



(51) International Patent Classification:
G06F 12/14 (2006.01)

(21) International Application Number:
PCT/CN2016/070868

(22) International Filing Date:
14 January 2016 (14.01.2016)

(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): YAO, Jiewen Jacques [CN/CN]; Room 404, Building 3, 818 Ming Sheng Road, Shanghai 200135 (CN). ZIMMER, Vincent J. [US/US]; 1937 S. 369th Street, Federal Way, Washington 98003 (US). COURY, Bassam N. [US/US]; 2800 Center Drive N., Dupont, Washington 98327 (US).

(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, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, 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).

Published:

— with international search report (Art. 21(3))

(54) Title: SECURE COMMUNICATION CHANNEL FOR SYSTEM MANAGEMENT MODE

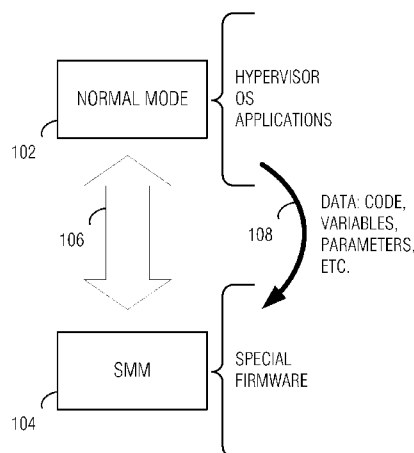


FIG. 1

(57) Abstract: Embodiments are directed to securing system management mode (SMM) in a computer system. A CPU is configurable to execute first code in a normal mode, and second code in a SSM. A SMM control engine is operative to transition the CPU from the normal mode to the SMM in response to a SMM transition call, and to control access by the CPU in the SMM to data from an originator of the SMM transition call. The access is controlled based on an authorization state assigned to the SMM transition call. An authorization engine is operative to perform authentication of the originator of the SMM transition call and to assign the authorization state based on an authentication result. The CPU in the SMM is prevented from accessing the data in response to the authentication result being a failure of authentication.

SECURE COMMUNICATION CHANNEL FOR SYSTEM MANAGEMENT MODE

5

TECHNICAL FIELD

[0001] Embodiments described herein generally relate to information processing and security, more particularly, to assuring the authenticity of code to be executed in the system management mode (SMM) of a computer system.

10

BACKGROUND

[0002] In personal computer (PC) systems, system management mode (SMM) is an operating mode of x86-family central processor units (CPUs) in which all normal execution, including the operating system, is suspended and special system software is executed with high privileges. SMM is generally used handling system-wide functions like power management, system hardware control, operating system (OS) validation, cryptography, or proprietary original equipment manufacturer (OEM)-designed code. SMM is generally used only by system firmware, not by applications or general-purpose systems software. A benefit of SMM is that it offers a distinct and isolated computing environment that operates independently of the operating system, hypervisor, or application software.

[0003] However, in recent years SMM has increasingly become an attack vector for malware, such as rootkits. Even though the executable code in the SMM may be firmware-based and error-free, a variety of techniques for tampering with the operation of the SMM have been seen. For instance, data objects (such as variables, parameter values, function calls, etc.) may be passed to the SMM by the OS, and these data objects may be exploited to gain access to the SMM's high privilege level. For instance, commands may be sent to the SMM that may cause the SMM to perform a series of actions to compromise security features of the OS. In another example, techniques as buffer overflow attack may be used to inject unexpected code into the SMM. These vulnerabilities are further compounded by the fact that SMM operates outside the reach of any anti-malware programs installed and operational under the

control of the operating system. Solutions are needed to reduce the susceptibility of the SMM to attack.

BRIEF DESCRIPTION OF THE DRAWINGS

- 5 **[0004]** In the drawings, which are not necessarily drawn to scale, like numerals may describe similar components in different views. Like numerals having different letter suffixes may represent different instances of similar components. Some embodiments are illustrated by way of example, and not limitation, in the figures of the accompanying drawings.
- 10 **[0005]** FIG. 1 is a high-level block diagram illustrating a basic relationship between operating modes of a central processing unit (CPU) according to embodiments described herein.
- [0006]** FIG. 2 is a block diagram illustrating a computer system in the example form of a general-purpose machine.
- 15 **[0007]** FIG. 3 is a diagram illustrating an exemplary hardware and software architecture of a computing device such as the one depicted in FIG. 2, in which various interfaces between hardware components and software components are shown.
- [0008]** FIG. 4 is a block diagram illustrating processing devices according to some embodiments.
- 20 **[0009]** FIG. 5 is a block diagram illustrating example components that may be implemented on a computer system, such as the computer system described with reference to FIGs. 2-4, according to some embodiments.
- [0010]** FIG. 6 is a flow diagram illustrating a process of transitioning a CPU
- 25 mode from normal mode to SMM according to some embodiments.

DETAILED DESCRIPTION

- [0011]** Aspects of the embodiments are directed to securing system management mode (SMM) in a computer system. The computer system may be
- 30 one physical machine, or may be distributed among multiple physical machines, such as by role or function, or by process thread in the case of a cloud computing distributed model. In various embodiments certain operations may run in virtual machines that in turn are executed on one or more physical machines. It will be

understood by persons of skill in the art that features of the invention may be realized by a variety of different suitable machine implementations.

[0012] FIG. 1 is a high-level block diagram illustrating a basic relationship between operating modes of a central processing unit (CPU) according to
5 embodiments described herein. As depicted, there are two operating modes that are the subject of the present disclosure: Normal mode 102, and SMM 104. There may be additional operating modes for the CPU, and a practical computer system may support many additional operations; but for the sake of clarity the two modes of interest are detailed. In normal mode 102 the CPU executes the
10 code of a hypervisor (e.g., virtual machine monitor) when the computer system is configured to run virtual machines. In addition, in normal mode 102 the computer system executes one or more operating systems, including a variety of device and system drivers, and application programs. SMM 104 is reserved for the execution of specialized firmware, typically for the purpose of configuring
15 the computer system, checking the integrity of software to be executed in normal mode 102, and various other purposes; however, as discussed above, SMM 104 may be compromised, which presents a dangerous scenario given the high degree of privilege that SMM 104 is granted in the computer system.

[0013] As represented at 106, the CPU may transition from one mode to
20 another. The transition may be effected in various ways, but most notably for the present discussion, the transition from normal mode 102 to SMM 104 may be called by a program, such as a system driver, or some other portion of an operating system, for instance, running in normal mode 102. In general, the call to transition from normal mode 102 to SMM 104 may be accompanied by a
25 passing of some data, e.g., code, variables, parameters, and the like, that may affect or even direct the functionality of SMM 104.

[0014] FIG. 2 is a block diagram illustrating a computer system in the example form of a general-purpose machine. In certain embodiments, programming of the computer system 200 according to one or more particular algorithms
30 produces a special-purpose machine upon execution of that programming. In a networked deployment, the computer system 200 may operate in the capacity of either a server or a client machine in server-client network environments, or it may act as a peer machine in peer-to-peer (or distributed) network environments. The computer system 200 may take any suitable form factor, such as a personal

computer (PC) workstation, a server, whether rack-mounted, or stand-alone, a mainframe computer, a cluster computing system, or the like, a set-top box, as well as a mobile or portable computing system, such as a laptop/notebook PC, an onboard vehicle system, wearable device, a tablet PC, a hybrid tablet, a personal
5 digital assistant (PDA), a mobile telephone or, more generally, any machine capable of executing instructions (sequential or otherwise) that specify actions to be taken by that machine.

[0015] Example computer system 200 includes at least one processor 202 (e.g., a central processing unit (CPU), a graphics processing unit (GPU) or both,
10 processor cores, compute nodes, etc.), a main memory 204 and a static memory 206, which communicate with each other via a link 208 (e.g., bus). The computer system 200 may further include a video display unit 210, an alphanumeric input device 212 (e.g., a keyboard), and a user interface (UI) navigation device 214 (e.g., a mouse). In an embodiment, the video display unit
15 210, input device 212 and UI navigation device 214 are incorporated into a touch screen display. The computer system 200 may additionally include a storage device 216 (e.g., a drive unit), a signal generation device 218 (e.g., a speaker), a network interface device (NID) 220, and one or more sensors (not shown), such as a global positioning system (GPS) sensor, compass, accelerometer, or other
20 sensor.

