

Technical Disclosure Commons

Defensive Publications Series

September 26, 2016

A method and apparatus for faster, reliable and consolidated logging of information during system failure

Clarete Riana Crasta

Sandeep S Yelandur

Suhas Shivanna

Follow this and additional works at: http://www.tdcommons.org/dpubs_series

Recommended Citation

Crasta, Clarete Riana; Yelandur, Sandeep S; and Shivanna, Suhas, "A method and apparatus for faster, reliable and consolidated logging of information during system failure", Technical Disclosure Commons, (September 26, 2016)
http://www.tdcommons.org/dpubs_series/281



This work is licensed under a [Creative Commons Attribution 4.0 License](https://creativecommons.org/licenses/by/4.0/).

This Article is brought to you for free and open access by Technical Disclosure Commons. It has been accepted for inclusion in Defensive Publications Series by an authorized administrator of Technical Disclosure Commons.

A method and apparatus for faster, reliable and consolidated logging of information during system failure


Abstract

For servers today, that run mission critical workloads, downtime is not an option and any outage of these servers usually translates to reduced revenue, reduced profitability and potential customer loss. Any interruption in the operation or availability of these workloads will have a ripple effect throughout the organization. Gathering valid and necessary data about the event of failure from all possible sources plays a significant role in determining how quickly and accurately the root cause for the server down-time is identified. The data required for such analysis is spread across Firmware and Operating System (OS) and comes from different sources on the server. This information comprises of data collected and logged by the firmware such as the error log buffers, event logs and also the state of the system at the time of failure, collected by the operating systems in the core dump files. Most often the challenge faced is with collection of the set of interdependent information originating and stored at different locations on the system. The proposed solution enables a high availability design by eliminating single point of failure during the log collection and retrieval process. This disclosure proposes a method and apparatus for faster, reliable and consolidated logging of necessary data from different sources on occurrence of a system failure.

Problem Statement

System error detection, is a critical element of a highly reliable and fault tolerant computing environment. Error detection and analysis is aided by the data gathered across system firmware and operating system on the occurrence of an error. This data comprises of error logs, event logs and OS dump files. This information will be used by various tools that monitor server health. In many implementations, this data gets stored in pieces in a non-volatile storage connected to the Baseboard Management Controller(BMC) and/or the disk connected via the Input/Output(I/O) Hub as shown in “Traditional approach for storing MCA logs” in Figure 1. Today there are few challenges in gathering the required data for analysis. Each of the challenges are listed and explained below:

- 1) Links to the store becoming inaccessible.
 - a. Non-volatile store
 - b. Disk store

The storage locations used for logging the data during server error are behind the I/O hub. If the links connecting these fail or is the source of a machine check error condition, or if the device itself fails, then there is no alternate path to continue the logging. The links marked in the diagram 1 with  represent points of failures in the logging process. This will mean the entire error data or a part of the data will be lost depending on the point of failure on the system.

2) Slow nature of the store causing hindrance in gathering data.

Transactions to the disk and NVRAM (Non-Volatile Random Access Memory) are slow in nature as they reside behind the I/O hub. Writing to these devices on an event of failure takes considerable time and reading the data back as well incurs the same penalty, chances of secondary failures on the system reducing the possibilities of capturing a complete log. Considering the server failure scenarios will mostly be time critical ones, this will turn-out to be a critical aspect hindering the amount of data gathered.

3) Data scattered across different storage locations and NVRAM space limitation in systems.

Even with the slow transactions and single point of failure links the data gathered is still spread across multiple locations. This means that consumers of this data will not get a consolidated set resulting in an incomplete analysis of the error data. Also, in most design, the NVRAM size is limited and is not designed to store large sized error information.

The apparatus and methodology brought out in this proposal will provide fast data gathering containing all the necessary data store in one location and ensure there are no single points of failures, hindering the data gathering process.

Solution

We propose using the advancements in the area of persistent memory and platform firmware and OS enhancements to enable consolidated collection of Machine Check Abort (MCA) logs, OS dumps and other important error information. The solution proposes a method and apparatus for consolidation and management of platform error and support logs (MCA logs, OS dumps, event logs and other error information) in a faster, reliable and highly available design using a combination of NVRAM and PMEM (Persistent Memory) blocks connected to the processor, which can be accessed as a block device or as byte addressable device. The solution helps in consolidation of firmware logs, event logs, OS states all in a single reserved memory area carved out for this purpose in PMEM, in a way that it is available for use by both Operating system and firmware. The design ensures high availability by addressing all single point of failures/MCA conditions that obstruct storage or retrieval of the logs from the failed hardware including processor failure, chip set failure, link failure and NVM failure by the use of redundant memory storage areas and by allowing the consolidated information to be accessible in band or out of band from a BMC through Unified Extended Firmware Interface (UEFI) or from the OS agents in the Operating system stack.

