

AC482
Application Note
PolarFire FPGA: How to Perform On-Demand Digest
Check



a  MICROCHIP company



a  MICROCHIP company

Microsemi Headquarters

One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113

Outside the USA: +1 (949) 380-6100

Sales: +1 (949) 380-6136

Fax: +1 (949) 215-4996

Email: sales.support@microsemi.com

www.microsemi.com

©2018 Microsemi, a wholly owned subsidiary of Microchip Technology Inc. All rights reserved. Microsemi and the Microsemi logo are registered 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, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), 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. Learn more at www.microsemi.com.

Contents

1	Revision History	1
1.1	Revision 1.0	1
2	How to Perform On-Demand Digest Check	2
2.1	On-Demand Digest Check	2
2.2	Design Requirements	3
2.3	Design Files	3
2.4	Design Description	4
2.5	How to Run the Demo	9
2.5.1	Tera Term Setup	9
2.5.2	Board Setup	9
2.5.3	Program the Device	10
2.5.4	Running the Demo	10

Figures

Figure 1	Design Block Diagram	4
Figure 2	Sequence of Instructions to read DSN or USERCODE	5
Figure 3	Sequence of Instructions to perform Digest Check on FPGA Fabric	6
Figure 4	CoreReset_PF Ports	7
Figure 5	Reset Structure	7
Figure 6	Hardware Implementation of the Design	8
Figure 7	JTAG USERCODE or Silicon Signature Configuration	8
Figure 8	Select Serial Port	9
Figure 9	User Interface Menu	10
Figure 10	Fetch DSN System Service	10
Figure 11	Fetch JTAG USERCODE System Service	10
Figure 12	Fabric Digest Check System Service	11

Tables

Table 1	Resource Requirements	3
Table 2	OPTIONS[15:0]	6
Table 3	Jumper Settings	9

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 1.0

The first publication of this document.

2 How to Perform On-Demand Digest Check

PolarFire devices have a built-in self-test mechanism that can be used (optionally) to check the design integrity and security of the device automatically upon power-up, or on-demand. The contents of all the nonvolatile configuration memory segments, including security keys, security settings, and the FPGA fabric configuration, plus any sNVM memory pages declared as ROM by the user (all the write-protected pages) are tested using the digest check feature. This test provides assurance against both natural and maliciously induced failures.

In the factory and user security segment, each logical page contains an automatically generated digest, calculated dynamically at the time of programming the data to be written. For the FPGA fabric, the digest includes an overall value covering the data to be programmed. In addition, digests are calculated and stored for the sNVM pages marked as ROM. The digests can be verified on-demand by the user, either internally using a system service, or externally using a programming instruction. In addition, the user can automatically run digest checks on each power-up.

An endurance limit specifies how many times a digest of the FPGA fabric can be run. See the [DS0141: PolarFire FPGA Datasheet](#) for more information about the FPGA configuration memory endurance limits. Therefore, depending upon how the system is deployed and used (for example, how often it is powered-up), the on-demand digest check may be more appropriate for testing the integrity of the FPGA fabric.

2.1 On-Demand Digest Check

The on-demand digest check recalculates and compares digests of selected non-volatile memories (FPGA fabric, sNVM, and security segments) with the stored digests. A failure of any digest results in the tamper event being triggered for a user action. The on-demand digest check is invoked by calling digest check design system service.

If the digest check is performed on the FPGA fabric, then the FPGA fabric is automatically placed in suspend state before commencing the digest check operation. Except LSRAMs, all the FPGA fabric logic elements and uSRAMs holds their current state during the suspend state. Upon completion of the fabric digest, the suspend state is automatically exited. Since the LSRAMs does not retain the user data after performing digest check on FPGA fabric. The status of the fabric digest check must be monitored by a state machine implemented in the fabric.

