



Welcome to [E-XFL.COM](https://www.e-xfl.com)

### Understanding [Embedded - FPGAs \(Field Programmable Gate Array\)](#)

Embedded - FPGAs, or Field Programmable Gate Arrays, are advanced integrated circuits that offer unparalleled flexibility and performance for digital systems. Unlike traditional fixed-function logic devices, FPGAs can be programmed and reprogrammed to execute a wide array of logical operations, enabling customized functionality tailored to specific applications. This reprogrammability allows developers to iterate designs quickly and implement complex functions without the need for custom hardware.

### Applications of Embedded - FPGAs

The versatility of Embedded - FPGAs makes them indispensable in numerous fields. In telecommunications.

#### Details

Product Status	Obsolete
Number of LABs/CLBs	-
Number of Logic Elements/Cells	-
Total RAM Bits	147456
Number of I/O	300
Number of Gates	1000000
Voltage - Supply	1.14V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	0°C ~ 85°C (TJ)
Package / Case	484-BGA
Supplier Device Package	484-FPBGA (23x23)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microsemi/a3p1000l-1fg484">https://www.e-xfl.com/product-detail/microsemi/a3p1000l-1fg484</a>

Table 2-4 summarizes the Flash\*Freeze mode implementations.

**Table 2-4 • Flash\*Freeze Mode Usage**

Flash*Freeze Mode Type	Description	Flash*Freeze Pin State	Instantiate ULSICC Macro	LSICC Signal	Operating Mode
1	Flash*Freeze mode is controlled only by the FF pin.	Deasserted	No	N/A	Normal operation
		Asserted	No	N/A	Flash*Freeze mode
2	Flash*Freeze mode is controlled by the FF pin and LSICC signal.	"Don't care"	Yes	Deasserted	Normal operation
		Deasserted	Yes	"Don't care"	Normal operation
		Asserted	Yes	Asserted	Flash*Freeze mode

*Note:* Refer to Table 2-3 on page 26 for Flash\*Freeze pin and LSICC signal assertion and deassertion values.

## IGLOO, ProASIC3L, and RT ProASIC3 I/O State in Flash\*Freeze Mode

In IGLOO and ProASIC3L devices, when the device enters Flash\*Freeze mode, I/Os become tristated. If the weak pull-up or pull-down feature is used, the I/Os will maintain the configured weak pull-up or pull-down status. This feature enables the design to set the I/O state to a certain level that is determined by the pull-up/-down configuration.

Table 2-5 shows the I/O pad state based on the configuration and buffer type.

Note that configuring weak pull-up or pull-down for the FF pin is not allowed. The FF pin can be configured as a Schmitt trigger input in IGLOOe, IGLOO nano, IGLOO PLUS, and ProASIC3EL devices.

**Table 2-5 • IGLOO, ProASIC3L, and RT ProASIC3 Flash\*Freeze Mode (type 1 and type 2)—I/O Pad State**

Buffer Type		I/O Pad Weak Pull-Up/-Down	I/O Pad State in Flash*Freeze Mode
Input/Global		Enabled	Weak pull-up/pull-down*
		Disabled	Tristate*
Output		Enabled	Weak pull-up/pull-down
		Disabled	Tristate
Bidirectional / Tristate Buffer	E = 0 (input/tristate)	Enabled	Weak pull-up/pull-down*
		Disabled	Tristate*
	E = 1 (output)	Enabled	Weak pull-up/pull-down
		Disabled	Tristate

\* Internal core logic driven by this input/global buffer will be tied High as long as the device is in Flash\*Freeze mode.

## Chip and Quadrant Global I/Os

The following sections give an overview of naming conventions and other related I/O information.

### Naming of Global I/Os

In low power flash devices, the global I/Os have access to certain clock conditioning circuitry and have direct access to the global network. Additionally, the global I/Os can be used as regular I/Os, since they have identical capabilities to those of regular I/Os. Due to the comprehensive and flexible nature of the I/Os in low power flash devices, a naming scheme is used to show the details of the I/O. The global I/O uses the generic name Gmn/IOuxwByVz. Note that Gmn refers to a global input pin and IOuxwByVz refers to a regular I/O Pin, as these I/Os can be used as either global or regular I/Os. Refer to the I/O Structures chapter of the user's guide for the device that you are using for more information on this naming convention.

Figure 3-4 represents the global input pins connection. It shows all 54 global pins available to access the 18 global networks in ProASIC3E families.

