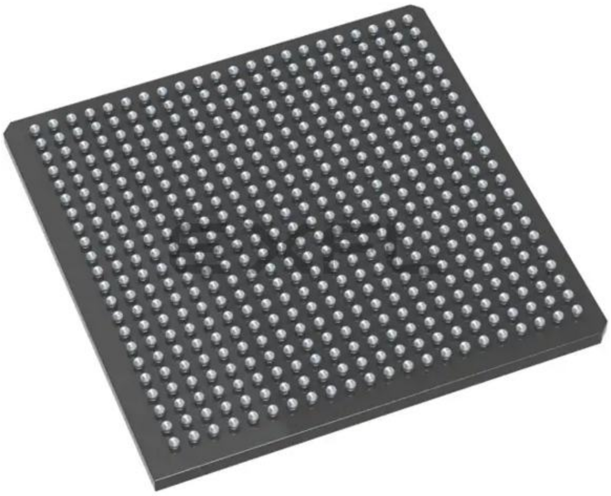
Microchip Technology - M1A3PE3000L-1FG484I Datasheet



Welcome to E-XFL.COM

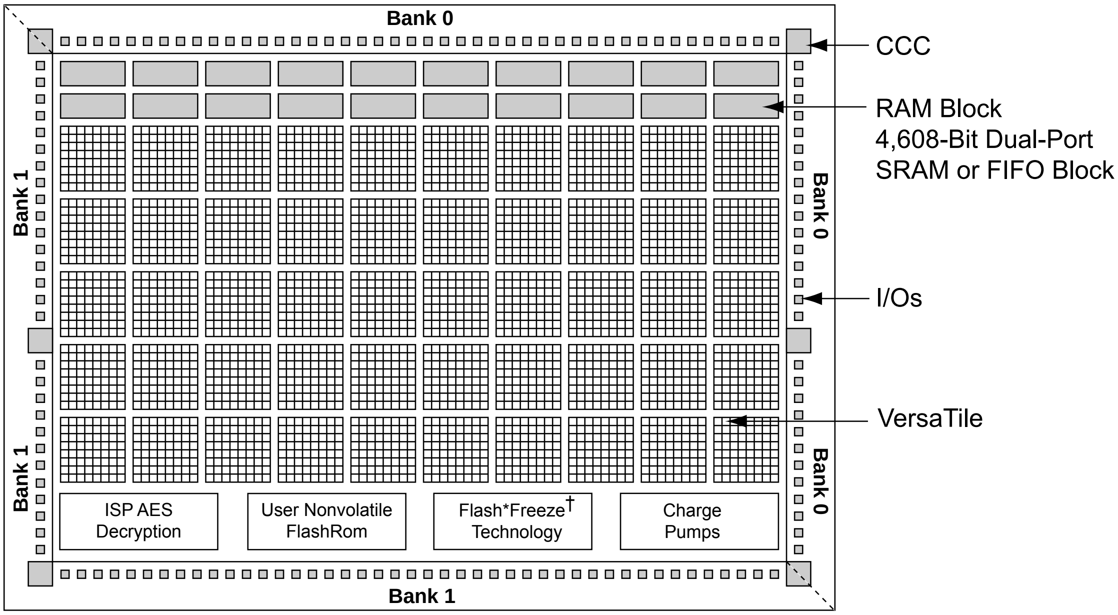
Understanding Embedded - FPGAs (Field Programmable Gate Array)

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

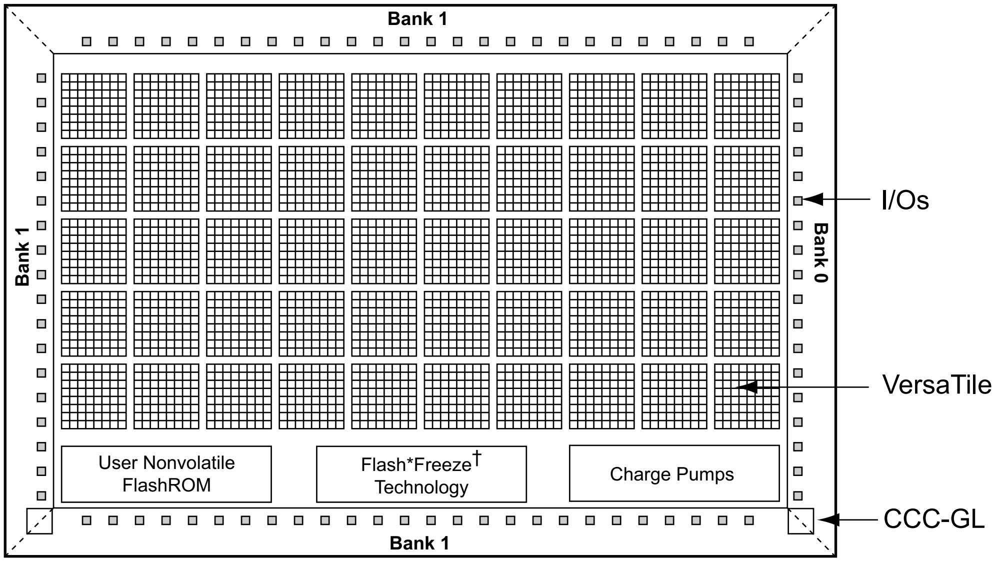
The versatility of Embedded - FPGAs makes them indispensable in numerous fields. In telecommunications,

Details	
Product Status	Active
Number of LABs/CLBs	-
Number of Logic Elements/Cells	-
Total RAM Bits	516096
Number of I/O	341
Number of Gates	3000000
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/m1a3pe3000l-1fg484i



Note: † Flash*Freeze mode is supported on IGLOO devices.

