Welcome to 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	154
Number of Gates	1000000
Voltage - Supply	1.14V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 100°C (TJ)
Package / Case	208-BFQFP
Supplier Device Package	208-PQFP (28x28)
Purchase URL	https://www.e-xfl.com/product-detail/microsemi/a3p1000l-1pq208i

2 – Flash*Freeze Technology and Low Power Modes

Flash*Freeze Technology and Low Power Modes

Microsemi IGLOO,® IGLOO nano, IGLOO PLUS, ProASIC®3L, and Radiation-Tolerant (RT) ProASIC3 FPGAs with Flash*Freeze technology are designed to meet the most demanding power and area challenges of today's portable electronics products with a reprogrammable, small-footprint, full-featured flash FPGA. These devices offer lower power consumption in static and dynamic modes, utilizing the unique Flash*Freeze technology, than any other FPGA or CPLD.

IGLOO, IGLOO nano, IGLOO PLUS, ProASIC3L, and RT ProASIC3 devices offer various power-saving modes that enable every system to utilize modes that achieve the lowest total system power. Low Power Active capability (static idle) allows for ultra-low power consumption while the device is operational in the system by maintaining SRAM, registers, I/Os, and logic functions.

Flash*Freeze technology provides an ultra-low power static mode (Flash*Freeze mode) that retains all SRAM and register information with rapid recovery to Active (operating) mode. IGLOO nano and IGLOO PLUS devices have an additional feature when operating in Flash*Freeze mode, allowing them to retain I/O states as well as SRAM and register states. This mechanism enables the user to quickly (within 1 µs) enter and exit Flash*Freeze mode by activating the Flash*Freeze (FF) pin while all power supplies are kept in their original states. In addition, I/Os and clocks connected to the FPGA can still be toggled without impact on device power consumption. While in Flash*Freeze mode, the device retains all core register states and SRAM information. This mode can be configured so that no power is consumed by the I/O banks, clocks, JTAG pins, or PLLs; and the IGLOO and IGLOO PLUS devices consume as little as 5 µW, while IGLOO nano devices consume as little as 2 µW. Microsemi offers a state management IP core to aid users in gating clocks and managing data before entering Flash*Freeze mode.

This document will guide users in selecting the best low power mode for their applications, and introduces Microsemi's Flash*Freeze management IP core.

Figure 2-1 shows the concept of FF pin control in Flash*Freeze mode type 1.

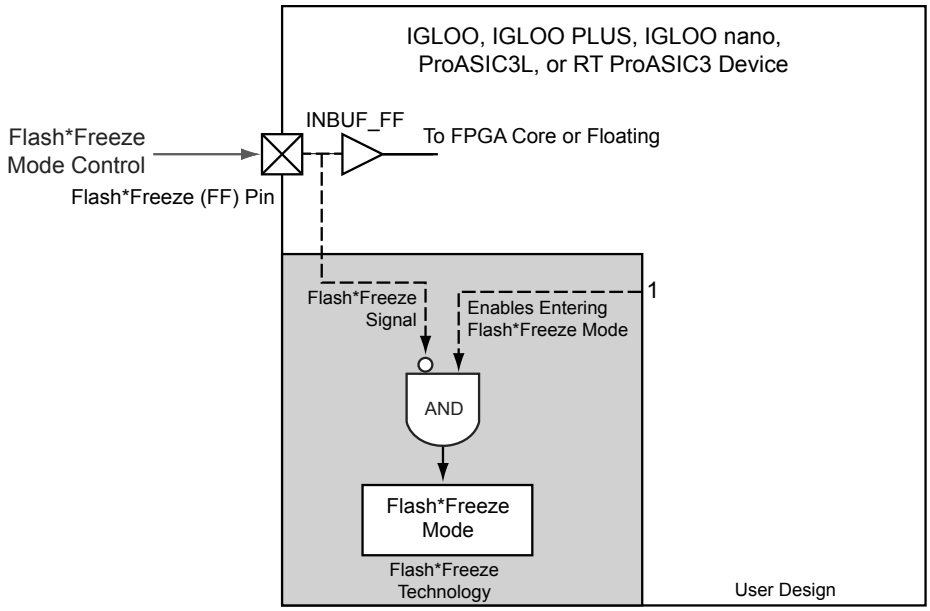


***Figure 2-1* • Flash*Freeze Mode Type 1 – Controlled by the Flash*Freeze Pin**

Figure 2-2 shows the timing diagram for entering and exiting Flash*Freeze mode type 1.

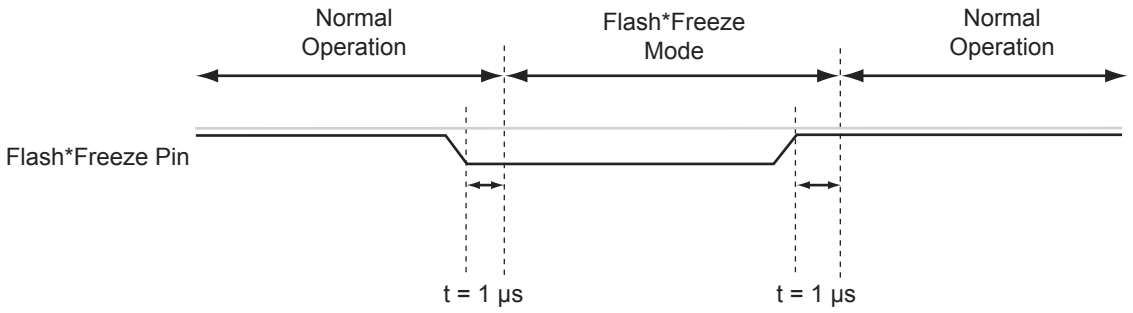


***Figure 2-2* • Flash*Freeze Mode Type 1 – Timing Diagram**

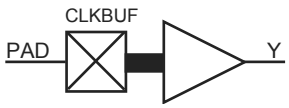
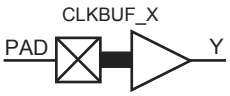
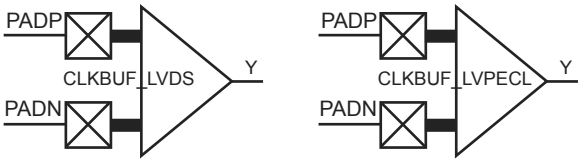
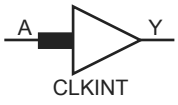
Design Recommendations

The following sections provide design flow recommendations for using a global network in a design.

