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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 December 2007 (06.12.2007)

PCT

(10) International Publication Number
WO 2007/140300 A1

(51) International Patent Classification:
G06F 9/06 (2006.01)

(21) International Application Number:
PCT/US2007/069742

(22) International Filing Date: 25 May 2007 (25.05.2007)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/442,230 26 May 2006 (26.05.2006) US

(71) Applicant *(for all designated States except US)*: **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95052 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **DATTA, Shamanna** [US/US]; 532 NE Lenox Street, Hillsboro, Oregon 97124 (US). **KUMAR, Mohan** [IN/US]; 18680 SW Marko Lane, Aloha, Oregon 97007 (US).

(74) Agents: **VINCENT, Lester, J.** et al.; Blakely Sokoloff Taylor & Zafman, 12400 Wilshire Boulevard||7th Floor, Los Angeles, CA 90025 (US).

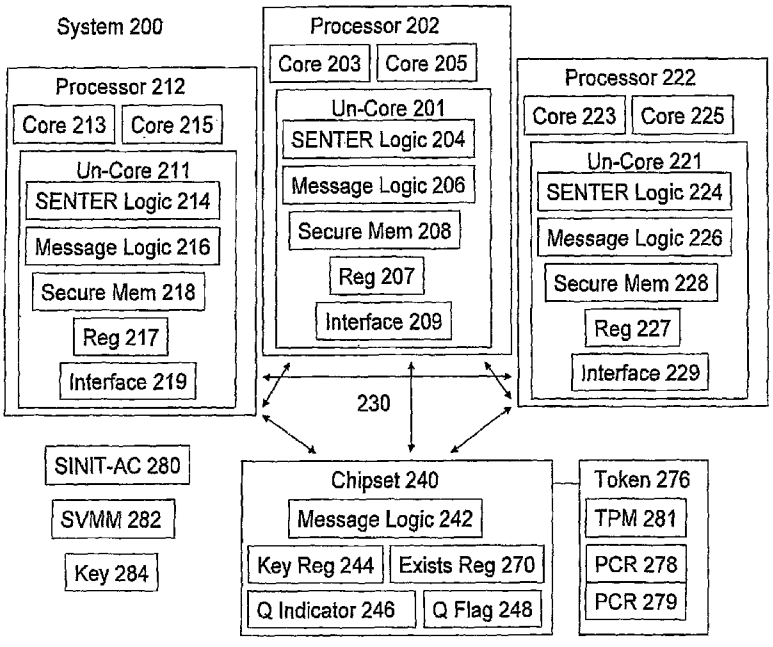
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, 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, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) **Title:** EXECUTION OF A SECURED ENVIRONMENT INITIALIZATION INSTRUCTION ON A POINT-TO-POINT INTERCONNECT SYSTEM



(57) **Abstract:** Methods and apparatus for initiating secure operations in a microprocessor system are described. In one embodiment, a system includes a processor to execute a secured enter instruction, and a chipset to cause the system to enter a quiescent state during execution of the secured enter instruction.

WO 2007/140300 A1

EXECUTION OF A SECURED ENVIRONMENT INITIALIZATION INSTRUCTION ON A POINT-TO-POINT INTERCONNECT SYSTEM

FIELD

[0001] The present invention relates generally to microprocessor systems, and more specifically to microprocessor systems that may operate in a trusted or secured environment.

BACKGROUND

[0002] The increasing number of financial and personal transactions being performed on local or remote microcomputers has given impetus for the establishment of "trusted" or "secured" microprocessor environments. The problem these environments try to solve is that of loss of privacy, or data being corrupted or abused. Users do not want their private data made public. They also do not want their data altered or used in inappropriate transactions. Examples of these include unintentional release of medical records or electronic theft of funds from an on-line bank or other depository. Similarly, content providers seek to protect digital content (for example, music, other audio, video, or other types of data in general) from being copied without authorization.

[0003] Existing trusted systems may utilize a complete closed set of trusted software. This method is relatively simple to implement, but has the disadvantage of not allowing the simultaneous use of common, commercially available operating system and application software. This disadvantage limits the acceptance of such a trusted system.

[0004] Other approaches require systems to have their processors connected to each other through a bus.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

[0006] **Figure 1** is a diagram of an exemplary software environment executing in a microprocessor system.

[0007] **Figure 2** is a diagram of certain exemplary trusted or secured software modules and exemplary system environment, according to one embodiment of the present invention.

[0008] **Figure 3** is a diagram of an exemplary trusted or secured software environment, according to one embodiment of the present invention.

[0009] **Figure 4** is a flowchart of software and other process blocks, according to a method embodiment of the present invention.

DETAILED DESCRIPTION

[0010] The following description describes techniques for initiating a trusted or secured environment in a microprocessor system. In the following description, numerous specific details such as logic implementations, software module allocation, encryption techniques, bus signaling techniques, and details of operation may be set forth in order to provide a more thorough understanding of the present invention. It will be appreciated, however, by one skilled in the art that the invention may be practiced without such specific details. In other instances, control structures, gate level circuits and full software instruction sequences have not been shown in detail in order not to obscure the invention. Those of ordinary skill in the art, with the included descriptions, will be able to implement appropriate functionality without undue experimentation. The invention is disclosed in the form of a microprocessor system. However, the invention may be practiced in other forms, such as in a digital signal processor, a minicomputer, or a mainframe computer.

