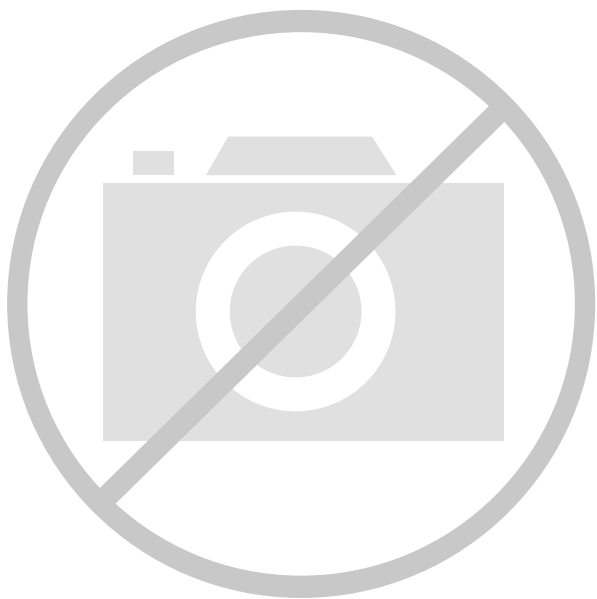




Welcome to [E-XFL.COM](https://www.e-xfl.com)

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

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

Applications of Embedded - FPGAs

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

Details

Product Status	Active
Number of LABs/CLBs	-
Number of Logic Elements/Cells	-
Total RAM Bits	110592
Number of I/O	177
Number of Gates	600000
Voltage - Supply	1.14V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 100°C (TJ)
Package / Case	144-LBGA
Supplier Device Package	144-FPBGA (13x13)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/m1a3p600l-fg144i

Sleep and Shutdown Modes

Sleep Mode

IGLOO, IGLOO nano, IGLOO PLUS, ProASIC3L, and RT ProASIC3 FPGAs support Sleep mode when device functionality is not required. In Sleep mode, V_{CC} (core voltage), V_{JTAG} (JTAG DC voltage), and VPUMP (programming voltage) are grounded, resulting in the FPGA core being turned off to reduce power consumption. While the device is in Sleep mode, the rest of the system can still be operating and driving the input buffers of the device. The driven inputs do not pull up the internal power planes, and the current draw is limited to minimal leakage current.

Table 2-7 shows the power supply status in Sleep mode.

Table 2-7 • Sleep Mode—Power Supply Requirement for IGLOO, IGLOO nano, IGLOO PLUS, ProASIC3L, and RT ProASIC3 Devices

Power Supplies	Power Supply State
VCC	Powered off
VCCI = VMV	Powered on
VJTAG	Powered off
VPUMP	Powered off

Refer to the "Power-Up/-Down Behavior" section on page 33 for more information about I/O states during Sleep mode and the timing diagram for entering and exiting Sleep mode.

Shutdown Mode

Shutdown mode is supported for all IGLOO nano and IGLOO PLUS devices as well the following IGLOO/e devices: AGL015, AGL030, AGL0600, AGL03000, and A3PE3000L. Shutdown mode can be used by turning off all power supplies when the device function is not needed. Cold-sparing and hot-insertion features enable these devices to be powered down without turning off the entire system. When power returns, the live-at-power-up feature enables operation of the device after reaching the voltage activation point.

External I/O or Local signal as Clock Source

External I/O refers to regular I/O pins are labeled with the I/O convention IOuxwByVz. You can allow the external I/O or internal signal to access the global. To allow the external I/O or internal signal to access the global network, you need to instantiate the CLKINT macro. Refer to Figure 3-4 on page 51 for an example illustration of the connections. Instead of using CLKINT, you can also use PDC to promote signals from external I/O or internal signal to the global network. However, it may cause layout issues because of synthesis logic replication. Refer to the "Global Promotion and Demotion Using PDC" section on page 67 for details.

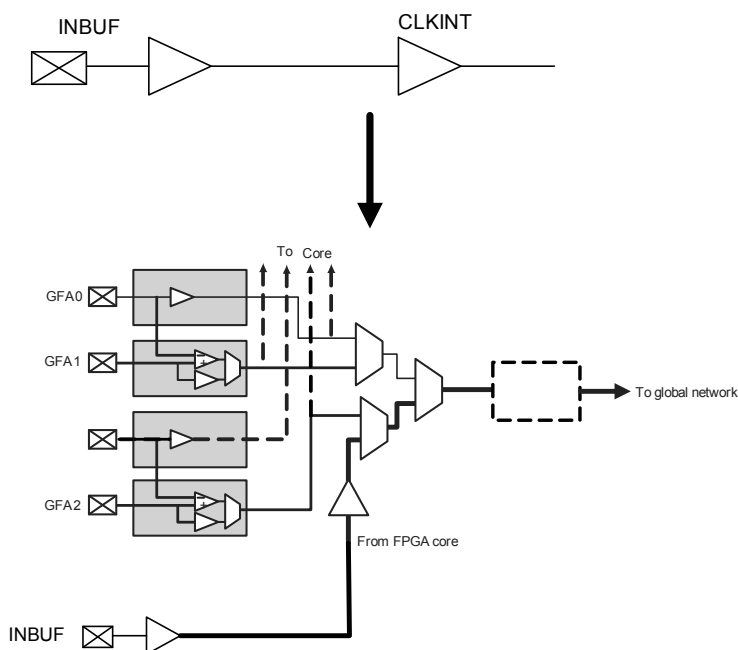


Figure 3-14 • CLKINT Macro

Using Global Macros in Synplicity

The Synplify® synthesis tool automatically inserts global buffers for nets with high fanout during synthesis. By default, Synplicity® puts six global macros (CLKBUF or CLKINT) in the netlist, including any global instantiation or PLL macro. Synplify always honors your global macro instantiation. If you have a PLL (only primary output is used) in the design, Synplify adds five more global buffers in the netlist. Synplify uses the following global counting rule to add global macros in the netlist:

1. CLKBUF: 1 global buffer
2. CLKINT: 1 global buffer
3. CLKDLY: 1 global buffer
4. PLL: 1 to 3 global buffers
 - GLA, GLB, GLC, YB, and YC are counted as 1 buffer.
 - GLB or YB is used or both are counted as 1 buffer.
 - GLC or YC is used or both are counted as 1 buffer.

4 – Clock Conditioning Circuits in Low Power Flash Devices and Mixed Signal FPGAs

Introduction

This document outlines the following device information: Clock Conditioning Circuit (CCC) features, PLL core specifications, functional descriptions, software configuration information, detailed usage information, recommended board-level considerations, and other considerations concerning clock conditioning circuits and global networks in low power flash devices or mixed signal FPGAs.

Overview of Clock Conditioning Circuitry

In Fusion, IGLOO, and ProASIC3 devices, the CCCs are used to implement frequency division, frequency multiplication, phase shifting, and delay operations. The CCCs are available in six chip locations—each of the four chip corners and the middle of the east and west chip sides. For device-specific variations, refer to the "Device-Specific Layout" section on page 94.

The CCC is composed of the following:

- PLL core
- 3 phase selectors
- 6 programmable delays and 1 fixed delay that advances/delays phase
- 5 programmable frequency dividers that provide frequency multiplication/division (not shown in Figure 4-6 on page 87 because they are automatically configured based on the user's required frequencies)
- 1 dynamic shift register that provides CCC dynamic reconfiguration capability

Figure 4-1 provides a simplified block diagram of the physical implementation of the building blocks in each of the CCCs.

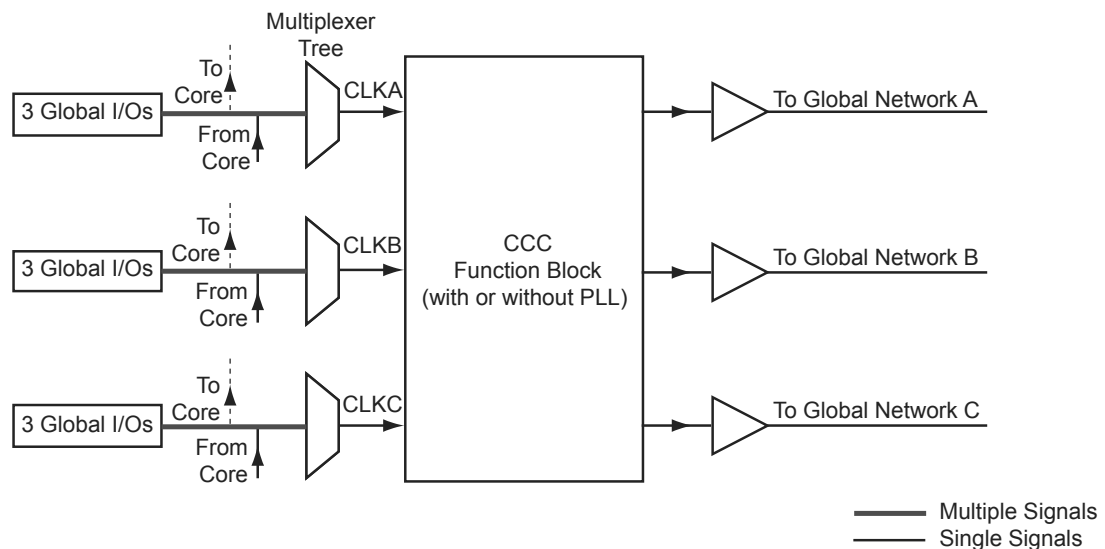


Figure 4-1 • Overview of the CCCs Offered in Fusion, IGLOO, and ProASIC3

Feedback Configuration

The PLL provides both internal and external feedback delays. Depending on the configuration, various combinations of feedback delays can be achieved.

Internal Feedback Configuration

This configuration essentially sets the feedback multiplexer to route the VCO output of the PLL core as the input to the feedback of the PLL. The feedback signal can be processed with the fixed system and the adjustable feedback delay, as shown in Figure 4-24. The dividers are automatically configured by SmartGen based on the user input.

Indicated below is the System Delay pull-down menu. The System Delay can be bypassed by setting it to 0. When set, it adds a 2 ns delay to the feedback path (which results in delay advancement of the output clock by 2 ns).

Figure 4-24 • Internal Feedback with Selectable System Delay

Figure 4-25 shows the controllable Feedback Delay. If set properly in conjunction with the fixed System Delay, the total output delay can be advanced significantly.

