

# Microchip Technology CoreJTAGDebug Processors User Guide

[Home](#) » [Microchip Technology](#) » Microchip Technology CoreJTAGDebug Processors User Guide 

## Microchip Technology Core JTAG Debug Processors User Guide



## Contents

- 1 Introduction
- 2 Functional Description
  - 2.1 Device Chaining
- 3 Interface
- 4 Register Map and Descriptions
- 5 Tool Flow
  - 5.1 License
  - 5.2 Configuring CoreJTAGDebug in SmartDesign
  - 5.3 Simulation Flows
  - 5.4 Synthesis in Libero
  - 5.5 Place-and-Route in Libero
  - 5.6 Device Programming
- 6 System Integration
  - 6.1 System Level Design for IGLOO2/RTG4
  - 6.2 System Level Design for SmartFusion2
  - 6.3 UJTAG\_SEC
- 7 Design Constraints
- 8 Revision History
- 9 Product Change Notification Service
- 10 Microchip Devices Code Protection Feature
- 11 Legal Notice
- 12 Documents / Resources
  - 12.1 References
- 13 Related Posts

## Introduction

Core JTAG Debug v4.0 facilitates the connection of Joint Test Action Group (JTAG) compatible soft core processors to the JTAG TAP or General Purpose Input/Output (GPIO) pins for debugging. This IP core facilitates the debugging of a maximum of 16 soft core processors within a single device, and also provides support for debugging of processors on four separate devices over GPIO.

## Features

**CoreJTAGDebug has the following key features:**

- Provides the fabric access to the JTAG interface through the JTAG TAP.
- Provides the fabric access to the JTAG interface through the GPIO pins.
- Configures the IR Code support for the JTAG tunneling.
- Supports the linking of multiple devices through the JTAG TAP.
- Supports the multi-processor debugging.
- Promotes separate clock and reset signals to the low-skew routing resources.
- Supports both active-low and active-high target resetting.
- Supports the JTAG Security Monitor Interface (UJTAG\_SEC) for PolarFire devices.

## Core Version

This document applies to CoreJTAGDebug v4.0

## Supported Families

- PolarFire®
- RTG4™
- IGLOO® 2
- SmartFusion® 2
- SmartFusion
- ProASIC3/3E/3L
- IGLOO
- IGLOOe/+

### Device Utilization and Performance

Utilization and performance data is listed in the following table for the supported device families. The data listed in this table is only indicative. The overall device utilization and performance of the core is system dependent.

**Table 1. Device Utilization and Performance**

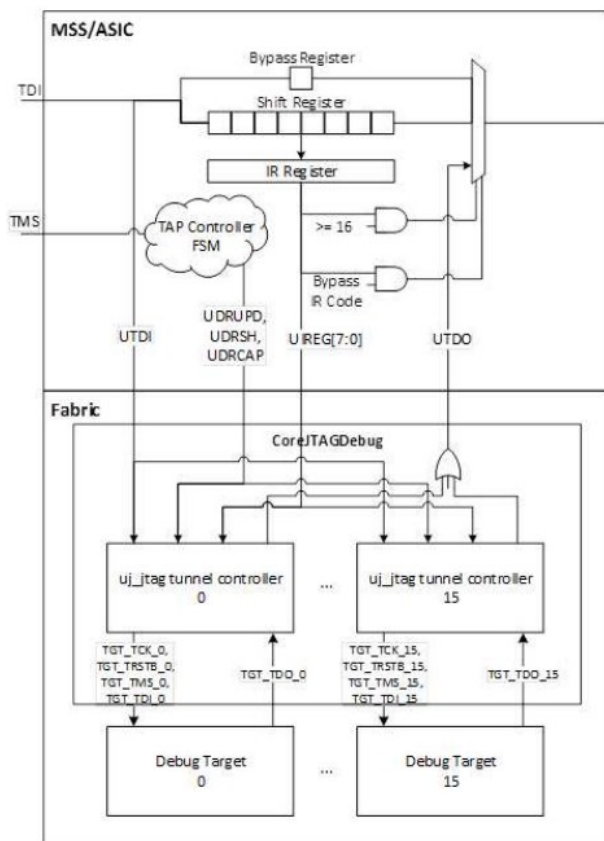
Family	Tiles Sequential	Combinatorial	Total	Utilization Device	Total %	Performance (MHz)
PolarFire	17	116	299554	MPF300TS	0.04	111.111
RTG4	19	121	151824	RT4G150	0.09	50
SmartFusion2	17	120	56340	M2S050	0.24	69.47
IGLOO2	17	120	56340	M2GL050	0.24	68.76
SmartFusion	17	151	4608	A2F200M3F	3.65	63.53
IGLOO	17	172	3072	AFL125V5	6.15	69.34
ProASIC3	17	157	13824	A3P600	1.26	50

**Note:** Data in this table was achieved using the Verilog RTL with typical synthesis and layout settings on -1 parts. Top-level parameters or generics were left at default settings.

### Functional Description

CoreJTAGDebug uses the UJTAG hard macro to provide access to the JTAG interface from the FPGA fabric. The UJTAG hard macro facilitates connecting to the output of the MSS or ASIC TAP controller from the fabric. Only, one instance of the UJTAG macro is allowed in the fabric.

