(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6070150号
(P6070150)
(45)発行日　平成29年2月1日(2017.2.1)　(24)登録日　平成29年1月13日(2017.1.13)

(51)Int.Cl.　F I
G06F 9/50 (2006.01)　G06F 9/46 462A

請求項の数 8 (全 26 頁)

(21)出願番号　特願2012-273986(P2012-273986)
(22)出願日　平成24年12月14日(2012.12.14)
(65)公開番号　特開2014-119918(P2014-119918A)
(43)公開日　平成26年6月30日(2014.6.30)
審査請求日　平成27年8月4日(2015.8.4)

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100089118
弁理士　酒井　宏明
(72)発明者　中島　耕太
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(72)発明者　成瀬　彰
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

審査官　大塚　俊範

最終頁に続く

(54)【発明の名称】情報処理装置、情報処理装置の制御方法及び情報処理装置の制御プログラム

(57)【特許請求の範囲】
【請求項１】
ポーリングによる受信処理が割り当てられた通信制御スレッドを含む複数のスレッドをそれぞれ実行する演算処理部を備え、
前記演算処理部は、自己において実行される複数の前記スレッドの中の何れかが各々に割り当てられ、且つ、割り当てられたスレッドを実行する複数のスレッド実行部を有し、
前記スレッド実行部のうち、前記通信制御スレッドを実行するスレッド実行部は、データの到着の通知を示すメモリ領域に対してポーリングを行いデータの到着まで受信処理の実行を待機し、且つ、他のスレッドを実行するスレッド実行部が、前記他のスレッドに割り当てられた処理を実行している場合、物理資源の使用を抑制する省消費資源モードに遷移する
ことを特徴とする情報処理装置。
【請求項２】
前記他のスレッドは、前記受信処理以外のアプリケーションの実行処理が割り当てられたアプリスレッドを含み、
前記複数のスレッド実行部のうち、前記通信制御スレッドを実行するスレッド実行部は、ポーリングを行いデータの到着まで受信処理の実行を待機し、且つ、前記アプリスレッドを実行するスレッド実行部が割り当てられた処理を実施している場合、前記演算処理部が有する物理資源の使用を抑制する省消費資源モードに遷移する
ことを特徴とする請求項１に記載の情報処理装置。

【請求項３】
前記複数のスレッド実行部のうち、前記通信制御スレッドを実行するスレッド実行部は、メモリの所定領域が割り当てられており、前記所定領域が更新された場合に、データが到着したと判定し、受信処理を開始することを特徴とする請求項１又は請求項２に記載の情報処理装置。
【請求項４】
前記スレッドを実行する複数の前記スレッド実行部のうちいずれか一つは、省消費資源モードに遷移していないことを特徴とする請求項１～３のいずれか一つに記載の情報処理装置。
【請求項５】
前記通信制御スレッドは、前記通信制御スレッドを実行するスレッド実行部の動作状態として、停止状態と、ポーリングを行いデータの到着まで受信処理の実行を待機している受信待ち状態と、データが到着し受信処理を実施している処理実行状態とを有し、
前記アプリスレッドは、前記アプリスレッドを実行するスレッド実行部の動作状態として、待機状態と、割り当てられた処理を実施しているアプリ実行状態とを有し、
前記通信制御スレッドを実行するスレッド実行部が前記処理実行状態であり、前記アプリスレッドを実行するスレッド実行部が前記アプリ実行状態でない場合、前記アプリスレッドを実行するスレッド実行部は前記省消費資源モードに遷移し、
前記アプリスレッドを実行するスレッド実行部が前記アプリ実行状態であり、前記通信制御スレッドを実行するスレッド実行部が前記処理実行状態でない場合、前記通信制御スレッドを実行するスレッド実行部は前記省消費資源モードに遷移し、
前記アプリスレッドを実行するスレッド実行部が前記待機状態で、前記通信制御スレッドを実行するスレッド実行部が前記停止状態の場合、前記通信制御スレッドを実行するスレッド実行部は前記省消費資源モードに遷移し、
前記アプリスレッドを実行するスレッド実行部が前記待機状態で、前記通信制御スレッドを実行するスレッド実行部が前記受信待ち状態の場合、前記アプリスレッドを実行するスレッド実行部は前記省消費資源モードに遷移する
ことを特徴とする請求項２に記載の情報処理装置。
【請求項６】
前記通信制御スレッドを実行するスレッド実行部は、動作状態として、データが到着し受信処理を実施している処理実行状態を有し、
前記アプリスレッドを実行するスレッド実行部は、前記省消費資源モードへの遷移は行わず、
前記通信制御スレッドを実行するスレッド実行部は、自己が前記処理実行状態以外の場合、前記省消費資源モードに遷移する
ことを特徴とする請求項２に記載の情報処理装置。
【請求項７】
情報処理装置が有する演算処理部が、
ポーリングによる受信処理が割り当てられた通信制御スレッドを含む前記演算処理部において実行される複数のスレッドが各々に割り当てられ、且つ、割り当てられたスレッドをそれぞれ実行する複数のスレッド実行部のうち、データの到着の通知を示すメモリ領域に対するポーリングによる受信処理が割り当てられた通信制御スレッドを実行するスレッド実行部を決定し、
決定された前記通信制御スレッドを実行するスレッド実行部を、前記メモリ領域に対してポーリングを行いデータの到着まで受信処理の実行を待機させ、且つ、他のスレッドを実行するスレッド実行部が、前記他のスレッドに割り当てられた処理を実行している場合、物理資源の使用を抑制する省資源モードに遷移させる
ことを特徴とする情報処理装置の制御方法。
【請求項８】
情報処理装置が有する演算処理部に、

ポーリングによる受信処理が割り当てられた通信制御スレッドを含む前記演算処理部において実行される複数のスレッドが各々に割り当てられ、且つ、割り当てられたスレッドをそれぞれ実行する複数のスレッド実行部のうち、データの到着の通知を示すメモリ領域に対するポーリングによる受信処理が割り当てられた通信制御スレッドを実行するスレッド実行部を決定させ、

