UG0743 User Guide PolarFire FPGA Debugging





Power Matters."

Microsemi Corporate Headquarters One Enterprise, Aliso Viejo, CA 92656 USA Within the USA: +1 (800) 713-4113 Outside the USA: +1 (949) 380-6100 Fax: +1 (949) 215-4996 Email: sales.support@microsemi.com www.microsemi.com

© 2018 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners. Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.



Contents

1			tory	
	1.1		n 4.0	
	1.2		า 3.0	
	1.3		ח 2.0	
	1.4	Revisior	ווו 1.0	1
2	PolarF	Fire FP	GA Debugging	2
	2.1	SmartD	ebug	2
		2.1.1	Standalone Mode	
		2.1.2	Integrated Mode from the Libero SoC PolarFire Design Flow	
		2.1.3	View Device Status	
		2.1.4	Debug FPGA Array	
		2.1.5		
		2.1.6 2.1.7		
		2.1.7	Signal IntegritysNVM Debug	
	2.2		10de	
	2.2	2.2.1	View Device Status	
		2.2.1	Debug FPGA Array	
		2.2.3	Debug TRANSCEIVER	
	2.3	DDR De	ebug	
	2.4		ч 	
		2.4.1	System Components	
		2.4.2	Libero Design Flow With Identify	
	2.5	ModelSi	im	. 44
3	Debug	gging E	xamples	. 45
4	Board	Desig	n Recommendations For Probes	. 46



Figures

Figure 1	SmartDebug—Standalone Mode	3
Figure 2	SmartDebug–Integrated Mode	
Figure 3	Viewing Device Status	
Figure 4	Viewing Device Status Report	
Figure 5	Debug FPGA Array	
Figure 6	Adding Probe Points	
Figure 7	Reserve Pins for Live Probes	
Figure 8	Assign Live Probes	
Figure 8	Add Memory Block	
Figure 9 Figure 10		
Figure 10 Figure 11	View Memory Blocks 1 Read-Write Memory Blocks 1	
Figure 12	Inserting Probes	
Figure 13	Debug µPROM	
Figure 14	View User Design	
Figure 15	Viewing Direct Address	
Figure 16	Debug Transceiver	
Figure 17	Configuration Report	
Figure 18	SmartBert with PRBS Generator	
Figure 19	SmartBERT in Transceiver	
Figure 20	Viewing the Status of TXPLL, RXPLL Lock to Data, Data Rate, and BER 1	
Figure 21	Viewing Error Counter and Error Injection	
Figure 22	Transceiver Loopback	8
Figure 23	Static Pattern Transmit	9
Figure 24	Viewing the Eye Monitor Diagram 2	0
Figure 25	View Signal Integrity for SmartBERT 2	'1
Figure 26	Viewing Signal Integrity for a Selected Lane	2
Figure 27	Applying the Changes for the Signal Integrity 2	2
Figure 28	Viewing a Confirmation Message	
Figure 29	Viewing the Design Default Constraints	
Figure 30	Exporting the Selected Lane Details	
Figure 31	Exporting All Lane Details	
Figure 32	Importing a Selected Lane Details	
Figure 33	Importing All Lane Details	
Figure 34	Viewing Signal Integrity Features in Demo Mode	
Figure 35	SmartDebug—Debug SNVM	
Figure 36	Viewing the Client View	
Figure 37	Reading the Client View Details	
Figure 38	Viewing the Page Status Report	
Figure 39	Viewing Page View	
Figure 40	Viewing Page View Details	
Figure 40	Launching SmartDebug in Demo Mode	
Figure 41 Figure 42		
Figure 42 Figure 43	Viewing Device Status 3 Assigning Live Probes to Channels 3	
•	Reading or Writing Live Probe Values	
Figure 44		
Figure 45	Memory Blocks in Demo Mode	
Figure 46	Debug TRANSCEIVER—Configuration Report	
Figure 47	Debug TRANSCEIVER—SmartBERT	
Figure 48	Debug TRANSCEIVER—Loopback Modes	
Figure 49	Debug TRANSCEIVER—Static Pattern Transmit	
Figure 50	Debug TRANSCEIVER—Eye Monitor	
Figure 51	Host PC-Identify Interface	
Figure 52	Libero Design Flow With Identify	
Figure 53	ModelSim Window	4



1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the current publication.

1.1 Revision 4.0

This version of the document is the update with respect to Libero[®] SoC PolarFire v2.1 release.

1.2 Revision 3.0

The following is a summary of the changes made in revision 3.0 of this document:

- Added the Demo Mode section (see Demo Mode, page 32).
- Updated the Eye Monitor section as part of the Debug TRANSCEIVER section (see Eye Monitor, page 19).
- Updated the sNVM Debug section for its features (see sNVM Debug, page 27).
- Updated the Signal Integrity sections for its features (see Signal Integrity, page 20).
- Updated the SmartBERT section for its features (see SmartBERT, page 15).

1.3 Revision 2.0

The following was a summary of the changes made in revision 2.0 of this document.

- Added additional information about SmartDebug. For more information, see SmartDebug, page 2.
- Updated the view device status section. For more information, see View Device Status, page 4.
- Updated the debug FPGA array section. For more information, see Debug FPGA Array, page 5.
- Updated the debug μPROM section. For more information, see Debug μPROM, page 12.
- Updated the debug transceiver section. For more information, see Debug TRANSCEIVER, page 14.

1.4 Revision 1.0

Revision 1.0 was the first publication of this document.



2 PolarFire FPGA Debugging

Microsemi PolarFire[®] devices support the following on-chip debug capabilities:

- Built-in dedicated probe interface in the fabric (hard block)
- Embedded logic analyzer (soft block)

Microsemi's SmartDebug tool is integrated in the Libero[®] SoC PolarFire Design Suite. SmartDebug enables hardware debugging using the in-built dedicated signal probe. It offers the following advantages:

- Enables live on-chip debugging
- Avoids extra FPGA resource utilization
- Reduces the FPGA design debug cycles
- Enables read-write capability to logic elements and memory blocks during debug

The Identify debug tool integrated in to Libero SoC PolarFire, enables hardware debugging using the embedded logic analyzer.

In addition to SmartDebug and Identify, external test equipments such as Oscilloscopes and Logic Analyzer are also supported. For functional debugging of the FPGA design, Libero SoC PolarFire also integrates the ModelSim simulator.

SmartDebug, Identify, and ModelSim are described in the following sections. For information about debugging examples, see Debugging Examples, page 45.

2.1 SmartDebug

Design debugging is a critical phase of the FPGA design flow. SmartDebug enables you to debug the design by allowing verification and troubleshooting at the hardware level. It provides access to probe points, non-volatile memory (NVM), fabric and memory, transceivers, and the DDR controller.

SmartDebug uses the JTAG interface to retrieve information from the fabric probe points via the fabric control bus. As SmartDebug does not require any fabric logic or change in the design flow, it runs in two modes–standalone and integrated with Libero SoC PolarFire.



2.1.1 Standalone Mode

SmartDebug can also be installed separately with program and debug installer. This setup provides a lean installation that configures all of the programming and debugging tools in a lab environment for debugging. In this mode, SmartDebug opens as a separate tool. The debug process is invoked through SmartDebug after programming the FPGA with the programming file. When SmartDebug is invoked in standalone mode, create a new project and import the design debug data container file (.ddc file that needs to be exported from Libero SoC PolarFire) to access all of the debug features.

Figure 1 • SmartDebug—Standalone Mode

🕸 SmartDebug		_	×
<u>File V</u> iew <u>H</u> elp			
Device: MPF300T_ES (MPF300T_ES)	Programmer: E200 1RUX6Y (E200 1RUX6Y)		Ŧ
ID code read from device: 2F8131CF			
View Device Status	Debug FPGA Array		
Debug UPROM	Debug SNVM		
	Debug TRANSCEIVER		
Log			8×
Hessages SErrors A Warnings () Info			

Note: In standalone mode, the probe insertion feature is not available for FPGA Array Debug, as this feature needs incremental place and route to connect the user net to the specified I/O.

2.1.2 Integrated Mode from the Libero SoC PolarFire Design Flow

SmartDebug has access to all design files that allow you to debug the device. Certain I/O states during programming, and JTAG clock frequency, however, cannot be changed through SmartDebug. To invoke SmartDebug in the Integrated mode, expand **Debug Design** and double-click **SmartDebug Design**.

Figure 2 • SmartDebug–Integrated Mode

Project File	<u>E</u> dit <u>V</u> iew Design Tools <u>H</u> elp
0	
Design Flow	₽ ×
top	🗆 🕒 📄 🌮
Tool	_
	Configure Programming Options
	🐻 Configure Security
	Program Design
V	Generate Bitstream
	Run PROGRAM Action
	Debug Design
	😔 SmartDebug Design
	Configure Permanent Locks for Production
L	S Configure OTP Security
□ ▶	Handoff Design for Production
V	🛃 Export Bitstream
	😢 Export SPI Flash Image
	• Export Pin Report
	• Export BSDL
	Handoff Design for Debugging Tool has not run yet
	😻 Export SmartDebug Data 🔍 👻
	-
Design Flow	Design Hierarchy Stimulus Hierarchy Catalog Files



The following sections describe SmartDebug features, which include:

- View Device Status, page 4
- Debug FPGA Array, page 5
- Debug µPROM, page 12
- Debug TRANSCEIVER, page 14
- Signal Integrity, page 20
- sNVM Debug, page 27
- DDR Debug, page 40

2.1.3 View Device Status

The View Device Status feature provides the device status report.

To view the device status report, click View Device Status in the SmartDebug window.

Figure 3 • Viewing Device Status

SmartDebug	_	×
<u>F</u> ile <u>V</u> iew <u>H</u> elp		
Device: MPF300T_ES (MPF300T_ES)	Programmer: E2001RUX6Y (E2001RUX6Y)	 Y
ID code read from device: 2F8131CF		
View Device Status	Debug FPGA Array	
Debug UPROM	Debug SNVM	
	Debug TRANSCEIVER	
og		8 ×
🗐 Messages 🔞 Errors 🔒 Warnings 🌒 Info		
Selecting programmer E2001RUX6Y for debug.		_
Selecting programmer E2001K0X01 for debug.		
1		



The device status report is displayed. It is a complete summary of ID Code, device certificate, design information, programming information, digest, and device security information, as shown in the following figure.



Device Statu	us Report				?	\times
Device: MPF3001	T_ES (MPF300T_F	ES) Programmer: E2001EIM59 (E	200 1EIM59)	Save	🖨 Pr	int
Device Status:	IDCode (read fi	rom the device) (HEX):	2F8131CF			-
	Device Certifica	ate Certificate is valid .				
	Design Informa	tion Design Name: Design checksum (HEX): Design Version:	SmartDebug_Top 98FC 0			
	Digest Informat	tion Fabric Digest (HEX):	6ec0b6d5bf06e2c47dc945 de290850ba4e40b6deb42			
		SNVM Digest (HEX):	55b852781b9995a44c939l 24b96f99c8f4fb9a141cfc9			
	Device Security	Settings				
	Programming In	nformation Cycle count:	690			
		*Algorithm Version: * Programmer: * Software Version: * Programming Software: * Programming Interface Protoc		able		
Help					Clos	e

Device certificate—displays that certificate is valid if device certificate is installed on a device. If the device certificate is not installed, a message is displayed, stating that the device certificate needs to be installed to generate the device certificate related information.

