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	516096
Number of I/O	620
Number of Gates	3000000
Voltage - Supply	1.14V ~ 1.575V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 100°C (TJ)
Package / Case	896-BGA
Supplier Device Package	896-FBGA (31x31)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/m1a3pe3000l-1fg896i

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, AGLE600, AGLE3000, 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.

```
    wire VCC, GND;

    VCC VCC_1_net(.Y(VCC));
    GND GND_1_net(.Y(GND));
    PLL Core(.CLKA(CLKA), .EXTFB(GND), .POWERDOWN(POWERDOWN),
        .GLA(GLA), .LOCK(LOCK), .GLB(), .YB(), .GLC(), .YC(),
        .OADIV0(GND), .OADIV1(GND), .OADIV2(GND), .OADIV3(GND),
        .OADIV4(GND), .OAMUX0(GND), .OAMUX1(GND), .OAMUX2(VCC),
        .DLYGLA0(GND), .DLYGLA1(GND), .DLYGLA2(GND), .DLYGLA3(GND)
        , .DLYGLA4(GND), .OBDIV0(GND), .OBDIV1(GND), .OBDIV2(GND),
        .OBDIV3(GND), .OBDIV4(GND), .OBMUX0(GND), .OBMUX1(GND),
        .OBMUX2(GND), .DLYYB0(GND), .DLYYB1(GND), .DLYYB2(GND),
        .DLYYB3(GND), .DLYYB4(GND), .DLYGLB0(GND), .DLYGLB1(GND),
        .DLYGLB2(GND), .DLYGLB3(GND), .DLYGLB4(GND), .OCDIV0(GND),
        .OCDIV1(GND), .OCDIV2(GND), .OCDIV3(GND), .OCDIV4(GND),
        .OCMUX0(GND), .OCMUX1(GND), .OCMUX2(GND), .DLYYC0(GND),
        .DLYYC1(GND), .DLYYC2(GND), .DLYYC3(GND), .DLYYC4(GND),
        .DLYGLC0(GND), .DLYGLC1(GND), .DLYGLC2(GND), .DLYGLC3(GND)
        , .DLYGLC4(GND), .FINDIV0(VCC), .FINDIV1(GND), .FINDIV2(
        VCC), .FINDIV3(GND), .FINDIV4(GND), .FINDIV5(GND),
        .FINDIV6(GND), .FBDIV0(VCC), .FBDIV1(GND), .FBDIV2(VCC),
        .FBDIV3(GND), .FBDIV4(GND), .FBDIV5(GND), .FBDIV6(GND),
        .FBDLY0(GND), .FBDLY1(GND), .FBDLY2(GND), .FBDLY3(GND),
        .FBDLY4(GND), .FBSEL0(VCC), .FBSEL1(GND), .XDLYSEL(GND),
        .VCOSEL0(GND), .VCOSEL1(GND), .VCOSEL2(GND));
    defparam Core.VCOFREQUENCY = 33.000;
endmodule
```

The "PLL Configuration Bits Description" section on page 106 provides descriptions of the PLL configuration bits for completeness. The configuration bits are shown as busses only for purposes of illustration. They will actually be broken up into individual pins in compilation libraries and all simulation models. For example, the FBSEL[1:0] bus will actually appear as pins FBSEL1 and FBSEL0. The setting of these select lines for the static PLL configuration is performed by the software and is completely transparent to the user.

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

Place-and-Route Stage Considerations

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

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

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

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

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

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

5 – FlashROM in Microsemi's Low Power Flash Devices

Introduction

The Fusion, IGLOO, and ProASIC3 families of low power flash-based devices have a dedicated nonvolatile FlashROM memory of 1,024 bits, which provides a unique feature in the FPGA market. The FlashROM can be read, modified, and written using the JTAG (or UJTAG) interface. It can be read but not modified from the FPGA core. Only low power flash devices contain on-chip user nonvolatile memory (NVM).

Architecture of User Nonvolatile FlashROM

Low power flash devices have 1 kbit of user-accessible nonvolatile flash memory on-chip that can be read from the FPGA core fabric. The FlashROM is arranged in eight banks of 128 bits (16 bytes) during programming. The 128 bits in each bank are addressable as 16 bytes during the read-back of the FlashROM from the FPGA core. Figure 5-1 shows the FlashROM logical structure.

