Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2795489 C 2018/01/23

(11)(21) 2 795 489

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) Date de dépôt PCT/PCT Filing Date: 2010/09/22
(87) Date publication PCT/PCT Publication Date: 2011/10/13
(45) Date de délivrance/Issue Date: 2018/01/23
(85) Entrée phase nationale/National Entry: 2012/10/04
(86) N° demande PCT/PCT Application No.: US 2010/049816
(87) N° publication PCT/PCT Publication No.: 2011/126499
(30) Priorité/Priority: 2010/04/07 (US61/321,616)

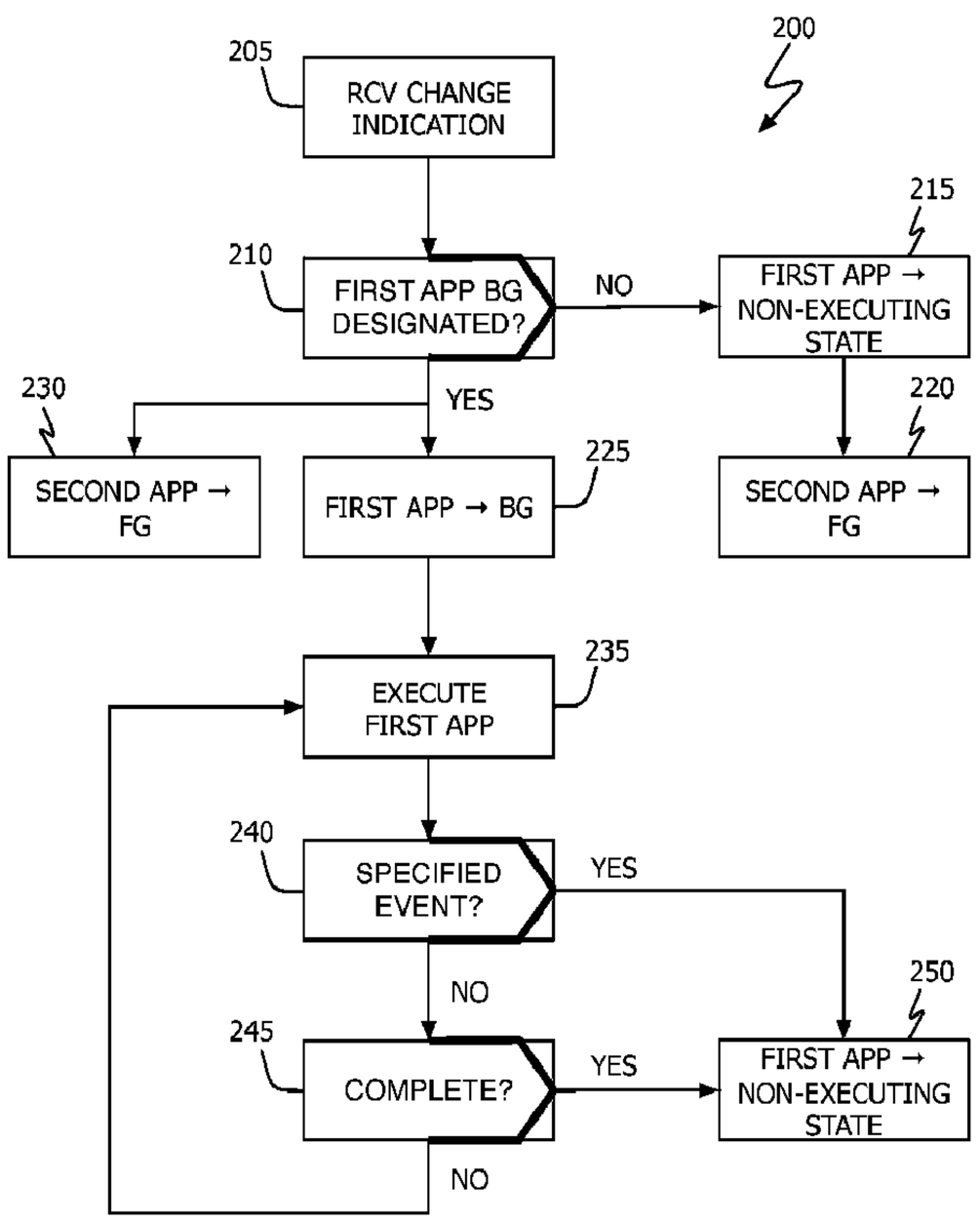
(51) Cl.Int./Int.Cl. *G06F 9/48* (2006.01), *G06F 9/46* (2006.01), *G06F 9/54* (2006.01)

(72) Inventeurs/Inventors:
ALBERT, ERIC, US;
BALLARD, LUCIA, US;
CHAPMAN, GREGORY R., US;
CRANE, NEIL GARETH, US;
DUFFY, THOMAS BROGAN, US;
FORSTALL, SCOTT, US;
...

(73) Propriétaire/Owner:
APPLE INC., US

(74) Agent: RICHES, MCKENZIE & HERBERT LLP

(54) Titre : TRAITEMENT MULTITACHE OPPORTUNISTE
(54) Title: OPPORTUNISTIC MULTITASKING



(57) Abrégé/Abstract:
Services for a personal electronic device are provided through which a form of background processing or multitasking is supported. The disclosed services permit user applications to take advantage of background processing without significant

CA 2795489 C 2018/01/23

(11)(21) 2 795 489

(13) C

(72) Inventeurs(suite)/Inventors(continued): FREEDMAN, GORDON J., US; GOODWIN, DAVID WILLIAM, US; IAROCCI, JOHN, US; LITZINGER, DARREN, US; MARCELLINO, CHRISTOPHER, US; MASON, HENRY GRAHAM, US; MASPUTRA, CAHYA, US; MYSZEWSKI, DAVID, US; NOVICK, GREGORY, US; ROTHERT, CURTIS, US; SCHREYER, RICHARD, US; SOKOL, JOSEPH, US; SORRESSO, DAMIEN, US; SRISUWANANUKORN, CHARLES, US; VAN MILLIGAN, MICHAEL, US; WASTON, MATTHEW G., US

(57) Abrégé(suite)/Abstract(continued):
negative consequences to a user's experience of the foreground process or the personal electronic device's power resources. To effect the disclosed multitasking, one or more of a number of operational restrictions may be enforced. A consequence of such restrictions may be that a process will not be able to do in the background state, what it may be able to do if it were in the foreground state. Implementation of the disclosed services may be substantially transparent to the executing user applications and, in some cases, may be performed without the user application's explicit cooperation

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 October 2011 (13.10.2011)

PCT

(10) International Publication Number
WO 2011/126499 A3

(51) International Patent Classification:
G06F 9/48 (2006.01) *G06F 9/54* (2006.01)
G06F 9/46 (2006.01)

(21) International Application Number:
PCT/US2010/049816

(22) International Filing Date:
22 September 2010 (22.09.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/321,616 7 April 2010 (07.04.2010) US

(71) Applicant *(for all designated States except US)*: **APPLE INC.** [US/US]; 1 Infinite Loop, Cupertino, CA 95014 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **ALBERT, Eric** [US/US]; 2 Infinite Loop, Ms 302-1ns, Cupertino, CA 95014 (US). **BALLARD, Lucia** [US/US]; 2 Infinite Loop, MS 302-4K, Cupertino, CA 95014 (US). **CHAPMAN, Gregory, R.** [US/US]; 2 Infinite Loop, MS 302-3KS, Cupertino, CA 95014 (US). **CRANE, Neil, Gareth** [GB/US]; 2 Infinite Loop, MS 302-4K, Cupertino, CA 95014 (US). **DUFFY, Thomas, Brogan** [US/US]; 2 Infinite Loop, MS 302-4K, Cupertino, CA 95014 (US). **FORSTALL, Scott** [US/US]; 2 Infinite Loop, MS 302-1NS, Cupertino, CA 95014 (US). **FREEDMAN, Gordon, J.** [US/US]; 2 Infinite Loop, MS 302-1NS, Cupertino, CA 95014 (US). **GOODWIN, David, William** [US/US]; 2 Infinite Loop, MS 302-1NS, Cupertino, CA 95014 (US). **IAROCCI, John** [US/US]; 2 Infinite Loop, MS 301-2COS, Cupertino, CA 95014 (US). **LITZINGER, Darren** [US/US]; 3 Infinite Loop, MS 303-1NS, Cupertino, CA 95014 (US). **MARCELLINO, Christopher** [US/US]; 1 Infinite Loop, MS 301-2COS, Cupertino, CA 95014 (US). **MASON, Henry, Graham** [US/US]; 2 Infinite Loop, MS 302-1NS, Cupertino, CA 95014 (US). **MASPUTRA, Cahya** [ID/US]; 2 Infinite Loop, MS 302-4K, Cupertino, CA 95014 (US).

[Continued on next page]

(54) Title: OPPORTUNISTIC MULTITASKING

(57) Abstract: Services for a personal electronic device are provided through which a form of background processing or multitasking is supported. The disclosed services permit user applications to take advantage of background processing without significant negative consequences to a user's experience of the foreground process or the personal electronic device's power resources. To effect the disclosed multitasking, one or more of a number of operational restrictions may be enforced. A consequence of such restrictions may be that a process will not be able to do in the background state, what it may be able to do if it were in the foreground state. Implementation of the disclosed services may be substantially transparent to the executing user applications and, in some cases, may be performed without the user application's explicit cooperation

