National Aeronautics and Space Administration

FPGA Insertion Guideline

Douglas Sheldon
Jet Propulsion Laboratory
Pasadena, California

National Aeronautics and Space Administration

FPGA Insertion Guideline

NASA Electronic Parts and Packaging (NEPP) Program
Office of Safety and Mission Assurance

Douglas Sheldon
Jet Propulsion Laboratory
Pasadena, California

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the National Aeronautics and Space Administration Electronic Parts and Packaging (NEPP) Program.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

Copyright 2008. California Institute of Technology. Government sponsorship acknowledged.

Table of Contents

Executive Summary

This insertion guideline has been developed to address the issues of implementing Field Programmable Gate Arrays (FPGAs) into earth-orbiting and deep-space missions. FPGAs provide the design engineer an enormously powerful tool for all areas of spacecraft design, including command and data handling, avionics, and instrumentation. Modern FPGAs also present unique qualification and verification challenges to the Mission Assurance community. This insertion guideline is meant to address the Mission Assurance needs.

The insertion guideline has three main sections:

1. Technology and Qualification
2. Design Flow
3. Device Specific Characterization

Successful insertion of an FPGA into a space mission requires activities in all three of the main areas.

Technology and Qualification means a thorough understanding of the details of how the FPGA is manufactured and how it might fail. The understanding of these failure mechanisms is then used to develop qualification tests and milestones to ensure the highest quality FPGA is obtained for use.

The Design Flow refers to the processes used by the design organization to ensure that the design and implementation of the FPGA are adequate. The FPGA is a design-centric device and the design process plays a pivotal role in the success of the overall insertion process.

Device Specific Characterization is a new and developing area for FPGAs. Modern, highly complex FPGAs represent a significant challenge to historical methodologies for parts review and insertion. Well-established concepts such as burn-in screening and life prediction have now become application-specific parameters. Due to this application-specific nature, on-board characterization of each design on each device becomes a requirement for successful risk mitigation and management.

1.0   Introduction

FPGAs are semiconductor devices that are programmed by the end user to perform a wide variety of circuit implementations. Programming an FPGA means to implement a custom design by making physical connections on the FPGA chip. These physical connections are made using several different technological approaches. The FPGA as delivered is a generic "blank slate." It contains programmable circuit and interconnect logic elements. This allows generic chips to become fully customized to meet specific requirements.

The number of times an FPGA can be programmed depends on the technology used to manufacture the FPGA:

- Once: Antifuse based, e.g., devices manufactured by Aeroflex/Actel
- Several times: Flash based, e.g., devices manufactured by Actel/Lattice
- Indefinitely: Static random access memory (SRAM) based, e.g. devices manufactured by Altera/Atmel/Xilinx

An example of an FPGA made on a 90-nm complementary metal oxide semiconductor (CMOS) process is shown in Figure 1.
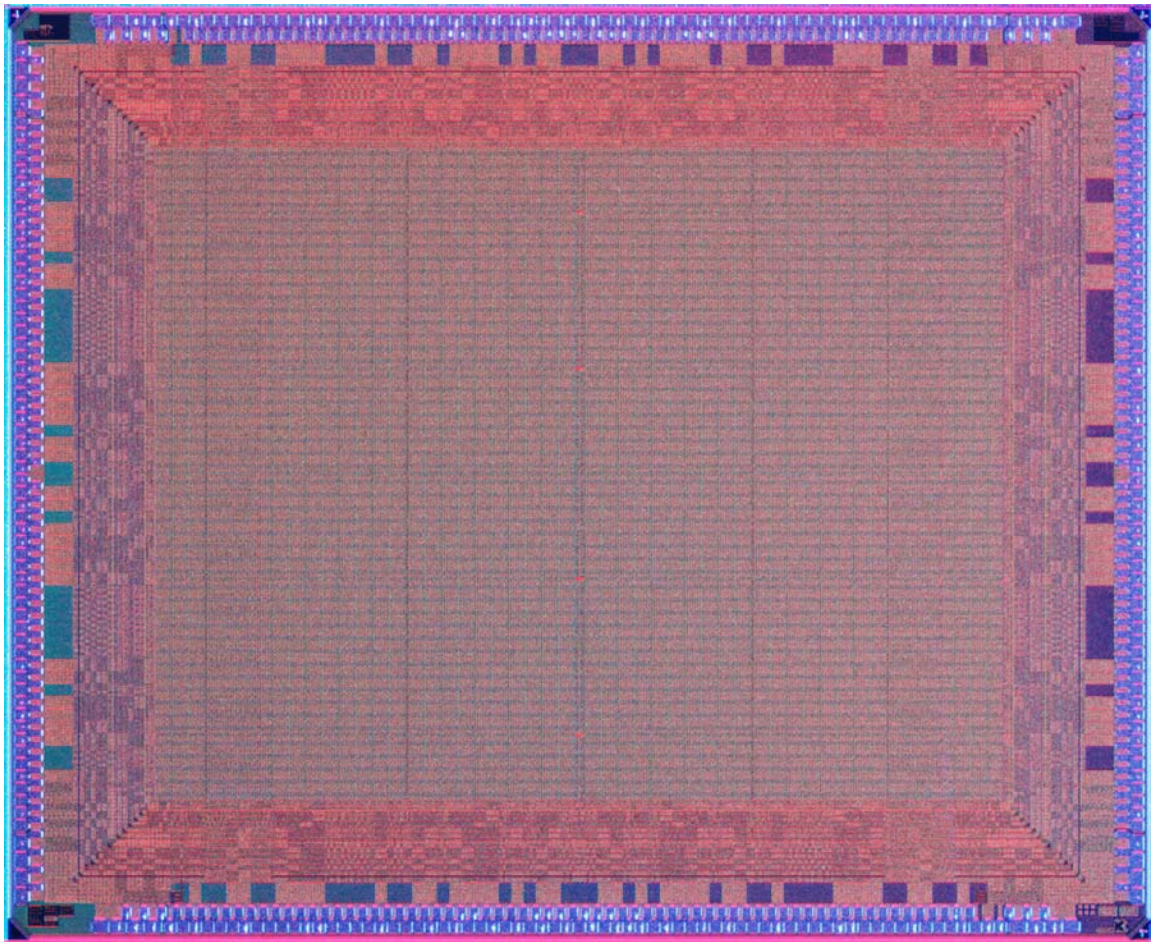

Figure 1. Xilinx Spartan-3 FPGA[1]

---

[1] Courtesy of Xilinx

The die photo of the Spartan-3 shows that the vast majority of the FPGA is made up of the same design layout. This area contains the logic fabric that is programmed to give the chip its unique functionality. Only the outside ring of the chip has different transistor layout. The outer areas of the die contain input/output (I/O), clock, and power-related circuits.

There are a variety of families of FPGAs available from different semiconductor companies. These device families differ in their architecture and feature set. Most devices allow a common approach: A regular, flexible, programmable architecture of Configurable Logic Blocks (CLBs), surrounded by a perimeter of programmable Input/Output Blocks (IOBs). These functional elements are interconnected by a powerful hierarchy of versatile routing channels.

The IOBs provide the interface between the package pins and the internal logic while the CLBs provide the functional elements for constructing most logic. On-board memory is also available. Clock Delay Locked Loops (DLLs) for clock-distribution delay compensation and clock domain control are provided to implement complex timing requirements.

