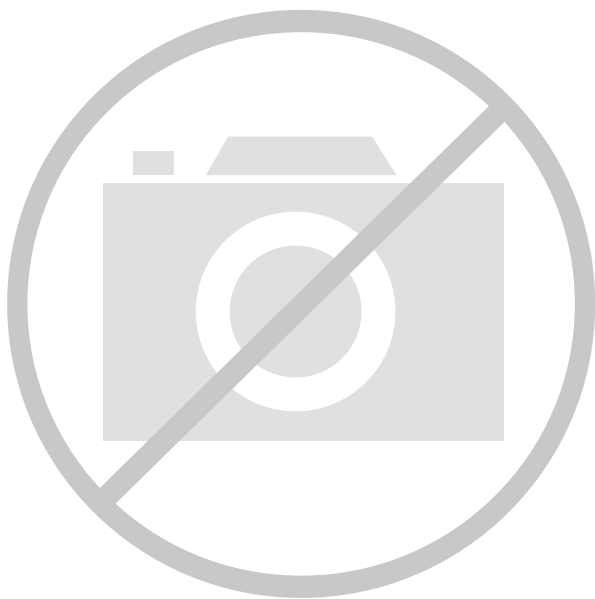




Welcome to [E-XFL.COM](https://www.e-xfl.com)

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

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

Applications of Embedded - FPGAs

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

Details

Product Status	Obsolete
Number of LABs/CLBs	-
Number of Logic Elements/Cells	-
Total RAM Bits	110592
Number of I/O	177
Number of Gates	600000
Voltage - Supply	1.14V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	0°C ~ 85°C (TJ)
Package / Case	256-LBGA
Supplier Device Package	256-FPBGA (17x17)
Purchase URL	https://www.e-xfl.com/product-detail/microsemi/m1a3p600l-fgg256

power supply and board-level configurations, the user can easily calculate how long it will take for the core to become inactive or active. For more information, refer to the "Power-Up/-Down Behavior of Low Power Flash Devices" section on page 373.

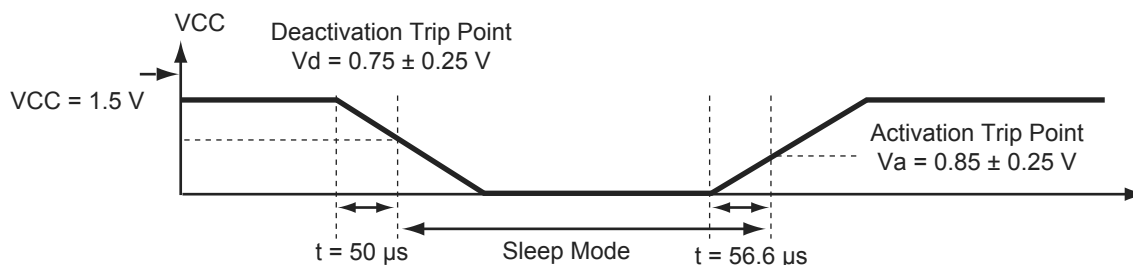


Figure 2-8 • Entering and Exiting Sleep Mode, Typical Timing Diagram

Context Save and Restore in Sleep or Shutdown Mode

In Sleep mode or Shutdown mode, the contents of the SRAM, state of the I/Os, and state of the registers are lost when the device is powered off, if no other measure is taken. A low-cost external serial EEPROM can be used to save and restore the contents of the device when entering and exiting Sleep mode or Shutdown mode. In the *Embedded SRAM Initialization Using External Serial EEPROM* application note, detailed information and a reference design are provided for initializing the embedded SRAM using an external serial EEPROM. The user can easily customize the reference design to save and restore the FPGA state when entering and exiting Sleep mode or Shutdown mode. The microcontroller will need to manage this activity; hence, before powering down V_{CC} , the data will be read from the FPGA and stored externally. In a similar way, after the FPGA is powered up, the microcontroller will allow the FPGA to load the data from external memory and restore its original state.

Flash*Freeze Design Guide

This section describes how designers can create reliable designs that use ultra-low power Flash*Freeze modes optimally. The section below provides guidance on how to select the best Flash*Freeze mode for any application. The "Design Solutions" section on page 35 gives specific recommendations on how to design and configure clocks, set/reset signals, and I/Os. This section also gives an overview of the design flow and provides details concerning Microsemi's Flash*Freeze Management IP, which enables clean clock gating and housekeeping. The "Additional Power Conservation Techniques" section on page 41 describes board-level considerations for entering and exiting Flash*Freeze mode.

Selecting the Right Flash*Freeze Mode

Both Flash*Freeze modes will bring an FPGA into an ultra-low power static mode that retains register and SRAM content and sets I/Os to a predetermined configuration. There are two primary differences that distinguish type 2 mode from type 1, and they must be considered when creating a design using Flash*Freeze technology.

First, with type 2 mode, the device has an opportunity to wait for a second signal to enable activation of Flash*Freeze mode. This allows processes to complete prior to deactivating the device, and can be useful to control task completion, data preservation, accidental Flash*Freeze activation, system shutdown, or any other housekeeping function. The second signal may be derived from an external or in-to-out internal source. The second difference between type 1 and type 2 modes is that a design for type 2 mode has an opportunity to cleanly manage clocks and data activity before entering and exiting Flash*Freeze mode. This is particularly important when data preservation is needed, as it ensures valid data is stored prior to entering, and upon exiting, Flash*Freeze mode.

