APPLICATION NOTE | AN:802

VITA 62 Power Supply 270V_{DC} Input I^2C Communication

Vincent Gucciardo
Senior Firmware Engineer

Contents **Page**

Introduction

The Vicor VITA 62 power supply is a COTs power supply that is designed for 3U OpenVPX systems. The module utilizes Vicor 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. For information regarding parallel operation please refer to AN:801. This document details I^2C communication for all versions of VIT270H3U600yzzz (product data sheet available at the Vicor website).

Overview

I^2C Communication is standard on all of the Vicor VITA 62 Power Supplies. This application note is specific to all options powered from a $270V_{DC}$ input (220 – $320V_{DC}$). If the user chooses not to utilize the I^2C interface, the power supply is fully functional with discrete lines for controlling & monitoring the power supply.

- Discrete control lines (ENABLE & INHIBIT) are used for switching the state of the outputs. After power-up, the default is for the outputs to be controlled by the discrete control lines (see Table 1 below). The user can send a command over the I^2C port to disable discrete control (See pages 13 – 14 for details).

Table 1
ENABLE / INHIBIT discrete control lines

ENABLE	INHIBIT	Outputs
Logic High or Float	Logic High or Float	All outputs (VS1, VS2, VS3, $+12V_{AUX}$, –12V AUX, $+3.3V_{AUX}$) are OFF
Logic High or Float	Low	All outputs (VS1, VS2, VS3, $+12V_{AUX}$, –12V AUX, $+3.3V_{AUX}$) are OFF
Low	Logic High or Float	All outputs (VS1, VS2, VS3, $+12V_{AUX}$, –12V AUX, $+3.3V_{AUX}$) are ON
Low	Low	VS1, VS2, VS3, $+12V_{AUX}$, $-12V_{AUX}$ are OFF, only $+3.3V_{AUX}$ is ON

- Discrete monitor lines (Fail & System Reset) can be used to monitor health of the power supply

The I^2C Communication IPMI Interface can be used to monitor the outputs and the health of the power supply (see the IPMI Interface section, pages 3 – 6). The IPMI data interface is compliant with the requirements of VITA 46.11, VITA 62-2016, and the IPMI v2.0 specifications. In addition to the IPMI interface, the Vicor VITA 62 Power Supply provides the user with the following additional monitoring and control capabilities:

- Composite data output for the status register, all six outputs (voltage and current), rail temperature, P/N, S/N, date code, hardware & firmware revision (see pages 12 – 14)
- Data output for the six output voltages (see page 15)
- Data output for the three main currents (see page 16)
- Data output for the three AUX output currents (see page 16)
- Data output for the two outside rail temperatures (see page 17)

Two bytes are used for voltage and current readings in the composite data output; these readings will have better resolution than the 1 byte IPMI readings. For example: The resolution for the +5V reading is 20mV using the IPMI method (pages 7 – 12) as compaired to 1mV using the methods shown in page 15.

I^{2}C Configuration

Hardware Interface

The I^{2}C interfaces comply with the Philips I^{2}C design requirements. The electrical interface is based on I^{2}C parameters at 100kHz, and the backplane or I^{2}C parent controller must provide pull-up resistors on SDA and SCL lines to a 3.3V rail (typical value for the pull-up resistors is 4.7kΩ). Table 2 shows the I^{2}C pins of the VITA 62 connection and the pin connection to the system management bus.

Table 2
I^{2}C ports

Pin Number	Pin Name	Description
C5	SM0	Primary I^{2}C Clock Line
D5	SM1	Primary I^{2}C Data Line
A6	SM2	Redundant I^{2}C Clock Line
B6	SM3	Redundant I^{2}C Data Line

Geographical Addressing

The 7-bit I^{2}C address is configured using Geographical Addressing pins defined by VITA 46.11. The *GA pins have pull-up resistors internal to the power supply to 3.3V. When left open, the address will be 0x20, with both grounded the address will be 0x23, see Table 3 below. The IPMB address is defined by VITA 46.11 for communication over the IPMI interface. The 8-bit IPMB address is determined by multiplying its 7-bit I^{2}C address by 2.

