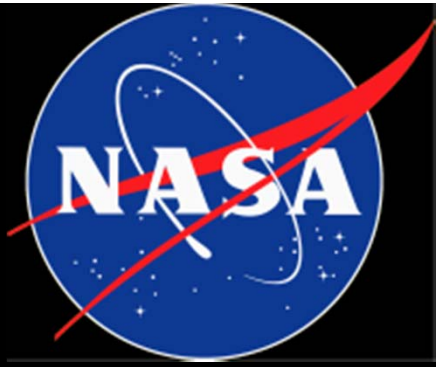
Differentiating Scrub Rates between Space-Flight Applications and Accelerated Single Event Radiation Testing for SRAM based Field Programmable Gate Arrays.



Presented by: Melanie Berg, MEI Technologies in support of NASA Goddard Space Flight Center

Melanie.D.Berg@NASA.gov

Ken LaBel, Jonathan Pellish, Ray Ladbury: NASA Goddard Space Flight Center

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

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/.

Introduction

- The configuration memory of un-hardened static random access memory (SRAM)-Based Field Programmable Gate Arrays (FPGAs) is highly susceptible to single event upsets (SEUs)
- We address configuration susceptibility via scrubbing: Scrubbing is the act of simultaneously writing into FPGA configuration memory as the device's functional logic area is operating with the intent of correcting configuration memory bit errors
- Two questions are addressed:
  - How often should we scrub?
  - What is the difference between scrubbing in a space environment and scrubbing in an accelerated single event test environment

Misperception #1: Space Application Scrub Rates

- It is a common misperception that space applications scrub configuration constantly. They don't:
  - Power,
  - Area, and
  - Risk with additional logic complexity.
- It is important to understand:
  - When configuration memory scrubbing is necessary and if so…
  - How often configuration memory scrubbing should occur (scrub rate)
- We use SEU cross sections (σ_{SEU}) to determine required scrub rates

FPGA SEU Categorization as defined by NASA Goddard Radiation Effects and Analysis Group (REAG):

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

Design σ_{SEU}

Configuration σ_{SEU}

Functional logic σ_{SEU}

SEFI σ_{SEU}

Sequential (DFF) and Combinatorial logic (CL) in data path

Single Event Functional Interrupt (SEFI): Global Routes and Hidden Logic

Depending on the FPGA type and/or mitigation, one of these σ_{SEU}s will be significantly more dominant than the others.

Scrubbing pertains to configuration σ_{SEU}s

Xilinx SX55: Radiation Test Data

– ***Xilinx Consortium: VIRTEX-4VQ STATIC SEU CHARACTERIZATION SUMMARY: April/2008***

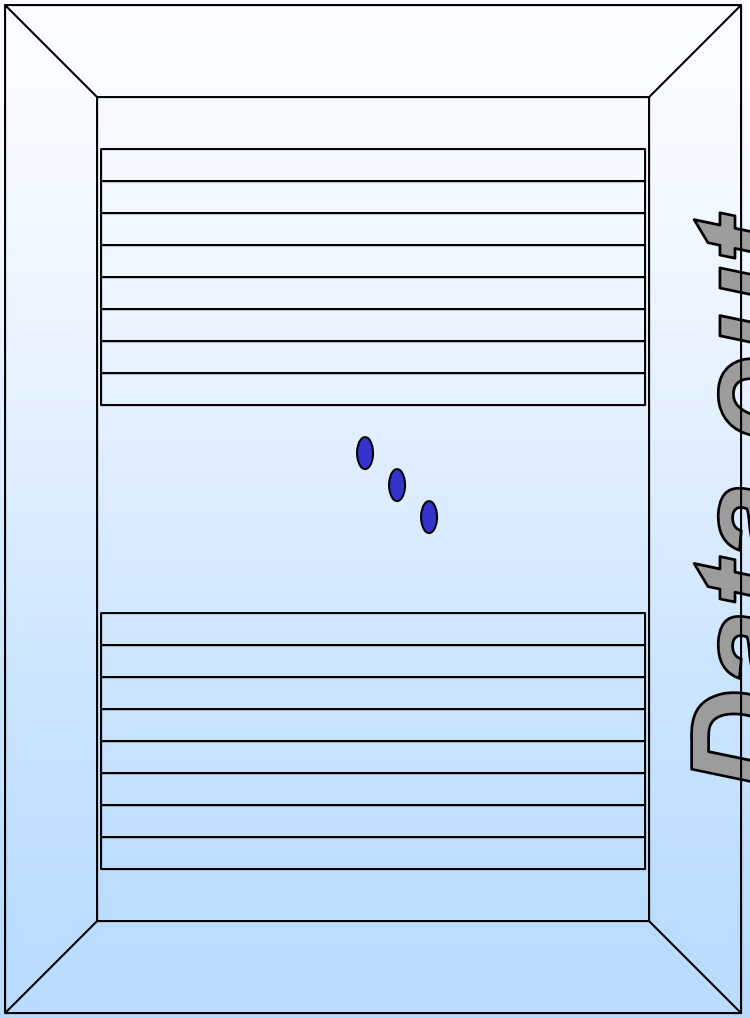
	Probability (σ_{SEU})	Error Rate	LEO $\frac{Upsets}{device-day}$	GEO $\frac{Upsets}{device-day}$
Configuration Memory: XQR4VSX55	$P_{configuration}$	$\frac{dE_{configuration}}{dt}$	**7.43**	**4.2**
Combined SEFIs per device	P_{SEFI}	$\frac{dE_{SEFI}}{dt}$	**7.5×10^{-5}**	**2.7×10^{-5}**

