P10 SBE PSU Interface Specification

Version 1.0

Owner: Amit J Tendolkar

amit.tendolkar@in.ibm.com

1 Document Control Section

1.1 Document Location

This document is shared at https://github.com/open-power/docs/sbe

The reader (possessor of this document) is responsible for verification that this is the latest version of the document.

1.2 Document Completeness

The final page of this document contains the text "End of Document".

1.3 Document Update Process

The document will be updated whenever the functional content has changed in such a manner to impact the OpenPower SBE firmware users, or whenever the functional content has changed significantly.

Change History

Version	Date	Changes
06/14/23	1.0	Initial version

Table 3: Change History

Copyright and Disclaimer

© Copyright International Business Machines Corporation 2019
IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and Service names might be trademarks of IBM or other companies. A current list of IBM Trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Other company, product, and service names may be trademarks or service marks of others.
All information contained in this document is subject to change without notice. The products described in this document are NOT intended for use in applications such as implantation, life support, or other hazardous uses where malfunction could result in death, bodily injury, or catastrophic property damage. The information contained in this document does not affect or change IBM product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. All information contained in this document was obtained in specific environments, and is presented as an illustration. The results obtained in other operating environments may vary.
While the information contained herein is believed to be accurate, such information is preliminary, and should not be relied upon for accuracy or completeness, and no representations or warranties of accuracy or completeness are made.
Note: This document contains information on products in the design, sampling and/or initial production phases of development. This information is subject to change without notice. Verify with your IBM field applications engineer that you have the latest version of this document before finalizing a design.
THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED ON AN “AS IS” BASIS. In no event will IBM be liable for damages arising directly or indirectly from any use of the information contained in this document.
IBM Systems and Technology Group 2070 Route 52, Bldg. 330 Hopewell Junction, NY 12533-6351
The IBM home page can be found at ibm.com®.

Table of Contents

1 Introduction and Overview

1.1 Introduction

1.1.1 Overview of Host-SBE Communication

The Host to SBE communication is via a set of data and control registers provided by the PSU Block in P10 processor. The registers are used as follows

- 4 8 Byte Registers (PSU_HOST_SBE_MBOX0_REG to PSU_HOST_SBE_MBOX3_REG) for Host to send command packets to SBE. If the data to be sent to SBE cannot be accommodated in these registers, then an indirect data transfer method is followed where the data is stored in memory and the address is shared via these registers.
- 4 8 Byte Registers PSU_HOST_SBE_MBOX4_REG to PSU_HOST_SBE_MBOX7_REG) for SBE to send response packets to Host
- The control registers (PSU_SBE_DOORBELL_REG and PSU_HOST_DOORBELL_REG) have mechanisms to alert party when data is placed in the above-mentioned registers.

1.1.1.1 Protocol Constraints

1. Only one command is accepted in the command buffer until the response for the command is en-queued in the response buffer by SBE

1.1.1.2 Protocol Version Definition

Major version will be incremented when a new API is added either to the SP-SBE interfaces or Host-SBE interfaces.

Minor version will be incremented when an existing API is modified in the SP-SBE interfaces or Host-SBE interfaces.

Major version	1
Minor version	0

1.1.2 Doorbell Status Register Definition

Table 4 Describes the specification for host to sbe doorbell register (PSU_SBE_DOORBELL_REG) communication. Whenever host sets one of the bits in this doorbell register, an interrupt is triggered towards sbe. It is the responsibility of the sbe to clear the bits in this doorbell register that triggered the interrupt.

Table 5 Describes the specification for sbe to host doorbell (PSU_HOST_DOORBELL_REG) register communication. Whenever sbe sets one of the bits in this doorbell register, an interrupt is triggered towards host. It is the responsibility of the host to clear the bits in this doorbell register that triggered the interrupt.

Bits	Description
0	A message is waiting in the Host/SBE mailbox registers

1..15	Reserved

Table 4: Host to SBE Doorbell Register Specification

Bits	Description
0	A response is waiting in the Host/SBE mailbox registers
1	The message in Host/SBE mailbox registers has been read by sbe and is being processed
2	Trigger Stop15 exit on thread 0 on the primary core. This is used to trigger hostboot in istep 16
3	“I am back” - SBE has come back from reset
4	Host Pass-through command received
5..13	Reserved
14	A timer with resolution x has expired
15	Reserved

Table 5: SBE to Host Doorbell Register Specification

1.1.3 Message Format for passing Direct Data

1.1.3.1 Command with Direct Data

	Byte 0	***Byte 1***	***Byte 2***	***Byte 3***	***Byte 4***	***Byte 5***	***Byte 6***	***Byte 7***
Host/SBE Mbx Reg 0	reserved				Seq ID		Command-Class	Command
Host/SBE Mbx Reg 1	[Data Word 0]				[Data Word 1]			
Host/SBE Mbx Reg 2	[Data Word 2]				[Data Word 3]			
Host/SBE	[Data Word 4]				[Data Word 5]			

	Byte 0	*Byte 1*	*Byte 2*	*Byte 3*	*Byte 4*	*Byte 5*	*Byte 6*	*Byte 7*
Mbx Reg 3								

Once this data packet is transferred to the command buffer, the host has to set bit 0 in the PSU_SBE_DOORBELL_REG to trigger an interrupt to SBE. SBE will set the PSU_HOST_DOORBELL_REG bit 1 to acknowledge to the Host that it has received the command, only if the Host has explicitly requested for the same by setting the Control Flag to 0x0100 in the command.

1.1.3.2 Response with Direct Data

SBE will always write a response to the PSU mailbox registers as shown in the table below. However, SBE will set the PSU_HOST_DOORBELL_REG bit 0 to notify the host of the pending response from the SBE, only if the Host has explicitly requested for the same by setting the Control Flags in the command to 0x0200.

