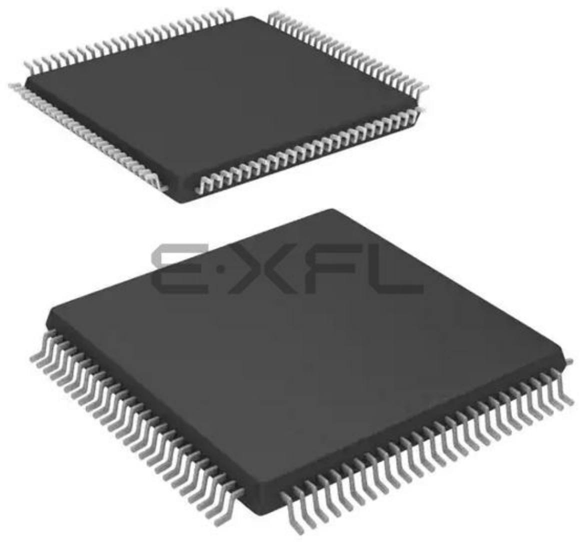
Microchip Technology - A3PN125-ZVQG100I Datasheet



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	Active
Number of LABs/CLBs	-
Number of Logic Elements/Cells	-
Total RAM Bits	36864
Number of I/O	71
Number of Gates	125000
Voltage - Supply	1.425V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 100°C (TJ)
Package / Case	100-TQFP
Supplier Device Package	100-VQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/a3pn125-zvqg100i

Microsemi

2 – Low Power Modes in ProASIC3/E and ProASIC3 nano FPGAs

Introduction

The demand for low power systems and semiconductors, combined with the strong growth observed for value-based FPGAs, is driving growing demand for low power FPGAs. For portable and battery-operated applications, power consumption has always been the greatest challenge. The battery life of a system and on-board devices has a direct impact on the success of the product. As a result, FPGAs used in these applications should meet low power consumption requirements.

ProASIC®3/E and ProASIC3 nano FPGAs offer low power consumption capability inherited from their nonvolatile and live-at-power-up (LAPU) flash technology. This application note describes the power consumption and how to use different power saving modes to further reduce power consumption for power-conscious electronics design.

Power Consumption Overview

In evaluating the power consumption of FPGA technologies, it is important to consider it from a system point of view. Generally, the overall power consumption should be based on static, dynamic, inrush, and configuration power. Few FPGAs implement ways to reduce static power consumption utilizing sleep modes.

SRAM-based FPGAs use volatile memory for their configuration, so the device must be reconfigured after each power-up cycle. Moreover, during this initialization state, the logic could be in an indeterminate state, which might cause inrush current and power spikes. More complex power supplies are required to eliminate potential system power-up failures, resulting in higher costs. For portable electronics requiring frequent power-up and -down cycles, this directly affects battery life, requiring more frequent recharging or replacement.

$$\text{SRAM-Based FPGA Total Power Consumption} = P_{static} + P_{dynamic} + P_{inrush} + P_{config} \quad \textit{EQ 1}$$

$$\text{ProASIC3/E Total Power Consumption} = P_{static} + P_{dynamic} \quad \textit{EQ 2}$$

Unlike SRAM-based FPGAs, Microsemi flash-based FPGAs are nonvolatile and do not require power-up configuration. Additionally, Microsemi nonvolatile flash FPGAs are live at power-up and do not require additional support components. Total power consumption is reduced as the inrush current and configuration power components are eliminated.

Note that the static power component can be reduced in flash FPGAs (such as the ProASIC3/E devices) by entering User Low Static mode or Sleep mode. This leads to an extremely low static power component contribution to the total system power consumption.

The following sections describe the usage of Static (Idle) mode to reduce the power component, User Low Static mode to reduce the static power component, and Sleep mode and Shutdown mode to achieve a range of power consumption when the FPGA or system is idle. Table 2-1 on page 22 summarizes the different low power modes offered by ProASIC3/E devices.

Table 2-2 **• Using ULSICC Macro***

VHDL	Verilog

`COMPONENT ULSICC`
`       port (`
`             LSICC         : in     STD_ULOGIC);`
`END COMPONENT;`
 	
Example:
`COMPONENT ULSICC`
`       port (`
`             LSICC         : in     STD_ULOGIC);`
`END COMPONENT;`
 	
`attribute syn_noprune : boolean;`
`attribute syn_noprune of u1 : label is true;`
`u1: ULSICC port map(myInputSignal);`
	`module ULSICC(LSICC);`
	` input LSICC;`
	`endmodule`
	
	Example:
	`ULSICC U1(.LSICC(myInputSignal))`
	`/* synthesis syn_noprune=1 */;`

*Note: *Supported in Libero® software v7.2 and newer versions.*

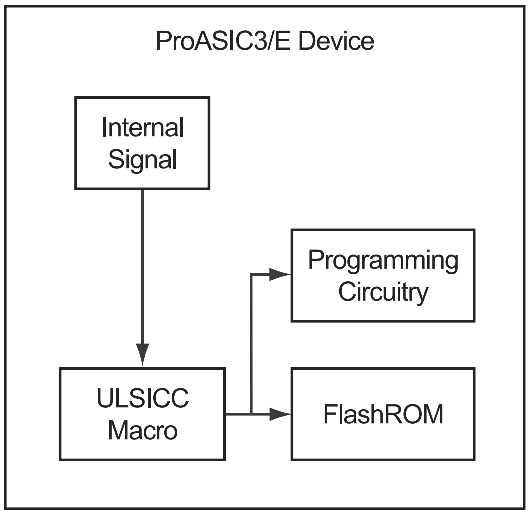


Figure 2-2 **• User Low Static (Idle) Mode Application—Internal Control Signal**

Figure 3-6 shows all nine global inputs for the location A connected to the top left quadrant global network via CCC.

