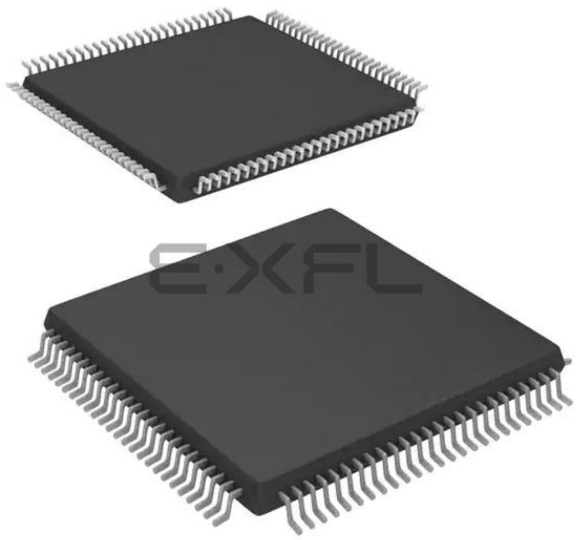


Understanding Embedded - FPGAs (Field Programmable Gate Array)

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	36864
Number of I/O	68
Number of Gates	250000
Voltage - Supply	1.14V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	0°C ~ 85°C (TJ)
Package / Case	100-TQFP
Supplier Device Package	100-VQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microsemi/a3p250l-vqg100

- The device is reset upon exiting Flash*Freeze mode or internal state saving is not required.
- State saving is required, but data and clock management is performed external to the FPGA. In other words, incoming data is externally guaranteed and held valid prior to entering Flash*Freeze mode.

Type 2 Flash*Freeze mode is ideally suited for applications with the following design criteria:

- Entering Flash*Freeze mode is dependent on an internal or external signal in addition to the external FF pin.
- State saving is required and incoming data is not externally guaranteed valid.
- The designer wants to use his/her own Flash*Freeze management IP for clock and data management.
- The designer wants to use his/her own Flash*Freeze management logic for clock and data management.
- Internal housekeeping is required prior to entering Flash*Freeze mode. Housekeeping activities may include loading data to SRAM, system shutdown, completion of current task, or ensuring valid Flash*Freeze pin assertion.

There is no downside to type 2 mode, and Microsemi's Flash*Freeze management IP offers a very low tile count clock and data management solution. Microsemi's recommendation for most designs is to use type 2 Flash*Freeze mode with Flash*Freeze management IP.

Design Solutions

Clocks

- Microsemi recommends using a completely synchronous design in Type 2 mode with Flash*Freeze management IP cleanly gating all internal and external clocks. This will prevent narrow pulses upon entrance and exit from Flash*Freeze mode (Figure 2-5 on page 30).
- Upon entering Flash*Freeze mode, external clocks become tied off High, internal to the clock pin (unless hold state is used on IGLOO nano or IGLOO PLUS), and PLLs are turned off. Any clock that is externally Low will realize a Low to High transition internal to the device while entering Flash*Freeze. If clocks will float during Flash*Freeze mode, Microsemi recommends using the weak pull-up feature. If clocks will continue to drive the device during Flash*Freeze mode, the clock gating (filter) available in Flash*Freeze management IP can help to filter unwanted narrow clock pulses upon Flash*Freeze mode entry and exit.
- Clocks may continue to drive FPGA pins while the device is in Flash*Freeze mode, with virtually no power consumption. The weak pull-up/-down configuration will result in unnecessary power consumption if used in this scenario.
- Floating clocks can cause totem pole currents on the input I/O circuitry when the device is in active mode. If clocks are externally gated prior to entering Flash*Freeze mode, Microsemi recommends gating them to a known value (preferably '1', to avoid a possible narrow pulse upon Flash*Freeze mode exit), and not leaving them floating. However, during Flash*Freeze mode, all inputs and clocks are internally tied off to prevent totem pole currents, so they can be left floating.
- Upon exiting Flash*Freeze mode, the design must allow maximum acquisition time for the PLL to acquire the lock signal, and for a PLL clock to become active. If a PLL output clock is used as the primary clock for Flash*Freeze management IP, it is important to note that the clock gating circuit will only release other clocks after the primary PLL output clock becomes available.

Flash*Freeze management IP. Additional information on this IP core can be found in the Libero online help.

The Flash*Freeze management IP is comprised of three blocks: the Flash*Freeze finite state machine (FSM), the clock gating (filter) block, and the ULSICC macro, as shown in Figure 2-10.

