

(19) 日本国特許庁 (JP)

(12) 公表特許公報 (A)

(11) 特許出願公表番号

特表2008-538043

(P2008-538043A)

(43) 公表日 平成20年10月2日 (2008.10.2)

(51) Int.Cl.		F I		テーマコード (参考)	
G 0 6 F	9/54	(2006.01)	G 0 6 F	9/46	4 8 0 C
G 0 6 F	9/48	(2006.01)	G 0 6 F	9/46	4 5 2 B

審査請求 有 予備審査請求 未請求 (全 20 頁)

(21) 出願番号 特願2008-506796 (P2008-506796) (86) (22) 出願日 平成18年4月12日 (2006.4.12) (85) 翻訳文提出日 平成19年12月13日 (2007.12.13) (86) 国際出願番号 PCT/US2006/014306 (87) 国際公開番号 W02006/110914 (87) 国際公開日 平成18年10月19日 (2006.10.19) (31) 優先権主張番号 11/105, 913 (32) 優先日 平成17年4月13日 (2005.4.13) (33) 優先権主張国 米国 (US)	(71) 出願人 595020643 クォアルコム・インコーポレイテッド QUALCOMM INCORPORATED アメリカ合衆国、カリフォルニア州 92 121-1714、サン・ディエゴ、モア ハウス・ドライブ 5775 (74) 代理人 100058479 弁理士 鈴江 武彦 (74) 代理人 100091351 弁理士 河野 哲 (74) 代理人 100088683 弁理士 中村 誠 (74) 代理人 100108855 弁理士 蔵田 昌俊
---	---

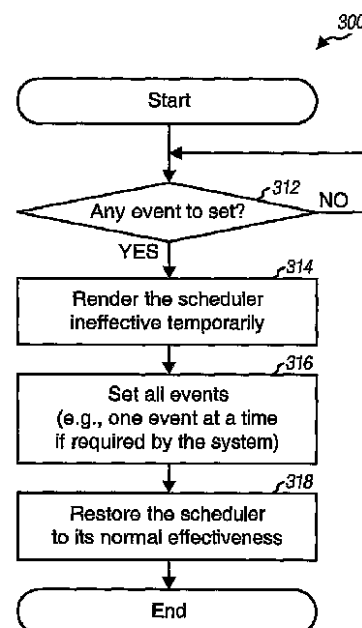
最終頁に続く

(54) 【発明の名称】 マルチスレッドシステムの中でイベントを設定するための技術

(57) 【要約】

【解決手段】 マルチスレッドシステムの中で、スレッドの中間スケジューリングを伴わずに1つまたは複数のイベントを設定するために、スケジューラは最初に本質的に無効にされる。次いで、例えば設定されるべき各イベントにつき1回、単一のイベントを設定するために関数を繰り返し呼び出すことによってイベントが設定される。次いで、スケジューラはその通常の効果に回復される。スケジューラは、処理されている現在のスレッドの優先権を最も高い可能な優先レベルに人為的に設定することによって、無効にされてもよい。このことは、イベントが設定される間に、現在のスレッドが別のスレッドによって割り込まれることを回避する。すべてのイベントが設定された後、現在のスレッドの優先権は本来の優先レベルに回復され、このことが、その後スケジューラをその通常の効果に回復する。

【選択図】 図3



【特許請求の範囲】**【請求項 1】**

現在のスレッドの優先権を高い優先レベルに設定するためのコードと、
少なくとも 1 つのイベントを設定するためのコードと、
前記現在のスレッドの前記優先権を本来の優先レベルに回復するためのコードとを記憶するためのプロセッサ可読媒体。

【請求項 2】

前記現在のスレッドの前記優先権を前記高い優先レベルに設定する前に、前記本来の優先レベルを保存するためのコードをさらに記憶するための請求項 1 に記載のプロセッサ可読媒体。

10

【請求項 3】

少なくとも 1 つのイベントを設定するための前記コードが、
1 回につき 1 つのイベントで、少なくとも 1 つのイベントを設定するためのコードを含む請求項 1 に記載のプロセッサ可読媒体。

【請求項 4】

少なくとも 1 つのイベントを設定するための前記コードが、
前記少なくとも 1 つのイベントの各々につき 1 回、単一のイベントを設定するために関数呼び出すためのコードを含む請求項 1 に記載のプロセッサ可読媒体。

【請求項 5】

少なくとも 1 つのイベントを設定するための前記コードが、
前記少なくとも 1 つのイベントの各々を個々に設定するために、オペレーティングシステムの中で関数呼び出すためのコードを含む請求項 1 に記載のプロセッサ可読媒体。

20

【請求項 6】

前記現在のスレッドの前記優先権を設定するための前記コードが、
前記現在のスレッドの前記優先権を最も高い可能な優先レベルに設定するためのコードを含む請求項 1 に記載のプロセッサ可読媒体。

【請求項 7】

前記現在のスレッドの前記優先権を設定するための前記コード、前記少なくとも 1 つのイベントを設定するための前記コード、および前記現在のスレッドの前記優先権を回復するための前記コードが、1 つまたは複数のイベントを設定するためにスレッドによって呼び出されることが可能な関数の一部である請求項 1 に記載のプロセッサ可読媒体。

30

【請求項 8】

前記関数が、オペレーティングシステムのためのカーネルの上の抽象層に存在する請求項 7 に記載のプロセッサ可読媒体。

【請求項 9】

現在のタスクの優先権を高い優先レベルに設定することと、
少なくとも 1 つのイベントを設定することと、
前記現在のタスクの前記優先権を本来の優先レベルに回復することを含むイベントを設定するための方法。

【請求項 10】

前記現在のタスクの前記優先権を前記高い優先レベルに設定する前に、前記本来の優先レベルを保存することをさらに含む請求項 9 に記載の方法。

40

【請求項 11】

前記少なくとも 1 つのイベントを設定することが、1 回につき 1 つのイベントで、前記少なくとも 1 つのイベントを設定することを含む請求項 9 に記載の方法。

【請求項 12】

前記現在のタスクの前記優先権を前記高い優先レベルに設定することが、前記現在のタスクの前記優先権を最も高い可能な優先レベルに設定することを含む請求項 9 に記載の方法。

【請求項 13】

50

現在のタスクの優先権を高い優先レベルに設定するための手段と、
少なくとも1つのイベントを設定するための手段と、
前記現在のタスクの前記優先権を本来の優先レベルに回復するための手段とを含む装置。