As can be seen in Figure 2, the CLBs form the central logic structure with easy access to all support and routing structures. The IOBs are located around all the logic and memory elements for easy and quick routing of signals on and off the chip.
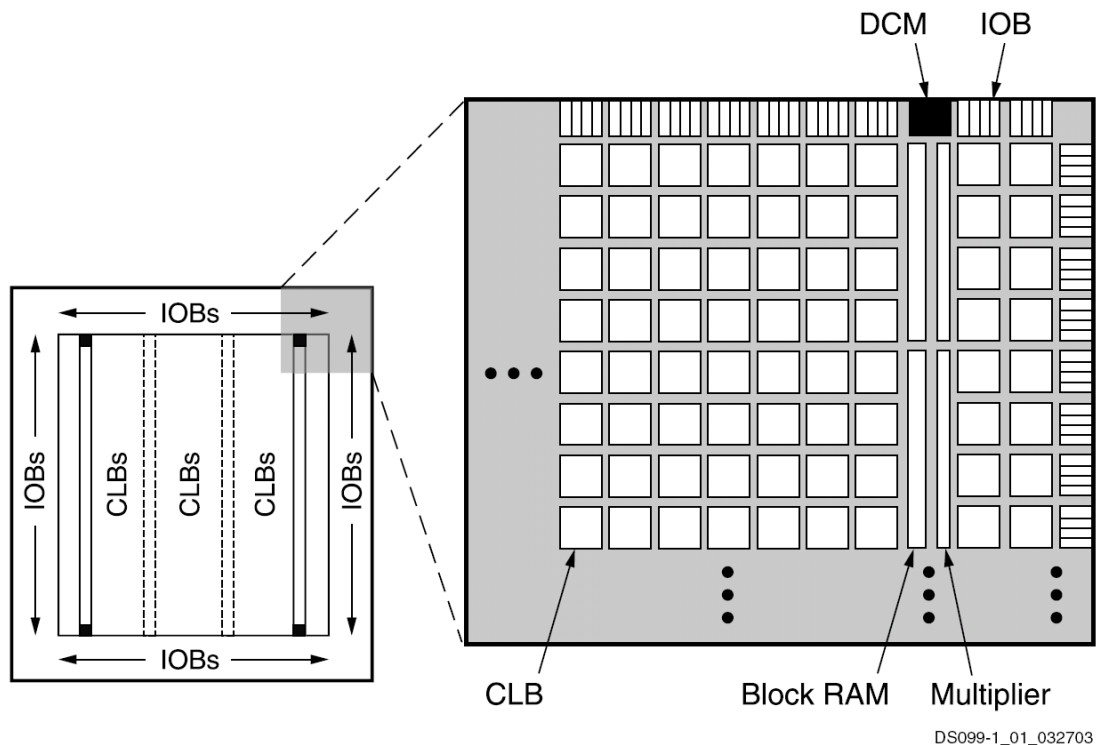

DS099-1_01_032703

Figure 2. Overview Layout of Spartan-3 FPGA[2]

---

[2] Courtesy of Xilinx

2.0   Technology and Qualification

The particular FPGA device technology and its related qualification procedures form the historical or "classical" approach to integrated circuit insertion. For modern FPGAs, this information must now be supplemented with design and specific on-board characterization procedures. These latter procedures will be discussed in sections 3 and 4. This section will focus on qualification and certification of FPGA technology.

The FPGA vendor must supply information and data related to the specific manufacturing process that the desired FPGA is produced with. Conceptually the overall technology and device qualification process can be diagramed in Figure 3 below:
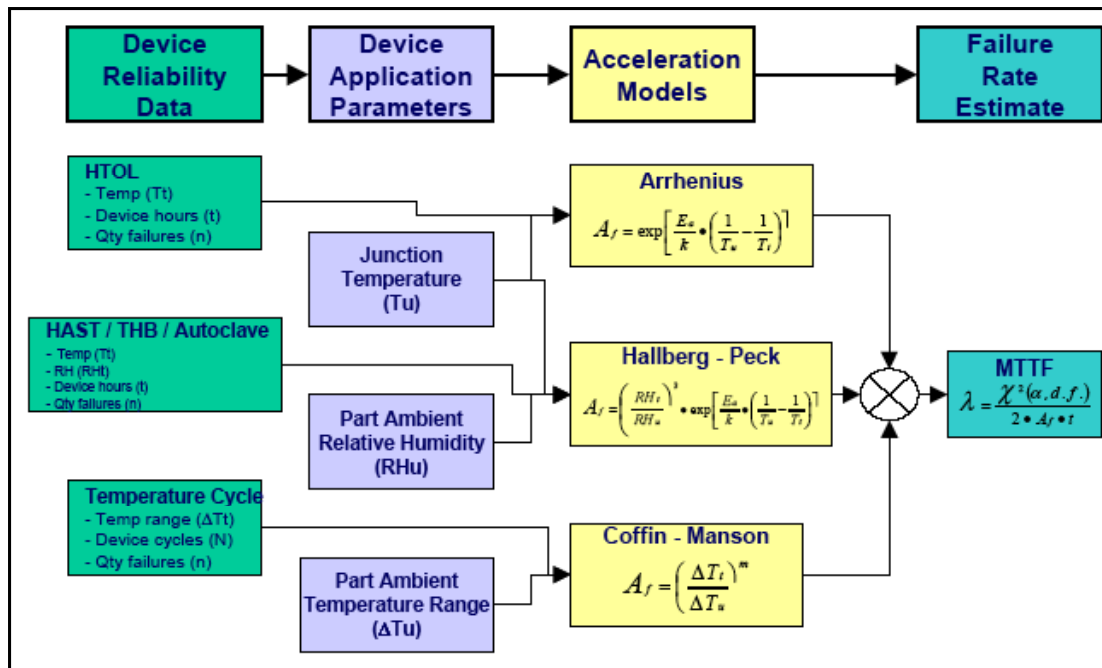

Figure 3. Conceptual Flow of Technology Qualification

Technology qualification is the responsibility of the FPGA vendor (and their foundry if necessary). This qualification requires specialized test structures and complex analysis to obtain predictions for lifetime and failure rate.

2.1   Basic Failure Mechanisms

Failure mechanisms relate to the entire process of making the FPGA, the wafer fab technology, as well as the packaging and assembly technology. While many of the CMOS technology process failure mechanisms are understood, many FPGAs use slight variations to meet specific performance goals. Examples of these variations to normal CMOS processing can be as significant as antifuse technology that is used as the core programming technology to the inclusion of a third (or middle) oxide thickness for a mix of moderate leakage, moderate speed transistors that are used in large number on SRAM-based FPGAs.

Successful insertion requires an understanding of all such processes. Vendors must supply detailed documentation that describes these mechanisms. As an example, antifuse failure mechanisms are highlighted.

Antifuse reliability is characterized both in the "*On*" and "*Off*" state conditions. In the *"Off"* state condition, the Time-Dependent Dielectric Breakdown (TDDB) of the antifuse insulator layers is measured. The antifuse TDDB for a 150-nm CMOS process is evaluated at 1.5V and is extrapolated to be > 100 years.

In *"On"* state reliability, the stability of the antifuse is of concern. The antifuse is a filament of titanium nitride (Ti_xN_y). It is formed by the displacement and then melting of local amounts of titanium and silicon nitride that are separated by a dielectric layer of amorphous silicon. This silicon is heated via current injection to change state and allow the melting process to initiate. Properly formed antifuses are stressed as a function of applied current to determine lifetime.

These programming current values are higher than the normal operation values. Once the amount of current flow in the low resistance antifuse reaches the value of the original formation current, the local temperature will be high enough to initiate another melting cycle and signals the end of life of the antifuse. The antifuses are stressed at these higher levels and a failure rate and failure population is extrapolated to use conditions of current. Once the test current drops below 75% of the initial programming current, the lifetime of a properly formed antifuse is extrapolated to be > 100 years.

The characterization of antifuses so far discusses average properties of a large collection of antifuses. Arithmetic mean properties of new material structures such as the antifuse are one part of the overall insertion process. Critical understanding is also required at the "time-to-first-fail" level as well. This thinking requires an understanding of the defect processes that might occur in technologies. Defect processes are usually the most important for high-risk spacecraft missions. The technologies have been screened to meet or exceed mission requirements but defects (both time independent and dependent) are often what results in measurable failures.

