



- (51) International Patent Classification:
G06F 12/08 (2006.01) *G06F 15/163* (2006.01)
G06F 13/364 (2006.01)
- (21) International Application Number:
PCT/EP2009/062554
- (22) International Filing Date:
28 September 2009 (28.09.2009)
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data:
12/241,332 30 September 2008 (30.09.2008) US
- (71) Applicant (for all designated States except US): **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, New York 10504 (US).
- (72) Inventors; and
(75) Inventors/Applicants (for US only): **KIM, Daeik** [KR/US]; IBM Corporation, MP 27A, 2070 Route 52, Hopewell Junction, New York, New York 12533-6683 (US). **DUVALSAINT, Karl** [US/US]; IBM Corporation, MP P328, 2455 South Road, Poughkeepsie, New York, New York 12601-5400 (US). **HOFSTEE, Harm, Peter** [NL/US]; IBM Corporation, MP 9062007D, 11501 Burnet Road, Austin, Texas 78758-3400 (US). **KIM, MOON, Ju** [US/US]; IBM Corporation, Dept IQ0A/Bldg 040-3, 1701 North Street, Endicott, New York 13760 (US).
- (74) Agent: **LING, Christopher, John**; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester Hampshire SO21 2JN (GB).
- (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, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO,

[Continued on next page]

(54) Title: DELEGATED VIRTUALIZATION IN A MULTI-CORE PROCESSOR (MCP)

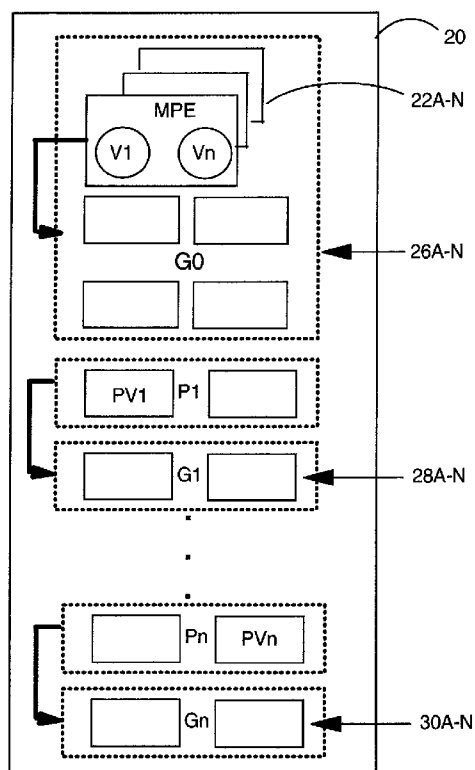


FIG. 2

(57) Abstract: The disclosure is applied to a generic microprocessor architecture with a set (e.g., one or more) of controlling elements (e.g., MPEs) and a set of groups of sub-processing elements (e.g., SPEs). Under this arrangement, MPEs and SPEs are organized in a way that a smaller number MPEs control the behavior of a group of SPEs using program code embodied as a set of virtualized control threads. The arrangement also enables MPEs delegate functionality to one or more groups of SPEs such that those group(s) of SPEs will act as pseudo MPEs. The pseudo MPEs will utilize pseudo virtualized control threads to control the behavior of other groups of SPEs. In a typical embodiment, the apparatus includes a MCP coupled to a power supply coupled with cores to provide a supply voltage to each core (or core group) and controlling-digital elements and multiple instances of sub-processing elements.



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, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, 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, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

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

DELEGATED VIRTUALIZATION IN A MULTI-CORE PROCESSOR (MCP)

FIELD OF THE INVENTION

The present invention generally relates to delegated virtualization in a (e.g., symmetric) multi-core processor (MCP). Specifically, the present invention delegates functions found in a main processing element to a group of sub-processing elements to yield a pseudo main processing element.

BACKGROUND OF THE INVENTION

Low utilization of Multi-Core Processors (MCPs) has been a major drawback of symmetric MCPs. Also, design inflexibility forces continuous leakage current in the unloaded and stand-by sub-elements, such as Sub-Processing Element (SPE), so that the power is wasted. For example, in a symmetric MCP, there can be a Main Processing Element (MPE) and 8 SPEs. In many cases, only a portion of SPEs are utilized and the overall MCP utilization is usually low. Such stand-by SPEs consume high levels of power and continuously leak. Typically, a MCP is used for the high performance digital processor scaling, but due to the complexity of the MCP design, the utilization and the efficiency of the software become challenging to optimize as the MCP dimension increases.

