Introduction
A soft error in the context of this article, can be defined as an unintended change in electrical state of a device or circuit that has an origin, external to the system’s designed inputs and outputs. A “soft” error is one which causes no direct permanent damage to the systems components, such that the unintended system behavior can be corrected with some form of “re-set”.

For real time systems, such as automated car navigation, biological assisting devices or commercial data centers, a soft error, whilst not permanently damaging the electronics, can have dangerous consequences if the error is not detected and corrected in real time.

It is important, therefore, that the rate of these software “Failures In Time” (often referred to as the “FIT” rate), is fully characterized for critical systems. An unfortunate consequence resulting from device dimensional scaling to smaller geometries, is that all other things being equal, the soft error rate performance of any given circuit greatly deteriorates as devices shrink in size. Consequently, the most advanced process nodes, are those most at risk for such failures, which is why this topic is of increasing importance.

The “Failures In Time” or FIT rate is usually expressed as the number of device failures per one billion hours of operation. This may at first appear to be something that is very unlikely to ever happen, since one billion hours is over 100,000 years. However, a circuit need only consist of ten million devices, and an average FIT rate per device of unity, translates to the circuit suffering a soft error rate once every 100 hours of operation, or once every 4 days. For a number of applications, this error rate, if uncorrected, would be unacceptable. One error per year would be a more acceptable number.

The Source of Soft Errors and the Effect of Device Scaling
The usual source of soft errors, is an energetic ionizing particle or photon, such as an electron, proton, neutron, gamma or X-rays etc. The origin of these particles (from now on, “particles” will also include high energy photons for the sake of brevity), is usually galactic in nature (Cosmic Rays), but can also be locally caused by radio-active decay of materials, such as lead based solder.

A primary energetic particle from a galactic source, can form a cascade of other particles when interacting with the earth’s atmosphere, as schematically depicted in Figure 1. The important take away is that the flux of these energetic particles at ground level, is inversely related to their remaining energy. In other words, being...
struck by a high energy particle is a rare occurrence, but the flux rate of incoming low energy particles, is very high, as shown in Figure 2. There is also a strong elevation effect on the flux rate of incoming particles. For example, at sea level in New York, the incoming neutron flux rate is approximately $13 \text{ cm}^{-2} \text{h}^{-1}$, whereas at 10,000 feet, this increases more than an order of magnitude to approximately $144 \text{ cm}^{-2} \text{h}^{-1}$.

The issue with shrinking the geometry of active devices, is that it takes a lower deposited charge, and therefore a lower energy particle, to upset the circuit. The resultant effect is that smaller active devices have a strong tendency to get upset more often, compared to larger geometry devices, as can be inferred from the graph of particle flux versus particle energy in Figure 2.

In addition to charge tracks being caused directly by the primary incoming ion, the incoming particle can create additional charge tracks by way of nuclear reactions with the constituent atoms of the device. This phenomenon is often referred to as “spallation”. Both neutrons and protons can interact with nuclei near the sensitive region of the device, spawning charge tracks in addition to the primary track.

Various neutron interactions with Silicon 28, can yield secondary “fission” reaction tracks of neutrons, protons, alpha and beta particles, converting Silicon 28 atoms into magnesium or aluminum in the process. Other incoming neutron reactions with Boron 10 dopant, can yield a high energy Lithium ion and an alpha particle track, both in the 1MeV range as “fission” products. The Boron 10 in BPSG layers being a primary source of soft upsets from these nuclear reactions. It is for this reason that it is important to use high purity boron sources in BPSG to remove the boron 10 isotope responsible for these additional nuclear reactions.

Other nuclear reactions with incoming protons, convert Silicon 28 into Magnesium, Aluminum, Carbon, Oxygen or Sodium, with alpha and/or other protons and sometimes gamma rays as secondary products and particle tracks.

### High Energy Particle Strike Effects on Devices and Circuits

When a particle strikes an electronic device, the path of the strike becomes a narrow track of positively and negatively charged ions. In the semiconductor, this ionization event creates a charge track that will consist of mobile electrons and holes, as shown in Figure 3. The initial density of these charged particles is described by the quantity known as “Linear Energy Transfer” or LET, which depends on the actual particle species, it's energy and the material properties in which the charge track is formed (bandgap, density etc.)

There are two parts of the ionized track that are of interest in the semiconductor.