An example of such understanding of the defects in antifuse technologies is shown in Figure 4 [Ives et al. 2005]. Here, the antifuse is shown to actually be a formation of complex voids and metallizations. This has been uniquely determined by the application of custom Focused Ion Beam and Scanning Electron Microscope techniques.
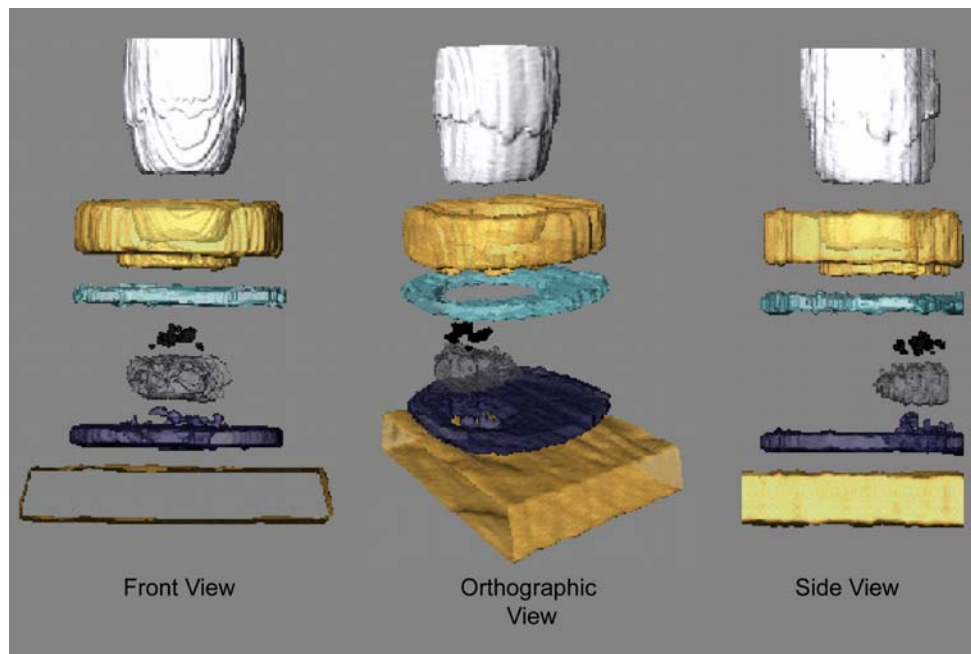

Figure 4. An exploded view at different angles for surface-contour 3-D reconstruction of the component objects in an FPGA antifuse. [Ives et al. 2005]

As a result of these techniques, the effects caused by possible defect sources (surface condition, dopant re-distribution, etc.) can now be understood and conceptualized.

2.2 Design Rules Built around Basic Failure Mechanisms

The design rules and the technology development processes are fundamentally united and intermixed. However, proper insertion requirements need to have an explicit statement of design rule implementation.

2.3 A Process Flow that Contains a "Maverick Control Process"

This type of control process means that the wafer foundry has an established procedure for identifying and eliminating abnormal and/or low yielding wafers and lots.

2.4 Reliability of the Transistors, Metallization, and Overall Product

There are a variety of CMOS technologies available. There are also a variety of ways to implement programmability to provide functional FPGA circuits. Accurate and precise understanding of the reliability of all these various process steps is required. An example cross-section of a 150-nm FPGA is shown in Figure 5.
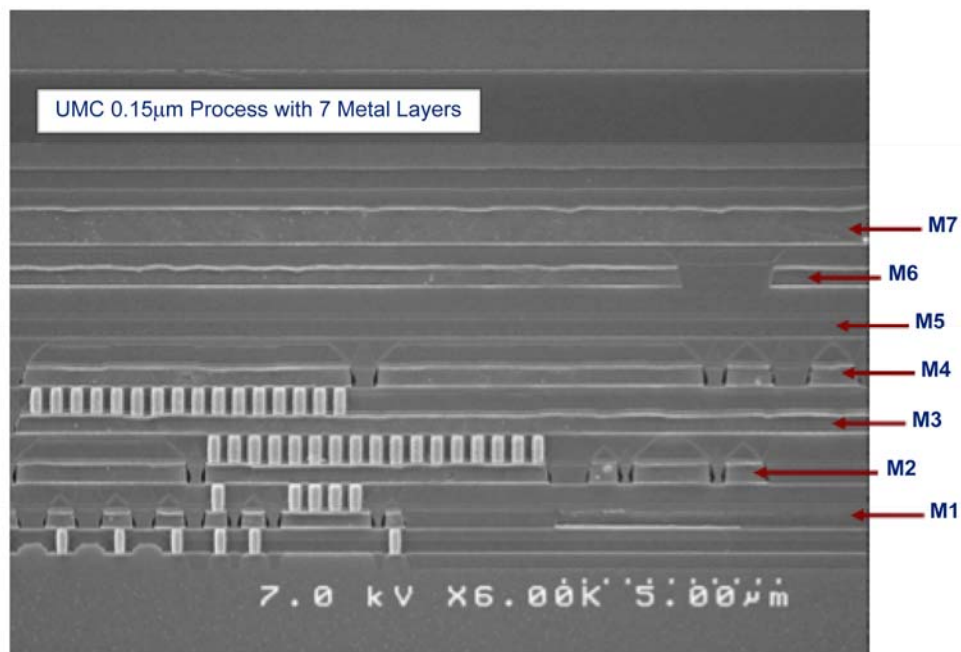

Figure 5. 150-nm FPGA Cross Section

Such an FPGA process will have a variety of fundamental device physics degradation mechanisms that could result in long-term reliability failures. These are:

- Hot Carrier Immunity
- Electromigration
- Time-Dependent Dielectric Breakdown
- Negative Bias Temperature Instability

2.4.1   Hot Carrier Immunity

Hot Carrier Immunity (HCI) is an evaluation of the core transistor(s) for immunity to hot carrier-induced degradation when they are subjected to accelerated stress condition. Results of accelerated tests are then fit to a model and the transistor lifetime is calculated. The lifetime model, τ, is:

$$\tau\, I_{dsat} = C\, (\, I_{sub} / I_{dsat})^{-m}$$

where *C* is a proportionality constant and is dependent on the dielectric technology, and *m* is equal to the critical hot-carrier energy for creating an interface state due to impact ionization. The failure criterion for HCI is:

$$\Delta I_{dsat} / I_{dsat} \geq 10\%$$

Given this failure criterion, the UMC specification for AC HCI lifetime is ≥10 years at 0.1% cumulative failure under worst-case operating condition of 1.1Vcc @ –55°C. RTAX devices are not planned to be in an environment where such cold temperatures can be experienced. Evaluations of minimum feature N-channel, P-channel, and high-voltage N-channel devices resulted in lifetimes > 22 years. Therefore, HCI is not a concern for this particular technology node.

2.4.2   Electromigration Testing

Electromigration (EM) testing is done to evaluate the endurance of metal lines, metal vias, and interconnect contacts when they are subjected to high-current density and high temperature. The projection of median time to fail (MTTF) for this test is Black's formula:

$$MTTF = A\, J^{n} \exp\, (Ea\, /\, kT)$$

where *A* is a proportionality constant and a function of material and the geometry of interconnect, *J* is the current density, *n* (~2) is the acceleration factor, and *Ea* = activation energy (~1eV).

Modern fabs usually use the following as an EM failure criteria:

$$\Delta R\, /\, R > 20\% \text{ or}$$
$$\text{Spiking leakage } IL > 1\mu A$$

where *R* is the line resistance of a standard wafer EM test structure, prior to the start of any temperature or current stress. UMC specification for EM lifetime is that each item must exceed 10 years @ 0.1% cumulative failure under operating condition of 125°C junction temperature and the maximum current density specified in the design rule. UMC data show EM lifetimes for all seven metal layers, vias, and contacts to be at least > 45 years. As a result, EM is not a concern for this technology.

2.4.3   Time Dependent Dielectric Breakdown

Time Dependent Dielectric Breakdown (TDDB) is done to evaluate the long-term stability of the insulator layers used in the transistors. UMC specification for TDDB lifetime is that it must exceed 10 years @ 0.1% cumulative failure under normal operating voltage. The low voltage and high voltage gate oxides for the UMC process have TDDB lifetimes >> 10 years. Therefore, TDDB is not a concern for this technology.

2.4.4   Negative Bias Temperature Instability

Negative Bias Temperature Instability (NBTI) has become a serious long-term reliability concern as the dimensions of CMOS devices are continually scaled. It has been reported that when the

oxide thickness is less than 3.5 nm, the threshold voltage shift (V_{th}) of the p-channel metal oxide semiconductor field effect transistor (pMOSFET) due to NBTI begins to limit the device lifetime. The gate oxide thicknesses for the 150-nm UMC process are listed as 2.8 nm, so NBTI effects need to be considered.

The threshold voltage shift during NBTI stress originates from electrochemical reactions at the SiO_2/Si interface. Special processing steps need to be taken to reduce the NBTI effects caused by these reactions. Several technical papers by UMC mention the advanced processing steps that they have taken to minimize NBTI degradation. At this current UMC gate length, 150 nm, the change in V_{th} that NBTI may induce is expected to be less than 10% of the original threshold voltage. This amount of shift is not considered to be a long-term concern.

At the 90-nm CMOS node, however, thin gate oxides and small channel length transistors make NBTI risk mitigation a major architectural as well as technology concern. Actel's competitor, Xilinx, already provides specific timing and design information for their 90-nm Virtex-4 device. Such NBTI concerns will need to be addressed on any future 90-nm Actel devices as well.

2.5   Performance in a Radiation Environment

Screening flow(s) required to eliminate any potential unreliable parts are provided below.