	Byte 0	*Byte 1*	*Byte 2*	*Byte 3*	*Byte 4*	*Byte 5*	*Byte 6*	*Byte 7*
Host/SBE Mbx Reg 4	Primary Status		Secondary Status		Seq ID		Command-Class	Command
Host/SBE Mbx Reg 5	[Response Data Word 0]				[Response Data Word 1]			
Host/SBE Mbx Reg 6	[Response Data Word 2]				[Response Data Word 3]			
Host/SBE Mbx Reg 7	[Response Data Word 4]				[Response Data Word 5]			

1.1.4 Message Format for passing Indirect Data

1.1.4.1 Command with Indirect Data

	Byte 0	*Byte 1*	*Byte 2*	*Byte 3*	*Byte 4*	*Byte 5*	*Byte 6*	*Byte 7*
Host/SBE Mbx Reg 0	Size allocated at specified memory address in double words				Seq ID		Command-Class	Command
Host/SBE Mbx Reg 1	[Data Word 0]				[Data Word 1]			
Host/SBE Mbx Reg 2	[Data Word 2]				[Data Word 3]			
Host/SBE	Address (Mainstore / PBA) where data is placed							

	Byte 0	*Byte 1*	*Byte 2*	*Byte 3*	*Byte 4*	*Byte 5*	*Byte 6*	*Byte 7*
Mbx Reg 3								

Once this data packet is transferred to the command buffer, the host has to set bit 0 in the PSU_SBE_DOORBELL_REG o trigger an interrupt to SBE. SBE will set the PSU_HOST_DOORBELL_REG bit 1 to acknowledge to the Host that it has received the command, only if the Host has explicitly requested for the same by setting the Control Flag to 0x0100 in the command.

1.1.4.2 Response with Indirect Data

SBE will always write a response to the PSU mailbox registers as shown in the table below. However, SBE will set the PSU_HOST_DOORBELL_REG bit 0 to notify the host of the pending response from the SBE, only if the Host has explicitly requested for the same by setting the Control Flags to 0x0200 in the command.

	Byte 0	*Byte 1*	*Byte 2*	*Byte 3*	*Byte 4*	*Byte 5*	*Byte 6*	*Byte 7*
Host/SBE Mbx Reg 4	Primary Status		Secondary Status		Seq ID		Command-Class	Command
Host/SBE Mbx Reg 5	[Response Data Word 0]				[Response Data Word 1]			
Host/SBE Mbx Reg 6	[Response Data Word 2]				[Response Data Word 3]			
Host/SBE Mbx Reg 7	reserved				Length of data at address specified in the Command in double words			

1.1.5 FFDC Package

FFDC Package								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Dword 0	Magic Bytes: 0xFFDC		Length in words (N+4)		Sequence Id		Command-Class	Command
Dword 1	Return Code				FFDC – Word 0			
...	...				...			
Dword M	FFDC – Word N-1				FFDC – Word N			

The First Failure Data Capture (FFDC) package is common across the SP side FIFO based interface and Host side PSU based interface. It starts with a header word (Word 0) that has an unique magic identifier code of 0xFFDC followed by the length of the FFDC package including the header itself. Word 1 contains a sequence id , command-class and command fields. The sequence id field is ignored on the SP side FIFO based interface and should match that

of the failed chip-op command when used on the Host side PSU interface. Word 2 contains a 32 bit Return Code which acts like the key to the contents of subsequent FFDC Data Words (0-N).
A FFDC package can typically contain debug data from either:

1. A failed hardware procedure (e.g. local variable values at point of failure)
2. Hardware Registers collected at the point of failure from hardware procedures and
3. SBE firmware (e.g. traces, attributes and other information)

that can assist debug the reason of the failure. The former (hardware procedure failure FFDC) is always returned as a separate FFDC package. The later (firmware debug data) is returned as an additional FFDC package, if the user has set Mailbox Scratch Register#3, bit#5 before starting the SBE. The setting is cached on SBE start-up and any changes when the SBE is running will take effect only after a SBE reset.
The Return Code at word 2 along with the subsequent FFDC Data Words 0-N are expected to be added verbatim, as an opaque blob, to an error log or a dump by the SP for offline parsing and debug.
The success case response “data” returned for the Get SBE FFDC command will also adhere to the same FFDC package format above starting at word 0 and can contain more than one FFDC package.
SBE debug utilities will help parse the FFDC package into user readable debug information.

1.1.6 Target Types and Ids Supported by SBE

The SBE chip-ops accept the “Pervasive Chiplet Id” as a consistent input to map target types / ids across different calling entities (HB, HWSV and Cronus) to corresponding target types and instances in the SBE platform. The pervasive chiplet ids listed in the table below are identical to the hardware chiplet ids. When non-obvious (e.g. no unique mapping between a logical target type and a pervasive chiplet id is available), the SBE will derive a target type and it's instance from a combination of the input target type and pervasive chiplet id. Some logical target type-instances cannot be uniquely mapped to an SBE target, via a physical chiplet id. In such cases , the SBE specification defines virtual chiplet ids for such targets that start with 0b1XXXXXXX. Please refer the table below for details.

