

Prototyping RTAX-S Using Axcelerator Devices

Introduction

With the introduction of Actel's RTAX-S devices, designers now have access to the most powerful FPGAs available for aerospace and radiation intensive applications. Building upon the success of the RTSX-S family, the RTAX-S family offers higher densities, higher performance, and new features such as integrated RAM blocks and increased I/O standards.

Prototype verification is an important step in system integration where accurate behavioral simulation and static timing analysis are crucial. Since the enhanced radiation characteristics of radiation-tolerant devices are not required in this prototyping phase of the design, commercial Axcelerator devices can be used. This provides a cost-effective solution while maintaining the short time-to-market associated with Actel FPGAs. The prototyping flow presented in this application note is carefully designed to simplify the roadmap to production. The document describes the main architectural differences between the families and how to address the conversion issues that can occur as a result of these differences.

Conversion Considerations

Because RTAX-S devices are derived from the Axcelerator family, the basic architecture remains the same. With reliability in space being the overriding concern, all key RTAX-S features are hardened against radiation. Any Axcelerator features that could not be hardened to meet the stringent radiation requirements have been removed. The only exceptions to this are the Axcelerator's RAM FIFO controllers and the JTAG Boundary Scan Chain, which are still available in the RTAX-S. These are not hardened and they will not affect silicon reliability provided they are not used. Their design significance is discussed in the "Nonhardened Features" section on page 10. Table 1 provides an explicit listing of the changes made in the RTAX-S devices. (See Actel's website for future application notes discussing the differences between RTAX-S and Axcelerator features).

Table 1 • Major Architectural Differences between Axcelerator and RTAX-S

Feature	Modification from Axcelerator
Flip-Flops	D-type flip-flops in R-modules and I/O modules are implemented with Triple Module Redundancy (TMR) to ensure data bits stored within are SEU resistant
Clock lines	Clock lines and global signals are hardened to be SEU resistant
PLLs	The eight PLLs have been removed from the silicon
PerPin FIFOs	The PerPin FIFO and its controller have been removed from the silicon
LP Enable Mode	The Low Power Enable Mode has been removed from the silicon
Clock Input Delays	The Input Delay switch has been disabled
Packaging Pins	The pins dedicated for PLLs will be made as NC (no connect) pins to ensure backward compatibility with future Axcelerator hermetic devices. The GND/LP pin has been changed to GND
RAM blocks	While these <i>remain unchanged</i> in silicon, an Error Detection and Correction (EDAC) IP core is available through the ACTgen Macro Builder. Please see Actel's website for future application notes discussing EDAC
FIFO Controller	The internal RAM FIFO controllers are not radiation hardened, and their use is left to the discretion of the user. When left unused, these controllers do not interact with the rest of the device
Others	Changes to any AC parameters, timing and operating limits resulting from design modifications can be obtained by comparing the Axcelerator and RTAX-S datasheets

Synthesis tools do not prevent users from instantiating unsupported features such as PLLs and PerPin FIFOs, since the RTAX-S and Axcelerator devices share the same library for synthesis. However the unsupported features will be detected during design compilation within Actel's Designer software. Also, features such as the clock input delays and TRST pin selection are grayed out in Actel's Designer software to reflect their fixed status. Finally, the architectural differences between the two devices will yield different internal timing, which must be taken into consideration.

As a result of these differences, Actel developed a special RTAX-S prototyping flow to aid designers in using commercial Axcelerator devices as prototyping vehicles. In spite of the safeguards built into the software tool, the designer should adhere to the following two recommendations for a successful design:

- Designers are not to include any unsupported macros in their RTAX-S design (Table 2 on page 3).
- Designers should employ synchronous design techniques in order to reduce the risk of timing errors when prototyping.

RTAX-S Design Flow

RTAX-S uses the same design flow and tools as other Actel antifuse devices. Unlike the prototyping flow for older RadTolerant devices (e.g. RTSX-S), the designer will not begin work based on a commercial device but rather design for the RTAX-S from the beginning to the end. Actel's Designer software generates the fuse file for the commercial Axcelerator prototype device (Figure 1).

