IPMI System Manager R&D

J. Tikalsky, T. Gorski

University of Wisconsin

April 9, 2013

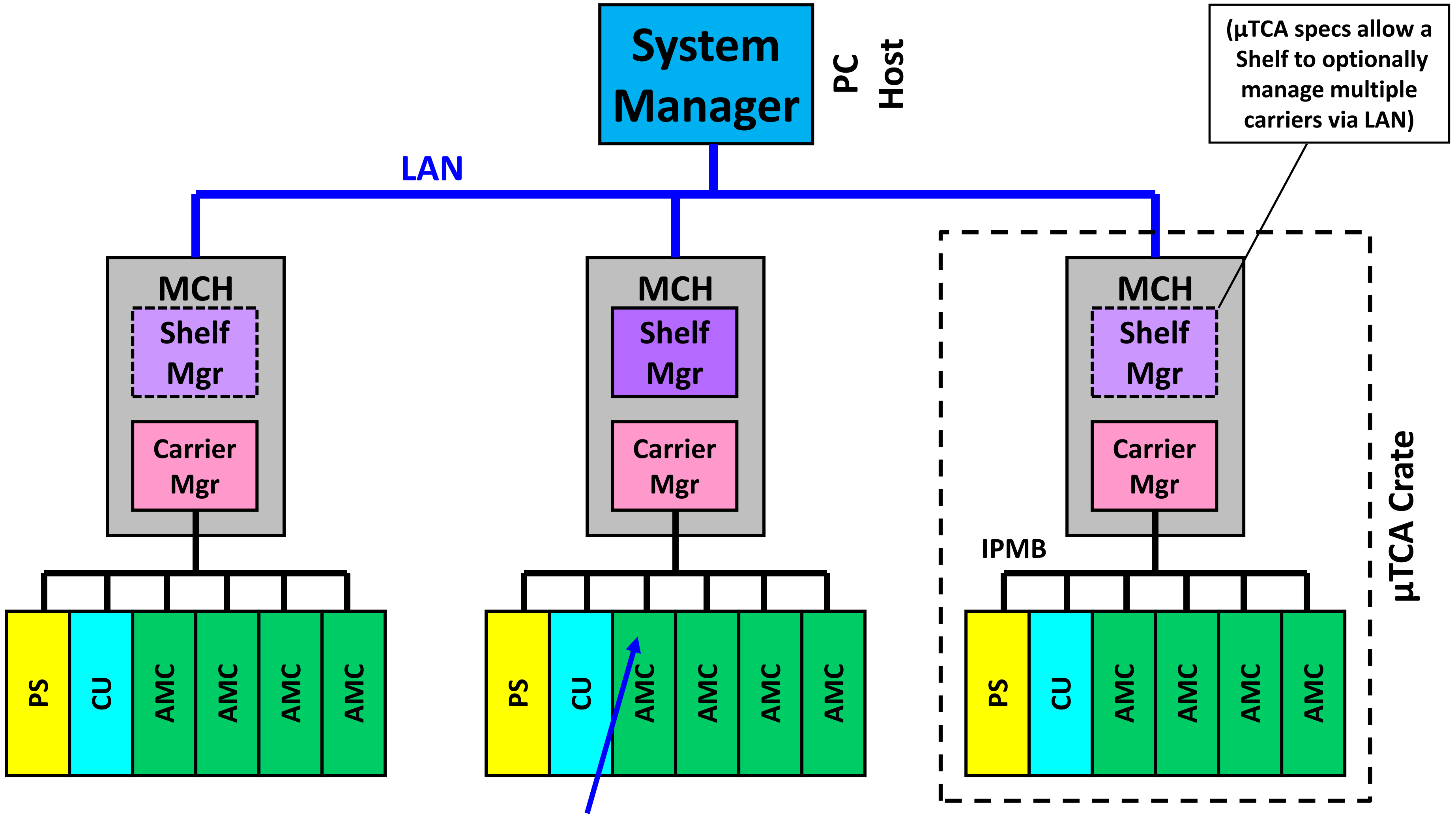
The Starting Point

- In the CMS Upgrade, MicroTCA crates replace VME crates
- MicroTCA prominently features Gigabit Ethernet to address individual slots and IPMI to manage the power and sensors of individual cards
- We undertook a targeted R&D effort to best understand how to interface to the IPMI system and use it to best advantage to operate and maintain our crates in Layer 1 of the CMS Calorimeter Trigger upgrade
- Development of the system computer-based component, known as the *System Manager* in IPMI terminology, is the subject of this talk

