HB0470

Handbook

CoreSysServices v3.2

Microsemi.

Power Matters.™

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
E-mail: sales.support@microsemi.com
www.microsemi.com

About Microsemi

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

50200470. 4 3/18

Contents

Figures

Tables

1 Revision History

1.1 Revision 4.0

Updated for CoreSysServices v3.2.

1.2 Revision 3.0

Updated for CoreSysServices v3.1.

1.3 Revision 2.0

Updated for CoreSysServices v3.0.

1.4 Revision 1.0

Revision 1.0 was the first publication of this document. Created for CoreSysServices v2.0.

2 Introduction

2.1 Overview

System services are system controller actions initiated by asynchronous events from the master in the field programmable gate array (FPGA) fabric or external user interface corresponding to each service. CoreSysServices soft IP provides access to the system services. The CoreSysServices soft IP communicates with the COMM_BLK through one of the fabric interface controllers (FICs). Each system service consists of a service request phase and a response phase.

2.1.1 Service Request Phase

A service request consists of a command followed by a command-specific sub-commands (descriptor) and/or data. The sub-commands contain the address pointer (given by the user configuration) to point to the data to be read from or written to the system controller through the COMM BLK. The data corresponding to the descriptor and the input data is written directly to the embedded static random access memory (eSRAM) address space by the CoreSysServices soft IP at the configured location.

Services requiring non-trivial qualifiers operate by passing a pointer to an auxiliary data structure. The auxiliary data structure must not be modified by the user after the service has been requested (by writing the command code to the COMM_BLK FIFO).

Commands F0H to FFH are used for high priority services. If a high priority command is received during execution of a low priority command, then the low priority command is aborted.

2.1.2 Service Response Phase

Upon completion of the requested service, a response is sent back providing the status of the service. A system service response phase consists of command, status, and the descriptor (if applicable) format. The output result data is read directly from the eSRAM memory space by the CoreSysServices soft IP from the configured location.

- The system services are grouped together into the following groups of services:
- Device and Design Information Services
- Cryptographic Services
- Differential power analysis (DPA) - Resistant Key-Tree Services
- Non-Deterministic Random Bit Generator (NRBG) Services
- Zeroization Service
- Programming Services: In-application programming (IAP)
- Digest Check Service
- Tamper Service
- Elliptic Curve Services
- Physically Uncloneable function (PUF) Service
- Asynchronous Messaging Service
- Flash Freeze Service

Various configuration parameters or generics are applied to CoreSysServices IP core.

Table 1 • **System Services Command/Response Values**

Category	System Service Name	Command Value (Hex)	Response Status
Device and Design Information Services	Serial Number Service	01	0: Successful 127: MSS/HPMS memory access error (HRESP)
	USERCODE Service	04	
	Device Certificate Service	00	
	User Design Version Service	05	
	Secondary Device Certificate Service	1E	
Flash Freeze		02	0: Successful 254: Service disabled by factory security 255: Service disabled by user security
Cryptographic Services	128-bit AES Cryptographic Service	03	0: Successful 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security
	256-bit AES Cryptographic Service	06	
	SHA 256-bit AES Cryptographic Service	0A	
	HMAC Cryptographic Service	0C	
DPA Resistant Key Tree Services	Key Tree Cryptographic Service	09	0: Successful 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security
	Challenge -Response Service	0E	

***Table 1 •* System Services Command/Response Values**

Category	System Service Name	Command Value (Hex)	Response Status
PUF Services	PUFUSERAC: Create or Delete the User Activation Code	19	0: Successful 1: NVM error 2: PUF error, when creating 3: Invalid Sub-command 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security
	PUFUSERKC: Get number of Key Codes, Create Intrinsic/Extrinsic Key Codes, Export/Import All Key Codes and Delete Key Codes.	1A	0: Successful 1: NVM error 2: PUF error, when creating 3: Invalid Sub-command 4: Invalid memory address 5: Invalid hash 6: Invalid User AC 8: Incorrect key-size for renewing a KC 10: Private eNVM user digest mismatch 11: Invalid sub-command 12: DRBG error 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security
	PUFUSERKEY: Fetch a User PUF Key	1B	0: Successful 1: NVM error 2: PUF error, when creating 3: Invalid Sub-command 4: Invalid memory address 5: Invalid hash 10: Private eNVM user digest mismatch 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security
	PUFPUBLICKEY: Fetch a PUF ECC Public Key	1C	0: Successful 1: NVM error 3: Invalid Sub-command 4: Invalid memory address 10: Private eNVM user digest mismatch 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security
	PUFSEED: Get a PUF Seed	1D	0: Successful 2: PUF error, when creating 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security

Table 1 **• System Services Command/Response Values**

Category	System Service Name	Command Value (Hex)	Response Status
Elliptic Curve Services	Point Multiplication	10	0: Successful 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security
	Point Addition	11	
Non-Deterministic Random Bit Generator (NRBG) Services	Self Test Service	28	0: Successful 1: Fatal error 2: Maximum instantiations exceeded 3: Invalid handle 4: Generate request too big 5: Maximum length of additional data exceeded 127: MSS/HPMS memory access error (HRESP) 253: Not licensed 254: Service disabled by factory security 255: Service disabled by user security
	Instantiate Service	29	
	Generate Service	2A	
	Reseed Service	2B	
	Uninstantiate Service	2C	
	Reset Service	2D	
Zeroization Service		F0	No response status is sent.
Tamper Control Service	Tamper Control Service (Refer Table 10)	1F	0 is returned (No error possible)

Table 1 • **System Services Command/Response Values**

Category	System Service Name	Command Value (Hex)	Response Status
Programming Service	IAP	14	0: Successful 1-127: Authentication error 128-191: Programming Error Code 255: Service disabled by user security For more information, refer to Table 13.
	Digest Check Service	17	Digest error byte Bit[7]: SVCDISABLED 0: Digest service has been enabled by the user lock 1: Digest service has been disabled by the user lock Bit[6]: Reserved Bit[5]: ENVMUPERR (For M2S/M2GL060/090/150/T/TS devices only) 0: Private eNVM user digest check passed 1: Private eNVM user digest mismatch Bit[4]: ENVMFPERR (For M2S/M2GL060/090/150/T/TS devices only) 0: private eNVM factory digest check passed 1: private eNVM factory digest mismatch Bit[3] - SYSERR 0: No error has been detected in the System Controller ROM 1: Error has been detected in the System Controller ROM Bit[2] - eNVM1 Error (For M2S/M2GL060/090/150/T/TS devices only) 0: ENVM1 digest check passed 1: ENVM1 digest check mismatch If no digest is present then the result wll be a digest mismatch for the requested eNVM. Bit[1] - eNVM0 Error 0: ENVM0 digest check passed 1: ENVM0 digest check mismatch If no digest is present then the result shall be a digest mismatch for the requested eNVM. Bit[0] - Fabric Error 0: Fabric FPGA configuration digest check passed 1: Fabric FPGA configuration digest check mismatch

Table 1 • **System Services Command/Response Values**

Category	**System Service Name**	**Command Value (Hex)**	**Response Status**
Asynchronous Message Service	Power-on-Reset (POR) Digest Error Service	F1	Digest error byte Bit[7]: SVCDISABLED 0: Digest service has been enabled by the user lock 1: Digest service has been disabled by the user lock Bit[6]: Reserved Bit[5]: ENVMUPERR (For M2S/M2GL060/090/150/T/TS devices only) 0: Private eNVM user digest check passed 1: Private eNVM user digest mismatch Bit[4]: ENVMFPERR (For M2S/M2GL060/090/150/T/TS devices only) 0: private eNVM factory digest check passed 1: private eNVM factory digest mismatch Bit[3] - SYSERR 0: No error has been detected in the System Controller ROM 1: Error has been detected in the System Controller ROM Bit[2] - eNVM1 Error (For M2S/M2GL060/090/150/T/TS devices only) 0: ENVM1 digest check passed 1: ENVM1 digest check mismatch Bit[1] - eNVM0 Error 0: ENVM0 digest check passed 1: ENVM0 digest check mismatch If no digest is present then the result shall be a digest mismatch for the requested eNVM. Bit[0] - Fabric Error 0: Fabric FPGA configuration digest check passed 1: Fabric FPGA configuration digest check mismatch
	Tamper Detect (Refer Table 7)	80-8F	Detection of an attempt
		90-9F	Detection of a failure
		A0	Clock monitor error (For M2S/M2GL060/090/150/T/TS devices only)
		B<HFLAGS> (Refer Table 9)	Hardware monitor error detected

2.2 Key Features

CoreSysServices provide the following features:

- System services mentioned in the section Overview, page 2
- Shared request user interface for all the system services
- Advanced high-performance bus (AHB)-Lite AHBL master interface to the FIC
- Shared response user interface for all the system services
- Provides support to high priority system services over on-going low priority service

2.3 Core Version

This handbook applies to CoreSysServices version 3.2.

2.4 Supported Families

- SmartFusion®2
- IGLOO®2

Table 2 • **Supported Features**

		M2S005	**M2S060**	**M2S005S**	**M2S060T/TS**
		M2S010	**M2S090**	**M2S010S/T/TS**	**M2S090T/TS**
		M2S025	**M2S150**	**M2S025T/TS**	**M2S150T/TS**
System Service	**Device**	**M2S050**		**M2S050T/TS**	
Device and Design Information Services	Serial Number	X	X	X	X
	User Code	X	X	X	X
	Device Certificate	X	X	X	X
	User Design Version	X	X	X	X
	Secondary Device Certificate		X		X
Flash Freeze		X	X	X	X
Cryptographic Services	128-bit AES Cryptographic Service			X	X
	256-bit AES Cryptographic Service			X	X
	SHA-256 Cryptographic Service			X	X
	HMAC Cryptographic Service			X	X
DPA-Resistant Key-Tree Services	Key-Tree Cryptographic Service			X	X
	Challenge-Response Cryptographic Service			X	X
PUF Services					X
Elliptic Curve Services					X
Non-Deterministic Random Bit Generator (NRBG) Services				X	X
Zeroization Service		X	X	X	X
Programming Services:	IAP	X	X	X	X
	Digest Check Service	X	X	X	X

Table 2 • **Supported Features**

System Service	**Device**	**M2S005**	**M2S060**	**M2S005S**	**M2S060T/TS**
		M2S010	**M2S090**	**M2S010S/T/TS**	**M2S090T/TS**
		M2S025	**M2S150**	**M2S025T/TS**	**M2S150T/TS**
		M2S050		**M2S050T/TS**	
Asynchronous Messages	Power-on-Reset (POR) Digest Error Service	X	X	X	X
	Tamper Detect	X	X	X	X
Tamper Control Service		X	X	X	X

Note: 'X' denotes available. Also supports IGLOO2 (M2GLXXX) device family.

2.5 Device Utilization and Performance

Utilization and performance data is listed in Table 3 for the SmartFusion2 (M2S150TS) and IGLOO2 (M2GL150TS) device family. The data listed in this table is indicative only. The overall device utilization and performance of the core is system dependent.

Table 3 • **Device Utilization and Performance**

														Tiles				
FAMILY	**SNSERVICE**	**UCSERVICE**	**DCSERVICE**	**SECDCSERVICE**	**UDVSERVICE**	**ECCPOINTMULTSERVICE**	**ECCPOINTADDSERVICE**	**CHRESPSERVICE**	**NRBGSERVICE**	**PUFSERVICE**	**PROGIAPSERVICE**	**PROGNVMDISERVICE**	**TAMPERCONTROLSERVICE**	**Combinational**	**Sequential**	**Total**	**Utilization (%)**	**Frequency MHz)**
SmartFusion2	1	0	0	0	0	0	0	0	0	0	0	0	0	1008	390	1398	1	125
SmartFusion2	0	1	0	0	0	0	0	0	0	0	0	0	0	1008	390	1398	1	112
SmartFusion2	0	0	1	0	0	0	0	0	0	0	0	0	0	1007	391	1398	1	131
SmartFusion2	0	0	0	1	0	0	0	0	0	0	0	0	0	1003	393	1396	1	106
SmartFusion2	0	0	0	0	1	0	0	0	0	0	0	0	0	1014	390	1404	1	126
SmartFusion2	0	0	0	0	0	1	0	0	0	0	0	0	0	1008	390	1398	1	125
SmartFusion2	0	0	0	0	0	0	1	0	0	0	0	0	0	1007	390	1397	1	122
SmartFusion2	0	0	0	0	0	0	0	1	0	0	0	0	0	1239	533	1772	1	113
SmartFusion2	0	0	0	0	0	0	0	0	1	0	0	0	0	1164	453	1617	1	109
SmartFusion2	0	0	0	0	0	0	0	0	0	1	0	0	0	1477	567	2044	1	111
SmartFusion2	0	0	0	0	0	0	0	0	0	0	1	0	0	1108	474	1582	1	112
SmartFusion2	0	0	0	0	0	0	0	0	0	0	0	1	0	1145	394	1539	1	116
SmartFusion2	0	0	0	0	0	0	0	0	0	0	0	0	1	1139	391	1530	1	118
IGLOO2	1	0	0	0	0	0	0	0	0	0	0	0	0	1061	402	1463	1.01	130.1

Note: The data in this table was achieved using typical synthesis and layout settings. Frequency (in MHz) was set to 100 and speed grade was -1.

