



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

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

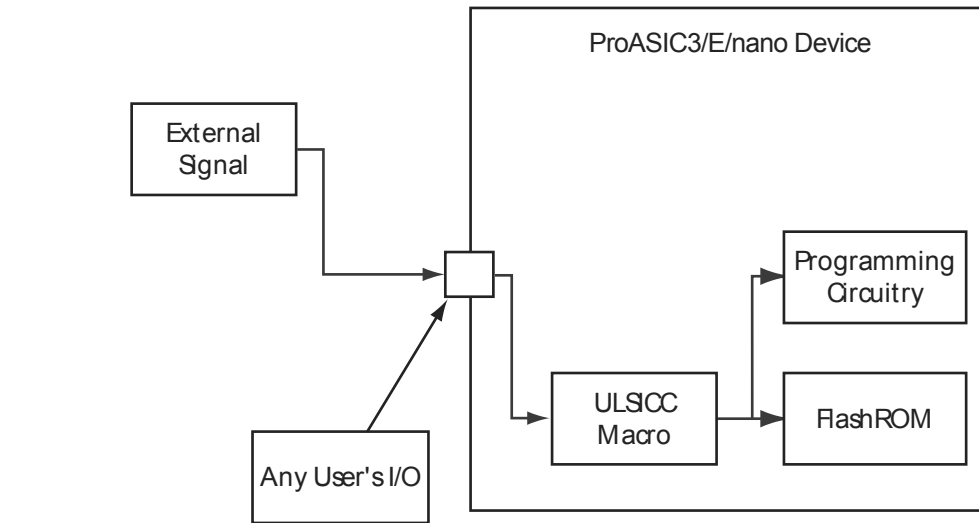
Embedded - FPGAs, or Field Programmable Gate Arrays, are advanced integrated circuits that offer unparalleled flexibility and performance for digital systems. Unlike traditional fixed-function logic devices, FPGAs can be programmed and reprogrammed to execute a wide array of logical operations, enabling 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

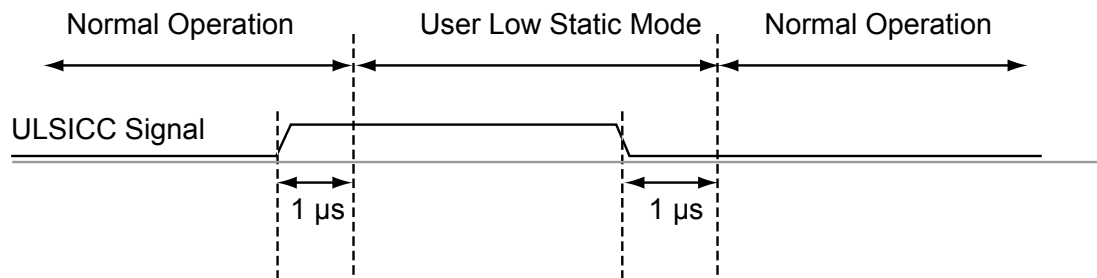
The versatility of Embedded - FPGAs makes them indispensable in numerous fields. In telecommunications.

#### Details

Product Status	Active
Number of LABs/CLBs	-
Number of Logic Elements/Cells	-
Total RAM Bits	-
Number of I/O	49
Number of Gates	20000
Voltage - Supply	1.425V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 100°C (TJ)
Package / Case	68-VFQFN Exposed Pad
Supplier Device Package	68-QFN (8x8)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microchip-technology/a3pn020-2qng68i">https://www.e-xfl.com/product-detail/microchip-technology/a3pn020-2qng68i</a>



**Figure 2-3 • User Low Static (Idle) Mode Application—External Control Signal**



**Figure 2-4 • User Low Static (Idle) Mode Timing Diagram**

## Sleep Mode

ProASIC3/E and ProASIC3 nano FPGAs support Sleep mode when device functionality is not required. In Sleep mode, the VCC (core voltage), VJTAG (JTAG DC voltage), and VPUMP (programming voltage) are grounded, resulting in the FPGA core being turned off to reduce power consumption. While the ProASIC3/E device is in Sleep mode, the rest of the system is still operating and driving the input buffers of the ProASIC3/E device. The driven inputs do not pull up power planes, and the current draw is limited to a minimal leakage current.

Table 2-3 shows the status of the power supplies in Sleep mode. When a power supply is powered off, the corresponding power pin can be left floating or grounded.

**Table 2-3 • Sleep Mode—Power Supply Requirements for ProASIC3/E/nano Devices**

Power Supplies	ProASIC3/E/nano Device
VCC	Powered off
VCCI = VMV	Powered on
VJTAG	Powered off
VPUMP	Powered off

The following will happen during demotion of a global signal to regular nets:

- CLKBUF\_x becomes INBUF\_x; CLKINT is removed from the netlist.
- The essential global macro, such as the output of the Clock Conditioning Circuit, cannot be demoted.
- No automatic buffering will happen.

Since no automatic buffering happens when a signal is demoted, this net may have a high delay due to large fanout. This may have a negative effect on the quality of the results. Microsemi recommends that the automatic global demotion only be used on small-fanout nets. Use clock networks for high-fanout nets to improve timing and routability.

## Spine Assignment

The low power flash device architecture allows the global networks to be segmented and used as clock spines. These spines, also called local clock networks, enable the use of PDC or MVN to assign a signal to a spine.

