
FlashPro User's Guide v8.4

Actel Corporation, Mountain View, CA 94043

© 2008 Actel Corporation. All rights reserved.

Printed in the United States of America

Part Number: 5029138-13

Release: July 2008

No part of this document may be copied or reproduced in any form or by any means without prior written consent of Actel.

Actel makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability or fitness for a particular purpose. Information in this document is subject to change without notice. Actel assumes no responsibility for any errors that may appear in this document.

This document contains confidential proprietary information that is not to be disclosed to any unauthorized person without prior written consent of Actel Corporation.

Trademarks

Actel and the Actel logotype are registered trademarks of Actel Corporation.

Adobe and Acrobat Reader are registered trademarks of Adobe Systems, Inc.

Mentor Graphics, Precision RTL, Exemplar Spectrum, and LeonardoSpectrum are registered trademarks of Mentor Graphics, Inc.

WaveFormer Lite is a registered trademark of SynaptiCAD, Inc.

Synplify is a registered trademark of Synplicity, Inc.

Sun and Sun Workstation, SunOS, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc

Synopsys is a registered trademark of Synopsys, Inc.

Verilog is a registered trademark of Open Verilog International.

Viewlogic, ViewSim, ViewDraw and SpeedWave are trademarks or registered trademarks of Viewlogic Systems, Inc.

Windows is a registered trademark and Windows NT is a trademark of Microsoft Corporation in the U.S. and other countries.

UNIX is a registered trademark of X/Open Company Limited.

All other products or brand names mentioned are trademarks or registered trademarks of their respective holders.

Table of Contents

About FlashPro v8.4.....	9
Programming Tool Model Overview.....	9
Getting Started	13
Installing FlashPro Software.....	13
FlashPro3 Programmer Installation.....	19
FlashPro3 Driver Installation Rules.....	20
FlashPro Hardware Installation	21
FlashPro Lite Hardware Installation	22
FlashPro3 Hardware Installation.....	22
Starting Standalone FlashPro	23
Creating a New Project.....	24
Opening a Project	26
Saving a Project.....	26
Understanding Parallel Programming	26
Understanding Serialization.....	27
Understanding SVF	28
Understanding 1532 File Format.....	29
Creating a Programming Database (PDB) File in Designer	31
Understanding the FlashPro GUI.....	33
Understanding the Toolbar.....	34
Understanding the Menu Bar	36
Understanding the Flow Window	39
Understanding the Log Window.....	39
Understanding the Status Bar	40
Understanding the Programmer List Window.....	40
Understanding the Programmer Details Window.....	42
Understanding the Single Device Configuration Window.....	43
Understanding the Chain Configuration Window.....	47
Customizing the Toolbar.....	49
Customizing the Programming Window	50
Preferences	52
Programming Settings and Operations	55
Introduction	55
Programmer Settings Information.....	55
Ping Programmers	59

Performing a Self-Test	59
Scanning a Chain.....	59
Enabling and Disabling Programmers.....	59
Renaming a Programmer.....	60
Removing a Programmer.....	60
Selecting Programmers	60
Single Device Configuration.....	61
Single Device Programming	61
Loading a Programming File.....	62
Select Target Device	63
Chain Settings	65
Serial Settings	66
Skip Serial Data	66
Reuse Serial Data.....	67
Serialization with Parallel Programming	67
Chain Programming.....	69
Understanding the Chain Order.....	69
Introduction to Multiple Device Chain Programming.....	69
Chain Configuration Window.....	71
Chain Editing	74
Using the Organize Buttons in the Chain Programming Grid	75
Cutting, Copying, and Pasting Devices from the Chain.....	75
Removing Devices from the Chain.....	76
Moving Devices within the Chain.....	77
Chain Editing	79
Adding an Actel Device.....	79
Adding an Actel Device from Files	79
Adding a Non-Actel Device	79
Configuring a Programmer.....	81
Selecting an Action	81
Using Serialization	81
Modifying Programming Settings in FlashPro with a PDB File	83
Configuring Security	85
Configuring Security, FPGA and FlashROM Settings in FlashPro.....	85
Configuring Security Settings in FlashPro	86
Custom Security Settings.....	88
Configuring FPGA Array Settings in FlashPro	92
Configuring FlashROM Settings in FlashPro.....	93



Express Configuration	94
AFS Programming	95
Introduction	95
Programming File Actions for AFS Devices	95
ProASIC3 Family Programming	99
Introduction	99
Programming File Actions for ProASIC3 Family Devices	99
ProASIC ^{PLUS} and ProASIC Families Programming Introduction	101
Generating Programming Files.....	103
Generate a Programming File.....	103
Generate a Programming File for CoreMP7/Cortex-M1 Device Support.....	104
Generate a Programming File for AFS Device Support.....	106
Generate a Programming File for Serialization Support in In House Programming (IHP).....	107
Programming Embedded Flash Memory Block.....	110
Programming the FPGA Array	112
Programming the FlashROM.....	112
Silicon Signature	113
Programming Security Settings	114
Custom Security Levels	115
Custom Serialization Data for FlashROM Region	121
Custom Serialization Data File Format.....	122
Basic Programming Tutorials	125
Single STAPL/PDB File Basic Tutorial	125
Single Actel Device with Serialization Tutorial.....	130
Chain Programming Tutorial.....	137
Modifying Memory Contents and Programming a Device.....	145
Modifying FlashROM Contents and Programming a Device Tutorial.....	147
Programming Only Security Settings Tutorial	151
Fusion Calibration Backup and Recovery Tutorial.....	153
Advanced Tutorials	157
Multiple Device Chain Programming	157
Multiple Device Serialization Chain Programming	164
Multiple Programmer Multiple Device Chain Programming	172
Multiple Programmer and Multiple Device Serialization Chain Programming	179
Parallel Port Cable Information.....	188
Importing and Exporting Files.....	191
Importing Configuration Files.....	191

Exporting Configuration Files.....	191
Exporting a Chain STAPL File	191
Exporting Single Device STAPL Files.....	191
Exporting Single Device SVF Files	192
Exporting Single Device 1532 Files.....	192
Using Hot Keys	195
General Hot Keys	195
Single Device Programming Hot Keys.....	195
Chain Programming Hot Keys.....	196
Batch Mode	196
About TCL Commands	197
Running Tcl scripts from within FlashPro	198
Running Tcl Scripts from the Command Line	199
Exporting Tcl Scripts from within FlashPro	199
add_actel_device	200
add_non_actel_device	201
close_project.....	201
configure_flashpro_prg	202
configure_flashpro3_prg	202
configure_flashproLite_prg.....	203
connect_cable.....	203
copy_device	204
cut_device.....	204
dump_tcl_support	205
enable_device	205
enable_prg.....	206
enable_prg_type	206
enable_procedure	207
enable_serialization	207
export_config	208
export_script.....	208
export_stapl.....	209
export_single_stapl.....	209
export_single_svf.....	210
export_single_1532	210
new_project	211
open_project.....	212
paste_device	212
ping_prg.....	213
refresh_prg_list	213
remove_prg	214



run_selected_actions	215
save_log	215
save_project	216
save_project_as	216
scan_chain_prg	216
select_serial_range	217
select_target_device	218
self_test_prg	218
set_bsdl_file	218
set_chain_param	219
set_device_ir	219
set_device_name	220
set_device_order	220
set_device_tck	221
set_device_type	221
set_main_log_file	222
set_prg_name	222
set_programming_action	223
set_programming_file	223
set_programming_mode	224
set_serialization_log_file	224
set_serialization_mode	225
Signature	225
update_programming_file	225
Troubleshooting	231
Loopback Test	231
Fusion, IGLOO, and ProASIC3 Families Exit Codes	231
ProASIC ^{PLUS} and ProASIC Exit Codes	236
Electronic Parameters	245
DC Characteristics for FlashPro3	245
DC Characteristics for FlashPro Lite	246
DC Characteristics for FlashPro	247
Electronic Specifications	249
FlashPro3	249
FlashPro Lite	250
FlashPro	253
FlashPro3 Characteristics	255
FlashPro and FlashPro Lite Characteristics	255
Illustration of the JTAG Switching Characteristics	256
Actel Headquarters	256

Technical Support.....256
Customer Service.....257

Product Support..... 259

Customer Service.....259
Actel Customer Technical Support Center259
Actel Technical Support259
Website259
Contacting the Customer Technical Support Center.....259



About FlashPro v8.4

FlashPro v8.4 is Actel's new, redesigned programming software tool for the Flash family of devices (IGLOO, Fusion, ProASIC3, ProASIC^{PLUS}, and ProASIC families). You will be able to navigate easily through the FlashPro software because of its similarities with other Actel software tools. The FlashPro software includes the following new features:

- Supports direct multiple Actel device chain programming and serialization
- Supports single device STAPL files generation
- Supports single device SVF files generation
- Supports single device IEEE 1532 files generation
- Supports Chain STAPL file generation
- Supports a single GUI to drive multiple FlashPro3 USB programmers for parallel programming
- **Note:** Parallel programming via FlashPro (USB/LPT1) or FlashPro Lite programmers is not supported.
- Supports device serialization for parallel programming
- A redesigned GUI, which features a project manager to manage the programming files and data
- Enhanced In-System Programming (ISP) Support

An optional In-House Programming (IHP) service is available if you are purchasing Actel devices in volume. Contact [Actel](#) for more information.

For step-by-step instructions on how to use these features, see the [FlashPro Tutorial](#).

Programming Tool Model Overview

The FlashPro software is designed for use in the operation, user design, and production programming flows.

Design Debug

The figure below illustrates the programming design flow when an engineer is in debug mode. In the programming design flow, the new Programming files (STAPL/ PDB) are generated for a design change and are sent to the FlashPro software for testing and debugging the design.

Note: FlashPoint is integrated into Designer; therefore, the STAPL file is generated from Designer. FlashPro v6.2 and greater can be used to export STAPL files from PDB files created by FlashPoint (Designer).

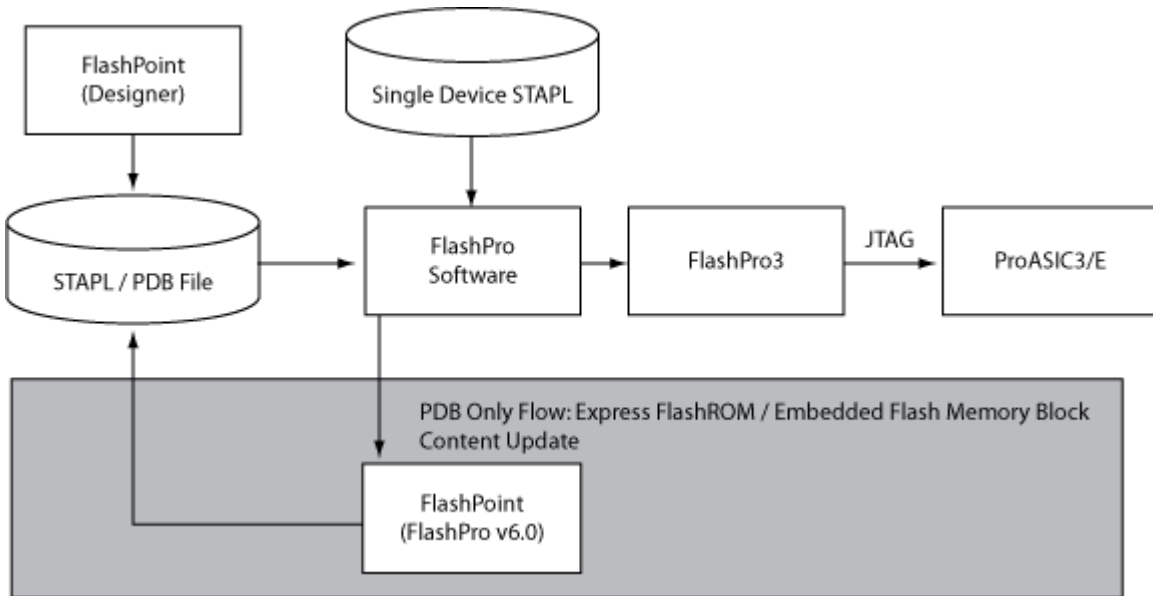


Figure 1 · Programming Design Flow

Operation/Production Planning

The figure below shows an illustration of the operation flow. In this illustration, the production coordinator generates the programming files (STAPL/PDB) file with or without serialization and/or security settings (see Programming application note for further information). The production coordinator loads the programming file in the FlashPro software to set up the configurations for production programming, such as Serialization options, Action selections, and Procedure selections, etc.

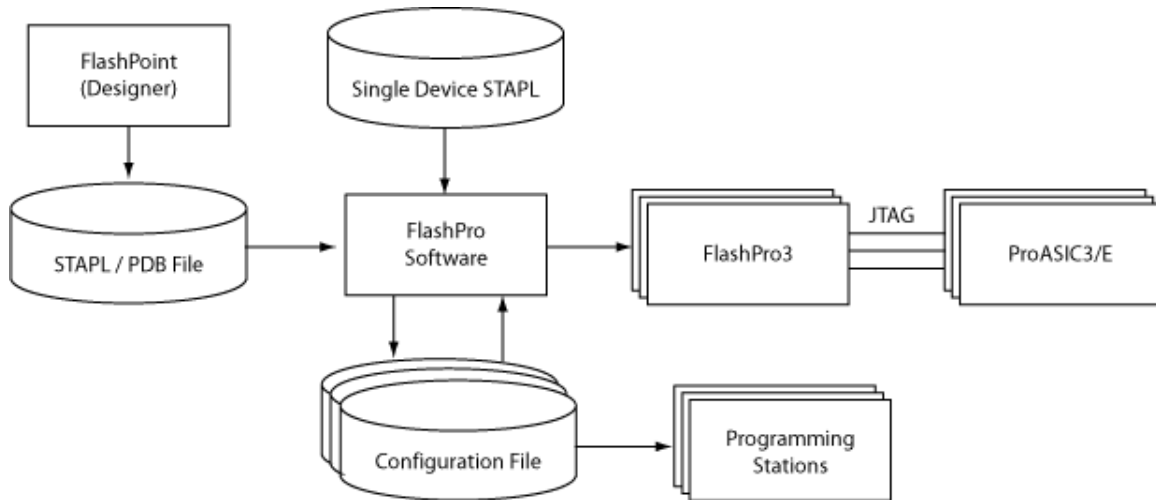


Figure 2 · Operation Flow

The production coordinator may want to generate different configuration files for each programming station (depending on the logistics and serialization options). For example, if the Programming file contains 10,000 serial data and the production coordinator decides to split the serial data designation to one thousand for each programming station, then ten configuration files will be generated (one for each of the ten programming stations). However, if you are not using serialization, you only need one configuration file.



Note: The production coordinator can test the configuration files with one or more FlashPro3 programmers before sending it to the production programming floor. If PDB files are used in the production flow, warning icons may appear on the FlashPro/FlashPoint GUI because the automatic audit cannot find the source file on the production environment; the PDB file contains the valid programming data. If STAPL files are used, currently loaded STAPL files will be audited on execution of an action to determine if the original STAPL file has been modified. If it has not been modified, the action will continue to run. If it has been modified you will be prompted to reload the modified STAPL file, or to continue running the current action. If you select to reload the modified STAPL file, all previous programming settings will be refreshed and will need to be performed again.

Operation/Production Programming

The figure below shows an illustration of the production programming flow. The operator imports the configuration file and begins programming the devices by clicking the **Run** button. The operator's interaction with FlashPro should be limited.

At the end of a programming session, the serialization log file (if applicable) and the programming log file are sent back to the production coordinator for record keeping.

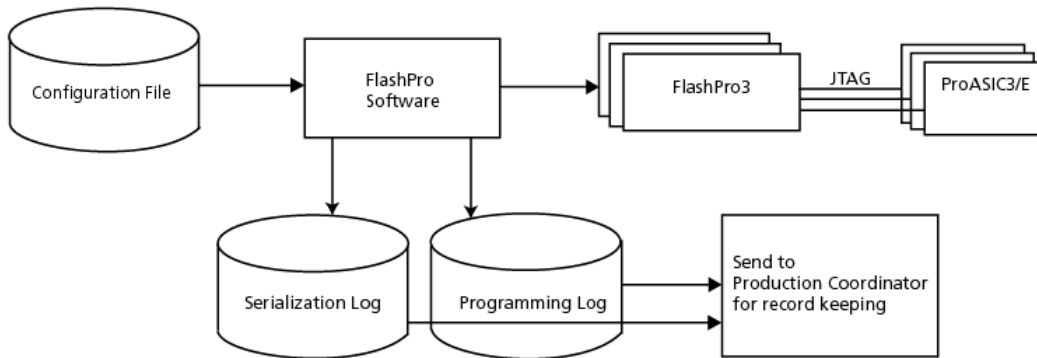


Figure 3 · Production Programming Flow

Express Configuration Programming (IGLOO, Fusion, and ProASIC3 family devices only)

The figure below illustrates the Express Configuration Programming Flow. In this flow, you can program the security setting into the IGLOO, Fusion, and ProASIC3 family device directly from the FlashPro software.

Note: FlashPoint is integrated into the FlashPro v6.0 and later software.

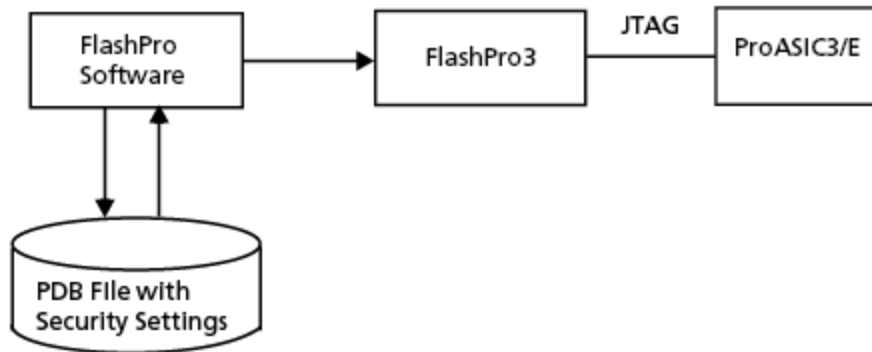


Figure 4 · Express Configuration Programming Flow



Getting Started

Installing FlashPro Software

Before installing the FlashPro software, make sure your USB drivers are up to date.

FlashPro is included in the Actel Libero IDE software and automatically installs in the following program folder: **Actel Libero IDE > FlashPro**. You can also download a standalone version of FlashPro from the Actel website <http://www.actel.com>. For both installation options, follow the FlashPro **InstallShield Wizard** instructions.

To install the FlashPro software:

1. Insert the FlashPro CD into your drive or double click on the appropriate link from the Actel website.
2. The **InstallShield Wizard** starts.
3. From the InstallShield Wizard for FlashPro, click **Next** to see the license agreement.
4. Click **Yes**, and then **Next** to accept the license agreement.
5. If you have FlashPro3/FlashPro (USB) hardware connected to your PC, disconnect it now. See figure below.

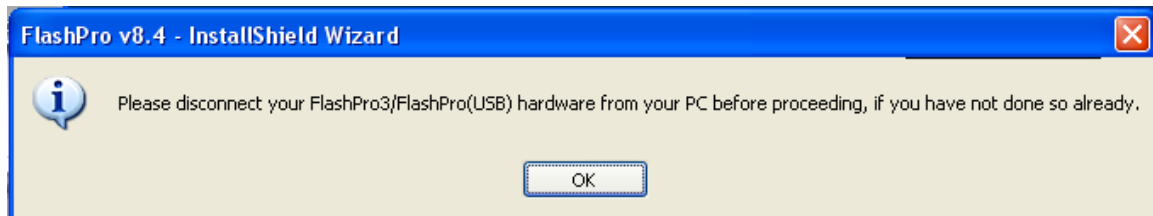


Figure 5 · Disconnect FlashPro3/FlashPro(USB) Hardware

6. Click **Next** to install FlashPro to the default destination folder. To install to a different folder, click **Browse** and select another folder.
7. Click **Next** to complete your installation. The FlashPro software installs.
8. After the software is installed, you can connect your FlashPro3/FlashPro (USB) hardware to your PC. Locate the manual USB drivers for FlashPro3 if manual driver installation is necessary. See figure below.

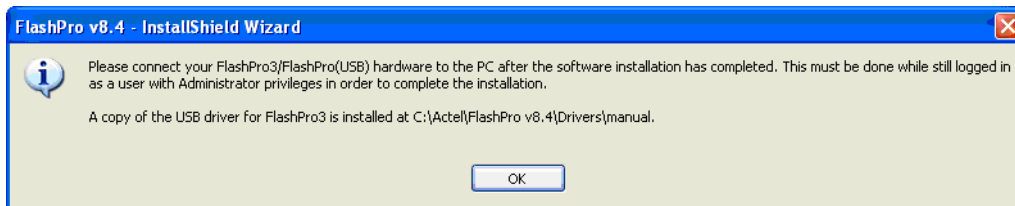


Figure 6 · USB Driver Location

9. Click **OK**. Software setup is complete. You do not need to reboot the computer. You are asked if you would like to reboot. See figure below. Click **Finish** to close the Wizard.

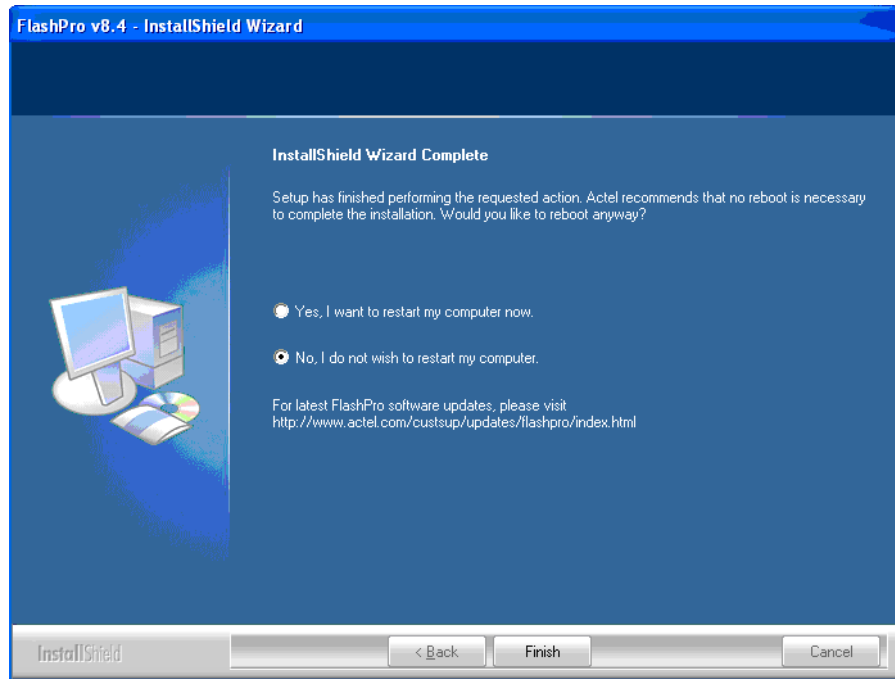


Figure 7 · Software Setup Complete

10. If you have plugged in a FlashPro programmer, the **Found New Hardware Wizard** starts as shown in the figure below. If you are going to use FlashPro3 and have not already plugged in the FlashPro programmer, plug it in now.



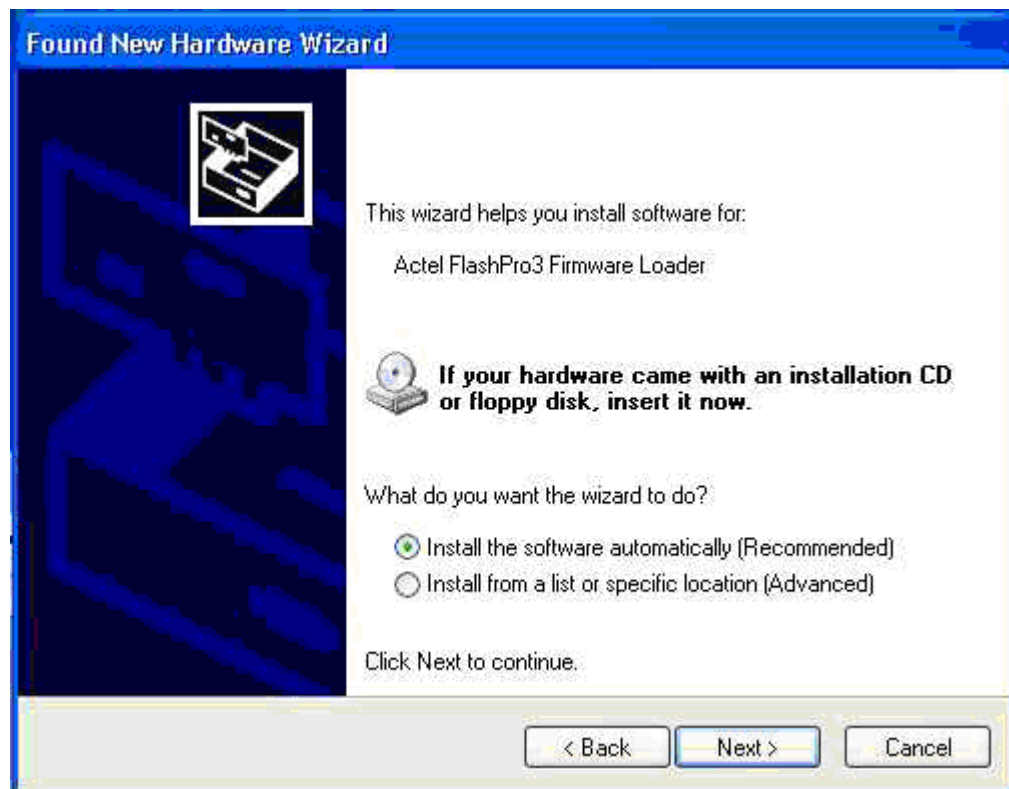


Figure 8 · New Hardware Wizard Welcome and Installation Options

11. The **New Hardware Wizard** asks you to select from two installation options (Recommended or Advanced). Choose **Recommended** and click **Next**.
12. In some cases, you may be asked to specify the driver location from the **Files Needed** dialog box. If so, click the **Browse** button to find your driver location (if necessary). Once the file location is in the **Copy Files From** text box, click **OK**.
13. The **Found New Hardware Wizard** displays as shown in the figure below.

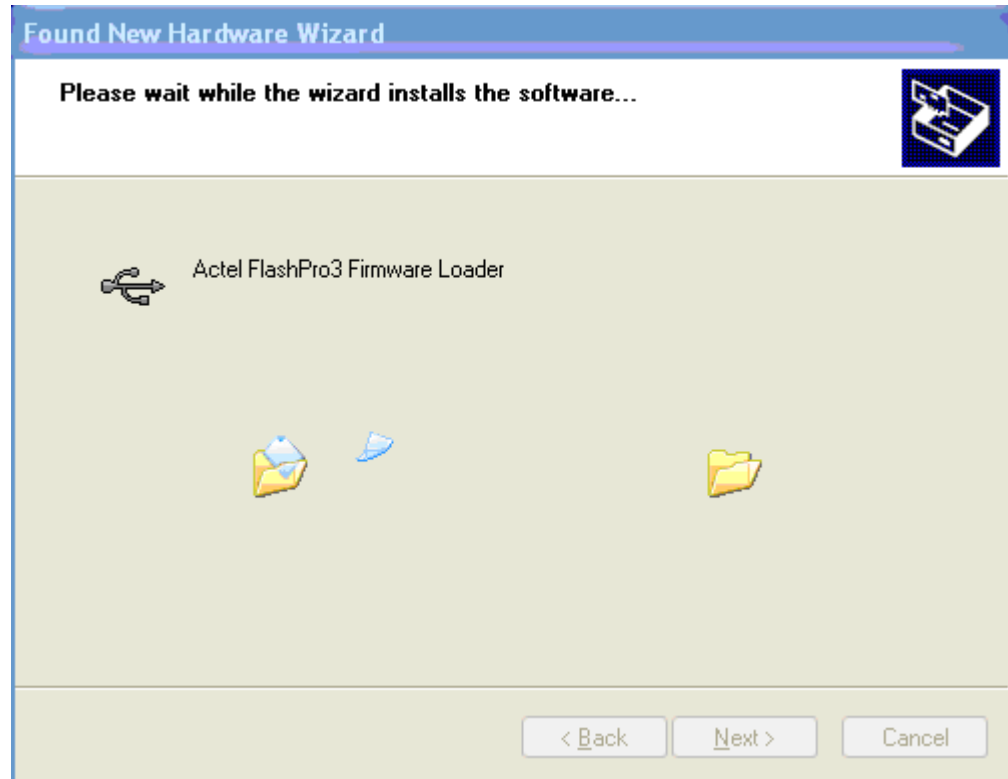


Figure 9 · FlashPro3 Firmware Loader

If you are using Windows XP, the Hardware Installation box displays a warning message (see figure below).



Figure 10 · Hardware Installation Dialog Box

14. Click Continue Anyway. The Found New Hardware Wizard completes the installation as shown in the figure below.



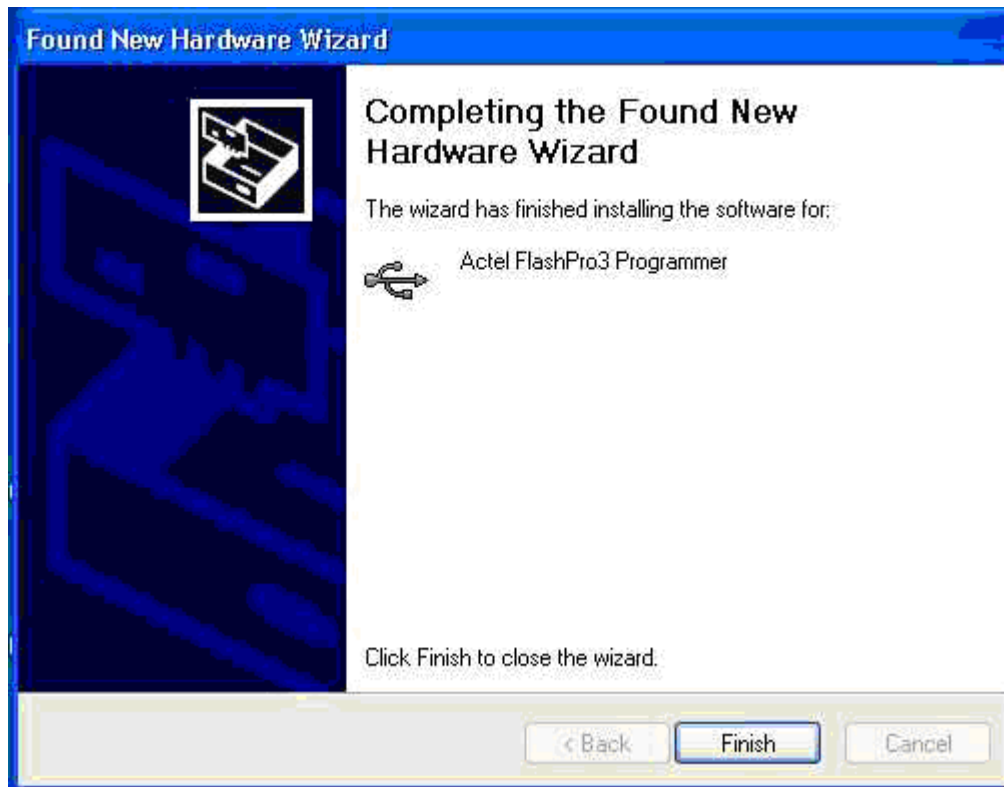


Figure 11 · Programmer Software Installation Complete

Click **Finish** to complete the hardware installation process

The **New Found Hardware Wizard** starts again (see figure below). The **New Hardware Wizard** asks you to select from two installation options (Recommended or Advanced). Choose **Recommended** and click **Next**.

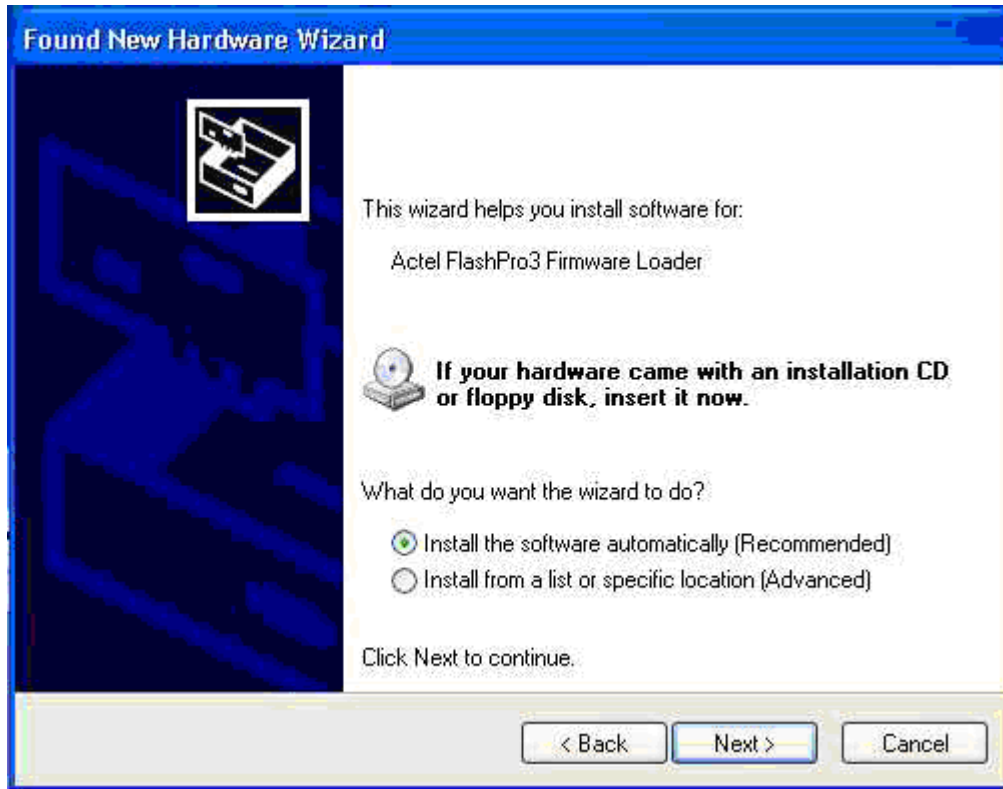


Figure 12 · FlashPro3 Firmware Loader Installation Options

In some cases, you may be asked to specify the driver location from the **Files Needed** dialog box. If so, click the **Browse** button to find your driver location (if necessary). Once the file location is in the **Copy Files From** text box, click **OK**. If you are using Windows XP, the **Hardware Installation** box displays a warning message (see figure below).

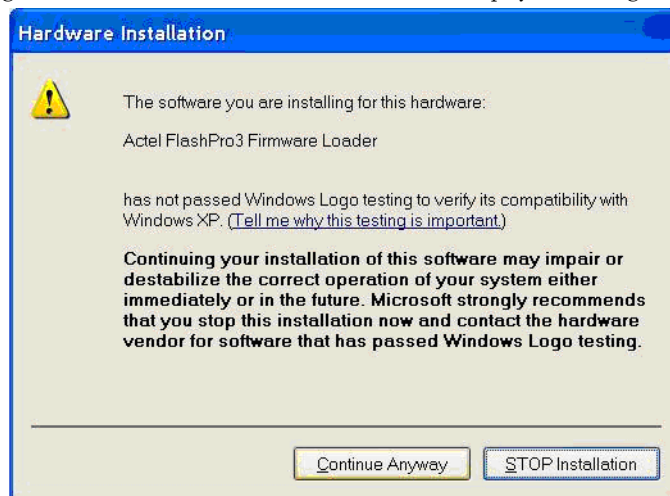


Figure 13 · Hardware Installation Dialog Box

Note: Click Continue Anyway.



15. The Found New Hardware Wizard completes the installation as shown in the figure below. Click **Finish** to complete the hardware installation process .



Figure 14 · Programmer Software Installation Complete

Note: If at any time you want to remove the FlashPro software, FlashPro will provide a warning message from the Question box asking you, "Are you sure you want to remove Actel FlashPro v8.4?" Click **Yes**, or **No**.

FlashPro3 Programmer Installation

The new FlashPro3 dedicated driver provides a whole host of usability benefits compared to the previously used HID driver. However, as it is a completely new driver, older versions of the FlashPro software will no longer work once the new software version is installed. The following bullet points explain the additional guidelines you must follow when the automatic software removal tool encounters older versions of the software.

- When the FlashPro v5.1 software is detected, the FlashPro v6.0 installer will uninstall FlashPro v5.1 and then exit the program. You must manually run the FlashPro v6.0 installer again to install FlashPro v6.0.
- Registry cleanup does not always work, due to permissions issues – if the FlashPro3 programmer is not recognized, you need to follow the instructions in FlashPro3 driver [installation rules](#).
- If you try to run the FlashPro v5.1 installer after installing FlashPro v6.0, FlashPro v6.0 will be uninstalled. FlashPro v5.1 will not work properly after installing FlashPro v6.0 – the FlashPro v6.0 installer will try to uninstall FlashPro v5.1 to prevent you from encountering this problem.
- Libero installations have links to their own installed FlashPro v5.1 components which will not work after installing FlashPro v6.0. You need to manually change your Libero profile settings to link Libero with FlashPro v6.0 (you are informed via a popup message before the installation of FlashPro v6.0).

- When installing/using an FP3 programmer for the first time, the Windows device wizard cannot always find the driver automatically. If you are asked for the location of the FlashPro3 driver, you can find it in <FlashPro software installation directory>/Drivers folder.
- When installing/using a FlashPro3 programmer for the first time, the Windows Device Wizard has to install the driver twice in a row before the FlashPro3 will work properly.

FlashPro3 Driver Installation Rules

Follow the instructions below for the FlashPro3 installation requirements.

Fixing Installation problems with FP3 Device Drivers

If Windows uses the wrong device driver, the FlashPro3 will not work. However, because Windows retains information about each device ever plugged in, it may not be as simple as uninstalling one software package and installing another. You may need to manually uninstall the wrong device driver. Below are the steps to manually uninstall the wrong device driver and indicate to Windows to use the correct FP3 driver.

Simple Method of Uninstalling the Driver

The easy method of uninstalling the wrong device driver is listed below.

To uninstall the wrong device driver (easy method):

1. Right-click on the My Computer icon on the PC desktop.
2. Select **Properties** to bring up the **System Properties** dialog box. Go to the **Hardware** tab.
3. Open the **Device Manager**.

If Windows is using the wrong device driver for the FlashPro3, the offending device driver will need to be removed. To do this, the device driver needs to be identified. The device driver could appear under:

1. The Human Interface Devices (HID) list: If you have multiple HID devices, you will need to check the driver properties and find one with Vendor ID (VID)=1514 and Product ID (PID)=2003.
2. The Universal Serial Bus (USB) controllers list: If there is an unknown USB device with a yellow ! or red X through it, check the driver properties and view the VID/PID to verify it is referring to the FlashPro3 (VID=1514 and PID=2003).
3. The Other Devices list: A yellow ! or red X will display indicating there is a problem communicating with this device.
4. Once you find the incorrect device, right-click on the device and select **Uninstall** from the drop-down menu. Click **OK** on the warning message.
5. Unplug the FP3 probe and wait, then plug the FP3 probe back in. If the correct drivers are installed on the PC when the FP3 is plugged back in, the **Found New Hardware Wizard** should appear and you will be able to direct it to the correct driver files.

Detailed Method of Uninstalling the Driver

If the above steps did not work, you will need to try a more detailed method involving editing the registry.

To uninstall the driver (Detailed method) for Windows NT:

1. Select **Start > Run** and type "regedit" in the **Run** dialog box.
2. In regedit, backup the registry so you can recover it if the registry somehow gets corrupt.
3. Select **File > Export**. Find a location and name for the registry file.
4. Go to **HKEY_LOCAL_MACHINE > SYSTEM > CurrentControlSet > Enum\USB** and look for the followings keys:



- Vid_1514&Pid_2003
 - Vid_1514&Pid_2004
 - Vid_1514&Pid_2005
5. Before you can delete the above keys, you will need to change the permissions. Right-click on the key and select **Permissions**.
 6. In the permissions area for the key, ensure that Everyone is selected to Full Control (see figure below).
 7. Click the **Advanced** button. The **Advanced** dialog box displays.
 8. In the **Advanced** dialog box, ensure that the "Inherit from parent the permission entries that apply to the child objects." Include these with entries explicitly defined here." Check box is checked.
 9. Delete the highlighted key.
 10. Repeat steps 5-7 for all keys listed above.

For Windows 2000:

1. Follow steps 1 and 2 above and save the registry.
2. Exit regedit.
3. Select Start->Run and type "regedt32."
4. Open HKEY_LOCAL_MACHINE>SYSTEM>CurrentControlSet>Enum\USB and then look for the followings keys:
 - Vid_1514&Pid_2003
 - Vid_1514&Pid_2004
 - Vid_1514&Pid_2005
5. Before you can delete these keys you will need to change the permissions. Go to **Security > Permissions**
6. Follow steps 5 and 6 above and check "Full Control" for everyone.
7. Highlight and delete all the keys listed above.

Note: Deleting only from the CurrentControlSet is all that is normally required to get back to a "First-Time install state". But, if for some reason that does not work on your PC, you may need to remove these keys from the ControlSet001 and ControlSet002 as well.

FlashPro Hardware Installation

This section details the hardware installation process for FlashPro.

To connect the FlashPro programmer to your PC using a parallel port:

1. Connect the programmer to a parallel printer port on your PC. Connect one end of the IEEE 1284 cable to the programmer's connector. Plug the other end of the cable into your parallel printer port and tighten the screws. You should not have any licensing dongles connected between the parallel port and cable. Your port settings should be EPP or bidirectional. Actel also supports the ECP mode with the FlashPro version 2.1 software and newer versions.
2. Verify that you are connected to the correct parallel port on your computer. Actel recommends that you dedicate a port to the programmer. Connecting to a serial port or a third party card may damage the programmer. This type of damage is not covered by the warranty.
3. Verify that the FlashPro power switch is in the 0 position.
4. Power up the programmer. Plug the DC adapter into a power socket. Plug the other end of the AC power supply to the DC-IN input at the back of the FlashPro.
5. Turn on the programmer. Turn the FlashPro power switch to the 1 position. The power LED on the front of the programmer lights up. If it does not, contact Actel technical support at (888) 99-ACTEL.

To connect a FlashPro programmer to your PC using a USB port:

1. Connect the programmer to a USB port on your PC. Connect one end of the USB cable to the programmer's USB connector. Plug the other end of the cable into your USB port.
2. From the "To connect the FlashPro to your PC:" instructions above, follow steps three through five.

Note: USB programming is slightly slower than programming through the parallel port for a single device. USB is hot-swappable, which means you do not have to power down the PC when plugging/unplugging the FlashPro programmer. Do not unplug the programmer while programming is active and performing a programming sequence.

FlashPro Lite Hardware Installation

This section details the hardware installation process for FlashPro Lite.

To connect the FlashPro Lite programmer to your PC:

1. Connect the programmer to a parallel printer port on your PC. Connect one end of the IEEE 1284 cable to the programmer's connector. Plug the other end of the cable into your parallel printer port and tighten the screws. You should not have any licensing dongles connected between the parallel port and cable. Your port settings must be EPP or bidirectional. Actel also supports the ECP mode with the FlashPro v2.1 software and newer.
2. Verify that you are connected to the correct parallel port on your computer. Actel recommends that you dedicate a port to the programmer. Connecting to a serial port or a third party card may damage the programmer. This type of damage is not covered by the warranty.
3. Connect the FlashPro ribbon cable with the programming header and turn the target board on.

Note: If you see two blinking LEDs on the programmer after you have connected the programmer to the parallel port, make sure the parallel port cable is connected firmly to the PC parallel port.

FlashPro3 Hardware Installation

This section details the hardware installation process for FlashPro3.

Before installing the FlashPro software, make sure your USB drivers are up to date.

To initially connect a single FlashPro3 to your PC using a USB port:

1. Insert the FlashPro3 CD into the CD-ROM drive
2. Connect one end of the USB cable to the programmer's USB connector.
3. Plug the other end of the cable into a USB port on your PC.
4. A Found New Hardware Wizard window displays. Follow the wizard steps.

Note: The fp3bload.sys file is on the FlashPro3 CD. The CyUSB.sys file is found in \Libero\FlashPro\Drivers.

5. You will observe the left amber LED initially flashes and then the amber Power LED will remain illuminated, indicating a connection to a powered USB port. If the LEDs remain flashing, make sure the USB cable is connected firmly to the programmer's USB connector.

USB 1.1 programming is slightly slower than programming through the parallel port for a single device, but using high-speed USB 2.0 programming is faster than a parallel port. By using a hub, multiple devices can be programmed at once, which saves time (see next section).

To connect multiple FlashPro3s using a powered USB hub:

1. Make sure your hub is a powered USB hub. Only powered USB hubs can be used.
2. Connect the USB hub to the PC. If the powered USB hub is not already connected, then follow the instructions for the hub setup and connect the cable from the hub to the PC.
3. Connect programmer to powered USB hub.



- Connect one end of the USB cable to the programmer's USB connector. Plug the other end of the cable into a free port on the USB hub. Follow steps 16 and 17 in [Software Installation](#).

After you have successfully installed the driver, you will observe the amber LEDs initially flashing and then the amber Power LED will remain illuminated indicating a connection to a powered USB port.

- Repeat step 3-4 for each programmer you wish to add.

Note: USB is hot-swappable, which means you do not have to power down the PC when plugging/unplugging the FlashPro3 programmer. Do not unplug the programmer while programmer is active and performing a programming sequence.

Starting Standalone FlashPro

You can start the FlashPro software from **Programs > Actel FlashPro vx.x >**

FlashPro. If you installed the program in a folder other than FlashPro, choose that folder from the **Programs** menu.

The figure below shows the FlashPro GUI. From this GUI, you can create a new project by clicking the **New Project** button or open an existing project by clicking the **Open Project** button.

You can also access the above features from the menu bar. You can access all the other features after you [open](#) or [create](#) a new project.

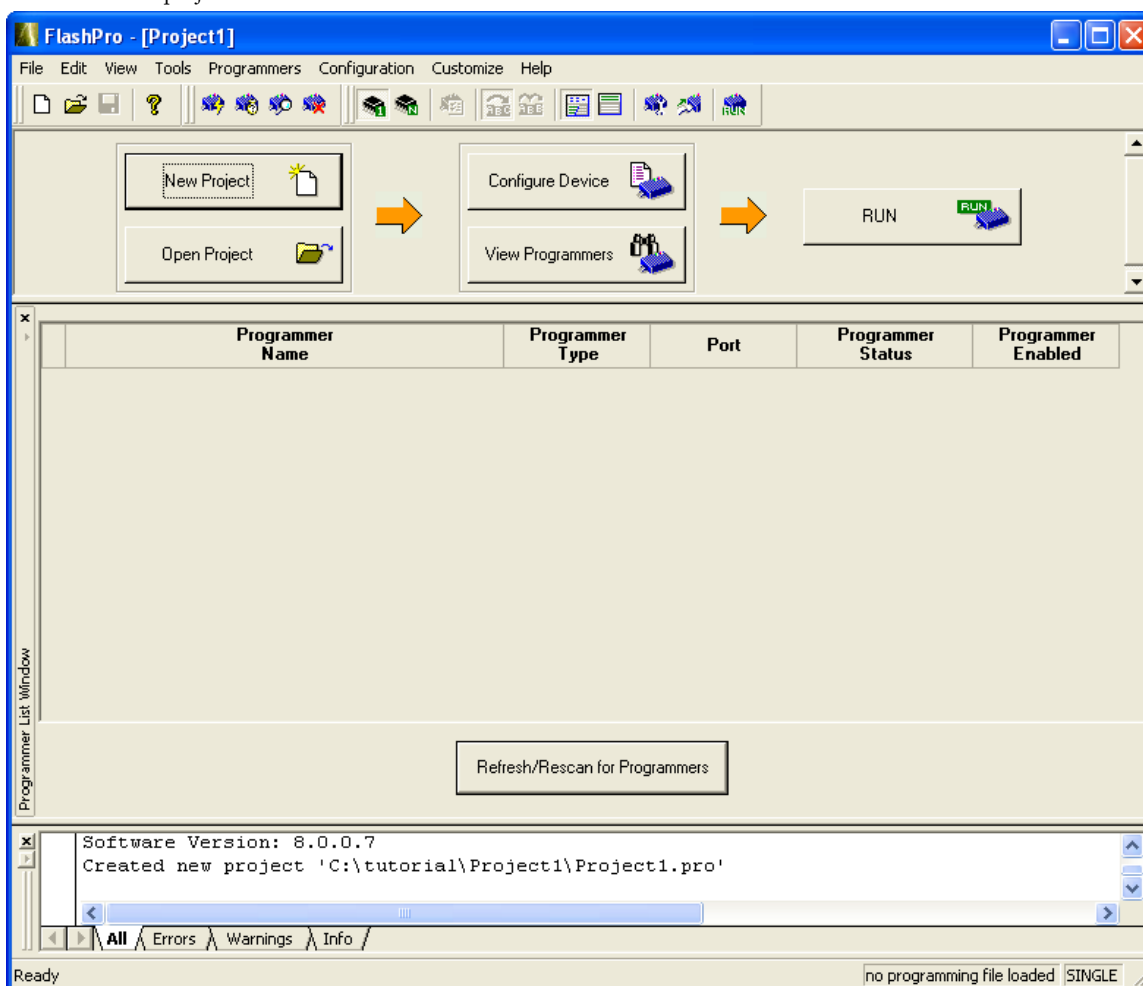


Figure 15 · FlashPro (Single Device Mode)

Creating a New Project

With the FlashPro software, you have the option of choosing either the **Single STAPL file** or **Chain programming mode**. You make this choice through the **New Project** dialog box (see figure below). By choosing the **Chain Programming mode**, you are enabling chain programming. The **Single STAPL file Programming mode** functions with the same programming capabilities as the FlashPro software v4.2.

To create a new project:

1. Click the **New Project** button or choose **New Project** from the **File** menu.
2. From the **New Project** dialog box, type in the name of your project in the **Project Name** field.



Figure 16 · New Project Dialog Box

3. If necessary, change the default location of your project in the **Project Location** field.
4. Choose your **Programming mode** (Single device or Chain).
5. Click **OK**. The FlashPro GUI displays (see figure below).

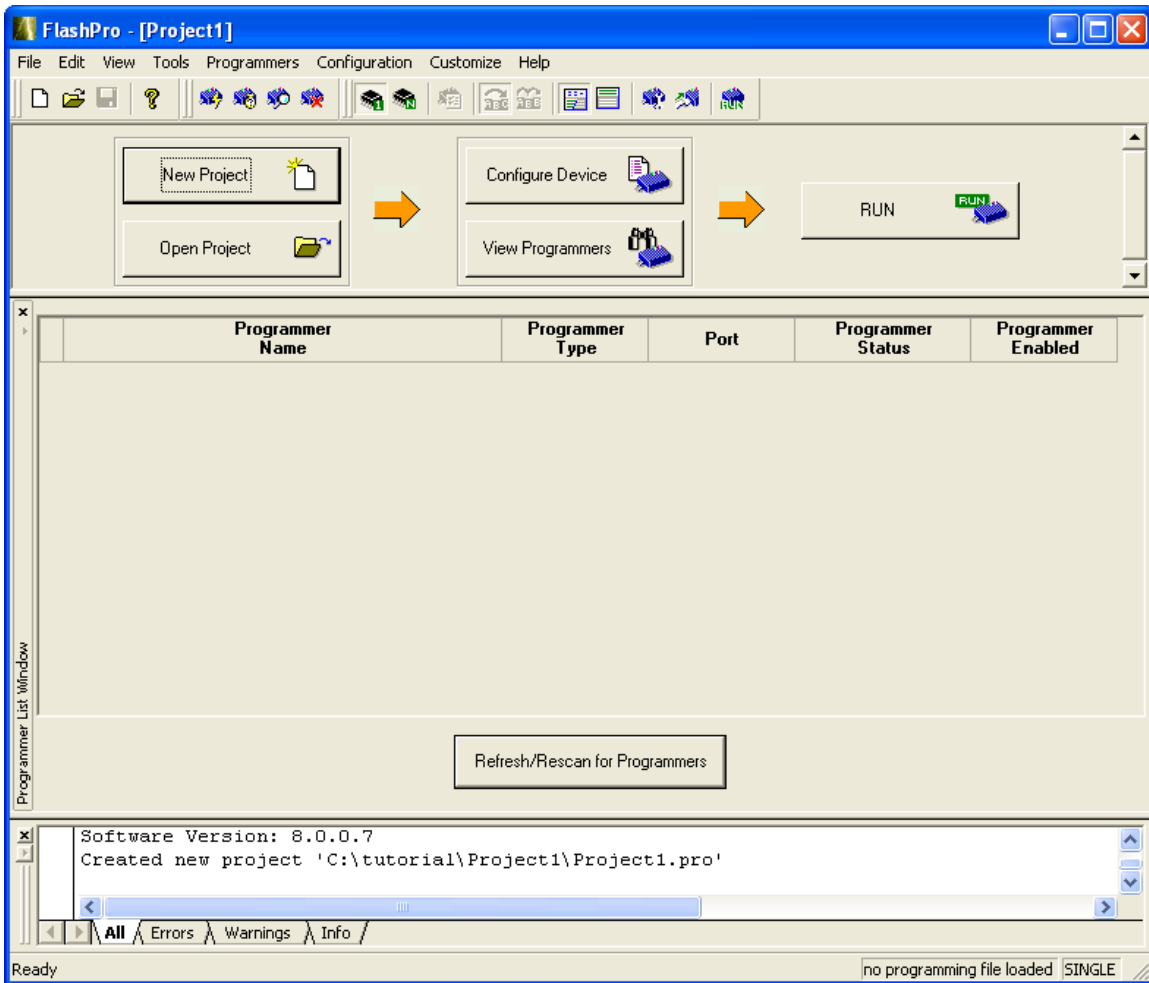


Figure 17 · FlashPro GUI

Note: You can switch between the two programming modes from Tools > Mode. From there, you can choose either Single Device Programming or Chain Programming.

Opening a Project

You can open a project from the **File** menu or by clicking on the **Open Project** button in the [flow window](#) or [toolbar](#).

To open a project:

1. From the **File** menu, choose **Open Project**. The **Open Project** dialog box displays.

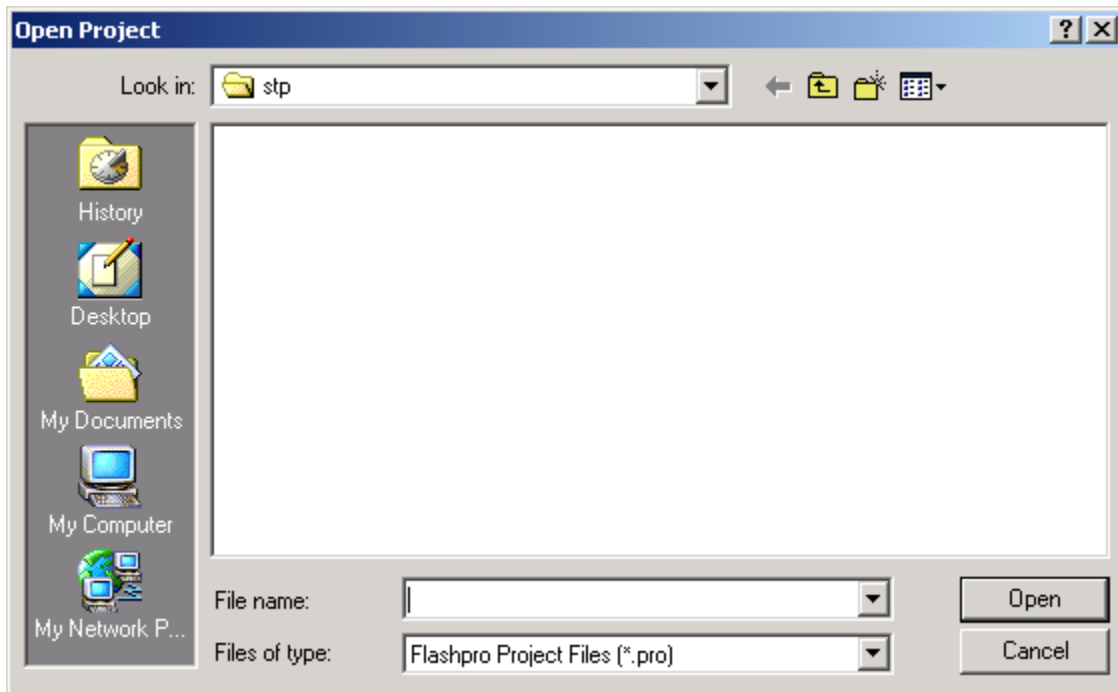


Figure 18 · Open Project Dialog Box

2. Find your project file or type in your project file name in the **File name** text box.
3. Click **Open**.

Saving a Project

Click the **Save** button on the toolbar or choose **Save Project** from the **File** menu to save your project.

If you want to save your project under a different name/path, choose **Save Project As** from the **File** menu and save your project with the new name.

Understanding Parallel Programming

Parallel programming enables you to program multiple Actel devices in parallel with multiple programmers. In parallel programming, all targeted devices are programmed with the same programming file (STAPL). The targeted device or chain configuration that is connected to each programmer must be identical.

Note: Parallel programming via FlashPro (USB/LPT1) or FlashPro Lite programmers is not supported.

The FlashPro software together with the FlashPro3 programmers supports parallel programming via a USB port. You can connect up to 16 FlashPro3s to a PC via a USB v1.1 or a USB v2.0 port. FlashPro3 requires a self-powered hub.

Note: Connecting FlashPro3 (a USB v2.0 enabled programmer) to USB v1.1 port increases device programming time due to a slow data transfer rate on the USB v1.1 port in comparison to a USB v2.0 port.

The following figure illustrates how you can connect a FlashPro3 programmer for parallel programming.

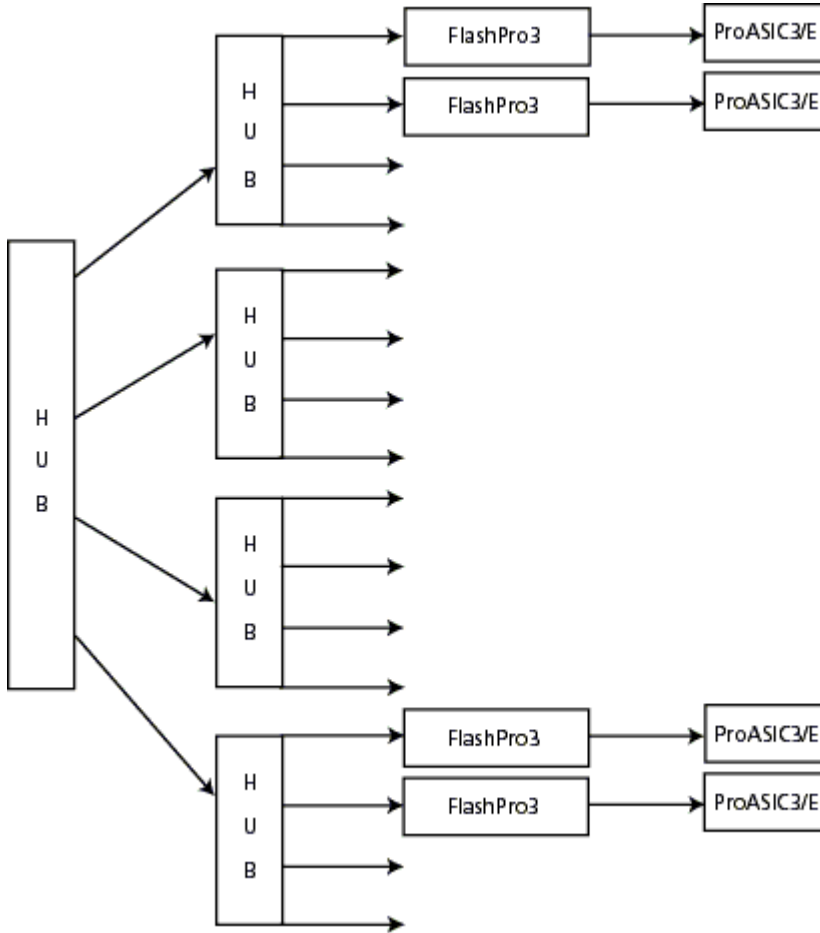


Figure 19 · Connecting a FlashPro3 Programmer

An independent thread processes the STAPL file during parallel programming. In an Actel test, parallel programming is approximately five times faster than programming 16 devices sequentially.

Note: Actel has tested Belkin, Adaptec, and D-Link PCI-USB cards and hubs. We have found that parallel programming works best with the vendor's latest driver installed and with the matching hubs.

Understanding Serialization

You can use the FlashROM in the ProASIC3 device for serialization. For each target ProASIC3 device, different FlashROM contents are generated.

Serial Programming enables you to program a sequence of ProASIC3/E devices in serial with an identical FPGA program and with different serialization data. Serialization data can consist of different FlashROM content and/or AES key values. To learn how to activate the serialization feature, see [Skip Serial Data](#) or [Reuse Serial Data](#).

There are two different STAPL formats that support serial programming, multiple actions to multiple serial data and single action to multiple FlashROM.

Multiple Actions to Multiple FlashROM Serial Data

This format supports a generic STAPL player because the STAPL player does not provide a mechanism for Serial Programming. One programming action is created to target different serial data. See examples below:

- PROGRAM_1 programs the FPGA Array and the first serial data.
- PROGRAM_2 programs the FPGA Array and the second serial data.

Single Action to Multiple FlashROM Serial Data

This format is created when the target programmer is FlashPro, Sculptor II, or BP auto programmer, where the newly innovated Actel Serial Programming mechanism is supported. One programming action will program multiple serial data in serial.

Understanding SVF

SVF (Serial Vector Format) is an industry standard file format that is used to describe JTAG operations. Like STAPL files, SVF files are used for describing the in-system programming algorithm for IGLOO, Fusion, and ProASIC3 family devices. Unlike STAPL files, SVF files support only one ACTION or programming flow per file, due to language limitations. In addition, the SVF specification does not support message display and flow control, such as conditional statements or loops.

As a result, Actel tools (Designer and FlashPro software) generate a set of SVF files corresponding to the equivalent STAPL ACTIONS that are applicable to the silicon features selected.

For example, for a typical STAPL file that has the following ACTIONS: ERASE, ERASE_ALL, PROGRAM, PROGRAM_ARRAY, VERIFY, VERIFY_ARRAY, DEVICE_INFO, READ_IDCODE, and VERIFY_DEVICE_INFO, a set of corresponding SVF files are generated and named: ERASE.svf, ERASE_ALL.svf, PROGRAM.svf, PROGRAM_ARRAY.svf, etc. These files are generated in a folder, <Programming File Name>_svf, created during generation. The diagram below demonstrates the differences between the STAPL and SVF files that are created.

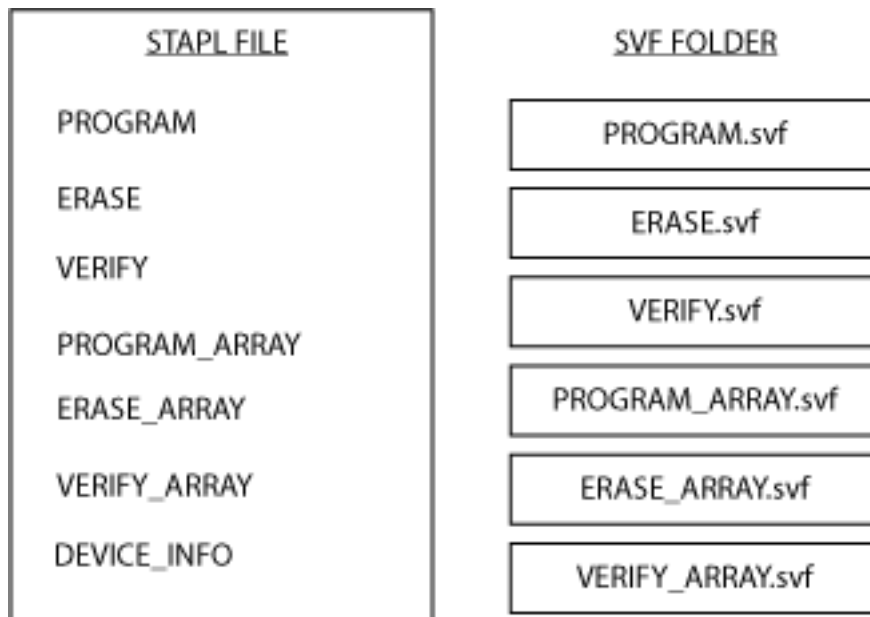


Figure 20 · STAPL vs SVF files

NOTE: DEVICE_INFO.svf file is not generated because SVF files do not support message display or flow control.

Table 1 · SVF Outline

SVF File	Array	FROM	NVM (Flash Memory System Builder)	Security Settings	Previously Programmed Device?
ERASE	X	X			YES or NO
ERASE_ALL	X	X	X	X	YES or NO
ERASE_ARRAY	X				YES or NO
ERASE_FROM		X			YES or NO
ERASE_SECURITY				X	YES or NO
PROGRAM	X	X			YES or NO
PROGRAM_ARRAY	X				YES or NO
PROGRAM_FROM		X			YES or NO
PROGRAM_NVM			X		YES or NO
PROGRAM_SECURITY				X	YES or NO
VERIFY	X	X	X		YES or NO
VERIFY_ARRAY	X				YES or NO
VERIFY_FROM		X			YES or NO
VERIFY_NVM			X		NO
ENC_DATA_AUTHENTICATION	X				YES

STAPL Actions not Available with SVF

The following STAPL actions are not available with SVF: DEVICE_INFO, VERIFY_DEVICE_INFO, READ_IDCODE

Understanding 1532 File Format

1532 is an IEEE industry standard file format that is used to describe JTAG operations. Like STAPL files, 1532 files are used for describing the in-system programming algorithm for IGLOO, Fusion, and ProASIC3 family devices.

1532 programming file generation will generate two files (*.isc, *.bsd) within a folder.

The folder will be created with the following name <Programming File Name>_1532. The *.bsd file contains the IEEE 1532 programming algorithm. The *.isc file contains the programming data to be programmed into the device.

STAPL to 1532 Action Mapping

The IEEE 1532 standard requires using default ACTION names in order to function with 1532 compliant players. The table below describes the STAPL to 1532 ACTION name mappings.

NOTE: 1532 ACTIONs can have a data member parameter to allow reuse of the same ACTION name for different features.

Table 2 · STAPL to 1532 Action Name Mapping

STAPL Action	1532 Action
ERASE_FROM	ERASE(FROM)
PROGRAM_FROM	PROGRAM(FROM)
VERIFY_FROM	VERIFY(FROM)
PROGRAM	PROGRAM
PROGRAM_ARRAY	PROGRAM(ARRAY)
ERASE_ARRAY	ERASE(ARRAY)
ERASE	ERASE
ERASE_ALL	ERASE(ALLDATA)
VERIFY	VERIFY
VERIFY_ARRAY	VERIFY(ARRAY)
READ_IDCODE	READ(IDCODE)
ENC_DATA_AUTHENTICATION	VERIFY(ENCDATA)
PROGRAM_SECURITY	PROGRAM(SEcurity)
DEVICE_INFO	READ
VERIFY_NVM	VERIFY_NVM
VERIFY_SECURITY	VERIFY(SEcurity)
PROGRAM_NVM	PROGRAM_NVM

STAPL Actions not Available with 1532

The following STAPL action is not available with 1532: VERIFY_DEVICE_INFO



Creating a Programming Database (PDB) File in Designer

The programming database (PDB) file supports IGLOO, Fusion, and ProASIC3, devices only. This allows reconfiguration of the security settings, FlashROM, FPGA Array, and Embedded Flash Memory Blocks. You create the file in Designer using FlashPoint and you modify the file in FlashPro.

1. From the Designer main window, click the **Programming File** button. This brings up FlashPoint (see figure below).

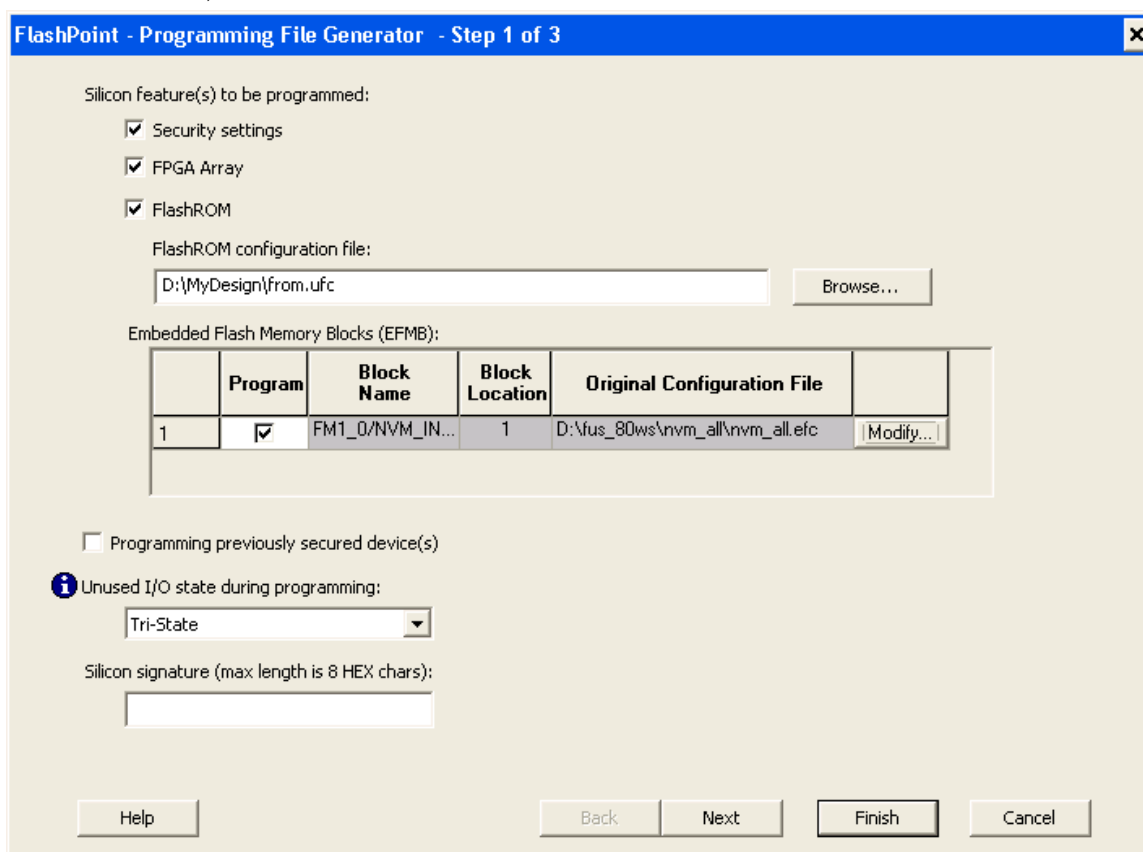


Figure 21 · FlashPoint Programming File Generator - PDB File

2. Select the [silicon feature\(s\) to be programmed](#): [Security Settings](#), [FPGA array](#), [FlashROM](#), and [Embedded Flash Memory Block](#). If you are programming a previously secured device, check the [Programming previously secured device\(s\)](#) and enter the [silicon signature](#).
3. Click **Finish** to create the PDB file.

See Also

Configuring security, FPGA Array and FlashROM settings in FlashPro

Configuring security settings in FlashPro

Configuring FPGA array settings in FlashPro

Configuring FlashROM settings in FlashPro

Configuring Embedded Flash Memory Block settings in FlashPro

Understanding the FlashPro GUI

The main FlashPro consists of two views, one for Single Device Programming and the other for Chain Programming, see figure below. Each view consists of a [menu bar](#), [toolbar](#), and a [flow window](#). The GUI also consists of a [Log](#) window and a [Status](#) bar. The [Log](#) window displays programming information, error messages, and warning messages. The [Status](#) bar displays your programming mode (chain programming or single device programming) and file status.