Figure 1-3 • IGLOO Device Architecture Overview with Two I/O Banks with RAM and PLL (60 k and 125 k gate densities)



Note: † Flash*Freeze mode is supported on IGLOO devices.

Figure 1-4 • IGLOO Device Architecture Overview with Three I/O Banks (AGLN015, AGLN020, A3PN015, and A3PN020)

Table 3-3 • **Quadrant Global Pin Name (continued)**

Differential I/O Pairs	GAAO/IOuxwByVz GAA1/IOuxwByVz	The output of the different pair will drive the global.
	GABO/IOuxwByVz GAB1/IOuxwByVz	The output of the different pair will drive the global.
	GACO/IOuxwByVz GAC1/IOuxwByVz	The output of the different pair will drive the global.
	GBAO/IOuxwByVz GBA1/IOuxwByVz	The output of the different pair will drive the global.
	GBBO/IOuxwByVz GBB1/IOuxwByVz	The output of the different pair will drive the global.
	GBCO/IOuxwByVz GBC1/IOuxwByVz	The output of the different pair will drive the global.
	GDAO/IOuxwByVz GDA1/IOuxwByVz	The output of the different pair will drive the global.
	GDBO/IOuxwByVz GDB1/IOuxwByVz	The output of the different pair will drive the global.
	GDCO/IOuxwByVz GDC1/IOuxwByVz	The output of the different pair will drive the global.
	GEAO/IOuxwByVz GEA1/IOuxwByVz	The output of the different pair will drive the global.
	GEBO/IOuxwByVz GEB1/IOuxwByVz	The output of the different pair will drive the global.
	GECO/IOuxwByVz GEC1/IOuxwByVz	The output of the different pair will drive the global.

Note: Only one of the I/Os can be directly connected to a quadrant at a time.

Unused Global I/O Configuration

The unused clock inputs behave similarly to the unused Pro I/Os. The Microsemi Designer software automatically configures the unused global pins as inputs with pull-up resistors if they are not used as regular I/O.

I/O Banks and Global I/O Standards

In low power flash devices, any I/O or internal logic can be used to drive the global network. However, only the global macro placed at the global pins will use the hardwired connection between the I/O and global network. Global signal (signal driving a global macro) assignment to I/O banks is no different from regular I/O assignment to I/O banks with the exception that you are limited to the pin placement location available. Only global signals compatible with both the VCCI and VREF standards can be assigned to the same bank.

You can control the maximum number of shared instances allowed for the legalization to take place using the Compile Option dialog box shown in Figure 3-17. Refer to Libero SoC / Designer online help for details on the Compile Option dialog box. A large number of shared instances most likely indicates a floorplanning problem that you should address.

***Figure 3-17* • Shared Instances in the Compile Option Dialog Box**

Designer Flow for Global Assignment

To achieve the desired result, pay special attention to global management during synthesis and place-and-route. The current Synplify tool does not insert more than six global buffers in the netlist by default. Thus, the default flow will not assign any signal to the quadrant global network. However, you can use attributes in Synplify and increase the default global macro assignment in the netlist. Designer v6.2 supports automatic quadrant global assignment, which was not available in Designer v6.1. Layout will make the choice to assign the correct signals to global. However, you can also utilize PDC and perform manual global assignment to overwrite any automatic assignment. The following step-by-step suggestions guide you in the layout of your design and help you improve timing in Designer:

1. Run Compile and check the Compile report. The Compile report has global information in the "Device Utilization" section that describes the number of chip and quadrant signals in the design. A "Net Report" section describes chip global nets, quadrant global nets, local clock nets, a list of nets listed by fanout, and net candidates for local clock assignment. Review this information. Note that YB or YC are counted as global only when they are used in isolation; if you use YB only and not GLB, this net is not shown in the global/quadrant nets report. Instead, it appears in the Global Utilization report.
2. If some signals have a very high fanout and are candidates for global promotion, promote those signals to global using the compile options or PDC commands. Figure 3-18 on page 70 shows the Globals Management section of the compile options. Select **Promote regular nets whose fanout is greater than** and enter a reasonable value for fanouts.

Step 1

Run Synthesis with default options. The Synplicity log shows the following device utilization:

```
Cell usage:
                    cell count      area      count*area
DFN1E1C1               1536          2.0        3072.0
BUFF                    278          1.0         278.0
INBUF                    10          0.0           0.0
VCC                       9          0.0           0.0
GND                       9          0.0           0.0
OUTBUF                    6          0.0           0.0
CLKBUF                    3          0.0           0.0
PLL                       2          0.0           0.0
TOTAL                  1853                     3350.0
```

Step 2

Run Compile with the **Promote regular nets whose fanout is greater than** option selected in Designer; you will see the following in the Compile report:

```
Device utilization report:
==========================
CORE                Used:   1536  Total:  13824   (11.11%)
IO (W/ clocks)      Used:     19  Total:    147   (12.93%)
Differential IO     Used:      0  Total:     65    (0.00%)
GLOBAL              Used:      8  Total:     18   (44.44%)
PLL                 Used:      2  Total:      2  (100.00%)
RAM/FIFO            Used:      0  Total:     24    (0.00%)
FlashROM            Used:      0  Total:      1    (0.00%)
........................
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   : QCLK1_c
                      Driver: QCLK1_pad_CLKINT
                      Source: AUTO PROMOTED
256     CLK_NET       Net   : QCLK2_c
                      Driver: QCLK2_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
```

Designer will promote five more signals to global due to high fanout. There are eight signals assigned to global networks.