Global Macros and I/O Standards

The larger low power flash devices have six chip global networks and four quadrant global networks. However, the same clock macros are used for assigning signals to chip globals and quadrant globals. Depending on the clock macro placement or assignment in the Physical Design Constraint (PDC) file or MultiView Navigator (MVN), the signal will use the chip global network or quadrant network. Table 3-8 lists the clock macros available for low power flash devices. Refer to the *IGLOO, ProASIC3, SmartFusion, and Fusion Macro Library Guide* for details.

Table 3-8 • **Clock Macros**

Macro Name	Description	Symbol
CLKBUF	Input macro for Clock Network	
CLKBUF_x	Input macro for Clock Network with specific I/O standard	
CLKBUF_LVDS/LVPECL	LVDS or LVPECL input macro for Clock Network (not supported for IGLOO nano or ProASIC3 nano devices)	
CLKINT	Macro for internal clock interface	
CLKBIBUF	Bidirectional macro with input dedicated to routed Clock Network	

Use these available macros to assign a signal to the global network. In addition to these global macros, PLL and CLKDLY macros can also drive the global networks. Use I/O–standard–specific clock macros (CLKBUF_x) to instantiate a specific I/O standard for the global signals. Table 3-9 on page 63 shows the list of these I/O–standard–specific macros. Note that if you use these I/O–standard–specific clock macros, you cannot change the I/O standard later in the design stage. If you use the regular CLKBUF macro, you can use MVN or the PDC file in Designer to change the I/O standard. The default I/O

standard for CLKBUF is LVTTL in the current Microsemi Libero® System-on-Chip (SoC) and Designer software.

***Table 3-9* • I/O Standards within CLKBUF**

Name	Description
CLKBUF_LVCMOS5	LVCMOS clock buffer with 5.0 V CMOS voltage level
CLKBUF_LVCMOS33	LVCMOS clock buffer with 3.3 V CMOS voltage level
CLKBUF_LVCMOS25	LVCMOS clock buffer with 2.5 V CMOS voltage level[1]
CLKBUF_LVCMOS18	LVCMOS clock buffer with 1.8 V CMOS voltage level
CLKBUF_LVCMOS15	LVCMOS clock buffer with 1.5 V CMOS voltage level
CLKBUF_LVCMOS12	LVCMOS clock buffer with 1.2 V CMOS voltage level
CLKBUF_PCI	PCI clock buffer
CLKBUF_PCIX	PCIX clock buffer
CLKBUF_GTL25	GTL clock buffer with 2.5 V CMOS voltage level[1]
CLKBUF_GTL33	GTL clock buffer with 3.3 V CMOS voltage level[1]
CLKBUF_GTLP25	GTL+ clock buffer with 2.5 V CMOS voltage level[1]
CLKBUF_GTLP33	GTL+ clock buffer with 3.3 V CMOS voltage level[1]
CLKBUF_ HSTL _I	HSTL Class I clock buffer[1]
CLKBUF_ HSTL _II	HSTL Class II clock buffer[1]
CLKBUF_SSTL2_I	SSTL2 Class I clock buffer[1]
CLKBUF_SSTL2_II	SSTL2 Class II clock buffer[1]
CLKBUF_SSTL3_I	SSTL3 Class I clock buffer[1]
CLKBUF_SSTL3_II	SSTL3 Class II clock buffer[1]

Notes:

1. *Supported in only the IGLOOe, ProASIC3E, AFS600, and AFS1500 devices*
2. *By default, the CLKBUF macro uses the 3.3 V LVTTL I/O technology.*

The current synthesis tool libraries only infer the CLKBUF or CLKINT macros in the netlist. All other global macros must be instantiated manually into your HDL code. The following is an example of CLKBUF_LVCMOS25 global macro instantiations that you can copy and paste into your code:

VHDL

```
component clkbuf_lvcmos25
  port (pad : in std_logic; y : out std_logic);
end component

begin
-- concurrent statements
u2 : clkbuf_lvcmos25 port map (pad =>  ext_clk, y => int_clk);
end
```

Verilog

```
module design (______);

input _____;
output ______;

clkbuf_lvcmos25 u2 (.y(int_clk), .pad(ext_clk);

endmodule
```

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
	Notes were added where appropriate to point out that IGLOO nano and ProASIC3 nano devices do not support differential inputs (SAR 21449).	N/A
	The "Global Architecture" section and "VersaNet Global Network Distribution" section were revised for clarity (SARs 20646, 24779).	47, 49
	The "I/O Banks and Global I/Os" section was moved earlier in the document, renamed to "Chip and Quadrant Global I/Os", and revised for clarity. Figure 3-4 • Global Connections Details, Figure 3-6 • Global Inputs, Table 3-2 • Chip Global Pin Name, and Table 3-3 • Quadrant Global Pin Name are new (SARs 20646, 24779).	51
	The "Clock Aggregation Architecture" section was revised (SARs 20646, 24779).	57
	Figure 3-7 • Chip Global Aggregation was revised (SARs 20646, 24779).	59
	The "Global Macro and Placement Selections" section is new (SARs 20646, 24779).	64
v1.4 (December 2008)	The "Global Architecture" section was updated to include 10 k devices, and to include information about VersaNet global support for IGLOO nano devices.	47
	The Table 3-1 • Flash-Based FPGAs was updated to include IGLOO nano and ProASIC3 nano devices.	48
	The "VersaNet Global Network Distribution" section was updated to include 10 k devices and to note an exception in global lines for nano devices.	49
	Figure 3-2 • Simplified VersaNet Global Network (30 k gates and below) is new.	50
	The "Spine Architecture" section was updated to clarify support for 10 k and nano devices.	57
	Table 3-4 • Globals/Spines/Rows for IGLOO and ProASIC3 Devices was updated to include IGLOO nano and ProASIC3 nano devices.	57
	The figure in the CLKBUF_LVDS/LVPECL row of Table 3-8 • Clock Macros was updated to change CLKBIBUF to CLKBUF.	62
v1.3 (October 2008)	A third bullet was added to the beginning of the "Global Architecture" section: In Fusion devices, the west CCC also contains a PLL core. In the two larger devices (AFS600 and AFS1500), the west and east CCCs each contain a PLL.	47
	The "Global Resource Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	48
	Table 3-4 • Globals/Spines/Rows for IGLOO and ProASIC3 Devices was updated to include A3PE600/L in the device column.	57
	Table note 1 was revised in Table 3-9 • I/O Standards within CLKBUF to include AFS600 and AFS1500.	63
v1.2 (June 2008)	The following changes were made to the family descriptions in Table 3-1 • Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	48

