Cross Functional Design Tools for Radiation Mitigation and Power Optimization of FPGA Circuits

Matthew French[1], Paul Graham[2], Michael Wirthlin[3], and Li Wang[1]
[1]University of Southern California, Information Sciences Institute, Arlington, VA
[2]Los Alamos National Laboratory, Los Alamos, NM
[3]Brigham Young University, Proto, UT

***Abstract*- The Reconfigurable Hardware in Orbit (RHinO) project is focused on creating a set of design tools that facilitate and automate design techniques for reconfigurable computing in space, using SRAM-based field-programmable-gate-array (FPGA) technology. In the final year of the project, design tools that have been created to visualize and analyze an FPGA circuit for radiation weaknesses and power inefficiencies were validated and verified in radiation testing and power measurement testbeds. For radiation, a single event Upset (SEU) emulator, persistence analysis and mitigation tool, and a half-latch removal tool for Xilinx Virtex-II devices have been created. For power, dynamic power visualization, analysis, and optimization tools have been completed. In this paper, the final combined test results are presented for an image convolution test circuit with different levels of radiation mitigation and power optimization.**

I. Introduction

SRAM-based FPGAs have become a promising solution to processing on space-based payloads. They offer features that anti-fuse FPGAs do not, such as reprogrammability, embedded multipliers, and embedded processors, while also offering 5-10x more logic gates. These features allow SRAM-based FPGAs to address resource multiplexing, fault tolerance, mission obsolescence and design flaws in on-orbit payloads that directly impact design cost and mission risk, while also providing better processing performance. However, a significant barrier to developing space-ready SRAM-based FPGA applications is the difficulty in designing for the rigorous constraints mandated by the operational environment. Two main issues limit the use of conventional FPGAs to such designs: 1) SRAM-based FPGAs are sensitive to radiation effects, namely, total ionizing dose (TID), single event latchup (SEL), and single-event-upsets (SEUs), because of their high proportion of memory structures; and 2) SRAM-based FPGAs designs tools optimize for throughput at the expense of power.

Current foundry process technology for Xilinx FPGA devices provides enough tolerance for a large number of ESE orbits for total dose and latch up (no destructive latchups have been reported), however the SEU presence is a major design/operational issue. The large amount of static memory within SRAM-based FPGAs, such as look-up tables, routing switch tables, etc., makes them sensitive to SEUs. While traditional hardware redundancy techniques improve the reliability of FPGA designs (at the expense of increases in hardware, power, etc.), novel FPGA-specific techniques are required to address the unique vulnerabilities of SRAM-based FPGA architectures, while incurring less hardware overhead. Therefore, design automation tools evaluating and assessing the reliability of FPGA designs, inserting appropriate redundant hardware, and manipulating the low-level structures of the FPGA design are needed for robust operation and SEU and latch-up tolerance.

Available FPGA synthesis tools optimize for speed or area, but not for real-time power consumption. Limited power estimation tools are available, such as Xilinx's XPower; however, these are difficult to use and have limited utility to the actual FPGA design process. Accurate power estimates are only achievable after completing an entire iteration of the design cycle and provide no power optimization guidance. To make effective use of FPGAs in space, tools providing accurate power estimation and dynamic power optimization, operating on the FPGA's gate logic or on individual configurable logic blocks (CLBs), are needed; specifically: 1) to monitor power consumption early in the design process at a useful granularity (e.g., at CLB); 2) to aid in the design analysis that captures data-dependent transients as well as overall power consumption; and 3) to perform automated dynamic power optimization.

Both the radiation-induced and power consumption effects are currently handled through manual intervention or, at best, through ad-hoc in-house tools. There is a real need in the community for validated design tool automation to raise the technology readiness level (TRL) of SRAM-based FGPA user designs. The RHinO project is leveraging an established, open-source tool-suite that accepts output from commercially available synthesis tools to create tools that allow the developer of a space-based FPGA application to automatically analyze and optimize a Xilinx Virtex II FPGA circuit for both space radiation effects and power utilization.

In the final year of this effort, the space radiation and power tools were validated and verified by running a test circuit through the complete tool flow and testing in the relevant radiation and power measurement testbeds. The remainder of this paper sequentially introduces the relevant tools in design flow chain. First the JHDL tool suite and extensions made to it for the RHinO toolkit are discussed section II. The SEU effects tools are discussed in Section III, and power tools are discussed in Section IV. Section V presents the testing results of the testbench circuit from both a radiation and power perspective. Section VI will summarize the progress and draw conclusions.

II. The JHDL Tool Suite

A. Background

As outlined in [1], the RHinO tools suite is built upon the open-sourced JHDL [2] FPGA design environment. The tool suite, shown in Figure 1, contains a digital circuit simulator,

a circuit hierarchy browser, FPGA library primitives, and tools for exporting user designs into EDIF and VHDL. JHDL provides an open API into the circuit structure to facilitate the creation of application-specific design aids for viewing, revising, manipulating, or interacting with a user design. The integrated design aids, circuit API, and flexibility of JHDL make it an ideal tool for aiding the development of radiation-hardened and power-aware space-based FPGA designs. A variety of application-specific tools can be created to analyze and improve the reliability of FPGA circuits.

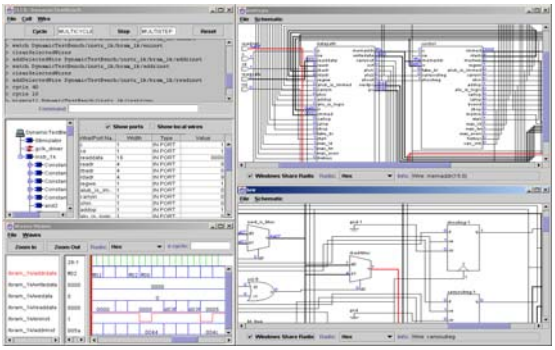


Figure 1. JHDL Tool Suite

Under this effort, RHinO is devising new features for JHDL, specific to space environments, which would enable SRAM-based FPGA payload developers to confidently manage the limiting on-board spacecraft design constraints for power, radiation effects, fault-tolerance, reliability, etc. A key goal of the effort is interoperation with existing commercial tool flows based on VHDL/Verilog, through seamless JHDL-EDIF translation. Alternatively, the user can work entirely in the JHDL design environment, using the RHinO power and SEU tools in concert with the normal JHDL features for simulation, netlisting, and runtime control, all within a single user interface.

