December 2019

UEFI EMBEDDED DEVICE FIRMWARE RECOVERY

HP INC

UEFI Embedded Device Firmware Recovery

Abstract: The firmware for embedded devices in a PC system is recoverable from a combined embedded device firmware image stored on the primary mass storage device of the PC.

This disclosure relates to the field of system recovery.

A technique is disclosed that recovers the firmware of devices embedded in a PC system.

PC systems often include embedded peripheral devices, such as for example a camera controller, a touch controller, and the like. Many such embedded devices integrate a small capacity SPI ROM that stores the firmware for the device. If the firmware of the embedded device somehow becomes corrupted, the embedded device will fail to function properly. Unfortunately, the small capacity of the SPI ROM usually precludes the inclusion of a firmware image backup in the ROM.

According to the present disclosure, the BIOS build process combines the firmware files of the various embedded devices to generate a combined firmware image, stored on a partition of the primary mass storage device of the PC, that can be used to restore the firmware of an embedded device if it becomes corrupted.

A BIOS driver module handles creation and utilization of the embedded device firmware backup image. A Windows application will trigger the embedded device firmware recovery method when the embedded device fails to function due to firmware corruption. A BIOS module detects the firmware corruption and recovers the embedded device firmware from the backup image.

The BIOS build process generates a combined firmware image that includes standard BIOS functions, the firmware images of the embedded devices, and the BIOS modules and extensions that support recovery of embedded device firmware. Hash256 calculations are utilized to verify the status of the firmware image.

The embedded device firmware backup image is updated when Windows Update, or another firmware update utility, performs a firmware update, according to the following process:

1. Windows Update requests a firmware update. The PC is rebooted after completion.
2. When the PC boots up again, the BIOS detects that the OS requested a firmware update by Windows Update in the previous boot.
3. The BIOS reads the combined firmware binary from EFI partition on Disk. The F10 (stratus) reads the combined firmware image from the USB key list. If the USB key does not find the combined firmware image, it requests a stored copy from the EFI partition on Disk.
4. The BIOS parses the combined firmware image and extracts the desired individual firmware image.
5. The BIOS reprograms the firmware of the embedded device(s) and return embedded device firmware updated status for each individual device.
6. When successfully updated, the enhanced BIOS module stages the recovery firmware for the various devices on the EFI partition. A hash256 value is calculated for each individual device recovery firmware image and the value is

appended to the firmware image. The hash256 value is also recorded in non-volatile memory.

7. The PC is then cold-booted, and normal operation proceeds.

If the firmware of an embedded device is corrupted, the firmware is recovered during the pre-OS environment according to the following process:

1. Windows Application triggers firmware recovery in the next reboot when the embedded device fails to function due to firmware corruption.
2. When the PC boots up again, the BIOS detect that the OS has requested embedded device firmware recovery by Windows Application in the previous boot. The BIOS then determines whether the embedded device firmware of any device has been corrupted.
3. The BIOS reads the firmware image from the EFI partition and verifies its correctness.
4. The BIOS recovers the firmware of various embedded devices and verifies that the status indicates successful update.
6. The PC is then cold-booted, and normal operation proceeds.

The disclosed technique advantageously provides an automatic recovery procedure for embedded device firmware in a PC system without incurring any additional hardware cost.

Disclosed by Tom Hung, Wei Ze Liu, Nung-Kai Chen, and Harry Chang, HP Inc.