**Figure 1-1. CoreJTAGDebug Block Diagram**



CoreJTAGDebug contains an instantiation of the `uj_jtag` tunnel controller, which implements a JTAG tunnel controller to facilitate JTAG tunneling between a FlashPro programmer and a target softcore processor. The softcore processor is connected through the dedicated FPGA's JTAG interface pins. IR scans from the JTAG interface are inaccessible in the FPGA fabric. Hence, the tunnel protocol is required to facilitate IR and DR scans to the debug target, which supports the industry standard JTAG interface. The tunnel controller decodes the tunnel packet transferred as a DR scan and generates a resultant IR or DR scan, based on the contents of the tunnel packet and the contents of the IR register provided through UIREG. The tunnel controller also decodes the tunnel packet, when the contents of the IR register matches its IR code.

**Figure 1-2. Tunnel Packet Protocol**

Entry TMS Length [2:0]	Entry TMS Data (0-7)	Payload Data Length [5:0]	Payload Data Data (0-63)	Exit TMS Length [2:0]	Exit TMS Data (0-7)
---------------------------	-------------------------	------------------------------	-----------------------------	--------------------------	------------------------

A configuration parameter provides configuration of the IR code used by the tunnel controller. To facilitate the debugging of multiple softcore processors inside a single design, the number of tunnel controllers instantiated are configurable from 1-16, providing a JTAG compliant interface to each target processor. These target processors are each addressable through a unique IR code set at instantiation time.

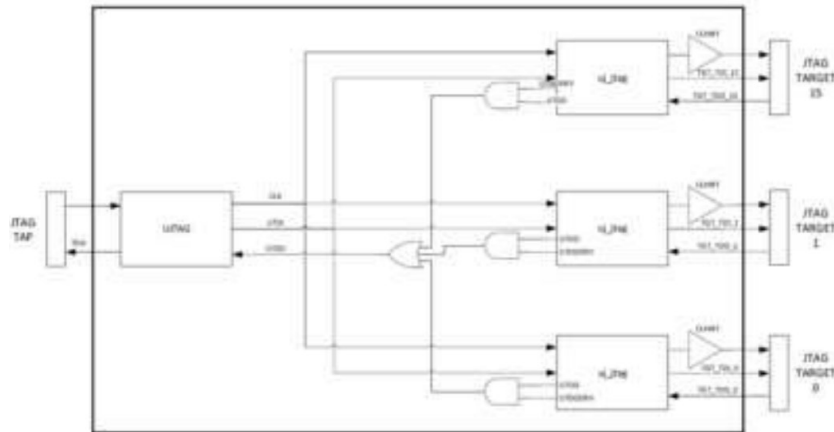
A CLKINT or BFR buffer is instantiated on the TGT\_TCK line of each target processor debug interface.

The URSTB line from the UJTAG macro (TRSTB) is promoted to a global resource within CoreJTAGDebug. An optional inverter is placed on the TGT\_TRST line within CoreJTAGDebug for connection to a debug target, which is then expected to be connected to an active-high reset source. It is configured when it is assumed that the incoming TRSTB signal from the JTAG TAP is active low. If this configuration requires one or more debug targets, an additional global routing resource will be consumed.

The URSTB line from the UJTAG macro (TRSTB) is promoted to a global resource within CoreJTAGDebug. An optional inverter is placed on the TGT\_TRST line within CoreJTAGDebug for connection to a debug target, which is then expected to be connected to an active-high reset source. It is configured when it is assumed that the incoming TRSTB signal from the JTAG TAP is active low. TGT\_TRSTN is the default active low output for the

debug target. If this configuration requires one or more debug targets, an additional global routing resource will be consumed.

**Figure 1-3. CoreJTAGDebug Serial Data and Clocking**



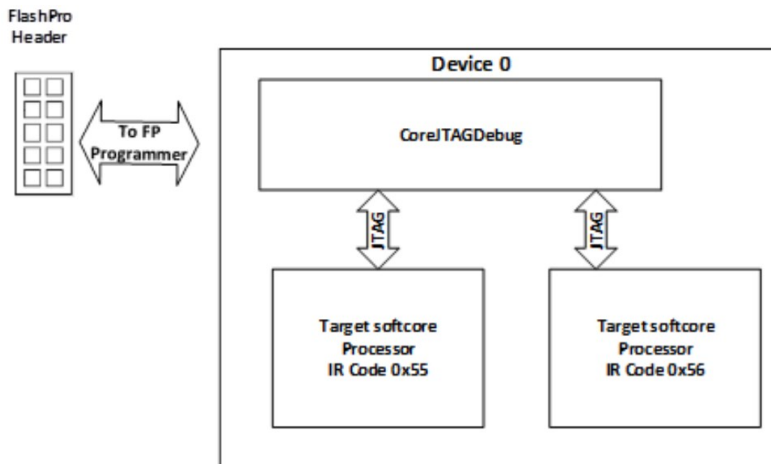
### Device Chaining

Refer to the FPGA Programming User Guides for the specific development board or family. Each development board may operate at different voltages, and you may choose to verify if it is possible with their development platforms. Also, if you are using multiple development boards, ensure that, they share a common ground.

