Exhibit A

US007383453B2

(12) **United States Patent**
Youngs

(10) **Patent No.: US 7,383,453 B2**
(45) **Date of Patent: *Jun. 3, 2008**

(54) **CONSERVING POWER BY REDUCING VOLTAGE SUPPLIED TO AN INSTRUCTION-PROCESSING PORTION OF A PROCESSOR**

(75) Inventor: **Lynn R. Youngs**, Cupertino, CA (US)

(73) Assignee: **Apple, Inc**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 55 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **11/213,215**

(22) Filed: **Aug. 25, 2005**

(65) **Prior Publication Data**

US 2005/0283628 A1 Dec. 22, 2005

Related U.S. Application Data

(63) Continuation of application No. 11/103,911, filed on Apr. 11, 2005, now Pat. No. 6,973,585, which is a continuation of application No. 10/135,116, filed on Apr. 29, 2002, now Pat. No. 6,920,574.

(51) **Int. Cl.**
G06F 1/00 (2006.01)
G06F 1/26 (2006.01)

(52) **U.S. Cl.** **713/300**; 713/320; 713/324

(58) **Field of Classification Search** 713/300
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,666,537	A *	9/1997	Debnath et al.	713/322
6,347,379	B1 *	2/2002	Dai et al.	713/320
6,721,892	B1 *	4/2004	Osborn et al.	713/300
6,792,551	B2 *	9/2004	Dai	713/320
6,795,896	B1 *	9/2004	Hart et al.	711/118
2003/0056127	A1 *	3/2003	Vaglica	713/300
2003/0120959	A1	6/2003	Bohrer et al.	713/320
2003/0120962	A1	6/2003	Dai et al.	713/320

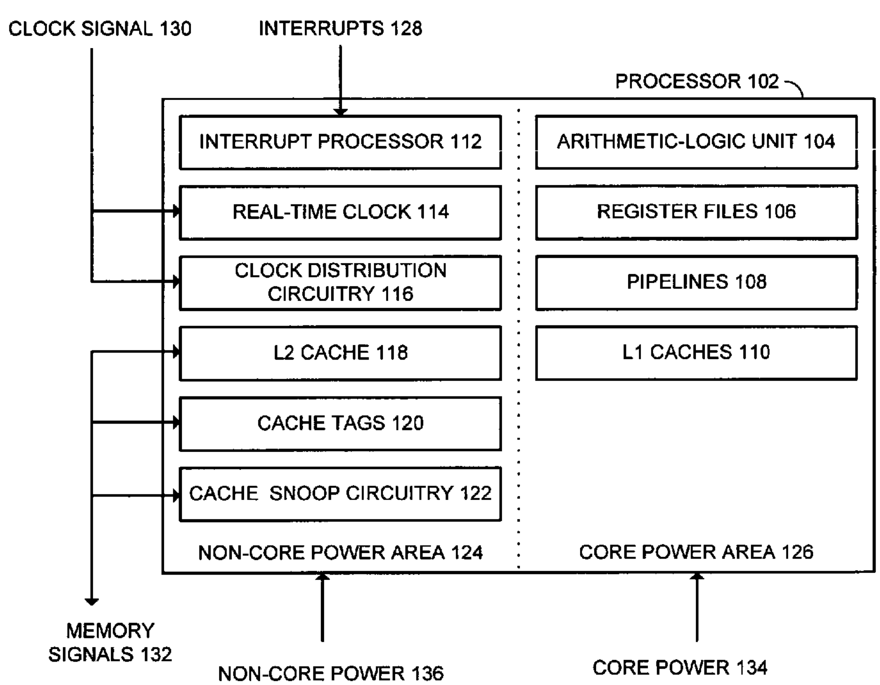
* cited by examiner

Primary Examiner—Rehana Perveen
Assistant Examiner—Stefan Stoynov
(74) *Attorney, Agent, or Firm*—Park, Vaughan & Fleming, LLP; Edward J. Grundler

(57) **ABSTRACT**

One embodiment of the present invention provides a system that facilitates reducing static power consumption of a processor. During operation, the system receives a signal indicating that instruction execution within the processor is to be temporarily halted. In response to this signal, the system halts an instruction-processing portion of the processor, and reduces the voltage supplied to the instruction-processing portion of the processor. Full voltage is maintained to a remaining portion of the processor, so that the remaining portion of the processor can continue to operate while the instruction-processing portion of the processor is in reduced power mode.

21 Claims, 3 Drawing Sheets



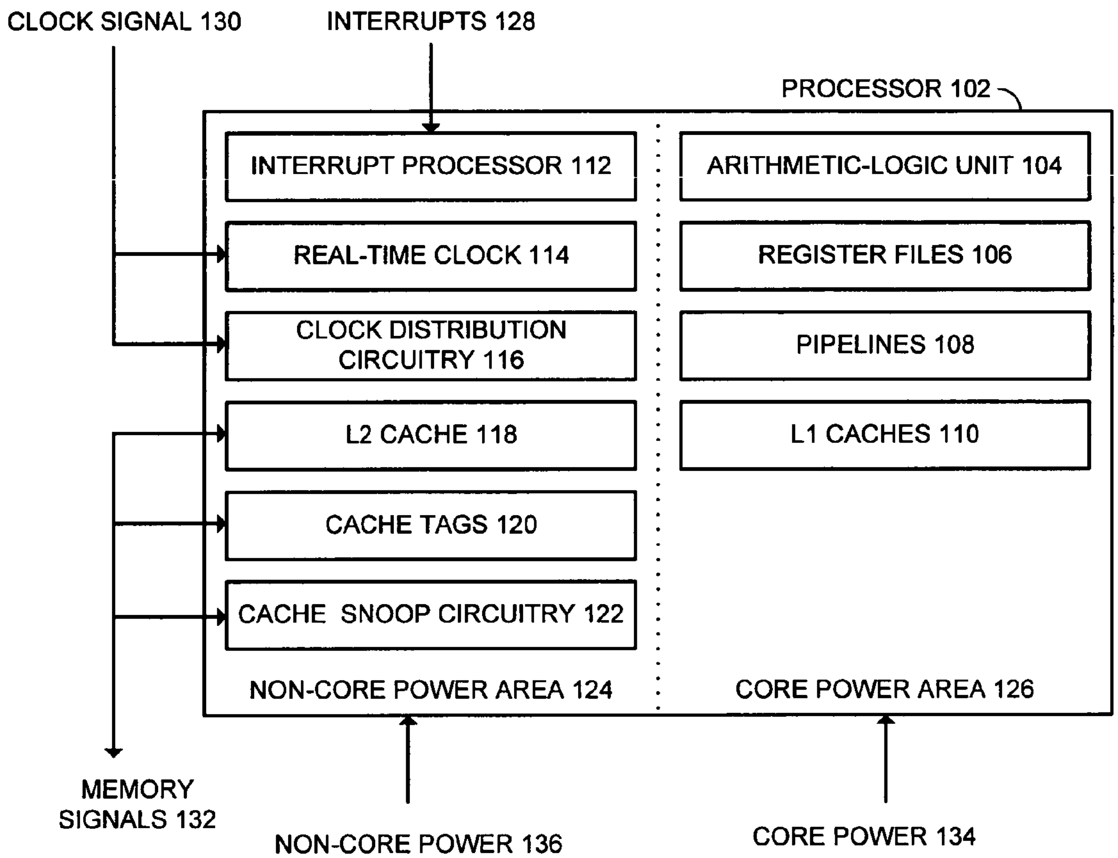

CLOCK SIGNAL 130
INTERRUPTS 128
PROCESSOR 102
INTERRUPT PROCESSOR 112
ARITHMETIC-LOGIC UNIT 104
REAL-TIME CLOCK 114
REGISTER FILES 106
CLOCK DISTRIBUTION CIRCUITRY 116
PIPELINES 108
L2 CACHE 118
L1 CACHES 110
CACHE TAGS 120
CACHE SNOOP CIRCUITRY 122
NON-CORE POWER AREA 124
CORE POWER AREA 126
MEMORY SIGNALS 132
NON-CORE POWER 136
CORE POWER 134

FIG. 1A

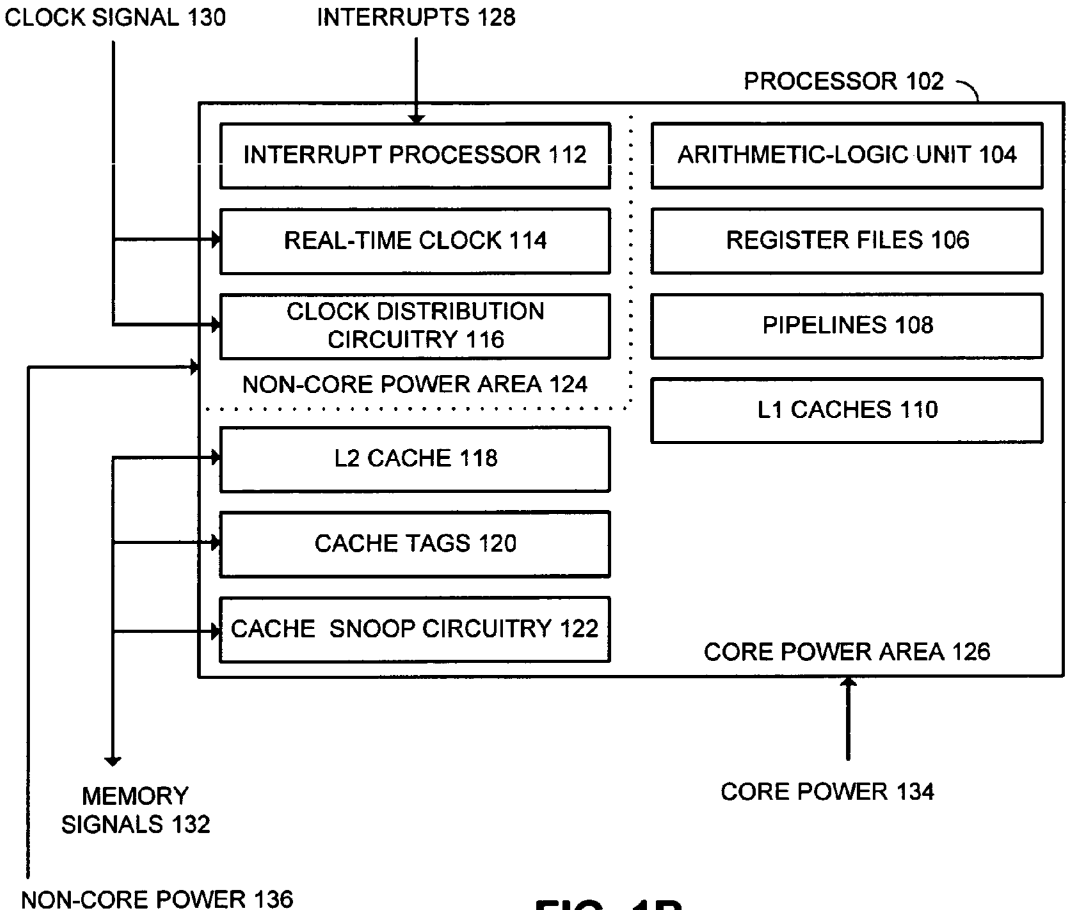

CLOCK SIGNAL 130
INTERRUPTS 128
PROCESSOR 102
INTERRUPT PROCESSOR 112
ARITHMETIC-LOGIC UNIT 104
REAL-TIME CLOCK 114
REGISTER FILES 106
CLOCK DISTRIBUTION CIRCUITRY 116
PIPELINES 108
NON-CORE POWER AREA 124
L1 CACHES 110
L2 CACHE 118
CACHE TAGS 120
CACHE SNOOP CIRCUITRY 122
CORE POWER AREA 126
MEMORY SIGNALS 132
CORE POWER 134
NON-CORE POWER 136

FIG. 1B

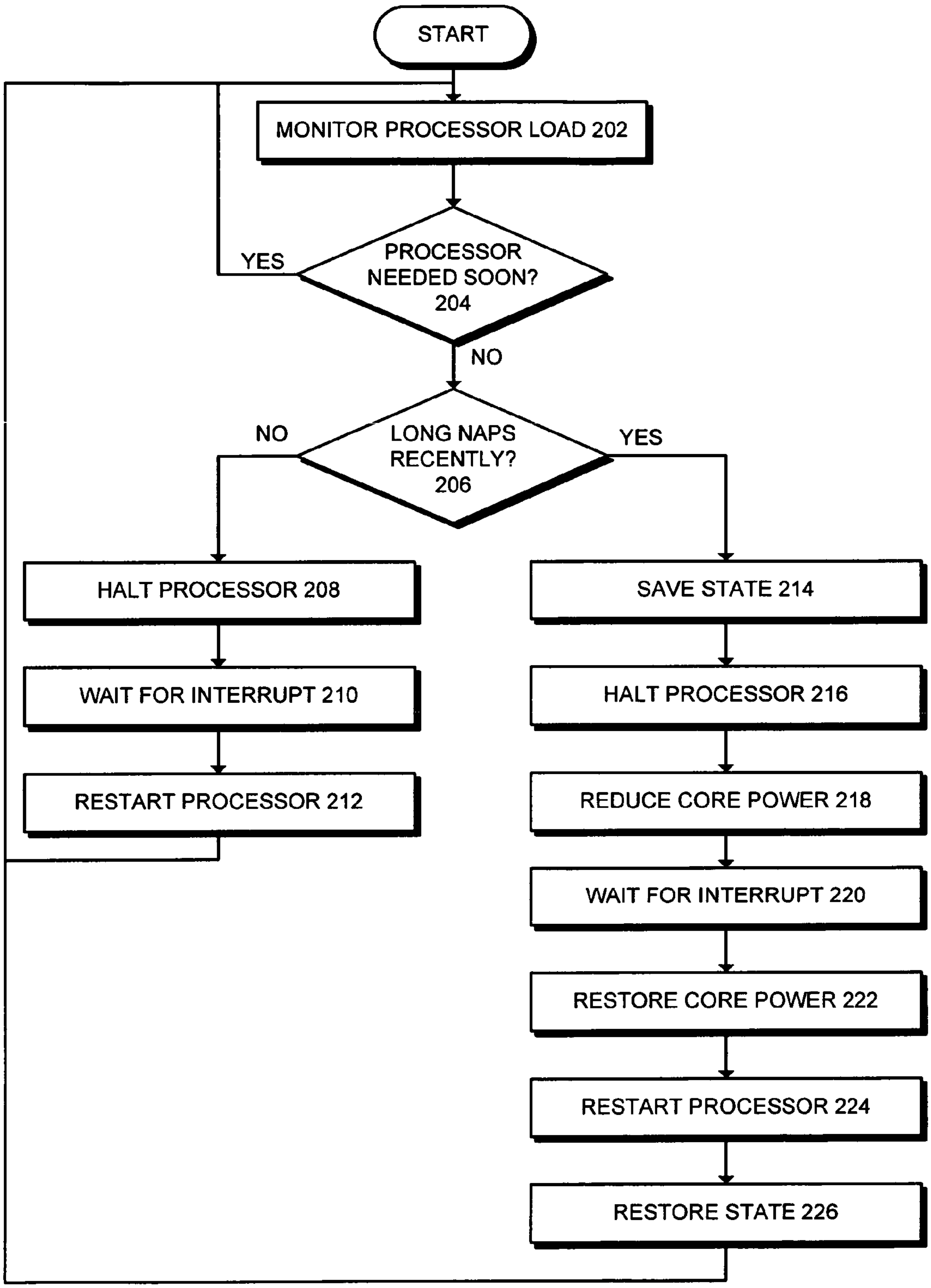

START
MONITOR PROCESSOR LOAD 202
PROCESSOR NEEDED SOON? 204
YES
NO
LONG NAPS RECENTLY? 206
NO
YES
HALT PROCESSOR 208
WAIT FOR INTERRUPT 210
RESTART PROCESSOR 212
SAVE STATE 214
HALT PROCESSOR 216
REDUCE CORE POWER 218
WAIT FOR INTERRUPT 220
RESTORE CORE POWER 222
RESTART PROCESSOR 224
RESTORE STATE 226

FIG. 2

CONSERVING POWER BY REDUCING VOLTAGE SUPPLIED TO AN INSTRUCTION-PROCESSING PORTION OF A PROCESSOR

RELATED APPLICATION

This application is a continuation of U.S. patent application Ser. No. 11/103,911, filed 11 Apr. 2005 now U.S. Pat No. 6,973,585. This application hereby claims priority under 35 U.S.C. §120 to the above-listed application. Note that pending U.S. patent application Ser. No. 11/103,911 is itself a continuation of U.S. patent application Ser. No. 10/135,116, filed 29 Apr. 2002 now U.S. Pat. No. 6,920,574.

BACKGROUND

1. Field of the Invention

The present invention relates to techniques for conserving power usage in computer systems. More specifically, the present invention relates to a method and an apparatus for reducing power consumption in a processor by reducing voltage supplied to an instruction-processing portion of the processor, while maintaining voltage to other portions of the processor.

2. Related Art

Dramatic advances in integrated circuit technology have led to corresponding increases in processor clock speeds. Unfortunately, these increases in processor clock speeds have been accompanied by increased power consumption. Increased power consumption is undesirable, particularly in battery-operated devices such as laptop computers, for which there exists a limited supply of power. Any increase in power consumption decreases the battery life of the computing device.

Modern processors are typically fabricated using Complementary Metal Oxide Semiconductor (CMOS) circuits. CMOS circuits typically consume more power while the circuits are switching, and less power while the circuits are idle. Designers have taken advantage of this fact by reducing the frequency of (or halting) clock signals to certain portions of a processor when the processor is idle. Note that some portions of the processor must remain active, however. For example, a cache memory with its associated snoop circuitry will typically remain active as well as interrupt circuitry and real-time clock circuitry.

Although reducing the frequency of (or halting) a system clock signal can reduce the dynamic power consumption of a processor, static power consumption is not significantly affected. This static power consumption is primarily caused by leakage currents through the CMOS devices. As integration densities of integrated circuits continue to increase, circuit devices are becoming progressively smaller. This tends to increase leakage currents, and thereby increases static power consumption. This increased static power consumption results in reduced battery life, and increases cooling system requirements for battery operated computing devices.

What is needed is a method and an apparatus that reduces static power consumption for a processor in a battery operated computing device.

SUMMARY

One embodiment of the present invention provides a system that facilitates reducing static power consumption of a processor. During operation, the system receives a signal

indicating that instruction execution within the processor is to be temporarily halted. In response to this signal, the system halts an instruction-processing portion of the processor, and reduces the voltage supplied to the instruction-processing portion of the processor. Full voltage is maintained to a remaining portion of the processor, so that the remaining portion of the processor can continue to operate while the instruction-processing portion of the processor is in reduced power mode.

In one embodiment of the present invention, reducing the voltage supplied to the instruction-processing portion of the processor involves reducing the voltage to a minimum value that maintains state information within the instruction-processing portion of the processor.

In one embodiment of the present invention, reducing the voltage supplied to the instruction-processing portion of the processor involves reducing the voltage to zero.

In one embodiment of the present invention, the system saves state information from the instruction-processing portion of the processor prior to reducing the voltage supplied to the instruction-processing portion of the processor. This state information can either be saved in the remaining portion of the processor or to the main memory of the computer system.

In one embodiment of the present invention, upon receiving a wakeup signal, the system: restores full voltage to the instruction-processing portion of the processor; restores state information to the instruction-processing portion of the processor; and resumes processing of computer instructions.

In one embodiment of the present invention, maintaining full voltage to the remaining portion of the processor involves maintaining full voltage to a snoop-logic portion of the processor, so that the processor can continue to perform cache snooping operations while the instruction-processing portion of the processor is in the reduced power mode.

In one embodiment of the present invention, the system also reduces the voltage to a cache memory portion of the processor. In this embodiment, the system writes cache memory data to main memory prior to reducing the voltage.

In one embodiment of the present invention, the remaining portion of the processor includes a control portion of the processor containing interrupt circuitry and clock circuitry.

In one embodiment of the present invention, the remaining portion of the processor includes a cache memory portion of the processor.

BRIEF DESCRIPTION OF THE FIGURES

FIG. **1**A illustrates different power areas within processor **102** in accordance with an embodiment of the present invention.

FIG. **1**B illustrates alternate power areas within processor **102** in accordance with an embodiment of the present invention.

FIG. **2** is a flowchart illustrating the process of monitoring processor load and switching to power saving modes in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION

The following description is presented to enable any person skilled in the art to make and use the invention, and is provided in the context of a particular application and its requirements. Various modifications to the disclosed embodiments will be readily apparent to those skilled in the art, and the general principles defined herein may be applied to other embodiments and applications without departing from the spirit and scope of the present invention. Thus, the present invention is not intended to be limited to the embodiments shown, but is to be accorded the widest scope consistent with the principles and features disclosed herein.

Processor **102**

FIG. **1**A illustrates different power areas within processor **102** in accordance with an embodiment of the present invention. Processor **102** is divided into a core power area **126**, and a non-core power area **124**. Core power area **126** includes the instruction-processing portion of processor **102**. Specifically, core power area **126** includes arithmetic-logic unit **104**, register files **106**, pipelines **108**, and possibly level one (L**1**) caches **110**. Note that L**1** caches **110** can alternatively be located in non-core power area **124**.

Arithmetic-logic unit **104** provides computational and logical operations for processor **102**. Register files **106** provide source operands, intermediate storage, and destination locations for instructions being executed by arithmetic-logic unit **104**. Pipelines **108** provides a steady stream of instructions to arithmetic-logic unit **104**. Instructions in pipelines **108** are decoded in transit. Therefore, pipelines **108** may contain instructions in various stages of decoding and execution. L**1** caches **110** include data caches and instruction caches for arithmetic-logic unit **104**. L**1** caches **110** are comprised of very high-speed memory to provide fast access for instructions and data. In one embodiment of the present invention, L**1** caches **110** includes a write-through data cache.

Non-core power area **124** comprises the remaining portion of processor **102** and includes interrupt processor **112**, real-time clock **114**, clock distribution circuitry **116**, level two (L**2**) caches **118**, cache tags **120**, and cache snoop circuitry **122**. In general, non-core power area **124** includes portions of processor **102** that are not directly involved in processing instructions, and that need to operate while instruction processing is halted.

Interrupt processor **112** monitors interrupts **128** and periodically interrupts the execution of applications to provide services to external devices requiring immediate attention. Interrupt processor **112** can also provide a wake-up signal to core power area **126** as described below. Real-time clock **114** provides time-of-day services to processor **102**. Typically, real-time clock **114** is set upon startup from a battery operated real-time clock in the computer and thereafter provides time to the system. Clock distribution circuitry **116** provides clock signals for processor **102**. Distribution of these clock signals can be switched off or reduced for various parts of processor **102**. For example, clock distribution to core power area **126** can be stopped while the clock signals to non-core power area **124** continue. The acts of starting and stopping of these clock signals are known in the art and will not be described further. Real-time clock **114** and clock distribution circuitry **116** receive clock signal **130** from the computer system. Clock signal **130** is the master clock signal for the system.

L**2** cache **118** provides a second level cache for processor **102**. Typically, an L**2** cache is larger and slower that an L**1** cache, but still provides faster access to instructions and data than can be provided by main memory. Cache tags **120** provide an index into data stored in L**2** cache **118**. Cache snoop circuitry **122** invalidates cache lines base primarily on other processors accessing their own cache lines, or I/O devices doing memory transfers, even when instruction processing has been halted. L**2** cache **118**, cache tags **120**, and cache snoop circuitry **122** communicate with the computer system through memory signals **132**.

Non-core power area **124** receives non-core power **136** and core power area **126** receives core power **134**. The voltage applied for non-core power **136** remains at a voltage that allows circuitry within non-core power area **124** to remain fully active at all times. In contrast, non-core power **136** may provide different voltages to non-core power area **124** based upon the operating mode of processor **102**. For example, if processor **102** is a laptop attached to external electrical power, the voltage provided to non-core power **136** (and to core power **134** during instruction processing) may be higher than the minimum voltage, thus providing faster execution of programs.

The voltage applied to core power **134** remains sufficiently high during instruction processing so that core power area **126** remains fully active. However, when processor **102** receives a signal that processing can be suspended, the voltage supplied by core power **134** can be reduced.

In one embodiment of the present invention, the voltage in core power **134** is reduced to the minimum value that will maintain state information within core power area **126**, but this voltage is not sufficient to allow processing to continue. In another embodiment of the present invention, the voltage at core power **134** is reduced to zero. In this embodiment, the state of core power area **126** is first saved before the voltage is reduced to zero. This state can be saved in a dedicated portion of L**2** cache **118**, in main memory, or in another dedicated storage area. Upon receiving an interrupt or other signal indicating that processing is to resume, the voltage in core power **134** is restored to a normal level, saved state is restored, and processing is restarted.

FIG. **1**B illustrates an alternative partitioning of power areas within processor **102** in accordance with an embodiment of the present invention. As shown in FIG. **1**B, L**2** cache **118**, cache tags **120**, and cache snoop circuitry **122** are included in core power area **126** rather than in non-core power area **124**. In this embodiment, the voltage supplied as core power **134** is reduced or set to zero as described above, however, the cache circuitry within processor **102** is also put into the reduced power mode. Prior to reducing the voltage supplied to core power area **126**, data stored in L**2** cache **118** is flushed to main memory. Additionally, if the voltage at core power **134** is reduced to zero, the state of processor **102** is first saved in main memory.

Monitoring and Switching

FIG. **2** is a flowchart illustrating the process of monitoring processor load and switching to power saving modes in accordance with an embodiment of the present invention. The system starts by monitoring the processor load (step **202**). Next, the system determines if the processor will be needed soon (step **204**). This determination is made based on the current execution pattern and the cost of entering and recovering from nap mode. This cost, calculated in power usage, must be less than the power wasted by not going into nap mode. If the processor will be needed soon at step **204**, the process returns to step **202** to continue monitoring the processor load.

If the processor will not be needed soon at step **204**, the system determines if the processor has been taking long naps recently (step **206**). If not, the system enters a normal nap mode, which involves halting the processor without reducing any voltages (step **208**). Typically, halting the processor involves removing the clock signals to the core power area of the processor. After halting the processor, the system waits for an interrupt (step **210**). Upon receiving an interrupt or other signal requiring a restart, the system restarts instruction processing (step **212**). After restarting instruction processing, the process returns to step **202** to continue monitoring the processor load.

If the processor has recently been taking long naps at step **206**, the system enters a deep nap mode, which involves saving the state information from the core power area (step **214**), halting the processor (step **216**), and then reducing the voltage supplied to the core power area (step **218**). After reducing the voltage, the system waits for an interrupt (step **220**).

Upon receiving the interrupt or other signal requiring a restart, the system restores the voltage to the core power area (step **222**). Next, the modules within the core power area are restarted (step **224**). The system then restores the state information that was saved at step **214** (step **226**). After the processor has been restarted, the process returns to step **202** to continue monitoring the processor load. Note that the above description applies when the processor is used to save and restore the state information. In cases where dedicated hardware saves and restores the state information, steps **214** and **216**, and steps **224** and **226** can be reversed. Note also that if the voltage supplied to the core power area **126** is reduced but maintained at a level where modules in the core power do not lose state information, steps **216** and **224** are not required.

The foregoing descriptions of embodiments of the present invention have been presented for purposes of illustration and description only. They are not intended to be exhaustive or to limit the present invention to the forms disclosed. Accordingly, many modifications and variations will be apparent to practitioners skilled in the art. Additionally, the above disclosure is not intended to limit the present invention. The scope of the present invention is defined by the appended claims.

What is claimed is:

1. An instruction-processing system with minimal static power leakage, the instruction-processing system comprising:

- a core with instruction-processing circuitry;
- an area coupled to the core;
- a core voltage provided to the core; and
- an area voltage provided to the area;
- wherein in a normal operation mode:
  - a clock signal to the core is active;
  - the core voltage is a first value;
  - the core is active;
  - the area voltage is a second value; and
  - the area is active;
- wherein in a first power-saving mode that is exited upon receipt of an interrupt signal:
  - the clock signal to the core is inactive;
  - the core voltage is equal to or greater than the first value; and
  - the area voltage is equal to or greater than the second value;
- wherein in a second power-saving mode that can be exited upon receipt of a signal that is not an interrupt signal:
  - the clock signal to the core is inactive;
  - the core voltage is less than the first value; and
  - the area voltage is equal to or greater than the second value.

2. The instruction-processing system of claim **1**, wherein the first power-saving mode can be exited upon receipt of a signal that is not an interrupt signal.

3. The instruction-processing system of claim **1**, wherein the area comprises a cache.

4. The instruction-processing system of claim **3**, wherein the area further comprises cache tags.

5. The instruction-processing system of claim **1**, wherein prior to entering the second power-saving mode, the state of the core is saved to a memory.

6. The instruction-processing system of claim **1**, wherein upon exiting the second power-saving mode, the state of the core is restored.

7. The instruction-processing system of claim **1**, wherein in the second power-saving mode, the core voltage is at zero.

8. A method for minimizing static power leakage in an instruction-processing system, wherein the instruction-processing system comprises a core with instruction-processing circuitry, an area coupled to the core, a core voltage provided to the core, and an area voltage provided to the area, the method comprising:

- entering a normal operation mode by:
  - providing a clock signal to the core;
  - providing the core with a core voltage that is equal to a first value;
  - providing the area with an area voltage that is equal to a second value;
- entering a first power-saving mode by:
  - disabling the clock signal to the core;
  - providing the core with a core voltage that is equal to or greater than the first value; and
  - providing the area with an area voltage that is equal to or greater than the second value;
- exiting the first power-saving mode upon receipt of an interrupt signal;
- entering a second power-saving mode by:
  - disabling the clock signal to the core;
  - setting the core voltage to a value less than the first value; and
  - providing the area with an area voltage that is equal to or greater than the second value; and
- exiting the second power-saving mode upon receipt of a signal that is not an interrupt signal.

9. The method of claim **8**, further comprising exiting the first power-saving mode upon receipt of a signal that is not an interrupt signal.

10. The instruction-processing system of claim **8**, wherein the area comprises a cache.

11. The method of claim **10**, wherein the area further comprises cache tags.

12. The method of claim **8**, further comprising saving the state of the core to a memory prior to entering the second power-saving mode.

13. The method of claim **8**, further comprising restoring the state of the core upon exiting the second power-saving mode.

14. The method of claim **8**, wherein in the second power-saving mode, setting the core voltage to the value less than the first value comprises setting the core voltage to zero.

15. A computer-readable medium containing data representing an instruction-processing system with minimal static power leakage, the instruction- processing system comprising:

- a core with instruction-processing circuitry;
- an area coupled to the core;
- a core voltage provided to the core; and
- an area voltage provided to the area;
- wherein in a normal operation mode:
  - a clock signal to the core is active;
  - the core voltage is a first value;
  - the core is active;

the area voltage is a second value; and
the area is active;
wherein in a first power-saving mode that is exited upon receipt of an interrupt signal:
the clock signal to the core is inactive;
the core voltage is equal to or greater than the first value; and
the area voltage is equal to or greater than the second value;
wherein in a second power-saving mode that can be exited upon receipt of a signal that is not an interrupt signal:
the clock signal to the core is inactive;
the core voltage is less than the first value; and
the area voltage is equal to or greater than the second value.

16. The computer-readable medium of claim **15**, wherein the first power-saving mode can be exited upon receipt of a signal that is not an interrupt signal.