B. RHinO Enhancements

During this project the JHDL infrastructure was enhanced and continually refined to support the desired SEU mitigation and power tool functionality. A GUI event API was developed to support intercommunication and interoperability with other modules, or tools that could be dropped into JHDL. As shown in Figure 2, this has led to the development of multiple tool modules being able to leverage the core JHDL capabilities.

Considerable effort was spent enhancing the EDIF netlist tool, originally created to support importing 3rd party IP. The EDIF netlist parser and data structure software provides the central design database for both the RHinO power analysis tools and RHinO design reliability and mitigation tools. These tools provide two important capabilities for the RHinO tool suite. First, these tools provide the capability of *importing* an FPGA design created with a third-party tool into the RHinO infrastructure. Second, these tools provide a consistent circuit database for each of the tools created in the RHinO project.

The relationship between the EDIF tools and other RHinO tools is shown below in Figure 2. An FPGA design is loaded into the RHinO suite through the EDIF parser and into the EDIF data structure. At this point, the design can be manipulated or analyzed using one of several RHinO tool components. For example, power estimates of the design can be made by using the JHDL/RHinO power estimator tool chain. In this mode, a dynamic simulation of the design is created in JHDL to obtain the activity rates of design components and nets. The power estimation and viewer tools are available for browsing and viewing the results of this design simulation. Alternatively, the design reliability analysis tools may be invoked from the EDIF data structure. With these tools, the reliability of the design can be analyzed and presented to the user.

A major goal of the final year of this project for BYU was to strengthen the community of users of the EDIF tool suite. Several tasks were completed to address this goal. First, the web-site for hosting the EDIF code has been expanded to include more documentation and distribution information. This web site, http://reliability.ee.byu.edu/edif/, is the repository for all of the EDIF community resources. Second, an online EDIF forum was created to provide a way to share information among the EDIF users. Third, the distribution process was simplified to provide a variety of distribution options at each distribution checkpoint. Fourth, several API examples were created and posted on this web site. With these new resources, we expect that the EDIF community will continue to grow and encourage more cross organizational collaboration.

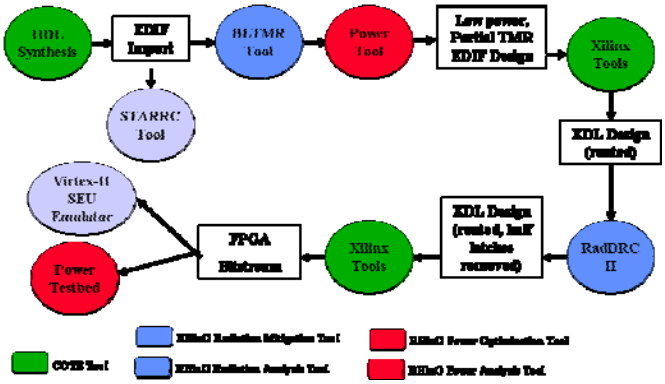


Figure 2. RHinO Tool Infrastructure

III. SEU Radiation Effects

A. Background

To further advance the TRL level of Virtex-II FPGAs for space applications, the RHinO project has a goal of improving the reliability of user designs in the presence of SEUs. SEUs are the main radiation concern since these FPGAs have been shown to have acceptable tolerance to TID as well as to SEL for low earth orbits (LEO). SEUs can occur in several memory structures on these SRAM-based FPGAs [3 4], namely in the support and control logic, the user design state, the programming memory (often called the *configuration memory*), and half-latches. Upsets in the support and control logic can have a range of effects, from fairly benign to totally erasing the contents of the FPGA

configuration memory. Upsets in the user design state, such as in flip-flops and on-chip memories, may cause faults to occur in the user design's operation, while upsets in the configuration memory may change the user's design directly by changing the connectivity of logic or changing the logic functions themselves.

This year, the SEU radiation research focused on four main areas: (1) developing a FPGA design analysis to provide early feedback on the reliability of a given design after HDL synthesis (2) validating a low-cost approach to provide design robustness through selective triple-modular redundancy (TMR) using the BLTmr tool [5] to reduce error persistence, (3) additional characterization of multi-bit upset (MBU) events due to single ion strikes in multiple device families, and (4) additional characterization of half-latch SEUs in Virtex-II. The remaining portion of this section will discuss each of these aspects in the order they appear in the RHinO tool chain depicted in Figure 1.

B. STAR-C and Dynamic SEU Cross-section Analysis

Until now, there have been relatively few ways to estimate the SEU sensitivity of user FPGA designs. At least for methods are possible. First is the engineering "rule of thumb" that states that a design will pessimistically have 1/10 to 1/5 of the configuration bitstream's static cross-section. This method is very imprecise, not very inciteful, and may lead to over or under engineering of space systems.

The second approach, SEU emulation in hardware (like with V2SE), provides a much tighter bound on the actual cross-section. But this approach requires the final placed-and-routed design as well as the hardware setup to actually perform the tests. Further, it may require a significant number of test runs to get very good statistics on the total sensitivity of designs since many errors can be data and operation dependent.

The third approach involves using the Xilinx SEUPPI tool for determing the worst case number of configuration bitstream bits that can affect your design. Past experience has demonstrated that this really is worst case, especially, when SEUPPI has no way of account for the redundancy in the design. Again, you must also have the final placed and routed design available for the analysis.

The fourth, and most common, approach is accelerator testing. Of course, this can be quite costly when trying to determine the precise sentivities of a design since it may be hard to pinpoint the source of a problem since faults are injected randomly and sometimes in unobservable portions of the chip. On the other hand, radiation testing is still a good sanity check.

All of these solutions have their limitations and most are hard to correlate back to the original design. During the last half of the project, we have been developing a tool that can perform SEU cross-section analysis of designs based on analysis of the designs' EDIF. The tool, Scalable Tool for the Analysis of Reliable Circuits (STAR-C), was originally developed for analyzing the reliability of nano-scale computing systems. Though not originally a part of the RHinO project, STAR-C is actually based on the same EDIF and JHDL tools that the power analysis work has used and helped to develop.