Some Useful Definitions

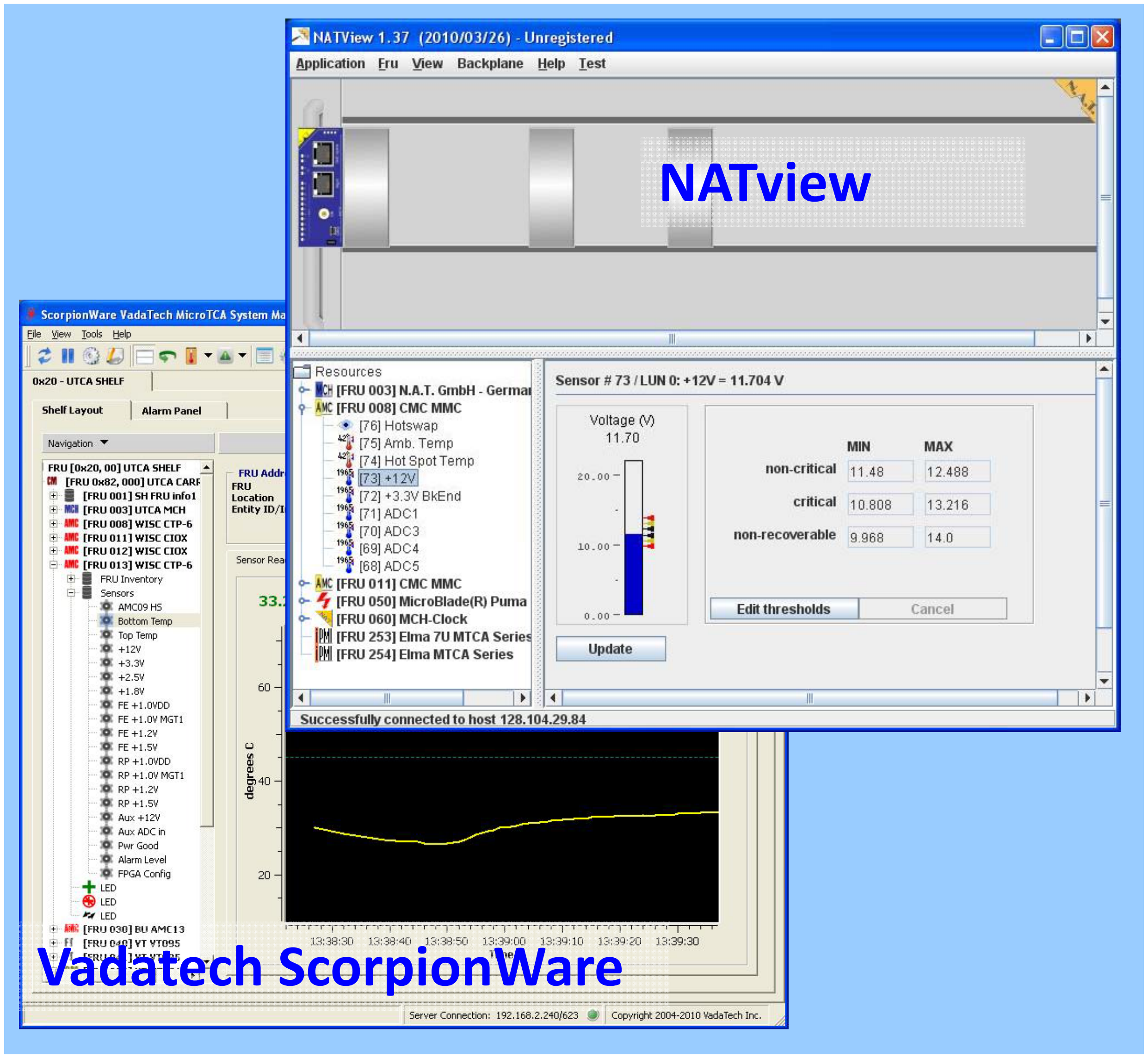
- **Intelligent Platform Management Interface (IPMI) – mechanism for management, monitoring and logging of elements in an electronics system**
- **Advanced Mezzanine Card (AMC) – circuit board conforming to an industry form factor, with a 170-pin backplane connector**
- **MicroTCA – form factor for operating AMCs directly on a backplane**
- **MicroTCA Carrier Hub (MCH) – Module performing the carrier functions of the MicroTCA crate, including power distribution & regulation, platform management, fabric connectivity, clock distribution**
- **Module Management Controller (MMC) – management controller on AMCs which interface to the Carrier Manager on the MCH**
- **Field Replaceable Unit (FRU) – replaceable module in the MicroTCA crate, including power modules (PMs), cooling units (CUs), AMCs, and MCHs**
- **Carrier Manager – logical function on MCH that manages the AMCs, PMs and CUs through the Intelligent Platform Management Bus (IPMB)**
- **Shelf Manager – entity for routing messages between the System Manager and Carrier Manager, provides access to system repositories and manages cooling**
- **System Manager – level of management above the Shelf Manager, whatever that may mean in a specific application**