2.1.4 Debug FPGA Array

The Debug FPGA array provides an interface to probe the user logic implemented in the logic elements (LEs) of the FPGA using active and live probes, read-write access to the fabric flip-flops, and read-write access to the memories implemented using LSRAMs/ μ RAMs.

Probe insertion allows assignment of the internal signals to the assigned or unassigned pins. These signals can be monitored using the oscilloscope in real-time.



The Debug FPGA array supports the following four features:

- Active Probes, page 6
- Live Probes, page 8
- Memory Blocks, page 10
- Probe Insertion, page 11

Figure 5 • Debug FPGA Array

🕸 SmartDebug	- C	x נ
<u>F</u> ile <u>V</u> iew <u>H</u> elp		
Device: MPF300T_ES (MPF300T_ES)	Programmer: E2001RUX6Y (E2001RUX6Y)	<u>~</u>
ID code read from device: 2F8131CF		
View Device Status	Debug FPGA Array	
Debug UPROM	Debug SNVM	
	Debug TRANSCEIVER	
.og		₽×
🔳 Messages 🛛 Errors 🗼 Warnings 🍈 Info		
Selecting programmer E2001RUX6Y for debug.		
Selecting programmer Exportion of debug.		
1		
		1

2.1.4.1 Active Probes

Active probes enable you to read or change the values of probe points in a design through JTAG.

The value of probe points maybe changed for various reasons, such as:

- To verify that a reset signal is in the active and required state.
- To test a logic function by writing to a probe point.
- To initiate a state machine transition by quickly setting an input value to isolate a control flow problem.

Active probes dynamically and asynchronously read or write to any logic element register bit. The probe points of a design are selected using active probes. Active probes are particularly useful for a quick observation of an internal signal. All of the probe points for the design are displayed in **Hierarchical View** and **Netlist View** in the left pane of the Active Probe window.

- Hierarchical View—available probe points are listed in hierarchical order.
- **Netlist View**—available probe points are listed with the Name and Type, which are physical locations of flip-flops.

To add probe points to a list:

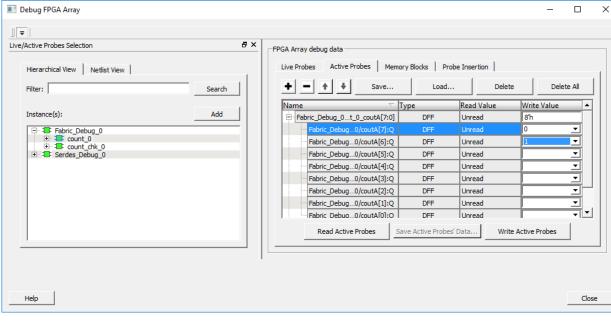
- 1. Select the **Active Probes** tab in the right pane. The probe signals are displayed in the left pane.
- 2. Select the probe points that you want to add from the **Hierarchical View** or **Netlist View** in the left pane.

Adding Probe Points

Figure 6 •



3. Right-click the selected points and click **Add** to add them to the **Active Probes**. You can also add the selected probe points by clicking **Add** in the top-right corner of the left pane. The probes signals can be filtered with the **Filter** option.



The added probe points appear in the active probe data chart and you can read or write multiple probe points at a time.

Note: All of the registered locations of a device are not accessible for writing new probe values due to a silicon limitation. This will be fixed in the next release of the PolarFire silicon.



2.1.4.2 Live Probes

Live probes enable the monitoring of two internal signals at a time in the design without having to rerun place and route.

PolarFire devices have two dedicated live probe channels (for example, pin H6 and G6 of PolarFire MPF300TS device). To use live probes, reserve pins, using **Reserve Pins for Probes** under **Constraints Manager** in the Libero SoC PolarFire. If you do not reserve pins for live probes, the live probe I/O's function as GPIOs and are used for routing nets in the design.

Figure 7 • Reserve Pins for Live Probes

Desired File Edit View Desire Teals Hale

Design Flow		₽×	Reports & X	StartPage	ax s	Top SD 🗗	× Constra	int Manager 🗗 🗙 📗
Top_SD) 🖋	I/O Attributes T					
Tool		•	New	Import	Link	Edit	Check	Help
	Create HDL						Place and Rou	ite
	🔡 Create SmartDesign Testbench		constraint\io\use	ornda [Ta	raat 1			
-	Create HDL Testbench		constraint (io (use	enpuc [la	rgerj			
	🚟 Simulate							
<u> </u>	Constraints							
L	anage Constraints							
	Implement Design							
	🖓 Netlist Viewer							
V	Synthesize		,					
V	Place and Route		I/0 Settin	0.05				
	Verify Post Layout Implementation		1/0 5000	igs				
V	🗠 🔍 Verify Timing		Reserve	e Pins for De	vice Migration			
	Open SmartTime							
	Verify Power		Select the	e devices you	are targetting	for migration	. Pins not bonded	on these devices will be re
	Program and Debug Design		Selected D	Device: MPF	300TS ES - FC	G1152		
	Generate FPGA Array Data				MPF300T_ES			_
	Design and Memory Initialization Configure Hardware				WIPPSUULES			
	Configure Hardware Programming Connectivity and Interface		Target De	vices:				
	Configure Programmer							
	Device I/O States During Programming - JTAG Mode Only							
	Configure Programming Options							
			General —					
	Configure Security				_			
~	Configure Security				and a second			
y =			Reserv	ve Pins for P	robes			

Any probe point from the design can be routed to one of these channels without having to re-run place and route. The probe points assigned to live probe channels can be modified through the SmartDebug live probes **Assign** and **Unassign** options without having to recompile and reprogram the design.

Channel A and Channel B are available in live probes. When an internal signal is selected, it can be assigned to either, Channel A or Channel B.

To assign and unassign probe points to Channel A or Channel B:

- 1. Select the required probe points in the left pane and click **Add** in the top-right corner. The signals are displayed in the right pane.
- Click the signal to be monitored and click Assign to Channel A or Assign to Channel B. The signal is assigned to the selected channel. When the assignment is complete, the probe name appears next to channels. SmartDebug configures Channel A and Channel B I/O to monitor the desired probe points.



3. Click **Unassign Channels** to disconnect the selected internal net from the live probe channel that appears in the bottom-right corner of the live probes window.

Figure 8 • Assign Live Probes

Active Probes Selection	5×	FPGA Array debug data Live Probes Active Probes Memory Blocks Probe Insertion		
Filter:	Search		Delete Delet	e All
Instance(s):	Add	Name	Туре	<u> </u>
	Add	Fabric_Debug_0/count_0_coutA[7]:Fabric_Debug_0/count_0/coutA[7]:	Q DFF	-
Fabric_Debug_0 E Gunt_0		Fabric_Debug_0/count_0_coutA[6]:Fabric_Debug_0/count_0/coutA[6]:	Q DFF	
		Fabric_Debug_0/count_0_coutA[5]:Fabric_Debug_0/count_0/coutA[5]:	D DFF	
		rabiic_bebug_0/count_0_coutA[3]:rabiic_bebug_0/count_0/coutA[3]:		
		Fabric_Debug_0/count_0_coutA[4]:Fabric_Debug_0/count_0/coutA[4]:	Q DFF	-
		Assign to Channel A -> Fabric_Debug_0/count_0_coutA[7]:Fabric_ Assign to Channel B ->		assign



2.1.4.3 Memory Blocks

SmartDebug provides the **Memory Blocks** tab to dynamically and asynchronously read from and write to a selected FPGA fabric SRAM block. Memory blocks are categorized into two views:

- Physical View—shows the actual memory view of the RAM in FPGA.
- Logical View—shows a logical representation of RAM block.

Using the Memory Blocks tab, you can select the required memory block to:

- Read
- Capture a snapshot of the memory
- Modify memory values, and then write the values back to that block.

This feature is useful for checking or setting data buffers used in communication interfaces; debugging complex data dependent errors.

Note: For RAM blocks used in the design through RTL inference, logical representation of the memory blocks may not be available.

To read and write memory blocks:

- 1. Select the Memory Blocks tab in the right pane of the SmartDebug window.
- 2. View the memory blocks in the left pane in the Hierarchical View.
- 3. Select the memory block in the left pane and click **Select** in the top-right corner of the pane.
- 4. Right-click the selected memory block and click **Add**.

Figure 9 • Add Memory Block

Debug FPGA Array																- 0
Memory Blocks Selection 6 × Filter: Search Memory Blocks: Select	FPGA Array deb Live Probes User Design N Data Width:	Active Probes				PSRAM_0/Fabric	_Debug_DPSRA	M_0_DPSRAM_	K_0/Fabric_Deb	oug_DPSRAM_0	_DPSRAM_K_Fa	ibric_Debug_DP	SRAM_0_DPSR/	W_K_0_PF_DP	SRAM_R0C0/IM	IST_RAM1K20_JP
Instance Tree	0	1	2	3	4	5	6	7	8	9	<u> </u>	B	C	D	E	F
нер							Read Block	Save Bloc	k Data	Write Block] 					Clos

The memory blocks are displayed in the right pane.

Figure 10 • View Memory Blocks

ry Blocks Selection 🗗 🗙	FPGA Array	debug data																
ter: Search	Live Prob	es Activ	e Probes M	emory Blocks	Probe Inser	tion												
mory Blocks: Select	User Des Data Wid	ign Memory E th:	llock: Fabric 10-bit		ric_Debug_DPS	RAM_0_DPSRA	M_0/Fabric_D	ebug_DPSRAM	_0_DPSRAM_K	_0/Fabric_Debu	Ig_DPSRAM_0_	DPSRAM_K_Fa	bric_Debug_DF	SRAM_0_DPSF	RAM_K_0_PF_C	OPSRAM_ROCO/	INST_RAM1	K20_1
istance Tree		0	1	2	3	4	5	6	7	8	9	A	В	с	D	E	F	-
Fabric_Debug_0	0000	000	001	002	003	004	005	006	007	008	009	00A	006	00C	00D	00E	00F	
E IPSRAM_1_DPS E IPSRAM_1_DPS	0010	010	011	012	013	014	015	016	017	018	019	01A	01B	01C	01D	D OOE OOF D 01E 01F D 02E 02F D 03E 03F		
INST_R Fabric Debug DPSRAM	0020	020	021	022	023	024	025	026	027	028	029	02A	028	02C	02D	02E	02F	
Fabric_Debug_DPSR Fabric_Debug_D Primitives	0030	030	031	032	033	034	035	036	037	038	039	03A	038	03C	03D	E F ▲ 00E 00F 01E 01F 02E 02F 03E 03F		
1977 - HEL Primitives	0040	040	041	042	043	044	045	046	047	048	049	04A	048	04C	04D	04E	04F	۰.
	,							Read Block	Save Block	Data	Write Block							
																		-

- 5. Click Read Block. The specified memory block is read.
- 6. Enter a hexadecimal value in the memory block locations and click **Write Block**. The memory blocks are replaced with the specified values.



