

## Introduction (Ask a Question)

The PolarFire® family of devices include a System Controller, which accepts and responds to system service requests from the user. System services range from requesting the device and design specific information to requesting an IAP or auto-update.

This user guide describes the system services available in the PolarFire family. The FPGA fabric is common to the PolarFire family, which consists of the following FPGA devices.

- PolarFire FPGAs** Microchip's PolarFire® FPGAs are the fifth-generation family of non-volatile FPGA devices, built on state-of-the-art 28 nm non-volatile process technology. PolarFire FPGAs deliver the lowest power at mid-range densities. PolarFire FPGAs lower the cost of mid-range FPGAs by integrating the industry's lowest power FPGA fabric, lowest power 12.7 Gbps transceiver lane, built-in low power dual PCI Express Gen2 (EP/RP), and, on select data security (S) devices, an integrated low-power crypto co-processor.
- PolarFire SoC FPGAs** Microchip's PolarFire SoC FPGAs are the fifth-generation family of non-volatile SoC FPGA devices, built on state-of-the-art 28 nm non-volatile process technology. The PolarFire SoC family offers industry's first RISC-V based SoC FPGAs capable of running Linux. It combines a powerful 64-bit 5x core RISC-V Microprocessor Subsystem (MSS), based on SiFive's U54-MC family, with the PolarFire FPGA fabric in a single device.
- RT PolarFire FPGAs** Microchip's RT PolarFire® FPGAs combine our 60 years of space flight heritage with the industry's lowest-power PolarFire FPGA family to enable new capabilities for space and mission-critical applications. RT PolarFire FPGA family includes RTPF500T, RTPF500TL, RTPF500TS, RTPF500TLS, RTPF500ZT, RTPF500ZTL, RTPF500ZTS, and RTPF500ZTLS devices.
- RT PolarFire SoC FPGAs** Designed to enable high-performance data processing, our radiation-tolerant PolarFire SoC FPGA is the industry's first embedded, real-time, Linux®-capable, RISC-V®-based Microprocessor Subsystem (MSS) on the flight-proven RT PolarFire FPGA fabric. With our extensive Mi-V ecosystem, designers can develop lower-power solutions for the challenging thermal environments seen in space.

The following table lists the categories of system services available in the PolarFire Family.

**Table 1.** PolarFire Family System Services

System Services	PolarFire® FPGA (MPF)	RT PolarFire (RTPF)	PolarFire SoC FPGA (MPFS)	RT PolarFire SoC (RTPFS)
Device and Design Information Services	✓	✓	✓	✓
Design Programming Services	✓	✓	✓	✓
Data Security Services	✓	✓	✓	✓
Fabric Services	✓	✓	✓	✓
Debug Services	—	—	✓	✓
Passcode Services	—	—	✓	✓
SPI Flash Memory Read Service	—	—	✓	✓

## References (Ask a Question)

- For information about PolarFire FPGA CoreSysServices\_PF driver and example SoftConsole project, see *Firmware Catalog*. Firmware Catalog is available in the Libero SoC installation package.

- For information about PolarFire SoC FPGA mss\_system\_services driver and example SoftConsole project, see [GitHub](#).
- For information about PolarFire FPGA system services SgCore and its configuration, see [UG0848 PolarFire System Services User Guide](#).
- For information about device serial number, see [PolarFire Family Security User Guide](#).
- For information about eNVM parameters, see [PolarFire SoC Datasheet](#) and *RT PolarFire SoC Datasheet*, respectively.

# Table of Contents

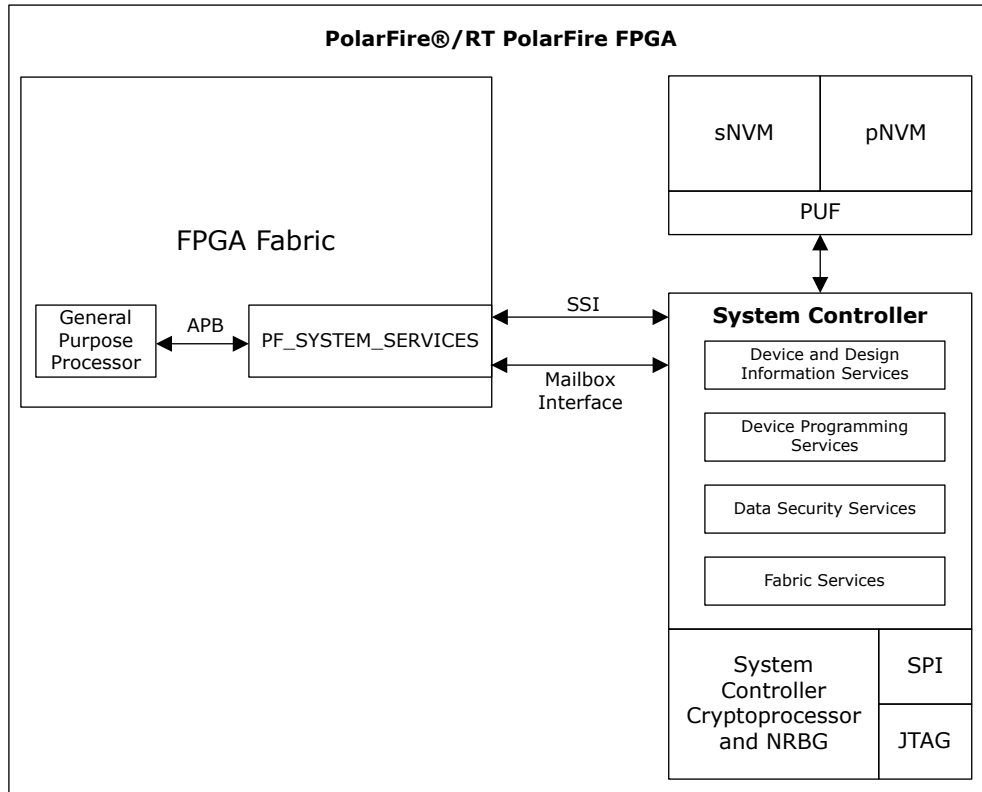
Introduction.....	1
References.....	1
1. PolarFire and RT PolarFire FPGA System Services.....	5
2. PolarFire SoC and RT PolarFire SoC FPGA System Services.....	6
3. System Service Request.....	7
4. Device and Design Information Services.....	8
4.1. Serial Number Service.....	8
4.2. USERCODE Service.....	8
4.3. Design Information Service.....	9
4.4. Device Certificate Service.....	9
4.5. Read Digests Service.....	10
4.6. Query Security Service.....	11
4.7. Read Debug Information Service.....	13
5. Design Programming Services.....	17
5.1. Bitstream Authentication Service.....	17
5.2. IAP Image Authentication Service.....	18
6. Data Security Services.....	19
6.1. Digital Signature Service.....	20
6.2. Secure NVM (sNVM) Services.....	20
6.3. PUF Emulation Service.....	23
6.4. Nonce Service.....	24
7. Fabric Services.....	26
7.1. Digest Check Service.....	26
7.2. In-Application Programming (IAP) Service.....	28
7.3. Auto Update Service.....	30
8. Debug Services (For PolarFire SoC and RT PolarFire SoC FPGA Only).....	31
8.1. Probe Read Debug Service.....	32
8.2. Probe Write Debug Service.....	32
8.3. Live Probe Debug Service.....	33
8.4. MEM Select Debug Service.....	33
8.5. MEM Read Debug Service.....	34
8.6. MEM Write Debug Service.....	35
8.7. APB Read Debug Service.....	35
8.8. APB Write Debug Service.....	36
8.9. Debug Snapshot Service.....	36
8.10. Terminate Debug Service.....	36
9. Passcode Services (For PolarFire SoC and RT PolarFire SoC FPGA Only).....	38
9.1. Generate OTP Service.....	38
9.2. Match OTP Service.....	39
9.3. Unlock Debug Passcode Service.....	39

9.4. One Way Passcode Service.....	39
10. SPI Flash Memory Read Service (For PolarFire SoC and RT PolarFire SoC FPGA Only).....	41
10.1. SPI Copy Service.....	41
11. Using System Services in PolarFire and RT PolarFire FPGA.....	42
12. Using System Services in PolarFire SoC and RT PolarFire SoC FPGA.....	43
13. System Controller Suspend Mode.....	45
14. Revision History.....	48
Microchip FPGA Support.....	49
Microchip Information.....	49
Trademarks.....	49
Legal Notice.....	49
Microchip Devices Code Protection Feature.....	50

## 1. PolarFire and RT PolarFire FPGA System Services [\(Ask a Question\)](#)

In PolarFire and RT PolarFire FPGAs, system services are system controller actions initiated by the fabric user logic through the system controller's system service interface (SSI). For initiating the system services, the fabric user logic requires the PF\_SYSTEM\_SERVICES SgCore IP available in the Libero catalog. The PF\_SYSTEM\_SERVICES SgCore IP provides an easy user interface to run the system services. The following figure shows the design interface between fabric and system controller.

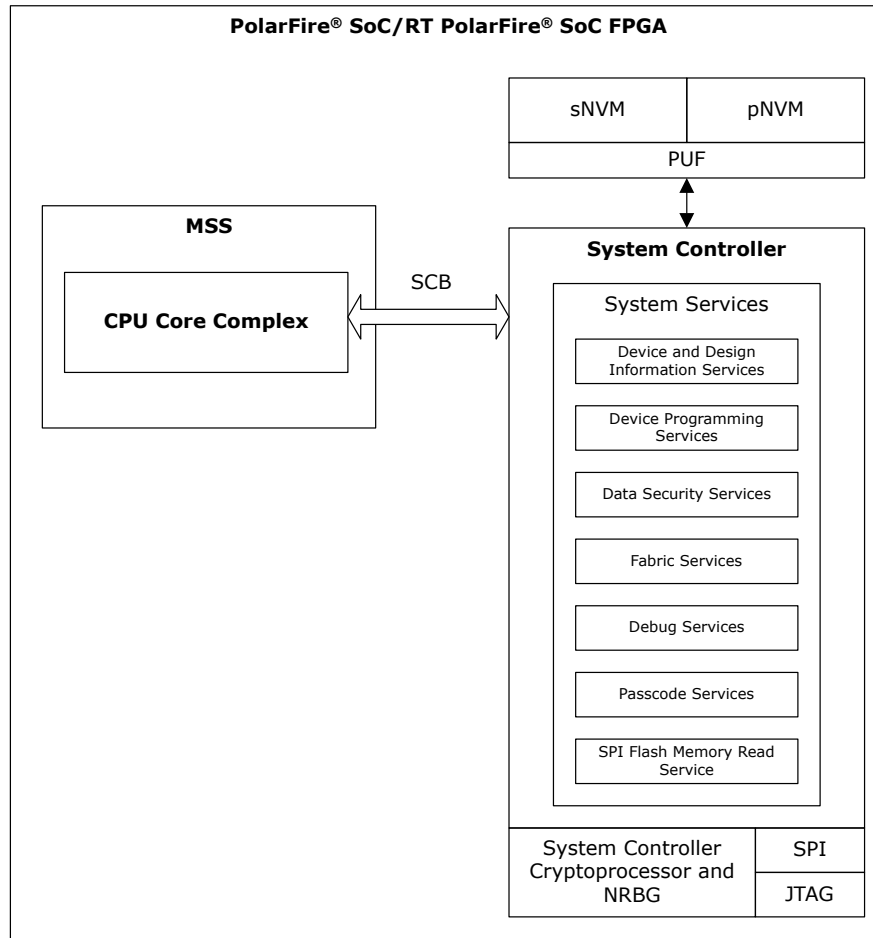
**Figure 1-1.** Design Interface Between Fabric and System Controller



## 2. PolarFire SoC and RT PolarFire SoC FPGA System Services [\(Ask a Question\)](#)

In PolarFire SoC and RT PolarFire SoC FPGA, system services are System Controller actions initiated by the MSS, which communicates with the System Controller over System Controller Bridge (SCB) bus. The following figure shows the design interface between the MSS and System Controller.