Using Spines of Occupied Global Networks

When a signal is assigned to a global network, the flash switches are programmed to set the MUX select lines (explained in the "Clock Aggregation Architecture" section on page 61) to drive the spines of that network with the global net. However, if the global net is restricted from reaching into the scope of a spine, the MUX drivers of that spine are available for other high-fanout or critical signals (Figure 3-20).

For example, if you want to limit the CLK1_c signal to the left half of the chip and want to use the right side of the same global network for CLK2_c, you can add the following PDC commands:

```
define_region -name region1 -type inclusive 0 0 34 29
assign_net_macros region1 CLK1_c
assign_local_clock –net CLK2_c –type chip B2
```

Figure 3-20 **• Design Example Using Spines of Occupied Global Networks**

Conclusion

IGLOO, Fusion, and ProASIC3 devices contain 18 global networks: 6 chip global networks and 12 quadrant global networks. These global networks can be segmented into local low-skew networks called spines. The spines provide low-skew networks for the high-fanout signals of a design. These allow you up to 252 different internal/external clocks in an A3PE3000 device. This document describes the architecture for the global network, plus guidelines and methodologies in assigning signals to globals and spines.

Related Documents

User's Guides

IGLOO, ProASIC3, SmartFusion, and Fusion Macro Library Guide

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

Available I/O Standards

Table 4-4 **• Available I/O Standards within CLKBUF and CLKBUF_LVDS/LVPECL Macros**

CLKBUF_LVCMOS5
CLKBUF_LVCMOS33 [1]
CLKBUF_LVCMOS25 [2]
CLKBUF_LVCMOS18
CLKBUF_LVCMOS15
CLKBUF_PCI
CLKBUF_PCIX [3]
CLKBUF_GTL25 [2,3]
CLKBUF_GTL33 [2,3]
CLKBUF_GTLP25 [2,3]
CLKBUF_GTLP33 [2,3]
CLKBUF_HSTL_I [2,3]
CLKBUF_HSTL_II [2,3]
CLKBUF_SSTL3_I [2,3]
CLKBUF_SSTL3_II [2,3]
CLKBUF_SSTL2_I [2,3]
CLKBUF_SSTL2_II [2,3]
CLKBUF_LVDS [4,5]
CLKBUF_LVPECL[5]

Notes:

1. *By default, the CLKBUF macro uses 3.3 V LVTTL I/O technology. For more details, refer to the IGLOO, ProASIC3, SmartFusion, and Fusion Macro Library Guide.*
2. *I/O standards only supported in ProASIC3E and IGLOOe families.*
3. *I/O standards only supported in the following Fusion devices: AFS600 and AFS1500.*
4. *B-LVDS and M-LVDS standards are supported by CLKBUF_LVDS.*
5. *Not supported for IGLOO nano and ProASIC3 nano devices.*

Global Synthesis Constraints

The Synplify® synthesis tool, by default, allows six clocks in a design for Fusion, IGLOO, and ProASIC3. When more than six clocks are needed in the design, a user synthesis constraint attribute, syn_global_buffers, can be used to control the maximum number of clocks (up to 18) that can be inferred by the synthesis engine.

High-fanout nets will be inferred with clock buffers and/or internal clock buffers. If the design consists of CCC global buffers, they are included in the count of clocks in the design.

The subsections below discuss the clock input source (global buffers with no programmable delays) and the clock conditioning functional block (global buffers with programmable delays and/or PLL function) in detail.

```
DLYGLC[4:0]     00000
DLYYB[4:0]      00000
DLYYC[4:0]      00000
VCOSEL[2:0]     100

Primary Clock Frequency 33.000
Primary Clock Phase Shift 0.000
Primary Clock Output Delay from CLKA 1.695

Secondary1 Clock Frequency 40.000
Secondary1 Clock Phase Shift 0.000
Secondary1 Clock Global Output Delay from CLKB 0.200

Secondary2 Clock Frequency 50.000
Secondary2 Clock Phase Shift 0.000
Secondary2 Clock Global Output Delay from CLKC 0.200

######################################
# Dynamic Stream Data
######################################
--------------------------------------
|NAME    |SDIN     |VALUE   |TYPE     |
--------------------------------------
|FINDIV  |[6:0]    |0000101 |EDIT     |
|FBDIV   |[13:7]   |0100000 |EDIT     |
|OADIV   |[18:14]  |00100   |EDIT     |
|OBDIV   |[23:19]  |00000   |EDIT     |
|OCDIV   |[28:24]  |00000   |EDIT     |
|OAMUX   |[31:29]  |100     |EDIT     |
|OBMUX   |[34:32]  |000     |EDIT     |
|OCMUX   |[37:35]  |000     |EDIT     |
|FBSEL   |[39:38]  |01      |EDIT     |
|FBDLY   |[44:40]  |00000   |EDIT     |
|XDLYSEL |[45]     |0       |EDIT     |
|DLYGLA  |[50:46]  |00000   |EDIT     |
|DLYGLB  |[55:51]  |00000   |EDIT     |
|DLYGLC  |[60:56]  |00000   |EDIT     |
|DLYYB   |[65:61]  |00000   |EDIT     |
|DLYYC   |[70:66]  |00000   |EDIT     |
|STATASEL|[71]     |X       |MASKED   |
|STATBSEL|[72]     |X       |MASKED   |
|STATCSEL|[73]     |X       |MASKED   |
|VCOSEL  |[76:74]  |100     |EDIT     |
|DYNASEL |[77]     |X       |MASKED   |
|DYNBSEL |[78]     |X       |MASKED   |
|DYNCSEL |[79]     |X       |MASKED   |
|RESETEN |[80]     |1       |READONLY |
```

