(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
20 August 2020 (20.08.2020)

WIPO | PCT

(10) International Publication Number
WO 2020/163977 A1

(51) International Patent Classification:
G06F 21/00 (2013.01)

(21) International Application Number:
PCT/CN2019/074776

(22) International Filing Date:
11 February 2019 (11.02.2019)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **INTEL CORPORATION** [US/US]; 2200 Mission College Blvd., Santa Clara, California 95054 (US).

(72) Inventors; and

(71) Applicants *(for BZ only)*: **VAN DE GROENENDAAL, Johan** [US/US]; 1203 NW Mayfield Rd., Portland, Oregon 97229 (US). **WANG, Qian** [CN/US]; Intel Corporation JF5 2111 NE 25th Ave, Hillsboro, Oregon 97124 (US). **WSZOLEK, Kasper** [PL/PL]; Intel Technology Poland Slowackiego 173, 80-298 Gdansk PM (PL). **ZENG, Yong** [CN/CN]; No. 880 Zi Xing Road, Zizhu Science Park, Shanghai 200241 (CN). **YU, Yang Yang** [CN/CN]; Building 5, No.14-16, Lane 647, Songtao RD, China (Shanghai) Pilot Free Trade Zone, Shanghai 201203 (CN).

(74) Agent: **CHINA PATENT AGENT (H.K.) LTD.**; 22/F., Great Eagle Center, 23 Harbour Road, Wanchai, Hong Kong (CN).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:
— *of inventorship (Rule 4.17(iv))*

Published:
— *with international search report (Art. 21(3))*

(54) **Title:** VIRTUAL FLASH

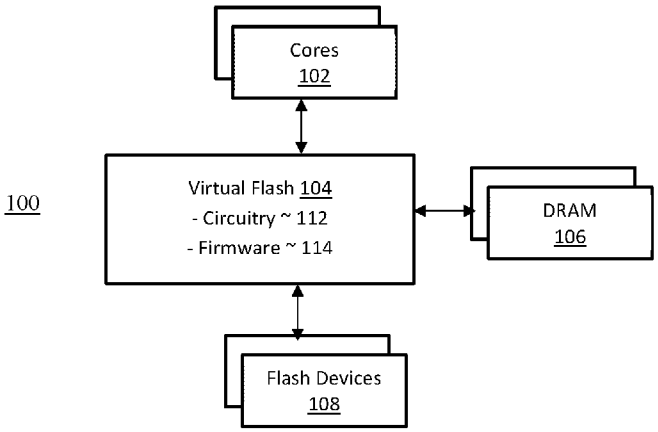


Figure 1

(57) **Abstract:** Methods, apparatuses and at least one computer-readable medium associated with provision of a virtual flash (104) for computing are provided herein. Provided is a virtual flash (104) that includes circuitry (112) and firmware (114) in cooperation to facilitate accesses by one or more processor cores (102) of a computing platform for one or more platform firmware images from one or more flash devices (108) of the computing platform. The facilitation includes return of the one or more platform firmware images for the one or more processor cores (102) from one or more secured memory areas of the computing platform instead.

WO 2020/163977 A1

VIRTUAL FLASH

BACKGROUND

In traditional computing systems, firmware/BIOS images are often stored in flash devices, with the host processors of the computer systems retrieving the firmware/BIOS images from the flash devices of the computer systems. In emerging computing systems, there is a desire to enhance the security of the firmware/BIOS images.

BRIEF DESCRIPTION OF THE FIGURES

Embodiments will be readily understood by the following detailed description in conjunction with the accompanying drawings. To facilitate this description, like reference numerals designate like structural elements. Embodiments are illustrated by way of example and not by way of limitation in the figures of the accompanying drawings.

Figure 1 illustrates an overview of an architecture of a computing platform incorporated with the teachings of the present disclosure, according to various embodiments.

Figure 2 illustrates example circuitry of a virtual flash, according to various embodiments.

Figure 3 illustrates example virtual flash firmware, according to various embodiments.

Figure 4 illustrates an example virtual flash process, according to various embodiments.

Figure 5 illustrates, wherein an example virtual flash initialization process, according to various embodiments.

Figure 6 illustrates an example virtual flash operation process, according to various embodiments.

Figure 7 illustrates an example computing device suitable for use to practice various aspects of the present disclosure, in accordance with various embodiments.

Figure 8 illustrates a storage medium having executable instructions for implementing aspects of the present disclosure, in accordance with various embodiments.

DETAILED DESCRIPTION

To address the challenge/problem described in the Background section, the present disclosure provides a virtual flash to a computing platform to facilitate a host processor of the computing platform to access all platform firmware/BIOS images. At initialization, the virtual flash copies the platform firmware/BIOS images from various flash devices of the

computing platform, and stores the platform firmware/BIOS images into secured memory areas of the computing platform. A host processor of the computing platform accesses a platform firmware/BIOS image via the virtual flash. In response to an attempted access by a host processor, the virtual flash returns the platform firmware/BIOS image of interest from the secured memory areas instead.

Resultantly, the host processors of the computing platform are isolated from the platform firmware/BIOS images, as desired. Further, the present disclosure allows data center products to have a more secure firmware management architecture. It also provides out-of-band (OOB) access for data center administrators to manage all firmware entities on a computing platform. These and other aspects of the present disclosure will be further described below with references to the accompanying drawings.

In referencing the accompanying drawings, the same reference numbers may be used in different drawings to identify the same or similar elements. In the following description, for purposes of explanation and not limitation, specific details are set forth such as particular structures, architectures, interfaces, techniques, etc. in order to provide a thorough understanding of the various aspects of various embodiments. However, it will be apparent to those skilled in the art having the benefit of the present disclosure that the various aspects of the various embodiments may be practiced in other examples that depart from these specific details. In certain instances, descriptions of well-known devices, circuits, and methods are omitted so as not to obscure the description of the various embodiments with unnecessary detail.

Various operations will be described as multiple discrete operations, in turn, in a manner that is most helpful in understanding the illustrative embodiments; however, the order of description should not be construed as to imply that these operations are necessarily order dependent. In particular, these operations may not be performed in the order of presentation.

The phrase "in various embodiments," "in some embodiments," and the like are used repeatedly. The phrase generally does not refer to the same embodiments; however, it may. The terms "comprising," "having," and "including" are synonymous, unless the context dictates otherwise. The phrase "A and/or B" means (A), (B), or (A and B). The phrases "A/B" and "A or B" mean (A), (B), or (A and B), similar to the phrase "A and/or B." For the purposes of the present disclosure, the phrase "at least one of A and B" means (A), (B), or (A and B). The description may use the phrases "in an embodiment," "in

embodiments," "in some embodiments," and/or "in various embodiments," which may each refer to one or more of the same or different embodiments. Furthermore, the terms "comprising," "including," "having," and the like, as used with respect to embodiments of the present disclosure, are synonymous.

Example embodiments may be described as a process depicted as a flowchart, a flow diagram, a data flow diagram, a structure diagram, or a block diagram. Although a flowchart may describe the operations as a sequential process, many of the operations may be performed in parallel, concurrently, or simultaneously. In addition, the order of the operations may be re-arranged. A process may be terminated when its operations are completed, but may also have additional steps not included in the figure(s). A process may correspond to a method, a function, a procedure, a subroutine, a subprogram, and the like. When a process corresponds to a function, its termination may correspond to a return of the function to the calling function and/or the main function.

Example embodiments may be described in the general context of computer-executable instructions, such as program code, software modules, and/or functional processes, being executed by one or more of the aforementioned circuitry. The program code, software modules, and/or functional processes may include routines, programs, objects, components, data structures, etc., that perform particular tasks or implement particular data types. The program code, software modules, and/or functional processes discussed herein may be implemented using existing hardware in existing communication networks. For example, program code, software modules, and/or functional processes discussed herein may be implemented using existing hardware at existing network elements or control nodes.

As used herein, the term "circuitry" refers to, is part of, or includes hardware components such as an electronic circuit, a logic circuit, a processor (shared, dedicated, or group) and/or memory (shared, dedicated, or group), an Application Specific Integrated Circuit (ASIC), a field-programmable device (FPD), (for example, a FPGA, a programmable logic device (PLD), a complex PLD (CPLD), a high-capacity PLD (HCPLD), a structured ASIC, or a programmable System on Chip (SoC)), digital signal processors (DSPs), etc., that are configured to provide the described functionality. In some embodiments, the circuitry may execute one or more software or firmware programs to provide at least some of the described functionality.

As used herein, the term "processor circuitry" may refer to, is part of, or includes circuitry capable of sequentially and automatically carrying out a sequence of arithmetic or logical operations; recording, storing, and/or transferring digital data. The term "processor circuitry" may refer to one or more application processors, one or more baseband processors, a physical central processing unit (CPU), a single-core processor, a dual-core processor, a triple-core processor, a quad-core processor, and/or any other device capable of executing or otherwise operating computer-executable instructions, such as program code, software modules, and/or functional processes. As used herein, the term "interface circuitry" may refer to, is part of, or includes circuitry providing for the exchange of information between two or more components or devices. The term "interface circuitry" may refer to one or more hardware interfaces (for example, buses, input/output (I/O) interfaces, peripheral component interfaces, network interface cards, and/or the like).

As used herein, the term "computing platform" may describe any physical hardware device capable of sequentially and automatically carrying out a sequence of arithmetic or logical operations, equipped to record/store data on a machine readable medium, and transmit and receive data from one or more other devices in a communications network. A computing platform may be considered synonymous to, and may hereafter be occasionally referred to, as a computer, computing device, etc. The term "computer system" may include any type interconnected electronic devices, computer devices, or components thereof. Additionally, the term "computer system" and/or "system" may refer to various components of a computer that are communicatively coupled with one another. Furthermore, the term "computer system" and/or "system" may refer to multiple computer devices and/or multiple computing systems that are communicatively coupled with one another and configured to share computing and/or networking resources. As used herein, the term "user equipment" or "UE" may refer to a device, such as a computer device, with radio communication capabilities and may describe a remote user of network resources in a communications network. The term "user equipment" or "UE" may be considered synonymous to, and may hereafter be occasionally referred to as client, mobile, mobile device, mobile terminal, user terminal, mobile unit, mobile station, mobile user, subscriber, user, remote station, access agent, user agent, receiver, radio equipment, reconfigurable radio equipment, reconfigurable mobile device, etc.