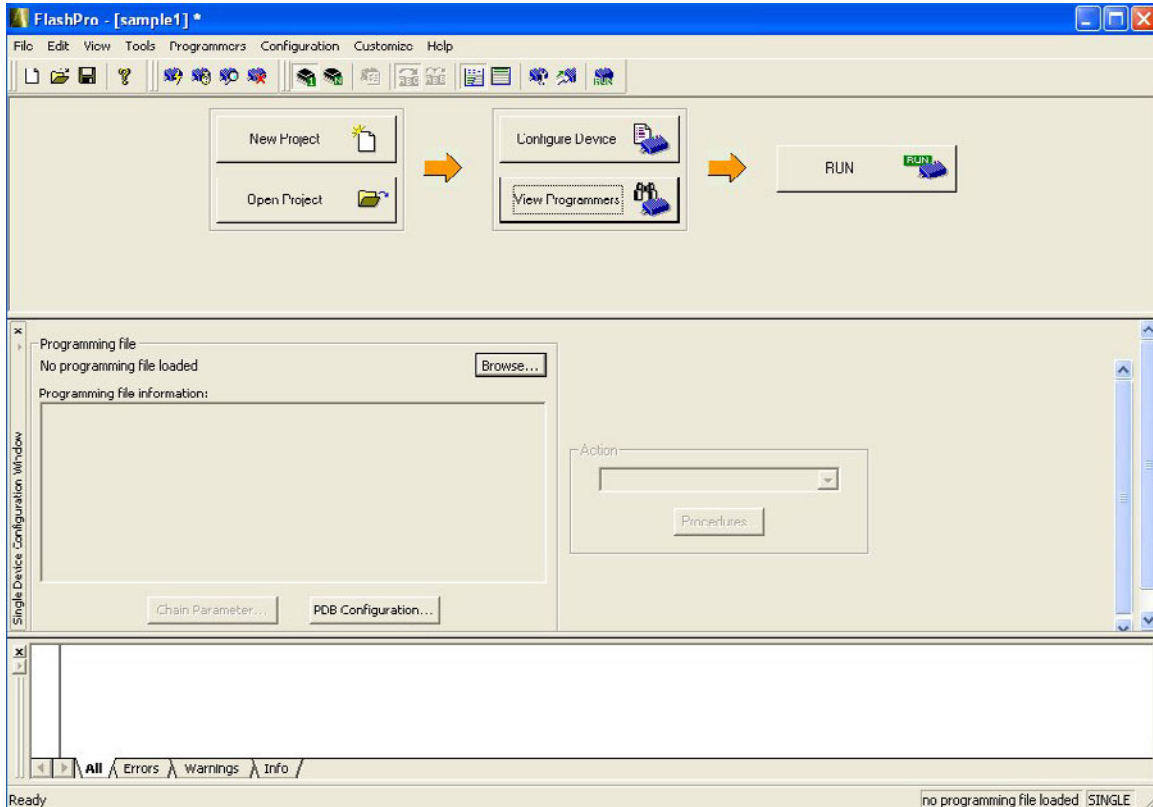


Figure 22 · FlashPro GUI for Single Device File Programming

Note the different options in the **Flow** window for the **Chain Programming** GUI and the **Single Device Programming** GUI. In addition to the different **Flow** window options, the **Chain Programming** GUI view consists of the **Chain Configuration** window which displays the devices in your chain.

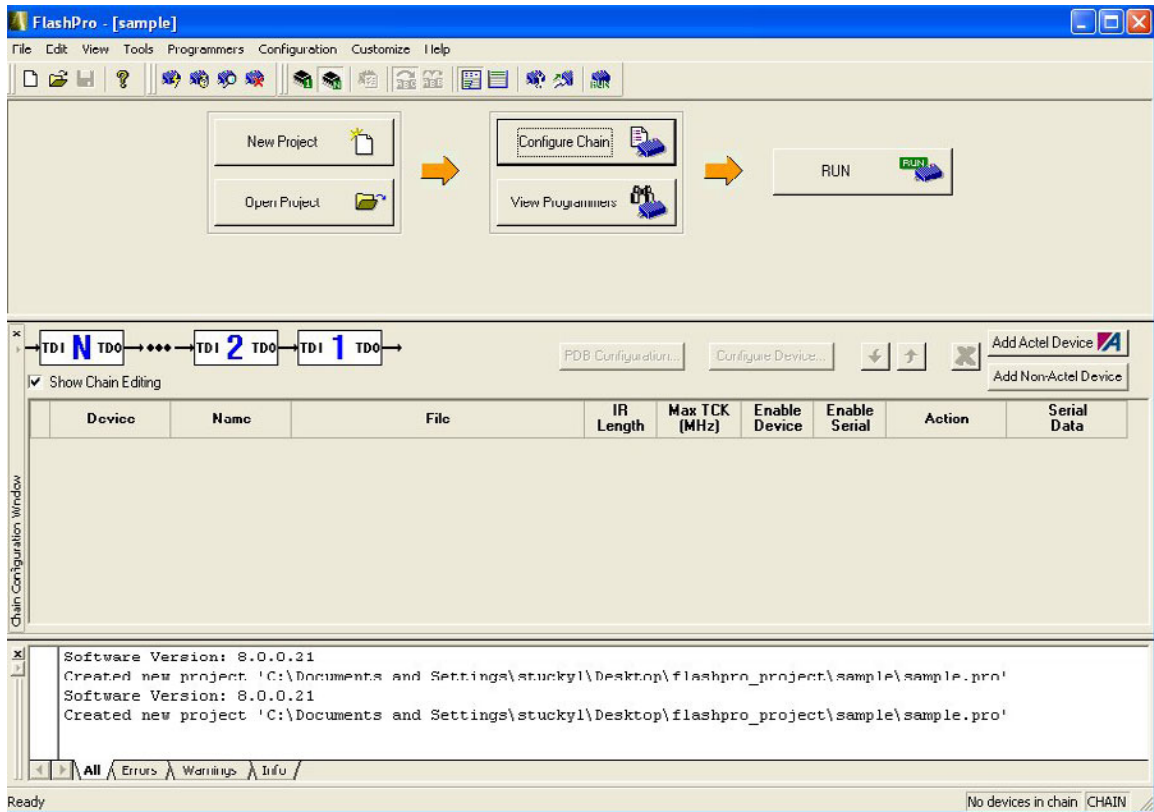


Figure 23 · FlashPro GUI for Chain Programming

For a description of the items in the menu bar, toolbar and Flow window see: [Understanding the menu bar](#), [Understanding the toolbar](#), and [Understanding the flow window](#).

The FlashPro GUI also includes the following windows: the **Programmer List** window, the **Programmer Details** window, **Chain Programming** window and the **Single Device Configuration** window. For more information about these windows and how to access them, see [Understanding the programmer list window](#), [Understanding the programming detail window](#), and [Understanding the single device configuration](#) window.

Understanding the Toolbar


The figure below shows the buttons in the FlashPro toolbar and the table below lists the toolbar buttons and their descriptions.



Figure 24 · Toolbar

Table 3 · Toolbar Button Description

Toolbar Name	Button	Description
New Project		Creates a new project. Opens the New Project dialog box.
Open Project		Opens a new project. Opens the Open Project dialog box.
Save		Saves a project.
About		Gives a short description of the FlashPro software.
Ping Programmer		Opens the Select Programmer(s) dialog box for you to select the programmers you want to ping.
Self-test Programmer(s)		Opens the Select Programmer(s) dialog box for you to select the programmers you want to self-test.
Scan Chain on Programmer(s)		Opens the Select Programmer(s) dialog box for you to select the programmers you want to scan.
Remove Programmer		Opens the Select Programmer(s) dialog box for you select the programmers you want to remove.
Single Device Programming		Enables single device programming.
Chain Programming		Enables Chain programming.
Select Action		Select an action to be performed by the device.
Skip Serial Data		Select to skip serial data when failed.
Reuse Serial Data		Select to reuse serial data when failed.
Classic Display Mode		Displays one window view at a time. You must switch between the Programmer List Window and the Device Configuration Window . Use this mode for multiple programming runs when frequent device setting changes are not necessary.
Advanced Display Mode		Displays the Programmer List Window and the Device Configuration Window in the same GUI. Use this mode if you need to change device settings between programming runs.
Programmer Settings		Select to changes programmer settings.
Connect Programmers from a parallel port cables		Displays the Connect Parallel Port Cable dialog box.

Toolbar Name	Button	Description
Run		Select to execute programming.

Understanding the Menu Bar

The menu bar consists of the following menus: **File**, **Edit**, **View**, **Tools**, **Programmers**, **Configuration**, **Customize**, and **Help** as shown in the figure below.

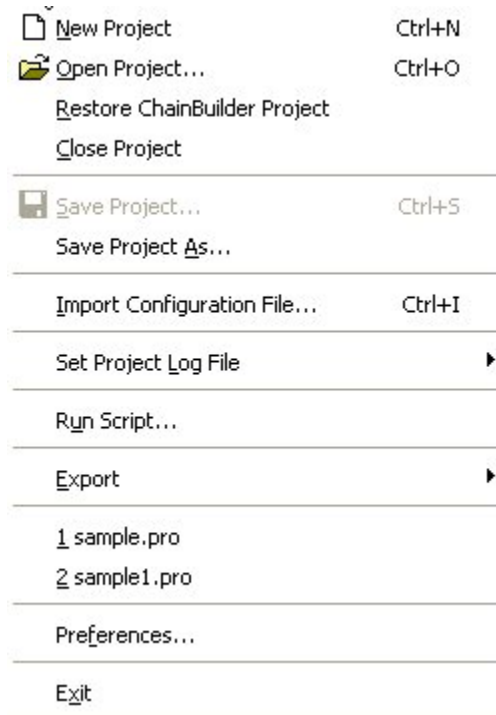
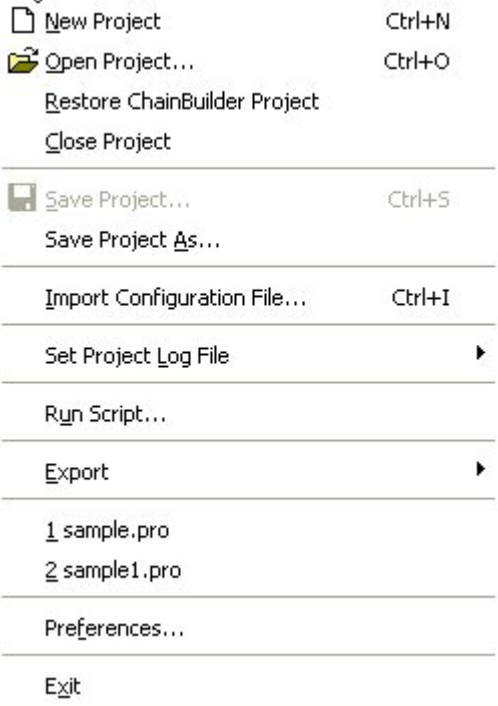




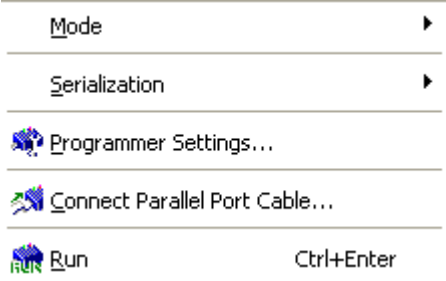
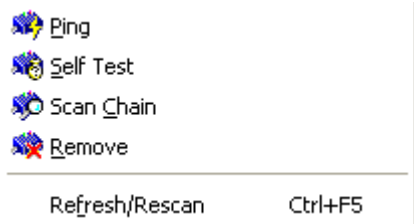


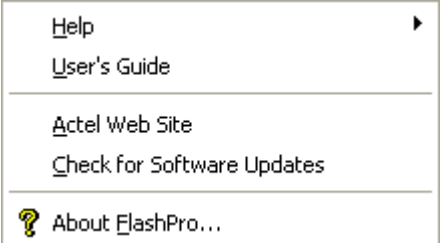
Figure 25 · Menu Bar

The table below describes each menu in the menu bar. In the **Chain Programming** mode, the **Edit** menu and the **Configuration** menu changes. The table notes these changes.



Table 4 · Menu Description

Menu Name	Menu	Description
File	 <p>The screenshot shows the File menu with the following items: New Project (Ctrl+N), Open Project... (Ctrl+O), Restore ChainBuilder Project, Close Project, Save Project... (Ctrl+S), Save Project As..., Import Configuration File... (Ctrl+I), Set Project Log File (with a right-pointing arrow), Run Script..., Export (with a right-pointing arrow), a list of project files (1 sample.pro, 2 sample1.pro), Preferences..., and Exit.</p>	<p>Use the File menu to create, save, or open a project. You can also import configuration files, set the project log file location, run scripts, and set preferences from this menu.</p>
Edit	 <p>The screenshot shows the Edit menu with the following items: Cut Devices (Ctrl+Shift+X), Copy Devices (Ctrl+Shift+C), Paste Devices (Ctrl+Shift+V), and Clear Log Window.</p>	<p>From the Edit menu, you can activate the cut, copy, and paste options; and you can clear the Log window</p>
View	 <p>The screenshot shows the View menu with the following items: Status Bar (checked), Programmer List Window, Programmer Details Window, Single Device Configuration Window (checked), Log Window (checked), Single Device Configuration (with a right-pointing arrow), and Refresh Views (F5).</p>	<p>From the View menu, you can customize the toolbar and view the following windows: Status bar, Programmer List, Programmer Details, Single Device Configuration, Log, and Refresh Views.</p>

Menu Name	Menu	Description
Tools		<p>The Tools menu includes the Serialization feature for skipping serial data or reusing serial data. You can also change your Programmer Settings from this menu.</p>
Programmers		<p>The Programmers menu includes the following programming features: Ping, Self-test, Scan Chain, Remove, and Refresh/Rescan for programmers.</p>
Configuration		<p>From the Configuration menu, you can select actions, use the serialization feature, load and unload your programming file, configure the PDB, select chain parameter, and select your target device. The menu options are different for Chain and Single Device programming.</p>
Customize		<p>You can customize the toolbar from this menu.</p>
Help		<p>The Help menu list help topics, contains the FlashPro User's Guide, a link to the Actel web site, a check for software updates and provides general information about the software.</p>



Understanding the Flow Window

The **Flow** window (located between the toolbar and Log window in the FlashPro GUI) consists of the following buttons: [New Project](#), [Open Project](#), [Configure Device](#), [View Programmers](#), and [Run](#). See the table below for a description of these features.

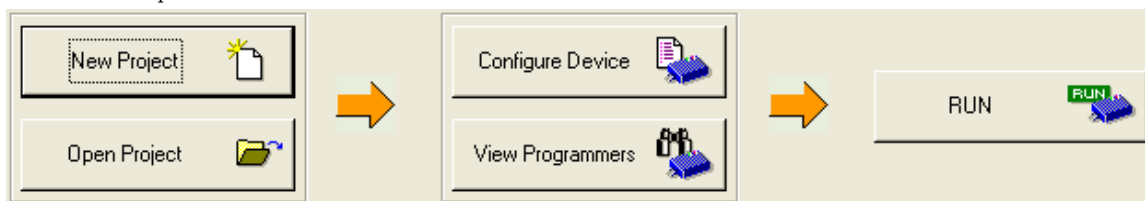


Figure 26 · Flow Window

Table 5 · Flow Window Button Description

Button	Description
New Project	Creates a new project. Opens the New Project dialog box.
Open Project	Opens a new project. Opens the Open Project dialog box.
Configure Device	Opens the Single Device Configuration window to configure your file.
View Programmers	Opens the Programmer List Window for you to view your programmers.
Refresh/Rescan for Programmers	Rescans for programmers.
Run	Executes programming.

Understanding the Log Window

The **Log** window displays errors, warnings, and basic information about your device. Click the tabs at bottom of the **Log** window to toggle between messages or click the **All** tab to display all of the messages.

You can access the **Log** window from the **View** menu.

Setting Log window preferences

From the **Preferences** dialog box, you can change the text color of the messages that appear in the **Log** window.

To set Log window preferences:

1. From the **File** menu, choose **Preferences**.
2. Follow the directions in the **Color Settings** area or click the **Restore Defaults** button.

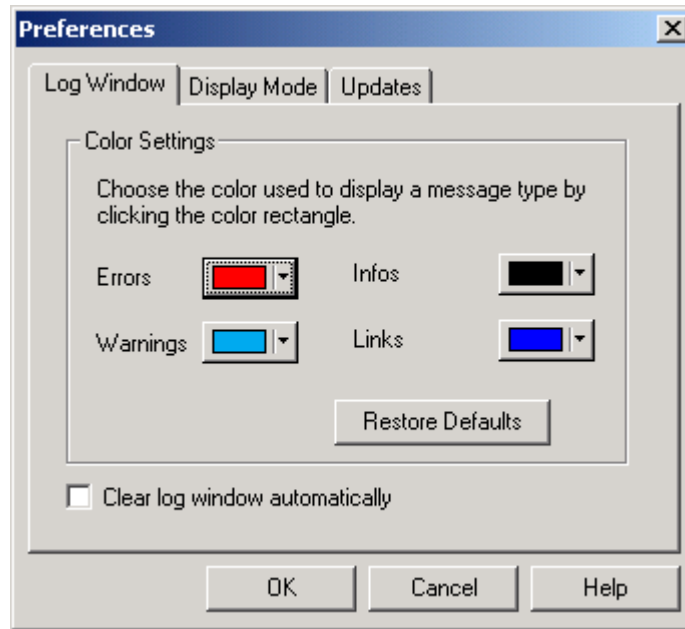


Figure 27 · Preferences

3. Click **OK** to apply settings.

Note: You can clear the Log window by choosing Clear Log window from the Edit menu or you can automatically erase the information by checking the Clear Log Window Automatically checkbox.

Understanding the Status Bar

The **Status** bar displays the program status, the programmer information (the file name for single device programming and number of devices for chain programming), and the programming mode (single or chain).

Understanding the Programmer List Window

To activate the **Programmer List** Window, select **View > Programmer**. The FlashPro **Programmer List** Window consists of a spreadsheet with the programmer name, programmer type, port number, programmer status, programmer enable check box, and a **Refresh/Rescan for New Programmers** button as shown in the figure below.

Note: Double-clicking any of the spreadsheet columns opens the [Programmer Details window](#).



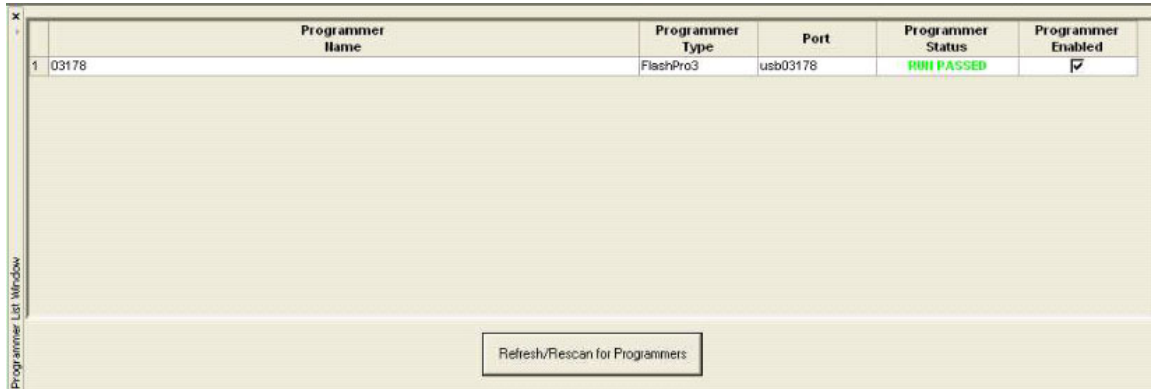


Figure 28 · Programmer List Window

Changing the Name of your Programmer

You can change the name of your programmer by double-clicking in the spreadsheet cell or you can choose **Edit Cell** from the right-click menu.

Connecting New Programmers

You can connect new programmers by clicking the **Refresh/Rescan for Programmers** button.

Accessing Right-Click Menus

If you have checked the **Programmer Enabled** checkbox, you can right-click on any of the spreadsheet fields to access the menu in the figure below.

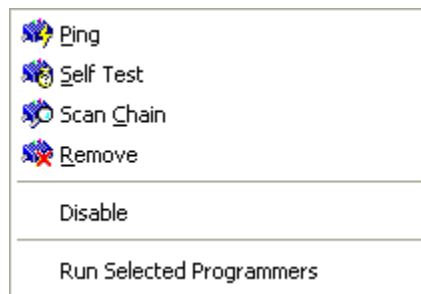


Figure 29 · Right-click Menu

If you have not checked the **Programmer Enabled** checkbox, you can right-click in the on any of the spreadsheet fields, to access a menu to remove or enable the programmer (see figure below).

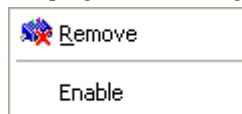


Figure 30 · Right-click Menu

Understanding the Programmer Details Window

The **Programmer Details Window** displays information about your programmer (see figure below).

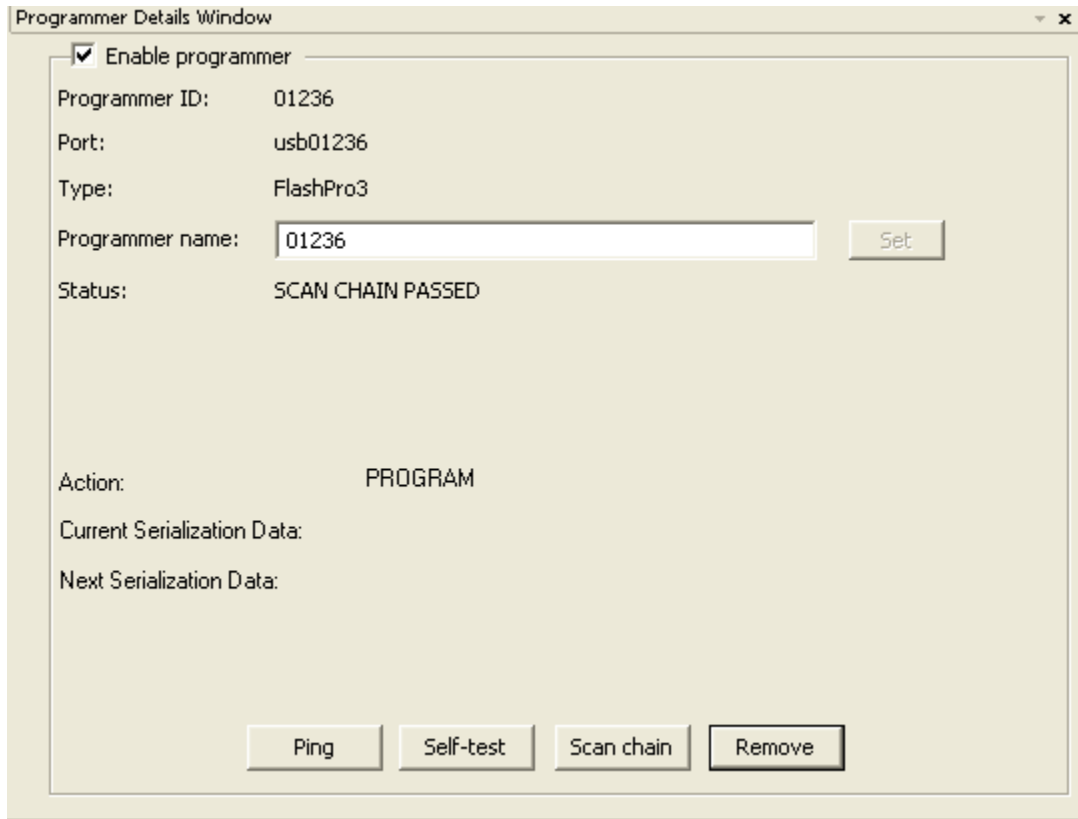


Figure 31 · Programmer Details Window

You can access this window from the **View** menu or you can double click on any of the fields in the **Programmer List Window** (Programmer Name, Programmer Type, Port, Programmer Status, and Programmer Enabled).

Click the **Enable Programmer** checkbox to enable your programmer and activate the **Programmer Details Window**.

From the **Programmer Details Window**, you can [ping a programmer](#), [perform a self-test](#), [scan a programmer](#), or [remove a programmer](#).

Understanding the Single Device Configuration Window

To access the **Single Device Configuration Window** simply click on the **Configure Device** button from the **Flow window**. The **Single Device Configuration window** displays PDB/STAPL file and serialization information (see figure below). You can also deactivate serialization by clicking the **Serialization** checkbox.



Figure 32 · Single Device Configuration Window

Loading the PDB/STAPL File

You can load your PDB/STAPL file from the **Configuration** menu by choosing **Load Programming File** or by clicking the **Browse** button in the **Single Device Configuration Window**.

You can set chain parameter settings by clicking the **Chain Parameter** button.

Selecting Serialization Indexes

To select the serialization indexes:

1. Check the **Serialization** checkbox to activate serialization.
2. Click the **Select Serialization Indexes** button.

The **Serial Settings** dialog box displays (see figure below).

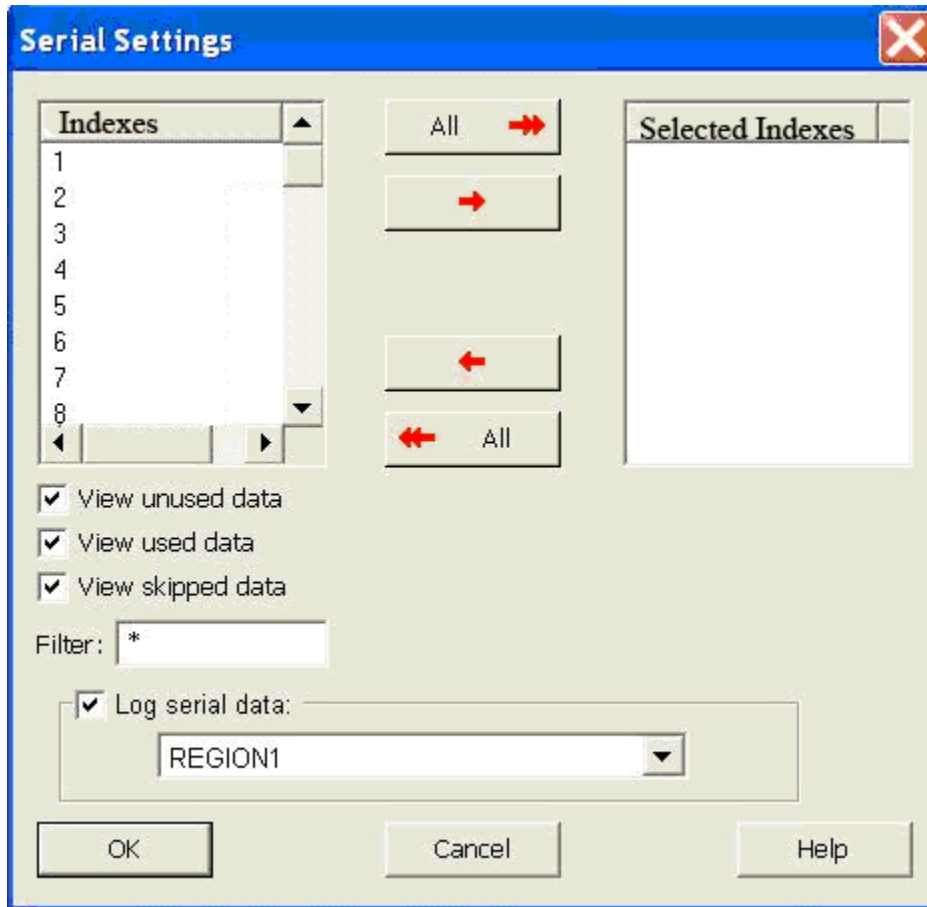


Figure 33 · Serial Settings Dialog Box

3. Select an index in the **Indexes** column by clicking on it.
4. Click on the red arrow button that is pointing toward the **Selected Indexes** column to move an index to the **Selected Indexes** column.
5. If you want to move all the **Indexes** to the **Selected Indexes** column, click the **All** button.
6. Click **OK**. Information about your serial indexes displays.
You can move indexes from the **Selected Indexes** column to the **Indexes** column by clicking the red arrow buttons pointing toward the **Indexes** column.
You can set your indexes by choosing the following check boxes: **View unused data**, **View used data**, and **View skipped data**.

Selecting Action and Procedures

You can select actions from the Basic mode and the Advanced mode. The Basic mode is provided for users that only require the Program, Verify and Erase actions. In Basic mode, other actions are not visible, and the Procedures run by an Action cannot be modified.

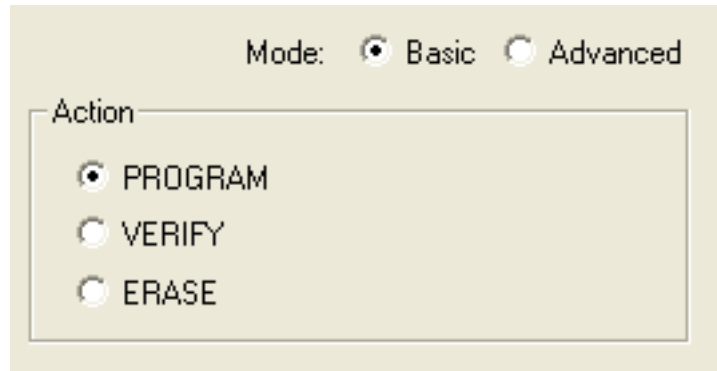


Figure 34 · Basic Mode

The Advanced mode allows users to select an action and modify the procedures for the selected action. In Advanced mode, actions are determined by the stapl standard used.

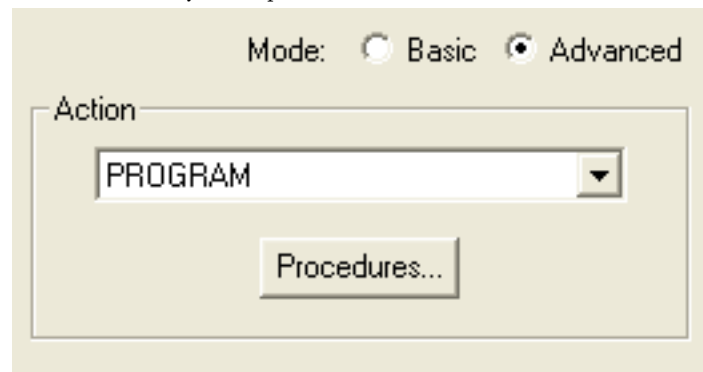


Figure 35 · Advanced Mode

To select an action (Advanced Mode):

1. Click the down arrow in the **Action** menu and select an action (see figure below).

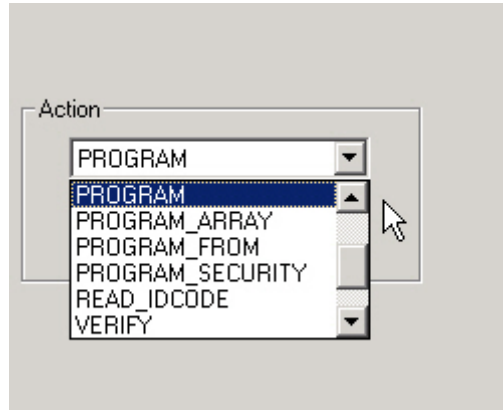


Figure 36 · Action Menu

2. Click the Procedures button.
The Select Action and Procedures dialog box displays.

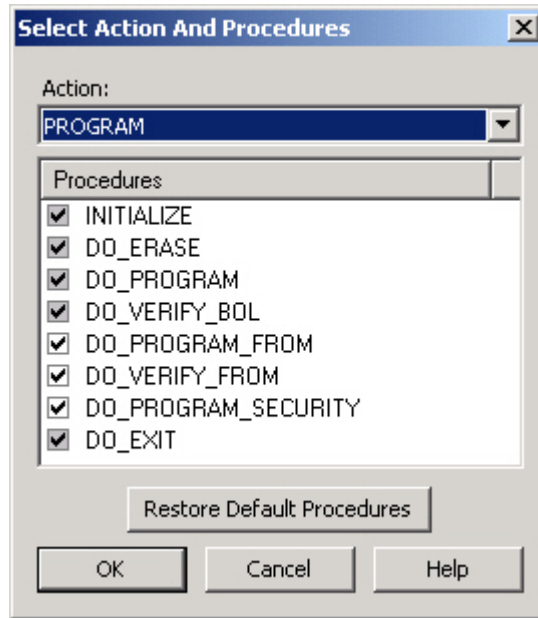


Figure 37 · Select Action And Procedures Dialog Box

3. Select a procedure or click the **Restore Default Procedures** button. Grey checkboxes indicate that the procedure is mandatory.

Note: The procedures in the Select Action And Procedures dialog box are determined by the STAPL standard. The Recommended procedures are selected by default and the Optional procedures are unselected by default.

4. Click **OK**.

Note: You can also click the Select Action and Procedures dialog box from the toolbar.



Understanding the Chain Configuration Window

The **Chain Configuration Window** displays the chain order, the chain editing options, and the chain configuration grid (see figure below).

The **Show Chain Editing** checkbox, when checked displays your chain editing options (Configure device, Add Actel Device, Add Non-Actel Device, and organization buttons to move your device within the grid).

Note: For information on how to add Actel and Non-Actel devices, see [Chain Editing](#).

Note: For information on how to use the Organize buttons (located next to the Add Actel and Add Non-Actel buttons) in the Chain Configuration grid, see [Using the Organize buttons in the Chain Programming grid](#).

You can enable programming and serialization by checking the **Enable Device** checkbox and the **Enable Serial** checkbox in the Chain Configuration grid.

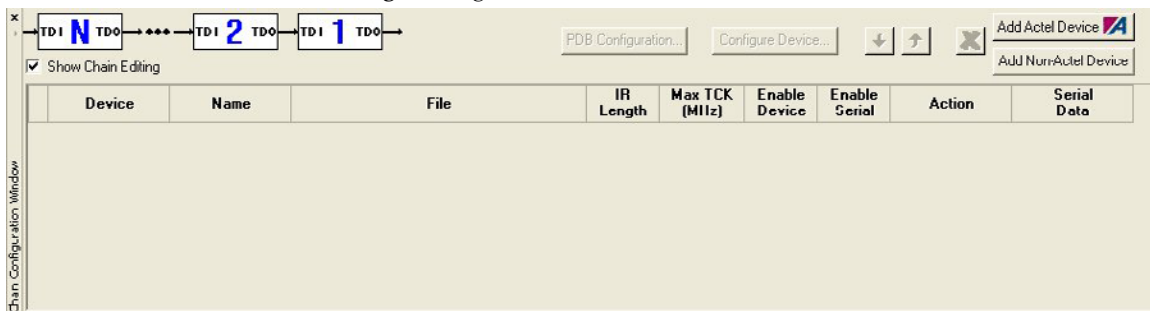


Figure 38 · Chain Configuration Window

Chain Editing Options

The FlashPro software enables you to edit your chain by adding Actel and Non-Actel devices. You can add devices by clicking the **Add Actel Device** button and the **Add Non-Actel Device** button or you can select these options from the **Configuration** menu.

Note: For more information about how to edit the chain, see [Chain Editing](#).

Editing the Chain Configuration Grid

The **Chain Configuration Grid** enables you to select an **Action** for your device, **Enable Serialization**, and edit the grid using the right-click menu.

To select an Action from the Configuration Grid:

1. Choose the device you would like to program and check the **Enable Device** checkbox.
2. In the **Action** column, click the down arrow to expose the drop-down menu (see figure below).
3. Select your desired action.

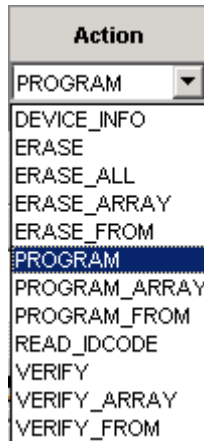


Figure 39 · Drop-Down Menu for Select Action

To enable Serialization:

1. Check the **Enable Serial** checkbox. By enabling serialization, the action options change.
Note: Before you can enable serialization, you must check the **Enable Device** checkbox.
2. In the **Action** column, click the down arrow to expose the drop-down menu (see figure below).

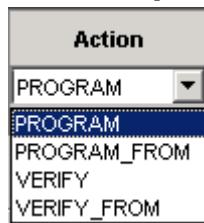


Figure 40 · Drop-Down Menu for Select Action

3. Select your desired action.
4. Choose **Select** button from the **Serial Data** column, which is next to the **Action** column (see figure below).



Figure 41 · Serial Data Column

The **Serial Settings** dialog box displays.

5. Choose your serial settings from the **Serial Settings** dialog box.
See [Serial Settings](#) for more information about this topic.
Note: Uncheck the **Enable Serial** checkbox to disable serialization.

To edit the Chain Configuration Grid:

1. Select the device you would like to edit and right click anywhere in the row of the selected device. The right-click menu below displays.
2. Select and click an option from the right-click menu.
Note: The **Device Configuration** menu includes options for configuring your device.



Customizing the Toolbar

Display the tools and commands you frequently use in the toolbar by customizing it.

To customize the toolbar:

1. From the **Customize** menu, select **Toolbars**. The **Customize** dialog displays.
2. Click the **Toolbar** tab and check the tools you want to display by checking their respective boxes, (see figure below).

Note: You can remove tools from your toolbar by deselecting tools from the **Toolbar** field.

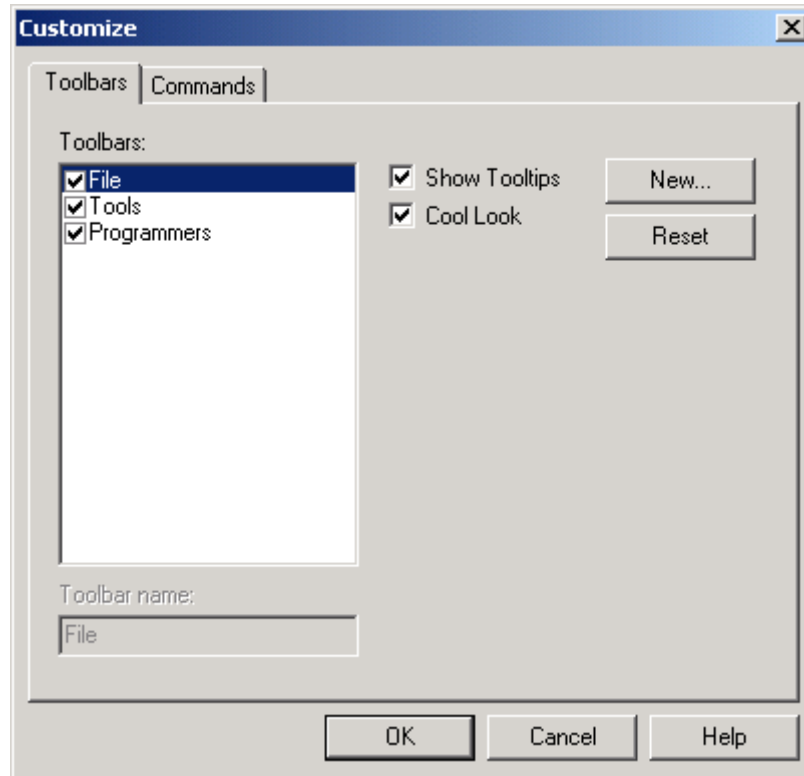


Figure 42 · Customize Dialog Box

3. Click inside the **Show Tooltips** checkbox for assistance in identifying icons on your toolbar when you scroll across them with your mouse.
4. Click inside the **Cool Look** checkbox to change the look of your toolbar.
5. Click **OK**.

You can create multiple toolbars and assign names to them. Click the **New** button and type in a name in the **New toolbar** dialog box to create a new toolbar. The name of your toolbar will display in the **Toolbar** field. Reset your toolbar to the default settings by clicking the **Reset** button.

Note: You can remove tools from your toolbar by deselecting tools from the **Toolbar** field.

To customize commands:

1. From the **Customize** menu, select **Toolbars**. The **Customize** dialog displays.
2. Click the **Commands** tab.

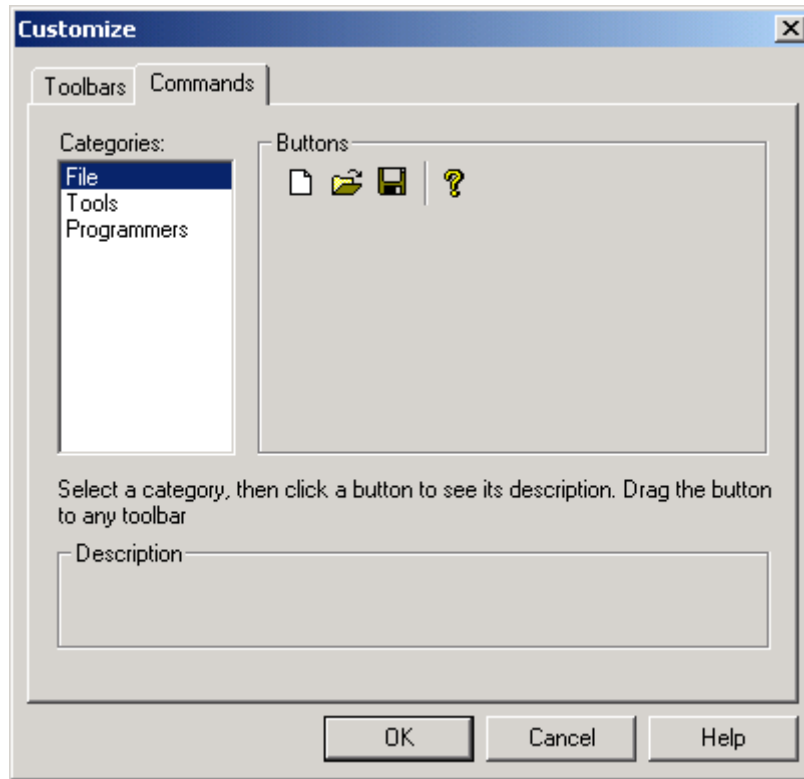


Figure 43 · Customize Dialog Box

3. Select a category by clicking one of three options (**File**, **Tools**, or **Programmers**). As you click an option, the buttons to the right of the category area change accordingly.
4. Click and drag a button to your toolbar.
5. Click **OK** after you have customized your toolbar.

You can also remove commands from your toolbar by reversing the click and drag method described in the steps above. Click and drag tools from your toolbar to the **Buttons** field in the **Customize** dialog box.

Customizing the Programming Window

The FlashPro software also enables you to customize the programmer window by right-clicking on the programmer window's header (see figure below).

Programmer ID	Programmer Name	Programmer Type	Port	Programmer Status	Programmer Enabled
---------------	-----------------	-----------------	------	-------------------	--------------------

Figure 44 · Programming Window Header

The following right-click menu displays (see figure below).



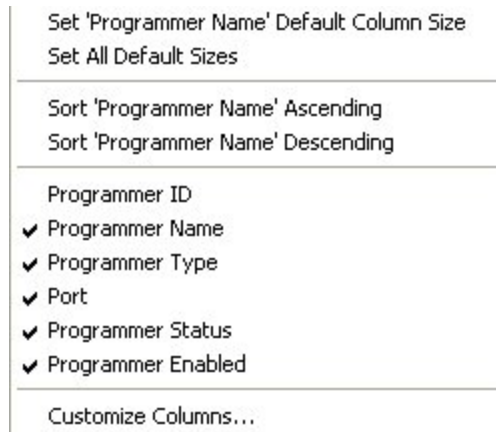


Figure 45 · Right-click Menu

The **Customize** right-click menu (as shown above) is divided into three sections. Click an item in the first section to set default sizes. Click an item in the second/middle section to add that item to the programmer window, and click the **Advanced** or last section to customize the columns in the **Programmer** window from the **Customize Columns** dialog box (see figure below).

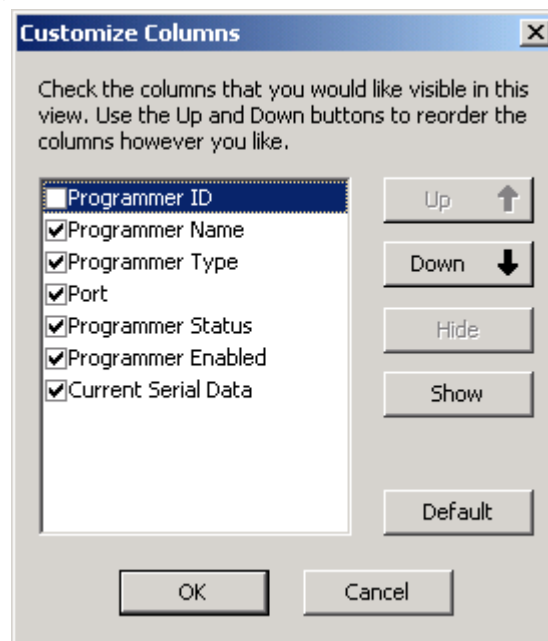


Figure 46 · Customize Columns Dialog Box

Note: Follow the instructions in the **Customize Columns** dialog box to customize the programming window. Use the **Up** and **Down** buttons to move through the list. Use the **Show** and **Hide** buttons to hide or show columns in the programmer window.

Preferences

The **Preferences** dialog box includes three tabs: **Log Window**, **Display Mode**, and **Updates** (see figure below). You can access the **Preferences** dialog box by choosing **File > Preferences**.

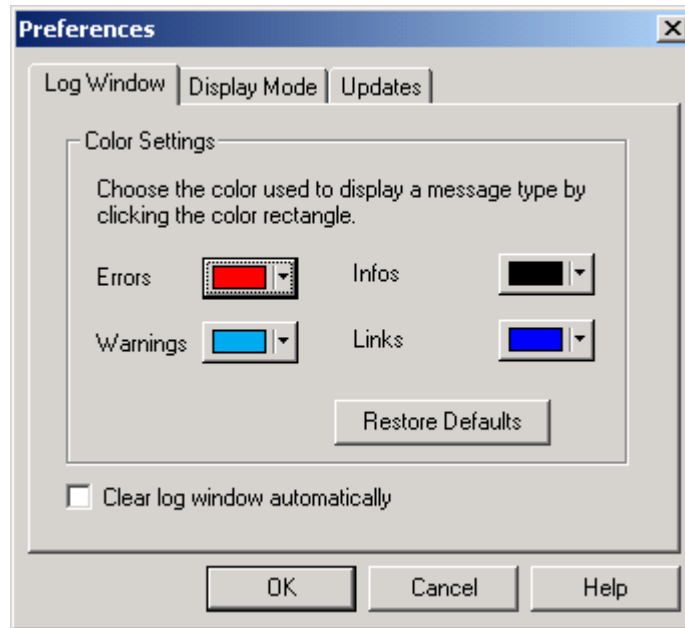


Figure 47 · Preferences Dialog Box

Log Window

The **Log Window** tab includes options for you to choose color settings for the various messages (Errors, Warnings, Information, Links) displayed in the **Log** window (see figure above).

Display Mode

The **Display Mode** tab describes the two display modes available in the FlashPro software (see figure below). Read each option carefully and choose the mode that will meet your programming needs. As the **Preferences** dialog box indicates, the **Classic Mode** is designed for multiple programming runs when it is not necessary for you to change your device settings. The **Advanced Mode** differs from the **Classic Mode** because displays both windows (**Programmer List** and **Device Configuration**) in the same GUI. Use this mode when you need to change device settings frequently.



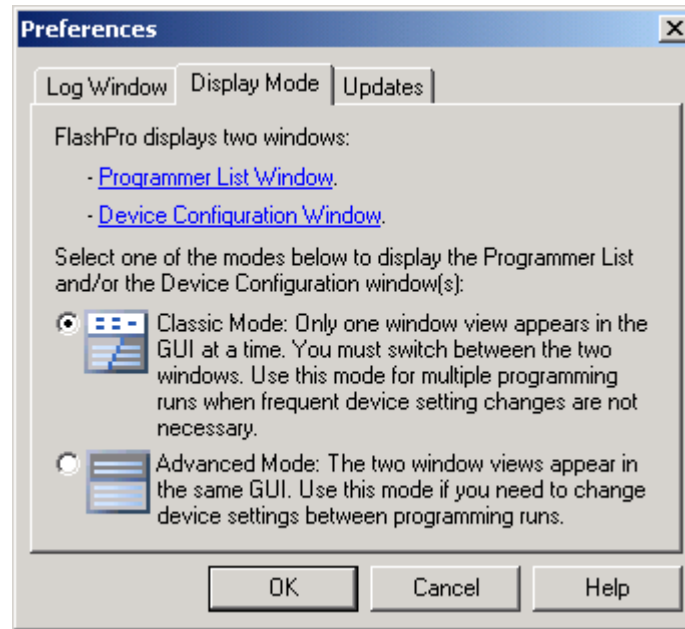


Figure 48 · Preferences Dialog Box- Display Mode

Software Updates

The **Updates** tab lists the FlashPro software setting options. You can choose to have the FlashPro software automatically check for updates at startup (from the Actel website) or remind you to check for updates at startup (requires you to go to the Actel website). If you want to decline both options, choose the last option: Do not check for updates or remind me at startup.

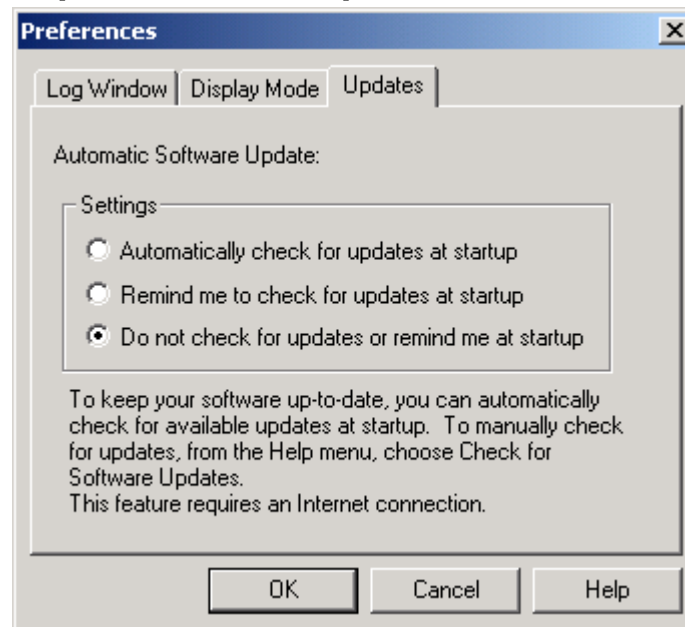


Figure 49 · Preferences Dialog Box- Updates

Software version is up to date

This informational message notifies you that there are no software updates available from Actel at this time. You can set your update preferences to automatically check for [software updates](#).



Programming Settings and Operations

Introduction

The FlashPro software enables you to connect multiple programmers to your computer. With each programmer you select, you can connect the programmer, perform a self-test, customize, add, and remove and analyze the JTAG chain.

Programmer Settings Information

The **Programmer Settings** dialog box includes setting options for FlashPro, FlashPro Lite, and FlashPro3.

Note: The user can set the TCK setting in the PDB/STAPL file by selecting the TCK frequency in the programming setting dialog box.

Limitation of the TCK frequency for the selected programmer:

- FlashPro Lite is limited to 4 MHz only.
- FlashPro Lite RevC supports 1-4MHz.
- FlashPro supports 1-8 MHz.
- FlashPro3 supports 1-6 MHz.

Limitation of the TCK frequency for the target device:

- APA/A500K – 10 MHz.
- A3P/E and Fusion – 10MHz to 20MHz

During execution, the frequency set by the **FREQUENCY** statement in the PDB/STAPL file will override the TCK frequency setting selected by user in the programmer setting dialog box unless the “Force TCK Frequency” checkbox is selected.

To set your programmer settings:

1. From the File menu, choose **Programmer Settings**.
The **Programmer Settings** dialog box displays (see figure below).

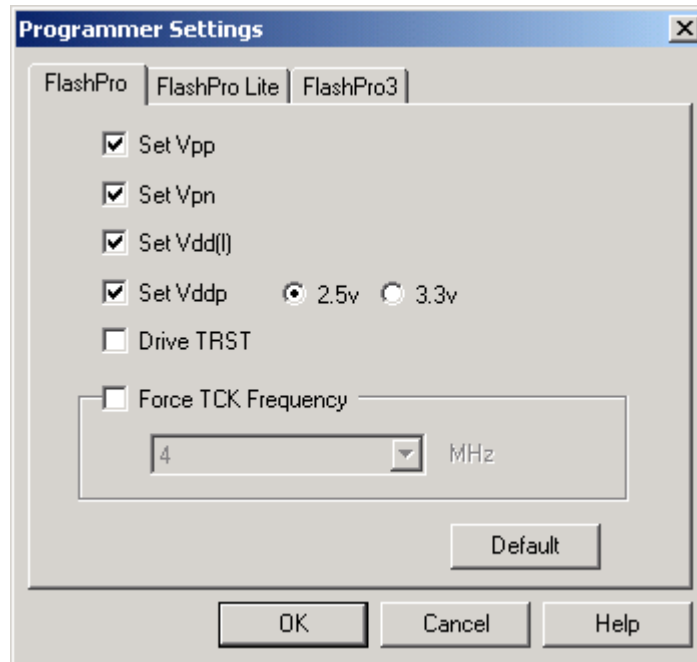


Figure 50 · Programmer Settings Dialog Box for FlashPro

2. Click on a programmer tab and check the appropriate settings for your programmer.
3. Click OK.

FlashPro Programmer Settings

Choose your programmer settings for FlashPro (see above figure). If you choose to add the Force TCK Frequency, select the appropriate MHz frequency. After you have made your selection(s), click OK.

FlashPro Default Setting

The default settings for FlashPro are listed below:

- The Vpp, Vpn, Vdd(l), and Vddp options are checked (Vddp is set to 2.5V) to instruct the FlashPro programmer(s) to supply Vpp, Vpn, Vdd(l) and Vddp.
- The Driver TRST option is unchecked to instruct the FlashPro programmer(s) NOT to drive the TRST pin.
- The Force TCK Frequency option is unchecked to instruct FlashPro to use the TCK frequency specified by the Frequency statement in the STAPL file(s).

FlashPro Lite Programmer Settings

Choose your programmer settings for FlashPro Lite (see figure below). If you choose to add the Force TCK Frequency, select the appropriate MHz frequency. After you have made your selection(s), click OK.



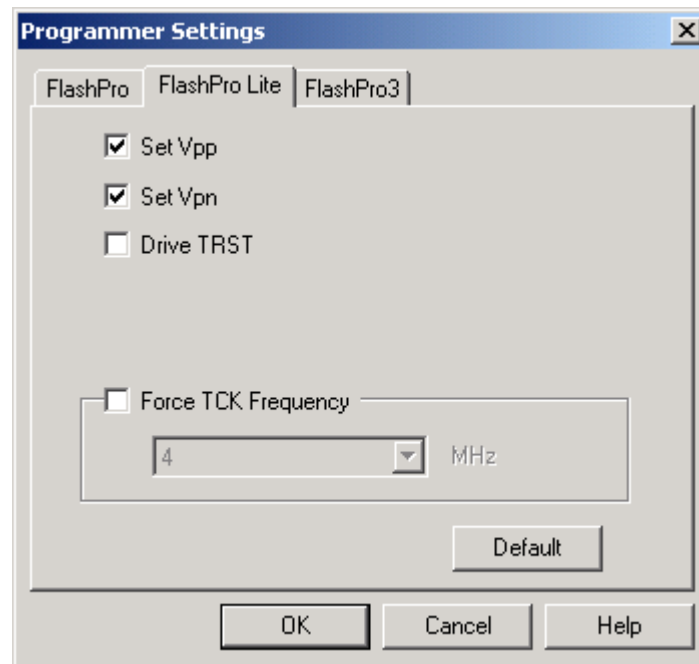


Figure 51 · Programmer Settings Dialog Box for FlashPro Lite

FlashPro Lite Default Setting

The default settings for FlashPro Lite are listed below:

- The Vpp and Vpn options are checked to instruct the FlashPro Lite programmer(s) to supply Vpp and Vpn.
- The Driver TRST option is unchecked to instruct the FlashPro Lite programmer(s) NOT to drive the TRST pin.
- The Force TCK Frequency option is unchecked to instruct the FlashPro Lite to use the TCK frequency specified by the Frequency statement in the STAPL file(s).

FlashPro3 Programmer Settings

Choose your programmer settings for FlashPro3 (see figure below). For FlashPro3, you have the option of choosing the Set Vpump setting or the Force TCK Frequency. If you choose the Force TCK Frequency, select the appropriate MHz frequency. After you have made your selections(s), click OK.

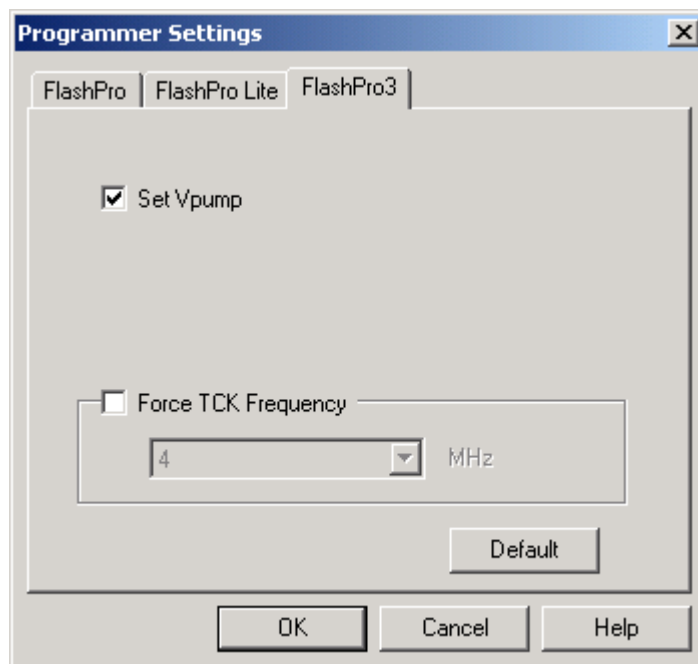


Figure 52 · Programmer Settings Dialog Box for FlashPro3

FlashPro3 Default Setting

The default settings for FlashPro3 are listed below:

- The Vpump option is checked to instruct the FlashPro3 programmer(s) to supply Vpump to the device.
- The Force TCK Frequency option is unchecked to instruct the FlashPro3 to use the TCK frequency specified by the Frequency statement in the PDB/STAPL file(s).

TCK Setting (Forcing TCK Frequency)

If Force TCK Frequency is checked (in the Programmer Setting) then the selected TCK value is set for the programmer and the Frequency statement in the PDB/STAPL file is ignored.

Note: FlashPro Lite RevA supports only 4MHz on TCK.

Default TCK frequency

When the PDB/STAPL file or Chain does not exist, the default TCK frequency is set to 4MHz. In the **Single Device File Programming** mode, FlashPro will parse through the file and search for the "freq" keyword and the "MAX_FREQ" Note field, which are expected in all Actel Flash device files. The FlashPro software will use the lesser value between the two as the default TCK frequency.

In **Chain Programming** mode, when more than one Actel Flash device is targeted in the chain, the FlashPro software passes through all of the files and searches for the "freq" keyword and the "MAX_FREQ" Note field. The FlashPro software will use the lesser value of all the TCK frequency settings and the "MAX_FREQ" Note field values.

Ping Programmers

To ping a programmer(s):

1. From the **Programmers** menu, choose **Ping**.
2. Select the programmers you want to connect from the **Select Programmer(s)** dialog box.
3. Click **OK**.

Note: You can click the Refresh/Rescan for Programmers button to quickly ping new programmers.

Performing a Self-Test

To perform a self-test:

1. From the **Programmers** menu, choose **Self Test**.
2. Select the programmer(s) you want to self-test from the **Select Programmer(s)** dialog box.
3. Click **OK**.

Note: You must connect the programmer to the self-test board that comes with your programmer before performing a self-test.

You can also perform self-test by right-clicking on a specific programmer from the **Programmer List Window** and selecting **Self-Test**.

Note: Self-test is not supported with FlashPro Lite programmers.

Scanning a Chain

The scan chain operation scans and analyzes the JTAG chain connected to programmer(s) you have selected.

To scan a chain:

1. From the **Programmers** menu, choose **Scan Chain**.
2. Select the programmers you want to scan from the **Select Programmer(s)** dialog box.
3. Click **OK**.

You can also perform Scan Chain by right-clicking on a specific programmer from the **Programmer List Window** and selecting **Scan Chain**.

Enabling and Disabling Programmers

Once your programmer is enabled you can connect the programmer, perform a self-test, scan the chain, or remove it.

To enable a programmer:

1. From the **View** menu, choose **Programmer Details Window**.
2. Check the **Enable programmer** checkbox in the upper left corner of the **Programmer Details Window**.

The **Programmer Details** window displays all the information about your programmer.

Note: You can also enable your programmer from the **Programmer List** window by checking the checkbox in the **Programmer Enabled** column.

Disable your programmer by unchecking the **Enable programmer** checkbox from the **Programmer Details Window** or by unchecking the checkbox in the **Programmer Enabled** column in the **Programmer** window.

Renaming a Programmer

Enter the new programmer name in the **Programmer Details** window to rename the programmer. By default, the programmer name is the same as the programmer ID.

Removing a Programmer

To remove a programmer:

1. From the **Programmers** menu, choose **Remove**.
2. Select the programmers you want to remove from the **Select Programmer(s)** dialog box.
3. Click **OK**.

Selecting Programmers

The **Select Programmer(s)** dialog box gives you the option of selecting all and unselecting all of the programmers that you want to ping. See figure below.

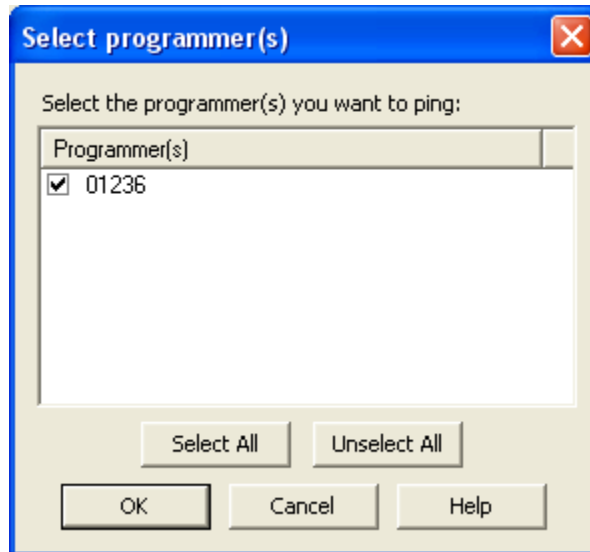


Figure 53 · Selecting Programmer(s) Dialog Box

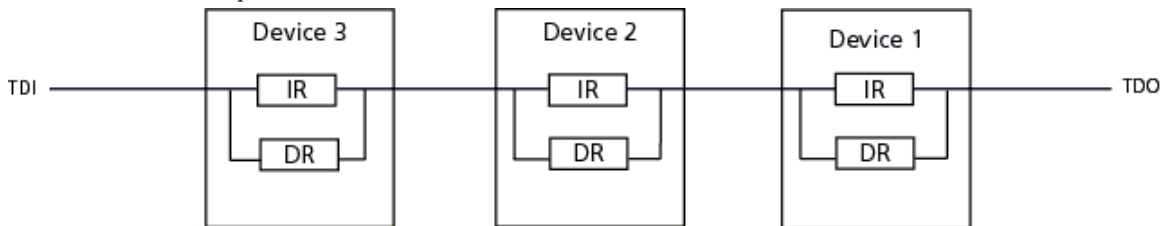
Single Device Configuration

Single Device Programming

When devices are joined together in a JTAG chain, all of their Instruction Registers (IR) and Data Registers (DR) are put in a long shift register from TDI to TDO. The IR length defers from device to device and the DR length depends on the instruction that shifts into the instruction register.

When targeting Device 2 (see figure below), you need to know the IR length for Device 1 and Device 3. Given this information, you can bypass both devices by shifting an all one pattern into their instruction registers before and after the instruction targeted at Device 2. The number of bits you shift before Device 2's instruction is the pre IR length, and the number of bits you need to add after Device 2's instruction is the post IR length. In this case, the pre IR bits are shifted into

Device 1 and post IR bits are shifted into Device 3.



Targeting Device 2

When the bypass instruction is shifted into Device 1 and Device 3, the TDI and TDO of the two devices are connected to the 1-bit bypass register at the Shift-DR state. To correctly shift the data in and data out of Device 2's register, you need to shift one bit of data before and after Device 2's data.

The number of bits you need to shift into the data register for Device 1 is the pre-DR length and the number of bits we need to shift into the data register for Device 3 is the post-DR length. With the IR and DR length information, you can shift instructions and data into Device 2 with the correct registration.

To create the JTAG chain (shown in the above figure):

1. Connect the TCK and TMS from the programmer to all of the devices.
2. Connect the programmer's TDI pin to the TDI pin of device 3.
3. For all devices in the chain, connect the TDO output of one device to the TDI input of the next device.
4. Connect the TDO output of the last device to the programmer's TDO input.

The order of devices in the chain is set by the connections of TDI to TDO.

The ChainBuilder software takes the order of the devices in a chain and their IR lengths and adds the pre-IR, post-IR, pre-DR, and post-DR padding bits in the device you want to program, which properly aligns the instructions and data within the IR and DR of the devices.

If you do not use the ChainBuilder software, the FlashPro software tries to find the pre-IR, post IR, pre-DR, and post-DR values during the Analyze Chain operation.

For more information, see ProASIC programming and [ProASICPLUS and ProASIC programming introduction](#).

To find out how to set the IR length, see [Chain Settings](#).

Loading a Programming File

You can either load a programming file from the **Configuration** menu (see figure below) or from the **Single Device Programming Window**. The section below describes how to load a programming file from the **Single Device Programming Window**.

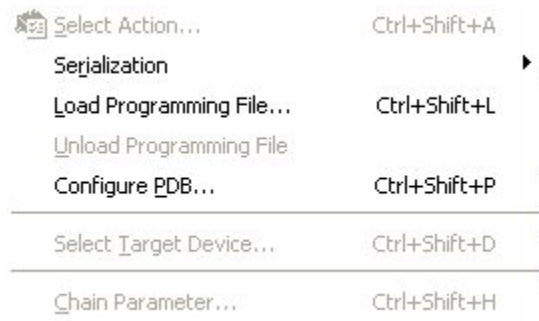


Figure 54 · Configuration Menu

To Load a programming file from the Single Device Programming window:

1. From the **View** menu, select **Single Device Configuration** to activate the **Single Device Configuration Window**.
2. Click the **Browse** button in the **Single Device Configuration Window** (see figure below).

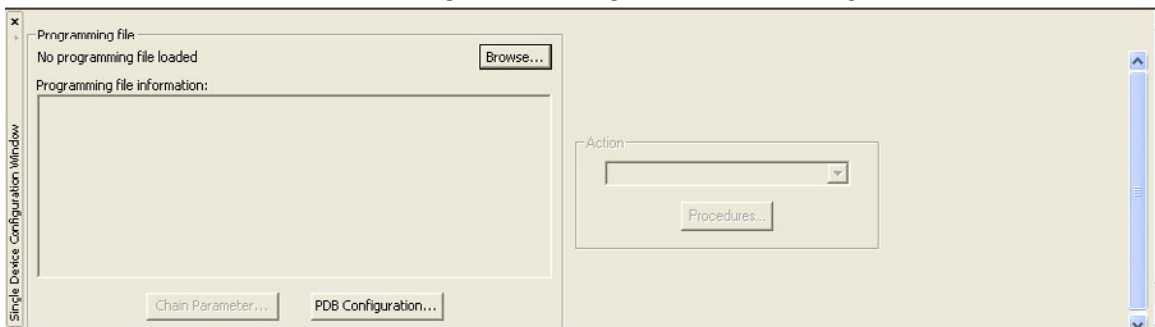


Figure 55 · Signal Device Configuration Window

The **Load Programming File** dialog box appears (see figure below).



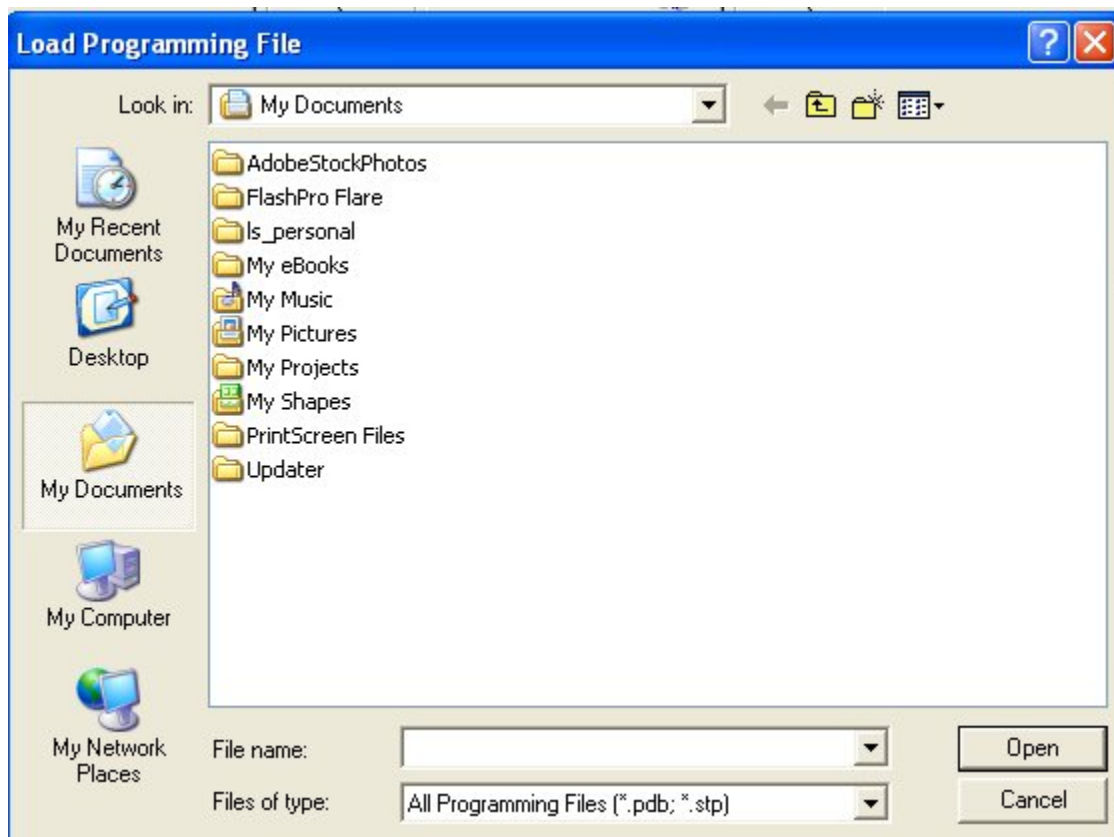


Figure 56 · Load Programming File Dialog Box

3. Find your programming file and click **Open**.
The programming file is loaded and the **Single Device Configuration** Window updates.

Select Target Device

The **Select Target Device** dialog box is located in the **Configuration** menu (see figure below).

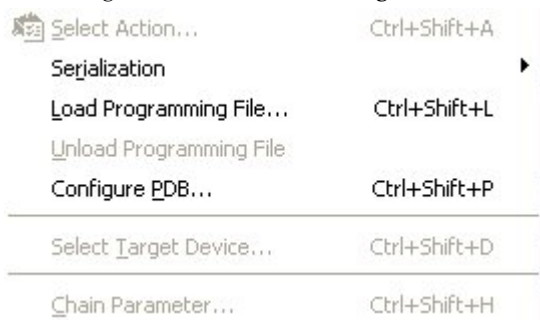


Figure 57 · Configuration Menu

The **Select Target Device** dialog box enables you to select the target device you want to program. If you are only programming one device in a chain, there is no need for you to make a selection. The **Select Target Device** dialog box automatically displays your device (see figure below).

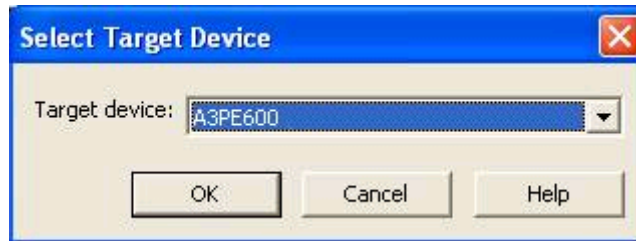


Figure 58 · Select Target Device (One Device in a Chain)

If you are programming more than one Flash device in a chain, you need to select the target device you want to program. If you attempt to program your device without selecting a target device, the following warning message displays:

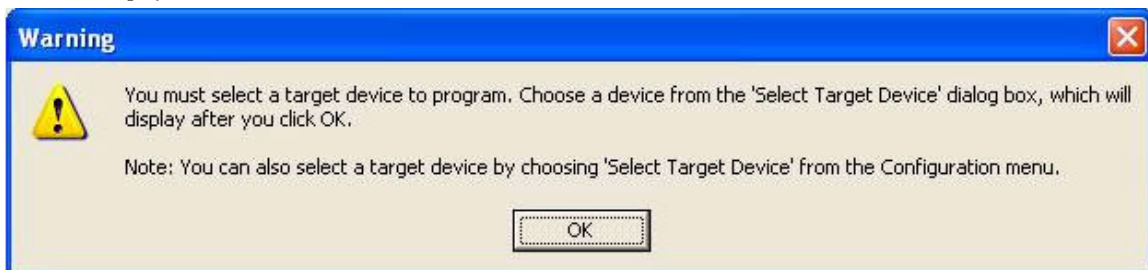


Figure 59 · Select Target Device Warning Message

If this warning message appears, click OK and the **Select Target Device** dialog box will display. From the **Select Target Device** dialog box, select the device you want to program (see figure below).