> RLAT for each lot
> Total ionizing dose (TID)—Method 1019
> Single-event latchup (SEL)—linear energy transfer (LET) > 120 MeV-cm^2/mg
> Single-event upset (SEU)—immune for LET of > 75MeV-cm^2/mg or with mitigation or by design

Details of FPGA radiation performance are available in [Adel 2008].

2.6   Mature Wafer Foundry Technology

Modern FPGAs are made on high volume, state of semiconductor fabrication processes. In some cases, the FPGAs selected for NASA missions are made using a CMOS technology that is two to three generations behind the most advanced process available at the foundry. In other cases, the FPGAs may be made on a process that is only one generation behind the smallest process feature sizes available. FPGAs are too difficult to debug from a yield point of view for the foundry. This is why FPGAs are not used as designs to "drive" new process developments.

NASA FPGAs should be made on a "mature" CMOS process. A mature CMOS manufacturing process is defined as having at least 1 year of data available on it in terms of designs made with it for both long term reliability and yield information. These processes should be contributing at least 10% of the overall gross income for the wafer foundry business.

2.7   Package Part Testing and Screening

Packaging modern FPGAs is a real technical challenge given the large number of pins (>500) in these devices. Each new generation of FPGAs is usually accompanied by a new evolutionary development in packaging technology. Qualifying these new package offerings is a major effort by the FPGA vendor. Proper insertion of FPGAs into space programs requires that the end user is very well versed in the details and results of the manufacturers' package qualification procedures. Manufacturers are quickly transitioning from wire bonding to flip-chip attachment, particularly for devices that have greater than 1,000 pins.

Thermal Runaway: Thermal runaway is a positive feedback mechanism between leakage current and silicon junction temperature. Leakage currents in highly scaled CMOS processes like the 150-nm one that Actel uses for the RTAX can increase in a steep, non-linear manner as the

ambient temperature of the device is raised. Increasing leakage current causes an increase in power dissipation and hence junction temperature, which causes further increases in leakage current, which drives further power increases. If allowed to continue, this condition can cause the current at some point to instantaneously reach a system short circuit condition and the device to catastrophically fail and self-destruct.

The thermal runaway condition is determined mostly by the leakage characteristics of the transistors given the coupled nature of the temperature dependence in the equation of subthreshold leakage to overall device power. The threshold voltage, V_{th}, is the appropriate parametric value to monitor thermal runaway. Actel has demonstrated an empirical fit of leakage current as a function of temperature data on the RTAX device in order to obtain a practical model to predict thermal runaway behavior as a function of V_{th}.

The results of this model of the 150-nm UMC process show that for Vth values of 435mV, thermal runaway will occur at junction temperatures between 135°C and 150°C. The difference in the temperatures is due to the estimated dynamic power of the device. For a device with dynamic power of 131mW, the thermal runaway temperature is expected to be 135°C. For a design with a dynamic power of 1W, the thermal runaway temperature is expected to be 150°C.

As the V_{th} decreases, so does the temperature for thermal runaway to occur. For a V_{th} of 350mV, the junction temperature for thermal runaway is expected to be between 99°C and 115°C, for designs of 131mW to 1W, respectively.

To address this issue of thermal runaway, JPL has taken the following actions:

- Lower the maximum allowable junction temperature to 110°C.
- Purchase all JPL devices from lots that been screened to have at least 430mV V_{th}.
- Purchase all JPL devices from lots that receive an additional 168hr/125°C thermal runaway screening step.

Junction Temperature: In order to support the concern raised by the thermal runaway issue, JPL is working to correlate the measured case/junction temperature of the device to the design tools. If such a correlation can be developed (as is currently expected), then the designers will have a high degree of confidence that the tools are accurately predicting design power usage, and with appropriately detailed board-level thermal analysis, the resulting junction temperature.

Using this approach, potentially risky designs that produce power dissipation and junction temperature values near the thermal runaway conditions can be identified early in the design cycle and then re-designed to reduce power and junction temperature values. Assuming this correlation can be made, the use and extraction of junction temperature and power dissipation values will be required for all Mars Science Laboratory (MSL) designs. Each design will be risk rated based on these values.

All JPL RTAX designers should expect and be able to accommodate the standby current to at least double if not triple in value when the temperature changes from room temperature to the maximum junction temperature of 110°C.

In support of this work, JPL is also measuring the junction temperature directly on the die via one of the unused I/O pins. An external current sensing circuit has been developed that extracts temperature values from forward biased I/O protection diodes. This technique is being developed and traded off with other temperature measurement methods.

Post-Programming Testing: Post-programming testing is conceptually designed to evaluate the programmed antifuses. This is very difficult to accomplish without using the exact design-related operational vectors and inputs. Also, acceleration for antifuse failure is more strongly voltage-related than temperature, making an effective screen impractical. As a result, operational

conditions are defined for post-programming testing to provide a practical, non-accelerated test and screening procedure.

For example, all MSL flight systems must meet the Design Principle requirements of high fidelity, full coverage testing of >200 hours pre-assembly, test, and launch operations (pre-ATLO) and >500 hours in ATLO for each redundant assembly. This requirement is 1000 hours for single string assemblies. Payload assemblies must meet the Design Principle requirements of high fidelity, full coverage testing of >300 hours pre-ATLO and >200 hours in ATLO.

Tri-Temp Testing: Tri-temp testing is normally a part of device evaluation and screening flow. However, due to practical test vector implementation issues mentioned above, tri-temp testing is not being required at this time. The use of the Actel Rev E flow procurement (Table 1) provides room temperature parametric and timing information on a per die basis. Die within specifications but with high parametric values would be embargoed from use on designs that are expected to have the highest power usage.

The justification for this practice is that Tri-temp testing is not an effective screen for an incompletely programmed part. Tri-temp testing measures parametric data and basic function timing parameters. The only access to programmed antifuses at this point is to exercise the logic of the part using design-specific test vectors and conditions. The parts at this point are ready for use. Any extra testing with CGA packaged parts is problematic.

Table 1. Actel E Flow

Step	Screen	Method	Requirement
1.	Destructive In-Line Bond Pull	2011, Condition D	Sample
2.	Internal Visual	2010, Condition A	100%
3.	Serialization		100%
4.	Temperature Cycling	1010, Condition C	100%
5.	Constant Acceleration	2001, Condition B for CQ352, LG624 Condition D for CQ208 Condition TBD for LG1152Y$_1$ Orientation Only	100%
6.	Partide Impact Noise Detection	2020, Condition A	100%
7.	Radiographic	2012 (one view only)	100%
8.	Pre-Burn-In Test	In accordance with applicable Actel device specification	100%
9.	Dynamic Burn-in	1015, Condition D, 240 hours at 125°C or 120 hours at 150°C minimum	100%
10.	Interim (Post-Burn-In) Electrical Parameters	In accordance with applicable Actel device specification	100%
11.	Static Burn-in	1015, Condition C, 72 hours at 150°C or 144 hours at 125°C minimum	100%
12.	Interim (Post-Burn-In) Electrical Parameters	In accordance with applicable Actel device specification	100%
13.	Percent Defective Allowable (PDA) Calculation	5%, 3% Functional Parameters at 25°C	All Lots
14.	Final Electrical Test	In accordance with applicable Actel device specification, which includes a, b, and c	100%
	a. Static Tests (1) 25°C (Subgroup 1, Table 1) (2) –55°C and +125°C (Subgroup 2, 3, Table 1)	5005 5005	100%
	b. Functional Tests (1) 25°C (Subgroup 7, Table 15) (2) –55°C and +125°C (Subgroup 8A and B, Table 1)	5005 5005	100%
	c. Switching Tests at 25°C (Subgroup 9, Table 1)	5005	100%
15.	Seal a. Fine b. Gross	1014	100%
16.	External Visual	2009	100%

Source: [Actel 2007]

3.0   Design Flow

The design flow of an FPGA has two fundamental pieces, process and documentation. The process refers to the intellectual and engineering activities that define the FPGA design from conception to implementation. For a successful FPGA insertion, this process must be a rigorous and methodical one. Documentation refers to the accepted and approved standards that define requirements and success criteria at each stage of the process. A successful process must include clear and meaningful documentation and the documentation must accurately reflect the goals and requirements of the design process.