Type 1 Flash*Freeze mode is ideally suited for applications with the following design criteria:

- Entering Flash*Freeze mode is not dependent on any signal other than the external FF pin.
- Internal housekeeping is not required prior to entering Flash*Freeze.

- Avoid using pull-ups and pull-downs on I/Os because these resistors draw some current. Avoid driving resistive loads or bipolar transistors, since these draw a continuous current, thereby adding to the static current.
- When partitioning the design across multiple devices, minimize I/O usage among the devices.

Conclusion

Microsemi IGLOO, IGLOO nano, IGLOO PLUS, ProASIC3L, and RT ProASIC3 family architectures are designed to achieve ultra-low power consumption based on enhanced nonvolatile and live-at-power-up flash-based technology. Power consumption can be reduced further by using Flash*Freeze, Static (Idle), Sleep, and Shutdown power modes. All these features result in a low power, cost-effective, single-chip solution designed specifically for power-sensitive and battery-operated electronics applications.

Related Documents

Application Notes

Embedded SRAM Initialization Using External Serial EEPROM

http://www.microsemi.com/soc/documents/EmbeddedSRAMInit_AN.pdf

List of Changes

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

Date	Changes	Page
July 2010	This chapter is no longer published separately with its own part number and version but is now part of several FPGA fabric user's guides.	N/A
v2.3 (November 2009)	The "Sleep Mode" section was revised to state the VJTAG and VPUMP, as well as VCC, are grounded during Sleep mode (SAR 22517).	32
	Figure 2-6 • Controlling Power-On/-Off State Using Microprocessor and Power FET and Figure 2-7 • Controlling Power-On/-Off State Using Microprocessor and Voltage Regulator were revised to show that VJTAG and VPUMP are powered off during Sleep mode.	33
v2.2 (December 2008)	IGLOO nano devices were added as a supported family.	N/A
	The "Prototyping for IGLOO and ProASIC3L Devices Using ProASIC3" section was removed, as these devices are now in production.	N/A
	The "Additional Power Conservation Techniques" section was revised to add RT ProASIC3 devices.	41
v2.0 (October 2008)	The "Flash*Freeze Management FSM" section was updated with the following information: 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.	37

Chip and Quadrant Global I/Os

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

Naming of Global I/Os

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

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

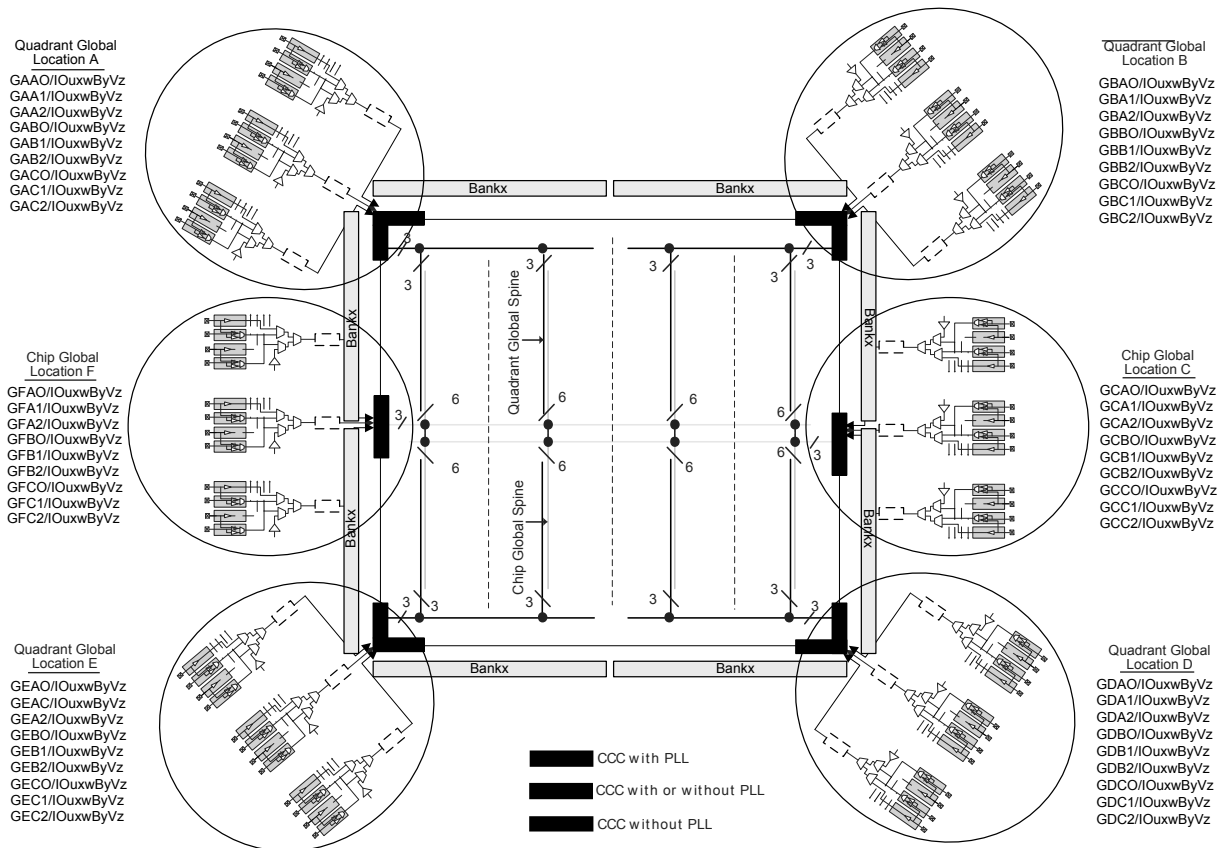


Figure 3-4 • Global Connections Details

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

Table 3-9 • I/O Standards within CLKBUF

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

Notes:

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

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

VHDL

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

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

Verilog

```
module design (____);

  input ____;
  output ____;

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

endmodule
```

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.

IGLOOe and ProASIC3E CCC Locations

IGLOOe and ProASIC3E 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 4-15).

All six CCCs are integrated with PLLs, except in PQFP-208 package devices. PQFP-208 package devices also have six CCCs, of which two include PLLs and four are simplified CCCs. The CCCs with PLLs are implemented in the middle of the east and west sides of the device (middle right and middle left). The simplified CCCs without PLLs are located in the four corners of the device (Figure 4-16).

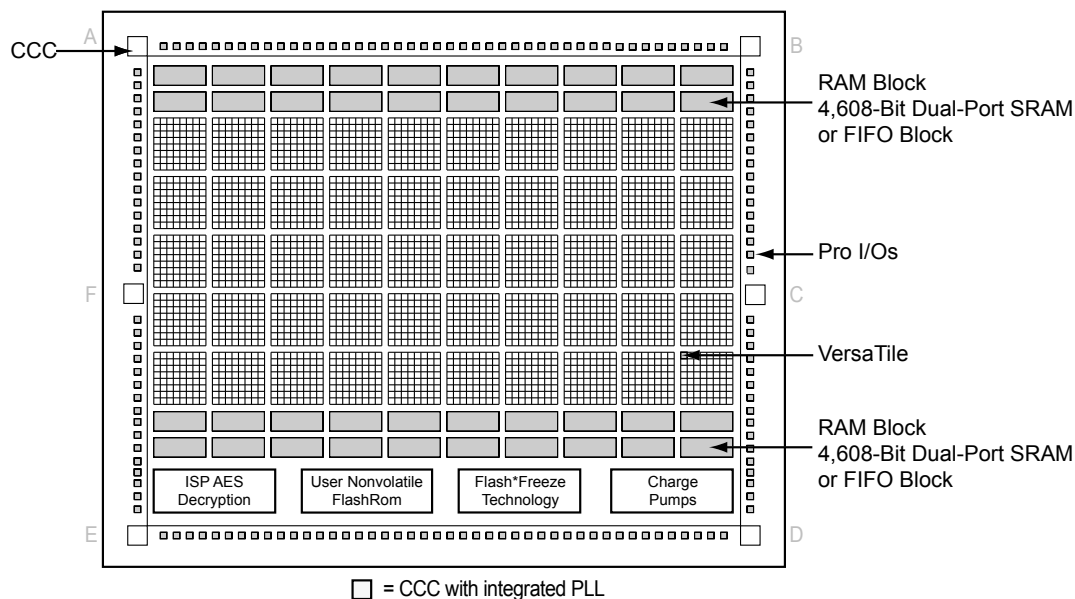


Figure 4-15 • CCC Locations in IGLOOe and ProASIC3E Family Devices (except PQFP-208 package)