[0011] Referring now to Figure 1, a diagram of an exemplary software environment executing in a microprocessor system is shown. The software shown in Figure 1 is not trusted (untrusted). When operating in a high privilege level, the size and constant updating of the operating system 150 make it very difficult to perform any trust analysis in a timely manner. Much of the operating system sits within privilege ring zero (0),

the highest level of privilege. The applications 152, 154, and 156 have much reduced privilege and typically reside within privilege ring three (3). The existence of the differing privilege rings and the separation of the operating system 150 and applications 152, 154 and 156 into these differing privileged rings would appear to allow operating of the software of Figure 1 in a trusted mode, based on making a decision to trust the facilities provided by the operating system 150. However, in practice making such a trust decision is often impractical. Factors that contribute to this problem include the size (number of lines of code) of the operating system 150, the fact that the operating system 150 may be the recipient of numerous updates (new code modules and patches) and the fact that the operating system 150 may also contain code modules such as device drivers supplied by parties other than the operating system developer. Operating system 150 may be a common one such as Microsoft ® Windows ®, Linux, or Solaris ®, or may be any other appropriate known or otherwise available operating system. The particular types or names of applications or operating systems run or running are not critical.

[0012] Referring now to Figure 2, a diagram of certain exemplary trusted or secured software modules and exemplary system environment 200 is shown, according to one embodiment of the present invention. In the Figure 2 embodiment, processor 202, processor 212, processor 222, and optional other processors (not shown) are shown as separate hardware entities. Although Figure 2 shows three processors, embodiments of the invention may include any number of processors, including a single processor.

[0013] System 200 also includes chipset 240, which may include an input/output (I/O) controller or hub, along with any other system or other logic as described below. Other embodiments may include other components instead or in addition to the components shown in Figure 2, and the number of processors or other components may differ, as may their arrangement with respect to each other.

[0014] Processors 202, 212, and 222 may include one or more execution cores, such as cores 203, 205, 213, 215, 223, and 225. In

some embodiments the processors or cores may be replaced by separate hardware execution threads running on one or more physical processors or cores. These threads possess many of the attributes of additional physical processors. In order to have a generic expression to discuss using any mixture of multiple physical processors and multiple threads upon processors, the expression "logical processor" may be used to describe either a physical processor or a thread operating in one or more physical processors. Thus, one single-threaded processor may be considered a logical processor, and multi-threaded or multi-core processors may be considered multiple logical processors.

[0015] Processors 202, 212, and 222 may also include "un-core" logic, such as un-core logic 201, 211, and 221, where "un-core" logic is logic that is separate from any of the execution cores. Un-core logic may include registers or any other storage locations, such as registers 207, 217, and 227, which may be used to store information regarding the processor on which is resides. Such registers or storage locations may be programmable or hard-coded, and such information may include the number of cores and/or logical processors included in the corresponding processor. Registers 207, 217, and 227 may be standard global memory mapped registers that are updated during power-on reset.

[0016] Processors 202, 212, and 222 may also contain certain special circuits or logic elements to support secure or trusted operations. For example, processor 202 may contain secure enter (SENTER) logic 204 to support the execution of special SENTER instructions that may initiate trusted operations. Processor 202 may also contain interconnection message logic 206 to support special interconnection messages between processors and other components in support of special SENTER operations. The use of special interconnection messages may increase the security or trustability of the system for several reasons. Circuit elements such as processors 202, 212, and 222 or chipset 240 may only issue or respond to such messages if they contain the appropriate logic elements of embodiments of the present disclosure. Therefore successful exchange of the special interconnection messages may help ensure proper system

configuration. Special interconnection messages may also permit activities that should normally be prohibited, such as resetting a platform configuration register 278. The ability of potentially hostile untrusted code to spy on certain interconnection transactions may be curtailed by allowing special interconnection messages to be issued only in response to special security instructions. SENTER logic 204, interconnection message logic 206, and any other logic used in embodiments of the invention may be implemented using any known approach, including logic circuitry, microcode, and firmware.

[0017] Additionally, processor 202 may contain secure memory 208 to support secure initialization operations. In one embodiment secure memory 208 may be an internal cache of processor 202, perhaps operating in a special mode. In alternate embodiments secure memory 208 may be special memory. Other processors such as processor 212 and processor 222 may also include SENTER logic 214, 224, bus message logic 216, 226, and secure memory 218, 228.

[0018] A "chipset" may be defined as a group of circuits and logic that support memory and/or I/O operations for a connected processor or processors. Individual elements of a chipset may be grouped together on a single chip, a pair of chips, or dispersed among multiple chips, including processors. In the Figure 2 embodiment, chipset 240 may include circuitry and logic to support I/O operations for processors 202, 212, and 222, while each processor includes circuitry and logic to support memory operations. Alternatively, chipset 240 may also include circuitry and logic to support memory operations for processors 202, 212, and 222. The functions of chipset 240 may be allocated among one or more physical devices in alternate embodiments.