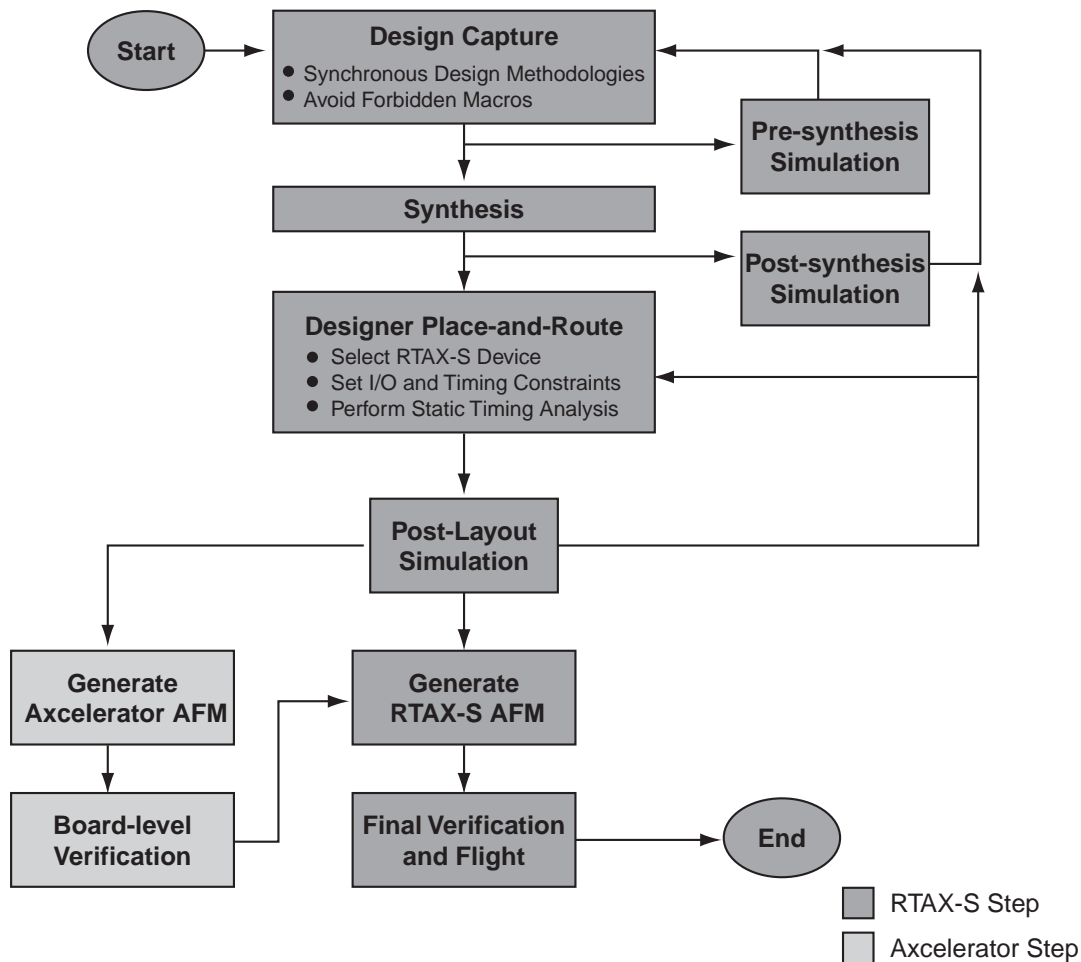


Figure 1 • The RTAX-S Prototyping Design Flow

This flow ensures that the design accurately reflects RTAX-S and eliminates the risk of the designer incorporating removed features or falling short of his/her timing requirements. In addition, the flow ensures that any errors are discovered early, instead of late in the design flow after the board-level verification has already been completed. Preventing re-layout and the subsequent re-verification preserves quick time-to-market. Therefore, it is essential that the design, pin assignment, place-and-route, timing verification, and simulation are done using the RTAX-S device. Actel's Designer software converts the RTAX-S design to a compatible Accelerator device and package. The reverse flow is not permitted.

List of Unsupported Macros

Table 2 lists the macros that cannot be included in the design source.

Table 2 • Unsupported Macros for RTAX-S Designs

Removed Feature	Forbidden Macros
PerPin FIFO	IOFIFO, IOFIFOCTL, DDR_FIFO
PLL	PLL, PLLFB, PLLINT, PLLOUT, PLLHCLK, PLLRCLK

Synchronous Design Methodology

Although RTAX-S devices are library compatible with Accelerator devices, architectural differences between the two devices make their internal timing significantly different. An asynchronous RTAX-S design that had its timing verified in the software will run faster in an Accelerator prototype, which might cause race conditions or other such violations rendering the prototype device useless for board-level verification. (The higher speed of Accelerator is also the reason why the RTAX-S internal performance could not be gauged through board-level verification). The designer should be aware of the consequences when mixing timing differences with asynchronous designs.

To minimize such timing related issues, Actel recommends that users implement fully synchronous designs and make use of both dynamic timing simulation tools and static timing analysis to diagnose any timing problems. Commercial design automation tools are generally optimized towards fully synchronous methodologies and do not account for potential asynchronous timing violations or glitches.