Available I/O Standards

***Table 4-4* • Available I/O Standards within CLKBUF and CLKBUF_LVDS/LVPECL Macros**

CLKBUF_LVCMOS5
CLKBUF_LVCMOS33 [1]
CLKBUF_LVCMOS25 [2]
CLKBUF_LVCMOS18
CLKBUF_LVCMOS15
CLKBUF_PCI
CLKBUF_PCIX [3]
CLKBUF_GTL25 [2,3]
CLKBUF_GTL33 [2,3]
CLKBUF_GTLP25 [2,3]
CLKBUF_GTLP33 [2,3]
CLKBUF_HSTL_I [2,3]
CLKBUF_HSTL_II [2,3]
CLKBUF_SSTL3_I [2,3]
CLKBUF_SSTL3_II [2,3]
CLKBUF_SSTL2_I [2,3]
CLKBUF_SSTL2_II [2,3]
CLKBUF_LVDS [4,5]
CLKBUF_LVPECL[5]

Notes:

1. *By default, the CLKBUF macro uses 3.3 V LVTTL I/O technology. For more details, refer to the IGLOO, ProASIC3, SmartFusion, and Fusion Macro Library Guide.*
2. *I/O standards only supported in ProASIC3E and IGLOOe families.*
3. *I/O standards only supported in the following Fusion devices: AFS600 and AFS1500.*
4. *B-LVDS and M-LVDS standards are supported by CLKBUF_LVDS.*
5. *Not supported for IGLOO nano and ProASIC3 nano devices.*

Global Synthesis Constraints

The Synplify® synthesis tool, by default, allows six clocks in a design for Fusion, IGLOO, and ProASIC3. When more than six clocks are needed in the design, a user synthesis constraint attribute, syn_global_buffers, can be used to control the maximum number of clocks (up to 18) that can be inferred by the synthesis engine.

High-fanout nets will be inferred with clock buffers and/or internal clock buffers. If the design consists of CCC global buffers, they are included in the count of clocks in the design.

The subsections below discuss the clock input source (global buffers with no programmable delays) and the clock conditioning functional block (global buffers with programmable delays and/or PLL function) in detail.

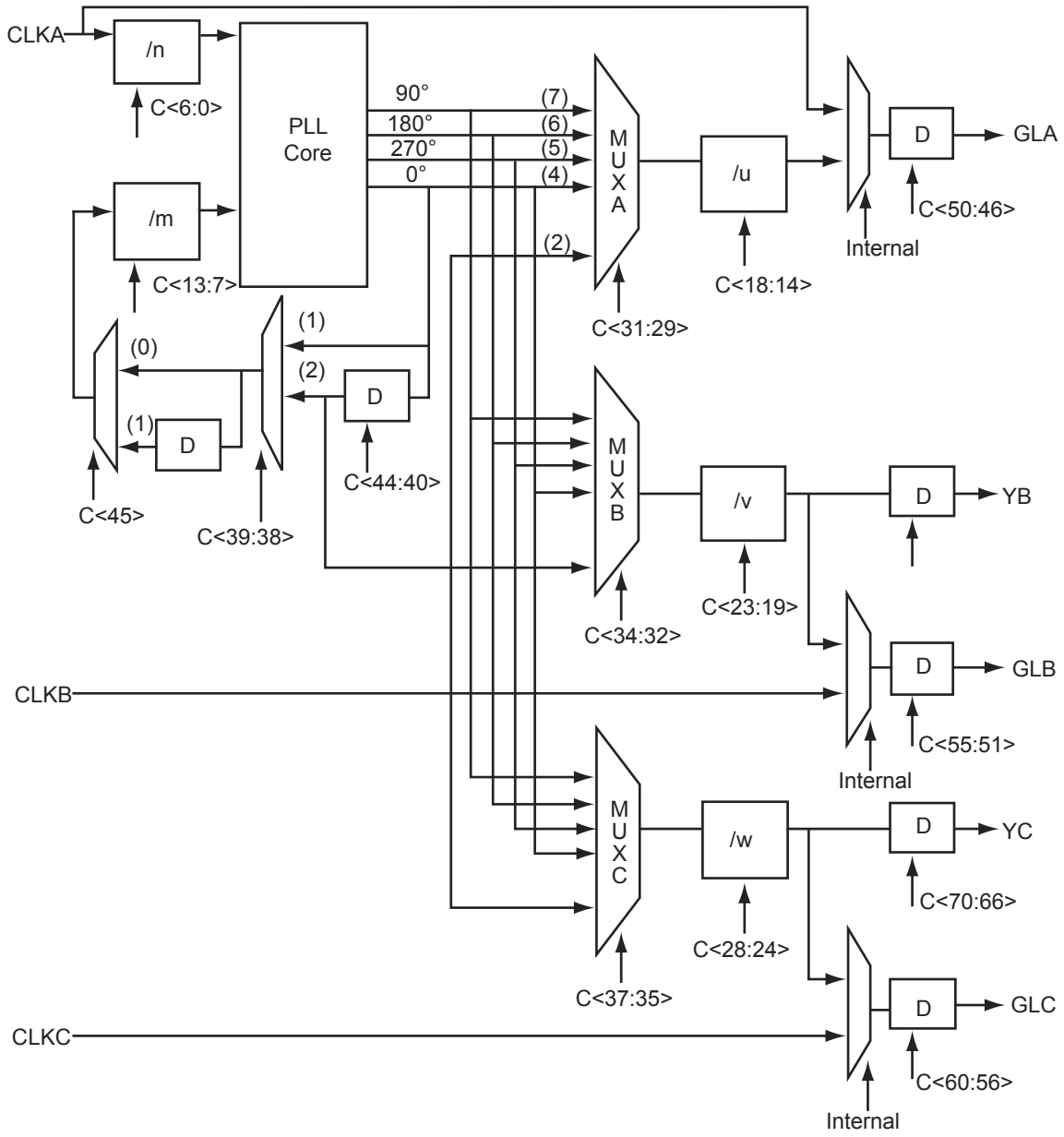


***Figure 4-22* • CCC Block Control Bits – Graphical Representation of Assignments**

global assignments are not allocated properly. See the "Physical Constraints for Quadrant Clocks" section for information on assigning global signals to the quadrant clock networks.