Table 3
I^{2}C address assignment

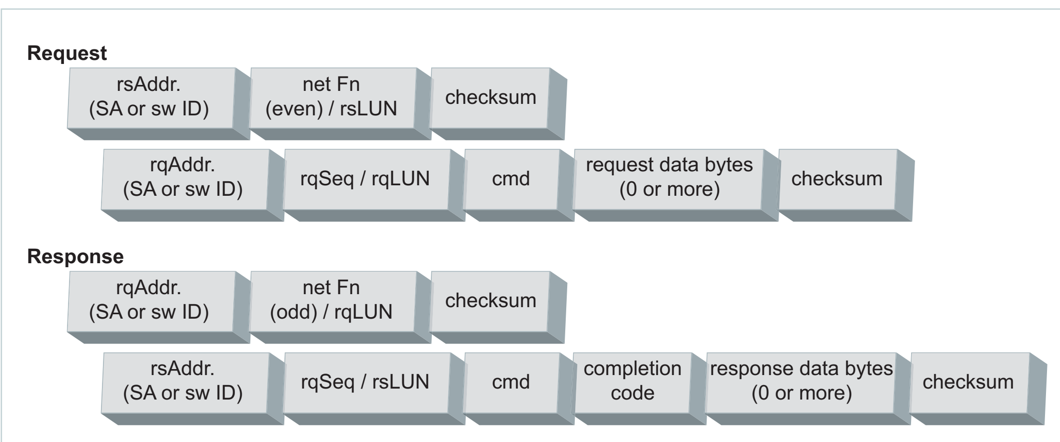
GA1* (Pin A5)	GA0* (Pin A5)	I^{2}C Address	IPMB Address
Unconnected	Unconnected	20h	40h
Unconnected	Biased to ground on the backplane	21h	42h
Biased to ground on the backplane	Unconnected	22h	44h
Biased to ground on the backplane	Biased to ground on the backplane	23h	46h

Both primary and redundant I^{2}C buses have the same I^{2}C address and are identical in their functionality, but operate independently.

IPMI Interface

The IPMI interface allows the user to retrieve power status information using standard IPMB Communication shown below:

Figure 1
IPMI LAN message formats [(Intelligent Platform Management Interface Specification, Second Generation v.2.0, figure 13-4, page 136)]



The responder's Child Address (rsSA) is an 8-bit IPMI address per the I^{2}C Configuration section above. The supply supports Sensor/Event messages (netFn 04, 05). The checksum is the 2's complement of the preceding bytes.

IPMI Sensor Overview

Sensor Reading

A variety of temperature, voltage, and current readings are taken within the power supply. The results are provided as a single byte of data within the interface protocol for data interchange upon request from a system management function within the chassis.

Sensor Warnings

In addition to the parametric readings from the supply, it has the ability to generate warning events to the system controller. These are "flags" set, or reset, based upon the internal readings of various parameters. They will generate a message to the central controller on the bus, and their status can be read as sensor data as well.

Critical Warnings

These are set when an operating parameter (voltage, temperature or current) reaches a critical value approaching the limits of the power supply specifications. They are "health" events indicating that some parameter is approaching a value beyond which the supply may not remain in tolerance.

Non-Recoverable (NR) Warnings

These are set when parameters fall outside of the limits of the power supply operation. If the high-side NR threshold is exceeded, the power supply will turn all six of the outputs OFF. If a shut down occurs, the internal 3.3-volt supply which powers the microcontroller is kept active and I^2C communications will be active and valid.

Data Format

The following formula defined in the IPMI specification is used to convert the one-byte "raw" sensor readings to a value in the desired units.

$$Y = \left((Mx + B) \bullet 10^{K1}\right) \bullet 10^{K2} \qquad (1)$$

Where:

x is a one-byte unsigned integer, received form the VPX power supply

Y is the calculated value in the appropriate units.

M is a signed integer (the multiplier)

B is a signed integer (the offset)

K1 is a signed exponent (for the B term)

K2 is a signed exponent (for the Result)

Analog Sensors

Table 4 provides the conversion coefficients for all of the analog sensors. It also includes a simple equation with all of the coefficients already factored in that can be used to convert the reading to the value with the appropriate units.