1. Electrons and holes that are in the immediate vicinity of an electric field, often near metallurgical junctions, will quickly separate, and “drift” to the active region, creating a current pulse that is usually in the pico-second time frame.

2. Electrons and holes that are close to, but not in, a high field region, can slowly and randomly diffuse into the drift region over time. Since charges in the low field “diffusion” regions move much slower, the current pulse from this part of the charge track will be much lower in magnitude but longer in time, usually in the nanoseconds time frame. This is simply a reflection of the equation:

   \[
   \text{Current} = \frac{\text{Charge}}{\text{Time}}
   \]

   ![Figure 3. Schematic representation of the charge track generated by a high energy particle strike.](image)
If a MOSFET is biased in its “off” (non conducting) condition, the two separate phenomena of track charge arriving via the fast “drift” then slow “diffusion” processes, are clearly seen in the device drain current versus time response curve after a high energy particle strike, as shown in Figure 4.

In a real circuit, such as in an inverter configuration, the drain voltage is not held constant, but will be effected by other circuit elements. In this case, the drain current response to the high energy particle strike, can be more complicated than that shown in Figure 4.

A further possible circuit response complication to being struck by a high energy particle, is that the device may not be held at a static bias, but be part of a clocked circuit. In this case, the response of an individual device being struck, could cause a cascade of corrupt logical responses. Figure 5 shows the response simulation of a simple clocked “D” type flip flop, to a high energy particle strike. The simulations shows that corrupt outputs occur for several clock cycles before the circuit corrects itself.

In order to fully characterize a circuit response to a high energy particle strikes, it is important to realize that the response of the device to the strike also depends on the incoming angle of the particle. Generally, a strike can occur at any angle. So in theory, full circuit characterization, requires the simulation of a matrix of angles at all possible strike locations and energies for different “on” an “off” circuit states, which quickly becomes a large design of experiments.

The reason the incoming angle can have a strong effect on device response, not withstanding the simple geometric effects, is that the separating electric field and the direction of the track can be in different directions. Consider a device with a vertical electric field. If the particle strike is also vertical, the separation of the charges largely occurs within the track itself. However, in a horizontal strike, the charge separation field is at right angles to the strike, so the field effectively tries to rip the charge track apart. In this case, if the track is near the horizontal channel, the charges can have a shorter average path length to reach the channel.

What the Circuit Designer Needs to Know

Since the surface area of most chips is much greater in linear dimensions than it’s electrically active depth, the most practical metric that a circuit designer needs to know, is the vulnerable cross sectional area of the chip that is sensitive to an incoming particle strike from any angle, versus the energy of that particle expressed in terms of Linear Energy Transfer (LET).

Once the designer knows the relative or absolute area of the chip that is sensitive to upsets, he/she can now look up the flux rate of ions at this energy in the working environment of the chip, multiply this flux by the sensitive area, and determine the number of failures in time (the FIT rate) of that chip for a particular particle energy. This exercise can then be repeated for all expected ion strike energies, to arrive at the overall FIT rate of the circuit in any given environment. In summary, therefore, what needs to be simulated is a matrix of all strike incoming locations, at all angles, versus all expected in-coming particle energies (LET).

In general, for a given process node or geometry, there will be a minimum particle energy, expressed in Linear Energy Transfer (LET) terms, required to cause any upset at all, at any incoming angle. Above this minimum particle energy (LET), the sensitivity of the circuit to an upset, usually rises rapidly, and then saturates, as shown in Figure 6. The saturation of the sensitive cross sectional area is a result of the finite diameter of the charged track created by the incoming ion, such that a strike that occurs in a non active area will not affect the circuit. Also not all strikes that do occur within an active area, create an upset, as transistors that were already in a conducting state at the time, are usually not effected by a strike that would have caused an error if that same device was in a non conducting state.
This curve of a circuit’s sensitive area (cross section) versus incoming particle energy, expressed in units of Linear Energy Transfer (LET), can be approximated by a “Weibull function”. The circuit’s sensitive cross sectional area for at least four relevant particle energies is required for a good fit. The “Weibull” curve is given by equation 1.

\[
\alpha(LET) = \alpha_{SAT} \left[1 - \exp\left(-\frac{(LET-LET_0)}{W}\right)^S\right]
\]

Where:
- \(\alpha(LET)\) is the cross sectional area at LET.
- \(\alpha_{SAT}\) is the saturated area.
- \(LET_0\) is the threshold LET.
- \(W\) is the width of the rising portion of the curve.
- \(S\) is the power that determines shape.