Fully synchronous designs follow three basic rules:

1. All registers that share the same data path should be connected to a common, low-skew, high-drive global clock network, as shown in Figure 2. Accelerator and RTAX-S devices have eight global clock networks that are accessible from special clock pins. These clock networks are the four hardwired clocks HCLKA/B/C/D and the four routed clocks CLKE/F/G/H. All eight networks can also be accessed from special internal clock routing macros.
2. Every element running on the same clock should be triggered on the same clock edge (either rising or falling). This practice removes any dependencies on the duty cycle of the clock.
3. If multiple clock domains are employed, and data must cross between these domains, the data should be resynchronized between the different clock domains using one or two registers to ensure that the data is always in a known state. This is illustrated in Figure 3 on page 4.

Implementing fully synchronous design techniques may use slightly more resources on the device, but the advantage is a more stable circuit with shorter debug time. The designer can obtain complete, reliable, and accurate data from timing verification and analysis in less time since worst-case static timing analysis will suffice in most cases.

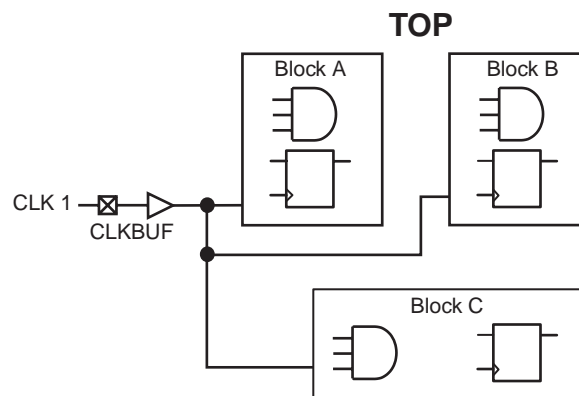
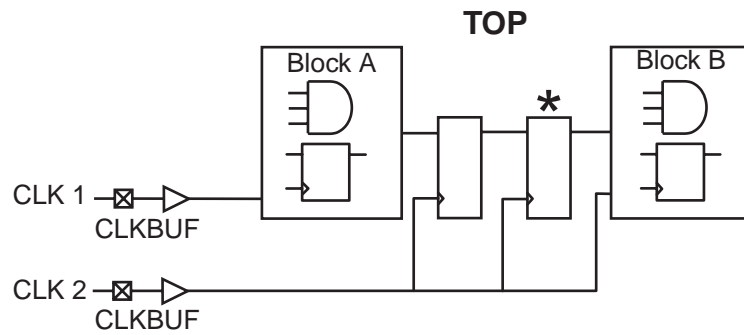


Figure 2 • Using the Same Clock Network for Each Element



Note *If the metastability settling time is relatively small compared to the clock period, the user may want to eliminate the second flip-flop.

Figure 3 • Synchronous Data between Clock Domains

Common Violations

Clock/Register Enable

Implementing a gated clock by using an AND gate to enable a clock is not recommended because it could result in a glitch on the clock signal (Figure 4). The registers available in Axcelerator and RTAX-S are implemented with an active low enable. Users should employ this enable to gate the register instead of trying to modify the clock signal, illustrated in Figure 5.

Using Combinatorial Logic to Drive a Clock

Another common practice is to drive a clock using combinatorial logic, as shown in Figure 6 on page 5. The problem occurs when the decoding of combinatorial logic generates glitches. A safer approach is to employ a system clock to drive the clock signal of the register and use the combinatorial logic as an enable, as shown in Figure 7 on page 5. There is no risk of a glitch being interpreted as a clock in this case.