Figure 60 · Select Target Device (Multiple Devices in a Chain)

Click the down arrow to display the list of devices in your chain. Then, make your selection and click OK.

Note: When the FlashPro software does not detect your chain configuration, you must specify the Pre/Post IR fields by entering these values in the Set Pre/Post IR Values dialog box (see figure below).



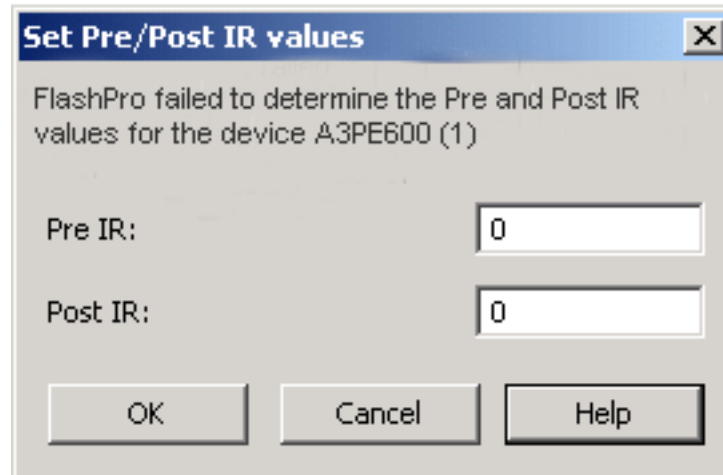


Figure 61 · Pre IR and Post IR Values for the Target Device

For more information, see [Single STAPL file programming information](#).

Chain Settings

Click the **Chain Parameter** button in the **Single Device Configuration** window to set the chain settings (see the **Chain Settings** dialog box below). See [Single Device Programming Information](#) for more information about these STAPL settings.

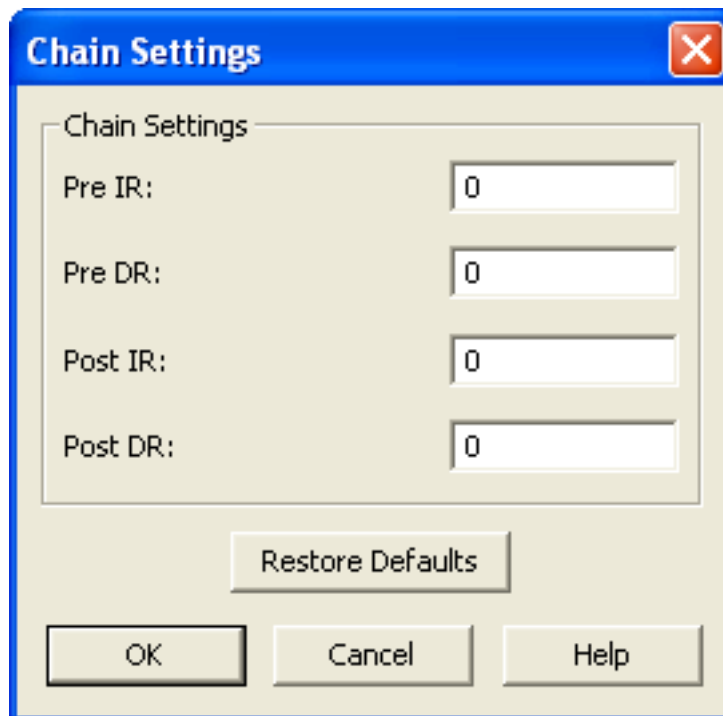


Figure 62 · Chain Settings Dialog Box

Serial Settings

Click the **Select Serialization Indexes/Select Serialization Actions** button from the **Single Device Configuration** Window. The **Serial Settings** dialog box displays. The figure below is an example.

Note: Depending on the STAPL file format (Actel format or generic format) used, you will either see Indexes columns or Actions columns in the Serial Settings dialog box.

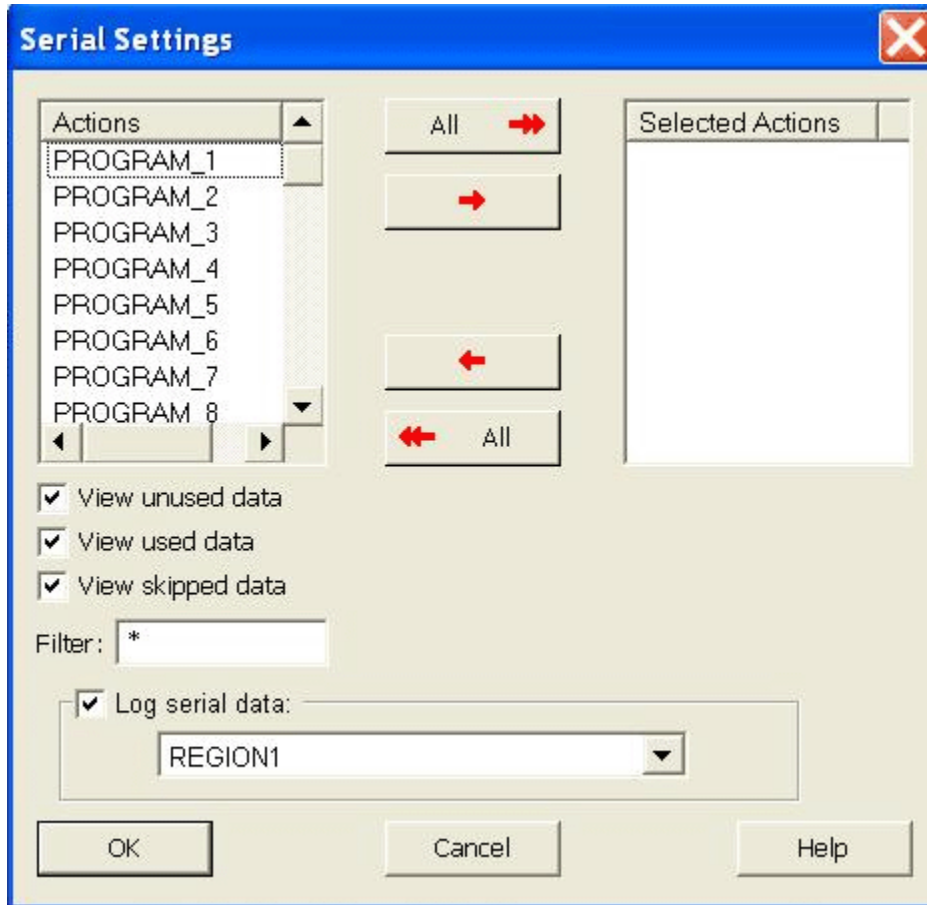


Figure 63 · Serial Settings Dialog Box

Click the red arrow buttons in the center of the dialog box to move from the **Actions** column to the **Selected Actions** column. The indexes/actions available for selection are located on the left and the indexes/actions you choose to select are located on the right column. Viewing options are available in the checkboxes under the **Actions** column. If you check **Log serial data**, you can select the FlashROM region name where the serial data will be stored.

Skip Serial Data

If you are unable to perform the programming action on your device, if your device fails to program, and you have selected the **Skip Serial Data** serialization setting, the software automatically uses the next serial data when you program the next device. By default, the software is set to **Skip Serial Data**.

You can change the serialization setting by selecting **Tools > Serialization** or you can click the **Skip serial data** icon or the **Reuse serial data** icon from the toolbar.

Reuse Serial Data

If your device fails to program, and you have selected the **Reuse Serial Data** serialization setting, the software automatically reuses the current serial data when you program the next device.

Note: The FlashPro default setting is [Skip Serial Data](#).

You can change the serialization setting by selecting **Tools > Serialization** or you can click the **Skip serial data** icon or the **Reuse serial data** icon from the toolbar.

Serialization with Parallel Programming

When programming the multiple ProASIC3 devices in parallel, while performing serialization at the same time, each target device is assigned a Serial Index/Action for each programming run. Upon each successful completion of each programming run, a new index is assigned to the each target device for the next programming run. This process continues until the selected **Serial Indices/Actions** are exhausted.

Note: If programming failure is encountered, depending on the user setting, the failed serial data may be reused or skipped in the next programming run.

Note: If, in the last programming run, the remaining number selected Serial Indices/Actions is less than the number of targeted ProASIC3 devices, the targeted devices without an assigned Serial Index/Action are skipped in the final serial programming run.

Chain Programming

Understanding the Chain Order

The chain order is located in the [Chain Configuration Window](#). The devices you add to the chain must be in the correct order and must match the physical chain to be programmed. The TDO for the first device connects to the programmer, and the last device's TDI connects to the programmer. The devices in the chain go in order from a device's TDI into the next device's TDO, as shown in the figure below.

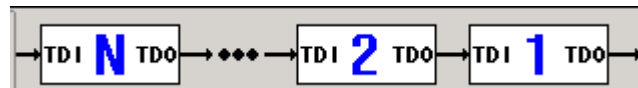


Figure 64 · Chain Order

Introduction to Multiple Device Chain Programming

The FlashPro software enables direct chain programming without generating a chain STAPL file. Each device will be programmed in sequential order starting from device 1 to device N. See example below. For more information about chain order, see [Understanding the Chain Order](#).

TDI > Device N > Device N-1 > ... > Device 2 > Device 1 > TDO

You have the advantage of using the Chain Builder GUI interface to construct the target physical chain. Therefore, you do not need to calculate the PRE/POST IR/DR value of the target device. Instead, you must provide either a valid BSDL file or the IR length and TCK Fmax values when you add a non-Actel device to the chain.

Note: Even though the FlashPro software enables direct chain programming without generating a Chain STAPL file, this functionality is still available. For more information, see [Export Chain STAPL file](#).

Device Programming Compatibility

The following is a list of Flash Family devices that can be programmed together in a chain.

Note: IGLOO, Fusion, and ProASIC3, excluding ProASIC3L, families can be programmed in the same chain.

Note: ProASIC^{PLUS} can only be programmed with other ProASIC^{PLUS} devices.

Note: ProASIC can only be programmed with other ProASIC devices.

Programmer Support

FlashPro3 supports only IGLOO, Fusion, and ProASIC3 family devices. The Vpump on FlashPro3 is designed to support the programming of only one device. Please make sure that Vpump, Vcc and Vjtag are provided on board for chain programming. Connect the Vpump to the header as the FlashPro software will attempt to check for all external supplies, including Vpump, to ensure successful programming. There is no limitation to the chain length; however, ensure that the JTAG signal integrity and the timing are preserved.

FlashPro and FlashPro Lite support both ProASIC^{PLUS} and ProASIC family devices. However you cannot program both devices in the same chain.

Unless all supplies are provided on board, there is a limitation of programming eight ProASIC^{PLUS} or ProASIC family devices in a chain.

Multiple Device Serialization in a Chain

When you program multiple IGLOO, Fusion, and ProASIC3 family parts, you can use the serialization functionality for more than one device. You must generate STAPL files with the correct serialization data in them to use this functionality.

Each serialization enabled STAPL file may contain a different number of serialization data, but you may only select the same number of serialization data to program in a single Serialization/Programming session.

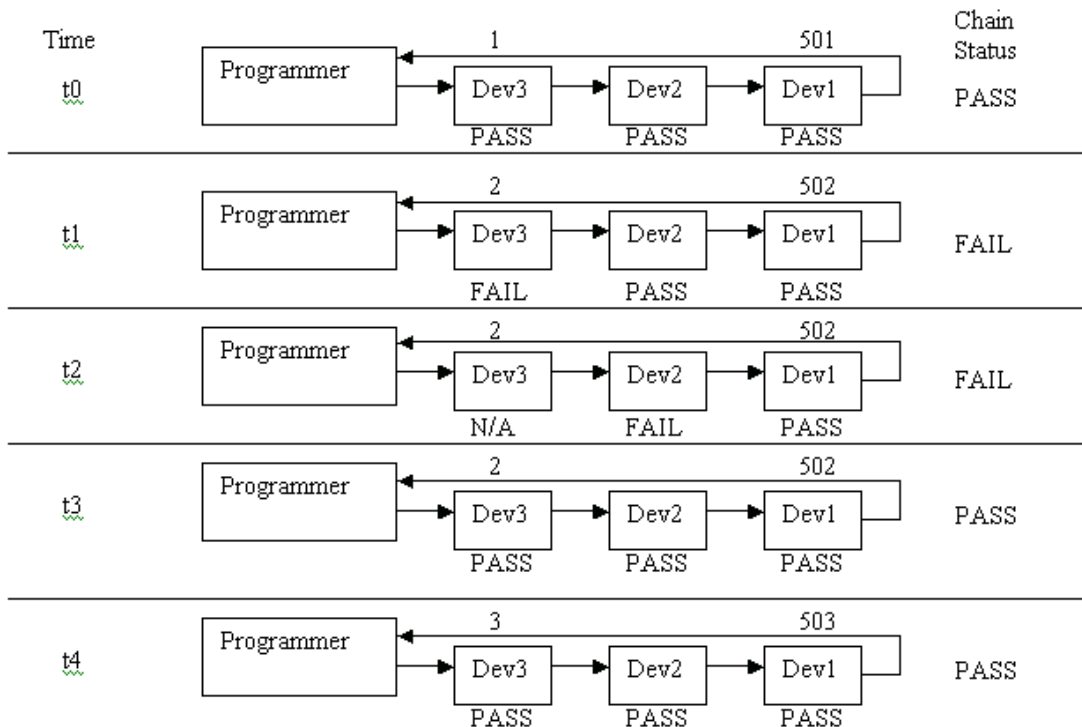
See the example below for further explanation:

In this example, you have a chain of two devices (one device is an A3P250 and the other device is an A3PE600). The STAPL file for the A3P250 contains 10,000 serialization data and the STAPL file for the A3PE600 has 5,000 serialization data. In a single Serialization/Programming session, you are allowed to select serialization data indexed from 1 to 1,000 for the A3PE600 device and serialization data indexed from 5,001 to 6000 for the A3P250 device. However, FlashPro will error out (at the beginning of a programming operation) when the amount of the Serialization data you select is different from the devices.

Reuse Serial Data That Failed Programming

If any of the devices in the chain fail programming, the entire chain fails. All of the devices with serialization enabled will fail as well. The serialization data will be reused or skipped based on your settings. See the example below for more information:

You have a chain with three devices. Device 1 and Device 3 are serialization enabled. You have selected Serialization Data 1 to 100 for Device 1 and 501 to 600 for Device 3, and you have set to reuse any unused Serialization Data. Device 2 is targeted for programming without serialization. See the figure below for an illustration.



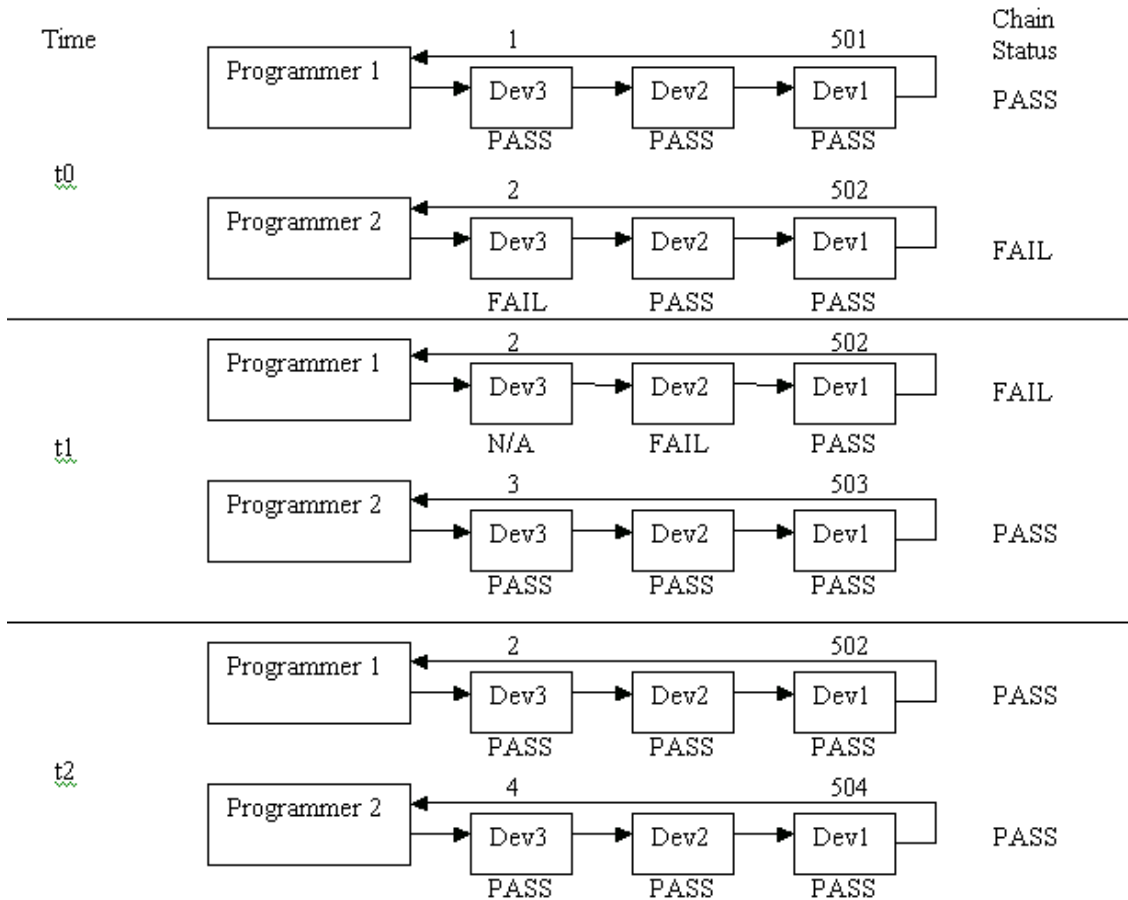
At time t0, all devices in the chain passed programming so the Serialization Data indexes are advance to 2 and 502 for Device 3 and Device 1 respectively. At time t1 and t2, one of the devices failed programming so the device indexes are reused for time t3.



Note that at time t2, when Device 2 failed to program; Device 3 will not be programmed.

Multiple Device Serialization and Parallel Programming

The FlashPro software enables parallel programming for ProASIC3, excluding ProASIC3L, family devices using multiple FlashPro3 programmers. The following figure illustrates how the indexes are reused in a parallel programming environment.



At time t0, the chain failed to program so index 2 and 502 are reassigned to Device 3 and Device 1 respectively at time t1. The failed indexes are not assigned to the programmer that previously failed. It will always be assigned to the devices in the first programmer in the Programmer List.

Chain Configuration Window

The **Chain Configuration** window displays the chain order, the chain editing options, and the chain configuration grid (see figure below).

The **Show Chain Editing** checkbox, when checked displays your chain editing options (Configure device, Add Actel Device, Add Non-Actel Device, and organization buttons to move your device within the grid).

Note: For information on how to Add Actel and Non-Actel devices, see [Chain Editing](#).

Note: For information on how to use the Organize buttons (located next to the Add Actel and Add Non-Actel buttons) in the Chain Configuration grid, see [Using the Organize buttons in the Chain Programming grid](#).

You can enable programming and serialization by checking the **Enable Device** checkbox and the **Enable Serial** checkbox in the **Chain Configuration** grid.

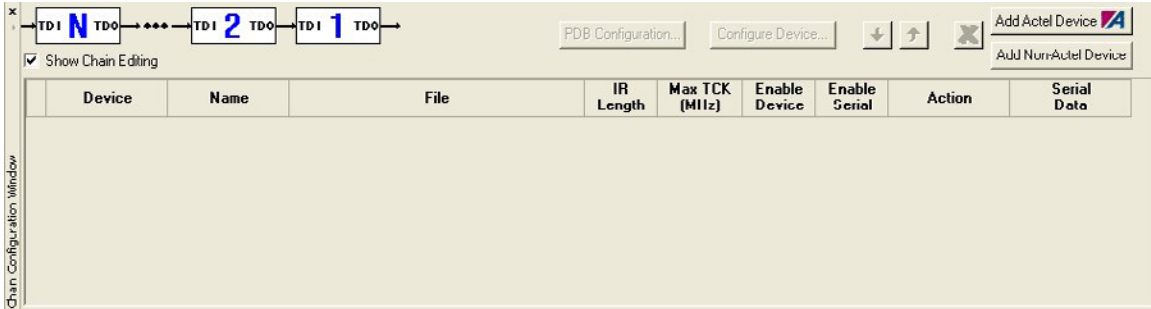


Figure 65 · Chain Configuration Window

Chain Editing Options

The FlashPro software enables you to edit your chain by adding Actel and Non-Actel devices. You can add devices by clicking the **Add Actel Device** button and the **Add Non-Actel Device** button or you can select these options from the **Configuration** menu.

Note: For more information about how to edit the chain, see [Chain Editing](#).

Editing the Chain Configuration Grid

The **Chain Configuration Grid** enables you to select an **Action** for your device, **Enable Serialization**, and edit the grid using the right-click menu.

To select an Action from the Configuration Grid:

1. Choose the device you would like to program and check the **Enable Device** checkbox.
2. In the **Action** column, click the down arrow to expose the drop down menu (see figure below).
3. Select your desired action.

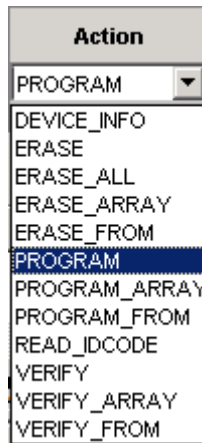


Figure 66 · Drop Down Menu for Select Action

Before you can enable serialization, you must check the **Enable Device** checkbox.

To enable Serialization:

1. Check the **Enable Serial** checkbox. By enabling serialization, the action options change.
2. In the **Action** column, click the down arrow to expose the drop down menu (see figure below).

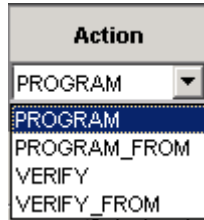


Figure 67 · Drop Down Menu for Select Action

3. Select your desired action.



Figure 68 · Serial Data Column

4. Click the **Select** button from the **Serial Data** column, which is next to the **Action** column (see above figure). The **Serial Settings** dialog box displays.
5. Choose your serial settings from the **Serial Settings** dialog box.
See [Serial Settings](#) for more information about this topic.
Note: Uncheck the **Enable Serial** checkbox to disable serialization.

To edit the Chain Configuration Grid:

1. Select the device you would like to edit and right click any where in the row of the selected device. The right-click menu below displays.

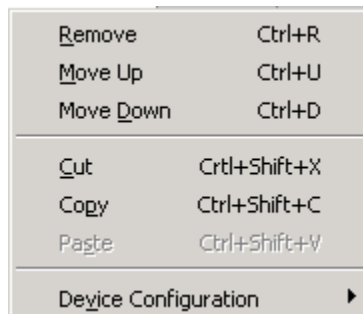


Figure 69 · Right-click Menu for Grid Editing

2. Select and click an option from the right-click menu.
Note: The **Device Configuration** menu (see figure below) includes options for configuring your device.

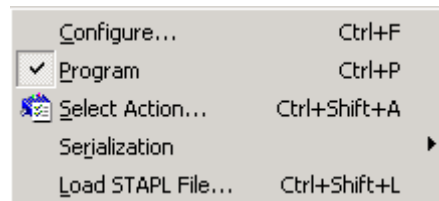


Figure 70 · Device Configuration Menu

Chain Editing

The chain order is located in the [Chain Configuration Window](#) (see figure below). The devices you add to the chain must be in the correct order and must match the physical chain to be programmed.

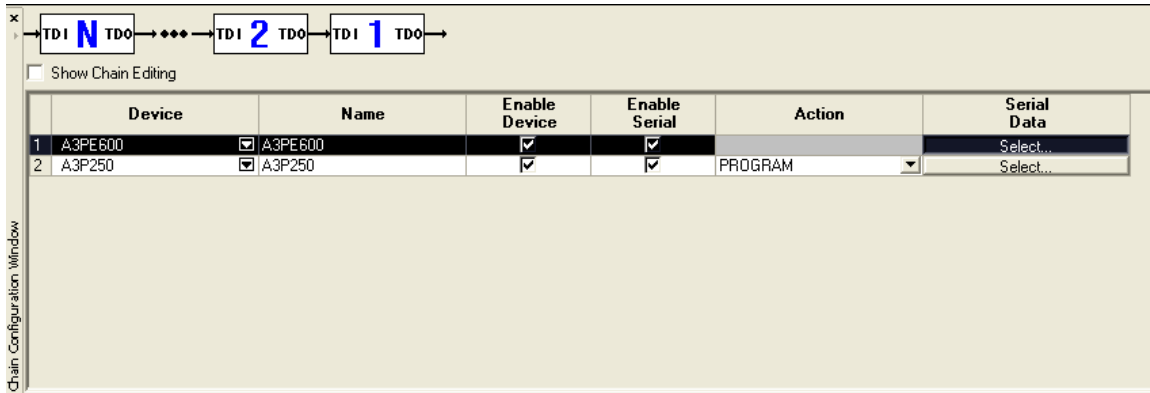


Figure 71 · Chain Configuration Window

Check the **Show Chain Editing** checkbox to display chain editing options (**Add Actel Device**, **Add Non-Actel Device**). See figure below.

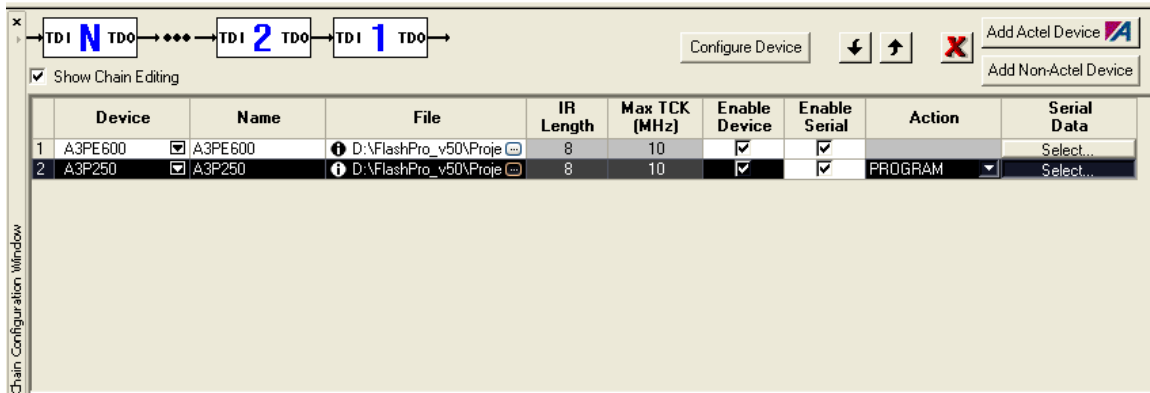


Figure 72 · Chain Configuration Window

You can edit the chain by adding Actel and Non-Actel Devices, for information refer to:

- [Adding Actel Devices](#)
- [Adding Non-Actel Devices](#)
- Adding Actel Devices from a STAPL File
- [Chain Programming Tutorial](#)

Using the Organize Buttons in the Chain Programming Grid

The organize buttons enable you to select the order of the devices in your **Chain Programming** grid (see figure below).

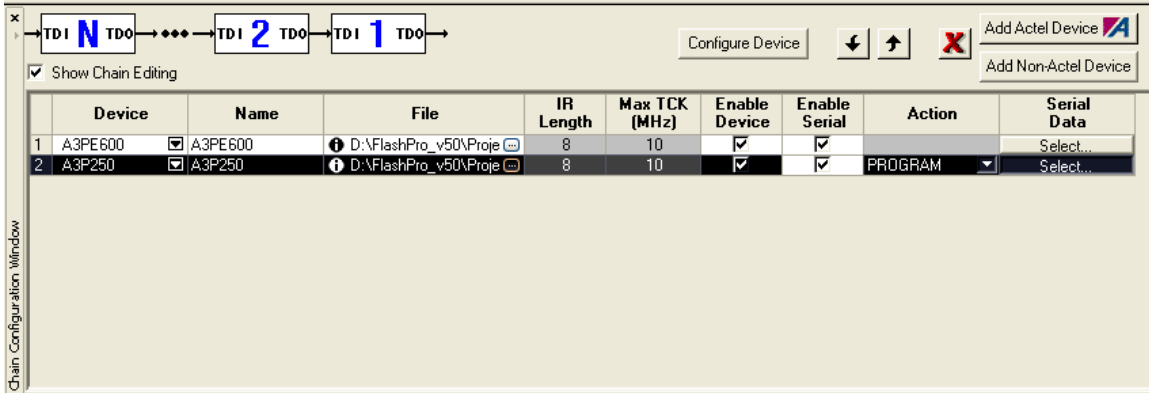


Figure 73 · Chain Configuration Window (Displaying Organize Buttons)

You can move devices up and down or delete devices within the grid. See the table below for a description of each button.

Table 6 · Organize Buttons

Button	Description
	Moves your device up in the Chain Programming grid.
	Moves your device down in the Chain Programming grid.
	Deletes your device from the Chain Programming grid.

Cutting, Copying, and Pasting Devices from the Chain

If you want to make changes to your chain, you must make these changes from the spreadsheet in the **Chain Programming** grid.

To copy or cut a device from the chain programming grid:

1. Select the device you would like to edit and right click anywhere in the row of the selected device. The right-click menu below displays.

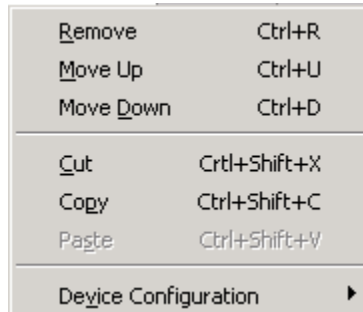


Figure 74 · Right-Click Menu

2. Select **Copy** or **Cut** from the right-click menu to copy your device.

To paste a device from the chain programming grid:

1. Right-click the location where you would like to **Paste** the device.
2. Select **Paste** from the right-click menu (see figure below).

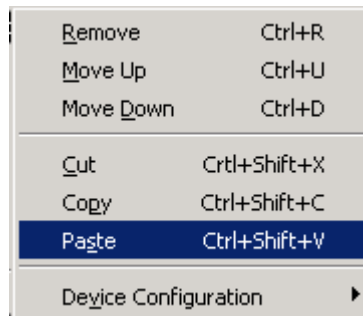


Figure 75 · Right-Click Menu

Removing Devices from the Chain

If you want to make changes to your chain, you must make these changes from the spreadsheet in the **Chain Programming** grid.

To remove a device from the chain programming grid:

1. Select the device you would like to remove and right click any where in the row of the selected device. The right-click menu below displays.

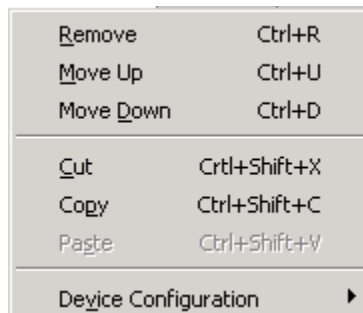


Figure 76 · Right-Click Menu

2. Select **Remove** from the right-click menu to delete your device.



Moving Devices within the Chain

You can move devices within the chain by using the **Organize** buttons (located next to the Add Actel and Add Non-Actel Device buttons) in the **Chain Programming** grid. See figure below.



Figure 77 · Organize Buttons

To move a device within the chain:

1. Select the device you would like to move by clicking on it.
2. Click one of the **Organize** button arrows to move your device up or down the spreadsheet.

Note: You can remove a device by choosing the red X Organize button.

For more information about the **Organize** buttons, see [Using the Organize buttons in the Chain Programming grid](#).

Chain Editing

Adding an Actel Device

To add an Actel device:

1. Click the Add Actel Device button from the Chain Programming grid. The Add Actel Device dialog box displays (see figure below).

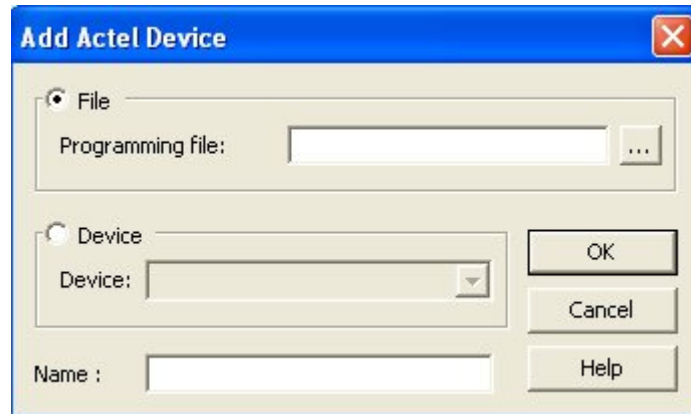


Figure 78 · Add Actel Device

2. Select your device from the **Device** text box by clicking the down arrow to expose the device list.
3. Click the **File** radio button, then click the **Browse** button (...) in the **Programming file** text box to find your PDB/STAPL file. The **Use File** dialog box displays.
4. Find your PDB/STAPL file and click **Open**. Your PDB/STAPL file name appears in the **Name** text box. You can change the name by clicking in the text box.
5. Click **OK**. Your device displays in the **Chain Programming** grid.
You can also add an Actel device from **Configuration > Add Actel Device**.

Adding an Actel Device from Files

To add an Actel device from a file:

1. From the **Configuration** menu, choose **Add Actel Devices From Files**. The **Add Actel Devices From Files** dialog box displays.
2. Locate your file and click **Open**. Your device displays in the **Chain Programming** grid.

Adding a Non-Actel Device

When adding a non-Actel device, you must choose either a BSDL file or customize the Instruction Register (IR) length and the Max TCK frequency of the device.

IR Length

The IR length specifies the number of IR bits in a specific device.

Max Frequency

The Max TCK is the maximum TCK (clock) frequency of a specific device. FlashPro can use this information to ensure that the programmer operates at a frequency lower than the slowest device in the chain.

BSDL File

Boundary Scan Description Language (BSDL) files describe the characteristics of a specific device. When using a BSDL file, FlashPro extracts the IR length and TCK frequency for the specific device and uses the information to build the FlashPro STAPL file. If you do not have a BSDL file for your specific device, you must manually enter the IR length and Max TCK for your device. This information should be found in the datasheet for the device.

To add a non Actel device using a BSDL file:

1. Click the **Add Non-Actel Device** button in the **Chain Programming** grid. The **Add Non-Actel Device** dialog box displays (see figure below).

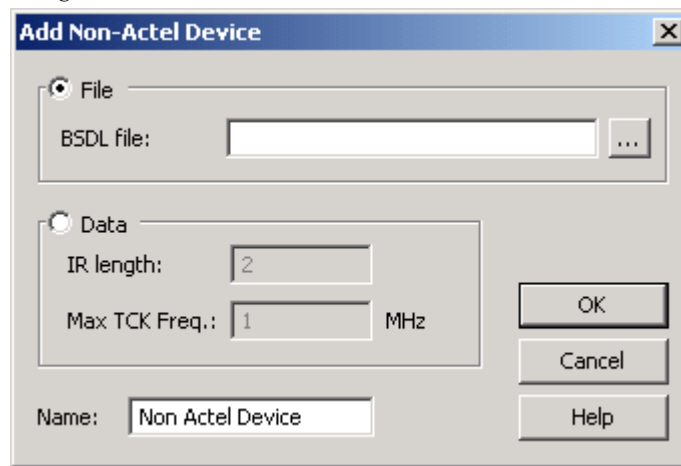


Figure 79 · Add Non-Actel Device Dialog Box

2. Type in the BSDL file or locate it by clicking the **Browse** button. If you click the **Browse** button to find your BSDL file, the **Use File** dialog box displays.
3. Select your BSDL file from the **Use File** dialog box, and click **Open**.
4. Click **OK**, and your device appears in the **Chain Programming** grid.

When closing the **Add Non-Actel Device** window, if you specified a BSDL file, it is parsed and its IR length and Max TCK frequency are retrieved.

Note: If you select a BSDL file, you cannot specify an IR length and Max TCK frequency.

To add a Non-Actel device using an IR length and a Max TCK frequency:

1. Click the **Add Non-Actel Device** button from the **Chain Programming** grid.
2. Click the **Data** option from the **Add Non-Actel Device** dialog box.
3. Enter the IR length AND the Max TCK frequency in MHz.
4. Click **OK**.

If you decide to use custom data, you must specify both an IR length and Max TCK frequency.

Note: The IR length must be an integer greater than or equal to 2, and the Max TCK frequency must be a float greater than or equal to 1.

Configuring a Programmer

Selecting an Action

The actions you have available are based upon what type of STAPL file you have loaded into the software.

To configure a programmer:

1. From the **Configure** menu, choose **Select Action and Procedures**. The **Select Action and Procedures** dialog box displays (see figure below).

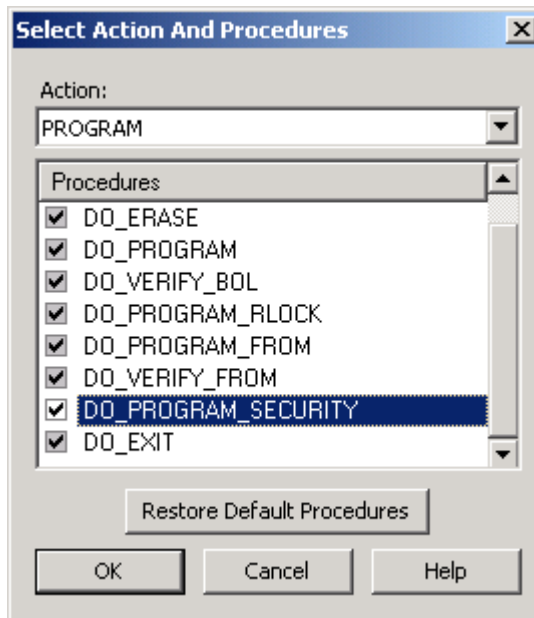


Figure 80 · Select Action of Procedures Dialog Box

2. Click the checkboxes of the available procedures or click the **Restore Default Procedures** button in the **Select Action and Procedures** dialog box.
3. Click **OK**.

Note: Restore Default Procedures by clicking on the appropriate button.

Using Serialization

You can activate serialization by following the steps below.

To use serialization:

1. Enable serialization by checking the **Serialization** checkbox.
2. Select an action in the **Action** text box.
3. Click the **Select Serialization Indexes** button. The **Serial Settings** dialog box displays (see figure below).

Note: Depending on the STAPL file format (Actel format or generic format) used, you will either see **Indexes** columns or **Actions** columns in the **Serial Settings** dialog box.

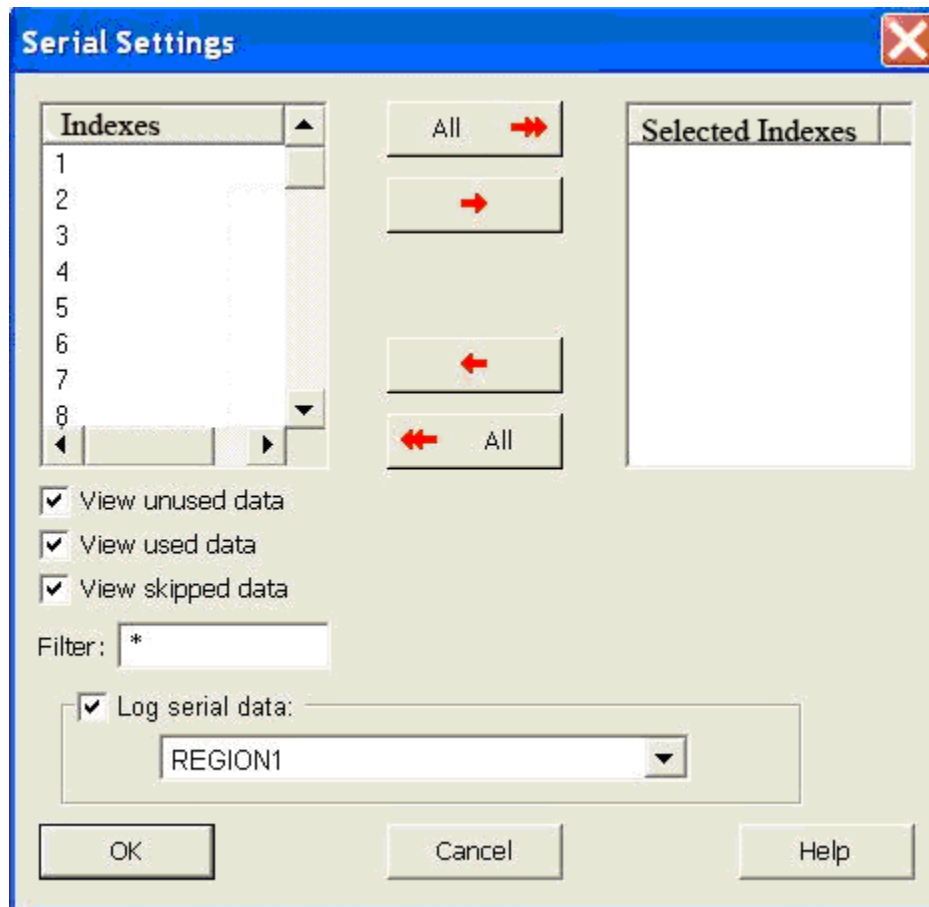


Figure 81 · Serial Settings Dialog Box

Note: Depending on your STAPL file, you would click the Select Serialization Action button (see figure below).

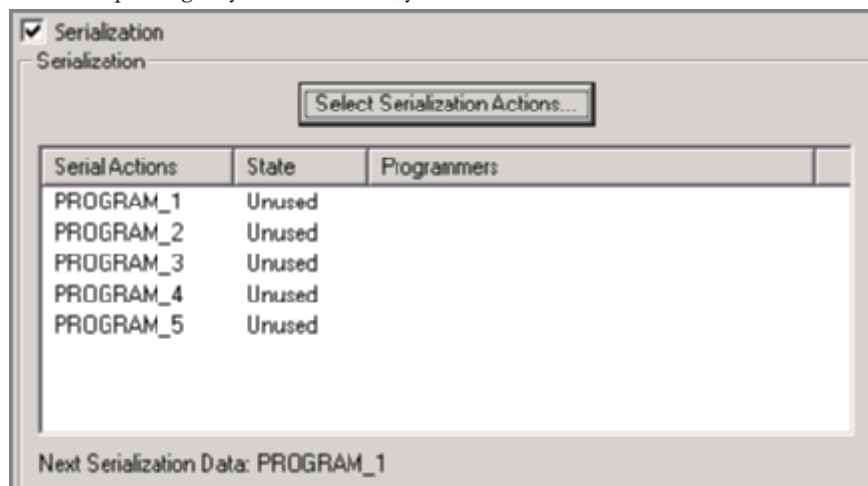


Figure 82 · Select Serialization Actions Button



Modifying Programming Settings in FlashPro with a PDB File

1. From the **Configuration** menu, choose **Load Programming File (PDB)**.
2. Select the PDB file and click **Open**, this loads the programming file.

See Also

Configuring security, FPGA, FlashROM and Embedded Flash Memory Block settings in FlashPro

Configuring Security

Configuring Security, FPGA and FlashROM Settings in FlashPro

1. From the Configuration menu, choose PDB Configuration. The Programming File Generator appears (see image below).

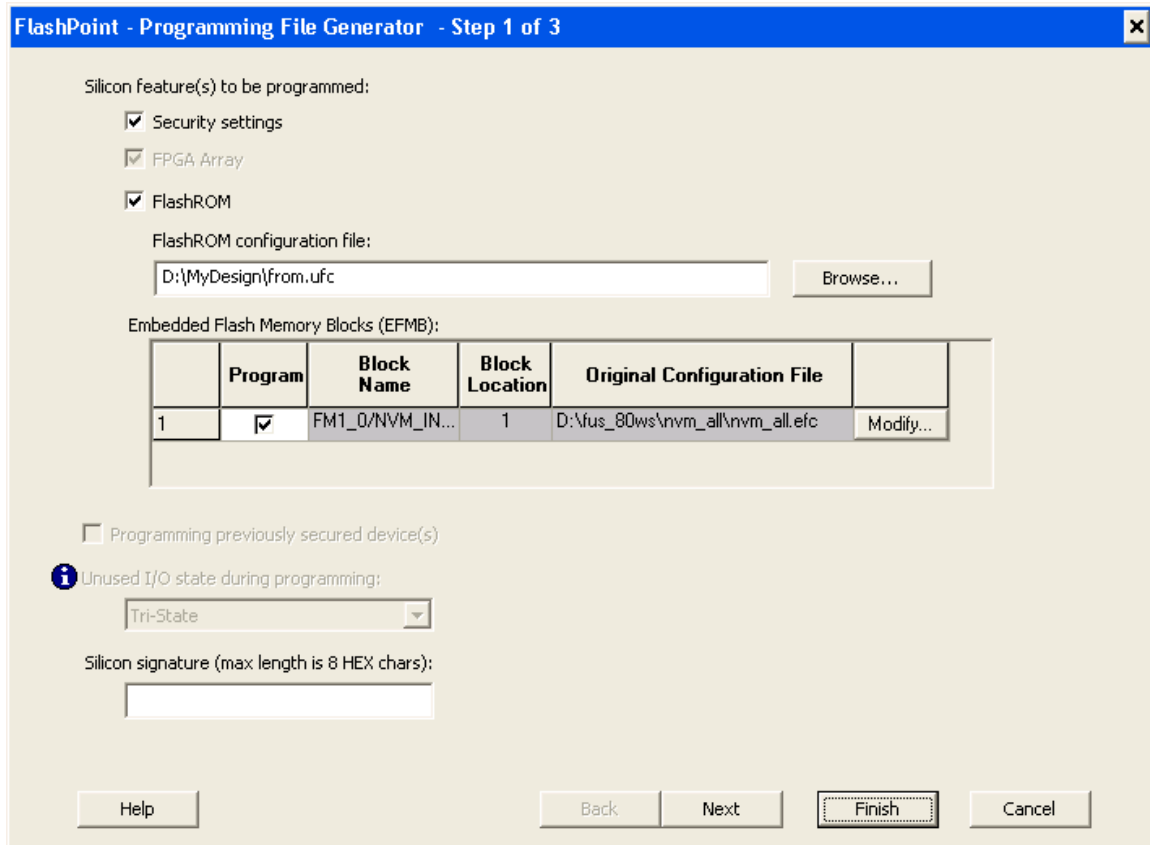


Figure 83 · Programming File Generator

2. Select the Silicon feature(s) you want to program:
 - [Security settings](#)
 - [FPGA Array](#)
 - [FlashROM](#)
 - [Embedded Flash Memory Block](#)
3. Check the **Programming previously secured device(s)** box if you are reprogramming a device that has been secured.

Because the IGLOO, Fusion, and ProASIC3 families enable you to program the Security Settings separately from the FPGA Array and/or FlashROM, you must indicate if the Security Settings were previously programmed into the target device. This requirement also applies when you generate programming files for reprogramming.
4. Enter the **silicon signature** (0-8 HEX characters).
5. Click **Next**.

Configuring Security Settings in FlashPro

To configure the security settings:

1. From the **Configuration** menu, choose **PDB Configuration**. The **Programming File Generator** appears (see image below).

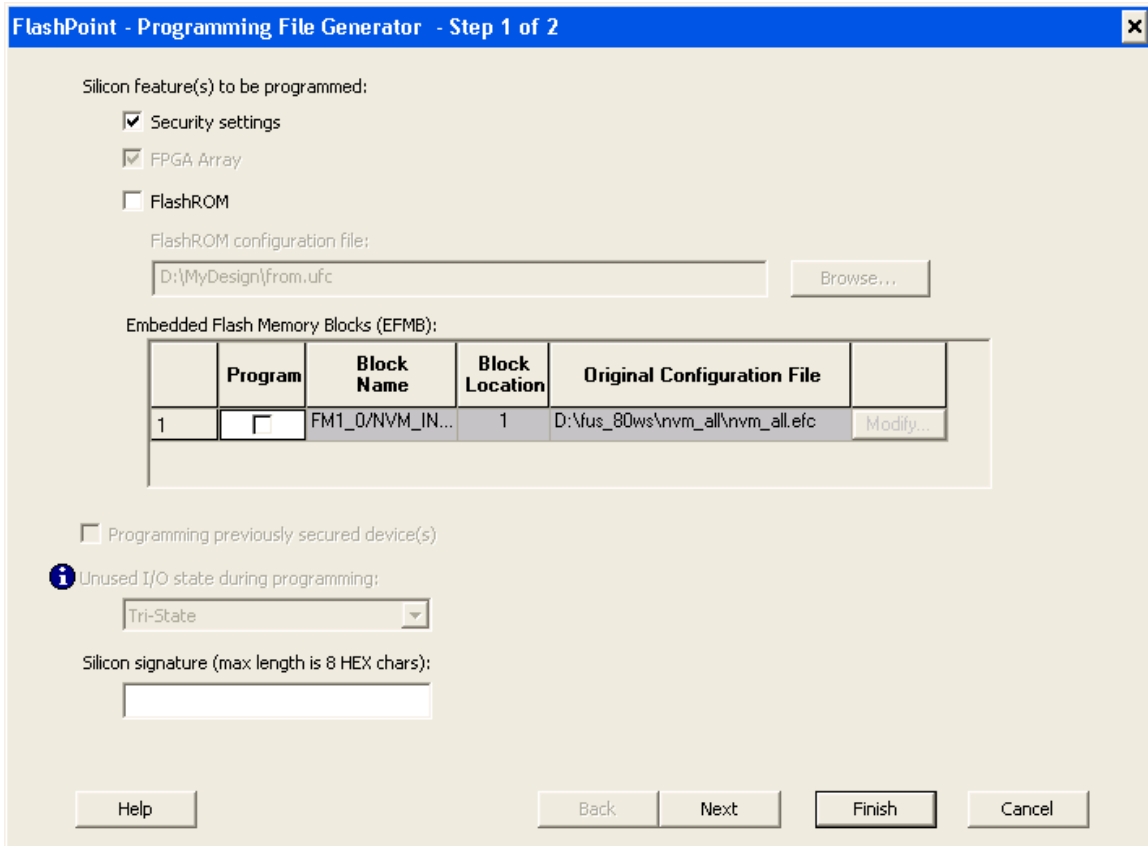


Figure 84 · Programming File Generator

2. Check the **Security Settings** checkbox and click **Next**. This brings up the **Security Settings** dialog box (see image below).

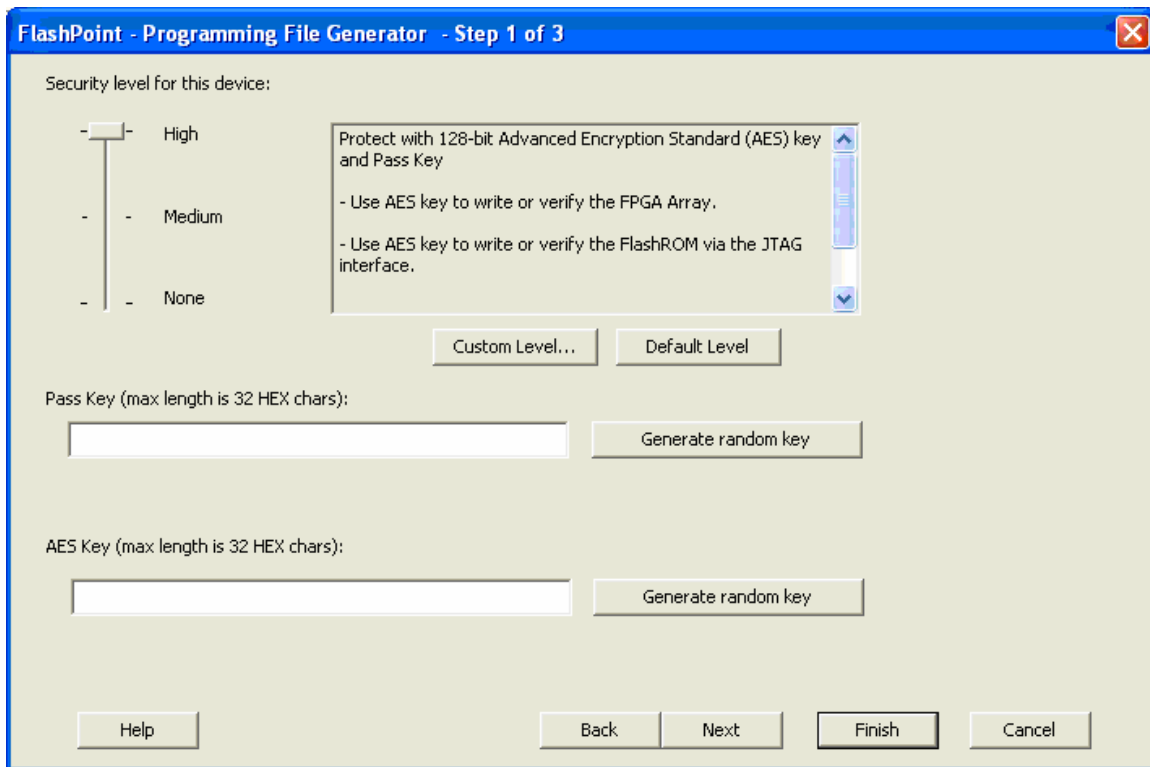


Figure 85 · Security Settings Dialog Box

3. Move the sliding bar to select the security level for FPGA, FlashROM, and EFMB (see table for a description of the security levels).

Table 7 · Security Levels Descriptions

Security Level	Security Option	Description
High	Protect with a 128-bit Advanced Encryption Standard (AES) key and a Pass Key	Access to the device is protected by an AES Key and the Pass Key. The Write and Verify operations of the FPGA Array use a 128-bit AES encrypted bitstream. From the JTAG interface, the Write and Verify operations of the FlashROM use a 128-bit AES encrypted bitstream. Read back of the FlashROM content via the JTAG interface is protected by the Pass Key. Read back of the FlashROM content is allowed from the FPGA Array.
Medium	Protect with Pass Key	The Write and Verify operations of the FPGA Array require a Pass Key. From the JTAG interface, the Read and Write operations on the FlashROM content require a Pass Key. You can Verify the FlashROM content via the JTAG interface without a Pass Key. Read back of the FlashROM content is allowed from the FPGA Array.
None	No Security	The Write and Verify operations of the FPGA Array do not require keys. The Read, Write, and Verify operations of the FlashROM content also do not require keys.

4. Enter the **Pass Key** and/ or the **AES Key** as appropriate. You can generate a random key by clicking the **Generate random key** button.
The **Pass Key** protects all the Security Settings for the FPGA Array and/or FlashROM.

The AES Key decrypts FPGA Array and/or FlashROM programming file content. Use the AES Key if you intend to program the device at an unsecured site or if you plan to update the design at a remote site in the future.

5. Click **Finish**.

You can also customize the security levels by clicking the [Custom Level](#) button.

Custom Security Settings

For advanced use, you can customize your security levels.

To set custom security levels:

1. Click the **Custom Level** button in the **Setup Security** page. The **Custom Security** dialog box appears (see figure below).

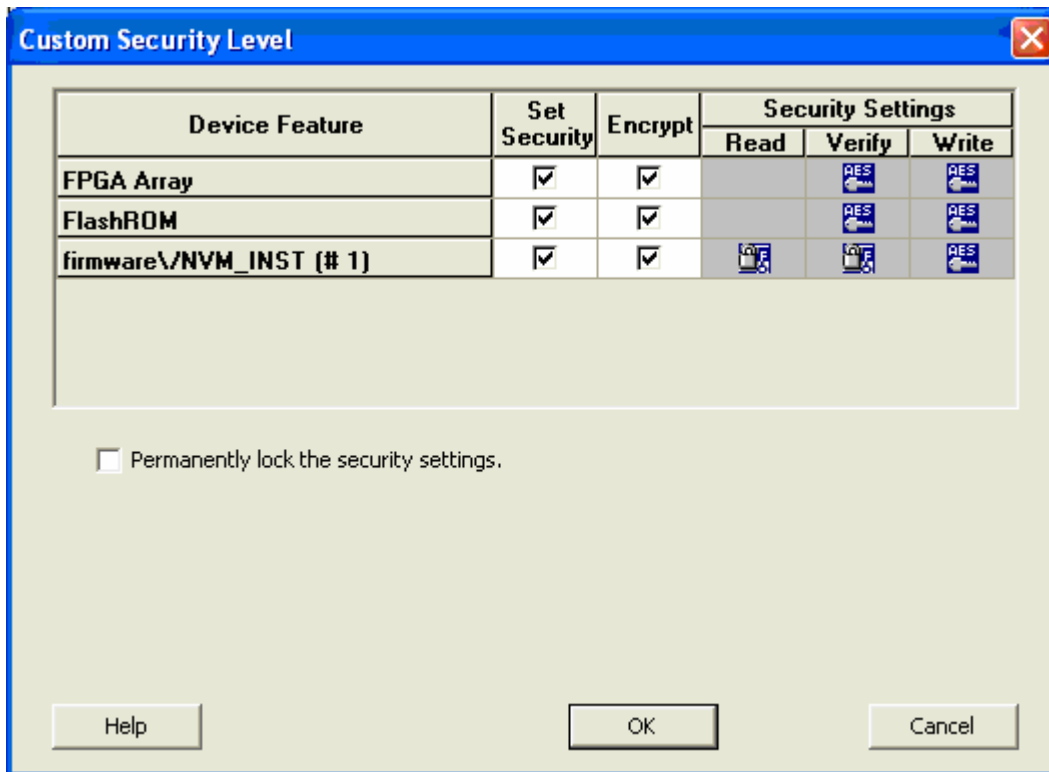


Figure 86 · Custom Security Level

2. Select the **FPGA Array Security**, the **FlashROM Security**, and **Embedded Flash Memory** block levels.

Note: The silicon features can have different Security Settings. See the tables below for a description of the custom security option levels for FPGA Array, FlashROM, and Embedded Flash Memory block.

Table 8 · FPGA Array

Security Option							Description															
Lock for both writing and verifying	<table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FPGA Array</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FPGA Array	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Allows writing/erasing and verification of the FPGA Array via the JTAG interface only with a valid Pass Key.
Device Feature	Set Security	Encrypt	Security Settings																			
			Read	Verify	Write																	
FPGA Array	<input checked="" type="checkbox"/>	<input type="checkbox"/>																				
Lock for writing	<table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FPGA Array</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FPGA Array	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Allows the writing/erasing of the FPGA Array only with a valid Pass Key. Verification is allowed without a valid Pass Key.
Device Feature	Set Security	Encrypt	Security Settings																			
			Read	Verify	Write																	
FPGA Array	<input checked="" type="checkbox"/>	<input type="checkbox"/>																				
Use the AES Key for both writing and verifying	<table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FPGA Array</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FPGA Array	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				Allows the writing/erasing and verification of the FPGA Array only with a valid AES Key via the JTAG interface. This configures the device to accept an encrypted bitstream for reprogramming and verification of the FPGA Array. Use this option if you intend to complete final programming at an unsecured site or if you plan to update the design at a remote site in the future. Accessing the device security settings requires a valid Pass Key.
Device Feature	Set Security	Encrypt	Security Settings																			
			Read	Verify	Write																	
FPGA Array	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>																				
Allow write and verify	<table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FPGA Array</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FPGA Array	<input type="checkbox"/>	<input type="checkbox"/>				Allows writing/erasing and verification of the FPGA Array with plain text bitstream and without requiring a Pass Key or an AES Key. Use this option when you develop your product in-house.
Device Feature	Set Security	Encrypt	Security Settings																			
			Read	Verify	Write																	
FPGA Array	<input type="checkbox"/>	<input type="checkbox"/>																				

Note: Note: The ProASIC3 family FPGA Array is always read protected regardless of the Pass Key or the AES Key protection.

Table 9 · FlashROM

Security Option							Description															
Lock for both reading and writing	<table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FlashROM</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FlashROM	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Allows the writing/erasing and reading of the FlashROM via the JTAG interface only with a valid Pass Key. Verification is allowed without a valid Pass Key.
Device Feature	Set Security	Encrypt	Security Settings																			
			Read	Verify	Write																	
FlashROM	<input checked="" type="checkbox"/>	<input type="checkbox"/>																				
Lock for writing							Allows the writing/erasing of the FlashROM via the JTAG															

Security Option						Description															
<table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FlashROM</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FlashROM	<input checked="" type="checkbox"/>	<input type="checkbox"/>				interface only with a valid Pass Key. Reading and verification is allowed without a valid Pass Key.
Device Feature	Set Security	Encrypt	Security Settings																		
			Read	Verify	Write																
FlashROM	<input checked="" type="checkbox"/>	<input type="checkbox"/>																			
<p>Use the AES Key for both writing and verifying</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FlashROM</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FlashROM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				Allows the writing/erasing and verification of the FlashROM via the JTAG interface only with a valid AES Key. This configures the device to accept an encrypted bitstream for reprogramming and verification of the FlashROM. Use this option if you complete final programming at an unsecured site or if you plan to update the design at a remote site in the future. Note: The bitstream that is read back from the FlashROM is always unencrypted (plain text).
Device Feature	Set Security	Encrypt	Security Settings																		
			Read	Verify	Write																
FlashROM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>																			
<p>Allow reading, writing, and verifying</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FlashROM</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FlashROM	<input type="checkbox"/>	<input type="checkbox"/>				Allows writing/erasing, reading and verification of the FlashROM content with a plain text bitstream and without requiring a valid Pass Key or an AES Key.
Device Feature	Set Security	Encrypt	Security Settings																		
			Read	Verify	Write																
FlashROM	<input type="checkbox"/>	<input type="checkbox"/>																			

Note: The FPGA Array can always read the FlashROM content regardless of these Security Settings.

Table 10 · Embedded Flash Memory Block

Security Option						Description															
<p>Lock for reading, verifying, and writing</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>firmware\NVM_INST (# 1)</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Allows the writing and reading of the Embedded Flash Memory Block via the JTAG interface only with a valid Pass Key. Verification accomplished by reading back and compare.
Device Feature	Set Security	Encrypt	Security Settings																		
			Read	Verify	Write																
firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>																			
<p>Lock for writing</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>firmware\NVM_INST (# 1)</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>						Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Allows the writing of the Embedded Flash Memory Block via the JTAG interface only with a valid Pass Key. Reading and verification is allowed without a valid Pass Key.
Device Feature	Set Security	Encrypt	Security Settings																		
			Read	Verify	Write																
firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>																			



Security Option						Description
Use AES Key for writing						Allows the writing of the Embedded Flash Memory Block via the JTAG interface only with a valid AES Key. This configures the device to accept an encrypted bitstream for reprogramming of the Embedded Flash Memory Block. Use this option if you complete final programming at an unsecured site or if you plan to update the design at a remote site in the future. The bitstream that is read back from the Embedded Flash Memory Block is always unencrypted (plain text), when a valid pass key is provided.
Device Feature	Set Security	Encrypt	Security Settings			
firmwareVNVM_INST (# 1)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				
Allow reading, writing, and verifying						Allows writing, reading and verification of the Embedded Flash Memory Block content with a plain text bitstream and without requiring a valid Pass Key or an AES Key.
Device Feature	Set Security	Encrypt	Security Settings			
firmwareVNVM_INST (# 1)	<input type="checkbox"/>	<input type="checkbox"/>				

- To make the Security Settings permanent, select the **Permanently lock the security settings** check box. This option prevents any future modifications of the Security Setting of the device. A Pass Key is not required if you use this option.

Note: When you make the Security Settings permanent, you can never reprogram the Silicon Signature. If you lock the write operation for the FPGA Array or the FlashROM, you can never reprogram the FPGA Array or the FlashROM, respectively. If you use an AES key, this key cannot be changed once you permanently lock the device.

Note: To use the Permanent FlashLock™ feature, select **Lock for both writing and verifying** for FPGA Array and **Lock for both reading and writing** for FlashROM and select the **Permanently lock the security settings** checkbox as shown in the figure below. This will make your device one-time-programmable.

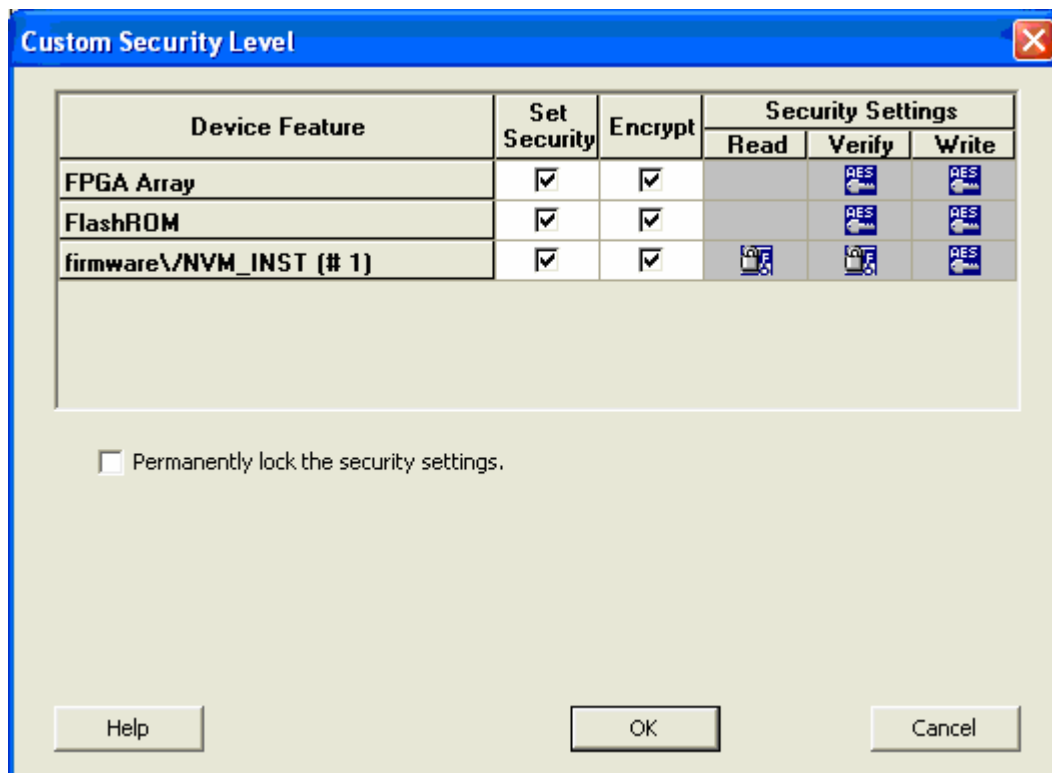


Figure 87 · Custom Security Level- Permanent Lock

4. Click the OK button. The Security Settings page appears with the Custom security setting information.

Configuring FPGA Array Settings in FlashPro

To configure the FlashROM settings:

1. From the Configuration menu, choose PDB Configuration. The Programming File Generator appears.
2. Check the FPGA Array checkbox and click Finish.



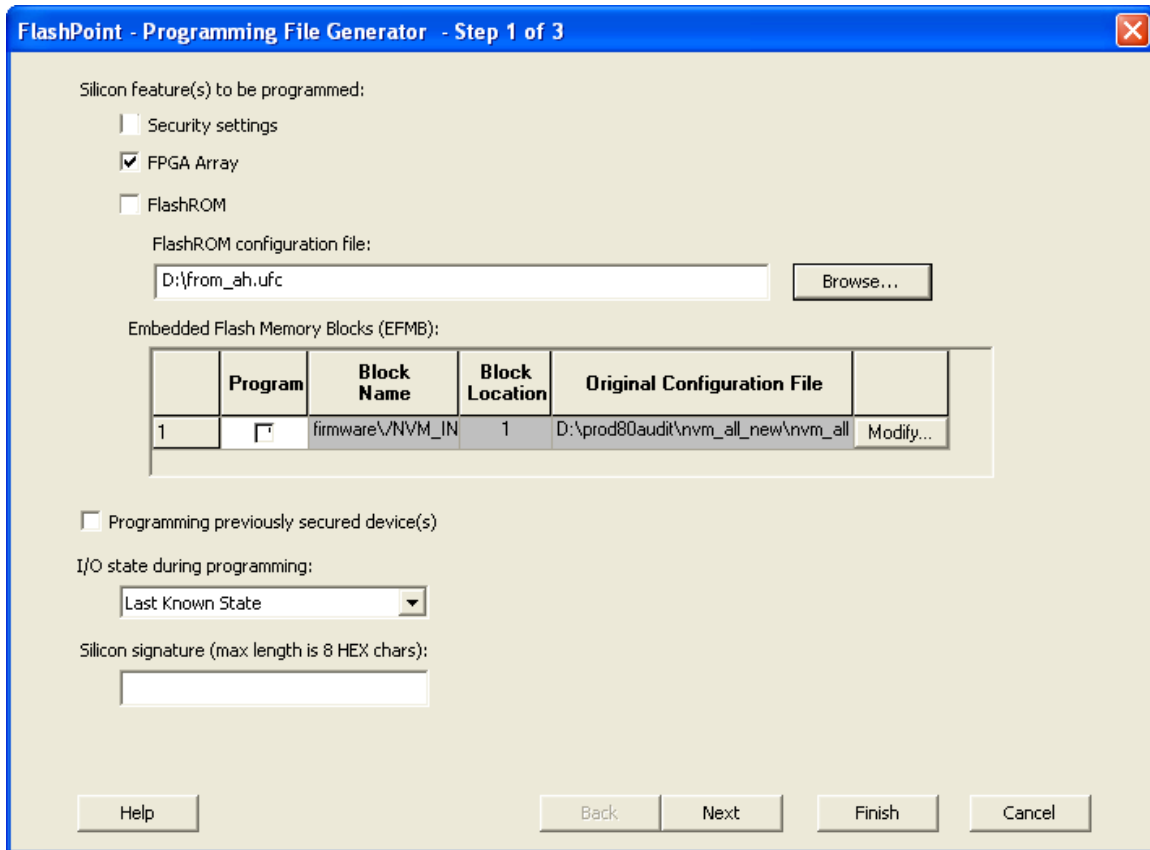


Figure 88 · Programming File Generator

Configuring FlashROM Settings in FlashPro

To configure the FlashROM settings:

1. From the **Configuration** menu, choose **PDB Configuration**. The **Programming File Generator** appears.
2. Check the **FlashROM** checkbox and click **Browse** to load a FlashROM configuration file. Click **Next**. This brings up the **FlashROM Settings** dialog box (see image below).

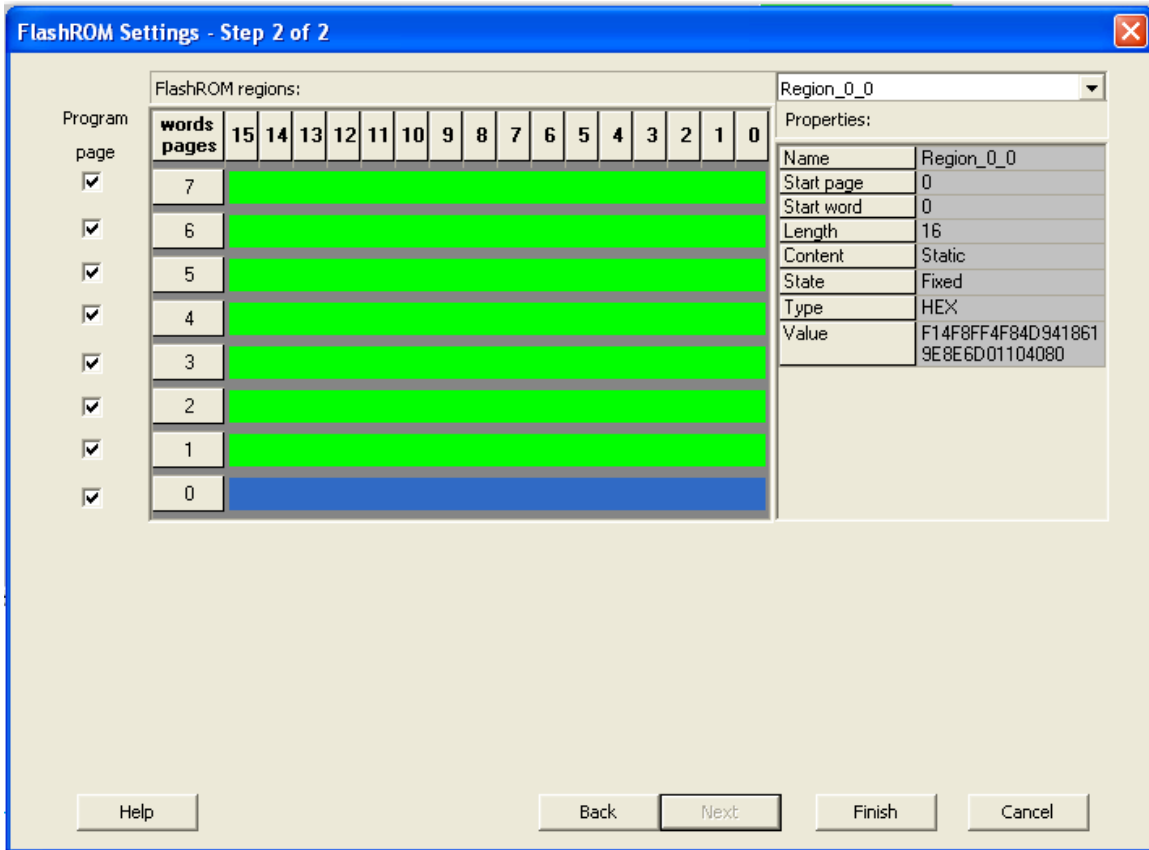


Figure 89 · FlashROM Settings Dialog Box

3. Select the FlashROM memory page that you want to program.
4. Enter the data value for the configured regions.
5. If you selected the region with a **Read From File**, specify the file location.
6. If you selected the **Auto Increment** region, specify the **Start** and **Max** values.
7. Enter the number of devices you want to program.
8. Click the **Target Programmer** button.