7. Click Save Block Data to save the recent changes.

Figure 11 • Read-Write Memory Blocks

ary Blocks Selection 🗗 🗙	FPGA Array	debug data																
ter: Search	Live Prob	es Activ	e Probes M	emory Blocks	Probe Inser	tion												
mory Blocks: Select	User Des Data Wid	ign Memory B Ith:	llock: Fabric 10-bit		ic_Debug_DPS	RAM_0_DPSRA	M_0/Fabric_De	ebug_DPSRAM_	0_DPSRAM_K_	0/Fabric_Debu	Ig_DPSRAM_0_	DPSRAM_K_Fa	bric_Debug_DF	SRAM_0_DPSF	RAM_K_0_PF_C	PSRAM_R0C0	/INST_RAM1	K20_IP
stance Tree		0	1	2	3	4	5	6	7	8	9	A	В	с	D	E	F	-
Fabric_Debug_0	0000	000	001	002	003	004	005	005	007	008	009	00A	008	00C	00D	00E	00F	
DPSRAM_1_D DPSRAM_1_DPS P IP Primitives	0010	010	011	012	013	014	015	016	017	018	019	01A	01B	01C	0 1D	01E	01F	
INST_R Fabric Debug DPSRAM	0020	020	021	022	023	024	025	026	027	028	029	02A	028	02C	02D	02E	02F	
B Fabric_Debug_DPSR Fabric_Debug_D B 18 Fabric_Debug_D F 18 Primitives	0030	030	031	032	033	034	035	036	037	038	039	03A	038	03C	03D	03E	03F	
to an Primitives	0040	040	041	042	043	044	045	046	047	048	049	04A	048	04C	04D	04E	04F	-
								Read Block	Save Block	Data	Write Block							
																		_

2.1.4.4 **Probe Insertion**

Probe insertion is a post-layout process that enables you to insert probes into the design and to bring signals out to the FPGA package pins. Probe insertion enables selecting internal ports in the design, connecting of those ports to used or unused pins, and then running the layout.

Note: FlashPro programmer must be connected to SmartDebug before inserting probes.

To insert probes into the design:

- 1. Double-click **SmartDebug Design** in the **Design Flow** window. The **SmartDebug Design** window is displayed.
- 2. Select Debug FPGA Array and click the Probe Insertion tab.

Figure 12 • Inserting Probes

Hierarchical View Netlist View		Live Probes Ac	tive Probes Memory Blocks Probe Insertion
Filter:	Search		Delete All
		Net	Driver
Instance(s):	Add	B_DOUT_c[15]	Fabric_Debug_0/DPSRAM_1_0/DPSRAM_1_0/DPSRAM_1_DPSRAM_1_0_PF_DPS
Instance Tree		B_DOUT_c[14]	Fabric_Debug_0/DPSRAM_1_0/DPSRAM_1_0/DPSRAM_1_DPSRAM_1_0_PF_DPS
		B_DOUT_c[13]	Fabric_Debug_0/DPSRAM_1_0/DPSRAM_1_0/DPSRAM_1_DPSRAM_1_0_PF_DPS
DPSRAM_1_0 Fabric_Debug_DPSRAM_0_DPSRAM_0 count_0		B_DOUT_c[12]	Fabric_Debug_0/DPSRAM_1_0/DPSRAM_1_0/DPSRAM_1_DPSRAM_1_0_PF_DPS
		B_DOUT_c[11]	Fabric_Debug_0/DPSRAM_1_0/DPSRAM_1_0/DPSRAM_1_DPSRAM_1_0_PF_DPS
E Serdes_Debug_0		 	
			Insert probe(s) and program the device Run

- 3. Select the **Probe Insertion** tab in the right pane. The probe signals are displayed in the left pane.
- 4. Select probe points from the **Hierarchical View** or **Netlist View** in the left pane.
- 5. Select the probe points and right-click and then click **Add** to add them in the right pane. You can also add the selected probe points by clicking **Add** in the top-right corner in the left pane. The probes signals can be filtered with the **Filter** option.
- 6. Assign a package pin to the probe using the drop-down list in the package pin. You can assign a probe to any unused package pin or spare I/O.



- 7. Click **Run** in the bottom-right corner of the window to run the **Place and Route** in the incremental mode. The selected probe nets are routed to the selected package pins. After incremental place and route, Libero SoC automatically reprograms the device with the added probes.
- 8. To delete the probe, select the probe and click **Delete or Delete All**. Deleting probes from the probe list without performing the **Run** functionality does not remove probes from the design.

2.1.5 Debug µPROM

SmartDebug enables debugging μ PROM and reading its μ PROM contents. The clients added in the design can be debugged using the SmartDebug Debug uPROM feature.

To debug µPROM and its contents, select Debug µPROM in the SmartDebug window.

Figure 13 • Debug µPROM

SmartDebug	_	×
<u>F</u> ile <u>V</u> iew <u>H</u> elp		
Device: MPF300T_ES (MPF300T_ES)	Programmer: E2001RUX6Y (E2001RUX6Y)	¥
ID code read from device: 2F8131CF		
View Device Status	Debug FPGA Array	
Debug UPROM	Debug SNVM	
	Debug TRANSCEIVER	
Log		₽×
🔳 Messages 🛛 😵 Errors 🔺 Warnings 🏾 🕕 Info		
Selecting programmer E2001RUX6Y for debug.		

The μ PROM Debug dialog box displays μ PROM instance used in the design. There are two tabs in the μ PROM debug window:

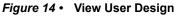
- User Design View—lists all the µPROM clients configured in the design.
- Direct Address View—provides access to µPROM memory. You can read a part of a client or more than one client by specifying the start address and number of 9-bit words.
- **Note:** µPROM clients added in **Design and Memory Initialization** tab and the UIC clients cannot be debugged through SmartDebug.

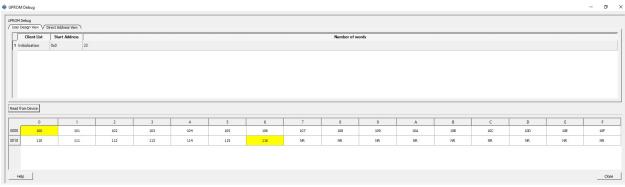
To view µPROM clients in the user design:

- 1. Select the User Design View tab.
- 2. Select the **Client Name** from the client list. The start address and number of words that are reprogrammed are displayed in the window.



3. Click **Read from Device**. The data of the selected client address is displayed. The client address is associated with a start address and number of 9-bit words. Therefore, the table contains as many locations as the number of 9-bit words.

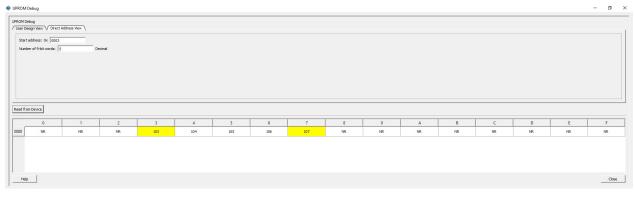




To view direct address of a μ PROM client:

- 1. Select the Direct Address View tab.
- 2. Enter the Start address and Number of 9-bit words.
- Note: The start address must be a hexadecimal value.
 - 3. Click **Read from Device**. The data of the range specified is displayed.







2.1.6 Debug TRANSCEIVER

SmartDebug enables transceiver debugging, which includes checking lane functionality and health for different settings of lane parameters. To access the debug transceiver feature, select **Debug TRANSCEIVER** in the **SmartDebug** window.

Debug Transceiver supports the following features:

- Configuration Report, page 15
- SmartBERT, page 15
- LoopBack Modes, page 18
- Static Pattern Transmit, page 18
- Eye Monitor, page 19

Figure 16 • Debug Transceiver

SmartDebug		×
<u>File V</u> iew <u>H</u> elp		
Device: MPF300T_ES (MPF300T_ES) Programmer: E2001RUX6Y (E2001RUX	6Y)	 *
ID code read from device: 2F8131CF		
View Device Status Debug FPGA Array		
Debug UPROM Debug SNVM		
Debug TRANSCEIVER		
Log		 ₽×
🔳 Messages 🔞 Errors 🗼 Warnings 🇃 Info		
Selecting programmer E2001RUX6Y for debug.		
		//



2.1.6.1 Configuration Report

The **Configuration Report** feature creates a report that shows the physical location, TX and RX PLL lock status, and data width of all enabled transceiver lanes. This report includes the following lane parameters:

- Physical Location—physical location of the transceiver lanes in the system.
- **Tx PMA Ready**—Tx lane of the transceiver is powered up and ready for transactions.
- **Rx PMA Ready**—Rx lane is powered up and ready for transactions.
- **TX PLL**—TX PLL of the transceiver is Locked.
- **Rx PLL**—Rx PLL of the transceiver is Locked.
- Data Width—configured data width of the corresponding lanes in the transceiver.

Figure 17 • Configuration Report

anes	Serdes_Debug_0/bert_xvier_chk_0(SmartBERT IP)	
LANEO		
Physical Location	Q2_LANE0	
Tx PMA Ready		
Rx PMA Ready		
TX PLL		
RX PLL		
Data Width	40 bit	
		Refresh
		Refresh

Note: Red indicates that the lanes are not configured in the current system.

2.1.6.2 SmartBERT

For any transceiver design, PRBS tests from XCVR PMA are available by default. SmartBERT enables you to run diagnostic tests on the transceiver lanes.

2.1.6.2.1 Running with PRBS generator

SmartBERT uses the PRBS generator and checker functionality available in each transceiver lane to determine the bit error rate (BER) of a lane. The various PRBS patterns supported are PRBS7, PRBS9, PRBS15, PRBS23, and PRBS31. Near-end loopback can be performed using one of these PRBS patterns. Bit Error Rate (BER) displays the BER for the PRBS test in progress. The formula for calculating BER is as follows:

BER = (1 + Error Count)/(Data Rate × Seconds)

To run a PRBS pattern:

- 1. Select one of the **Patterns** from the drop-down list.
- 2. Select the **EQ-NearEnd** check box.
- 3. Click **Start** in the bottom-left corner of the window. The loopback cycle is initiated and the result is displayed.