3 Functional Description

The CoreSysServices soft IP provides a user interface for each of the system services and an AHB-Lite master interface on the FIC side. The core communicates with the COMM_BLK through the FIC interface.

The CoreSysServices soft IP consists of the following sub-modules:

- User Interface Block, page 11
- Command Decoder Block, page 12
- FSM Control Block, page 12
- AHBL Master Interface Block, page 12
- Timing Diagrams, page 25

Figure 1 • **CoreSysServices Functional Block Diagram**

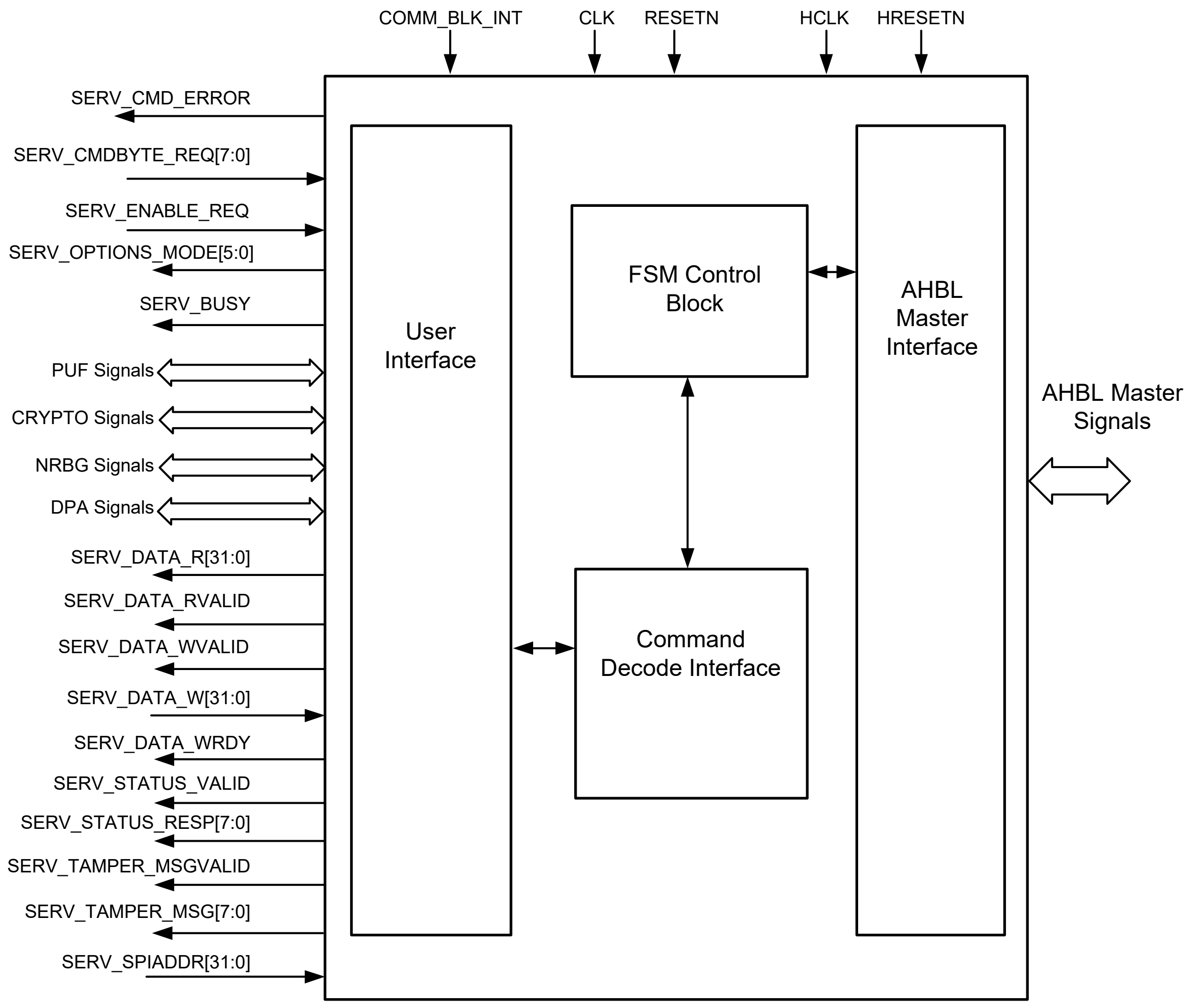
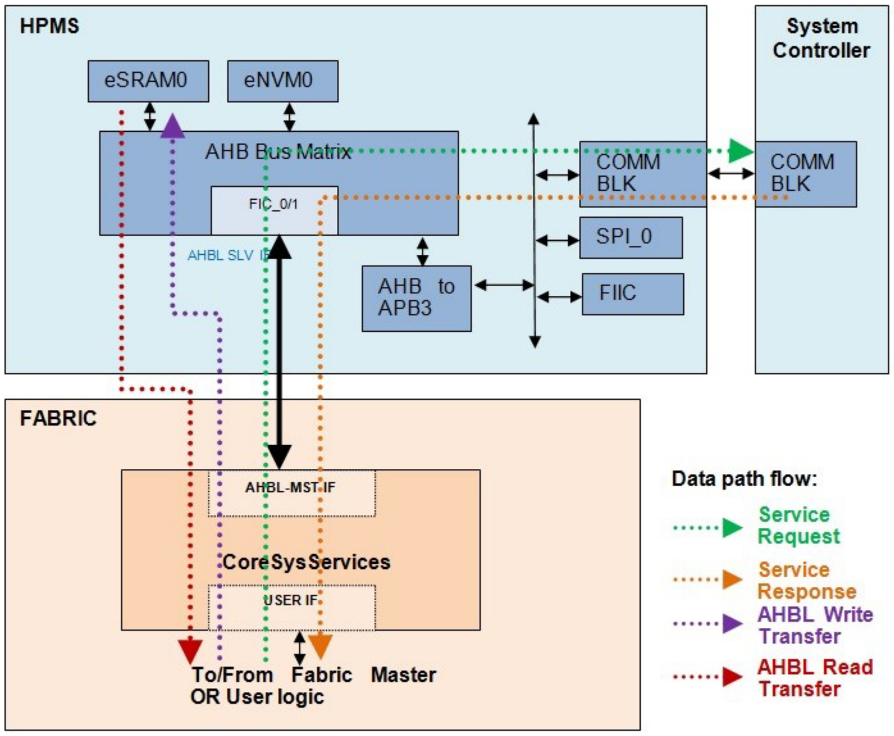


Figure 2 represents the data flow diagram for a service. This CoreSysServices data flow diagram shows the following four transactions of CoreSysServices IP:

1. Writes to eSRAM memory
2. Communicates with the system controller through the FIC and COMM_BLK (service request)
3. Communicates with the system controller through the FIC and COMM_BLK (service response)
4. Reads from the eSRAM memory

Figure 2 • **CoreSysServices Data Flow Diagram**



3.1 User Interface Block

The user interface block is the front-end to the user logic. In the service request phase, the user logic drives the system service through this interface as per Figure 3. The service-enable needs to be configured for the desired service and provide the buffer pointer if required (depends on the service).

In the service response phase, the response received by the core from the system controller is output on the SERV_STATUS_RESP [7:0] along with SERV_STATUS_VALID.

All the services are handled sequentially by the core one after another.

The service request is valid only if the SERV_BUSY from the CoreSysServices IP is low and SERV_ENABLE_REQ is HIGH. Refer to the 'Service Request' data flow as shown in Figure 2.

3.2 Command Decoder Block

The command decoder block latches the service command. It decodes and determines the type of system service requested. It generates the SERV_BUSY signal to indicate that the requested service is in progress. The core can accept a new system service request only when the SERV_BUSY signal is Low.

The command decoder block is also responsible for handling the interrupt (COMM_BLK_INT) coming from the microcontroller subsystem (MSS) in SmartFusion2 or the high-performance memory system (HPMS) in IGLOO2.

3.3 FSM Control Block

The FSM control block is the interface between the user interface on one side and the AHBL master interface on the other. It consists of the following three state machines:

- Main state machine
- Request phase state machine
- Response phase state machine

Upon receiving a valid command, it invokes the main finite state machine (FSM). The FSM control block drives the AHBL master transaction on the bus. It consists of the command byte and/or sub-command or data descriptor.

In service response phase, the status byte received on the AHBL master interface is sent back to the user interface.

3.4 AHBL Master Interface Block

This is the standard advanced microcontroller bus architecture (AMBA$^{®}$) AHB-Lite master interface to the FIC on the MSS/HPMS. The CoreSysServices soft IP communicates with the MSS/HPMS COMM_BLK through the AHBL master FIC interface. Each service requested through the user interface is translated into the corresponding AHBL master transaction and is addressed to the MSS/HPMS COMM_BLK. The FSM control block generates the transaction on the AHBL master interface.

An AHBL master write is performed to write the data descriptor and/or data to the user memory space for the system controller to access it. It also performs AHBL read transactions to the user memory space to get the result data stored by the system controller in the memory as shown in Figure 4.

This interface transmits the system service request and receives the corresponding response.

4 Core Interfaces

4.1 I/O Signals

I/O signal descriptions for CoreSysServices are listed in the below mentioned table Table 4.

Table 4 • **CoreSysServices I/O Signals and Description**

Portname	Type	Description
Clocks and Resets		
CLK	In	Fabric clock
RESETN	In	System Reset. Active low asynchronous reset
HCLK	In	AHB clock. Same as fabric clock.
HRESETN	In	AHB reset. Active low asynchronous reset
Handshaking Signals		
SERV_BUSY	Out	Service busy. When de-asserted indicates that no service has been requested. When asserted indicates that current service is in progress.
SERV_DATA_WVALID	In	Enable for User Write Data
SERV_DATA_W[31:0]	In	User Write Data
SERV_DATA_WRDY	Out	Ready to accept User Write Data
SERV_DATA_RVALID	Out	Data Valid for User Read Data
SERV_DATA_R[31:0]	Out	User Read Data
User Interface Signals: Request Signals		
SERV_ENABLE_REQ	In	Active high request to start the service
SERV_CMDBYTE_REQ[7:0]	In	Command byte to indicate the type of system service. Command byte for each of the requested System Service is described in the Core Description section.
Crypto Signals		
SERV_CRYPTO_KEY[255:0]	In	Encryption key to be used. **Note:** Only [127:0] bits are needed for the key for AES128 services.
SERV_CRYPTO_IV[127:0]	In	Initialization vector
SERV_CRYPTO_MODE[7:0]	In	Bit 7 Selects encryption or decryption • 0: input data-in is treated as plain text for encryption • 1: input data-in is treated as cipher text for decryption Bits [1:0] Selects operating mode for AES engine • 00: ECB (Electronic Code Book) • 01: CBC (Cipher-Block Chaining) • 10: OFB (Output Feedback) • 11: CTR (Counter) All other bits are unused.

Table 4 • **CoreSysServices I/O Signals and Description**

Portname	Type	Description
SERV_CRYPTO_NBLOCKS[15:0]	In	Number of 128-bit blocks to process
SERV_CRYPTO_LENGTH[31:0]	In	Length of data pointed to by DATAINPTR parameter
DPA Signals		
SERV_DPA_KEY[255:0]	In	256-bit key to be modified
SERV_DPA_OPTYPE[7:0]	In	Key tree optype parameters
SERV_DPA_PATH[127:0]	In	Path to be used
NRBG Signals		
SERV_NRBG_LENGTH[7:0]	In	Length of string in bytes
SERV_NRBG_HANDLE[7:0]	In	Handle to the DRBG instance
SERV_NRBG_ADDLENGTH[7:0]	In	Length of additional input in bytes
SERV_NRBG_PRREQ[7:0]	In	Prediction resistance request
PUF Services		
SERV_PUF_SUBCMD[7:0]	In	Sub command
SERV_PUF_INKEYNUM[7:0]	In	Key number from 0 to 57
SERV_PUF_KEYSIZE[7:0]	In	Key size 0 to 64: • 0 means 4096 bit • 1 means 64 bit • 63 means 63*64 = 4032 bit
SERV_PUFUSERKEYADDR[31:0]	In	PUF User Key Fetch address, when creating PUF User KC address, to export to or import from
SERV_USEREXTRINSICKEYADDR[31:0]	In	User Extrinsic Key address, when creating
Tamper Interface Signals		
SERV_TAMPER_MSGVALID	Out	Tamper message valid indicator (Refer Table 7)
SERV_TAMPER_MSG[7:0]	Out	Tamper message (Refer Table 7)
User Interface Signals: Response Signals		
SERV_STATUS_RESP[7:0]	Out	Response Status of the requested service. The status for each of the System Service is described in System Controller User Guide.
SERV_STATUS_VALID	Out	Response Status valid
SERV_CMD_ERROR	Out	Command error. This indicates that an unsolicited command is received during the response phase. This is output is asserted when the requested System service command opcode is not the same as that of the received response command opcode. The service is then aborted. It is de-asserted when the busy goes low.
Note: All the above request and response ports are common between all the System Services.		

