US 20130046967A1

(19) **United States**

(12) **Patent Application Publication**

FULLERTON et al.

(10) **Pub. No.: US 2013/0046967 A1**

(43) **Pub. Date: Feb. 21, 2013**

(54) **PROACTIVE POWER MANAGEMENT USING A POWER MANAGEMENT UNIT**

(75) Inventors: **Mark FULLERTON**, Austin, TX (US); **John WALLEY**, Ladera Ranch, CA (US); **Hwisung JUNG**, Irvine, CA (US)

(73) Assignee: **Broadcom Corporation**, Irvine, CA (US)

(21) Appl. No.: **13/533,480**

(22) Filed: **Jun. 26, 2012**

Related U.S. Application Data

(60) Provisional application No. 61/524,538, filed on Aug. 17, 2011.

Publication Classification

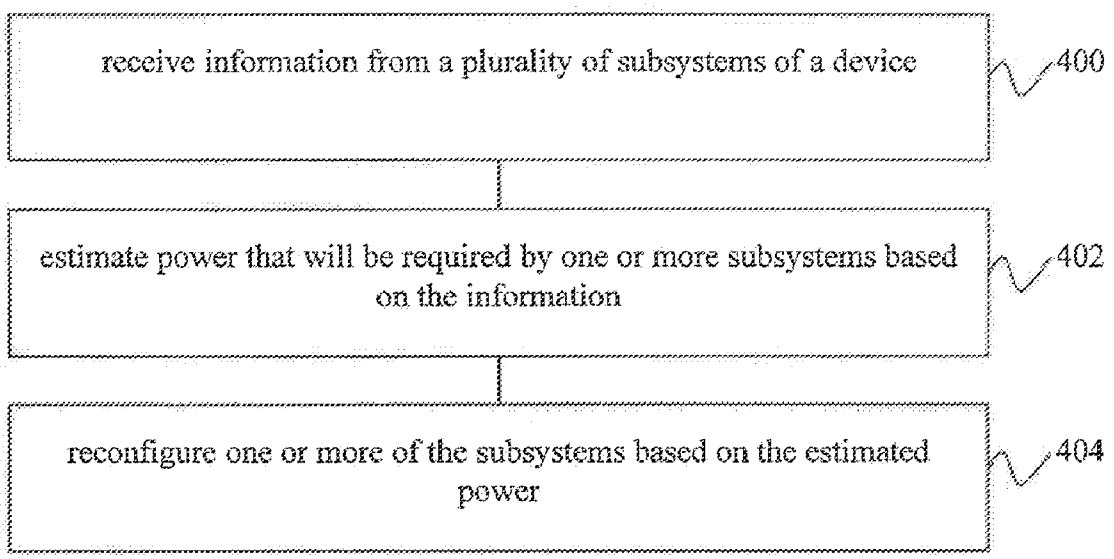
(51) **Int. Cl.**

G06F 1/26	(2006.01)
G06F 1/32	(2006.01)
G06F 9/00	(2006.01)

(52) **U.S. Cl.** **713/100**

(57) **ABSTRACT**

Embodiments of the present disclosure provide systems and methods for proactively managing power in a device. A power management unit (PMU) receives information from various subsystems of a device and estimates the total power required by each subsystem of the device. Based on this information, the PMU can predict power requirements for a particular subsystem or for one or more application(s) to execute. Based on this prediction, the PMU can reconfigure the subsystems so that the device executes more efficiently given the current battery life of the device. Proactive power management advantageously gives the PMU the capability to predict power needs of various subsystems of a device so that the power supplied to these subsystems can be managed in an intelligent way before battery resources are exhausted.



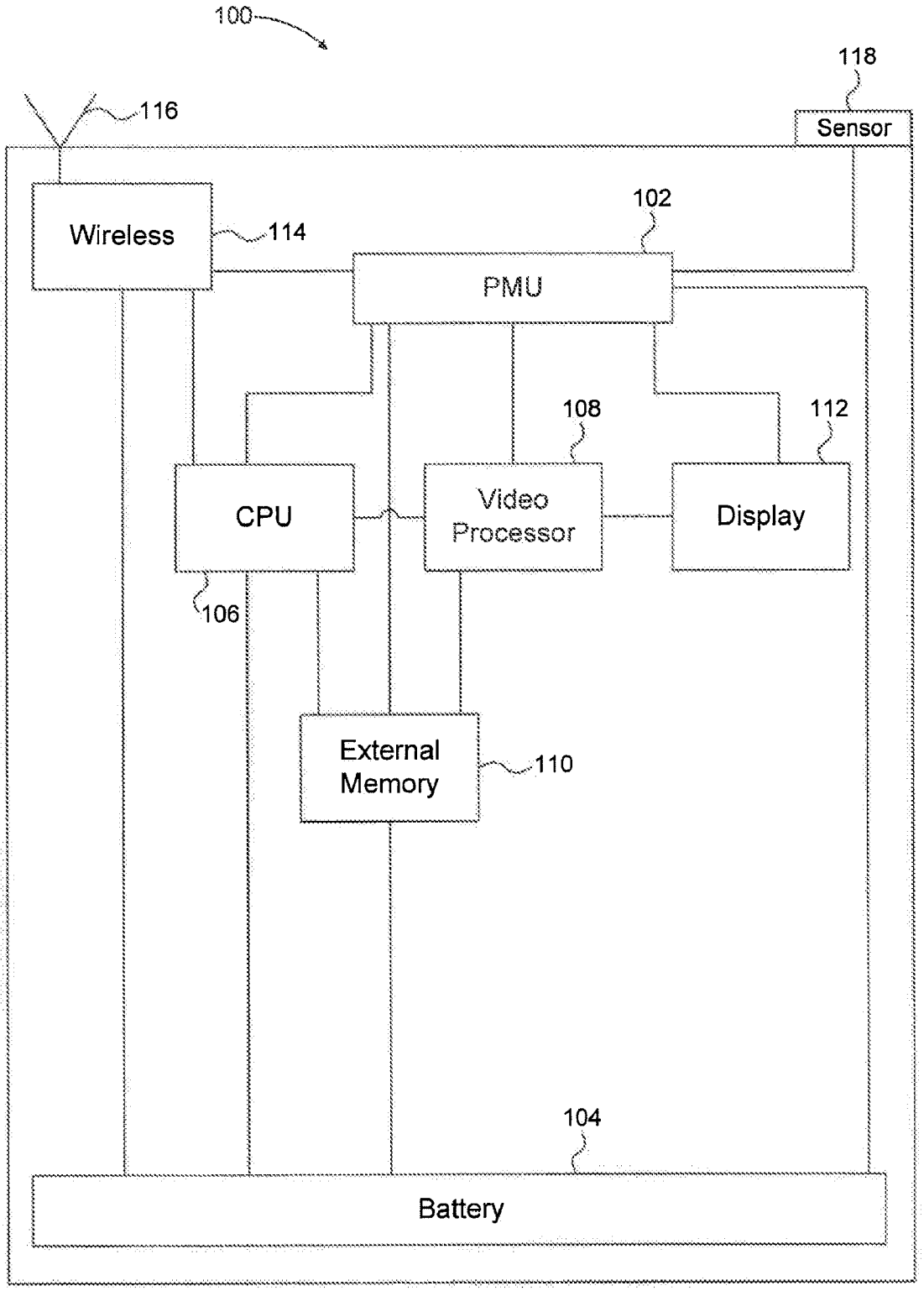

100
118
Sensor
116
Wireless
114
102
PMU
108
112
CPU
Video Processor
Display
106
External Memory
110
104
Battery