**Figure 2-1.** Design Interface Between MSS and System Controller



### 3. System Service Request [\(Ask a Question\)](#)

In the PolarFire Family, the system service request is initiated by passing a 16-bit system service descriptor to the System Controller. The lower seven bits of the descriptor specify the service to be performed and the upper nine bits specify mailbox address offset. There is a 2 Kbytes internal mailbox RAM memory space. This space is used for passing the input data and storing the service request output that is returned by the System Controller. The mailbox address specifies the service-specific data structure that is used for any additional inputs to or outputs from the service. On completion of service, the System Controller writes a status code indicating the successful completion of the system service or an error code. The following table lists the system service request descriptor bits. For information about mailbox read/write communication from Fabric, see [UG0848 PolarFire System Services User Guide](#).

**Table 3-1.** PolarFire Family System Service Request Descriptor

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Specifies the address offset in mailbox RAM to access minimum four bytes of memory. Mailbox addresses are specified using a word offset (0-511).
6:0	SERVICECMD	Service command for System Controller to execute the request.

## 4. Device and Design Information Services [\(Ask a Question\)](#)

These services provide return information about the device and current user design. The requested information is copied to a location whose address is included in the service descriptor. The size of the data returned is service dependent. The following table lists all the device and design information system services with their command values, description, and return status code.

**Table 4-1.** Device and Design Information System Services

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
<a href="#">Serial Number Service</a>	00	Fetches the 128-bit device serial number.	0: Success
<a href="#">USERCODE Service</a>	01	Fetches the 32-bit USERCODE/Silicon signature.	
<a href="#">Design Information Service</a>	02	Returns design information including 256-bit user defined design ID, 16-bit design version, and 16-bit design back-level protection value.	
<a href="#">Device Certificate Service</a>	03	Fetches the device's Supply Chain Assurance Certificate from pNVM.	<ul style="list-style-type: none"> <li>0: Success (Certificate is valid and consistent with device.)</li> <li>1: Device mismatch (Public key or factory serial number do not match device.)</li> <li>2: Certificate signature is invalid</li> <li>3: PUF or storage failure</li> </ul>
<a href="#">Read Digests Service</a>	04	Returns the stored digests for the device.	0: Success
<a href="#">Query Security Service</a>	05	Fetches non-volatile states of user security locks.	
<a href="#">Read Debug Information Service</a>	06	Fetches debug information on programming, user initialization, device programming cycle count, and In-application programming (IAP) actions.	

### 4.1. Serial Number Service [\(Ask a Question\)](#)

Fetches the 128-bit device serial number.

**Table 4-2.** Serial Number Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 4-3</a> .
6:0	00H	Serial number service command

The following table lists the Serial Number Service mailbox format.

**Table 4-3.** Serial Number Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	16	DSN	Output	Device Serial Number

For more information about Device Serial Number, see [PolarFire Family Security User Guide](#).

### 4.2. USERCODE Service [\(Ask a Question\)](#)

Fetches the 32-bit USERCODE/Silicon signature.

**Table 4-4.** USERCODE Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 4-5</a> .
6:0	01H	USERCODE service command

The following table lists the USERCODE Service mailbox format.

**Table 4-5.** USERCODE Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	4	USERCODE	Output	Device USERCODE

### 4.3. Design Information Service [\(Ask a Question\)](#)

Returns design information including 256-bit user defined design ID, 16-bit design version, and 16-bit design back-level protection value.

**Table 4-6.** Design Information Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 4-7</a> .
6:0	02H	Design Information service command

The following table lists the Design Information Service mailbox format.

**Table 4-7.** Design Information Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	32	DESIGNID	Output	256-bit user-defined design ID
32	2	DESIGNVER	Output	16-bit design version
34	2	BACKLEVEL	Output	16-bit design back-level

### 4.4. Device Certificate Service [\(Ask a Question\)](#)

Fetches the device's Supply Chain Assurance Certificate from pNVM. The certificate data is stored as a 1024-bit entity but the actual certificate size may be smaller. Any excess bytes should be discarded by the user.

The device validates the certificate by checking its signature using the Microchip public key, MCPK. In addition:

- The certificate DSN is checked against the device's serial number (DSN).
- The certificate public key is checked by recalculating the value using factory key and comparing against the certificate.

In the event of an error, the certificate content is still returned for inspection.

**Table 4-8.** Design Certificate Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 4-9</a> .
6:0	03H	Design Certificate service command.

The following table lists the Design Certificate Service mailbox format.

**Table 4-9.** Design Certificate Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	1024	CERTIFICATE	Output	Device Certificate

#### 4.5. Read Digests Service [\(Ask a Question\)](#)

During the programming of a device, cryptographic hashes (SHA-256) are calculated over all non-volatile component of the device and stored within the device for future reference. These hashed values are known as Digests and are unique to the content programmed into the device. The Read Digest Service returns the stored digests from the device. These digest can be read from the device at any time to compare to the values reported during original programming to determine if any programmed content change occurred in the device. These can also be helpful in determining if a specific programming file was used to program the device. For more details, see the [PolarFire Family FPGA Security User Guide](#).

The following table lists the read digest service request format.

**Table 4-10.** Read Digest Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 4-11</a> .
6:0	04H	Read Digest service command.

The following table lists the Read Digest Service mailbox format.

**Table 4-11.** Read Digests Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	416 (For PolarFire® and RT PolarFire FPGA only) 576 (For PolarFire SoC and RT PolarFire SoC FPGA only)	DIGESTS	Output	Digest Array

The following table lists the returned digests format.

**Table 4-12.** Returned Digests Format

Offset (byte)	Size (bytes)	Value	Description
0	32	FD	Fabric digest
32	32	CCDIGEST	Digest of fabric configuration parameters such as cycle count, design version, and back level protection value
64	32	SNVMDIGEST	Digest of sNVM pages marked as ROM
96	32	ULDIGEST	User security segment digest
128	32	UKDIGEST0	Digest of user key segment containing SRAM-PUF data
160	32	UKDIGEST1	Digest of user key segment containing KUP (User EC key)
192	32	UKDIGEST2	Digest of user key segment containing UPK1
224	32	UKDIGEST3	Digest of user key segment containing UEK1
256	32	UKDIGEST4	Digest of user key segment containing DPK
288	32	UKDIGEST5	Digest of user key segment containing UPK2
320	32	UKDIGEST6	Digest of user key segment containing UEK2
352	32	UPDIGEST(UPREM)	Digest of permanent lock security segments
384	32	FDIGEST(SYS)	Digest of factory lock segment, factory key segment in pNVM, and System Controller ROM
416	32	UKDIGEST7	Digest of One-Way Passcode HWM (For PolarFire® SoC and RT PolarFire® SoC FPGA only)
448	32	ENVMDIGEST	Digest of eNVM (For PolarFire SoC and RT PolarFire SoC FPGA only)
480	32	UKDIGEST8	Digest of MSS Boot mode Information (For PolarFire SoC and RT PolarFire SoC FPGA only)

**Table 4-12.** Returned Digests Format (continued)

Offset (byte)	Size (bytes)	Value	Description
512	32	UKDIGEST9	Digest of SNVM_RW_ACCESS_MAP (For PolarFire SoC and RT PolarFire SoC FPGA only)
544	32	UKDIGEST10	Digest of Secure Boot Image Certificate (SBIC) (For PolarFire SoC and RT PolarFire SoC FPGA only)

#### 4.6. Query Security Service [\(Ask a Question\)](#)

Fetches non-volatile states of user security locks. The following table lists the description of returned LOCKS array.

**Table 4-13.** Query Security Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 4-14</a> .
6:0	05H	Query Security service command

The following table lists the Query Security Service mailbox format.

**Table 4-14.** Query Security Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	9 (For PolarFire® and RT PolarFire FPGA only) 33 (For PolarFire SoC and RT PolarFire SoC FPGA only)	Locks	Output	Lock Array

**Table 4-15.** Returned LOCKS Array

Byte	Bit	Lock	Description
0	0	UL_DEBUG	Debug instructions disable
0	1	UL_SNVM_DEBUG	sNVM debug disable
0	2	UL_LIVEPROBE	Live probes disable
0	3	UL_UJTAG	User JTAG interface disable
0	4	UL_JTAG_BS	JTAG boundary scan disable
0	5	UL_TVS_MONITOR	External access to System TVS monitor disable
0	6	UL_JTAG_MONITOR	JTAG fabric monitor enable
0	7	UL_JTAG	JTAG TAP disable
1	0	UL_PLAINTEXT	Plaintext passcode unlock disable
1	1	UL_FAB_PROTECT	Fabric erase/write disable
1	2	UL_EXT_DIGEST	External digest check disable
1	3	UL_VERSION	Replay protection enable
1	4	UL_FACT_UNLOCK	Factory test disable
1	5	UL_IAP	IAP disable
1	6	UL_EXT_ZEROIZE	External zeroization disable
1	7	UL_SPI_SLAVE	SPI port disable
2	0	UL_USL	UFS UL segment protect
2	1	UL_BS_AUTHENTICATE	External bitstream authentication disable
2	2	UL_BS_PROGRAM	External bitstream program mode disable
2	3	UL_BS_VERIFY	External bitstream verify mode disable
2	4	UL_BITS_KEYMD[0]	Bitstream key mode disable
2	5	UL_BITS_KEYMD[1]	Bitstream key mode disable
2	6	UL_BITS_KEYMD[2]	Bitstream key mode disable

**Table 4-15. Returned LOCKS Array (continued)**

Byte	Bit	Lock	Description
2	7	UL_BITS_KEYMD[3]	Bitstream key mode disable
3	0	UL_BITS_KEYMD[4]	Bitstream key mode disable
3	1	UL_BITS_KEYMD[5]	Bitstream key mode disable
3	2	UL_BITS_KEYMD[6]	Bitstream key mode disable
3	3	UL_BITS_KEYMD[7]	Bitstream key mode disable
3	4	UL_BITS_KEYMD[8]	Bitstream key mode disable
3	5	UL_BITS_KEYMD[9]	Bitstream key mode disable
3	6	UL_BITS_KEYMD[10]	Bitstream key mode disable
3	7	UL_BITS_KEYMD[11]	Bitstream key mode disable
4	0	UL_BITS_KEYMD[12]	Bitstream key mode disable
4	1	UL_BITS_KEYMD[13]	Bitstream key mode disable
4	2	UL_BITS_KEYMD[14]	Bitstream key mode disable
4	3	UL_BITS_KEYMD[15]	Bitstream key mode disable
4	4	UL_KEYMD[0]	Global key mode disable
4	5	UL_KEYMD[1]	Global key mode disable
4	6	UL_KEYMD[2]	Global key mode disable
4	7	UL_KEYMD[3]	Global key mode disable
5	0	UL_KEYMD[4]	Global key mode disable
5	1	UL_KEYMD[5]	Global key mode disable
5	2	UL_KEYMD[6]	Global key mode disable
5	3	UL_KEYMD[7]	Global key mode disable
5	4	UL_KEYMD[8]	Global key mode disable
5	5	UL_KEYMD[9]	Global key mode disable
5	6	UL_KEYMD[10]	Global key mode disable
5	7	UL_KEYMD[11]	Global key mode disable
6	0	UL_KEYMD[12]	Global key mode disable
6	1	UL_KEYMD[13]	Global key mode disable
6	2	UL_KEYMD[14]	Global key mode disable
6	3	UL_KEYMD[15]	Global key mode disable
6	4	UL_SNVM_PROTECT	sNVM bitstream write protection enable
6	5	UL_EXT_CHALLENGE	CHALLENGE instruction disable
6	6	UL_UEK_PROTECT	UEK overwrite protection
6	7	UL_HWM	High Water Mark Reset disable (For PolarFire SoC and RT PolarFire SoC FPGA only)
7	0	UL_ENVM_PROTECT	Disable bitstream programming of eNVM (For PolarFire SoC and RT PolarFire SoC FPGA only)
7	1	UL_USER_KEY	User Key1 write protect
7	2	UL_USER_KEY2	User Key2 write protect
7	3	UP_FACTORY	Permanent factory test disable
7	4	UP_DEBUG	Permanent debug disable
7	5	UP_FABRIC	Permanent fabric write protect
7	6	UP_UPK1	Permanent disable of UPK1
7	7	UP_UPK2	Permanent disable of UPK2
8	0	UP_DPK	Permanent disable of DPK
8	1	UP_PROTECT	Write disable for UPERM segment