4. Click **Stop** in the bottom-right corner of the window to stop the loopback.

Figure 18 • SmartBert with PRBS Generator

onfiguration Report	Smart BERT Loopba	ack Modes	Static Pattern Transmit	Eye Monitor							
Ianucolver Herarchy ⇒ 20 PF. XCUR,0 → LANIO	Physical Location Q2_LANE1		PF_XCVR_O/LANEO	Pattern	EQ-NearEnd 1	IX PLL RX F	D	Signal Integrity: PF_XCR_0_LANE0 LANE5_TDD_PM TCEmphasis Ampliade #00mr/_wrth_3.5.68 TX Integrations (ofms) [150 TX Transmit Common Mode Adjustment (* 50 Padrity (PM reversal) Normal	- - % of VDDA) - -	LANED_DOD_P/N RX Insertion Loss [5:58] The transceiver data rate is set to 10312_5% pps the current settings will configure this port in CD RX CRE [504]_e15.582_5.183 RX Termination (ohme) [130] RX PLM Seard Connection [AC_COUPED_DWTH_DIX_CAP RX Loss of Signal Detector - Low [FCIE] RX Loss of Signal Detector - High [FCIE]	
Phy Reset			.◀ Start]	_	Stop		Export Import All Lanes		Optimize DFE Design Defaults	Apply

2.1.6.2.2 Running with SmartBERT IP

In addition to the PRBS generator and checker available on transceiver lane, SmartDebug provides the user interface to control the SmartBERT IP instantiated in the design. The transceiver lanes configured with SmartBERT IP show PRBS patterns generated from XCVR PMA as well as the PRBS patterns generated from IP residing in the fabric in the dropdown. Each SmartBERT IP can have 4 lanes configured and each lane can have PRBS7, PRBS9, PRBS23, and PRBS31 patterns configured in the design.

The Debug Transceiver feature shows all lanes that are configured in the design. SmartDebug automatically detects the presence of the CoreSmartBERT in the design. It also identifies the lanes that use SmartBERT IP and distinguish them by appending SmartBERT IP next to the SmartBERT IP instance.

To run SmartBERT in transceiver follow these steps:

- 1. Select the **SmartBERT** tab in the **SmartDebug TRANSCEIVER** window.
- 2. Select the **Pattern** from the drop-down list where the lanes are visible.
- Select the EQ-Nearend check box. When checked, the selected lane gets added to the right hand side where PRBS test can be performed. When unchecked, the selected lane gets removed from the added list.

Figure 19 • SmartBERT in Transceiver

Bebug TRANSCEIVER										8 <u>8</u>		×
Configuration Report	Smart BERT Loopbac	ck Modes	Static Pattern Transmit Eye Monitor									
Transceiver Hierarchy	Physical Location	4		Pattern	EQ-NearEnd	TX PLL	RX PLL	Lock to Data	Cumulative Error Count	4		
Serdes_Debug_(Serdes_Debug_0/bert_xvier_chk_0(SmartBERT IP)/LANE(PRBS23(SmartBERT IP)	Finable	•			0			
			۹				I		Þ			
Phy Reset			Start						Stop			
Help											Clo	se



4. Click **Start** on the lower-left corner of the pane. It enables both transmitter and the receiver for a particular lane and for a particular PRBS pattern. The GUI shows the status of the TXPLL, RXPLL,Lock to Data, Data rate, and the BER.

Figure 20 • Viewing the Status of TXPLL, RXPLL Lock to Data, Data Rate, and BER

sceiver Hierarchy Physical Lo	cation 4		Pattern	EO-NearEnd	TX PLL RX P	LL Lock to Data	Cumulative Error Count	Data Rate (Gbps)	BER	Error Counter	Error Injection		-	
Serdes_Debug_0 Serdes_Debug_0 Serdes_ther LANE0 Q2_LANE0		Serdes_Debug_0/bert_xvier_chk_0(SmartBERT IP)/LANE0	·							Reset	Inject Error			
y Reset		Start										Stop		

Error Injection

When a SmartBERT IP lane is added, the **Error Injection** column is displayed in the right pane. The Error Injection feature is provided to inject an error while running a PRBS pattern. This feature is unavailable if regular lanes are added. Also, this feature is disabled for a SmartBERT IP lane that has a non-configured PRBS pattern selected.

Error Count

Error count is displayed when the lane is added and PRBS pattern is run. Click **Reset** to clear the error count under **Error Counter**.

Figure 21 • Viewing Error Counter and Error Injection

E Debug TRANSCEIVER				- 🗆 ×
Configuration Report Smart BERT Loopback Modes Static Pattern Transmit Eye N	Manitar			1
Transceiver Hierarchy Physical Location	Pattern EQ-	Q-NearEnd TX PLL RX PLL Look to Data Cumulativ	ive Error Count Data Rate (Gbps) BER Error Counter Error Injection	4
B	chk_0(SmartBERT IP)/LANE0 PRBS23(SmartBERT IP) 🗾 🗖	Brable O O O 4	5 4.88e-12 Reset Inject Error	
Phy Reset Start				Stop
Help				Close

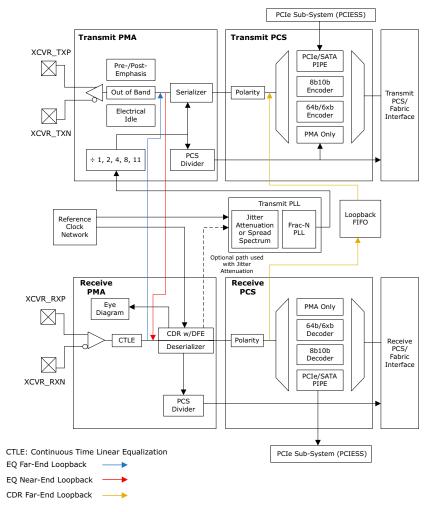


2.1.6.3 LoopBack Modes

Loopback modes help you perform the following types of loopback tests:

- EQ-Near End Looback—Serialized data from PMA is looped from Tx to Rx internally before the transmit buffer. This is called near-end serial loopback. EQ-Near End Looback supports data transmission rates of up to 10.315 Gbps.
- **EQ-Far End Looback**—Serialized data from Rx is looped back to Tx in PMA. This is called far-end serial loopback. EQ-Far End Looback supports data transmission rates of up to 1.25 Gbps.
- CDR-Far End Looback—De-serialized data from PCS Rx channel is looped back to Tx.
- **No-Loop Back**—Data is not looped internally.

Figure 22 • Transceiver Loopback



2.1.6.4 Static Pattern Transmit

Static pattern transmit enables the selection of pattern to be transmitted on a specific transceiver (Tx) lane. The following patterns are supported:

- Fixed pattern
- Max run length pattern
- User pattern—The pattern is defined in the value column. It must be hex numbers and not greater than the configured data width.
- TX-PLL—Indicates lane lock onto TX PLL when a static pattern is transmitted.
- RX-PLL—Indicates RX PLL lock when a static pattern is transmitted.
- Data Width—Displays the data width configured for a transceiver lane.

To view static pattern transmit:



- 1. Select the Static Pattern Transmit tab.
- 2. Select the **Transceiver Hierarchy** in the left pane of the window. The selected lane data is displayed in the right pane.
- 3. Select a pattern from the **Pattern** drop-down list.
- 4. Click Start. The static pattern for the selected lanes is transmitted.
- 5. Click Stop. The static pattern transmission is stopped for the selected lanes.

Figure 23 • Static Pattern Transmit

Configuration Report Smart BERT Loopback Modes Static Pattern Transmit Eye Monitor	
	1
Transceiver Hierarchy Physical Location 41 Pattern Value Mode TX PLI RX PLI DataWidth 41	
C Serdes_Debug_0 C Serdes_Debug_0/bert_xvier_chk_0(SmartBERT IP)/LANE0 Fixed Pattern V LANE0 Q2_LANE0	
PhyReaet Start Stop	Close

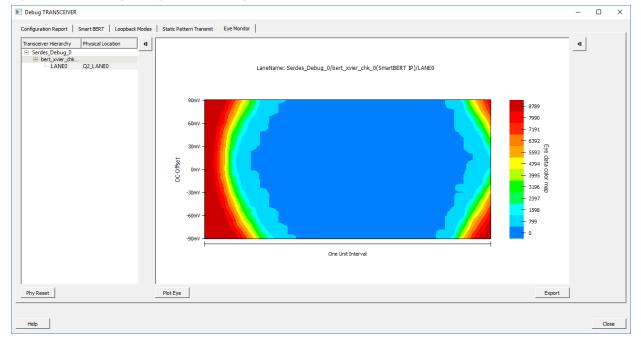
2.1.6.5 Eye Monitor

Eye monitor enables visualizing the eye diagram present within the receiver. This feature plots the receive eye after the CTLE and DFE functions. The diagram representation provides vertical and horizontal measurements of the eye and BER performance measurements. Eye Monitor is supported for data rates above 3.125 Gbps.



Click the **Eye Monitor** tab in the **Debug TRANSCEIVER** window to see the eye monitor representation within the receiver.

Figure 24 • Viewing the Eye Monitor Diagram



2.1.7 Signal Integrity

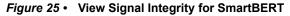
The signal integrity feature in SmartDebug works with Signal Integrity in the I/O Editor, allowing the import and export of .pdc files.

The Signal Integrity pane appears in the following SmartDebug pages:

- SmartBERT, page 15
- LoopBack Modes, page 18
- Static Pattern Transmit, page 18
- Eye Monitor, page 19



When you open Debug Transceiver in SmartDebug and click the **SmartBERT**, **Loopback Modes**, **Static Pattern Transmit**, or **Eye Monitor** tab, all parameters in the **Signal Integrity** pane are disabled. Only the **Export All Lanes** and **Import All Lanes** options are enabled, as shown in the following figure.



ransceiver Hierarchy Physical Location		Pattern I	Signal Integrity	[RX
PF_XCVR_0 PF_XCVR_2			TX TX Emphasis Amplitude	RX RX Insertion Loss
			,	- , _
			TX Impedance (ohms)	N/A N/A
			150	
			TX Transmit Common Mode Adjustment (% of VDE	
			50	No_Peak_+10.7dB
			Polarity (P/N reversal)	RX Termination (ohms)
			Normal	150 👻
				RX P/N Board Connection
				AC_COUPLED_WITH_EXT_CAP -
				RX Loss of Signal Detector - Low
				Off v
				, _
				RX Loss of Signal Detector - High
				Off 💌
1	•	•	Export Import Optimi:	ze DFE Design Defaults Apply
Phy Reset	Charle 1	Chan	Export All Lanes Import All Lanes	
Phy Reset	Start	Stop	Export Air Laries	



When a lane is selected in the **SmartBERT, Loopback Modes, Static Pattern Transmit,** or **Eye Monitor** pages, the corresponding Signal Integrity parameters that are configured in the I/O Editor or changed in SmartDebug, are enabled, as shown in the following figure.