### Through FlashPro Header

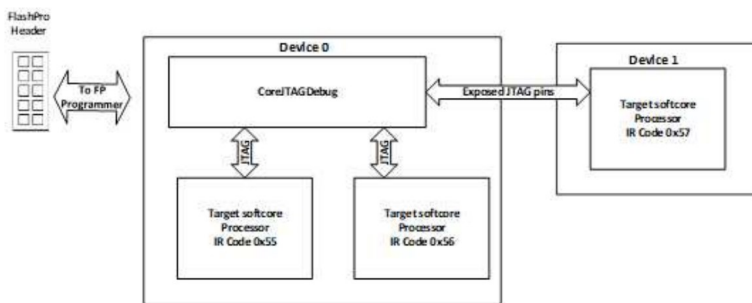
To support the chaining of multiple devices in the fabric using the FlashPro header, multiple instances of `uj_jtag` are required. This version of the core provides access to the maximum of 16 cores without the need for manually instantiating `uj_jtag`. Each core has a unique IR Code (from 0x55 to 0x64) that will provide access to the specific core matching the ID code.

**Figure 1-4. Multiple Processors in a Single Device Single Device**



To use CoreJTAGDebug across multiple devices, one of the devices needs to become the master. This device contains the CoreJTAGDebug core. Each processor is then connected as follows:

**Figure 1-5. Multiple Processors Across Two Devices**



To debug a core on another board, the JTAG signals from CoreJTAGDebug are promoted to top-level pins in the SmartDesign. These are then connected to the JTAG signals directly on the processor.

**Note:** A CoreJTAGDebug, in the second board design, is optional. Note that the UJ\_JTAG macro and the FlashPro header are unused in the second board design.

To select a processor for debugging in SoftConsole, click the debug configurations, and then click the Debugger tab.

The command, shown in the following image, is executed.

**Figure 1-6. Debugger Configuration UJ\_JTAG\_IRCODE**



The UJ\_JTAG\_IRCODE can be changed depending on which processor you are debugging. For example: to debug a processor in Device 0, the UJ\_JTAG\_IRCODE can be set to 0x55 or 0x56.

## Through GPIO

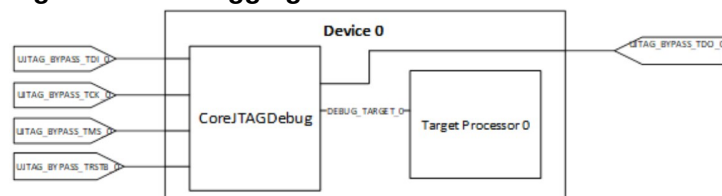
To debug over GPIO, the parameter UJTAG\_BYPASS is selected. One and four cores can be debugged over GPIO headers or pins. To run a debug session using GPIOs from SoftConsole v5.3 or higher, the Debug Configuration must be set up as follows:

**Figure 1-7. Debugger Configuration GPIO**



**Note:** If you are debugging over GPIO, you cannot concurrently debug the processor through the FlashPro Header or the Embedded FlashPro5, on the development boards. For example: FlashPro Header or Embedded FlashPro5 are available to facilitate debug using Identify or SmartDebug.

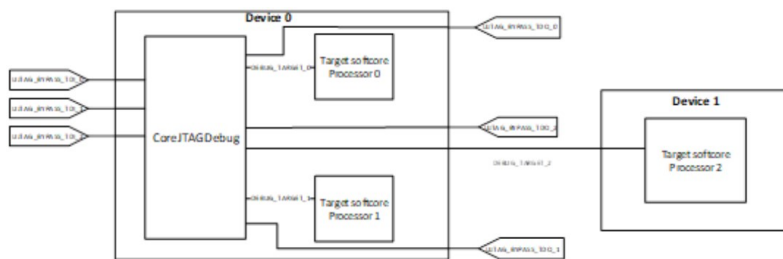
**Figure 1-8. Debugging Over GPIO Pins**



## Device Chaining via GPIO Pins

To support the chaining of multiple devices through GPIO, the UJTAG\_BYPASS parameter needs to be selected. Then the TCK, TMS, and TRSTb signals can be promoted to top-level ports. All target processors have TCK, TMS, and TRSTb. These are not shown below.

**Figure 1-9. Device Chaining Through GPIO Pins**



In a basic JTAG chain, the TDO of a processor connects to the TDI of another processor, and it continues until all processors are chained, in this manner. The TDI of the first processor and the TDO of the last processor connects to the JTAG programmer chaining all the processors. The JTAG signals from the processors are routed to CoreJTAGDebug, where they can be chained. If the chaining across multiple devices is completed, the device with CoreJTAGDebug becomes the master device.

In a GPIO debug scenario, where an IR Code is unallocated to each processor, a modified OpenOCD script is used to select, which device is being debugged. An OpenOCD script is modified to select, which device is debugged. For an Mi-V design, the file is found in the SoftConsole install location, under the openocd/scripts/board/ microsemi-riscv.cfg. For the other processors, the files are found in the same openocd location.