17. The computer-readable medium of claim **15**, wherein the area comprises a cache.

18. The computer-readable medium of claim **17**, wherein the area further comprises cache tags.

19. The computer-readable medium of claim **15**, wherein prior to entering the second power-saving mode, the state of the core is saved to a memory.

20. The computer-readable medium of claim **15**, wherein upon exiting the second power-saving mode, the state of the core is restored.

21. The computer-readable medium stem of claim **15**, wherein in the second power-saving mode, the core voltage is at zero.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 7,383,453 B2
APPLICATION NO. : 11/213215
DATED : June 3, 2008
INVENTOR(S) : Lynn R. Youngs

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Title Page item [73]

In the Assignee Name (on page 1), please delete "Apple, Inc.".

In the Assignee name (on page 1), please insert --APPLE INC.--.

Signed and Sealed this

Thirteenth Day of January, 2009

Jon W. Dudas

JON W. DUDAS
Director of the United States Patent and Trademark Office

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 7,383,453 B2
APPLICATION NO. : 11/213215
DATED : June 3, 2008
INVENTOR(S) : Lynn R. Youngs

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Title page Item [73]
In the Assignee Name (on page 1), please delete “Apple, Inc.”.

Title page Item [73]
In the Assignee Name (on page 1), please insert --APPLE INC.--.

Signed and Sealed this

Seventeenth Day of February, 2009

John Doll

JOHN DOLL
Acting Director of the United States Patent and Trademark Office

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 7,383,453 B2
APPLICATION NO. : 11/213215
DATED : June 3, 2008
INVENTOR(S) : Lynn R. Young

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

In claim 1 (at column 5, line 36), please delete the word “minimal” and insert the word, --minimized-- so the line reads “A instruction-processing system with minimized static”.

In claim 1 (at column 5, line 45), please insert the words, --that is sufficient to maintain the state information of the instruction-processing circuitry-- so the line reads “the core voltage is a first value that is sufficient to maintain the state information of the instruction-processing circuitry”.

In claim 1 (at column 5, line 47), please insert the words, --that is sufficient to maintain the data stored in the area-- so the line reads “the area voltage is a second value that is sufficient to maintain the data stored in the area”.

In claim 1 (at column 5, line 49), please delete the word “is” and insert the words, --can be-- so the line reads “wherein in a first power-saving mode that can be exited upon”.

In claim 1 (at column 5, line 53), please delete the words “equal to or greater than the first value” and insert the words, --sufficient to maintain the state information of the instruction-processing circuitry-- so the line reads “the core voltage is sufficient to maintain the state information of the instruction-processing circuitry”.

In claim 1 (at column 5, line 55), please delete the words “equal to or greater than the second value” and insert the words, --sufficient to maintain the data stored in the area-- so the line reads “the area voltage is sufficient to maintain the data stored in the area”.

In claim 1 (at column 5, line 61), please delete the words “equal to or greater than the second value” and insert the words, --sufficient to maintain the data stored in the area-- so the line reads “the area voltage is sufficient to maintain the data stored in the area”.

In claim 8 (at column 6, line 19), please delete the words “that is” so the line reads “providing the core with a core voltage equal to”.

Signed and Sealed this

Eighth Day of June, 2010

David J. Kappos

David J. Kappos
Director of the United States Patent and Trademark Office

In claim 8 (at column 6, line 20), please insert the words, --that is sufficient to maintain the state information of the instruction-processing circuitry-- so the line reads “a first value that is sufficient to maintain the state information of the instruction-processing circuitry”.

In claim 8 (at column 6, line 21), please delete the words “that is” so the line reads “providing the area with an area voltage equal to”.

In claim 8 (at column 6, line 22), please insert the words, --that is sufficient to maintain the data stored in the area-- so the line reads “a second value that is sufficient to maintain the data stored in the area”.

In claim 8 (at column 6, line 25), please delete the words “equal to” and insert the words, --sufficient to maintain the state information of the instruction-processing circuitry-- so the line reads “providing the core with a core voltage that is sufficient to maintain the state information of the instruction-processing circuitry”.

In claim 8 (at column 6, line 26), please delete the words “or greater than the first value”.

In claim 8 (at column 6, line 27), please delete the words “equal to” and insert the words, --sufficient to maintain the data stored in the area-- so the line reads “providing the area with an area voltage that is sufficient to maintain the data stored in the area”.

In claim 8 (at column 6, line 28), please delete the words “or greater than the second value”.

In claim 10 (at column 6, line 42), please delete the words “instruction-processing system” and insert the word, --method-- so the line reads “the method of claim 8, wherein”.

In claim 15 (at column 6, line 56), please delete the words “containing data repre-” and insert the words, --storing code which represents-- so the line reads “A computer-readable medium storing code which represents”.

In claim 15 (at column 6, line 57), please delete the words “senting” and “minimal” and insert the word, --minimized-- so the line reads “an instruction-processing system with minimized static”.

In claim 15 (at column 6, line 66), please insert the words, --that is sufficient to maintain the state information of the instruction-processing circuitry-- so the line reads “the core voltage is a first value that is sufficient to maintain the state information of the instruction-processing circuitry”.

In claim 15 (at column 7, line 1), please insert the words, --that is sufficient to maintain the data stored in the area-- so the line reads “the area voltage is a second value that is sufficient to maintain the data stored in the area”.

In claim 15 (at column 7, line 3), please delete the word “is” and insert the words, --can be-- so the line reads “wherein in a first power-saving mode that can be exited upon”.

In claim 15 (at column 7, line 6), please delete the words "equal to or greater than the first" and insert the words, --sufficient to maintain the state information of the instruction-processing circuitry-- so the line reads "the core voltage is sufficient to maintain the state information of the instruction-processing circuitry".

In claim 15 (at column 7, line 7), please delete the word "value".

In claim 15 (at column 7, line 8), please delete the words "equal to or greater than the second" and insert the words, --sufficient to maintain the data stored in the area-- so the line reads "the area voltage is sufficient to maintain the data stored in the area".

In claim 15 (at column 7, line 9), please delete the word "value".

In claim 15 (at column 7, line 14), please delete the words "is equal to or greater than the second" and insert the words, --sufficient to maintain the data stored in the area-- so the line reads "the area voltage is sufficient to maintain the data stored in the area".

In claim 15 (column 7, line 15), delete the word "value".

In claim 21 (column 8, line 12), please delete the word "stem" so the line reads "The computer-readable medium of claim 15".

Exhibit B

US007657849B2

(12) **United States Patent**
Chaudhri et al.

(10) **Patent No.: US 7,657,849 B2**
(45) **Date of Patent: Feb. 2, 2010**

(54) **UNLOCKING A DEVICE BY PERFORMING GESTURES ON AN UNLOCK IMAGE**

(75) Inventors: **Imran Chaudhri**, San Francisco, CA (US); **Bas Ording**, San Francisco, CA (US); **Freddy Allen Anzures**, San Francisco, CA (US); **Marcel Van Os**, San Francisco, CA (US); **Stephen O. Lemay**, San Francisco, CA (US); **Scott Forstall**, Mountain View, CA (US); **Greg Christie**, San Jose, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 394 days.

(21) Appl. No.: **11/322,549**

(22) Filed: **Dec. 23, 2005**

(65) **Prior Publication Data**

US 2007/0150842 A1 Jun. 28, 2007

(51) **Int. Cl.**
G06F 3/033 (2006.01)

(52) **U.S. Cl.** **715/863**; 345/173; 345/179

(58) **Field of Classification Search** 713/154, 713/156, 182; 715/853, 863; 345/173, 179, 345/156
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,465,084 A 11/1995 Cottrell 340/825.31
5,559,961 A 9/1996 Blonder 395/188.01
5,677,710 A 10/1997 Thompson-Rohrlich 345/173
5,821,933 A * 10/1998 Keller et al. 715/741
5,907,327 A 5/1999 Ogura et al. 345/339
6,151,208 A 11/2000 Bartlett 361/683
6,160,555 A 12/2000 Kang et al. 345/358
6,192,478 B1 2/2001 Elledge 713/202
6,249,606 B1 6/2001 Kiraly et al. 702/195
6,323,846 B1 11/2001 Westerman et al. 345/173
6,347,290 B1 2/2002 Bartlett 702/150

(Continued)

FOREIGN PATENT DOCUMENTS

EP 1 284 450 A2 2/2003

(Continued)

OTHER PUBLICATIONS

GridLock 1.32, Oct. 8, 2003, pp. 1-2, http://gridlock.en.softonic.com/palm.*

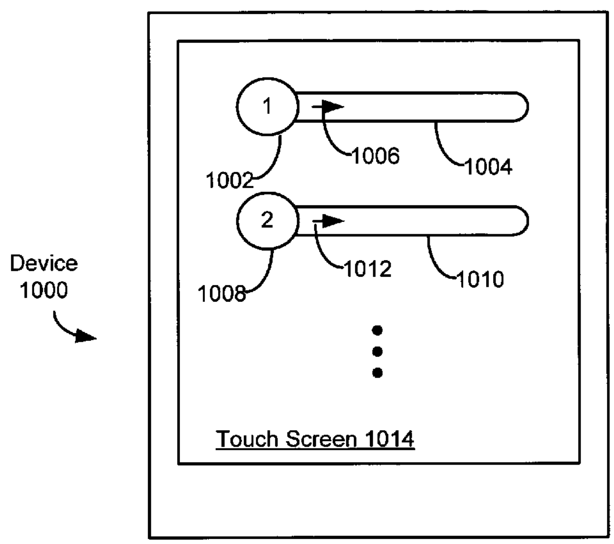
(Continued)

Primary Examiner—Dennis-Doon Chow
Assistant Examiner—Andres E Gutierrez
(74) *Attorney, Agent, or Firm*—Morgan, Lewis & Bockius LLP

(57) **ABSTRACT**

A device with a touch-sensitive display may be unlocked via gestures performed on the touch-sensitive display. The device is unlocked if contact with the display corresponds to a predefined gesture for unlocking the device. The device displays one or more unlock images with respect to which the predefined gesture is to be performed in order to unlock the device. The performance of the predefined gesture with respect to the unlock image may include moving the unlock image to a predefined location and/or moving the unlock image along a predefined path. The device may also display visual cues of the predefined gesture on the touch screen to remind a user of the gesture.

23 Claims, 15 Drawing Sheets



U.S. PATENT DOCUMENTS

6,421,453	B1	7/2002	Kanevsky et al.	382/115
6,570,557	B1	5/2003	Westerman et al.	345/173
6,573,883	B1	6/2003	Bartlett	345/156
6,633,310	B1	10/2003	Andrew et al.	345/728
6,677,932	B1	1/2004	Westerman	345/173
6,720,860	B1	4/2004	Narayanaswami	340/5.54
6,735,695	B1	5/2004	Gopalakrishnan et al.	713/186
7,124,433	B2	10/2006	Little	726/2
7,151,843	B2	12/2006	Rui et al.	382/103
7,174,462	B2	2/2007	Pering et al.	713/182
7,263,670	B2	8/2007	Rekimoto	715/837
2001/0011308	A1	8/2001	Clark et al.	710/20
2001/0012022	A1	8/2001	Smith	345/768
2002/0015024	A1	2/2002	Westerman et al.	345/173
2002/0196274	A1	12/2002	Comfort et al.	345/741
2003/0142138	A1	7/2003	Brown et al.	345/797
2004/0030934	A1	2/2004	Mizoguchi et al.	713/202
2004/0034801	A1*	2/2004	Jaeger	713/202
2004/0085351	A1	5/2004	Tokkonen	345/741
2004/0088568	A1	5/2004	Tokkonen	713/200
2004/0230843	A1	11/2004	Jansen	713/202
2004/0250138	A1	12/2004	Schneider	713/202
2004/0260955	A1	12/2004	Mantyla	713/202
2004/0268267	A1	12/2004	Moravcsik	715/821
2005/0050477	A1	3/2005	Robertson et al.	715/853
2005/0060554	A1	3/2005	O'Donoghue	713/183
2005/0079896	A1	4/2005	Kokko et al.	455/566
2005/0212760	A1*	9/2005	Marvit et al.	345/156
2005/0216862	A1	9/2005	Shinohara et al.	715/825
2005/0248542	A1	11/2005	Sawanobori	345/173
2005/0253817	A1*	11/2005	Rytivaara et al.	345/173
2005/0264833	A1	12/2005	Hiraoka et al.	358/1.9
2006/0174339	A1	8/2006	Tao	726/18
2006/0267955	A1	11/2006	Hino	345/173
2008/0034292	A1	2/2008	Brunner et al.	715/700
2008/0072172	A1	3/2008	Shinohara et al.	715/772

FOREIGN PATENT DOCUMENTS

GB	2 313 460	A	11/1997
JP	60 171560		9/1985
JP	2 249062		10/1990
JP	6 214954		8/1994
WO	WO 02/33882	A1	4/2002
WO	WO 04/001560	A1	12/2003
WO	WO 2004/021108	A2	3/2004

OTHER PUBLICATIONS

A Passive-Style Buttonless Mobile Terminal; Youichi Horry, Issyu Nakajima, Takeshi Hoshino, and Yukinobu Maruyama; IEEE Transactions on Consumer Electronics; vol. 49; No. 3; Aug. 2003; pp. 530-535.*

PCT International Search Report for International Application No. PCT/US2006/061370, mailed May 25, 2007.

IBM Corp., "Access/Control Icons (Icon Keys)," IBM Technical Disclosure Bulletin, vol. 38, No. 4, Apr. 1995, pp. 407-409.

"Touch Password Protection," JGUI Professional, http://www.jgui.net/touch/index.html, Dec. 8, 2003.

Office Action dated Oct. 31, 2007 for related U.S. Appl. No. 11/322,550, filed Dec. 23, 2005, pp. 1-26.

Office Action dated Apr. 21, 2008 for related U.S. Appl. No. 11/322,550, filed Dec. 23, 2005, pp. 1-22.

International Search Report for International Application No. PCT/US2006/061380, mailed Apr. 23, 2007.

Notice of Allowance dated Sep. 19, 2008, for related U.S. Appl. No. 11/322,550.

Bardram, J., "The Trouble with Login: on usability and computer security in Ubiquitous Computing," Centre for Pervasive Healthcare, Department of Computer Science, University of Aahus, Published online: Jul. 23, 2005.

Fitzpatrick, G. et al., "Method for Access Control via Gestural Verification," IBM Technical Disclosure Bulletin, vol. 36, No. 09B, Sep. 1993, 2 pages.

IBM, "Touch Pad Authentication," Sep. 21, 2004, 2 pages.

Jansen, W., "Authenticating Users on Handheld Devices," The National Institute of Standards and Technology, Gaithersburg, Maryland, 2003, 13 pages.

Jermyn, I. et al., "The Design and Analysis of Graphical Passwords," Submission to the 8th Usenix Security Symposium, Aug. 1999, 15 pages.

Monrose, N., "Towards Stronger User Authentication," Ph.d dissertation, 1999, New York University, vol. 60/05-B of Dissertation Abstract International, Order No. AAD99-30229, 115 pages.

Najjar, L., "Graphical Passwords," International Technology Disclosures vol. 10, No. 1, Jan. 25, 1992, 1 page.

Renaud, K. et al., "My Password is Here! An Investigation into Visuo-Spatial Authentication Mechanisms," 2004, 25 pages, www.sciencedirect.com.

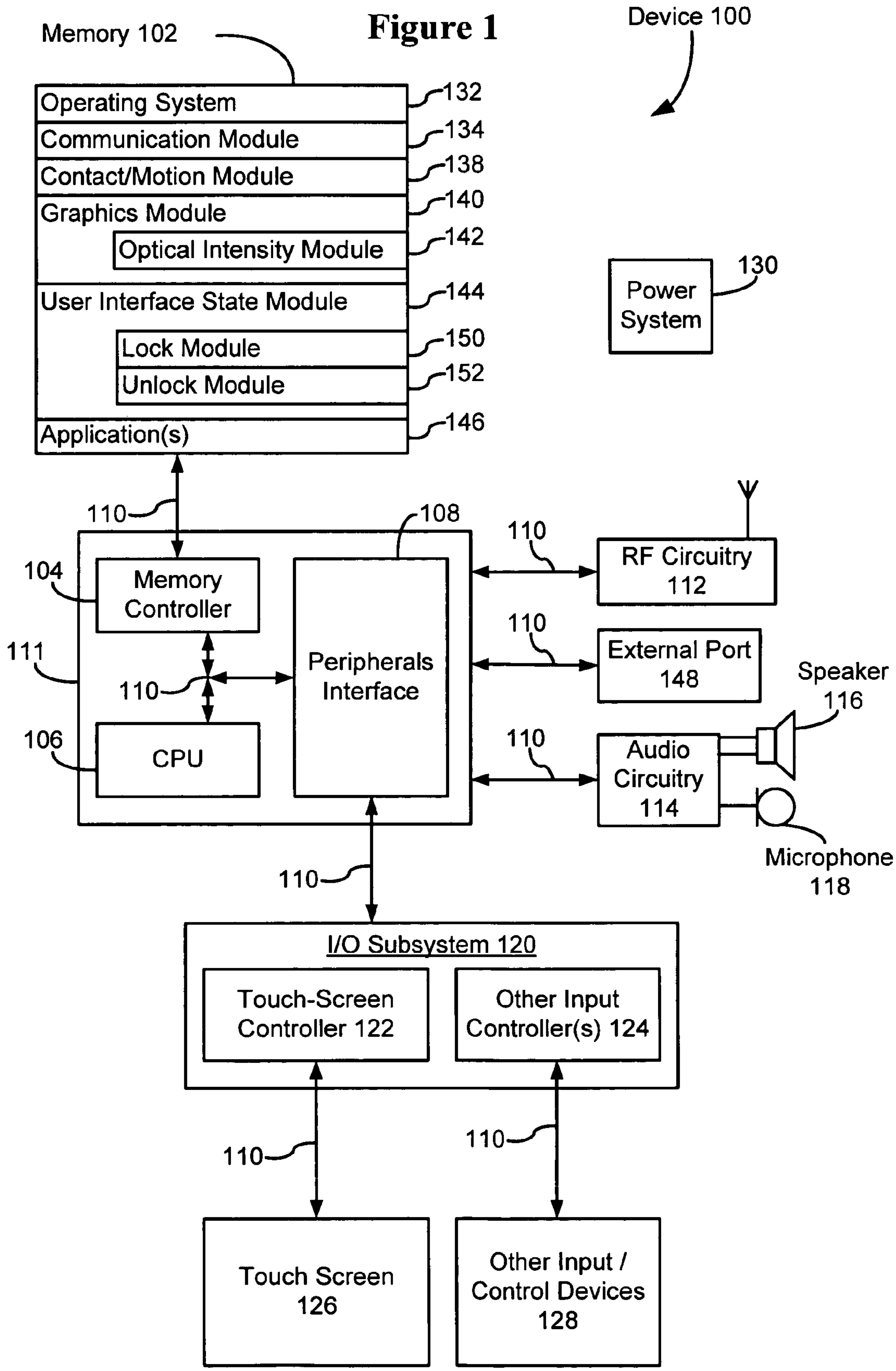
Wiedenbeck, S., et al. "PassPoints: Design and Longitudinal Evaluation of a Graphical Password System," International Journal of Human-Computer Studies, vol. 63, Issues 1-2, Jul. 2005, pp. 102-127.

Office Action dated Feb. 4, 2009, received in the German patent application which corresponds to U.S. Appl. No. 11/322,549.

Office Action dated Mar. 25, 2009, received in the European patent application which corresponds to U.S. Appl. No. 11/322,549.

* cited by examiner

Figure 1


Memory 102
Device 100
Operating System
132
Communication Module
134
Contact/Motion Module
138
Graphics Module
140
Optical Intensity Module
142
User Interface State Module
144
Lock Module
150
Unlock Module
152
Application(s)
146
Power System
130
110
108
104
Memory Controller
111
110
106
CPU
Peripherals Interface
RF Circuitry 112
External Port 148
Audio Circuitry 114
Speaker 116
Microphone 118
I/O Subsystem 120
Touch-Screen Controller 122
Other Input Controller(s) 124
Touch Screen 126
Other Input / Control Devices 128

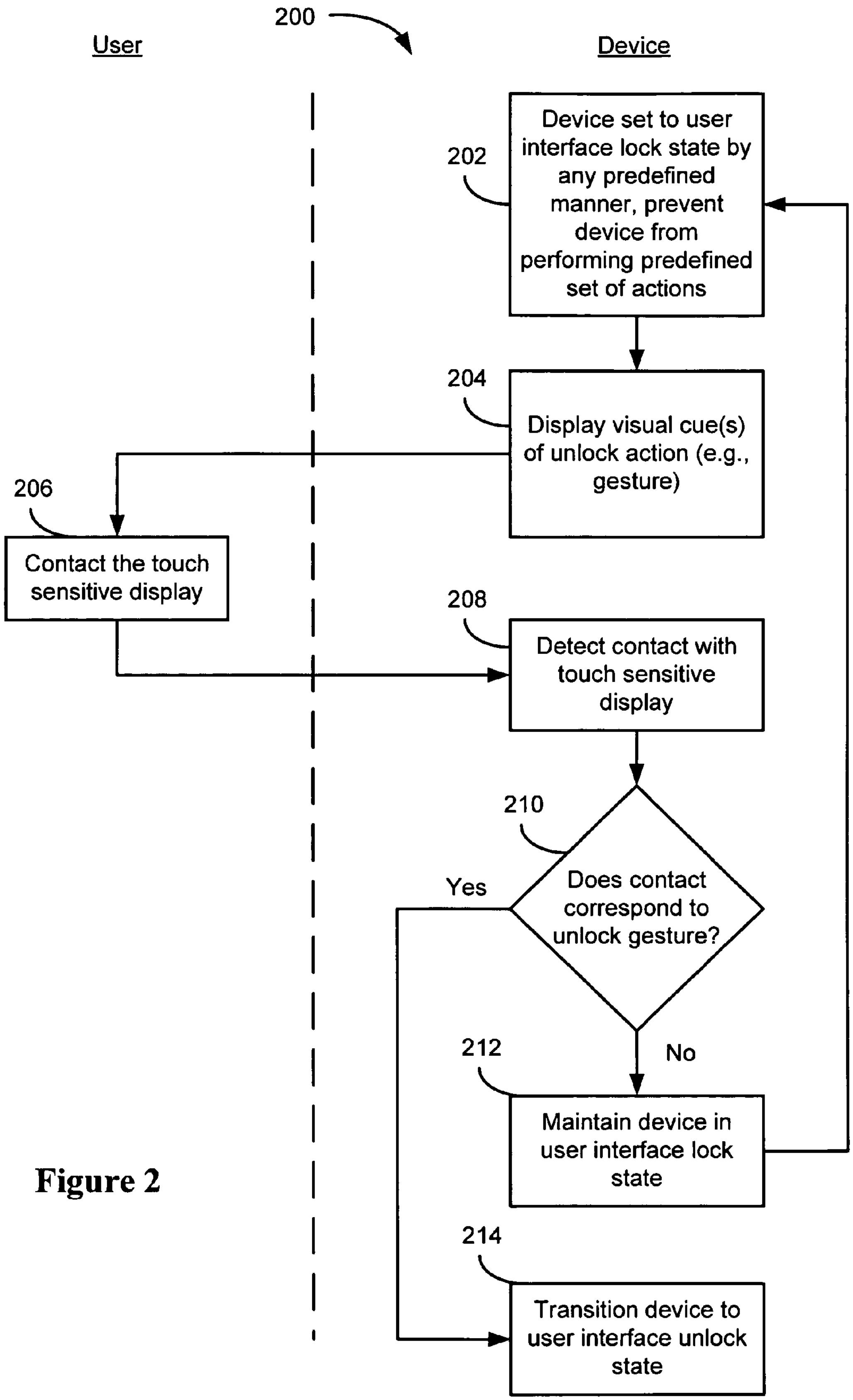

200
User
Device
202
Device set to user interface lock state by any predefined manner, prevent device from performing predefined set of actions
204
Display visual cue(s) of unlock action (e.g., gesture)
206
Contact the touch sensitive display
208
Detect contact with touch sensitive display
210
Does contact correspond to unlock gesture?
Yes
No
212
Maintain device in user interface lock state
214
Transition device to user interface unlock state

Figure 2

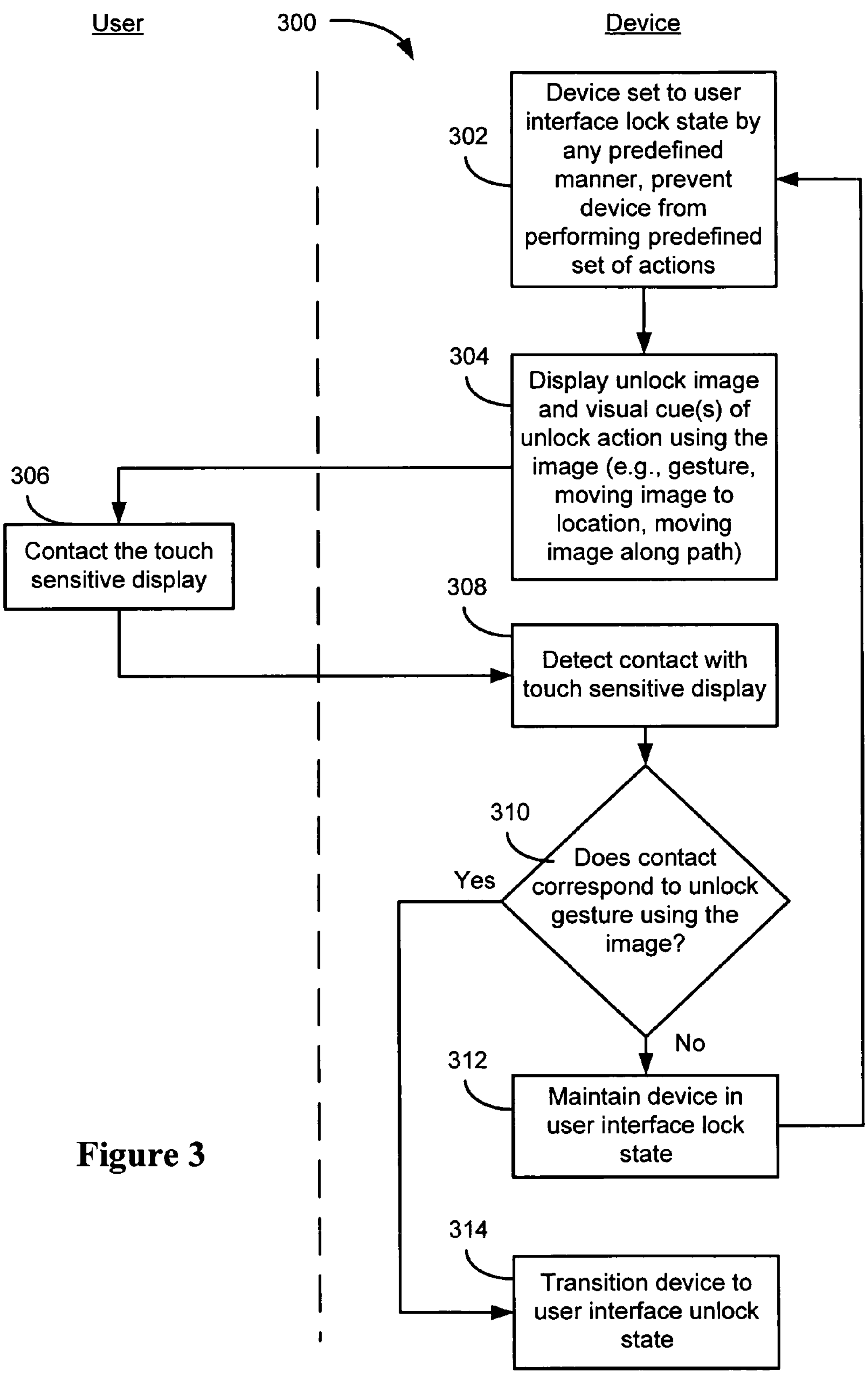

User
300
Device
302
Device set to user interface lock state by any predefined manner, prevent device from performing predefined set of actions
304
Display unlock image and visual cue(s) of unlock action using the image (e.g., gesture, moving image to location, moving image along path)
306
Contact the touch sensitive display
308
Detect contact with touch sensitive display
310
Does contact correspond to unlock gesture using the image?
Yes
No
312
Maintain device in user interface lock state
314
Transition device to user interface unlock state

