
ProASIC^{PLUS} in Casino Gaming Applications

Introduction

The casino gaming industry has undergone many changes in the last decade. Since the introduction of the first mechanically-based slot machine in the late 1890s, the slot machine has cycled through hundreds of iterations to reach its present state of a networked, interactive, multimedia platform. Integrating the latest technology, today's themed slot machines are more attractive to consumers, offering vivid graphics and exciting sound effects that rival the latest electronic games and media.

Slots are the most popular form of casino gambling, accounting for a staggering 40–50% of total casino revenue. Since these games require minimal support overhead and lock in a fixed ratio of profits, casinos are increasingly focusing on slot games and shifting away from table games. The demand for these next-generation machines is growing as the life cycle of these slots decreases due to the higher expectation of the gamers. In short, fast time to market is crucial to slot manufacturers.

As these systems become more sophisticated, programmable logic has become a key technology to enabling the rapid development of these advanced systems. These devices typically provide everything from glue logic to critical control and processing functions within the system.

To ensure the integrity of slot machines, gaming manufacturers must get approval for their machines before they can be placed on the casino floor. Gaming regulators employ complex systems to verify that the slot machines pay out consistently with the system's defined odds, and cannot be tampered with. When a large jackpot has been won, it is also crucial for casinos to be able to verify that the critical components of the system have not been modified or altered in any way from the original design. Game developers must select a programmable logic technology that will enable them to build complex digital systems while simultaneously preventing illegal modification or tampering. In some cases, as an added security measure, it is advised to select an FPGA technology that can be audited and checked for proper functionality once deployed in a system.

There are three primary FPGA technologies available to logic designers and system architects; Antifuse, SRAM, and Flash. Antifuse, a one-time programmable technology, is considered the most secure programmable technology by most experts, and has long been favored by industry and government for the most challenging design problems. Reprogrammability has created a significant demand for SRAM-based FPGAs, but as SRAM FPGAs require an external configuration device, they are easily cloned by intercepting and copying the bitstream. For designers who require a secure reprogrammable solution, Actel's second-generation, Flash-based ProASIC^{PLUS} FPGAs provide a reprogrammable, secure technology platform with densities of up to a million gates, memory, and PLLs, as well as support for in-circuit programming. This application note details how these new devices can be used to address the security and regulatory requirements for modern gaming systems.

Security

Unlike SRAM-based FPGAs that require external configuration devices, Actel flash-based ProASIC^{PLUS} devices are nonvolatile single-chip FPGAs that are live at power-up, and that provide a comprehensive series of barriers to tampering and design theft. Moreover, the design information is stored in nonvolatile memory cells which are effectively small capacitors, and any physical deconstruction of the device will disrupt the programmed data. Even though today's gaming systems already employ sophisticated security measures, the overall security of every gaming system can be further enhanced using Actel ProASIC^{PLUS} devices. For more information, please refer to Actel's *Design Security in Nonvolatile Flash and Antifuse FPGAs* white paper.

Actel offers two types of security:

FlashLockTM

The FlashLock feature in ProASIC^{PLUS} works via a mechanism where the user locks or unlocks the device with a user-defined key (79 bits to 263 bits). When the device is locked, functions such as device write, verify, and erase are disabled. Without the correct key, the design in the FPGA cannot be copied or modified. To gain access to the FPGA, the device must first be unlocked using the correct key. Since the maximum clock frequency of the JTAG port is 20 MHz, using a brute-force technique to crack the security key would take about a billion years for the shortest user-defined security key.

Permanent FlashLockTM

The purpose of the permanent lock feature is to provide the highest level of security for ProASIC^{PLUS} devices. The permanent FlashLock feature will create a barrier that prevents any access to the contents of the device. This barrier is created by breaking the key after the device has been secured. After permanently locking the device, access to the device will not be possible even with the proper key. The device is effectively rendered as a one-time programmable circuit.

With these security features, critical circuitry in gaming designs can be made safe from tampering. Data communication between different components within the system can be secured further by using Actel's CoreAES128 intellectual property (IP) core that provides 128-bit AES encryption or decryption. This eliminates the need for expensive physical security measures and other tamper-resistant enclosures to prevent tampering with odds.