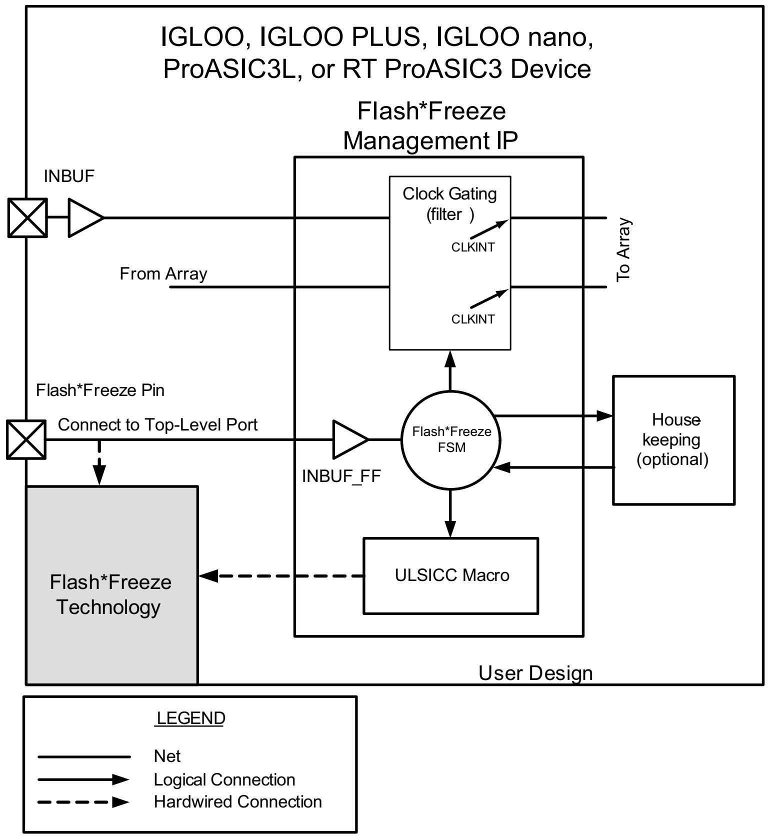


Figure 2-10 **• Flash*Freeze Management IP Block Diagram**

*Flash*Freeze Management FSM*

The Flash*Freeze FSM block is a simple, robust, fully encoded 3-bit state machine that ensures clean entrance to and exit from Flash*Freeze mode by controlling activities of the clock gating, ULSICC, and optional housekeeping blocks. The state diagram for the FSM is shown in Figure 2-11 on page 38. In normal operation, the state machine waits for Flash*Freeze pin assertion, and upon detection of a request, it waits for a short period of time to ensure the assertion persists; then it asserts WAIT_HOUSEKEEPING (active High) synchronous to the user's designated system clock. This flag can be used by user logic to perform any needed shutdown processes prior to entering Flash*Freeze mode, such as storing data into SRAM, notifying other system components of the request, or timing/validating the Flash*Freeze request. The FSM also asserts Flash_Freeze_Enabled whenever the device enters Flash*Freeze mode. This occurs after all housekeeping and clock gating functions have completed. The Flash_Freeze_Enabled signal remains asserted, even during Flash*Freeze mode, until the Flash*Freeze pin is deasserted. Use the Flash_Freeze_Enabled signal to drive any logic in the design that needs to be in a particular state during Flash*Freeze mode. The DONE_HOUSEKEEPING (active High) signal should be asserted to notify the FSM when all the housekeeping tasks are completed. If the user chooses not to use housekeeping, the Flash*Freeze management IP core generator in Libero SoC will connect WAIT_HOUSEKEEPING to DONE_HOUSEKEEPING.

3 – Global Resources in Low Power Flash Devices

Introduction

IGLOO, Fusion, and ProASIC3 FPGA devices offer a powerful, low-delay VersaNet global network scheme and have extensive support for multiple clock domains. In addition to the Clock Conditioning Circuits (CCCs) and phase-locked loops (PLLs), there is a comprehensive global clock distribution network called a VersaNet global network. Each logical element (VersaTile) input and output port has access to these global networks. The VersaNet global networks can be used to distribute low-skew clock signals or high-fanout nets. In addition, these highly segmented VersaNet global networks contain spines (the vertical branches of the global network tree) and ribs that can reach all the VersaTiles inside their region. This allows users the flexibility to create low-skew local clock networks using spines. This document describes VersaNet global networks and discusses how to assign signals to these global networks and spines in a design flow. Details concerning low power flash device PLLs are described in the "Clock Conditioning Circuits in Low Power Flash Devices and Mixed Signal FPGAs" section on page 77. This chapter describes the low power flash devices' global architecture and uses of these global networks in designs.

Global Architecture

Low power flash devices offer powerful and flexible control of circuit timing through the use of global circuitry. Each chip has up to six CCCs, some with PLLs.

- In IGLOOe, ProASIC3EL, and ProASIC3E devices, all CCCs have PLLs—hence, 6 PLLs per device (except the PQ208 package, which has only 2 PLLs).
- In IGLOO, IGLOO nano, IGLOO PLUS, ProASIC3, and ProASIC3L devices, the west CCC contains a PLL core (except in 10 k through 30 k devices).
- 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.

Refer to Table 4-6 on page 100 for details. Each PLL includes delay lines, a phase shifter (0°, 90°, 180°, 270°), and clock multipliers/dividers. Each CCC has all the circuitry needed for the selection and interconnection of inputs to the VersaNet global network. The east and west CCCs each have access to three chip global lines on each side of the chip (six chip global lines total). The CCCs at the four corners each have access to three quadrant global lines in each quadrant of the chip (except in 10 k through 30 k gate devices).

The nano 10 k, 15 k, and 20 k devices support four VersaNet global resources, and 30 k devices support six global resources. The 10 k through 30 k devices have simplified CCCs called CCC-GLs.

The flexible use of the VersaNet global network allows the designer to address several design requirements. User applications that are clock-resource-intensive can easily route external or gated internal clocks using VersaNet global routing networks. Designers can also drastically reduce delay penalties and minimize resource usage by mapping critical, high-fanout nets to the VersaNet global network.

Note: Microsemi recommends that you choose the appropriate global pin and use the appropriate global resource so you can realize these benefits.

The following sections give an overview of the VersaNet global network, the structure of the global network, access point for the global networks, and the clock aggregation feature that enables a design to have very low clock skew using spines.