Figure 4-25 • Internal Feedback with Selectable Feedback Delay

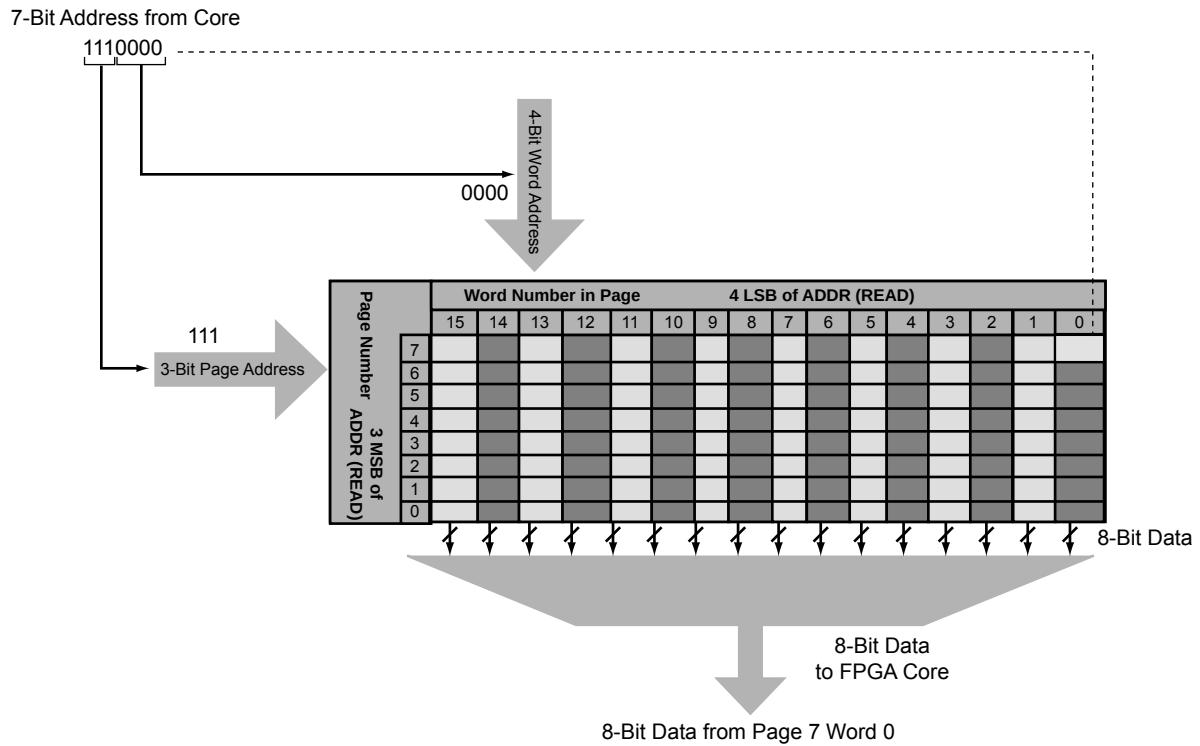


Figure 5-7 • Accessing FlashROM Using FPGA Core

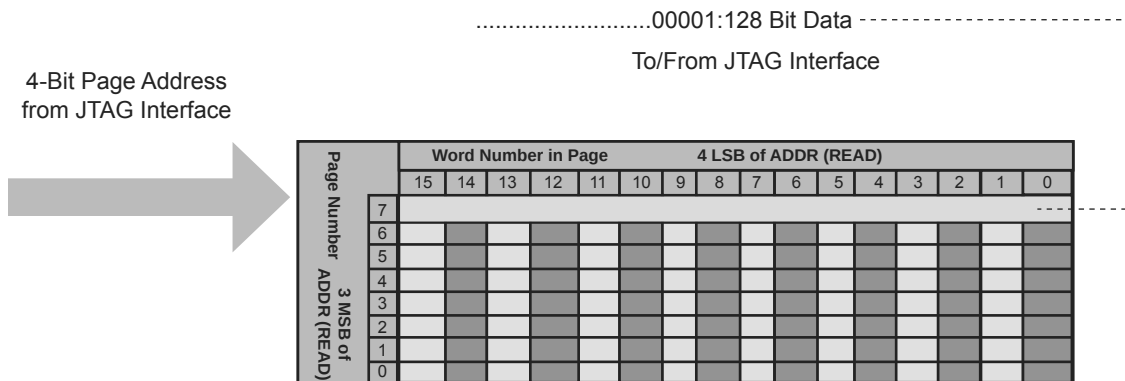


Figure 5-8 • Accessing FlashROM Using JTAG Port

Pipeline Register

```
module D_pipeline (Data, Clock, Q);

input [3:0] Data;
input Clock;
output [3:0] Q;

reg [3:0] Q;

always @ (posedge Clock) Q <= Data;

endmodule
```

