



(51) International Patent Classification:
G06F 11/08 (2006.01)

(21) International Application Number:
PCT/US2019/017459

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

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 10300 Energy Drive, Spring, Texas 77389 (US).

(72) Inventors: **JEANSONNE, Jeffrey Kevin**; 10300 Energy Dr., Spring, Texas 77389 (US). **ALI, Valiuddin**; 10300 Energy Dr., Spring, Texas 77389 (US). **BRAMLEY, Richard**; 12 Gloria Dr, Mansfield, Massachusetts 02048 (US).

(74) Agent: **SU, Benjamin** et al.; HP Inc., Mail Stop 35 3390 E. Harmony Road, Fort Collins, Colorado 80528 (US).

(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).

(54) Title: RECOVERY FROM CORRUPTION

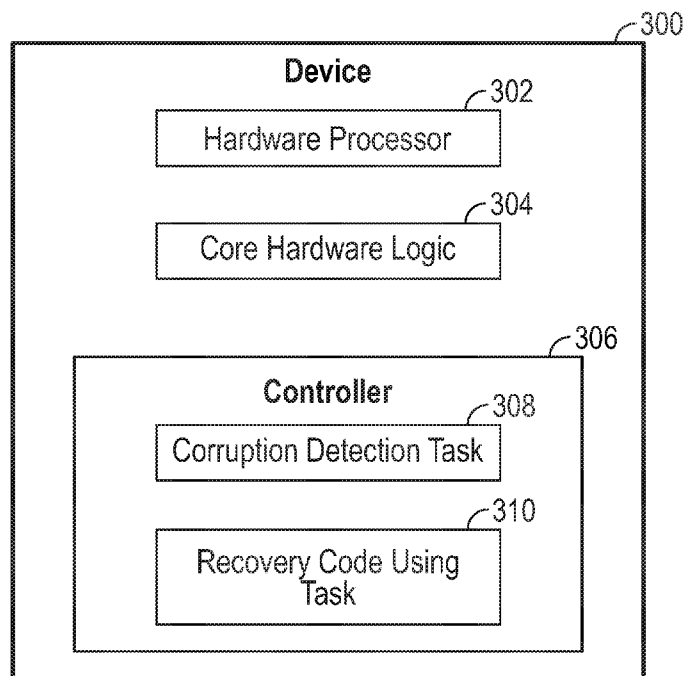


FIG. 3

(57) **Abstract:** In some examples, a device includes a processor, a core hardware logic to execute instructions to perform a task in the device, and a controller separate from the processor. The controller detects corruption of the instructions, and in response to detecting the corruption, load a recovery code to the core hardware logic to trigger recovery of the core hardware logic from the corruption of the instructions.

Declarations under Rule 4.17:

- *as to the identity of the inventor (Rule 4.17(i))*
- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

- *with international search report (Art. 21(3))*

RECOVERY FROM CORRUPTION

Background

[0001] Electronic devices can include various components for performing different tasks. For example, the components can include a processor, a memory, an embedded controller, an input/output (I/O) device, and other components. Various code (in the form of machine-readable instructions including firmware and/or software) are executable on the embedded controller, the processor, and other components.

Brief Description of the Drawings

[0002] Some implementations of the present disclosure are described with respect to the following figures.

[0003] Fig. 1 is a block diagram of an electronic device according to some examples.

[0004] Fig. 2 is a flow diagram of a process according to some examples.

[0005] Fig. 3 is a block diagram of a device according to some examples.

[0006] Fig. 4 is a block diagram of a storage medium storing machine-readable instructions according to some examples.

[0007] Fig. 5 is a block diagram of a process according to further examples.

[0008] Throughout the drawings, identical reference numbers designate similar, but not necessarily identical, elements. The figures are not necessarily to scale, and the size of some parts may be exaggerated to more clearly illustrate the example shown. Moreover, the drawings provide examples and/or implementations consistent with the description; however, the description is not limited to the examples and/or implementations provided in the drawings.

Detailed Description

[0009] In the present disclosure, use of the term “a,” “an,” or “the” is intended to include the plural forms as well, unless the context clearly indicates otherwise. Also, the term “includes,” “including,” “comprises,” “comprising,” “have,” or “having” when used in this disclosure specifies the presence of the stated elements, but do not preclude the presence or addition of other elements.

[0010] Machine-readable instructions executable by various components of an electronic device may be corrupted for a number of reasons. For example, an unauthorized entity, such as malware, may modify the instructions or data used by the instructions to cause the instructions to no longer be operational or to operate in an unintended manner. Corruption of instructions can also be due to other causes, including failure to update the instructions with a latest code update, errors introduced by defective hardware or a program, and so forth.

[0011] Examples of electronic devices include any or some combination of the following: computers (e.g., desktop computers, notebook computers, tablet computers, server computers, etc.), handheld devices (e.g., smartphones, game appliances, etc.), wearable devices (e.g., smart watches, head-mounted devices, smart eyeglasses, etc.), Internet-of-Things (IoT) devices, controllers in vehicles, storage systems, communication nodes, and so forth.

[0012] If the machine-readable instructions (e.g., firmware) of certain components of an electronic device become corrupted, then the electronic device may not be able to boot successfully and thus can become non-operational. For example, an electronic device may include core hardware logic that performs certain functionalities for the electronic device. For example, the core hardware logic can include any or some combination of the following: an input/output (I/O) control logic to manage interaction with I/O components of the electronic device, a display control logic to control display of images at a display device, a memory control logic to manage access of a memory, a storage control logic to manage access of a persistent storage such as a disk-based storage or solid state drive, and so forth.

[0013] The core hardware logic of the electronic device can also be referred to as a “chipset” of the electronic device, where the chipset can include one integrated circuit (IC) chip or multiple IC chips, depending on the implementation.