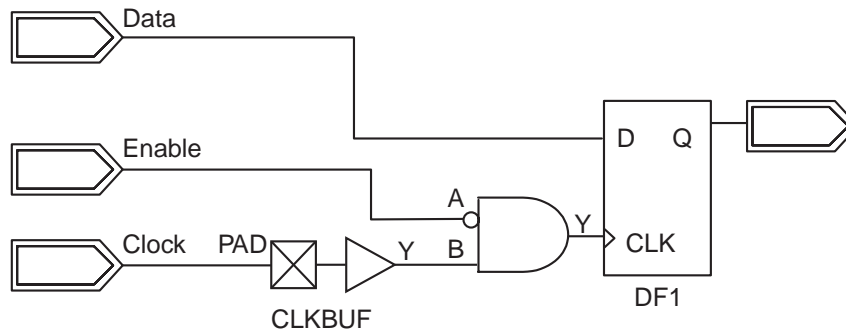


Figure 4 • Risky Practices of Using an AND Gate as an Enable

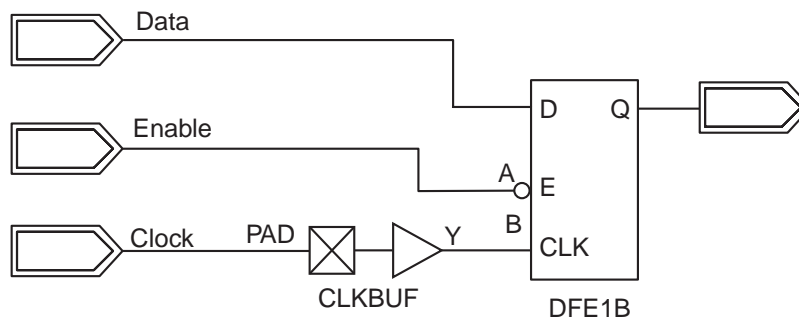


Figure 5 • Proper Use of Built-in Register Enable

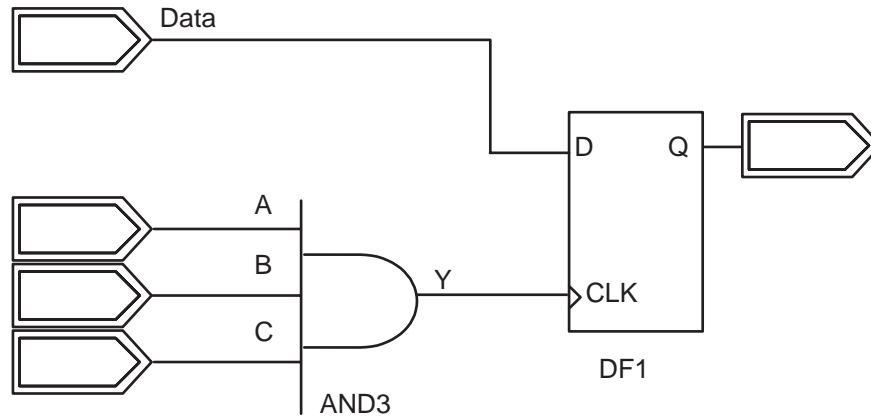


Figure 6 • Glitch Prone Combinatorial Logic Used as a Clock

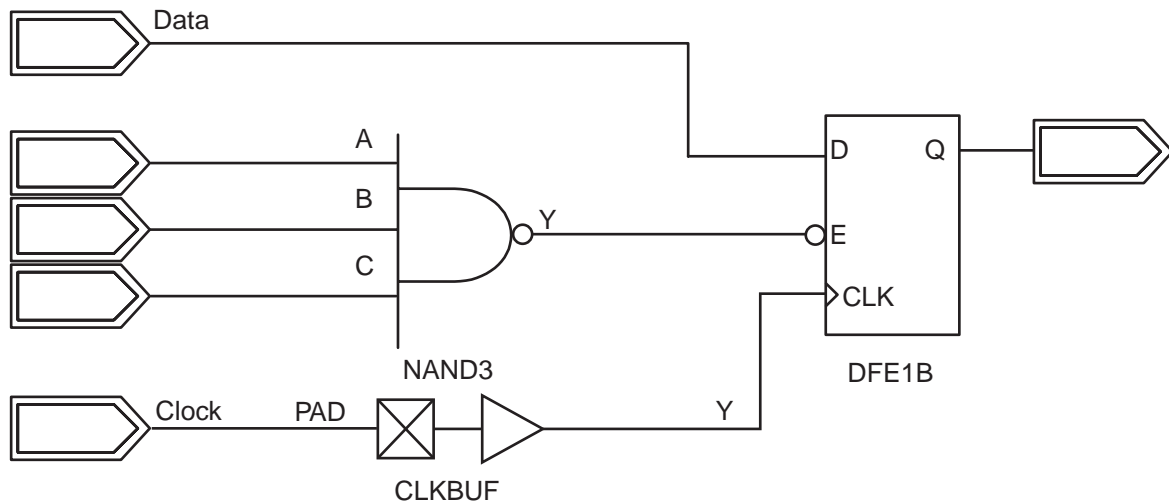


Figure 7 • Using Combinatorial Logic as Enable with a System Clock

Using a Divided Clock

A common way to divide a clock is to use a register and an inverted buffer for feedback. This method is acceptable to divide an incoming clock as it enters the device and redistribute it on a local routing or a global clock network. However, this method becomes a problem if there are many small divided clock networks that share data or merge data back to the original clock (Figure 8 on page 6). Since there is a limit to how many internally routed clock networks each device includes, many smaller divided clocks may not be able to use the global routing resources. In this case, the skew incurred by using normal routing resources as well as the delay incurred by dividing the clock makes the integration and analysis of these clock domains very arduous.