For detailed information on how to generate the programming file with different security features, please refer to the *Implementation of Security in Actel's ProASIC and ProASIC^{PLUS} Flash-Based FPGAs* application note.

Checksum Verification

Whether the user is using FlashLock or permanent FlashLock, the user can always identify the device by reading back the design name and CHECKSUM for the device. The design name can be defined in the Designer software, under Tools > Setup Design > Design Name (Figure 1). The CHECKSUM is a unique design-dependent value which is automatically generated when the user generates a programming file. Using the FlashPro / FlashPro Lite programmer together with the FlashPro programming software, the design name and CHECKSUM can be read back by performing the action DEVICE_INFO (Figure 2 on page 3).

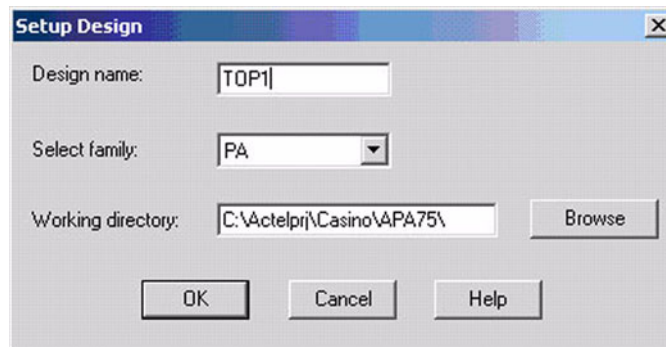


Figure 1 • Change the Design Name

It is also possible to use a local processor to communicate with the FPGA via its JTAG interface for in system verification. Further details on this can be found on the Actel website under the *DirectC* section.

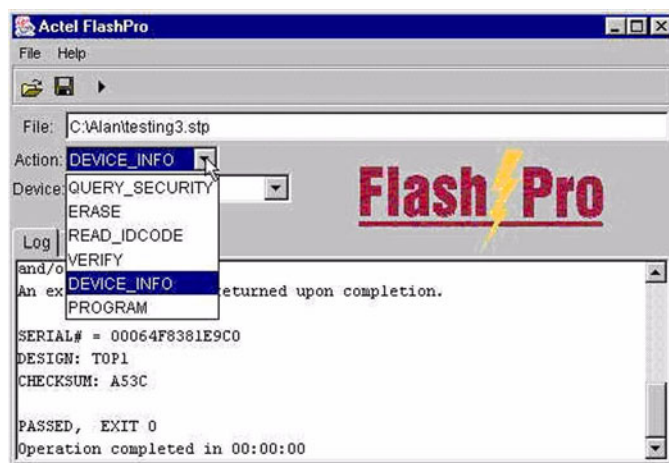


Figure 2 • Checksum Verification Using FlashPro Software

In-System Verification with FlashLock™

If the device is not locked by Permanent FlashLock, the content of the device can also be verified using the programming file. Today's progressive jackpots can reach millions of U.S. dollars. With such high stakes, regulatory bodies such as the various state gaming commissions often require a system to ensure the device has not been tampered with. This often requires the system to be inspected thoroughly, and critical components to be checked to ensure they are the same as originally installed. This requires not only that a component be locked to prevent changes, but also that a system be employed to verify the content of the device bit by bit. With the use of the FlashLock security feature in ProASIC^{PLUS}, users can employ the security key to unlock the device, and perform an in-system verification using one of Actel's device programmers to certify that the content of the device has not been tampered with.

To perform the in-system verification, either a casino technician is required to use Actel's device programmer or the manufacturer must design this capability into the system with a local processor together with the original programming file for verification. With the current generation of FPGAs, this programming file contains the FlashLock key and should be secured in either a trusted environment or encrypted into local system memory for the processor to use only when performing the device verification. Please refer to for instructions for performing in-system verification.

Repeating the Design Process to Verify Final Programming File

Each slot machine must be approved before it can be placed on the casino floor. Manufacturers must submit their systems to gaming regulators for licensing and verification. Either the gaming regulators or an independent testing laboratory will perform a set of sophisticated tests on these electronic gaming machines to ensure the machines comply with the standards specified by the local regulatory agency.