【請求項14】

前記現在のスレッドの前記優先権を前記高い優先レベルに設定する前に、前記本来の優先レベルを保存するための手段をさらに含む請求項13に記載の装置。

【請求項15】

もしあれば、設定されている前記少なくとも1つのイベントに基づいて、実行のために準備ができているタスクを識別するための手段と、

実行のために準備ができている前記タスクの中で、最も高い優先権を有するタスクをスケジュールするための手段とをさらに含む請求項13に記載の装置。

【請求項16】

スケジューラを一時的に無効にすることと、

少なくとも1つのイベントを設定することと、

前記スケジューラを通常の効果に回復することを含むイベントを設定する方法。

【請求項17】

前記スケジューラを一時的に無効にすることが、現在のタスクの優先権を高い優先レベルに設定することを含む請求項16に記載の方法。

【請求項18】

前記スケジューラを通常の効果に回復することが、前記現在のタスクの前記優先権を本来の優先レベルに回復することを含む請求項17に記載の方法。

【請求項19】

前記少なくとも1つのイベントを設定することが、1回につき1つのイベントで、前記少なくとも1つのイベントを設定することを含む請求項16に記載の方法。

【請求項20】

タスクをスケジュールするために動作可能なスケジューラと、

前記スケジューラを一時的に無効にし、少なくとも1つのイベントを設定するため、かつ前記スケジューラを通常の効果に回復するために動作可能な処理モジュールとを含み、前記スケジューラが、設定される前記少なくとも1つのイベントにしたがって前記タスクをスケジュールするために動作可能である装置。

【請求項21】

前記処理モジュールが、現在のタスクの優先権を高い優先レベルに設定して、前記スケジューラを無効にするために動作可能である請求項20に記載の装置。

【請求項22】

前記処理モジュールが、前記現在のタスクの前記優先権を本来の優先レベルに回復して、前記スケジューラを通常の効果に回復するために動作可能である請求項21に記載の装置。

【請求項23】

前記処理モジュールが、1回につき1つのイベントで、前記少なくとも1つのイベントを設定するために動作可能である請求項20に記載の装置。

【請求項24】

前記スケジューラが、もしあれば、設定される前記少なくとも1つのイベントに基づいて、実行のために準備ができているタスクを識別するため、かつ実行のために準備ができていない前記タスクの中で最も高い優先権を有するタスクをスケジュールするために動作可能である請求項20に記載の装置。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は一般にコンピュータ処理に関し、より詳細には、マルチスレッドシステムの中

10

20

30

40

50

でイベントを設定するための技術に関する。

【背景技術】

【0002】

オペレーティングシステムは、1つの一般的な種類のマルチスレッドシステムである。オペレーティングシステムは様々なシステムリソース（例えばハードウェア装置、記憶装置、インタフェースなど）を管理し、システムリソースにアクセスするためにアプリケーションがオペレーティングシステム上でランすることを可能にするサービスを提供する。典型的にオペレーティングシステムは、適切に起動、スケジュールおよび処理されるスレッドを介して、これらのサービスを提供する。スレッドは、実行されるべきタスクとして考えられてもよい。

10

【0003】

リアルタイムオペレーティングシステムはオペレーティングシステムの1つのクラスであり、一般に無線装置（例えば携帯電話）、携帯情報端末（PDA）などの多くの電子装置のために使用される。リアルタイムオペレーティングシステムの主要な属性は、割り込み優先権に基づくスケジューリングである。この種類のスケジューリングは、実行のために準備ができていてすべてのスレッドの中で、最も高い優先権を備えたスレッドが最初に処理されるように、スレッドをスケジュールしようとする。各スレッドは特定の優先権、および場合によっては、スレッドが処理されることが可能となる前に設定されなければならないイベントに関連している。イベントが設定されるたびに、オペレーティングシステムは、どのスレッドが実行のために準備ができていて（すなわち任意のイベント上で待機中ではないか、または設定済みのイベント上で待機中であるか）を判定し、準備ができていて最も高い優先権のスレッドをスケジュールする。現在処理されているスレッドは、より高い優先権を有し、実行のために準備ができていて別のスレッドによって割り込まれてもよい。

20

【0004】

リアルタイムオペレーティングシステムによって、アクティブスレッドは1回につき単一のイベントを設定することができてよい。しかしながら、アクティブスレッドは設定される複数のイベントを有している場合があり、以下で説明されるように、アクティブスレッドがこれらのイベントを設定する順序が、スレッドがスケジュールされる順序に影響を及ぼす場合がある。イベントが設定される順序のせいで、スレッドが望ましくない順序でスケジュールされる場合、システムの処理能力は低下する場合がある。

30

【発明の開示】

【0005】

したがって当技術分野では、マルチスレッドシステムの中でイベントを設定するための技術が必要とされている。

【0006】

本明細書では、マルチスレッドシステム（例えばリアルタイムオペレーティングシステム）の中で、スケジューラによるスレッドの中間スケジューリングを伴わずにイベントを設定するための技術が説明される。マルチスレッドシステムの中で1つまたは複数のイベントを設定するために、最初にスケジューラは本質的に無効にされる。次いで、例えば単一のイベントを設定するために関数を繰り返し呼び出すことによって、イベントが設定される。この関数は、設定されるべき各イベントについて1回呼び出されてもよい。スケジューラはその後、その通常の効果に回復される。

40

【0007】

1つの実施形態で、処理されている現在のスレッドの優先権を、例えばシステム中で最も高い可能な優先レベルなどの高い優先レベルに人為的に設定することによって、スケジューラは無効にされる。このことは、イベントが設定される時間に、別のスレッドがスケジュールされること、および現在のスレッドが別のスレッドによって割り込まれることを回避する。すべてのイベントが設定された後、現在のスレッドの優先権は本来の優先レベルに回復される。ここで現在のスレッドはより高い優先権を備えた別のスレッドによって

50

割り込まれることが可能であるので、このことはその後スケジューラをその通常の効果に回復する。

【0008】

本明細書で説明されるイベント設定技術はソフトウェアシステム、ハードウェアシステム、またはハードウェアとソフトウェアの両方を備えたシステムのために使用されてもよい。例えば、中間スケジューリングを伴わずにイベントを設定するためのコードは、スレッドによって呼び出されてもよいソフトウェア関数として実装されてもよい。この関数はその後、1回につき単一のイベントを設定するために、（例えばオペレーティングシステムによって提供される）別のソフトウェア関数を呼び出してもよい。

【0009】

本発明の様々な態様および実施形態は、以下でより詳細に説明される。

【0010】

本発明の機能および性質は、全体を通じて同じ参照符号が対応して識別を行う図面と合わせて考慮する場合に、以下で言及される詳細な説明から、より明らかとなろう。

【発明を実施するための最良の形態】

【0011】

本明細書では「例示的」という語は、「例、実例、または具体例としての役割を果たす」ということを意味するために使用される。本明細書で「例示的」として説明される任意の実施形態または設計は、必ずしもその他の実施形態もしくは設計よりも好ましいか、または有利なものとして解釈されるわけではない。