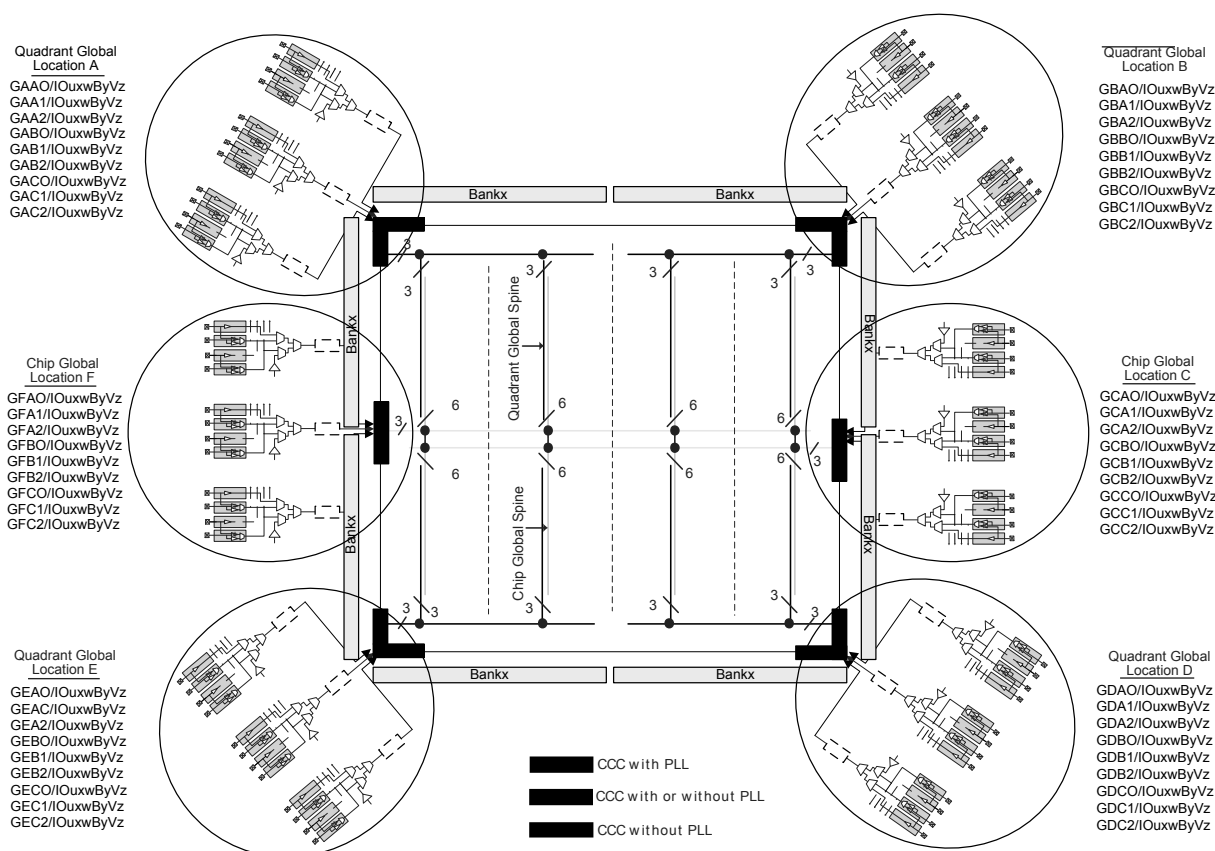
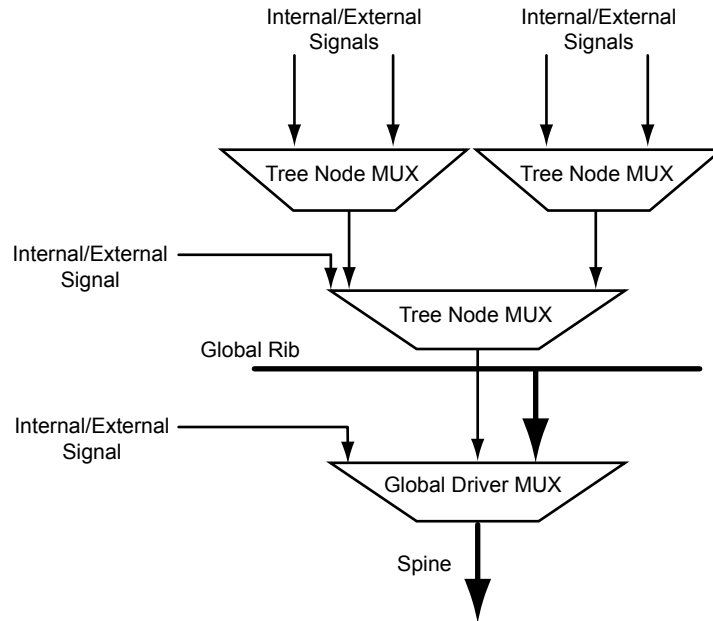


Figure 3-4 • Global Connections Details

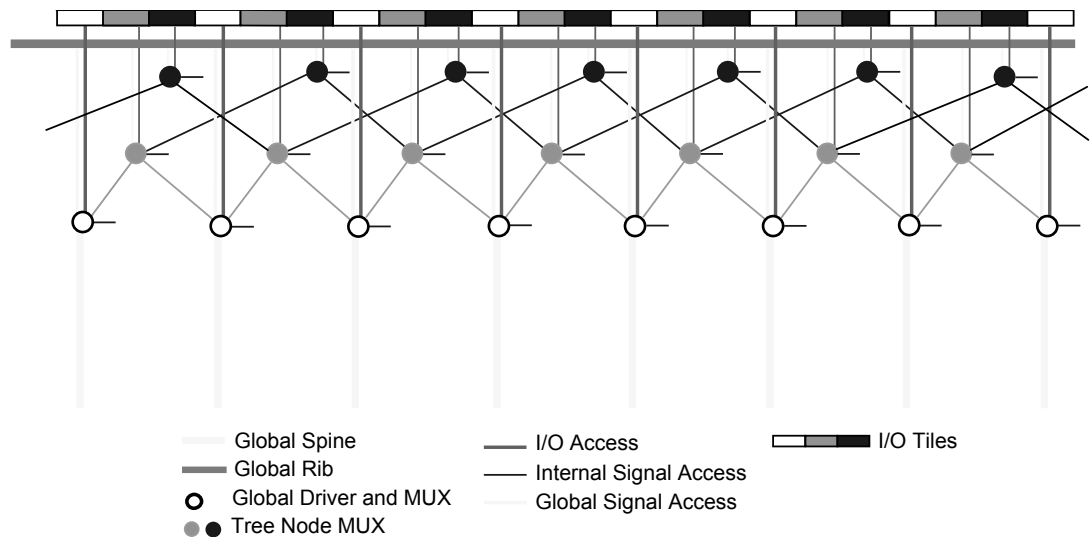
## Using Clock Aggregation

Clock aggregation allows for multi-spine clock domains to be assigned using hardwired connections, without adding any extra skew. A MUX tree, shown in Figure 3-8, provides the necessary flexibility to allow long lines, local resources, or I/Os to access domains of one, two, or four global spines. Signal access to the clock aggregation system is achieved through long-line resources in the central rib in the center of the die, and also through local resources in the north and south ribs, allowing I/Os to feed directly into the clock system. As Figure 3-9 indicates, this access system is contiguous.

There is no break in the middle of the chip for the north and south I/O VersaNet access. This is different from the quadrant clocks located in these ribs, which only reach the middle of the rib.



**Figure 3-8 • Spine Selection MUX of Global Tree**



**Figure 3-9 • Clock Aggregation Tree Architecture**



## Using Spines of Occupied Global Networks

When a signal is assigned to a global network, the flash switches are programmed to set the MUX select lines (explained in the "Clock Aggregation Architecture" section on page 61) to drive the spines of that network with the global net. However, if the global net is restricted from reaching into the scope of a spine, the MUX drivers of that spine are available for other high-fanout or critical signals (Figure 3-20).

