SBE FIFO Interface Specification for P10 Systems

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. This may be determined by checking the current version on the Web site.

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

Date	Version	Changes
14/06/23	1.0	Initial version

Table 3: Change History

Copyright and Disclaimer

Table of Contents

2 Introduction and Overview

2.1 Introduction

2.1.1 Overview of SBE FIFO Operation

The SP to SBE communication is via a SBE FIFO device (address CFAM 0x240X, SCOM 0x000B001X) in the P10 processor. The P10 processor has an additional instance of the FIFO device (address CFAM 0x241X, SCOM 0x000B002X) exclusively for Host to communicate with the SBE. Both the FIFO devices are functionally equivalent. Each FIFO is implemented as two circular buffers.

- One circular buffer is used by SP/Host to send commands to SBE. This buffer shall be referred to as “command buffer” henceforth in this document. The write pointer on this buffer is controlled by HW via write operations originating from SP/Host while the read pointer is controlled by HW via read operations originating from SBE
- The other circular buffer is used by SBE to send responses back to SP/Host. This buffer shall be referred to as “response buffer” henceforth in this document. The write pointer on this buffer is controlled by HW via write operations originating from SBE while the read pointer is controlled by HW via read operations originating from SP/Host

Both the buffers are of equal length, and it is the responsibility of the party writing to the buffers to do flow control.

2.1.2 SBE FIFO Hardware Mechanism

Each entry in the command and response buffer consists of the following additional flags:

- End-Of-Transfer (EOT)
- Data Valid (DV)

It is the responsibility of the party holding the write pointer to set these flags appropriately.

2.1.2.1 Status/Control Register

The status / control register has the following functions corresponding to the command buffer for communications from the SP to SBE.

Field	Description
Parity / Error	Parity/Error
Requesting FIFO Reset	Indicates that the service processor is requesting for resetting both FIFOs
End-Of-Transfer	A '1' indicates that the recently dequeued entry has the EOT flag set
FIFO Full	A '1' indicates that all entries are occupied in the upstream FIFO
FIFO Empty	A '1' indicates that the upstream FIFO is empty
Field	**Description**

FIFO entry count	represents the number of currently hold FIFO entries
FIFO Valid Flags	represents the valid flags of ALL currently held FIFO entries; one dedicated bit per FIFO entry, starting with bit 16 which reflects the valid flag of first FIFO entry
FIFO EOT Flags	represents the EOT flags of ALL curently hold FIFO entries; one dedicated bit per FIFO entry, starting with bit 24 which reflects the EOT flag of the first FIFO entry

The status / control register has the following functions corresponding to the response buffer for communications from the SBE to SP.

Field	Description
Parity / Error	Parity/Error
Requesting FIFO Reset	Indicates that the SBE is requesting Service Processor for resetting both FIFOs
End-Of-Transfer	A '1' indicates that the recently dequeued entry has the EOT flag set
FIFO Full	A '1' indicates that all entries are occupied in the upstream FIFO
FIFO Empty	A '1' indicates that the upstream FIFO is empty
FIFO entry count	represents the number of currently hold FIFO entries
FIFO Valid Flags	represents the valid flags of ALL curently hold FIFO entries; one dedicated bit per FIFO entry, starting with bit 16 which reflects the valid flag of first FIFO entry
FIFO EOT Flags	represents the EOT flags of ALL curently hold FIFO entries; one dedicated bit per FIFO entry, starting with bit 24 which reflects the EOT flag of the first FIFO entry

2.1.3 Protocol Constraints

1. The SP or Host always behave as the initiators of the commands (also called Chip-Ops).
2. The SBE always behaves as the receiver of the commands.
3. Only one command is accepted in the command buffer until the response for the command is en-queued in the response buffer by SBE and de-queued completely by the initiator and EOT is acknowledged in the response buffer
4. Each command from initiator must be explicitly completed with a EOT.
5. Each response from SBE must be explicitly completed with a EOT.
6. A response cannot be sent back on the response buffer, while the command is still being received on the command buffer. This is not a hardware limitation but a design choice to keep the SP/Host driver as well as the SBE code simple.
7. A FIFO reset will always be initiated from the initiator and acknowledged by the SBE.
8. As the SBE behaves as a receiver of the command, it will never initiate a FIFO reset.

2.1.3.1 Protocol Version Definition

The Major version will be incremented when a new interface/chip-op is added.

The Minor version will be incremented when a change is made to an existing interface/chip-op.

Major version	2
Minor version	1

2.1.3.2 Message Format for Command Buffer

	Byte 0	***Byte 1***	***Byte 2***	***Byte 3***
Word 0	Command length in number of words			
Word 1	reserved	reserved	Command-Class	Command
Word 2	Data Word 0			
...	...			
Word N+2	Data Word N			

Once this data packet is transferred to the command buffer, the service processor has to signal EOT which will create a dummy entry in the upstream FIFO (command buffer) along with the EOT Flag set.

2.1.3.3 Message Format for Response Buffer

Once this data packet is transferred to the response buffer, the SBE has to signal EOT which will create a dummy entry in the downstream FIFO (response buffer) along with the EOT Flag set.

Note that the primary, secondary status and related FFDC are placed towards the end of the response buffer. This is to accommodate use-cases where a variable length of data may be transferred over the response buffer and errors may be encountered during SBE chip-ops operations.

Words N+1 and N+2 below, comprise the status header. These along with the last word, which indicates distance to the status header, are mandatory for every response from the SBE.

[Entries] = Optional Entries. May not be present in some responses.

	Byte 0	***Byte 1***	***Byte 2***	***Byte 3***
Word 0	[Response Data Word 0]			
...	...			
Word N	[Response Data Word N]			

	Byte 0	***Byte 1***	***Byte 2***	***Byte 3***
Word N+1	**Magic Bytes: 0xC0DE**		**Command-Class**	**Command**
Word N+2	**Primary Status**		**Secondary Status**	
Word N+3	[FFDC Package-0..Word-0]			
	...			
Word N+M+4	[FFDC Package-0..Word-M]			
	..			
Word S	[FFDC Package-P..Word-0]			
	...			
Word S+Q+1	[FFDC Package-P..Word-Q]			
Word S+Q+2	**Distance to Status header at Word N+1 including this word.**			

When both the Primary and Secondary Status indicate a success or good response, there would be no FFDC package between the status header and the last word which points to the location of the status word. A bad Primary or Secondary Status in the status header could indicate the presence of zero or one or more FFDC package after the status header.

2.1.3.4 FFDC Package

FFDC Package				
	Byte 0	***Byte 1***	***Byte 2***	***Byte 3***
Word 0	Magic Bytes: 0xFFDC		Length in words (N+4)	
Word 1	[Sequence ID]		Command-Class	Command
Word 2	Return Code 0..31			
Word 3	FFDC Data – Word 0			
...	...			
Word N+3	FFDC Data – 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 planned to be used on the Host side PSU based 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) or
2. 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.

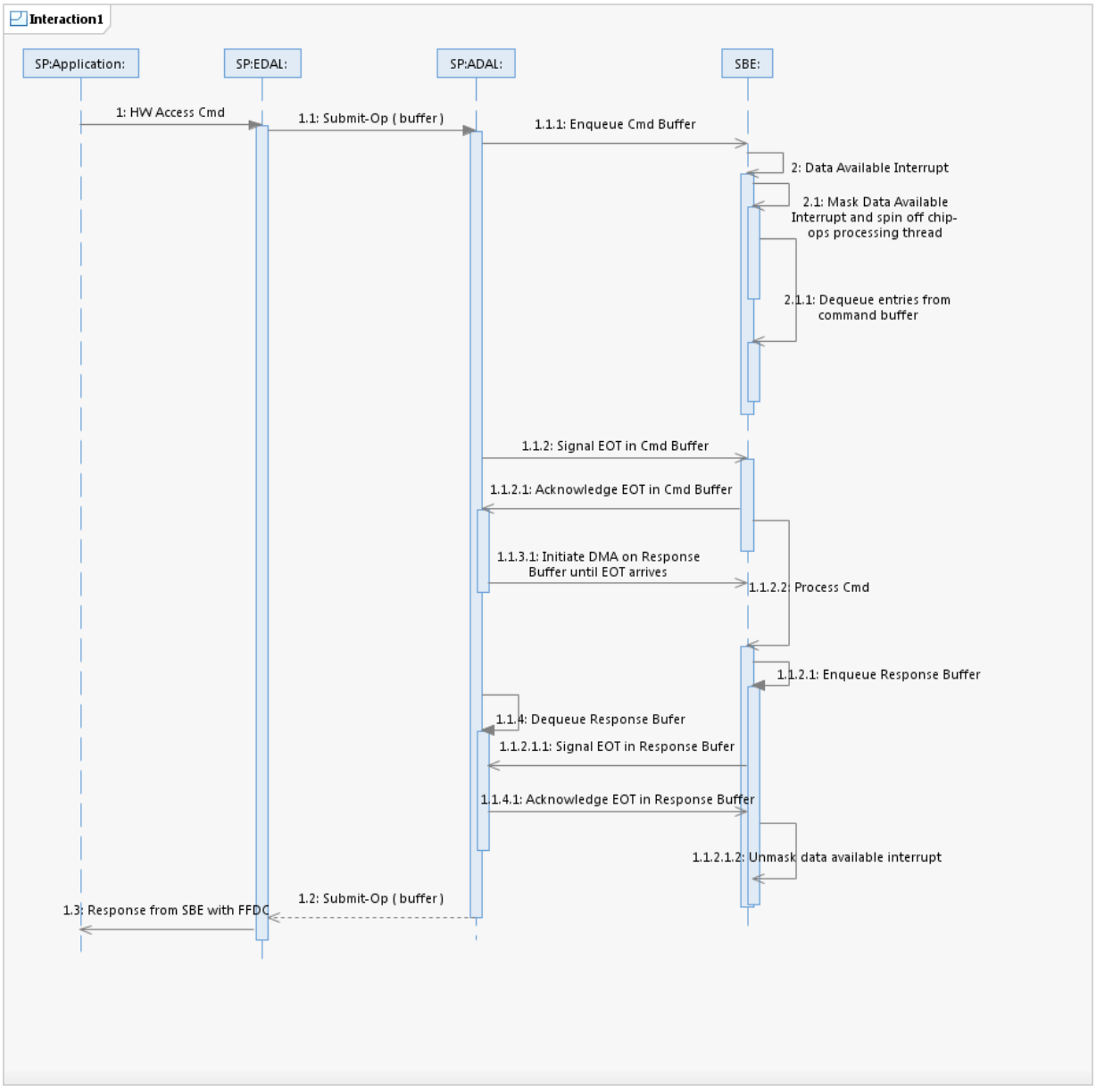
The Return Code at word 2 along with the subsequent FFDC Data Words 0-N are expected to be added verbatim, as an opaque blob, to an error log or a dump by the SP for offline parsing and debug.