【0012】

図1は、本明細書で説明されるイベント設定技術が用いられてもよい例示的なマルチスレッドシステム100の図を示す。スレッド110x、110yおよび110zはアプリケーション（図1に不図示）によって起動されてもよく、スケジュールかつ実行されるべきものである。スレッドとは、これもまたスレッドであってよい主体によって実行されるべきタスクのことである。スレッドは、（1）タスクのために実行されるべきソフトウェアコード、および（2）例えばレジスタ、スタック、変数などのタスクについての状態情報に関連していてもよい。スレッドは、任意の所与の時点での複数の可能な状態（例えば「保留」、「準備済み」および「アクティブ」）のうちの1つであってよい。本明細書で使用される際、「保留」スレッドとは、スケジュールされることが可能になる前のイベントで待機中のスレッドのことであり、「準備済み」スレッドとは、実行のために準備ができていて（すなわち任意のイベント上で待機中ではない）、スケジュールされてもよいスレッドのことであり、「アクティブ」スレッドとは、スケジュールが完了していて、現在ラン中のスレッドのことである。

【0013】

各スレッド110は、スレッドの重要性を示す優先レベル／値と関連している。各スレッドの優先権は、例えばスレッドを起動するアプリケーションに割り当てられた優先権によって決定されてもよい。優先値の値域はシステムのために使用されてもよく、より高い優先値はより大きな重要度に対応してもよい。次いで、各スレッドについての優先値は、そのスレッドの相対的重要度に基づいて可能な優先値の値域から選択される。各スレッド110はさらに、ゼロ、1つまたは複数の保留イベントに関連している。イベントは、スレッドの状態または状況を示している。オペレーティングシステムのコンテキストでは、イベントは信号、フラグ、メッセージ、またはその他の用語とも呼ばれる。保留イベントとは、スレッドがスケジュールされることが可能になる前に、少なくとも1つのスレッドが待機しているイベントのことである。

【0014】

スケジューラ120はスレッド110を受信し、実行のためにこれらのスレッドをスケジュールする。典型的に、任意の所与の時点で1つだけのスレッド110がスケジュールされ、スケジュールされたスレッドはアクティブスレッドとなり、実行する。スケジューラ120は、（1）これらのスレッドについての優先値および保留イベント、ならびに（

10

20

30

40

50

2) 現在ラン中であるアクティブスレッド130により設定されたイベントに基づいて、スレッド110をスケジュールする。スケジューラ120はさらに、任意の所与の時点で最も高い優先権を備え、実行のための準備ができていないスレッドがランするように、スレッドをスケジュールする。

【0015】

アクティブスレッド130は、スケジューラ120によってスケジュールされ、現在ラン中であるスレッドである。アクティブスレッド130はまた、アクティブスレッドのために適用可能であり、別の保留スレッド110をスケジュールするためにスケジューラ120によって使用されてもよいイベントを設定する。アクティブスレッド130は、関連した状態または状況が生じるたびに、各イベントを設定してもよい。

10

【0016】

マルチスレッドシステム100はソフトウェアシステム、ハードウェアシステム、またはソフトウェアとハードウェアの両方を備えたシステムであってよい。明快であるために、以下の説明のうちのある部分は、スケジューラ120がオペレーティングシステム（例えばリアルタイムオペレーティングシステム）の一部である、特定のマルチスレッドシステムについてのものである。

【0017】

オペレーティングシステムによって、スレッド（例えばアクティブスレッド130）は1回につき単一のイベントを設定することができる。イベントが設定されるたびに、オペレーティングシステムは、ちょうど設定されたイベントに適した動作を実行する。例えばオペレーティングシステムは、アクティブスレッド130によって設定されたイベントに基づいて、保留スレッドをスケジュールしてもよい。イベントが設定されるたびに、オペレーティングシステムは、ちょうど設定されたイベント上で待機中の保留スレッドおよび現在処理されているスレッドを含む、実行のために準備ができていないすべてのスレッドを識別する。次いでオペレーティングシステムは、最も高い優先権を備えた準備済みスレッドを実行のためにスケジュールする。

20

【0018】

アクティブスレッド130はイベントが生じるときにそれらを設定してもよく、オペレーティングシステムによって必要とされる場合には、1回につき単一のイベントを設定してもよい。一般にアクティブスレッド130は、アクティブスレッドの状態に応じて、現在のスレッドの処理の間に任意の数のイベントおよび任意の特定のイベントを設定してもよい。アクティブスレッド130は、(1) スケジュールされることを待機している保留スレッド、または(2) これらの保留スレッドが待機しているイベントを認識しなくてもよい。結果的に、アクティブスレッド130は、保留スレッドおよびそれらの保留イベントに基づいて決定された優先順位で、イベントを設定しようとするのではない。

30

【0019】

アクティブスレッド130は、現在のスレッドの処理を完了する間に、またはそれを完了した上で、複数のイベントを設定してもよい。さらに、スケジューラ120はイベントが設定されるたびに、スレッド110をスケジュールしてもよい。この場合スレッド110は、以下で示されるように、イベントが設定される順序に応じて、異なる順序でスケジュールされてもよい。

40

【0020】

図2Aは、2つのイベント上で待機中の2つの保留スレッドのためのスケジューリング例を示す。この例で、はじめにアクティブスレッド130は、低い優先権10を有するスレッドWである。スレッドXは高い優先権30を有し、イベントA上で待機中であり、スレッドYは中間の優先権20を有し、イベントB上で待機中である。スレッドXとYとは両方とも、アクティブスレッド130がメモリへの書き込みを完了するのを待機していてもよいので、スレッドXとYとはそれら自体のメモリへの書き込みを実行することができる。イベントAは「書き込み完了」イベントであってもよく、イベントBは「メモリドライバ利用可能」イベントであってもよい。この場合、アクティブスレッド130は現在の

50

書き込みを終了したときに、両方のイベントを設定してよい。しかしながら、スレッド 130 はオペレーティングシステムによる強制を受けてもよく、1 回につき 1 つのイベントだけを設定することができてよい。アクティブスレッド 130 は、これらの 2 つのイベント上で待機中の保留スレッド X および Y の優先権についての情報を持たない。アクティブスレッド 130 は、任意の順序で、1 回につき 1 つのイベントで、これらの 2 つのイベントを設定してもよい。

【0021】

図 2 A に示されている実施例では、アクティブスレッド 130 は最初にイベント A を設定し、次いでイベント B を後で設定する。イベント A を設定する指示を受信すると、スレッド X が現時点で実行の準備ができており、現在のスレッド W よりも高い優先権を有していることから、スケジューラ 120 はスレッド X をスケジュールする。スレッド X は現在のスレッド W に割り込み、アクティブスレッド 130 として実行する。その後スレッド W は準備済み状態に移り、イベント B を設定する機会を得ることはない。スレッド X が完了され、保留状態に戻った後で、スケジューラ 120 は準備ができてい