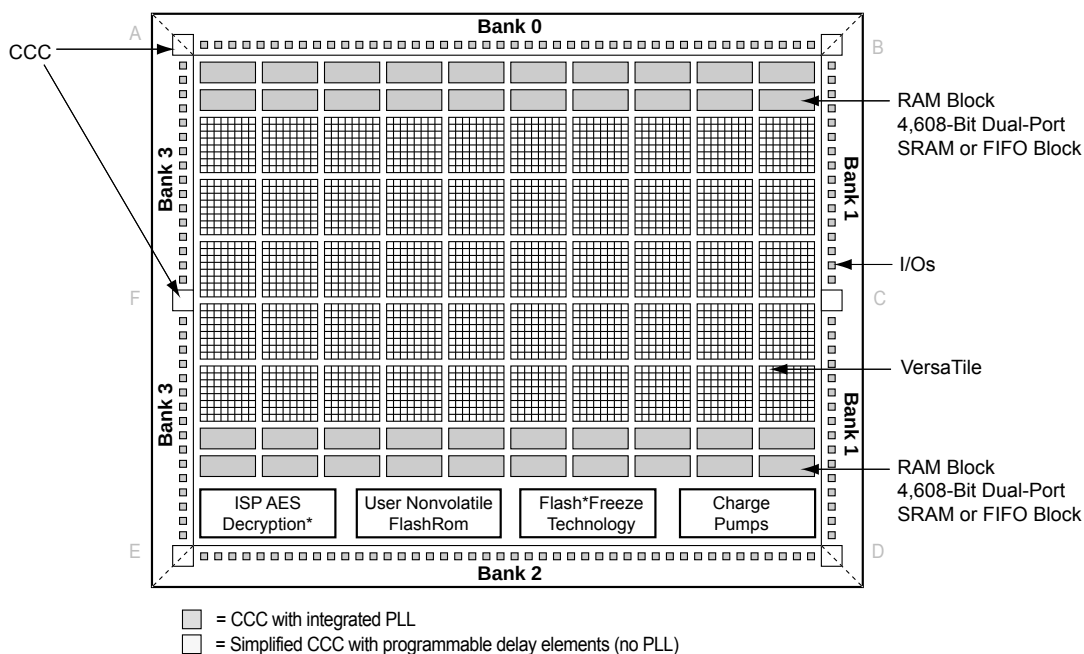


Figure 4-16 • CCC Locations in ProASIC3E Family Devices (PQFP-208 package)

DEVICE_INFO displays the FlashROM content, serial number, Design Name, and checksum, as shown below:

```
EXPORT IDC0DE[32] = 123261CF
EXPORT SILSIG[32] = 00000000
User information :
CHECKSUM: 61A0
Design Name:      TOP
Programming Method: STAPL
Algorithm Version: 1
Programmer: UNKNOWN
=====
FlashROM Information :
EXPORT Region_7_0[128] = FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
=====
Security Setting :
Encrypted FlashROM Programming Enabled.
Encrypted FPGA Array Programming Enabled.
=====
```

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

Custom Serialization Using FlashROM

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

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

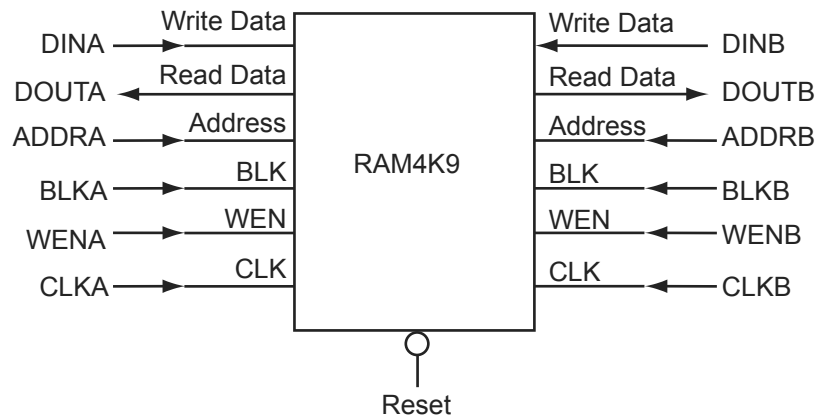
1. Generate FlashROM using SmartGen. From the Properties section in the FlashROM Settings dialog box, select the **Auto Inc** or **Read From File** region. For the Auto Inc region, specify the desired step value. You will not be able to modify this value in the FlashPoint software.
2. Go through the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

SRAM Features

RAM4K9 Macro

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

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



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

Figure 6-4 • RAM4K9 Simplified Configuration

Signal Descriptions for RAM4K9

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

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

WIDTHA and WIDTHB

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

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

FIFO Flag Usage Considerations

The AEVAL and AFVAL pins are used to specify the 12-bit AEMPTY and AFULL threshold values. The FIFO contains separate 12-bit write address (WADDR) and read address (RADDR) counters. WADDR is incremented every time a write operation is performed, and RADDR is incremented every time a read operation is performed. Whenever the difference between WADDR and RADDR is greater than or equal to AFVAL, the AFULL output is asserted. Likewise, whenever the difference between WADDR and RADDR is less than or equal to AEVAL, the AEMPTY output is asserted. To handle different read and write aspect ratios, AFVAL and AEVAL are expressed in terms of total data bits instead of total data words. When users specify AFVAL and AEVAL in terms of read or write words, the SmartGen tool translates them into bit addresses and configures these signals automatically. SmartGen configures the AFULL flag to assert when the write address exceeds the read address by at least a predefined value. In a 2k×8 FIFO, for example, a value of 1,500 for AFVAL means that the AFULL flag will be asserted after a write when the difference between the write address and the read address reaches 1,500 (there have been at least 1,500 more writes than reads). It will stay asserted until the difference between the write and read addresses drops below 1,500.

The AEMPTY flag is asserted when the difference between the write address and the read address is less than a predefined value. In the example above, a value of 200 for AEVAL means that the AEMPTY flag will be asserted when a read causes the difference between the write address and the read address to drop to 200. It will stay asserted until that difference rises above 200. Note that the FIFO can be configured with different read and write widths; in this case, the AFVAL setting is based on the number of write data entries, and the AEVAL setting is based on the number of read data entries. For aspect ratios of 512×9 and 256×18, only 4,096 bits can be addressed by the 12 bits of AFVAL and AEVAL. The number of words must be multiplied by 8 and 16 instead of 9 and 18. The SmartGen tool automatically uses the proper values. To avoid halfwords being written or read, which could happen if different read and write aspect ratios were specified, the FIFO will assert FULL or EMPTY as soon as at least one word cannot be written or read. For example, if a two-bit word is written and a four-bit word is being read, the FIFO will remain in the empty state when the first word is written. This occurs even if the FIFO is not completely empty, because in this case, a complete word cannot be read. The same is applicable in the full state. If a four-bit word is written and a two-bit word is read, the FIFO is full and one word is read. The FULL flag will remain asserted because a complete word cannot be written at this point.

