VITA 62 Power Supply IPMI/I²C™ Communication

Vincent Gucciardo
Senior Firmware Engineer

Contents **Page**

Introduction

The VITA 62 power supply is a COTs power supply that is designed for OpenVPX™ systems. The module utilizes proprietary technology to enable high efficiency and power density for this highly rugged, conduction-cooled model.

Up to four power supplies can be paralleled to increase output power capability of VS1, VS2, VS3 outputs with proprietary wireless current sharing. Conventional current-share pins are eliminated. This document details the capabilities of the Intelligent Platform Management Interface (IPMI) available on VITA 62 power supplies (product information available at the Vicor website). The IPMI can be used for the purpose of monitoring the health of the system hardware by monitoring elements such as temperature, voltage, current, power and communications.

Note: A glossary of terminology and acronyms used in this application note is located at the rear of the document.

Message Interface

The Message Interface is defined as a 'request/response' interface. That is, a request message is used to initiate an action or set data, and a response message is returned to the Requestor. In this document, Request Messages are often referred to as 'commands' or 'requests', and Response Messages as 'responses.'

The following are the common components of messages specified in this document:

Table 1
Common message components

Message Component	Description
Network Function (NetFn)	A field that identifies the functional class of the message. The Network Function clusters IPMI commands into different sets. See *IPMI Specification* section 5.1 (page 40), Network Function Codes, for more information.
Request/Response Identifier	A field that unambiguously differentiates Request Messages from Response Messages. In the IPMB Protocol, this identifier is 'merged' with the Network Function code such that 'Even' network function codes identify Request Messages, and 'Odd' network function codes identify Response Messages.
Requestor's ID	Information that identifies the source of the Request. This information must be sufficient to allow the Response to be returned to the correct Requestor. For example, for the IPMB the Requestor's ID consists of the Child Address and LUN of the Requestor device. For a multiple-stream system interface the Requestor's ID is the 'stream id' for the stream through which the request was issued.
Responder's ID	A field that identifies the Responder to the Request. In Request Messages this field is used to address the Request to the desired Responder, in Response Messages this field is used to assist the Requestor in matching up a response with a given request.
Command	The messages specified in this document contain a one-byte command field. Commands are unique within a given Network Function. Command values can range from 00h through FDh. Code FEh is reserved for future extension of the specification, and code FFh is reserved for message interface level error reporting on potential future interfaces.
Data	The Data field carries the additional parameters for a request or a response, if any.

Power Supply Address

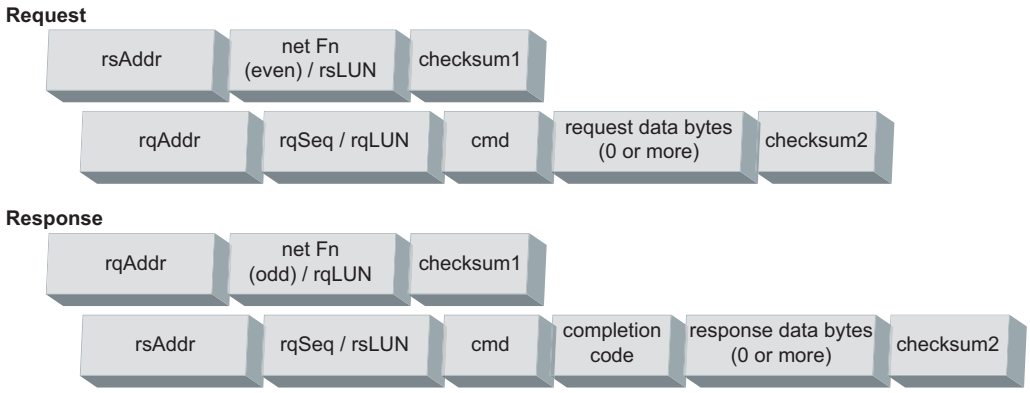
The hardware Child address of the power supply is the global address based upon jumpers GA0, GA1, etc. The IPMB Child address of the VITA power supply is hardware Child address left shifted 1 bit with the read/write bit set low.

Example

A power supply with address pins GA0 and GA1 pins set 3.3V will have a hardware Child address of 20h and its corresponding IPMB address is 40h.

IPMB Message Format

Figure 1
IPMI LAN message formats



Source: *IPMI Specification*, v2.0, Rev 1.1. October 1, 2013, page 136.

Where:

checksum1	An 8-bit additive checksum derived from all the bytes in the connection header preceding checksum1 including the first address byte. Checksums are derived by summing data bytes modulo 256 and taking the 2's complement of the sum. When all of the bytes in a message including the checksum are added together modulo 256, the result should be zero.
checksum2	An 8-bit additive checksum derived from all the bytes between checksum1 and the last byte, checksum2
cmd	Command Byte
completion code	Completion code returned in the response to indicated success/failure status of the request.
data	As required by the particular request or response for the command
LUN	The Logical Unit Number is represented by the lower 2 bits of the netFn byte. The LUN provides further sub-addressing within the FRU.
netFn	Network Function code
rq	Abbreviation for 'Requestor.'
rqLUN	Requestor's LUN.
rqAddr	Requestor's IPMB Address. The upper 7 bits hold the Child Address, the LS bit is always zero. This byte is always 20h when the BMC is the Requestor.
rqSeq	Sequence number, generated by the Requestor.
rs	for 'Responder.'
rsLUN	Responder's LUN
rsAddr	Responder's IPMB Child Address. The upper 7 bits hold the Child Address, the LS bit is always zero. This byte is always 20h when the BMC is the responder.
Seq	Sequence number. This field is used to verify that a response is for a particular instance of a request. Refer to IPMI v2.0 Specification for additional information on use and operation of the Seq field.

Intelligent Platform Management Interface Network Function (NetFn) Codes

The Vicor VITA 62 power supply supports Network Function (NetFn) commands from the following categories found in the VITA 46.11 and IPMI v2.0 Specifications.

Table 2
Supported NetFn command categories

NetFn Category	Description	Codes
Group Extension	Non-IPMI Group Requests and Responses	**2Ch** command / request
		2Dh response
Application	Application Requests and Responses	**06h** command / request
		07h response
Sensor/Event	Sensor and Event Requests and Responses	**04h** command / request
		05h response
Storage	Non-volatile Storage Requests and Responses	**0Ah** command / request
		0Bh response

Supported Command List and Index

Table 3
Supported NetFn commands

Command Name	Code	Description	Page
Group Extension			
Get FRU Address Info	40h	This command is used to retrieve Chassis Address Table information	6
Application			
Get Device ID	01h	This command is used to retrieve the Intelligent Device's Hardware Revision, Firmware/Software Revision and Sensor and Event Interface Command specification revision information	8
Sensor/Event			
Set Event Receiver	00h	This command sets the address for Event message transmits	10
Get Event Receiver	01h	This command retrieves the address for Event message transmits	11
Platform Event (Message)	02h	This command is a request for the BMC to process the event data that the command contains.	12
Get Device SDR Info	20h	This command returns general information about the collection of sensors in a Dynamic Sensor Device.	14
Get Device SDR	21h	This command allows SDR information for sensors for a Sensor Device to be returned.	16
Reserve Device SDR Repository	22h	This command is used to obtain a Reservation ID.	18
Get Sensor Hysteresis	25h	This command retrieves the present hysteresis values for the specified sensor. If the sensor hysteresis values are 'fixed', then the hysteresis values can be obtained from the SDR for the sensor.	19
Get Sensor Thresholds	27h	This command retrieves the thresholds for the given sensor.	20
Get Sensor Event Enable	29h	This command returns the enabled/disabled state for Event Message Generation from the selected sensor. The command also returns the enabled/disabled state for scanning on the sensor.	22
Get Sensor Event Status	2Bh	The Get Sensor Event Status command is provided to support systems where sensor polling is used in addition to, or instead of, Event Messages for event detection.	25
Get Sensor Reading	2Dh	This command returns the present reading for the specified sensor.	28
Get Sensor Type	2Fh	This command is used to retrieve the Sensor Type and Event/Reading Type for the specified sensor.	30
Storage			
Get FRU Inventory Area Info	10h	Returns the overall size of the FRU Inventory Area in this device, in bytes.	31
Read FRU data	11h	The command returns the specified data from the FRU Inventory Info area. This is effectively a 'low level' direct interface to a non-volatile storage area.	32

Group Extension Command 40h: Get FRU Address Info

This command is a reduced variant of the "Get Chassis Address Table Info" command that is implemented by IPMCs. See *VITA 46.11 Specification* section 10.1.3.3 "Get FRU Address Info" for requirements related to this command.