After checking the status of the fabric digest check, the state machine needs to issue a design reset or device reset depending on the design requirements. The device reset reinitializes the device and design memories according to the user configuration. The user design reset does not reinitialize the device or LSRAMs. The user design reset must be implemented to reset the user design. If the design uses LSRAMs initialization at power-up, then the user design needs to issue a device reset otherwise design reset is sufficient to restart the design. Use RESET_DEVICE tamper response signal for device reset.

The FPGA fabric design including LSRAMs content is not affected when the digest check is performed on sNVM or security segments. see [UG0753: PolarFire FPGA Security User Guide](#) for more information on Digest Check Service.

This application note demonstrates how to perform on-demand digest check with a design example.

2.2 Design Requirements

The following table lists the hardware and software required to perform fabric digest check operation:

Table 1 • Resource Requirements

Requirement	Version
Host PC Operating system	Windows 7, 8.1, or 10
Hardware	
PolarFire Evaluation Kit with MPF300TS-1FCG1152 device	Rev D
Software	
Libero SoC PolarFire	v2.3
Tera Term	4.95

2.3 Design Files

Download the design files from the following location:

http://soc.microsemi.com/download/rsc/?f=mpf_ac482_liberosocpolarfirev2p3_df

Note: The provided design example is created for MPF300TS-1FCG1152 device (Non-ES device).

2.4 Design Description

The design example provided with the application note performs the following operations:

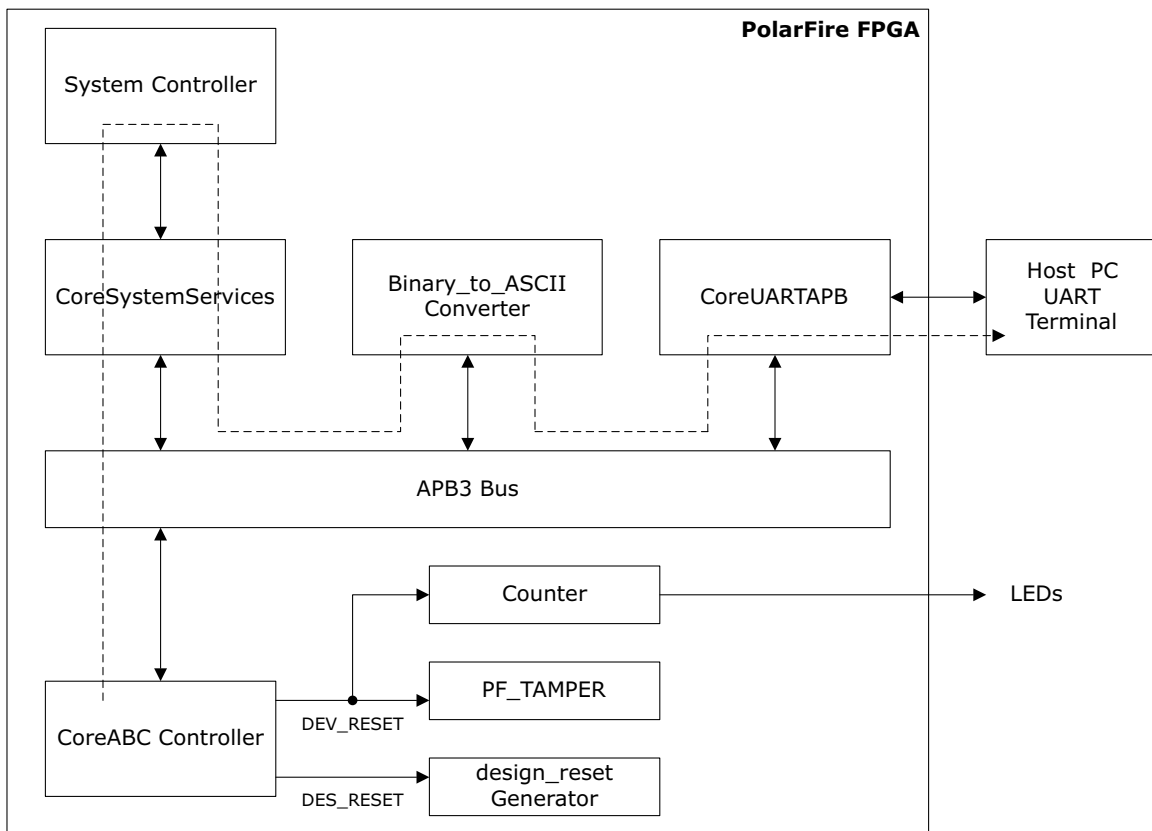
1. Initiates system service request to perform the following system services:
 - Fetch Device Serial Number (DSN)
 - Fetch USERCODE
 - Digest Check on FPGA Fabric
2. Demonstrates how to perform design reset or device reset after fabric digest check
3. Provides user interface through UART terminal