The success case response “data” returned for the Get SBE FFDC (0xA801) command will also adhere to the same FFDC package definition starting at word 0 and can contain more than one FFDC package.

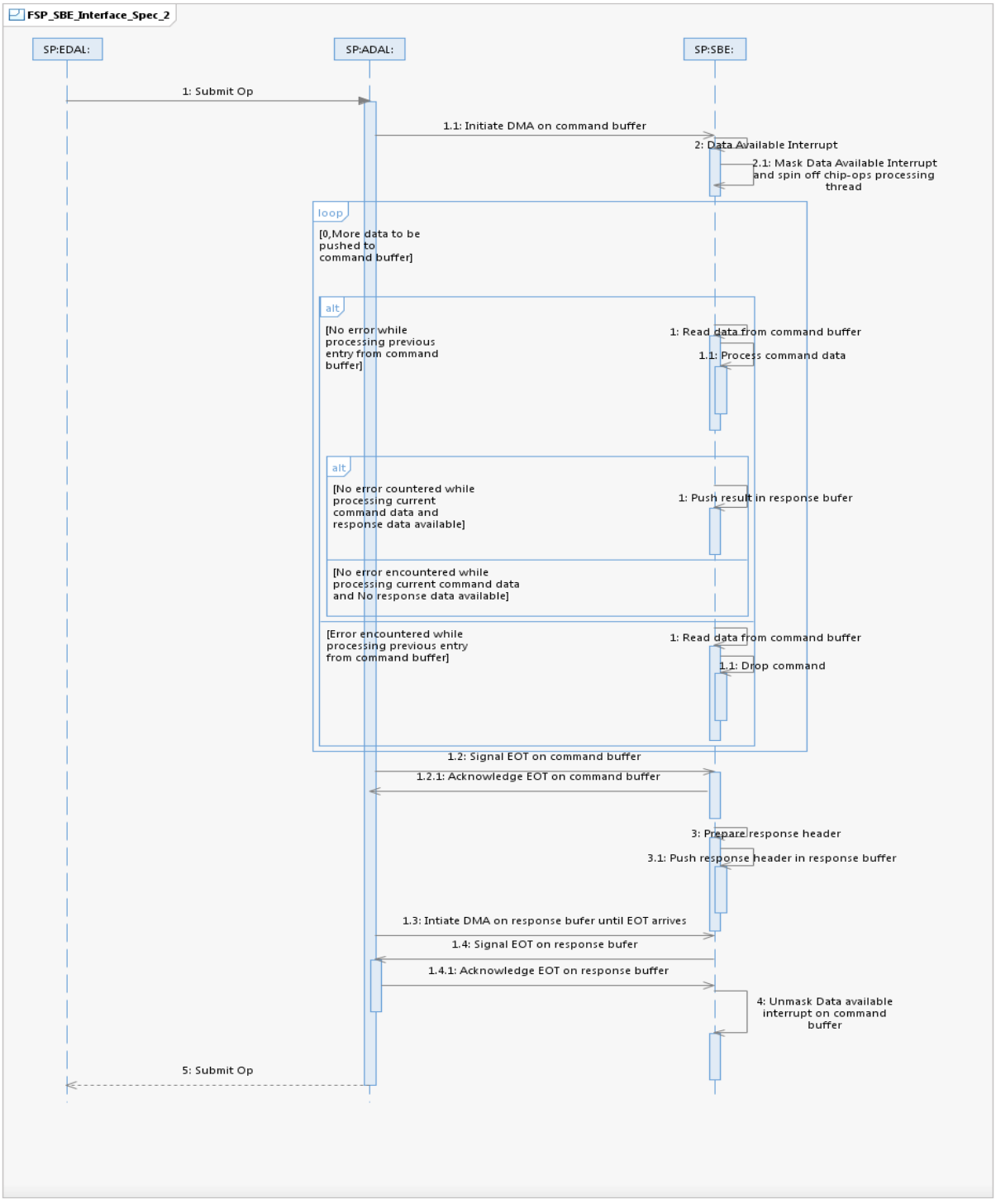
SBE debug utilities will help parse the FFDC package into user readable debug information.

2.1.4 SP SBE Interaction Sequences

2.1.4.1 SP SBE Small Command Interaction Sequence



2.1.4.2 SP SBE Large Command Interaction Sequence



2.1.5 Target Types and Ids Supported

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 are as defined in the table below and map to the pervasive units in the processor hardware. 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, as shown in the table below.

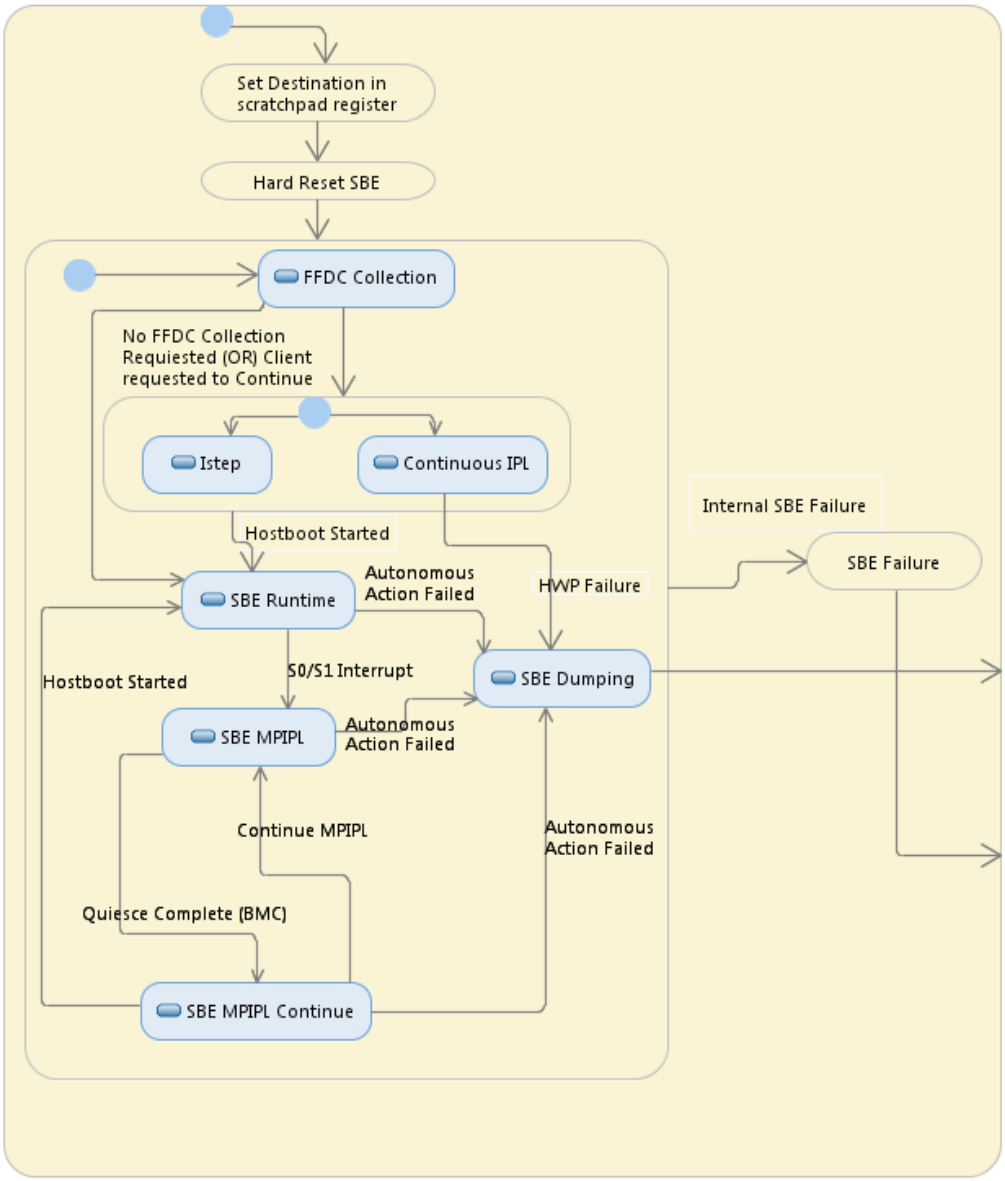
Target Type	*Pervasive Chiplet Id(s)*	*Implied SBE Target*	*Physical Chiplet(s)*	*Notes / Comments*
`TARGET_TYPE_PROC = 0x0000`	`N.A.`	Processor Chip	Any chiplet in the processor	The chiplet id used is a reserved value for processor targets and does not correspond to any particular pervasive chiplet in the processor.
`TARGET_TYPE_PERV = 0x0001`	`0x01`	Pervasive	TP	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	
	`0x08-0x9`	PCI	PCI0-PCI1 respectively	
	`0x0C-0x0xF`	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 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, irrespective of whether the system is configured to 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 to 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. There is no requirement to accept multicast targets via chip-ops.