[0019] Chipset 240 may additionally include its own interconnection message logic 242 to support special interconnection messages on PTP fabric 230 in support of special SENTER operations. Some of these special interconnection messages may include transferring the contents of a key register 244 to a processor 202, 212, or 222, permitting a processor to set or clear a special "QUIESCE" indicator 246 to cause chipset 240 to

quiesce or de-quiese system 200 (as described below), or permitting a special "QUIESCED" flag 248 to be examined by a processor. An additional feature of bus message logic 242 may be to register the existence or participation of processors in system 200 in an "EXISTS" register 270.

[0020] Processors 202, 212, 222 may be connected with each other, to chipset 240, and to any other components or agents through point-to-point (PTP) interconnection fabric 230. Processors 202, 212, and 222, and chipset 240 may include interface units 209, 219, 229, and 239, respectively, to connect to PTP fabric 230 and to transmit and receive messages to and from other each other and any other agents existing or participating in system 200. Each of interface units 209, 219, 229, and 239 may include any number of unidirectional and/or bidirectional ports for communication with any number of other components. Communications may be made through PTP fabric 230 according to a layered point-to-point interconnection architecture, for example, where packets, including signals representing a messages and/or data, framed by any or all of a linking layer, a protocol layer, a routing layer, a transport layer, a physical layer, and any other such layer, are transmitted from one agent to another agent (point-to-point). Accordingly, each of interface units 209, 219, 229, and 239 may include circuitry or logic to generate signals corresponding to each layer. The packets may also include redundant or other information for the detection or correction of errors.

[0021] Token 276, containing one or more platform configuration registers (PCR) 278, 279 may be connected to chipset 230. In one embodiment, token 276 may contain special security features, and in one embodiment may include the trusted platform module (TPM) 281 disclosed in the Trusted Computing Platform Alliance (TCPA) Main Specification, version 1.1a, 1 December 2001, issued by the TCPA (available at www.trustedpc.com).

[0022] Two software components identified in system environment 200 are a Secure Virtual Machine Monitor (SVMM) 282 module and a Secure

Initialization Authenticated Code (SINIT-AC) 280 module. The SVMM 282 module may be stored on a system disk or other mass storage, and moved or copied to other locations as necessary. In one embodiment, prior to beginning the secure launch process SVMM 282 may be moved or copied to one or more memory pages in system 200. Following the secure enter process, a virtual machine environment may be created in which the SVMM 282 may operate as the most privileged code within the system, and may be used to permit or deny direct access to certain system resources by the operating system or applications within the created virtual machines.

[0023] Some of the actions required by the secure enter process may be beyond the scope of simple hardware implementations, and may instead advantageously use a software module whose execution may be implicitly trusted. In one embodiment, these actions may be performed by Secure Initialization (SINIT) code. One exemplary action may require that various controls representing critical portions of the system configuration be tested to ensure that the configuration supports the correct instantiation of the secure environment. A second exemplary action may be to calculate and register the SVMM 282 module's identity and transfer system control to it. Here "register" means placing a trust measurement of SVMM 282 into a register or other storage location, for example into PCR 278 or into PCR 279. When this second action is taken, the trustworthiness of the SVMM 282 may be inspected by a potential system user.

[0024] The SINIT code may be produced by the manufacturer of the processors or of the chipsets. For this reason, the SINIT code may be trusted to aid in the secure launch of chipset 240. In order to distribute the SINIT code, in one embodiment a well-known cryptographic hash is made of the entire SINIT code, producing a value known as a "digest". One embodiment produces a 160-bit value for the digest. The digest may then be encrypted by a private key, held in one embodiment by the manufacturer of the processor, to form a digital signature. When the SINIT code is bundled with the corresponding digital signature, the

combination may be referred to as SINIT authenticated code (SINIT-AC) 280. Copies of the SINIT-AC 280 may be later validated as discussed below.

[0025] The SINIT-AC 280 may be stored on system disk or other mass storage or in a fixed media, and moved or copied to other locations as necessary. In one embodiment, prior to beginning the secure launch process SINIT-AC 280 may be moved or copied into one or more memory pages of system 200 to form a memory-resident copy of SINIT-AC.

[0026] Any privileged software running on system 200, such as an operating system, may initiate the secure launch process on a logical processor, which may then be referred to as the initiating logical processor (ILP). In the present example processor 202 is the ILP, although any of the processors on PTP fabric 230 could be the ILP. Neither memory-resident copy of SINIT-AC 280 nor memory-resident copy of SVMM 282 may be considered trustworthy at this time.

[0027] The ILP (processor 202) executes a special instruction to initiate the secure launch process. This special instruction may be referred to as a secured enter (SENTER) instruction, and may be supported by SENTER logic 204. The SENTER instruction may first verify that every logical processor in system 200 is registered in chipset 240, for example in EXISTS register 270. Each processor and other agent connected to PTP fabric 230 includes a register or other storage location, such as registers 207, 217, and 227, to indicate the number of logical processors that it includes. These registers are read to verify that the system topology is accurately represented in chipset 240.

