Procedures and Guidelines

DIRECTIVE NO.	500-PG-8700.2.7-	**APPROVED BY Signature:**	Original signed by
EFFECTIVE DATE:	08/12/2005	**NAME:**	Madeline Butler
EXPIRATION DATE:	08/12/2010	**TITLE:**	Deputy Chief Engineer

COMPLIANCE IS MANDATORY

Responsible Office: 500 / Applied Engineering and Technology Directorate

Title: Design of Space Flight Field Programmable Gate Arrays

PREFACE

P.1   PURPOSE

The purpose of this document is to discuss guidelines and criteria that form a basis for the specification, design, and evaluation of Field Programmable Gate Arrays (FPGAs) for spaceborne applications. The goal is to help ensure that the design of the hardware is flight worthy by addressing those items that must be considered for a successful and robust FPGA design.

P.2   APPLICABILITY

This procedure applies to the electrical design and development of all Applied Engineering and Technology Directorate flight products that include the use of FPGAs.

P.3   AUTHORITY

GPR 8700.2, Design Development

P.4   REFERENCES

NASA SP-8070, Space Vehicle Design Criteria, Spacebourne Digital Computer Systems

P.5   CANCELLATION

561-PG-8700.2.1, Flight Field Programmable Gate Array Design Guidelines

P.6   SAFETY

NONE

P.7   TRAINING

NONE

P.8   RECORDS

NONE

P.9   METRICS

NONE

P.10   DEFINITIONS

NONE

PROCEDURES

In this document, a requirement is identified by "shall," a good practice by "should," permission by "may" or "can," expectation by "will," and descriptive material by "is."

Introduction

As space vehicle missions have become more complex, the use of onboard digital computers and logic has become more prevalent. The functions that the avionics are assigned to perform are also expanding in number and magnitude. As a result, the problem of specifying and designing digital avionics for space vehicles has increased in complexity.

The flight performance of spaceborne digital avionics has generally, but not always, been successful. A number of recurring problems have been experienced during the design, development, and testing of these machines. Previous systems have been very costly, have required major redesigns, have caused significant schedule delays, or have launched with a needlessly high level of risk. Most difficulties have resulted from:

   a. Poor design/analysis practices
   b. Incomplete knowledge of the newer technologies and tools coupled with their impact on the design and analysis
   c. Inadequate reviews

The material presented in this PG will concentrate on items that are often seen to be problems in space flight digital hardware, particularly FPGAs. This information does not state exactly how to design a particular circuit, perform an analysis, or prepare the results, but instead addresses items that need to be considered for a successful and robust design.

It is tempting to provide a checklist for designs. However, the unique nature of many spaceflight applications as well as unique requirements do not make an exhaustive checklist practical; there are simply too many possible cases and the technology changes too rapidly. The approach taken in this document is to discuss the principles, the guidelines, and the criteria to be used for design and analysis of spaceflight digital avionics. This gives the design engineers the freedom to pursue solutions to fit their unique challenges. An exhaustive checklist, to serve as a mechanical pass/fail test, will needlessly restrict and constrain the design engineer from valid and reliable solutions. That approach is not taken in this document.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 3 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

In addition to what is textually included in this document, you'll find that each of the following sections includes one or more links to additional, web-based material in the form of case studies, application notes, papers, and other material and references.

1. Special Pins

One of the most common problems identified during design reviews is the improper termination of special pins. For every device, the data sheet and design schematics shall be carefully reviewed such that it can be shown that each special pin is properly terminated. Termination of many of these special pins cannot be verified by test.

1.1 MODE Pin: This pin, present on early generation of Actel devices, must be grounded. It is recommended that this pin be grounded with a 10 kohm resistor and a hard jumper to ground in parallel, with the default setting the hard ground installed.

1.2 JTAG Interface: Many modern digital microcircuits have this interface. One optional pin, which is highly desired for high-reliability designs, is the TRST*. If present, this must be hard grounded since the IEEE 1149.1 specification requires a pull-up resistor inside of the part. Use of a pull-down resistor, such as what some designers use for the MODE pin, can result in the TRST* pin's input voltage being at or above the logic threshold. If the TRST* pin is not present, then the TCLK should be a free-running independent system clock with TMS held to a logic '1'. Do not use the system clock as the TCLK input, because during a malfunction, the chip's operational clock input may turn into an output and clamp the clock.

1.3 Unused Inputs: In general, all devices should have properly terminated inputs. For normal CMOS devices, this is a requirement. Certain programmable devices such as FPGAs will often take care of unused pins via software, exploiting the programmable nature of the microcircuit. However, the "fine print" for each pin must be read carefully. For example, in Actel SX and SX-S, clock inputs such as HCLK or the global routed clocks do not have an output stage -- they are special purpose -- and thus shall be terminated by the user. Failure to do so can result in large currents. As another example, unused LVDS receiver inputs should be left unconnected, as advised by the UTMC documentation. Depending on the device, pins labeled as "N/C" may be used for internal purposes and terminating them on the board may result in problems; conversely, not terminating "N/C"'s in certain cases can be bad. Check each pin carefully according to the specification and contact the manufacturer if necessary.

1.4 Test Interface: Many devices have custom test interfaces and will have to be handled on a case-by-case basis. Since they hook up to test equipment, care should be taken in following the manufacturer's instructions. For example, Actel SX-S device test pins should be series terminated. Other device pins, such as inputs, should be terminated; others have internal terminations. There are no general rules.

1.5 Configuration Pins: Ensure that each configuration pin is carefully checked against the latest data sheet. Some pins have very high internal pull-up resistors and can be switched by high-speed signals on the board level; design defensively and ensure that the levels are solid. Also, some configuration pins can naturally just happen to float to the desired state with nominal operation observed. Lastly, beware of special pins such as programming pins that are required to be terminated appropriately for flight.

1.6 Others: Different devices will have different pins and there is no overarching, general rule, other than that each pin shall be checked.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 4 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

1.7 References, Notes, and Related Documents

a. It is critical to ensure that all pins are properly terminated. Some will affect the functionality of the chip and these may or may not be caught in test. Some unterminated pins will have parametric and perhaps long-term reliability effects. As an example of this, please view the following plot showing unconnected clock pins in the total dose environment. clock_unterminated.gif

b. "Special Pins" from "Advanced Design: Designing for Reliability," presented at the 2001 MAPLD International Conference, Laurel, MD, September 2001.

c. "TRST* and the IEEE JTAG 1149.1 Interface," *OLD News* #7, January 2003.

d. "Terminators for Silicon Explorer," *OLD News* #1.

e. "Use of SX Series Devices and IEEE 1149.1 JTAG Circuitry." This white paper reviews basic 1149.1 principles, radiation results on SX Series devices, and finishes with mitigation techniques and design considerations.

f. "GROUND THE MODE PIN NOW!!!!!!!!!!!!!!!!!!!!!," Termination of MODE Pins in Actel Field Programmable Gate Arrays.

g. Issue on the use of the SDI and DCLK pins in some date codes of the RH1020 and RT1020s. Please see the RH1020 Special Pins Advisory and the SDI Report.

2. Input/Output

This section looks at the various aspects of device I/O that a designer should consider.

2.1 Simultaneous Switching Outputs: There are sometimes limits to the number of output pins that can switch at one time; sometimes these are specified by the manufacturer in a data sheet; sometimes it is described in an application note; and sometimes the designer is left on his own. With devices that switch faster and with large pin counts and lower AC and DC noise margins, ground/V_{DD} bounce can be a serious issue. These are a few guidelines to minimize "bounce" issues and items to consider.

a. Use low slew outputs unless needed.

b. Don't group SSOs together; break them up (Xilinx: two for each side of a ground pin).

c. Control number of SSOs through sequencing Example: Do address and data busses need to switch at the same time?

d. For some families, programming "unused" outputs will improve grounding or supply for output stages if terminated to the rail on the printed circuit board.

e. Use buffers, particularly for large memory arrays or long lines. Everything does not have to be inside of the FPGA or ASIC.

f. Avoid sockets.

g. For spare pad locations, pre-wire power, ground, and bypass capacitor connections. "Haywired" power and ground connections will have unneeded inductance.

h. Choose input thresholds wisely.

    1) TTL $V_{IL} = 0.8V$ - very sensitive, try and avoid this setting, as it is sensitive to both ground bounce and ringing.

    2) Some devices offer programmable 5V CMOS or other input voltage threshold options. This does not reduce ground bounce but mitigates the effects of ground bounce

i. Keep clocks physically away from pins that can cause ground bounce.

j. Keep clocks close to ground pins.

k. When Using JTAG and driving board with test data over multiple parts you can induce data pattern sensitivities, particularly with large data busses, perhaps switching patterns from FFFFFFFF $\Rightarrow$ 00000000. This may be an artificial failure, an artifact of the test, but can damage or potentially overstress hardware through a loss of control. JTAG, or the use of reconfigurable components, can be exploited to run ground bounce tests on engineering model and flight hardware.

l. Test cabling, particularly for vibration, thermal/vacuum, and EMI tests will present different conditions for normal bench testing or systems application. Design for the worst-case over the entire project flow.

m. High-speed parts that are "haywired" in will often have non-optimal connections to the power and ground and poor bypassing. This should be done with care and then properly tested, looking for ground bounce affects.

n. For many devices, t_{PD} can be negatively affected by the number of SSOs.

o. Ground/V_{DD} bounce can dynamically affect input switching thresholds, decreasing system noise margins.

2.2 Signal Termination: Ensure that signals are terminated properly.

a. Clock signals shall be addressed with special care to ensure that there is a smooth transition through the threshold. For loaded clocks, perhaps with long runs, reflections may often result in a non-monotonic transitions causing false or double clocking. Note that this may happen on the "inactive" edge. Similarly, overshoot and ringing can also cause false clocking, particularly on the transition to ground.

b. Most manufacturers have tight limits on how far outside the rail a signal may travel, sometimes coupled with maximum times outside of the recommended limits. Ensure good signal quality as damage to I/O's may and has happened.

c. Do plan on termination resistors in advance. Adding them later is painful, will have lower quality than components and traces located on the circuit board, and will be a major headache for the assemblers.

d. Inspect schematics for proper terminations on interfaces such as RS-422, as an example. This is difficult to detect in test as the system may function, to a large extent, without the resistors, although with decreased noise margins and/or increased stress.

2.3 Tri-State Bus Considerations: Do not allow any overlap in actively driving tri-state busses. This will waste power, needlessly generate noise, and stress components. Have a guaranteed off-time between drivers on the bus in the worst-case. Do not allow the bus to float or have slow transition times, as this will increase power and noise and may negatively affect reliability.

2.4 Input Transition Times: Some high-speed or modern devices have very stringent restrictions on input transition times, often being surprisingly tight. Failure to meet the requirements may result in oscillations, multiple clocking, or damage. Simple pull-up or pull-down resistors, with transition times in the hundreds of nanoseconds, may often be unacceptable with modern components. In these instances, use bus hold or soft latch circuits, which will also reduce power. In other cases, some older digital logic families may have outputs that are not compatible with modern devices, with the transition times just being too slow.

Waveform measurement is typically from 10% to 90% but not always; sometimes the parameter measurement method is not specified. Care must be taken when protection circuits are used on signals coming onto a board or into a box.

Laboratory tests have shown that not all qualified devices will meet the data sheet. One case was when a part was "shrunk" and there was a migration to a faster process with oscillations observed. Thus, conservative margins are recommended.