A preferable method in this case is to use the divided clock as an enable signal while clocking the registers from the original routed clock (Figure 9 on page 6). This allows the same system clock to be used throughout the circuit and reduces the number of global clock networks required for this section of the circuit. However, although more resources are required, the speed of the circuit and the ease of analysis may improve.

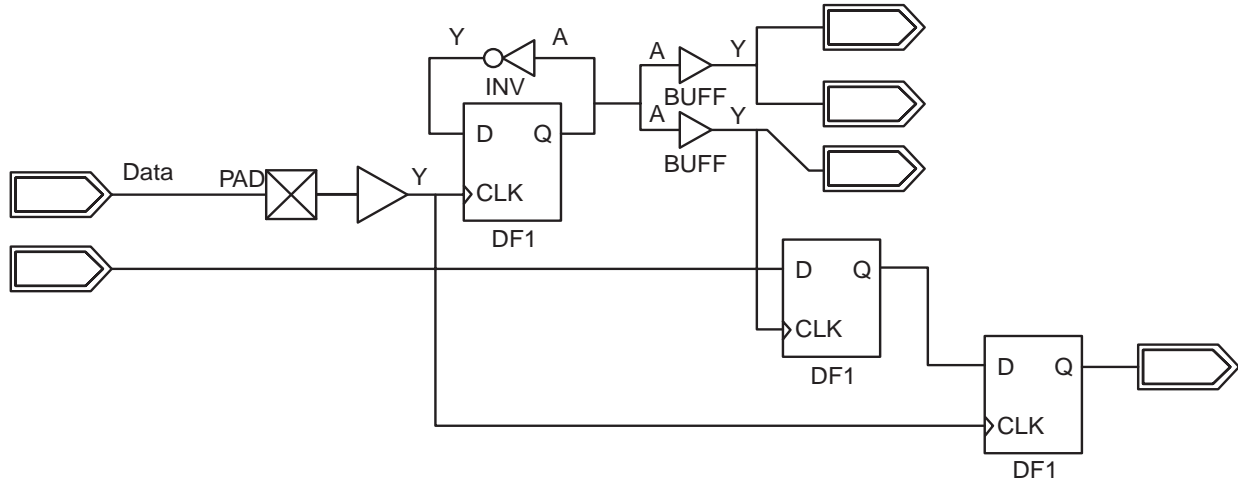


Figure 8 • Merging Clock Domains Back to the Original Clock

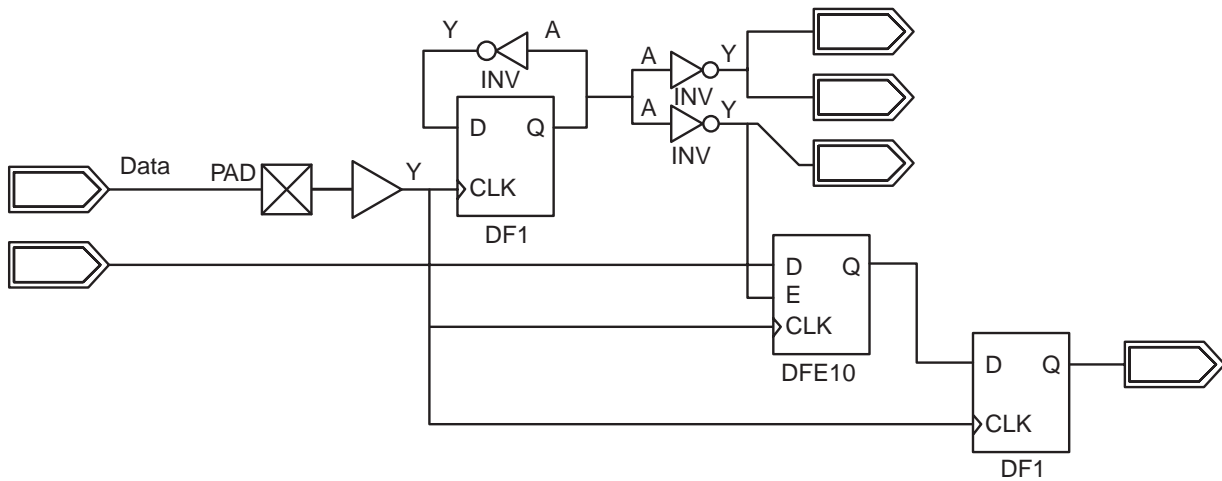


Figure 9 • Using a Single System Clock with Divided Clock Signals

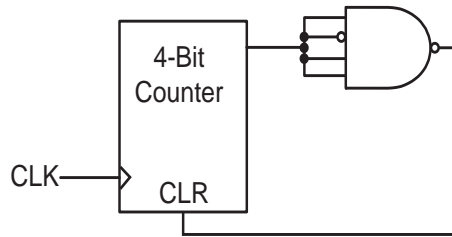
Asynchronous Feedback Loops

Variations in timing delays may make asynchronous feedback circuits unreliable. Actel recommends replacing these circuits with one that changes with the system clock (Figure 10). Note that the circuit shown in the left side of Figure 10 may glitch when the count changes from 0111 to 1000. A reliable timing analysis that proves the absence of a glitch to the asynchronous clear is close to impossible to

