

Mitigation techniques for electronics in Single Event Upset environments



By Anthony Lai
[Aitech](#)

A variety of mitigation techniques are introduced herein to minimize Single Event Upsets (SEUs) in electronics and enhance the reliability of military and space applications. These techniques can be applied to all kinds of systems but are especially applicable in volatile and nonvolatile memory functions.

Electronics for space and military applications need to operate reliably in natural radiation environments. Although public perception typically equates radiation issues with space applications, observations and investigations by major semiconductor companies over the past 30 years have found that “harmless” particles like alpha particles and neutrons can have direct contributions to single event effects in semiconductors — even in airborne and ground-based applications. To minimize impacts from a variety of radiation-induced effects, mitigation techniques are implemented to enhance reliable mission operations.

What is a Single Event Upset?

A *SEU* is one of the categories of radiation-induced failures on semiconductors. There are two main types of SEUs: Single Event Functional Interrupts (SEFI) and recoverable upsets. A SEFI typically can hang a system or cause a microcircuit to enter into a “hung” state permanently, such as with the upset of a critical register in a processor, which then can only be cleared by a system reset or power cycle. A recoverable upset temporarily prevents a microcircuit from operating in its nominal state, but the microcircuit can recover over time without additional user intervention. In either case, there is no permanent destruction to any microcircuit.

System-level SEUs occur in common applications such as a laptop operated at sea level or as far-reaching as spacecraft systems operated outside of the atmosphere. There are a few key facts to know about SEU:

- . The higher the altitude, the more upsets - Many studies by major semiconductor manufacturers have shown many more SEUs occurring in systems deployed in high altitudes, like Denver, CO (approximately 5,277 feet above sea level) versus San Jose, CA (approximately 87 feet above sea level).
- . Most SEUs inside the atmosphere are caused by alpha particles and neutrons. Neutron abundance is the highest between 60,000 and 70,000 feet.
- . When the atmosphere reduces in density at a higher altitude, the dominant radiation will be caused by proton and heavy ion, and much more often, will be due to trapped particles around the Earth’s magnetic field.
- . In Low-Earth Orbit (LEO), 500 to 2000 km above the surface of the Earth, the density of protons and electrons dictates the frequency of SEU occurrences. Density varies both in altitude and inclination.

How to mitigate SEU

Several well-known and proven SEU mitigation techniques can be implemented by means of hardware logic and digital microcircuits. All of these techniques are required to work with existing semiconductors instead of forcing costly changes to the dies in low-volume markets in military and aerospace applications. Although some semiconductor vendors have introduced many radiation-hardened methods by design or process, these techniques are usually specific to the unique features of individual components, which can equate to extremely high component and subsequent integrated subsystems costs.

To keep the trade-offs of cost versus functionality in check, component-level SEU mitigation techniques prove beneficial to reduce radiation-induced failures. One common requirement to these techniques is the ability to perform a localized reset to the specific circuitry by an external reset mechanism, typically implemented with a SEU-tolerant FPGA. The list of component-level SEU mitigation techniques is shown in Table 1.

Function	Mitigation Method(s)
Processor	Triple Voting
Volatile Memory	Triple Voting Periodic Memory Scrubbing Error Correcting Coding (ECC) Memory Data and Address Buses Layout Physical separation of components for the same memory bank
Telemetry and Commands	Error Correcting Coding (ECC)
Non-Volatile Memory (Flash and EEPROM)	Error Correcting Coding (ECC) Memory Data and Address Buses Layout Physical separation of components for the same memory bank
Data Buffers	Triple Voting

Table 1

Triple voting

To maximize reliability and minimize high-energy particle upsets, three physically separated and independent memory banks or processors can be voted by a *triple voted controller*. In the event of a memory or a processor malfunction, the system can continue to operate by voting out the suspect memory bank or processor. When system resources become available, the suspect memory bank can be scrubbed at (a) specific memory location(s) and returned to nominal operations. In the case of triple voting processors, the three processors' data and instruction contents can be flushed to memory and reset, and all three units can continue operations. A typical triple voted memory function is depicted in Figure 1.