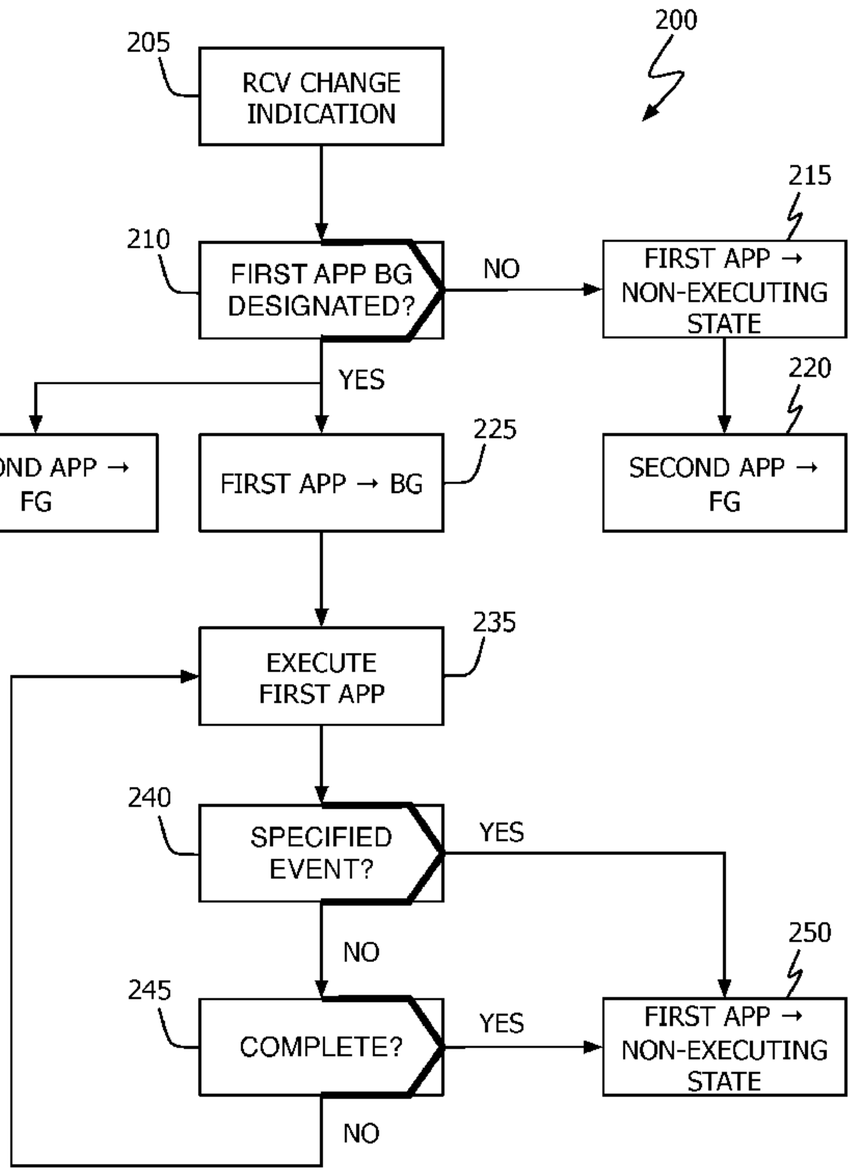


FIG. 2

WO 2011/126499 A3

WO 2011/126499 A3

MYSZEWSKI, David [US/US]; 2 Infinite Loop, MS 302-3NS, Cupertino, CA 95014 (US). **NOVICK, Gregory** [US/US]; 2 Infinite Loop, MS 302-1NS, Cupertino, CA 95014 (US). **ROTHERT, Curtis** [US/US]; 2 Infinite Loop, MS 302-1NS, Cupertino, CA 95014 (US). **SCHREYER, Richard** [US/US]; 2 Infinite Loop, MS 302-3KS, Cupertino, CA 95014 (US). **SOKOL, Joseph** [US/US]; 1 Infinite Loop, MS 301-2COS, Cupertino, CA 95014 (US). **SORRESSO, Damien** [US/US]; 1 Infinite Loop, MS 301-2COS, Cupertino, CA 95014 (US). **SRISUWANANUKORN, Charles** [US/US]; 2 Infinite Loop, MS 302-3NS, Cupertino, CA 95014 (US). **VAN MILLIGAN, Michael** [US/US]; 1 Infinite Loop, MS 302-4K, Cupertino, CA 95014 (US). **WASTON, Matthew, G.** [US/US]; 2 Infinite Loop, MS 302-1NS, Cupertino, CA 95014 (US).

(74) **Agent**: **MILES, Coe, F.**; Wong, Cabello, Lutsch, Rutherford & Brucculeri LLP, 20333 Sh 249, Suite 600, Houston, TX 77070 (US).

(81) **Designated States** *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) **Date of publication of the international search report**:
19 April 2012

OPPORTUNISTIC MULTITASKING

[0001] The invention to be disclosed and claimed in this application was prematurely and without Apple's authorization disclosed to the public when a prototype of Apple's iPhone 4 was apparently stolen from an Apple engineer on March 25, 2010. The U.S. priority application, on which this application is based, was not filed before the apparent theft.

Cross Reference to Related Applications

[0002] This is an international application based on U.S. Provisional Application Serial No. 61/321,616 entitled "Opportunistic Multitasking" filed April 7, 2010.

Background

[0003] This disclosure relates generally to the field of computer science. More particularly, this disclosure relates to a technique for improving the user experience and power management in a personal electronic device.

[0004] Power constrained, hand-held devices (*e.g.*, mobile phones, personal entertainment devices, and electronic pad computers) are resource limited compared to larger, fixed, systems such as desk-top, workstation and notebook computers. In such systems, the computational power available simply cannot support the execution of a large number of concurrent processes/threads without significantly degrading the user experience and consuming the device's limited power resources. In light of this recognition, system designers for these types of devices have not traditionally supported multitasking at the user application level. While this approach has the benefit of minimizing the drain on the device's limited power resources, it also limits the ability to provide the user (via a user application) with an interactive environment.

Summary

[0005] Services for a personal electronic device are provided through which a form of background processing or multitasking is supported. The disclosed services permit user applications to take advantage of background processing without significant negative consequences to a user's experience of the foreground process or the personal electronic device's power resources. Implementation of the disclosed

services may be substantially transparent to the executing user applications and, in some cases, may be performed without the user application's explicit cooperation. To effect the disclosed multitasking, a number of operational restrictions may be enforced. A consequence of such restrictions may be that a process will not be able to do in background, what it may be able to do if it were in the foreground.

[0006] In one service, a foreground user application is transitioned to a non-executing state as it is moved out of the foreground state. In another service, a background process is given a maximum amount of time to complete a task before it is transitioned to a non-executing state. In still another service, audio applications (*e.g.*, user applications generating or recording audio signals) are permitted to execute in background until, and unless, they are paused by a user. In yet another service, communication socket(s) instantiated for a user application may be maintained even if the user application instantiating them is placed in a non-executing state. One illustrative type of application that can take advantage of this service is voice over Internet protocol (VOIP) user applications. In still other embodiments, user applications are permitted to receive notifications (*e.g.*, location events) when in the non-executing state. Each of the disclosed services relies on or uses one or more restrictions that, in operation, can interfere with or prevent full-time multitasking operations. That is, the disclosed services permit multitasking only when it will not significantly interfere with foreground processes or unduly utilize the personal electronic device's power.

[0006a] Accordingly, in one aspect, the present invention provides a multitasking method for an electronic device, comprising: executing, on an electronic device, a task of a first user application in a first foreground state; receiving, on the electronic device, a user request to execute a second user application in a second foreground state, wherein the second user application is in a first background state when receiving the user request; receiving a declaration from the first user application executing the task in the first foreground state whether or not the first user application is capable of continuing execution of the task in a second background state, wherein the declaration is received at a run time of the first user application in code from the first user application; in response to receiving the user request and receiving the declaration, moving the first user application from the first foreground state to the second background state while continuing the execution of the task of

the first user application, and moving the second user application from the first background state to the second foreground state; executing, on the electronic device, the second user application in the second foreground state; based upon receiving the declaration, moving the first user application from the first foreground state to the second background state and moving the second user application from the first background state to the second foreground state, placing the first user application in an executing state in the second background state to complete the task began while the first user application was in the first foreground state; and placing the first user application in a non-executing state upon one of (i) completion of the execution of the task by the first user application while in the executing state and (ii) an attempt by the first user application to access a specific hardware resource while in the executing state.

[0006b] In a further aspect, the present invention provides a personal electronic device, comprising: memory; computer program instructions, stored in the memory; and a programmable control device operatively coupled to the memory and configured to execute the computer program instructions to cause the programmable control device to: execute, on the personal electronic device, a task of a first user application in a first foreground state; receive, on the personal electronic device, a user request to execute a second user application in a second foreground state, wherein the second user application is in a first background state when receiving the user request; receive a declaration from the first user application executing the task in the first foreground state whether or not the first user application is capable of continuing execution of the task in a second background state, wherein the declaration is received at a run time of the first user application in code from the first user application; in response to receiving the user request and receiving the declaration, moving the first user application from the first foreground state to the second background state while continuing the execution of the task of the first user application, and moving the second user application from the first background state to the second foreground state; execute, on the personal electronic device, the second user application in the second foreground state, and based upon receiving the declaration, moving the first user application from the first foreground state to the second background state, and moving the second user application from the first background state to the second foreground state, placing the first user application in

an non-executing state upon one of (i) completion of the execution of the task by the first user application while in an executing state and (ii) an attempt by the first user application to access a specific hardware resource while in the executing state.

[0006c] In a further aspect, the present invention provides a multitasking method for a personal electronic device, comprising: executing, on a personal electronic device, an audio operation of a first user application in a first foreground state; receiving, on the personal electronic device, a user request to execute a second user application in a second foreground state, wherein the second user application is in a first background state when receiving the user request to execute the second user application; receiving a declaration from the first user application executing the audio operation in the first foreground state whether or not the audio operation is capable of continuing execution in a second background state, wherein the declaration is received at a run time of the first user application in code from the first user application; in response to receiving the user request to execute the second user application and receiving the declaration, moving the first user application from the first foreground state to the second background state while continuing the execution of the audio operation of the first user application, and moving the second user application from the first background state to the second foreground state; executing, on the personal electronic device, the second user application in the second foreground state; receiving, on the personal electronic device, a user request to pause the audio operation; and placing, in response to receiving the user request to pause the audio operation, the first user application into a non-executing state.

[0006d] In yet a further aspect, the present invention provides a personal electronic device, comprising: memory; computer program instructions stored in the memory; a programmable control device operatively coupled to the memory, said programmable control device configured to execute computer program instructions to: executing, on a personal electronic device, an audio operation of a first user application in a first foreground state; receive, on the personal electronic device, a user request to execute a second user application in a second foreground state, wherein the second user application is in a first background state when receiving the user request to execute the second user application; receive a declaration from the first user application executing the audio operation in the first foreground state whether or not the audio operation is capable of continuing execution in a second

background state, wherein the declaration is received at a run time of the first user application in code from the first user application; in response to receipt of the user request to execute the second user application and receipt of the declaration, moving the first user application from the first foreground state to the second background state while continuing the execution of the audio operation of the first user application, and moving the second user application to the second foreground state; execute, on the personal electronic device, the second user application in the second foreground state; receive, on the personal electronic device, a user request to pause the audio operation; and place, in response to receipt of the request to pause the audio operation, the first user application into a non-executing state.

[0006e] In yet a further aspect, the present invention provides a multitasking method, comprising: initiating a first user Voice Over Internet Protocol (VOIP) application at a first time; instantiating one or more communication sockets for the first user VOIP application; placing the first user VOIP application into a non-executing state at a second time; after placing the first user VOIP application into a non-executing state at the second time, receiving an automatic notification associated with the one or more communication sockets for the first user VOIP application at a third time; in response to receiving the automatic notification associated with the one or more communication sockets for the first user VOIP application at the third time, placing the first user VOIP application from the non-executing state into a background state; and after the first user VOIP application has responded to the automatic notification associated with the one or more communication sockets for the first user VOIP application at the third time, returning the first user VOIP application to the non-executing state, wherein the first time precedes the second time and the second time precedes the third time, and wherein the one or more communication sockets for the first user VOIP application are maintained during the first, second, and third times.

[0006f] In yet a further aspect, the present invention provides a personal hand-held electronic device, comprising: a memory; a receiver operatively coupled to the memory; a transmitter operatively coupled to the memory; and a programmable control device operatively coupled to the memory and configured to execute instructions stored therein, said instructions for causing the programmable control device to: initiating a first user Voice Over Internet Protocol (VOIP) application at a

first time; instantiating one or more communication sockets for the first user VOIP application; placing the first user VOIP application into a non-executing state at a second time; after placing the first user VOIP application into a non-executing state at the second time, receiving an automatic notification associated with the one or more communication sockets for the first user VOIP application at a third time; in response to receiving the automatic notification associated with the one or more communication sockets for the first user VOIP application at the third time, placing the first user VOIP application from the non-executing state into a background state; and after the first user VOIP application has responded to the automatic notification associated with the one or more communication sockets for the first user VOIP application at the third time, returning the first user VOIP application to the non-executing state, wherein the first time precedes the second time and the second time precedes the third time, and wherein the one or more communication sockets for the first user VOIP application are maintained during the first, second, and third times.

[0006g] In yet a further aspect, the present invention provides a hand-held digital device multitasking method, comprising: instantiating, at a first time, at least one Voice Over Internet Protocol (VOIP) socket for a first user application; placing, at a second time and after instantiating, at the first time, at least one VOIP socket for the first user application, the first user application into a suspended state; receiving, at a third time and after placing the first user application into the suspended state, a message for the first user application; placing, at a fourth time and after receiving the message for the first user application, the first user application from the suspended state into a background state; delivering, at a fifth time and after placing the first user application from the suspended state into a background state, the message to the first user application; receiving, at a sixth time and after delivering the message to the first user application, an automatic notification from the first user application that the first user application has responded to the message; and placing, at a seventh time and after receiving automatic notification from the first user application that the first user application has responded to the message, the first user application into the suspended state, wherein the at least one Voice Over Internet Protocol (VOIP) socket for the first user application is maintained during the first, second, third, fourth, fifth, sixth, and seventh times.