Table 4
Get FRU address request and response

Byte	Data Field
Request Data	
1	VSO Identifier
2	FRU Device ID
3*	Address Key Type
4*	Address Key
5*	Site Type
Response Data	
1	Completion Code
2	VSO Identifier
3	Hardware Address
4	IPMB Address
5	Reserved(0xFF)
6	FRU Device ID
7	Site Number
8	Site Type
9	Reserved(0xFF)
10	Address on IPMI Channel 7

Example Get FRU Info request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 5
Get FRU Info REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Group Extension Request 2C (even)						rsLUN is 0		B0h
3	Checksum for the connection Header								10h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 04h						rqLUN is 2		12h
6	Command 40h: Get FRU Address Info								40h
7	Identifier (03h = VITA Standards Organization)								03h
8	Checksum for preceding bytes between the previous checksum								21h

Table 6
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Group Extension Response 2D (odd)						rqLUN is 2		B6h
3	Checksum for the connection Header								CAh
4	rsSA = 40h (Responder's Child address)								40h
5	rqSeq = 04h						rsLUN is 0		10h
6	Command 40h- Get FRU Address Info								40h
7	(Completion Code 00 = 'OK')								00h
8	Defining Body – VITA Standards Organization								03h
9	Hardware Address								20h
10	IPMB Address								40h
11	Reserved								FFh
12	FRU Device ID								00h
13	Site Number								02h
14	Site Type (Front Loading VPX Plug-In Module)								00h
15	Reserved								FFh
16	Address on IPMI Channel 7 (FF if not used)								FFh
17	Checksum for bytes following Connection Header checksum								1Eh

Application Command 01h: Get Device ID

This command is used to retrieve the Intelligent Device's Hardware Revision, Firmware/Software Revision and Sensor and Event Interface Command specification revision information. The command also returns information regarding the additional 'logical device' functionality (beyond 'Application' and 'IPM' device functionality) that is provided within the intelligent device, if any. These are the Device ID and the Product ID fields. A controller that just implements standard IPMI commands can set the Device ID and the Product ID fields to 'unspecified.' Additional specifications and descriptions for the Device ID response fields can be found in *IPMI Specification* section 20 (p. 243).

Table 8
Get Device ID request and response

Byte	Data Field
Request Data	
No request data	
Response Data	
1	Completion Code
2	Device ID (0x00 = unspecified)
3	Device Revision **d7** SDRs available **0** = device SDRs not provided **1** = device SDRs provided **d6:d4** reserved; return as 000b **d3:d0** Device Revision
4	Firmware Revision 1 – major revision binary encoded **d7** Device available **0** = normal operation **1** = update in progress **d6:d0** Major revision BCD encoded
5	Firmware Revision 2 – Minor revision BCD encoded
6	IPMI Version – BCD encoded **d7:d4** Least significant digit of the revision **d3:d0** Most significant digit of the revision
7	Additonal Device Support **d7** Chassis device **d6** Bridge (accepts Bridge NetFn cmds) **d5** IPMB Event Generator **d4** IPMB Event Receiver **d3** FRU Inventory Device **d2** SEL Device **d1** SDR Repository Device **d0** Sensor Device
8:10	Manufacturer ID – 20-bit IANA ID (LS byte first) **0x000000** = unspecified **0xFFFF** = reserved
11:12	Product ID (LS byte first) **0x000000** = unspecified **0xFFFF** = reserved
13:16	Auxiliary firmware revision information [a]

[a] Optional 4-digit hexadecimal number specific to vendor. Note that these bytes are not transmitted by the VITA 62 power supply.

Example Get Device ID request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 9
Get Device ID REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Application 06 (even)						rsLUN is 0		18h
3	Checksum for the connection Header								A8h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 08h						rqLUN is 2		22h
6	Command 40h: Get FRU Address Info								01h
7	Command 01h: Get Device ID								5Dh

Table 10
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (VPX IPMB Child address)								80h
2	NetFn is Application Response 07 (odd)						rqLUN is 2		1Eh
3	Checksum for the connection Header								62h
4	rsSA = 40h (Responder's VPX IPMB Child address)								40h
5	rqSeq = 08h						rsLUN is 0		20h
6	Command 01h: Get Device ID								01h
7	(Completion Code 00 = 'OK')								00h
8	Device ID								01h
9	Device Revision								81h
10	Firmware Revision 1 – major								03h
11	Firmware Revision 2 – minor								07h
12	IPMI Version 2.0								02h
13	Additional Device Support								2Dh
14	Manufacturers ID LSB(20 bits)								B5h
15	Manufacturers ID								6Ah
16	N/A				Manufacturer's ID most-significant 4 bits				00h
17	Product ID LSB								0Ah
18	Product ID MSB								11h
19	Checksum for bytes following Connection Header checksum								B4H

Sensor/Event Command 00h: Set Event Receiver

This global command tells a controller where to send Event Messages. The Child address and LUN of the Event Receiver must be provided. A value of 0xFF for the Event Receiver Child Address disables Event Message generation entirely. This command is only applicable to management controllers that act as IPMB Event Generators.

Table 12
Set Event Receiver request and response

Byte	Data Field
Request Data	
1	Event Receiver
2	LUN
Response Data	
1	Completion Code

Example Set Event Receiver request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 13
Set Event Receiver REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor/Event 04 (even)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 0Ch						rqLUN is 2		32h
6	Command 00h- Set Event Receiver								00h
7	Event Receiver IPMB Child Address = 80h								80h
8	Event Receiver LUN = 02h								02h
9	Checksum for preceding bytes between the previous checksum								CCh

Table 14
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 0Ch						rsLUN is 0		30h
6	Command 00h - Set Event Receiver								00h
7	(Completion Code 00 = 'OK')								00h
8	Checksum for preceding bytes between the previous checksum								90h

Sensor/Event Command 01h: Get Event Receiver

This global command is used to retrieve the present setting for the Event Receiver Child Address and LUN. This command is only applicable to management controllers that act as IPMB Event Generators.

Table 15
Get Event Receiver request and response

Byte	Data Field
Request Data	
No request data	
Response Data	
1	Completion Code
2	Event Receiver Child Address 0FFh indicates Event Message Generation has been disabled; otherwise: **d7:d1** IPMB (I2C) 7-bit Child Address **d0** always 0b
3	Event Receiver LUN **d7:d2** reserved **d1:d0** Event Receiver LUN

Example Get Event Receiver request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 16
Get Event ReceiverREQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor/Event 04 (even)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 0Ch						rqLUN is 2		32h
6	Command 01h - Get Event Receiver								01h
7	Checksum for preceding bytes between the previous checksum								4Dh

Table 17
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 0Ch						rsLUN is 0		30h
6	Command 01h - Get Event Receiver								01h
7	(Completion Code 00 = 'OK')								00h
8	Event Receiver Address								80h
9	Event Receiver LUN								02h
10	Checksum for preceding bytes between the previous checksum								0Dh

Sensor/Event Command 02h: Platform Event Message Command

This command may be considered a request for the BMC to process the event data that the command contains. Typically, the data will be logged to the System Event Log (SEL). Depending on the implementation, the data may also go to the Event Message Buffer and processed by Platform Event Filtering (PEF).

The Generator ID field is a required element of an Event Request Message. For IPMB messages, this field is equated to the Requestor's Child Address and LUN fields. Thus, the Generator ID information is not carried in the data field of an IPMB request message.

For 'system side' interfaces, it is not as useful or appropriate to 'overlay' the Generator ID field with the message source address information, so it is specified as being carried in the data field of the request.

Table 18
Platform Event Message request and response

Byte	Data Field
Request Data	
1	EvMRev
2	Sensor Type
3	Sensor Number
4	Event Direction / Type **d7** Event direction **0b** = assertion event **1b** = de-assertion event **d6:d0** Event Type Code; see *IPMI Specification* section 42.1 (p. 502) Event/Reading Type Codes
5	Event Data 1
6	Event Data 2 [b]
7	Event Data 3 [b]
Response Data	
1	Completion Code

[b] Optional per VITA 46.11 and unused in VITA 62 power supplies.

Event Request Message Fields

Table 19
Event Request Messages

Message	Description
Generator ID	This field identifies the device that has generated the Event Message. This is the 7-bit Requestor's Child Address (RqSA) and 2-bit Requestor's LUN (RqLUN) if the message was received from the IPMB.
EvMRev	One byte. Event Message Revision. This field is used to identify different revisions of the Event Message format. The revision number shall be 04h for Event Messages that comply with the format given in this specification. IPMI v1.0 messages use 03h. It is recommended that software be able to interpret both versions.
Sensor Type	One byte. Indicates the event class or type of sensor that generated the Event Message. Codes are specified in *IPMI Specification,* Table 42-3 (p. 505), Sensor Type Codes.
Sensor Number	One byte. A unique number (within a given sensor device) representing the 'sensor' within the management controller that generated the Event Message. Sensor numbers are used for both identification and access of sensor information, such as getting and setting sensor thresholds.
Event Direction	1-bit. Indicates the event transition direction: **0** = Assertion event **1** = De-assertion event
Event Type	This 7-bit field indicates the type of threshold crossing or state transition (trigger) that produced the event. This is encoded using the Event/Reading Type Code. See *IPMI Specification* section 42 (p. 502) Sensor and Event Code Tables.
Event Data	One to three Bytes. The remainder of the Event Message data according to the class of the Event Type for the sensor (threshold, discrete, or OEM). The contents and format of this field are found in *IPMI Specification,* Table 29-6 (p. 405) Event Request Message Event Data Field Contents.

