University of Wisconsin
High Energy Physics
MMC Reference Design
Quick Specification

Revision 1

T. Gorski

April 26, 2011

Overview

The UW MMC Reference Design is an AVR32 UC3A1512 Microcontroller-based circuit designed to provide the IPMI Module Management Controller function to AMC modules. The circuit is powered by the +3.3V Management Power supply defined in the AMC specification. This supply comes active shortly after the system detects the presence of the AMC module. Upon application of power, the software in the microcontroller begins execution, scanning the geographic address pins GA0-GA2 (part of the AMC backplane interface) to determine the Intelligent Platform Management Bus (IPMB) address of module within the MicroTCA crate. With its IPMB address calculated, the software completes its initialization and waits for IPMI commands from the system to continue the booting of the AMC module.

The software in the UC3A1512 Microcontroller is divided into two main functional blocks: the Module Management Controller (MMC), and the Payload Manager. The MMC is an intelligent controller as defined in PICMG and IPMI specifications. Its purpose is to provide the platform management system with sufficient information about the state and capabilities of the module so as to allow its general operation in the MicroTCA crate. Major functions of the MMC include negotiating the delivery of Payload (+12V) power to the module, and facilitating the insertion/remove of the module per the front panel handle and Hot Swap sensor. The Payload Manager provides a set of custom functions that lie outside the scope of the PICMG and IPMI specifications. These functions support control, configuration and monitoring of the AMC module circuitry operating on the +12V supply. Both functional blocks are controlled through the IPMB interface.

The current supports 10 sensors: the Hot Swap sensor, 8 analog sensors for measuring voltage and temperature, and 1 digital sensor to read the state of eight GPIO pins. Of the analog sensors, two provide the ambient and hot spot temperature measurements mandated by the AMC specification, and two more monitor the +12V payload power and derived +3.3V (or +2.5V) backend power. The remaining 4 inputs monitor secondary voltages on the AMC module. IPMI front panel LED support includes the Blue LED, LED1, and LED2. The microcontroller firmware supports all mandatory and some optional IPMI commands for MMCs per the specifications.

Electrical Interface

All digital signals use 3.3V LVCMOS logic levels. The analog input signals ADCx have maximum input levels which are dependent on the resistor divider specific to that input, with the full scale signal range given on a per signal basis in the description below.

- **Logic Ground**. Same signal as defined on the AMC backplane connector, which should also be the bulk ground for the AMC module.
- **Mgmt Pwr (+3.3VMP).** This is the management +3.3V power provided by the power supply for use by the MMC. Connected to the MP signal (pin 4) on the AMC backplane connector.
- **IPMIGA0-IPMIGA2**. *Inputs*. Geographic Address signals GA0-GA2. Used to assign the address of a module on the IPMB-L. See section 3.2.1 of the AMC.0 Specification. Connected to pins 5, 17, and 26, respectively at the AMC backplane connector.
- **IPMB_SCL**. *Input.* I^2C Clock signal for IPMB. Connected to SCL_L (pin 56) at the AMC backplane connector.
- **IPMB_SDA**. *Bidirectional.* I^2C Data/Address signal for the IPMB. Connected to SDA_L (pin 71) at the AMC backplane connector.
- **ADC_SENSE_12V**. *Analog Input.* ADC input for the payload +12V power supply, which is the PWR signal at the AMC backplane connector. If the +12V supply is fused on the AMC board, this signal should be connected at the backplane, or unfused side.
- **ADC_SENSE_BKEND_3.3V**. *Analog Input.* ADC input for the backend +3.3V power supply that is derived from the payload +12V. (Can be configured for a 2.5V backend voltage).
- **AMC_TEMP**. *Analog Input.* ADC input for the AMC module temperature sensor. This pin should be connected to the output of a TMP36 temperature sensor that is located on the AMC module near the hottest device, and which is powered by the back end +3.3V supply.
- **ADC_SENSE1**. *Analog Input.* Uncommitted ADC channel. Has a 1.33X attenuator on its input. It monitors voltages up to 4.0V.
- **ADC_SENSE3-ADC_SENSE5**. *Analog Inputs.* Uncommitted ADC channels. They have 1X attenuators, and a full scale input range of 3.0 V.
- **~IPMI_BLLED**. *Open Collector Output*. Drive signal for IPMI Blue LED. Connect cathode of blue LED on the AMC module to this signal, and the anode to management power through the appropriate bias resistor.
- **~IPMI_LED1**. *Open Collector Output*. Drive signal for IPMI LED1 (red). Connect cathode of LED1 on the AMC module to this signal, and the anode to management power through the appropriate bias resistor.
- **~IPMI_LED2**. *Open Collector Output*. Drive signal for IPMI LED2 (green). Connect cathode of LED2 on the AMC module to this signal, and the anode to management power through the appropriate bias resistor.
- **~IPMI_ENAIN**. *Input.* Enable signal for the microcontroller. Connect to the ENABLE# signal (pin 41) on the AMC connector.