Promoted global signals will be instantiated with CLKINT macros to drive these signals onto the global network. This is automatically done by Designer when the Auto-Promotion option is selected. If the user wishes to assign the signals to the quadrant globals instead of the default chip globals, this can done by using ChipPlanner, by declaring a physical design constraint (PDC), or by importing a PDC file.

Physical Constraints for Quadrant Clocks

If it is necessary to promote global clocks (CLKBUF, CLKINT, PLL, CLKDLY) to quadrant clocks, the user can define PDCs to execute the promotion. PDCs can be created using PDC commands (pre-compile) or the MultiView Navigator (MVN) interface (post-compile). The advantage of using the PDC flow over the MVN flow is that the Compile stage is able to automatically promote any regular net to a global net before assigning it to a quadrant. There are three options to place a quadrant clock using PDC commands:

- Place a clock core (not hardwired to an I/O) into a quadrant clock location.
- Place a clock core (hardwired to an I/O) into an I/O location (set_io) or an I/O module location (set_location) that drives a quadrant clock location.
- Assign a net driven by a regular net or a clock net to a quadrant clock using the following command:

  ```
  assign_local_clock -net <net name> -type quadrant <quadrant clock region>
  ```

  where

  `<net name>` is the name of the net assigned to the local user clock region.

  `<quadrant clock region>` defines which quadrant the net should be assigned to. Quadrant clock regions are defined as UL (upper left), UR (upper right), LL (lower left), and LR (lower right).

Note: If the net is a regular net, the software inserts a CLKINT buffer on the net.

For example:

```
assign_local_clock -net localReset -type quadrant UR
```

Keep in mind the following when placing quadrant clocks using MultiView Navigator:

Hardwired I/O–Driven CCCs

- Find the associated clock input port under the Ports tab, and place the input port at one of the Gmn* locations using PinEditor or I/O Attribute Editor, as shown in Figure 4-32.

***Figure 4-32* • Port Assignment for a CCC with Hardwired I/O Clock Input**

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:

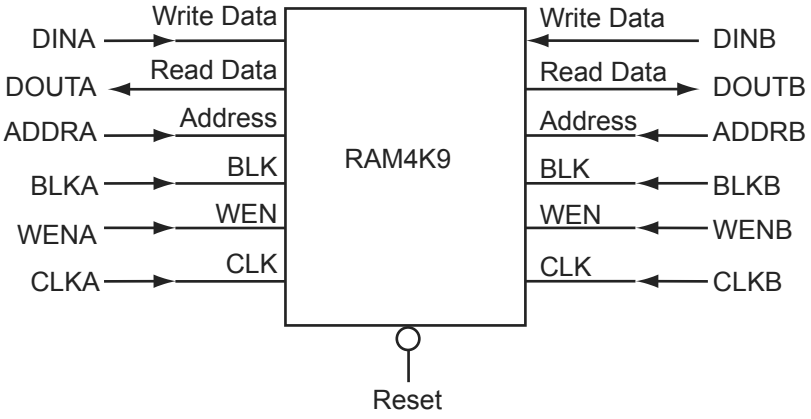
1. Generate FlashROM using SmartGen. From the Properties section in the FlashROM Settings dialog box, select the **Auto Inc** or **Read From File** region. For the Auto Inc region, specify the desired step value. You will not be able to modify this value in the FlashPoint software.
2. Go through the 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 Features

RAM4K9 Macro

RAM4K9 is the dual-port configuration of the RAM block (Figure 6-4). The RAM4K9 nomenclature refers to both the deepest possible configuration and the widest possible configuration the dual-port RAM block can assume, and does not denote a possible memory aspect ratio. The RAM block can be configured to the following aspect ratios: 4,096×1, 2,048×2, 1,024×4, and 512×9. RAM4K9 is fully synchronous and has the following features:

- Two ports that allow fully independent reads and writes at different frequencies
- Selectable pipelined or nonpipelined read
- Active-low block enables for each port
- Toggle control between read and write mode for each port
- Active-low asynchronous reset
- Pass-through write data or hold existing data on output. In pass-through mode, the data written to the write port will immediately appear on the read port.
- Designer software will automatically facilitate falling-edge clocks by bubble-pushing the inversion to previous stages.



Note: For timing diagrams of the RAM signals, refer to the appropriate family datasheet.

***Figure 6-4* • RAM4K9 Simplified Configuration**

Signal Descriptions for RAM4K9

Note: Automotive ProASIC3 devices support single-port SRAM capabilities, or dual-port SRAM only under specific conditions. Dual-port mode is supported if the clocks to the two SRAM ports are the same and 180° out of phase (i.e., the port A clock is the inverse of the port B clock). Since Libero SoC macro libraries support a dual-port macro only, certain modifications must be made. These are detailed below.

The following signals are used to configure the RAM4K9 memory element:

WIDTHA and WIDTHB

These signals enable the RAM to be configured in one of four allowable aspect ratios (Table 6-2 on page 154).

Note: When using the SRAM in single-port mode for Automotive ProASIC3 devices, WIDTHB should be tied to ground.

Advanced I/Os—IGLOO, ProASIC3L, and ProASIC3

Table 7-2 and Table 7-3 show the voltages and compatible I/O standards for the IGLOO, ProASIC3L, and ProASIC3 families.

I/Os provide programmable slew rates (except 30 K gate devices), drive strengths, and weak pull-up and pull-down circuits. 3.3 V PCI and 3.3 V PCI-X can be configured to be 5 V–tolerant. See the "5 V Input Tolerance" section on page 194 for possible implementations of 5 V tolerance.