PDC syntax to promote a net to a spine/local clock:

```
assign_local_clock -net netname -type [quadrant|chip] Tn|Bn|Tn:Bm
```

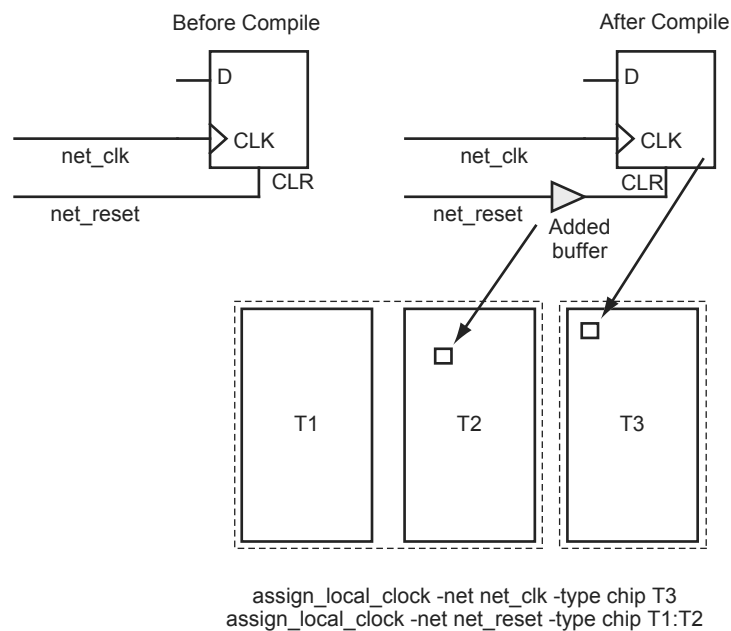
If the net is driven by a clock macro, Designer automatically demotes the clock net to a regular net before it is assigned to a spine. Nets driven by a PLL or CLKDLY macro cannot be assigned to a local clock.

When assigning a signal to a spine or quadrant global network using PDC (pre-compile), the Designer software will legalize the shared instances. The number of shared instances to be legalized can be controlled by compile options. If these networks are created in MVN (only quadrant globals can be created), no legalization is done (as it is post-compile). Designer does not do legalization between non-clock nets.

As an example, consider two nets, net\_clk and net\_reset, driving the same flip-flop. The following PDC constraints are used:

```
assign_local_clock -net net_clk -type chip T3
assign_local_clock -net net_reset -type chip T1:T2
```

During Compile, Designer adds a buffer in the reset net and places it in the T1 or T2 region, and places the flip-flop in the T3 spine region (Figure 3-16).



**Figure 3-16 • Adding a Buffer for Shared Instances**

## Global Buffers with No Programmable Delays

Access to the global / quadrant global networks can be configured directly from the global I/O buffer, bypassing the CCC functional block (as indicated by the dotted lines in Figure 4-1 on page 61). Internal signals driven by the FPGA core can use the global / quadrant global networks by connecting via the routed clock input of the multiplexer tree.

There are many specific CLKBUF macros supporting the wide variety of single-ended I/O inputs (CLKBUF) and differential I/O standards (CLKBUF\_LVDS/LVPECL) in the low power flash families. They are used when connecting global I/Os directly to the global/quadrant networks.

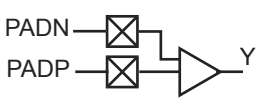
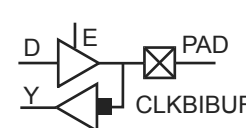
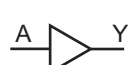

Note: IGLOO nano and ProASIC nano devices do not support differential inputs.

When an internal signal needs to be connected to the global/quadrant network, the CLKINT macro is used to connect the signal to the routed clock input of the network's MUX tree.

To utilize direct connection from global I/Os or from internal signals to the global/quadrant networks, CLKBUF, CLKBUF\_LVPECL/LVDS, and CLKINT macros are used (Figure 4-2).

- The CLKBUF and CLKBUF\_LVPECL/LVDS<sup>1</sup> macros are composite macros that include an I/O macro driving a global buffer, which uses a hardwired connection.
- The CLKBUF, CLKBUF\_LVPECL/LVDS<sup>1</sup> and CLKINT macros are pass-through clock sources and do not use the PLL or provide any programmable delay functionality.
- The CLKINT macro provides a global buffer function driven internally by the FPGA core.

The available CLKBUF macros are described in the *IGLOO, ProASIC3, SmartFusion, and Fusion Macro Library Guide*.

Clock Source		Clock Conditioning	Output
<p>CLKBUF_LVDS/LVPECL Macro</p> 	<p>CLKBIBUF Macro</p> 	None	GLA, GLB, or GLC
<p>CLKINT Macro</p> 	<p>CLKBUF Macro</p> 		

Note: IGLOO nano and ProASIC nano devices do not support differential inputs.

**Figure 4-2 • CCC Options: Global Buffers with No Programmable Delay**

## Global Buffer with Programmable Delay

Clocks requiring clock adjustments can utilize the programmable delay cores before connecting to the global / quadrant global networks. A maximum of 18 CCC global buffers can be instantiated in a device—three per CCC and up to six CCCs per device.

Each CCC functional block contains a programmable delay element for each of the global networks (up to three), and users can utilize these features by using the corresponding macro (Figure 4-3 on page 65).

