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

(19) **World Intellectual Property Organization**
International Bureau

(43) **International Publication Date**
**5 July 2012 (05.07.2012)**

WIPO | PCT

(10) **International Publication Number**
**WO 2012/089564 A1**

(51) **International Patent Classification**:
*G06F 11/34* (2006.01) *G06F 9/50* (2006.01)

(21) **International Application Number**:
PCT/EP2011/073458

(22) **International Filing Date**:
20 December 2011 (20.12.2011)

(25) **Filing Language**: English

(26) **Publication Language**: English

(30) **Priority Data**:
3173/DEL/2010 30 December 2010 (30.12.2010) IN
61/442,928 15 February 2011 (15.02.2011) US

(71) **Applicant** *(for all designated States except US)*: **ST-ERICSSON SA** [CH/CH]; Chemin du Champ-des-Filles 39, CH-1228 Plan-les-Ouates (CH).

(72) **Inventor; and**
(75) **Inventor/Applicant** *(for US only)*: **PRAKASH, Shyam** [IN/IN]; B2/215, 2nd Floor, Purvanchal Silver City 2, Tower 15, Greater Noida, Uttar Pradesh IN-201308 (IN).

(74) **Agent**: **STRÖM & GULLIKSSON AB (LUND)**; Box 793, S-220 07 Lund (SE).

(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, 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, QA, RO, RS, RU, RW, 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,

*[Continued on next page]*

# (54) **Title**: LOAD DETERMINATION METHOD

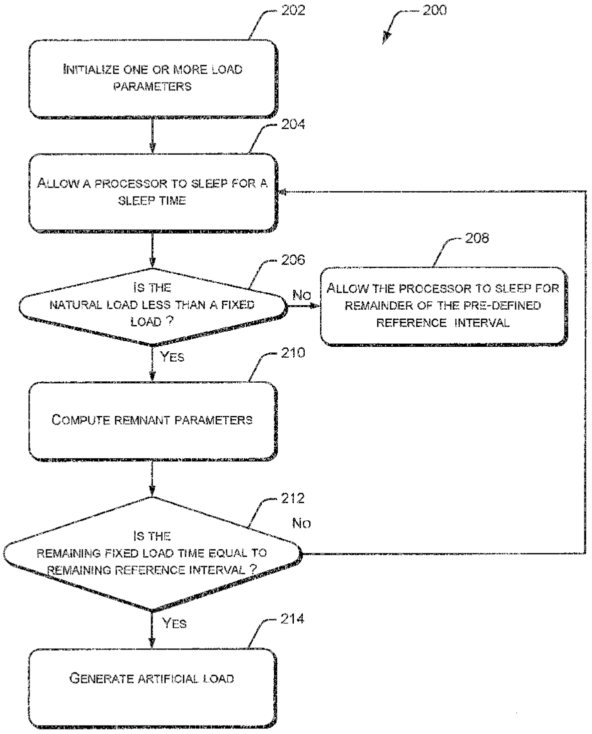
200
202
INITIALIZE ONE OR MORE LOAD PARAMETERS
204
ALLOW A PROCESSOR TO SLEEP FOR A SLEEP TIME
206
IS THE NATURAL LOAD LESS THAN A FIXED LOAD ?
NO
208
ALLOW THE PROCESSOR TO SLEEP FOR REMAINDER OF THE PRE-DEFINED REFERENCE INTERVAL
YES
210
COMPUTE REMNANT PARAMETERS
212
NO
IS THE REMAINING FIXED LOAD TIME EQUAL TO REMAINING REFERENCE INTERVAL ?
YES
214
GENERATE ARTIFICIAL LOAD

Fig. 2

(57) **Abstract**: Methods and devices for determining load on a computing device (100) are described. The method includes scheduling at least one idle thread for a sleep time, determining a natural load at a lapse of the sleep time, and modifying at least one device parameter based at least on the determination.

WO 2012/089564 A1

GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, 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, ML, MR, NE, SN, TD, TG).

**Published**:

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

# LOAD DETERMINATION METHOD

## TECHNICAL FIELD

[0001] The present subject matter relates, in general, to a method and a device for determination of load and, in particular, to determination of load on a computing device.

## BACKGROUND

[0002] Generally, the load on a computing device, such as a desktop computer, a portable computer, and a mobile phone, may be measured in terms of various performance parameters. Examples of the performance parameters include central processing unit (CPU) utilization, memory utilization, input/output capacity, network load, or a combination thereof. The measurement of load may be useful in various applications. In one scenario, the load of a computing device measured, say in terms of CPU load, can be used to alter voltage and frequency of a clock signal provided to the CPU by using techniques, such as dynamic voltage and frequency scaling.

[0003] In another scenario, the measured load may be useful in load testing of the computing device, specifically in generation of an artificial load. Load testing is the process of creating demand or artificial load on the computing device and subsequently, measuring the device's response to the total load. The total load may include both the artificial load and a default natural load of the computing device. Such testing further involves monitoring the performance of the computing device under various load conditions, identifying potential scalability problems, determining the maximum capacity of the computing device, and identifying bottlenecks, if any, that might interfere with the computing device operating at that capacity.

[0004] However, conventional schemes of load measurement operate while the CPU is in execution. In other words, the measurement is performed during a time interval within which one or more processes are scheduled for execution. In effect, the result of such measurements may be inaccurate. Since the measured load is provided as input to various applications, for example, load-testing applications, such applications also fail to provide correct results.

## SUMMARY

**[0005]** This summary is provided to introduce concepts related to a method and a device for load determination. The concepts are further described below in the detailed description. This summary is not intended to identify essential features of the claimed subject matter nor is it intended for use in determining or limiting the scope of the claimed subject matter.

**[0006]** In a first aspect, the invention concerns the method for determining a natural load on a computing device. The method includes scheduling at least one idle thread for a sleep time and determining the natural load at a lapse of the sleep time. The determining includes scheduling a *loadgen* thread to determine the natural load. The priority of the *loadgen* thread is higher than a priority of the idle thread and lower than a priority of a worker thread. The method further includes modifying a device parameter based at least on the determination. The device parameter is at least one of an operating voltage of a computing device, a frequency of a clock signal provided to the computing device, and an artificial load provided to the computing device, based at least on the natural load. In an implementation, the method also includes initializing one or more load parameters. The load parameters include at least one of a pre-defined reference interval, a fixed load, and a priority of the *loadgen* thread.

**[0007]** The determining further includes comparing the natural load and a fixed load after lapse of the sleep time, computing at least one remnant load parameter if the natural load is less than the fixed load, and based on the computing, determining whether a remaining fixed load time is equal to a remaining reference interval. If the remaining fixed load time is equal to the remaining reference interval, the method includes generating an artificial load based on the determining. However, if the remaining fixed load time is less than remaining reference interval, then the method includes re-scheduling the idle thread for a re-computed value of the sleep time, where the re-computed value of the sleep time is based at least on a remaining portion of the fixed load.