2.5 Shorting Outputs Together: This is sometimes done to increase drive on the board. This should be avoided since it may damage components if the switching speeds are not matched and it can be difficult or impractical to

test this redundant topology. If this needs to be done, the two outputs should be from the same integrated circuit and the manufacturer consulted.

2.6 Pin Assignment: Care and planning is important for pin assignments. First, note the considerations above for simultaneous switching outputs and noise immunity and quiet designs. Take care that clocks and critical signals can be routed on the printed circuit boards for short runs with a minimal of cross-talk, perhaps surrounding it or placing it near to ground pins. Pin assignments that "look pretty" with all the data bits on a bus lined up in a row have been notorious for causing both ground bounce problems on the printed circuit card and routing problems inside FPGAs.

2.7 Mixed Voltage Interfacing, DC Compatibility. and Noise Margins: When mixing devices from multiple families, even from the same manufacturer, extreme care must be taken to ensure that the devices are reliably operated and that there is sufficient noise margin. This may be problematic when substituting parts for either upgrading circuit performance or dealing with obsolescence issues.

For inputs, many CMOS technology devices advertise "TTL compatible" inputs. However, these inputs may in fact differ rather significantly from their TTL counterparts. The first major difference for many but not all devices is the impedance presented to interface when power is removed for the device. As an example, when radiation-hardened CMOS latches were substituted for SEU-soft 54LS373's in the Galileo attitude control computer's memory units, block redundancy circuits failed since the engineers didn't take into account the sneak path through the inputs ESD protection diodes when power was removed. Another related difference is the maximum voltage that can be applied. Some bipolar devices are useful for reliable level shifting from higher voltages to lower ones; CMOS replacement devices will forward bias the protection diodes resulting in unintended current flows and possible damage or circuit failure. Lastly, many CMOS inputs have logic thresholds which are not truly TTL compatible. That is, the TTL V_{IH} specification is often not met, with V_{IH}(max) values of 2.2V, 2.4V, and sometimes 2.5V being specified whereas true TTL devices have a threshold defined by two diode drops, typically in the range of 1.2V to 1.4V. TTL outputs are only guaranteed to drive to $V_{OH} = 2.4V$ so there may be little or even negative noise margins present in these situation. The switching point difference can also lead to circuit failure, depending on the signal integrity. Often TTL outputs, when switching, have a "bump" in the waveform, particularly with heavy and/or long loads. While this "bump" is often at a high enough voltage so that TTL devices operate correctly, the often higher V_{IH} of CMOS devices may result in multiple clocking. Pull-up resistors can restore adequate DC noise margins in these situations if given enough time to settle, which may be quite a while for this passive circuit. Note, however, that TTL $\rightarrow$ CMOS clock interfaces designed in this fashion will often fail since the CMOS input may see multiple transitions resulting in double clocking.

CMOS output stages can also be tricky and subtle device characteristics can cause errors. Check all specifications carefully! For example, many CMOS devices when driving loads are specified at only very low current levels for high or logic '1' signals. However, TTL inputs take substantial currents and do not present the high impedance seen by CMOS FET inputs and the output may be dragged down. For output loads that are a mix of CMOS and TTL inputs, they often must be split to guarantee the high voltage needed for the CMOS inputs, typically 70% of V_{DD}, and the high current needed for TTL inputs, with the lower VIH of 2.0V. Another factor to consider is the structure of the output stage in the CMOS device. For example, some devices will not swing all the way to the high rail and are voltage limited. This may result in some totem-pole current if the p-channel FET in the next input stage is not cut off. Some devices, even with a 5V I/O supply like the RT54SX series, will only drive outputs to the core voltage of 3.3V, making this CMOS output incompatible with 5V CMOS inputs on the same board! This was fixed in the 2.5V core RT54SXS series, with full 5V voltage swings when supplied with a 5V I/O bias.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 7 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

Components today can typically have many supply voltages of say 1.5V, 1.8V, 2.5, 3.3V, and 5.0V. There are also an abundance of I/O standards with the newest devices being very programmable so their characteristics are not obvious or even knowable from a circuit schematic. Thus, I/O compatibility must be carefully verified, particularly when substituting "new and improved" devices or alternate devices.

2.8 Power Switching and Cold Sparing: When designing system with blocks that are independently powered, for either redundancy or power savings mode, considerable care must be taken. Many CMOS devices present a low impedance when powered down through either the intrinsic or ESD protection diodes; others, with cold sparing inputs, may have high input impedance that is suitable for operation. For programmable devices, selecting 3.3V PCI compatibility, as one example, can result in a "cold sparing" device no longer being high impedance since a clamping diode will be enabled. While many bipolar devices are compatible with cold sparing architectures, some devices have a sneak path to V_{CC} through the output.

2.9 References, Notes, and Related Documents
    a. "SX-A/RT54SX-S SSO Preliminary Results," October 2, 2002, Actel Corp. sso-10-1-02_actel.pdf
    b. "Input Transition Times for SX-S FPGAs," *OLD News* #3, June 24, 2002.
    c. "Input Transition Times," Section 6 of **Programmable Logic Application Notes**, November, 2000.
    d. "Supply-Voltage Migration, 5V to 3.3V." Covers background on processing technologies with implications for supply voltages, distributing multiple supply voltages on a PCB, interfacing between devices operated at different supply voltages, supply voltage sequencing considerations, and migrating designs. xapp080.pdf[XL]
    e. "Input Stages," presented at the 2001 MAPLD International Conference, Laurel, MD.
    f. Typical bus hold circuit. This version exploits the general purpose I/O structure in an FPGA.
    g. Slow transition times on the clock input of an RH1020 shows oscillation, although the rise time is less than the specified 500 ns. For this series of tests, the conditions were room temperature and $V_{CC} = 5.0$ V. Oscillations detected consistently at $t_R = 360$ ns and sporadic output pulses at $t_R = 300$ ns. Note that the transition time performance of the input stages were not symmetric with oscillations detected consistently at $t_F = 1.5$ µs and sporadic output pulses observed at $t_F = 1.0$ µs.
    h. With the input held at the threshold level, representing the case of a floating input, an RH1020 input stage break's into full oscillation, as seen on the output of the device. For some input stages, the oscillation is not easily seen on the input pin but will propagate within the device.
    i. This example shows the multiple clocking of an RT54SX16 input. A zoomed in view.
    j. "Signals Into Unpowered CMOS" provides additional discussion.
    k. "A radiation-hardened cold sparing input/output buffer manufactured on a commercial process line," Benedetto, J.M. Jordan, A., Radiation Effects Data Workshop, 1999, Location: Norfolk, VA. pp. 87-91. *Abstract*: The radiation hardness of a cold sparing buffer manufactured on a commercial process line is demonstrated. The buffer is shown to be resistant to total dose ionizing radiation and immune (>128 MeV-cm^2/mg) to effects from heavy ions such as single event upset (SEU) and single event latch-up (SEL)
    l. "Input Transition Time," *Introduction*: Inputs to most CMOS inputs have rise and fall time limitations for reliable operation. Although most if not all programmable logic devices have at least some hysteresis on their inputs, the transition time requirements vary considerably. Below is a table with input transition time requirements for many military and aerospace programmable logic devices.
    m. The specifications for inputs must be carefully read as not all device or MCM inputs are truly TTL compatible.
    n. The note "Signal Integrity: IBM Luna C DRAM" gives examples of the requirements for signal integrity, noise levels, and included not only logic signals but the power line. Note that for this device, non-monotonic switching on the control lines may result in unpredictable results.

o. "Designing For Signal and Power Integrity in FPGA Systems," Mark Alexander, 2002 MAPLD International Conference, Laurel, MD, September 2002.

p. "Drive Strength of Actel FPGAs," ***Introduction:*** Many modern CMOS digital microcircuits have very strong drivers; the device characteristics have changed over the years. Another change is the widespread use of HDL synthesis for logic generation and simulators for logic simulation. These simulators do not replace the need to perform proper **electrical engineering** of spaceborne digital electronics, in particular signal and power integrity.

q. "IBIS Models and Simulation," presented at "Design Seminar on Actel SX-A and RTSX-S Programmed Antifuses," Tuesday, April 13, 2004, NASA Goddard Space Flight Center. Review of IBIS and tools along with flight design samples used as case studies.

r. "The Effects of Slew Rate on SX-S Series FPGAs," July 18, 2004.

3. Clocks

Clocking, finite state machine design, and timing analysis are all intimately interrelated. This section will discuss some things to check for clocks and design criteria.

3.1 Use of non-low-skew clocks: Many designers, when first designing with programmable logic, will use "regular" routing resources for clocks, assuming that they are zero-skew paths, such as that found [to a certain extent] on circuit boards using discrete digital microcircuits. However, even though a "net" on a schematic or a clock signal in HDL text appears to be a constant signal, the electrical design of the net must be analyzed. In general, when designers use the non-low-skew resources, the chip may more or less "work" with perhaps some unexplained glitches or a poor "programming yield" that is susceptible to specific routing. So, when sequentially adjacent flip-flops are clock on a common edge, ensure that low-skew clock resources are used. It is acceptable to design with non-low-skew clocks and this can often result in a reduction of power or an effective increase in the number of clocks available. However, careful skew-tolerant design techniques and analysis must be used.

3.2 Chip-to-Chip Timing Strategy: Many analysis tools are good at analyzing logic within a single chip. However, many are ineffective at analyzing system or chip-to-chip timing. It is tempting to simply use a low-skew clock on a board to hook up various digital devices. However, that is not always guaranteed to work and the proper analysis must be met. In addition to setup time, which is based on a clock period, hold time analysis must be analyzed. This is often overlooked or done improperly. While the worst-case behavior of the clock-to-out of the source chip is easily analyzed using "minimum" or "best case" timing parameters, the hold time of the sink chip must be analyzed assuming a slow path for the clock and the fast path for the data, for the same calculation. Automated tools often do all min or all max but are not capable of doing min-max analysis; often the human must complete this. A good goal for the sink chip is to have a hold time of 0 ns or less (negative hold) but many devices, particularly some models of FPGA, do not satisfy this condition. So, alternate techniques for passing signals must be used, such as opposite edge clocking, treating signals as asynchronous (not preferred), etc. The criteria for passing is that all worst-case setup and hold times are always satisfied or that sufficient metastable state protection is included. Note that some parts marked with a specific speed grade may actually be faster then marked and minimums from the fastest family should be used in the worst-case analysis. Binning criteria for many chips is single ended; a sample timing path is not to exceed some threshold. It is not uncommon for devices that are a speed grade or two faster to be marked as "slower," since device markings are a contractual requirement and devices marked as faster will not be accepted during incoming inspection.

3.3 Clock Tree

3.3.1 Use of DLLs and PLLs: DLLs and PLLs can have many useful functions in digital systems. However, they have some requirements that must be satisfied. First check that the worst-case frequencies (both slowest and fastest) are compatible with the circuits; often the acceptable ranges are very limited.   Additionally, there are often signal quality conditions that must be satisfied.   Next, when these circuits clock finite state machines or other sequential logic, note the time to lock and stabilize for these circuits and ensure that the device and system powers up safely. Another item to check is the worst-case performance when the DLL or PLL is hit by an SEU. This can result in a change of programming of the DLL or PLL, which is sometimes a little subtle, or a change in mode." Safe operation of the system must be ensured during these off-nominal conditions.