The FlashROM can only be programmed via the IEEE 1532 JTAG port. It cannot be programmed directly from the FPGA core. When programming, each of the eight 128-bit banks can be selectively reprogrammed. The FlashROM can only be reprogrammed on a bank boundary. Programming involves an automatic, on-chip bank erase prior to reprogramming the bank. The FlashROM supports synchronous read. The address is latched on the rising edge of the clock, and the new output data is stable after the falling edge of the same clock cycle. For more information, refer to the timing diagrams in the DC and Switching Characteristics chapter of the appropriate datasheet. The FlashROM can be read on byte boundaries. The upper three bits of the FlashROM address from the FPGA core define the bank being accessed. The lower four bits of the FlashROM address from the FPGA core define which of the 16 bytes in the bank is being accessed.

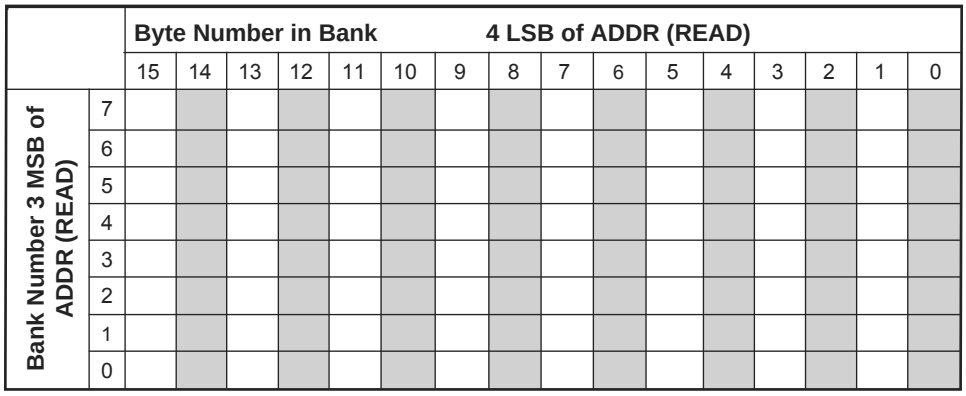


***Figure 5-1* • FlashROM Architecture**

FlashROM Applications

The SmartGen core generator is used to configure FlashROM content. You can configure each page independently. SmartGen enables you to create and modify regions within a page; these regions can be 1 to 16 bytes long (Figure 5-4).

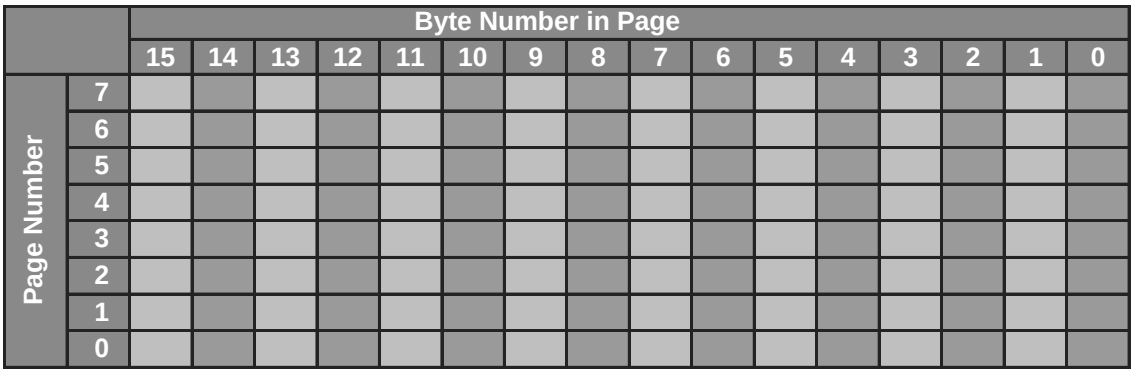


Figure 5-4 • **FlashROM Configuration**

The FlashROM content can be changed independently of the FPGA core content. It can be easily accessed and programmed via JTAG, depending on the security settings of the device. The SmartGen core generator enables each region to be independently updated (described in the "Programming and Accessing FlashROM" section on page 138). This enables you to change the FlashROM content on a per-part basis while keeping some regions "constant" for all parts. These features allow the FlashROM to be used in diverse system applications. Consider the following possible uses of FlashROM:

- Internet protocol (IP) addressing (wireless or fixed)
- System calibration settings
- Restoring configuration after unpredictable system power-down
- Device serialization and/or inventory control
- Subscription-based business models (e.g., set-top boxes)
- Secure key storage
- Asset management tracking
- Date stamping
- Version management