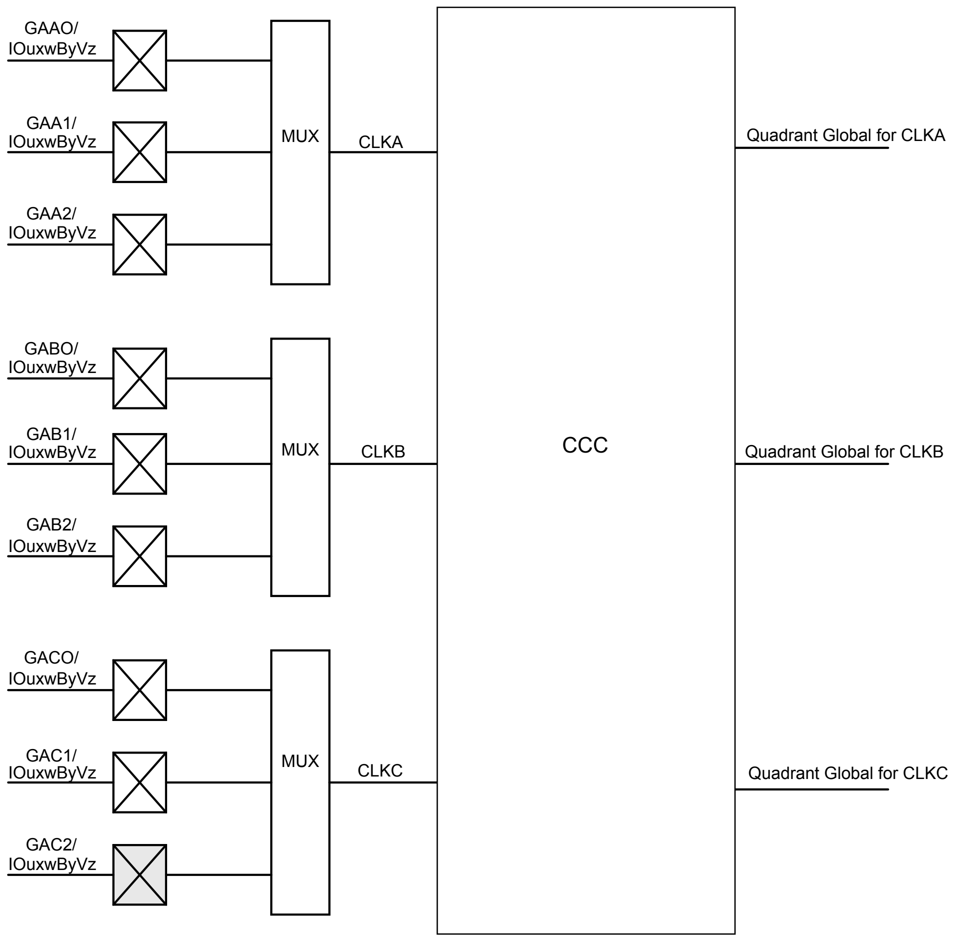


Figure 3-6 **• Global Inputs**

Since each bank can have a different I/O standard, the user should be careful to choose the correct global I/O for the design. There are 54 global pins available to access 18 global networks. For the single-ended and voltage-referenced I/O standards, you can use any of these three available I/Os to access the global network. For differential I/O standards such as LVDS and LVPECL, the I/O macro needs to be placed on (A0, A1), (B0, B1), (C0, C1), or a similar location. The unassigned global I/Os can be used as regular I/Os. Note that pin names starting with GF and GC are associated with the chip global networks, and GA, GB, GD, and GE are used for quadrant global networks. Table 3-2 on page 38 and Table 3-3 on page 39 show the general chip and quadrant global pin names.

Table 3-3 • **Quadrant Global Pin Name (continued)**

Differential I/O Pairs	GAAO/IOuxwByVz GAA1/IOuxwByVz	The output of the different pair will drive the global.
	GABO/IOuxwByVz GAB1/IOuxwByVz	The output of the different pair will drive the global.
	GACO/IOuxwByVz GAC1/IOuxwByVz	The output of the different pair will drive the global.
	GBAO/IOuxwByVz GBA1/IOuxwByVz	The output of the different pair will drive the global.
	GBBO/IOuxwByVz GBB1/IOuxwByVz	The output of the different pair will drive the global.
	GBCO/IOuxwByVz GBC1/IOuxwByVz	The output of the different pair will drive the global.
	GDAO/IOuxwByVz GDA1/IOuxwByVz	The output of the different pair will drive the global.
	GDBO/IOuxwByVz GDB1/IOuxwByVz	The output of the different pair will drive the global.
	GDCO/IOuxwByVz GDC1/IOuxwByVz	The output of the different pair will drive the global.
	GEAO/IOuxwByVz GEA1/IOuxwByVz	The output of the different pair will drive the global.
	GEBO/IOuxwByVz GEB1/IOuxwByVz	The output of the different pair will drive the global.
	GECO/IOuxwByVz GEC1/IOuxwByVz	The output of the different pair will drive the global.

Note: Only one of the I/Os can be directly connected to a quadrant at a time.

Unused Global I/O Configuration

The unused clock inputs behave similarly to the unused Pro I/Os. The Microsemi Designer software automatically configures the unused global pins as inputs with pull-up resistors if they are not used as regular I/O.

I/O Banks and Global I/O Standards

In low power flash devices, any I/O or internal logic can be used to drive the global network. However, only the global macro placed at the global pins will use the hardwired connection between the I/O and global network. Global signal (signal driving a global macro) assignment to I/O banks is no different from regular I/O assignment to I/O banks with the exception that you are limited to the pin placement location available. Only global signals compatible with both the VCCI and VREF standards can be assigned to the same bank.

Spine Access

The physical location of each spine is identified by the letter T (top) or B (bottom) and an accompanying number (T*n* or B*n*). The number *n* indicates the horizontal location of the spine; 1 refers to the first spine on the left side of the die. Since there are six chip spines in each spine tree, there are up to six spines available for each combination of T (or B) and *n* (for example, six T1 spines). Similarly, there are three quadrant spines available for each combination of T (or B) and *n* (for example, four T1 spines), as shown in Figure 3-7.