[0016] The storage device 216 includes a machine-readable medium 222 on which is stored one or more sets of data structures and instructions 224 (e.g., software) embodying or utilized by any one or more of the methodologies or functions described herein. The instructions 224 may also reside, completely or
25 at least partially, within the main memory 204, static memory 206, and/or within the processor 202 during execution thereof by the computer system 200, with the main memory 204, static memory 206, and the processor 202 also constituting machine-readable media.

[0017] While the machine-readable medium 222 is illustrated in an example
30 embodiment to be a single medium, the term “machine-readable medium” may include a single medium or multiple media (e.g., a centralized or distributed database, and/or associated caches and servers) that store the one or more instructions 224. The term “machine-readable medium” shall also be taken to include any tangible medium that is capable of storing, encoding or carrying

instructions for execution by the machine and that cause the machine to perform any one or more of the methodologies of the present disclosure or that is capable of storing, encoding or carrying data structures utilized by or associated with such instructions. The term “machine-readable medium” shall accordingly be
5 taken to include, but not be limited to, solid-state memories, and optical and magnetic media. Specific examples of machine-readable media include non-volatile memory, including but not limited to, by way of example, semiconductor memory devices (e.g., electrically programmable read-only memory (EPROM), electrically erasable programmable read-only memory
10 (EEPROM)) and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks.

[0018] NID 220 according to various embodiments may take any suitable form factor. In one such embodiment, NID 220 is in the form of a network
15 interface card (NIC) that interfaces with processor 202 via link 208. In one example, link 208 includes a PCI Express (PCIe) bus, including a slot into which the NIC form-factor may removably engage. In another embodiment, NID 220 is a network interface circuit laid out on a motherboard together with local link circuitry, processor interface circuitry, other input/output circuitry, memory
20 circuitry, storage device and peripheral controller circuitry, and the like. In another embodiment, NID 220 is a peripheral that interfaces with link 208 via a peripheral input/output port such as a universal serial bus (USB) port. NID 220 transmits and receives data over transmission medium 226, which may be wired or wireless (e.g., radio frequency, infra-red or visible light spectra, etc.), fiber
25 optics, or the like.

[0019] FIG. 3 is a diagram illustrating an exemplary hardware and software architecture of a computing device such as the one depicted in FIG. 2, in which various interfaces between hardware components and software components are shown. As indicated by HW, hardware components are represented below the
30 divider line, whereas software components denoted by SW reside above the divider line. On the hardware side, processing devices 302 (which may include one or more microprocessors, digital signal processors, etc., each having one or more processor cores, are interfaced with memory management device 304 and system interconnect 306. Memory management device 304 provides mappings

between virtual memory used by processes being executed, and the physical memory. Memory management device 304 may be an integral part of a central processing unit which also includes the processing devices 302.

[0020] Interconnect 306 includes a backplane such as memory, data, and control lines, as well as the interface with input/output devices, e.g., PCI, USB, etc. Memory 308 (e.g., dynamic random access memory - DRAM) and non-volatile memory 309 such as flash memory (e.g., electrically-erasable read-only memory – EEPROM, NAND Flash, NOR Flash, etc.) are interfaced with memory management device 304 and interconnect 306 via memory controller 310. This architecture may support direct memory access (DMA) by peripherals in some embodiments. I/O devices, including video and audio adapters, non-volatile storage, external peripheral links such as USB, Bluetooth, etc., as well as network interface devices such as those communicating via Wi-Fi or LTE-family interfaces, are collectively represented as I/O devices and networking 312, which interface with interconnect 306 via corresponding I/O controllers 314.

[0021] On the software side, a pre-operating system (pre-OS) environment 316, which is executed at initial system start-up and is responsible for initiating the boot-up of the operating system. One traditional example of pre-OS environment 316 is a system basic input/output system (BIOS). In present-day systems, a unified extensible firmware interface (UEFI) is implemented. Pre-OS environment 316, is responsible for initiating the launching of the operating system, but also provides an execution environment for embedded applications according to certain aspects of the invention.

[0022] System management mode (SMM) represents code executed in a special-purpose operating mode of processing devices 302 that is provided for handling system-wide functions like power management, system hardware control, or proprietary OEM designed code. In general, SMM is intended for use by system firmware, not by applications software or general-purpose systems software, though SMM may be called by a system management interrupt (SMI) initiated by the software stack. SMM provides a distinct and isolated processor environment that operates transparently to the operating system or executive and software applications. Processing devices 302 execute the SMM code in a separate address space that is inaccessible to other operating modes of the processing devices 302.

[0023] Operating system (OS) 318 provides a kernel that controls the hardware devices, manages memory access for programs in memory, coordinates tasks and facilitates multi-tasking, organizes data to be stored, assigns memory space and other resources, loads program binary code into memory, initiates execution of the application program which then interacts with the user and with hardware devices, and detects and responds to various defined interrupts. Also, operating system 318 provides device drivers, and a variety of common services such as those that facilitate interfacing with peripherals and networking, that provide abstraction for application programs so that the applications do not need to be responsible for handling the details of such common operations. Operating system 318 additionally provides a graphical user interface (GUI) that facilitates interaction with the user via peripheral devices such as a monitor, keyboard, mouse, microphone, video camera, touchscreen, and the like.

[0024] Runtime system 320 implements portions of an execution model, including such operations as putting parameters onto the stack before a function call, the behavior of disk input/output (I/O), and parallel execution-related behaviors. Runtime system 320 may also perform support services such as type checking, debugging, or code generation and optimization.

[0025] Libraries 322 include collections of program functions that provide further abstraction for application programs. These include shared libraries, dynamic linked libraries (DLLs), for example. Libraries 322 may be integral to the operating system 318, runtime system 320, or may be added-on features, or even remotely-hosted. Libraries 322 define an application program interface (API) through which a variety of function calls may be made by application programs 324 to invoke the services provided by the operating system 318. Application programs 324 are those programs that perform useful tasks for users, beyond the tasks performed by lower-level system programs that coordinate the basis operability of the computing device itself.

[0026] FIG. 4 is a block diagram illustrating processing devices 302 according to some embodiments. CPU 410 may contain one or more processing cores 412. As an illustrative example, CPU 410 may be a x86-type of processor. Processing devices 302 may also include a graphics processing unit (GPU) 414. In these embodiments, GPU 414 may be a specialized co-processor that offloads certain computationally-intensive operations, particularly those associated with

graphics rendering, from CPU 410. Notably, CPU 410 and GPU 414 generally work collaboratively, sharing access to memory resources, I/O channels, etc.

[0027] Processing devices 302 also include caretaker processor 416. Caretaker processor 416 generally does not participate in the processing work to carry out software code as CPU 410 and GPU 414 do. In some embodiments, caretaker processor 416 does not share memory space with CPU 410 and GPU 414, and is therefore not arranged to execute operating system or application programs. Instead, caretaker processor 416 may execute dedicated firmware that supports the technical workings of CPU 410, GPU 414, and other components of the computer system. In some embodiments, caretaker processor is implemented as a microcontroller device, which may be physically present on the same integrated circuit die as CPU 410, or may be present on a distinct integrated circuit die. Caretaker processor 416 may also include a dedicated set of I/O facilities to enable it to communicate with external entities. In one type of embodiment, caretaker processor 416 is implemented using a manageability engine (ME) or platform security processor (PSP).