4x4 RAM Block (created by SmartGen Core Generator)

```
module mem_block(DI, DO, WADDR, RADDR, WRB, RDB, WCLOCK, RCLOCK);

input [3:0] DI;
output [3:0] DO;
input [1:0] WADDR, RADDR;
input WRB, RDB, WCLOCK, RCLOCK;

wire WEBP, WEAP, VCC, GND;

VCC VCC_1_net(.Y(VCC));
GND GND_1_net(.Y(GND));
INV WEBUBBLEB(.A(WRB), .Y(WEBP));
RAM4K9 RAMBLOCK0(.ADDRA11(GND), .ADDRA10(GND), .ADDRA9(GND), .ADDRA8(GND),
    .ADDRA7(GND), .ADDRA6(GND), .ADDRA5(GND), .ADDRA4(GND), .ADDRA3(GND), .ADDRA2(GND),
    .ADDRA1(RADDR[1]), .ADDRA0(RADDR[0]), .ADDRB11(GND), .ADDRB10(GND), .ADDRB9(GND),
    .ADDRB8(GND), .ADDRB7(GND), .ADDRB6(GND), .ADDRB5(GND), .ADDRB4(GND), .ADDRB3(GND),
    .ADDRB2(GND), .ADDRB1(WADDR[1]), .ADDRB0(WADDR[0]), .DINA8(GND), .DINA7(GND),
    .DINA6(GND), .DINA5(GND), .DINA4(GND), .DINA3(GND), .DINA2(GND), .DINA1(GND),
    .DINA0(GND), .DINB8(GND), .DINB7(GND), .DINB6(GND), .DINB5(GND), .DINB4(GND),
    .DINB3(DI[3]), .DINB2(DI[2]), .DINB1(DI[1]), .DINB0(DI[0]), .WIDTHA0(GND),
    .WIDTHA1(VCC), .WIDTHB0(GND), .WIDTHB1(VCC), .PIPEA(GND), .PIPEB(GND),
    .WMODEA(GND), .WMODEB(GND), .BLKA(WEAP), .BLKB(WEBP), .WENA(VCC), .WENB(GND),
    .CLKA(RCLOCK), .CLKB(WCLOCK), .RESET(VCC), .DOUTA8(), .DOUTA7(), .DOUTA6(),
    .DOUTA5(), .DOUTA4(), .DOUTA3(DO[3]), .DOUTA2(DO[2]), .DOUTA1(DO[1]),
    .DOUTA0(DO[0]), .DOUTB8(), .DOUTB7(), .DOUTB6(), .DOUTB5(), .DOUTB4(), .DOUTB3(),
    .DOUTB2(), .DOUTB1(), .DOUTB0());
INV WEBUBBLEA(.A(RDB), .Y(WEAP));

endmodule
```

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.

Power-Up Behavior

Low power flash devices are power-up/-down friendly; i.e., no particular sequencing is required for power-up and power-down. This eliminates extra board components for power-up sequencing, such as a power-up sequencer.

During power-up, all I/Os are tristated, irrespective of I/O macro type (input buffers, output buffers, I/O buffers with weak pull-ups or weak pull-downs, etc.). Once I/Os become activated, they are set to the user-selected I/O macros. Refer to the "Power-Up/-Down Behavior of Low Power Flash Devices" section on page 373 for details.

Drive Strength

Low power flash devices have up to seven programmable output drive strengths. The user can select the drive strength of a particular output in the I/O Attribute Editor or can instantiate a specialized I/O macro, such as `OUTBUF_S_12` (slew = low, out_drive = 12 mA).

The maximum available drive strength is 24 mA per I/O. Though no I/O should be forced to source or sink more than 24 mA indefinitely, I/Os may handle a higher amount of current (refer to the device IBIS model for maximum source/sink current) during signal transition (AC current). Every device package has its own power dissipation limit; hence, power calculation must be performed accurately to determine how much current can be tolerated per I/O within that limit.

I/O Interfacing

Low power flash devices are 5 V–input– and 5 V–output–tolerant if certain I/O standards are selected (refer to the "5 V Input and Output Tolerance" section on page 194). Along with other low-voltage I/O macros, this 5 V tolerance makes these devices suitable for many types of board component interfacing.

5 V Input and Output Tolerance

IGLOO and ProASIC3 devices are both 5 V-input- and 5 V-output-tolerant if certain I/O standards are selected. Table 8-6 on page 218 shows the I/O standards that support 5 V input tolerance. Only 3.3 V LVTTTL/LVCMOS standards support 5 V output tolerance. Refer to the appropriate family datasheet for detailed description and configuration information.

This feature is not shown in the I/O Attribute Editor.

5 V Input Tolerance

I/Os can support 5 V input tolerance when LVTTTL 3.3 V, LVCMOS 3.3 V, LVCMOS 2.5 V, and LVCMOS 2.5 V / 5.0 V configurations are used (see Table 8-13 on page 231). There are four recommended solutions for achieving 5 V receiver tolerance (see Figure 8-10 on page 233 to Figure 8-13 on page 235 for details of board and macro setups). All the solutions meet a common requirement of limiting the voltage at the input to 3.6 V or less. In fact, the I/O absolute maximum voltage rating is 3.6 V, and any voltage above 3.6 V may cause long-term gate oxide failures.

Solution 1

The board-level design must ensure that the reflected waveform at the pad does not exceed the limits provided in the recommended operating conditions 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 two external resistors as explained below. Relying on the diode clamping would create an excessive pad DC voltage of $3.3\text{ V} + 0.7\text{ V} = 4\text{ V}$.

This solution requires two board resistors, as demonstrated in Figure 8-10 on page 233. Here are some examples of possible resistor values (based on a simplified simulation model with no line effects and $10\ \Omega$ transmitter output resistance, where $R_{tx_out_high} = [V_{CCI} - V_{OH}] / I_{OH}$ and $R_{tx_out_low} = V_{OL} / I_{OL}$).