System services are system controller actions initiated from the user design. CoreSysServices IP must be used to initiate system service requests. CoreSysServices IP provides a set of registers, which are accessible through APB interface for executing the system services. On the other hand, the CoreSysServices_PF IP interfaces to the PolarFire System Controller through a system service interface. The System Controller has a 2 KB mailbox RAM for any additional input parameters and any outputs from the service.

A CoreABC processor is used as a controller to perform the above operations. CoreABC (APB Bus Controller) is a simple, configurable, low gate count, programmable state machine/controller primarily targeted towards the implementation of AMBA APB-based designs. For information about the instructions supported by the CoreABC, see *CoreABC handbook*.

The following figure shows the block diagram of the example design. The design is implemented without consuming LSRAMs since the LSRAMs content is not retained after performing fabric digest check.

Figure 1 • Design Block Diagram



The dotted line in the [Figure 1](#), page 4 shows the data flow. The data flow in [Figure 1](#), page 4 is explained as follows:

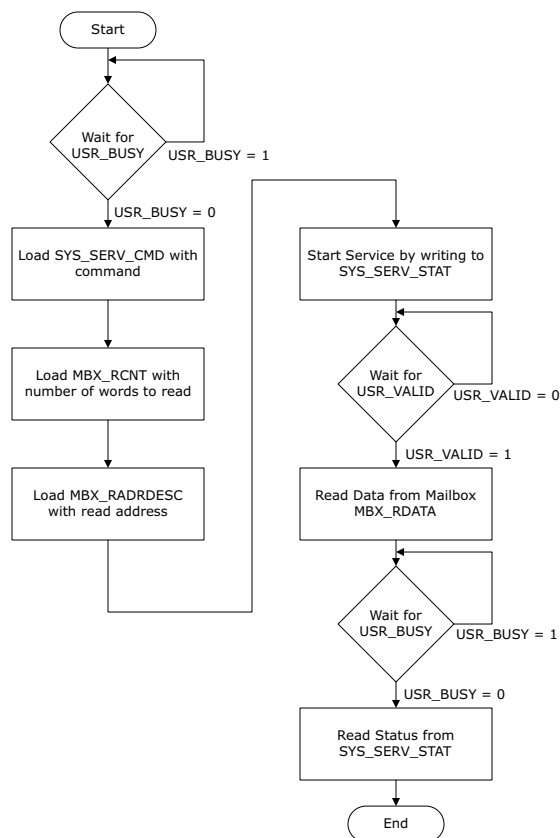
1. CoreABC processor initiates a system service through CoreSystemServices IP.
2. System Controller performs the requested action and writes the result to the System Controller mailbox.
3. CoreABC processor reads the result and converts it into ASCII format using Binary_to_ASCII converter.
4. CoreABC processor prints the result on a host PC UART terminal through CoreUARTAPB controller.
5. After digest check, the CoreABC processor issues a device reset or a design reset based on user input. The PF_TAMPER macro is used to issue a device reset.