[0014] Certain components of the core hardware logic can execute machine-readable instructions, such as firmware and/or software. One such component includes a management engine of the core hardware logic. Examples of management engines include the Management Engine (ME) from Intel, the Secure Processor from Advanced Micro Devices (AMD), and so forth. More generally, the core hardware logic can include a core controller (e.g., the ME or Secure Processor) that performs certain basic tasks in an electronic device to enable operation of the electronic device.

[0015] An example of a task that can be performed by firmware executed on the core controller include configuration of settings of a processor, such as configuring a clock of the processor, configuring a voltage of the processor, configuring power management control by the processor, and so forth. In other examples, the firmware executed on the core controller can also provide security for the electronic device (such as by storing an encryption key in a storage), enable remote management of the electronic device (in which an entity that is coupled to the electronic device over a network can manage the electronic device), and so forth.

[0016] If the firmware of the core controller becomes corrupted for any reason, then the core controller would not be able to perform its tasks during startup of an electronic device. As a result, the processor of the electronic device would not be configured properly (because the core controller has experienced a fault), and thus the processor may remain non-operational. As a result, boot code of the electronic device cannot start, and the electronic device will not operate.

[0017] In accordance with some implementations of the present disclosure, an electronic device includes a controller (e.g., an embedded controller), which is separate from the processor of the electronic device. The embedded controller is able to detect corruption of machine-readable instructions (e.g., firmware) executable

by a core hardware logic (such as by a core controller). In response to detecting the corruption of the machine-readable instructions of the core hardware logic, the embedded controller can retrieve a recovery code and load the recovery code to the core hardware logic. The core hardware logic can execute the recovery code to allow the core hardware logic to perform certain basic tasks, including configuring the processor to enable functioning of the processor. Once the functioning of the processor is enabled, a boot code of the electronic device can execute on the processor, and the boot code can perform a full recovery process to recover from the corruption of the machine-readable instructions of the core hardware logic, including retrieving a full recovery image for the core hardware logic.

[0018] Fig. 1 is a block diagram of an example electronic device 100 that includes an embedded controller 102, a chipset 104, and a processor 106. The chipset 104 is considered the core hardware logic of the electronic device 100, in some examples. For example, the chipset 104 can include a Platform Controller Hub (PCH) from Intel, a Fusion Controller Hub from AMD, and so forth.

[0019] The chipset 104 includes a core controller 108. The core controller 108 can perform various tasks as noted above. One of the tasks that can be performed by the core controller 108 is configuring settings of the processor 106, to enable the processor 106 to operate during startup of the electronic device 100.

[0020] In some examples, the embedded controller 102 can be used to perform specific predefined tasks. In some examples, the tasks of the embedded controller 102 are performed by embedded controller (EC) code (that is part of EC information) 118, in the form of machine-readable instructions such as EC firmware or EC software, executed on the embedded controller 102. In other examples, the tasks of the embedded controller 102 can be performed by a hardware processing circuit of the embedded controller 102. Examples of tasks that can be performed by the embedded controller 102 include any one or some combination of the following: power supply control in the electronic device 100 (for controlling a power supply that supplies power supply voltages to various components in the electronic device 100), charging and control of a battery in the electronic device 100, thermal monitoring (to

monitor a temperature in the electronic device 100), fan control (to control a fan in the electronic device 100), and interaction with a user input device (such as performing a scan of a keyboard of the electronic device 100 or interaction with a pointing device such as a mouse, touchpad, touchscreen, and so forth). In other examples, the embedded controller 102 can perform additional or alternative tasks. The embedded controller 102 can be implemented with a microcontroller, an application-specific integrated circuit (ASIC), a programmable gate array (PGA), or any other type of programmable circuit.

[0021] The electronic device 100 further includes an embedded controller (EC) storage device 110, a shared storage device 112, and a mass storage device 114. In some examples, the EC storage device 110 is a private storage device for the embedded controller 102. In such examples, the data in the EC storage device 110 is accessible by the embedded controller 102, but not accessible by any other component of the electronic device 100, including the processor 106 and the chipset 104.

[0022] The shared storage device 112 is accessible by multiple components, including the embedded controller 102, the chipset 104, and the processor 106. Access of the shared storage device 112 can be performed over a shared bus 115.

[0023] The mass storage device 114 is a storage device that has a larger capacity than the shared storage device 112 or the EC storage device 110. Each of the EC storage device 110, the shared storage device 112, and the mass storage device 114 can be implemented using a nonvolatile memory. A nonvolatile memory can include one nonvolatile memory device, or multiple nonvolatile memory devices. Examples of nonvolatile memory devices include a flash memory device, a phase change memory device, a memristor memory device, and so forth. In further examples, a nonvolatile memory can also be implemented using a disk-based storage device. In further examples, the data in the mass storage device 114 may also be accessible by the processor 106 and the embedded controller 102. In such further examples, the embedded controller 102 is able to access the data in the mass storage device 114 even when the processor 106 is non-operational, such as

during the initial stages of startup of the electronic device 100, or due to a corruption in the electronic device 100.

[0024] The shared storage device 112 can be used to store various different information, including a core controller information 116, the EC information 118, and boot code 120. The core controller information 116 includes core controller code (in the form of machine-readable instructions) and core controller data that is useable by the core controller code. In some examples, the core controller code includes core controller firmware executable by the core controller 108. The EC information 118 includes EC code (in the form of machine-readable instructions) and EC data useable by the EC code. The EC code can include EC firmware executable by the embedded controller 102.

[0025] Although Fig. 1 shows the various storage devices 110, 112, and 114 storing respective different pieces of information, it is noted that in other examples, the respective pieces of information can be stored by different storage devices (e.g., just one storage device or more than one storage device).