***Table 6-10* • RAM and FIFO Memory Block Consumption**

			Depth									
			256		512	1,024	2,048	4,096	8,192	16,384	32,768	65,536
			Two-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port
Width	1	Number Block	1	1	1	1	1	1	2	4	8	16 × 1
		Configuration	Any	Any	Any	1,024 × 4	2,048 × 2	4,096 × 1	2 × (4,096 × 1) Cascade Deep	4 × (4,096 × 1) Cascade Deep	8 × (4,096 × 1) Cascade Deep	16 × (4,096 × 1) Cascade Deep
	2	Number Block	1	1	1	1	1	2	4	8	16	32
		Configuration	Any	Any	Any	1,024×4	2,048 × 2	2 × (4,096 × 1) Cascaded Wide	4 × (4,096 × 1) Cascaded 2 Deep and 2 Wide	8 × (4,096 × 1) Cascaded 4 Deep and 2 Wide	16 × (4,096 × 1) Cascaded 8 Deep and 2 Wide	32 × (4,096 × 1) Cascaded 16 Deep and 2 Wide
	4	Number Block	1	1	1	1	2	4	8	16	32	64
		Configuration	Any	Any	Any	1,024 × 4	2 × (2,048 × 2) Cascaded Wide	4 × (4,096 × 1) Cascaded Wide	4 × (4,096 × 1) Cascaded 2 Deep and 4 Wide	16 × (4,096 × 1) Cascaded 4 Deep and 4 Wide	32 × (4,096 × 1) Cascaded 8 Deep and 4 Wide	64 × (4,096 × 1) Cascaded 16 Deep and 4 Wide
	8	Number Block	1	1	1	2	4	8	16	32	64	
		Configuration	Any	Any	Any	2 × (1,024 × 4) Cascaded Wide	4 × (2,048 × 2) Cascaded Wide	8 × (4,096 × 1) Cascaded Wide	16 × (4,096 × 1) Cascaded 2 Deep and 8 Wide	32 × (4,096 × 1) Cascaded 4 Deep and 8 Wide	64 × (4,096 × 1) Cascaded 8 Deep and 8 Wide	
	9	Number Block	1	1	1	2	4	8	16	32		
		Configuration	Any	Any	Any	2 × (512 × 9) Cascaded Deep	4 × (512 × 9) Cascaded Deep	8 × (512 × 9) Cascaded Deep	16 × (512 × 9) Cascaded Deep	32 × (512 × 9) Cascaded Deep		
	16	Number Block	1	1	1	4	8	16	32	64		
		Configuration	256 × 18	256 × 18	256 × 18	4 × (1,024 × 4) Cascaded Wide	8 × (2,048 × 2) Cascaded Wide	16 × (4,096 × 1) Cascaded Wide	32 × (4,096 × 1) Cascaded 2 Deep and 16 Wide	32 × (4,096 × 1) Cascaded 4 Deep and 16 Wide		
	18	Number Block	1	2	2	4	8	18	32			
		Configuration	256 × 8	2 × (512 × 9) Cascaded Wide	2 × (512 × 9) Cascaded Wide	4 × (512 × 9) Cascaded 2 Deep and 2 Wide	8 × (512 × 9) Cascaded 4 Deep and 2 Wide	16 × (512 × 9) Cascaded 8 Deep and 2 Wide	16 × (512 × 9) Cascaded 16 Deep and 2 Wide			
	32	Number Block	2	4	4	8	16	32	64			
		Configuration	2 × (256 × 18) Cascaded Wide	4 × (512 × 9) Cascaded Wide	4 × (512 × 9) Cascaded Wide	8 × (1,024 × 4) Cascaded Wide	16 × (2,048 × 2) Cascaded Wide	32 × (4,096 × 1) Cascaded Wide	64 × (4,096 × 1) Cascaded 2 Deep and 32 Wide			
	36	Number Block	2	4	4	8	16	32				
		Configuration	2 × (256 × 18) Cascaded Wide	4 × (512 × 9) Cascaded Wide	4 × (512 × 9) Cascaded Wide	4 × (512 × 9) Cascaded 2 Deep and 4 Wide	16 × (512 × 9) Cascaded 4 Deep and 4 Wide	16 × (512 × 9) Cascaded 8 Deep and 4 Wide				
	64	Number Block	4	8	8	16	32	64				
		Configuration	4 × (256 × 18) Cascaded Wide	8 × (512 × 9) Cascaded Wide	8 × (512 × 9) Cascaded Wide	16 × (1,024 × 4) Cascaded Wide	32 × (2,048 × 2) Cascaded Wide	64 × (4,096 × 1) Cascaded Wide				
	72	Number Block	4	8	8	16	32					
		Configuration	4 × (256 × 18) Cascaded Wide	8 × (512 × 9) Cascaded Wide	8 × (512 × 9) Cascaded Wide	16 × (512 × 9) Cascaded Wide	16 × (512 × 9) Cascaded 4 Deep and 8 Wide					

Note: Memory configurations represented by grayed cells are not supported.