**[0008]** In a second aspect, the invention concerns a computing device configured to determine natural load on say, a processor or a device bus. The device includes a processor and a memory coupled to the processor. The memory includes a load determination module configured to schedule at least one idle thread for the sleep time and determine the natural load at a lapse of the sleep time. The memory also includes a modification module configured to modify a device parameter based on the natural load. The computing device is one of a mobile phone, a desktop computer, a portable computer, a hand-held device, a smart phone, an a load-testing device.

**[0009]** In a third aspect, the invention concerns a computer readable medium method having embodied thereon a computer program for executing a method for determining the natural load on a computing device. The method includes initializing a pre-defined reference interval and determining the natural load at any instant within the pre-defined reference interval by scheduling a *loadgen* thread. The priority of the *loadgen* thread is higher than a priority of an idle thread and lower than a priority of a worker thread. The method further includes comparing the natural load and a fixed load, computing at least one remnant load parameter if the natural load is less than the fixed load and based on the computing, determining whether a remaining fixed load time is equal to a remaining reference interval. If the remaining fixed load time is equal to the remaining reference interval, the method includes generating an artificial load on conformance with remaining portion of the fixed load.

## Brief Description of The Drawings

**[00010]** The detailed description is provided with reference to the accompanying figures. In the figures, the left-most digit(s) of a reference number identifies the figure in which the reference number first appears. The same numbers are used throughout the drawings to reference like features and components. For simplicity and clarity of illustration, elements in the figures are not necessarily to scale. Some embodiments of devices and/or methods in accordance with embodiments of the present subject matter are now described, by way of example only, and with reference to the accompanying figures, in which:

**[00011]** Figure 1(a) illustrates an exemplary computing device configured to compute natural load, according to an embodiment of the present subject matter.

**[00012]** Figure 1(b) illustrates the generation of fixed load in a pre-defined reference interval, according to an embodiment of the present subject matter.

**[00013]** Figure 2 illustrates an exemplary method for determining natural load on a computing device, according to an embodiment of the present subject matter.

**[00014]** Figure 3 illustrates a graph representing constant fixed load over a period of time, according to an embodiment of the present subject matter.

## DETAILED DESCRIPTION

**[00015]** With the advancements in computer technology, computing devices, for example, personal computers and laptops, have become more compact and portable. Improved processing speed of the central processing units (CPU) located within the computing devices and a decreased or optimum power consumption of such devices is a desired attribute. Such management of power consumption may be dependent on the measurement of the load on the CPU as in certain schemes an operating voltage and/or frequency of the clock signal provided to the CPU can be varied based on the load. If the load is not measured correctly, the voltage and the frequency could be incorrectly scaled. This, in turn, translates into excessive power consumption and reduced carbon footprints.

**[00016]** Another application in which the measurement of the load may be useful is in load testing, for example, for generation of an artificial load while load testing. Generally, load testing involves modeling an expected usage of the computing device by simulating real-world conditions. For example, the load testing involves simulating the actions of relatively large number of users connected to the computing device by subjecting the computing device with a fixed load and evaluating its performance. The fixed load is typically defined for a predetermined time interval or a series of time intervals. The fixed load includes both the load generated naturally by the computing device, also referred to as natural load, and artificial load that is generated by a load-testing device, in case the natural load is not equal to the fixed load.

**[00017]** The natural load should be accurately determined as the artificial load is generated such that the fixed load requirement is met within a pre-determined time interval. Further, in conventional schemes, the artificial load is generated immediately after the natural load is computed. However, if any interrupts are received after the artificial load is generated, the fixed load requirement is violated as the interrupts contribute towards the natural load in the pre-defined time interval. In other words, due to unscheduled interrupts, the computation of the natural load is inaccurate and a higher amount of the artificial load is unnecessarily generated. Further, one or more threads generating the artificial load are typically scheduled with normal priority and therefore, consume considerable resources of the computing device. Since such schemes do not ensure fixed load, troughs in load generation are observed.

**[00018]** Typically, to determine the natural load for applications, such as load testing or dynamic voltage frequency scaling, the natural load is determined during the execution of device processes. The device processes may be implemented in the form of threads, for example, worker threads. In an example, the natural load may be computed after the execution of a first worker thread and during the execution of a second worker thread. In such a scenario, however, the load determination may not take into account the load generated by the second worker thread, thereby rendering the result of the load determination inaccurate.

**[00019]** To this end, devices and methods for determining load on one or more computing based components, for example processor, are described. Although the description may have been explained with reference to processor load, the description can be extended to other loads, such as device bus load, as well. In an implementation, the exemplary method includes initializing one or more load parameters, for example, pre-defined reference interval, fixed load, and prioritization of threads, such as the worker, idle and *loadgen* threads. The pre-defined reference interval is a time interval; say 100 milliseconds, within which the load, such as natural load, is determined. In case of load testing, the pre-defined reference interval may also be the time interval within which the device's response is determined corresponding to a fixed load.

**[00020]** Subsequent to initialization, the natural load is determined. In one implementation, the natural load can be determined through execution of a *loadgen* thread. A *loadgen* thread can be considered to be any process, which when executed, would result in the determination of the natural load at the time when the *loadgen* thread is executed. In one implementation, the *loadgen* thread is scheduled before an idle thread for determining the natural load. Such scheduling ensures that the natural load is not computed between the execution of the worker threads, thereby ensuring that the natural load determination is accurate.

**[00021]** In one embodiment, the determination of the natural load at a pre-determined instant, for example, any instant within the pre-defined reference interval, may be used as an input for varying the frequency of the clock signal or operating voltage being provided to the processor, while the computing device is in operation. Based on the value of the natural load, the operating point, which can be a function of voltage and frequency, may be varied. For example, if the natural load is 100%, the processor may be configured to work on 600 MHz, whereas if the natural load is 50%, the processor may be configured to work on 300MHz, thereby reducing the power consumed. Thus, power consumption can be reduced, as well as device performance can be significantly improved as and when required.

**[00022]** As was indicated previously, load testing may also be based on determination of the natural load. Based on the natural load, the load testing schemes generate the artificial load, which together with the natural load is useful in testing the performance of the computing device at a desired fixed load. Generation of an incorrect artificial load can, in turn, affect the load testing of the computing device. Therefore, an accurate natural load determination would consequently ensure better load testing.

**[00023]** In one implementation, once the load parameters, such as fixed load, are initialized, the processor is allowed to sleep for a sleep time. For example, an idle thread may be scheduled for a sleep time. In one implementation, the sleep time may be determined based on the fixed load. During the sleep time, a scheduler may have scheduled one or more worker threads. Such worker threads together contribute towards the natural load. At a lapse of the sleep time, a *loadgen* thread is scheduled. After the *loadgen* thread has determined the natural load at the end of the sleep time, an inquiry is made to check whether the natural load is less than the fixed load. If the answer to this inquiry yields a "yes", the sleep time and one or more remnant load parameters are computed based at least on the natural load. Examples of remnant load parameters include, but are not limited to, remaining fixed load time, remaining reference interval, and remaining portion of the fixed load.