[0006h] In yet a further aspect, the present invention provides a method, comprising: scheduling a first foreground process with a scheduling priority from a first band of priority levels; scheduling, based on the scheduling a first foreground process with a scheduling priority from a first band of priority levels, a first background process with a scheduling priority from a second band of priority levels; associating, with at least one of the first foreground process and the first background process, an allotted quantum of one or more specified system resources; and adjusting the scheduling priority of one of the first foreground process and the first background process in response to one of the first foreground process and the first background process using a full amount of the allotted quantum of one or more specified system resources by the one of the first foreground process and the first background process, wherein the priority levels in the first band and the second band overlap by one, but not all, priority levels.

[0006i] In yet a further aspect, the present invention provides a personal electronic device, comprising: memory; computer program instructions, stored in the memory; and a programmable control device operatively coupled to the memory and configured to execute the computer program instructions to cause the programmable control device to perform operations of: scheduling a first foreground process with a scheduling priority from a first band of priority levels; scheduling, based on the scheduling a first foreground process with a scheduling priority from a first band of priority levels, a first background process with a scheduling priority from a second band of priority levels; associating, with at least one of the first foreground process and the first background process, an allotted quantum of one or more specified system resources; and adjusting the scheduling priority of one of the first foreground process and the first background process in response to one of the first foreground process and the first background process using a full amount of the allotted quantum of one or more specified system resources by the one of the first foreground process and the first background process, wherein the priority levels in the first band and the second band overlap by one, but not all, priority levels.

[0006j] Further aspects of the invention will become apparent upon reading the following detailed description and drawings, which illustrate the invention and preferred embodiments of the invention.

Brief Description of the Drawings

[0007] Figure **1** shows an illustrative thread scheduling scheme in accordance with one embodiment.

[0008] Figure **2** shows, in flow chart form, a Task Completion service operation in accordance with one embodiment.

[0009] Figure **3** shows, in flow chart form, an Audio service operation in accordance with one embodiment.

[0010] Figure **4** shows, in flow chart form, a Network service operation in accordance with one embodiment.

[0011] Figure **5** shows, in block diagram form, a personal electronic device in accordance with one embodiment.

[0012] Figure **6** shows, in block diagram form, a personal electronic device in accordance with another embodiment.

[0013] Figure **7** shows, in block diagram form, radio and location processing elements in accordance with one embodiment.

[0014] Figure **8** shows, in block diagram form, a personal electronic device in accordance with one embodiment.

Detailed Description

[0015] Services for a personal electronic device are provided through which a limited form of background processing or multitasking is supported. Use of one or more of the disclosed services, in combination with disclosed operating restrictions, permit user applications to execute in the background to give the user a robust interactive environment with little impact on the device's power resources. In some embodiments, background processing may be limited to the completion of a specific task, a specified amount of time (*e.g.,* 5-30 minutes) or a particular type of task (*e.g.,* audio operations). In other embodiments a user application may cease operation after notifying the operating system that it may be reanimated on the occurrence of one or more specified events. On reanimation, the application may perform additional operations in background. Implementation of the disclosed services may be substantially transparent to the executing user applications and, in some cases, may be performed without the user application's explicit cooperation.

[0016] As used herein, the phrase "personal electronic device" is a portable digital device such as a mobile phone, a personal digital assistant, a hand-held entertainment device, a pad or tablet computer system or a set top box *(e.g.,* an

Apple TV® or cable converter box). (APPLE TV is a registered trademark of Apple Inc.) As used herein, the term "service" refers to a utility, function or program code module that performs some task for a calling process, which has no user interface and which is accessed programmatically through a call interface such as an Application Programming Interface (API).

[0017] In the following description, for purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the invention. It will be apparent to one skilled in the art, however, that the invention may be practiced without these specific details. In other instances, structures and devices are shown in block diagram form in order to avoid obscuring the invention. References to numbers without subscripts are understood to reference all instance of subscripts corresponding to the referenced number. Moreover, the language used in this disclosure has been principally selected for readability and instructional purposes, and may not have been selected to delineate or circumscribe the inventive subject matter, resort to the claims being necessary to determine such inventive subject matter. Reference in this specification to "one embodiment" or to "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiments is included in at least one embodiment of the invention, and multiple references to "one embodiment" or "an embodiment" should not be understood as necessarily all referring to the same embodiment.

[0018] It will be appreciated that in the development of any actual implementation numerous programming and component decisions must be made to achieve the developers' specific goals (*e.g.,* compliance with system- and business-related constraints), and that these goals will vary from one implementation to another. It will also be appreciated that such development effort might be complex and time-consuming, but would nevertheless be a routine undertaking for those of ordinary skill in the personal computational device development field having the benefit of this disclosure.

[0019] To effect background processing or multitasking without significant negative consequences to a user's experience of the foreground process or the personal electronic device's power resources, a number of operational restrictions

may be enforced. A consequence of such restrictions may be that a process will not be able to do in background, what it may be able to do if it were in the foreground.

[0020] As used herein, the phrase "foreground process" means that process that currently has access to system resources (*e.g.,* the platform's central processing unit and graphical processing unit) and presents a user interface (UI) or graphical user interface (GUI) to a user. (Accordingly, the "foreground" is a state in which a foreground process executes.) In contrast, a "background process" is a process that may be scheduled to access system resources but which does not currently present a UI/GUI to a user. (Accordingly, the "background" is a state in which a background process executes.) As used herein, the term "process" means a user application. A user application, in turn, is an executable code module(s) that is capable of presenting a UI/GUI.

[0021] Operational Restrictions: The following illustrative list of operational restrictions may be enforced to limit the deleterious effect a background process may have on a foreground process.

[0022] 1. Processor Scheduling. Foreground and background processes may be assigned different processor scheduling "priorities" in such a way that background processes do not interfere with foreground processes. One of ordinary skill in the art will recognize that in operational environments such as UNIX® and Mach, processes are not scheduled, threads are. (UNIX is a registered trademark of the American Telephone and Telegraph Company.) It will further be recognized that a thread is that portion of a program, application or process that can run independently of, and concurrently with, other portions of the program, application or process.

[0023] Referring to FIG. **1**, in one embodiment background threads may be assigned a priority from a band of priorities that overlaps in part with those priority levels assigned to foreground threads. The overlapping priority bands increase the likelihood that a background process/thread will make progress (*i.e.,* execute) even when there is a foreground process/thread running. In this embodiment, to improve thread responsiveness it has been found beneficial to demote those threads (*i.e.,* reduce their scheduling priority) that use their entire quantum without blocking for

an input/output (I/O) operation. In one embodiment, a thread's priority is reduced by a single count every time it uses its full allotted quantum without blocking. (It will be recognized that a processes use of a full quantum is an indicator that the process is using the central processing unit.) In another embodiment, a thread's priority may be reduced by more than one count when this occurs. The amount of "overlap" in priority levels between the background band and the foreground band may be one (1) or more.

[0024] 2. Disk Scheduling: In one embodiment, foreground threads are given priority over background threads when accessing system storage units (*e.g.,* magnetic hard drives and solid state hard disk units). In addition, background processes (*i.e.,* threads associated with processes in the background state) may be rate-limited in their access of system storage units.

[0025] 3. Incoming Network Activity. In one embodiment, incoming network traffic may be moderated by dropping all packets destined for an application in the background or a non-executing state (*e.g.,* suspended). Further, artificially small buffer sizes may be reported to distal network sites so as to throttle incoming TCP (Transmission control protocol) traffic. One of ordinary skill in the art will recognize that TCP has an existing set of traffic flow control mechanisms relating to buffer sizes, so that traffic isn't sent over a network faster than the recipient can handle it. As previously noted, in one embodiment artificially small buffer sizes may be reported so that the sender slows or stops the flow of incoming traffic.

[0026] 4. Outgoing Network Activity. In one embodiment, network access is mediated through the use of two (2) queues; one for jobs associated with foreground processes and one for jobs associated with background processes. It has been found beneficial to give the foreground queue priority over the background queue. By way of example, the foreground queue may be given 100% priority over the background queue. At this setting, the only time a job in the background queue will be serviced is if the foreground queue is empty. In another embodiment, jobs in the foreground queue may be serviced in any desired ratio with those from the background queue (*e.g.,* 90/10, 80/20, 75/27 or 50/50). In still another embodiment, once a job has been placed into a queue (*e.g.,* the foreground queue)

it is not moved if the job's associated process is subsequently placed into a different operating state (*e.g.,* background). In yet another embodiment, a job's queue location is updated to reflect the operational status of its associated process.

[0027] 5. Graphics Operations. In one embodiment such as, for example, the iPhone® Operating System, there are at least two mechanisms through which an application may write or paint to the display screen. (IPHONE is a registered trademark of Apple Inc.) A first mechanism is through the use of specialized hardware such as a graphics processing unit (GPU). A second mechanism is through use of one or more of the platform's central processing units (CPUs). In accordance with one embodiment, in the iPhone operating environment background processes are not permitted access to specialized graphics hardware; for example, any background process attempting to access the GPU may be terminated. By way of another example, background processes attempting to generate viewable graphics via the CPU are ignored – that is, any attempt to execute commands directed to using the CPU to write to a drawing buffer/memory are ignored. In some embodiments, drawing memory associated with a process/application placed into the background state may be marked as "purgeable." Thus, if a foreground process needs more memory it may use (through standard memory allocation techniques) display memory originally granted to a process that is now in the background state.

[0028] 6. Notifications. When a process transitions from the background state to the foreground state, it may be permitted to receive one (1) notification for each type of notification. More particularly, in one embodiment multiple notification events may be consolidated. For example, if a user rotates their personal electronic device (*e.g.,* the iPhone) from portrait presentation to landscape presentation to portrait presentation and then back to landscape presentation, the resulting three (3) orientation notification events may be coalesced into a single notification – the last state, landscape. This can improve an application's response as it does not have to respond to multiple events. Illustrative notification types include orientation change notifications (for example, from accelerometer or gyroscope sensor input), address book database notifications (occurs when an address book entry is modified) and

camera roll notifications (occurs when an image in a shared resource image library has been changed).

[0029] 7. Locks. In one embodiment, any process that has a lock on a shared resource that is going into the suspended state may be terminated (rather than suspended). Illustrative shared resources include specialized graphics hardware and file descriptors. In another embodiment, a shared resource is "taken away" from a user application when it goes into the background state. For example, if an application (process) has a lock on camera hardware, this lock is removed when the user application is placed into the background state. As used herein, the "suspended state" is a condition in which a process is not permitted to schedule threads for execution, but whose state is retained in main memory.

[0030] 8. Hardware Restrictions. In one embodiment background processes are prevented from gaining access to certain system hardware resources. Illustrative hardware not available to background processes include, but are not limited to: camera, GPU, accelerometer, gyroscope, proximity sensor and microphone.

[0031] 9. Memory Management. In order to retain sufficient memory to accommodate executing processes, as the personal electronic device's core memory approaches a specified "critical" point, processes may be removed from memory. In one embodiment, a process so removed may be hibernated. In another embodiment a process so removed may be terminated. As used herein, the phrase "core memory" or "main memory" means a personal electronic device's main random access memory from which user applications execute; a "hibernated process" is a process whose operational state has been written to non-volatile memory so that (1) it cannot schedule threads for execution and (2) its main memory "footprint" is substantially zero; and a "terminated process" is a process whose operational state has been deleted and which, therefore, cannot be executed until restarted. In one embodiment, during the act of hibernation memory which that can be recreated (purgeable) or reloaded (clean) on demand is disposed of; only memory that has been modified or otherwise un-reconstitutable is written to non-volatile memory. In another embodiment, a specified region of non-volatile memory is set aside at boot time for use as a "hibernation" memory. When a hibernated process is reanimated,

its data in non-volatile memory is brought back into main memory, thereby freeing up space in non-volatile storage. In one embodiment, applications not accessed for more than a specified time may be "hibernated."