2.1.6 SBE States

The SBE firmware transitions through software defined states as shown in the state diagram below. Note that the state diagram below is only indicative and may not show all the state transitions.



SBE updates a 4-bit previous state and 4-bit current state fields in the SBE Messaging Register as it transitions across different states. Please refer to the table below for a comprehensive list and details all the SBE states supported.

State	*Value*	*Description*
Unknown	0x0	Unknown, initial state before SBE is started.
IPL	0x1	SBE is in a continuous IPL mode. SBE enters runtime state once hostboot is started. In case of any failures encountered in the “Continuous IPL” state, SBE will reach “Dumping” state.
Istep	0x2	SBE is in istep mode and awaits commands from SP to execute specific isteps. SBE enters runtime state once Hostboot is started. This is a lab only debug configuration and all chip-ops will be allowed as an exception.
MPIPL	0x3	SBE has entered Memory-Preserving IPL state based on request by SP or Host. SBE will reach Runtime state once Hostboot is re-started.
Runtime	0x4	SBE is ready to accept any chip-op commands.
DMT	0x5	`Dead Man Timer` (transient) `State`, where primary SBE waits for all the secondary cores across the system to be initialized during system IPL
Dumping	0x6	SBE encountered an error during any autonomous execution process (e.g., Continuous IPL). This is most likely due to a HW procedure failure during autonomous execution by SBE
Fail	0x7	SBE encountered an internal failure. A Hard Reset of SBE is required to exit this state.
Quiesce	0x8	SBE was issued a Quiesce command and has entered a state where it no longer accepts any commands from the SP (over FIFO) or the Host (over PSU). All asynchronous or background operations are stopped. SBE would eventually need a reset to exit this state.. This state quiesces any SEEPROM access.
Halted	0x9	SBE halted by Host. SBE needs a reset to exist this state.

2.1.7 SBE Configuration

During initialization, the values set up in various Mailbox Scratch Registers (CFAM 28XX, SCOM 0x5003X), configure the SBE for multiple functions like role (Primary or Secondary SBE), system type (native SP based or BMC based), boot mode (istep mode or continuous IPL), destination state on boot (e.g. IPL or Runtime or FFDC), enabling internal FFDC package in responses and so on.

2.1.8 SBE Messaging Register

SBE updates information about its current and previous state to the SBE Messaging Register (part of the SP GP Mailbox) at CFAM address **0x2809** or SCOM address **0x00050009**. The SP can read this register to query the SBE state and other auxiliary information like progress in the SBE IPL steps. The table below lists the bit description of the SBE Messaging Register.

Bit(s)	*Field*	*Description*
0	SBE Booted	SBE kernel is initialized and SBE is ready to accept chip-ops.

Bit(s)	*Field*	*Description*
1	Async FFDC	There was a failure executing asynchronous operations and the SP can collect FFDC via the GetSBEFFDC chip-op. e.g. IPL, MPIPL, DMT. Clears once FFDC is collected.
2-3	Reserved	Reserved
4-7	Previous State	Last state the SBE was in before it entered the current state
8-11	Current State	Current state the SBE is in
12-19	Istep Major	Last major istep executed (irrespective of success or fail). Refer section 'Execute Istep'
20-25	Istep Minor	Last minor istep executed (irrespective of success of fail). Refer section 'Execute Istep'
26-27	Reserved	Reserved
28-31	Boot Progress	4-bit SBE boot progress code, as defined in section 2.1.8 below

2.1.9 SBE Boot Progress Codes

SBE updates a 4-bit boot progress code in the SBE Messaging Register as it boots up to runtime.
The boot codes are as defined in the below table.

Boot Code	**Description**
0x0	SBE not started
0x1	OTPROM boot loader started
0x2	Measurement SEEPROM started
0x3	Measurement SEEPROM PIB memory repair started
0x4	Measurement SEEPROM PIB initialization sequence started
0x5	Measurement SEEPROM TPM reset
0x6	Measurement SEEPROM pibmem started
0x7	Measurement SEEPROM PAU DPLL lock
0x8	Measurement SEEPROM pibmem TPM initialization started
0x9	Measurement SEEPROM SHA512 compute done
Boot Code	**Description**
0xA	Verification pibmem main start
0xB	SBE FW secure header verification complete

0xC	HBBL secure header verification complete
0xD	Boot See prom L1 loader started
0xE	Boot See prom L2 loader started
0xF	PIBMEM application started

3 Summary of SBE FIFO Commands

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 short, long and very Long. The recommended timeout values for short, long and very long running commands are 10 ms, 30 ms and 120 ms respectively.

Cmd Class (hex)	Cmd (hex)	Name	Changed from P9 to P10	Supported States	Timeout
0xA1		IPL Control Messages			
	0x01	Execute istep	NO	Istep	Long
	0x02	Suspend IO	NO	Istep, SBE runtime	Short
0xA2		SCOM Access Messages			
	0x01	Get SCOM	NO	Istep, SBE runtime, SBE dumping	Short
	0x02	Put SCOM	NO	Istep, SBE runtime, SBE dumping	Short
	0x03	Modify SCOM	NO	Istep, SBE runtime, SBE dumping	Short
	0x04	Put SCOM under mask	NO	Istep, SBE runtime, SBE dumping	Short
0xA3		Ring Access Messages			
	0x01	Get Ring	NO	Istep, SBE runtime, SBE MPIPL, SBE dumping	Long
	0x02	Put Ring	NO	Istep, SBE runtime, SBE MPIPL, SBE dumping	Long

0xA4		**Memory Access Messages**			
	0x01	**GetMemory**	**YES**	**Istep, SBE runtime, SBE dumping**	**Long**
	0x02	**PutMemory**	**YES**	**Istep, SBE runtime, SBE dumping**	**Long**
	0x03	**GetSRAM**	**YES**	**Istep, SBE runtime, SBE dumping**	**Long**
	0x04	**PutSRAM**	**YES**	**Istep, SBE runtime, SBE dumping**	**Long**
0xA5		**Register Access Messages**			
	0x01	**Get Architected Register**	**YES**	**Istep, SBE runtime, SBE dumping**	**Long**
	0x02	**Put Architected Register**	**YES**	**Istep, SBE runtime, SBE dumping**	**Long**
	0x03	**Get Hw Register**	**YES**	**Istep, SBE runtime, SBE dumping**	**Short**
	0x04	**Put Hw Register**	**YES**	**Istep, SBE runtime, SBE dumping**	**Short**
0xA6		**Array Access Messages**			
	0x01	**Control Fast Array**	**YES**	**Istep, SBE runtime, SBE dumping, SBE MPIPL**	**Long**
	0x02	**Control Trace Array**	**YES**	**Istep, SBE runtime, SBE dumping, SBE MPIPL**	**Long**
0xA7		**Instruction Control Messages**			

	0x01	**Control Instructions**	**YES [Core Chiplet to Core Instance & no special wake up mode]**	**Istep, SBE runtime**	**Short**
0xA8		**Generic Messages**			
	0x01	**Get SBE FFDC**	**NO**	**Istep, SBE runtime, SBE dumping, SBE MPIPL, SBE Failure, FFDC Collection**	**Short**
	0x02	**Get SBE Capabilities**	**NO**	**All States**	**Short**
	0x03	**Get Supported Frequencies**	**Not Supported**	**N.A.**	**Short**
	0x06	**Quiesce SBE**	**NO**	**All States**	**Long**
	0x07	**Get SBE Capabilities 2**	**YES**	**All States**	**Short**
	0x08	**Get LPC Status**	**YES**	**SBE runtime**	**Short**
0xA9		**MPIPL Commands**			
	0x01	**Enter MPIPL**	**NO**	**SBE runtime**	**Long**
	0x02	**Continue MPIPL**	**NO**	**SBE MPIPL**	**Long**
	0x03	**Stop Clocks**	**YES**	**SBE runtime, SBE MPIPL**	**Short**
	0x4	**Get TI Info**	**YES**	**SBE runtime**	**Long**
0xAA		**Dump Commands**			
	0x01	**Get Dump**	**YES**	**SBE runtime**	**Very Long**

0xAB		**Host FIFO Commands**			
	0x01	**SBE Halt**	**YES**	**SBE runtime**	**Short**
0xAC		**Telemetry Commands**			
	0x01	**PMIC Health Check**	**YES**	**SBE runtime**	**Short**

3.1 Responses to Commands

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 primary and secondary status fields have to be decoded as follows

Primary Status	Bits 0..15 : Primary status codes as defined in Table 4
Secondary Status	Bits 0..15 : Secondary status codes as defined in Table 5

The following table defines the primary status responses

Value	**Definition**
0x0000	Operation Successful.
0x0001	invalid or unsupported command
0x0002	Invalid data passed
0x0003	User Error
0x0004	SBE Internal Error
0x0005	Unsecure Access Denied
0x00FE	Generic Failure in Execution

Table 4: Primary Status Codes

The following table defines the secondary status responses.