The System Services are invoked by writing to appropriate CoreSysServices IP registers with the system service command and any additional information required to perform a system service. Then the CoreSysServices IP initiates necessary transactions on the system service interface. The CoreSysServices IP asserts USR_BUSY signal when the IP is busy in executing the service. Upon completion of a service, a status code is written to the status register and the USR_BUSY signal gets de-asserted. For services requiring the mailbox reads, the signal is de-asserted only when the required number of reads for the service are completed. If a new service request is initiated when the USR_BUSY is in progress, the new request is ignored until the current service completes. For more information about CoreSysServices IP, see *CoreSysServices IP handbook*.

The following flowchart shows the sequence of instructions coded in the CoreABC program to read DSN or USERCODE using system services. The command value and number of words (32 bits) to read (MBX_RCNT) are service dependent:

- For read DSN service, the command is 0x00 and MBX_RCNT is 4
- For read USERCODE service, the command is 0x01 and MBX_RCNT is 1

Figure 2 • Sequence of Instructions to read DSN or USERCODE



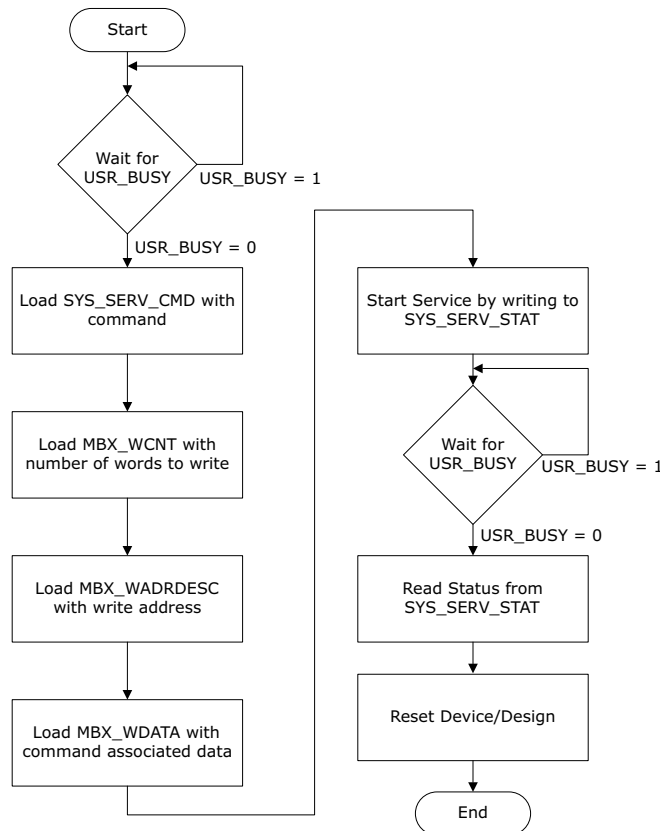
The command value to perform digest check service is 0x47 and this service request requires additional information (OPTIONS[15:0], as listed in the following table) to specify the area for which the digest check must be performed. The additional information is provided through mailbox writes. In this reference design, 0x1 is written to the mailbox (MBX_WDATA) to perform digest check on FPGA fabric. Users can change this parameter to perform digest on other NVM components. The digest check service does not expect any return data other than status of the service.

Table 2 • OPTIONS[15:0]

OPTIONS[i]	Name	Description
0	CHECK FABRIC	Enables fabric digest
1	CC	Enables digest of fabric configuration data
2	SNVM	Enables digest of sNVM pages marked as ROM
3	UL	Enables digest of user security segment
4	UKDIGEST0	Enables digest of user key segment containing SRAMPUF data
5	UKDIGEST1	Enables digest of user key segment containing UEK (User EC key)
6	UKDIGEST2	Enables digest of user key segment containing UPK1
7	UKDIGEST3	Enables digest of user key segment containing UEK1

The following flowchart shows the sequence of instructions coded in the CoreABC program to perform digest check on the FPGA fabric using system service.