**Note:** The Debug Configuration options also needs to be updated, if the file is renamed

**Figure 1-10. Debug Configuration**



Open username-riscv-gpio-chain.cfg, following is an example of what must be seen:

**Figure 1-11. MIV Configuration File**

```
#-----
# Microsemi RISC-V board
#-----

# FlashPro
source [find interface/microsemi-flashpro.cfg]

# Device
source [find target/microsemi-riscv.cfg]

# Board specific initialization
proc do_board_reset_init () {
}
```

The following settings works for a single device debugging over GPIO. For debugging a chain, additional commands need to be added, so that the devices that are not debugged are put in the bypass mode.

```
jtag newtap IGNORE tap -irlen 5 -expected-id 0 -ignore-version
```

**For two processors in a chain, the following sample command is executed:**

```
# Device
jtag newtap IGNORE tap -irlen 5 -expected-id 0 -ignore-version
source [find target/microsemi-riscv.cfg]
```

This allows debugging of Target software Processor 1 by putting Target software Processor 0 into the bypass mode. To debug the Target software Processor 0, the following command is used:

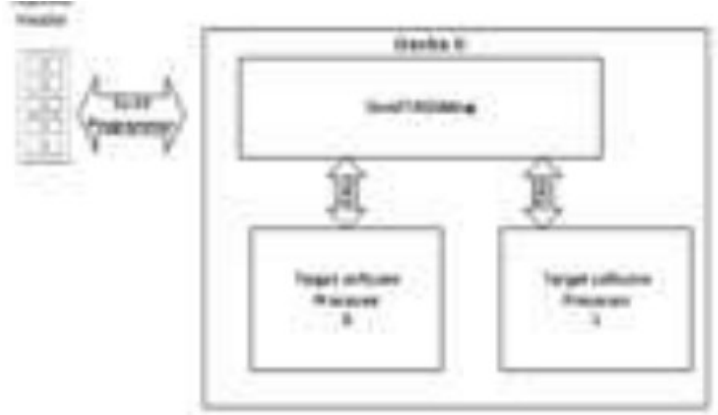
```
# Device
source [find target/microsemi-riscv.cfg]
jtag newtap IGNORE tap -irlen 5 -expected-id 0 -ignore-version
```

**Note:** The only difference between these two configurations is that the source, which is calling the Microsemi RISC-V configuration file (microsemi-riscv.cfg) either comes first, when debugging Target software Processor 0, or

second, when debugging Target Softcore Processor 1. For more than two devices in the chain, additional jtag newtaps is added. For example, if there are three processors in a chain, then the following command is used:

```
# Device
source [find target/microsemi-riscv.cfg]
jtag newtap IGNORE tap -irlen 5 -expected-id 0 -ignore-version
jtag newtap IGNORE tap -irlen 5 -expected-id 0 -ignore-version
```

**Figure 1-12. Example Debug System**



**Interface**

The following sections discuss interface related information.

**Configuration Parameters**

The configuration options for CoreJTAGDebug are described in the following table. If a configuration other than the default is required, use the Configuration dialog box in SmartDesign to select the appropriate values for the configurable options.

**Table 2-1. CoreJTAGDebug Configuration Options**



Name	Valid Range	Default	Description
NUM_DEBUG_TGTS	1-16	1	The number of available debug targets through FlashPro (UJTAG_DEBUG = 0) is 1-16. The number of available debug targets through GPIO (UJTAG_DEBUG = 1) is 1-4.
IR_CODE_TGT_x	0X55-0X64	0X55	JTAG IR Code, one per debug target. The value specified must be unique to this debug target. The tunnel controller associated with this debug target in interface only drives TDO and drives the target debug interface, when the contents of the IR register matches this IR code.
TGT_ACTIVE_HIGH_RESET_x	0-1	0	0: TGT_TRSTN_x output is connected to a global form of the active-low URSTB output of the UJTAG macro. 1: TGT_TRST output is internally connected to a global inverted form of the active-low URSTB output of the UJTAG macro. An extra global routing resource is consumed if this parameter is set to 1 for any debug target.
UJTAG_BYPASS	0-1	0	0: GPIO Debug is disabled, Debug is available through the FlashPro Header or Embedded FlashPro5. 1: GPIO Debug is enabled, Debug is available through a user selected GPIO pins on the board. <b>Note:</b> When the Debugging is done through GPIO, the following debug command is executed in the SoftConsole debug options: “—command “set FPGA_TAP N”“.
UJTAG_SEC_EN	0-1	0	0: UJTAG macro is selected if UJTAG_BYPASS = 0. 1: UJTAG_SEC macro is selected if UJTAG_BYPASS = 1. <b>Note:</b> This parameter only applies to PolarFire. That is, FAMILY = 26.

### Signal Descriptions

The following table lists the signal descriptions for CoreJTAGDebug.

Table 2-2. CoreJTAGDebug I/O Signals

Name	Valid Range	Default	Description
NUM_DEBUG_TGTS	1-16	1	The number of available debug targets through FlashPro (UJTAG_DEBUG = 0) is 1-16. The number of available debug targets through GPIO (UJTAG_DEBUG = 1) is 1-4.
IR_CODE_TGT_x	0X55-0X64	0X55	JTAG IR Code, one per debug target. The value specified must be unique to this debug target. The tunnel controller associated with this debug target in interface only drives TDO and drives the target debug interface, when the contents of the IR register matches this IR code.
TGT_ACTIVE_HIGH_RESET_x	0-1	0	0: TGT_TRSTN_x output is connected to a global form of the active-low URSTB output of the UJTAG macro. 1: TGT_TRST output is internally connected to a global inverted form of the active-low URSTB output of the UJTAG macro. An extra global routing resource is consumed if this parameter is set to 1 for any debug target.
UJTAG_BYPASS	0-1	0	0: GPIO Debug is disabled, Debug is available through the FlashPro Header or Embedded FlashPro. 1: GPIO Debug is enabled, Debug is available through a user selected GPIO pins on the board. <b>Note:</b> When the Debugging is done through GPIO, the following debug command is executed in the SoftConsole debug options: “—command “set FPGA_TAP N”“.
UJTAG_SEC_EN	0-1	0	0: UJTAG macro is selected if UJTAG_BYPASS = 0. 1: UJTAG_SEC macro is selected if UJTAG_BYPASS = 1. <b>Note:</b> This parameter only applies to PolarFire. That is, FAMILY = 26.

