Microchip Technology - A3P600L-FGG484I 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	110592
Number of I/O	235
Number of Gates	600000
Voltage - Supply	1.14V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 100°C (TJ)
Package / Case	484-BGA
Supplier Device Package	484-FPBGA (23x23)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/a3p600l-fgg484i

1 – FPGA Array Architecture in Low Power Flash Devices

Device Architecture

Advanced Flash Switch

Unlike SRAM FPGAs, the low power flash devices use a live-at-power-up ISP flash switch as their programming element. Flash cells are distributed throughout the device to provide nonvolatile, reconfigurable programming to connect signal lines to the appropriate VersaTile inputs and outputs. In the flash switch, two transistors share the floating gate, which stores the programming information (Figure 1-1). One is the sensing transistor, which is only used for writing and verification of the floating gate voltage. The other is the switching transistor. The latter is used to connect or separate routing nets, or to configure VersaTile logic. It is also used to erase the floating gate. Dedicated high-performance lines are connected as required using the flash switch for fast, low-skew, global signal distribution throughout the device core. Maximum core utilization is possible for virtually any design. The use of the flash switch technology also removes the possibility of firm errors, which are increasingly common in SRAM-based FPGAs.

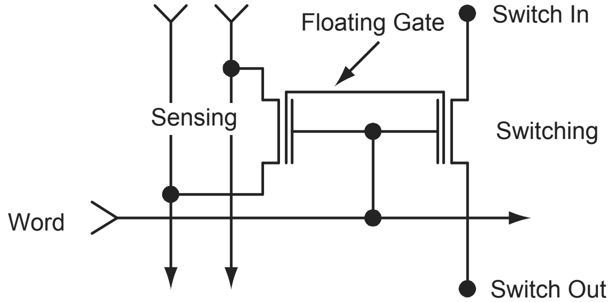
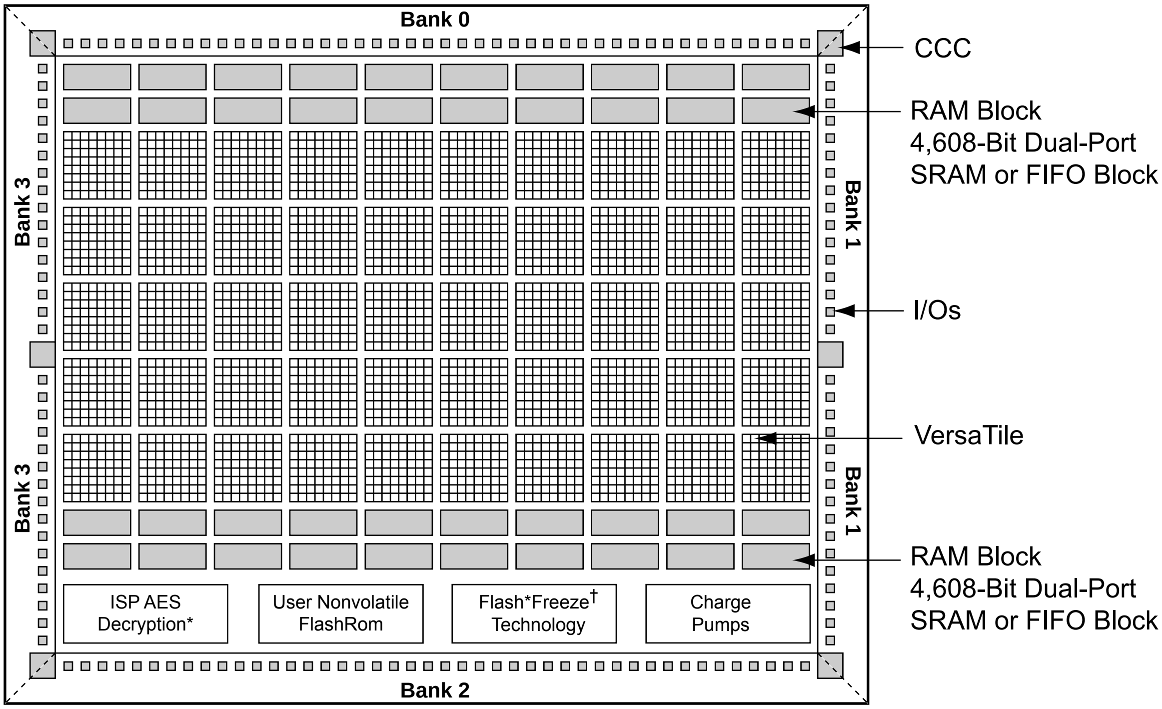
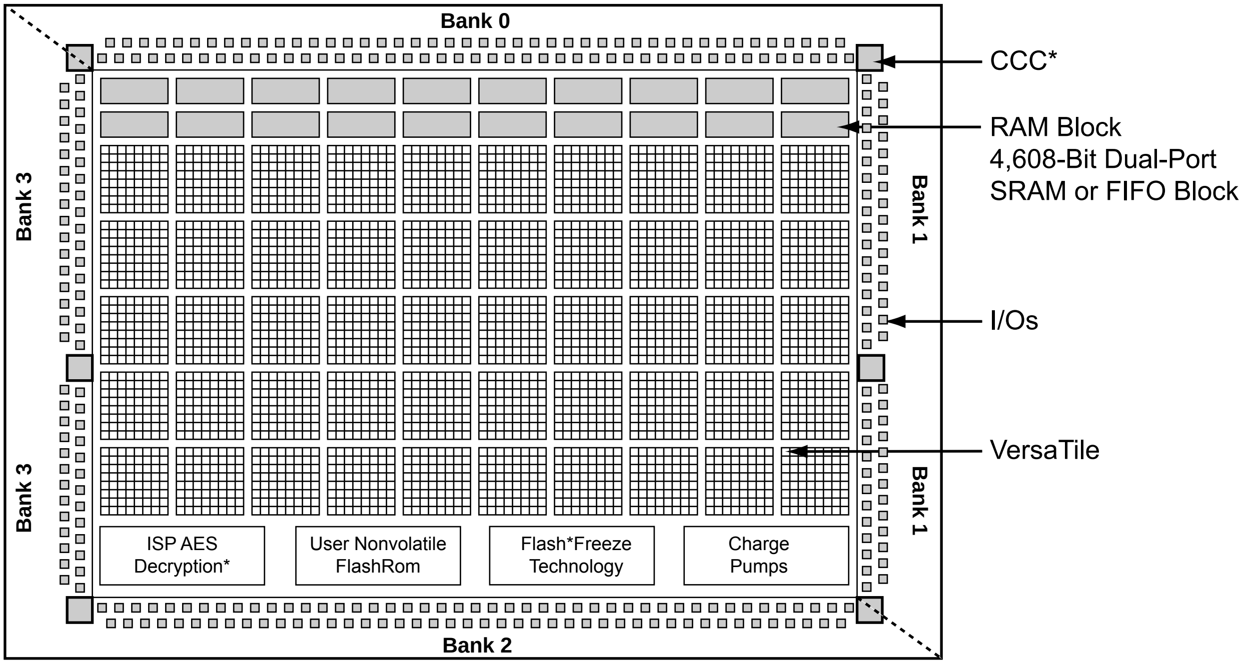