Equation 1: The Weibull curve used to fit a circuit’s upset sensitive area (it’s cross section), to incoming particle energy.

Figure 7 shows the calculated Weibull curve for a saturated cross sectional area of 1e-5cm², a threshold LET of 20 MeV-cm²/mg, an LET width of 20 MeV-cm²/mg and with the curve fitting parameter, “S” of unity. These parameters were chosen to approximate the measured curve shown in Figure 6.

Clearly, the most accurate characterization of a cell, is to simulate all strike locations, angles and energies for all particle energies as described above. However, for simplicity, often this capture cross sectional area curve versus particle LET, shown in Figure 6, is approximated to a step function, as shown by the “Ideal” curve in that same figure.

Figure 6. Chip cross sectional area that is sensitive to upset, versus incoming particle energy. The actual response is often simplified to a step function when volumetric FIT rate calculation methods are used instead of areal methods.

Since Linear Energy Transfer (LET) is a measure of charge deposited per unit length, if you choose the “critical charge within a sensitive volume” method to determine upset rates, the implication is that the charge deposited for a given LET, depends on the track length of the particle through the critical volume of the device. The particle track length, and therefore the deposited charge, is dependent on the angle of the particle through the critical volume and the entry point of the strike relative to the critical volume.

A simple geometric integration, of all incoming angles versus path lengths in this “critical volume” for each particle flux at each energy, yields the “Failures In Time” number that the circuit designer needs to know. However, now we have moved from needing to calculate the critical “surface area” of the chip versus energy, to needing to calculate the critical “volume” of the chip versus energy, which now requires knowledge of the maximum sensitive depth of the strike which has a material effect on the circuit response. A further simplification often made using the “critical charge” calculation method, is an assumption that this “critical volume” is also not dependent on incoming particle energy. So there are pros and cons to using “areal” versus “volumetric” methods to calculate the FIT rate of a circuit.

A simplified example of the “volumetric critical charge” approach to calculating the Failures In Time (FIT) rate, is shown in Figure 8. For simplicity, we shall use a circuit that has a critical surface area of 1um x 1um, that is exposed to an environment of 8 particles per second per um². Of those 8 particles per second per um², 50% of them have an LET of 50pC/um of track length, and 50% of them have an LET of 25pC/um of track length. Let us also assume from simulations or measurements that we have determined that the minimum (critical) charge required to cause an upset in our cell is 50pC of charge.

Now in order to calculate the number of particles with a total charge greater than the critical charge required to cause an upset in the cell, we have to integrate all pos-
Suppose: Sensitive Collection Volume with Surface Area of = 1μm²

Within this Sensitive Volume:
- 100% of all chord lengths are > 0.5um
- 50% of all chord lengths are > 1.0um
- 25% of all chord lengths are > 2.0um
- 0% of all chord lengths are > 4.0um

Figure 8. An example of a sensitive volume method of calculating FIT rates.

Possible path lengths through the critical volume of the cell, and group them into categories. For example, let us say all possible path lengths through the critical volume can be categorized as shown in Figure 8.

Since the minimum charge for an upset in our particular cell has been determined to be 50pC, then in order for a particle which deposits only 25pC of charge per micron of track length, to cause an upset, it must have a total path length through our sensitive volume of at least 2um in length. From Figure 8, we notice that only 25% of all possible path lengths though the sensitive volume have a path length of 2um or greater. So remembering that only 50% of our 8 particles per um² per second deposit 25pC/um, then the number of failures in time from our 25pC particles is given by:

\[ 8 \text{ particles/um}^2/\text{second} \times 50\% \times 25\% = 1 \text{ error per second}. \]

Now we calculate how many upsets are caused by the 50pC/um particles that deposit 50pC of charge per um of track length. Remembering that at least 50pC of deposited charge is required to upset our cell, this tells us that the 50pC/um particles must have a track length of at least 1um through our sensitive volume to cause an upset. We notice from Figure 8 that 50% of all possible track lengths through are cell are greater than 1um, so 50% of our 50pC/um particles will cause an upset. Recall also that only half of our total particles create a charge track that deposits 50pC/um, then failures in time caused by our 50pC/um particles, is given by:

\[ 8 \text{ particles/um}^2/\text{second} \times 50\% \times 50\% = 2 \text{ errors per second}. \]

Adding up the failure rate caused by the 25pC/um and 50pC/um flux of particles, we arrive at a total Failures In Time (FIT rate) of:

\[ 1 \text{ error per second} + 2 \text{ errors per second} = 3 \text{ errors per second}. \]

In summary, in order to calculate the Failures In Time rate for this example, we need to simulate:

1. The minimum LET energy required to upset our cell.
2. The maximum depth of our cell that is sensitive to deposited charge
3. The surface dimensions of our cell (L x W) that is sensitive to a strike.