SUMMARY OF THE INVENTION

The present invention provides, in a first aspect, a multi-core processor, comprising: a main processing element comprising a set of virtualized control threads; a first group of sub-processing elements associated with main processing element via at least one set of virtualized control threads; and a group of sub-processing elements designated as a pseudo main processing element, the pseudo main processing element comprising a set of pseudo virtualized control threads.

Preferably, the pseudo main processing element is associated with a second group of sub-processing elements via the set of pseudo virtualized control threads.

Preferably, the first group of sub-processing elements comprises a plurality of sub-processing elements.

Preferably, the set of virtualized control threads is configured to send program code and data to the first group of sub-processing elements.

Preferably, the set of virtualized control threads is further configured to collect computation results from the first group of sub-processing elements.

Preferably, the set of virtualized control threads controls a clock speed, power consumption and computation loading of the set of first group of sub-processing elements.

Preferably, the set of virtualized control threads is configured to send and receive power control requests between the main processing element and the first group of sub-processing elements.

Preferably, the pseudo main processing element is associated with a second group of sub-processing elements via the set of pseudo virtualized control threads.

In a second aspect, there is provided a processing system, comprising: a main processing element; a first group of sub-processing elements; a virtualized control thread associating the main processing element with the first group of sub-processing elements, the set of virtualized control threads controlling the first group of sub-processing elements; a pseudo main processing element; a second group of sub-processing elements; and a pseudo virtualized control thread associating the pseudo main processing element with the second group of sub-processing elements, the set of virtualized control threads controlling the group of sub-processing elements.

Preferably, the pseudo main processing element is a third group of sub-processing elements that have been delegated a set of functions from at least one of a set of main processing elements.

Preferably, the virtualized control thread and the pseudo virtualized control thread are configured to send program code and data to the first group of sub-processing elements and the second group of sub-processing elements.

Preferably, the virtualized control thread and the pseudo virtualized control thread are further configured to collect computation results from the first group of sub-processing elements and the second group of sub-processing elements.

Preferably, the virtualized control thread and the pseudo virtualized control thread control a clock speed, power consumption and computation loading of the first group of sub-processing elements and the second group of sub-processing elements.

In a third aspect, there is provided a processing method, comprising: associating a main processing element with a first group of sub-processing element using a set of virtualized control threads; delegating a set of functions of the main processing element to a second group of sub-processing elements to cause the second group of sub-processing elements to act as a pseudo main processing element; associating the pseudo main processing element with a third group of sub-processing elements using a set of pseudo virtualized control threads; and controlling the first group of sub-processing elements using the set of virtualized control threads and the third group of sub-processing elements using the set of pseudo virtualized control threads.

The processing method may further comprise sending program code and data to the first group of sub-processing elements via the set of virtualized control threads and to the third set of groups of sub-processing elements via the set of pseudo virtualized control threads.

The processing method may further comprise receiving computation results from the first group of sub-processing elements via the set of virtualized control threads and from the third group of sub-processing elements via the set of pseudo virtualized control threads.

The controlling may comprise controlling a clock speed, a power consumption and a computation loading of the first group of sub-processing elements and the third group of sub-processing elements.

The set of virtualized control threads may be embodied as program code within the set of main processing elements, and the set of pseudo virtualized control threads being embodied as program code within the pseudo main processing element. .

In the processing method, the main processing element, the first group of sub-processing elements, the set of virtualized control threads, the pseudo main processing element, the third group of sub-processing elements, and the set of pseudo virtualized control threads may comprise a multi-core processor.

In a fourth aspect, there is provided a method for deploying a processing system, comprising: providing a multi-core processor comprising: a main processing element comprising a set of virtualized control threads; a first group of sub-processing elements associated with main processing element via at least one of set of virtualized control threads; and a group of sub-processing elements designated as a pseudo main processing element, the pseudo main element comprising a set of pseudo virtualized control threads.

This disclosure thus describes an apparatus, computer architecture, method, operating system, compiler, and application program products for MPEs as well as delegated virtualization in a (e.g., symmetric) MCP. The disclosure is applied to a generic microprocessor architecture with a set (e.g., one or more) of controlling elements (e.g., MPEs) and a set of groups of sub-processing elements (e.g., SPEs). Under this arrangement, MPEs and SPEs are organized in a way that a smaller number MPEs control the behavior of a group of SPEs using program code embodied as a set of virtualized control threads. The arrangement also enables MPEs to delegate functionality to one or more groups of SPEs such that those group(s) of SPEs may act as pseudo MPEs. The pseudo MPEs may utilize pseudo virtualized control threads to control the behavior of other groups of SPEs. In a typical embodiment, the apparatus includes a MCP coupled to a power supply coupled with cores to provide a supply voltage to each core (or core group) and controlling-digital

elements and multiple instances of sub-processing elements.