For example, if you want to limit the CLK1\_c signal to the left half of the chip and want to use the right side of the same global network for CLK2\_c, you can add the following PDC commands:

```
define_region -name region1 -type inclusive 0 0 34 29  
assign_net_macros region1 CLK1_c  
assign_local_clock -net CLK2_c -type chip B2
```

---

---

**Figure 3-20 • Design Example Using Spines of Occupied Global Networks**

## Conclusion

IGLOO, Fusion, and ProASIC3 devices contain 18 global networks: 6 chip global networks and 12 quadrant global networks. These global networks can be segmented into local low-skew networks called spines. The spines provide low-skew networks for the high-fanout signals of a design. These allow you up to 252 different internal/external clocks in an A3PE3000 device. This document describes the architecture for the global network, plus guidelines and methodologies in assigning signals to globals and spines.

## Related Documents

### User's Guides

*IGLOO, ProASIC3, SmartFusion, and Fusion Macro Library Guide*  
[http://www.microsemi.com/soc/documents/pa3\\_libguide\\_ug.pdf](http://www.microsemi.com/soc/documents/pa3_libguide_ug.pdf)

Date	Changes	Page
v1.1 (March 2008)	The "Global Architecture" section was updated to include the IGLOO PLUS family. The bullet was revised to include that the west CCC does not contain a PLL core in 15 k and 30 k devices. Instances of "A3P030 and AGL030 devices" were replaced with "15 k and 30 k gate devices."	47
v1.1 (continued)	Table 3-1 • Flash-Based FPGAs and the accompanying text was updated to include the IGLOO PLUS family. The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	48
	The "VersaNet Global Network Distribution" section, "Spine Architecture" section, the note in Figure 3-1 • Overview of VersaNet Global Network and Device Architecture, and the note in Figure 3-3 • Simplified VersaNet Global Network (60 k gates and above) were updated to include mention of 15 k gate devices.	49, 50
	Table 3-4 • Globals/Spines/Rows for IGLOO and ProASIC3 Devices was updated to add the A3P015 device, and to revise the values for clock trees, globals/spines per tree, and globals/spines per device for the A3P030 and AGL030 devices.	57
	Table 3-5 • Globals/Spines/Rows for IGLOO PLUS Devices is new.	58
	CLKBUF_LVCMOS12 was added to Table 3-9 • I/O Standards within CLKBUF.	63
	The "User's Guides" section was updated to include the three different I/O Structures chapters for ProASIC3 and IGLOO device families.	74
v1.0 (January 2008)	Figure 3-3 • Simplified VersaNet Global Network (60 k gates and above) was updated.	50
	The "Naming of Global I/Os" section was updated.	51
	The "Using Global Macros in Synplicity" section was updated.	66
	The "Global Promotion and Demotion Using PDC" section was updated.	67
	The "Designer Flow for Global Assignment" section was updated.	69
	The "Simple Design Example" section was updated.	71
51900087-0/1.05 (January 2005)	Table 3-4 • Globals/Spines/Rows for IGLOO and ProASIC3 Devices was updated.	57

## PLL Core Specifications

PLL core specifications can be found in the DC and Switching Characteristics chapter of the appropriate family datasheet.

### Loop Bandwidth

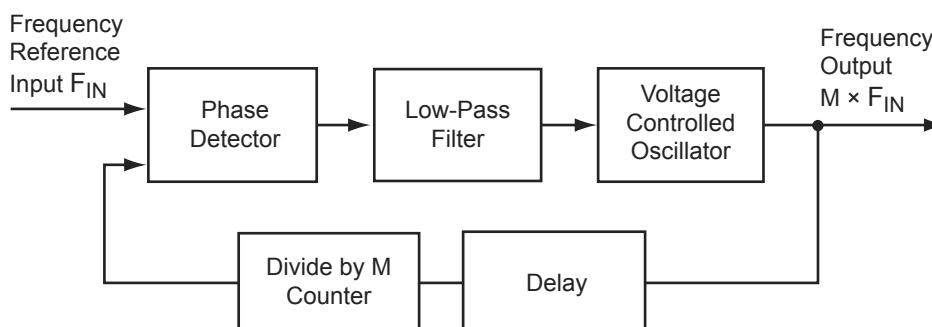
Common design practice for systems with a low-noise input clock is to have PLLs with small loop bandwidths to reduce the effects of noise sources at the output. Table 4-6 shows the PLL loop bandwidth, providing a measure of the PLL's ability to track the input clock and jitter.

**Table 4-6 • -3 dB Frequency of the PLL**

	Minimum ( $T_a = +125^\circ\text{C}$ , $V_{CCA} = 1.4\text{ V}$ )	Typical ( $T_a = +25^\circ\text{C}$ , $V_{CCA} = 1.5\text{ V}$ )	Maximum ( $T_a = -55^\circ\text{C}$ , $V_{CCA} = 1.6\text{ V}$ )
<b>-3 dB Frequency</b>	15 kHz	25 kHz	45 kHz

### PLL Core Operating Principles

This section briefly describes the basic principles of PLL operation. The PLL core is composed of a phase detector (PD), a low-pass filter (LPF), and a four-phase voltage-controlled oscillator (VCO). Figure 4-19 illustrates a basic single-phase PLL core with a divider and delay in the feedback path.



**Figure 4-19 • Simplified PLL Core with Feedback Divider and Delay**

The PLL is an electronic servo loop that phase-aligns the PD feedback signal with the reference input. To achieve this, the PLL dynamically adjusts the VCO output signal according to the average phase difference between the input and feedback signals.

The first element is the PD, which produces a voltage proportional to the phase difference between its inputs. A simple example of a digital phase detector is an Exclusive-OR gate. The second element, the LPF, extracts the average voltage from the phase detector and applies it to the VCO. This applied voltage alters the resonant frequency of the VCO, thus adjusting its output frequency.

Consider Figure 4-19 with the feedback path bypassing the divider and delay elements. If the LPF steadily applies a voltage to the VCO such that the output frequency is identical to the input frequency, this steady-state condition is known as lock. Note that the input and output phases are also identical. The PLL core sets a LOCK output signal HIGH to indicate this condition.