[0026] During startup of the electronic device 100, the core controller 108 is able to retrieve the core controller code (part of the core controller information 116) from the shared storage device 112 over the shared bus 115, for execution on the core controller 108 to perform tasks of the core controller 108. Prior to the loading of the core controller code on the core controller 108, the embedded controller 102 can first verify the integrity of the core controller code and the core controller data.

[0027] The embedded controller 102 includes a core recovery logic 122 to perform verification of the core controller information 116. The core recovery logic 122 can be implemented using a portion of the hardware processing circuit of the embedded controller 102, or alternatively, can be implemented as machine-readable instructions executable by the embedded controller 102.

[0028] Verifying a piece of information can refer to cryptographically validating that the piece of information has not been changed and/or confirming that the piece

of information is from a trusted source. In some examples, a cryptographic-based verification technique can include a Rivest, Shamar, and Adleman (RSA) verification technique that employs cryptographic encryption. The RSA verification technique involves use of a public key and a private key. The public key is known to multiple entities, but the private key is only known to a single entity. Data encrypted with a private key can be decrypted using the corresponding public key. Alternatively, data can be encrypted using the private key, but decrypted using the public key.

[0029] For example, the core controller information 116 stored in the shared storage device 112 can be encrypted using a public key or a private key. To verify the integrity of the core controller information 116, the embedded controller 102 can decrypt the encrypted core controller information 116 using the corresponding private key or public key. If decryption is successful, then the core controller information 116 is verified.

[0030] In other examples, other cryptographic-based verification techniques can be employed, such as a Secure Hashing Algorithm (SHA), and so forth.

[0031] In further examples, to verify the source and integrity of a piece of information, a signature can be stored in association with the piece of information (either part of the piece of information or as metadata associated with the piece of information). In some examples, the verification of the core controller information 116 can be accomplished by decrypting the associated signature using an encryption key accessible by the embedded controller 102. Decrypting the signature produces a respective value (e.g., a hash value) that can be compared with a corresponding calculated value (e.g., hash value) of the core controller information 116. If the foregoing values match, then the integrity and source of the core controller information 116 is verified.

[0032] If the core recovery logic 122 is unable to verify the core controller information 116, then that is an indication that the core controller information 116 is corrupted. The core recovery logic 122 can, in response to detection of corruption of

the core controller information 116, initiate a recovery process for the core controller 108.

[0033] The EC storage device 110 stores a core recovery code 124. Note that the core recovery code 124 can be a copy of a core recovery code 125 that is a portion of the core controller code (in the core controller information 116) stored in the shared storage device 112. At some point during initial operation of the electronic device 100, which can be during initialization at the factory, initialization by a user, etc., the embedded controller 102 can copy the recovery code portion 125 of the core controller code to the EC storage device 110, for storage as the core recovery code 124.

[0034] In response to detection of corruption of the core controller information 116, the core recovery logic 122 prevents the core controller code in the shared storage device 112 from being loaded onto the core controller 108 for execution. For example, the core recovery logic 122 can disable the ability of the chipset 104 to access the shared storage device 112 in response to detection of corruption of the core controller information 116.

[0035] To trigger recovery from the corruption of the core controller information 116, the core recovery logic 122 retrieves the core recovery code 124 from the EC storage device 110, and replaces the core recovery code 125 in the shared storage device 112 with the core recovery code 124. The replaced core recovery code 125 (as replaced with the core recover code 124) is loaded by the core controller 108 when the core controller 108 is booted. The replaced core recovery code 125 when executed by the core controller 108 causes the core controller 108 to perform various basic tasks, including configuring the settings of the processor 106 so that the processor 106 can function.

[0036] Once the processor 106 is configured, the core controller 108 allows the remaining startup process of the electronic device 100 to proceed, including loading of the boot code 120 for execution on the processor 106.

[0037] In some examples, the replaced core recovery code 125 when executed by the core controller 108 can set a flag, such as a recovery flag 126 in the shared storage device 112, to indicate that a core controller recovery process is in progress. The core controller recovery process refers to a process of recovering from corruption of instructions executable by the core controller 108.

[0038] A “flag” can refer to any information element, in the form of a bit or a combination of bits, that is settable to different values. If the recovery flag 126 is set to a first value, then that indicates that the core controller recovery process is in progress. If the recovery flag is set to a second value different from the first value, that indicates that a normal startup process of the electronic device 100 is in progress.

[0039] When the boot code 120 executes on the processor 106, the boot code 120 determines the state of the recovery flag 126. If the recovery flag 126 is set to the first value, then the boot code 120 performs the core controller recovery process. In the core controller recovery process, the boot code 120 retrieves a full core controller code image 128 from the mass storage device 114. The full core controller code image 128 is a known uncorrupted version of core controller code.

[0040] The boot code 120 can replace the corrupted core controller information 116 in the shared storage device 112 with the full core controller code image 128. Once replaced, the full core controller code image 128 becomes the core controller information 116 of the electronic device 100, and the core controller code of the core controller information 116 can be executed by the core controller 108 to perform further tasks of the core controller 108.

[0041] Subsequently, the boot code 120 can complete the remaining boot process, which includes initializing various hardware components of the electronic device 100, and loading an operating system (OS) 130 for execution on the processor 106.

[0042] Fig. 2 is a flow diagram of a process that can be performed by the electronic device 100 according to some examples. The process of Fig. 2 can be performed during startup of the electronic device 100, or in response to some other trigger event of the electronic device 100. As part of the startup, the embedded controller 102 can load (at 202) EC code (of the EC code information 118) to execute on the embedded controller 102. The EC code that is executed by the embedded controller 102 can include the core recovery logic 122, in some examples. In some examples, verification of the EC code information 118 can be performed prior to execution of the EC code to confirm that the EC code information 118 has not been corrupted.