[0028] FIG. 5 is a block diagram illustrating example components that may be implemented on a computer system, such as the computer system and components described with reference to FIGs. 2-4, according to various embodiments. Examples, as described herein, may include, or may operate on, logic or a number of components, modules, or engines, which for the sake of consistency are termed engines, although it will be understood that these terms may be used interchangeably. Engines may be hardware, software, or firmware communicatively coupled to one or more processors in order to carry out the operations described herein. Engines may be hardware engines, and as such engines may be considered tangible entities capable of performing specified operations and may be configured or arranged in a certain manner. In an example, circuits may be arranged (e.g., internally or with respect to external entities such as other circuits) in a specified manner as an engine. In an example, the whole or part of one or more computer systems (e.g., a standalone, client or server computer system) or one or more hardware processors may be configured by firmware or software (e.g., instructions, an application portion, or an application) as an engine that operates to perform specified operations. In an example, the software may reside on a machine-readable medium. In an

example, the software, when executed by the underlying hardware of the engine, causes the hardware to perform the specified operations. Accordingly, the term hardware engine is understood to encompass a tangible entity, be that an entity that is physically constructed, specifically configured (e.g., hardwired), or temporarily (e.g., transitorily) configured (e.g., programmed) to operate in a specified manner or to perform part or all of any operation described herein. With reference to FIG. 3, for instance, an engine may include one, or any combination, of the blocks depicted, so long as at least one block from the HW side is included.

5 [0029] Considering examples in which engines are temporarily configured, each of the engines need not be instantiated at any one moment in time. For example, where the engines comprise a general-purpose hardware processor configured using software; the general-purpose hardware processor may be configured as respective different engines at different times. Software may accordingly configure a hardware processor, for example, to constitute a particular engine at one instance of time and to constitute a different engine at a different instance of time.

15 [0030] Turning now to the details of the example system, the hardware and software components of a host computer system are operative (e.g., configured, programmed, arranged, etc., to be operational when activated), to implement system 500. System 500 includes SMM transition call originator engine 502. In various embodiments, SMM transition call originator engine 502 is an operative portion of the OS, such as a system or device driver. SMM transition call originator engine 502 initiates the call to transition the CPU to operate in SMM from the normal mode, and passes data, such as operational parameters, etc. to be accessed by the CPU in its SMM operation. Notably, SMM transition call originator engine 502 has a digital signature 504, such as one based on a public-key cryptosystem. Digital signature 504 may be used in conjunction with a digital certificate to prove the identity of SMM transition call originator engine 502.

25 [0031] SMM calling engine 506, according to some embodiments, is a driver, or kernel-level agent implemented as part of the operating system. In the embodiment depicted, SMM calling engine 506 performs certain operations on behalf of SMM transition call originator engine 502 to effect the transition of the

CPU from normal mode of operation to SMM. SMM calling engine 506 may include firmware interface engine 508 that is operative to write data to one or more buffers in shared data store 512, that initiate the mode transition. In related embodiments, SMM calling engine 506 includes signing engine 510, which is
5 operative to append digital signature 504 to data being passed to the CPU for SMM operation.

[0032] In some embodiments, data store 512 may be implemented as specially-allocated memory space in RAM, NVRAM, disk, or on a remote machine, along with the interconnect(s) and other components suitable for implementing the
10 data store 512.

[0033] In an example operation, firmware interface engine 508 writes data to be passed to SMM to data communication buffer 514, and writes to system management interrupt (SMI) control register 516 to trigger the SMI to instantiate the mode transition. In an example, SMI control register has a defined address at
15 0xB2. In addition, signing engine 510 appends the digital signature of SMM transition call originator engine 502 to the data to be transferred in data communication buffer 514. In a related embodiment, signing engine 510 retrieves, or generates, a digital certificate, and writes the digital certificate to data communication buffer 514.

20 [0034] In a related embodiment, SMM transition call originator engine 502 and SMM calling engine 506 may be implemented as a single construct, as indicated at 505.

[0035] The triggering of the SMI activates SMM control engine 518, which is operative to conditionally carry out the mode transition from normal mode to
25 SMM. In the embodiment depicted, SMM control engine 518 includes communication handler engine 520, and SMM code execution engine 522. Communication handler engine 520 responds to the SMI by transitioning the CPU to SMM. However, before communication handler engine 520 allows SMM access to the data stored in data communication buffer 514,
30 communication handler engine 520 calls authorization engine 524 to perform authentication of SMM transition call originator 502 based on the digital signature 504 or digital certificate, according to the various embodiments. In an example, communication handler engine 520 of SMM control engine 518 writes to the digital signature 504, digital certificate, or other proof of identity of SMM

transition call originator engine 502 to interface buffer 526 in shared data store 512.

[0036] Authorization engine 524 may be implemented at least in part by a caretaker processor according to some embodiments. It includes verification engine 528, and reference database 530. Verification engine 528 accesses the proof of identity, e.g., digital signature 504, digital certificate, etc., stored in engine interface buffer 526, and compares it against reference information in database 530.

[0037] Database 530 may include a whitelist and a blacklist of known entities. Identifications of trusted entities may be stored in the whitelist; whereas known bad entities, or known risky entities, such as those whose digital certificates may have expired, or who may have compromised their private key, for example, may be stored in a blacklist. In an embodiment, the contents of database 530 are provisioned at the time of manufacture of the computer system. In a related embodiment, authorization engine 524 has access to an external communication channel, such as a network interface device, and uses the channel to connect with a database update service with such information as new trusted entities, new revocations of signatures/certificates, for example.

[0038] In an example, an authentication result may be achieved when the identity in question is confirmed to be in the whitelist and absent from the blacklist. In a related embodiment, database 530 receives periodic updates from a trusted authority via secure external communication channel accessible to authorization engine 524. The result of authentication produces an authorization state for the SMM transition call. This authorization state may be communicated to data communication buffer 514, for example, or by any other suitable channel, so that it may be accessed by communication handler engine 520.

[0039] Upon confirmation that the SMM transition call is authorized, communication handler engine 520 permits access to the data in data communication buffer for the CPU running in SMM, and code execution engine 522 is called to read, and operate upon, the data.

[0040] In the opposite case, if authentication by verification engine 528 fails, an authorization state indicating the failure may be reported to communication handler engine 520, in which case communication handler engine 520 will

prevent the CPU in SMM from accessing the data in data communication buffer 514.

[0041] In a related embodiment, authorization engine 524 logs the authentication result, whatever it may be, for future reference.

5 **[0042]** FIG. 6 is a flow diagram illustrating a process of transitioning a CPU mode from normal mode to SMM according to some embodiments. In some embodiments, the entire process is carried out by a single computer system; in other embodiments, certain operations may be distributed between multiple computer systems using client-server, peer-peer, cloud, or other distributed
10 computing model. As shown in the flow diagram, groups of operations are organized in three columns corresponding to operations carried out by the CPU in the normal mode under the control of the OS (indicated at 600), operations performed by the CPU in the SMM domain (620), and operations by the caretaker processor (640).

15 **[0043]** At operation 602, the buffer in shared memory space for passing data from the OS domain 600 to the SMM domain 620 is configured. In an embodiment, an OS kernel-level entity, such as a driver process, uses an advanced configuration and power interface (ACPI) of the computer system to allocate a portion of RAM for the data communications buffer. At 604, the OS
20 kernel entity writes data to the established buffer that is meant to be passed to the CPU following transition to SMM. At 606, the OK kernel-level entity appends a digital certificate, digital signature, or some other proof of authenticity, to the data in the buffer. In a related embodiment, as depicted, a unique value, such as a nonce, is also appended to protect against replay events – attacks or errors.

25 **[0044]** At 608, the SMI is triggered by the kernel-based entity (e.g., writing to address 0xb2), which causes the CPU to transition to SMM.

[0045] The process advances to block 622, where the communications handler of the SMM invokes an authentication process that is carried out by caretaker processor domain 640. At 642, the caretaker processor domain checks the
30 authentication policy settings. This operation accommodates an embodiment where there may be a master setting in the caretaker processor domain to either require, or not require, authentication for data passed to SMM. Accordingly, at decision 644, if authentication is not called for, the process advances to 652, where the state is set equivalent to a passing of the authentication. Thus, the

SMM domain is notified of the positive authentication (e.g., that the passing of the data to the SMM is authorized), and the SMM is permitted to read the buffer at 624. As a result of the read operation, the process proceeds to 610, where the SMM is transitioned back to normal mode with a successful indication of SMM operation.