Below is the resultant Verilog HDL description of a legal dynamic PLL core configuration generated by SmartGen:

```
module dyn_pll_macro(POWERDOWN, CLKA, LOCK, GLA, GLB, GLC, SDIN, SCLK, SSHIFT, SUPDATE,
  MODE, SDOUT, CLKB, CLKC);

input POWERDOWN, CLKA;
output  LOCK, GLA, GLB, GLC;
input  SDIN, SCLK, SSHIFT, SUPDATE, MODE;
output  SDOUT;
input  CLKB, CLKC;

  wire VCC, GND;

  VCC VCC_1_net(.Y(VCC));
  GND GND_1_net(.Y(GND));
```

Recommended Board-Level Considerations

The power to the PLL core is supplied by VCCPLA/B/C/D/E/F (VCCPLx), and the associated ground connections are supplied by VCOMPLA/B/C/D/E/F (VCOMPLx). When the PLLs are not used, the Designer place-and-route tool automatically disables the unused PLLs to lower power consumption. The user should tie unused VCCPLx and VCOMPLx pins to ground. Optionally, the PLL can be turned on/off during normal device operation via the POWERDOWN port (see Table 4-3 on page 84).

PLL Power Supply Decoupling Scheme

The PLL core is designed to tolerate noise levels on the PLL power supply as specified in the datasheets. When operated within the noise limits, the PLL will meet the output peak-to-peak jitter specifications specified in the datasheets. User applications should always ensure the PLL power supply is powered from a noise-free or low-noise power source.

However, in situations where the PLL power supply noise level is higher than the tolerable limits, various decoupling schemes can be designed to suppress noise to the PLL power supply. An example is provided in Figure 4-38. The VCCPLx and VCOMPLx pins correspond to the PLL analog power supply and ground.

Microsemi strongly recommends that two ceramic capacitors (10 nF in parallel with 100 nF) be placed close to the power pins (less than 1 inch away). A third generic 10 µF electrolytic capacitor is recommended for low-frequency noise and should be placed farther away due to its large physical size. Microsemi recommends that a 6.8 µH inductor be placed between the supply source and the capacitors to filter out any low-/medium- and high-frequency noise. In addition, the PCB layers should be controlled so the VCCPLx and VCOMPLx planes have the minimum separation possible, thus generating a good-quality RF capacitor.

For more recommendations, refer to the *Board-Level Considerations* application note.

Recommended 100 nF capacitor:

- Producer BC Components, type X7R, 100 nF, 16 V
- BC Components part number: 0603B104K160BT
- Digi-Key part number: BC1254CT-ND
- Digi-Key part number: BC1254TR-ND

Recommended 10 nF capacitor:

- Surface-mount ceramic capacitor
- Producer BC Components, type X7R, 10 nF, 50 V
- BC Components part number: 0603B103K500BT
- Digi-Key part number: BC1252CT-ND
- Digi-Key part number: BC1252TR-ND