A first embodiment of the present invention provides a multi-core processor, comprising: a main processing element comprising a set of virtualized control threads; a first group of sub-processing elements associated with main processing element via at least one set of virtualized control threads; and a group of sub-processing elements designated as a pseudo main processing element, the pseudo main element comprising a set of pseudo virtualized control threads.

A second embodiment of the present invention provides a processing system comprising: a main processing element; a first group of sub-processing elements; a virtualized control thread associating the main processing element with the first group of sub-processing elements, the set of virtualized control threads controlling the first group of sub-processing elements; a pseudo main processing element; a second group of sub-processing elements; and a pseudo virtualized control thread associating the pseudo main processing element with the second group of sub-processing elements, the set of virtualized control threads controlling the group of sub-processing elements.

A third embodiment of the present invention provides a processing method comprising: associating a main processing element with a first group of sub-processing element using a set of virtualized control threads; delegating a set of functions of the main processing element to a second group of sub-processing elements to cause the second group of sub-processing elements to act as a pseudo main processing element; associating the pseudo main processing element with a third group of sub-processing elements using a set of pseudo virtualized control threads; and controlling the first group of sub-processing elements using the set of virtualized control threads and the third group of sub-processing elements using the set of pseudo virtualized control threads.

A fourth embodiment of the present invention provides a method for deploying a processing system comprising: providing a multi-core processor comprising: a main processing element comprising a set of virtualized control threads; a first group of sub-processing elements associated with main processing element via at least one of set of virtualized control threads;

and a group of sub-processing elements designated as a pseudo main processing element, the pseudo main element comprising a set of pseudo virtualized control threads.

A fifth embodiment of the present invention provides a computer-implemented business method comprising: associating a main processing element with a first group of sub-processing element using a set of virtualized control threads; delegating a set of functions of the main processing element to a second group of sub-processing elements to cause the second group of sub-processing elements to act as a pseudo main processing element; associating the pseudo main processing element with a third group of sub-processing elements using a set of pseudo virtualized control threads; and controlling the first group of sub-processing elements using the set of virtualized control threads and the third group of sub-processing elements using the set of pseudo virtualized control threads.

BRIEF DESCRIPTION OF THE DRAWINGS

A preferred embodiment of the present invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Fig. 1 shows a related art processor.

Fig. 2 shows multi-core processor according to the preferred embodiment.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

As indicated above, this disclosure describes an apparatus, computer architecture, method, operating system, compiler, and application program products for MPEs as well as delegated virtualization in a (e.g., symmetric) MCP. The disclosure is applied to a generic microprocessor architecture with a set (e.g., one or more) of controlling elements (e.g., MPEs) and a set of groups of sub-processing elements (e.g., SPEs). Under this arrangement, MPEs and SPEs are organized in a way that a smaller number MPEs control the behavior of a group of SPEs using program code embodied as a set of virtualized control threads. The arrangement also enables MPEs delegate functionality to one or more groups of SPEs such that those group(s) of SPEs will act as pseudo MPEs. The pseudo MPEs will utilize pseudo

virtualized control threads to control the behavior of other groups of SPEs. In a typical embodiment, the apparatus includes a MCP coupled to a power supply coupled with cores to provide a supply voltage to each core (or core group) and controlling-digital elements and multiple instances of sub-processing elements.

Under related art systems, such as that shown in Fig. 1, the MCP 10 provides a conventional operation: a MPE 12 controls a group of SPEs 14A-N. As only a portion of SPEs are utilized, the overall MCP utilization is usually low. Moreover, stand-by SPEs 16A-N consume high levels of power and continuously leak. Typically, a MCP is used for the high performance digital processor scaling, but due to the complexity of the MCP design, the utilization and the efficiency of the software become challenging to optimize as the MCP dimension increases.