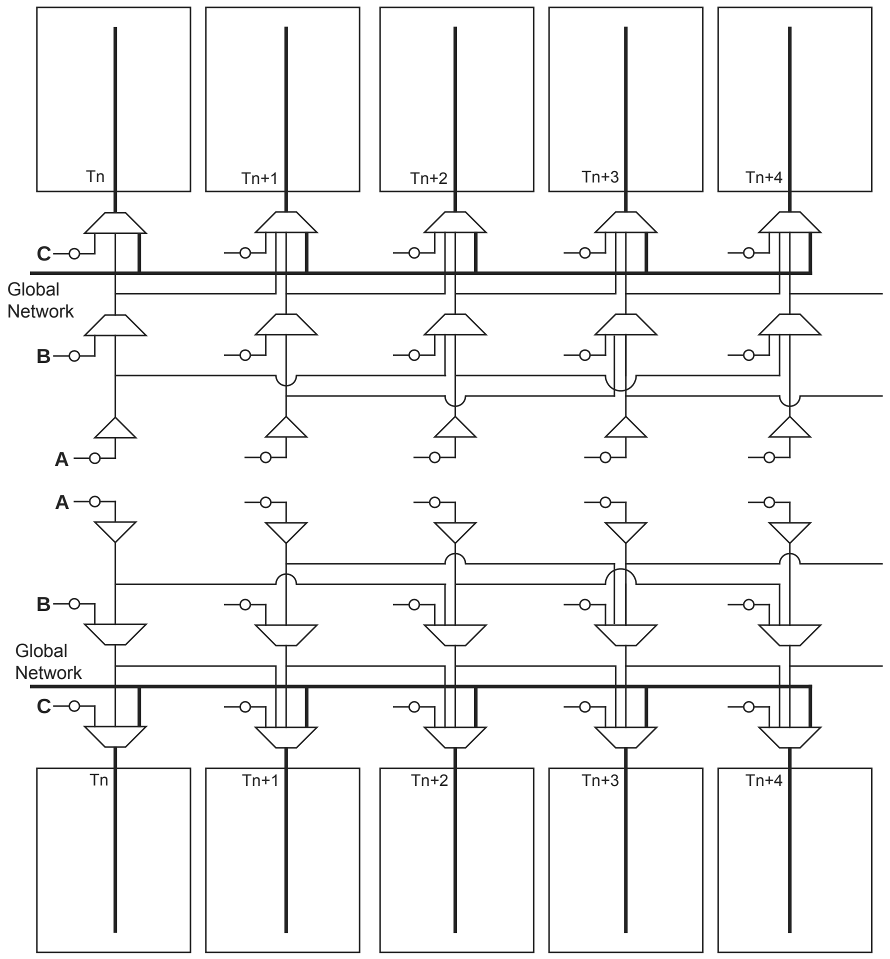


Figure 3-7 **• Chip Global Aggregation**

A spine is also called a local clock network, and is accessed by the dedicated global MUX architecture. These MUXes define how a particular spine is driven. Refer to Figure 3-8 on page 44 for the global MUX architecture. The MUXes for each chip global spine are located in the middle of the die. Access to the top and bottom chip global spine is available from the middle of the die. There is no control dependency between the top and bottom spines. If a top spine, T1, of a chip global network is assigned to a net, B1 is not wasted and can be used by the global clock network. The signal assigned only to the top or bottom spine cannot access the middle two rows of the architecture. However, if a spine is using the top and bottom at the same time (T1 and B1, for instance), the previous restriction is lifted.

The MUXes for each quadrant global spine are located in the north and south sides of the die. Access to the top and bottom quadrant global spines is available from the north and south sides of the die. Since the MUXes for quadrant spines are located in the north and south sides of the die, you should not try to drive T1 and B1 quadrant spines from the same signal.

Fusion CCC Locations

Fusion devices have six CCCs: one in each of the four corners and one each in the middle of the east and west sides of the device (Figure 4-17 and Figure 4-18). The device can have one integrated PLL in the middle of the west side of the device or two integrated PLLs in the middle of the east and west sides of the device (middle right and middle left).

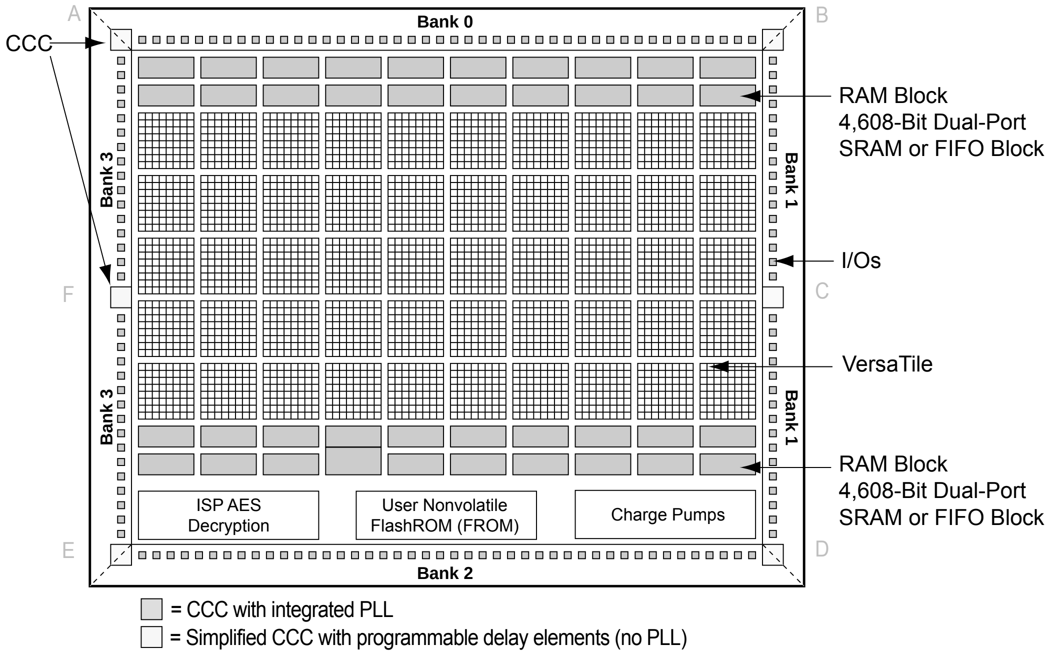

A
B
C
D
E
F
CCC
Bank 0
Bank 1
Bank 2
Bank 3
RAM Block 4,608-Bit Dual-Port SRAM or FIFO Block
I/Os
VersaTile
ISP AES Decryption
User Nonvolatile FlashROM (FROM)
Charge Pumps
= CCC with integrated PLL
= Simplified CCC with programmable delay elements (no PLL)

***Figure 4-17* • CCC Locations in Fusion Family Devices (AFS090, AFS250, M1AFS250)**

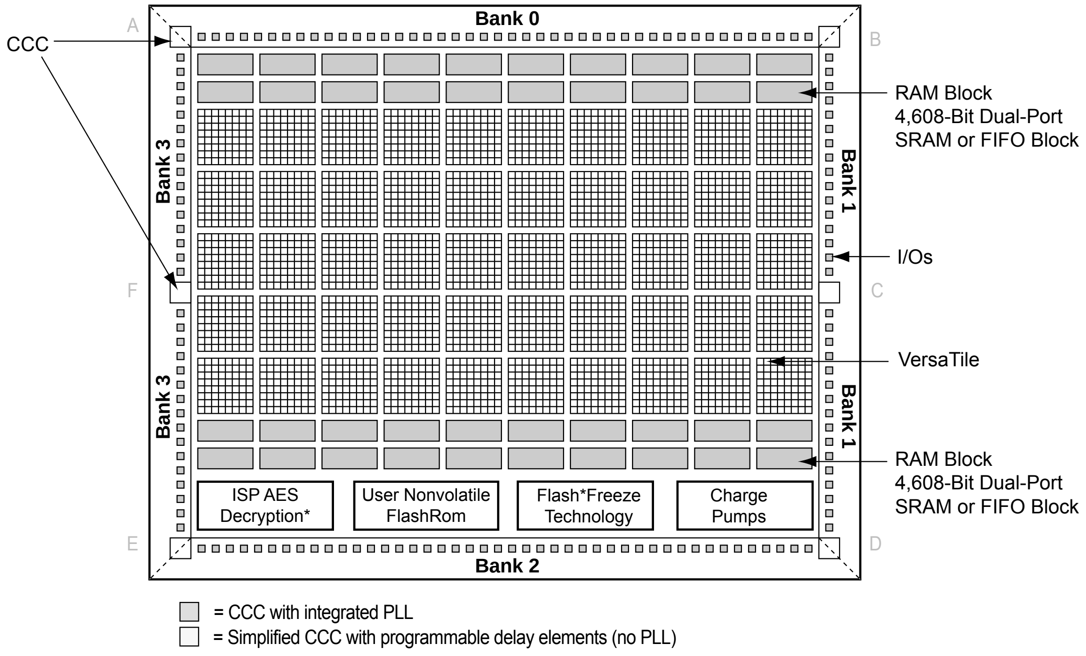

A
B
C
D
E
F
CCC
Bank 0
Bank 1
Bank 2
Bank 3
RAM Block 4,608-Bit Dual-Port SRAM or FIFO Block
I/Os
VersaTile
ISP AES Decryption*
User Nonvolatile FlashRom
Flash*Freeze Technology
Charge Pumps
= CCC with integrated PLL
= Simplified CCC with programmable delay elements (no PLL)