Examples of "computer devices", "computer systems", "UEs", etc. may include cellular phones or smart phones, feature phones, tablet personal computers, wearable computing devices, an autonomous sensors, laptop computers, desktop personal computers, video game consoles, digital media players, handheld messaging devices, personal data assistants, an electronic book readers, augmented reality devices, server computer devices (e.g., stand-alone, rack-mounted, blade, etc.), cloud computing services/systems, network elements, in-vehicle infotainment (IVI), in-car entertainment (ICE) devices, an Instrument Cluster (IC), head-up display (HUD) devices, onboard diagnostic (OBD) devices, dashtop mobile equipment (DME), mobile data terminals (MDTs), Electronic Engine Management System (EEMS), electronic/engine control units (ECUs), electronic/engine control modules (ECMs), embedded systems, microcontrollers, control modules, engine management systems (EMS), networked or "smart" appliances, machine-type communications (MTC) devices, machine-to-machine (M2M), Internet of Things (IoT) devices, and/or any other like electronic devices. Moreover, the term "vehicle-embedded computer device" may refer to any computer device and/or computer system physically mounted on, built in, or otherwise embedded in a vehicle.

A computing system or a platform may use various devices coupled to a computer bus extensively. A computer bus may include related hardware components (wire, optical fiber, etc.) and software, including communication protocols. A peripheral component interconnect (PCI) bus or a PCI Express (PCIe, PCI-E) may be a computer bus based on a specification that provides a mechanism for system software, or a system driver, to perform various operations related to the configuration of a device coupled to the PCI bus or the PCIe bus. Devices, or components coupled to a computer bus may also be referred to as functions. PCIe may operate in consumer, server, and industrial applications, as a motherboard-level interconnect (to link motherboard-mounted peripherals), a passive backplane interconnect, and as an expansion card interface for add-in boards. PCIe devices communicate via a logical connection called an interconnect or link. A link is a point-to-point communication channel between two PCIe ports allowing both of them to send and receive ordinary PCI requests, e.g., configuration, input/output (I/O), or memory read/write, and interrupts. At the physical level, a link may be composed of one or more lanes. Low-speed peripherals, such as an 802.11 Wi-Fi card, use a single-lane (×1) link, while a graphics adapter typically uses a much wider and faster 16-lane link.

Referring now to **Figure 1**, wherein an overview of an architecture of a computing platform incorporated with the teachings of the present disclosure, according to various embodiments, is shown. As illustrated, computing platform 100 includes processor cores 102, virtual flash 104 of the present disclosure, dynamic random access memory (DRAM) 106, and flash devices 108, coupled with each other. Virtual flash 104 includes circuitry 112 and complementary firmware 114. Circuitry 112 and complementary firmware 114 cooperates to implement the functions of virtual flash 104. Processor cores 102, except for their use of virtual flash 104 to retrieve firmware/BIOS images of interest, may be any one of a number of processor cores known in the art. Virtual flash 104, as described earlier, facilitates accesses of processor cores 102 to all platform firmware/BIOS images in all flash devices of the computing platform. In other words, virtual flash 104 behaves as a flash device to processor cores 102, abstracting all flash devices of the computing platform for processor cores 102. At initialization, virtual flash 104 copies the platform firmware/BIOS images from flash devices 108, and stores the platform firmware/BIOS images into DRAM 106, which are secured memory areas of computing platform 100 (or causes the copy and storage operations to be performed e.g., by a service of an operating system (OS)). In various embodiments, virtual flash firmware 114 controls and limits the access of processor cores 102 to the firmware/BIOS images stored in DRAM 106 according to a predefined security configuration of those images. During operation, a processor core 102 accesses a platform firmware/BIOS image of interest via virtual flash 104. In response to an attempted access by a processor core 102, virtual flash 104 returns the platform firmware/BIOS image of interest from DRAM 106, the secured memory areas instead. In various embodiments, processor core 102 may be part of a system-on-chip (SoC) or a compute element of computing platform 100 (e.g., a peripheral controller hub (PCH). Hereinafter, processor core 102 may simply be referred to as core 102.

In various embodiments, as earlier described, virtual flash 104 is implemented as a combination of circuitry 112 complemented with a set of virtual flash firmware 114. In various embodiments, circuitry 112 are implemented as synchronous circuits, including a frame buffer, command forward circuit, command translation circuit, and a number of interface circuits. In various embodiments, the synchronous circuits are implemented in Register Transfer Logic (RTL), using a Field Programming Gate Arrays (FPGA). In various embodiments, complementary virtual flash firmware 114 includes initialization, decryption and synchronization functions. In various embodiments, cores 102 and virtual

flash 104 communicates in accordance with a serial peripheral interface (SPI) protocol, in particular, with cores 102 acting as the SPI master. These and other aspects will be further described below with references to **Figure 2-4**.

DRAM 106, except for its use as secured memory areas to store the flash/BIOS images, to allow virtual flash 104 to service accesses of cores 102, may be any dynamic random access memory known in the art. In various embodiments, DRAM 106 may be double data rate synchronous DRAM (DDRAM), static random access memory (SRAM), On-Chip embedded memory or any other low-latency high capacity memory capable of storing firmware/BIOS images. Similarly, except for its use to provide the flash/BIOS images for virtual flash 104 to copy and store the flash/BIOS images into DRAM 106 to service processor cores 102, flash devices 108 may be any one of a number of flash devices known in the art, including but are not limited to solid state drive (SSD) storage, embedded multi-media controller (eMMC), and NAND.

In various embodiments where at least one of cores 102 is a processor core, processor core 102 and circuitry 112 of virtual flash 104 may be co-disposed in the same integrated circuit package, e.g., a system-on-chip (SoC). For example, circuitry 112 of virtual flash 104 is implemented with the FPGA of the SoC, and virtual flash firmware 114 is implemented with (executed by) the hard processor system (HPS) of the SoC. In various embodiments, virtual flash 104 may serve as a virtual building block for the Platform Root of Trust (PRoT). Additionally, before further describing virtual flash 104, it should be noted that while for ease of understanding, virtual flash 104 is illustrated as coupling to DRAM 106 and flash devices 108, without showing any intermediate elements, in various embodiments, virtual flash 104 may be directly or indirectly coupled to DRAM 106. For example, virtual flash 104 may be coupled to DRAM 106 via a DRAM controller of DRAM 106 of computing platform 100.

Referring now to **Figure 2**, wherein example synchronous circuitry of a virtual flash, according to various embodiments, is shown. As illustrated, example synchronous circuitry of virtual flash 200, which may be part of virtual flash 104 of **Figure 1**, includes frame buffer 212, command forward circuitry 214, command translate circuitry 216, transmit buffer 218, and a number of interface circuitry 202-206, coupled with each other. In various embodiments, interface circuitry 202-206 includes core interface circuitry 202, firmware interface 204, and DRAM interface 206. Further, for the illustrated

embodiments, example circuitry of virtual flash 200 includes a number of registers 220 (also referred to as local registers).

As described earlier, in various embodiments, synchronous circuitry of virtual flash 200 is implemented in RTL, using a FPGA. In various embodiments, the elements 202-220 of virtual flash 200 are configured to support communication with an accessing core in accordance with a SPI protocol, with the accessing core acting as a SPI master to virtual flash 200. For these embodiments, the elements 202-220 of virtual flash 200 cooperate to read and write data to external DRAM according to various SPI commands. For SPI read commands (received through firmware interface 204), virtual flash 200 reads data from the external DRAM (via DRAM interface 206), and returns them to e.g., a core of the computing platform (not shown), via core interface 202. These read transactions do not affect the processor cores or the flash devices. For SPI write data commands (received through firmware interface 204), virtual flash 200 writes data to the external DRAM (via DRAM interface 206), and updates local registers 220.

In various embodiments, virtual flash 200 notifies the firmware 114 about the written data's DRAM starting address and length using the interrupt service of the computing platform. Virtual flash 200 may set a busy status on triggering an interrupt to notify firmware 114, and clear the busy status when the processor cores clear the interrupt. In various embodiments, virtual flash 200 maintains the data locally.

In various embodiments, core interface 202 is configured to interface with various cores, e.g., Host CPU, PCH, network interface (NIC) and so forth, with each at the corresponding core's clock rate. Frame buffer 212 is configured to receive data/commands from the various cores, via the core interface 202, at the core's clock rate, and buffer the received data/commands. In various SPI embodiments, frame buffer 212 supports multiple modes of operation, including a single mode and an extended single mode of operation, providing different number of data lines at the physical layer. Additionally, frame buffer 212 is configured to be responsive to both SPI Register Read and Register Write commands. In various embodiments, frame buffer 212 employs pre-fetch for SPI Read commands. Further, it is configured to add dummy cycles to the pre-fetches for SPI Fast Read commands. In some embodiments, for pre-fetch, frame buffer 212 asks for 512 bytes in bulk. In various embodiments, frame buffer 212 transitions from the SPI clock rate to the local clock rate via over sampling. Still further, frame buffer 212 transitions to a bit width of 8 bit, regardless of whether the inputs are 1, 2 or 4 bits.

Command forward circuitry 214, coupled to frame buffer 212, is configured to retrieve the buffered data/commands at a local clock rate, and forward the retrieved data/commands for selective translation, in particular, to command translate circuitry 216, at the local clock rate.

In various embodiments, command translate circuitry 216, coupled to command forward circuitry 214, is configured to receive the forwarded commands at the local clock rate, selectively translate the forwarded commands, and forward the translated commands to firmware 114 or the memory controller of the secured memory areas, at the local clock rate. In the former case, forwarding of the translated or "as is" commands to firmware 114 is through the firmware interface 204, which is configured to interface with the firmware. In the latter case, forwarding of the translated or "as is" commands to the DRAM is through the DRAM interface 206, which is configured to interface with e.g., the memory controller of the secured memory areas.

In various embodiments, command translate circuitry 216 ignores all SPI Register Read commands. All SPI PRPOGRAM/ ERASE/ Non-Volatile WRITE commands are forwarded to the processor cores "as is" through the processor core interface 202. SPI PROGRAM commands are translated from 8 bit format to 128 bit and forwarded to the DRAM through the DRAM interface 206. All SPI ERASE commands are translated to memory write command and forwarded to the DRAM through the DRAM interface 206. All SPI READ commands are translated from 8 bit to 128 bit and forwarded to the DRAM through the DRAM interface 206.

In various embodiments, transmit buffer 218 is configured to receive data/commands from the DRAM, via DRAM interface 206, at the local clock rate, and buffer the received data/commands. Transmit buffer 218 is further configured to output the buffered data/commands for the cores, via core interface 202, at the core's clock rate. In various embodiments, it uses a First-in First-Out (FIFO) arrangement to accommodate wait requests from the memory controller of the external DRAM. In various embodiments, transmit buffer 218 supports an input data width of 128 bits at the local clock rate, and an output data width of 8 bit at the SPI clock rate. In various embodiments, transmit buffer 218 is implemented with random access memory (RAM), with the SPI address byte serving as the read address.