IPMI Hierarchical Mgmt Model (MicroTCA Perspective)



Module Management Controller (MMC) in each Advanced Mezzanine Card (AMC)

Example System Managers from MCH Vendors



Resources
[FRU 003] N.A.T. GmbH - German
[FRU 008] CMC MMC
[76] Hotswap
[75] Amb. Temp
[74] Hot Spot Temp
[73] +12V
[72] +3.3V BkEnd
[71] ADC1
[70] ADC3
[69] ADC4
[68] ADC5
[FRU 011] CMC MMC
[FRU 050] MicroBlade(R) Puma
[FRU 060] MCH-Clock
[FRU 253] Elma 7U MTCA Series
[FRU 254] Elma MTCA Series

Sensor # 73 / LUN 0: +12V = 11.704 V

Voltage (V) 11.70

	MIN	MAX
non-critical	11.48	12.488
critical	10.808	13.216
non-recoverable	9.968	14.0

Successfully connected to host 128.104.29.84

FRU [0x20, 00] UTCA SHELF
[FRU 0x82, 000] UTCA CARR
[FRU 001] SH FRU info1
[FRU 003] UTCA MCH
[FRU 008] WISC CTP-6
[FRU 011] WISC CIOX
[FRU 012] WISC CIOX
[FRU 013] WISC CTP-6
FRU Inventory
Sensors
AMC09 HS
Bottom Temp
Top Temp
+12V
+3.3V
+2.5V
+1.8V
FE +1.0VDD
FE +1.0V MGT1
FE +1.2V
FE +1.5V
RP +1.0VDD
RP +1.0V MGT1
RP +1.2V
RP +1.5V
Aux +12V
Aux ADC in
Pwr Good
Alarm Level
FPGA Config
LED
LED
LED
[FRU 030] BU AMC13
[FRU 040] VT VT095

FRU Addr
FRU Location
Entity ID/I
Sensor Rea
33.2

Server Connection: 192.168.2.240/623
Copyright 2004-2010 VadaTech Inc.

- Offerings from two MCH Vendors: Vadatech and NAT
- Primarily GUI-based viewers of sensors, events and board info
- Practical observation: they do not always handle changes to crate hardware gracefully
- Single-MCH-platform applications (vendor specific)

Background & Motivations

- Per specs each AMC card needs a MMC to negotiate for +12V payload power, so we wrote an MMC
  - Based on Atmel AVR32 microcontroller
  - Runs on +3.3V, 150mA Management Power supply to AMC
- Realized that with some extensions, the geographic addressing used by MMCs could become a useful mechanism for IP address assignment to the slots
  - Current VME hardware has address by slot
  - MicroTCA Spec has a GbE connection from the MCH to each AMC card (referred to as "Fabric A")
  - As GbE replaces VME in MicroTCA, wanted to transfer this slot-dependent address property to the new hardware
  - Prefer to see this done *automatically*
- Viewing the situation more broadly, since IPMI is operational on an AMC card before the payload is active, the IPMI path provides an opportunity to control the specifics of the FPGA boot (e.g., image select)