3.3.2 One Root for Each Oscillator: A diagram should be drawn showing the clock trees for the circuit, with one root for each of the oscillators. These diagrams should include PLLs, DLLs, all chips in the clock domain, and all chips that "talk" to the chips in the clock domain. The latter is for the identification of signals that are asynchronous.

3.3.3 Show Logic Blocks and Signals Crossing Clock Domains: Based on analysis of the clock trees, identify all blocks and signals crossing clock domains and determine the need for metastable state resolution. Additionally, ensure that the latency involved in signal synchronization is tolerable to the system.

3.3.4 Duty Cycle Analysis for Opposite Edge Clocking: For designs passing data from one edge of a clock to the other, ensure that the worst-case duty cycle for each phase is properly computed. Often designers will assume a 50% duty cycle which is not the case. Sources of duty cycle distortion include oscillator characteristics with 50 +/- 10% duty cycles being common; uneven delays through logic gates and buffers, etc.

3.4 Asynchronous Interfaces and Failure Rate Calculations for Metastable States: Ensure that proper synchronizers are used for each asynchronous signal. Often designers will simply use two series D flip-flops. While an often used and acceptable topology, for very high-speed circuits for the technology in question the failure rate of this synchronizer is non-negligible; the calculations must be done for these situations. Also note the conditions for which the flip-flop's metastable parameters are taken, with large differences possible in resolution time when moving from nominal temperature and voltage to the extremes. Ensure that there is a lot of margin in these circuits as they are impractical to test and verify. Also note that for ASICs, different flip-flop macros may have significantly different metastable parameters. This can also be a consideration in FPGA. Some discrete devices that are "metastable state hardened" used to be available. Modern flip-flops are pretty good but are not a magical solution.

3.5 References, Notes, and Related Documents
   a. "Clock Skew" from "Logic Design: Clocking, Timing Analysis, and State Machine Design," presented at the 2002 MAPLD International Conference, Laurel, MD, September 2002.
   b. "Clock Timing and Skew: Real Devices" from "Logic Design: Clocking, Timing Analysis, and State Machine Design," presented at the 2002 MAPLD International Conference, Laurel, MD, September 2002.
   c. Skew-Tolerant Circuit Design, David Harris, Harvey Mudd College © 2001 by Academic Press ISBN 1-55860-636-X.
   d. Start times of oscillators may be a function of power supply rise time and may not start up clean. Example with a 50 ms power supply rise time. For the same oscillator, this is a summary of performance over a range of rise times.

e. "Some Characteristics of Crystal Clock Oscillators During the Turn-On Transient." This application note discusses and shows what the output of an oscillator may be during the turn-on transient. Examples shows include runt pulses of various sizes and polarities.

f. "Startup Transient," from **Advanced Design: Designing for Reliability**, 2001 MAPLD International Conference, Laurel, MD, September 10, 2001.

g. Timing Analysis of Asynchronous Signals

h. Discussion of Metastable States

i. "Clock Skew and Short Paths Timing," Actel Corporation, March 2004.

4. Finite State Machines

Finite state machine design can become a religious issue. What is the best style for a finite state machine? Should the human or the machine perform state assignment? How do we design safe finite state machines? There is no best answer for all situations and there is no magical style to be checked. It does, however, have to follow the basic principles of good logic design. It is noted that many engineers now use HDLs to design the state machine and never see the logic. This must be done with extreme care for critical applications.

Some design guidelines for high-reliability circuits mandate that one-hot state machines not be used since the larger number of flip-flops increase the probability of an SEU. While true, a careful analysis of the state machines shows that the one-hot topology has a Hamming distance of 2, making all single bit errors detectable. By comparison, an equivalent binary coded state machine has a Hamming distance of 1 and, while not having any lockup states if all 2^n states are used, you can not detect illegal transitions without an extra checking mechanism, which itself will be subject to SEUs and together will have lockup states.

4.1 Strategy and Analysis of Lockup States

4.1.1 Schematic-based machines: For critical state machines, the analysis must cover all possible logic states and demonstrate that the machine behaves in a deterministic and desired fashion. The analysis must consider off-nominal events. One such example is an SEU, a credible failure mode. Finite state machines in many consumer-grade IC's do have lockup states, such as SDRAMs. For critical controllers this is not acceptable. Analysis should include all possible 2^n states. It is a credible failure mode to be in any of these states as a result of a disturbance on the power bus, an ESD event, etc. Any high reliability machine must be robust under all credible failure modes. Additionally, one must ensure that the FSM starts out in a legal state and than transitions through the desired sequences. One method is to use a power-on reset (POR) indicator. This must be checked to ensure that it is synchronous with the clock. It is not necessary to have the POR go into the asynchronous or reset input of every flip-flop. Indeed, this is often undesirable as it increases the load on the reset signal distribution and makes it tougher to meet removal times for all flip-flops in high-speed circuits. Indeed, one may not need any reset for a finite state machine if it can be shown to always go into a desired state. This can be done in the trivial case of a divide by n master counter, for example, where a reset is not needed and a fault on the reset line can halt the machine. Another technique is to gate the inputs with the POR signal and design an FSM such that it is guaranteed to go into a hold state. One consideration with the reset function is design-for-test and design-for-simulation, which sometimes results in additional reset connections.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 11 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

4.1.2 HDL Synthesized Machines: Obviously, all of the criteria for schematic-based machines apply. However, there are special considerations for designing with HDL, as the CAE writer might generate circuits that are not desirable for high-reliable circuits. Hence, for critical circuits, one must examine the output reports from the synthesizer very carefully. Common things to check for include: lockup states; outputs of Gray encoded machines that can glitch: unintended flip-flop replication; not implementing the desired and specified style (sometimes the synthesizers just think they know better than the human and will substitute one type of state machine for another). Additionally, some logic synthesizers will generate "safe" state machines. Examine the generated design carefully. For instance, it has been seen that sometimes the logic will explode with excessive gates. Other times there are resets generated on the opposite edge of the clock resulting in tight timing for the removal of clears which are not visible to the designer. Note that languages such as VHDL can not cover all physical states, just logical ones. Hence, the "others" clause will only refer to states in the enumerated type and not the physical realization. The HDL doesn't know if it is a one-hot or binary or gray coded implementation and what flip-flops have been replicated. This is not detectable at the black box simulation level nor by Boolean equations for logical equivalence.

4.2 Flip-Flop Replication:

Logic designers often replicate logic for reliability or performance reasons. For example, if the load on an output is too high, then the load will often be split between multiple drivers (in some cases outputs may be joined together but this is not preferred and is usually avoidable). In other cases, cutting the load and duplicating the driver can help make timing by distributing the capacitive load. The replication of combinational logic is quite straightforward.

However, if this concept is extended to sequential logic and finite state machine design then the situation is trickier since state information is involved. Indeed, the logic may present different information to different parts of the circuit and, for example, may be inconsistent in the presence of a transient fault such as a single event upset, ESD event, etc. That is, the logical flip-flop can present different values to different parts of the circuit depending on which physical flip-flop they connected to. This is a call for caution in high-reliability applications. Software CAE tools are more than happy to generate circuits of this class and do not generate logic to ensure self-consistency. Examples are given in an application note.

4.3 Error Detection and Correction Requirements and Implementation: It is often tempting to design robust state machines by simply appending a Hamming code and correction circuits. If you realize that SEUs or ESD events are not synchronized to the system clock and that the logic network is not guaranteed to be glitch free, then you will have second thoughts about the ability of this type of structure to provide robust operation. In the general case, you must analyze the combinational circuits which implement the next-state logic and their inputs to the flip-flops making up the state register. In particular, for any of these schemes, you must look at whether or not the circuit implementations are static hazard free and, if not, can an erroneous transition to a state (or set of states) be made.

4.4 References, Notes, and Related Documents
    a.   For visibility into the operation of the system, debug, and test, bring FSM state flip-flops to spare I/Os and test points.
    b.   "Sequential Circuit Design for Spaceborne and Critical Electronics," R. Barto, presented at the 2000 MAPLD International Conference.
    c.   "Logic Design: Clocking, Timing Analysis, Finite State Machines, and Verification," Presented at the 2002 MAPLD International Conference, Laurel, MD, September 9, 2002.

    d. Flip-Flop Replication. This application note gives an introduction to the topic and examples. Cases examined are VHDL synthesis, netlist translation, and backend place and routing.

    e. "XC4000XL/Spartan PAR - Router duplicates registers for use as output-to-output route-thrus,"[XL] Xilinx answers database #3813.

    f. "Asynchronous & Synchronous Reset Design Techniques - Part Deux"

    g. "Startup Transient," from **Advanced Design: Designing for Reliability**, 2001 MAPLD International Conference, Laurel, MD, September 10, 2001.

    h. Analysis of POR Circuit Topologies

    i. Discussion of MetastableStates

    j. Timing Analysis of Asynchronous Signals

5. Reset

5.1 Reset logic circuit (consider transient behavior): Transient effects must be considered on the reset circuit; indeed, this must be the focus of the analysis. For the application of power, the output of the POR or reset circuit should ideally be a solid logic level and be glitch-free. Inrush currents to timing capacitors must not exceed the maximum for that capacitor type. Rise times to logic gates, if used as a comparator, must not exceed the input's specifications; often gates with hysteresis inputs are used. Note that even with that type of input, output glitches may occur and several stages of logic gates must be used. The most robust solutions often utilize a comparator. Another transient factor to consider is the rise time of the flight power supply, both best and worst cases. These will often differ substantially from laboratory supplies and may be non-monotonic or have substantial overshoot and ringing. Note that flight power supplies are often slew-rate limited to minimize conducted emissions on the power bus. The time constant of the supply may exceed that of the POR circuit! For discharge, ensure that there is a low impedance path for timing capacitor discharge and that the inputs of logic gates are protected. Most CMOS inputs, but not all, have ESD diodes from the input to the supply rail. Discharging a large capacitance through that input may damage it. Also, consider the requirements and response of the circuit to momentary disruptions on the power bus. While many circuits may recover or be recoverable from a power-on reset, this is not true for all circuits. One such example is non-volatile, erasable memories, which need to be carefully protected.

Many diagrams of reset circuits show "asynchronous application, synchronous removal" of the reset circuit. However, note that for many devices, in particular many programmable devices, the inputs can be blocked or ignored during the power-on transient. This may be because of the need for charge pumps to start or configurations to be loaded and then released. For devices with synchronized inputs, the clock oscillators must start, perhaps taking many tens of milliseconds, before the reset can be applied. Outputs of these devices must be handled at the system level as the reset, which may look just fine on the schematic or in the HDL code, will be ignored by the real circuits.

Steady state or DC effects are also important. Check the leakage currents of timing capacitors and logic gates, as the amount of leakage current times the resistance of the timing resistor may result in a voltage drop that eliminates all noise margins.

5.2 Tree: Drawing a tree of all of the reset sources is often helpful in ensuring that the reset logic is well defined. Often there are multiple forms of reset from system resets, software resets, watchdog timers, etc., and having a good tree diagram shows the relationships between them. Ensure that proper synchronization is made when required. Additionally, if the reset needs to be activated fast, for instance to protect non-volatile memories from false writes, or other circuits from initiating one-time events such as firing pyrotechnics, the tree will help ensure that the logic and delays are well understood.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 13 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

5.3 Reset active time vs. startup of components such as FPGAs and crystal clock oscillators: Ensure that the guaranteed reset time is sufficiently long for all circuits in the system. This is often overlooked. For instance, many FPGAs require time to "start," where the charge pump must build up voltage and charge internal capacitances, wait for a delay, and then release it's outputs. Premature release of the POR signal may result in an indeterminate state. Other FPGAs may require a sequence of resets for proper loading and release, with many circuits having internal power-on reset circuits. The timing of all of these resets must be analyzed for best and worst-case behavior. Additionally, some standard components on digital logic boards such as crystal clock oscillators can have a substantial startup time, often many tens of milliseconds. Complicating this further, components such as FPGAs and crystal clock oscillators may have startup times that are a function of the rise time of the power supply. Even worse, this behavior is often poorly specified or not specified at all. Robust start times are critical.