For the RHinO project, STAR-C has been modified to estimate the number of configuration bits that cause single-point failures in designs. Basically, the tool has a characterized library describing the potential sensitivities of individual FPGA resources. STAR-C combines this knowledge with a knowledge of the effectiveness of TMR in the circuit to calculate the number of bits in the unmitigated portion of the design. Currently, the tool can analyze NMR/TMR as well as unmitigated circuits for the SEU cross-sections of their logic and memory resources.

Future work on the tool includes efforts to better model the SEU sensitivities of the routing network and to provide designers with the ability to better analyze selective parts of highly redundant designs. Further, we would like to add better visualization capabilities so that designers can find

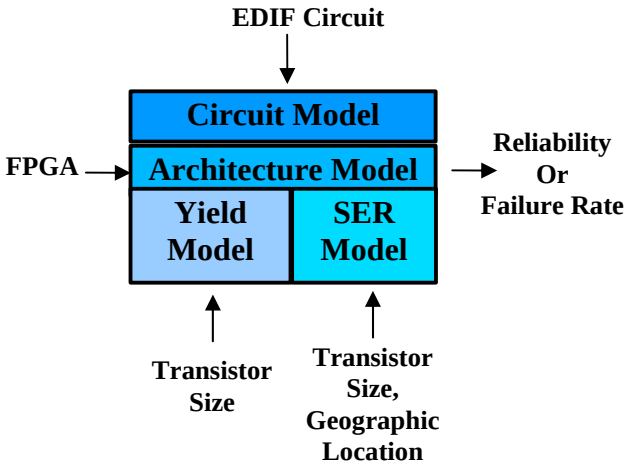


Figure 3. STAR-C Analysis Tool

"reliability hot-spots" in their designs.

C. BLTMR for SEU Mitigation

In Figure 1, after reliability estimation is performed with the STAR-C tool, mitigation techniques can be implemented using the BLTmr tool. TMR can require greater than a three times increase in the amount of logic, I/O and memory used for a design, especially, when considering the costs of voting. This significant resource cost also affects the speed of the design and the size of the chip and/or algorithms that can be used in the system.

As an alternative to this costly approach, we have been researching a technique that prioritizes the design components that are mitigated so that the "most important" components are mitigated first and then, if space allows, other components may also be mitigated. In our case, we have placed a priority on those circuit elements that, if disturbed via an SEU, will remain in an error state for an extended period of time despite scrubbing of the FPGA's programming data (or *configuration bitstream)*—in other words, we are targeting user circuit structures that experience error persistence. These circuits are primarily circuits with feedback, such as finite state machines and infinite-impulse response filters. By emphasizing these feedback structures,

we first reduce the portions of the circuit that would require a circuit reset to bring the circuit back to a "normal" operating state and then focus on portions of the circuit that cause more temporary (or *non-persistent*) error conditions.

The BLTmr tool was developed as a proof-of-concept tool that allows a designer to choose the level of mitigation that is required or desired for a given design, using selective, partial TMR and up to full TMR to provide the requested result while balancing resource and reliability constraints. Originally developed by BYU and LANL for the Xilinx Virtex series of FPGAs, the BLTmr tool was ported to support Virtex-II for the RHinO project. This porting effort involved the creation of an architectural independent layer and architectural specific layers for both the Virtex and Virtex-II architectures. With this architectural independent layer, porting this tool to other architectures should be relatively straight forward. One of the most important improvements was the ability to estimate the device utilization of the FPGA during the partial TMR process. Device utilization estimates are essential for deciding how much TMR to apply for a given design. These estimates, however, are difficult to make at the EDIF level as the technology specific mapping has not taken place. A variety of tool options are provided to allow the user to experiment and control this estimation process. The BLTmr tool is now able to automatically apply partial TMR to any given design in a way that fully utilizes the FPGA resources.

The current version (0.1.2) of the tool allows the designer to select the mitigation approach in several flexible ways, including: (1) The designer can select a certain utilization level of a given FPGA device, using partial TMR as appropriate to maximize design reliability for that utilization level; (2) the designer can instruct the tool to mitigate (a) the feedback portions of the circuit, (b) the input cone of logic for these feedback regions, (c) the output logic of the feedback, (d) or various combinations of the three; and (3) the designer can select to fully TMR the design.

An evaluation [5] of this technique using the original Virtex family of FPGAs indicated for a significant digital signal processing (DSP) design that a 40% increase in slice utilization through partial TMR resulted in a 100x decrease in the number of configuration bits that can cause persistent output errors, leading to a 90x increase in the mean time before failure (MTBF) of the design in a Global Positioning System satellite (GPS) orbit[1]. Further, several levels of costs and mitigation were possible. To achieve a similar level of MTBF improvement for a design that had configuration bits that would only cause persistent errors required full TMR, requiring a 370% increase in slice utilization. The validation of this tool on the testbench circuits is discussed further in the Results section.

D. Multi-bit Upsets in SRAM FPGAs

Multi-bit upsets (MBUs) due to single event are becoming increasing common as the critical stored charged in memory elements decreases with decreased CMOS transistor feature sizes and with the increased transistor densities due to these same decreased feature sizes. With the increased probability of such events, the probability of defeating SEU mitigation schemes increases for many techniques, including simple error control coding and even TMR. Our work with characterizing MBUs has been a step toward understanding the vulnerabilities of TMR due to MBUs.

Previously, we reported on proton radiation results for Virtex-II and other devices. For Virtex-II, 63-MeV proton radiation produces a multi-bit event about 1% of the time while, for the original Virtex devices, MBUs account for only 0.04% of SEU events. Additional work has been done to characterize Virtex-4 for MBUs for 63 MeV protons, demonstrating that Virtex-4 continues the trend for increasing sensitivity to MBUs, experiencing an MBU about 3% of the time for SEU events on an XC4VLX25 and only 2.2% of the time on the XC4VSX35.

Additionally, considerably more data has been taken for Virtex-4 with regard to heavy-ions and MBUs. Figure 4 illustrates that, as expected, the trend toward more MBU susceptibility for small-featured CMOS devices continues, with MBUs approaching 50% of all events for Virtex-4 at an LET of about 57 MeV/mg/cm^2.

Additionally, we have started to quantify the effects of incidence angle of MBUs and have seen with Virtex-4 that changing the angle of incidence from 0 to 60 degrees can increase the percentage of MBUs by factors of 4.5-6 depending on whether the ions are traversing adjacent CLB columns or traveling down a single CLB column, respectively. Similar MBU angle data needs to be collected for Virtex-II to understand the effects.