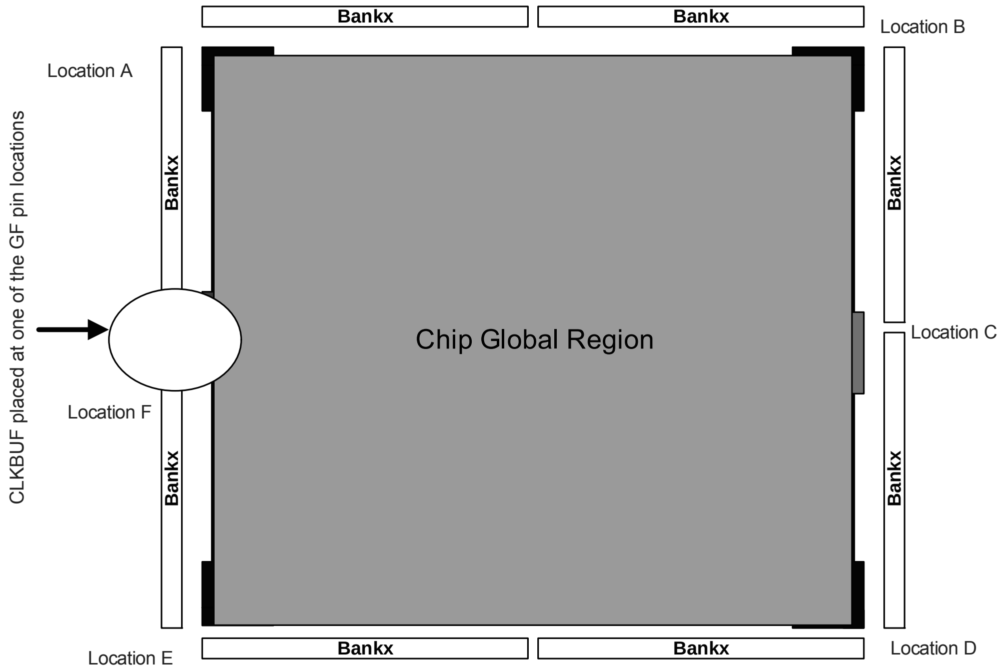


Figure 3-12 **• Chip Global Region**

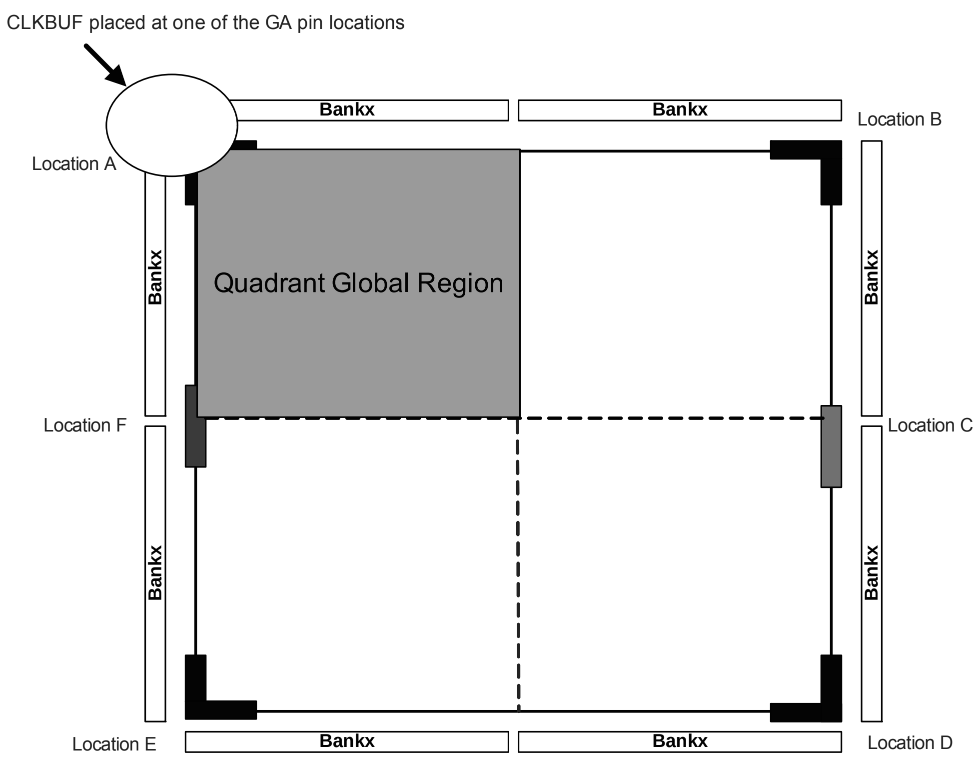


Figure 3-13 **• Quadrant Global Region**

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 77). 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.

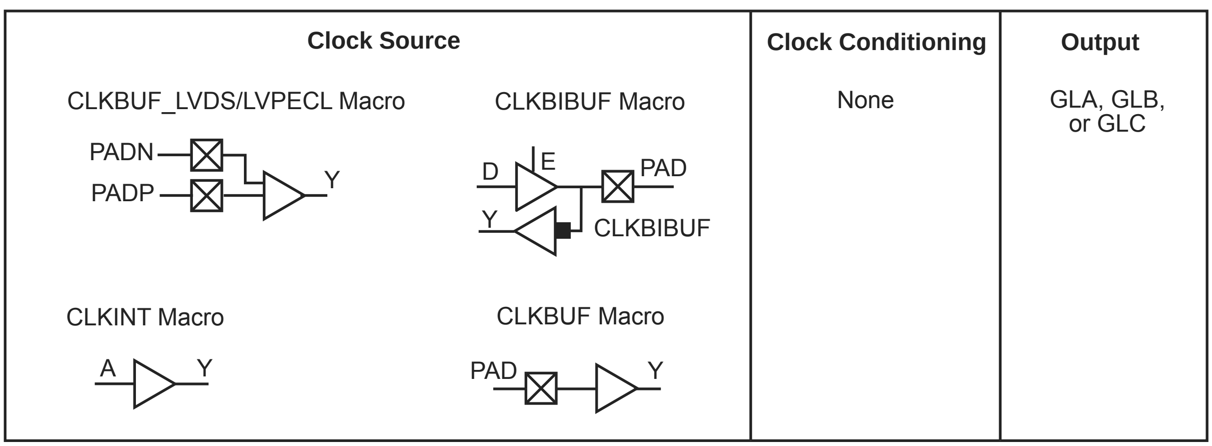
Note: IGLOO nano and ProASIC nano devices do not support differential inputs.

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[1] macros are composite macros that include an I/O macro driving a global buffer, which uses a hardwired connection.
- The CLKBUF, CLKBUF_LVPECL/LVDS[1] 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.*



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 81).

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

Each global buffer, as well as the PLL reference clock, can be driven from one of the following:

- 3 dedicated single-ended I/Os using a hardwired connection
- 2 dedicated differential I/Os using a hardwired connection (not applicable for IGLOO nano and ProASIC3 nano devices)
- The FPGA core

Since the architecture of the devices varies as size increases, the following list details I/O types supported for globals:

IGLOO and ProASIC3

- LVDS-based clock sources are available only on 250 k gate devices and above (IGLOO nano and ProASIC3 nano devices do not support differential inputs).
- 60 k and 125 k gate devices support single-ended clock sources only.
- 15 k and 30 k gate devices support these inputs for CCC only and do not contain a PLL.
- nano devices:
  – 10 k, 15 k, and 20 k devices do not contain PLLs in the CCCs, and support only CLKBUF and CLKINT.
  – 60 k, 125 k, and 250 k devices support one PLL in the middle left CCC position. In the absence of the PLL, this CCC can be used by CLKBUF, CLKINT, and CLKDLY macros. The corner CCCs support CLKBUF, CLKINT, and CLKDLY.

Fusion

- AFS600 and AFS1500: All single-ended, differential, and voltage-referenced I/O standards (Pro I/O).
- AFS090 and AFS250: All single-ended and differential I/O standards.

Clock Sources for PLL and CLKDLY Macros

The input reference clock (CLKA for a PLL macro, CLK for a CLKDLY macro) can be accessed from different sources via the associated clock multiplexer tree. Each CCC has the option of choosing the source of the input clock from one of the following:

- Hardwired I/O
- External I/O
- Core Logic
- RC Oscillator (Fusion only)
- Crystal Oscillator (Fusion only)

The SmartGen macro builder tool allows users to easily create the PLL and CLKDLY macros with the desired settings. Microsemi strongly recommends using SmartGen to generate the CCC macros.

Hardwired I/O Clock Source

Hardwired I/O refers to global input pins that are hardwired to the multiplexer tree, which directly accesses the CCC global buffers. These global input pins have designated pin locations and are indicated with the I/O naming convention *Gmn* (*m* refers to any one of the positions where the PLL core is available, and *n* refers to any one of the three global input MUXes and the pin number of the associated global location, *m*). Choosing this option provides the benefit of directly connecting to the CCC reference clock input, which provides less delay. See Figure 4-9 on page 90 for an example illustration of the connections, shown in red. If a CLKDLY macro is initiated to utilize the programmable delay element of the CCC, the clock input can be placed at one of nine dedicated global input pin locations. In other words, if Hardwired I/O is chosen as the input source, the user can decide to place the input pin in one of the GmA0, GmA1, GmA2, GmB0, GmB1, GmB2, GmC0, GmC1, or GmC2 locations of the low power flash devices. When a PLL macro is used to utilize the PLL core in a CCC location, the clock input of the PLL can only be connected to one of three GmA* global pin locations: GmA0, GmA1, or GmA2.