[0032] Any desired approach for selection of which applications to remove from main memory may be made. For example, the least recently used application may be removed first. Alternatively, the largest currently non-executing application may be removed. In one embodiment, applications selected for removal that are greater than a specified size (*e.g.,* 8 MB, 16 MB or 32 MB) may be terminated while applications less than the specified size may be hibernated. In still another embodiment, background audio user applications may be terminated only after suspended user applications have been terminated, but before foreground user applications.

[0033] Similarly, the "critical" point may be any value the designer wishes. Illustrative critical points include 50%, 75% and 90%. Using 75% as an exemplar critical point, applications would be selected for removal from main memory once the main memory reaches 75% of its capacity – for example, when a 256 megabyte (MB) main memory reaches 192 MB full.

[0034] Operational States: As an aid in understanding how the above restrictions are applied to the various services described below, Table 1 identifies and describes five (5) operational states for a user process executing on a personal electronic device.

Table 1. Operational States

Mode or State	Description	Restrictions
Foreground	A process in this state has access to system resources and currently presents a user interface to a user.	None.
Background	A process in this state has access to system resources but does not currently present a user interface to a user.	Real-time processes are subject to real-time restrictions (*i.e.,* they are demoted to non-real-time priorities when placed into background – except for audio and VOIP/network communication's applications); all other processes are subject to all other restrictions.
Suspended	A process in this state has its operational state in main memory and could be scheduled for execution except for the fact that it has been suspended.	Not eligible for execution.
Hibernated	A process in this state has had its operational state written to non-volatile memory. Only that information necessary to identify, locate and reanimate the process is retained in main memory.	Not eligible for execution.
Terminated	A process whose operational state is no longer accessible in main memory.	Not eligible for execution.

As used herein, the term "non-executing state" means one of the suspended, hibernated or terminated states.

[0035] Illustrative Services: In the following, the "environment" within which the described services operate is the iPhone operating system. This is illustrative only, the disclosed services may be equally applicable to any personal electronic device.

[0036] 1. Fast Task Switching Service. In the past, when an application left the foreground state in the iPhone operating system, it was simply terminated. The Fast-Task Switching service permits an application to receive notification that it is leaving the foreground state and, at that time, is given an opportunity to do some processing (*e.g.,* a few seconds). After this time, the application is suspended (*i.e.,* placed into the suspended state). In one embodiment, an operating environment in accordance with this disclosure can apply Fast Task-Switching to all executing processes (unless making use of another service as described below). In another embodiment, each application must specify programmatically that it may be suspended.

[0037] While suspended a process/application can execute no instructions. In a threaded operating environment such as Unix or Mach, this means that a process cannot schedule a thread for execution. In some embodiments, however, kernel efforts initiated on behalf of a process may continue to execute even after the process itself has been suspended (or hibernated or terminated). Illustrative "kernel efforts" include in-progress I/O operations and virtual memory and network buffer operations. Completion of such operations may or may not trigger the reanimation of the suspended process.

[0038] It is noted that process suspension of the type discussed here is different from system-level operations such as prior art "sleep" or "hibernate" actions – both of which are "system-wide" operations. In accordance with the prior art, no process may execute while in the sleep or hibernate mode. In contrast, the suspend mode described herein applies to individual processes; suspension of a first process does not interfere with the execution of a second process.

[0039] 2. Task Completion Service. The Task Completion service provides a process with the limited ability to complete a task begun in the foreground state while in the background state. In one embodiment, background processes (*i.e.,* processes executing in the background state) are throttled to avoid interfering with foreground processes. Illustrative tasks that are well suited to use this service are network upload and download operations, disk storage and retrieval operations and image processing operations (*e.g.,* those operations that do not require specialized graphics hardware such as GPU). In one embodiment, a background process is permitted to continue processing (*i.e.,* receiving CPU time in accordance with its scheduling priority as discussed above) until a specified event occurs. Illustrative events include a system timer (giving the process a specified amount of time to complete processing, *e.g.,* 5 to 30 minutes of "clock time" or 5 millisecond of CPU time), loss of external power to the device, or activation of a primary or secondary display.

[0040] Referring to FIG. **2**, illustrative Task Completion operation **200** begins when notice that a non-foreground application (SECOND APP) is to be brought into the foreground (block **205**). This may result from, for example, a user's request to start a second user application, transition to a system application, or an incoming call to a VOIP user application. If the currently foreground user application (FIRST APP) has not designated itself as an application capable of executing in background (the "NO" prong of block **210**), FIRST APP is placed into a non-executing state (block **215**) and SECOND APP is brought to the foreground (block **220**). By way of example, FIRST APP may be suspended (*i.e.,* placed into the suspended state). If FIRST APP has designated itself as an application capable of executing in background, such as though a p-list entry (the "YES" prong of block **210**), FIRST APP is placed into the background state and given a specified amount of time to complete its current operation (block **225**). SECOND APP may then be brought into foreground and permitted to execute (block **230**). Once in background, FIRST APP may be scheduled to execute in due course (see discussion above regarding processor scheduling) (block **235**). FIRST APP is permitted to execute in background until a specified event occurs and it has not completed its task. Once either of the

event occurs or the process completes its task (the "YES" prongs of blocks **240** and **245**), FIRST APP may be placed into a non-executing state; for example, the suspended or terminated state. If neither of these events occur (the "NO" prongs of block **240** and **245**), FIRST APP continues to execute (block **235**).

[0041] In one embodiment, non-active foreground processes may be similarly limited in their ability to access full system resources. For example, one or more non-active foreground processes (*i.e.,* a foreground process not currently receiving user input) may have their access to storage disks (restriction 2), incoming and outgoing network buffers (restrictions 3 and 4) and/or their ability to interact with specified hardware (restriction 8) limited in the same manner as described here for background processes. In like manner, on power loss all foreground processes except the active foreground process may be suspended or "pushed" into the background, where their behavior is thereafter controlled in accordance with the background processing regime described here.

[0042] In one embodiment, a process must declare programmatically that it wants to avail itself of the Task Completion service (*e.g.,* during acts in accordance with block **210**). This may be done, for example, by statically declaring the user application can avail itself of the Task Completion service though an API call, a p-list entry or a changeable setting specified by the user at install time or first run rime. In another embodiment, task completion is granted based on an application declaring, at run time and in code, that it is beginning an operation (then subsequently declaring that it has ended that operation). If such tasks are still outstanding when the application leaves the foreground state, the application may be granted some time to complete them. On occurrence of the specified event, or sooner if the task(s) completes, the application may be suspended. In yet another embodiment, any process or user application executing one of a specified number of eligible tasks may avail itself of the Task Completion service (*e.g.,* one of the aforementioned operations). For example, any process executing a file upload or download operation could be placed into background in accordance with Task Completion service restrictions.

[0043] 3. Audio Service. The Audio service permits a process or application to play or record audio while in the background state. In one embodiment, audio processes utilize real-time threads and are scheduled accordingly (See FIG. **1**). While real-time processes are, in general, demoted to a non-real-time priority when placed into background, this is not true for an application using the Audio service. Because the described embodiments utilize real-time threads for audio applications, to minimize interference with foreground processes only one real-time audio process at a time may be allowed in background state. (It will be recognized that this number is merely illustrative. If the processing power of the personal electronic device is sufficient, multiple audio processes may be active in the background at the same time.)

[0044] Referring to FIG. **3**, illustrative Audio service operation **300** begins when a foreground user application executing an audio operation (*e.g.,* music playback or voice recording operations) receives notice that another application not currently in the foreground (SECOND APP) is to be brought into the foreground (block **305**). The user's audio application is then placed into background (block **310**) and the SECOND APP is brought into foreground (block **315**). In one embodiment, real-time background threads may be limited to execute from a specified set of libraries. For example, real-time background threads may be prevented from using third party libraries (*e.g.,* libraries supplied by an application developer). At some point in the future, the user may issue a "pause" command to their background audio application (block **320**). In response, the user's background audio application is moved from the background state to the suspended state (block **325**). The audio application is kept in the suspended state until the user issues a "resume" command (the "YES" prong of block **330**), after which the audio application is moved back into the background state where it may continue to execute it's audio operation (block **335**). In another embodiment, the audio application may be placed into the hibernate mode rather than the suspended mode.

[0045] In some embodiments, a process may statically declare it can avail itself of the Audio service though an API call, a p-list entry or a changeable setting specified by the user at install time or first run rime. In other embodiments, any process

either playing or recording audio may have the Audio service applied to it. When executing code within these applications, the associated process could not be suspended during audio operations in accordance with the Fast Task-Switching or Task Completion services.

[0046] 4. Notification Service. The Notification service permits a process, in whatever state (foreground, background, suspended or hibernated), to receive and respond to messages. These messages may originate from an external source (a "push" notification) or the receiving process/application itself ("local" notification).

[0047] Generally speaking, on message receipt an operating system module (*e.g.,* the "Springboard" application in the iPhone operating system) identifies the target application (*i.e.,* that application the notification message is directed towards), launches that application in background and passes the message to the now reanimated application. In another embodiment, on message receipt the operating system (*e.g.,* Springboard) generates a dialog box through which the user can elect to switch to the target application (*i.e.,* the application to which the notification message is directed). If so elected, the target application is brought into foreground. (It will be recognized that if the target application is already in the background, it need not be reanimated.) The application is thereafter given an opportunity to respond to the message, including presenting a dialog to a user through, for example, an operating system request. Control of the receiving process is then provided in accordance with the relevant service (*e.g.,* Fast Task Switching, Task Completion, Audio, VOIP or Location and Geo-Fencing). In another embodiment, notifications destined for a background process may be queued up and delivered at a later time (*e.g.,* every 5 minutes) rather than being delivered substantially immediately after notification receipt.

[0048] 5. Network Service. The Network service permits a suspended network application to maintain (*i.e.,* keep alive) its communication socket(s). As used herein, a network application is a user application that uses network communication sockets. An illustrative network application is a user VOIP application. In the prior art when a user's network application/process is suspended, it loses its ability to

maintain its communication socket(s) and, as a result, cannot continue to receive communication data *(e.g.,* an incoming VOIP phone call).

[0049] Referring to FIG. **4,** an illustrative Network service operation is demonstrated by VOIP service operation **400** which begins with a user's VOIP application being transitioned into the suspended state (block **405).** In one embodiment, as a VOIP process is being transitioned into the suspended state it specifies to the operating system *(e.g.,* via a network daemon) what time interval it requires for the sending of "heartbeat" messages **(410).** This daemon is responsible for reanimating the process when data is received at the processes associated socket(s) or on expiration of the specified time - the latter for heartbeat maintenance. For more information on this aspect, see U.S. Patent Publication 20090305732 entitled Managing Notification Service Connections and Displaying Icon Badges.

[0050] At some later time, the network daemon receives a message (block **415).** The target application is then identified (block **420),** reanimated into the background state (block **425)** and passed the message (block **430).** The background VOIP application may then respond to the message *(e.g.,* an incoming VOIP call or a "heartbeat needed" message) (block **435).** If the needed task was to issue a heartbeat, the VOIP application may be returned to the suspended state. If the needed task was to respond to an incoming VOIP call, the VOIP application may request the operating system issue a UI so that a user may select whether to accept or ignore the call. If the call is ignored, the VOIP application may be returned to the suspended state. If the call is to be answered, the VOIP application may be brought to the foreground *(e.g.,* by means of the user selecting the appropriate element in a UI presented by the operating system). While it will be recognized that operating system daemons are well-known in the art, the maintenance of network communication *(e.g.,* VOIP and wireless fidelity or WiFi) socket(s) for a suspended process in a personal electronic device of the type described herein is novel.