5.4 Protection of signals for mission critical or one-time events (i.e., pyrotechnic initiation): This topic was partially addressed in B, above. However, because of the importance of this signal, it also earns its own section. Note that many logic elements do not follow their truth tables as the power supply ramps up. Thus, the POR signal must act as a gate, blocking false signals during the power supply rise time transient and then release after all circuits are stable. On the other side, when the power comes down, the POR circuit may need to be asserted early, ensuring that critical circuits are safe before the logic elements lose control as the voltage drops. Devices that often need protection are pyrotechnic initiators, EEPROMs, flash memories, etc. Note that some devices such as microcontrollers have internal flash memories, so evaluate all components and system interfaces for necessary protection by the POR signals.

5.5 References, Notes, and Related Documents
   a. Start times of oscillators may be a function of power supply rise time and may not start up clean. Example with a 50 ms power supply rise time. For the same oscillator, this is a summary of performance over a range of rise times.
   b. "Some Characteristics of Crystal Clock Oscillators During the Turn-On Transient." This application note discusses and shows what the output of an oscillator may be during the turn-on transient. Examples shows include runt pulses of various sizes and polarities.
   c. "Asynchronous & Synchronous Reset Design Techniques - Part Deux"
   d. Timing Analysis of Asynchronous Signals
   e. "Analysis of POR Circuit Topologies"

6. Hazard Analysis

6.1 Discussion

A *static hazard* exits when a change to a single variable to a combinational network causes a transient or momentary change in other variables to occur, which should not occur (e.g., $1 \rightarrow 0 \rightarrow 1$) then a static hazard is present. Normally this is not a problem in synchronous design as long as there is sufficient time for the signals to settle. A similar condition, a *dynamic hazard*, exists if there is a transition of the form $1 \rightarrow 0 \rightarrow 1 \rightarrow 0$. That is, it did not switch cleanly. Any circuit free of static hazards will be free of dynamic hazards. *Essential hazards* are out of scope of this discussion.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 14 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

This topic is not covered in many logic classes and with the use of HDLs and functional simulation many designers are not familiar with these concepts. For 100% synchronous designs with a single clock and a common edge there are normally no concerns. Yet during reviews hazards are often present, unknown to the designer. One example of this is the use of TMR circuits to generate a clock signal to a finite state machine. The change in one input to the voter, there to mitigate the effects of SEUs, can result in a double clock from the "glitch" coming out of the voter, unless the voter is hazard free. Often a component will appear to be hazard free; one must look carefully at the implementation in the logic family that you are using. For example, are multiplexers, the foundation of some FPGA families glitch free? There is no guarantee that they will be and hence can not be considered safe clock generators without a lot of care. Another example is when a voted output is brought off-chip and used as a clock input for an external device. Logic synthesizers have been observed to generate hazards in the circuits they generate, unknown to the engineer running the tool.

6.2 References, Notes, and Related Documents
   a. "Hazards," from **Advanced Design: Designing for Reliability**, 2001 MAPLD International Conference, Laurel, MD, September 10, 2001.
   b. **Analysis and Design of Digital Circuits and Computer Systems**, Paul M. Chirlian, Stevens Institute of Technology, ©1976. pp. 261-264.

7. Power Systems

7.1 Supply Sequencing (some devices can be damaged by incorrect sequencing): There are two major cases to be concerned with here. This section will cover multiple supplies for a single device. The next circuit will cover the interface between multiple devices. Many of the newer technology devices require two or more power supplies. Often these are divided into supplies to power the core of a logic device and a second supply to operate the Input/Output cells. Additional supplies may be needed for PLLs and DLL's, special I/O standards, or various bias supplies such as external charge pumps. It is obvious that the supplies must meet all of the DC standards as well as ripple characteristics, particularly for circuits such as PLLs. What is often not obvious is that the sequence that power is supplied to a single device can, in certain cases, affect both circuit behavior and performance as well as reliability. For certain devices, such as SX-S series devices, if the I/O supply is brought up before the logic core, then a large inrush current may be present; this would not be the case if the order of the supplies was reversed. For certain devices, incorrect power sequencing can result in overstress or damage. This is the case for multiple vendors. Often the requirements for sequencing are in either application notes or the "fine print." One case to consider is when the oscilloscope probe "slips" and the ground ring momentarily contacts either a power lead or a capacitor terminal, momentarily shorting a supply. This may result incorrect sequencing and can in principle overstress or damage certain chips. In general, avoid parts with power supply sequencing requirements. When present, they should be flagged and the design should be done very carefully, incorporating circuit protection as required.

7.2 Signals Into Unpowered CMOS I/O's: Similar to A, above, the power supply sequencing between interfacing IC's, either on the same or separate boards, must be carefully considered. Many IC's, particularly CMOS ones, present a low impedance to the system when powered off. Most of these IC's require that the power supply be brought up prior to the application of signals on either the inputs or the outputs (many FPGA outputs also have inputs active in the general purpose I/O modules). Some programmable IC's are not analyzable by inspection; the particular design configuration must be known. For instance, some I/O modules provide for cold sparing; that is, they present a high impedance to the system when powered off. That same I/O, configured differently, may have clamp diodes switched in while powered off for PCI compatibility. The design details are often needed to do a proper worst-case analysis.

7.3 Startup Current Transients: Startup current transients are common in many modern devices. The size of the current can be a function of time between power cycles, temperature, ramp rate of the supply, radiation exposure history, power supply sequencing, etc. These currents can be rather large for certain devices, often as high as several amps. It is critical that the power supply systems do not limit current in these cases to steady state levels with margin as insufficient current during the startup sequence can result either a failure to properly initialize, power device shutdown or recycling in an infinite loop, or a system lockup, the deadly embrace. Similarly, some parts have hard restrictions on minimum and maximum power supply rise times; failure to meet these levels may result in circuit failure.

7.4 Bypassing and Distribution: Modern logic devices can be rather large, consisting of multiple millions of gates. Synchronous design techniques, high operating frequencies, and large I/O counts can result in a challenge to the power distribution and conditioning system. Most of the manufacturers supply details in application notes. While some of their recommendations may seem like overkill with a large number of bypass capacitors, at times consisting of multiple capacitors of different values, it is understood that these notes are not written to make their parts harder to design with. These rules should be followed unless suitable care is given to the analysis and test of the system for worst-case conditions. Reconfigurable logic can be exploited to generate worst-case patterns to ensure high-fidelity power and then replaced with the flight application. JTAG interfaces may also be used and care should be given that the JTAG test patterns do not violate design limits, such as SSOs. Common errors often include simply not following the manufacturers' recommendations with, for example, not having bypass capacitors on all sides of a quad flat pack.

7.5 References, Notes, and Related Documents
   a. Designers Must Take Care When Powering High-Speed CMOS^{XL}, Robert M. Hanrahan, ED Online ID #5415, Electronic Design, August 4, 2003.
   b. "RT54SX32S High I_{CCI} Inrush Current," OLD News #10, May 16, 2003.
   c. "Analysis of Printed Circuit Board Artwork: Bypassing," Rod Barto, Office of Logic Design, March 2004.
   d. "PCB Layout Issues," presented at "Design Seminar on Actel SX-A and RTSX-S Programmed Antifuses," Tuesday, April 13, 2004, NASA Goddard Space Flight Center. Discusses layout issues for bypass capacitors, vias, and power and ground planes, in the context of "before and after" of a flight printed circuit board.

8. EEPROM, Flash, FRAM, and other Non-volatile memories including embedded memories

8.1 Definitions: While normally associated with computers, many of the concepts in this section also apply to the "configuration memory" of FPGAs.

a. **Fixed**
  1) The contents of the memory are physically fixed by the structure of the memory element.
  2) Examples: core rope memories (wire wound through or around a core), fusible link PROMs, and antifuse-based PROMs.
b. **Erasable**
  1) The contents of the memory are non-volatile, like the fixed memories, but the contents can be changed. In many cases this involves an erase operation and then a write.
  2) Examples: core, plated wire, electrically erasable programmable read only memories (EEPROM), erasable read only memories (EPROM), ferroelectric memories, and flash. The "ROM" in EPROM and EEPROM is a poor part of the name as it implies permanence, which is incorrect. Devices such as EEPROM may need "refreshing" over long missions as many are rated with a 10 year storage lifetime, giving them volatile characteristics.
c. **Volatile**
  1) The contents of the memory are volatile; they do not retain contents either after the cycling of power or during "brown out" conditions. This class is subdivided into two subclasses, static, which will retain state indefinitely and dynamic, where the memory must be read and subsequently refreshed.
  2) Examples include SRAM, DRAM, and SDRAM.

8.2 Protection During Power-Up/Down Transitions: This has been noted as a common problem for erasable non-volatile memories. The analysis and test must carefully examine all of the signals for proper and safe operation during power-up, power-down, and brown out transients. Note that the real power supply and its bounded characteristics must be used, not laboratory supplies which most likely will have substantially different characteristics. Some devices have a reset pin to help protect against inadvertent writes. The design, analysis, and test/evaluation of this circuit under all conditions is critical for maintaining the integrity of the non-volatile memories contents. Consider circuit operation if the power is shut down during a write cycle, either planned or unexpected and the design should ensure the proper completion of write cycles to ensure that the contents of the non-volatile memory is protected. The write cycle often includes not only the time for the bus operation to complete, but for the time for writing internal to the part, which can take on order of 10 ms. Another related consideration is the unexpected application of a system reset signal. Shutdown states should be entered to help ensure that write cycles are fully completed and properly shut down, with the critical signals safed.

8.3 Analysis of Damage During Write Cycles: The technology of the non-volatile memory must be carefully considered if the memory is to be written in flight. Some of these devices, such as EEPROMs, use high voltage to write the cell. If struck by a heavy ion with high voltage applied, the result can be a hard fault. Thus, writing should be done with caution and the technology used for storage chosen wisely.

8.4 Cycle Count (i.e., # of write cycles for EEPROM, all cycles for FRAM, etc.): Many non-volatile erasable memories have limited number of cycles. There is no hard and fast rule with the numbers of cycles ranging from 104 to 105 or higher. Each device must be treated on a case-by-case basis with system lifetime and radiation factored in. There are some subtle specifications that will be noted here, as examples. The popular 128k x 8 Hitachi die, for example, has a lifetime write specification limit of 103 cycles in byte mode with 104 cycles in page mode. The write mechanism for this device utilizes an 8-byte subpage as the smallest unit that can be written. Hence, writing the same memory space one byte at a time is more stressful than page writes since entire subpages must first be fetched and then re-written. Another subtlety is the operation of some FERAM (ferroelectric RAMs). In these devices, read cycles operate in destructive readout mode (DRO) and an internal write cycle is executed after every read. Hence, the number of read cycles must also be managed in addition to write cycles, since each read access generates a subsequent write cycle.