Table 4-9 to Table 4-15 on page 110 provide descriptions of the configuration data for the configuration bits.

Table 4-9 **• Input Clock Divider, FINDIV[6:0] (/n)**

FINDIV<6:0> State	Divisor	New Frequency Factor
0	1	1.00000
1	2	0.50000
⋮	⋮	⋮
127	128	0.0078125

Table 4-10 **• Feedback Clock Divider, FBDIV[6:0] (/m)**

FBDIV<6:0> State	Divisor	New Frequency Factor
0	1	1
1	2	2
⋮	⋮	⋮
127	128	128

Table 4-11 **• Output Frequency Dividers**
A **Output Divider, OADIV <4:0> (/u);**
B **Output Divider, OBDIV <4:0> (/v);**
C **Output Divider, OCDIV <4:0> (/w)**

OADIV<4:0>; OBDIV<4:0>; CDIV<4:0> State	Divisor	New Frequency Factor
0	1	1.00000
1	2	0.50000
⋮	⋮	⋮
31	32	0.03125

Table 4-12 **• MUXA, MUXB, MUXC**

OAMUX<2:0>; OBMUX<2:0>; OCMUX<2:0> State	MUX Input Selected
0	None. Six-input MUX and PLL are bypassed. Clock passes only through global MUX and goes directly into HC ribs.
1	Not available
2	PLL feedback delay line output
3	Not used
4	PLL VCO 0° phase shift
5	PLL VCO 270° phase shift
6	PLL VCO 180° phase shift
7	PLL VCO 90° phase shift

```
Primary Clock Output Delay from CLKA -3.020

Secondary1 Clock frequency 40.000
Secondary1 Clock Phase Shift 0.000
Secondary1 Clock Global Output Delay from CLKA 2.515
```

Next, perform simulation in Model*Sim* to verify the correct delays. Figure 4-30 shows the simulation results. The delay values match those reported in the SmartGen PLL Wizard.

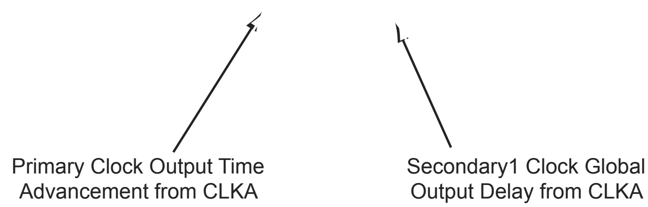


***Figure 4-30* • Model*Sim* Simulation Results**

The timing can also be analyzed using SmartTime in Designer. The user should import the synthesized netlist to Designer, perform Compile and Layout, and then invoke SmartTime. Go to **Tools** > **Options** and change the maximum delay operating conditions to **Typical Case**. Then expand the Clock-to-Out paths of GLA and GLB and the individual components of the path delays are shown. The path of GLA is shown in Figure 4-31 on page 123 displaying the same delay value.

***Figure 4-31* • Static Timing Analysis Using SmartTime**

Place-and-Route Stage Considerations

Several considerations must be noted to properly place the CCC macros for layout.

For CCCs with clock inputs configured with the Hardwired I/O–Driven option:

- PLL macros must have the clock input pad coming from one of the GmA* locations.
- CLKDLY macros must have the clock input pad coming from one of the Global I/Os.

If a PLL with a Hardwired I/O input is used at a CCC location and a Hardwired I/O–Driven CLKDLY macro is used at the same CCC location, the clock input of the CLKDLY macro must be chosen from one of the GmB* or GmC* pin locations. If the PLL is not used or is an External I/O–Driven or Core Logic–Driven PLL, the clock input of the CLKDLY macro can be sourced from the GmA*, GmB*, or GmC* pin locations.

For CCCs with clock inputs configured with the External I/O–Driven option, the clock input pad can be assigned to any regular I/O location (IO******** pins). Note that since global I/O pins can also be used as regular I/Os, regardless of CCC function (CLKDLY or PLL), clock inputs can also be placed in any of these I/O locations.

By default, the Designer layout engine will place global nets in the design at one of the six chip globals. When the number of globals in the design is greater than six, the Designer layout engine will automatically assign additional globals to the quadrant global networks of the low power flash devices. If the user wishes to decide which global signals should be assigned to chip globals (six available) and which to the quadrant globals (three per quadrant for a total of 12 available), the assignment can be achieved with PinEditor, ChipPlanner, or by importing a placement constraint file. Layout will fail if the

SRAM/FIFO Support in Flash-Based Devices

The flash FPGAs listed in Table 6-1 support SRAM and FIFO blocks and the functions described in this document.

Table 6-1 **• 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 6-1. 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 6-1. 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*.

The ROM emulation application is based on RAM block initialization. If the user's main design has access only to the read ports of the RAM block (RADDR, RD, RCLK, and REN), and the contents of the RAM are already initialized through the TAP, then the memory blocks will emulate ROM functionality for the core design. In this case, the write ports of the RAM blocks are accessed only by the user interface block, and the interface is activated only by the TAP Instruction Register contents.

Users should note that the contents of the RAM blocks are lost in the absence of applied power. However, the 1 kbit of flash memory, FlashROM, in low power flash devices can be used to retain data after power is removed from the device. Refer to the "SRAM and FIFO Memories in Microsemi's Low Power Flash Devices" section on page 147 for more information.

Sample Verilog Code

Interface Block

```
`define Initialize_start 8'h22 //INITIALIZATION START COMMAND VALUE
`define Initialize_stop 8'h23 //INITIALIZATION START COMMAND VALUE

module interface(IR, rst_n, data_shift, clk_in, data_update, din_ser, dout_ser, test,
  test_out,test_clk,clk_out,wr_en,rd_en,write_word,read_word,rd_addr, wr_addr);

input [7:0] IR;
input [3:0] read_word; //RAM DATA READ BACK
input rst_n, data_shift, clk_in, data_update, din_ser; //INITIALIZATION SIGNALS
input test, test_clk; //TEST PROCEDURE CLOCK AND COMMAND INPUT
output [3:0] test_out; //READ DATA
output [3:0] write_word; //WRITE DATA
output [1:0] rd_addr; //READ ADDRESS
output [1:0] wr_addr; //WRITE ADDRESS
output dout_ser; //TDO DRIVER
output clk_out, wr_en, rd_en;

wire [3:0] write_word;
wire [1:0] rd_addr;
wire [1:0] wr_addr;
wire [3:0] Q_out;
wire enable, test_active;

reg clk_out;

//SELECT CLOCK FOR INITIALIZATION OR READBACK TEST
always @(enable or test_clk or data_update)
begin
  case ({test_active})
    1 : clk_out = test_clk ;
    0 : clk_out = !data_update;
    default : clk_out = 1'b1;
  endcase
end

assign test_active = test && (IR == 8'h23);
assign enable = (IR == 8'h22);
assign wr_en = !enable;
assign rd_en = !test_active;
assign test_out = read_word;
assign dout_ser = Q_out[3];

//4-bit SIN/POUT SHIFT REGISTER
shift_reg data_shift_reg (.Shiften(data_shift), .Shiftin(din_ser), .Clock(clk_in),
  .Q(Q_out));

//4-bit PIPELINE REGISTER
D_pipeline pipeline_reg (.Data(Q_out), .Clock(data_update), .Q(write_word));
```