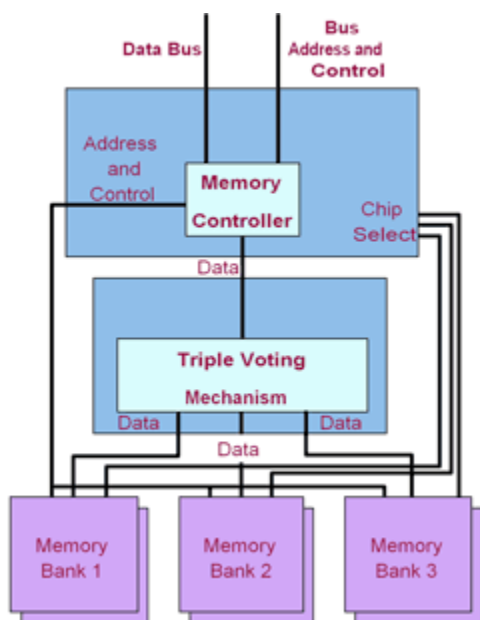


Figure 1

Triple voting is typically used for functional blocks with high clock rates of at least 66 MHz between a processor and banks of volatile memory. To maximize processor performance, this mechanism must allow for zero wait-state operation such that the processor memory fetch and write performance is not degraded for additional delays in implementing this voting methodology. From the processor perspective, triple voted memory is therefore achieved to perform the voting function within the same CPU read-memory request transaction; it is completely transparent to the processor. A typical triple voted controller is implemented in a radiation-tolerant FPGA and includes the memory data integrity logic and circuitry, which are needed to enhance system reliability in space environment applications. The digital logic for triple voting is shown in Figure 2, and the general procedure to operate a triple voting memory mechanism is shown in Figure 3.

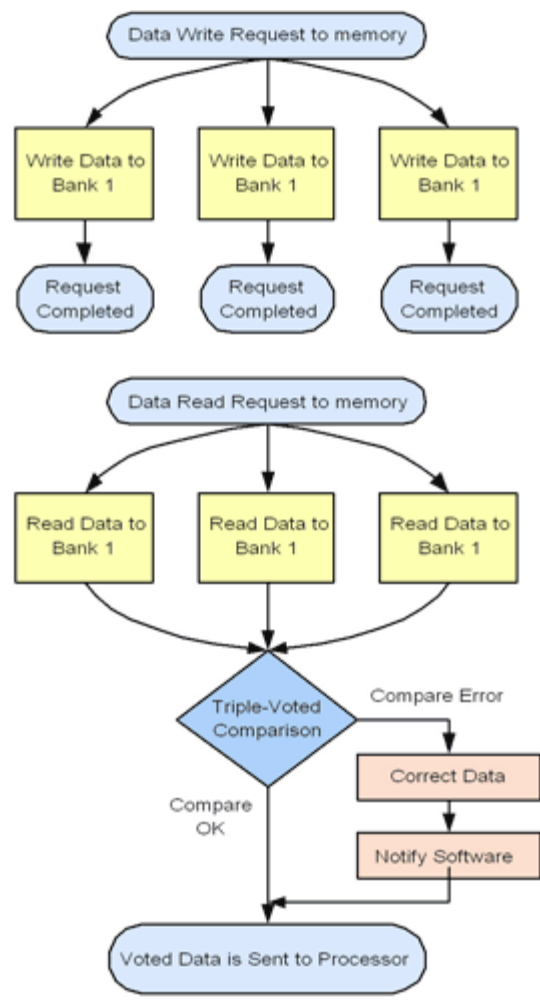
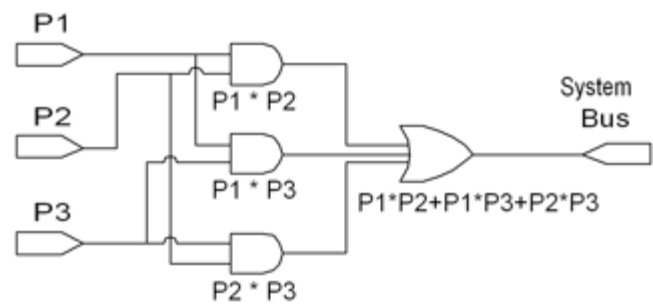