[0051] 6. Location and Geo-Fencing Service. The Location and Geo-Fencing service permits a background or suspended process to receive notice *(e.g.,* a message from on operating system daemon) when a specified location event occurs.

Illustrative location events include, but are not limited to, arriving at a specified location, leaving a specified location, entering a specified region, leaving a specified region or moving a "significant" distance (where what constitutes "significant" can be specified by the user or the user application). In one embodiment, a process/application must declare programmatically that it wants to avail itself of the Location and Geo-Fencing Operation service (*e.g.,* via a p-list parameter). In another embodiment, Geo-Fencing operations do not require the use of a p-list entry – all Geo-Fencing applications may avail themselves of the Geo-Fencing service.

[0052] In location operations, as long as a location tracking operation is underway, the application may be kept in background – receiving processing time in accordance with system scheduling mechanisms (see discussion above). In one embodiment, when a tracking operation either terminates or is otherwise halted, the process may be suspended (*i.e.,* placed into the suspended state; the process could also be terminated or hibernated). Once the process responds to the message, it may quiesce (returning to the suspended state) or block in the background state awaiting the next message. In one embodiment the number of user applications using Location services in the background state relates to the usage of other system resources; for example, if system memory runs low, user applications currently performing location tracking may be terminated. In another embodiment, a centralized daemon aggregates all of the location requests (particularly with respect to the desired accuracy requested), and starts appropriate hardware for the most aggressive mode requested. For example, if one user application requests the user's location to within 3 kilometers (in which case mobile phone cell-based positioning may be used), a second user application requests the user's location to within 500 meters (in which case WiFi scanning may be used), and a third application requests the user's location to within 100 meters (for which GPS is necessary), GPS positioning may be used and all applications that are tracking location in the background will get the benefit of the finer positioning.

[0053] In geo-fencing operations, whenever a specified region boundary is crossed the process receives a message from the operating system (*e.g.,* via a daemon). If the process is executing (*e.g.,* during its allocated time while in

background), it would be given the opportunity to process the message. If the application were not processing (*e.g.,* it was in background and not processing or it was suspended or hibernated), the process/application would be reanimated after which it would be allowed to process the event (in background). In addition, if the process had been terminated, it may be reanimated into the background state in a manner similar to the Notification service described earlier. See the related disclosure entitled "Location-Based Application Program Management," Serial No. 12/123,456. In an embodiment where location processing uses a separate processor than that which executes the operating system for user application execution (hereinafter referred to as the "application processor"), location and geo-fencing operations of the type described here may occur even when the application processor is asleep. As with Location services, the number of user applications that may avail themselves of Geo-Fencing services in the background state relates to the usage of other system resources; for example, if system memory runs low, user applications currently performing location tracking may be terminated.

[0054] Personal Electronic Device Architecture: Referring to FIG. **5**, in one embodiment personal electronic device **500** could implement one or more of the above services using one or more of the above-described restrictions to permit multitasking operations when it is beneficial to do so (*e.g.,* opportunistic multitasking). As illustrated, user applications **505A** to **505N** interact with application launcher **510** (*e.g.,* "Springboard" in the iPhone operating system) for standard operating system calls and multitasking services library **515** to access and make use of the above-described services.

[0055] Referring to FIG. **6**, system **600** in accordance with another embodiment could also include radio means **605** and location processing means **610**. As shown in FIG. **7**, radio means **605** may include receiver circuit **700**, transmitter circuit **705** and memory **710**. Receiver and transmitter circuits **700** and **705** may be used to connect to, for example, a mobile telephone network, a satellite communications network or a digital computer network (*e.g.,* the Internet via a local area network WiFi connection). Memory **710** may be used to store received data, data to be transmitted and operational parameter information as is known in the art. Interface

715 may be provided to facilitate communications between radio means **605** and location means **610**. Location means **610** may include processor **720**, memory **725** and operating system interface **730**. In general, location means **610** receives requests for location updates and geo-fencing limits (*e.g.,* from one or more user applications **505A** to **505N** through interface **730**), processes location information received from radio means **605** (received via interface **715**) and returns location information and updates to the requesting application(s) through interface **730**. Processor **720** may be one or more off-the-shelf processors, one or more custom designed processors or a combination of off-the-shelf and custom designed processors. Memory **725** may be any form or combination of volatile and non-volatile solid state, magnetic or optical memories.

[0056] In one embodiment, one or more of the described services may be provided based on the availability of power. For example, if the personal electronic device is plugged into a power source (*e.g.,* AC power), multitasking capabilities may be provided in accordance with standard modern operating system procedures (*i.e.,* full or unrestricted multitasking). If, on the other hand, the personal electronic device is not plugged into a power source, multitasking capabilities may be provided in accordance with this disclosure (*e.g.,* partial or opportunistic multitasking).

[0057] Referring to FIG. **8**, personal electronic device **800** may be embodied in a physical apparatus that includes processing unit **805** and support devices **810**. Processing unit **805** may include processor **815**, memory **820** and input-output circuit **825**. Device **800** may also include radio means **605** and location means **610** as shown. Input-output circuit **825** couples processor **815** and memory **820** to one or more display units **830** (*e.g.,* a touch-screen display), one or more input-output devices **835** (*e.g.,* touch-screen, keyboard, mouse, joy stick), one or more program storage devices **840** (*e.g.,* magnetic, optic or solid-state memory) and other hardware **845** (*e.g.,* a network interface). It is noted that in the architecture presented in FIGS. **6**, **7** and **8** location and geo-fencing processing may continue via radio means **605** and location means **610** even when/if the application making the location or geo-fencing request (*e.g.,* application **505A**) has been suspended, hibernated or terminated. Further still, location and geo-fencing operations may

continue even if application processor **815** is hibernated or asleep. That is, location and geo-fencing operations may continue via processor **720** regardless of what state application processor **815** is in.

[0058] Various changes in the materials, components, circuit elements, as well as in the details of the illustrated operational methods are possible without departing from the scope of the following claims. The provision of multitasking capabilities in accordance with this disclosure may be performed by a programmable control device executing instructions organized into one or more program modules. A programmable control device (*e.g.,* processors **720** and **215**) may include any programmable controller device including, for example, one or more members of the Intel Atom®, Core®, Pentium® and Celeron® processor families from Intel Corporation and the Cortex and ARM processor families from ARM or custom designed state machines. (INTEL, INTEL ATOM, CORE, PENTIUM, and CELERON are registered trademarks of the Intel Corporation. CORTEX is a registered trademark of the ARM Limited Corporation. ARM is a registered trademark of the ARM Limited Company.) Custom designed state machines may be embodied in a hardware device such as an integrated circuit including, but not limited to, application specific integrated circuits ("ASICs") or field programmable gate array ("FPGAs").

[0059] Storage devices suitable for tangibly embodying program instructions (*e.g.,* memories **710**, **725**, **220** and **820**) include, but are not limited to: magnetic disks (fixed, floppy, and removable) and tape; optical media such as CD-ROMs and digital video disks ("DVDs"); and semiconductor memory devices such as Electrically Programmable Read-Only Memory ("EPROM"), Electrically Erasable Programmable Read-Only Memory ("EEPROM"), Programmable Gate Arrays and flash devices.

[0060] Finally, it is to be understood that the above description is intended to be illustrative, and not restrictive. For example, the above-described embodiments may be used in combination with each other. Many other embodiments will be apparent to those of skill in the art upon reviewing the above description. The scope of the invention therefore should be determined with reference to the appended claims, along with the full scope of equivalents to which such claims are entitled. In the

appended claims, the terms “including” and “in which” are used as the plain-English equivalents of the respective terms “comprising” and “wherein.”

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A method, comprising:

scheduling a first foreground process with a scheduling priority from a first band of priority levels;

scheduling, based on the scheduling a first foreground process with a scheduling priority from a first band of priority levels, a first background process with a scheduling priority from a second band of priority levels;

associating, with at least one of the first foreground process and the first background process, an allotted quantum of one or more specified system resources; and

adjusting the scheduling priority of one of the first foreground process and the first background process in response to one of the first foreground process and the first background process using a full amount of the allotted quantum of one or more specified system resources by the one of the first foreground process and the first background process,

wherein the priority levels in the first band and the second band overlap by one, but not all, priority levels.

2. The method of claim 1, further comprising scheduling a third process with a priority from a real-time priority band, wherein all priority levels in the real-time priority band are greater than all priority levels in the first band.

3. The method of claim 2, wherein the scheduling a third process comprises scheduling a background process.

4. The method of claim 1, wherein the one or more specified system resources are selected from a group including a camera, a graphics processing unit, an accelerometer, a proximity sensor, and a microphone.

5. The method of claim 1, wherein the adjusting the scheduling priority of one of the first foreground process and the first background process includes reducing the scheduling priority of the one of the first foreground process and the first background process by a single count.

6. The method of claim 1, wherein the adjusting the scheduling priority of one of the first foreground process and the first background process includes reducing the scheduling priority of the one of the first foreground process and the first background process by more than one count.

7. The method of claim 1, wherein the first background process is rate-limited in accessing system storage.

8. The method of claim 1, wherein the first foreground process is given priority over the first background process when the first foreground process accesses system storage.

9. A program storage device, readable by a programmable control device, comprising instructions stored thereon for causing the programmable control device to perform a method in accordance with any one of claims 1 to 8.

10. A personal electronic device, comprising:

memory;

computer program instructions, stored in the memory; and

a programmable control device operatively coupled to the memory and configured to execute the computer program instructions to cause the programmable control device to perform operations of:

scheduling a first foreground process with a scheduling priority from a first band of priority levels;

scheduling, based on the scheduling a first foreground process with a scheduling priority from a first band of priority levels, a first background process with a scheduling priority from a second band of priority levels;

associating, with at least one of the first foreground process and the first background process, an allotted quantum of one or more specified system resources; and

adjusting the scheduling priority of one of the first foreground process and the first background process in response to one of the first foreground process and the first background process using a full amount of the allotted quantum of one or more specified system resources by the one of the first foreground process and the first background process,

wherein the priority levels in the first band and the second band overlap by one, but not all, priority levels.

11. The personal electronic device of claim 10, further comprising operations of scheduling a third process with a priority from a real-time priority band, wherein all priority levels in the real-time priority band are greater than all priority levels in the first band.

12. The personal electronic device of claim 11, wherein the scheduling a third process comprises scheduling a background process.

13. The personal electronic device of claim 10, wherein the one or more specified system resources are selected from a group including a camera, a graphics processing unit, an accelerometer, a proximity sensor, and a microphone.

14. The personal electronic device of claim 10, wherein the adjusting the scheduling priority of one of the first foreground process and the first background process includes reducing the scheduling priority of the one of the first foreground process and the first background process by a single count.

15. The personal electronic device of claim 10, wherein the adjusting the scheduling priority of one of the first foreground process and the first background process includes reducing the scheduling priority of the one of the first foreground process and the first background process by more than one count.

16. The personal electronic device of claim 10, wherein the first background process is rate-limited in accessing system storage.

17. The personal electronic device of claim 10, wherein the first foreground process is given priority over the first background process when the first foreground process accesses system storage.

