System Firmware Progress Sensor for CP3260 Specification

1. Introduction

IPMI v1.5 specification defines a new sensor called 'System Firmware Progress'. The purpose of this sensor is to model the firmware running on the payload and provide various states to the external management software(e.g ShMM) using standard IPMI event mechanism. The firmware states of interest includes 'Progress', 'Hang' and 'Error' with various sub-states as defined in the specification. The purpose of this document is to describe the modelling of 'System Firmware Progress' sensor for CP3260 as requested per ALu RFE#6680276

2. Description

2.1 CP3260 Firmware Description

CP3260 board includes the following various firmware components and there is a need to clearly identify which of these firmware components shall be modelled for 'System Firmware Progress' sensor.

2.1.1 IPMC Firmware

This firmware runs on a dedicated micro controller H8 and provides the PICMG 3.x standard Management Controller functionality for the ATCA board CP3260.

2.1.2 SP Firmware

This firmware includes ILOM, vBSC components and runs on a dedicated PowerPC processor and provides Sun specific Service Processor functionality for the main payload ultraSparc T2 processor.

2.1.3 Host Firmware

This firmware includes POST, HV, OpenBoot components and runs on ultraSparc T2 processor and provides environment for running multiple guest Operating Systems including Solaris/Linux etc..

Since the payload for CP3260 is ultraSparc T2 which runs customer applications, it is proposed to model the host firmware components(POST, HV, OpenBoot) for 'System Firmware Progress' sensor.

2.2 System Firmware State Functional Requirements

IPMIv1.5 specification defines various states for the 'System Firmware Progress' sensor with sub-states as listed in Appendix B. Since most of the states are x86/BIOS centric, for the purpose of this RFE, a smaller subset as applicable to CP3260 is requested as below.

Typical firmware initialization sequence starts with 'Primary Processor Initialization' followed by 'Memory Initialization' followed by 'OS Boot Process'. Each of these progress states will be reported using 'system firmware progress' sensor states. At each of phase, the firmware is monitored for its progress and any hangs that are detected using timers will be reported using 'system firmware hang' sensor states. Any memory errors detected during Memory Initialization phase will be reported using 'system firmware error' sensor states.

More detailed information about each of the states(definition, detection and propagation) can be found in Section 2.4

2.2.1 System Firmware Error(sensor specific offset: 00h)

Unspecified.(evt data2: 00h)

- Indicates that ultra Sparc T2 POST detected failure(s) for some of the device(s) tested.

No system memory is physically installed in the system.(evt data2: 01h)

- Indicates that there is no system memory physically installed in CP3260. Typically, CP3260 supports up to 16GB system memory.

No usable system memory, all installed memory has experienced an unrecoverable failure.(evt data2: 02h)

- Indicates the CP3260 does not have any usable system memory and/or all installed memory has experienced unrecoverable failure. Typically, CP3260 supports up to 16GB system memory.

2.2.2 System Firmware Progress(sensor specific offset: 02h)

Primary processor initialization(event data2: 19h)

- Indicates firmware is progressing properly and has successfully completed primary processor(e.g core0) initialization.

Memory initialization.(evt data2: 01h)

- Memory initialization is done in 2 phases with vBSC doing memory controller initialization and HV doing the memory scrubbing. The definition for this state is to indicate that both of these phases are done.

Starting operating system boot process, e.g. calling Int 19h(evt data2: 13h)

- Indicates that firmware is progressing properly and is starting Operating System boot process.

2.2.3 System Firmware Hang(sensor specific offset:01h)

Primary processor initialization(evt data2: 19h)

- Indicates that firmware hung while initializing primary processor(e.g core 0)

Memory initialization(evt data2: 01h)

- Memory initialization is done in 2 phases with vBSC doing memory controller initialization and HV doing the memory scrubbing. The definition for this state is to indicate a hang in either of these phases.

Starting operating system boot process, e.g. calling Int 19h(evt data2: 13h)

- Indicates that firmware is hung and is unable to start operating system boot process.