To address these issues, a configuration such as that shown in Fig. 2 is provided. As depicted, MCP 20 comprises a set of MPEs 22A-N. Each MPE 22 comprises a set of virtualized control threads V1-VN. Under the preferred embodiment MPEs 22A-N allow multiple virtualized control threads V1-VN that are each assigned to/associated with a (e.g., single or multiple) group of SPEs. For example, as further shown in Fig. 1, MCP 20 further includes groups G0-Gn of SPEs. In the example shown, group G0 comprises SPEs 26A-N, group G1 comprises SPEs 28A-N, and group Gn comprises SPEs 30A-N. It should be understood that the configuration shown is for illustrative purposes only and neither the quantity of SPEs nor their grouping is intended to limit the teachings recited herein. In any event, as can be seen, virtualized control threads V1-Vn are each associated with a specific group of SPEs. That is, virtualized control thread V1 of MPE 22A is associated with SPE group G0. In a typical embodiment, there is a one-to-one relationship between threads and groups (however, this need not be the case in that a single virtualized control thread could be associated with multiple groups). In addition, all virtualized control threads V1-Vn are shown as being embodied within a single, common MPE 22. However, it should be understood that this need not be the case. For example, virtualized control threads could be embodied in two or more different MPEs 22A-N.

Under the preferred embodiment one or more groups of SPEs could be delegated function(s) from MPE 22A-N, which would render such groups of SPEs pseudo MPEs. Examples of this are shown in Fig. 2 (after delegation has occurred). Specifically, pseudo MPEs P1-Pn are shown in Fig. 2. Each pseudo MPE has pseudo virtualized control threads PV1-n, that perform similar or identical functionality as virtualized control threads V1-n in MPE 22A-n. Such delegation of functionality typically occurs by leveraging one or more virtualized control threads V1-n on MPE 22A.

In any event, each virtualized control thread and pseudo virtualized control thread serves to associate/link at least one MPE with at least one group of SPEs. This allows the SPEs to be controlled. Moreover, each virtualized control thread and pseudo virtualized control thread PV1-n is typically implemented as computer program code (e.g., contained on a computer readable medium). Virtualized control threads V1-n and pseudo virtualized control threads PV1-n have several functions including (among others) the following: sending program code and data to the set of groups of sub-processing elements; collecting computation results from the set of groups of sub-processing elements; sending and receiving power control requests between the set of main/pseudo processing elements and the set of groups of sub-processing element; and controlling a clock speed, power consumption and computation loading of the set of groups of sub-processing elements.

In the example shown in Fig. 2, pseudo MPE P1 is associated/linked with SPE group G1, while pseudo MPE Pn is associated/linked with SPE group Gn. Pseudo MPEs P1-Pn generally interact with and SPE control groups G1-n in a manner similar or identical to that which MPE 22A interacts with and controls group G0.

A more specific illustrative embodiment of the association of virtualized control threads with groups of SPEs will be given below:

Using the teachings recited above, each MPE and pseudo MPE allows additional virtualizations, based upon the SPEs availability and the number of requested SPEs. The MPE and pseudo MPE each keep a log of which SPEs “belong” thereto. The computation-loaded SPEs as well as unloaded & free SPEs are accounted within the log. The

virtualizations are initiated by either a software or hardware request. An Operating System (OS) is designed to enable multiple independent threads with virtualization. Computer programming languages and the machine code compilers support the virtualization with libraries. The MPE and pseudo MPE also can tune the rest of the virtualized control threads and pseudo virtualized control threads using time-division resource sharing and OS-level preemptive task management to allow for multi-tasking and virtualization. By allowing main and pseudo MPEs, the MPE loading is reduced and becomes more efficient. Therefore the MCP performance vs. power efficiency increases.

As described above in conjunction with Fig. 2, groups of SPEs are assigned to the virtualized control threads, either in MPE or pseudo MPE. The MPE or pseudo MPE maintains different virtualized control threads over the SPEs, as shown in the diagram. The virtualized control threads and pseudo virtualized control threads load executable program codes and data to the SPE group, and monitor the progress and control the performance and power consumption of the SPE groups. When the SPE group produces computation results, the virtualized control threads and pseudo virtualized control threads send the results to the MPE, allowing further computations or external input/output (I/O). The number of total virtualization is limited by the MPE and SPE capacity to hold virtualization, and the number of available SPEs. Further requests for virtualization can be denied, or accepted by sharing time-division thread sharing.