10

20

【0022】

図 2 B は、2 つのイベントが異なる順序で設定されることによる、2 つの保留スレッド X と Y との異なる順序でのスケジューリングを示す。この例の中で、アクティブスレッド 130 は最初にイベント B を設定し、次いでイベント A を後で設定する。イベント B を設定する指示を受信すると、スケジューラ 120 はスレッド Y が現時点で実行のために準備ができてい

30

【0023】

図 2 A および 2 B に示されているように、保留スレッドはイベントが設定される順序に応じて、異なる順序で（かつ時には望ましくない順序で）スケジュールされる場合がある。スケジューラが、各イベントが設定された後で中間スケジューリングを実行しないような方法でイベントを設定することによって、改善された処理能力が達成されることができ

40

【0024】

図 3 は、スケジューラ 120 によるスレッドの中間スケジューリングを伴わずに、任意の数のイベントを設定するためのプロセス 300 を示す。プロセス 300 は、アクティブスレッド 130 によって実行されてもよい。

【0025】

最初に、設定される任意のイベントが存在するかどうかについての判定が行われる（ブロック 312）。答えが「いいえ」である場合、プロセス 300 はブロック 312 へ戻り

50

、待機する。一方、1つまたは複数のイベントが設定されることになっており、ブロック312について答えが「はい」である場合、スケジューラは、イベントが設定される期間、本質的に無効にされる（ブロック314）。スケジューラは、例えば（1）以下で説明されるように、現在のスレッドの優先権を操作することによるもの、（2）ハードウェアタイマの中断を介して時間を刻むタイマに基づいたスケジューリングを妨げる中断（例えばタイムチェック中断）をロックすることによるもの、または（3）適切なフラグを設定して、このフラグが設定されている間はスレッドをスケジュールしないようにスケジューラに指示することによるものなどの様々なメカニズムを使用して、無効にされてもよい。その後すべてのイベントは、例えばシステムに必要とされるのであれば、1回に1つのイベントで、設定される（ブロック316）。すべてのイベントが設定された後、スケジューラはその通常の効果に回復される（ブロック318）。このとき、スケジューラは設定されたすべてのイベントに基づいた適切な順序で、スレッドをスケジュールすることができる。

10

【0026】

イベントを設定する目的でスケジューラを無効にするための特定の実施形態は、以下で説明される。この実施形態は、スケジューラを本質的に無効にするために、現在のスレッドの優先権を操作する。

【0027】

図4は、優先権を操作することによって、スケジューラ120によるスレッドの中間スケジューリングを伴わずに任意の数のイベントを設定するためのプロセス400を示す。プロセス400は、設定される1つまたは複数のイベントが存在するときにはいつでも、アクティブスレッド130によって実行されてもよい。プロセス400は、図3のプロセス300のブロック314、316および318のために使用されてもよい。プロセス400はまた、イベントを設定するためにアクティブスレッド130によって呼び出される関数（例えばイベント設定関数）として実施されてもよい。

20

【0028】

最初に、図2Aおよび2Bの例についてはスレッドWである現在のスレッドの優先レベルは、一時的ロケーションに保存される（ブロック412）。次いで、現在のスレッドの優先権は、例えばシステム中で最も高い可能な優先レベルなどの高い優先レベルに、人為的に設定される（ブロック414）。現在のスレッドよりも高い優先権を備えた保留スレッドは特定のイベント上で待機していてもよいということが考えられる。割り込み優先権に基づくスケジューリングを用いるリアルタイムオペレーティングシステムでは、この特定のイベントが設定される場合には、保留スレッドがスケジュールされ、現在のスレッドに割り込みをする。イベントが設定される間の別のスレッドによる現在のスレッドへの割り込みを回避するために、現在のスレッドの優先権は最も高い可能な優先レベルに一時的に高められてもよい。

30

【0029】

次いで、例えば単一のイベントを設定するための関数（例えばイベント設定関数）を呼び出すことによって、イベントが設定される（ブロック416）。このイベントの設定によって、スケジューラはスケジューリングアルゴリズムを実行し、実行のために準備ができていて、すべてのスレッドの中で最も高い優先権を備えたスレッドをスケジュールしてもよい。しかしながら、現在のスレッドの優先権は最も高い可能なレベルにまで高められているので、別のスレッドがスケジュールされることはなく、現在のスレッドは処理され続けることになる。次いで、別のイベントが設定されることを必要としているかどうかについての判定が行われる（ブロック418）。答えが「はい」であれば、プロセス400は、例えば単一のイベントを設定するためのイベント設定関数を再度呼び出すことによって、別のイベントを設定するためにブロック416へ戻る。一方、設定される別のイベントが存在せず、ブロック418についての答えが「いいえ」である場合、現在のスレッドの優先権は本来の優先レベルに回復される（ブロック420）。

40

【0030】

50

現在のスレッドの優先権は、適切な関数（例えばスレッド優先権設定関数）を呼び出すことによって変更されてもよい。典型的に、この関数を呼び出すことはスケジューラを起動させ、スケジューラはその後適切にスレッドをスケジュールする。ブロック 4 1 4 で現在のスレッドが最も高い可能な優先レベルに設定されると、現在のスレッドが最も高いレベルにあることから、別のスレッドがスケジュールされることはない。ブロック 4 2 0 で現在のスレッドが本来の優先レベルに回復されると、スケジューラは通常のスケジューリングを実行し、現在のスレッドは、準備ができており、より高い優先権を有する別のスレッドによって割り込まれてもよい。

【0031】

図 2 C は、本明細書で説明される技術を使用する、「同時」に設定されている 2 つのイベントを用いる 2 つの保留スレッド X および Y のスケジューリングを示す。この例の中で、スケジューラ 1 2 0 は、例えばスレッド W を本来の優先レベルに維持しておくことによって、アクティブスレッド 1 3 0 が実行している間、有効である。イベント A および B を設定する前に、アクティブスレッド 1 3 0 は、例えばスレッド W の優先権を最も高い可能なレベルにまで高めることによって、スケジューラ 1 2 0 を本質的に無効にする。次いでアクティブスレッド 1 3 0 は、たとえスレッド Y が準備ができていても、スレッド Y のスケジューリングを起動することのないイベント B を設定する。次いでアクティブスレッド 1 3 0 は、たとえスレッド X または Y が準備ができていても、これらのスレッドのスケジューリングを起動することのないイベント A を設定する。次いでアクティブスレッド 1 3 0 は、例えばスレッド W の優先権を本来のレベルに回復することによって、スケジューラ 1 2 0 をその通常の効果に回復する。次いでスケジューラ 1 2 0 は、2 つのスレッド X および Y のうちでより高い優先レベルを有し、準備ができているスレッド X をスケジュールする。スレッド X が完了されると、スケジューラはスレッド Y をスケジュールする。