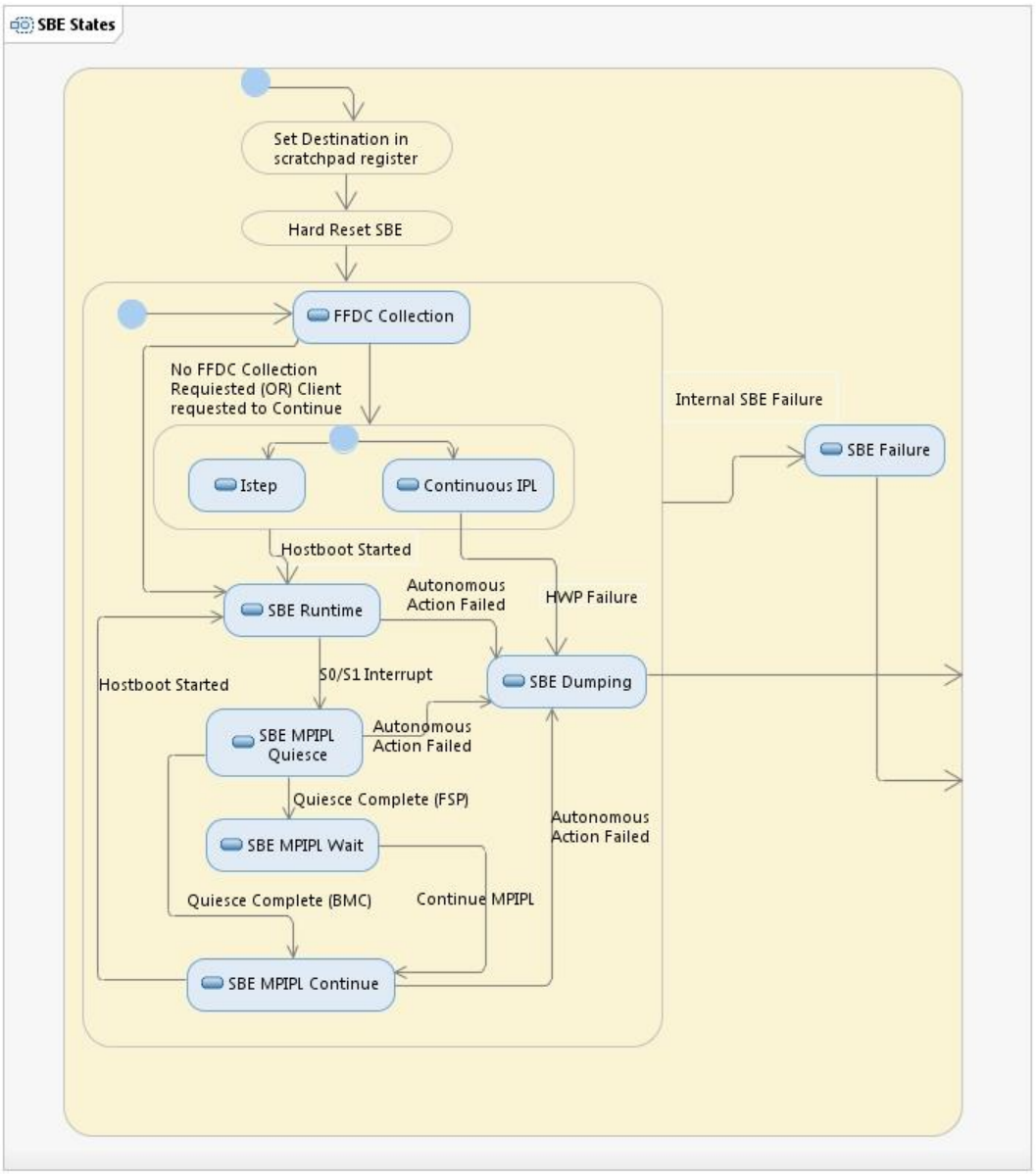
Target Type	*Pervasive Chiplet Id(s)*	*Implied SBE Target*	*Physical Chiplet(s)*	*Notes / Comments*
`TARGET_TYPE_PROC = 0x0000`	`N.A.`	Processor Chip	Any chiplet in the processor	The chiplet id used is a reserved value for processor targets and does not correspond to any particular pervasive chiplet in the processor.
`TARGET_TYPE_PERV = 0x0001`	`0x01`	Pervasive	PERV	This combination **must** always be used when there is an unique, one-to-one mapping between the intended target's type-instance and it's pervasive chiplet id.
	`0x02-0x03`	Nest	N0-N1 respectively	
	`0x08-0x09`	PCI	PCI0–PCI1 respectively	
	`0x0C-0x0F`	MC	MC0 and MC3 respectively	
	`0x10-0x14`	PAU	PAU0-PAU3 respectively	
	`0x18-0x1F`	IOHS	IOHS0-IOHS3 respectively	
	`0x20-0x27`	Cache	EQ0-EQ7 respectively	
`TARGET_TYPE_EQ = 0x0002`	`0xFF`	All Caches	All EQ0-EQ7	No such pervasive chiplet id exists in hardware. For the purpose of this

Target Type	*Pervasive Chiplet Id(s)*	*Implied SBE Target*	*Physical Chiplet(s)*	*Notes / Comments*
				interface, represents all cache targets in the chip
TARGET_TYPE_CORE = 0x0003	0xFF	All Cores	All EC00-EC31	No such pervasive chiplet id exists in hardware. For the purpose of this interface,, represents all "normal" core targets in the chip
TARGET_TYPE_OCMB = 0x0004	0x00-0x0F	OCMB	OCMB 0 -OCMB 15	No such pervasive chiplet id exists in HW, This interface is to read/write registers out of the OCMB chip

SBE will always accept core ids (0x0-0x1F) as in the table above as in the table above, irrespective of whether the system is configured in normal (SMT4) or big (SMT8) core mode.
In addition, the SBE will always accept virtual thread numbers (0-3), irrespective of whether the system is configured in normal (SMT4) or big (SMT8) core mode. It is the responsibility of higher-level firmware to map logical thread number to the right SMT4 core chiplet and virtual thread number.
Note that composite target types will not be accepted in SBE chip-ops.