Table 4 • **CoreSysServices I/O Signals and Description**

Portname	Type	Description
SERV_OPTIONS_MODE[5:0]	In	This defines the MODE options for IAP and Digest Check service. For Flash*Freeze (refer Table 11): • Bit 0: ENVM0PD • Bit 1: ENVM1PD • Bit 2: MPLLPD For Digest Check (refer Table 12): • Bit 0: Fabric • Bit 1: ENVM0 • Bit 2: ENVM1 (For M2S/M2GL060/090/150/T/TS devices only) • Bit 3: SYS • Bit 4: ENVMFP (For M2S/M2GL060/090/150/T/TS devices only) • Bit 5: ENVMUP (For M2S/M2GL060/090/150/T/TS devices only) For IAP (refer Table 13): • Bit [1:0] = 2'b00 : Authenticate • Bit [1:0] = 2'b01 : Program • Bit [1:0] = 2'b10 : Verify For TAMPERCONTROL Service (refer Table 10): • Bit 0: CLKMON_START (For M2S/M2GL060/090/150/T/TS devices only) • Bit 1: CLKMON_STOP (For M2S/M2GL060/090/150/T/TS devices only) • Bit 2: PUF_OFF (For M2S/M2GL060/090/150/T/TS devices only) • Bit 3: PUF_ON (For M2S/M2GL060/090/150/T/TS devices only) • Bit 4: CLEAR_LOCKPARITY • Bit 5: CLEAR_MESHSHORT
SERV_SPIADDR[31:0]	In	Base address of bitstream in MSS/HPMS SPI 0
Interrupts		
COMM_BLK_INT	In	Interrupt from the COMM_BLK. Active high level interrupt coming from MSS/HPMS to the fabric. This interrupt is polled in order to determine when to send the next write data to the COMM_BLK or when to read the data from the COMM_BLK. When this interrupt is asserted, the STATUS register within the COMM_BLK is read to determine the reason for the assertion of the interrupt.
AHB-Lite Master Interface Signals		
HSEL	Out	AHBL slave select - this signal indicates that the current transfer is intended for the selected slave.
HADDR[31:0]	Out	AHBL address - 32 bit address on the AHBL interface.
HWRITE	Out	AHBL write - When HIGH, this signal indicates that the current transaction is a write. When low indicates that the current transaction is a read.

Table 4 • **CoreSysServices I/O Signals and Description**

Portname	Type	Description
HTRANS[1:0]	Out	AHBL transfer type - Indicates the transfer type of the current transaction. • b00: IDLE • b01: BUSY • b10: NONSEQUENTIAL • b11: SEQUENTIAL
HSIZE[2:0]	Out	AHBL transfer size - Indicates the size of the current transfer (8/16/32/64 bit transactions only) • bx00: 8 bit (byte) transaction • bx01: 16 bit (half word) transaction • bx10: 32 bit (word) transaction • bx11: 64 bit(double word) transaction
HBURST[2:0]	Out	AHBL Burst type
HWDATA[31:0]	Out	AHBL write data - Write data from the AHBL master to the AHBL slave
HREADY	In	When HIGH, the HREADY signal indicates to the master that the previous transfer is complete.
HRESP	In	AHBL response status - When driven HIGH at the end of a transaction, this signal indicates that the transaction has completed with errors. When driven low at the end of a transaction indicates that the transaction has completed successfully.
HRDATA[31:0]	In	AHBL read data - Read data from the AHBL slave to the AHBL master

4.2 Core Parameters

4.2.1 CoreSysServices Configurable Options

There are a number of configurable options that are applied to CoreSysServices as listed in Table 5. If a configuration other than the default is required, the configuration dialog box in SmartDesign should be used to select appropriate values for the configurable options.

Note: The address range of eSRAM0 is 0x20000000 to 0x20007FFF and the address range of eSRAM1 is 0x20008000 to 0x2000FFFF.

Table 5 • **CoreSysServices Configuration Options**

Parameter Name	Valid Range	Default	Description
Device and Design Information Services			
SNSERVICE	0-1	0	Serial Number Service • 0: Disable • 1: Enable
DSNPTR	-	-	Pointer to 16-byte buffer to receive 128-bit serial number
UCSERVICE	0-1	0	User Code Service • 0: Disable • 1: Enable
USERCODEPTR	-	-	Pointer to 4-byte buffer to receive 32-bit USERCODE

Table 5 • **CoreSysServices Configuration Options**

Parameter Name	Valid Range	Default	Description
DCSERVICE	0-1	0	Device Certificate Service • 0: Disable • 1: Enable
DEVICECERTPTR	-	-	Pointer to 768-byte buffer to receive the device certificate
SECDCSERVICE	0-1	0	Secondary Device Certificate Service • 0: Disable • 1: Enable
SECONDECCCERTPTR	-	-	Pointer to 640-byte buffer to receive the device certificate
UDVSERVICE	0-1	0	User Design Version Service • 0: Disable • 1: Enable
DESIGNVERPTR	-	-	Pointer to 2-byte buffer to receive the 16-bit design version
Flash Freeze Service			
FFSERVICE	0-1	0	Flash*Freeze service • 0: Disable • 1: Enable
Cryptographic Services			
CRYPTOAES128SERVICE	0-1	0	Cryptographic Service for AES128 • 0: Disable • 1: Enable
CRYPTOAES128DATAPTR	-	-	Pointer to AES128 data descriptor in eSRAM memory space describing the transaction to be performed.
CRYPTOAES256SERVICE	0-1	0	Cryptographic Service for AES256 • 0: Disable • 1: Enable
CRYPTOAES256DATAPTR	-	-	Pointer to AES256 data descriptor in eSRAM memory space describing the transaction to be performed.
CRYPTOSHA256SERVICE	0-1	0	Cryptographic Service for SHA256 • 0: Disable • 1: Enable
CRYPTOSHA256DATAPTR	-	-	Pointer to SHA256 data descriptor in eSRAM memory space describing the transaction to be performed.
CRYPTORSLTPTR	-	-	Pointer to 32-byte buffer to receive 256-bit hash result.
CRYPTODATAINPPTR	-	-	Pointer to data to be hashed
CRYPTOHMACSERVICE	0-1	0	Cryptographic Service for HMAC • 0: Disable • 1: Enable
CRYPTOHMACDATAPTR	-	-	Pointer to HMAC data descriptor in eSRAM memory space describing the transaction to be performed.

Table 5 • **CoreSysServices Configuration Options**

Parameter Name	**Valid Range**	**Default**	**Description**
CRYPTOSRCADPTR	-	-	For AES services, pointer to source data buffer for data to be encrypted or decrypted. For SHA256 and HMAC services, pointer to data to be hashed.
CRYPTODSTADPTR	-	-	For AES services, pointer to return data buffer. For SHA256 and HMAC services, pointer to 32-byte buffer to receive 256-bit hash result.
Elliptic Curve Services			
ECCPOINTMULTSERVICE	0-1	0	Elliptic Curve Point Multiplication • 0: Disable • 1: Enable
ECCPMULTDESC	-	-	Pointer to ECCPMULT descriptor structure
ECCPMULTPPTR	-	-	Pointer to (X, Y) coordinates of P
ECCPMULTDPTR	-	-	Pointer to 384-bit scalar, d (big endian)
ECCPMULTQPTR	-	-	Pointer to (X,Y) coordinates of result Q
ECCPOINTADDSERVICE	0-1	0	Elliptic Curve Point Addition • 0: Disable • 1: Enable
ECCPADDDESC	-	-	Pointer to ECCPADD descriptor structure
ECCPADDPPTR	-	-	Pointer to (X, Y) coordinates of input point P
ECCPADDQPTR	-	-	Pointer to (X, Y) coordinates of input point Q
ECCPADDRPTR	-	-	Pointer to (X,Y) coordinates of result R
DPA Resistant Key Tree Services			
KEYTREESERVICE	0-1	0	DPA Resistant Key Tree Service • 0: Disable • 1: Enable
KEYTREEDATAPTR	-	-	Pointer to KEYTREEDATA structure
Challenge Response Services			
CHRESPSERVICE	0-1	0	DPA Resistant Pseudo PUF Challenge Response Service • 0: Disable • 1: Enable
CHRESPPTR	-	-	Pointer to a CHRESP descriptor structure
CHRESPKEYADDR	-	-	Pointer to 32-byte buffer to receive result
NRBG Services			
NRBGSERVICE	0-1	0	Non-deterministic Random Number Generator Self Test/Instantiate/ Generate/ Reseed/ Uninstantiate/Reset Service • 0: Disable • 1: Enable
NRBGINSTPTR	-	-	Pointer to DRBGINSTANTIATE structure
NRBGPERSTRINGPTR	-	-	Pointer to RBG personalization string in eSRAM address space
NRBGGENPTR	-	-	Pointer to DRBG GENERATE structure.

Table 5 • **CoreSysServices Configuration Options**

Parameter Name	**Valid Range**	**Default**	**Description**
NRBGREQDATAPTR	-	-	Pointer to buffer to receive generated random data.
NRBGRESEEDPTR	-		Pointer to DRBGRESEED structure
NRBGADDINPPTR			Pointer to additional input parameter in MSS/HPMS address space
PUF Services			
PUFSERVICE	0-1	0	Physical Uncloneable Function Create or delete UAC/ Create or delete User Key Code and Import / Export All / Fetch User PUF Key / Fetch User PUF ECC Public Key /Get a PUF seed Service • 0: Disable • 1: Enable
PUFUSERACPTR	-	-	Pointer to USERAC structure
PUFUSERKCPTR	-	-	Pointer to PUFUSERKC structure
PUFUSERKEYPTR	-	-	Pointer to PUFUSERKEY structure
PUFPUBLICKEYADDR	-	-	PUF Public Key address
PUFPUBLICKEYPTR	-	-	Pointer to PUFPUBLICKEY structure
PUFSEEDPTR	-	-	Pointer to PUFSEED structure
PUFSEEDADDR	-	-	PUF SEED address
Zeroization Service			
ZERSERVICE	0-1	0	Zeroization Service • 0: Disable • 1: Enable **Note:** Zeroization service is a high priority service.
Programming Service			
PROGIAPSERVICE	0-1	0	Programming IAP Service • 0: Disable • 1: Enable
PROGNVMDISERVICE	0-1	0	Programming NVM Digest Check Service • 0: Disable • 1: Enable
Asynchronous Messages Service			
PORDSERVICE	0-1	0	Asynchronous Messages Power-on-Reset Digest Error Service • 0: Disable • 1: Enable
TAMPERDETECTSERVICE	0-1	0	Asynchronous Messages Tamper Detect Event Service • 0: Disable • 1: Enable
TAMPERCONTROLSERVICE	0-1	0	Asynchronous Messages Tamper Control Service • 0: Disable • 1: Enable

Some System Service command includes a pointer to a buffer in the eSRAM memory space to data descriptor handle/data handle/receive result. Table 6 lists the eSRAM address space pointer for the mentioned services and their size required in terms of bytes.

Note: It is recommended to select the address pointer such that it does not overlap with address pointer of any other service. The address range of eSRAM0 is 0x20000000 to 0x20007FFF and the address range of eSRAM1 is 0x20008000 to 0x2000FFFF.

Table 6 • **eSRAM Address Space Pointer for System Service**

System Service		**Address Pointer**	**Size (bytes)**
Device and Design Information Services	Serial Number	DSNPTR	16
	User Code	USERCODEPTR	4
	Device Certificate	DEVICECERTPTR	768
	User Design Version	DESIGNVERPTR	2
	Secondary Device Certificate	SECONDECCCERTPTR	640
Cryptographic Services	128-bit AES Cryptographic Service	CRYPTOAES128DATAPTR	44
		CRYPTOSRCADPTR	16
		CRYPTODSTADPTR	16
	256-bit AES Cryptographic Service	CRYPTOAES256DATAPTR	60
		CRYPTOSRCADPTR	32
		CRYPTODSTADPTR	32
	SHA-256 Cryptographic Service	CRYPTOSHA256DATAPTR	12
		CRYPTORSLTPTR	32
		CRYPTODATAINPPTR	Upto 512Mbytes
	HMAC Cryptographic Service	CRYPTOHMACDATAPTR	44
		CRYPTODATAINPPTR	Upto 512Mbytes
		CRYPTORSLTPTR	32
DPA-Resistant Key-Tree Services	Key-Tree Cryptographic Service	KEYTREEDATAPTR	49
	Challenge-Response Cryptographic Service	CHRESPPTR	21
		CHRESPKEYADDR	32
PUF Services	User Activation Code	PUFUSERACPTR	1
	User Key Code	PUFUSERKCPTR	11
	Fetch User Key	PUFUSERKEYPTR	5
	ECC Public Key	PUFPUBLICKEYPTR	4
	Seed	PUFSEEDPTR	4
Elliptic Curve Services	Point Multiplication	ECCPMULTDESC	12
		ECCPMULTPPTR	48
		ECCPMULTDPTR	96
		ECCPMULTQPTR	96

***Table 6* • eSRAM Address Space Pointer for System Service**

System Service		**Address Pointer**	**Size (bytes)**
	Point Addition	ECCPADDDESC	12
		ECCPADDPPTR	96
		ECCPADDQPTR	96
		ECCPADDRPTR	96
Deterministic Random Bit Generator (DRBG)	Instantiate	NRBGINSTPTR	7
		NRBGPERSTRINGPTR	4-128
	Generate	NRBGGENPTR	12
		NRBGREQDATAPTR	4-128
		NRBGADDINPPTR	4-128
	Reseed	NRBGRESEEDPTR	6
		NRBGADDINPPTR	4-128
Zeroization Service		-	-
Programming Services: IAP		-	-
Digest Check Service		-	-
Asynchronous Messages	Power-on-Reset (POR) Digest Error Service	-	-
	Tamper Detect	-	-
Tamper Control Service		-	-