- **For non-mitigated designs the most significant SEU factor:**

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

- **For non-mitigated designs, statistically, there is no need to be concerned with the other terms (P_{SEFI} and $P_{functionalLogic}$)**

Traditional SRAM … One Data Word at a Time

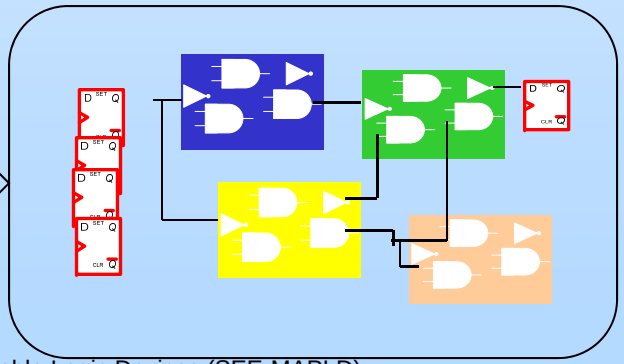


Data out

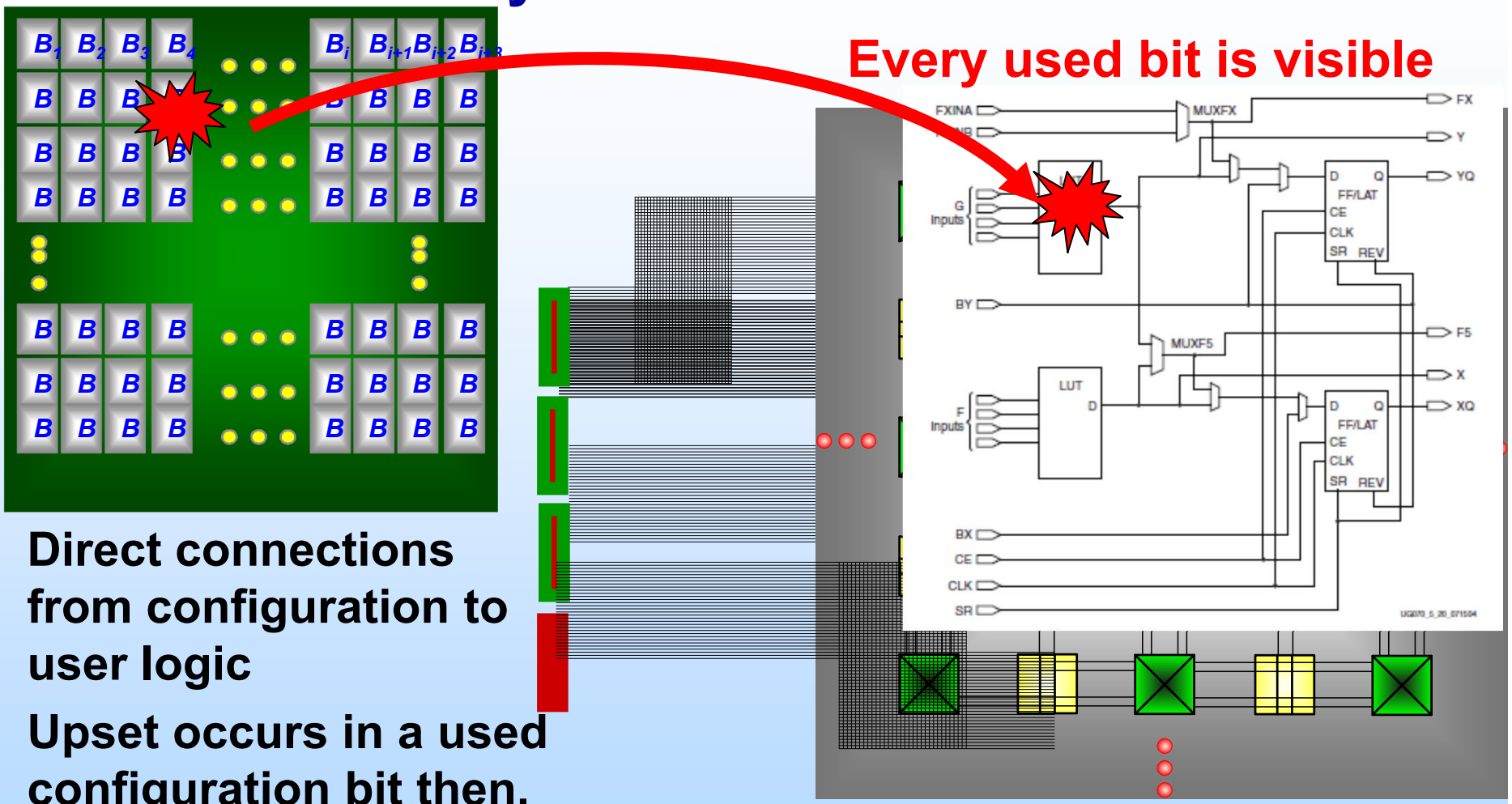
EDAC

- Upsets have no effect until Address containing upset is read out of SRAM
- Error detection and correction (EDAC) are placed after data out
- EDAC circuits only work one data word at a time

User Circuitry



Configuration SRAM is NOT Utilized the Same Way as Traditional SRAM



- Direct connections from configuration to user logic
- Upset occurs in a used configuration bit then, upset occurs in logic