***Figure 4-18* • CCC Locations in Fusion Family Devices (except AFS090, AFS250, M1AFS250)**

Programming and Accessing FlashROM

The FlashROM content can only be programmed via JTAG, but it can be read back selectively through the JTAG programming interface, the UJTAG interface, or via direct FPGA core addressing. The pages of the FlashROM can be made secure to prevent read-back via JTAG. In that case, read-back on these secured pages is only possible by the FPGA core fabric or via UJTAG.

A 7-bit address from the FPGA core defines which of the eight pages (three MSBs) is being read, and which of the 16 bytes within the selected page (four LSBs) are being read. The FlashROM content can be read on a random basis; the access time is 10 ns for a device supporting commercial specifications. The FPGA core will be powered down during writing of the FlashROM content. FPGA power-down during FlashROM programming is managed on-chip, and FPGA core functionality is not available during programming of the FlashROM. Table 5-2 summarizes various FlashROM access scenarios.

***Table 5-2* • FlashROM Read/Write Capabilities by Access Mode**

Access Mode	FlashROM Read	FlashROM Write
JTAG	Yes	Yes
UJTAG	Yes	No
FPGA core	Yes	No

Figure 5-6 shows the accessing of the FlashROM using the UJTAG macro. This is similar to FPGA core access, where the 7-bit address defines which of the eight pages (three MSBs) is being read and which of the 16 bytes within the selected page (four LSBs) are being read. Refer to the "UJTAG Applications in Microsemi's Low Power Flash Devices" section on page 297 for details on using the UJTAG macro to read the FlashROM.

Figure 5-7 on page 123 and Figure 5-8 on page 123 show the FlashROM access from the JTAG port. The FlashROM content can be read on a random basis. The three-bit address defines which page is being read or updated.

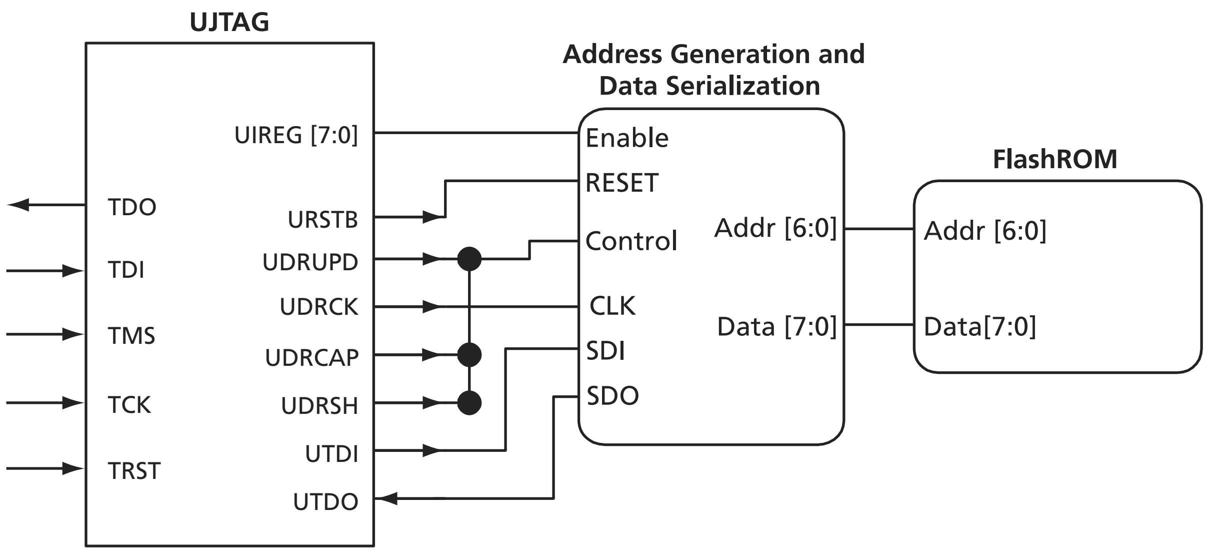


***Figure 5-6* • Block Diagram of Using UJTAG to Read FlashROM Contents**

7 – I/O Structures in nano Devices

Introduction

Low power flash devices feature a flexible I/O structure, supporting a range of mixed voltages (1.2 V, 1.5 V, 1.8 V, 2.5 V, and 3.3 V) through bank-selectable voltages. IGLOO® and ProASIC3 nano devices support standard I/Os with the addition of Schmitt trigger and hot-swap capability.

Users designing I/O solutions are faced with a number of implementation decisions and configuration choices that can directly impact the efficiency and effectiveness of their final design. The flexible I/O structure, supporting a wide variety of voltages and I/O standards, enables users to meet the growing challenges of their many diverse applications. The Microsemi Libero® System-on-Chip (SoC) software provides an easy way to implement I/O that will result in robust I/O design.

This document describes Standard I/O types used for the nano devices in terms of he supported standards. It then explains the individual features and how to implement them in Libero SoC.