We have developed Virtex and Virtex-II SEU emulators that emulate the effects of the most common MBU events—2-bit events. Through improvements over the last year, the V2SE can now inject all adjacent 2-bit MBUs for designs, requiring only about four times the time required to do all single bit upsets. We have also recently developed a few ways of emulating triplicated clocks and resets for the V2SE so we can more easily understand the effects of single-bit and multi-bit upsets on TMR designs. Because of this progress,

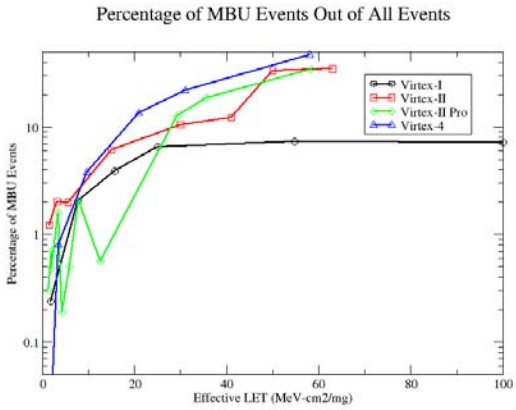


Figure 4. Percentage of MBU Events vs. Effective LET for Four FPGA Generations

[1] Using the following models: AP-8 Solar Minimum, JPL Solar Proton Quiet, and CREME96 Solar Minimum

we are only just now ready to perform a significant amount of SEU emulation to determine the real effects of MBUs on actual user FPGA designs.

In summary, MBUs are becoming increasingly more common in Xilinx SRAM FPGAs due to using ever decreasing CMOS feature sizes. We have yet to see the MBU percentage curves saturate vs. LET and at LETs near 60 MeV/mg/cm^2, more than 35% of all events appear to be MBU events. Additional work is needed to take this information and apply it to actual FPGA designs, including a significant number of fault-injection experiments using the V2SE.

C. Half-Latch Mitigation

To meet the goals for automated SEU mitigation of Virtex-II user designs, the V2SE was employed in validating the RadDRC-II tool, which was developed to eliminate half-latch SEU sensitivities from user FPGA designs. Unlike the results of using SEU emulation for Virtex, SEU emulation for Virtex-II does not seem to upset half-latches as frequently. In fact, there was no noticeable difference between unmitigated Virtex-II designs and those mitigated by RadDRC-II.

Both we and the Xilinx Radiation Test Consortium (XRTC) have done additional radiation testing to determine the effectiveness of half-latch SEU mitigation for Virtex-II. The testing by both organizations appears to confirm that the effect of half-latches on Virtex-II designs is significantly smaller than with the original Virtex FPGAs. In fact, no noticeable difference was detected for mitigated and unmitigated designs that effectively filled the entire FPGAs with a shift register when testing them at Crocker Nuclear Laboratory with 63 MeV protons. In the process, some "lock-up" like events for the designs did occur when the configuration bitstream was fine and the designs were reset, but with a cross-section of 4.16×10^{-13} cm^2 per device, these are not likely to be the half-latch SEU events seen with Virtex. The XRTC have seen similar results, noticing that half-latches appear to recover quickly from SEUs (frequently << 1s).

In summary, half-latch mitigation does not appear to provide a significant level of robustness to designs in the tests conducted so far. However, there may be cases were replacing half-latches with constants directly controlled by the configuration bitstream may still be desirable—especially, when a designer wants to reduce the number of "undetectable" circuit interruptions. Still, the need for half-latch mitigation appears to be somewhat debatable at this point.

IV. Power

A. Background

While SRAM-based FPGAs are widely used in embedded systems and handheld devices because of their short design cycles, and growing performances, the power consumption remains a concern. The existing COTS power tools, such as Xilinx's XPower, have limited functionality, and are difficult for users to extract information for power optimization. The goal of the power tools we introduce here is to yield immediate results on current devices and interoperate with COTS CAD tools. Many FPGA power reduction approaches have been introduced in previous works by other researchers [6, 7, 8], such as low-power VLSI design, LUT-based mapping, glitch reduction, and leakage power minimization. All these approaches address the power issue by either changing the FPGA architecture, or changing the original circuit design. The power optimization methodology developed under this program distinguishes itself from the others by not changing the functionality of the circuit, in order to preserve desired radiation fault mitigation circuit alterations, such as TMR or half-latch removal. These novel power optimization techniques convert power optimization goals into constraints compliant with throughput-based Place and Route (PAR) tools in order to minimize the power consumption of a circuit's routing interconnect.

The power techniques are implemented in the Low-Power Intelligent Tool Environment (LITE). Figure 1 shows the tool flow. The LITE tool infrastructure provides the capability of querying circuit components, running simulations, and tracking signal transitions. LITE consists of four components: power analysis, power modeling, power optimization, and power visualization. The power calibration component interacts with Xilinx CAD tools to extract parameters which are fed into the power modeling component to create post-synthesis level power estimation. The power visualization component displays circuit's power consumption during simulation, highlights power intensive modules, and plots various power consumption metrics of the design. The power optimization component applies the power optimization techniques discussed in the next section. The power optimization techniques in LITE do not modify design logic, but rather feed additional constraints to Xilinx tool to guide the Place and Route process producing low power circuit implementations. A more detailed description of the LITE tools can be found in [9].

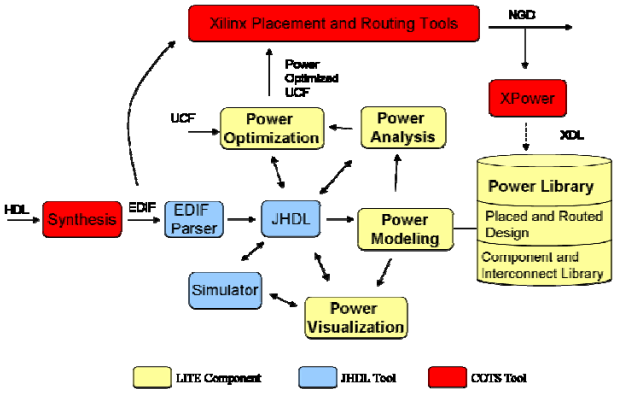


Figure 5. LITE Tool Flow

B. Power Optimization Techniques

In this section we will present and discuss the four power optimization techniques that have been developed in the past year: Slack Minimization, Clock Tree Paring, N-Terminal Net Co-location, and Area Minimization.