To certify the gaming machine, the FPGA device controlling the critical functionalities of the machine and the related software must also be tested. Inspection of the source code and other design files may be required to ensure that no "back door" or other embedded code exists that might cause the system to operate inappropriately. To demonstrate this, game regulators may require access to the source code (HDL or netlist). In the case of programmable logic designs, regeneration of the programming file is required to confirm that the same programming file was supplied by the game developer. Checksum verification can be performed to verify the device is programmed by the programming file supplied by the game developer. If the device is not locked by permanent FlashLock, in-system verification can also be performed to verify the content of the embedded device with the programming file bit by bit.

To ensure that the identical programming file can be regenerated by Designer software using the original netlist, it is important to use the original SDC timing constraint file, the GCF constraint file exported in the Post-Layout stage, and the same random seed value (default value 0) as defined in the original design. The

SDC timing constraint file (*.sdc) is a Tcl-based constraint file, and the GCF constraint file (*.gcf) is an ASCII file specifying design constraints for ProASIC and ProASIC^{PLUS} families. These optional constraint files specify the design constraints for the place-and-route tool to ensure that a design meets the timing performance and the required pin assignments. The types of constraints that can be defined in a GCF constraint file include:

- Global resource constraints
- Netlist optimization constraints
- Placement constraints

The original user-defined GCF file(s) imported into Designer may not have all the design constraints used in the Compile and Layout stages because users can use MultiView Navigator to add constraints to their design. To reproduce the same place-and-route result, the user needs to use the Post-Layout GCF file. To export the Post-Layout GCF file, the user must finish the Compile and Layout stage, then go to File > Export > Constraint Files (Figure 3 on page 5) and select All GCF constraints in the dialog box. The Post-Layout GCF file and the SDC timing constraint file contain all the design constraints used in the place-and-route. Please refer to the *Designer User's Guide* for the available constraints and syntax. Under the same operating system platform, the original netlist, optional SDC and Post-Layout GCF constraint files running on the same Designer software version will always generate the same programming file. The header information, such as time stamp or filename-related information in the STAPL or BIT programming file, will be different from the original programming file. This difference in the header information will not affect the in-system verification or checksum.

Using the same netlist file(s), SDC timing constraint file and GCF constraint file exported in the Post-Layout stage, Actel's Designer software can regenerate the same programming file under a few conditions. These conditions are summarized as follows:

- Same Actel's Designer & Synthesis tool version with the same configuration settings
- Same operating system platform

Actel's Designer software has a proprietary place-and-route algorithm, and a newer version of the software release will generally have new features and performance enhancements. As a result of these changes, the user may not be able to generate the same programming file if a different software version is used.

The place-and-route algorithm has a random seed generation algorithm that depends on the floating point unit of the central processing unit. The floating-point unit has a special set of instructions that focus on large number mathematical operations. For different computer platforms, the number of bytes used to represent the floating-point number is not the same. For example, Intel or AMD processors use ten bytes for floating-point number representation, while a UNIX-based workstation uses only eight bytes. The difference in the architecture of the processor will result in a different random seed number, and a different random seed number will generate a different placement and routing for the same design. The default value of the random seed number is 0, and users generally do not need to modify this value. Users can change this default value by using the Option > Set Variable command in the Designer software (Figure 4 on page 5), or it will be changed automatically when the Use Multiple Passes option is selected in Layout Option. The variable name for the random seed number is PROASIC_QDPLACER_RANDOM_SEED, and its value ranges from 1 to $2^{32} - 1$. Please note that a different random seed value can produce a different performance result for the same design.