Table 4
Analog sensors reading conversion factors

Sensor			Units	Simplified Conversion Equation [a]	Sensor Type Code	Conversion Coefficients			
#	Hex	Name				M	B	K1	K2
7	0x07	Input Voltage	V	Reserved Not supported at this time	02h Voltage	N/A	N/A	N/A	N/A
8	0x08	+12V Voltage	V	Actual = Reading • 0.054		54	0	0	-3
9	0x09	+3.3V Voltage	V	Actual = Reading • 0.018		18	0	0	–3
10	0x0A	+5V Voltage	V	Actual = Reading • 0.046		46	0	0	–3
11	0x0B	$+3.3V_{AUX}$ Voltage	V	Actual = Reading • 0.018		18	0	0	–3
12	0x0C	$+12V_{AUX}$ Voltage	V	Actual = Reading • 0.054		54	0	0	–3
13	0x0D	$–12V_{AUX}$ Voltage	V	Actual = Reading • –0.054		–54	0	0	–3
14	0x0E	Input Current	A	Reserved Not supported at this time	03h Current	N/A	N/A	N/A	N/A
15	0x0F	+12V Current	A	Actual = Reading • 0.20		20	0	0	–2
16	0x10	+3.3V Current	A	Actual = Reading • 0.20		20	0	0	–2
17	0x11	+5V Current	A	Actual = Reading • 0.20		20	0	0	–2
18	0x12	Card Edge P6 Left	K	Actual = 200 + Reading	01h Temperature	1	20	1	0
19	0x13	Card Edge P1 Right	K	Actual = 200 + Reading		1	20	1	0
20	0x14	Mid-chassis Temp	K	Reserved Not supported at this time		N/A	N/A	N/A	N/A
21	0x15	Input Power	W		0Bh Other Units-based Sensor	N/A	N/A	N/A	N/A
22	0x16	+12V Power	W	Actual = Reading • 2.5		25	0	0	–1
23	0x17	+3.3V Power	W	Actual = Reading • 0.4		4	0	0	–1
24	0x18	+5V Power	W	Actual = Reading		10	0	0	–1
25	0x19	$+3.3V_{AUX}$ Current	A	Actual = Reading • 0.05	03h Current	50	0	0	–3
26	0x1A	$+12V_{AUX}$ Current	A	Actual = Reading • 0.02		20	0	0	–3
27	0x1B	$–12V_{AUX}$ Current	A	Actual = Reading • 0.02		20	0	0	–3
28	0x1C	AUX Power	W	Actual = Reading • 0.8	0Bh Other Units-based Sensor	8	0	0	–1
33	0x22	Output Power	W	Actual = Reading • 3		30	0	0	–1

[a] Reading is a one-byte value (0 – 255).

Analog Sensors (Cont.)

Table 5 provides the threshold limits used for each of the Analog Sensors.

- *Critical Threshold:* The power supply is still within specification, but one (or more) of the sensors is close to the boundary of its operating range.
- *Non-Recoverable (NR) Thresholds*: One (or more) of the sensors is not within its allowable operating range. If a Non-Recoverable threshold is continuously exceeded for a duration of two minutes, the power supply will pull the FAIL & SYSTEM RESET lines low and turn all six outputs OFF.

Table 5
Analog sensors
Critical and NR thresholds

Sensor			Conversion Coefficient				Low Critical Threshold		High Critical Threshold		Low NR Threshold		High NR Threshold		Hysteresis	
#	**Name**	**Unit**	**M**	**B**	**K1**	**K2**	**x**	**Y**	**x**	**Y**	**x**	**Y**	**x**	**Y**	**x**	**Y**
7	Input Voltage	V	Reserved Not supported at this time													
8	+12V Voltage	V	54	0	0	–3	213	11.5	231	12.474	209	11.286	236	12.744	3	0.162
9	+3.3V Voltage	V	18	0	0	–3	180	3.24	191	3.438	176	3.168	193	3.474	3	0.054
10	+5V Voltage	V	46	0	0	–3	105	4.83	112	5.15	102	4.692	113	5.198	3	0.138
11	$+3.3V_{AUX}$ Voltage	V	18	0	0	–3	180	3.24	191	3.438	175	3.15	192	3.456	3	0.054
12	$+12V_{AUX}$ Voltage	V	54	0	0	–3	212	11.44	231	12.474	208	11.232	235	12.69	3	0.162
13	$-12V_{AUX}$ Voltage	V	–54	0	0	–3	227	–12.26	210	–11.34	208	–11.23	232	–12.53	3	–0.162
14	Input Current	A	Reserved Not supported at this time													
15	+12V Current	A	20	0	0	–2	0	0	180	36	0	0	230	46	20	4
16	+3.3V Current	A	20	0	0	–2	0	0	80	16	0	0	120	24	10	2
17	+5V Current	A	20	0	0	–2	0	0	13	26	0	0	175	35	10	2
18	Card Edge P6	K	1	20	1	0	38	238	158	358	32	232	168	368	10	10
19	Card Edge P1	K	1	20	1	0	38	238	158	358	32	232	168	368	10	10
20	Mid-chassis Temperature	K	Reserved Not supported at this time													
21	Input Power	W														
22	+12V Power	W	25	0	0	–1	0	0	192	480	0	0	220	550	6	15
23	+3.3V Power	W	4	0	0	–1	0	0	165	66	0	0	195	78	6	2.4
24	+5V Power	W	10	0	0	–1	0	0	130	130	0	0	172	172	6	6
25	$+3.3V_{AUX}$ Current	A	50	0	0	–3	0	0.0	130	6.5	0	0.0	165	8.25	5	0.25
26	$+12V_{AUX}$ Current	A	20	0	0	–3	0	0.0	45	0.9	0	0.0	70	1.4	5	0.1
27	$-12V_{AUX}$ Current	A	20	0	0	–3	0	0.0	45	0.9	0	0.0	70	1.4	5	0.1
28	AUX Power	W	8	0	0	–1	0	0	123	98.4	0	0	140	112	5	4
33	Output Power	W	30	0	0	–1	0	0	192	576	0	0	215	645	4	12