Should the input frequency increase slightly, the PD detects the frequency/phase difference between its reference and feedback input signals. Since the PD output is proportional to the phase difference, the change causes the output from the LPF to increase. This voltage change increases the resonant frequency of the VCO and increases the feedback frequency as a result. The PLL dynamically adjusts in this manner until the PD senses two phase-identical signals and steady-state lock is achieved. The opposite (decreasing PD output signal) occurs when the input frequency decreases.

Now suppose the feedback divider is inserted in the feedback path. As the division factor  $M$  (shown in Figure 4-20 on page 101) is increased, the average phase difference increases. The average phase

```
DLYGLC[4:0]    00000
DLYYB[4:0]    00000
DLYYC[4:0]    00000
VCOSEL[2:0]    100
```

```
Primary Clock Frequency 33.000
Primary Clock Phase Shift 0.000
Primary Clock Output Delay from CLKA 1.695
```

```
Secondary1 Clock Frequency 40.000
Secondary1 Clock Phase Shift 0.000
Secondary1 Clock Global Output Delay from CLKB 0.200
```

```
Secondary2 Clock Frequency 50.000
Secondary2 Clock Phase Shift 0.000
Secondary2 Clock Global Output Delay from CLKC 0.200
```

```
#####
# Dynamic Stream Data
#####
```

NAME	SDIN	VALUE	TYPE
FINDIV	[6:0]	0000101	EDIT
FBDIV	[13:7]	0100000	EDIT
OADIV	[18:14]	00100	EDIT
OBDIV	[23:19]	00000	EDIT
OCDIV	[28:24]	00000	EDIT
OAMUX	[31:29]	100	EDIT
OBMUX	[34:32]	000	EDIT
OCMUX	[37:35]	000	EDIT
FBSEL	[39:38]	01	EDIT
FBDLY	[44:40]	00000	EDIT
XDLYSEL	[45]	0	EDIT
DLYGLA	[50:46]	00000	EDIT
DLYGLB	[55:51]	00000	EDIT
DLYGLC	[60:56]	00000	EDIT
DLYYB	[65:61]	00000	EDIT
DLYYC	[70:66]	00000	EDIT
STATASEL	[71]	X	MASKED
STATBSEL	[72]	X	MASKED
STATCSEL	[73]	X	MASKED
VCOSEL	[76:74]	100	EDIT
DYNASEL	[77]	X	MASKED
DYNBSEL	[78]	X	MASKED
DYNCSEL	[79]	X	MASKED
RESETEN	[80]	1	READONLY

Below is the resultant Verilog HDL description of a legal dynamic PLL core configuration generated by SmartGen:

```
module dyn_pll_macro(POWERDOWN, CLKA, LOCK, GLA, GLB, GLC, SDIN, SCLK, SSHIFT, SUPDATE,
    MODE, SDOUT, CLKB, CLKC);

input POWERDOWN, CLKA;
output LOCK, GLA, GLB, GLC;
input SDIN, SCLK, SSHIFT, SUPDATE, MODE;
output SDOUT;
input CLKB, CLKC;

wire VCC, GND;

VCC VCC_1_net(.Y(VCC));
GND GND_1_net(.Y(GND));
```

## Simulation of FlashROM Design

The MEM file has 128 rows of 8 bits, each representing the contents of the FlashROM used for simulation. For example, the first row represents page 0, byte 0; the next row is page 0, byte 1; and so the pattern continues. Note that the three MSBs of the address define the page number, and the four LSBs define the byte number. So, if you send address 0000100 to FlashROM, this corresponds to the page 0 and byte 4 location, which is the fifth row in the MEM file. SmartGen defaults to 0s for any unspecified location of the FlashROM. Besides using the MEM file generated by SmartGen, you can create a binary file with 128 rows of 8 bits each and use this as a MEM file. Microsemi recommends that you use different file names if you plan to generate multiple MEM files. During simulation, Libero SoC passes the MEM file used as the generic file in the netlist, along with the design files and testbench. If you want to use different MEM files during simulation, you need to modify the generic file reference in the netlist.

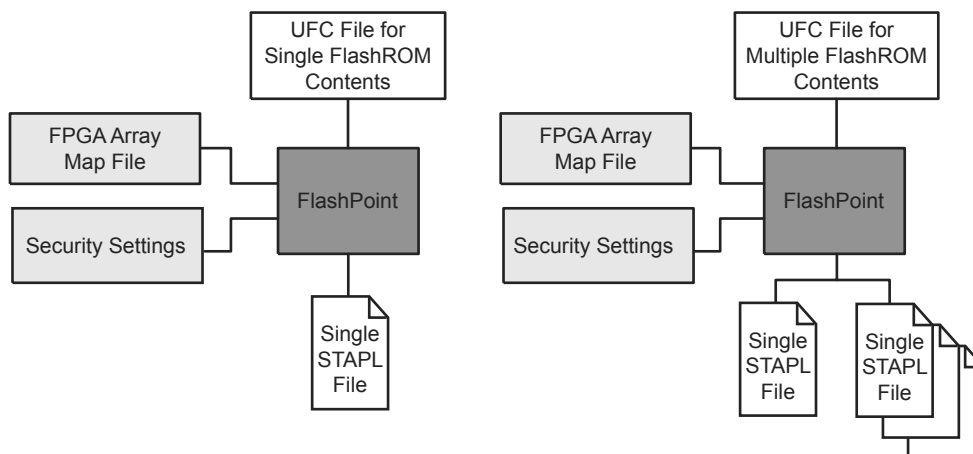
```
.....
UFROM0: UFROM
--generic map(MEMORYFILE => "F:\Appsnotes\FROM\test_designs\testa\smartgen\FROM_a.mem")
--generic map(MEMORYFILE => "F:\Appsnotes\FROM\test_designs\testa\smartgen\FROM_b.mem")
.....
```

The VITAL and Verilog simulation models accept the generics passed by the netlist, read the MEM file, and perform simulation with the data in the file.

## Programming File Generation for FlashROM Design

FlashPoint is the programming software used to generate the programming files for flash devices. Depending on the applications, you can use the FlashPoint software to generate a STAPL file with different FlashROM contents. In each case, optional AES decryption is available. To generate a STAPL file that contains the same FPGA core content and different FlashROM contents, the FlashPoint software needs an Array Map file for the core and UFC file(s) for the FlashROM. This final STAPL file represents the combination of the logic of the FPGA core and FlashROM content.