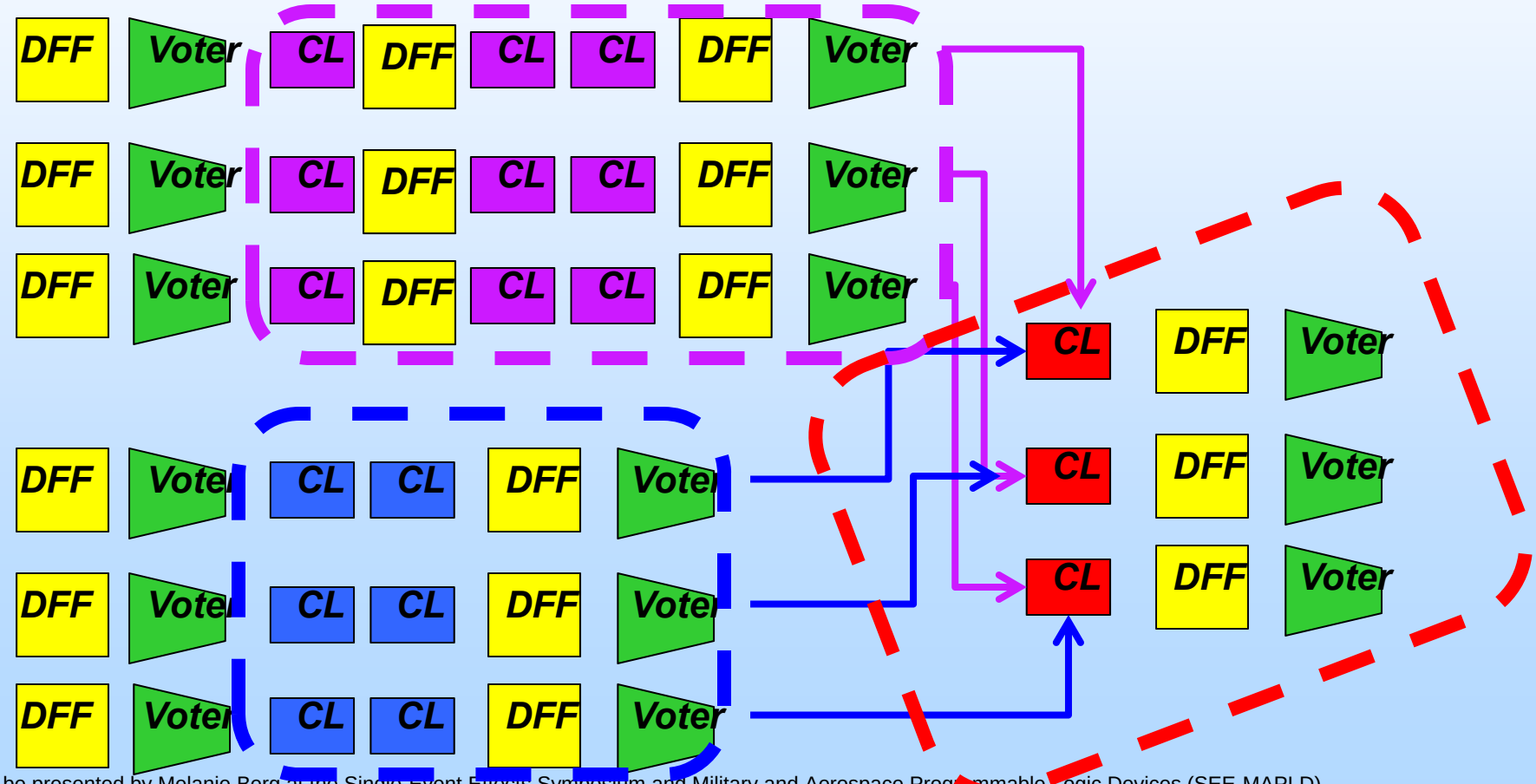
- We're not dealing with data words anymore. Traditional SRAM EDAC schemes don't quite apply for configuration SRAM

Mitigation and SRAM Based FPGAs

- Mitigation is the act of dealing with an upset
- Upsets need to be clearly defined:
  - Does an upset mean that the circuit is malfunctioning?
  - Does an upset mean non-recoverable failure or is recoverable but needs a reset?
  - Configuration upsets versus Functional upsets?
- There is a difference with how we handle upsets:
  - Detecting: Determining an upset exists in the circuitry
  - Masking: blocking an error from affecting functional behavior
  - Correcting:
    - Can use detection circuitry to correct; e.g. error correction and detection (EDAC)
    - Can blindly (automatically) correct (e.g. triple modular redundancy (TMR))