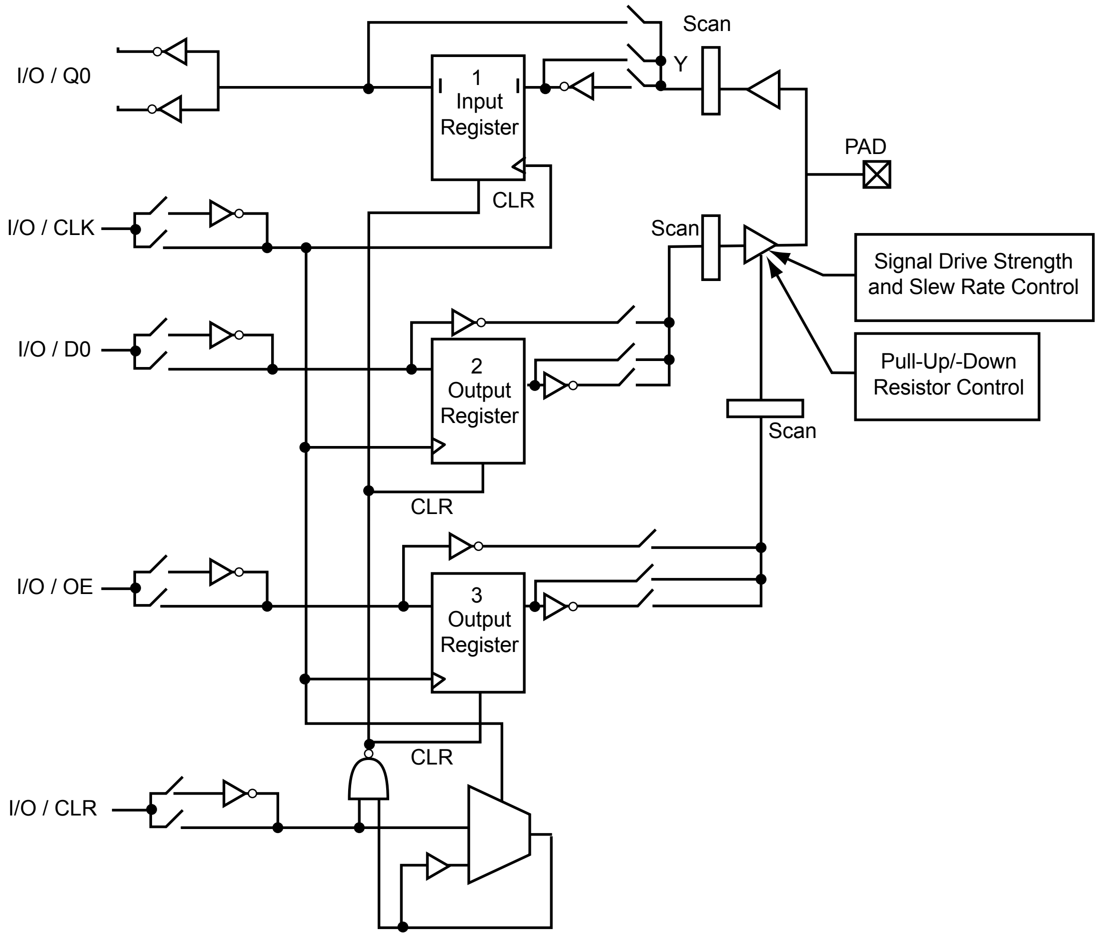


Figure 7-1 • **I/O Block Logical Representation for Single-Tile Designs (10 k, 15 k, and 20 k devices)**

Table 7-13 **• Comparison Table for 5 V–Compliant Receiver Solutions**

Solution	Board Components	Speed	Current Limitations
1	Two resistors	Low to High[1]	Limited by transmitter's drive strength
2	Resistor and Zener 3.3 V	Medium	Limited by transmitter's drive strength
3	Bus switch	High	N/A

Notes:

1. *Speed and current consumption increase as the board resistance values decrease.*
2. *Resistor values ensure I/O diode long-term reliability.*
3. *At 70°C, customers could still use 420 Ω on every I/O.*
4. *At 85°C, a 5 V solution on every other I/O is permitted, since the resistance is lower (150 Ω) and the current is higher. Also, the designer can still use 420 Ω and use the solution on every I/O.*
5. *At 100°C, the 5 V solution on every I/O is permitted, since 420 Ω are used to limit the current to 5.9 mA.*

5 V Output Tolerance

nano Standard I/Os must be set to 3.3 V LVTTL or 3.3 V LVCMOS mode to reliably drive 5 V TTL receivers. It is also critical that there be NO external I/O pull-up resistor to 5 V, since this resistor would pull the I/O pad voltage beyond the 3.6 V absolute maximum value and consequently cause damage to the I/O.

When set to 3.3 V LVTTL or 3.3 V LVCMOS mode, the I/Os can directly drive signals into 5 V TTL receivers. In fact, V_{OL} = 0.4 V and V_{OH} = 2.4 V in both 3.3 V LVTTL and 3.3 V LVCMOS modes exceeds the V_{IL} = 0.8 V and V_{IH} = 2 V level requirements of 5 V TTL receivers. Therefore, level 1 and level 0 will be recognized correctly by 5 V TTL receivers.

Schmitt Trigger

A Schmitt trigger is a buffer used to convert a slow or noisy input signal into a clean one before passing it to the FPGA. Using Schmitt trigger buffers guarantees a fast, noise-free input signal to the FPGA.

nano devices have Schmitt triggers built into their I/O circuitry. Schmitt Trigger is available on all I/O configurations.

This feature can be implemented by using a Physical Design Constraints (PDC) command (Table 7-5 on page 163) or by selecting a check box in the I/O Attribute Editor in Designer. The check box is cleared by default.

I/O Register Combining

Every I/O has several embedded registers in the I/O tile that are close to the I/O pads. Rather than using the internal register from the core, the user has the option of using these registers for faster clock-to-out timing, and external hold and setup. When combining these registers at the I/O buffer, some architectural rules must be met. Provided these rules are met, the user can enable register combining globally during Compile (as shown in the "Compiling the Design" section in the "I/O Software Control in Low Power Flash Devices" section on page 185.

This feature is supported by all I/O standards.

Rules for Registered I/O Function:

1. The fanout between an I/O pin (D, Y, or E) and a register must be equal to one for combining to be considered on that pin.
2. All registers (Input, Output, and Output Enable) connected to an I/O must share the same clear or preset function:
   – If one of the registers has a CLR pin, all the other registers that are candidates for combining in the I/O must have a CLR pin.

   - If one of the registers has a PRE pin, all the other registers that are candidates for combining in the I/O must have a PRE pin.
   - If one of the registers has neither a CLR nor a PRE pin, all the other registers that are candidates for combining must have neither a CLR nor a PRE pin.
   - If the clear or preset pins are present, they must have the same polarity.
   - If the clear or preset pins are present, they must be driven by the same signal (net).
3. For single-tile devices (10 k, 15 k, and 20 k): Registers connected to an I/O on the Output and Output Enable pins must have the same clock function (both CLR and CLK are shared among all registers):
   - Both the Output and Output Enable registers must not have an E pin (clock enable).
4. For dual-tile devices (60 k, 125 k, and 250 k): Registers connected to an I/O on the Output and Output Enable pins must have the same clock and enable function:
   - Both the Output and Output Enable registers must have an E pin (clock enable), or none at all.
   - If the E pins are present, they must have the same polarity. The CLK pins must also have the same polarity.

   In some cases, the user may want registers to be combined with the input of a bibuf while maintaining the output as-is. This can be achieved by using PDC commands as follows:

```
set_io <signal name> -REGISTER yes ------register will combine
set_preserve <signal name> ----register will not combine
```

Weak Pull-Up and Weak Pull-Down Resistors

nano devices support optional weak pull-up and pull-down resistors on each I/O pin. When the I/O is pulled up, it is connected to the V_{CCI} of its corresponding I/O bank. When it is pulled down, it is connected to GND. Refer to the datasheet for more information.