Both the design and process activities of FPGAs are centered on the Computer Aided Engineering (CAE) or Integrated Systems Environment (ISE) tools. These FPGA tools are integrated throughout the entire design flow. The design methodology flows from synthesis and simulation directly through to place and route. The documentation that defines the completion and transition to the next portion of the design flow is contained in the ISE.

Any organization that is designing an FPGA for space applications *must have* a specific and formally approved design guideline. The design and development process of an FPGA for complex applications in spacecraft absolutely requires a thorough and well-vetted design process that is described in detail and adhered to rigorously. Examples of these are [Burke 2004] and [Butler 2006].

It is not the intent of this insertion guideline to provide the level of detail and requirements that documents such as these provide. This insertion guideline provides an overview discussion and review of these processes and requirements.

3.1   FPGA Design Process—Overview

An overview of the FPGA design process flow is illustrated in Figure 6. At its highest level, the FPGA design process is an iterative one. Specifications are developed, then reviewed, then implemented. The next level of requirements and specifications is then developed, reviewed, and so on.

The FPGA review process is the basic template from which all insertion activities take place. There are three general types of reviews:

- Formal
- Engineering
- Peer

Formal reviews take place at well-defined lifecycle points during the overall mission development cycle. Examples of these lifecycle points are:

- Initial conceptual review (ICR)
- Preliminary design review (PDR)
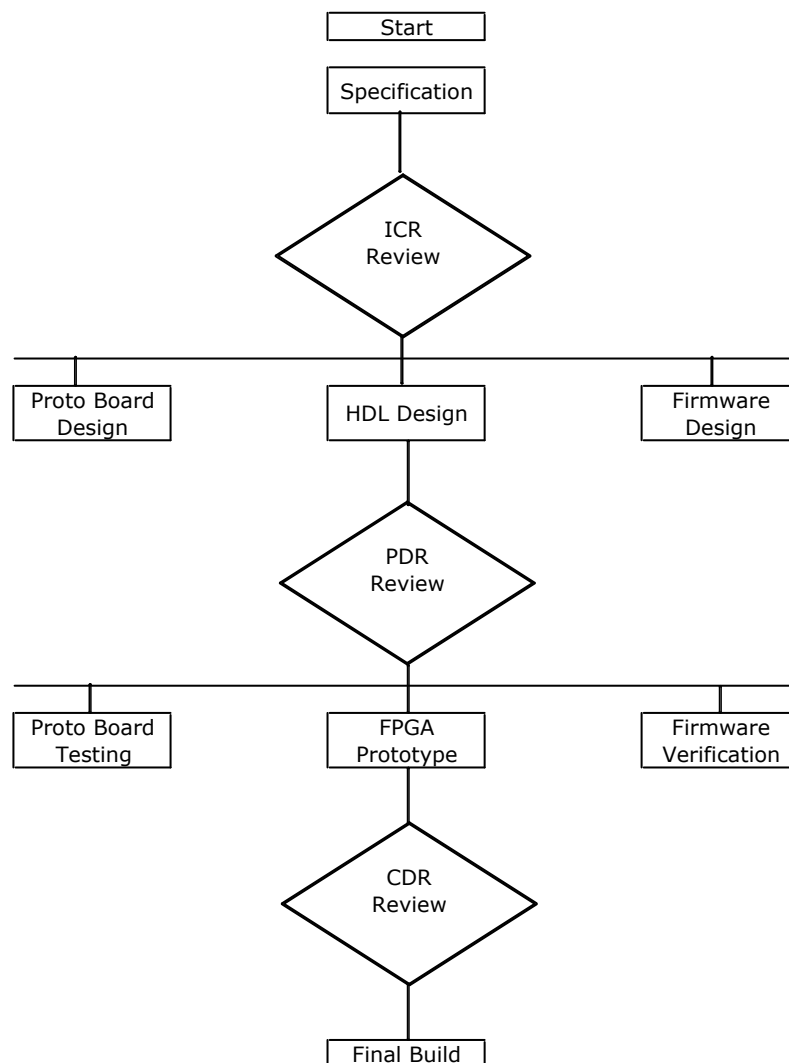- Critical design review (CDR)

Figure 6. Overview of FPGA Design Process

Formal reviews are by definition "formal" because they are explicitly defined in a controlled document [Rose 2006]. These documents will include such topics as required attendance, expected documents, etc. The overall cost and schedule performance of the design should be discussed during these reviews. Formal reviews occur at the highest level and the participant list will include representatives from areas other than FPGA design (Program Management, Mission Assurance, etc.). As such, detailed engineering problem-solving discussions are not possible in a formal review.

Engineering (or checklist) reviews are subsets of formal reviews. Engineering reviews produce signed agreements that a design has met a certain set of milestones. These milestones are the results of the formal reviews and usually reflect action items that the formal reviews produce. The number and substance of engineering reviews can vary with the project, but such engineering reviews are designed to provide requirements for the formal reviews. The audience of the engineering review is more focused to allow for discussions in greater detail than is possible with the formal review.

Peer reviews are intended to address specific progress on technical and/or schedule issues during the design and development of the FPGA. These reviews can occur at any time during the development cycle. A formal subject for the peer review is required as well as follow-on action and recommendations

3.2   FPGA Specification Description

The specification document will describe the implementation and the plan to achieve the given implementation. The specification contains the following elements:

- Preliminary implementation technology choice (describes the choice of FPGA and other components)

- Initial partitioning firmware/hardware/external components
    - o   How the design will be partitioned between FPGAs, other components, and firmware
    - o   Block diagram showing the partitioning of the design

- Preliminary 'intellectual property' (IP) selection
    - o   IPs are existing designs that can be incorporated into the FPGA. These may be purchased, licensed, or they may already be JPL property. The use of IPs can reduce the cost and schedule of the design process. This section of the specification lists those IPs that will be included, with justification.

- Test approach
    - o   Much of the FPGA schedule is related to test. Unless a function is tested completely, there is no guarantee that it will work correctly. This section of the specification details the test approach to be used.

- Preliminary FPGA device and package selection
    - o   An FPGA needs to be chosen that meets the flight requirements and is available in a package that can be used in the flight assembly. Although the package is likely to change, the specification will list several suitable candidate FPGAs and packages.

- Configuration management approach
    - o   Even a perfect design can be corrupted if an incorrect version is ultimately implemented. Configuration management and version control are extremely important in maintaining the integrity of the design. This section of the specification details the configuration management approach.

- Review plan
    - o   This shows the formal reviews and peer reviews and when they will occur.

- Designation of a fault-tolerant design approach
    - o   For flight FPGAs, fault tolerance is of high concern. Expected reliability under defined radiation levels, temperature ranges, temperature cycling, anticipated lifetime would be listed. This section explains how the design will meet those requirements.

- Electrical computer-aided engineering (ECAE) tools to be used
    - o   The FPGA designers will use a variety of sophisticated design tools. This section lists those software/prototyping tools planned for the work.
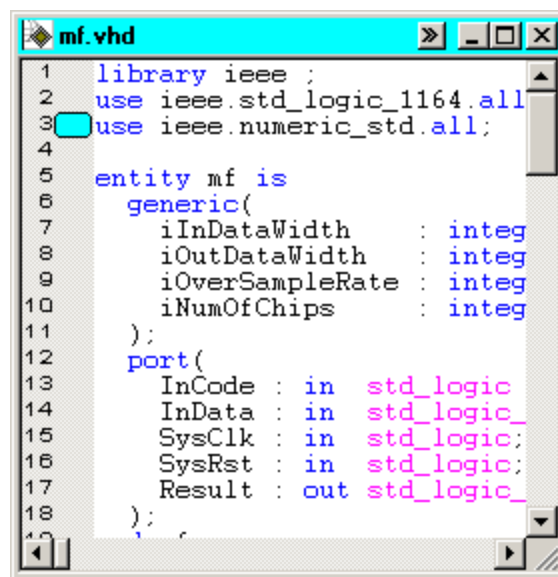
3.3   Design Entry

Design entry is the process of creating the design and entering it into the development system. Design tools are the heart of FPGA design and development and represent a constantly changing and ever improving product from FPGA and EDA vendors. The design entry flow is only briefly discussed in this insertion guideline. This discussion does serve the important point to provide an overview and initial insight into the design entry process. This process has a significant impact on the overall quality and reliability of the FPGA design. Engineers concerned with proper FPGA insertion need to be versed in this design activity in order to understand and contribute to it.