Figure 3

- . The data bus is connected to the voting system, which drives the data bus of the three memory banks.
- . In the write cycle, the voting system gets the data from the memory controller and drives it directly to the three memory banks simultaneously for a write operation.
- . In the read cycle, the voting system receives the data from the three memory banks.
- . The system uses a triple voted logic to decide the right value by the majority vote.
- . When an error is detected, the memory controller FPGA captures the voted memory address and asserts an interrupt so that the corrected data can be rewritten to the memory address.

Triple voting has also been demonstrated as a feasible solution to mitigate SEUs for a set of three identical processors performing the same instructions at the same time in a single board computer. The concept of triple voting has been fielded in a triplex subsystem configuration for a variety of aerospace avionics subsystems such as the vehicle management computers in NASA's space shuttle. However, in this article, we focus solely on board and component-level mitigation techniques.

Periodic memory scrubbing

“Bit flipped” is a common observation when an SEU takes place in memory contents. Bit flip can cause unexpected behaviors to software or application data, and this phenomenon will require additional attention by peripheral software or hardware. One of the methods to minimize the probability of such failure is achieved by performing a periodic scrubbing or refreshing of data contents in the memory location(s). From the data collected in various space missions, the frequency to perform scrubbing has a direct correlation to the reliability of memory. In general, reliability reduces when the time interval between two consecutive memory-scrubbing events increases. The data noted in Table 2 is collected from the Advanced Research and Global Observations Satellite (ARGOS) LEO space mission.

Note: A reliability value of 100 percent implies no error during the specified time interval.

Scrubbing Interval	Reliability	
	EDAC Scheme 1	EDAC Scheme 2
10 minutes	93.5506%	99.9999%
20 minutes	93.5504%	99.9998%
30 minutes	93.5503%	99.9997%
40 minutes	93.5502%	99.9996%
1 day	93.5319%	99.9862%

Table 2

Error Correcting Coding (ECC)

Nonvolatile memory devices with slower access timing such as flash and EEPROM are typically protected by an ECC mechanism allowing single-bit detection/correction and multi-bit error detection. In some cases, “super” error correcting code can provide two-bit correction as well. A popular topology to implement ECC is to store ECC syndromes calculated during the writing or programming operations to an additional flash device for each bank of memory. A reference diagram of this topology is shown in Figure 4.