The **Select Programmer Type** dialog box appears (see figure below).

9. Click **Finish**. FlashPoint generates your programming file.

Note: You cannot change the FlashROM region configuration from FlashPoint. You can only change the configuration from the SmartGen FlashROM core generator.

Express Configuration

The express configuration feature in FlashPro allows you to set the security settings as well as FlashROM content without a design. This allows the production flow to be executed in parallel to the design effort if needed.

For example, you can pre-program the security settings with the **High Security** setting and serialize the device using FlashROM without the FPGA design in a secured programming environment. The FPGA Array and EFMB design can be programmed in unsecured programming environment using encrypted programming file.

Note: This feature is only available for IGLOO, Fusion, and ProASIC3 family devices.



AFS Programming

Introduction

FlashPro enables you to program security settings, FPGA Array, embedded flash memory blocks (FB), and FlashROM features for AFS device support. You can program these features separately using different programming files or you can combine them into one programming file.

Programming File Actions for AFS Devices

The STAPL files for AFS devices include actions targeted at one, two, or all four of the AFS programming features (FPGA Array and FlashROM, Security Settings, and Embedded Flash Memory Block). The combinations of the features you selected to target, results in different actions that are available in the STAPL file. See the following table for an illustration.

Table 11 · AFS Programming File Actions

AFS Programming Features	Features Selected											
FPGA Array	X				X			X	X		X	X
FlashROM		X				X		X		X	X	X
Security			X				X		X	X		X
Embedded Flash Memory Block (FB)				X	X	X	X				X	X
	STAPL Actions Available (correspond with Features Selected above)											
PROGRAM	X	X			X	X		X	X	X	X	X
VERIFY	X	X			X	X		X	X	X	X	X
ERASE	X	X			X	X		X	X	X	X	X
ERASE_ALL	X	X	X		X	X	X	X	X	X	X	X
DEVICE_INFO	X	X	X	X	X	X	X	X	X	X	X	X
READ_IDCODE	X	X	X	X	X	X	X	X	X	X	X	X
ERASE_FROM		X				X		X		X	X	X
PROGRAM_ARRAY	X				X			X	X		X	X
VERIFY_ARRAY	X				X			X	X		X	X

AFS Programming Features	Features Selected													
ENC_DATA_AUTHENTICATION	X				X			X	X		X	X		
PROGRAM_SECURITY			X				X		X	X		X	X	
ERASE_SECURITY			X				X		X	X		X	X	
VERIFY_SECURITY			X				X		X	X		X	X	
PROGRAM_FP				X	X	X	X					X	X	X
VERIFY_FP				X	X	X	X					X	X	X

Note: The ENC_DATA_AUTHENTICATION Action is only available when you choose encrypted programming.

STAPL Actions

See the table below for a list of all the actions for the STAPL file. **STAPL File Actions**

Action	Description
PROGRAM	Programs all selected ProASIC3 family features (FPGA Array, targeted FlashROM pages, security setting and silicon signature) (if provided).
VERIFY	Verifies all selected ProASIC3 family features (FPGA Array, targeted FlashROM pages, security setting and silicon signature) (if provided).
ERASE	Erases all selected ProASIC3 family features (FPGA Array, targeted FlashROM pages, security setting and silicon signature) (if provided).
ERASE ALL	Erases all features in the targeted ProASIC3 family device regardless of the features selected to generate the STAPL file.
DEVICE_INFO	Displays the IDCODE, Silicon Signature, the design name, the checksum, and device security settings and programming environment information programmed into the device.
READ_IDCODE	Reads the device ID code from the device.
ERASE_FROM	Erases only the targeted FlashROM pages, not the entire FlashROM.
PROGRAM_FROM	Programs only the targeted FlashROM pages.
VERIFY_FROM	Verifies only the targeted FlashROM pages.
PROGRAM_ARRAY	Programs the FPGA Array and Silicon Signature (if applicable) into the ProASIC3/E device.
VERIFY_ARRAY	Verifies the FPGA Array and Silicon Signature (if applicable) into the ProASIC3/E device.



Action	Description
ERASE_ARRAY	Erases the FPGA Array and Silicon Signature (if provided).
PROGRAM_SECURITY	Programs only the Security Settings.
ERASE_SECURITY	Erases only the Security Settings.

Options available in STAPL Actions

The table below shows the available actions in the STAPL file. STAPL File Actions

Action	Description
PROGRAM	When you target the Security Setting, you have the option of not erasing and programming the Security Setting by deselecting the following 2 procedures before executing the action. - SET_ERASE_SEC, - DO_PROGRAM_SECURITY. When you perform encrypted programming, you have the option of skipping the data authentication before programming by deselecting the DO_ENC_AUTHENTICATION procedure before executing the action.
ERASE	When you target the Security Setting, you have the option of not erasing the Security Setting by deselecting the SET_ERASE_SEC procedures before executing the action.
PROGRAM_ARRAY	When you perform encrypted programming, you have the option of skipping the data authentication before programming by deselecting the DO_ENC_AUTHENTICATION procedure before executing the action.

Note: The DO_ENC_AUTHENTICATION procedure prevents you from proceeding with encrypted programming with incorrect data due to corruption or an operator error. If incorrect data is detected during encrypted programming, the device will not be functional after programming.

ProASIC3 Family Programming

Introduction

ProASIC3 family devices support the following features: security settings, FPGA Array, and FlashROM. You can program these features separately or together using different programming files or by using one programming file.

Programming File Actions for ProASIC3 Family Devices

The STAPL files for ProASIC3, excluding ProASIC3L, devices include actions targeted at one, two, or all three of the ProASIC3 features (FPGA Array and FlashROM and Security Settings). The combinations of the features you selected to target, results in different actions that are available in the STAPL file. See the following table for an illustration.

Table 12 · ProASIC3 Family Devices Programming Actions

ProASIC3/E Features	Features Selected									
FPGA Array	X				X	X		X	X	
FlashROM		X			X		X	X		X
Security			X			X	X		X	X
Embedded Flash Memory Block (Fusion only)				X				x	x	x
	STAPL Actions Available (correspond with Features Selected above)									
PROGRAM	X	X			X	X	X	X	X	X
VERIFY	X	X			X	X	X	X	X	X
ERASE	X	X			X	X	X	X	X	X
ERASE_ALL	X	X	X	X	X	X	X	X	X	X
DEVICE_INFO	X	X	X	X	X	X	X	X	X	X
READ_IDCODE	X	X	X	X	X	X	X	X	X	X
ERASE_FROM		X			X		X	X		X
PROGRAM_FROM										
VERIFY_FROM		X			X		X	X		X
ERASE_ARRAY	X				X	X		X	X	
VERIFY_ARRAY	X				X	X		X	X	

ProASIC3/E Features	Features Selected									
ENC_DATA_AUTHENTICATION	X				X	X		X	X	
PROGRAM_SECURITY			X			X	X		X	X
ERASE_SECURITY			X			X	X		X	X
PROGRAM_NVM (Fusion only)				X				X	X	X
VERIFY_NVM (Fusion only)				X				X	X	X

Note: The ENC_DATA_AUTHENTICATION Action is only available when you choose encrypted programming.

Programming Actions

See the table below for a list of all the actions for the programming file. **Programming File Actions**

Action	Description
PROGRAM	Programs all selected ProASIC3 family features (FPGA Array, targeted FlashROM pages, security setting and silicon signature) (if provided).
VERIFY	Verifies all selected ProASIC3/E features (FPGA Array, targeted FlashROM pages, security setting and silicon signature) (if provided).
ERASE	Erases all selected ProASIC3 family features (FPGA Array, targeted FlashROM pages, security setting and silicon signature) (if provided).
ERASE_ALL	Erases all features in the targeted ProASIC3 family device regardless of the features selected to generate the STAPL file.
DEVICE_INFO	Displays the IDCODE, Silicon Signature, the design name, the checksum, and device security settings and programming environment information programmed into the device.
READ_IDCODE	Reads the device ID code from the device.
ERASE_FROM	Erases only the targeted FlashROM pages, not the entire FlashROM.
PROGRAM_FROM	Programs only the targeted FlashROM pages.
VERIFY_FROM	Verifies only the targeted FlashROM pages.
PROGRAM_ARRAY	Programs the FPGA Array and Silicon Signature (if applicable) into the ProASIC3 device.
VERIFY_ARRAY	Verifies the FPGA Array and Silicon Signature (if applicable) into the ProASIC3 device.
ERASE_ARRAY	Erases the FPGA Array and Silicon Signature (if provided).
PROGRAM_SECURITY	Programs only the Security Settings.
ERASE_SECURITY	Erases only the Security Settings.



Note: FIX_INT_ARRAYS: This function is only applicable to STAPL file only. Depending on the STAPL player implementation, the indexing of an integer array may start from different direction. The STAPL standard did not clearly specify how it should be implemented. The FIX_INT_ARRAYS function detects the indexing implemented by the STAPL player and flips the content of the integer array if needed.

Note: UNLOCK_UKEY: This function unlocks a secured device if it is locked by FlashLock.

Options Available in Programming Actions

The table below shows the available actions in the programming file. **Programming File Actions**

Action	Description
PROGRAM	When you target the Security Setting, you have the option of not erasing and programming the Security Setting by deselecting the following 2 procedures before executing the action. - SET_ERASE_SEC, - DO_PROGRAM_SECURITY. When you perform encrypted programming, you have the option of skipping the data authentication before programming by deselecting the DO_ENC_AUTHENTICATION procedure before executing the action.
ERASE	When you target the Security Setting, you have the option of not erasing the Security Setting by deselecting the SET_ERASE_SEC procedures before executing the action.
PROGRAM_ARRAY	When you perform encrypted programming, you have the option of skipping the data authentication before programming by deselecting the DO_ENC_AUTHENTICATION procedure before executing the action.

Note: The DO_ENC_AUTHENTICATION procedure prevents you from proceeding with encrypted programming with incorrect data due to corruption or an operator error. If incorrect data is detected during encrypted programming, the device will not be functional after programming.

ProASIC^{PLUS} and ProASIC Families Programming Introduction

ProASIC^{PLUS} and ProASIC family devices support security settings and FPGA Array. You can program these features together using one programming file. The table below shows the STAPL actions for the FPGA Array only. **FPGA Array Only STAPL File**

Action	Description
PROGRAM	Checks the security status of the device. If you are programming the security key, you may select to program only the FPGA Array by unselecting the PROGRAM_SECURITY procedure. If the device is programmed with the security key, then this command returns with Read inhibit:1 Write inhibit:1. If the security key is not present, the values are Read inhibit:0 Write inhibit:0.
ERASE_ARRAY	Erases the device.
READ_IDCODE	Reads the device ID code.
VERIFY	Verifies whether the device was programmed with the loaded STAPL file. Can be used to ensure that the bitstream programmed in the device is the same as the original STAPL file. If you load the wrong STAPL file, Exit 11 appears in the log window. A successful operation results in Exit 0.
PROGRAM	Programs the device.
DEVICE_INFO	Displays the serial number of the device, the Design Name that is programmed into the device, and the checksum that is programmed into the device.

Generating Programming Files

Generate a Programming File

FlashPoint enables you to program security settings, FPGA Array, and FlashROM features for IGLOO, Fusion, ProASIC3, and ProASIC family devices. You can program these features separately using different programming files or you can combine them into one programming file. Each feature is listed as a silicon feature in the GUI.

You can generate a programming file with one, two, or all of the silicon features from the **Programming File Generator** first page.

To generate a programming file:

1. Select the **Silicon feature(s)** you want to program.
 - [Security settings](#)
 - [FPGA Array](#)
 - [FlashROM](#)

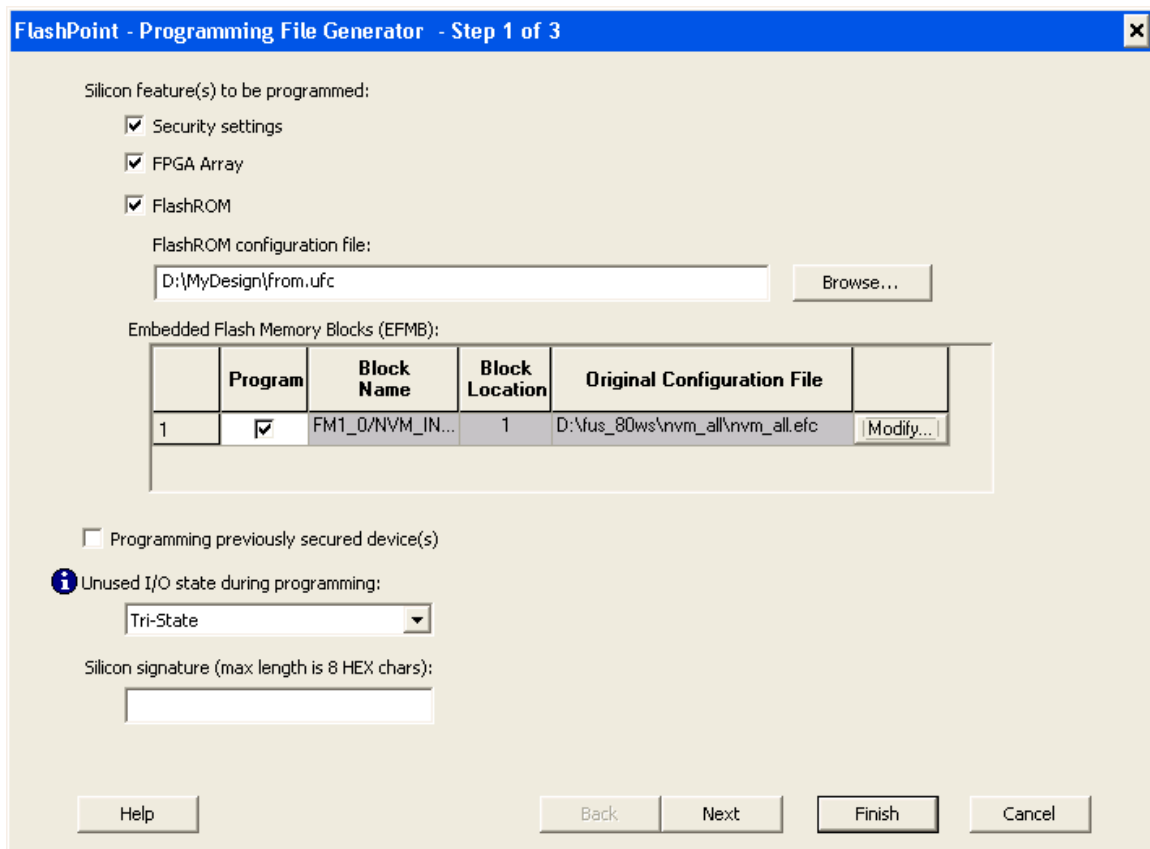


Figure 90 · Programming File Generator – Step 1 of 2

2. Click the **Programming previously secured device(s)** check box if you are reprogramming a device that has been secured.

Because the IGLOO, Fusion, ProASIC3, and ProASIC families enable you to program the Security Settings separately from the FPGA Array and/or FlashROM, you must indicate if the Security Settings were previously

programmed into the target device. This requirement also applies when you generate programming files for reprogramming.

- Specify the **Unused I/O state during programming** (Tri-State, Last Known State, High, or Low) in the drop down menu. See the table below for an explanation of these settings in the programming mode.

Note: Unused I/O state during programming only applies to the unused I/Os; to modify the I/O state during programming for used I/Os, use the I/O Attribute Editor in the MultiView Navigator. For more information about changing the used I/O state during programming, see the [MultiView Navigator online help](#).

Table 13 · I/O Settings in Programming Mode

Setting	Description
Tri-State	I/Os are tri-stated
Last Known State	I/Os are set to this state prior to entering the programming mode
High	I/Os are set to drive out logic High
Low	I/Os are set to drive out logic Low

- Enter the silicon signature (0-8 HEX characters). See [Silicon Signature](#) for more information.
- Depending upon the Silicon features you selected, click **Next** or **Finish**.

If you click **Next**, follow the instructions in the appropriate dialog box. If you click **Finish**, the **Save As** dialog box appears. From the **Save As** dialog box, choose [STAPL file](#), [SVF file](#), [PDB file](#), [1532 file](#), or multiple files.

Generate a Programming File for CoreMP7/Cortex-M1 Device Support

FlashPoint enables you to program FPGA Array and FlashROM features for CoreMP7/Cortex-M1 devices. You can program these features separately using different programming files or you can combine them into one programming file. Each feature is listed as a silicon feature in the GUI. You can generate a programming file with one, two, or all of the silicon features from the Programming File Generator first page. For CoreMP7/Cortex-M1 device support, you cannot select your own security settings. The generated programming file always has the encrypted FPGA Array content. The programming file generation is the same as the ProASIC3 family devices.

To generate a programming file:

- Select the Silicon feature(s) you want to program.

[FPGA Array](#)

[FlashROM](#)



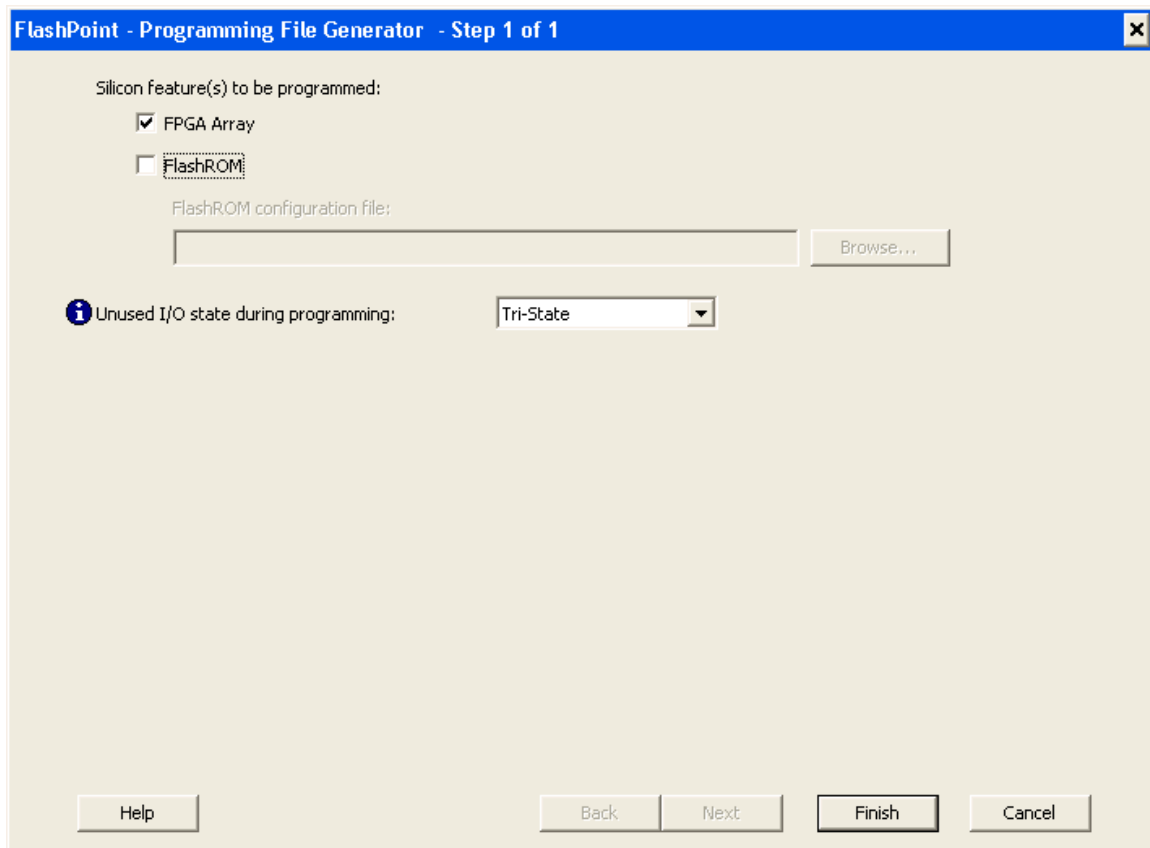


Figure 91 · FlashPoint- Programming File Generator for ProASIC3- Step 1 of 2

2. Specify the Unused I/O state during programming (Tri-State, Last Known State , High, or Low) in the drop down menu. See the table from [Generate a programming file](#) for an explanation of the settings in the programming mode.
3. Click **Next** or **Finished** depending on the silicon features you selected.

If you click **Next**, follow the instructions in the appropriate dialog box. If you click **Finish**, the **Save As** dialog box appears. From the **Save As** dialog box, choose [STAPL file](#), [SVF file](#), [PDB file](#), [1532 file](#), or multiple files.

CoreMP7/Cortex-M1 Device Security

CoreMP7/Cortex-M1 devices are shipped with the following security enabled:

- FPGA Array enabled for AES encrypted programming and verification.
- FlashROM enabled for plain text read and write.

You cannot select your own security settings. The generated programming file includes the encrypted FPGA Array content. **Programming FlashROM and FPGA Array**

For CoreMP7/Cortex-M1 device support, the programming generation for [FlashROM](#) and [FPGA Array](#) is the same as the programming generation for ProASIC3 and ProASIC family devices.

Generate a Programming File for AFS Device Support

FlashPoint enables you to program Security Settings, FPGA Array, Embedded Flash Memory Blocks, and FlashROM features for AFS device support. You can program these features separately using different programming files or you can combine them into one programming file. Each feature is listed as a silicon feature in the GUI. You can generate a programming file with one, two, or all of the silicon features from the **Programming File Generator** first page.

AFS Programming

In addition to FPGA Array, FlashROM and security setting, the Fusion devices provide Embedded Flash Memory Blocks (FB) for both Analog configuration initialization and regular memory storage. Depending on the targeted AFS device, you may have one, two, or four FBs available to you. FlashPoint enables you to initialize the FB Instance(s) as specified in SmartGen. (Please refer to the [SmartGen User's Guide](#) for details).

To generate a programming file:

1. Select the Silicon feature(s) you want to program.

[Security Settings](#)

[FPGA Array](#)

[FlashROM](#)

[Embedded Flash Memory Block](#)

FlashPoint - Programming File Generator - Step 1 of 3

Silicon feature(s) to be programmed:

- Security settings
- FPGA Array
- FlashROM

FlashROM configuration file:

D:\MyDesign\from.ufc Browse...

Embedded Flash Memory Blocks (EFMB):

	Program	Block Name	Block Location	Original Configuration File	
1	<input checked="" type="checkbox"/>	FM1_0\NVM_IN...	1	D:\fus_80ws\nvm_all\nvm_all.efc	Modify...

Programming previously secured device(s)

i Unused I/O state during programming:

Tri-State

Silicon signature (max length is 8 HEX chars):

Help Back Next Finish Cancel

Figure 92 · FlashPoint- Programming File Generator for AFS



Note: Check the check box in the Program column to enable block modification.

2. Check the **Programming previously secured devices(s)** box if you want to program previously secured devices.
3. Specify the Unused I/O state during programming (Tri-State, Last Known State, High, or Low) in the drop down menu. See the table from [Generate a programming file](#) for an explanation of the settings in the programming mode.
4. Enter the **Silicon signature**.
5. Depending upon the Silicon features you selected, click **Finish** or **Next**.

If you click **Next**, follow the instructions in the appropriate dialog box. If you click **Finish**, the **Save As** dialog box appears. From the **Save As** dialog box, choose [PDB file](#), [STAPL file](#), [SVF file](#), [1532 file](#), or multiple files.

Programming Security Settings, FlashROM, and FPGA Array

For AFS device support, the programming generation for [Security Settings](#), [FlashROM](#) and [FPGA Array](#) is the same as the programming generation for ProASIC3 family devices.

Generate a Programming File for Serialization Support in In House Programming (IHP)

FlashPoint allows you to program security settings, FPGA Array, and FlashROM features for IGLOO, Fusion, ProASIC3, and ProASIC family devices. You can program these features separately using different programming files or you can combine them into one programming file. Each feature is listed as a silicon feature in the GUI.

SVF Serialization Support in IHP

In addition to FPGA Array, FlashROM, and security setting, FlashPoint supports generating SVF files with serialization support in IHP.

To generate SVF with serialization support:

1. Select the **Silicon feature(s)** you want to program.
 - [Security settings](#)
 - [FPGA Array](#)
 - [FlashROM](#)
 - Programming Embedded Flash Memory Block
2. Import the UFC file which contains serialization data to FlashROM. See figure below. Click **Next**.

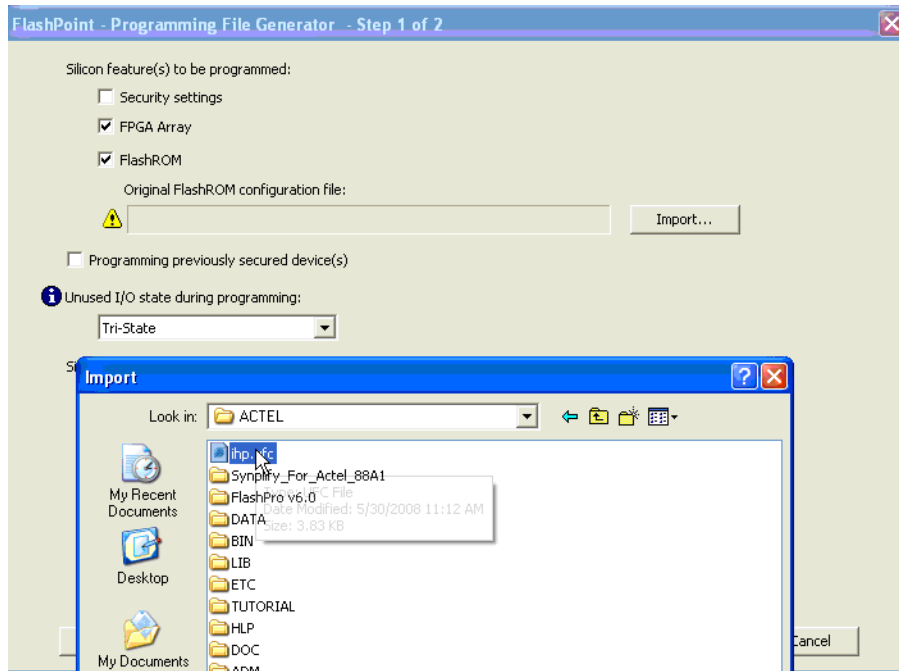


Figure 93 · Generate SVF File for Serialization Support in IHP

3. Type in the number of devices to program. See figure below.



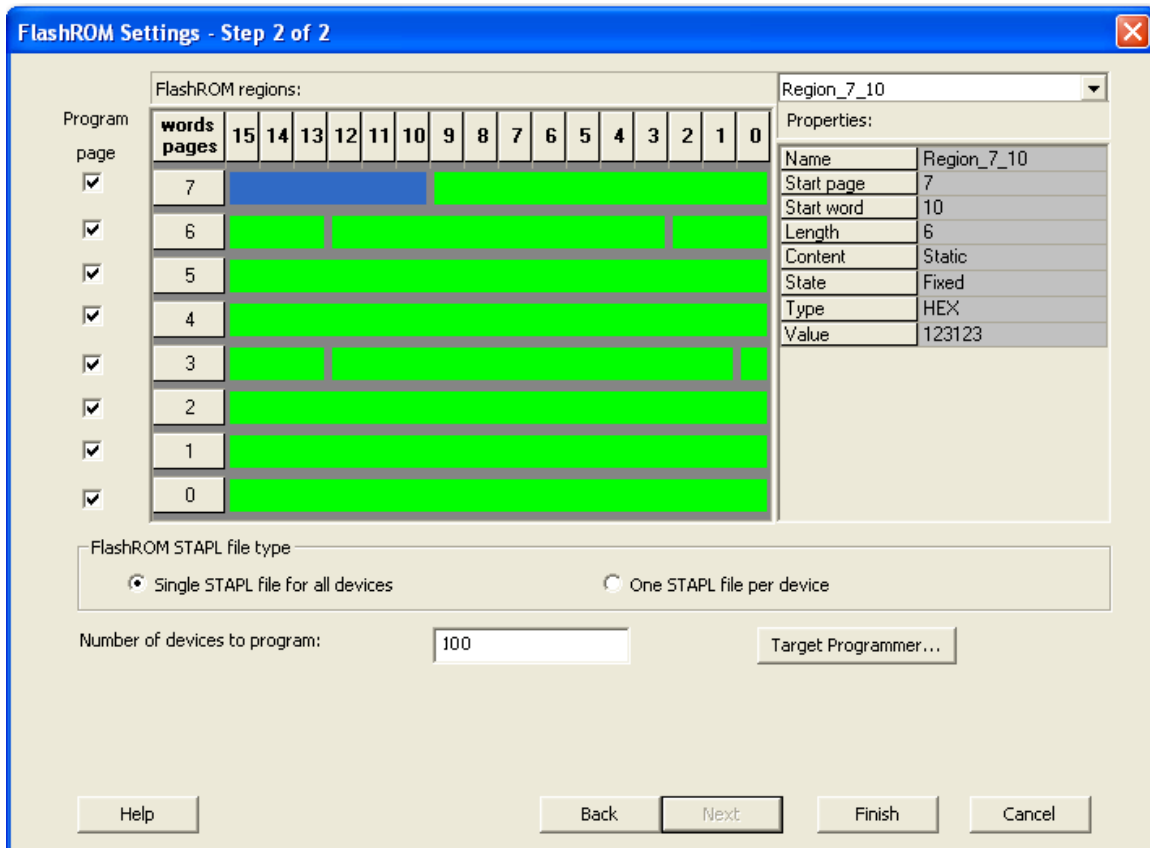


Figure 94 · Type Number of Devices

4. Click **Target Programmer** and select **Actel IHP**.

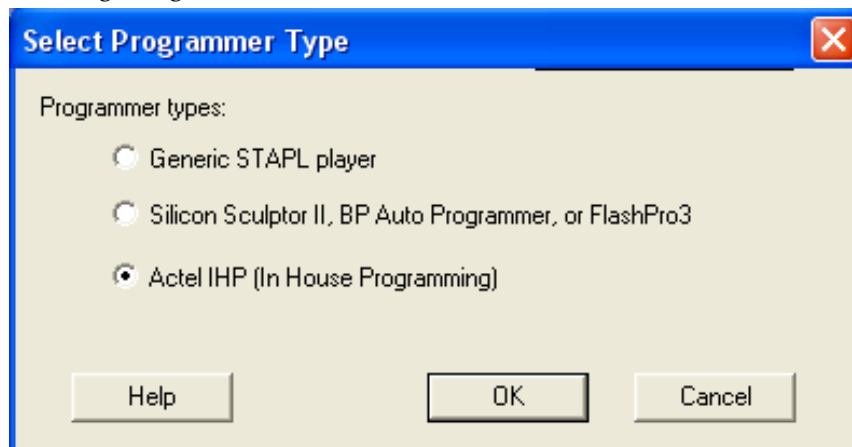


Figure 95 · Select Actel IHP

5. Click **OK**. A **Generate Programming Files** window displays. See figure below. Select **Serial Vector Files (*.svf)**.

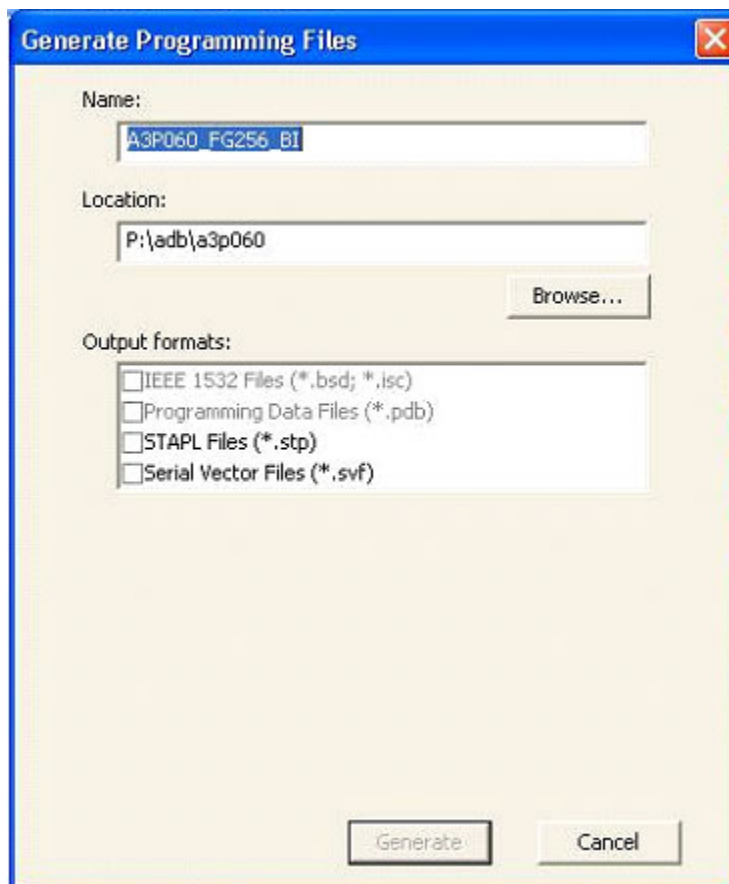


Figure 96 · Select Serial Vector Files

6. Click **Generate**. An Actel specific SVF file will be generated with a corresponding serialization data file.
Note: Generated SVF files will only work with IHP.

Programming Embedded Flash Memory Block

For more information about the Embedded Flash Memory Block, see the [Flash Memory System Builder](#) online help.

To program the Embedded Flash Memory Block:

1. Check the **Program** box to enable Embedded Flash Memory Block modification.



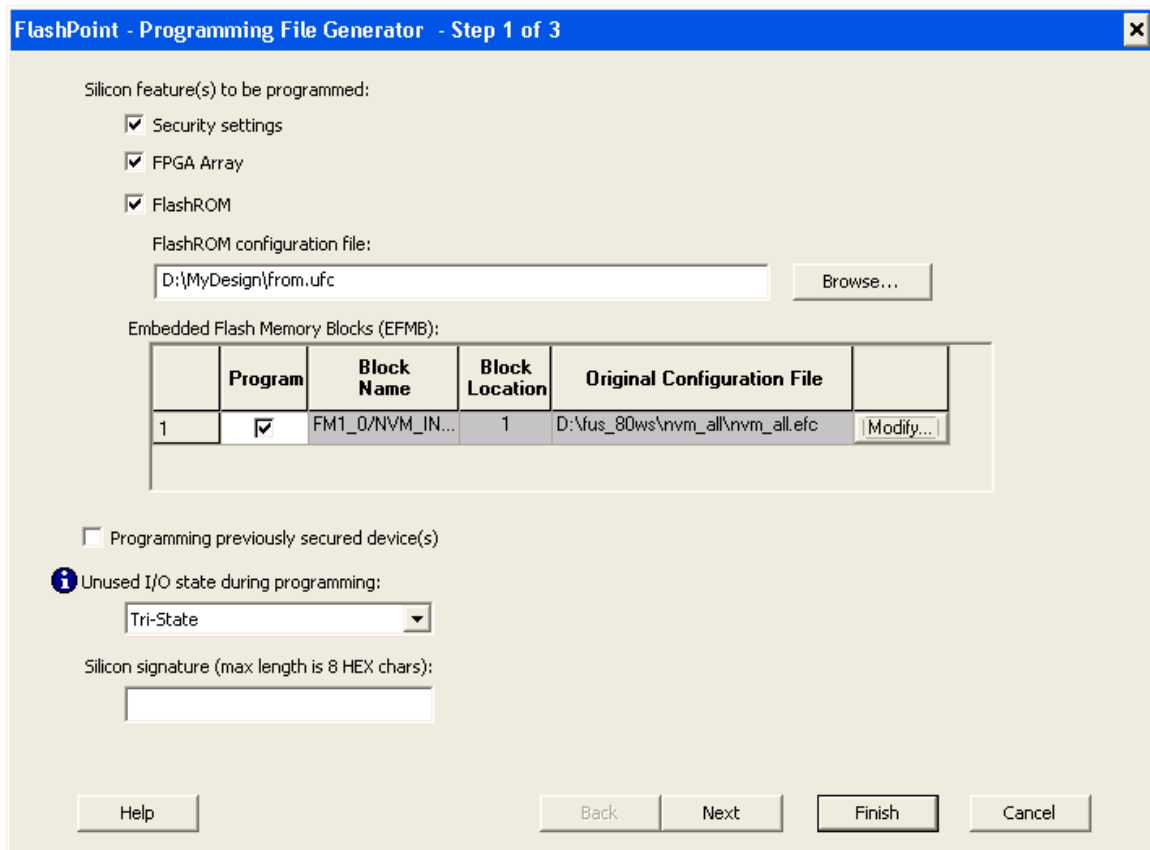


Figure 97 · Program File Generator first page

2. Click the **Modify** button to import Embedded Flash Memory Block configuration and memory content. The **Modify Embedded Flash Memory Block** dialog box appears.

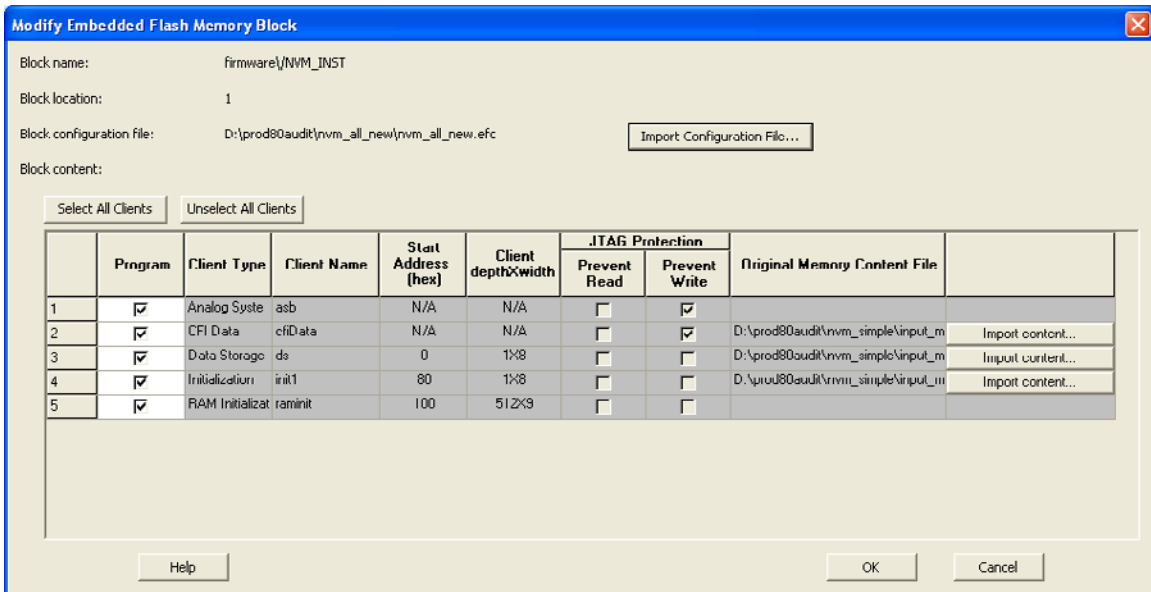


Figure 98 · Modify Embedded Flash Memory Block Content Dialog Box

3. Click the **Import Configuration File** button to import the Embedded Flash Memory Block configuration and memory content from the EFC file. This will populate the client table below. All clients that belong to this block will be selected by default.
4. Click the **Import content** button if you want to change the client memory content.
5. Click **OK**.

Note: FlashPoint audits original configuration and memory content files and warns the user if the files cannot be located or if they have been updated.

Programming the FPGA Array

You can program the FPGA Array by selecting the silicon feature **FPGA Array** in the Generate Programming File page and clicking **OK**.

See [Generate a programming file](#) for more information.

Programming the FlashROM

You can program selected memory pages and specify the region values of the FlashROM.

To program FlashROM:

1. Select **FlashROM** from the **Generate Programming File** page.
2. Enter the location of the FlashROM configuration file. The **FlashROM Settings** page appears (see figure below).



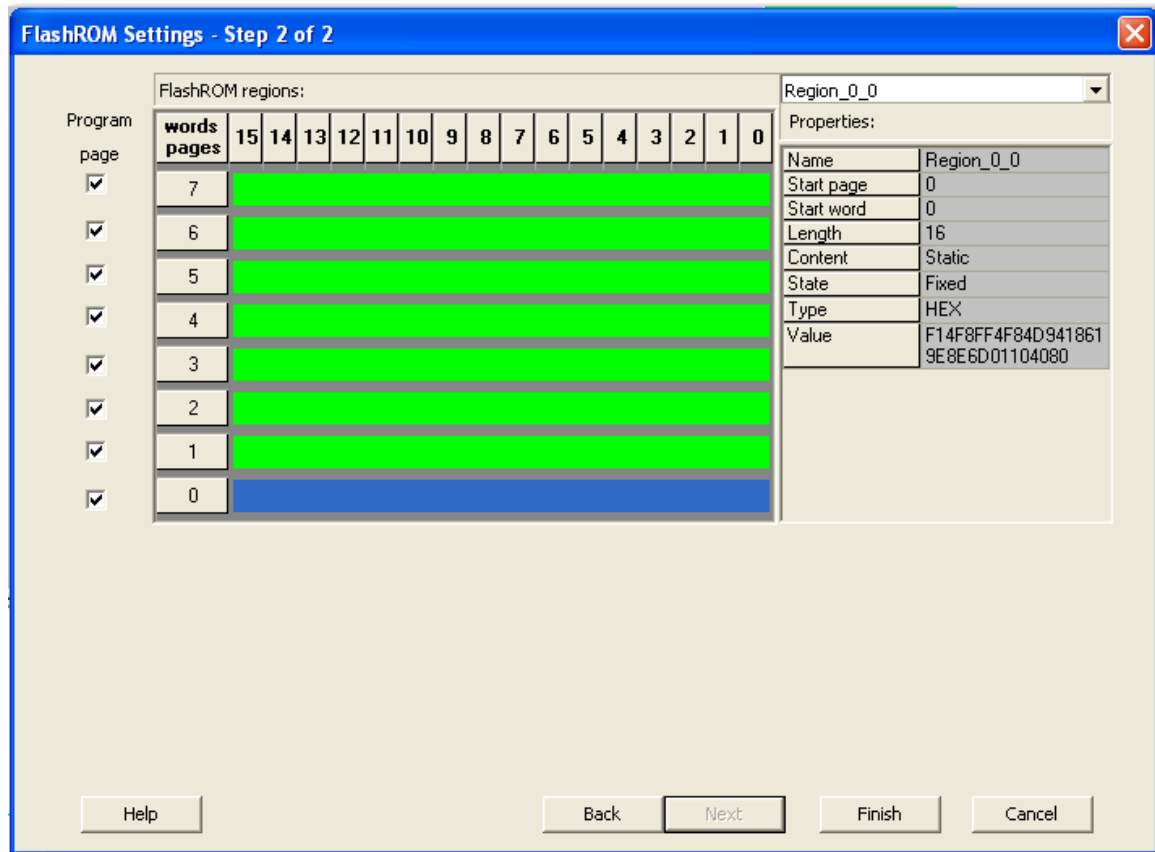


Figure 99 · FlashROM Settings

3. Select the FlashROM memory page that you want to program.
4. Enter the data value for the configured regions.
5. If you selected the region with a **Read From File**, specify the file location.
6. If you selected the **Auto Increment** region, specify the **Start** and **Max** values.

Click **Finish**.

FlashPoint generates your programming file.

Note: You cannot change the FlashROM region configuration from FlashPoint. You can only change the configuration from the SmartGen FlashROM core generator.

For more information, see the SmartGen online help.

Silicon Signature

With Designer tools, you can use the silicon signature to identify and track Actel designs and devices. When you generate a programming file, you can specify a unique silicon signature to program into the device. This signature is stored in the design database and in the programming file, and programmed into the device during programming.

The silicon signature is accessible through the USERCODE JTAG instruction.

Note: If you set the security level to high, medium, or custom, you must program the silicon signature along with the Security Setting. If you have already programmed the Security Setting into the target device, you cannot reprogram the silicon signature without reprogramming the Security Setting.

The previously programmed silicon signature will be erased if:

- You have already programmed the silicon signature and
- You are programming the security settings, but you do not have an entry in the silicon signature field

Programming Security Settings

FlashPoint allows you to set a security level of high, medium, or none.

To program Security Settings on the device:

1. If you choose to program Security Settings on the device from the **Generate Programming File** page, the wizard takes you to the **Security Settings** page (see figure below).
2. Move the sliding bar to select the security level for FPGA and FlashROM (see table for a description of the security levels).

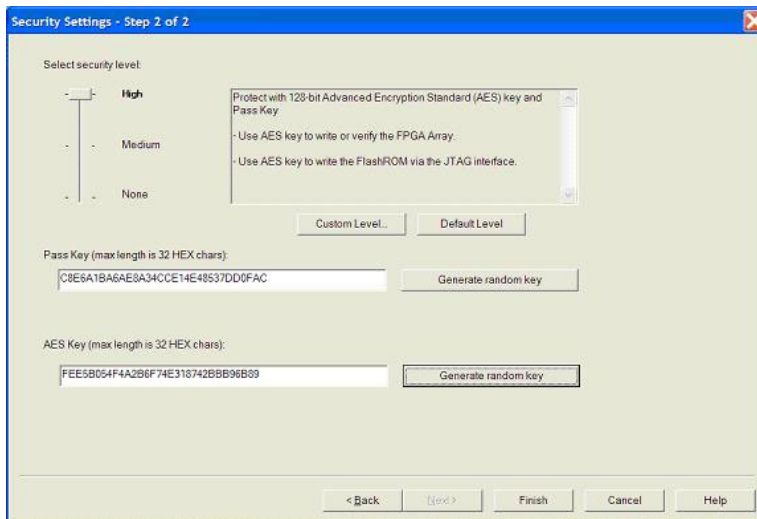


Figure 100 · Security Settings Page



Figure 101 · FPGA and FlashROM Security Levels

Security Level	Security Option	Description
High	Protect with a 128-bit Advanced Encryption Standard (AES) key and a Pass Key	<p>Access to the device is protected by an AES Key and the Pass Key.</p> <p>The Write and Verify operations of the FPGA Array use a 128-bit AES encrypted bitstream.</p> <p>From the JTAG interface, the Write and Verify operations of the FlashROM use a 128-bit AES encrypted bitstream. Read back of the FlashROM content via the JTAG interface is protected by the Pass Key.</p> <p>Read back of the FlashROM content is allowed from the FPGA Array.</p>
Medium	Protect with Pass Key	<p>The Write and Verify operations of the FPGA Array require a Pass Key.</p> <p>From the JTAG interface, the Read and Write operations on the FlashROM content require a Pass Key. You can Verify the FlashROM content via the JTAG interface without a Pass Key.</p> <p>Read back of the FlashROM content is allowed from the FPGA Array.</p>
None	No security	<p>The Write and Verify operations of the FPGA Array do not require keys.</p> <p>The Read, Write, and Verify operations of the FlashROM content also do not require keys.</p>

- Enter the **Pass Key** and/ or the **AES Key** as appropriate. You can generate a random key by clicking the **Generate random key** button.

The **Pass Key** protects all the Security Settings for the FPGA Array and/or FlashROM.

The **AES Key** decrypts FPGA Array and/or FlashROM programming file content. Use the AES Key if you intend to program the device at an unsecured site or if you plan to update the design at a remote site in the future.

You can also customize the security levels by clicking the **Custom Level** button. For more information, see the [Custom Security Levels](#) section.

Custom Security Levels

For advanced use, you can customize your security levels.

To set custom security levels:

- Click the **Custom Level** button in the **Setup Security** page. The **Custom Security** dialog box appears (see figure below).

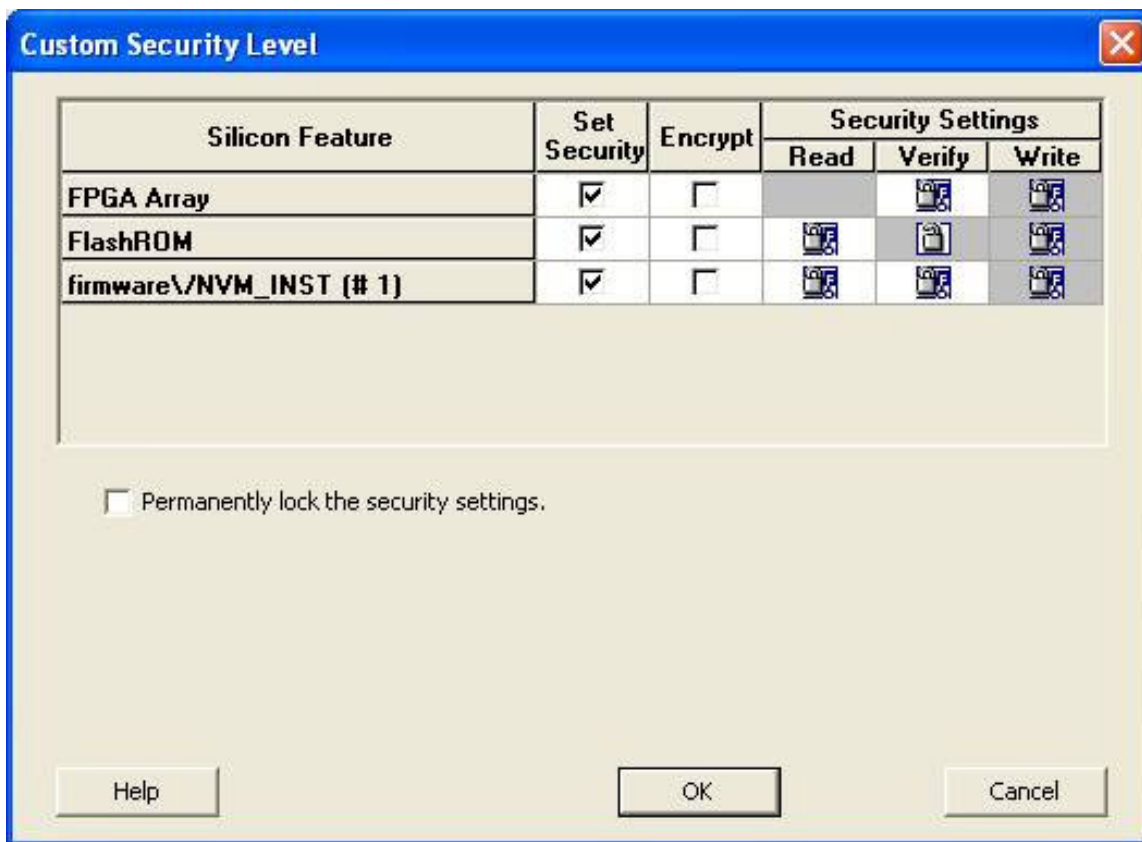


Figure 102 · Custom Security Level

2. Select the **FPGA Array Security** and the **FlashROM Security** levels. For Fusion devices, you can also choose the Embedded Flash Memory Block level of security. The FPGA Array and the FlashROM can have different Security Settings. See the tables below for a description of the custom security option levels for FPGA Array and FlashROM.

Table 14 · FPGA Array

Security Option	Description																				
Lock for both writing and verifying <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FPGA Array</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FPGA Array	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Allows writing/erasing and verification of the FPGA Array via the JTAG interface only with a valid Pass Key.					
Device Feature				Set Security	Encrypt	Security Settings															
	Read	Verify	Write																		
FPGA Array	<input checked="" type="checkbox"/>	<input type="checkbox"/>																			
Lock for writing <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FPGA Array</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FPGA Array	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Allows the writing/erasing of the FPGA Array only with a valid Pass Key. Verification is allowed without a valid Pass Key.					
Device Feature				Set Security	Encrypt	Security Settings															
	Read	Verify	Write																		
FPGA Array	<input checked="" type="checkbox"/>	<input type="checkbox"/>																			
Use the AES Key for both writing and verifying <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FPGA Array</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FPGA Array	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				Allows the writing/erasing and verification of the FPGA Array only with a valid AES Key via the JTAG interface. This configures the device to accept an encrypted bitstream for reprogramming and verification of the FPGA Array. Use this option if you intend to complete final programming at an unsecured site or if you plan to update the design at a remote site in the future. Accessing the device security settings requires a valid Pass Key.					
Device Feature				Set Security	Encrypt	Security Settings															
	Read	Verify	Write																		
FPGA Array	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>																			
Allow write and verify <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FPGA Array</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FPGA Array	<input type="checkbox"/>	<input type="checkbox"/>				Allows writing/erasing and verification of the FPGA Array with plain text bitstream and without requiring a Pass Key or an AES Key. Use this option when you develop your product in-house.					
Device Feature				Set Security	Encrypt	Security Settings															
	Read	Verify	Write																		
FPGA Array	<input type="checkbox"/>	<input type="checkbox"/>																			

Note: The ProASIC3 family FPGA Array is always read protected regardless of the Pass Key or the AES Key protection.

Table 15 · FlashROM

Security Option	Description															
<p>Lock for both reading and writing</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FlashROM</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FlashROM	<input checked="" type="checkbox"/>	<input type="checkbox"/>				<p>Allows the writing/erasing and reading of the FlashROM via the JTAG interface only with a valid Pass Key. Verification is allowed without a valid Pass Key.</p>
Device Feature				Set Security	Encrypt	Security Settings										
	Read	Verify	Write													
FlashROM	<input checked="" type="checkbox"/>	<input type="checkbox"/>														
<p>Lock for writing</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FlashROM</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FlashROM	<input checked="" type="checkbox"/>	<input type="checkbox"/>				<p>Allows the writing/erasing of the FlashROM via the JTAG interface only with a valid Pass Key. Reading and verification is allowed without a valid Pass Key.</p>
Device Feature				Set Security	Encrypt	Security Settings										
	Read	Verify	Write													
FlashROM	<input checked="" type="checkbox"/>	<input type="checkbox"/>														
<p>Use the AES Key for both writing and verifying</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FlashROM</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FlashROM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<p>Allows the writing/erasing and verification of the FlashROM via the JTAG interface only with a valid AES Key. This configures the device to accept an encrypted bitstream for reprogramming and verification of the FlashROM. Use this option if you complete final programming at an unsecured site or if you plan to update the design at a remote site in the future. Note: The bitstream that is read back from the FlashROM is always unencrypted (plain text).</p>
Device Feature				Set Security	Encrypt	Security Settings										
	Read	Verify	Write													
FlashROM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>														
<p>Allow reading, writing, and verifying</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>FlashROM</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	FlashROM	<input type="checkbox"/>	<input type="checkbox"/>				<p>Allows writing/erasing, reading and verification of the FlashROM content with a plain text bitstream and without requiring a valid Pass Key or an AES Key.</p>
Device Feature				Set Security	Encrypt	Security Settings										
	Read	Verify	Write													
FlashROM	<input type="checkbox"/>	<input type="checkbox"/>														

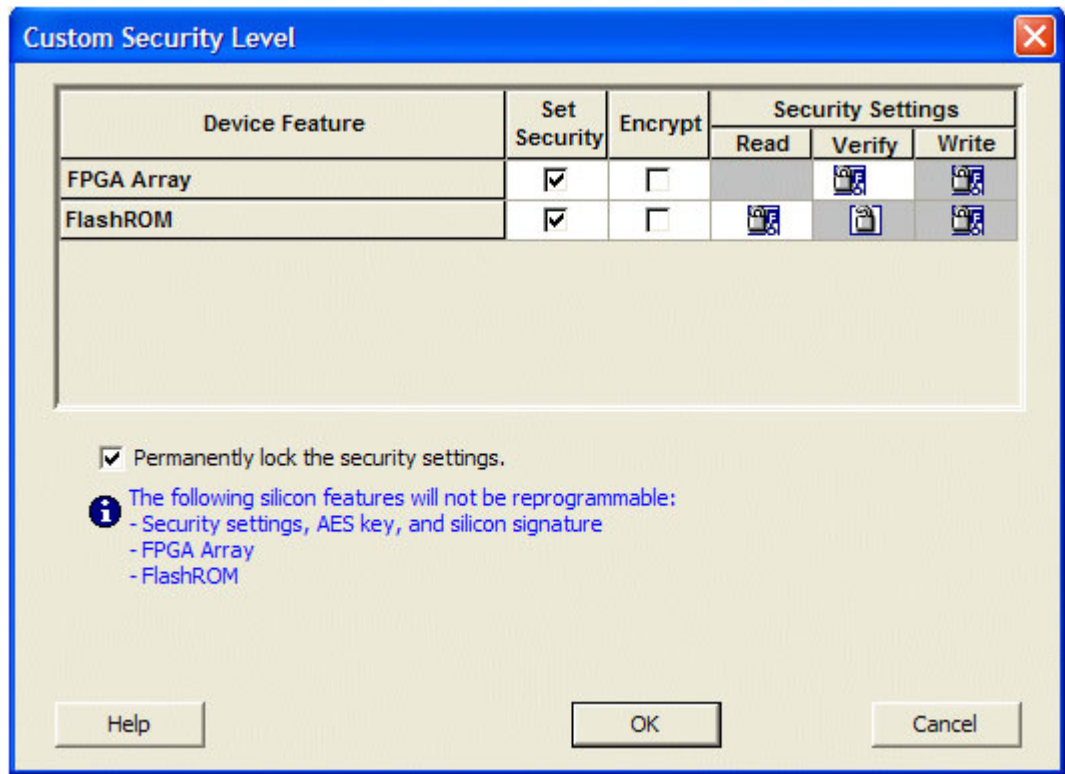
Note: The FPGA Array can always read the FlashROM content regardless of these Security Settings.



Table 16 · Embedded Flash Memory Block

Security Option	Description																				
<p>Lock for reading, verifying, and writing</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>firmware\NVM_INST (# 1)</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>				<p>Allows the writing and reading of the Embedded Flash Memory Block via the JTAG interface only with a valid Pass Key. Verification accomplished by reading back and compare.</p>					
Device Feature				Set Security	Encrypt	Security Settings															
	Read	Verify	Write																		
firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>																			
<p>Lock for writing</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>firmware\NVM_INST (# 1)</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>				<p>Allows the writing of the Embedded Flash Memory Block via the JTAG interface only with a valid Pass Key. Reading and verification is allowed without a valid Pass Key.</p>					
Device Feature				Set Security	Encrypt	Security Settings															
	Read	Verify	Write																		
firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>																			
<p>Use AES Key for writing</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>firmware\NVM_INST (# 1)</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<p>Allows the writing of the Embedded Flash Memory Block via the JTAG interface only with a valid AES Key. This configures the device to accept an encrypted bitstream for reprogramming of the Embedded Flash Block. Use this option if you complete final programming at an unsecured site or if you plan to update the design at a remote site in the future. The bitstream that is read back from the Embedded Flash Memory Block is always unencrypted (plain text), when a valid pass key is provided.</p>					
Device Feature				Set Security	Encrypt	Security Settings															
	Read	Verify	Write																		
firmware\NVM_INST (# 1)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>																			
<p>Allow reading, writing, and verifying</p> <table border="1"> <thead> <tr> <th rowspan="2">Device Feature</th> <th rowspan="2">Set Security</th> <th rowspan="2">Encrypt</th> <th colspan="3">Security Settings</th> </tr> <tr> <th>Read</th> <th>Verify</th> <th>Write</th> </tr> </thead> <tbody> <tr> <td>firmware\NVM_INST (# 1)</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Device Feature	Set Security	Encrypt	Security Settings			Read	Verify	Write	firmware\NVM_INST (# 1)	<input type="checkbox"/>	<input type="checkbox"/>				<p>Allows writing, reading and verification of the Embedded Flash Memory Block content with a plain text bitstream and without requiring a valid Pass Key or an AES Key.</p>					
Device Feature				Set Security	Encrypt	Security Settings															
	Read	Verify	Write																		
firmware\NVM_INST (# 1)	<input type="checkbox"/>	<input type="checkbox"/>																			

- To make the Security Settings permanent, select **Permanently lock the security settings** check box. This option prevents any future modifications of the Security Setting of the device. A Pass Key is not required if you use this option.
Note: When you make the Security Settings permanent, you can never reprogram the [Silicon Signature](#). If you Lock the write operation for the FPGA Array or the FlashROM, you can never reprogram the FPGA Array or the FlashROM, respectively. If you use an AES key, this key cannot be changed once you permanently lock the device.
- To use the Permanent FlashLock™ feature, select **Lock for both writing and verifying** for FPGA Array and **Lock for both reading and writing** for FlashROM and select the **Permanently lock the security settings** checkbox as shown in the figure below. This will make your device one-time-programmable.



Custom Security Level

5. Click the OK button.

The Security Settings page appears with the Custom security setting information as shown in the figure below.

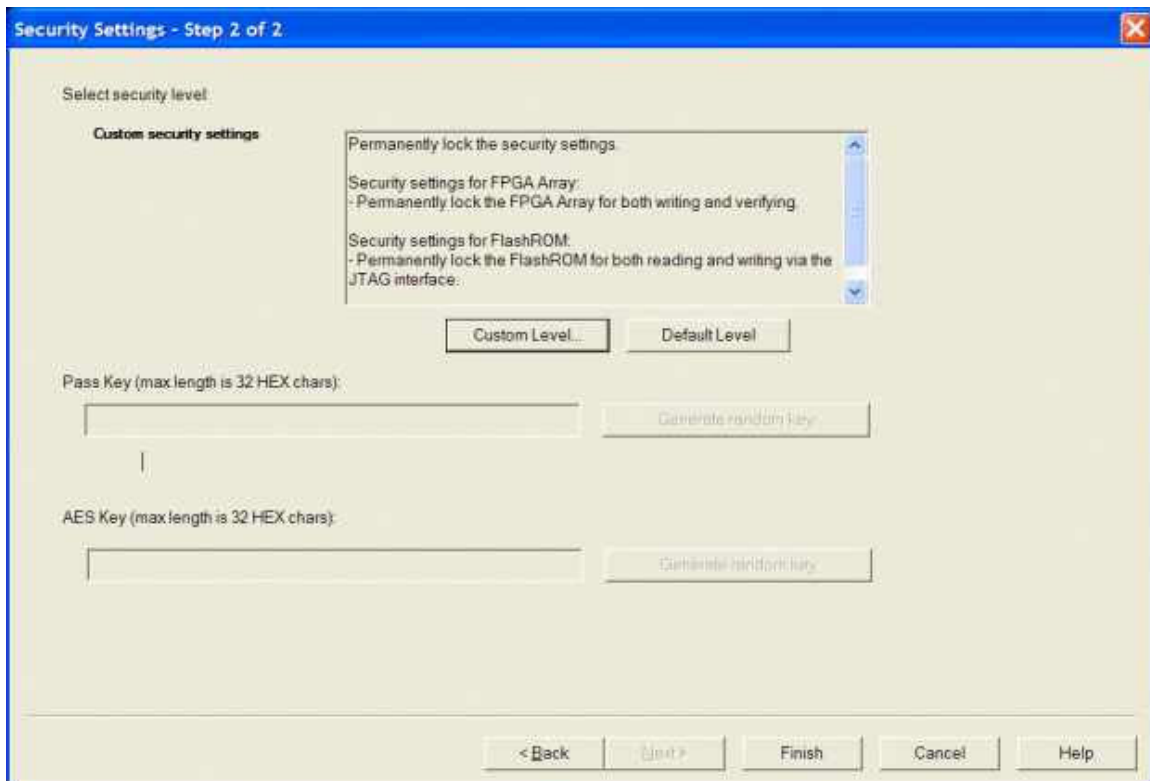


Figure 103 · Security Settings

Custom Serialization Data for FlashROM Region

FlashPoint enables you to specify a custom serialization file as a source to provide content for programming into a Read from file FlashROM region. You can use this feature for serializing the target device with a custom serialization scheme.

To specify a FlashROM region:

1. From the Properties section in the FlashROM Settings page, select the file name of the custom serialization file (see figure below). For more information on custom serialization files, see [Custom Serialization Data File Format](#).

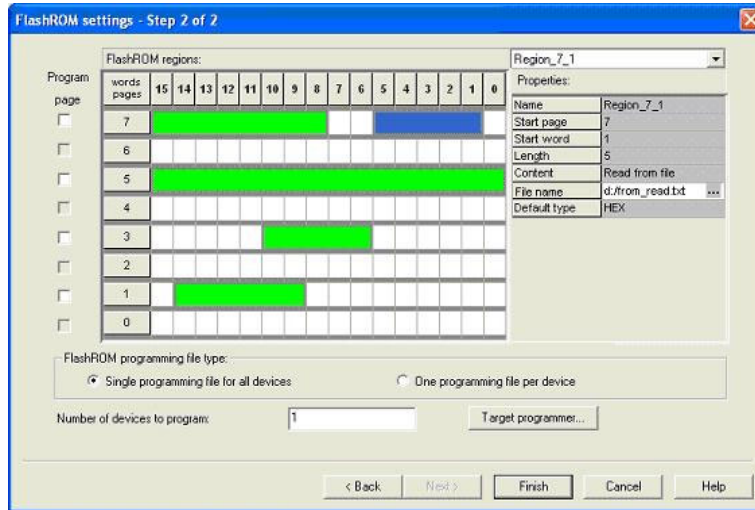


Figure 104 · FlashROM Settings

2. Select the FlashROM programming file type you want to generate from the two options below:
 - Single programming file for all devices option: generates one programming file with all the values in the custom serialization file.
 - One programming file per device: generates one programming file for each value in the custom serialization file.
3. Enter the number of devices you want to program.
4. Click the **Target Programmer** button.
5. Select your target Programmer type.
6. Click OK.

Custom Serialization Data File Format

FlashPoint supports custom serialization data files that specify the data in binary, HEX, decimal, or ASCII text. The custom serialization data files may contain multiple data with the Line Feed (LF) character as the delimiter. You can create a file by entering serialization data into any type of text editor. Depending on the serialization data format (hex, ASCII, binary, decimal), input the serialization data according to the size of the region you specified in the FlashROM settings page.

Semantics

Each custom serialization file has only one type of data format (binary, decimal, Hex or ASCII text). For example, if a file contains two different data formats (i.e. binary and decimal) it is considered an invalid file.

The length of each data file must be shorter or equal to the selected region length. If the data is shorter than the selected region length, the most significant bits shall be padded with 0's. If the specified region length is longer than the selected region length, it is considered an invalid file.

The digit / character length is as follows:

- Binary digit: 1 bit
- Decimal digit: 4 bits
- Hex digit: 4 bits
- ASCII Character: 8 bits

Note the standard example below:



If you wanted to use, for example, device serialization for three devices with serialization data 123, 321, and 456, you would create file name from_read.txt. Each line in from_read.txt corresponds to the serialization data that will be programmed on each device. For example, the first line corresponds to the first device to be programmed, the second line corresponds to the second device to be programmed, and so on.

Hex serialization data file example

The following example is a Hex serialization data file for a 40-bit region. Enter the serialization data below into file created by any text editor:

```
123AEd210
AeB1
0001242E
```

Note: If you enter an invalid Hex digit such as 235SedF1, an error occurs. An error will also occur if you enter data that is out of range, i.e. 4300124EFE.

The following is an example of programming "AeB1" into Region_7_1 located on page 7, from Word 5 to Word 1 in the FlashROM settings page. See [Custom serialization data for FlashROM region](#) for more information.

	Table 15	...		Word 5	Word 4	Word 3	Word 2	Word 1	Word 0
Page 7	00	00	00	AE	B1	...

Binary serialization data file example

The following example is a binary serialization data file for a 16-bit region:

```
1100110011010001
100110011010011
11001100110101111 (This is an error: data out of range)
1001100110110111
1001100110110112 (This is an error: invalid binary digit)
```

Decimal serialization data file example

The following example is a decimal serialization data file for a 16-bit region:

```
65534
65535
65536 (This is an error: data out of range)
6553A (This is an error: invalid decimal digit)
```

Text serialization data file example

The following example is a text serialization data file for a 32-bit region:

```
AESB
A )e
ASE3 23 (This is an error: data out of range)
65A~
1234
AEbF
```

Syntax

```
Custom serialization data file
hex region data list > | <decimal region data list> |
  <binary region data list> | <ascii text data list>
  Hex region data list = hex data> <new line> { < hex data> <new line> }
Decimal region data list = <decimal data> <new line> {<decimal data><new line> }
Binary region data list = <binary data> <new line> { <binary data> <new line> }
ASCII text region data list = < ascii text data> <new line>
{ < ascii text data> <new line> }
hex data = <hex digit> {<hex digit>}
decimal data = < decimal digit> {< decimal digit>}
binary data = < binary digit> {< binary digit>}
ASCII text data = <ascii character> {< ascii character >}
new line = LF
binary digit = '0'|'1'
decimal digit = '0'|'1'|'2'|'3'|'4'|'5'|'6'|'7'|'8'| '9'
hex digit = '0'|'1'|'2'|'3'|'4'|'5'|'6'|'7'|'8'|'9'|'A'|'B'|'C'|'D'| 'E'| 'F'|
'a'| 'b'| 'c'| 'd'| 'e'| 'f'
ascii character = characters from SP(0x20) to '~'(0x7E).
```



Basic Programming Tutorials

Single STAPL/PDB File Basic Tutorial

This section provides step-by-step instructions to familiarize you with the basic features of the FlashPro software, specifically how to program a device. For more detailed step-by-step instructions and help with advanced features of the software, please see specific topics in the online help.

Note: This tutorial assumes that you have already installed the FlashPro software and have started the program. If you need instructions on how to install the software or start the program, see [Software Installation](#) and [Starting Standalone FlashPro](#).

First, create a new project and name it Tutorial. If FlashPro is launched through Libero, a new project will be created automatically and a PDB file loaded, if available.

To Create a Project:

1. Click the **New Project** button from the FlashPro GUI.
2. From the **New Project** dialog box, type "Tutorial" in the **Project Name** field.

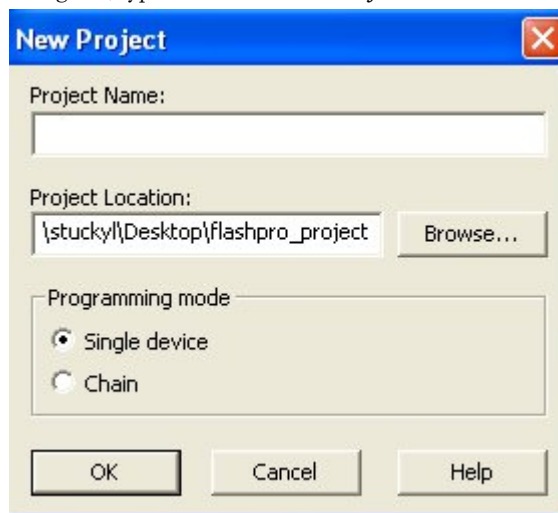


Figure 105 · New Project Dialog Box

3. If necessary, change the default location of your project in the **Project Location** field.
4. Select the **Single device** **Programming mode**
5. Click **OK**. The FlashPro GUI displays (see figure below). The **Programmer List Window** updates with your programmer information.