IPMI Commands

Sensor Device Command

The power supply supports the IPMI Get Sensor Reading command in VITA 46.11 and IPMI specifications.

Table 6
Sensor Device commands

Command Name	NetFn	CMD
Get Sensor Reading	Sensor/Event	2Dh

Get Sensor Reading	
Data Type	**Description**
Request Data 1	Sensor Number (FFh = reserved)
Response Data 1	Completion Code
Response Data 2	Sensor Reading Byte 1: byte of reading Write as 00h if sensor does not return a numeric (analog) reading, ignore on read
Response Data 3	Provides information on the sensor.
Response Data 4	For Analog Sensor (7-28): Indicates where the reading stands against the threshold values For Discrete Sensor (2, 3, or 4) Indicates the state of the sensor

Event Commands

The power supply supports Event Messages as defined in VITA 46.11 for FRU Health Sensor, FRU Voltage and FRU Temperature sensors.

Table 7
Event commands

Command Name	NetFn	CMD
Set Event Receiver	Sensor/Event	00h
Get Event Receiver	Sensor/Event	01h
Platform Event	Sensor/Event	02h

FRU Health Sensor Event Message (Pending)	
Data Type	**Description**
Request Data 1	Event Message Rev=04h (IPMI 1.5)
Request Data 2	Sensor Type = F2h (VITA-defined FRU Health)
Request Data 3	Sensor Number
Request Data 4	[7] – Event Direction: 0b = Assertion, 1b = Deassertion [6:0] – Event Type: 04h (Predictive Failure)
Request Data 5	Event Data 1 [7:4] – 0000b [3:0] – 0h = (change in) Predictive Failure Deasserted – 1h = (change in) Predictive Failure Asserted

FRU Voltage Sensor Event Message (Pending)	
Data Type	**Description**
Request Data 1	Event Message Rev=04h (IPMI 1.5)
Request Data 2	Sensor Type = 02h (Voltage)
Request Data 3	Sensor Number
Request Data 4	[7] – Event Direction: 0b = Assertion, 1b = Deassertion [6:0] – Event Type: 05h (Predictive Failure)
Request Data 5	Event Data 1 [7:4] – 0000b [3:0] – 0h = "Limit Not Exceeded" status bit – 1h ="Limit Exceeded" status bit

IPMI Commands (Cont.)

Event Commands (Cont.)

Table 7 (Cont.)
Event commands

FRU Temperature Sensor Event Message (Pending)	
Data Type	**Description**
Request Data 1	Event Message Rev=04h (IPMI 1.5)
Request Data 2	Sensor Type = 03h (Temperature)
Request Data 3	Sensor Number
Request Data 4	[7] – Event Direction: 0b = Assertion, 1b = Deassertion [6:0] – Event Type: 06h (Sensor-specific discrete)
Request Data 5	Event Data 1 [7:4] – 0000b [3:0] 0h = Change in bit 0 (temp at or below lower non-critical) state 1h = Change in bit 1 (temp at or below lower critical) state 2h = Change in bit 2 (temp at or below lower non-recoverable) state 3h =Change in bit 0 (temp at or above upper non-critical) state 4h = Change in bit 1 (temp at or above upper critical) state 5h = Change in bit 2 (temp at or above upper non-recoverable) state

Event Messages

Each of the threshold (analog) sensors will create their own message with specific information about crossing a threshold as either being Asserted or Deasserted. In addition, there are three summary discrete sensors (2, 3 and 4) which contain summary information which will also create event messages.