#### Notes:

- All signals in the JTAG TAP ports list above must be promoted to top-level ports in SmartDesign.
- The SEC Ports are available only when UJTAG\_SEC\_EN is enabled through CoreJTAGDebug's configuration GUI.
- Take a particular care when connecting the EN\_SEC input. If EN\_SEC is promoted to a top-level port (device input pin), you must access the Configure I/O States During JTAG Programming section of Program Design in the Libero flow and ensure that the I/O State (Output Only) for the EN\_SEC port is set to 1.

## Register Map and Descriptions

There are no registers for CoreJTAGDebug.

## Tool Flow

The following sections discuss tool flow related information.

## License

A license is not required to use this IP Core with Libero SoC.

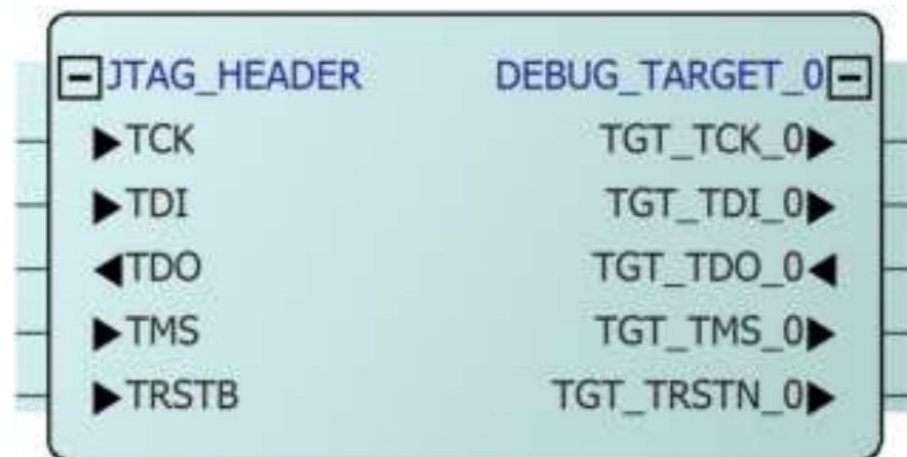
## RTL

Complete RTL code is provided for the core and testbenches, allowing the core to be instantiated with SmartDesign. Simulation, Synthesis, and Layout can be performed within Libero SoC.

## SmartDesign

An example instantiated view of CoreJTAGDebug is shown in the following figure. For more information on using SmartDesign to instantiate and generate cores, refer to the Using DirectCore in Libero® SoC User Guide.

**Figure 4-1. SmartDesign CoreJTAGDebug Instance View using JTAG Header**



**Figure 4-2. SmartDesign CoreJTAGDebug Instance using GPIO Pins**



## Configuring CoreJTAGDebug in SmartDesign

The core is configured using the configuration GUI in SmartDesign. An example of the GUI is shown in the following figure.

**Figure 4-3. Configuring CoreJTAGDebug in SmartDesign**



For PolarFire, UJTAG\_SEC selects the UJTAG\_SEC macro instead of the UJTAG macro when UJTAG\_BYPASS is disabled. It is ignored for all other families.

The Number of Debug Targets is configurable up to 16 debug targets, with UJTAG\_BYPASS disabled and up to 4 debug targets, with UJTAG\_BYPASS enabled.

UJTAG\_BYPASS selects debugging through UJTAG and the FlashPro header, and debugging through GPIO pins. The Target # IR Code is the JTAG IR Code given to the debug target. This must be a unique value within the range specified in **Table 2-1**.

## Simulation Flows

A user testbench is provided with CoreJTAGDebug. To run simulations:

1. Select the user testbench flow within the SmartDesign.
2. Click Save and Generate in the Generate pane. Select the user testbench from the Core Configuration GUI.

When SmartDesign generates the Libero project, it installs the user testbench files. To run the user testbench:

1. Set the design root to the CoreJTAGDebug instantiation in the Libero design hierarchy pane.
2. Click Verify Pre-Synthesized Design > Simulate in the Libero Design Flow window. This starts ModelSim and automatically runs the simulation.

## Synthesis in Libero

### To run Synthesis:

1. Click the Synthesize icon in the Libero SoC Design Flow window to synthesize the core. Alternatively, right-click the Synthesize option in the Design Flow window, and select Open Interactively. The Synthesis window displays the Synplify® project.
2. Click the Run icon.

**Note:** For RTG4, there is an event transient (SET) mitigated warning, which can be ignored as this IP is only used for development purposes and is not going to be used in a radiation environment.

**Place-and-Route in Libero**

Once Synthesis is completed, click the Place and Route icon in Libero SoC to start the placement process.