i. Slack Minimization

The slack minimization algorithm assumes that the PAR tools will leave each net at or just under the user's specified timing requirements, in many cases leaving slack, or extra net length that could be further tightened to reduce capacitance. For this algorithm slack is defined as

$$Slack = T_{Spec} - T_{Logic} - T_{\min wr} \qquad (2)$$

where T_{Spec} is the user's timing specification, T_{Logic} is the timing delay of any combinatorial logic in between flip-flops on the net, and $T_{\min wr}$ is the minimal wire timing delay. For example, in the left hand side of Fig. 6, a flip-flop to flip-flop path has two intermediate components, with 1 ns and 2 ns individual delay. The user's specified clock is running at 100 MHz, i.e. 10 ns in period. Therefore, the slack of the path is 7 ns. Without additional constraints, the PAR tools will typically meet the maximum delay necessary to still meet the constraints as it should, creating a wire delay of up to 7 ns. If we allow 1 ns delay between each logic element, we can reduce the interconnect length to 3 ns and reduce the interconnect capacitance.

The slack minimization technique uses the LITE analysis component to prioritize high capacitance, high toggle rate nets, calculate the slack, and tighten the timing constraints on these nets allowing for only minimal wire length. In practice, nets with ample slack are typically those with two or less levels of combinational logic between flip-flops.

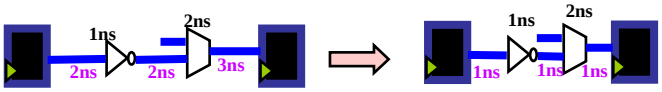


Figure 6. Slack Minimization

ii. Clock Tree Paring

The clock tree paring algorithm targets the largest contributor to dynamic power consumption in most circuits by utilizing placement constraints to minimize the size of the clock net tree utilized. As introduced in Section II, in the Xilinx Virtex-II FPGAs, clock nets are distributed on dedicated routing resources. Through FPGA editor and experimentation, we observe that clock network is like a tree, with the main trunk traveling north to south in the middle of the chip, and branches extending west and east into clock regions. The number of clock regions varies depending on the size of the device. The clock tree is gated such that completely unused branches of the tree are effectively turned off. Therefore by placing logic closest to the clock trunk, clocking power can be reduced by gating more of the clock tree.

In Xilinx Virtex-II FPGAs, clocks are distributed on dedicated global clock lines. Through FPGA editor and experimentation, we observe that clock network is like a tree, with the main trunk traveling north to south in the middle of the chip, and branches extending west and east into clock regions. In clock regions, each clock branch has many sub-branches that connect directly to flip-flops. Figure 7 depicts the clock tree and clock regions in the XC2V6000 FPGA device. The clock tree is gated such that the entire unused branches or sub-branches, can be completely turned off. Therefore by grouping clock-driven logic closer to each other, clocking power can be reduced by leaving more clock branches/sub-branches idling.

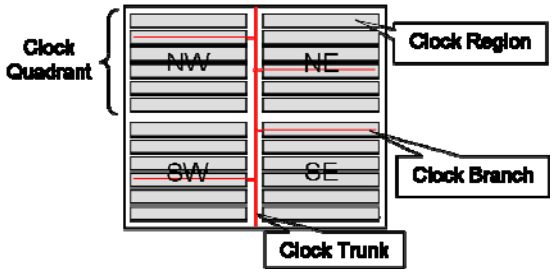


Figure 7. Clock Tree and Clock Regions in XC2V6000 FPGA

The clock tree paring algorithm analyzes a user's circuit, computes a minimum logic resources needed to contain all the logic associated with a clock net, and generates area constraints to specify where the associated clock logic may be placed. The size of the area is determined by a clock's fanout. For multiple clock cases, the LITE tool can prioritize high power consumption clock to be placed. The LITE tool records the locations of placed clocks and prevents over-usage of a region of resources by avoiding area overlaps of multiple clocks. It should be noted that the clock groups do not have to be placed close to the main trunk to save power. In the cases that IO timing is critical, the clock groups can be placed close to IOs to achieve the power benefits.

iii. N-terminal Net Co-location

N-terminal net co-location power optimization is targeted to reduce the power consumed by signal nets. "Terminal" is defined as the sum of the fanin and fanout of a net. For a simplified case, a 2-terminal net is a net with a single fanout. N-terminal net co-location restricts net terminals to be placed in adjacent slices. As depicted in Figure 8, net terminals are grouped in pairs, and for each pair, a constraint is used to restrict the two terminals to be located close to each other, and thus reducing the signal net length and power. The algorithm takes advantage of lower capacitance east-west routing sources, and avoids putting constraints on the nets that would naturally be mapped to dedicated low-capacitance lines such as carry chains, shift registers, and nets that are mapped internally to slices. The nets are sorted and prioritized by power consumption using the LITE power analysis environment to target high-capacitance and high toggle rate nets.

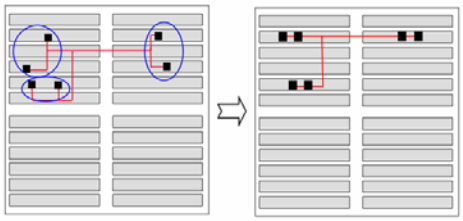


Figure 8. N-Terminal Placement

iv. Area Minimization

The area minimization power optimization technique is based on the observation that routing interconnect lengths highly depend on the placement of components. This technique is expected to work well on circuits that under-utilize the logic available on the chip due to I/O bound designs or poor device size selection. These designs are usually placed loosely over the whole chip by the COTS PAR tools. Consequently, the circuits contain a lot of longer connection wires and hence increase the total net power. By using area minimization constraints, a design is compacted more tightly in a given area of a chip. Net lengths are shortened and thus power is saved. The size of the area is estimated by analyzing and computing the slice count each design element needs by the LITE tools. The sides of the area are proportional to the chip dimensions to balance the north-south bias of the clock trunk.

