SEEs

*Introduction*

A natural consequence of data acquisition in high energy environments is a significant increase in single event effects (SEEs). SEEs are not a new problem in electronics but have been recorded as anomalies in electric equipment in nuclear testing from as early as 1954. Additional instances of these apparently random errors and interruptions in otherwise perfectly operating ICs were observed in space electronics. The first paper describing this phenomenon wasn’t published until 1975.

SEEs are a phenomenon resulting from interactions and collisions of high energy particles with devices in integrated circuits. When a high energy particle passes through the silicon substrate of a device it generates charged particles along its path through a series of collisions. If the charges are generated at or near a transistor junction, the new charge can induce an upset in the transistor resulting in a change of stat,e usually manifesting in a memory bit flip, or a sudden spike in voltage or current. The particle itself can be charged, but usually it is an uncharged particle, such as a neutron, which only begins generating this ionizing path after collision with a doped substrate.

A picture containing diagram

Description automatically generated

Figure 1: High Energy Neutron generating an ionizing path after collision.

These high energy particles are present nearly everywhere, however there are specific environments and sources where they become a genuine reliability concern. Galactic cosmic rays (GCR) from space are the most common source. GCR are made up of subatomic particles and light ions traveling at nearly light speed. While they might not necessarily directly strike ICs, they generate high energy neutrons through nuclear spallation resulting in air showers. If these neutrons retain energy, generally greater than 10 MeV, they can induce an SEE. The density or flux of these Neutrons is dependent both on altitude and longitude. For example, a New Yorker would experience twice the neutron flux as someone in Singapore, whereas airplane passengers would experience 600 times the neutron flux the New Yorker would. Given that altitude has the greatest influence on the probability of an SEE occurring, mitigation and recovery is usually considered in space and aviation applications. Although the devices in the LHC are contained and shielded from atmospheric effects, the collisions the LHC is emulating generate an equivalent environment.

Single event effects are really an umbrella term for several possible errors that can occur, categorized as soft and hard errors. Hard errors are errors that cause lasting or permanent damage, and generally can’t be solved using logical techniques and will not be a focus of this paper. Soft errors are upsets to a device’s operation but are self-correcting in time or are correctable. These can generally be described with two events: upsets and transients.

Diagram

Description automatically generated

Figure : SEE Categorizations

Single event upsets (SEUs) occur when high energy particles impact transistors in memory cells, which result in an immediate bit flip. A single bit flipped is referred to as a SBU (Single Bit Upset) and if multiple bits are affected it is referred to as a MBU (Multiple Bits Upset). If the bitflip occurs in a Datapath that otherwise doesn’t influence any system state registers, the flip propagates but ultimately is flushed out. However, if the bitflip modifies a state register, such as an FSM state, a control counter or another register that doesn’t naturally flush its contents, the effects can be more severe up to requiring a system reset. Even worse, in FPGAs, is when the bitflip occurs within the configuration memory, resulting in LUT contents or routing to change, referred to as routing errors.

Single event transients (SETs) result when high-energy particles impact a combinatorial path of an IC and result in voltage or current spikes on a wire. There are two instances where this results in an error. The first is when the transient occurs on any data or signal line leading to a clocking element. If the pulse-width of the spike is wide enough and happens at the right time, the glitch can propagate through the circuit. The second is when the glitch appears on critical system signals such as a clock or a reset. Unless the signal trees for these are built to resist errors like this, SETs can result in extra transitions unintended system resets. While transients don’t directly influence memory cells the way SEEs do, their influence on data as well as on critical system wires can result in similar effects to SEUs or worse. When the consequences of either of the above noticeably freeze the regular function of the system, they are further categorized as SEFIs (Single Event Functional Interrupts).

*Characterization*

If it can be measured, it can be managed. Characterizing SEEs happens at the physical level and the functional level. An SEE is inherently a physical phenomenon and for that reason in order to recreate and emulate it, understanding and replicating the physics behind what causes the SEEs is important to understand its effects and the efficacy of mitigation methods. TODO: FINISH THIS PARGRAPH

At higher levels of abstraction, we’re more in measurements of interruption and data loss, which aren’t unique to SEEs. These might include real time values like MTBFs (mean time between failures), MDT (mean down time) or FIT (failures in time), the last of which is representative of the number of failures in 10^9 device hours. At a more granular level we’d also be interest in the mean amount of data lost per interrupt or the % amount of corrupt or lost data to valid data.

Box and whisker chart

Description automatically generated with medium confidence

Figure : TBF (time between failures)

All that considered, SEEs aren’t unstoppable and when they are they can often be corrected. Solutions, mitigation and detection and recovery schemes exist to fit a variety of failures but come with their own tradeoffs. In mitigation or prevention systems the difference in FIT rates and MTBFs is a standard measure of efficacy. In recovery, detection time of a failure as well as the following recovery time are the primary metrics. Both however also consider the costs of their implementation, including device performance and implications because of these added complexities, which often result in device resource, time to market and monetary costs.