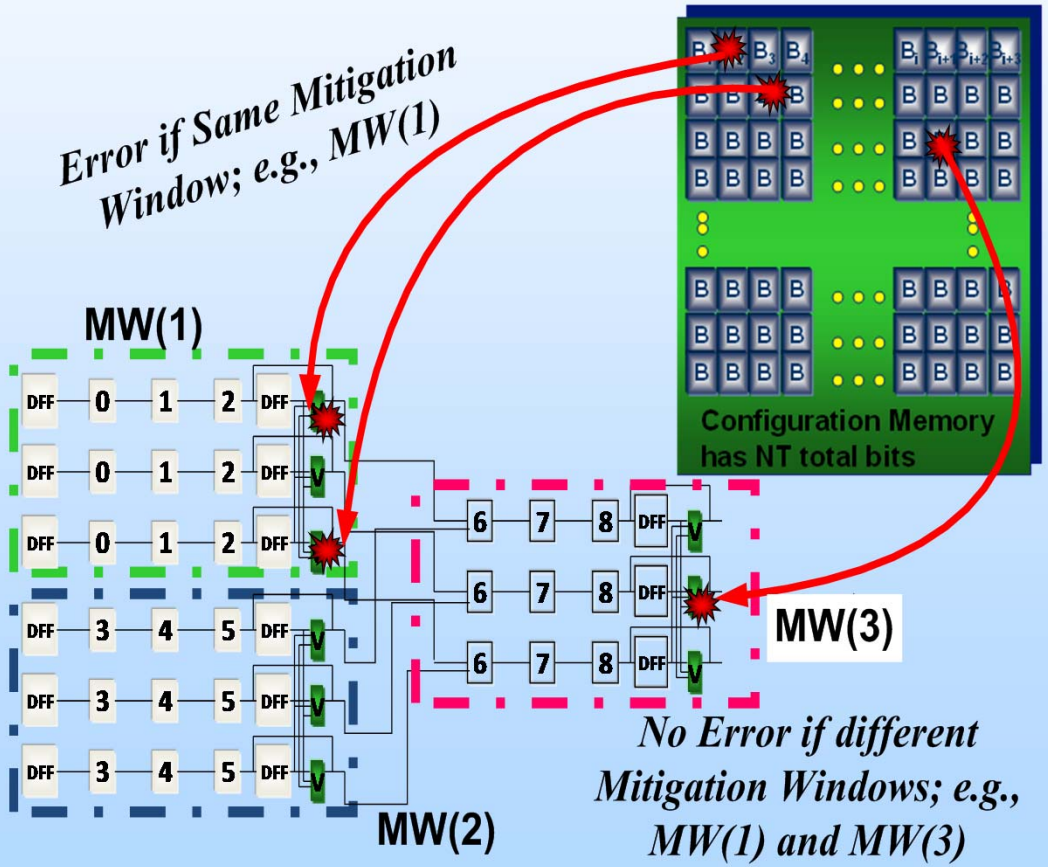
TMR Mitigation Window Definition

- **MW boundaries: Start at a voter-output or a device input; End at a DFF-voter pair or a device output**
- **Internal MW elements can be CL or DFFs (i.e., if a DFF does not have a voter, then it is not a MW boundary)**



TMR in SRAM Based FPGAs and Mitigation Windows (MW)

- **Two upsets in the same MW and in different redundant paths will break the TMR protection**
- **Uncorrected upsets accumulate in MWs and can eventually break TMR**
- **Large MWs can be misguiding and will not provide the expected protection (too many bits in a MW)**
- **Strong mitigation has correction+masking**



Global TMR (GTMR) and Scrubbing

- GTMR only masks configuration upsets it does not correct configuration upsets
- We scrub to reduce accumulation in order to help protect the GTMR mitigation
- Scrubbing corrects the configuration memory
  - Does not reduce $dE_{Configuration}/dt$
  - Reduces the accumulation bit error rate
  - Does not correct functional upsets
- Scrub Rate (dC/dt) must be fast enough to beat accumulation
- Misperception #2:

Source: Xilinx Consortium:V4QV

Reported: $dC/dt > 10x(dE_{configuration}/dt)$

Problem: Only considers configuration, does not take into account mitigation strength; e.g., MW size and GTMR configuration masking

Do Not Scrub Configuration Memory of Non-Mitigated Designs

- **Scrubbing is a weak/secondary mitigation strategy:**
  - **It only protects against accumulation**
  - **There is no masking when only scrubbing configuration memory (masking needs mitigation)**
    - **If a utilized bit is hit – and its circuitry is active, then…**
    - **A functional error can occur with no time allowable to wait for a scrubber**
- **If the decision is made to use a non-mitigated design in a radiation environment, then the application is expected to have a high upset rate**
- **Do not add additional circuitry if it doesn't improve upset rates:**
  - **Power, area, Risk to project completion**

Scrubbers: Blind versus Read-back

Blind Scrubber

- Write golden configuration into configuration
- Scrub cycle in the order of *ms*
- Pros: simple, less area and power, no need for additional non-volatile memory, very fast (great for accelerated testing)
- Cons: Write pointer can get hit during writing and write bad data into configuration- however, insignificant probability of occurrence (proven in heavy ion SEU testing)

Read-back

- Read configuration, calculate correct data; if there is an upset, write corrected data.
- Scrub cycle in the order of *s*
- Pros, only writes if there is an upset
- Cons, additional non-volatile memory required; slow (only a problem for accelerated testing); takes more area and power; Correction scheme can break (e.g. be limited to detecting and correcting one upset) and consequently write bad data to configuration

Differentiate Scrubbing for Space Applications and Scrubbing for Radiation Testing