Event Data Field Formats

Table 20
Field formats for event data

Event Data	Field	Contents
Threshold Sensor Events		
Event Data 1	d7:d6	**00b** unspecified byte 2
		01b trigger reading in byte 2
		10 OEM code in byte 2
		11 sensor-specific event extension code in byte 2
	d5:d4	**00b** unspecified byte 3
		01b trigger reading in byte 3
		10 OEM code in byte 3
		11 sensor-specific event extension code in byte 3
	d3:d0	Offset from Event/Reading Code for threshold event
Event Data 2	-	Reading that triggered event; 0xFF or not present if unspecified
Event Data 3	-	Threshold value that triggered event; 0xFF or not present if unspecified
Discrete Sensor Events		
Event Data 1	d7:d6	**00b** unspecified byte 2
		01b previous state and/or severity in byte 2
		10 OEM code in byte 2
		11 sensor-specific event extension code in byte 2
	d5:d4	**00b** unspecified byte 3
		01b previous state and/or severity in byte 3
		10 OEM code in byte 3
		11 sensor-specific event extension code in byte 3
	d3:d0	Offset from Event/Reading Code for discrete event state
Event Data 2 (Optional OEM code or severity/previous state fields)	d7:d4	Optional offset from Severity / Event / Reading Code (0x0F if unspecified)
	d3:d0	Optional offset from Event / Reading Type Code for previous discrete event state (0x0F if unspecified)
Event Data 3	-	Optional OEM code; 0xFF or not present if unspecified
OEM Sensor Events		
Event Data 1	d7:d6	**00b** unspecified byte 2
		01b previous state and/or severity in byte 2
		10 OEM code in byte 2
		11 reserved
	d5:d4	**00b** unspecified byte 3
		01b previous state and/or severity in byte 3
		10 OEM code in byte 3
		11 reserved
	d3:d0	Offset from Event/Reading Code for discrete event
Event Data 2 (Optional OEM code or severity/previous state fields)	d7:d4	Optional OEM code bits or offset from Severity / Event / Reading Code (0x0F if unspecified)
	d3:d0	Optional OEM code bits or offset from Event / Reading Type Code for previous discrete event state (0x0F if unspecified)
Event Data 3	-	Optional OEM code; 0xFF or not present if unspecified

Note: Event Data 2 and Event Data 3 are not present for all sensors.

Sensor/Event Command 20h: Get Device SDR Info

This command returns general information about the collection of sensors in a Dynamic Sensor Device.

Note: If the command is issued with no parameter for the request, the Device Sensor information is LUN based. That is, it is returned individually for each LUN. For example, a device could implement eight sensors under one LUN, and ten under another. The SDR Info does not return the aggregate of the sensor information. Rather, separate 'Get Device SDR Info' commands need to be issued to each LUN. The 'Device LUNs' field is provided in the response to support this.

Table 21
Get Device SDR Info request and response

Byte	Data Field
Request Data	
1	Operation (optional) **d7:d1** reserved **d0** **0b** – Get Sensor Count returns the number of sensors implemented on the LUN this command was addressed to **1b** – Get SDR count returns the total number of sensors in the device.
Response Data	
1	Completion Code
2	For operation: **Get Sensor Count** (or if byte 1 not present in request): the number of sensors in device for the LUN this command was addressed to. **Get SDR Count:** the number of SDRs in the device.
3	Flags **d7** Dynamic population **0b** Static sensor population **1b** Dynamic sensor population that may vary during "run time" **d6:d4** reserved **d3** LUN 3 has sensors **d2** LUN 2 has sensors **d1** LUN 1 has sensors **d0** LUN 0 has sensors
4:7	Sensor population change indicator (4-byte timestamp)

Note: timestamp will return zeroes since sensor population is static.

Example Get Device SDR Info request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 22
Get Device SDR Info REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor/Event 04 (even)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 10h						rqLUN is 2		42h
6	Command 20h- Get Device SDR Info								20h
7	Checksum for preceding bytes between the previous checksum								0Eh

Table 23
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 0Ch						rsLUN is 0		40h
6	Command 20h – Get Device SDR Info								20h
7	(Completion Code 00 = 'OK')								00h
8	Number of sensors in device								16h
9	Dynamic sensor population, LUN 0 has sensors								81h
10	Sensor population change indicator (timestamp) LSB								00h
11	Sensor population change indicator (timestamp)								00h
12	Sensor population change indicator (timestamp)								00h
13	Sensor population change indicator (timestamp) MSB								00h
14	Checksum for preceding bytes between the previous checksum								DEh

Sensor/Event Command 21h: Get Device SDR

The 'Get Device SDR' command allows SDR information for sensors for a Sensor Device (typically implemented in a satellite management controller) to be returned. The Get Device SDR Command can return any type of SDR, not just Types 01h and 02h. This is an optional command for Static Sensor Devices and mandatory for Dynamic Sensor Devices. The format and action of this command is similar to that for the 'Get SDR' command for SDR Repository Devices.

A Sensor Device shall always utilize the same sensor number for a particular sensor. This is mandatory to keep System Event Log information consistent.

Sensor Devices that support the 'Get Device SDR' command return SDR Records that match the SDR Repository formats. See section 0, This command returns general information about the collection of sensors in a Dynamic Sensor Device.

Table 24
Get Device SDR request and response

Byte	Data Field
Request Data	
1	Reservation ID. LS Byte; only required for partial reads with a non-zero 'Offset into record' field; use 0x0000 for reservation ID otherwise
2	Reservation ID. MS Byte
3	Record ID LS Byte (0x0000 returns first record)
4	Record ID MS Byte
5	Offset into record
6	Number of bytes to read (0xFF for entire record)
Response Data	
1	Completion Code
2	Record ID for next record LS Byte
3	Record ID for next record MS Byte
4-N+3	Requested bytes from record

Example Get Device SDR request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 25
Get Device SDR REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor/Event 04 (even)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 10h						rqLUN is 2		42h
6	Command 21h - Get Device SDR								21h
7	Reservation ID LSB								02h
8	Reservation ID MSB								00h
9	Record ID LSB (0000h returns first record)								00h
10	Record ID MSB								00h
11	Offset into record								00h
12	Length of data to read (bytes)								05h
13	Checksum for preceding bytes between the previous checksum								0Eh

Table 26
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 10h						rsLUN is 0		40h
6	Command 210h - Get Device SDR								21h
7	(Completion Code 00 = 'OK')								00h
8	Reservation ID LSB								02h
9	Reservation ID MSB								00h
10	Record ID LSB								01h
11	Record ID MSB								00h
12	SDR Version 1.5								51h
13	Record Type – Management Controller Device Locator Record								12h
14	Record Length								16h
15	Checksum for preceding bytes between the previous checksum								D3h

Sensor/Event Command 22h: Reserve Device SDR Repository

This command is used to obtain a Reservation ID. The Reservation ID is part of a mechanism that is used to notify the Requestor that a record may have changed during the process of a multi-part read. See *IPMI Specification* section 33.11 (p. 440), Reserve SDR Repository, for more information on the function and use of Reservation IDs.

Table 27
Reserve Device SDR Repository request and response

Byte	Data Field
Request Data	
No request data	
Response Data	
1	Completion Code
2	Reservation ID LS Byte
3	Reservation ID MS Byte

Example Reserve Device SDR Repository request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 28
Reserve Device SDR Repository REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor/Event 04 (even)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 08h						rqLUN is 2		22h
6	Command 22h - Reserve Device SDR Repository								22h
7	Checksum for preceding bytes between the previous checksum								3Ch

Table 29
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 08h						rsLUN is 0		20h
6	Command 22h – Reserve Device SDR Repository								22h
7	(Completion Code 00 = 'OK')								00h
8	Reservation ID LSB								02h
9	Reservation ID MSB								00h
10	Checksum for preceding bytes between the previous checksum								7Ch

Sensor/Event Command 25h: Get Sensor Hysteresis

This command retrieves the present hysteresis values for the specified sensor. If the sensor hysteresis values are 'fixed', then the hysteresis values can be obtained from the SDR for the sensor.

Table 33
Get Sensor Hysteresis request and response

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
2	Reserved for future hysteresis mask (write as 0xFF)
Response Data	
1	Completion code
2	Positive-going threshold hysteresis value
3	Negative-going threshold hysteresis value

Note: Returns a value of 0x00 if hysteresis is N/A.