**Device Programming**

If the UJAG\_SEC feature is used and EN\_SEC is promoted to a top level port (device input pin), you must access the Configure I/O States During JTAG Programming section of Program Design in the Libero flow and ensure that the I/O State (Output Only) for the EN\_SEC port is set to 1.

This configuration is necessary to maintain access to the JTAG port for device reprogramming, because the defined Boundary Scan Register (BSR) value overrides any external logic level on EN\_SEC during reprogramming.

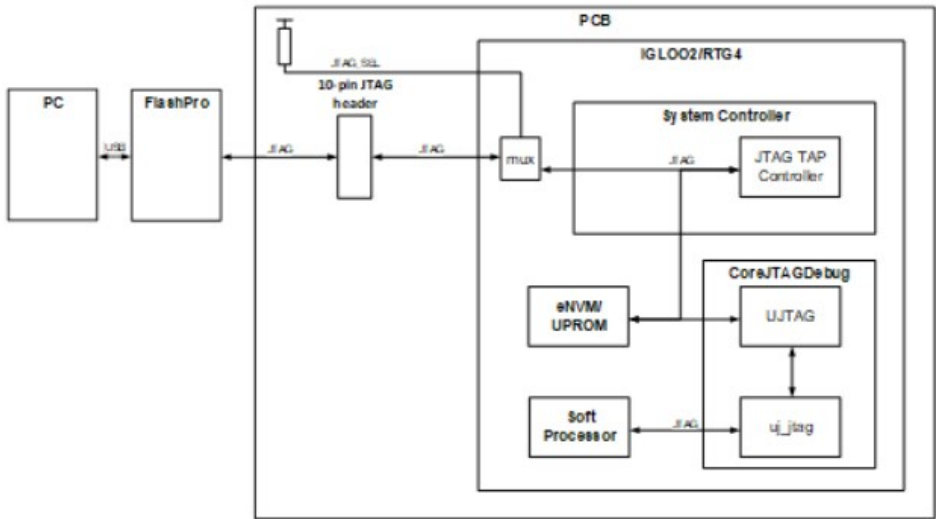
**System Integration**

The following sections discuss the system integration related information.

**System Level Design for IGLOO2/RTG4**

The following figure shows the design requirements to perform JTAG debugging of a softcore processor, located in the fabric from SoftConsole to the JTAG interface for IGLOO2 and RTG4 devices.

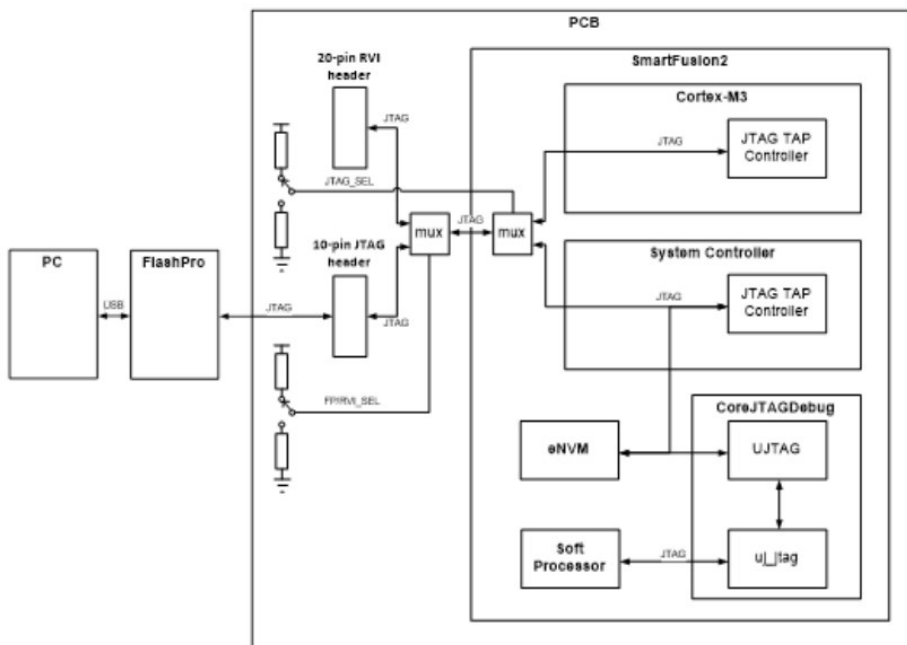
**Figure 5-1. RTG4/IGLOO2 JTAG Debug Design**



**System Level Design for SmartFusion2**

The following figure shows the design requirements to perform JTAG debugging of a softcore processor, located in fabric from SoftConsole to the JTAG interface for SmartFusion2 devices.