There are several methods used for design entry. These include:

- HDL Editor
- State Machine Editor
- Block Diagram Editor

Typing a design into an HDL Editor is the most obvious way of entering high-level languages like VHDL into the development system. Recent editors offer functionality like syntax highlighting, auto completion, or language templates to speed-up design entry. The main advantage of using an HDL Editor for design entry is that text files are simple to share across tools, platforms, and sites. An example of this is shown in Figure 7.

Figure 7. Example of VHDL Editor[3]
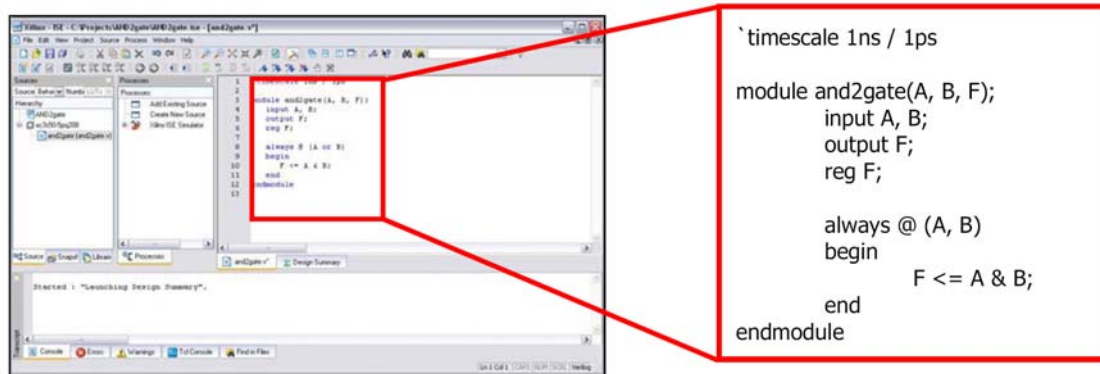
---

[3] Courtesy of Xilinx

Figure 8. Example of Behavioral Simulation[4]

After design entry, the design is verified by performing behavioral simulation as in Figure 8. To do so, a high level or behavioral simulator is used, which executes the design by interpreting the VHDL code like any other programming language, i.e., regardless of the target architecture. At this stage, FPGA development is much like software development; signals and variables can be watched, procedures and functions may be traced, and breakpoints may be set. The entire process is very fast, as the design is not synthesized, thus giving the developer a quick and complete understanding of the design. The downside of behavioral simulation is that specific properties of the target architecture, namely timing and resource usage, are not covered.

The next step is synthesis. Synthesis is the process of translating VHDL to a netlist, which is built from a structure of macros, e.g., adders, multiplexers, and registers. Chip synthesizers perform optimizations, especially hierarchy flattening and optimization of combinational paths. Specific cores, like RAMs or ROMs are treated as black boxes. Recent tools can duplicate registers, perform re-timing, or optimize their results according to given constraints.

After performing chip synthesis, post-synthesis simulation is performed. Timing information may or may not be available at this time. Sometimes preliminary simulations are performed that are based on statistical assumptions. Due to the mapping of the design into very basic macros, simulation time is lengthy. When post-synthesis results differ from behavioral simulation, initialization values have usually been omitted, or don't-cares have been resolved in unexpected ways.

Implementation is the process of translating the synthesis output into a bitstream suited for a specific target device. This process consists of the following steps:

- Translation
- Mapping
- Place and route

During translation, all instances of target-specific or external cores, especially RAMs and ROMs, are resolved. This step is much like the linking step in software development. The result is a single netlist containing all instances of the design.

During mapping, all macro instances are mapped onto the target architecture consisting of Look Up Tables (LUTs), I/O Blocks (IOBs), and registers. With this step completed, the design is completely described in primitives of the target architecture.

---

[4] Courtesy of Xilinx

During place and route, all instances are assigned to physical locations on the silicon. This is usually an iterative process, guided by timing constraints provided by the designer. The process continues, until the timing constraints are either met, or the tool fails to further improve the timing.

After implementation, all timing parameters are known; therefore, a real timing simulation may be performed. Timing simulation is a lengthy task, as the structure of the silicon including timing is simulated. It can be difficult to create test benches, which exercise the critical timing paths. This is a key area however for high reliability FPGA designs. This is a step early in the process that will help analyze potential quality and reliability weaknesses. Sometimes organizations choose to not perform timing simulation, but instead perform a combination of behavioral simulation and static timing analysis.

Static timing analysis computes the timing of combinational patches between registers and compares it against the timing constraints provided by the designer. The confidence level of this method depends on the coverage and correctness of the timing constraints.

4.0   Device Specific Characterization

Testing FPGAs is an enormously complicated task. Each FPGA implementation is a custom ASIC. Custom ASICs require custom test programs. Testing takes place at many points in the overall process of manufacture, development, and validation of an FPGA. Testing can be designed to detect low yielding parts, done at package or wafer level, and be used as a burn-in screen.

The manufacturer of FPGAs implements testing at wafer and package part level, designed to screen out defects and poor performing parts. This testing requires enormous technical and intellectual resources. Large (>500 pin) and expensive digital testers are needed as well as a fundamental knowledge of how the FPGA is constructed and implemented. These details are highly proprietary and not available to engineers outside the FPGA manufacturer.

Successful insertion of FPGAs is built upon this complex testing process. Large-scale FPGAs offer practically infinite combinations of implementations in terms of connection paths and resources used. Testing exactly the resources used for a particular design can only be accomplished by testing the design itself. The assumption that all manufacturing testing covers all possible user designs is implicit in FPGA operation. However, for high reliability applications, this assumption needs to be verified and supported.

The uniting of test vectors to possible device degradation is done with a fault model in mind. There are a wide variety of fault models and it remains an important and active area of research. Testing an ASIC from a logic point of view involves:

- Functional patterns
- Structural or scan-based testing
- Built-in Self Test (BIST)

In developing a logic fault model approach to test, fault models are developed from a Verilog point of view and applied to gate-level circuit view. These models are usually designed to be easy to compute and as a result are not "*defect-accurate.*" This means such test vectors can miss the normal distributions of defects across a die. One reason for this is the application focus to gate interconnects. Often fanout stems and branches are treated separately.

An example of this is a single stuck-at fault model. Here, every net, one at a time, is assumed fixed at a logic 0 or logic 1. Tests are then generated, which elicit errors if a fault is present [Eldred 1959]. An example of this kind of stuck-at fault is shown in Figure 9:
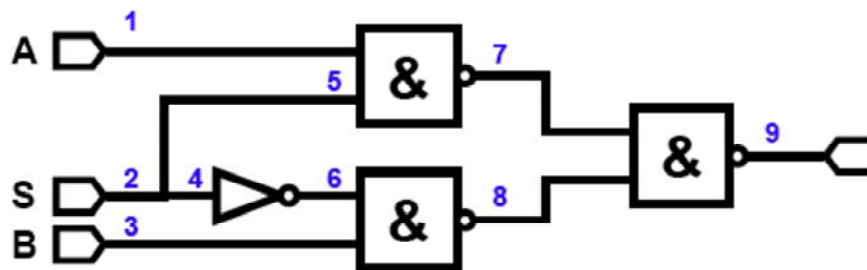

Figure 9. Example of Stuck-At Fault Testing [Bulter 2006]

Here, inputs A, B, and S provide 9 sites with two cases (0,1) of possible conditions. This makes for a total of 18 possible single stuck-at faults.

Other logic fault models are summarized in Table 2.

Table 2. Logic Fault Models

Fault Model	Comments
Bridging	Extracted from layout using neighbors and capacitance
Transition (delay)	Same as a stuck-at, but must pre-condition net to opposite value to force a transition
Path Delay	Delay through a specific circuit path. Based on static timing analysis

Proper insertion practices bring out the differences between designers and mission assurance people. Designers always have patterns or test cases to simulate for design correctness because of working in the simulation environment. Simulation is event-driven but test equipment is cycle based. There is a lengthy conversion process between these two very different worlds.

Fault simulation is extremely slow and costly and typically not done. A trade-off between controllability and observability exists with reliability testing. Controllability is the ability to control an internal circuit node to a particular logic value. Observability is the ability to observe the value on an internal circuit node. A test has access to only a few hundred pins while the FPGA is made up of literally millions of gates. Functional patterns require many pins at high speed. This is very costly and often incompatible with some burn-in setups.