**Table 4-15.** Returned LOCKS Array (continued)

Byte	Bit	Lock	Description
9:11	—	RESERVED	Reserved
12	0	UATHENA_ENA	User F5200 enable (For PolarFire® SoC and RT PolarFire SoC FPGA only)
12	1:2	UATHENA_MODE	User F5200 mode (For PolarFire SoC and RT PolarFire SoC FPGA only)
12	3:5	U_CLKMON_FREQ	System Controller Clock Frequency monitor configuration (For PolarFire SoC and RT PolarFire SoC FPGA only)
13	—	RESERVED	Reserved
14-15	—	PORDIGEST[15:0]	This field specifies a mask of device digests that should be checked upon each power-up of the user design (For PolarFire SoC and RT PolarFire SoC FPGA only)
16-31	—	HWM	High Water Mark value (For PolarFire SoC and RT PolarFire SoC FPGA only) All 1s are returned if the HWM is invalid.
32	—	PORDIGEST[23:16]	This field specifies a mask of device digests that should be checked upon each power-up of the user design (For PolarFire SoC and RT PolarFire SoC FPGA only)

#### 4.7. Read Debug Information Service [\(Ask a Question\)](#)

Fetches debug information on programming, user initialization, device programming cycle count, and In-application programming (IAP) actions. The device programming cycle count increases for device PROGRAM and ERASE actions. The following table lists the debug information reported by this service.

**Table 4-16.** Read Debug Information Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 4-17</a> .
6:0	06H	Read Debug service command.

**Table 4-17.** Debug Information

Size (Bytes)	Byte Offset	Parameter	Description
32	0	Reserved	Reserved
4	32	TOOL_INFO	Reflects the tool specific data passed in during programming. IAP sets this field to 0.
1	36	TOOL_TYPE	Tool type used to program device: <ul style="list-style-type: none"> <li>• 1: JTAG</li> <li>• 2: IAP</li> <li>• 3: SPI_SLAVE</li> </ul>
4	37	Reserved	Reserved
1	41	FRAME_ERRORCODE	An error has occurred during bitstream frame processing and the error is identified by the FRAME_ERRORCODE field. See <a href="#">Table 4-20</a> .
6	42	Reserved	Reserved
1	48	UIC_STATUS	Device and design initialization Status. <ul style="list-style-type: none"> <li>• 0: Successful completion.</li> <li>• Others: Device and design initialization failed.</li> </ul>
11	49	Reserved	Reserved

**Table 4-17. Debug Information (continued)**

Size (Bytes)	Byte Offset	Parameter	Description
4	60	CYCLECOUNT	Programming cycle count. CYCLECOUNT gets incremented for both Programming and Erase operations as ERASE is internally a Programming action.
1	64	IAP_ERRORCODE	IAP Error Information. Returns ERRORCODE 21-27, see <a href="#">Table 4-20</a> .
7	65	Reserved	Reserved
4	72	IAP Location	SPI address that was last run in IAP
4	76	SYSCTRL_STATUS	System Controller status
4	80	RESET_REASON	When a reset occurs, a bit is set in the reset reason register that allows the source of the reset to be determined post reset. Multiple bits are set in the reset reason register when a reset occurs. See <a href="#">Table 4-19</a> for bit position of the bits set in the RESET_REASON register to find out the source of reset.

**Table 4-18. SYSCTRL\_STATUS**

Field Name	Width (bits)	Field Offset
SRAM_INIT_COMPLETE	1	0
UIC_SCRIPT_COMPLETE	1	8
US_RAM_INIT_DONE	1	16
RESET_REASON	5	24

**Table 4-19. Reset Reason**

Reset Reason	Description	RESET_REASON[13:0] Register
POR	Master POR asserted.	0
POR1P05	POR asserted due to the 1P05 supply is not at correct level.	1
POR1P8	POR asserted due to the 1P8 supply is not at correct level.	2
POR2P5	POR asserted due to the 2P5 supply is not at correct level.	3
Reserved	Reserved.	4
DEVIRST	Indicates that the device is reset by the DEVIRSTN pin.	5
AVIONICS	Indicates that the device has exited System Controller suspend mode.	6
WATCHDOG	Indicates that the System Controller's watchdog had triggered the reset.	7
SYSCON_SYSRESET	Indicates that the System Controller requested the system reset.	8
MESH	Indicates that the security mesh triggered the reset.	9
SECURITY_LOCKS	Indicates that the security locks system detected a security issue and reset the system.	10
CLOCK_GLITCH	Indicates that the clock glitch system detected an issue and reset the system.	11
VIRGIN	Indicates that the virgin security system detected an issue and reset the system.	12
TAMPER_RESPONSE	Indicates that the user asserted the system reset tamper response.	13

Table 4-20. ERRORCODE

ERRORCODE	Description	Additional Notes
0	No error	—
1	Bitstream authentication failed	Invalid bitstream or wrong key used.
2	Unexpected data received	Additional data is received after end of bitstream component.
3	Invalid/corrupt encryption key	The requested key mode is disabled or the key could not be read/reconstructed.
4	Invalid component header	Invalid bitstream
5	Back level not satisfied	Bitstream version is older than that of the current back level value set in the device.
6	Illegal bitstream key mode	Bitstream key mode is not initialized or bitstream key mode is disabled by user security.
7	DSN binding mismatch	Bitstream is rejected because DSN in the bitstream does not match with the DSN present in the device. A bitstream can be bound to device's unique DSN such that only a specific device can be programmed with that bitstream.
8	Illegal component sequence	Incorrect bitstream
9	Insufficient device capabilities	Bitstream is rejected because the capabilities specified in the bitstream do not match the target device's capabilities.
10	Incorrect DEVICEID	Bitstream is rejected because an attempt by the DEVICEID specified in the bitstream does not match the part identification field (for example, MPF300, MPF500 and so on) of the target device.
11	Unsupported bitstream protocol version (bitstream regeneration required)	Bitstream is rejected because of an attempt made by the old version of a device to decode a bitstream created in new format or by the new version of a device to decode a bitstream created in old format.
12	Verify not permitted on this bitstream	Verify programming action is disabled in the bitstream.
13	Invalid Device Certificate	Device certificate is invalid or not present.
14	Invalid DIB	Device integrity bits (DIB) are invalid.
21	Device not in SPI Master Mode	Error may occur only when bitstream is executed through IAP mode. The System Controller SPI controller is not configured in master mode.
22	No valid images found	Error may occur when bitstream is executed through Auto Update mode. Occurs when No valid image pointers are found.
23	No valid images found	Error may occur when bitstream is executed through IAP mode via Index Mode. Occurs when No valid image pointers are found.
24	Programmed design version is the same as the Auto Update image found	Error may occur when bitstream is executed through Auto Update mode.
25	Reserved	Reserved
26	Selected image was invalid and no recovery was performed due to valid design in device.	Error may occur only when bitstream is executed through Auto Update or IAP mode. Error could also occur due to BACKLEVEL protection.
27	Selected and Recovery image failed to program	Error may occur only when bitstream is executed through Auto Update or IAP mode.
127	Abort	Non-bitstream instruction is executed during bitstream loading.
128	NVMVERIFY	Fabric or security segment verification failed.
129	PROTECTED	Device security is prevented modification of non-volatile memory.
130	NOTENA	Programming mode not enabled

**Table 4-20. ERRORCODE (continued)**

ERRORCODE	Description	Additional Notes
131	PNVMVERIFY	pNVM verify operation failed
132	SYSTEM	System hardware error (PUF or DRBG)
133	BADCOMPONENT	An internal error was detected in a bitstream component payload.
134	HVPROGERR	Failure in programming subsystem.
135	HVSTATE	Error in the programming subsystem.

## 5. Design Programming Services [\(Ask a Question\)](#)

An IAP image contains the image descriptor, bitstream, and optional design initialization data. The design programming services are used to authenticate entire IAP image, bitstream portion, or program the device. The following table lists all the Design Programming system services with their command values, description, and return status code.

**Table 5-1.** Design Programming System Services

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
<a href="#">Bitstream Authentication Service</a>	23	Analyzes a bitstream image stored in SPI Flash and checks for all conditions, which may result in an authentication error.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: ERRORCODE (see <a href="#">Table 4-20</a>)</li> </ul>
<a href="#">IAP Image Authentication Service</a>	22	Allows the user to validate an IAP image stored in SPI flash.	

### 5.1. Bitstream Authentication Service [\(Ask a Question\)](#)

Prior to using the IAP service, it may be required to first validate the new bitstream before committing the device to reprogramming, thus avoiding the need to invoke recovery procedures if the bitstream is invalid.

The bitstream authentication service analyzes a bitstream image stored in SPI Flash and checks for all conditions which may result in an authentication error. While the authentication is in progress, the user design continues to operate normally, but without access to SPI Flash and system services until the authentication process is complete.

If the authentication service is called while a new bitstream is being loaded through the JTAG interface, the system service takes precedence and the JTAG interface is stalled during the authentication and will ultimately fail.

**Table 5-2.** Bitstream Authentication Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 5-3</a> .
6:0	23H	Bitstream authentication service command.

The following table lists the bitstream authentication service mailbox format.

**Table 5-3.** Bitstream Authentication Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	4	SPIADDR	Input	Address of the bitstream in SPI Flash. If the external SPI Flash device does not support 32-bit addresses, SPIADDR[31:24] is ignored.

## 5.2. IAP Image Authentication Service [\(Ask a Question\)](#)

Allows the user to validate an IAP image stored in SPI Flash. The service authenticates the entire IAP image containing the image descriptor, the referenced bitstream, and optional initialization data. If the image is authenticated successfully, the image is guaranteed to be valid when used by an IAP programming service.

The SPI\_IDX parameter passed to this service identifies the index in the SPI directory to be used. To support recovery, SPI\_IDX = 1 must be an empty slot and the recovery image must be located in SPI\_IDX = 0. Since SPI\_IDX = 1 must be an empty slot, it should not be passed into the system service. The following table lists the fields contained in an IAP image authentication service request.

**Table 5-4.** IAP Image Authentication Service Request

System Service Descriptor Bit Field	Value	Description
15	—	Reserved.
14:7	IMAGEID[7:0]	Identifies the image index in the SPI directory for image authentication.
6:0	22H	IAP Image Authenticate service command.

## 6. Data Security Services [\(Ask a Question\)](#)

The data security services are used to authenticate the device, generate unique random number, and store the encrypted data.



### Important:

Data security services require the use of PolarFire/RT PolarFire/PolarFire SoC/RT PolarFire SoC 'S' version devices. Non-S version devices support the following subset of data security services:

- Non-authenticated plaintext sNVM service.
- Nonce service. The 'S' version devices provide the addition of nonce DPA protection to better protect against template attacks.

The following table lists all the Data Security system services with their command values, description, and return status code.