Threshold Events

The sensors are continuously reading and the most recent reading is stored for the responses to requests for sensor values. In addition, each reading is compared to the critical threshold and to the non-recoverable threshold. No non-critical thresholds have been defined for the power supply.

A sensor reading will give the most recent reading (expressed as a single byte with the value interpreted according to Equation 1) along with the status of any warning flags. All sensor events are self-resetting, so no additional command is needed to reset the event.

Discrete Sensors

Sensors 2, 3, and 4 contain summary information and will generate their own event message.

- Sensor 2 is a summary of all the data and is an indication of the overall health of the power supply.
- Sensor 3 contains a summary of all voltage thresholds that are supported.
- Sensor 4 contains a summary of all temperature threshold events. Individual sensors should be read for specific information.

Non-Recoverable Events

The analog sensors all have a non-recoverable threshold beyond which the power supply may be irreversibly damaged and such an event will start an automated shut-down sequence for the power supply.

First, an event message will be created indicating that a non-recoverable event has occurred. This could be due to a cooling failure or an overload condition on an output which, potentially, could be reversed. Sending a message starts a timer. At the end of the timer, providing that the event has not been reset, the power supply will turn off all of the outputs. It will pull the FAIL* and SYSRESET* lines low. If a shut down occurs, the internal 3.3-volt supply which powers the microcontroller is kept active, meaning the I^2C communications will be active and valid.

IPMI Message Handling

Message Queuing

The Power Supply can queue up to 16 incoming messages from the requester. If the requester sends more than 16 messages in a row while the response from the power supply is still pending, subsequent messages will be ignored.

Corrupted Request

If a request is received with a bad checksum, the power supply will ignore the request entirely and let the requestor retry the message.

Unexpected Request

If an IPMI request message is received by the power supply that contains an invalid sensor number, the request will be ignored and no response will be generated. If an IPMI request message is received for a valid sensor number that is not supported by the power supply, the power supply will respond with a message containing a completion code of C1, indicating that sensor is not supported. (See IPMI Example 3 on page 12.)

Message Timeout

Once a valid request is received and a response is generated by the power supply, it is expected that the requestor will ACK the response message. However, if the response is not transmitted successfully for any reason, (e.g., bad checksum), the power supply will attempt the response three times. If the message is still not transmitted after three attempts, the response will be disregarded.

IPMI Examples

Example 1: Get Sensor Reading for Rail temperature on P1 side of Unit (Sensor #19)

Table 8
REQUEST example for reading the P1 rail temperature from the IPMI SENSOR for unit at address 0x20

Byte	Bits								Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB address, LS is always 0)								40h
2	Net Fn (even) is 04 (Sensor/Event)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requester's child address, LS always 0)								80h
5	rqSeq = 1						LUN is 0		04h
6	Command 2Dh- Get Sensor Reading								2Dh
7	Sensor number 13h (card edge P1)								13h
8	Checksum for preceding bytes between the previous checksum								3Ch

Table 9
RESPONSE message transmitted

Byte	Bits								Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h								80h
2	Net Fn (odd) is 05 (Sensor/Event)						rqLUN is 0		14h
3	Checksum for the connection Header								6Ch
4	rsSA = 40h (Responder's child address)								40h
5	rqSeq = 1						rsLUN is 0		04h
6	Command 2Dh- Get Sensor Reading								2Dh
7	00h (00h means Completion Code = 'OK')								00h
8	71h Sensor Reading (Temperature = 200 kelvin + 71h kelvin = 313 kelvin)								71h
9	40h sensor information (Scanning enabled and all event messages disabled from this sensor)								40h
10	C0h Threshold Comparison (Sensor reading is within normal range) See Table 10 for breakdown for Byte 10 (Threshold Comparison)								C0h
11	Checksum for preceding bytes between the previous checksum								1Eh

Table 10
Definition for Byte 10 of Response Message (Threshold Comparison)

Bit	Reading to Indicate High Threshold
7	Always 1 - Reserved Ignore on read
6	Always 1 - Reserved Ignore on read
5	at or above upper Non-recoverable Threshold
4	at or above upper Critical Threshold
3	Always 0, since the non-critical thresholds are not used.
2	at or below lower Non-recoverable Threshold
1	at or below lower Critical Threshold
0	Always 0, since the non-critical thresholds are not used.