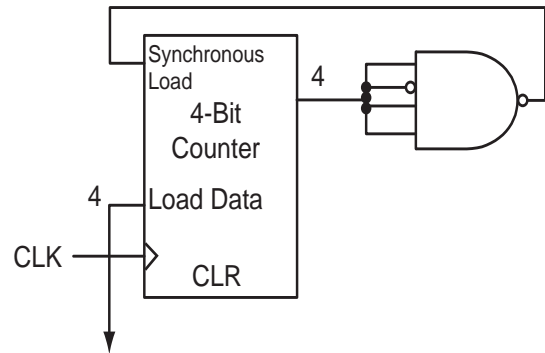
obtain. The circuit on the right side of Figure 10 shows a reliable replacement for the circuit shown on the left.

Asynchronous One-shot Circuits

Asynchronous one-shot circuits are susceptible to timing problems similar to the asynchronous feedback loop, as shown in Figure 11.

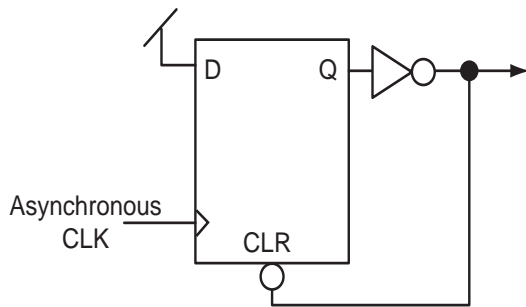
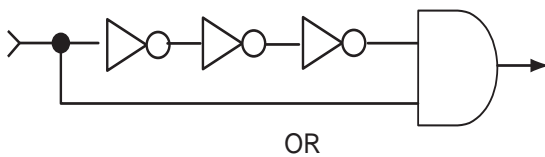


Asynchronous Feedback Bad Example

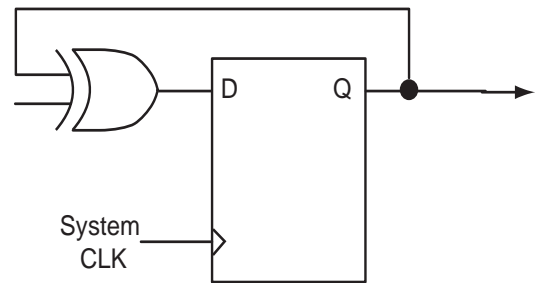


Asynchronous Feedback Good Example

Figure 10 • Asynchronous Feedback Circuits



Asynchronous One Shot Bad Examples



Synchronous One Shot Good Example

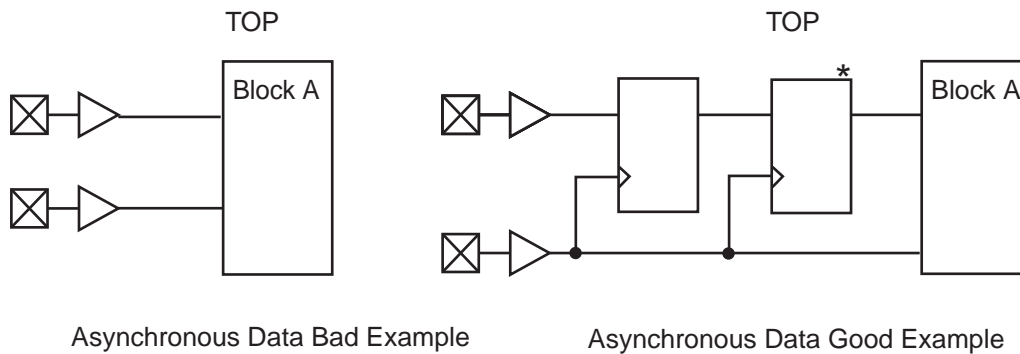
Figure 11 • Asynchronous One-shot Circuits

Delay Insertion

Because of variations in routing and process parameters, the use of delay-specific circuits should be avoided. Circuits that depend on a specific module or buffer delay are vulnerable to routing, process, temperature, and voltage variations. Additionally, the use of delay circuits is not advisable because most EDA tools optimize out any redundant delay circuits.