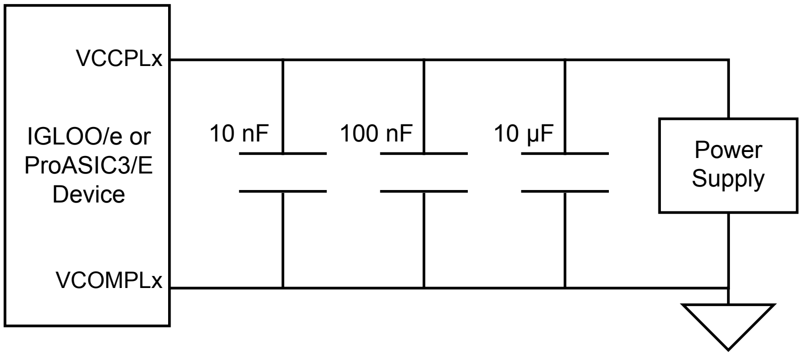


***Figure 4-38* • Decoupling Scheme for One PLL (should be replicated for each PLL used)**

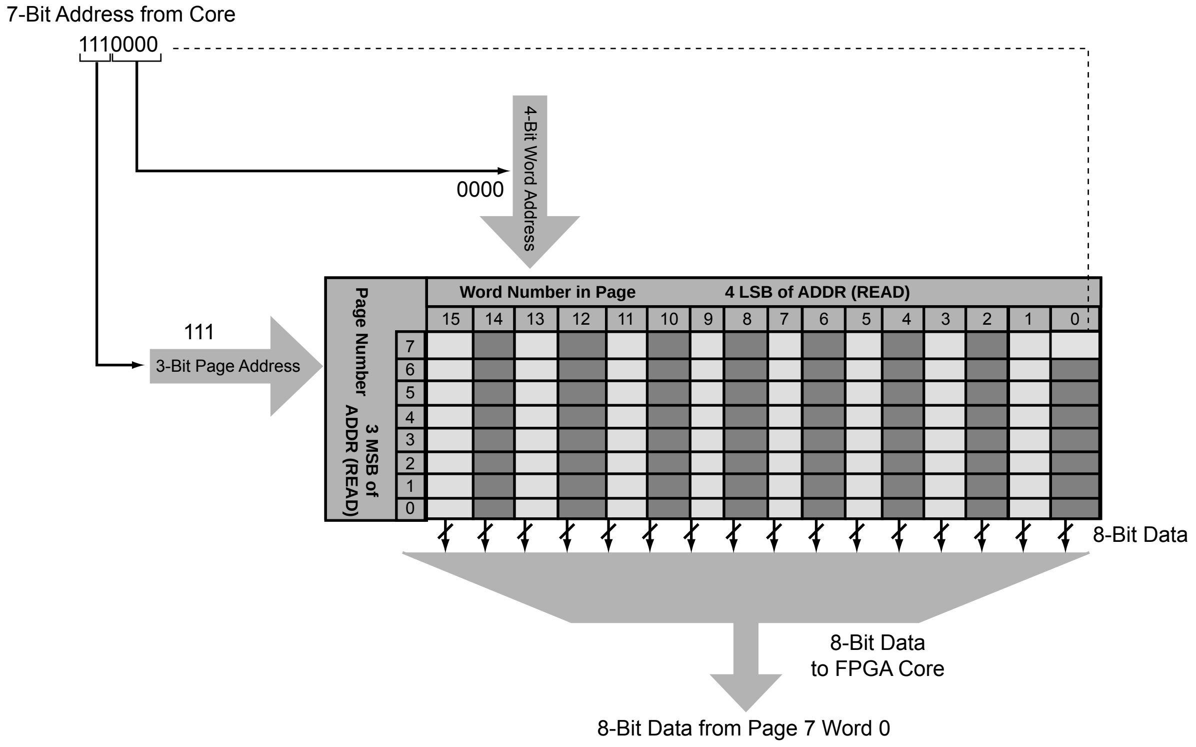


***Figure 5-7* • Accessing FlashROM Using FPGA Core**

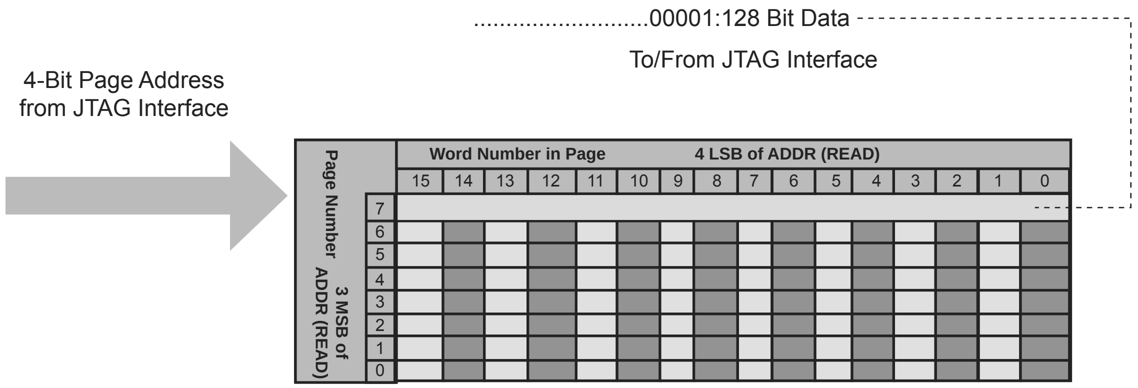


***Figure 5-8* • Accessing FlashROM Using JTAG Port**

Initializing the RAM/FIFO

The SRAM blocks can be initialized with data to use as a lookup table (LUT). Data initialization can be accomplished either by loading the data through the design logic or through the UJTAG interface. The UJTAG macro is used to allow access from the JTAG port to the internal logic in the device. By sending the appropriate initialization string to the JTAG Test Access Port (TAP) Controller, the designer can put the JTAG circuitry into a mode that allows the user to shift data into the array logic through the JTAG port using the UJTAG macro. For a more detailed explanation of the UJTAG macro, refer to the "FlashROM in Microsemi's Low Power Flash Devices" section on page 133.

A user interface is required to receive the user command, initialization data, and clock from the UJTAG macro. The interface must synchronize and load the data into the correct RAM block of the design. The main outputs of the user interface block are the following:

- Memory block chip select: Selects a memory block for initialization. The chip selects signals for each memory block that can be generated from different user-defined pockets or simple logic, such as a ring counter (see below).
- Memory block write address: Identifies the address of the memory cell that needs to be initialized.
- Memory block write data: The interface block receives the data serially from the UTDI port of the UJTAG macro and loads it in parallel into the write data ports of the memory blocks.
- Memory block write clock: Drives the WCLK of the memory block and synchronizes the write data, write address, and chip select signals.