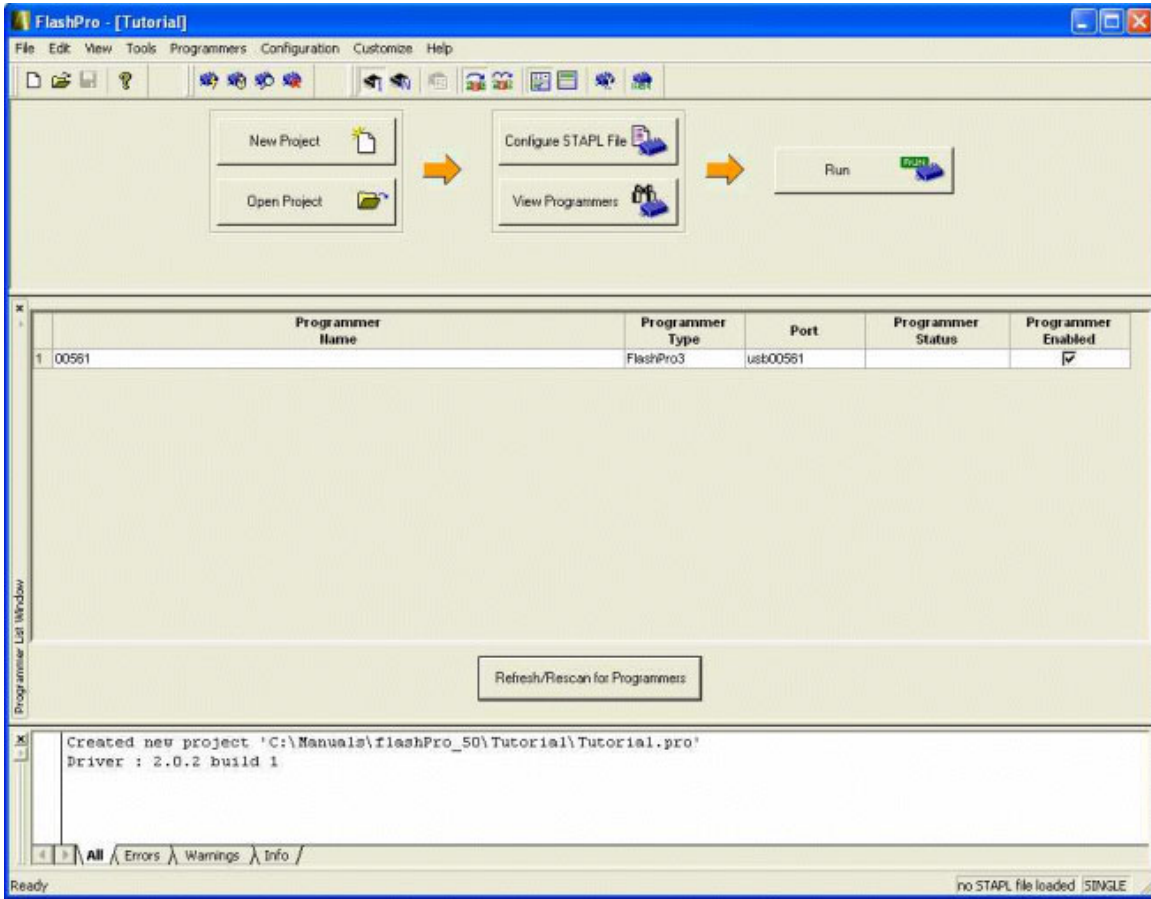


Figure 106 · Main GUI

Loading and Configuring a Programming File

Once you have created your project and connected your programmer, you are ready to load your PDB or STAPL file.

To load a Programming file:

1. Click the **Configure device** button. The **Single Device Configuration** window displays in the FlashPro GUI (see figure below).



Figure 107 · Single Device Configuration Window

2. Click the **Browse** button to find your Programming file.



3. From the **Load Programming File** dialog box, find your Programming file and click **Open** (See figure below).

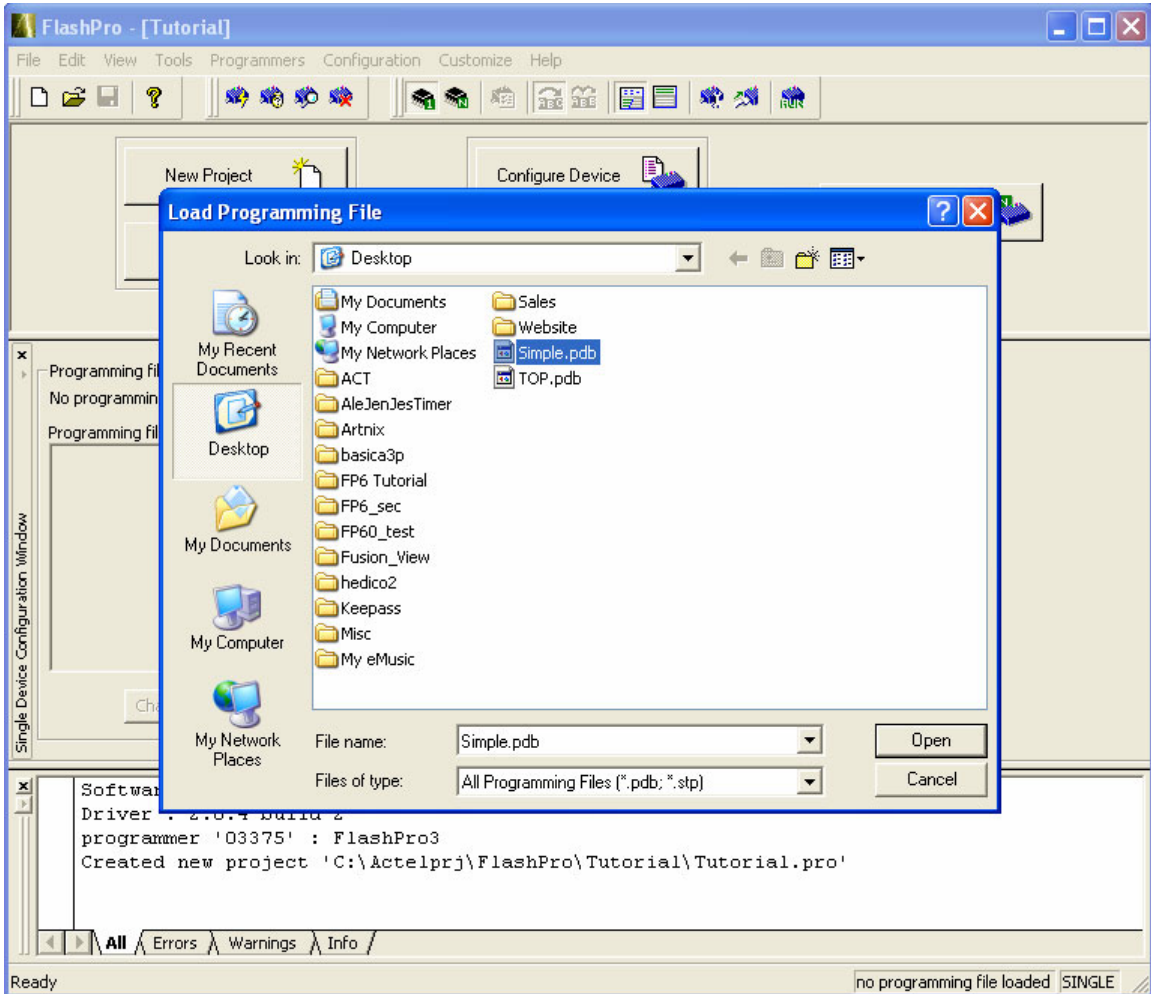


Figure 108 · Load Programming File Dialog Box

The **Single Device Configuration Window** updates to list your Programming file information and the actions available with your Programming file in the **Action** list box (see figure below). Program is the default action displayed in the **Action** list box.

Note: Actel recommends using the default settings.

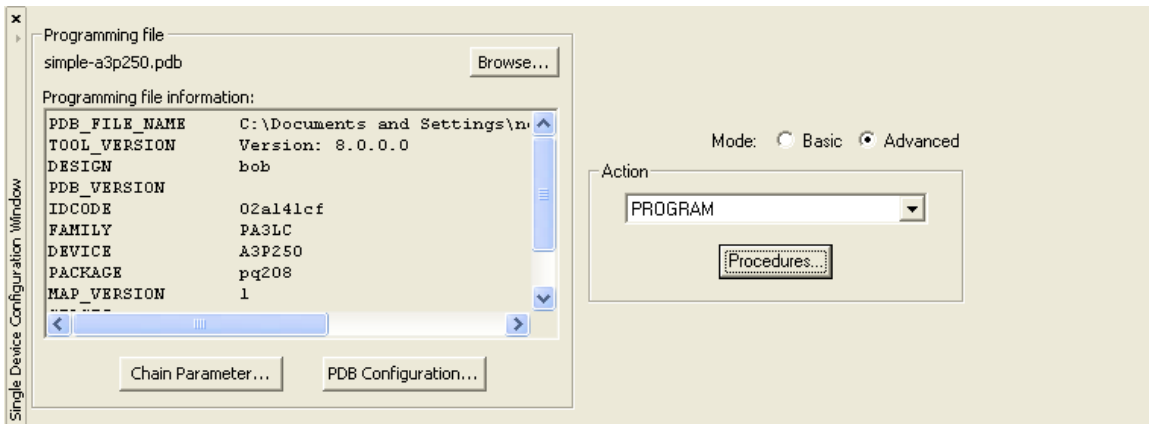


Figure 109 · Single Device Configuration Window

This tutorial gives instructions on how to program a device. For an explanation on the other actions available, see [Programming File Actions](#).

Programming a Device

Now that you have loaded your PDB file, programming a device is the next step.

To program a device:

1. From the **Action** list, select **Program** (see figure below).

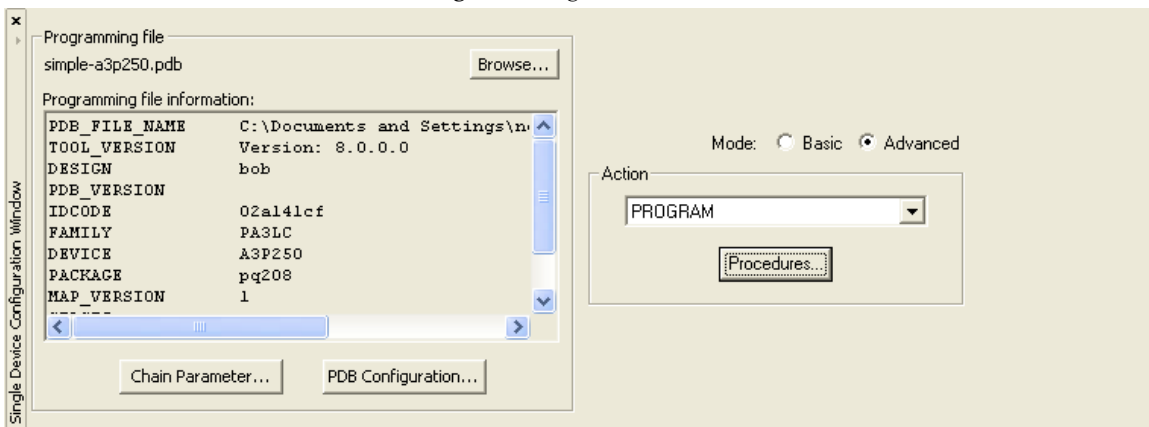


Figure 110 · Selecting Program from the Action List box

2. Click the **Procedures** button (see figure below).

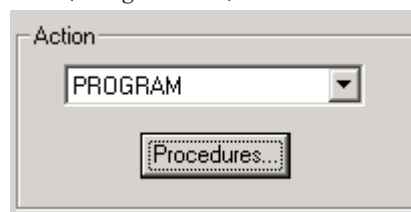


Figure 111 · Procedures Button

The **Select Action And Procedures** dialog box displays showing the procedures for the Programming action (see figure below). Actel recommends using the default settings.

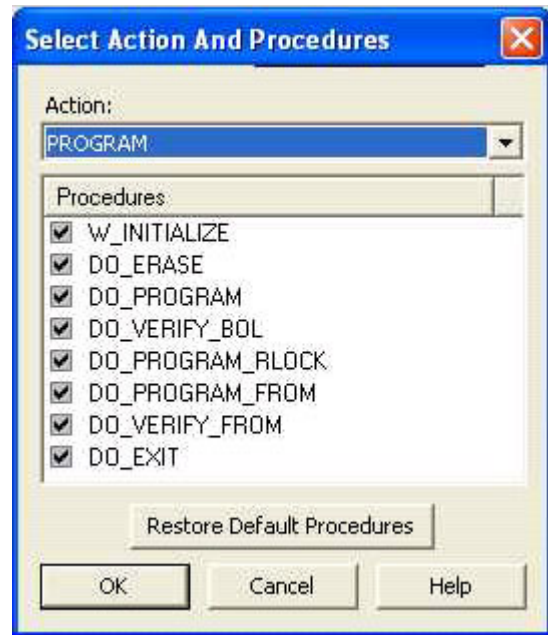


Figure 112 · Select Action and Procedures dialog box

3. Click the **Restore Default Procedures** button.
4. From the FlashPro main GUI, click the **Program** button to program your device.

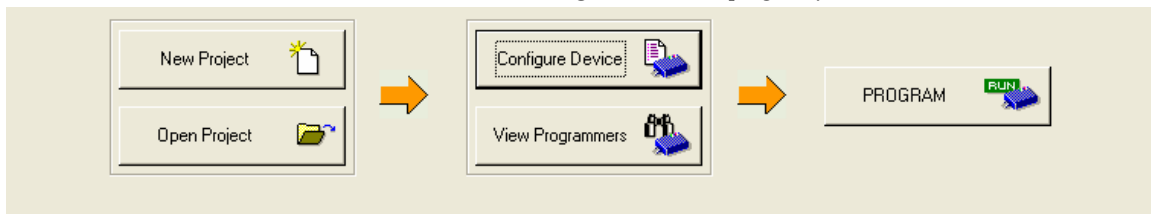


Figure 113 · Flow Window: Program Button

The **Programmer List Window** updates the **Programmer Status** column with "Run Passed" indicating that you have successfully programmed the device (see figure below).

Note: The status indicator updates during programming to show the programming progress, then it will change to a pass or fail result when the operation is complete.

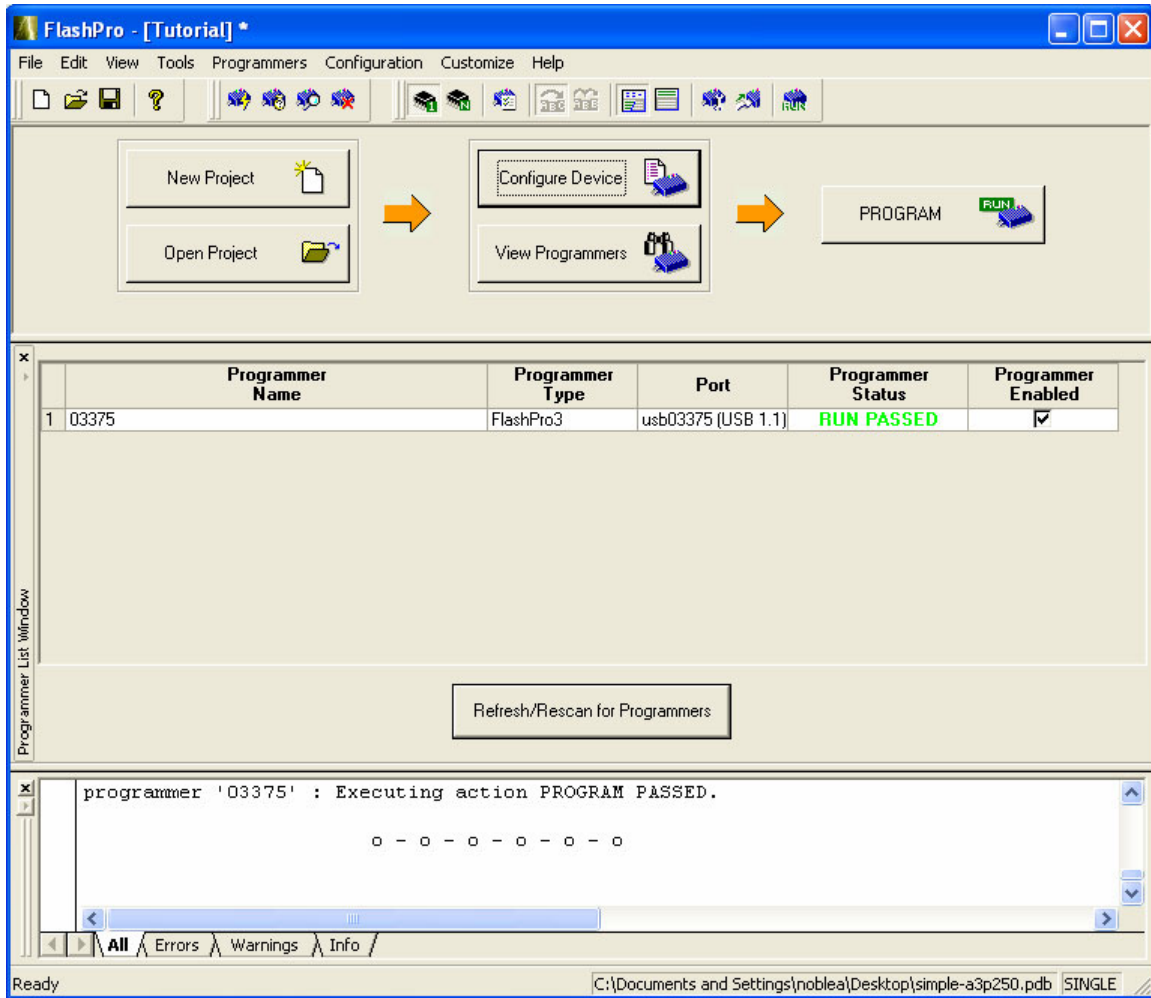


Figure 114 · Successfully Programmed Device

5. View the Log window and take note of the details about your programmed device.

Single Actel Device with Serialization Tutorial

This tutorial provides step-by-step instructions on how to program a single Actel Device with Serialization. Before you begin this tutorial, make sure you have already installed the FlashPro software and that you are familiar with the basic features of using the FlashPro software.

First, create the file generator using FROM for device serialization. You must have access to the Libero v7.3 or later software to complete this step.

To Configure the FROM data for serialization:

1. Generate FROM using SmartGen.
2. From the **Properties** section in the **FlashROM Settings** dialog box, select **Auto Inc** or **Read From File** region. For the **Auto Inc** region, specify the step value. You will not be able to modify this value in the FlashPoint software.
3. Complete the normal design flow and finish place and route.



4. Select **Program FlashROM**.
5. Click **Browse** to find the UFC file.
6. Check the FPGA Array box and click **Next**.

The FlashROM Settings window displays (see figure below).

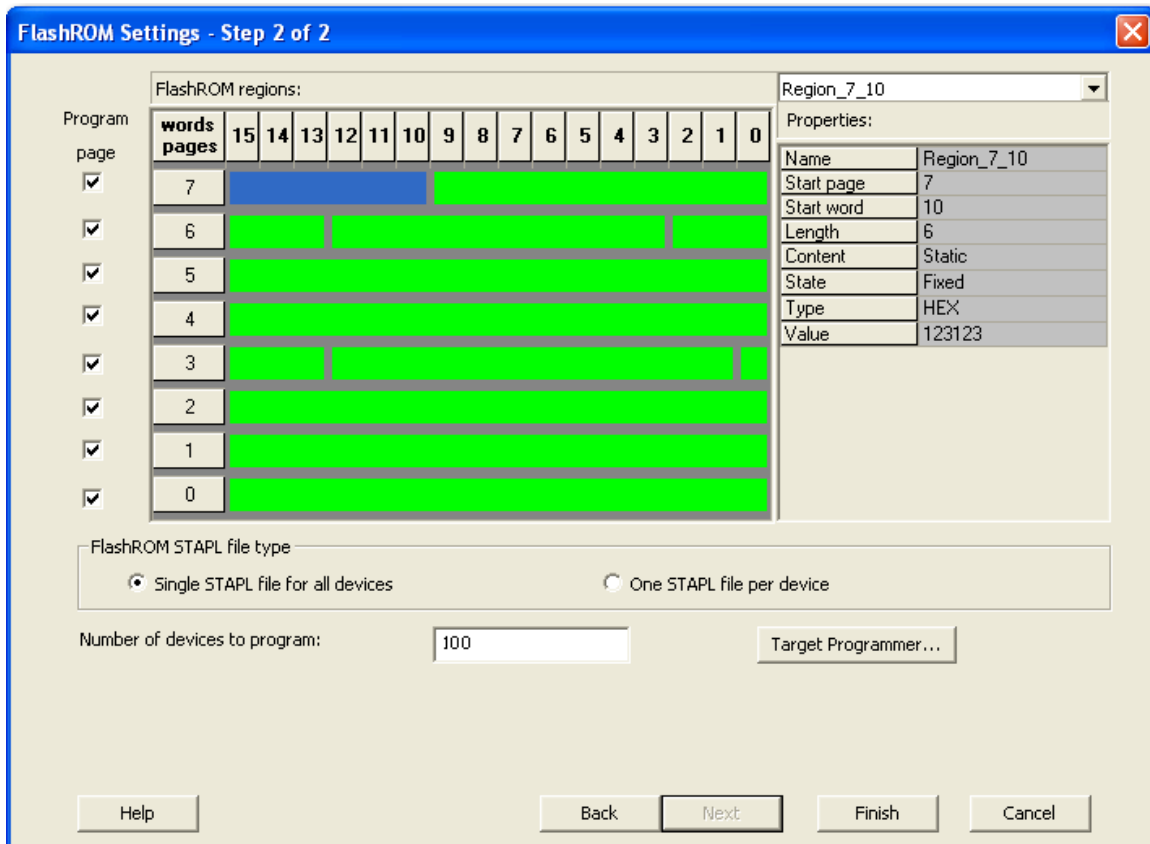


Figure 115 · FlashROM Settings- Step 2 of 2

7. Select the FROM page you want to program and data value for the configured regions.

Note: The generated STAPL file contains only the data that targets the selected FROM page.
8. Modify properties for the serialization by specifying the **Start** and **Max** values. For the **Auto Inc** region, specify the **Start** and **Max** values. For the **Read From File** region, select the file name of the custom serialization file.
9. Select the FlashROM programming file type you want to generate from the two options below:
 - Choose single STAPL file for all devices: generates one programming file with all FROM values.
 - Choose one STAPL file per device: generates a separate programming file for each FROM value.
10. Enter the number of devices you want to program and generate the required programming file.
11. Click the **Finish** button.

You have completed the steps to enable device serialization. Now, from FlashPro software, you are ready to program a device using Device Serialization.

To program a device using device serialization:

1. Click the **New Project** button from the FlashPro GUI.
2. From the **New Project** dialog box, type “Tutorial” in the **Project Name** field. See Figure below.

3. Check the **Single STAPL** file option from the **Programming Mode** area.

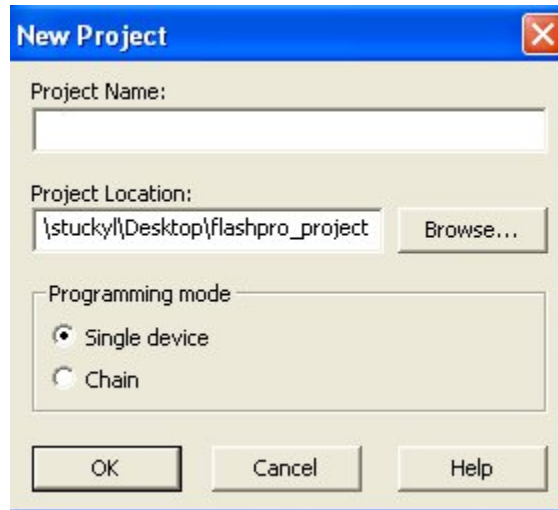


Figure 116 · New Project Dialog Box

4. If necessary, change the default location of your project in the **Project Location** field.
5. Click **OK**. The FlashPro GUI displays (see figure below).



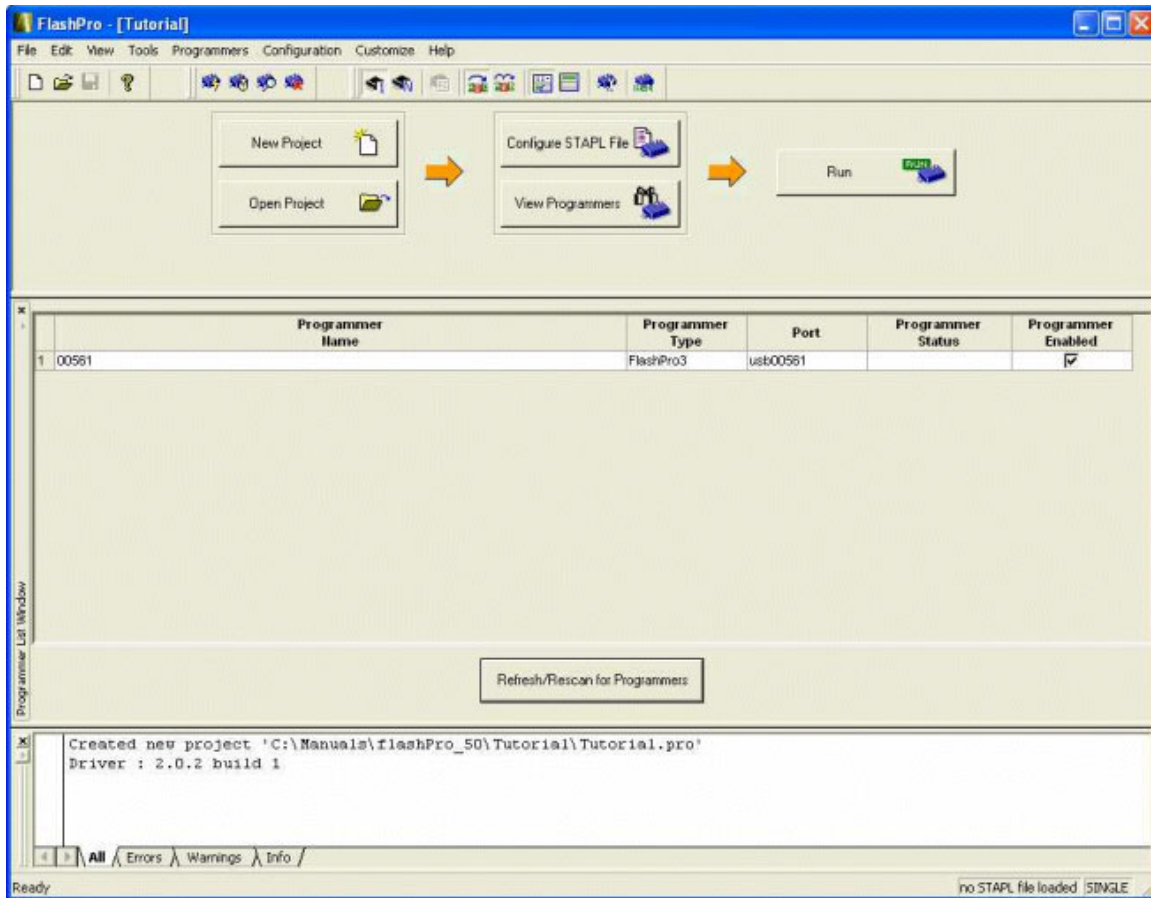


Figure 117 · FlashPro Main GUI

The Programmer List Window updates with your programmer information.

6. Click the **Configure STAPL File** button to load the STAPL file. The **Single STAPL Configuration Window** displays in the FlashPro GUI (see figure below).

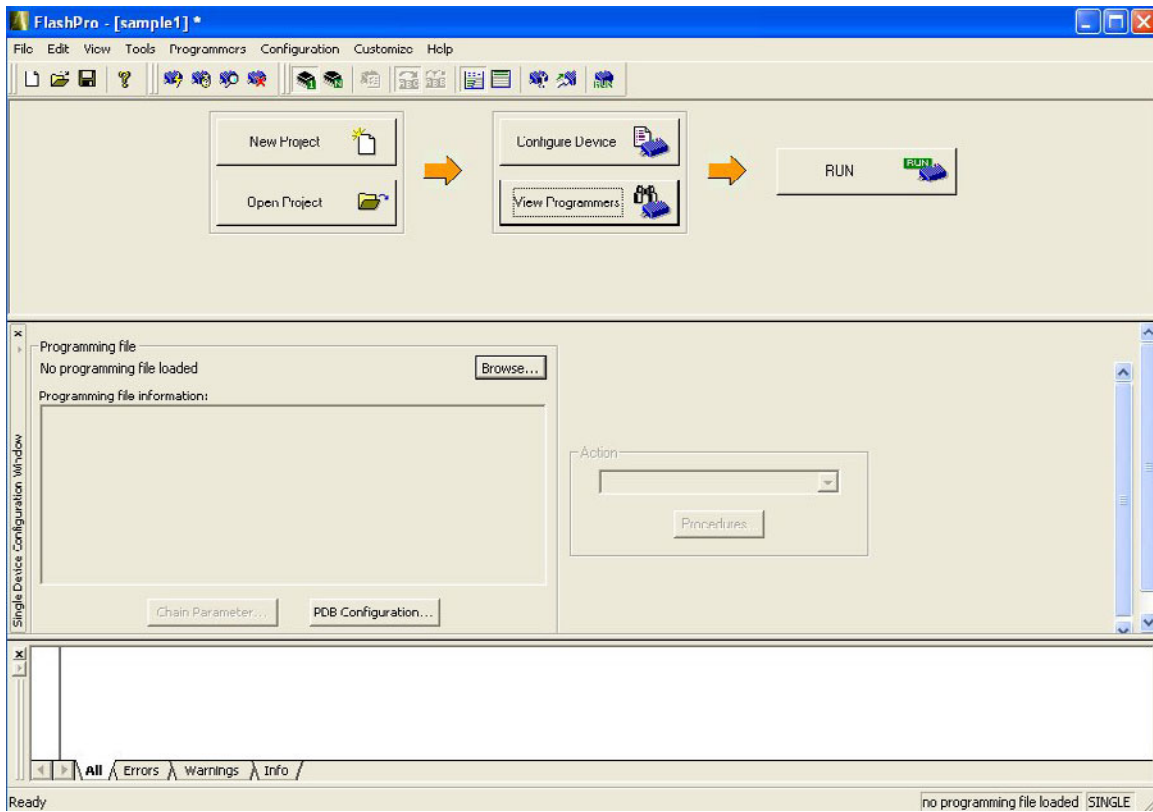


Figure 118 · Single STAPL Configuration Window

7. Click the **Browse** button to find your STAPL file.
8. From the **Load STAPL File** dialog box, find your STAPL file and click **Open**. The **Single STAPL Configuration Window** updates to list your STAPL file information and the actions available with your STAPL file in the **Action** list box (see figure below).



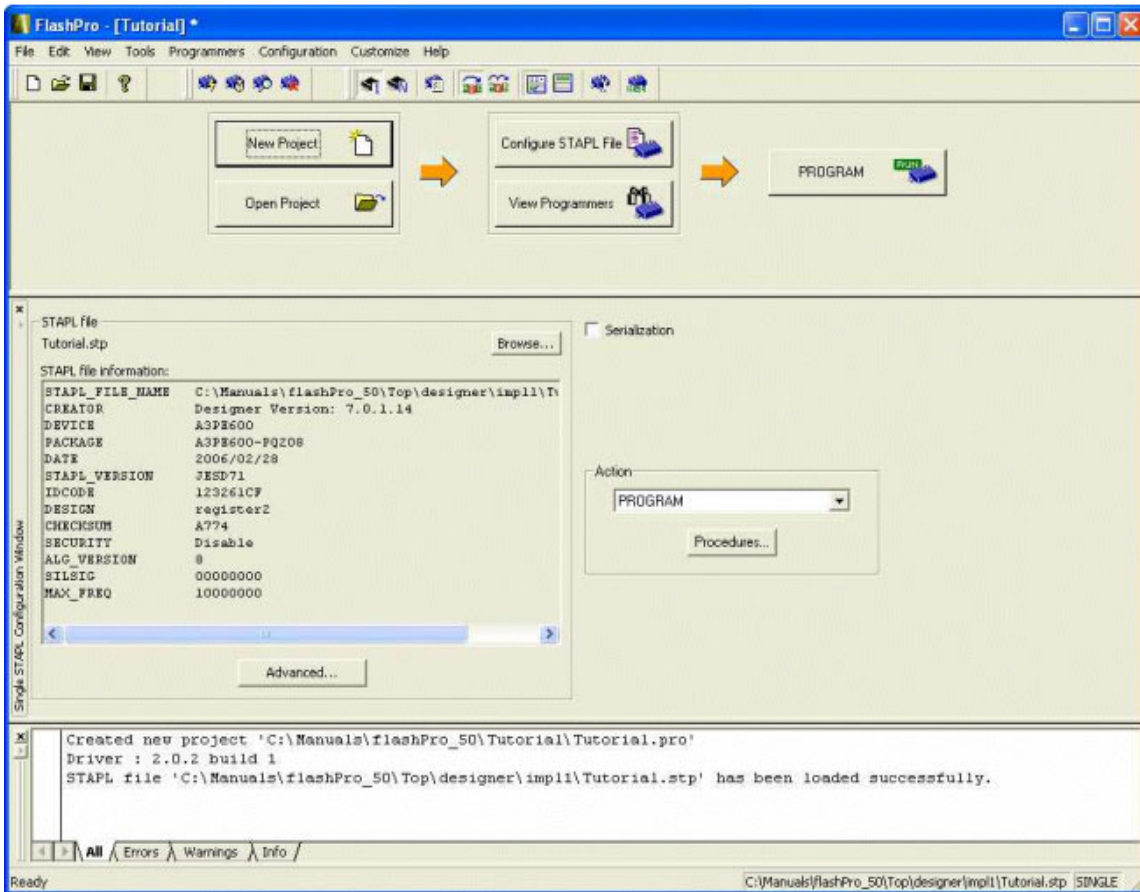


Figure 119 · Single STAPL Configuration Window with STAPL File Uploaded

- From the **Single Device Configuration Window** in the FlashPro software, check the **Serialization** box and click the **Select Serialization Indexes** button (see figure below).



Figure 120 · Single Device Configuration Window- Serialization

The **Serial Settings** dialog box displays (see figure below).

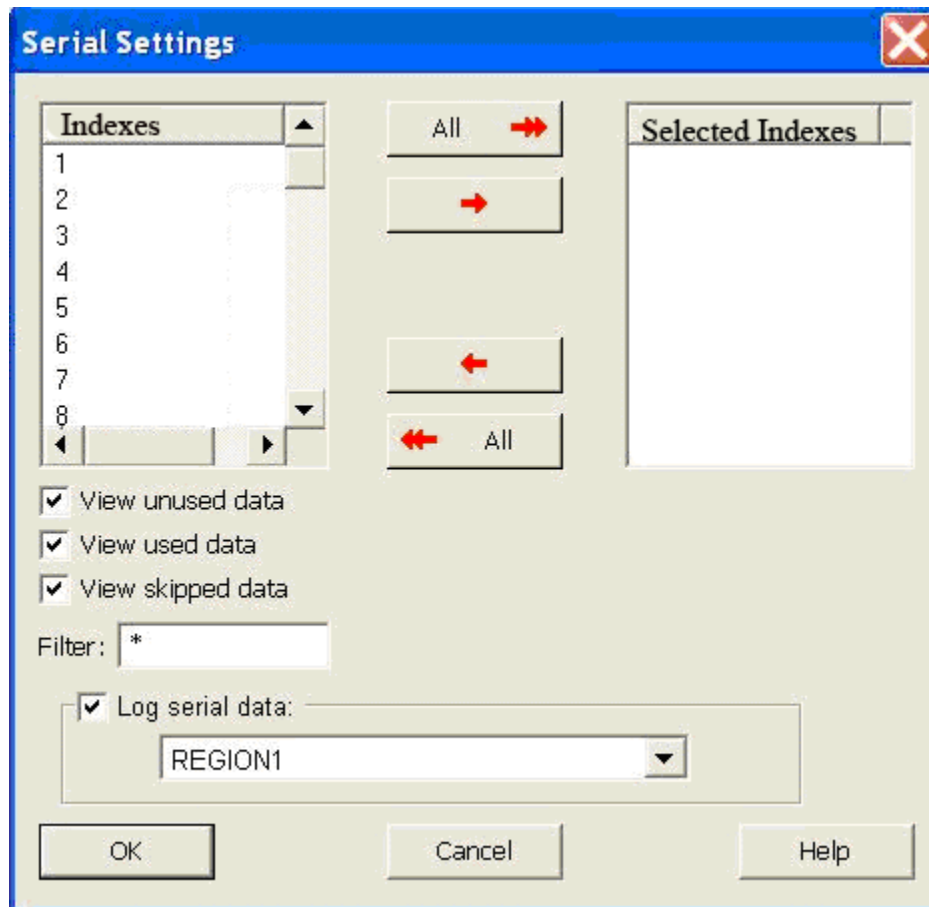


Figure 121 · Serial Settings Dialog Box

10. From the **Serial Settings** dialog box, click the **All** button to select all the serial data.
11. Click **OK**.
The **Serialization Indexes** text box updates (see figure below).



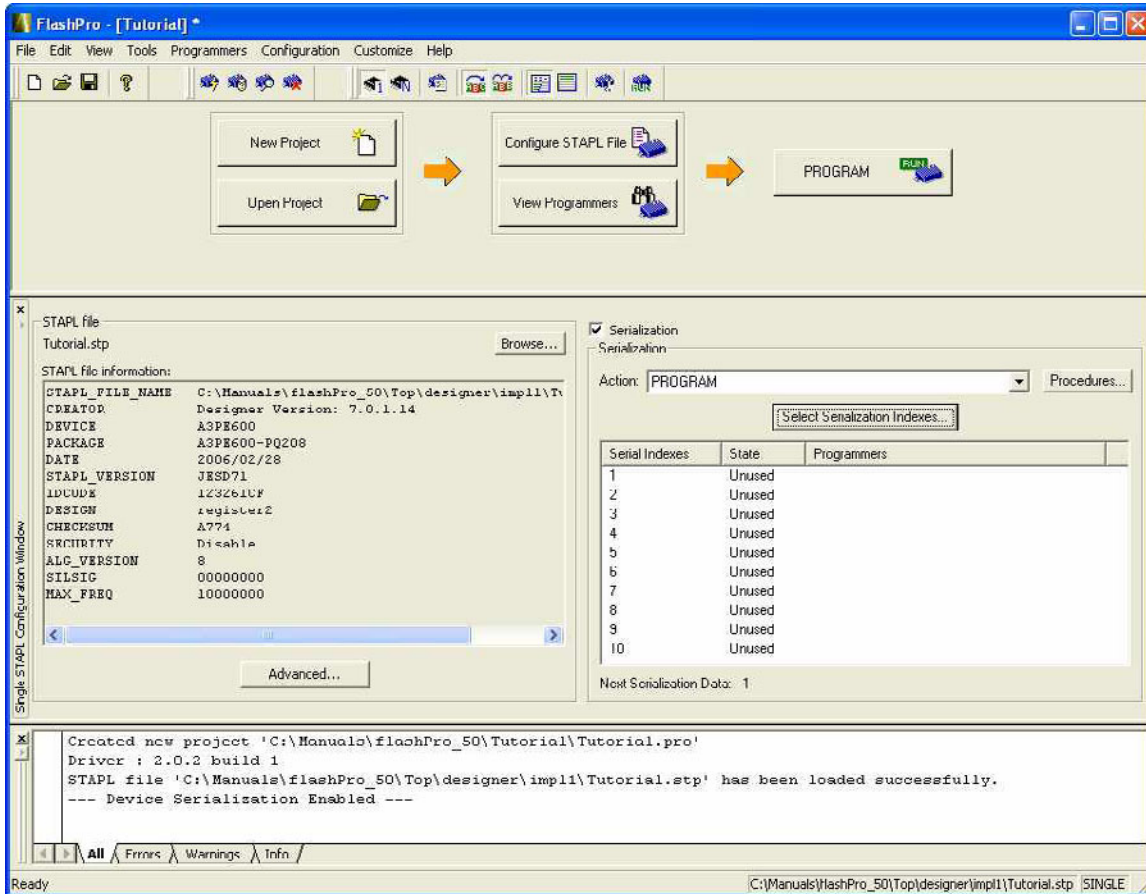


Figure 122 · Single STAPL File Configuration Window- Serialization Indexes Update

- Click the **Program** button to program your device using serialization.
- Congratulations, you have just completed the FlashPro tutorial for Single Actel Devices with Serialization.

Chain Programming Tutorial

This tutorial demonstrates how to directly program an APA300 device that is part of a heterogeneous JTAG chain. The example in this tutorial uses one APA300 device and three non-Actel devices configured as shown in the figure below.

Note: This tutorial is performed in Advanced Mode. You can change your display mode to Advanced Mode from the [Preferences](#) dialog box.

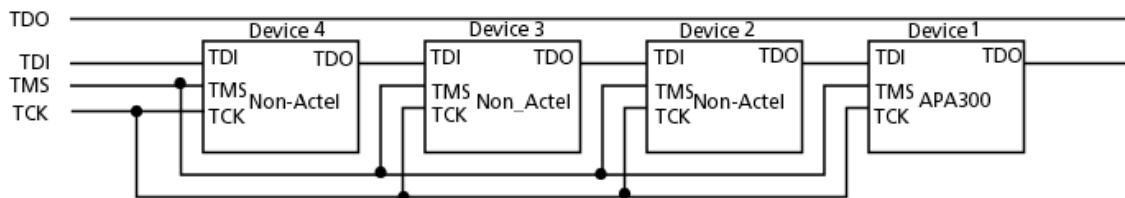


Figure 123 · APA Device Tutorial Example

First, create a new project.

To create a new project:

1. Click the **New Project** button from the FlashPro GUI.
2. From the **New Project** dialog box, type “Tutorial” in the **Project Name** field.
3. Check the **Chain** option from the **Programming Mode** area.

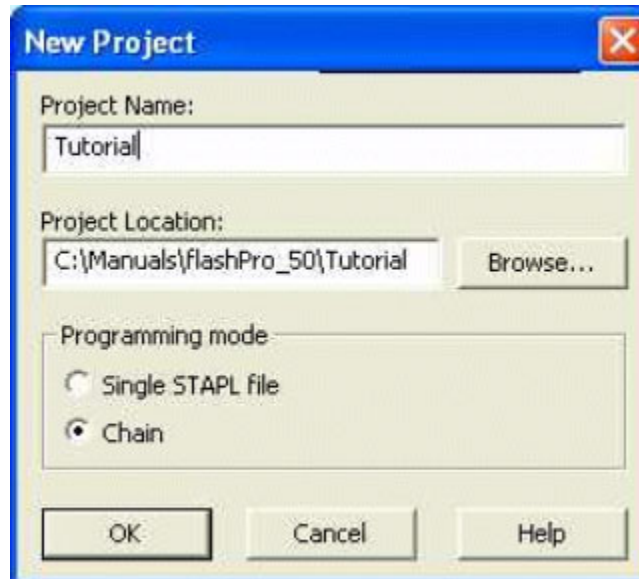


Figure 124 · New Project Dialog Box

4. If necessary, change the default location of your project in the **Project Location** field.
5. Click **OK**. The FlashPro GUI displays (see figure below).



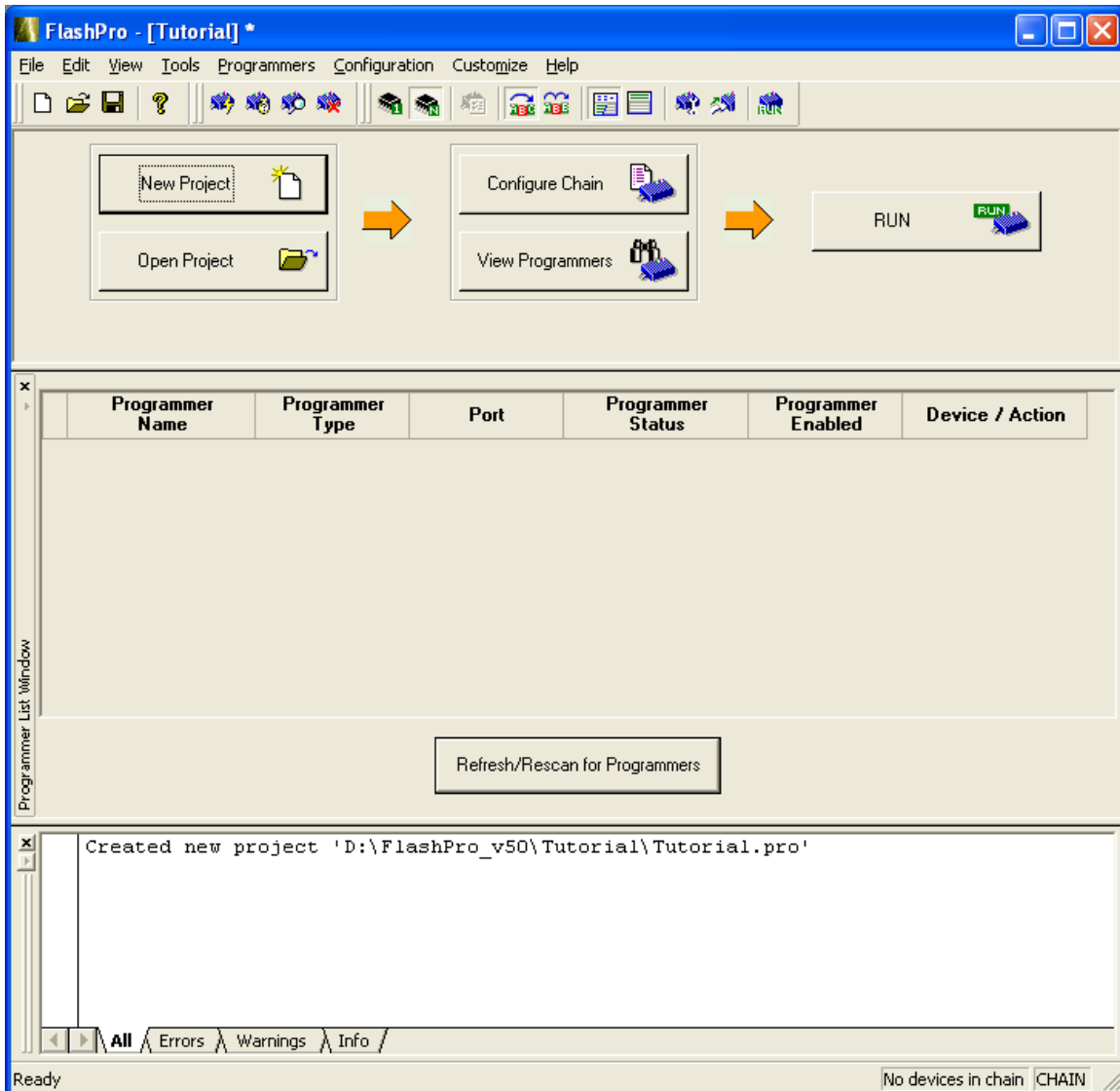


Figure 125 · FlashPro Main GUI

Note: The Programmer List Window updates with your programmer information.

- From the **Menu** bar, click **Programmers** > **Scan Chain** (or select the programmer in the **Programmer List Window**, right click, and then choose **Scan Chain**). See figure below.

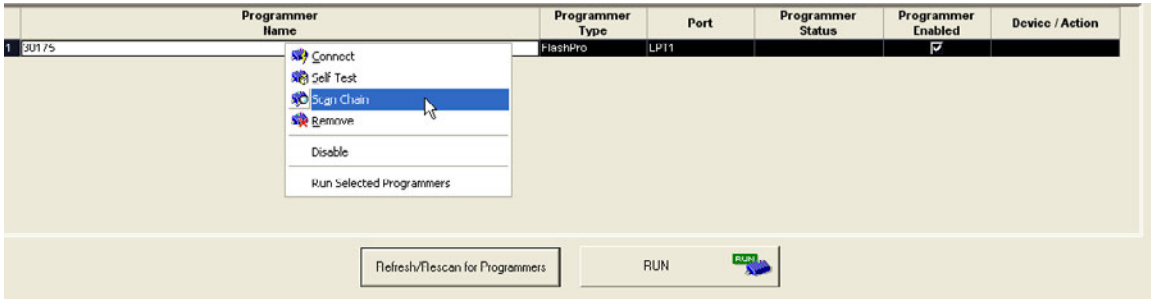


Figure 126 · Scan Chain

Scan Chain shows how the devices are ordered in the chain in the Log window (see figure below). In this example, APA300 is the first device and will be programmed first in the chain since it is connected directly to TDO.

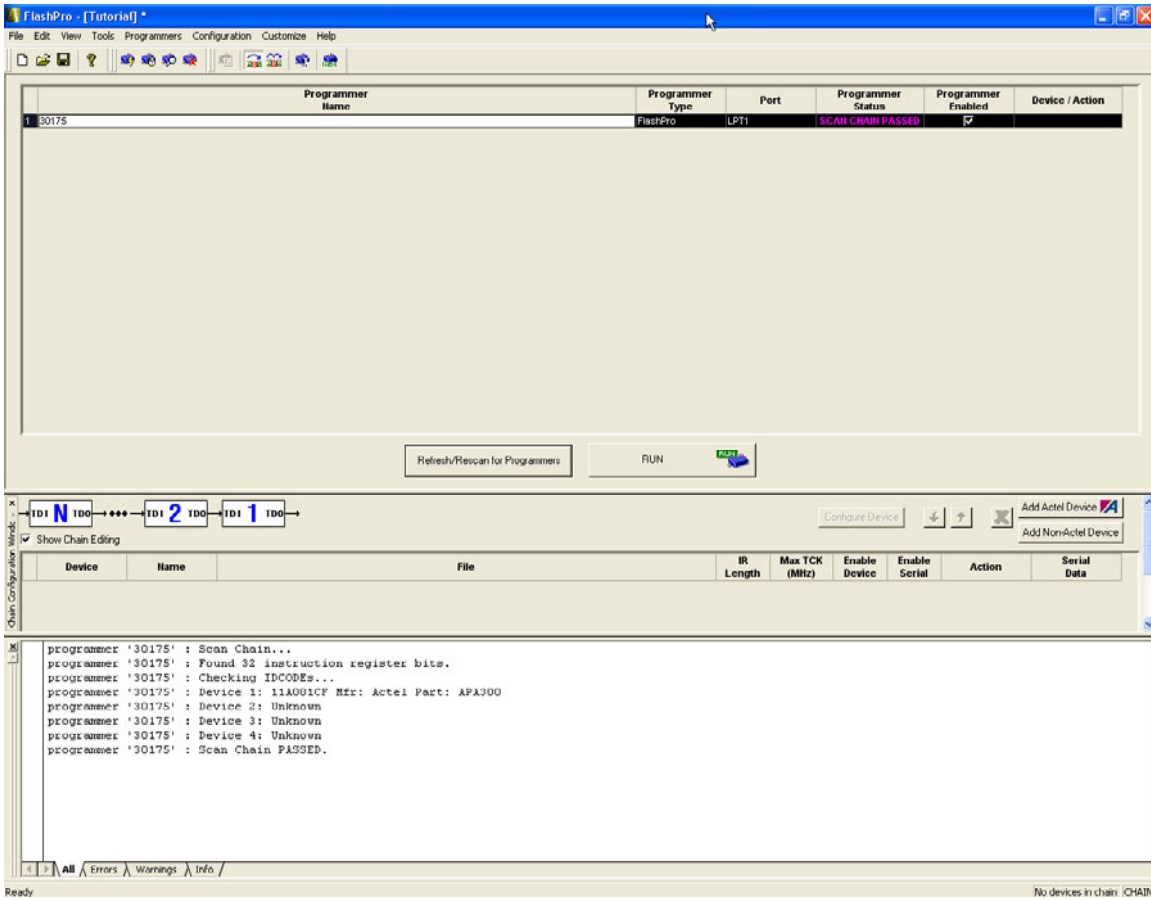

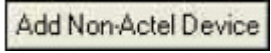


Figure 127 · Log Window Scan Chain Order

- From the Chain Configuration window, click either  or  buttons to add devices to the chain. In this example, click the Add Actel Device button because the APA300 is the first device in the chain. The Add Actel Device dialog box displays (see figure below).



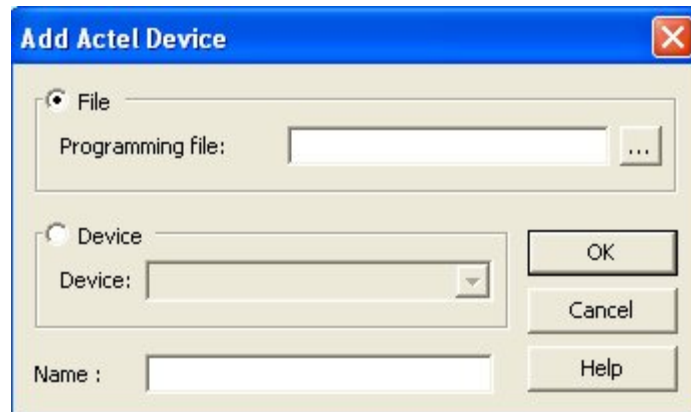


Figure 128 · Add Actel Device Window

8. Select the **File** radio button and click the **Browse** button to find your programming file.
9. Select the **Device** radio button, then choose the APA300 device from the **Device** drop-down.
10. In the **STAPL File** field, load the APA300.stp file by using the **Browse** button (...) to locate the file.
11. In the **Name** field, keep APA300 as the default name.
12. The APA300 device is added to the **Chain Configuration Window** (see figure below).

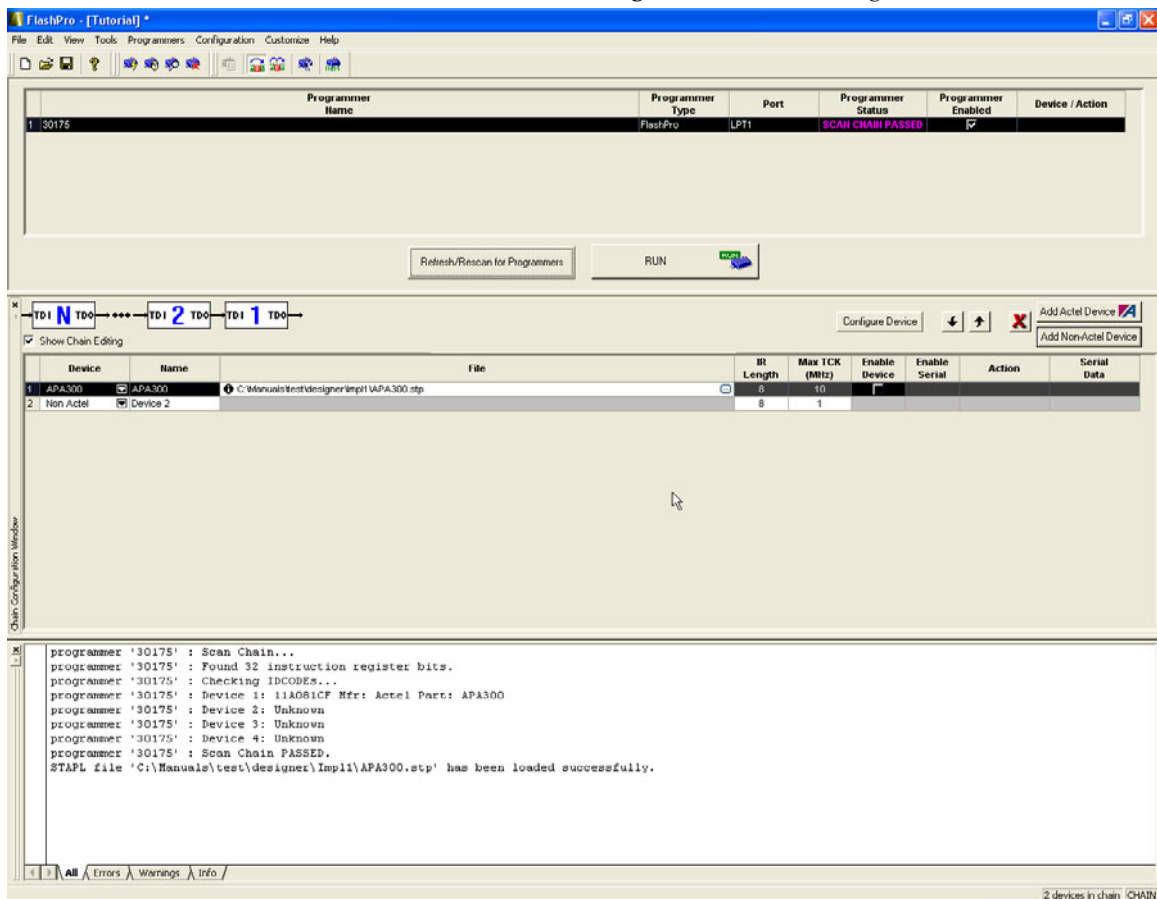


Figure 129 · Chain Configuration Window: Device One

13. Click the **Add Non-Actel Device** button to add the non-Actel device.

The **Add Non-Actel Device** dialog box appears (see figure below). You can load the BSDL file or enter the IR length and Max TCK Frequency of the device. In this tutorial, you will enter the IR length and Max TCK frequency for this device.

14. Enter the IR length and Max TCK frequency.

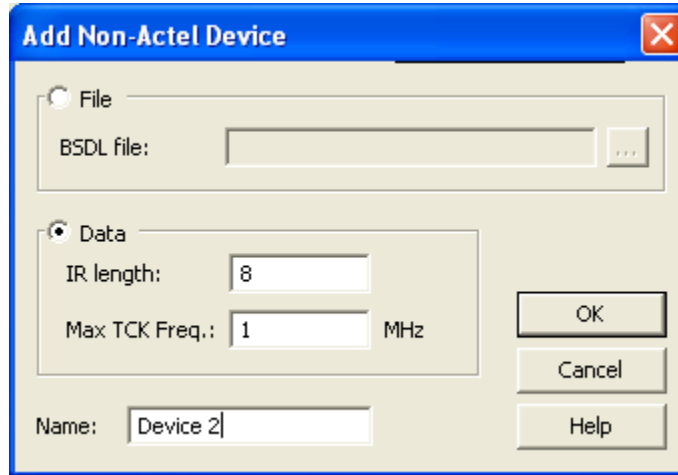


Figure 130 · Add Non-Actel Device Window

15. For this device, enter 8 in the IR length field and keep the Max TCK freq default to 1MHz.

16. Name the device, "Device 2" and click OK.

The second device now appears in the **Chain Configuration Window**. See figure below.



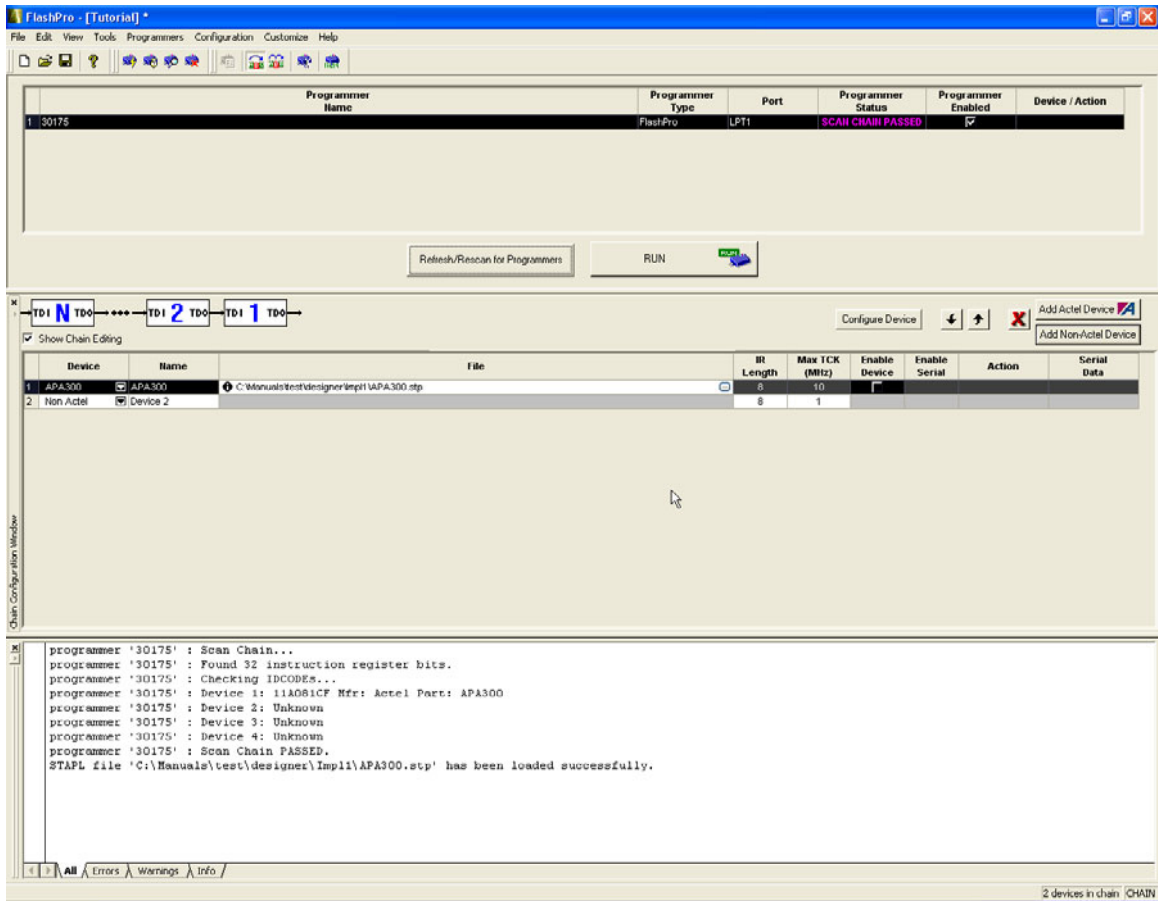


Figure 131 · Chain Configuration Window: Device Two

17. Repeat step 15 for Device 3 and Device 4.
18. Check the **Enable Device** box for the APA300 device. After you add all the devices in the chain, the **Chain Configuration Window** should look like the figure below.

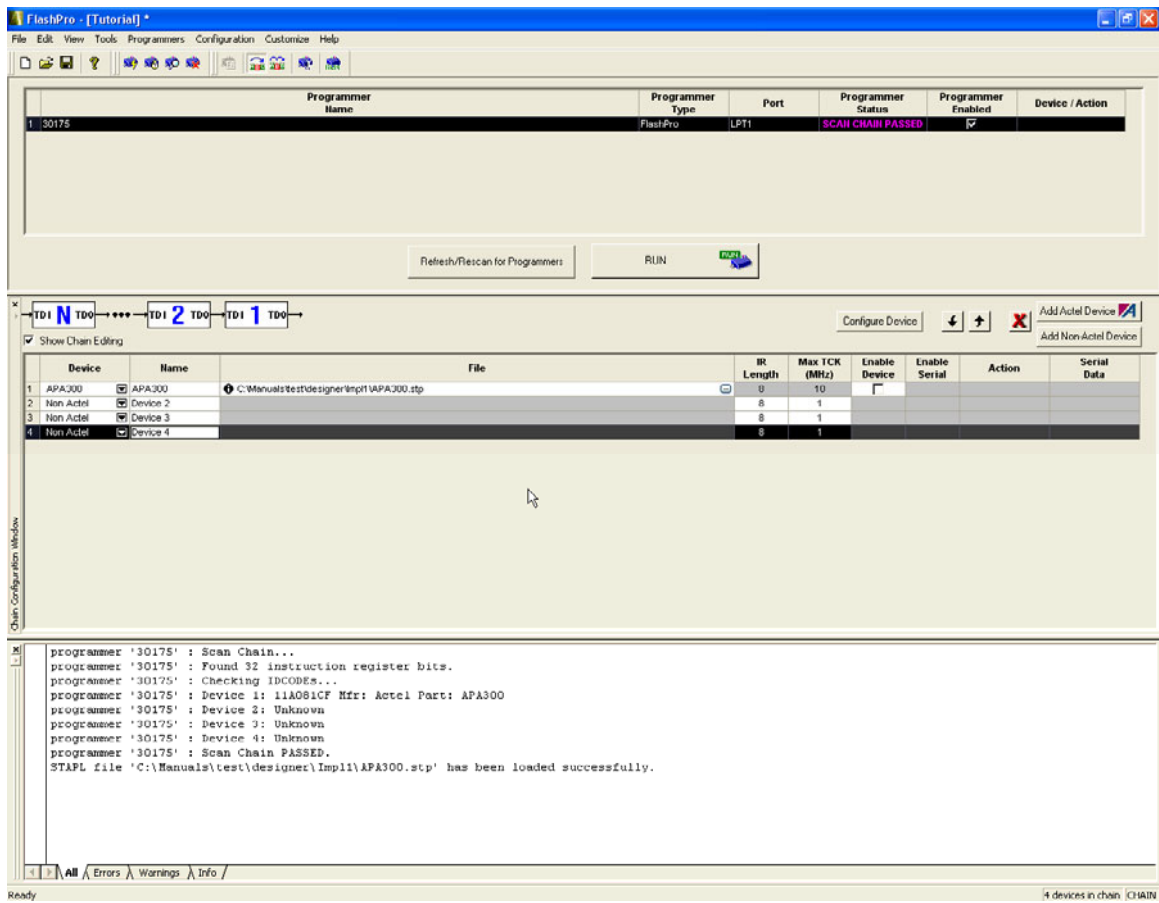


Figure 132 · Chain Configuration Window: All Devices in the Chain

19. After you have added all of the devices to the chain in the correct order, click the **Run** button from the **Flow** window to program the chain.
20. When programming is complete, the **Programmer List Window** displays (see figure below).



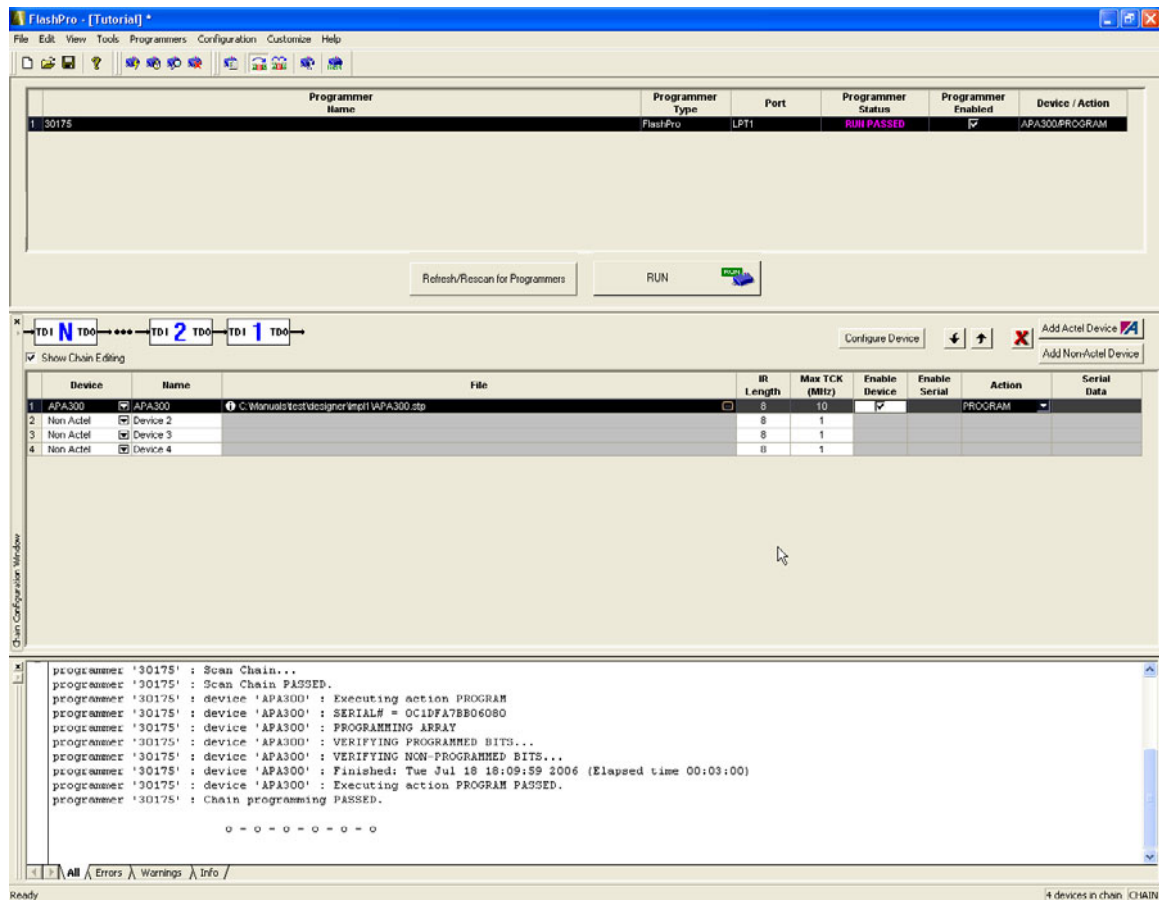


Figure 133 · Programmer List Window: Programming Complete

Congratulations, you have just completed the FlashPro Chain Programming Tutorial.

Modifying Memory Contents and Programming a Device

This tutorial provides step-by-step instructions on how to load a Program Database (PDB) file, modify the memory contents, and program the device.

Before you begin this tutorial, you should have a design with an EFMB client in it with a generated programming file for this design. You will first create a new project and title it "tutorial." If FlashPro is launched through Libero IDE, a new project will automatically be created and a PDB file will be loaded, if available.

Creating a New Project

If you are familiar with this feature, follow the basic procedures for [creating a new project](#). However, if you would like step-by-step instructions, see the creating a new project section in the [Single STAPL/PDB file basic tutorial](#).

Loading and Configuring a PDB File

Once you have created your project and connected your programmer, you are ready to load your PDB file.

To load a PDB file:

1. Click the **Configure Device** button. The **Single PDB Configuration** window displays in the FlashPro GUI.

2. Click the **Browse** button to find your PDB file.
3. From the **Load PDB File** dialog box, find your PDB file and click **Open**.

Modify Embedded Flash Memory Block Content

Now, you are ready to modify the Embedded Flash Memory Block content.

To modify Embedded Flash Memory Block content:

1. Click the PDB Configuration button to open FlashPoint.

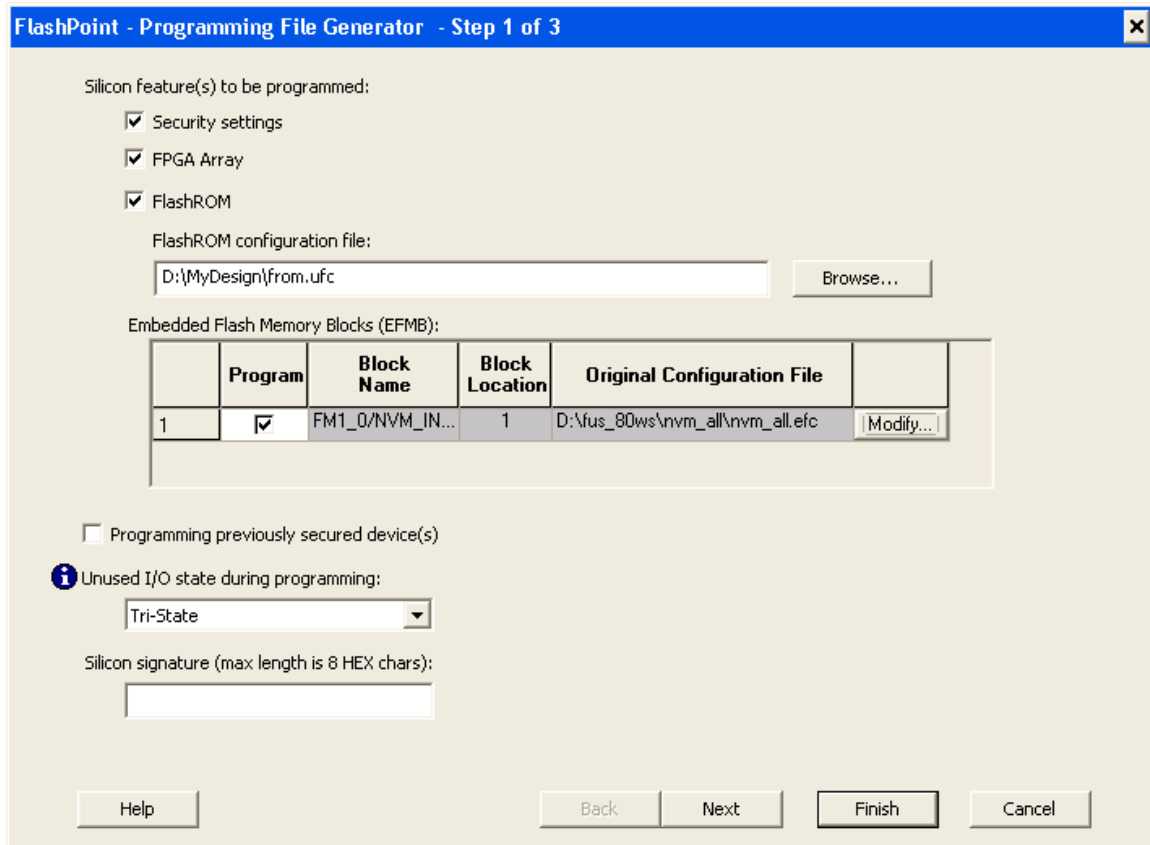


Figure 134 · Program File Generator

2. Check the **Program** box.
3. Click the **Modify** button to import Embedded Flash Memory Block configuration and memory content file. The **Modify Embedded Flash Memory Block** dialog box appears.