決定された前記通信制御スレッドを実行するスレッド実行部を、前記メモリ領域に対してポーリングを行いデータの到着まで受信処理の実行を待機させ、且つ、他のスレッドを実行するスレッド実行部が、前記他のスレッドに割り当てられた処理を実行している場合、物理資源の使用を抑制する省資源モードに遷移させる

処理を行わせることを特徴とする情報処理装置の制御プログラム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、情報処理装置、情報処理装置の制御方法及び情報処理装置の制御プログラムに関する。

【背景技術】

【０００２】

近年、ビジネスアプリケーションの分野において、レスポンスの速度の向上などが求められることが増えてきている。ビジネスアプリケーションとしては、証券取引、データベース処理又はゲームなどのアプリケーションがある。例えば、証券取引システムでは、取引が成立するまでの時間の許容範囲が、従来はｍｓｅｃ単位であったものが、近年ではμｓｅｃ単位になってきている。また、データベース処理では、高いＩＯＰＳ（Input/Output Per Second）が求められてきている。さらに、最近のゲームでは、実時間性が重要になってきており、高速な処理が求められている。

【０００３】

このようなことから、ビジネスアプリケーションの分野では、各サーバ間の通信時間を短縮するために低遅延サーバ間通信技術が用いられることが増えてきている。

【０００４】

低遅延サーバ間通信技術としては、スーパーコンピュータといったHigh Performance Computing（ＨＰＣ）の分野において、例えば、片道１μｓｅｃ以下の通信を行うＩｎｆｉｎｉＢａｎｄ（インフィニバンド）などの技術が提供されている。

【０００５】

このような低遅延サーバ間通信技術におけるデータの受信を検知するための受信検知方式として、割り込み方式又はポーリング方式といった方式が用いられることがある。

【０００６】

割り込み方式は、データを受信した場合に割り込みが発生し、割り込みの発生を契機にデータの受信処理が実行される方法である。一方、ポーリング方式は、データの受信処理を起動させた状態で、データを受信したか否かを定期的に確認するポーリングを行い、データを受信していれば受信処理を進ませる方法である。割り込み方式では、割り込みが発生してから受信処理が動作するため、遅延が２～３μｓｅｃとなる場合がある。これに対して、ポーリング方式では、受信処理は最初から起動しているため、遅延は１μｓｅｃ以下というように割り込み方式よりも短く抑えることができる。ただし、ポーリング方式は、ポーリングを実行している間、その処理を行う部位を占有してしまう。例えば、演算処理部であるＣＰＵ（Central Processing Unit）コアの単位で処理が割り振られる場合、ポーリング方式ではポーリングの間、データ受信処理が１つのＣＰＵコアを占有してしまう。

【０００７】

この点、従来のＨＰＣアプリケーションの場合、実施される処理の数がＣＰＵコアの数以下であったため、ポーリング方式による受信検出を行った場合でも、データの受信処理がＣＰＵコアを占有することの弊害は少なかった。しかし、ビジネスアプリケーションの

場合、実施する処理の数が多くＣＰＵコアの数を超えてしまう。この場合、データ受信処理によるＣＰＵコアの占有が他の処理に影響を与えてしまうおそれがある。

【０００８】

さらに、ＣＰＵコアの占有を回避するため、ＳＭＴ（Simultaneous Multi-Threading）を利用したポーリングが提案されている。ＳＭＴとは、演算処理部としてのＣＰＵコア又はＣＰＵコアに含まれる実行部がそれぞれ別個に処理を行うスレッドという単位にプログラムを分け、各スレッドの処理を複数のＣＰＵコア又は実行部により同時に実行させるＣＰＵの機能である。ＳＭＴを用いてポーリングを行う場合、例えば、１つのスレッドはアプリケーションの実行用とし、他の１つのスレッドをポーリング用としてＣＰＵコア又は実行部を動作させる。このポーリングを行うスレッドを、以下では「通信制御スレッド」と呼ぶ。これにより、ポーリングを行うデータ受信処理は通信制御スレッドのみを占有することとなり、ポーリングを行っている場合でも ＣＰＵコア又は実行部は、他のスレッドでアプリケーションを実行することができる。

【０００９】

ここで、複数のＣＰＵコアがそれぞれ別の処理を実行するマルチタスクの技術として、各ＣＰＵコアが周辺機器に処理依頼を行い、あるＣＰＵコアが処理結果を待つ状態のタスクは優先度を下げる従来技術がある（例えば、特許文献１参照）。また、Ｉ／Ｏドライバに対して処理要求を行った後、Operation System（ＯＳ）の負荷が所定値以下であれば、割り込み要求を禁止した上で、Ｉ／Ｏデバイスにポーリングを行い、所定時間経過後割り込み要求の禁止を解除する従来技術がある（例えば、特許文献２参照）。

【００１０】

また、メイン処理装置の代わりにタスクを実行する補助処理装置を有し、補助処理装置の動作に基づいてメイン処理装置を省電力モードに移行させる従来技術もある（例えば、特許文献３参照）。

【先行技術文献】

【特許文献】

【００１１】

【特許文献１】特開２００６－２６０３７７号公報
【特許文献２】特開２００１－２１６１７０号公報
【特許文献３】特開２００５－３４６７０８号公報

【発明の概要】

【発明が解決しようとする課題】

【００１２】

しかしながら、ＳＭＴを用いた場合にも、各スレッドが共通で利用するＣＰＵコアの資源が存在する。そのため、通信制御スレッドがポーリングを行っている場合のＣＰＵコアの資源の消費により、アプリケーションを実行しているスレッドの性能の低下を招いてしまうおそれがある。