I/O Architecture

I/O Tile

The I/O tile provides a flexible, programmable structure for implementing a large number of I/O standards. In addition, the registers available in the I/O tile can be used to support high-performance register inputs and outputs, with register enable if desired (Figure 7-2). The registers can also be used to support the JESD-79C Double Data Rate (DDR) standard within the I/O structure (see the "DDR for Microsemi's Low Power Flash Devices" section on page 271 for more information). In addition, the registers available in the I/O tile can be used to support high-performance register inputs and outputs, with register enable if desired (Figure 7-2).

As depicted in Figure 7-2, all I/O registers share one CLR port. The output register and output enable register share one CLK port.

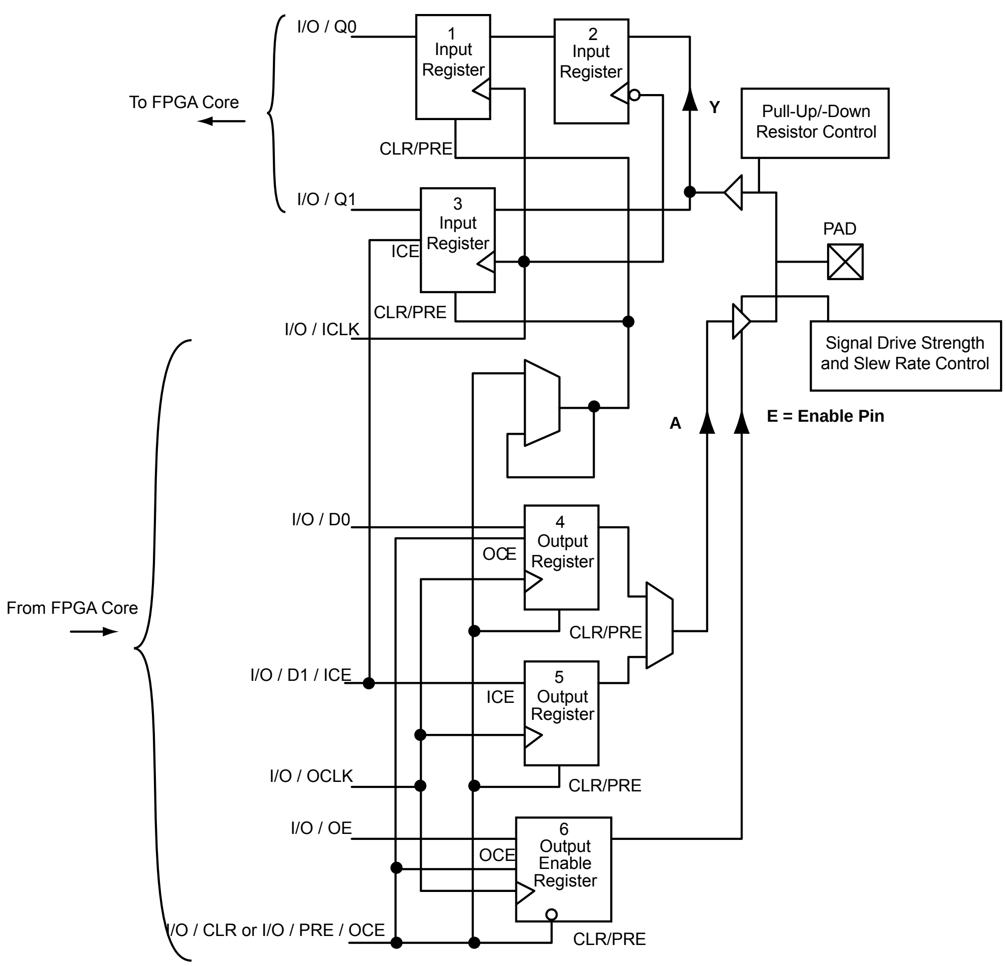


Figure 7-2 • **DDR Configured I/O Block Logical Representation**

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

Electrostatic Discharge Protection

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

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

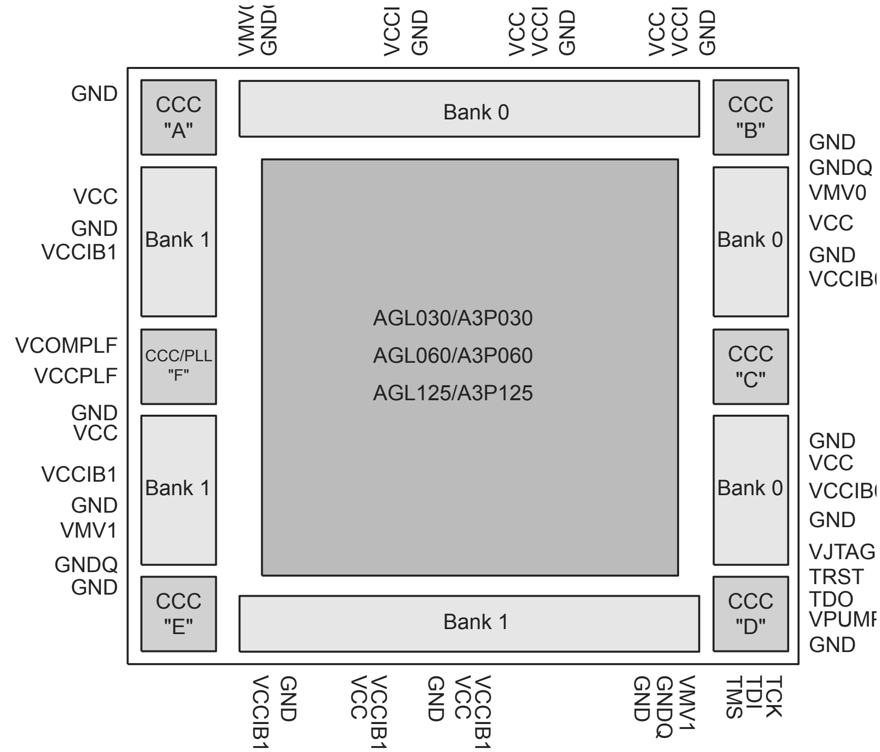
All IGLOO and ProASIC3 devices are tested to the Human Body Model (HBM) and the Charged Device Model (CDM).