So whilst the critical charge in a critical volume method of calculating FIT rates is a conceptually simpler approach compared to a full matrix TCAD simulation method, it still relies on accurate knowledge of sensitive depths and areas, which are best characterized using TCAD methods if the budget for real world characterization using high energy particle sources is simply too cost prohibitive.

**Soft Error Simulation Methods**

By now it should be clear that what we really need to know is the fraction of energetic particles that strike our circuit from all angles and for any given energy that will cause a functional upset. This is usually called the “sensitive cross section” which has units of area. The way we calculate this sensitive area, depends on what data is already available, how much we know about the process node, and how accurate the answer needs to be.

The essence of soft error simulation basically distills down to the ability of the simulation method, to accurately reproduce the transitory current pulse at the effected device electrodes, that results from a high energy particle strike at any angle and for the full energy range of particles.

Silvaco offers a number of soft error simulation methodologies. Some of the main choices are listed below:

1. SmartSpice only simulations
2. TCAD calibrated Spice simulations
3. Simultaneous TCAD and Spice (Mixed Mode) simulations
4. TCAD only simulations

As with most simulation tools, the simulation choice is a compromise between simulation accuracy and capability, versus simulation speed. The pros and cons of each of these approaches are listed below:

**1. SmartSpice Only Simulations:**

For these simulations, the current pulse is calculated by SmartSpice itself, from user input parameters, such a particle LET, and ionized track length etc., The calculated pulse has a sharp rise time, followed by an exponential decay, as shown in Figure 9, for a 1 LET particle strike.
on a 65nm device. The calculated current pulse, when integrated over the pulse time, yields the deposited charge in the track, which equates to the particle LET x the specified track length. This method can therefore be a good tool for investigation of the “critical charge” required to create a circuit upset.

An example of applying the pulse to a static 65nm SRAM circuit is shown in Figure 10 where it can be seen that a soft error occurs, causing the SRAM to change state.

The advantages are that the total simulation time is very short, allowing for many simulations of a hit at many circuit nodes and LET values. SmartSpice simulations are a good tool for investigation of the “critical charge” required to create an upset at any circuit node, which can be used in the volumetric method of FIT rate calculation previously described.

What can also be investigated are the effects of circuit’s sensitivity to particle strikes under various operating conditions, since variables such as operating in low power mode (reduced Vdd) increases the circuit sensitivity to soft errors, by reducing the particle energy required to create an upset logic state. Also factors such as “logic masking” can greatly enhance or reduce upset sensitivity, depending on the logic state a gate is in for the majority of the time. “Logic masking” is described later in this article.

The disadvantages are that these kinds of “Spice only” simulations do not provide accurate information on the effects of the strike’s angular dependence or the areal sensitivity of the strike. In other words, there is no information concerning the sensitive volume or surface area (cross section), which is needed to calculate the Failure In Time (FIT) rate, that the circuit designer needs to know.

2. TCAD calibrated Spice simulations:

A step up in versatility and accuracy, is to calibrate each active device response to a single event strike using TCAD simulations. This allows for additional simulation analysis of different angled strikes and different strike locations versus incoming particle energy (LET), so that the graph of device cross section versus incoming energy, shown in Figure 6, can be constructed. Alternatively, the TCAD simulations could be used to simulate sensitive depth and sensitive surface area to construct the “sensitive volume box” for use in the “critical charge” method of FIT calculation as previously described.

First a 3D TCAD model of all representative devices in the circuit are created an example of a 65nm nMOS transistor model is shown in Figure 11. The TCAD model

![Figure 9. Drain current pulse resulting from a 1 LET particle strike into a 65nm nMOS device, emulated directly in SmartSpice.](image)

![Figure 10. An SRAM memory bit flip resulting from the incoming particle described in figure 8, simulated directly in SmartSpice.](image)

![Figure 11. A basic 3D TCAD model of a 65nm nMOS device, used for calibrating areal and angular responses to incoming energetic particles of various energies (LETs). It can also be used for simulating the geometry of the “critical volume” for the “critical charge” method.](image)
must be 3D, due to the 3D nature of the charge track’s cylindrical shape. However, a simple extrusion of a 2D model to the correct width is often sufficient for simulation of basic information, as shown here.