Figure 3 • Export Post-Layout GCF File Dialog Box

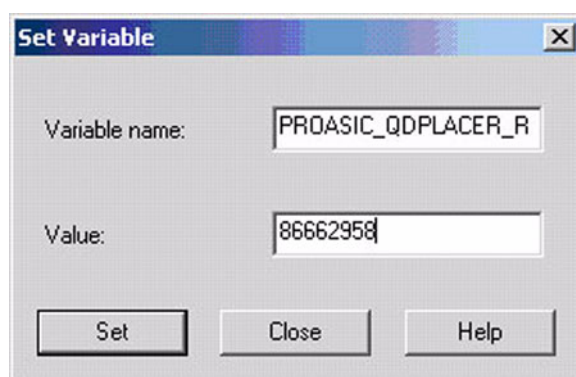


Figure 4 • Change the Default Random Seed Value

Using the Tcl script

One easy way to make sure that the GUI setting in Designer for a given design is the same as another is to make use of the Tcl script. Commands executed in the Designer GUI can be exported to a Tcl script and the user can later execute the Tcl script. To export a Tcl script, go to menu File > Export > Script Files. To run a Tcl script, go to menu File > Execute Script and select the corresponding Tcl script.

Below is a sample Tcl script that runs through the whole design flow. Please note that when you are trying to regenerate the same programming file using the Tcl script, the original netlist and Post-Layout GCF file must be used and imported using the `import_source` Tcl command. If a non-default random seed number was used, the same random seed number must be used with the `placer_seed` option in the Layout command. For a detail Tcl script command, please refer to the [Designer User's Guide](#).

Sample Tcl Script:

```
#Setup a new design and device information
new_design -name "TOP1" -family "PA" -path {..} set_design -name "TOP1" -family "PA" -
path {..}
set_device -die "APA075" -package "208 PQFP" -speed "STD" -voltage "2.5" -jtag "yes"
-trst "yes" -temprange "COM" -voltrange "COM"

#Import netlist and Post-Layout file
#Post-Layout file can be exported when compile and layout stages were completed.
#It contains all the design constraints used in the design
import_source -format "edif" -edif_flavor "GENERIC" \
```

```
{.\top.edn} -format "gcf" -merge "no" \  
{.\post_layout_gcf.gcf}  
  
#Compile and Layout the design using the default random seed value  
compile  
layout -timing_driven -place "On" -place_incremental "Off" -route "On" -  
route_incremental "Off"  
  
#Export the programming file with the security key  
export -format "STP" -lock_mode "keyed" -key "5829AF5DF7D66FC22CB2" \  
{.\apa75_spine1.stp}  
#Save the Design  
save_design \  
{.\apa75_spine1.adb}
```

Sample Tcl script for non-default random seed number:

```
layout -timing_driven -place "On" -place_incremental "Off" -placer_seed 86662958 -  
route "On" -route_incremental "Off"
```

Conclusion

With the unbreakable security features offered by the Actel ProASIC^{PLUS} family, users can decide which security setting best suits their application. Actel's flash-based technology delivers an ideal solution by providing users with a single-chip device that is live at power-up, secure, and highly reliable. By including ProASIC^{PLUS} FPGAs with FlashLock, users can be confident that their designs will not be compromised, while still being allowed a choice of methods for in-system verification.

Related Documents

Application Notes

Implementation of Security in Actel's ProASIC and ProASIC^{PLUS} Flash-Based FPGAs
www.actel.com/documents/Flash_Security_AN.pdf

In-System Programming ProASIC^{PLUS} Devices
www.actel.com/documents/APA_External_ISP_AN.pdf

User's Guides

Designer User's Guide
www.actel.com/documents/designer_UG.pdf

Silicon Sculptor User's Guide
www.actel.com/documents/sculptor_DOS_UG.pdf

FlashPro User's Guide
www.actel.com/documents/flashpro_UG.pdf

Miscellaneous

DirectC
www.actel.com/download/program_debug/directc/default.aspx

Design Security in Nonvolatile Flash and Antifuse FPGAs
http://www.actel.com/documents/DesignSecurity_WP.pdf

Appendix A: In-System Verification

The following section details the necessary steps in performing in-system verification with Actel's device programmers. The verification process will first verify the programmed bit and then the erased (unprogrammed) bits on the device as specified in the programming file. This verification process ensures that every single bit programmed on the device is 100% equivalent to the programming file.