- **BKEND_PWR_ENA**. *LVTTL Output*. LVTTL power enable signal for backend power converters on the AMC module. All backend power supplies on the AMC module should be enabled directly or indirectly through this signal. Active high—logic 1 to enable back end power, logic 0 to disable.
- **BOOT_MODE**. *LVTTL Output.* Selects the FPGA boot mode for the AMC module. Logic 0 selects Trigger (operational) Mode, while Logic 1 selects Maintenance mode. Signal controlled remotely with custom commands via the IPMB interface.
- **~FPGA_CPU_RST**. *LVTTL Open Collector Output.* Signal may optionally be connected to the reset signal for any CPU operating on the AMC module. Permits remote assertion of the reset signal via the IPMB interface.
- **FPGA_FLASH_LOAD**. Reserved signal for future development. Purpose will be to signal FPGAs on the AMC module to reload their programming from the on-board FLASH ICs.
- **~FSIO_SS{0:3}**. Reserved for future development. Signals will act as dedicated SPI port Slave Select inputs to up to 4 FPGA on the AMC module. Signals are also used as outputs during the FPGA detection procedure (see the FSIO_SCANSLV signal).
- **~FSIO_SCANSLV**. Used to detect FPGAs on the AMC module with a SPI Configuration Port that are connected to the microcontroller SPI peripheral. When the FSIO_SCANSLV signal goes active, any FPGAs on the main board that have finished loading their firmware image from FLASH memory will respond by asserting its associated ~FSIO_SSx signal. These signals are scanned by the microcontroller software to determine which FPGA have active configuration ports.
- **FSIO_MISO**. SPI port Master In Slave Out signal. Driven by the selected slave FPGA during SPI port transactions.
- **FSIO_MOSI**. Reserved SPI port Master Out Slave In. Broadcast from the microcontroller to all FPGA SPI ports on the AMC module.
- **FSIO_SCK**. Reserved SPI port Serial Clock. Broadcast from the microcontroller to all FPGA SPI ports on the AMC module.
- **GPIO_0-GPIO_9**. Reserved General Purpose I/Os between reference design and AMC module.

IPMI Sensors

The UC3A1512 microcontroller has 8 ADC inputs, all of which used as sensors by the reference design. 4 of those 8 sensors are core sensors which should be connected on any AMC that uses the reference design, and the other 4 are optional. It also has the mandatory IPMI Hot Swap Sensor and an additional sensor that reads out the value of GPIO_0—GPIO_7. The table below shows the relationship between UC3A1512 ADC channels and IPMI Sensor numbers and names.

ADC Ch. Num.	Default IPMI Sensor Name	IPMI Sensor Number	Description	Connection Mandatory?
	Hotswap	0	IPMI Hot Swap sensor	*yes*
0	*+12V*	1	Payload +12V power to AMC slot from the crate supply.	*yes*
1	*ADC1*	2	ADC1 Input, 4.0V input full scale range.	*no*
2	*+3.3V BkEnd*	3	Backend +3.3V power output, powered from +12V supply.	*yes*
3	*ADC3*	4	ADC3 Input, 3.0V input full scale range.	*no*
4	*ADC4*	5	ADC4 Input, 3.0V input full scale range.	*no*
5	*ADC5*	6	ADC5 Input, 3.0V input full scale range.	*no*
6	*Hot Spot Temp*	7	TMP36 temperature sensor input, located on AMC module near hottest device, powered by +3.3V backend supply.	*yes*
7	*Amb. Temp*	8	TMP36 temperature sensor input, located on AMC module near MMC microcontroller, powered by +3.3V backend supply.	*yes*
	GPIO 7:0	9	Current value of GPIO pins 7:0	*no*

As indicated in the table, ADC channels 0, 2, 6, and 7 must be connected when operating the reference design. The two temperature sensors, on channels 6 and 7, are required by the AMC specification. The two voltage sensors, on channels 0 and 2, are required for proper operation of the Payload Manager.