Figure 26 • Viewing Signal Integrity for a Selected Lane

nfiguration Report	Smart BERT Loopbad			1							
ansceiver Hierarchy	Physical Location	•		Pattern	EQ-NearEnd	TX PLL	RX PLL	Lock to Data	Ð	Signal Integrity: PF_XCVR_0/LANE0	
PF_XCVR_0	Q2 LANE0				F	-		-		HPC1_SERDES_2_TX0_P/N	HPC1_SERDES_2_RX0_P/N
PF_XCVR_2	Q2_LAINE0		PF_XCVR_2/LANE0	PRBS7 💌	🗌 Enable	•	•	•		TX Emphasis Amplitude	RX Insertion Loss
LANE0	Q1_LANE0									1000mV_with6dB	
			PF_XCVR_0/LANE0	PRBS7 💌	Enable					TX Impedance (ohms)	The transceiver data rate is set to 6000Mbps for thi
										100 -	The current settings will configure this port in CDR n
										TX Transmit Common Mode Adjustment (% of VDDA) RX CTLE
										50 -	3GHz_+5.5dB_2.1dB
										Polarity (P/N reversal)	RX Termination (ohms)
										Normal	
										_	RX P/N Board Connection
											AC COUPLED WITH EXT CAP
											RX Loss of Signal Detector - Low
											SATA
											,
											RX Loss of Signal Detector - High
											SATA
			•					•		Export Import	Optimize DFE Design Defaults Appl
Phy Reset			Start					Stop	- L	Export All Lanes Import All Lanes	
ily Neber			Stort				-	σωμ			

The selected lane instance name is displayed in the **Signal Integrity** group box, and the **Export, Import,** and **Design Defaults** options are enabled.

You can select options for each parameter from the drop-down for that parameter.

Click Apply to set the selected transceiver instance with the selected options.

Note: The **Apply** button is enabled when you make a selection for any parameter, as shown in the following figure.

Figure 27 • Applying the Changes for the Signal Integrity

Configuration Report Instantion Expected TX RUL REVENT Expected TX RUL REVENT Figuration Report Instantion Instantinstantion Instantion Instantion<	Debug TRANSCEIVE	R											- 1	o ×
Instruction reading Image:	Configuration Report	Smart BERT Loopback Mc	odes	Static Pattern Transmit	Eye Monitor									
P XCR.0 PC XCR.0 PC XCR.0 PC XCR.0 PC XCR.0 PC XCR.2	Transceiver Hierarchy	Physical Location	a []		Pattern	EQ-NearEnd	TX PU	RXPU	Lock to Data	n [Signal Integrity: PF_XCVR_0/LANE0			
PryRest Start Stop PopriAl Lanes PopriAla	E PF XCVR 0											HPC1_SERDES_2_RX0_P/N		
PLANED Q1_LANED PE_XCVR_OLANED PEBS7 Enable The data PE_XCVR_OLANED PEBS7 Enable PEDVECT PEDVECT </td <td>PE V LANEO</td> <td>Q2_LANE0</td> <td></td> <td>PF_XCVR_2/LANE0</td> <td>PRBS7 •</td> <td>Enable</td> <td>•</td> <td>•</td> <td>•</td> <td></td> <td></td> <td></td> <td></td> <td></td>	PE V LANEO	Q2_LANE0		PF_XCVR_2/LANE0	PRBS7 •	Enable	•	•	•					
100 The current settings will configure this port in CCR mode 100 The The manife Common Mode Adjustment (% of VDDA) 101 RX CTLE 102 Polarity (PM reversal) 100 RX Thermark Common Mode Adjustment (% of VDDA) 101 Polarity (PM reversal) 100 RX Thermark Common Mode Adjustment (% of VDDA) 101 Polarity (PM reversal) 100 RX Thermark Common Mode Adjustment (% of VDDA) <	LANE0	Q1_LANE0									-	,		
TX Transit Common Mode Adjustment (% of VDA) RX CTLE S0 RX Transit Common Mode Adjustment (% of VDA) Polarity (P) reversal) RX Termination (Aming) Tommal RX Termination (Aming) Stat Stat				PF_XCVR_0/LANE0	PRBS7 -	Enable								
S0 S12++5.58/2.188 Polenty (PM reversel) RX Temission (phms) Normal ID0 Normal RX Consistion (phms) RX Consistion (phms) RX Consistion (phms) Site Sport Import All Lanes Import All Lanes											-		portinico	Childre
Polarity (P) reversal) RX Termination (phmg) Normal RX Termination (phmg) RX Termination (phmg) RX Termination														
Phy Rest Start Stop Export Al Lanes Import Al Lanes Optimize DFE Design Defaults Apply											-	,		-
Phy Reset Start Stop Export Al Lanes Proof Al Lanes														
Phy Reset Start Stop Export. Al Lanes Import. Al Lanes												P		
Phy Reset Stat Stop Export Al Lanes Proport Al Lanes Apply Telestor														
Ply Reset Start Stop Export Al Lanes Import Al Lanes												RX Loss of Signal Detector - Low		_
Phy Reset Start Stop Export Al Lanes Import Al Lanes Optimize DFE Design Defaults Apply												SATA		•
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
Phy Reset Start Stop Export Al Lanes Import Al Lanes												SATA		•
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
Phy Reset Start Stop Export Al Lanes Import Al Lanes														
				•					•		Export Import	Optimize DFE Design Defaults	A	pply
Heb Cose	Phy Reset			Start					Stop		Export All Lanes Import All Lanes			
Help								-						
Hep														
	Help												_	Close



If you change parameter options and click another lane, move to another tab, or click **Import**, **Import All**, or **Design Defaults** without applying the changes, an alert message is displayed.

Figure 28 • Viewing a Confirmation Message

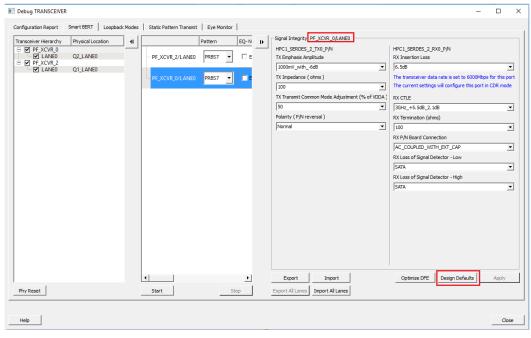
nsevver Herarchy Physical Location 4 □ PF XCVR 0 □ PF XCVR 2	Pattern Yalue D TX TX TX TX Emphasis Anplitude [100m/_with_0.08 TX Transceiver Deburg Y Some Signal Integrity options are modified. Discard Apply	RX RX Insertion Loss RX Insertion Loss N/A N/A N/A RX Entransition (and the provided of the pro
ny Reset St		imize DFE Design Defaults Apply

Click Apply to apply or click Discard to discard the changes.

2.1.7.1 Design Defaults

Click **Design Defaults** to load the Signal Integrity parameter options for the selected lane instance.

Figure 29 • Viewing the Design Default Constraints



Design Default parameter options are applied to the device and updated in Modified Constraints.



Note: Modified Constraints is a list of I/O constraints set on the TXP and RXP lane ports. For a selected lane, this set is created in the SmartDebug session and is updated when a Signal Integrity parameter option is modified and applied or an external PDC file is imported.

2.1.7.2 Export

Click **Export** to export the current selected parameter options along with other physical information of the selected lane instance to an external PDC file.

Figure 30 • Exporting the Selected Lane Details

Debug TRANSCEIVER			- 🗆 ×
Configuration Report Smart BERT Loopback Mon	des Static Pattern Transmit Eye Monitor		
Transceiver Hierarchy Physical Location		Signal Integrity: PF_XCVR_0/LANE0	
B PF_XCVR_0		HPC1_SERDES_2_TX0_P/N	HPC1_SERDES_2_RX0_P/N
LANE0 Q2_LANE0	PF_XCVR_2/LANE0 PRBS7 V 🗆 E	TX Emphasis Amplitude	RX Insertion Loss
PF_XCVR_2 LANE0 Q1_LANE0		1000mV_with6dB	6.5dB
	PF_XCVR_0/LANE0 PRBS7 V E	TX Impedance (ohms)	The transceiver data rate is set to 6000Mbps for this port
		100 💌	The current settings will configure this port in CDR mode
		TX Transmit Common Mode Adjustment (% of VDDA)	RX CTLE
		50 💌	3GHz_+5.5dB_2.1dB
		Polarity (P/N reversal)	RX Termination (ohms)
		Normal	100
			RX P/N Board Connection
			AC_COUPLED_WITH_EXT_CAP
			RX Loss of Signal Detector - Low
			SATA
			RX Loss of Signal Detector - High
			SATA
		Export Import	Optimize DFE Design Defaults Apply
1			oponice of a bengin beroard (1997)
Phy Reset	Start Stop	Export All Lanes Import All Lanes	
Help			Close

The exported content is in the form of two set_io commands, one for the TXP port and one for the RXP port of the selected lane instance.



2.1.7.3 Export All Lanes

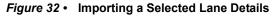
Click **Export All Lanes** to export the current selected parameter options and other physical information for all lane instances in the design to an external PDC file.

Figure 31 • Exporting All Lane Details

Debug TRANSCEIVER			– – ×
Configuration Report Smart BERT Loopback Modes Sta	atic Pattern Transmit Eye Monitor		
Configuration Report Smart BERT Loopback Modes Stat Transcever Herschy Physical Location 4 I: PF XCVR 2 E: PF XCVR 2 LANEO IQ1_LANEO	atic Pattern Transmit Eye Monitor	Signal Integrity: PF_XCIR_2/LANED HPC2_SERDES_1_TX0_P/N TX Emphase Annohude [J000mV_with6d8 _ TX Impedance (ohms) J00 _ TX Transmit Common Mode Adjustment (% of VDOA) 50 _ Polarity (P/N reversal) Normal _	HPC2_SERDES_1_RX0_P/N RX Insertion Loss 25.088 ✓ The transceiver data rate is set to 6000Mpps for this port The current settings will configure this port in CDR mode RX CTLE 394z_5.448_12.0.48 ✓ RX Termination (ohms) 100 ✓ RX P/N Board Connection RX Loss of Signal Detector - Low [SATA ✓ RX Loss of Signal Detector - High [SATA ✓
Phy Reset Piot	st Eye Export	Export Import Export Al Lanes	Optimize DFE Design Defaults Apply
PD	LADOIT	Lapor Chill Conco	
Help			Close

2.1.7.4 Import

Click **Import** to import Signal Integrity parameter options and other physical information for the selected lane from an external PDC file.



Transceiver Hierarchy Physical Location U PF. XCVR.0 PF.XCVR.2 LANE0 Q1 LANE0		D	TX Impedance (ohms) 100 TX Transmit Common Mode Adjustment (% of VDC 50 Polarity (P/N reversal)	HPC2_SERDES_I_RX0_P/N RX1 Insertion Loss The transceiver data rate is set to 6000Mbps for this port The current settings will configure this port in CDR mode RX CTLE RX Termination (ohms) ID0 RX.The Mode Connection RX.Loss of Signal Detector - Low SATA RX Loss of Signal Detector - High
Phy Reset	Plot Eye	Export	Export Import Export All Lanes	Optimize DFE Design Defaults Apply

The Signal Integrity parameter options are applied to the device and updated in Modified Constraints.