Figure 3

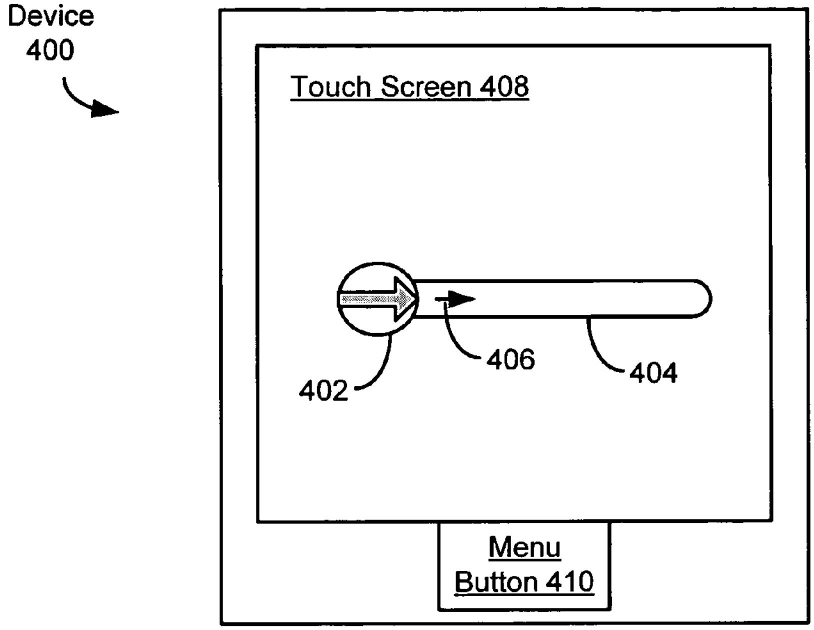


Figure 4A

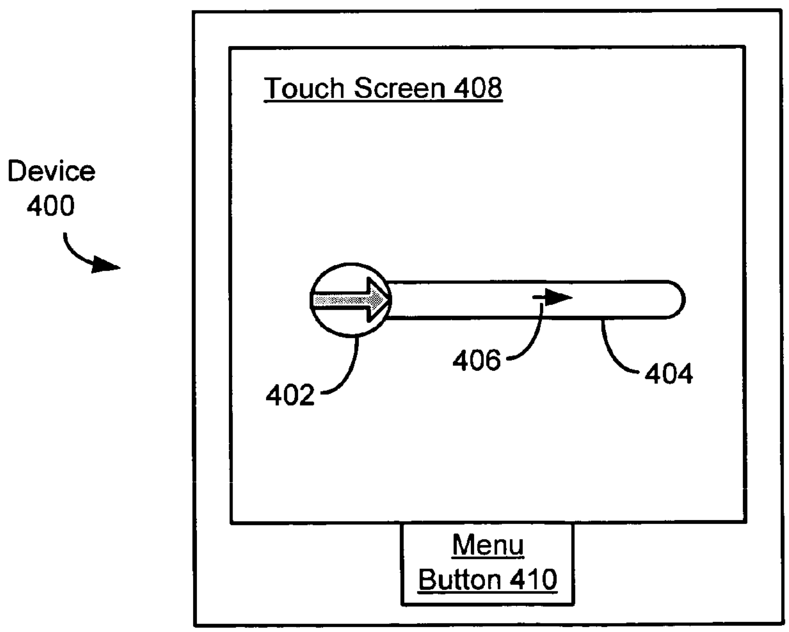


Figure 4B

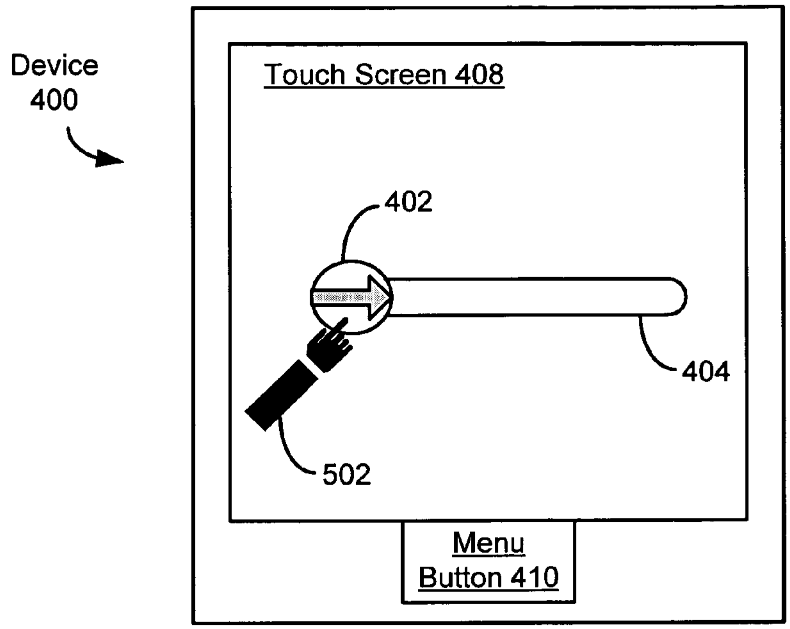


Figure 5A

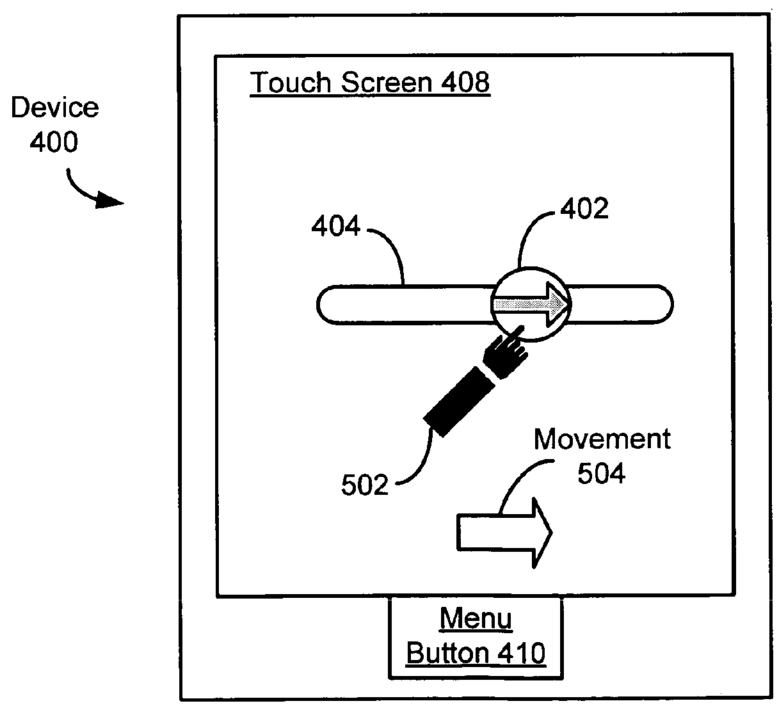


Figure 5B

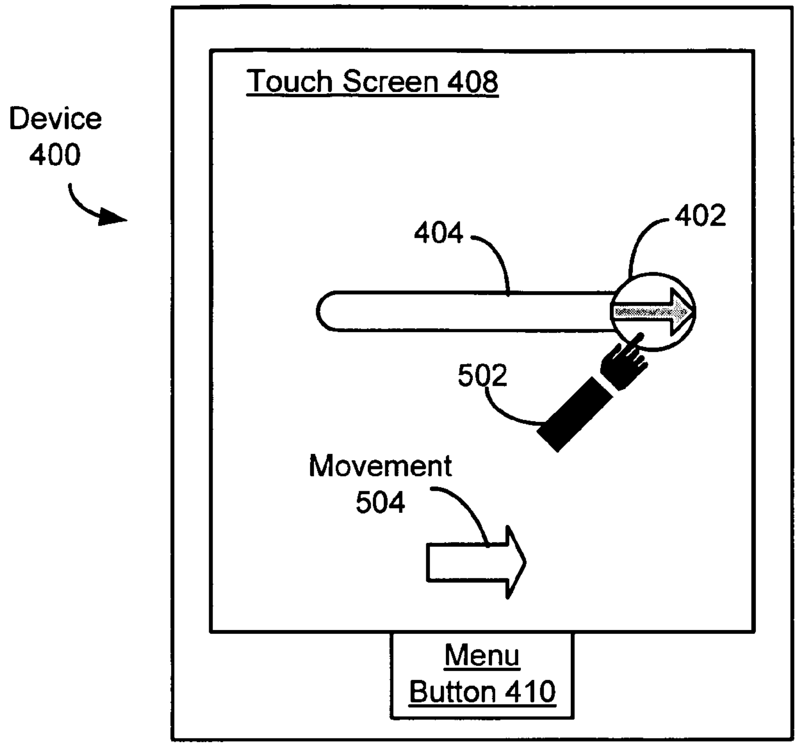


Figure 5C

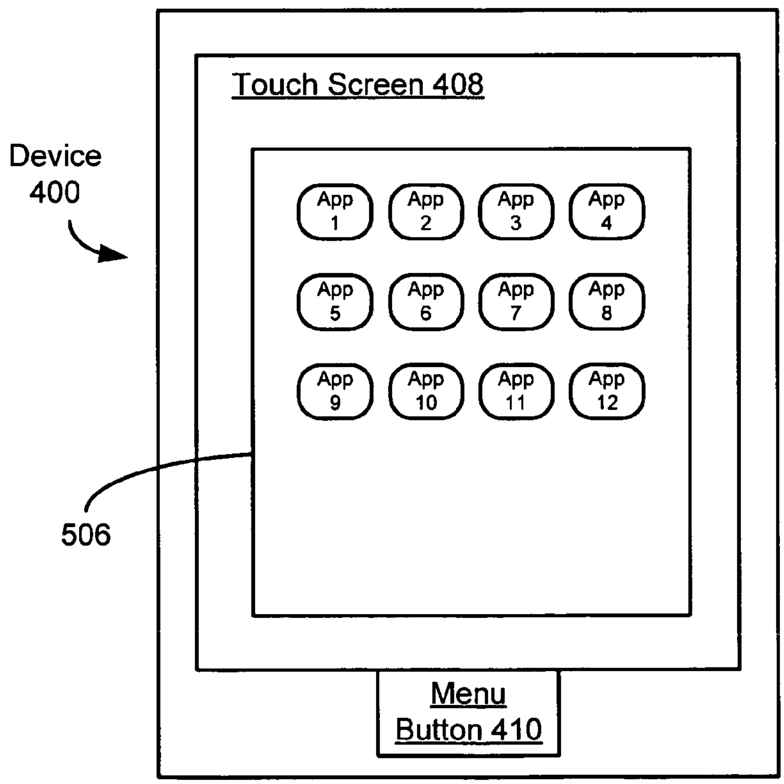


Figure 5D

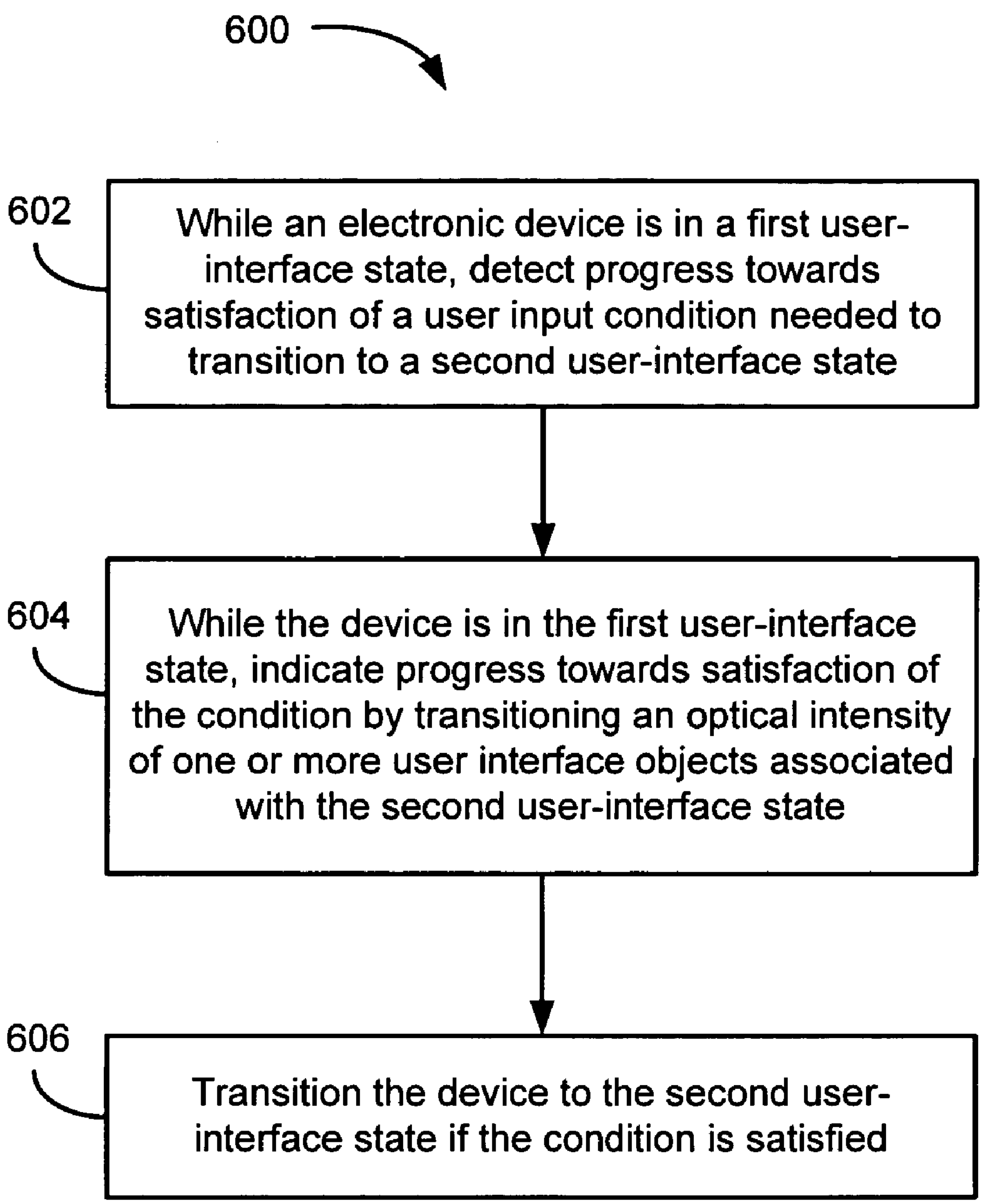

600
602
While an electronic device is in a first user-interface state, detect progress towards satisfaction of a user input condition needed to transition to a second user-interface state
604
While the device is in the first user-interface state, indicate progress towards satisfaction of the condition by transitioning an optical intensity of one or more user interface objects associated with the second user-interface state
606
Transition the device to the second user-interface state if the condition is satisfied

Figure 6

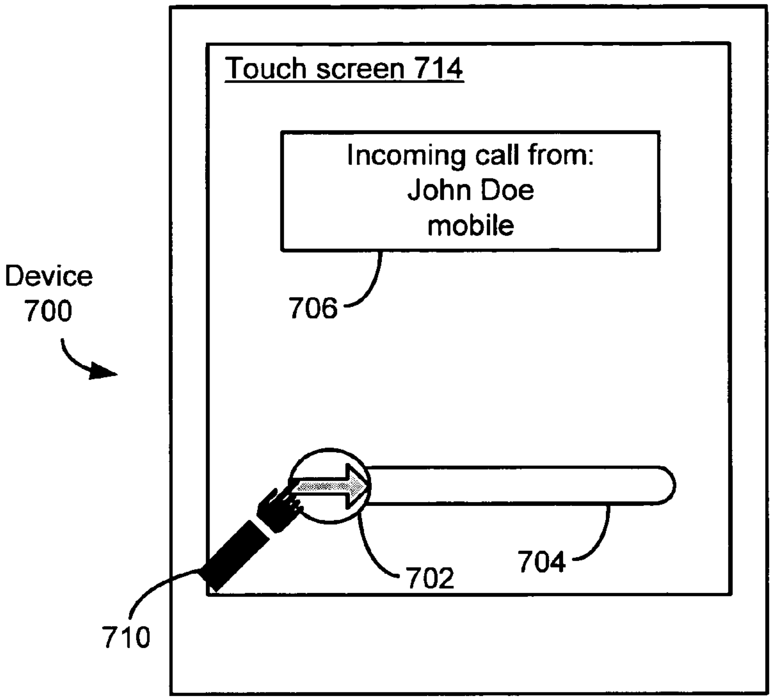


Figure 7A

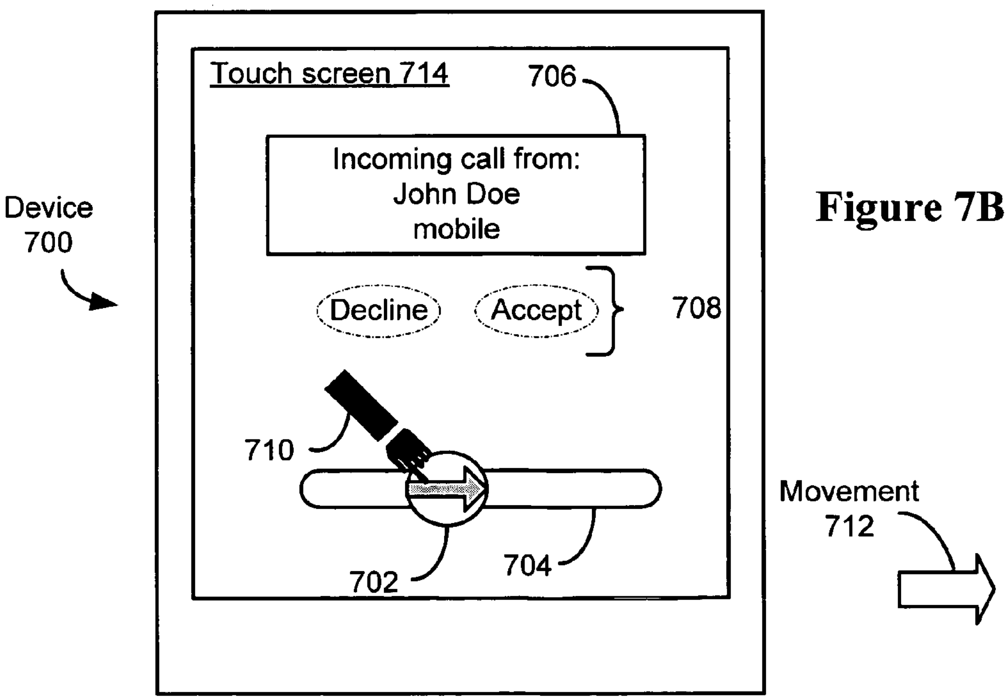


Figure 7B

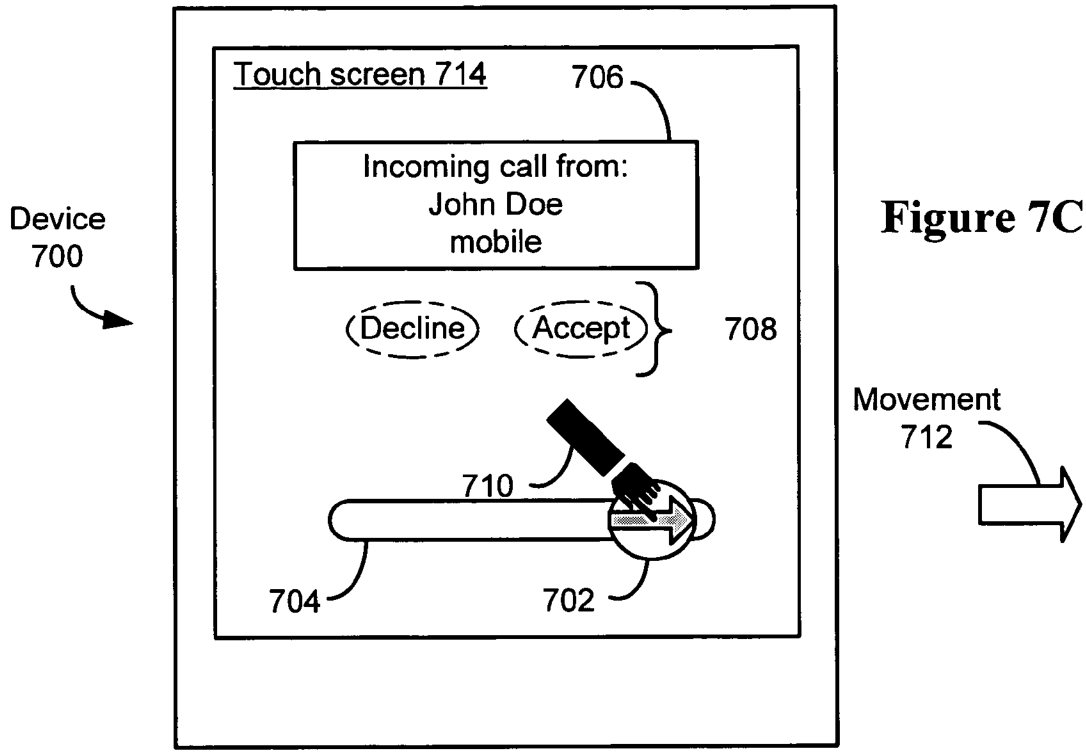


Figure 7C

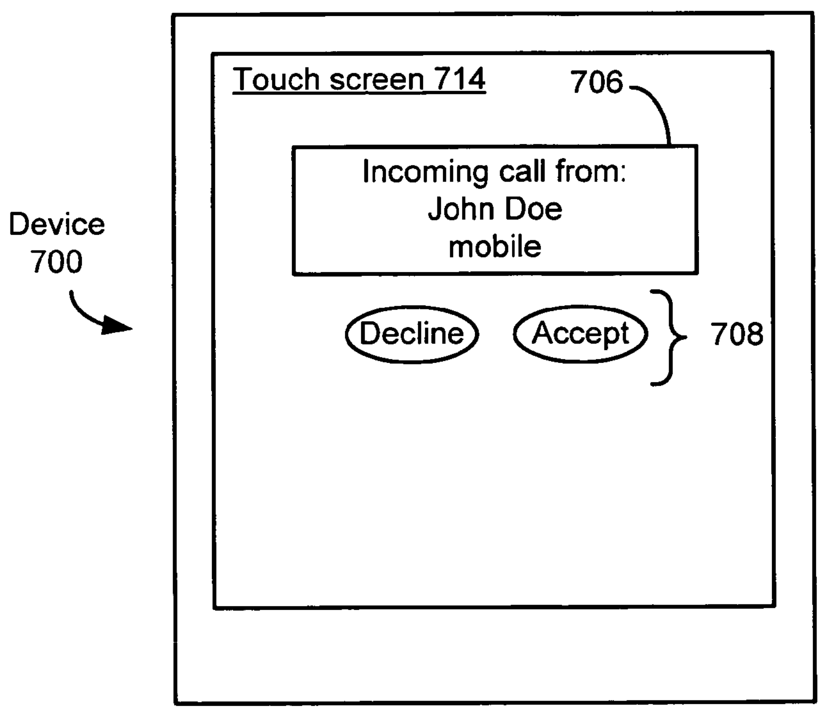


Figure 7D

Figure 8A

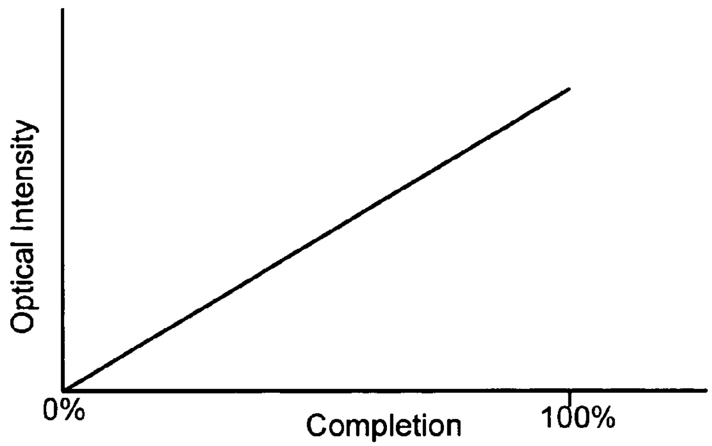


Figure 8B

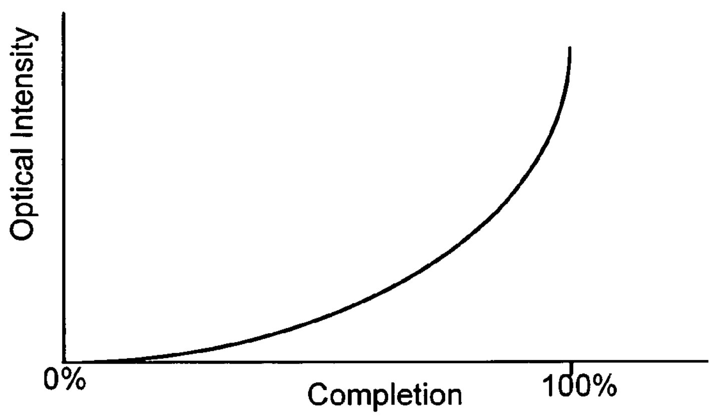
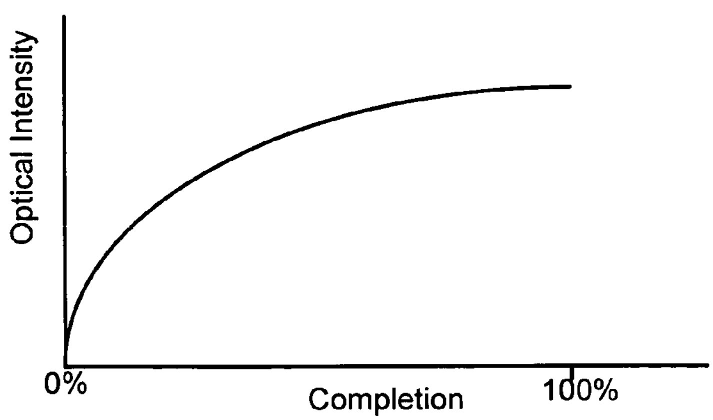


Figure 8C



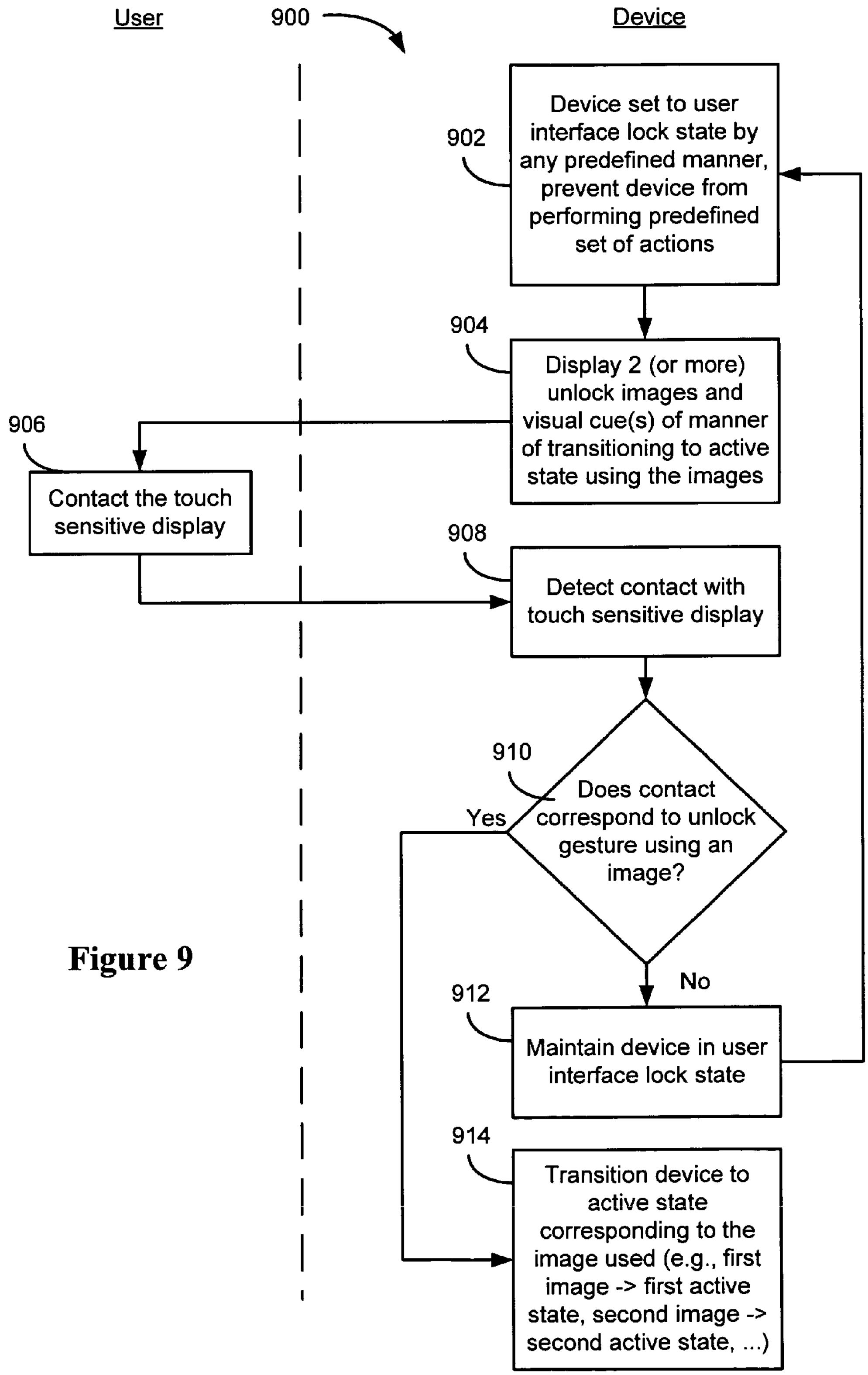

User
900
Device
902
Device set to user interface lock state by any predefined manner, prevent device from performing predefined set of actions
904
Display 2 (or more) unlock images and visual cue(s) of manner of transitioning to active state using the images
906
Contact the touch sensitive display
908
Detect contact with touch sensitive display
910
Does contact correspond to unlock gesture using an image?
Yes
No
912
Maintain device in user interface lock state
914
Transition device to active state corresponding to the image used (e.g., first image -> first active state, second image -> second active state, ...)