Examples:

- If ‘reading’ is within the normal range, Byte 10 = C0
- If ‘reading’ < Lower NR Threshold, Byte 10 = C6
- If Lower NR Threshold < ‘reading’ < Lower Critical Threshold, Byte 10 = C2
- If Upper Critical Threshold < ‘reading’ < Upper NR Threshold, Byte 10 = D0
- If ‘reading’ > Upper NR Threshold, Byte 10 = F0

IPMI Examples (Cont.)

Example 2: Get Sensor Reading for +12V Voltage

Table 11
REQUEST example for reading +12V voltage the IPMI SENSOR for unit at address 0x20

Byte	Bits								Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB address, LS is always 0)								40h
2	Net Fn (even) is 04 (Sensor/Event)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requester's child address, LS always 0)								80h
5	rqSeq = 1						LUN is 0		04h
6	Command 2Dh- Get Sensor Reading								2Dh
7	Sensor number 08h (+12V voltage)								08h
8	Checksum for preceding bytes between the previous checksum								47h

Table 12
RESPONSE message transmitted

Byte	Bits								Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h								80h
2	Net Fn (odd) is 05 (Sensor/Event)						rqLUN is 0		14h
3	Checksum for the connection Header								6Ch
4	rsSA = 40h (Responder's child address)								40h
5	rqSeq = 1						rsLUN is 0		04h
6	Command 2Dh – Get Sensor Reading								2Dh
7	00h (00h means Completion Code = 'OK')								00h
8	95h Sensor Reading (Voltage = 9 + (149 • 0.02) = 11.98 volts)								95h
9	40h sensor information (Scanning enabled and all event messages disabled from this sensor)								40h
10	C0h Threshold Comparison (Sensor reading is within normal range) Table 10 shows the breakdown for Byte 10 (Threshold Comparison)								C0h
11	Checksum for preceding bytes between the previous checksum								FAh

IPMI Examples (Cont.)

Example 3: Get Sensor Reading- If attempted for a reserved item that is not supported at this time

Table 13
REQUEST example for reading the input voltage from the IPMI SENSOR for unit at address 0x20

Byte	Bits								Value
	7	6	5	4	3	2	1	0	
1	rsSA = 40h (VPX IPMB address, LS is always 0)								40h
2	Net Fn (even) is 04 (Sensor/Event)						rsLUN is 0		10h
3	Checksum for the connection Header								B0h
4	rqSA = 80h (Requester's child address, LS always 0)								80h
5	rqSeq = 1						LUN is 0		04h
6	Command 2Dh- Get Sensor Reading								2Dh
7	Sensor number 07h (card edge P1)								07h
8	Checksum for preceding bytes between the previous checksum								48h

Table 14
RESPONSE message transmitted

Byte	Bits								Value
	7	6	5	4	3	2	1	0	
1	rqSA = 80h								80h
2	Net Fn (odd) is 05 (Sensor/Event)						rqLUN is 0		14h
3	Checksum for the connection Header								6Ch
4	rsSA = 40h (Responder's child address)								40h
5	rqSeq = 1						rsLUN is 0		04h
6	Command 2Dh – Get Sensor Reading								2Dh
7	C1h (C1h means the sensor is not supported)								C1h
8	00h Sensor Reading (since the sensor is not supported)								00h
9	00h sensor information								00h
10	00h Threshold Comparison								00h
11	Checksum for preceding bytes between the previous checksum								CEh

Special I²C Commands & Reading

For several of the applications, the user may find the special commands useful for polling the Power Supply.

The format for using this set of commands to get sensor reading is shown in the table below.

Table 15
Special command format

Command Name	CMD Hex	Description
Composite Sensor	21h	64 bytes of scanned sensor data. See sensor device command section on page 7 for details and Table 6 for a complete breakdown of the 64 bytes (0 to 63)
Status Write Command	55h	Write Status byte on Composite Sensor.
Firmware release date	44h	22 byte response. Month/Day/Year Hr/Min/Sec in ASCII form.
Hardware Address	45h	3 byte response. Reports address set by GA*-GA4

General Information and Start Up Check

Composite Sensor Read Command (0x21)

This command is extremely useful for the initial check of the power supply. In addition to providing high resolution voltage and current readings for the six outputs, all of the pertinent information for the power supply (part number, serial number, date code, hardware revision, and software revision) is included in the response.

Transmit the I^2C address of the power supply followed by the two data bytes shown below:

- data[0] = 0x21 (command)
- data[1] = 0xDF (checksum for 0x21)

The power supply will respond with a 64-byte message. The breakdown of the 64-byte response is shown in Table 16 on the next sheet. Even though this command (21h) can be used to continuously monitor the power supply, the other commands detailed on pages 10 – 12 and 15– 16 should be more direct and useful.