[0043] The core recovery logic 122 verifies (at 204) the integrity of the core controller information 116 stored in the shared storage device 112. The core recovery logic 122 determines (at 206) whether the core controller information 116 is corrupted based on whether or not the core controller information 116 is successfully verified. If the core controller information 116 is successfully verified, then the core controller information 116 is not corrupted. However, if the core controller information 116 is not successfully verified, then the core controller information 116 is corrupted.

[0044] If the core controller information 116 is not corrupted, then the core recovery logic 122 allows (at 208) the remainder of a startup process to continue, including loading of the core controller code onto the core controller 108 for execution by the core controller 108, followed by loading of the boot code 120 for execution on the processor 106 to perform boot tasks.

[0045] If the core recovery logic 122 determines (at 206) that the core controller information 116 is corrupted, then the core recovery logic 122 retrieves (at 210) the core recovery code 124 from the EC storage device 110, and replaces the core recovery code 125 (that is part of the core controller information 116) with the retrieved core recover code 124. The replaced core recovery code 125 is loaded (at 212) by the core recovery logic 122

[0046] The replaced core recovery code 125 executed by the core controller 108 configures (at 214) the processor 106 to enable functioning of the processor 106. After the processor is enabled, the boot code 120 is executed (at 216) by the processor 106.

[0047] The boot code 120 executed on the processor 106 determines (at 218) the value of the recovery flag 126, and if the recovery flag 126 is set to the first value that indicates that the core controller recovery process is in progress, the boot code 120 retrieves (at 220) the full core controller code image 128 from the mass storage device 114. The boot code 120 replaces (at 222) the corrupted core controller information 116 in the shared storage device 112 with the full core controller code image 128. The recovered core controller code is then loaded (at 224) for execution by the full core controller 108, which performs the remaining tasks of the core controller 108, followed by performing (at 226) the boot process by the boot code 120.

[0048] If the boot code 120 determines (at 218) that the value of the recovery flag 126 is set to the second value that indicates a normal startup process, the boot process is performed (at 226) by the boot code 120.

[0049] Fig. 3 is a block diagram of a device 300 including a hardware processor 302 and a core hardware logic 304 to execute instructions to perform a task in the device 300. A hardware processor can include a microprocessor, a core of a multi-core microprocessor, a microcontroller, a programmable integrated circuit, a programmable gate array, a digital signal processor, or another hardware processing circuit.

[0050] A controller 306 (e.g., an embedded controller) separate from the hardware processor 302 is able to perform various tasks. The tasks include an instructions corruption detection task 308 to detect corruption of the instructions, and a recovery code using task 310 to, in response to detecting the corruption, use a recovery code to trigger recovery of the core hardware logic from the corruption of the instructions.

[0051] The recovery code is executable on the core hardware logic 304 to enable functioning of the hardware processor 302, by configuring the hardware processor 302. As discussed above, the recovery code can be retrieved from an EC storage device (e.g., 110 in Fig. 1), to replace a corresponding recovery code in a shared storage device (e.g., 112 in Fig. 1). The replaced recovery code is executed by the core hardware logic to configure the hardware processor 302 to enable functioning of the hardware processor 302.

[0052] Fig. 4 is a block diagram of a non-transitory machine-readable or computer-readable storage medium 400 storing machine-readable instructions that upon execution cause a controller of a device to perform various tasks. The machine-readable instructions include corruption detection instructions 402 to detect corruption of information comprising data and instructions executable by a core hardware logic in the device. Note that the corruption detected in the information can refer to corruption of the data, or corruption of the instructions, or corruption of both the data and instructions.

[0053] The machine-readable instructions further include recovery code loading instructions 404 to, in response to detecting the corruption of the information, retrieve a recovery code from a controller storage accessible by the controller, and replace a recovery code in the information. The machine-readable instructions further include recovery code execution instructions 406 to execute the replaced recovery code by the core hardware logic to configure a processor separate from controller. The machine-readable instructions additionally include boot code execution instructions 408 to, after the configuring of the processor, execute a boot code on the processor.

[0054] Fig. 5 is a flow diagram of a process according to some examples. The process of Fig. 5 includes detecting (at 502), by an embedded controller in a device, corruption of information including data and instructions executable by a core hardware logic in the device. In response to detecting the corruption of the information, the process of Fig. 5 includes uses (at 504) a recovery code in a controller storage to trigger a recovery process by the core hardware logic. Using the recovery code in the controller storage (e.g., the EC storage device 110 in Fig. 1)

to trigger the recovery process can include replacing recovery code in a shared storage device (e.g., 112 in Fig. 1) with the recovery code retrieved from the controller storage, and then executing the replaced recovery code by the core hardware logic to perform the recovery process.

[0055] The process of Fig. 5 further includes enabling (at 506), by the core hardware logic in the recovery process, functioning of a processor that is separate from the controller.

[0056] After enabling of the functioning of the processor by the core hardware logic, the process of Fig. 5 includes executing (at 508) a boot code on the processor to retrieve a full code image for the core hardware logic, the full code image useable to recover the core hardware logic from the corruption of the information.

[0057] The storage medium 400 of Fig. 4 can include any or some combination of the following: a semiconductor memory device such as a dynamic or static random access memory (a DRAM or SRAM), an erasable and programmable read-only memory (EPROM), an electrically erasable and programmable read-only memory (EEPROM) and flash memory; a magnetic disk such as a fixed, floppy and removable disk; another magnetic medium including tape; an optical medium such as a compact disc (CD) or a digital video disc (DVD); or another type of storage device. Note that the instructions discussed above can be provided on one computer-readable or machine-readable storage medium, or alternatively, can be provided on multiple computer-readable or machine-readable storage media distributed in a large system having possibly plural nodes. Such computer-readable or machine-readable storage medium or media is (are) considered to be part of an article (or article of manufacture). An article or article of manufacture can refer to any manufactured single component or multiple components. The storage medium or media can be located either in the machine running the machine-readable instructions, or located at a remote site from which machine-readable instructions can be downloaded over a network for execution.