Next, the TCAD model has to be calibrated against either measurements or a trusted Spice model card. Figure 12 shows simulated I-V curves from these TCAD simulations that have been calibrated to a Spice model card. Once calibrated, we now have a trusted TCAD model of our process node devices.

Now we have a trusted TCAD model of our sensitive devices, it is possible to simulate single event particle strikes at any angle, location and energy level, and capture the drain current responses in a data file of current-time pairs, which can then be injected into any appropriate circuit node during a Spice simulation, and at any time during that simulation. An example plot of TCAD generated drain current versus time pairs was shown back in Figure 4. When this data file is used as the input strike to a 65nm static RAM Spice simulation, a data corruption event occurs as shown in Figure 13.

The advantages of using TCAD calibrated data file in Spice simulations, is that each simulation is just as fast as a normal Spice simulations, but now the previously missing data of angle and position, is inherently included in each current-time pair data file.

The disadvantage of using TCAD in the way described above, is that the simulation, is of a single isolated device. So the TCAD calibrated drain current response, does not take account of how other circuit elements connected to that device, may modify the response of the device being struck. For example, if the current pulse caused the drain voltage to collapse, the current-time data may be different, as will be shown in the next section. Having said this, the circuit being investigated would have to be of critical importance for this to be a material issue for the designer, but it does warrant a mention for such cases.

3. Simultaneous TCAD and Spice (Mixed Mode) Simulations:

In order to mitigate the possibility, that the transient response of other circuit elements effect the actual response of the device being struck, it it possible to simultaneously simulate the struck TCAD device whilst being connected to the rest of the circuit elements, which are simulated using a Spice simulator. This mixed TCAD and Spice self consistent simulation method is called “Mixed Mode”, not to be confused with the same term often used to describe mixed analogue and digital Spice simulations.

As can be seen by comparing Figure 4 with Figure 14, the rest of the SRAM circuit does indeed make a material difference to the form of the drain current response of the struck TCAD device. However, the end effect of the strike to the circuit response, although slightly different, is materially the same in this case, resulting in corrupted SRAM data as before. Comparing the circuit response of the strike to the SRAM simulated using MixedMode shown in Figure 15, to the previous Spice simulation using the TCAD generated current time data file, shows the similarity of the two responses in this case.
The advantages of MixedMode simulations is that the full interactive feedback response of the entire circuit on the device being struck is now accurately simulated. The main disadvantage of this method is that now each different strike location simulation requires a separate simulation of the TCAD device, such that the total simulation time for a large DOE is now much greater. However, this may be tolerable for “array” type circuits, where circuit blocks are repeated many times, such as in memory applications.

One further disadvantage, especially for very aggressive geometries, is that the charge from the high energy particle strike, may effect more than one device. In some designs, devices are deliberately arranged, such that this is the case, in an attempt to get the effects of the charge track to create a self canceling phenomenon, resulting in more tolerant circuits. Up until now, obviously, we are only simulating the effects of a particle strike on a single device.

4. TCAD Only Simulations:
As mentioned above, in some cases, devices are deliberately positioned in an attempt to get the effects of the charge track to create a self cancelling phenomenon. In order to simulate this strategy, characterization of these designs is best effected by simulating the whole cell using 3D TCAD. The left picture in Figure 16 shows a 6 transistor SRAM cell, using an aggressive planar geometry, created using Silvaco’s 3D Victory TCAD. The picture on the right shows the location of the simulated strike.

Device simulation of the entire cell during the strike then yields the SRAM response to the strike shown in Figure 17. The advantages of using TCAD only simulations is that there is no restriction on the ability to simulate any interdependency of the reaction of the whole cell to a particle strike, from any angle, position or energy, affording the designer the ultimate in characterization accuracy.

The obvious downside of simulating more than one device in TCAD is the simulation time. However, with a new breed of solvers coupled with the multi-threading capability of Silvaco’s TCAD tools, it can be a practical tool for designing very critical cells.