Value	**Definition**
0x0000	Operation successful.
0x0001	Command Class not supported
0x0002	Command not supported

0x0003	Invalid address passed
0x0004	Invalid Target Type passed
0x0005	Invalid Chiplet Id passed
0x0006	Target not present
0x0007	Target not functional
0x0008	Command not allowed in current state.
0x0009	Functionality not Supported
0x000A	Generic failure in execution
0x000B	Blacklisted register accessed
0x000C	SBE Operating System Failure
0x000D	SBE FIFO Access Failure
0x000E	Insufficient data passed as part of command
0x000F	Excess data passed as part of command
0x0010	Hardware timeout
0x0011	PCB-PIB Error
0x0012	SBE FIFO Parity Error
0x0013	Unused
0x0014	Blacklisted memory accessed
0x0015	Unsecured memory region not found
0x0016	Exceeded maximum supported unsecured memory regions
0x0017	Unsecured memory region amend attempted

0x0018	Input buffer overflow
0x0019	Invalid parameters
0x0020	Blacklisted chip-op access
0x0021	Deadman Timer time out
0x0022	System check-stop
0x0023	Blacklisted register access blocked
0x0024	Start MPIPL Failed (SP less MPIPL)
0x0025	Stop Clock Failed (SP less MPIPL)
0x0026	Continue MPIPL Failed (SP less MPIPL)
0x027	Periodic IO Toggle failed
0x028	Special Wakeup time out
0x029	Special Wakeup Scom failure
0x02A	Architecture Register dump fail
0x02B	Lpc Access fail
0x02C	Hardware Procedure Fail
0x02D	Special Attention Core Scratch Read Fail
0x02E	Special Attention Data Read Fail
0x02F	Put Sram Fail
0x030	Get Sram Fail
0x031	Thread Control instruction Fail
0x032	Ram Core Setup Fail

0x033	Ram Core Access Fail
0x034	Ram Core Cleanup Fail
0x035	Suspend IO Procedure Fail
0x036	Enter Mpipl Fail
0x037	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 5: Secondary Status Codes

3.2 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.

3.2.1 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.

3.3 Reset / Reload Considerations

It is possible for a reset / reload to occur during a data transfer process. Once the service processor is back from reset/reload condition, it has to issue a SBE FIFO reset request to SBE to clear all pending operations.

3.4 Handling Protocol Violations

Two types of reset requests are supported by SBE.

1. FIFO Reset: The SP can trigger a FIFO reset request by writing to the Upstream Reset Request register of the SBE FIFO device (CFAM 0x2403), which will be acknowledged by the SBE. This mechanism will flush the command and response buffers and re-initialize the hardware state of the FIFO device to accept a new command. Any operation over the FIFO device, which was still in progress, will be aborted by the SBE as a result of the FIFO reset request. As this mechanism requires the SBE to acknowledge the FIFO reset request, it cannot work when the SBE itself is halted or hung due to an unrecoverable error condition. In such cases, where the FIFO reset times out, the SP can choose to use the Hard Reset mechanism described below to recover a belly-up SBE.
2. Hard Reset: The SP can trigger a hard reset of the SBE, by setting bit 12 of the Self-boot Control / Status (SBCS) Register (CFAM 0x2808) in the FSI GP Mailbox device. The SBE stops executing instructions, resets it hardware logic and reboots from the OTPROM. Prior to performing a hard reset of the SBE:
   a) as a hard reset of the SBE flushes out the hardware state of the SBE, which is critical for debug, the SP must collect all possible debug data from the SBE hardware, via the legacy FSI2PIB SCOM engine after unlocked the SBE by setting the Secure Debug Bit. The mechanism to collect the FFDC of a belly-up SBE from the SP is out-of-band to the SBE and hence beyond the scope of this document.
   b) the SP must set the appropriate destination state bits in the Mailbox Scratch Register as described in section 2.1.6. Based on the overall state of the system at the time of the hard reset, the SBE can be configured to reboot in Istep mode, or Continuous IPL mode, or directly enter the runtime state.

Use-case	Actions from SBE	Actions from SP
Command times out at SBE. SBE has dequeued command buffer and acknowledged EOT but no data populated in the response buffer	Upon receiving request for FIFO reset, SBE will collect its FFDC and perform a FIFO reset	SP requests for a SBE FIFO reset.

		Upon completion of FIFO reset, SP then requests for SBE FFDC. If SBE does not reset the FIFOs or if SBE FFDC collection fails, then SP will perform a SBE Hard reset.
SBE does not send EOT as part of response buffer	Upon receiving request for FIFO reset, SBE will collect its FFDC and perform a FIFO reset	SP keeps waiting for more data which results in the command timing out at the SP end resulting in the SP requesting a SBE FIFO reset. Upon completion of FIFO reset, SP then requests for SBE FFDC. If SBE does not reset the FIFOs or if SBE FFDC collection fails, then SP will perform a SBE Hard reset
SP does not send EOT as part of command buffer	SBE continues to process command until EOT arrives. Upon receiving FIFO reset request from SP, SBE will collect its FFDC and reset the FIFO	This will result in the command timing out at the service processor end resulting in the service processor requesting a SBE FIFO reset. Upon completion of FIFO reset, SP will collect SBE FFDC. If SBE does not reset the FIFOs or if SBE FFDC collection fails, then SP will perform a SBE Hard reset
SP does not dequeue from response buffer	SBE waits indefinitely for SP to pull data. Upon receiving FIFO reset request from SP, SBE will collect its FFDC and reset the FIFO	This will result in the command timing out at the service processor end resulting in the service processor requesting a SBE FIFO reset. Upon completion of FIFO reset, SP will collect SBE FFDC. If SBE does not reset the FIFOs or if SBE FFDC collection fails, then SP will perform a SBE Hard reset
SBE does not dequeue data from command buffer		This will result in the command timing out at the service processor end resulting in the service processor requesting a SBE FIFO Reset.

	Upon receiving FIFO reset request from SP, SBE will collect its FFDC and reset the FIFO	Upon completion of FIFO reset, SP will collect SBE FFDC. If SBE does not reset the FIFOs or if SBE FFDC collection fails, then SP will perform a SBE Hard reset
SP does not acknowledge EOT in response buffer	SBE waits indefinitely for SP to acknowledge EOT in response buffer. Upon receiving FIFO reset request from SP, SBE will collect its FFDC and reset the FIFO	This will result in the command timing out at the service processor end resulting in the service processor requesting a SBE FIFO reset. Upon completion of FIFO reset, SP will collect SBE FFDC.
SBE does not acknowledge EOT in command buffer	Upon receiving FIFO reset request from SP, SBE will collect its FFDC and reset the FIFO	This will result in the command timing out at the service processor end resulting in the service processor requesting a SBE FIFO Reset. Upon completion of FIFO reset, SP will collect SBE FFDC. If SBE does not reset the FIFOs or if SBE FFDC collection fails, then SP will perform a SBE Hard reset
SP sends invalid command class or command or an inconsistent payload.	Upon detecting invalid command-class and/or invalid command or inconsistent command payload, SBE will read the upstream FIFO until EOT is received and discard the entries from upstream FIFO. Once EOT is acknowledged, SBE will send response data with appropriate status.	

4 IPL Control Messages

4.1 Execute Istep

This command is sent by SP to SBE to execute a specific istep. The istep to execute is specified by the istep Major and Minor numbers passed as part of the command. This is a synchronous command. SBE will enqueue the response buffer after completing execution of the command. Note that this chip-op is debug only to support lab testing and bring-up. This command is accepted only when the SBE started after configuring it in the istep mode, by setting bit 0 of the FW Control flags register (Mailbox scratch 3 - CFAM 283A, SCOM 0x5003A).

Message (request) from SP			
0x03			
reserved	reserved	**0xA1**	**0x01**
reserved	**Istep Major Number**	reserved	**Istep Minor Number**

Bits 8-15: Istep Major Number
Istep Major Numbers 2 to 5
MPIPL Major Numbers 96 - 98

Bits 24:31: Istep Minor Number
Istep Minor Numbers : x..y
MPIPL Minor Numbers: p..q

Note that the SBE control loop is not ready to accept chip-ops up to istep 2.1. Hence only isteps 2.2 till 5.3 are supported via this command.

Response from SBE		
0xC0DE	**0xA1**	**0x01**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
[..]		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

4.2 Suspend IO

This command is sent by SP to SBE to suspend IO traffic from and to the processor chip on which the SBE is located. This is a synchronous command where SBE en queues the response after completing the operation.

Message (request) from SP			
0x02			
reserved	reserved	**0xA1**	**0x02**

Response from SBE		
0xC0DE	**0xA1**	**0x02**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
[..]		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status: See section 3.1		

5 SCOM Access Messages

Note that within the context of a chip-op, the SBE will atomically attempt a retry / recovery for SCOM accesses failing with timeout or parity errors on the PCB/PIB network. On errors, PCB/PIB error code will be returned in the chip-op response as part of its FFDC package.

5.1 Get SCOM

This command is sent by SP to SBE to perform a SCOM read operation on a specified address. It should be noted that no address translation is performed on SBE. Hence, absolute SCOM addresses must be passed in this command.

Message (request) from SP			
0x04			
reserved	reserved	**0xA2**	**0x01**
Register Address (0..31)			
Register Address (32..63)			

Response from SBE		
Register Data (0..31)		
Register Data (32..63)		
0xC0DE	**0xA2**	**0x01**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
[..]		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

5.2 Put SCOM

This command is sent by SP to SBE to perform a SCOM write operation on a specified register address.