All I/Os are in a known state during power-up, and any power-up sequence is allowed without current impact. Refer to the "I/O Power-Up and Supply Voltage Thresholds for Power-On Reset (Commercial and Industrial)" section in the datasheet for more information. During power-up, before reaching activation levels, the I/O input and output buffers are disabled while the weak pull-up is enabled. Activation levels are described in the datasheet.

***Table 7-2* • Supported I/O Standards**

IGLOO	**AGL015**	**AGL030**	**AGL060**	**AGL125**	**AGL250**		**AGL600**	**AGL1000**
ProASIC3	**A3P015**	**A3P030**	**A3P060**	**A3P125**	**A3P250/ A3P250L**	**A3P400**	**A3P600/ A3P600L**	**A3P1000/ A3P1000L**
Single-Ended								
LVTTL/LVCMOS 3.3 V, LVCMOS 2.5 V / 1.8 V / 1.5 V / 1.2 V LVCMOS 2.5 V / 5.0 V	✓	✓	✓	✓	✓	✓	✓	✓
3.3 V PCI/PCI-X	–	–	✓	✓	✓	✓	✓	✓
Differential								
LVPECL, LVDS, B-LVDS, M-LVDS	–	–	–	–	✓	✓	✓	✓

I/O Banks and I/O Standards Compatibility

I/Os are grouped into I/O voltage banks.

Each I/O voltage bank has dedicated I/O supply and ground voltages (VMV/GNDQ for input buffers and VCCI/GND for output buffers). This isolation is necessary to minimize simultaneous switching noise from the input and output (SSI and SSO). The switching noise (ground bounce and power bounce) is generated by the output buffers and transferred into input buffer circuits, and vice versa. Because of these dedicated supplies, only I/Os with compatible standards can be assigned to the same I/O voltage bank. Table 7-3 shows the required voltage compatibility values for each of these voltages.

There are four I/O banks on the 250K gate through 1M gate devices.

There are two I/O banks on the 30K, 60K, and 125K gate devices.

I/O standards are compatible if their VCCI and VMV values are identical. VMV and GNDQ are "quiet" input power supply pins and are not used on 30K gate devices (Table 7-3).

***Table 7-3* • VCCI Voltages and Compatible IGLOO and ProASIC3 Standards**

VCCI and VMV (typical)	**Compatible Standards**
3.3 V	LVTTL/LVCMOS 3.3, PCI 3.3, PCI-X 3.3 LVPECL
2.5 V	LVCMOS 2.5, LVCMOS 2.5/5.0, LVDS, B-LVDS, M-LVDS
1.8 V	LVCMOS 1.8
1.5 V	LVCMOS 1.5
1.2 V	LVCMOS 1.2

I/O Bank Structure

Low power flash device I/Os are divided into multiple technology banks. The number of banks is device-dependent. The IGLOOe, ProASIC3EL, and ProASIC3E devices have eight banks (two per side); and IGLOO, ProASIC3L, and ProASIC3 devices have two to four banks. Each bank has its own V**CCI** power supply pin. Multiple I/O standards can co-exist within a single I/O bank.

In IGLOOe, ProASIC3EL, and ProASIC3E devices, each I/O bank is subdivided into VREF minibanks. These are used by voltage-referenced I/Os. VREF minibanks contain 8 to 18 I/Os. All I/Os in a given minibank share a common VREF line (only one VREF pin is needed per VREF minibank). Therefore, if an I/O in a VREF minibank is configured as a VREF pin, the remaining I/Os in that minibank will be able to use the voltage assigned to that pin. If the location of the VREF pin is selected manually in the software, the user must satisfy VREF rules (refer to the "I/O Software Control in Low Power Flash Devices" section on page 251). If the user does not pick the VRE_F pin manually, the software automatically assigns it.

Figure 7-3 is a snapshot of a section of the I/O ring, showing the basic elements of an I/O tile, as viewed from the Designer place-and-route tool's MultiView Navigator (MVN).

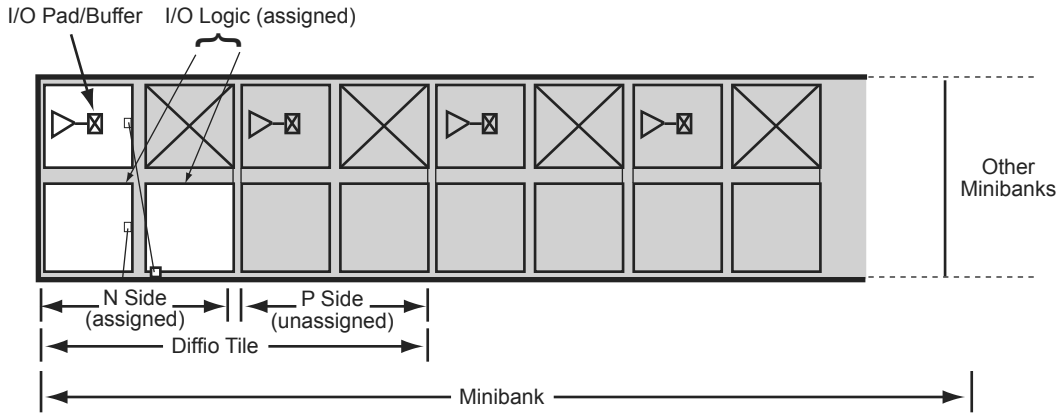


***Figure 7-3* • Snapshot of an I/O Tile**

Low power flash device I/Os are implemented using two tile types: I/O and differential I/O (diffio).

The diffio tile is built up using two I/O tiles, which form an I/O pair (P side and N side). These I/O pairs are used according to differential I/O standards. Both the P and N sides of the diffio tile include an I/O buffer and two I/O logic blocks (auxiliary and main logic).

Every minibank (E devices only) is built up from multiple diffio tiles. The number of the minibank depends on the different-size dies. Refer to the "I/O Architecture" section on page 181 for an illustration of the minibank structure.

Figure 7-4 on page 183 shows a simplified diagram of the I/O buffer circuitry. The Output Enable signal (OE) enables the output buffer to pass the signal from the core logic to the pin. The output buffer contains ESD protection circuitry, an n-channel transistor that shunts all ESD surges (up to the limit of the device ESD specification) to GND. This transistor also serves as an output pull-down resistor.