[0028] After this verification, the SENTER instruction writes to QUIESCE indicator 246 to cause chipset 240 to quiesce system 200. Chipset 240 begins a handshake sequence on PTP fabric 230 to cause all processors and other agents on PTP fabric 230, except for one processor (the quiescent state master), to enter a quiescent state. In this embodiment, processor 202 is the quiescent state master as well as the ILP. The quiescence sequence may include sending a "STOP_REQ" signal to each non-master processor to cause them to finish their event

processing, drain their buffers, and return a "STOP_ACK" signal back to chipset 240 to acknowledge their entry into a quiescent state. The quiescent state is a state in which they execute no instructions and generate no transactions on PTP fabric 230. After chipset 240 receives a STOP_ACK signal from every agent on behalf of every logical processor registered in chipset 240, QUIESCED flag 248 may be set.

[0029] After the system is quiesced, the SENTER instruction may securely execute the security module(s) as described below. For this purpose, PTP fabric 230 is still functional and TPM 281 is still accessible to the quiescent state master. After the execution of the security module(s) is complete, the quiescent state master may clear QUIESCE indicator 246 to cause chipset 240 to bring system 200 out of the quiesced state.

[0030] To execute the security module(s), the ILP (processor 202) may first move both a copy of SINIT-AC 280 and key 284 into secure memory 208 for the purpose of authenticating and subsequently executing the SINIT code included in SINIT-AC 280. In one embodiment, this secure memory 208 may be an internal cache of the ILP (processor 202), perhaps operating in a special mode. Key 284 represents the public key corresponding to the private key used to encrypt the digital signature included in the SINIT-AC 280 module, and is used to verify the digital signature and thereby authenticate the SINIT code. In one embodiment, key 284 may already be stored in the processor, perhaps as part of the SENTER logic 204. In another embodiment, key 284 may be stored in a read-only key register 244 of chipset 240, which is read by the ILP. In yet another embodiment, either the processor or the chipset's key register 244 may actually hold a cryptographic digest of key 284, where key 284 itself is included in the SINIT-AC 280 module. In this last embodiment, the ILP reads the digest from key register 244, calculates an equivalent cryptographic hash over the key 284 embedded in SINIT-AC 280, and compares the two digests to ensure the supplied key 284 is indeed trusted.

[0031] A copy of SINIT-AC and a copy of a public key may then exist within secure memory 208. The ILP may now validate the copy of SINIT-AC by decrypting the digital signature included in the copy of the SINIT-AC using the copy of a public key. This decryption produces an original copy of a cryptographic hash's digest. If a newly-calculated digest matches this original digest then the copy of SINIT-AC and its included SINIT code may be considered trustable.

[0032] The ILP may now register the unique identity of the SINIT-AC module by writing the SINIT-AC module's cryptographic digest value to a platform configuration register 272 in the security token 276, as outlined below. The ILP's execution of its SENTER instruction may now terminate by transferring execution control to the trusted copy of the SINIT code held within the ILP's secure memory 208. The trusted SINIT code may then perform its system test and configuration actions and may register the memory-resident copy of SVMM, in accordance with the definition of "register" above.

[0033] Registration of the memory-resident copy of SVMM may be performed in several manners. In one embodiment, the SENTER instruction running on the ILP writes the calculated digest of SINIT-AC into PCR 278 within the security token 276. Subsequently, the trusted SINIT code may write the calculated digest of the memory-resident SVMM to the same PCR 278 or another PCR 279 within the security token 276. If the SVMM digest is written to the same PCR 278, the security token 276 hashes the original contents (SINIT digest) with the new value (SVMM digest) and writes the result back into the PCR 278. In embodiments where the first (initializing) write to PCR 278 is limited to the SENTER instruction, the resulting digest may be used as a root of trust for the system.

[0034] Once the trusted SINIT code has completed its execution, and has registered the identity of the SVMM in a PCR, the SINIT code may transfer ILP execution control to the SVMM. In a typical embodiment, the first SVMM instructions executed by the ILP may represent a self-

initialization routine for the SVMM. System 200 may then be operated in trusted mode, as outlined in the discussion of Figure 3 below, under the supervision of the now-executing copy of SVMM. From this point onwards, the overall system is operating in trusted mode as outlined in the discussion of Figure 3 below.

[0035] Referring now to Figure 3, a diagram of an exemplary trusted or secured software environment is shown, according to one embodiment of the present invention. In the Figure 3 embodiment, trusted and untrusted software may be loaded simultaneously and may execute simultaneously on a single computer system. A SVMM 350 selectively permits or prevents direct access to hardware resources 380 from one or more untrusted operating systems 340 and untrusted applications 310 through 330. In this context, "untrusted" does not necessarily mean that the operating system or applications are deliberately misbehaving, but that the size and variety of interacting code makes it impractical to reliably assert that the software is behaving as desired, and that there are no viruses or other foreign code interfering with its execution. In a typical embodiment, the untrusted code might consist of the normal operating system and applications found on today's personal computers.

[0036] SVMM 350 also selectively permits or prevents direct access to hardware resources 380 from one or more trusted or secure kernels 360 and one or more trusted applications 370. Such a trusted or secure kernel 360 and trusted applications 370 may be limited in size and functionality to aid in the ability to perform trust analysis upon it. The trusted application 370 may be any software code, program, routine, or set of routines which is executable in a secure environment. Thus, the trusted application 370 may be a variety of applications, or code sequences, or may be a relatively small application such as a Java applet.