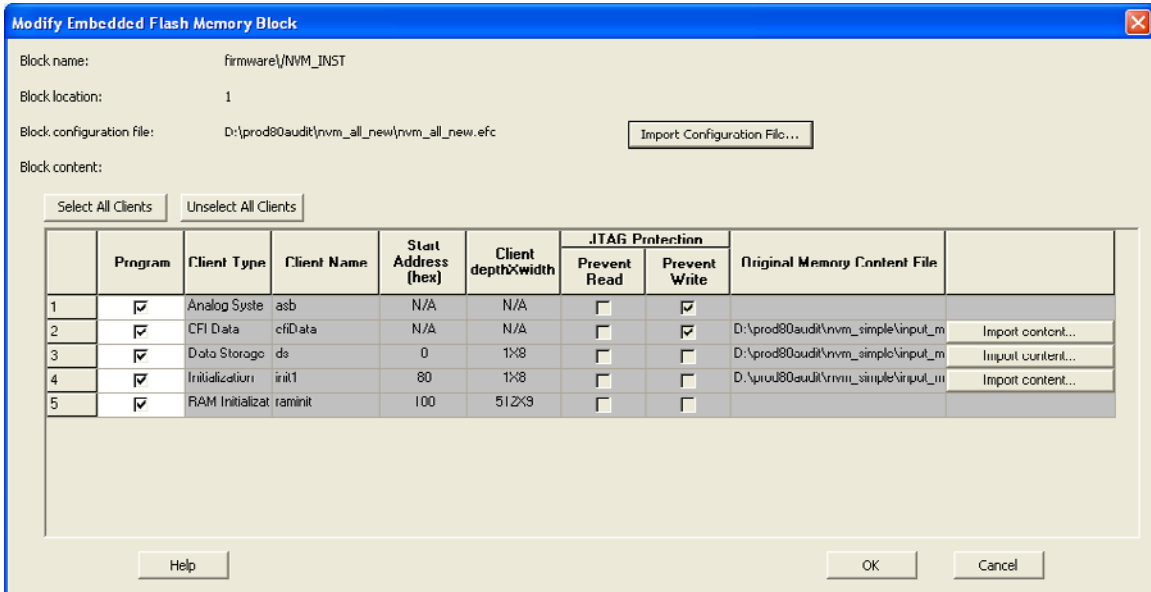


Figure 135 · Modify Embedded Flash Memory Block Content Dialog Box

4. Click the **Import Configuration File** button to import the Embedded Flash Memory Block configuration and memory content from the EFC file. This will populate the client table below. All clients that belong to this block will be selected by default.
5. Click the **Import content** button if you want to change the client memory content.
6. Click **OK**.
7. Click **Finish**.

Note: FlashPoint audits original configuration and memory content files and warns the user if the files cannot be located or if they have been updated. These files are not required as the last updated configuration and memory content is stored in the PDB.



Figure 136 · Audit Warning

Proceed to program the device. For steps on how to program a device, see the [Programming a device](#) section of the [Single STAPL/PDB file basic tutorial](#).

Modifying FlashROM Contents and Programming a Device Tutorial

This tutorial provides step-by-step instructions on how to load a Program Database (PDB) file, modify the memory contents, and program the device.

Before you begin this tutorial, you should have a design with an EFMB client in it with a generated programming file for this design. You will first create a new project and title it "tutorial." If FlashPro is launched through Libero, a new project will automatically be created and a PDB file will be loaded, if available.

Creating a New Project

If you are familiar with this feature, follow the basic procedures for [creating a new project](#). However, if you would like step-by-step instructions, see the creating a new project section in the [Single STAPL/PDB file basic tutorial](#).

Loading and Configuring a PDB File

Once you have created your project and connected your programmer, you are ready to load your PDB file.

To load a PDB file:

1. Click the **Configure Device** button. The **Single Device Configuration** Window displays in the FlashPro GUI (see figure below).



Figure 137 · Single Device Configuration Window

2. Click the **Browse** button to find your PDB file. From the **Load Programming File** dialog box, find your PDB file and click **Open**. See figure below.



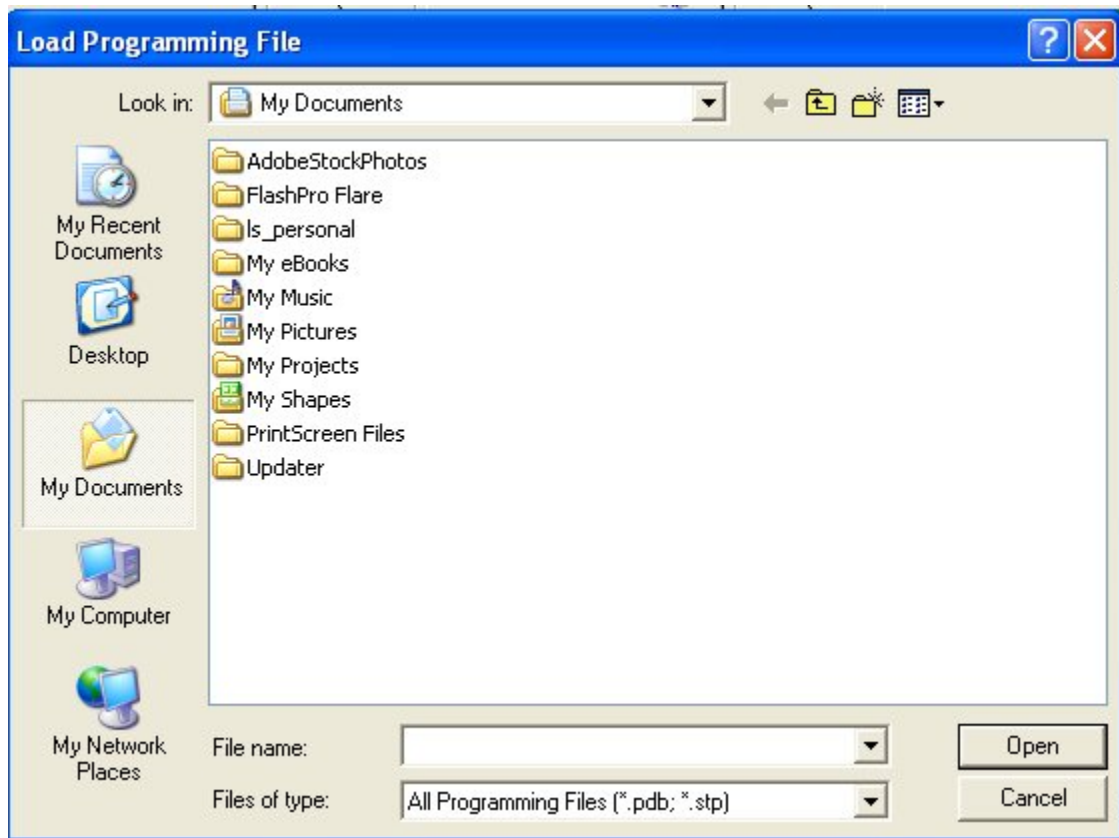


Figure 138 · Load Programming File dialog box

Modify FlashROM Content

Now you are ready to modify the FlashROM content.

1. Click the **PDB Configuration** button. This will open FlashPoint.
2. Select **FlashROM** under Silicon feature(s) to be programmed (see figure below).

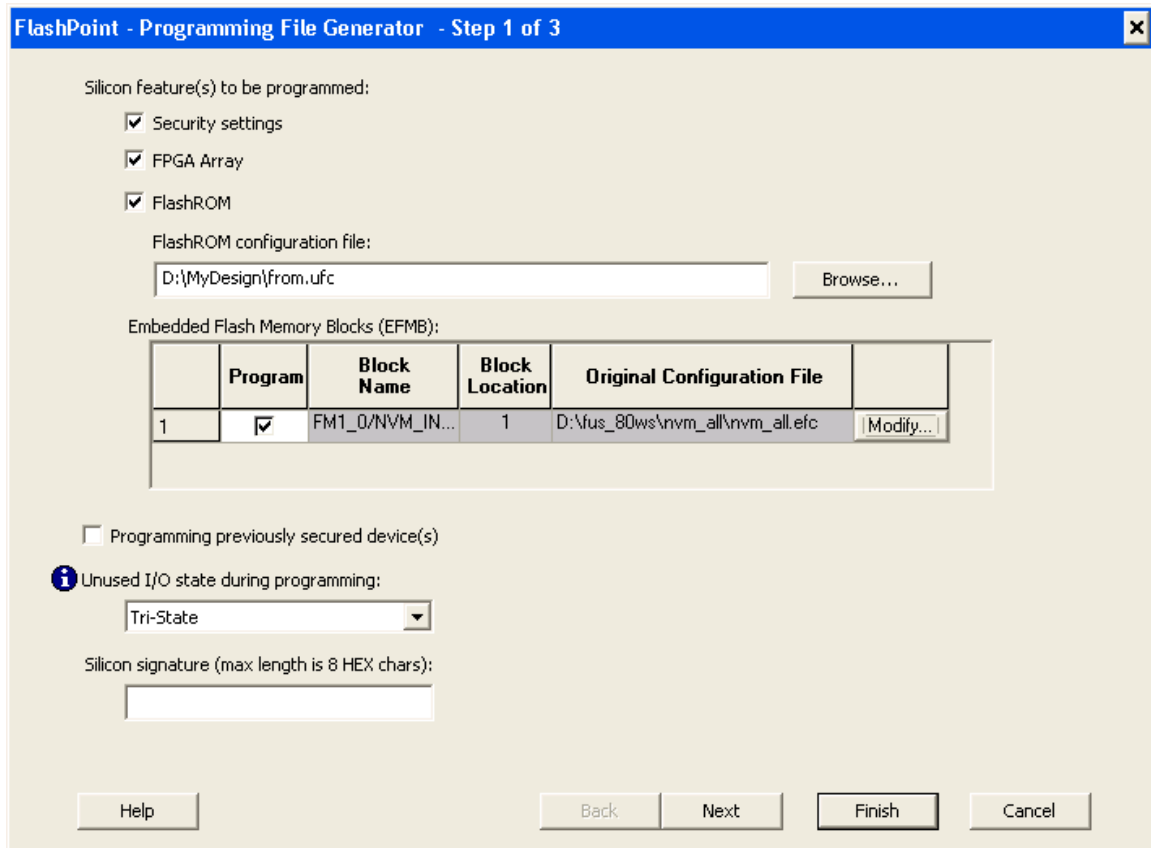


Figure 139 · FlashPoint Programming File Generator- Step 1 of 1

3. Select the .ufc FlashROM configuration file by clicking the **Browse** button and navigating to the configuration file. This file is normally present in the SmartGen subfolder of the Libero project, in a folder with the FlashROM IP block's name.
4. Click **Next**.
5. Select the FlashROM pages you want to program (see figure below).



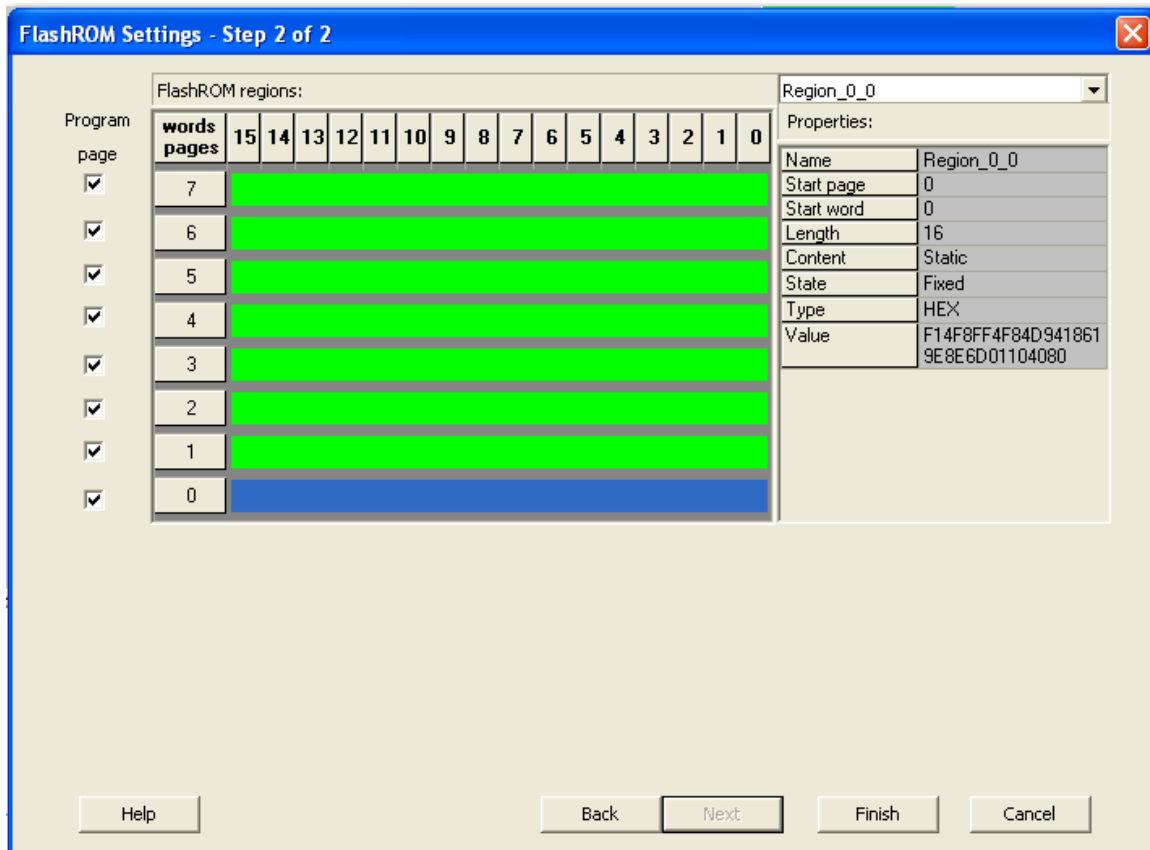


Figure 140 · FlashROM Settings Dialog Box

- Click **Finish**.

Proceed to program the device. For steps on how to program a device, see the [Programming a device](#) section of the [Single STAPL/PDB file basic tutorial](#).

Programming Only Security Settings Tutorial

This tutorial provides step-by-step instructions on how to program only the security settings into a device.

No design or PDB file is needed to follow this tutorial.

First create a new project and title it "tutorial." If FlashPro is launched through Libero, a new project will automatically be created and a PDB file will be loaded, if available. For this tutorial you always need to create a new project.

Creating a New Project

If you are familiar with this feature, follow the basic procedures for [creating a new project](#). However, if you would like step-by-step instructions, see the creating a new project section in the [Single STAPL/PDB file basic tutorial](#).

Configuring the Security Settings

Once you have created your project and connected your programmer, you are ready to load your PDB file.

To configure the security settings:

1. Click the **Configure Device** button. The **Single Device Configuration** window displays in the FlashPro GUI (see figure below).



Figure 141 · Single Device Configuration Window

2. Click **PDB Configuration**. This opens a message box indicating that no PDB was loaded for the current device (see image below).



Figure 142 · No PDB Loaded for Current Device Message

3. Click **Yes**. This automatically creates an empty PDB and opens the **Select Device** dialog box (see image below).

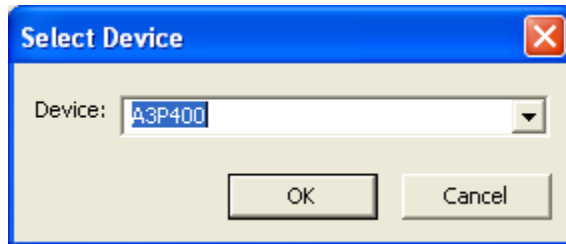


Figure 143 · Select Device Dialog Box

4. Select the desired device from the drop down list and click **OK**. FlashPoint opens. IGLOO, Fusion, and ProASIC3family devices support securing the device with a pass key as well as encrypting programming files using an AES key. Flash devices can also be permanently locked, preventing reprogramming.
5. Check the **Security Settings** checkbox to secure the unsecured device.

Note: When this box is checked, the keys will be available as plain text in the generated programming file. Unselecting **Security Settings**, but selecting **Programming previously secured device(s)** will encrypt the programming file with AES when the **High security level** is selected.

Note: **Warning:** Make a note of the security keys that you are using. Once a device is secured, it cannot be reprogrammed without those keys.



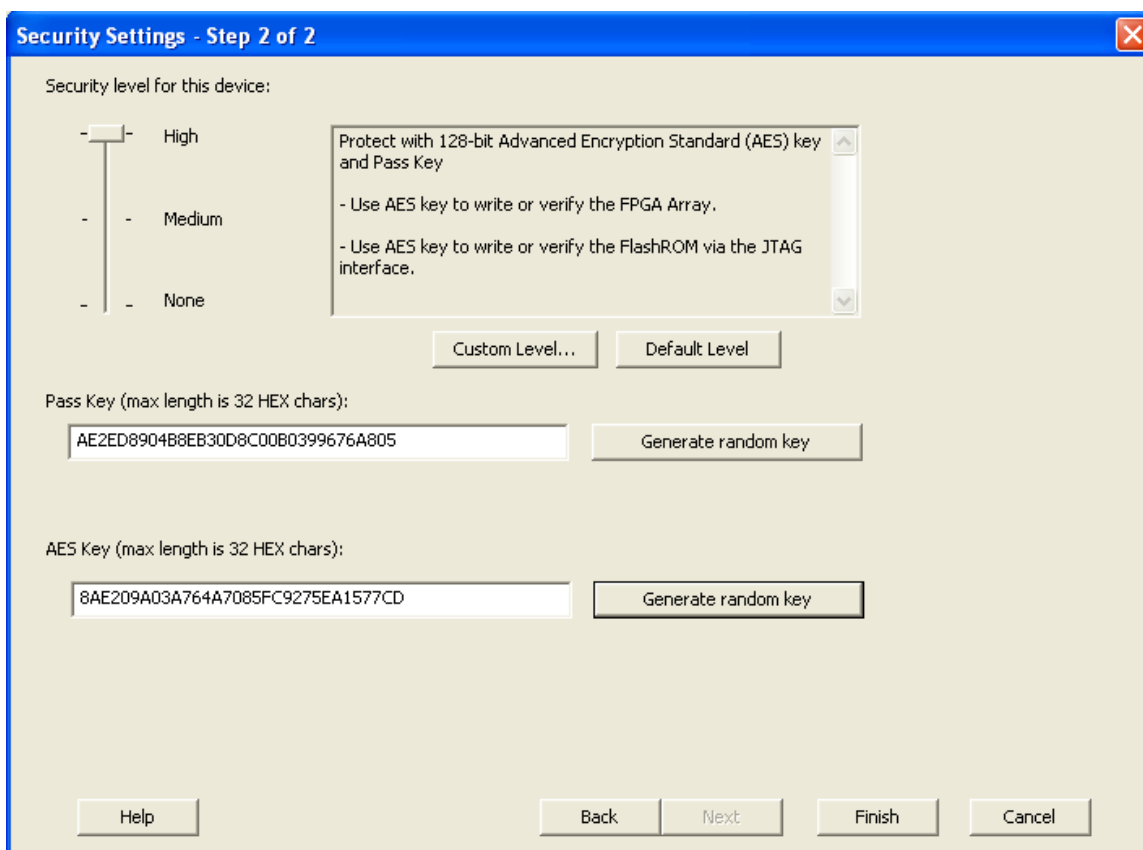


Figure 144 · Security Settings Dialog Box

6. Click **Finish**.

Proceed to program the device. For steps on how to program a device, see the [Programming a device](#) section of the [Single STAPL/PDB file basic tutorial](#).

Fusion Calibration Backup and Recovery Tutorial

This tutorial provides step-by-step instructions on how to backup and recover default calibration data on a Fusion device. It assumes that you have created a new project, connected your programmer, and loaded a Fusion PDB/STAPL file created in Designer v8.4 or above.

If you would like step-by-step instructions on how to create a new project, see the Creating a New Project section in the [FlashPro Single STAPL Basic tutorial](#).

If you would like step-by-step instructions on loading a programming file, see the Loading and Configuring a Programming File section in the [FlashPro Single STAPL Basic tutorial](#).

Note: This feature is only supported in STAPL and PDB programming files.

Backing Up Default Fusion Calibration Data

A backup copy of the Fusion calibration data is created once after ANY programming ACTION, except READ_IDCODE, is executed. The copy will be stored in the spare pages of eNVM. The FlashPro Log window shows that a backup copy of the calibration data has been created. See figure below.

```
programmer '07898' : Scan Chain...
programmer '07898' : Scan Chain PASSED.
programmer '07898' : Executing action PROGRAM
programmer '07898' : Checking for Backup Calibration Data...
programmer '07898' : Reading Master Calibration Data...
programmer '07898' : Writing Calibration Backup Copy
programmer '07898' : Erase ...
programmer '07898' : Completed erase
programmer '07898' : Programming FPGA Array
```

Figure 145 · FlashPro Log Window

Recovering Default Fusion Calibration Data

1. Load the PDB/STAPL file created in Designer v8.4 or above.
2. From the FlashPro Configuration window, click **Advanced** and select **RECOVER_CALIB**. See figure below.

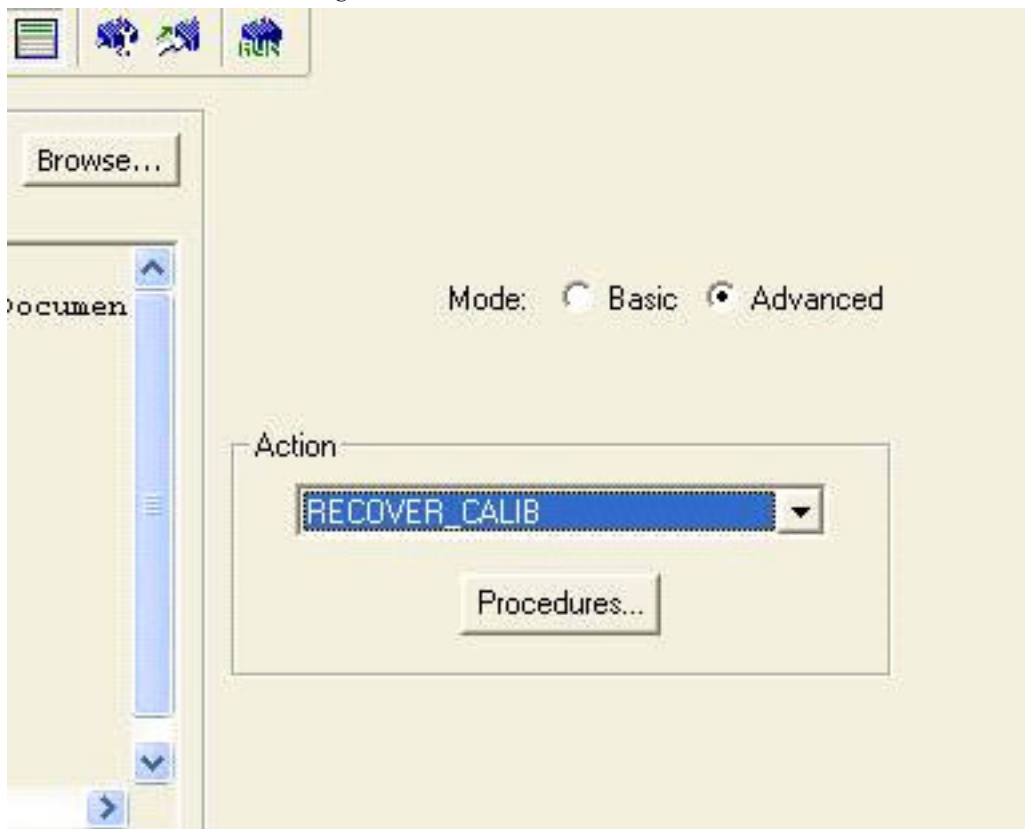


Figure 146 · RECOVER_CALIB

3. Click **Run** to restore the original Fusion calibration data. The Log window shows the data is restored. See figure below.



```
programmer '07898' : Scan Chain...
programmer '07898' : Scan Chain PASSED.
programmer '07898' : Executing action RECOVER_CALIB
programmer '07898' : Checking for Backup Calibration Data...
programmer '07898' : Reading Master Calibration Data...
programmer '07898' : Writing Calibration Backup Copy
programmer '07898' : Checking for Backup Calibration Data...
programmer '07898' : Restoring Master Calibration Data.
```

Figure 147 · Restoring Original Calibration Data

Note: The Calibration data can only be restored after a backup has been made.

Advanced Tutorials

Multiple Device Chain Programming

This tutorial provides step-by-step instructions on how to program multiple Actel devices in a chain. You should already be familiar with the basic features of the FlashPro software.

Note: This tutorial does not provide software installation instructions. Please have FlashPro already installed before you begin. If you have any questions regarding software installation, see [Software Installation](#).

In the figure below, there are three devices in a chain (two A3P250 and one A3PE600). In this section, we will program these three devices in the chain.

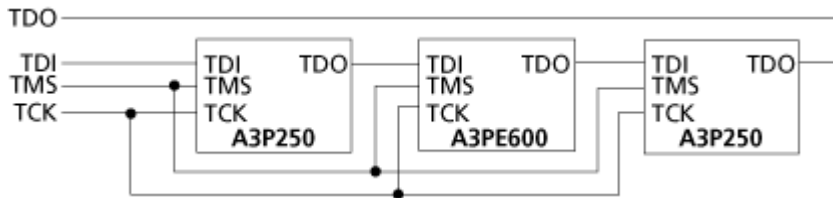


Figure 148 · APA Device Tutorial Example

First you need to create a project.

To create a new project:

1. Click the New Project button from the FlashPro GUI.
2. From the New Project dialog box, type “Tutorial” in the Project Name field.
3. Check the Chain box. See figure below.

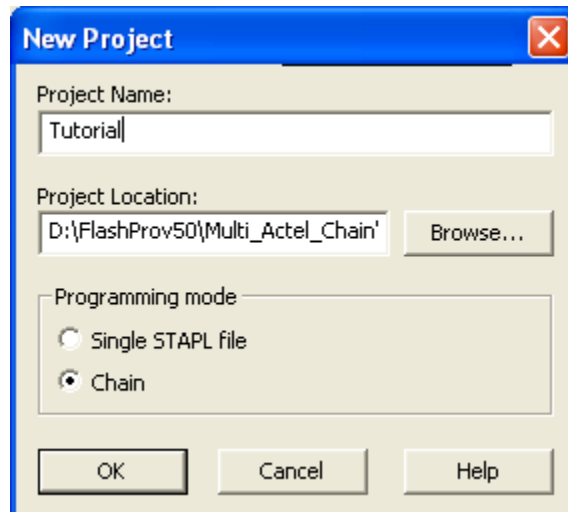


Figure 149 · New Project Dialog Box

4. If necessary, change the default location of your project in the Project Location field.
5. Click OK. The FlashPro GUI displays (see figure below).

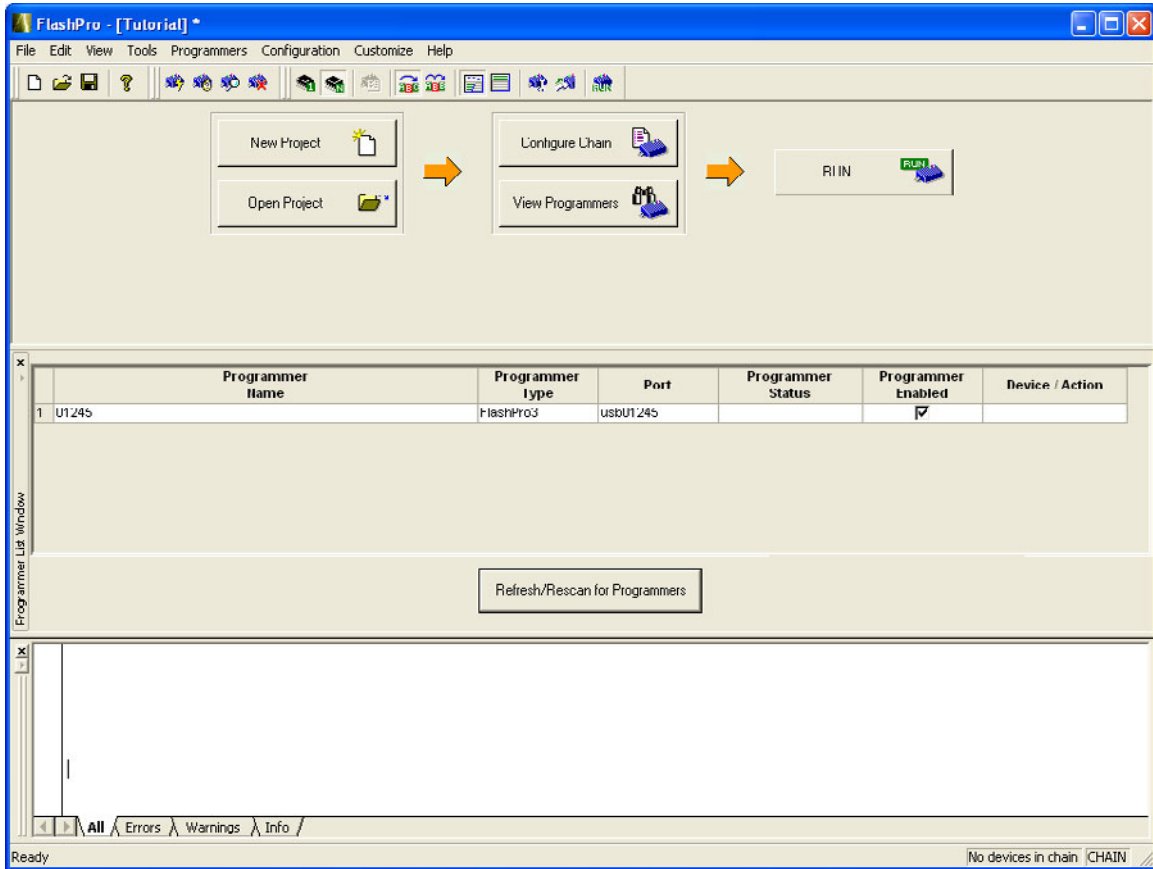


Figure 150 · FlashPro User Interface

Note: The **Programmer List** window updates with your programmers information.

- From the **Programmers** menu, choose **Scan Chain** (or select the programmer in the **Programmer List** window, right-click, then choose **Scan Chain**) (see figure below).

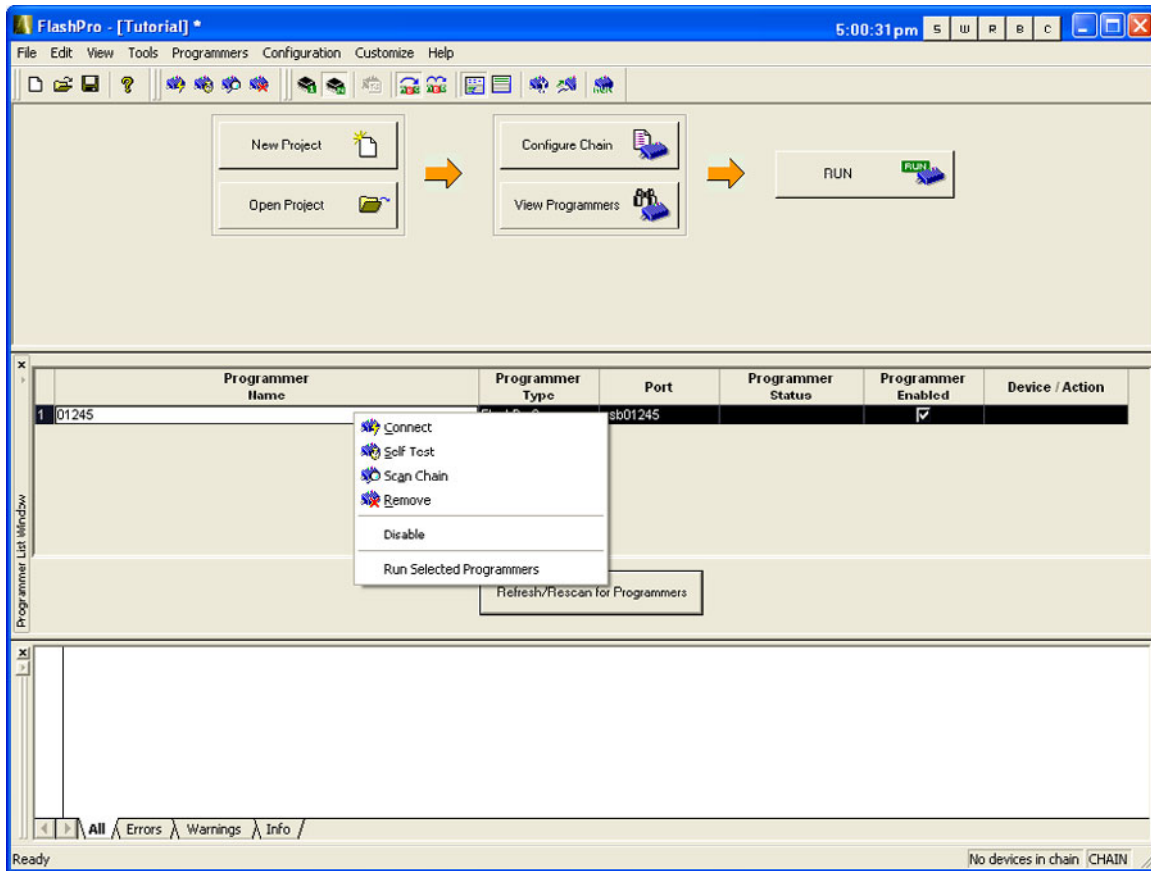


Figure 151 · Select Programmer Window

Scan Chain shows how the devices are ordered in the chain in the log window (see figure below). In this case, A3P250 is the first device that will be programmed in the chain since it is connected directly to TDO.

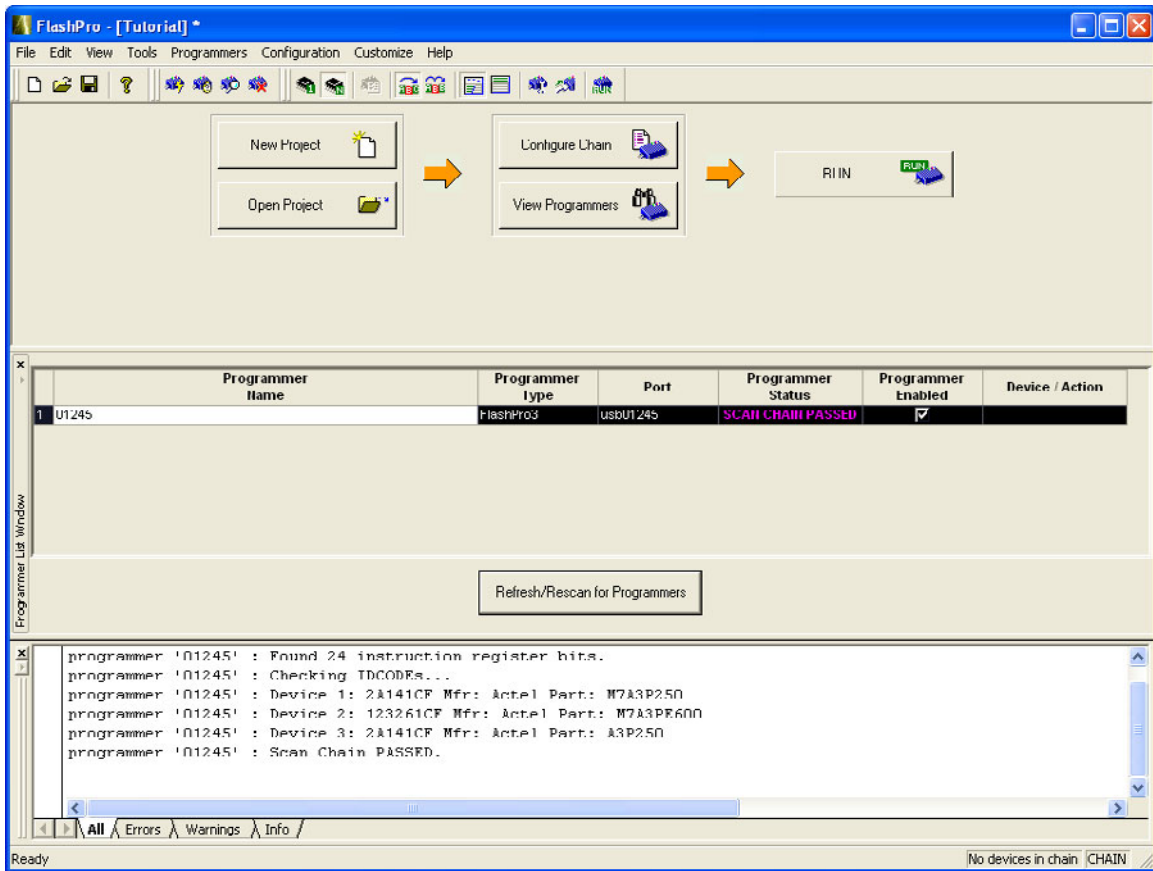


Figure 152 · Scan Chain Order in the Log Window

7. Click the **Configure Chain** button. The **Chain Configuration** window displays (see figure below).



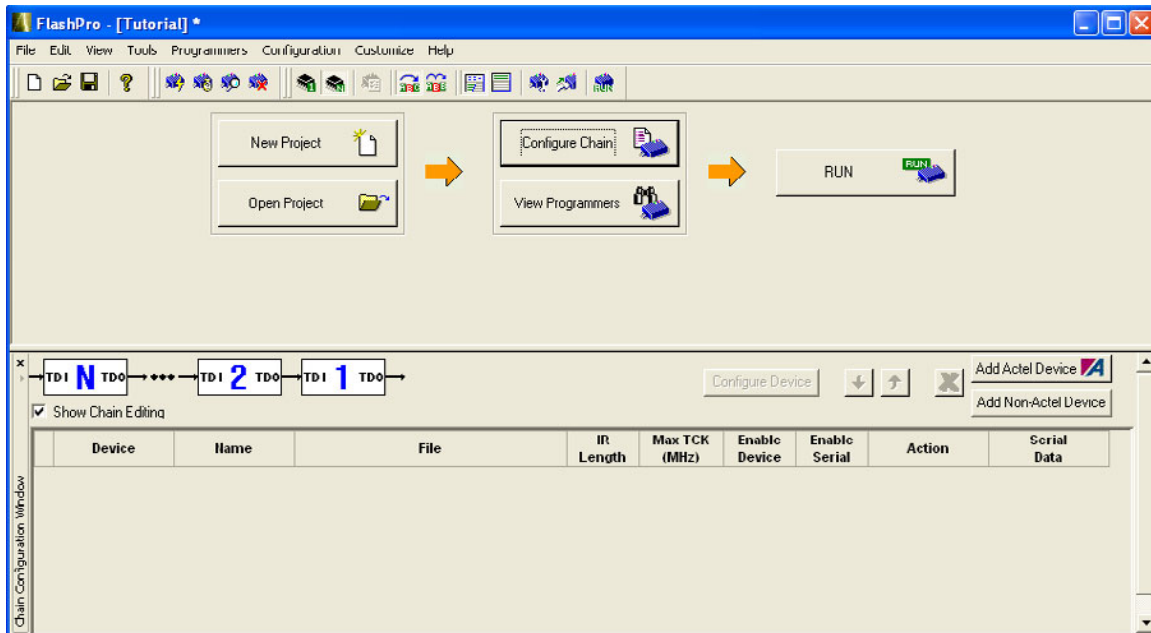


Figure 153 · Chain Configuration Window

- In the **Chain Configuration** window, click the **Add Device** button to add devices to the chain. The **Add Actel Device** dialog box displays (see figure below).

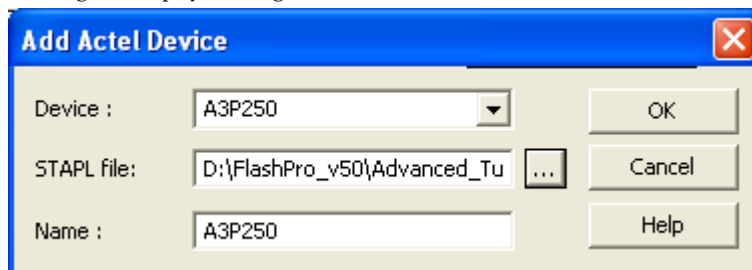


Figure 154 · Add Actel Device Dialog Box

- Choose the "A3P250" device from the **Device** drop-down.
- In the **STAPL file** field, use the **Browse** button to locate the A3P250.stp file.
- In the **Name** field, leave A3P250 as default. The A3P250 device is added into the **Chain Configuration** window (see figure below).

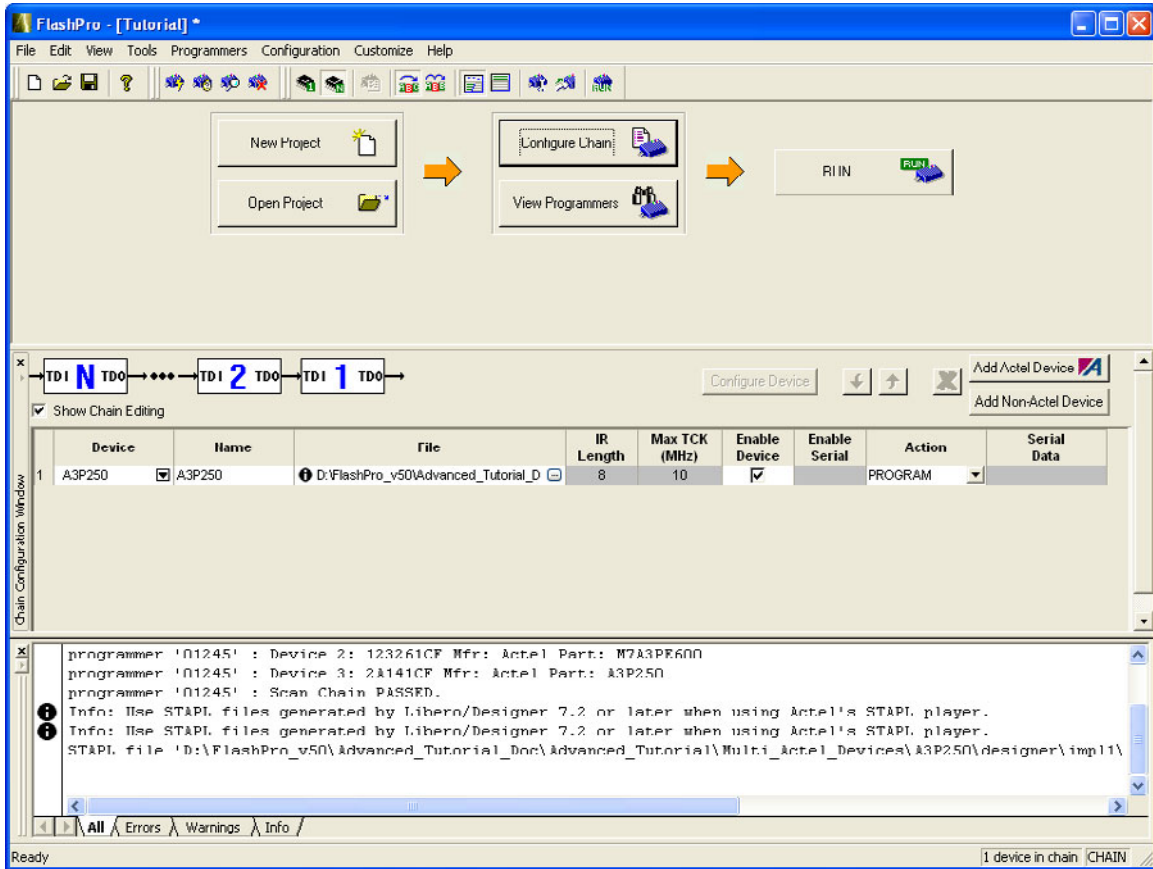


Figure 155 · Device One Chain Configuration Window

12. Repeat the same process for A3PE600 and the other A3P250 respectively.
13. After you have finished adding all of the devices in the chain, the Chain Configuration window updates.



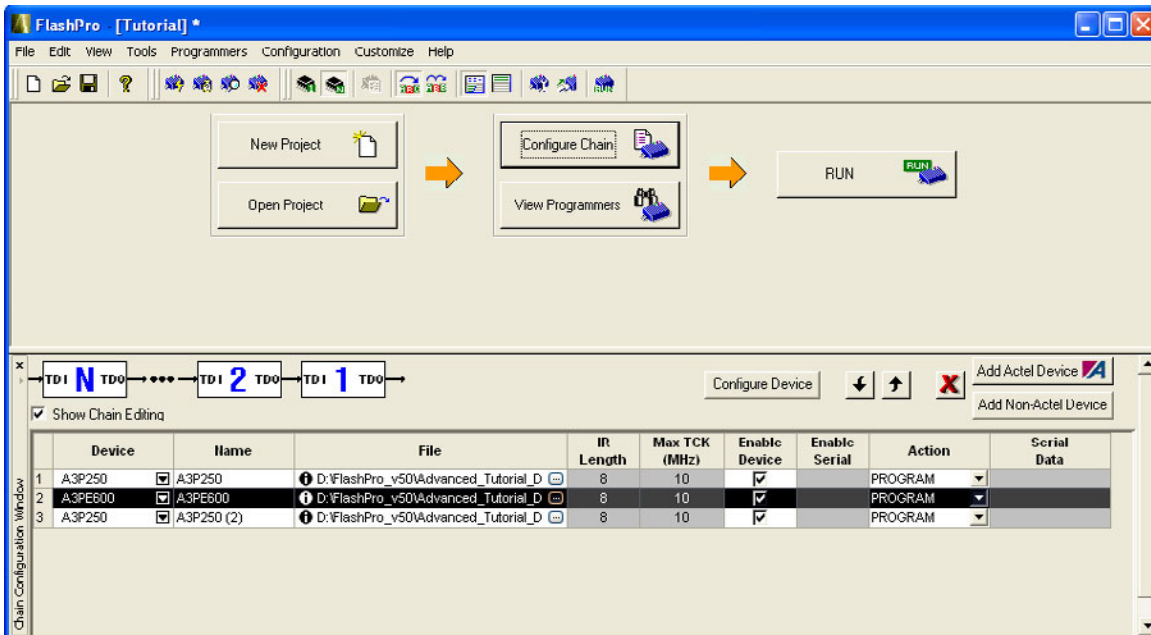


Figure 156 · Chain Configuration Window: All Devices in the Chain

14. Once all the devices have been added to the chain in the correct order, click the **Run** button to program the chain.
15. When Programming is complete, the **Programmer List** window displays. See figure below.

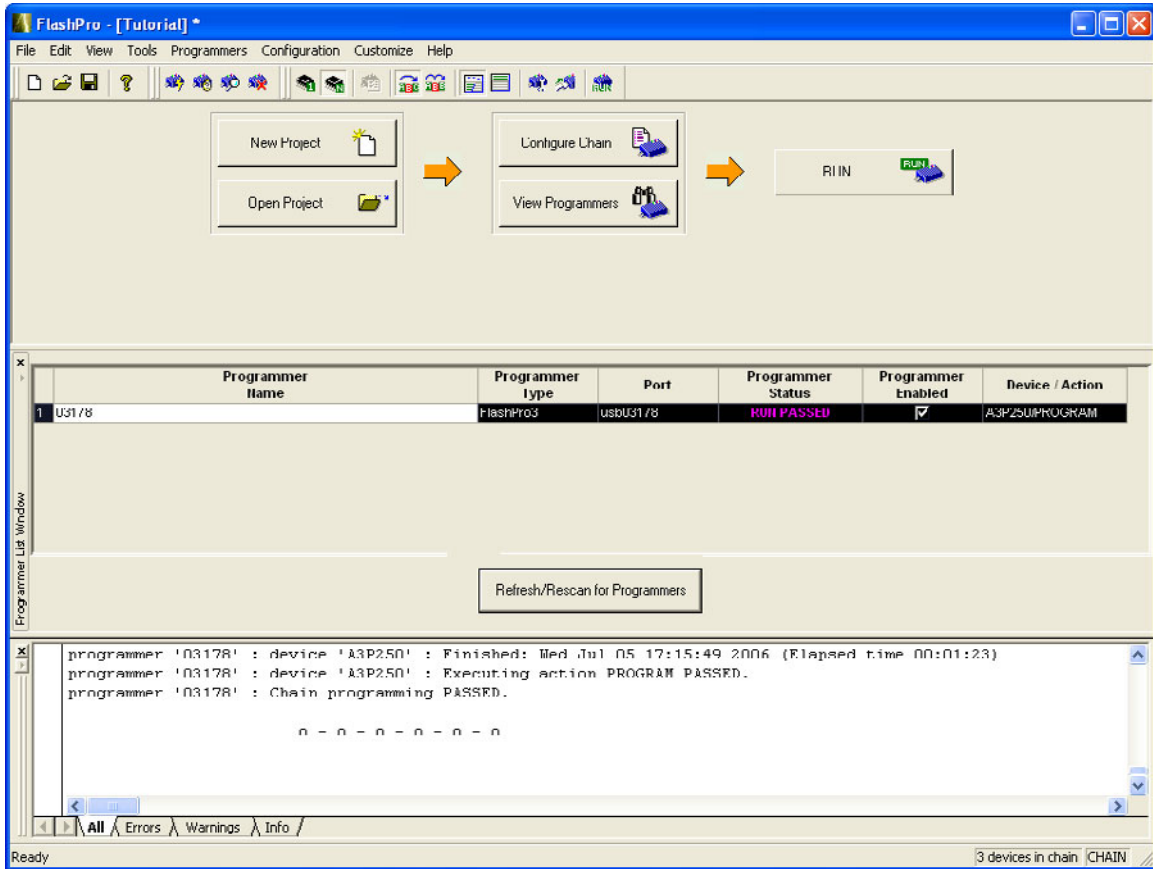


Figure 157 · Programmer List Window Done

Congratulations! You have just completed the FlashPro Multiple Actel Device Chain Programming tutorial.

Multiple Device Serialization Chain Programming

This tutorial provides step-by-step instructions on how to program multiple Actel devices with serialization. Before you begin this tutorial, you should already be familiar with the basic features of the FlashPro software.

Note: This tutorial does not provide software installation instructions. Please have FlashPro already installed before you begin. If you have any questions regarding software installation, see [Software Installation](#).

In this tutorial you will program two devices in a chain (one device is A3P250 and the other is A3PE600). The STAPL file for the first A3P250 device contains 10 serialization data. See figure below.

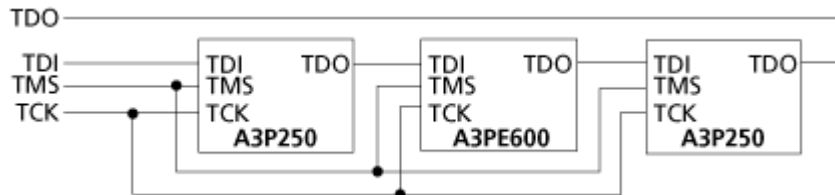


Figure 158 · APA Device Tutorial Example

First you need to create a project.



To create a new project:

1. Click the **New Project** button from the FlashPro GUI.
2. From the **New Project** dialog box, type “Tutorial” in the **Project Name** field.
3. Check the **Chain** box. See figure below.

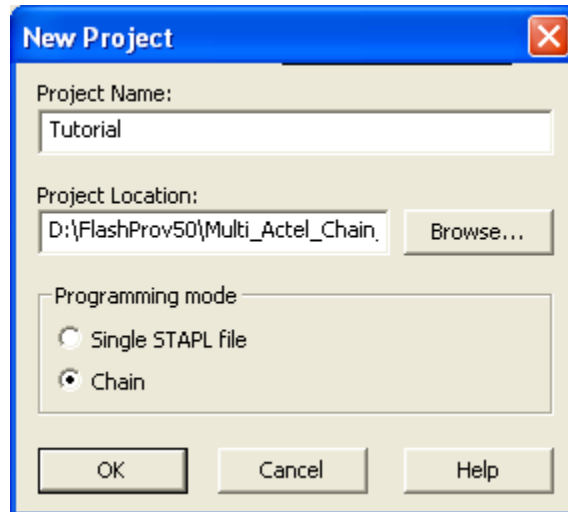


Figure 159 · New Project Dialog Box

4. If necessary, change the default location of your project in the **Project Location** field.
5. Click **OK**. The FlashPro GUI displays (see figure below).

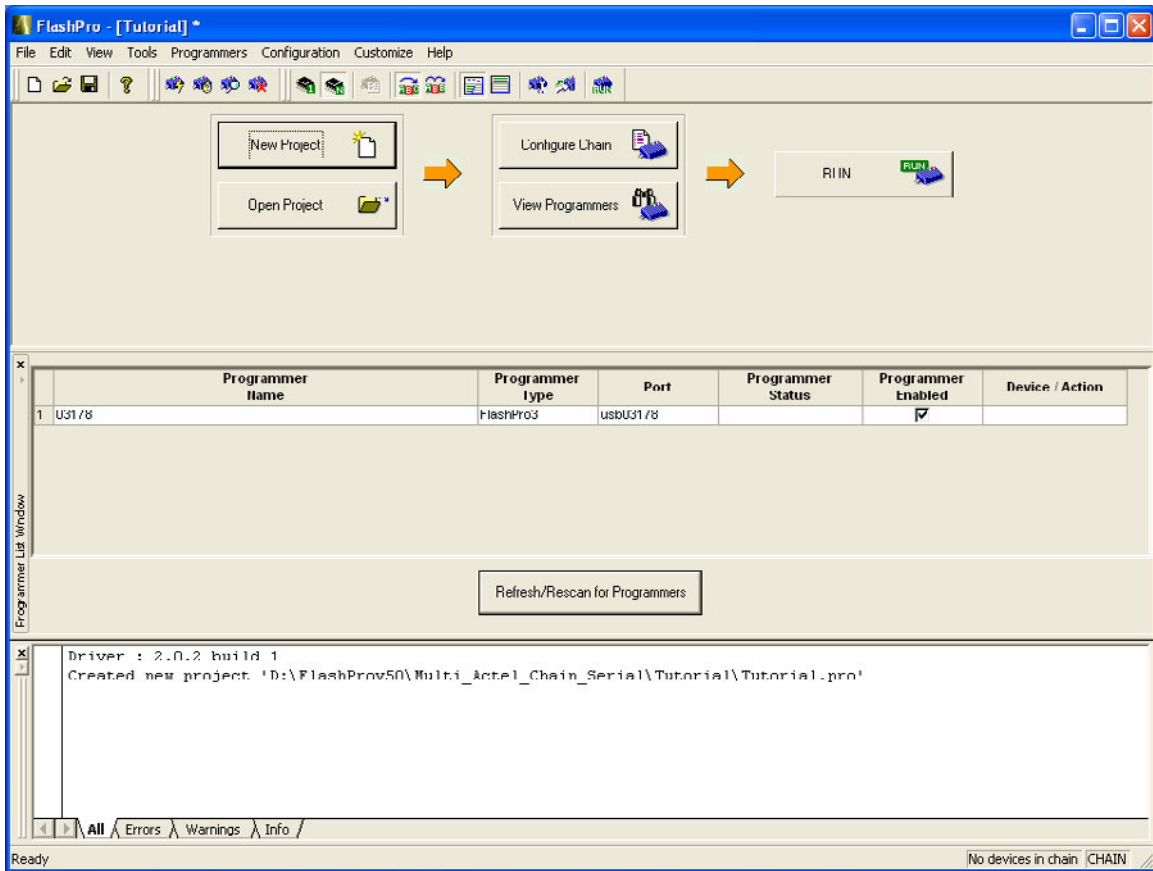


Figure 160 · FlashPro User Interface

Note: The Programmer List window updates with your programmer information.

- From the **Programmers** menu, choose **Scan Chain** (or select the programmer in the Programmer List window, right-click, then choose **Scan Chain**) (see figure below).



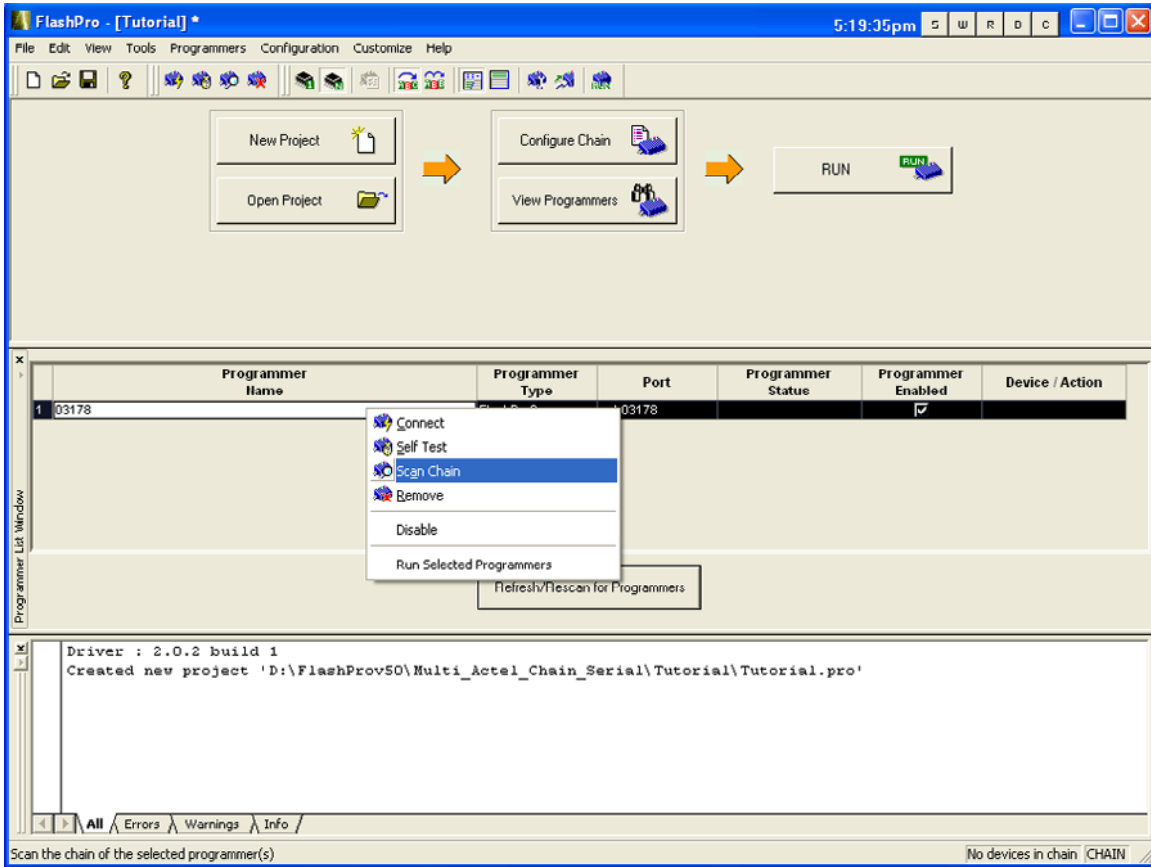


Figure 161 · Scan Chain Selection

Scan Chain shows how the devices are ordered in the chain in the log window (see figure below). In this case, A3P250 is the first device will be programmed in the chain since it is connected directly to TDO.

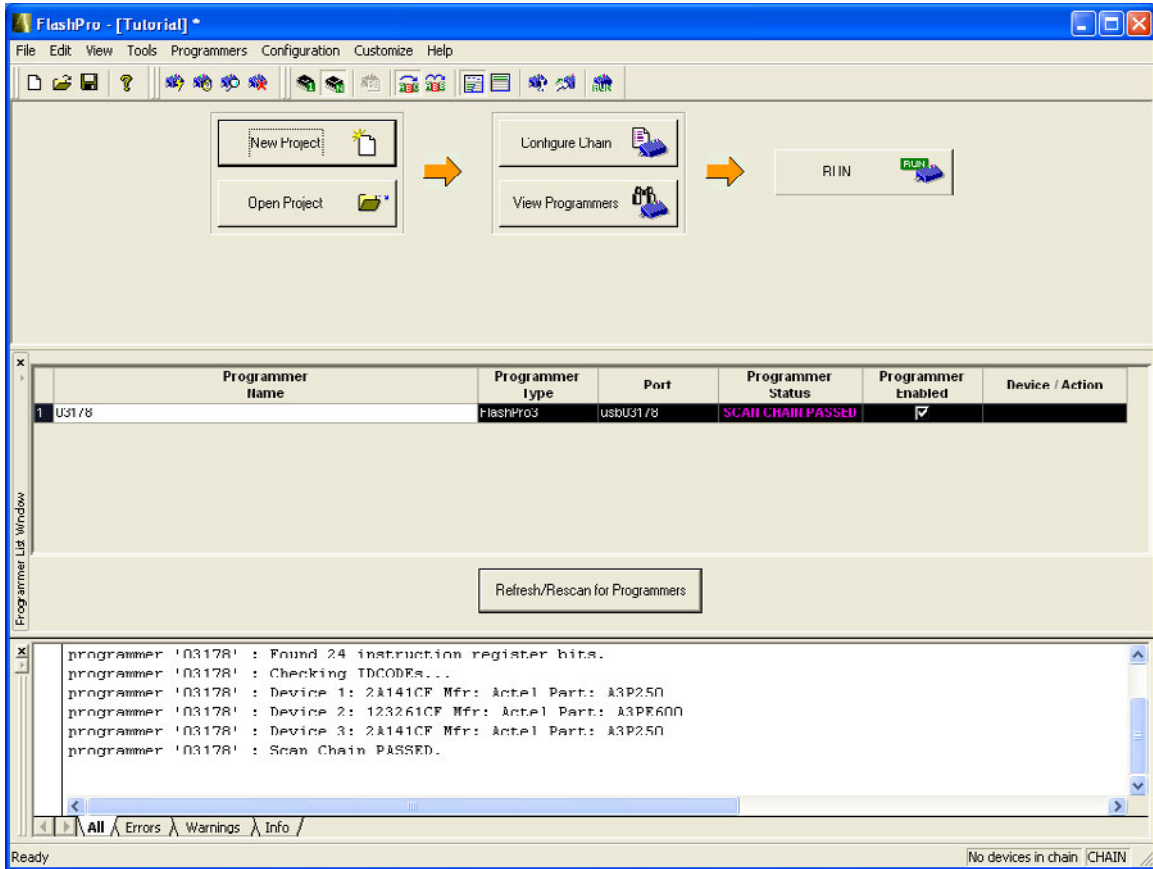


Figure 162 · Scan Chain Order in the Log Window



- Click the **Configure Chain** button

The **Chain Configuration** window displays (see figure below).



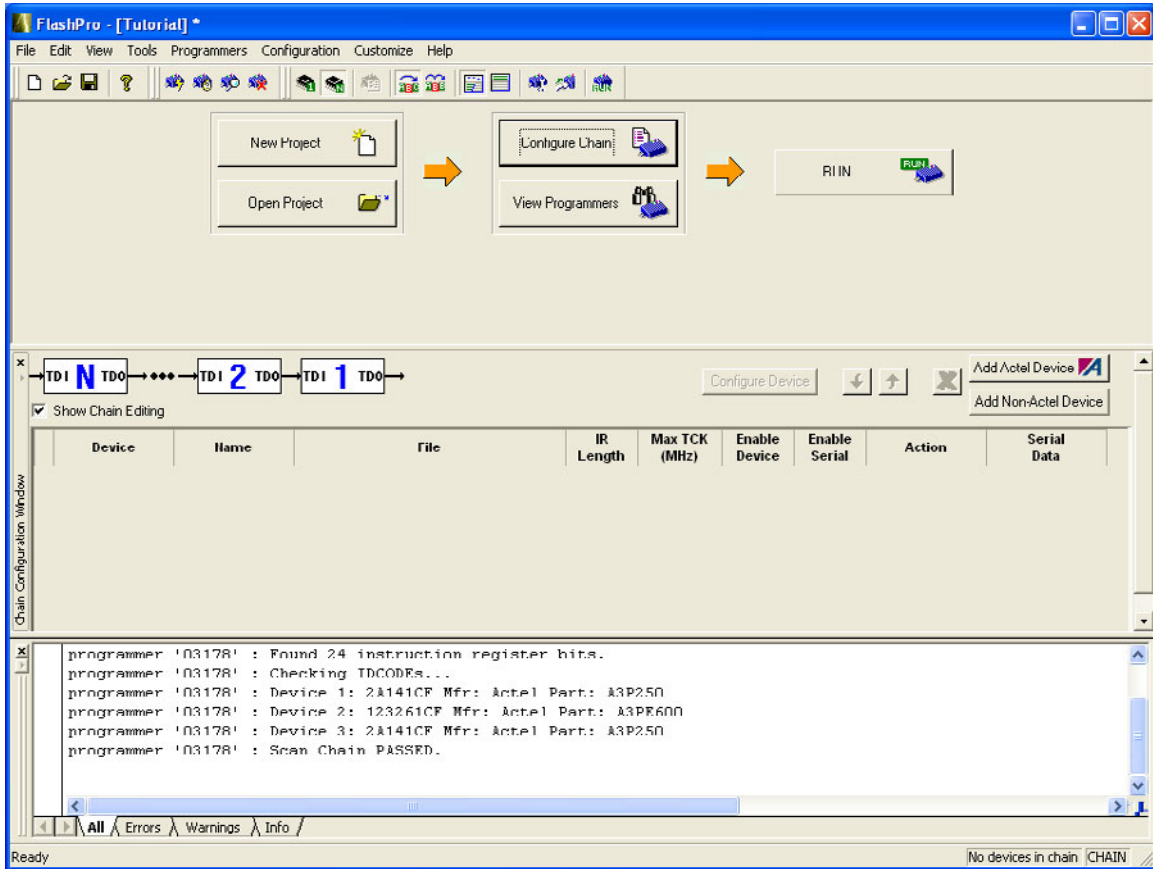



Figure 163 · Chain Configuration Window

8. In the Chain Configuration window, click the Add Device  button to add devices to the chain. The Add Actel Device dialog box displays.
9. Choose "A3P250" device from the Device drop-down. (See figure below).

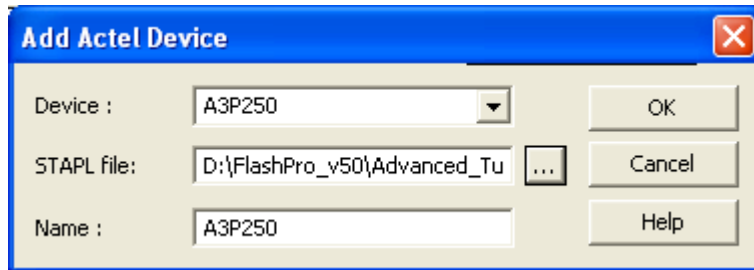


Figure 164 · Add Actel Device Dialog Box

10. In the STAPL file field, use the Browse button to locate the A3P250.stp file.
11. In the Name field, leave "A3P250" as default.
12. The A3P250 device is added into the Chain Configuration window.
13. Repeat steps 8 to 11 for A3PE600 and A3P250 respectively. After you're finished adding all devices in the chain, the Chain Configuration window updates (see figure below).

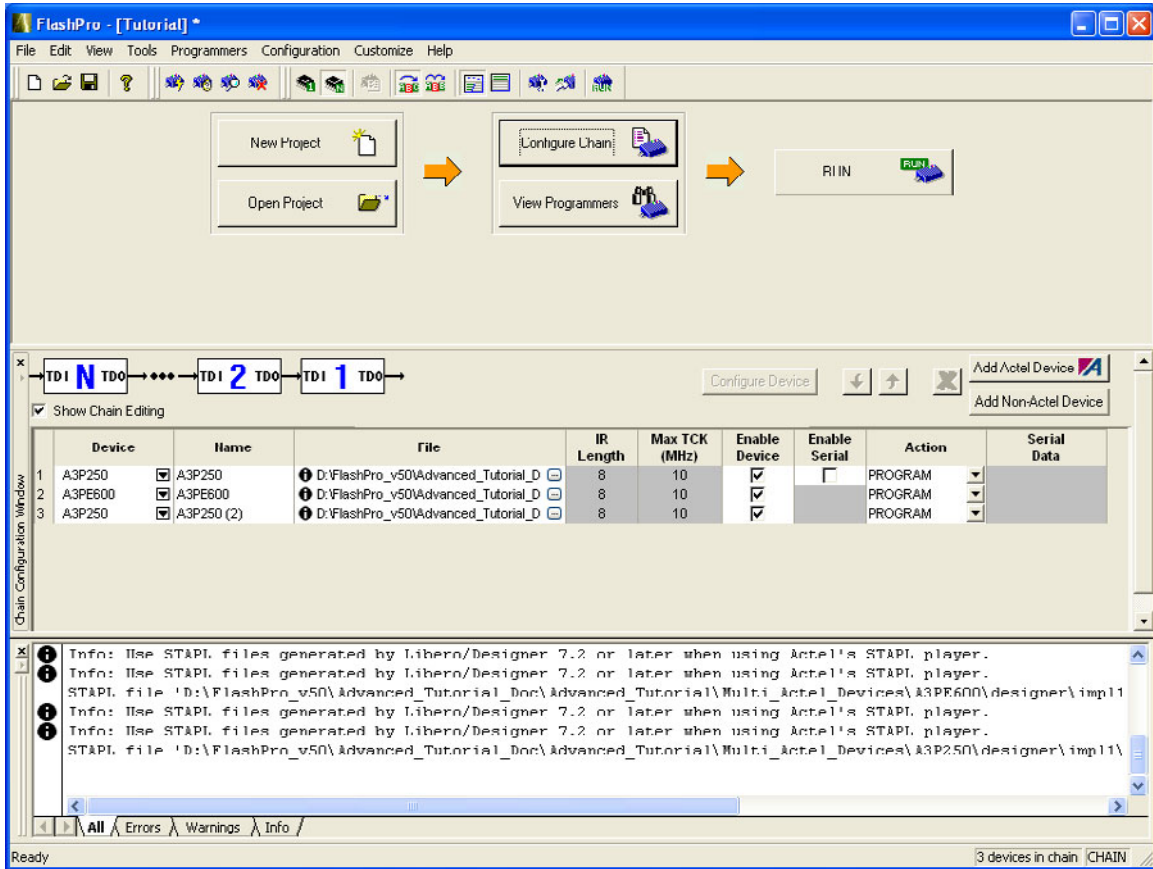


Figure 165 · Chain Configuration Window for all Devices

- From the Chain Configuration Window, check the Enable Serial box and the Chain Configuration window displays (see figure below).

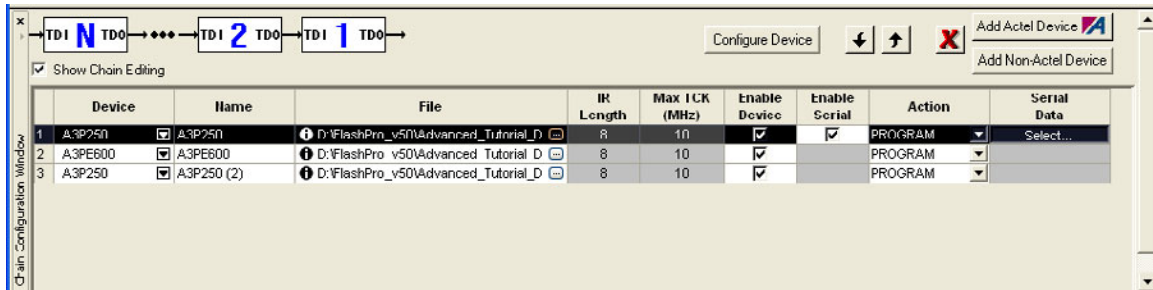


Figure 166 · Chain Configuration Window with Enable Serial Selected

- Click Select in the Serial Data column, the Serial Settings dialog box displays as shown in the figure below.



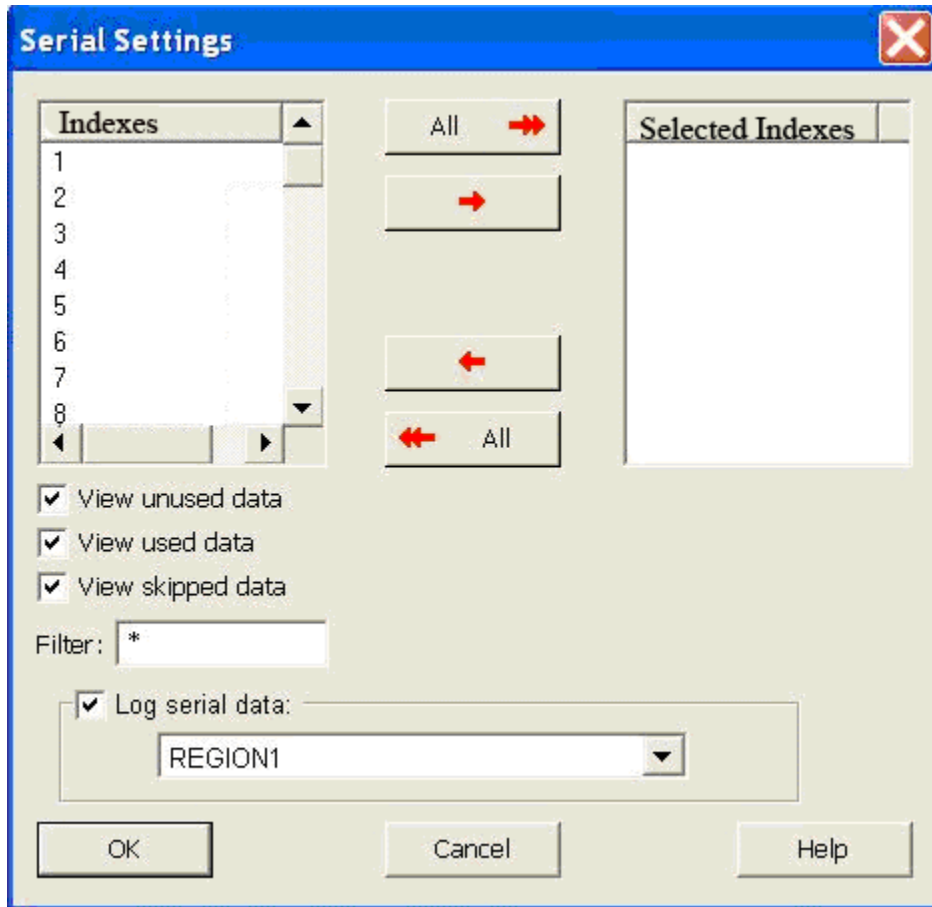


Figure 167 · Serial Settings Dialog Box

16. From the **Serial Settings** dialog box, click the **All** button to select all the serial data.
17. Click **OK**.
18. Once all the devices have been added to the chain in the correct order and serialization has been selected, click the



Run button to program the chain.