Example Get Sensor Hysteresis request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 34
Get Sensor Hysteresis REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor/Event 04 (even)						rsLUN is 0		B0h
3	Checksum for the connection Header								10h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 04h						rqLUN is 2		12h
6	Command - Get Sensor Hysteresis								25h
7	Sensor number (0xFF = reserved)								08h
8	Reserved for future hysteresis mask (write as 0xFF)								FFh
9	Checksum for preceding bytes between the previous checksum								42h

Table 35
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 08h						rsLUN is 0		20h
6	Command – Get Sensor Hysteresis								25h
7	(Completion Code 00 = 'OK')								00h
8	Positive-going Threshold Hysteresis Value								03h
9	Negative-going Threshold Hysteresis Value								03h
10	Checksum for preceding bytes between the previous checksum								75h

Sensor/Event Command 27h: Get Sensor Thresholds

This command retrieves the threshold for the given sensor.

Table 36
Get Sensor Thresholds request and response

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
Response Data	
1	Completion code
2	Readable Thresholds **d7:d6** reserved(return as 00b) **d5** upper non-recoverable threshold **d4** upper critical threshold **d3** upper non-critical threshold **d2** lower non-recoverable threshold **d1** lower critical threshold **d0** lower non-critical threshold
3	Lower non-critical threshold
4	Lower critical threshold
5	Lower non-recoverable threshold
6	Upper non-critical threshold
7	Upper critical threshold
8	Upper non-recoverable threshold

Example Get Sensor Thresholds for VS1 request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 37
Get Sensor Thresholds for VS1 REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Group Extension Request 2C (even)						rsLUN is 0		B0h
3	Checksum for the connection Header								10h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 04h						rqLUN is 2		12h
6	Command - Get Sensor Thresholds								27h
7	Sensor Number (0xFF = reserved)								08h
8	Checksum for preceding bytes between the previous checksum								3Fh

Table 38
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 08h						rsLUN is 0		20h
6	Command – Get Sensor Thresholds								27h
7	(Completion Code 00 = 'OK')								00h
8	Readable Thresholds								36h
9	Lower non-critical threshold [c]								00h
10	Lower critical threshold								7Eh
11	Lower non-recoverable threshold								72h
12	Upper non-critical threshold [c]								00h
13	Upper critical threshold								AEh
14	Upper non-recoverable threshold								BBh
15	Checksum for preceding bytes between the previous checksum								CAh

[c] Non-critical thresholds are not supported.

Sensor/Event Command 29h: Get Sensor Event Enable

This command returns the enabled/disabled state for Event Message Generation from the selected sensor. The command also returns the enabled/disabled state for scanning on the sensor.

A typical sensor will come up with Event Messages (EvM) enabled for all thresholds. Sensors are not required to have individual or per-event Event Message enables. The type of enable/disable support that a sensor provides can be obtained from the Sensor Data Record for the sensor.

Table 39
Get Sensor Event Enable request and response (sensors with threshold-based events)

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
Response Data (all cases)	
1	Completion code
2	Sensor status **d7** All Event Messages Enable **d6** Sensor Scanning Enable **d5:d0** reserved - ignore on read
Response Data (for sensors with threshold-based events)	
3	**d7** assertion event for upper non-critical going high **d6** assertion event for upper non-critical going low **d5** assertion event for lower non-recoverable going high **d4** assertion event for lower non-recoverable going low **d3** assertion event for lower critical going high **d2** assertion event for lower critical going low **d1** assertion event for lower non-critical going high **d0** assertion event for lower non-critical going low
4	**d7:d4** reserved - write as 0000b **d3** assertion event for upper non-recoverable going high **d2** assertion event for upper non-recoverable going low **d1** assertion event for upper critical going high **d0** assertion event for upper critical going low
5	**d7** de-assertion event for upper non-critical going high **d6** de-assertion event for upper non-critical going low **d5** de-assertion event for lower non-recoverable going high **d4** de-assertion event for lower non-recoverable going low **d3** de-assertion event for lower critical going high **d2** de-assertion event for lower critical going low **d1** de-assertion event for lower non-critical going high **d0** de-assertion event for lower non-critical going low
6	**d7:d4** reserved - write as 0000b **d3** de-assertion event for upper non-recoverable going high **d2** de-assertion event for upper non-recoverable going low **d1** de-assertion event for upper critical going high **d0** de-assertion event for upper critical going low

Table 40
Get Sensor Event Enable request and response (sensors with descrete events)

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
Response Data (all cases)	
1	Completion code
2	Sensor status **d7** All Event Messages Enable **d6** Sensor Scanning Enable **d5:d0** reserved - ignore on read
Response Data (for sensors with discrete events)	
3	**d7** assertion event message for state bit 7 **d6** assertion event message for state bit 6 **d5** assertion event message for state bit 5 **d4** assertion event message for state bit 4 **d3** assertion event message for state bit 3 **d2** assertion event message for state bit 2 **d1** assertion event message for state bit 1 **d0** assertion event message for state bit 0
4	**d7** reserved **d6** assertion event message for state bit 14 **d5** assertion event message for state bit 13 **d4** assertion event message for state bit 12 **d3** assertion event message for state bit 11 **d2** assertion event message for state bit 10 **d1** assertion event message for state bit 9 **d0** assertion event message for state bit 8
5	**d7** de-assertion event message for state bit 7 **d6** de-assertion event message for state bit 6 **d5** de-assertion event message for state bit 5 **d4** de-assertion event message for state bit 4 **d3** de-assertion event message for state bit 3 **d2** de-assertion event message for state bit 2 **d1** de-assertion event message for state bit 1 **d0** de-assertion event message for state bit 0
6	**d7** reserved **d6** de-assertion event message for state bit 14 **d5** de-assertion event message for state bit 13 **d4** de-assertion event message for state bit 12 **d3** de-assertion event message for state bit 11 **d2** de-assertion event message for state bit 10 **d1** de-assertion event message for state bit 9 **d0** de-assertion event message for state bit 8

Example Get Sensor Event Enable for Sensor 8: VS1 request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 41
Get Sensor Event Enable for Sensor 8: VS1 REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Group Extension Request 2C (even)						rsLUN is 0		B0h
3	Checksum for the connection Header								10h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 04h						rqLUN is 2		12h
6	Command - Get Event Enable								29h
7	Sensor Number (0xFF = reserved)								08h
8	Checksum for preceding bytes between the previous checksum								3Dh

Table 42
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 08h						rsLUN is 0		20h
6	Command – Get Sensor Event Enable								29h
7	(Completion Code 00 = 'OK')								00h
8	Sensor Status								40h
9	Assertion Event Enable status								3Ch
10	Assertion Event Enable status								0Fh
11	De-assertion Event Enable status								3Ch
12	De-assertion Event enable status								0Fh
13	Checksum for preceding bytes between the previous checksum								A1h

Sensor/Event Command 2Bh: Get Sensor Event Status

The Get Sensor Event Status command is provided to support systems where sensor polling is used in addition to, or instead of, Event Messages for event detection.

A device that implements a sensor ***must generate only a single Event Message*** for a given sensor event. However, retries of the same message will be allowed.

All of the analog threshold sensors are 'auto- re-arm' sensors, clearing their internal flag when the event condition goes away. The Get Sensor Event Status commands may be considered as returning the state of these internal flags.

The event status gets updated when the controller detects a state change or transition between the present state and the previous state (conditioned by hysteresis as appropriate). The exception to this is when a sensor is re-armed by a Set Event Receiver command. In this case, the event status gets updated after the controller gets its first reading for the sensor.

The format of the Get Sensor Event Status response is dependent on whether the sensor was threshold based or discrete.

Threshold-based
Present threshold comparison event status.

Discrete
Present event status represented by a bit mask indicating the event conditions that are presently active on the sensor.

Note: this is redundant to the status returned with the 'Get Sensor Reading' command if there is no hysteresis associated with the sensor.

Table 45
Get Sensor Event Status request and response (sensors with threshold-based events)

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
Response Data (all cases)	
1	Completion code
2	Sensor status **d7** All Event Messages Enable **d6** Sensor Scanning Enable **d5:d0** reserved - ignore on read
Response Data (for sensors with threshold-based events)	
3	**d7** assertion event for upper non-critical going high **d6** assertion event for upper non-critical going low **d5** assertion event for lower non-recoverable going high **d4** assertion event for lower non-recoverable going low **d3** assertion event for lower critical going high **d2** assertion event for lower critical going low **d1** assertion event for lower non-critical going high **d0** assertion event for lower non-critical going low
4	**d7:d4** reserved - write as 0000b **d3** assertion event for upper non-recoverable going high **d2** assertion event for upper non-recoverable going low **d1** assertion event for upper critical going high **d0** assertion event for upper critical going low
5	**d7** de-assertion event for upper non-critical going high **d6** de-assertion event for upper non-critical going low **d5** de-assertion event for lower non-recoverable going high **d4** de-assertion event for lower non-recoverable going low **d3** de-assertion event for lower critical going high **d2** de-assertion event for lower critical going low **d1** de-assertion event for lower non-critical going high **d0** de-assertion event for lower non-critical going low
6	**d7:d4** reserved - write as 0000b **d3** de-assertion event for upper non-recoverable going high **d2** de-assertion event for upper non-recoverable going low **d1** de-assertion event for upper critical going high **d0** de-assertion event for upper critical going low