Each output buffer also contains programmable slew rate, drive strength, programmable power-up state (pull-up/-down resistor), hot-swap, 5 V tolerance, and clamp diode control circuitry. Multiple flash switches (not shown in Figure 7-4 on page 183) are programmed by user selections in the software to activate different I/O features.

***Table 8-7* • Maximum I/O Frequency for Single-Ended and Differential I/Os in All Banks in ProASIC3E Devices (maximum drive strength and high slew selected)**

Specification	Maximum Performance		
	ProASIC3E	IGLOOe V2 or V5 Devices, 1.5 V DC Core Supply Voltage	IGLOOe V2, 1.2 V DC Core Supply Voltage
LVTTL/LVCMOS 3.3 V	200 MHz	180 MHz	TBD
LVCMOS 2.5 V	250 MHz	230 MHz	TBD
LVCMOS 1.8 V	200 MHz	180 MHz	TBD
LVCMOS 1.5 V	130 MHz	120 MHz	TBD
PCI	200 MHz	180 MHz	TBD
PCI-X	200 MHz	180 MHz	TBD
HSTL-I	300 MHz	275 MHz	TBD
HSTL-II	300 MHz	275 MHz	TBD
SSTL2-I	300 MHz	275 MHz	TBD
SSTL2-II	300 MHz	275 MHz	TBD
SSTL3-I	300 MHz	275 MHz	TBD
SSTL3-II	300 MHz	275 MHz	TBD
GTL+ 3.3 V	300 MHz	275 MHz	TBD
GTL+ 2.5 V	300 MHz	275 MHz	TBD
GTL 3.3 V	300 MHz	275 MHz	TBD
GTL 2.5 V	300 MHz	275 MHz	TBD
LVDS	350 MHz	300 MHz	TBD
M-LVDS	200 MHz	180 MHz	TBD
B LVDS	200 MHz	180 MHz	TBD
LVPECL	350 MHz	300 MHz	TBD

***Table 8-9* • Hot-Swap Level 1**

Description	Cold-swap
Power Applied to Device	No
Bus State	–
Card Ground Connection	–
Device Circuitry Connected to Bus Pins	–
Example Application	System and card with Microsemi FPGA chip are powered down, and the card is plugged into the system. Then the power supplies are turned on for the system but not for the FPGA on the card.
Compliance of IGLOO and ProASIC3 Devices	30 k gate devices: Compliant Other IGLOO/ProASIC3 devices: Compliant if bus switch used to isolate FPGA I/Os from rest of system IGLOOe/ProASIC3E devices: Compliant I/Os can, but do not have to be set to hot-insertion mode.

***Table 8-10* • Hot-Swap Level 2**

Description	Hot-swap while reset
Power Applied to Device	Yes
Bus State	Held in reset state
Card Ground Connection	Reset must be maintained for 1 ms before, during, and after insertion/removal.
Device Circuitry Connected to Bus Pins	–
Example Application	In the PCI hot-plug specification, reset control circuitry isolates the card busses until the card supplies are at their nominal operating levels and stable.
Compliance of IGLOO and ProASIC3 Devices	30 k gate devices, all IGLOOe/ProASIC3E devices: Compliant I/Os can but do not have to be set to hot-insertion mode. Other IGLOO/ProASIC3 devices: Compliant

I/O Software Support

In Libero SoC software, default settings have been defined for the various I/O standards supported. Changes can be made to the default settings via the use of attributes; however, not all I/O attributes are applicable for all I/O standards. Table 8-16 lists the valid I/O attributes that can be manipulated by the user for each I/O standard.

Single-ended I/O standards in low power flash devices support up to five different drive strengths.

***Table 8-16* • IGLOOe and ProASIC3E I/O Attributes vs. I/O Standard Applications**

I/O Standard	SLEW (output only)	OUT_DRIVE (output only)	SKEW (all macros with OE)	RES_PULL	OUT_LOAD (output only)	COMBINE_REGISTER	IN_DELAY (input only)	IN_DELAY_VAL (input only)	SCHMITT_TRIGGER (input only)	HOT_SWAPPABLE
LVTTL/LVCMOS 3.3 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 2.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 2.5/5.0 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 1.8 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 1.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PCI (3.3 V)			✓		✓	✓	✓	✓		
PCI-X (3.3 V)	✓		✓		✓	✓	✓	✓		
GTL+ (3.3 V)			✓		✓	✓	✓	✓		✓
GTL+ (2.5 V)			✓		✓	✓	✓	✓		✓
GTL (3.3 V)			✓		✓	✓	✓	✓		✓
GTL (2.5 V)			✓		✓	✓	✓	✓		✓
HSTL Class I			✓		✓	✓	✓	✓		✓
HSTL Class II			✓		✓	✓	✓	✓		✓
SSTL2 Class I and II			✓		✓	✓	✓	✓		✓
SSTL3 Class I and II			✓		✓	✓	✓	✓		✓
LVDS, B-LVDS, M-LVDS			✓			✓	✓	✓		✓
LVPECL						✓	✓	✓		✓

Table 8-17 on page 243 lists the default values for the above selectable I/O attributes as well as those that are preset for each I/O standard.

– The I/O standard of technology-specific I/O macros cannot be changed in the I/O Attribute Editor (see Figure 9-6).
– The user MUST instantiate differential I/O macros (LVDS/LVPECL) in the design. This is the only way to use these standards in the design (IGLOO nano and ProASIC3 nano devices do not support differential inputs).
– To implement the DDR I/O function, the user must instantiate a DDR_REG or DDR_OUT macro. This is the only way to use a DDR macro in the design.

***Figure 9-6* • Assigning a Different I/O Standard to the Generic I/O Macro**

Performing Place-and-Route on the Design

The netlist created by the synthesis tool should now be imported into Designer and compiled. During Compile, the user can specify the I/O placement and attributes by importing the PDC file. The user can also specify the I/O placement and attributes using ChipPlanner and the I/O Attribute Editor under MVN.

Defining I/O Assignments in the PDC File

A PDC file is a Tcl script file specifying physical constraints. This file can be imported to and exported from Designer.

Table 9-3 shows I/O assignment constraints supported in the PDC file.