**[00024]** For sake of explanation only, the remaining portion of the fixed load can be computed from the difference between the fixed load and the natural load. Similarly, the remaining fixed load time is the time for which the remaining portion of the fixed load is to be generated. Based on the remnant parameters, it is further determined whether the remaining fixed load time is equal to the remaining reference interval. If the remaining fixed load time is equal to the remaining reference interval, an artificial load, which is in conformance with the remaining portion of the fixed load, is generated. During the artificial load generation, a worker thread may get re-scheduled. In such an event, the priority is handed over to the worker thread, followed by re-computation of the sleep time and the remnant load parameters.

**[00025]** However, if the remaining fixed load time is less than the remaining reference interval, the sleep time is computed based on the remaining portion of the fixed load. Again, at the lapse of the sleep time, the *loadgen* thread is computed. The process continues until either the required portion of the fixed load is generated naturally or the remaining fixed load time is equal to the remaining reference interval, at which point, the desired artificial load is generated as explained hereinabove.

**[00026]** The operation can be better understood with the help of the following example. It will be appreciated that the example is not limiting to the scope of the present subject matter. Consider a load testing scenario in which at the start of a reference interval of say 100 milliseconds (ms), a fixed load of 40% is to be assumed. This implies that the fixed load consumes 40% of the pre-defined reference interval. In such a case, the processor is made to sleep for the first 60 ms assuming that if the processor does not generate the natural load of 40%, the processor will have to artificially generate load in the last 40 ms. At the lapse of 60 ms, the natural load is determined via the *loadgen* thread. As mentioned before, the natural load may be a result of the contribution by one or more worker threads. In an example, assuming that the *loadgen* thread computes the natural load as 30% during the sleep time, the remaining fixed load time and remaining portion of the fixed load are computed at the lapse of the sleep time. In the current example, the remaining fixed load time is 10 ms and the remaining portion of the fixed load is 10%. Therefore, the processor is allowed to sleep for another 30 ms. Again, during the 30 ms of the sleep time, the processor may receive one or more worker threads, further adding to the natural load. If at the lapse of the re-computed sleep time, the natural load is computed to be equal to the fixed load, i.e., 40%, the processor sleeps for the remainder of the pre-defined reference interval as no artificial load needs to be generated. However, if the natural load is not sufficient at the lapse of the re-computed sleep time, the processor has no other choice but to generate the remaining portion of fixed load artificially by itself.

**[00027]** As can be clearly seen, the present subject matter provides for accurately determining the natural load, which in turn, can be further used for generating the artificial load when it is determined that the processor can no longer generate load naturally based on an accurate determination of natural load, say during load testing. In other words, the processor is tested with least minimum artificial load. As a result, responses of the resources meeting the requirements of fixed load are better observed during load testing purposes. Additionally, the *loadgen* thread, which is used for determining the natural load at a pre-determined instant, may be configured to run in the background and therefore, need not be scheduled. Further, since the *loadgen* thread is a low priority thread, device resources and power are minimally consumed.

**[00028]** The present subject matter also provides for balancing load between one or more computing based components, for example processors, based on the values of natural load. Devices that can implement the disclosed method include, but are not limited to, desktop computers, hand-held devices, laptops or other portable computers, mobile phones, landline phones, and the like.

**[00029]** Further, it should be noted that the description and figures merely illustrate the principles of the present subject matter. It will thus be appreciated that those skilled in the art will be able to devise various arrangements that, although not explicitly described or shown herein, embody the principles of the present subject matter and are included within its spirit and scope. Moreover, all statements herein reciting principles, aspects, and embodiments of the present subject matter, as well as specific examples thereof, are intended to encompass equivalents thereof.

**[00030]** It will be appreciated by those skilled in the art that the words during, while, and when as used herein are not exact terms that mean an action takes place instantly upon an initiating action but that there may be some small but reasonable delay, such as a propagation delay, between the initial action, and the reaction that is initiated by the initial action. Additionally, the word "connected" is used throughout for clarity of the description and can include either a direct connection or an indirect connection. The descriptions and details of well-known components are omitted for simplicity of the description. Further, it will be understood that the scheduling of a process or thread also indicates the execution of the thread.

**[00031]** Figure 1(a) illustrates an exemplary computing device 100 configured to determine load, for example default or natural load on the computing device 100, according to an embodiment of the present subject matter. The computing device 100 may be a desktop computer, a hand-held device, a laptop or other portable computer, a mobile phone, and a landline phone. Alternatively, the computing device 100 may be a load-testing device connected to device, such as a mobile phone, for load testing of the mobile phone. In an embodiment, the computing device 100 includes a central processing unit, hereinafter referred to as processor 102, and a memory 104. The processor 102 can be a single processing unit or a combination of multiple processing units. The processor 102 may be implemented as one or more microprocessors, microcomputers, microcontrollers, digital signal processors, central processing units, state machines, logic circuitries, and/or any devices that manipulate signals based on operational instructions. Among other capabilities, the processor(s) 102 is configured to fetch and execute computer-readable instructions stored in the memory 104.

**[00032]** The memory 104 may include any computer-readable medium known in the art including, for example, volatile memory such as SRAMs and DRAMs and/or non-volatile memory such as EPROMs and flash memories. The memory 104 includes program module(s) 106 and program data 108. In one implementation, the program module(s) 106 include, for example, a load determination module 110, a load generation module 112, a voltage-frequency module 114, and other module(s) 116. It will be appreciated that each of the program module(s) 106 can be implemented as a combination of one or more different modules. For example, the load generation module 112 and the voltage-frequency module 114 may be included within a single modification module (not shown in the figure). The other module(s) 116 include programs that supplement applications or functions performed by the computing device 100. The program data 108 serves, amongst other things, as repository for storing data pertinent to functioning of the program modules 106.

**[00033]** As per one implementation, the computing device 100 is configured to determine a natural load and accordingly modify one or more device parameters of the computing device 100 through one of the load generation module 112 and the voltage-frequency module 114. Examples of device parameters include, but are not limited to, an artificial load, an operating voltage of the computing device 100, a frequency of a clock signal provided to the computing device 100, etc. The natural load can be indicative of, in one example, the processor load. The processor load can be considered to be the amount or percentage of time the processor 102 is busy in executing one or more device processes. This is further explained with reference to an example illustrated in Fig. 1(b).

**[00034]** According to the present example, a 40% processor load in a pre-defined reference interval of say, 100 milliseconds, indicates that the processor 102 is busy executing device processes for 40 milliseconds (shown in shaded block) and idle or waiting for interrupts for the rest of the 60 milliseconds. Thus, the natural load, i.e., 40%, may be computed using the following relationship (1):

$$Natural\ Load = \left(\frac{Pre\text{−}defined\ Reference\ Interval\text{−}Idle\ Time}{Pre\text{−}defined\ Reference\ Interval}\right) \times 100 \quad (1)$$

**[00035]** The natural load, in another example, can be understood as the default load on a device bus (not shown in the figure), which interconnects sub-components in a system-on-chip.