C. Experimental Results and Discussion

A test suite of ten circuit benchmarks of unmitigated ground based designs was utilized to verify the power optimization algorithms, listed in Table I. This suite represents a fairly wide taxonomy of applications, from glue logic (Mem) to cores (CRC, FM, VGA, USBF, PCI, and DES3) to end to end applications (Conv, S1, and S2), spanning a wide range of device sizes. Each design is mapped into the smallest device that can accommodate the circuit. The baseline power, shown in column 3, is the internal dynamic power of each circuit as reported by XPower, i.e., the dynamic power consumed by logic elements, clock, and signal nets. Figure 9 shows the Slice / IOB utilizations of these designs. Slice occupation ranges from 14% to 86%, and IOB occupation 11% to 90%. All designs also had UCF files specifying I/O pin locations and minimum clocking requirements.

TABLE I
BENCHMARK CIRCUITS AND RESULTS

Design	*Device (XC2V)*	*Base-line Power (mW)*	*Clock Par-ing*	*Slack Min*	*2Term Net*	*Area Min*	*Comb-ined*
CRC	80	31	5.9%	0.3%	-2.9%	1.2%	6.7%
FM	250	102	2.9%	0.0%	-0.4%	0.4%	2.9%
VGA	250	138	12.5%	-0.7%	0.7%	0.4%	12.7%
USBF	500	82	10.7%	-0.7%	-4.5%	0.2%	10.7%
PCI	1000	39	18.7%	0.0%	-3.8%	0.6%	19.4%
Conv	1000	163	4.9%	0.1%	-0.4%	1.2%	7.1%
DES3	2000	139	8.6%	-1.7%	-0.7%	6.9%	8.6%
Mem	6000	643	0.7%	2.1%	0.4%	1.6%	3.3%
S1	6000	251	10.7%	-0.9%	-0.6%	2.8%	10.7%
S2	6000	1020	19.4%	4.1%	1.0%	0.0%	19.4%

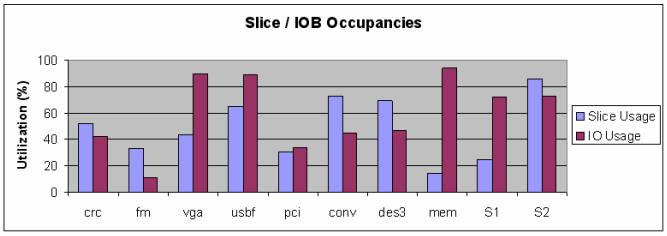


Figure 9. Benchmark Slice/IOB Utilization

The process of power verification is illustrated in Figure 1. The benchmark designs are imported to the LITE tool through the EDIF parser. The LITE tools analyze the circuits and generate new constraint files using the power optimization techniques. The Xilinx tools, with the guidance of the new constraint files, place and route the design to create power optimized implementations. To measure the power results, we use the Xilinx XPower tool with placed and routed netlists and the Value Change Dump (VCD) data created in the back-annotated simulation.

The remaining columns in Table I show the results achieved for each of the optimization techniques mentioned herein. Clock paring proved to be the dominate technique. These four power techniques can be combined to further reduce power consumption. In all the circuits, clock power is the most dominant. So for the combined power optimization testing, we combine the clock tree paring technique with the next best technique for each circuit allowing the additional constraints to fine tune the power consumption. Table II shows the overall positive results for each individual power optimization technique as well as the maximum power reduction achieved when combining the techniques. As shown in the table, 5 out of 10 benchmark designs reach their maximum power reduction by using a combination of techniques. The power reduction ranges from 2.9% to 19.4, and the average improvement is 10.2%. For a more detailed discussion of the algorithms and their results, the reader is referred to [10].

V. Cross Functional Test Results

A. Radiation Mitigation

Validation of the partial TMR mitigation approach and the BLTmr tool, which automates this style of mitigation for Xilinx Virtex and Virtex-II FPGA designs, was a two-stage process. First, unmitigated and mitigated designs were tested using the Virtex-II SEU Emulator (V2SE) [11] to identify configuration bitstream bits that caused errors on the designs' outputs as well as to classify these bits as causing persistent or non-persistent errors. The second stage was to take the designs to a cyclotron and expose them to proton radiation to verify the results of the V2SE.

To perform this validation, the V2SE was updated to test each configuration bit that caused an output error when upset for the persistence of the produced output error. This was done using the following general algorithm found in Figure 10.

Note that the accelerator tests were performed using 50 MeV proton radiation at the Lawrence Berkeley National Laboratory's Berkeley Accelerator Space Effects Facility. The hardware and software used for the tests are directly derived from those used in the V2SE. The capability to correctly "skip" BlockRAM data portions of the bitstream during configuration scrubbing and readback were added to the V2SE application programming interfaces (APIs) to perform this testing. In these tests, one FPGA hosted the *design under test* (DUT) and the other hosted the *Golden* design. The DUT FPGA was exposed to proton radiation while the Golden FPGA produced the input vectors and compared output vectors in real time to identify output errors generated by SEUs. All designs executed at a frequency of 6 MHz (identical to the speed in the V2SE).

Given the set of configuration bits causing output errors, S_{OE}

I. For each $i \varepsilon S_{OE}$, do
   a. Upset configuration bit
   b. Reset design and run
   c. Test for output error
      i. If an output error occurred, repair configuration bit, wait, and test for an additional output error
         1. If it occurred, record as causing a persistent error
         2. Otherwise, no persistent error occurred
      ii. Otherwise, no error occurred and repair upset bit

Figure 10 Error Persistence Algorithm for V2SE

For these validation experiments for Virtex-II designs, we used two different designs. The first, called *synthetic* was a design developed by BYU to specifically test the BLTmr tool's effectiveness at mitigating error persistence due to SEUs. The second design, which we will call *iconv*, is a 3x3 image convolution kernel benchmark circuit representative of NASA ESTO applications.

For *synthetic*, two versions of the design were tested, an unmitigated version (called *synthetic*) and a partial TMR version (called *synthetic_partial*) which allowed the BLTmr tool to utilize as many resources as possible in the device to mitigate configuration bits causing persistent and non-persistent output errors. The *synthetic* design was intentionally created to be too big to fully TMR, consuming 41% of the slices, 10% of the look-up tables (LUTs), and 35% of the flip-flops of a XC2V1000. After partial TMR, the design's slices utilization increased by a factor of 2.42, its LUT utilization increased by 3.25x and it flip-flop utilization increased by 2.68x.

The results of both SEU emulation and accelerator testing can be found in Table II BLTmr Results: Synthetic Design. In the table and in the following discussion, "sensitive" bits are those configuration bits causing output errors if they are flipped while "persistent" bits are those "sensitive" bits that cause persistent output errors. The results demonstrate the BLTmr does effectively target the "persistent" configuration bits when performing partial TMR. The targeted TMR reduced the persistent bits by 63% while only reducing the sensitive bits by 40%.