Example 1 (high speed, high current):

$$R_{tx_out_high} = R_{tx_out_low} = 10\ \Omega$$

$$R1 = 36\ \Omega (\pm 5\%), P(r1)_{min} = 0.069\ \Omega$$

$$R2 = 82\ \Omega (\pm 5\%), P(r2)_{min} = 0.158\ \Omega$$

$$I_{max_tx} = 5.5\text{ V} / (82 \times 0.95 + 36 \times 0.95 + 10) = 45.04\text{ mA}$$

$$t_{RISE} = t_{FALL} = 0.85\text{ ns at } C_{pad_load} = 10\text{ pF (includes up to 25\% safety margin)}$$

$$t_{RISE} = t_{FALL} = 4\text{ ns at } C_{pad_load} = 50\text{ pF (includes up to 25\% safety margin)}$$

Example 2 (low-medium speed, medium current):

$$R_{tx_out_high} = R_{tx_out_low} = 10\ \Omega$$

$$R1 = 220\ \Omega (\pm 5\%), P(r1)_{min} = 0.018\ \Omega$$

$$R2 = 390\ \Omega (\pm 5\%), P(r2)_{min} = 0.032\ \Omega$$

$$I_{max_tx} = 5.5\text{ V} / (220 \times 0.95 + 390 \times 0.95 + 10) = 9.17\text{ mA}$$

$$t_{RISE} = t_{FALL} = 4\text{ ns at } C_{pad_load} = 10\text{ pF (includes up to 25\% safety margin)}$$

$$t_{RISE} = t_{FALL} = 20\text{ ns at } C_{pad_load} = 50\text{ pF (includes up to 25\% safety margin)}$$

Other values of resistors are also allowed as long as the resistors are sized appropriately to limit the voltage at the receiving end to $2.5\text{ V} < V_{in(rx)} < 3.6\text{ V}$ when the transmitter sends a logic 1. This range of $V_{in_dc(rx)}$ must be assured for any combination of transmitter supply ($5\text{ V} \pm 0.5\text{ V}$), transmitter output resistance, and board resistor tolerances.

Temporary overshoots are allowed according to the overshoot and undershoot table in the datasheet.

I/O Software Support

In Libero SoC software, default settings have been defined for the various I/O standards supported. Changes can be made to the default settings via the use of attributes; however, not all I/O attributes are applicable for all I/O standards. Table 8-16 lists the valid I/O attributes that can be manipulated by the user for each I/O standard.

Single-ended I/O standards in low power flash devices support up to five different drive strengths.

Table 8-16 • IGLOOe and ProASIC3E I/O Attributes vs. I/O Standard Applications

I/O Standard	SLEW (output only)	OUT_DRIVE (output only)	SKEW (all macros with OE)	RES_PULL	OUT_LOAD (output only)	COMBINE_REGISTER	IN_DELAY (input only)	IN_DELAY_VAL (input only)	SCHMITT_TRIGGER (input only)	HOT_SWAPPABLE
LVTTL/LVCMOS 3.3 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 2.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 2.5/5.0 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 1.8 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 1.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PCI (3.3 V)			✓		✓	✓	✓	✓		
PCI-X (3.3 V)	✓		✓		✓	✓	✓	✓		
GTL+ (3.3 V)			✓		✓	✓	✓	✓		✓
GTL+ (2.5 V)			✓		✓	✓	✓	✓		✓
GTL (3.3 V)			✓		✓	✓	✓	✓		✓
GTL (2.5 V)			✓		✓	✓	✓	✓		✓
HSTL Class I			✓		✓	✓	✓	✓		✓
HSTL Class II			✓		✓	✓	✓	✓		✓
SSTL2 Class I and II			✓		✓	✓	✓	✓		✓
SSTL3 Class I and II			✓		✓	✓	✓	✓		✓
LVDS, B-LVDS, M-LVDS			✓			✓	✓	✓		✓
LVPECL						✓	✓	✓		✓

Table 8-17 on page 243 lists the default values for the above selectable I/O attributes as well as those that are preset for each I/O standard.

Output Buffers

There are two variations: Regular and Special.

If the **Regular** variation is selected, only the Width (1 to 128) needs to be entered. The default value for Width is 1.

The **Special** variation has Width, Technology, Output Drive, and Slew Rate options.

Bidirectional Buffers

There are two variations: Regular and Special.

The **Regular** variation has Enable Polarity (Active High, Active Low) in addition to the Width option.

The **Special** variation has Width, Technology, Output Drive, Slew Rate, and Resistor Pull-Up/-Down options.

Tristate Buffers

Same as Bidirectional Buffers.

DDR

There are eight variations: DDR with Regular Input Buffers, Special Input Buffers, Regular Output Buffers, Special Output Buffers, Regular Tristate Buffers, Special Tristate Buffers, Regular Bidirectional Buffers, and Special Bidirectional Buffers.

These variations resemble the options of the previous I/O macro. For example, the Special Input Buffers variation has Width, Technology, Voltage Level, and Resistor Pull-Up/-Down options. DDR is not available on IGLOO PLUS devices.

4. Once the desired configuration is selected, click the **Generate** button. The Generate Core window opens (Figure 9-4).
 5. Enter a name for the macro. Click **OK**. The core will be generated and saved to the appropriate location within the project files (Figure 9-5 on page 257).
-

Figure 9-4 • Generate Core Window

6. Instantiate the I/O macro in the top-level code.

The user must instantiate the DDR_REG or DDR_OUT macro in the design. Use SmartGen to generate both these macros and then instantiate them in your top level. To combine the DDR macros with the I/O, the following rules must be met:

10 – DDR for Microsemi’s Low Power Flash Devices

Introduction

The I/Os in Fusion, IGLOO, and ProASIC3 devices support Double Data Rate (DDR) mode. In this mode, new data is present on every transition (or clock edge) of the clock signal. This mode doubles the data transfer rate compared with Single Data Rate (SDR) mode, where new data is present on one transition (or clock edge) of the clock signal. Low power flash devices have DDR circuitry built into the I/O tiles. I/Os are configured to be DDR receivers or transmitters by instantiating the appropriate special macros (examples shown in Figure 10-4 on page 276 and Figure 10-5 on page 277) and buffers (DDR_OUT or DDR_REG) in the RTL design. This document discusses the options the user can choose to configure the I/Os in this mode and how to instantiate them in the design.