**[00036]** Referring back to Fig. 1(a), the memory 104 also includes threads 118. In general, when processes are complex and utilize a large amount of computational resources during execution, each complex process is divided into one or more paths of execution, also referred to as threads, such as the threads 118. In one implementation, the threads 118 may include at least one worker thread, at least one idle thread, and a load generation thread (also referred to as *loadgen* thread). The worker threads are associated with device processes, whereas the idle thread when scheduled indicates that the processor 102 is in an idle state. A *loadgen* thread can be considered to be any process, which when executed, would result in the determination of the natural load at the instant when the *loadgen* thread is executed. Furthermore, the worker thread has the highest priority while the idle thread has the lowest priority. In an example, the priority of the *loadgen* thread is just one higher than the priority of the idle thread. In a scenario where an interrupt is generated, the interrupt is given the same priority as a worker thread, and is scheduled before the *loadgen* thread and the idle thread.

**[00037]** The program module(s) 106 and their operations are discussed in detail in the following paragraphs. It will be understood that even though the description may have been explained with reference to processor load, the description can be extended to other device loads as well.

**[00038]** As per one implementation, the load determination module 110 initializes one or more load parameters, for example, pre-defined reference interval, priority of the *loadgen* thread, and fixed load. The pre-defined reference interval can be a time interval, say 100 milliseconds, within which the natural load is determined. In a scenario in which the computing device 100 is a load-testing device, the pre-defined reference interval is also the time interval within which a device's response is determined corresponding to a fixed load. In one example, the fixed load may be user-defined based on the computing device 100 to be tested. The fixed load may include both the natural load and the artificial load.

**[00039]** Once the load parameters are initialized, the load determination module 110 schedules the *loadgen* thread to compute a natural load. The natural load is defined as the load that is naturally created by default applications executing on the computing device 100. For example, applications of a mobile phone such as receiving calls, receiving software updates, etc., may contribute towards its natural load. As mentioned earlier, the *loadgen* thread is scheduled just before an idle thread, thus ensuring that the natural load is not computed between the execution of worker threads. This, in turn, ensures that the measurement of natural load is accurate since the completion of all the worker threads is kept into consideration.

**[00040]** In one implementation, the measurement of the natural load at any instant may be used as an input to a voltage-frequency module 114. In response to the values of the natural load, the operating point, which is a function of voltage and frequency, may be dynamically varied. It will be understood that the processor 102 may operate at various operating points based on the voltage and frequency of the clock signal provided to the processor 102. In an example, if the natural load is 100%, the processor may be configured to work on 600 MHz, whereas if the natural load is 50%, the processor may be configured to work on 300 MHz, thereby reducing the power consumed. Accordingly, power consumption can be reduced, as well as device performance can be significantly improved based on requirements.

**[00041]** In an alternative or additional implementation, the load determination module 110 may allow the processor 102 to sleep for a sleep time by, for example, scheduling at least one idle thread for the sleep time. For determination of the natural load while the computing device 100 is in operation, such sleep time may be pre-configured by a user or by an administrator. However, for determination of the natural load on a computing device 100 connected to a load-testing device, the sleep time may be determined based on the fixed load. For example, consider a load testing scenario in which at the start of a pre-defined reference interval, say 100 milliseconds (ms), a fixed load of 60% is assumed. In such a case, the processor 102 is configured to sleep for the first 40 ms of the pre-defined reference interval assuming that if the processor 102 does not generate natural load, the processor 102 will have to artificially generate an artificial load of 60% in the last 60 ms of the pre-defined reference interval. Thus, the selection of the sleep time ensures that if the processor 102 does not generate natural load during the sleep time, the load generation module 112 has enough time in the remainder of the pre-defined reference interval to generate load artificially.

**[00042]** In said implementation, during the sleep time, there may be a case in which a scheduler (not shown in the figure) within the computing device 100, may have scheduled one or more worker threads, requesting device resource allocation as a consequence of device interrupts or otherwise. Such worker threads would contribute towards the natural load.

**[00043]** Once the sleep time lapses, the load determination module 112 schedules a *loadgen* thread to determine the natural load. After the *loadgen* thread determines the natural load at the end of the sleep time, the load determination module 110 compares the natural load with the fixed load. If, on comparison, it is determined that the natural load is equal to the fixed load, the load determination module 110 does not generate an artificial load and allows the processor 102 to sleep for the remainder of the pre-defined reference interval. On the other hand, if it is determined that the natural load is less than the fixed load, the load determination module 110 computes one or more remnant load parameters, for example, remaining fixed load time, remaining reference interval, and remaining portion of the fixed load. For sake of explanation only, the remaining portion of the fixed load may be defined as the difference between the fixed load and the natural load. Similarly, whereas the remaining fixed load time is the time required for which the remaining portion of the fixed load is to be generated. The remaining reference interval can be understood as time of pre-defined reference interval remaining after the processor 102 has slept for the sleep time.

**[00044]** Based on the remnant parameters, the load determination module 110 makes a second comparison to determine whether the remaining fixed load time is equal to the remaining reference interval. If the load determination module 110 determines that the remaining fixed load time is equal to the remaining reference interval, the load generation module 112 generates an artificial load. In an implementation, the artificial load is generated in conformance with the remaining portion of the fixed load. It will be appreciated by a person skilled in the art that during artificial load generation, an interrupt may need to get processed. In such an event, the interrupt acts like a worker thread and preempts the *loadgen* thread based on priority. Thus, the worker thread performs execution after which the sleep time and remnant load parameters are re-computed.

**[00045]** However, if the load determination module 110 determines that the remaining fixed load time is less than the remaining reference interval, the load determination module 110 re-computes sleep time based at least on the remaining portion of the fixed load. The load determination module 110 re-schedules the idle thread for the re-computed value of the sleep time.

**[00046]** Again, during the sleep time, other worker threads may be allowed to get scheduled. After the lapse of the sleep time, the load determination module 110 re-schedules the *loadgen* thread, which re-computes the natural load, as the value may have changed on account of the additional worker threads during the sleep time.

**[00047]** Subsequently, the load determination module 110 determines whether or not the updated natural load is less than the fixed load. If the natural load is equal to the fixed load, the processor 102 is allowed to sleep for rest of the pre-defined reference interval. However, if the updated natural load is less than the fixed load, the load determination module 110 computes one or more remnant load parameters, for example, remaining fixed load time and remaining portion of the fixed load. Based on the comparison of the one or more remnant parameters with each other, the load generation module 112 either generates the desired artificial load or allows the processor 102 to further sleep. In this fashion, the computing device 100 allows for generation of as much natural load as possible by deferring the generation of the artificial load. This process continues till either the fixed load is matched with the natural load only, or as a combination of the natural load and the artificial load.

**[00048]** Figure 2 illustrates an exemplary method 200 for determination of load, such as the natural load, on a computing device, in accordance with an embodiment of the present subject matter. The method 200 is described in the context of processor load, however the method 200 may be extended to cover other kinds of device loads. Additionally, even though the method is described in the context of the computing device 100 as a load-testing device, the method is also implementable on other applications as will be understood by a person skilled in the art. Herein, some embodiments are also intended to cover program storage devices, for example, digital data storage media, which are machine or computer readable and encode machine-executable or computer-executable programs of instructions, wherein said instructions perform some or all of the steps of the described method. The program storage devices may be, for example, digital memories, magnetic storage media such as a magnetic disks and magnetic tapes, hard drives, or optically readable digital data storage media.