Asynchronous Data Sampling

The problems associated with sampling asynchronous data are similar to those encountered when transferring data from one clock domain to another. The use of one or two registers to buffer the data is recommended, as shown in Figure 12.



Note *If the metastability settling time is relatively small compare to the clock period, the user may want to eliminate the second flip-flop.

Figure 12 • Asynchronous Data Sampling

Synchronous Preset and Clear

If the circuit on the left in Figure 13 is preset, and data is clocked on the same clock edge, the circuit could suffer from metastability problems.

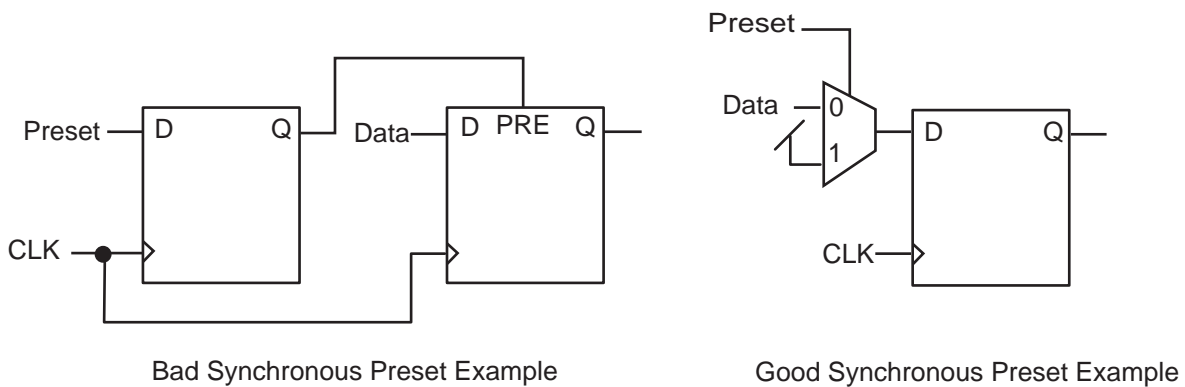


Figure 13 • Preset and Clear

Design Conversion to Axcelerator

After the design has passed functional and post-layout simulation in RTAX-S, the user can generate an .afm file for the prototype Axcelerator device. This automatic flow prevents the user from accidentally incorporating features exclusive to the Axcelerator device into the prototype device.

There are two ways to generate the Axcelerator AFM file:

1. Using the GUI: After saving your .adb file, click on Tools, and then click on “Generate Prototype. . .” as shown on Figure 14. The dialog box in Figure 15 opens. Select your options and click OK to generate prototype .afm file.
2. Executing the script directly: Click on Execute Scripts as shown on Figure 16. Enter the path to the prototype.tcl script file, add any parameters as necessary, and click OK to generate the prototype .afm file. For more information, refer to Actel’s *Designer* online help.

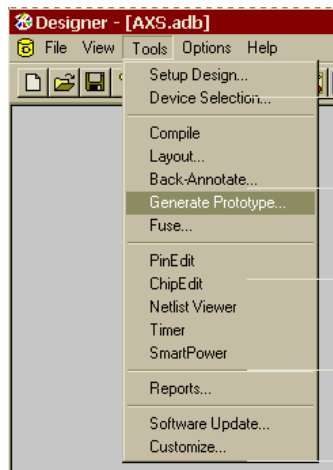


Figure 14 • Tools Menu Showing the Generate Prototype Command

Libero™ IDE and Other Software Tools

RTAX-S is incorporated into the software suite as a new member in the Axcelerator family, and hence there is no impact on the Libero Integrated Design Environment (IDE) project manager. Schematic capture, testbench generation, and simulation tools all share the same infrastructure and libraries as other devices in the Axcelerator family. In the initial releases, synthesis tools/libraries may not have

Table 3 • Package Migration Path Using Actel Extender

From	To	Extender Part Number
RTAX1000S 352 CQFP	AX1000 896 FBGA	Contact Actel Sales
RTAX2000S 352 CQFP	AX2000 896 FBGA	SK-CQ352A-RTAX
RTAX1000S 624 CCGA	AX1000 896 FBGA	Contact Actel Sales
RTAX2000S 624 CCGA	AX2000 896 FBGA	SK-CG624A-RTAX
RTAX2000S 1152 CCGA	AX2000 1152 FBGA	Contact Actel Sales