Figure 6-8 shows the user interface between UJTAG and the memory blocks.

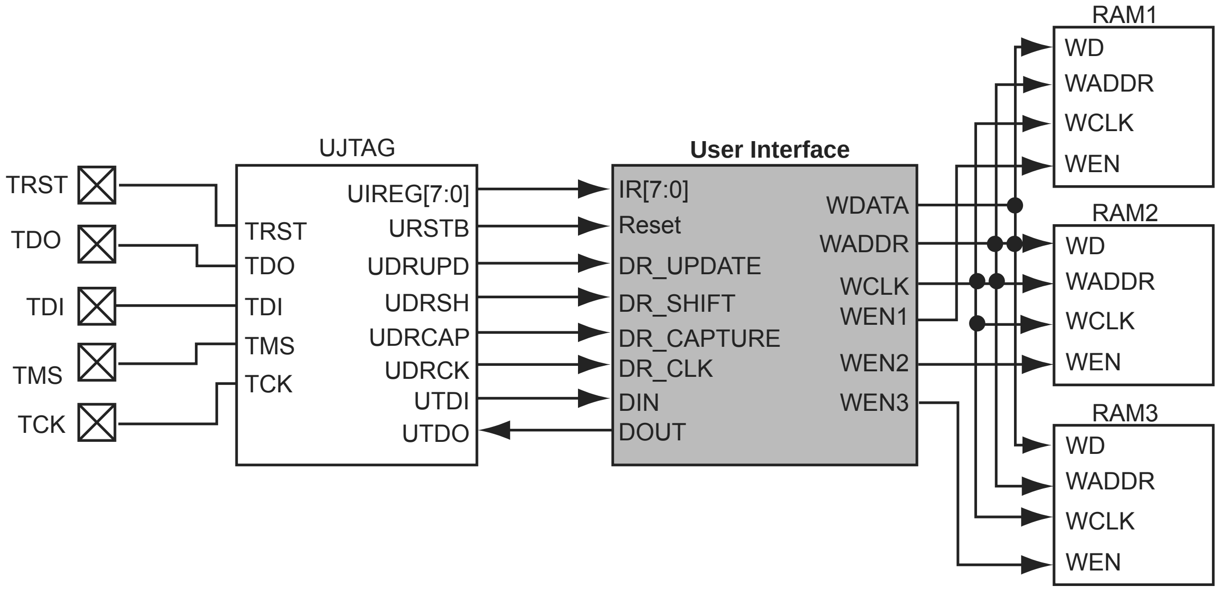


Figure 6-8 **• Interfacing TAP Ports and SRAM Blocks**

An important component of the interface between the UJTAG macro and the RAM blocks is a serial-in/parallel-out shift register. The width of the shift register should equal the data width of the RAM blocks. The RAM data arrives serially from the UTDI output of the UJTAG macro. The data must be shifted into a shift register clocked by the JTAG clock (provided at the UDRCK output of the UJTAG macro).

Then, after the shift register is fully loaded, the data must be transferred to the write data port of the RAM block. To synchronize the loading of the write data with the write address and write clock, the output of the shift register can be pipelined before driving the RAM block.

The write address can be generated in different ways. It can be imported through the TAP using a different instruction opcode and another shift register, or generated internally using a simple counter. Using a counter to generate the address bits and sweep through the address range of the RAM blocks is