***Table 7-6* • Maximum I/O Frequency for Single-Ended and Differential I/Os in All Banks in IGLOO and ProASIC Devices (maximum drive strength and high slew selected)**

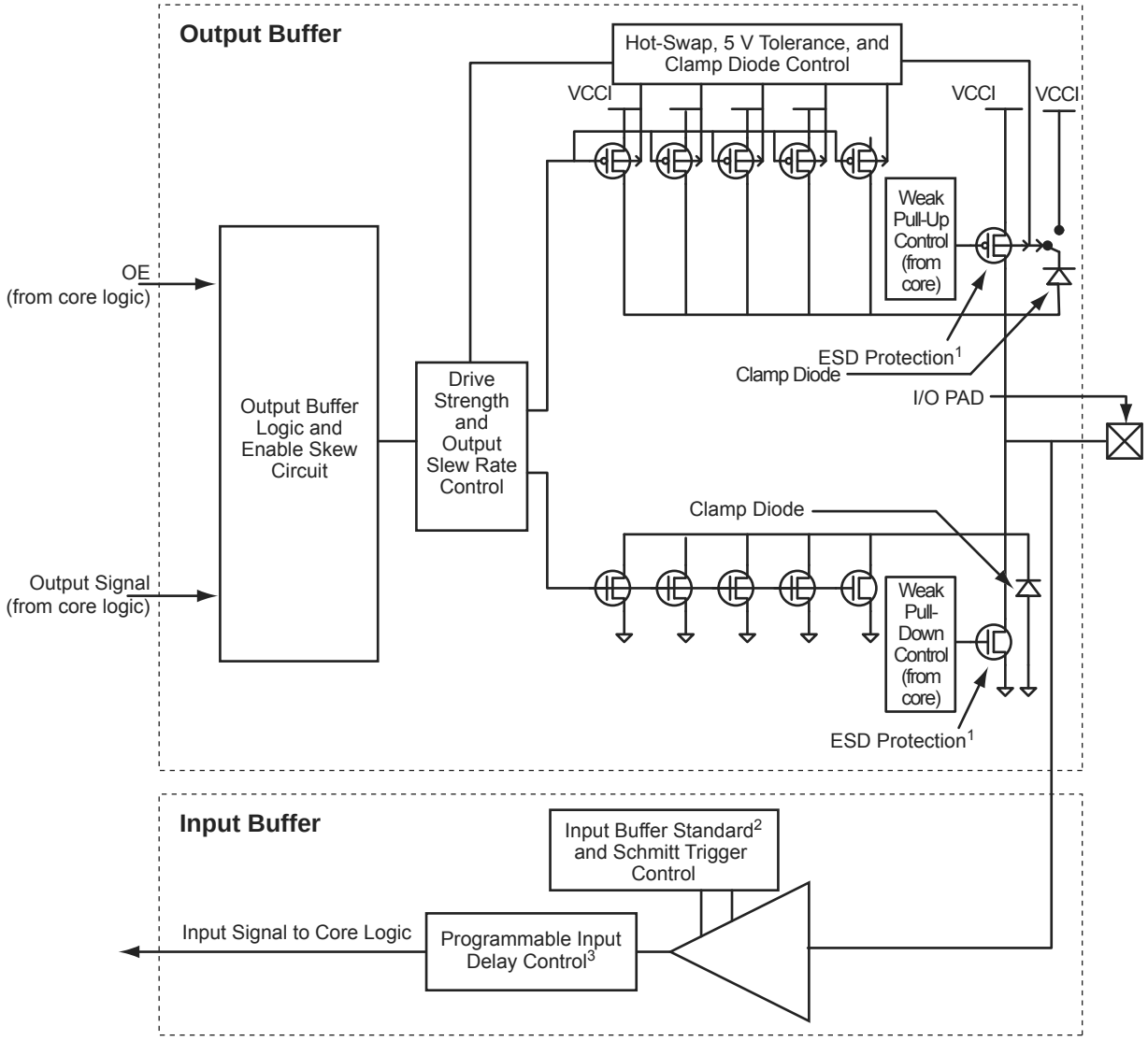
Specification	Maximum Performance		
	ProASIC3	IGLOO V2 or V5 Devices, 1.5 V DC Core Supply Voltage	IGLOO V2, 1.2 V DC Core Supply Voltage
LVTTL/LVCMOS 3.3 V	200 MHz	180 MHz	TBD
LVCMOS 2.5 V	250 MHz	230 MHz	TBD
LVCMOS 1.8 V	200 MHz	180 MHz	TBD
LVCMOS 1.5 V	130 MHz	120 MHz	TBD
PCI	200 MHz	180 MHz	TBD
PCI-X	200 MHz	180 MHz	TBD
LVDS	350 MHz	300 MHz	TBD
LVPECL	350 MHz	300 MHz	TBD

Features Supported on Every I/O

Table 8-6 lists all features supported by transmitter/receiver for single-ended and differential I/Os. Table 8-7 on page 219 lists the performance of each I/O technology.

