IPMI Firmware User Guide

for the

CP6001

Board

kontron

1. Copyright

Copyright © 2008 Kontron AG

Kontron Modular Computers makes no representations or warranties with respect to the contents or use of this manual, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose.

Kontron Modular Computers makes no representations or warranties with respect to this embedded Linux package, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose.

Permission is granted to make and distribute verbatim copies of this manual provided that the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this documentation under the conditions for verbatim copying, provided also that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this documentation into another language, under the above conditions for modified versions.

The PICMG® and CompactPCI® names and the PICMG®, CompactPCI®, ATCA®, and AdvancedTCA® logos are registered trademarks and AdvancedMC is a trademark of the PCI Industrial Computer Manufacturers Group.

Intel is a registered trademark of Intel Corporation.

I^{2}C is a trademark of Phillips Semiconductors.

Linux is a registered trademark of Linus Torvalds.

All other trademarks, registered trademarks, and trade names are the property of their respective owners.

2.   Revision History

Manual/Product Title:	IPMI Firmware User Guide for the CP6001 Module	
Manual ID Number:	37494	
Revision Index	**Brief Description of Changes**	**Date of Issue**
1.0	Initial Issue	21-Jan-2008

Imprint

Kontron Modular Computers GmbH may be contacted via the following:

MAILING ADDRESS

Kontron Modular Computers GmbH

Sudetenstraße 7

D - 87600 Kaufbeuren Germany

TELEPHONE AND E-MAIL

+49 (0) 800-SALESKONTRON

sales@kontron.com

For further information about other Kontron products, please visit our Internet web site: www.kontron.com

Disclaimer

3. Contents

4. Introduction

4.1 Acronym Definitions

BMC Baseboard Management Controller
In a compact CPCI chassis, there can be only one BMC present. The BMC administrates the SEL and the SDRR for the complete system. The BMC is connected to the other boards in the shelf via a dedicated bus (IPMB-0). The CP6001 management controller can be set in SMC mode and in BMC mode by an IPMI OEM command. The factory setting is SMC.

BSP Board Support Package

FRU Field Replaceable Unit
Every board is a FRU. The FRU data contains information about the board such as the part number and the serial number. See PICMG Specification 2.9 for complete details on the FRU data structure. The free Linux tool 'ipmitool' [1] can be used to update or to display the FRU data.

FWH Firmware Hub. Memory location where a complete BIOS code is stored.

I²C Inter-Integrated Circuit

IPMB Intelligent Platform Management Bus
The dedicated I²C management bus where the BMC and the SMCs communicate.

IPMB-0 Intelligent Platform Management Bus which connects all SMCs with the BMC or a Shelf Manager.

IPMI Intelligent Platform Management Interface.

KCS Keyboard Controller Style (Interface)
This is the IPMI mandatory interface on the host system (payload) to communicate with the BMC.

MP Management Power.
This powers the BMC's or SMC's controller and is available as soon as the board is inserted. The Handle switch needs not be closed for that.

PICMG PCI Industrial Computer Manufacturer Group

PWR Payload Power.
This powers the host side of the board where the application software runs. It is granted by the BMC or SMC after all prerequisites are met. prerequisites are a closed handle switch, power on the backplane etc.

SDR Sensor Data Record
This is the IPMI data structure that defines a sensor.

SDRR Sensor Data Record Repository
Is the device in the BMC where all SDRs of the chassis' boards are administrated. A free Linux utility named 'ipmitool' [1] makes a full chassis discovery and fills the SDRR with the SDRs being found. The factory default repository contains only the local board's SDRs. TODO refer to ipmitool

SEL System Event Log
Is the device in the BMC where all the events in the chassis which are reported are administrated. If an event occurs on any board, the sensor event is sent throught the IPMB bus to the BMC which additionally stores its own events as well.

SMBIOS System Management BIOS

SMS System Management Software (designed to run under the OS)

SMC   Satellite Management Controller
In a compact PCI chassis, there can be many SMCs. Each SMC is connected to the BMC via a dedicated bus (IPMB-0). The CP6001 management controller can be set in SMC mode and in BMC mode by an IPMI OEM command. The factory setting is SMC.

4.2 Related Documentation

IPMI specifications: (http://www.intel.com/design/servers/ipmi/spec.htm)

- IPMI-Intelligent Platform Management Interface Specification v1.5 Document revision 1.1, February 2002

- Addenda, Errata, and Clarifications document revision 4 for IPMI v1.5 rev 1.1 specification

- IPMI- Intelligent Platform Management Bus Communications Protocol Specification v1.0 Document Revision 1.0, November 1999

- IPMI- IPMB v1.0 Address Allocation Document Revision 1.0, September 1998

- [4] IPMI- Platform Management FRU Information Storage Definition v1.0 Document Revision 1.1, September 1999

PICMG specifications: http://www.picmg.org

- PICMG 2.9 R1.0 CompactPCI System Management Specification, February 2000

Open tools documentation

- [1] ipmitool documentation: http://ipmitool.sourceforge.net. Refer as well to "12, OS Support / Tools"

- [2] OpenIPMI documentation: http://www.openipmi.sourceforge.net . Refer as well to "12, OS Support / Tools"

Kontron manuals and specifications: http://www.kontron.com/

- CP6001 User's Guide
- [3] CP6001 Linux Board Support Package

 ID 37494, Rev. 1.0

4.3 Product Overview

This product fully supports Intelligent Platform Management Interface 1.5 (IPMIv1.5) and PICMG 2.9 R1.0 specifications. All its functionalities run under an autonomous management controller even if the board is held in reset or power down by a management card within a system designed for High Availability such as XL-VHDS or XL-LP42.

While the CP6001 IPMI implementation is fully compliant to IPMI v1.5 and should work with any System Management Software that respects this specification, it has been designed to be easily integrated with the Service Availability Forum-Hardware Platform Interface (SAF-HPI) specification.

You can find more information about the IPMI at the following Web site:

http://www.saforum.org/home

IPMI is an extensible and open standard that defines autonomous system monitoring. It is autonomous because every management controller within a compact PCI chassis monitors its own sensors and sends critical events through a dedicated bus to a Baseboard Management Controller (BMC) that logs it into a non volatile System Event Log (SEL). The CP6001 IPMI implementation includes a device SDR repository module that allows the user's System Management Software (SMS) to discover all system's components and to build a database of all management controller sensors.

You can find more information about the IPMI at the following Web site:

http://www.intel.com/design/servers/ipmi/

5. IPMI Setup

5.1   IPMI in a Compact PCI Chassis

Kontron's IPMI implementation in the cPCI environment is compliant to the PCMIG 2.9 R1.0 specification. The specification defines the pinout of J1 and J2 as well as the addressing scheme. There should be only one BMC in the chassis, or at least on the IPMB segment. The BMC may reside either on an SBC blade or on an external system management card (SMC) or in a shelf management controller (ShMC). The specification allows all of these variants.
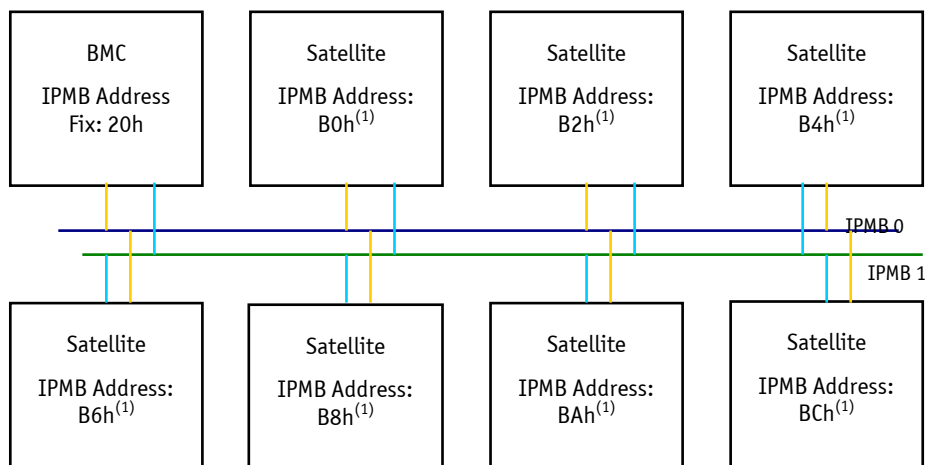
(1) IPMB address for satellte is determined via the location of the slot in the chassis.