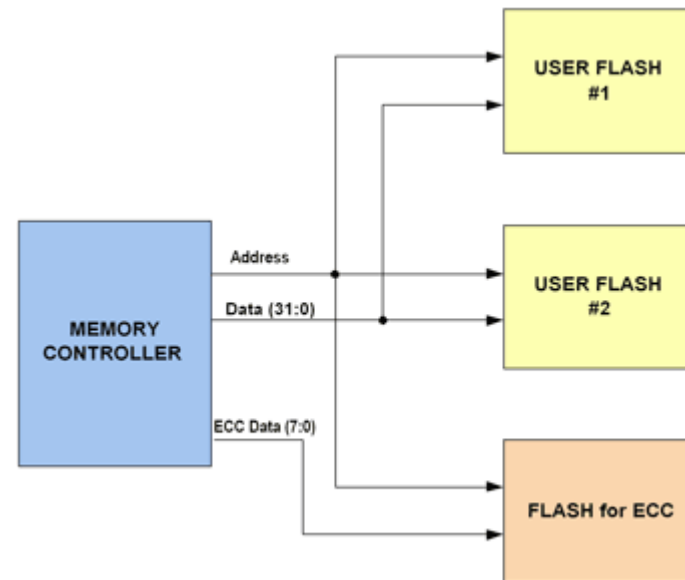
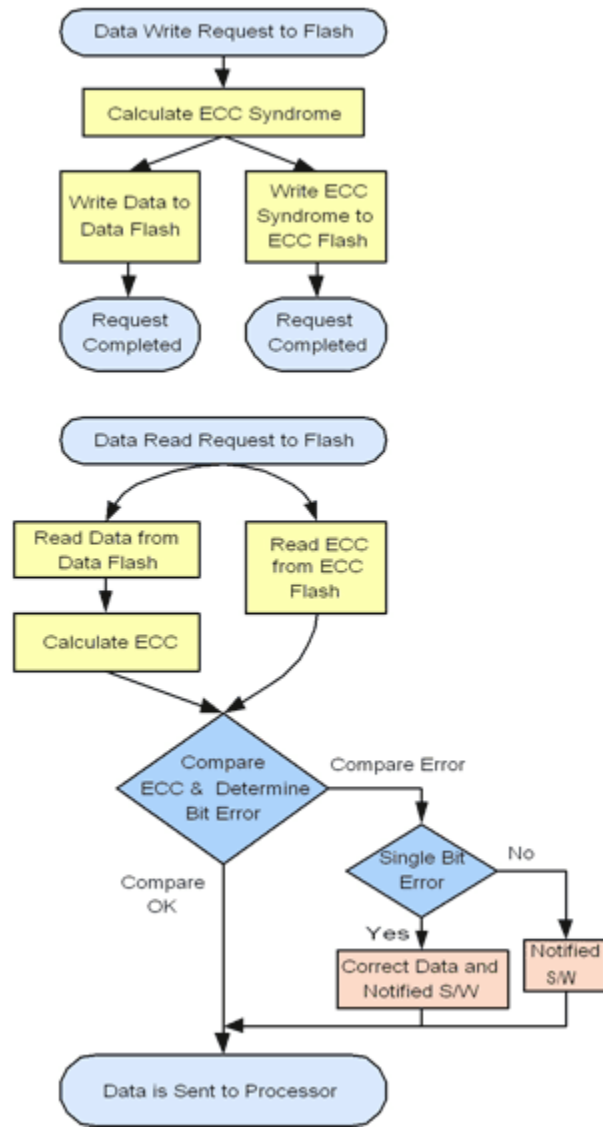


Figure 4

A 32-bit CRC (Cyclic Redundancy Checksum) is calculated when data is written during every write access to the nonvolatile memory. When data is requested from the memory during a read operation, a checksum will be calculated again and compared against the value stored in the additional device. The processor-transparent hardware implementation of an ECC mechanism can be realized in an FPGA, a custom IC, or in software. Because software-implemented ECC also requires SEU-prone memory to execute such instructions, hardware-implemented ECC can usually achieve much higher reliability and higher performance than a similar software-based solution.

Figure 5 shows a summary of a typical ECC algorithm for nonvolatile memory.



- . During a write request, the ECC check word is generated and programmed or written to an additional ECC flash device.
- . During a read request, the ECC check word is generated again for the read data.
- . At the end of a read request, the ECC check word generated in the write request (located in the ECC flash device) is compared with the ECC check word generated in the read request.

There are three possible cases or actions after each ECC check word is compared:

- . ECC words match - The data read from the device is sent to the requestor, such as the processor. No action is needed.
- . ECC words don't match (one-bit error) - Correct the data that was read and drive the corrected data to the requestor. Drive interrupt indicates one error and enables the software to write the corrected data back to the memory devices.
- . ECC words don't match (multi-bit errors) - No correction; the data read from the user flash is driven directly to the requestor. Drive interrupt indicates multi-bit errors to the software.

Along with a CRC 32-bit checksum, other coding schemes can also be used. In one exercise, a software implementation of an error-correction mechanism with four popular coding schemes was investigated. The overhead tradeoff in terms of calculation time and potential throughput efficiency is summarized in Table 3.

Scheme	Program Size (bytes)	Check-bit Overhead = check-bit/data (words)	Decoding Speed (MBytes/s)	Detection/Correction Capability
Hamming	14,307	8/64=12.5%	187.8	bit-slice single error correct and double error detection per block

Cyclic	6,731	8/64=12.5%	29.24	bit-slice single error correct and double error detection per block
Parity	6,747	2/32=6.25%	34.68	single error correct and double error detection per block
RS (d=3)	6,723	2/64=3.125%*	24.41	single bit error correct and multi-bit error detection per block

Table 3

In addition to the selection of an error coding scheme, the implementation by means of hardware or software also has direct impact to overall reliability. The chart in Figure 6 shows a normalized reliability prediction of using the same error scheme for both software and hardware solutions. The hardware implementation indicates a minimal drop-off in reliability experienced over the hours of testing. However, the software implementation shows a significant degradation in reliability over time.