【００１３】

また、補助処理装置にタスクを実行させる技術は、ポーリングの状態に対応させて省電力モードの動作を制御する技術であり、アプリケーションの実行に対するポーリングの影響は考慮されていない。そのため、この技術を用いても、ポーリングによるアプリケーションの処理への影響を軽減することは困難である。また、複数のＣＰＵコアが個別に周辺機器に処理依頼を行う従来技術やＯＳの負荷が所定値以下の場合にポーリングを行う従来技術では、ＣＰＵコア毎に処理が行われているマルチタスクを対象としており、ＳＭＴを用いた場合については考慮されていない。したがって、このような従来技術を用いても、ＳＭＴのように１つのＣＰＵコアの中で複数のスレッドを動作させる状況で、高い通信性能とアプリケーションの処理能力の維持とを両立させることは困難である。

【００１４】

開示の技術は、上記に鑑みてなされたものであって、高い通信性能を維持しつつアプリケーションの処理性能の低下を抑制する情報処理装置、情報処理装置の制御方法及び情報

処理装置の制御プログラムを提供することを目的とする。
【課題を解決するための手段】
【００１５】
本願の開示する情報処理装置、情報処理装置の制御方法及び情報処理装置の制御プログラムは、一つの態様において、ポーリングによる受信処理が割り当てられた通信制御スレッドを含む複数のスレッドをそれぞれ実行する演算処理部を備える。前記演算処理部は、自己において実行される複数の前記スレッドの中の何れかが各々に割り当てられ、且つ、割り当てられたスレッドを実行する複数のスレッド実行部を有する。前記スレッド実行部のうち、前記通信制御スレッドを実行するスレッド実行部は、データの到着の通知を示すメモリ領域に対してポーリングを行いデータの到着まで受信処理の実行を待機し、且つ、他のスレッドを実行するスレッド実行部が、前記他のスレッドに割り当てられた処理を実行している場合、物理資源の使用を抑制する省消費資源モードに遷移することを特徴とする。
【発明の効果】
【００１６】
本願の開示する情報処理装置、情報処理装置の制御方法及び情報処理装置の制御プログラムの一つの態様によれば、高い通信性能を維持しつつアプリケーションの処理性能の低下を抑制することができるという効果を奏する。
【図面の簡単な説明】
【００１７】
【図１】図１は、実施例１に係る情報処理システムの構成を示す構成図である。
【図２】図２は、スレッドを説明するための図である。
【図３】図３は、スレッドが使用するＣＰＵコアの資源を説明するための図である。
【図４】図４は、消費資源モードとＣＰＵコアの省電力モードとの関係を示す図である。
【図５】図５は、各消費資源モードでの受信処理及びアプリケーション処理の関係を説明するための図である。
【図６】図６は、実施例１に係る各スレッドの消費資源モードの設定の一例を説明するための図である。
【図７】図７は、実施例１に係るアプリスレッドのＩｄｌｅルーチンにおける動作のフローチャートである。
【図８】図８は、実施例１に係る通信制御スレッドのＩｄｌｅルーチンにおける動作のフローチャートである。
【図９】図９は、実施例１に係る通信制御スレッドのｍｐｏｌｌルーチンにおける動作のフローチャートである。
【図１０】図１０は、実施例１に係る各スレッドの動作モード及び消費資源モードの遷移の一例を示すタイムチャートである。
【図１１】図１１は、ポーリングによるアプリケーション処理への影響を説明するための図である。
【図１２】図１２は、実施例２に係る各スレッドの消費資源モードの設定の一例を説明するための図である。
【図１３】図１３は、実施例２に係るアプリスレッドのＩｄｌｅルーチンにおける動作のフローチャートである。
【図１４】図１４は、実施例２に係る通信制御スレッドのＩｄｌｅルーチンにおける動作のフローチャートである。
【図１５】図１５は、実施例２に係る通信制御スレッドのｍｐｏｌｌルーチンにおける動作のフローチャートである。
【図１６】図１６は、実施例２に係る各スレッドの動作モード及び消費資源モードの遷移の一例を示すタイムチャートである。
【図１７】図１７は、情報処理システムの他の構成を示す構成図である。
【発明を実施するための形態】

【００１８】
以下に、本願の開示する情報処理装置、情報処理装置の制御方法及び情報処理装置の制御プログラムの実施例を図面に基づいて詳細に説明する。なお、以下の実施例により本願の開示する情報処理装置、情報処理装置の制御方法及び情報処理装置の制御プログラムが限定されるものではない。
【実施例１】
【００１９】
図１は、実施例１に係る情報処理システムの構成を示す構成図である。図１に示すように、本実施例に係る情報処理システムは情報処理装置であるサーバ１及びサーバ２を有している。
【００２０】
サーバ１は、ＣＰＵコア１０及び１１、メモリ１２、Ｉ／Ｏ（Input/Output）１３、Ｉ／Ｏ１４、メモリコントローラ１５、並びにＩ／Ｏブリッジ１６を有している。そして、ＣＰＵコア１０及び１１とメモリコントローラ１５とはバスで接続されており、メモリコントローラ１５とメモリ１２とはバスで接続されている。また、メモリコントローラ１５とＩ／Ｏブリッジ１６とはバスで接続されており、Ｉ／Ｏブリッジ１６とＩ／Ｏ１３及びＩ／Ｏ１４とはバスで接続されている。ここで、Ｉ／Ｏ１３は、ディスクなどである。
【００２１】
Ｉ／Ｏ１４は、ホスト　チャネル　アダプタ（Host　Channel　Adapter：ＨＣＡ）などのインフィニバンド用の通信路である。
【００２２】
サーバ２も同様に、ＣＰＵコア２０及び２１、メモリ２２、Ｉ／Ｏ２３、Ｉ／Ｏ２４、メモリコントローラ２５、並びにＩ／Ｏブリッジ２６を有している。また、サーバ２における各部の接続もサーバ１と同様である。
【００２３】
サーバ１とサーバ２とは、例えば、インフィニバンド用のバス３で接続されている。ここで、説明の便宜上、図１ではサーバ１及びサーバ２はバス３により互いに接続されているが、実際には、サーバ１とサーバ２とはスイッチを経由して接続されている。また、本実施例では、説明の便宜上、サーバ１及びサーバ２という２台のサーバのみを記載しているが、複数であればサーバの台数に特に制限はなく、それぞれがインフィニバンド用のケーブルを用いたバス３で接続されていればよい。
【００２４】
そして、バス３には、ネットワークスイッチ４が接続されている。ネットワークスイッチ４は、バス３を介して送受信されるデータの伝送経路を制御する。
【００２５】
次に、サーバ１及びサーバ２の各部について詳細に説明する。サーバ１及びサーバ２は、同様の構成を有するため、ここでは、サーバ１を例に説明する。
【００２６】
ＣＰＵコア１０及びＣＰＵコア１１は、上述したようにメモリコントローラ１５に接続される演算処理部である。ここで、本実施例では、サーバ１は、ＣＰＵコア１０及びＣＰＵコア１１という２つのＣＰＵコアを有するように説明しているが、ＣＰＵコアの数には特に制限は無い。また、ＣＰＵコア１０及び１１は、内部に論理的なハードウェアの単位としてのスレッドを有している。本実施例では、ＣＰＵコア１０及び１１は、後述するように、２つのスレッドを有している。スレッドについては後で詳細に説明する。
【００２７】
メモリ１２は、後述するＣＰＵコア１０の各スレッドに割り当てられたメモリ領域を有している。例えば、ＣＰＵコア１０の２つのスレッドに、メモリ領域１２１及び領域１２２がそれぞれ割り当てられている。
【００２８】
メモリコントローラ１５は、ＣＰＵコア１０及び１１からの指示を受けて、メモリ１２