***Table 8-6* • IGLOOe and ProASIC3E I/O Features**

Feature	Description
All I/O	• High performance (Table 8-7 on page 219) • Electrostatic discharge protection • I/O register combining option
Single-Ended and Voltage-Referenced Transmitter Features	• Hot-swap in every mode except PCI or 5 V–input–tolerant (these modes use clamp diodes and do not allow hot-swap) • Activation of hot-insertion (disabling the clamp diode) is selectable by I/Os. • Output slew rate: 2 slew rates • Weak pull-up and pull-down resistors • Output drive: 5 drive strengths • Programmable output loading • Skew between output buffer enable/disable time: 2 ns delay on rising edge and 0 ns delay on falling edge (see "Selectable Skew between Output Buffer Enable and Disable Times" section on page 236 for more information) • LVTTL/LVCMOS 3.3 V outputs compatible with 5 V TTL inputs
Single-Ended Receiver Features	• 5 V–input–tolerant receiver (Table 8-13 on page 231) • Schmitt trigger option • Programmable delay: 0 ns if bypassed, 0.625 ns with '000' setting, 6.575 ns with '111' setting, 0.85-ns intermediate delay increments (at 25°C, 1.5 V) • Separate ground plane for GNDQ pin and power plane for VMV pin are used for input buffer to reduce output-induced noise.
Voltage-Referenced Differential Receiver Features	• Programmable delay: 0 ns if bypassed, 0.46 ns with '000' setting, 4.66 ns with '111' setting, 0.6-ns intermediate delay increments (at 25°C, 1.5 V) • Separate ground plane for GNDQ pin and power plane for VMV pin are used for input buffer to reduce output-induced noise.
CMOS-Style LVDS, B-LVDS, M-LVDS, or LVPECL Transmitter	• Two I/Os and external resistors are used to provide a CMOS-style LVDS, DDR LVDS, B-LVDS, and M-LVDS/LVPECL transmitter solution. • Activation of hot-insertion (disabling the clamp diode) is selectable by I/Os. • High slew rate • Weak pull-up and pull-down resistors • Programmable output loading
LVDS, DDR LVDS, B-LVDS, and M-LVDS/LVPECL Differential Receiver Features	• Programmable delay: 0 ns if bypassed, 0.46 ns with '000' setting, 4.66 ns with '111' setting, 0.6-ns intermediate delay increments (at 25°C, 1.5 V)



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.

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**

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

DDR Input Register

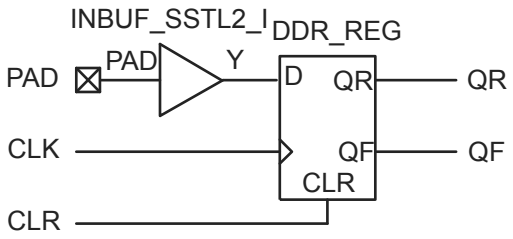


***Figure 10-5* • DDR Input Register (SSTL2 Class I)**

The corresponding structural representations, as generated by SmartGen, are shown below:

Verilog

```
module DDR_InBuf_SSTL2_I(PAD,CLR,CLK,QR,QF);

input   PAD, CLR, CLK;
output  QR, QF;

wire Y;

  INBUF_SSTL2_I INBUF_SSTL2_I_0_inst(.PAD(PAD),.Y(Y));
  DDR_REG DDR_REG_0_inst(.D(Y),.CLK(CLK),.CLR(CLR),.QR(QR),.QF(QF));

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
--The correct library will be inserted automatically by SmartGen
library proasic3; use proasic3.all;
--library fusion; use fusion.all;
--library igloo; use igloo.all;

entity DDR_InBuf_SSTL2_I is
  port(PAD, CLR, CLK : in std_logic;  QR, QF : out std_logic) ;
end DDR_InBuf_SSTL2_I;

architecture DEF_ARCH of  DDR_InBuf_SSTL2_I is

  component INBUF_SSTL2_I
    port(PAD : in std_logic := 'U'; Y : out std_logic) ;
  end component;

  component DDR_REG
    port(D, CLK, CLR : in std_logic := 'U'; QR, QF : out std_logic) ;
  end component;

signal Y : std_logic ;

begin

  INBUF_SSTL2_I_0_inst : INBUF_SSTL2_I
  port map(PAD => PAD, Y => Y);
  DDR_REG_0_inst : DDR_REG
  port map(D => Y, CLK => CLK, CLR => CLR, QR => QR, QF => QF);

end DEF_ARCH;
```