[0037] Instructions or operations normally performed by operating system 340 or kernel 360 that could alter system resource protections or privileges may be trapped by SVMM 350, and selectively permitted, partially permitted, or rejected. As an example, in a typical embodiment,

instructions that change the processor's page table that would normally be performed by operating system 340 or kernel 360 would instead be trapped by SVMM 350, which would ensure that the request was not attempting to change page privileges outside the domain of its virtual machine.

[0038] Referring now to Figure 4, a flowchart of software and other process blocks is shown, according to an embodiment of the present invention in method 400.

[0039] In block 410 of method 400, a logical processor makes a copy of the SINIT-AC and SVMM modules available for access by a subsequent SENTER instruction. In this example, an ILP loads the SINIT-AC and SVMM code from mass storage into physical memory. In alternative embodiments, any logical processor may do so, not just the ILP. A processor becomes the ILP by executing the SENTER instruction, as identified in block 412.

[0040] In block 420, the SENTER instruction verifies that every logical processor in system 200 is registered in chipset 240, for example in EXISTS register 270. In block 422, the SENTER instruction writes to QUIESCE indicator 246 to cause chipset 240 to quiesce system 200. In block 424, chipset 240 sends "STOP_REQ" signals to each non-master processor to cause them to finish their event processing, drain their buffers, and return a "STOP_ACK" signal back to chipset 240 to acknowledge their entry into a quiescent state. In block 426, QUIESCED flag 248 is set to indicate that chipset 240 has received a STOP_ACK signal from every agent on behalf of every logical processor registered in chipset 240.

[0041] In block 430, the ILP moves the public key of the chipset and the memory-resident copy of SINIT-AC into its own secure memory for secure execution. The ILP, in block 432, uses the key to validate the secure-memory-resident copy of SINIT-AC, and then executes it. The execution of SINIT-AC may perform tests of the system configuration and the SVMM copy, then registers the SVMM identity, and finally begins the execution of

SVMM in block 434. In block 436, the quiescent state master clears QUIESCE indicator 246 to cause chipset 240 to bring system 200 out of the quiesced state in block 438.

[0042] In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

CLAIMS

What is claimed is:

1. A system, comprising:

a first processor to execute a secured enter instruction; and

a chipset to cause the system to enter a quiescent state during execution of the secured enter instruction.

2. The system of claim 1, further comprising a point-to-point fabric to couple the first processor to the chipset.

3. The system of claim 2, further comprising a second processor coupled to the chipset by the point-to-point fabric, wherein the chipset is also to cause the second processor to enter the quiescent state during the execution of the secured enter instruction.

4. The system of claim 2, wherein the chipset is also to cause the system to enter the quiescent state in which only the first processor and the chipset communicate through the point-to-point fabric.

5. The system of claim 1, further comprising a trusted platform module, wherein the trusted platform module is accessible to the first processor during the quiescent state.

6. The system of claim 1, wherein the first processor is also to execute a secure initialization authenticated code module.

7. The system of claim 1, wherein the first processor is also to execute a secure virtual machine monitor module.

8. The system of claim 1, wherein the chipset includes an indicator accessible to the first processor to cause the chipset to cause the system to enter the quiescent state.

9. The system of claim 3, wherein:

the chipset includes:

an indicator accessible to the first processor to cause the chipset to cause the system to enter the quiescent state; and

a first storage location to indicate the identity of each processor in the system;

the second processor includes a second storage location to indicate that the second processor is connected to the point-to-point fabric; and

the first processor includes logic to verify that the identity of the second processor is indicated in the first storage location before accessing the indicator to cause the chipset to cause the system to enter the quiescent state.

10. A method, comprising:

starting to execute a secured enter instruction on a first processor coupled to a chipset through a point-to-point fabric;

causing the chipset to cause a second processor coupled to the point-to-point fabric to enter a quiescent state during the execution of a secured enter instruction.

11. The method of claim 10, further comprising the chipset maintaining a list of agents coupled to the point-to-point fabric.

12. The method of claim 11, further comprising reading a storage location in the second processor to identify the number of agents coupled to the point-to-point fabric through the second processor.

13. The method of claim 10, further comprising the first processor accessing a trusted platform module during the quiescent state.

14. The method of claim 13, further comprising executing a secure initialization authentication module.

15. The method of claim 14, further comprising reading an indicator in the chipset to verify that the quiescent state has been entered before executing the secure initialization authentication module.

16. The method of claim 13, further comprising executing a secure virtual machine monitor module.

17. The method of claim 16, further comprising reading an indicator in the chipset to verify that the quiescent state has been entered before executing the secure virtual machine monitor module.

18. The method of claim 10, further comprising causing the second processor to finish its event processing and drain its buffers before entering the quiescent state.

19. The method of claim 18, further comprising the second processor sending a signal to the chipset to acknowledge its entry into the quiescent state.

20. A processor, comprising:

secure enter logic to execute a first instruction to invoke secure operation initialization; and

interconnection messaging logic to cause a chipset to cause an agent coupled to the chipset through a point-to-point fabric to enter a quiescent state.