PROCESSOR THREAD SCHEDULING STRUCTURE

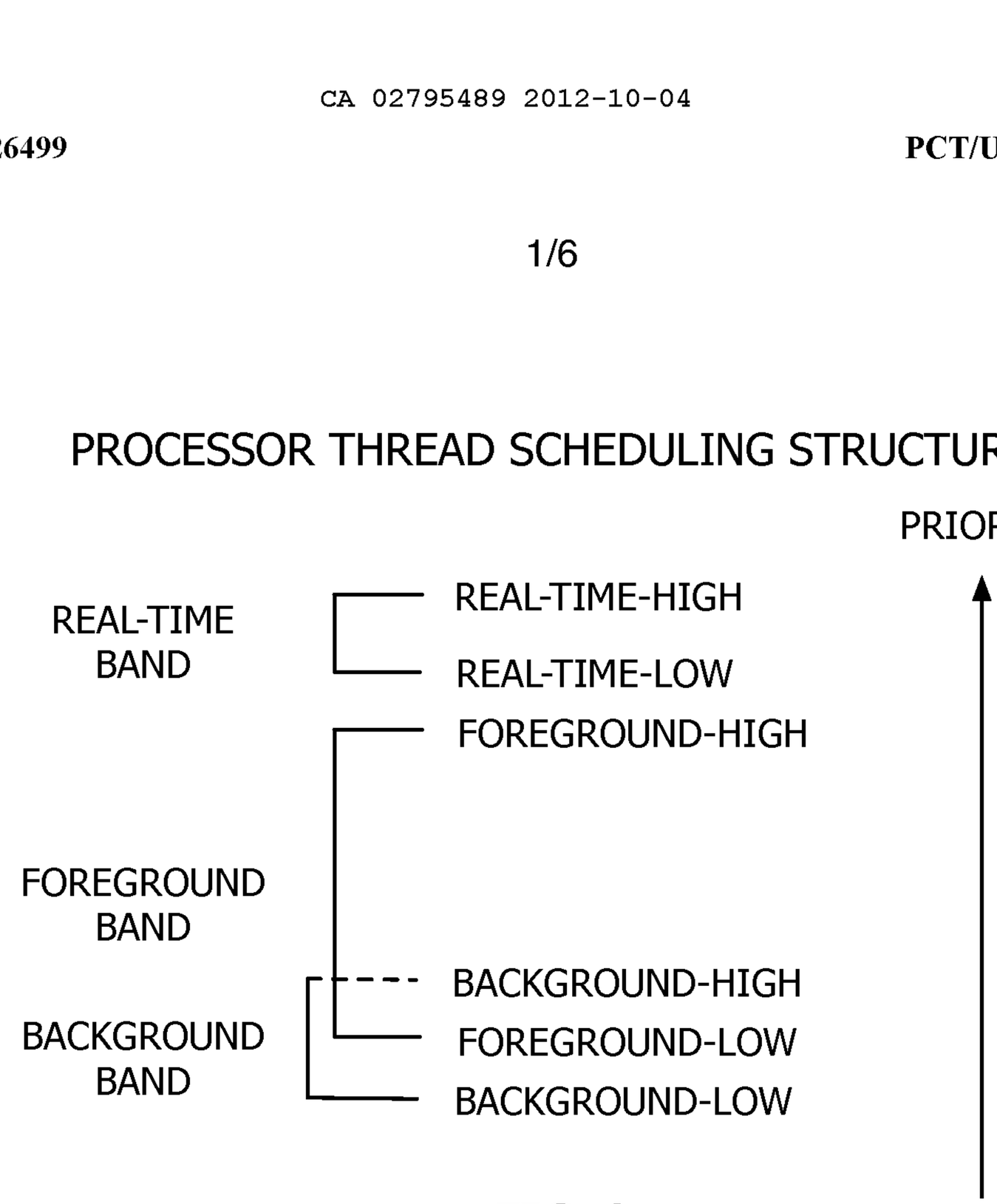


FIG. 1

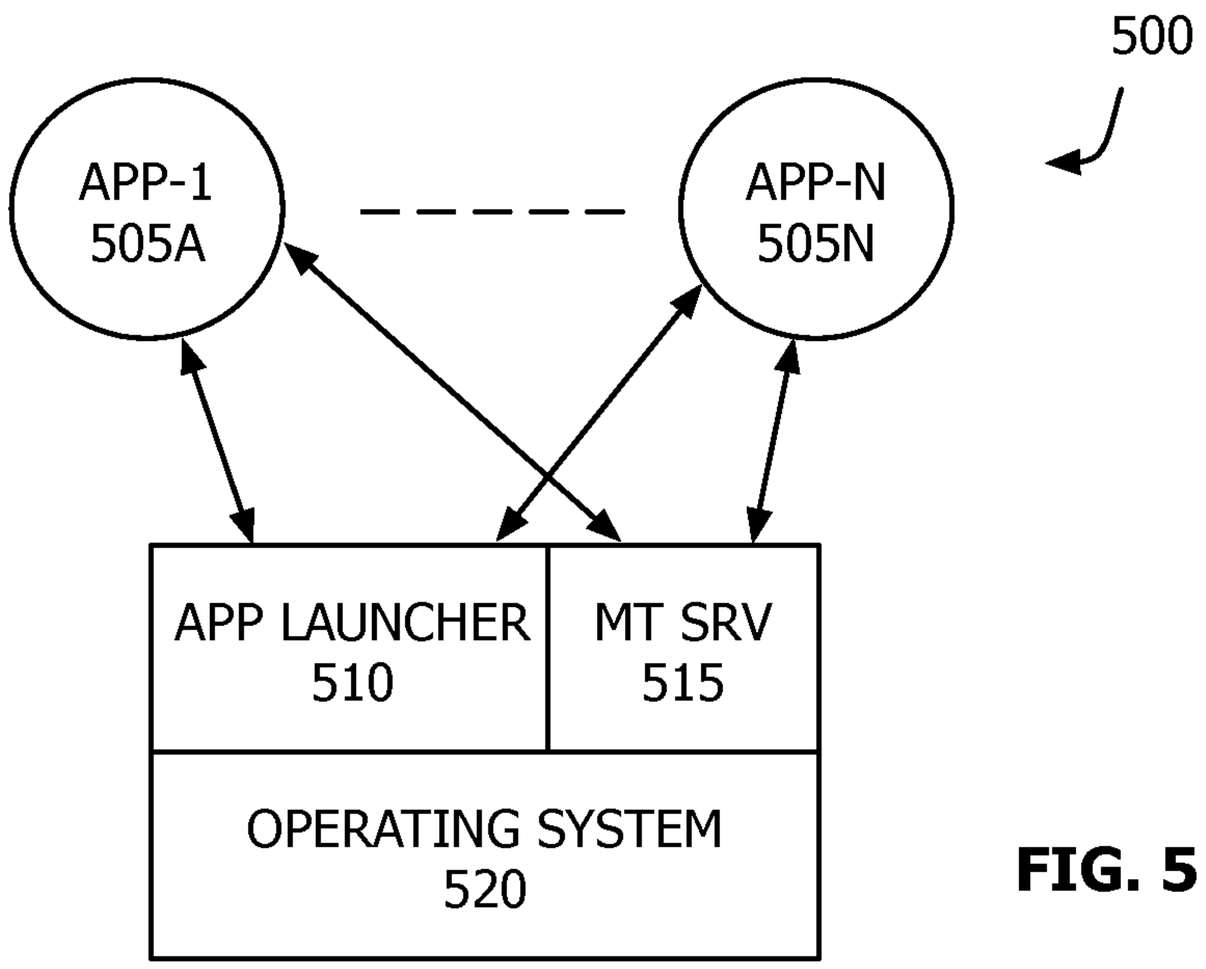


FIG. 5

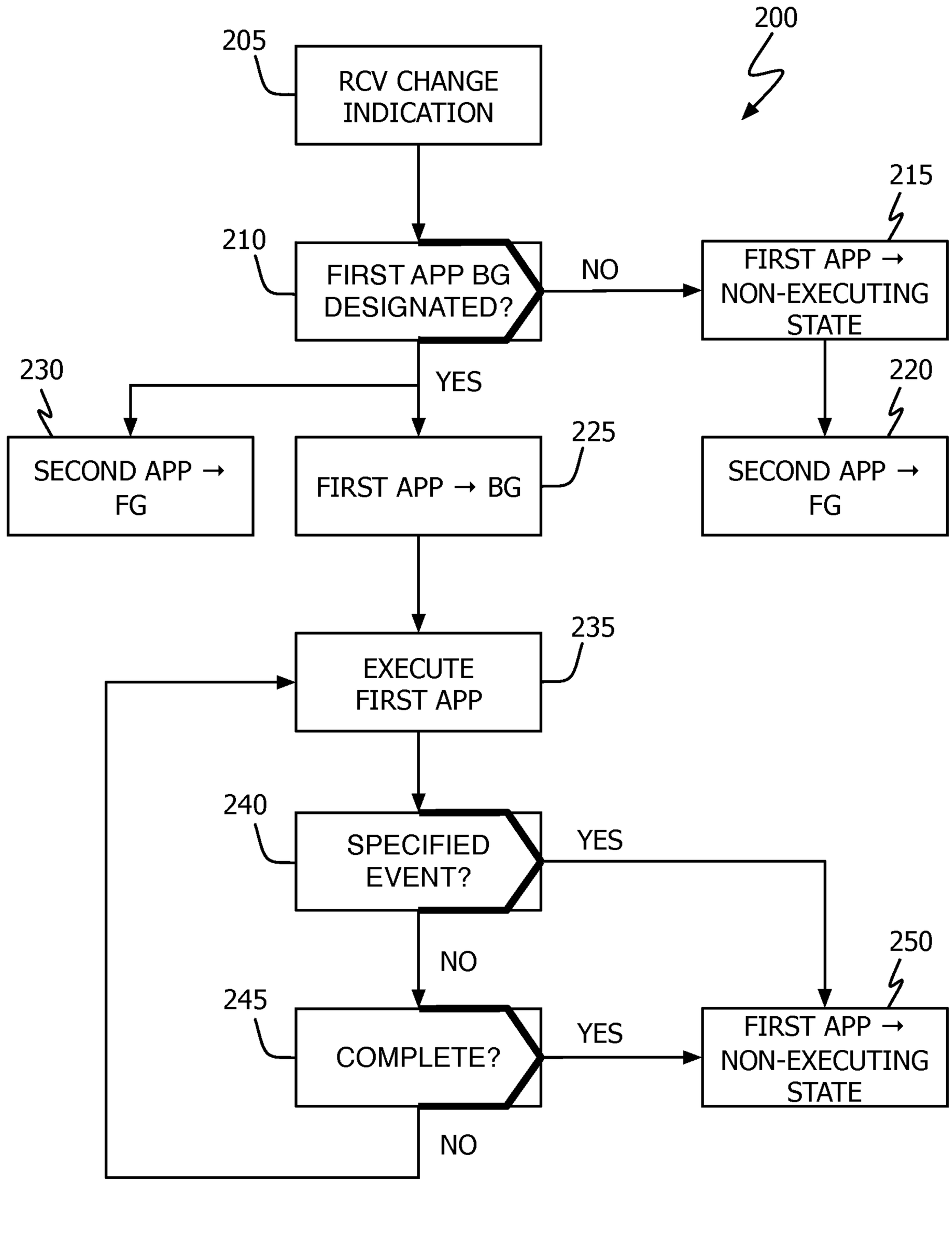

200
205
RCV CHANGE INDICATION
210
FIRST APP BG DESIGNATED?
NO
215
FIRST APP → NON-EXECUTING STATE
YES
230
220
SECOND APP → FG
225
FIRST APP → BG
SECOND APP → FG
235
EXECUTE FIRST APP
240
SPECIFIED EVENT?
YES
NO
250
245
COMPLETE?
YES
FIRST APP → NON-EXECUTING STATE
NO