To use the IPMI resources in a rack, some steps are needed. The system operator must perform the following steps.

5.2   IPMI Setup for the CP6001

First of all the IPMI Management Controller of the CP6001 has to be configured. For the first time please use the BIOS start up menu (Select Advanced->IPMI 1.5 Configuration).

- In line 'Status of IPMC' the BMC/SMC IPMI controller should be announced as 'Working'. Otherwise the controller is not programmed or not working and a setup is not possible.

- In line 'Management Controller Config' you may select the function of the CP6001's IPMI controller you need i.e. set it to BMC ('Baseboard') or SMC ('Satellite').

- In line 'KCS-SMS IRQ' you may select whether IRQ10, IRQ11 or no interrupt shall be used on payload side for the communication between payload and BMC or SMC.

The default factory setting of a CP6001 is "SMC, no interrupts". If it is that what you want you need not enter the BIOS setup. When storing the configuration, the BIOS creates an 'IPMI Device Information Record (Type 38h)' entry in the SMBIOS table. This record contains information about:

- Supported IPMI specification revision (v1.5)

- type of the supported interface (KCS style)

- chosen interrupt (10, 11 or none).

This information is needed by the IPMI OS kernel drivers for Linux during their loading time. After the loading most available IPMI communications tools which access the IPMI controller via IPMI OS drivers should work (e.g. ipmicmd, ipmitool [1] etc.). Now it's possible to use such a tool to issue the "Set Firmware Parameters" OEM IPMI command (refer to 8.2.2, Set Firmware Parameters) to modify the configuration again. But the change of the interrupt number always needs a BIOS restart for a correct set up of the SMBIOS table.

5.3   IPMI Setup for the rack

For a working IPMI configuration the SDRR of the BMC has to be filled with all sensors data records of all IPMI controllers in the rack. After every system start the BMC uses the SDRR to initialize all sensors of all boards. The SDRR setup must be done by a management tool e.g. the open Linux tool 'ipmitool' [1]. The command then is

```
ipmitool sdr fill sensors
```

This will only work if the IPMI controller of the BMC is addressed. This addressing is the default if you use ipmitool on the payload side of the board where the BMC is residing.

6. Management Controller Hardware

On the CP6001 module, the Management Controller is implemented using the Renesas H8S/2166 controller with 512 kB of internal Flash and 40 kB of RAM. An additional 1 MB serial EEPROM chip provides redundant firmware image storage. A separate 32 kB serial EEPROM chip is used for firmware private data and 4 kB EEPROM serves as FRU Inventory storage.

The Management Controller circuit implements two local Keyboard Style Interfaces (KCS) with interrupt support for communication with system side management software and BIOS. The Management Controllers in the rack are all connected by the IPMB-0 bus.

The Management Controller implements a wide range of sensors that permit the monitoring of:

- main power voltages: 5V (PWR), 3.3V, 5V (MP), 1.5V, 1.8V, 0.9V (DDR) ,

- temperatures: CPU die temperature, 2 board temperatures

- Power Good, IPMB-0 link, board reset, post code, boot error, processor state, Health error, IPMI watchdog etc.

7.   Management Controller Firmware

7.1   Key Features

- Compliant with IPMI specification 1.5, revision 1.1

- Compliant with PICMG 2.9 specification

- Firmware designed and specially made for compact PCI implementation and easy integration with SAF-HPI

- KCS SMS interface with interrupt support

- Dual Port IPMB configurable as two independent channels or in redundant mode (refer to 8.2.2, Set Firmware Parameters).

- Out of band management and monitoring using IPMB interface permits access to sensors regardless of the board's CPU state

- Sensor thresholds fully configurable

- Sensor names prefixed with identification of owner (BMC without slot number or SMC with slot number)

- Complete IPMI watchdog functionality

- Complete SEL, SDR repository and FRU functionality on BMC

- Complete FRU functionality

- Master Write-Read I2C supports for external I2C devices communications (FRU, EEPROM, FAN).

- Two IPMI controller firmware banks implemented to allow an automatic backup. This allows manual and automatic firmware image roll-back (in case of upgrade failure).

- The downloading of a new firmware image does not break currently running firmware or payload activities (refer to 7.2.2, Firmware Upgrade).

- Firmware bank management is done by the open tool ipmitool [1] (function fwum) which can update the firmware in the field.

- Firmware fully customizable to the customer's needs by OEM IPMI commands (refer to 8.2.2, Set Firmware Parameters).

- FRU data can be updated in the field by the open tool ipmitool [1] (function fru write)

- Interoperable with other IPMI solutions

- OEM board supervision and control extensions such as boot device flash selection (refer to 7.2.4.2, Boot Flash Selection by OEM IPMI Command) and firmware boot order configuration (refer to 7.2.4.4, Boot Order selection by OEM IPMI Command).

- Automatic switching to an alternative BIOS image after having detected a not working BIOS (refer to 7.2.4.3, Automatic Boot Flash Selection during the Boot Process).

- Graceful shutdown support (refer to 7.2.5., Hot Swap and Shut Down).

- Handle switch and blue Hot Swap LED are working like on an ATCA blade (refer to 7.2.5., Hot Swap and Shut Down).

- An "Out of Service" (OOS) LED shows whether the IPMI controller is working and pulses when there is IPMB-0 traffic (refer to 10., IPMI Communication LEDs).

- The "Health" LED shows the IPMI controller's heartbeat and pulses if the KCS interface is active (refer to 10., IPMI Communication LEDs).

7.2 Firmware Code

7.2.1 Structure and Functionality

The IPMI controller firmware code is organized into boot code and operational code, both of which are stored in a flash module. Upon an IPMI controller reset, the IPMI controller first executes the boot code which does:

- A self test to verify the status of the Management Controller's hardware including its memory.

- Performs a checksum of the operational code.

After successful verification of the operational code checksum, the firmware will jump to the operational code.

Only the operational code is upgradeable in-the-field.

7.2.2 Firmware Upgrade

The standard way to upgrade the IPMI controller's operational code is to use the open tool 'ipmitool' [1] together with an image file. Ipmitool allows the downloading ('ipmitool fwum download …') and activation ('ipmitool fwum upgrade') of the new operational code and saves an existing one. The rollback to the formerly running operational code is possible as well ('ipmitool fwum rollback'). The status command ('ipmitool fwum status') displays what firmware is stored and in what state it is ('last known good' = running, 'previous good' = running before upgrade).

All IPMI interfaces which are offered by ipmitool (KCS interface, IPMB bus, LAN) are usable for the upgrade. This allows local upgrade and remote upgrade. Please note that the KCS interface is only usable on a powered payload.

Files which contain an image of operational code have the board name "CP6001" and the string "FWUM" in its name.

During the download process the currently running operational code is still operating in a normal way until the upgrade command is issued. During the now starting upgrade process the IPMI controller is off line for about 20 seconds when the boot code is re-organizing the

firmware storage. Afterwards the new operational code is started. If the new operational code doesn't operate well, e.g. hangs, the boot code will perform an automatic rollback to the last working operational code and start this again.