Table 46
Get Sensor Event Status request and response (sensors with descrete events)

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
Response Data (all cases)	
1	Completion code
2	Sensor status **d7** All Event Messages Enable **d6** Sensor Scanning Enable **d5:d0** reserved - ignore on read
Response Data (for sensors with discrete events)	
3	**d7** assertion event message for state bit 7 **d6** assertion event message for state bit 6 **d5** assertion event message for state bit 5 **d4** assertion event message for state bit 4 **d3** assertion event message for state bit 3 **d2** assertion event message for state bit 2 **d1** assertion event message for state bit 1 **d0** assertion event message for state bit 0
4	**d7** reserved **d6** assertion event message for state bit 14 **d5** assertion event message for state bit 13 **d4** assertion event message for state bit 12 **d3** assertion event message for state bit 11 **d2** assertion event message for state bit 10 **d1** assertion event message for state bit 9 **d0** assertion event message for state bit 8
5	**d7** de-assertion event message for state bit 7 **d6** de-assertion event message for state bit 6 **d5** de-assertion event message for state bit 5 **d4** de-assertion event message for state bit 4 **d3** de-assertion event message for state bit 3 **d2** de-assertion event message for state bit 2 **d1** de-assertion event message for state bit 1 **d0** de-assertion event message for state bit 0
6	**d7** reserved **d6** de-assertion event message for state bit 14 **d5** de-assertion event message for state bit 13 **d4** de-assertion event message for state bit 12 **d3** de-assertion event message for state bit 11 **d2** de-assertion event message for state bit 10 **d1** de-assertion event message for state bit 9 **d0** de-assertion event message for state bit 8

Example Get Sensor Event Status for VS1 current request sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 47
Sensor Event Status for VS1 current REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor / Event 04 (even)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 04h						rqLUN is 2		12h
6	Command - Get Sensor Event Status								2Ah
7	Sensor Number (0xFF = reserved)								0Eh
8	Checksum for preceding bytes between the previous checksum								36h

Table 48
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 08h						rsLUN is 0		20h
6	Command – Get Sensor Event Status								2Ah
7	(Completion Code 00 = 'OK')								00h
8	Sensor status								40h
9	No Assertion Events								00h
10	Assertion Event for Upper Critical going high								02h
11	No De-assertion Events								00h
12	No De-assertion Events								00h
13	Checksum for preceding bytes between the previous checksum								34h

Sensor/Event Command 2Dh: Get Sensor Reading

This command returns the present reading for a sensor. The sensor device may return a stored version of a periodically updated reading, or the sensor device may scan to obtain the reading after receiving the request.

The meaning of the state bits returned by Discrete sensors is based on the Event/Reading Type code from the SDR for the sensor. This can also be obtained directly from the controller if the optional Get Sensor Type command is supported for the sensor. Refer to *IPMI Specification* section 41.2 (p. 498), Event/Reading Type Code, for information on interpreting Event/Reading Type codes when used for present readings.

Table 48
Get Sensor Reading request and response (sensors with threshold-based events)

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
Response Data (all cases)	
1	Completion code
2	Sensor reading (ignore on read if sensor does not return an analog value)
3	Sensor status **d7** All event messages disable for this sensor **d6** Sensor scanning enable **d5** Reading unavailable **d4:d0** reserved ignore on read
Response Data (for sensors with threshold-based events)	
4	Present Threshold Comparison Status **d7:d6** reserved. Returned as 11b; ignore on read **d5** upper non-recoverable threshold **d4** upper critical threshold **d3** upper non-critical threshold [c] **d2** lower non-recoverable threshold **d1** lower critical threshold **d0** lower non-critical threshold [c] *Where* ***dx*** = **0**: sensor has not reached threshold **1**: sensor has exceeded threshold

[c] Non-critical thresholds are not supported.

Table 49
Get Sensor Reading request and response (sensors with discrete events)

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
Response Data (all cases)	
1	Completion code
2	Sensor reading (ignore on read if sensor does not return an analog value)
3	Sensor status **d7** All event messages disable for this sensor **d6** Sensor scanning enable **d5** Reading unavailable **d4:d0** reserved ignore on read
Response Data (for sensors with discrete events)	
4	**d7** state 7 **d6** state 6 **d5** state 5 **d4** state 4 **d3** state 3 **d2** state 2 **d1** state 1 **d0** state 0 *Where **dx** =* **0**: state de-asserted **1**: state asserted
Response Data (optional: for discrete reading sensors only)	
5 [d]	**d7** reserved; returned as 1b; ignore on read **d6** state bit 14 **d5** state bit 13 **d4** state bit 12 **d3** state bit 11 **d2** state bit 10 **d1** state bit 9 **d0** de-assertion event for lower non-critical going low

[d] Discrete sensor will return a value of 0x00 if this byte is not used.

Sensor/Event Command 2Fh: Get Sensor Type

This command is used to retrieve the Sensor Type and Event/Reading Type for the specified sensor. This command is mandatory for sensors that respond to the Set Sensor Type command.

Table 50
Get Sensor Type request and response

Byte	Data Field
Request Data	
1	Sensor number (0xFF = reserved)
Response Data (all cases)	
1	Completion code
2	Sensor type (ignore on read if sensor does not return an analog value)
3	Sensor status **d7** reserved **d6:d0** Event/Reading type code (see Table 52, Generic Event/Reading Type Codes)

Table 51
Event/Reading Type Code Ranges

Category	Range	Sensor Class	Description
[unspecified]	0x00	n/a	Event/Reading Type unspecified
Threshold	0x01	Threshold	Threshold-based. Indicates a sensor that utilizes values that represent discrete threshold states in sensor access and/or events
Generic	0x02 – 0x0C	Discrete	Generic discrete
Sensor-Specific	0x6F	Discrete	Sensor-specific discrete; indicates that the discrete state information is specific to the sensor type
OEM	0x70 – 0x7F	OEM	OEM discrete; indicates that the discrete state information is specific to the OEM identified by the Manufacturer ID for the IPM device that is providing access to the sensor.

Table 52
Generic Event/Reading Type Codes

Type Code	Class	Offset	Description
0x00	[unspecified]	n/a	Event/Reading Type unspecified
Threshold-Based States			
0x01	Threshold	0x00	Lower Non-critical - going low
		0x01	Lower Non-critical - going high
		0x02	Lower Critical - going low
		0x03	Lower Critical - going high
		0x04	Lower Non-recoverable - going low
		0x05	Lower Non-recoverable - going high
		0x06	Upper Non-critical - going low
		0x07	Upper Non-critical - going high
		0x08	Upper Critical - going low
		0x09	Upper Critical - going high
		0x0A	Upper Non-recoverable - going low
		0x0B	Upper Non-recoverable - going high
Digital/Discrete Event States			
0x03	"digital" Discrete	0x00	State De-asserted
		0x01	State Asserted
0x04	"digital" Discrete	0x00	Predictive Failure de-asserted.
		0x01	Predictive Failure asserted.
0x05	"digital" Discrete	0X00	Limit Not Exceeded.
		0X01	Limit Exceeded.

Storage Command 10h: Get FRU Inventory Area Info

Returns overall the size of the FRU Inventory Area in this device, in bytes.

Table 54 *Get FRU Inventory Area Info request and response*

Byte	Data Field
Request Data	
1	FRU Device ID (0xFF = reserved)
Response Data	
1	Completion code
2	FRU Inventory area size in bytes, LS Byte
3	FRU Inventory area size in bytes, MS Byte

Example Get FRU Inventory Area Info sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 55 *Get FRU Inventory Area Info REQUEST*

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Storage 0A (even)						rsLUN is 0		28h
3	Checksum for the connection Header								98h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 0Ch						rqLUN is 2		32h
6	Command 10h - Get FRU Inventory Area Info								10h
7	Command 10h - Get FRU Inventory Area Info								00h
8	Checksum for preceding bytes between the previous checksum								3Eh

Table 56 *RESPONSE message transmitted*

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 0B (odd)						rqLUN is 2		2Eh
3	Checksum for the connection Header								52h
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 0Ch						rsLUN is 0		30h
6	Command 10h - Get FRU Inventory Area Info								10h
7	(Completion Code 00 = 'OK')								00h
8	FRU Inventory Area Size – LSB								00h
9	FRU Inventory Area Size – MSB								02h
10	Device is accessed by bytes								00h
11	Checksum for preceding bytes between the previous checksum								7Eh

Storage Command 11h: Read FRU Data

The command returns the specified data from the FRU Inventory Info area. This is effectively a 'low level' direct interface to a non-volatile storage area. This means that the interface does not interpret or check any semantics or formatting for the data being accessed. The offset used in this command is a 'logical' offset that may or may not correspond to the physical address used in device that provides the non-volatile storage. For example, FRU information could be kept in FLASH at physical address 1234h, however offset 0000h would still be used with this command to access the start of the FRU information. IPMI FRU device data (devices that are formatted per [FRU]) as well as processor and DIMM FRU data always starts from offset 0000h unless otherwise noted.