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

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

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

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



Note: *The 30 k gate devices do not support a PLL (V_{COMPLF} and V_{CCPLF} pins).*

Figure 7-19 **• Naming Conventions of IGLOO and ProASIC3 Devices with Two I/O Banks – Top View**

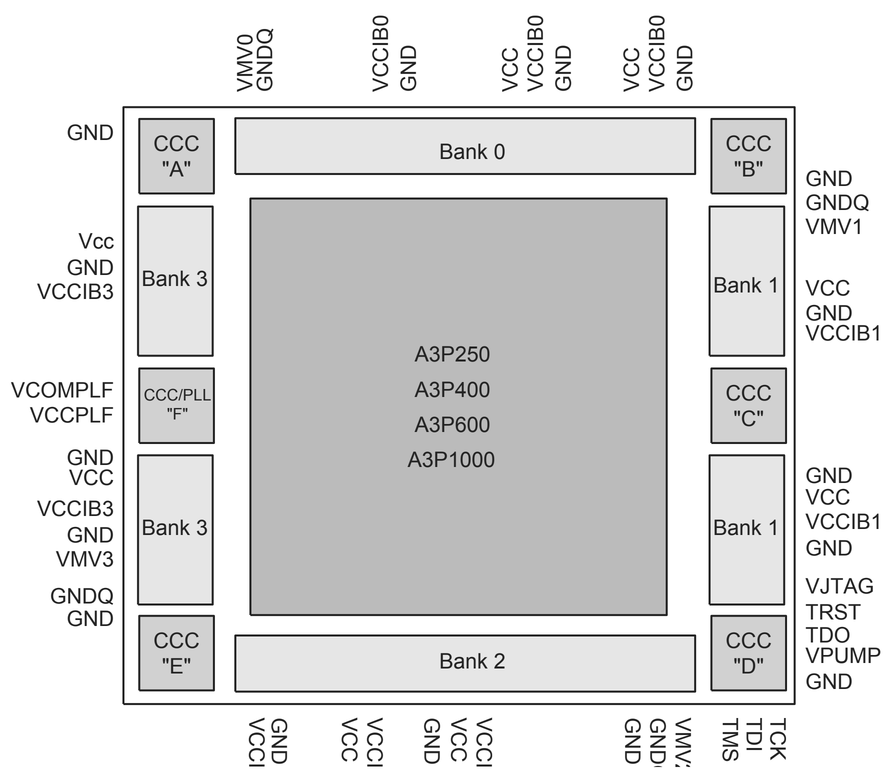


Figure 7-20 **• Naming Conventions of IGLOO and ProASIC3 Devices with Four I/O Banks – Top View**

B-LVDS/M-LVDS

Bus LVDS (B-LVDS) refers to bus interface circuits based on LVDS technology. Multipoint LVDS (M-LVDS) specifications extend the LVDS standard to high-performance multipoint bus applications. Multidrop and multipoint bus configurations may contain any combination of drivers, receivers, and transceivers. Microsemi LVDS drivers provide the higher drive current required by B-LVDS and M-LVDS to accommodate the loading. The driver requires series terminations for better signal quality and to control voltage swing. Termination is also required at both ends of the bus, since the driver can be located anywhere on the bus. These configurations can be implemented using TRIBUF_LVDS and BIBUF_LVDS macros along with appropriate terminations. Multipoint designs using Microsemi LVDS macros can achieve up to 200 MHz with a maximum of 20 loads. A sample application is given in Figure 8-9. The input and output buffer delays are available in the LVDS sections in the datasheet.

Example: For a bus consisting of 20 equidistant loads, the terminations given in EQ 8-1 provide the required differential voltage, in worst case industrial operating conditions, at the farthest receiver:

$R_S = 60\ \Omega$, $R_T = 70\ \Omega$, given $Z_O = 50\ \Omega$ (2") and $Z_{stub} = 50\ \Omega$ (~1.5").

EQ 8-1

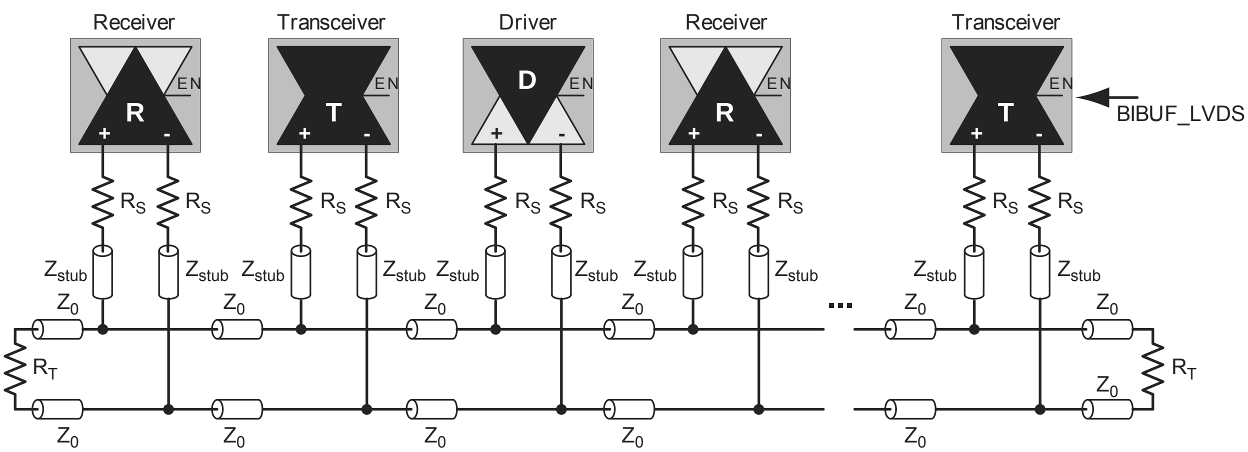


Figure 8-9 **• A B-LVDS/M-LVDS Multipoint Application Using LVDS I/O Buffers**

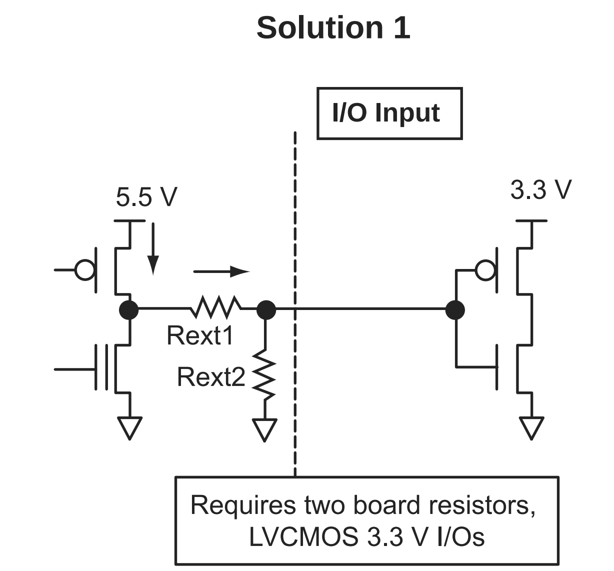


Figure 8-10 **• Solution 1**

Solution 2

The board-level design must ensure that the reflected waveform at the pad does not exceed the voltage overshoot/undershoot limits provided in the datasheet. This is a requirement to ensure long-term reliability.