19. When programming is complete, the **Programmer List** window displays (see figure below).

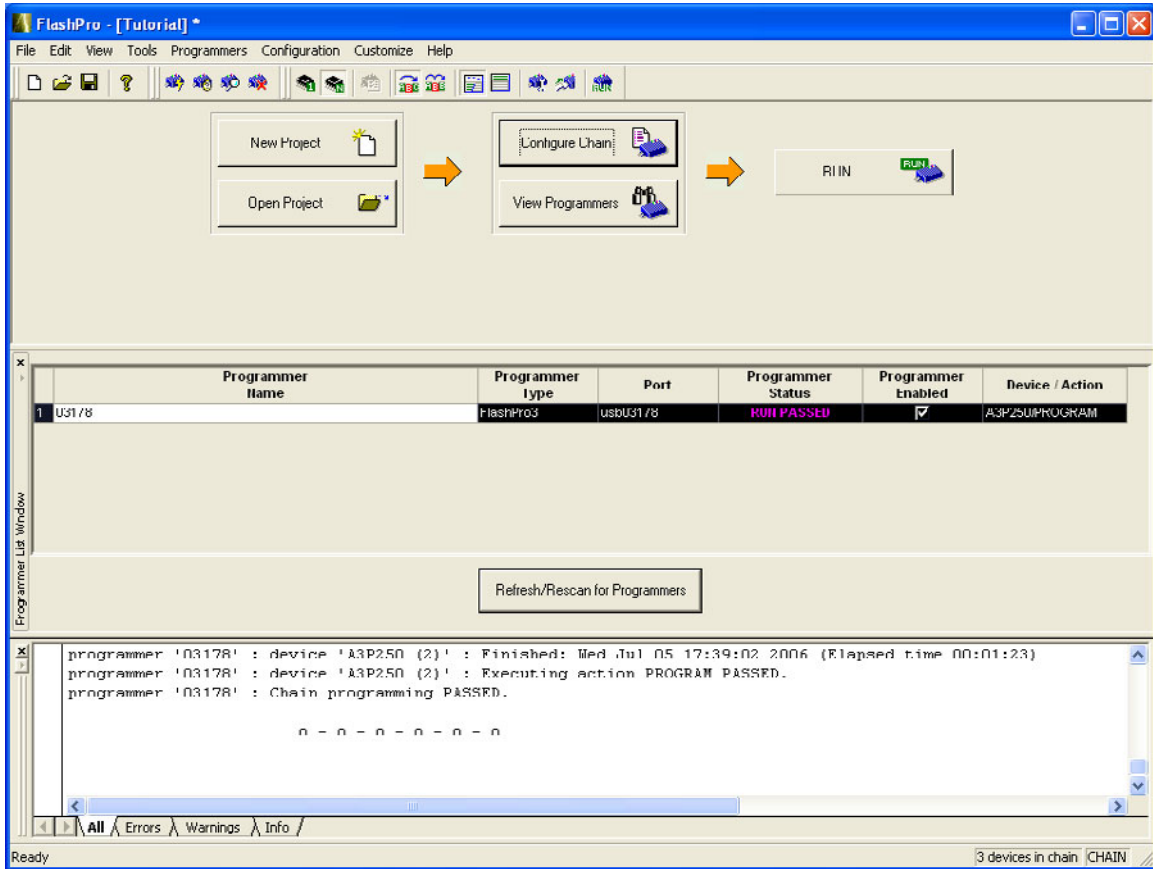


Figure 168 · Programmer List Window Done

Congratulations! You have just completed the FlashPro Multiple Device Serialization Chain Programming tutorial.

Multiple Programmer Multiple Device Chain Programming

This tutorial demonstrates step-by-step instructions on how to parallel program two chains using two programmers, each with two Actel Devices (A3P250 and A3PE600). See the figure below for an illustration.

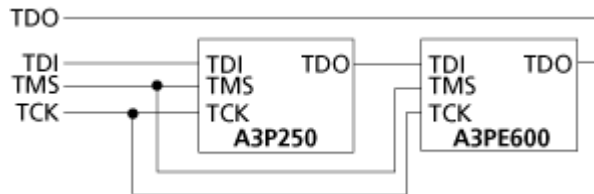


Figure 169 · APA Device Tutorial Example

You should already be familiar with the basic features of the FlashPro software before you begin this tutorial.

Note: This tutorial does not provide software installation instructions. Please have FlashPro already installed before you begin. If you have any questions regarding software installation, see [Software Installation](#).

First you need to create a project.

1. Click the **New Project** button from the FlashPro GUI.
2. From the **New Project** dialog box, type “Tutorial” in the **Project Name** field.



3. Check the Chain box (see figure below).

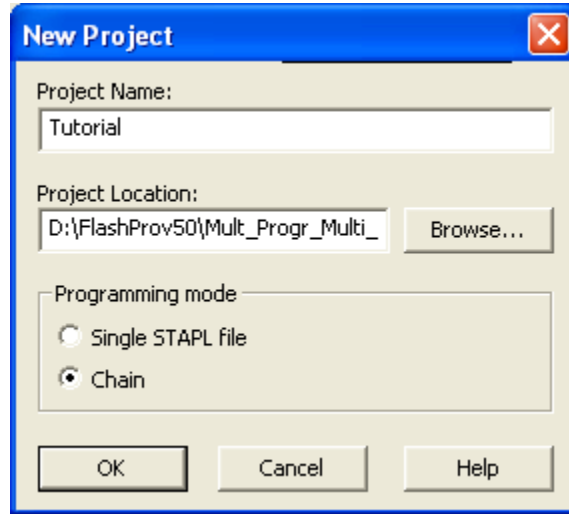


Figure 170 · New Project Dialog Box

4. If necessary, change the default location of your project in the Project Location field.
5. Click OK. The FlashPro GUI displays (see figure below).

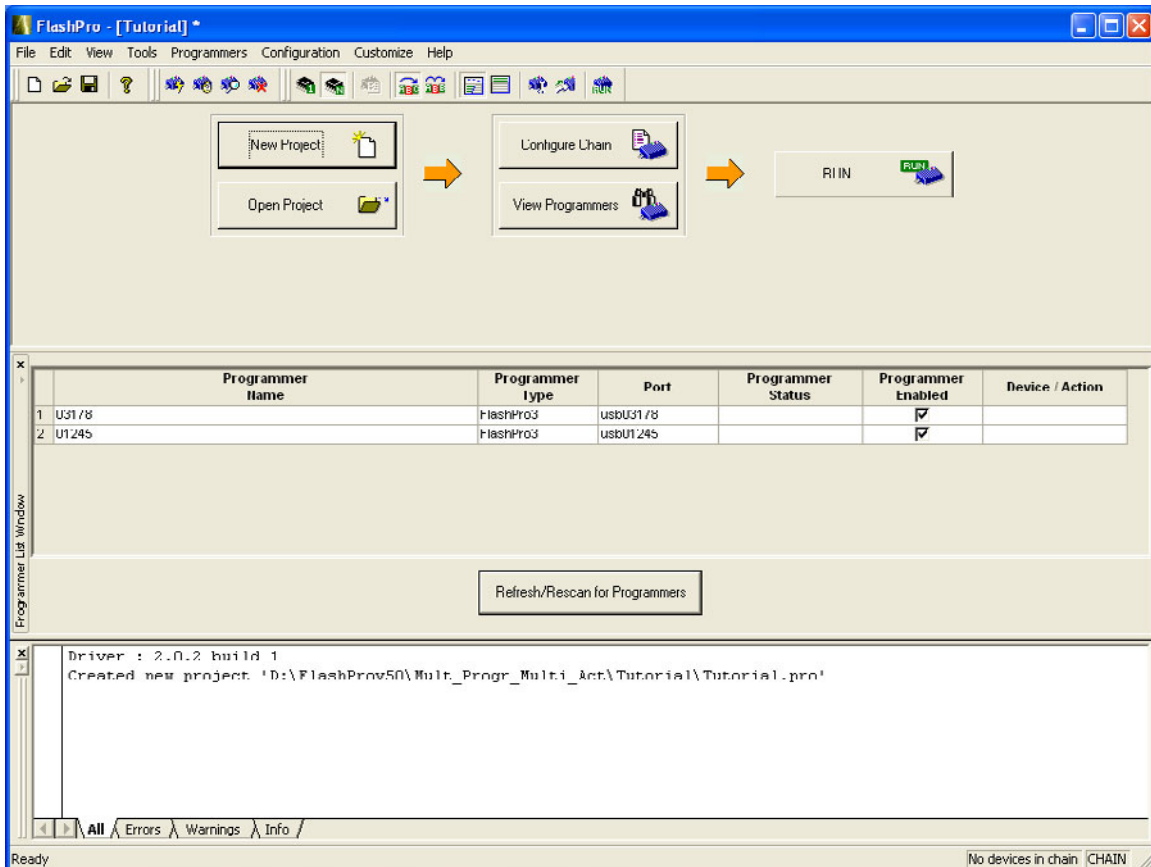


Figure 171 · FlashPro User Interface

- Note:** The Programmer List window updates with your programmer information.
- From the **Programmers** menu, choose **Scan Chain** (or select the programmer in the **Programmer List** window, right-click, then choose **Scan Chain**). The **Select Programmer(s)** dialog box displays (see figure below).

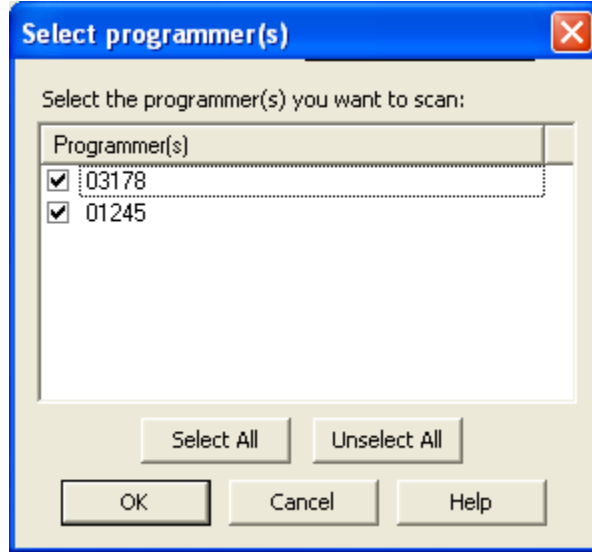


Figure 172 · Select Programmer Window

The **Programmer List** window shows the Scan Chain Test was passed and how the devices are ordered in the chain (see figure below).



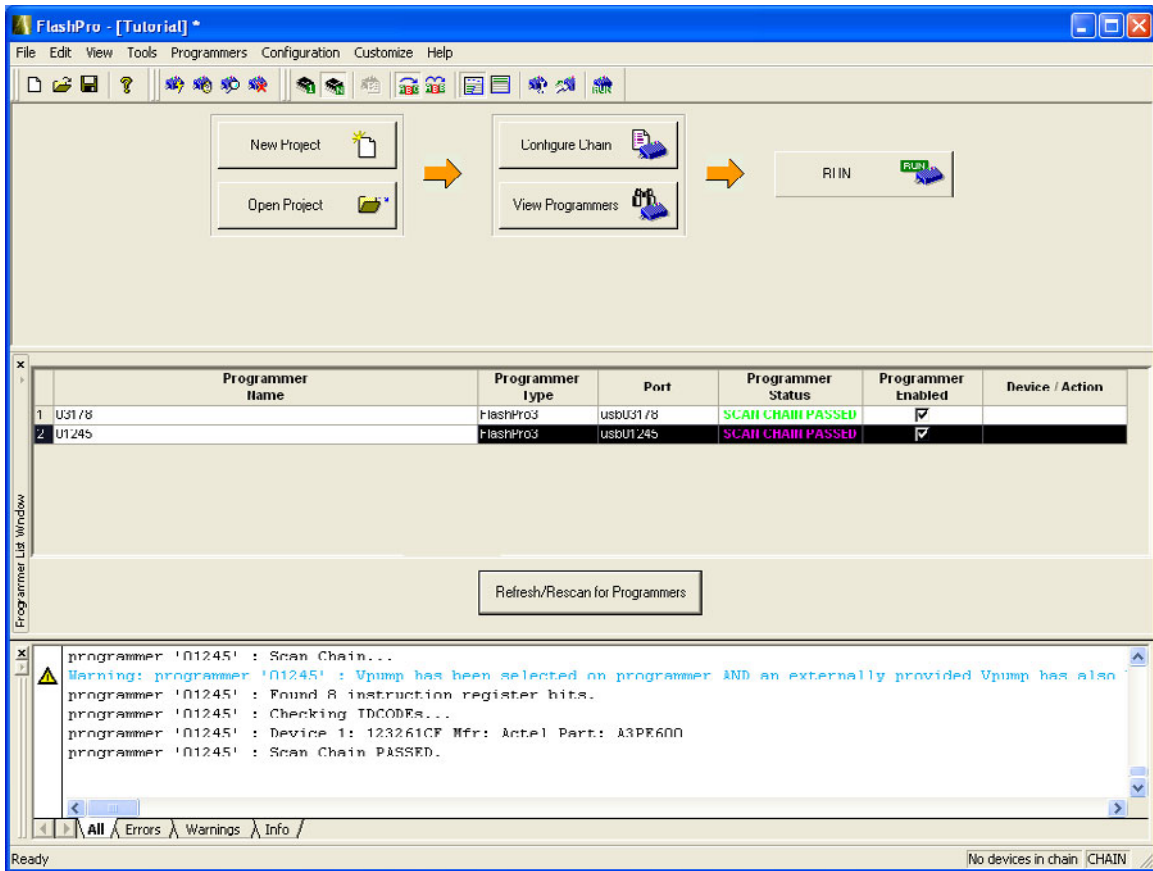


Figure 173 · Scan Chain Order in the Log Window



- Click the **Configure Chain** button

The **Chain Configuration** window displays (see figure below).

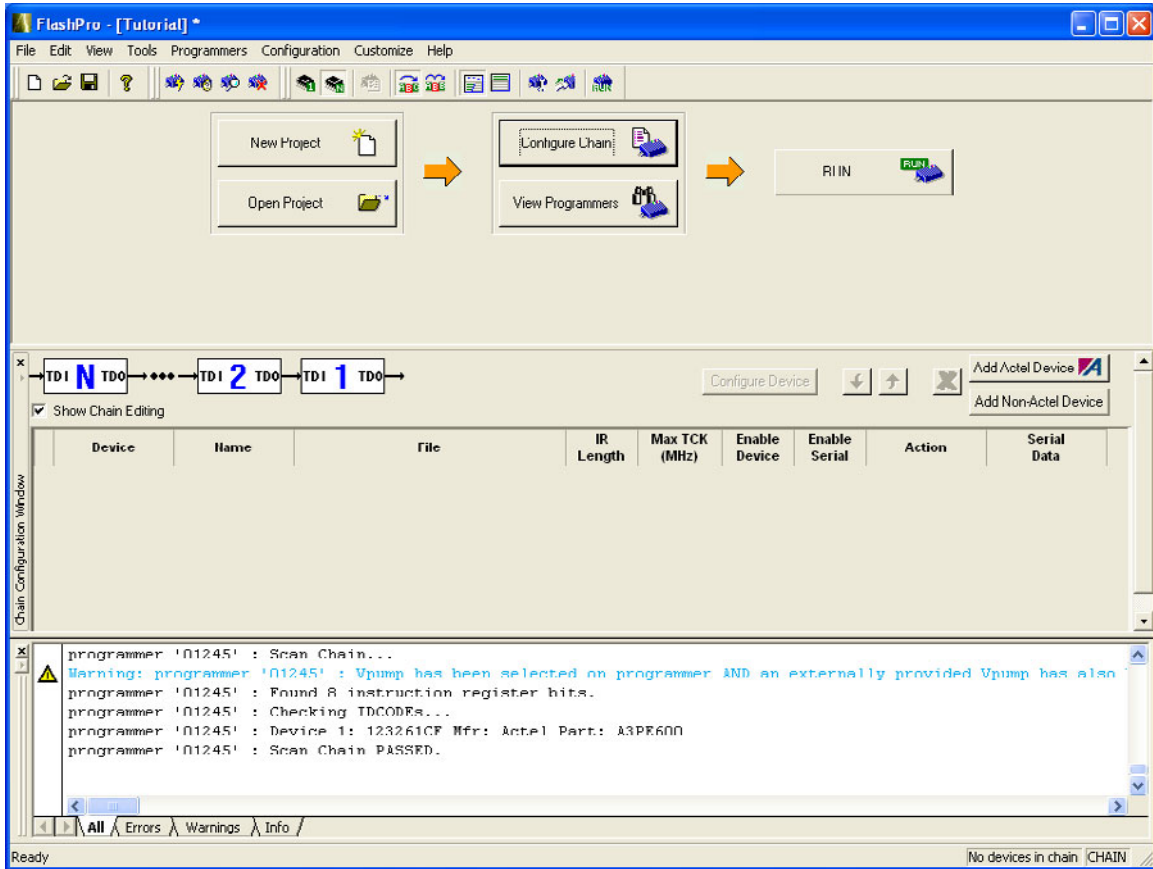



Figure 174 · Chain Configuration Window

8. In the Chain Configuration window, click the Add Device  button to add devices to the chain. The Add Actel Device dialog box displays.
9. Choose "A3PE600" device from the Device drop-down. (See figure below).

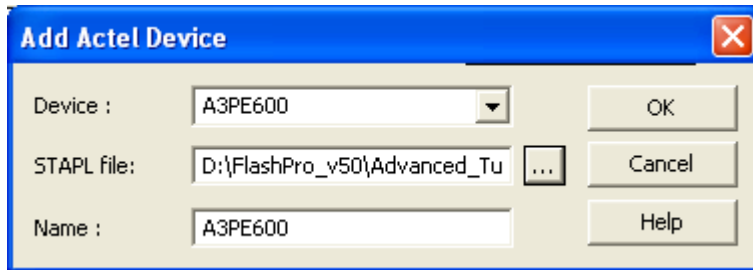


Figure 175 · Add Actel Device Dialog Box

10. In the STAPL file field, use the Browse button to locate the A3PE600.stp file.
11. In the Name field, leave "A3PE600" as default.
12. The A3PE600 device is added into the Chain Configuration window. See the figure below.



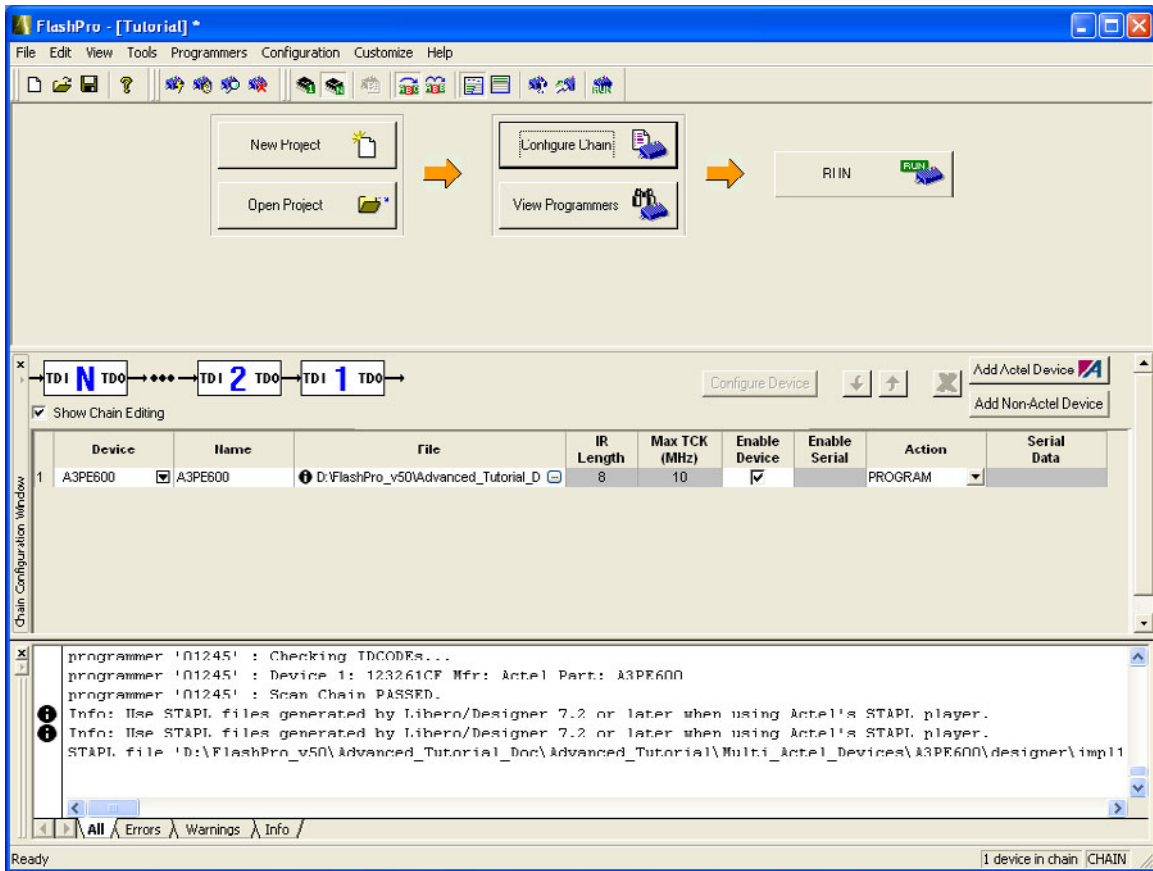


Figure 176 · Device One Chain Configuration Window

- Repeat steps 8 to 11 for A3P250. After you're finished adding all devices in the chain, the Chain Configuration window updates (see figure below).

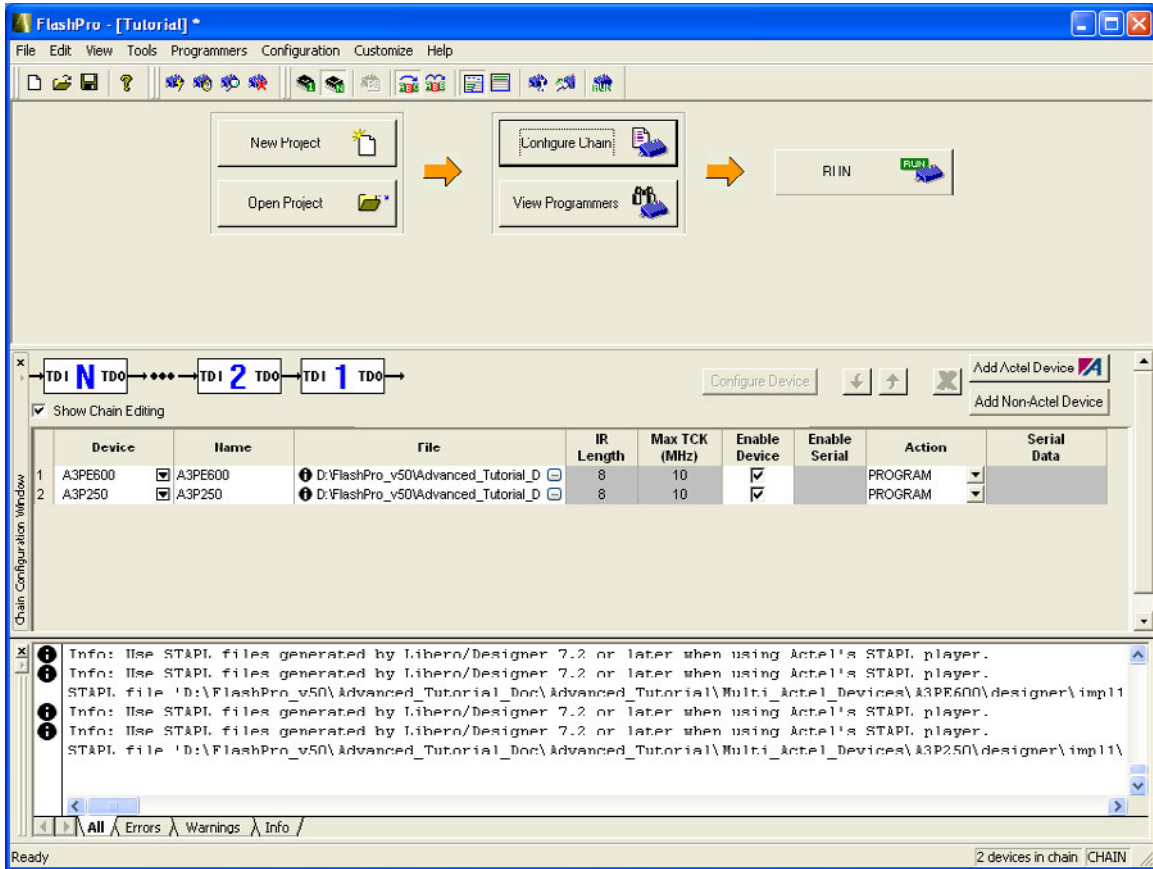


Figure 177 · Chain Configuration Window for all Devices

- Once all the devices have been added to the chain in the correct order and serialization has been selected,



click the **Run** button to program the chain.

- When programming is complete, the **Programmer List Window** displays (see figure below).



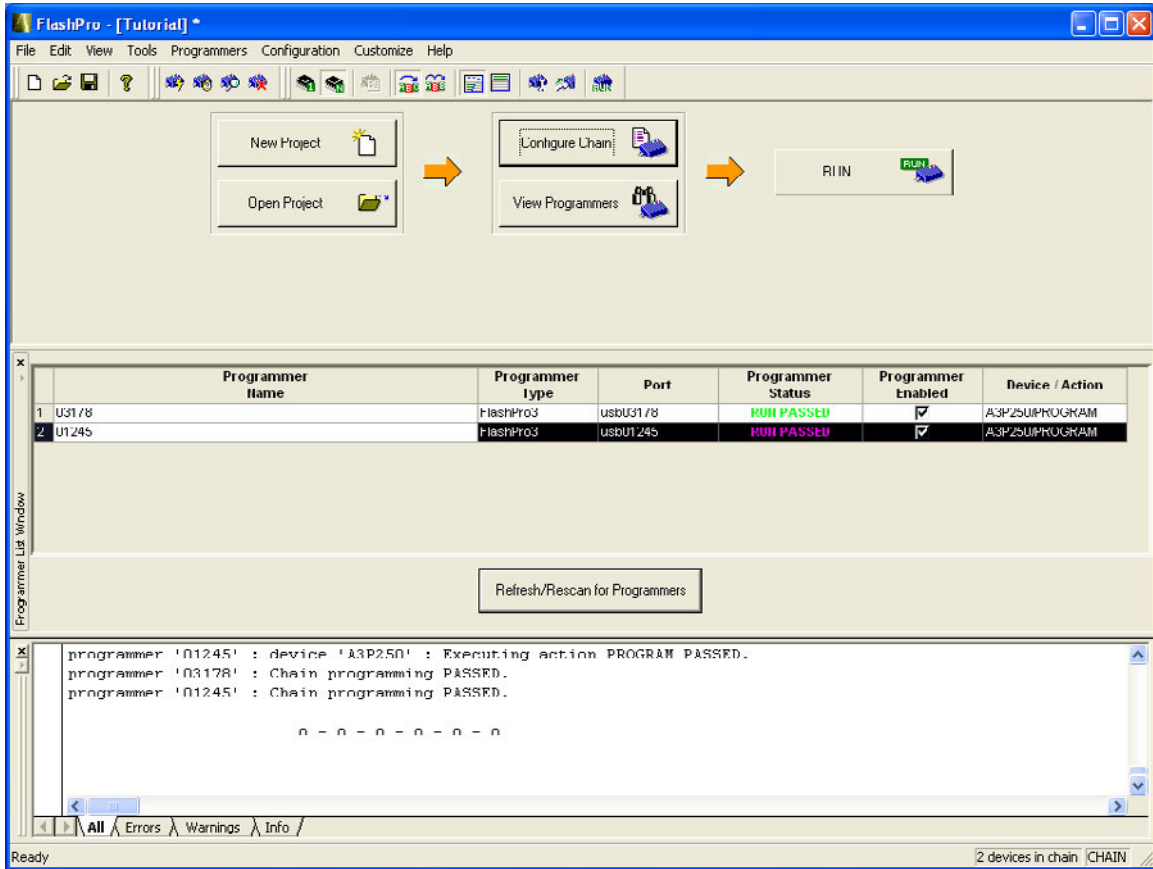


Figure 178 · Programmer List Window Done

Congratulations! You have just completed the FlashPro Multiple Device Serialization Chain Programming tutorial.

Multiple Programmer and Multiple Device Serialization Chain Programming

This tutorial demonstrates step-by-step instructions on how to parallel program two chains using two programmers, each with two Actel Devices (A3P250 with Serialization and A3PE600). See the figure below for an illustration.

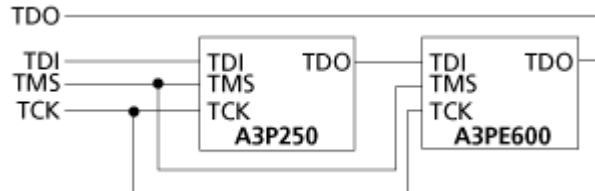


Figure 179 · APA Device Tutorial Example

You should already be familiar with the basic features of the FlashPro software before you begin this tutorial. The STAPL file for the A3P250 device contains 10 serialization data.

Note: This tutorial does not provide software installation instructions. Please have FlashPro already installed before you begin. If you have any questions regarding software installation, see [Software Installation](#).

First you need to create a project.

1. Click the **New Project** button from the FlashPro GUI.
2. From the **New Project** dialog box, type “Tutorial” in the **Project Name** field.
3. Check the **Chain** box (see figure below).

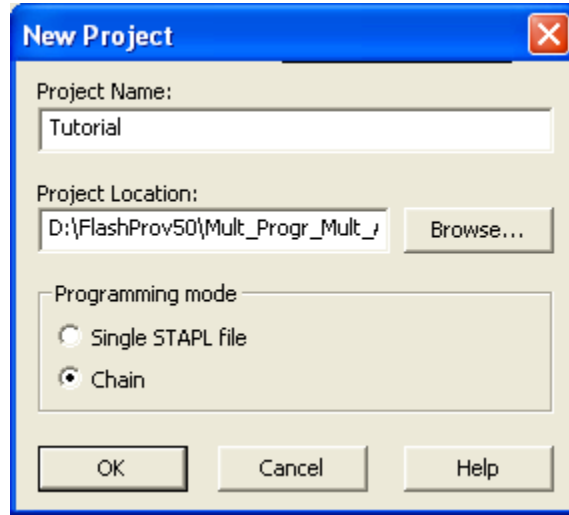


Figure 180 · New Project Dialog Box

4. If necessary, change the default location of your project in the **Project Location** field.
5. Click **OK**. The FlashPro GUI displays (see figure below).



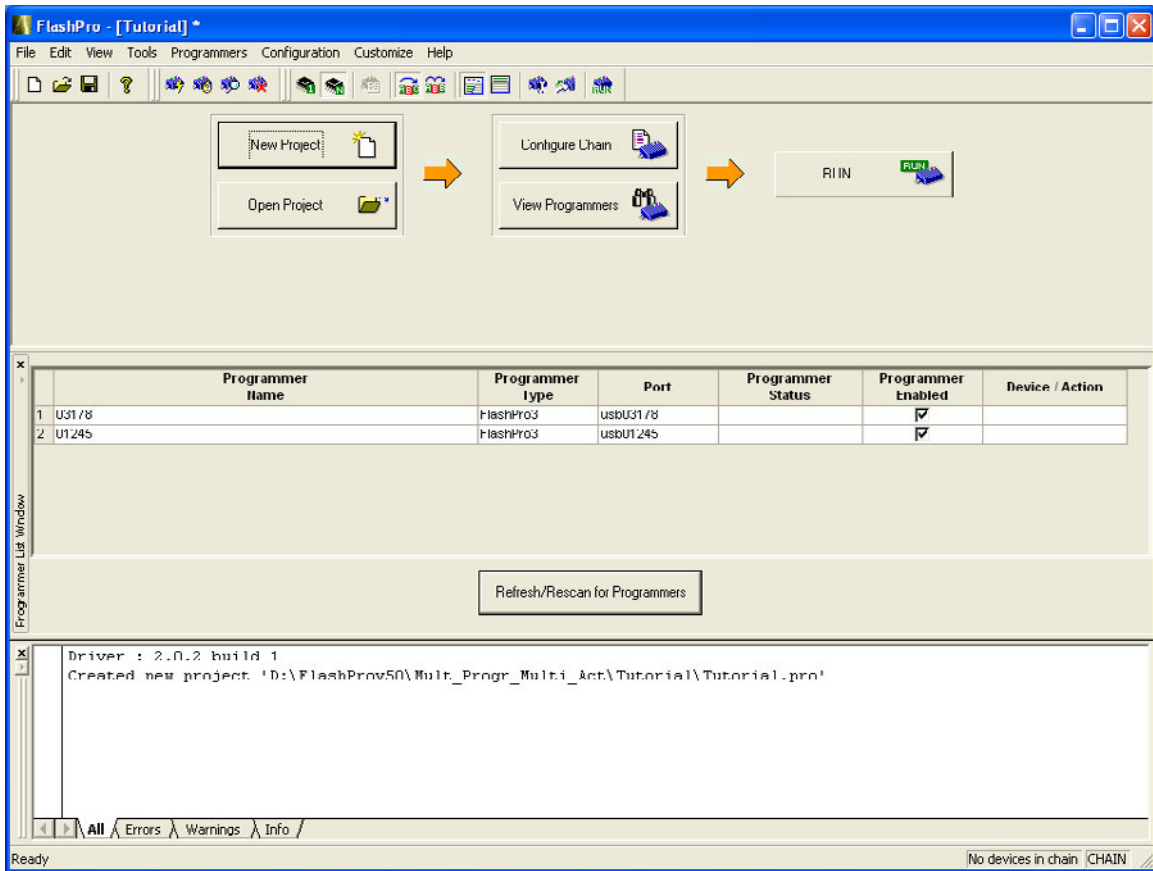


Figure 181 · FlashPro User Interface

Note: The Programmer List window updates with your programmer information.

- From the **Programmers** menu, choose **Scan Chain** (or select the programmer in the **Programmer List** window, right-click, then choose **Scan Chain**). The **Select Programmer(s)** dialog box displays (see figure below).

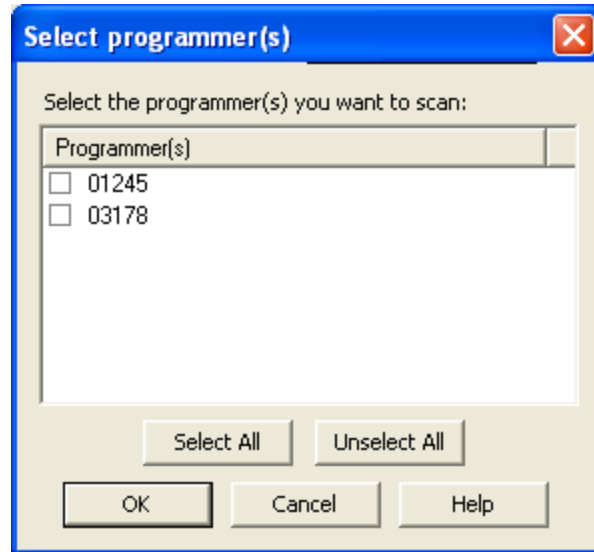


Figure 182 · Select Programmer Window

The **Programmer List** window shows the Scan Chain Test was passed and how the devices are ordered in the chain (see figure below). In this example, A3PE600 will be programmed first in the chain since it is connected directly to TDO.

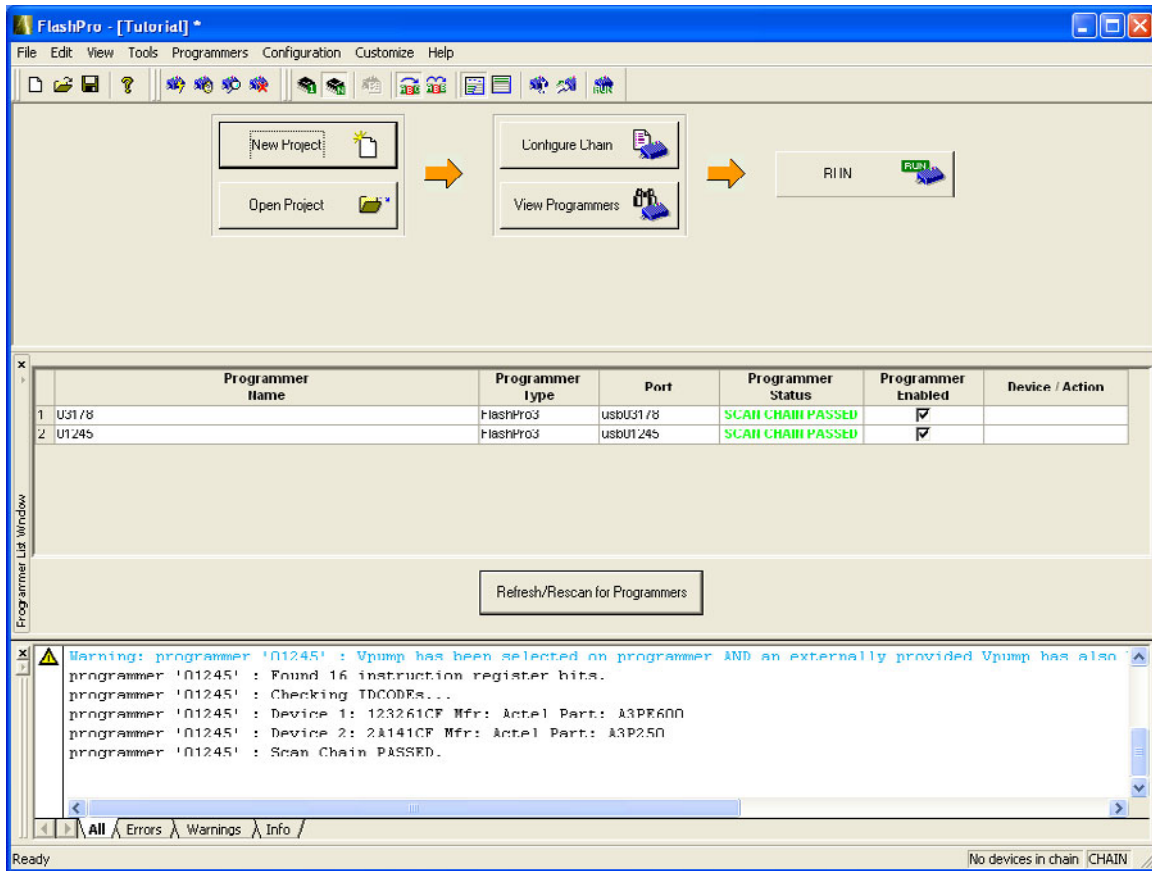


Figure 183 · Scan Chain Order in the Log Window



- Click the **Configure Chain** button

The **Chain Configuration** window displays (see figure below).

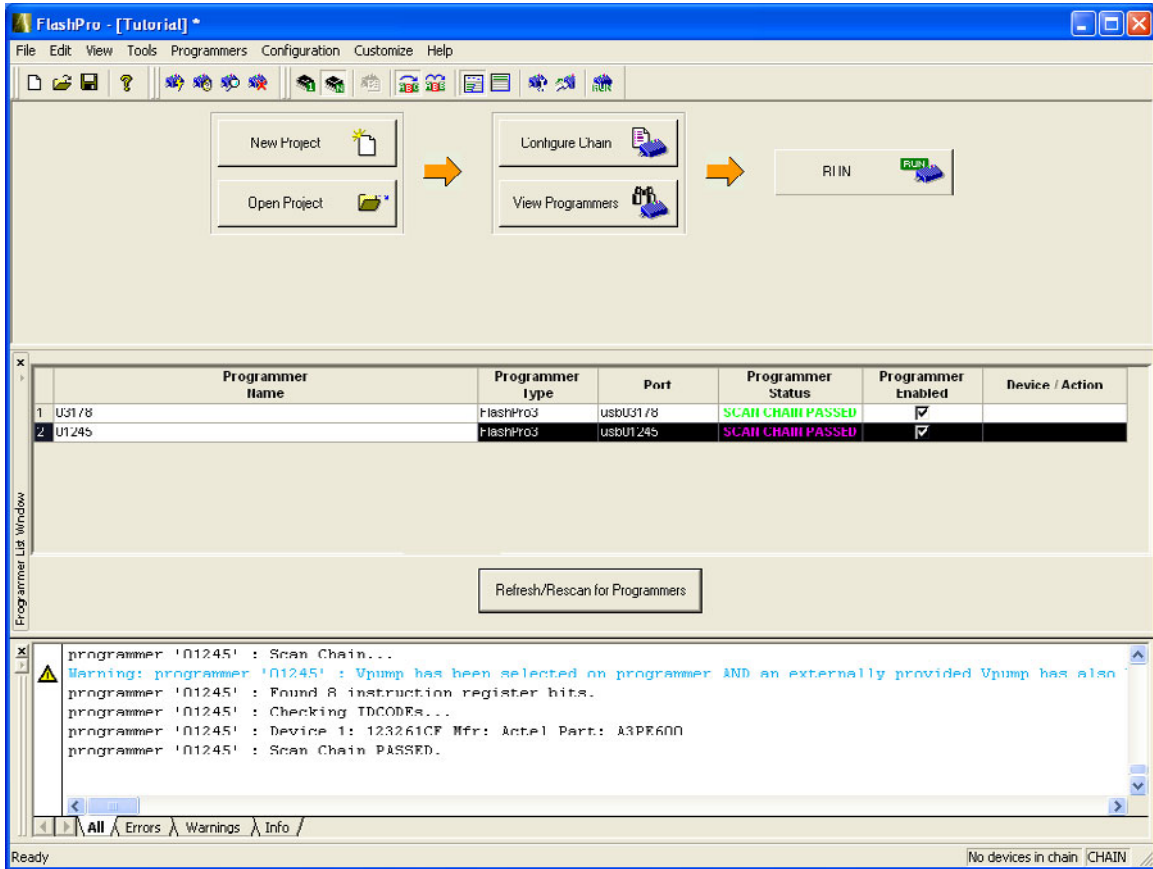



Figure 184 · Chain Configuration Window

8. In the Chain Configuration window, click the Add Device  button to add devices to the chain.
The Add Actel Device dialog box displays.
9. Choose "A3PE600" device from the Device drop-down. (See figure below).

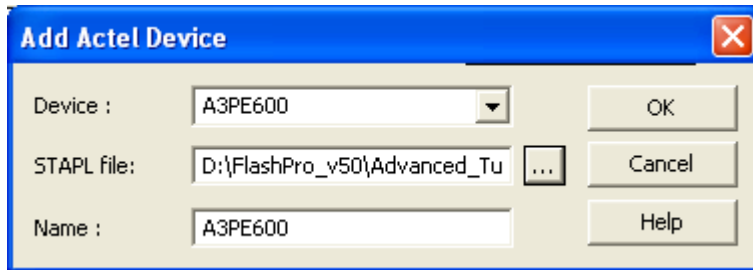


Figure 185 · Add Actel Device Dialog Box

10. In the STAPL file field, use the Browse button to locate the A3PE600.stp file.
11. In the Name field, leave "A3PE600" as default.
12. The A3PE600device is added into the Chain Configuration window.



- Repeat steps 8 to 11 for A3P250. After you're finished adding all devices in the chain, the Chain Configuration window updates (see figure below).

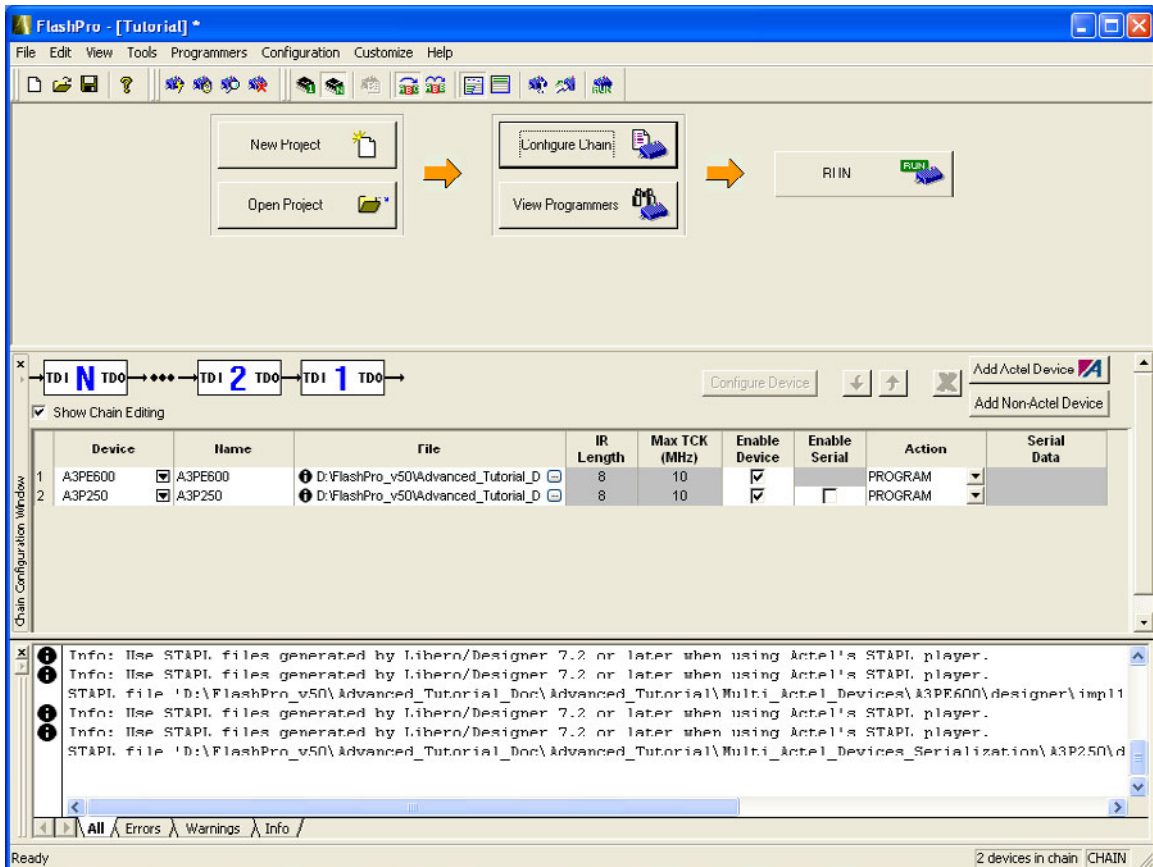


Figure 186 · Chain Configuration Window for all Devices

- From the Chain Configuration Window, check the Enable Serial box and the Chain Configuration window displays (see figure below).

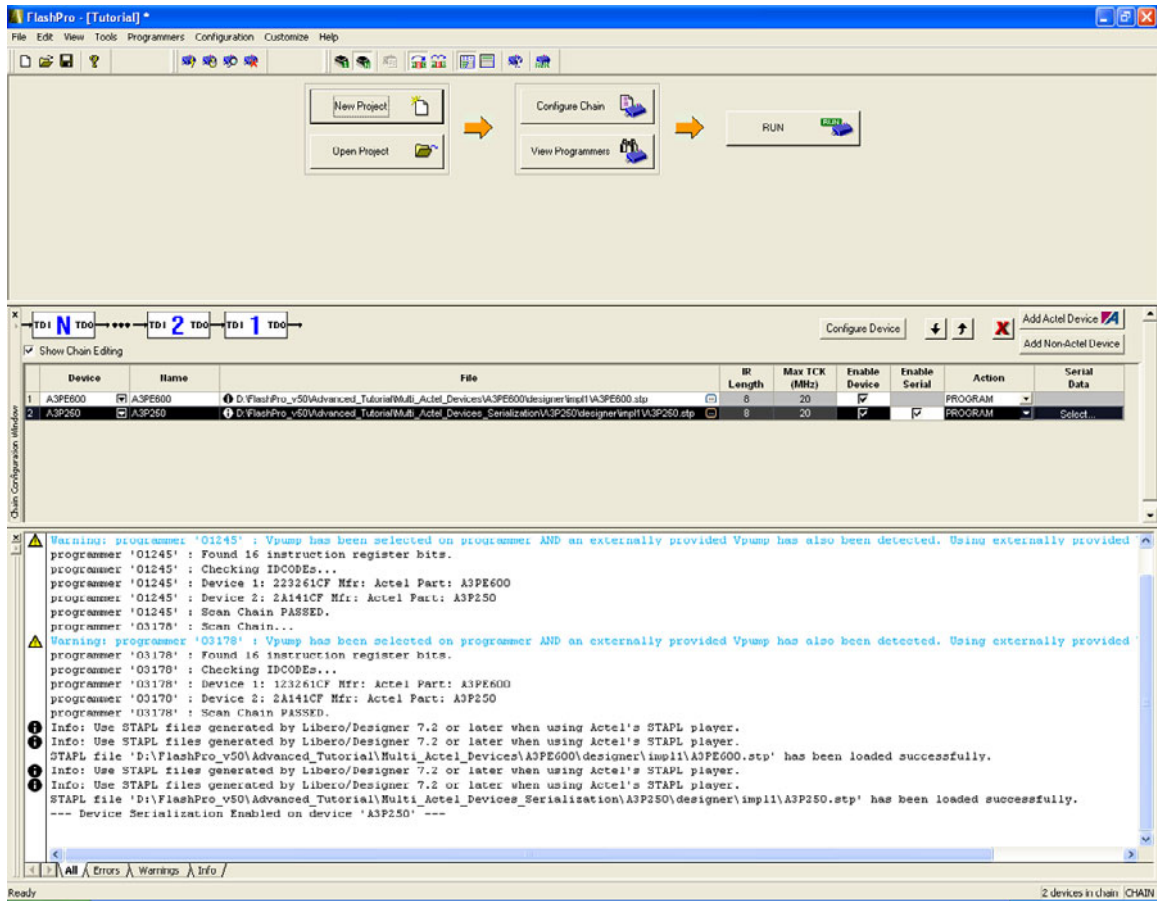


Figure 187 · Chain Configuration Window with Enable Serial Selected

15. Click Select in the Serial Data column. The Serial Settings dialog box displays. See figure below.



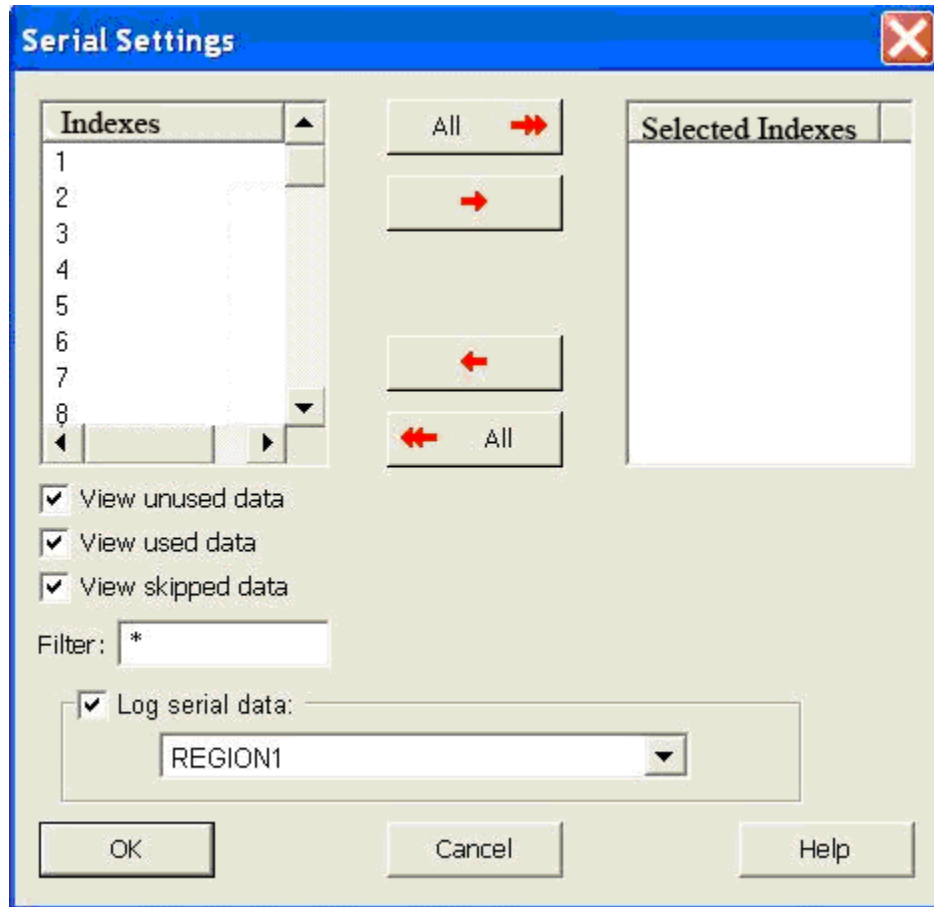


Figure 188 · Serial Settings Dialog Box

16. From the **Serial Settings** dialog box, click the **All** button to select all the serial data.
17. Click **OK**.
18. Once all the devices have been added to the chain in the correct order and serialization has been selected,



click the **Run** button to program the chain.

19. When programming is complete, the **Programmer List** window displays (see figure below).

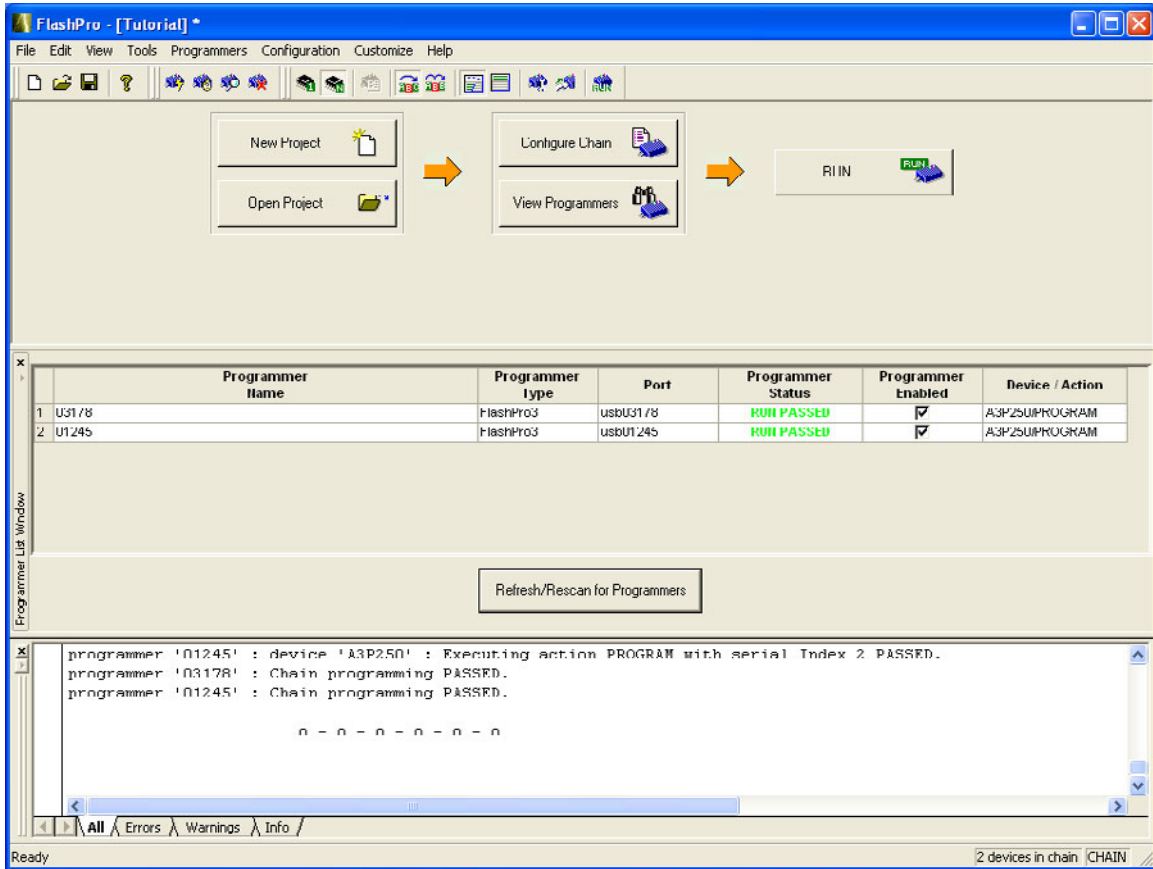


Figure 189 · Programmer List Window Done

Congratulations! You have just completed the FlashPro Multiple Programmer and Multiple Device Serialization Chain Programming tutorial.

Parallel Port Cable Information

The FlashPro software supports the generic Parallel Port Cable.

To connect to the Parallel Port Cable:

1. From the **Parallel Port Cable** text box, select the Parallel Port Buffer Cable. See figure below.
2. Select the parallel port that is connected to the cable from the **Parallel Port** text box.



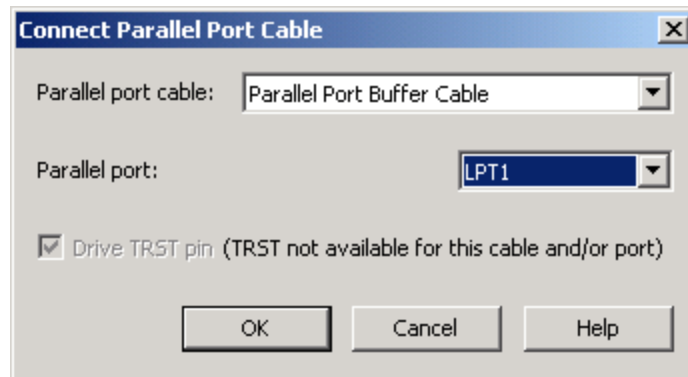


Figure 190 · Connect Parallel Port Cable

3. Click OK.
The "Para2Buff" programmer is added to the programmer list.

Importing and Exporting Files

Importing Configuration Files

Below are the steps for importing a configuration file.

To import a configuration file:

1. From the **File** menu, select **Import Configuration File**. The **Import Configuration File** dialog box displays.
2. Find your file and click **Open**.

Exporting Configuration Files

Below are the steps for exporting a configuration file.

To export a configuration file:

1. From the **File** menu, select **Export** and then choose **Export Configuration File**. The **Export Configuration File** dialog box displays.
2. Find your file and click **Save**.

Exporting a Chain STAPL File

Below are the steps for exporting a chain STAPL file.

To export a chain STAPL file:

1. From the **File** menu, select **Export** and then choose **Export Chain STAPL File**. The **Export Chain STAPL File** dialog box displays.
2. Name your file and click **Save**.

Note: Chain STAPL file export is supported if all selected IGLOO, Fusion, and ProASIC3 family devices have STAPL or PDB files loaded.

Exporting Single Device STAPL Files

Below are the steps for exporting a single STAPL file.

To export a single STAPL file in single mode:

This option is available only for single programming mode projects with a PDB file loaded (refer to [Single STAPL file basic tutorial](#) for more information).

1. From the **File** menu, choose **Export > Export Single Device STAPL File**. The **Export Single Device STAPL File** dialog box appears.
2. Name your file and click **Save**.

To export a single device STAPL file in chain mode:

This option is available only for chain programming mode projects with a PDB file loaded (refer to [Chain programming tutorial](#) for more information). Exporting a single device STAPL file is only supported for one device in the chain.

1. Select only one device from the chain, and from the **File** menu, select **Export** and then choose **Export Single Device STAPL File**. The **Export Single Device STAPL File** dialog box appears.

2. Name your file and click **Save**.

Or

1. Right-click a device in the Chain Configuration Window, and then choose **Export Single Device STAPL File**. The **Export Single Device STAPL File** dialog box appears.
2. Name your file and click **Save**.

Exporting Single Device SVF Files

The following steps describe how to export SVF files.

To export single device SVF files in single mode:

This option is available only for single programming mode projects with a PDB file loaded (refer to [Single STAPL file basic tutorial](#) for more information).

1. From the **File** menu, select **Export** and then choose **Export Single Device SVF File**. The **Export Single Device SVF File** dialog box appears.
2. Name your file and click **Save**.

Note: Multiple SVF files will be generated from a single PDB. Each file corresponds to a PDB action, and will be saved in the `<SVF_filename>` folder as `<SVF_filename>_<action name>.SVF`.

To export single device SVF files in chain mode:

This option is available only for chain programming mode projects with a PDB file loaded (refer to [Chain programming tutorial](#) for more information).

1. Select only one device from the chain, and from the **File** menu, select **Export** and then choose **Export Single Device SVF File**. The **Export Single Device SVF File** dialog box appears.
2. Name your file and click **Save**.

Or

1. Right-click a device in the Chain Configuration Window, and then choose **Export Single Device SVF File**. The **Export Single Device SVF File** dialog box appears.
2. Name your file and click **Save**.

Note: Multiple SVF files will be generated from a single PDB. Each file corresponds to a PDB action, and will be saved in the `<SVF_filename>` folder as `<SVF_filename>_<action name>.SVF`.

Exporting Single Device 1532 Files

The following steps describe how to export 1532 files.

To export single device 1532 files in single mode:

This option is available only for single programming mode projects with a PDB file loaded (refer to [Single STAPL file basic tutorial](#) for more information).

1. From the **File** menu, choose **Export Single Device 1532 File**. The **Export Single Device 1532 File** dialog box appears.
2. Name your file and click **Save**.

Note: Two files will be generated from a single PDB and will be saved in the `<1532_filename>_1532` folder as

Note: IEEE 1532 BSDL file - `<1532_filename>.bsd`

Note: IEEE 1532 Data file - `<1532_filename>.isc`



To export single device 1532 files in chain mode:

This option is available only for chain programming mode projects with a PDB file loaded (refer to [Chain programming tutorial](#) for more information). Exporting a single device STAPL file is only supported for one device in the chain.

1. Select only one device from the chain, and from the **File** menu, select **Export** and then choose **Export Single Device 1532 File**. The **Export Single Device 1532 File** dialog box appears.
2. Name your file and click **Save**.

Or

1. Right-click a device in the Chain Configuration Window, and then choose **Export Single Device 1532 File**. The **Export Single Device 1532 File** dialog box appears.

Note: Name your file and click **Save**. Two files will be generated from a single PDB and will be saved in the <1532_filename>_1532 folder as

Note: IEEE 1532 BSD - <1532_filename>.bsd

Note: IEEE 153 file - <1532_filename>.isc

Using Hot Keys

General Hot Keys

You can use hot keys for a lot of the features of the FlashPro software. See the table below for a list of general hot keys.

Table 17 · FlashPro Software General Hot Keys

Feature	Hot Key
New Project	Ctrl+N
Open Project	Ctrl+O
Save Project	Ctrl+S
Import Configuration File	Ctrl+I
Refresh Views	F5
Refresh/Rescan for Programmers	Ctrl+F5

See Also

[Single STAPL programming hot keys](#)

[Chain programming hot keys](#)

Single Device Programming Hot Keys

See the table below for the hot keys for single device programming.

Table 18 · Single Device Programming Hot Keys

Feature	Hot Key
Load a STAPL file	Ctrl + Shift + L
Select Action and Procedures	Ctrl + Shift + A
Enable Serialization	Ctrl + Shift + S
Select Serialization Data	Ctrl + Shift + R
View Serialization Status	Ctrl + Shift + U
View Chain Parameter (Pre/Post IR/DR)	Ctrl + Shift + H
Configure Target Device	Ctrl + Shift + D
Run	Ctrl + Return

Chain Programming Hot Keys

See the table below for the hot keys for chain programming.

Table 19 · Chain Programming Hot Keys

Feature	Hot Key
Add Actel Device	Ctrl + Shift + T
Add non Actel Device	Ctrl + Shift + N
Remove Device	Ctrl + R
Configure Device	Ctrl + F
Load STAPL File	Ctrl + Shift + L
Load BSDL File	Ctrl + Shift + B
Enable Device	Ctrl + E
Select Action and Procedures	Ctrl + Shift + A
Enable Serialization	Ctrl + Shift + S
Select Serialization Data	Ctrl + Shift + R
View Serialization Data	Ctrl + Shift + u
Copy Device	Ctrl + Shift + C
Cut Device	Ctrl + Shift + X
Paste Device	Ctrl + Shift + V
Move Device Down	Ctrl + D
Move Device Up	Ctrl + U
Run	Ctrl + Return

Batch Mode

Batch mode programming can be achieved by executing FlashPro [TCL scripts from the command line](#).

The example below executes The FlashPro TCL script *batch.tcl* from the command line:

```
<location of Actel software>/bin/flashpro.exe script:batch.tcl
```

Batch.tcl contains the following script:

```
new_project -name {newproject} -location {./newproject} -mode {single}
  set_programming_file -file {./design.stp}
set_programming_action -action {PROGRAM}
```



`run_selected_actions`
`close_project`

About TCL Commands

TCL, the Tool Command Language, pronounced *tickle*, is a scripting language that is compatible with FlashPro. You can run scripts from the command line or store and run a series of commands in a “.tcl” batch file. The TCL commands supported by FlashPro are listed below:

[add_actel_device](#)
[add_non_actel_device](#)
[close_project](#)
[configure_flashpro3_prg](#)
[configure_flashproLite_prg](#)
[configure_flashpro_prg](#)
[connect_cable](#)
[copy_device](#)
[cut_device](#)
[dump_tcl_support](#)
[enable_device](#)
[enable_prg](#)
[enable_prg_type](#)
[enable_procedure](#)
[enable_serialization](#)
[export_config](#)
[export_script](#)
[export_single_1532](#)
[export_stapl](#)
[import_config](#)
[new_project](#)
[open_project](#)
[paste_device](#)
[ping_prg](#)
[refresh_prg_list](#)
[remove_device](#)
[remove_prg](#)
[run_selected_actions](#)
[save_log](#)
[save_project](#)
[save_project_as](#)
[scan_chain_prg](#)
[select_serial_range](#)
[select_target_device](#)
[self_test_prg](#)
[set_bsd1_file](#)
[set_chain_param](#)
[set_device_ir](#)
[set_device_name](#)
[set_device_order](#)

[set_device_tck](#)
[set_device_type](#)
[set_main_log_file](#)
[set_prg_name](#)
[set_programming_action](#)
[set_programming_file](#)
[set_programming_mode](#)
[set_serialization_log_file](#)
[set_serialization_mode](#)
[signature](#)
[update_progr_file.htm](#)

Running Tcl scripts from within FlashPro

Instead of running scripts from the command line, you can use FlashPro's Run Script dialog box to run a script.

To execute a Tcl script file within FlashPro:

1. From the File menu, choose Run Script to display the Execute Script dialog box.

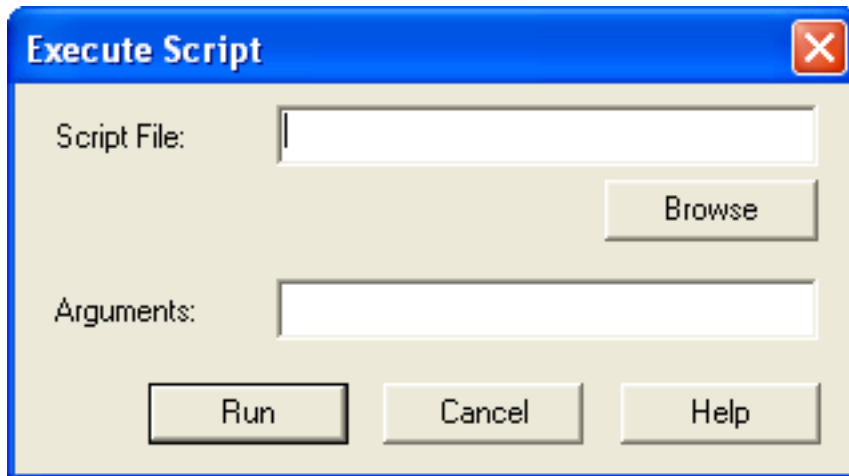


Figure 191 · Execute Script Dialog Box

2. Click **Browse** to display the **Open** dialog box, in which you can navigate to the folder containing the script file to open. When you click **Open**, FlashPro enters the full path and script filename into the Execute Script dialog box for you.
3. In the Arguments box, enter the arguments to pass to your Tcl script. Separate each argument by a space character. For information about accessing arguments passed to a Tcl script, see [Running Scripts from the command line](#).
4. Click **Run**.



Running Tcl Scripts from the Command Line

You can run Tcl scripts from your Windows command line.

To execute a Tcl script file in the FlashPro software from a shell command line:

1. At the prompt, type the path to the Actel software followed by the word "SCRIPT" and a colon, and then the name of the script file as follows:

```
<location of Actel software>/bin/flashpro.exe SCRIPT:<filename>
```

The example below executes in batch mode the script *foo.tcl*:

```
<location of Actel software>/bin/flashpro.exe script:foo.tcl
```

The example below executes in batch mode the script *foo.tcl* and exports the log in the file *foo.txt*:

```
<location of Actel software>/bin/flashpro.exe script:foo.tcl logfile:foo.txt
```

The example below executes in batch mode the script *foo.tcl*, creates a console where the log is displayed and exports the log in the file *foo.txt*:

```
<location of Actel software>/bin/flashpro.exe script:foo.tcl console_mode:show  
logfile:foo.txt
```

Arguments passed to a Tcl script can be accessed through the Tcl variables *argc* and *argv*. The example below demonstrates how a Tcl script accesses these arguments:

```
puts "Script name: $argv0"  
puts "Number of arguments: $argc"  
set i 0  
foreach arg $argv {  
    puts "Arg $i : $arg"  
    incr i  
}
```

Note: Script names can contain spaces if the script name is protected with double quotes:

```
flashpro.exe script:"flashpro tcl/foo 1.tcl"
```

Exporting Tcl Scripts from within FlashPro

To export a set of Tcl commands from the FlashPro history:

1. From the File menu, choose **Export > Export Script**.
2. Enter the filename and click **Save**. The Script Export Options dialog is appears (see image below).

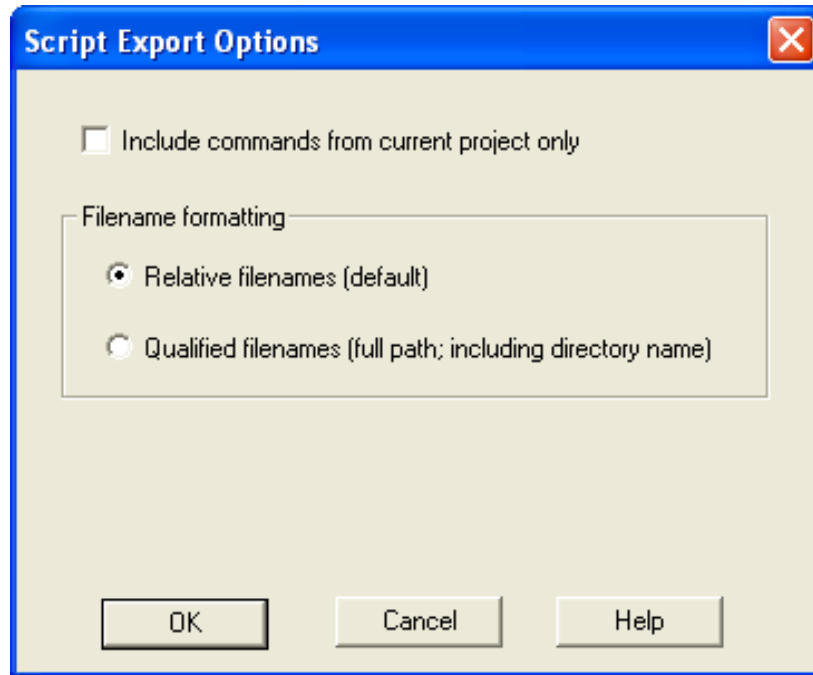


Figure 192 · Script Export Options Dialog Box

Check the **Include commands from current project only** to export commands of the current project only. You can specify the filename formatting by selecting **Relative filenames** (relative to the current directory) or **Qualified filenames** (absolute path, including the directory name).