[0046] Returning to decision 644, if authentication is called for, the process advances to block 646, where the caretaker processor domain 640 reads the digital certificate, digital signature, or other proof of identity, from the buffer. At decisions 648 and 650, the digital certificate, digital signature, or other proof of identity is checked against a blacklist BL and a whitelist WL. As depicted the case where the digital certificate, digital signature, or other proof of identity is absent from the blacklist and present on the whitelist results in a passing authentication result at 652. In any other scenario, the authentication is deemed a failure at 654, which is logged at 656, and reported to the SMM domain at 626, and further to the normal mode at 612. At 614, the OS kernel-level entity may request more details on the failure of authentication from the caretaker processor domain. In an example, in response to the details, corrective measures may be taken, such as performing a rootkit scan, updating a driver, etc.

[0047] Additional Notes & Examples:

[0048] Example 1 is a system for securing system management mode (SMM) in a computer system that includes computing hardware (HW), the system comprising: a central processing unit (CPU) of the HW that, when operational, is configurable to execute first code in a normal mode, and second code in a SSM; a SMM control engine implementable at least in part on the HW to transition the CPU from the normal mode to the SMM in response to a SMM transition call, and to control access by the CPU in the SMM to data from an originator of the SMM transition call based on an authorization state assigned to the SMM transition call; and an authorization engine implementable at least in part on the HW to perform authentication of the originator of the SMM transition call in response to the SMM transition call and to assign the authorization state based on an authentication result; wherein the SMM control engine is to prevent the CPU in the SMM from accessing the data in response to the authentication result being a failure of authentication

[0049] In Example 2, the subject matter of Example 1 optionally includes, wherein the first code includes an operating system (OS), and wherein the second code includes SMM firmware.

5 [0050] In Example 3, the subject matter of any one or more of Examples 1–2 optionally include, wherein the SMM transition call causes generation of a system management interrupt in response to writing of the data in a data communication buffer accessible to the SMM control engine.

10 [0051] In Example 4, the subject matter of any one or more of Examples 1–3 optionally include, wherein the SMM control engine is to invoke the authorization engine to perform the authentication in response to the SMM transition call.

[0052] In Example 5, the subject matter of any one or more of Examples 1–4 optionally include, wherein the originator of the SMM transition call is a component of the first code.

15 [0053] In Example 6, the subject matter of any one or more of Examples 1–5 optionally include, wherein the originator of the SMM transition call is a driver engine incorporated into an operating system, wherein the operating system is a portion of the first code.

20 [0054] In Example 7, the subject matter of any one or more of Examples 1–6 optionally include, further comprising: a SMM calling engine implementable at least in part on the HW to access the data from the originator of the SMM transition call and, in response, to effect the transition of the CPU from the normal mode to the SMM.

25 [0055] In Example 8, the subject matter of any one or more of Examples 1–7 optionally include, further comprising: a SMM calling engine implementable at least in part on the HW to access: the data from the originator of the SMM transition call; and a digital signature associated with the originator of the SMM transition call; and wherein the SMM calling engine is further to append the digital signature to the data from the originator and to effect the transition of the
30 CPU from the normal mode to the SMM.

[0056] In Example 9, the subject matter of any one or more of Examples 1–8 optionally include, wherein the authorization engine is to perform the authentication based on a validation of a digital certificate of the originator of the SMM transition call.

[0057] In Example 10, the subject matter of any one or more of Examples 1–9 optionally include, wherein the authorization engine is to perform the authentication based on a proof of identity of the originator of the SMM transition call.

5 [0058] In Example 11, the subject matter of any one or more of Examples 1–10 optionally include, wherein the authorization engine is to perform the authentication based on a comparison of an identification of the originator of the SMM transition call against a list of trusted identifications.

[0059] In Example 12, the subject matter of any one or more of Examples 1–11
10 optionally include, wherein the authorization engine is implementable on a caretaker processor of the HW that is isolated from the CPU and is to execute dedicated firmware code.

[0060] Example 13 is at least one computer-readable medium comprising instructions that, when executed on computing hardware (HW) including a
15 central processing unit (CPU) that is operative in a normal mode and a system management mode (SMM), cause the computing hardware to: execute, by the CPU, first code in a normal mode, and second code in a SSM; transition the CPU from the normal mode to the SMM in response to a SMM transition call, and to control access by the CPU in the SMM to data from an originator of the
20 SMM transition call based on an authorization state assigned to the SMM transition call; perform authentication of the originator of the SMM transition call in response to the SMM transition call and to assign the authorization state based on an authentication result; and prevent the CPU in the SMM from accessing the data in response to the authentication result being a failure of
25 authentication

[0061] In Example 14, the subject matter of Example 13 optionally includes, wherein the first code includes an operating system (OS), and wherein the second code includes SMM firmware.

[0062] In Example 15, the subject matter of any one or more of Examples 13–
30 14 optionally include, wherein the SMM transition call causes generation of a system management interrupt in response to writing of the data in a data communication buffer accessible to the SMM control engine.

[0063] In Example 16, the subject matter of any one or more of Examples 13–15 optionally include, wherein the originator of the SMM transition call is a component of the first code.

5 [0064] In Example 17, the subject matter of any one or more of Examples 13–16 optionally include, wherein the originator of the SMM transition call is a driver engine incorporated into an operating system, wherein the operating system is a portion of the first code.

10 [0065] In Example 18, the subject matter of any one or more of Examples 13–17 optionally include, wherein the instructions are further to cause the computing hardware to access the data from the originator of the SMM transition call and, in response, to effect the transition of the CPU from the normal mode to the SMM.

15 [0066] In Example 19, the subject matter of any one or more of Examples 13–18 optionally include, wherein the instructions are further to cause the computing hardware to: access the data from the originator of the SMM transition call; access a digital signature associated with the originator of the SMM transition call; and append the digital signature to the data from the originator and to effect the transition of the CPU from the normal mode to the SMM.

20 [0067] In Example 20, the subject matter of any one or more of Examples 13–19 optionally include, wherein the instructions are further to cause the computing hardware to perform the authentication based on a validation of a digital certificate of the originator of the SMM transition call.

25 [0068] In Example 21, the subject matter of any one or more of Examples 13–20 optionally include, wherein the instructions are further to cause the computing hardware to perform the authentication based on a proof of identity of the originator of the SMM transition call.

30 [0069] In Example 22, the subject matter of any one or more of Examples 13–21 optionally include, wherein the instructions are further to cause the computing hardware to perform the authentication based on a comparison of an identification of the originator of the SMM transition call against a list of trusted identifications.

[0070] Example 23 is a method for securing system management mode (SMM) in a computer system that includes computing hardware (HW) including a

- central processing unit (CPU) that is operative in a normal mode and SMM, the method comprising: executing, by the CPU, first code in a normal mode, and second code in a SSM; transitioning the CPU from the normal mode to the SMM in response to a SMM transition call, and controlling access by the CPU in the
- 5 SMM to data from an originator of the SMM transition call based on an authorization state assigned to the SMM transition call; performing authentication of the originator of the SMM transition call in response to the SMM transition call and assigning the authorization state based on an authentication result; and preventing the CPU in the SMM from accessing the
- 10 data in response to the authentication result being a failure of authentication
- [0071]** In Example 24, the subject matter of Example 23 optionally includes, wherein the first code includes an operating system (OS), and wherein the second code includes SMM firmware.
- [0072]** In Example 25, the subject matter of any one or more of Examples 23–
- 15 24 optionally include, wherein the SMM transition call causes generation of a system management interrupt in response to writing of the data in a data communication buffer accessible to the SMM control engine.
- [0073]** In Example 26, the subject matter of any one or more of Examples 23–
- 20 25 optionally include, wherein the originator of the SMM transition call is a component of the first code.
- [0074]** In Example 27, the subject matter of any one or more of Examples 23–
- 26 optionally include, wherein the originator of the SMM transition call is a driver engine incorporated into an operating system, wherein the operating system is a portion of the first code.
- 25 **[0075]** In Example 28, the subject matter of any one or more of Examples 23–
- 27 optionally include, further comprising accessing the data from the originator of the SMM transition call and, in response, effecting the transition of the CPU from the normal mode to the SMM.
- [0076]** In Example 29, the subject matter of any one or more of Examples 23–
- 30 28 optionally include, further comprising: accessing the data from the originator of the SMM transition call; accessing a digital signature associated with the originator of the SMM transition call; and appending the digital signature to the data from the originator and to effect the transition of the CPU from the normal mode to the SMM.