Note that while the offsets are 16-bit values, allowing FRU devices of up to 64k words, the count to read, count returned and count written fields are only 8 bits. This is in recognition of the limitations on the sizes of messages. For example, as of this writing, IPMB messages are limited to 32-bytes total.

Table 57
Get FRU Inventory Area Info request and response

Byte	Data Field
Request Data	
1	FRU Device ID (0xFF = reserved)
2	FRU Inventory Offset to read LS Byte
3	FRU Inventory Offset to read MS Byte
4	Count to read
Response Data	
1	Completion code
2	Count returned
3-N+2	Requested data

Example Read FRU Data sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 58
Read FRU Data REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Storage 0A (even)						rsLUN is 0		28h
3	Checksum for the connection Header								98h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 0Ch						rqLUN is 2		32h
6	Command 11h - Read FRU data								11h
7	FRU Device ID								00h
8	FRU Inventory Offset LSB								00h
9	FRU Inventory Offset MSB								00h
10	Count to read (bytes)								08h
11	Checksum for preceding bytes between the previous checksum								35h

Table 59
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is 0B Storage (odd)						rqLUN is 2		2Eh
3	Checksum for the connection Header								52h
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 0Ch						rsLUN is 0		30h
6	Command 11h - Read FRU data								11h
7	(Completion Code 00 = 'OK')								00h
8	Count returned								08h
9	Format version 1								01h
10	Internal use area starting offset (00h indicates area not present)								00h
11	Chassis info area starting offset (00h indicates area not present)								00h
12	Board area starting offset (00h indicates area not present)								00h
13	Product info area starting offset								16h
14	Multi-record area starting offset (00h indicates area not present)								00h
15	PAD								00h
16	Common Header zero checksum								E9h
17	Checksum for preceding bytes between the previous checksum								77h

Sensor System (Get Sensor Reading Command 2Dh)

VITA 62 Power Supply Sensors

Vicor VITA 62 Power Supplies contain Sensor Data Records (SDRs) defined by the IPMI v2.0 Specification. All SDRs are Type 01h, Full Sensor Record. See *IPMI Specification* section 43 (p.520), Sensor Data Records for details.

Table 60
VITA 62 sensor data records

Sensor Number	Sensor Type	Description	SDR Record ID
0	Mandatory FRU sensor	FRU State	7
1		FRU IPMB Link	6
2		FRU Health	5
3		FRU Voltage	4
4		FRU Temperature	8
5		Payload Test Results	3
6		Payload Test Status	2
7	Analog threshold sensor	Input Voltage	9
8		VS1 Voltage	10
9		VS2 Voltage	11
10		VS3 Voltage	12
11		AUX2 Voltage	13
12		AUX3 Voltage	14
13		AUX1 Voltage	15
14		Input Current	16
15		VS1 Current	17
16		VS2 Current	18
17		VS3 Current	19
18		P6 Temperature	20
19		P1 Temperature	21
20		Mid-chassis Temperature	30
21	Analog threshold sensor	Input Power	25
22		VS1 Power	26
23		VS2 Power	27
24		VS3 Power	28
25		AUX2 Current	22
26		AUX3 Current	23
27		AUX1 Current	24
28		AUX Power	29

Mandatory FRU Sensors

This section describes the mandatory sensors defined by VITA 46.11.
Intelligent Platform Management Controllers (IPMC) are required to implement these sensors for an Intelligent FRU (FRU #0).

Table 61
Mandatory FRU sensors

Sensor Number	Sensor Type	Event/Reading Type	Description
0	**F0** Operational State	**0x6F** Sensor specific	FRU State Sensor
1	**F1** IPMB Link	**0x6F** Sensor specific	System IPMB Link Sensor
2	**F2** FRU Health	**0x04** "digital" Discrete Predictive Failure	FRU Health Sensor
3	**02** Voltage	**0x05** "digital" Discrete Limit Exceeded or Not	FRU Voltage Sensor
4	**F4** FRU Temperature	**0x6F** Sensor specific	FRU Temperature Sensor
5	**F5** Payload Test	**0x04** "digital" Discrete Predictive Failure	Payload Test Results
6	**F6** Payload Test Status	**0x03** "digital" Discrete State Asserted or Not	Payload Test Status

Sensor 0: Get Sensor Reading for FRU State Sensor

Table 62
Get Sensor 0 Reading: FRU State Sensor request and response

Byte	Data Field
Request Data	
1	Sensor number
Response Data	
1	Completion code
2	Sensor Reading N/A: write as 0x00, ignore on read
3	IPMI Information **d7** Event Messages Enable **d6** Sensor Scanning Enable **d5** Reading Unavailable **d4:d0** reserved
4	Current State Mask **d7** EFRU State M7 – Communication Lost **d6** FRU State M6 – Deactivation in Progress **d5** FRU State M5 – Deactivation Request **d4** FRU State M4 – FRU Active **d3:d2** N/A: write as 0x00, ignore on read **d1** FRU State M1 – FRU Inactive **d0** FRU State M0 – IPMC Inactive

Sensor 0: Event Message for FRU State Sensor

Table 63
FRU State Sensor event message request

Byte	Data Field
Request Data	
1	Event Message Rev 0x04
2	Sensor Type 0xF0
3	Sensor number
4	Event Type / Direction **d7** Event Direction: 0 = Assertion **d6:d0** Event Type 0x6F
5	Event Data 1 **d7:d4** 0x0A OEM code in Event Data 2, OEM code in Event Data 3 **d3:d0** Current State **0** – M0 – IPMC Inactive **1** – M1 – FRU Inactive **4** – M4 – FRU Active **5** – M5 – Deactivation Request **6** – M6 – Deactivation in Progress **7** – M7 – FRU Communication Lost **Other** – reserved
6	Event Data 2 **d7:d4** Cause of state change (see table 64) **d3:d0** Previous State **0** – M0 – IPMC Inactive **1** – M1 – FRU Inactive **4** – M4 – FRU Active **5** – M5 – Deactivation Request **6** – M6 – Deactivation in Progress **7** – M7 – FRU Communication Lost **Other** – reserved
7	Event Data 3 **d7:d0** FRU Device ID

Table 64
State-change causes

Value	Description
0x00	Normal State Change
0x01	Change commanded by chassis manager
0x02	Reserved
0x03	State Change due to programmatic action
0x04	Communication Lost or Regained
0x05	Communication Lost or Regained – locally detected
0x06	Surprise State Change of IPMC
0x07	State Change due to provided information
0x08	Invalid Hardware Address Detected
0x09	Unexpected Deactivation
0x0F	State Change cause unknown
0x10 – 0xFF	Reserved

Sensor 1: Get Sensor Reading for System IPMB Link Sensor

Table 65
Get Sensor 1 Reading: System IPMB Link Sensor request and response

Byte	Data Field
Request Data	
1	Sensor number
Response Data	
1	Completion code
2	Sensor reading **d7** IPMB-B override state **0** – Override state, bus isolated **1** – Local control state **d6:d4** IPMB-B Local Status **0** – No failure; bus enabled (if no override) **1** [e] – Unable to drive clock HI **2** [e] – Unable to drive data HI **3** [e] – Unable to drive clock LO **4** [e] – Unable to drive data LO **5** [e] – Clock Low timeout **6** [e] – Under test **7** – Undiagnosed Communications Failure **d3** IPMB-A override state **0** – Override state, bus isolated **1** – Local Control StateI **d2:d0** IPMB-B Local Status **0** – No failure; bus enabled (if no override) **1** [e] – Unable to drive clock HI **2** [e] – Unable to drive data HI **3** [e] – Unable to drive clock LO **4** [e] – Unable to drive data LO **5** [e] – Clock Low timeout **6** [e] – Under test **7** – Undiagnosed Communications Failure
3	IPMI Information **d7** Event Messages Enable **d6** Sensor Scanning Enable **d5** Reading Unavailable **d4:d0** Reserved; ignore on read
4	IPMI Link State **d7:d4** Reserved;write as 0, ignore on read **d3** IPMB-A enabled, IPMB-B enabled **d2** IPMB-A enabled, IPMB-B disabled **d1** IPMB-A disabled, IPMB-B enabled **d0** IPMB-A disabled, IPMB-B disabled **Note:** only one of the bits d0:d3 should bet set to 1 to indicate the status of bus A and bus B.