Sensor Thresholds and Monitoring

Both the required and optional analog sensors have default definitions in the MMC's Sensor Device Repository (SDR), which is an IPMI-defined method of representing remote sensors to the management system. Per the specification, analog sensors can have

up to three upper and three lower threshold levels for alarm event generation. The thresholds fall into three classes:

- Non-critical. Least severe threshold, closest to the normal operating range for the sensor.
- Critical. More severe threshold, further from normal operating range but still within operating limits.
- Non-recoverable. Indicates a threshold beyond which the AMC module should not attempt to operate.

The default sensor definition in the reference design is to define both upper and lower thresholds for the +12V and +3.3V Back End sensors, upper thresholds only for the two temperature sensors, and no thresholds for any of the optional ADC sensors. Under the IPMI specification, there are several ways for the platform management application to maintain awareness of the threshold state of sensors on an individual AMC module. One method is to monitor the System Event Log (SEL) for threshold events generated automatically by the MMC. Another method is for the platform management application to periodically poll the sensor values directly via the IPMI *Get Sensor Reading* command. Both of these methods have their advantages and disadvantages, but both require the presence of a platform management application running on a remote computer in communication with the MMC and ready to intervene should a Non-recoverable event threshold be reached.

In recognition of the fact that such an application may not always be available, especially during the project development phase, the Payload Manager also monitors these sensors and thresholds. Should a sensor exceed its Non-recoverable threshold, the Payload Manager can intervene to turn off the BKEND_PWR_ENA signal, which should result in all secondary supplies (those derived from the +12V) being deactivated, given the proper implementation of back end power on the AMC. When it takes the action, the Payload Manager will generate a Hot Swap sensor Backend Power Shutdown/Failure event. Upon receipt of this event the Carrier Manager on the MCH will command the crate power supply to turn off +12V to the slot. This local response capability allows the fastest possible reaction to a fault condition, but can be masked off should it be determined that action should originate only from a higher level. The important point to remember is that neither the Shelf Manager nor Carrier Manager resident on the MCH automatically take corrective action at the occurrence of a Non-recoverable event on the analog sensors.

IPMI LED1 is used by the Payload Manager to display the current threshold state of the sensors. This LED is normally off. When any sensor is at its Non-critical threshold, LED1 has a slow blink (~1.5 Hz) flashing on for about 20% of the time. When a sensor reaches the Critical threshold, LED1 has a fast blink (~2.5 Hz), and is on about 50% of the time. Should the highest alerting sensor pass below a threshold (e.g., cooling is restored, temperature drops), LED1 will change accordingly. If the abnormal conditions persist and a sensor reaches the Non-recoverable threshold, LED1 will latch and stay on continuously (and back end power will be shut off). The Payload Manager will keep the module latched in this state until either power is cycled to the entire crate, or until the

module extraction handle is opened and closed. Note that the *Set Handle Override* custom command makes it possible to electronically simulate the cycling of the module extraction handle, such that a physical cycling is not necessary.

Non-IPMI LEDs

The reference design also contains three "general purpose" LEDs. Their function is described in the table below. All LEDs are wired to be lit with an active low output at the Microcontroller pin, and should have their anode pulled up to 3.3V Management Power through an appropriate resistor.

LED Name	Microcontroller Pin	Description
GPLED0	PB16 (pin 96)	Blinks at 1Hz Rate when MMC software is running on the microcontroller.
GPLED1	PB17 (pin 98)	Flashes when an incoming IPMI message is detected.
GPLED2	PB18 (pin 99)	Flashes when activity occurs on the SPI configuration interface.
GPLED3	PB19 (pin 100)	On when Boot Mode = 0 (Trigger Mode), off when Boot Mode = 1 (Maintenance Mode).