**Table 6-1.** Data Security System Services

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
Digital Signature Service	19, 1A	Takes a user-supplied SHA-384 hash and signs it with the device's 384-bit private "factory" EC key, FEK, which is the private half of the key pair whose public key (DCPK) is certified by Microchip in the device's X.509-compliant supply chain assurance certificate.	<ul style="list-style-type: none"> <li>• 0: Success</li> <li>• 1: FEK error</li> <li>• 2: DRBG error (Failed to generate nonce.)</li> <li>• 3: ECDSA error</li> </ul>
Secure NVM Write Service	10, 11, 12	Provides write access to pages in the sNVM.	<ul style="list-style-type: none"> <li>• 0: Success</li> <li>• 1: Invalid SNVMADDR (Illegal page address.)</li> <li>• 2: Write failure (PNVM program or verify failed.)</li> <li>• 3: PUF or storage failure</li> <li>• 4: Write not permitted</li> <li>• 5: Access failure (For PolarFire® and RT PolarFire FPGA, write access from Fabric is blocked. For PolarFire SoC and RT PolarFire SoC FPGA, write access from either Fabric or MSS is blocked.)</li> </ul>
Secure NVM Read Service	18	Provides access to the data stored by the Secure NVM Write service or data programmed through a bitstream.	<ul style="list-style-type: none"> <li>• 0: Success</li> <li>• 1: Invalid SNVMADDR (Illegal page address.)</li> <li>• 2: Authentication failure (Page blank, storage corrupt, or incorrect USK.)</li> <li>• 3: PUF or storage failure</li> <li>• 5: Access failure (For PolarFire and RT PolarFire FPGA, read access from Fabric is blocked. For PolarFire SoC and RT PolarFire SoC FPGA, read access from either Fabric or MSS is blocked.)</li> </ul>

**Table 6-1.** Data Security System Services (continued)

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
PUF Emulation Service	20	Provides a mechanism for authenticating a device, or for generating pseudo-random bit strings that can be used for many different purposes.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: Internal error</li> </ul>
Nonce Service	21	Generates a 256-bit random number derived from the start-up states of a dedicated SRAM.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: Error fetching PUK</li> <li>2: Error generating seed</li> </ul>

## 6.1. Digital Signature Service [\(Ask a Question\)](#)

Takes a user-supplied SHA-384 hash and signs it with the device's 384-bit private "factory" EC key, KFP, which is the private half of the key pair whose public key (DCPK) is certified by Microchip in the device's X.509-compliant supply chain assurance certificate. The resulting P-384 ECDSA signature can either be formatted using ASN.1 DER or simply returned in a raw format compatible with the user cryptoprocessor. As ECDSA requires the use of a nonce, the service returns a different result each time, even if the hash input is the same.

The System Controller cryptoprocessor does not directly support generating a nonce with the required numerical range required for ECDSA. It is therefore possible that the generated nonce is rejected, in which case a new nonce is automatically generated until a good value is found. This makes the execution time of this service non-deterministic, however, the probability of an out-of-range nonce being initially generated is extremely low and the probability of a second bad nonce is infinitesimal.

SIGNATURE = ECDSA (KFP, HASH).

If the Raw format is selected, the SIGNATURE field contains two unsigned little-endian 12-word (48 byte) values compatible with the user cryptoprocessor.

If the DER format is selected, the SIGNATURE field is returned in a minimal length DER encoding using a maximum of 104 bytes. If the encoded signature is less than 104 bytes, the output is padded with zeroes. The extra bytes, if any, must be deleted by the user.

**Table 6-2.** Digital Signature Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 6-3</a> .
6:0	19H	Digital signature Raw format service command
	1AH	Digital Signature DER format service command

The following table lists the Digital Signature Service mailbox format.

**Table 6-3.** Digital Signature Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	48	HASH	Input	SHA384 hash to be signed
48	96 (Raw)	SIGNATURE	Output	ECDSA signature (r, s)
	104 (DER)			

## 6.2. Secure NVM (sNVM) Services [\(Ask a Question\)](#)

sNVM data can be stored in any of the following formats:

- Non-authenticated plaintext,

- Authenticated plaintext
- Authenticated ciphertext

When the data is authenticated, 236 bytes of storage per page is available. When the data is not authenticated, 252 bytes can be stored. Non-authenticated plaintext provides the fastest access time and authenticated ciphertext provides the highest level of security. When authentication is used, a user-provided 96-bit key USK is used to enhance security.

sNVM can be marked as ROM during bitstream programming. In this case, sNVM cannot be modified by the Secure NVM services but can be read.

### 6.2.1. Secure NVM Write Service [\(Ask a Question\)](#)


Secure NVM write service provides write access to pages in the sNVM. Data can be stored as encrypted and authenticated ciphertext, authenticated plaintext, and non-authenticated plaintext.

For authenticated plaintext and authenticated ciphertext, a 512-bit sNVM master key (SMK) is the primary key used, with 256-bits allocated for authentication and 256 bits for encryption. SMK is common for all the sNVM pages. The 96-bit user-supplied key, USK is used to authenticate the sNVM page content. The USK does not need to be stored on the device, but it must be presented to the sNVM read system service to correctly retrieve the data.

For crypto-enabled options, the System Controller uses AES-256 in synthetic initialization vector (SIV) mode, which supports authenticated encryption. In SIV mode, the initialization vector used for the encryption function is computed from the data, preventing initialization vector misuse, and doubles as the authentication tag. The computed 128-bit initialization vector is stored in the same page as the user data, reducing the available space for user data by 16 bytes compared to the non-authenticated plaintext-only option.

Besides the user-supplied plaintext data, both the device families also submit additional metadata for authentication that effectively provides a “tweak” to the encryption and authentication functions. Some of the data included are the page address and the page write-counter. This means that the ciphertext and the authentication tag are different even if the same data is written to two different sNVM pages, or even if the same data is written to the same page again (since the page-write counter advances).

The USK is used as another element in the “tweak”. Without the same 96-bit USK, which was used during the write command, the read command fails authentication, however, the plaintext data is still available in the fabric. The user may choose to set this key differently for each page, or for groups of pages, or the same for all pages—either as a secret key for added security, or to a invalid value such as all zeroes if this feature is not needed.

 **Important:** In Libero, the added USK client is stored in the user specified sNVM page and this USK is used for all the authenticated plaintext or authenticated ciphertext clients created in the Libero project. User application in the fabric may use a different USK and overwrite any of the sNVM data clients (not marked as ROM) using sNVM write system service during runtime. However, it causes design verification failure using bitstream, even if the data is same.

sNVM modules marked as ROM cannot be overwritten by this service. The service cannot be used to create ROM modules (write-protected pages). ROM is declared when a bitstream is generated, and a page's ROM status can only be changed with a new bitstream, and not at run-time.

**Table 6-4.** Secure NVM Write Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 6-5</a> and <a href="#">Table 6-6</a> .

**Table 6-4.** Secure NVM Write Request (continued)

System Service Descriptor Bit Field	Value	Description
6:0	10H	Non-authenticated plaintext service command
	11H	Authenticated plaintext service command
	12H	Authenticated ciphertext service command

The following table lists the Secure NVM Write Service Mailbox Format for non-authenticated plaintext (10H).

**Table 6-5.** Secure NVM Write Service Mailbox Format (10H)

Offset	Length (bytes)	Parameter	Direction	Description
0	1	SNVMADDR	Input	sNVM address
1	3	RESERVED		Reserved
4	252	DATA	Input	Data to write to sNVM

The following table lists the Secure NVM Write Service Mailbox Format for authenticated plaintext (11H) and authenticated ciphertext (12H).

**Table 6-6.** Secure NVM Write Service Mailbox Format (11H, 12H)

Offset	Length (bytes)	Parameter	Direction	Description
0	1	SNVMADDR	Input	sNVM address
1	3	RESERVED		Reserved
4	236	DATA	Input	Data to write to sNVM
240	12	USK	Input	User Secret Key



**Important:** If  $V_{DD}$ ,  $V_{DD18}$ , or  $V_{DD25}$  power is lost or experiences a brownout while an sNVM page write is in process, the page may corrupt, where it is no longer capable of being written. This occurs because the page is first erased during the write process, which sets the page as 'read only' bit. If the page is not restored during the subsequent data storage step of the write, the page becomes read only. This can be detected by a subsequent page write after power is restored, which results in a write return status = 0x4 indicating 'write not permitted'. Alternatively, a failed digest check of sNVM also indicates this condition. Reprogramming the device with a bitstream containing affected sNVM page content is required to restore the page to allow writes.

## 6.2.2. Secure NVM Read Service [\(Ask a Question\)](#)

Secure NVM read service provides access to the data stored by the Secure NVM Write service or data programmed through a bitstream. If the data is programmed using authentication, the USK key used at the time of programming must also be provided.

**Table 6-7.** Secure NVM Read Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 6-8</a> .
6:0	18H	Secure NVM Read service command

The following table lists the Secure NVM Read Service Mailbox Format (18H).

**Table 6-8.** Secure NVM Read Service Mailbox Format (18H)

Offset	Length (bytes)	Parameter	Direction	Description
0	1	SNVMADDR	Input	sNVM address
1	3	RESERVED	—	Reserved
4	12	USK	Input	User Secret Key (ignored if page is plaintext)
16	4	ADMIN	Output	Page admin data contains current write counter value, page type, and ROM flag. (see <a href="#">Table 6-9</a> )
20	236 or 252	DATA	Output	Data read from sNVM. 236 bytes of data per page is available when the data is authenticated. 252 bytes of data per page is available when the data is not authenticated.

**Table 6-9.** sNVM Page Admin Data

Field	Offset	Size (bits)	Description
CYCLES	0	20	The current write counter for the page. Since there is no redundant copy, the counter cannot be guaranteed to survive a failed programming attempt.
PAGETYPE	20	2	Specifies how the DATA field is used. Each page may contain plaintext, authenticated plaintext, or authenticated ciphertext. See <a href="#">Table 6-10</a> .
RESERVED	22	1	—
ROMFLAG	23	1	Specifies whether the page can be modified at runtime by the sNVM system services. If '1', the page cannot be written by the sNVM system services.
UNUSED	24	8	—

**Table 6-10.** PAGETYPE and DATA

PAGETYPE	Offset	Size (bits)	DATA Usage	Description
0	—	—	Blank	Blank page
1	0	1888	CT	Authenticated and Encrypted
	1888	128	SIVTAG	
2	0	1888	PT	Authenticated Plaintext
	1888	128	SIVTAG	
3	0	2016	PT	Plaintext

The page admin word (bits 31:0) is stored in ones-complement form. This is necessary to avoid a time consuming operation during zeroization to make the sNVM page look blank upon completion of zeroization.

### 6.3. PUF Emulation Service [\(Ask a Question\)](#)

Provides a mechanism for authenticating a device, or for generating pseudo-random bit strings that can be used for many different purposes. The service accepts a 128-bit challenge and an 8-bit optype, and returns a 256-bit response unique to the given challenge, optype, and device.

RESPONSE = KeyTree(PEK, OPTYPE, CHALLENGE)

Where PEK is the factory-defined PUF emulation key and KeyTree is a function that uses the 8-bit OPTYPE concatenated with the 128-bit CHALLENGE to navigate a binary key tree with the 256-bit secret PEK at its root. The leaf of the tree that is computed as a result of the 136 internal hashing operations (one for each level in the binary tree), is a 256 bit secret. The root key, PEK, the result, RESPONSE, and the intermediate results are protected against side-channel attacks due to the nature of the protocol. The SHA algorithm implemented in the System Controller's cryptoprocessor also has strong DPA countermeasures. The OPTYPE and CHALLENGE are not protected against side-channel leakage. The OPTYPE allows the user to conceptualize that there are 256 different 128-bit key trees, each with  $2^{128}$  possible output responses, which can be put to different uses without much danger of collision.

The function emulates a “strong PUF”, which means that it takes a cryptographically large challenge space and computes a pseudo-random repeatable output response from it, but in this implementation, it does not use unclonable physical properties developed during the manufacturing of the device for the challenge-response calculation, instead using classical cryptographic algorithms; thus the “emulation” disclaimer. The root key PEK is, however, protected as an encrypted/authenticated PUF key code, so the unclonable physical properties of the device do enter into the reconstruction of the PUF secret and decryption of the key code to unwrap PEK for use in this function.