**Figure 5-2. SmartFusion2 JTAG Debug Design**

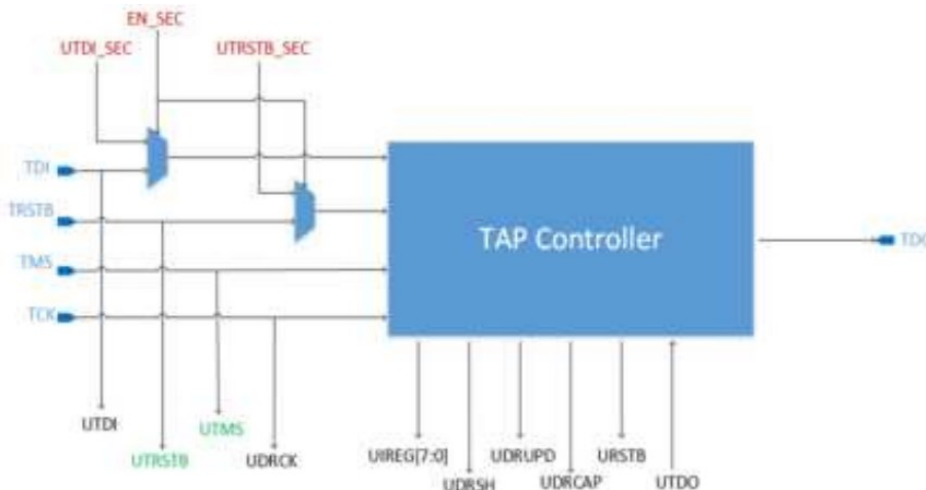


## UJTAG\_SEC

For the PolarFire family of devices, this release allows the user to choose between UJTAG and UJTAG\_SEC, the UJTAG\_SEC\_EN parameter in the GUI will be used to select which one is desired.

The following figure shows a simple diagram that represents the physical interfaces of UJTAG/UJTAG\_SEC in PolarFire.

**Figure 5-3. PolarFire UJTAG\_SEC Macro**



## Design Constraints

The designs with CoreJTAGDebug require the application to follow the constraints, in the design flow, for allowing timing analysis to be used on the TCK clock domain.

### To add the constraints:

1. If the Enhanced Constraint flow in Libero v11.7 or higher is used, double-click Constraints > Manage Constraints in the DesignFlow window and click the Timing tab.
2. In the Timing tab of the Constraint Manager window, click New to create a new SDC file, and name the file. The

Design constraints include the clock source constraints that can be entered in this blank SDC file.

3. If the Classic Constraint flows in Libero v11.7 or higher is used, right-click Create Constraints > Timing Constraint, in the Design Flow window, and then click Create New Constraint. It creates a new SDC file. The design constraints includes the clock source constraints, which is entered in this blank SDC file.
4. Calculate the TCK period and half period. TCK is set to 6 MHz when debugging is done with FlashPro, and is set to a maximum frequency of 30 MHz when debugging is supported by FlashPro5. After you have completed this step, enter the following constraints in the SDC file:

```
create_clock -name { TCK } \
```

- period TCK\_PERIOD \
- waveform { 0 TCK\_HALF\_PERIOD } \ [ get\_ports { TCK } ] For example, the following constraints is applied for a design that uses a TCK frequency of 6 MHz.

```
create_clock -name { TCK } \
```

- period 166.67 \
- waveform { 0 83.33 } \ [ get\_ports { TCK } ]

5. Associate all the constraints files with the Synthesis, Place-and-Route, and Timing Verification stages in the **Constraint Manager** > Timing tab. This is completed by selecting the related check boxes for the SDC files in which the constraints were entered in

## Revision History

Port Name	Width	Direction	Description
<b>JTAG TAP Ports</b>			
TDI	1	Input	Test Data In. Serial data input from TAP.
TCK	1	Input	Test Clock. Clock source to all sequential elements within CoreJTAGDebug.
TMS	1	Input	Test Mode Select.
TDO	1	Output	Test Data out. Serial data output to TAP.
TRSTB	1	Input	Test Reset. Active low reset input from TAP.
<b>JTAG Target X Ports</b>			
TGT_TDO_x	1	Input	Test data out from debug target x to the TAP. Connect to the target TDO port.
TGT_TCK_x	1	Output	Test Clock output to debug target x. TCK is promoted to a global, low skew net internally within CoreJTAGDebug.
TGT_TRST_x	1	Output	Active-High Test Reset. Only used when TGT_ACTIVE_HIGH_RESET_x =1
TGT_TRSTN_x	1	Output	Active-Low Test Reset. Only used when TGT_ACTIVE_HIGH_RESET_x =0
TGT_TMS_x	1	Output	Test Mode Select output to debug target x.
TGT_TDI_x	1	Output	Test Data In. Serial data input from debug target x.
UJTAG_BYPASS_TCK_x	1	Input	Test Clock input to debug target x from GPIO pin.
UJTAG_BYPASS_TMS_x	1	Input	Test Mode Select to debug target x from GPIO pin.
UJTAG_BYPASS_TDI_x	1	Input	Test Data In, Serial data to debug target x from GPIO pin.
UJTAG_BYPASS_TRSTB_x	1	Input	Test Reset. Reset input to debug target x from GPIO pin.
UJTAG_BYPASS_TDO_x	1	Output	Test Data Out, Serial data from debug target x from GPIO pin.
<b>SEC Ports</b>			
EN_SEC	1	Input	Enables Security. Enables the user design to override the external TDI and TRSTB input to the TAP. <b>Caution:</b> Take particular care when connecting this port. See the note below and Device Programming for more details.
TDI_SEC	1	Input	TDI Security override. Overrides the external TDI input to the TAP when EN_SEC is HIGH.
TRSTB_SEC	1	Input	TRSTB Security override. Overrides the external TRSTB input to the TAP when SEC_EN is HIGH.
UTRSTB	1	Output	Test Reset Monitor
UTMS	1	Output	Test Mode Select Monitor



Microchip provides online support via our website at [www.microchip.com/](http://www.microchip.com/). This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## **Product Change Notification Service**