IPMI Command List

The table below lists all of the standard and custom IPMI commands supported by the current version of the MMC software. Note that this may expand with future revisions of the firmware.

	CMD	NetFn
Get Device ID	01h	App (06h/07h)
Cold Reset	02h	App (06h/07h)
Broadcast “Get Device ID”	01h	App (06h/07h)
Set Event Receiver	00h	S/E (04h/05h)
Get Event Receiver	01h	S/E (04h/05h)
Platform Event (a.k.a. “Event Message”)	02h	S/E (04h/05h)
Get Device SDR Info	20h	S/E (04h/05h)
Get Device SDR	21h	S/E (04h/05h)
Reserve Device SDR Repository	22h	S/E (04h/05h)
Get Sensor Hysteresis	25h	S/E (04h/05h)
Set Sensor Threshold	26h	S/E (04h/05h)
Get Sensor Threshold	27h	S/E (04h/05h)
Set Sensor Event Enable	28h	S/E (04h/05h)
Get Sensor Event Enable	29h	S/E (04h/05h)
Get Sensor Reading	2Dh	S/E (04h/05h)
Get FRU Inventory Area Info	10h	Storage (0Ah/0Bh)
Read FRU Data	11h	Storage (0Ah/0Bh)
Write FRU Data	12h	Storage (0Ah/0Bh)
Get PICMG Properties	00h	PICMG (2Ch/2Dh)
FRU Control	04h	PICMG (2Ch/2Dh)
Get FRU LED Properties	05h	PICMG (2Ch/2Dh)
Get LED Color Capabilities	06h	PICMG (2Ch/2Dh)
Set FRU LED State	07h	PICMG (2Ch/2Dh)
Get FRU LED State	08h	PICMG (2Ch/2Dh)
Get Device Locator Record ID	0Dh	PICMG (2Ch/2Dh)
FRU Control Capabilities	1Eh	PICMG (2Ch/2Dh)
Set Backend Power	01h	Custom (32h/33h)
Get Backend Power	02h	Custom (32h/33h)
Set Payload Manager Settings	03h	Custom (32h/33h)
Get Payload Manager Settings	04h	Custom (32h/33h)
Get Fault Status	05h	Custom (32h/33h)
Set Boot Mode	06h	Custom (32h/33h)
Get Boot Mode	07h	Custom (32h/33h)
Set Sensor Alarm Mask	0Bh	Custom (32h/33h)
Get Sensor Alarm Mask	0Ch	Custom (32h/33h)
Set Handle Override	0Fh	Custom (32h/33h)
Set Current Requirement	10h	Custom (32h/33h)
Set Analog Scale Factor	13h	Custom (32h/33h)
Get Analog Scale Factor	14h	Custom (32h/33h)

Get Time Statistics	28h	Custom (32h/33h)
Set System Time	29h	Custom (32h/33h)
Get System Time	2Ah	Custom (32h/33h)
Poll FPGA Configuration Port	30h	Custom (32h/33h)
FPGA Configuration Read Status Register	31h	Custom (32h/33h)
FPGA Configuration Write Control Register	32h	Custom (32h/33h)
FPGA Configuration Write Data	33h	Custom (32h/33h)
FPGA Configuration Read Data	34h	Custom (32h/33h)
FPGA Configuration Nonvolatile Header Write	35h	Custom (32h/33h)
FPGA Configuration Nonvolatile Header Read	36h	Custom (32h/33h)
Get Nonvolatile Area Info	40h	Custom (32h/33h)
Raw Nonvolatile Write	41h	Custom (32h/33h)
Raw Nonvolatile Read	42h	Custom (32h/33h)
Check EEPROM Busy	43h	Custom (32h/33h)
EEPROM Erase	44h	Custom (32h/33h)

Custom Command Reference

Set Backend Power

This command controls the on/off state of the back end power.

	Byte	Data Field
Request Data	1	Backend on/off power enable. Set to 1 to enable backend power, 0 to disable.
Response Data	1	Completion Code

Get Backend Power

This command returns the current on/off state of the back end power control. It returns the command state, and also the instantaneous pin state. The "disabled" pin state may be returned even when the back end power is enabled but other system conditions prevent its delivery. Such conditions can include a fault condition, or the +12V payload power not being enabled to the module (e.g., result of handle being open).