7.2.3 Firmware / Module Identification

There are two ways to verify by means of IPMI that the Management Controller resides on a CP6001.

The response on the IPMI command "Get Device ID" offers among others the following response data:

- Manufacturer ID = 3A98h (Kontron IANA ID)

- Device ID = 04h (H8S2166)

- Product ID = 6001 which means that the board is a CP6001

- Firmware Revision in bytes 4:5 - depends on the core version of the running firmware.

- The SDR revision in byte 13 (OEM part of the response) is a sub revision of the firmware revision. It is unique for all versions of the board's firmware i.e. the Firmware Revision mentioned above is not really needed for the identification of the firmware.

- The Device ID String which can be found by reading the Management Controller Device Locator Record (SDR Type 12h) contains the string "BMC:CP6001". For e.g. the ipmitool [1] command 'ipmitool sdr list mcloc' the Device ID Strings of all available boards will be displayed. If the CP6001 is a BMC for it this string will be displayed without change. If the CP6001 is a SMC then the string will be changed into "Sxx: CP6001" where xx is the slot number where the board is residing, e.g. "S09: CP6001".

7.2.4 The Payload Boot Process

When the CP6001's payload starts, the first code to be executed is the BIOS. There are two Flash devices, numbered 0 and 1, which may contain different BIOS code. Which one of them will be selected from the **next** boot process on is selectable this way:

a. The contents of a user (payload) writeable register (refer to the CP6001 User Guide) tells which Boot Flash to use. This is the primary selection.

b. The firmware's parameter EEPROM contains a parameter which's value decides whether to invert the primary selection register's contents or not when the Management Controller's firmware selects the boot flash. For this the Management Controller sets or resets a control signal which inverts or inverts not the boot flash selection.

7.2.4.1 Boot Flash Selection by writing to a board register

Please refer to the CP6001 User Guide.

7.2.4.2 Boot Flash Selection by OEM IPMI Command

The OEM IPMI command "Set Control State" (refer to 8.2.3, Set Control State (Firmware Hub/BIOS Flash, Boot Order)) adjusts whether the Management Controller has to invert the register based Flash selection or not from the **next** boot process on. The Management Controller stores this decision in a parameter in the EEPROM. Please refer to 8.2.3, Set Control State (Firmware Hub/BIOS Flash, Boot Order).

7.2.4.3 Automatic Boot Flash Selection during the Boot Process

After each payload reset the Management Controller selects the boot flash by applying the related EEPROM parameter. Physically the Management Controller sets or resets a signal line. Afterwards it waits for a special message from the BIOS. This message contains the checksum report, i.e. it reports whether the boot Flash's checksum is right or wrong. If the checksum is wrong or the message is not received within 20 seconds, then the currently used BIOS Flash is assumed to contain an invalid or a corrupted image. In this case the Management Controller toggles the parameter value in the EEPROM and issues a "Boot Error (Invalid boot sector) event" by setting the appropriate sensor value (sensor 'FWHx Boot Err'. x = 0..1). x is simply the value of the parameter in EEPROM and not the absolute number of the used boot flash. Afterwards it causes a payload reset and continues as being described at the beginning of this chapter. When a timeout error is recognized and the count of boot errors exceeds 2 or when a checksum error is recognized and the count of boot errors exceeds 4 the Management Controller gives up, i.e. causes no more payload reset to stabilize the system. Not until the next payload power on event the Management Controller will care about booting.

7.2.4.4 Boot Order selection by OEM IPMI Command

Normally the BIOS will apply the boot order which was selected in the BIOS setup menu "Boot/Boot device priority". But there is another alternative boot order which is held in the Management Controller's non volatile memory. This boot order can be set and read by IPMI OEM commands (refer to 8.2.3., Set Control State (Firmware Hub/BIOS Flash, Boot Order) and 0,). At payload start the Management Controller writes it into a register where the BIOS can read it. If this Management Controller's boot order has a non zero value the BIOS will use it instead of its own boot order. Please refer to 8.2.3, Set Control State (Firmware Hub/BIOS Flash, Boot Order).

7.2.4.5 Communication between Management Controller and BIOS

For communication between BIOS and Management Controller there is a "private" KCS interface. During the boot process the BIOS sends the following IPMI commands to the Management Controller:

- An OEM command which reports a good or a bad checksum.

- A Standard IPMI command "Set Watchdog Timer" to stop a possibly running IPMI watchdog timer.

- A Standard IPMI command "Set SEL Time" to set the event log time to the time which is kept by the RTC.

- An OEM IPMI command (refer to 8.2.2, Set Firmware Parameters) with some parameters which e.g. set the Management Controller to a BMC or a SMC as selected in the BIOS menu.

- A Standard IPMI command "Set ACPI Power State" to set the state "ACPI legacy on"

- Etc.

7.2.5 Hot Swap and Shut Down

7.2.5.1 Handle Switch and Hot Swap (blue) LED

As a hot-swappable field replaceable unit (FRU), the CP6001 behaves like an ATCA blade and internally uses similar "M-states". The blue Hot Swap LED (HS LED) of an inserted board in a powered rack in general shows the board's Hot Swap state:

- On = the board is inactive and may be a) activated by closing the Handle Switch or b) may be extracted. The "M-state" is 1. An **exception** is the case when module power is off e.g. after a shut down and the handle is still closed. We have here the M-state 4. To show the operator that the power is off the blue LED will be on in spite of the closed handle.

- Blinking = changing from active state to inactive state or vice versa. Don't extract the board now. The "M-state" is 2, 5 or 6.

- Off = the board is active. Don't extract the board now. Normally the extraction is impossible because the Handle Switch is closed. The "M-state" is 3 or 4.

Normally the logical states "active" and "inactive" of a board are identical to the physical states "handle open" and "handle closed" or "payload power on" and "payload power off". But this is only true if we exclusively use the Handle Switch to select the board's state.

If we e.g. switch the power on or off using an IPMI chassis commands or we let shut down the payload by the OS then the position of the handle switch and the power state necessarily might become asynchronous. This is avoided by a special behavior of the blue LED in that case. Please refer to the description of 'on' state above.

Example 1: The handle switch is closed, the blue LED is off and power is on. If we switch off power by the IPMI chassis command the blue LED will be switched on despite the handle is still closed. The states of the LED and the position of the handle switch are not corresponding in this case but for the operator it is easier to be notified by the glowing LED that this board needs service. If we now switch on the power again by an IPMI chassis command the LED will be switched off again. The position of the handle switch again is corresponding with the power state. Another possibility is to power up the module is to open the handle and to wait until the blue LED stops blinking and stays on. The closing of the handle then powers up the module.

Example 2: The handle switch is closed, the blue LED is off and power is on. If we use an OS with ACPI support a "Shut Down" command will lead us to the state "payload power off" and the blue LED will be switched on despite the handle is still closed. The states of the LED and the position of the handle switch are not corresponding in this case but for the operator it is

easier to be notified by the glowing LED that this board needs service. To power up the board again we can proceed like in example 1 (see above).

7.2.5.2 The Hot Swap and Shut Down processes

When we open the Handle Switch of a board which's payload is running we want to switch the payload off i.e. shut the power down in a way which causes no loss of data. We aim the same if we order the operating system (OS) to perform a "Shut Down".

We must distinguish here between three scenarios. We might have on payload side …

1) …a "dumb" OS which doesn't support ACPI at all.

2) …an OS which supports ACPI.

3) …a "dumb" OS which doesn't support ACPI but behaves towards the Management Controller as if it supports ACPI. This will be treated like 2)

A "hot Swap" or a "Shut down" in a system 2) or 3) which is normally caused by an operator's command will both end in a "Graceful Shut Down".