8.5 Transients and Noise: It is critical that the signals interfacing with non-volatile memories be clean and system noise kept to a minimum and always meet all specifications. In this case, signals includes not only logic signals but power and ground connections; robust bypassing should be used. Noise glitches on EEPROMs, for example, can cause false write cycles to be generated, resulting in advertent altering of the device's contents. Illegal timing to a non-volatile memory, even with the write signal not asserted, can result in the corruption of the memories contents.

8.6 Reliability, Refreshing, and Reloading

The required reliability of the non-volatile, erasable memory device is highly dependent on its application. If the device operates as part of a large memory array, then some bit failures and even page failures can be tolerated either by error correction techniques or by error detection and mapping the failed segment out of service. Applications such as boot ROM for a central processing unit or memory contents for an FPGA, require perfect system performance. For single bit failures a Hamming code may suffice, although that may be awkward for serial PROMs. Note that some failure modes of non-volatile memory devices may result in a bit oscillating or not providing a valid logic level; in this case, an EDAC device may or may not correct the single bit error, depending on the logic design of the EDAC device being used and whether or not it is static hazard free. In any event, the devices employed, combined with the architecture of the particular system, must ensure that there are no lockup states from any credible failures. Credible failures include any single bit error and an inadvertent corrupt of a non-permanent memory's contents.

Other forms of redundancy may be required such as TMR with switchable spares. Some options include the ability to switch in alternate devices, the use of permanent memory such as PROM, or the use of storage buffers to replace erasable non-volatile memory functions, using operational overhead to manage the risk. For example, if a configuration memory device for an FPGA fails, a storage buffer and CPU may configure the FPGA using a different loading mode, assuming that, of course, the FPGA isn't needed to run the computer. In general, for critical applications, permanent memories such as PROM are to be used to ensure that the spacecraft or other system can not be permanently lost. This can take the form of boot and safe-hold code for a processor or a basic operating configuration for an FPGA.

Another consideration is the guaranteed storage time of the device vs. mission length. There is no hard and fast rule and each device must be analyzed on a case by case basis. Ten years is a frequent specification for the retention of memory contents. However, system lifetimes of several decades is not uncommon. Refreshing can be risky and the usefulness of it should be verified with the manufacturer's assistance, to ensure a guarantee of storage integrity, particularly in the radiation environment. Obviously, when the device is refreshed, it may be susceptible to damage in the space environment by heavy ions, as noted above. Other errors can occur, damaging the contents, such as a computer crash, brown out, or the unexpected removal of power due to a bus fault or a spacecraft entering a safe mode. Also, each write cycles takes away from the operational lifetime of the component.

8.7 Recommendations and Tips

a. Many designers use a simple RC timing circuit for the generation of a POR or "Power On Reset" signal. Looking closely at the acronym, is has the word "on" in it and the "O" does not stand for "Off." Use of such a circuit will often protect memories for power up but assertion of the protection circuit will lag either during a brown out or when power is removed.

b. POR circuits are often best generated in the power supply module.

c. Ensure that critical memory controls behave properly during power transient conditions. They are often incorrectly implemented by an FPGA that is not guaranteed to be under control during the power-on, power-off, and periods when power is disrupted. FPGA and configuration memory device internal power-on reset circuits may be active along with initialization sequences, charge pumps have to supply sufficient charge and voltage to turn on high-voltage isolation FETs, etc.

d. Erasable memory device protection is an analog function and digital components must be used with extreme care. Along with timing, many memory devices require non-standard voltage levels and currents for protection.

e. Consider the likelihood of a software fault is 100%.

f. Device Protection: Many erasable devices implement "software write protection" to prevent against inadvertent writes to the memory. JEDEC has published a standard on this type of protection. Do not keep the "keys" to unlock the memory on-board unless absolutely necessary.

g. Subsystem Protection: System level write protection limits should be implemented in hardware, to protect against software faults. Some systems implement this in software which is risky; see bullet #5 above. Use external hardware discrete command as an additional barrier to prevent inadvertent writes.

h. Analyze and test devices for lockup states. These can occur in many memory types from illegal loads into command registers, poor signal integrity, poor power quality, or an SEU. Some device lockup states require power cycling to clear. Lockup states in memory devices are often not considered either in memory controller designs (soft repairs) or system designs (power cycle required for clearing of faults).

i. Critical switching between memory images for booting implemented as a software function can not be guaranteed to function under all credible faults resulting in system lockup. Use a discrete hardware signal to implement recovery from faults to prevent system lockups.

j. Consider the likelihood of an EEPROM or flash device fault to be 100%. There are enough failures in the industry to justify such an approach.

k. Boot and Safe-Hold Code: High-reliability, radiation-hardened, fixed memories should normally be employed for boot and safe-hold functions. For applications such as instruments, DMA functions, properly implemented, can load memories with boot code. In this case, the instrument should be safed by hardware logic. DMA functions should not require any operational software. A hardware discrete command to clamp a processor into reset is also recommended.

l. "Refreshing" of critical code, such as boot code, that is stored in erasable memory should not be done to mitigate faulty devices. Instead, use reliable fixed memory technology.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 19 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

m. Verify Margins of All Protection Signals: DC voltage margin; AC voltage margins (e.g., cross talk); Timing (protection signals for power up, power down, and during glitches). The power down rate of voltage buses is often ignored or idealized.

n. Third party device packaging houses: Verify that they fully understand the technology and the original manufacturer's test procedures and screening criteria. Compare failure rates of third party houses with those reported by the original die manufacturer. Ensure that proper and complete testing for space missions is performed.

o. Multiple copies of the same code in the same technology is risky, if the fundamental technology is not reliable. With the current rash of industry failures of EEPROM, for example, multiple copies of the same device type, even with hardware selection, is a form of Russian Roulette. Storing redundant copies of code in separate blocks of one device can be subject to common mode failures.

p. Treating bit, block, and device failures in software can be done in many instances, such as recorders. For critical boot code, as an example, treating failures as a software maintenance issue that must be done before a reset, should not be a function relegated to software. That would be a form of "foam logic."

8.8 References, Notes, and Related Documents

a. "Summary of Recent EEPROM Failures," OLD News #12, July 3, 2003.

b. "Maxwell EEPROM Bit and Page Failure Investigation Report," Y. Chen, June 3, 2003. e-mail for access

c. "EEPROM Bit and Page Failure Investigation," Yuan Chen, Rich Kemski, Duc Nguyen, Frank Stott, Ken Erickson, Leif Scheick, Richard Bennett, and Tien Nguyen, 2003 MAPLD International Conference, Washington, D.C., September 9-11, 2003.

d. Reliability Report: HN58C1001 Series CMOS 1M EEPROM

e. EEPROM Evaluation and Reliability Analysis, Aerospace Report No. TOR-2000(3000)-01 June 28, 2000. e-mail for access.

f. "Usage of EEPROM in Digital Designs," Saab Ericsson Space, D-G-NOT-00385-SE, 2004

g. "Design of Memory Systems for Spaceborne Computers," 2004 MAPLD International Conference, Washington D.C., September 8-10, 2004.

h. "An Application Engineer's View," 2004 MAPLD International Conference, Washington D.C., September 8-10, 2004.

i. "Observations in Characterizing a Commercial MNOS EEPROM for Space," 2004 MAPLD International Conference, Washington D.C., September 8-10, 2004.

j. "Maintaining Data Integrity in EEPROMs," 2004 MAPLD International Conference, Washington D.C., September 8-10, 2004.

9. Timing Analysis

Timing analysis of digital systems can be summarized quite simply: ensure that **every** parameter on the data sheet is met for all elements of the design. In practice it can be a significant effort and care must be taken to ensure that the calculations are performed correctly. A circuit properly designed and analyzed will work properly for all combinations of components over the entire specified operating environment. **Every** time.

It is tempting to simply use Computer Aided Engineering (CAE) software tools and "push the button." This does not work in the general case. Typically, if the design style and circuits fit the model that the CAE tool vendor desires, than a lot of analysis can be done accurately and rapidly. However, this doesn't work for all analyses and not all legitimate circuits fit the tool vendor's model, as we often have stringent requirements in power, area, functionality, etc. CAE tools are not the answer.

9.1 Basics of Timing Analysis

The basis of all timing analysis is the clock and the flip-flop.

For the clock, which is covered in detail in a previous section, it must be well understood parametrically and glitch-free. Thus, the timing analysis must ensure that any clocks that are generated by the logic are clean, are of bounded period and duty cycle, and of a known phase relationship to other clock signals of interest.   Using voting circuits, for example, to generate clocks may result in problems if the voter is not hazard-free. Logic synthesizers are capable of and have generated logic hazards in clock generation circuits.

The flip-flop (or latch) is the basis of this section. Quite simply, one must prove that all of the flip-flops parameters are always met. The only exception to this is when synchronizers are used to synchronize asynchronous signals, the topic of another section of these guidelines.

The clock must, for both high and low phases, meet the minimum pulse width requirements. Certain circuits, such as PLLs, may have other requirements such as maximum jitter. As the clock speeds increase, jitter becomes an increasingly important parameter. For clocks that are close to the device's specifications, note how the high and low time are measured and the characteristics of the clock, as the threshold voltage may differ between the clock specification and the input device's.   Also, the transition time of the clock signal, effected by loading and the environmental factors, can degrade the available pulse width. Failure to maintain a proper pulse width can result in the flip-flop going "metastable."

For asynchronous presets and clears, there are two basic parameters that must be met. Obviously, there is a pulse width requirement that must be guaranteed. However, removing the preset or clear from a device asynchronously to the clock may result in metastable states in the sequential circuit. This parameter is frequently called the removal time and is denoted as t_{REM}. Unfortunately, many data sheets do not specify the removal time. That does not mean that it is not a requirement.

For data (or J, K, T, EN, synchronous clear, etc.) inputs, show that all setup and hold times are met for the earliest/latest arrival times for the clock.   Setup times are generally calculated by designers and suitable margins can be demonstrated under test. Hold times, however, are frequently not calculated by designers and CAE tools sometimes calculate this incorrectly, use inaccurate databases, or some combination of the two. One of the leading causes of digital logic malfunction is hold time violations. Check the specification for the device carefully, for FPGAs, to see if the manufacturer will guarantee that hold times will always be met when using the global clocks. This is not always the case.

When "passing" data from one clock edge to the other, ensure that the worst-case duty cycle is used for the calculation. A frequent source of error is the analyst assuming that every clock will have a 50% duty cycle.

When passing data from one clock domain to another, ensure that there is either known phase relationships which will guarantee meeting setup and hold times or that the circuits are properly synchronized.

If you are relying on measured values to "screen" the parts for meeting the worst-case analysis, ensure that the parts' testing is done for both the best and the worst case access times. For example:
   a. Differences of 4:1 have been found for ROM/PROM access times when addressing different parts of the array.
   b. Some memories may be slower at higher supply voltages, as the threshold for the sense amplifier may rise, more than offsetting the increased speed from the higher supply voltage.

   c. The access times from some memory devices may be a function of the sequence in which data is read. For example, the preceding read, depending on location, may leave lines and sense amplifiers in particular states. So, reading a '0' preceded by a '0' may give a different result than reading a '0' preceded by a '1'.