FlashPoint generates the STAPL files you can use to program the desired FlashROM page and/or FPGA core of the FPGA device contents. FlashPoint supports the encryption of the FlashROM content and/or FPGA Array configuration data. In the case of using the FlashROM for device serialization, a sequence of unique FlashROM contents will be generated. When generating a programming file with multiple unique FlashROM contents, you can specify in FlashPoint whether to include all FlashROM content in a single STAPL file or generate a different STAPL file for each FlashROM (Figure 5-11). The programming software (FlashPro) handles the single STAPL file that contains the FlashROM content from multiple devices. It enables you to program the FlashROM content into a series of devices sequentially (Figure 5-11). See the *FlashPro User's Guide* for information on serial programming.



**Figure 5-11 • Single or Multiple Programming File Generation**

DEVICE\_INFO displays the FlashROM content, serial number, Design Name, and checksum, as shown below:

```
EXPORT IDCODE[32] = 123261CF
EXPORT SILSIG[32] = 00000000
User information :
CHECKSUM: 61A0
Design Name:      TOP
Programming Method: STAPL
Algorithm Version: 1
Programmer: UNKNOWN
=====
FlashROM Information :
EXPORT Region_7_0[128] = FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
=====
Security Setting :
Encrypted FlashROM Programming Enabled.
Encrypted FPGA Array Programming Enabled.
=====
```

The Libero SoC file manager recognizes the UFC and MEM files and displays them in the appropriate view. Libero SoC also recognizes the multiple programming files if you choose the option to generate multiple files for multiple FlashROM contents in Designer. These features enable a user-friendly flow for the FlashROM generation and programming in Libero SoC.

## Custom Serialization Using FlashROM

You can use FlashROM for device serialization or inventory control by using the Auto Inc region or Read From File region. FlashPoint will automatically generate the serial number sequence for the Auto Inc region with the **Start Value**, **Max Value**, and **Step Value** provided. If you have a unique serial number generation scheme that you prefer, the Read From File region allows you to import the file with your serial number scheme programmed into the region. See the *FlashPro User's Guide* for custom serialization file format information.

The following steps describe how to perform device serialization or inventory control using FlashROM:

1. Generate FlashROM using SmartGen. From the Properties section in the FlashROM Settings dialog box, select the **Auto Inc** or **Read From File** region. For the Auto Inc region, specify the desired step value. You will not be able to modify this value in the FlashPoint software.
2. Go through the regular design flow and finish place-and-route.
3. Select **Programming File in Designer** and open **Generate Programming File** (Figure 5-12 on page 144).
4. Click **Program FlashROM**, browse to the UFC file, and click **Next**. The FlashROM Settings window appears, as shown in Figure 5-13 on page 144.
5. Select the FlashROM page you want to program and the data value for the configured regions. The STAPL file generated will contain only the data that targets the selected FlashROM page.
6. Modify properties for the serialization.
  - 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.
7. Select the FlashROM programming file type you want to generate from the two options below:
  - Single programming file for all devices: generates one programming file with all FlashROM values.
  - One programming file per device: generates a separate programming file for each FlashROM value.
8. Enter the number of devices you want to program and generate the required programming file.
9. Open the programming software and load the programming file. The programming software, FlashPro3 and Silicon Sculptor II, supports the device serialization feature. If, for some reason, the device fails to program a part during serialization, the software allows you to reuse or skip the serial data. Refer to the *FlashPro User's Guide* for details.

## SRAM Usage

The following descriptions refer to the usage of both RAM4K9 and RAM512X18.

### Clocking

The dual-port SRAM blocks are only clocked on the rising edge. SmartGen allows falling-edge-triggered clocks by adding inverters to the netlist, hence achieving dual-port SRAM blocks that are clocked on either edge (rising or falling). For dual-port SRAM, each port can be clocked on either edge and by separate clocks by port. Note that for Automotive ProASIC3, the same clock, with an inversion between the two clock pins of the macro, should be used in design to prevent errors during compile.

Low power flash devices support inversion (bubble-pushing) throughout the FPGA architecture, including the clock input to the SRAM modules. Inversions added to the SRAM clock pin on the design schematic or in the HDL code will be automatically accounted for during design compile without incurring additional delay in the clock path.

The two-port SRAM can be clocked on the rising or falling edge of WCLK and RCLK.

If negative-edge RAM and FIFO clocking is selected for memory macros, clock edge inversion management (bubble-pushing) is automatically used within the development tools, without performance penalty.

### Modes of Operation

There are two read modes and one write mode:

- Read Nonpipelined (synchronous—1 clock edge): In the standard read mode, new data is driven onto the RD bus in the same clock cycle following RA and REN valid. The read address is registered on the read port clock active edge, and data appears at RD after the RAM access time. Setting PIPE to OFF enables this mode.
- Read Pipelined (synchronous—2 clock edges): The pipelined mode incurs an additional clock delay from address to data but enables operation at a much higher frequency. The read address is registered on the read port active clock edge, and the read data is registered and appears at RD after the second read clock edge. Setting PIPE to ON enables this mode.
- Write (synchronous—1 clock edge): On the write clock active edge, the write data is written into the SRAM at the write address when WEN is HIGH. The setup times of the write address, write enables, and write data are minimal with respect to the write clock.

### RAM Initialization

Each SRAM block can be individually initialized on power-up by means of the JTAG port using the UJTAG mechanism. The shift register for a target block can be selected and loaded with the proper bit configuration to enable serial loading. The 4,608 bits of data can be loaded in a single operation.