Table 16
Composite Sensor Read command (21h)

Response Byte #	Data Type	Description
0	Completion Code	Echo the command (21h)
1	Status Register 0, MS bit	Refer to Table below
2 – 3	Signed Integer, MSB 1st	Hottest Rail Temperature in °C (Reading/16384 • 100°C)
4 – 5	Unsigned Integer, MSB 1st	Voltage on VS1 ((Reading/16384 • 12V)
6 – 7		Voltage on VS2 ((Reading/16384 • 3.3V)
8 – 9		Voltage on VS3 ((Reading/16384 • 5V)
10 – 11		Voltage on +3.3V_{AUX} ((Reading/16384 • 3.3V)
12 – 13		Voltage on +12V_{AUX} ((Reading/16384 • 12V)
14 – 15		Absolute Voltage on –12V_{AUX} ((Reading/16384 • 12V)
16 – 17		Current on VS1 ((Reading/16384 • 30A)
18 – 19		Current on VS2 ((Reading/16384 • 20A)
20 – 21		Current on VS3 ((Reading/16384 • 40A)
22 – 23		Current on +3.3V_{AUX} ((Reading/16384 • 4A)
24 – 25		Current on +12V_{AUX} ((Reading/16384 • 1A)
26 – 27		Current on –12V_{AUX} ((Reading/16384 • 1A)
28 – 29		Internal Reference ((Reading/16384 • 2.5V)
30 – 31	Reserved	Reserved
32 – 51	Character String	Part Number
52 – 53	Unsigned Integer, MSB 1st	S/N High
54 – 55		S/N Low
56 – 57		Date Code (Year/Week)
58 – 59		Hardware Rev
60 – 61		Firmware Rev
62	Reserved	Reserved
63	Zero checksum	The value required to make the sum of the response bytes add to a multiple of 256 (decimal).

Composite Sensor Read Command (0x21) (Cont.)

Table 17
Byte #1 (Status Reg 0) in Composite Sensor readings

Status Reg 0	N/A	R/Set	R/Set	R/W	R/W	R/W	R	R
Bit	7	6	5	4	3	2	1	0
Reading	x	FAIL	OTWarning	SWpriority	*SW Inh	*SW En	*HW Inh	*HW En

Bits 5 and 6 (OTWarning and FAIL) are Read and Write. They are clear at start up. The user can set them with a Status Write Command. Hardware will clear them if there is a fault.

Bit 4 (SWPriority) is Read and Write. It is clear at start up. When clear the unit will be controlled by the hardware ENABLE and INHIBIT signals. When set, the unit will be controlled by the SW En and SW Inh signals.

Bits 2 and 3 (SW En and SW Inh) are read and write. Their logics works the same as the logic for the discrete hardware control line (see page 2).

Table 18
Bits 0 and 1 for ENABLE and INHIBIT (read only)

SW En	SW Inh	Outputs
0	0	+3.3V_{AUX} is ON, the other five outputs are OFF
0	1	All six of the outputs are ON
1	0	All six of the outputs are OFF
1	1	All six of the outputs are OFF

Bits 0 and 1 (HW En and HW Inh) are read only. If the SWPriority is low, the state of the "ENABLE" and "INHIBIT" are shown.

The defaults used for bits 2 – 7, and bits 0 & 1 allows the user to monitor the status of the discrete hardware control line at the input connector.

Status Register 0 Command (0x55)

Table 19
Write command for Status Register 0 (0x55)

Byte #	Data Type	Description
0	U Character – 55h	Command
1	U Character	Data
2	Zero Checksum	The value required to make the sum of the response bytes 0 and 1 add to a multiple of 256 (decimal).

Example
To send a command to clear the faults and turn ON all of the outputs, the following data byte sequence must be sent:

55h (command)
78h (data for bits 01111000)
33h (checksum)

Note: If the input power is cycled, 'Status Register 0' with return to its default settings.

Status Register 0 Command (0x55)

Table 20
Read commands (0x44 and 0x45)

Response Byte #	Data Type	Description
Read Firmware Release Date (0x44)		
0	Completion code – 44h	Echo of the command
1 – 20	Character	Date
21	Zero Checksum	The value required to make the sum of the response bytes 0 to 20 add to a multiple of 256 (decimal).
Read Hardware Address Date (0x45)		
0	Completion code – 45h	Echo of the command
1	U Character	I^2C Hardware Address
2	Zero Checksum	The value required to make the sum of the response bytes 0 and 1 add to a multiple of 256 (decimal).