**[0001]** The order in which the method is described is not intended to be construed as a limitation, and any number of the described method blocks can be combined in any order to implement the method, or an alternative method. Additionally, individual blocks may be deleted from the method without departing from the spirit and scope of the subject matter described herein. Furthermore, the method can be implemented in any suitable hardware, software, firmware, or combination thereof.

**[00049]** At block 202, one or more load parameters are initialized. In an example, the load parameters, for example, pre-defined reference interval, priority of the *loadgen* thread, and fixed load. In one implementation, a load determination module 110, initializes the load parameters.

**[00050]** At block 204, a processor is allowed to sleep for a sleep time. In an implementation, the sleep time may be determined based at least, in part, on the fixed load. For example, at the start of a pre-defined reference interval, say 100 milliseconds (ms), a fixed load of 60% can be assumed. In such a case, the processor 102 is made to sleep for the first 40 ms assuming that if the processor 102 does not generate natural load, the processor will have to artificially generate load in the last 60 ms.

**[00051]** At block 206, it is determined whether the natural load is less than the fixed load. For example, the load determination module 110 determines the natural load at the lapse of the sleep time. Based on the natural load, a device parameter may be modified. Examples of device parameter include an operating voltage, a frequency of a clock signal, and an artificial load. In an implementation, the load determination module 110 checks whether the artificial load should be generated based on a comparison between the natural load with the fixed load. If the natural load is greater than or equal to the fixed load ("No" branch from block 206), the processor 102 is allowed to sleep for the remainder of the pre-defined reference interval (block 208). In such a case, the idle threads are scheduled for the remainder of the pre-defined reference interval, thereby indicating that the fixed load requirement has been met and no artificial load is to be generated. However, if the natural load is less than the fixed load ("Yes" branch from block 206), one or more remnant parameters are computed at block 210. The remnant parameters include, for example, the remaining portion of the fixed load, the remaining reference interval, and the remaining fixed load time. In one implementation, the load determination module 110 computes the remaining portion of the fixed load as the difference between the fixed load and the natural load. In another implementation, the load determination module 110 further computes a remaining fixed load time, which is the time required for generating the remaining portion of the fixed load. Further, a remaining reference interval, which is the difference between the pre-defined reference interval and the sleep time, is also calculated.

**[00052]** At block 212, it is determined whether the remaining fixed load time is less than the remaining reference interval. In one implementation, the load determination module 110 compares the remaining fixed load time and the remaining reference interval. If it is determines that the remaining fixed load time is equal to the remaining reference interval ("Yes" branch from block 212), the required artificial load is generated (block 214). In one implementation, the load generation module 112 generates the artificial load.

**[00053]** If it is determined that the remaining fixed load time is less than the remaining reference interval ("No" branch from block 212), the processor is allowed to sleep for a recomputed value of sleep time (block 204). During this period, one or more idle threads may be scheduled. In one implementation, the sleep time is computed by the load determination module 110 based on the remaining portion of the fixed load. Additionally, during the sleep time, the processor 102 may receive additional worker threads resulting from device interrupts. Subsequently, the method proceeds to block 206 where the natural load is re-computed after the lapse of the sleep time. The process continues until either the required fixed load is generated naturally or the remaining fixed load time is equal to the remaining reference interval, at which point, the load generation module 112 generates the desired artificial load as described at block 214.

**[00054]** In another implementation, based at least on the natural load, at least one of the voltage and frequency of the clock signal to the processor 102 may be dynamically varied to operate on a higher or lower operating point.

**[00055]** Figure 3 illustrates a graph 300 representing constant fixed load over a period of time. As shown in the graph 300, the fixed load is substantially constant with time. For example, the curve 302-1 shows that for a fixed load of 95% is maintained over time. Similarly, the curves 302-2, 302-3, 302-4, 302-5, 302-6, 302-7, and 302-8 illustrate that a fixed load of 90%, 70%, 50%, 40%, 30%, 20%, and 10%, respectively, is maintained over a pre-defined reference interval. Further, as the pre-defined reference interval increases, the fixed load continues to be almost constant and the accuracy also increases. In an example, the accuracy of the load determination is about 98% to about 100% in case the pre-defined reference interval is in milliseconds. In another example, the accuracy is about 99% to about 100% in case the pre-defined reference interval is in minutes.

**[00056]** Although embodiments for load determination have been described in language specific to structural features and/or methods, it is to be understood that the invention is not necessarily limited to the specific features or methods described. Rather, the specific features and methods are disclosed as exemplary embodiments for the load determination.

I/We claim:

1. A method comprising:

   scheduling at least one idle thread for a sleep time;

   determining a natural load at a lapse of the sleep time; and

   modifying at least one device parameter of a computing device (100) based at least on the determining.

2. The method as claimed in claim 1, wherein the determining comprises scheduling a *loadgen* thread, wherein a priority of the *loadgen* thread is higher than a priority of the idle thread and lower than a priority of a worker thread.

3. The method as claimed in claim 1, wherein the determining comprises:

   comparing the natural load and a fixed load at the lapse of the sleep time;

   computing at least one remnant load parameter if the natural load is less than the fixed load; and

   based on the computing, further determining whether a remaining fixed load time is equal to a remaining reference interval.

4. The method as claimed in claim 3, wherein the modifying comprises generating an artificial load if the remaining fixed load time is equal to the remaining reference interval.

5. The method as claimed in claim 3 further comprising:

   based on the computing, further determining whether the remaining fixed load time is less than the remaining reference interval; and

   if the remaining fixed load time is less than the remaining reference interval, re-scheduling the idle thread for a re-computed value of the sleep time, wherein the re-computed value of the sleep time is based at least on a remaining portion of the fixed load.

6. The method as claimed in claim 1 further comprising initializing one or more load parameters, wherein the load parameters comprise at least one of a pre-defined reference interval, a fixed load, and a priority of a *loadgen* thread.

7. The method as claimed in claim 3, wherein the remnant load parameter is at least one of a remaining fixed load time, a remaining portion of the fixed load, and a remaining reference interval.
8. The method as claimed in claim 1, wherein the device parameter is one of an operating voltage, a frequency of a clock signal, and an artificial load.
9. The method as claimed in claim 1, wherein the natural load comprises at least one of a processor load and a device bus load.
10. A computing device (100) comprising:

    a processor (102); and

    a memory (104) coupled to the processor (102), the memory comprising,

    a load determination module (110) configured to, schedule at least one idle thread for a sleep time; and

    determine a natural load at a lapse of the sleep time; and

    a modification module configured to modify at least one device parameter based at least on the natural load.
11. The computing device (100) as claimed in claim 10 further comprising initializing one or more load parameters, wherein the load parameters comprise at least one of a pre-defined reference interval, a fixed load, and a priority of a *loadgen* thread.
12. The computing device (100) as claimed in claim 11, wherein the load determination module (110) is configured to determine the natural load by scheduling the *loadgen* thread, wherein a priority of the *loadgen* thread is higher than a priority of the idle thread and lower than a priority of a worker thread.
13. The computing device (100) as claimed in claim 10, wherein the modification module further comprises a voltage-frequency module (114) to vary at least one of an operating voltage and a frequency of a clock signal based at least on the natural load.
14. The computing device (100) as claimed in claim 10, wherein the load determination module (110) is further configured to:

    compare the natural load and a fixed load after a lapse of the sleep time;

    if the natural load is less than the fixed load, then compute at least one remnant load parameter; and based on the computation of the at least one remnant load parameter, determine whether a remaining fixed load time is equal to a remaining reference interval.