Figure 9

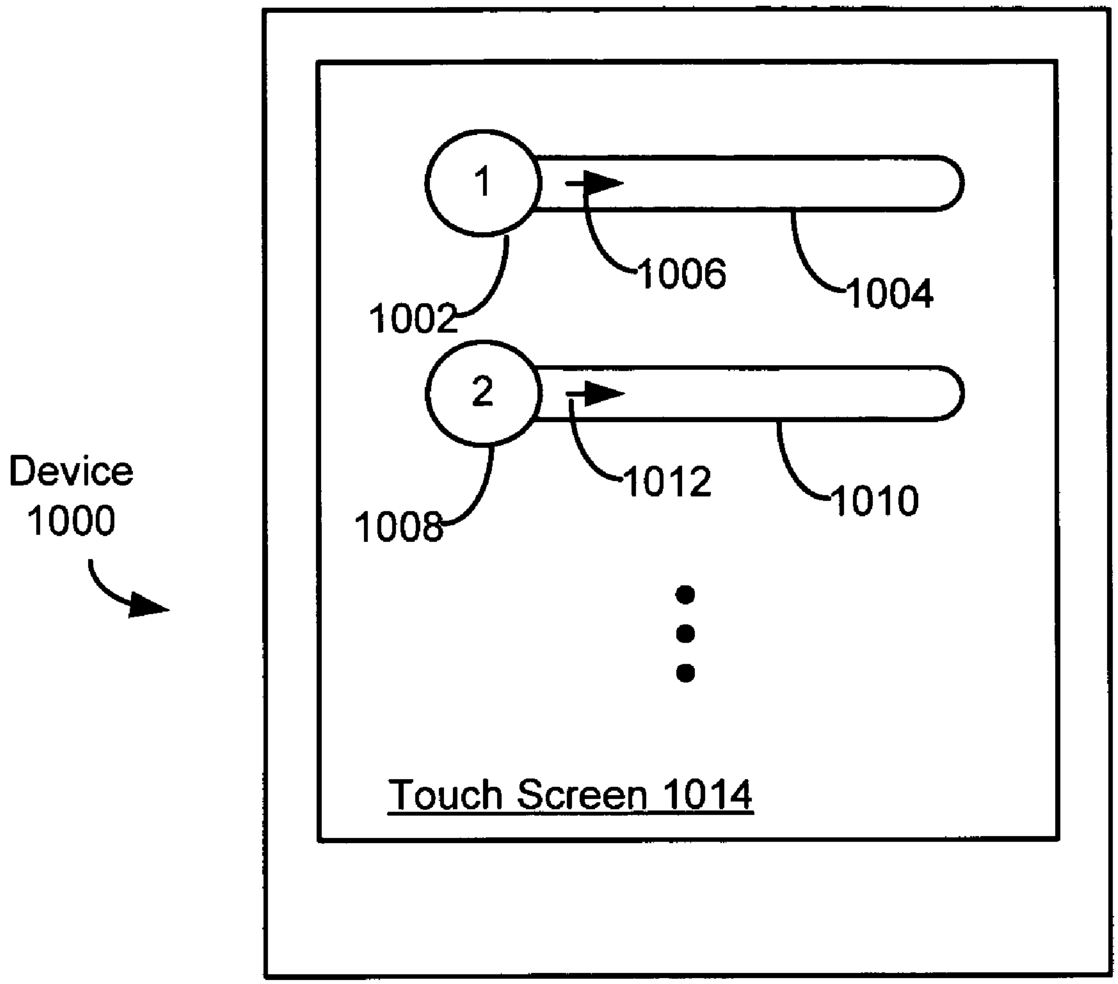

1
1006
1004
1002
2
1012
1010
1008
Device
1000
Touch Screen 1014

Figure 10

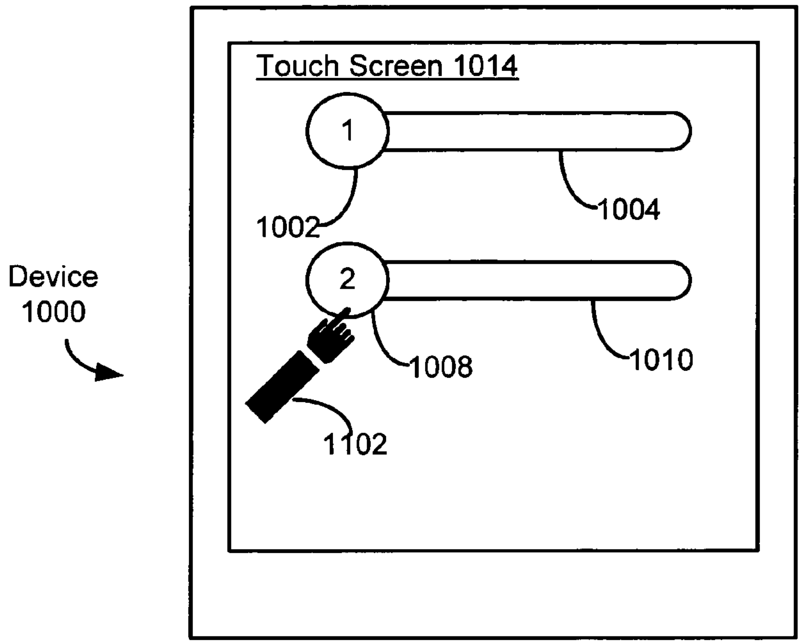


Figure 11A

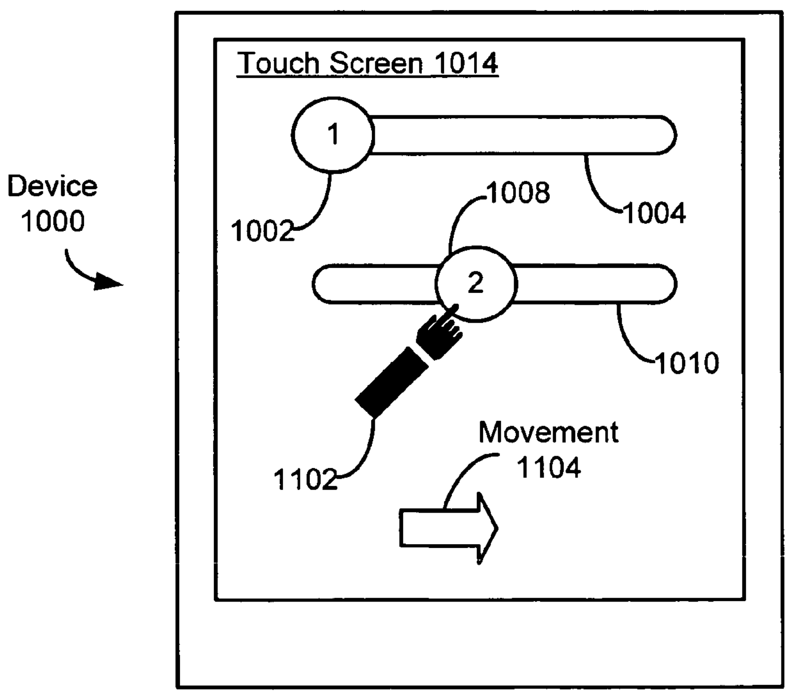


Figure 11B

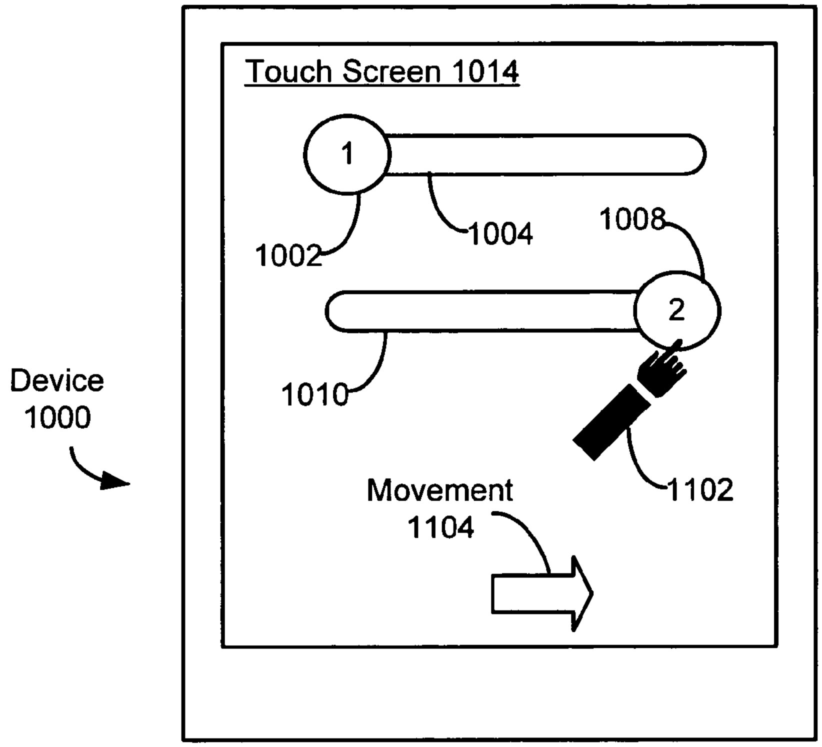


Figure 11C

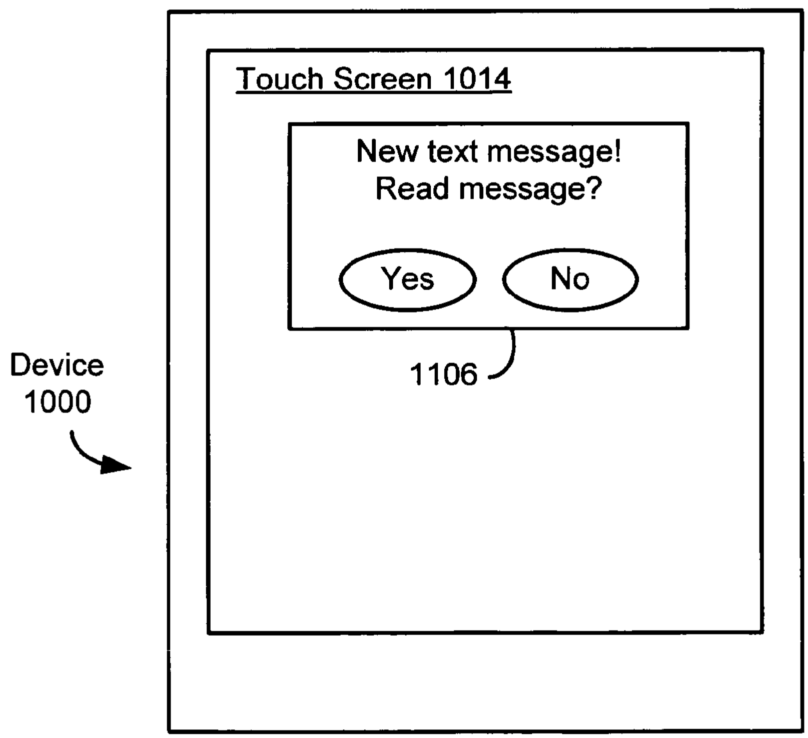


Figure 11D

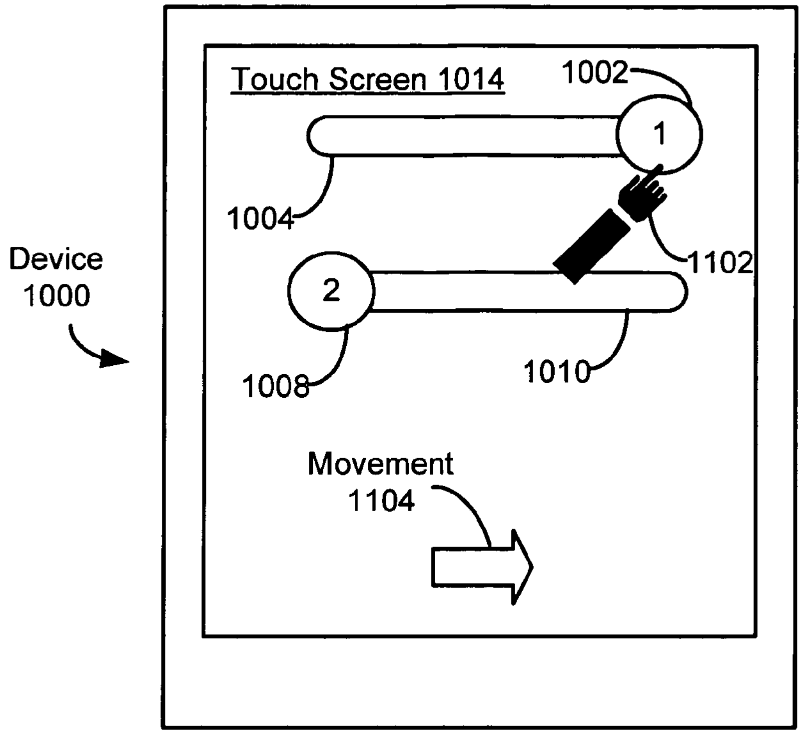


Figure 11E

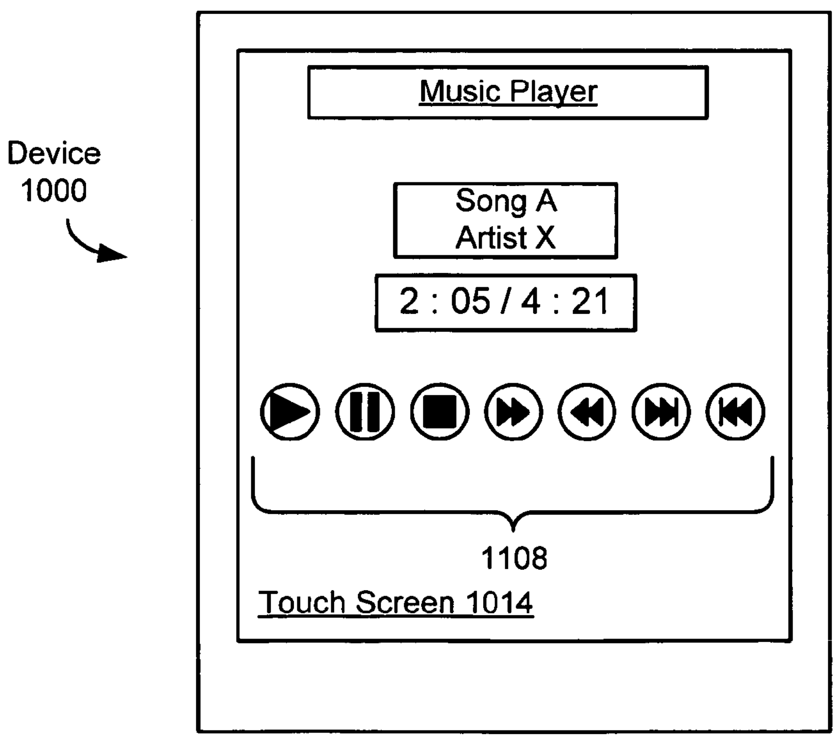


Figure 11F

UNLOCKING A DEVICE BY PERFORMING GESTURES ON AN UNLOCK IMAGE

RELATED APPLICATIONS

This application is related to U.S. patent application Ser. No. 11/322,550, titled "Indication of Progress Towards Satisfaction of a User Input Condition," filed Dec. 23, 2005, which application is incorporated by reference herein in its entirety.

TECHNICAL FIELD

The disclosed embodiments relate generally to user interfaces that employ touch-sensitive displays, and more particularly, to the unlocking of user interfaces on portable electronic devices.

BACKGROUND

Touch-sensitive displays (also known as "touch screens" or "touchscreens") are well known in the art. Touch screens are used in many electronic devices to display graphics and text, and to provide a user interface through which a user may interact with the devices. A touch screen detects and responds to contact on the touch screen. A device may display one or more soft keys, menus, and other user-interface objects on the touch screen. A user may interact with the device by contacting the touch screen at locations corresponding to the user-interface objects with which she wishes to interact.

Touch screens are becoming more popular for use as displays and as user input devices on portable devices, such as mobile telephones and personal digital assistants (PDAs). One problem associated with using touch screens on portable devices is the unintentional activation or deactivation of functions due to unintentional contact with the touch screen. Thus, portable devices, touch screens on such devices, and/or applications running on such devices may be locked upon satisfaction of predefined lock conditions, such as upon entering an active call, after a predetermined time of idleness has elapsed, or upon manual locking by a user.

Devices with touch screens and/or applications running on such devices may be unlocked by any of several well-known unlocking procedures, such as pressing a predefined set of buttons (simultaneously or sequentially) or entering a code or password. These unlock procedures, however, have drawbacks. The button combinations may be hard to perform. Creating, memorizing, and recalling passwords, codes, and the like can be quite burdensome. These drawbacks may reduce the ease of use of the unlocking process and, as a consequence, the ease of use of the device in general.

Accordingly, there is a need for more efficient, user-friendly procedures for unlocking such devices, touch screens, and/or applications. More generally, there is a need for more efficient, user-friendly procedures for transitioning such devices, touch screens, and/or applications between user interface states (e.g., from a user interface state for a first application to a user interface state for a second application, between user interface states in the same application, or between locked and unlocked states). In addition, there is a need for sensory feedback to the user regarding progress towards satisfaction of a user input condition that is required for the transition to occur.

SUMMARY

In some embodiments, a method of controlling an electronic device with a touch-sensitive display includes: detecting contact with the touch-sensitive display while the device is in a user-interface lock state; moving an image corresponding to a user-interface unlock state of the device in accordance with the contact; transitioning the device to the user-interface unlock state if the detected contact corresponds to a predefined gesture; and maintaining the device in the user-interface lock state if the detected contact does not correspond to the predefined gesture.

In some embodiments, a method of controlling a device with a touch-sensitive display includes: displaying an image on the touch-sensitive display while the device is in a user-interface lock state; detecting contact with the touch-sensitive display; transitioning the device to a user-interface unlock state if the detected contact corresponds to moving the image to a predefined location on the touch-sensitive display; and maintaining the device in the user-interface lock state if the detected contact does not correspond to moving the image to the predefined location.

In some embodiments, a method of controlling a device with a touch-sensitive display includes: displaying an image on the touch-sensitive display while the device is in a user-interface lock state; detecting contact with the touch-sensitive display; and transitioning the device to a user-interface unlock state if the detected contact corresponds to moving the image on the touch-sensitive display according to a predefined path on the touch-sensitive display; and maintaining the device in the user-interface lock state if the detected contact does not correspond to moving the image according to the predefined path.

In some embodiments, a method of controlling a device with a touch-sensitive display includes: displaying first and second images on the touch-sensitive display while the device is in a user-interface lock state; detecting contact with the touch-sensitive display; transitioning the device to a first active state corresponding to the first image if the detected contact corresponds to a predefined gesture with respect to the first image; and transitioning the device to a second active state distinct from the first active state if the detected contact corresponds to a predefined gesture with respect to the second image.

The aforementioned methods may be performed by a portable electronic device having a touch-sensitive display with a graphical user interface (GUI), one or more processors, memory and one or more modules, programs or sets of instructions stored in the memory for performing these methods. In some embodiments, the portable electronic device provides a plurality of functions, including wireless communication.

Instructions for performing the aforementioned methods may be included in a computer program product configured for execution by one or more processors. In some embodiments, the executable computer program product includes a computer readable storage medium (e.g., one or more magnetic disk storage devices, flash memory devices, or other non-volatile solid state memory devices) and an executable computer program mechanism embedded therein.

BRIEF DESCRIPTION OF THE DRAWINGS

For a better understanding of the aforementioned embodiments of the invention as well as additional embodiments thereof, reference should be made to the Description of Embodiments below, in conjunction with the following drawings in which like reference numerals refer to corresponding parts throughout the figures.

FIG. **1** is a block diagram illustrating a portable electronic device, according to some embodiments of the invention.

FIG. **2** is a flow diagram illustrating a process for transitioning a device to a user-interface unlock state, according to some embodiments of the invention.

FIG. **3** is a flow diagram illustrating a process for transitioning a device to a user-interface unlock state, according to some embodiments of the invention.

FIGS. **4**A-**4**B illustrate the GUI display of a device in a user-interface lock state, according to some embodiments of the invention.

FIGS. **5**A-**5**D illustrate the GUI display of a device at various points of the performance of an unlock action gesture, according to some embodiments of the invention.

FIG. **6** is a flow diagram illustrating a process for indicating progress towards satisfaction of a user input condition according to some embodiments of the invention.

FIGS. **7**A-**7**D illustrate the GUI display of a device that is transitioning the optical intensity of user-interface objects, according to some embodiments of the invention.

FIGS. **8**A-**8**C are graphs illustrating optical intensity as a function of the completion of the user input condition, according to some embodiments of the invention.

FIG. **9** is a flow diagram illustrating a process for transitioning a device to a user interface active state, according to some embodiments of the invention.

FIG. **10** illustrates the GUI of a device in a user-interface lock state that displays a plurality of unlock images, according to some embodiments of the invention.

FIGS. **11**A-**11**F illustrate the GUI display of a device at various points in the performance of an unlock action gesture, according to some embodiments of the invention.

DESCRIPTION OF EMBODIMENTS

Reference will now be made in detail to embodiments, examples of which are illustrated in the accompanying drawings. In the following detailed description, numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it will be apparent to one of ordinary skill in the art that the present invention may be practiced without these specific details. In other instances, well-known methods, procedures, components, and circuits have not been described in detail so as not to unnecessarily obscure aspects of the embodiments.

FIG. **1** illustrates a portable electronic device, according to some embodiments of the invention. The device **100** includes a memory **102**, a memory controller **104**, one or more processing units (CPU's) **106**, a peripherals interface **108**, RF circuitry **112**, audio circuitry **114**, a speaker **116**, a microphone **118**, an input/output (I/O) subsystem **120**, a touch screen **126**, other input or control devices **128**, and an external port **148**. These components communicate over the one or more communication buses or signal lines **110**. The device **100** can be any portable electronic device, including but not limited to a handheld computer, a tablet computer, a mobile phone, a media player, a personal digital assistant (PDA), or the like, including a combination of two or more of these items. It should be appreciated that the device **100** is only one example of a portable electronic device **100**, and that the device **100** may have more or fewer components than shown, or a different configuration of components. The various components shown in FIG. **1** may be implemented in hardware, software or a combination of both hardware and software, including one or more signal processing and/or application specific integrated circuits.

The memory **102** may include high speed random access memory and may also include non-volatile memory, such as one or more magnetic disk storage devices, flash memory devices, or other non-volatile solid state memory devices. In some embodiments, the memory **102** may further include storage remotely located from the one or more processors **106**, for instance network attached storage accessed via the RF circuitry **112** or external port **148** and a communications network (not shown) such as the Internet, intranet(s), Local Area Networks (LANs), Wide Local Area Networks (WLANs), Storage Area Networks (SANs) and the like, or any suitable combination thereof. Access to the memory **102** by other components of the device **100**, such as the CPU **106** and the peripherals interface **108**, may be controlled by the memory controller **104**.

The peripherals interface **108** couples the input and output peripherals of the device to the CPU **106** and the memory **102**. The one or more processors **106** run various software programs and/or sets of instructions stored in the memory **102** to perform various functions for the device **100** and to process data.

In some embodiments, the peripherals interface **108**, the CPU **106**, and the memory controller **104** may be implemented on a single chip, such as a chip **111**. In some other embodiments, they may be implemented on separate chips.

The RF (radio frequency) circuitry **112** receives and sends electromagnetic waves. The RF circuitry **112** converts electrical signals to/from electromagnetic waves and communicates with communications networks and other communications devices via the electromagnetic waves. The RF circuitry **112** may include well-known circuitry for performing these functions, including but not limited to an antenna system, an RF transceiver, one or more amplifiers, a tuner, one or more oscillators, a digital signal processor, a CODEC chipset, a subscriber identity module (SIM) card, memory, and so forth. The RF circuitry **112** may communicate with the networks, such as the Internet, also referred to as the World Wide Web (WWW), an Intranet and/or a wireless network, such as a cellular telephone network, a wireless local area network (LAN) and/or a metropolitan area network (MAN), and other devices by wireless communication. The wireless communication may use any of a plurality of communications standards, protocols and technologies, including but not limited to Global System for Mobile Communications (GSM), Enhanced Data GSM Environment (EDGE), wideband code division multiple access (W-CDMA), code division multiple access (CDMA), time division multiple access (TDMA), Bluetooth, Wireless Fidelity (Wi-Fi) (e.g., IEEE 802.11a, IEEE 802.11b, IEEE 802.11g and/or IEEE 802.11n), voice over Internet Protocol (VoIP), Wi-MAX, a protocol for email, instant messaging, and/or Short Message Service (SMS)), or any other suitable communication protocol, including communication protocols not yet developed as of the filing date of this document.

The audio circuitry **114**, the speaker **116**, and the microphone **118** provide an audio interface between a user and the device **100**. The audio circuitry **114** receives audio data from the peripherals interface **108**, converts the audio data to an electrical signal, and transmits the electrical signal to the speaker **116**. The speaker converts the electrical signal to human-audible sound waves. The audio circuitry **114** also receives electrical signals converted by the microphone **116** from sound waves. The audio circuitry **114** converts the electrical signal to audio data and transmits the audio data to the peripherals interface **108** for processing. Audio data may be may be retrieved from and/or transmitted to the memory **102** and/or the RF circuitry **112** by the peripherals interface **108**. In some embodiments, the audio circuitry **114** also includes a headset jack (not shown). The headset jack provides an interface between the audio circuitry **114** and removable audio input/output peripherals, such as output-only headphones or a headset with both output (headphone for one or both ears) and input (microphone).

The I/O subsystem **120** provides the interface between input/output peripherals on the device **100**, such as the touch screen **126** and other input/control devices **128**, and the peripherals interface **108**. The I/O subsystem **120** includes a touch-screen controller **122** and one or more input controllers **124** for other input or control devices. The one or more input controllers **124** receive/send electrical signals from/to other input or control devices **128**. The other input/control devices **128** may include physical buttons (e.g., push buttons, rocker buttons, etc.), dials, slider switches, sticks, and so forth.

The touch screen **126** provides both an output interface and an input interface between the device and a user. The touch-screen controller **122** receives/sends electrical signals from/ to the touch screen **126**. The touch screen **126** displays visual output to the user. The visual output may include text, graphics, video, and any combination thereof. Some or all of the visual output may correspond to user-interface objects, further details of which are described below.

The touch screen **126** also accepts input from the user based on haptic and/or tactile contact. The touch screen **126** forms a touch-sensitive surface that accepts user input. The touch screen **126** and the touch screen controller **122** (along with any associated modules and/or sets of instructions in the memory **102**) detects contact (and any movement or break of the contact) on the touch screen **126** and converts the detected contact into interaction with user-interface objects, such as one or more soft keys, that are displayed on the touch screen. In an exemplary embodiment, a point of contact between the touch screen **126** and the user corresponds to one or more digits of the user. The touch screen **126** may use LCD (liquid crystal display) technology, or LPD (light emitting polymer display) technology, although other display technologies may be used in other embodiments. The touch screen **126** and touch screen controller **122** may detect contact and any movement or break thereof using any of a plurality of touch sensitivity technologies, including but not limited to capacitive, resistive, infrared, and surface acoustic wave technologies, as well as other proximity sensor arrays or other elements for determining one or more points of contact with the touch screen **126**. The touch-sensitive display may be analogous to the multi-touch sensitive tablets described in the following U.S. Pat. Nos. 6,323,846 (Westerman et al.), 6,570,557 (Westerman et al.), and/or 6,677,932 (Westerman), and/or U.S. Patent Publication 2002/0015024A1, each of which is hereby incorporated by reference. However, the touch screen **126** displays visual output from the portable device, whereas touch sensitive tablets do not provide visual output. The touch screen **126** may have a resolution in excess of 100 dpi. In an exemplary embodiment, the touch screen **126** may have a resolution of approximately 168 dpi. The user may make contact with the touch screen **126** using any suitable object or appendage, such as a stylus, finger, and so forth.

In some embodiments, in addition to the touch screen, the device **100** may include a touchpad (not shown) for activating or deactivating particular functions. In some embodiments, the touchpad is a touch-sensitive area of the device that, unlike the touch screen, does not display visual output. The touchpad may be a touch-sensitive surface that is separate from the touch screen **126** or an extension of the touch-sensitive surface formed by the touch screen **126**.

The device **100** also includes a power system **130** for powering the various components. The power system **130** may include a power management system, one or more power sources (e.g., battery, alternating current (AC)), a recharging system, a power failure detection circuit, a power converter or inverter, a power status indicator (e.g., a light-emitting diode (LED)) and any other components associated with the generation, management and distribution of power in portable devices.

In some embodiments, the software components include an operating system **132**, a communication module (or set of instructions) **134**, a contact/motion module (or set of instructions) **138**, a graphics module (or set of instructions) **140**, a user interface state module (or set of instructions) **144**, and one or more applications (or set of instructions) **146**.

The operating system **132** (e.g., Darwin, RTXC, LINUX, UNIX, OS X, WINDOWS, or an embedded operating system such as VxWorks) includes various software components and/or drivers for controlling and managing general system tasks (e.g., memory management, storage device control, power management, etc.) and facilitates communication between various hardware and software components.

The communication module **134** facilitates communication with other devices over one or more external ports **148** and also includes various software components for handling data received by the RF circuitry **112** and/or the external port **148**. The external port **148** (e.g., Universal Serial Bus (USB), FIREWIRE, etc.) is adapted for coupling directly to other devices or indirectly over a network (e.g., the Internet, wireless LAN, etc.).

The contact/motion module **138** detects contact with the touch screen **126**, in conjunction with the touch-screen controller **122**. The contact/motion module **138** includes various software components for performing various operations related to detection of contact with the touch screen **122**, such as determining if contact has occurred, determining if there is movement of the contact and tracking the movement across the touch screen, and determining if the contact has been broken (i.e., if the contact has ceased). Determining movement of the point of contact may include determining speed (magnitude), velocity (magnitude and direction), and/or an acceleration (including magnitude and/or direction) of the point of contact. In some embodiments, the contact/motion module **126** and the touch screen controller **122** also detects contact on the touchpad.

The graphics module **140** includes various known software components for rendering and displaying graphics on the touch screen **126**. Note that the term "graphics" includes any object that can be displayed to a user, including without limitation text, web pages, icons (such as user-interface objects including soft keys), digital images, videos, animations and the like.

In some embodiments, the graphics module **140** includes an optical intensity module **142**. The optical intensity module **142** controls the optical intensity of graphical objects, such as user-interface objects, displayed on the touch screen **126**. Controlling the optical intensity may include increasing or decreasing the optical intensity of a graphical object. In some embodiments, the increase or decrease may follow predefined functions.

The user interface state module **144** controls the user interface state of the device **100**. The user interface state module **144** may include a lock module **150** and an unlock module **152**. The lock module detects satisfaction of any of one or more conditions to transition the device **100** to a user-interface lock state and to transition the device **100** to the lock state. The unlock module detects satisfaction of any of one or more conditions to transition the device to a user-interface unlock state and to transition the device **100** to the unlock state. Further details regarding the user interface states are described below.

The one or more applications **130** can include any applications installed on the device **100**, including without limitation, a browser, address book, contact list, email, instant messaging, word processing, keyboard emulation, widgets, JAVA-enabled applications, encryption, digital rights management, voice recognition, voice replication, location determination capability (such as that provided by the global positioning system (GPS)), a music player (which plays back recorded music stored in one or more files, such as MP3 or AAC files), etc.

In some embodiments, the device **100** may include the functionality of an MP3 player, such as an iPod (trademark of Apple Computer, Inc.). The device **100** may, therefore, include a 36-pin connector that is compatible with the iPod. In some embodiments, the device **100** may include one or more optional optical sensors (not shown), such as CMOS or CCD image sensors, for use in imaging applications.

In some embodiments, the device **100** is a device where operation of a predefined set of functions on the device is performed exclusively through the touch screen **126** and, if included on the device **100**, the touchpad. By using the touch screen and touchpad as the primary input/control device for operation of the device **100**, the number of physical input/control devices (such as push buttons, dials, and the like) on the device **100** may be reduced. In one embodiment, the device **100** includes the touch screen **126**, the touchpad, a push button for powering the device on/off and locking the device, a volume adjustment rocker button and a slider switch for toggling ringer profiles. The push button may be used to turn the power on/off on the device by depressing the button and holding the button in the depressed state for a predefined time interval, or may be used to lock the device by depressing the button and releasing the button before the predefined time interval has elapsed. In an alternative embodiment, the device **100** also may accept verbal input for activation or deactivation of some functions through the microphone **118**.

The predefined set of functions that are performed exclusively through the touch screen and the touchpad include navigation between user interfaces. In some embodiments, the touchpad, when touched by the user, navigates the device **100** to a main, home, or root menu from any user interface that may be displayed on the device **100**. In such embodiments, the touchpad may be referred to as a "menu button." In some other embodiments, the menu button may be a physical push button or other physical input/control device instead of a touchpad.

User Interface States

The device **100** may have a plurality of user interface states. A user interface state is a state in which the device **100** responds in a predefined manner to user input. In some embodiments, the plurality of user interface states includes a user-interface lock state and a user-interface unlock state. In some embodiments, the plurality of user interface states includes states for a plurality of applications.

In the user-interface lock state (hereinafter the "lock state"), the device **100** is powered on and operational but ignores most, if not all, user input. That is, the device **100** takes no action in response to user input and/or the device **100** is prevented from performing a predefined set of operations in response to the user input. The predefined set of operations may include navigation between user interfaces and activation or deactivation of a predefined set of functions. The lock state may be used to prevent unintentional or unauthorized use of the device **100** or activation or deactivation of functions on the device **100**. When the device **100** is in the lock state, the device **100** may be said to be locked. In some embodiments, the device **100** in the lock state may respond to a limited set of user inputs, including input that corresponds to an attempt to transition the device **100** to the user-interface unlock state or input that corresponds to powering the device **100** off. In other words, the locked device **100** responds to user input corresponding to attempts to transition the device **100** to the user-interface unlock state or powering the device **100** off, but does not respond to user input corresponding to attempts to navigate between user interfaces. It should be appreciated that even if the device **100** ignores a user input, the device **100** may still provide sensory feedback (such as visual, audio, or vibration feedback) to the user upon detection of the input to indicate that the input will be ignored.

In embodiments where the device **100** includes the touch screen **126**, while the device **100** is locked, a predefined set of operations, such as navigation between user interfaces, is prevented from being performed in response to contact on the touch screen **126** when the device **100** is locked. In other words, when the contact is being ignored by the locked device **100**, the touch screen may be said to be locked. A locked device **100**, however, may still respond to a limited class of contact on the touch screen **126**. The limited class includes contact that is determined by the device **100** to correspond to an attempt to transition the device **100** to the user-interface unlock state.

In the user-interface unlock state (hereinafter the "unlock state"), the device **100** is in its normal operating state, detecting and responding to user input corresponding to interaction with the user interface. A device **100** that is in the unlock state may be described as an unlocked device **100**. An unlocked device **100** detects and responds to user input for navigating between user interfaces, entry of data and activation or deactivation of functions. In embodiments where the device **100** includes the touch screen **126**, the unlocked device **100** detects and responds to contact corresponding to navigation between user interfaces, entry of data and activation or deactivation of functions through the touch screen **126**.

Unlocking a Device via Gestures

FIG. **2** is a flow diagram illustrating a process **200** for transitioning a device to a user-interface unlock state, according to some embodiments of the invention. As used herein, transitioning from one state to another refers to the process of going from one state to another. The process may be, as perceived by the user, instantaneous, near-instantaneous, gradual or at any suitable rate. The progression of the process may be controlled automatically by the device, such as the device **100** (FIG. **1**), independent of the user, once the process is activated; or it may be controlled by the user. While the process flow **200** described below includes a number of operations that appear to occur in a specific order, it should be apparent that these processes may include more or fewer operations, which may be executed serially or in parallel (e.g., using parallel processors or a multi-threading environment).