2.1.7.5 Import All Lanes

Click **Import All Lanes** to import Signal Integrity parameter options and other physical information for all lanes from an external PDC file.

Figure 33 • Importing All Lane Details

	RX CTLE JGAT_2-5.488 _ 12.008
Export Import	Optimize DFE Design Defaults Apply

The Signal Integrity parameter options are applied to the device and updated in Modified Constraints.



2.1.7.6 Demo Mode

Signal Integrity in **Demo Mode**, lets users experience and understand the debug activities that can be performed with the Signal Integrity feature in SmartDebug. All debug activities except Apply are available in demo mode.

Configuration Report	Smart BERT	Loopback	Modes	Static Pattern Transmit	Eye Monitor	1				
Transceiver Hierarchy	Physical Loc	ation	∎		Pattern	EQ-NearEnd	TX PLL RX PLL	D	Signal Integrity: PF_XCVR_2/LANE0	
PF_XCVR_0			<u> </u>		_				HPC2_SERDES_1_TX0_P/N	HPC2_SERDES_1_RX0_P/N
PF_XCVR_2	Q1_LANE0			PF_XCVR_2/LANE0	PRBS7 -	Enable			TX Emphasis Amplitude	RX Insertion Loss
El canto	Q1_LANCO								1000mV_with6dB	25.0dB
									TX Impedance (ohms)	The transceiver data rate is set to 6000Mbps for this
									100	
									TX Transmit Common Mode Adjustment (% of VDDA	
									50 💌	3GHz5.4dB_12.0dB
									Polarity (P/N reversal)	RX Termination (ohms)
									Normal	
										RX P/N Board Connection
										AC_COUPLED_WITH_EXT_CAP
										RX Loss of Signal Detector - Low
										SATA
										RX Loss of Signal Detector - High
										SATA
				•			Þ		Export Import	Optimize DFE Design Defaults Appl
Phy Reset				Start			Stop		Export All Lanes Import All Lanes	
ing reader										
					* SMA	RTDEBUG	IS RUNNIN	G IN	DEMO MODE *	

Figure 34 • Viewing Signal Integrity Features in Demo Mode

2.1.8 sNVM Debug

sNVM Debug enables reading from and writing to the sNVM during debug. It can be done by reading each page or reading multiple pages based on the authentication. Debug Pass Key is required to carry out SNVM_DEBUG instruction. This feature supports debugging of plain text non-authenticated, authenticated plain text, and cipher authenticated plain text.

Following are the two ways of sNVM debugging which can be performed in SmartDebug:

- When the USK is programmed using USK client, it applies to all the authenticated pages or clients.
 SmartDebug has to read the USK client and store it as the key whenever the content is read
- SmartDebug has to read the USK client and store it as the key whenever the content is read from a client or a page.
- You can override the USK at runtime by changing it for a specific client or page using system services (IP).
 - If you select the option of using sNVM client as ROM, then USK cannot be overridden by system services where the authenticated pages always use the USK client to unlock and read.
 - You can override the type of page using system services. For example, if a page is non-authenticated, then it can be authenticated using system service routines at runtime.

You can also perform the following operations even after the design is programmed into the device:

- Change the content of a page
- Encrypt a page
- · Change the security key of each page configured
- Note: The preceding operations are not possible if the page is used as ROM.

The following sections describe the two views of sNVM Debug:

- Client View, page 28
- Page View, page 31



2.1.8.1 Client View

To view the client view, follow these steps:

1. Click **Debug SNVM** in the **SmartDebug** window. The **sNVM Debug** window is displayed.

Figure 35 • SmartDebug—Debug SNVM

SmartDebug	_	×
<u>Eile V</u> iew <u>H</u> elp		
Device: MPF300T_ES (MPF300T_ES) Programmer: E2001RUX6Y (E2001RUX6Y)		Ŧ
ID code read from device: 2F8131CF		
View Device Status Debug FPGA Array		
Debug UPROM Debug SNVM		
Debug TRANSCEIVER		
Log		Ð ×
Errors 🗼 Warnings 🕦 Info		
Selecting programmer E2001RUX6Y for debug.		



 Click the Client View tab. The client view details are listed. It shows Client Names, Start Page, Number of Bytes, Write Cycles, Page Type, Used as ROM, and USK Status as shown in the following figure.

Figure 36 • Viewing the Client View

	Ŧ	Start Page	Number of Bytes	Write cycles	Page Type	Used as ROM	USK status					
⊿ (1	lient3	3	500		- uge type							
	Page 5	5	28	10	Authenticated	No	USK Client					
	Page 4	4	236	5	Authenticated	No	Enter USK					
	Page 3	3	236	4	Authenticated	No	USK Client					
⊿ (lient2	2	236									
	Page 2	2	236	3	Authenticated PT	No	Enter USK					
⊿ C	lient1	0	500									
	Page 1	1	248	1	Non-Authentic	Yes	-NA-					
	Page 0	0	252	1	Non-Authentic	Yes	-NA-					
	Content Re Page Status	trieved from De	vice:		Page Ni	umber: 1						
	Page Status		vice:			umber: 1						
			3 4	5 6	Page No	umber: 1 9	A B	С	D	E	F	
w All F	Page Status]		5 6			A B	С	D	E	F	
w All F	Page Status]		5 6			A B	С	D	E	F	-
w All F Row1 Row2	Page Status]		5 6			A B	C	D	E	F	-
Row1 Row2 Row3	Page Status]		5 6			A B	С	D	E	F	
	Page Status]		5 6			A B	C	D	E	F	



~ ~

3. Select a client from the list in the **Client View** and click **Read From Device**. The client details are listed in the second pane.

Figure 37 • Reading the Client View Details

efresh Cli	ent Details															
lient List		Start Page	End Page	Number of Bytes	Write cycles	Page Type	Used as ROM	USK status								
	TAGE_2_3_SNVM_CLIENT	0	3				No									
- Pa	ige 3	3		252	33	Plain Text Plain Text		N/A N/A								
- Pa	ige 2 ige 1	2		252	33	Plain Text Plain Text	No	N/A N/A								
Pa	ige 0	0			34	Plain Text	No	N/A								
	TAGE_1_SNVM_CLIENT ge 220	219	220	252	34	Plain Text	No No	N/A								
Pa	ge 219	219			34	Plain Text	No	N/A								
							N	0								
d from De	nite															
	ent Retrieved from Devi															Tue Nov 07 12:27:0
eved Con	tent: Client 'INIT_STAGE	2_3_5NVM_CL	IENT".													
								View All Pa	ge Status							
	0	1	2	3	4	5	6	7	8	9	A	В	с	D	E	F
000	02	00	06	ω	10	00	E6	00	02	02	00	BO	08	00	11	01
010	02	00	06	ω	03	01	00	00	02	02	00	BO	00	00	11	01
020	02	00	06	0	03	01	00	00	02	02	00	BO	00	00	11	00
0030	02	00	06	ω	03	01	00	00	02	02	00	BO	00	10	10	01
0040	02	00	06	ω	03	01	00	00	02	02	00	BO	00	10	10	00
0050	02	00	06	00	2F	00	00	00	02	02	00	BO	4C	10	10	01
0060	02	00	06	ω	06	00	00	00	02	02	00	BO	78	10	10	01
0070	02	00	06	ω	47	03	03	03	02	02	00	BO	88	10	10	00
080	02	00	06	00	03	00	00	00	02	02	00	BO	6C	10	10	00
0090	02	00	06	00	12	00	00	00	02	02	00	BO	10	10	10	00
0A0	02	00	06	0	11	16	A0	00	02	02	00	BO	58	10	10	00
060	02	00	06	0	00	00	OF	00	02	02	00	BO	5C	10	10	00
000	02	00	06	0	1F	1F	00	00	02	02	00	BO	50	10	10	00
0000	02	00	06	ω	88	00	00	00	02	02	00	BO	08	10	10	00
DOED	02	00	06	00	58	00	00	00	02	02	00	BO	0C	10	10	00
DOF0	02	00	06	œ	04	04	00	00	02	02	00	BO	68	10	10	00
0100	02	00	06	ω	47	00	00	00	02	02	00	BO	88	10	10	00
0110	02	00	06	ω	00	00	04	00	02	02	00	BO	0C	10	10	01
0120	02	00	06	0	00	00	00	04	02	02	00	BO	24	10	10	01
0130	02	00	06	œ	AD	00	00	00	02	02	00	BO	34	10	10	01

Note: Only one client can be selected at a time to read the client details. Pages inside the client cannot be selected.

2.1.8.1.1 Refreshing sNVM Status

To view the latest details in the client page, Click **Check SNVM status**. The client pages are refreshed and display the recently updated details.



2.1.8.1.2 Viewing the Page Status

Click **View All Page Status** to view the page status such as Write Cycle Count, Page Type, Use as ROM, and Data Read Status.

Figure 38 • Viewing the Page Status Report

Secured NVM Details			?	×
		Save	📇 Pri	int
Secured Flash Memory Content [SNVM Page sNVM Page #0: Page Status: Write Cycle Count: Page Type: Use as ROM: Data Read Status: sNVM Page #1: Page Status: Write Cycle Count: Page Type: Use as ROM: Data Read Status: sNVM Page #2: Page Status: Write Cycle Count: Page Type: Use as ROM: Data Read Status: sNVM Page #3: Page Status: Write Cycle Count: Page Type: Use as ROM: Data Read Status: sNVM Page #3: Page Status: Write Cycle Count: Page Type: Use as ROM: Data Read Status:	s] 34 Plaintext Off Success 33 Plaintext Off Success 33 Plaintext Off Success 33 Plaintext Off Success			
Help			Close	<u> </u>

2.1.8.2 Page View

Page view displays the client details of the required pages. You can read pages from 0-220 in the page view.

To read data in page view:

- 1. Click the Page View in the sNVM Debug window.
- 2. Enter the page number that you want to read in the **Start Page** and **Number of Bytes** in the respective boxes.
- 3. Click Check Page Status. The page status information is displayed.