***Table 9-3* • PDC I/O Constraints**

Command	Action	Example	Comment
I/O Banks Setting Constraints			

set_iobank	Sets the I/O supply voltage, V_{CCI}, and the input reference voltage, V_{REF}, for the specified I/O bank.	`set_iobank bankname [-vcci vcci_voltage] [-vref vref_voltage]`
		
		`set_iobank Bank7 -vcci 1.50 -vref 0.75`
			Must use in case of mixed I/O voltage (V_{CCI}) design

set_vref	Assigns a V_{REF} pin to a bank.	`set_vref -bank [bankname] [pinnum]`
		
		`set_vref -bank Bank0 685 704 723 742 761`
			Must use if voltage-referenced I/Os are used

set_vref_defaults	Sets the default V_{REF} pins for the specified bank. This command is ignored if the bank does not need a V_{REF} pin.	`set_vref_defaults bankname`
		
		`set_vref_defaults bank2`	

Note: Refer to the Libero SoC User's Guide *for detailed rules on PDC naming and syntax conventions.*

Input Support for DDR

The basic structure to support a DDR input is shown in Figure 10-2. Three input registers are used to capture incoming data, which is presented to the core on each rising edge of the I/O register clock. Each I/O tile supports DDR inputs.

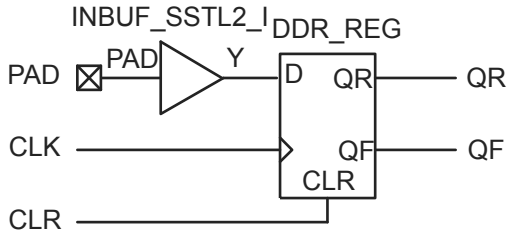


***Figure 10-2* • DDR Input Register Support in Low Power Flash Devices**

Output Support for DDR

The basic DDR output structure is shown in Figure 10-1 on page 271. New data is presented to the output every half clock cycle.

Note: DDR macros and I/O registers do not require additional routing. The combiner automatically recognizes the DDR macro and pushes its registers to the I/O register area at the edge of the chip. The routing delay from the I/O registers to the I/O buffers is already taken into account in the DDR macro.

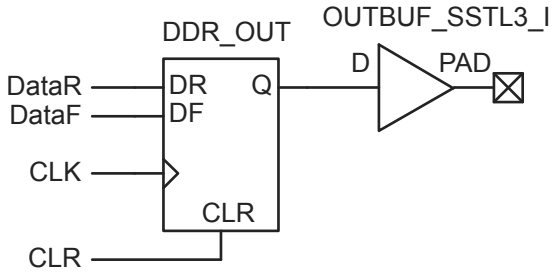


***Figure 10-3* • DDR Output Register (SSTL3 Class I)**

- Programming Centers

  Microsemi programming hardware policy also applies to programming centers. Microsemi expects all programming centers to use certified programmers to program Microsemi devices. If a programming center uses noncertified programmers to program Microsemi devices, the "Noncertified Programmers" policy applies.

Important Programming Guidelines

Preprogramming Setup

Before programming, several steps are required to ensure an optimal programming yield.

Use Proper Handling and Electrostatic Discharge (ESD) Precautions

Microsemi FPGAs are sensitive electronic devices that are susceptible to damage from ESD and other types of mishandling. For more information about ESD, refer to the *Quality and Reliability Guide,* beginning with page 41.

Use the Latest Version of the Designer Software to Generate Your Programming File (recommended)

The files used to program Microsemi flash devices (*.bit, *.stp, *.pdb) contain important information about the switches that will be programmed in the FPGA. Find the latest version and corresponding release notes at http://www.microsemi.com/soc/download/software/designer/. Also, programming files must always be zipped during file transfer to avoid the possibility of file corruption.

Use the Latest Version of the Programming Software

The programming software is frequently updated to accommodate yield enhancements in FPGA manufacturing. These updates ensure maximum programming yield and minimum programming times. Before programming, always check the version of software being used to ensure it is the most recent. Depending on the programming software, refer to one of the following:

- FlashPro: http://www.microsemi.com/soc/download/program_debug/flashpro/
- Silicon Sculptor: http://www.microsemi.com/soc/download/program_debug/ss/

Use the Most Recent Adapter Module with Silicon Sculptor

Occasionally, Microsemi makes modifications to the adapter modules to improve programming yields and programming times. To identify the latest version of each module before programming, visit http://www.microsemi.com/soc/products/hardware/program_debug/ss/modules.aspx.

Perform Routine Hardware Self-Diagnostic Test

- Adapter modules must be regularly cleaned. Adapter modules need to be inserted carefully into the programmer to make sure the DIN connectors (pins at the back side) are not damaged.
- FlashPro

  The self-test is only applicable when programming with FlashPro and FlashPro3 programmers. It is not supported with FlashPro4 or FlashPro Lite. To run the self-diagnostic test, follow the instructions given in the "Performing a Self-Test" section of http://www.microsemi.com/soc/documents/FlashPro_UG.pdf.
- Silicon Sculptor

  The self-diagnostic test verifies correct operation of the pin drivers, power supply, CPU, memory, and adapter module. This test should be performed with an adapter module installed and before every programming session. At minimum, the test must be executed every week. To perform self-diagnostic testing using the Silicon Sculptor software, perform the following steps, depending on the operating system:

  – DOS: From anywhere in the software, type **ALT + D**.

  – Windows: Click **Device** > choose **Actel Diagnostic** > select the **Test** tab > click **OK**.

  Silicon Sculptor programmers must be verified annually for calibration. Refer to the *Silicon Sculptor Verification of Calibration Work Instruction* document on the website.

15 – Microprocessor Programming of Microsemi's Low Power Flash Devices

Introduction

The Fusion, IGLOO, and ProASIC3 families of flash FPGAs support in-system programming (ISP) with the use of a microprocessor. Flash-based FPGAs store their configuration information in the actual cells within the FPGA fabric. SRAM-based devices need an external configuration memory, and hybrid nonvolatile devices store the configuration in a flash memory inside the same package as the SRAM FPGA. Since the programming of a true flash FPGA is simpler, requiring only one stage, it makes sense that programming with a microprocessor in-system should be simpler than with other SRAM FPGAs. This reduces bill-of-materials costs and printed circuit board (PCB) area, and increases system reliability.