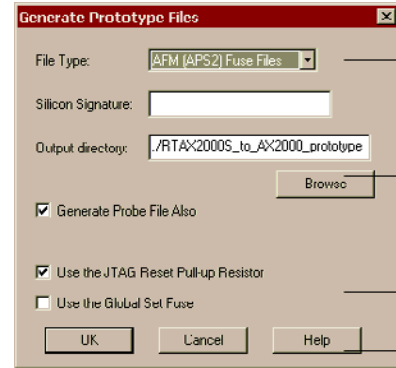


Figure 15 • Generate Prototype Dialog Window

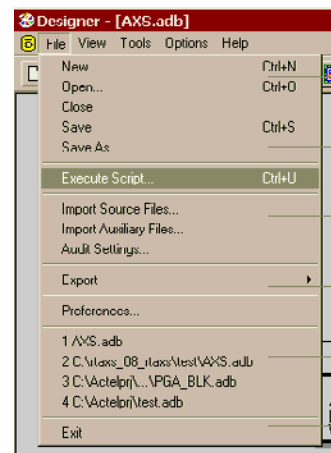


Figure 16 • File Menu Showing the Execute Script Command

RTAX-S devices; therefore, users should choose the Axcelerator. Users are expected to include the RTAX-S synthesis libraries when they become available to obtain a more accurate performance report.

Migration Path and Package Extender

Since the RTAX-S packaging uses the Ceramic Column Grid Array (CCGA), and the Axcelerator uses the Fine Ball Grid Array (FBGA), the board must include a prototype adapter with the appropriate footprints to accommodate prototyping. Table 3 lists the package migration path available and the part number for the associated Extender. Please contact your local Actel sales representative for ordering information.

Nonhardened Features

The following features are not radiation hardened:

- The built-in RAM FIFO controllers are not SEU enhanced for RTAX-S devices. Actel does not recommend using them in devices that will be exposed to radiation. While they are fast and do not consume user gates, these RAM FIFO controllers are vulnerable to upset. Designers should be aware that an SEU can result in the loss of the RAM FIFO counting state or flag errors that can only be corrected by clearing the FIFO. In addition, the memory implemented using these controllers is subject to SEU since they do not use EDAC. A user employing RAM FIFOs will be warned by Actel's Designer software during design netlist compilation. Users are encouraged to employ EDAC IP techniques to mitigate the SEU in the RAM blocks and to implement a FIFO controller employing user gates. This will help to ensure that the RAM FIFOs and associated memory are resistant to SEU.
- The JTAG circuitry is duplicated from the AX architecture and is not hardened against radiation. Users must keep the extended TRST pin asserted and cannot utilize the JTAG boundary-scan logic during flight.

For more information, please consult the *RTAX-S Family FPGAs* datasheet.

Moving to Production

When board-level verification is completed and the design functions as expected, simply open the RTAX-S .adb file in Actel's Designer software and generate an .afm file to program the radiation tolerant part. No further timing verification is necessary, since this was already done before the board-level verification.

Conclusion

Designers using Actel's RTAX-S devices have three distinct advantages over those using radiation-tolerant ASICs. Users can begin by employing commercial design techniques and readily-available EDA software. They save money by prototyping with equivalent commercial devices. Furthermore, users can be assured of the quality and reliability of the silicon, since the parts shipped are thoroughly screened. RTAX-S devices, derived from the commercial Accelerator family, share the same architecture and use the same design tools. This makes it very easy for users to move from design conception to prototyping and then to full production.

Provided that users adhere to the design flow and other guidelines described in this application note, any potential conversion issues can be easily mitigated thereby greatly reducing the amount of time required for the design cycle.

Actel and the Actel logo are registered trademarks of Actel Corporation.
All other trademarks are the property of their owners.



<http://www.actel.com>

Actel Corporation

955 East Arques Avenue
Sunnyvale, California 94086
USA

Tel: (408) 739-1010

Fax: (408) 739-1540

Actel Europe Ltd.

Dunlop House, Riverside Way
Camberley, Surrey GU15 3YL
United Kingdom

Tel: +44 (0)1276 401450

Fax: +44 (0)1276 401490

Actel Japan

EXOS Ebisu Bldg. 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150 Japan

Tel: +81 03-3445-7671

Fax: +81 03-3445-7668

Actel Hong Kong

39th Floor
One Pacific Place
88 Queensway
Admiralty, Hong Kong
Tel: 852-22735712