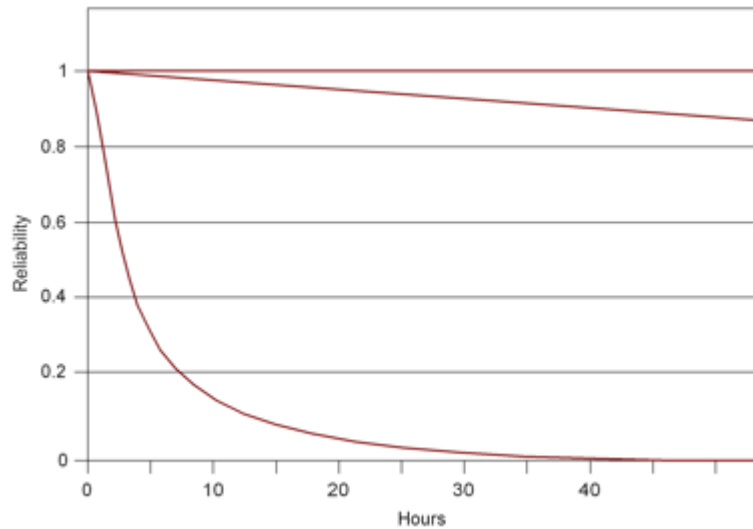


Figure 6

Physical layout considerations

In order to minimize SEU, other board-level design practices can be applied to gain reliability. Two typical practices are in the layout of data and address buses and in the physical separation of components for the same function, like a memory bank. For the case of triple voting, the three triple-modular redundant devices shall be placed as far away from each other as possible on a physical layout, such that a localized SEU effort will not affect more than one of the three redundant components at any given time. The likelihood that two out of three components will exhibit the same SEU phenomenon during exposure to the same environment is almost nonexistent.

Selection of devices to build a memory bank topology can also have a direct impact on the quality of an error-correcting mechanism. Figures 7 and 8 show two different topologies, both of which can be used to implement 512 KB memory. In the case of Figure 7, the best correcting scheme (not shown in the diagram) can only correct 1 bit for every 8 bits of data. In the case of Figure 8, a possible error corrected for 2 bits per byte is possible by separating the upper and lower 4 bits of data into two separate devices.