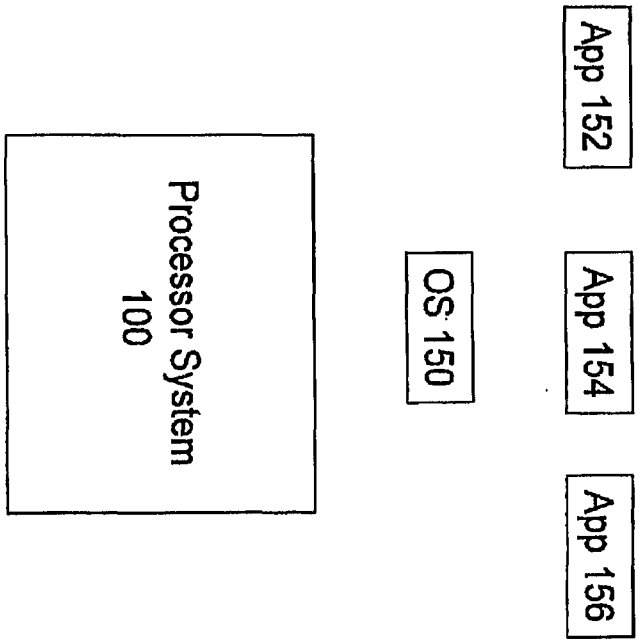


FIGURE 1

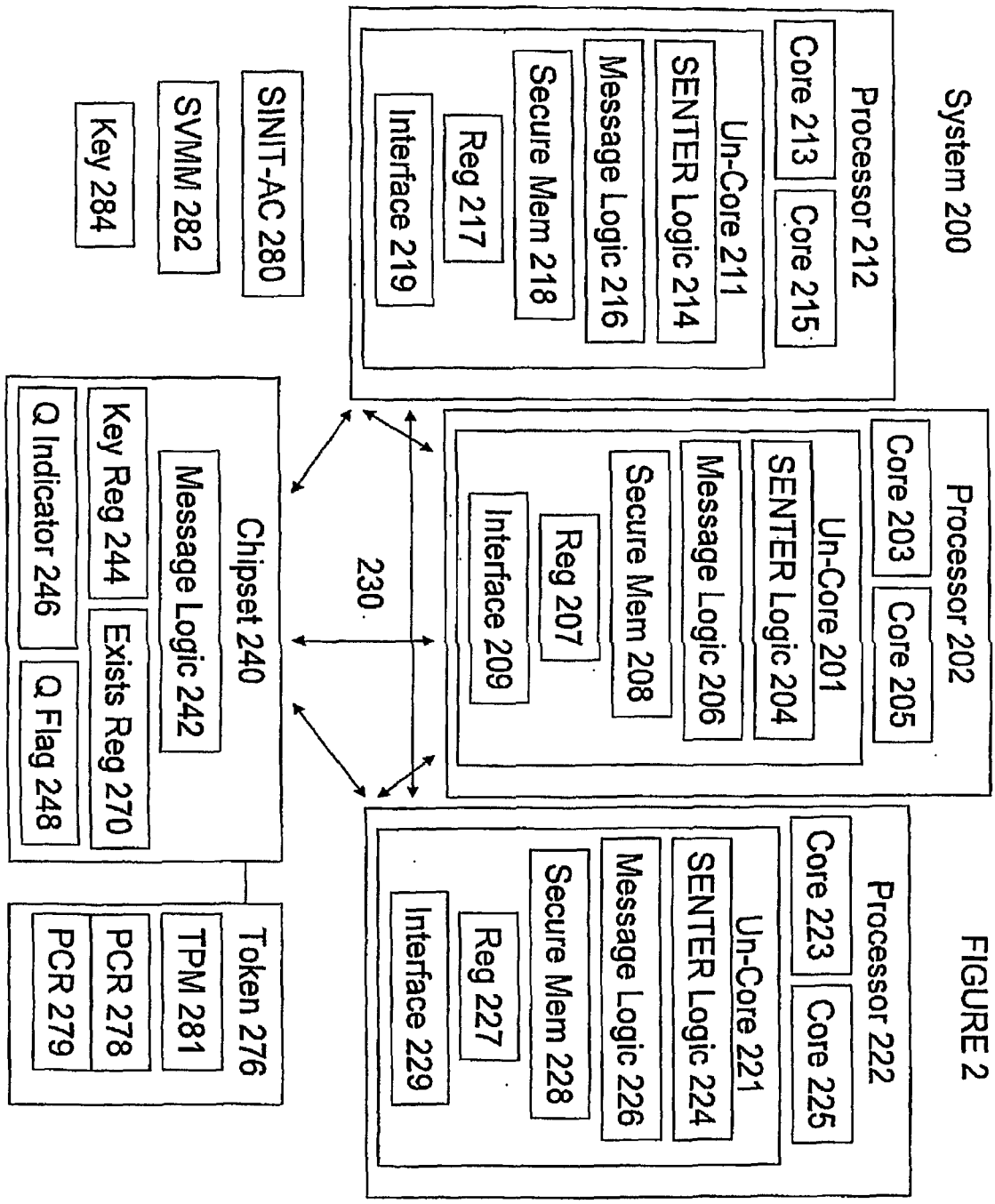

System 200
Processor 212
Core 213
Core 215
Un-Core 211
SENTER Logic 214
Message Logic 216
Secure Mem 218
Reg 217
Interface 219
Processor 202
Core 203
Core 205
Un-Core 201
SENTER Logic 204
Message Logic 206
Secure Mem 208
Reg 207
Interface 209
Processor 222
Core 223
Core 225
Un-Core 221
SENTER Logic 224
Message Logic 226
Secure Mem 228
Reg 227
Interface 229
230
SINIT-AC 280
SVMM 282
Key 284
Chipset 240
Message Logic 242
Key Reg 244
Exists Reg 270
Q Indicator 246
Q Flag 248
Token 276
TPM 281
PCR 278
PCR 279