In various embodiments, firmware interface 204 is configured to inform firmware 114 with interrupts, in response to receipt of SPI PRPOGRAM/ ERASE / Non-Volatile

WRITE command/ Address/ Length commands. Additionally, processor core interface 202 is configured to update non-volatile pre-setting from the processor cores, in response to receipt of SPI Non-Volatile Write commands. Further, firmware interface 204 is configured to update a local registers 220, in response to receipts of SPI Write commands. Still further, processor core interface 202 is configured to generate local virtual registers and pass to command forward circuitry 214, and receive ID/ Parameter/ Lock register information from the processor cores.

In various embodiments, registers 220 are configured to store the virtual flash configuration parameters, and/or operation/transaction information. In various embodiments, registers 220 may include status and/or flag registers, similar to the registers sets exposed by flash devices 108. These registers 220 are accessible by various cores through core interface 202.

Referring now to **Figure 3**, wherein an example virtual flash firmware, according to various embodiments, is illustrated. As shown, virtual flash firmware 300, which may be virtual flash firmware 114 of **Figure 1**, includes a set of virtual flash parameters 302, a decryption module 304, and a flash image synchronization module 306. Virtual flash parameters 302 are used to configure the virtual flash at initialization time, setting the configuration for a particular instantiation of the virtual flash. In various embodiments, virtual flash parameters 302 configure the virtual flash to support a particular SPI implementation. Examples of the virtual flash parameters 302 may include, but are not limited to, virtual flash ID, virtual flash parameters to support "Read Serial Flash Discovery Parameter" command, non-volatile configuration, whether firmware/BIOS images are encrypted, and so forth. In various embodiments, the virtual flash parameters also provide the security configuration of the images, including the access permission of the cores, i.e., Read-Only, Read-Write or no access. In various embodiments, virtual flash parameters 302 need to be initialized only once per power on cycle.

For the illustrated embodiments, virtual flash parameters 302 are retrieved by an initialization module 312 of a kernel 310 of an OS of the computing platform. Initialization module 312, in turn, uses the virtual flash parameters 302 to configure virtual flash 322, which may be virtual flash 104 of **Figure 1** or virtual flash 200 of **Figure 2**. In alternate embodiments, in lieu of disposing initialization module 312 as part of kernel 310, initialization module 312 may be packaged together with virtual flash parameters 302, decryption module 304, and/or flash image synchronization module 306.

In various embodiments, platform firmware/BIOS images stored in various flash devices may be encrypted. For these embodiments, decryption module 304 is configured to decrypt the flash/BIOS images, upon receiving them at initialization time, prior to storing them into the secured memory areas of the computing platform. In various embodiments decryption 304 may retrieve the encrypted firmware/BIOS images, from the flash devices 326, via memory technology device (MTD) driver 316 of kernel 310. As described earlier, in various embodiments, flash devices 326 may be any non-volatile media accessible by the virtual flash firmware, i.e. SPI Flash, eMMC, SSD and NAND. In various embodiments, decryption module 304 may be configured to support decryption of multiple encryption protocols, including but are not limited to Data Encryption Standard (DES), Advanced Encryption Standard (AES), International Data Encryption Algorithm (IDEA), Message Digest Algorithm (MD5), Secure Hash Algorithm (SHA) and so forth. In various embodiments, on decryption, decryption module 304 provides a decrypted firmware/BIOS image to a DRAM driver 314 of kernel 310 to write the decrypted firmware/BIOS image into the secured memory areas of the computing platform. In various embodiments, the secured memory areas may be DDRAM, and DRAM driver 314 may be a DDRAM driver.

In various embodiments, flash image synchronization module 306 is configured to update a flash/BIOS image in a flash device, when the copy of the flash/BIOS image stored and maintained in the secured memory areas had been changed, e.g., by the virtual flash firmware, to ensure the nominal copies of the flash/BIOS images stored in the flash drives and the operational copies of the flash/BIOS images stored in the secured memory area provided to the processor cores are in sync, i.e., remains identical.

In various embodiments, flash image synchronization module 306 is configured to provide the updates to MTD driver 316 of kernel 310, which in turn, updates the flash/BIOS images in the flash devices 326 accordingly. In various embodiments, flash image synchronization module 306 performs the synchronization in response to the processor cores receiving a trigger signal or an application programming interface (API) notification, informing the system to shut down. In response, flash image synchronization module 306 synchronizes the following data to the flash drives, and non-volatile parameters to backend mass storage, via MTD driver 316. Synchronization may be performed as soon as DC power of the computing platform is off. In some embodiments, Synchronization API notification may be called by BIOS/ME/OOB management (ME =

Manageability Engine). In other embodiments, synchronization signal may be provided by the main board of the computing platform.

Referring now to **Figure 4**, wherein an example virtual flash process, according to various embodiments, is illustrated. As shown, virtual flash process 400 includes operations performed at blocks 402-408. Process 400 may be performed by e.g., virtual flash 104 of **Figure 1**. In other embodiments, process 400 may include more or less operation, or some of the operations may be performed in different order.

Process 400 starts at block 402. At block 402, a virtual flash is initialized. The operations to initialize a virtual flash will be further described later with references to **Figure 5**.

Next, at block 404, on initialization, the virtual flash proceeds to operate and services the processor cores with respect to their needs for firmware/BIOS images. The operations to service a processor core with respect to its needs for firmware/BIOS image will be further described later with references to **Figure 6**.

At block 406, a determination is made if any of the firmware/BIOS images stored and maintained in the secured memory areas has been updated. In various embodiments, the determination may be performed in real time periodically. In other embodiments, the determination may be performed during power down of the computing platform.

Process 400 returns to block 404 and continues therefrom as earlier described, if a the determination is performed periodically, and a result of the determination indicates that none of the firmware/BIOS images stored and maintained in the secured memory areas has been altered. In alternate embodiments, where the determination is performed only on power down, process 400 may proceed to block 410 and continue with power down, if a result of the determination indicates that none of the firmware/BIOS images stored and maintained in the secured memory areas has been altered.

However, process 400 continues to block 408 and save the updated firmware/BIOS images from the secured memory areas into the flash devices, if a result of the determination indicates that at least one of the firmware/BIOS images stored and maintained in the secured memory areas has been altered. On saving updated firmware/BIOS images from the secured memory areas into the flash devices, process 400 proceeds to block 410, and continuing with power down.

Referring now to **Figure 5**, wherein an example virtual flash initialization process, according to various embodiments, is illustrated. As shown, virtual flash initialization

process 500 includes operations performed at blocks 502-508. Process 500 may be performed by e.g., virtual flash 104 of **Figure 1**, and/or initialization module 312 of kernel 310 of **Figure 3**. In other embodiments, process 500 may include more or less operation, or some of the operations may be performed in different order.

Process 500 starts at block 502. At block 502, a set of virtual flash parameters are received. Next, at block 504, the virtual flash is configured to operate, in accordance with the received virtual flash parameters.

Next, at block 506, firmware/BIOS images are retrieved from various flash devices of a computing platform.

At block 508, the retrieved firmware/BIOS images are decrypted, if necessary, then stored into the secured memory areas of the computing platform.

Referring now to **Figure 6**, wherein an example virtual flash operation process, according to various embodiments, is illustrated. As shown, virtual flash operation process 600 includes operations performed at blocks 602-604. Process 600 may be performed by e.g., virtual flash 104 of **Figure 1**. In other embodiments, process 600 may include more or less operation, or some of the operations may be performed in different order.

Process 600 starts at block 602. At block 602, a request for a firmware/BIOS image of interest to a processor core is received. The request may be received in accordance with a SPI protocol, with the processor core acting as a SPI master to the virtual flash.

Next at block 604, the request is serviced, with the firmware/BIOS image of interest being retrieved and return to the processor core, from the secured memory areas of the computing platform instead (as opposed to from a flash device).

Figure 7 illustrates an example computing device suitable for use to practice various programmatic aspects of the present disclosure, in accordance with various embodiments. As shown, the device 700 may include one or more processors 702, and virtual flash 703. Each processor 702 may include one or more processor cores. The processor cores 702 may also include one or more hardware accelerators (not shown), which may be an ASIC or a FPGA. Virtual flash 703 may be virtual flash 104 of **Figure 1**, having circuitry and complementary firmware, earlier described with references to Figures 2-6. In various embodiments, processor cores 702 and virtual flash 703 may be integrated together on a SOC, with circuitry of virtual flash 703 implemented with a

FPGA of the SoC, and the virtual flash firmware implemented with (executed by) a HPS of the SoC.

Additionally, computing device 700 may include a memory and memory controller 704, which may be any one of a number of known non-persistent storage medium and memory controller, and a number of flash devices 708 including firmware/BIOS images 709. Flash devices 708 may be any non-volatile storage, e.g., eMMC, SSD, NAND, Flash and so forth.

In addition, computing device 700 may include an I/O interface 718, coupled to one or more sensors 714, and a display screen 713. The I/O interface 718 may include a transmitter 723 and a receiver 717. Furthermore, computing device 700 may include communication circuitry 705 including a transceiver (Tx) 711, and network interface controller (NIC) 712. The elements may be coupled to each other via system buses 706, which may represent one or more buses, e.g., one or more PCIe buses. In the case of multiple buses, they may be bridged by one or more bus bridges (not shown).

In various embodiments, processor(s) 702, memory and memory controller 704, I/O interface 718, communication circuitry 705, and/or system buses 706 may be co-disposed on a printed circuit board (PCB), which may be referred to as a main board or a motherboard.

In embodiments, the processor(s) 702 (also referred to as "processor circuitry 702") may be one or more processing elements configured to perform basic arithmetical, logical, and input/output operations by carrying out instructions. Processor circuitry 702 may be implemented as a standalone system/device/package or as part of an existing system/device/package. The processor circuitry 702 may be one or more microprocessors, one or more single-core processors, one or more multi-core processors, one or more multithreaded processors, one or more GPUs, one or more ultra-low voltage processors, one or more embedded processors, one or more DSPs, one or more FPDs (hardware accelerators) such as FPGAs, structured ASICs, programmable SoCs (PSoCs), etc., and/or other processor or processing/controlling circuit. The processor circuitry 702 may be a part of a SoC in which the processor circuitry 702 and other components discussed herein are formed into a single IC or a single package. As examples, the processor circuitry 702 may include one or more Intel Pentium®, Core®, Xeon®, Atom®, or Core M® processor(s); Advanced Micro Devices (AMD) Accelerated Processing Units (APUs), Epyc®, or

Ryzen® processors; Apple Inc. A series, S series, W series, etc. processor(s); Qualcomm snapdragon® processor(s); Samsung Exynos® processor(s); and/or the like.

In embodiments, the processor circuitry 702 may include a sensor hub, which may act as a coprocessor by processing data obtained from the one or more sensors 714. The sensor hub may include circuitry configured to integrate data obtained from each of the one or more sensors 714 by performing arithmetical, logical, and input/output operations. In embodiments, the sensor hub may capable of timestamping obtained sensor data, providing sensor data to the processor circuitry 702 in response to a query for such data, buffering sensor data, continuously streaming sensor data to the processor circuitry 702 including independent streams for each sensor of the one or more sensors 714, reporting sensor data based upon predefined thresholds or conditions/triggers, and/or other like data processing functions.