9.2 Environmental Effects

For robust circuits, designs must be tolerant of various environmental effects. These include:
   a. Temperature
   b. Voltage
   c. Life time
   d. Radiation
   e. Process, Speed Grade, and Programming

In general, analysts will do an extreme value analysis (EVA) based on the widest possible corners of each environmental factor, simultaneously. This will result in a system with very wide margins and tolerance of unforeseen, off-nominal conditions. However, this process will also in many cases needlessly limit performance, increase resource consumption, or force more complex architectures and analysis. For example, for two flip-flops located on the same die just a few microns apart, one flip-flop will not be at -55 ºC while its neighbor is at +125 ºC. In this case, it would be reasonable to "sharpen the pencil." Assuming 100% tracking is not valid either for this parameter; for others, no tracking can be assumed.   Often the designer/analyst will be limited by the data and/or models available and will not be able to determine how much tracking will occur. In this case, the least amount of tracking will have to be assumed, a conservative approach.

The temperatures and voltages used will be a function of each particular mission and the location of the electronics. Ensure that worst-case values are used plus margin, as specified in the project's reliability plan, and not the more optimistic expected values. There have been many missions where the actual values were outside the bounds of the expected values.

Components do age and their characteristics change. However, one can not assume that all propagation delays, as an example, will track and that the relative delays will remain unchanged. For examples, for certain FPGAs, several studies of life test data showed that not only will the delays not track, but that they may not even have the same sign, with devices sampled from a single manufacturing lot. Hence, one can not demonstrate hold time margin by test. In general, most programs will specify ±10% for propagation delay change over the mission lifetime.

The approach for radiation is similar to life, above. One can not assume perfect tracking. Again, ±10% for propagation delay change over the mission is generally used.

Typical timing analysis programs will allow one to select a setting for process, typically best, typical and worst. To effectively use these settings, note that this is not a predictor of circuit speed but a bound for circuit speed. Many engineers and analysts assume that this will predict speed or prove that two circuits can not lose a race. This is not the case. For example, no two transistors will be processed identically, although often it will be fairly close. There are lot to lot variations, wafer to wafer variations within a lot, die to die variations on a wafer, and transistor to transistor variation on a die. Hence, one must treat these values as bounds and not as actual values. There will be a certain degree of tracking. How much you can use in an analysis depends on the data available and algorithms available in the CAE tools.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 22 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

The speed grade setting can often be misleading for timing analyses. For the worst or slowest case, the speed grade, as stamped on the part, is the correct setting to use. For the best or fastest case, using the grade on the case can give you an incorrect answer. For example, some FPGAs are binned by measuring the speed of a special test circuit and ensuring that the speed is less than some threshold value. This is a one-sided relation and parts that would pass a faster speed grade will be binned and stamped with the lower one; otherwise the parts can not be shipped, since the part number would be wrong. So, for the best or fastest case, use the fastest speed grade the tool supplies, unless you have specific knowledge. This may come from a specification or from read and record data from the vendor.

For antifuse based FPGAs, the amount of "tracking" that can be assumed in an analysis will be less than is often found in other device types. While the transistors on a die will track to a certain degree, as they are fabricated together, the distribution of programmed antifuse resistance will resemble a random variable.

Taken together, this means that if you wish to guarantee that signal A always arrives before signal B by T nanoseconds, running a dynamic simulation with all values set to the worst-case will give an incorrect answer as there is no guarantee that all path will be the worst. In reality, they will not. That is why min-max or extreme value analysis is required for accurate timing analysis.

9.3 References, Notes, and Related Documents
   a. Digital Timing Analysis Tools and Techniques
   b. Root-Sum-Square (RSS) Calculations of Digital Timing Delays
   c. NSCAT Digital Subsystem Design Documentation and Analyses
   d. Galileo AACSE: Worst Case Analyses (WCA) Description and Criteria
   e. "Propagation Delay and Aging," *OLD News* #4, August 3, 2002.
   f. "Minimum Delays and Clock Skew in SX-A and SX-S FPGAs," *OLD News* #13, July 15, 2003.
   g. "Logic Design: Clocking, Timing Analysis, Finite State Machines, and Verification," Presented at the 2002 MAPLD International Conference, Laurel, MD, September 9, 2002.
   h. Timing Analysis of Asynchronous Signals
   i. Discussion of MetastableStates.
   j. Signal integrity of the clocks is important, not only for ensuring that the propagation delays are calculated correctly, but that the devices function properly. Often the clock inputs must meet more stringent requirements than typical signals, with fast transition times specified as well as lower values for V_{IL} and higher values for V_{IH}.
   k. "RT54SX72S: Propagation Delay vs. Life," June 6, 2004.

10. Miscellaneous Design Guidelines and Criteria

10.1 Noise Immunity and Quiet Designs: Take steps to ensure adequate and robust noise immunity.

   a. Key steps are listed with respect to simultaneous switching outputs and signal terminations.
   b. Choose differential signals, particularly for connections between cards. Newer logic devices are directly supporting differential standards. Additionally, high-speed, lower power differential devices support standards such as LVDS are now qualified.
   c. SERDES components/cores can cut down the number of lines, reducing noise, and hence, increase the noise immunity of the system.
   d. Use hysteresis inputs when available to reject noise.

DIRECTIVE NO. 500-PG-8700.2.7-	Page 23 of 34
EFFECTIVE DATE: 08/12/2005	
EXPIRATION DATE: 08/12/2010	

e. Avoid having flip-flops or logic devices with internal memory driving cables or massive capacitive loads.

f. Inputs that are "TTL compatible" often have specifications and real thresholds that are not TTL compatible, particularly for V_{IH}.

g. Outputs, particularly from some CMOS families, may not be able to drive TTL loads to a valid logic '1' with sufficient noise immunity. Calculate worst-case currents and voltage output vs. worst case input thresholds.

h. DC margins for TTL interfaces should be no less then 400 mV, with at least 500 mV recommended.

i. For TTL outputs driving CMOS logic thresholds, a pull-up resistor can give adequate DC noise margin, after allowing sufficient time for the voltage to rise. However, when used as a clock input, multiple triggers are reasonably likely to occur as the waveform will have a "hump" in it. This should be avoided and is a poor interface.

10.2 Defensive Design and Designing for Off-Nominal Events: Consider credible but unplanned events. Often many of these situations can be economically handled with a bit of thought. Here are a few sample issues to consider.

a. **Perform limit and validity checking.** The system should respond in a reasonable fashion to unreasonable inputs. For data passed from one source to another, simple bounds checks can detect and cause appropriate action for many off-nominal conditions, such as a disconnected source, perhaps resulting in all F's being returned on a data bus. For floating point numbers, is the input in a valid format? A minimum criteria is that any credible input should not damage hardware and prevent recovery. Assume that the probability of software failure is 100%.

b. **Provide fail-safe interfaces.** Analyze the performance and safety of the circuits if a wire breaks in a connector, for each wire. For power, use multiple wires such that if any one wire breaks the remaining set can carry the load (and be sure to test this redundancy). For signals, consider on-board terminations that will pull floating signals into a safe and operational state. This can also provide protection if the board or subsystem is powered with a connector not hooked up, perhaps by test error. Avoid putting signals such as power and ground on adjacent pins, as a short can take out the system (remember SEASAT).

c. **Lockup states**: Ensure that all devices you design do not have lockup states in the finite state machines. Choose commercial or commercial devices wisely and operate them defensively. For example, SDRAMs have lockup states that may require power cycling to clear and may cause damage. Noise, single event upsets, or even invalid commands can cause this condition. Refresh command words often. Many microprocessors, non-volatile memories such as EEPROMs, etc., can have various lockup states. Some devices may be cleared by a reset; others often require the cycling of power to clear.

d. **Power Glitches**: Power glitches may occur for a number of reasons, electrical discharges, switching loads on or off, faults in loads, firing pyrotechnic initiators, lightning strikes, relay switching, etc. These transients may result in the maloperation of circuits and effectively multiple bit upsets in finite state machines and lockup of commercially derived components. Careful attention needs to be paid to circuits that have permanent state such non-volatile memories. Can a write cycle be interrupted resulting in a system being put into an inconsistent state such as during a memory upload? Can false writes be generated corrupting the system state? Can one-time events be falsely initiated? Are driving circuits for latching relays holding system state information adequately protected from false triggers? Will a glitch on one supply result in a power sequencing violation for a component with multiple supplies or a set of components operating off of different supplies?

e. **Shorting Test Points**: Since test points will be accessed by humans, assume that they can be abused. That is, a short to ground from the ground ring on an oscilloscope probe is an example of a credible and expected fault. Use isolation resistors to ensure that a failure in the test interface does not damage flight hardware.

10.3 Various Tips, Considerations, and Criteria

a. **ESD Ratings:** Many of the modern high speed components have low ESD thresholds. Some components designed and qualified for spacecraft interfaces have been seen to have 300V limits. Often these values are not listed on the data sheets and qualification test reports must be obtained.

b. **Spares, Pins, Gates**: Leave spare space and pads on the printed circuit boards. Pre-wire footprints for power, ground, and bypass capacitors to fit SSI/MSI components that may be needed to fix an "oops" or to meet a changing requirement. For programmable devices, ensure that adequate gates, flip-flops, and pins are available. While this sounds obvious, designs with no spares have been presented as early as PDRs, leaving zero room for change. For programmable devices with spare pins, program in a variety of simple functions such as inverters, AND gates, flip-flops, etc., with the input pins terminated through resistors. For small fixes this will provide available logic on the board and potentially eliminate a re-spin of an ASIC or FPGA and a rework cycle on the board.

c. **Holes In the Board**: Place holes strategically around the printed circuit board. For late fixes on double sided boards, this can simply the modifications.

d. **Test Points:** While it is common and useful to place test points on boards, also strategically place points to hook up the oscilloscope ground lead. With the move to surface mount capacitors, good termination points are often hard to find, resulting in long ground leads, poor connections, and inaccurate waveforms for fast moving signals.

e. **Grounding of Lids**: Verify that lids are grounded for operation in a charged environment. Indeed, a charged environment can include test where moving air is heated and cooled and then blown into an environmental chamber. Some devices' lids are not grounded, even on parts sold into the space market. In some instances, lids that are grounded have been made floating by the manufacturer prior to shipment. A drain wire should be used to ensure that no buildup of charge is possible, preventing ESD damage.

10.4 References, Notes, and Related Documents

a. Note the drain wire attached to the lid of the FPGA (lower left hand corner) of the MOLA-2 PC-2 Electronics which is orbiting Mars on Mars Global Surveyor. Conductive epoxy was used.

b. The gold trace coming from only one corner of the lid identifies Pin 1 on this Virtex FPGA.. The manufacturer cut through the trace prior to the delivery leaving the lid floating, based on some customers' inputs. We requested the opposite to ensure that there will be no buildup of charges and thus prevent ESD events. Here is the cut, in a magnified view.

c. *OLD News* #11 Interface Components and ESD, May 28, 2003. ESD and proper device handling practices are nothing new and normally would not warrant an OLD News posting. Indeed, ESD practice and component tolerance have improved so much over the years that ESD damage hasn't been a major source of problems for quite a while, for regular digital integrated circuits and interface components. However, there have been some recent surprises. ...

d. The specifications for inputs must be carefully read as not all device or MCM inputs are truly TTL compatible.

e. "Case Study: Simultaneous Switching Outputs," presented at "Design Seminar on Actel SX-A and RTSX-S Programmed Antifuses," Tuesday, April 13, 2004, NASA Goddard Space Flight Center. Presents 4 cases of "staggering" I/O switching, trading off lower di/dt for increased data transfer time and analyzes software performance and the effect of module placement.