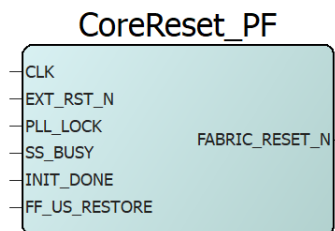
Figure 3 • Sequence of Instructions to perform Digest Check on FPGA Fabric



The FPGA fabric is put into suspend state during fabric digest check calculation as explained in [On-Demand Digest Check](#), page 2. During the suspend state, the clocks are gated off to low, the PLLs are powered down, and all the I/O input buffers are disabled. After digest check, the FPGA fabric automatically comes out of suspend state, and the PLLs and clocks are re-enabled. The PLLs take time to re-acquire lock. Since the PLL lock is used in design reset generation, the PLL lock signal needs to be bypassed until the lock is re-acquired after exiting from suspend state. If not, the design gets reset because of loss of PLL lock.

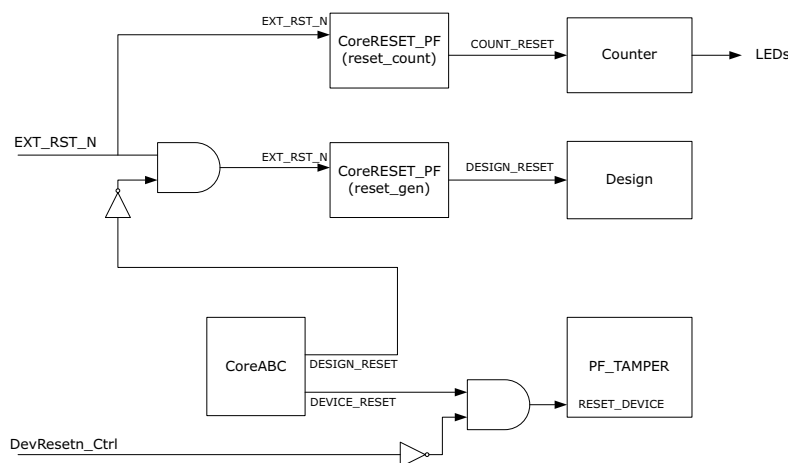
The example design uses CoreReset_PF IP core (as shown in following figure) to serve the above purpose. The CoreReset_PF IP generates a reset, which is asserted asynchronously by one of the multiple potential sources and de-asserted synchronously with a specified clock. The potential sources of reset can be external reset through a GPIO, PLL lock, INIT_DONE from PolarFire Initialization Monitor (PF_INIT_MONITOR). The CoreReset_PF IP uses USR_BUSY signals from CoreSysServices_PF IP to bypass PLL_LOCK and EXT_RST_N at the time of suspend state exit. This ensures that no reset occurs while the PLL is re-acquiring lock after exiting suspend state. The CoreReset_PF IP also uses FF_US_RESTORE from CoreSysServices_PF IP to avoid glitches on asynchronous set or reset signals generated from flip-flops at the time of suspend state exit.

Figure 4 • CoreReset_PF Ports



If LSRAMs are used in the design, then the user design needs to issue a device reset or design reset after completing digest check on the FPGA fabric. Based on the user input, the CoreABC processor asserts reset_device input of PF_TAMPER macro for device reset or asserts EXT_RST_N input of CoreReset_PF IP for design reset. To show differentiation between device reset and design reset, the example design uses two CoreReset_PF IPs: one is used to generate reset to complete design except a counter, which blinks LEDs on board and the other IP is used to reset the counter. The counter does not get reset when design reset is initiated from CoreABC processor. The counter gets reset only when the device reset is issued.