This scheme will also work for a 3.3 V PCI/PCI-X configuration, but the internal diode should not be used for clamping, and the voltage must be limited by the external resistors and Zener, as shown in Figure 8-11. Relying on the diode clamping would create an excessive pad DC voltage of 3.3 V + 0.7 V = 4 V.

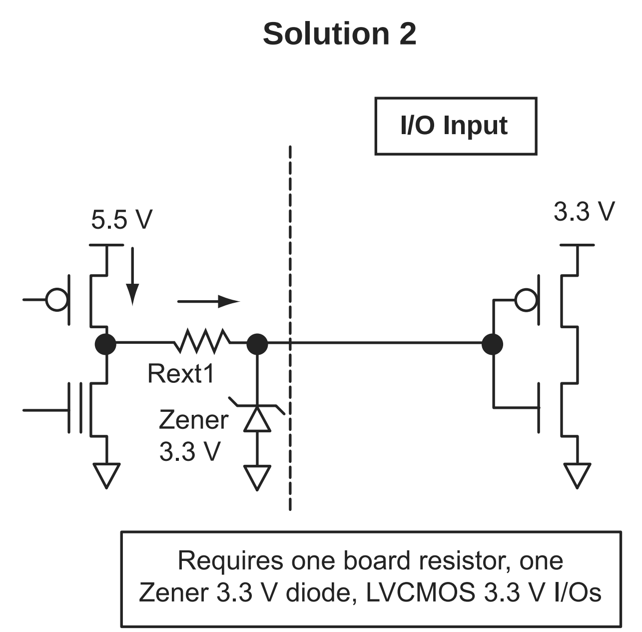


Figure 8-11 **• Solution 2**

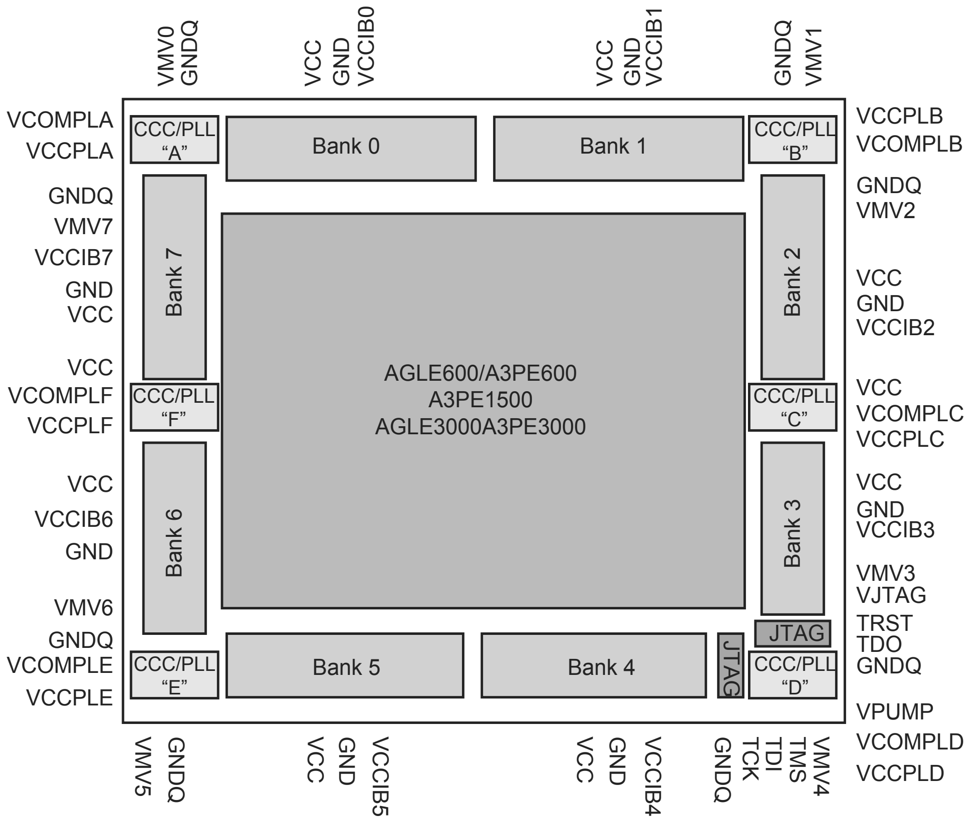


Figure 8-20 **• User I/O Naming Conventions of IGLOOe and ProASIC3E Devices – Top View**

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" chapter of the appropriate 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, Fusion, and ProASIC3 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.

List of Changes

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

Date	Changes	Page
August 2012	Figure 8-1 • DDR Configured I/O Block Logical Representation and Figure 8-3 • DDR Configured I/O Block Logical Representation were revised to indicate that resets on registers 1, 3, 4, and 5 are active high rather than active low. The title of the figures was revised from "I/O Block Logical Representation" (SAR 40685).	213, 220
	AGLE1500 was removed from Table 8-2 • Supported I/O Standards because it is not a valid offering. LVCMOS 1.2 was added to the single-ended standards. LVCMOS 1.2 was added to Table 8-3 • VCCI Voltages and Compatible IGLOOe and ProASIC3E Standards (SAR 33207).	215, 217
	Lack of a heading for the "User I/O Naming Convention" section made the information difficult to locate. A heading now introduces the user I/O naming conventions (SAR 38059).	245
	Figure 8-5 • Simplified I/O Buffer Circuitry and Table 8-8 • Programmable I/O Features (user control via I/O Attribute Editor) were modified to indicate that programmable input delay control is applicable only to ProASIC3E, IGLOOe, ProASIC3EL, and RT ProASIC3 devices (SAR 39666).	222, 227
	The hyperlink for the *Board-Level Considerations* application note was corrected (SAR 36663).	246, 248
June 2011	Figure 8-1 • DDR Configured I/O Block Logical Representation and Figure 8-3 • DDR Configured I/O Block Logical Representation were revised so that the I/O_CLR and I/O_OCLK nets are no longer joined in front of Input Register 3 but instead on the branch of the CLR/PRE signal (SAR 26052).	213, 220
	The "Pro I/Os—IGLOOe, ProASIC3EL, and ProASIC3E" section was revised. Formerly it stated, "3.3 V PCI and 3.3 V PCI-X are 5 V–tolerant." This sentence now reads, "3.3 V PCI and 3.3 V PCI-X can be configured to be 5 V–tolerant" (SAR 20983).	215
	Table 8-5 • Legal IGLOOe and ProASIC3E I/O Usage Matrix within the Same Bank was revised as follows (SAR 22467): The combination of 3.3 V I/O bank voltage with 1.50 V minibank voltage and LVDS, B-LVDS, M-LVDS, and DDR was made an illegal combination (now gray instead of white). The combination of 2.5 V I/O bank voltage with no minibank voltage and LVDS, B-LVDS, M-LVDS, and DDR was made a valid combination (now white instead of gray).	217
	The following sentence was removed from the "LVCMOS (Low-Voltage CMOS)" section (SAR 22634): "All these versions use a 3.3 V–tolerant CMOS input buffer and a push-pull output buffer."	223
	The "Electrostatic Discharge Protection" section was revised to remove references to tolerances (refer to the *Reliability Report* for tolerances). The Machine Model (MM) is not supported and was deleted from this section (SAR 24385).	231
	The "I/O Interfacing" section was revised to state that low power flash devices are 5 V–input– and 5 V–output–tolerant if certain I/O standards are selected, removing "without adding any extra circuitry," which was incorrect (SAR 21404).	247
July 2010	This chapter is no longer published separately with its own part number and version but is now part of several FPGA fabric user's guides.	N/A
v1.4 (December 2008)	The terminology in the "Low Power Flash Device I/O Support" section was revised.	214