DDR Output Register

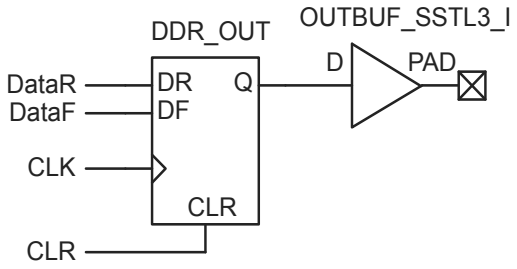


***Figure 10-6* • DDR Output Register (SSTL3 Class I)**

Verilog

```
module DDR_OutBuf_SSTL3_I(DataR,DataF,CLR,CLK,PAD);

input   DataR, DataF, CLR, CLK;
output  PAD;

wire Q, VCC;

  VCC VCC_1_net(.Y(VCC));
  DDR_OUT DDR_OUT_0_inst(.DR(DataR),.DF(DataF),.CLK(CLK),.CLR(CLR),.Q(Q));
  OUTBUF_SSTL3_I OUTBUF_SSTL3_I_0_inst(.D(Q),.PAD(PAD));

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
library proasic3; use proasic3.all;

entity DDR_OutBuf_SSTL3_I is
  port(DataR, DataF, CLR, CLK : in std_logic;  PAD : out std_logic) ;
end DDR_OutBuf_SSTL3_I;

architecture DEF_ARCH of  DDR_OutBuf_SSTL3_I is

  component DDR_OUT
    port(DR, DF, CLK, CLR : in std_logic := 'U'; Q : out std_logic) ;
  end component;

  component OUTBUF_SSTL3_I
    port(D : in std_logic := 'U'; PAD : out std_logic) ;
  end component;

  component VCC
    port( Y : out std_logic);
  end component;

signal Q, VCC_1_net : std_logic ;

begin

  VCC_2_net : VCC port map(Y => VCC_1_net);
  DDR_OUT_0_inst : DDR_OUT
  port map(DR => DataR, DF => DataF, CLK => CLK, CLR => CLR, Q => Q);
  OUTBUF_SSTL3_I_0_inst : OUTBUF_SSTL3_I
  port map(D => Q, PAD => PAD);

end DEF_ARCH;
```

```
module ddr_test(DIN, CLK, CLR, DOUT);

input  DIN, CLK, CLR;
output DOUT;

  Inbuf_ddr Inbuf_ddr (.PAD(DIN), .CLR(clr), .CLK(clk), .QR(qr), .QF(qf));
  Outbuf_ddr Outbuf_ddr (.DataR(qr),.DataF(qf), .CLR(clr), .CLK(clk),.PAD(DOUT));

  INBUF INBUF_CLR (.PAD(CLR), .Y(clr));
  INBUF INBUF_CLK (.PAD(CLK), .Y(clk));

endmodule
```

Simulation Consideration

Microsemi DDR simulation models use inertial delay modeling by default (versus transport delay modeling). As such, pulses that are shorter than the actual gate delays should be avoided, as they will not be seen by the simulator and may be an issue in post-routed simulations. The user must be aware of the default delay modeling and must set the correct delay model in the simulator as needed.

Conclusion

Fusion, IGLOO, and ProASIC3 devices support a wide range of DDR applications with different I/O standards and include built-in DDR macros. The powerful capabilities provided by SmartGen and its GUI can simplify the process of including DDR macros in designs and minimize design errors. Additional considerations should be taken into account by the designer in design floorplanning and placement of I/O flip-flops to minimize datapath skew and to help improve system timing margins. Other system-related issues to consider include PLL and clock partitioning.

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

***Figure 12-11* • All Silicon Features Selected for Fusion**

***Figure 12-15* • Programming Fusion Security Settings Only**

2. Choose the desired security level setting and enter the key(s).
   – The **High** security level employs FlashLock Pass Key with AES Key protection.
   – The **Medium** security level employs FlashLock Pass Key protection only.