***Figure 1-1* • Flash-Based Switch**



*Note: Flash*Freeze technology only applies to IGLOO and ProASIC3L families.*

Figure 1-5 • **IGLOO, IGLOO nano, ProASIC3 nano, and ProASIC3/L Device Architecture Overview with Four I/O Banks (AGL600 device is shown)**



*Note: * AGLP030 does not contain a PLL or support AES security.*

Figure 1-6 • **IGLOO PLUS Device Architecture Overview with Four I/O Banks**

- 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

Table 3-5 • **Globals/Spines/Rows for IGLOO PLUS Devices**

IGLOO PLUS Devices	Chip Globals	Quadrant Globals (4×3)	Clock Trees	Globals/ Spines per Tree	Total Spines per Device	VersaTiles in Each Tree	Total VersaTiles	Rows in Each Spine
AGLP030	6	0	2	9	18	384*	792	12
AGLP060	6	12	4	9	36	384*	1,584	12
AGLP125	6	12	8	9	72	384*	3,120	12

*Note: *Clock trees that are located at far left and far right will support more VersaTiles.*

Table 3-6 • **Globals/Spines/Rows for Fusion Devices**

Fusion Device	Chip Globals	Quadrant Globals (4×3)	Clock Trees	Globals/ Spines per Tree	Total Spines per Device	VersaTiles in Each Tree	Total VersaTiles	Rows in Each Spine
AFS090	6	12	6	9	54	384	2,304	12
AFS250	6	12	8	9	72	768	6,144	24
AFS600	6	12	12	9	108	1,152	13,824	36
AFS1500	6	12	20	9	180	1,920	38,400	60

Using Clock Aggregation

Clock aggregation allows for multi-spine clock domains to be assigned using hardwired connections, without adding any extra skew. A MUX tree, shown in Figure 3-8, provides the necessary flexibility to allow long lines, local resources, or I/Os to access domains of one, two, or four global spines. Signal access to the clock aggregation system is achieved through long-line resources in the central rib in the center of the die, and also through local resources in the north and south ribs, allowing I/Os to feed directly into the clock system. As Figure 3-9 indicates, this access system is contiguous.

There is no break in the middle of the chip for the north and south I/O VersaNet access. This is different from the quadrant clocks located in these ribs, which only reach the middle of the rib.