【0032】

図 5 は、オペレーティングシステムのコンテキストでの図 4 のプロセス 4 0 0 の例示的な実施を示す。オペレーティングシステムは、基本サービスを提供するために呼び出されてもよい様々なソフトウェア関数を備えたカーネルを含む。そのようなソフトウェア関数の 1 つは、スレッドが単一のイベントを設定することを可能にするイベント設定関数であってよい。オペレーティングシステムは規格品のオペレーティングシステムであってもよく、カーネルの中のソフトウェア関数を変更することができないか、または簡単には変更することができなくてもよい。

【0033】

さらなる機能を提供するため、かつ／またはカスタマイゼーションのために、抽象層がカーネルの上部に実装されてもよい。この抽象層は、スレッドが任意の数のイベントを設定することを可能にするイベント設定関数などの、さらなるソフトウェア関数を含んでもよい。この抽象層の上のアプリケーション層でランするスレッドは、抽象層の中のソフトウェア関数を呼び出してよく、この関数は、その後カーネルの中のソフトウェア関数を呼び出してよく。プロセス 4 0 0 は抽象層の中のイベント設定関数によって実施されてよく、この関数はその後設定されるべき各イベントについて 1 回、カーネルの中のイベント設定関数を呼び出してよく。

【0034】

図 5 は、図 4 のプロセス 4 0 0 の特定の実施態様を示す。その他の実施態様も可能であり、これも本発明の範囲内にある。

【0035】

プロセス 3 0 0 および 4 0 0 によって、スレッドは、スケジューラによる中間スケジューリングを伴わずに、同時にすべてのイベントを設定することができる。スケジューラは、各イベントが設定された後でスレッドのスケジュールを試みることがあるが、スケジューラは基本的に無効にされ、別のスレッドがスケジュールされることはない。プロセス 4 0 0 は、現在のスレッドの優先権を変更することによってスケジューラを無効にするので、イベントの設定の間にこのスレッドが割り込みされることはない。

【0036】

スケジューラを無効にするためのその他のメカニズムもまた使用されてよい。例えば、非同期化されたイベントが再スケジュールを引き起こすことを回避する時間に基づくメカニズムを用いて、スケジューラは無効にされてもよい。例として、ユーザインタフェース（UI）スレッドは、状態インジケータを1秒ごとにフラッシュすることを求めてもよい。その後オペレーティングシステムは、1秒ごとにアイコンをフラッシュするためにUIスレッドを起動する。UIスレッドがスレッドX、YおよびWよりも高い優先権を有する場合、これらのスレッドは、UIが起動されている場合UIによって割り込まれる。タイマ中断が一定方向に向けられることができず、したがってUIスレッドは起動されない。10
がないので、中断をロックすることはUIスレッドのスケジューリングを回避することができる。スケジューラを無効にするためのその他のメカニズムは、スケジュールを求めるたびにスケジューラがチェックするフラグ（例えばスケジュール不可フラグ）を設定することによるものである。スケジューラはこのフラグが設定されている場合にはスケジューリングを止め、このフラグが設定解除／クリアされている場合にはスケジューリングを続行する。アクティブスレッドは、スケジューラを無効にしたいときにはいつでもこのフラグを設定してよく、その後スケジューラを再び使用可能にするためにフラグを設定解除してもよい。スケジューラが無効にされている間、複数のイベントは中間スケジューリングを伴わずに設定されることができる。

【0037】

明快であるために、イベント設定技術はオペレーティングシステムのコンテキストで説明されてきた。一般に、これらの技術は、タスクが優先権およびイベントに基づいてスケジュールされる任意のマルチスレッドシステムのために使用されてよい。マルチスレッドシステムはソフトウェアシステム、ハードウェアシステム、またはハードウェアとソフトウェアの両方を備えたシステムであってよい。20

【0038】

図6は、イベント設定技術が用いられてもよいマルチスレッドシステム600の図を示す。タスク610x、610yおよび610zは、図6に示されていない1つまたは複数の主体から受信され、スケジュールされて実行されることになる。タスク610はスレッド、プロセスなどであってよい。タスク610はまた、1つまたは複数のプロセッサまたはアプリケーションによって生成されてもよく、かつ／あるいは1つまたは複数のインタフェースを介して受信されてもよい。30

【0039】

スケジューラ620はタスク610を受信し、処理モジュール630による処理のために、これらのタスクをスケジュールする。スケジューラ620は（1）これらのタスクについての優先値および保留イベント、ならびに（2）処理モジュール630によって設定されるイベントに基づいて、タスク610をスケジュールしてもよい。スケジューラ620はさらに、任意の時点で、実行のために準備ができていてすべてのタスクの中で最も高い優先権を備えたタスクが処理モジュール630によって処理されるように、タスク610をスケジュールしてもよい。処理モジュール630は、スケジューラ620によってスケジュールされた各タスク610を処理する。処理モジュール630はまた、イベントを適切に設定する。40

【0040】

スケジューラ620は様々な方法で実装されてもよい。例えば、スケジューラ620はコンピューティング装置またはパーソナル電子装置のために使用されるオペレーティングシステムの一部であってもよい。スケジューラ620はまた、処理モジュール630のためにタスク610をスケジュールするように設計されたコントローラであってもよい。したがってスケジューラ620はハードウェア、ソフトウェア、またはそれらの組合せの中で実装されてもよい。

【0041】

処理モジュール630も、ハードウェア、ソフトウェアまたはそれらの組合せの中で実

10

20

30

40

50

装されてもよい。処理モジュール630は、スケジュールされたタスクのための任意の種類の処理を実行してもよい。例えば、処理モジュール630は記憶装置コントローラ、メモリドライバ、バスドライバ、直接メモリアクセス(DMA)コントローラ、デジタル信号プロセッサ(DSP)、縮小命令セットコンピュータ(RISC)(reduced instruction set computer)、マイクロプロセッサなどであってもよい。

【0042】

本明細書で説明されるイベント設定技術およびマルチスレッドシステムは、様々な電子装置の中で用いられてもよい。例えば、これらの技術およびマルチスレッドシステムは、コンピュータシステム、パーソナル電子装置などの中で使用されてもよい。パーソナル電子装置は無線装置、PDAなどであってもよい。