***Table 7* • Tamper Detect Message**

7	6	5	4	3	2	1	0	Description
1	0	0	0	CATEGORY				Attempt detected (128-143)
1	0	0	1	CATEGORY				Failure detected (144-159)
A0H								Clock monitor error (160) (For M2S/M2GL060/090/150/T/TS devices only)
1	0	1	1	0	HFLAGS			Hardware monitor error detected

***Table 8* • Tamper Detect Categories (CATEGORY)**

Value	**Name**
0	BOUNDARY_SCAN
1	BUFFER_ACCESS
2	DEBUG
3	CHECK_DIGESTS
4	ECC_SETUP_INSTRUCTION
5	FACTORY_PRIVATE
6	KEY_VALIDATION

Table 8 • **Tamper Detect Categories (CATEGORY)**

Value	**Name**
7	MISC
8	PASSCODE_MATCH
9	PASSCODE_SETUP_INSTRUCTION
10	PROGRAMMING
11	PUBLIC_INFORMATION
12	PUF_KEY_MANAGEMENT
13	UNUSED
14	USER_JTAG
15	ZEROIZATION_RECOVERY

Table 9 • **Tamper Detect Message HFLAGS**

2	1	0
MESHSHORT	LOCKPARITY	JTAGACTIVE

Table 10 • **Tamper Control Request Options**

7	6	5	4	3	2	1	0
		CLEAR_ MESHSHORT	CLEAR_LOCKPARITY	PUF_ON(For M2S/ M2GL060/090 / 150/T/TS devices only)	PUF_OFF(For M2S/ M2GL060/090 / 150/T/TS devices only)	CLKMON_STOP(For M2S/ M2GL060/090 / 150/T/TS devices only)	CLKMON_START(For M2S/ M2GL060/090 / 150/T/TS devices only)

If CLKMON_START is 1 then the clock monitor is started if the user configuration allows it (refer Table 10). Otherwise, if CLKMON_STOP is 1 then the clock monitor is stopped. Use this service in response to a clock monitor tamper detect event. (CMD=160) (Refer Table 7). When such an event is detected, and the interrupt occurred, the clock monitor detect interrupt is disabled. To enable it again, use this tamper control service with options [0] set to 1.

If PUF_OFF is 1, turn off PUF now and power it off after every key fetch, key creation or import or get seed. This is the default behavior. If PUF_ON is 1, do not power off PUF, keep it on after any key fetch, creation or import or get seed. When this service is called with PUF_ON set to 1, as the first call, one should realize that some initialization will require a puf seed, which then will turn of the PUF and keep it on from the start.

If CLEAR_LOCKPARITY is 1, then the LOCKPARITY tamper flag is cleared provided the hardware conditions are no longer present to raise this flag. If CLEAR_MESHSHORT is 1, then the MESHSHORT tamper flag is cleared provided the hardware conditions are no longer present to raise this flag.

Table 11 • **For Flash*Freeze Request Options**

7	6	5	4	3	2	1	0
					MPLLPD	ENVM1PD	ENVM0PD

If MPLLPD is ‘1’, then the MSS PLL is powered down for the duration of the FlashFreeze period.

If ENVM0PD is ‘1’, then eNVM module 0 is placed in its deep-power down state.

If ENVM1PD is ‘1’, then eNVM module 1 is placed in its deep-power down state.

If either eNVM module is not present on a device, then the value of ENVMxPD is ‘1’.

If both ENVM0PD and ENVM1PD are ‘1’, then the eNVM RC oscillator is disabled.

Table 12 • **For Digest Check Request Options**

7	6	5	4	3	2	1	0
		ENVMUP	ENVMFP	SYS	ENVM1	ENVM0	FABRIC

If FABRIC is '1', then fabric digest is checked.

If ENVM0 or ENVM1 is ‘1’, then the corresponding eNVM digests are checked.

If SYS is ‘1’, then the System Controller ROM digest is checked.

If ENVMFP is ‘1’, then the private eNVM factory digest is checked.

If ENVMUP is ‘1’, then the private eNVM user digest is checked.

Table 13 • **Programming Service Status Codes**

STATUS								Description
7	6	5	4	3	2	1	0	
0								Success
0	AUTHERRCODE							Authentication Error (refer to Table 14)
1	0	ERRORCODE						Programming Error (refer to Table 15)
255								Service disabled

Table 14 • **AUTHERRCODES**

AUTHERRCODE	**Description**
0	No error
1	Validator or hash chaining mismatch
2	Unexpected data received
3	Invalid/corrupt encryption key
4	Invalid component header
5	Back level not satisfied
6	Not Used
7	DSN binding mismatch
8	Illegal component sequence
9	Insufficient device capabilities
10	Incorrect DEVICEID
11	Unsupported bitstream protocol version (regeneration required)
12	Verify not permitted on this bitstream
13	Invalid (or inaccessible) Device Certificate
127	Abort

Table 15 • **ERRORCODES**

ERRORCODE	**Name**	**Description**
0	SUCCESS	No error encountered (ERROR=0)
1	NVMVERIFY	Fabric verification failed (min or weak limit)
2	PROTECTED	Device security prevented operation
3	NOTENA	Programming mode not enabled
4	ENVMPROG	eNVM programming operation failed
5	ENVMVERIFY	eNVM verify operation failed
6	MSSACCESS	MSS access error
7	PUFERROR	PUF access error
8	BADCOMPONENT	An internal error has been detected in a component payload

5 Timing Diagrams

A service request consists of a command followed by a command-specific sub-commands (descriptor) and/or data. The CoreSysServices IP generates the request enable (SERV_ENABLE_REQ) and command byte (SERV_CMDBYTE_REQ) to the System Controller. The sub-commands contain the eSRAM memory address pointer (given through user configuration) to point to the data to be read from or written to the System Controller through the COMM BLK. The data corresponding to the descriptor and the input data is written directly to the embedded static random access memory (eSRAM) address space by the CoreSysServices soft IP at the configured location.

Upon completion of the requested service, a response is sent back providing the status (SERV_STATUS) of the service with SERV_STATUS_VALID. A system service response phase consists of command, status, and the descriptor (if applicable) format. The output result data (SERV_DATA_R) is read directly from the configured eSRAM memory space with every read data valid signal (SERV_DATA_RVALID) by the CoreSysServices Soft IP.

The busy output (SERV_BUSY) is asserted on the clock after the request enable pulse is generated. It remains asserted until the current system service is completed.

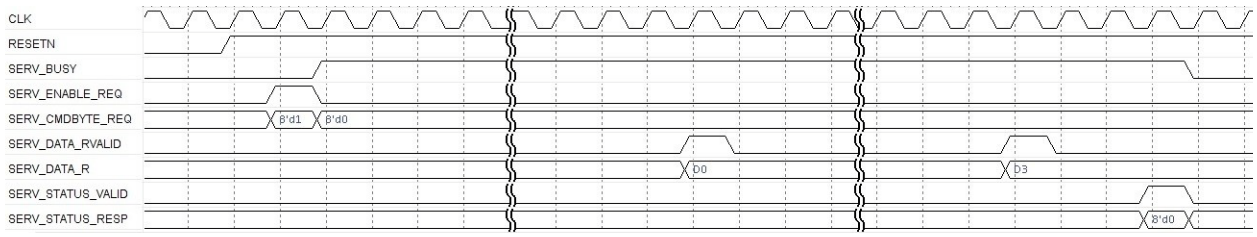
5.1 Device and Design Information Services

The device and design information services return information about the device and current user design as described below. The service request includes a service command and a pointer to a buffer in MSS/HPMS memory space to receive the result. The requested information is copied to a user-specified buffer whose address is included in the service request.

5.1.1 Serial Number

Figure 3 shows the timings for Serial Number system service

Figure 3 • **Serial Number System Service**

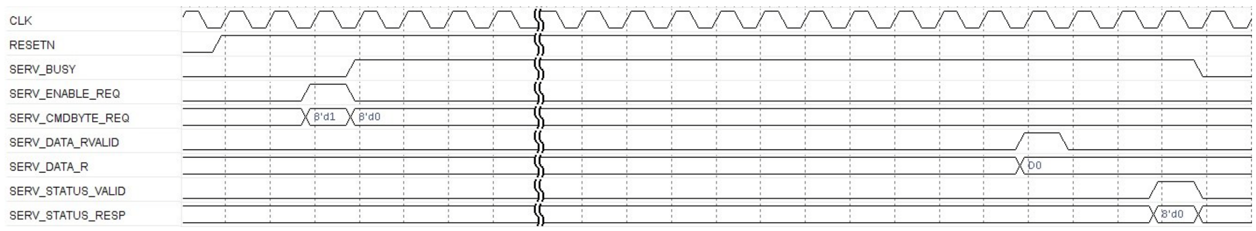


This service fetches the 128-bit device serial number (DSN). The CoreSysServices Soft IP outputs the received 128-bit Serial Number on SERV_DATA_R in terms of words (D0-D3) with every SERV_DATA_RVALID. The DSN is a 128-bit quantity unique to every device, set during manufacturing.

5.1.2 User Code

Figure 4 shows the timings for User Code System Service.

Figure 4 • **User Code System Service**

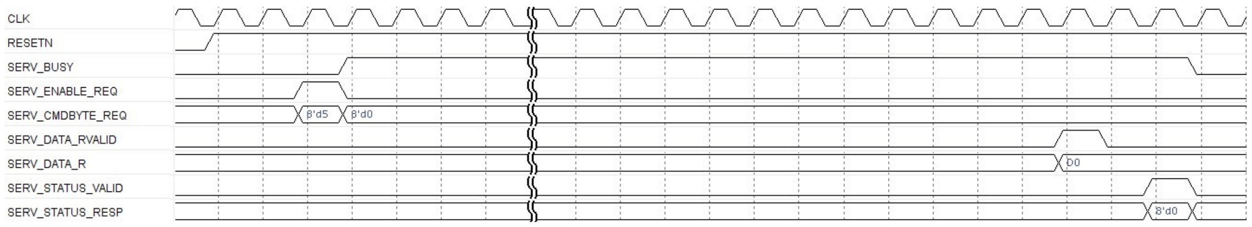


This service fetches the 32-bit JTAG USERCODE programmed by the user.

5.1.3 Design Version

Figure 5 shows the timings for Design Version System Service.

Figure 5 • **Design Version System Service**

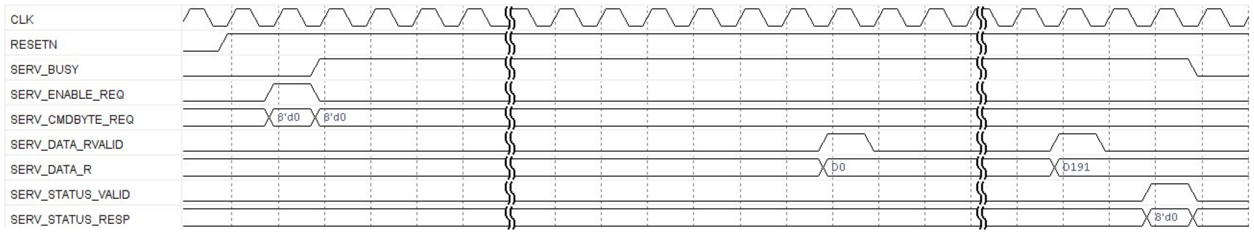


This service fetches the 16-bit user design version.

5.1.4 Device Certificate

Figure 6 shows the timings for Device Certificate System Service.

Figure 6 • **Device Certificate System Service**

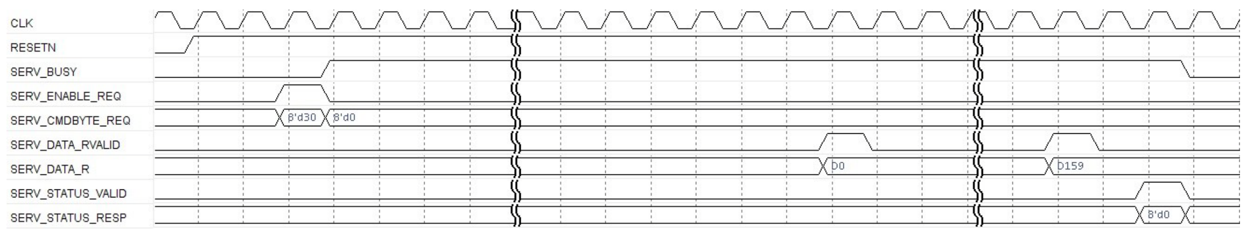


This service fetches the 768-byte device certificate from the eNVM. The device certificate is digitally-signed X-509 certificate programmed during manufacturing. The certificate is used to guarantee the authenticity of a device and its characteristics.

5.1.5 Secondary Device Certificate

Figure 7 shows the timings for Secondary Device Certificate System Service.

Figure 7 • **Secondary Device Certificate System Service**



This service fetches the 640-byte secondary device certificate from the eNVM.