***Figure 12-16* • High Security Level to Implement FlashLock Pass Key and AES Key Protection**

signal deactivated, which also has the effect of disabling the input buffers. The SAMPLE/PRELOAD instruction captures the status of pads in parallel and shifts them out as new data is shifted in for loading into the Boundary Scan Register (BSR). When the device is in an unprogrammed state, the OE and output BSR will be undefined; however, the input BSR will be defined as long as it is connected and being used. For JTAG timing information on setup, hold, and fall times, refer to the *FlashPro User's Guide*.

ISP Support in Flash-Based Devices

The flash FPGAs listed in Table 13-1 support the ISP feature and the functions described in this document.

***Table 13-1* • Flash-Based FPGAs Supporting ISP**

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
SmartFusion	SmartFusion	Mixed signal FPGA integrating ProASIC3 FPGA fabric, programmable microcontroller subsystem (MSS) which includes programmable analog and an ARM® Cortex™-M3 hard processor and flash memory in a monolithic device
Fusion	Fusion	Mixed signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device
ProASIC	ProASIC	First generation ProASIC devices
	ProASIC PLUS	Second generation ProASIC devices

*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 13-1. 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 13-1. 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*.

List of Changes

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

Date	Changes	Page
August 2012	This chapter will now be published standalone as an application note in addition to being part of the IGLOO/ProASIC3/Fusion FPGA fabric user's guides (SAR 38769).	N/A
	The "ISP Programming Header Information" section was revised to update the description of FP3-10PIN-ADAPTER-KIT in Table 13-3 • Programming Header Ordering Codes, clarifying that it is the adapter kit used for ProASIC^{PLUS} based boards, and also for ProASIC3 based boards where a compact programming header is being used (SAR 36779).	335
June 2011	The VPUMP programming mode voltage was corrected in Table 13-2 • Power Supplies. The correct value is 3.15 V to 3.45 V (SAR 30668).	329
	The notes associated with Figure 13-5 • Programming Header (top view) and Figure 13-6 • Board Layout and Programming Header Top View were revised to make clear the fact that IGLOO nano V2 devices can be programmed at 1.2 V (SAR 30787).	335, 337
	Figure 13-6 • Board Layout and Programming Header Top View was revised to include resistors tying TCK and TRST to GND. Microsemi recommends tying off TCK and TRST to GND if JTAG is not used (SAR 22921). RT ProASIC3 was added to the list of device families.	337
	In the "ISP Programming Header Information" section, the kit for adapting ProASIC^{PLUS} devices was changed from FP3-10PIN-ADAPTER-KIT to FP3-26PIN-ADAPTER-KIT (SAR 20878).	335
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
	References to FlashPro4 and FlashPro3X were added to this chapter, giving distinctions between them. References to SmartGen were deleted and replaced with Libero IDE Catalog.	N/A
	The "ISP Architecture" section was revised to indicate that V2 devices can be programmed at 1.2 V VCC with FlashPro4.	327
	SmartFusion was added to Table 13-1 • Flash-Based FPGAs Supporting ISP.	328
	The "Programming Voltage (VPUMP) and VJTAG" section was revised and 1.2 V was added to Table 13-2 • Power Supplies.	329
	The "Nonvolatile Memory (NVM) Programming Voltage" section is new.	329
	Cortex-M3 was added to the "Cortex-M1 and Cortex-M3 Device Security" section.	331
	In the "ISP Programming Header Information" section, the additional header adapter ordering number was changed from FP3-26PIN-ADAPTER to FP3-10PIN-ADAPTER-KIT, which contains 26-pin migration capability.	335
	The description of NC was updated in Figure 13-5 • Programming Header (top view), Table 13-4 • Programming Header Pin Numbers and Description and Figure 13-6 • Board Layout and Programming Header Top View.	335, 336
	The "Symptoms of a Signal Integrity Problem" section was revised to add that customers are expected to troubleshoot board-level signal integrity issues by measuring voltages and taking scope plots. "FlashPro4/3/3X allows TCK to be lowered from 6 MHz down to 1 MHz to allow you to address some signal integrity problems" formerly read, "from 24 MHz down to 1 MHz." "The Scan Chain command expects to see 0x2" was changed to 0x1.	337

Programming Algorithm

JTAG Interface

The low power flash families are fully compliant with the IEEE 1149.1 (JTAG) standard. They support all the mandatory boundary scan instructions (EXTEST, SAMPLE/PRELOAD, and BYPASS) as well as six optional public instructions (USERCODE, IDCODE, HIGHZ, and CLAMP).