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2	Backend Power Setting (1=enabled, 0=disabled)
	3	Backend Power Pin State (1=enabled, 0=disabled)

Set Payload Manager Settings

This command makes changes to the general settings in the Payload Manager. Settings are retained in nonvolatile memory.

	Byte	Data Field
Request Data	1	Field select mask. Setting a bit=1 selects associated field for update, bit=0 leaves field unchanged. Bit definitions as follows: [7] - Default backend auto-power enable [6] – Default boot mode [5] - FPGA auto-configuration inhibit [4] - Backend power shutdown event-enable [3] - Backend Monitor global alarm level [2:0] - Unused, reserved
	2	Default backend auto-power enable setting (1=enable backend pwr when payload power turns on, 0=disable backend power)
	3	Default boot mode (0=trigger mode, 1=maintenance mode)
	4	FPGA auto-configuration inhibit setting (1=inhibit FPGA-auto-configuration function, 0=permit FPGA-auto-configuration)
	5	Backend power shutdown event-enable setting (1=send hotswap event if backend power is commanded to shut down, 0=do not send event)
	6	Backend Monitor global alarm level (0=no mask, 1=non-critical, 2=critical, 3=non-recoverable)
Response Data	1	Completion Code

Get Payload Manager Settings

This command returns the general settings currently in effect in the Payload Manager.

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2	Default backend auto-power enable setting (1=enable backend pwr when payload power turns on, 0=disable backend power)
	3	Default boot mode (0=trigger mode, 1=maintenance mode)
	4	FPGA auto-configuration inhibit setting (1=inhibit FPGA-auto-configuration function, 0=permit FPGA-auto-configuration)
	5	Backend power shutdown event-enable setting (1=send hotswap event if backend power is commanded to shut down, 0=do not send event)
	6	Backend Monitor global alarm mask level (0=no mask, 1=noncritical, 2=critical, 3=non-recoverable)

Get Fault Status

Returns current fault status and source type.

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2	Alarm severity level (0=normal, 1=noncritical, 2=critical, 3=nonrecoverable)
	3	Alarm source (0=none, 1=temperature sensor, 2=voltage sensor, 3=backend device)

Set Boot Mode

This command controls the value of the BOOT_MODE I/O pin. At startup the BOOT_MODE pin is set to the Default Boot Mode value in nonvolatile memory (see the *Set Payload Manager Settings* command). During operation the boot mode can be changed as necessary using this command.

	Byte	Data Field
Request Data	1	Boot mode. Set to 0 for Trigger Mode and 1 for Maintenance mode.
Response Data	1	Completion Code

Get Boot Mode

This command returns the current state of the BOOT_MODE pin.

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2	Current boot mode. Returns 0 for Trigger Mode and 1 for Maintenance mode.

Sensor Alarm Mask

Command sets the alarm level mask in the Payload Manager for the specified sensor. Alarms at or below the specified level are masked off so that the Payload Manager ignores them. IPMI sensor events may still be generated, however. Setting is stored in nonvolatile memory.

	Byte	Data Field
Request Data	1	IPMI Sensor Number, as defined in the section *Voltage and Temperature Sensor Connections.*
	2	Alarm Mask Value (0=no mask, 1=noncritical, 2=critical, 3=nonrecoverable)
Response Data	1	Completion Code

Get Sensor Alarm Mask

Command returns the current alarm level mask in the Payload Manager for the specified sensor. Alarms at or below the specified level are masked off so that the Payload Manager ignores them.

	Byte	Data Field
Request Data	1	IPMI Sensor Number, as defined in the section *Voltage and Temperature Sensor Connections.*
Response Data	1	Completion Code
	2	Alarm Mask Value (0=no mask, 1=noncritical, 2=critical, 3=nonrecoverable)

Set Handle Override

Command sets or clears an override value for module insertion handle. Use of the override makes it possible to perform certain IPMI state transitions, such as advancing out of the failed state, without physical manipulation of the handle.