FIG. 2

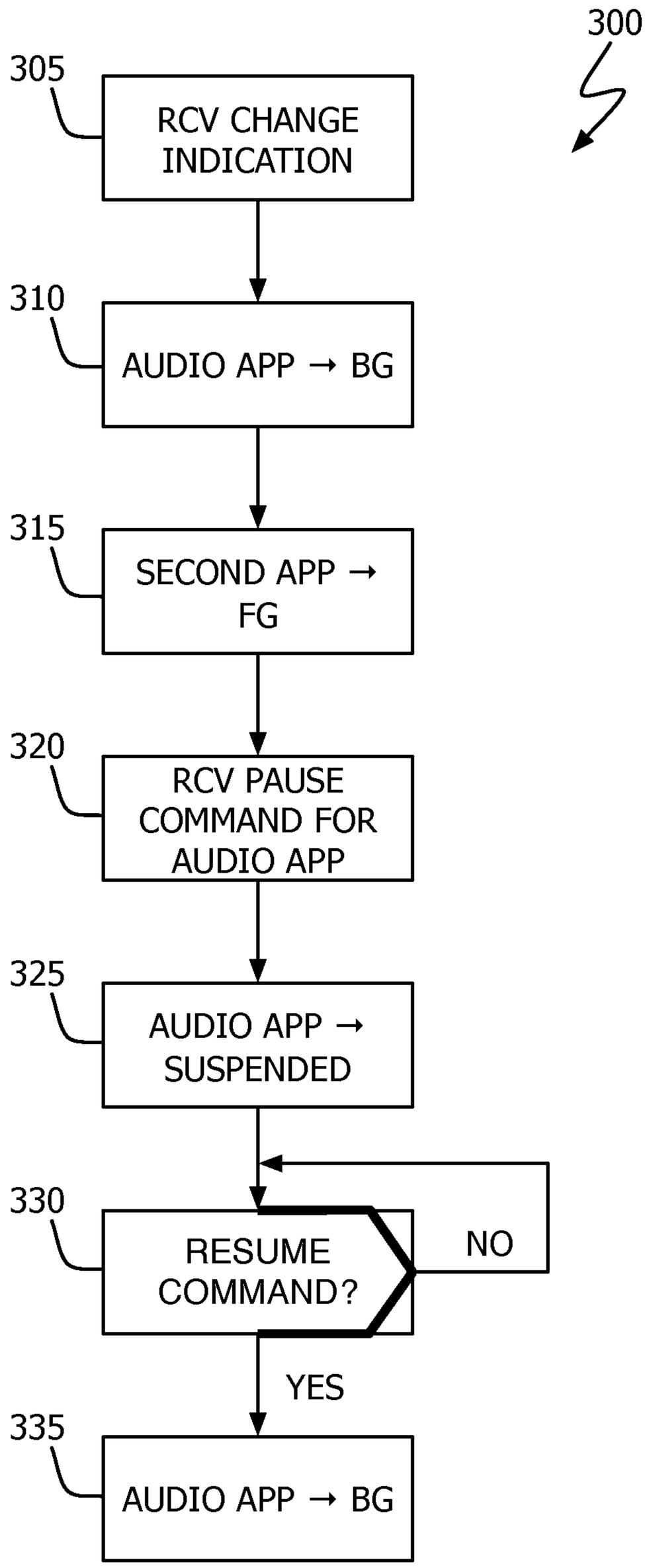

300
305
RCV CHANGE INDICATION
310
AUDIO APP → BG
315
SECOND APP → FG
320
RCV PAUSE COMMAND FOR AUDIO APP
325
AUDIO APP → SUSPENDED
330
RESUME COMMAND?
NO
YES
335
AUDIO APP → BG

FIG. 3

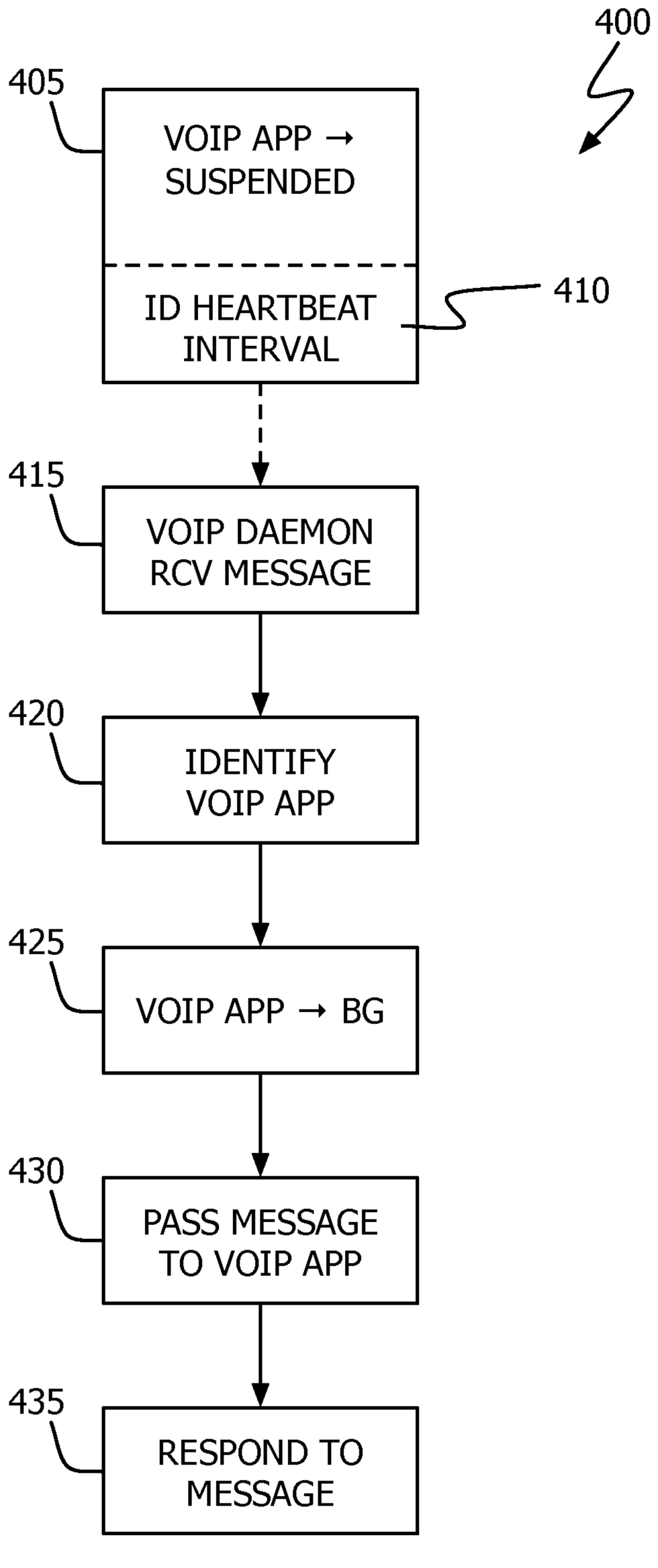

400
405
VOIP APP → SUSPENDED
ID HEARTBEAT INTERVAL
410
415
VOIP DAEMON RCV MESSAGE
420
IDENTIFY VOIP APP
425
VOIP APP → BG
430
PASS MESSAGE TO VOIP APP
435
RESPOND TO MESSAGE