## FIFO Features

The FIFO4KX18 macro is created by merging the RAM block with dedicated FIFO logic (Figure 6-6 on page 158). Since the FIFO logic can only be used in conjunction with the memory block, there is no separate FIFO controller macro. As with the RAM blocks, the FIFO4KX18 nomenclature does not refer to a possible aspect ratio, but rather to the deepest possible data depth and the widest possible data width. FIFO4KX18 can be configured into the following aspect ratios: 4,096×1, 2,048×2, 1,024×4, 512×9, and 256×18. In addition to being fully synchronous, the FIFO4KX18 also has the following features:

- Four FIFO flags: Empty, Full, Almost-Empty, and Almost-Full
- Empty flag is synchronized to the read clock
- Full flag is synchronized to the write clock
- Both Almost-Empty and Almost-Full flags have programmable thresholds
- Active-low asynchronous reset
- Active-low block enable
- Active-low write enable
- Active-high read enable
- Ability to configure the FIFO to either stop counting after the empty or full states are reached or to allow the FIFO counters to continue

- In Active and Static modes:
  - Input buffers with pull-up, driven Low
  - Input buffers with pull-down, driven High
  - Bidirectional buffers with pull-up, driven Low
  - Bidirectional buffers with pull-down, driven High
  - Output buffers with pull-up, driven Low
  - Output buffers with pull-down, driven High
  - Tristate buffers with pull-up, driven Low
  - Tristate buffers with pull-down, driven High
- In Flash\*Freeze mode:
  - Input buffers with pull-up, driven Low
  - Input buffers with pull-down, driven High
  - Bidirectional buffers with pull-up, driven Low
  - Bidirectional buffers with pull-down, driven High

## Electrostatic Discharge Protection

Low power flash devices are tested per JEDEC Standard JESD22-A114-B.

These devices contain clamp diodes at every I/O, global, and power pad. Clamp diodes protect all device pads against damage from ESD as well as from excessive voltage transients.

All IGLOO and ProASIC3 devices are tested to the Human Body Model (HBM) and the Charged Device Model (CDM).

Each I/O has two clamp diodes. One diode has its positive (P) side connected to the pad and its negative (N) side connected to VCCI. The second diode has its P side connected to GND and its N side connected to the pad. During operation, these diodes are normally biased in the off state, except when transient voltage is significantly above VCCI or below GND levels.

In 30K gate devices, the first diode is always off. In other devices, the clamp diode is always on and cannot be switched off.

By selecting the appropriate I/O configuration, the diode is turned on or off. Refer to Table 7-12 on page 193 for more information about the I/O standards and the clamp diode.

The second diode is always connected to the pad, regardless of the I/O configuration selected.



**Table 8-3 • VCCI Voltages and Compatible IGLOOe and ProASIC3E Standards**

VCCI and VMV (typical)	Compatible Standards
3.3 V	LVTTL/LVCMOS 3.3, PCI 3.3, SSTL3 (Class I and II), GTL+ 3.3, GTL 3.3, LVPECL
2.5 V	LVC MOS 2.5, LVC MOS 2.5/5.0, SSTL2 (Class I and II), GTL+ 2.5, GTL 2.5, LVDS, DDR LVDS, B-LVDS, and M-LVDS
1.8 V	LVC MOS 1.8
1.5 V	LVC MOS 1.5, HSTL (Class I and II)
1.2 V	LVC MOS 1.2

**Table 8-4 • VREF Voltages and Compatible IGLOOe and ProASIC3E Standards**

VREF (typical)	Compatible Standards
1.5 V	SSTL3 (Class I and II)
1.25 V	SSTL2 (Class I and II)
1.0 V	GTL+ 2.5, GTL+ 3.3
0.8 V	GTL 2.5, GTL 3.3
0.75 V	HSTL (Class I and II)

**Table 8-5 • Legal IGLOOe and ProASIC3E I/O Usage Matrix within the Same Bank**

I/O Bank Voltage (typical)	Minibank Voltage (typical)	LVTTL/LVCMOS 3.3 V	LVC MOS 2.5 V	LVC MOS 1.8 V	LVC MOS 1.5 V	3.3 V PCI/PCI-X	GTL+ (3.3 V)	GTL+ (2.5 V)	GTL (3.3 V)	GTL (2.5 V)	HSTL Class I and II (1.5 V)	SSTL2 Class I and II (2.5 V)	SSTL3 Class I and II (3.3 V)	LVDS, B-LVDS, and M-LVDS, DDR (2.5 V ± 5%)	LVPECL (3.3 V)
3.3 V	–														
	0.80 V														
	1.00 V														
	1.50 V														
2.5 V	–														
	0.80 V														
	1.00 V														
	1.25 V														
1.8 V	–														
1.5 V	–														
	0.75 V														

Note: White box: Allowable I/O standard combination  
Gray box: Illegal I/O standard combination

Date	Changes	Page
v1.3 (October 2008)	The "Low Power Flash Device I/O Support" section was revised to include new families and make the information more concise.	214
v1.2 (June 2008)	<p>The following changes were made to the family descriptions in Table 8-1 · Flash-Based FPGAs:</p> <ul style="list-style-type: none"> <li>• ProASIC3L was updated to include 1.5 V.</li> <li>• The number of PLLs for ProASIC3E was changed from five to six.</li> </ul>	214
v1.1 (March 2008)	<p>This document was previously part of <i>I/O Structures in IGLOO and ProASIC3 Devices</i>. To provide information specific to IGLOOe, ProASIC3E, and ProASIC3EL, the content was separated and made into a new document.</p> <p>For information on other low power flash family I/O structures, refer to the following documents:</p> <p><i>I/O Structures in IGLOO and ProASIC3 Devices</i> contains information specific to IGLOO, ProASIC3, and ProASIC3L I/O features.</p> <p><i>I/O Structures in IGLOO PLUS Devices</i> contains information specific to IGLOO PLUS I/O features.</p>	N/A

## Flash FPGAs I/O Support

The flash FPGAs listed in Table 9-1 support I/Os and the functions described in this document.

**Table 9-1 • Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: \*The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

### **IGLOO Terminology**

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 9-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

### **ProASIC3 Terminology**

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 9-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

---

## 11 – Programming Flash Devices

---

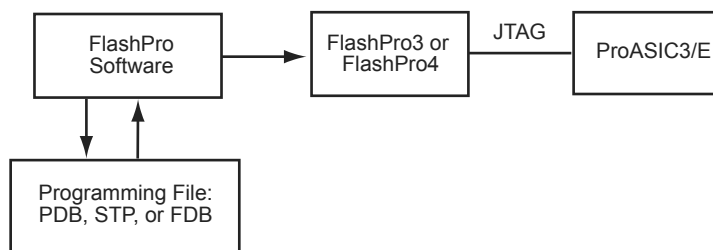
### Introduction

This document provides an overview of the various programming options available for the Microsemi flash families. The electronic version of this document includes active links to all programming resources, which are available at <http://www.microsemi.com/soc/products/hardware/default.aspx>. For Microsemi antifuse devices, refer to the *Programming Antifuse Devices* document.