In embodiments, the memory 704 (also referred to as "memory circuitry 704" or the like) may be circuitry configured to store data or logic for operating the computing device 700. The memory circuitry 704 may include number of memory devices may be used to provide for a given amount of system memory. As examples, the memory circuitry 704 can be any suitable type, number and/or combination of volatile memory devices (e.g., random access memory (RAM), dynamic RAM (DRAM), static RAM (SAM), etc.) and/or non-volatile memory devices (e.g., read-only memory (ROM), erasable programmable read-only memory (EPROM), electrically erasable programmable read-only memory (EEPROM), flash memory, antifuses, etc.) that may be configured in any suitable implementation as are known. In various implementations, individual memory devices may be formed of any number of different package types, such as single die package (SDP), dual die package (DDP) or quad die package, dual inline memory modules (DIMMs) such as microDIMMs or MiniDIMMs, and/or any other like memory devices. To provide for persistent storage of information such as data, applications, operating systems and so forth, the memory circuitry 704 may include one or more mass-storage devices, such as a solid state disk drive (SSDD); flash memory cards, such as SD cards, microSD cards, xD picture cards, and the like, and USB flash drives; on-die memory or registers associated with the processor circuitry 702 (for example, in low power implementations); a micro hard disk drive (HDD); three dimensional cross-point (3D XPOINT) memories from Intel® and Micron®, etc.

Where FPDs are used, the processor circuitry 702 and memory circuitry 704 (and/or flash 708) may comprise logic blocks or logic fabric, memory cells, input/output (I/O) blocks, and other interconnected resources that may be programmed to perform various functions of the example embodiments discussed herein. The memory cells may be used to store data in lookup-tables (LUTs) that are used by the processor circuitry 702 to implement various logic functions. The memory cells may include any combination of various levels of memory/storage including, but not limited to, EPROM, EEPROM, flash memory, SRAM, anti-fuses, etc.

In embodiments, flash 708 (also referred to as "flash device or flash circuitry 708" or the like), with shared or respective controllers, may provide for persistent storage of information such as firmware/BIOS images 709, operating systems, etc. The flash circuitry 178 may be implemented as solid state drives (SSDs); solid state disk drive (SSDD); serial AT attachment (SATA) storage devices (e.g., SATA SSDs); flash drives; flash memory cards, such as SD cards, microSD cards, xD picture cards, and the like, and USB flash drives; three-dimensional cross-point (3D Xpoint) memory devices; on-die memory or registers associated with the processor circuitry 702; hard disk drives (HDDs); micro HDDs; resistance change memories; phase change memories; holographic memories; or chemical memories; among others. As shown, the flash circuitry 708 is included in the computer device 700; however, in other embodiments, the flash circuitry 708 may be implemented as one or more devices separated from the other elements of computer device 700.

In some embodiments, the flash circuitry 708 may further include an operating system (OS) (not shown), which may be a general purpose operating system or an operating system specifically written for and tailored to the computer device 700. The OS may include one or more drivers, libraries, and/or application programming interfaces (APIs), which provide program code and/or software components, and/or control system configurations to control and/or obtain/process data from the one or more sensors 714.

Firmware/BIOS image 709 may be software modules/components used to perform various basic functions/services of the computing device 700 and/or to carry out functions of the example embodiments discussed herein. In embodiments, where the processor circuitry 702 and memory circuitry 704, including hardware accelerators (e.g., FPGA cells, the hardware accelerator 703) as well as processor cores, the hardware accelerators (e.g., the FPGA cells) may be pre-configured (e.g., with appropriate bit streams, logic

blocks/fabric, etc.) with the logic to perform some functions of the embodiments herein (in lieu of employment of programming instructions to be executed by the processor core(s)).

The components of computing device 700 may communicate with one another over the system bus 706. The system bus 706 may include any number of technologies, such as a Local Interconnect Network (LIN); industry standard architecture (ISA); extended ISA (EISA); PCI; PCI extended (PCIx); PCIe; an Inter-Integrated Circuit (I2C) bus; a Parallel Small Computer System Interface (SPI) bus; Common Application Programming Interface (CAPI); point to point interfaces; a power bus; a proprietary bus, for example, Intel® Ultra Path Interface (UPI), Intel® Accelerator Link (IAL), or some other proprietary bus used in a SoC based interface; or any number of other technologies. In some embodiments, the bus 706 may be a controller area network (CAN) bus system, a Time-Trigger Protocol (TTP) system, or a FlexRay system, which may allow various devices (e.g., the one or more sensors 714, etc.) to communicate with one another using messages or frames.

The communications circuitry 705 may include circuitry for communicating with a wireless network or wired network. For example, the communication circuitry 705 may include transceiver (Tx) 711 and network interface controller (NIC) 712. Communications circuitry 705 may include one or more processors (e.g., baseband processors, modems, etc.) that are dedicated to a particular wireless communication protocol.

NIC 712 may be included to provide a wired communication link to a network and/or other devices. The wired communication may provide an Ethernet connection, an Ethernet-over-USB, and/or the like, or may be based on other types of networks, such as DeviceNet, ControlNet, Data Highway+, PROFIBUS, or PROFINET, among many others. An additional NIC 712 may be included to allow connect to a second network (not shown) or other devices, for example, a first NIC 712 providing communications to a network over Ethernet, and a second NIC 712 providing communications to other devices over another type of network, such as a personal area network (PAN) including a personal computer (PC) device. In some embodiments, the various components of the computing device 700, such as the one or more sensors 714, etc. may be connected to the processor(s) 702 via the NIC 712 as discussed above rather than via the I/O circuitry 718 as discussed infra.

The Tx 711 may include one or more radios to wirelessly communicate with a network and/or other devices. The Tx 711 may include hardware devices that enable

communication with wired networks and/or other devices using modulated electromagnetic radiation through a solid or non-solid medium. Such hardware devices may include switches, filters, amplifiers, antenna elements, and the like to facilitate the communications over the air (OTA) by generating or otherwise producing radio waves to transmit data to one or more other devices, and converting received signals into usable information, such as digital data, which may be provided to one or more other components of computing device 700. In some embodiments, the various components of the device 700, such as the one or more sensors 714, etc. may be connected to the computing device 700 via the Tx 711 as discussed above rather than via the I/O circuitry 718 as discussed infra. In one example, the one or more sensors 714 may be coupled with computing device 700 via a short range communication protocol.

The Tx 711 may include one or multiple radios that are compatible with any number of 3GPP (Third Generation Partnership Project) specifications, notably Long Term Evolution (LTE), Long Term Evolution-Advanced (LTE-A), Long Term Evolution-Advanced Pro (LTE-A Pro), and Fifth Generation (5G) New Radio (NR). It can be noted that radios compatible with any number of other fixed, mobile, or satellite communication technologies and standards may be selected. These may include, for example, any Cellular Wide Area radio communication technology, which may include e.g. a 5G communication systems, a Global System for Mobile Communications (GSM) radio communication technology, a General Packet Radio Service (GPRS) radio communication technology, or an Enhanced Data Rates for GSM Evolution (EDGE) radio communication technology. Other Third Generation Partnership Project (3GPP) radio communication technology that may be used includes UMTS (Universal Mobile Telecommunications System), FOMA (Freedom of Multimedia Access), 3GPP LTE (Long Term Evolution), 3GPP LTE Advanced (Long Term Evolution Advanced), 3GPP LTE Advanced Pro (Long Term Evolution Advanced Pro)), CDMA2000 (Code division multiple access 2000), CDPD (Cellular Digital Packet Data), Mobitex, 3G (Third Generation), CSD (Circuit Switched Data), HSCSD (High-Speed Circuit-Switched Data), UMTS (3G) (Universal Mobile Telecommunications System (Third Generation)), W-CDMA (UMTS) (Wideband Code Division Multiple Access (Universal Mobile Telecommunications System)), HSPA (High Speed Packet Access), HSDPA (High-Speed Downlink Packet Access), HSUPA (High-Speed Uplink Packet Access), HSPA+ (High Speed Packet Access Plus), UMTS-TDD (Universal Mobile Telecommunications System - Time-Division Duplex), TD-CDMA

(Time Division - Code Division Multiple Access), TD-SCDMA (Time Division - Synchronous Code Division Multiple Access), 3GPP Rel. 8 (Pre-4G) (3rd Generation Partnership Project Release 8 (Pre-4th Generation)), 3GPP Rel. 9 (3rd Generation Partnership Project Release 9), 3GPP Rel. 10 (3rd Generation Partnership Project Release 10) , 3GPP Rel. 11 (3rd Generation Partnership Project Release 11), 3GPP Rel. 12 (3rd Generation Partnership Project Release 12), 3GPP Rel. 13 (3rd Generation Partnership Project Release 13), 3GPP Rel. 14 (3rd Generation Partnership Project Release 14), 3GPP LTE Extra, LTE Licensed-Assisted Access (LAA), UTRA (UMTS Terrestrial Radio Access), E-UTRA (Evolved UMTS Terrestrial Radio Access), LTE Advanced (4G) (Long Term Evolution Advanced (4th Generation)), cdmaOne (2G), CDMA2000 (3G) (Code division multiple access 2000 (Third generation)), EV-DO (Evolution-Data Optimized or Evolution-Data Only), AMPS (1G) (Advanced Mobile Phone System (1st Generation)), TACS/ETACS (Total Access Communication System/Extended Total Access Communication System), D-AMPS (2G) (Digital AMPS (2nd Generation)), PTT (Push-to-talk), MTS (Mobile Telephone System), IMTS (Improved Mobile Telephone System), AMTS (Advanced Mobile Telephone System), OLT (Norwegian for Offentlig Landmobil Telefoni, Public Land Mobile Telephony), MTD (Swedish abbreviation for Mobiltelefonisystem D, or Mobile telephony system D), Autotel/PALM (Public Automated Land Mobile), ARP (Finnish for Autoradiopuhelin, „car radio phone“), NMT (Nordic Mobile Telephony), Hicap (High capacity version of NTT (Nippon Telegraph and Telephone)), CDPD (Cellular Digital Packet Data), Mobitex, DataTAC, iDEN (Integrated Digital Enhanced Network), PDC (Personal Digital Cellular), CSD (Circuit Switched Data), PHS (Personal Handy-phone System), WiDEN (Wideband Integrated Digital Enhanced Network), iBurst, Unlicensed Mobile Access (UMA, also referred to as also referred to as 3GPP Generic Access Network, or GAN standard)), Wireless Gigabit Alliance (WiGig) standard, mmWave standards in general (wireless systems operating at 10-90 GHz and above such as WiGig, IEEE 802.11ad, IEEE 802.11ay, and the like. In addition to the standards listed above, any number of satellite uplink technologies may be used for the uplink transceiver, including, for example, radios compliant with standards issued by the ITU (International Telecommunication Union), or the ETSI (European Telecommunications Standards Institute), among others. The examples provided herein are thus understood as being applicable to various other communication technologies, both existing and not yet formulated. Implementations, components, and details of the

aforementioned protocols may be those known in the art and are omitted herein for the sake of brevity.