Variable Aspect Ratio and Cascading

Variable aspect ratio and cascading allow users to configure the memory in the width and depth required. The memory block can be configured as a FIFO by combining the basic memory block with dedicated FIFO controller logic. The FIFO macro is named FIFO4KX18. Low power flash device RAM can be configured as 1, 2, 4, 9, or 18 bits wide. By cascading the memory blocks, any multiple of those widths can be created. The RAM blocks can be from 256 to 4,096 bits deep, depending on the aspect ratio, and the blocks can also be cascaded to create deeper areas. Refer to the aspect ratios available for each macro cell in the "SRAM Features" section on page 153. The largest continuous configurable memory area is equal to half the total memory available on the device, because the RAM is separated into two groups, one on each side of the device.

The SmartGen core generator will automatically configure and cascade both RAM and FIFO blocks. Cascading is accomplished using dedicated memory logic and does not consume user gates for depths up to 4,096 bits deep and widths up to 18, depending on the configuration. Deeper memory will utilize some user gates to multiplex the outputs.

Generated RAM and FIFO macros can be created as either structural VHDL or Verilog for easy instantiation into the design. Users of Libero SoC can create a symbol for the macro and incorporate it into a design schematic.

Table 6-10 on page 163 shows the number of memory blocks required for each of the supported depth and width memory configurations, and for each depth and width combination. For example, a 256-bit deep by 32-bit wide two-port RAM would consist of two 256×18 RAM blocks. The first 18 bits would be stored in the first RAM block, and the remaining 14 bits would be implemented in the other 256×18 RAM block. This second RAM block would have four bits of unused storage. Similarly, a dual-port memory block that is 8,192 bits deep and 8 bits wide would be implemented using 16 memory blocks. The dual-port memory would be configured in a 4,096×1 aspect ratio. These blocks would then be cascaded two deep to achieve 8,192 bits of depth, and eight wide to achieve the eight bits of width.

Simultaneously Switching Outputs (SSOs) and Printed Circuit Board Layout

Each I/O voltage bank has a separate ground and power plane for input and output circuits (VMV/GNDQ for input buffers and VCCI/GND for output buffers). This isolation is necessary to minimize simultaneous switching noise from the input and output (SSI and SSO). The switching noise (ground bounce and power bounce) is generated by the output buffers and transferred into input buffer circuits, and vice versa.

Since voltage bounce originates on the package inductance, the VMV and VCCI supplies have separate package pin assignments. For the same reason, GND and GNDQ also have separate pin assignments.

The VMV and VCCI pins must be shorted to each other on the board. Also, the GND and GNDQ pins must be shorted to each other on the board. This will prevent unwanted current draw from the power supply.

SSOs can cause signal integrity problems on adjacent signals that are not part of the SSO bus. Both inductive and capacitive coupling parasitics of bond wires inside packages and of traces on PCBs will transfer noise from SSO busses onto signals adjacent to those busses. Additionally, SSOs can produce ground bounce noise and VCCI dip noise. These two noise types are caused by rapidly changing currents through GND and VCCI package pin inductances during switching activities (EQ 2 and EQ 3).

$$\text{Ground bounce noise voltage} = L(\text{GND}) \times di/dt \quad \text{EQ 2}$$

$$\text{VCCI dip noise voltage} = L(\text{VCCI}) \times di/dt \quad \text{EQ 3}$$

Any group of four or more input pins switching on the same clock edge is considered an SSO bus. The shielding should be done both on the board and inside the package unless otherwise described.

In-package shielding can be achieved in several ways; the required shielding will vary depending on whether pins next to the SSO bus are LVTTTL/LVCMOS inputs, LVTTTL/LVCMOS outputs, or GTL/SSTL/HSTL/LVDS/LVPECL inputs and outputs. Board traces in the vicinity of the SSO bus have to be adequately shielded from mutual coupling and inductive noise that can be generated by the SSO bus. Also, noise generated by the SSO bus needs to be reduced inside the package.

PCBs perform an important function in feeding stable supply voltages to the IC and, at the same time, maintaining signal integrity between devices.

Key issues that need to be considered are as follows:

- Power and ground plane design and decoupling network design
- Transmission line reflections and terminations

For extensive data per package on the SSO and PCB issues, refer to the "ProASIC3/E SSO and Pin Placement and Guidelines" chapter of the *ProASIC3 FPGA Fabric User's Guide*.

Conclusion

IGLOOe and ProASIC3E support for multiple I/O standards minimizes board-level components and makes possible a wide variety of applications. The Microsemi Designer software, integrated with Libero SoC, presents a clear visual display of I/O assignments, allowing users to verify I/O and board-level design requirements before programming the device. The IGLOOe and ProASIC3E device I/O features and functionalities ensure board designers can produce low-cost and low power FPGA applications fulfilling the complexities of contemporary design needs.