FIG. 1

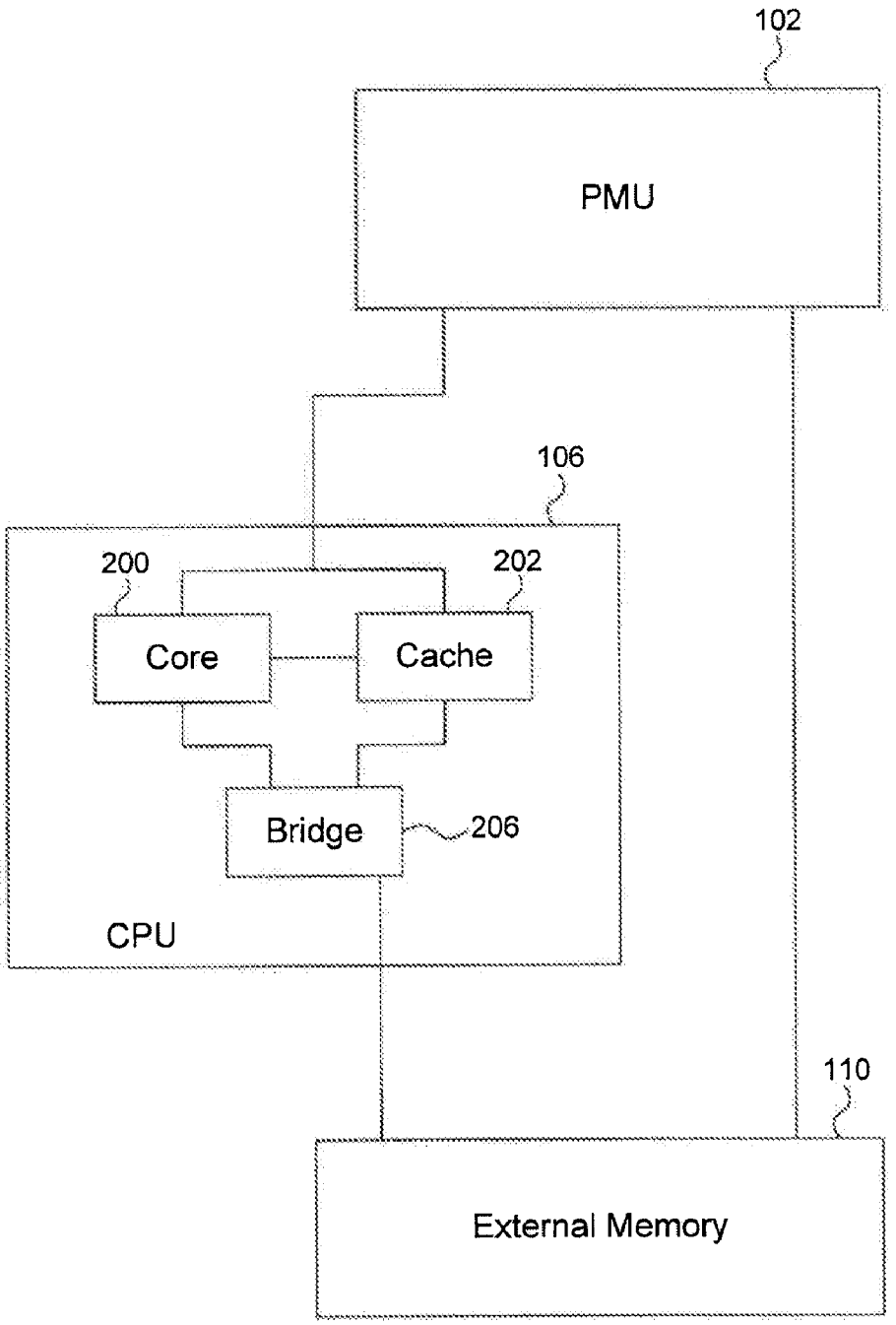

102
PMU
106
200
202
Core
Cache
Bridge
206
CPU
110
External Memory

FIG. 2

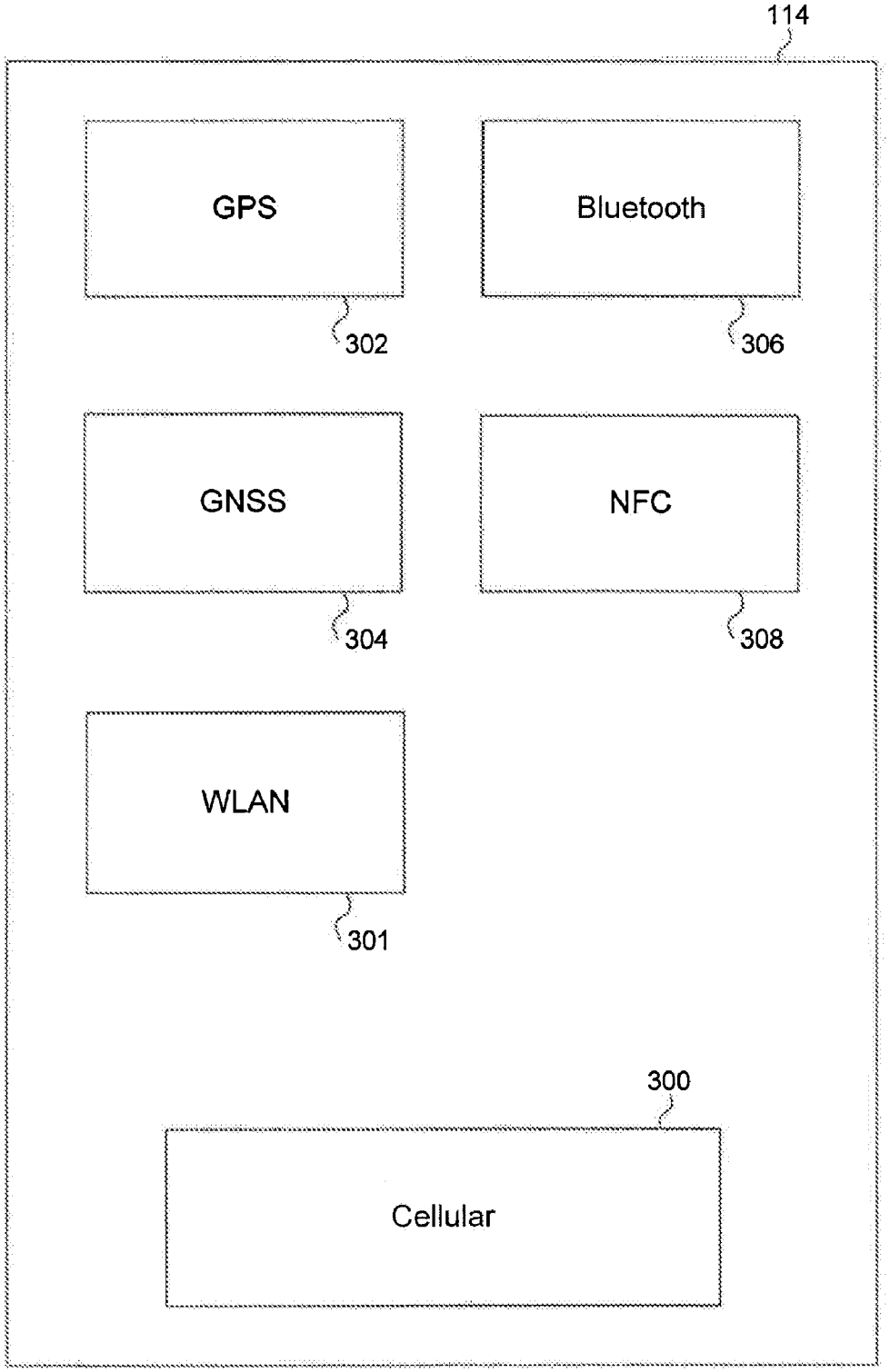

114
GPS
302
Bluetooth
306
GNSS
304
NFC
308
WLAN
301
300
Cellular

FIG. 3

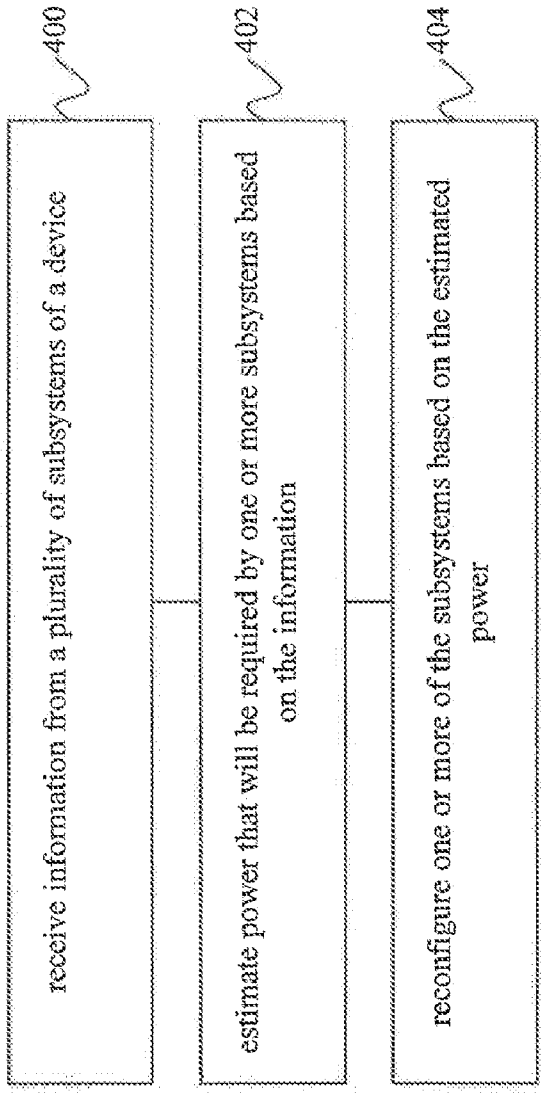

400
receive information from a plurality of subsystems of a device
402
estimate power that will be required by one or more subsystems based on the information
404
reconfigure one or more of the subsystems based on the estimated power

FIG. 4

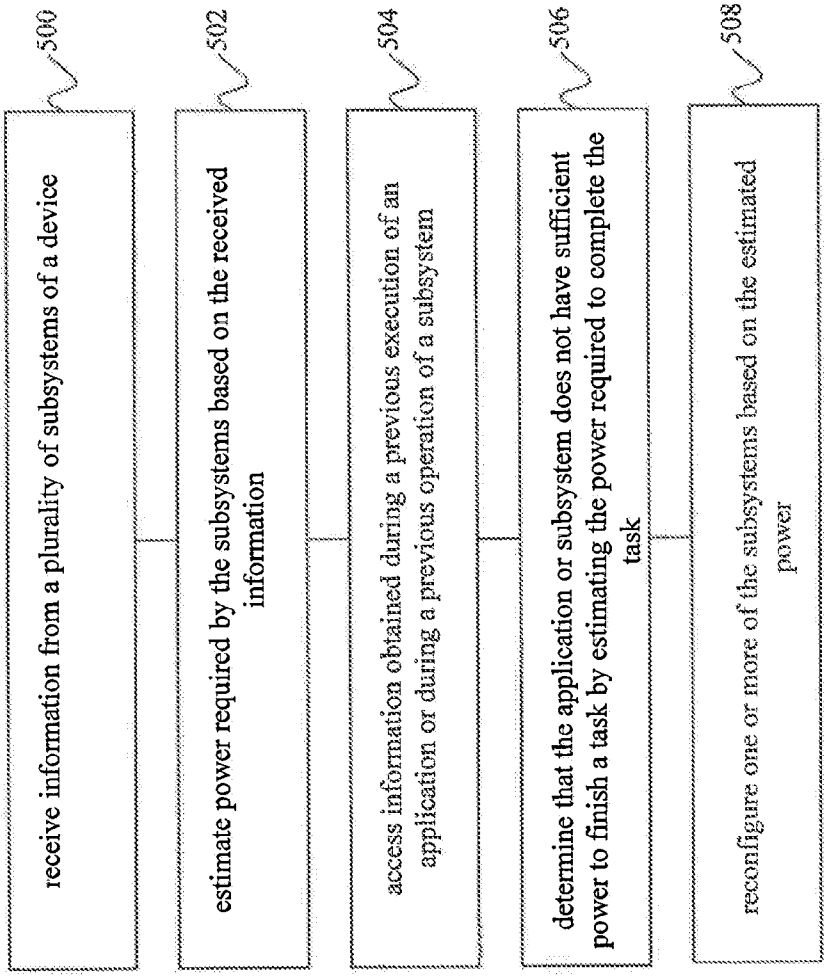

500
receive information from a plurality of subsystems of a device
502
estimate power required by the subsystems based on the received information
504
access information obtained during a previous execution of an application or during a previous operation of a subsystem
506
determine that the application or subsystem does not have sufficient power to finish a task by estimating the power required to complete the task
508
reconfigure one or more of the subsystems based on the estimated power

FIG. 5

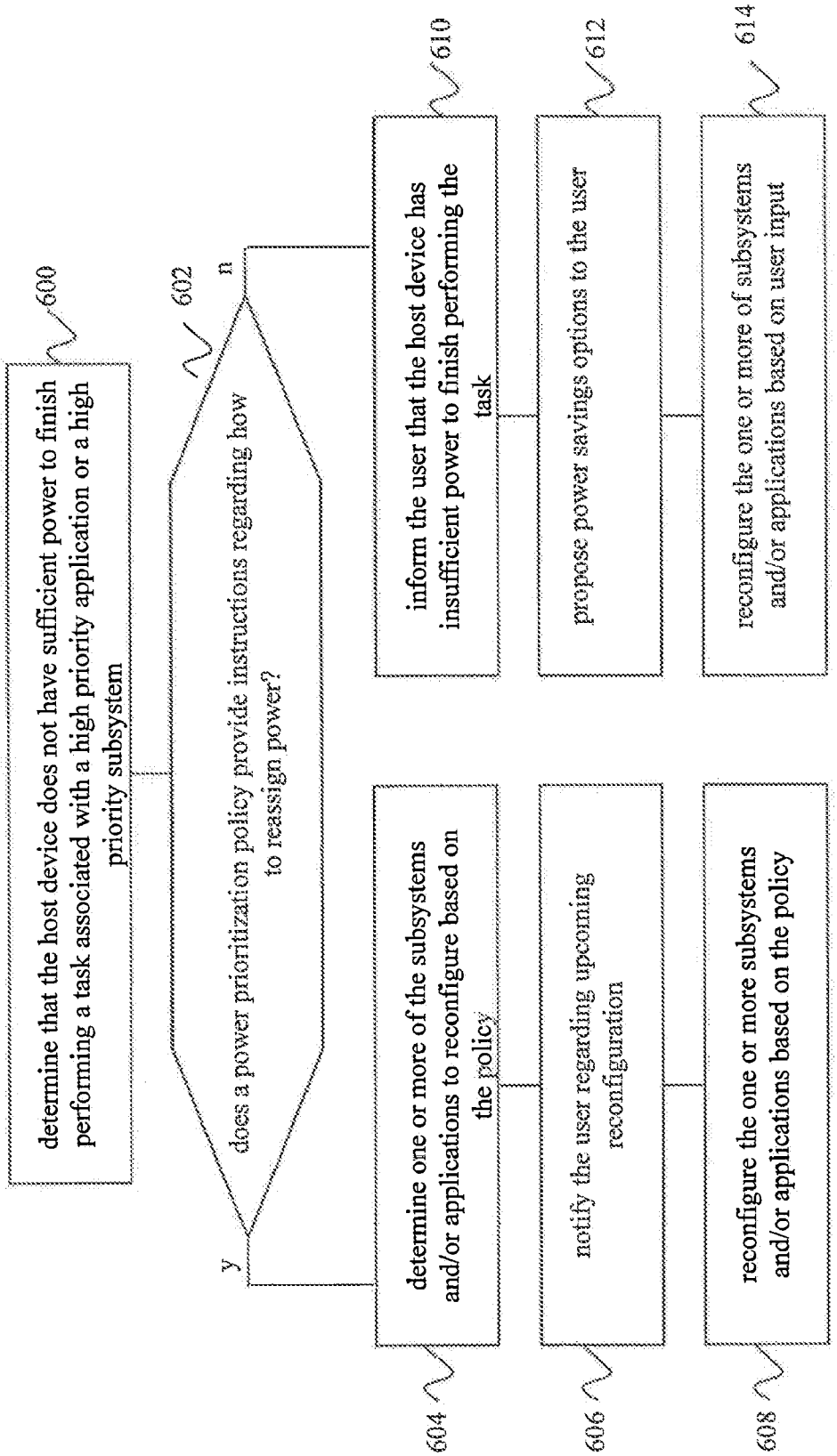

600
determine that the host device does not have sufficient power to finish performing a task associated with a high priority application or a high priority subsystem
602
does a power prioritization policy provide instructions regarding how to reassign power?
n
y
610
inform the user that the host device has insufficient power to finish performing the task
612
propose power savings options to the user
614
reconfigure the one or more of subsystems and/or applications based on user input
604
determine one or more of the subsystems and/or applications to reconfigure based on the policy
606
notify the user regarding upcoming reconfiguration
608
reconfigure the one or more subsystems and/or applications based on the policy

FIG. 6

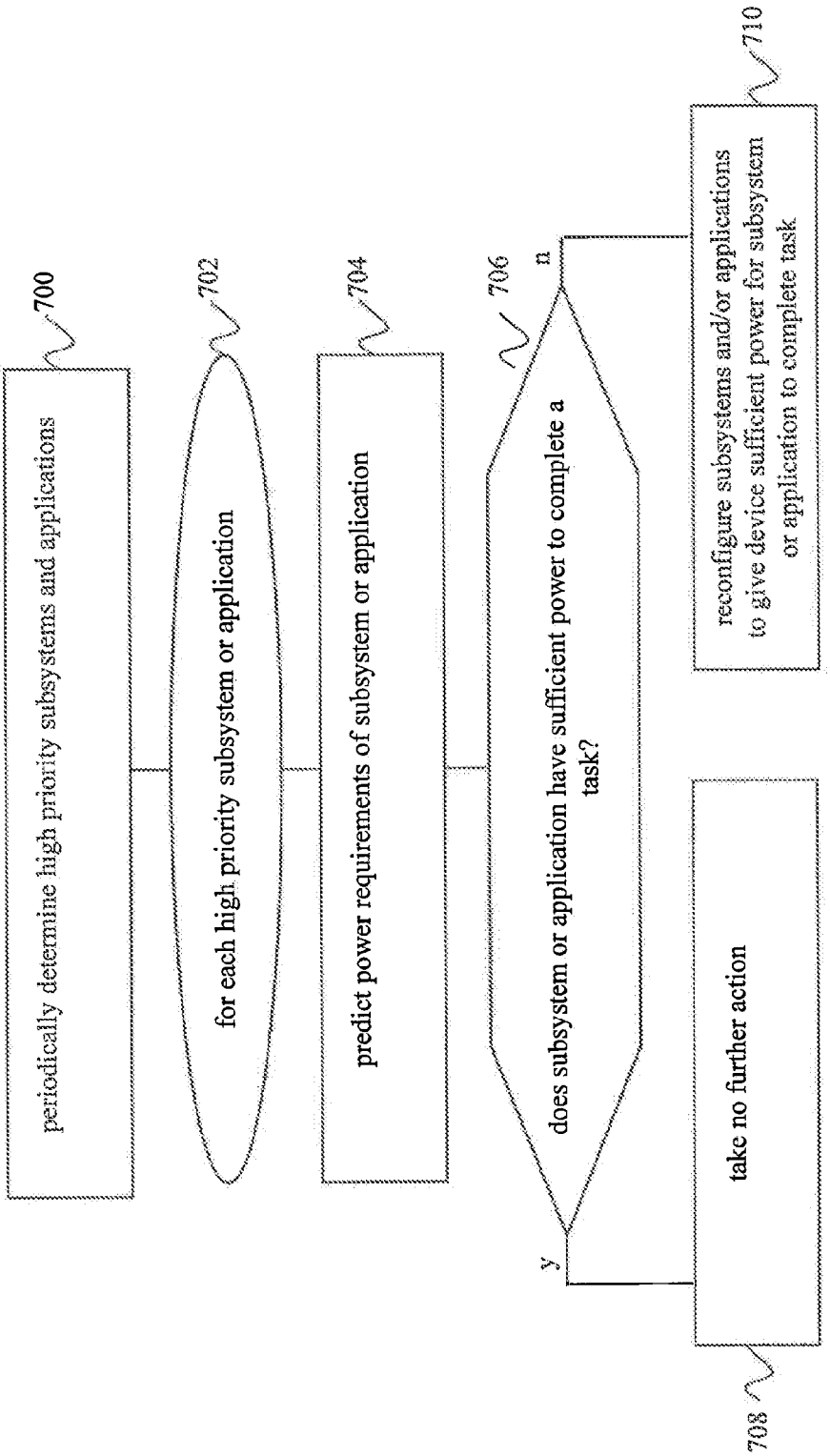

700
periodically determine high priority subsystems and applications
702
for each high priority subsystem or application
704
predict power requirements of subsystem or application
706
does subsystem or application have sufficient power to complete a task?
n
y
710
reconfigure subsystems and/or applications to give device sufficient power for subsystem or application to complete task
708
take no further action