15. The computing device (100) as claimed in claim 14, wherein the modification module comprises a load generation module (112) to generate an artificial load if the remaining fixed load time is equal to the remaining reference interval.

16. The computing device (100) as claimed in claim 14, wherein the load determination module (110) is further configured to:

determine whether the remaining fixed load time is less than the remaining reference interval; and

re-schedule the idle thread for a re-computed value of the sleep time, wherein the re-computed value of the sleep time is based at least on a remaining portion of the fixed load.

17. The computing device (100) as claimed in claim 10 is one of a mobile phone, a desktop computer, a portable computer, a hand-held device, a smart phone, and a load-testing device.

18. A computer-readable medium having embodied thereon a computer program for executing a method comprising:

initializing a pre-defined reference interval;

determining a natural load on a computing device (100) at a pre-determined instant within the pre-defined reference interval by scheduling a *loadgen* thread, wherein a priority of the *loadgen* thread is higher than a priority of an idle thread and lower than a priority of a worker thread; and

modifying a device parameter of a computing device (100) based at least on the determining.

19. The computer-readable medium as claimed in claim 18, wherein the determining comprises:

    comparing the natural load and a fixed load;

    computing at least one remnant load parameter if the natural load is less than the fixed load; and

    based on the computing, determining whether a remaining fixed load time is equal to a remaining reference interval.

20. The computer-readable medium as claimed in claim 19, wherein the modifying comprises generating an artificial load if the remaining fixed load time is equal to the remaining reference interval.

21. The computer-readable medium as claimed in claim 18 further comprising modifying at least one of an operating voltage of the computing device (100) and frequency of a clock signal provided to the computing device (100), based at least on the natural load.

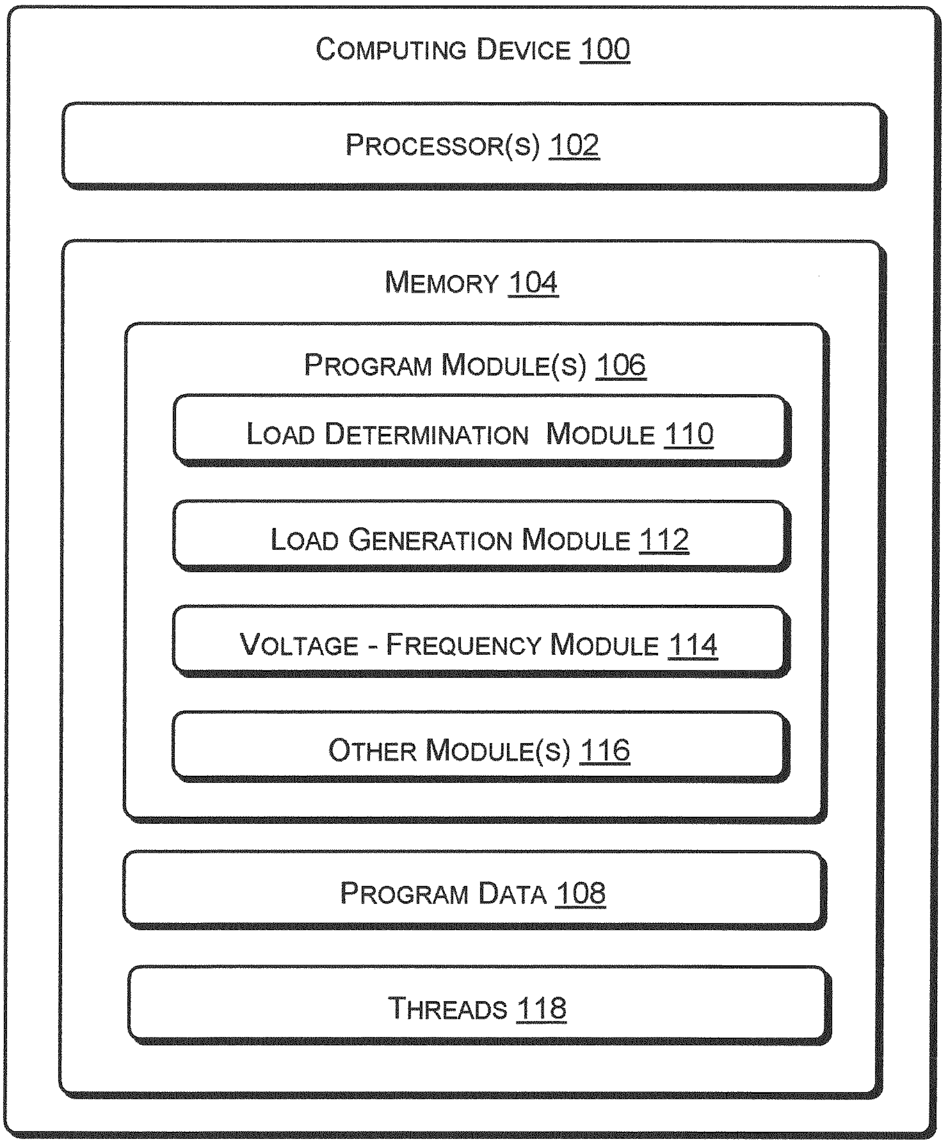
COMPUTING DEVICE 100
PROCESSOR(S) 102
MEMORY 104
PROGRAM MODULE(S) 106
LOAD DETERMINATION MODULE 110
LOAD GENERATION MODULE 112
VOLTAGE - FREQUENCY MODULE 114
OTHER MODULE(S) 116
PROGRAM DATA 108
THREADS 118

Fig. 1(a)