に対するデータの書き込み及びメモリ１２からのデータの読み出しを行う。
【００２９】
Ｉ／Ｏ１３は、例えばハードディスクや入出力装置などを有している。入出力装置としては、例えば、キーボードやモニタなどがある。Ｉ／Ｏ１３のキーボード及びモニタなどをユーザインタフェースとして、操作者は、データの入力などを行う。また、ＣＰＵコア１０及び１１は、メモリコントローラ１５及びＩ／Ｏブリッジ１６を介してＩ／Ｏ１３のハードディスク等にデータを格納する。
【００３０】
また、ＣＰＵコア１０及び１１は、メモリコントローラ１５、Ｉ／Ｏブリッジ１６、Ｉ／Ｏ１３及びバス３を介して、サーバ２のＣＰＵコア２０及び２１などと通信を行い、データの送受信を行う。このように、ＣＰＵ１０及び１１は、メモリコントローラ１５やＩ／Ｏブリッジ１６を経由してＣＰＵ２０及び２１とデータの送受信を行うが、以下では説明の都合上、メモリコントローラ１５やＩ／Ｏブリッジ１６を省いてデータの送受信を説明する場合がある。
【００３１】
図２は、スレッドを説明するための図である。本実施例では、ＣＰＵコア１０は、アプリスレッド２００及び通信制御スレッド２０１という２つのスレッドを有する。また、ＣＰＵコア２０は、アプリスレッド２０２及び通信制御スレッド２０３という２つのスレッドを有する。スレッドとは、単一のＣＰＵコアにおいて別個に処理を行う論理的な単位である。そして、ＣＰＵ１０及びＣＰＵ２０は、自己が有する各スレッドのそれぞれに処理を同時に実行させる機能、すなわちＳＭＴの機能を有している。ＣＰＵコア１０及び１１は、同様の構成を有し同様の機能を有するため、ここでは、ＣＰＵコア１０を例に説明する。また、以下の説明では、説明の便宜上、各スレッドが処理を実行しているように説明する場合があるが、実際には各スレッドを実行する演算処理部がそれらの処理を実行している。この演算処理部とは、例えば、各スレッドを実行するＣＰＵコア又はＣＰＵコアの中の実行部がその一例にあたる。
【００３２】
アプリスレッド２００は、アプリケーションにおける通信制御処理以外の処理を行うスレッドである。例えば、サーバ１で実行されるアプリケーションが証券取引のアプリケーションであれば、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、証券の売買を成立させるための処理などを実施する。アプリスレッド２００が実施する処理を、以下では「アプリケーション処理」と言う場合がある。図２におけるアプリケーション２１０が、アプリスレッド２００を実行するＣＰＵコア１０又は実行部が実施しているアプリケーション処理を表している。また、アプリケーション２１４が、アプリスレッド２０２を実行するＣＰＵコア１０又は実行部が実施しているアプリケーション処理を表している。
【００３３】
アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、アプリケーション処理を実行しているモードと、アプリケーション処理を実行していないモードという２つの動作モードを有している。以下では、アプリケーション処理を実行していないモードを「Ｉｄｌｅ」という。以下では、アプリスレッド２００を実行するＣＰＵコア１０又は実行部における動作モードを、単にアプリスレッド２００の動作モードと言う場合がある。
【００３４】
また、アプリスレッド２００は、メモリの中の領域が割り当てられている。本実施例では、アプリスレッド２００は、例えば、図１に示すメモリ１２のメモリ領域１２１が割り当てられている。そして、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、メモリ領域１２１が更新されたか否かを監視している。
【００３５】
通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、サーバ２のＣＰＵ２０などからデータの受信処理などを行うスレッドである。例えば、通信制御スレッド２０

１を実行するＣＰＵコア１０又は実行部は、通信制御スレッド２０３が送信したデータを受信する。通信制御スレッド２０１が実施するデータの受信処理などを含む処理を、以下では「通信制御処理」という場合がある。図２における通信制御２１１が、通信制御スレッド２０１が実施している通信制御処理を表している。また、通信制御２１３が、通信制御スレッド２０１が実施している通信制御処理を表している。さらに、通信制御処理の中の受信処理には、データが送信されてきたか否かを判定するポーリングの処理が含まれている。ポーリングを実行している状態では、通信制御スレッド２０１は、単にメモリ領域が更新されたか否かを判定するだけなので、他の通信制御処理とは異なりＣＰＵコア１０の資源の使用を制限することができる。