FIG. 7

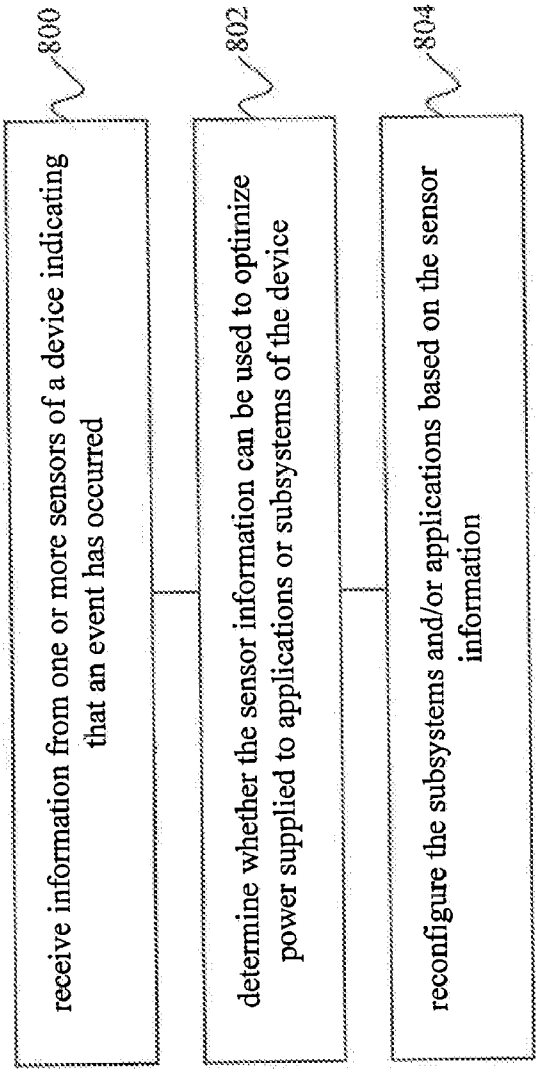

800
receive information from one or more sensors of a device indicating that an event has occurred
802
determine whether the sensor information can be used to optimize power supplied to applications or subsystems of the device
804
reconfigure the subsystems and/or applications based on the sensor information

FIG. 8

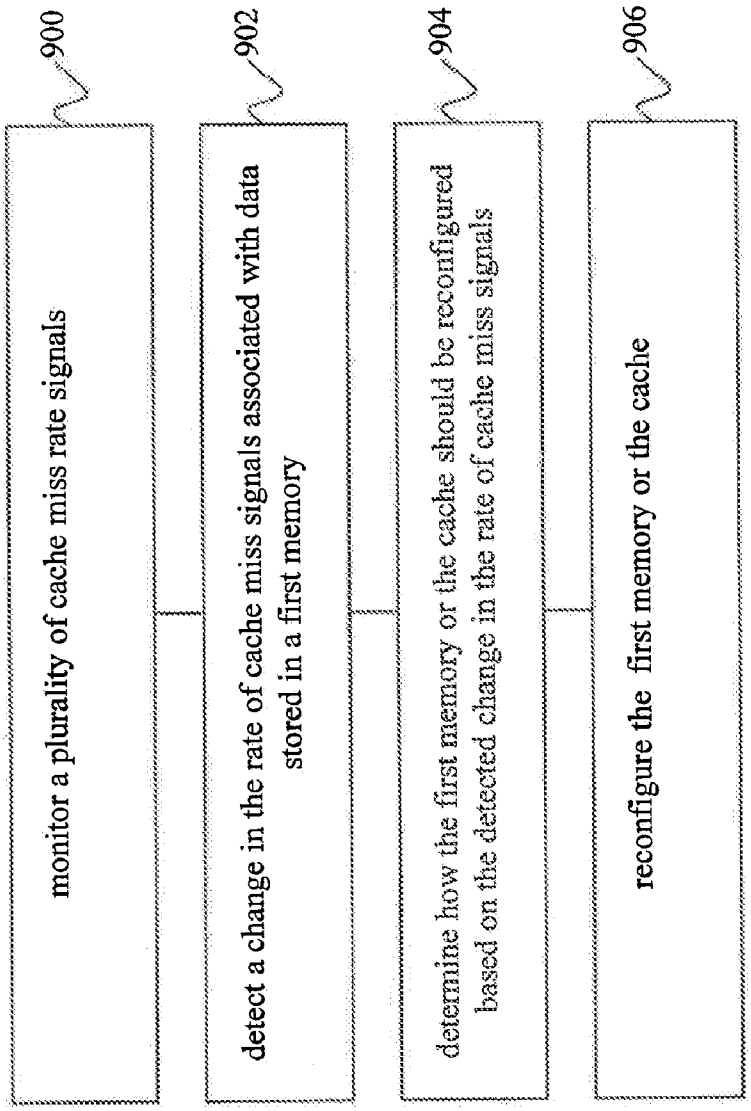

900
monitor a plurality of cache miss rate signals
902
detect a change in the rate of cache miss signals associated with data stored in a first memory
904
determine how the first memory or the cache should be reconfigured based on the detected change in the rate of cache miss signals
906
reconfigure the first memory or the cache

FIG. 9

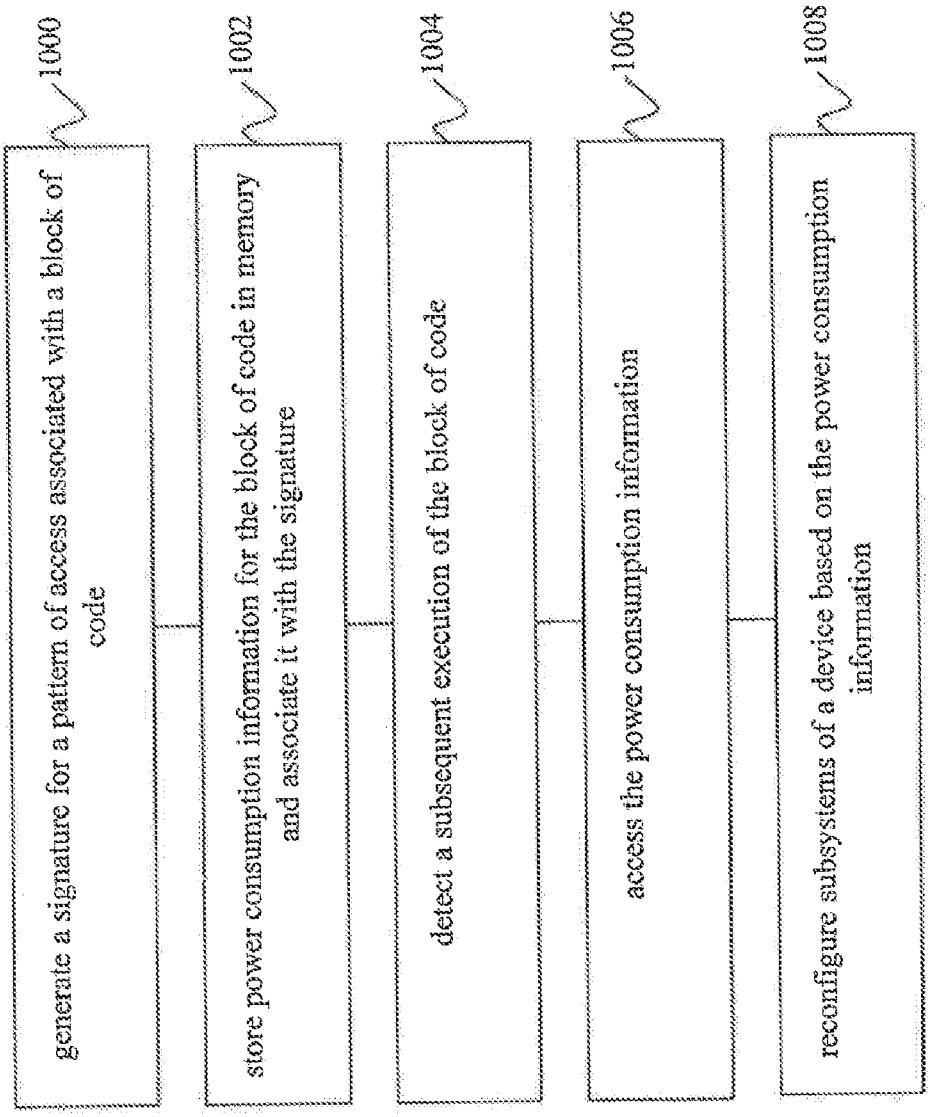

1000
generate a signature for a pattern of access associated with a block of code
1002
store power consumption information for the block of code in memory and associate it with the signature
1004
detect a subsequent execution of the block of code
1006
access the power consumption information
1008
reconfigure subsystems of a device based on the power consumption information

FIG. 10

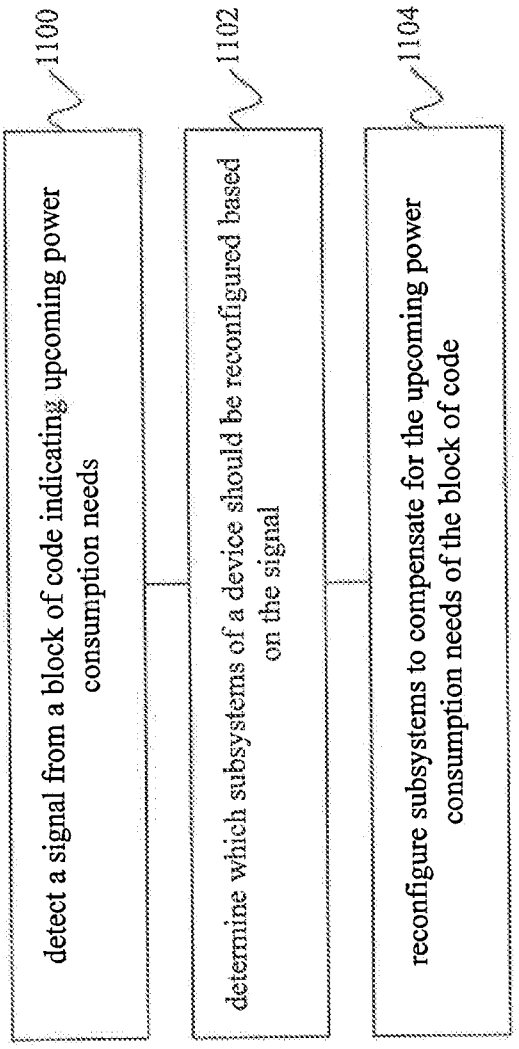

1100
detect a signal from a block of code indicating upcoming power consumption needs
1102
determine which subsystems of a device should be reconfigured based on the signal
1104
reconfigure subsystems to compensate for the upcoming power consumption needs of the block of code

FIG. 11

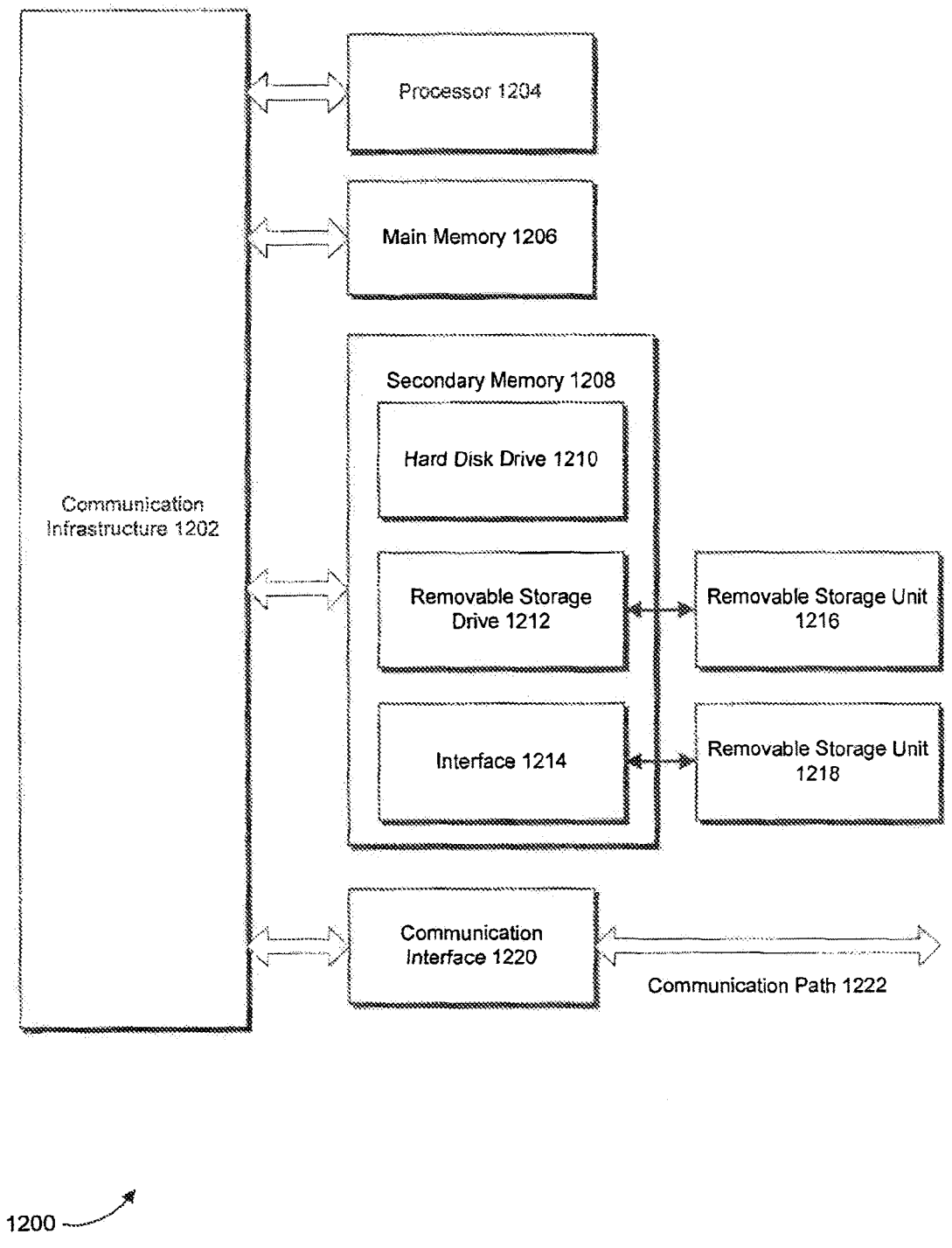

Processor 1204
Main Memory 1206
Secondary Memory 1208
Hard Disk Drive 1210
Communication Infrastructure 1202
Removable Storage Drive 1212
Removable Storage Unit 1216
Interface 1214
Removable Storage Unit 1218
Communication Interface 1220
Communication Path 1222
1200

FIG. 12

PROACTIVE POWER MANAGEMENT USING A POWER MANAGEMENT UNIT

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Patent Application No. 61/524,538, filed on Aug. 17, 2011, which is incorporated by reference herein in its entirety.