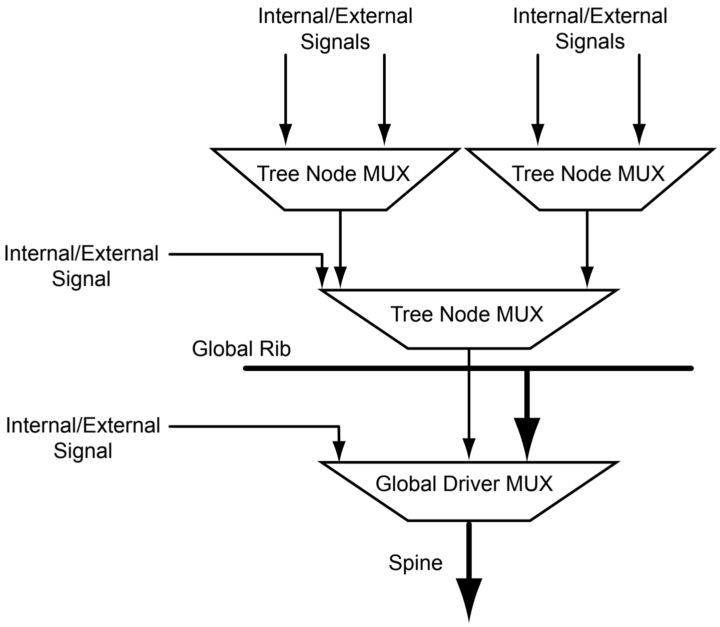


Figure 3-8 • **Spine Selection MUX of Global Tree**

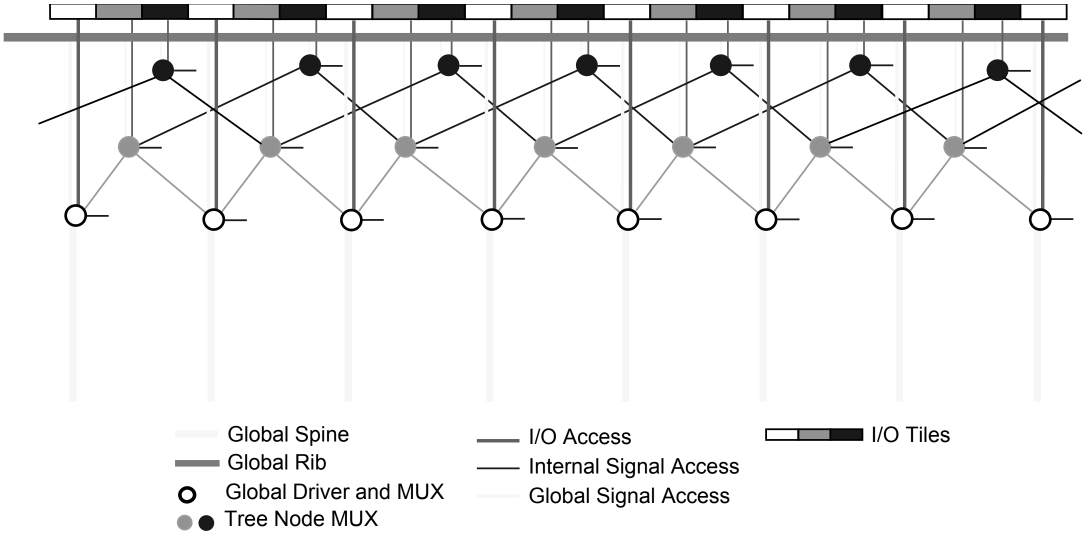


Figure 3-9 • **Clock Aggregation Tree Architecture**

Design Recommendations

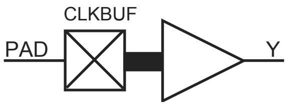
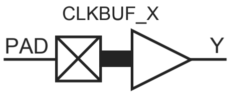
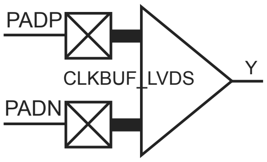
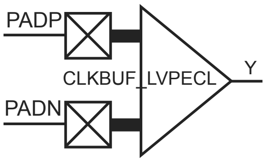
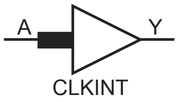
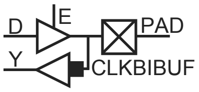
The following sections provide design flow recommendations for using a global network in a design.

- "Global Macros and I/O Standards"
- "Global Macro and Placement Selections" on page 64
- "Using Global Macros in Synplicity" on page 66
- "Global Promotion and Demotion Using PDC" on page 67
- "Spine Assignment" on page 68
- "Designer Flow for Global Assignment" on page 69
- "Simple Design Example" on page 71
- "Global Management in PLL Design" on page 73
- "Using Spines of Occupied Global Networks" on page 74

Global Macros and I/O Standards

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

***Table 3-8* • Clock Macros**

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

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

During Layout, Designer will assign two of the signals to quadrant global locations.

Step 3 (optional)

You can also assign the QCLK1_c and QCLK2_c nets to quadrant regions using the following PDC commands:

```
assign_local_clock –net QCLK1_c  –type quadrant UL
assign_local_clock –net QCLK2_c  –type quadrant LL
```

Step 4

Import this PDC with the netlist and run Compile again. You will see the following in the Compile report:

```
The following nets have been assigned to a global resource:
Fanout  Type          Name
--------------------------
1536    INT_NET       Net   : EN_ALL_c
                      Driver: EN_ALL_pad_CLKINT
                      Source: AUTO PROMOTED
1536    SET/RESET_NET Net   : ACLR_c
                      Driver: ACLR_pad_CLKINT
                      Source: AUTO PROMOTED
256     CLK_NET       Net   : QCLK3_c
                      Driver: QCLK3_pad_CLKINT
                      Source: AUTO PROMOTED
256     CLK_NET       Net   : $1N14
                      Driver: $1I5/Core
                      Source: ESSENTIAL
256     CLK_NET       Net   : $1N12
                      Driver: $1I6/Core
                      Source: ESSENTIAL
256     CLK_NET       Net   : $1N10
                      Driver: $1I6/Core
                      Source: ESSENTIAL
The following nets have been assigned to a quadrant clock resource using PDC:
Fanout  Type          Name
--------------------------
256     CLK_NET       Net   : QCLK1_c
                      Driver: QCLK1_pad_CLKINT
                      Region: quadrant_UL
256     CLK_NET       Net   : QCLK2_c
                      Driver: QCLK2_pad_CLKINT
                      Region: quadrant_LL
```

Step 5

Run Layout.

Global Management in PLL Design

This section describes the legal global network connections to PLLs in the low power flash devices. For detailed information on using PLLs, refer to "Clock Conditioning Circuits in Low Power Flash Devices and Mixed Signal FPGAs" section on page 77. Microsemi recommends that you use the dedicated global pins to directly drive the reference clock input of the associated PLL for reduced propagation delays and clock distortion. However, low power flash devices offer the flexibility to connect other signals to reference clock inputs. Each PLL is associated with three global networks (Figure 3-5 on page 52). There are some limitations, such as when trying to use the global and PLL at the same time:

- If you use a PLL with only primary output, you can still use the remaining two free global networks.
- If you use three globals associated with a PLL location, you cannot use the PLL on that location.
- If the YB or YC output is used standalone, it will occupy one global, even though this signal does not go to the global network.

Table 4-13 **• 2-Bit Feedback MUX**

FBSEL<1:0> State	MUX Input Selected
0	Ground. Used for power-down mode in power-down logic block.
1	PLL VCO 0° phase shift
2	PLL delayed VCO 0° phase shift
3	N/A

Table 4-14 **• Programmable Delay Selection for Feedback Delay and Secondary Core Output Delays**

FBDLY<4:0>; DLYYB<4:0>; DLYYC<4:0> State	Delay Value
0	Typical delay = 600 ps
1	Typical delay = 760 ps
2	Typical delay = 920 ps
⋮	⋮
31	Typical delay = 5.56 ns

Table 4-15 **• Programmable Delay Selection for Global Clock Output Delays**

DLYGLA<4:0>; DLYGLB<4:0>; DLYGLC<4:0> State	Delay Value
0	Typical delay = 225 ps
1	Typical delay = 760 ps
2	Typical delay = 920 ps
⋮	⋮
31	Typical delay = 5.56 ns