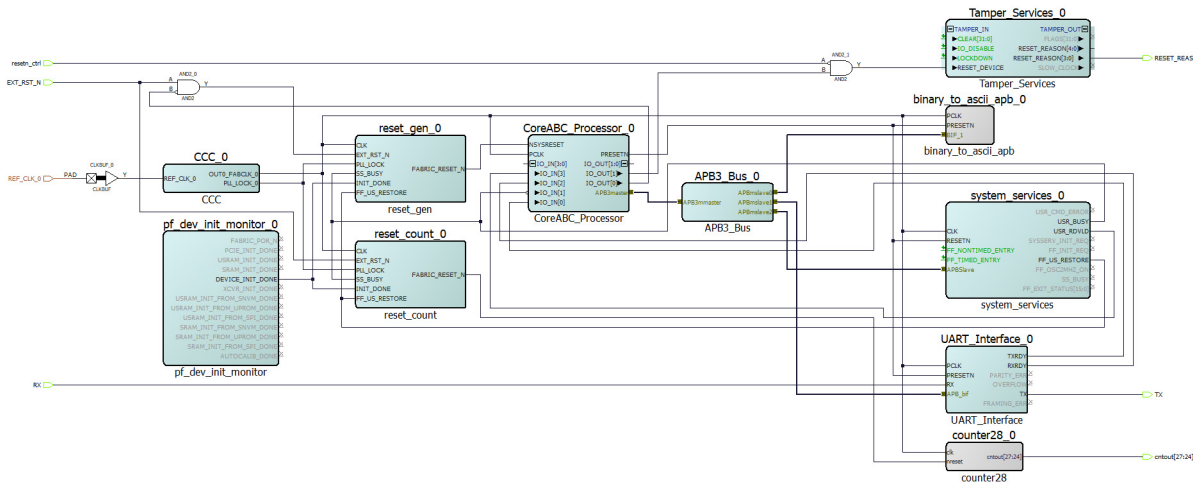
Figure 5 • Reset Structure



Note: RESET_DEVICE input of PF_TAMPER macro should not be promoted to top-level I/Os directly. The design example provides device reset control from an external DIP switch (SW11-DIP1) through DevResetn_Ctrl port. See [Figure 5](#), page 7 for DevResetn_Ctrl port connection. SW11-DIP1 switch position should be set to ON to control the device reset from the user design. Logic '0' is driven on DevResetn_Ctrl port when the DIP1 switch position is set to ON.

The following figure shows the hardware implementation of the design example using Libero.

Figure 6 • Hardware Implementation of the Design

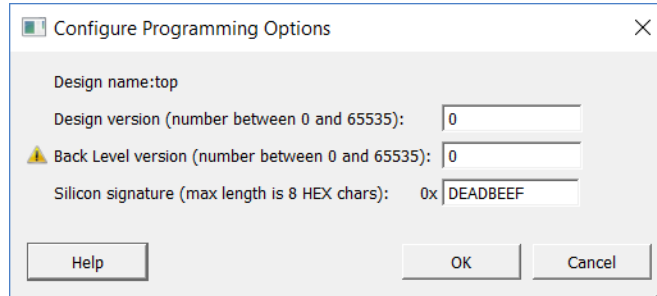


The design example also demonstrates the following system services:

- Fetch Device Serial Number system service
- Fetch JTAG USERCODE system service

The device serial number (DSN) is a 128-bit unique device ID set in the factory. JTAG USERCODE is a 32-bit user configurable silicon signature to be programmed into the device. It can be set using **Program Design > Configure Programming Options** available in the Libero Design Flow as shown in the following figure.

Figure 7 • JTAG USERCODE or Silicon Signature Configuration



2.5 How to Run the Demo

This section provides instructions to run the demo of the design example.

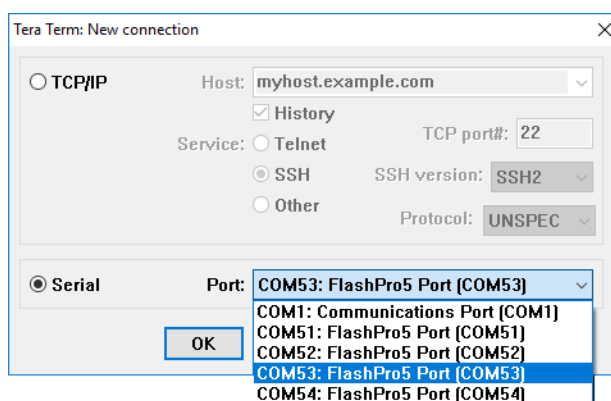
2.5.1 Tera Term Setup

The design example provides a user interface on the Tera Term terminal through the UART interface.