FIELD OF THE INVENTION

[0002] This invention relates to power management and more specifically to power management units.

BACKGROUND

[0003] Many devices include power management units (PMUs) for managing power supplied to different elements of a device (e.g., to memories, processors, various hardware subsystems, and software). Many conventional PMUs are limited to adjusting power reactively. That is, many conventional PMUs adjust the power supplied to a particular element of a device in response to the occurrence of a certain event. For example, when an element in a device begins to draw a large amount of power, a reactive PMU can detect this event and can temporarily increase the amount of power supplied to the element. However, many conventional PMUs typically do not predict the amount of power a particular element of a device will require.

[0004] PMUs that are only capable of reactively managing power have several disadvantages. For example, because such PMUs typically do not attempt to make power consumption predictions for applications and/or for elements of a device, they cannot plan for future power consumption needs of applications and/or device elements.

[0005] What is needed are methods and systems for incorporating a proactive power management scheme into a device.

BRIEF DESCRIPTION OF THE DRAWINGS/FIGURES

[0006] The accompanying drawings, which are incorporated in and constitute part of the specification, illustrate embodiments of the disclosure and, together with the general description given above and the detailed descriptions of embodiments given below, serve to explain the principles of the present disclosure. In the drawings:

[0007] FIG. **1** is a block diagram of a device incorporating PMU in accordance with an embodiment of the present disclosure.

[0008] FIG. **2** is a more detailed block diagram showing elements of a CPU.

[0009] FIG. **3** is a more detailed block diagram showing elements of a wireless subsystem.

[0010] FIG. **4** is a flowchart of a method of proactively managing power for a device using a PMU in accordance with an embodiment of the present disclosure.

[0011] FIG. **5** is a flowchart of a method for reconfiguring one or more subsystems of a device based on information from a previous execution of an application or a previous operation of a subsystem of a device.

[0012] FIG. **6** is a flowchart of a method for reconfiguring one or more subsystems of a device based on a power priority determination.

[0013] FIG. **7** is a flowchart of a method for optimizing power used by a device based on a periodic evaluation of the subsystems and applications of the device.

[0014] FIG. **8** is a flowchart of a method for optimizing power supplied to subsystems and applications based on information received from sensors.

[0015] FIG. **9** is a flowchart of a method for reconfiguring a memory based on cache miss rates.

[0016] FIG. **10** is a flowchart of a method for reconfiguring subsystems of a device based on predictions made during previous executions of blocks of code.

[0017] FIG. **11** is a flowchart of a method for reconfiguring subsystems of a device based on signals generated from blocks of code.

[0018] FIG. **12** illustrates an example computer system that can be used to implement aspects of the present disclosure.

[0019] Features and advantages of the present disclosure will become more apparent from the detailed description set forth below when taken in conjunction with the drawings, in which like reference characters identify corresponding elements throughout. In the drawings, like reference numbers generally indicate identical, functionally similar, and/or structurally similar elements. The drawing in which an element first appears is indicated by the leftmost digit(s) in the corresponding reference number.

DETAILED DESCRIPTION OF THE INVENTION

[0020] In the following description, numerous specific details are set forth to provide a thorough understanding of the disclosure. However, it will be apparent to those skilled in the art that the disclosure, including structures, systems, and methods, may be practiced without these specific details. The description and representation herein are the common means used by those experienced or skilled in the art to most effectively convey the substance of their work to others skilled in the art. In other instances, well-known methods, procedures, components, and circuitry have not been described in detail to avoid unnecessarily obscuring aspects of the disclosure.

[0021] References in the specification to "one embodiment," "an embodiment," "an example embodiment," etc., indicate that the embodiment described may include a particular feature, structure, or characteristic, but every embodiment may not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the art to affect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

1. OVERVIEW

[0022] Embodiments of the present disclosure provide systems and methods for proactively managing power in a device. A power management unit (PMU) receives information from various subsystems of a device (e.g., from one or more processors, memories, batteries, sensors, etc.) and estimates the total power required by subsystems of the device. Based on this information, the PMU can predict the amount of power required for a particular subsystem or for one or more application(s) to execute. For example, the PMU can use data collected from previous executions of an application (e.g., a video application) to predict an amount of power subsequent

executions of the application will require. Based on this prediction, the PMU can reconfigure the characteristics and/or properties of subsequent executions of applications so that these applications execute more efficiently given the current battery life of the device (e.g., by lowering the resolution of the video to a level where the device will have sufficient power to finish playing it). The PMU can also modify the voltage supplied to one or more subsystems used by the application so that more or less power is available for execution of the application. The PMU can also reconfigure a voltage regulator and/or select a different voltage regulator based on predicted power needs of an application. For example, in an embodiment, a more or less efficient and/or a more or less responsive voltage regulator can be selected based on predicted power needs of an application.

[0023] The PMU can also use additional collected data to optimize power management of the device. For example, the PMU can be configured to make power adjustments based on data collected from sensors and cache memory miss signals. Further, the PMU can predict the power requirements of blocks of code based on previous code executions, and the PMU can receive signals from executing code that alert the PMU to upcoming power requirements.

[0024] By proactively managing power in a device using, for example, one or more of the techniques discussed above, power management can be handled more efficiently and with less noise introduction into the device. Proactive power management advantageously gives the PMU the capability to predict power needs of various subsystems of a device so that the power supplied to these subsystems can be managed in an intelligent way to conserve battery resources.

2. POWER MANAGEMENT SYSTEM

[0025] FIG. **1** is a block diagram of a device **100** incorporating a PMU **102** in accordance with an embodiment of the present disclosure. In an embodiment, device **100** is a communications device (e.g., a cellular handset). However, it should be understood that device **100** can be any computer system that includes a PMU **102**. In an embodiment, PMU **102** is a microcontroller implemented on an integrated circuit (IC). However, it should be understood that, in an embodiment, PMU **102** could be implemented in any form of hardware, software, or a combination of hardware and software.

[0026] PMU **102** is configured to receive information from various subsystems of device **100** and is configured to manage power supplied to these subsystems based at least in part on this received information. For example, in an embodiment, PMU **102** receives information from one or more power sources (e.g., from one or more batteries), represented in FIG. **1** by battery subsystem **104**. For example, PMU **102** can receive information from battery subsystem **104** regarding the available power from battery subsystem **104** and the amount of remaining battery life of battery subsystem **104**. PMU **102** also receives information from various processors of device **100**. For example, PMU **102** can receive information from one or more central processing units (CPUs), represented in FIG. **1** by CPU **106**. For example, this information can include information regarding currently executing code and cache memory miss signals. PMU **102** can also receive information from one or more video processors, represented in FIG. **1** by video processor **108**. This information can include, for example, information regarding the amount of power required to execute video and/or graphics applications and current characteristics of these applications.

[0027] Additionally, the one or more processors (e.g., CPU **106** and video processor **108**) can have access to one or more external memories, represented in FIG. **1** by external memory subsystem **110**. CPU **106** and video processor **108** can access stored data in external memory subsystem **110** when a cache miss occurs from on-chip cache memory. External memory subsystem **110** can include any type of memory, such as, without limitation, read only memory (ROM), electronically erasable programmable read only memory (EEPROM), random access memory (RAM), static RAM (SRAM), dynamic RAM (DRAM), synchronous DRAM (SDRAM), double data rate SDRAM (DDRSDRAM), flash memory, one or more hard disks, etc. In an embodiment, PMU **102** also receives information from external memory subsystem **110**. For example, in an embodiment, information regarding previous executions of applications can be stored in external memory **110**, and PMU **102** can use this information to make predictions regarding subsequent executions of these applications.

[0028] In an embodiment, PMU **102** can also receive information from one or more displays of device **100**, represented in FIG. **1** by display subsystem **112**. For example, this information can include information regarding the current amount of power used by display subsystem **112** and/or current settings of display subsystem **112** (e.g., resolution, brightness, etc.). PMU **102** also receives information from one or more wireless modules of device **100**, represented in FIG. **1** by wireless subsystem **114**. Wireless subsystem **114** can support any type of wireless module including, without limitation, a cellular module, a wireless local area network (WLAN) module, a Bluetooth module, a near field communication (NFC) module, a Global Positioning Satellite (GPS) module, a Global Navigation Satellite System (GNSS) module, etc. In an embodiment, wireless subsystem **114** is coupled to an antenna **116** (e.g., for communications with a cellular base station). PMU **102** can receive information from wireless subsystem **114** including, for example, information regarding the current amount of power used by wireless subsystem **114** and current operating conditions of wireless subsystem **114**.

[0029] PMU **102** can also receive information from one or more sensors, represented in FIG. **1** by sensor subsystem **118**. Sensor subsystem **118** can include, without limitation, acoustic sensors, sound sensors and/or microphones, vibration sensors, motion, angle, displacement, and/or position sensors, electric current and/or electric potential sensors, environmental sensors, velocity and/or acceleration sensors, optical and/or light sensors, pressure and/or force sensors, thermal, heat, and/or temperature sensors, proximity and/or presence sensors, etc. It should be noted that while various sensors of device **100** are drawn in FIG. **1** as being grouped together in sensor subsystem **118**, sensors can be located anywhere on or within device **100** according to embodiments of the present disclosure. PMU **102** can use information from sensors subsystem **118** when determining an amount of power to supply to subsystems of device **100**. For example, in an embodiment, PMU **102** can use information from a proximity sensor of sensor subsystem **118** to decide to lower the resolution of a playing application (e.g., a video application) to conserve power if no user is nearby.

[0030] It should be understood that not all possible subsystems of device **100** are shown in FIG. **1**. For example, in accordance with an embodiment of the present invention, device **100** can include several other subsystems, such as one

or more input/output (I/O) subsystems, one or more speakers, one or more peripheral subsystems, etc.

[0031] FIG. **2** is a more detailed block diagram showing elements of CPU **106**. As shown in FIG. **2**, CPU **106** includes a processor core **200**, a cache memory **202**, and a bridge **206** that couples CPU **106** to external memory subsystem **110**. It should be understood that while only one cache memory **202** is shown in FIG. **2**, multiple cache memories can be used according to embodiments of the present disclosure. For example, in an embodiment, CPU **106** is configured to access multiple levels of cache memory. When a cache miss occurs, CPU **106** accesses a memory of external memory subsystem **110**. In an embodiment, CPU **106** also sends a cache miss signal to PMU **102** when a cache miss occurs. PMU **102** can be configured to optimize power supplied to subsystems of device **100** based on these cache miss signals. For example, PMU **102** can supply more power to external memory subsystem **110** whenever PMU **102** detects an increase in cache misses. Power optimization based on cache misses is explained in greater detail in section **6**.

[0032] FIG. **3** is a more detailed block diagram showing elements of wireless subsystem **114**. As previously discussed, wireless subsystem **114** can include, without limitation, a cellular module (e.g., a cellular modem) **300**, a WLAN module **301**, a GPS module **302**, a GNSS module **304**, a Bluetooth module **306**, and/or a NFC module **308**. Cellular module **300** may be configured to support a variety of standards including, without limitation, 3G, 4G, Long Term Evolution (LTE), etc. PMU **102** can receive information from these modules and can adjust power supplied to subsystems of device **100** based on this information.

[0033] The various elements of FIGS. **1-3** can be integrated onto one or more ICs. In an embodiment, the elements of FIG. **1** are integrated onto a single IC. In another embodiment, the elements of FIG. **1** are implemented on one or more separate ICs. For example, in an embodiment, PMU **102**, CPU **106**, video processor **108**, and wireless subsystem **114** are implemented on a single IC.

3. DEVICE RECONFIGURATION BASED ON POWER ESTIMATION

[0034] PMU **102** can construct a power profile based on information received from various subsystems of FIG. **1** (e.g., from CPU **106**, video processor **108**, and/or subsystems **104**, **110**, **112**, **114**, and **118**). In an embodiment, PMU **102** receives information from every power-using subsystem of device **100**. In another embodiment, PMU **102** only receives information from subsystems of device **100** that use the most power. For example, some subsystems of device **100** (e.g., speakers) may use a relatively small amount of power. Thus, information from subsystems of device **100** that do not consume much power may not be needed to obtain an accurate picture of the total power consumed by various subsystems of device **100**. Rather, in an embodiment, PMU **102** may use information from the most power-expensive subsystems to predict total power consumption by device **100**, and PMU **102** may create a power profile based on this prediction. Using this power profile and the information received from various subsystems of device **100**, PMU **102** can make predictions regarding an amount of power these subsystems will consume and can proactively manage power for the various subsystems of device **100**.