[e] Power supply is unable to detect this condition

Sensor 1: Event Message for System IPMB Link Sensor

Table 66
System IPMB Link Sensor event message request

Byte	Data Field
Request Data	
1	Event Message Rev 0x04
2	Sensor Type 0xF1 (VITA46.11 defined)
3	Sensor number
4	Event Type / Direction **d7** Event Direction: 0 = Assertion **d6:d0** Event Type 0x6F
5	Event Data 1 **d7:d4** 0x0A OEM code in Event Data 2, OEM code in Event Data 3 **d3:d0** Generic Offset **00h** IPMB-A disabled, IPMB-B disabled **01h** IPMB-A enabled, IPMB-B disabled **02h** IPMB-A disabled, IPMB-B enabled **03h** IPMB-A enabled, IPMB-B enabled
6	Event Data 2 **d7:d4** Channel number: typically 0x00 for VITA46.11 to indicate system IPMB **d3:d0** reserved
7	Event Data 3 **d7** IPMB-B override state **0** – Override state, bus isolated **1** – Local control state **d6:d4** IPMB-B Local Status **0** – No failure; bus enabled (if no override) **1** [e] – Unable to drive clock HI **2** [e] – Unable to drive data HI **3** [e] – Unable to drive clock LO **4** [e] – Unable to drive data LO **5** [e] – Clock Low timeout **6** [e] – Under test **7** – Undiagnosed Communications Failure **d3** IPMB-A override state **0** – Override state, bus isolated **1** – Local Control StateI **d2:d0** IPMB-A Local Status **0** – No failure; bus enabled (if no override) **1** [e] – Unable to drive clock HI **2** [e] – Unable to drive data HI **3** [e] – Unable to drive clock LO **4** [e] – Unable to drive data LO **5** [e] – Clock Low timeout **6** [e] – Under test **7** – Undiagnosed Communications Failure **Note:** Local Status will only indicate No Failure, Bus Enabled if no override or undiagnosed Communications Failure.

[e] Power supply is unable to detect this condition

Sensor 2: Get Sensor Reading for FRU Health Sensor

Table 67
Get Sensor 2 Reading: FRU Health Sensor request and response

Byte	Data Field
Request Data	
1	Sensor number
Response Data	
1	Completion code
2	Sensor Reading N/A Write as 0x00, ignore on read
3	IPMI Information **d7** Event messages enable **d6** Sensor Scanning Enable **d5** Reading Unavailable **d4:d0** Reserved
4	FRU Health **d7:d2** Reserved; write as 0x00, ignore on read **d1:d0** **00** – Reserved **01** – FRU functioning properly **10** – FRU not functioning properly **11** – Reserved

Sensor 2: Event Message for FRU Health Sensor

Table 68
FRU Health Sensor event message request

Byte	Data Field
Request Data	
1	Event Message Rev 0x04
2	Sensor Type 0xF2 (VITA46.11 defined)
3	Sensor number
4	Event Type / Direction **d7** Event Direction: 0 = Assertion **d6:d0** Event Type 0x04 Predictive Failure
5	Event Data 1 **d7:d4** 000b = unspecified bytes 2 and 3 **d3:d0** Generic Offset for state transition **0x00** Predictive Failure De-asserted **0x01** Predictive Failure Asserted

Sensor 3: Get Sensor Reading for FRU Voltage Sensor

Table 69
Get Sensor 3 Reading: FRU Voltage Sensor request and response

Byte	Data Field
Request Data	
1	Sensor number
Response Data	
1	Completion code
2	Sensor Reading N/A Write as 0x00, ignore on read
3	IPMI Information **d7** Event messages enable **d6** Sensor Scanning Enable **d5** Reading Unavailable **d4:d0** Reserved
4	Voltage Health **d7:d2** Reserved; write as 0x00, ignore on read **d1:d0** **00** – Reserved **01** – FRU voltages within normal range **10** – FRU asserts at least one voltage is out of normal range **11** – Reserved

Sensor 3: Event Message for FRU Voltage Sensor

Table 70
FRU Voltage Sensor event message request

Byte	Data Field
Request Data	
1	Event Message Rev 0x04
2	Sensor Type 0x02 (Voltage)
3	Sensor number
4	Event Type / Direction **d7** Event Direction: 0 = Assertion **d6:d0** Event Type 0x05 "digital" discrete limit exceeded/not exceeded
5	Event Data 1 **d7:d4** 000b = unspecified bytes 2 and 3 **d3:d0** Generic Offset for state transition **0x00** Limit not exceeded **0x01** Limit exeeded

Sensor 4: Get Sensor Reading for FRU Temperature Sensor

Table 71
Get Sensor 4 Reading: FRU Temperature Sensor request and response

Byte	Data Field
Request Data	
1	Sensor number
Response Data	
1	Completion code
2	Sensor Reading N/A Write as 0x00, ignore on read
3	IPMI Information **d7** Event messages enable **d6** Sensor Scanning Enable **d5** Reading Unavailable **d4:d0** Reserved
4	Temperature States **d7:d6** Reserved; ignore on read **d5** Temperature at or above upper non-recoverable threshold; **d4** Temperature at or above upper critical threshold **d3** Temperature at or above upper non-critical threshold **d2** Temperature at or below lower non-recoverable threshold **d1** Temperature at or below lower critical threshold **d0** Temperature at or below lower non-critical threshold **Note:** for data bits d5:d0 a value of 1 indicates an exceeded threshold.

Sensor 4: Event Message for FRU Temperature Sensor

Table 72
FRU Temperature Sensor event message request

Byte	Data Field
Request Data	
1	Event Message Rev 0x04
2	Sensor Type 0xF3 (VITA-defined OEM)
3	Sensor number
4	Event Type / Direction **d7** Event Direction: 0 = Assertion **d6:d0** Event Type 0x6F sensor-specific discrete
5	Event Data 1 **d7:d4** 000b = unspecified bytes 2 and 3 **d3:d0** **0x00** Change in bit 0 **0x01** Change in bit 1 **0x02** Change in bit 2 **0x03** Change in bit 3 **0x04** Change in bit 4 **0x05** Change in bit 5 [All other values reserved]

Sensor 5: Get Sensor Reading for Payload Test Results Sensor

Table 73
Get Sensor 5 Reading: Payload Test Results Sensor request and response

Byte	Data Field
Request Data	
1	Sensor number
Response Data	
1	Completion code
2	Sensor Reading N/A Write as 0x00, ignore on read
3	IPMI Information **d7** Event messages enable **d6** Sensor Scanning Enable **d5** Reading Unavailable **d4:d0** Reserved
4	Test Results **d7:d2** Reserved; ignore on read **d1:d0** **00** Reserved **01** FRU asserts last payload test succeeded **10** FRU asserts last payload test failed **11** Reserved

Sensor 5: Event Message for Payload Test Results Sensor

Table 74
Payload Test Results Sensor event message request

Byte	Data Field
Request Data	
1	Event Message Rev 0x04
2	Sensor Type 0xF4 (VITA-defined Payload Results)
3	Sensor number
4	Event Type / Direction **d7** Event Direction: 0 = Assertion **d6:d0** Event Type 0x04 Predictive Failure
5	Event Data 1 **d7:d4** 000b = unspecified bytes 2 and 3 **d3:d0** Generic Offset for state transition **0x00** Predictive Failure De-asserted **0x01** Predictive Failure Asserted

Sensor 6: Get Sensor Reading for Payload Test Results Status

Table 75
Get Sensor 6 Reading: Payload Test Results Status request and response

Byte	Data Field
Request Data	
1	Sensor number
Response Data	
1	Completion code
2	Sensor Reading N/A Write as 0x00, ignore on read
3	IPMI Information **d7** Event messages enable **d6** Sensor Scanning Enable **d5** Reading Unavailable **d4:d0** Reserved
4	Test Status **d7:d2** Reserved; write as 0x00, ignore on read **d1:d0** **00** Reserved **01** FRU asserts payload test is not in progress **10** FRU asserts payload test is in progress **11** Reserved

Sensor 6: Event Message for Payload Test Results Status

Table 76
Payload Test Results Status event message request

Byte	Data Field
Request Data	
1	Event Message Rev 0x04
2	Sensor Type 0xF5 (VITA-defined Payload Results)
3	Sensor number
4	Event Type / Direction **d7** Event Direction: 0 = Assertion **d6:d0** Event Type 0x03 State Assertion
5	Event Data 1 **d7:d4** 000b = unspecified bytes 2 and 3 **d3:d0** Generic Offset for state transition **0x00** State De-asserted **0x01** State Asserted

Analog Threshold Sensors

Threshold-based sensors update event status by comparing the analog reading from a sensor to a set of threshold values. Threshold enumerations may be considered a special case of the discrete sensor type. The Event/Reading Type Code for threshold-based sensors is specified in the *IPMI Specification* Table 42-2 (p.503), Generic Event/Reading Type Codes (Table 52 in this document is an excerpt). The offsets specify each particular possible threshold state. Threshold-based sensors return a different response to the Get Sensor Reading command than discrete sensors. The response to a Get Sensor Reading command for a threshold-based sensor contains the present analog reading from the sensor in addition to the discrete threshold comparison status bit field.

Not all sensors are available on every VITA 62 power supply.
Refer to the power supply data sheet for a list of valid sensors.