For the image convolution design, *iconv*, a larger variety of TMR options were explored. Additionally, some of the designs were power optimized using the clock-tree pruning technique. The list of design tested includes the following:

- *iconv, iconv_lp*: These represent the *iconv* design without SEU mitigation. (Note: Design with the "*_lp*" designation are the power optimized versions);
- *iconv_partial, iconv_partial_lp*: These used partial TMR on the feedback portions of the circuit only (not the input or output logic);
- *iconv_full, iconv_full_lp*: These designed were produced using BLTmr's full TMR option.

Note that for the full TMR case, none of the inputs or the outputs were triplicated at the chip-level, so some single points of failure still exist in the circuits. This was intentional since many designs cannot afford to use 3x the FPGA pins.

Table III summarizes the results for the image convolution design for both SEU emulation and the proton accelerator tests. Note that the results from the proton accelerator test have been scaled to account for the fact that BRAM data upsets were not measured during the experiment. The BRAM had to be skipped (as mentioned above) to avoid unintentionally corrupting the data stored in the designs' BRAM.

Table II BLTmr Results: Synthetic Design

		SEU Emulator		Accelerator			
Design	**Total Configuration Bits (XC2V1000)**	**Sensitivity (%)**	**Persistence (%)**	**Sensitivity % (Total events)**	**Persistence % (Total events)**	**Total Upsets**	**% of bitstream upset**
Synthetic	3744736	3.42%	1.65%	5.20% (492)	2.46% (233)	9468	0.25%
synthetic_partial	3744736	2.04%	0.61%	3.05% (375)	1.03% (127)	12314	0.33%

From the SEU emulator results, it is clear that these designs were significantly less sensitive to SEUs in the configuration bitstream to begin with. Further, by comparing the unmitigated *iconv* designs with the *iconv_partial* designs, it is quite clear that the BLTmr tool is successfully

prioritizing what is mitigated based on persistence—in both the normal power and power optimized designs a slight reduction of sensitivity is observed, but the persistence is reduced 41-57%. In comparing the SEU emulator results for the unmitigated and full TMR designs, BLTmr is again clearly operating quite well. Reductions of 90-93% can be observed for both sensitivity and persistence.

In comparing the normal power designs with the power optimized designs, a clear 6.4-10.7% increase in sensitivity can be observed for these designs. Since the power optimized designs started with identical EDIF netlists to the normal power designs, these designs use effectively the same number of resources as the unoptimized original designs. From this it is clear that the main difference between the designs is in the routing between the resources. In the power optimized designs, the amount of clock routing is reduced through placement constraints on the designs. These reductions in clock routing, though, have apparently lead to a significant increase in the average number of configuration bits used per connection between logic resources. Though additional work will be necessary to determine the exact cause, the placement of the design resources into a tighter area may likely lead to less optimal routes in terms of length and types of routing.

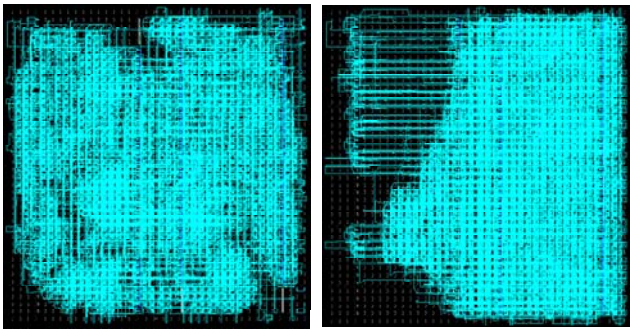
The illustrations in Figure 11 provide a good example of this for the *iconv_full* variations of the design. The *iconv_full* design is distributed through the entire chip, potentially allowing the Xilinx placement and routing tools to reduce average distances between resources. The *iconv_full_lp* design intentionally packs the resources to the left side of the chip to reduce the active portions of FPGA's clock tree. In the process, though, the reader may note a significant number of longer routes in the upper left hand corner of the power-optimized design. These are the routes between logic resources and the embedded hardware multipliers of the FPGA. Though a majority of the design can be packed nicely together, the physical layout of the chip itself causes this packing to increase the length of some routes. This is only an example of how the SEU sensitivity of FPGA designs can be noticeably affected by placement and routing.

With regards to the accelerator results, there is a fairly good correlation between the SEU emulator results and the accelerator results, especially considering the relative small number of events observed and the fact that the SEU emulator does not upset flip-flop state or effectively BRAM state for this design. Though the numbers of events in the data do not provide strong statistically significant results, they do suggest that the results of SEU emulation are on the right approximate order of magnitude. More events were not collected due to sampling frequency limitations—during the tests, an experimenter desires to have about one upset per configuration data readback so that output errors can be more strongly correlated with specific observed SEUs. With the current test fixture, this amounts to about 2-3 upsets per second.

One significant anomaly in the accelerator data does exist, if the *iconv_full_lp* results are compared with the SEU emulator results. Considering the performance of the other designs, there are many possible explanations, but the most likely explanation, at this point, would tend to be timing problems with test setup. As noted earlier, this is the design that is most likely to have timing problems in keeping the DUT and Golden designs synchronized. Even a slight timing problem would skew results considering the number of events involved. Additional work will be necessary to understand this issue.

In summary, the BLTmr validation study has shown that the BLTmr tool is effective in reducing the SEU sensitivity of designs as well as designs' susceptibility to error persistence due to SEUs. Further, the SEU emulator was demonstrated to be very useful tool in understanding the sensitivity and error persistence of designs.