A device is set to the lock state (**202**). The device may be set (that is, transition completely to the lock state from any other state) to the locked state upon satisfaction of any of one or more lock conditions. The lock conditions may include events such as the elapsing of a predefined time of inactivity, entry into an active call, or powering on the device. The lock conditions may also include user intervention, namely the user locking the device by a predefined user input. In some embodiments, the user may be allowed to specify the events that serve as lock conditions. For example, the user may configure the device to transition to the lock state upon the elapsing of a predefined time of inactivity but not upon powering on the device.

In some embodiments, the locked device displays on the touch screen one or more visual cues of an unlock action that the user may perform to unlock the device (**204**). The visual cue(s) provide hints or reminders of the unlock action to the user. The visual cues may be textual, graphical or any combination thereof. In some embodiments, the visual cues are displayed upon particular events occurring while the device is locked. The particular events that trigger display of the visual cues may include an incoming call, incoming message, or some other event that may require the user's attention. In some embodiments, the visual cues may also be displayed upon particular user inputs, such as the user interacting with the menu button, the user making contact with the locked touch screen and/or the user interacting with any other input/control device. The locked device, when not displaying the visual cues, may power down the touch screen (which helps to conserve power) or display other objects on the touch screen, such as a screen saver or information that may be of interest to the user (e.g., battery charge remaining, date and time, network strength, etc.).

The unlock action includes contact with the touch screen. In some embodiments, the unlock action is a predefined gesture performed on the touch screen. As used herein, a gesture is a motion of the object/appendage making contact with the touch screen. For example, the predefined gesture may include a contact of the touch screen on the left edge (to initialize the gesture), a horizontal movement of the point of contact to the opposite edge while maintaining continuous contact with the touch screen, and a breaking of the contact at the opposite edge (to complete the gesture).

While the touch screen is locked, the user may initiate contact with the touch screen, i.e., touch the touch screen (**206**). For convenience of explanation, contact on the touch screen in the process **200** and in other embodiments described below will be described as performed by the user using at least one hand using one or more fingers. However, it should be appreciated that the contact may be made using any suitable object or appendage, such as a stylus, finger, etc. The contact may include one or more taps on the touch screen, maintaining continuous contact with the touch screen, movement of the point of contact while maintaining continuous contact, a breaking of the contact, or any combination thereof.

The device detects the contact on the touch screen (**208**). If the contact does not correspond to an attempt to perform the unlock action, or if the contact corresponds to a failed or aborted attempt by the user to perform the unlock action (**210**—no), then the device remains locked (**212**). For example, if the unlock action is a horizontal movement of the point of contact across the touch screen while maintaining continuous contact with the touch screen, and the detected contact is a series of random taps on the touch screen, then the device will remain locked because the contact does not correspond to the unlock action.

If the contact corresponds to a successful performance of the unlock action, i.e., the user performed the unlock action successfully (**210**—yes), the device transitions to the unlock state (**214**). For example, if the unlock action is a horizontal movement of the point of contact across the touch screen while maintaining continuous contact with the touch screen, and the detected contact is the horizontal movement with the continuous contact, then the device transitions to the unlock state.

In some embodiments, the device begins the process of transitioning to the unlock state upon detection of any contact on the touch screen and aborts the transition as soon as the device determines that the contact does not correspond to an unlock action or is a failed/aborted unlock action. For example, if the unlock action is a predefined gesture, the device may begin the process of transitioning to the unlock state as soon as it detects the initial contact of the gesture and continues the progression of the transition as the gesture is performed. If the user aborts the gesture before it is completed, the device aborts the transition and remains in the lock state. If the gesture is completed, the device completes the transition to the unlock state and becomes unlocked. As another example, if the unlock action is a horizontal movement of the point of contact across the touch screen while maintaining continuous contact with the touch screen, and the user taps the touch screen once, the device begins the process of the state transition as soon as it detects the tap but also aborts the process soon after because it realizes that the tap is just a tap and does not correspond to the unlock action.

While the device is unlocked, the device may display on the touch screen user-interface objects corresponding to one or more functions of the device and/or information that may be of interest to the user. The user-interface objects are objects that make up the user interface of the device and may include, without limitation, text, images, icons, soft keys (or "virtual buttons"), pull-down menus, radio buttons, check boxes, selectable lists, and so forth. The displayed user-interface objects may include non-interactive objects that convey information or contribute to the look and feel of the user interface, interactive objects with which the user may interact, or any combination thereof. The user may interact with the user-interface objects by making contact with the touch screen at one or more touch screen locations corresponding to the interactive objects with which she wishes to interact. The device detects the contact and responds to the detected contact by performing the operation(s) corresponding to the interaction with the interactive object(s).

While the device is locked, the user may still make contact on the touch screen. However, the locked device is prevented from performing a predefined set of actions in response to any detected contact until the device is unlocked. The prevented predefined set of action may include navigating between user interfaces and entry of data by the user.

While the device is locked, the device may display one or more visual cues of the unlock action, as described above. In some embodiments, the device may also display, along with the visual cues, an unlock image. The unlock image is a graphical, interactive user-interface object with which the user interacts in order to unlock the device. In other words, the unlock action is performed with respect to the unlock image. In some embodiments, performing the unlock action with respect to the image includes dragging the unlock image in a predefined manner, which moves the unlock image across the touch screen. In some embodiments, if the unlock action is not completed, the GUI display can show reverse progress towards the locked state by gradually returning the unlock image to its position in the locked state

In some embodiments, in addition to visual feedback, the electronic device supplies non-visual feedback to indicate progress towards completion of the unlock action. In some embodiments, in addition to visual feedback, the electronic device supplies non-visual feedback to indicate completion of the unlock action. The additional feedback may include audible feedback (e.g., sound(s)) or physical feedback (e.g., vibration(s)).

FIG. **3** is a flow diagram illustrating a process **300** for transitioning a device to a user-interface unlock state using an unlock image, according to some embodiments of the invention. The process **300** is similar to the process **200** (FIG. **2**) with the addition of an unlock image that is displayed with the visual cues. The unlock action in the process **300** is performed with respect to the unlock image, i.e., the unlock action includes interaction with the unlock image. While the process flow **300** described below includes a number of operations that appear to occur in a specific order, it should be apparent that these processes can include more or fewer operations, which can be executed serially or in parallel (e.g., using parallel processors or a multi-threading environment).

The device is locked upon satisfaction of a lock condition (**302**), similar to the operation **202** (FIG. **2**). An unlock image and visual cues of the unlock action using the unlock image are displayed (**304**). The operation **304** is the same as the operation **204** (FIG. **2**), except that in the operation **304** an unlock image is displayed in addition to the visual cues.

As described above, the unlock action includes interaction with the unlock image. In some embodiments, the unlock action includes the user performing a predefined gesture with respect to the unlock image. In some embodiments, the gesture includes dragging the unlock image to a location on the touch screen that meets one or more predefined unlock criteria. In other words, the user makes contact with the touch screen at a location corresponding to the unlock image and then performs the predefined gesture while maintaining continuous contact with the touch screen, dragging the image to the location that meets the predefined unlock criteria. In some embodiments, the unlock action is completed by breaking the contact with the touch screen (thus releasing the unlock image) upon completion of the predefined gesture.

A location meeting one or more predefined unlock criteria is simply a location on the touch screen that is predefined as a location to which the unlock image is to be dragged in order to unlock the device. The location(s) may be defined narrowly or broadly and may be one or more particular locations on the touch screen, one or more regions on the touch screen, or any combination thereof. For example, the location may be defined as a particular marked location, areas at each of the four corners of the touch screen, or a quadrant of the touch screen, etc.

In some embodiments, the interaction includes dragging the unlock image to a predefined location on the touch screen. For example, the unlock action may include dragging the unlock image from one corner of the touch screen to another corner of the touch screen. As another example, the unlock action may include dragging the unlock image from one edge of the touch screen to the opposite edge. The emphasis here is on the final destination of the unlock image (and of the finger) . Thus, the user can drag the unlock image from its initial location along any desired path. As long as the unlock image reaches the predefined location and is released at that location, the device is unlocked. It should be appreciated that the predefined location may be, as described above, defined narrowly or broadly and may be one or more particular locations on the touch screen, one or more regions on the touch screen, or any combination thereof.

In some other embodiments, the unlock action includes dragging the unlock image along a predefined path. For example, the unlock action may include dragging the unlock image clockwise along the perimeter of the touch screen (the path being the perimeter of the touch screen), from one of the corners and back. As another example, the unlock action may include dragging the unlock image from one edge of the touch screen to the opposite edge in a linear path. The emphasis here is on the path along which the unlock image (and the finger) moves. Because of the emphasis on the path, the final location to which the unlock image is to be moved may be defined broadly. For example, the unlock action may be to drag the unlock image from its initial location, along the predefined path, to any spot within a predefined region on the touch screen. The predefined path may include one or more straight lines or lines with twists and turns.

The user makes contact with the touch screen (**306**), similar to the operation **206** (FIG. **2**). The device detects the contact with the touch screen (**308**), similar to the operation **208** (FIG. **2**). If the contact does not correspond to successful performance of the unlock action with respect to the image (**310**—no), the device remains locked. If the contact does correspond to successful performance of the unlock action with respect to the image (**310**—yes), the device is unlocked (**314**).

FIGS. **4**A-**4**B illustrate the GUI display of a device in a user-interface lock state, according to some embodiments of the invention. In FIG. **4**A, device **400** includes a touch screen **408** and a menu button **410**. The device **400** is locked and the touch screen **408** is displaying an unlock image **402** and visual cues. The visual cues shown include a channel **404** indicating the path of the gesture/movement along which the unlock image **402** is to be dragged, similar to a groove along which a slider switch moves; and one or more arrows **406** indicating the direction of the gesture/movement. The end of the channel **404** (in FIGS. **4**A-**4**B and **5**A-**5**D, the "end" of the channel is the right end) also serves as a predefined location to which the unlock image **402** is to be dragged. The unlock image **402** may also include an arrow to further remind the user the direction of the gesture/movement. As described above, the visual cues and the unlock image may be displayed by the device **400** upon an event that may require the user's attention (e.g., incoming call or message) or upon user intervention (e.g., the user pressing the menu button **410** while the device is locked).

In some embodiments, the arrows **406** and the arrow on the unlock image **402** may be animated. For example, the arrow on the unlock image **402** may appear and disappear in a pulse-like manner and the arrows **406** may emanate from one end of the channel **406** in sync with the pulsing of the arrow on the unlock image **402**. As shown in FIG. **4**B, the arrow **406** may move along the channel **404** and disappear when it moves to the end of the channel **404**.

The visual cues illustrated in FIGS. **4**A and **4**B remind the user that the unlock action is a predefined gesture that includes a horizontal movement of the finger (and thus moving the point of contact) along the channel **404**, from the beginning of the channel **404**, where the unlock image is initially located, to the end of the channel **404**. It should be appreciated, however, that the visual cues shown in FIGS. **4**A-**4**B are merely exemplary and that more or fewer visual cues, or alternative visual cues may be used. The content of the visual cues may be based on the particulars of the unlock action.

FIGS. **5**A-**5**D illustrate the GUI display of a device at various points of the performance of an unlock action gesture, according to some embodiments of the invention. In FIG. **5**A, the user, represented by the hand and finger **502** (not drawn to scale), begins the unlock action by touching the touch screen **408** of device **400** with her finger **502**. In some embodiments, the touch screen **408** is initially in sleep mode and/or dark, and the screen **408** displays the unlock image **402** when touched. The user touches the touch screen **408** at the location corresponding to the unlock image **402**, which is located initially at the left end of the channel **404**. The contact, either overlapping with the unlock image **402** or in proximity to the unlock image **402**, is detected by the device **400** and is determined to be an attempt to unlock the touch screen, based on the fact that the user **502** is interacting with the unlock image **402**.

In FIG. **5**B, the user is in the process of performing the gesture by moving her finger, which is in continuous contact with the touch screen **408**, in the direction of movement **504**. The unlock image **402** is dragged along the channel **404** as a result of the gesture. The channel **404** reminds the user that the unlock gesture is a horizontal motion. In some embodiments, the channel **404** indicates the predefined location (in FIGS. **5**A-**5**D, the right end of the channel) to which the user drags the unlock image **402** to complete the unlock action and/or the predefined path along which the user drags the unlock image **402** to complete the unlock action.

In FIG. **5**C, the user has dragged the unlock image **402** to the right end of the channel **404**. Once the user releases the unlock image **402** at the right end of the channel **404**, the unlock action is complete. Upon completion of the unlock gesture, the device unlocks and displays on the touch screen **408** user-interface objects associated with normal operation of the device **400**. FIG. **5**D illustrates an example of user-interface objects that may be displayed when the device **400** is unlocked. In FIG. **5**D, the device **400** displays a menu **506**. The menu **506** includes interactive user-interface objects corresponding to various applications or operations. A user may interact with the user-interface objects to activate an application or perform an operation. It should be appreciated, however, that the device **400**, upon being unlocked, may display additional or alternative user-interface objects.

In some embodiments, the unlock image **402** may also be used to indicate failure of performance of the unlock action. For example, if the user breaks the contact with the touch screen before the unlock image reaches the right end of the channel **404**, the unlock action has failed. The device **400** may display the unlock image **402** returning to its initial position on the left end of the channel **404**, allowing the user to attempt the unlock action again, if she so chooses. In some embodiments, the device goes back to sleep if no gesture is applied in a predetermined period of time.

In some embodiments, the user may unlock the device **400** by contacting the touch screen **408** and moving the point of contact horizontally along a fraction of the channel **404**, i.e., the user need not move all the way to the right end of the channel. In some embodiments, the user may unlock the device **400** by making contact anywhere on the touch screen **408** and moving the point of contact horizontally as if he or she were following the channel **404**.

In some embodiments, the lock/unlock feature may apply to specific applications that are executing on the device **400** as opposed to the device **400** as a whole. In some embodiments, an unlock gesture transitions from one application to another, for example, from a telephone application to a music player or vice versa. The lock/unlock feature may include a hold or pause feature. In some embodiments, as the user transitions from a first application and to a second application, a user interface for the second application may fade in (i.e., increase in intensity) and a user interface for the first application may fade out (i.e., decrease in intensity). The fade in and fade out may occur smoothly over a pre-determined time interval, such as 0.2 s, 1 s or 2 s. The pre-determined time interval may be in accordance with the unlock gesture, such as the time it takes the user to perform the gesture.

Indication of Progress Towards Satisfaction of a User Input Condition

FIG. **6** is a flow diagram illustrating a process **600** for indicating progress towards satisfaction of a user input condition according to some embodiments of the invention. While the process flow **600** described below includes a number of operations that appear to occur in a specific order, it should be apparent that these processes can include more or fewer operations, which can be executed serially or in parallel (e.g., using parallel processors or a multi-threading environment).

While an electronic device is in a first user-interface state, progress is detected (**602**) towards satisfaction of a user input condition needed to transition to a second user-interface state. In some embodiments, the first user-interface state is for a first application and the second user-interface state is for a second application. In some embodiments, the first user-interface state is a lock state and the second user-interface state is an unlock state.

While the device is in the first user-interface state, progress is indicated (**604**) towards satisfaction of the condition by transitioning an optical intensity of one or more user interface objects associated with the second user-interface state. The change in optical intensity of the user-interface objects provides a user with sensory feedback of the progress in transitioning between user interface states.

In some embodiments, in addition to visual feedback, the device supplies non-visual feedback to indicate progress towards satisfaction of the user input condition. The additional feedback may include audible feedback (e.g., sound(s)) or physical feedback (e.g., vibration(s)).

The device transitions (**606**) to the second user-interface state if the condition is satisfied. In some embodiments, in addition to visual feedback, the device supplies non-visual feedback to indicate satisfaction of the user input condition. The additional feedback may include audible feedback (e.g., sound(s)) or physical feedback (e.g., vibration(s)).

The optical intensity of a user-interface object, as used herein, is the object's degree of visual materialization. The optical intensity may be measured along a scale between a predefined minimum and a predefined maximum. In some embodiments, the optical intensity may be measured along a logarithmic scale. In some embodiments, the optical intensity may be perceived by users as a transparency effect (or lack thereof) applied to the user-interface object. In some embodiments, the minimum optical intensity means that the object is not displayed at all (i.e., the object is not perceptible to the user), and the maximum optical intensity means that the object is displayed without any transparency effect (i.e., the object has completely materialized visually and is perceptible to the user). In some other embodiments, the optical intensity may be the visual differentiation between the user-interface object and the background, based on color, hue, color saturation, brightness, contrast, transparency, and any combination thereof.

In some embodiments, the optical intensity of the user-interface objects to be displayed in the second user-interface state is increased smoothly. Smoothly may include a transition time that is greater than a pre-defined threshold, for example, 0.2 s, 1 s or 2 s. The rate of the transition of the optical intensity may be any predefined rate.

In some embodiments, the indication of progress towards completion of the user input condition is a function of the user's satisfaction of the condition. For example, for a transition to an unlock state, the indication of progress towards completion is a function of the user's performance of an unlock action. For a linear function, the indication of progress is 10% complete when the unlock action is 10% complete; the indication of progress is 50% complete when the unlock action is 50% complete, and so forth, up to 100% completion of the unlock action, at which point the transition to the unlock state occurs. Correspondingly, for a linear function, the transition of the optical intensity from an initial value to a final value is 10% complete when the unlock action is 10% complete; the transition is 50% complete when the unlock action is 50% complete, and so forth, up to 100% completion of the unlock action, at which point the optical intensity is at its final value. In some embodiments, the user may perceive the optical intensity transition as a fading in of the user-interface objects as the unlock action is performed. It should be appreciated that the function need not be linear and alternative functions may be used, further details of which are described below, in relation to FIGS. **8**A-**8**C.

If the user input condition includes a predefined gesture then the indication of progress of the gesture may be defined in terms of how much of the gesture is completed and how much of the gesture is remaining. For example, if the gesture includes moving the finger from one edge of the screen to the opposite edge horizontally, then the indication of progress may be defined in terms of the distance between the two edges because the distance remaining objectively measures how much further the user has to move her finger to complete the gesture.

If the user input condition includes dragging an image to a predefined location, then the indication of progress may be defined in terms of the distance between the initial location of the image and the predefined location to which the image is to be dragged in order to complete the input condition.

If the user input condition includes dragging an image along a predefined path, then the indication of progress may be defined in terms of the length of the predefined path.

FIGS. **7**A-**7**D illustrate the GUI display of a device that is transitioning the optical intensity of user-interface objects concurrent with a transition from a first user interface state to a second user interface state, according to some embodiments of the invention. In FIG. **7**A, the device **700** is locked and has received an incoming call. The device **700** is displaying a prompt **706** to the user, informing the user of the incoming call, on the touch screen **714**. The device is also displaying the unlock image **702** and channel **704** so that the user can unlock the device **700** in order to accept or decline the incoming call. The user begins the unlock action by making contact on the touch screen with her finger **710** on the unlock image **702**.

In FIG. **7**B, the user is in the process of dragging the unlock image **702** along the channel **704** in the direction of movement **712**. As the user drags the unlock image, a set of virtual buttons **708** appears and increases in optical intensity. The virtual buttons **708** are shown with dotted outlines to indicate that they are not yet at their final optical intensity levels. The virtual buttons **708** are associated with the prompt **706**; the virtual buttons shown in FIG. **7**B-**7**D allow the user to decline or accept the incoming call. However, the user cannot interact with the virtual buttons **708** until the device is unlocked and the virtual buttons have reached their final optical intensity. In FIG. **7**C, the user drags the unlock image **702** further along the channel **704** in the direction of movement **712**. The virtual buttons **708** have increased further in optical intensity relative to their optical intensity in FIG. **7**B, as illustrated by their different style of dotted outlines. The increases in optical intensity indicate to the user progress towards completion of the unlock action.

In FIG. **7**D, the user completes the unlock action by dragging the unlock image to the right end of the channel **704** and releasing the unlock image **702**. The device **700** transitions to the unlock state. The unlock image **702** and the channel **704** disappear from the display and the virtual buttons **708** are at their final optical intensity levels, as illustrated by their solid outlines. At this point the user may interact with the virtual buttons **708** and accept or decline the incoming call.

As described above in relation to FIGS. **5**A-**5**D, if the unlock action fails because the user releases the unlock image prematurely, the unlock image may return to its initial location. In some embodiments, the optical intensity of the virtual buttons **708** or other user-interface objects that were increasing in optical intensity as the unlock action was performed may, concurrent with the return of the unlock image to its initial location, have their optical intensity decreased smoothly, back to their initial levels.

FIGS. **8**A-**8**C are graphs illustrating optical intensity as a function of the completion of the user input condition, according to some embodiments of the invention. In FIG. **8**A, the optical intensity is a linear function of the completion of the user input condition. At 0% completion, the optical intensity is at an initial value (in this case, the initial value is 0). As the completion percentage increases, the optical intensity increases linearly with the completion percentage, until it reaches the final value at 100% completion.

In FIG. **8**B, the optical intensity is a nonlinear function of the completion of the user input condition. At 0% completion, the optical intensity is at an initial value (in this case, the initial value is 0). As the completion percentage increases, the optical intensity increases gradually at first, but the increase becomes steeper as the completion percentage increases, until it reaches the final value at 100% completion.

In FIG. **8**C, the optical intensity is another nonlinear function of the completion of the user input condition. At 0% completion, the optical intensity is at an initial value (in this case, the initial value is 0). As the completion percentage increases, the optical intensity increases steeply at first, but the increase becomes more gradual as the completion percentage increases, until it reaches the final value at 100% completion. In some embodiments, the optical intensity may increase according to a logarithmic scale.

In some embodiments, the optical intensity may reach its final value prior to 100% completion of the user input condition (e.g., at 90% completion).

User Interface Active States Corresponding to Events or Applications

FIG. **9** is a flow diagram illustrating a process **900** for transitioning a device to a user interface active state corresponding to one of a plurality of unlock images, according to some embodiments of the invention. In some embodiments, the device may have one or more active applications running when the device becomes locked. Additionally, while locked, the device may continue to receive events, such as incoming calls, messages, voicemail notifications, and so forth. The device may display multiple unlock images on the touch screen, each unlock image corresponding to an active application or incoming event. Performing the unlock action using one of the multiple unlock images unlocks the device and displays the application and/or event corresponding to the unlock image. The user interface active state, as used herein, means that the device is unlocked and a corresponding application or event is displayed on the touch screen to the user. While the process flow **900** described below includes a number of operations that appear to occur in a specific order, it should be apparent that these processes can include more or fewer operations, which can be executed serially or in parallel (e.g., using parallel processors or a multi-threading environment).

The device is locked upon satisfaction of a predefined lock condition (**902**). The device may have active applications running when it is locked and the active applications may continue running while the device is locked. Additionally, while the device is locked, the device may receive events, such as incoming calls, messages, and voicemail notifications.

The device displays a plurality of unlock images, each displayed unlock image corresponding to an active application running or an event received while the device is locked (**904**). In some embodiments, the device also displays visual cues of the unlock action with respect to each unlock image. The device may display additional unlock images and visual cues as additional events are received. The user makes contact with the touch screen (**906**). The device detects the contact gesture (**908**). If the detected contact gesture does not correspond to successful performance of the unlock action with respect to any one of the displayed unlock images (e.g., because the contact is not an attempt to perform the unlock action or the unlock action failed/was aborted) (**910**—no), the device remains locked (**912**). If the detected contact gesture does correspond to successful performance of the unlock action with respect to one of the displayed unlock images (**910**—yes), the touch screen is unlocked and the running application or event corresponding to the one of the unlock images is displayed on the touch screen (**914**). In other words, the device transitions to a first active state corresponding to the first image if the detected contact corresponds to a predefined gesture with respect to the first image; the device transitions to a second active state distinct from the first active state and corresponding to the second image if the detected contact corresponds to a predefined gesture with respect to the second image; and so on.

The device becomes unlocked and makes the corresponding event or application visible to the user, active, or running in the foreground, as opposed to running in the background, upon performance of the unlock action with respect to the particular unlock image. The user-interface active state includes the running application or incoming event corresponding to the particular unlock image with which the user interacted being displayed prominently on the touch screen, in addition to the device being unlocked. Thus, unlocking using a first unlock image (if multiple unlock images are displayed) transitions the device to a first user-interface active state, in which the device is unlocked and the application/event corresponding to the first unlock image is displayed prominently. Unlocking using a second image transitions the device to a second user-interface active state, in which the device is unlocked and the application/event corresponding to the second unlock image is displayed prominently.

In some embodiments, the device may prioritize which unlock images to display. The device may display a subset of the corresponding unlock images on the touch screen at one time. The device may decide which subset to display based on one or more predefined criteria. For example, the device may display only unlock images corresponding to the most recent events and/or running applications. As another example, the device may display only unlock images corresponding to incoming events.

FIG. **10** illustrates the GUI of a device **1000** in a user-interface lock state that displays a plurality of unlock images, according to some embodiments of the invention. In FIG. **10**, the touch screen **1014** of the device **1000** is locked. A first unlock image **1002** is displayed with corresponding visual cues, such as the first channel **1004** and arrow **1006**. A second unlock image **1008** is displayed with corresponding visual cues, such as the second channel **1010** and arrow **1012**. The touch screen **1014** may display additional unlock images and visual cues. The first unlock image **1002** corresponds to a first running application or received event. The second unlock image **1008** corresponds to a second running application or received event. The first and second unlock images and visual cues are similar to the unlock image and visual cues described above, in relation to FIGS. **4**A and **4**B. The arrows **1006** and **1012** may be animated to move from one end of the channels **1004** and/or **1010** to the other end, in order to indicate the proper direction of the predefined gesture or movement of the unlock image.

FIGS. **11**A-**11**F illustrate the GUI display of a device at various points in the performance of an unlock action gesture corresponding to one of a plurality of unlock images, according to some embodiments of the invention. In FIG. **11**A, the user makes contact with the touch screen **1014** using her finger **1102** (not shown to scale), at the location corresponding to the second unlock image **1008**. The user performs the unlock action gesture by moving the point of contact, dragging the second unlock image **1008**. FIG. **11**B shows a snapshot of the device **1000** during the pendency of the unlock action. The second unlock image **1008** is moved along in the channel **1010** in the direction of movement **1104**.

FIG. **11**C shows the second unlock image **1008** moved to the end of the channel **1010**, where the unlock action with respect to the second unlock image **1008** will be completed once the user breaks the contact (and releases the second unlock image **1008**). In some embodiments, the unlock action is completed when the unlock image **1008** is moved to the end of the channel **1010**, with or without the user breaking contact, and the second unlock image **1008** disappears. As shown in FIG. **11**D, upon completion of the unlock action with respect to the second unlock image **1008**, the device displays on the touch screen the user-interface objects **1106** associated with the application or event corresponding to the second unlock image **1008**. In FIG. **11**D, the event corresponding to the second unlock image is an incoming text message event and a prompt for the user to read it.

The user, instead of performing the unlock action with respect to the second unlock image **1108**, may instead perform the unlock action gesture with respect to the first unlock image **1002**. In FIG. **11**E, the user does so and performs the unlock action with respect to the first unlock image **1002** by dragging the first unlock image, in the direction **1104**, to the right end of the channel **1004**. Upon completion of the unlock action, the device **1000** displays the user-interface objects **1108** associated with the application or event corresponding to the first unlock image **1002**. In FIG. **11**F, the application corresponding to the first unlock image is a music player application.

In some embodiments, the transition to a user interface active state, as described in FIGS. **9** and **11**A-**11**E, may also include a concurrent transition in the optical intensity of user-interface objects, similar to that described above in relation to FIGS. **6**, **7**A-**7**D, and **8**A-**8**C. Concurrent with the transition to a user interface active state, the user-interface objects associated with the application or event corresponding to the unlock image with which the user interacted to unlock the device increase in intensity. For example, the optical intensity of the user-interface objects **1106** associated with the text message prompt in FIG. **11**D may be increased smoothly, as a function of the progress towards completion of the unlock action with respect to the second unlock image **1008**. As another example, the optical intensity of the user-interface objects **1108** associated with music player application in FIG. **11**F may be increased smoothly, as a function of the progress towards completion of the unlock action with respect to the first unlock image **1002**.

The foregoing description, for purpose of explanation, has been described with reference to specific embodiments. However, the illustrative discussions above are not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated.

What is claimed is:

1. A method of controlling an electronic device with a touch-sensitive display, comprising:

detecting contact with the touch-sensitive display while the device is in a user-interface lock state;

moving an unlock image along a predefined displayed path on the touch-sensitive display in accordance with the contact, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

transitioning the device to a user-interface unlock state if the detected contact corresponds to a predefined gesture; and

maintaining the device in the user-interface lock state if the detected contact does not correspond to the predefined gesture.

2. The method of claim **1**, further comprising, while the device is in the user-interface lock state, preventing the device from performing a predefined set of actions in response to detecting any contact with the touch-sensitive display that does not correspond to the predefined gesture.

3. The method of claim **1**, wherein the predefined displayed path is a channel.

4. The method of claim **1**, wherein the detected contact is a movement of a point of contact across the touch-sensitive display while maintaining continuous contact with the touch-sensitive display.

5. The method of claim **4**, wherein the movement of the point of contact across the touch-sensitive display while maintaining continuous contact with the touch-sensitive display is a horizontal movement.

6. A method of controlling a device comprising a touch-sensitive display, comprising:

displaying an unlock image on the touch-sensitive display while the device is in a user-interface lock state, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

detecting contact with the touch-sensitive display;

transitioning the device to a user-interface unlock state if the detected contact corresponds to moving the unlock image along a predefined displayed path on the touch-sensitive display to a predefined location on the touch-sensitive display; and

maintaining the device in the user-interface lock state if the detected contact does not correspond to moving the unlock image along the predefined displayed path on the touch-sensitive display to the predefined location.

7. The method of claim **6**, further comprising, while the device is in the user-interface lock state, preventing the device from performing a predefined set of actions in response to detecting any contact with the touch-sensitive display that does not correspond to moving the unlock image along the predefined displayed path on the touch-sensitive display to the predefined location.

8. A method of controlling a device comprising a touch-sensitive display, comprising:

displaying an unlock image on the touch-sensitive display while the device is in a user-interface lock state, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

detecting contact with the touch-sensitive display; and

transitioning the device to a user-interface unlock state if the detected contact corresponds to moving the unlock image across the touch-sensitive display according to a predefined displayed path on the touch-sensitive display; and

maintaining the device in the user-interface lock state if the detected contact does not correspond to moving the unlock image across the touch-sensitive display according to the predefined displayed path.

9. The method of claim **8**, further comprising, while the device is in a user-interface lock state, preventing the device from performing a predefined set of actions in response to detecting any contact with the touch-sensitive display that does not correspond to moving the unlock image across the touch-sensitive display according to the predefined displayed path.

10. The method of claim **8**, wherein the detected contact corresponding to moving the unlock image across the touch-sensitive display according the predefined displayed path comprises contact corresponding to moving the unlock image across the touch-sensitive display to an endpoint of the predefined displayed path.

11. A method of controlling a device comprising a touch-sensitive display, comprising:

displaying a first unlock image and a second unlock image on the touch-sensitive display while the device is in a user-interface lock state;

detecting contact with the touch-sensitive display;

transitioning the device to a first active state corresponding to the first unlock image if the detected contact corresponds to a predefined gesture with respect to the first unlock image that moves the first unlock image along a first predefined displayed path on the touch sensitive display; and

transitioning the device to a second active state distinct from the first active state if the detected contact corresponds to a predefined gesture with respect to the second unlock image that moves the second unlock image along a second predefined displayed path on the touch-sensitive display.