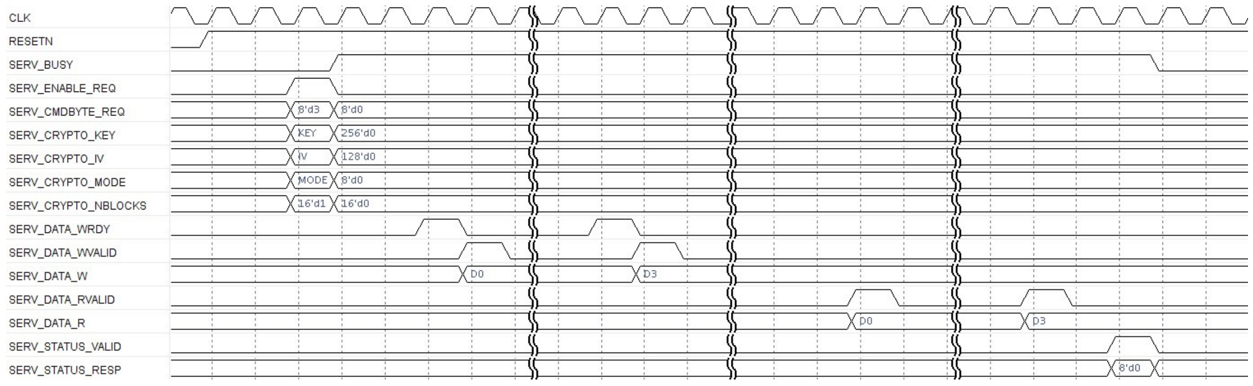
5.2 Cryptographic Services

5.2.1 AES-128

An AES service request includes a service command and a pointer to a descriptor in MSS/HPMS memory space describing the transaction to be performed. The referenced key data and plain text / cipher text is provided by the CoreSysServices IP. The resultant cipher text / plain text is copied back to the referenced MSS/HPMS memory space. The MODE parameter defined in the data descriptor specifies the cipher operating mode and whether the source text must be encrypted or decrypted. The IV parameter is a 16-byte array containing the initialization vector that will be used as a part of the requested encryption/decryption operation. Its use is different, depending on the mode. ECB mode ignores the IV parameter and CTR mode uses the content of the IV parameter as its initial counter value. NBLOCKS provides the Number of 128-bit blocks to process (max 65535).

Figure 8 shows the timings for AES-128 Cryptographic System service.

Figure 8 • **AES128 System Service**

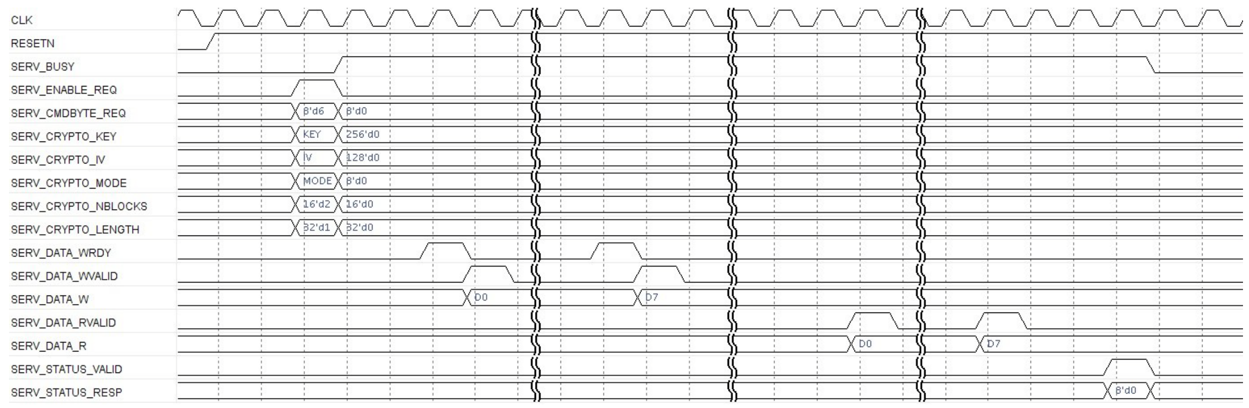


This service fetches the 128-bit AES result data from the referenced eSRAM location pointed by CRYPTODSTADPTR.

5.2.2 AES-256

Figure 9 shows the timings for AES-256 Cryptographic System service.

Figure 9 • **AES256 System Service**



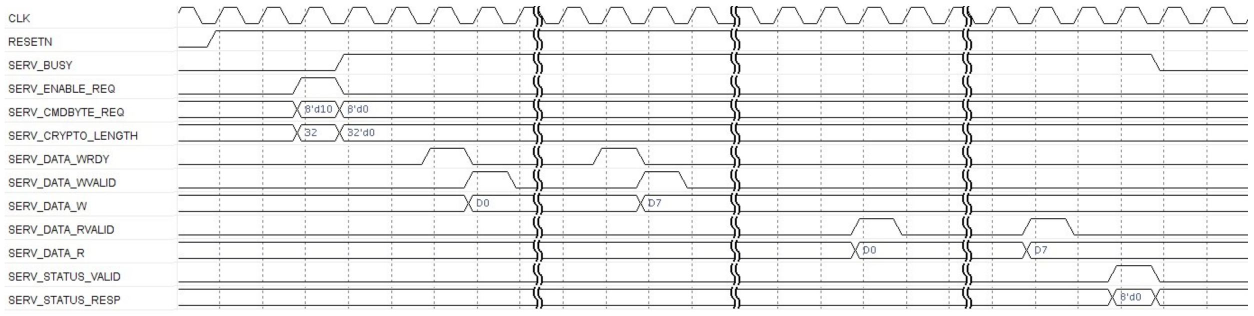
This service fetches the 256-bit AES result data from the referenced eSRAM location pointed by CRYPTODSTADPTR.

5.2.3 SHA-256

The service request includes a service command and a pointer to a descriptor in eSRAM memory space which includes the associated data.

Figure 10 shows the timings for generating SHA-256 Cryptographic System service.

Figure 10 • **SHA-256 System Service**



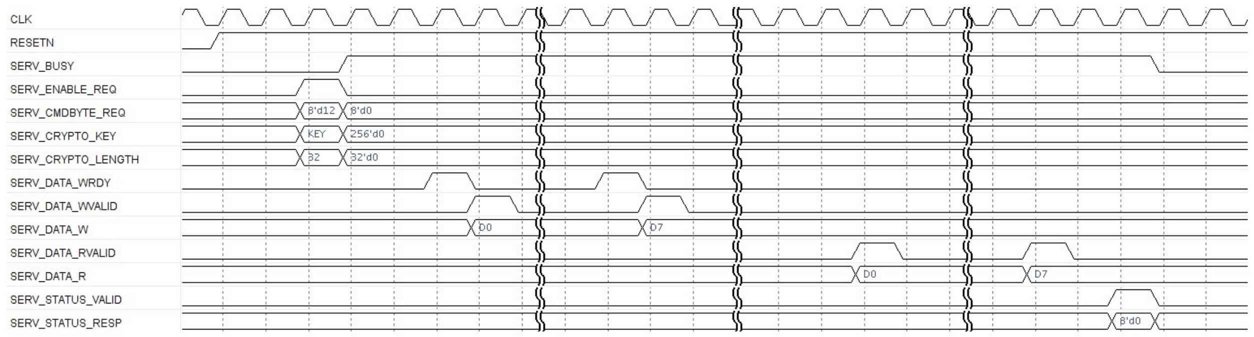
This service fetches the 256-bit SHA result data from the referenced eSRAM location pointed by CRYPTORSLTPTR.

5.2.4 HMAC

The Keyed-hash message authentication code (HMAC) service implements the FIPS 198 HMAC algorithm using SHA-256 as the approved hash function. Key length up to 32 bytes (256 bits) is used to generate the message authentication code. If the key length is less than 256 bits, the unused upper bits should be set to 0. The service allows for lengths up to 2^32 bits of data to hash.

Figure 11 shows the timings for generating HMAC Cryptographic System service.

Figure 11 • **HMAC System Service**



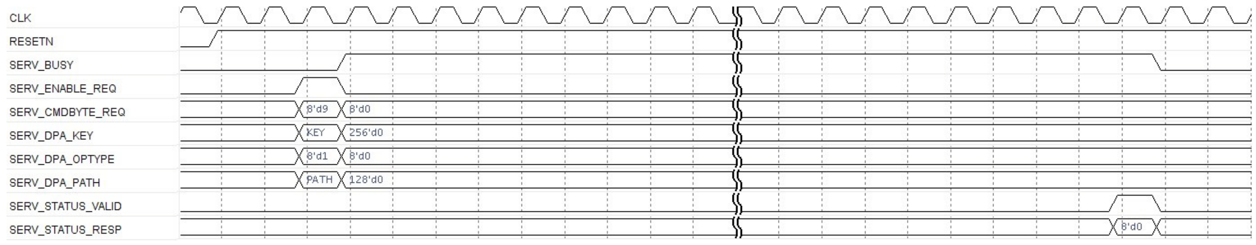
This service fetches the 256-bit HMAC result data from the referenced eSRAM location pointed by CRYPTORSLTPTR.

5.3 Key Tree Services

The generic key-tree service begins with a user-supplied root key and derives an output key based on a 7-bit parameter (SERV_DPA_OPTYPE) which can be used to create uniqueness for different applications using the same root key (32 bytes SERV_DPA_KEY), and a 128-bit path variable (SERV_DPA_PATH).

Figure 12 shows the timings for generating the DPA-Resistant Key-Tree System service.

Figure 12 • **Key-Tree System Service**

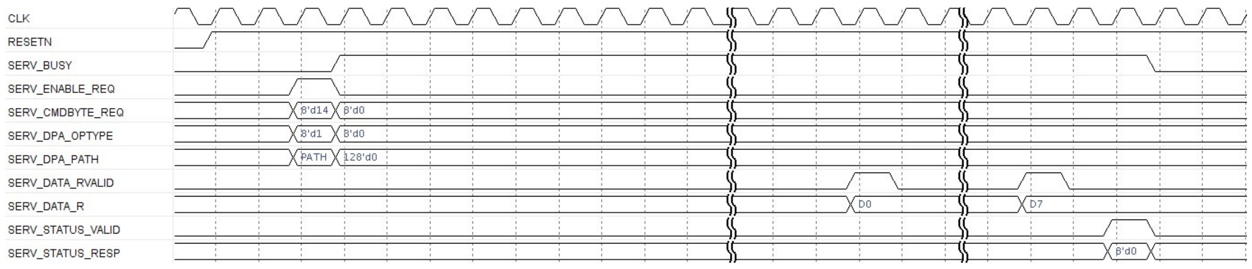


5.4 Challenge Response Service

The Challenge Response Service provides a mechanism for authenticating a device. The service accepts a challenge comprising a 7-bit OPTYPE (SERV_DPA_OPTYPE) and 128-bit path (SERV_DPA_PATH).

Figure 13 shows the timings for generating the Challenge Response System Service.

Figure 13 • **Challenge Response System Service**



This service fetches the 256 bit result data from the referenced eSRAM location pointed by CHRESPKEYADDR.

5.5 PUF Services

The PUF services provide access to the System Controller’s PUF (Physical Uncloneable Function) core. The PUF core provides services for device authentication and key generation.

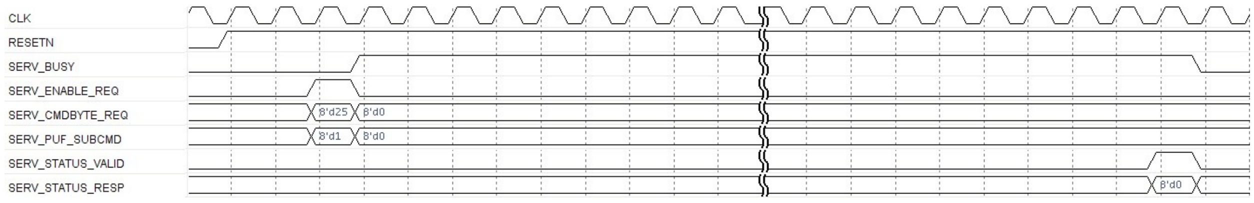
5.5.1 Create or Delete Activation Code

If SUBCMD is '0' (CREATE_AC), then the PUF core is requested to enroll a new user activation code. The 1192 byte AC is stored in eNVM.

If SUBCMD is '1' (DELETE_AC), then the user AC gets deleted together with all user keycodes and ECC public key.

Figure 14 shows the timings for generating the PUF Create/Delete Activation Code System service.

Figure 14 • **PUF AC System Service**



5.5.2 Create or Delete User Key Code and Export / Import All

Figure 15 shows the timings for generating the Create/Delete User Key Code System service.

The user key code can be generated from the PUF, using the existing activation code, stored in eNVM. The user key code is also stored in eNVM, for future use in generating user keys.

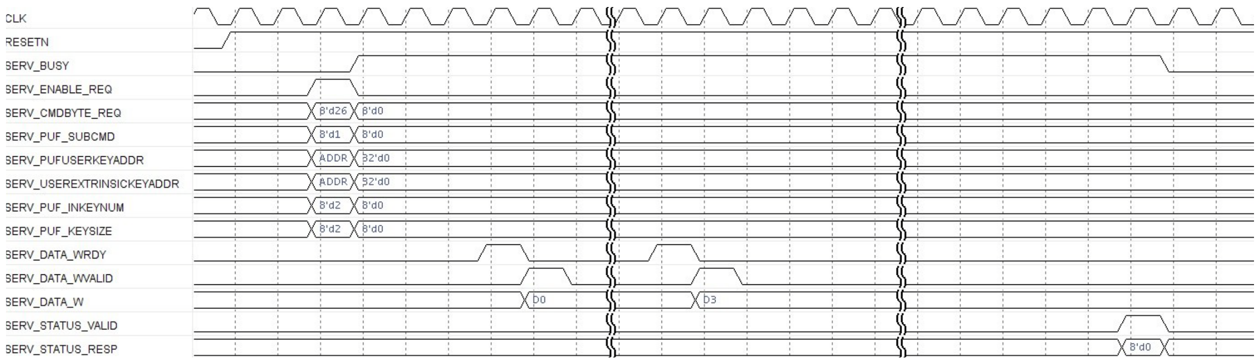
And also stored in eNVM, is an MSS address called PUFKEYADDR that is used for writing the Key to when it gets fetched.