10

【0043】

図7は、イベント設定技術が用いられてもよい無線装置700のブロック図である。無線装置700は携帯電話、端末、ハンドセットまたは何らかのその他の装置であってもよい。無線装置700は符号分割多重アクセス(CDMA)システム、移動通信のための世界システム(GSM)のシステムなどと通信することができてよい。

【0044】

無線装置700は、受信経路と送信経路とを經由する双方向通信を提供する。受信経路については、基地局によって送信される順方向リンク信号はアンテナ712によって受信され、デュプレクサ(D)714を通じて發送され、受信装置(RCVR)716に与えられる。受信装置716は受信された信号を調整かつデジタル化し、さらなる処理のためにデジタル部分720に入力サンプルを与える。送信経路については、送信装置(TMTX)718は送信されるべきデータをデジタル部分720から受信して、そのデータを処理かつ調整し、デュプレクサ714を通じて發送され、アンテナ712を介して基地局に送信される逆方向リンク信号を生成する。

20

【0045】

デジタル部分720は、例えばモデムプロセッサ722、ビデオ／グラフィックプロセッサ724、アプリケーションプロセッサ726、コントローラ／マイクロプロセッサ730、外部バスインタフェース(EBI)740および742、ならびに入力／出力(I/O)装置744などの、様々な処理装置およびインタフェース装置を含む。モデムプロセッサ722は、データの送信および受信のための処理(例えば符号化、変調、復調、復号など)を実行する。ビデオ／グラフィックプロセッサ724は、静止画像、動画、動くテキスト、ビデオゲームのためのグラフィック、3次元アバターなどに対する処理を実行する。アプリケーションプロセッサ726は、多元呼び出し、ウェブ閲覧、メディアプレーヤ、ユーザインタフェースなどのアプリケーションのための処理を実行する。EBI740はデジタル部分720と、RAM、SRAM、DRAM、SDRAMなどであってもよい揮発性メインメモリ750との間のデータの転送を促進する。EBI742はデジタル部分720と、フラッシュメモリまたはROMであってもよい不揮発性メモリ752との間のデータの転送を促進する。I/O装置744はLCD装置、キーパッド、スピーカ、マイクロフォンなどの外部I/O装置(図7に不図示)とインタフェースする。デジタル部分720内の様々な装置は、1つまたは複数のバス732を介して互いに通信してもよい。

30

40

【0046】

デジタル部分720は、1つまたは複数のDSP、マイクロプロセッサ、RISCなどととも実装されてもよい。デジタル部分720はまた、1つまたは複数の特定用途向け集積回路(ASIC)または何らかのその他の種類の集積回路(IC)上で作られてもよい。マルチスレッドシステム100および／または600は、デジタル部分720内で実装されてもよい。マルチスレッドシステム100の例として、マイクロプロセッサ730がリアルタイムオペレーティングシステムでランしていてもよく、デジタル部分720内の様々なプロセッサによってスレッド110が起動されてもよい。マルチスレッドシステ

50

ム 6 0 0 のその他の例として、マイクロプロセッサ 7 3 0 がスケジューラ 6 2 0 を実装し、E B I 7 4 0 が処理モジュール 6 3 0 を実装し、デジタル部分 7 2 0 内の様々なプロセッサによってタスク 6 1 0 が生成されてもよい。

【0047】

図 8 は、ここでもまたイベント設定技術が用いられてもよい記憶システム 8 0 0 のブロック図を示す。記憶システム 8 0 0 はマスタコントローラ 8 1 0 と複数の記憶装置 8 2 0 とを含む。マスタコントローラ 8 1 0 は独立型のコンピューティング装置、コンピューティング装置の一部、およびその他であってもよい。メモリ装置 8 1 2 は、マスタコントローラ 8 1 0 によって使用されるデータおよびコードを記憶する。

【0048】

マスタコントローラ 8 1 0 は、1 つまたは複数の外部インタフェース 8 1 4 を介して外部装置（図 8 に不図示）と通信する。マスタコントローラ 8 1 0 はさらに、1 つまたは複数の内部インタフェース 8 1 6 を介して記憶装置 8 2 0 に結合している。各インタフェースはまた、バスまたは何らかのその他の通信リンクであってもよい。各記憶装置 8 2 0 は直接マスタコントローラ 8 1 0 に結合するか、またはその他の記憶装置を介してマスタコントローラ 8 1 0 に結合してもよい。

【0049】

マスタコントローラ 8 1 0 は、記憶装置 8 2 0 へのアクセスのスケジューリングに対して責任を負うスケジューラを実装してもよい。マスタコントローラ 8 1 0 は、外部インタフェース 8 1 4 を介してタスクを受信してもよい。これらのタスクは、記憶装置 8 2 0 のアクセスを要求する外部プロセッサおよび／またはアプリケーションからののものであってもよい。マスタコントローラ 8 1 0 は次いで、タスクについての優先権および保留イベント、記憶装置の利用率、これらの記憶装置のために設定されたイベント、およびその他に基づいてタスクをスケジュールする。マスタコントローラ 8 1 0 は、準備ができてい最も高い優先権が各記憶装置にアクセスするように、タスクのスケジュールを試みてもよい。

【0050】

1 つの実施形態で、独立の処理モジュールは各記憶装置 8 2 0 のアクセスを管理する。各記憶装置のための処理モジュールは、記憶装置へのデータの書き込みと記憶装置からのデータの読み取りに対して責任を負うドライバであってもよい。処理モジュールは、処理モジュールおよび／または関連した記憶装置の状態に基づいて、イベントを設定してもよい。各記憶装置 8 2 0 について、マスタコントローラ 8 1 0 は、その装置のためにイベントが設定されるたびに、記憶装置 8 2 0 へアクセスするために最も高い優先権の準備ができていタスクをスケジュールしてもよい。各記憶装置のための処理モジュールは、スケジューラによる中間スケジューリングを伴わずにイベントを設定する目的で、（例えば現在のタスクの優先権を操作することによって）マスタコントローラ 8 1 0 内のスケジューラを無効にするために、プロセス 3 0 0 または 4 0 0 を実施してもよい。

【0051】

別の実施形態で、記憶装置 8 2 0 はマスタコントローラ 8 1 0 によってスケジュールされたタスクを表す。例えば、複数の記憶装置は所与の外部インタフェースへのアクセスを必要としてもよい。その後マスタコントローラ 8 1 0 は、記憶装置の優先権、外部インタフェースのために設定されたイベントなどに基づいて、この外部インタフェースを使用するためにこれらの記憶装置をスケジュールしてもよい。

【0052】

本明細書で説明された技術は、上で説明されたように、イベントを設定するために使用されてよい。これらの技術はまた、中断を設定するため、信号またはメッセージを送信するため、フラグを設定するため、およびその他のために使用されてもよい。