*Mitigation, Detection and Correction: TMR*

The simplest and most implemented mitigation method implemented is TMR (triple modular redundancy). TMR is a fault tolerance methodology which duplicates system units in anticipation of one failing. By having three duplicate copies of a component, usually a DFF or other memory storage, and a voter circuit, which just forwards whichever value appears twice or more on its input lines (see Figure 4). If any single DFF encounters a bit flip or other upset, the remaining two will hold majority and will outweigh the single corrupt DFF. The corrupted DFF can eventually expect to be rewritten or refreshed to the correct value and will no longer be in contention with the other two.

Diagram

Description automatically generated

Figure : Generic TMR (triple modular redundancy)

This version of TMR resolves SEUs, but SETs remain a threat. Since all three registers receive the same input and latch onto it at the same time, an SET can simultaneously corrupt all three of them. A relatively straight forward solution to this is just to add buffers or delays onto the clock input line of the DFFs. By setting the difference between each clock’s arrival time to some time *t* (15ps in Figure 5), the TMR would be able to mitigate any SET corruptions with glitch lengths of less than *t*.

Diagram, schematic

Description automatically generated

Figure : TMR with delayed clocks to mitigate SET interrupts

In circumstances when the DFFs are updated infrequently the TMR described above can simply delay a failure rather than mitigate it. In critical memory, failures of any kind, delayed or not, are unacceptable and so correction within the TMR registers is sometimes necessary. By adding a feedback loop (Figure 6) from the voter output back into DFF input, the system can ensure not just mitigation but also correction, assuming only a single bit flips at a time.

Diagram

Description automatically generated

Figure : TMR with voter feedback for correction

Further measures could be added to improve reliability in TMR. Spacing the TMR components out on a chip is important to keep two or all the DFFs from flipping through an MBU. Spaced components also means that high fanout signals like clock and reset are separated through buffers, which would stop simultaneous upsets as well. Alternatively, the three components can be reduced to in with DMR (dual module redundancy) so that while SEEs aren’t mitigated they are still detected. Regardless, the basic TMR and its variations all come with a cost in resource utilization on the device. In resource or power-hungry applications this may be beyond the constraints or would require larger and more expensive devices.

*Mitigation, Detection and Correction: ECC Memory and Scrubbing*

TMR is useful in ASICs as well as the configured logic of FPGAs, however it becomes prohibitively expensive when applying it to the configuration memory of an FPGA. When it comes to large and system critical memories, duplicating and running each bit or word through additional logic to read it out becomes cumbersome. For these systems, ECC (error correcting code) is utilized to provide upset detection and correction to memory blocks.

ECC memory is a data storage that uses extra bits and logic to detect and correct memory corruption. This data is stored into memory the same way that data is often transmitted in communication, so that the sender, the device writing to memory, and the receiver, the same device reading from memory, can reliably expect the data to be read out correctly, or if not then to be alerted of an error and respond appropriately. ECCs come in many formats, with the most known utilizing Hamming code.

Hamming code works by allocating specific bits to be parity for different sections of the data word. TODO: FINISH THIS PARGRAPH/SECTION

While configuration memory ECC and TMR are the most common methods of mitigating SEEs, other methods have been utilized for various applications as well. Watchdog timers are broadly useful with SEEs and other functional disrupts. This timer is set up to force a system reset once it expires, but has a reset built in that will restart the timer provided some checkpoint is regularly reached by the device logic. Watchdog timers add small amounts of complexity but tend to have a fixed high upper bound for error detection and an inelegant method of correction. Safe state machines are a passive recovery scheme which utilizes FSM transitions to safely guide control state along a recovery path when illegal states or data inputs are read.

*Application to the LHC DAQ*

There are instances where none of the above are applicable, reliable or perform to par with what a system requires. In these circumstances, application specific detection, correction and or mitigation systems must be developed from the ground up. Such is the case in DAQ systems of the LHC.

Unlike the RD53 pixel readout chip mixed in with the action of the LHC, the DAQs enjoy a safer, less chaotic environment hidden behind radiation insolation. However, they don’t go unaffected from SEEs. While SEEs aren’t expected to occur within the DAQ, it is expected to occasionally receive bad data from the readout chip which will be constantly bombarded by high energy particles. The pixel readout chip is directly connected to the DAQs and passes serial data across parallel lines with minimal overhead to maximize the data throughput to accommodate the increase in luminosity expected at the LHC.

The DAQ reads bits and organizes them into distinct data blocks with a header to distinguish individual blocks. However, since these headers can only be in a particular format, the DAQ will realize that an issue has occurred once it has seen sufficient invalid headers. These invalid headers will be caused by SEEs on the readout chips end and will generally manifest in bit flips within the data block, bits added into the data block and the counterpart bits or chunks removed from the data block. Since the data is coming in from a remote location, TMR is irrelevant, as are most other SEE mitigation techniques which check whether interrupts occurred in a particular system. ECC checking on the transmitted bits sounds attractive, but since the data is sent as a stream and bits can be added or removed ECC schemes would be unable to recover.

*YARR Tx/Rx*

To understand the SEE detection and recovery system, an understanding of the logic and data surrounding the transmitting end of the RD53 readout chip and receiving end of a DAQ is necessary…