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to [www.microchip.com/pcn](http://www.microchip.com/pcn) and follow the registration instructions Customer Support **Users of Microchip products can receive assistance through several channels:**

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE) Technical Support Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

**Technical support is available through the website at:** [www.microchip.com/support](http://www.microchip.com/support)

## **Microchip Devices Code Protection Feature**

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods being used in attempts to breach the code protection features of the Microchip devices. We believe that these methods require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Attempts to breach these code protection features, most likely, cannot be accomplished without violating Microchip's intellectual property rights.
- Microchip is willing to work with any customer who is concerned about the integrity of its code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable." Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

**Legal Notice**

Information contained in this publication is provided for the sole purpose of designing with and using Microchip products. Information regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANT ABILITY, AND FITNESS FOR A PARTICULAR PURPOSE OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE. IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL OR CONSEQUENTIAL LOSS, DAMAGE, COST OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION. Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
----------	--------------	--------------	--------

**Corporate Office** 2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: [www.microchip.com/support](http://www.microchip.com/support) Web Address: [www.microchip.com](http://www.microchip.com) Atlanta Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455  
**Austin, TX** Tel: 512-257-3370  
**Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088  
**Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075  
**Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924  
**Detroit** Novi, MI Tel: 248-848-4000  
**Houston, TX** Tel: 281-894-5983  
**Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380  
**Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800  
**Raleigh, NC** Tel: 919-844-7510  
**New York, NY** Tel: 631-435-6000  
**San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270  
**Canada – Toronto** Tel: 905-695-1980 Fax: 905-695-2078

**Australia – Sydney** Tel: 61-2-9868-6733  
**China – Beijing** Tel: 86-10-8569-7000  
**China – Chengdu** Tel: 86-28-8665-5511  
**China – Chongqing** Tel: 86-23-8980-9588  
**China – Dongguan** Tel: 86-769-8702-9880  
**China – Guangzhou** Tel: 86-20-8755-8029  
**China – Hangzhou** Tel: 86-571-8792-8115  
**China – Hong Kong SAR** Tel: 852-2943-5100  
**China – Nanjing** Tel: 86-25-8473-2460  
**China – Qingdao** Tel: 86-532-8502-7355  
**China – Shanghai** Tel: 86-21-3326-8000  
**China – Shenyang** Tel: 86-24-2334-2829  
**China – Shenzhen** Tel: 86-755-8864-2200  
**China – Suzhou** Tel: 86-186-6233-1526  
**China – Wuhan** Tel: 86-27-5980-5300  
**China – Xian** Tel: 86-29-8833-7252  
**China – Xiamen** Tel: 86-592-2388138  
**China – Zhuhai** Tel: 86-756-3210040

**India – Bangalore** Tel: 91-80-3090-4444  
**India – New Delhi** Tel: 91-11-4160-8631  
**India – Pune** Tel: 91-20-4121-0141  
**Japan – Osaka** Tel: 81-6-6152-7160  
**Japan – Tokyo** Tel: 81-3-6880-3770  
**Korea – Daegu** Tel: 82-53-744-4301  
**Korea – Seoul** Tel: 82-2-554-7200  
**Malaysia – Kuala Lumpur** Tel: 60-3-7651-7906  
**Malaysia – Penang** Tel: 60-4-227-8870  
**Philippines – Manila** Tel: 63-2-634-9065  
**Singapore** Tel: 65-6334-8870  
**Taiwan – Hsin Chu** Tel: 886-3-577-8366  
**Taiwan – Kaohsiung** Tel: 886-7-213-7830  
**Taiwan – Taipei** Tel: 886-2-2508-8600  
**Thailand – Bangkok** Tel: 66-2-694-1351  
**Vietnam – Ho Chi Minh** Tel: 84-28-5448-2100

**Austria – Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393  
**Denmark – Copenhagen** Tel: 45-4485-5910 Fax: 45-4485-2829  
**Finland – Espoo** Tel: 358-9-4520-820  
**France – Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79  
**Germany – Garching** Tel: 49-8931-9700  
**Germany – Haan** Tel: 49-2129-3766400  
**Germany – Heilbronn** Tel: 49-7131-72400  
**Germany – Karlsruhe** Tel: 49-721-625370  
**Germany – Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44  
**Germany – Rosenheim** Tel: 49-8031-354-560  
**Israel – Ra'anana** Tel: 972-9-744-7705  
**Italy – Milan** Tel: 39-031-742611 Fax: 39-0331-466781  
**Italy – Padova** Tel: 39-049-7625286  
**Netherlands – Drunen** Tel: 31-416-690399 Fax: 31-416-690340  
**Norway – Trondheim** Tel: 47-72884388  
**Poland – Warsaw** Tel: 48-22-3325737  
**Romania – Bucharest** Tel: 40-21-407-87-50  
**Spain – Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91  
**Sweden – Gothenberg** Tel: 46-31-704-60-40  
**Sweden – Stockholm** Tel: 46-8-5090-4654  
**UK – Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820







**Documents / Resources**



[Microchip Technology CoreJTAGDebug Processors](#) [pdf] User Guide  
CoreJTAGDebug Processors, CoreJTAGDebug, Processors

## References

-  2
-  Empowering Innovation | Microchip Technology
-  Empowering Innovation | Microchip Technology
-  Product Change Notification | Microchip Technology
-  Quality | Microchip Technology
-  Microchip Lightning Support