1. B-LVDS and M-LVDS are supported with the LVDS macro.



## Phase Adjustment

The four phases available (0, 90, 180, 270) are phases with respect to VCO (PLL output). The VCO is divided to achieve the user's CCC required output frequency (GLA, YB/GLB, YC/GLC). The division happens after the selection of the VCO phase. The effective phase shift is actually the VCO phase shift divided by the output divider. This is why the visual CCC shows both the actual achievable phase and more importantly the actual delay that is equivalent to the phase shift that can be achieved.

## Dynamic PLL Configuration

The CCCs can be configured both statically and dynamically.

In addition to the ports available in the Static CCC, the Dynamic CCC has the dynamic shift register signals that enable dynamic reconfiguration of the CCC. With the Dynamic CCC, the ports CLKB and CLKC are also exposed. All three clocks (CLKA, CLKB, and CLKC) can be configured independently.

The CCC block is fully configurable. The following two sources can act as the CCC configuration bits.

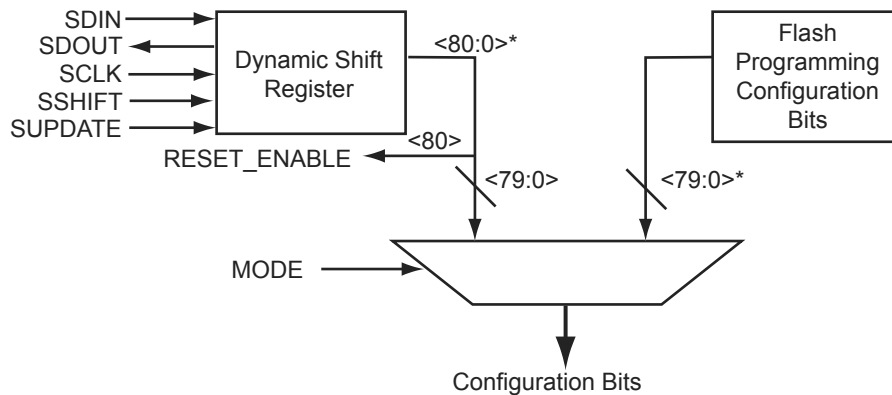
### Flash Configuration Bits

The flash configuration bits are the configuration bits associated with programmed flash switches. These bits are used when the CCC is in static configuration mode. Once the device is programmed, these bits cannot be modified. They provide the default operating state of the CCC.

### Dynamic Shift Register Outputs

This source does not require core reprogramming and allows core-driven dynamic CCC reconfiguration. When the dynamic register drives the configuration bits, the user-defined core circuit takes full control over SDIN, SDOUT, SCLK, SSHIFT, and SUPDATE. The configuration bits can consequently be dynamically changed through shift and update operations in the serial register interface. Access to the logic core is accomplished via the dynamic bits in the specific tiles assigned to the PLLs.

Figure 4-21 illustrates a simplified block diagram of the MUX architecture in the CCCs.



*Note: \*For Fusion, bit <88:81> is also needed.*

**Figure 4-21 • The CCC Configuration MUX Architecture**

The selection between the flash configuration bits and the bits from the configuration register is made using the MODE signal shown in Figure 4-21. If the MODE signal is logic HIGH, the dynamic shift register configuration bits are selected. There are 81 control bits to configure the different functions of the CCC.

When SmartGen is used to define the configuration that will be shifted in via the serial interface, SmartGen prints out the values of the 81 configuration bits. For ease of use, several configuration bits are automatically inferred by SmartGen when the dynamic PLL core is generated; however, <71:73> (STATASEL, STATBSEL, STATCSEL) and <77:79> (DYNASEL, DYNBSEL, DYNCSEL) depend on the input clock source of the corresponding CCC. Users must first run Layout in Designer to determine the exact setting for these ports. After Layout is complete, generate the "CCC\_Configuration" report by choosing **Tools > Reports > CCC\_Configuration** in the Designer software. Refer to "PLL Configuration Bits Description" on page 90 for descriptions of the PLL configuration bits. For simulation purposes, bits <71:73> and <78:80> are "don't care." Therefore, it is strongly suggested that SmartGen be used to generate the correct configuration bit settings for the dynamic PLL core.