Message (request) from SP			
0x06			
reserved	reserved	**0xA2**	**0x02**
Register Address (0..31)			
Register Address (32..63)			
Register Data (0..31)			
Register Data (32..63)			

Response from SBE		
0xC0DE	**0xA2**	**0x02**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
[..]		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

5.3 Modify SCOM

This command is sent by SP to SBE to execute read+modify+write operation to a SCOM register.

Message (request) from SP			
0x07			
reserved	reserved	**0xA2**	**0x03**
reserved	reserved	reserved	**Operation**
Register Address (0..31)			
Register Address (32..63)			
Modifying Data (0..31)			
Modifying Data (32..63)			
Bits 28-31: Operation **0 – No Operation. Same as Put SCOM** **1 – OR** **2 – AND** **3 - XOR**			

Response from SBE		
0xC0DE	**0xA2**	**0x03**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
[..]		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

5.4 Put SCOM Under Mask

This command is sent by SP to SBE to execute a SCOM write operation under mask. This is a synchronous command where SBE enqueues the response after completing the operation.

The following steps are done as part of this command

1. Read Register Data
2. 'AND' the Mask with the data read from register
3. 'OR' the modifying data with the result of step 2
4. Write the result of step 3 into the register

Message (request) from SP			
0x08			
reserved	reserved	**0xA2**	**0x04**
Register Address (0..31)			
Register Address (32..63)			
Modifying Data (0..31)			
Modifying Data (32..63)			
Mask Data (0..31)			
Mask Data (32..63)			

Response from SBE		
0xC0DE	**0xA2**	**0x04**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
[..]		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

5.5 Execute Multi SCOM

This command is sent by SP to SBE to execute a group of SCOM operations. This is a synchronous command where SBE will start populating the response buffer once the operation is completed. Only the following SCOM operations are allowed as part of this command.

1. Get SCOM
2. Put SCOM
3. Modify SCOM
4. Put SCOM Under Mask

A maximum of 32 SCOM operations (any combination of the above mentioned operations) are allowed as part of this command. SBE will stop executing the operations upon encountering the first failure.

Message (request) from SP			
Length			
reserved	reserved	**0xA2**	**0x05**
Number of SCOM Operations (M)			
SCOM Operation Package1..1			
..			
SCOM Operation Package1..N			
..			
SCOM Operation PackageM..1			
..			
SCOM Operation PackageM..X			
SCOM Operation Package : See the tables below			

Get SCOM Operation Package			
Magic Bytes : 0xFACE		**Length**	**0x01**
reserved	**reserved**	**Operation Sequence #**	
Register Address (0..31)			

Register Address (32..63)
Length : Includes the word containing the length itself

Put SCOM Operation Package			
Magic Bytes : 0xFACE		**Length**	**0x02**
reserved	**reserved**	**Operation Sequence #**	
PCBPIB Error Mask			
Register Address (0..31)			
Register Address (32..63)			
Register Data (0..31)			
Register Data (32..63)			
Length : Includes the word containing the length itself			

Modify SCOM Operation Package			
Magic Bytes : 0xFACE		**Length**	**0x03**
reserved	reserved	reserved	**Operation**
reserved	**reserved**	**Operation Sequence #**	
Operational Mode			
PCBPIB Error Mask			
Register Address (0..31)			
Register Address (32..63)			
Register Data (0..31)			
Register Data (32..63)			

Modifying Data (0..31)
Modifying Data (32..63)
Length : Includes the word containing the length itself **Bits 28-31: Operation:** **0 – No Operation. Same as Put SCOM** **1 – OR** **2 – AND** **3 - XOR**

Put SCOM Under Mask Operation Package			
Magic Bytes : 0xFACE		**Length**	**0x04**
reserved	**reserved**	**Operation Sequence #**	
PCBPIB Error Mask			
Register Address (0..31)			
Register Address (32..63)			
Register Data (0..31)			
Register Data (32..63)			
Modifying Data (0..31)			
Modifying Data (32..63)			
Mask Data (0..31)			
Mask Data (32..63)			
Length : Includes the word containing the length itself			

Response from SBE
Response data for SCOM Operation Package1..1
..

Response data for SCOM Operation Package1..N		
..		
Response data for SCOM Operation PackageM..1		
..		
Response data for SCOM Operation PackageM..X		
0xC0DE	**0xA2**	**0x05**
Primary Status	**Secondary Status**	
Number of SCOM Operations executed		
First Failed SCOM Address (0..31)		
First Failed SCOM Address (32..63)		
[FFDC Package 0]		
[..]		
[FFDC Package Y]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1 **If the all the operations were successful, then the First Failed SCOM Address field will be populated as 0.**		

Response data package for Get SCOM Operation		
Magic Bytes : 0xFACE	**length**	0x01
reserved **reserved**	**Operation Sequence #**	
Register Data (0..31)		
Register Data (32..63)		

Response data package for Put SCOM Operation		
Magic Bytes : 0xFACE	**length**	0x02
reserved **reserved**	**Operation Sequence #**	

Response data package for Modify SCOM Operation			
Magic Bytes : 0xFACE		**length**	0x03
reserved	**reserved**	**Operation Sequence #**	

Response data package for Put SCOM Under Mask Operation			
Magic Bytes : 0xFACE		**length**	0x04
reserved	**reserved**	**Operation Sequence #**	

6 Ring Access Messages

6.1 Get Ring

This command is sent by SP to SBE to scan read from a ring specified by the ring address on a target corresponding to the target type on the processor where the SBE is placed. The SBE will not do scan address translation and hence does not expect a target type or chiplet id in the command. The higher level firmware is expected to pass a fully translated, absolute ring address. The SBE will stream back the ring data in 64 bit granules or multiples of that.

Message (request) from SP			
0x05			
reserved	reserved	**0xA3**	**0x01**
Ring Address (0..31)			
Length of the Ring in multiple of 64 bits (N)			
Reserved		**Ring Mode**	

Ring Address:
Absolute address of the ring from engineering data.

Bits 16-31: 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
0x0020: Fast Array

Response from SBE		
Ring Data..(0..31)		
Ring Data..(32..63)		
..		
Ring Data..(N-32..N-1)		
Length of the ring data returned in bits		
0xC0DE	**0xA3**	**0x01**
Primary Status	**Secondary Status**	
[FFDC Package..0]		

[..]
[FFDC Package..N]
Distance to status header word with magic byte 0xC0DE
Primary Status, Secondary Status : See section 3.1

6.2 Put Ring

This command is sent by SP to SBE to scan write into a ring specified by the RS4 ring container. The RS4 header should have all details like ring address, region, type and length to scan in a particular chiplet on the processor where the SBE is placed. The RS4 container will also have the compressed data payload to be scanned in to the chip.. It is the responsibility of the caller to ensure that the RS4 header has a fully translated, absolute ring address for the instance of the chiplet being scanned. This is a synchronous command where SBE starts to enqueue the response buffer once the operation is completed.

Message (request) from SP			
Length (N+3)			
reserved	**reserved**	**0xA3**	**0x02**
Reserved		**Ring Mode**	
RS4 Ring Container Word..0			
..			
RS4 Ring Container Word..N-1			

Bits 16-31: 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
0x0040: Apply Overrides

Response from SBE		
0xC0DE	**0xA3**	**0x02**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
[..]		

[FFDC Package..N]
Distance to status header word with magic byte 0xC0DE
Primary Status, Secondary Status : See section 3.1

7 Memory Access Messages

7.1 Memory Access Modes

Main-store memory and cache can be accessed via the ADU or PBA hardware blocks. The below table summarizes the various flags used by the memory access chip-ops (0xA401 and 0xA402) to select the required hardware interface, access type and other access modes

Flag	*Value*	*Comments*	*Interface*	*Access*	*Constraints*
ADU	0x0001	Use Alter/Display Unit to access memory	ADU	R+W	Default data sizes supported are in multiples of 8 bytes granule size
PBA	0x0002	Use the Power Bus Access Unit to access memory	PBA	R+W	Default data sizes supported are in multiples of 128 bytes granule size
Auto Increment	0x0004	Enable automatic address increment in hardware	ADU / PBA	R+W	Not applicable on Cache Inhibited ADU write access
ECC Required / ECC Override	0x0008	On reads, every 8 bytes of data is followed by 1 byte of ECC. On writes, user specified ECC byte overrides HW generated ECC	ADU	R / W	If both ECC and TAG flags are set, every 8 bytes of data (byte#0-byte#7) is followed by 1 byte (byte#8) of TAG data followed by 1 byte (byte#9) of ECC data.
TAG Required	0x0010	Every 8 bytes of data is followed by 1 byte of TAG data	ADU	R+W	
Fast Mode	0x0020	Enable fast mode in hardware	ADU / PBA	R+W	None
LCO	0x0040	Enable lateral cast out mode	PBA	W	Applicable only to PBA writes
Cache Inhibit	0x0080	Bypass the cache e.g. for memory or registers of an IO adapter	ADU	R+W	Only 1, 2, 4 and 8 byte data access supported in this mode. This mode does not support the default data access sizes of multiples of 8 bytes.
Host Pass-through	0x0100	Read-from / write-to a pre known address and notify the Host on writes	PBA	R+W	Host has already set the Pass-through command / response address via the PSU 'SetFFDCAddress' chip-op
Cache Inject	0x0200	Use the cache inject mode	PBA	W	Applicable only to PBA writes
Pre switch AB	0x0400	Do no issue PB cmd, pre-set for AB operation	ADU	W	Applicable only to ADU writes
Pre switch CD	0x0800	Do no issue PB cmd, pre-set for CD operation	ADU	W	Applicable only to ADU writes

Flag	*Value*	*Comments*	*Interface*	*Access*	*Constraints*
Post switch	0x1000	Clear switch CD/AB flags	ADU	W	Applicable only to ADU writes

7.2 GetMemory

This command is sent by SP to SBE to read data starting at a specified address from the mainstore memory attached to the processor on which this SBE is placed or from L3 cache in the processor in which this SBE is placed. SP must ensure that clocks are running. The base or start address must be granule size aligned and the length of data to be read should be a multiple of the granule size.