Figure 39 • Viewing Page View

Debug ent View V Page	View					
itart page:	0	(Addr) Check	Page Status			
lumber of bytes:	1023	(Addr)				
Page List	Number of Byte	es Page Type	Write Cycles	Used as ROM	USK status	
1 Page 1	252	Non-Authentic	4	Yes	-NA-	
2 Page 2	236	Authenticated	2	No	Use USK client	-
2 Page 2		Authenticated	12	No	IIIse IISK client	1



4. Select pages from the list, and click **Read from Device**. The page details are displayed.

Figure 40 •	Viewing Page View Detai	ils
1 1941 0 40	Thomas ago thom bota	

1 Debug ent View	V Page View																
Start page		Check Pag	. Carton I														
nd Page:		(3 Pages)															
Page Lis		of Bytes Page Type	Write Cycles	Used as ROM	USK Status												
Page		Plain Text	34	No	NA												
Page1 252 Plain Test 33 No N/A Page2 252 Plain Test 33 No N/A																	
	nice															Tue Nov 07 13:29:	
								View All Pag	ge Status								
	0	1	2	3	4	5	6	7	8	9	A	B	C DH	D	E	F	
:0	02	00	06	CD	4F	01	04	00	02	02	00	B0	0C	10	10	01	
0	02	00	06	C0	00	00	00	00	02	02	00	80	14	10	10	01	
0	02	00	06	CO	00	4F	00	00	02	02	00	BO	18	10	10	01	
0	02	00	06	0	01	32	01	00	02	02	00	B0	ED	10	10	01	
0	02	00	06	CD	15	07	3F	FF	02	02	00	80	DO	10	10	01	
0	02	00	06	CD	FC	48	01	01	02	02	00	B0	D4	10	10	01	
0	02	00	06	C0	10	00	80	00	02	02	00	80	D8	10	10	01	
0	02	00	06	CD	01	00	00	04	02	02	00	BO	24	10	10	01	
10	02	00	06	CD	01	00	00	00	02	02	00	B0	2C	10	10	01	
50	02	00	06	CO	D7	00	00	00	02	02	00	80	30	10	10	01	
50	02	00	06	CD	45	00	00	00	02	02	00	B0	34	10	10	01	
0	02	00	06	CD	D0	07	00	00	02	02	00	B0	38	10	10	01	
0	02	00	06	C0	00	00	00	00	02	02	00	B0	œ	10	10	01	
0	02	00	06	CD	00	00	00	11	02	02	00	B0	9C	10	10	01	
40	02	00	06	CD	0D	00	00	00	02	02	00	B0	AD	10	10	01	
30	02	00	06	CD	00	00	00	00	02	02	00	BO	98	10	10	01	
:0	02	00	06	CD	00	75	00	00	02	02	00	B0	90	10	10	01	
00	02	00	06	CO	00	00	00	00	02	02	00	B0	00	00	11	01	
	02	00	06	CD	00	00	00	00	02	02	00	BO	00	10	10	01	
E0										NR				NR			

5. Click View All Page Status. The page status details are displayed.

2.2 Demo Mode

The demo mode allows you to experience SmartDebug functionalities (Active Probe, Live Probe, Memory Blocks, and Debug Transceiver) without having to connect a board to the system running SmartDebug.

Note: The demo mode is for demonstration purposes only, and does not provide the full functionality of the standalone mode.

You cannot switch between demo mode and normal mode while SmartDebug is running.



The following figure is displayed when you open SmartDebug in demo mode.

Figure 41 • Launching SmartDebug in Demo Mode

SmartDebug (DEMO MODE)			- 0
ile Yiew Help			
Device: MPF300T_ES (MPF300T_ES)	<u>_</u>	Programmer: 99861 (usb99861)	
ID code read from device: HARDWARE NOT CON		INING IN DEMO MODE *	
LO COLE FEED FOR DEVICE: NAROWARE NOT COM	ACCIED .		
View Device Sta	itus	Debug FPGA Array	
		Debug SWM	
		Debug TRANSCEIVER	
🗐 Messages 🛛 🚱 Errors 🗼 Warnings 🌗 Info			

Demo mode supports all the following functionalities that Standalone mode supports:

- View Device Status, page 34
- Debug FPGA Array, page 34
- Debug TRANSCEIVER, page 37



2.2.1 View Device Status

The following figure shows an example of the Device Status Report in Demo Mode.

Figure 42 • Viewing Device Status

Device Status Report (DEMO MODE)	?	\times
Device: Hardware not connected Programmer: simulation (simulation) Save	🖨 Pri	nt
* SmartDebug is running in Demo Mode. The device status seen below is an indicati	ive example	▲
Device Status: IDCode (read from the device) (HEX): Hardware not connected		
Device Certificate Certificate is valid .		
Design Information Design Name: sd Design checksum (HEX): F485		
Design Version: 0 Digest Information		
Fabric Digest (HEX): 6b0ee2f257cefa4c49a80942a123 85ff2e979d7eb27d34c165de4c479631	7a9b	
SNVM Digest (HEX): 55b852781b9995a44c939b64e44 24b96f99c8f4fb9a141cfc9842c4b0e3	1ae27	
Device Security Settings		
Help	Clos	e

2.2.2 Debug FPGA Array

In the Debug FPGA Array dialog box, you can view your Live Probes, Active Probes, Memory Blocks, and Insert Probes (Probe Insertion).

Note: Insert Probes (Probe Insertion) is not supported in Demo mode.



2.2.2.1 Live Probes

You can assign and unassign Live Probes to Channel A and Channel B. The following figure shows an example of Live Probes assignment.

Figure 43 • Assigning Live Probes to Channels

Filter Search Delete Delete Intarce(s): Add Image: (b): Add: Image: (b): Image: (b): Image: (b): Image: (b): Image: (b): Image: (b):	Herarchical View	e ×	- FPGA Array debug data Live Probes Active Probes Memory Blocks Probe Insertion	
Handley: Add Image: Horizon (Internet) Image: Horizon (Internet) Image: Horizon (Intern	iter:	Search		Delete Delete All
Image: State and as by the state and the	istance(s):	Add	Name	Туре
B Primitive Implementation Imp	⊡ 📮 serdes_sb_0		serdes_sb_0/CORECONFIGP_0/INIT_DONE_q2:serdes_sb_0/CORECONFIGP_0/INIT_DONE_q2:Q	DFF
FIC 2, ARB, M PREADY FIC 2, ARB, M PREADY FIC 2, ARB, M PSWERR Fic 2, ARB, M PSWERR,	Primitives		serdes_sb_0/CORECONFIGP_0/SDIF_RELEASED_q1:serdes_sb_0/CORECONFIGP_0/SDIF_RELEASED_q1:Q	DFF
INIT_DONE_q1 INIT_DONE_q2 INIT_DONE_q2 SOIP_FILABLE_ SOIP_FILABLE_0 SOIP_FILABLE_0 SOIP_FILABLE_0 SOIP_FILABLE_0 SOIP_FILABLE_0 DFF SOIP_FILABLE_0 DFF SOIP_FILABLE_0 DFF	FIC_2_APB_M_PREADY		serdes_sb_0/serdes_sb_HPMS_TMP_0_FIC_2_APB_MASTER_PSLVERR:serdes_sb_0/CORECONFIGP_0/FIC_2_APB_M_PSLVERR:Q	
B prodata Assign to Channel A -> serdes_b_0/CORECONFIGP_0/INIT_DONE_q2:serdes_b_0/CORECONFIGP_0/INIT_DONE_q2:Q Assign to Channel A -> serdes_b_0/CORECONFIGP_0/INIT_DONE_q2:serdes_b_0/CORECONFIGP_0/INIT_DONE_q2:Q Assign to Channel A -> serdes_b_0/serdes_b_MMS_TMP_0_FIC_2_APB_MASTER_PSUVERR:serdes_b_0/CORECONFIGP_0/FIC_2_APB_M_PSUVERR:Q B B Primitives	INIT_DONE_q2 SOIF_RELEASED_q1 SOIF_RELEASED_q2 SOIF_RELEASED_q2 Ocontrol_reg_1	=11	serdes_sb_0_SDIP0_INIT_AP8_PENABLE:serdes_sb_0/CORECONFIGP_0/SDIF0_PENABLE:Q	DFF
Asign to Channels -> series to Uparcies to	pwdata pwrite soft reset reg			
	CORERESETP_0			_0/FIC_2_APB_M_P5LVERKIQ



 \times

2.2.2.2 **Active Probes**

In demo mode, a temporary probe data file with details of current and previous values of probes is added in the active probes. The write values of probes are updated to this file and the GUI is updated with values from this file when you click Write Probes. The values are read when you click Read Probes. If

📧 Debug FPGA Array (DEMO MODE)						-
Live/Active Probes Selection	₽×	FPGA Array debug data				
Hierarchical View Netlist View		Live Probes Active Pro	bes Memory Block	s Probe Ins	ertion	
Filter:	Search	+ - + +	Save	Load	Delete	
		Name		e	Read Value	Write
Instance(s):	Add	+ serdes_sb_0/serdTEF	R_PRDATA[31:0]	DFF	32'h23	32'h
Sterdes, b, 0 Serdes, b, 0		serdes_sb_0/COR/IN		DFF	0	
⊕ pwdata	-	Read Active	Probes Save A	ctive Probes' Da	ta Write Ac	tive Probe

Fig

2.2.2.3 **Memory Blocks**

Help

A temporary memory data file is created in the designer folder for each type of RAM selected. Different data file is created based on the memory type, for example, separate files are created in the µSRAM, LSRAM, and other RAM types in the designer folder. All memory data of all instances of µSRAM, LSRAM, and other RAM types are written to their respective data files. The default value of all memory locations is shown as 0s, and is updated based on your changes. The demo mode supports physical view as well as logical view of RAM blocks (LSRAM and µRAM).



The following figure shows an example of memory blocks in demo mode.

Figure 45 • Memory Blocks in Demo Mode

mory Blocks Selection	₽×	FPGA Arra	y debug (data —															
lter:	Search	Live Pro	bes	Active Pro	obes	Memory	Blocks	Probe	Insertion	1									
lemory Blocks:	Select	Lines De	sign Mem	an i Black		5RAM_0													
		Data Wi		DI Y BIOCK	18-	_													
Instance Tree		Port Use	ed:		Po	rt A	•	[
DPSRAM_0 DPSRAM_0DPSRAM_R0C0 DPSRAM_R0C0			0	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F	
⊞ : URAM_0		0000	00123	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	H.
			-																
		0010	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	41
		0020	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		0030	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		0040	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		0050	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		0060	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	1
		0070	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		0080	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	-
		0090	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		00A0	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		0080	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		0000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	
		0000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	-
							Re	ad Block	Sa	ave Block	Data	Writ	e Block						
																			_
	1																		

2.2.3 Debug TRANSCEIVER

Transceiver demo mode does not create any temporary data file and is provided to give the user a feel of all the GUI features of Debug Transceiver.



2.2.3.1 Configuration Report

The following figure shows the configuration report of the debug transceiver.

Figure 46 • Debug TRANSCEIVER—Configuration Report

Debug TRANSCEIVER (DEM	10 MODE)	_		\times
Configuration Report Smart	BERT Loopback Modes Static Pattern Transmit I	Eye Mon	itor	1
Lanes	Serdes_Debug_0/bert_xvier_chk_0(SmartBERT IP)			
				-
Physical Location	Q2_LANE0			
Tx PMA Ready	•			
Rx PMA Ready	•			
TX PLL	•			
RX PLL	•			
Data Width	40 bit			
				- 1
				_
			Refresh	
* SMARTI	DEBUG IS RUNNING IN DEMO MOD)E *		
Help			Clos	e

2.2.3.2 SmartBERT

The following figure shows the SmartBERT options of the debug transceiver.