	Byte	Data Field
Request Data	1	[7] - 0b = normal mode, 1b = override mode [6:1] - reserved [0] - 0b=force handle to report closed (in) state, 1b=force handle to report an open (out) state. This bit is ignored if bit 7 is set to 0.
	2	Override duration (optional). If this byte is present, it specifies the duration of the override, in 100ms units, with zero indicating that the override is indefinite. Ignored if normal mode is specified.
Response Data	1	Completion Code

Set Current Requirement

This command updates the current requirement record in the FRU mulltirecord area in nonvolatile storage. The change will take effect the next time the FRU data area is read by the Carrier Manager. Current is given in 100 mA units, with a minimum of 0.5A and maximum of 7.0A.

	Byte	Data Field
Request Data	1	12V Current Requirement in 100 mA units.
Response Data	1	Completion Code

Set Analog Scale Factor

Analog inputs have scale factors which convert the external signal in its native units (e.g., degrees C, Volts, etc.) to corresponding mV value, as observed at the ADC input. In the case of voltage inputs, the scaling factors capture the effect of any resister divider network acting as an input attenuator. For temperature inputs, they capture the ºC/mV gain ratio, as well as any offset voltage present on the sensor output, such as the 10mV/ºC gain and 500mV @ 0 ºC offset of the TMP36 sensor.

Each sensor has a gain scaling factor, M, and an offset scaling factor, B. Both are expressed using signed 32-bit integers, and the M is defined as a fraction with separate numerator and denominator values. The scaling factors are used at startup to determine the mapping between the 10-bit ADC + front end sensors/circuitry and the 8-bit readout range associated with the sensor reading factors in the sensor's SDR record. At startup, the code determines the maximum and minimum signal values represented by the SDR record, and uses the analog scaling factors to calculate the projected ADC output values when those min/max signals are present at the input. During runtime this allows the microcontroller to quickly and efficiently convert from a 10-bit ADC value to an 8-bit sensor readout value, automatically factoring in the effects of sensor offsets, input attenuator resistor values, and the SDR sensor reading factors. Using the TMP36 as an example, its gain setting has M_{num}=10, and M_{denom}=1, with an offset B=500. Scaling factors are stored in nonvolatile memory.

	Byte	Data Field
Request Data	1	ADC channel number, starting at 0.
	2	Field index. Fields are indexed as follows: 0 - Scaling factor M numerator 1 - Scaling factor M denominator 2 - Offset B
	3-6	Field value as signed 32-bit integer, least significant byte first.
Response Data	1	Completion Code

Get Analog Scale Factor

This command returns the scaling factors associated with an ADC input.

	Byte	Data Field
Request Data	1	ADC channel number, starting at 0. If omitted, the command returns the number of ADC channels.
Response Data	1	Completion Code
	2	ADC channel number specified in the request. If no ADC channel number was give, this field contains the total number of ADC channels defined.
	3-6	Returns the M_{num} scaling factor for the specified channel as a signed 32-bit integer, LS byte first.
	7-10	Returns the M_{denom} scaling factor for the specified channel as a signed 32-bit integer, LS byte first.
	11-14	Returns the B scaling factor for the specified channel as a 32-bit integer, LS byte first.

Get Time Statistics

Command returns the elapsed time in seconds since microcontroller start-up (up time), and since the back end power was enabled (hot time). Both values are returned as 32-bit unsigned integers. These timers are driven from the reference design 12MHz CPU crystal.

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2-5	MMC uptime, in seconds, as 32-bit unsigned integer, LS byte first.
	6-9	Back end hot time, in seconds, as 32-bit unsigned integer, LS byte first.

Set System Time

The reference design has a 32-bit system clock which increments at a 1 second rate. The clock is driven by a dedicated 32.768 kHz crystal. This command sets the clock to a system time as determined by the platform management application. Although the design places no constraints on the time format, a convenient approach is to use the Unix/Linux convention of a zero value corresponding to the start of January 1, 1970. As the circuit does not have a battery-backed circuit for maintaining time when power is disconnected, the clock resets to 0.

	Byte	Data Field
Request Data	1-4	System time as a 32-bit unsigned integer, LS byte first
Response Data	1	Completion Code

Get System Time

This command returns the current system time for the microcontroller.

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2-5	System time as 32-bit unsigned integer, LS byte first.