Table 4-16 **• Fusion Dynamic CCC Clock Source Selection**

RXASEL	DYNASEL	Source of CLKA
1	0	RC Oscillator
1	1	Crystal Oscillator
RXBSEL	**DYNBSEL**	**Source of CLKB**
1	0	RC Oscillator
1	1	Crystal Oscillator
RXBSEL	**DYNCSEL**	**Source of CLKC**
1	0	RC Oscillator
1	1	Crystal Oscillator

Table 4-17 **• Fusion Dynamic CCC NGMUX Configuration**

GLMUXCFG<1:0>	NGMUX Select Signal	Supported Input Clocks to NGMUX
00	0	GLA
	1	GLC
01	0	GLA
	1	GLINT
10	0	GLC
	1	GLINT

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

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

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

Custom Serialization Using FlashROM

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

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

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

Conclusion

The Fusion, IGLOO, and ProASIC3 families are the only FPGAs that offer on-chip FlashROM support. This document presents information on the FlashROM architecture, possible applications, programming, access through the JTAG and UJTAG interface, and integration into your design. In addition, the Libero tool set enables easy creation and modification of the FlashROM content.

The nonvolatile FlashROM block in the FPGA can be customized, enabling multiple applications.

Additionally, the security offered by the low power flash devices keeps both the contents of FlashROM and the FPGA design safe from system over-builders, system cloners, and IP thieves.

Related Documents

User's Guides

FlashPro User's Guide

http://www.microsemi.com/documents/FlashPro_UG.pdf

List of Changes

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

Date	Changes	Page
July 2010	This chapter is no longer published separately with its own part number and version but is now part of several FPGA fabric user's guides.	N/A
v1.4 (December 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 5-1 • Flash-Based FPGAs.	134
v1.3 (October 2008)	The "FlashROM Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	134
	Figure 5-2 • Fusion Device Architecture Overview (AFS600) was replaced. Figure 5-5 • Programming FlashROM Using AES was revised to change "Fusion" to "Flash Device."	135, 137
	The *FlashPoint User's Guide* was removed from the "User's Guides" section, as its content is now part of the *FlashPro User's Guide*.	146
v1.2 (June 2008)	The following changes were made to the family descriptions in Table 5-1 • Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	134
v1.1 (March 2008)	The chapter was updated to include the IGLOO PLUS family and information regarding 15 k gate devices. The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	N/A

recommended, since it reduces the complexity of the user interface block and the board-level JTAG driver.

Moreover, using an internal counter for address generation speeds up the initialization procedure, since the user only needs to import the data through the JTAG port.

The designer may use different methods to select among the multiple RAM blocks. Using counters along with demultiplexers is one approach to set the write enable signals. Basically, the number of RAM blocks needing initialization determines the most efficient approach. For example, if all the blocks are initialized with the same data, one enable signal is enough to activate the write procedure for all of them at the same time. Another alternative is to use different opcodes to initialize each memory block. For a small number of RAM blocks, using counters is an optimal choice. For example, a ring counter can be used to select from multiple RAM blocks. The clock driver of this counter needs to be controlled by the address generation process.

Once the addressing of one block is finished, a clock pulse is sent to the (ring) counter to select the next memory block.

Figure 6-9 illustrates a simple block diagram of an interface block between UJTAG and RAM blocks.

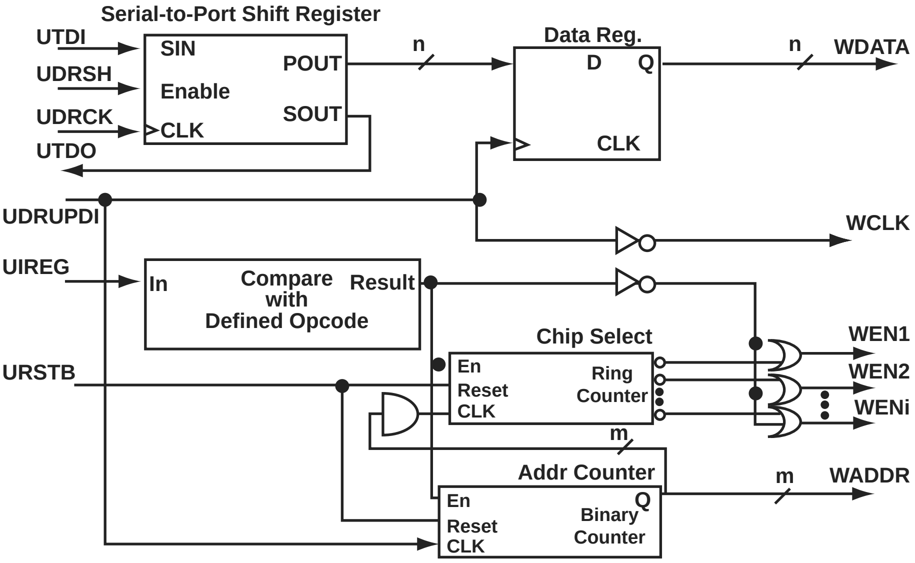


Figure 6-9 • **Block Diagram of a Sample User Interface**

In the circuit shown in Figure 6-9, the shift register is enabled by the UDRSH output of the UJTAG macro. The counters and chip select outputs are controlled by the value of the TAP Instruction Register. The comparison block compares the UIREG value with the "start initialization" opcode value (defined by the user). If the result is true, the counters start to generate addresses and activate the WEN inputs of appropriate RAM blocks.

The UDRUPD output of the UJTAG macro, also shown in Figure 6-9, is used for generating the write clock (WCLK) and synchronizing the data register and address counter with WCLK. UDRUPD is HIGH when the TAP Controller is in the Data Register Update state, which is an indication of completing the loading of one data word. Once the TAP Controller goes into the Data Register Update state, the UDRUPD output of the UJTAG macro goes HIGH. Therefore, the pipeline register and the address counter place the proper data and address on the outputs of the interface block. Meanwhile, WCLK is defined as the inverted UDRUPD. This will provide enough time (equal to the UDRUPD HIGH time) for the data and address to be placed at the proper ports of the RAM block before the rising edge of WCLK. The inverter is not required if the RAM blocks are clocked at the falling edge of the write clock. An example of this is described in the "Example of RAM Initialization" section on page 166.