A "Graceful Shut Down" denotes a shut down where all processes are terminated before the power is switched off.

7.2.5.2.1 Dumb system with no ACPI support

After payload power on the starting BIOS will inform the Management Controller by sending the IPMI command "Set ACPI Power State / Set Legacy on". This means that a Hot Swap (opening of a closed handle) shall immediately lead to a power off by the Management controller. The operator is responsible for the termination of processes to avoid the loss of date.

7.2.5.2.2 System having ACPI support

When after BIOS there is an OS starting which supports ACPI, this will cause the transmission of the IPMI command "Set ACPI Power State / S0/G0 working" to the Management Controller. This means that the OS has reprogrammed the chip set in a manner that a "power button" signal doesn't lead to an immediate power off but only causes an event that can be detected by the OS.

Case 1, Hot Swap:
When the handle switch is opened, the Management Controller pulls the "power button" signal to notify the OS. The OS then will shut down all processes and afterwards will cause the transmission of the IPMI command "Set ACPI Power State / S5/G2 soft off" to the Management Controller which will now switch the power off. Please note that the Management Controller will switch off the power immediately if no more that 20 seconds have elapsed between entering "S0/G0" state and Hot Swap. This is to speed up the Hot Swap while the OS has not yet started an application.

Case 2, Shut Down:
When the OS is forced by the user to perform a shut down it will shuts down all processes. Afterwards it will cause the transmission of the IPMI command "Set ACPI Power State / S5/G2 soft off" to the Management Controller which will now switch the power off.

7.2.5.2.3 System which emulates ACPI support

An OS which not really supports ACPI like e.g. VxWorks is able to get "Graceful Shut Down" support from the Management Controller if it behaves in the following way.

After start such an OS has to manipulate the chip set in a way that prevents an immediate power off on a "power button" signal.

Then it has to send the IPMI command "Set ACPI Power State / S0/G0 working" to the Management Controller to enable this to process later a "S5/G2 soft off" command.

During its run time the system shall cyclically read the "Hot Swap Sensor" (sensor #0) using the IPMI command "Get Sensor Reading". This allows the tracking of the board's state. After the board has once reached "M-state" 4 (sensor reading is 10h) the leaving of this announces that the handle switch was opened. Now the time has come to terminate all processes.

After all critical processes have been terminated the OS has to send the IPMI command "Set ACPI Power State / S5/G2 soft off" to the Management Controller which will set the power off immediately.

7.2.6 Setting of the SEL time

The Management Controller has no own hardware real time clock. Therefore after start, restart or upgrade of the Management Controller first its software clock has to be supplied with the current time. The Management Controller uses the time when handling event messages which otherwise will have an out-of-date time stamp.

Every time when the BIOS comes up it supplies the Management Controller with the payload's current real time clock time. A problem is a re-start of the Management Controller without a following BIOS start. Because during re-start the Management Controller's time gets lost it must be set again by issuing the IPMI command "Set SEL Time". This may be done by application software on the payload side via the KCS interface or by a remote Management Controller via the IPMB-0.

Ipmitool [1] offers a command for setting the SEL time as well. e.g.

```
ipmitool sel time set "12/20/2007 10:59:51"
```

8. Supported IPMI Commands

8.1 Standard Commands

Part of the command list in IPMI specification 2.0

M = mandatory, O = optional

	IPMI 2.0 Spec. section	NetFn	CMD	Kontron support On MMC
IPM Device "Global" Commands				**M**
Get Device ID	20.1	App	01h	M / **Yes [1]**
Cold Reset	20.2	App	02h	O / **Yes**
Warm Reset	20.3	App	03h	O / No
Get Self Test Results	20.4	App	04h	O / **Yes**
Manufacturing Test On	20.5	App	05h	O / No
Set ACPI Power State	20.6	App	06h	O / No
Get ACPI Power State	20.7	App	07h	O / No
Get Device GUID	20.8	App	08h	O / No
Broadcast "Get Device ID"	20.9	App	01h	M / **Yes**
BMC Watchdog Timer Commands				**O**
Reset Watchdog Timer	27.5	App	22h	O / **Yes**
Set Watchdog Timer	27.6	App	24h	O / **Yes**
Get Watchdog Timer	27.7	App	25h	O / **Yes**

BMC Device and Messaging Commands				O
Set BMC Global Enables	22.1	App	2Eh	O / **Yes**
Get BMC Global Enables	22.2	App	2Fh	O / **Yes**
Clear Message Flags	22.3	App	30h	O / **Yes**
Get Message Flags	22.4	App	31h	O / **Yes**
Enable Message Channel Receive	22.5	App	32h	O / **Yes**
Get Message	22.6	App	33h	O / **Yes**
Send Message	22.7	App	34h	O / **Yes**
Read Event Message Buffer	22.8	App	35h	O / **Yes**
Get BT Interface Capabilities	22.9	App	36h	O / **Yes**
Get System GUID	22.14	App	37h	O / No
Get Channel Authentication Capabilities	22.13	App	38h	O / No
Get Session Challenge	22.15	App	39h	O / No
Activate Session	22.17	App	3Ah	O / No
Set Session Privilege Level	22.18	App	3Bh	O / No
Close Session	22.19	App	3Ch	O / No
Get Session Info	22.20	App	3Dh	O / No
Get AuthCode	22.21	App	3Fh	O / No
Set Channel Access	22.22	App	40h	O / No
Get Channel Access	22.23	App	41h	O / No
Get Channel Info	22.24	App	42h	O / No
Set User Access	22.26	App	43h	O / No
Get User Access	22.27	App	44h	O / No
Set User Name	22.28	App	45h	O / No
Get User Name	22.29	App	46h	O / No
Set User Password	22.30	App	47h	O / No
Activate Payload	24.1	App	48h	O / No
Deactivate Payload	24.2	App	49h	O / No
Get Payload Activation Status	24.4	App	4Ah	O / No
Get Payload Instance Info	24.5	App	4Bh	O / No
Set User Payload Access	24.6	App	4Ch	O / No
Get User Payload Access	24.7	App	4Dh	O / No
Get Channel Payload Support	24.8	App	4Eh	O / No
Get Channel Payload Version	24.9	App	4Fh	O / No
Get Channel OEM Payload Info	24.10	App	50h	O / No
Master Write-Read	22.11	App	52h	O / **Yes**
Get Channel Cipher Suits	22.15	App	54h	O / No
Suspend/Resume Payload Encryption	24.3	App	55h	O / No
Set Channel Security Keys	22.25	App	56h	O / No
Get System Interface Capabilities	22.9	App	57h	O / No

Chassis Device Commands				O / Yes
Get Chassis Capabilities	28.1	Chassis	00h	O / **Yes**
Get Chassis Status	28.2	Chassis	01h	O / **Yes**
Chassis Control	28.3	Chassis	02h	O / No
Chassis Reset	28.4	Chassis	03h	O / No
Chassis Identify	28.5	Chassis	04h	O / No
Set Chassis Capabilities	28.7	Chassis	05h	O / No
Set Power Restore Policy	28.8	Chassis	06h	O / No
Get System Restart Cause	28.11	Chassis	07h	O / No
Set System Boot Options	28.12	Chassis	08h	O / No
Get System Boot Options	28.13	Chassis	09h	O / No
Get POH Counter	28.14	Chassis	0Fh	O / **Yes** **[2]**
Event Commands				**M**
Set Event Receiver	29.1	S/E	01h	M / **Yes**
Get Event Receiver	29.2	S/E	02h	M / **Yes**
Platform Event (a.k.a. "Event Message")	29.3	S/E	03h	M / **Yes**
PEF and Alerting Commands				**O**
Get PEF Capabilities	30.1	S/E	10h	O / No
Arm PEF Postpone Timer	30.2	S/E	11h	O / No
Set PEF Configuration Parameters	30.3	S/E	12h	O / No
Get PEF Configuration Parameters	30.4	S/E	13h	O / No
Set Last Processed Event ID	30.5	S/E	14h	O / No
Get Last Processed Event ID	30.6	S/E	15h	O / No
Alert Immediate	30.7	S/E	16h	O / No
PET Acknowledge	30.8	S/E	17h	O / No