The input/output (I/O) interface 718 may include circuitry, such as an external expansion bus (e.g., Universal Serial Bus (USB), FireWire, Thunderbolt, PCI/PCIe/PCIx, etc.), used to connect computer device 700 with external components/devices, such as one or more sensors 714, etc. I/O interface circuitry 718 may include any suitable interface controllers and connectors to interconnect one or more of the processor circuitry 702, memory circuitry 704, flash circuitry 708, communication circuitry 705, and the other components of computing device 700. The interface controllers may include, but are not limited to, memory controllers, storage controllers (e.g., redundant array of independent disk (RAID) controllers, baseboard management controllers (BMCs), input/output controllers, host controllers, etc. The connectors may include, for example, busses (e.g., bus 706), ports, slots, jumpers, interconnect modules, receptacles, modular connectors, etc. The I/O circuitry 718 may couple the computing device 700 with the one or more sensors 714, etc. via a wired connection, such as using USB, FireWire, Thunderbolt, RCA, a video graphics array (VGA), a digital visual interface (DVI) and/or mini-DVI, a high-definition multimedia interface (HDMI), an S-Video, and/or the like.

The one or more sensors 714 may be any device configured to detect events or environmental changes, convert the detected events into electrical signals and/or digital data, and transmit/send the signals/data to the computing device 700. Some of the one or more sensors 714 may be sensors used for providing computer-generated sensory inputs. Some of the one or more sensors 714 may be sensors used for motion and/or object detection. Examples of such one or more sensors 714 may include, inter alia, charged-coupled devices (CCD), Complementary metal–oxide–semiconductor (CMOS) active pixel sensors (APS), lens-less image capture devices/cameras, thermographic (infrared) cameras, Light Imaging Detection And Ranging (LIDAR) systems, and/or the like. In some implementations, the one or more sensors 714 may include a lens-less image capture mechanism comprising an array of aperture elements, wherein light passing through the array of aperture elements define the pixels of an image. In embodiments, the motion detection one or more sensors 714 may be coupled with or associated with light generating devices, for example, one or more infrared projectors to project a grid of infrared light onto a scene, where an infrared camera may record reflected infrared light to compute depth information.

Some of the one or more sensors 714 may be used for position and/or orientation detection, ambient/environmental condition detection, and the like. Examples of such one or more sensors 714 may include, inter alia, microelectromechanical systems (MEMS) with piezoelectric, piezoresistive and/or capacitive components, which may be used to determine environmental conditions or location information related to the computer device 700. In embodiments, the MEMS may include 3-axis accelerometers, 3-axis gyroscopes, and/or magnetometers. In some embodiments, the one or more sensors 714 may also include one or more gravimeters, altimeters, barometers, proximity sensors (e.g., infrared radiation detector(s) and the like), depth sensors, ambient light sensors, thermal sensors (thermometers), ultrasonic transceivers, and/or the like.

Each of these elements, e.g., one or more processors 702, the virtual flash 703, the memory 704, the flash circuitry 708 including the firmware/BIOS images 709, the input/output interface 718, the one or more sensors 714, the communication circuitry 705 including the Tx 711, the NIC 712, the system bus 706, may perform its conventional functions known in the art. In addition, they may be employed to store and host execution of programming instructions implementing various operating system functions and/or applications. The various programming isntructions may be implemented by assembler instructions supported by processor(s) 702 or high-level languages, such as, for example, C, that can be compiled into such instructions. Operations associated with the computing device 700 not implemented in software may be implemented in hardware, e.g., via a hardware accelerator of processor 702.

The number, capability and/or capacity of these elements may vary, depending on the number of other devices the device 700 is configured to support. Otherwise, the constitutions of these elements, except for the teachings of the present disclosure, are known, and accordingly will not be further described.

As described, aspect of the present disclosure may take the form of a computer program product embodied in any tangible or non-transitory medium of expression having computer-usable program code embodied in the medium. **Figure 8** illustrates an example computer-readable non-transitory storage medium that may be suitable for use to store instructions that cause an apparatus, in response to execution of the instructions by the apparatus, to practice selected aspects of the present disclosure. As shown, non-transitory computer-readable storage medium 802 may include a number of programming instructions 804. Programming instructions 804 may be configured to enable a device,

e.g., device 700, in response to execution of the programming instructions, to perform, e.g., various programming operations associated with operating system functions and/or applications. In various embodiments, programming instructions 804 may be the instructions to configure e.g., the firmware of virtual flash 703, to perform the virtual flash operations described herein.

In alternate embodiments, programming instructions 804 may be disposed on multiple computer-readable non-transitory storage media 802 instead. In alternate embodiments, programming instructions 804 may be disposed on computer-readable transitory storage media 802, such as, signals. Any combination of one or more computer usable or computer readable medium(s) may be utilized. The computer-usable or computer-readable medium may be, for example but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, device, or propagation medium. More specific examples (a non- exhaustive list) of the computer-readable medium would include the following: an electrical connection having one or more wires, a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a transmission media such as those supporting the Internet or an intranet, or a magnetic storage device. Note that the computer-usable or computer-readable medium could even be paper or another suitable medium upon which the program is printed, as the program can be electronically captured, via, for instance, optical scanning of the paper or other medium, then compiled, interpreted, or otherwise processed in a suitable manner, if necessary, and then stored in a computer memory. In the context of this document, a computer-usable or computer-readable medium may be any medium that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device. The computer-usable medium may include a propagated data signal with the computer-usable program code embodied therewith, either in baseband or as part of a carrier wave. The computer usable program code may be transmitted using any appropriate medium, including but not limited to wireless, wireline, optical fiber cable, RF, etc.

Computer program code for carrying out operations of the present disclosure may be written in any combination of one or more programming languages, including an object oriented programming language such as Java, Smalltalk, C++ or the like and conventional

procedural programming languages, such as the "C" programming language or similar programming languages. The program code may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider).

Thus example embodiments described include:

Example 1 is an apparatus for computing, comprising: a virtual flash that includes circuitry and firmware in cooperation to facilitate accesses by one or more cores of a computing platform for one or more platform firmware images from one or more flash devices of the computing platform, wherein the facilitation includes return of the one or more platform firmware images for the one or more cores from one or more secured memory areas of the computing platform instead.

Example 2 is example 1, wherein the circuitry comprises synchronous digital circuits including: a frame buffer to receive commands from one of the one or more cores at a clock rate of the core, and buffer the received commands; and command forward circuitry coupled to the frame buffer to retrieve the buffered commands at a local clock rate and forward the retrieved commands for translation, and selective delivery of the translations to complementary virtual flash firmware of the virtual flash or a memory controller of the secured memory areas at the local clock rate.

Example 3 is example 2, wherein the synchronous digital circuits further include command translate circuitry coupled to the command forward circuitry to receive the forwarded commands at the local clock rate, translate the forwarded commands, and selectively forward the translations to the complementary virtual flash firmware of the virtual flash or the memory controller of the secured memory areas at the local clock rate.

Example 4 is example 3, wherein the synchronous digit circuits further comprising a virtual flash firmware interface coupled with the command forward circuitry and the command translate circuitry to interface with the complementary virtual flash firmware of the virtual flash.

Example 5 is example 3, further comprising a memory interface coupled with the command translate circuitry to interface with the memory controller of the secured memory areas at the local clock rate.

Example 6 is example 2, further comprising a core interface coupled with the frame buffer to selectively couple the virtual flash to the one or more cores at the corresponding cores' clock rates.

Example 7 is example 1, wherein the virtual flash further comprises a plurality of registers to store a plurality of operational parameters of the virtual flash.

Example 8 is example 1, wherein the circuitry are implemented using a field programmable gate array (FPGA).

Example 9 is example 8, wherein the apparatus is a system-on-chip having the FPGA and one of the one or more cores, coupled with each other.

Example 10 is example 1, wherein the firmware includes an initialization module to load the one or more platform firmware images into the secured memory areas from the one or more flash devices by retrieving the one or more platform firmware images and storing the one or more platform firmware images into the secured memory areas.

Example 11 is example 10, wherein the one or more platform firmware images in the one or more flash devices are encrypted, and the flash firmware further includes a decryption engine to recover the one or more platform firmware images from the encrypted one or more platform firmware images stored in the one or more flash devices, prior to the one or more platform firmware images being stored into the secured memory areas.

Example 12 is example 10, wherein the firmware includes a synchronizing engine to synchronize the encrypted versions of the one or more platform firmware images stored in the one or more flash devices, with the one or more platform firmware images stored into the secured memory areas, in response to the one or more platform firmware images stored into the secured memory areas having been updated.

Example 13 is a method for computing, comprising: retrieving, with a virtual flash, one or more platform firmware images from one or more flash drives of a computing platform; storing, by the virtual flash, the one or more retrieved platform firmware images in one or more secured memory areas of the computing platform, the one or more secured memory areas being managed by complementary virtual flash firmware of the virtual flash; receiving, by the virtual flash, attempts to access the one or more platform firmware

images by one or more cores of the computing platform from the one or more flash drives; and retrieving and returning the one or more platform firmware images, by the virtual flash, from the one or more secured memory areas managed by the complementary virtual flash firmware of the virtual flash instead.

Example 14 is example 13, wherein the one or more platform firmware images in the one or more flash drives are encrypted, and the method further comprises, on retrieval by the virtual flash from the one or more flash drives, decrypting, by the virtual flash, the one or more encrypted platform firmware images, prior to storing the one or more platform firmware images in the one or more secured memory areas managed by the complementary virtual flash firmware of the virtual flash.

Example 15 is example 13, further comprising synchronizing the encrypted versions of the one or more platform firmware images stored in the one or more flash devices, with the one or more platform firmware images stored into the one or more secured memory areas, when the one or more platform firmware images stored into the one or more secured memory areas are updated.

Example 16 is at least one computer readable medium (CRM) having instructions stored therein, to cause a computing platform, in response to execution of the instructions, to: receive attempts to access one or more platform firmware images by one or more cores of the computing platform from one or more flash drives, with a virtual flash having associated virtual flash firmware; and retrieve and return the one or more platform firmware images from secured memory areas of the computing platform, with the virtual flash firmware, the virtual flash firmware having access control of the secured memory areas of the computing platform.