【００３６】

通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、受信処理などのポーリング以外の通信制御処理を実行しているモードと、ポーリングを実行しており受信待ちのモードと、通信制御処理を実行していないモードという３つの動作モードを有している。以下では、ポーリングを実行しており受信待ちの動作モードを、「ｍｐｏｌｌ」と言い、通信制御処理を実行していないモードを、「Ｉｄｌｅ」という。以下では、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部における動作モードを、単に通信制御スレッド２０１の動作モードと言う場合がある。

【００３７】

また、通信制御スレッド２０１は、メモリの中の領域が割り当てられている。本実施例では、通信制御スレッド２０１は、例えば、図１に示すメモリ１２のメモリ領域１２２が割り当てられている。そして、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、メモリ領域１２２が更新されたか否かを監視している。

【００３８】

例えば、インテル（登録商標）製のＣＰＵの場合であれば、通信制御スレッド２０１は、ｍｐｏｌｌの状態では、ｍｗａｉｔ命令によるポーリング処理を行う。ｍｗａｉｔ命令は、指定したアドレスのメモリ領域が更新されるまでスレッドの動作を停止する命令である。指定したアドレスのメモリ領域とは、例えば、図１における通信制御スレッド２０１に割り当てられたメモリ領域１２２である。すなわち、通信制御スレッド２０１は、ｍｐｏｌｌの場合、メモリコントローラ１５によりメモリ領域１２２が更新されると動作の停止を解除し通信制御処理を実行するモードに復帰する。また、メモリ領域１２２の更新の他、割り込み発生時にも、通信制御スレッド２０１は、通信制御処理を実行するモードに復帰する。

【００３９】

ここで、本実施例では、説明の都合上、ＣＰＵ１０及び２０がそれぞれ２つのスレッドを有しているとして説明したが、複数であればスレッドの数に特に制限は無い。

【００４０】

図３は、スレッドが使用するＣＰＵコアの資源を説明するための図である。図３に示すように、アプリスレッド２００及び通信制御スレッド２０１のそれぞれが処理を行う場合に使用する資源である共通資源３２０が存在する。共通資源３２０には、演算器３２２を含む命令パイプライン３２１及びキャッシュ３２３などが含まれる。

【００４１】

アプリスレッド２００は、ＣＰＵコア１０の資源として、例えば、プログラムカウンタ３０１、命令フェッチ３０２、レジスタセット３０３、命令パイプライン３２１及びキャッシュ３２３を使用する。また、通信制御スレッド２０１は、ＣＰＵコア１０の資源として、例えば、プログラムカウンタ３１１、命令フェッチ３１２、レジスタセット３１３、命令パイプライン３２１及びキャッシュ３２３を使用する。

【００４２】

共通資源３２０は、アプリスレッド２００及び通信制御スレッド２０１のいずれのスレッドでも用いられ、一方のスレッドが使用している共通資源３２０は、他方のスレッドは使用できない。例えば、アプリスレッド２００が演算器３２２を使用している場合には、

通信制御スレッド２０１は演算器３２２を使用することはできない。

【００４３】

そして、アプリスレッド２００及び通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、ＣＰＵコア１０の資源の使用状態を決定する消費資源モードとして、通常モードと省消費資源モードという２つの消費資源モードを有している。省消費資源モードとは、ＣＰＵコア１０の資源の使用を抑制するモードである。通常モードは、ＣＰＵコア１０の資源の使用を抑制しないモードである。特に、省消費資源モードの場合、アプリスレッド２００及び通信制御スレッド２０１による共通資源３２０の使用も軽減される。そのため、一方のスレッドが省消費資源モードで動作している場合、その一方のスレッドによる共通資源３２０の利用が減るため、他方のスレッドは共通資源３２０をほぼいつでも利用可能であり、処理を迅速に行うことができる。以下では、アプリスレッド２００及び通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部における消費資源モードを、単に各スレッドの消費資源モードと言う場合がある。

【００４４】

消費資源モードは、例えば、インテル（登録商標）製のＣＰＵの場合であれば、省電力モードを用いることで実現できる。具体的には、インテル製のＣＰＵには、Ｃ０～Ｃ６という消費電力に関するモードが設けられている。Ｃ０は、処理を実行しているときのモードである。Ｃ１～Ｃ６は、消費電力を抑えるための省電力モードである。そして、Ｃ１～Ｃ６は、数字が大きいほどより消費電力が抑えられるモードである。この省電力モードでは、消費電力を下げるためにＣＰＵコア１０の資源の使用が抑制される。そのため、インテルから提供されている省電力モードを、本実施例における「省消費資源モード」として用いることができる。特に、Ｃ２～Ｃ６を本実施例における省消費資源モードとして用いた場合、処理を実行するためにＣ０のモードに移行する復帰時間が増大してしまう。そのため、Ｃ２～Ｃ６では通信遅延時間が増大するので、通信遅延を短くするために「省消費資源モード」としてＣ１を利用することが好ましい。また、インテルから提供されているＣ０の省電力モードが、本実施例に係る通常モードの消費資源モードにあたる。

【００４５】

そして、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、動作モードがＩｄｌｅの場合、消費資源モードに遷移できる。また、通信制御スレッド２０１は、動作モードがｍｐｏｌｌの場合、ｗａｉｔｅ命令による指示を受けることで、停止時に省消費資源モードか否かのいずれかの消費資源モードを選択することができる。