Message (request) from SP			
0x06			
reserved	**reserved**	**0xA4**	**0x01**
Reserved		**Flags**	
Memory address (0..31)			
Memory address (32..63)			
Length of data to be read in bytes			
Bits 0-15: Reserved **Bits 16-31: Flags :** **Refer section 7.1 for details of the flags supported and their intended usage.** **LCO (0x0040) and Cache Inject (0x0200): Not applicable on read access.**			

Response from SBE		
Memory Data..1		
..		
Memory Data..N		
Length of Data read in bytes (includes ECC and Tag data)		
0xC0DE	**0xA4**	**0x01**

Primary Status	Secondary Status
[FFDC Package..0]	
..	
[FFDC Package..N]	
Distance to status header word with magic byte 0xC0DE	
Primary Status, Secondary Status : See section 3.1	

7.3 PutMemory

This command is sent by SP to SBE to write data starting at a specified address in the mainstore memory attached to the processor on which this SBE is placed or in the L3 cache attached to the processor on which this SBE is placed. SP must ensure that clocks are running. The base or start address must be granule size aligned and the length of data to be written should be a multiple of the granule size.

Message (request) from SP			
Length			
reserved	**reserved**	**0xA4**	**0x02**
[Core Chiplet Id]	**[ECC Override byte]**	**Flags**	
Memory address (0..31)			
Memory address (32..63)			
Length of data to be written in bytes (Does not include ECC)			
Mainstore Data..1			
..			
Mainstore data..N			

Bits 0-7: Core instance Id:

0x00-0x1F: The core id (even or odd core) belonging to the logical EQ target on which the memory is to be written. Note that this is applicable only on PBA write-only operations with the LCO flag asserted

Bits 8-15: ECC Override Byte:

The ECC value which will be inserted as ECC for every 8 bytes of data written. Applicable only when the ECC Override flag is set on ADU operations.

Bits 16-31: Flags:

Refer section 7.1 for details of the flags supported and their intended usage.

Response from SBE		
Length of Mainstore Data actually written in bytes (includes ECC and TAG)		
0xC0DE	0xA4	0x02
Primary Status	Secondary Status	
[FFDC Package..0]		
..		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

7.4 GetSRAM

This command is sent by SP to SBE to read data from different SRAM chips of the processor in which this SBE is placed. SP must ensure that clocks are running. The support SRAM which SBE can read are IOPPEs, QMEs, PCIs and OCC. The SRAM address from where data is to be read must be 8B aligned.

Note: User is required to pass the mode and channel to be used for the Occ Sram access.

Message (request) from SP			
0x06			
reserved	**reserved**	**0xA4**	**0x03**
Perv Chiplet Id		**reserved**	**Mode**
SRAM Address (0..31)			
SRAM Address (32..63)			
Length of data to be read in multiples of 8B			

Perv Chiplet (bit 0-15): Supports PAUC, EQ, PCI and Proc Target for Occ

Reserved (bit 16-23)

Mode (bit 24-31): [Applicable for OCC & PCI]

For OCC : Bit 24-25 : Mode

0b01: Normal
0b10: Debug
0b11: Circular

Bit 26-28 : Channel Number

0b001: OCB_CHAN0
0b010: OCB_CHAN1
0b011: OCB_CHAN2
0b100: OCB_CHAN3

Bit 29-31 : Unused

For PCI : Bit 24 : io_top_number

0b0 : io_top_0
0b1 : io_top_1

Bit 25 : PHYP

0b0 : Phyp0
0b1 : Phyp1

Bit 26-31 : Unused

Response from SBE		
SRAM Data..1		
..		
SRAM Data..N		
Length of SRAM Data read		
0xC0DE	0xA4	0x03
Primary Status	Secondary Status	
[FFDC Package..0]		
..		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

7.5 PutSRAM

This command is sent by SP to SBE to write data to different SRAM of the processor in which this SBE is placed. The support SRAM which SBE can write are IOPPEs, QMEs, PCIs and OCC. SP must ensure that clocks are running. The SRAM address at which data is to be written must be 8B aligned.

The address range at which data is to be written must be from a white-listed secure window. In circular access mode, the address field will be ignored.

Note: User is required to pass the mode and channel to be used for the Occ Sram access.

Message (request) from SP			
0x6 + N			
reserved	**reserved**	**0xA4**	**0x04**
Perv Chiplet Id	**Mcast Bit**	**reserved**	**Mode**
SRAM address (0..31)			
SRAM address (32..63)			
Length of data to be written in multiples of 8B			
SRAM Data..1			
..			
SRAM data..N			

Perv Chiplet (bit 0-15): Supports PAUC, EQ, PCIe and Proc Target for OCC

M-Cast (bit 16) : Only applicable for PAUC, QME and PCIe chiplets

0b0 - UniCast Access
0b1 – Multicast Access

Reserved (Bit 17-23)

Mode (bit 24-31): [Applicable for OCC & PCI]

For OCC : Bit 24-25 : Mode
0b01: Normal
0b10: Debug
0b11: Circular

Bit 26-28 : Channel Number

0b001: OCB_CHAN0
0b010: OCB_CHAN1
0b011: OCB_CHAN2

0b100: OCB_CHAN3 **Bit 29-31 : Unused** **For PCIe : Bit 24 : io_top_number** 0b0 : io_top_0 0b1 : io_top_1 **Bit 25 : PHYP** 0b0 : Phyp0 0b1 : Phyp1 **Bit 26-31 : Unused**

Response from SBE		
Length of SRAM Data actually written		
0xC0DE	**0xA4**	**0x04**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		
[FFDC Package..M]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

8 Register Access Messages

8.1 Get Architected Register

This command is sent by SP to SBE to read a list of architected registers of a specified type (Special Purpose, General purpose, Floating Point) in a SMT thread of a Core Instance Id. This is a synchronous command where SBE en-queues the response buffer upon collecting data from the registers.

Message (request) from SP				
Length (3+N)				
Reserved	Reserved	0xA5		0x01
Reserved	Core Instance Id	Thread Nr.	Reg Type	Nr. of Registers (N)
Register Nr..1				
..				
Register Nr..N				

Bits 0-7: Reserved

Bits 8-15: Core Instance Id

0x00 - 0x1F: SMT4 Core Instance Id

Bits 16-19: SMT Thread Number

0x0: Thread 0
0x1: Thread 1
0x2: Thread 2
0x3: Thread 3

Bits 20-23: Register Type

0x0: GPR
0x1: SPR
0x2: FPR

Bit 24-31: Number of Registers (0<N<=64)
Total number of registers to be read as part of this command. Current memory limitations in the SBE allow up to a maximum of 64 registers of a given type.

Register Number(s):
Register number(s) should be related to the register type and are as specified in the POWER ISA at https://openpowerfoundation.org/specifications/isa/

Register Data Package
Register Data (bits 0-31)
Register Data (bits 32-63)

Response from SBE		
Register Data Package 1		
..		
Register Data Package N		
0xC0DE	0xA5	0x01
Primary Status	Secondary Status	
[FFDC Package..0]		
..		
[FFDC Package..M]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1 Register Data 1..N will be returned in the same order of register numbers passed via command.		

8.2 Put Architected Register

This command is sent by SP to SBE to write to a list of architected registers of a specified type (Special Purpose, General purpose, Floating Point) in a SMT thread of a Core instance Id. This is a synchronous command where SBE en-queues response buffer after writing data to the register(s).

Register Package
Register Number
Register Data (bits 0-31)
Register Data (bits 32-63)

Message (request) from SP				
Length 3 + (N * 3)				
Reserved	**Reserved**	**0xA5**		**0x02**
Reserved	**Core Instance Id**	**Thread Nr,**	**Reg. Type**	**Nr. of Registers (N)**
Register Package..1				
..				
Register Package..N				

Bits 0-7: Reserved

Bits 8-15: Core Instance Id

0x00 - 0x1F: SMT4 Core Instance Id

Bits 16-19: SMT Thread Number

0x0: Thread 0
0x1: Thread 1
0x2: Thread 2
0x3: Thread 3

Bits 20-23: Register Type

0x0: GPR
0x1: SPR
0x2: FPR

Bit 24-31: Number of Registers (0<N<=64)

Total number of registers to be written as part of this command. Current memory limitations in the SBE allow up to a maximum of 64 registers of a given type.

Register Package(s):

Contains register number (related to the register type and as specified in the POWER ISA / Programmer's Manual) and 64 bit data to be written.

Response from SBE		
0xC0DE	**0xA5**	**0x02**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		
[FFDC Package..M]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

8.3 Get HW Register

This command is sent by SP to SBE to perform a direct HW register read operation on a specified address. Based on the target type, SBE will call direct scom for proc chip or OCMB chip basis the instance Id. It should be noted that no address translation is performed by SBE. Hence, absolute HW addresses must be passed in this command.