Using FlashPro or FlashPro Lite for In-System Verification

1. From the File menu, click Connect. The FlashPro Connect to Programmer dialog box displays. (Figure 5)
2. In the Port list, select the port the FlashPro programmer is connected to.
3. In the Configuration list, select the ProASIC^{PLUS} device family.
4. From the File menu, select Analyze Chain. Chain details appear in the Log window.
5. Select your devices. In the Device list, select your device before you perform any action. If you have only one device in the chain, performing Analyze Chain selects that device automatically from the Device list. If you have multiple devices in the chain, you must select the device you wish to verify.
6. Load the STAPL (*.stp) file by clicking on the Open File button in the toolbar. Select your STAPL file and click Open.
7. Select the VERIFY action from the Action list and click the Execute button in the toolbar.
8. Use the default settings as they appear in the Execute Action dialog box. Click the Execute button to start the verification process.

Successful verification will result in Exit 0 and is shown in Figure 6 on page 8. If the content of the device does not match the STAPL file, FAILED, Exit 11 appears in the Log Window, as shown in Figure 7 on page 8.

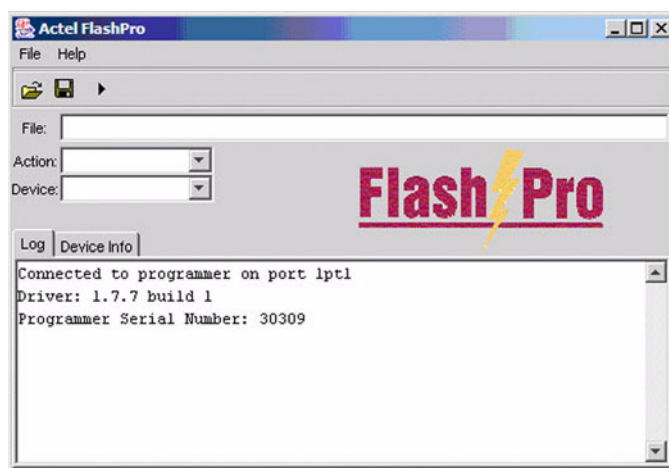


Figure 5 • Connect in FlashPro or FlashPro Lite

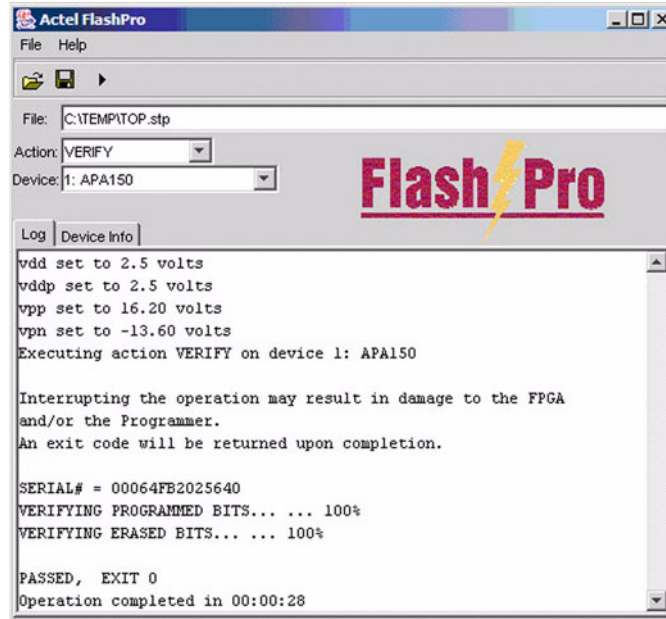


Figure 6 • Verification Completed Successfully

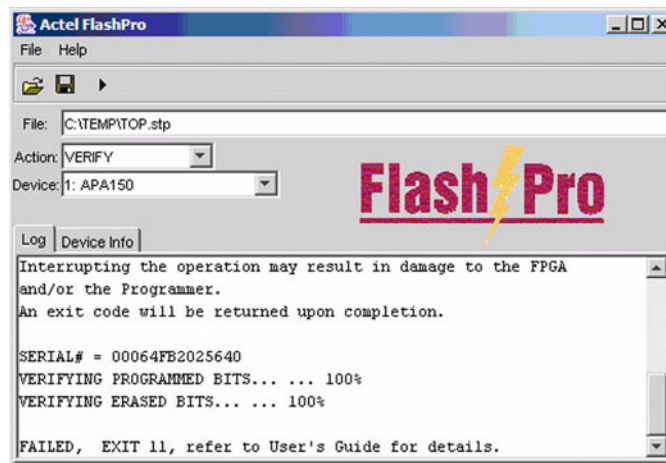


Figure 7 • Verification Failed

Using Silicon Sculptor 2 for In-System Verification

Besides using FlashPro or FlashPro Lite to perform the verification, the user can also use the Silicon Sculptor 2 tool to perform the verification. This section gives you an overview of how to perform the in-system verification using Silicon Sculptor 2 with the corresponding adaptor module.

1. Click Device to select the device and verify its content.
2. Click Data Pattern to load the programming file (BIT or STAPL) that contains the security key.
3. Click the Verify Tab on the GUI and click VERIFY (Figure 8).

A successful verification will give a Function Complete message with an EXIT 0 code (Figure 9 on page 10). If for any reason the content of the device does not match the programming file, an error message will be displayed.