Sensor Device Commands				M
Get Device SDR Info	35.2	S/E	20h	M / **Yes**
Get Device SDR	35.3	S/E	21h	M / **Yes**
Reserve Device SDR Repository	35.4	S/E	22h	M / **Yes**
Get Sensor Reading Factors	35.5	S/E	23h	O / No
Set Sensor Hysteresis	35.6	S/E	24h	O / **Yes**
Get Sensor Hysteresis	35.7	S/E	25h	O / **Yes**
Set Sensor Threshold	35.8	S/E	26h	O / **Yes**
Get Sensor Threshold	35.9	S/E	27h	O / **Yes**
Set Sensor Event Enable	35.10	S/E	28h	O / **Yes**
Get Sensor Event Enable	35.11	S/E	29h	O / **Yes**
Re-arm Sensor Events	35.12	S/E	2Ah	O / No
Get Sensor Event Status	35.13	S/E	2Bh	O / No
Get Sensor Reading	35.14	S/E	2Dh	M / **Yes**
Set Sensor Type	35.15	S/E	2Eh	O / No
Get Sensor Type	35.16	S/E	2Fh	O / No
FRU Device Commands				M
Get FRU Inventory Area Info	34.1	Storage	10h	M / **Yes**
Read FRU Data	34.2	Storage	11h	M / **Yes**
Write FRU Data	34.3	Storage	12h	M / **Yes**
SDR Device Commands				O
Get SDR Repository Info	33.9	Storage	20h	O / **Yes**
Get SDR Repository Allocation Info	33.10	Storage	21h	O / **Yes**
Reserve SDR Repository	33.11	Storage	22h	O / **Yes**
Get SDR	33.12	Storage	23h	O / **Yes**
Add SDR	33.13	Storage	24h	O / **Yes**
Partial Add SDR	33.14	Storage	25h	O / **Yes**
Delete SDR	33.15	Storage	26h	O / **Yes**
Clear SDR Repository	33.16	Storage	27h	O / **Yes**
Get SDR Repository Time	33.17	Storage	28h	O / No
Set SDR Repository Time	33.18	Storage	29h	O / No
Enter SDR Repository Update Mode	33.19	Storage	2Ah	O / No
Exit SDR Repository Update Mode	33.20	Storage	2Bh	O / No
Run Initialization Agent	33.21	Storage	2Ch	O / **Yes**

SEL Device Commands				O
Get SEL Info	40.2	Storage	40h	O / **Yes**
Get SEL Allocation Info	40.3	Storage	41h	O / **Yes**
Reserve SEL	40.4	Storage	42h	O / **Yes**
Get SEL Entry	40.5	Storage	43h	O / **Yes**
Add SEL Entry	40.6	Storage	44h	O / **Yes**
Partial Add SEL Entry	40.7	Storage	45h	O / No
Delete SEL Entry	40.8	Storage	46h	O / **Yes**
Clear SEL	40.9	Storage	47h	O / **Yes**
Get SEL Time	40.10	Storage	48h	O / **Yes**
Set SEL Time	40.11	Storage	49h	O / **Yes**
Get Auxiliary Log Status	40.12	Storage	5Ah	O / No
Set Auxiliary Log Status	40.13	Storage	5Bh	O / No

AdvancedTCA®[10]	PICMG® 3.0 Table			M
Get PICMG Properties	3-9	PICMG	00h	M / **Yes**
Get Address Info	3-8	PICMG	01h	N/A
Get Shelf Address Info	3-13	PICMG	02h	N/A
Set Shelf Address Info	3-14	PICMG	03h	N/A
FRU Control	3-22	PICMG	04h	M / **Yes [3]**
Get FRU LED Properties	3-24	PICMG	05h	M / **Yes**
Get LED Color Capabilities	3-25	PICMG	06h	M / **Yes**
Set FRU LED State	3-26	PICMG	07h	M / **Yes**
Get FRU LED State	3-27	PICMG	08h	M / **Yes**
Set IPMB State	3-51	PICMG	09h	N/A
Set FRU Activation Policy	3-17	PICMG	0Ah	N/A
Get FRU Activation Policy	3-18	PICMG	0Bh	N/A
Set FRU Activation	3-16	PICMG	0Ch	N/A
Get Device Locator Record ID	3-29	PICMG	0Dh	M / **Yes**
Set Port State	3-41	PICMG	0Eh	N/A
Get Port State	3-42	PICMG	0Fh	N/A
Compute Power Properties	3-60	PICMG	10h	N/A
Set Power Level	3-62	PICMG	11h	N/A
Get Power Level	3-61	PICMG	12h	N/A
Renegotiate Power	3-66	PICMG	13h	N/A
Get Fan Speed Properties	3-63	PICMG	14h	N/A
Set Fan Level	3-65	PICMG	15h	N/A
Get Fan Level	3-64	PICMG	16h	N/A
Bused Resource	3-44	PICMG	17h	N/A
Get IPMB Link Info	3-49	PICMG	18h	N/A

[1] Has oem extensions. Please refer to 8.2.1, Get Device ID Command with OEM Extensions

[2] Response byte 2: hours, byte 3: minutes after module start. Bytes 4..6: void

[3] Only 1 = Cold Reset and 2 = Warm Reset

8.2 OEM Commands and Extensions

8.2.1 Get Device ID Command with OEM Extensions

	LUN	NetFn	CMD
Get Device ID command with OEM extensions	00h	App = 06h	01h

	Byte	Data Field
Request Data	-	-
Response Data	1	Completion Code
	2:12	Regular Get Device ID Command response fields
	13	Release number 1… of the IPMI controller firmware. The open ipmi tool ' ipmitool' [1] displays this as 'SDR' in the answer on command 'ipmitool fwum status'.
	14	Module Geographical Address (slot number): 1 … = Module in chassis slot 1…
	15	Reserved
	16	Reserved

8.2.2 Set Firmware Parameters

The command below permits the selection of interrupts to be used during KCS communication.

	LUN	NetFn	CMD
Set Firmware Parameters	03h	OEM = 3Eh	05h

	Byte	Data Field
Request data	1	Reserved B4h
	2	Reserved 90h
	3	Reserved 91h
	4	Reserved 8Bh

	5	Cmd Flags
		[6:2] Reserved
		[1] 0b = get only, 1b = set parameters
		[0] 0b = do not reset,
		    1b = reset Management Controller after setting
		        parameters

	6	Operating Modes [7:5] Reserved [4] 0b = IPMB in redundancy [3:1] Reserved [0] 0b = BMC, 1b = SMC
	7	IRQ number FFh = do not use interrupts 0Ah = use IRQ10 0Bh = use IRQ11 Any other values Reserved.
Response data	1	Completion code
	2	Cmd Flags
	3	Operating Modes
	4	IRQ number

8.2.3 Set Control State (Firmware Hub/BIOS Flash, Boot Order)

	LUN	NetFn	CMD
Set Control State (Firmware Hub/BIOS Flash, Boot Order)	00h	OEM = 3Eh	20h