B. Power Optimization Results on SEU-mitigated Circuits



iconv_full (full TMR, no power opt) *iconv_full_lp* (full TMR, power opt)

Figure 11 Layout of normal and optimized designs

Table III BLTmr Results: Image Convolution

		SEU Emulator		Accelerator (BRAM adjusted)			
Design	**Total Bitstream Bits (2V1000)**	**Sensitivity (%)**	**Persistence (%)**	**Sensitivity % (Total Events)**	**Persistence % (Total Events)**	**Total Upsets (estimated)**	**% of bitstream upset**
iconv	3744736	0.70%	0.32%	0.66% (81)	0.14% (17)	12214	0.33%
iconv_lp	3744736	0.78%	0.46%	N/A	N/A	N/A	N/A
iconv_partial	3744736	0.62%	0.19%	N/A	N/A	N/A	N/A
iconv_partial_lp	3744736	0.66%	0.20%	0.61% (50)	0.02% (2)	6291	0.22%
iconv_full	3744736	0.05%	0.03%	0.17% (24)	0.08% (12)	14470	0.39%
iconv_full_lp	3744736	0.05%	0.03%	0.86% (181)	0.12% (25)	21075	0.56%

In this experiment, the benchmark and its radiation-mitigated designs are applied with the LITE tool, and the power improvement results are analyzed. The benchmark is a highly pipelined image convolution engine that enables to process three input data in parallel and create one output each clock cycle. The design is implemented in a Xilinx Virtex2 1000 FPGA, and it utilizes nine multipliers and three block RAMs. Triple Modular Redundancy (TMR) is applied to the image convolution design using the BLTmr tool. For the power analysis, we will consider two partially triplicated versions of the benchmark design using the BLTmr tool. In the first design, only the structures that cause "persistent" errors are triplicated. In this case, about 35% of the original instances are triplicated. The second design is a full TMR version.

The four power minimization techniques and their combinations are applied to the image convolution kernel benchmark. Eight sets of random tap values are utilized to each circuit. Figure 12 shows the results of the maximum power saved using the techniques. The solid lines are the power consumption of the three designs before power optimization: the original convolution engine design, the partial TMR design, and the full TMR design. The dotted lines show the power consumption of the corresponding power-optimized circuits. Power is improved by 8.4% on the original design. The partial and full TMR designs have power reduction of 9.0% and 14.2% respectively.

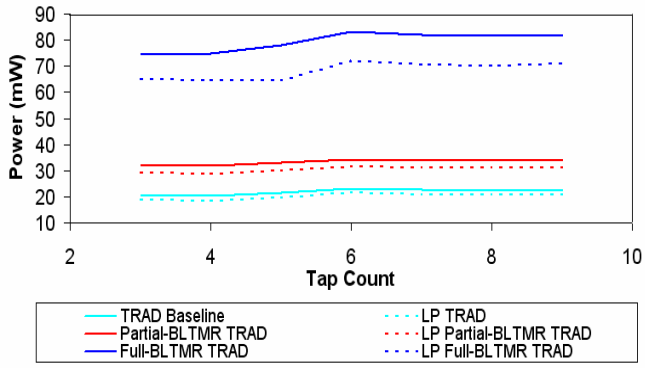


Figure 12. Power Optimization Results of Convolution Kernel

The TMR designs respond well to the presented power algorithms. The 2-terminal net co-location method shows increasingly better results with more TMR, achieving the best power minimization in the full TMR design. This result illustrates that more crowded circuits present more opportunities to optimize signal power. The slack minimization technique reduces the power on all the three designs, however it saves the most power on the unmitigated design. In small circuits that have low Slice occupation like non-TMR'ed design, nets usually have more slack and thus have more opportunity to optimize. The area minimization approach produces positive power reduction on all the designs. However, the clock paring technique works best of all the algorithms.

VI. Conclusions

In the final year of this effort, all the tools were finalized and verified. The image convolution benchmark circuit was mitigated with varying levels of overhead and robustness utilizing the BLTmr tool. Each of these designs was then optimized for power using the algorithms developed. Power optimization was verified using the XPower testbed and radiation robustness was verified using both the V2SE and proton radiation experiments. By selectively controlling the amount of triplication, 25-50% reductions in power and size could be obtained with little impact on persistent error mitigation. Using power optimization techniques another 5-15% dynamic power reduction could be achieved. It was found that the power optimization techniques did result in a minor increase in errors, most likely due to dense packing of the bitstream.

REFERENCES

[1] French, et al., "Design Tools for Reconfigurable Hardware in Orbit," Earth Science Technology Conference 2004, Palo Alto, California, June 2004.

[2] "Adaptive Computing Systems"; 1997- 2003 DARPA effort; see www.jhdl.org

[3] Carl Carmichael, Earl Fuller, Phil Blain, and Michael Caffrey, "SEU Mitigation Techniques for Virtex FPGAs in Space Applications", Proceeding of the Military and Aerospace Programmable Logic Devices International Conference (MAPLD), Sept. 1999, Laurel, MD, pp. C2.1-8.

[4] Michael Caffrey, Paul Graham, Michael Wirthlin, Eric Johnson, and Nathan Rollins, "Single-Event Upsets in SRAM FPGAs", Proceedings of the 5th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD), Sept. 2002, pp. P8.1-6.

[5] B. Pratt, E. Johnson, M. Wirthlin, M. Caffrey, K. Morgan, and P. Graham, "Improving FPGA Design Robustness with Partial TMR," MAPLD '05, Washington, D.C., September 2005.

[6] J. H. Anderson, and F. N. Najm "A novel low-power FPGA routing switch," *IEEE Custom Integrated Circuits Conference (CICC), Orlando, FL, pp. 719-722, October 3-6, 2004*

[7] F. Li, Y. Lin, L. He, and J. Cong "Low-power FPGA using Pre-defined Dual-Vdd/Dual-Vt Fabrics," *Proceedings of the 2004 ACM International Symposium on Field-Programmable Gate Arrays*, Feb. 2004.

[8] J. H. Anderson, and F. N. Najm "Power-aware Technology Map-ping for LUT-based FPGAs," *IEEE International Conference on Field-Programmable Technology*, Dec. 2002.

[9] Matthew French, Li Wang, and Michael Wirthlin "Power Visualization, Analysis, and Optimization Tools for FPGAs," *IEEE Symposium on Field-Programmable Custom Computing Machines*, April 2006, Napa, CA.

[10] Li Wang, Matthew French, Azadeh Davoodi, and Deepak Agarwal "FPGA Dynamic Power Minimization through Placement and Routing Constraints," *EURASIP Journal on Embedded Systems, Special Issue on Filed Programmable Gate Arrays,* 4th Quarter, 2006.

[11] M. French, P. Graham , M. Wirthlin, L. Wang, G. Larchev, "Radiation Mitigation and Power Optimization Design Tools for Reconfigurable Hardware in Orbit", Earth Science Technology Conference, June 2005, Washington, D.C.