2 Summary of SBE Host Interface Commands

Cmd Class (hex)	Cmd (hex)	Name	Supported States	Timeout
D1		Core State Control Messages		
	01	Control Deadman Loop	SBE runtime	Short
	02	Exit Cache Contained Mode	SBE runtime	Long
	03	Update Core Functional States	SBE runtime	Short
D3		Ring Access Messages		
	01	Put Ring From Image	SBE runtime	Long
D4		Timer Control Messages		
	01	Control Timer	SBE runtime	Short
D5		Register Access Messages		
	01	Get Hardware Register	SBE runtime	Short
D6		Security Control Messages		
	01	Set Unsecure Memory Region	SBE runtime	Short
D7		Generic Messages		
	02	Get SBE Capabilities	All States	Short
	03	Read SBE SEEPROM	SBE Runtime	Long
	04	Set FFDC Address	SBE runtime	Short
	05	Quiesce SBE	All States	Long
	06	Set System Fabric Id Map	All States	Short
	07	Stash MPIPL Config	SBE runtime	Short
	08	Security List Binary Dump	SBE runtime	Short
	09	Update OCMB Target	SBE runtime	Short
	0A	Sync Fabric Topology ID Table	SBE runtime	Short
	0B	Memory Config	SBE runtime	Short
	0C	PMIC Health Check	SBE runtime	Shortf



FFDC Collection : SBE is booting from OTPROM, initializing PK Kernel and collecting FFDC. SBE will not wait in this state if the client has not requested for FFDC collection post reset.
Istep : SBE is in istep mode and is awaiting commands from SP to execute specific isteps.

Continuous IPL : SBE is in continuous IPL mode. SP has to wait for SBE to reach "SBE runtime" state to send any other commands except "Is SBE IPL Done". Note that SBE will reach "SBE runtime" state once hostboot is started. In case of any failures encountered in the "Continuous IPL" state, SBE will reach "SBE Failure" state
SBE runtime : SBE is ready to process any chip-ops commands.
SBE Failure : SBE encountered an internal failure. A Hard Reset of SBE is required to exit this state
SBE dumping : SBE encountered an error during IPL process due to which it could not reach SBE Runtime
SBE MPIPL : SBE has entered MPIPL state based on request by Host. SBE will reach Runtime state once Hostboot is started
There is no priority implied by the Command Class value. Prioritization for sending each message is established by the sender. Prioritization for handling each message is on a FCFS basis by SBE.

The category of timeout for each command is specified in the commands table. If a response exceeds these time-outs, it is considered as a protocol violation. See section “Handling Protocol Violations” for details.
The timeouts for commands are categorized as Long and Short. The timeout value for Short running commands is 100 msecs while the timeout value for Long running commands is 30 secs.

2.1 Responses to Commands

All messages with responses have a number of valid primary and secondary status values returned with the response. Primary status codes are one of the following:
The primary and secondary status fields have to be decoded as follows

Primary Status	Bits 0..15 : Primary status codes as defined in Table 6
Secondary Status	Bits 0..15 : Secondary status codes as defined in Table 7

All message with responses have a number of valid primary and secondary status values returned with the response. Primary status codes are one of the following:
The following table defines the primary status responses

Value	**Definition**
0x00	Operation successful.
0x01	invalid or unsupported command
0x02	Invalid data passed
0x03	User Error
0x04	SBE Internal Error
0x05	Unsecure Access Denied
0xFE	Generic Failure in Execution

Table 6: Primary Status Codes

The following table defines the secondary status responses.

Value	**Definition**
0x00	Operation successful.
0x01	Command Class not supported
0x02	Command not supported
0x03	Invalid address passed
0x04	Invalid target type passed
0x05	Invalid chiplet ID passed
0x06	Specified target is not present
0x07	Specified target is not functional
0x08	Command not allowed in this state.
0x09	Functionality not Supported
0x0A	Generic failure in execution

0x0B	Denied register accessed
0x0C	SBE Operating System Failure
0x0D	Host MBX Reg Access Failure
0x0E	Insufficient data passed as part of command
0x0F	Excess data passed as part of command
0x10	Hardware timeout
0x11	PCB-PIB Error
0x12	SBE FIFO Parity Error
0x13	Unused
0x14	Denied memory accessed
0x15	Unsecured memory region not found
0x16	Exceeded maximum supported unsecured memory regions
0x17	Unsecured memory region amend attempted
0x18	Input buffer overflow
0x19	Invalid Parameters
0x20	Denied chip-op access
0x21	Deadman Timer time out
0x22	System check-stop
0x23	Denied register access blocked
0x24	Start MPIPL Failed (SP less MPIPL)
0x25	Stop Clock Failed (SP less MPIPL)
0x26	Continue MPIPL Failed (SP less MPIPL)
0x27	Periodic IO Toggle failed
0x28	Special Wakeup time out
0x29	Special Wakeup Scom failure
0x2A	Architecture Register dump fail
0x2B	Lpc Access fail
0x2C	Hardware Procedure Fail
0x2D	Special Attention Core Scratch Read Fail
0x2E	Special Attention Data Read Fail
0x2F	Put Sram Fail
0x30	Get Sram Fail
0x31	Thread Control instruction Fail
0x32	Ram Core Setup Fail
0x33	Ram Core Access Fail
0x34	Ram Core Cleanup Fail
0x35	Suspend IO Procedure Fail
0x36	Enter Mpipl Fail

0x37	Stop Clock Fail
0x38	OCMB SCOM Failed
0x39	Get Dump Failed
0x40	Invalid Dump Type Input
0x41	Invalid OCMB Instance
0x42	Failed to trigger check stop
0x43	Get Dump Stream failed
0x44	MPIPL Dump invalid input parameters
0x45	OCMB target not present
0x46	OCMB target not functional
0x47	Invalid fast array collection info
0x48	PMIC Health Check failed
0x49	Invalid I2C config version
0x4A	Invalid I2C target
0x4B	LPC Error
0x4C	Command not privileged on this FIFO interface

Table 7: Secondary Status Codes

2.2 Command Definitions

The remainder of this document details the messages and responses. It is organized by function. Many sections contain a protocol flow or other overall description of the functions and message sequences involved. Note that all flows are at the protocol level and do not include the hardware level acknowledgments.