Space Application

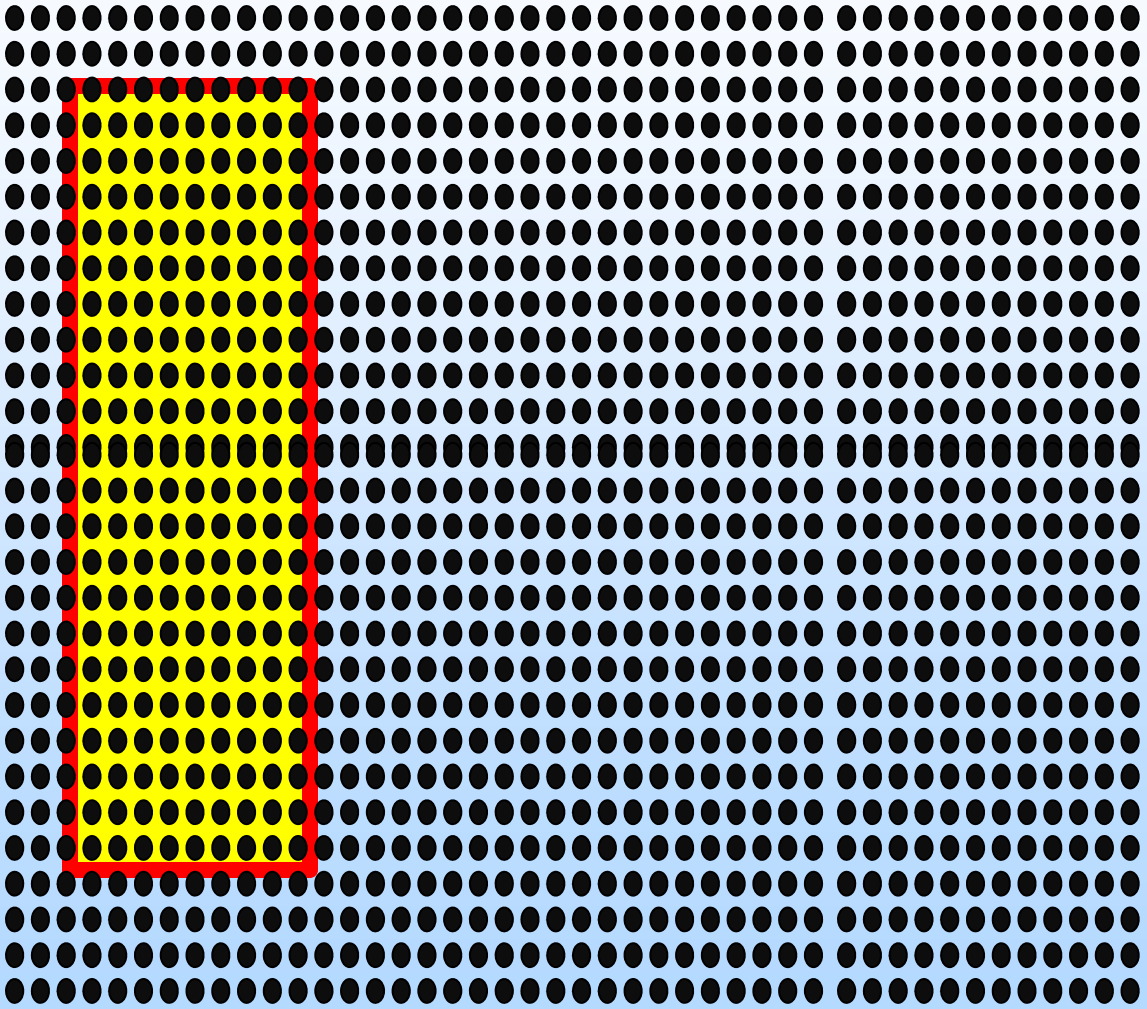
- Only scrub if there is mitigation
- Make scrubber simple to reduce project risk
- Do not scrub constantly – not necessary and not good for the system
- Single error correction double error detection (SECDED) scrubbers may not work well due to multiple bit upsets (MBUs)
- Blind scrubbing is the simplest scheme yet read-back will also work

Accelerated SEU Testing

- We must scrub!
- Particles cannot overtake the scrubber – i.e., scrubber must be fast enough to stop fast accumulation of configuration SEUs – SCRUB CONSTANTLY
- SECDED scrubbing schemes do not work well during accelerated testing because of MBUs and accumulation
- Generally no time for read-back of configuration – hence blind scrubber is the best fit for accelerated testing

Not All Configuration is used for a Design.