After setting all the required parameters, users can generate one or more PLL configurations with HDL or EDIF descriptions by clicking the **Generate** button. SmartGen gives the option of saving session results and messages in a log file:

```
*****
Macro Parameters
*****

Name                : dyn_pll_hardio
Family              : ProASIC3E
Output Format        : VERILOG
Type                : Dynamic CCC
Input Freq(MHz)     : 30.000
CLKA Source         : Hardwired I/O
Feedback Delay Value Index : 1
Feedback Mux Select : 1
XDLY Mux Select     : No
Primary Freq(MHz)   : 33.000
Primary PhaseShift  : 0
Primary Delay Value Index : 1
Primary Mux Select  : 4
Secondary1 Freq(MHz) : 40.000
Use GLB             : YES
Use YB              : NO
GLB Delay Value Index : 1
YB Delay Value Index : 1
Secondary1 PhaseShift : 0
Secondary1 Mux Select : 0
Secondary1 Input Freq(MHz) : 40.000
CLKB Source         : Hardwired I/O
Secondary2 Freq(MHz) : 50.000
Use GLC             : YES
Use YC              : NO
GLC Delay Value Index : 1
YC Delay Value Index : 1
Secondary2 PhaseShift : 0
Secondary2 Mux Select : 0
Secondary2 Input Freq(MHz) : 50.000
CLKC Source         : Hardwired I/O

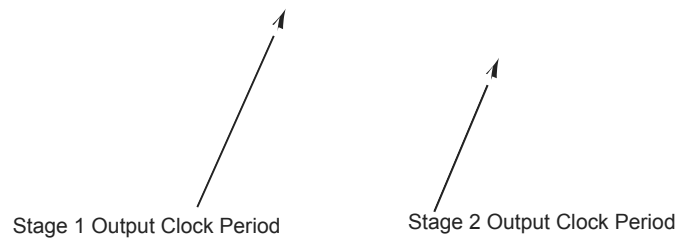
Configuration Bits:
FINDIV[6:0]         0000101
FBDIV[6:0]          0100000
OADIV[4:0]          00100
OBDIV[4:0]          00000
OCDIV[4:0]          00000
OAMUX[2:0]          100
OBMUX[2:0]          000
OCMUX[2:0]          000
FBSEL[1:0]          01
FBDLY[4:0]          00000
XDLYSEL             0
DLYGLA[4:0]         00000
DLYGLB[4:0]         00000
```

---

**Figure 4-36 • Second-Stage PLL Showing Input of 256 MHz from First Stage and Final Output of 280 MHz**

Figure 4-37 shows the simulation results, where the first PLL's output period is 3.9 ns (~256 MHz), and the stage 2 (final) output period is 3.56 ns (~280 MHz).

---



---

**Figure 4-37 • ModelSim Simulation Results**

## FlashROM Support in Flash-Based Devices

The flash FPGAs listed in Table 5-1 support the FlashROM feature and the functions described in this document.

**Table 5-1 • 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 5-1. 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 5-1. 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*.

## RD

This is the output data bus and is 18 bits wide. Not all 18 bits are valid in all configurations. Like the WD bus, high-order bits become unusable if the data width is less than 18. The output data on unused pins is undefined (Table 6-7).

**Table 6-7 • Input Data Signal Usage for Different Aspect Ratios**

D×W	WD/RD Unused
4k×1	WD[17:1], RD[17:1]
2k×2	WD[17:2], RD[17:2]
1k×4	WD[17:4], RD[17:4]
512×9	WD[17:9], RD[17:9]
256×18	—

## ESTOP, FSTOP

ESTOP is used to stop the FIFO read counter from further counting once the FIFO is empty (i.e., the EMPTY flag goes HIGH). A HIGH on this signal inhibits the counting.

FSTOP is used to stop the FIFO write counter from further counting once the FIFO is full (i.e., the FULL flag goes HIGH). A HIGH on this signal inhibits the counting.

For more information on these signals, refer to the "ESTOP and FSTOP Usage" section.

## FULL, EMPTY

When the FIFO is full and no more data can be written, the FULL flag asserts HIGH. The FULL flag is synchronous to WCLK to inhibit writing immediately upon detection of a full condition and to prevent overflows. Since the write address is compared to a resynchronized (and thus time-delayed) version of the read address, the FULL flag will remain asserted until two WCLK active edges after a read operation eliminates the full condition.

When the FIFO is empty and no more data can be read, the EMPTY flag asserts HIGH. The EMPTY flag is synchronous to RCLK to inhibit reading immediately upon detection of an empty condition and to prevent underflows. Since the read address is compared to a resynchronized (and thus time-delayed) version of the write address, the EMPTY flag will remain asserted until two RCLK active edges after a write operation removes the empty condition.

For more information on these signals, refer to the "FIFO Flag Usage Considerations" section on page 145.

## AFULL, AEMPTY

These are programmable flags and will be asserted on the threshold specified by AFVAL and AEVAL, respectively.

When the number of words stored in the FIFO reaches the amount specified by AEVAL while reading, the AEMPTY output will go HIGH. Likewise, when the number of words stored in the FIFO reaches the amount specified by AFVAL while writing, the AFULL output will go HIGH.

## AFVAL, AEVAL

The AEVAL and AFVAL pins are used to specify the almost-empty and almost-full threshold values. They are 12-bit signals. For more information on these signals, refer to the "FIFO Flag Usage Considerations" section on page 145.

## FIFO Usage

### ESTOP and FSTOP Usage

The ESTOP pin is used to stop the read counter from counting any further once the FIFO is empty (i.e., the EMPTY flag goes HIGH). Likewise, the FSTOP pin is used to stop the write counter from counting any further once the FIFO is full (i.e., the FULL flag goes HIGH).

The FIFO counters in the device start the count at zero, reach the maximum depth for the configuration (e.g., 511 for a 512×9 configuration), and then restart at zero. An example application for ESTOP, where the read counter keeps counting, would be writing to the FIFO once and reading the same content over and over without doing another write.