At the core of the issue is how to select patterns that provide adequate stress. Functional patterns are typically lower power than structural tests. The process between designers and reliability/mission assurance is shown in Figure 10.
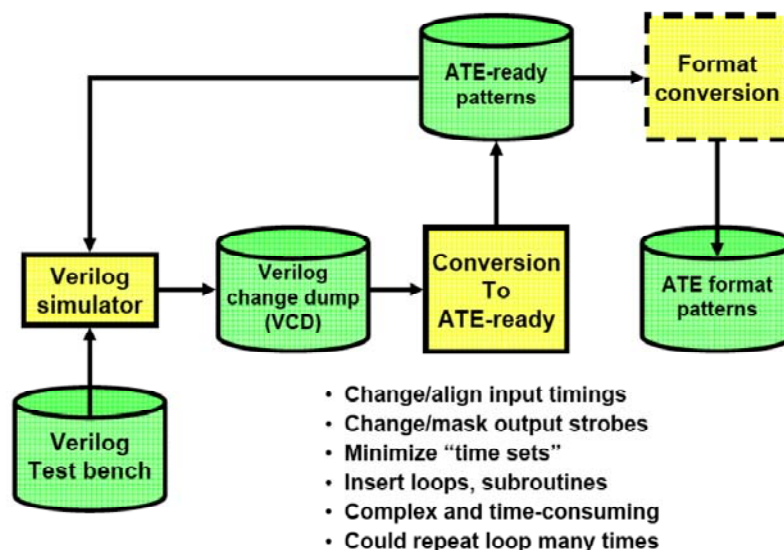

Figure 10. Test Vectors from Simulation Environment

Besides digital logic, modern FPGAs employ large numbers of RAM cells. Testing memory cells provides another set of problems unique to their architecture. Besides common models such as stuck-at faults in cells, other fault modes are:

- (State, idempotent, inversion) coupling faults
- Disturb faults
- Retention faults

RAM tests are usually developed as algorithms rather than fault models.

Mapping of electrical testing to device phenomena remains a major area for development. Test vectors need to be constructed to be able to map underlying physical mechanisms that cause reliability degradation. The relationship of the design being tested to test vector used to the device level parameter being examined is very complex. The role of specifically designed test circuits is becoming more and more important and relevant to proper qualification of FPGAs.

Figure 11 shows a logic map of potential device degradation phenomena to potential FPGA test vectors. The mapping of such a relationship can be complex and subtle.
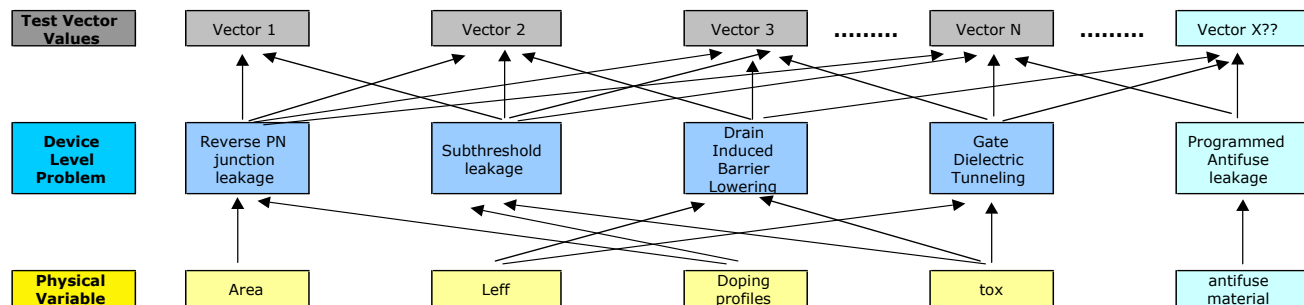

Figure 11. Mapping of Test Vectors to Possible Device Degradation

5.0   Risk Management Approaches for FPGAs

The end result of an insertion process is to define the risk of using an FPGA for a NASA mission. The expression of this risk reflects the synthesis of all the topics discussed in this guideline as well as a multitude of other factors not discussed such as mission requirements, possible contractor expertise, and a variety of other variables. The goal of this final synthesis is often to produce a simple "Yes" or "No" in terms of use of the FPGA, where "Yes" reflects the risk is low enough (or has been reduced enough), while "No" means the risk is too high (and/or the risks have not been mitigated).

A simple "check-the-box" approach to risk synthesis often becomes problematic because there are usually shades of completion and various mitigating circumstances that occur in the development and implementation of a complex FPGA. The assurance engineer might have 80% completion of a given task listed in the insertion process. Such a high level of completion leaves open for debate whether or not that box should be considered checked.

Modern, complex FPGAs require a sophisticated risk mitigation scheme. An integrated qualification scheme has been proposed [Sheldon 2005]. Risk management is defined as "the process of determining what areas could produce a reduction in performance and/or an actual failure of an FPGA device." Once these potential failure areas have been identified, plans and methodologies are implemented to address and mitigate the concerns.

This approach provides a qualification plan that is application-specific based upon the knowledge base of technology, mission, and system requirements. A multi-tiered approach to qualification can then be developed to provide a robust, layered risk reduction methodology. Such a multi-tiered approach allows for "trade-offs" of various tests and screens to provide realistic cost management capability for missions while explicitly acknowledging risk.

This approach results in a risk matrix being developed for FPGAs [Sheldon 2007]. Conceptually, this matrix is defined in Table 3.

Table 3. Conceptual Risk Matrix

Activity	Low Risk	Medium Risk	High Risk
Technology			
Tech-1	X	X	
Tech-2	X		
Antifuse			
Antifuse-1	X	X	
Antifuse-2	X		
Design			
Design-1	X	X	
Design-2	X		
Screening			
Screening-1	X	X	
Screening-2	X		

NOTE: X implies a particular risk reduction activity.

In a formal mathematical representation, this matrix can be interpreted as:

$$Risk\ Factor\ (T,F,D,S) = \sum_i a_i T_i + b_i F_i + c_i D_i + e_i S_i$$

Here risk factor (RF) is defined as a function of technology (T), antifuse (F), design (D), and screening (S) steps or activities. Each step/activity has an associated parameter that is used to "weight" its contribution. Screening step one (screening-1 or $e_1 S_1$) can be weighted more than screening step two, $e_2 S_2$, for example.

Numerically this means that e_1 may be assigned to 10 while e_2 is assigned to 1. If the step/activity is not performed or included, then the parameter is assigned to zero. The final risk factor is determined by the total number of activities/steps that are performed or implemented. The lower the final number, the higher the risk factor.

In a completely discrete representation (and discrete *interpretation* of the matrix), an example RF would be:

$$RF\ (Medium) = a_1T_1 + b_1F_1 + c_1D_1 + e_1S_1$$
$$and$$
$$RF\ (Low) = a_1T_1 + b_1F_1 + c_1D_1 + e_1S_1 + a_2T_2 + b_2F_2 + c_2D_2 + e_2S_2$$

Because of the weighting of the parameters, the matrix can also be used to define continuous regions of risk. An example of this type of continuous risk region is shown in Figure 12. Figure 12 shows Risk Factor on the z-axis as a function of two risk reduction parameters, screening and design. There are 3 levels of design activity and 10 levels of screening. Bands or regions of risk can now be defined.
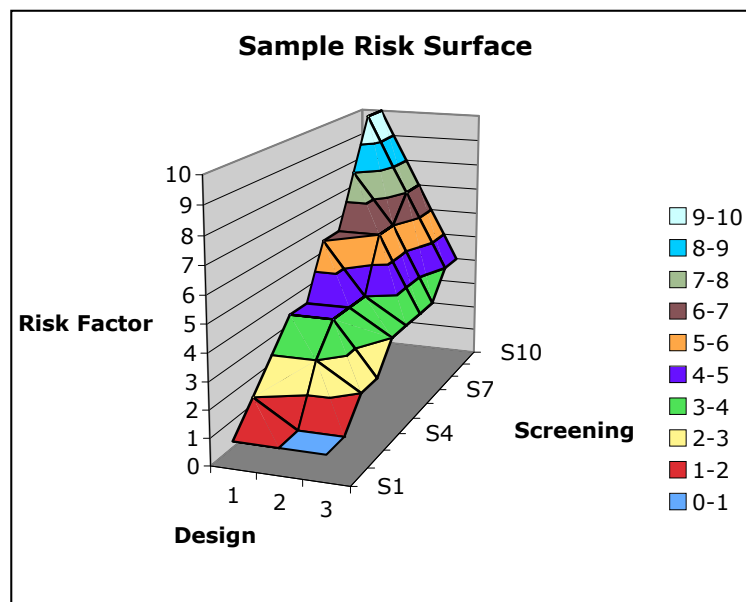

Figure 12. Sample Risk Surface/Space

In Figure 12, regions 1 and 2 could be interpreted as "High," regions 3 to 6 as "Medium," and regions 7 to 10 as "Low." This means the same RF can be obtained through several different combinations of activities. Practically, the assignment of the weighting factors is done in an ad hoc/engineering judgment approach.