I/O Architecture

I/O Tile

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

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

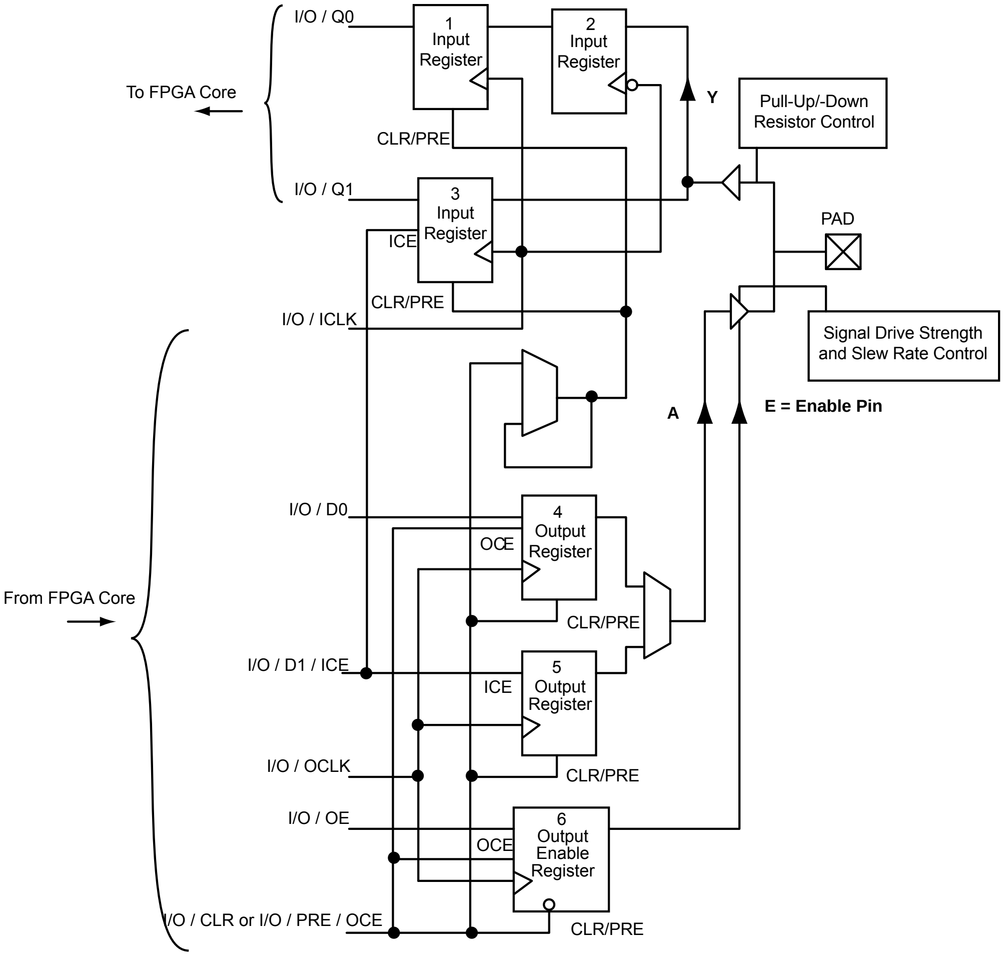


Figure 7-2 • **DDR Configured I/O Block Logical Representation**

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)

I/O Register Combining

Every I/O has several embedded registers in the I/O tile that are close to the I/O pads. Rather than using the internal register from the core, the user has the option of using these registers for faster clock-to-out timing, and external hold and setup. When combining these registers at the I/O buffer, some architectural rules must be met. Provided these rules are met, the user can enable register combining globally during Compile (as shown in the "Compiling the Design" section on page 261).

This feature is supported by all I/O standards.

Rules for Registered I/O Function

1. The fanout between an I/O pin (D, Y, or E) and a register must be equal to one for combining to be considered on that pin.
2. All registers (Input, Output, and Output Enable) connected to an I/O must share the same clear or preset function:
   - If one of the registers has a CLR pin, all the other registers that are candidates for combining in the I/O must have a CLR pin.
   - If one of the registers has a PRE pin, all the other registers that are candidates for combining in the I/O must have a PRE pin.
   - If one of the registers has neither a CLR nor a PRE pin, all the other registers that are candidates for combining must have neither a CLR nor a PRE pin.
   - If the clear or preset pins are present, they must have the same polarity.
   - If the clear or preset pins are present, they must be driven by the same signal (net).
3. Registers connected to an I/O on the Output and Output Enable pins must have the same clock and enable function:
   - Both the Output and Output Enable registers must have an E pin (clock enable), or none at all.
   - If the E pins are present, they must have the same polarity. The CLK pins must also have the same polarity.

In some cases, the user may want registers to be combined with the input of a bibuf while maintaining the output as-is. This can be achieved by using PDC commands as follows:

```
set_io <signal name> -REGISTER yes ------register will combine
set_preserve <signal name> ----register will not combine
```

Weak Pull-Up and Weak Pull-Down Resistors

When the I/O is pulled up, it is connected to the VCCI of its corresponding I/O bank. When it is pulled down, it is connected to GND. Refer to the datasheet for more information.

For low power applications, configuration of the pull-up or pull-down of the I/O can be used to set the I/O to a known state while the device is in Flash*Freeze mode. Refer to the "Flash*Freeze Technology and Low Power Modes in IGLOO and ProASIC3L Devices" chapter in the *IGLOOe FPGA Fabric User’s Guide* or *ProASIC3E FPGA Fabric User’s Guide* for more information.

The Flash*Freeze (FF) pin cannot be configured with a weak pull-down or pull-up I/O attribute, as the signal needs to be driven at all times.

Output Slew Rate Control

The slew rate is the amount of time an input signal takes to get from logic LOW to logic HIGH or vice versa.

It is commonly defined as the propagation delay between 10% and 90% of the signal's voltage swing. Slew rate control is available for the output buffers of low power flash devices. The output buffer has a programmable slew rate for both HIGH-to-LOW and LOW-to-HIGH transitions. Slew rate control is available for LVTTL, LVCMOS, and PCI-X I/O standards. The other I/O standards have a preset slew value.

The slew rate can be implemented by using a PDC command (Table 8-6 on page 218), setting it "High" or "Low" in the I/O Attribute Editor in Designer, or instantiating a special I/O macro. The default slew rate value is "High."

IGLOOe and ProASIC3E devices support output slew rate control: high and low. Microsemi recommends the high slew rate option to minimize the propagation delay. This high-speed option may introduce noise into the system if appropriate signal integrity measures are not adopted. Selecting a low slew rate reduces this kind of noise but adds some delays in the system. Low slew rate is recommended when bus transients are expected.

Output Drive

The output buffers of IGLOOe and ProASIC3E devices can provide multiple drive strengths to meet signal integrity requirements. The LVTTL and LVCMOS (except 1.2 V LVCMOS) standards have selectable drive strengths. Other standards have a preset value.