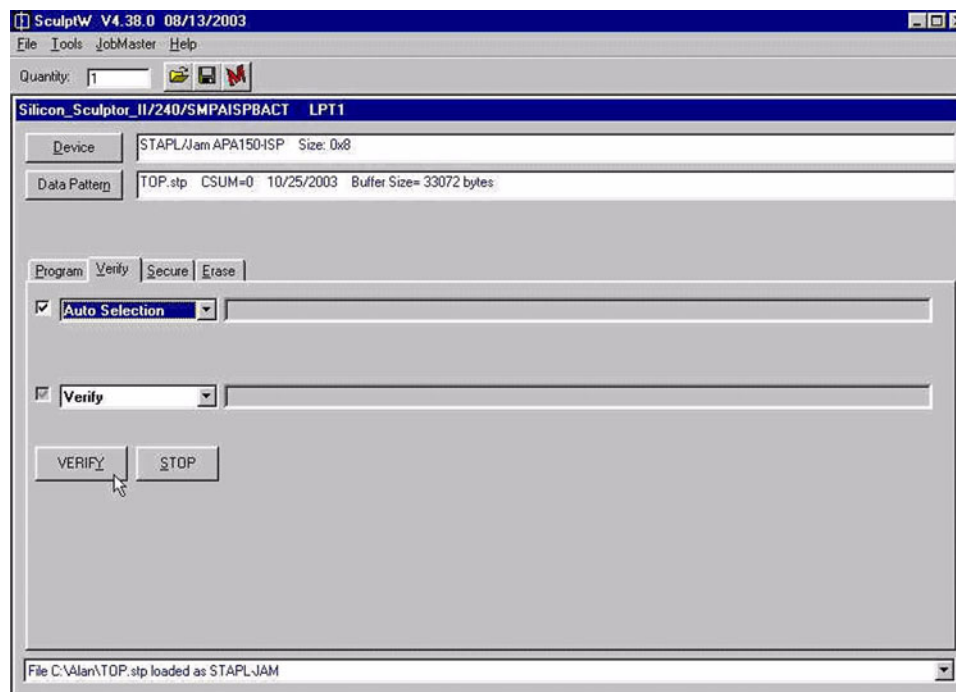


Figure 8 • Performing In-System Verification

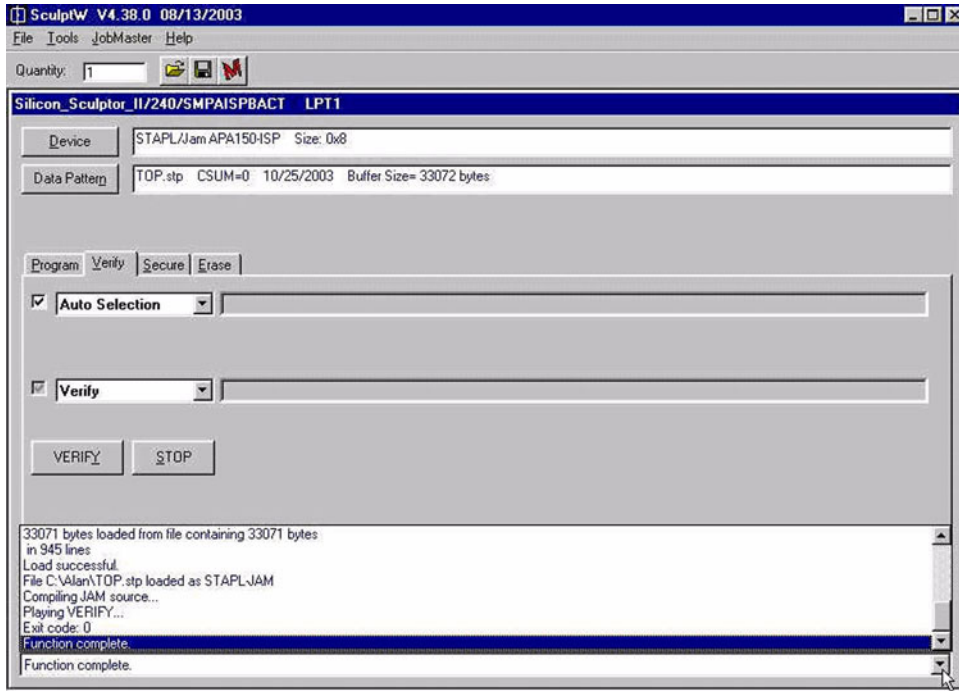


Figure 9 • Verification Completed Successfully

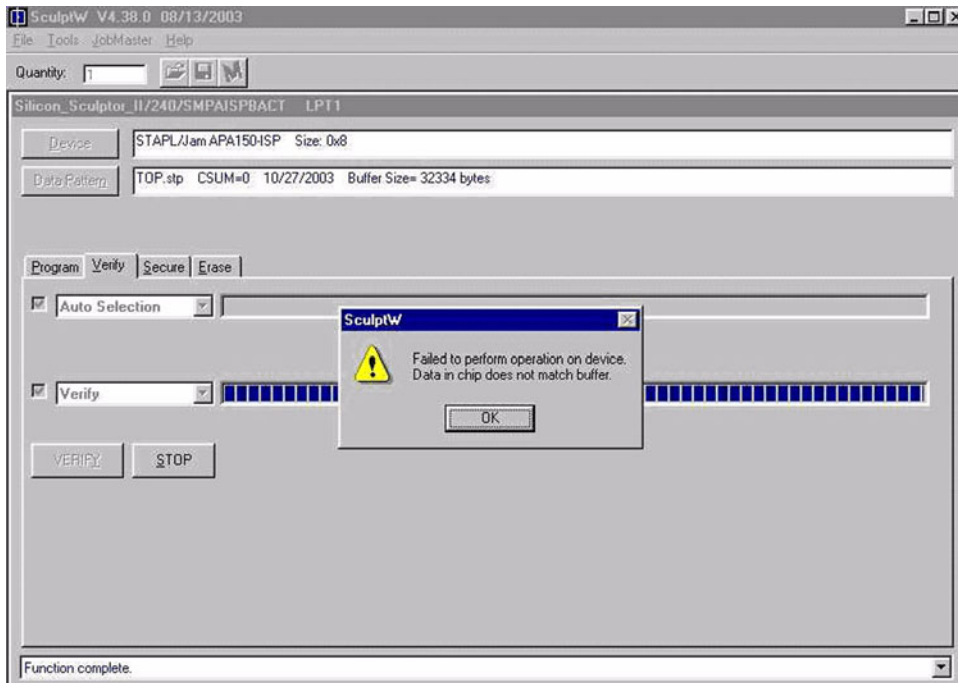


Figure 10 • Verification Failed



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

2061 Stierlin Court
Mountain View, CA
94043-4655 USA

Phone 650.318.4200
Fax 650.318.4600

Actel Europe Ltd.

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

Phone +44 (0) 1276 401 450
Fax +44 (0) 1276 401 490

Actel Japan

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

Phone +81.03.3445.7671
Fax +81.03.3445.7668

Actel Hong Kong

39th Floor, One Pacific Place
88 Queensway, Admiralty
Hong Kong

Phone +852.227.35712
Fax +852.227.35999