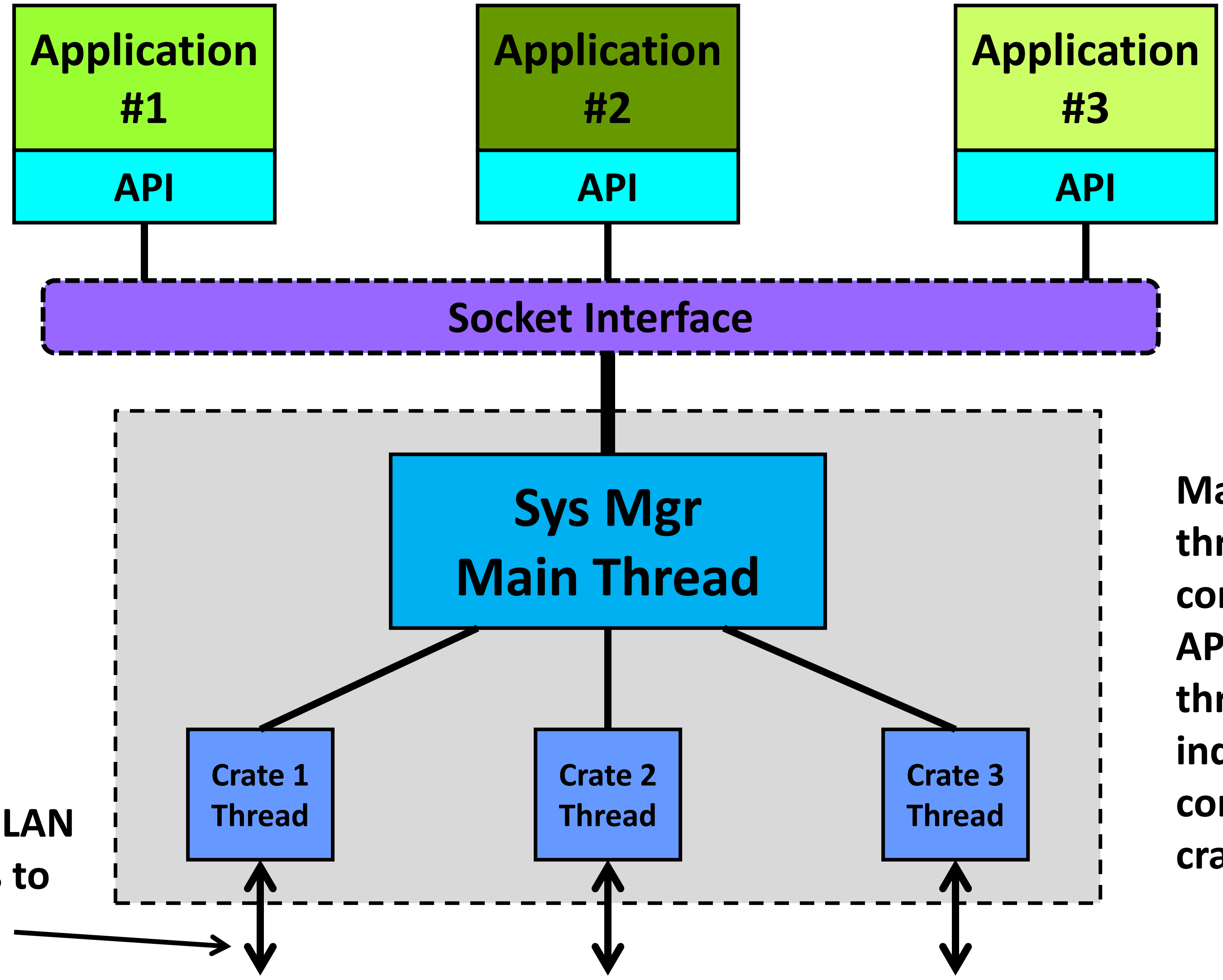
Background & Motivations, cont'd

- Sensors—seemed logical to utilize the existing IPMI infrastructure for new and traditional sensors
  - Use new sensors to support the configuration process
  - Make it straightforward to tie electrical and thermal sensor data into DCS or some other appropriate monitoring system
- In the course of our work, things were learned:
  - MCHs support a limited number of LAN sessions (about 5)
  - There is a variable lag time between physical card events in the crate and the reporting of the updated state by the Shelf Manager. For example, sensor events can show up in the SEL before the sensors show up in the updated SDR
  - Ordering of sensors in the Shelf Manager's SDR is circumstance-dependant. Accessing sensors by crate #, FRU, and sensor-name is more useful to end-users

Design Objectives

1. **Act as a gateway for other applications into the IPMI system**
   - **Socket interface encapsulated by an API**
   - **Provide access path for IPMI commands and sensor operations**
   - **Map sensors by crate / FRU / sensor-name**
2. **Provide a robust IPMI-over-LAN connection to crates**
   - **Gracefully handle configuration changes and the MCH response to them**
   - **Encapsulates issues associated with LAN connections to the MCHs (e.g., crate power-cycling)**
3. **Provide a service for *automatic* low-level initialization of AMC cards at startup**

System Manager Block Diagram



Main server thread for connections via API, separate threads for individual connections on crate LAN

192.168.x.x LAN connections to crate Shelf Managers

System Manager Key Features

- Three main API classes of functions:
  - Sensor Event callback facility
  - Sensor Read/Management interface
  - Send Message delivery path
- API uses socketed connections to System Manager
- Separate crate threads for robust multi-crate support
- Automatic configuration service runs in System Manager core process
  - Detects hot swap insertion and removal of new AMC cards from the crates as quickly as the MCH will allow
  - Delivers a low-level configuration record to those AMCs that request it
  - Goal is to make primary configuration of the AMCs fully automatic at post-insertion of card

Sensor API

- Used for sensors and basic sensor management
  - Functionality set by the mandatory MMC supported-command list, per AMC.0 R2.0 3.15.2
- Provides a reliable & deterministic mapping of sensor addresses by (crate, FRU, sensor_name)
  - Carrier & Shelf Managers on MCHs concatenate sensor information from individual FRUs, with the ordering being dependent on circumstances
  - System Manager interprets the reported sensor mapping from the MCHs to provide transparent addressing at the API
  - Sensors must have unique names on each AMC