Figure 47 • Debug TRANSCEIVER—SmartBERT

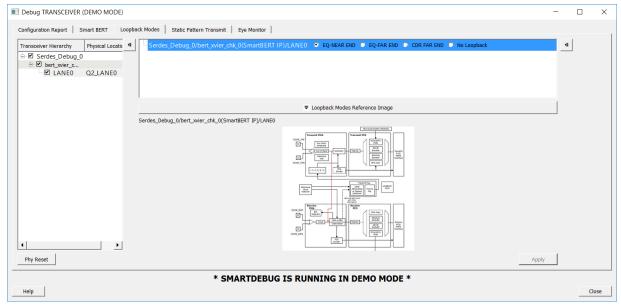
Debug TRANSCEIVER (DEMO MODE)						-		\times
Configuration Report Smart BERT Loopback Modes Static Pattern Transmit Eye Monitor								1
Transceiver Hierarchy Physical Locatic	Pattern E	EQ- NearEnd	TX PLL	RX PLL	Lock to Data	Cumulative	4	
Serdes_Debug_0 Bet_wier_c Serdes_Debug_0/bert_wier_chk_0(SmartBERT IP)/LANE0 Serdes_Debug_0/bert_wier_chk_0(SmartBERT IP)/LANE0	PRBS7(SmartBERT IP)	Enable	٠	٠	•	0		
						•		
Phy Reset Start						Stop		
* SMARTDEBUG IS RU	NNING IN DEMO MOI	DE *						
Help								Close



2.2.3.3 Loopback Modes

The following figure shows the loopback modes of the debug transceiver.

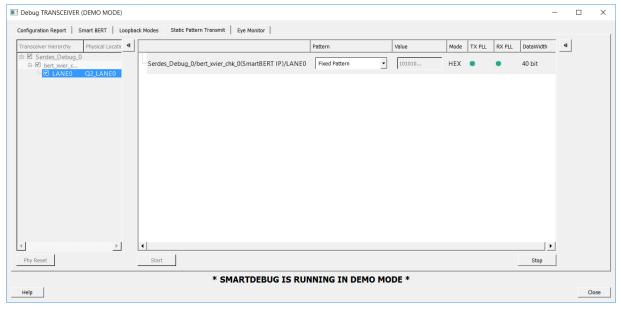
Figure 48 • Debug TRANSCEIVER—Loopback Modes



2.2.3.4 Static Pattern Transmit

The following figure shows the Static Pattern Transmit of the debug transceiver.

Figure 49 • Debug TRANSCEIVER—Static Pattern Transmit

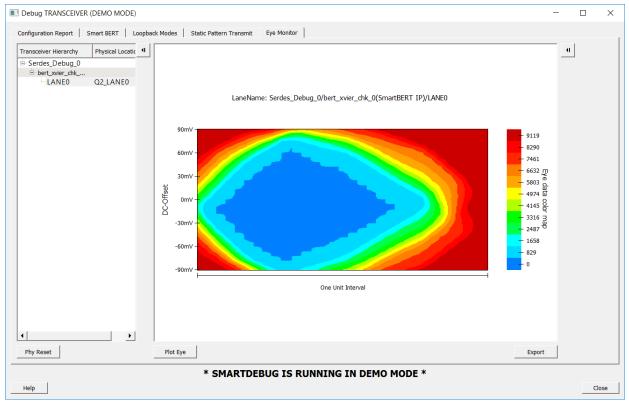




2.2.3.5 Eye Monitor

The following figure shows the Eye Monitor of the debug transceiver.

Figure 50 • Debug TRANSCEIVER—Eye Monitor



2.3 DDR Debug

DDR Debug enables reading the key DDR registers to assess the configuration and status of the DDR controller.

Note: This feature is currently not supported and will be provided in future releases.

2.4 Identify

Identify embeds a logic analyzer. This logic analyzer functionality is embedded in the design using the FPGA fabric and embedded memory blocks. This is triggered when:

- The hardware state machine enters a certain stage
- The critical RTL signals are set to invalid values
- The input pin goes high or low

When triggered, Identify selects and captures complex signal interactions. The trace data of the RTL signal before, after, or around the trigger event is stored in the on-chip, or fabric memory. The trigger event normally reflects a hardware error condition and may be calculated as a combinatorial function of several signals in the RTL.

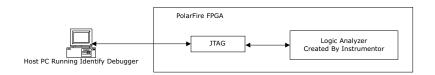
To simplify interface requirements, the logic analyzer is accessed through the standard JTAG port for control and data transfer. The captured data can be displayed on a PC using common viewing software, and typically mirrors a logic simulator waveform output.

Identify adds and configures a logic analyzer to monitor RTL signals within the FPGA design. This tool lets you debug the hardware at the HDL level in the target environment. It is a dual-component system and part of the HDL design flow process. It can be integrated easily with your existing Libero SoC PolarFire design flow. Minor modifications are required for this integration.



The following figure shows the interface between the host PC running the identify Debugger and logic analyzer created in the FPGA fabric.

Figure 51 • Host PC-Identify Interface



Using Identify, the following activities can be performed in the HDL code:

- Activate breakpoints
- Set watch-points
- View captured data related to the original source code or as waveforms
- Set trigger conditions that determine when to capture signal data
 A trigger condition is a set of on-chip signal values or events. When a trigger condition occurs, data
 is transferred from the hardware device to Identify RTL through the JTAG communication cable.

Identify has the following advantages:

- Additional FPGA I/O pins are not required, only standard JTAG signals are used.
- Identify is integrated with the Libero SoC PolarFire design flow.

2.4.1 System Components

The Identify system consists of the following components:

- IICE, page 41
- Identify Instrumentor, page 41
- Identify Debugger, page 42

2.4.1.1 IICE

The Intelligent In-Circuit Emulator (IICE) is a custom block inserted into the design. It is connected to signals in the design by the Identify Instrumentor according to the interface specifications. The IICE block samples internal signals and feeds this information back to the Identify Debugger, where the data is transformed for interpretation at the HDL level.

The IICE block comprises of the Probe block and the Controller block. The probe block samples internal signal data and communicates with the controller block. It contains the sample buffer where signal value data is stored. The probe block also contains the trigger logic that determines when signal data is stored in the sample buffer.

The controller block receives sample data from the probe block and sends to Identify Debugger through the JTAG port.

2.4.1.2 Identify Instrumentor

Identify Instrumentor reads and analyzes the pre-synthesis HDL design and provides detailed information about the signals that can be observed. Based on this analysis, Identify Instrumentor enables you to specify how to control signal observation.

Identify Instrumentor uses HDL design files and user-selection information to create a custom logic analyzer block. It connects the logic analyzer to the appropriate signals in the design.

Identify Instrumentor enables the following:

- Selecting the required design signals at the HDL level
- Creating an on-chip logic analyzer to give access to the Identify Debugger



2.4.1.3 Identify Debugger

Identify Debugger lets you interact with the debug-enabled hardware at the HDL level.

Using Identify Debugger, trigger conditions are set that determine when to capture the signal data. A trigger condition is a set of on-chip signal values or events. When triggered, data is transferred from the hardware device to the Identify Debugger through the JTAG communication cable.

Identify Debugger enables the following:

- Interacting with the on-chip logic analyzer
- Getting information about the required signals in the design

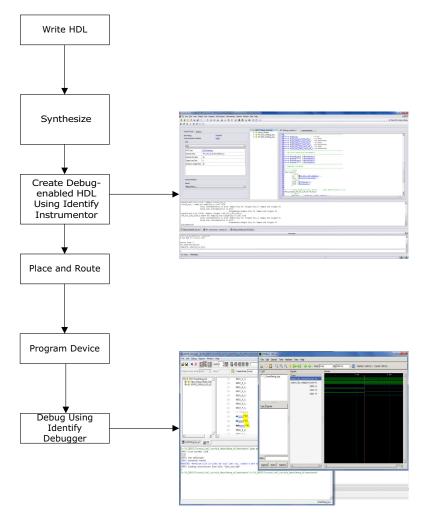
With these components, Identify debugs your design faster, easier, and more efficiently.



2.4.2 Libero Design Flow With Identify

This section describes the Libero SoC PolarFire design flow with Identify. The following figure shows the Libero SoC PolarFire design flow with the Identify system components.

Figure 52 • Libero Design Flow With Identify



The Libero SoC PolarFire design flow with Identify comprises the following steps:

- 1. Create the HDL design.
- 2. Synthesize the design to the target device.
- 3. Use Identify Instrumentor to create a debug-enabled HDL design.
- 4. Place and route the debug-enabled design on the target device.
- 5. Implement the debug-enabled design on the device by programming it.
- 6. Use Identify Debugger to debug the design while it is running on the target device.



4.4.94.4

2.5 **ModelSim**

ModelSim enables debugging of the FPGA design. In ModelSim design simulation, all design components are modeled mathematically as software processes that are executed sequentially to represent the functional, logical, or timing information of the design. A wide-range of stimuli are applied to the design and the expected output is checked against the design requirements. A typical ModelSim

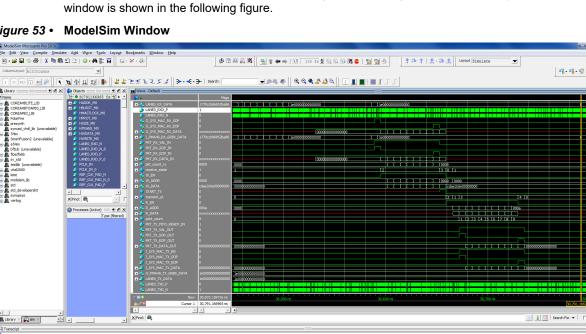


Figure 53 • ModelSim Window

Now: 30.894.933.984 fs Delta: 1 sim:/testbench 10a/top 10a

Simulation tools (like ModelSim) have the following advantages:

- Simulation is performed using software and testbench stimuli to represent the design requirements. Design hardware is not required.
- Simulators with testbenches catch many design errors including:
 - Incorrect specifications ٠
 - Incorrect Interface requirements
 - Function errors
 - Other gross errors detected by stimulus vectors

Simulation is particularly effective when extensive stimulus combinations are used, and when the results are well known. In these cases, simulation can do an exhaustive test of a design.

Simulation tools have the following disadvantages:

- Most designs do not have easy access to extensive test suites and creating them can be time consuming and often impossible for large FPGA designs.
- Execution is slow, when many iterations are involved, and rendering them is time-consuming and expensive during the development process.



3 Debugging Examples

This section explains the uses of different tools for debugging the following peripherals:

Fabric

•

- LSRAM/µSRAM
- Transceiver
- DDR Memories
- Coretex-M1 based embedded system
- Note: More information will be provided in future releases.



4 Board Design Recommendations For Probes

This section describes the board design recommendations for board level debugging and the guidelines for hardware debugging.

Note: More information will be provided in future releases.