Double Data Rate (DDR) Architecture

Low power flash devices support 350 MHz DDR inputs and outputs. In DDR mode, new data is present on every transition of the clock signal. Clock and data lines have identical bandwidths and signal integrity requirements, making them very efficient for implementing very high-speed systems. High-speed DDR interfaces can be implemented using LVDS (not applicable for IGLOO nano and ProASIC3 nano devices). In IGLOOe, ProASIC3E, AFS600, and AFS1500 devices, DDR interfaces can also be implemented using the HSTL, SSTL, and LVPECL I/O standards. The DDR feature is primarily implemented in the FPGA core periphery and is not tied to a specific I/O technology or limited to any I/O standard.

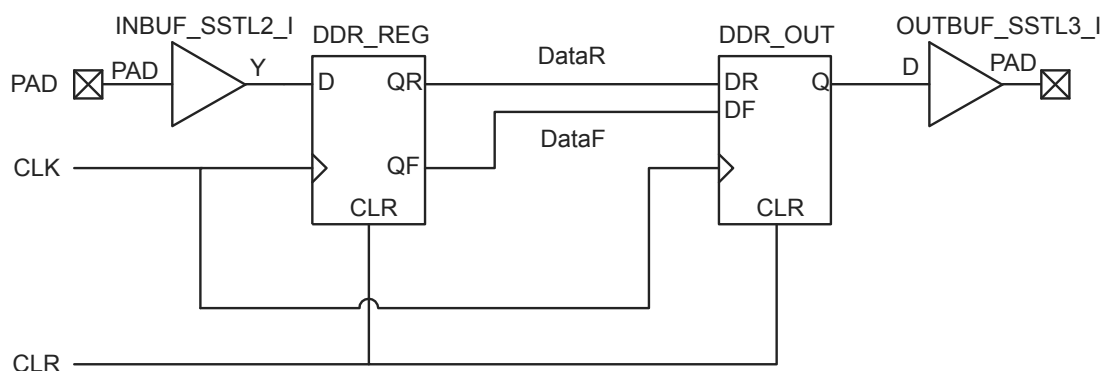


Figure 10-1 • DDR Support in Low Power Flash Devices

11 – Programming Flash Devices

Introduction

This document provides an overview of the various programming options available for the Microsemi flash families. The electronic version of this document includes active links to all programming resources, which are available at <http://www.microsemi.com/soc/products/hardware/default.aspx>. For Microsemi antifuse devices, refer to the *Programming Antifuse Devices* document.

Summary of Programming Support

FlashPro4 and FlashPro3 are high-performance in-system programming (ISP) tools targeted at the latest generation of low power flash devices offered by the SmartFusion,[®] Fusion,[®] IGLOO,[®] and ProASIC[®]3 families, including ARM-enabled devices. FlashPro4 and FlashPro3 offer extremely high performance through the use of USB 2.0, are high-speed compliant for full use of the 480 Mbps bandwidth, and can program ProASIC3 devices in under 30 seconds. Powered exclusively via USB, FlashPro4 and FlashPro3 provide a VPUMP voltage of 3.3 V for programming these devices.

FlashPro4 replaced FlashPro3 in 2010. FlashPro4 supports SmartFusion, Fusion, ProASIC3, and IGLOO devices as well as future generation flash devices. FlashPro4 also adds 1.2 V programming for IGLOO nano V2 devices. FlashPro4 is compatible with FlashPro3; however it adds a programming mode (PROG_MODE) signal to the previously unused pin 4 of the JTAG connector. The PROG_MODE goes high during programming and can be used to turn on a 1.5 V external supply for those devices that require 1.5 V for programming. If both FlashPro3 and FlashPro4 programmers are used for programming the same boards, pin 4 of the JTAG connector must not be connected to anything on the board because FlashPro4 uses pin 4 for PROG_MODE.

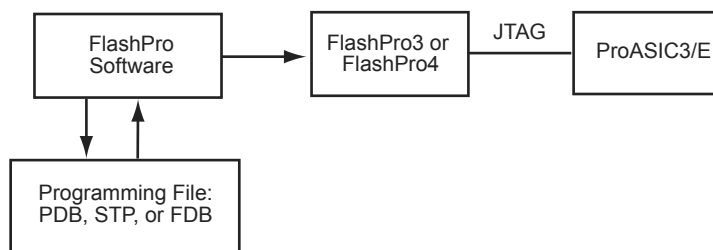
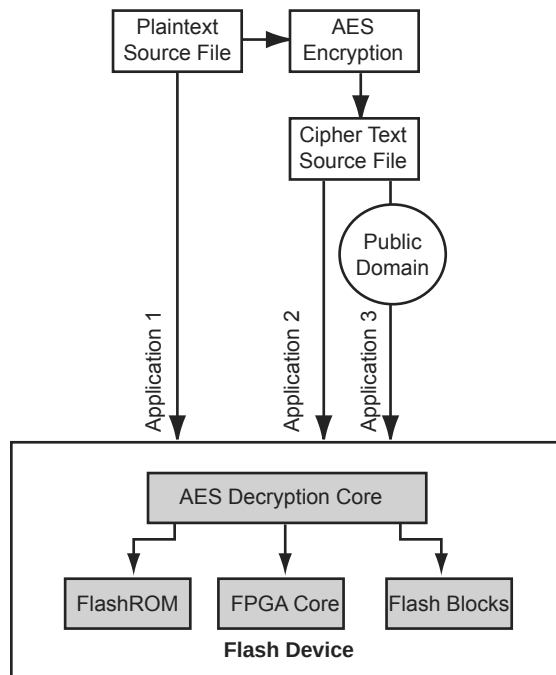