[0077] In Example 30, the subject matter of any one or more of Examples 23–29 optionally include, wherein the authentication is based on a validation of a digital certificate of the originator of the SMM transition call.

5 [0078] In Example 31, the subject matter of any one or more of Examples 23–30 optionally include, wherein the authentication is based on a proof of identity of the originator of the SMM transition call.

[0079] In Example 32, the subject matter of any one or more of Examples 23–31 optionally include, wherein the authentication is based on a comparison of an identification of the originator of the SMM transition call against a list of trusted
10 identifications.

[0080] Example 33 is a system for securing system management mode (SMM) in a computer system that includes computing hardware (HW) including a central processing unit (CPU) that is operative in a normal mode and SMM, the system comprising: means for executing, by the CPU, first code in a normal
15 mode, and second code in a SSM; means for transitioning the CPU from the normal mode to the SMM in response to a SMM transition call, and for controlling access by the CPU in the SMM to data from an originator of the SMM transition call based on an authorization state assigned to the SMM transition call; means performing authentication of the originator of the SMM
20 transition call in response to the SMM transition call and for assigning the authorization state based on an authentication result; and means for preventing the CPU in the SMM from accessing the data in response to the authentication result being a failure of authentication

[0081] In Example 34, the subject matter of Example 33 optionally includes,
25 wherein the first code includes an operating system (OS), and wherein the second code includes SMM firmware.

[0082] In Example 35, the subject matter of any one or more of Examples 33–34 optionally include, wherein the SMM transition call causes generation of a system management interrupt in response to writing of the data in a data
30 communication buffer accessible to the SMM control engine.

[0083] In Example 36, the subject matter of any one or more of Examples 33–35 optionally include, wherein the originator of the SMM transition call is a component of the first code.

[0084] In Example 37, the subject matter of any one or more of Examples 33–36 optionally include, wherein the originator of the SMM transition call is a driver engine incorporated into an operating system, wherein the operating system is a portion of the first code.

5 [0085] In Example 38, the subject matter of any one or more of Examples 33–37 optionally include, further comprising: means for accessing the data from the originator of the SMM transition call and, in response, for effecting the transition of the CPU from the normal mode to the SMM.

[0086] In Example 39, the subject matter of any one or more of Examples 33–10 38 optionally include, further comprising: means for accessing the data from the originator of the SMM transition call; means for accessing a digital signature associated with the originator of the SMM transition call; and means for appending the digital signature to the data from the originator and to effect the transition of the CPU from the normal mode to the SMM.

15 [0087] In Example 40, the subject matter of any one or more of Examples 33–39 optionally include, wherein the authentication is based on a validation of a digital certificate of the originator of the SMM transition call.

[0088] In Example 41, the subject matter of any one or more of Examples 33–20 40 optionally include, wherein the authentication is based on a proof of identity of the originator of the SMM transition call.

[0089] In Example 42, the subject matter of any one or more of Examples 33–41 optionally include, wherein the authentication is based on a comparison of an identification of the originator of the SMM transition call against a list of trusted identifications.

25 [0090] In Example 43, the subject matter of any one or more of Examples 33–42 optionally include, wherein the means performing authentication is implementable on a caretaker processor of the HW that is isolated from the CPU and is to execute dedicated firmware code.

[0091] Example 44 is at least one computer-readable medium comprising 30 instructions that, when executed on computing hardware (HW) including a central processing unit (CPU) that is operative in a normal mode and a system management mode (SMM), cause the computing hardware to execute the subject matter according to any one of Examples 23-32.

[0092] In Example 45 is a system for securing system management mode (SMM) in a computer system that includes computing hardware (HW) including a central processing unit (CPU) that is operative in a normal mode and SMM, the system comprising means for executing the subject matter according to any one of Examples 23-32.

[0093] The above detailed description includes references to the accompanying drawings, which form a part of the detailed description. The drawings show, by way of illustration, specific embodiments that may be practiced. These embodiments are also referred to herein as “examples.” Such examples may include elements in addition to those shown or described. However, also contemplated are examples that include the elements shown or described. Moreover, also contemplated are examples using any combination or permutation of those elements shown or described (or one or more aspects thereof), either with respect to a particular example (or one or more aspects thereof), or with respect to other examples (or one or more aspects thereof) shown or described herein.

[0094] Publications, patents, and patent documents referred to in this document are incorporated by reference herein in their entirety, as though individually incorporated by reference. In the event of inconsistent usages between this document and those documents so incorporated by reference, the usage in the incorporated reference(s) are supplementary to that of this document; for irreconcilable inconsistencies, the usage in this document controls.

[0095] In this document, the terms “a” or “an” are used, as is common in patent documents, to include one or more than one, independent of any other instances or usages of “at least one” or “one or more.” In this document, the term “or” is used to refer to a nonexclusive or, such that “A or B” includes “A but not B,” “B but not A,” and “A and B,” unless otherwise indicated. In the appended claims, the terms “including” and “in which” are used as the plain-English equivalents of the respective terms “comprising” and “wherein.” Also, in the following claims, the terms “including” and “comprising” are open-ended, that is, a system, device, article, or process that includes elements in addition to those listed after such a term in a claim are still deemed to fall within the scope of that claim. Moreover, in the following claims, the terms “first,” “second,” and

“third,” etc. are used merely as labels, and are not intended to suggest a numerical order for their objects.

[0096] The above description is intended to be illustrative, and not restrictive.

For example, the above-described examples (or one or more aspects thereof)

- 5 may be used in combination with others. Other embodiments may be used, such as by one of ordinary skill in the art upon reviewing the above description. The Abstract is to allow the reader to quickly ascertain the nature of the technical disclosure. It is submitted with the understanding that it will not be used to interpret or limit the scope or meaning of the claims. Also, in the above Detailed
- 10 Description, various features may be grouped together to streamline the disclosure. However, the claims may not set forth every feature disclosed herein as embodiments may feature a subset of said features. Further, embodiments may include fewer features than those disclosed in a particular example. Thus, the following claims are hereby incorporated into the Detailed Description, with
- 15 a claim standing on its own as a separate embodiment. The scope of the embodiments disclosed herein is to be determined with reference to the appended claims, along with the full scope of equivalents to which such claims are entitled.

20

25

30

CLAIMS

What is claimed is:

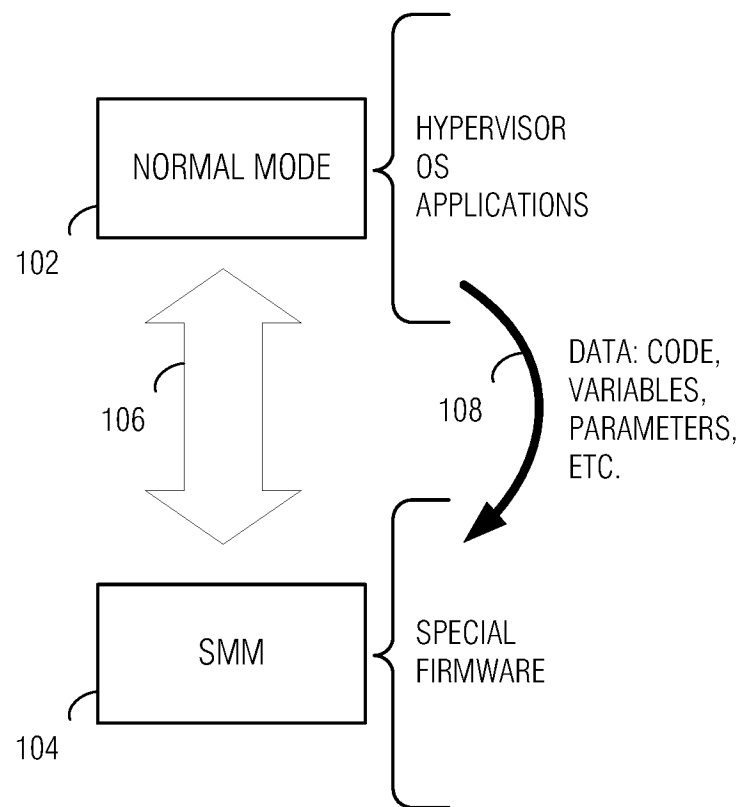
1. A system for securing system management mode (SMM) in a computer
5 system that includes computing hardware (HW), the system comprising:
a central processing unit (CPU) of the HW that, when operational, is
configurable to execute first code in a normal mode, and second code in a SMM;
a SMM control engine implementable at least in part on the HW to transition
the CPU from the normal mode to the SMM in response to a SMM transition call, and
10 to control access by the CPU in the SMM to data from an originator of the SMM
transition call based on an authorization state assigned to the SMM transition call; and
an authorization engine implementable at least in part on the HW to perform
authentication of the originator of the SMM transition call in response to the SMM
transition call and to assign the authorization state based on an authentication result;
15 wherein the SMM control engine is to prevent the CPU in the SMM from
accessing the data in response to the authentication result being a failure of
authentication.
2. The system of claim 1, wherein the first code includes an operating system
20 (OS), and wherein the second code includes SMM firmware.
3. The system of claim 1, wherein the SMM transition call causes generation of a
system management interrupt in response to writing of the data in a data communication
buffer accessible to the SMM control engine.
25
4. The system of claim 1, wherein the SMM control engine is to invoke the
authorization engine to perform the authentication in response to the SMM transition
call.
- 30 5. The system of claim 1, wherein the originator of the SMM transition call is a
component of the first code.
6. The system of claim 1, wherein the originator of the SMM transition call is a
driver engine incorporated into an operating system, wherein the operating system is a
35 portion of the first code.
7. The system of claim 1, further comprising:
a SMM calling engine implementable at least in part on the HW to access the
data from the originator of the SMM transition call and, in response, to effect the
40 transition of the CPU from the normal mode to the SMM.

8. The system of claim 1, further comprising:
a SMM calling engine implementable at least in part on the HW to access:
the data from the originator of the SMM transition call; and
5 a digital signature associated with the originator of the SMM transition call;
and
wherein the SMM calling engine is further to append the digital signature to
the data from the originator and to effect the transition of the CPU from the normal
mode to the SMM.
10
9. The system of claim 1, wherein the authorization engine is to perform the
authentication based on a validation of a digital certificate of the originator of the SMM
transition call.
- 15 10. The system of claim 1, wherein the authorization engine is to perform the
authentication based on a proof of identity of the originator of the SMM transition call.
11. The system of claim 1, wherein the authorization engine is to perform the
authentication based on a comparison of an identification of the originator of the SMM
20 transition call against a list of trusted identifications.
12. The system of claim 1, wherein the authorization engine is implementable on
a caretaker processor of the HW that is isolated from the CPU and is to execute
dedicated firmware code.
25
13. A method for securing system management mode (SMM) in a computer
system that includes computing hardware (HW) including a central processing unit
(CPU) that is operative in a normal mode and SMM, the method comprising:
executing, by the CPU, first code in a normal mode, and second code in a
30 SSM;
transitioning the CPU from the normal mode to the SMM in response to a
SMM transition call, and controlling access by the CPU in the SMM to data from an
originator of the SMM transition call based on an authorization state assigned to the
SMM transition call;
35 performing authentication of the originator of the SMM transition call in
response to the SMM transition call and assigning the authorization state based on an
authentication result; and
preventing the CPU in the SMM from accessing the data in response to the
authentication result being a failure of authentication.
40

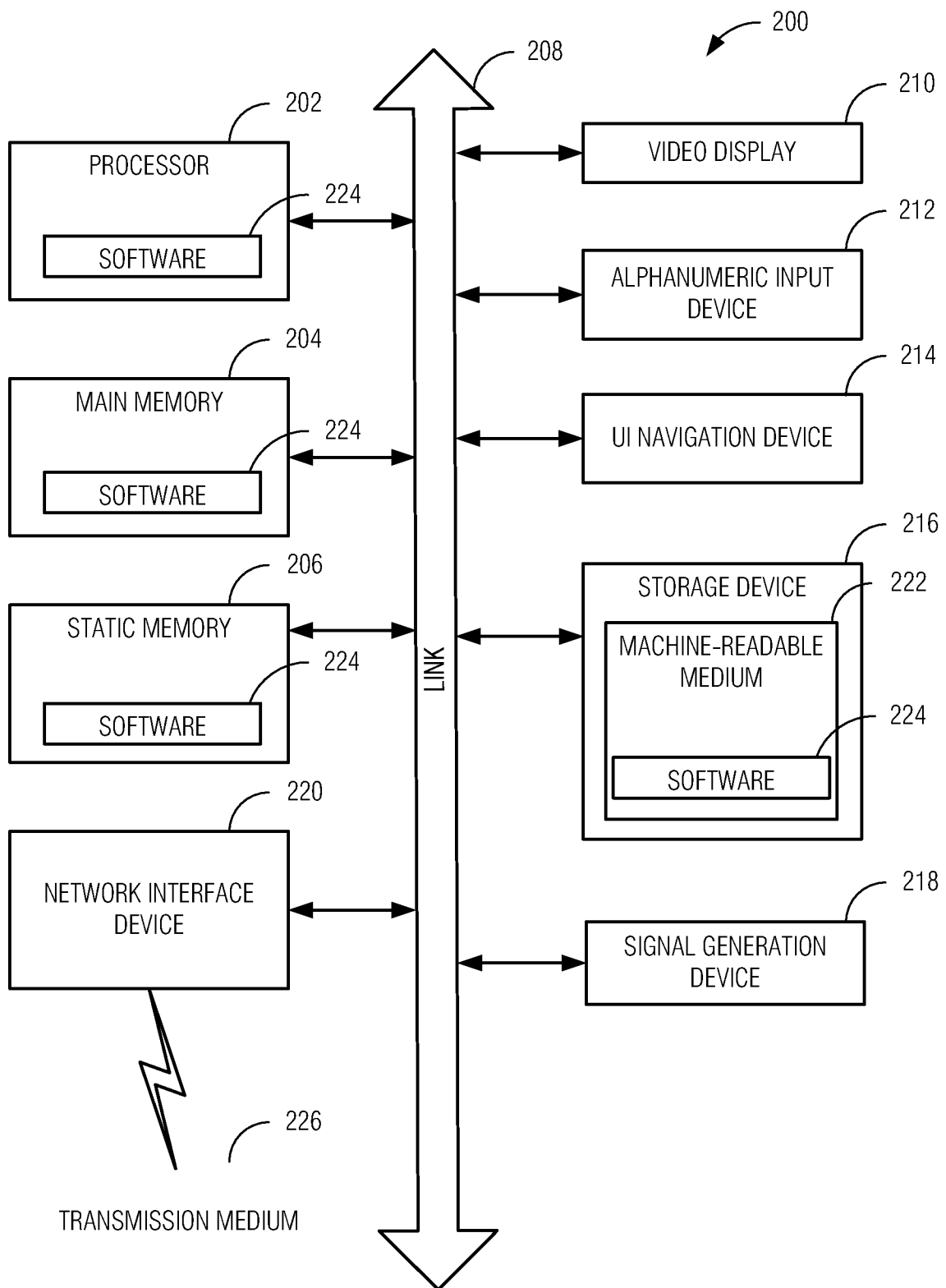
14. The method of claim 13, wherein the first code includes an operating system (OS), and wherein the second code includes SMM firmware.
15. The method of claim 13, wherein the SMM transition call causes generation of a system management interrupt in response to writing of the data in a data communication buffer accessible to the SMM control engine.
16. The method of claim 13, wherein the originator of the SMM transition call is a component of the first code.
17. The method of claim 13, wherein the originator of the SMM transition call is a driver engine incorporated into an operating system, wherein the operating system is a portion of the first code.
18. The method of claim 13, further comprising accessing the data from the originator of the SMM transition call and, in response, effecting the transition of the CPU from the normal mode to the SMM.
19. The method of claim 13, further comprising:
accessing the data from the originator of the SMM transition call;
accessing a digital signature associated with the originator of the SMM transition call; and
appending the digital signature to the data from the originator and to effect the transition of the CPU from the normal mode to the SMM.
20. The method of claim 13, wherein the authentication is based on a validation of a digital certificate of the originator of the SMM transition call.
21. The method of claim 13, wherein the authentication is based on a proof of identity of the originator of the SMM transition call.
22. The method of claim 13, wherein the authentication is based on a comparison of an identification of the originator of the SMM transition call against a list of trusted identifications.
23. At least one computer-readable medium comprising instructions that, when executed on computing hardware (HW) including a central processing unit (CPU) that is operative in a normal mode and a system management mode (SMM), cause the computing hardware to execute the method according to any one of claims 13-22.