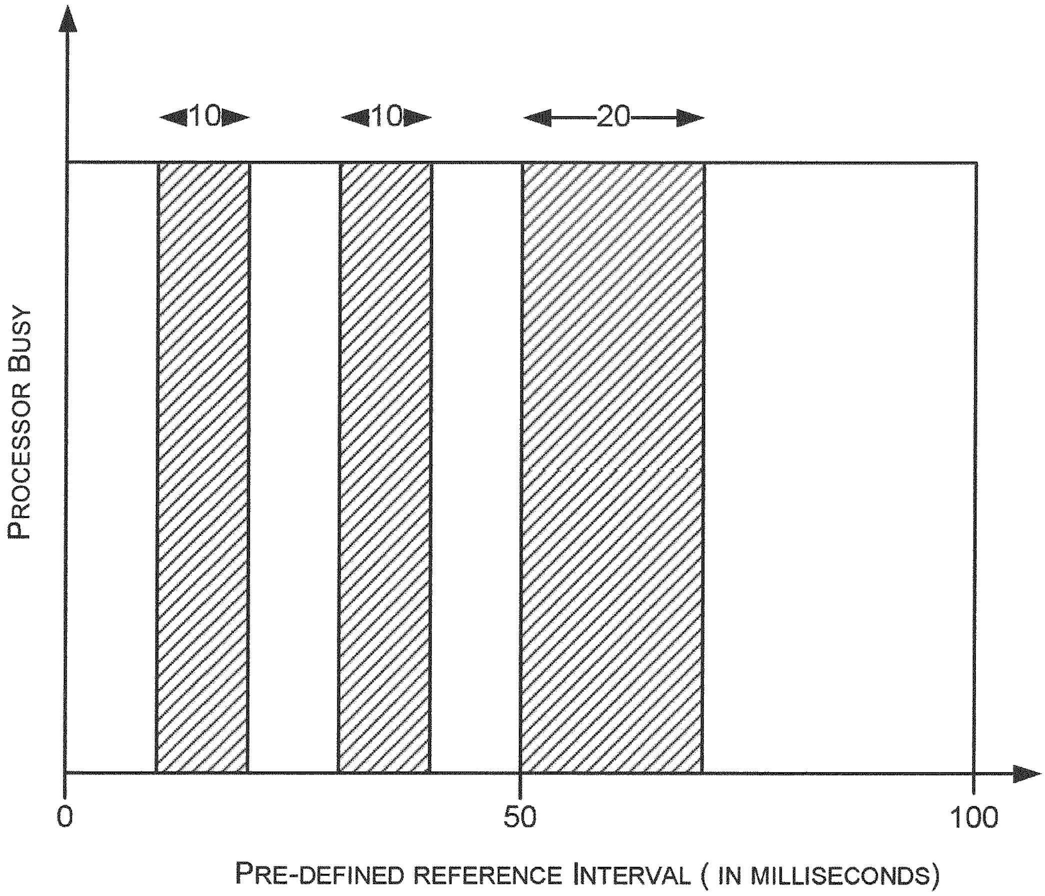
10
10
20
PROCESSOR BUSY
0
50
100
PRE-DEFINED REFERENCE INTERVAL ( IN MILLISECONDS)

Fig. 1(b)

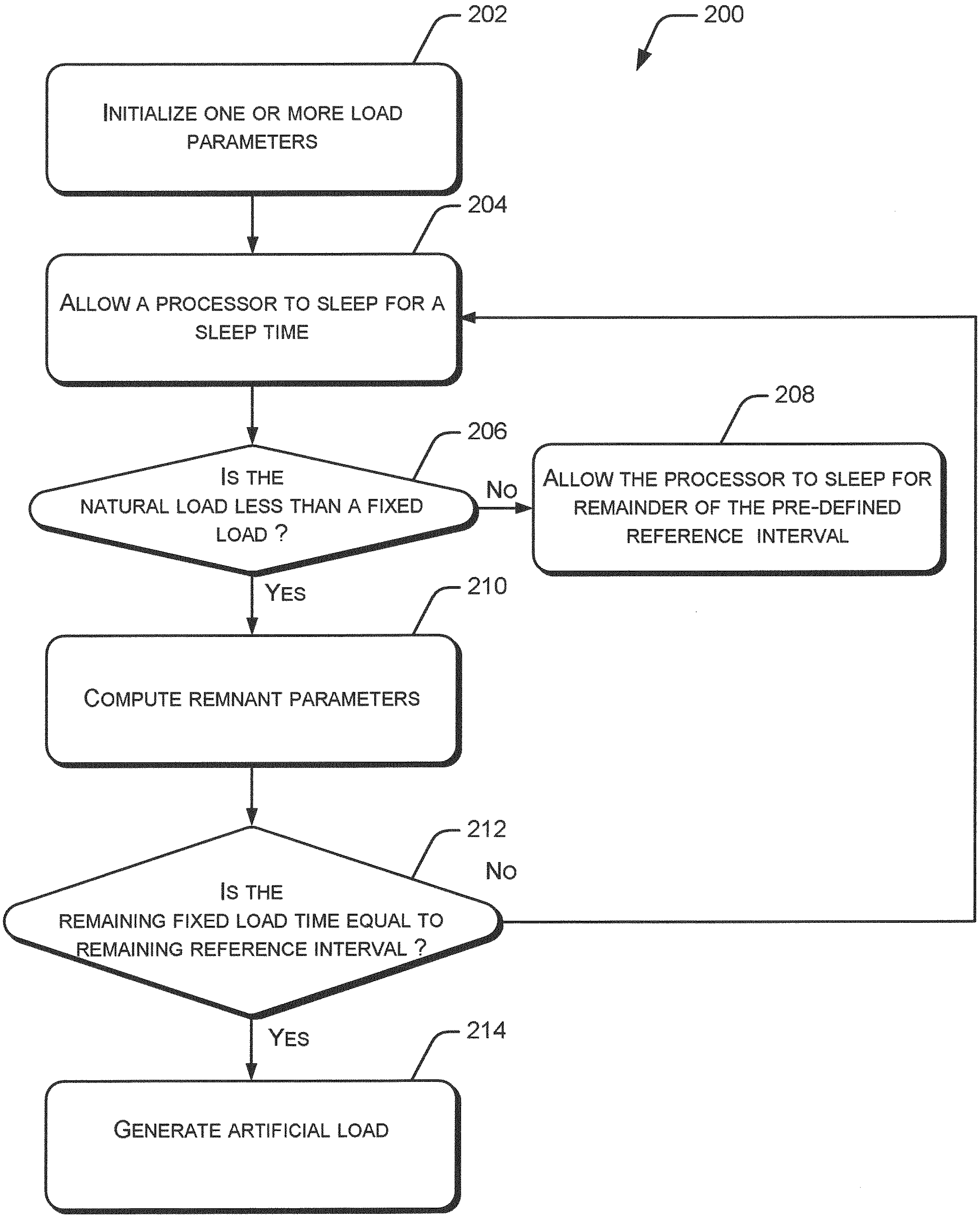
200
202
INITIALIZE ONE OR MORE LOAD PARAMETERS
204
ALLOW A PROCESSOR TO SLEEP FOR A SLEEP TIME
206
IS THE NATURAL LOAD LESS THAN A FIXED LOAD ?
NO
208
ALLOW THE PROCESSOR TO SLEEP FOR REMAINDER OF THE PRE-DEFINED REFERENCE INTERVAL
YES
210
COMPUTE REMNANT PARAMETERS
212
NO
IS THE REMAINING FIXED LOAD TIME EQUAL TO REMAINING REFERENCE INTERVAL ?
YES
214
GENERATE ARTIFICIAL LOAD