[0035] FIG. **4** is a flowchart of a method of proactively managing power for a device using a PMU in accordance with an embodiment of the present disclosure. In step **400**, information from a plurality of subsystems of a device is received. For example, PMU **102** can receive information from CPU **106**, video processor **108**, and subsystems **104**, **110**, **112**, **114**, and **118**. In step **402**, power that will be required by one or more subsystems based on the information received is estimated. For example PMU **102** can create and/or update a power profile that represents total power required by the device **100** based on the information received in step **400**. Using this power profile, PMU **102** can estimate power that will be required by particular subsystems of device **100** and/or a particular application executing on device **100**.

[0036] PMU **102** can use a variety of technique to estimate the power that will be required by one or more of the subsystems of device based on the information received in step **400**. For example, PMU **102** can estimate the amount of power video processor **108** and display subsystem **112** will require to finish playing a video on a media player application based on information obtained during a previous execution of the media player application. PMU **102** can also make a prediction for power requirements of code being executed based on a previous execution of the same (or similar) code or based on receiving a signal from executing code indicating an amount of power the code will require to finish executing.

[0037] In step **404**, one or more of the subsystems are reconfigured based on the estimated power. PMU **102** can use a variety of techniques to reconfigure the power supplied to one or more of the subsystems of device **100** to optimize the power supplied to various subsystems of device **100**. For example, based on the power estimated in step **402**, PMU **102** can determine that device **100** does not have sufficient power to finish playing a video based on the current power requirements of the various subsystems of device **100**. PMU **102** can lower the quality of the video (e.g., by increasing compression, lowering resolution, dimming the screen, etc.) to save power so device **100** has enough power to finish playing the video. Alternatively, PMU **102** can instruct certain subsystems to shut down to save power.

[0038] In another example of subsystem reconfiguration, PMU **102** can use infor nation obtained from sensor subsystem **118** to determine which subsystems can be reconfigured for power savings. For example, PMU **102** can receive information from a user presence detecting sensor indicating that a user is far away. In such a case, PMU **102** can determine that video quality should be lowered to save power until the user comes closer to device **100**.

[0039] In a further example, PMU **102** can use cache miss rate information received to optimize power supplied to various subsystems of device **100**. For example, if the cache miss rate is sufficiently low, PMU can determine that less power can be supplied to external memory subsystem **110**.

[0040] In an additional example, the frequency of one or more processors (e.g., CPU **106** and/or video processor **108**) can be reduced to save power. Additionally, leakage of the device increases with a corresponding increase in temperature, so running CPU **106** and/or video processor **108** more slowly can lead to additional power efficiency savings due to a decrease in leakage. If PMU **102** predicts that a higher performance by a device will be necessary, PMU **102** can temporarily increase the voltage supplied to a device (e.g., CPU **106** and/or video processor **108**) in anticipation of this higher performance.

[0041] In an additional example, one or more wireless links (e.g., 60 GHz wireless) can be manipulated by PMU **102** to

conserve or increase power usage by altering functions such as video processing. For example, in an embodiment, PMU **102** can decrease the data rate of information transmitted over one or more wireless links to save power. Additionally, PMU **102** can alter the way in which wireless data is transmitted and/or received to conserve or increase power usage. For example, in an embodiment, PMU **102** can adjust a communication scheme using Multiple Input Multiple Output (MIMO) data transmission and/or reception to a communication scheme that uses less power.

[0042] As described above, embodiments of the present disclosure provide multiple techniques for proactively estimating power required by a device and reconfiguring subsystems of the device based on this estimation. Various systems and methods for power estimation and device reconfiguration will now be described in turn in greater detail.

4. RECONFIGURATION BASED ON APPLICATION POWER CONSUMPTION

[0043] Embodiments of the present disclosure provide systems and methods for reconfiguring device **100** based on estimating power required by one or more applications or subsystems of device **100**. Based on information obtained during a previous execution of an application or a previous operation of a subsystem, the amount of power required for the application or subsystem to finish preforming a task can be predicted. Using these predictions, power can be redistributed among applications and/or subsystems so that device **100** has sufficient power to finish performing the task. PMU **102** can determine how to redistribute power using, for example, a power prioritization policy or user input. Additionally, in an embodiment, PMU **102** can periodically evaluate currently active subsystems and/or applications to determine whether power should be redistributed so that important subsystems and/or applications have sufficient power to finish performing tasks.

4.1 Application Power Consumption Predictions

[0044] FIG. **5** is a flowchart of a method for reconfiguring one or more subsystems of device **100** based on information from a previous execution of an application or a previous operation of a subsystem of device **100**. In step **500**, information from a plurality of subsystems of a device is received. For example, PMU **102** can receive information from CPU **106**, video processor **108**, and/or subsystems **104**, **110**, **112**, **114**, and **118**.

[0045] In step **502**, power required by the subsystems is estimated based on the received information. For example, PMU **102** can create and/or update a power profile that represents power required by the various subsystems of device **100** based on the information received in step **500**. PMU **102** can use this power profile to estimate power that will be required by a particular application executing on device **100** or a particular subsystem of device **100**. For example, at any given time, one or more applications or subsystems may be performing a task. In some cases, certain applications or subsystems may not have sufficient power to complete the task. Using this received information, PMU **102** can determine whether an application or subsystem has sufficient power to finish a task that it has begun.

[0046] In step **504**, PMU **102** accesses information obtained during a previous execution of an application or during a previous operation of a subsystem to predict an amount of power that will be required to finish a task (e.g., to finish executing the application or to finish powering the subsystem until a certain task is completed). For example, PMU can access data gathered during a previous execution of a media player application to determine an amount of power that will be required to finish playing a video using the media player application. In an embodiment, this information can be gathered after (or during) execution of applications and can be stored, for example, in external memory subsystem **110**. For example, in an embodiment, PMU **102** can monitor an amount of power a video executing on media player application requires to execute. PMU **102** can store information regarding an amount of power consumed by the media player application.

[0047] In an embodiment, PMU **102** can also optionally record information regarding various characteristics of applications and/or subsystems. For example, PMU **102** can store information regarding a resolution used to play a video, an amount of compression used (e.g., scalable video coding (SVC)), the brightness and contrast used for display subsystem **112** during execution of the video, the total length of the video, etc. This data can be collected after an application has finished executing (or after the application has been closed), or it can be periodically collected as an application is executing or during the operation of a subsystem. Based on this collected information, PMU **102** can create a database of information that it can use later to determine the impact of various characteristics (e.g., application execution time, video compression, brightness, contrast, etc.) on power requirements of an application and/or a subsystem. PMU **102** can optionally use this information in the future to modify these characteristics during execution of applications so that power consumption by applications and subsystems can be optimized. Further, PMU **102** can use this information to intelligently predict future power needs of applications and/or subsystems of device **100**.

[0048] In step **506**, PMU **102** determines that the application or subsystem does not have sufficient power to finish the task. Based on the information obtained in step **504**, PMU **102** can predict the amount of power necessary for the application or subsystem to finish the task. For example, in an embodiment, PMU **102** can determine that the application or subsystem consumes power at a known rate. Based on predicting the amount of time that will be required for the application or subsystem to finish the task, PMU **102** can estimate the amount of power that will be required to finish the task, and PMU **102** can determine that battery resources will be exhausted before the task is completed.

[0049] For example, PMU **102** can estimate the amount of power that will be required to finish playing a video executing on a media player application. PMU **102** can perform this estimation based on the power profile for the system created in step **502**, the remaining power available from battery subsystem **104**, and the information obtained during a previous execution of the media player application obtained in step **504**. PMU **102** can access stored information from previous executions of the media player application to predict an amount of power the video will require to finish executing. For example, if the current video being played is half as long as a previously played video, PMU **102** can estimate that the current video will require half the amount of power as the previously played video.

[0050] In an embodiment, PMU **102** also predicts power required for the application based on an examination of the

impact of application characteristics such as application length, application resolution, the amount of compression used (e.g., SVC) for the application, and the brightness and contrast used for display subsystem **112** during execution of the application. In an embodiment, the impact of these characteristics is known by PMU **102**, and PMU can calculate an amount of power the application will require given the current value of these characteristics. For example, in an embodiment, PMU **102** can know that increasing the resolution of the application by a certain amount will result in a predictable increase in the amount of power required by the application. In another embodiment, PMU **102** can make better predictions for the impact of these characteristics on the power requirements of the application based on stored information gathered during previous executions of applications. Using this stored information, PMU **102** can train itself to more reliably predict power consumption requirements of an application based on current values of application characteristics such as application length, application resolution, the amount of compression used (e.g., SVC) for the application, and the brightness and contrast used for display subsystem **112** during execution of the application.

[0051] In step **508**, one or more of the subsystems is reconfigured based on the estimated power in step **506**. For example, when PMU **102** estimates the power that will be required for the video to finish playing, PMU **102** can determine that battery subsystem **104** does not have enough power left to finish playing the video. In such a case, PMU **102** can determine (e.g., based on a policy or by prompting the user) that the power supplied to one or more subsystems of device **100** should be reconfigured and/or redistributed so that the device has sufficient power to finish playing the video. Thus, in step **508**, one or more of the subsystems of device **100** are reconfigured based on the estimated power from step **506**. For example, one or more subsystems can be temporarily disabled by PMU **102** or instructed by PMU **102** to operate at a reduced functionality so that more battery power can be conserved to execute the media player application. PMU **102** can also determine that one or more characteristics of the media player application should be modified for power conservation. For example, PMU **102** can calculate (e.g., in step **506**) that device **100** will not have sufficient power to finish playing the video with the current SVC mode being used but will have sufficient power to finish playing the video if a higher SVC mode is used. Therefore, PMU **102** can determine that the video quality should be lowered (e.g., by using a higher SVC mode) to conserve power so that the entire video can be played.

4.2 Power Distribution Prioritization

[0052] PMU **102** can determine how various subsystems of device **100** should be reconfigured to conserve power using a variety of methods. In an embodiment, PMU **102** can access a power prioritization policy to determine which subsystems and/or applications should be disabled and/or reconfigured to operate at a lower power mode if device **100** is operating in a low power state and needs to assign more power to a high priority application or subsystem. PMU **102** can also prompt a user for permission to reassign power once PMU **102** determines (e.g., in step **506**) that an application or subsystem does not have sufficient power.

[0053] FIG. **6** is a flowchart of a method for reconfiguring one or more subsystems of a device based on a power priority determination. In step **600**, PMU **102** determines (e.g., based on a power estimate such as the power estimate performed in step **506** of FIG. **5**) that device **100** does not have sufficient power to finish a task associated with a high priority application or a high priority subsystem. For example, PMU **102** can determine that wireless subsystem **114** does not have sufficient power to finish an in-progress download or upload of data. In a media player example, PMU **102** can determine that a media player application does not have sufficient power to finish playing a video.

[0054] In step, **602**, PMU **102** determines whether a power prioritization policy provides instructions regarding how to reassign power when device **100** does not have sufficient power to finish performing the task. For example, PMU **102** can access a power prioritization policy (e.g., stored in external memory subsystem **110**). This power prioritization policy can dictate which subsystems of device **100** should be prioritized and which subsystems can be shut down or put into a lower power state if battery resources need to be conserved. For example, the power prioritization policy may state that certain subsystems of wireless subsystem **114** (e.g., Bluetooth subsystem **306**, GPS subsystem **302**, or GNSS subsystem **304**) can be powered off to save power if a higher priority subsystem requires additional power to perform a task. The policy may also provide instructions for assigning less power to certain subsystems. For example, the policy may instruct PMU **102** to lower the brightness, resolution, etc. of display subsystem **112** to save power.

[0055] In an embodiment, the power prioritization policy can also specify one or more high priority applications. For example, in an embodiment, important applications like operating system applications and cellular telephone applications can always be assigned a high priority. The policy can provide a list of certain lower priority applications that can be disabled if power needs to be conserved. The policy can also dictate that certain active applications (e.g., a video currently being played on a media player application) should be assigned a high priority and can provide instructions for disabling certain lower priority subsystems and/or applications so that device **100** has sufficient power to finish performing the task.

[0056] If the power prioritization policy provides instructions for power reassignment, PMU **102** determines which subsystems and/or applications to reconfigure in step **604**. In an embodiment, PMU **102** can calculate an estimated amount of power that will be saved by disabling and/or reconfiguring subsystems and/or applications and can determine how application and/or subsystem reconfiguration will be necessary for device **100** to have sufficient power to finish performing the task.

[0057] For example, a power prioritization policy may state that if there is insufficient battery power to finish playing a video, video quality should be lowered until enough battery power will be conserved to finish the video. Several options may be available for reducing video quality to save power. Such options can include, without limitation: playing the video using a higher SVC mode with battery depletion occurring before the video ends; using a lower SVC mode wherein battery will prove sufficient with minimal margin; adapting SVC with I-frame sync on the fly to minimize higher power requirement portions of the stream or more accurately utilize the full remaining battery power; playing the video in a much lower SVC mode than is required to maximize battery savings; reducing video resolution; reducing screen brightness; and/or reducing power supplied to the backlight display.

[0058] Based on the current power requirements of device **100** (e.g., as determined in step **502** of FIG. **5**) and the battery power available to device **100**, PMU **102** can estimate an amount of power it needs to save to finish playing the video. PMU **102** can then determine an amount of compression necessary for device **100** to meet these required power savings. For example, in an embodiment, PMU **102** can know in advance an amount of estimated power that can be saved by increasing compression, reducing resolution, reducing screen brightness, etc. Based on this information, PMU **102** can determine which actions it can take to achieve sufficient power-savings for device **100** to have sufficient power to finish playing the video. In an embodiment, PMU **102** can also access information obtained during previous executions of the media player application to assist it in estimating an amount of power that will be saved by increasing compression, reducing resolution, reducing screen brightness, etc.