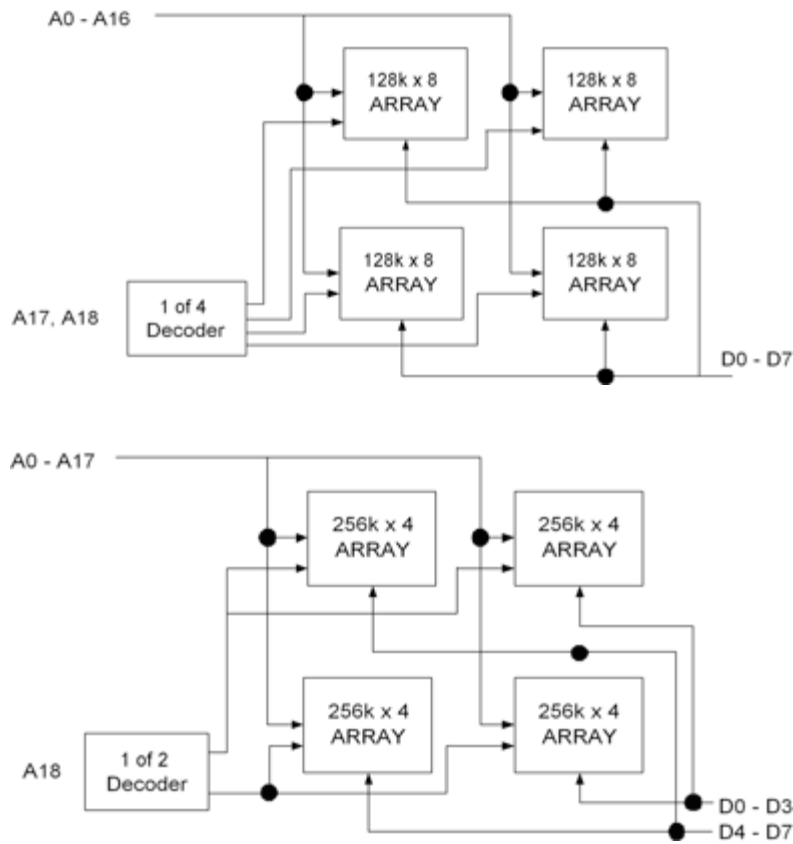


Figure 8

Reliability counts, but cost *is* the object

Ideally, no one should ever be faced with potentially compromising a subsystem's reliability for cost, but reality usually sets in, of course, unless money is no object. However, in real life, a system designer must continually make design trade-offs and compromises in terms of subsystem performance, functionality, reliability, and costs, with seemingly continual end-customer pressure to decrease development and deployment times. Sometimes, the best all-around and most cost-effective path to success is to choose a path already taken.

There is a wide variety of SEU mitigation schemes that can serve as alternatives to reduce transient radiation effects and allow a system to work through the end-application's environment and complete the mission successfully.

A list of references is included with this article to provide additional details into these relevant subjects.

.....

Anthony Lai is Space Product Business Development Manager for Aitech. He has more than 15 years of experience in space avionics design and development for radiation-tolerant computer products. Prior to joining Aitech, he led several avionics payload designs at Jet Propulsion Laboratory, including the Mars 2003 Rover prototype FIDO. Anthony has an MS in Electrical Engineering from UCLA as well as a BS in Electrical and Computer Engineering with a minor in Applied Mathematics from UC Irvine.

For further information, go to the Aitech website at www.rugged.com.

Additional reading:

- [1] S. Mitra, E. McCluskey, "Word-Voter: A New Voter Design for Triple Modular Redundant Systems," Center for Reliable Computing at Stanford University, 2000.
- [2] C.I. Underwood, "The Single-Event-Effect Behavior of Commercial-Off-The-Shelf Memory Devices - A Decade in Low-Earth Orbit," Proc. 4th European Conf. on Radiation and Its Effects on Components and Systems, pp. 251-8, Palm Beach, Cannes, France, Sep. 1997.
- [3] P. Shirvani, "Fault-Tolerant Computing for Radiation Environment," Center for Reliable Computing at Stanford University, 1997.
- [4] G.-C. Yang, et al., "Reliability of Semiconductor RAMs with Soft-Error Scrubbing Techniques," IEE Proc. - Computers and Digital Techniques, Vol. 142, No. 5, pp. 337-44, Sept. 1995.
- [5] R.M. Goodman, et al., "The Reliability of Semiconductor RAM Memories with On-Chip Error-Correction Coding," IEEE Trans. on Information Theory, Vol. 37, No. 3, pp. 884-96, May 1991.

- [6] A.M. Saleh, et al., "Reliability of Scrubbing Recovery Techniques for Memory Systems," IEEE Trans. on Reliability, Vol. 39, No. 1, pp. 114-22, April 1990.
- [7] Johnson, B.W., "Design and Analysis of Fault Tolerant Digital Systems," Addison-Wesley, Second edition, 1989.
- [8] Ricketts, L.W., "Fundamental of Nuclear Hardening of Electronic Equipment.," Wiley-Interscience, 1972.



©MMVI *Military Embedded Systems*. An [OpenSystems Publishing, LLC](#) publication.

Related Websites:

[Embedded-Computing.com](#) | [PC/104 and Small Form Factors](#) | [VMEbus Systems](#) | [DSP-FPGA.com](#)
[PXI, Test and Measurement](#) | [CompactPCI and AdvancedTCA Systems](#) | [Industrial Embedded Systems](#)

[About this Magazine and Website](#) | [Contact Us](#) | [Military Embedded Systems Media Kits](#)