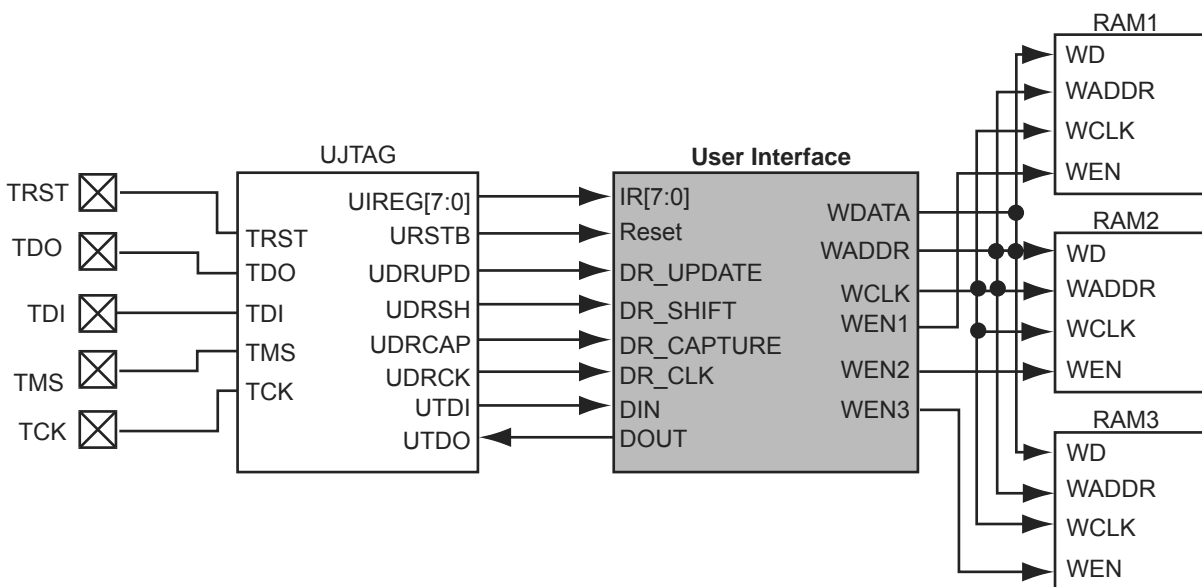
## Initializing the RAM/FIFO

The SRAM blocks can be initialized with data to use as a lookup table (LUT). Data initialization can be accomplished either by loading the data through the design logic or through the UJTAG interface. The UJTAG macro is used to allow access from the JTAG port to the internal logic in the device. By sending the appropriate initialization string to the JTAG Test Access Port (TAP) Controller, the designer can put the JTAG circuitry into a mode that allows the user to shift data into the array logic through the JTAG port using the UJTAG macro. For a more detailed explanation of the UJTAG macro, refer to the "FlashROM in Microsemi's Low Power Flash Devices" section on page 117.

A user interface is required to receive the user command, initialization data, and clock from the UJTAG macro. The interface must synchronize and load the data into the correct RAM block of the design. The main outputs of the user interface block are the following:

- Memory block chip select: Selects a memory block for initialization. The chip selects signals for each memory block that can be generated from different user-defined pockets or simple logic, such as a ring counter (see below).
- Memory block write address: Identifies the address of the memory cell that needs to be initialized.
- Memory block write data: The interface block receives the data serially from the UTDI port of the UJTAG macro and loads it in parallel into the write data ports of the memory blocks.
- Memory block write clock: Drives the WCLK of the memory block and synchronizes the write data, write address, and chip select signals.

Figure 6-8 shows the user interface between UJTAG and the memory blocks.



**Figure 6-8 • Interfacing TAP Ports and SRAM Blocks**

An important component of the interface between the UJTAG macro and the RAM blocks is a serial-in/parallel-out shift register. The width of the shift register should equal the data width of the RAM blocks. The RAM data arrives serially from the UTDI output of the UJTAG macro. The data must be shifted into a shift register clocked by the JTAG clock (provided at the UDRCK output of the UJTAG macro).

Then, after the shift register is fully loaded, the data must be transferred to the write data port of the RAM block. To synchronize the loading of the write data with the write address and write clock, the output of the shift register can be pipelined before driving the RAM block.

The write address can be generated in different ways. It can be imported through the TAP using a different instruction opcode and another shift register, or generated internally using a simple counter. Using a counter to generate the address bits and sweep through the address range of the RAM blocks is

## Features Supported on Every I/O

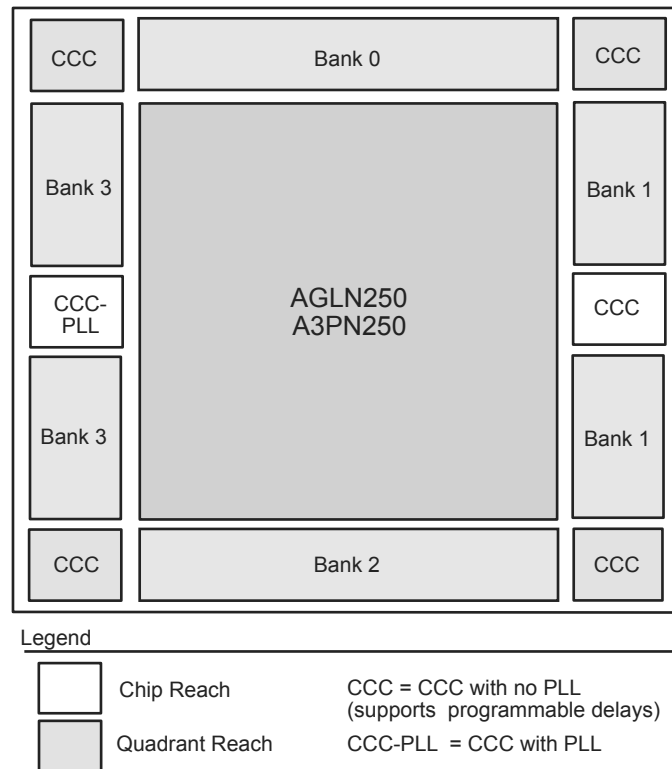
Table 7-5 lists all features supported by transmitter/receiver for single-ended I/Os. Table 7-6 lists the performance of each I/O technology.