Example 17 is example 16, wherein the computing platform is further caused to: pre-retrieve the one or more platform firmware images from one or more flash drives of the computing platform, by the virtual flash firmware; and pre-store the one or more pre-retrieved platform firmware images in the secured memory area, by the virtual flash firmware.

Example 18 is example 17, wherein the one or more platform firmware images in the one or more flash drives are encrypted, and the computing platform is further caused to: on pre-retrieval of the one or more encrypted platform firmware images, decrypt, the one or more encrypted platform firmware images, prior to pre-store the one or more platform firmware images in the secured memory area, with the virtual flash firmware.

Example 19 is example 17, wherein the computing platform is further caused to: synchronize the encrypted versions of the one or more platform firmware images stored in the one or more unsecured flash devices, with the one or more platform firmware images stored into the secured memory area, when the one or more platform firmware images stored into the secured memory area are updated by the virtual flash firmware.

Example 20 is a system-on-chip (SOC) comprising: one or more processor cores; and a field programmable gate array (FPGA) coupled with the one or more processor cores, and configured to operate as a virtual flash to facilitate accesses by the one or more cores for one or more basic input/output system (BIOS) images, acting as one or more serial peripheral interface (SPI) flash masters; wherein the FPGA, operating as a virtual flash, returns the one or more BIOS images from one or more secured memory areas co-disposed on a same computing platform with the SoC.

Example 21 is example 20, wherein the FPGA is configured to provide: a frame buffer to receive commands from one of the one or more cores acting as a SPI flash master at a clock rate of the SPI master, and buffer the received commands; and command forward circuitry coupled to the frame buffer to retrieve the buffered commands at a local clock rate and forward the retrieved commands for translation and on translation, selective delivery the translations to associated virtual flash firmware of the virtual flash or a memory controller of the secured memory areas.

Example 22 is example 21, wherein the FPGA is further configured to provide command translate circuitry coupled to the command forward circuitry to receive the forwarded commands at the local clock rate, translate the forwarded commands, and selectively forward the translations to associated virtual flash firmware of the virtual flash or the memory controller of the secured memory areas.

Example 23 is example 22, wherein the FPGA is further configured to provide a virtual flash interface coupled with the command forward circuitry and the command translate circuitry to interface with the associated virtual flash firmware of the virtual flash, and a memory interface coupled with the command translate circuitry to interface with the memory controller of the secured memory areas.

Example 24 is example 21, wherein the FPGA is further configured to provide a core interface coupled with the frame buffer to couple the FPGA to the accessing SPI master.

Example 25 is example 20, wherein the FPGA is further configured to provide a plurality of registers to store a plurality of operational parameters of the FPGA.

The present disclosure is described with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems) and computer program products according to embodiments of the disclosure. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer program instructions. These computer program instructions may be provided to a processor of a general purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

These computer program instructions may also be stored in a computer-readable medium that can direct a computer or other programmable data processing apparatus to function in a particular manner, such that the instructions stored in the computer-readable medium produce an article of manufacture including instruction means which implement the function/act specified in the flowchart and/or block diagram block or blocks.

The computer program instructions may also be loaded onto a computer or other programmable data processing apparatus to cause a series of operational steps to be performed on the computer or other programmable apparatus to produce a computer implemented process such that the instructions which execute on the computer or other programmable apparatus provide processes for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

The flowchart and block diagrams in the figures illustrate the architecture, functionality, and operation of possible implementations of systems, methods and computer program products according to various embodiments of the present disclosure. In this regard, each block in the flowchart or block diagrams may represent a module, segment, or portion of code, which comprises one or more executable instructions for implementing the specified logical function(s). It should also be noted that, in some alternative implementations, the functions noted in the block may occur out of the order noted in the figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustration, and combinations of blocks in

the block diagrams and/or flowchart illustration, can be implemented by special purpose hardware-based systems that perform the specified functions or acts, or combinations of special purpose hardware and computer instructions. As used herein, "computer-implemented method" may refer to any method executed by one or more processors, a computer system having one or more processors, a mobile device such as a smartphone (which may include one or more processors), a tablet, a laptop computer, a set-top box, a gaming console, and so forth.

Embodiments may be implemented as a computer process, a computing system or as an article of manufacture such as a computer program product of computer readable media. The computer program product may be a computer storage medium readable by a computer system and encoding a computer program instructions for executing a computer process.

The corresponding structures, material, acts, and equivalents of all means or steps plus function elements in the claims below are intended to include any structure, material or act for performing the function in combination with other claimed elements are specifically claimed. The description of the present disclosure has been presented for purposes of illustration and description, but is not intended to be exhaustive or limited to the disclosure in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill without departing from the scope and spirit of the disclosure. The embodiment are chosen and described in order to best explain the principles of the disclosure and the practical application, and to enable others of ordinary skill in the art to understand the disclosure for embodiments with various modifications as are suited to the particular use contemplated.

The foregoing description of one or more implementations provides illustration and description, but is not intended to be exhaustive or to limit the scope of embodiments to the precise form disclosed. Modifications and variations are possible in light of the above teachings or may be acquired from practice of various embodiments.

CLAIMS

1. An apparatus for computing, comprising:

a virtual flash that includes circuitry and firmware in cooperation to facilitate accesses by one or more cores of a computing platform for one or more platform firmware images from one or more flash devices of the computing platform, wherein the facilitation includes return of the one or more platform firmware images for the one or more cores from one or more secured memory areas of the computing platform instead.

2. The apparatus of claim 1, wherein the circuitry comprises synchronous digital circuits including:

a frame buffer to receive commands from one of the one or more cores at a clock rate of the core, and buffer the received commands; and

command forward circuitry coupled to the frame buffer to retrieve the buffered commands at a local clock rate and forward the retrieved commands for translation, and selective delivery of the translations to complementary virtual flash firmware of the virtual flash or a memory controller of the secured memory areas at the local clock rate.

3. The apparatus of claim 2, wherein the synchronous digital circuits further include command translate circuitry coupled to the command forward circuitry to receive the forwarded commands at the local clock rate, translate the forwarded commands, and selectively forward the translations to the complementary virtual flash firmware of the virtual flash or the memory controller of the secured memory areas at the local clock rate.

4. The apparatus of claim 3, wherein the synchronous digit circuits further comprising a virtual flash firmware interface coupled with the command forward circuitry and the command translate circuitry to interface with the complementary virtual flash firmware of the virtual flash.

5. The apparatus of claim 3, further comprising a memory interface coupled with the command translate circuitry to interface with the memory controller of the secured memory areas at the local clock rate.

6. The apparatus of claim 2, further comprising a core interface coupled with the frame buffer to selectively couple the virtual flash to the one or more cores at the corresponding cores' clock rates.

7. The apparatus of claim 1, wherein the virtual flash further comprises a plurality of registers to store a plurality of operational parameters of the virtual flash.

8. The apparatus of claim 1, wherein the circuitry are implemented using a field programmable gate array (FPGA).

9. The apparatus of claim 8, wherein the apparatus is a system-on-chip having the FPGA and one of the one or more cores, coupled with each other.

10. The apparatus of claim 1, wherein the firmware includes an initialization module to load the one or more platform firmware images into the secured memory areas from the one or more flash devices, by retrieving the one or more platform firmware images and storing the one or more platform firmware images into the secured memory areas.

11. The apparatus of claim 10, wherein the one or more platform firmware images in the one or more flash devices are encrypted, and the flash firmware further includes a decryption engine to recover the one or more platform firmware images from the encrypted one or more platform firmware images stored in the one or more flash devices, prior to the one or more platform firmware images being stored into the secured memory areas.

12. The apparatus of claim 10, wherein the firmware includes a synchronizing engine to synchronize the encrypted versions of the one or more platform firmware images stored in the one or more flash devices, with the one or more platform firmware images stored into the secured memory areas, in response to the one or more platform firmware images stored into the secured memory areas having been updated.

13. A method for computing, comprising:

retrieving, with a virtual flash, one or more platform firmware images from one or more flash drives of a computing platform;

storing, by the virtual flash, the one or more retrieved platform firmware images in one or more secured memory areas of the computing platform, the one or

more secured memory areas being managed by complementary virtual flash firmware of the virtual flash;

receiving, by the virtual flash, attempts to access the one or more platform firmware images by one or more cores of the computing platform from the one or more flash drives; and

retrieving and returning the one or more platform firmware images, by the virtual flash, from the one or more secured memory areas managed by the complementary virtual flash firmware of the virtual flash instead.

14. The method of claim 13, wherein the one or more platform firmware images in the one or more flash drives are encrypted, and the method further comprises, on retrieval by the virtual flash from the one or more flash drives, decrypting, by the virtual flash, the one or more encrypted platform firmware images, prior to storing the one or more platform firmware images in the one or more secured memory areas managed by the complementary virtual flash firmware of the virtual flash.

15. The method of claim 13, further comprising synchronizing the encrypted versions of the one or more platform firmware images stored in the one or more flash devices, with the one or more platform firmware images stored into the one or more secured memory areas, when the one or more platform firmware images stored into the one or more secured memory areas are updated.

16. At least one computer readable medium (CRM) having instructions stored therein, to cause a computing platform, in response to execution of the instructions, to:

receive attempts to access one or more platform firmware images by one or more cores of the computing platform from one or more flash drives, with a virtual flash having associated virtual flash firmware; and

retrieve and return the one or more platform firmware images from secured memory areas of the computing platform, with the virtual flash firmware, the virtual flash firmware having access control of the secured memory areas of the computing platform.

17. The CRM of claim 16, wherein the computing platform is further caused to:

pre-retrieve the one or more platform firmware images from one or more flash drives of the computing platform, by the virtual flash firmware; and

pre-store the one or more pre-retrieved platform firmware images in the secured memory area, by the virtual flash firmware.

18. The CRM of claim 17, wherein the one or more platform firmware images in the one or more flash drives are encrypted, and the computing platform is further caused to: on pre-retrieval of the one or more encrypted platform firmware images, decrypt, the one or more encrypted platform firmware images, prior to pre-store the one or more platform firmware images in the secured memory area, with the virtual flash firmware.

19. The CRM of claim 17, wherein the computing platform is further caused to: synchronize the encrypted versions of the one or more platform firmware images stored in the one or more unsecured flash devices, with the one or more platform firmware images stored into the secured memory area, when the one or more platform firmware images stored into the secured memory area are updated by the virtual flash firmware.

20. A system-on-chip (SOC) comprising:

one or more processor cores; and

a field programmable gate array (FPGA) coupled with the one or more processor cores, and configured to operate as a virtual flash to facilitate accesses by the one or more cores for one or more basic input/output system (BIOS) images, acting as one or more serial peripheral interface (SPI) flash masters; wherein the FPGA, operating as a virtual flash, returns the one or more BIOS images from one or more secured memory areas co-disposed on a same computing platform with the SoC.