11. Design and Analysis Documentation

11.1 Label Schematics

It is good practice to label every instance, symbol, and net in all schematics. This is useful for two reasons:
   a.   The backend engineering and analysis tools will provide output that will reference the labels, either the ones that you supply or the default ones provided by the tools. Having to find I$25/I$2/N$32 where the label is a hidden attribute is difficult and makes engineering analysis of the tool's analysis difficult.
   b.   When you are troubleshooting or performing internal probing, having the labels ready to go lets the engineer concentrate on what he or she is doing.

The following labeling scheme works well:
   a.   Give every net a name. Preferably it will make some sense but any visible name is better than a name that has a invisible display attribute attached to it. Also, the net name should reflect the sense of the signal. There are various conventions for this, Viewlogic has a '~' prefix in the database and uses an over score on the schematics. Many engineers use a '_N' suffix. It is recommended not to use an '*' as some programs or systems may reference that as a wildcard. Additionally, signal or net names should be in all capital letters to promote portability between tools.
   b.   Give every instance a name. That includes I/O symbols, modules for hierarchical designs, flip-flops, gates, etc. The following works well and will help you find the components quickly. It also minimizes the overhead in maintaining the drawings but keeping the instance numbers local to a single sheet (which also makes some CAE tools checks perform a useful function). The general form of a reference is XYn.

      1.   X is one of:
             a)   M - module to show another level of hierarchy and is affixed to the symbol
             b)   F - flip-flop
             c)   G - gate
      2.   Y is from the ordered set {A, B, C, ...} where the letter corresponds to the sheet number in that module.
      3.   n is the reference number on a single sheet. Each sheet starts from 1 and is unique from all other sheets, saving time when looking to add a new component.

Schematic with Labels
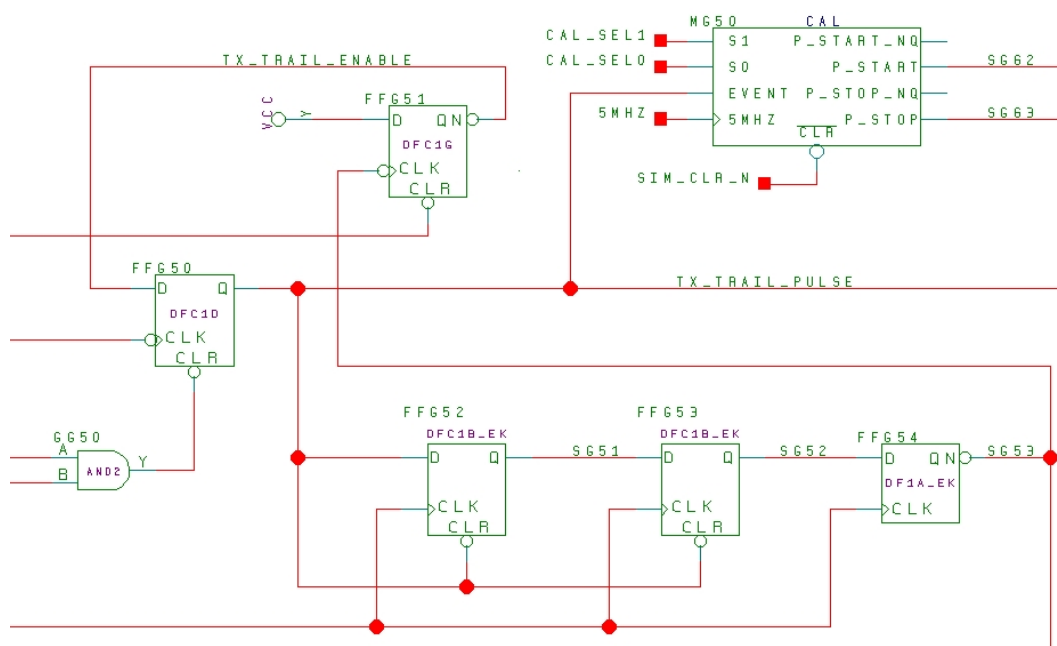
Schematic without Labels
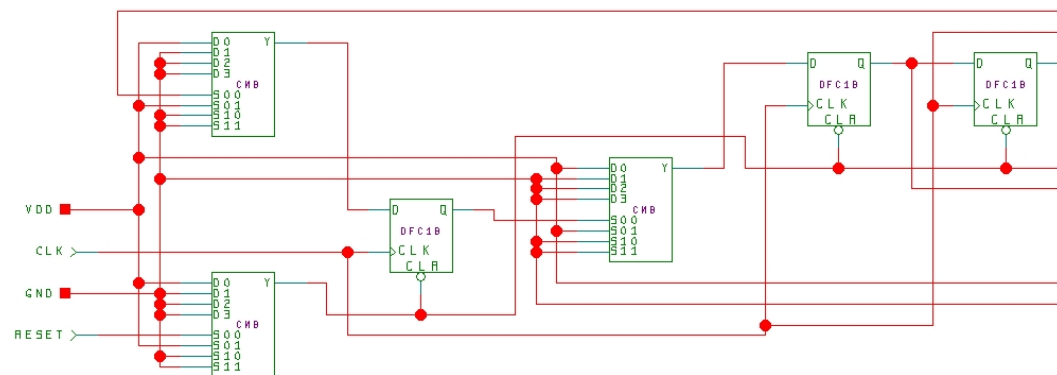
Signals that go off-page should be marked with references to the sheets that they either come from or go to. It is difficult and error-prone to go through a 20 or more page schematic and search each page for where off-page signals have gone.

Auxiliary Synthesis Files are considered primary design documentation. In particular any files that control or guide the synthesis process, or that document the output from the synthesizer, should be controlled and included with any documentation package. For instance, an auxiliary file may determine the style of a finite state machine or perhaps whether "safe" encoding is used. The HDL description does not specify the design in this case. Many designers like to keep directives out of the HDL source file to keep it free of synthesizer-specific directives to make the design more portable. It is preferred to put the directives in the HDL code to minimize the chances of errors in the design, synthesis, analysis, or modification processes. Similarly, synthesis output files which document what the synthesizer has actually done is also primary design documentation. For even something as trivial as the flip-flop, the HDL source is insufficient as the circuit design is not knowable from the ASCII text input. The synthesizer output file is needed to determine whether or not the flip-flop has been replicated, which is critical in many high-reliability applications. Similarly, for a second example, the state encodings will be listed in the output documentation; it is noted that in some cases the synthesizer decides that it knows better then the designer and does not always follows the guidance provided it.

11.2 References, Notes, and Related Documents

a. "Suggestions for VHDL Design Presentation": Detailed design review and worst case analysis are the best tools for ensuring mission success. The goal here is not to make more work for the designer, but to: Enhance efficiency of reviews; Make proof of design more clear; Make design more transferable; and Improve design quality.

12. Review of Digital Electronic Circuits

12.1 Introduction

Reviewing an FPGA is quite similar to the design/analysis process, minus the synthesis process, as it employs identical skills and techniques. In this usage, synthesis refers to the process of synthesizing a logic design to perform the required functions reliability and not to the process of simply pushing a button on a computer screen and having software replace the human.

As such, the review of a digital electronic circuit is simply no more or no less than proving that the design will reliably meet all requirements and specifications. That of course is the job of the designer/analyst and the reviewer's function is redundant.

This section gives some insight into the process by explaining the steps to be taken in reviewing an FPGA-type digital design. Although it is written with a currently popular target FPGA (RT54SX-S series) and synthesis tool (Synplicity), the overall methodology and most of the individual steps will apply to any FPGA. It is assumed that the reader is tasked to review a design having no prior knowledge of its function.

12.2 Be Sure You Have the Correct Specification for the FPGA You Are Reviewing

Device specifications can be updated at any time, and there may be subtle differences between a manufacturer's part types, so be sure you have the correct specification sheet for the part being reviewed. Check the manufacturer's web site for the latest specification, as with web-based specification distribution, updates can come at any time and often without notice. The governing military specifications are also now available on-line. Of course, similar care must be utilized for all devices in the system being reviewed.

12.3 Collect the Necessary Review Files

Most FPGA designs are done in an HDL (hardware description language) such as VHDL or, less often, in Verilog, and so that is the assumption here. The use of standard tools from a well-known manufacturer and, preferably, from the FPGA vendor, is encouraged. This enables the review process by not forcing the reviewer to have to obtain and learn new tools, which drives up the cost and time for a review as well as making it less effective. For this discussion, we will assume that the Synplicity synthesis tool is being used; the principles are similar for other manufacturers.

The files required for review include those that describe the system, the FPGA, and the electronics surrounding the FPGA:

  a.   A system description: PDR/CDR package, system specification, etc.
  b.   A set of board schematics.
  c.   The FPGA HDL files along with other files that guide the synthesis process.
  d.   Existing test benches
  e.   Results of synthesis and timing analysis runs
  f.   The .srr file: Synplicity synthesis log file
  g.   The .adb file: the database file after place and route -- this is the design.

Before starting the review, familiarize yourself with the system operation and requirements and look over the board schematics to get a feel for the design. Make note of the FPGAs place in the overall system and its criticality.

  h.   Is the correct operation of the device safety critical?
  i.   Does the device control pyrotechnic initiation circuits, thrusters, or high-voltage power supplies?
  j.   Does the device issue spacecraft critical or mission critical commands, set latching relays, or otherwise perform configuration functions?
  k.   How many different power sources feed the board, and how are they sequenced? Consider both power-up and power-down sequences.
  l.   How is the circuitry reset?
  m. Is it critical that the FPGAs functions be performed correctly the first time tried, or is there opportunity for retries?
  n.   Does the FPGA receive asynchronous data or commands or perform processing on asynchronous events?
  o.   How many clock sources are there, and what are their frequencies, duty cycles and phase relationships? A clock tree should be provided by the designer or it can be generated as part of the review process.
  p.   The printed circuit board artwork should be readily available, as needed, to support signal and power integrity analysis.

Reading through the HDL at this point usually isn't fruitful, as most HDL is poorly written and documented; not much can be gleaned from it at this time.

12.4 Performing the Review

There are several levels of detail to which a review can be performed, and ideally every design receives the most detailed review in which correct FPGA usage and the overall electronic and logic design are considered and proven correct. This isn't always possible because of time and budget limitations, but the steps in reviewing a design are the same regardless of the ultimate review level, the difference being how many steps in the review are accomplished. Often, a "spot check" or "scan" of a design is all that may be performed, because of the aforementioned limitations.

One critical thing to remember is that the HDL is not the design, but simply the designer's description of the desired logic. Running HDL simulations and test benches is insufficient proof of a design's correctness.

The design is the output of the back end place and route function and the hardware is the physical chip, after configuration.

The fidelity of the actual design to the intended design depends on the quality of the synthesizer, which is unknowable, and the ability of the designer to

a. Write synthesizable HDL
b. Understand the synthesis process and tool employed
c. Control the synthesis process
d. Verify that the synthesis process produced what was intended
e. Correctly guide the back-end place and route tools. These tools may also alter the intended design through logic replication, combining, elimination of logic functions, setting I/O module parameters such as I/O thresholds, output slew rates, the presence or absence of clamping diodes, cold-spare functionality, etc. While not as abstract and complex as logic synthesizers, failure to understand the processes in the back-end design process has been seen to cause design errors.

One of the limitations of FPGA design is that the static timing analysis tool is primarily designed to analyze fully synchronous logic that uses only one clock edge. Dynamic logic simulators are insufficient for proving design correctness. Asynchronous design techniques are extremely difficult to analyze with the available tools, are error-prone, and are thus discouraged where these techniques are not required. Therefore, an important part of the review process is ferreting out design techniques that are error-prone and should not be used in an FPGA.