The virtualized control threads and pseudo virtualized control threads control the power supply voltage and clock frequency to each SPE group (G_k), so that the active and stand-by currents are optimized for the required computations. Typically, a digital circuit speed increases when the supply voltage is raised, and the clock speed can be increased. Within the allowed supply voltage range, it is adjusted based on the computation requirements. When the loaded computation requires intensive operation, the voltage and clock frequency are raised so that it is completed within the time frame. When the loaded computation is loose and if there are plenty of time for processing, the voltage and the clock frequency are lowered to maximize performance/power ratio. An extreme case is when SPEs are standing by. The supply voltage can be reduced to 0 voltage to put the SPE in sleep mode. Slightly

higher voltages can be used with a trade-off between leakage current and wake up time. It takes more time to wake an element when it is in a deeper sleep mode.

It should be understood that the preferred embodiment could be deployed on one or more computing devices (e.g., servers, clients, etc.) within a computer infrastructure. This is intended to demonstrate, among other things, that the preferred embodiment could be implemented within a network environment (e.g., the Internet, a wide area network (WAN), a local area network (LAN), a virtual private network (VPN), etc.), or on a stand-alone computer system. In the case of the former, communication throughout the network can occur via any combination of various types of communications links. For example, the communication links can comprise addressable connections that may utilize any combination of wired and/or wireless transmission methods. Where communications occur via the Internet, connectivity could be provided by conventional TCP/IP sockets-based protocol, and an Internet service provider could be used to establish connectivity to the Internet. Still yet, the computer infrastructure is intended to demonstrate that some or all of the components of such an implementation could be deployed, managed, serviced, etc. by a service provider who offers to implement, deploy, and/or perform the functions of the preferred embodiment for others.

Where computer hardware is provided, it is understood that any computers utilized will include standard elements such as a processing unit, a memory medium, a bus, and input/output (I/O) interfaces. Further, such computer systems can be in communication with external I/O devices/resources. In general, processing units execute computer program code, such as the software and functionality described above, which is stored within memory medium(s). While executing computer program code, the processing unit can read and/or write data to/from memory, I/O interfaces, etc. The bus provides a communication link between each of the components in a computer. External devices can comprise any device (e.g., keyboard, pointing device, display, etc.) that enable a user to interact with the computer system and/or any devices (e.g., network card, modem, etc.) that enable the computer to communicate with one or more other computing devices.

The hardware used to implement the preferred embodiment can comprise any specific purpose computing article of manufacture comprising hardware and/or computer program code for performing specific functions, any computing article of manufacture that comprises a combination of specific purpose and general purpose hardware/software, or the like. In each case, the program code and hardware can be created using standard programming and engineering techniques, respectively. Moreover, the processing unit therein may comprise a single processing unit, or be distributed across one or more processing units in one or more locations, e.g., on a client and server. Similarly, the memory medium can comprise any combination of various types of data storage and/or transmission media that reside at one or more physical locations. Further, the I/O interfaces can comprise any system for exchanging information with one or more external device. Still further, it is understood that one or more additional components (e.g., system software, math co-processing unit, etc.) can be included in the hardware.

While shown and described herein as delegated virtualization in a multi-core processor, it is understood that the invention further provides various alternative embodiments. For example, in one embodiment, the invention provides a computer-readable/useable medium that includes computer program code to enable a computer infrastructure to provide delegated virtualization in a multi-core processor. To this extent, the computer-readable/useable medium includes program code that implements the process(es) of the invention. It is understood that the terms computer-readable medium or computer useable medium comprises one or more of any type of physical embodiment of the program code. In particular, the computer-readable/useable medium can comprise program code embodied on one or more portable storage articles of manufacture (e.g., a compact disc, a magnetic disk, a tape, etc.), on one or more data storage portions of a computing device (e.g., a fixed disk, a read-only memory, a random access memory, a cache memory, etc.), and/or as a data signal (e.g., a propagated signal) traveling over a network (e.g., during a wired/wireless electronic distribution of the program code).

In another embodiment, the invention provides a method (e.g., business) that performs the process of the invention on a subscription, advertising, and/or fee basis. That is, a service provider, such as a Solution Integrator, could offer to provide delegated virtualization in a

multi-core processor. In this case, the service provider can create, maintain, support, etc., a computer infrastructure, such as computer infrastructure that performs the process of the invention for one or more customers. In return, the service provider can receive payment from the customer(s) under a subscription and/or fee agreement and/or the service provider can receive payment from the sale of advertising content to one or more third parties.