FIG. 4

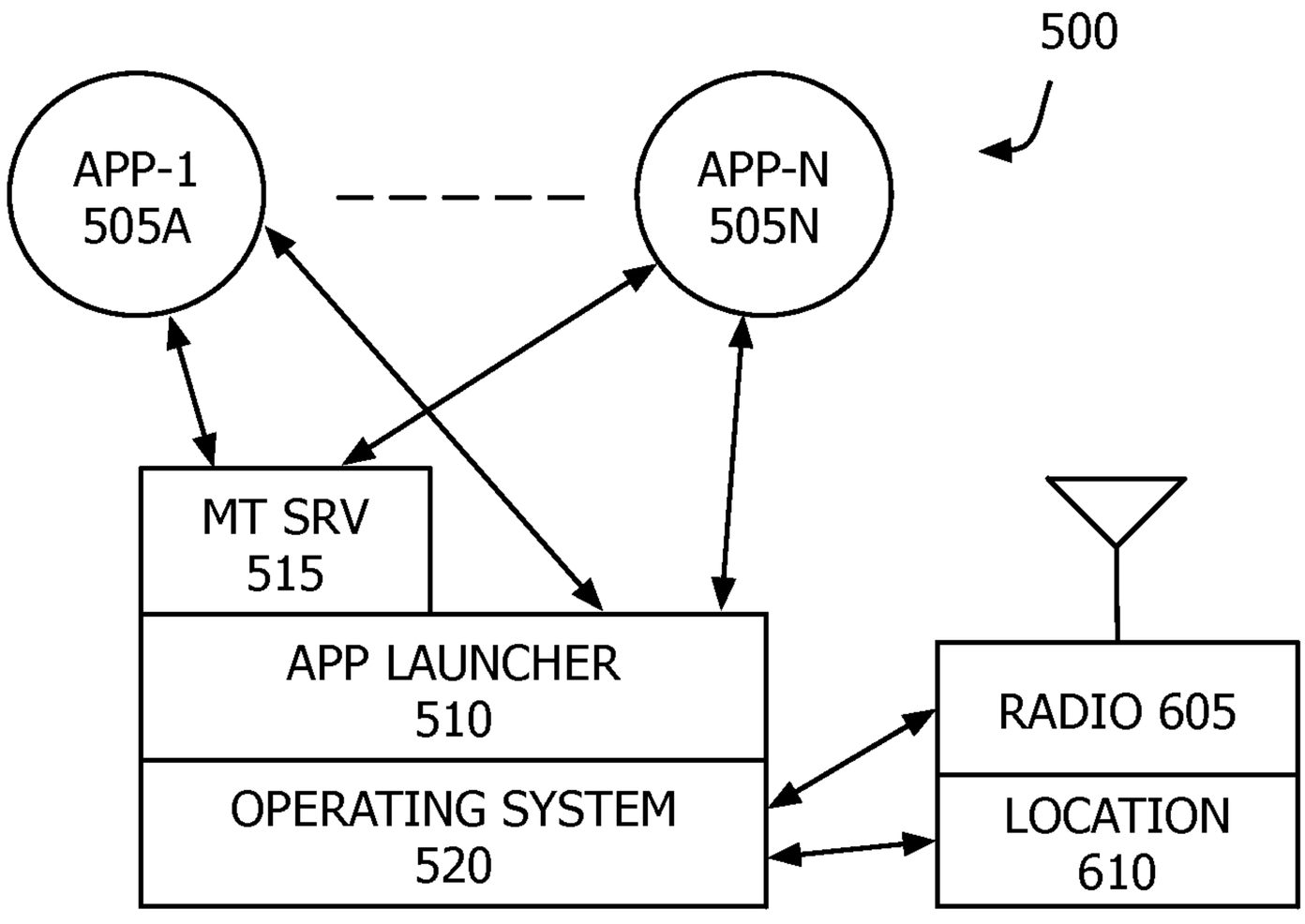

500
APP-1
505A
APP-N
505N
MT SRV
515
APP LAUNCHER
510
OPERATING SYSTEM
520
RADIO 605
LOCATION
610

FIG. 6

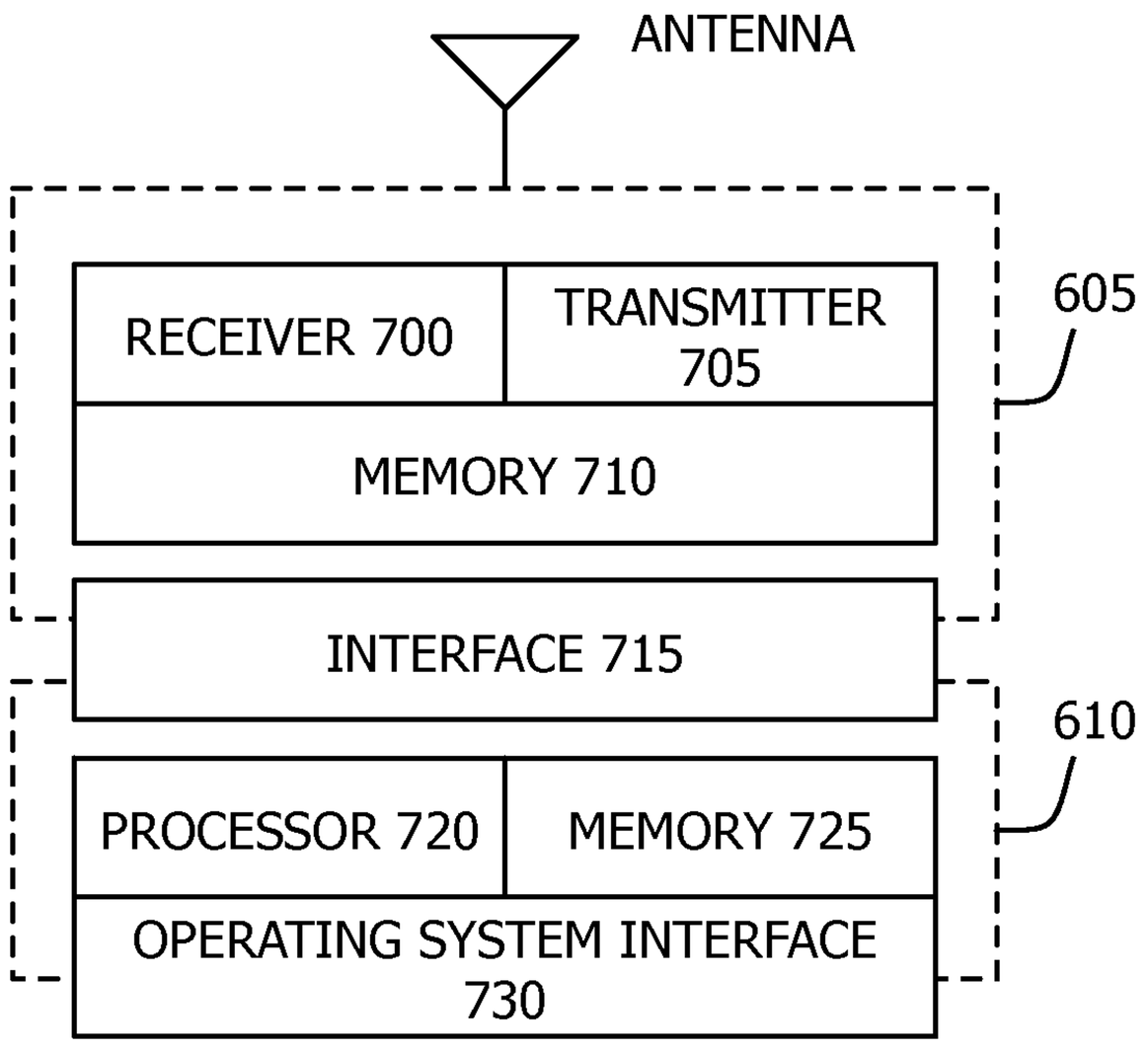

ANTENNA
RECEIVER 700
TRANSMITTER
705
MEMORY 710
605
INTERFACE 715
610
PROCESSOR 720
MEMORY 725
OPERATING SYSTEM INTERFACE
730

FIG. 7

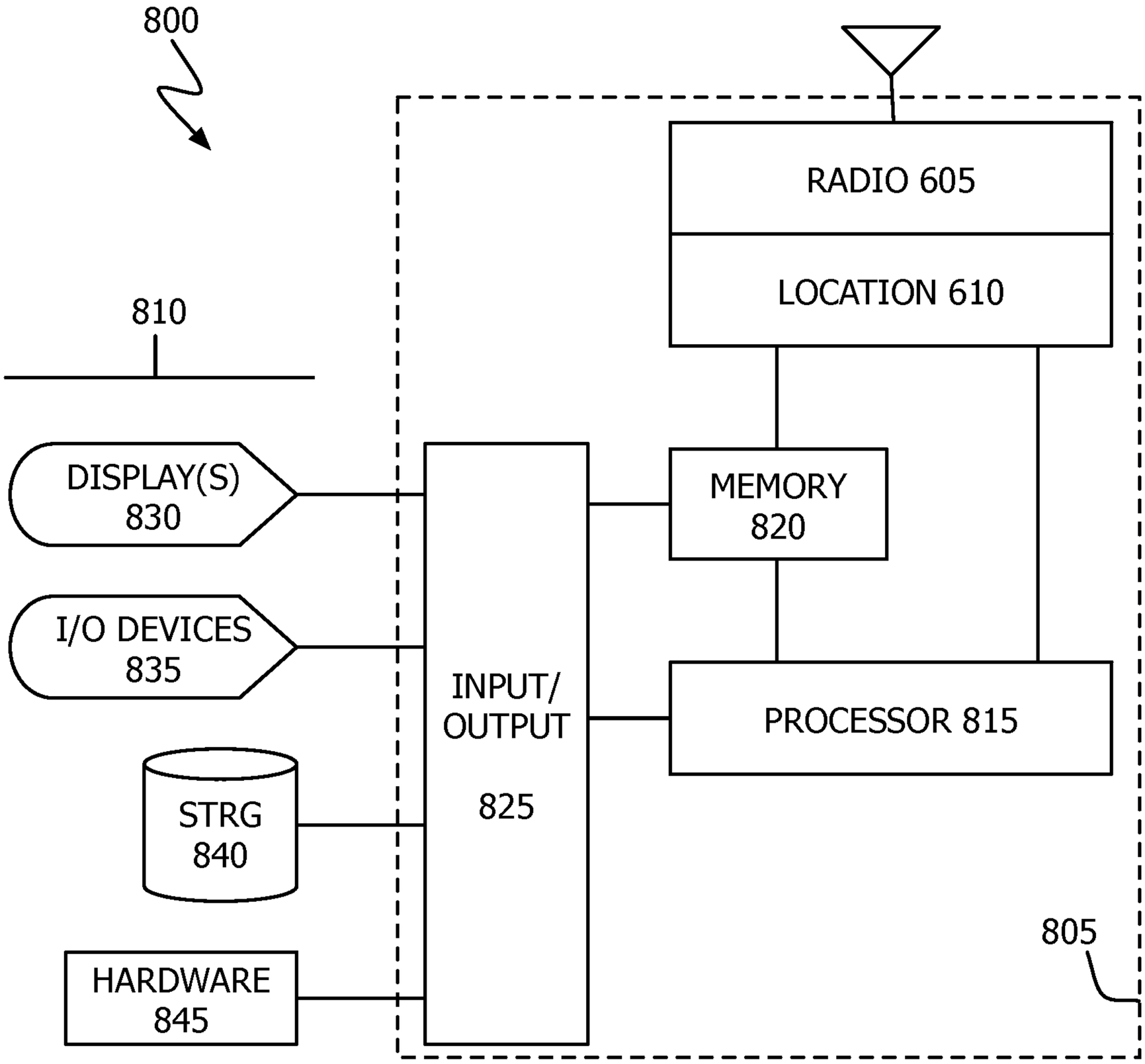
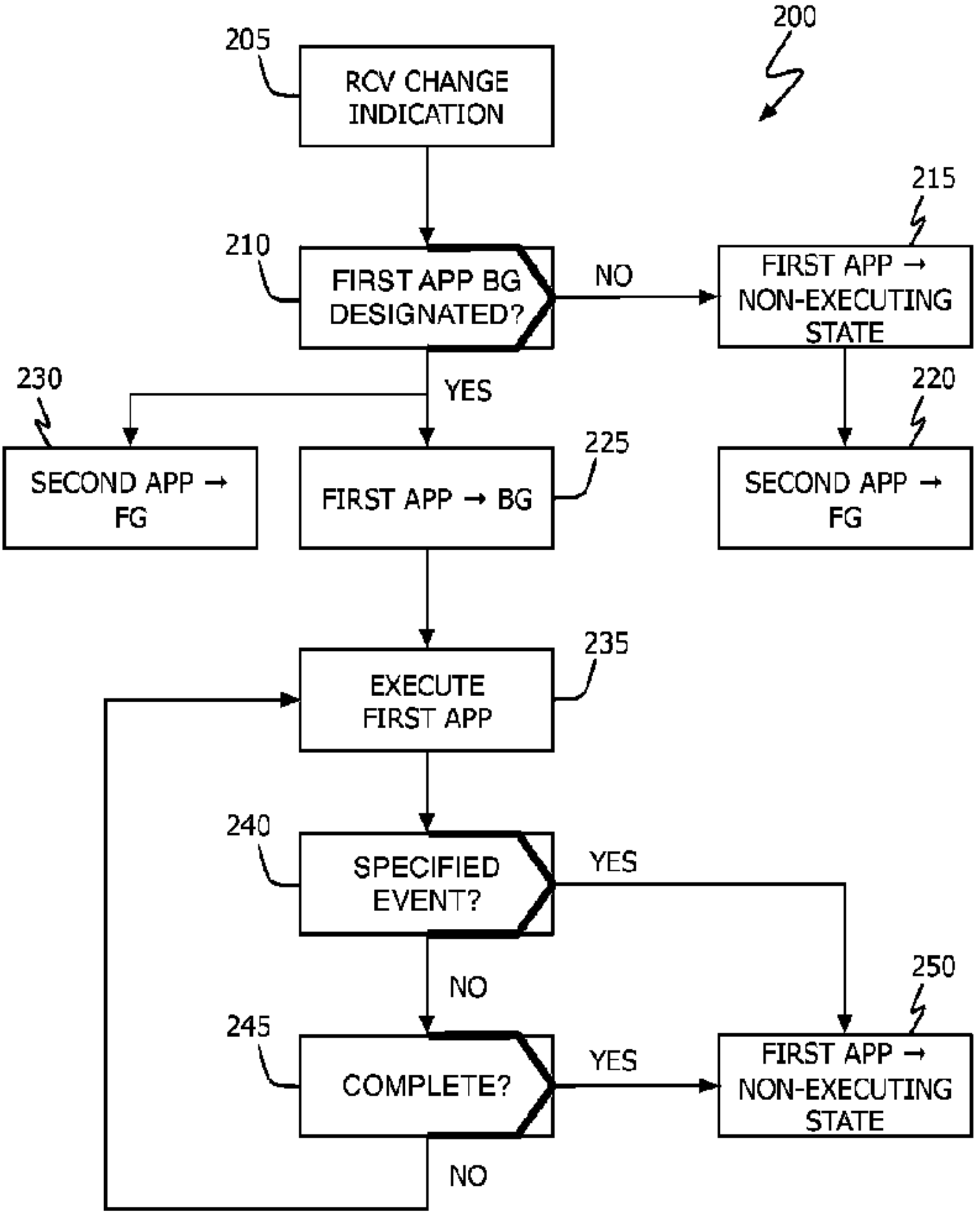

800
RADIO 605
LOCATION 610
810
DISPLAY(S)
830
MEMORY
820
I/O DEVICES
835
INPUT/
OUTPUT
825
PROCESSOR 815
STRG
840
805
HARDWARE
845

FIG. 8


200
205
RCV CHANGE INDICATION
210
FIRST APP BG DESIGNATED?
NO
215
FIRST APP → NON-EXECUTING STATE
YES
230
SECOND APP → FG
225
FIRST APP → BG
220
SECOND APP → FG
235
EXECUTE FIRST APP
240
SPECIFIED EVENT?
YES
NO
245
COMPLETE?
YES
NO
250
FIRST APP → NON-EXECUTING STATE