IGLOO and ProASIC3

For boards and cards with three levels of staging, card power supplies must have time to reach their final values before the I/Os are connected. Pay attention to the sizing of power supply decoupling capacitors on the card to ensure that the power supplies are not overloaded with capacitance.

Cards with three levels of staging should have the following sequence:

- Grounds
- Powers
- I/Os and other pins

For Level 3 and Level 4 compliance with the 30K gate device, cards with two levels of staging should have the following sequence:

- Grounds
- Powers, I/Os, and other pins

Cold-Sparing Support

Cold-sparing refers to the ability of a device to leave system data undisturbed when the system is powered up, while the component itself is powered down, or when power supplies are floating.

The resistor value is calculated based on the decoupling capacitance on a given power supply. The RC constant should be greater than 3 µs.

To remove resistor current during operation, it is suggested that the resistor be disconnected (e.g., with an NMOS switch) from the power supply after the supply has reached its final value. Refer to the "Power-Up/-Down Behavior of Low Power Flash Devices" section on page 373 for details on cold-sparing.

Cold-sparing means that a subsystem with no power applied (usually a circuit board) is electrically connected to the system that is in operation. This means that all input buffers of the subsystem must present very high input impedance with no power applied so as not to disturb the operating portion of the system.

The 30 k gate devices fully support cold-sparing, since the I/O clamp diode is always off (see Table 7-12 on page 193). If the 30 k gate device is used in applications requiring cold-sparing, a discharge path from the power supply to ground should be provided. This can be done with a discharge resistor or a switched resistor. This is necessary because the 30K gate devices do not have built-in I/O clamp diodes.

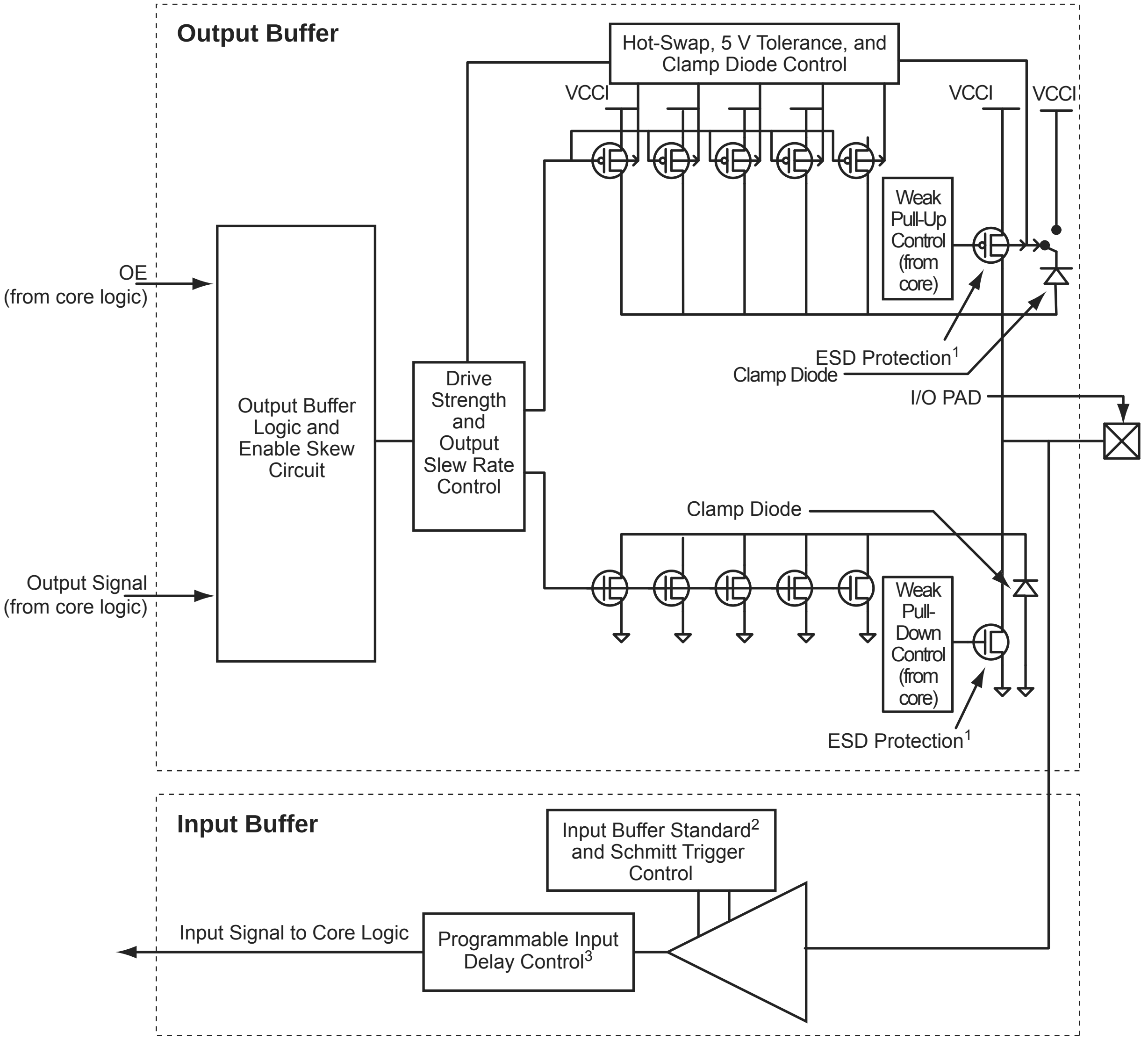
For other IGLOO and ProASIC3 devices, since the I/O clamp diode is always active, cold-sparing can be accomplished either by employing a bus switch to isolate the device I/Os from the rest of the system or by driving each I/O pin to 0 V. If the resistor is chosen, the resistor value must be calculated based on decoupling capacitance on a given power supply on the board (this decoupling capacitance is in parallel with the resistor). The RC time constant should ensure full discharge of supplies before cold-sparing functionality is required. The resistor is necessary to ensure that the power pins are discharged to ground every time there is an interruption of power to the device.