Fig. 2

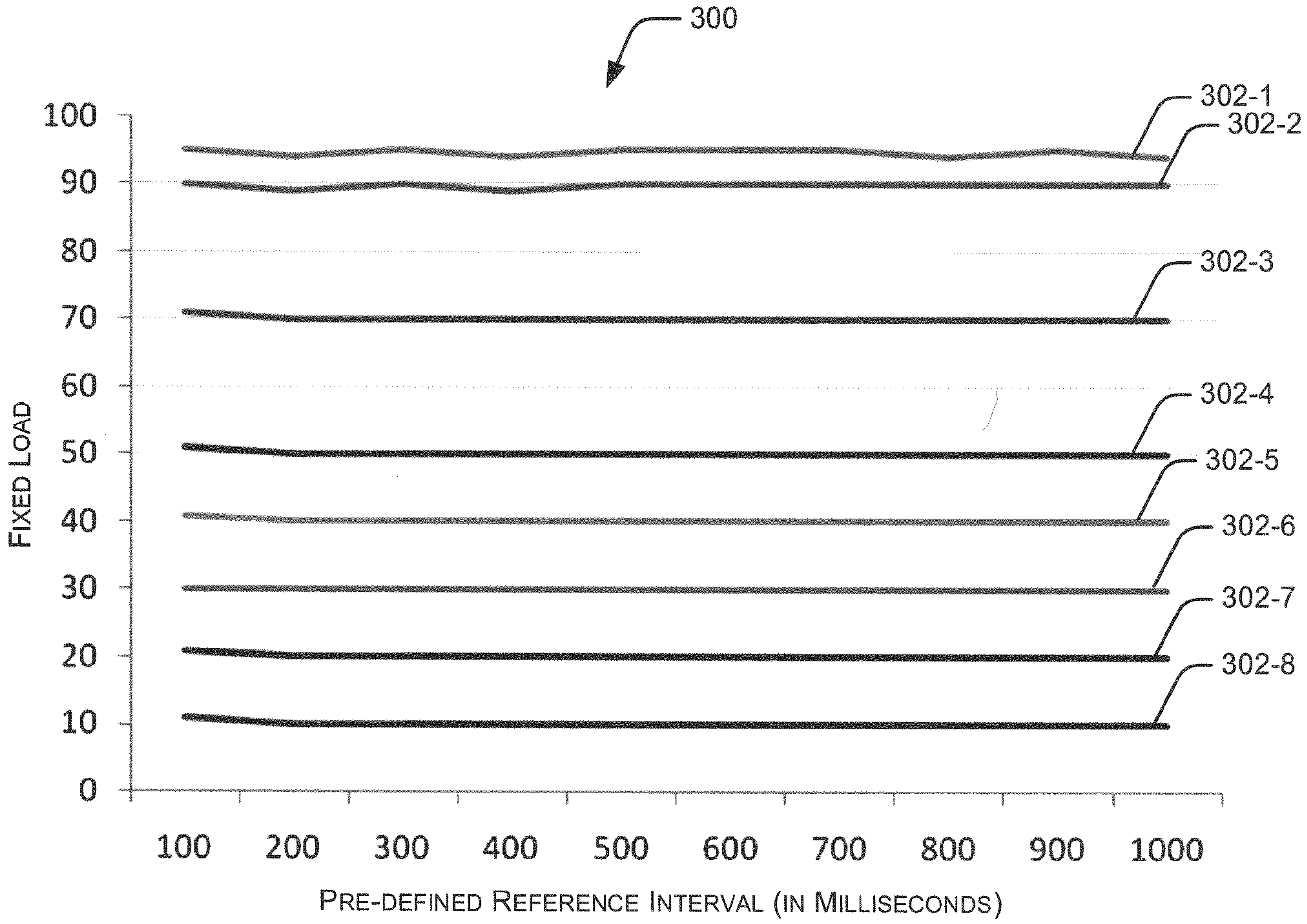
300
302-1
302-2
302-3
302-4
302-5
302-6
302-7
302-8
100
90
80
70
60
50
40
30
20
10
0
FIXED LOAD
100
200
300
400
500
600
700
800
900
1000
PRE-DEFINED REFERENCE INTERVAL (IN MILLISECONDS)

Fig. 3

# INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2011/073458

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F11/34 G06F9/50
ADD.

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)

EPO-Internal, COMPENDEX, INSPEC, IBM-TDB, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| A | US 2009/271152 A1 (BARRETT TIM [AU])<br>29 October 2009 (2009-10-29)<br>paragraphs [0005], [0007], [0010], [0011], [0014], [0015], [0028], [0037], [0038], [0040] - [0045], [0048], [0050]; claims 1-20; figures 3,4<br>----- | 1-21 |
| A | US 2010/274530 A1 (CHEN CHI-HSIEN [TW])<br>28 October 2010 (2010-10-28)<br>paragraphs [0012], [0014], [0020], [0025]; figures 2,3<br>----- | 1-21 |
| A | GB 2 468 291 A (1E LTD [GB])<br>8 September 2010 (2010-09-08)<br>page 1, lines 7-13<br>page 2, line 10 - page 3, line 10<br>page 6, lines 9-14<br>-----<br>-/-- | 1-21 |

☒ 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<br>29 March 2012 | Date of mailing of the international search report<br>04/04/2012 |
| Name and mailing address of the ISA/<br>European Patent Office, P.B. 5818 Patentlaan 2<br>NL - 2280 HV Rijswijk<br>Tel. (+31-70) 340-2040,<br>Fax: (+31-70) 340-3016 | Authorized officer<br>Dixon-Hundertpfund |

3

International application No

PCT/EP2011/073458

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| A | US 6 804 627 B1 (MAROKHOVSKY SERGE G [US] ET AL) 12 October 2004 (2004-10-12)<br>column 2, lines 40-62<br>column 4, lines 1-14<br>column 6, lines 1-21<br>column 9, lines 10-30<br>column 10, lines 52-57; claim 9; figure 5<br>----- | 1-21 |
| A | US 7 509 646 B1 (MAW PATRICIA LYNN [US] ET AL) 24 March 2009 (2009-03-24)<br>column 1, lines 20-25<br>column 5, lines 46-48,54-60<br>column 6, lines 12-15,49-52<br>column 7, lines 1,31-38<br>column 8, line 44 - column 9, line 50<br>----- | 1-21 |

3

# INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/EP2011/073458

| Patent document cited in search report | | Publication date | Patent family member(s) | | | Publication date |
|---|---|---|---|---|---|---|
| US 2009271152 | A1 | 29-10-2009 | NONE | | | |
| US 2010274530 | A1 | 28-10-2010 | CN | 101876933 | A | 03-11-2010 |
| | | | US | 2010274530 | A1 | 28-10-2010 |
| GB 2468291 | A | 08-09-2010 | GB | 2468291 | A | 08-09-2010 |
| | | | US | 2010229014 | A1 | 09-09-2010 |
| US 6804627 | B1 | 12-10-2004 | NONE | | | |
| US 7509646 | B1 | 24-03-2009 | US | 7509646 | B1 | 24-03-2009 |
| | | | US | 2009094613 | A1 | 09-04-2009 |