3. Click OK.

add_actel_device

Adds an Actel device to the chain. Either the *file* or *device* parameter must be specified. Chain programming mode must have been set.

```
add_actel_device [-file {filename}] [-device {device}] -name {name}
```

Arguments

Where:

- file{*filename*}
Specifies a programming filename.
- device{*device*}
Specifies the Actel device family.
- name{*name*}
Specifies the device user name.

Supported Families



Exceptions

Example

```
add_actel_device -file {e:/design/stp/TOP.stp} -name {MyDevice1}
add_actel_device -device {A3P250} -name {MyDevice2}
```

add_non_actel_device

Adds a non-Actel device in the chain. Either the file, or (-tck And -ir) parameters must be specified. The Chain programming mode must have been set.

```
add_non_actel_device [-file {file}] [-ir {ir}] [-tck {tck}] -name {name}
```

Arguments

-file {filename}
Specifies a BSDL file.

-ir {ir}
Specifies the IR length.

-tck {tck}
Specifies the maximum TCK frequency (in MHz).

-name {name}
Specifies the device user name.

Supported Families

Exceptions

Examples

```
add_non_actel_device -file {e:/design/bsdl/DeviceX.bsd1 } -name {MyDevice3}
add_non_actel_device -ir 8 - tck 5 -name {MyDevice4}
```

close_project

Closes the FlashPro project.

```
close_project
```

Arguments

None

Supported Families

Exceptions

Example

```
close_project
```

configure_flashpro_prg

Changes FlashPro programmer settings.

```
configure_flashpro_prg [-vpp {ON|OFF}] [-vpn {ON|OFF}] [-vddl {ON|OFF}] [-force_vddp {ON|OFF}] [-vddp {2.5|3.3}] [-drive_trst {ON|OFF}] [-force_freq {ON|OFF}] [-freq {freq}]
```

Arguments

```
-vpp {ON|OFF}
-vpn {ON|OFF}
-vddl {ON|OFF}
-force_vddp {ON|OFF}
-vddp {2.5|3.3}
-drive_trst {ON|OFF}
-force_freq {ON|OFF}
-freq {freq}
    Specifies the frequency in MHz.
```

Supported Families

Exceptions

Example

configure_flashpro3_prg

Changes FlashPro3 programmer settings.

```
configure_flashpro3_prg [-vpump {ON|OFF}] [-force_freq {ON|OFF}] [-freq {freq}]
```

Arguments

```
-vpump {ON|OFF}
-force_freq {ON|OFF}
```



```
-freq {freq}
```

Supported Families

Exceptions

Example

configure_flashproLite_prg

Changes FlashProLite programmer settings.

```
configure_flashproLite_prg [-vpp {ON|OFF}] [-vpn {ON|OFF}] [-drive_trst {ON|OFF}] [-force_freq {ON|OFF}] [-freq {freq}]
```

Arguments

```
-vpp {ON|OFF}  
-vpn {ON|OFF}  
-drive_trst {ON|OFF}  
-force_freq {ON|OFF}  
-freq {freq}
```

Supported Families

Exceptions

Example

connect_cable

Connects a cable to a port.

```
connect_cable -cable_name {cable_name} -port_name {port_name} [-drive_trst {ON|OFF}]
```

Arguments

```
-cable_name {cable_name}  
-port_name {port_name}
```

Supported Families

Exceptions

Example

```
connect_cable -cable_name {Parallel_Port_Buffer_Cable} -port_name {Lpt1} -drive_trst {ON}
```

copy_device

Copies a device in the chain to the clipboard. Chain programming mode must be set. See the [paste_device command](#) for more information.

```
copy_device (-name {name})*
```

Arguments

-name {name}

Specifies the device name. Repeat this argument to copy multiple devices.

Supported Families

Exceptions

Example

```
copy_device -name {MyDevice1} -name {MyDevice2}
```

cut_device

Removes one or more devices from the chain. It places the removed device in the clipboard. Chain programming mode must be set to use this command. See the [paste_device](#) command for more information.

```
cut_device (-name {name})*
```

Arguments

-name {name}

Specifies the device name. You can repeat this argument for multiple devices.

Supported Families

Exceptions



Example

```
cut_device -name {MyDevice1} -name {MyDevice2}
```

dump_tcl_support

Unloads the list of supported FlashPro Tcl commands.

```
dump_tcl_support -file {file}
```

Arguments

-file {file}

Supported Families

Exceptions

Example

enable_device

Enables or disables a device in the chain (if the device is disabled, it is bypassed). Chain programming mode must be set. The device must be an Actel device.

```
enable_device -name {name} -enable {TRUE|FALSE}
```

Arguments

-name {name}

Specifies the user device name

-enable {TRUE|FALSE}

Specifies whether the device is to be enabled or disabled. If you specify multiple devices, this argument applies to all specified devices. (TRUE = enable. FALSE = disable)

Supported Families

Exceptions

Example

```
enable_device -name {MyDevice1} -enable {FALSE}
```

enable_prg

Enables or disables one or more programmers.

```
enable_prg (-name {name})* -enable {TRUE|FALSE}
```

Arguments

-name {*name*}*

Specifies the programmer name. You can repeat this argument for multiple programmers.

-enable {TRUE|FALSE}

Specifies whether the programmer is to be enabled or disabled. If you specify multiple programmers, this argument applies (TRUE = enable. FALSE = disable).

Supported Families

Exceptions

Example

```
enable_prg -name {FP300085} -name {FP300086} -enable {TRUE}
```

enable_prg_type

Enables or disables all programmers of a specified programmer type.

```
enable_prg_type -prg_type {prg_type} -enable { TRUE | FALSE }
```

Arguments

-prgType { FP | FPLite | FP3 | PP }

Specifies the programmer type to be enabled/disabled (FP–FlashPro type programmers, FPLite–FlashPro Lite type programmers, FP3–FlashPro3 type programmers, PP–Parallel port cable type programmers).

-enable {TRUE|FALSE}

Specifies whether the programmers are to be enabled or disabled (TRUE–enable, FALSE–disable).

Exceptions

Example

```
enable_prg_type -prg_type{FP3} -enable{TRUE}
```



enable_procedure

To enable/disable an optional procedure of an action. The device name parameter must be specified only in chain programming mode. A programming file must have been loaded.

```
enable_procedure [-name {name}] -action {action} -procedure {procedure} -enable {TRUE|FALSE}
```

Arguments

```
-name {name}  
-action {action}  
-procedure {procedure}  
-enable {TRUE|FALSE}
```

Supported Families

Exceptions

Example

In single programming mode:

```
enable_procedure -action {PROGRAM} -procedure {DO_ERASE} -enable {TRUE}
```

In chain programming mode:

```
enable_serialization -name {MyDevice2} -action {PROGRAM} -procedure {DO_ERASE} -enable {FALSE}
```

enable_serialization

To enable/disable serialization for a device. The device name parameter must be specified only in chain programming mode. A programming file allowing serialization must have been loaded.

```
enable_serialization [-name {name}] -enable {TRUE|FALSE}
```

Arguments

```
-name {name}  
    Specifies device user name  
-enable {TRUE|FALSE}
```

Supported Families

Exceptions

Example

In single programming mode:

```
enable_serialization -enable {TRUE}
```

In chain programming mode:

```
enable_serialization -name {MyDevice2} -enable {FALSE}
```

export_config

Exports a configuration file.

```
export_config -file {file}
```

Arguments

-file {file}

Specifies the file to export.

Supported Families

Exceptions

Example

export_script

Exports the history in a TCL script

```
export_script -file {file} -relative_path {TRUE|FALSE}
```

Arguments

-file {file}

Specifies the file to export.

-relative_path {TRUE|FALSE}

Specifies whether the file path must be exported as a relative path or an absolute path.

Supported Families

Exceptions

Example

```
export_script -file {./history.tcl} -relative_path {FALSE}
```



export_stapl

Exports the ChainBuilder STAPL file in chain programming mode.

```
export_stapl -file {file}
```

Arguments

-file {file}
Specifies the file to be exported.

Supported Families

Exceptions

Example

export_single_stapl

Exports a single device STAPL file.

```
export_single_stapl -file {file} [-name {name}]
```

Arguments

-file {file}
Specifies the file to export.
-file {file}
Specifies the name of the device in chain mode to export a single device STAPL file.

Supported Families

Exceptions

Example

Single Mode
export_single_stapl -file {D:/STAPL/my_design.stp}
Chain Mode
export_single_stapl -name {A3P250} -file {./my_design.stp}

export_single_svf

Exports a single device STAPL file.

```
export_single_svf -file {file} [-name {name}]
```

Arguments

-file {file}

Specifies the file to export.

-file {file}

Specifies the name of the device in chain mode to export a single device SVF file.

Supported Families

Exceptions

Example

Single Mode

```
export_single_svf -file {D:/STAPL/my_design.stp}
```

Chain Mode

```
export_single_svf -name {A3P250} -file {./my_design.stp}
```

export_single_1532

Exports a single device 1532 file.

```
export_single_1532 -file {file} [-name {name}]
```

Arguments

-file {file}

Specifies the directory to export 1532 files. -name {name} Specifies the name of the device in chain mode to export single device 1532 files.

Supported Families

Exceptions

Example

Single Mode

```
export_single_1532 -file {D:/1532/MY_DESIGN_1532}
```

Chain Mode

```
export_single_1532 -name {A3P250} -file {./MY_DESIGN_1532}
```



import_config

Note: Imports a configuration file.

```
import_config -file {file}
```

Arguments

-file {file}
Specifies the file to import.

Supported Families

Note:

Exceptions

Note:

Example

Note:

new_project

Creates a new FlashPro project or to convert an old ChainBuilder project into a new FlashPro project (the mode parameter must be 'chain' in this case).

```
new_project -name {name} -location {location} -mode {single|chain} [-convert_chb {convert_chb}]
```

Arguments

-name {name}
Specifies the project name.

-location {location}
Specifies the project location.

-mode {single|chain}
Specifies programming mode; either single or chain.

-convert_chb {convert_chb}
An optional argument that specifies the ChainBuilder project to be converted.

Supported Families

Exceptions

Example

```
new_project -name {FPPrj1} -location {./FPProject1} -mode {single}
new_project -name {FPPrjChb} -location {./ChbProject1} -mode {chain} -convert_chb
{./chb_prj/prj1.chb}
```

open_project

Opens a FlashPro project.

```
open_project -project {project}
```

Arguments

-project {*project*}
Specifies the project name.

Supported Families

Exceptions

Example

```
open_project -project {./FPProject1/FPPrj1.pro}
```

paste_device

Pastes the devices that are on the clipboard in the chain, immediately above the *position_name* device, if this parameter is specified. Otherwise it places the devices at the end of the chain. The chain programming mode must be set.

```
paste_device [-position_name {position_name}]
```

Arguments

-position_name {*position_name*}
Optional argument that specifies the name of a device in the chain.

Supported Families

Exceptions



Examples

```
paste_device  
paste_device -position_name {MyDevice3}
```

ping_prg

Runs ping on one or more programmers.

```
ping_prg (-name {name})*
```

Arguments

-name {*name*}

Specifies the programmer to be pinged. Repeat this argument for multiple programmers.

Supported Families

Exceptions

Example

```
ping_prg -name {FP300085} -name {FP300086}
```

refresh_prg_list

Refreshes the programmer list. This is particularly useful to have FlashPro detect a programmer that you have just connected.

```
refresh_prg_list
```

Arguments

Supported Families

Exceptions

Example

```
refresh_prg_list
```

remove_device

Note: Removes the device from the chain. Chain programming mode must be set.

```
remove_device (-name {name})*
```

Arguments

-name {*name*}

Specifies the device name. You can repeat this argument for multiple devices.

Supported Families

Note:

Exceptions

Note:

Example

Note:

remove_prg

Removes the programmer from the programmer list.

```
remove_prg (-name {name})*
```

Arguments

-name {*name*}*

Specifies the programmer to be removed. You can repeat this argument for multiple programmers.

Supported Families

Exceptions

Example



run_selected_actions

Runs the selected action on the specified programmer and returns the exit code from the action. If no programmer name is specified, the action is run on all connected programmers. Only one exit code is returned, so return code cannot be used when action is run on more than one programmer. A programming file must be loaded.

```
run_selected_actions [(-name {name}) *]
```

Arguments

-name {*name*}

Optional argument that specifies the programmer name. You can repeat this argument for multiple programmers.

Supported Families

Exceptions

Example

```
run_selected_action
```

```
run_selected_action -name {FP300085} -name {FP300086}
```

Example using return code:

```
if {[catch {run_selected_actions} return_val]} {puts "Error running Action"} else {puts "exit code $return_val"}
```

Example returning exit code to the command line (returns exit 99 on script failure, otherwise returns exit code from selected action):

```
if {[catch {run_selected_actions} return_val]}{exit 99} else {exit $return_val}
```

save_log

Saves the log file.

```
save_log -file {file}
```

Arguments

-file {*file*}

Specifies the log filename.

Supported Families

Exceptions

Example

save_project

Saves the FlashPro project as under a new project name.

```
save_project
```

Arguments

Supported Families

Exceptions

Example

```
save_project
```

save_project_as

Saves the FlashPro project as under a new project name.

```
save_project_as -name {name} -location {location}
```

Arguments

- name {*name*}
Specifies the project name.
- location {*location*}
Specifies the project location.

Supported Families

Exceptions

Example

```
save_project_as -name {FPPPrj2} -location {./FPPProject2}
```

scan_chain_prg

Runs Scan Chain on a programmer.

```
scan_chain_prg (-name {name})*
```



Arguments

- name *{name}*
Specifies the programmer name.

Supported Families

Exceptions

Example

select_serial_range

Selects the serialization data. Either the *data* or the *from_data* and *to_data* arguments must be specified. A programming file allowing serialization must be loaded.

```
select_serial_range [-name {name}] [(-data {data})*] [-from_data {from_data} -to_data {to_data}]
```

Arguments

- name *{name}*
Specifies the device name. This argument must be specified in chain programming mode.
- data *{data}*
Specifies the data. You can repeat this argument for multiple serialization data.
- from_data *{from_data}*
Specifies the start of a range of data. Must be used along with the *to_data* argument.
- to_data *{to_data}*
Specifies the end of a range of data. Must be used along with the *from_data* argument.

Supported Families

Exceptions

Examples

in single programming mode:

```
select_serial_range -data {1} -data {2} -data {3} -data {4}
select_serial_range -from_data {1} -to_data {4}
```

in chain programming mode:

```
select_serial_range -name {MyDevice1} -data {1} -data {2} -data {3} -data {4}
select_serial_range -name {MyDevice1} -from_data {1} -to_data {4}
```

select_target_device

Selects the target device in single programming mode. Single programming mode must be set.

```
select_target_device -name {name}
```

Arguments

-name {*name*}
Specifies the target device name.

Supported Families

Exceptions

Example

```
select_target_device -name {A3PE600(1)}
```

self_test_prg

Runs Self-Test on a programmer.

```
self_test_prg (-name {name})*
```

Arguments

-name {*name*}
Specifies the programmer name. You can repeat this argument for multiple programmers.

Supported Families

Exceptions

Example

set_bsdfile

To set a BSDL file to a non-Actel device in the chain. Chain programming mode must have been set. The device must be a non-Actel device.

```
set_bsdfile -name {name} -file {file}
```



Arguments

- name {*name*}
Specifies the device name.
- file {*file*}
Specifies the BSDL file.

Supported Families

Exceptions

Example

```
set_bsd_file -name {MyDevice3} -file {e:/design/bsd/ NewBSDL2. bsd}
```

set_chain_param

Sets the chain parameters in single programming mode. Single programming mode must be set .

```
set_chain_param [-pre_ir {pre_ir}] [-pre_dr {pre_dr}] [-post_ir {post_ir}] [-post_dr {post_dr}]
```

Arguments

- pre_ir {*pre_ir*}
Specifies the pre IR length.
- pre_dr {*pre_dr*}
Specifies the pre DR length.
- post_ir {*post_ir*}
Specifies the post IR length.
- post_dr {*post_dr*}
Specifies post DR length.

Supported Families

Exceptions

Example

```
set_chain_param -pre_ir {2} -pre_dr {3} -post_ir {4} -post_dr {5}
```

set_device_ir

Sets the IR length of a non-Actel device in the chain. Chain programming mode must be set. The device must be a non-Actel device.

```
set_device_ir -name {name} -ir {ir}
```

Arguments

- name {*name*}
Specifies the device name.
- ir {*ir*}
Specifies the IR length.

Supported Families

Exceptions

Example

```
set_device_ir -name {MyDevice4} -ir {2}
```

set_device_name

Changes the user name of a device in the chain. Chain programming mode must be set .

```
set_device_name -name {name} -new_name {new_name}
```

Arguments

- name {*name*}
Specifies the old device name.
- new_name {*new_name*}
Specifies the new device name.

Supported Families

Exceptions

Example

```
set_device_name -name {MyDevice4} -new_name {MyDev4}
```

set_device_order

Sets the order of the devices in the chain to the order specified. Chain programming mode must have been set. Unspecified devices will be at the end of the chain.

```
set_device_order (-name {name})*
```



Arguments

-name {*name*}

Specifies the device name. To specify a new order you must repeat this argument and specify each device name in the order desired.

Supported Families

Exceptions

Example

```
set_device_order -name {MyDevice3} -name {MyDevice1} -name {MyDevice4}
```

the new order is:

```
MyDevice3 MyDevice1 MyDevice4 MyDevice2
```

set_device_tck

Sets the maximum TCK frequency of a non-Actel device in the chain. Chain programming mode must be set. The device must be a non-Actel device.

```
set_device_tck -name {name} -tck {tck}
```

Arguments

-name {*name*}

Specifies the device name.

-tck {*tck*}

Specifies the maximum TCK frequency (in MHz).

Supported Families

Exceptions

Example

```
set_device_tck -name {MyDevice4} -tck {2.25}
```

set_device_type

Changes the family of an Actel device in the chain. The device must be an Actel device. The device parameter below is now optional.

```
set_device_type -name {name} -type {type}
```

Arguments

- name {*name*}
Specifies
- type {*type*}
Specifies the device family.

Supported Families

Exceptions

Example

```
set_device_type -name {MyDevice2} -type {A3PE600}
```

set_main_log_file

Sets the FlashPro log file.

```
set_main_log_file -file {file}
```

Arguments

- file {*file*}
Specifies the log file.

Supported Families

Exceptions

Example

```
set_main_log_file -file {e:/log/log1000.txt}
```

set_prg_name

Changes the user name of a programmer.

```
set_prg_name -name {name} -new_name {new_name}
```

Arguments

- name {*name*}
Specifies the old programmer name.
- new_name {*new_name*}
Specifies the new programmer name.



Supported Families

Exceptions

Example

```
set_prg_name -name {FP300086} -new_name {FP3Prg2}
```

set_programming_action

Selects the action for a device. The device name parameter must be specified only in chain programming mode. A programming file must be loaded. The device must be an Actel device.

```
set_programming_action [-name {name}] -action {action}
```

Arguments

- name {*name*}
Specifies the device name.
- action {*action*}
Specifies the action.

Supported Families

Exceptions

Example

in single programming mode:

```
set_programming_action -action {PROGRAM}
```

in chain programming mode:

```
set_programming_action -name {MyDevice1} -action {ERASE}
```

set_programming_file

Sets the programming file of a device. Either the *file* or the *no_file* flag must be specified. A programming file must be loaded. The device must be an Actel device .

```
set_programming_file [-name {name}] [-file {file}] [-no_file { }]
```

Arguments

- name {*name*}
Specifies the device name. This argument must be specified only in chain programming mode.
- file {*file*}

- Specifies the programming file.
- no_file
 - Specifies to unload the current programming file.

Supported Families

Exceptions

Examples

```
in single programming mode:
set_programming_file -file {e:/design/pdb/TopA3P250.pdb}
in chain programming mode:
set_programming_file -name {MyDevice2} -file {e:/design/pdb/TopA3P250.pdb}
set_programming_file -name {MyDevice1} -no_file
```

set_programming_mode

Sets the programming mode.

```
set_programming_mode -mode {single|chain}
```

Arguments

- mode {single|chain}
 - Specifies the mode, either single programming or chain programming.

Supported Families

Exceptions

Examples

set_serialization_log_file

Sets the FlashPro log file to be used for serialization.

```
set_serialization_log_file -file {file}
```

Arguments

- file {file}
 - Specifies the log file name.



Supported Families

Exceptions

Example

```
set_serialization_log_file -file {e:/log/log1000_serial.txt}
```

set_serialization_mode

Sets the serialization mode.

```
set_serialization_mode -mode {skip|reuse}
```

Arguments

-mode {skip|reuse}

Supported Families

Exceptions

Example

Signature

Optional for .afm, .dio, and .fus file types.

update_programming_file

Updates the programming file with the selected parameters.

```
update_programming_file[(-name {name})*]
```

Arguments

-name {name}

Specifies the device name. This argument must be specified only in chain programming mode.

-feature {value}

Optional argument that specifies the features to be programmed. Select the silicon feature(s) you want to program.
Possible values for this option are:

Value	Description
setup_security	Sets up the security settings
prog_from	Programs the FlashROM
prog_fpga	Programs the FPGA Array

To program the Embedded Flash Memory Block, use the following EFM arguments: [-efm_block](#), [-efm_client](#), and [-efm_block_security](#).

`-signature {value}`

Optional argument that identifies and tracks Actel designs and devices.

`-from_content {name}`

Optional argument that identifies the source file for the FlashROM content. The file type is UFC or PDB (default). This argument only applies when programming the FlashROM (prog_from option).

`-from_config_file {name}`

Optional argument that specifies the location of the FlashROM configuration file. This argument only applies when programming the FlashROM (prog_from option) and the from_content is set to UFC.

`-number_of_devices{value}`

Optional argument that specifies the number of devices to be programmed. This argument only applies when FlashROM has serialization regions. This argument only applies when programming the FlashROM (prog_from option).

`-from_program_pages {value}`

Optional argument that identifies the program pages in FlashPoint. This argument only applies when programming the FlashROM (prog_from option).

`-custom_security {value}`

Optional argument that specifies the security level. This argument only applies when programming the security settings (setup_security) or programming previously secured devices. The following table shows the acceptable values for this argument:

Value	Description
Yes	Custom security level
No	Standard security level

`-security_permanent {value}`

Optional argument that specifies whether the security settings for this file are permanent or not. This argument only applies when programming the security settings (setup_security) or programming previously secured devices. The following table shows the acceptable values for this argument:

Value	Description
Yes	Permanently disables future modification of security settings
No	Enables future modifications of security settings

`-fpga_security_level {value}`

Optional argument that specifies the security level for the FPGA Array. This argument only applies when programming the security settings (setup_security) or programming previously secured devices. Possible values:



Value	Description
write_verify_protect	The security level is medium (standard) and the FPGA Array cannot be written or verified without a Pass Key
write_protect	The security level is write protected. The FPGA Array cannot be written without a Pass Key, but it is open for verification (custom FPGA)
encrypted	The security level is high (standard) and uses a 128-bit AES encryption
none	The FPGA Array can be written and verified without a Pass Key

`-from_security_level {value}`

Optional argument that specifies the security level for the FlashROM. This argument only applies when programming the security settings (setup_security) or programming previously secured devices. Possible values:

Value	Description
write_verify_protect	The security level is medium (standard) and the FlashROM cannot be read, written or verified without a Pass Key
write_protect	The security level is write protected. The FlashROM cannot be written without a Pass Key, but it is open for reading and verification (custom FlashROM)
encrypted	The security level is high (standard) and uses a 128-bit AES encryption
none	The FlashROM can be written and verified without a Pass Key

`-efm_block_security{location:X;security_level: value}`

This option is available only for Fusion; X identifies an Embedded Flash Memory Block instance from 0-3.

Possible values:

Value	Description
write_verify_protect	The security level is medium (standard) and the Embedded Flash Memory Block cannot be read, written or verified without a Pass Key
write_protect	The security level is write protected. The Embedded Flash Memory Block cannot be written without a Pass Key, but it is open for reading (custom FB)
encrypted	The security level is high (standard) and uses a 128-bit AES encryption
none	The Embedded Flash Memory Block can be written and read without a Pass Key

`-pass_key {value}`

Protects all the security settings for FPGA Array, FlashROM, and Embedded Flash Memory Block. The maximum length of this value is 32 characters. You must use hexadecimal characters for the pass key value.

`-aes_key {value}`

Decrypts FPGA Array and/or FlashROM and Embedded Flash Memory Block programming file content.

Max length is 32 HEX characters.

`-efm_content {location:X;source: value}`

This option is available only for Fusion; X identifies an Embedded Flash Memory Block from 0-3. Option identifies the source file for the Embedded Flash Memory Block configuration content, either an EFC or PDB file. If you specify EFC as your source, you need to specify the `-efm_block` parameter. Possible values:

Value	Description
PDB	Embedded Flash Memory Block configuration and content is taken from your PDB
EFC	FlashPoint uses the Embedded Flash Memory Block configuration and content from the EFC file specified in <code>-efm_block</code> parameter

`-efm_block {location:X;config_file: value}`

This option is available only for Fusion; X identifies an Embedded Flash Memory Block (EFMB) from 0-3.

`Config_file` specifies the location of the EFMB configuration file (must be an EFC file with full pathname).

`-efm_client {location:X;client:value; mem_file: value}`

This option is available only for Fusion; X identifies an EFMB from 0-3.

You must specify the client name and its memory content file for each client of EFMB you wish to program.

`Mem_file` specifies the file with the memory content for the client. If you want to program a client with a PDB or EFC file memory content (as defined by the `-efm_content` argument), the `mem_file` path should be empty (see [example 3](#)); but if a `mem_file` path is specified, the memory content from this file will overwrite the client content in PDB or EFC (as defined by the `-efm_content` argument).

`-tie_off_arch {value}`

This optional argument is used only for IT6X6M2 and M7IT6X6M2 devices. Possible values:

Value	Description
pull-down	Pull-down resistor: reduced quiescent power consumption
pull-up	Pull-up resistor: compatible behavior for migrated ProASIC ^{PLUS} designs

Supported Families

Exceptions

Example

Fusion example 1

```
update_programming_file \
-feature {setup_security} \
  -feature {prog_from} \
  -from_content {ufc} \
-from_config_file {D:/from_ah.ufc} \
-number_of_devices {1} \
-from_program_pages {1 2 3 4 5 6 7} \
-custom_security {no} \
-security_permanent {no} \
-fpga_security_level {write_verify_protect} \
```



```

-from_security_level {write_verify_protect} \
-efm_block_security {location:1;security_level:write_verify_protect} \
-efm_content {location:1;source:efc} \
-efm_block {location:1;config_file:{D:\ nvm_all_new.efc}} \
-efm_client {location:1;client:asb;mem_file:{}} \
-efm_client {location:1;client:cfiData;mem_file:{D:\cfid.mem}} \
-efm_client {location:1;client:ds;mem_file:{D:\ds.hex}} \
-efm_client {location:1;client:init1;mem_file:{D:\init1.hex}} \
-efm_client {location:1;client:raminit;mem_file:{}}

```

Update loaded PDB file: use ufc file for FlashROM configuration and content; use efc file for block 1 configuration; efc memory content will be overwritten by memory content from specified mem files for each client.

Fusion example 2:

```

update_programming_file \
-feature {prog_from} \
-from_content {pdb} \
-from_program_pages {1} \
-efm_content {location:1;source:pdb} \
-efm_client {location:1;client:cfiData;mem_file:{D:\cfid.mem}}

```

Update loaded PDB file: use pdb data for FlashROM; program only page 1; use pdb data for block 1; program only client cfiData; overwrite memory content for this client with memory content from the specified file.

Fusion example 3:

```

update_programming_file \
-efm_content {location:1;source:pdb} \
-efm_client {location:1;client:cfiData;mem_file:{D:\cfid.mem}} \
-efm_client {location:1;client:init1;mem_file:{}}

```

Update loaded PDB file: use pdb data for block 1; program client cfiData using memory content from the specified file; program client init1 using memory content from pdb (no mem_file path is specified)

Troubleshooting

Loopback Test

The console software supports all JTAG functions and a diagnostic test.

To perform the diagnostic test

1. Connect the loopback test board to the FlashPro.
2. Connect the FlashPro to the parallel/USB port of the PC.
3. Power-on the FlashPro.
4. From the **Start** menu, choose **Programs > Actel FlashPro > Diagnostics**. This opens the diagnostics console program.
5. Connect to the FlashPro by entering *openport lpt1* or *openport usb*. The parallel port number depends on the port used to connect the FlashPro.
6. Enter *test*. The unit runs into self-test mode. Do not interrupt the unit during self-test mode.

Note: To see a complete list of all functions, enter *help*. To get a more detail description of each function, enter *help <command>*.

Fusion, IGLOO, and ProASIC3 Families Exit Codes

The table below lists exit codes Fusion, IGLOO, and ProASIC3 family devices.

Note: Exit codes with positive integers are reserved for current and future standard EXIT codes of the STAPL standard. Exit codes with negative integers are reserved for vendor-specific EXIT codes.

Table 20 · Exit Codes for Fusion, IGLOO, and ProASIC3 Family Devices

Exit Code	Exit Message	Possible Cause	Possible Solution
0	This message means passed. This does not indicate an error.		
1	A physical chain does not match the expected set up from the STAPL file. Also known as Checking Chain Error.	Physical chain configuration has been altered. Something has become disconnected in the chain. The specific IR length of non-Actel devices may be incorrect. The order of the specified chain may be incorrect.	
5	Failed to enter programming mode.	Older software or programming file used. Signal integrity issues on JTAG pins. Bad device.	Generate STAPL file with the latest version of Designer/FlashPro. Use latest version of FlashPro software. Measure JTAG voltages, noise, and reflection. Try programming with a new device.

Exit Code	Exit Message	Possible Cause	Possible Solution
6	Failed to verify IDCODE.	Incorrect programming file. Incorrect device. Signal integrity issues on JTAG pins.	Choose the correct programming file and select the correct device. Measure JTAG pins and noise or reflection. TRST should be floating or tied high. Reduce the length of ground connection.
8	Failed Erase Operation.	Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG voltages, noise, and reflection. Try programming with a new device.
10	Failed to program FPGA array at row ", rowNumber,".	Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG voltages, noise, or reflection. Try programming with a new device.
10	Failed to enable FPGA Array.	Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG voltages, noise, or reflection. Try programming with a new device.
10	Failed to program FlashROM.	Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG voltages, noise, and reflection. Try programming with a new device.
11	Verify 0 failed at row",rowNumber,". Verify 1 failed at row",rowNumber,". Failed to verify FlashROM at row",from rowNumber-1.	Device is programmed with a different design. Signal integrity issues on JTAG pins. Bad device.	Run VERIFY_DEVICE_INFO to verify the device is programmed with the correct data/design. Monitor VPUMP voltage during programming. Measure JTAG voltages, noise and reflection. Try programming with a new device.
14	Failed to program Silicon Signature. Failed to program security lock settings.	Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG voltages, noise, and reflection. Try programming with a new device.



Exit Code	Exit Message	Possible Cause	Possible Solution
-18	Failed to authenticate the encrypted data.	Generation failure for encrypted data.	Verify the programming file is generated from the latest version of Designer/FlashPro.
-20	Failed to verify FlashROM at row ", FRowRowIndex-1.	Device is programmed with a different design. Signal integrity issues on JTAG pins. Bad device.	Program with the correct data/design. Monitor VPUMP level during programming. Measure JTAG pins and noise or reflection. Try programming with a new device.
-22	Failed to program pass key.	Unstable VPUMP voltage level. Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG voltages, noise, and reflection. Try programming with a new device.
-23	Failed to program AES key.	Unstable VPUMP voltage level. Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG pins and noise or reflection.
-24	Failed to program UROW.	Unstable VPUMP voltage level. Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG voltages, noise, and reflection. Make sure you mounted 0.01iF and 0.33iF caps on V _{pump} (close to the pin). Try programming with a new device.
-25	Failed to enter programming mode.	Signal integrity issues on JTAG pins. Bad device.	Measure JTAG voltages, noise, and reflection. Try programming with a new device.
-26	Failed to enter programming mode.	Signal integrity issues on JTAG pins. Bad device.	Measure JTAG voltages, noise, and reflection. Try programming with a new device.
-27	FlashROM Write/Erase is protected by the passkey. A valid passkey needs to be provided.	File contains no passkey and device is secured with a passkey. Passkey in the file does not match device.	Provide a programming file with a passkey that matches the passkey programmed into the device.
-28	FPGA Array Write/Erase is protected by the passkey. A valid pass key needs to be provided.	File contains no passkey and device is secured with a passkey. Passkey in the file does not match	Provide a programming file with a passkey that matches the passkey programmed into the device.

Exit Code	Exit Message	Possible Cause	Possible Solution
		device.	
-29	FlashROM Read is protected by passkey. A valid passkey needs to be provided.	File contains no passkey and device is secured with a passkey. Passkey in the file does not match device.	Provide a programming file with a pass key that matches the passkey programmed into the device
-30	FPGA Array verification is protected by a passkey. A valid passkey needs to be provided.	File contains no passkey and device is secured with a passkey. Passkey in the file does not match device.	Provide a programming file with a passkey that matches the passkey programmed into the device.
-31	Failed to verify AES key.	AES key in the file does not match the device. Unstable JTAG/VPUMP voltage level.	Provide a programming file with an AES key that matches the AES key programmed into the device. Monitor VPUMP/VJTAG voltage during programming. Measure JTAG voltages, noise, and reflection.
-32	Failed to verify IDCODE. Target is an M7 device.	File is not for M7, but target device is an M7. Signal integrity issues on JTAG pins.	Check that the target device is M7 enabled. Make sure programming file generated is for M7 enabled device. Measure JTAG pins , noise, and reflection.
-32	Failed to verify IDCODE. Target is an M1 device.	File is not for M1, but target device is an M1 device. Signal integrity issues on JTAG pins.	Check that the target device is M1 enabled. Make sure programming file generated is for M1 enabled device. Measure JTAG pins, noise, and reflection.
-32	Failed to verify IDCODE. Core enabled device detected.	File is not for target device. Signal integrity issues on JTAG pins	Check the target device. Make sure programming file generated for target device. Measure JTAG voltages, noise, and reflection.
-32	Failed to verify IDCODE. The target is not an M7 device.	File is for M7, but target device is not M7. Signal integrity issues on JTAG pins.	Check that the target device is not M7 enabled. Make sure programming file generated is for non M7 enabled device. Measure JTAG voltages, noise, and reflection.



Exit Code	Exit Message	Possible Cause	Possible Solution
-32	Failed to verify IDCODE. The target is not an M1 device.	File is for M1, but target device is not an M1 device.	Check that the target device is not M1 enabled. Make sure programming file generated is for non M1 enabled device. Measure JTAG voltages, noise, and reflection.
-33	FPGA Array encryption is enforced. A programming file with encrypted FPGA array data needs to be provided.	File contains unencrypted array data, but device contains AES key.	Provide a programming file with an encrypted FPGA Array data.
-34	FlashROM encryption is enforced. A programming file with encrypted FlashROM data needs to be provided.	File contains unencrypted FlashROM data, but the device contains an AES key.	Provide a programming file with an encrypted FlashROM data.
-35	Failed to match pass key.	Pass key in file does not match pass key in device.	Provide a programming file with a pass key that matches the pass key programmed into the device.
-38	Failed to program pass key.	Unstable VPUMP voltage level. Signal integrity issues on JTAG pins. Bad device.	Monitor VPUMP voltage during programming. Measure JTAG pins and noise or reflection. Try programming with a new device.
-39	Failed to verify Embedded Flash Block (EFBM).	Device is programmed with a different design. Signal integrity issues on JTAG pins. The EFMB data was modified through user FPGA design after programming; this could occur during standalone verify. Bad device.	Verify the device is programmed with the correct data/design. Monitor VPUMP voltage during programming. Measure JTAG pins and noise or reflection. Try programming with a new device.
-40	Embedded Flash Memory Block (EFMB) MAC Failure.	Data in the file is encrypted with a different AES key than the device.	Verify the programming file is generated from the latest version of Designer/FlashPro.
-41	Error programming Embedded Flash Block.	Signal integrity issues on JTAG pins. Bad device.	Measure JTAG pins and noise or reflection. Try programming with a new device.
-42	Failed to verify security settings.	File security settings do not match device.	Provide a programming file with security setting that match the security settings programmed into the device.

Exit Code	Exit Message	Possible Cause	Possible Solution
-43	Failed to verify design information.	File checksum and design name do not match the device.	Verify the device is programmed with the correct data and design.
-44	Failed to verify AES key.	The AES key in the file does not match the AES key in the device. File does not contain an AES key and the device is secured with an AES key.	Provide a programming file with an AES key that matches the AES key programmed into the device.
-45	Device package does not match the programming file.		
-47	Embedded Flash Memory Block (EFMB), block X encryption is enforced. A programming file with encrypted EFMB data needs to be provided.	The programming EFMB data is not encrypted, but the device contains an AES key with encryption enforced.	Provide a programming file with encrypted EFMB data.

ProASIC^{PLUS} and ProASIC Exit Codes

This section describes the exit codes for ProASIC^{PLUS} and ProASIC family devices.

Table 21 · ProASIC^{PLUS} and ProASIC Family Devices Exit Codes

Exit Code	Exit Message	Possible Cause	Possible Solution
0	This message means passed. This does not indicate an error.		
1	A physical chain does not match the expected set up from the STAPL file. Also known as Checking Chain Error.	Physical chain configuration has been altered. Something has become disconnected in the chain. The specific IR length of non-Actel devices may be incorrect. The order of the specified chain may be incorrect.	
2	There is a reading device ID failure.	The device either does not have a valid device ID or the data cannot be read correctly.	Check the device ID.
3	This occurs when using ProASIC ^{PLUS} devices.	Connect was set up for a ProASIC device and the device is actually ProASIC ^{PLUS} .	Set up for a ProASIC ^{PLUS} device.
5	Programming set up problem. Also known as Entering ISP Failure.	The A500K device senses the VDDL power supply as being on.	Power the VDDL down during programming. Check the device has the correct voltages on VDDP, VDDL, VPP, and VPN.



Exit Code	Exit Message	Possible Cause	Possible Solution
6	The IDCODE of the target device does not match the expected value in the STAPL file. This is a JEDEC standard message.	<p>The device targeted in the STAPL file does not match the device being programmed.</p> <p>User selected wrong device.</p> <p>Device TRST pin is grounded.</p> <p>Noise or reflections on one or more of the JTAG pins caused by the IR Bits reading it back incorrectly.</p>	<p>Choose the correct STAPL file and select the correct device.</p> <p>Measure JTAG pins and noise or reflection. TRST should be floating or tied high.</p> <p>Cut down the extra length of ground connection.</p>
7	Unknown algorithm: alg=x, prev=x Invalid data read from device	<p>This occurs with current STAPL files when the revision written into the factory row is not rev 1 for ProASICPLUS or rev 2 for ProASIC devices family devices. The STAPL files from last year may "exit 7" with newer devices or the older revision may cause this failure if the STAPL file used is from latest version.</p> <p>It can occur if you are using Engineering Sample parts that are no longer supported, such as ProASIC Engineering Sample parts.</p> <p>This error can also occur if the programmer has trouble reading the factory row due to signal noise, crosstalk, or reflections on the JTAG signal and clock lines.</p> <p>It can occur if you program an -F ProASICPLUS device with an old STAPL file.</p> <p>This error occurs if you connected VPP and VPN the wrong way.</p> <p>It occurs if there are no bypass Caps on VPP VPN, which damaged the device.</p>	<p>Re-generate STAPL file from Designer 6.1 SP1.</p> <p>Replace A500K ES parts with commercial parts.</p> <p>Double check VPP and VPN connections.</p> <p>Make sure VPP and VPN have correct bypass caps.</p>
8	FPGA failed during the erase operation.	<p>The device is secured, and the corresponding STAPL file is not loaded.</p> <p>The device has been permanently secured and cannot be unlocked.</p>	<p>Load the correct STAPL file.</p>
11	FPGA failed verify	<p>The device is secured and the corresponding STAPL file is not loaded.</p> <p>You used the Libero IDE software v2.3 or earlier or the Designer R1-2003 software or earlier to generate the STAPL file.</p> <p>VPN caps were soldered in the wrong polarity.</p>	<p>Load the correct STAPL file.</p> <p>Use later software versions —at least Libero v2.3 SP1 and Designer R1-2003 SP1.</p> <p>Double-check the VPN bypass caps polarity.</p>

Exit Code	Exit Message	Possible Cause	Possible Solution
12	Security is enabled.	The device is secured and the wrong key/STAPL file was entered. The device is damaged. The verification was interrupted and therefore fails, causing the software to think the device is secure.	
14	Program security failure.		
15	This is a factory Calibration Data CRC error.	During program, erase, or verify, you must read back Calibration Data from the FPGA. The data contains a CRC. You use the CRC to ensure the data is not corrupted/wrong. Device is damaged. Noise on the FTAL signals causes the programmer to read back wrong data.	
17	The device has been secured. Write-security is enabled.	The device is secured and the wrong key or STAPL file was entered. The device is damaged.	Load the correct STAPL file.
80	Error code results from STAPL files for A500K devices.	An internal calibration (based on DDP and VPP) failed.	Check voltages on the device pins. Check voltages on the VDDP and VPP pins.
90	Unexpected RCK detected.	Noise on the RCK signal. You connected a CLK source to the RCK signal. The polarized bypass capacitors on VPP or VPN are reversed-biased and are affecting the programmer's VPP or VPN output voltage. This causes programming to fail. Several FlashPros are programming at the same time and are too close to each other. Programmer not properly installed by Admin.	Disconnect the RCK and make sure TCK has a clean signal. Separate FlashPros away from each other while they are programming Internal ISP. Connect programmer as an Admin in FlashPro.
91	Calibration data parity error.	Device is damaged.	Replace the device.
	Null	Several FlashPros are programming at the same time and are too close to each other. FlashPro connects to PC parallel port through a dongle key. Data length mismatch when performing DRSCAN on STAPL file.	



Exit Code	Exit Message	Possible Cause	Possible Solution
	Cable to target is not connected properly.	When the Analyze command is executed, the FlashPro looks for target devices. If the cable connection is wrong, FlashPro assumes that nothing is connected at all.	Confirm the connection between the header to the device. If the board supplies the power to the device, make sure the voltage level is correct.
	Chain integrity test failed: xx	<p>The connection between the FlashPro programmer and the device is broken.</p> <p>The programmer cable might not be securely inserted into the header.</p> <p>The header is not connected to the JTAG pins of the FPGA correctly.</p> <p>The configuration setting (ProASIC/ProASICPLUS) does not match the target device.</p> <p>Noise or reflections on the JTAG pins has caused communication between the programmer and the device to fail.</p> <p>A dongle is plugged in between the PC parallel port and the FlashPro parallel port cable.</p>	<p>Secure the connections.</p> <p>Check the JTAG pins for signal activity.</p> <p>Check for broken TDO, TMS, and TCK pins.</p> <p>After checking all type of connections if the failure exists, you may need to replace the first device (the devices closest to the TDO of the programming header) in the chain.</p> <p>Remove the dongle.</p>
	Could not connect to programmer on port lp1 or parallel port device does not support IEEE-1284 negotiation protocol	The remote device does not respond to the negotiation protocol, for a variety of reasons.	<p>Make sure the port is connected.</p> <p>Make sure the connected device is a FlashPro/Lite programmer.</p> <p>Turn the programmer on.</p> <p>Check parallel port setting in BIOS.</p> <p>Make sure that there are no dongles in between the parallel port and the FlashPro connection.</p> <p>Try another parallel cable, the parallel cable might be defective.</p> <p>Check to see if the programmer is damaged.</p> <p>Make sure the FlashPro Lite has power. The FlashPro Lite is powered from the target board through the Vdd pin of the programming header.</p> <p>Make sure the Vdd pin is connected and the target board is powered up.</p> <p>Secure the connection between the cable connector and the programming header.</p> <p>Before you program any devices, you should run the self-diagnostic test (see "Self-test" on page 16). The diagnostic</p>

Exit Code	Exit Message	Possible Cause	Possible Solution
			software can be found on the Actel web site. If the test fails, please contact Actel Customer Technical Support at tech@actel.com for credit and replacement. Note: The Self-test is only available for FlashPro, not FlashPro Lite.
	External voltage detected on <Supply>	The voltage supply for the FPGA is driven by another source (board, external power-supply), but the user forgot to turn off the supply in the Connect menu.	Set appropriate options in the Connect menu.
	VDPP Disconnected.	There is no Vddp voltage supply to the FPGA. You accidentally turned off the Vddp supply in the Connect menu. The Vddp supply on the board is not functioning.	Check the Vddp supply on the board for appropriate voltages and correct the Connect menu.
	More than one unidentified device.		
	If you want to perform an operation on the ProASIC device, the rest of the devices in the chain must be in bypass mode. To put devices in bypass mode, select Configuration > Chain Parameter (or click the Chain Parameter button in the Single STAPL Configuration window), then set the Pre IR, Pre DR, Post IR or Post DR.	STAPL settings of Pre IR, Pre DR, Post IR, and Post DR do not match the chain configuration. One or more of the devices in the chain is damaged and the ID CODE cannot be read back.	Make sure you have set Pre IR, Pre DR, Post IR, and Post DR to match the chain configuration. If you are still experiencing the failure, it is likely that the device's ID CODE cannot be read and you need to replace the device.
	Cannot find the programmer with ID xxx	The programmer is removed from the PC.	Delete programmer (or reconnect programmer) and select the Refresh Programmer button. See Connecting Programmers for more information.
	Fatal Error: Please check programmer set up.	Software cannot resolve the error encountered in the programmer.	Save the project file, restart the software, and power cycle the programmer.
	External voltage xxx mV is detected on xxx.	You have specified the programmer to drive the xxx but external xxx is detected.	Deselect the xxx in the programmer setting.
	Executing action xxx failed.	The STAPL runtime failed.	
	Executing action xxx with serial index/action xx failed.	The STAPL runtime failed.	



Exit Code	Exit Message	Possible Cause	Possible Solution
	No Vpump voltage source is detected.		Select the Vpump in the Programmer setting. Make sure the external Vpump is properly turned on.
	Vpump short detected.		Use a different programmer. If the problem persists, check the board layout.
	xxx Mhz TCK frequency in this STAPL file is not supported by the FlashPro Lite detected. It supports only 4 MHz TCK frequency.		Check FlashPro Lite version being used. Use FlashPro Lite Rev C or modify the STAPL file to 4 MHz.
	xxx Mhz TCK frequency in this STAPL file is not supported by the FlashPro Lite RevC detected. It supports only 1, 2 or 4 Mhz TCK frequency.		Modify STAPL file to 1, 2, or 4 MHz.
	Cannot find the serial Index/Action xxx in STAPL file.	Mismatch between STAPL file and the Index/Action selection.	Make sure the STAPL file was not overwritten. Save the project with updated serial/action selection.
	Duplicated serial Index/Action xxx was removed.	Mismatch between STAPL file and the Index/Action selection.	Make sure the STAPL file was not overwritten. Save the project with updated serial/action selection.
	Using local backup copy xxx	Cannot find original copy.	Check for available space on the disk. Check that write permissions are enabled.
	FlashPro cannot rename the programmer/device with an existing name.	Name is already in use.	Create a new name.
	FlashPro cannot rename the programmer/device with an invalid character.	Invalid character used in programmer/device name.	Do not use invalid characters.
	Automatic check for updates.		FlashPro can check the Actel website to find if an updated version of the software is available. If you would like to have FlashPro automatically check for software updates, choose Preferences from the File menu. From the Updates tab, you can choose your automatic software update settings. You can also select Software Updates from the Help menu for updates to the FlashPro software.

Exit Code	Exit Message	Possible Cause	Possible Solution
	FlashPro parse error.	FlashPro software failed to parse the file.	
	FlashPro does not support STAPL files for xxx.	STAPL file not allowed.	Use a STAPL file for your device that is supported by FlashPro.

Cannot find the programmer with ID 'xxx'

This section contains a description and a solution for the Cannot find the programmer with ID 'xxx' error message.

Description

The programmer is removed from PC.

Solution

Delete programmer (or reconnect programmer) and select the **Refresh Programmer** button.

See [Connecting Programmers](#) for more information.

Fatal Error: Please check programmer setup.

This section contains a description and a solution for the "Fatal Error: Please check programmer setup error message."

Description

Software cannot resolve the error encountered in the programmer.

Solution

Save the project file, restart the software, and power cycle the programmer.

External voltage 'xxx' mV is detected on 'xxx'

This section contains a description and a solution for the External Voltage 'xxx' mV is detected on 'xxx' error message.

Description

You have specified the programmer to drive the 'xxx,' but external 'xxx' is detected.

Solution

Please deselect the 'xxx' in the programmer setting.

Executing action xxx failed

This error message means that the STAPL runtime failed.

Executing action xxx with serial index/action xxx failed

This error message means that the STAPL runtime failed.

No Vpump voltage source is detected

This section contains a description and solutions for the "No Vpump voltage source is detected. Please provide Vpump voltage source" error message.

Description

For a description of this error message, refer to

Solutions

- Select the Vpump in the **Programmer** setting.
- Make sure the external Vpump is properly turned on.



Vpump short detected

This section contains a solution for the "Vpump short detected" error message.

Solution

Use a different programmer. If problem persists, please check board layout.

xxx Mhz TCK frequency in this STAPL file is not supported by the FlashPro Lite detected. It supports only 4 Mhz TCK frequency.

This section contains a description and solutions for the "xxx Mhz TCK frequency in this STAPL file is not supported by the FlashPro Lite detected. It supports only 4 Mhz TCK frequency" error message.

Description

FlashPro Lite non RevC supports only 4 MHz.

Solution

Please use FlashPro Lite RevC or modify the STAPL file to 4MHz.

xxx Mhz TCK frequency in this STAPL file is not supported by the FlashPro Lite RevC detected. It supports only 1, 2 or 4 Mhz TCK frequency.

This section contains a description and solutions for the "xxx Mhz TCK frequency in this STAPL file is not supported by the FlashPro Lite RevC detected. It supports only 1, 2 or 4 Mhz TCK frequency" error message.

Description

FlashPro Lite RevC supports only 1, 2 or 4 MHz.

Solution

Please modify STAPL file to 1, 2 or 4 MHz.

Cannot find the serial Index/Action 'xxx' in STAPL file

This section contains a description and solutions for the "Cannot find the Serial Index/Action 'xxx' in STAPL file. The serial Index/Action ignored" warning message.

Description

Mismatch between STAPL file and the Index/Action selection.

Solutions

- Make sure STAPL file was not overwritten.
- Save the project with updated serial index/action selection.

Duplicated serial Index/Action 'xxx' was removed

This section contains a description and solutions for the "Duplicated serial Index/Action 'xxx' was removed" warning message.

Description

Mismatch between STAPL file and Index/Action selection.

Solutions

- Make sure STAPL file was not overwritten.
- Save the project with updated serial index/action selection.

Using local backup copy xxx

This section contains a description for the "Using local backup copy xxx" warning message.

Description

The original copy is not found; the local backup copy is used instead.

Action

- Check for available space on the disk.
- Check that write permissions are enabled.

Flashpro cannot rename the programmer/device with an existing name

This section contains a description for the "FlashPro cannot rename the programmer/device with an existing name" error message.

Description

The FlashPro software failed to rename the programmer/device because the name is already in use.

Flashpro cannot rename programmer/device with an invalid character

This section contains a description for the "FlashPro cannot rename programmer/device with an invalid character" error message.

Description

The FlashPro software failed to rename the programmer/device because of invalid characters.

Automatic check for updates

This section contains a description for the "Automatic check for updates" message.

Description

FlashPro can check the Actel website to find if an updated version of the software is available. If you would like to have FlashPro automatically check for software updates, choose **Preferences** from the **File** menu. From the **Updates** tab, you can choose your automatic software update settings.

You can also select **Software Updates** from the **Help** menu for updates to the FlashPro software.

FlashPro parse error

This section contains a description for the "Failed to PARSE File xxx" error message.

Description

The FlashPro software failed to parse the file.

STAPL file not allowed

This section contains a description for the "FlashPro does not support STAPL files for "xxx" message.

Description

FlashPro does not support STAPL files for your device.



Electronic Parameters

DC Characteristics for FlashPro3

The table below shows DC characteristics for FlashPro3.

Table 22 · DC Characteristic for FlashPro3

Description	Symbol	Min	Max	Unit
Input low voltage, TDO	VIL	-0.5	0.35*VJTAG	V
Input high voltage, TDO	VIH	0.65*VJTAG	3.6	V
Input current, TDO	IIL, IIH	-20	+20	mA
Input capacitance, TDO			40	pF
Output voltage, VPUMP, operating	VPP	+3.0	+3.6	V
Output current, VPUMP	IPP		250	mA
VJTAG = 1.5V				
Output low voltage, TCK, TMS, TDI, 100µA load	VOL	0.0	0.2	V
Output low voltage, TCK, TMS, TDI, 4mA load	VOL	0.0	0.30*VJTAG	V
Output high voltage, TCK, TMS, TDI, 100µA load	V	VJTAG-0.2	VJTAG	V
Output high voltage, TCK, TMS, TDI, 4mA load	VOH	0.70*VJTAG	VJTAG	V
Output current, TCK, TMS, TDI	IOL, IOH	-4	+4	mA
VJTAG = 2.5V				
Output low voltage, TCK, TMS, TDI, 100µA load	VOL	0.0	0.2	V
Output low voltage, TCK, TMS, TDI, 8mA load	VOL	0.0	0.6	V
Output high voltage, TCK, TMS, TDI, 100µA load	VOH	VJTAG-0.2	VJTAG	V
Output high voltage, TCK, TMS, TDI, 8mA load	VOH	1.8	VJTAG	V
Output current, TCK, TMS, TDI	IOL, IOH	-8	+8	mA
VJTAG = 3.3V				
Output low voltage, TCK, TMS, TDI, 100µA load	VOL	0.0	0.2	V

Description	Symbol	Min	Max	Unit
Output low voltage, TCK, TMS, TDI, 8mA load	VOL	0.6	V	
Output high voltage, TCK, TMS, TDI, 100 μ A load	VOH	VJTAG-0.2	VJTAG	V
Output high voltage, TCK, TMS, TDI, 8mA load	VOH	2.4	VJTAG	V
Output current, TCK, TMS, TDI	IOL, IOH	-8	+8	mA

Note: The target board must provide the VCC, VCCI, VPUMP, and VJTAG during programming. However, if there is only one ProASIC3 device on the target board, the FlashPro3 can provide the VPUMP power supply via the USB port.

DC Characteristics for FlashPro Lite

The table below shows DC characteristics for FlashPro Lite.

Table 23 · DC Characteristic for FlashPro Lite

Description	Symbol	Min	Max	Unit
Input low voltage, TDO	VIL	-0.5	0.7	V
Input high voltage, TDO	VIH	1.7	5.0	V
Input current, TDO	IIL, IIH	-10	+10	μ A
Input capacitance, TDO			40	pF
Input voltage, VDD, operating (see note)		+2.3	+3.5	V
Input voltage, VDD, power off		-1.0	+1.0	V
Input current, VDD	IVDD		500	mA
Output voltage, VPP, operating	VPP	+15.9	+16.5	V
Output voltage, VPN, operating	VPN	-13.8	-13.4	V
Output current, IPP	IPP	0	35	mA
Output current, IPN	IPN	0	-15	mA
Output low voltage, TCK, TMS, TDI, 100 μ A load	VOL	0.0	0.2	V
Output low voltage, TCK, TMS, TDI, 1mA load	VOL	0.0	0.5	V
Output low voltage, TCK, TMS, TDI, 2mA load	VOL	0.0	0.8	V
Output high voltage, TCK, TMS, TDI, 100 μ A load	VOH	2.1	2.5	V



Description	Symbol	Min	Max	Unit
Output high voltage, TCK, TMS, TDI, 1mA load	VOH	1.9	2.5	V
Output high voltage, TCK, TMS, TDI, 2mA load	VOH	1.6	2.5	V
Output current, TCK, TMS, TDI, nTRST	IOL, IOH	-2	+2	mA

Note: Up to 3.5 V can be supplied to the FlashPro Lite on the VDD pin. However, if the VDD supply for the FlashPro is also connected to the APA VDD supply, the voltage for the VDD pin cannot exceed 2.7 V.

DC Characteristics for FlashPro

The table below shows DC characteristics for FlashPro.

Table 24 · DC Characteristic for FlashPro

Description	Symbol	Min	Max	Unit
Input low voltage, TDO	VIL	-0.5	0.30 * VDDP	V
Input high voltage, TDO	VIH	0.70 * VDDP	5.5	V
Input current, TDO	IIL, IIH	-10	+10	uA
Input voltage, VDDP, VDDL		0	5.25	V
Input voltage, VPP		0	21.0	V
Input voltage, VPn		-21.0		V
Input current, VDDP, VDDL, VPn, VP	IVCC		5.0	mA
Output voltage range, VDDP	VDDP	1.5	3.3	V
Output voltage range, VPP	VPP	15.0	18.0	V
Output voltage range, VPn	VPn	-16.0	-12.0	V
Output voltage resolution / accuracy			100 / ±50	mV
Output current, IDDP	IDDP	-135(1)	+135	mA
Output current, IDDL	IDDL	-135(1)	+135	mA
Output current, IPP	IPP	-270(1)	+270	mA
Output current, IPn	IPn	-270	+270(1)	mA
Output low voltage, TCK, TMS, TDI, OUT0, nTRST	VOL	0.0	0.4	V

Description	Symbol	Min	Max	Unit
Output high voltage, TCK, TMS, TDI, OUT0, nTRST	VOH	0.85 * VDDP	+ 0.3 VDDP	V
Output current, TCK, TMS, TDI, OUT0, nTRST	IOL, IOH	-12	+12	mA

Note: When power supply mode is set to `ABI_GROUND`.

Note: If you want to power-up the device from the board power supply, clear the checkboxes for `VDDL` and `VDDP`. `VPP` and `VPN` are required during programming only and are supplied by the FlashPro programmer.



Electronic Specifications

FlashPro3

The FlashPro3 output is supplied via a connector to which a detachable 10-pin cable is fitted. The connector on the FlashPro3 unit is a 2x5, RA male Header connector, which is manufactured by AMP and has a manufacturer's part number of 103310-1. This is a standard 2x5, 0.1 pitch connector which is keyed. Use the 10 pin right-angle header, AMP P/N 103310-1 (DigiKey P/N A26285-ND) for FlashPro3 and use the 10 pin straight header, AMP P/N 103308-1 (DigiKey P/N A26267-ND) for the straight version.

The signals on the pins of the FlashPro3 10-pin connector are shown in the figure below (extracted from FlashPro3 product specification):

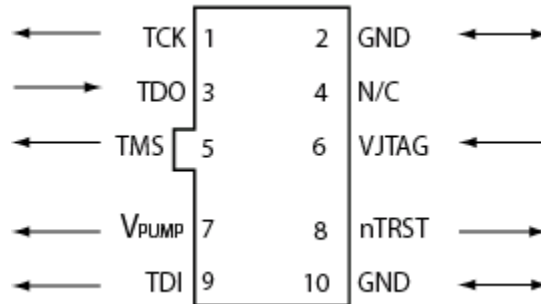


Figure 193 · FlashPro3 10-Pin Connector for FlashPro3

Note: All ground pins must be connected. The rectangular shape shows connections on the programmer itself. Arrows show current flow towards or from the rectangular programmer.

The table below shows a description of the signals.

Table 25 · FlashPro3 Signal Description

Signal	Description
VPUMP	3.3V Programming voltage
GND	Signal reference
TCK	JTAG clock
TDI	JTAG data input to device
TDO	JTAG data output from device
TMS	JTAG mode select
nTRST	Programmable output pin may be set to off, toggle, low, or high level
VJTAG	Reference voltage from the target board
N/C	Programmer does not connect to this pin

Some designers of high-integrity boards (military and avionic) may arrange their boards so that TRST is tied to ground via a weak pull-down resistor. The purpose of this is to hold the JTAG state-machine in a reset state by default, so that even with TCK oscillating, some sudden ion bombardment or other electrical event will not suddenly throw the JTAG state-machine into an unknown state. If your design also uses a weak pull-down resistor on TRST on your board, then enabling the “Drive TRST” flag will be required to force the JTAG state-machine out of reset to permit programming to take place. With most boards, there is no need to select this flag.

FlashPro Lite

For FlashPro Lite, the existing 26-pin connector is shown in the figure below.

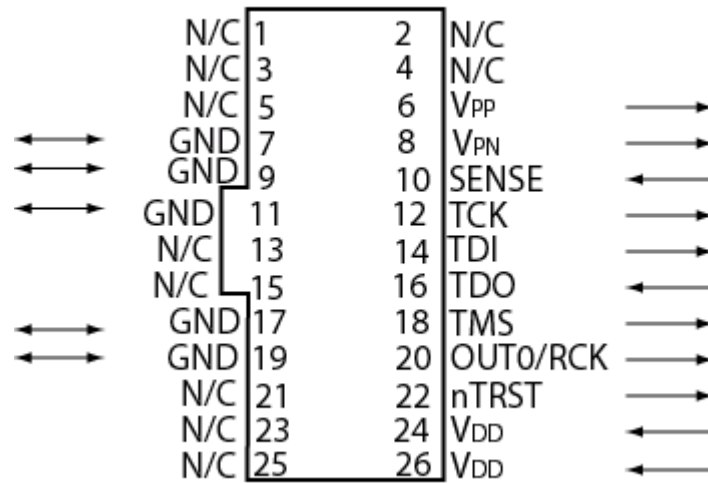


Figure 194 · 26-pin Connector for FlashPro Lite

Note: All ground pins must be connected. The rectangular shape shows connections on the programmer itself. Arrows show current flow towards or from the rectangular programmer.

The appropriate SAMTEC micro connector target cable for this is:

Samtec FFSD-13-D-12.00-01-N.

The 12 inch cable is specified. This is likely to be more than enough to connect to the board and reducing the inductance will help compared with 18 inches, which is supplied by the default with FlashPro Lite.

See the table below for a description of the signals.

Table 26 · FlashPro Lite Signal Description

Signal	Description
VDDP	VDD supply for logic I/O pads
VDDL	VDD supply for core
VPP	Positive programming supply (+16.5V)
VPN	Negative programming supply(-13.8V)
GND	Signal reference
SENSE	Input from target board to programmer to indicate connection to ground
TCK	JTAG clock
TDI	JTAG data input to device
TDO	JTAG data output from device
TMS	JTAG mode select
nTRST	Programmable output pin may be set to off, toggle, low, or high level
RCK/OUT0	Programmable output pin may be set to off, toggle, low, or high level
N/C	Programmer does not connect to this pin

Some designers of high-integrity boards (military and avionic) may arrange their boards so that TRST is tied to ground via a weak pull-down resistor. The purpose of this is to hold the JTAG state-machine in a reset state by default, so that even with TCK oscillating, some sudden ion bombardment or other electrical event will not suddenly throw the JTAG state-machine into an unknown state. If your design also uses a weak pull-down resistor on TRST on your board, then enabling the "Drive TRST" flag will be required to force the JTAG state-machine out of reset to permit programming to take place. With most boards, there is no need to select this flag.



FlashPro

For FlashPro, you can use the same 26-pin target cable you used for FlashPro Lite, but the connections are shown in the figure below.

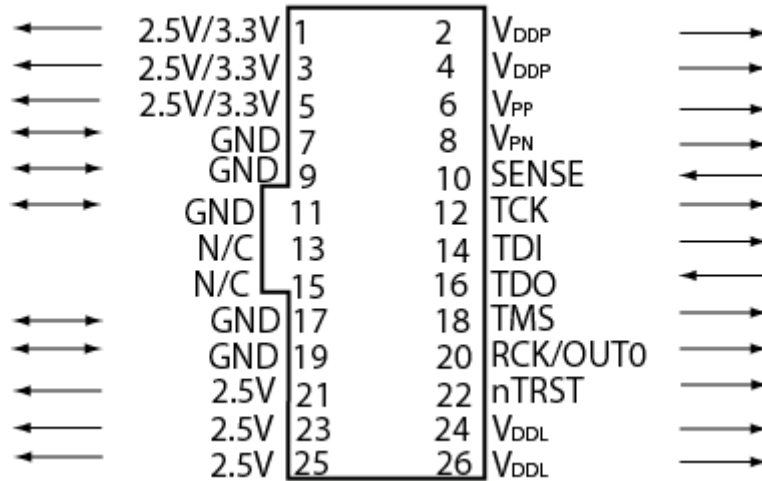


Figure 195 · 26-pin connections for FlashPro

Note: All ground pins must be connected. The rectangular shape shows connections on the programmer itself. Arrows show current flow towards or from the rectangular programmer.

The table below shows the signal pin descriptions for FlashPro.

Table 27 · FlashPro Signal Description

Signal	Description
VDDP	VDD supply for logic I/O pads
VDDL	VDD supply for core
VPP	Positive programming supply (+16.5 V)
VPN	Negative programming supply (-13.8 V)
GND	Signal reference
SENSE	Input from target board to programmer to indicate connection to ground
TCK	JTAG clock
TDI	JTAG data input to device
TDO	JTAG data output from device
TMS	JTAG mode select
nTRST	Programmable output pin may be set to off, toggle, low, or high level
RCK/OUT0	Programmable output pin may be set to off, toggle, low, or high level
2.5V, 2.5V/3.3V, N/C	Programmer does not connect to these pins

Some designers of high-integrity boards (military and avionic) may arrange their boards so that TRST is tied to ground via a weak pull-down resistor. The purpose of this is to hold the JTAG state-machine in a reset state by default, so that even with TCK oscillating, some sudden ion bombardment or other electrical event will not suddenly throw the JTAG state-machine into an unknown state. If your design also uses a weak pull-down resistor on TRST on your board, then enabling the "Drive TRST" flag will be required to force the JTAG state-machine out of reset to permit programming to take place. With most boards, there is no need to select this flag.



FlashPro3 Characteristics

The table below shows the JTAG switching characteristics for FlashPro3.

Table 28 · JTAG Switching Characteristics for FlashPro3

Description	Symbol	Min	Max	Unit
Output delay from TCK to TDI, TMS	TTCKTDI	-2	2	ns
TDO setup time before TCK rising, VJTAG=3.3	TTDOTCK	12		ns
TDO setup time before TCK rising, VJTAG=1.5	TTDOTCK	14.5		ns
TDO hold time after TCK rising	TTCKTDO	0		ns
TCK period	TTCK	41.7	10667	ns

FlashPro and FlashPro Lite Characteristics

The table below shows the JTAG switching characteristics for FlashPro and FlashPro Lite measured at the programmer end of the JTAG cable.

Table 29 · JTAG Switching Characteristics for FlashPro and FlashPro Lite

Description	Symbol	Min	Max	Unit
Output delay from TCK falling to TDI, TMS	TTCKTDI	-2	2	ns
TDO setup time before TCK rising	TTDOTCK	5.0		ns
TDO hold time after TCK rising		0		ns
TCK period	TTCK	40	10240	ns

Illustration of the JTAG Switching Characteristics

The figure below is an illustration of the JTAG switching characteristics.

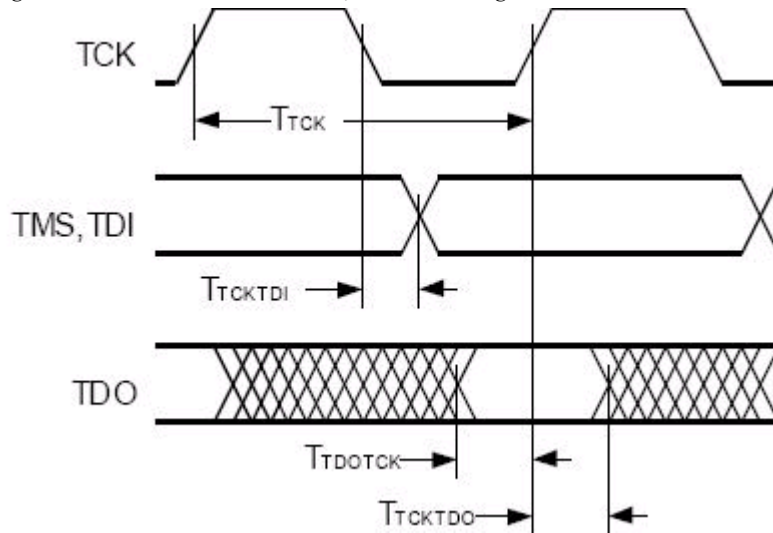


Figure 196 · JTAG Switching Characteristics

Actel Headquarters

Actel Corporation is a supplier of innovative programmable logic solutions, including field-programmable gate arrays (FPGAs) based on Antifuse and Flash technologies, high-performance intellectual property (IP) cores, software development tools, and design services targeted for the high-speed communications, application-specific integrated circuit (ASIC) replacement, and radiation-tolerant markets.

Address:	Actel Corporation 2061 Stierlin Court Mountain View, CA 94043-4655 USA
Phone:	(650) 318-4200 (650) 318-4600

See Also

- [Customer service](#)
- [Technical support](#)
- [Sales](#)

Technical Support

Highly skilled engineers staff the Technical Support Center from 7:00 AM to 6:00 PM Pacific Time, Monday through Friday.



Visit Tech Support Online

For 24-hour support resources, visit Actel Technical Support at <http://www.actel.com/support/search/advance/>.

Contacting Technical Support

Contact us with your technical questions via e-mail or by phone. When sending your request to us, please be sure to include your full name, company name, email address, and telephone number.

E-mail (Worldwide):	tech@actel.com
Telephone (In U.S.):	(650) 318-4460 (800) 262-1060
Telephone (Outside the US):	Contact a local sales office

See Also

[Actel headquarters](#)

[Customer service](#)

[Sales](#)

[Documentation feedback](#)

Customer Service

Contact Customer Service for order status, order expedites, return material authorizations (RMA), and first article processing. For technical issues, contact [Technical Support](#).

From	Call
Northeast and North Central U.S.A.	(650) 318-4480
Southeast and Southwest U.S.A.	(650) 318-4480
South Central U.S.A.	(650) 318-4434
Northwest U.S.A.	(650) 318-4434
Canada	(650) 318-4480
Europe	(650) 318-4252 or +44 (0) 1276 401500
Japan	(650) 318-4743
From the rest of the world	(650) 318-4743

Product Support

Actel backs its products with various support services including Customer Service, a Customer Technical Support Center, a web site, an FTP site, electronic mail, and worldwide sales offices. This appendix contains information about contacting Actel and using these support services.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From Northeast and North Central U.S.A., call 650.318.4480

From Southeast and Southwest U.S.A., call 650.318.4480

From South Central U.S.A., call 650.318.4434

From Northwest U.S.A., call 650.318.4434

From Canada, call 650.318.4480

From Europe, call 650.318.4252 or +44 (0) 1276 401 500

From Japan, call 650.318.4743

From the rest of the world, call 650.318.4743

Fax, from anywhere in the world 650.318.8044

Actel Customer Technical Support Center

Actel staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions. The Customer Technical Support Center spends a great deal of time creating application notes and answers to FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Actel Technical Support

Visit the [Actel Customer Support website \(http://www.actel.com/custsup/search.html\)](http://www.actel.com/custsup/search.html) for more information and support. Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on the Actel web site.

Website

You can browse a variety of technical and non-technical information on Actel's [home page](http://www.actel.com/), at <http://www.actel.com/>.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center from 7:00 A.M. to 6:00 P.M., Pacific Time, Monday through Friday. Several ways of contacting the Center follow:

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is tech@actel.com.

Phone

Our Technical Support Center answers all calls. The center retrieves information, such as your name, company name, phone number and your question, and then issues a case number. The Center then forwards the information to a queue where the first available application engineer receives the data and returns your call. The phone hours are from 7:00 A.M. to 6:00 P.M., Pacific Time, Monday through Friday. The Technical Support numbers are:

650.318.4460

800.262.1060

Customers needing assistance outside the US time zones can either contact technical support via email (tech@actel.com) or contact a local sales office. [Sales office listings](#) can be found at www.actel.com/contact/offices/index.html.





For more information about Actel's products, visit our website at
www.actel.com

Actel Corporation • 2061 Stierlin Court • Mountain View, CA 94043 • USA

Phone 650.318.4200 • Fax 650.318.4600 • Customer Service: 650.318.1010 • Customer Applications Center: 800.262.1060

Actel Europe Ltd. • River Court, Meadows Business Park • Station Approach, Blackwater • Camberley Surrey GU17 9AB • United Kingdom

Phone +44 (0) 1276 609 300 • Fax +44 (0) 1276 607 540

Actel Japan • EXOS Ebisu Building 4F • 1-24-14 Ebisu Shibuya-ku • Tokyo 150 • Japan

Phone +81.03.3445.7671 • Fax +81.03.3445.7668 • <http://jap.actel.com>

Actel Hong Kong • Room 2107, China Resources Building • 26 Harbour Road • Wanchai • Hong Kong

Phone +852 2185 6460 • Fax +852 2185 6488 • www.actel.com.cn

5029138-13/7.08