Message (request) from SP			
0x05			
Reserved	Reserved	0xA5	0x03
Target Type		reserved	Instance Id
HW Register Address (0..31)			
HW Register Address (32..63)			

Bits 0-15: Target Type

See section 2.1.4
Supported Targets are Proc and Ocmb

Bits 15-23 : Reserved

Bits 24-31: Ocmb Instance Id

Ocmb Id range for a processor Chip
0x00 - 0x0F

Response from SBE		
HW Register Data (0..31)		
HW Register Data (32..63)		
0xC0DE	0xA5	0x03
Primary Status	Secondary Status	
[FFDC Package..0]		
..		
[FFDC Package..M]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

8.4 Put HW Register

This command is sent by SP to SBE to perform a direct HW register write operation on a specified address. Based on the target type, SBE will call direct scom for proc chip or OCMB chip basis the instance Id. It should be noted that no address translation is performed by SBE. Hence, absolute HW addresses must be passed in this command.

Message (request) from SP			
0x07			
Reserved	**Reserved**	**0xA5**	**0x04**
Target Type		**reserved**	**Instance Id**
HW Register Address (0..31)			
HW Register Address (32..63)			
HW Register Data (0..31)			
HW Register Data (32..63)			

Bits 0-15: Target Type

See section 2.1.4
Supported Targets are Proc and Ocmb

Bits 15-23 : Reserved

Bits 24-31: Ocmb Instance Id
Ocmb Id range for a processor Chip
0x00 - 0x0F

Response from SBE		
0xC0DE	**0xA5**	**0x04**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		
[FFDC Package..M]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

9 Array Access Messages

9.1 Control Fast Array

This command is sent by SP to SBE to stream out a specified fast array from a specified EC/Pervasive Target. This is a synchronous command where SBE en-queues the response buffer upon executing the requested action on from the specified Fast arrays.

SP must ensure that the Core is powered ON, clocks to the Core are turned OFF and Core is functional.

Message (request) from SP			
0x04 + N			
Reserved	**Reserved**	**0xA6**	**0x01**
Target Type		**Chiplet Id**	**Control_set**
Length of Custom Override Data in Bytes			
Custom Override Data1			
...			
Custom Override DataN			

Word 2 Bits 0-15: Target Type
Fast arrays will be supported for Core and Pervasive targets.
See section 2.1.4

Word 2 Bits 16-23: ChipletId for Pervasive Targets, Instance Id for Core Target
See section 2.1.4

Word 2 Bits 24-31: Control_set
0x00: Custom Data
0x01: ec_cl2_far
0x02: ec_mma_far

Words 3: Length of custom override data

Custom override data length pass by the user

Word 4..4+N: Custom Override Data

Note: if Control_set is ec_cl2_far / ec_mma_far, length field to be 0 and no custom data expected.

Custom Data & length field is only to be used with control_set as 0.

Response from SBE

Fast Array Data 1		
...		
Fast Array Data N		
Length of the Fast Array Data in Bytes		
0xC0DE	**0xA6**	**0x01**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

9.2 Control Trace Array

This command is sent by SP to SBE to reset/restart/stop/collect a Trace Array pointed to by the Trace Array Id from the target indicated by the Target Type and Chiplet Id. This is a synchronous command and SBE enqueues the response after completing the specified operation on the specified trace array.

Message (request) from SP			
0x04			
Reserved	**Reserved**	**0xA6**	**0x02**
Target Type		**Reserved**	**Chiplet Id**
Trace Array Id		**Operation**	

Word 2, Bits 0-15: Target Type
See Section 2.1.4

Word 2, Bits 24-31: Chiplet Id
See Section 2.1.4

Word 3, Bits 0-15: Trace Array Id
Refer enumeration `proc_gettracearray_bus_id` in file at: `import/chips/p9/procedures/hwp/perv/p9_tracearray_defs.H`

Word 3, Bits 16-31:	Operation
0x0001:	Reset Trace Array
0x0002:	Restart Trace Array
0x0004:	Stop Trace Array
0x0008:	Collect Trace Dump
0x0010:	Ignore Mux Setting

Other Values : Reserved

Response from SBE		
[Trace Array Data Word 0]		
...		
[Trace Array Data Word N]		
Length of the data read in words (N+1)		
0xC0DE	**0xA6**	**0x02**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		

[FFDC Package..N]
Distance to status header word with magic byte 0xC0DE
Primary Status, Secondary Status : See section 3.1

10 Instruction Control Messages

10.1 Control Instructions

This command is sent by SP to SBE to start/stop/sreset/step instructions on a specific SMT thread of a specific Core Target. This is a synchronous command where SBE will enqueue its response after completing the specified operation. The special wakeup mode is taken off from the user control and will be handled by SBE within the chip-op. Here are the use-case how SBE is going to handle the special wakeup on cores -

During stop instruction chip-op, SBE asserts the special awake up on the requested core and does stop instruction on the requested core. Keeps the special wakeup asserted.

During start/sreset instruction chip-op, SBE does starts / sreset instruction on the requested core and de-asserts the special wakeup on the request core if it is already asserted.

During step instruction chip-op, SBE does step instruction on the requested core. No special wakeup assert/de-assert.

Message (request) from SP				
0x03				
Reserved	**Reserved**	**0xA7**	**0x01**	
Reserved	**Mode**	**Core Instance Id**	**Thread Nr.**	**Thread Op.**

Bits 0-11: Reserved

Bits 12-15: Mode:
0x0: Bail out on first error
0x1: Ignore hardware errors, attempt best case

Bits 16-23: Core Chiplet Id:
0x00-0x1F: SMT4 Core Instance Id
0xFF: All SMT4 Cores

Bits 24-27: Thread Nr.:
0x0-3: SMT4 thread number
0xF: All 4 threads in the SMT4 core

Bits 28-31: Thread Op.:
0x0: Start Instructions
0x1: Stop Instructions
0x2: Step Instructions
0x3: Sreset Instructions

Response from SBE		
0xC0DE	**0xA7**	**0x01**
Primary Status	**Secondary Status**	

[FFDC Package..0]
..
[FFDC Package..N]
Distance to status header word with magic byte 0xC0DE
Primary Status, Secondary Status : See section 3.1

11 Generic Messages

11.1 Get SBE FFDC

This command is sent by SP to SBE to retrieve SBE FFDC. This is a synchronous command where SBE will enqueue its response after completing the command. This command returns FFDC corresponding to the internal SBE operations and internal hardware procedure FFDC, if available. The response data sent back conforms to the same FFDC package format as described in FFDC Package section of this document.

The typical use-case of this command is to get FFDC from any asynchronous operation failure in the SBE, when the SBE enters the Dumping state. In the Dumping state, if processing this command also fails, there would be no additional FFDC package(s) in the response that point to the reason of this command's failure. This can be considered as a double-fault in the SBE.

Message (request) from SP			
0x02			
reserved	reserved	**0xA8**	**0x01**

Response from SBE		
FFDC Package..0		
..		
[FFDC Package..N]		
0xC0DE	**0xA8**	**0x01**
Primary Status	**Secondary Status**	
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

11.2 Get SBE Capabilities

This command is sent by SP to SBE to retrieve the SBE 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. This is a synchronous command where SBE will enqueue its response after completing the command.

Message (request) from SP			
0x02			
reserved	reserved	**0xA8**	**0x02**

Response from SBE			
Byte 0	**Byte 1**	**Byte 2**	**Byte 3**
Major Version		**Minor Version**	
8 Hex Character SBE Firmware GIT Commit ID			
SBE Firmware Release Tag..0			
..			
SBE Firmware Release Tag ..4			
Capabilities..0			
..			
Capabilities..23			
0xC0DE		**0xA8**	**0x02**
Primary Status		**Secondary Status**	
[FFDC Package..0]			
..			
[FFDC Package..N]			
Distance to status header word with magic byte 0xC0DE			
Primary Status, Secondary Status : See section 3.1 **Major Version : Increments whenever there is a new chip-op added** **Minor Version : Increments whenever the format of existing Chip-op is changed** **Both Major and Minor versions are defined in section**			

2.1.2.3

Capabilities..0	0xC0000001	HWP FFDC Collection supported
	0xC0000002	SBE FFDC Collection supported
	0xC0000004	Address Blacklisting 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	0xA1000001	Execute Istep - Supported
	0xA1000002	Suspend IO Supported
Capabilities..3	0xA1800000	Reserved for capabilities of A1 class
Capabilities..4	0xA2000001	Get Scom - Supported
	0xA2000002	Put Scom - Supported
	0xA2000004	Modify Scom - Supported
	0xA2000008	Put Scom Under Mask - Supported
	0xA2000010	Multi Scom - Supported
Capabilities..5	0xA2800000	Reserved for capabilities of A2 class
Capabilities..6	0xA3000001	Get Ring - Supported
	0xA3000002	Put Ring - Supported
	0xA3000004	Put Ring from Image - Supported
Capabilities..7	0xA3800000	Reserved for capabilities of A3 class
Capabilities..8	0xA4000001	Get Memory - Supported
	0xA4000002	Put Memory - Supported
	0xA4000004	Get OCC SRAM - Supported

	0xA4000008	Put OCC SRAM – Supported
Capabilities..9	0xA4800000	Reserved for capabilities of A4 class
Capabilities..10	0xA5000001	Get Register Supported
	0xA5000002	Put Register Supported
Capabilities..11	0xA5800000	Reserved for capabilities of A5 class
Capabilities..12	0xA6000001	Read Fast Array Supported
	0xA6000002	Control Trace Array Supported
Capabilities..13	0xA6800000	Reserved for capabilities of A6 class
Capabilities..14	0xA7000001	Control Instructions Supported
Capabilities..15	0xA7800000	Reserved for capabilities of A7 class
Capabilities..16	0xA8000001	Get SBE FFDC Supported
	0xA8000002	Quiesce SBE Supported
Capabilities..17	0xA8800000	Reserved for capabilities of A8 class
Capabilities..18	0xA9000001	Enter MPIPL Supported
	0xA9000002	Continue MPIPL Supported
	0xA9000004	Stop Clocks Supported
Capabilities..19	0xA9800000	Reserved for capabilities of A9 class
Capabilities...20	0xAA000001	Get Dump Supported
Capabilities...21	0xAA800000	Reserved for capabilities of AA class
Capabilities…22	0xAB000001	SBE Halt Supported
Capabilities…23	0xAB800000	Reserved for capabilities of AB class

11.3 Quiesce SBE

This command is sent by SP to place the SBE into a quiescent state. Note that this command cannot be used to quiesce a previously issued command from SP. The SBE will stop all ongoing operations and enter a state where it cannot accept any further chip-ops from the SP FIFO interface as well as the Host PSU interface. This is a synchronous command where SBE will enqueue its response after completing the command. The Quiesce command should never fail with a bad response and hence does not provide for FFDC package in the response.