### Summary of Programming Support

FlashPro4 and FlashPro3 are high-performance in-system programming (ISP) tools targeted at the latest generation of low power flash devices offered by the SmartFusion,<sup>®</sup> Fusion,<sup>®</sup> IGLOO,<sup>®</sup> and ProASIC<sup>®</sup>3 families, including ARM-enabled devices. FlashPro4 and FlashPro3 offer extremely high performance through the use of USB 2.0, are high-speed compliant for full use of the 480 Mbps bandwidth, and can program ProASIC3 devices in under 30 seconds. Powered exclusively via USB, FlashPro4 and FlashPro3 provide a VPUMP voltage of 3.3 V for programming these devices.

FlashPro4 replaced FlashPro3 in 2010. FlashPro4 supports SmartFusion, Fusion, ProASIC3, and IGLOO devices as well as future generation flash devices. FlashPro4 also adds 1.2 V programming for IGLOO nano V2 devices. FlashPro4 is compatible with FlashPro3; however it adds a programming mode (PROG\_MODE) signal to the previously unused pin 4 of the JTAG connector. The PROG\_MODE goes high during programming and can be used to turn on a 1.5 V external supply for those devices that require 1.5 V for programming. If both FlashPro3 and FlashPro4 programmers are used for programming the same boards, pin 4 of the JTAG connector must not be connected to anything on the board because FlashPro4 uses pin 4 for PROG\_MODE.



---

**Figure 11-1 • FlashPro Programming Setup**

## Device Programmers

### Single Device Programmer

Single device programmers are used to program a device before it is mounted on the system board.

The advantage of using device programmers is that no programming hardware is required on the system board. Therefore, no additional components or board space are required.

Adapter modules are purchased with single device programmers to support the FPGA packages used. The FPGA is placed in the adapter module and the programming software is run from a PC. Microsemi supplies the programming software for all of the Microsemi programmers. The software allows for the selection of the correct die/package and programming files. It will then program and verify the device.

- Single-site programmers

A single-site programmer programs one device at a time. Microsemi offers Silicon Sculptor 3, built by BP Microsystems, as a single-site programmer. Silicon Sculptor 3 and associated software are available only from Microsemi.

- Advantages: Lower cost than multi-site programmers. No additional overhead for programming on the system board. Allows local control of programming and data files for maximum security. Allows on-demand programming on-site.
- Limitations: Only programs one device at a time.

- Multi-site programmers

Often referred to as batch or gang programmers, multi-site programmers can program multiple devices at the same time using the same programming file. This is often used for large volume programming and by programming houses. The sites often have independent processors and memory enabling the sites to operate concurrently, meaning each site may start programming the same file independently. This enables the operator to change one device while the other sites continue programming, which increases throughput. Multiple adapter modules for the same package are required when using a multi-site programmer. Silicon Sculptor I, II, and 3 programmers can be cascaded to program multiple devices in a chain. Multi-site programmers, such as the BP2610 and BP2710, can also be purchased from BP Microsystems. When using BP Microsystems multi-site programmers, users must use programming adapter modules available only from Microsemi. Visit the Microsemi SoC Products Group website to view the part numbers of the desired adapter module:

[http://www.microsemi.com/soc/products/hardware/program\\_debug/ss/modules.aspx](http://www.microsemi.com/soc/products/hardware/program_debug/ss/modules.aspx).

Also when using BP Microsystems programmers, customers must use Microsemi programming software to ensure the best programming result will occur.

- Advantages: Provides the capability of programming multiple devices at the same time. No additional overhead for programming on the system board. Allows local control of programming and data files for maximum security.
  - Limitations: More expensive than a single-site programmer
- Automated production (robotic) programmers

Automated production programmers are based on multi-site programmers. They consist of a large input tray holding multiple parts and a robotic arm to select and place parts into appropriate programming sockets automatically. When the programming of the parts is complete, the parts are removed and placed in a finished tray. The automated programmers are often used in volume programming houses to program parts for which the programming time is small. BP Microsystems part number BP4710, BP4610, BP3710 MK2, and BP3610 are available for this purpose. Auto programmers cannot be used to program RTAX-S devices.

Where an auto-programmer is used, the appropriate open-top adapter module from BP Microsystems must be used.

## Related Documents

Below is a list of related documents, their location on the Microsemi SoC Products Group website, and a brief summary of each document.

### Application Notes

*Programming Antifuse Devices*

[http://www.microsemi.com/soc/documents/AntifuseProgram\\_AN.pdf](http://www.microsemi.com/soc/documents/AntifuseProgram_AN.pdf)

*Implementation of Security in Actel's ProASIC and ProASIC<sup>PLUS</sup> Flash-Based FPGAs*

[http://www.microsemi.com/soc/documents/Flash\\_Security\\_AN.pdf](http://www.microsemi.com/soc/documents/Flash_Security_AN.pdf)

### User's Guides

#### **FlashPro Programmers**

FlashPro4,<sup>1</sup> FlashPro3, FlashPro Lite, and FlashPro<sup>2</sup>

[http://www.microsemi.com/soc/products/hardware/program\\_debug/flashpro/default.aspx](http://www.microsemi.com/soc/products/hardware/program_debug/flashpro/default.aspx)

*FlashPro User's Guide*

[http://www.microsemi.com/soc/documents/FlashPro\\_UG.pdf](http://www.microsemi.com/soc/documents/FlashPro_UG.pdf)

The FlashPro User's Guide includes hardware and software setup, self-test instructions, use instructions, and a troubleshooting / error message guide.

#### **Silicon Sculptor 3 and Silicon Sculptor II**

[http://www.microsemi.com/soc/products/hardware/program\\_debug/ss/default.aspx](http://www.microsemi.com/soc/products/hardware/program_debug/ss/default.aspx)

### Other Documents

<http://www.microsemi.com/soc/products/solutions/security/default.aspx#flashlock>

The security resource center describes security in Microsemi Flash FPGAs.