21. The SOC of claim 20, wherein the FPGA is configured to provide:

a frame buffer to receive commands from one of the one or more cores acting as a SPI flash master at a clock rate of the SPI master, and buffer the received commands; and

command forward circuitry coupled to the frame buffer to retrieve the buffered commands at a local clock rate and forward the retrieved commands for translation and on translation, selective delivery the translations to associated virtual flash firmware of the virtual flash or a memory controller of the secured memory areas.

22. The SoC of claim 21, wherein the FPGA is further configured to provide command translate circuitry coupled to the command forward circuitry to receive the forwarded commands at the local clock rate, translate the forwarded commands, and selectively forward the translations to associated virtual flash firmware of the virtual flash or the memory controller of the secured memory areas.

23. The SoC of claim 22, wherein the FPGA is further configured to provide a virtual flash interface coupled with the command forward circuitry and the command translate circuitry to interface with the associated virtual flash firmware of the virtual flash, and a memory interface coupled with the command translate circuitry to interface with the memory controller of the secured memory areas.

24. The SoC of claim 21, wherein the FPGA is further configured to provide a core interface coupled with the frame buffer to couple the FPGA to the accessing SPI master.

25. The SoC of claim 20, wherein the FPGA is further configured to provide a plurality of registers to store a plurality of operational parameters of the FPGA.

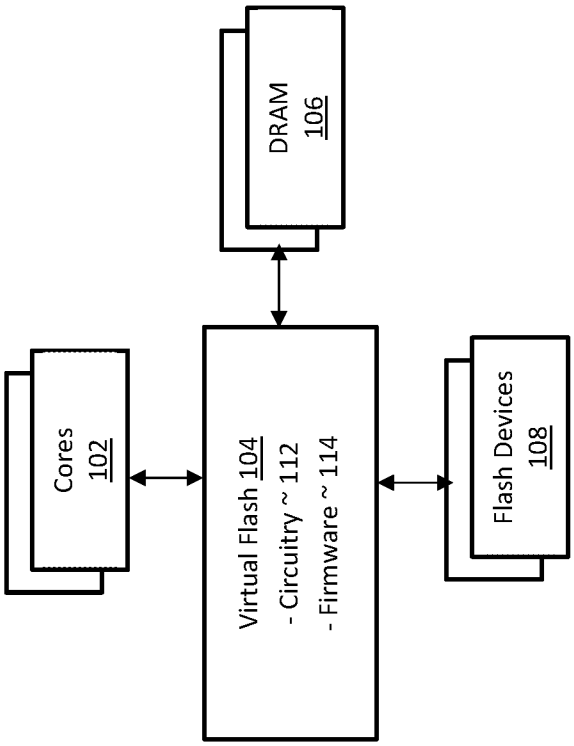

DRAM
106
Cores
102
Virtual Flash 104
- Circuitry ~ 112
- Firmware ~ 114
Flash Devices
108
100

Figure 1

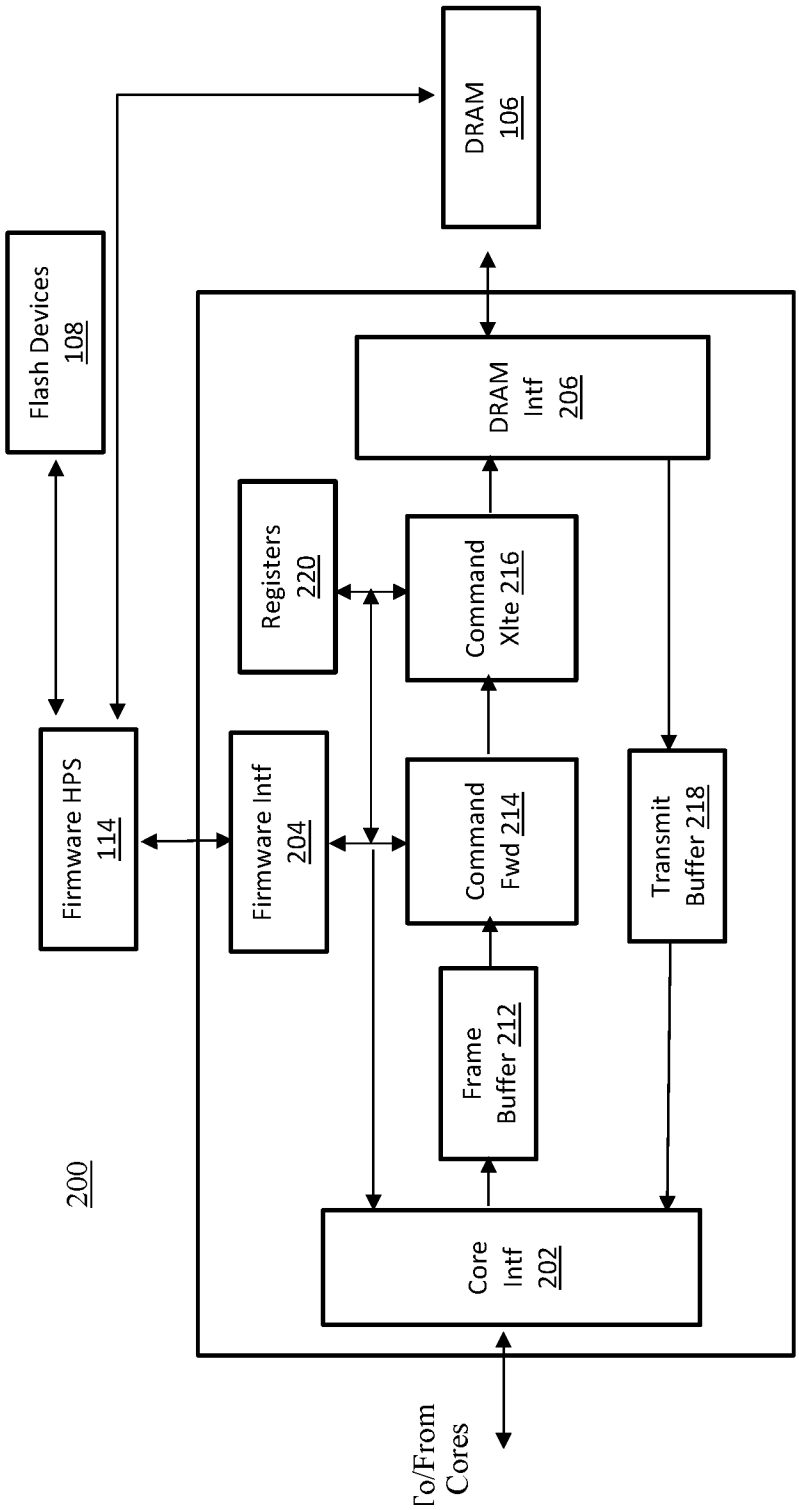

DRAM 106
Flash Devices 108
DRAM Intf 206
Registers 220
Command Xlte 216
Firmware HPS 114
Firmware Intf 204
Command Fwd 214
Transmit Buffer 218
Frame Buffer 212
Core Intf 202
200
To/From Cores

Figure 2

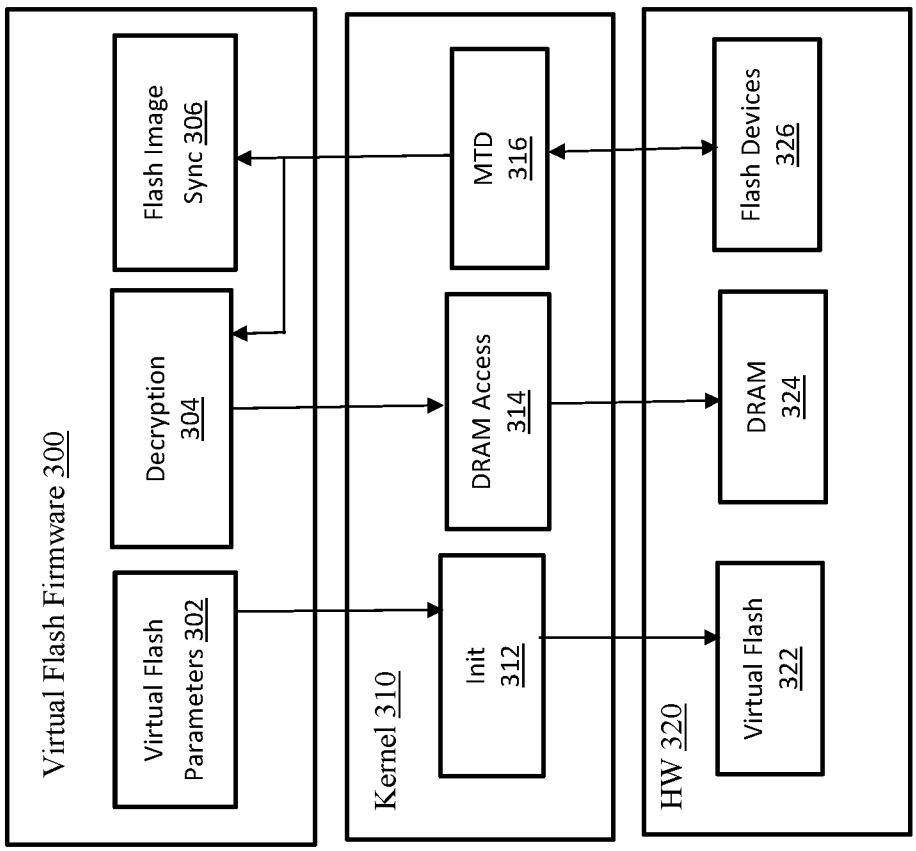
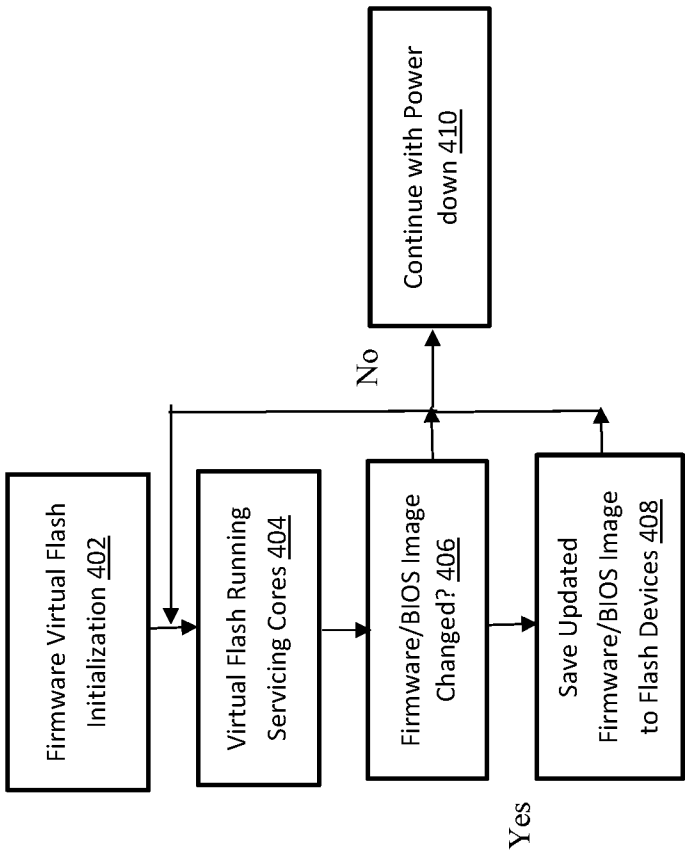