【００４６】

図４は、消費資源モードとＣＰＵコアの省電力モードとの関係を示す図である。図４の欄４００に示すように、アプリスレッド２００及び通信制御スレッド２０１の両方が省消費資源モードで動作している場合、ＣＰＵコア１０は省電力モードに移行してしまう。これに対して、欄４００以外の欄で示されるように、アプリスレッド２００又は通信制御スレッド２０１のいずれか一方が通常モードで動作していれば、ＣＰＵコア１０は、アクティブになっており省電力モードには移行しない。ＣＰＵコア１０が省電力モードに移行した場合、ＣＰＵコア１０が有する演算器などの動作が抑制されてしまう。つまり、アプリスレッド２００及び通信制御スレッド２０１が使用する共通資源の動作が抑制されてしまう場合がある。そのため、ＣＰＵコア１０が省電力モードに移行した場合、通信制御スレッド２０１の起動までに時間が掛かってしまい、受信処理の遅延が大きくなってしまう。

【００４７】

図５は、各消費資源モードでの受信処理及びアプリケーション処理の関係を説明するための図である。

【００４８】

アプリスレッド２００及び通信制御スレッド２０１のいずれも通常モードの場合、通信制御スレッド２０１による受信処理の遅延は０．７７μｓｅｃとなる。この場合、通信制御スレッド２０１が共通資源などを用いて処理を行うため、アプリスレッド２００によるアプリケーション処理の性能が低下する。

【００４９】
また、通信制御スレッド２０１が通常モードでアプリスレッド２００が省消費資源モードの場合、通信制御スレッドによる受信処理の遅延は０．７３μｓｅｃである。すなわち、両方のスレッドが通常モードである場合に比べて、アプリスレッド２００を省消費資源モードにすると、受信処理の遅延が短くなる。このことから、アプリスレッド２００が処理を実行中で無いのであれば、省消費資源モードに遷移させた方が、通信制御スレッド２０１による受信処理は速くなる。
【００５０】
通信制御スレッド２０１が省消費資源モードであり、アプリスレッド２００が通常モードの場合、受信処理の遅延は１．０３μｓｅｃとなる。この場合、通信制御スレッド２０１が通常モードに遷移した後に受信処理が行われるため、受信処理は遅延する。ただし、通信制御スレッド２０１による資源の消費が軽減されるため、アプリスレッド２００によるアプリケーションの処理性能を維持できる。
【００５１】
通信制御スレッド２０１及びアプリスレッド２００が共に省消費資源モードの場合、欄４０１に示すように、ＣＰＵコア１０が省電力モードに入ってしまう。この場合、受信処理の遅延は、１．８９μｓｅｃとなる。この場合、ＣＰＵコア１０の省電力モードを解除した後、通信制御スレッド２０１が通常モードになって受信処理を開始するため、受信処理の遅延が大きくなってしまう。特に通信制御スレッド２０１が省消費資源モードでアプリスレッド２００が通常モードの場合と比較しても、受信処理の遅延は非常に大きい。そこで、通信制御スレッド２０１による受信処理の遅延を軽減するために、通信制御スレッド２０１又はアプリスレッド２００の少なくとも一方は省消費資源モードに遷移しないようにしておくことが望ましい。
【００５２】
そこで、図５に示す各省消費資源モードにおけるアプリケーション処理及び受信処理の関係から、例えば、アプリスレッド２００及び通信制御スレッド２０１は、図６のテーブル５００を満たすように、それぞれの動作モードに応じて消費資源モードを変更する。図６は、実施例１に係る各スレッドの消費資源モードの設定の一例を説明するための図である。
【００５３】
まず、テーブル５００におけるグレーアウトされている欄５０１～５０５は、対応するスレッドが処理を行っている状態であるため、そのスレッドの消費資源モードが通常モードに設定される場合を表している。例えば、アプリスレッド２００がアプリケーション処理を実行している場合、欄５０１～５０３で示されるように、アプリスレッド２００は、通常モードに設定される。また、通信制御スレッド２０１が通信制御処理を行っている場合、欄５０４及び５０５で示されるように、通信制御スレッド２０１は、通常モードに設定される。
【００５４】
そして、アプリスレッド２００又は通信制御スレッド２０１のいずれか一方が通常モードの場合、他方が省消費資源モードに遷移できるならば、他方を省消費資源モードに設定することが好ましい。まず、欄５０３及び５０４の場合、いずれのスレッドも省消費資源モードに遷移できないので、この場合は、両方のスレッドを通常モードで動作させる。一方、通信制御スレッド２０１は、通信制御処理を実行していない場合（以下では、「Ｉｄｌｅ」という。）及びポーリングをしている場合（以下では、「ｍｐｏｌｌ」という。）には、省消費資源モードへ遷移することができる。そこで、アプリスレッド２００がアプリケーション処理を行っている場合で、通信制御スレッド２０１がｍｐｏｌｌ又はＩｄｅｌの場合には、通信制御スレッド２０１は、省消費資源モードに設定される。
【００５５】
また、アプリスレッド２００がアプリケーション処理を実行していない場合（以下では、「Ｉｄｌｅ」という。）、アプリスレッド２００は、省消費資源モードに遷移できる。

そこで、通信制御スレッド２０１が通信制御処理を行っており、アプリスレッド２００がＩｄｌｅの場合、アプリスレッド２００は省消費資源モードに設定される。

【００５６】

また、アプリスレッド２００がＩｄｌｅで、通信制御スレッド２０１がＩｄｌｅの場合、アプリスレッド２００は通常モードに設定され、通信制御スレッド２０１は省消費資源モードに設定される。この場合、通信制御スレッド２０１は、Ｉｄｌｅであり、通信制御処理を行わないので、省消費資源モードであっても問題はない。一方、アプリスレッド２００もＩｄｌｅであるので、本来であれば、消費資源モードであってもよい。しかし、アプリスレッド２００も省消費資源モードに遷移してしまうと、ＣＰＵコア１０が省電力モードに入ってしまう。それを避けるために、アプリスレッド２００は、通常モードに設定されている。このように、アプリスレッド２００が通常モードであるので、ＣＰＵコア１０は、アクティブの状態を維持できる。