Drive strength should also be selected according to the design requirements and noise immunity of the system.

The output slew rate and multiple drive strength controls are available in LVTTL/LVCMOS 3.3 V, LVCMOS 2.5 V, LVCMOS 2.5 V / 5.0 V input, LVCMOS 1.8 V, and LVCMOS 1.5 V. All other I/O standards have a high output slew rate by default.

For other IGLOOe and ProASIC3E devices, refer to Table 8-15 for more information about the slew rate and drive strength specification.

There will be a difference in timing between the Standard Plus I/O banks and the Advanced I/O banks. Refer to the I/O timing tables in the datasheet for the standards supported by each device.

***Table 8-15* • IGLOOe and ProASIC3E I/O Standards—Output Drive and Slew Rate**

I/O Standards	2 mA	4 mA	6 mA	8 mA	12 mA	16 mA	24 mA	Slew	
LVTTL/LVCMOS 3.3 V	✓	✓	✓	✓	✓	✓	✓	High	Low
LVCMOS 2.5 V	✓	✓	✓	✓	✓	✓	✓	High	Low
LVCMOS 2.5/5.0 V	✓	✓	✓	✓	✓	✓	✓	High	Low
LVCMOS 1.8 V	✓	✓	✓	✓	✓	✓	–	High	Low
LVCMOS 1.5 V	✓	✓	✓	✓	✓	–	–	High	Low

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**

DDR Support in Flash-Based Devices

The flash FPGAs listed in Table 10-1 support the DDR feature and the functions described in this document.

Table 10-1 **• 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
ProASIC3	ProASIC3	Low power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 10-1. 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 10-1. 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*.

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.

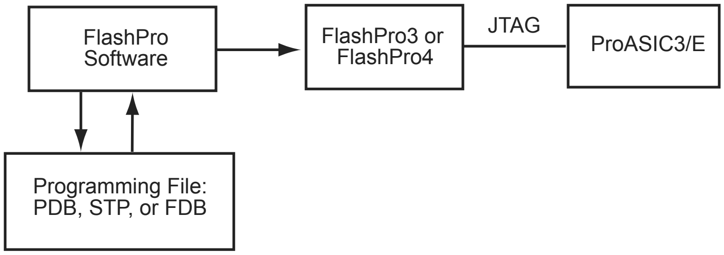


***Figure 11-1* • FlashPro Programming Setup**

IEEE 1532 (JTAG) Interface

The supported industry-standard IEEE 1532 programming interface builds on the IEEE 1149.1 (JTAG) standard. IEEE 1532 defines the standardized process and methodology for ISP. Both silicon and software issues are addressed in IEEE 1532 to create a simplified ISP environment. Any IEEE 1532 compliant programmer can be used to program low power flash devices. Device serialization is not supported when using the IEEE1532 standard. Refer to the standard for detailed information about IEEE 1532.

Security

Unlike SRAM-based FPGAs that require loading at power-up from an external source such as a microcontroller or boot PROM, Microsemi nonvolatile devices are live at power-up, and there is no bitstream required to load the device when power is applied. The unique flash-based architecture prevents reverse engineering of the programmed code on the device, because the programmed data is stored in nonvolatile memory cells. Each nonvolatile memory cell is made up of small capacitors and any physical deconstruction of the device will disrupt stored electrical charges.

Each low power flash device has a built-in 128-bit Advanced Encryption Standard (AES) decryption core, except for the 30 k gate devices and smaller. Any FPGA core or FlashROM content loaded into the device can optionally be sent as encrypted bitstream and decrypted as it is loaded. This is particularly suitable for applications where device updates must be transmitted over an unsecured network such as the Internet. The embedded AES decryption core can prevent sensitive data from being intercepted (Figure 13-1 on page 331). A single 128-bit AES Key (32 hex characters) is used to encrypt FPGA core programming data and/or FlashROM programming data in the Microsemi tools. The low power flash devices also decrypt with a single 128-bit AES Key. In addition, low power flash devices support a Message Authentication Code (MAC) for authentication of the encrypted bitstream on-chip. This allows the encrypted bitstream to be authenticated and prevents erroneous data from being programmed into the device. The FPGA core, FlashROM, and Flash Memory Blocks (FBs), in Fusion only, can be updated independently using a programming file that is AES-encrypted (cipher text) or uses plain text.

***Table 13-4* • Programming Header Pin Numbers and Description**

Pin	Signal	Source	Description
1	TCK	Programmer	JTAG Clock
2	GND[1]	–	Signal Reference
3	TDO	Target Board	Test Data Output
4	NC	–	No Connect (FlashPro3/3X); Prog_Mode (FlashPro4). See note associated with Figure 13-5 on page 335 regarding Prog_Mode on FlashPro4.
5	TMS	Programmer	Test Mode Select
6	VJTAG	Target Board	JTAG Supply Voltage
7	VPUMP[2]	Programmer/Target Board	Programming Supply Voltage
8	nTRST	Programmer	JTAG Test Reset (Hi-Z with 10 kΩ pull-down, HIGH, LOW, or toggling)
9	TDI	Programmer	Test Data Input
10	GND[1]	–	Signal Reference

Notes:

1. *Both GND pins must be connected.*
2. *FlashPro4/3/3X can provide VPUMP if there is only one device on the target board.*

Index