Virtual Flash Firmware 300
Virtual Flash Parameters 302
Decryption 304
Flash Image Sync 306
Kernel 310
Init 312
DRAM Access 314
MTD 316
HW 320
Virtual Flash 322
DRAM 324
Flash Devices 326

Figure 3


Firmware Virtual Flash Initialization 402
Virtual Flash Running Servicing Cores 404
Firmware/BIOS Image Changed? 406
Save Updated Firmware/BIOS Image to Flash Devices 408
Continue with Power down 410
No
Yes

400

Figure 4

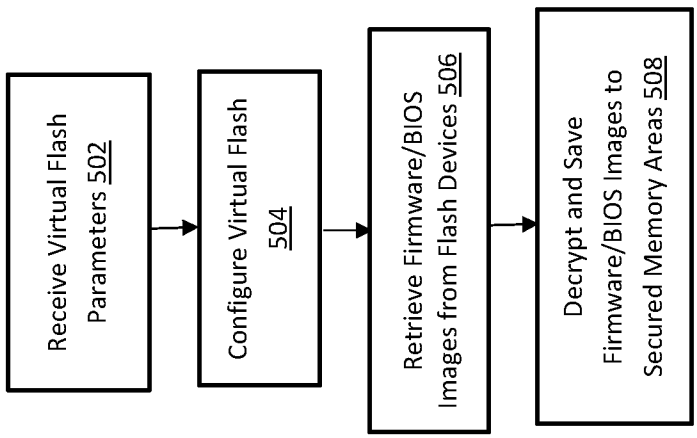

Receive Virtual Flash Parameters 502
Configure Virtual Flash 504
Retrieve Firmware/BIOS Images from Flash Devices 506
Decrypt and Save Firmware/BIOS Images to Secured Memory Areas 508
500

Figure 5

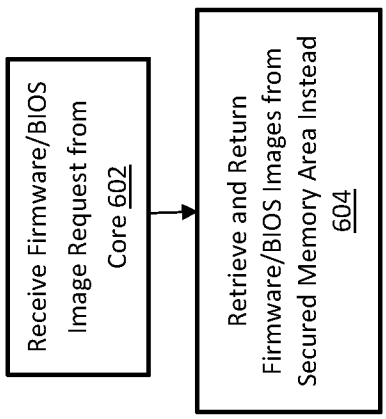

Receive Firmware/BIOS Image Request from Core 602
Retrieve and Return Firmware/BIOS Images from Secured Memory Area Instead 604
600

Figure 6

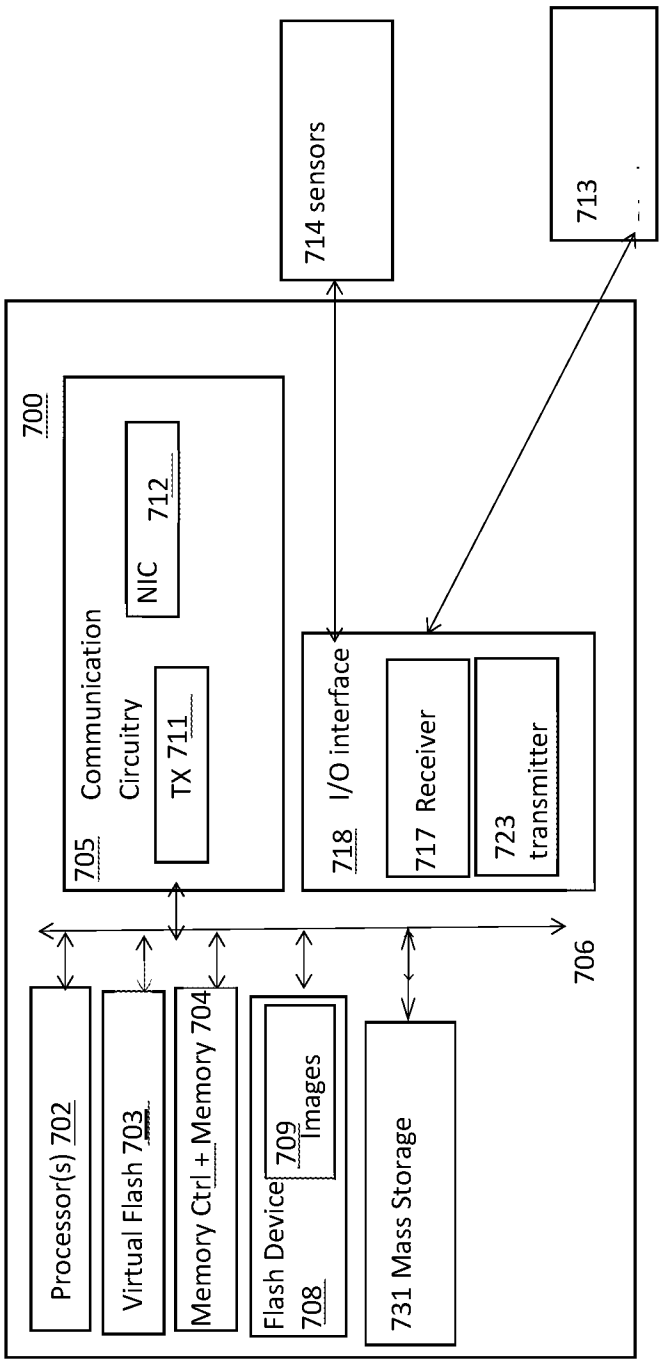

714 sensors
713
700
705 Communication Circuitry
NIC 712
TX 711
718 I/O interface
717 Receiver
723 transmitter
706
Processor(s) 702
Virtual Flash 703
Memory Ctrl + Memory 704
Flash Device 708
709 Images
731 Mass Storage

Figure 7

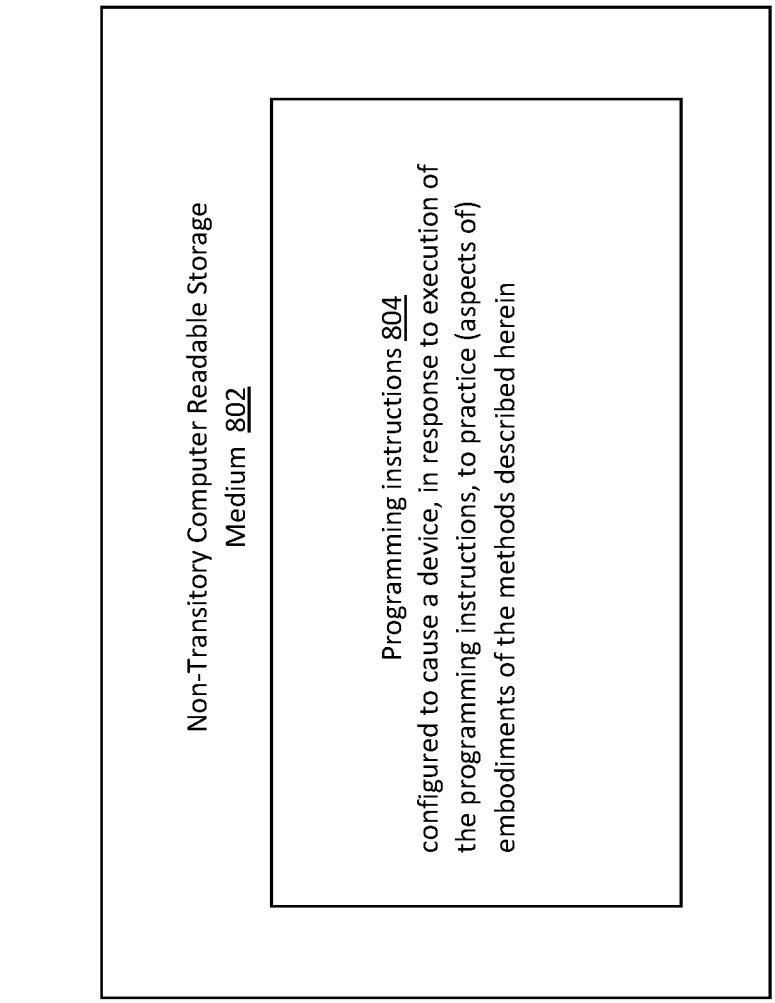

800
Non-Transitory Computer Readable Storage Medium 802
Programming instructions 804
configured to cause a device, in response to execution of the programming instructions, to practice (aspects of) embodiments of the methods described herein

Figure 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2019/074776

A. CLASSIFICATION OF SUBJECT MATTER

G06F 21/00(2013.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNPAT, WPI, EPODOC, CNKI, IEEE, GOOGLE: BIOS, firmware, image, security, safe, FPGA, SoC, DRAM, memory, retrieve, return, encrypted

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2017147356 A1 (INTEL CORPORATION) 25 May 2017 (2017-05-25) description, paragraphs [0017], [0020], [0036]-[0037]	1-25
X	CN 103593622 A (INSPUR GROUP CO., LTD.) 19 February 2014 (2014-02-19) description, paragraphs [0013]-[0015]	1-25
A	CN 102663301 A (BEIJING BNC TECHNOLOGIES CO., LTD.) 12 September 2012 (2012-09-12) the whole document	1-25
A	CN 106997438 A (SHANDONG ITL DATA TECHNIQUE CO., LTD.) 01 August 2017 (2017-08-01) the whole document	1-25
A	US 2015149815 A1 (AMERICAN MEGATRENDS, INC.) 28 May 2015 (2015-05-28) the whole document	1-25

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
10 October 2019	**29 October 2019**

Name and mailing address of the ISA/CN	Authorized officer
National Intellectual Property Administration, PRC **6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing** **100088** **China**	**SUN,Taomin**
Facsimile No. **(86-10)62019451**	Telephone No. **86-(10)-53961420**

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2019/074776

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
US 2017147356 A1	25 May 2017	CN 106462707 A	22 February 2017
		KR 20160126067 A	01 November 2016
		WO 2015165000 A1	05 November 2015
		TW 201602786 A	16 January 2016
		EP 3138040 A1	08 March 2017
CN 103593622 A	19 February 2014	None	
CN 102663301 A	12 September 2012	None	
CN 106997438 A	01 August 2017	None	
US 2015149815 A1	28 May 2015	None	