Because of the existence of these different mitigation combinations, this risk management approach provides the ability to customize risk. For example, one of the screening activities designed to reduce risk is tri-temp testing. Tri-temp testing provides a time t=0 result of performance and cannot be easily correlated with long-term reliability. As such, its weighting parameter can be assigned to a small number when compared to other screening activities such as wafer-level screening. This means a lower weighted activity can be waived or justified/rationalized without lowering the final risk rating.

The risk matrix provides a decision framework to balance cost and schedule. It is conceptually structured to provide both flexibility and rigor to the FPGA risk management process.

Appendix 1—Power Insertion Review for FPGAs

The importance of understanding the power consumption and related mitigation techniques for modern FPGAs cannot be understated or underestimated. Modern flip-chip packages used in high-performance FPGAs have multiple heat-flow paths and are designed to be as thermally efficient as possible.

The use of the basic "one-resistor" figure of merit thermal resistance—Theta-Ja (Θ_{ja})—in estimating temperature does not accurately reflect the thermal efficiency of modern packages. Alternate and more accurate approaches to obtain T_j predictions have been developed. Examples of this are the boundary condition-independent compact thermal model (BCI-CTM).

In a specific system implementation, the actual component T_j may be different from the arithmetic predictions using the published Θ_{ja}. The prediction depends on the environment and the prevailing conditions in the system. The following equation governs the relationship:

$$T_j = T_a + P * \Theta_{ja}$$

where
Θ_{ja} = thermal resistance between the device junction and ambient
T_j = junction temperature of the device
T_a = ambient temperature
P = package power dissipation

Determining T_j, T_a, and P, representing the thermal resistance in an application, is not easy, particularly for packages with multiple thermal paths. The single parameter Θ_{ja} is strongly influenced by the application environment and therefore does not represent a suitable thermal resistance.

Theta-ja has become the base thermal parameter most engineers gravitate toward when estimating component T_j with known T_a. But for a more demanding, higher wattage component on a large multilayer system board—particularly with other components around it—this approach often leads to an erroneous prediction of T_j.

In a design with loose margins in the thermal budget, the simple prediction using published Θ_{ja} data may not be an issue. Indeed, it will likely lead to a system running at a lower than predicted T_j, because most common board types are more efficient than the largest standardized thermal board. Increasingly, with higher wattage components where margins are tight, "conservative" data may be the difference between selection and rejection of the component in a specific program.

Table A1. Tj vs. board size and layer for Virtex-5 FPGA

Xilinx 35 x 35 mm FF1136-5VLX50T*		Board Size		
		4" x 4" Board	10" x 10" Board	20" x 20" Board
Layer Count of Mounted Board**	4	68.2°C	64.3°C	-
	8	63.0°C	50.9°C	48.3°C
	12	60.4°C	47.0°C	45.7°C
	16	59.1°C	46.6°C	44.9°C
	24	-	45.3°C	44.0°C

NOTE: *Single component considered at 25°C ambient
**All layers have 1 oz Cu with 80% coverage except outer layers that have 2 oz with 20% coverage.
SOURCE: [Garrault 2003]

Total power in an FPGA (or any semiconductor device) is the sum of two components: static power and dynamic power. Static power results primarily from transistor leakage current, the small current that "leaks" from either source to drain or through the gate oxide of the transistor even when it is logically "off." Dynamic power is the power consumed during switching events in the core or I/O of the device and is therefore frequency-dependent.

As transistor size shrinks (for example, moves from 90-nm to 65-nm devices), leakage currents will increase. The shorter channel lengths and thinner gate oxides used at the new process node make it easier for current to leak, either across the channel region or through the gate oxide of the transistor. Modern CMOS technologies provide several different transistor designs to help address this process. Two or three different gate oxide thickness transistors are available.

For a two gate-oxide thickness process, a thin gate transistor will be used for the high-performance. These would be lower operating voltage transistors in the FPGA core. These devices will have high leakage current however. Thicker gate oxide devices are used for the larger, high-voltage-tolerant transistors in the I/O blocks. Some processes also offer a third, medium thickness gate oxide transistor that has much less leakage than the thin-oxide core transistor.

The "midox" transistors are used in the core of the device for non-performance-critical circuits (like configuration memory) or circuits that do not require fast switching times in response to a changing gate voltage (like routing pass gates). These transistors have much improved leakage currents when compared to high performance, thin gate oxide transistors.

The thin-oxide, highest leakage transistors are reserved only for the portions of the speed path that require very fast switching times. Using three different transistors has shown to offer the possibility of improving leakage while continuing to shrink device size.

Dynamic power is defined as:

$$\text{Dynamic Power} = CV^2f$$

where C is the capacitance of the node switching, V is the supply voltage, and f is the switching frequency. Shrinking processes enables FPGAs to have significantly greater logic capacity and higher performance than older devices. This all translates into more nodes that are switching at higher frequencies. All this tends to increase dynamic power for decreasing CMOS process.

There are competing factors however. The core FPGA supply voltage (V) and node capacitance (C) generally reduce with each new process node, providing a reduction in dynamic power when compared to previous generation FPGAs. For example, 90-nm FPGAs will use a core supply voltage of 1.2V. 65-nm FPGAs use a core voltage of 1.0V. Node capacitance tends to decrease because of smaller parasitic capacitances (associated with the smaller transistors) and shorter, less capacitive interconnects between logic. 65-nm processes also use reduced-K dielectric material between metal interconnect layers to minimize routing capacitance.

FPGA vendors can address (and reduce) dynamic power through improvements in architecture. For example, most of the node capacitance that contributes to dynamic power is attributed to the routing or interconnect between logic functions. The Virtex-5 architecture reduces routing capacitance in two ways:

1. Virtex-5 CLBs are based on a six-input look-up table (6-LUT) logic architecture, as opposed to the 4-LUT architecture used in older devices. This means that more logic is implemented within each LUT, translating to fewer levels of logic and thus a reduced need for higher capacitance routing between logic functions.

2. The Virtex-5 routing architecture includes diagonally symmetric routes. This means that every CLB now has a direct "one hop" connection to all of its neighbors, including diagonal neighbors. When a connection is required between logic functions, it is now more likely that this connection is a less-capacitive "one hop" connection, whereas previous routing architectures may have required two or more hops for the same connectivity.

References

Actel, "RTAX-S Testing and Reliability Update," 2007.

Adel, Philippe and Greg Allen, "Assessing and Mitigating Radiation Effects in Xilinx FPGAs," NASA Electronic Parts Program, JPL Publication 08-9, February 2008.

Burke, Gary, "Field Programmable Gate Arrays and Application Specific Integrated Circuits Design Process," *JPL D-29097*, June 2004.

Butler, K., "Tutorial - IC Test for Reliability Engineers," *International Reliability Physics Symposium*, 2006.

Butler, Madeline, "Field Programmable Gate Array (FPGA) Development Methodology," *NASA Directive 500-PG-8700.2.8*, November 2006.

Curd, Derek, "Reduce Power with Virtex-5 FPGAs," pp. 33, *XCell Journal*, Fourth Quarter 2006.

Garrault, Philippe "Methodologies for Efficient FPGA Integration into PCBs," Xilinx White Paper WP174, March 2003.

Eldred, R. D., "Test Routines Based on Symbolic Logical Statements," *J. Assoc. Computing Machinery*, Vol. 6, No. 1, pp. 33–36, 1959.

Ives, Neil A., Martin S. Leung, Gary W. Stupian, Steven C. Moss, Nathan Presser, and Terence S. Yeoh, "Nanoscale Three-Dimensional Imaging: An Innovative Tool for Failure Analysis," *Crosslink*, Aerospace Corporation, Vol. 6, Number 3, Fall 2005.

Rose, James, "Planning and Implementing Project Reviews," *JPL DocID 56973*, March 2006.

Sheldon, Douglas, "Integrated Qualification Strategies for FPGAs," Microelectronics Reliability and Quality Workshop, *MRQW 2005*, December 2005.

Sheldon, Douglas, "Interpretation of FPGA Risk Matrix," *JPL IOM 5141-07-034*, October 2007.