For low power applications and when using IGLOO nano devices, configuration of the pull-up or pull-down of the I/O can be used to set the I/O to a known state while the device is in Flash*Freeze mode. Refer to "Flash*Freeze Technology and Low Power Modes" in an applicable FPGA fabric user's guide for more information.

The Flash*Freeze (FF) pin cannot be configured with a weak pull-down or pull-up I/O attribute, as the signal needs to be driven at all times.

Output Slew Rate Control

The slew rate is the amount of time an input signal takes to get from logic LOW to logic HIGH or vice versa.

It is commonly defined as the propagation delay between 10% and 90% of the signal's voltage swing. Slew rate control is available for the output buffers of low power flash devices. The output buffer has a programmable slew rate for both HIGH-to-LOW and LOW-to-HIGH transitions.

The slew rate can be implemented by using a PDC command (Table 7-5 on page 163), setting it "High" or "Low" in the I/O Attribute Editor in Designer, or instantiating a special I/O macro. The default slew rate value is "High."

Microsemi recommends the high slew rate option to minimize the propagation delay. This high-speed option may introduce noise into the system if appropriate signal integrity measures are not adopted. Selecting a low slew rate reduces this kind of noise but adds some delays in the system. Low slew rate is recommended when bus transients are expected.

Output Drive

The output buffers of nano devices can provide multiple drive strengths to meet signal integrity requirements. The LVTTL and LVCMOS (except 1.2 V LVCMOS) standards have selectable drive strengths.

Drive strength should also be selected according to the design requirements and noise immunity of the system.

User I/O Naming Convention

Due to the comprehensive and flexible nature of nano Standard I/Os, a naming scheme is used to show the details of each I/O (Figure 7-8). The name identifies to which I/O bank it belongs.

I/O Nomenclature = FF/Gmn/IOuxwBy

Gmn is only used for I/Os that also have CCC access—i.e., global pins.

FF	=	Indicates the I/O dedicated for the Flash*Freeze mode activation pin
G	=	Global
m	=	Global pin location associated with each CCC on the device: A (northwest corner), B (northeast corner), C (east middle), D (southeast corner), E (southwest corner), and F (west middle)
n	=	Global input MUX and pin number of the associated Global location m—either A0, A1, A2, B0, B1, B2, C0, C1, or C2. Refer to the "Global Resources in Low Power Flash Devices" section on page 31 for information about the three input pins per clock source MUX at CCC location m.
u	=	I/O pair number in the bank, starting at 00 from the northwest I/O bank and proceeding in a clockwise direction
x	=	R (Regular—single-ended) for the I/Os that support single-ended standards.
w	=	S (Single-Ended)
B	=	Bank
y	=	Bank number (0–3). The Bank number starts at 0 from the northwest I/O bank and proceeds in a clockwise direction.

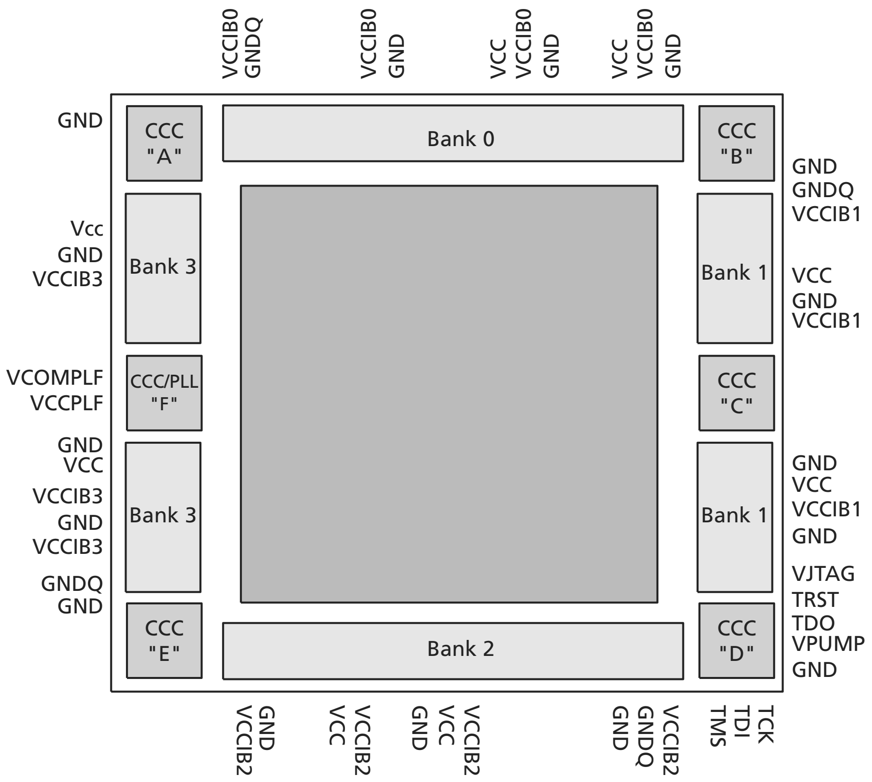


Figure 7-8 • **I/O Naming Conventions for nano Devices – Top View**

8 – I/O Software Control in Low Power Flash Devices

Fusion, IGLOO, and ProASIC3 I/Os provide more design flexibility, allowing the user to control specific features by enabling certain I/O standards. Some features are selectable only for certain I/O standards, whereas others are available for all I/O standards. For example, slew control is not supported by differential I/O standards. Conversely, I/O register combining is supported by all I/O standards. For detailed information about which I/O standards and features are available on each device and each I/O type, refer to the I/O Structures section of the handbook for the device you are using.

Figure 8-1 shows the various points in the software design flow where a user can provide input or control of the I/O selection and parameters. A detailed description is provided throughout this document.