Figure 11-1 • FlashPro Programming Setup

Security in Action

This section illustrates some applications of the security advantages of Microsemi's devices (Figure 12-6).



Note: Flash blocks are only used in Fusion devices

Figure 12-6 • Security Options

FlashROM Security Use Models

Each of the subsequent sections describes in detail the available selections in Microsemi Designer as an aid to understanding security applications and generating appropriate programming files for those applications. Before proceeding, it is helpful to review Figure 12-7 on page 309, which gives a general overview of the programming file generation flow within the Designer software as well as what occurs during the device programming stage. Specific settings are discussed in the following sections.

In Figure 12-7 on page 309, the flow consists of two sub-flows. Sub-flow 1 describes programming security settings to the device only, and sub-flow 2 describes programming the design contents only.

In Application 1, described in the "Application 1: Trusted Environment" section on page 309, the user does not need to generate separate files but can generate one programming file containing both security settings and design contents. Then programming of the security settings and design contents is done in one step. Both sub-flow 1 and sub-flow 2 are used.

In Application 2, described in the "Application 2: Nontrusted Environment—Unsecured Location" section on page 309, the trusted site should follow sub-flows 1 and 2 separately to generate two separate programming files. The programming file from sub-flow 1 will be used at the trusted site to program the device(s) first. The programming file from sub-flow 2 will be sent off-site for production programming.

In Application 3, described in the "Application 3: Nontrusted Environment—Field Updates/Upgrades" section on page 310, typically only sub-flow 2 will be used, because only updates to the design content portion are needed and no security settings need to be changed.

In the event that update of the security settings is necessary, see the "Reprogramming Devices" section on page 321 for details. For more information on programming low power flash devices, refer to the "In-System Programming (ISP) of Microsemi's Low Power Flash Devices Using FlashPro4/3/3X" section on page 327.

Figure 12-10 • All Silicon Features Selected for IGLOO and ProASIC3 Devices

Figure 12-11 • All Silicon Features Selected for Fusion

STAPL vs. DirectC

Programming the low power flash devices is performed using DirectC or the STAPL player. Both tools use the STAPL file as an input. DirectC is a compiled language, whereas STAPL is an interpreted language. Microprocessors will be able to load the FPGA using DirectC much more quickly than STAPL. This speed advantage becomes more apparent when lower clock speeds of 8- or 16-bit microprocessors are used. DirectC also requires less memory than STAPL, since the programming algorithm is directly implemented. STAPL does have one advantage over DirectC—the ability to upgrade. When a new programming algorithm is required, the STAPL user simply needs to regenerate a STAPL file using the latest version of the Designer software and download it to the system. The DirectC user must download the latest version of DirectC from Microsemi, compile everything, and download the result into the system (Figure 15-4).