【００５７】

次に、アプリスレッド２００がＩｄｌｅで、通信制御スレッド２０１がｍｐｏｌｌの場合、アプリスレッド２００は省消費資源モードに設定され、通信制御スレッド２０１は通常モードに設定される。この場合、アプリスレッド２００は、Ｉｄｌｅであり、アプリケーション処理を行わないので、省消費資源モードであっても問題はない。一方、通信制御スレッド２０１は、ｍｐｏｌｌの状態であるので、通常モード及び省消費資源モードのいずれでもよい。ただし、通信制御スレッド２０１が省消費資源モードに遷移してしまうと、ＣＰＵコア１０が省電力モードに入ってしまう。それを避けるために、通信制御スレッド２０１は、通常モードに設定されている。このように、通信制御スレッド２０１が通常モードであるので、ＣＰＵコア１０は、アクティブの状態を維持できる。

【００５８】

そして、図６で表される消費資源モードの状態を実現するため、アプリスレッド２００及び通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、以下のような処理を行う。すなわち、アプリスレッド２００及び通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、動作モードが通信制御処理を実行するモード又はｍｐｏｌｌからＩｄｌｅの状態に遷移した場合、ＯＳを利用してＩｄｌｅルーチンを走らせて、消費資源モードの切替えを行う。また、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、動作モードが通信制御処理を実行するモード又はＩｄｌｅの状態からｍｐｏｌｌの状態に遷移した場合、ＯＳを利用してｍｐｏｌｌルーチンを走らせて、消費資源モードの切替えを行う。

【００５９】

ここで、Ｉｄｌｅルーチン実行時のアプリスレッド２００を実行するＣＰＵコア１０又は実行部の動作を説明する。アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、Ｒｅａｄｙ状態のアプリケーション処理のプロセスがあるか否かを判定する。これは、Ｉｄｌｅに切り替わりＩｄｌｅルーチンを走らせる間に、アプリケーション処理のプロセスがＲｅａｄｙ状態になることが考えられるからである。そして、Ｒｅａｄｙ状態のプロセスがある場合、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、通信制御スレッド２０１に割り当てられたメモリ領域１２２を更新する。これにより、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、動作モードのＩｄｌｅからアプリケーション処理を実行するモードへの切り替わりを、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部に対して通知する。その後、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、スケジューラを呼び出し、アプリケーション処理のプロセスの実行を待つ。

【００６０】

一方、Ｒｅａｄｙ状態のプロセスがない場合、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、Ｉｄｌｅルーチン以前に、自己の動作モードがＩｄｌｅであったか否かを判定する。これは、後述するように、Ｉｄｌｅの状態のアプリスレッド２００が、割り込み又は通信制御スレッド２０１の動作モードの変化が発生することによりＩｄｌ

ｅルーチンを繰り返すためである。そして、以前の動作モードがＩｄｌｅでなかった場合、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、通信制御スレッド２０１に割り当てられたメモリ領域１２２を更新し、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部に対して、自己の動作モードが変化したことを通知する。その後、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、通信制御スレッド２０１の動作モードがＩｄｌｅか否かを判定する。通信制御スレッド２０１の動作モードがＩｄｌｅの場合、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、自己の消費資源モードを通常モードに設定する。これに対して、通信制御スレッド２０１の動作モードがＩｄｌｅでない、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、自己の消費資源モードを省消費資源モードに設定する。

【００６１】

その後、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、通信制御スレッド２０１の動作モードの変化により自己に割り当てられたメモリ領域１２１が更新される又は割り込みが発生するまで待機する。メモリ領域１２１が更新される又は割り込みが発生すると、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、Ｉｄｌｅルーチンを繰り返す。

【００６２】

また、アプリスレッド２００を実行するＣＰＵコア１０又は実行部は、アプリケーション処理を実行するモードに遷移した場合、消費資源モードを通常モードに設定する。

【００６３】

次に、Ｉｄｌｅルーチン実行時の通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部の動作を説明する。通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、Ｒｅａｄｙ状態の通信制御処理のプロセスがあるか否かを判定する。これは、Ｉｄｌｅに切り替わりＩｄｌｅルーチンを走らせる間に、通信制御処理のプロセスがＲｅａｄｙ状態になることが考えられるからである。そして、Ｒｅａｄｙ状態のプロセスがある場合、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、アプリスレッド２００に割り当てられたメモリ領域１２１を更新する。これにより、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、動作モードのＩｄｌｅからアプリケーション処理を実行するモードへの切り替わりを、アプリスレッド２００を実行するＣＰＵコア１０又は実行部へ通知する。その後、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、スケジューラを呼び出し、通信制御処理のプロセスの実行を待つ。

【００６４】

一方、Ｒｅａｄｙ状態のプロセスがない場合、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、Ｉｄｌｅルーチン以前に、自己の動作モードがＩｄｌｅであったか否かを判定する。これは、Ｉｄｌｅの状態の通信制御スレッド２０１が、割り込み又はアプリスレッド２００の動作モードの変化が発生することによりＩｄｌｅルーチンを繰り返すためである。そして、以前の動作モードがＩｄｌｅでなかった場合、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、アプリスレッド２００に割り当てられたメモリ領域１２１を更新し、アプリスレッド２００を実行するＣＰＵコア１０又は実行部に対して、自己の動作モードが変化したことを通知する。その後、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、自己の消費資源モードを省消費資源モードに設定する。

【００６５】

その後、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、アプリスレッド２００の動作モードの変化により自己に割り当てられたメモリ領域１２２が更新される又は割り込みが発生するまで待機する。メモリ領域１２２が更新される又は割り込みが発生すると、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、Ｉｄｌｅルーチンを繰り返す。