Related Documents

Application Notes

Board-Level Considerations

http://www.microsemi.com/soc/documents/ALL_AC276_AN.pdf

User's Guides

ProASIC3 FPGA Fabric User's Guide

http://www.microsemi.com/soc/documents/PA3_UG.pdf

ProASIC3E FPGA Fabric User's Guide

http://www.microsemi.com/soc/documents/PA3E_UG.pdf

IGLOOe FPGA Fabric User's Guide

http://www.microsemi.com/soc/documents/IGLOOe_UG.pdf

Libero SoC User's Guide

http://www.microsemi.com/soc/documents/libero_ug.pdf

IGLOO, Fusion, and ProASIC3 Macro Library Guide

http://www.microsemi.com/soc/documents/pa3_libguide_ug.pdf

SmartGen Core Reference Guide

http://www.microsemi.com/soc/documents/genguide_ug.pdf

Implementing I/Os in Microsemi Software

Microsemi Libero SoC software is integrated with design entry tools such as the SmartGen macro builder, the ViewDraw schematic entry tool, and an HDL editor. It is also integrated with the synthesis and Designer tools. In this section, all necessary steps to implement the I/Os are discussed.

Design Entry

There are three ways to implement I/Os in a design:

1. Use the SmartGen macro builder to configure I/Os by generating specific I/O library macros and then instantiating them in top-level code. This is especially useful when creating I/O bus structures.
2. Use an I/O buffer cell in a schematic design.
3. Manually instantiate specific I/O macros in the top-level code.

If technology-specific macros, such as `INBUF_LVCMOS33` and `OUTBUF_PCI`, are used in the HDL code or schematic, the user will not be able to change the I/O standard later on in Designer. If generic I/O macros are used, such as `INBUF`, `OUTBUF`, `TRIBUF`, `CLKBUF`, and `BIBUF`, the user can change the I/O standard using the Designer I/O Attribute Editor tool.

Using SmartGen for I/O Configuration

The SmartGen tool in Libero SoC provides a GUI-based method of configuring the I/O attributes. The user can select certain I/O attributes while configuring the I/O macro in SmartGen. The steps to configure an I/O macro with specific I/O attributes are as follows:

1. Open Libero SoC.
2. On the left-hand side of the Catalog View, select **I/O**, as shown in Figure 9-2.

Figure 9-2 • SmartGen Catalog

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 9-17). 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 9-16).

Figure 9-16 • 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 9-17.

Figure 9-17 • I/O Bank Assigner Displays Messages in Output Window

List of Changes

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

Date	Changes	Page
August 2012	The notes in Table 9-2 • Designer State (resulting from I/O attribute modification) were revised to clarify which device families support programmable input delay (SAR 39666).	253
June 2011	Figure 9-2 • SmartGen Catalog was updated (SAR 24310). Figure 8-3 • Expanded I/O Section and the step associated with it were deleted to reflect changes in the software.	254
	The following rule was added to the "VREF Rules for the Implementation of Voltage-Referenced I/O Standards" section: Only minibanks that contain input or bidirectional I/Os require a VREF. A VREF is not needed for minibanks composed of output or tristated I/Os (SAR 24310).	265
July 2010	Notes were added where appropriate to point out that IGLOO nano and ProASIC3 nano devices do not support differential inputs (SAR 21449).	N/A
v1.4 (December 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 9-1 • Flash-Based FPGAs.	252
	The notes for Table 9-2 • Designer State (resulting from I/O attribute modification) were revised to indicate that skew control and input delay do not apply to nano devices.	253
v1.3 (October 2008)	The "Flash FPGAs I/O Support" section was revised to include new families and make the information more concise.	252
v1.2 (June 2008)	The following changes were made to the family descriptions in Table 9-1 • Flash-Based FPGAs: - ProASIC3L was updated to include 1.5 V. - The number of PLLs for ProASIC3E was changed from five to six.	252
v1.1 (March 2008)	This document was previously part of the <i>I/O Structures in IGLOO and ProASIC3 Devices</i> document. The content was separated and made into a new document.	N/A
	Table 9-2 • Designer State (resulting from I/O attribute modification) was updated to include note 2 for IGLOO PLUS.	253

Design Example

Figure 10-9 shows a simple example of a design using both DDR input and DDR output registers. The user can copy the HDL code in Libero SoC software and go through the design flow. Figure 10-10 and Figure 10-11 on page 283 show the netlist and ChipPlanner views of the ddr_test design. Diagrams may vary slightly for different families.