12. A portable electronic device, comprising:

a touch-sensitive display;

memory;

one or more processors; and

one or more modules stored in the memory and configured for execution by the one or more processors, the one or more modules including instructions:

to set the device to a user-interface lock state;

to display an unlock image on the touch-sensitive display while the device is in the user-interface lock state, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

to detect contact with the touch-sensitive display;

to move the unlock image along a predefined displayed path on the touch-sensitive display in accordance with the contact;

to transition the device to a user-interface unlock state if the detected contact corresponds to a predefined gesture; and

to maintain the device in the user-interface lock state if the detected contact does not correspond to the predefined gesture.

13. A portable electronic device, comprising:

a touch-sensitive display;

memory;

one or more processors; and

one or more modules stored in the memory and configured for execution by the one or more processors, the one or more modules including instructions:

to set the device to a user-interface lock state;

to display an unlock image on the touch-sensitive display while the device is in the user-interface lock state, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

to detect contact with the touch-sensitive display;

to transition the device to a user-interface unlock state if the detected contact corresponds to moving the unlock image along a predefined displayed path on the touch-sensitive display to a predefined location on the touch-sensitive display; and

to maintain the device in the user-interface lock state if the detected contact does not correspond to moving the unlock image along the predefined displayed path on the touch-sensitive display to the predefined location.

14. A portable electronic device, comprising:

a touch-sensitive display;

memory;

one or more processors; and

one or more modules stored in the memory and configured for execution by the one or more processors, the one or more modules including instructions:

to set the device to a user-interface lock state;

to display an unlock image on the touch-sensitive display while the device is in the user-interface lock state;

to detect contact with the touch-sensitive display;

to transition the device to a user-interface unlock state if the detected contact corresponds to moving the unlock image across the touch-sensitive display according to a predefined displayed path on the touch-sensitive display; and

to maintain the device in the user-interface lock state if the detected contact does not correspond to moving the unlock image across the touch-sensitive display according to the predefined displayed path.

15. A portable electronic device, comprising:

a touch sensitive display;

memory;

one or more processors;

one or more modules stored in the memory and configured for execution by the one or more processors, the one or more processors including instructions:

to set the device to a user-interface lock state;

to display a first unlock image and a second unlock image on the touch-sensitive display;

to detect contact with the touch-sensitive display;

to transition the device to a first active state corresponding to the first unlock image if the detected contact corresponds to a predefined gesture with respect to the first unlock image that moves the first unlock image along a first predefined displayed path on the touch-sensitive display; and

to transition the device to a second active state distinct from the first active state if the detected contact corresponds to a predefined gesture with respect to the second unlock image that moves the second unlock image along a second predefined displayed path on the touch-sensitive display.

16. A portable electronic device, comprising:

a touch-sensitive display;

means for detecting contact with the touch-sensitive display while the device is in a user-interface lock state;

means for moving an unlock image along a predefined displayed path on the touch-sensitive display in accordance with the contact, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

means for transitioning the device to the user-interface unlock state if the detected contact corresponds to a predefined gesture; and

means for maintaining the device in the user-interface lock state if the detected contact does not correspond to the predefined gesture.

17. A portable electronic device, comprising:

a touch-sensitive display;

means for displaying an unlock image on the touch-sensitive display while the device is in a user-interface lock state, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

means for detecting contact with the touch-sensitive display;

means for transitioning the device to a user-interface unlock state if the detected contact corresponds to moving the unlock image along a predefined displayed path on the touch-sensitive display to a predefined location on the touch-sensitive display; and

means for maintaining the device in the user-interface lock state if the detected contact does not correspond to moving the unlock image along the predefined displayed path on the touch-sensitive display to the predefined location.

18. A portable electronic device, comprising:

a touch-sensitive display;

means for displaying an unlock image on the touch-sensitive display while the device is in a user-interface lock state, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

means for detecting contact with the touch-sensitive display; and

means for transitioning the device to a user-interface unlock state if the detected contact corresponds to moving the unlock image across the touch-sensitive display according to a predefined displayed path on the touch-sensitive display; and

means for maintaining the device in the user-interface lock state if the detected contact does not correspond to moving the unlock image across the touch-sensitive display according to the predefined displayed path.

19. A portable electronic device, comprising:

a touch-sensitive display;

means for displaying a first unlock image and a second unlock image on the touch-sensitive display while the device is in a user-interface lock state;

means for detecting contact with the touch-sensitive display;

means for transitioning the device to a first active state corresponding to the first unlock image if the detected contact corresponds to a predefined gesture with respect to the first unlock image that moves the first unlock image along a first predefined displayed oath on the touch-sensitive display; and

means for transitioning the device to a second active state distinct from the first active state if the detected contact corresponds to a predefined gesture with respect to the second unlock image that moves the second unlock image along a second predefined displayed oath on the touch-sensitive display.

20. A computer program product for use in conjunction with a portable electronic device comprising a touch-sensitive display, the computer program product comprising a computer readable storage medium and an executable computer program mechanism embedded therein, the executable computer program mechanism comprising instructions for:

detecting contact with the touch-sensitive display while the device is in a user-interface lock state;

moving an unlock image along a predefined displayed path on the touch-sensitive display in accordance with the contact, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

transitioning the device to the user-interface unlock state if the detected contact corresponds to a predefined gesture; and

maintaining the device in the user-interface lock state if the detected contact does not correspond to the predefined gesture.

21. A computer program product for use in conjunction with a portable electronic device comprising a touch-sensitive display, the computer program product comprising a computer readable storage medium and an executable computer program mechanism embedded therein, the executable computer program mechanism comprising instructions for:

displaying an unlock image on the touch-sensitive display while the device is in a user-interface lock state, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

detecting contact with the touch-sensitive display;

transitioning the device to a user-interface unlock state if the detected contact corresponds to moving the unlock image along a predefined displayed path on the touch-sensitive display to a predefined location on the touch-sensitive display; and

maintaining the device in the user-interface lock state if the detected contact does not correspond to moving the unlock image along the predefined displayed path on the touch-sensitive display to the predefined location.

22. A computer program product for use in conjunction with a portable electronic device comprising a touch-sensitive display, the computer program product comprising a computer readable storage medium and an executable computer program mechanism embedded therein, the executable computer program mechanism comprising instructions for:

displaying an unlock image on the touch-sensitive display while the device is in a user-interface lock state, wherein the unlock image is a graphical, interactive user-interface object with which a user interacts in order to unlock the device;

detecting contact with the touch-sensitive display; and

transitioning the device to a user-interface unlock state if the detected contact corresponds to moving the unlock image across the touch-sensitive display according to a predefined displayed path on the touch-sensitive display; and

maintaining the device in the user-interface lock state if the detected contact does not correspond to moving the unlock image across the touch-sensitive display according to the predefined displayed path.

23. A computer program product for use in conjunction with a portable electronic device comprising a touch-sensitive display, the computer program product comprising a computer readable storage medium and an executable computer program mechanism embedded therein, the executable computer program mechanism comprising instructions for:

displaying a first unlock image and a second unlock image on the touch-sensitive display while the device is in a user-interface lock state;

detecting contact with the touch-sensitive display;

transitioning the device to a first active state corresponding to the first unlock image if the detected contact corresponds to a predefined gesture with respect to the first unlock image that moves the first unlock image along a first predefined displayed oath on the touch-sensitive display; and

transitioning the device to a second active state distinct from the first active state if the detected contact corresponds to a predefined gesture with respect to the second unlock image that moves the second unlock image along a second predefined displayed oath on the touch-sensitive display.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 7,657,849 B2
APPLICATION NO. : 11/322549
DATED : February 2, 2010
INVENTOR(S) : Chaudhri et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

On the Title page, at Item (75) Inventors:

please delete “Marcel Van Os” and insert --Marcel van Os--.

In the Claims:

Col. 23, line 7, please delete “oath” and insert --path--.

line 13, please delete “oath” and insert --path--.

Col. 24, line 41, please delete “oath” and insert --path--.

line 47, please delete “oath” and insert --path--.

Signed and Sealed this

Sixteenth Day of March, 2010

David J. Kappos

David J. Kappos
Director of the United States Patent and Trademark Office

Exhibit C

US006282646B1

(12) **United States Patent**
Hendry et al.

(10) **Patent No.: US 6,282,646 B1**
(45) **Date of Patent: Aug. 28, 2001**

(54) **SYSTEM FOR REAL-TIME ADAPTATION TO CHANGES IN DISPLAY CONFIGURATION**

(75) Inventors: **Ian Hendry**, San Jose; **Eric Anderson**, Los Gatos, both of CA (US); **Fernando Urbina**, Colorado Springs, CO (US)

(73) Assignee: **Apple Computer, Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/074,300**

(22) Filed: **May 8, 1998**

(51) **Int. Cl.**[7] **G06F 1/24**
(52) **U.S. Cl.** **713/100**
(58) **Field of Search** 713/100; 710/8, 710/10, 17, 46, 47, 48; 714/5, 7

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,014,193	*	5/1991	Garner et al.	364/200
5,276,630	*	1/1994	Baldwin et al.	364/505
5,282,268	*	1/1994	Mieras et al.	395/164
5,469,223	*	11/1995	Kimura	348/581
5,682,529	*	10/1997	Hendry et al.	395/653
5,825,359		10/1998	Derby et al.	345/344

* cited by examiner

Primary Examiner—William Grant
Assistant Examiner—Ronald D. Hartman, Jr.
(74) *Attorney, Agent, or Firm*—Burns, Doane, Swecker & Mathis, L.L.P.

(57) **ABSTRACT**

A hot-plugging capability for video devices is achieved by shifting the responsibility for recognizing changes in the configuration of a display environment from a computer's operating system to a device manager. When an input/output device is added to or removed from the computer system, an interrupt signal informs a device manager of the fact that a change in configuration has occurred. In response thereto, the device manager determines whether the changed component relates to the computer's display function. If so, the device manager makes a call to the computer's display manager, to inform it of the fact that the display configuration has changed. In response to this call, the display manager reconfigures the display space for the computer system and notifies clients as appropriate, to accommodate display features associated with the added component. With this change in the configuration of the display space, the added component becomes immediately available for use.

33 Claims, 3 Drawing Sheets

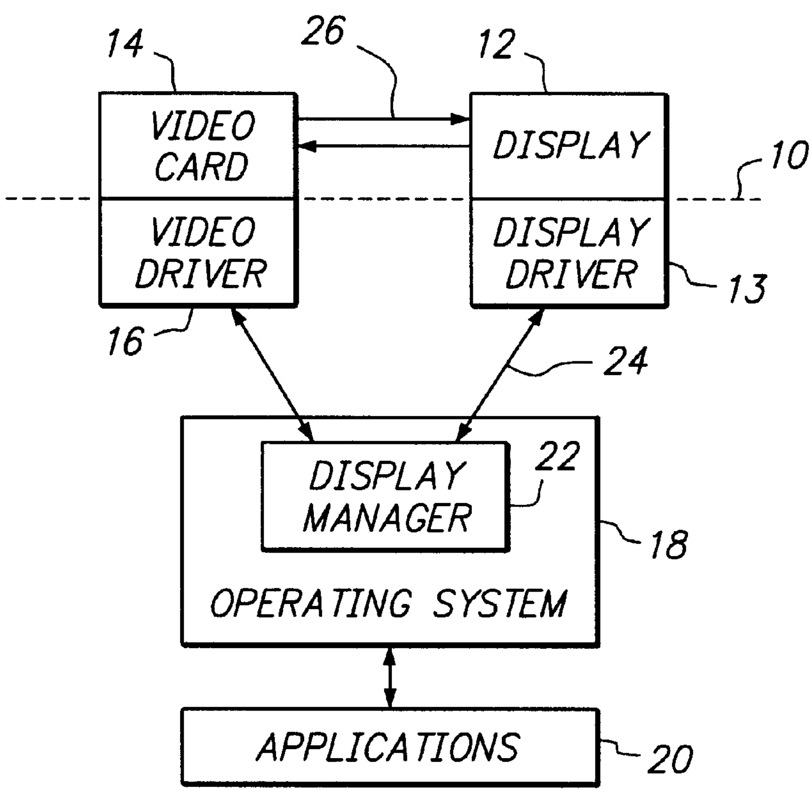


FIG. 1

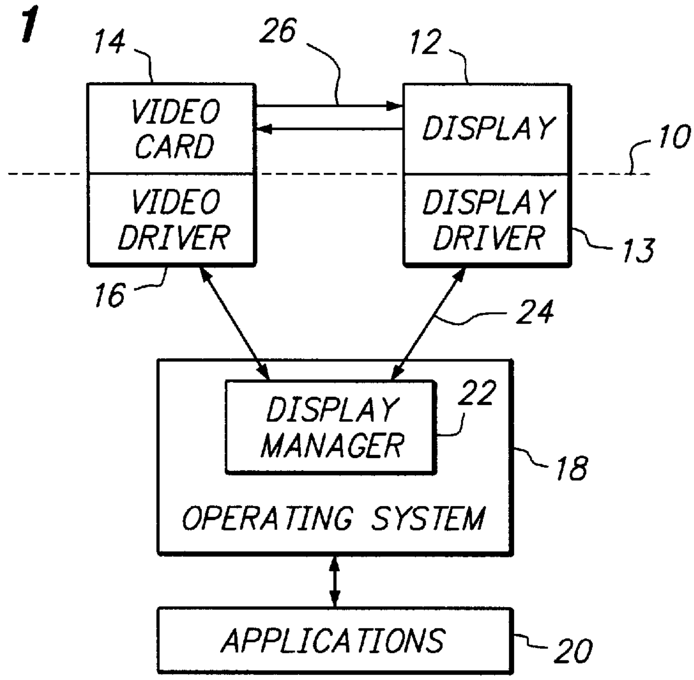

14
26
12
VIDEO CARD
DISPLAY
10
VIDEO DRIVER
DISPLAY DRIVER
13
16
24
DISPLAY MANAGER
22
18
OPERATING SYSTEM
APPLICATIONS
20

FIG. 2

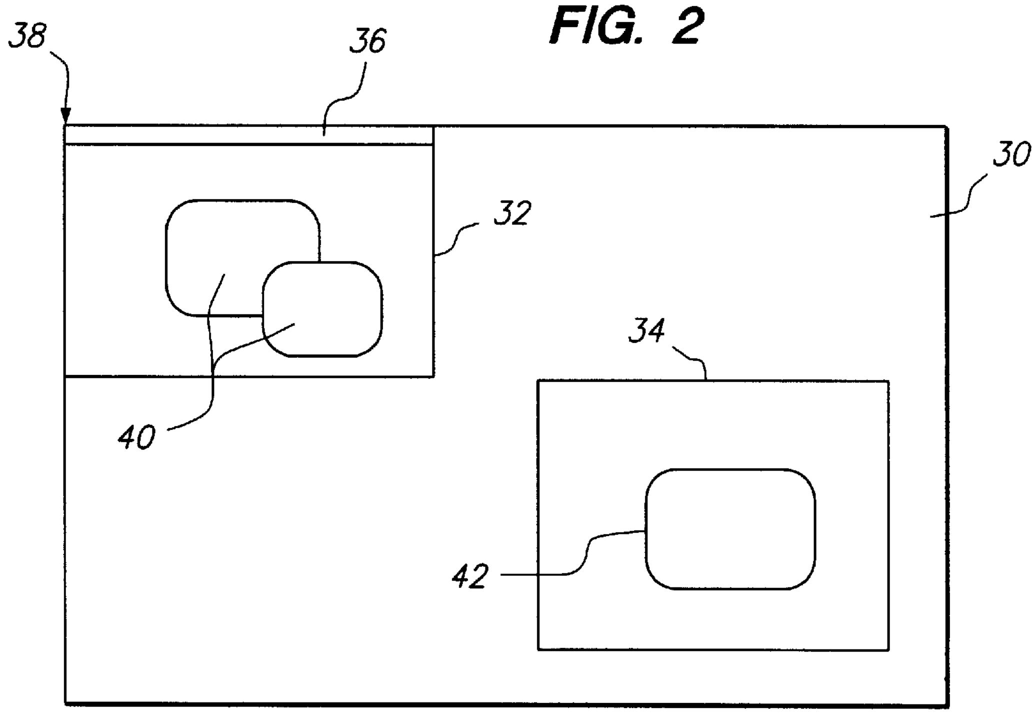

38
36
30
32
34
40
42

FIG. 3

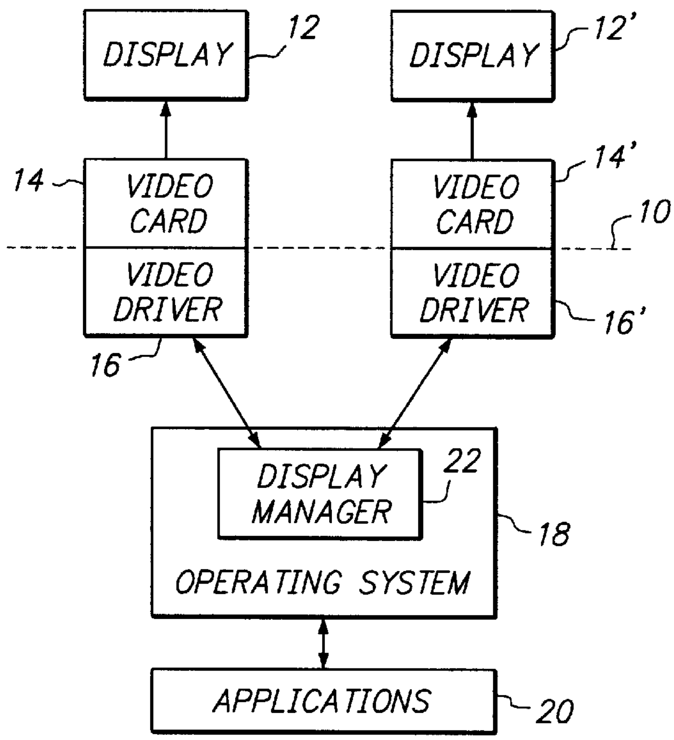
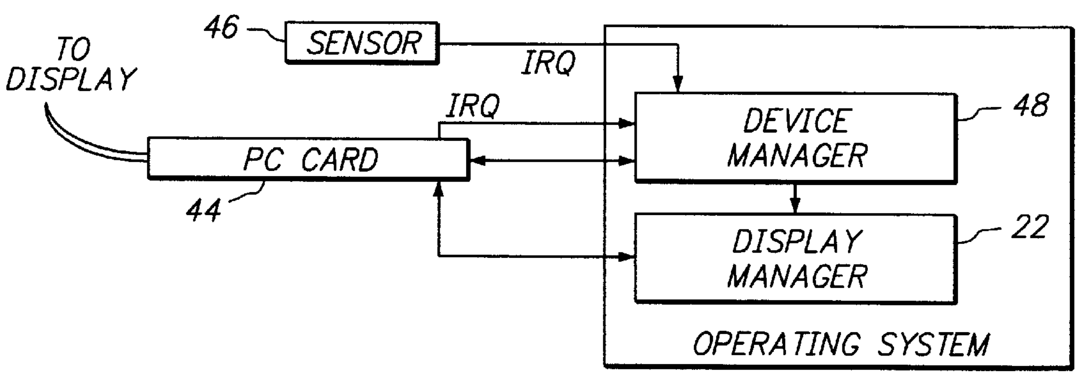

DISPLAY
12
DISPLAY
12'
14
VIDEO CARD
VIDEO DRIVER
VIDEO CARD
VIDEO DRIVER
14'
10
16
16'
DISPLAY MANAGER
22
18
OPERATING SYSTEM
APPLICATIONS
20

FIG. 4


46
SENSOR
TO DISPLAY
IRQ
IRQ
PC CARD
44
DEVICE MANAGER
48
DISPLAY MANAGER
22
OPERATING SYSTEM

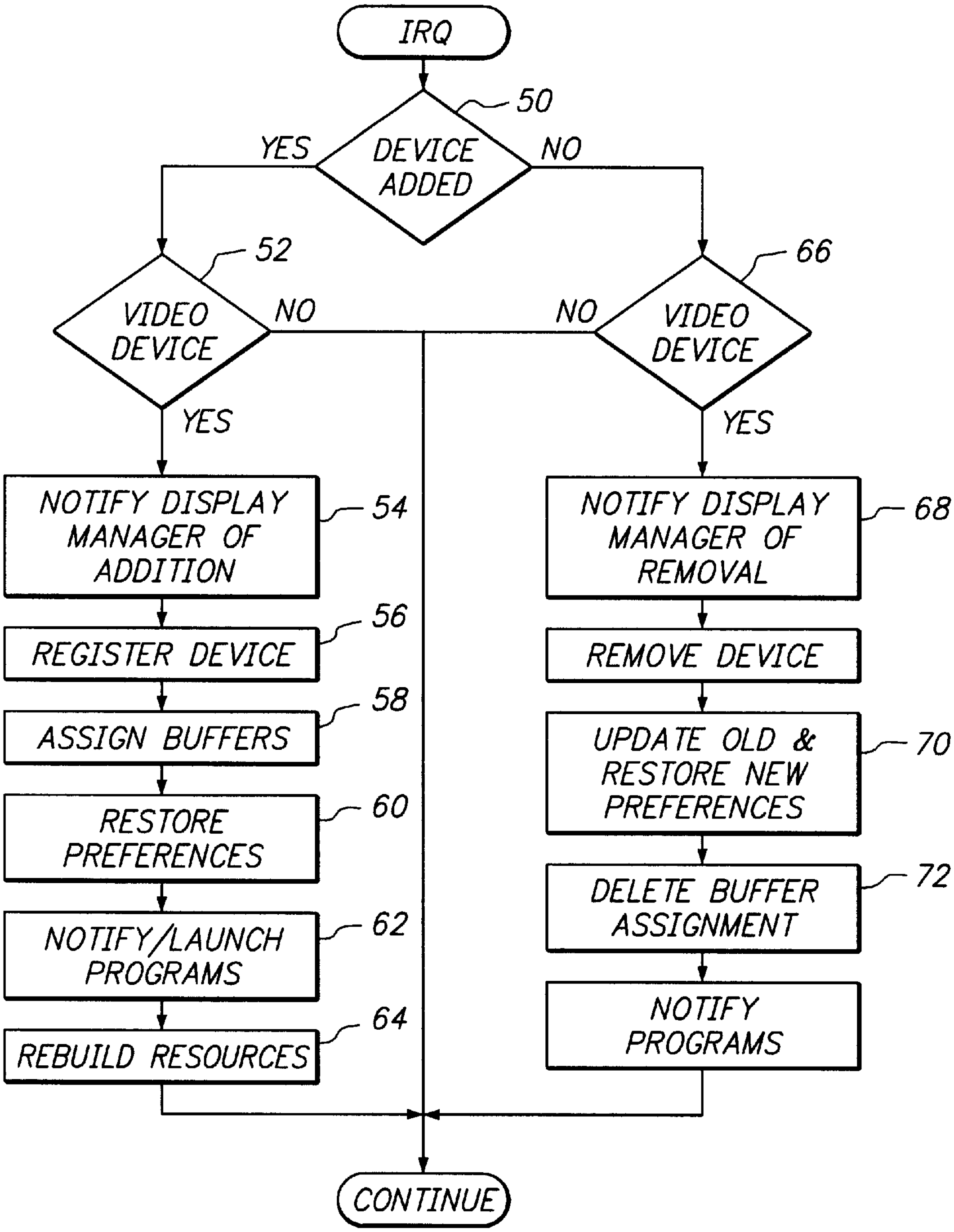

IRQ
50
DEVICE ADDED
YES
NO
52
VIDEO DEVICE
NO
66
VIDEO DEVICE
NO
YES
YES
NOTIFY DISPLAY MANAGER OF ADDITION
54
REGISTER DEVICE
56
ASSIGN BUFFERS
58
RESTORE PREFERENCES
60
NOTIFY/LAUNCH PROGRAMS
62
REBUILD RESOURCES
64
NOTIFY DISPLAY MANAGER OF REMOVAL
68
REMOVE DEVICE
UPDATE OLD & RESTORE NEW PREFERENCES
70
DELETE BUFFER ASSIGNMENT
72
NOTIFY PROGRAMS
CONTINUE

FIG. 5

SYSTEM FOR REAL-TIME ADAPTATION TO CHANGES IN DISPLAY CONFIGURATION

FIELD OF THE INVENTION

The present invention is directed to computer display systems, and more particularly to a display system which is capable of instantaneously accommodating changes in the configuration of a computer system.

BACKGROUND OF THE INVENTION

As computers become more prevalent in everyday use, particularly personal type computers, users are employing them in a variety of different situations. Depending upon the particular situation, the user may desire to change the configuration of the display devices connected to the computer. For example, portable computers of the so-called laptop or notebook type have become increasingly popular because of their small size and light weight, making them suitable for use while traveling. Due to the need to keep their dimensions to a minimum, the display screens built into such computers are relatively small in size, and may offer only limited display capabilities. Therefore, when using one of these types of computers in an office environment, the user may connect it to a monitor having a larger display area and/or enhanced display capabilities. Such a connection might be made, for example, by means of a docking station which enables the portable computer to be conveniently connected to a variety of peripheral devices, or by inserting a video card in a PC Card slot.

Subsequently, the user might remove the added monitor, for example to take the computer home or to use it while traveling. In this situation, the built-in display screen must be used. In other words, the computer must route all information to be displayed to the built-in screen, rather than the port to which the external monitor was connected. In addition, the displayed information must be reformatted, or otherwise processed, to accommodate the display parameters of the built-in device.

In the past, changes in the configuration of the computer system, such as the addition or removal of display devices, only became effective upon a restart, or reboot, of the computer system. As part of its initial startup procedure, the computer's operating system detects the presence of each device driver loaded on the system, and registers each such detected driver to permit communications to be carried out between the operating system and the device with which the driver is associated. If a new device and corresponding driver are added to the system after this initialization procedure, the driver is not registered with the operating system, and therefore communications do not take place until the operating system goes through its initialization procedure again, e.g. upon the next reboot of the computer. Hence, if a user adds a monitor to the computer system, the monitor cannot be used to display information generated by the computer until it has been rebooted.

U.S. Pat. No. 5,682,529 discloses a system for dynamically accommodating changes in the display configuration of a computer, without the need to restart the computer. In the system of this patent, changes can be made to the display environment for a computer system while it is in a sleep mode, in which the computer's central processing unit is maintained in a minimal operating state. When the computer is "awakened" from this sleep mode, the system of the '529 patent enables the changed configuration to be immediately recognized, and thereafter utilized in the display of information generated by the computer.

As the capabilities offered by personal computers continue to expand, the opportunities for changing the configuration of computers grows in a concomitant manner. In some situations, users may desire to have changes in the configuration of the computer's display environment become instantaneously effective, without the need to restart the computer or even place it in a sleep mode. For example, the user may create a slide presentation on a notebook computer. During the course of a meeting, a user may desire to immediately display the slide presentation, by connecting the computer to a suitable video projector, or the like. It is desirable to be able to carry out this operation without the need to first put the computer to sleep, and thereby reduce the time needed to operate within the changed configuration. It is an objective of the present invention, therefore, to expand upon the capabilities of the system of the '529 patent, by providing a display environment in which so-called "hot plugging" of displays is possible, wherein a display becomes immediately available for use as soon as it is plugged into the computer system.

SUMMARY OF THE INVENTION

In accordance with the present invention, the foregoing objective is achieved by utilizing a device manager to automatically recognize and react to changes in the configuration of a display environment, rather than wait for the computer's operating system to proactively determine the status of the display environment, for example upon restarting. When an input/output device is added to or removed from the computer system, an interrupt signal informs the device manager of the fact that a change in configuration has occurred. In response thereto, the device manager determines whether the changed component relates to the computer's display function. For example, it may determine whether an added device is a video card. If so, the device manager makes a call to the computer's display manager, to inform it of the fact that the display configuration has changed. In response to this call, the display manager reconfigures the display space for the computer system, to accommodate an additional frame buffer that is associated with the added component. With this change in the configuration of the display space, the added component becomes immediately available for use.

By means of this approach, the user can add a second monitor or other hardware component to a computer and begin to use the monitor as soon as it has been connected, without the need to reboot the computer or otherwise interrupt its current operating state.

Further features and advantages of the invention are explained in detail hereinafter in the context of specific embodiments that are described with reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an overall display system architecture;

FIG. **2** illustrates an example of a display environment;

FIG. **3** is a block diagram of a display system architecture which includes plural video cards and display devices;

FIG. **4** is a block diagram illustrating the operation of the device manager; and

FIG. **5** is a flowchart illustrating the process by which changes in the configuration of the display environment become immediately available to the user, in accordance with the principles of the invention.

DETAILED DESCRIPTION

The present invention is directed to the display environment of a computer system. A block diagram of the overall architecture for a display environment is illustrated in FIG. **1**. In this figure, hardware components of the computer system are illustrated above a dashed line **10**, and software components are depicted below the line. These software components are stored in a suitable computer-readable medium, such as a magnetic disk, and loaded into the computer's working memory, i.e. RAM, for execution. The system can include display devices **12**, e.g. monitors, LCD screens and/or plasma displays, although actual display devices need not be physically present in order for the principles of the invention to be operative. Each display device is connected to, and controlled by, a video card **14** which operates in accordance with video driver software **16**. Although depicted as being on a separate substrate, such as a printed circuit board, the components of at least one video card could be incorporated with other components on a single substrate, such as the computer's motherboard.

One or more software programs, such as application programs **20**, generate information to be displayed on the display devices. Examples of such information include text, windows and other graphical objects, and control structures such as menus and dialog boxes. This information is presented to the display device through the computer's operating system **18**, which also generates its own information to be presented on the display. The operating system communicates with the display device through an associated display driver **13**, which constitutes a software component that corresponds to the hardware of the display device **12**.

The operating system includes a display manager **22**, which provides communication between each of the software components, and dynamically configures the display devices **12**. The communication between the various software components and the hardware devices takes place via their associated drivers, e.g. the video driver and the display driver. In this regard, many video displays have the capability to provide information regarding their available modes of operation and/or timing specifications. Some displays, so-called "smart displays," are capable of providing information about their modes of operation directly, for example in response to inquiries. For these types of displays, the display manager **22** communicates directly with the display device, by means of the display driver **13**, over a communication channel **24**. This communication channel can be a bus within the computer, a serial line, or any other suitable path for exchanging information between the display manager and the display driver **13** of the display device.

In some cases, the display device may not be able to communicate its capabilities directly. However, through the use of a lookup table or the like, the display driver **13** can obtain information regarding the display's capabilities, and provide them to the display manager.

The display manager also communicates with other parts of the operating system **18** and the other software programs **20** that are running on the computer. For example, in response to operator commands, the operating system can instruct the display manager to add a new device to a list of active displays, or remove a device therefrom. In response thereto, the display manager informs the application programs **20** of the new display configurations, to enable them to update their displayed information accordingly.