	Byte	Data Field
Request data	1	Control ID 00h: BIOS Flash selection 9Dh: BIOS Boot Order Configuration
	2	Control State for BIOS Flash selection: (These settings are stored in EEPROM and applied (to logic) each time the IPMI controller detects power-on) 00h = BIOS Flash selection is not inverted 01h = BIOS Flash selection is logically inverted Please note that this selection will be automatically toggled by the IPMI controller during a failing boot process. Other payload sided settings may additionally modify this selection. Control State for BIOS Boot Order Configuration: 00h .. 07h = Selected BIOS Boot Order Configuration. 00h selects the default Boot Order which is selected in the BIOS menu. For values 01h..07h refer to the CP6001 user's guide, register "IPMI Boot Configuration Register". (These settings are stored in EEPROM and applied (to logic) each time the IPMI controller detects power-on)
Response data	1	Completion code

8.2.4 Get Control State (Firmware Hub/BIOS Flash, Boot Order)

	LUN	NetFn	CMD
Get Control State (Firmware Hub/BIOS Flash, Boot Order)	00h	OEM = 3Eh	21h

	Byte	Data Field
Request data	1	Control ID 00h: BIOS Flash selection 9Dh: BIOS Boot Order Configuration
Response data	1	Completion code
	4	Current Control State (refer to 8.2.3, Set Control State (Firmware Hub/BIOS Flash, Boot Order)

9. Board Sensors

The Management Controller includes many sensors for voltage or temperature monitoring and various others for pass/fail type signal monitoring.

Every sensor is associated with a Sensor Data Record (SDR). Sensor Data Records contain information about the sensors identification such as sensor type, sensor name, sensor unit. SDRs also contain the configuration of a specific sensor such as thresholds, hysteresis, event generation capabilities, etc. that specify the sensor's behavior. Some fields of the sensor SDR are configurable through IPMI v1.5 command and are set to a built-in initial value.

Module sensors that have been implemented are listed in the sensor list below.

9.1 Sensor List

The sensor name (ID string) has a name prefix which is 'NNN:' in the lists below. When reading the sensor name after module insertion this prefix becomes automatically adapted to the role (BMC or SMC) and the physical position (slot number) of the module in a rack. If the module's Management Controller is set up as a BMC the prefix will be 'BMC:' independent of the slot where it resides. If the module's Management Controller is set up as a SMC the prefix will be 'Sxx:' where xx is the slot number (e.g. 09).

The sensor number is the number which identifies the sensor e.g. when using the IPMI command "Get Sensor Reading". Please not that 'ipmitool' [1] accepts sensor numbers in decimal (e.g. '10') or hexadecimal (e.g. '0xa') notation.

Please note that the IPMI tool 'ipmitool' displays for command 'ipmitool sdr list' the contents of the sensor data record repository (SDRR) of the whole rack if the SDRR is generated. The generation of the SDRR has always to be done new after adding or subtracting any board to or from the rack. Refer to 5.3, *IPMI Setup for the rack*.

For OEM (Kontron) specific sensor types and reading types in the following table please refer to the next chapter.

SENSOR Number / ID string	SENSOR TYPE (CODE) / EVENT/READING TYPE (CODE)	Ass. Mask / Deass. Mask / Reading Mask	DESCRIPTION	Causes red Health LED on error [*)] / Reading Mask
0 / NNN:Hot Swap	ATCA/CTCA Hot Swap (F0h) / Sensor-specific (6Fh)	00FFh / 0000h / 00FFh	Hot swap sensor	N
1 / NNN:Temp CPU	Temperature (01h) / Threshold (01h)	7A95h / 7A95h / 3F3F	CPU die temperature	Y / 0F3Ch
2 / NNN:Temp Board 1	Temperature (01h) / Threshold (01h)	7A95h / 7A95h / 3F3F	Temp Board 1	Y / 0F3Ch
3 / Temp Board 2	Temperature (01h) / Threshold (01h)	7A95h / 7A95h /	Temp Board 2	Y / 0F3Ch

SENSOR Number / ID string	SENSOR TYPE (CODE) / EVENT/READING TYPE (CODE)	Ass. Mask / Deass. Mask / Reading Mask	DESCRIPTION	Causes red Health LED on error *) / Reading Mask
		3F3F		
4 / Pwr Good	Power supply (08h) / OEM (73h)	0000h / 0000h / 402Fh	Status of all power lines	N
5 / NNN:Pwr Good Evt	Power supply (08h) / OEM (73h)	0000h / 18B7h / 402Fh	Power fail events for all power lines	Y
6 / NNN:Board 3.3V	Voltage (02h) / Threshold (01h)	2204h / 2204h / 1212h	Board 3.3V supply	Y / 0F3Ch
7 / NNN:Board 5VIPMI	Voltage (02h) / Threshold (01h)	2204h / 2204h / 1212h	Management Power (MP) 5V	Y / 0F3Ch
8 / NNN:Volt Battery	Voltage (02h) / Threshold (01h)	2204h / 2000h / 1212h	Battery	Y / 0F3Ch
9 / NNN:Board 1.8V	Voltage (02h) / Threshold (01h)	2204h / 2204h / 1212h	Board 1.8V supply	Y / 0F3Ch
10 / NNN:Board Vtt.9V	Voltage (02h) / Threshold (01h)	2204h / 2204h / 1212h	DDR termination supply	Y / 0F3Ch
11 / NNN:Board 5.0V	Voltage (02h) / Threshold (01h)	2204h / 2204h / 1212h	Board 5V supply	Y / 0F3Ch
12 / NNN:Board 1.5V	Voltage (02h) / Threshold (01h)	2204h / 2204h / 1212h	Board 1.5V supply	Y / 0F3Ch
13 / NNN:Fan1 Speed	Fan (04h) / Threshold (01h)	0000h / 0000h / 1B1Bh	Speed [rpm] Fan 1	N
14 / NNN:Fan2 Speed	Fan (04h) / Threshold (01h)	0000h / 0000h / 1B1Bh	Speed [rpm] Fan 2	N
15 / NNN:Board Reset	OEM (CFh) / 'digital' Discrete (03h)	0002h / 0000h / 0003h	Board reset event	Y / 0002h
16 / NNN:Slot System	Entity presence (25h) / Sensor-specific (6Fh)	0000h / 0000h / 0003h	Board is in System Slot (SYSEN)	N
17 / NNN:PCI Present	Entity presence (25h) / Sensor-specific (6Fh)	0000h / 0000h / 0003h	Board is selected (BDSEL) and in system slot (SYSEN)	N
18 /	Entity presence (25h) /	0000h /	Value is always 1	N