12.4.1 Reviewing the Board Schematics

The FPGA application can not be properly reviewed without knowing it's electrical environment. The following list, which is not exhaustive, shows several classes of issues that must be examined.

a. Of primary importance are that the special pins, e.g., TRST*, are treated properly. Review the FPGA specification for the requirements of unused clock pins and other special pins such as device configuration or programming pins and verify they have been properly terminated on the board.
b. Look for unusual loads (e.g., high capacitance or non-logic loads).
c. Look for unusual sources (e.g., questionable logic levels, excessive transition times, mixing of logic families, devices powered by different supplies, etc.).
d. Note circuitry that may be powered up or down independently of the FPGA and the cold-sparing capability of each device.
e. Determine the number of simultaneously switching outputs and their distribution around the FPGAs I/O ring.
f. Determine the length of PCB traces and how the signals are terminated, ensuring that overshoot and undershoot specifications are met. In particular, carefully examine all signals that leave the PCB.
g. Ensure that the manufacturer's recommendation for bypass capacitors and power/ground planes are being followed. Past reviews have found boards with inadequate capacitance and routing, including one case where zero bypass capacitors were used and another where placement of the capacitors led to poor performance.

12.4.2 Reading the Synthesizer Output Log File

The .srr file, the output log file written by Synplicity as it reads though and processes the HDL files, can tell the reviewer quite a bit about the design. Experience shows that designers rarely read .srr files after synthesis.

    a.   The first part of the .srr file shows two passes made through the VHDL. In the first, Synplicity finds the VHDL modules and state machines, and in the second the state machines are revisited and reset logic is created for any which the designer gave the "safe" attribute to deal with illegal states.

        Note the state machine names found in the first pass, then note in the second pass any for which reset logic is not created. All state machines should have their illegal states handled, because otherwise illegal states may cause inappropriate behavior. The requirements for each state machine must be determined and either handled in the logic, by periodic local resets, or by a POR or other reset command. It is often found that designers concentrate on the correct functioning of the circuits and not on either the effects of "glitches" or recovering from them. Glitches may result from, for example, power transients, radiation, or ESD.

        Having the reset logic created, however, does not mean the FPGA will perform the correct functions if illegal states are entered. One subtlety of Synplicity's synthesizer-generated reset logic is that under some conditions a half-edge flip-flop (e.g., a falling edge flip-flop in a rising edge design) is used to generate the reset. The designer generally doesn't recognize this because Synplicity doesn't point it out, and its timing isn't analyzed. This timing analysis relies on the duty cycle of the state machine's clock, which may vary considerably, and not the period, which is generally quite accurate, as crystal controlled clock oscillators are the norm.

    b.   The next section shows the part replications. Most important among these are flip-flop replications, and the most important of these are replications of flip-flops which are part of synchronizers of asynchronous signals; replicated flip-flops in synchronizers are of course not permitted. In general, replicated flip-flops could cause inappropriate operation if transient events cause logically equivalent flip-flops to take on different values, and should be discouraged. If replicated flip-flops are employed in the design, then the acceptability of each instance must be thoroughly analyzed and documented. This task is both labor intensive and error-prone and must be performed after each synthesizer run.

    c.   The next section gives a list of the logic types used. Compare the flip-flops used with the FPGA manufacturer's macro library to see if any of the following types are used:

        1)   Flip-flops without sets or clears, indicating circuitry that will not be reset on POR or reset command;

        2)   Flip-flops with both sets and clears, indicating possible asynchronous design techniques (the absence of set/clear flip-flops does not indicate the absence of asynchronous design techniques);

        3)   Latches, for which the timing must be checked by hand;

        4)   Opposite edge flip-flops (e.g., falling edge flip-flops in a predominantly rising edge design) that could place constraints on clock symmetry and be more difficult to analyze with the timing verifier. Some opposite edge flip-flops could result from the use of the "safe" attribute, noted above, and the designer is often unaware of their presence.

        While the above are not necessarily design errors, they indicate items that must be checked. Some HDL coding errors can result in unexpected latches or set/clear flip-flops.

d. The next section discusses the clocks found by the synthesizer. The logic type list discussed above will also note which of the clock resources were used, and give statements such as "clock found" or "clock inferred."

    1) If local clocks were used (i.e., clocks that do not use the global clock resources), they will show up here, e.g., when there are 4 clocks in an FPGA with 3 clock drivers. Local clocks potentially have much higher skew than is acceptable and their use should not be allowed for clocking sequentially adjacent flip-flops that are triggered on the same edge.

    2) If the routed clocks (CLKA/B) are used in RT54SX-S, RTSX-SU, or A54SX-A devices, the circuitry must be shown to incorporate appropriate skew tolerant design techniques.

    3) A table will show which clock edges have logic between them, e.g., from the rising edge to falling edge of HCLK, or between edges of different clocks. Carefully scrutinize logic crossing clock domains, and the symmetry requirements of clocks of which both edges are used.

The remainder of the .srr file contains timing analysis information that is calculated before place and route, and is thus of dubious value. The correct timing analysis will be shown by Timer when the .adb file is accessed; note that Timer will not guarantee minimum values. However, Timer will not analyze half-edge clock flip-flops or any asynchronous techniques on its own. Such instances will have to be analyzed by hand, in conjunction with Timer.

When reading the .srr file, carefully note any warnings given. Designs can synthesize even when warnings are given, but each warning must be dispositioned, after each synthesis run.

12.4.3 Back-End Tools: Using the .adb file

The .adb file contains the details of the design, including the actual netlist, timing analysis, pin information, etc. Designer incorporates a netlist viewer so that the reviewer may see the design in a schematic representation, although it is an awkward view (as most schematic generators produce). As noted above, the schematic given here, rather than the HDL description, is the real design

a. Check the temperature, voltage, and radiation settings for which the timing analysis was done. These should be the full military ranges for temperature and voltage, and whatever the program radiation requirement is. If the analysis is done with a reduced temperature or voltage range, analyses must be presented justifying the reductions. Note that the tools assume that temperature is the device junction temperature and not the case temperature of the temperature of the board's thermal control surfaces.

b. Run Timer to see how much timing margin there is. Even when the full military ranges are used, as above, there should be some margin for aging, inaccuracy in calculation, etc. A $\pm\,10\%$ margin for propagation delay is appropriate.

c. Run Pinedit to determine what I/O options were chosen. Verify that the choices were appropriate by comparing them with the inputs and outputs seen on the schematics.

d. Open the Netlist Viewer and view the schematic to resolve the issues found in section 3.2, above, especially to understand the unusual flip-flop usages found in section 3.2(c).

e. In the Netlist viewer, scan through the schematic looking for flip-flops with gated sets or clears, and to assure that all the settable and resettable flip-flops connect to a valid reset and are not involved in asynchronous design techniques. Starting at the reset or POR input, the reset lines can be highlighted and followed through all the pages.

f. In the Netlist viewer, ensure that all mission-critical and safety-critical circuits are implemented correctly. HDL synthesizers have been seen to implement poor circuits such as static hazards in clock generation circuitry.

12.5 Review the POR and Reset Circuitry and Power-Up Conditions

The ideal power-on-reset (POR) is asserted as soon as the power supplies are turned on and remains asserted until the voltages reach valid operating levels. Some factors may require that the POR be asserted beyond that point in the power-up cycle:

   a.   If there is an oscillator on the board, it should remain asserted until the oscillator has begun proper operation, which could be as long as tens of milliseconds after its power supply has reached a valid level.
   b.   If there are flip-flops that require the oscillator to be running in order to be reset, the reset must be asserted until these flip-flops are reset.

Beyond this, the criticality of the FPGA and its potential to cause damage to the spacecraft or an instrument, as discussed in section 2 above, may require careful scrutiny of its reset and power up/down conditions. During some portion of the power up time, the FPGA is not a circuit but simply a collection of unconnected gates, and transients may appear on its outputs.

External circuitry capable of causing damage or undesirable operation during power up and down, or during brown-outs, should be carefully reviewed to verify safe operation during these periods. For example, circuits constituting an arm and fire mechanism should not have both the arm signal and the fire signal originate in FPGAs that are powering up or down simultaneously. FPGAs should also not be used to pass POR signals to other circuits without sufficient care; this is a frequently seen cause of problems. External components should also be reviewed to determine whether special reset requirements exist. Notable in this class are EEPROMs, which must be protected during power-on, power-off, and other transient conditions such as brown-outs.

12.6 Review the Overall Design to Verify Correct Operation

The more difficult part of a design review is the verification that the circuit does what the specification says it is supposed to do. This includes, but it not limited to as seen above, going into the HDL and determining what it says the circuitry is supposed to do. One of the problems with HDL is that even if the design is broken into reasonably sized modules, the connectivity between the modules is difficult to determine. One method for dealing with this is the following:

   a.   Import the HDL modules into Actel Libero
   b.   Create a symbol for each of the modules
   c.   Using ViewDraw (in the Libero toolset) place the symbols on a large sheet and print it out.
   d.   Use colored pencils or highlighters to draw the connections between the modules.

This will give a high-level view of the FPGA design. Each module should be documented as it serves as a component in the hierarchical design. Ideally, components will be based on constructs that are essential building blocks of digital systems such as counters, decoders, logical units, sequencers, etc. Components of arbitrary functionality make the design of reliable systems, as well as their verification, difficult and error-prone. This is one of the results of a design approach that treats hardware as software.

The remainder of the review consists of examining the modules and reading the specification to determine how the various functions are implemented and whether their requirements are met. It may be useful to write test benches and simulate some of the modules or even parts of modules if their behavior is not obvious.

Pay careful attention to the timing of interfaces between the FPGA and external components, as these are not usually analyzed by designers.

12.7 References, Notes, and Related Documents

a. Suggestions for VHDL Design Presentation
b. "VHDL Design Review and Presentation," 2004 MAPLD International Conference
c. "A Designer's Checklist," 2004 MAPLD International Conference
d. OLD News #13: Minimum Delays and Clock Skew in SX-A and SX-S FPGAs
e. Index of DSCC Mil Specs & Drawings
f. "PCB Layout Issues," Design Seminar on Actel SX-A and RTSX-S Programmed Antifuses, Tuesday, April 13, 2004
g. "Case Study: Simultaneous Switching Outputs," Design Seminar on Actel SX-A and RTSX-S Programmed Antifuses, Tuesday, April 13, 2004
h. "Drive Strength." Design Seminar on Actel SX-A and RTSX-S Programmed Antifuses, Tuesday, April 13, 2004
i. "Sequential Circuit Design for Spaceborne and Critical Electronics," Rod Barto, 2000 MAPLD International Conference.
j. "Is It Safe?" from "Programmable Logic Applications Notes, EEE Links," August 1999.
k. "When Should You and When Should You Not Use VHDL?" 2004 MAPLD International Conference
l. "Some Characteristics of Crystal Clock Oscillators During the Turn-On Transient"
m. Appendix F of the WIRE Mishap Investigation Board Report, June 8, 1999.
n. "SX-S Output Transients"
o. "Act 3 Output Transients"

DIRECTIVE NO.   500-PG-8700.2.7-
EFFECTIVE DATE: 08/12/2005
EXPIRATION DATE: 08/12/2010

CHANGE HISTORY LOG

Revision	Effective Date	Description of Changes
Baseline	08/12/2005	Initial Release