2.2.1 General Notes on Message Parameters

For most messages, there is some amount of data transferred with the message. This data is packaged along with the command in the same message. The length field specified in the message should indicate the total size of the message buffer passed via the command buffer including the length field itself.

2.3 Reset / Reload Considerations

None

2.4 Handling Protocol Violations

Three types of reset are supported by SBE

Soft Reset : This will reset the SBE and take the program counter to the start of the SBE firmware. But, any ongoing memory transactions will be allowed to complete before the reset is honored.

Hard Reset: This will reset the SBE and take the program counter to the start of the SBE firmware immediately.

Prior to performing Hard Reset on SBE, Host has to set the appropriate destination state bits in the scratchpad register **0x5003A**. Three destination states are supported by SBE

a. ISTEP Mode : Upon reset, SBE will wait for commands from SP to execute isteps

b. PLCK Mode : Upon reset, SBE will autonomously start the IPL procedures and reach SBE runtime

c. Runtime Mode : Upon reset, SBE will wait for Chip-Ops from SP.

3 Core State Control Messages

3.1 Control Deadman Loop

This command is sent by Host to SBE to enter deadman loop for exit STOP15. This is a synchronous command where SBE starts to enqueue the response buffer once the operation is completed.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	**Reserved**		**Control Flags**		**Seq ID**		**D1**	**01**
Mbx Reg 1	**Time to wait in deadman loop in milliseconds**							
Mbx Reg 2	Reserved							
Mbx Reg 3	Reserved							
	Control Flags : 0x0001: Start Deadman Loop using the timeout specified 0x0002 : Stop Deadmap Loop. No Time value required when this bit is asserted 0x0100 : SBE Response Required 0x0200 : SBE Ack Required for this message Other values : Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	**Primary Status**		**Secondary Status**		**Seq ID**		**D1**	**01**
Mbx Reg 5	**Reserved**							
Mbx Reg 6	**Reserved**							
Mbx Reg 7	**Reserved**							
	Primary Status, Secondary Status : See section 2.1							

3.2 Exit Cache Contained Mode

This command is sent by Host to SBE to execute the exit cache contained mode hardware sequence and with the address-data pair entries available at the address specified in the command. This is a synchronous command where SBE starts to enqueue the response buffer once the operation is completed.

Message (request) from Host

	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mbx Reg 0	Reserved		Control Flags		Seq ID		D1	02
Mbx Reg 1	Number of Address-Data Pair (XSCOM) Entries stored at Cache Address				Step Information for Exit Cache Contained Procedure Execution			
Mbx Reg 2	Cache Address to fetch XSCOM address-data pair							
Mbx Reg 3	Reserved							
	Control Flags: 0x0100 : SBE Response Required 0x0200 : SBE Ack Required for this message Other values : Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		D1	02
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

3.3 Update Core Functional Status

This command is sent by Host to SBE to synchronize its view of the functional cores in the chip with SBE's view of functional cores in the chip. This is a synchronous command where SBE starts to enqueue the response buffer once the operation is completed.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		D1	03
Mbx Reg 1	Core Functional States				Reserved			
Mbx Reg 2	Reserved							
Mbx Reg 3	Reserved							

	Core Functional States: Bits 0:31 - Each bit represents whether the respective core number (0:31) is functional or not, where: 0b0: Core Functional 0b1: Core Non-Functional ***Control Flags:*** 0x0100 : SBE Response Required 0x0200 : SBE Ack Required for this message Other values : Reserved

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		D1	03
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

4 Ring Access Messages

4.1 Put Ring From Image

This command is sent by Host to SBE to scan write into a ring specified by the ring id on a chiplet on the processor where the SBE is placed. The data to be scanned-in must be available as a part of the SBE SEEPROM Image and is identified by a combination of the specified ring id, target type and chiplet id. This is a synchronous command where SBE starts to enqueue the response buffer once the operation is completed.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD3	0x01
Mbx Reg 1	Target Type		Reserved	Chiplet Id	Ring Id		Ring Mode	
Mbx Reg 2	Reserved							
Mbx Reg 3	Reserved							
	Control Flags: 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other Values : Reserved **Target Type and Chiplet Id:** Refer section 1.1.7 for more details **Ring Id:** Refer enumeration in file at: `src/import/chips/p9/utils/imageProcs/p9_ring_id.H` **Ring Mode:** 0x0001: No Header Check 0x0002: Set Pulse with no OPCG conditioning 0x0004: Set Pulse with NSL pulse 0x0008: Set Pulse with SL pulse 0x0010: Set Pulse with pulse to all hold types							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	**Primary Status**		**Secondary Status**		**Seq ID**		**D3**	**01**
Mbx Reg 5	**Reserved**							
Mbx Reg 6	**Reserved**							
Mbx Reg 7	**Reserved**							
	Primary Status, Secondary Status : See section 2.1							

5 Timer Control Messages

5.1 Control Timer

This command is sent by Host to SBE to start a one-shot timer for a specified duration or stop a timer that was already started. This is a synchronous command where SBE starts to enqueue the response buffer once the operation is completed. On timer expiry, SBE will asynchronously notify the host by setting bit 14 of the PSU doorbell register[PSU_HOST_DOORBELL_REG]. Note that, the PSU Doorbell notification for the elapsed timer will be seen only in the processor hosting the SBE with which the timer was started.

SBE will support only one timer at a time. Restarting a timer while it is still running, will cause the timer to start afresh with the new timeout. Stopping a timer that already expired will be no-operation and return success.