There are many uses in cryptography for such a per-device unique, pseudo-random function. One use is to identify a particular chip by first recording (possibly several) challenge-response pairs, then later seeing if the target chip provides the same response as expected for one of the recorded challenges-response pairs. Another application is deriving many keys from one.

**Table 6-11.** PUF Emulation Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 6-12</a> .
6:0	20H	PUF Emulation service command

The following table lists the PUF Emulation Service Mailbox Format.

**Table 6-12.** PUF Emulation Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	1	OPTYPE	Input	Operational type
1	3	RESERVED	—	Reserved
4	16	CHALLENGE	Input	Challenge input
20	32	RESPONSE	Output	Response output


## 6.4. Nonce Service [\(Ask a Question\)](#)

Generates a 256-bit random number derived from the startup states of a dedicated SRAM. The nonce service provides the user with the ability to strengthen the NRBG of the User Cryptoprocessor random bit generator by providing an alternate entropy source to use as additional seed data in its DRBG functions.

NONCE = KeyTree256(PUK, 0, PUFSEED)

Where, PUFSEED is a 256-bit conditioned true random output of the SRAM-PUF. PUK is a 256-bit device-generated nonce set in the factory.

To generate maximum entropy and forward and backward resistance, the SRAM-PUF is automatically power-cycled before generating the seed.

 **Important:** The nonce service is supported in both ‘S’ and ‘non-S’ versions of the devices. However, the ‘S’ version devices provide the addition of nonce DPA protection to better protect against template attacks.

**Table 6-13.** Nonce Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 6-14</a> .
6:0	21H	Nonce service command

The following table lists the Nonce Service Mailbox Format.

**Table 6-14.** Nonce Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	32	NONCE	Output	Generated nonce

## 7. Fabric Services [\(Ask a Question\)](#)

Fabric services are used to calculate digests of non-volatile memories and program the device. The following table lists all the Fabric system services with their command values, description, and return status code.

**Table 7-1.** Fabric System Services

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
<a href="#">Digest Check Service</a>	47	Recalculates digests of selected non-volatile memories and compares against stored values.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: Returned if any of DIGESTERR bits are set (For PolarFire® SoC FPGA only)</li> </ul> See <a href="#">Table 7-5</a> for digest error codes.
<a href="#">In-Application Programming (IAP) Service</a>	42, 43, 44, 45	IAP reprograms the device with a specific programming image.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: ERRORCODE (see <a href="#">Table 4-20</a>)</li> </ul>
<a href="#">Auto Update Service</a>	46	The newest image of the first two images in the SPI directory is chosen to be programmed.	See <a href="#">Table 4-20</a> for error codes.

### 7.1. Digest Check Service [\(Ask a Question\)](#)

During the programming of a device, cryptographic hashes (SHA-256) are calculated over all non-volatile component of the device and stored within the device for future reference. These hashed values are known as Digests and are unique to the content programmed into the device. The Digest Check Service recalculates digests of selected non-volatile components and compares against the digests stored in the device during programming. If the service returns a digest error it is an indication that the programmed content of the device is not the same as the originally programmed content. The device should be reprogrammed to restore the intended programming. For more information, see the [PolarFire Family FPGA Security User Guide](#).

The OPTIONS parameter passed in the digest check service selects the digests to be checked.

**Table 7-2.** Digest Check Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 7-3</a> .
6:0	47H	Digest Check service command

The following table lists the digest check service mailbox format.

**Table 7-3.** Digest Check Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	2 (For PolarFire® and RT PolarFire FPGA only)	OPTIONS	Input	Digest options. See <a href="#">Table 7-4</a> .
	4 (For PolarFire SoC and RT PolarFire SoC FPGA only)			
4	4	DIGESTERR	Output	See <a href="#">Table 7-5</a> .

**Table 7-4.** OPTIONS[31:0]

OPTIONS	Name	Description
0	CHECK FABRIC	Enables fabric design digest
1	CC	Enables digest of fabric configuration parameters such as cycle count, design version, and back level protection value.
2	sNVM	Enables digest of sNVM pages marked as ROM
3	UL	Enables digest of user security segment

**Table 7-4. OPTIONS[31:0] (continued)**

OPTIONS	Name	Description
4	UKDIGEST0	Enables digest of user key segment containing SRAM-PUF data
5	UKDIGEST1	Enables digest of user key segment containing KUP (User EC key)
6	UKDIGEST2	Enables digest of user key segment containing UPK1
7	UKDIGEST3	Enables digest of user key segment containing UEK1
8	UKDIGEST4	Enables digest of user key segment containing DPK
9	UKDIGEST5	Enables digest of user key segment containing UPK2
10	UKDIGEST6	Enables digest of user key segment containing UEK2
11	UPERM	Enables digest of permanent lock security segments
12	SYS	Enables digest of factory lock segment, factory key segment in pNVM, and System Controller ROM.
13	UKDIGEST7	Enables digest of One-Way Passcode HWM (HWM) (For PolarFire® SoC and RT PolarFire SoC FPGA only)
14	ENVM	Enables digest of eNVM (For PolarFire SoC and RT PolarFire SoC FPGA only)
15	UKDIGEST8	Enables digest of MSS Boot mode Information (For PolarFire SoC and RT PolarFire SoC FPGA only)
16	UKDIGEST9	Enables digest of SNVM_RW_ACCESS_MAP (For PolarFire SoC and RT PolarFire SoC FPGA only)
17	UKDIGEST10	Enables digests of Secure Boot Image Certificate (SBIC) (For PolarFire SoC and RT PolarFire SoC FPGA only)
[31:18]	Reserved	Reserved

In PolarFire and RT PolarFire FPGA, if CHECK FABRIC is '1', the FPGA fabric is placed in suspend state and I/Os behave in same way as programming mode. Upon completion of the fabric digest, the suspend state is automatically exited. LSRAMs does not retain the user data after performing digest check on FPGA fabric. The state of the flip-flops and uSRAMs is retained. Hence, the status of the fabric digest check must be monitored by a state machine (for example, CoreABC core) implemented in the fabric. After checking the status of the fabric digest check, the state machine issues a design reset, if LSRAMs are not initialized at power-up by the System Controller. If LSRAMs are initialized at power-up by the System Controller, a device reset is need to reinitialize the LSRAMs. If design does not use any LSRAMs, no need of any reset. Use RESET\_DEVICE tamper response signal for device reset.

In PolarFire SoC and RT PolarFire SoC FPGA, if CHECK FABRIC is '1', the FPGA fabric is placed in Suspend state and I/Os behave same as programming mode. Upon completion of the fabric digest, the Suspend state is automatically exited. LSRAMs do not retain the user data after performing digest check on FPGA fabric. The status of the fabric digest check must be monitored by MSS. After checking the status of the fabric digest check, the MSS needs to issue a design reset or device reset depending on the design requirements. Use RESET\_DEVICE tamper response signal for device reset.

If CHECK FABRIC is '0', the fabric continues to operate as normal during the requested digest calculations.

If a digest mismatch occurs, DIGESTERR indicates the selected digests are in error as listed in [Table 7-5](#). A failure of any digest results in the DIGEST tamper flag being triggered. The DIGESTERR indicates zero when it is successful.

**Table 7-5. DIGESTERR[31:0]**

DIGESTERR Bit Field	Name	Description
0	FABRICERR	Fabric digest error (0 if CHECK FABRIC is '0')
1	CCERR	Fabric configuration digest error
2	SNVMERR	sNVM (ROM pages) digest error (0 if CHECKSNVM is '0')
3	ULERR	User security segment digest error
4	UKOERR	Digest error in user security segment containing SRAM-PUF data

**Table 7-5. DIGESTERR[31:0] (continued)**

DIGESTERR Bit Field	Name	Description
5	UK0ERR	Digest error in user security segment containing KUP (User EC key)
6	UK2ERR	Digest error in user security segment containing UPK1
7	UK3ERR	Digest error in user security segment containing UEK1
8	UK4ERR	Digest error in user security segment containing DPK
9	UK5ERR	Digest error in user security segment containing UPK2
10	UK6ERR	Digest error in user security segment containing UEK2
11	UPERR	Digest error in permanent security lock segments
12	SYSEERR	Digest error in factory key segment, factory lock segment, or System Controller ROM.
13	UK7ERR	Digest error in One-Way Passcode HWM (For PolarFire® SoC and RT PolarFire SoC FPGA only)
14	ENVMERR	Digest error in eNVM (For PolarFire SoC and RT PolarFire SoC FPGA only)
15	UK8ERR	Digest error in MSS Boot Information (For PolarFire SoC and RT PolarFire SoC FPGA only)
16	UK9ERR	Digest error in SNVM_RW_ACCESS_MAP (For PolarFire SoC and RT PolarFire SoC FPGA only)
17	UK10ERR	Digest error in Secure Boot Image Certificate (SBIC) (For PolarFire SoC and RT PolarFire SoC FPGA only)
[31:18]	Reserved	Reserved

## 7.2. In-Application Programming (IAP) Service [\(Ask a Question\)](#)

IAP reprograms the device with a specific programming image. In IAP, regardless of the image version, the device chooses the programming image based on either the image index or the SPI image address. The MSS specifies the programming image and initiates reprogramming of the device using the IAP system service.

The user application initiates an IAP system service request using the SPI flash image index or address. The system service specifies whether the image is used for verification or programming. The System Controller automatically reads the bitstream from the SPI Flash to verify or program the device contents.

### Verify Operation

The verify operation compares the specified programming image contents with the device contents. The following table lists the fields in an IAP system service request using the image index.

**Table 7-6. IAP Verify Request by Image Index**

System Service Descriptor Bit Field	Value	Description
15	—	Reserved.
14:7	SPI_IDX[7:0]	Identifies the image index in the SPI directory for IAP operation.
6:0	44H	IAP verify operation.

An SPI Flash memory address can be specified instead of the image index within the SPI directory, as shown in the following table.

**Table 7-7. IAP Verify Request by Image Address**

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 7-10</a> .
6:0	45H	IAP verify operation.

If the IAP verification is successful, the status code 0 is generated. If the IAP verification fails, an 8-bit error code is generated.

**➔ Important:** [Digest Check Service](#) is recommended to verify the integrity of the device contents instead of IAP verify operation. For more information, see respective [PolarFire FPGA Datasheet](#), [RT PolarFire FPGA Datasheet](#), [PolarFire SoC Datasheet](#), or [RT PolarFire SoC Datasheet](#).

## Program Operation

The program operation updates the device contents using a specified programming image. The IAP program operation does not authenticate the image before executing the program. The image can be authenticated using the IAP image authentication system service.

The user application cannot obtain the status code in the following scenarios:

- If IAP is successful, the device is automatically restarted to initialize the new design.
- If IAP fails, the IAP recovery procedure attempts to program the device with image 0.

**➔ Important:** IAP recovery considers image 0 when the pointer to image 1 in the SPI directory is null.

The following table lists the fields in an IAP system service request using the image index.

**Table 7-8.** IAP Program Request by Image Index

System Service Descriptor Bit Field	Value	Description
15	—	Reserved.
14:7	SPI_IDX[7:0]	Identifies the image index in the SPI directory for IAP operation.
6:0	42H	IAP program operation.

An SPI Flash memory address can be specified instead of the image index within the SPI directory, as specified in the following table.

**Table 7-9.** IAP Request by Image Address

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	For the mailbox format, see <a href="#">Table 7-10</a> .
6:0	43H	IAP program operation.

The following table lists the mailbox format.

**Table 7-10.** IAP Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	4	SPIADDR	Input	Programming image address in SPI Flash memory. If the attached SPI Flash device does not support 32-bit addresses, SPIADDR[31:24] is ignored.