24. A system for securing system management mode (SMM) in a computer system that includes computing hardware (HW) including a central processing unit (CPU) that is operative in a normal mode and SMM, the system comprising means for executing the method according to any one of claims 13-22.

1/6

**FIG. 1**

2/6

**FIG. 2**

3/6

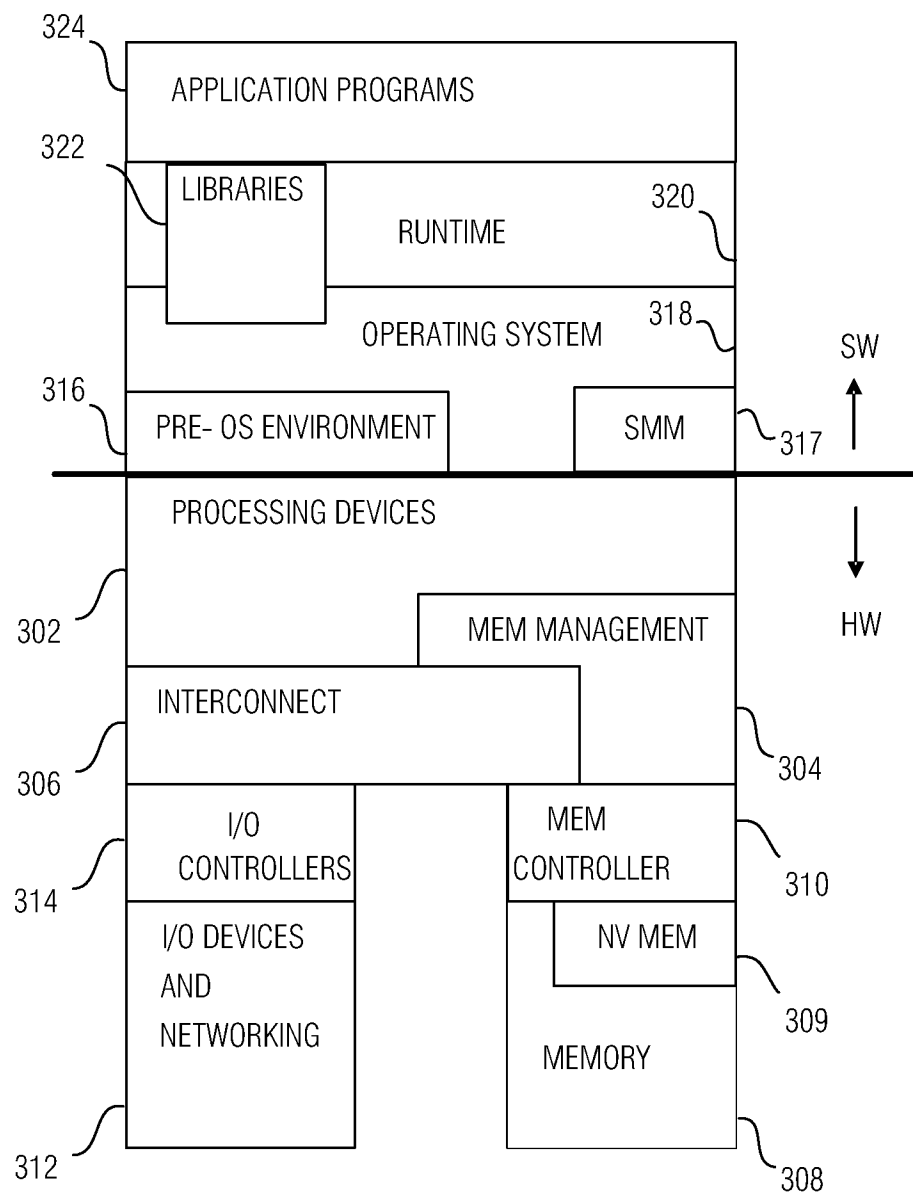


FIG. 3

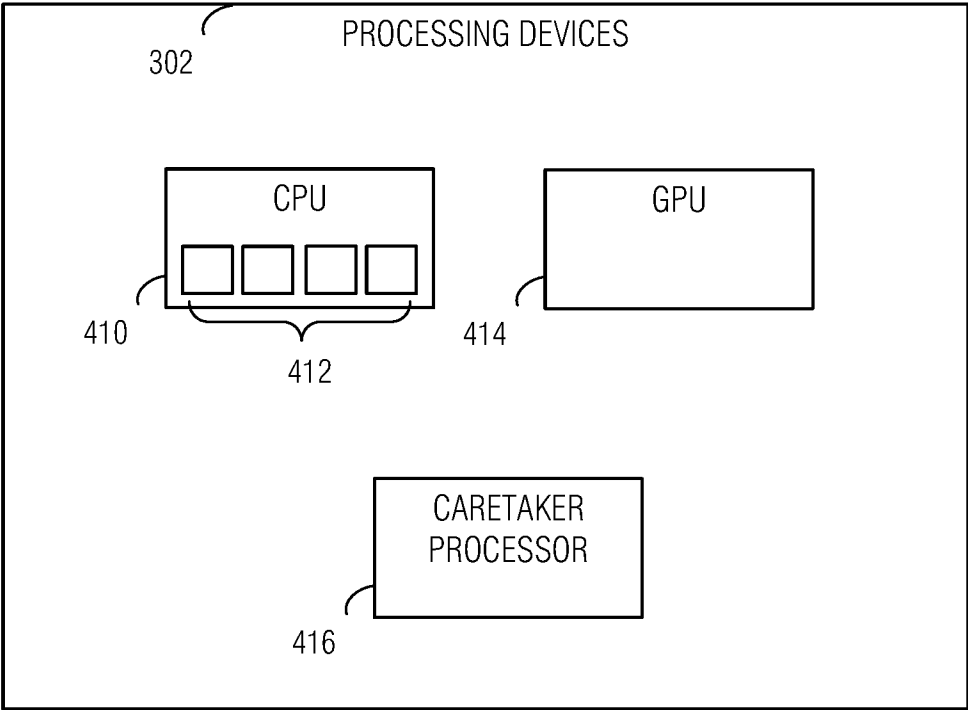
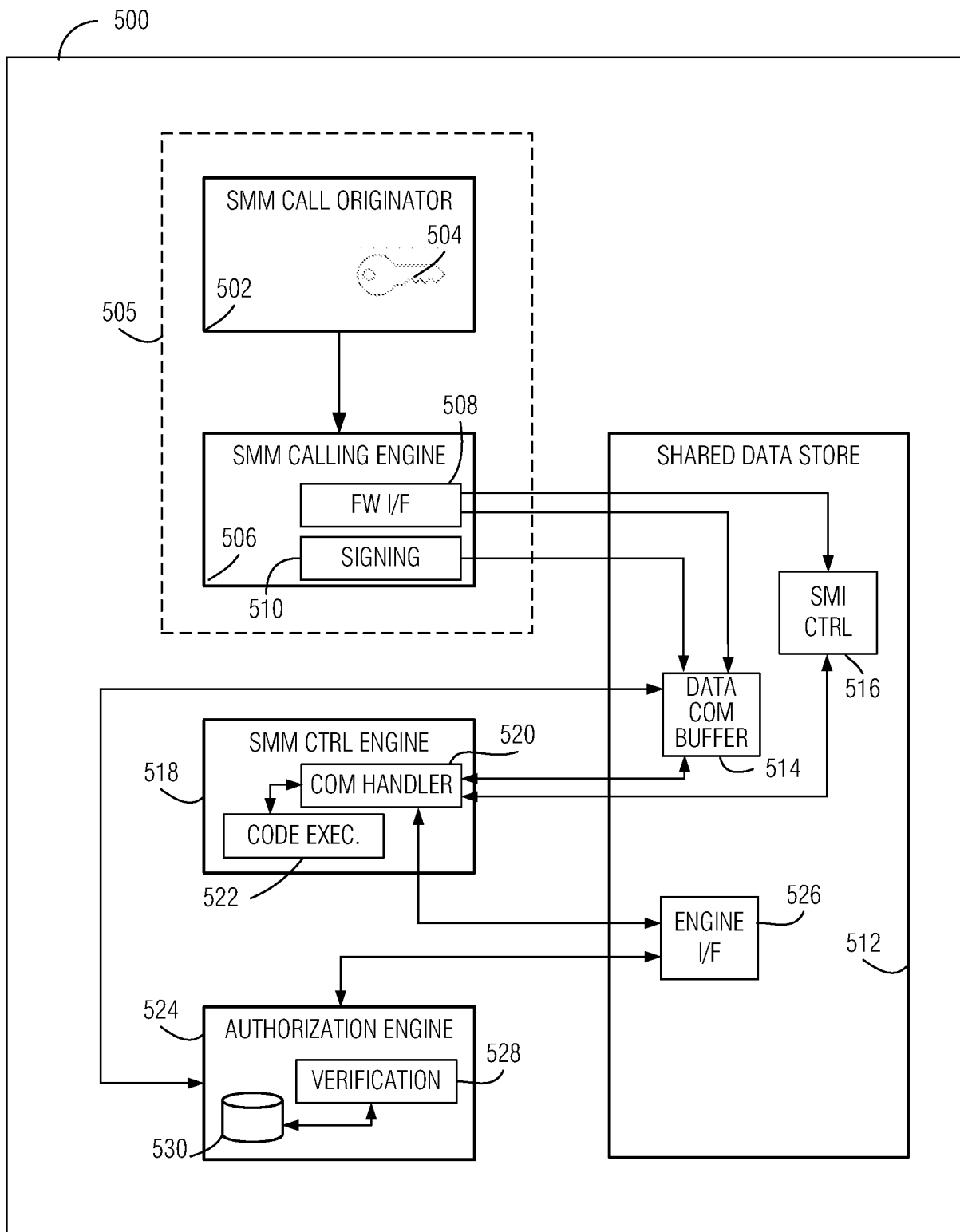