[0059] The power prioritization policy can also state that certain subsystems and/or applications should be assigned less power if reconfiguring characteristics of the media player application is insufficient to achieve the needed power savings. In an embodiment, the power prioritization policy can state that certain subsystems (e.g., Bluetooth subsystem **306**) and/or applications should be powered down until sufficient power savings are achieved. For example, PMU **102** can throttle wireless connectivity, alter display qualities (e.g., resolution), switch to a mode that uses more text and less graphics, dim the backlight display, reduce the times lights are turned on and off, and/or reduce the duty cycle of wireless subsystem **114** (e.g., lower the rate at which device **100** scans for available wireless networks). As discussed above, in an embodiment, PMU **102** can know in advance an amount of estimated power that will be saved by powering down these applications and/or subsystems. In an embodiment, PMU **102** can access information obtained during previous executions of applications and/or subsystems to aid it in determining an amount of power that will be saved by changing certain characteristics of an application.

[0060] In an embodiment, the power prioritization policy can also specify that device **100** should be placed in a certain power modes based on the current amount of power available from battery subsystem **104**. Each power mode can be associated with one or more instructions for assigning power to subsystems and/or applications. For example, in a high power mode, full power can be assigned to all subsystems and applications. In an intermediate power mode, certain subsystems and/or applications can be disabled to save power. In a low power mode, even important subsystems and/or applications can be assigned less power or can be instructed to operate at reduced performance so that power can be saved. In an embodiment, PMU **102** can switch the power mode of device **100** in response to determining (e.g., in step **604**) that subsystems and/or applications should be reconfigured to save power.

[0061] In an embodiment, PMU **102** ideally optimizes power to provide the most functionality that it is able to provide given the current battery life of device **100**. For example, in the case of a video playing on a media player application, PMU **102** ideally determines to optimize power so that the media player application will be able to play the video using the best quality possible while still having enough power available to finish playing the video. In some cases, PMU **102** may not be able to save sufficient power for device **100** to finish performing the task. In an embodiment, PMU **102** can take steps to conserve power so that the battery life of the device will be extended, even if the device will run out of power before the task has been finished.

[0062] After PMU **102** determines which subsystems and/or applications should be reconfigured in step **604**, PMU **102** can then optionally notify the user of the upcoming reconfigurations in step **606**. For example, in an embodiment, a message can be sent to a user informing the user of a change in power settings before PMU **102** reconfigures the power supplied to subsystems and/or applications. The user can also optionally be given the chance to choose to reject the proposed changes and continue with the current power settings. In an embodiment, the user can determine (e.g., based on a device or policy setting) whether he or she wants to receive such power notifications. For example, the user can configure device **100** to conserve power automatically without notifying the user, or the use can configure device **100** to take no automatic power savings action. Finally, in step **608**, PMU reconfigures the one or more subsystems and/or applications based on the policy.

[0063] In some cases, a power prioritization policy may not provide instructions for reassigning power if device **100** does not have sufficient power to finish performing the task. In such a case, PMU **102** can inform the user that the host device has insufficient power to finish performing the task in step **610**. PMU **102** can then optionally propose power savings options to the user in step **612**. For example, PMU **102** can initiate sending a message to the user (e.g., via a pop up display) giving the user options for reconfiguring the power settings of device **100**. For example, the user can be given the option to place device **100** in a lower power mode, to disable some applications (e.g., background applications and/or multi-task applications, such as applications enabling simultaneous video playback and file downloading) and/or subsystems, and/or to reduce the quality of certain active applications (e.g., reducing the quality of a playing video). PMU **102** can then reconfigure subsystems and/or applications based on the user input in step **614**. In an embodiment, PMU **102** can also update the policy based on the user input to avoid the need for future input from the user. By updating the policy based on user inputs, PMU **102** can train itself to adapt to the user's preferences. As previously discussed, user notification is optional, and the user can configure device **100** (e.g., in a device setting or in the power prioritization policy) so that notifications are not sent to the user.

4.3 Additional Embodiments

[0064] While reconfiguration of device **100** for power savings is described generally above with reference to an example of a media player application playing a video, it should be understood that embodiments of the present disclosure provide systems and methods for power savings for a variety of applications and/or subsystems of device **100**. For example, in an embodiment, PMU **102** may determine that battery power is low while a phone call is in progress. PMU **102** can then take actions to maximize battery life so that additional time is available to complete the phone call before the battery is out of power. For example, PMU **102** may determine (e.g., based on the power prioritization policy) that certain applications (e.g., graphics programs) and/or subsystems (e.g., Bluetooth subsystem **306**) should be disabled to maximize battery life. In another example, if battery power is low while a file is being downloaded using wireless subsystem **114**, PMU **102** can redistribute power from other

subsystems or applications to provide more power to wireless subsystem **114**. For example, if device **100** is not currently in use by the user, PMU **102** can determine (e.g., based on a stored power prioritization policy) that display system **112** can temporarily be powered down and/or that device **100** can be put into a standby mode until the user performs some action to wake up device **100** again.

[0065] Additionally, in an embodiment PMU **102** can coordinate with other devices to optimize power usage among several devices. For example, in an embodiment, device **100** can be a cellular phone running an application that is displayed on a different device (e.g., a tablet computer). In the case of a video executing using a media player application, device **100** can be executing the media player application, but the video itself can be displayed on the tablet. For example, device **100**'s cellular subsystem **300** can be used to receive a LTE video stream from a remote server, and device **100**'s WLAN subsystem **301** can be used to transmit the video stream to a tablet. PMU **102** can be configured to monitor the power consumption from the LTE network and the WiFi link and adjust the bitrate of the video accordingly to manage battery life. For example, if PMU **102** determines that the battery power of the tablet receiving the video is low, PMU **102** can send a message to WLAN subsystem **301** to compress the video so that video quality is lowered but power is saved on the tablet. Alternatively, if PMU **102** determines that the tablet has high battery power but that device **100** has low battery power, PMU **102** can determine that no video compression should be performed using device **100** to save power on device **100**. Video compression can then optionally be performed on the tablet, if necessary.

4.4 Recurring Power Prioritization

[0066] In an embodiment, PMU **102** can periodically evaluate currently active subsystems and/or applications to determine whether power should be redistributed so that important subsystems and/or applications have sufficient power to finish performing tasks. In an embodiment, the rate at which device **100** evaluates subsystems and applications can be set in a policy (e.g., a power prioritization policy), and a user can optionally reconfigure this policy to change this rate. The policy (and/or the user) can also set other parameters to determine when this periodic evaluation should occur. For example, in an embodiment, the policy and/or the user can determine that the periodic evaluation of subsystems and applications should only occur when device **100** is in a low power mode (and when device **100** is not plugged in to an electrical outlet). Alternatively, the policy and/or the user can determine that subsystems and applications should periodically be evaluated regardless of current battery life so that power supplied to subsystems and applications can be continually optimized.

[0067] FIG. **7** is a flowchart of a method for optimizing power used by device **100** based on a periodic evaluation of the subsystems and applications of device **100**. In step **700**, device **100** periodically identifies high priority subsystems and applications. For example, in an embodiment, these subsystems and applications can be listed in the power prioritization policy. In an embodiment, all currently executing applications and currently operating subsystems can be considered as having a high priority. In step **702**, each identified high priority subsystem and application is evaluated in turn to determine whether power supplied to device **100** needs to be rebalanced.

[0068] In step **704**, the power requirements of the high priority subsystem or application are predicted. For example, device **100** estimates an amount of power that will be required for the application or subsystem to finish performing a task. In step **706**, PMU **102** determines whether device **100** has sufficient power for the application or subsystem to complete the task. If PMU **102** determines that sufficient power is available, no further action is taken in step **708**, and PMU **102** evaluates the next identified high priority subsystem or application in step **702**. If PMU **102** determines that the subsystem or application does not have sufficient power to complete the task, PMU **102** initiates a reconfiguration of subsystems and/or applications of device **100** in step **710** so that sufficient power is conserved such that the subsystem or application can complete the task.

[0069] PMU **102** can also be configured to increase power supplied to subsystems and applications when more power becomes available to device **100**. For example, in an embodiment, when PMU **102** periodically evaluates the power needs of high priority subsystems and applications, PMU **102** can also determine whether device **100** has been supplied with additional power (e.g., if device **100** has been plugged in or supplied with a new battery). In an embodiment, if PMU **102** determines that device **100** currently has a high amount of power available (e.g., if device **100** has been plugged in), and if the subsystems or application is currently operating with a reduced quality to save power, PMU **102** can increase the amount of power supplied to the application or subsystem so that quality can be improved.

[0070] By periodically evaluating the power needs of subsystems and applications, PMU **102** can proactively optimize the power needs of device **100**. This proactive optimization can lead to a greater power efficiency than that achieved by purely reactive power optimization schemes. Further, the power savings resulting from this proactive power optimization can allow device **100** to support a higher quality for important applications and subsystems.

5. POWER ADJUSTMENTS BASED ON INPUTS FROM SENSORS

[0071] As previously discussed, PMU **102** can evaluate input from a variety of sources to determine how power supplied to subsystems and applications should be optimized. In an embodiment, PMU **102** can be configured to adjust power supplied to subsystems and/or applications based on information received by PMU **102** from various sensors (e.g., from sensors of sensor subsystem **118**). For example, a proximity or range sensor can sense whether a user is present near device **100** and/or how far a user is from device **100**. In an embodiment, the proximity or range sensor can determined whether the user is within a predetermined range of device **100**. If sensor information indicates that a user is far away, PMU **102** can determine (e.g., based on information stored in a power prioritization policy) that power supplied to certain subsystems and/or applications can be temporarily cut off or reduced until the user comes closer to device **100**. For example, PMU **102** can determine that the quality of a playing video should be temporarily reduced to save power until the user comes closer to device **100**.

[0072] In another example, a sensor (e.g., one of the sensors of sensor subsystem **118**) can detect whether the environment is bright or dark. Based on this information, PMU **102** can reduce device illumination (e.g., screen brightness) when the information from the sensor indicates that the environment is

brighter than a predetermined threshold value so that power can be conserved. When the environment is darker than a predetermined threshold value, PMU **102** can increase device illumination so that the user can see the screen more easily in the dark.

[0073] In an additional example, a sensor (e.g., one of the sensors of sensor subsystem **118**) can detect when device **100** is operating in a hot environment and can reduce the maximum allowed performance (e.g., by reducing processing speed) to avoid overheating the device. For example, in an embodiment, a thermal sensor of sensor subsystem **118** can detect the temperature of device **100** or one or more portions of device **100** (e.g., the case, battery, or circuitry of device **100**), and this information can be relayed to PMU **102**. If PMU **102** determines that the temperature has exceeded a predetermined threshold, PMU **102** can alter power management to account for this increase in temperature. For example, PMU **102** can reduce the maximum allowed performance (e.g., by reducing processing speed) once PMU **102** has determined that the temperature threshold has been exceeded. Additionally, in an embodiment, PMU **102** can use multiple temperature thresholds in a power management scheme. For example, when PMU **102** determines, based on sensor information, that a first temperature threshold has been exceeded, PMU **102** can reduce the maximum allowed performance (e.g., by reducing processing speed) by a first amount. If PMU **102** determines that a second (higher) temperature threshold has been exceeded (e.g., if the device is continuing to overheat further), PMU **102** can reduce the maximum allowed performance by an additional amount (e.g., by reducing processing speed even further).

[0074] For further example, a motion and/or position sensor can sense whether a device is being held, is stationary, is face down, is face up, is sideways, and/or is in a pocket. PMU **102** can use information from these sensors to adjust how power supplied to subsystems and applications. For example, if PMU **102** determines, based on sensor information, that device **100** is in the pocket of a user, PMU **102** can instruct display subsystem **112** to temporarily show no picture until device **100** is removed from the user's pocket.

[0075] In another example, an accelerometer can inform PMU **102** when a user is walking. In such a case, PMU **102** can instruct GPS subsystem **302** and/or GNSS subsystem **304** to reduce the rate at which they access satellite information to determine the geographic location of the device. Instead, information from the accelerometer can be used to update the user's location based on a previous GPS/GNSS location determination.

[0076] FIG. **8** is a flowchart of a method for optimizing power supplied to subsystems and applications based on information received from sensors. In step **800**, PMU **102** receives information from one or more sensors of device **100** indicating that an event has occurred. For example, PMU **102** can receive information from a proximity sensor of sensor subsystem **118** informing PMU **102** that no user is near device **100**. In step **802**, PMU **102** determines (e.g., based on a power prioritization policy) whether the sensor information can be used to optimize power supplied to applications or subsystems of device **100**. For example, PMU **102** can access a stored power prioritization policy to determine if it contains instructions for reconfiguring power supplied to subsystems or applications based on the sensor information. In step **804**, PMU **102** reconfigures (e.g., according to instructions in the policy) one or more subsystems and/or applications based on the sensor information. For example, PMU **102** can determine that a media player application can be instructed to operate at a reduced quality to save power until the user comes closer to device **100**.

[0077] In an embodiment, PMU **102** can use sensor information to assist in the reconfiguration of subsystems and applications when performing the methods of FIGS. **4-7** and **9-11**. For example, in the method of FIG. **6**, when PMU **102** identifies one or more subsystems and/or applications to reconfigure in step **604** based on the power prioritization policy in step **604**, PMU **102** can use sensor information to determine which subsystems and/or applications can be temporarily shut down or assigned less power. Additionally, for example, sensor information can be used to more efficiently allocate power to subsystems and applications when PMU **102** periodically determines whether power should be reallocated to subsystems and/or applications in the method of FIG. **7**.

6. POWER ADJUSTMENTS BASED ON CACHE MISSES

[0078] As previously discussed, PMU **102** can also use cache miss rate information when determining how power should be reallocated to subsystems and applications. Using cache miss rate information, cache memory (e.g., cache memory **202**) and memory of external memory subsystem **110** can be configured to operate more efficiently, leading to greater power savings.

[0079] Cache memories store copies of data stored elsewhere in memory (e.g., in external memory subsystem **110**). Cache memories can be configured to fetch data in advance based on predicting what data a processor (e.g., CPU **106** or video processor **108**) will attempt to access from memory. Because cache memories are typically faster than external memories, reading data from cache memory improves performance. Thus, to increase memory access speed, processors can first check cache memory for a copy of the data before attempting to access the original data in slower external memory. A cache miss or mistake occurs when a processor (e.g., CPU **106** or video processor **108**) fails to read from or write to a cache memory (e.g., cache memory **202**) and instead reads the data from external memory. For example, when CPU **106** needs to read data or write data to a main memory (e.g., to a memory of external memory subsystem **110**), CPU **106** or video processor **108** can first determine whether a copy of the data is present in cache memory **202**. If the data is present in cache memory **202**, CPU **106** can read the data from cache memory **202**. If the data is not present in cache memory **202**, CPU **102** can access the original data in external memory subsystem **110**.