### 7.3. Auto Update Service [\(Ask a Question\)](#)

In this service, the newest image of the first two images in the SPI directory is chosen to be programmed. If the version of the selected image is found to be different from the current programmed version, the auto update service reprograms the device with the selected image.

**Table 7-11.** Auto Update Service Request

System Service Descriptor Bit Field	Value	Description
15:7	Reserved	Reserved
6:0	46H	Auto Update service command

The user application cannot obtain the status code in the following scenarios:

- If the auto update program is successful, the device is automatically restarted to initialize the new version of the design.
- If the auto update program fails, the auto update recovery procedure attempts to program the device with the valid image again.
- If the device remains blank at the end of auto update, there is no indication through I/O and user intervention is required.

## 8. Debug Services (For PolarFire SoC and RT PolarFire SoC FPGA Only) (Ask a Question)

a Question)

The following table lists all the Debug system services with their command values, description, and return status code.

**Table 8-1.** Debug System Services

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
Probe Read Debug Service	70	Reads the content of a probe module (59 x 18-bit words).	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: SECERR (Blocked by Device security. This occurs when the permanent debug lock is set or the user software debug lock is active or the device is in the virgin state.)</li> </ul>
Probe Write Debug Service	71	Writes up to 18 bits of data to the selected probe address.	
Live Probe Debug Service	72, 73	Configures channel A or B of the live probe system.	
MEM Select Debug Service	74	Specifies a target fabric memory for the MEM read and MEM write services to access.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: SECERR (Blocked by Device security. This occurs when the permanent debug lock is set or the user software debug lock is active or the device is in the virgin state.)</li> <li>2: TIMEOUTERR</li> <li>3: LOCKERR (Target memory fail to lock.)</li> </ul>
MEM Read Debug Service	75	An interface to read data from the memory peripheral.	
MEM Write Debug Service	76	An interface to write data to the memory peripheral.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: SECERR (Blocked by Device security. This occurs when the permanent debug lock is set or the user software debug lock is active or the device is in the virgin state.)</li> <li>2: TIMEOUTERR</li> <li>3: LOCKERR (Target memory fail to lock.)</li> <li>4: AXI Error</li> <li>5: AXI Timeout Error</li> </ul>
APB Read Debug Service	77	Reads a specified number of bytes from the fabric APB bus to the specified MSS RAM area.	
APB Write Debug Service	78	Writes bytes of data to the current fabric APB address as specified by APBADDR.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: SECERR (Blocked by Device security. This occurs when the permanent debug lock is set or the user software debug lock is active or the device is in the virgin state.)</li> <li>2: SLVERR (Bus transaction error.)</li> <li>3: TIMEOUT</li> <li>4: AXI Error</li> <li>5: AXI Timeout Error</li> </ul>

**Table 8-1.** Debug System Services (continued)

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
<a href="#">Debug Snapshot Service</a>	79	Generates a snapshot of the volatile fabric content.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: SECERR (Blocked by Device security. This occurs when the permanent debug lock is set or the user software debug lock is active or the device is in the virgin state.)</li> <li>2: BUSERR (This may occur when the fabric power is off, or the fabric APB slave flagged an error, or the fabric APB slave is slow to assert PREADY Signal.)</li> </ul>
<a href="#">Terminate Debug Service</a>	7E	To end the debug session.	0: Success

### 8.1. Probe Read Debug Service [\(Ask a Question\)](#)

Reads the content of a probe module (59 x 18-bit words). The probe read debug service cannot be used when the fabric is powered down.

**Table 8-2.** Probe Read Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-3</a> .
6:0	70H	Probe read debug service command

The following table lists the Probe Read Debug service mailbox format.

**Table 8-3.** Probe Read Debug Service Mailbox Format

Offset (Byte)	Offset (Bit)	Length (Bits)	Parameter	Direction	Description
0	0	6	IPSEGADDR[5:0]	Input	Specifies a probe segment address.
0	6	5	IPROWADDR[4:0]	Input	Specifies a probe row address.
4+4*i	0	18	PRDATAi[17:0]	Output	The read data for probe word i (0 to 58) within the probe module.

### 8.2. Probe Write Debug Service [\(Ask a Question\)](#)

Writes up to 18 bits of data to the selected probe address.

**Table 8-4.** Probe Write Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-5</a> .
6:0	71H	Probe Write Debug service command

The following table lists the Probe Write Service mailbox format.

**Table 8-5.** Probe Write Service Mailbox Format

Offset (Byte)	Offset (Bit)	Length (Bits)	Parameter	Direction	Description
0	0	6	PRBADDR[5:0]	Input	Specifies one of 59 words within a probe module.
2	0	6	IPSEGADDR[5:0]	Input	Specifies the probe segment address.

**Table 8-5.** Probe Write Service Mailbox Format (continued)

Offset (Byte)	Offset (Bit)	Length (Bits)	Parameter	Direction	Description
2	6	5	IPROWADDR[4:0]	Input	Specifies the probe row address.
4	—	18	PWMASK[17:0]	Input	Specifies 18 bits of PWDATA is written to selected probe module.
8	—	18	PWDATA[17:0]	Input	Specifies the value to be written on selected probe registers. Example: If PWMASK[i] is '1', probe register i is written to the value specified by PWDATA[i].

### 8.3. Live Probe Debug Service [\(Ask a Question\)](#)

Configures channel A or B of the live probe system. A live probe is enabled by writing a local address register within one of the probe segment modules. Each probe segment module generates its own local channel A live probe outputs. The live probe outputs are combined to generate a chip-level live probe channel A signal.

The local live probe may be used internally within the user design or the global channel output may be sent to an IO. To support these options, an option is provided which clears all local channel configurations before configuring a new one.

When configuring channel A, channel B is not affected and vice-versa.

**Table 8-6.** Live Probe Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-7</a> .
6:0	72H	Live Probe Channel A service command
	73H	Live Probe Channel B service command

The following table lists the Live Probe Service mailbox format.

**Table 8-7.** Live Probe Service Mailbox Format

Offset (Byte)	Offset (Bit)	Length (Bits)	Parameter	Direction	Description
0	0	5	XADDR[4:0]	Input	Specifies the X co-ordinate within target probe module.
0	5	6	YADDR[5:0]	Input	Specifies the Y co-ordinate within target probe module.
2	0	6	IPSEGADDR[5:0]	Input	Specifies the probe segment address and the target address of probe module.
2	6	5	IPROWADDR[4:0]	Input	Specifies the probe row address and the target address of probe module.
4	0	1	CLEAR	Input	Clears the configurations of local channels A or B. If CLEAR is '1', all local channel x (the applicable channel A or B) configurations are cleared before applying the new configuration.
5	0	1	IOEN	Input	Activates the probe output pad. If IOEN is '1', the corresponding live probe output pad is activated. Note that setting IOEN to '0' does not disable the internal live probe configuration.

### 8.4. MEM Select Debug Service [\(Ask a Question\)](#)

Specifies a target fabric memory for the MEM read and MEM write services to access. A handshake mechanism is used to request access to the target memory. The memory lock can be acquired

immediately allowing multiple read/write operations to perform as one logical transaction or the lock can be acquired and released by individual read/write operations.

**Table 8-8.** MEM Select Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-9</a> .
6:0	74H	MEM select debug service command

The following table lists the MEM Select Service mailbox format.

**Table 8-9.** MEM Select Service Mailbox Format

Offset (Byte)	Offset (Bit)	Length (Bits)	Parameter	Direction	Description
0	0	3	IPBLKADDR[2:0]	Input	Specifies the block address of fabric memory.
0	3	6	IPSEGADDR[5:0]	Input	Specifies the segment address.
0	9	5	IPROWADDR[4:0]	Input	Specifies the row address of fabric memory to be accessed by MEM read and MEM write services.
2	0	3	MEMTYPE	Input	Specifies the type of fabric memory to be used for MEM read and write services.
3	0	2	MEMLOCKMODE	Input	Specifies the memory lock states for supported MEMLOCKMODE values.
4	0	13	TIMEOUT[12:0]	Input	When a lock is requested, the user design may not complete the requested handshake. To prevent the firmware from waiting indefinitely, the user must specify a timeout after which the handshake is aborted.

## 8.5. MEM Read Debug Service [\(Ask a Question\)](#)

An interface to read data from the memory peripheral that is specified. A handshake mechanism is used to request access to the target memory. The memory lock can be acquired immediately allowing multiple read/write operations to be performed as one logical transaction.

**Table 8-10.** MEM Read Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-9</a> .
6:0	75H	MEM read debug service command

The following table lists the MEM Read Service mailbox format.

**Table 8-11.** MEM Read Service Mailbox Format

Offset (Byte)	Offset (Bit)	Length (Bits)	Parameter	Direction	Description
0	0	15	MEMADDR[14:0]	Input	Sets the target address within the selected memory peripheral for subsequent MEM write and MEM read instructions.
2	0	15	NWORDS	Input	Depends on memory type. The maximum limit is the size of memory.
4	0	64	MSSADDR	Input	Specifies the MSS RAM area where to copy the MEM Read data to. Note that all accesses are done with MSS User privileges.

## 8.6. MEM Write Debug Service [\(Ask a Question\)](#)

An interface to write data to the memory peripheral. A handshake mechanism is used to request access to the target memory. The memory lock may be acquired immediately allowing multiple read/write operations to be performed as one logical transaction.

**Table 8-12.** MEM Write Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-13</a> .
6:0	76H	MEM write debug service command

The following table lists the MEM Write Service mailbox format.

**Table 8-13.** MEM Write Service Mailbox Format

Offset (Byte)	Offset (Bit)	Length (Bits)	Parameter	Direction	Description
0	0	15	MEMADDR[14:0]	Input	Sets the target address within the selected memory peripheral for subsequent MEM write and MEM read instructions.
2	0	15	NWORDS	Input	Depends on memory type. The maximum limit is the size of memory.
4	0	64	MSSADDR	Input	Specifies the MSS RAM area where to copy the MEM Write data from. Note that all accesses are done with MSS User privileges.

## 8.7. APB Read Debug Service [\(Ask a Question\)](#)

Reads a specified number of bytes from the fabric APB bus to the specified MSS RAM area. The operation makes the required number of read transactions using the selected transaction size, APBDWSIZE.

The addressed fabric peripheral generates an error or fail to respond within a user-defined window, in that case any subsequent transfers are aborted and corresponding error flags are returned.

**Table 8-14.** APB Read Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-15</a> .
6:0	77H	APB Read Debug service command.

The following table lists the APB Read Debug Service mailbox format.

**Table 8-15.** APB Read Debug Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	4	APBADDR[28:0]	Input	Specifies the target address and transfer size for the APB write and APB read operations.
4	1	APBDWSIZE[1:0]	Input	Specifies the data transfer size to be used by the APB write and APB read operations.
8	2	MAXBYTES[11:0]	Input	Calculates specified number of bytes from the fabric APB bus to the Shared Buffer. Number of bytes = MAXBYTES + 1.
16	8	MSSADDR	Input	Specifies the MSS RAM area where to copy MAXBYTES+1 of the APB Read data to. Note that all accesses are done with MSS User privileges.

## 8.8. APB Write Debug Service [\(Ask a Question\)](#)

Writes bytes of data to the current fabric APB address as specified by APBADDR. The addressed fabric peripheral generates an error or fail to respond within a user-defined window, in that case the corresponding error flags are returned.

**Table 8-16.** APB Write Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-17</a> .
6:0	78H	APB Write Debug service command

The following table lists the APB Write Debug Service mailbox format.

**Table 8-17.** APB Write Debug Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	4	APBADDR[28:0]	Input	Specifies the target address and transfer size for the APB write and APB read operations.
4	1	APBDWSIZE[1:0]	Input	Specifies the data transfer size to be used by the APB write and APB read operations.
8	2	MAXBYTES[11:0]	Input	Calculates specified number of bytes from the fabric APB bus to the Shared Buffer. Number of bytes = MAXBYTES + 1.
16	8	MSSADDR	Input	Specifies the MSS RAM area where to copy from MAXBYTES+1 of data to the APB. All accesses are done with MSS User privileges.