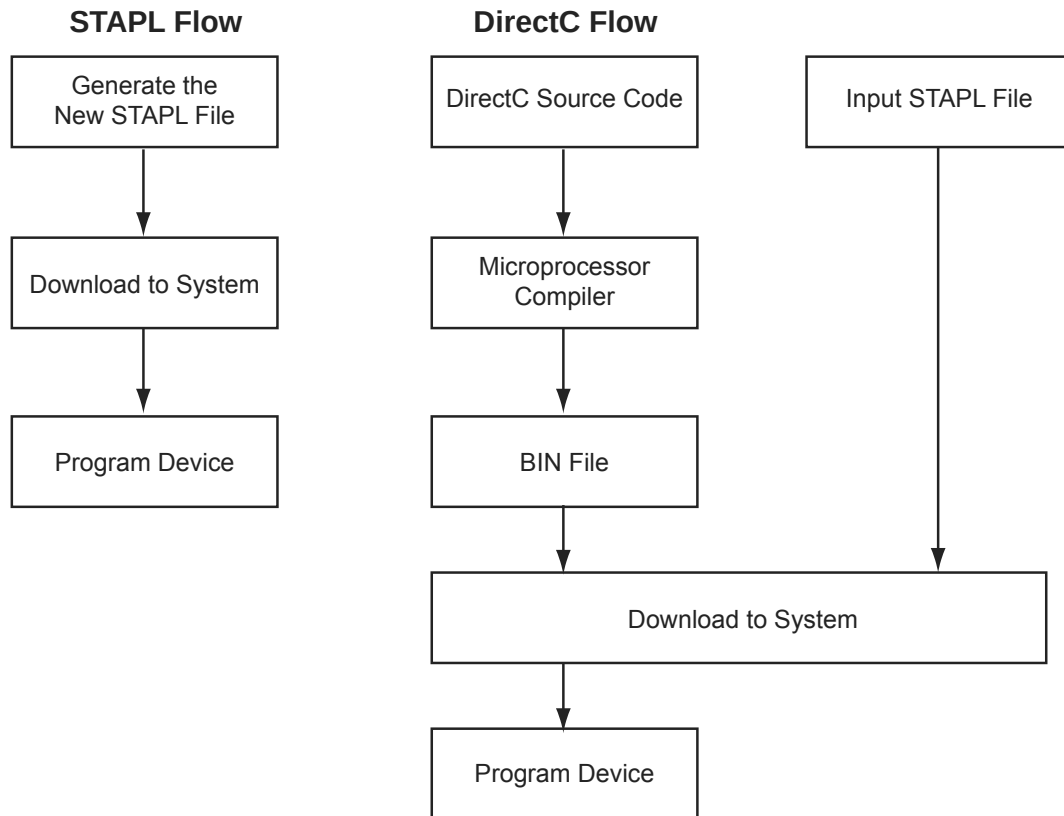


Figure 15-4 • STAPL vs. DirectC

Remote Upgrade via TCP/IP

Transmission Control Protocol (TCP) provides a reliable bitstream transfer service between two endpoints on a network. TCP depends on Internet Protocol (IP) to move packets around the network on its behalf. TCP protects against data loss, data corruption, packet reordering, and data duplication by adding checksums and sequence numbers to transmitted data and, on the receiving side, sending back packets and acknowledging the receipt of data.

The system containing the low power flash device can be assigned an IP address when deployed in the field. When the device requires an update (core or FlashROM), the programming instructions along with the new programming data (AES-encrypted cipher text) can be sent over the Internet to the target system via the TCP/IP protocol. Once the MCU receives the instruction and data, it can proceed with the FPGA update. Low power flash devices support Message Authentication Code (MAC), which can be used to validate data for the target device. More details are given in the "Message Authentication Code (MAC) Validation/Authentication" section.

Hardware Requirement

To facilitate the programming of the low power flash families, the system must have a microprocessor (with access to the device JTAG pins) to process the programming algorithm, memory to store the programming algorithm, programming data, and the necessary programming voltage. Refer to the relevant datasheet for programming voltages.

Security

Encrypted Programming

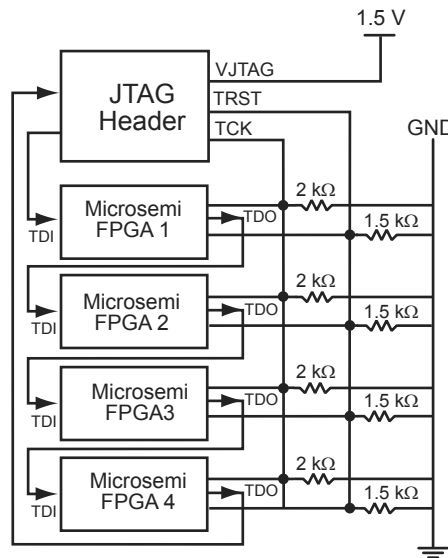
As an additional security measure, the devices are equipped with AES decryption. AES works in two steps. The first step is to program a key into the devices in a secure or trusted programming center (such as Microsemi SoC Products Group In-House Programming (IHP) center). The second step is to encrypt any programming files with the same encryption key. The encrypted programming file will only work with the devices that have the same key. The AES used in the low power flash families is the 128-bit AES decryption engine (Rijndael algorithm).

Message Authentication Code (MAC) Validation/Authentication

As part of the AES decryption flow, the devices are equipped with a MAC validation/authentication system. MAC is an authentication tag, also called a checksum, derived by applying an on-chip authentication scheme to a STAPL file as it is loaded into the FPGA. MACs are computed and verified with the same key so they can only be verified by the intended recipient. When the MCU system receives the AES-encrypted programming data (cipher text), it can validate the data by loading it into the FPGA and performing a MAC verification prior to loading the data, via a second programming pass, into the FPGA core cells. This prevents erroneous or corrupt data from getting into the FPGA.

Low power flash devices with AES and MAC are superior to devices with only DES or 3DES encryption. Because the MAC verifies the correctness of the data, the FPGA is protected from erroneous loading of invalid programming data that could damage a device (Figure 15-5 on page 355).

The AES with MAC enables field updates over public networks without fear of having the design stolen. An encrypted programming file can only work on devices with the correct key, rendering any stolen files



Note: TCK is correctly wired with an equivalent tie-off resistance of 500 Ω , which satisfies the table for VJTAG of 1.5 V. The resistor values for TRST are not appropriate in this case, as the tie-off resistance of 375 Ω is below the recommended minimum for VJTAG = 1.5 V, but would be appropriate for a VJTAG setting of 2.5 V or 3.3 V.

Figure 16-3 • Parallel Resistance on JTAG Chain of Devices

Advanced Boundary Scan Register Settings

You will not be able to control the order in which I/Os are released from boundary scan control. Testing has produced cases where, depending on I/O placement and FPGA routing, a 5 ns glitch has been seen on exiting programming mode. The following setting is recommended to prevent such I/O glitches:

1. In the FlashPro software, configure the advanced BSR settings for **Specify I/O Settings During Programming**.
2. Set the input BSR cell to **Low** for the input I/O.

Flash Devices Support Power-Up Behavior

The flash FPGAs listed in Table 18-1 support power-up behavior and the functions described in this document.

Table 18-1 • 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

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 18-1. 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 18-1. 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*.