FIGURE 2

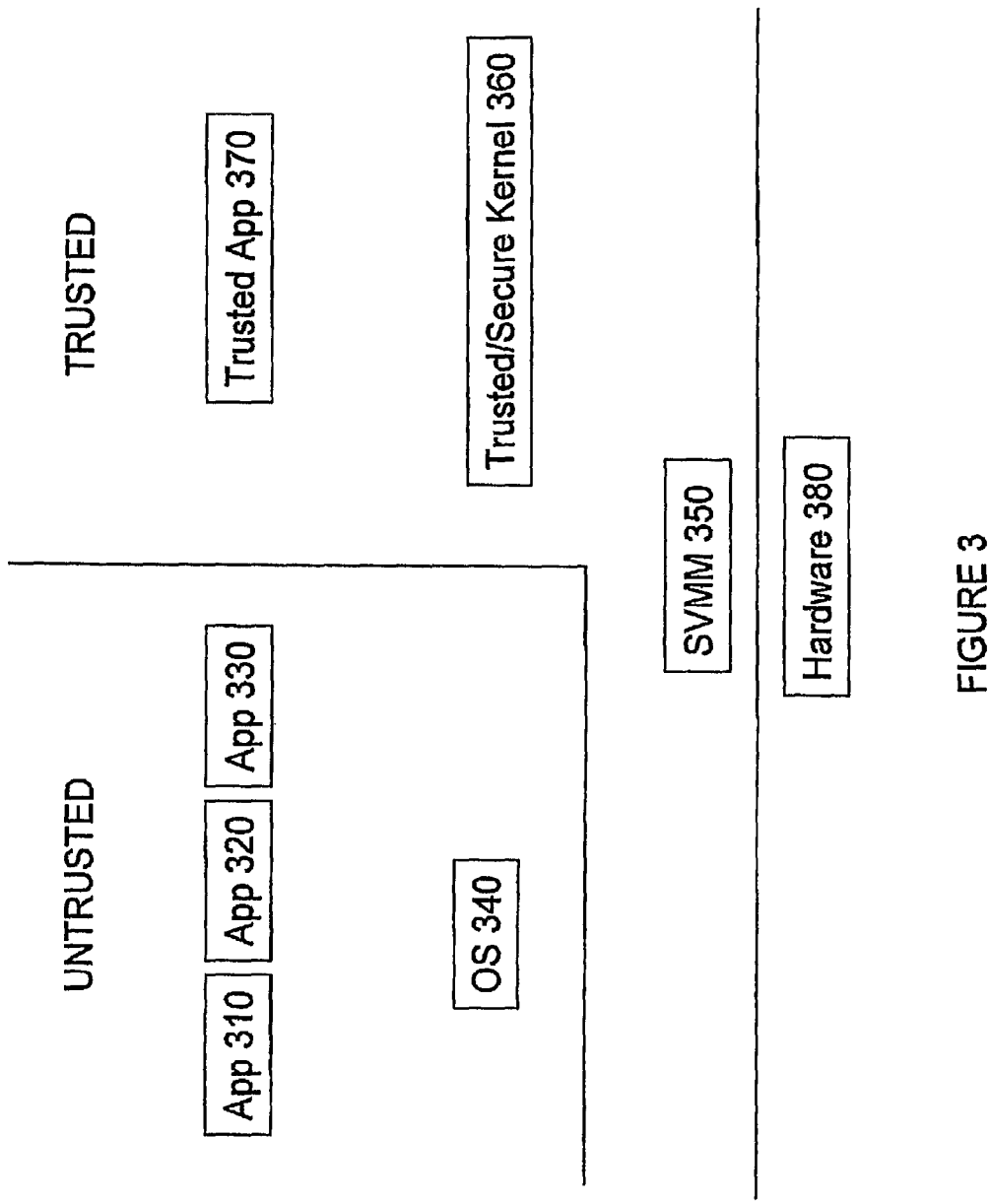

TRUSTED
Trusted App 370
Trusted/Secure Kernel 360
UNTRUSTED
App 310
App 320
App 330
OS 340
SVMM 350
Hardware 380