SENSOR Number / ID string	SENSOR TYPE (CODE) / EVENT/READING TYPE (CODE)	Ass. Mask / Deass. Mask / Reading Mask	DESCRIPTION	Causes red Health LED on error *) / Reading Mask
NNN:CTCA chassis	Sensor-specific (6Fh)	0000h / 0003h		
19 / NNN:EvRcv Lost	Cable/Interconnect (1Bh) / 'digital' Discrete (03h)	0003h / 0000h / 0003h	Connection to event receiver is lost	N
20 / NNN:IPMI WD	Watchdog2 (23h) / Sensor-specific (6Fh)	010Fh / 0000h / 010Fh	IPMI Watchdog	Y / 010Fh
21 / NNN:IPMB State	IPMB status change (F1h) / 'digital' Discrete (03h)	000Fh / 0000h / 000Fh	IPMB-0 state (refer to PICMG 3.0 Rev 2.0, 3-113, byte 4) Normal = 2: IPMB-A enabled, IPMB-B disabled	N
22 / NNN:ACPI State	System ACPI Power State (022h) / Sensor-specific (6Fh)	0003h / 0000h / 0003h	System ACPI Power State	N
23 / NNN:Health Error	Platform Alert (24h) / 'digital' Discrete (03h)	0000h / 0000h / 0003h	Aggregates states (power, temperatures etc.). Visualization by the Health LED.	N
24 / NNN:CPU 0 Status	Processor (07h) / Sensor-specific (6Fh)	0463h / 0400h / 04E3h	CPU status. Offset 0ah: "Processor Automatically Throttled"	Y / 0402h
25 / NNN:Memory	Memory (0Ch) / Sensor-specific (6Fh)	00BBh / 00BBh / 00BBh	Memory error	Y / 00BBh
26 / NNN:POST Value	POST value OEM (C6h) / Sensor-specific (6Fh)	4000h / 0000h / 40FFh	POST code value (port 80h)	N
27 / NNN:SysFirmError	System Firmware Progress (0Fh) / Sensor-specific (6Fh)	0001h / 0001h / 0001h	POST code doesn't change. BIOS sticks.	Y / 0001h
28 / NNN:Critical Int	Critical Interrupt (13h) / 'digital' Discrete (05h)	0001h / 0001h / 0001h	NMI currently active	Y / 0001h
29 / NNN:SMI Timeout	Critical Interrupt (13h) / 'digital' Discrete (05h)	0002h / 0002h / 0002h	SMI interrupt too long active	Y / 0002h
30 / NNN:FWH0 BootErr	Boot error (1Eh) / Sensor-specific (6Fh)	0008h / 0008h / 0008h	Firmware Hub 0 (Boot Flash 0) boot error	Y / 0008h
31 / NNN:FWH1 BootErr	Boot error (1Eh) / Sensor-specific (6Fh)	0008h / 0008h / 0008h	Firmware Hub 1 (Boot Flash 1) boot error	Y / 0008h
32 /	POST Memory resize (0Eh) /	0002h / 0002h /	BIOS reports memory size	Y

SENSOR Number / ID string	SENSOR TYPE (CODE) / EVENT/READING TYPE (CODE)	Ass. Mask / Deass. Mask / Reading Mask	DESCRIPTION	Causes red Health LED on error [*)] / Reading Mask
NNN:CmosMemSize	'digital' Discrete (03h)	0002h	change	/ 0002h
33 / NNN:Preboot Pswd	Platform Security Violation Attempt (06h) / Sensor-specific (6Fh)	0012h / 0012h / 0012h	Password violation	Y / 0012h
34 / NNN:Lan1 Link	LAN (27h) / Sensor-specific (6Fh)	0000h / 0000h / 0003h	LAN 1 heart beat in case of gigabit connection (2 = heart beat detected)	N
35 / NNN:Lan2 Link	LAN (27h) / Sensor-specific (6Fh)	0000h / 0000h / 0003h	LAN 2 heart beat in case of gigabit connection (2 = heart beat detected)	N
36 / NNN:Pwr Denied	Platform Alert (24h) / 'digital' Discrete (03h)	0002h / 0002h / 0003h	1 = o.k., no alert, power not denied	N
37 / NNN:FRU Agent	OEM FRU Agent (C5h) / Discrete (0Ah)	0140h / 0000h / 0147h	FRU Initialization Agent state	Y / 0140h
38 / NNN:IPMC Storage	Management Subsystem Health (28h) / Sensor-specific (6Fh)	0002h / 0000h / 0003h	IPMI controller storage access error	Y / 0002h
39 / NNN:Firm Upg Mng	Firmware Upgrade Manager (C7h) / Sensor-specific (6Fh)	010Fh / 0000h / 010Fh	Status of Firmware Upgrade Manager	N
40 / NNN:IpmC Reboot	Platform Alert (24h) / 'digital' Discrete (03h)	0002h / 0000h / 0003h	2 = Management controller is (re-)booting	N
41 / NNN:Ver change	Firmware version change (2Bh) / Sensor-specific (6Fh)	0002h / 0000h / 0002h	Management controller's firmware version changed	N
42 / NNN:SEL State	Event Logging Disabled (10h) / Sensor-specific (6Fh)	003Ch / 0000h / 003Ch	State of event logging	N
43 / NNN:IPMI Info-1	OEM Firmware Info 1 (C0h) / OEM (70h)	0003h / 0000h / 7FFFFh	For internal use only	N
44 / NNN:IPMI Info-2	OEM Firmware Info 2 (C0h) / OEM (71h)	0003h / 0000h / 7FFFFh	For internal use only	N
45 / NNN:IniAgent Err	Initialization Agent (C2h) / 'digital' Discrete (03h)	0002h / 0000h / 0003h	Initialization Agent error status. Used on BMC only. 1 = error free	Y / 0002h

[*)] Please note that the "Health" LED is always red if the payload is not active (Blue Hot Swap LED is on).

9.2   Sensor Thresholds

SENSOR Number / ID string	Lower non recoverable	Lower critical	Lower non critical	Normal min	Nominal	Normal max	Upper non critical	Upper critical	Upper non recoverable
1 / NNN:Temp CPU	-5 °C E2: -45 °C [1]	-3 °C E2: -43 °C[1]	-1 °C E2: -41 °C[1]	0 °C E2: -40 °C[1]	75 °C	85 °C	90 °C	100 °C	110 °C
2 / NNN:Temp Board 1	-5 °C E2: -45 °C[1]	-3 °C E2: -43 °C[1]	-1 °C E2: -41 °C[1]	0 °C E2: -40 °C[1]	65 °C	75 °C	80 °C	90 °C	100 °C
3 / NNN:Temp Board 2	-5 °C E2: -45 °C[1]	-3 °C E2: -43 °C[1]	-1 °C E2: -41 °C[1]	0 °C E2: -40 °C[1]	65 °C	75 °C	80 °C	90 °C	100 °C
6 / NNN:Board 3.3V	n.a. [2]	3.11 V	n.a. [2]	3.13 V	3.30 V	3.47 V	n.a. [2]	3.51 V	n.a. [2]
7 / NNN:Board 5VIPMI	n.a. [2]	4.66 V	n.a. [2]	4.70 V	5.0 V	5.31 V	n.a. [2]	5.36 V	n.a. [2]
8 / NNN:Volt Battery	n.a. [2]	1.98 V	n.a. [2]	2.0 V	3.0 V	3.6 V	n.a. [2]	3.63 V	n.a. [2]

SENSOR Number / ID string	Lower non recoverable	Lower critical	Lower non critical	Normal min	Nominal	Normal max	Upper non critical	Upper critical	Upper non recoverable
9 / NNN:Board 1.8V	n.a. [2]	1.68 V	n.a. [2]	1.7 V	1.8 V	1.9 V	n.a. [2]	1.93 V	n.a. [2]
10 / NNN:Board Vtt.9V	n.a. [2]	0,83 V	n.a. [2]	0.84 V	0.9 V	0.96 V	n.a. [2]	0.98 V	n.a. [2]
11 / NNN:Board 5.0V	n.a. [2]	4.66 V	n.a. [2]	4.70 V	5.0 V	5.31 V	n.a. [2]	5.36 V	n.a. [2]
12 / NNN:Board 1.5V	n.a. [2]	1.41 V	n.a. [2]	1.45 V	1.5 V	1.57 V	n.a. [2]	1.59 V	n.a. [2]

[1] Only for boards with E2 (extended temperature specification) grade
[2] Not applicable i.e. not used and not settable