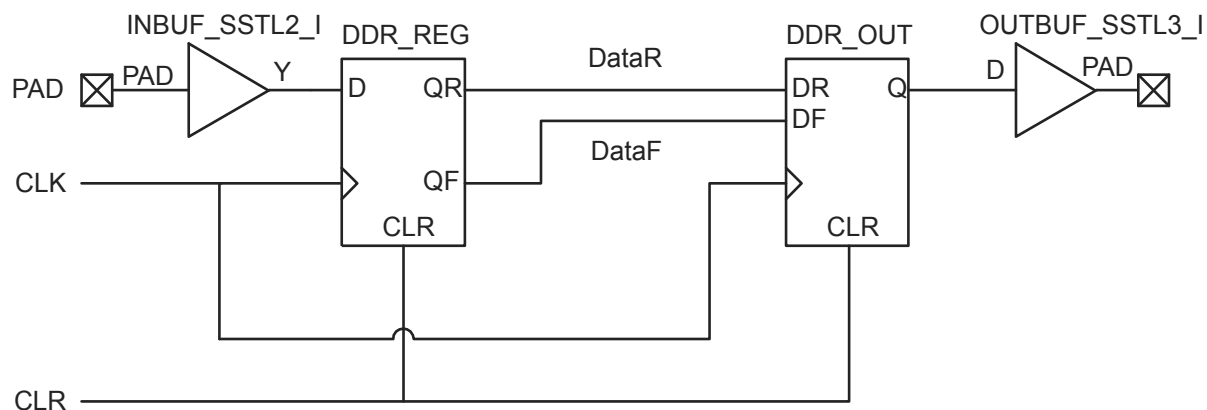


Figure 10-9 • Design Example

Figure 10-10 • DDR Test Design as Seen by NetlistViewer for IGLOO/e Devices

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 <i>Programming and Functional Failure Guidelines</i> 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

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

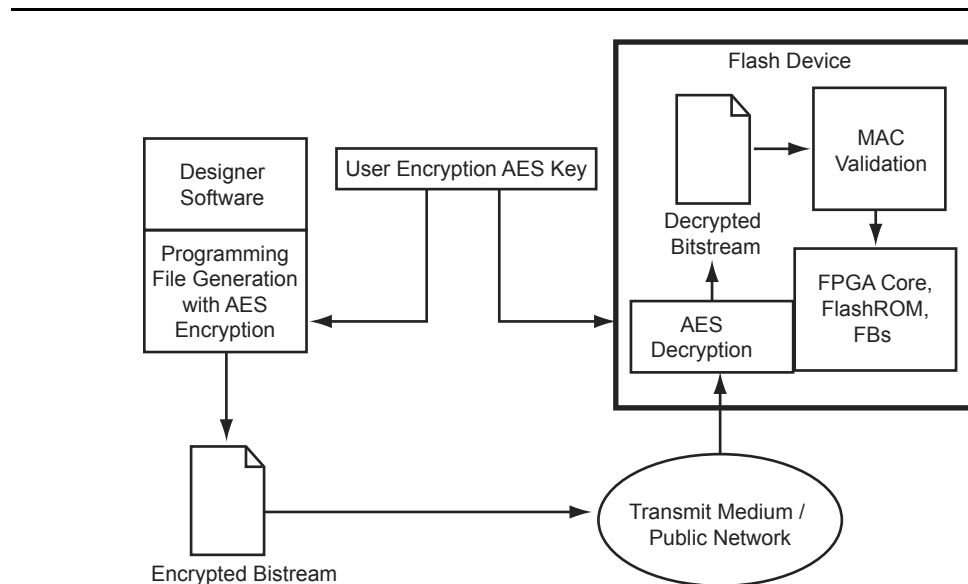


Figure 13-1 • AES-128 Security Features

Circuit Verification

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

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

Verification Steps

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

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

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

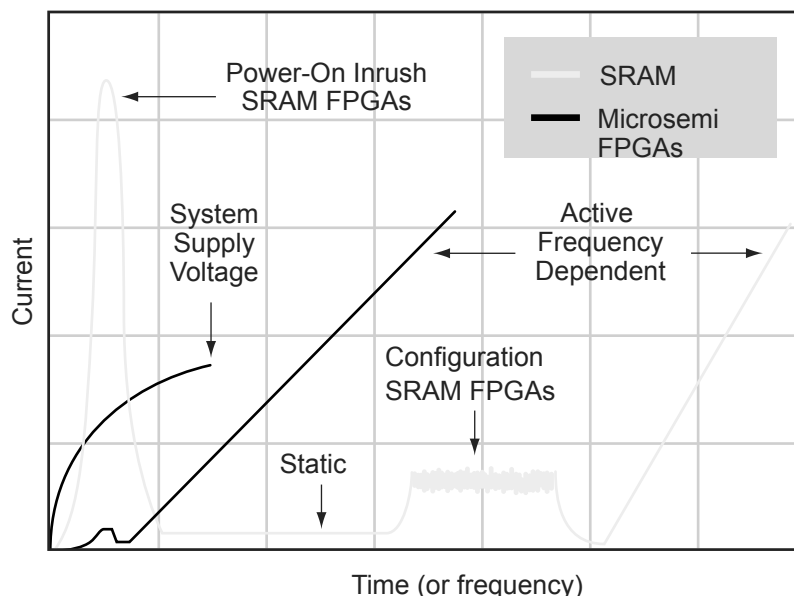


Figure 18-1 • Types of Power Consumption in SRAM FPGAs and Microsemi Nonvolatile FPGAs

Transient Current on VCC

The characterization of the transient current on VCC is performed on nearly all devices within the IGLOO, ProASIC3L, and ProASIC3 families. A sample size of five units is used from each device family member. All the device I/Os are internally pulled down while the transient current measurements are performed. For ProASIC3 devices, the measurements at typical conditions show that the maximum transient current on VCC, when the power supply is powered at ramp-rates ranging from 15 V/ms to 0.15 V/ms, does not exceed the maximum standby current specified in the device datasheets. Refer to the DC and Switching Characteristics chapters of the *ProASIC3 Flash Family FPGAS* datasheet and *ProASIC3E Flash Family FPGAs* datasheet for more information.