In EXPORT_ALL and IMPORT_ALL operations we use the MSS address PUFKEYADDR to write or read a stream of AC/KC data to/from. IMPORT_ALL will also write to that same memory area the MSS addresses of keys it has generated.

If SUBCMD (SERV_PUF_SUBCMD) is ‘0’ [GET_NUMBER_OF_KC], then the total number of user keys is returned in KEYNUM. This is a number from 2 to max 58. All keys valid and invalid are counted upto the last valid key. In this context an invalid key is one that got deleted, since it is followed by a valid key.

You can only create new keys in sequence from this number onwards. So the KEYNUM returned when SUBCMD is ‘0’, is the total number of keys, and is always the max valid keynum + 1.

Figure 15 • **PUF KC System Service**



To create a KC you have to give a number (SERV_PUF_KEYNUM) 2 to 57, since 0 and 1 are reserved. Only key code 2 to 57 can be created. Keys can only be created in order. This means if the number of valid + invalid keys upto the last valid key is M, then you can request to create a key with keynum M. You can also create a key with a lower keynum. If the keysize (SERV_PUF_KEYSIZE) is the same as the original one, then the total number of valid keys will stay the same. If the keysize is different, then service is denied.

An intrinsic key is created by ignoring SERV_USEREXTRINSICKEYADDR.

An extrinsic key is created by enrolling the key pointed at by SERV_USEREXTRINSICKEYADDR.

The SERV_PUFUSERKEYADDR address is defined by the user at this time (of creation) and will be used when fetching a PUF user key. This address should be unique and thus different for each key, since when importing the keycodes, all keys are automatically generated and stored in memory pointed by these addresses.

If SUBCMD is ‘3’ [EXPORT_ALL_KC], then keycodes 0 to 57 max will be exported in encrypted form. The stored user AC and all KC’s are first xor’ed with the onetime pad and copied to a contiguous memory space, addressed by SERV_PUFUSERKEYADDR.

If SUBCMD is ‘4’ [IMPORT_ALL_KC], then the user AC and all KC’s get read from a contiguous memory space, addressed by SERV_PUFUSERKEYADDR.

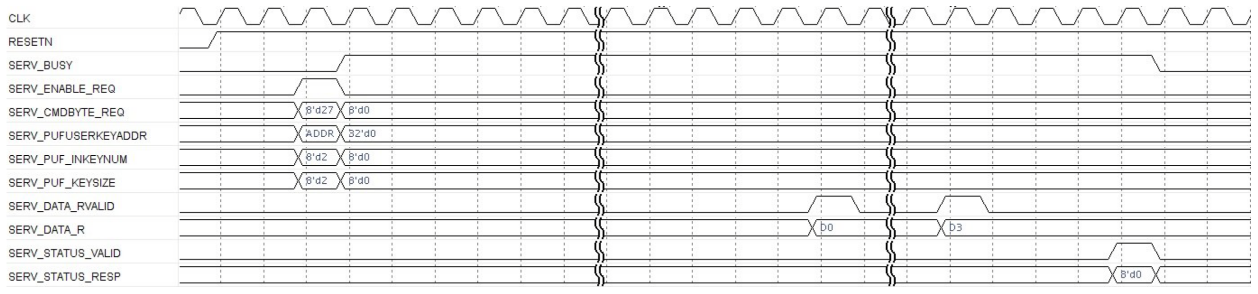
If SUBCMD is ‘5’ [DELETE_KC], then the KC corresponding to KEYNUM will be deleted. The keys 0 and 1 cannot delete.

5.5.3 Fetch User PUF Key

Figure 16 shows the timings for generating the Fetch User PUF Key System service.

The user PUF key can be generated from the PUF, using the existing activation code and key code, stored in eNVM. If successful, the user PUF key will be written at the address pointed to by PUFUSERKEYADDR. This address is returned by the service. It was defined at the time of creating the user KC.

Figure 16 • **PUF Fetch User Key System Service**

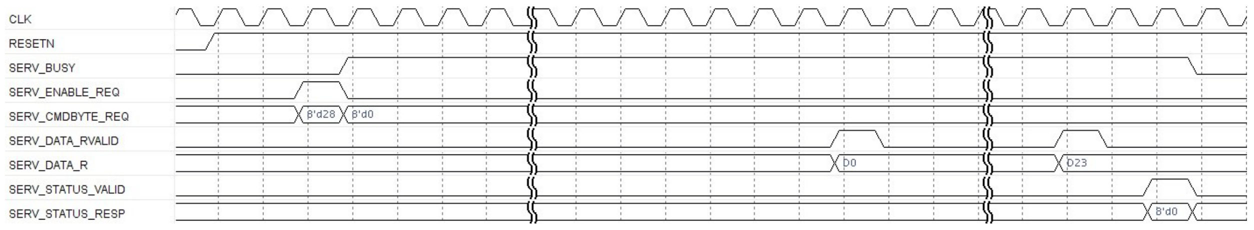


5.5.4 Fetch Public ECC Key

The PUF ECC public key can be retrieved from eNVM. If available and successful, it will be stored as 2x384bit (96 bytes) in MSS/HPMS memory pointed to by PUFPUBLICKEYADDR.

Figure 17 shows the timings for generating the PUF Fetch Public Key System service.

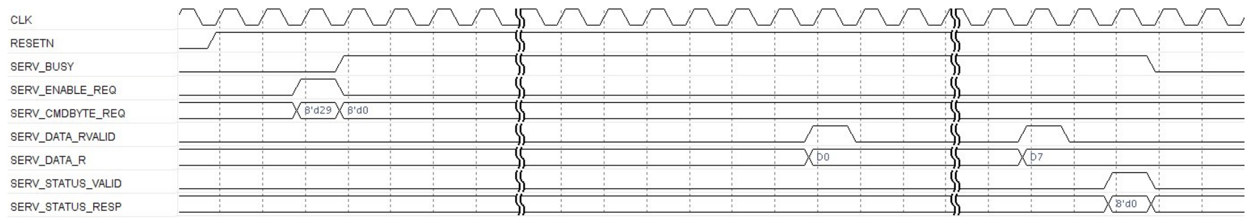
Figure 17 • **PUF Fetch Public Key System Service**



5.5.5 Get PUF Seed

Figure 18 shows the timings for generating the PUF Seed System service. The PUF core is used to generate a 256 bit seed.

Figure 18 • **PUF Seed System Service**



The PUF seed result is written at the address pointed to by PUFSEEDADDR.

5.6 NRGB Services

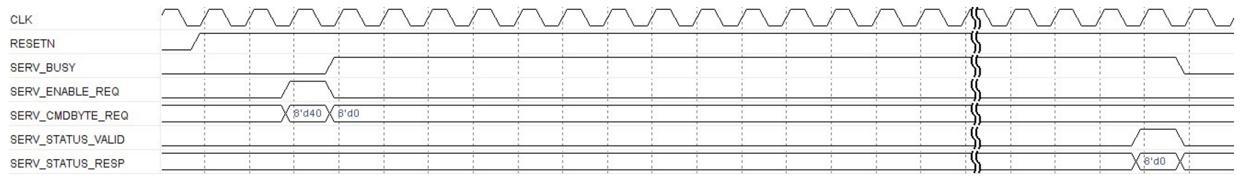
The NRBG services provide user access to the system controller deterministic random bit generator (DRBG), a part of the NRBG.

5.6.1 Self-Test

This service invokes all DRBG health tests. If any health test fails, a fatal error condition is entered. It requires device reset or user invocation of the DRBG reset service to recover from the fatal error.

Figure 19 shows the timings for generating the Self-Test System service.

Figure 19 • **DRBG Self-Test System Service**

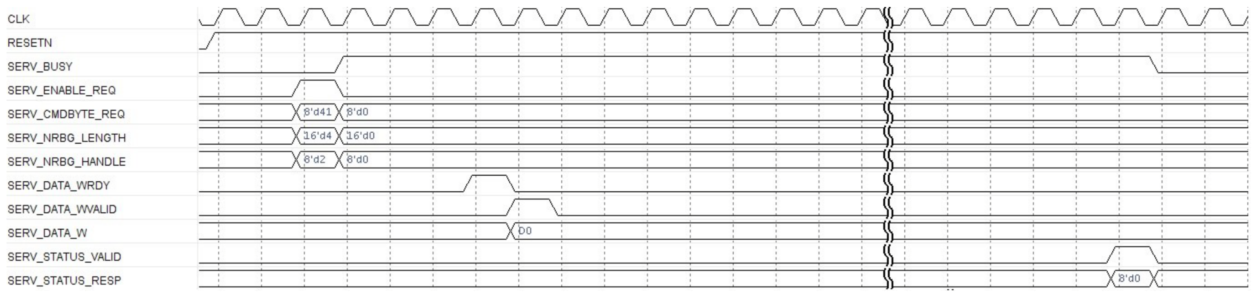


5.6.2 Instantiate

The instantiate service instantiates a DRBG instance with optional personalization string. A maximum of two concurrent user instances are available. The personalization string length must be in the range 0-128 bytes inclusive. An error will be returned from the DRBG if this field is out of range.

Figure 20 shows the timings for generating the DRBG Instantiate System service.

Figure 20 • **DRBG Instantiate System Service**

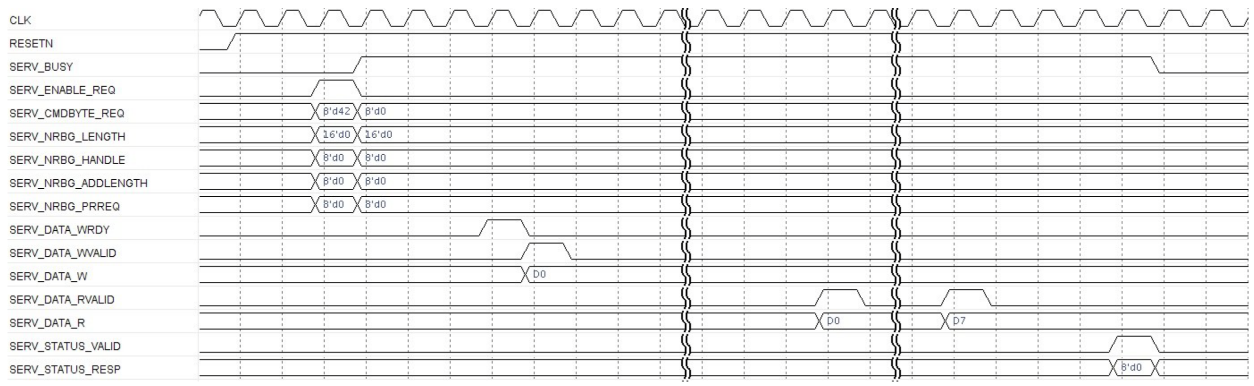


5.6.3 Generate

The generate service generates a random bit sequence up to 128 bytes long. The REQUESTEDLENGTH field must be in the range 0-128 inclusive. An error will be returned from the DRBG if this field is out of range. If PRREQ is non-zero, prediction resistance is provided.

Figure 21 shows the timings for generating the DRBG Generate System Service.

Figure 21 • **DRBG Generate System Service**

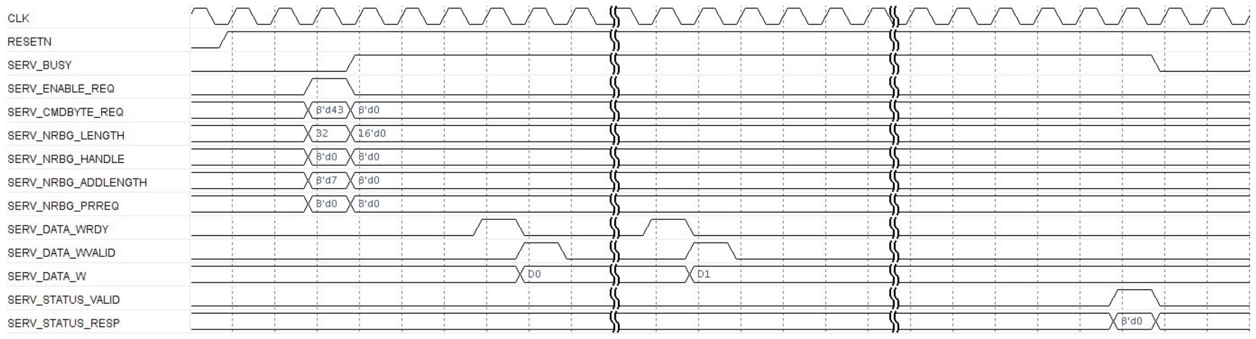


5.6.4 Reseed

Reseed service is used to reseed the random bit generator identified by the DRBG handle provided in the service request. A DRBG reseed service occurs automatically every 65,535 Generate requests. The reseed service may be used to force a reseed operation.