In still another embodiment, the invention provides a processing method. In this case, a computer infrastructure can be provided and one or more systems for performing the process of the invention can be obtained (e.g., created, purchased, used, modified, etc.) and deployed to the computer infrastructure. To this extent, the deployment of a system can comprise one or more of: (1) installing program code on a computing device from a computer-readable medium; (2) adding one or more computing devices to the computer infrastructure; and (3) incorporating and/or modifying one or more existing systems of the computer infrastructure to enable the computer infrastructure to perform the process of the invention.

As used herein, it is understood that the terms “program code” and “computer program code” are synonymous and mean any expression, in any language, code or notation, of a set of instructions intended to cause a computing device having an information processing capability to perform a particular function either directly or after either or both of the following: (a) conversion to another language, code or notation; and/or (b) reproduction in a different material form. To this extent, program code can be embodied as one or more of: an application/software program, component software/a library of functions, an operating system, a basic I/O system/driver for a particular computing and/or I/O device, and the like.

A data processing system suitable for storing and/or executing program code can be provided hereunder and can include at least one processor communicatively coupled, directly or indirectly, to memory element(s) through a system bus. The memory elements can include, but are not limited to, local memory employed during actual execution of the program code, bulk storage, and cache memories that provide temporary storage of at least some program code in order to reduce the number of times code must be retrieved from bulk storage during execution. Input/output or I/O devices (including, but not limited to, keyboards, displays, pointing devices, etc.) can be coupled to the system either directly or through intervening I/O controllers.

Network adapters also may be coupled to the system to enable the data processing system to become coupled to other data processing systems, remote printers, storage devices, and/or the like, through any combination of intervening private or public networks. Illustrative network adapters include, but are not limited to, modems, cable modems and Ethernet cards.

The foregoing description of various aspects of the invention has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed, and obviously, many modifications and variations are possible.

CLAIMS

1. A multi-core processor, comprising:
a main processing element comprising a set of virtualized control threads;
a first group of sub-processing elements associated with main processing element via at least one set of virtualized control threads; and
a group of sub-processing elements designated as a pseudo main processing element, the pseudo main processing element comprising a set of pseudo virtualized control threads.
2. The multi-core processor of claim 1, the pseudo main processing element associated with a second group of sub processing elements via the set of pseudo virtualized control threads.
3. The multi-core processor of claim 1 or claim 2, the first group of sub-processing elements comprising a plurality of sub-processing elements.
4. The multi-core processor of claim 1, claim 2 or claim 3, the set of virtualized control threads being configured to send program code and data to the first group of sub-processing elements.
5. The multi-core processor of claim 4, the set of virtualized control threads being further configured to collect computation results from the first group of sub-processing elements.
6. The multi-core processor of any preceding claim, the set of virtualized control threads controlling a clock speed, power consumption and computation loading of the set of first group of sub-processing elements.
7. The multi-core processor of any preceding claim, the set of virtualized control threads being configured to send and receive power control requests between the main processing element and the first group of sub-processing elements.

8. The multi-core processor of any preceding claim, the pseudo main processing element associated with a second group of sub processing elements via the set of pseudo virtualized control threads.
9. A processing method, comprising:
 - associating a main processing element with a first group of sub-processing element using a set of virtualized control threads;
 - delegating a set of functions of the main processing element to a second group of sub-processing elements to cause the second group of sub-processing elements to act as a pseudo main processing element;
 - associating the pseudo main processing element with a third group of sub-processing elements using a set of pseudo virtualized control threads; and
 - controlling the first group of sub-processing elements using the set of virtualized control threads and the third group of sub-processing elements using the set of pseudo virtualized control threads.
10. The processing method of claim 9, further comprising sending program code and data to the first group of sub-processing elements via the set of virtualized control threads and to the third set of groups of sub-processing elements via the set of pseudo virtualized control threads.
11. The processing method of claim 9 or claim 10, further comprising receiving computation results from the first group of sub-processing elements via the set of virtualized control threads and from the third group of sub-processing elements via the set of pseudo virtualized control threads.
12. The processing method of any of claims 9 to 11, the controlling comprising controlling a clock speed, a power consumption and a computation loading of the first group of sub-processing elements and the third group of sub-processing elements.
13. The processing method of any of claims 9 to 12, the set of virtualized control threads being embodied as program code within the set of main processing elements, and the set of

pseudo virtualized control threads being embodied as program code within the pseudo main processing element. .

14. The processing method of any of claims 9 to 13, the main processing element, the first group of sub-processing elements, the set of virtualized control threads, the pseudo main processing element, the third group of sub-processing elements, and the set of pseudo virtualized control threads comprising a multi-core processor.