Event API

- Mechanism for responding to sensor events
- Register event filters with the System Manager
- Register callback functions associated with each filter
- If System Manager sees events in the Sensor Event Logs (SELs) of the crate MCHs that correspond to the filter, it relays the events to the appropriate connection
- API receives the event notification on its socket connection to the System Manager and executes the registered callback function

Send Message API Function

- Interface for sending raw IPMI commands to FRUs in connected crates
- Similar to the “raw” mode of the IPMItool command line interface
- Avoids the session limits associated with the LAN controllers in the MCHs
- Useful for applications that utilize custom commands in the MMCs
- Uses same base addressing scheme as Sensor & Event access

3 Connection Example

1. **Monitoring Connection**
   - Reads out and logs a list of sensors at a regular interval (e.g., 1 minute)
   - Runs as a “background” activity
2. **GUI or Command-Line Control Connection**
   - Supports actions associated with the controls of GUI or command line interface
   - Could be some combination of sensor reading and raw commands sent to FRUs
3. **Event Notification Connection**
   - Calls specific user functions in response to filtered sensor events

2-Crate Auto-Config Demo: IP Address Assignment

Example AMC Card Fabric A IP Address Assignment: 192.168.<crate#>.<40+slot#>

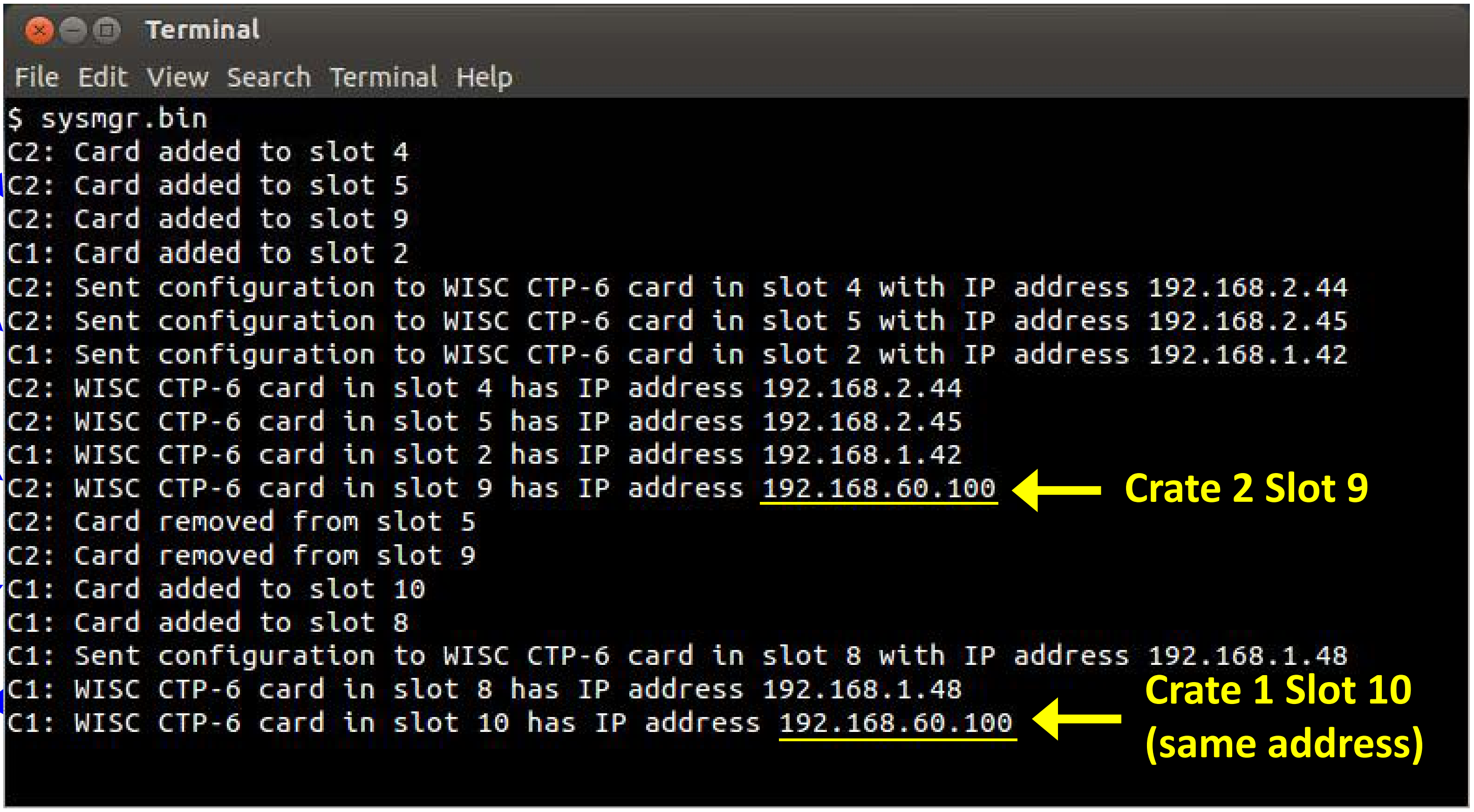
Startup:
Crate 2: 3 cards
Crate 1: 1 card