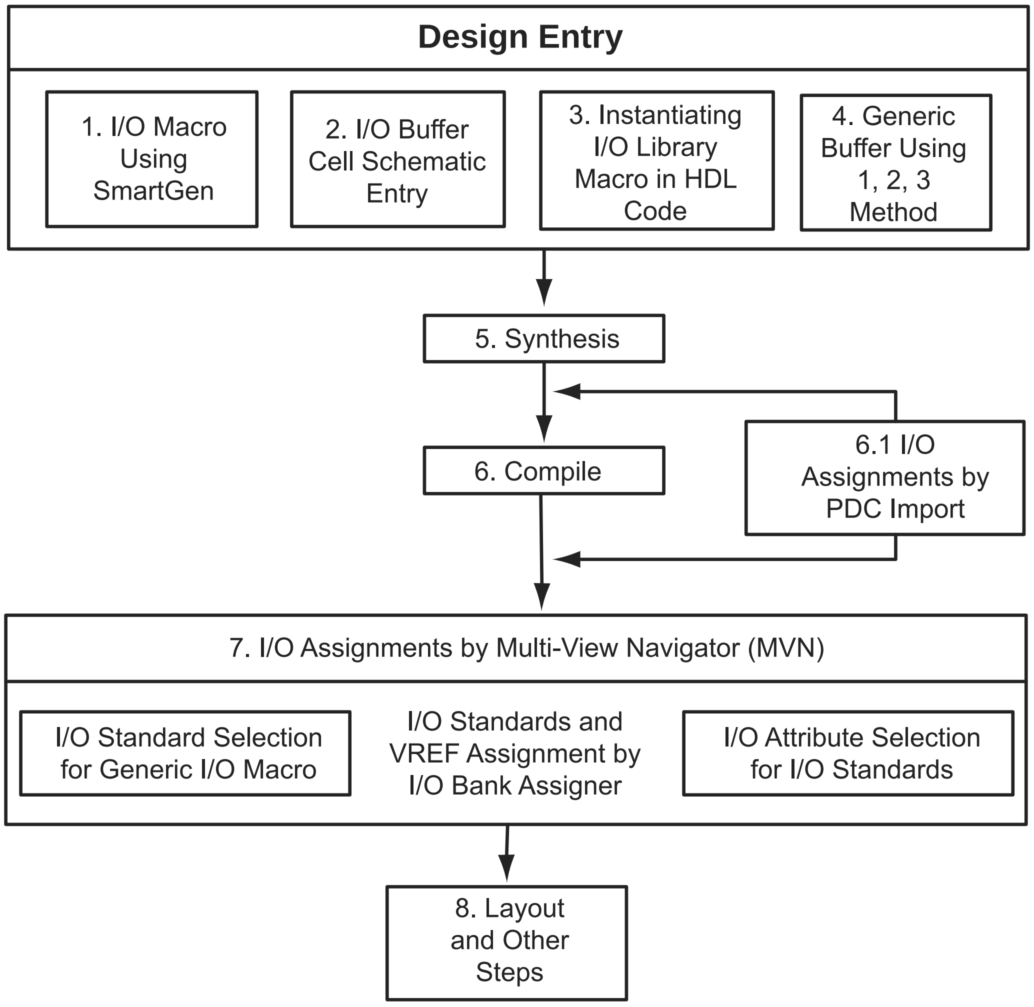

Design Entry
1. I/O Macro Using SmartGen
2. I/O Buffer Cell Schematic Entry
3. Instantiating I/O Library Macro in HDL Code
4. Generic Buffer Using 1, 2, 3 Method
5. Synthesis
6. Compile
6.1 I/O Assignments by PDC Import
7. I/O Assignments by Multi-View Navigator (MVN)
I/O Standard Selection for Generic I/O Macro
I/O Standards and VREF Assignment by I/O Bank Assigner
I/O Attribute Selection for I/O Standards
8. Layout and Other Steps

Figure 8-1 • **User I/O Assignment Flow Chart**

- 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
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.*

I/O Bank Resource Usage

This is an important portion of the report. The user must meet the requirements stated in this table. Figure 8-10 shows the I/O Bank Resource Usage table included in the I/O bank report:

Figure 8-10 **• I/O Bank Resource Usage Table**

The example in Figure 8-10 shows that none of the I/O macros is assigned to the bank because more than one VCCI is detected.

I/O Voltage Usage

The I/O Voltage Usage table provides the number of VREF (E devices only) and V_{CCI} assignments required in the design. If the user decides to make I/O assignments manually (PDC or MVN), the issues listed in this table must be resolved before proceeding to Layout. As stated earlier, VREF assignments must be made if there are any voltage-referenced I/Os.

Figure 8-11 shows the I/O Voltage Usage table included in the I/O bank report.

Figure 8-11 **• I/O Voltage Usage Table**

The table in Figure 8-11 indicates that there are two voltage-referenced I/Os used in the design. Even though both of the voltage-referenced I/O technologies have the same VCCI voltage, their VREF voltages are different. As a result, two I/O banks are needed to assign the VCCI and VREF voltages.

In addition, there are six single-ended I/Os used that have the **same VCCI voltage. Since two banks are already assigned with the same VCCI voltage and there are enough u**nused bonded I/Os in

Automatically Assigning Technologies to I/O Banks

The I/O Bank Assigner (IOBA) tool runs automatically when you run Layout. You can also use this tool from within the MultiView Navigator (Figure 8-17). The IOBA tool automatically assigns technologies and VREF pins (if required) to every I/O bank that does not currently have any technologies assigned to it. This tool is available when at least one I/O bank is unassigned.

To automatically assign technologies to I/O banks, choose I/O Bank Assigner from the **Tools** menu (or click the I/O Bank Assigner's toolbar button, shown in Figure 8-16).

Figure 8-16 **• I/O Bank Assigner's Toolbar Button**

Messages will appear in the Output window informing you when the automatic I/O bank assignment begins and ends. If the assignment is successful, the message "I/O Bank Assigner completed successfully" appears in the Output window, as shown in Figure 8-17.

Figure 8-17 **• I/O Bank Assigner Displays Messages in Output Window**

```
  DDR_OUT_0_inst : DDR_OUT
  port map(DR => DataR, DF => DataF, CLK => CLK, CLR => CLR, Q => Q);
  TRIBUFF_F_8U_0_inst : TRIBUFF_F_8U
  port map(D => Q, E => TrienAux, PAD => PAD);

end DEF_ARCH;
```

DDR Bidirectional Buffer

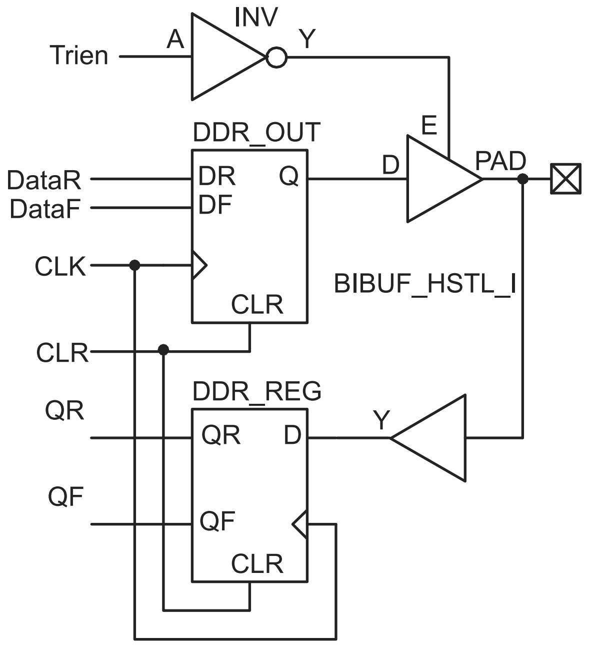


Figure 9-8 **• DDR Bidirectional Buffer, LOW Output Enable (HSTL Class II)**

Verilog

```
module DDR_BiDir_HSTL_I_LowEnb(DataR,DataF,CLR,CLK,Trien,QR,QF,PAD);

input   DataR, DataF, CLR, CLK, Trien;
output  QR, QF;
inout   PAD;

wire TrienAux, D, Q;

  INV Inv_Tri(.A(Trien), .Y(TrienAux));
  DDR_OUT DDR_OUT_0_inst(.DR(DataR),.DF(DataF),.CLK(CLK),.CLR(CLR),.Q(Q));
  DDR_REG DDR_REG_0_inst(.D(D),.CLK(CLK),.CLR(CLR),.QR(QR),.QF(QF));
  BIBUF_HSTL_I BIBUF_HSTL_I_0_inst(.PAD(PAD),.D(Q),.E(TrienAux),.Y(D));

endmodule
```