Poll FPGA Configuration Port

This command polls the SPI1 port for connected FPGA Configuration Port SPI slaves. It asserts the ~SCANSLV signal from the Microcontroller and samples the ~CSx pins as logic inputs. Any ~CSx lines observed to go to a logic 0 are considered to have FPGA Configuration Ports attached to them.

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2	Port detection flags. 1b = FPGA config port detected, 0b = no FPGA detected [7:4] – reserved, returned as 0000b [3] – SPI CS3 detect flag [2] – SPI CS2 detect flag [1] – SPI CS1 detect flag [0] – SPI CS0 detect flag

FPGA Configuration Read Status Register

This command reads and returns the contents of the FPGA SPI Configuration Port status register.

Request Data	1	FPGA Config Port ID (0-3)
	2	Data byte written to Control Register: [7] – 1b = select UHF for update, 0b = no select [6] – 1b = select LHF for update, 0b = no select [5] – 1b = select CFGRDY for update, 0b = no select [4:2] – reserved [1] – 1b = selected flags should be set, 0b = no change [0] – 1b = selected flags should be cleared, 0b = no chg
Response Data	1	Completion Code
	2	Data byte returned from status register [7] – Handshake Flag 1 (HF1) bit [6] – Handshake Flag 2 (HF2) bit [5] – Config Ready (CFGRDY) bit [4] – Request Config (REQCFG) bit [3:0] –Reserved

FPGA Configuration Write Control Register

This command writes to the Control Register of the selected FPGA SPI Configuration Port.

Request Data	1	FPGA Config Port ID (0-3)
	2	Data byte written to Control Register: [7] – 1b = select HF1 for update, 0b = no select [6] – 1b = select HF2 for update, 0b = no select [5] – 1b = select CFGRDY for update, 0b = no select [4:2] – reserved [1] – 1b = selected flags should be set, 0b = no change [0] – 1b = selected flags should be cleared, 0b = no chg
Response Data	1	Completion Code

FPGA Configuration Write Data

This command performs a raw write to a FPGA SPI Configuration Port.

Request Data	1	FPGA Config Port ID (0-3)
	2	Config Port destination byte address, LS byte
	3	Config Port destination byte address, MS byte
	4	Byte count n for read (20 byte maximum)
	5— 5+(n-1)	Write data
Response Data	1	Completion Code

FPGA Configuration Read Data

This command performs a raw read from a FPGA SPI Configuration Port.

Request Data	1	FPGA Config Port ID (0-3)
	2	Config Port source byte address, LS byte

	3	Config Port source byte address, MS byte
	4	Byte count n for read (20 byte maximum)
Response Data	1	Completion Code
	2—2+(n-1)	Read data

FPGA Configuration Nonvolatile Header Write

This command writes the header in EEPROM at the start of the FPGA Configuration area.

Request Data	1	Config Flags, 2 bits per chip select. (x0b = config data undefined, 01b = defined, but autoconfig disabled, 11b = autoconfig enabled) [7:6] – CS3 autoconfig setting [5:4] – CS2 autoconfig setting [3:2] – CS1 autoconfig setting [1:0] – CS0 autoconfig setting (no command data)
		Byte offset to CS0 config data
		Byte offset to CS1 config data
		Byte offset to CS2 config data
		Byte offset to CS3 config data
		Header Checksum
Response Data	1	Completion Code

FPGA Configuration Nonvolatile Header Read

This command returns the header in EEPROM at the start of the FPGA Configuration area.

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2	Config Flags, 2 bits per chip select. (x0b = config data undefined, 01b = defined, but autoconfig disabled, 11b = autoconfig enabled) [7:6] – CS3 autoconfig setting [5:4] – CS2 autoconfig setting [3:2] – CS1 autoconfig setting [1:0] – CS0 autoconfig setting
	3	Byte offset to CS0 config data
	4	Byte offset to CS1 config data
	5	Byte offset to CS2 config data
	6	Byte offset to CS3 config data
	7	Header Checksum

Get Nonvolatile Area Info

This command returns area definition information for nonvolatile storage. The reference design has 8 kB of EEPROM memory.

	Byte	Data Field
Request Data	1	Requested page index. Nonvolatile area info returned in response depends on which page index is specified.