Send IP address via IPMI to 3 cards (note 3rd octet)

4th card IP address set from MMC EEPROM, read by Sys Mgr

2 Cards moved from Crate 1 to Crate 2

Card in Crate 1 Slot 8 gets new IP address, other card keeps addr from EEPROM



```
$ sysmgr.bin
C2: Card added to slot 4
C2: Card added to slot 5
C2: Card added to slot 9
C1: Card added to slot 2
C2: Sent configuration to WISC CTP-6 card in slot 4 with IP address 192.168.2.44
C2: Sent configuration to WISC CTP-6 card in slot 5 with IP address 192.168.2.45
C1: Sent configuration to WISC CTP-6 card in slot 2 with IP address 192.168.1.42
C2: WISC CTP-6 card in slot 4 has IP address 192.168.2.44
C2: WISC CTP-6 card in slot 5 has IP address 192.168.2.45
C1: WISC CTP-6 card in slot 2 has IP address 192.168.1.42
C2: WISC CTP-6 card in slot 9 has IP address 192.168.60.100
C2: Card removed from slot 5
C2: Card removed from slot 9
C1: Card added to slot 10
C1: Card added to slot 8
C1: Sent configuration to WISC CTP-6 card in slot 8 with IP address 192.168.1.48
C1: WISC CTP-6 card in slot 8 has IP address 192.168.1.48
C1: WISC CTP-6 card in slot 10 has IP address 192.168.60.100
```

Crate 2 Slot 9

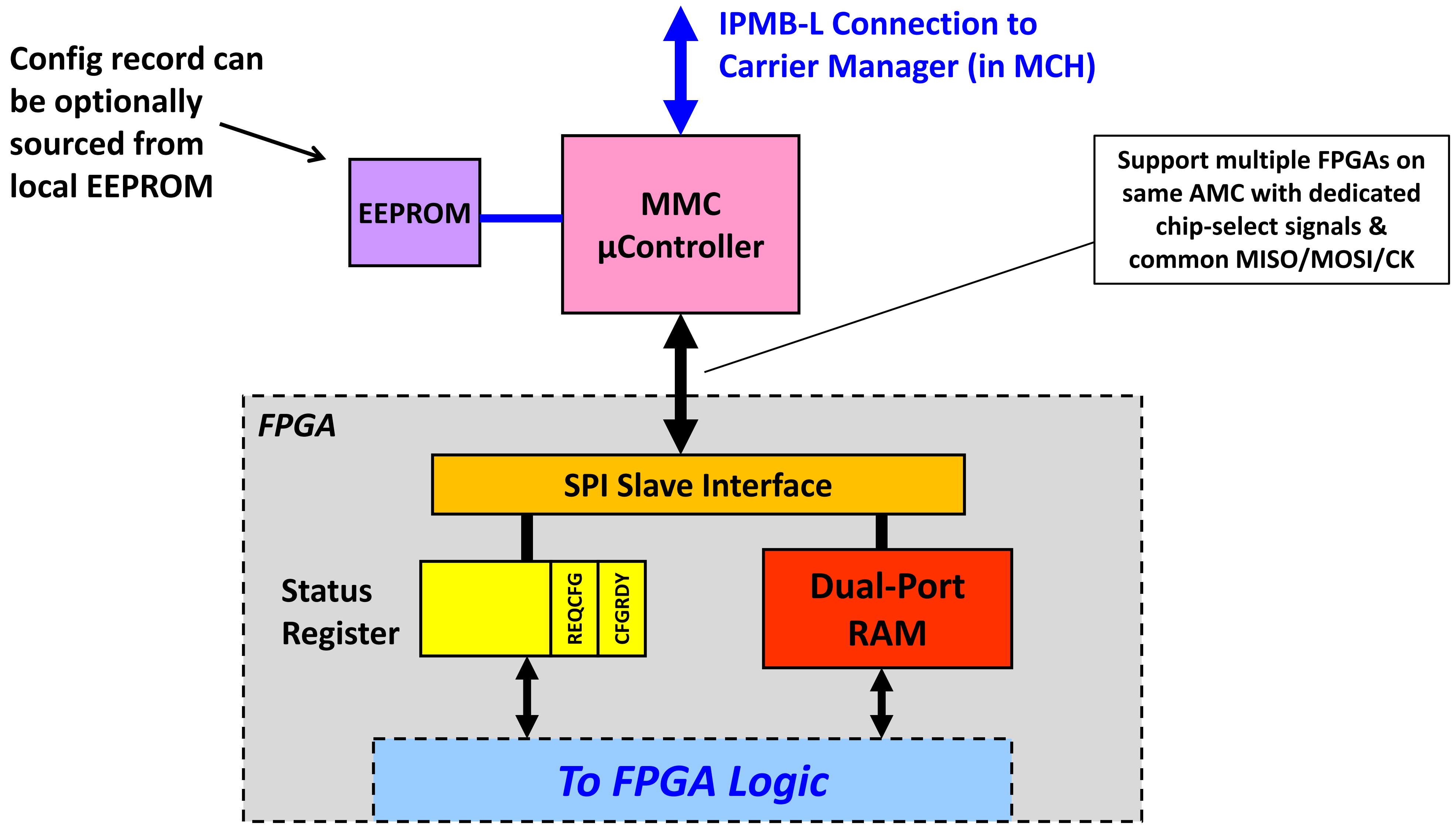
Crate 1 Slot 10 (same address)