Other considerations that affect real circuit FIT rates
Up until now, we have mostly discussed methods simulating and calculating soft error rates under various scenarios. However, what the buyer of any chip really needs to know, effectively boils down to a single figure of merit, namely, what is the number of Failures In Time (FIT) of the whole chip under normal operating conditions and environments. The failure rate of any circuit is strongly related to the technology node and type that was used to fabricate the chip, however, there are operational and design considerations that also have a strong effect on the failure rate. Some of the most important ones are mentioned below.
Average time that devices are in a conducting state:
The main end effect of a high energy particle strike on a biased transistor, is to create a pulse of current, which momentarily causes the device to be in a current conducting state. Clearly, if the device is already in a current conducting state, there is little effect on the transistor. It is only transistors that are not in this conducting state, which creates the possible vulnerability for a soft error in a logic cell, by making it flip state. Clearly then, if it can be arranged that the majority of the devices are in a conducting state most of the time, then the circuit will be more immune to soft errors by that same proportion.

Logic masking
Following on from the arguments in the section above, concerning the advantages of circuits where more of the transistors are in a conducting state most of the time, we can take this a step further and consider the advantages of certain types of logic gates.

As an example, consider a simple 4 input nMOS or bipolar NOR gate. The NOR gate will output a “zero” i.e. it will be in a conducting state, if ANY of the 4 transistors in the NOR gate, are in a conducting state. So when the output of this NOR gate is a “zero”, it does not matter if any of the other 3 inputs to the NOR gate are a “zero” or a “one”, since the logical output is the same, no matter what these other inputs are.

Therefore, when the NOR gate output is a “zero”, any of the 4 transistors can be struck by a high energy particle, with no consequence to the logical output of the circuit. In this conducting (zero) condition, therefore, the NOR gate is extremely immune to soft errors. It should be pointed out, however, that the converse is also true. In other words, if the NOR gate is in a non conducting state, a particle strike on any of the 4 transistors, will corrupt the output.

Reduced Supply voltage (power save mode)
There are two main ways to reduce power consumption when a circuit is in standby mode, the first method applicable to any logic circuit, is to reduce the supply voltage. Secondly, only applicable for synchronous circuits, is a reduction in the clock frequency.

Unfortunately, a reduction in power supply voltage has a detrimental effect on a circuit’s vulnerability to a soft error. In a CMOS circuit, the reduction in supply voltage, also reduces the maximum on current of the transistor pulling it’s partner to either of the supply rails. It therefore takes a correspondingly smaller current from the struck transistor, to flip the logic state of the cell. Therefore, the cell now becomes sensitive to particles with a lower incoming energy. Soft error simulations also clearly show this effect.

As can be seen from Figure 2, lower energy particles occur at higher flux rates, so the net effect of reducing the power supply is an undesirable increase in the soft error rate.

Reduced clock frequency (power save mode)
The other method of power reduction applicable to synchronous circuits, is to reduce the clock frequency. A clocked circuit brings the benefit of combinational logic gates only being vulnerable to a single event, if the struck transistor remains in an incorrect conducting state at the transition of the clock. Greatly reducing the clock rate, can therefore have a significant impact on a circuit’s sensitivity to a soft error, and to a good degree, can cancel out the increased vulnerability resulting from any reduction in power supply voltage.

The notable exception to the advantages of reduced clock rate, are positive feedback circuits, such as flip flops (static memory), since the internal feedback mechanism allows the stored state of the flip flop, to change at any time the cell is struck with a high energy particle, irrespective of the clock signal.

Triple redundancy (voting circuits)
The most extreme method for greatly reducing soft error rates, is to replicate the circuit 3 times. The logical outputs of all three circuits are then compared. If one of the outputs differs from the other two, the correct result is deemed to be that of the majority, and the one differing result is ignored. In essence it is similar to a whole circuit parity check. One advantage of this method is that more vulnerable, aggressive technology nodes can be used, giving faster processing power at lower soft error rates, at the obvious disadvantage of chip area and cost.

Conclusions
The essence of soft error rate simulation, is to plot the effective sensitive area, for which the circuit is sensitive to an upset from incoming particles of all energies and angles, as shown in Figure 6. Once characterized, this information is mapped to the known particle flux rates and energies in which the circuit is to operate to calculate the rate of circuit Failures In Time (FIT rate).

Whilst the essence of what needs to be achieved is simple enough, the practicalities of obtaining this information, can be a lengthy task, depending on the accuracy of the answer that is required.