9.3   OEM sensor types and OEM sensor event/reading types

OEM SENSOR TYPE (CODE)	OEM EVENT/READING TYPE (CODE)	DESCRIPTION	
Firmware Info 1 (C0h)	70h	Internal Diagnostic Data	
Firmware Info 2 (C0h)	71h	Internal Diagnostic Data	
Initialization Agent (C2h)	03h ('digital' Discrete)	Offsets / events: 0: Initialization O.K. 1: Initialization Error	
IPMBL State (C3h)	6Fh (sensor type specific)	Sensor discrete State	Meaning
		08h	IPMB-L running
		others	IPMB-L not running
FRU Agent (C5h)	0Ah (Discrete)	FRU initialization agent, using a standard reading type.	
Post Value (C6h)	6Fh (sensor type specific)	Error is detected if the POST code is != 0 and doesn't change for a defined amount of time. In case of no error: Bits [7:0] = POST code (payload Port 80h) In case of error: Bits [15:0] = 4000h Data2 = POST code, low nibble Data3 = POST code, high nibble	
Firmware Upgrade Manager (C7h)	6Fh (sensor type specific)	Offsets / events: 0 : First Boot after upgrade 1 : First Boot after rollback (error) 2 : First Boot after errors (watchdog) 3 : First Boot after manual rollback 4..7 : Reserved 8 : Firmware Watchdog Bite, reset occured	

 ID 37494, Rev. 1.0

OEM SENSOR TYPE (CODE)	OEM EVENT/READING TYPE (CODE)	DESCRIPTION
Board Reset (CFh)	03h ('digital' Discrete)	Data 2 contains the reset type: ...WARM = 0 ...COLD = 1 ...FORCED_COLD = 2 ...SOFT_RESET = 3 ...MAX = 4 Data 3 contains the reset source: ...IPMI_WATCHDOG = 0 ...IPMI_COMMAND = 1 ...PROC_INT_CHECKSTOP = 2 ...PROC_INT_RST = 3 ...RESET_BUTTON = 4 ...POWER_UP = 5 ...LEG_INITIAL_WATCHDOG = 6 ...LEG_PROG_WATCHDOG = 7 ...SOFTWARE_INITIATED = 8 ...SETUP_RESET = 9 ...UNKNOWN = 0xFF

OEM SENSOR TYPE (CODE)	OEM EVENT/READING TYPE (CODE)	DESCRIPTION	
e.g. for Power Good / Power Good Event	73h	Sensor-specific Offset	Event Bit set = o.k.
		0h	HS fault#
		1h	HS early fault#
		2h	DEG#
		3h	FAL#
		4h	n.a.
		5h	vccCore good
		6h	n.a.
		7h	n.a.
		8h	n.a.
		9h	n.a.
		Ah	All power good
		Bh	n.a.
		Ch	n.a.
Hot swap sensor (F2h)	6Fh (sensor type specific)	Sensor-specific Offset	Event
		00h	Handle close
		01h	Handle open
		02h	Quiesced

10. IPMI Communication LEDs

10.1 Overview

There are three IPMI communication LEDs on the face plate.

Blue Hot Swap LED
Color: Blue = 1
Labeled: 'HS'
Meaning: Indicates the Hot Swap state of the inserted and powered module.
Meaning: On = a) Module may be extracted or
                      b) power is off (e.g. after shut down) and handle is closed
            Blinking = Hot Swap active, don't extract
            Off = Module in normal operation. Don't extract.
Index in IPMI commands "Get/Set LED State": 0

LED I1 (Out Of Service)
Color: Green / red = 3 / 2
Labeled: 'I1'
Meaning: Indicates the "Out Of Service" state of the powered module.
Behavior: Solid red = Management controller out of service or not programmed or in reset
                      state
            Off = Management controller is running
            Pulsing green = Traffic on the IPMB-0 bus
Index in IPMI commands "Get/Set LED State": 1

LED I2 (Health)
Color: Green / red = 3 / 2
Labeled: 'I2'
Behavior: Blinking = Management controller is running, showing its heart beat.
            Pulsing = KCS interface active.
            Off = Management controller is not running.
            Any action, green = No health error detected (refer to sensor "Health Error").
            Any action, red = Health error detected.
Index in IPMI commands "Get/Set LED State"s: 2

10.2 Programming of the LEDs

All LEDs can be programmed by the IPMI commands "Set FRU LED State Command"
although this is not recommended because they have a dedicated function. But lamp test will
make sense.

LED I1 and I2 will not keep statically ON or OFF because they are continuously showing the
activities of IPMB-0 and KCS interface which will be affected by the programming.

11. FRU Data

11.1 Structure and Functionality

The Management Controller provides 4 kB non-volatile storage space for FRU information.

Please refer to [4] which defines the structure of FRU data.

Full low level access to read or write a module's FRU Information is provided by regular IPMI FRU Device commands. Please be careful when writing FRU information directly using standard IPMI commands because there is no write protection. Damaging the FRU Information e.g. may confuse a shelf management software which uses the FRU data.

To avoid this damage there is a Kontron Linux tool 'frum' (refer to 12.1, *Linux*), which allows to display and partially modify FRU data. E.g. the 'frum' tool makes it easy to modify Product Info Area fields like Product Version or Product Serial Number.

11.2 Board Specific FRU Data

Supported are the following FRU data areas and data fields:

FRU Board Info Area

Manufacturing date / time

Board manufacturer   (c7): "Kontron"

Board Product Name   (c6): "CP6001"

Board Serial Number   (CA): "1234567890" [*]

Board Part Number   (C8): "12345678" [*]

FRU File ID   (C8): "STD_0100"

FRU Product Info Area

Product manufacturer (c7): "Kontron"

Product Name   (c6): "CP6001"

Product Part Number   (C4): "0000" [*]

Product Version   (D9): "00000000000000000000000" [**]

Product Serial Number (D9): "00000000000000000000000" [**]

Asset Tag   (D9): "_____" [**]

FRU File ID   (C8): "STD_0100"

CustomData   (D5): 'MAC=CC:CC:CC:CC:CC:CC' [*]

[*] Field will be modified during the manufacturing process

[**] Field is free for user. Please note that changes need special care (checksums). Refer to [4].

11.3 Downloading the FRU Data

Normally the user doesn't need to download the FRU data because the module is supplied with it before shipping.

But if needed the standard way to download FRU information to the module is to use the open tool 'ipmitool' [1] for the download of an image file (e.g. 'ipmitool fru write 0 <file name>').

All IPMI interfaces which are offered by ipmitool are usable (KCS interface, IPMB bus, LAN). This allows local upgrade or remote upgrade. Please note that the KCS interface is only usable on a powered payload.

12. OS Support / Tools

12.1 Linux

Normally all drivers and kernel modules needed for communication between the payload sided software and the Management Controller firmware via the KCS interface come with the distribution. Newest sources can be downloaded from http://openipmi.sourceforge.net. There may be downloaded the OpenIPMI project as well. The OpenIPMI library package includes some applications and the needed libraries. One of the applications is 'ipmicmd' which makes it possible to send and receive raw IPMI sequences; another, the 'ipmi_ui', provides a higher level interface and thus it does not require deep IPMI knowledge from the user side.

Another very useful all-in-one tool is 'ipmitool' [1] (http://ipmitool.sourceforge.net). It provides a user friendly interface to many IPMI features and extensions, for example to PICMG LED control and for the upgrade of the MMC's firmware ("ipmitool fwum…").

There is a Kontron Modular Computers' IPMI ToolKit which contains some tools for the customer for the monitoring and the maintenance of some IPMI functionalities.

The following command line tools are included in the IPMI Tool Kit:

- frum: display and modification of FRU data

- temptool: selection, display, and storage of temperature and voltage sensor values

All these tools are OpenIPMI based. This toolkit is available on the "AMC Kit CD". If this CD was not shipped with your module then please contact our support (support@kontron-modular.com).

Please refer to the manual "MAN_LIN_IPMI_TOOLKIT_0103.pdf" being included in the package.

Before using any of the tools mentioned above the needed OpenIPMI kernel modules have to be loaded if not yet done automatically at Linux start up:

```
# modprobe ipmi_si
# modprobe ipmi_devintf
```