Nonvolatile flash technology also gives the low power flash devices the advantage of a secure, low power, live-at-power-up, and single-chip solution. Low power flash devices are reprogrammable and offer time-to-market benefits at an ASIC-level unit cost. These features enable engineers to create high-density systems using existing ASIC or FPGA design flows and tools.

This document is an introduction to microprocessor programming only. To explain the difference between the options available, user's guides for DirectC and STAPL provide more detail on implementing each style.

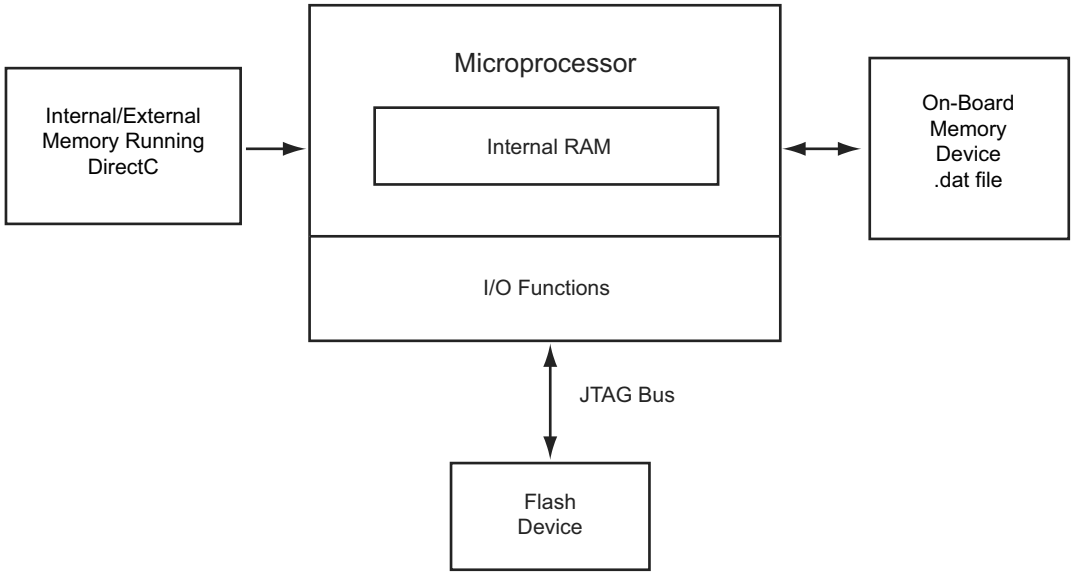
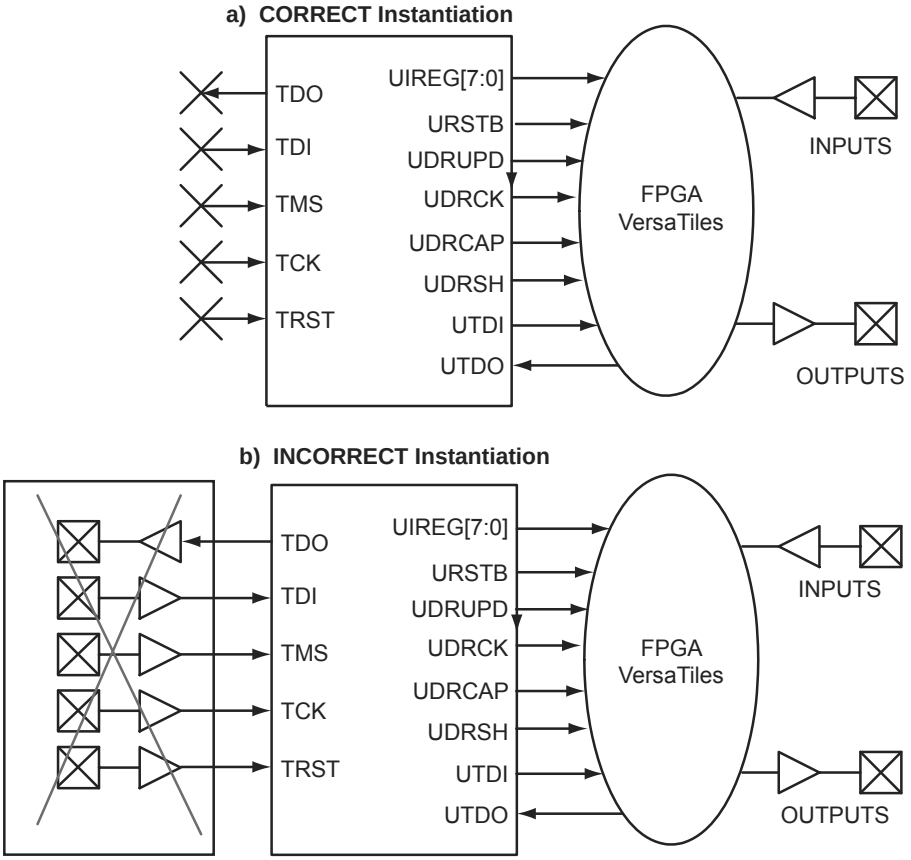


***Figure 15-1* • ISP Using Microprocessor**



Note: Do not connect JTAG pins (TDO, TDI, TMS, TCK, or TRST) to I/Os in the design.

Figure 17-3 • **Connectivity Method of UJTAG Macro**

UJTAG Operation

There are a few basic functions of the UJTAG macro that users must understand before designing with it. The most important fundamental concept of the UJTAG design is its connection with the TAP Controller state machine.

TAP Controller State Machine

The 16 states of the TAP Controller state machine are shown in Figure 17-4 on page 367. The 1s and 0s, shown adjacent to the state transitions, represent the TMS values that must be present at the time of a rising TCK edge for a state transition to occur. In the states that include the letters "IR," the instruction register operates; in the states that contain the letters "DR," the test data register operates. The TAP Controller receives two control inputs, TMS and TCK, and generates control and clock signals for the rest of the test logic.

On power-up (or the assertion of TRST), the TAP Controller enters the Test-Logic-Reset state. To reset the controller from any other state, TMS must be held HIGH for at least five TCK cycles. After reset, the TAP state changes at the rising edge of TCK, based on the value of TMS.