## 8.9. Debug Snapshot Service [\(Ask a Question\)](#)

Generates a snapshot of the volatile fabric content. The volatile fabric data is read from each LSRAM,  $\mu$ SRAM, and probe module and copied to the fabric APB debug port.

**Table 8-18.** Debug Snapshot Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 8-19</a> .
6:0	79H	Debug Snapshot service command.

The following table lists the Debug Snapshot Service mailbox format.

**Table 8-19.** Debug Snapshot Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	4	PORTADDR[28:0]	Input	Sets the address of the APB port. Bits 1:0 are ignored.
4	1	APBFASTWR	Input	Specifies the usage of fast APB protocol. If apb_fast_write is '1' during write transfers, the fast APB protocol is used and the address range is limited to PORTADDR[15:2] and PORTADDR[28:16] is ignored.

## 8.10. Terminate Debug Service [\(Ask a Question\)](#)

This service is called to end the debug session. Its purpose is to re-lock all the software based debug locks (SWL\_DEBUG) if needed and to release any memories previously locked using the MEM Select Debug Service.

**Table 8-20.** Terminate Debug Service Request

System Service Descriptor Bit Field	Value	Description
15:7	—	Reserved
6:0	7EH	System service command

## 9. Passcode Services (For PolarFire SoC and RT PolarFire SoC FPGA Only) [\(Ask a Question\)](#)

The following table lists all the Passcode system services with their command values, description, and return status code.

**Table 9-1.** Passcode System Services

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
<a href="#">Generate OTP Service</a>	7A	Sets up the device to receive a one-time passcode.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: SECERR (Blocked by Device security.)</li> <li>2: PROTOCOLERR (If an invalid key mode is specified or generation of the nonce fails, the returned nonce is 0. The operation is aborted and generates the tamper event PASSCODE_FAIL.)</li> </ul>
<a href="#">Match OTP Service</a>	7B	This service is the second part of the one-time passcode protocol.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: PROTOCOLERR (If service fails, all unlocked passcodes are re-locked and generates tamper event PASSCODE_FAIL.)</li> <li>2: MISMATCHERR (If service fails, all unlocked passcodes are re-locked and generates tamper event PASSCODE_FAIL.)</li> </ul>
<a href="#">Unlock Debug Passcode Service</a>	7C	Attempts to match the user debug pass code using the key loaded into the mailbox.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: SECERR (Blocked by Device security. Occurs when the lock is active or the permanent lock is set.)</li> <li>2: PASSCODE_ERR (If service fails, all unlocked passcodes are re-locked and generates tamper event PASSCODE_FAIL.)</li> </ul>
<a href="#">One Way Passcode Service</a>	7D	Provides a mechanism for overriding the software debug lock SWL_DEBUG without requiring any interaction with an external intelligence.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: OWPERR (If service fails, all the unlocked passcodes are re-locked and generates tamper event PASSCODE_FAIL.)</li> </ul>

### 9.1. Generate OTP Service [\(Ask a Question\)](#)

Sets up the device to receive a one-time passcode. A 128-bit nonce, NFGA, is generated and stored in volatile memory for later use in the rest of the protocol. A 128-bit user nonce, NUSER, is supplied by the user.

This service only unlocks the software debug lock SWL\_DEBUG.

**Table 9-2.** Generate OTP Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 9-3</a> .
6:0	7AH	Generate OTP service command

The following table lists the Generate OTP Service mailbox format.

**Table 9-3.** Generate OTP Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	1	KEYMODE	Input	Specifies the key mode used to transport the encrypted passcode. The KEYMODE parameter is not checked for validity until the MATCH OTP service is executed.
4	16	NUSER	Input	Specifies the user nonce, and is supplied by the user.
20	16	NFPGA	Output	The 128-bit nonce, NFPGA, is generated and stored in volatile memory for later use in the rest of the protocol.

## 9.2. Match OTP Service [\(Ask a Question\)](#)

This service is the second part of the one-time passcode protocol. Before using this service, the GENERATE OTP service must first be used to obtain a nonce, NFPGA, from the device.

**Table 9-4.** Match OTP Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 9-5</a> .
6:0	7BH	Match OTP service command

The following table lists the Match OTP Service the mailbox format.

**Table 9-5.** Match OTP Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	16	UID	Input	User ID
16	32	V	Input Output	Input: Validator Output: Set to 0 during system service execution.
48	32	OTP	Input Output	Input: One Time Passcode Output: Set to 0 during system service execution.

## 9.3. Unlock Debug Passcode Service [\(Ask a Question\)](#)

Attempts to match the user debug pass code using the key loaded into the mailbox. If the match is successful, the software debug lock SWL\_DEBUG is temporarily inactive.

**Table 9-6.** Unlock Debug Passcode Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address. See <a href="#">Table 9-7</a> .
6:0	7CH	Unlock Debug Passcode service command

The following table lists the Unlock Debug Passcode service mailbox format.

**Table 9-7.** Unlock Debug Passcode Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	32	DEBUG PASSCODE	Input Output	Input: Value of debug passcode Output: This is set to 0 during system service execution.

## 9.4. One Way Passcode Service [\(Ask a Question\)](#)

Provides a mechanism for overriding the software debug lock SWL\_DEBUG without requiring any interaction with an external intelligence. This service is similar to One-Time Passcode but has selective unlocking capability and does not require a FPGA nonce.

**Table 9-8.** One Way Passcode Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	Mailbox address.
6:0	7DH	One-way Passcode service command.

The following table lists the One Way Passcode service mailbox format.

**Table 9-9.** One Way Passcode Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	16	MSG ID	Input	Message identification
16	32	V	Input	Input Validator
48	2	BSVER	Input	Not used. Value Ignored
50	1	CTYPE	Input	
51	1	KEYMODE	Input	—
52	16	UID	Input	Not used. Value Ignored <sup>1</sup>
68	16	DSN	Input	—
84	256	USERDATA	Input	Not used. Value Ignored <sup>1</sup>
340	1	OPTIONS	Input	
341	1	RESERVED	Input	Reserved
342	2	SWVER	Input	Not used. Value Ignored <sup>1</sup>
344	8	RESERVED	Input	Reserved
352	32	HASH	Input	Hash of next 96 bytes
384	32	Plaintext Passcode	Input	Encrypted via bitstream mechanism
416	16	HWM	Input	Encrypted via bitstream mechanism
432	8	LOCKMASK	Input	Encrypted via bitstream mechanism Not used. Value Ignored <sup>1</sup> SWL_DEBUG always chosen.
440	1	PCYTP	Input	Encrypted via bitstream mechanism Not used. Value Ignored <sup>1</sup> PCTYPE=1 DPK always chosen.
441	7	RESERVED	Input	Reserved
448	32	TNEXT	Input	Not used. Value Ignored <sup>1</sup>

<sup>(1)</sup> Value of 0 must be used for value ignored.

## 10. SPI Flash Memory Read Service (For PolarFire SoC and RT PolarFire SoC FPGA Only) [\(Ask a Question\)](#)

The following table lists the SPI Flash Memory Read system service with their command values, description, and return status code.

**Table 10-1.** SPI Flash Memory Read System Services

System Service Name	SERVICECMD in Hexadecimal	Description	Return Status Code
<a href="#">SPI Copy Service</a>	50	Allows data to be copied from the SPI Flash to MSS memory.	<ul style="list-style-type: none"> <li>0: Success</li> <li>1: Device not configured for Master mode</li> <li>2: AXI Error</li> </ul>

### 10.1. SPI Copy Service [\(Ask a Question\)](#)

Allows data to be copied from the SPI Flash to MSS memory. The SPI SCK frequency is specified by a user-defined option allowing for a maximum SCK frequency of 80 MHz. A SPI Flash memory address and MSS destination address are specified using the mailbox.

**Table 10-2.** SPI Copy Service Request

System Service Descriptor Bit Field	Value	Description
15:7	MBOXADDR[10:2]	For the mailbox format, see <a href="#">Table 10-3</a> .
6:0	50H	SPI Copy Service command

The following table lists the SPI Copy Service mailbox format.

**Table 10-3.** SPI Copy Service Mailbox Format

Offset	Length (bytes)	Parameter	Direction	Description
0	8	DSTADDR	Input	MSS Destination address
8	4	SRCADDR	Input	SPI Flash address
12	4	NBYTES	Input	Number of bytes to transfer
16	1	OPTIONS	Input	OPTIONS (See <a href="#">Table 10-4</a> )

**Table 10-4.** OPTIONS[7:0]

OPTIONS	Name	Description
1:0	CLKDIV	SPI clock divider configuration. Following are the Clock Dividers and their Frequency <ul style="list-style-type: none"> <li>0: 80 MHz</li> <li>1: 40 MHz</li> <li>2: 20 MHz</li> <li>3: 13.33 MHz</li> </ul>
7:2	RESERVED	Reserved

## 11. Using System Services in PolarFire and RT PolarFire FPGA [\(Ask a Question\)](#)

In PolarFire and RT PolarFire FPGA designs, system services are implemented using the `PF_SYSTEM_SERVICES` SgCore IP and Mi-V soft processor IP (or CoreABC or custom state machine). For software implementation, the `CoreSysServices_PF` driver is provided as C source code. Using `CoreSysServices_PF` driver, the user application executes system services by providing the system service command and mailbox address. The system service request function writes service command and any input command data (depending upon the type of service) to the mailbox address. The System Controller executes the system service request and returns output data to the mailbox RAM along with the return status code. For information about `CoreSysServices_PF` driver and example SoftConsole project, see *Firmware Catalog*, which is available in the Libero SoC installation package.

To execute a system service, the `PF_SYSTEM_SERVICES` IP registers are configured as follows:

1. Writes service descriptor (service command and mailbox offset address) to the system service command register.
2. Writes input data length and mailbox write address offset to the Mailbox Write Count and Mailbox Write Address Descriptor registers.
3. Writes mailbox read count and mailbox read address offset to the Mailbox Read Count and Mailbox Read Address Descriptor registers.
4. Initiates the system service request by configuring the control register.
5. Writes command data into the Mailbox Write Data register.
6. Reads the service response from Mailbox Read Data register.
7. Reads the Status Register for return status code.

For more information about mailbox read/write communication from Fabric and the `PF_SYSTEM_SERVICE` IP registers, see [PolarFire System Services SgCore User Guide](#).

The user application configures the `CoreSysServices_PF` driver using the `SYS_init()` function and executes the system service. For example, to execute device serial number service, the `CoreSysServices_PF` driver provides the following function:

```
uint8_t
SYS_get_serial_number
(
    uint8_t * p_serial_number,
    uint16_t mb_offset
)
{
    uint8_t status = 0xFF;

    status = execute_ss_command(SERIAL_NUMBER_REQUEST_CMD,
                               (uint8_t*)0,
                               0,
                               p_serial_number,
                               SERIAL_NUMBER_RESP_LEN,
                               mb_offset,
                               0);

    return status;
}
```

Similarly, the remaining system service functions are also described in the source code of `CoreSysServices_PF` driver.

For a reference design using Mi-V soft processor, see [PolarFire FPGA System Services Application Note](#).

## 12. Using System Services in PolarFire SoC and RT PolarFire SoC FPGA (Ask a

### Question)

In PolarFire SoC and RT PolarFire SoC FPGA designs, system services are implemented using the MSS via the SCB bus. The `mss_system_services` driver configures the MSS SCB registers and initiates the system service request to the System Controller. For software implementation, the `mss_system_services` driver is provided as C source code. Using `mss_system_services` driver, the user application executes system services by providing the system service command and mailbox address. The system service request function writes service command and any input command data (depending upon the type of service) to the mailbox address. The System Controller executes the system service request and returns output data to the mailbox RAM along with the return status code. For information about `mss_system_services` driver and example SoftConsole project, see [GitHub](#).