Probability for Used Configuration SEUs Bit Based on $dE_{configuration}/dt$



Total Number of configuration bits (NT)

#used bits is a fraction of NT

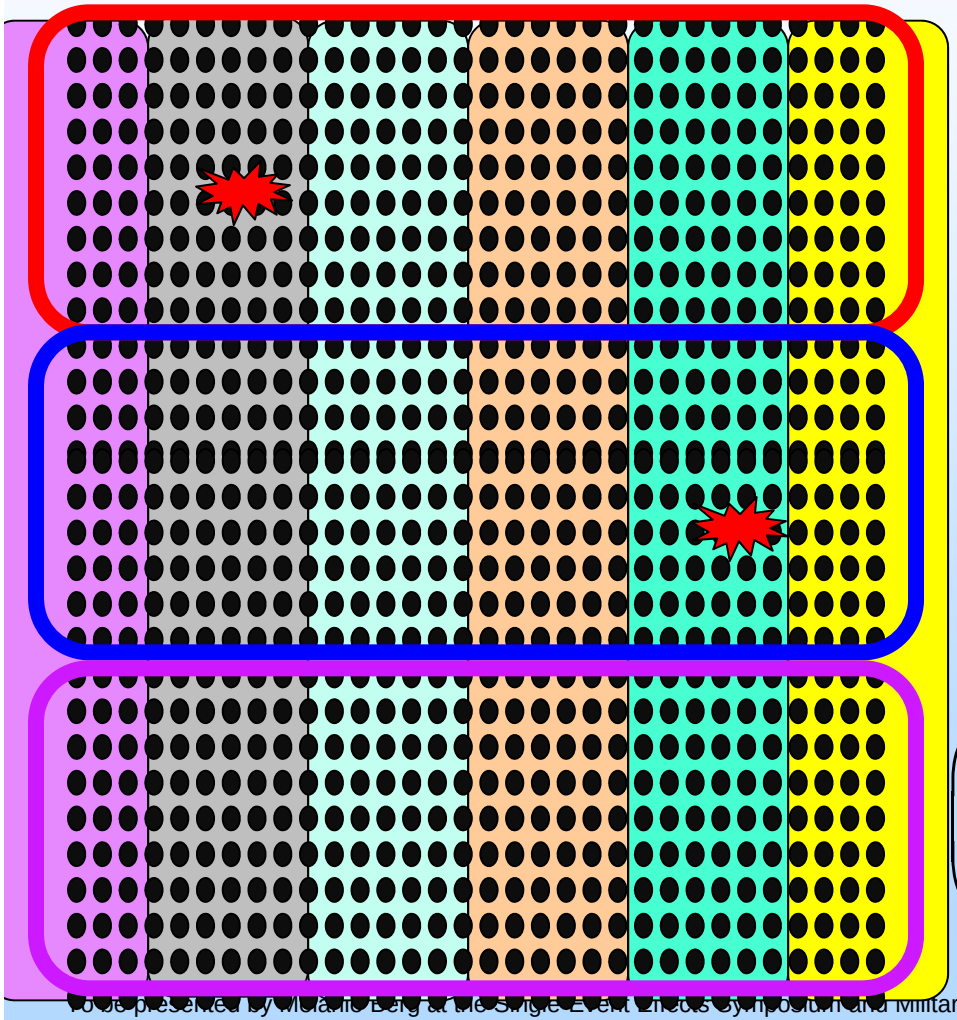
Event that one used bit is upset =

$$\left(\frac{dE_{configuration}}{dt}\right) * \left(\frac{\#usedbits}{NT}\right)$$

MWs Are Created out of Used Bits.