[0080] In an embodiment, each processor of device **100** has one or more on-chip cache memories. While FIG. **2** only shows a cache memory **202** of CPU **106**, it should be understood that other processors of device **100** (e.g., video processor **106**) can also have on-chip cache memories. Further, it should be understood that each processor of device **100** (e.g., CPU **106** and video processor **108**) can have access to multiple cache memories. For example, in an embodiment, CPU **106** has access to a multi-level cache with three different cache levels (e.g., L1, L2, and L3).

[0081] Memories of external memory subsystem **110** can have one or more power modes and can be placed in a sleep or low power state when not being accessed to save power. In an embodiment, PMU **102** can begin powering up a memory of

external memory subsystem **110** when PMU **102** receives a cache miss signal. For example, the cache miss signal can inform PMU **102** that a cache miss has occurred when CPU **106** tried to access a copy of a piece of data and that the original data is stored in a first memory (not shown) of external memory subsystem **110**. Based on this information, PMU **102** can power up the first memory so that the original data can be accessed. By keeping memories of external memory subsystem **110** in an inactive or low power state until a cache miss occurs, device **100** can use battery power more efficiently.

[0082] In an embodiment, an application can be characterized (e.g., by PMU **102**) using information stored during previous executions of the application. Using information stored during previous executions of an application, PMU **102** can track the activity of an application (e.g., memory accesses) to determine how to optimize subsystems (e.g., external memory subsystem **110**) for execution of the application. For example, in an embodiment, PMU **102** can track how much buffer is maintained between processing and display. If PMU **102** detects that too little buffer is used or too much buffer is used, PMU **102** can determine that the application is being run inefficiently and can initiate one or more subsystem reconfigurations to increase efficiency. For example, information can be placed into a frame header indicating a "cost" associated with decoding the frame (e.g., an estimate of how much power will be required to decode the frame), and a decoder can be reconfigured based on information in this header.

[0083] Thus, each application can have metadata that indicates some information regarding its behavior (e.g., memory accesses, cost of decoding frames, etc.) based on previous executions of the application, and this metadata can be used to reconfigure one or more subsystems and make one or more predications about how the application will execute (e.g., predicted memory accesses). Additionally, as an application executes, PMU **102** can detect whether the application is behaving according to predicted behavior. PMU **102** can initiate additional subsystem reconfigurations if the application is not behaving according to predicted behavior. Additionally, in an embodiment, PMU **102** can proactively "test" the application (e.g., by temporarily slowing it down) to see if it can take power savings measures (e.g., saving by reducing frequency) without significantly impacting performance. In this way, PMU **102** can optimize application power efficiency as the application executes.

[0084] For example, if PMU **102** determines (e.g., based on application metadata) that a previous execution of an application resulted in a very high cache miss rate to a particular cache memory, then PMU **102** can determine that the access rate of this cache memory should not be increased because an increase in the access rate of the cache memory will likely burn power without resulting in increased performance. In some cases, PMU **102** can determine that the access rate of the cache memory should be decreased to conserve power. Additionally, if PMU **102** detects that an application initiates relatively few accesses to memory, then PMU **102** can place this memory into a low power mode.

[0085] In an embodiment, memories of external memory subsystem **110** can have multiple power modes, including one or more intermediary power modes (e.g., power modes other than full power and no power). PMU **102** can use cache miss signals to determine which power mode to assign to each memory of memory subsystem **110**. For example, if a first memory of external memory subsystem **110** can be accessed at 400 MHz at full power, the first memory can be configured (e.g., by PMU **102**) to support a 100 MHz access rate in an intermediary state. By reducing the memory access rate (e.g., to 100 MHz) additional power can be saved while still allowing some memory access. In an embodiment, memories of external memory subsystem **110** can also be configured to save power by controlling memory lanes. By reducing the number of lanes available for memory access from a first memory, the first memory will use less power while still supporting some memory access. For example, the interface of the first memory can be configured to be 16 bits wide instead of 128 bits wide.

[0086] In an embodiment, PMU **102** can reconfigure the power modes of the memories of external memory subsystem **110** based on detecting an increase or a decrease in the rate of cache miss signals. For example, PMU **102** can place a first memory of external memory subsystem **110** in a higher power mode (e.g., a power mode supporting a higher access rate or a greater number of memory lanes) when cache miss rates from the first memory increase (e.g., when PMU **102** receives more cache miss signals from the first memory). To save power, PMU **102** can place the first memory in a lower power mode (e.g., a power mode supporting a lower access rate or a smaller amount of memory lanes) when cache miss rates from the first memory decrease.

[0087] In an embodiment, cache memory (e.g., cache memory **202**) can be reconfigured to operate in one or more intermediary power states for power savings. For example, cache **202** can be configured (e.g., by PMU **102**) to operate at a reduced access rate or to use fewer memory lanes for additional power savings. Additionally, in an embodiment, cache memory (e.g., cache memory **202**) can be programmed (e.g., by PMU **102**) to make it more efficient. For example, based on a prediction of what code is going to be executed by a processor, caches can be instructed to automatically fetch instructions so that they are available when the processor is ready to execute them. If PMU **102** determines (e.g., based on signals received from CPU **106** and/or video processor **108**) in advance that the same code is going to be continuously executed for a period of time, PMU can program a cache (e.g., cache memory **202**) and can lock it so that it no longer expends unnecessary resources fetching the same data.

[0088] FIG. **9** is a flowchart of a method for reconfiguring a memory based on cache miss rates. In step **900**, PMU **102** monitors a plurality of cache miss signals (e.g., sent by CPU **106**). In step **900**, PMU **102** detects a change in the rate of cache miss signals associated with data stored in a first memory. For example, PMU **102** can determine that the number of cache misses associated with data stored in a first memory of external memory subsystem **110** is increasing. In step **904**, PMU **102** determines how the first memory or the cache should be reconfigured based on the detected change in the rate of cache miss signals. In an embodiment, PMU **102** can access a power conservation policy that instructs PMU **102** how to respond to an increase or decrease in received cache miss signals. For example, if PMU **102** detects an increase in cache misses from cache **202** associated with a first memory of external memory subsystem **110**, PMU **102** can change the memory mode of the first memory to support a higher accessed rate and/or a greater number of memory lanes. PMU **102** can also deter nine (e.g., based on the policy) that that cache memory **202** should be reconfigured to a power mode supporting an increased access rate and/or a greater

number of memory lanes. Finally, in step **906**, PMU **102** reconfigures the first memory or the cache (e.g., based on the stored policy).

[0089] In an embodiment, PMU **102** can use cache miss signal information to assist in the reconfiguration of subsystems and applications when performing the methods of FIGS. **4-8**, **10**, and **11**. For example, in the method of FIG. **6**, when PMU **102** identifies one or more subsystems and/or applications to reconfigure in step **604** based on the power prioritization policy in step **604**, PMU **102** can use cache miss signal information to select memories (e.g., memories of external memory subsystem **110** and/or cache memory **202**) to reconfigure for power savings. Additionally, for example, cache miss signal information can be used to more efficiently allocate power to memories (e.g., memories of external memory subsystem **110** and/or cache memory **202**) when PMU **102** periodically determines whether power should be reallocated to subsystems and/or applications in the method of FIG. **7**.

7. POWER CONSUMPTION PREDICTIONS BASED PREVIOUS CODE EXECUTIONS

[0090] As previously discussed, PMU **102** can analyze data collected during previous executions of a program when making predictions regarding the overall power that will be needed to execute the program. PMU **102** can also use data gathered during previous executions of blocks of code to predict thermal and power consumption needs. For example, when a program is running, the same block of code is often repeatedly executed. Certain blocks of code can require a particularly large amount of power. PMU **102** can store information regarding an amount of power that was consumed during execution of a block of code and can later use this information to predict an amount of power a subsequent execution of the block of code will require. For example, PMU **102** can be configured to estimate power requirements for blocks of code when particular code elements (e.g., branch instructions) are encountered. These code elements can signal that a new block of code is about to be executed. For example, when PMU **102** encounters a branch instruction, PMU **102** can access memory to determine if the block of code following the branch instruction is associated with power consumption information stored in memory. Additionally, in an embodiment, PMU **102** can also estimate an amount of power a new block of code will require for execution based on data gathered from previous executions of similar code.

[0091] In an embodiment, power consumption information for blocks of code can be associated a pattern of execution of the block of code. For example, when a block of code is executed, the instructions in the block of code can follow a particular pattern of memory accesses during execution of the instructions in the block of code. Based on the addresses of the memory segments accessed during execution of the block of code, a signature (e.g., a hash) for the pattern of execution can be generated. This signature can be stored (e.g., in a table in memory), and each signature can be associated with power consumption information. For example, PMU **102** can record an amount of power that was consumed when the block of code was executed and can store this information in memory and associate it with the signature. When the block of code is executed again in the future, PMU **102** can access the stored information to determine whether there is a signature associated with the block of code. For example, PMU **102** can generating another hash (e.g., using the same hash algorithm) based on the memory addresses that will be accessed when the block of code is executed. PMU **102** can then determine whether this newly generated hash matches a hash already stored in memory. If a match is found, PMU **102** can access the power consumption information associated with the hash. Using the stored power consumption information, PMU **102** can predict an amount of power the block of code will require to execute.

[0092] Using this prediction, PMU **102** can determine (e.g., based on a policy) whether it should take any actions based on the predicted power consumption needs. For example, PMU **102** can determine that voltage supplied to a processor (e.g., to CPU **106** and/or video processor **108**) should be increased or that the frequency of the processor should be increased. PMU **102** can also determine to reconfigure memory (e.g., cache memory **202** or memories of external memory subsystem **110**) based on predicted power consumption needs. For example, PMU **102** can determine that cache memory **202** or memories of external memory subsystem **110** should be placed into a higher power mode if PMU **102** predicts that these memories will be accessed during execution of the block of code.

[0093] FIG. **10** is a flowchart of a method for reconfiguring subsystems of a device based on predictions made during previous executions of blocks of code. In step **1000**, a signature is generated for a pattern of access associated with a block of code. For example, PMU **102** determines which memory addresses will be accessed when the block of code is executed and generates a hash based on these addresses. In step **1002**, power consumption information for the block of code is stored in memory and associated with the signature. For example, PMU **102** can detect an amount of power that was required for execution of the block of code and can store this information in a table in memory associated with the generated hash. In step **1004**, a subsequent execution of the block of code is detected. For example, PMU **102** can detect that a block of code is about to be executed when a branch instruction is encountered. In an embodiment, a processor (e.g., CPU **106** or video processor **108**) can send a signal to PMU **102** when branch instructions are encountered. Based on these signals, PMU **102** can access the table in memory in step **1006** to determine if it contains stored information associated with the block of code. For example, PMU **102** can generate a hash of the addresses in memory that will be accessed when the block of code will be executed and can check the table to see if any entry matches the generated hash. If a match is found, PMU **102** can access the corresponding power consumption information to obtain a prediction of the power consumption requirements of the block of code.

[0094] Finally, in step **1008**, subsystems of device **100** can be reconfigured based on the power consumption information retrieved from memory. For example, PMU **102** can determine that voltage supplied to a processor (e.g., to CPU **106** and/or video processor **108**) should be increased or that the frequency of the processor should be increased. PMU **102** can also determine to reconfigure memory (e.g., cache memory **202** or memories of external memory subsystem **110**) based on predicted power consumption needs.

[0095] In an embodiment, PMU **102** can use this retrieved power consumption information to assist in the reconfiguration of subsystems and applications when performing the methods of FIGS. **4-9** and **11**. For example, in the method of FIG. **6**, when PMU **102** identifies one or more subsystems

and/or applications to reconfigure in step **604** based on the power prioritization policy in step **604**, PMU **102** can use retrieved power consumption information to determine whether CPU **106** and/or video processor **108** should be assigned more power (e.g., if a particularly power-expensive block of code is about to be executed).

8. ADJUSTING POWER SUPPLIED BASED ON CODE SIGNALS

[0096] In an embodiment, potions of code in a program can signal upcoming power requirements to PMU **102** so that PMU **102** can act proactively to meet these power requirements. For example, code for power consumption prediction (e.g., "energy hints") can be inserted into the code of power-expensive programs or functions so that PMU **102** can be notified of an increase in needed power before these programs are executed. When the code is executed, the energy hint code sends a message to PMU **102** informing PMU **102** that the power-expensive block of code is being executed. In an embodiment, no operation performed (NOP) instructions can be used as padding to give PMU **102** time to adjust power. NOPS periodically inserted into code can prevent a large, instantaneous burden from being placed on capacitors in device **100**. Using energy hints, PMU **102** can proactively reconfigure subsystems of device **100** before the power-expensive code starts drawing a large amount of power. For example, PMU **102** can determine, in response to receiving information from an energy hint, that voltage supplied to a processor (e.g., to CPU **106** and/or video processor **108**) should be increased or that the frequency of the processor should be increased.

[0097] In an embodiment, energy hints can be used to reconfigure characteristics of hardware elements of device **100**. For example, energy hints can be especially useful for configuring two-phase switchers in power supplies. A switched-mode power supply is a power supply that incorporates a switching regulator. Power supplies having two-phase switchers have two switches that change between on and off states. As the switching frequency of a two-phase switcher increases, more energy is used. Thus, two-phase switchers can be less efficient at higher frequencies. Using received energy hints, PMU **102** can adapt the switching frequency of two-phase switchers. For example, when PMU **102** determines that less power is expected to be used, PMU **102** can instruct two-phase switchers to reduce switching frequency to save energy. In an embodiment, two-phase switchers can be instructed to forgo operating at very high switching frequencies unless an energy hint is received by PMU **102** indicating that a program will require a large amount of power.