Message (request) from SP			
0x02			
reserved	reserved	**0xA8**	**0x06**

Response from SBE		
0xC0DE	**0xA8**	**0x06**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

11.4 Get SBE Capabilities 2

This command is sent by SP to SBE to retrieve the (extended) capabilities supported beyond what the Get SBE Capabilities (0xA802) command retrieves. The response is compatible with that for command 0xA802, however, with additional capability words starting with the capability word 24. Please refer to the table below for details on the extended capabilities supported by the SBE. This is a synchronous command where SBE will enqueue its response after completing the command.

Message (request) from SP
0x02

Response from SBE			
Byte 0	**Byte 1**	**Byte 2**	**Byte 3**
Major Version		Minor Version	
8 Hex Character SBE Firmware GIT Commit ID			
SBE Firmware Release Tag..0			
..			
SBE Firmware Release Tag..4			
Capabilities...0			
..			
Capabilities…25			
0xC0DE		0xA8	0x04
Primary Status		Secondary Status	
[FFDC Package..0]			
..			
[FFDC Package..N]			
Distance to status header word with magic byte 0xC0DE			
Primary Status, Secondary Status: See section 3.1			

Capabilities …0-23	Same as response to command 0xA802	See section 11.2 (Same as command 0xA802)
Capabilities…24	0xAC000001	PMIC Health Check Supported
Capabilities…25	0xAC800000	Reserved for capabilities of AC class

11.5 Get LPC Status

This command is sent by SP to the SBE to determine the LPC status. This is a synchronous command where SBE will enqueue its response after completing the command. SBE does lpc read operations on the LPC OPB Primay Status register, to determine if the LPC hardware logic is good or hung.

Message (request) from SP			
0x02			
reserved	reserved	0xA8	0x08

Response from SBE		
LPC Status		
0xC0DE	0xA8	0x08
Primary Status	Secondary Status	
[FFDC Package...0]		
..		
[FFDC Package…N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status: See section 3.1 ***LPC Status:*** ***0x0: LPC logic is good*** ***0x1: LPC logic is in error state***		

12 MPIPL Messages

12.1 Enter MPIPL

This command is sent by SP to SBE to enter a Memory Preserving IPL. This is a synchronous command where SBE will enqueue its response after completing the MPIPL quiesce sequence and is ready to accept chip-ops for dumping hardware facilities. Note that this command is supported only on native service processor (SP) based systems.

Message (request) from SP			
0x02			
reserved	reserved	**0xA9**	**0x01**

Response from SBE		
0xC0DE	**0xA9**	**0x01**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

12.2 Continue MPIPL

This command is sent by SP to the primary SBE to continue MPIPL after the SP completed dumping any hardware facilities. This is a synchronous command where SBE will en queue its response after loading and starting the Hostboot boot loader in MPIPL mode. Note that this chip-op is supported only on native service processor (SP) based systems. Any updates in the core and cache gard records will be picked up by the SBE, as in the normal IPL flow, from the Mailbox Scratch Register 1 (CFAM 0x2838 or SCOM 0x500038). On the secondary SBEs, Hostboot will use the same command to get the secondary SBEs out of MPIPL state to runtime state, after updating any core and cache gard records in Mailbox Scratch Register 1 as above..

Message (request) from SP			
0x02			
Reserved	Reserved	0xA9	0x02

Response from SBE		
0xC0DE	**0xA9**	**0x02**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

12.3 Stop Clocks

This command is sent by SP to SBE to stop clocks on the specified chiplets in the processor on which the SBE is placed. This is a synchronous command where SBE will enqueue its response after completing the operation. The SBE stops specific clock domains on the chiplets based on the target type and chiplet id passed in this command.

Message (request) from SP			
0x03			
Reserved	Reserved	0xA9	0x03
Target Type		Reserved	Chiplet Id

Bits 0-15: Target Type
See section 2.1.4

Bits 16-23: Reserved

Bits 24-31: Chiplet Id
See section 2.1.4

Response from SBE		
0xC0DE	**0xA9**	**0x03**
Primary Status	**Secondary Status**	
[FFDC Package..0]		
..		
[FFDC Package..N]		
Distance to status header word with magic byte 0xC0DE		

Primary Status, Secondary Status: See section 3.1

Note that only the target types listed in the table below are supported in this command.

Target Type	Chiplet Id	Clock Domains	Comments
`TARGET_TYPE_PROC = 0x0000`	`N.A.`	All except refresh clocks	Stop clocks on the whole processor except tp, sbe & vital
`TARGET_TYPE_EQ = 0x0002`	`0xFF`	eq clock regions.	Stop EQ clocks with region all
`TARGET_TYPE_CORE = 0x0003`	`0xFF`	core clock regions	Stop core clocks

12.4 Get TI Info

This command is sent by SP to SBE to get data from mainstore memory for a host initiated terminate immediate exception (TI). SBE will not retry on errors trying to collect data. This is a synchronous command where SBE will enqueue its response after completing the operation. The format, contents and interpretation of the data returned is contracted between the SP and Host subsystems and not in scope of this document

Message (request) from SP			
0x02			
Reserved	Reserved	0xA9	0x04

Response from SBE		
TI data word 0		
…		
TI data word M		
Length of TI data in words (M+1)		
0xC0DE	0xA9	0x04
Primary Status	Secondary Status	
[FFDC Package...0]		
..		
[FFDC Package…N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Statu : See section 3.1		

13 Dump Messages

13.1 Get Dump

This command is sent by SP to SBE to collect different types of dumps. This is a synchronous command where SBE will stream out its response as it collects the dump data from the system. For each dump type, SBE collects data from various hardware facilities like SCOM registers, SRAMs, fast arrays and so on, as specified in the hardware dump collection table (HDCT) schema.

Message (request) from SP			
0x03			
Reserved	Reserved	0xAA	0x01
Reserved	Collect FastArray	Clock State	Dump Type

Collect FastArray:

0x00: Collect Fast Arrays
0x01: Skip Fast Arrays

Clock State:

0x01: Clocks On
0x02: Clocks Off

Dump Type:

0x01: System Check Stop Dump
0x02: Memory Preserving IPL Dump
0x03: System Performance Dump
0x04: Core Check Stop Dump
0x05: Hostboot Dump

Response from SBE			
SBE section version	**Reserved**	**Reserved**	**Reserved**
HDCT version word 0			
HDCT version word 1			
HDCT Row 0 - Data word 0			
…			
HDCT Row 0 - Data word P			
…			
HDCT Row M - Data word 0			
…			
HDCT Row M - Data word Q			
0xC0DE		**0xAA**	**0x01**
[FFDC Package .. 0]			
[..]			
[FFDC Package .. N]			
Distance to status header word with magic byte 0xC0DE			
Primary Status, Secondary Status: See section 3.1			

14 Host FIFO Commands

14.1 SBE Halt

This command is sent by the host to the halt the SBE. This command is not accepted from the service processor and returns with a failure response. A failure to halt the SBE, will return with a failure response on the FIFO.

On success, the SBE Messaging and Status Register (section 2.1.7) Current State field (bits 8-11) will reflect the SBE state as Halted (0x9). There will be no chip-op response as the SBE halts itself.

Message (request) from SP			
0x02			
reserved	reserved	0xAB	0x01

Response from SBE (only on failure)		
0xC0DE	0xAB	0x01
Primary Status	Secondary Status	
[FFDC Package...0]		
..		
[FFDC Package…N]		
Distance to status header word with magic byte 0xC0DE		
Primary Status, Secondary Status : See section 3.1		

15 Telemetry Commands

15.1 PMIC Health Check

This command is sent by SP to SBE to collect PMIC Health and related telemetry data of DIMMs connected to the processor chip on which the SBE is placed. This is a synchronous command where SBE will enqueue its response after completing the operation. The format, contents and interpretation of the response data are governed by the hardware procedure team and beyond the scope of this document.

Message (request) from SP			
0x03			
Reserved	Reserved	0xAC	0x01
Target Type		Reserved	Instance ID

Bits 0-15: Target Type
See section 2.1.4. Only TARGET_TYPE_OCMB supported.

Bits 16-23: Reserved

Bits 24-31: Instance Id
See section 2.1.4. 0x00-0x0F

Response from SBE		
Data word 0		
…		
Data word N		
Length of Data in bytes ((N+1)*4)		
0xC0DE	0xAC	0x01
Primary Status	Secondary Status	
[FFDC Package...0]		
..		
[FFDC Package..M]		
Distance to status header word with magic byte 0xC0DE		

Primary Status, Secondary Status: See section 3.1

16 Open Items

16.1 Chip-Ops to be added

None.

16.2 Chip-Ops to be scrubbed

None

16.3 Miscellaneous updates

None

16.4 Review Rework

1. None

End

of

Document