IEEE 1532

The low power flash families are also fully compliant with the IEEE 1532 programming standard. The IEEE 1532 standard adds programming instructions and associated data registers to devices that comply with the IEEE 1149.1 standard (JTAG). These instructions and registers extend the capabilities of the IEEE 1149.1 standard such that the Test Access Port (TAP) can be used for configuration activities. The IEEE 1532 standard greatly simplifies the programming algorithm, reducing the amount of time needed to implement microprocessor ISP.

Implementation Overview

To implement device programming with a microprocessor, the user should first download the C-based STAPL player or DirectC code from the Microsemi SoC Products Group website. Refer to the website for future updates regarding the STAPL player and DirectC code.

http://www.microsemi.com/soc/download/program_debug/stapl/default.aspx

http://www.microsemi.com/soc/download/program_debug/directc/default.aspx

Using the easy-to-follow user's guide, create the low-level application programming interface (API) to provide the necessary basic functions. These API functions act as the interface between the programming software and the actual hardware (Figure 15-2).

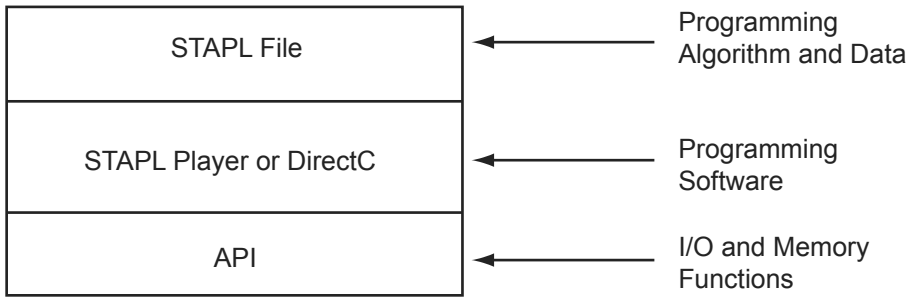


***Figure 15-2* • Device Programming Code Relationship**

The API is then linked with the STAPL player or DirectC and compiled using the microprocessor's compiler. Once the entire code is compiled, the user must download the resulting binary into the MCU system's program memory (such as ROM, EEPROM, or flash). The system is now ready for programming.

To program a design into the FPGA, the user creates a bitstream or STAPL file using the Microsemi Designer software, downloads it into the MCU system's volatile memory, and activates the stored programming binary file (Figure 15-3 on page 352). Once the programming is completed, the bitstream or STAPL file can be removed from the system, as the configuration profile is stored in the flash FPGA fabric and does not need to be reloaded at every system power-on.

The following devices and families do not support cold-sparing:

- IGLOO: AGL060, AGL125, AGL250, AGL600, AGL1000
- ProASIC3: A3P060, A3P125, A3P250, A3P400, A3P600, A3P1000
- ProASIC3L: A3P250L, A3P600L, A3P1000L
- Military ProASIC3: A3P1000

Hot-Swapping

Hot-swapping is the operation of hot insertion or hot removal of a card in a powered-up system. The I/Os need to be configured in hot-insertion mode if hot-swapping compliance is required. For more details on the levels of hot-swap compatibility in low power flash devices, refer to the "Hot-Swap Support" section in the I/O Structures chapter of the user's guide for the device you are using.

The following devices and families support hot-swapping:

- IGLOO: AGL015 and AGL030
- All IGLOO nano
- All IGLOO PLUS
- All IGLOOe
- ProASIC3L: A3PE3000L
- ProASIC3: A3P015 and A3P030
- All ProASIC3 nano
- All ProASIC3E
- Military ProASIC3EL: A3PE600L and A3PE3000L
- RT ProASIC3: RT3PE600L and RT3PE3000L

The following devices and families do not support hot-swapping:

- IGLOO: AGL060, AGL125, AGL250, AGL400, AGL600, AGL1000
- ProASIC3: A3P060, A3P125, A3P250, A3P400, A3P600, A3P1000
- ProASIC3L: A3P250L, A3P600L, A3P1000L
- Military ProASIC3: A3P1000

Conclusion

Microsemi's low power flash FPGAs provide an excellent programmable logic solution for a broad range of applications. In addition to high performance, low cost, security, nonvolatility, and single chip, they are live at power-up (meet Level 0 of the LAPU classification) and offer clear and easy-to-use power-up/-down characteristics. Unlike SRAM FPGAs, low power flash devices do not require any specific power-up/-down sequencing and have extremely low power-up inrush current in any power-up sequence. Microsemi low power flash FPGAs also support both cold-sparing and hot-swapping for applications requiring these capabilities.