Probability that an SEU occurs in a MW Based on $dE_{configuration}/dt$

Used Configuration bits



1 MW: Divide the used bit space into 3 (crude estimate)

Broken mitigation if upsets are in 2 separate MWs

As the number of MW increases, the number of used bits per MW decreases

Event that an upset occurs in an MW =

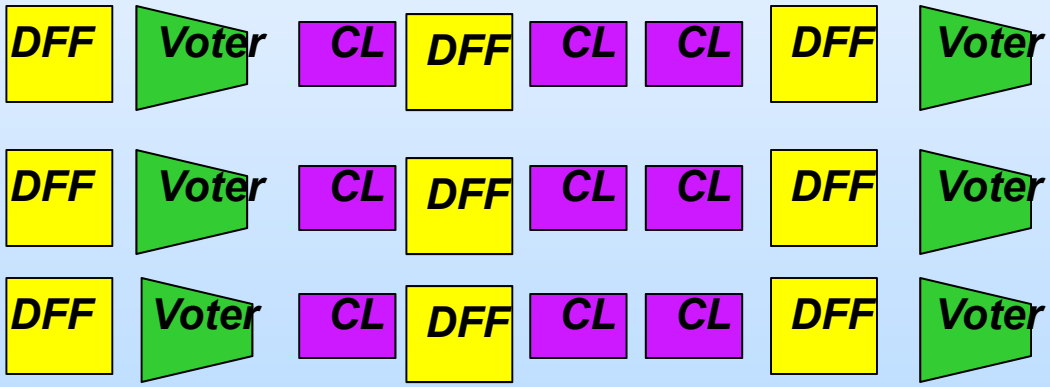
$$\left(\frac{dE_{configuration}}{dt}\right) * \left(\frac{\#usedbits}{NT}\right) * \left(\frac{1}{NMW}\right)$$

NMW = number of MWs

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/.

Scrub Rate Requirement Based on the Probability of Breaking the GTMR Mitigation:

$$\frac{dC}{dt} > \left(\frac{dE_{configuration}}{dt}\right)^2 * \left(\frac{\#usedbits}{NT}\right)^2 * 2/3 * \frac{1}{NMW} * \varphi$$



MW fan-out: a bit can be within multiple MWs

Makes a crude assumption that all MWs are the same size – but they are not. This is a rough estimate – but good enough

Scrub Rate Example for Accelerated Testing

Variable Definition	Variable	Example Number
Fraction of used bits per total number of bits	#usedbits/NT	0.1
Configuration Error rate	$dE_{configuration}/dt$	1000 bit-errors/device-s
Average fanout from MW	φ ; 1<φ<NMW	10
Number of GTMR MWs	NMW	5000
Fraction of bits in the same MW	$\left(\frac{\#usedbits}{NT}\right)*\frac{1}{NMW}$	2.0E-5

Calculated at Texas A&M with a flux of $1e^4$ and LET=26MeVcm²/mg

$$\frac{dC}{dt} > \left(\frac{dE_{configuration}}{dt}\right)^2 * \left(\frac{\#usedbits}{NT}\right)^2 * 2/3 * \frac{1}{NMW} * \varphi$$

$$\frac{dC}{dt} > 14 scrub/s$$

Scrub cycle must be in the order of ms; Blind scrubber works best

Scrub Rate Example for Space: Variation of Number of Mitigation Windows

$dE_{configuration}/dt$ = 4 bit errors/day; #userbits/NT = 0.1; φ = *10*

Number of MWs	dC/dt (scrub rate per day)	
1	1.07E+00	Once a day
10	1.07E-01	Once every 10 days
50	2.13E-02	Once every 50 days
500	2.13E-03	Once every 500 days
1000	1.07E-03	Once every 1000 days
5000	2.13E-04	Once every 5000 days

Accelerated $E_{configuration}/dt$ is $8.64x10^7$ times faster than the space environment $E_{configuration}/dt$…

Hence the required scrub rates are significantly reduced

Conclusion

- An argument is made to not use a scrubber with non-mitigated designs… risk reduction
- Scrub rates are determined by the configuration upset rate; number of used bits within an MW; and MW fan-out
- We differentiate between scrubbing in the accelerated test environment and the space environment. :
  - When operating in the accelerated test environment it is recommended to scrub as fast as possible in order to avoid unrealistic error signatures. Consequently, the blind-scrubber is optimal in a radiation test environment.
  - Contrary to common belief, we show that the required scrub rate of a mitigated design can be in the order of days. In this case the type of scrubber in inconsequential.