IGLOOe and ProASIC3E devices support cold-sparing for all I/O configurations. Standards, such as PCI, that require I/O clamp diodes can also achieve cold-sparing compliance, since clamp diodes get disconnected internally when the supplies are at 0 V.

When targeting low power applications, I/O cold-sparing may add additional current if a pin is configured with either a pull-up or pull-down resistor and driven in the opposite direction. A small static current is induced on each I/O pin when the pin is driven to a voltage opposite to the weak pull resistor. The current is equal to the voltage drop across the input pin divided by the pull resistor. Refer to the "Detailed I/O DC Characteristics" section of the appropriate family datasheet for the specific pull resistor value for the corresponding I/O standard.

For example, assuming an LVTTL 3.3 V input pin is configured with a weak pull-up resistor, a current will flow through the pull-up resistor if the input pin is driven LOW. For LVTTL 3.3 V, the pull-up resistor is ~45 kΩ, and the resulting current is equal to 3.3 V / 45 kΩ = 73 µA for the I/O pin. This is true also when a weak pull-down is chosen and the input pin is driven HIGH. This current can be avoided by driving the input LOW when a weak pull-down resistor is used and driving it HIGH when a weak pull-up resistor is used.

This current draw can occur in the following cases:



Notes:

1. *All NMOS transistors connected to the I/O pad serve as ESD protection.*
2. *See Table 8-2 on page 215 for available I/O standards.*
3. *Programmable input delay is applicable only to ProASIC3E, IGLOOe, ProASIC3EL, and RT ProASIC3 devices.*

***Figure 8-5* • Simplified I/O Buffer Circuitry**

I/O Registers

Each I/O module contains several input, output, and enable registers. Refer to Figure 8-5 for a simplified representation of the I/O block. The number of input registers is selected by a set of switches (not shown in Figure 8-3 on page 220) between registers to implement single-ended or differential data transmission to and from the FPGA core. The Designer software sets these switches for the user. A common CLR/PRE signal is employed by all I/O registers when I/O register combining is used. Input Register 2 does not have a CLR/PRE pin, as this register is used for DDR implementation. The I/O register combining must satisfy certain rules.

If the assignment is not successful, an error message appears in the Output window.

To undo the I/O bank assignments, choose **Undo** from the **Edit** menu. Undo removes the I/O technologies assigned by the IOBA. It does not remove the I/O technologies previously assigned.

To redo the changes undone by the Undo command, choose **Redo** from the **Edit** menu.

To clear I/O bank assignments made before using the Undo command, manually unassign or reassign I/O technologies to banks. To do so, choose **I/O Bank Settings** from the **Edit** menu to display the I/O Bank Settings dialog box.

Conclusion

Fusion, IGLOO, and ProASIC3 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 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

User's Guides

Libero SoC User's Guide

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

IGLOO, ProASIC3, SmartFusion, and Fusion 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

Volume Programming Services

Device Type Supported: Flash and Antifuse

Once the design is stable for applications with large production volumes, preprogrammed devices can be purchased. Table 11-2 describes the volume programming services.

Table 11-2 **• Volume Programming Services**

Programmer	Vendor	Availability
In-House Programming	Microsemi	Contact Microsemi Sales
Distributor Programming Centers	Memec Unique	Contact Distribution
Independent Programming Centers	Various	Contact Vendor

Advantages: As programming is outsourced, this solution is easier to implement than creating a substantial in-house programming capability. As programming houses specialize in large-volume programming, this is often the most cost-effective solution.

Limitations: There are some logistical issues with the use of a programming service provider, such as the transfer of programming files and the approval of First Articles. By definition, the programming file must be released to a third-party programming house. Nondisclosure agreements (NDAs) can be signed to help ensure data protection; however, for extremely security-conscious designs, this may not be an option.

- Microsemi In-House Programming

  When purchasing Microsemi devices in volume, IHP can be requested as part of the purchase. If this option is chosen, there is a small cost adder for each device programmed. Each device is marked with a special mark to distinguish it from blank parts. Programming files for the design will be sent to Microsemi. Sample parts with the design programmed, First Articles, will be returned for customer approval. Once approval of First Articles has been received, Microsemi will proceed with programming the remainder of the order. To request Microsemi IHP, contact your local Microsemi representative.

- Distributor Programming Centers

  If purchases are made through a distributor, many distributors will provide programming for their customers. Consult with your preferred distributor about this option.

Signal Integrity While Using ISP

For ISP of flash devices, customers are expected to follow the board-level guidelines provided on the Microsemi SoC Products Group website. These guidelines are discussed in the datasheets and application notes (refer to the “Related Documents” section of the datasheet for application note links). Customers are also expected to troubleshoot board-level signal integrity issues by measuring voltages and taking oscilloscope plots.

Programming Failure Allowances

Microsemi has strict policies regarding programming failure allowances. Please refer to *Programming and Functional Failure Guidelines* on the Microsemi SoC Products Group website for details.