In one known implementation for computer systems, the display environment can generally be considered to be defined by a global coordinate space **30**, as depicted in FIG. **2**. Objects and other information to be displayed can be positioned anywhere within this space, as determined by the user and/or the software program that generates the information. A reference point in this space, e.g. its origin or 0,0 coordinate point, is usually established with reference to some object that is always present in the display. For instance, most graphical user interfaces include some type of menu bar or other structure which enables the user to access basic commands to control the computer. The device which displays this menu bar is known as the main display device. If the computer system contains multiple display devices, only one of the devices is designated as the main device, even if multiple devices contain the menu bar. The origin of the coordinate display space is typically established with reference to the menu bar. For example, as illustrated in FIG. **2**, the 0,0 point **38** in the coordinate space can coincide with the top left corner of a menu bar **36**. The positions of all objects and other information to be displayed in the display space **30** are defined by their coordinates within this space. The operating system receives this coordinate information, for example from the software programs **20** which generate the information, and provides it to the display driver to cause the information to appear at the appropriate place on the screen of the display device located at the corresponding position in the display space.

In the example illustrated in FIG. **2**, the display environment consists of two display devices, **32** and **34**, within the global display space **30**. A menu bar **36** is displayed at the top of the screen for the device **32**, which is therefore the main display device. Accordingly, the origin **38** of the display space coincides with the top left corner of the device **32**. As illustrated in FIG. **2**, the user has caused some objects, e.g. windows **40**, to be displayed on the device **32**, and another object **42** to be displayed on the device **34**.

FIG. **3** illustrates the configuration of the computer system for the particular example illustrated in FIG. **2**, which includes two display devices. Each display device is connected to an associated video card, which includes a corresponding video driver. For the sake of simplicity in FIG. **3**, the display drivers are not separately illustrated, but are assumed to be present within the system, in a manner analogous to the arrangement shown in FIG. **1**. The embodiment of FIG. **3** includes two video cards **14** and **14'**, respectively associated with the two video display devices **12** and **12'**. Each of the video cards communicates with the display manager **22**, by means of its associated video driver **16** and **16'**.

Among other components, each video card includes a frame buffer, e.g. random access memory, which stores the data for the image that is displayed on its associated display device **12**. In essence, the display manager **22** assigns the frame buffer to a corresponding portion of the global coordinate space **30**. In the example of FIG. **2**, the two frame buffers are assigned to mutually exclusive portions of the global space. However, some or all of the portion assigned to one of the frame buffers could overlap with the area assigned to the other frame buffer. In this case, the same image, or portion of an image, appears on both display devices.

At any given time, there could be only one video card connected to the computer, or both cards could be connected. Furthermore, in the case of a network server or the like, it is possible that no video card would be present over certain periods of time. Even when both cards are present, only one of them may have a monitor or other display device connected to it at any particular point in time.

In the case of a conventional desktop or notebook computer system, one of the video cards might be incorporated within the structure of the computer system, and may not be designed to be repeatedly inserted and removed by the user. Additional video cards, however, might be capable of being easily inserted into and removed from the computer system. For example, the video card might be implemented in a removable card that conforms to the PC Card standard. This standard defines the form factor for relatively small, credit-card shaped I/O devices, which are designed to be easily inserted into and removed from computer housings, to provide a computer with different capabilities. Included among the types of I/O devices that can be embodied in such a card are modems, facsimile devices, network interface cards, wireless communications devices and hard disk drives.

Devices of this type which conform to this standard, commonly known as PC Cards, are designed to be readily inserted and removed from the computer housing. The detection of the presence of such devices, as well as their removal from the system, is handled by a portion of the computer's operating system that is referred to herein as a device manager. Referring to FIG. **4**, when a PC Card **44** is inserted into the housing of the computer, it actuates a switch **46**, or equivalent sensor device, which sends an interrupt signal IRQ to the device manager **48**. In response to this interrupt, the device manager determines the type of device which has been inserted, and informs the operating system **18**. In a similar manner, whenever the PC Card is removed from the computer housing, an interrupt is also sent to the device manager, which in turn notifies the operating system that the device is no longer available.

Another type of change which can be made to the display configuration of the computer is the addition or removal of a display device. In the example of FIG. **3**, for instance, either one of the display devices **12** or **12**' could be disconnected from its associated video card **14** or **14**'. Furthermore, if only one display device is present, it could be disconnected from one of the video cards **14** and connected to the other video card **14**'. Whenever a change of this nature occurs, an interrupt is sent to the device manager **48**. For instance, the interrupt could be generated by the video card, upon detecting that a display device has been physically connected to or disconnected from it. Alternatively, the interrupt could be provided by a bus that is capable of detecting such a change.

The addition or removal of other types of hardware can also result in a change in the display configuration of the computer system. For example, a graphics accelerator card can be added to the system by means of a PC Card slot. Again, upon the addition or removal of such a device, an interrupt signal IRQ is sent to the device manager.

The present invention is particularly directed to the situation in which the device that is added to or removed from the computer system is related to the display function. In the past, it was necessary to reboot the computer system in order for a change in video hardware to become effective. More particularly, unless a reboot occurred, the operating system was not prompted to undertake any action which would cause it to detect the presence of a new driver, resulting from the addition of an associated hardware device. Hence, it was necessary for the user to interrupt the operating state of the computer in order to utilize the additional functionality provided by a newly added hardware. Once the operating system became aware of the presence of the new driver, it could notify the display manager to incorporate the presence of the new frame buffer.

In accordance with the present invention, however, the display system can be immediately responsive to the addition or removal of video hardware, so that the capabilities of a revised configuration can be employed without the need to change the operating state of the computer. This functionality is accomplished by, in effect, bypassing the need to have the operating system actively determine the addition or removal of a display device. Rather, the notification of a change in the configuration of the display environment is provided directly to the display manager, so that it can directly account for the presence or absence of a particular video device.

To this end, whenever a PC Card is added to or removed from the computer system, the device manager determines whether the card relates to a display function. A similar determination is made whenever an interrupt is generated that indicates some other type of hardware has been added or removed, e.g. a display monitor. In addition to, or in lieu of interrupts, other approaches can be employed to determine when a device has been added or removed. For example, the operating system can periodically poll all of the computer system's I/O ports, to determine which devices are present and which ones might have been removed.

Referring to FIG. **5**, upon receipt of an indication that there has been a change in configuration, the device manager first determines at step **50** whether a device has been added or removed. If a device has been added to the system, the device manager communicates with the device to determine its type, at step **52**, and stores data in a register regarding the identity and type of the device. If the device responds with an indication that it is a video device, the device manager issues a call to the display manager **22**, at step **54**. Appropriate parameters can be included with the call, to indicate the type of device, the size of its frame buffer (if applicable), its resolution, and the like.

In response to this information, the display manager carries out a number of operations, depicted in Steps **56**–**64**. First, it registers the added hardware as a new device, along with the location of its associated drivers in memory, at step **56**. In some cases, the driver may already be present in memory, but in an inactive state because the device was not connected to the system at the time of initial boot. In this case, the display manager switches the driver to an active state.

After registering the device, the display manager matches each display device with an available frame buffer at step **58**. If a new video card is inserted, for example, the display manager assigns a portion of the global coordinate space **30** to the frame buffer in the video card. If a display device is connected to that video card, the display manager assigns that device to the frame buffer for that card, so that the proper data is displayed on the device. If a display device is disconnected from one video card and connected to a different video card, the display manager moves objects within the global space **30** so that they are presented to the appropriate frame buffer for the display device. For example, the display manager can move user interface control objects which are specific to that display, such as brightness and contrast controls, to the frame buffer associated with that display. Similarly, if the display has other attributes associated with it, such as a certain name or designation, the display manager ensures that they are directed to the proper frame buffer.

Once the display devices and frame buffers are matched up, the display manager consults a preferences file which indicates whether that device was connected to the system at some previous time. This file is preferably stored in permanent memory, such as a hard disk, and updated each time a video device is added to or removed from the computer system to change its configuration. The file stores the state of the display just prior to the change. For instance, it may store the location of the menu bar and all other objects on the desktop of the user interface, along with each device that made up the configuration. It can store relevant operating parameters for the device as well, such as timing values, color depth, and the like. When a new device is added, the display manager determines whether the changed configuration had existed previously, and if so it attempts to restore the display environment to the state that existed the last time that configuration was present, at step **60**. For example, it might move the menu bar and certain icons to the screen of a newly added display device, if they were previously located on that device before it was disconnected, or set the timing of an added frame buffer.

In addition to these actions, at step **62** the display manager can inform currently executing programs of the presence of the new video device, as described in detail in U.S. Pat. No. 5,682,529, the disclosure of which is incorporated herein by reference. Furthermore, other software which is not executing, but which relates to the new device, can be activated. For instance, if a graphics accelerator card is inserted in a PC Card slot, the software associated with that card can be automatically launched.

The display manager also functions at step **64** to rebuild or reconfigure resources that might be employed by the new device. For example, in some display modes a table lookup operation is carried out to determine the colors which are displayed on the monitor. Different application programs may utilize different tables for this purpose. Typically, the window which is in the foreground of a display controls the particular table that is used. Therefore, when windows are moved onto a new display device, for instance in accordance with the preferences file, the display manager rebuilds the color lookup table stored in the frame buffer for that device so that it corresponds to the appropriate window.

If the device manager determines at step **50** that a device has been removed from the system, rather than added, it determines at step **66** whether the removed hardware was a video device, for example by reference to previously stored information which indicated the type of card inserted into each PC Card slot, or the like. If a determination is made that the removed device was part of the video subsystem, the device manager sends a call to the display manager **22**, at step **68**, to inform it of this fact. In response thereto, the display manager updates the preferences file at step **70**, to record the relevant parameters that pertain to the most recent configuration. In addition, the display manager can attempt to restore preferences that pertain to the new condition. Thus, for example, if the computer system had two display devices and one was removed, the preferences file would be updated to store the relevant data for the two-monitor configuration, and then searched to determine whether it contains data for a one-monitor configuration. The display manager then rebuilds the display configuration, at step **72**. For example, if the removed device is a video card, the display manager deletes the assignment of a portion of the display space to the now-removed frame buffer. In concert with this action, the display manager can also function to move objects that were previously displayed on the removed display device to an area associated with a remaining display device, as described in greater detail in U.S. Pat. No. 5,682,529. If the removed device had executing software associated with it, the display manager can cause the software to shut down.

A special case can occur if all frame buffers, e.g. video cards, are removed from the system. Since the frame buffer is a memory-mapped I/O device, a program or other device could attempt to address memory that is no longer present. Typically, when such a situation occurs, the operating system recognizes it as an error condition, and shuts down the process which attempted to address the non-existent memory. However, in the system of the present invention, if an attempt is made to address memory in a video device, e.g. a frame buffer, the resulting error condition is treated as an interrupt. This interrupt is relayed to the display manager to cause it to reconfigure the display environment, and delete the assignment of display space to the frame buffer which is no longer present.

Another action that can occur upon removal of a display device is to change the operating mode of its display driver. More particularly, some display devices can operate in one mode in which adjustment of control buttons, such as brightness and contrast controls, is accompanied by user interface feedback, e.g. variation of a sliding scale on the display. This operating mode requires communication with the computer's operating system. In another operating mode, referred to as a remote mode, no such user feedback is provided. In the system of the present invention, when a display device is to be removed, the display manager instructs its display driver to switch to the remote mode, so that no attempt is made to affect the user interface while the device is not present.

From the foregoing, it can be seen that the present invention provides a hot-plugging capability for video devices, that enables users to immediately take advantage of changes in the display configuration of a computer system, such as the addition of a new video card. This functionality is attained by providing notification of the changed configuration directly to the display manager, rather than waiting for an action that prompts the operating system to review the current configuration, such as rebooting the computer.

It will be appreciated by those of ordinary skill in the art that the present invention can be embodied in other specific forms without departing from the spirit or essential characteristics thereof. For example, while one embodiment of the invention has been described in the context of the insertion and removal of video cards that are embodied in PC Cards, it will be appreciated that the principles which underlie the invention are not limited to this particular implementation. Rather, any other suitable mechanism which accommodates the addition and removal of a video device can benefit from the features of the present invention. The presently disclosed embodiments are therefore considered in all respects to be illustrative, and not restrictive. The scope of the invention is indicated by the appended claims, rather than the foregoing description, and all changes that come within the meaning and range of equivalents thereof are intended to be embraced therein.

What is claimed is:

1. A method for reconfiguring a computer system to accommodate changes in a display environment, comprising the steps of:

detecting the addition or removal of an input/output device in the computer system;

determining whether an input/output device which has been added or removed is a video device, in response to said detection;

providing a notification to a display manager when a determination is made that a video device has been added or removed; and

modifying the allocation of display space to display devices via said display manager, in accordance with the addition or removal of a video device.

2. The method of claim 1 wherein the video device comprises a video card that includes a frame buffer, and said modification step includes assigning a portion of the display space to the frame buffer of an added video card, or deleting the assignment of a portion of the display space to a removed video card.

3. The method of claim 1 wherein said display manager further carries out the step of storing a preferences file that identifies the status of displayed objects prior to a change in the configuration of a computer.

4. The method of claim 3 wherein, upon detection of the addition of a video device, said display manager repositions objects in said display space, in accordance with a status stored in said preferences file.

5. The method of claim 1 wherein said video device is a display device, and said display manager carries out the step of assigning a respective frame buffer, which corresponds to an allocated portion of the display space, to a corresponding display device.

6. The method of claim 1 wherein, upon detection of the addition of a video device, said display manager causes a software program associated with the added device to be launched.

7. The method of claim 1 wherein said display manager further carries out the step of reconfiguring a computer resource to correspond to the status of objects located in the display space.

8. The method of claim 7 wherein said computer resource is a color look-up table.

9. The method of claim 1, further including the step of recognizing an error condition resulting from an attempt to address a frame buffer that has been removed, providing a notification to said display manager in response to said error condition, and deleting an allocation of display space to the removed frame buffer.

10. A system which provides hot-plugging capabilities for display devices, comprising:

a video device including a frame buffer for storing data that defines an image to be displayed on an associated display device;

a display manager which defines a display space and assigns a portion of said display space to said frame buffer, and which provides data for images to be displayed to said frame buffer; and

a device manager which detects the addition or removal of a device in a computer system, determines whether a device which has been added or removed is a video device, and provides a notification of such addition or removal to the display manager when a video device is determined to have been added or removed, to cause the assignment of a portion of the display space to be modified in accordance with a detected addition or removal.

11. The system of claim 10, wherein said display manager launches a software program associated with the video device in response to notification that the video device has been added.

12. The system of claim 11 further including a preference file stored in memory which indicates the status of objects being displayed when a video device is removed.

13. A system which provides hot-plugging capabilities for display devices, comprising:

at least one display for displaying images;

a display manager which defines a display space and assigns a portion of said display space to a display device, and which provides data for images to be displayed on said display device; and

a device manager which detects the addition or removal of a device in a computer system, determines whether a device which has been added or removed is a display device, and provides a notification of such addition or removal to the display manager when a display device is determined to have been added or removed, to cause the assignment of a portion of the display space to be modified in accordance with a detected addition or removal.

14. The system of claim 13 further including a frame buffer which is associated with an assigned portion of the display space, and wherein said display manager modifies said assignment by associating said frame buffer with said display device.

15. The system of claim 13 further including a preference file stored in memory which indicates the status of objects being displayed when a display device is removed.

16. A computer-readable medium containing a device manager program and a display manager program, wherein said device manager program performs the steps of

detecting the addition or removal of an input/output device in a computer system,

determining whether the input/output device is a video device, and

providing a notification to the display manager program when a video device is added or removed;

and wherein said display manager performs the step of:

modifying the allocation of display space to display devices in response to said notification from the device manager.

17. The computer-readable medium of claim 16, wherein said display manager further performs the steps of storing a preference file relating to the status of objects appearing on a display device, and restoring objects to the status stored in the preferences file when a video device is added.

18. The computer-readable medium of claim 16, wherein said display manager performs the step of assigning a respective frame buffer to a display device in response to said notification of an added display device, or deleting the assignment of a respective frame buffer from said display device in response to said notification of a removed display device.

19. The computer-readable medium of claim 16, wherein said display manager performs the further step of launching a software program in response to said notification.

20. The computer-readable medium of claim 16, wherein said display manager performs the further step of reconfiguring at least one computer resource in accordance with the modification of the display space allocation.

21. The computer-readable medium of claim 20, wherein said computer resource is a color look-up table.

22. The method of claim 3, wherein said preference file stores the video devices which make up the configuration of the computer, and the locations of objects displayed on said video devices.

23. The method of claim 22, wherein said preferences file also stores operating parameters for said devices.

24. A method for reconfiguring a computer system to accommodate changes in a display environment, comprising the steps of:

detecting the addition or removal of a video device in the computer system;

providing a notification to a display manager that a video device has been added or removed;

modifying the allocation of display space to display devices via said display manager, in accordance with the addition or removal of a video device; and reconfiguring a color look-up table to correspond to the status of objects located in the display space.

25. A system which provides hot-plugging capabilities for display devices, comprising:

a video device including a frame buffer for storing data that defines an image to be displayed on an associated display device;

a display manager which defines a display space and assigns a portion of said display space to said frame buffer, and which provides data for images to be displayed to said frame buffer;

a device manager which detects the addition or removal of the video device to a computer system, and provides a notification of such addition or removal to the display manager to cause the assignment of a portion of the display space to be modified in accordance with a detected addition or removal; and

means responsive to the removal of a video device for storing a preference file in memory which indicates the status of objects being displayed.

26. The system of claim **25**, wherein said display manager is responsive to the addition or removal of a video device to restore displayed objects to a status stored in said preference file which corresponds to the configuration of the computer system after the video device is added or removed.

27. The system of claim **25**, wherein said preference file stores the video devices which make up the configuration of the computer, and the locations of objects displayed on said video devices.

28. The system of claim **27**, wherein said preferences file also stores operating parameters for said devices.

29. A computer-readable medium containing a device manager program and a display manager program, wherein said device manager program performs the steps of:

detecting the addition or removal of a video device in a computer system, and

providing a notification to the display manager program when a video device is added or removed;

and wherein said display manager performs the steps of:

modifying the allocation of display space to display devices in response to said notification from the device manager, and

storing a preference file relating to the status of objects appearing on a display device, and restoring objects to the status stored in the preferences file when a video device is added.

30. The computer-readable medium of claim **29**, wherein said preference file stores the video devices which make up the configuration of the computer, and the locations of objects displayed on said video devices.

31. The computer-readable medium of claim **30**, wherein said preferences file also stores operating parameters for said devices.

32. A computer-readable medium containing a device manager program and a display manager program, wherein said device manager program performs the steps of:

detecting the addition or removal of a video device in a computer system, and

providing a notification to the display manager program when a video device is added or removed;

and wherein said display manager performs the step of:

modifying the allocation of display space to display devices in response to said notification from the device manager, and

reconfiguring at least one computer resource in accordance with the modification of the display space allocation.

33. The computer-readable medium of claim **32**, wherein said computer resource is a color look-up table.

* * * * *

Exhibit D

US007380116B2

(12) **United States Patent**
Hendry et al.

(10) **Patent No.: US 7,380,116 B2**
(45) **Date of Patent: *May 27, 2008**

(54) **SYSTEM FOR REAL-TIME ADAPTATION TO CHANGES IN DISPLAY CONFIGURATION**

(75) Inventors: **Ian Hendry**, San Jose, CA (US); **Eric Anderson**, Los Gatos, CA (US); **Fernando Urbina**, Colorado Springs, CO (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 194 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **11/198,289**

(22) Filed: **Aug. 8, 2005**

(65) **Prior Publication Data**

US 2005/0273591 A1 Dec. 8, 2005

Related U.S. Application Data

(63) Continuation of application No. 09/927,411, filed on Aug. 13, 2001, now Pat. No. 6,928,543, which is a continuation of application No. 09/074,300, filed on May 8, 1998, now Pat. No. 6,282,646.

(51) **Int. Cl.**
G06F 1/24 (2006.01)

(52) **U.S. Cl.** **713/100**; 710/104; 345/545

(58) **Field of Classification Search** 713/100; 710/104; 345/545

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,922,448 A 5/1990 Kunieda et al.
5,014,193 A 5/1991 Garner et al.
5,276,458 A * 1/1994 Sawdon 345/698
5,276,630 A 1/1994 Baldwin et al.
5,282,268 A 1/1994 Mieras et al.
5,309,552 A 5/1994 Horton et al.
5,375,210 A 12/1994 Monnes et al.
5,386,567 A * 1/1995 Lien et al. 713/100
5,469,223 A 11/1995 Kimura
5,535,415 A 7/1996 Kondou et al.

(Continued)

OTHER PUBLICATIONS

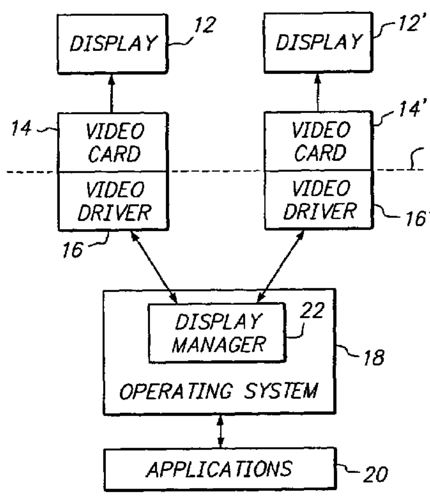
"Software Architecture for the Support of Multiple Adapters on an Interrupt Level"—IBM Technical Disclosure Bulletin—Sep. 1, 1986—vol. #29—pp. 1704-1707.*

Primary Examiner—Thuan Du
(74) *Attorney, Agent, or Firm*—Buchanan Ingersoll & Rooney PC

(57) **ABSTRACT**

A hot-plugging capability for video devices is achieved by shifting the responsibility for recognizing changes in the configuration of a display environment from a computer's operating system to a device manager. When an input/output device is added to or removed from the computer system, an interrupt signal informs a device manager of the fact that a change in configuration has occurred. In response thereto, the device manager determines whether the changed component relates to the computer's display function. If so, the device manager makes a call to the computer's display manager, to inform it of the fact that the display configuration has changed. In response to this call, the display manager reconfigures the display space for the computer system and notifies clients as appropriate, to accommodate display features associated with the added component. With this change in the configuration of the display space, the added component becomes immediately available for use.

43 Claims, 3 Drawing Sheets



U.S. PATENT DOCUMENTS

5,590,376	A		12/1996	Kou	
5,682,529	A		10/1997	Hendry et al.	
5,825,359	A		10/1998	Derby et al.	
5,923,307	A	*	7/1999	Hogle, IV	345/4
5,977,934	A		11/1999	Wada et al.	
6,049,316	A		4/2000	Nolan et al.	

* cited by examiner

FIG. 1

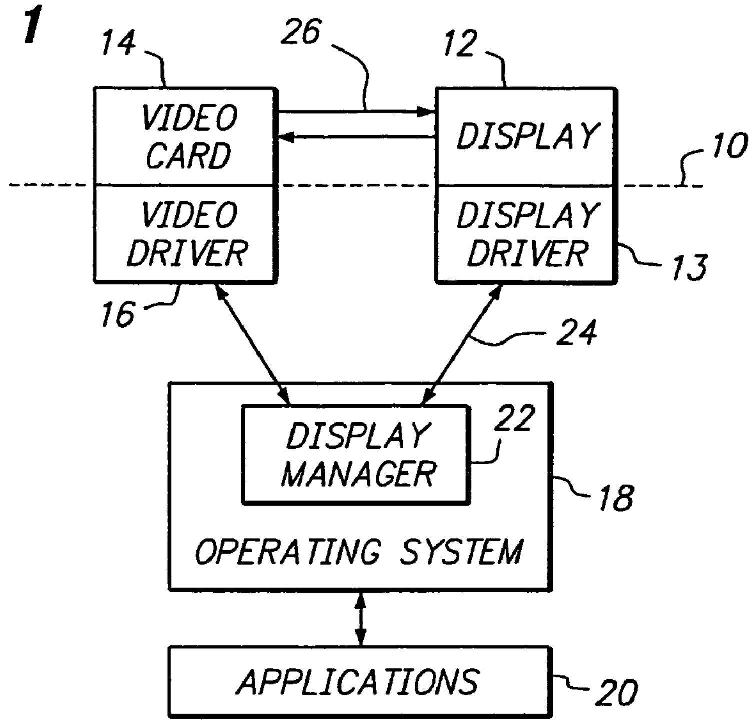

14
26
12
VIDEO CARD
DISPLAY
10
VIDEO DRIVER
DISPLAY DRIVER
13
16
24
DISPLAY MANAGER
22
18
OPERATING SYSTEM
APPLICATIONS
20

FIG. 2

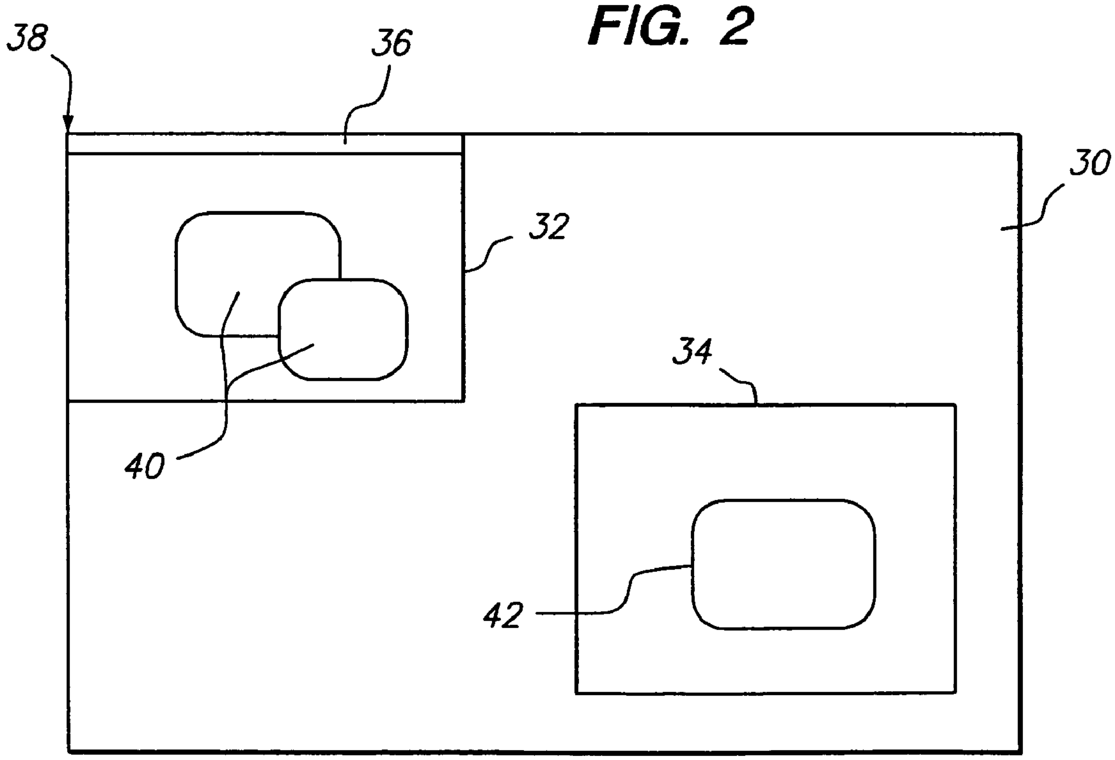

38
36
30
32
34
40
42

FIG. 3

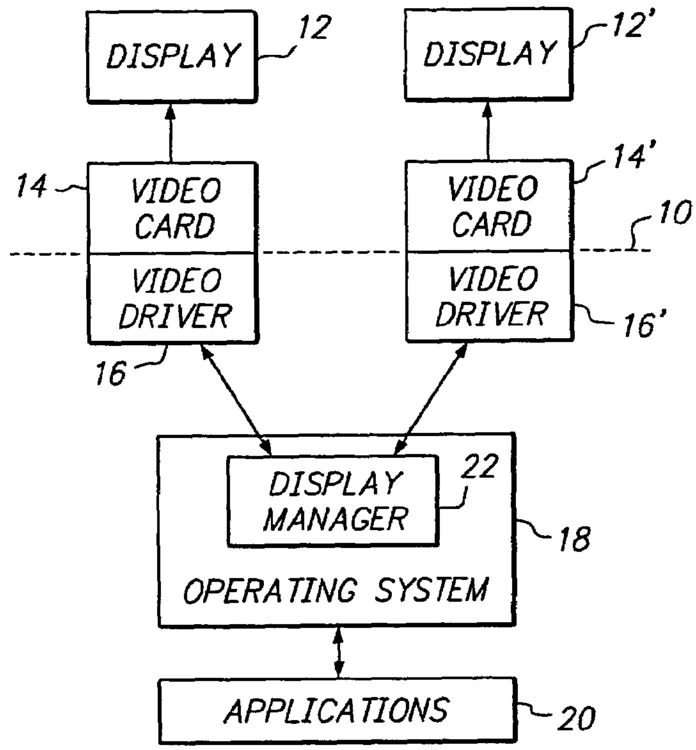
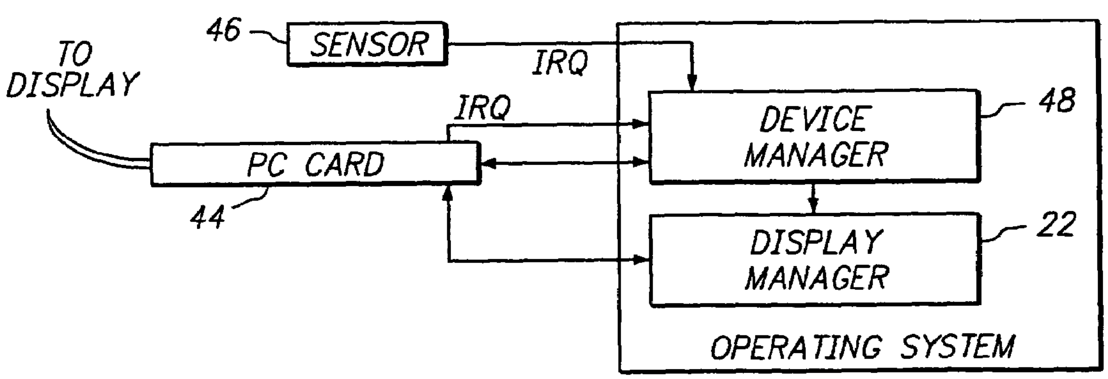

DISPLAY
12
DISPLAY
12'
14
VIDEO CARD
VIDEO DRIVER
VIDEO CARD
VIDEO DRIVER
14'
10
16
16'
DISPLAY MANAGER
22
OPERATING SYSTEM
18
APPLICATIONS
20

FIG. 4


TO DISPLAY
46
SENSOR
IRQ
IRQ
PC CARD
44
DEVICE MANAGER
48
DISPLAY MANAGER
22
OPERATING SYSTEM

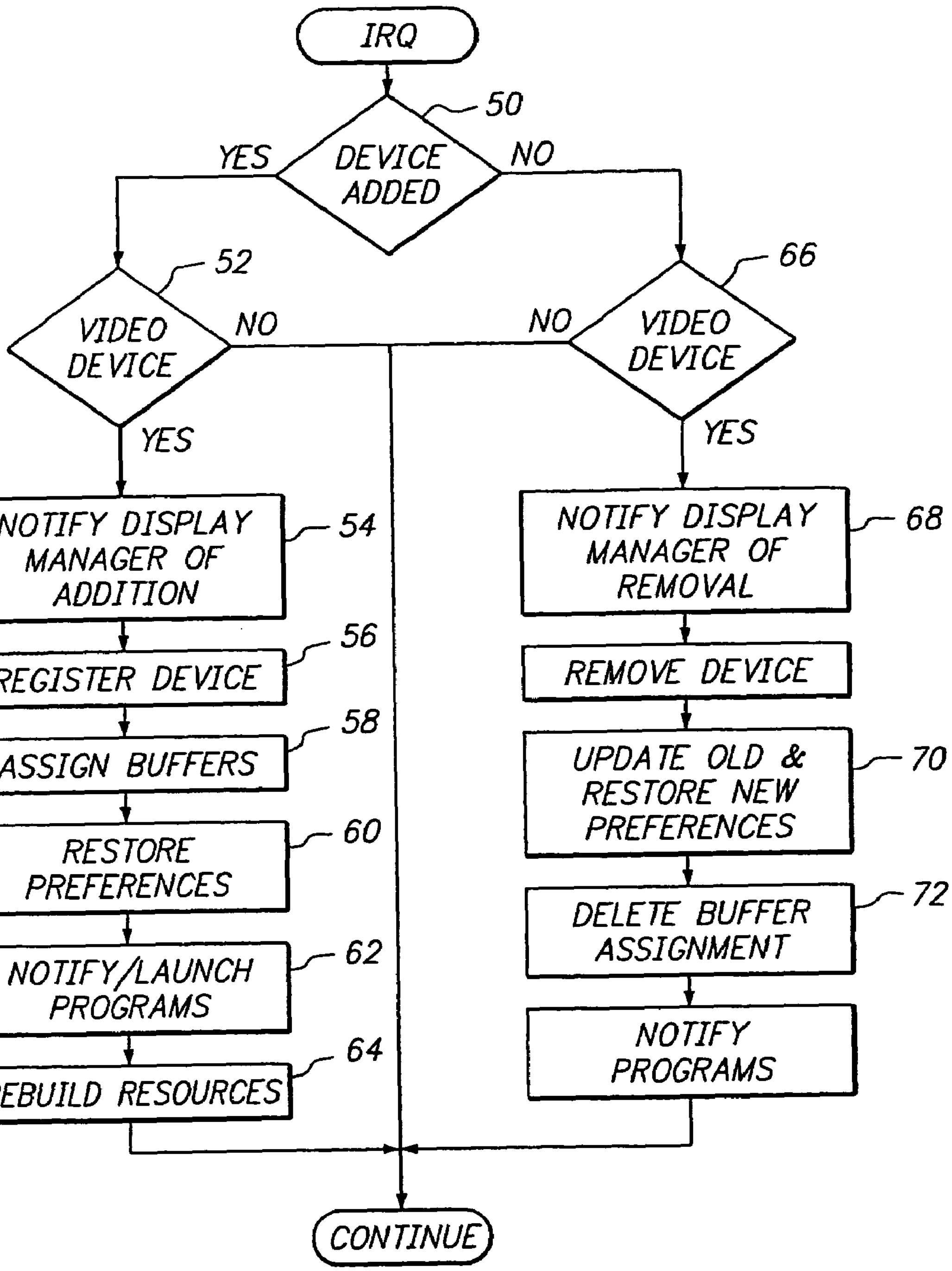

IRQ
50
DEVICE ADDED
YES
NO
52
VIDEO DEVICE
NO
66
VIDEO DEVICE
NO
YES
YES
NOTIFY DISPLAY MANAGER OF ADDITION
54
REGISTER DEVICE
56
ASSIGN BUFFERS
58
RESTORE PREFERENCES
60
NOTIFY/LAUNCH PROGRAMS
62
REBUILD RESOURCES
64
NOTIFY DISPLAY MANAGER OF REMOVAL
68
REMOVE DEVICE
UPDATE OLD & RESTORE NEW PREFERENCES
70
DELETE BUFFER ASSIGNMENT
72
NOTIFY PROGRAMS
CONTINUE

FIG. 5

SYSTEM FOR REAL-TIME ADAPTATION TO CHANGES IN DISPLAY CONFIGURATION

This application is a continuation of application Ser. No. 09/927,411 filed on Aug. 13, 2001 now U.S. Pat. No. 6,928,543, which is a continuation of application Ser. No. 09/074,300 filed on May 8, 1998 now U.S. Pat. No. 6,282, 646.