It is recommended that:

a) the host factors in the SBE hardware and protocol overhead of ~17 microseconds

b) the host waits (either polls response in mailbox register or subscribes for a notification via the doorbell mechanism) for a response to the Control Timer operation (eliminates variance if SBE is busy with other command)

c) the host uses a minimum timeout of 100 microseconds (guarantees that fixed doorbell notification error is less than 1.7%). For timeout values above the recommended 100 microsecond, the SBE timer can handle granularity of 1 microsecond.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	**Reserved**		**Control Flags**		**Seq ID**		**D4**	**01**
Mbx Reg 1	Reserved				[Timeout in microseconds]			
Mbx Reg 2	Reserved							
Mbx Reg 3	Reserved							
	Control Flags : 0x0001: Start a new Timer or Restart an active Timer for a the timeout specified in microseconds 0x0002: Stop Timer. Timeout field is ignored. 0x0100: Response ready notification is required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	**Primary Status**		**Secondary Status**		**Seq ID**		**D4**	**01**
Mbx Reg 5	**Reserved**							
Mbx Reg 6	**Reserved**							
Mbx Reg 7	**Reserved**							
	Primary Status, Secondary Status : See section 2.1							

6 Hardware Register Access Messages

6.1 Get Hardware Register

This command is sent by Hostboot to SBE to retrieve Memory Buffer SCOM registers. This is a synchronous command where SBE will enqueue its response after completing the command. This command currently supports only memory buffer target.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		D5	03
Mbx Reg 1	Target Type		Reserved	Instance	Address Word 0 (Bits 0:31)			
Mbx Reg 2	Address Word 1 (Bits 32:63)				Reserved			
Mbx Reg 3	Reserved							
	Control Flags: 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		D5	03
Mbx Reg 5	Register Data (Bits 0:63)							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

7 Security Control Messages

7.1 Set Unsecure Memory Region

This command is sent by Host to SBE to open or close access to a continuous block of memory. This is a synchronous command where SBE will enqueue its response after completing the command.
When the SBE boots, all memory accesses via the Get/Put Memory SBE FIFO Chip-Op are denied by default. By virtue of this command, the Host can dynamically add or remove up to eight un secure memory regions to each SBE. Each un secure memory region in the SBE can be either read-only accessible or read-write accessible. Each new secure memory region must be mutually exclusive of all other regions already created such that there are no overlapping regions or sub-regions. Each SBE only knows about the allowed list of un secure memory regions that it owns and will deny access to any memory outside this list. This command must be sent to either the appropriate SBE or to all SBEs in the system, based on the intended use-case of the memory region to be access over the FIFO.

If the SBE is reset or rebooted while the host is still running, the host will have to set all the un secure memory regions again.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	**Reserved**		**Control Flags**		**Seq ID**		**D6**	**01**
Mbx Reg 1	Reserved				[Size in bytes]			
Mbx Reg 2	Start address of unsecure memory region							
Mbx Reg 3	Reserved							
	Control Flags : 0x0011: Open an unsecure memory region, of size specified in Mbx Reg 1 and starting at the address specified in Mbx Reg 2, with **read-only access**. 0x0012: Open an unsecure memory region, of size specified in Mbx Reg 1 and starting at the address specified in Mbx Reg 2, with **read-write access**. 0x0020: Close an already open unsecure memory region starting at the address specified in Mbx Reg 2 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	**Primary Status**		**Secondary Status**		**Seq ID**		**D6**	**01**
Mbx Reg 5	**Reserved**							
Mbx Reg 6	**Reserved**							
Mbx Reg 7	**Reserved**							

	Primary Status, Secondary Status : See section 2.1

8 Generic Messages

8.1 Get SBE Capabilities

This command is sent by Host to SBE to retrieve SBE to Host Protocol Version as defined in this document. The SBE also responds with additional information like the shorthand (first eight hexadecimal characters) commit ID, a twenty character (5 words) alphanumeric string representing the SBE firmware release tag and the various capabilities supported by the level of firmware running on the SBE. Please refer to the table below for details on the various capabilities planned to be supported by the SBE over the Host (PSU side) interface. This is a synchronous command where SBE will enqueue its response after completing the command.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		D7	02
Mbx Reg 1	Size Allocated in bytes at address specified in Mbx Reg 2							
Mbx Reg 2	Address allocated by Host where SBE has to return Capabilities structure							
Mbx Reg 3	Reserved							
	Control Flags: 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values: Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		D7	02
Mbx Reg 5	Reserved				Length of Capabilities data returned in bytes at address specified by Host			
Mbx Reg 6	Major Version		Minor Version		8 Hex Characters Firmware Commit ID			
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status: See section 2.1							

Capabilities data returned by SBE								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Address 0	SBE Release Tag Word .. 0				SBE Release Tag Word .. 1			
Address 1	SBE Release Tag Word .. 2				SBE Release Tag Word .. 3			
Address 2	SBE Release Tag Word .. 4				Capabilities..0			
Address 3	Capabilities..1				Capabilities..2			
Address 4	Capabilities..3				Capabilities..4			
Address 5	Capabilities..5				Capabilities..6			
Address 6	Capabilities..7				Capabilities..8			
Address 7	Capabilities..9				Capabilities..10			
Address 8	Capabilities..11				Capabilities..12			
Address 9	Capabilities..13				Capabilities..14			
Address 10	Capabilities..15				Reserved			