[0058] In the foregoing description, numerous details are set forth to provide an understanding of the subject disclosed herein. However, implementations may be practiced without some of these details. Other implementations may include modifications and variations from the details discussed above. It is intended that the appended claims cover such modifications and variations.

What is claimed is:

1. A device comprising:
a processor;
a core hardware logic to execute instructions to perform a task in the device;
and
a controller separate from the processor to:
detect corruption of the instructions, and
in response to detecting the corruption, use a recovery code to trigger recovery of the core hardware logic from the corruption of the instructions.
2. The device of claim 1, further comprising:
a controller storage accessible by the controller when the processor is non-operational, the controller storage to store the recovery code.
3. The device of claim 2, wherein the recovery code is executable on the core hardware logic to enable functioning of the processor.
4. The device of claim 3, wherein the recovery code is executable on the core hardware logic to enable functioning of the processor by configuring the processor.
5. The device of claim 3, further comprising:
a boot code; and
a mass storage to store a full code image for the core hardware logic,
wherein the boot code is executable on the processor after the functioning of the processor is enabled to retrieve the full code image to perform the recovery.

6. The device of claim 5, further comprising:
a shared nonvolatile memory accessible by the processor and the controller,
wherein the shared nonvolatile memory is to store the instructions executable by the
core hardware logic.
7. The device of claim 6, wherein the boot code is to replace the instructions in
the shared nonvolatile memory using the full code image.
8. The device of claim 1, wherein the instructions comprise firmware executable
on the core hardware logic.
9. The device of claim 1, wherein the core hardware logic comprises another
controller to perform configuring of a hardware component of the device.
10. A non-transitory machine-readable storage medium comprising instructions
that upon execution cause a controller of a device to:
detect corruption of information comprising data and instructions executable
by a core hardware logic in the device; and
in response to detecting the corruption of the information, retrieve a recovery
code from a controller storage accessible by the controller, and replace a recovery
code in the information,
execute the replaced recovery code by the core hardware logic to configure a
processor separate from controller, and
after the configuring of the processor, execute a boot code on the processor.
11. The non-transitory machine-readable storage medium of claim 10, wherein
the controller is an embedded controller, and the controller storage is accessible by
the controller when the processor is non-operational.

12. The non-transitory machine-readable storage medium of claim 10, wherein the instructions upon execution cause the controller to:
- access, by the boot code, a code image from a storage; and
 - replace, by the boot code, the information with the code image to recover from the corruption of the information.
13. A method comprising:
- detecting, by an embedded controller in a device, corruption of information comprising data and instructions executable by a core hardware logic in the device;
 - in response to detecting the corruption of the information, use a recovery code in a controller storage to trigger a recovery process by the core hardware logic;
 - enabling, by the core hardware logic in the recovery process, functioning of a processor that is separate from the controller; and
 - after enabling of the functioning of the processor by the core hardware logic, executing a boot code on the processor to retrieve a full code image for the core hardware logic, the full code image useable to recover the core hardware logic from the corruption of the information.
14. The method of claim 13, wherein the device comprises a shared nonvolatile memory to store the information, and a mass storage to store the full code image, the method further comprising:
- replacing the information in the shared nonvolatile memory with the full code image to recover the core hardware logic responsive to the corruption of the information.
15. The method of claim 13, further comprising:
- replacing a recovery code in the information with the recovery code retrieved from the controller storage; and
 - executing the replaced recovery code by the core hardware logic to perform the recovery process.