FIELD OF THE INVENTION

The present invention is directed to computer display systems, and more particularly to a display system which is capable of instantaneously accommodating changes in the configuration of a computer system.

BACKGROUND OF THE INVENTION

As computers become more prevalent in everyday use, particularly personal type computers, users are employing them in a variety of different situations. Depending upon the particular situation, the user may desire to change the configuration of the display devices connected to the computer. For example, portable computers of the so-called laptop or notebook type have become increasingly popular because of their small size and light weight, making them suitable for use while traveling. Due to the need to keep their dimensions to a minimum, the display screens built into such computers are relatively small in size, and may offer only limited display capabilities. Therefore, when using one of these types of computers in an office environment, the user may connect it to a monitor having a larger display area and/or enhanced display capabilities. Such a connection might be made, for example, by means of a docking station which enables the portable computer to be conveniently connected to a variety of peripheral devices, or by inserting a video card in a PC Card slot.

Subsequently, the user might remove the added monitor, for example to take the computer home or to use it while traveling. In this situation, the built-in display screen must be used. In other words, the computer must route all information to be displayed to the built-in screen, rather than the port to which the external monitor was connected. In addition, the displayed information must be reformatted, or otherwise processed, to accommodate the display parameters of the built-in device.

In the past, changes in the configuration of the computer system, such as the addition or removal of display devices, only became effective upon a restart, or reboot, of the computer system. As part of its initial startup procedure, the computer's operating system detects the presence of each device driver loaded on the system, and registers each such detected driver to permit communications to be carried out between the operating system and the device with which the driver is associated. If a new device and corresponding driver are added to the system after this initialization procedure, the driver is not registered with the operating system, and therefore communications do not take place until the operating system goes through its initialization procedure again, e.g. upon the next reboot of the computer. Hence, if a user adds a monitor to the computer system, the monitor cannot be used to display information generated by the computer until it has been rebooted.

U.S. Pat. No. 5,682,529 discloses a system for dynamically accommodating changes in the display configuration of a computer, without the need to restart the computer. In the system of this patent, changes can be made to the display

environment for a computer system while it is in a sleep mode, in which the computer's central processing unit is maintained in a minimal operating state. When the computer is "awakened" from this sleep mode, the system of the '529 patent enables the changed configuration to be immediately recognized, and thereafter utilized in the display of information generated by the computer.

As the capabilities offered by personal computers continue to expand, the opportunities for changing the configuration of computers grows in a concomitant manner. In some situations, users may desire to have changes in the configuration of the computer's display environment become instantaneously effective, without the need to restart the computer or even place it in a sleep mode. For example, the user may create a slide presentation on a notebook computer. During the course of a meeting, a user may desire to immediately display the slide presentation, by connecting the computer to a suitable video projector, or the like. It is desirable to be able to carry out this operation without the need to first put the computer to sleep, and thereby reduce the time needed to operate within the changed configuration. It is an objective of the present invention, therefore, to expand upon the capabilities of the system of the '529 patent, by providing a display environment in which so-called "hot plugging" of displays is possible, wherein a display becomes immediately available for use as soon as it is plugged into the computer system.

SUMMARY OF THE INVENTION

In accordance with the present invention, the foregoing objective is achieved by utilizing a device manager to automatically recognize and react to changes in the configuration of a display environment, rather than wait for the computer's operating system to proactively determine the status of the display environment, for example upon restarting. When an input/output device is added to or removed from the computer system, an interrupt signal informs the device manager of the fact that a change in configuration has occurred. In response thereto, the device manager determines whether the changed component relates to the computer's display function. For example, it may determine whether an added device is a video card. If so, the device manager makes a call to the computer's display manager, to inform it of the fact that the display configuration has changed. In response to this call, the display manager reconfigures the display space for the computer system, to accommodate an additional frame buffer that is associated with the added component. With this change in the configuration of the display space, the added component becomes immediately available for use.

By means of this approach, the user can add a second monitor or other hardware component to a computer and begin to use the monitor as soon as it has been connected, without the need to reboot the computer or otherwise interrupt its current operating state.

Further features and advantages of the invention are explained in detail hereinafter in the context of specific embodiments that are described with reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an overall display system architecture;

FIG. **2** illustrates an example of a display environment;

FIG. **3** is a block diagram of a display system architecture which includes plural video cards and display devices;

FIG. **4** is a block diagram illustrating the operation of the device manager; and

FIG. **5** is a flowchart illustrating the process by which changes in the configuration of the display environment become immediately available to the user, in accordance with the principles of the invention.

DETAILED DESCRIPTION

The present invention is directed to the display environment of a computer system. A block diagram of the overall architecture for a display environment is illustrated in FIG. **1**. In this figure, hardware components of the computer system are illustrated above a dashed line **10**, and software components are depicted below the line. These software components are stored in a suitable computer-readable medium, such as a magnetic disk, and loaded into the computer's working memory, i.e. RAM, for execution. The system can include display devices **12**, e.g. monitors, LCD screens and/or plasma displays, although actual display devices need not be physically present in order for the principles of the invention to be operative. Each display device is connected to, and controlled by, a video card **14** which operates in accordance with video driver software **16**. Although depicted as being on a separate substrate, such as a printed circuit board, the components of at least one video card could be incorporated with other components on a single substrate, such as the computer's motherboard.

One or more software programs, such as application programs **20**, generate information to be displayed on the display devices. Examples of such information include text, windows and other graphical objects, and control structures such as menus and dialog boxes. This information is presented to the display device through the computer's operating system **18**, which also generates its own information to be presented on the display. The operating system communicates with the display device through an associated display driver **13**, which constitutes a software component that corresponds to the hardware of the display device **12**.

The operating system includes a display manager **22**, which provides communication between each of the software components, and dynamically configures the display devices **12**. The communication between the various software components and the hardware devices takes place via their associated drivers, e.g. the video driver and the display driver. In this regard, many video displays have the capability to provide information regarding their available modes of operation and/or timing specifications. Some displays, so-called "smart displays," are capable of providing information about their modes of operation directly, for example in response to inquiries. For these types of displays, the display manager **22** communicates directly with the display device, by means of the display driver **13**, over a communication channel **24**. This communication channel can be a bus within the computer, a serial line, or any other suitable path for exchanging information between the display manager and the display driver **13** of the display device.

In some cases, the display device may not be able to communicate its capabilities directly. However, through the use of a lookup table or the like, the display driver **13** can obtain information regarding the display's capabilities, and provide them to the display manager.

The display manager also communicates with other parts of the operating system **18** and the other software programs **20** that are running on the computer. For example, in response to operator commands, the operating system can instruct the display manager to add a new device to a list of active displays, or remove a device therefrom. In response thereto, the display manager informs the application programs **20** of the new display configurations, to enable them to update their displayed information accordingly.

In one known implementation for computer systems, the display environment can generally be considered to be defined by a global coordinate space **30**, as depicted in FIG. **2**. Objects and other information to be displayed can be positioned anywhere within this space, as determined by the user and/or the software program that generates the information. A reference point in this space, e.g. its origin or 0,0 coordinate point, is usually established with reference to some object that is always present in the display. For instance, most graphical user interfaces include some type of menu bar or other structure which enables the user to access basic commands to control the computer. The device which displays this menu bar is known as the main display device. If the computer system contains multiple display devices, only one of the devices is designated as the main device, even if multiple devices contain the menu bar. The origin of the coordinate display space is typically established with reference to the menu bar. For example, as illustrated in FIG. **2**, the 0,0 point **38** in the coordinate space can coincide with the top left corner of a menu bar **36**. The positions of all objects and other information to be displayed in the display space **30** are defined by their coordinates within this space. The operating system receives this coordinate information, for example from the software programs **20** which generate the information, and provides it to the display driver to cause the information to appear at the appropriate place on the screen of the display device located at the corresponding position in the display space.

In the example illustrated in FIG. **2**, the display environment consists of two display devices, **32** and **34**, within the global display space **30**. A menu bar **36** is displayed at the top of the screen for the device **32**, which is therefore the main display device. Accordingly, the origin **38** of the display space coincides with the top left corner of the device **32**. As illustrated in FIG. **2**, the user has caused some objects, e.g. windows **40**, to be displayed on the device **32**, and another object **42** to be displayed on the device **34**.

FIG. **3** illustrates the configuration of the computer system for the particular example illustrated in FIG. **2**, which includes two display devices.

Each display device is connected to an associated video card, which includes a corresponding video driver. For the sake of simplicity in FIG. **3**, the display drivers are not separately illustrated, but are assumed to be present within the system, in a manner analogous to the arrangement shown in FIG. **1**. The embodiment of FIG. **3** includes two video cards **14** and **14'**, respectively associated with the two video display devices **12** and **12'**. Each of the video cards communicates with the display manager **22**, by means of its associated video driver **16** and **16'**.

Among other components, each video card includes a frame buffer, e.g. random access memory, which stores the data for the image that is displayed on its associated display device **12**. In essence, the display manager **22** assigns the frame buffer to a corresponding portion of the global coordinate space **30**. In the example of FIG. **2**, the two frame buffers are assigned to mutually exclusive portions of the global space. However, some or all of the portion assigned to one of the frame buffers could overlap with the area assigned to the other frame buffer. In this case, the same image, or portion of an image, appears on both display devices.

At any given time, there could be only one video card connected to the computer, or both cards could be connected. Furthermore, in the case of a network server or the like, it is possible that no video card would be present over certain periods of time. Even when both cards are present, only one of them may have a monitor or other display device connected to it at any particular point in time.

In the case of a conventional desktop or notebook computer system, one of the video cards might be incorporated within the structure of the computer system, and may not be designed to be repeatedly inserted and removed by the user. Additional video cards, however, might be capable of being easily inserted into and removed from the computer system. For example, the video card might be implemented in a removable card that conforms to the PC Card standard. This standard defines the form factor for relatively small, credit-card shaped I/O devices, which are designed to be easily inserted into and removed from computer housings, to provide a computer with different capabilities. Included among the types of I/O devices that can be embodied in such a card are modems, facsimile devices, network interface cards, wireless communications devices and hard disk drives.

Devices of this type which conform to this standard, commonly known as PC Cards, are designed to be readily inserted and removed from the computer housing. The detection of the presence of such devices, as well as their removal from the system, is handled by a portion of the computer's operating system that is referred to herein as a device manager. Referring to FIG. **4**, when a PC Card **44** is inserted into the housing of the computer, it actuates a switch **46**, or equivalent sensor device, which sends an interrupt signal IRQ to the device manager **48**. In response to this interrupt, the device manager determines the type of device which has been inserted, and informs the operating system **18**. In a similar manner, whenever the PC Card is removed from the computer housing, an interrupt is also sent to the device manager, which in turn notifies the operating system that the device is no longer available.

Another type of change which can be made to the display configuration of the computer is the addition or removal of a display device. In the example of FIG. **3**, for instance, either one of the display devices **12** or **12'** could be disconnected from its associated video card **14** or **14'**. Furthermore, if only one display device is present, it could be disconnected from one of the video cards **14** and connected to the other video card **14'**. Whenever a change of this nature occurs, an interrupt is sent to the device manager **48**. For instance, the interrupt could be generated by the video card, upon detecting that a display device has been physically connected to or disconnected from it. Alternatively, the interrupt could be provided by a bus that is capable of detecting such a change.

The addition or removal of other types of hardware can also result in a change in the display configuration of the computer system. For example, a graphics accelerator card can be added to the system by means of a PC Card slot. Again, upon the addition or removal of such a device, an interrupt signal IRQ is sent to the device manager.

The present invention is particularly directed to the situation in which the device that is added to or removed from the computer system is related to the display function. In the past, it was necessary to reboot the computer system in order for a change in video hardware to become effective. More particularly, unless a reboot occurred, the operating system was not prompted to undertake any action which would cause it to detect the presence of a new driver, resulting from the addition of an associated hardware device. Hence, it was necessary for the user to interrupt the operating state of the computer in order to utilize the additional functionality provided by a newly added hardware. Once the operating system became aware of the presence of the new driver, it could notify the display manager to incorporate the presence of the new frame buffer.

In accordance with the present invention, however, the display system can be immediately responsive to the addition or removal of video hardware, so that the capabilities of a revised configuration can be employed without the need to change the operating state of the computer. This functionality is accomplished by, in effect, bypassing the need to have the operating system actively determine the addition or removal of a display device. Rather, the notification of a change in the configuration of the display environment is provided directly to the display manager, so that it can directly account for the presence or absence of a particular video device.

To this end, whenever a PC Card is added to or removed from the computer system, the device manager determines whether the card relates to a display function. A similar determination is made whenever an interrupt is generated that indicates some other type of hardware has been added or removed, e.g. a display monitor. In addition to, or in lieu of interrupts, other approaches can be employed to determine when a device has been added or removed. For example, the operating system can periodically poll all of the computer system's I/O ports, to determine which devices are present and which ones might have been removed.

Referring to FIG. **5**, upon receipt of an indication that there has been a change in configuration, the device manager first determines at step **50** whether a device has been added or removed. If a device has been added to the system, the device manager communicates with the device to determine its type, at step **52**, and stores data in a register regarding the identity and type of the device. If the device responds with an indication that it is a video device, the device manager issues a call to the display manager **22**, at step **54**. Appropriate parameters can be included with the call, to indicate the type of device, the size of its frame buffer (if applicable), its resolution, and the like.

In response to this information, the display manager carries out a number of operations, depicted in Steps **56**-**64**. First, it registers the added hardware as a new device, along with the location of its associated drivers in memory, at step **56**. In some cases, the driver may already be present in memory, but in an inactive state because the device was not connected to the system at the time of initial boot. In this case, the display manager switches the driver to an active state.

After registering the device, the display manager matches each display device with an available frame buffer at step **58**. If a new video card is inserted, for example, the display manager assigns a portion of the global coordinate space **30** to the frame buffer in the video card. If a display device is connected to that video card, the display manager assigns that device to the frame buffer for that card, so that the proper data is displayed on the device. If a display device is disconnected from one video card and connected to a different video card, the display manager moves objects within the global space **30** so that they are presented to the appropriate frame buffer for the display device. For example, the display manager can move user interface control objects which are specific to that display, such as brightness and contrast controls, to the frame buffer associated with that display. Similarly, if the display has other attributes associated with it, such as a certain name or designation, the display manager ensures that they are directed to the proper frame buffer.

Once the display devices and frame buffers are matched up, the display manager consults a preferences file which indicates whether that device was connected to the system at some previous time. This file is preferably stored in permanent memory, such as a hard disk, and updated each time a video device is added to or removed from the computer system to change its configuration. The file stores the state of the display just prior to the change. For instance, it may store the location of the menu bar and all other objects on the desktop of the user interface, along with each device that made up the configuration. It can store relevant operating parameters for the device as well, such as timing values, color depth, and the like. When a new device is added, the display manager determines whether the changed configuration had existed previously, and if so it attempts to restore the display environment to the state that existed the last time that configuration was present, at step **60**. For example, it might move the menu bar and certain icons to the screen of a newly added display device, if they were previously located on that device before it was disconnected, or set the timing of an added frame buffer.

In addition to these actions, at step **62** the display manager can inform currently executing programs of the presence of the new video device, as described in detail in U.S. Pat. No. 5,682,529, the disclosure of which is incorporated herein by reference. Furthermore, other software which is not executing, but which relates to the new device, can be activated. For instance, if a graphics accelerator card is inserted in a PC Card slot, the software associated with that card can be automatically launched.

The display manager also functions at step **64** to rebuild or reconfigure resources that might be employed by the new device. For example, in some display modes a table lookup operation is carried out to determine the colors which are displayed on the monitor. Different application programs may utilize different tables for this purpose. Typically, the window which is in the foreground of a display controls the particular table that is used. Therefore, when windows are moved onto a new display device, for instance in accordance with the preferences file, the display manager rebuilds the color lookup table stored in the frame buffer for that device so that it corresponds to the appropriate window.

If the device manager determines at step **50** that a device has been removed from the system, rather than added, it determines at step **66** whether the removed hardware was a video device, for example by reference to previously stored information which indicated the type of card inserted into each PC Card slot, or the like. If a determination is made that the removed device was part of the video subsystem, the device manager sends a call to the display manager **22**, at step **68**, to inform it of this fact. In response thereto, the display manager updates the preferences file at step **70**, to record the relevant parameters that pertain to the most recent configuration. In addition, the display manager can attempt to restore preferences that pertain to the new condition. Thus, for example, if the computer system had two display devices and one was removed, the preferences file would be updated to store the relevant data for the two-monitor configuration, and then searched to determine whether it contains data for a one-monitor configuration. The display manager then rebuilds the display configuration, at step **72**.

For example, if the removed device is a video card, the display manager deletes the assignment of a portion of the display space to the now-removed frame buffer. In concert with this action, the display manager can also function to move objects that were previously displayed on the removed display device to an area associated with a remaining display device, as described in greater detail in U.S. Pat. No. 5,682,529. If the removed device had executing software associated with it, the display manager can cause the software to shut down.

A special case can occur if all frame buffers, e.g. video cards, are removed from the system. Since the frame buffer is a memory-mapped I/O device, a program or other device could attempt to address memory that is no longer present. Typically, when such a situation occurs, the operating system recognizes it as an error condition, and shuts down the process which attempted to address the non-existent memory. However, in the system of the present invention, if an attempt is made to address memory in a video device, e.g. a frame buffer, the resulting error condition is treated as an interrupt. This interrupt is relayed to the display manager to cause it to reconfigure the display environment, and delete the assignment of display space to the frame buffer which is no longer present.

Another action that can occur upon removal of a display device is to change the operating mode of its display driver. More particularly, some display devices can operate in one mode in which adjustment of control buttons, such as brightness and contrast controls, is accompanied by user interface feedback, e.g. variation of a sliding scale on the display. This operating mode requires communication with the computer's operating system. In another operating mode, referred to as a remote mode, no such user feedback is provided. In the system of the present invention, when a display device is to be removed, the display manager instructs its display driver to switch to the remote mode, so that no attempt is made to affect the user interface while the device is not present.

From the foregoing, it can be seen that the present invention provides a hot-plugging capability for video devices, that enables users to immediately take advantage of changes in the display configuration of a computer system, such as the addition of a new video card. This functionality is attained by providing notification of the changed configuration directly to the display manager, rather than waiting for an action that prompts the operating system to review the current configuration, such as rebooting the computer.

It will be appreciated by those of ordinary skill in the art that the present invention can be embodied in other specific forms without departing from the spirit or essential characteristics thereof. For example, while one embodiment of the invention has been described in the context of the insertion and removal of video cards that are embodied in PC Cards, it will be appreciated that the principles which underlie the invention are not limited to this particular implementation. Rather, any other suitable mechanism which accommodates the addition and removal of a video device can benefit from the features of the present invention. The presently disclosed embodiments are therefore considered in all respects to be illustrative, and not restrictive. The scope of the invention is indicated by the appended claims, rather than the foregoing description, and all changes that come within the meaning and range of equivalents thereof are intended to be embraced therein.

What is claimed is:

1. A method for reconfiguring a computer system to accommodate changes in a display environment, comprising the steps of:

detecting the addition or removal of a display device in the computer system;

providing a notification to a component of an operating system executing on said computer system that a video device has been added or removed, in response to said detection; and

modifying the allocation of display space to display devices via said operating system component, in response to said notification and in accordance with the addition or removal of a video device.

2. The method of claim **1** wherein the video device comprises a video card that includes a frame buffer, and said modifying step includes assigning a portion of the display space to the frame buffer of an added video card, or deleting the assignment of a portion of the display space to a removed video card.

3. The method of claim **1** wherein said operating system component carries out the further step of storing a preferences file that identifies the status of displayed objects prior to a change in the configuration of a computer.

4. The method of claim **3** wherein, upon detection of the addition of a video device, said operating system component repositions objects in said display space, in accordance with a status stored in said preferences file.

5. The method of claim **3**, wherein said preferences file stores the video devices which make up the configuration of the computer, and the locations of objects displayed on said video devices.

6. The method of claim **5**, wherein said preferences file also stores operating parameters for said devices.

7. The method of claim **3** wherein, in response to said notification of a removed video device, said operating system component carries out the further step of searching the preferences file to retrieve a status of displayed objects corresponding to when the video device was previously removed.

8. The method of claim **1** wherein said operating system component carries out the step of assigning a respective frame buffer, which corresponds to an allocated portion of the display space, to a corresponding display device.

9. The method of claim **1** wherein, upon detection of the addition of a video device, said operating system component causes a software program associated with the added device to be launched.

10. The method of claim **1** wherein said operating system component further carries out the step of reconfiguring a computer resource to correspond to the status of objects located in the display space.

11. The method of claim **10** wherein said computer resource is a color look-up table.

12. The method of claim **1**, further including the step of recognizing an error condition resulting from an attempt to address a frame buffer that has been removed, providing a notification to said operating system component in response to said error condition, and deleting an allocation of display space to the removed frame buffer.

13. The method of claim **1**, wherein said operating system component carries out the further steps of:

registering an added video device as a new video device in response to said notification of an added video device;

determining a location in a memory of a video driver associated with the added video device; and

storing the location of the video driver associated with the added video device in the memory.

14. The method of claim **13**, wherein said operating system component carries out the further step of activating the video driver associated with the added video device when the video driver is present in the memory and inactive.

15. The method of claim **14**, wherein said operating system component carries out the further steps of:

storing a preferences file that identifies the status of displayed objects prior to a change in the configuration of the computer system;

accessing the preferences file to determine whether the added video device was previously connected to a frame buffer associated with the computer system; and

restoring a display environment to the status identified in the preferences file upon determining that the added video device was previously connected to the frame buffer associated with the computer system.

16. The method of claim of claim **1**, wherein said operating system component carries out the further step of notifying at least one software program executed by the operating system of a change in the configuration of the computer system in response to said notification that a video device has been added or removed.

17. The method of claim **1** wherein, in response to said notification of a removed video device, said operating system component carries out the further step of shutting down at least one application program associated with the removed video device that was executing on the operating system.

18. The method of claim **1** wherein, in response to said notification that a video device is removed, said operating system component carries out the further step of deleting an assignment of display space to a frame buffer associated with the removed video device.

19. A system which provides hot-plugging capabilities for display devices, comprising:

a video device including a frame buffer for storing data that defines an image to be displayed on an associated display device;

a first operating system component which defines a display space and assigns a portion of said display space to said frame buffer, and which provides data for images to be displayed to said frame buffer; and

a second operating system component which detects the addition or removal of a display device in a computer system, and provides a notification of such addition or removal to the first operating system component in response to said detection, to cause the assignment of a portion of the display space to be modified in accordance with a detected addition or removal.

20. The system of claim **19**, wherein said first operating system component launches a software program associated with the display device in response to notification that the display device has been added.

21. The system of claim **19**, further including a preferences file stored in memory which indicates the status of objects being displayed when a display device is removed.

22. The system of claim **21** wherein, in response to said detection of a removal of a display device, said first operating system component is further configured for searching the preferences file to retrieve a status of displayed objects corresponding to when the display device was previously removed.

23. The system of claim **19**, wherein said first operating system component is further configured for:

registering an added display device as a new display device in response to the notification of an added display device from said second operating system component;

determining a location in a memory of a display driver associated with the added display device; and

storing the location of the display driver associated with the added display device in the memory.

24. The system of claim **23**, wherein said first operating system component is further configured for activating the display driver associated with the added display device when the display driver is present in the memory and inactive.

25. The system of claim **19**, wherein said first operating system component is further configured for:

storing a preferences file that identifies the status of displayed objects prior to a change in the configuration of the computer system;

accessing the preferences file to determine whether an added display device was previously connected to said frame buffer; and

restoring a display environment to the status identified in the preferences file upon determining that the added display device was previously connected to said frame buffer.

26. The system of claim **19** wherein, in response to said detection of a removal of a display device, said first operating system component is further configured for shutting down at least one application program associated with the removed display device.

27. The system of claim **19** wherein, in response to said detection of a removal of a display device, said first operating system component is further configured for deleting an assignment of display space to a frame buffer associated with the removed display device.

28. The system of claim **19**, wherein said first operating system component is further configured for reconfiguring at least one computer resource used by an added or removed display device to correspond to a status of objects located in the display space after the change in the configuration of the computer system, in response to said notification of an added or removed display device.

29. The system of claim **19**, further comprising a detection unit operable to detect when an input/output device is added to or removed from the computer system, and provide an indication to said second operating system component of a detected addition or removal of an input/output device in the computer system,

wherein said second operating system component is further configured for receiving the indication of the addition or removal of the input/output device, and determining whether the added or removed input/output device is a display device.

30. The system of claim **29**, wherein said detection unit is a video card operable to detect that an input/output device is connected thereto or disconnected therefrom.

31. The system of claim **29**, wherein said second operating system component is further configured for:

communicating with an added input/output device to detect at least one of a type and identity of the added input/output device;

determining whether the added input/output device is a display device according to the detected at least one of the type and identity of the added input/output device;

storing the detected at least one of the type and identify of the added input/output device in a memory; and

accessing the memory each time an input/output device is added to determine whether the added input/output device is a display device.

32. The system of claim **31**, wherein said second operating system component is further configured for determining whether an input/output device removed from the computer system is a display device by referencing the at least one of the type and identify of the input/output device stored in the memory.

33. A computer-readable medium having a device manager program and a display manager program stored thereon, wherein said device manager program causes a computer system to perform the steps of:

detecting the addition or removal of a display device in the computer system, and

providing a notification to said display manager program when a display device is added or removed; and

wherein said display manager program causes the computer system to perform the step of:

modifying the allocation of display space to display devices in response to said notification from said device manager program.

34. The computer-readable medium of claim **33**, wherein said display manager program causes the computer system to perform the further steps of storing a preferences file relating to the status of objects appearing on a display device, and restoring objects to the status stored in the preferences file when a display device is added.

35. The computer-readable medium of claim **34** wherein, in response to said notification of a removal of a display device, said display manger program causes the computer system to perform the further step of searching the preferences file to retrieve a status of displayed objects corresponding to when the display device was previously removed.

36. The computer-readable medium of claim **33**, wherein said display manager program causes the computer system to perform the further step of assigning a respective frame buffer to a display device in response to said notification of an added display device, or deleting the assignment of a respective frame buffer from said display device in response to said notification of a removed display device.

37. The computer-readable medium of claim **33**, wherein said display manager program causes the computer system to perform the further step of launching a software program in response to said notification.

38. The computer-readable medium of claim **33**, wherein said display manager program causes the computer system to perform the further step of reconfiguring at least one computer resource in accordance with the modification of the display space allocation.

39. The computer-readable medium of claim **38**, wherein said computer resource is a color look-up table.

40. The computer-readable medium of claim **33**, wherein said display manager program causes the computer system to perform the further steps of:

registering an added display device as a new display device in response to the notification of an added display device from said device manager program;

determining a location in a memory of a display driver associated with the added display device; and

storing the location of the display driver associated with the added display device in the memory.

41. The computer-readable medium of claim **40**, wherein said display manager program causes the computer system to perform the further step of activating the display driver associated with the added display device when the display driver is present in the memory and inactive.

42. The computer-readable medium of claim **33** wherein, in response to said detection of a removal of a display device, said display manager program causes the computer system to perform the further step of deleting an assignment of display space to a frame buffer associated with the removed display device.

43. The computer-readable medium of claim **33**, further having a detection program stored thereon,

wherein said detection program causes the computer system to perform the step of detecting when an input/output device is added to or removed from the computer system, and

wherein said device manager program causes the computer system to perform the further step of determining whether an input/output device that was detected to be added or removed is a display device by detecting at least one of a type and identity of the added or removed input/output device.

* * * * *