List of Changes

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

Date	Changes	Page
July 2010	FlashPro4 is a replacement for FlashPro3 and has been added to this chapter. FlashPro is no longer available.	N/A
	The chapter was updated to include SmartFusion devices.	N/A
	The following were deleted: "Live at Power-Up (LAPU) or Boot PROM" section "Design Security" section Table 14-2 • Programming Features for Actel Devices and much of the text in the "Programming Features for Microsemi Devices" section "Programming Flash FPGAs" section "Return Material Authorization (RMA) Policies" section	N/A
	The "Device Programmers" section was revised.	291
	The Independent Programming Centers information was removed from the "Volume Programming Services" section.	292
	Table 11-3 • Programming Solutions was revised to add FlashPro4 and note that FlashPro is discontinued. A note was added for FlashPro Lite regarding power supply requirements.	293
	Most items were removed from Table 11-4 • Programming Ordering Codes, including FlashPro3 and FlashPro.	294
	The "Programmer Device Support" section was deleted and replaced with a reference to the Microsemi SoC Products Group website for the latest information.	294
	The "Certified Programming Solutions" section was revised to add FlashPro4 and remove Silicon Sculptor I and Silicon Sculptor 6X. Reference to *Programming and Functional Failure Guidelines* was added.	294
	The file type *.pdb was added to the "Use the Latest Version of the Designer Software to Generate Your Programming File (recommended)" section.	295
	Instructions on cleaning and careful insertion were added to the "Perform Routine Hardware Self-Diagnostic Test" section. Information was added regarding testing Silicon Sculptor programmers with an adapter module installed before every programming session verifying their calibration annually.	295
	The "Signal Integrity While Using ISP" section is new.	296
	The "Programming Failure Allowances" section was revised.	296

FlashROM and Programming Files

Each low power flash device has 1 kbit of on-chip, nonvolatile flash memory that can be accessed from the FPGA core. This nonvolatile FlashROM is arranged in eight pages of 128 bits (Figure 13-3). Each page can be programmed independently, with or without the 128-bit AES encryption. The FlashROM can only be programmed via the IEEE 1532 JTAG port and cannot be programmed from the FPGA core. In addition, during programming of the FlashROM, the FPGA core is powered down automatically by the on-chip programming control logic.

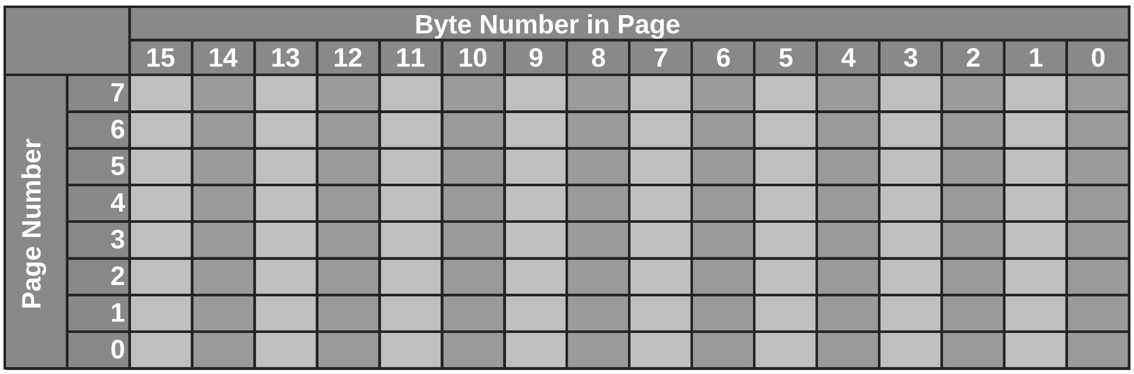


Figure 13-3 **• FlashROM Architecture**

When using FlashROM combined with AES, many subscription-based applications or device serialization applications are possible. The FROM configurator found in the Libero SoC Catalog supports easy management of the FlashROM contents, even over large numbers of devices. The FROM configurator can support FlashROM contents that contain the following:

- Static values
- Random numbers
- Values read from a file
- Independent updates of each page

In addition, auto-incrementing of fields is possible. In applications where the FlashROM content is different for each device, you have the option to generate a single STAPL file for all the devices or individual serialization files for each device. For more information on how to generate the FlashROM content for device serialization, refer to the "FlashROM in Microsemi’s Low Power Flash Devices" section on page 133.

Libero SoC includes a unique tool to support the generation and management of FlashROM and FPGA programming files. This tool is called FlashPoint.

Depending on the applications, designers can use the FlashPoint software to generate a STAPL file with different contents. In each case, optional AES encryption and/or different security settings can be set.

In Designer, when you click the Programming File icon, FlashPoint launches, and you can generate STAPL file(s) with four different cases (Figure 13-4 on page 334). When the serialization feature is used during the configuration of FlashROM, you can generate a single STAPL file that will program all the devices or an individual STAPL file for each device.

The following cases present the FPGA core and FlashROM programming file combinations that can be used for different applications. In each case, you can set the optional security settings (FlashLock Pass Key and/or AES Key) depending on the application.

1. A single STAPL file or multiple STAPL files with multiple FlashROM contents and the FPGA core content. A single STAPL file will be generated if the device serialization feature is not used. You can program the whole FlashROM or selectively program individual pages.
2. A single STAPL file for the FPGA core content