Capabilities..0	0xC0000001	HWP FFDC Collection supported
	0xC0000002	SBE FFDC Collection supported
	0xC0000004	Address Allow/Deny filtering supported
	0xC0000008	FIFO Reset supported
	0xC0000010	Host command interface supported
	0xC0000020	Service Processor less MPIPL supported
Capabilities..1	0xC8000000	Reserved for generic capabilities
Capabilities..2	0xD1000001	Control Deadman Loop
	0xD1000002	Exit Cache Contained Mode
	0xD1000004	Update Core Deconfig State
Capabilities..3	0xD1800000	Reserved for capabilities of D1 class
Capabilities..4	0xD2000001	Execute Multi Scom supported
Capabilities..5	0xD2800000	Reserved for capabilities of D2 class
Capabilities..6	0xD3000001	Put Ring from Image - Supported
Capabilities..7	0xD3800000	Reserved for capabilities of D3 class
Capabilities..8	0xD4000001	Control Timer supported
Capabilities..9	0xD4800000	Reserved for capabilities of D4 class
Capabilities..10	0xD5000001	Get Architected Registers supported
	0xD5000002	Clear Saved Architected Registers supported
	0xD5000004	Get Hardware Register supported
Capabilities..11	0xD5800000	Reserved for capabilities of D5 class
Capabilities..12	0xD6000001	Set Unsecure memory region supported
Capabilities..13	0xD6800000	Reserved for capabilities of D6 class
Capabilities..14	0xD7000001	Get SBE FFDC supported
	0xD7000002	Get SBE Capabilities supported
	0xD7000004	Read SBE SEEPROM supported
	0xD7000008	Set FFDC Address supported
	0xD7000010	Quiesce supported
	0xD7000020	Set System Fabric Id Map supported
	0xD7000040	Stash MPIPL config supported
	0xD7000080	Security List Binary Dump
	0xD7000100	Update OCMB Target
	0xD7000200	Sync Fabric Topology ID
	0xD7000400	Memory Config
	0xD7000800	PMIC Health Check
Capabilities..15	0xD7800000	Reserved for capabilities of D7 class

8.2 Read SBE SEEPROM

This command is sent by the Host to the SBE to read meta data from the SBE SEEPROM, by neither contending with the SBE for SEEPROM read access nor requiring a SBE reboot due to the Quiesce command. This is a synchronous command where SBE will en queue the response after completing the command.

Note that for SBE SEEPROM write access, the Host must still use the Quiesce command before writing to the SEEPROM.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD7	0x03
Mbx Reg 1	Offset (bytes) from SEEPROM base. Must be 8 bytes aligned				Size of data (multiples of 8 bytes) to read from the specified offset			
Mbx Reg 2	Address allocated by Host where SBE has to copy the SEEPROM data							
Mbx Reg 3	Reserved							
	Control Flags : 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		0xD7	0x03
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

8.3 Set FFDC Address

This command is sent by Host to SBE to specify the address where SBE can return the FFDC for any chip-op requested by Host. This command must be sent to all SBEs in the system. The FFDC data could contain multiple FFDC packages and is compliant to the FFDC package format described in section 1.1.4.3. Each SBE should have an exclusive FFDC address and range such that FFDC from multiple SBEs does not trample each other.

This command is also used by the Host to specify the address where SBE can write host pass-through commands to and read host pass-through responses from. When the SBE receives a Host Pass-through command from the BMC, it stores the command at this

address and notifies the host by asserting bit 4 of the SBE to Host Doorbell register. The host pass-through commands are used only via the Primary SBE and hence this address is reserved and unused on the secondary SBEs.
An address will be considered as valid, only if its corresponding size field as specified in the PSU Mailbox register 1 is non-zero.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	**Reserved**		**Control Flags**		**Seq ID**		**D7**	**04**
Mbx Reg 1	Size (in bytes) allocated for FFDC data starting at address specified in Mbx Reg 2				Size (in bytes) allocated for pass-through command / response starting at address specified in MBX Reg 3			
Mbx Reg 2	Address where SBE can return FFDC							
Mbx Reg 3	Address where SBE can write a pass-through command to the host, or read a pass-through response from the host							
	Control Flags: 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved Note: SBE uses the PBA hardware facility to read / write data to the mainstore. Hence, the start address should be 128 byte aligned and size allocated should be a multiple of 128 bytes.							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	**Primary Status**		**Secondary Status**		**Seq ID**		**D7**	**04**
Mbx Reg 5	**Reserved**							
Mbx Reg 6	**Reserved**							
Mbx Reg 7	**Reserved**							
	Primary Status, Secondary Status : See section 2.1							

8.4 Quiesce

This command is sent to place the SBE into a quiescent state. Note that this command cannot be used to quiesce a previously issued command. The SBE will stop any ongoing operation and enter a state where it will only accept a subset of chip-ops from the SP interface (FIFO) as well as the Host interface (PSU) that are guaranteed to execute without accessing the I2C bus. This is a synchronous command where SBE will en queue its response after completing the command. This command is usually used by

the SP on power down paths or by the Host when performing a SBE code update. A SBE reboot is required to exit the quiescent state.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD7	0x05
Mbx Reg 1	Reserved							
Mbx Reg 2	Reserved							
Mbx Reg 3	Reserved							
	Control Flags : 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		0xD7	0x05
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

8.5 Set System Fabric Id Map

This command is send by Hostboot to all the SBEs to set the system fabric id map indicating all functional processors in the system. This map is expected to hold good until the next system boot (IPL). For every functional processor in the system, the corresponding bit in map should be set. This is a synchronous command where SBE will en queue the response after completing the command.

Message (request) from Host

	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD7	0x06
Mbx Reg 1	System Fabric Id Map representing all functional processors in the system							
Mbx Reg 2	Reserved							
Mbx Reg 3	Reserved							
	Control Flags : 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved							

The System Fabric Id Map:

DWord Nibble	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Fabric Group Id	0		1		2		3		4		5		6		7	
Fabric Chip Id	0 1 2 3	4 5 6 7	0 1 2 3	4 5 6 7	0 1 2 3	4 5 6 7	0 1 2 3	4 5 6 7	0 1 2 3	4 5 6 7	0 1 2 3	4 5 6 7	0 1 2 3	4 5 6 7	0 1 2 3	4 5
Example (hex):	F	0	9	0	0	0	0	0	0	0	0	0	0	0	0	0

The above example data indicates that 4 consecutive processors (fabric chip id 0 to 3) in fabric group 0 are functional and 2 processors (fabric chip id 0 and 3) in fabric group 1 are functional. All other processors in fabric group 0 and 1 are either not functional or not present. All processors in a fabric group being not functional can also imply the particular fabric group is not present in the system.