To set up the Tera Term program:

1. Ensure that the USB cable connects the host PC to the J5 (USB) port on the PolarFire Evaluation board.
2. Start Tera Term.
3. Select **File > New connection...** from the Tera Term menu
4. Select **Serial** as the Connection type.
5. Set the Serial **Port** to the second highest COM port number from the drop-down list as shown in the following figure. For example, **COM53: FlashPro5 Port [COM53]** in this case.

Figure 8 • Select Serial Port



6. In the Tera Term window, go to Setup > **Serial port...**, set Baud rate as 115200. This completes the Tera Term program setup.

2.5.2 Board Setup

To program the PolarFire device, complete the following steps:

1. Ensure that the jumpers on the evaluation board are set as specified in the following table.

Table 3 • Jumper Settings

Jumper	Description
J18, J19, J20, J21, J22	Close pin 2 and 3 for programming the PolarFire FPGA through FTDI
J28	Close pin 1 and 2 for programming through the on-board FlashPro5
J26	Close pin 1 and 2 for programming through the FTDI SPI
J27	Close pin 1 and 2 for programming through the FTDI SPI
J23	Open pin 1 and 2 for programming SPI Flash
J4	Close pin 1 and 2 for manual power switching using SW3
J12	Close pin 3 and 4 for 2.5 V
SW11	DIP1 switch position should be set to ON.

2. Connect the power supply cable to the **J9** connector on the evaluation board.
3. Connect the USB cable from the host PC to **J5** (FTDI port) on the evaluation board.
4. Power on the evaluation board using the **SW3** slide switch.

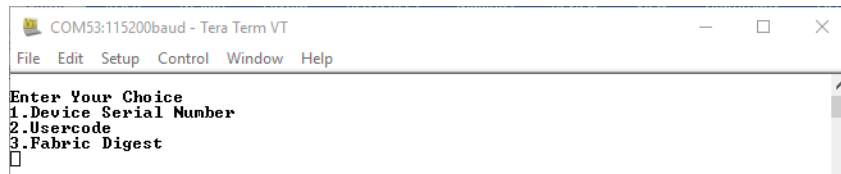
2.5.3 Program the Device

1. Open the provided Libero project.
2. Click **Run PROGRAM Action** to program the device.

2.5.4 Running the Demo

After the device is programmed with the provided design. The design prints the menu on the Tera Term program through the UART interface, as shown in following figure. The state of a 4-bit free-running counter is shown using four user LEDs on the board.

Figure 9 • User Interface Menu



```

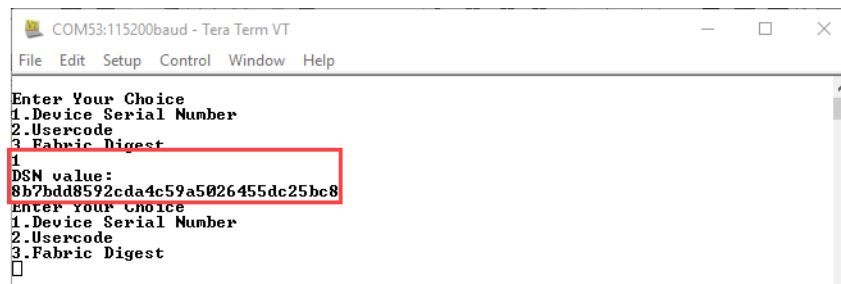
COM53:115200baud - Tera Term VT
File Edit Setup Control Window Help
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest

```

To run the demo:

1. Press '1' to run device serial number (DSN) system service. As a result, the DSN of the device present on the evaluation board is printed on the terminal, as shown in the following figure. The DSN returned from the service is same as the DSN exported during device programming.