Two basic methods have been described, either using a large design of simulation experiments, or by using the critical charge in a sensitive volume method of calculation.

Once the basic failure rate has been calculated, further design and operational considerations, some of which are described at the end of this article, can also be an additional factor in the final calculated FIT rate.
Q: What Types of 3D Delaunay Shape Refinement can be used in Victory Process?

A. The Victory Process cell mode Delaunay 3D device meshing algorithm already includes various TCAD-based local refinement algorithms to ensure accurate and robust device simulation. These include junction and interface distance refinement. One benefit of these approaches is that complex refinement behavior can be specified via a simple deck interface, but a limitation is that the results can only vary according to the small number of parameters of the schemes. In some cases, such as particle path refinement, it can be useful to have finer, more local control over the mesh and the shape distance refinement schemes have been produced to support this.

The new shape refinement algorithms plug into the existing framework for distance-based refinement based on a scalar sizing field. All of the distance-based refinement schemes share a subset of deck syntax which controls the size of the mesh elements on or inside the particular focus geometry and the distance over which the size of the elements increases to the uniform, background refinement size. The shape refinement algorithms have additional syntax which controls the parameters of the shape, such as the center of a sphere and its radius.

Examples
Syntax:

```plaintext
export victory(delaunay) \
   structure="delaunay_final_uniform.str" \
   max.size=0.1
```

Figure 1 shows a simple reference cube with elements of similar, uniform size. The right-hand image shows a cutaway view where the interior of the mesh can be seen. This example provides the basis for the further refinement produced by the shape distance algorithms.

Syntax:

```plaintext
export victory(delaunay) \
   structure="delaunay_final_box.str" \
   max.size=0.1 \ 
   box.min="0.616, 0.616, 0" \ 
   box.max="0.716, 0.716, 1" \ 
   max.box.size=0.01 \ 
   max.box.distance=0.5
```

Figure 2 shows the results of the box distance refinement algorithm. In this case, the axis-aligned box is defined by a pair of diagonally-opposite points which are on the boundary of the mesh. The plane defining the cutaway view passes through the center of the box. The elements on the interior of the box have a maximum size of 0.01 microns and the mesh is graded to the background size over a distance of 0.5 microns from the box.
Figure 2. Box refinement with cutaway.

Syntax:

```plaintext
export victory(delaunay) \
structure="delaunay_final_sphere.str" \
max.size=0.1 \
sphere.center="0.666, 0.666, 0.666" \
sphere.radius=0.1 \
max.sphere.size=0.01 \
max.sphere.distance=0.5
```

Figure 3 shows an example of the sphere refinement algorithm. The plane used to define the cutaway view passes through the center of the sphere which has a radius of 0.1 microns. The maximum element size within the sphere is 0.01 microns and the size of the elements increases to the uniform size over 0.5 microns distance to the sphere.

Syntax:

```plaintext
export victory(delaunay) \
structure="delaunay_final_cylinder.str" \
max.size=0.1 \
cylinder.start="1, 0.5, 0.5" \
cylinder.end="0, 1, 1" \
cylinder.radius=0.1 \
max.cylinder.size=0.01 \
max.cylinder.distance=0.5
```

Figure 4 shows an example of the cylinder refinement algorithm. In this case, the center of the cylinder passes through a corner of the cube and the middle of an opposite face. The maximum size of the element inside the cylinder and the distance over which the mesh is graded has been chosen to match Figures 2 and 3.

Figure 3. Sphere refinement with cutaway.

Figure 4. Cylinder refinement with cutaway.

Figure 5. Box refinement with cutaway.
Syntax:

```
export victory(delaunay) \\
    structure="delaunay_final_cone.str" \\
    max.size=0.1 \\
    cone.start="1, 0.5, 0.5" \\
    cone.end="0, 1, 1" \\
    start.cone.radius=0.01 \\
    end.cone.radius=0.1 \\
    max.cone.size=0.01 \\
    max.cone.distance=0.5
```

Figure 5 shows the results of refinement on distance to a truncated cone. The central axis of the cone passes through a corner of the cube mesh and the center of an opposite face. This refinement scheme generalizes the cylinder approach by allowing two radii to be specified, one for each end of the cone.

**Conclusions**

The shape distance refinement algorithms provide fine control over the resulting 3D Delaunay device mesh. Multiple instances of each shape can be specified within the same export statement. The new refinement algorithms can be combined with the existing distance-based schemes to ensure robust and accurate device simulation.