**Table 7-5 • I/O Features**

Feature	Description
All I/O	<ul style="list-style-type: none"> <li>High performance (Table 7-6)</li> <li>Electrostatic discharge (ESD) protection</li> <li>I/O register combining option</li> </ul>
Single-Ended Transmitter Features	<ul style="list-style-type: none"> <li>Hot-swap</li> <li>I/Os can be configured to behave in Flash*Freeze mode as tristate, HIGH, LOW, or to hold the previous state (not supported on ProASIC3 nano devices).</li> <li>Programmable output slew rate: high and low</li> <li>Optional weak pull-up and pull-down resistors</li> <li>Output drive: 2 drive strengths (except for LVCMOS 1.2 V)</li> <li>LVTTL/LVCMOS 3.3 V outputs compatible with 5 V TTL inputs</li> </ul>
Single-Ended Receiver Features	<ul style="list-style-type: none"> <li>Selectable Schmitt trigger</li> <li>5 V–input–tolerant receiver (Table 7-12 on page 171)</li> <li>Separate ground plane for GNDQ pin and power plane for V<sub>CC1</sub> pin are used for input buffer to reduce output-induced noise.</li> </ul>
DDR	<ul style="list-style-type: none"> <li>DDR is supported for 60 k gate devices and above.</li> </ul>

**Table 7-6 • Maximum I/O Frequency**

Specification	Maximum Performance		
	ProASIC3 nano 1.5 V DC Core Supply Voltage	IGLOO nano V2 or V5 Devices, 1.5 V DC Core Supply Voltage	IGLOO nano 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
LVCMOS 1.2 V	Not supported	TBD	TBD



**Figure 7-12 • I/O Bank Architecture of AGLN250/A3PN250 Devices**

## Board-Level Considerations

Low power flash devices have robust I/O features that can help in reducing board-level components. The devices offer single-chip solutions, which makes the board layout simpler and more immune to signal integrity issues. Although, in many cases, these devices resolve board-level issues, special attention should always be given to overall signal integrity. This section covers important board-level considerations to facilitate optimum device performance.

### Termination

Proper termination of all signals is essential for good signal quality. Nonterminated signals, especially clock signals, can cause malfunctioning of the device.

For general termination guidelines, refer to the *Board-Level Considerations* application note for Microsemi FPGAs. Also refer to the "Pin Descriptions and Packaging" chapter of the appropriate device datasheet for termination requirements for specific pins.

Low power flash I/Os are equipped with on-chip pull-up/-down resistors. The user can enable these resistors by instantiating them either in the top level of the design (refer to the *IGLOO*, *ProASIC3*, *SmartFusion*, and *Fusion Macro Library Guide* for the available I/O macros with pull-up/-down) or in the I/O Attribute Editor in Designer if generic input or output buffers are instantiated in the top level. Unused I/O pins are configured as inputs with pull-up resistors.

As mentioned earlier, low power flash devices have multiple programmable drive strengths, and the user can eliminate unwanted overshoot and undershoot by adjusting the drive strengths.



## Software-Controlled I/O Attributes

Users may modify these programmable I/O attributes using the I/O Attribute Editor. Modifying an I/O attribute may result in a change of state in Designer. Table 8-2 details which steps have to be re-run as a function of modified I/O attribute.

**Table 8-2 • Designer State (resulting from I/O attribute modification)**

I/O Attribute	Designer States <sup>1</sup>				
	Compile	Layout	Fuse	Timing	Power
Slew Control <sup>2</sup>	No	No	Yes	Yes	Yes
Output Drive (mA)	No	No	Yes	Yes	Yes
Skew Control	No	No	Yes	Yes	Yes
Resistor Pull	No	No	Yes	Yes	Yes
Input Delay	No	No	Yes	Yes	Yes
Schmitt Trigger	No	No	Yes	Yes	Yes
OUT_LOAD	No	No	No	Yes	Yes
COMBINE_REGISTER	Yes	Yes	N/A	N/A	N/A

Notes:

1. No = Remains the same, Yes = Re-run the step, N/A = Not applicable
2. Skew control does not apply to IGLOO nano, IGLOO PLUS, and ProASIC3 nano devices.
3. Programmable input delay is applicable only for ProASIC3E, ProASIC3EL, RT ProASIC3, and IGLOOe devices.

- The I/O standard of technology-specific I/O macros cannot be changed in the I/O Attribute Editor (see Figure 8-6).
- 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 8-6 • 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 8-3 shows I/O assignment constraints supported in the PDC file.