2.3 Design Proposal

After looking at the existing CP3260 system firmware architecture and the required firmware states, it is proposed to leverage existing mechanism for detection and propagation of various firmware states from firmware components to external System Management Software.

2.3.1 Firmware State Detection

During ultraSparc T2 poweron, vBSC closely interacts with the host firmware(POST, HV, OpenBoot) and has knowledge about various firmware states that are interested for this RFE. So it is decided to use this information for modelling the firmware states.

2.3.2 Firmware State Propagation

Per IPMI specification, the Management Controller(e.g IPMC), maintains the 'System Firmware Progress' sensor and sends out standard IPMI events to chassis wide BMC(e.g ShMM) using this sensor whenever associated transitions happens. So the firmware state information has to be propagated from vBSC to IPMC thru ILOM. The following is the sequence of information flow.

- vBSC detects the host firmware state change
- vBSC sends this information to ILOM using OEM protocol(TBD protocol)
- ILOM sends this to IPMC via Serial port using IPMI OEM Command(TBD command)
- IPMC sends standard IPMI event to ShMM using the 'System Firmware Progress' sensor

2.4 Implementation Proposal

In this section an attempt is made to describe various firmware states as applicable for CP3260 in detail especially the definition of each state, how that state is detected and how the state information is propagated to ShMM. The description follows a typical firmware initialization sequence starting with 'Primary Processor Initialization' followed by 'Memory Initialization' followed by 'OS Boot Process'. Each of these progress states will be reported using 'system firmware progress' sensor states. At each of phase, the firmware is monitored for its progress and any hangs that are detected using timers will be reported using 'system firmware hang' sensor states. Any memory errors detected during Memory Initialization phase will be reported using 'system firmware error' sensor states.

2.4.1 System Firmware Progress(Primary Processor Initialization)

Definition:

Indicates firmware is progressing properly and has successfully completed primary processor(e.g core0) initialization.

Detection:

vBSC has the information when the primary processor has been successfully initialized.

Propagation:

After successfully initializing primary processor(e.g core 0), vBSC sends out a message to ILOM which will be forwarded to IPMC which will generate standard IPMI event to ShMM.

2.4.2 System Firmware Hang(Primary Processor Initialization)

Definition:

Indicates that firmware hung while initializing primary processor(e.g core 0)

Detection:

vBSC initiates a timer before starting the primary processor initialization and if timer goes off before the primary processor initialization is done properly, a hang is said to be detected during primary processor initialization phase. The time out value is TBD secs.

Propagation:

If vBSC detects any hang during this phase of Primary Processor(e.g core 0) Initialization, it will send out a message to ILOM which will forward to IPMC which will generate standard IPMI event to ShMM.

2.4.3 System Firmware Progress(Memory Initialization)

Definition:

Memory initialization is done in 2 phases with vBSC doing memory controller initialization and HV doing the memory scrubbing. The definition for this state is to indicate that both of these phases are done.

Detection:

HV sends a message(TBD message) to vBSC after finishing the memory scrubbing and vBSC uses this to determine this firmware progress state.

Propagation:

After getting message from HV about successful memory scrubbing, vBSC sends out a message to ILOM which will be forwarded to IPMC which will generate standard IPMI event to ShMM.

2.4.4 System Firmware Hang(Memory Initialization)

Definition:

Memory initialization is done in 2 phases with vBSC doing memory controller initialization and HV doing the memory scrubbing. The definition for this state is to indicate a hang in either of these phases.

Detection:

vBSC initiates a timer before starting the memory initialization and if timer goes off before the memory initialization is done properly, a hang is said to be detected during memory initialization phase. The time out value is TBD secs.

Propagation:

If vBSC detects any hang during the memory initialization phase, it sends out message to ILOM which will be forwarded to IPMC which will generate standard IPMI event to ShMM.

2.4.5 System Firmware Error(No system memory is physically installed in the system)

Definition:

Indicates that there is no system memory physically installed in CP3260. Typically, CP3260 supports up to 16GB system memory.

Detection:

Since CP3260 is a virtualized system, the system memory is tested and internal ASR database is adjusted accordingly. vBSC has the information about whether the system memory is present or not.