[0098] FIG. **11** is a flowchart of a method for reconfiguring subsystems of a device based on signals generated from blocks of code. In step **1100**, a signal from a block of code indicating upcoming power consumption needs is detected. For example, a particularly power-expensive block of code can contain energy hint code to notify PMU **102** of an upcoming increase in power consumption. When the energy hint code is executed, CPU **102** sends a signal to PMU **102**. In step **1102**, PMU **102** determines (e.g., based on a stored policy) which subsystems of a device should be reconfigured based on the signal. For example, PMU **102** can determine that voltage supplied to a processor (e.g., to CPU **106** and/or video processor **108**) should be increased or that the frequency of the processor should be increased. PMU **102** can also determine to reconfigure memory (e.g., cache memory **202** or memories of external memory subsystem **110**) based on predicted power consumption needs. Additionally, in an embodiment, PMU **102** can instruct two-phase switchers to forgo operating at very high switching frequencies unless an energy hint is received by PMU **102** indicating that a program will require a large amount of power. Finally, in step **1104**, PMU **102** reconfigures the subsystems to compensate for the upcoming power consumption needs of the block of code.

[0099] In an embodiment, PMU **102** can use energy hints to assist in the reconfiguration of subsystems and applications when performing the methods of FIGS. **4-10**. For example, in the method of FIG. **6**, when PMU **102** identifies one or more subsystems and/or applications to reconfigure in step **604** based on the power prioritization policy in step **604**, PMU **102** can use energy hints to determine whether CPU **106** and/or video processor **108** should be assigned more power (e.g., if a particularly power-expensive block of code is about to be executed).

9. EXAMPLE COMPUTER SYSTEM IMPLEMENTATION

[0100] It will be apparent to persons skilled in the relevant art(s) that various elements and features of the present disclosure, as described herein, can be implemented in hardware using analog and/or digital circuits, in software, through the execution of instructions by one or more general purpose or special-purpose processors, or as a combination of hardware and software.

[0101] The following description of a general purpose computer system is provided for the sake of completeness. Embodiments of the present disclosure can be implemented in hardware, or as a combination of software and hardware. Consequently, embodiments of the disclosure can be implemented in the environment of a computer system or other processing system. An example of such a computer system **1200** is shown in FIG. **12**. At least portions of some, or all, of the modules depicted in FIGS. **1-3** (e.g., PMU **102**, CPU **106**, video processor **108**, subsystems **110-114**, etc.), can be implemented using one or more distinct computer systems **1200**. Furthermore, each of the steps of the flowcharts depicted in FIGS. **4-11** can be implemented on one or more distinct computer systems **1200**.

[0102] Computer system **1200** includes one or more processors, such as processor **1204**. Processor **1204** can be a special purpose or a general purpose digital signal processor. Processor **1204** is connected to a communication infrastructure **1202** (for example, a bus or network). Various software implementations are described in terms of this exemplary computer system. After reading this description, it will become apparent to a person skilled in the relevant art(s) how to implement the disclosure using other computer systems and/or computer architectures.

[0103] Computer system **1200** also includes a main memory **1206**, preferably random access memory (RAM), and can also include a secondary memory **1208**. Secondary memory **1208** can include, for example, a hard disk drive **1210** and/or a removable storage drive **1212**, representing a floppy disk drive, a magnetic tape drive, an optical disk drive, or the like. Removable storage drive **1212** reads from and/or writes to a removable storage unit **1216** in a well-known manner. Removable storage unit **1216** represents a floppy disk, magnetic tape, optical disk, or the like, which is read by and written to by removable storage drive **1212**. As will be appreciated by persons skilled in the relevant art(s), remov-

able storage unit **1216** includes a computer usable storage medium having stored therein computer software and/or data.

[0104] In alternative implementations, secondary memory **1208** can include other similar means for allowing computer programs or other instructions to be loaded into computer system **1200**. Such means can include, for example, a removable storage unit **1218** and an interface **1214**. Examples of such means can include a program cartridge and cartridge interface (such as that found in video game devices), a removable memory chip (such as an EPROM, or PROM) and associated socket, a thumb drive and USB port, and other removable storage units **1218** and interfaces **1214** which allow software and data to be transferred from removable storage unit **1218** to computer system **1200**.

[0105] Computer system **1200** can also include a communications interface **1220**. Communications interface **1220** allows software and data to be transferred between computer system **1200** and external devices. Examples of communications interface **1220** can include a modem, a network interface (such as an Ethernet card), a communications port, a PCMCIA slot and card, etc. Software and data transferred via communications interface **1220** are in the form of signals which can be electronic, electromagnetic, optical, or other signals capable of being received by communications interface **1220**. These signals are provided to communications interface **1220** via a communications path **1222**. Communications path **1222** carries signals and can be implemented using wire or cable, fiber optics, a phone line, a cellular phone link, an RF link and other communications channels.

[0106] As used herein, the terms "computer program medium" and "computer readable medium" are used to generally refer to tangible storage media such as removable storage units **1216** and **1218** or a hard disk installed in hard disk drive **1210**. These computer program products are means for providing software to computer system **1200**.

[0107] Computer programs (also called computer control logic) are stored in main memory **1206** and/or secondary memory **1208**. Computer programs can also be received via communications interface **1220**. Such computer programs, when executed, enable the computer system **1200** to implement the present disclosure as discussed herein. In particular, the computer programs, when executed, enable processor **1204** to implement the processes of the present disclosure, such as any of the methods described herein. Accordingly, such computer programs represent controllers of the computer system **1200**. Where the disclosure is implemented using software, the software can be stored in a computer program product and loaded into computer system **1200** using removable storage drive **1212**, interface **1214**, or communications interface **1220**.

[0108] In another embodiment, features of the disclosure are implemented primarily in hardware using, for example, hardware components such as application-specific integrated circuits (ASICs) and gate arrays. Implementation of a hardware state machine so as to perform the functions described herein will also be apparent to persons skilled in the relevant art(s).

10. CONCLUSION

[0109] It is to be appreciated that the Detailed Description, and not the Abstract, is intended to be used to interpret the claims. The Abstract may set forth one or more but not all exemplary embodiments of the present disclosure as contemplated by the inventor(s), and thus, is not intended to limit the present disclosure and the appended claims in any way.

[0110] The present disclosure has been described above with the aid of functional building blocks illustrating the implementation of specified functions and relationships thereof. The boundaries of these functional building blocks have been arbitrarily defined herein for the convenience of the description. Alternate boundaries can be defined so long as the specified functions and relationships thereof are appropriately performed.

[0111] The foregoing description of the specific embodiments will so fully reveal the general nature of the disclosure that others can, by applying knowledge within the skill of the art, readily modify and/or adapt for various applications such specific embodiments, without undue experimentation, without departing from the general concept of the present disclosure. Therefore, such adaptations and modifications are intended to be within the meaning and range of equivalents of the disclosed embodiments, based on the teaching and guidance presented herein. It is to be understood that the phraseology or terminology herein is for the purpose of description and not of limitation, such that the terminology or phraseology of the present specification is to be interpreted by the skilled artisan in light of the teachings and guidance.

[0112] The representative signal processing functions described herein (e.g. channel and source decoders, etc.) can be implemented in hardware, software, or some combination thereof. For instance, the signal processing functions can be implemented using computer processors, computer logic, application specific circuits (ASIC), digital signal processors, etc., as will be understood by those skilled in the art based on the discussion given herein. Accordingly, any processor that performs the signal processing functions described herein is within the scope and spirit of the present invention.

[0113] The above systems and methods may be implemented as a computer program executing on a machine, as a computer program product, or as a tangible and/or non-transitory computer-readable medium having stored instructions. For example, the functions described herein could be embodied by computer program instructions that are executed by a computer processor or any one of the hardware devices listed above. The computer program instructions cause the processor to perform the signal processing functions described herein. The computer program instructions (e.g. software) can be stored in a tangible non-transitory computer usable medium, computer program medium, or any storage medium that can be accessed by a computer or processor. Such media include a memory device such as a RAM or ROM, or other type of computer storage medium such as a computer disk or CD ROM. Accordingly, any tangible non-transitory computer storage medium having computer program code that cause a processor to perform the signal processing functions described herein are within the scope and spirit of the present disclosure.

[0114] While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example only, and not limitation. It will be apparent to persons skilled in the relevant art that various changes in form and detail can be made therein without departing from the spirit and scope of the invention. Thus, the breadth and scope of the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What is claimed is:

1. An apparatus comprising:

a power source;

a plurality of subsystems coupled to the power source, wherein the subsystems are configured to receive power from the power source; and

a power management unit (PMU) configured to manage distribution of power from the power source to the subsystems, wherein the PMU is further configured to:

receive information from the subsystems indicating power requirements of the subsystems;

predict power requirements for a first subsystem in the plurality of subsystems based on the information; and

reconfigure the first subsystem based on the predicted power requirements.

2. The apparatus of claim **1**, wherein the subsystems include a sensor, and wherein the PMU is further configured to:

receive information from the sensor indicating the occurrence of an event; and

determine whether to reconfigure the first subsystem or a second subsystem in the plurality of subsystems based on the event.

3. The apparatus of claim **2**, wherein the information indicates that no user is in the proximity of the apparatus, and wherein the PMU is further configured to reduce an amount of power supplied to the first subsystem in response to determining that no user is in the proximity of the apparatus.

4. The apparatus of claim **1**, wherein the subsystems include:

a first memory; and

a processor having a cache memory, and wherein the PMU is further configured to:

monitor a plurality of cache miss signals received from the processor;

detect an increase or a decrease in an amount of cache miss signals associated with data stored in the first memory; and

determine whether to reconfigure the first memory or the cache memory based on the detected increase or decrease.

5. The apparatus of claim **4**, wherein the PMU is further configured to:

increase an amount of power supplied to the first memory in response to detecting the increase in the amount of cache miss signals associated with data stored in the first memory; and

decrease the amount of power supplied to the first memory in response to detecting the decrease in the amount of cache miss signals associated with data stored in the first memory.

6. The apparatus of claim **1**, wherein the PMU is further configured to:

generate a signature based on a pattern of access associated with a block of code;

access power consumption information stored in a table in memory;

determine whether an entry in the table matches the signature;

access power prediction information corresponding to the signature in response to determining that the entry matches the signature; and

determine whether to reconfigure the first subsystem or a second subsystem in the plurality of subsystems based on the power prediction information.

7. The apparatus of claim **6**, wherein the signature is a hash of a plurality of addresses in memory accessed during execution of the block of code.

8. The apparatus of claim **1**, wherein the PMU is further configured to:

detect a signal generated during an execution of a block of code, wherein the signal indicates upcoming power consumption needs of the block of code; and

determine whether to reconfigure the first subsystem or a second subsystem in the plurality of subsystems based on the signal.

9. The apparatus of claim **8**, wherein the block of code includes energy hint code for notifying the PMU that power-expensive code is being executed, and wherein the signal is generated during execution of the energy hint code.

10. A communications device comprising:

a power source;

a plurality of subsystems configured to receive power from the power source; and

a power management unit (PMU) configured to manage distribution of power from the power source to the subsystems, wherein the PMU is further configured to:

receive information from the subsystems indicating power requirements of a plurality of applications;

access information stored after a previous execution of a first application in the plurality of applications;

predict power requirements for the first application based on the accessed information; and

reconfigure characteristics of the first application based on the predicted power requirements.

11. The communications device of claim **10**, wherein the PMU is further configured to:

determine whether the communications device has sufficient power to finish executing the application; and

reconfigure characteristics of the first application in response to determining that the communications device does not have sufficient power to finish executing the application.

12. The communications device of claim **11**, wherein the PMU is further configured to:

access a stored policy;

determine whether the policy contains instructions for reconfiguring the characteristics of the first application;

in response to determining that the policy contains instructions for reconfiguring the characteristics of the first application:

notify a user of the communications device of an upcoming reconfiguration of the characteristics of the first application, and

reconfigure the characteristics of the first application based on the stored policy; and

in response to determining that the policy does not contain instructions for reconfiguring the characteristics of the first application:

inform the user that the communications device has insufficient power to finish executing the application, and

propose a power saving option to the user.

13. The communications device of claim **10**, wherein the PMU is further configured to:

determine a power mode of the communications device;

in response to determining that the communications device is operating in a lower power mode, periodically evaluate currently executing applications to determine whether the first application is a high priority application; and

in response to determining that the first application is a high priority application:

determine whether the communications device has sufficient power to finish executing the application, and

reconfigure characteristics of the first application in response to determining that the communications device does not have sufficient power to finish executing the application.

14. The communications device of claim **10**, wherein the PMU is further configured to lower quality of a playing video in response to determining that the communications device has insufficient power to finish playing the video.

15. The communications device of claim **10**, wherein the PMU is further configured to:

determine an amount of power available to a second device in communication with the communications device; and

reconfigure characteristics of the first application based on the amount of power available to the second device.

16. A method comprising:

receiving, at a power management unit of a device, information from a plurality of subsystems of the device indicating power requirements of the subsystems;

predicting power requirements for a first subsystem in the plurality of subsystems based on the information; and

reconfiguring the first subsystem based on the predicted power requirements.

17. The method of claim **16**, further comprising:

receiving information from a sensor indicating the occurrence of an event; and

determining whether to reconfigure the first subsystem or a second subsystem in the plurality of subsystems based on the event.

18. The method of claim **16**, further comprising:

monitoring a plurality of cache miss signals received from a processor having a cache memory;

detecting an increase or a decrease in an amount of cache miss signals associated with data stored in a first memory; and

determining whether to reconfigure the first memory or the cache memory based on the detected increase or decrease.

19. The method of claim **16**, further comprising:

generating a signature based on a pattern of access associated with a block of code;

accessing power consumption information stored in a table in memory;

determining whether an entry in the table matches the signature;

accessing power prediction information corresponding to the signature in response to determining that the entry matches the signature; and

determining whether to reconfigure the first subsystem or a second subsystem in the plurality of subsystems based on the power prediction information.

20. The method of claim **16**, further comprising:

detecting a signal generated during an execution of a block of code, wherein the signal indicates upcoming power consumption needs of the block of code; and

determining whether to reconfigure the first subsystem or a second subsystem in the plurality of subsystems based on the signal.

* * * * *