【0053】

本明細書で説明される技術は、様々な方法で実装されてもよい。例えば、これらの技術はソフトウェア、ハードウェアまたはそれらの組合せの中で実装されてもよい。ソフトウ

10

20

30

40

50

ウェア実装のために、この技術は本明細書で説明される動作を実行するモジュール（例えば関数、手順など）を用いて実装されてもよい。ソフトウェアコードはメモリ装置（例えば図7のメモリ750、または図8のメモリ812）の中に記憶され、プロセッサ（例えば図7のマイクロプロセッサ730、または図8のコントローラ810）によって実行されてもよい。メモリ装置は、プロセッサ内か、またはプロセッサの外部に実装されてもよい。

【0054】

ハードウェア実装のために、イベントを設定するモジュールまたは関数は、1つまたは複数のASIC、DSP、デジタル信号処理装置（DSPD）、プログラム可能論理装置（PLD）、フィールドプログラム可能ゲートアレイ（FPGA）、プロセッサ、コントローラ、マイクロコントローラ、マイクロプロセッサ、電子装置、本明細書で説明される機能を実行するように設計されたその他の電子装置、またはそれらの組合せの中で実装されてもよい。

10

【0055】

先の開示される実施形態の説明は、任意の当業者が本発明を作成または使用することができるように提供される。これらの実施形態に対する様々な変更は、当業者には容易に理解されることと思われ、本明細書で定義される一般的な原理は、本発明の精神または範囲から逸脱することなく、別の実施形態に適用されてもよい。したがって、本発明は本明細書で示される実施形態に限定されることが意図されているわけではなく、本明細書で開示される原理および新しい機能に矛盾しない最も広い範囲に一致する。