Figure 10 • Fetch DSN System Service



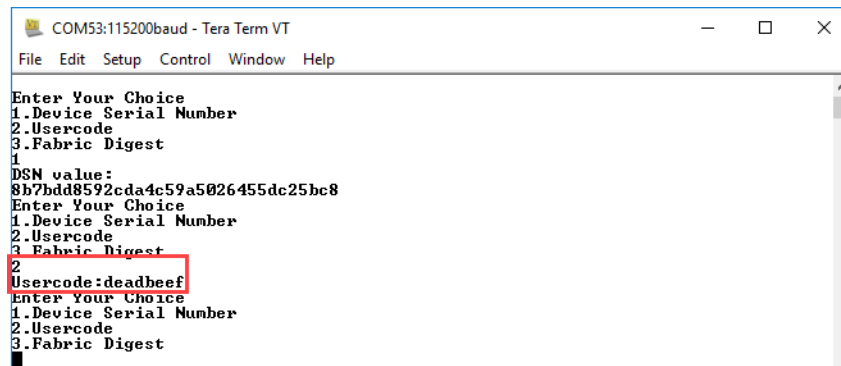
```

COM53:115200baud - Tera Term VT
File Edit Setup Control Window Help
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest
1
DSN value:
8b7bdd8592cda4c59a502645dc25bc8
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest

```

2. Press '2' to execute Fetch JTAG USERCODE system service. As a result, the USERCODE or silicon signature programmed in the device is printed on the terminal, as shown in the following figure.

Figure 11 • Fetch JTAG USERCODE System Service



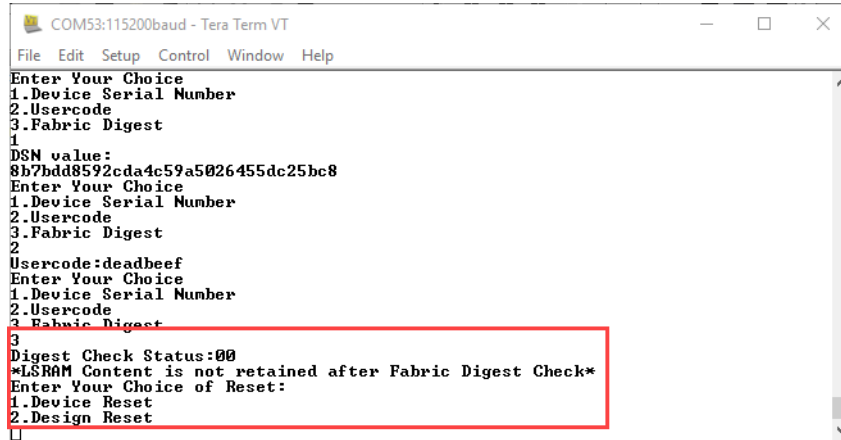
```

COM53:115200baud - Tera Term VT
File Edit Setup Control Window Help
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest
1
DSN value:
8b7bdd8592cda4c59a502645dc25bc8
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest
2
Usercode:deadbeef
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest

```

- Press '3' to perform digest check on FPGA fabric. Notice that the user LEDs on the board gets off while fabric digest check is being performed and then resume counting from previous state after exiting from suspend state.
The status of the fabric digest check is returned by the service and it is printed on the terminal. Status code of 0x00 means no error in the fabric digest check.

Figure 12 • Fabric Digest Check System Service



```

COM53:115200baud - Tera Term VT
File Edit Setup Control Window Help
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest
1
DSN value:
8b7bdd8592cda4c59a502645dc25bc8
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest
2
Usercode:deadbeef
Enter Your Choice
1.Device Serial Number
2.Usercode
3.Fabric Digest
3
Digest Check Status:00
*LSRAM Content is not retained after Fabric Digest Check*
Enter Your Choice of Reset:
1.Device Reset
2.Design Reset

```

- Press '1' to issue a device reset. Notice that the LEDs start counting from 0x0 since the counter gets reset. If you press '2' instead of '1' then design gets reset but the counter continues to count without reset. The rest of the design gets restarted in both the cases and terminal shows the start menu.

This concludes the demo of Fabric Digest Check system service.