Propagation:

If vBSC finds no system memory is installed, it sends out message to ILOM which will be forwarded to IPMC which will generate standard IPMI event to ShMM.

2.4.6 System Firmware Error(No usable system memory, all installed memory has experienced an unrecoverable failure)

Definition:

Indicates the CP3260 does not have any usable system memory and/or all installed memory has experienced unrecoverable failure. Typically, CP3260 supports up to 16GB system memory.

Detection:

Since CP3260 is a virtualized system, the System Memory is tested and internal ASR database is adjusted accordingly. vBSC uses this database to findout about the usable system memory.

Propagation:

If vBSC finds no usable memory, it sends out message to ILOM which will be forwarded to IPMC which will generate standard IPMI event to ShMM.

2.4.7 System Firmware Error(Unspecified)

Definition:

Indicates that ultra Sparc T2 POST detected failure(s) for some of the device(s) tested.

Detection:

Once POST is completed, it informs vBSC about the results, hence vBSC will be able to identify/detect any failure(s) during POST run.

Propagation:

When POST sends the results, vBSC examines and in case of any failure(s), it sends out a message to ILOM which will be forwarded to IPMC which will generate standard IPMI event to ShMM.

2.4.8~~7~~ System Firmware Progress(Starting Operating System boot process)

Definition:

Indicates that firmware is progressing properly and is starting Operating System boot process.

Detection:

vBSC gets an event from OpenBoot before starting the guest OS and is aware of this state.

Propagation:

When OpenBoot sends the state change to indicate OS boot is started, vBSC sends out a message to ILOM which will be forwarded to IPMC which will generate standard IPMI event to ShMM.

2.4.9~~8~~ System Firmware Hang(starting Operating System boot process)

Definition:

Indicates that firmware is hung and is unable to start operating system boot process.

Detection:

vBSC is notified by OpenBoot about 'starting OS boot process'. After both the memory and primary processor are initialized, if vBSC does not receive this event from OpenBoot within TBD secs, vBSC assumes the firmware is hung to 'start OS boot process'.

Propagation:

Upon detecting firmware hang to 'start OS boot process', vBSC sends out event to ILOM which will be forwarded to IPMC which will generate standard IPMI event to ShMM.

3 Use Cases

This section explains various use cases where the 'system firmware progress sensor' events are expected.

3.1 Shelf Power ON

During Shelf Power ON, the boards are activated(i.e powered up) by ShMM one board at a time and at this time, the CP3260 board powers up and the firmware goes thru the initialization process including memory checks, primary processor initialization, memory initialization and start OS boot process etc.. During this firmware initialization process any kind of errors, hangs, progress states are propagated using the 'system firmware progress sensor' as explained above. System Management Software running on ShMM will be receiving these events as standard IPMI events.

3.2 CP3260 Power ON

CP3260 board can be individually powered on by using ATCA shelf manager commands (e.g clia activate) after a previous command to power off(e.g clia deactivate). During this time, the board powers up and the firmware goes thru the initialization process including memory checks, primary processor initialization, memory initialization and start OS boot process etc.. During this firmware initialization process any kind of errors, hangs, progress states are propagated using the 'system firmware progress sensor' as explained above. System Management Software running on ShMM will be receiving these events as standard IPMI events.

3.3 CP3260 Hot Insertion

When the CP3260 board is hot inserted in to the chassis, the board powers up and the firmware goes thru the initialization process including memory checks, primary processor initialization, memory initialization and start OS boot process etc.. During this firmware initialization process any kind of errors, hangs, progress states are propagated using the 'system firmware progress sensor' as explained above. System Management Software running on ShMM will be receiving these events as standard IPMI events.

3.4 Payload Hard Reset

During Watch Dog Timer expiry and/or based on ShMM command(Frucontrol command with Frucontrol option as Cold Reset-00h), IPMC may initiate payload hard reset which will reset ultraSparc T2 processor. In this case, the firmware goes thru the initialization process including memory checks, primary processor initialization, memory initialization and start OS boot process etc.. During this firmware initialization process any kind of errors, hangs, progress states are propagated using the 'system firmware progress sensor' as explained above. System Management Software running on ShMM will be receiving these events as standard IPMI events.