Special I²C Commands & Reading (Cont.)

Polling Unit Output Voltages and Currents

These commands can be used to quickly poll the unit for the output voltages and currents through the I²C port. Two bytes are used for each of the readings, meaning the resolution is considerably better than the one-byte IPMI reading.

Read the six output voltages

Transmit the I²C address of the power supply followed by the two data bytes shown below:

- data[0] = 90h (command)
- data[1] = 70h (checksum for data byte 0)

The power supply will respond with a 16-byte message:

- dataR[0] = 0x90 (command)
- dataR[1] MSB & LSB dataR[2] +12V with scale in mV
- dataR[3] MSB & LSB dataR[4] +3.3V with scale in mV
- dataR[5] MSB & LSB dataR[6] +5V with scale in mV
- dataR[7] MSB & LSB dataR[8] +3.3V AUX with scale in mV
- dataR[9] MSB & LSB dataR[10] +12V AUX with scale in mV
- dataR[11] MSB & LSB dataR[12] –12V AUX with scale in mV
- dataR[13] MSB & LSB dataR[14] Input voltage not available at this time, transmits zero
- dataR[15] = checksum for dataR bytes 0 to 14

Figure 2
Example reading of output voltages [b]

Reading:

- dataR[1] reads 0x2E
- dataR[2] reads 0xD2
- 2ED2 hex to decimal is = 11986mV

12V voltage measurement:

$$11986mV \bullet \left(\frac{1V}{1000mV}\right) = 11.986V$$

[b] The 1mV resolution for this method is considerably better than IPMI method.

Read the three main output currents

Transmit the I^2C address of the power supply followed by the two data bytes shown below:

- data[0] = 99h (command)
- data[1] = 67h (checksum for data byte 0)

The power supply will respond with a ten-byte message:

- dataR[0] = 0x99
- dataR[1] MSB & dataR[2] LSB +12V current with scale in mA
- dataR[3] MSB & dataR[4] LSB +3.3V current with scale in mA
- dataR[5] MSB & dataR[6] LSB +5V current with scale in mA
- dataR[7] MSB & dataR[8] LSB Input current not available at this time, transmits zero
- dataR[9] = checksum for dataR bytes 0 to 8

Figure 3
Example reading of main output currents [c]

Reading:

- dataR[1] reads 0x07
- dataR[2] reads 0xC6
- 4FAB hex to decimal is = 1990mA

12V current measurement:

$$1990mA \bullet \left(\frac{1A}{1000mA}\right) = 1.99A$$

Read the three AUX output currents

Transmit the I^2C address of the power supply followed by the two data bytes shown below:

- data[0] = 91h (command)
- data[1] = 6Fh (The checksum)

The power supply will respond with an eight-byte message:

- dataR[0] = 0x91
- dataR[1] MSB & dataR[2] LSB +3.3V_{AUX} current with scale in mA
- dataR[3] MSB & dataR[4] LSB +12V_{AUX} current with scale in mA
- dataR[5] MSB & dataR[6] LSB –12V_{AUX} current with scale in mA
- dataR[7] = checksum for dataR bytes 0 to 6

Figure 4
Example reading of AUX output currents [c]

Reading:

- dataR[1] reads 0x08
- dataR[2] reads 0x2F
- 082F hex to decimal is = 2095mA

+3.3V_{AUX} current measurement:

$$2095mA \bullet \left(\frac{1A}{1000mA}\right) = 2.095A$$

[c] The 1mA resolution for this method is considerably better than IPMI method.

Read the two rail temperatures

Transmit the I^2C address of the power supply followed by the two data bytes shown below:

- data[0] = 92h (command)
- data[1] = 6Eh (checksum for data byte 0)

The power supply will respond with a six-byte message:

- dataR[0] = 0x92
- dataR[1] MSB & dataR[2] LSB Left Rail Temperature with scale in 0.1°C
- dataR[3] MSB & dataR[4] LSB Right Rail Temperature with scale in 0.1°C
- dataR[5] = checksum for dataR bytes 0 to 4

Figure 5
Example reading of rail temperatures [d]

Reading:

- dataR[1] reads 0x01
- dataR[2] reads 0xB5
- 01B5 hex to decimal is = 437

Left Rail Temperature measurement: 43.7°C

[d] The 0.1°C resolution for this method is considerably better than IPMI method.

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