**Table 8-3 • PDC I/O Constraints**

Command	Action	Example	Comment
<b>I/O Banks Setting Constraints</b>			
set_iobank	Sets the I/O supply voltage, $V_{CCI}$ , and the input reference voltage, $V_{REF}$ , for the specified I/O bank.	<pre>set_iobank bankname [-vcci vcci_voltage] [-vref vref_voltage]  set_iobank Bank7 -vcci 1.50 -vref 0.75</pre>	Must use in case of mixed I/O voltage ( $V_{CCI}$ ) design
set_vref	Assigns a $V_{REF}$ pin to a bank.	<pre>set_vref -bank [bankname] [pinnum]  set_vref -bank Bank0 685 704 723 742 761</pre>	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.	<pre>set_vref_defaults bankname  set_vref_defaults bank2</pre>	

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

VREF for GTL+ 3.3 V



---

**Figure 8-13 • Selecting VREF Voltage for the I/O Bank**

## Assigning VREF Pins for a Bank

The user can use default pins for VREF. In this case, select the **Use default pins for VREFs** check box (Figure 8-13). This option guarantees full VREF coverage of the bank. The equivalent PDC command is as follows:

```
set_vref_default [bank name]
```

To be able to choose VREF pins, adequate VREF pins must be created to allow legal placement of the compatible voltage-referenced I/Os.

To assign VREF pins manually, the PDC command is as follows:

```
set_vref -bank [bank name] [package pin numbers]
```

For ChipPlanner/PinEditor to show the range of a VREF pin, perform the following steps:

1. Assign VCCI to a bank using **MVN > Edit > I/O Bank Settings**.
2. Open **ChipPlanner**. Zoom in on an I/O package pin in that bank.
3. Highlight the pin and then right-click. Choose **Use Pin for VREF**.

**Table 9-2 • DDR I/O Options (continued)**

DDR Register Type	I/O Type	I/O Standard	Sub-Options	Comments
Transmit Register (continued)	Tristate Buffer	Normal	Enable Polarity	Low/high (low default)
		LVTTTL	Output Drive	2, 4, 6, 8, 12, 16, 24, 36 mA (8 mA default)
			Slew Rate	Low/high (high default)
			Enable Polarity	Low/high (low default)
			Pull-Up/-Down	None (default)
		LVCMOS	Voltage	1.5 V, 1.8 V, 2.5 V, 5 V (1.5 V default)
			Output Drive	2, 4, 6, 8, 12, 16, 24, 36 mA (8 mA default)
			Slew Rate	Low/high (high default)
			Enable Polarity	Low/high (low default)
			Pull-Up/-Down	None (default)
		PCI/PCI-X	Enable Polarity	Low/high (low default)
		GTL/GTL+	Voltage	1.8 V, 2.5 V, 3.3 V (3.3 V default)
			Enable Polarity	Low/high (low default)
		HSTL	Class	I / II (I default)
			Enable Polarity	Low/high (low default)
		SSTL2/SSTL3	Class	I / II (I default)
			Enable Polarity	Low/high (low default)
	Bidirectional Buffer	Normal	Enable Polarity	Low/high (low default)
		LVTTTL	Output Drive	2, 4, 6, 8, 12, 16, 24, 36 mA (8 mA default)
			Slew Rate	Low/high (high default)
			Enable Polarity	Low/high (low default)
			Pull-Up/-Down	None (default)
		LVCMOS	Voltage	1.5 V, 1.8 V, 2.5 V, 5 V (1.5 V default)
			Enable Polarity	Low/high (low default)
			Pull-Up	None (default)
		PCI/PCI-X	None	
			Enable Polarity	Low/high (low default)
		GTL/GTL+	Voltage	1.8 V, 2.5 V, 3.3 V (3.3 V default)
			Enable Polarity	Low/high (low default)
		HSTL	Class	I / II (I default)
			Enable Polarity	Low/high (low default)
		SSTL2/SSTL3	Class	I / II (I default)
			Enable Polarity	Low/high (low default)

Note: \*IGLOO nano and ProASIC3 nano devices do not support differential inputs.

## Security in ARM-Enabled Low Power Flash Devices

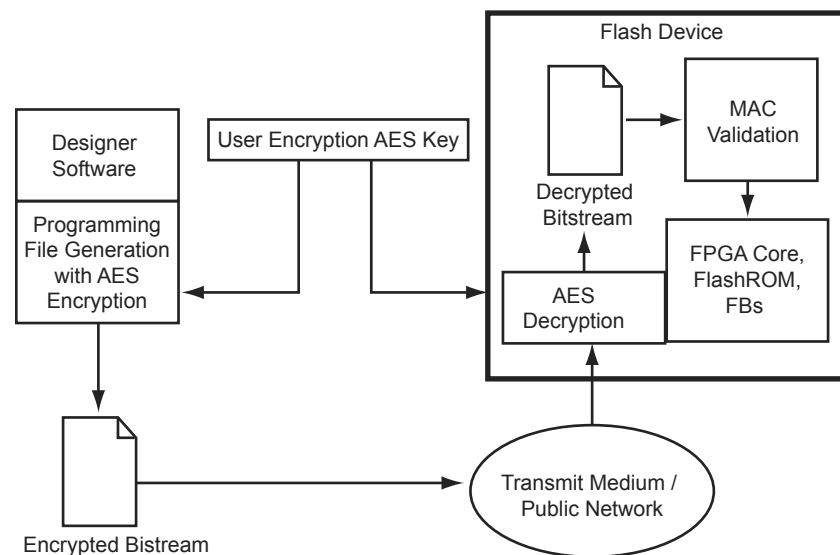
There are slight differences between the regular flash device and the ARM-enabled flash devices, which have the M1 prefix.