FIG. 4

5/6

**FIG. 5**

6/6

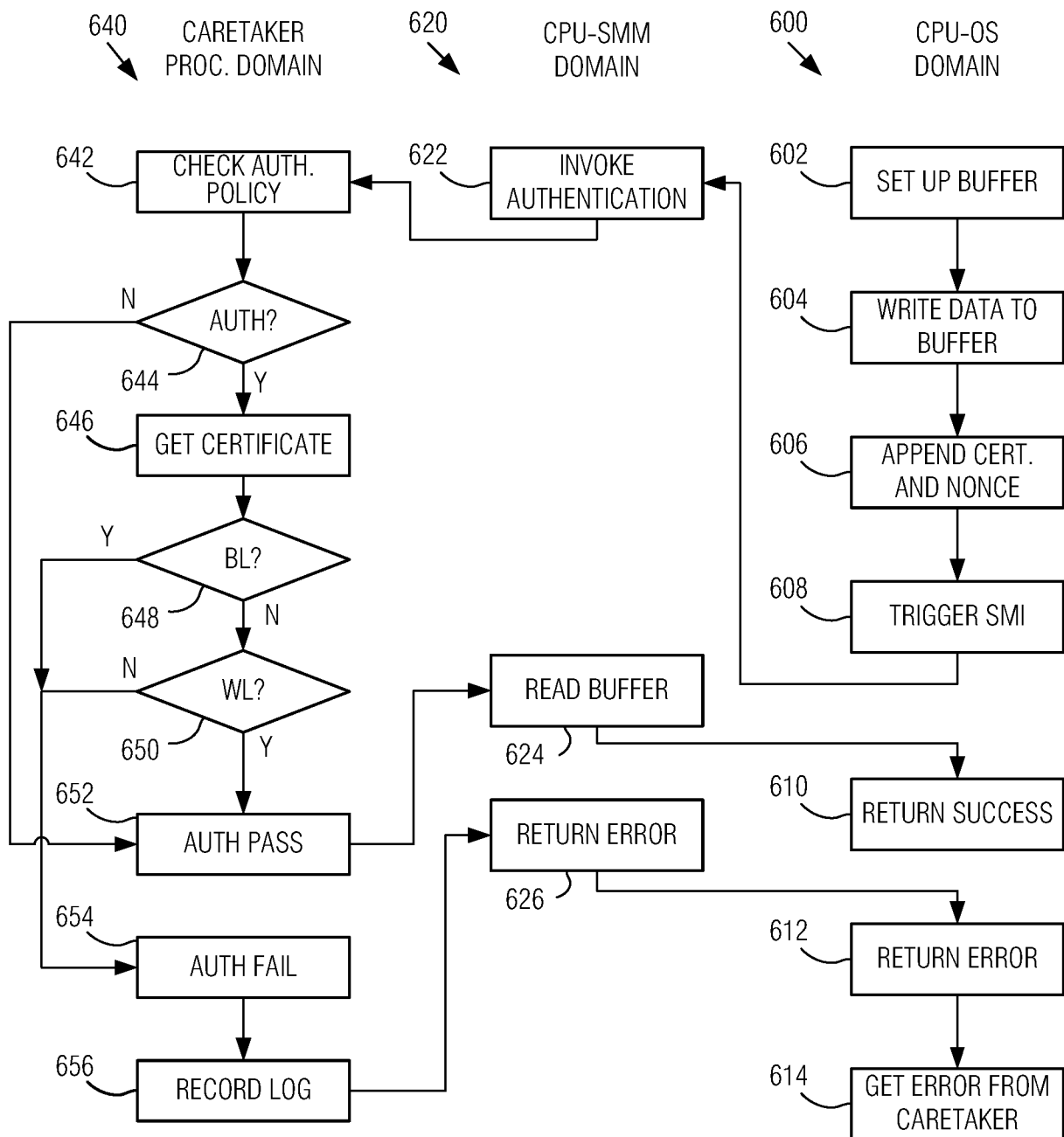


FIG. 6

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2016/070868

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/14(2006.01)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:CPU, normal 3d mode, security 3d mode, access+, authentication, response, code?, call+

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2009204823 A1 (ANALOG DEVICES, INC.) 13 August 2009 (2009-08-13) description, paragraphs[0004]-[0009], figures 1A-5	1-24
A	CN 103729219 A (HEFEI LCFC INFORMATION TECHNOLOGY CO., LTD.) 16 April 2014 (2014-04-16) the whole document	1-24
A	CN 101950262 A (SHENZHEN SONOSCAPE TECHNOLOGY CO., LTD.) 19 January 2011 (2011-01-19) the whole document	1-24

☐ 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	“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
“E”	earlier application or patent but published on or after the international filing date	“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
“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)	“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
“O”	document referring to an oral disclosure, use, exhibition or other means	“&”	document member of the same patent family
“P”	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search

05 February 2016

Date of mailing of the international search report

29 February 2016

Name and mailing address of the ISA/CN

STATE INTELLECTUAL PROPERTY OFFICE OF THE
P.R.CHINA
6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing
100088, China

Authorized officer

WANG,Xiaofei

Facsimile No. (86-10)62019451

Telephone No. (86-10)62413918

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2016/070868

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
US	2009204823	A1	13 August 2009	WO	2009099647	A1	13 August 2009
				CN	101978377	A	16 February 2011
				JP	2011511383	A	07 April 2011
				EP	2248063	A1	10 November 2010
CN	103729219	A	16 April 2014	None			
CN	101950262	A	19 January 2011	None			