Detailed description

We propose using a section of Persistent Memory (PMEM) and make it visible both to the firmware and the OS by reserving a few blocks of memory by the firmware for consolidated logs

in PMEM. The method of storing MCA logs in Persistent Memory (PMEM) will be policy based that can be selected during system configuration using RBSU or RIS in combination with selective OS dump. If this policy is not chosen then the traditional approach of storing the logs in non-volatile store via the BMC will be used.

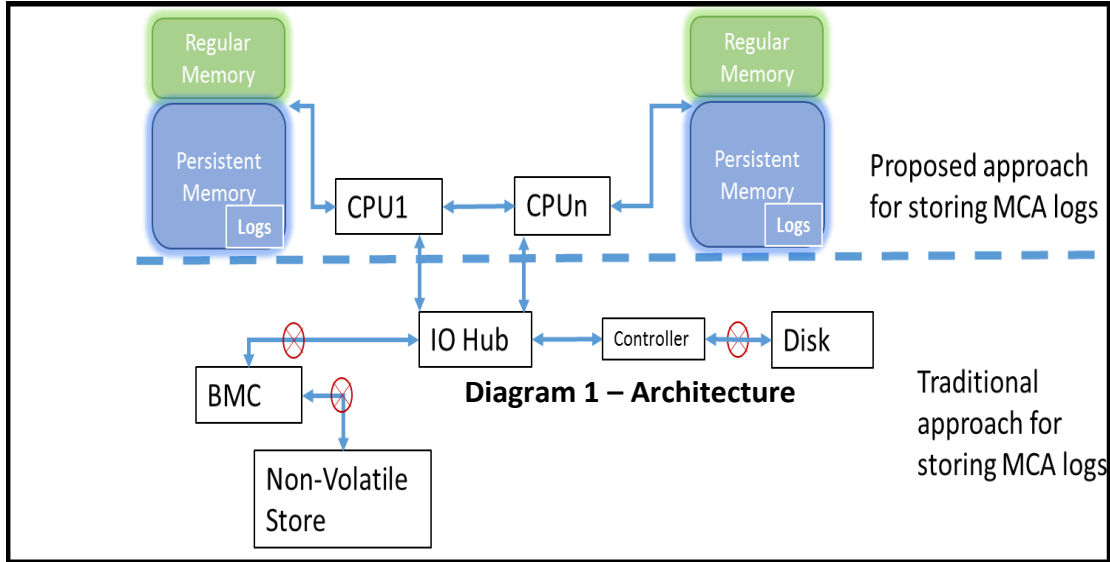
Upon choosing the FRC policy BIOS will select PMEM regions. BIOS will reserve PMEM regions and reflect the reserved blocks in the Advanced Configuration and Power Interface(ACPI) NVDIMM Firmware Interface Table (NFIT) tables. For highly mission critical systems, we propose a high availability design that allows reserving PMEM blocks in memory connected to at least two processors in a multi-socket system and/or the slow NVRAM area behind the Southbridge for storing a redundant copy. The PMEM region will be reserved by selecting regions across different PMEM units connected to different processors in the system. This region will accommodate the firmware logs and also partial/Selective OS crash dump along with other error information.

OS will create its system memory map with input from the ACPI NFIT tables, understand that the firmware has reserved PMEM for logs and accordingly configure to read the error logs from this space and/or write selective dumps to it. The OS data required for troubleshooting MCAs will be dumped via the OS selective dump feature, ensuring that only the necessary data is gathered resulting in much lesser space requirements.

Since PMEM is visible to the firmware, the firmware can directly write to the reserved blocks in PMEM with the retrieved log information. Since the OS knows the range of memory allocated for the logs through the NFIT tables, the OS can then read these blocks to gather information on the error log buffers avoiding single point failure scenarios and will consolidate the log with its own information and store it in either a PMEM partition maintaining a redundant copy based on the policy selected. Post hardware failure, on the next power on, boot the system with the minimum configuration such as the minimum boot or single user mode boot, and read the logs and dump from PMEM. Alternatively, the copy of consolidated logs stored in the NVRAM connected to BMC can also be retrieved through the management processor even if the server does not come up. The high level architecture is shown in Diagram 1.

As PMEM offers near DRAM(Dynamic Random Access Memory) speeds, the capture of the selective dump will be very fast compared to the traditional approach providing the required performance, both for reading the dump file and writing into the dump file and for the read and write of errorlog files. Also, most operating systems provide a minimum boot capability, where the system has the ability to boot with most devices excluded from the configuration. With this configuration, the system can be booted with the IO completely excluded and this ensures that the system is able to boot up for the purposes of root cause analysis, in case there was a link or a device failure in the IO subsystem as shown in diagram 1.

Also using PMEM along with NVRAM for logging makes redundant copies of error log buffers available. Thus a comprehensive, reliable and faster set of logs will be available for analysis of an error. The combination of the errorlog files, the selective dump and event logs in PMEM makes it an ideal solution for mission critical customers.



Disclosed by Clarete Riana Crasta, Sandeep Yelandur S and Suhas Shivanna, Hewlett Packard Enterprise