New “FPGA Config” IPMI Sensor

Offset	Function
0	FPGA Load Done
1	(Reserved)
2	SPI Port 0 Slave Detect
3	SPI Port 0 Request Cfg
4	SPI Port 0 Cfg Ready
5	SPI Port 1 Slave Detect
6	SPI Port 1 Request Cfg
7	SPI Port 1 Cfg Ready
8	SPI Port 2 Slave Detect
9	SPI Port 2 Request Cfg
10	SPI Port 2 Cfg Ready

- Sensor Type: C0h (OEM Reserved)
- Event Type: 6Fh (Sensor Specific)
- Sensor Name: “FPGA Config”
- Supports up to 3 FPGAs
- Slave Detect—indicates that a SPI slave supporting the config state machine is present & loaded in FPGA
- Request Config—set by the FPGA at initialization, requesting configuration info through the SPI path (low volume, e.g., 10-100 bytes)
- Config Ready–set by the initializing agent (MMC or Sys Mgr) after delivering config payload

FPGA-MMC SPI Config Interface Implementation Example


IPMB-L Connection to Carrier Manager (in MCH)
Config record can be optionally sourced from local EEPROM
EEPROM
MMC µController
Support multiple FPGAs on same AMC with dedicated chip-select signals & common MISO/MOSI/CK
FPGA
SPI Slave Interface
Status Register
REQCFG
CFGRDY
Dual-Port RAM
To FPGA Logic

Config Record and new IPMI Commands

Configuration Record

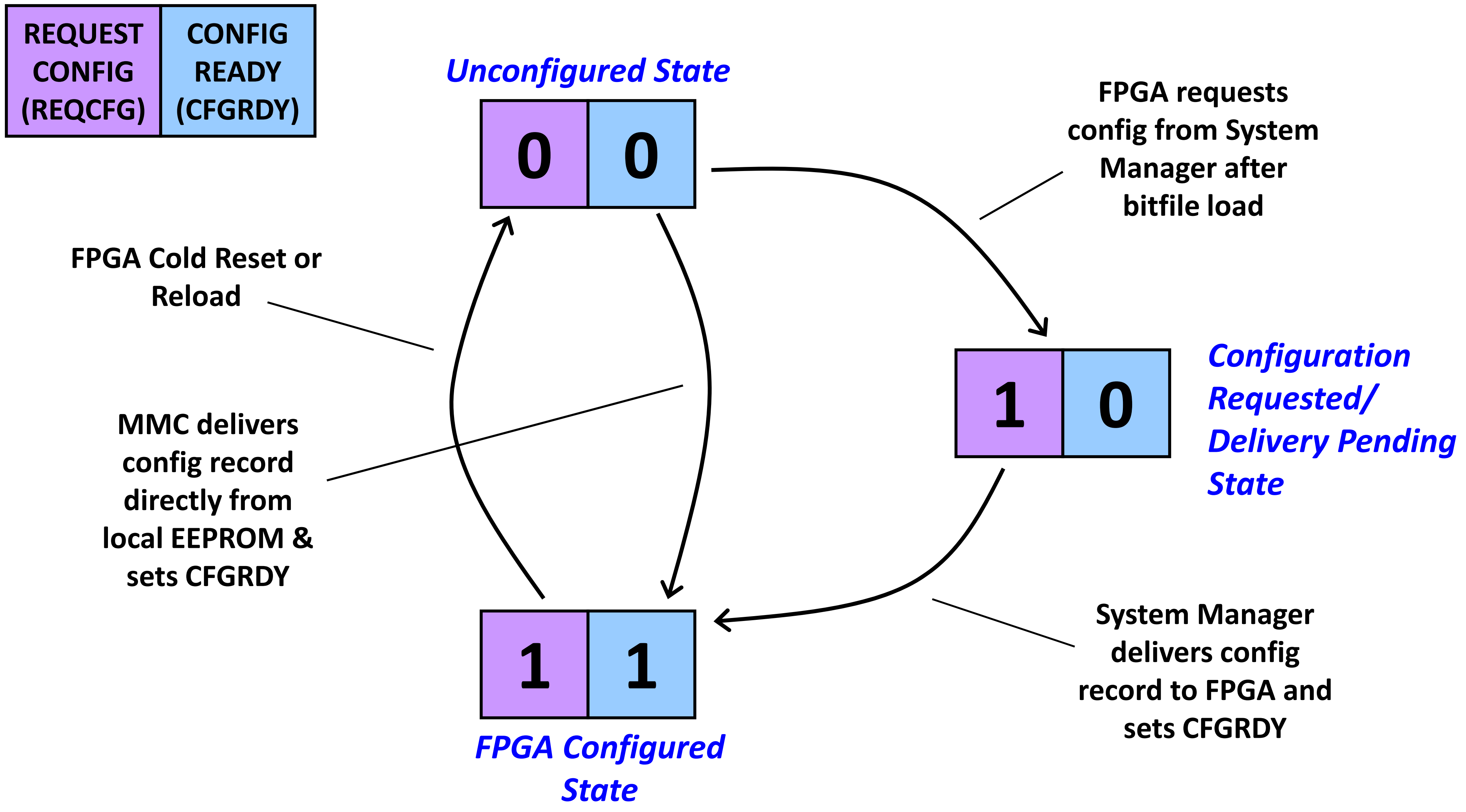
Byte Address	Contents
0	Reserved (Slot ID)
1-4	IPv4 Subnet Mask
5-9	IPv4 IP Address
10-*n*	Application-specific

