September 19, 2019

METHOD TO EXTEND VOLATILE UEFI VARIABLE ACROSS SX STATE

HP INC

Method to Extend Volatile UEFI Variable Across Sx State

Abstract: UEFI policies/variables are maintained across power cycles in faster, persistent random-access memory of an embedded controller.

This disclosure relates to the field of Unified Extensible Firmware Interface (UEFI)

A technique is disclosed that maintains UEFI policies/variables in faster random-access memory across power cycles.

UEFI specifies a mechanism that connects a computer's firmware to an operating system of the computer. This it is similar to BIOS, which it is eventually expected to supercede.

According to the UEFI specification, UEFI policies/variables are stored either in non-volatile storage or volatile normal memory. Variables/policies referenced after the Sx state need to be non-volatile. For example, UEFI firmware update is initiated during the OS bootloader but typically needs to happen during POST (power-on self-test). A flag is written by a runtime service to indicate that a device firmware update is pending. It will be referenced after reset,and deleted after the firmware update is complete. The flag is defined as a UEFI non-volatile variable. Other examples include memory timing information, memory module SPD data and training information are needed after S4/S5 state thus need to be defined as non-volatile UEFI variable.

Non-volatile storage is typically a SPI (Serial Peripheral Interface) serial flash memory. Such a memory disadvantageously has a limited capacity, takes a relatively long time to update, and which may wear out with time and usage. Volatile variables are faster and randomly accessible, thus providing much better performance. However, volatile normal memory can't retain its stored values across a power cycle, thus limiting its usefulness for UEFI policies and variables.

Some PCs include an embedded controller (EC) which has its own random-access memory that does maintain its values across the Sx power state and power cycling. Such memory can be accessed much more quickly than SPI flash, and particular locations accessed directly (i.e. randomly).

According to the present disclosure, volatile variables are stored in the embedded controller's memory instead of normal memory so that volatile variable can retain its value across the Sx power state.

The host and EC interfaces are enhanced to support UEFI variables.

In the UEFI Get/Setvariable functions, code is added to check variable attributes. If they are defined as other than EFI_VARIABLE_NON_VOLATILE, the EC interface is called to read/write the variable from/to the EC. If this operation does not succeed, the variable is read from/written to host normal memory.

The logic in UEFI that is used to get/set variables from/to normal (volatile) memory is modified to use the memory on the embedded controller instead. Furthermore, get/set variable function calls are changed from non-volatile to volatile, if the variable attributes meet the requirement.

With this enhanced usage of the embedded controller, a significant number of non-volatile UEFI variables can be migrated to volatile UEFI variables. This results in a smaller size of the variable service region in the SPI part; faster response; and less time-consuming variable service reclaim. Accordingly, by maximizing the use of volatile UEFI variables, performance of the UEFI variable service is significantly improved.

Disclosed by Wei Ze Liu, Rosilet Retnamoni Braduke, and Tom Hung, HP Inc.