STAPL File with AES Encryption

- Does not contain AES key / FlashLock Key information
- Intended for transmission through web or service to unsecured locations for programming

```
=============================================
NOTE "CREATOR" "Designer Version: 6.1.1.108";
NOTE "DEVICE" "A3PE600";
NOTE "PACKAGE" "208 PQFP";
NOTE "DATE" "2005/04/08";
NOTE "STAPL_VERSION" "JESD71";
NOTE "IDCODE" "$123261CF";
NOTE "DESIGN" "counter32";
NOTE "CHECKSUM" "$EF57";
NOTE "SAVE_DATA" "FRomStream";
NOTE "SECURITY" "ENCRYPT FROM CORE ";
NOTE "ALG_VERSION" "1";
NOTE "MAX_FREQ" "20000000";
NOTE "SILSIG" "$00000000";
```

Conclusion

The new and enhanced security features offered in Fusion, IGLOO, and ProASIC3 devices provide state-of-the-art security to designs programmed into these flash-based devices. Microsemi low power flash devices employ the encryption standard used by NIST and the U.S. government—AES using the 128-bit Rijndael algorithm.

The combination of an on-chip AES decryption engine and FlashLock technology provides the highest level of security against invasive attacks and design theft, implementing the most robust and secure ISP solution. These security features protect IP within the FPGA and protect the system from cloning, wholesale “black box” copying of a design, invasive attacks, and explicit IP or data theft.

Glossary

Term	Explanation
Security Header programming file	Programming file used to program the FlashLock Pass Key and/or AES key into the device to secure the FPGA, FlashROM, and/or FBs.
AES (encryption) key	128-bit key defined by the user when the AES encryption option is set in the Microsemi Designer software when generating the programming file.
FlashLock Pass Key	128-bit key defined by the user when the FlashLock option is set in the Microsemi Designer software when generating the programming file. The FlashLock Key protects the security settings programmed to the device. Once a device is programmed with FlashLock, whatever settings were chosen at that time are secure.
FlashLock	The combined security features that protect the device content from attacks. These features are the following: • Flash technology that does not require an external bitstream to program the device • FlashLock Pass Key that secures device content by locking the security settings and preventing access to the device as defined by the user • AES key that allows secure, encrypted device reprogrammability

References

National Institute of Standards and Technology. “ADVANCED ENCRYPTION STANDARD (AES) Questions and Answers.” 28 January 2002 (10 January 2005).
See http://csrc.nist.gov/archive/aes/index1.html for more information.

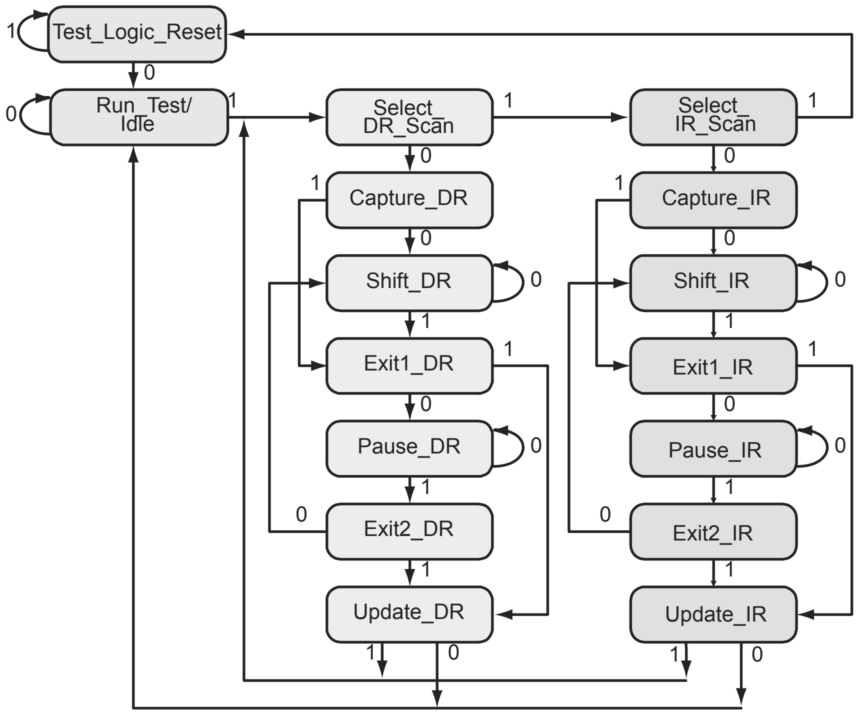


Figure 16-4 **• TAP Controller State Diagram**

UJTAG Port Usage

UIREG[7:0] hold the contents of the JTAG instruction register. The UIREG vector value is updated when the TAP Controller state machine enters the Update_IR state. Instructions 16 to 127 are user-defined and can be employed to encode multiple applications and commands within an application. Loading new instructions into the UIREG vector requires users to send appropriate logic to TMS to put the TAP Controller in a full IR cycle starting from the Select IR_Scan state and ending with the Update_IR state.

UTDI, UTDO, and UDRCK are directly connected to the JTAG TDI, TDO, and TCK ports, respectively. The TDI input can be used to provide either data (TAP Controller in the Shift_DR state) or the new contents of the instruction register (TAP Controller in the Shift_IR state).

UDRSH, UDRUPD, and UDRCAP are HIGH when the TAP Controller state machine is in the Shift_DR, Update_DR, and Capture_DR states, respectively. Therefore, they act as flags to indicate the stages of the data shift process. These flags are useful for applications in which blocks of data are shifted into the design from JTAG pins. For example, an active UDRSH can indicate that UTDI contains the data bitstream, and UDRUPD is a candidate for the end-of-data-stream flag.

As mentioned earlier, users should not connect the TDI, TDO, TCK, TMS, and TRST ports of the UJTAG macro to any port or net of the design netlist. The Designer software will automatically handle the port connection.

Index