Contacting the Customer Support Group

Highly skilled engineers staff the Customer Applications Center from 7:00 A.M. to 6:00 P.M., Pacific time, Monday through Friday. You can contact the center by one of the following methods:

Electronic Mail

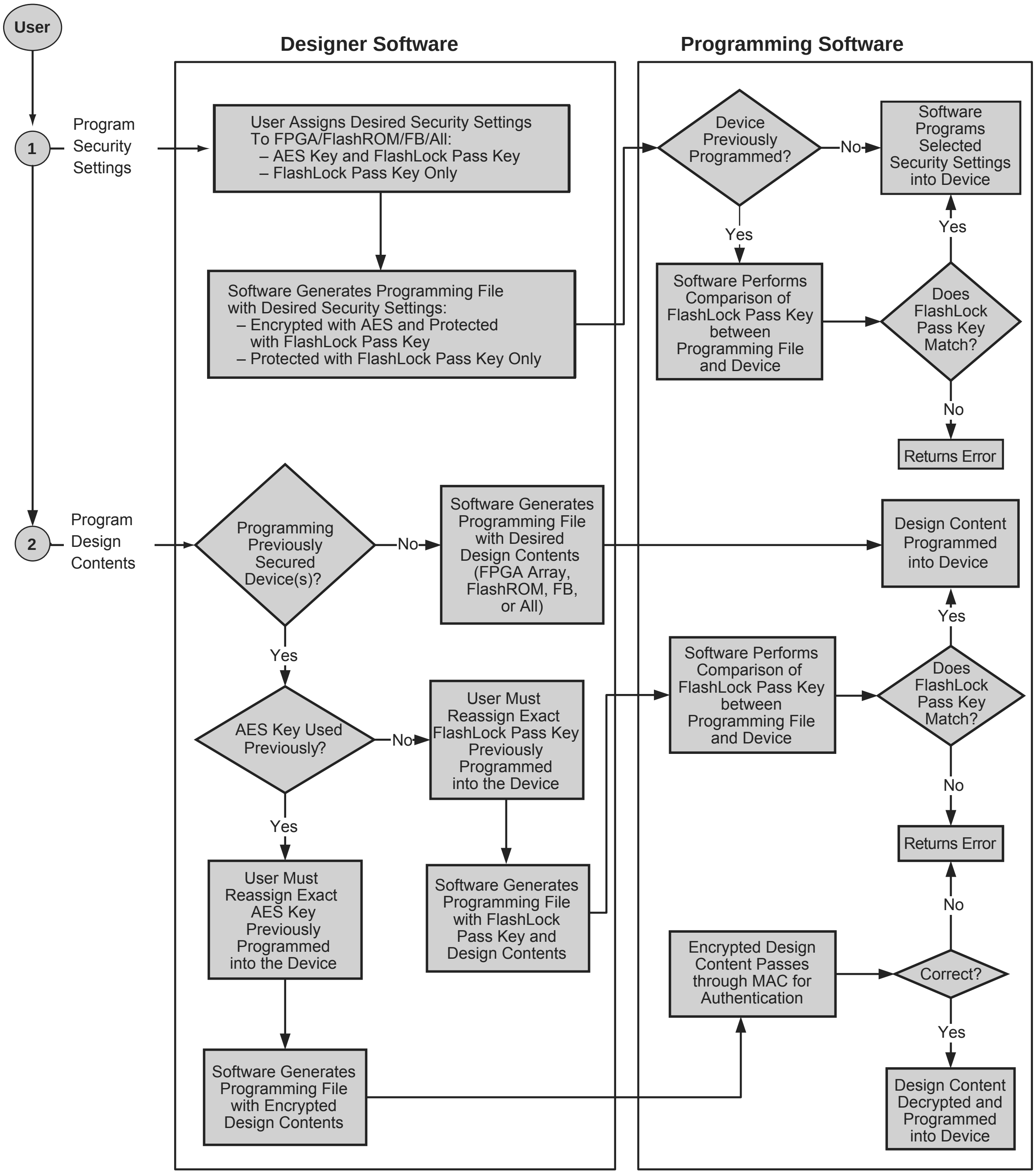
You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. Microsemi monitors the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and contact information for efficient processing of your request. The technical support email address is soc_tech@microsemi.com.

Telephone

Our Technical Support Hotline answers all calls. The center retrieves information, such as your name, company name, telephone number, and question. Once this is done, a case number is assigned. Then the center forwards the information to a queue where the first available applications engineer receives the data and returns your call. The phone hours are from 7:00 A.M. to 6:00 P.M., Pacific time, Monday through Friday.

The Customer Applications Center number is (800) 262-1060.

European customers can call +44 (0) 1256 305 600.



Note: *If programming the Security Header only, just perform sub-flow 1.*
If programming design content only, just perform sub-flow 2.

***Figure 12-9* • Security Programming Flows**

17 – UJTAG Applications in Microsemi's Low Power Flash Devices

Introduction

In Fusion, IGLOO, and ProASIC3 devices, there is bidirectional access from the JTAG port to the core VersaTiles during normal operation of the device (Figure 17-1). User JTAG (UJTAG) is the ability for the design to use the JTAG ports for access to the device for updates, etc. While regular JTAG is used, the UJTAG tiles, located at the southeast area of the die, are directly connected to the JTAG Test Access Port (TAP) Controller in normal operating mode. As a result, all the functional blocks of the device, such as Clock Conditioning Circuits (CCCs) with PLLs, SRAM blocks, embedded FlashROM, flash memory blocks, and I/O tiles, can be reached via the JTAG ports. The UJTAG functionality is available by instantiating the UJTAG macro directly in the source code of a design. Access to the FPGA core VersaTiles from the JTAG ports enables users to implement different applications using the TAP Controller (JTAG port). This document introduces the UJTAG tile functionality and discusses a few application examples. However, the possible applications are not limited to what is presented in this document. UJTAG can serve different purposes in many designs as an elementary or auxiliary part of the design. For detailed usage information, refer to the "Boundary Scan in Low Power Flash Devices" section on page 357.