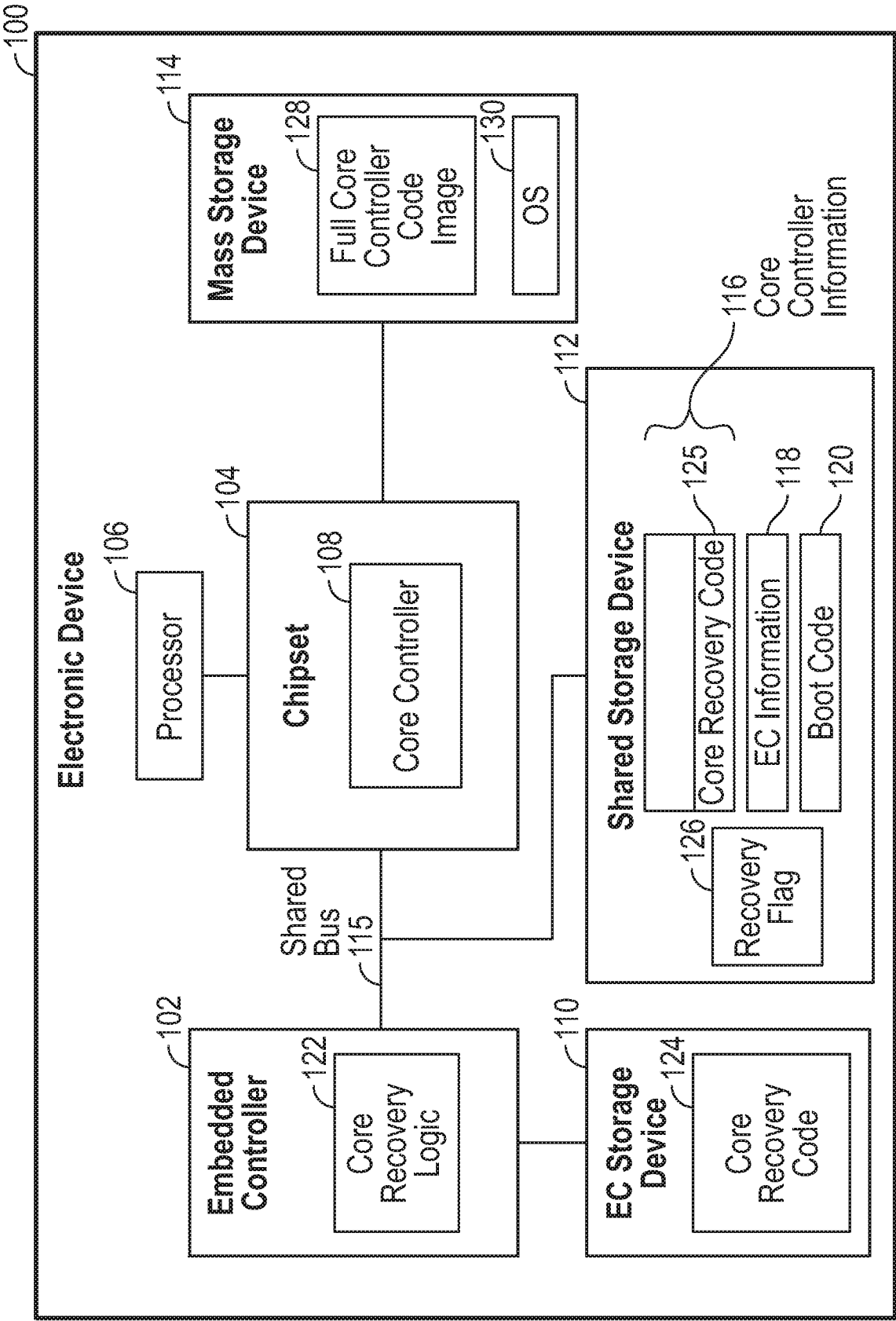


FIG. 1

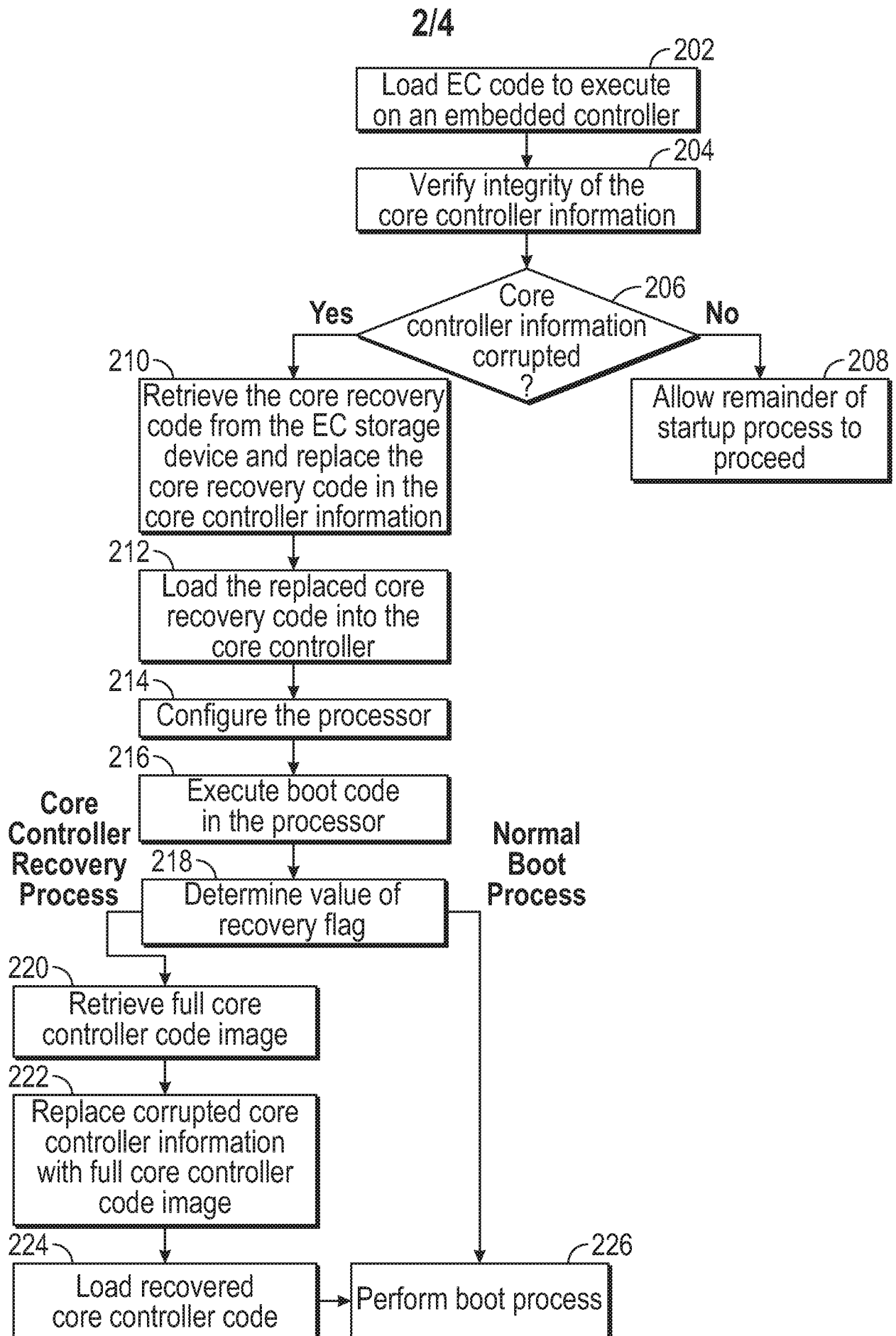


FIG. 2

3/4

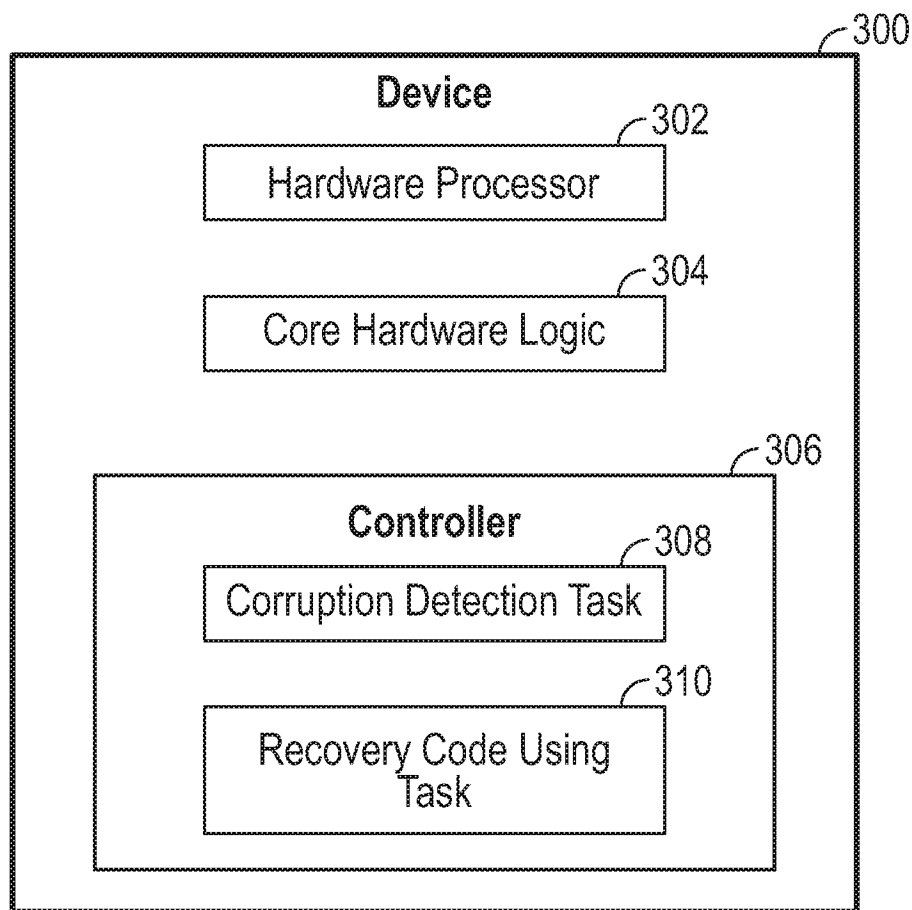


FIG. 3

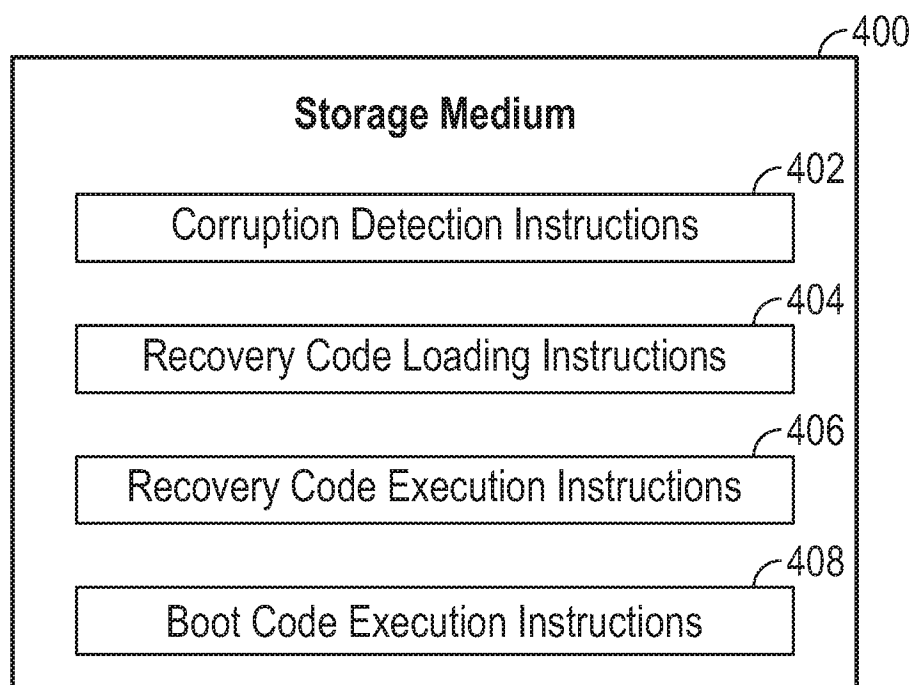


FIG. 4

4/4

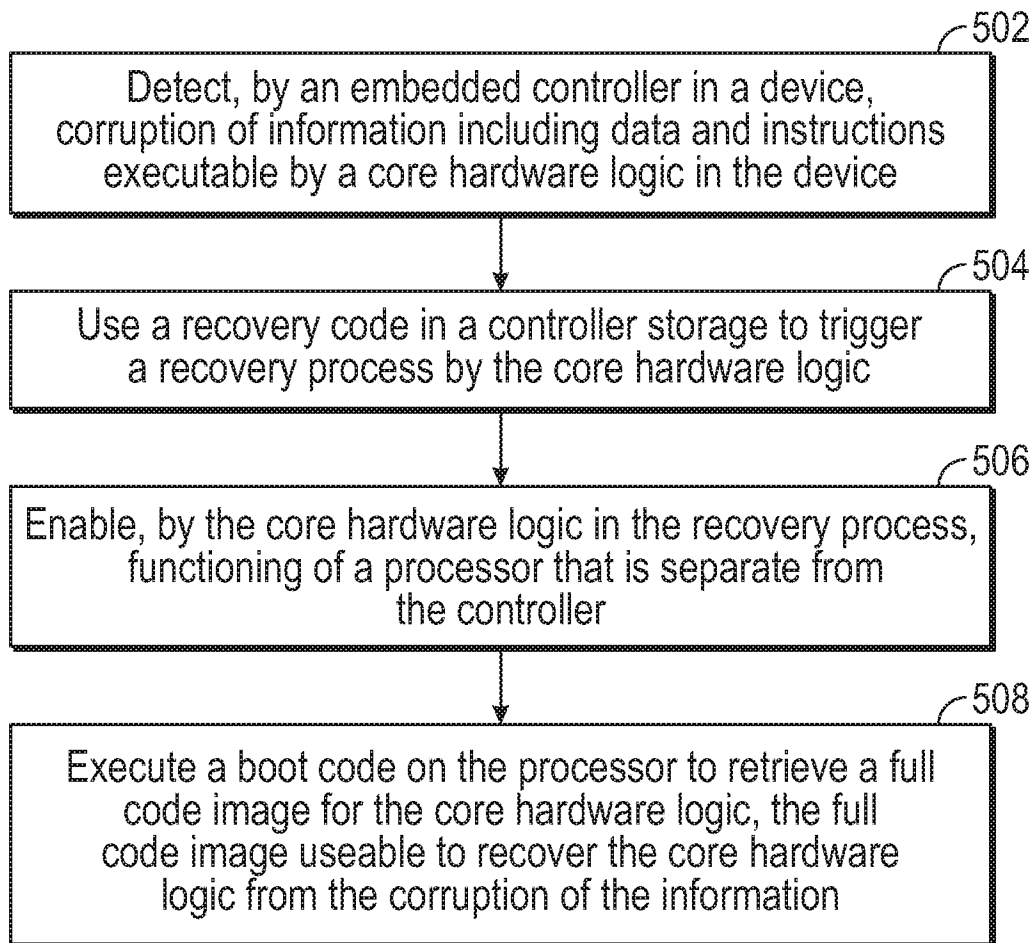


FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 2019/017459

A. CLASSIFICATION OF SUBJECT MATTER <p style="text-align: center;"><i>G06F 11/08 (2006.01)</i></p> <p>According to International Patent Classification (IPC) or to both national classification and IPC</p>				
B. FIELDS SEARCHED <p>Minimum documentation searched (classification system followed by classification symbols)</p> <p style="text-align: center;">G06F 11/08</p> <p>Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched</p> <p>Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)</p> <p style="text-align: center;">PatSearch (RUPTO Internal), USPTO, PAJ, Espacenet, Information Retrieval System of FIPS</p>				
C. DOCUMENTS CONSIDERED TO BE RELEVANT				
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.		
X	US 2018/0225167 A1 (INTEL CORPORATION) 09.08.2018, paragraphs [0022] – [0026], [0028], [0033] - [0035], [0059], [0067] - [0069], [0073], [0077], [0083], [0098], [0101], [0103], [0136], fig. 1	1-15		
A	US 4503535 A (INTEL CORPORATION) 05.03.1985	1-15		
A	US 6594785 B1 (UNISYS CORPORATION) 15.07.2003	1-15		
A	US 2002/0144177 A1 (THOMAS J. KONDO et al.) 03.10.2002	1-15		
☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.				
<p>* Special categories of cited documents:</p> <table border="0"> <tr> <td style="vertical-align: top;"> <p>“A” document defining the general state of the art which is not considered to be of particular relevance</p> <p>“E” earlier document but published on or after the international filing date</p> <p>“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)</p> <p>“O” document referring to an oral disclosure, use, exhibition or other means</p> <p>“P” document published prior to the international filing date but later than the priority date claimed</p> </td> <td style="vertical-align: top;"> <p>“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</p> <p>“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</p> <p>“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</p> <p>“&” document member of the same patent family</p> </td> </tr> </table>			<p>“A” document defining the general state of the art which is not considered to be of particular relevance</p> <p>“E” earlier document but published on or after the international filing date</p> <p>“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)</p> <p>“O” document referring to an oral disclosure, use, exhibition or other means</p> <p>“P” document published prior to the international filing date but later than the priority date claimed</p>	<p>“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</p> <p>“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</p> <p>“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</p> <p>“&” document member of the same patent family</p>
<p>“A” document defining the general state of the art which is not considered to be of particular relevance</p> <p>“E” earlier document but published on or after the international filing date</p> <p>“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)</p> <p>“O” document referring to an oral disclosure, use, exhibition or other means</p> <p>“P” document published prior to the international filing date but later than the priority date claimed</p>	<p>“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</p> <p>“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</p> <p>“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</p> <p>“&” document member of the same patent family</p>			
Date of the actual completion of the international search <p style="text-align: center;">07 October 2019 (07.10.2019)</p>		Date of mailing of the international search report <p style="text-align: center;">07 November 2019 (07.11.2019)</p>		
Name and mailing address of the ISA/RU: Federal Institute of Industrial Property, Berezhkovskaya nab., 30-1, Moscow, G-59, GSP-3, Russia, 125993 Facsimile No: (8-495) 531-63-18, (8-499) 243-33-37		Authorized officer <p style="text-align: center;">T. Mamedov</p> Telephone No. 8(495) 531-64-81		