Figure 22 shows the timings for generating the DRBG Reseed System service.

Figure 22 • **DRBG Reseed System Service**

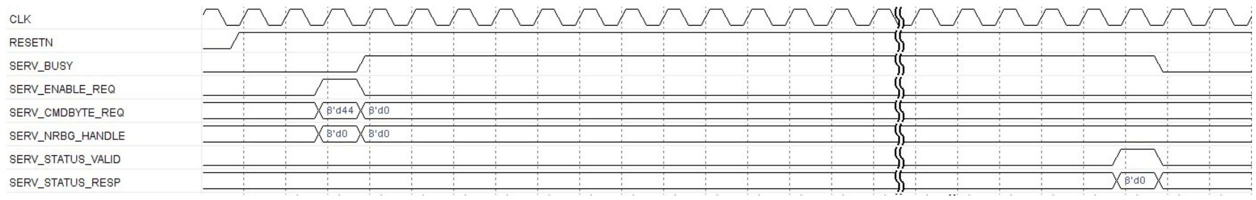


5.6.5 Uninstantiate

The uninstantiate operation removes a previously instantiated DRBG and releases the associated memory resources for later use by a new instantiation. The working state of the DRBG instantiation is zeroized before being released.

Figure 23 shows the timings for generating the DRBG Uninstantiate System service.

Figure 23 • **DRBG Uninstantiate System Service**



5.6.6 Reset

The reset service removes all DRBG instantiations and resets the DRBG. This service is the only mechanism by which to recover from a catastrophic DRBG error without physically resetting the device. All active instantiations are automatically destroyed.

Figure 24 • **DRBG Reset System Service**

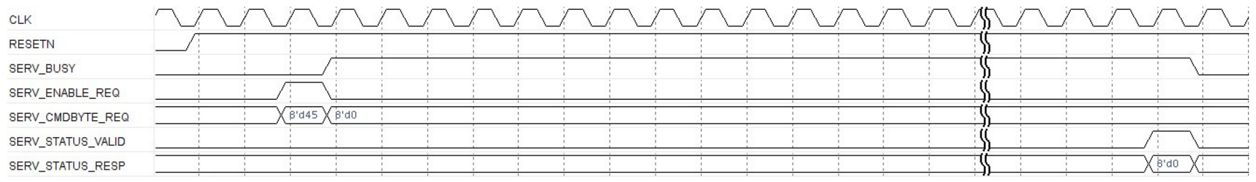


Figure 24 shows the timings for generating the request enable (SERV_ENABLE_REQ) and command byte (SERV_CMDBYTE_REQ) corresponding to DRBG Reset System service.

5.7 Elliptic Curve Services

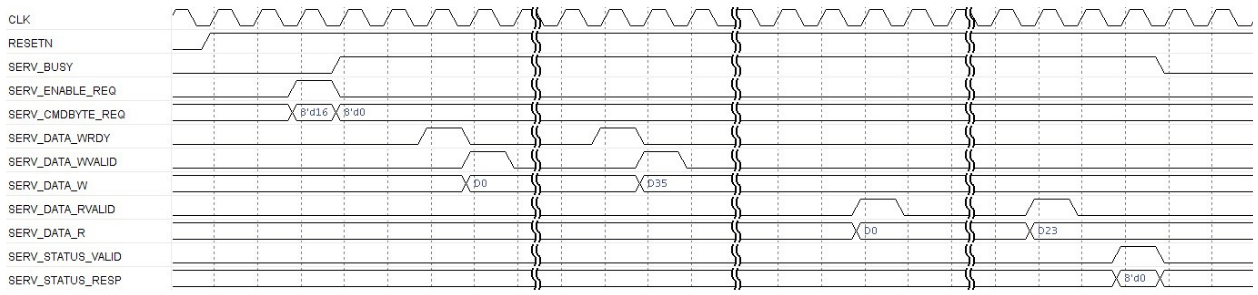
The Elliptic curve services provide access to the System Controller's Elliptic Curve Cryptography (ECC) core.

Points on the curve are encoded as two 384-bit big-endian numbers, (X, Y), stored in a two consecutive blocks of 48-bytes, with the X coordinate first

5.7.1 Point Multiplication

Figure 25 shows the timings for generating the ECC Point Multiplication System service.

Figure 25 • **ECC Point Multiplication System Service**

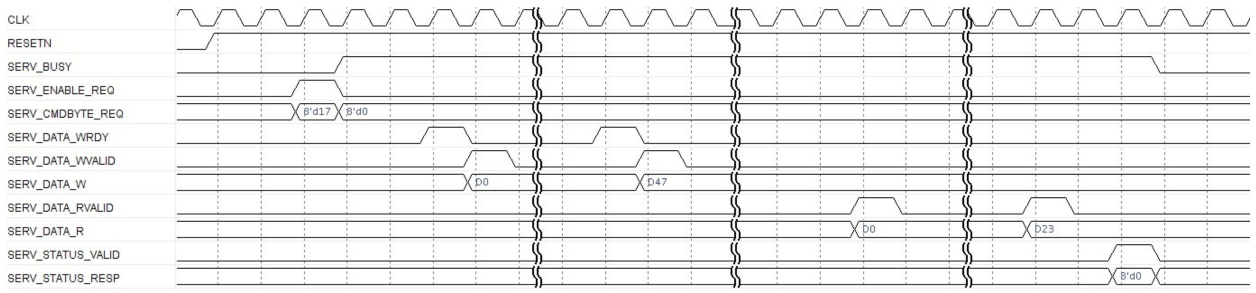


This service returns the two 384-bit (X, Y) coordinates of the result Q.

5.7.2 Point Addition

Figure 26 shows the timings for generating the ECC Point Addition System service.

Figure 26 • **ECC Point Addition System Service**



This service returns the two 384-bit (X, Y) coordinates of the result Q.

5.8 Programming Services

5.8.1 IAP

The IAP Service requests the System Controller to reprogram the device using a bitstream already programmed into MSS/HPMS SPI. The SPI peripheral must be configured by the user before initiating this request. At the time the request is initiated the user must guarantee exclusive access of SPI.

The system controller will perform one of the following operations based on the mode (SERV_OPTION_MODE) selected when invoking the service:

1. Authenticate the fabric design.
2. Program the fabric design into FPGA.
3. Verify that the programming was successful.

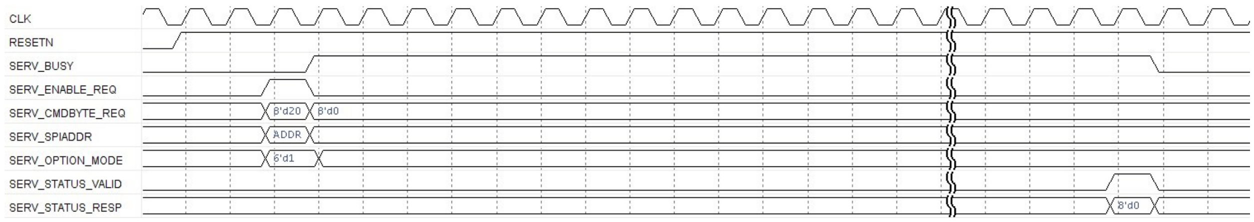
The base address of bitstream in MSS/HPMS SPI must be provided on SERV_SPIADDR. The system controller will return a response indicating the status of the operation on completing the execution of the

IAP service except in the case of successful programming. The system controller will reset the device and start execution of the new FPGA fabric design after successfully programming the device.

The system controller will not return any response information when a successful programming operation completes.

Figure 27 shows the timings for generating the IAP System service.

Figure 27 • **IAP System Service**



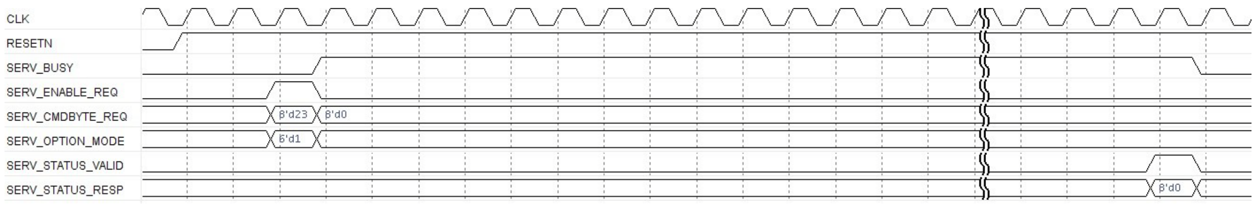
5.8.2 Digest Check

The NVM data integrity check service recalculates and compares cryptographic digests of the selected NVM component(s)-fabric, ENVM0, and ENVM1- to those previously computed and saved in NVM. The results are compared to values stored in dedicated nonvolatile memory words located in each segment. If the contents are unchanged from when the digests were computed and stored during the original programming steps-that is, if the current and stored digests match-the test will pass; otherwise a failure is flagged.

The OPTIONS (SERV_OPTION_MODE) fields in the NVM data integrity check service request selects the NVM components (FABRIC, SYS, ENVM0, ENVM1, ENVMFP and ENVMFP configuration) for data integrity check.

Figure 28 shows the timings for generating the Digest Check System service.

Figure 28 • **Digest Check System Service**



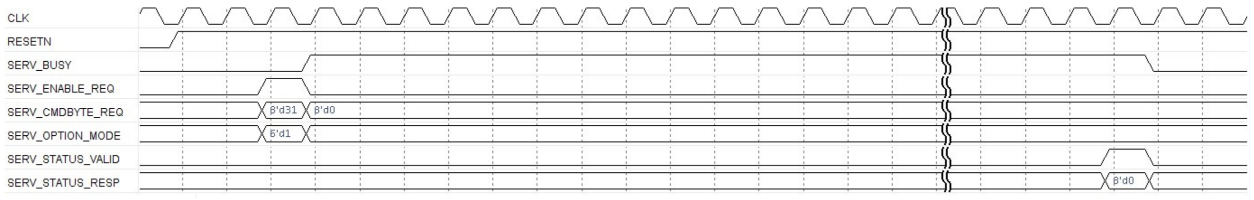
5.9 Tamper Control Service

The tamper control system services provide the following option mode:

- Enable/disable clock monitoring
- Control power to PUF
- Clear mesh short tamper
- Clear lock parity tamper

Figure 29 shows the timings for generating the Tamper Control System service.

Figure 29 • **Tamper Control System Service**



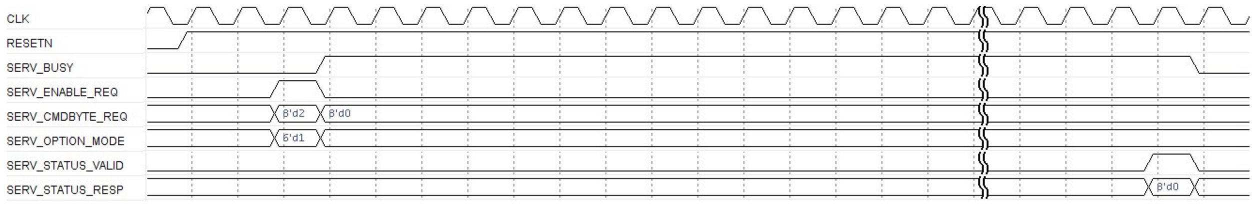
5.10 Flash Freeze Service

The Flash Freeze System service requests the system to enter into Flash*Freeze mode. Following F*F options are used along with the request command.

- ENVM0PD
- ENVM1PD
- MPLLPD

Figure 30 shows the timings for generating the Flash Freeze System service.

Figure 30 • **Flash Freeze System Service**



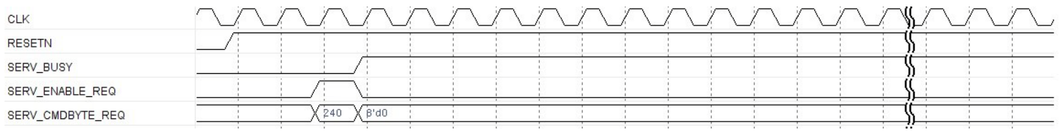
When the FlashFreeze shutdown sequence is complete the CoreSysServices IP responds with a standard service response as mentioned in Table 1.

5.11 Zeroization Service

Zeroization is a high priority service which destroys sensitive information on the device.

Figure 31 shows the timings for generating the request Zeroization System service.

Figure 31 • **Zeroization System Service**



5.12 Asynchronous Message Service

The System Controller sends asynchronous messages to the MSS/HPMS COMM_BLK when certain events are detected during the execution of the following system services:

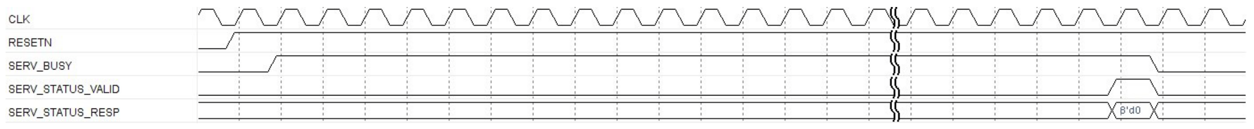
- Flash*Freeze
- Power-on-reset (POR) digest check
- Tamper detect events

Power-On-Reset Digest Error

The POR digest check service is enabled in the CoreSysServices IP and if enabled is automatically performed as part of the device's power up sequence. The System Controller sends a POR digest check error message to the MSS/HPMS COMM_BLK when the result of the POR digest check is a mismatch between the original stored digest and the current digest. The System Controller passes the command byte and the error flags byte from the error message and is output on the SERV_STATUS_RESP along with SERV_STATUS_VALID.