Similarly, IGLOO, IGLOO nano, IGLOO PLUS, and ProASIC3L devices exhibit very low transient current on VCC. The transient current does not exceed the typical operating current of the device while in active mode. For example, the characterization of AGL600-FG256 V2 and V5 devices has shown that the transient current on VCC is typically in the range of 1–5 mA.

Transient Current on VCCI

The characterization of the transient current on VCCI is performed on devices within the IGLOO, IGLOO nano, IGLOO PLUS, ProASIC3, ProASIC3 nano, and ProASIC3L groups of devices, similarly to VCC transient current measurements. For ProASIC3 devices, the measurements at typical conditions show that the maximum transient current on VCCI, when the power supply is powered at ramp-rates ranging from 33 V/ms to 0.33 V/ms, does not exceed the maximum standby current specified in the device datasheet. Refer to the DC and Switching Characteristics chapters of the *ProASIC3 Flash Family FPGAS* datasheet and *ProASIC3E Flash Family FPGAs* datasheet for more information.

Similarly, IGLOO, IGLOO PLUS, and ProASIC3L devices exhibit very low transient current on VCCI. The transient current does not exceed the typical operating current of the device while in active mode. For example, the characterization of AGL600-FG256 V2 and V5 devices has shown that the transient current on VCCI is typically in the range of 1–2 mA.

Brownout Voltage

Brownout is a condition in which the voltage supplies are lower than normal, causing the device to malfunction as a result of insufficient power. In general, Microsemi does not guarantee the functionality of the design inside the flash FPGA if voltage supplies are below their minimum recommended operating condition. Microsemi has performed measurements to characterize the brownout levels of FPGA power supplies. Refer to Table 18-3 for device-specific brownout deactivation levels. For the purpose of characterization, a direct path from the device input to output is monitored while voltage supplies are lowered gradually. The brownout point is defined as the voltage level at which the output stops following the input. Characterization tests performed on several IGLOO, ProASIC3L, and ProASIC3 devices in typical operating conditions showed the brownout voltage levels to be within the specification.

During device power-down, the device I/Os become tristated once the first supply in the power-down sequence drops below its brownout deactivation voltage.

Table 18-3 • Brownout Deactivation Levels for VCC and VCCI

Devices	VCC Brownout Deactivation Level (V)	VCCI Brownout Deactivation Level (V)
ProASIC3, ProASIC3 nano, IGLOO, IGLOO nano, IGLOO PLUS and ProASIC3L devices running at VCC = 1.5 V	0.75 V $\pm$ 0.25 V	0.8 V $\pm$ 0.3 V
IGLOO, IGLOO nano, IGLOO PLUS, and ProASIC3L devices running at VCC = 1.2 V	0.75 V $\pm$ 0.2 V	0.8 V $\pm$ 0.15 V

PLL Behavior at Brownout Condition

When PLL power supply voltage and/or V_{CC} levels drop below the V_{CC} brownout levels mentioned above for 1.5 V and 1.2 V devices, the PLL output lock signal goes LOW and/or the output clock is lost. The following sections explain PLL behavior during and after the brownout condition.

VCCPLL and VCC Tied Together

In this condition, both VCC and VCCPLL drop below the 0.75 V ($\pm$ 0.25 V or $\pm$ 0.2 V) brownout level. During the brownout recovery, once VCCPLL and VCC reach the activation point (0.85 $\pm$ 0.25 V or $\pm$ 0.2 V) again, the PLL output lock signal may still remain LOW with the PLL output clock signal toggling. If this condition occurs, there are two ways to recover the PLL output lock signal:

1. Cycle the power supplies of the PLL (power off and on) by using the PLL POWERDOWN signal.
2. Turn off the input reference clock to the PLL and then turn it back on.

Only VCCPLL Is at Brownout

In this case, only VCCPLL drops below the 0.75 V ($\pm$ 0.25 V or $\pm$ 0.2 V) brownout level and the VCC supply remains at nominal recommended operating voltage (1.5 V $\pm$ 0.075 V for 1.5 V devices and 1.2 V $\pm$ 0.06 V for 1.2 V devices). In this condition, the PLL behavior after brownout recovery is similar to initial power-up condition, and the PLL will regain lock automatically after VCCPLL is ramped up above the activation level (0.85 $\pm$ 0.25 V or $\pm$ 0.2 V). No intervention is necessary in this case.

Only VCC Is at Brownout

In this condition, VCC drops below the 0.75 V ($\pm$ 0.25 V or $\pm$ 0.2 V) brownout level and VCCPLL remains at nominal recommended operating voltage (1.5 V $\pm$ 0.075 V for 1.5 V devices and 1.2 V $\pm$ 0.06 V for 1.2 V devices). During the brownout recovery, once VCC reaches the activation point again (0.85 $\pm$ 0.25 V or $\pm$ 0.2 V), the PLL output lock signal may still remain LOW with the PLL output clock signal toggling. If this condition occurs, there are two ways to recover the PLL output lock signal:

1. Cycle the power supplies of the PLL (power off and on) by using the PLL POWERDOWN signal.
2. Turn off the input reference clock to the PLL and then turn it back on.

It is important to note that Microsemi recommends using a monotonic power supply or voltage regulator to ensure proper power-up behavior.