New MMC-supported IPMI Commands

Write_FPGA_SPI_Data
Read_FPGA_SPI_Data
Write_FPGA_SPI_Register
Read_FPGA_SPI_Register

- **New IPMI commands in MMCs needed to support the SPI interface to the FPGAs**
  - **Can be installed in new netFn command space (e.g., 32h/33h)**
- **Common config record and IPMI command formats for all AMCs simplifies interoperability**
- **IP address info for Fabric A the most obvious application, but others may arise**

FPGA Configuration State Diagram (FPGA Config sensor)



REQUEST CONFIG (REQCFG)
CONFIG READY (CFGRDY)
Unconfigured State
0 0
FPGA requests config from System Manager after bitfile load
1 0
Configuration Requested/ Delivery Pending State
System Manager delivers config record to FPGA and sets CFGRDY
1 1
FPGA Configured State
FPGA Cold Reset or Reload
MMC delivers config record directly from local EEPROM & sets CFGRDY

Geographic Address Assignment

- Preserves useful feature of current VME crates (slot-specific address)
- After loading its programming, FPGA sets REQCFG bit, relayed by MMC to System Manager via “FPGA Config” IPMI sensor
- System Manager transfers configuration record associated with that crate/slot and sets CFGRDY bit in interface
  - Config Records for the individual slots can be stored in master file or some other central resource available to System Manager
- FPGA completes its initialization, including GbE interface on Fabric A
- Automatic service built into System Manager

Non-Geographic IP Address Assignment

- **Two methods available—MMC EEPROM sourced, or System Manager sourced**
  - **Use same basic mechanisms as geographic method**
- **For EEPROM-sourced, MMC detects the REQCFG signal in the FPGA interface and transfers config record from its EEPROM and sets CFGRDY**
  - **System Manager sees that configuration is complete, takes no further action**
- **System Manager-sourced—**
  - **FPGA writes config key to SPI memory**
  - **System Manager reads key, locates matching config record from file & completes transfer**

Development Status & Plans

- Core crate threads (the hard part) working with Vadatech UTC002 MCH
  - MCH support: definitely Vadatech and hopefully NAT (some open issues we are working through)
  - Took some time to find an open-source IPMI library that was a good fit for what we're doing
- Basic API architecture established, completing definitions & working on implementation
- Targeting first version complete with API by end of May
- Plan to deploy to our test setup in the CMS Electronics Integration Center later in 2013

Summary

- Undertook a targeted R&D effort to determine how best to use IPMI in our MicroTCA crates in the CMS upgrade
- Developed a MMC to act as a platform for custom features, such as the config sensor and FPGA SPI connection to support low-level AMC initialization at startup
- Developed a System Manager with the capability to automatically recognize & initialize AMCs in crate startup & AMC hot swap situations
- Developing a socketed API for the System Manager to allow it to operate as a communications layer, providing IPMI access to remote applications
  - Insulates those remote applications from the idiosyncrasies of individual MCHs, particularly how they respond to hot swap changes to the crate hardware
- Being deployed this year