*Quality and Reliability Guide*

<http://www.microsemi.com/soc/documents/RelGuide.pdf>

*Programming and Functional Failure Guidelines*

[http://www.microsemi.com/soc/documents/FA\\_Policies\\_Guidelines\\_5-06-00002.pdf](http://www.microsemi.com/soc/documents/FA_Policies_Guidelines_5-06-00002.pdf)

---

1. FlashPro4 replaced FlashPro3 in Q1 2010.  
2. FlashPro is no longer available.

## Circuit Verification

The power switching circuit recommended above is implemented on Microsemi's Icicle board (Figure 14-2). On the Icicle board, VJTAGENB is used to control the N-Channel Digital FET; however, this circuit was modified to use TRST instead of VJTAGENB in this application. There are three important aspects of this circuit that were verified:

1. The rise on VCC from 1.2 V to 1.5 V when TRST is HIGH
2. VCC rises to 1.5 V before programming begins.
3. VCC switches from 1.5 V to 1.2 V when TRST is LOW.

## Verification Steps

1. The rise on VCC from 1.2 V to 1.5 V when TRST is HIGH.
- 

---

### **Figure 14-2 • Core Voltage on the IGLOO AGL125-QNG132 Device**

In the oscilloscope plots (Figure 14-2), the TRST from FlashPro3 and the VCC core voltage of the IGLOO device are labeled. This plot shows the rise characteristic of the TRST signal from FlashPro3. Once the TRST signal is asserted HIGH, the LTC3025 shown in Figure 14-1 on page 343 senses the increase in voltage and changes the output from 1.2 V to 1.5 V. It takes the circuit approximately 100  $\mu$ s to respond to TRST and change the voltage to 1.5 V on the VCC core.

## List of Changes

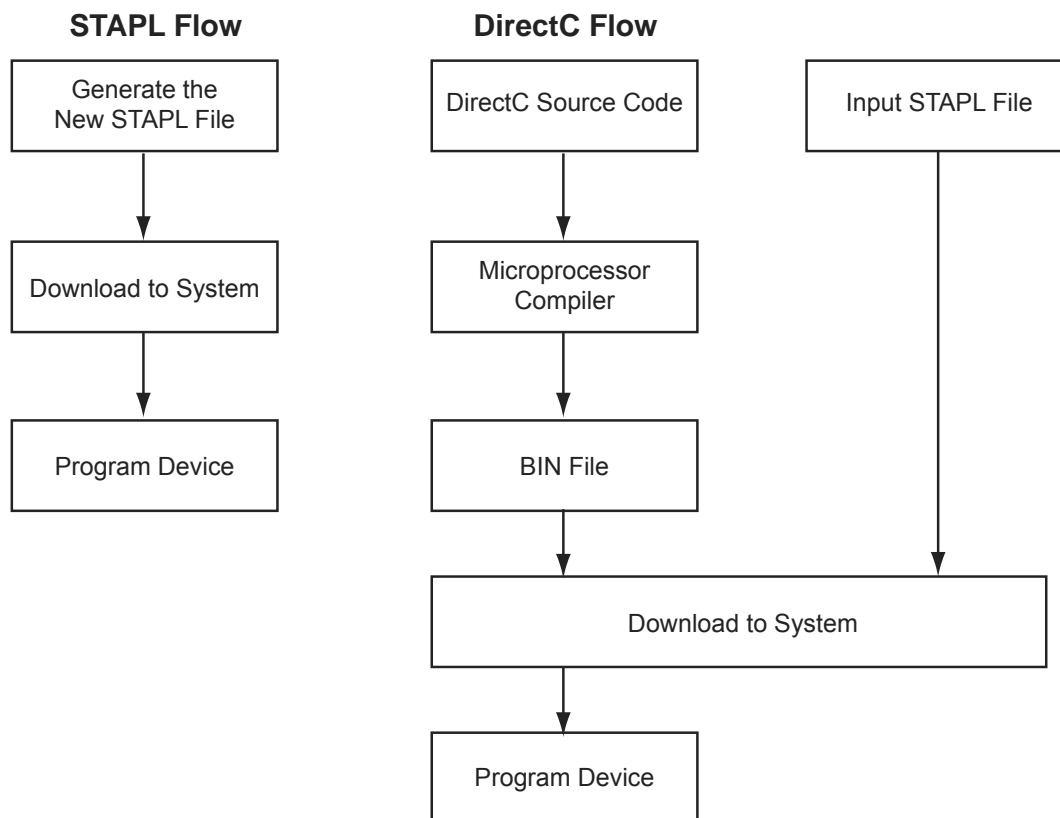
The following table lists critical changes that were made in each revision of the chapter.

Date	Changes	Page
July 2010	This chapter is no longer published separately with its own part number and version but is now part of several FPGA fabric user's guides.	N/A
v1.1 (October 2008)	The "Introduction" was revised to include information about the core supply voltage range of operation in V2 devices.	341
	IGLOO nano device support was added to Table 14-1 • Flash-Based FPGAs Supporting Voltage Switching Circuit.	342
	The "Circuit Description" section was updated to include IGLOO PLUS core operation from 1.2 V to 1.5 V in 50 mV increments.	343
v1.0 (August 2008)	The "Microsemi's Flash Families Support Voltage Switching Circuit" section was revised to include new families and make the information more concise.	342



## STAPL vs. DirectC

Programming the low power flash devices is performed using DirectC or the STAPL player. Both tools use the STAPL file as an input. DirectC is a compiled language, whereas STAPL is an interpreted language. Microprocessors will be able to load the FPGA using DirectC much more quickly than STAPL. This speed advantage becomes more apparent when lower clock speeds of 8- or 16-bit microprocessors are used. DirectC also requires less memory than STAPL, since the programming algorithm is directly implemented. STAPL does have one advantage over DirectC—the ability to upgrade. When a new programming algorithm is required, the STAPL user simply needs to regenerate a STAPL file using the latest version of the Designer software and download it to the system. The DirectC user must download the latest version of DirectC from Microsemi, compile everything, and download the result into the system (Figure 15-4).



**Figure 15-4 • STAPL vs. DirectC**