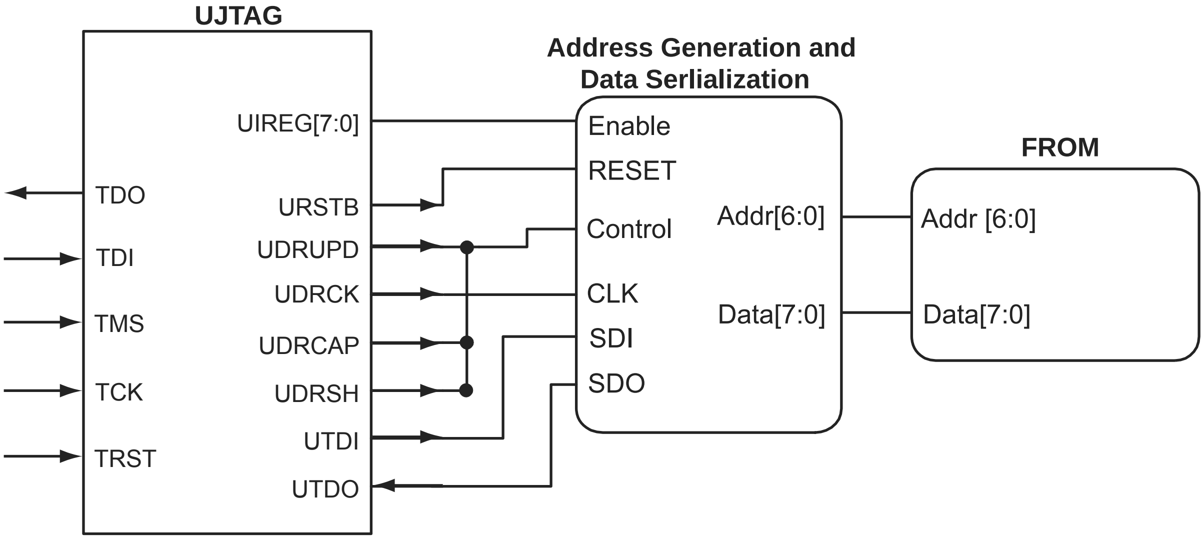


Figure 17-1 • **Block Diagram of Using UJTAG to Read FlashROM Contents**

Typical UJTAG Applications

Bidirectional access to the JTAG port from VersaTiles—without putting the device into test mode—creates flexibility to implement many different applications. This section describes a few of these. All are based on importing/exporting data through the UJTAG tiles.

Clock Conditioning Circuitry—Dynamic Reconfiguration

In low power flash devices, CCCs, which include PLLs, can be configured dynamically through either an 81-bit embedded shift register or static flash programming switches. These 81 bits control all the characteristics of the CCC: routing MUX architectures, delay values, divider values, etc. Table 17-3 lists the 81 configuration bits in the CCC.

Table 17-3 **• Configuration Bits of Fusion, IGLOO, and ProASIC3 CCC Blocks**

Bit Number(s)	Control Function
80	RESET ENABLE
79	DYNCSEL
78	DYNBSEL
77	DYNASEL
<76:74>	VCOSEL [2:0]
73	STATCSEL
72	STATBSEL
71	STATASEL
<70:66>	DLYC [4:0]
<65:61>	DLYB {4:0]
<60:56>	DLYGLC [4:0]
<55:51>	DLYGLB [4:0]
<50:46>	DLYGLA [4:0]
45	XDLYSEL
<44:40>	FBDLY [4:0]
<39:38>	FBSEL
<37:35>	OCMUX [2:0]
<34:32>	OBMUX [2:0]
<31:29>	OAMUX [2:0]
<28:24>	OCDIV [4:0]
<23:19>	OBDIV [4:0]
<18:14>	OADIV [4:0]
<13:7>	FBDIV [6:0]
<6:0>	FINDIV [6:0]

The embedded 81-bit shift register (for the dynamic configuration of the CCC) is accessible to the VersaTiles, which, in turn, have access to the UJTAG tiles. Therefore, the CCC configuration shift register can receive and load the new configuration data stream from JTAG.

Dynamic reconfiguration eliminates the need to reprogram the device when reconfiguration of the CCC functional blocks is needed. The CCC configuration can be modified while the device continues to operate. Employing the UJTAG core requires the user to design a module to provide the configuration data and control the CCC configuration shift register. In essence, this is a user-designed TAP Controller requiring chip resources.

Similar reconfiguration capability exists in the ProASIC$^{\underline{\text{PLUS}}}$® family. The only difference is the number of shift register bits controlling the CCC (27 in ProASIC$^{\underline{\text{PLUS}}}$ and 81 in IGLOO, ProASIC3, and Fusion).

Related Documents

Datasheets

ProASIC3 Flash Family FPGAs

http://www.microsemi.com/soc/documents/PA3_DS.pdf

ProASIC3E Flash Family FPGAs

http://www.microsemi.com/soc/documents/PA3E_DS.pdf

List of Changes

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

Date	Changes	Page
v1.2 (December 2008)	IGLOO nano and ProASIC3 nano devices were added to the document as supported device types.	
v1.1 (October 2008)	The "Introduction" section was updated to add Military ProASIC3EL and RT ProASIC3 devices to the list of devices that can have inputs driven in while the device is not powered.	373
	The "Flash Devices Support Power-Up Behavior" section was revised to include new families and make the information more concise.	374
	The "Cold-Sparing" section was revised to add Military ProASIC3/EL and RT ProASIC3 devices to the lists of devices with and without cold-sparing support.	382
	The "Hot-Swapping" section was revised to add Military ProASIC3/EL and RT ProASIC3 devices to the lists of devices with and without hot-swap support. AGL400 was added to the list of devices that do not support hot-swapping.	383
v1.0 (August 2008)	This document was revised, renamed, and assigned a new part number. It now includes data for the IGLOO and ProASIC3L families.	N/A
v1.3 (March 2008)	The "List of Changes" section was updated to include the three different I/O Structure handbook chapters.	384
v1.2 (February 2008)	The first sentence of the "PLL Behavior at Brownout Condition" section was updated to read, "When PLL power supply voltage and/or V_{CC} levels drop below the VCC brownout levels (0.75 V ± 0.25 V), the PLL output lock signal goes low and/or the output clock is lost."	381
v1.1 (January 2008)	The "PLL Behavior at Brownout Condition" section was added.	381