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		0xD7	0x06
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

8.6 Stash MPIPL Config

This command is send by Hostboot or OPAL to all the primary SBEs to save MPIPL related host settings to be used during the next Memory Preserving IPL. This is a synchronous command where SBE will en queue the response after completing the command. The configuration settings are saved as key-value pairs, which are opaque to the SBE. Consequently, the definition of the keys itself is beyond the scope of this document. Default or unused keys will be initialized to 0xFF. The same command can be used repeatedly to stash away up to a maximum of eight unique key-value pairs. A key-value pair, once set, can only be updated using a new value corresponding to its key. There is no use-case and hence no support to reset a key-value pair back to its default. These MPIPL configuration settings will be passed to the Hostboot loader during every IPL. Consequently, there is no explicit command to retrieve the MPIPL configuration settings directly from the SBE.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD7	0x07
Mbx Reg 1	Reserved							Key
Mbx Reg 2	Value							
Mbx Reg 3	Reserved							
	Control Flags: 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved **Key:** Mbx Reg 1, Bits 56-63							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		0xD7	0x07
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

8.7 Security List Binary Dump

This command is sent by Hostboot to SBE to get a dump of the security lists maintained by SBE. The memory address where the security lists should be dumped is specified in Mbx Reg 1. This is a synchronous command where SBE will enqueue its response after completing the command. The definition of the security list dump is a contract between the Host and SBE and is beyond the scope of this document.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD7	0x08
Mbx Reg 1	Address to write the Security List Dump contents							
Mbx Reg 2	Reserved							
Mbx Reg 3	Reserved							
	Control Flags: 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values: Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		0xD7	0x08
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

8.8 Update OCMB Target

This command is sent by Hostboot to SBE to update the OCMB (memory buffer) targets in the SBE. This is a synchronous command where SBE will enqueue its response after completing the command. The definition of the OCBM Target info data, written by Hostboot in Mbx Reg1 to Mbx Reg3 and interpreted by SBE, is a contract between the Host and SBE and is beyond the scope of this document. Note that this command is being deprecated and users are requested to start using the Configure Memory command (0xD70B) instead.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD7	0x09

Mbx Reg 1	OCMB Target Info
Mbx Reg 2	OCMB Target Info
Mbx Reg 3	OCMB Target Info
	Control Flags: 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values: Reserved

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		0xD7	0x09
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

8.9 Synchronize Fabric Topology ID Table

This command is sent by Hostboot to SBE to synchronize the Fabric Topology ID table.. The fabric topolody ID table is stored at the memory address specified in Mbx Reg 1 before sending this command. This is a synchronous command where SBE will en

-queue its response after completing the command. The definition of the fabric topology ID table is a contract between the Host and SBE and is beyond the scope of this document.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD7	0x0A
Mbx Reg 1	Topology Table Info Address							
Mbx Reg 2	Reserved							
Mbx Reg 3	Reserved							
	Control Flags: 0x0100: Response required from SBE							

	0x0200: SBE Ack Required for this message Other values : Reserved

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		0xD7	0x0A
Mbx Reg 5	Reserved							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							
	Primary Status, Secondary Status : See section 2.1							

8.10 Configure Memory

This command is sent by Hostboot to SBE to configure memory (DIMM) targets. The data used to configure the DIMM targets is populated at the address specified in Mbx Reg 2 before sending the command. This is a synchronous command where SBE will en queue its response after completing the command. The definition of the data is a contract between the Host and SBE and is beyond the scope of this document.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		0xD7	0x0B
Mbx Reg 1	Reserved							
Mbx Reg 2	Address where Hostboot has set up data for SBE to read the memory config data							
Mbx Reg 3	Reserved							
	Control Flags: 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values : Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		0xD7	0x0B

Mbx Reg 5	Reserved
Mbx Reg 6	Reserved
Mbx Reg 7	Reserved
	Primary Status, Secondary Status : See section 2.1

8.11 PMIC Health Check

This command is sent by the Host to SBE to perform PMIC health check of a DIMM. SBE will invoke the hardware sequence to analyze the PMIC status and write the data at the nemory addresses specified in Mbx Reg 2. This is a synchronous command where SBE will enqueue its response after completing the command. The definition of the data is a contract between the Host and SBE is beyond the scope of this document.

Message (request) from Host								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 0	Reserved		Control Flags		Seq ID		D7	0C
Mbx Reg 1	Target Type		Reserved	Instance	Reserved			
Mbx Reg 2	Address where PMIC data is returned							
Mbx Reg 3	Reserved							
	Target Type: *Only OCMB Target Type supported* ***Instance:*** *OCMB Target Instance ID (0x00-0x0F per processor chip)* ***Address:*** *Mainstore address where PMIC data will be placed by SBE* ***Control Flags:*** 0x0100: Response required from SBE 0x0200: SBE Ack Required for this message Other values: Reserved							

Response from SBE								
	Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**
Mbx Reg 4	Primary Status		Secondary Status		Seq ID		D7	0C
Mbx Reg 5	Length of PMIC Data, in bytes, written at address specified in Mbx Reg 2							
Mbx Reg 6	Reserved							
Mbx Reg 7	Reserved							

	Primary Status, Secondary Status : See section 2.1

9 Open Items / Work in Progress

9.1 Upcoming chip-ops

None

9.2 Miscellaneous updates

1. None

9.3 Open Items

1. None

1

2

3 # End of Document