The AES key is used by Microsemi and preprogrammed into the device to protect the ARM IP. As a result, the design will be encrypted along with the ARM IP, according to the details below.

### Cortex-M1 and Cortex-M3 Device Security

Cortex-M1-enabled and Cortex-M3 devices are shipped with the following security features:

- FPGA array enabled for AES-encrypted programming and verification
- FlashROM enabled for AES-encrypted write and verify
- Embedded Flash Memory enabled for AES encrypted write



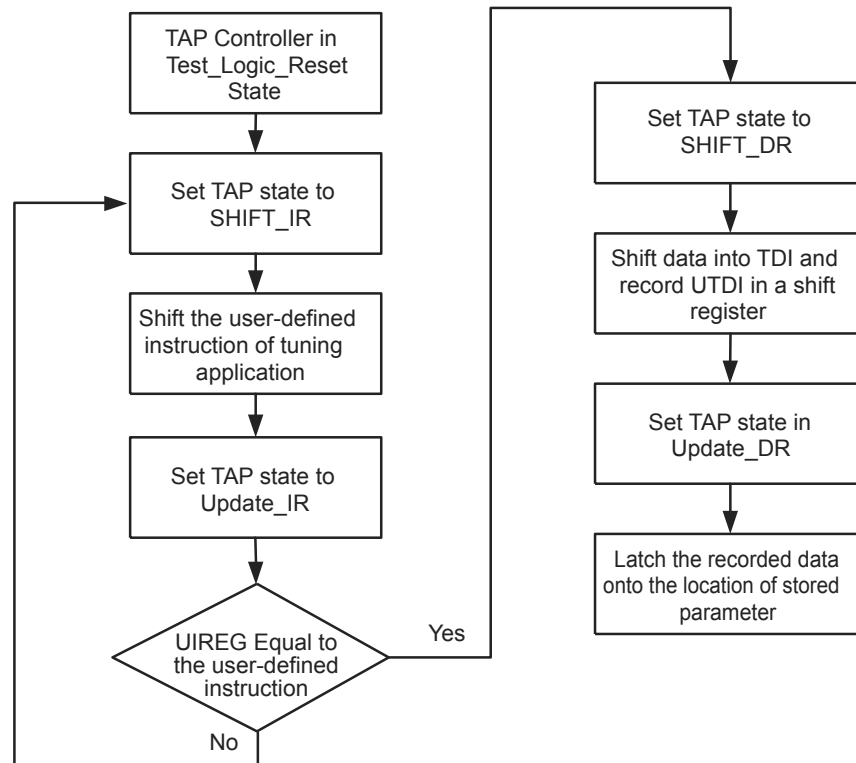
**Figure 12-1 • AES-128 Security Features**



## Fine Tuning

In some applications, design constants or parameters need to be modified after programming the original design. The tuning process can be done using the UJTAG tile without reprogramming the device with new values. If the parameters or constants of a design are stored in distributed registers or embedded SRAM blocks, the new values can be shifted onto the JTAG TAP Controller pins, replacing the old values. The UJTAG tile is used as the “bridge” for data transfer between the JTAG pins and the FPGA VersaTiles or SRAM logic. Figure 16-5 shows a flow chart example for fine-tuning application steps using the UJTAG tile.

In Figure 16-5, the TMS signal sets the TAP Controller state machine to the appropriate states. The flow mainly consists of two steps: a) shifting the defined instruction and b) shifting the new data. If the target parameter is constantly used in the design, the new data can be shifted into a temporary shift register from UTDI. The UDRSH output of UJTAG can be used as a shift-enable signal, and UDRCK is the shift clock to the shift register. Once the shift process is completed and the TAP Controller state is moved to the Update\_DR state, the UDRUPD output of the UJTAG can latch the new parameter value from the temporary register into a permanent location. This avoids any interruption or malfunctioning during the serial shift of the new value.



**Figure 16-5 • Flow Chart Example of Fine-Tuning an Application Using UJTAG**

## **B – Product Support**

---

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group 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 North America, call 800.262.1060

From the rest of the world, call 650.318.4460

Fax, from anywhere in the world, 650.318.8044

### **Customer Technical Support Center**

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues, and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

### **Technical Support**

Visit the Customer Support website ([www.microsemi.com/soc/support/search/default.aspx](http://www.microsemi.com/soc/support/search/default.aspx)) for more information and support. Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on the website.

### **Website**

You can browse a variety of technical and non-technical information on the SoC home page, at [www.microsemi.com/soc](http://www.microsemi.com/soc).

### **Contacting the Customer Technical Support Center**

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

#### **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 [soc\\_tech@microsemi.com](mailto:soc_tech@microsemi.com).



## My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.

## Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email ([soc\\_tech@microsemi.com](mailto:soc_tech@microsemi.com)) or contact a local sales office. Sales office listings can be found at [www.microsemi.com/soc/company/contact/default.aspx](http://www.microsemi.com/soc/company/contact/default.aspx).

## ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via [soc\\_tech\\_itar@microsemi.com](mailto:soc_tech_itar@microsemi.com). Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.