15. A method for deploying a processing system, comprising:
providing a multi-core processor comprising:
a main processing element comprising a set of virtualized control threads;
a first group of sub-processing elements associated with main processing element via at least one of set of virtualized control threads; and
a group of sub-processing elements designated as a pseudo main processing element, the pseudo main element comprising a set of pseudo virtualized control threads.

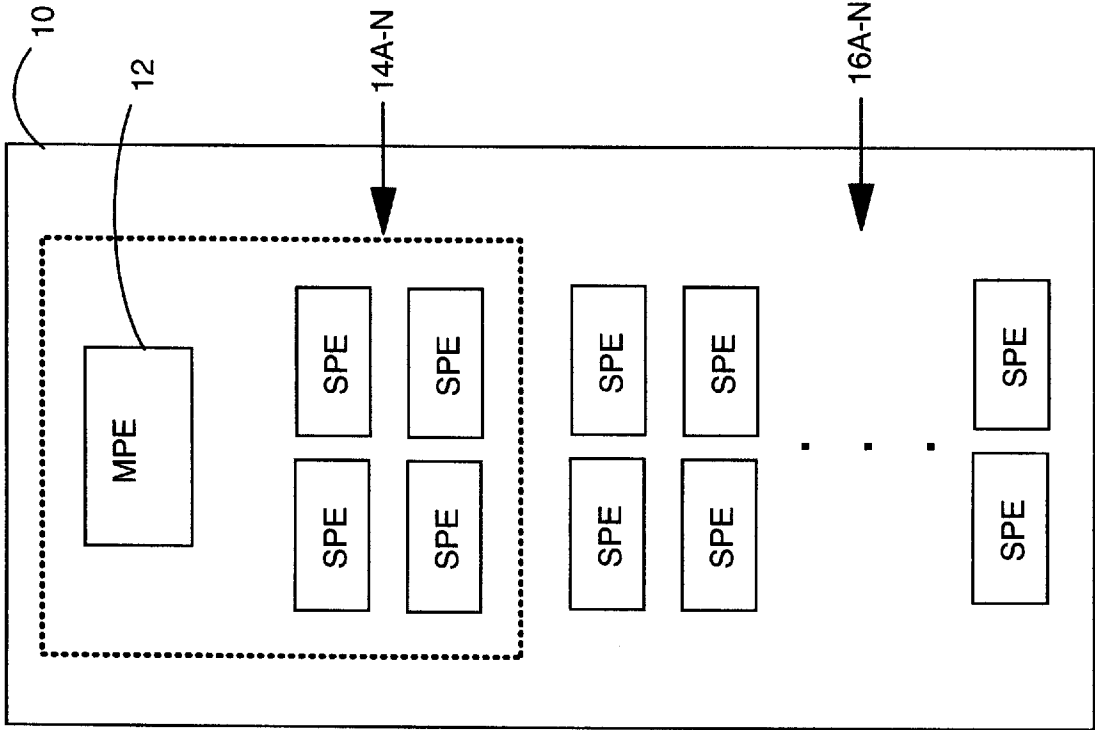


FIG. 1

2/2

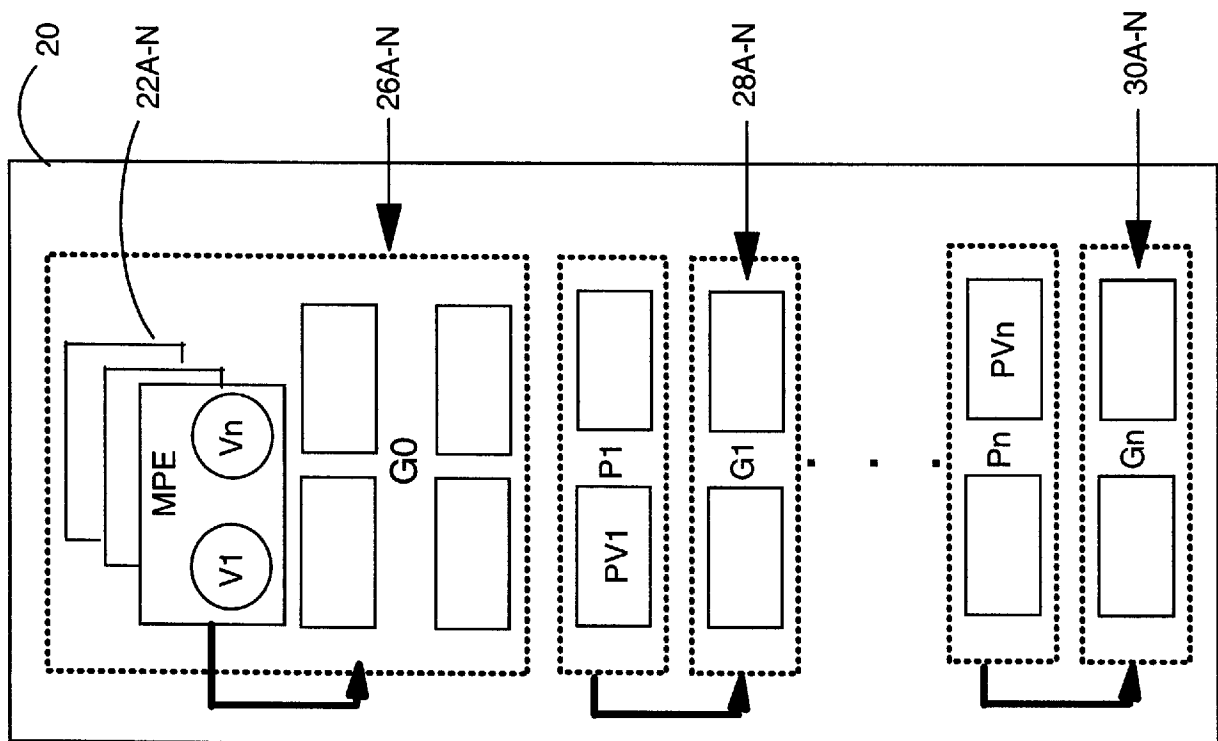


FIG. 2

INTERNATIONAL SEARCH REPORT

International application No

PCT/EP2009/062554

A. CLASSIFICATION OF SUBJECT MATTER

INV. G06F12/08 G06F13/364 G06F15/163

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 practical, search terms used)

EP0-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2002/138637 A1 (SUZUOKI MASAKAZU [JP] ET AL SUZUOKI MASAKAZU [JP] ET AL) 26 September 2002 (2002-09-26)	1-5, 8-11, 13-15
Y	abstract column 13 - column 14 column 65 - column 72	6-7, 12
X	US 2007/074207 A1 (BATES JOHN P [US] ET AL) 29 March 2007 (2007-03-29)	1-5, 8-11, 13-15
Y	abstract paragraph [0007] - paragraph [0016]	6-7, 12



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 document but published on or after the international filing date
- *L* document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
- *O* document referring to an oral disclosure, use, exhibition or other means
- *P* document published prior to the international filing date but later than the priority date claimed

- *T* later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