Table 77
Analog threshold sensors

Sensor Number	Sensor Type	Description	Unit
7	**02** Voltage	Input Voltage	V
8	**02** Voltage	VS1	V
9	**02** Voltage	VS2	V
10	**02** Voltage	VS3	V
11	**02** Voltage	AUX2	V
12	**02** Voltage	AUX3	V
13	**02** Voltage	AUX1	V
14	**03** Current	Input Current	A
15	**03** Current	VS1	A
16	**03** Current	VS2	A
17	**03** Current	VS3	A
18	**01** Temperature	P6 card edge	K
19	**01** Temperature	P16 card edge	K
20	N/A	N/A	N/A
21	**0B** other units	Input Power	W
22	**0B** other units	VS1	W
23	**0B** other units	VS2	W
24	**0B** other units	VS3	W
25	**03** Current	AUX2	A
26	**03** Current	AUX3	A
27	**03** Current	AUX1	A
28	**0B** other units	AUX Power	W

Example Get Sensor Reading for VS1 Voltage
sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 78
Get Sensor Reading for VS1 voltage REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor/Event 04 (even)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 0Ch						rqLUN is 2		32h
6	Command 2Dh- Get Sensor Reading								2Dh
7	Sensor Number – VS1 voltage = 08h								08h
8	Checksum for preceding bytes between the previous checksum								19h

Table 79
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 0Ch						rsLUN is 0		30h
6	Command 2Dh - Get Sensor Reading								2Dh
7	(Completion Code 00 = 'OK')								00h
8	Sensor reading								95h
9	Sensor Status								40h
10	Sensor Data 1 – Threshold status flags								C0h
11	Checksum for preceding bytes between the previous checksum								CEh

Reading Result

- VS1 voltage was 9 + 2.98 = 11.98V
- Sensor scanning enabled, Event Messages disabled for this sensor.
- VS1 voltage is within normal operating range (no thresholds exceeded).

Example Get Sensor Reading for VS3 Current
sent from device at IPMB address 0x80 to device at IPMB address 0x40.

Table 80
Get Sensor Reading for VS3 current REQUEST

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB Child address)								40h
2	NetFn is Sensor/Event 04 (even)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
5	rqSeq = 0Ch						rqLUN is 2		32h
6	Command 2Dh - Get Sensor Reading								2Dh
7	Sensor Number – VS3 current = 11h								11h
8	Checksum for preceding bytes between the previous checksum								10h

Table 81
RESPONSE message transmitted

Byte	Bits								HEX Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h (Requestor's VPX IPMB Child address)								80h
2	NetFn is Sensor/Event 05 (odd)						rqLUN is 2		16h
3	Checksum for the connection Header								6Ah
4	rsSA = 40h (VPX IPMB Child address)								40h
5	rqSeq = 0Ch						rsLUN is 0		30h
6	Command 2Dh - Get Sensor Reading								2Dh
7	(Completion Code 00 = 'OK')								00h
8	Sensor reading								63h
9	Sensor Status								40h
10	Sensor Data 1 – Threshold status flags								C0h
11	Checksum for preceding bytes between the previous checksum								00h

Reading Result

- VS3 current = 19.8A
- Sensor scanning enabled, Event Messages disabled for this sensor.
- VS3 current is within normal operating range (no thresholds exceeded).

Glossary

Asserted	Active-high (positive true) signals are asserted when in the high electrical state (near power potential). Active-low (negative true) signals are asserted when in the low electrical state (near ground potential).
BMC	Baseboard Management Controller
De-asserted	A signal is de-asserted when in the inactive state. Active-low signal names have "_L" appended to the end of the signal mnemonic. Active-high signal names have no "_L" suffix. To reduce confusion when referring to active-high and active-low signals, the terms one/zero, high/low and true/false are not used when describing signal states.
EvM	Notation for 'Event Message.' See text for definitions of 'Event Message.'
FPC	Front Panel Controller.
FRU	Field Replaceable Unit. A module or component which will typically be replaced in its entirety as part of a field service repair operation.
Hard Reset	A hardware reset event that initializes components and invalidates caches for a system or subsystem. In the context of this specification, the term Hard Reset is generally used to refer to System Hard Resets, where System Hard Resets are Hard Resets of the computer system that do not reset the BMC, Satellite Controllers, or other elements of the platform management subsystem. Unless explicitly stated, Hard Resets or System Hard Resets do not refer to resets of the BMC or other elements of the platform management subsystem.
ICMB	Intelligent Chassis Management Bus. A serial, differential bus designed for IPMI messaging between host and peripheral chassis. Refer to [ICMB] for more information.
IPM	Intelligent Platform Management.
IPMB	Intelligent Platform Management Bus. Name for the architecture, protocol and implementation of a special bus that interconnects the baseboard and chassis electronics and provides a communications media for system platform management information. The bus is built on I^2C and provides a communications path between 'management controllers' such as the BMC, FPC, HSC, PBC and PSC.
LUN	Logical Unit Number. In the context of the Intelligent Platform Management Bus protocol, this is a sub-address that allows messages to be routed to different 'logical units' that reside behind the same I^2C™ Child address.
Payload	For this specification, the term 'payload' refers to the information bearing fields of a message. This is separate from those fields and elements that are used to transport the message from one point to another, such as address fields, framing bits, checksums, etc. In some instances, a given field may be both a payload field and a transport field.
Re-arm	Re-arm, in the context of this document, refers to resetting internal device state that tracks that an event has occurred such that the device will re-check the event condition and re-generate the event if the event condition exists.
SDR	Sensor Data Record. A data record that provides platform management sensor type, locations, event generation and access information.
SEL	System Event Log. A non-volatile storage area and associated interfaces for storing system platform event information for later retrieval.
SMS	System Management Software. Designed to run under the OS.
Soft Reset	A reset event in the system which forces CPUs to execute from the boot address but does not change the state of any caches or peripheral devices.

References

Intelligent Platform Management Interface Specification Second Generation v2.0

Document Revision 1.1 October 1, 2013 February 11, 2014 E6 Markup Intel Hewlett-Packard NEC Dell; Copyright © 2014 Intel Corporation, Hewlett-Packard Company, NEC Corporation, Dell Inc., All rights reserved.
https://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/ipmi-second-gen-interface-spec-v2-rev1-1.pdf

Approved ANSI Standard June 2015

ANSI/VITA 46.11-2015, System Management on VPX; Copyright © VITA 2015. All Rights Reserved.
https://www.vita.com/Standards

Limitation of Warranties

Information in this document is believed to be accurate and reliable. HOWEVER, THIS INFORMATION IS PROVIDED "AS IS" AND WITHOUT ANY WARRANTIES, EXPRESSED OR IMPLIED, AS TO THE ACCURACY OR COMPLETENESS OF SUCH INFORMATION. VICOR SHALL HAVE NO LIABILITY FOR THE CONSEQUENCES OF USE OF SUCH INFORMATION. IN NO EVENT SHALL VICOR BE LIABLE FOR ANY INDIRECT, INCIDENTAL, PUNITIVE, SPECIAL OR CONSEQUENTIAL DAMAGES (INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR SAVINGS, BUSINESS INTERRUPTION, COSTS RELATED TO THE REMOVAL OR REPLACEMENT OF ANY PRODUCTS OR REWORK CHARGES).

Vicor reserves the right to make changes to information published in this document, at any time and without notice. You should verify that this document and information is current. This document supersedes and replaces all prior versions of this publication.

All guidance and content herein are for illustrative purposes only. Vicor makes no representation or warranty that the products and/or services described herein will be suitable for the specified use without further testing or modification. You are responsible for the design and operation of your applications and products using Vicor products, and Vicor accepts no liability for any assistance with applications or customer product design. It is your sole responsibility to determine whether the Vicor product is suitable and fit for your applications and products, and to implement adequate design, testing and operating safeguards for your planned application(s) and use(s).

VICOR PRODUCTS ARE NOT DESIGNED, AUTHORIZED OR WARRANTED FOR USE IN LIFE SUPPORT, LIFE-CRITICAL OR SAFETY-CRITICAL SYSTEMS OR EQUIPMENT. VICOR PRODUCTS ARE NOT CERTIFIED TO MEET ISO 13485 FOR USE IN MEDICAL EQUIPMENT NOR ISO/TS16949 FOR USE IN AUTOMOTIVE APPLICATIONS OR OTHER SIMILAR MEDICAL AND AUTOMOTIVE STANDARDS. VICOR DISCLAIMS ANY AND ALL LIABILITY FOR INCLUSION AND/OR USE OF VICOR PRODUCTS IN SUCH EQUIPMENT OR APPLICATIONS AND THEREFORE SUCH INCLUSION AND/OR USE IS AT YOUR OWN RISK.

Terms of Sale

The purchase and sale of Vicor products is subject to the Vicor Corporation Terms and Conditions of Sale which are available at: (http://www.vicorpower.com/termsconditionswarranty)

Export Control

This document as well as the item(s) described herein may be subject to export control regulations. Export may require a prior authorization from U.S. export authorities.

Contact Us: http://www.vicorpower.com/contact-us

Vicor Corporation
25 Frontage Road
Andover, MA, USA 01810
Tel: 800-735-6200
Fax: 978-475-6715
www.vicorpower.com

email
Customer Service: custserv@vicorpower.com
Technical Support: apps@vicorpower.com