Figure 32 shows the timings for generating the Power-On-Reset Digest System service.

Figure 32 • **Power-On-Reset Digest System Service**

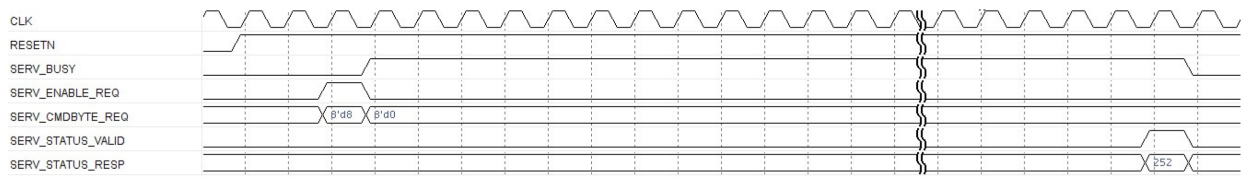


5.13 Unrecognized Command

If an unrecognized command is received, the system controller generates a response message to indicate an unrecognized command and is output on the SERV_STATUS_RESP along with SERV_STATUS_VALID.

Figure 33 shows the timings for generating the Unrecognized System service.

Figure 33 • **Unrecognized Command**



6 Register Map and Descriptions

CoreSysServices does not contain any registers.

7 Tool Flows

7.1 License

No license is required to use CoreSysServices.

7.1.1 RTL

Complete RTL source code is provided for the core and test bench.

7.2 SmartDesign

CoreSysServices is preinstalled in the SmartDesign IP Deployment design environment. An example instantiated view is shown in Figure 5. The core can be configured using the configuration GUI within SmartDesign, as shown in Figure 6.

For more information on using SmartDesign to instantiate and generate cores, refer to the Using DirectCore in Libero SoC Use Guide or consult the Libero SoC online help.

Figure 34 • **SmartDesign CoreSysServices Instance View**

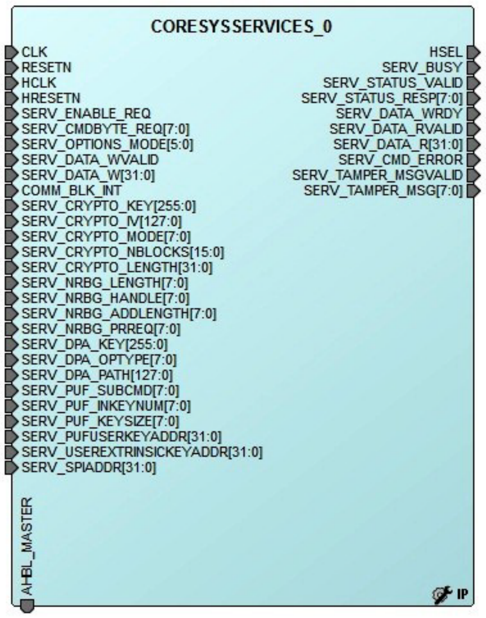
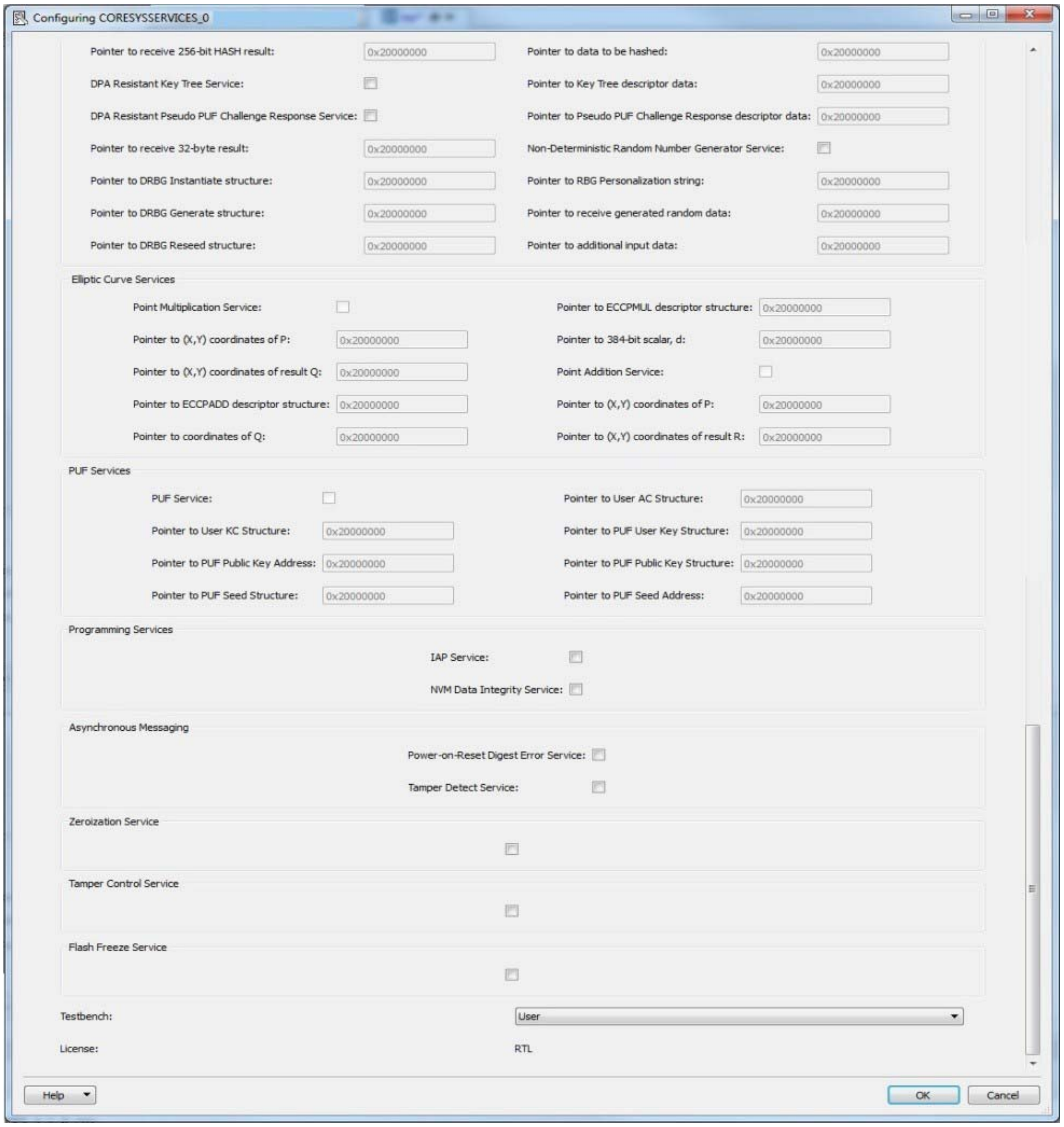


Figure 35 • **Figure 2 SmartDesign CoreSysServices Configuration Window**


Configuring CORESYSSERVICES_0

Pointer to receive 256-bit HASH result:	0x20000000	Pointer to data to be hashed:	0x20000000
DPA Resistant Key Tree Service:	☐	Pointer to Key Tree descriptor data:	0x20000000
DPA Resistant Pseudo PUF Challenge Response Service:	☐	Pointer to Pseudo PUF Challenge Response descriptor data:	0x20000000
Pointer to receive 32-byte result:	0x20000000	Non-Deterministic Random Number Generator Service:	☐
Pointer to DRBG Instantiate structure:	0x20000000	Pointer to RBG Personalization string:	0x20000000
Pointer to DRBG Generate structure:	0x20000000	Pointer to receive generated random data:	0x20000000
Pointer to DRBG Reseed structure:	0x20000000	Pointer to additional input data:	0x20000000

Elliptic Curve Services

Point Multiplication Service:	☐	Pointer to ECCPMUL descriptor structure:	0x20000000
Pointer to (X,Y) coordinates of P:	0x20000000	Pointer to 384-bit scalar, d:	0x20000000
Pointer to (X,Y) coordinates of result Q:	0x20000000	Point Addition Service:	☐
Pointer to ECCPADD descriptor structure:	0x20000000	Pointer to (X,Y) coordinates of P:	0x20000000
Pointer to coordinates of Q:	0x20000000	Pointer to (X,Y) coordinates of result R:	0x20000000

PUF Services

PUF Service:	☐	Pointer to User AC Structure:	0x20000000
Pointer to User KC Structure:	0x20000000	Pointer to PUF User Key Structure:	0x20000000
Pointer to PUF Public Key Address:	0x20000000	Pointer to PUF Public Key Structure:	0x20000000
Pointer to PUF Seed Structure:	0x20000000	Pointer to PUF Seed Address:	0x20000000

Programming Services

IAP Service: ☐

NVM Data Integrity Service: ☐

Asynchronous Messaging

Power-on-Reset Digest Error Service: ☐

Tamper Detect Service: ☐

Zeroization Service

☐

Tamper Control Service

☐

Flash Freeze Service

☐

Testbench: User

License: RTL

7.3 Simulation Flows

To run simulations, select the required Test bench flow within SmartDesign and Save and Generate on the Generate pane. The required testbench is selected through the core configuration GUI in SmartDesign. The following simulation environments are supported:

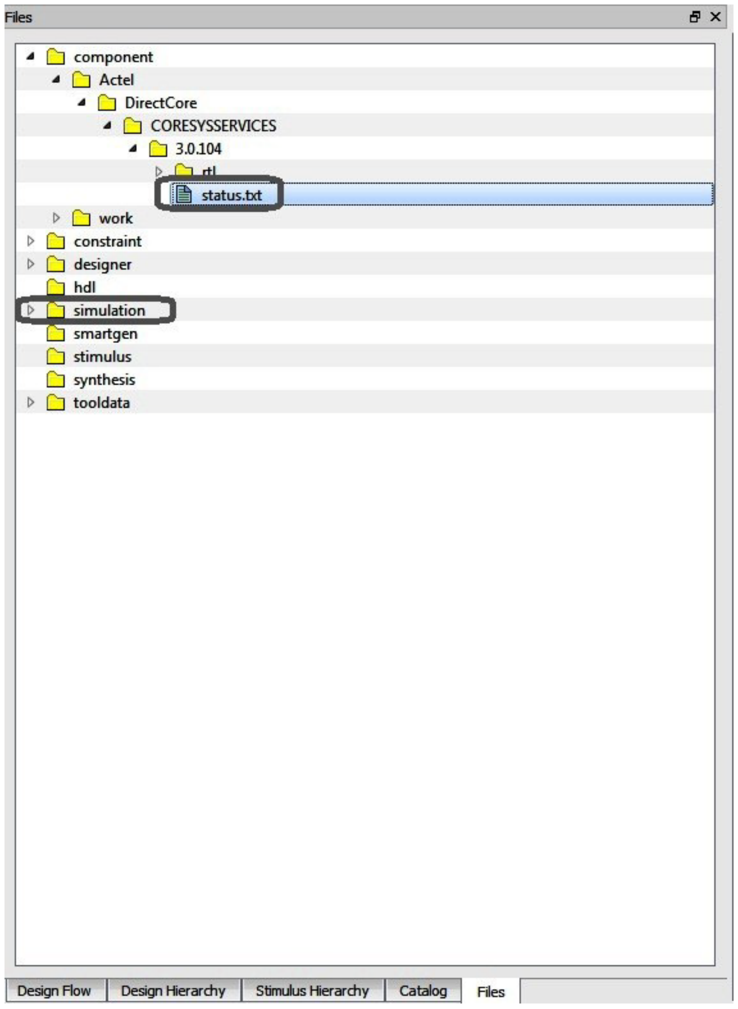
- Full verification environment
- Simple user testbench

When SmartDesign generates the Libero SoC project, it installs the user testbench files.

To run the User/Verif Testbench, set the design root to the CoreSysServices instantiation in the Libero SoC design hierarchy pane and click the Simulation icon in the Libero SoC design flow window. This invokes ModelSim® and automatically run the simulation.

Note: To run the full verification testbench, copy the Libero generated status.txt (*/component/Actel/DirectCore/<version_no>/status.txt*) file in to Simulation directory in the Libero SoC 'files' pane as shown in the Figure 36.

Figure 36 • **Simulation Directory in the Libero SoC files Pane**



7.4 Synthesis in Libero

To run synthesis on the CoreSysServices, set the design root to the IP component instance and run the synthesis tool from the Libero design flow pane.

7.5 Place-and-Route in Libero

After the design is synthesized, run the compilation and then place-and-route the tools.

CoreSysServices requires no special place-and-route settings.

8 References

For more information, refer to the following:

- ARM Cortex-M3 Processor and Subsystem in SmartFusion2 Devices User Guide
- UG0450: SmartFusion2 and IGLOO2 System Controller User Guide
- UG0448: IGLOO2 FPGA High Performance Memory Subsystem User Guide

9 Ordering Information

9.1 Ordering Codes

CoreSysServices can be ordered through the local sales representative. It should be ordered using the following number scheme: CoreSysServices -XX, where XX is listed in

Table 16 • **Ordering Codes**

XX	**Description**
RM	RTL for RTL sources-multiple use license