- *X* document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
- *Y* document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.
- *&* document member of the same patent family

Date of the actual completion of the international search

30 November 2009

Date of mailing of the international search report

08/12/2009

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Skomorowski, Markus

INTERNATIONAL SEARCH REPORT

International application No

PCT/EP2009/062554

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JOSE FLICH ET AL: "On the Potential of NoC Virtualization for Multicore Chips" COMPLEX, INTELLIGENT AND SOFTWARE INTENSIVE SYSTEMS, 2008. CISIS 2008. INTERNATIONAL CONFERENCE ON, IEEE, PISCATAWAY, NJ, USA, 4 March 2008 (2008-03-04), pages 801-807, XP031312240 ISBN: 978-0-7695-3109-0 abstract page 801, left-hand column, line 1 - page 802, right-hand column, line 47 -----	6-7,12

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/EP2009/062554

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2002138637 A1	26-09-2002	CN 1494690 A	05-05-2004
		EP 1370968 A1	17-12-2003
		WO 02077845 A1	03-10-2002
		JP 2002366533 A	20-12-2002
		JP 2003271570 A	26-09-2003
		KR 20080108588 A	15-12-2008
		TW 266200 B	11-11-2006
		US 2008256275 A1	16-10-2008
		US 2005138325 A1	23-06-2005
		US 2005160097 A1	21-07-2005
		US 2008250414 A1	09-10-2008
		US 2005268048 A1	01-12-2005
		US 2007186077 A1	09-08-2007
		US 2005081181 A1	14-04-2005
		US 2007168538 A1	19-07-2007
US 2007074207 A1	29-03-2007	EP 1934739 A1	25-06-2008
		JP 2009510613 T	12-03-2009
		WO 2007038457 A1	05-04-2007