To execute a system service, the SCB registers are configured as follows:

1. Writes SCB mailbox address with input data.
2. Writes service descriptor (service command and mailbox offset address) to the SCB control register.
3. Initiates the system service request by configuring the SCB control register.
4. Retrieves the system service response from SCB mailbox address offset when System Controller processes the data and command.
5. Reads the Status Register for return status code.

The SCB registers are described in [PolarFire SoC Register Map](#).

The user application configures the system service execution in polling mode or interrupt mode using the `MSS_SYS_service_mode()` function and executes the system service. To execute device serial number service, the `mss_system_services` driver provides the following function:

```
uint16_t
MSS_SYS_get_serial_number
(
    uint8_t * p_serial_number,
    uint16_t mb_offset
)
{
    uint16_t status = MSS_SYS_PARAM_ERR;

    if (MSS_SYS_SERVICE_INTERRUPT_MODE == g_service_mode)
    {
        status = execute_ss_interrupt_mode(
            (uint8_t)MSS_SYS_SERIAL_NUMBER_REQUEST_CMD,
            NULL_BUFFER,
            MSS_SYS_WITHOUT_CMD_DATA,
            p_serial_number,
            (uint16_t)MSS_SYS_SERIAL_NUMBER_RESP_LEN,
            mb_offset,
            MSS_SYS_COMMON_RET_OFFSET);
    }
    else
    {
        status = execute_ss_polling_mode(
            (uint8_t)MSS_SYS_SERIAL_NUMBER_REQUEST_CMD,
            NULL_BUFFER,
            MSS_SYS_WITHOUT_CMD_DATA,
            p_serial_number,
            (uint16_t)MSS_SYS_SERIAL_NUMBER_RESP_LEN,
            mb_offset,
            MSS_SYS_COMMON_RET_OFFSET);
    }
}
```

```
    return status;  
}
```

Similarly, the remaining system service functions are also described in the source code of `mss_system_services` driver.

### 13. System Controller Suspend Mode [\(Ask a Question\)](#)

PolarFire family of devices have a System Controller Suspend mode feature that can be used to force the System Controller into reset after device initialization is complete. This mode is desirable for safety critical applications to protect the device from unintended device programming or zeroization of the device due to single event upset events (SEU). When the user programs the System Controller to be in the 'Suspend mode' (using the Libero SoC tool), some device features are no longer available. The following table lists the availability of device features when the device is programmed with the System Controller Suspend mode enabled.

**Table 13-1.** PolarFire Family Device Feature Availability in System Controller Suspend Mode

Feature	System Controller Suspend Mode		Notes
	Enabled	Disabled	
<b>Programming</b>			
JTAG	Yes <sup>1</sup>	Yes	—
SPI Slave	Yes <sup>1</sup>	Yes	—
Auto-Update (POR/DEVRSTn)	Yes	Yes	Executes after power-up/DEVRSTn if the feature is enabled in the device.
Auto-Update (System Service)	No*	Yes	Auto-update requested through system services is not available in Suspend mode.
IAP	No*	Yes	
<b>System Services</b>			
Device and Design Info Services	No*	Yes	Serial Number Service, USERCODE, Design Info, and so on.
Design Programming Services	No*	Yes	Bitstream and IAP Image Authentication
Data Security Services	No*	Yes	Digital Signature, sNVM, PUF Emulation, Nonce
Zeroization	No*	Yes	—
Digest Check Service	No	Yes	—
SPI Flash Memory Read Service	No*	Yes	Only for PolarFire® SoC and RT PolarFire SoC FPGA.
Passcode Service	No*	Yes	Only for PolarFire SoC and RT PolarFire SoC FPGA.
<b>User Crypto Coprocessor</b>	Yes	Yes	—
<b>Tamper</b>			
POR Digest Checks	Yes	Yes	—
Tamper IO_Disable	Yes	Yes	—
Tamper LOCKDOWN	No*	Yes	—
Tamper RESET Device	No*	Yes	—
Tamper Slow Clock	No*	Yes	—
Tamper Flags	No <sup>2,*</sup>	Yes	—
User Voltage Detectors	Yes	Yes	—
UJTAG Sec Monitor	Yes	Yes	—
Clock Glitch Monitor	No*	Yes	—
Clock Freq Monitor	No*	Yes	—
Anti-tamper Mesh	No	Yes	—
Reset Reason	Yes <sup>3,*</sup>	Yes	—
TVS	Yes	Yes	—
<b>Debug</b>			
SmartDebug	Yes <sup>1</sup>	Yes	—
Debug System Services	No*	Yes	For PolarFire SoC and RT PolarFire SoC FPGA only.

**Notes:**

1. Requires JTAG\_TRST\_B pin to be driven logic high prior to executing operation and remain high until operation is complete.
2. When System Controller Suspend Mode is enabled, only Tamper Flag [13] (POR digest check) is available. Configure the TAMPER macro to latch the state of this output just before System Controller Suspend Mode is entered. Tamper Flags[31:14, 12:0] are unavailable when System Controller Suspend Mode is enabled.
3. When System Controller Suspend Mode is enabled, the Tamper macro RESET\_REASON[4:0] outputs are updated only during FPGA initialization before device enters System Controller Suspend Mode. The Tamper macro can be configured to latch the state of these outputs just before System Controller Suspend Mode is entered. Additionally, when System Controller Suspend Mode is enabled, only the following reset reasons are captured: POR, POR1P05, POR1P8, POR2P5, and DEVRST. See [Table 4-19](#) for more information.

A device is configured to be in Suspend mode when the System Controller Suspend mode bit is programmed into the device during the FPGA programming. Controlling the JTAG\_TRST\_B pin only affects SmartDebug, JTAG, and SPI Slave programming feature and does not affect any other features listed in [Table 13-1\\*](#).

The following points describe the System Controller Suspend Mode operation:

- To enable the System Controller suspend mode feature, the System Controller Suspend Mode Enable bit must be set in the Libero SoC design tool and the device must be programmed.
- The suspend feature is now enabled where the System Controller disables many features of the device as listed in [Table 13-1](#).
- To complete the solution, you must add a 1 k $\Omega$  pulldown resistor to the JTAG\_TRST\_B pin of the device to disable the JTAG port. This, along with the System Controller Suspend mode bit programmed into the device, forces the System Controller into an SEU protected Reset state.
- If it is required to reprogram the device or debug using the SmartDebug tool, it is first necessary to drive and maintain the JTAG\_TRST\_B pin to a logic high to take the System Controller out of the Reset state as the System Controller must be enabled for these operations. Driving JTAG\_TRST\_B = 1 does not enable all the device features but only those indicated in [Table 13-1\\*](#).

When a device is programmed with the System Controller Suspend mode enabled, at device power-up or after a DEVRSTn reset toggle, the System Controller is initially active to carry out device initialization activities. Once these activities are complete, the System Controller enters Suspend mode, provided JTAG\_TRST\_B = 0. If JTAG\_TRST\_B = 1, at this time, the System Controller is active but many System Controller managed features are unavailable as per [Table 13-1\\*](#).

**Note:** For PolarFire FPGA and SoC devices, other than RTPF500ZT, RTPFS460ZT and RTPFS160ZT, when the System Controller is forced out of suspend mode, by asserting JTAG\_TRST\_B = 1, the outputs of the PF\_INIT\_MONITOR or PFSOC\_INIT\_MONITOR macro and TAMPER macro are forced = 0\* (see the following note). Since the PF\_INIT\_MONITOR or PFSOC\_INIT\_MONITOR outputs are often used for resetting the user logic design, appropriate user design considerations must be made for this operational case.

**Note:**

- \* Except for the RTPF500ZT silicon and RT PolarFire SoC FPGAs. These devices are enhanced to allow temporary exit from Suspend mode during normal operation, for performing system services that require the system controller. The use-model for these devices operating in space flight is to maximize the time that Suspend mode is enabled during normal operation. Then, only as needed, it is possible to temporarily exit Suspend mode to run an essential system service to completion, and subsequently re-enter Suspend mode as soon as possible. Therefore, not all services in [Table 13-1](#) are suitable for use during a brief exit from Suspend mode. For example, the system controller must remain continuously active during normal operation to support some Tamper services. For more information, see [AN4903: RT PolarFire Device Family Enhancements Compared to PolarFire Devices](#).

## 14. Revision History [\(Ask a Question\)](#)

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

Revision	Date	Description
G	06/2025	The following is a summary of the changes made in this revision: <ul style="list-style-type: none"> <li>• Added support for RT PolarFire SoC throughout the document.</li> <li>• Added more details in the introduction paragraph and updated <a href="#">Table 4-12</a> in <a href="#">Read Digests Service</a> section.</li> <li>• Added more details in the introduction paragraph and updated <a href="#">Table 7-4</a> and <a href="#">Table 7-5</a> in <a href="#">Digest Check Service</a> section.</li> <li>• Updated Digest Check to "Auto Update" in the <a href="#">Auto Update Service</a> section.</li> <li>• Removed the Read eNVM service related information from the document.</li> <li>• Added a note at the end of the <a href="#">System Controller Suspend Mode</a> section to indicate that RTPF500ZT, RTPFS460ZT, and RTPFS160ZT devices support the temporary exit from Suspend mode for running system controller services.</li> </ul>
F	07/2023	The following is a summary of the changes made in this revision: <ul style="list-style-type: none"> <li>• Changed the document title to "PolarFire Family System Services".</li> <li>• Added a note about sNVM wrtie condition when there is a power cut or brownout during sNVM write. See <a href="#">Secure NVM Write Service</a>.</li> <li>• Added a note in <a href="#">Data Security Services</a> to mention that data security services are available in PolarFire/RT PolarFire/PolarFire SoC "S" version devices. Also, mentioned the data security services available in "non-S" version devices.</li> <li>• Added a note in <a href="#">Nonce Service</a> to mention the availability of nonce service in "S" and "non-S" devices.</li> </ul>
E	08/2022	The following is a summary of the changes made in this revision: <ul style="list-style-type: none"> <li>• Information about reset reason register was updated. See <a href="#">Table 4-19</a>.</li> <li>• Information about Tamper flag and reset reason was updated. See <a href="#">Table 13-1</a>.</li> </ul>
D	04/2022	In this revision, added information about System Controller Suspend Mode. See <a href="#">System Controller Suspend Mode</a> .
C	09/2021	The following is a summary of the changes made in this revision. <ul style="list-style-type: none"> <li>• Added the <a href="#">Using System Services in PolarFire and RT PolarFire FPGA</a> section.</li> <li>• Added the <a href="#">Using System Services in PolarFire SoC and RT PolarFire SoC FPGA</a> section.</li> <li>• Restructured the document.</li> </ul>
B	08/2021	Information about PolarFire FPGA System Services was merged with PolarFire SoC FPGA System Services.
A	01/2021	Document formatted to Microchip template. Document number is changed from 50200905 to DS60001677.
1.0	04/2020	Initial Revision.

## Microchip FPGA Support

Microchip FPGA products group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices. Customers are suggested to visit Microchip online resources prior to contacting support as it is very likely that their queries have been already answered.

Contact Technical Support Center through the website at [www.microchip.com/support](http://www.microchip.com/support). Mention the FPGA Device Part number, select appropriate case category, and upload design files while creating a technical support case.

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call **800.262.1060**
- From the rest of the world, call **650.318.4460**
- Fax, from anywhere in the world, **650.318.8044**

## Microchip Information

### Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at <https://www.microchip.com/en-us/about/legal-information/microchip-trademarks>.

ISBN: 979-8-3371-1427-9

### Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at [www.microchip.com/en-us/support/design-help/client-support-services](http://www.microchip.com/en-us/support/design-help/client-support-services).

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

## Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.