3.5 Payload Reboot

During OS reboot scenario, the OS will simply shutdown and comeback up through OpenBoot. In this case, the firmware does not go through the steps of memory initialization, primary processor initialization etc.. In this case the only states that will be sent thru 'system firmware progress sensor' are relating to 'starting OS boot process'. System Management Software running on ShMM will be receiving these events as standard IPMI events.

3.6 Graceful OS Reboot

If ShMM issues a command to graceful reboot(e.g Frucontrol command with Fru control option as Graceful Reboot-02h), the OS will simply shutdown and comeback up through OpenBoot. In this case, the firmware does not go through the steps of memory initialization, primary processor initialization etc.. In this case the only states that will be sent thru 'system firmware progress sensor' are relating to 'starting OS boot process'. System Management Software running on ShMM will be receiving these events as standard IPMI events.

3.7 OS Shutdown

Solaris OS can be shutdown using various methods(e.g issuing break command from ShMM, issuing break sequence on console, using OS provided command like shutdown). In this case the OS can be brought down to OpenBoot's ok prompt. At this time, there will not be any 'system firmware progress sensor' event. And the system can remain in this state until user tries to boot the Operating System. At the time the OpenBoot starts OS boot process, an IPMI event will be sent out to ShMM with 'system firmware progress sensor' corresponding to state 'starting OS boot process'. System Management Software running on ShMM will be receiving these events as standard IPMI events.

Appendix A

Abbreviations

IPMI	Intelligent Platform Management Interface
ATCA	Advanced Telecommunications Computing Architecture
SP	Service Processor
ILOM	Integrated Lights Out Management
vBSC	Virtual Blade System Controller
POST	Power On Self Test
HV	Hyper Visor
OpenBoot	Open Boot Prom(a.k.a OBP)
BIOS	Basic Input Output System
OEM	Open Equipment Manufacturer
IPMC	Intelligent Platform Management Controller
BMC	Baseboard Management Controller
ShMM	Shelf Management Mezzanine

Appendix B

System Firmware Progress Sensor States per IPMI v1.5

System Firmware Error

Unspecified.

No system memory is physically installed in the system.

No usable system memory, all installed memory has experienced an unrecoverable failure.

Unrecoverable hard-disk/ATAPI/IDE device failure.

Unrecoverable system-board failure.

Unrecoverable diskette subsystem failure.

Unrecoverable hard-disk controller failure.

Unrecoverable PS/2 or USB keyboard failure.

Removable boot media not found

Unrecoverable video controller failure

No video device detected

Firmware (BIOS) ROM corruption detected

CPU voltage mismatch (processors that share same supply have mismatched voltage requirements)

CPU speed matching failure

System Firmware Hang

Unspecified.

Memory initialization.

Hard-disk initialization

Secondary processor(s) initialization

User authentication

User-initiated system setup

USB resource configuration

PCI resource configuration

Option ROM initialization

Video initialization

Cache initialization

SM Bus initialization

Keyboard controller initialization

Embedded controller/management controller initialization

Docking station attachment

Enabling docking station

Docking station ejection

Disabling docking station

Calling operating system wake-up vector

Starting operating system boot process, e.g. calling Int 19h

Baseboard or motherboard initialization

reserved

Floppy initialization

Keyboard test

Pointing device test

Primary processor initialization

System Firmware Progress

Unspecified.

Memory initialization.

Hard-disk initialization

Secondary processor(s) initialization

User authentication

User-initiated system setup

USB resource configuration

PCI resource configuration

Option ROM initialization

Video initialization

Cache initialization

SM Bus initialization

Keyboard controller initialization

Embedded controller/management controller initialization

Docking station attachment

Enabling docking station

Docking station ejection

Disabling docking station

Calling operating system wake-up vector

Starting operating system boot process, e.g. calling Int 19h

Baseboard or motherboard initialization

reserved

Floppy initialization

Keyboard test

Pointing device test

Primary processor initialization