20

【図面の簡単な説明】

【0056】

【図1】例示的なマルチスレッドシステムを示す図である。

【図2A】図2Bとは異なる順序で設定される、2つのイベント上で待機中の2つのスレッドのスケジューリングを示す図である。

【図2B】図2Aとは異なる順序で設定される、2つのイベント上で待機中の2つのスレッドのスケジューリングを示す図である。

【図2C】中間スケジューリングを伴わずに設定される、2つのイベント上で待機中の2つのスレッドのスケジューリングを示す図である。

【図3】中間スケジューリングを伴わずにイベントを設定するためのプロセスを示す図である。

30

【図4】優先権を操作することによって、中間スケジューリングを伴わずにイベントを設定するためのプロセスを示す図である。

【図5】図4のプロセスの実施を示す図である。

【図6】別のマルチスレッドシステムを示す図である。

【図7】無線装置のブロック図である。

【図8】記憶システムのブロック図である。

【図 1】

図 1

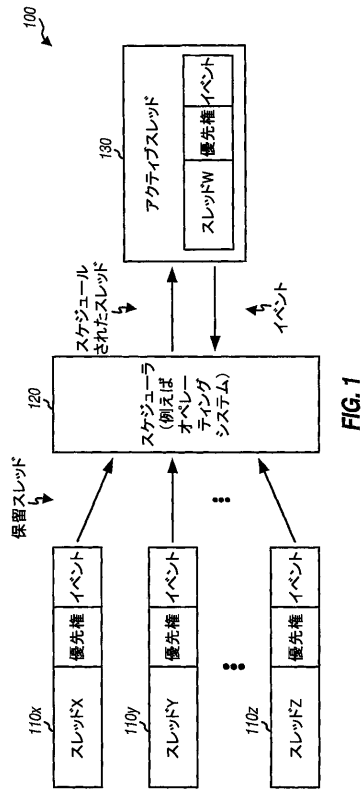


FIG. 1

【図 2 A】

図 2A

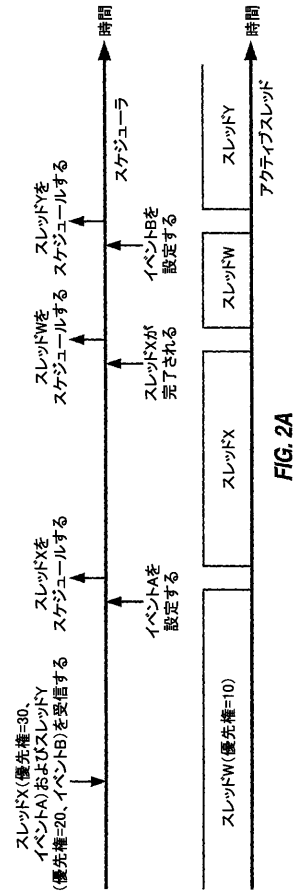


FIG. 2A

【図 2 B】

図 2B

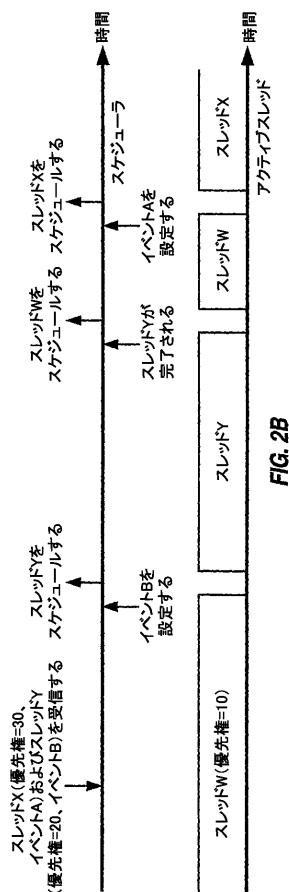


FIG. 2B

【図 2 C】

図 2C

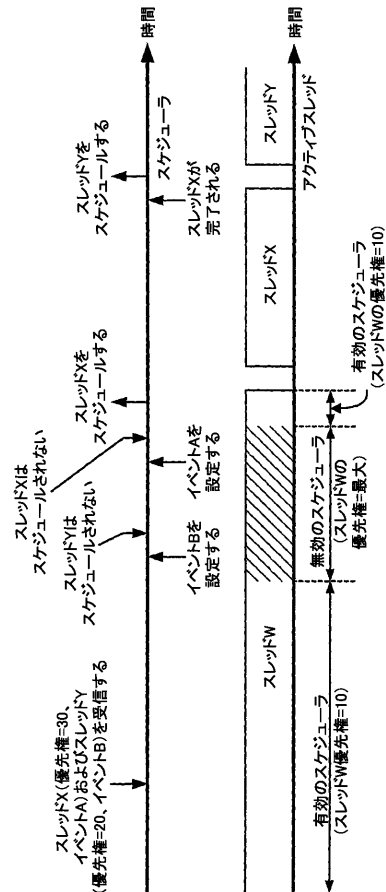


FIG. 2C

【図 3】

図 3

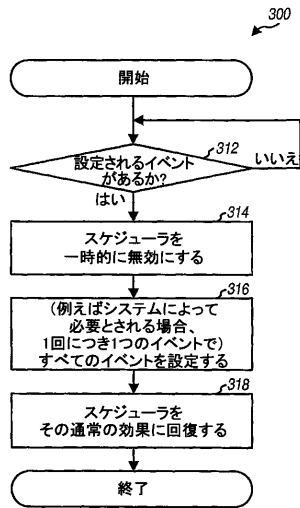


FIG. 3

【図 4】

図 4

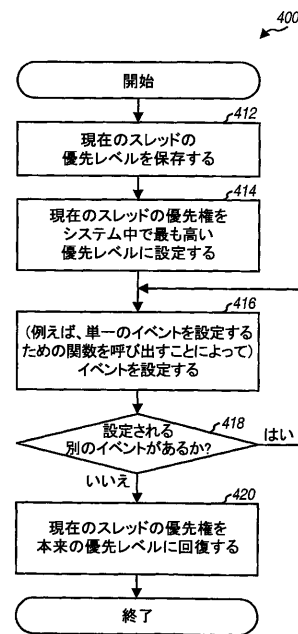


FIG. 4

【図 5】

図 5

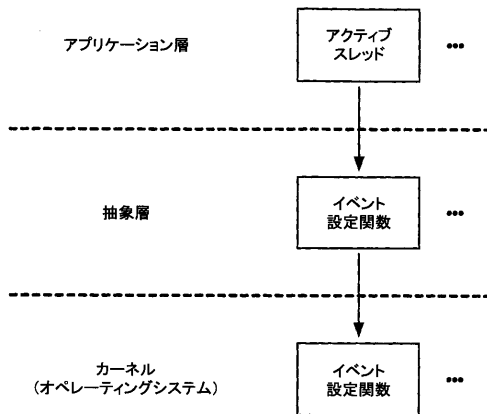


FIG. 5

【図 6】

図 6

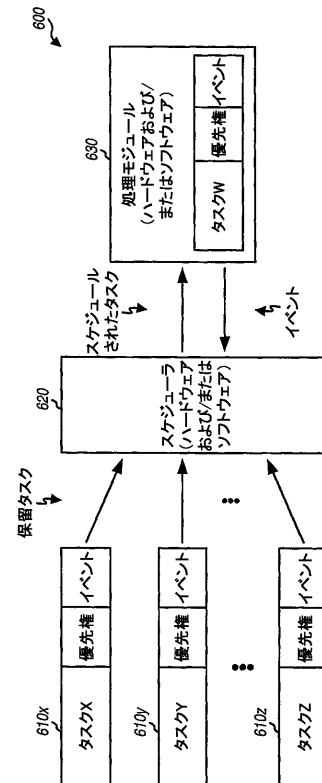
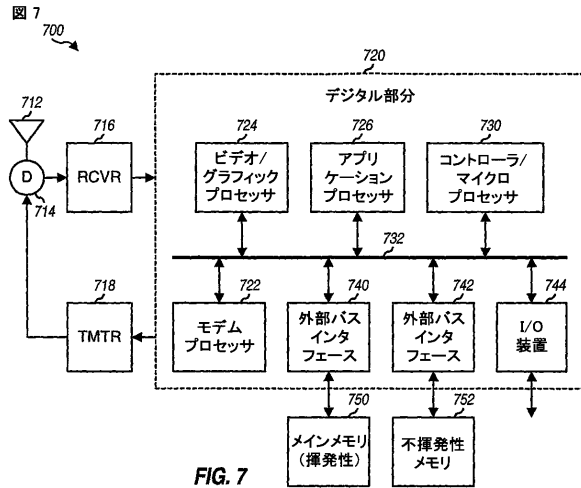


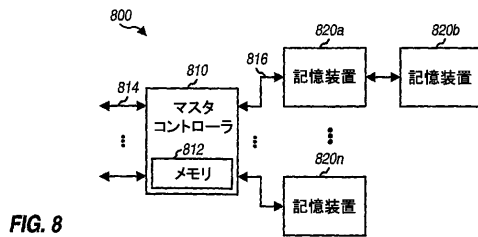
FIG. 6

【図 7】



【図 8】

図 8



【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2006/014306

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F9/46

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
L,X L,Y	KR 2000 060 827 A (LG ELECTRONICS INC) 16 October 2000 (2000-10-16)	1,9,13, 16,20 2-8, 10-12, 14,15, 17-19, 21-24 1
P,X	-& US 6 907 606 B1 (JANG MIN-SEOK) 14 June 2005 (2005-06-14) column 6, line 15 - line 28 ----- -/--	

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search

15 September 2006

Date of mailing of the international search report

28/09/2006

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,
Fax: (+31-70) 340-3016

Authorized officer

Ebert, Werner

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2006/014306

Q(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	LUI SHA ET AL: "PRIORITY INHERITANCE PROTOCOLS: AN APPROACH TO REAL-TIME SYNCHRONIZATION" IEEE TRANSACTIONS ON COMPUTERS, IEEE SERVICE CENTER, LOS ALAMITOS, CA, US, vol. 39, no. 9, 1 September 1990 (1990-09-01), pages 1175-1185, XP000149625 ISSN: 0018-9340 page 1177, section III. The Basic Priority Inheritance Protocol -----	2-8, 10-12, 14,15, 17-19, 21-24
A	WO 03/062988 A (KONINKLIJKE PHILIPS ELECTRONICS N.V; DIJKSTRA, HENDRIK; DIJKHOF, ANTON) 31 July 2003 (2003-07-31) page 3, line 20 - line 31 -----	1-24

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/US2006/014306

Patent document cited in search report		Publication date	Patent family member(s)	Publication date
KR 2000060827	A	16-10-2000	US 6907606 B1	14-06-2005
US 6907606	B1	14-06-2005	KR 2000060827 A	16-10-2000
WO 03062988	A	31-07-2003	CN 1615472 A	11-05-2005
			JP 2005516281 T	02-06-2005
			US 2005125789 A1	09-06-2005

フロントページの続き

(81)指定国 AP(BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), EP(AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG), AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW

(74)代理人 100075672

弁理士 峰 隆司

(74)代理人 100109830

弁理士 福原 淑弘

(74)代理人 100095441

弁理士 白根 俊郎

(74)代理人 100084618

弁理士 村松 貞男

(74)代理人 100103034

弁理士 野河 信久

(74)代理人 100140176

弁理士 砂川 克

(74)代理人 100092196

弁理士 橋本 良郎

(74)代理人 100100952

弁理士 風間 鉄也

(72)発明者 ブラックマン、デイビッド

アメリカ合衆国、カリフォルニア州 82126、サン・ディエゴ、ジェイド・コースト・ドライブ 8387