		Current supported range is 0-1.
Response Data	1	Completion Code
		Responses for Page Index 0
	2	Format code for nonvolatile storage. Returns 0x00 for this version
	3	Requested page index supplied in command (equals 0 for this section)
	4	EEPROM size in bytes, LS byte
	5	EEPROM size in bytes, MS byte
	6	Hardware Header Area byte offset, LS byte
	7	Hardware Header Area byte offset, MS byte
	8	Hardware Header Area size, 8-byte units
	9	Application Device Info Area byte offset, LS byte
	10	Application Device Info Area byte offset, MS byte
	11	Application Device Info Area size, 8-byte units
	12	FRU Data Area byte offset, LS byte
	13	FRU Data Area byte offset, MS byte
	14	FRU Data Area size, 8-byte units
	15	FPGA Configuration Area byte offset, LS byte
	16	FPGA Configuration Area byte offset, MS byte
	17	FPGA Configuration Area size, 32-byte units
		Responses for Page Index 1
	2	Format code for nonvolatile storage. Returns 0x00 for this version
	3	Requested page index supplied in command (equals 1 for this section)
	4	SDR Area byte offset, LS byte
	5	SDR Area byte offset, MS byte
	6	SDR Area size, 32-byte units
	7	Payload Manager Area byte offset, LS byte
	8	Payload Manager Area byte offset, MS byte
	9	Payload Manager Area size, 8-byte units
	10	ADC Scaling Factor Area byte offset, LS byte
	11	ADC Scaling Factor Area byte offset, MS byte
	12	ADC Scaling Factor Area size, 8-byte units

Raw Nonvolatile Write

This command allows direct write of the nonvolatile storage area. It should be used with great care as direct writes have the potential to corrupt the nonvolatile image and affect function of the microcontroller software.

	Byte	Data Field
Request Data	1	Starting byte offset, LS byte
	2	Starting byte offset, MS byte
	3	Byte count n for write (20 byte maximum)
	4—4+(n-1)	Write data

Response Data	1	Completion Code

Raw Nonvolatile Read

This command performs a direct read of the nonvolatile storage area.

	Byte	Data Field
Request Data	1	Starting byte offset, LS byte
	2	Starting byte offset, MS byte
	3	Byte count n for read (20 byte maximum)
Response Data	1	Completion Code
	2—2+(n-1)	Read data

Check EEPROM Busy

This function checks the current busy state of the nonvolatile storage EEPROM. After any write operation, the EEPROM will be busy and unavailable for access for approximately 5ms.

	Byte	Data Field
Request Data		(no command data)
Response Data	1	Completion Code
	2	Returns 1 if EEPROM is busy, and 0 if it is not.

EEPROM Erase

This command causes the microcontroller to erase the entire contents of the EEPROM. This not a normal procedure, and should be used only if the corrupted EEPROM contents prevent MMC operation. After the EEPROM is erased, the microcontroller should be restarted, either by cycling power to it or issuing an IPMI *Cold Reset* command. Upon startup with an empty EEPROM, the firmware will initialize it with a default image that sufficient to allow basic operation of the module. In order to prevent accidental erasures, the command must be executed twice, with two different subfunction codes (see command description). The first time the command is executed, it returns a 4-byte Erase Key, which is then transmitted back to the MMC on the second function call, which initiates the erase. The key will be valid for approximately 30 seconds after it is issued.

	Byte	Data Field
Request Data	1	Subfunction code. Use value AAh on the first call to have the function return a 4-byte erase key. Use value 55h on the second call, along with the Erase key in bytes 2-5 of the request, to complete the erase operation.
	2	Erase key byte 0. Ignored on first (AAh) call, required on second (55h) call
	3	Erase key byte 1. Ignored on first (AAh) call, required on second (55h) call
	4	Erase key byte 2. Ignored on first (AAh) call, required on second (55h) call
	5	Erase key byte 3. Ignored on first (AAh) call, required on second (55h) call
Response Data	1	Completion Code

2	Erase key byte 0. Returned on first (0xaa) call only
3	Erase key byte 1. Returned on first (0xaa) call only
4	Erase key byte 2. Returned on first (0xaa) call only
5	Erase key byte 3. Returned on first (0xaa) call only