FIGURE 3

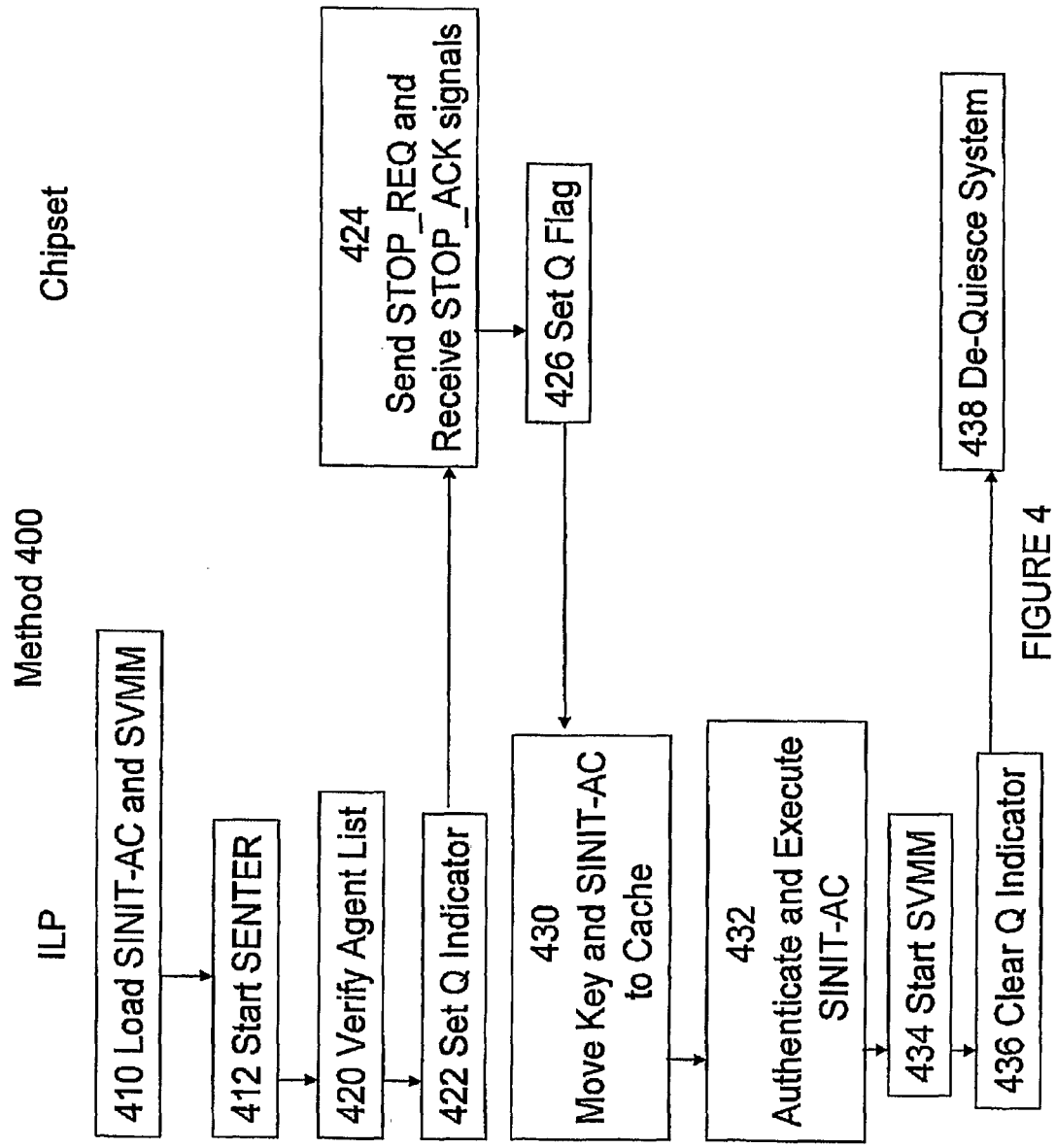

Method 400
ILP
Chipset
410 Load SINIT-AC and SVMM
412 Start SENTER
420 Verify Agent List
422 Set Q Indicator
424
Send STOP_REQ and
Receive STOP_ACK signals
426 Set Q Flag
430
Move Key and SINIT-AC
to Cache
432
Authenticate and Execute
SINIT-AC
434 Start SVMM
436 Clear Q Indicator
438 De-Quiesce System

FIGURE 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2007/069742

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/06(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)
IPC 8 G06F, H04Q

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean Utility models and applications for Utility models since 1975
Japanese Utility models and applications for Utility models since 1975

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKIPASS(KIPO internal) "Keywords: multiple processor, chipset, secured enter instruction, quiescent state and similar terms"

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5,905,861 A (LOVELL, W. S.) 18 May 1999 See the abstract, figures 2-4, and col. 3 line 49 - col. 4 line 58.	1-20
A	US 6,754,829 B1 (BUTT, A. B., et al.) 22 June 2004 See the abstract, figures 1 and 6, col. 3 line 14 - col. 5 line 25, and col. 10 line 16 - col. 11 line 25.	1-20
A	US 6,035,382 A (LEE, R. D. AND CURRY, S. J.) 07 March 2000 See the abstract, figures 1 and 20, and col.3 lines 31-42.	1-20
A	US 5,659,750 A (PRIEM, C. AND ROSENTHAL, D.) 19 August 1997 See the abstract, figure 2, and col. 5 lines 37-65.	1-20

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
21 SEPTEMBER 2007 (21.09.2007)	**21 SEPTEMBER 2007 (21.09.2007)**

Name and mailing address of the ISA/KR	Authorized officer
Korean Intellectual Property Office 920 Dunsan-dong, Seo-gu, Daejeon 302-701, Republic of Korea	NHO, Ji Myong
Facsimile No. 82-42-472-7140	Telephone No. 82-42-481-8528

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2007/069742

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 05905861 A	18.05.1999	None	
US 06754829 B1	22.06.2004	None	
US 06035382 A	07.03.2000	None	
US 05659750 A	19.08.1997	US 05764861 A	09.06.1998