【００６６】

次に、ｍｐｏｌｌルーチン実行時の通信制御スレッド２０１を実行するＣＰＵコア１０

又は実行部の動作を説明する。通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、ＣＰＵコア２０又は２１からデータが到着したか否かを判定する。これは、ｍｐｏｌｌに切り替わりｍｐｏｌｌルーチンを走らせる間に、データを受信することが考えられるからである。そして、データが到着している場合、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、アプリスレッド２００に割り当てられたメモリ領域１２１を更新して、動作モードがｍｐｏｌｌから通信制御処理を実行するモードに切り替わったことを、アプリスレッド２００を実行するＣＰＵコア１０又は実行部へ通知する。その後、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、データの受信処理を実施する。

【００６７】

一方、データが到着していない場合、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、ｍｐｏｌｌルーチン以前に、自己の動作モードがｍｐｏｌｌであったか否かを判定する。これは、ｍｐｏｌｌの状態の通信制御スレッド２０１が、割り込み又はアプリスレッド２００の動作モードの変化が発生することによりｍｐｏｌｌルーチンを繰り返すためである。そして、以前の動作モードがｍｐｏｌｌでなかった場合、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、アプリスレッド２００に割り当てられたメモリ領域１２１を更新し、アプリスレッド２００を実行するＣＰＵコア１０又は実行部に対して、自己の動作モードが変化したことを通知する。その後、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、自己の消費資源モードを省消費資源モードに設定する。

【００６８】

その後、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、アプリスレッド２００の動作モードの変化により自己に割り当てられたメモリ領域１２１が更新される又は割り込みが発生するまで待機する。メモリ領域１２１が更新される又は割り込みが発生すると、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、Ｉｄｌｅルーチンを繰り返す。

【００６９】

また、通信制御スレッド２０１を実行するＣＰＵコア１０又は実行部は、通信制御処理を実行するモードに遷移した場合、消費資源モードを通常モードに設定する。

【００７０】

メモリコントローラ１５は、メモリ１２に対するデータの書き込み及び読み出しを行う。例えば、ＣＰＵコア２０からデータがＣＰＵ１０に向けて送られてくると、メモリコントローラ１５は、メモリ１２の通信制御スレッド２０１に割り当てられたメモリ領域１２２を更新する。メモリコントローラ１５によるメモリ領域１２２の更新により、ポーリング中の通信制御スレッド２０１は、データの受信を検知し、データの受信処理を開始する。さらに、メモリコントローラ１５は、通信制御スレッド２０１からの指示を受けて、アプリスレッド２００に割り当てられたメモリ領域１２１を更新する。また、メモリコントローラ１５は、アプリスレッド２００からの指示を受けて、通信制御スレッド２０１に割り当てられたメモリ領域１２２を更新する。

【００７１】

次に、図７を参照し、本実施例に係るアプリスレッド２００のＩｄｌｅルーチンにおける動作について説明する。図７は、実施例１に係るアプリスレッドのＩｄｌｅルーチンにおける動作のフローチャートである。ここでは、説明を分かり易くするために、各スレッドが処理を行っているように説明する。

【００７２】

アプリスレッド２００は、Ｒｅａｄｙ状態のアプリケーション処理のプロセスがあるか否かを判定する（ステップＳ１０１）。

【００７３】

Ｒｅａｄｙ状態のプロセスがない場合（ステップＳ１０１：否定）、アプリスレッド２００は、今回のＩｄｌｅルーチン以前に、自己の動作モードがＩｄｌｅであったか否かを

判定する（ステップＳ１０２）。そして、以前の動作モードがＩｄｌｅでなかった場合（ステップＳ１０２：否定）、アプリスレッド２００は、通信制御スレッド２０１に対して、自己の動作モードが変化したことを通知する（ステップＳ１０３）。具体的には、アプリスレッド２００は、メモリ領域１２１を更新することで通信制御スレッド２０１への通知を行う。これに対して、以前の動作モードがＩｄｌｅであった場合（ステップＳ１０２：肯定）、アプリスレッド２００は、ステップＳ１０４へ進む。

【００７４】

アプリスレッド２００は、通信制御スレッド２０１の動作モードがＩｄｌｅか否かを判定する（ステップＳ１０４）。通信制御スレッド２０１の動作モードがＩｄｌｅの場合（ステップＳ１０４：肯定）、アプリスレッド２００は、自己の消費資源モードを通常モードに設定する（ステップＳ１０５）。これに対して、通信制御スレッド２０１の動作モードがＩｄｌｅでない場合（ステップＳ１０４：否定）、アプリスレッド２００は、自己の消費資源モードを省消費資源モードに設定する（ステップＳ１０６）。

【００７５】

その後、アプリスレッド２００は、通信制御スレッド２０１の動作モードの変化により自己に割り当てられたメモリ領域１２１が更新されたか又は割り込みが発生したかを判定する（ステップＳ１０７）。メモリ領域１２１の更新及び割り込みのいずれも発生していない場合（ステップＳ１０７：否定）、アプリスレッド２００は、メモリ領域１２１が更新される又は割り込みが発生するまで待機する。これに対して、メモリ領域１２１の更新又は割り込みが発生した場合（ステップＳ１０７：肯定）、アプリスレッド２００は、ステップＳ１０１に戻る。

【００７６】

一方、Ｒｅａｄｙ状態のプロセスがある場合（ステップＳ１０１：肯定）、アプリスレッド２００は、動作モードがＩｄｌｅからアプリケーション処理を実行するモードに切り替わったことを通信制御スレッド２０１へ通知する（ステップＳ１０８）。具体的には、アプリスレッド２００は、メモリ領域１２２を更新することで通信制御スレッド２０１への通知を行う。その後、アプリスレッド２００は、スケジューラを呼び出し（ステップＳ１０９）、アプリケーション処理のプロセスの実行を待つ。

【００７７】

次に、図８を参照し、本実施例に係る通信制御スレッド２０１のＩｄｌｅルーチンにおける動作について説明する。図８は、実施例１に係る通信制御スレッドのＩｄｌｅルーチンにおける動作のフローチャートである。ここでは、説明を分かり易くするために、各スレッドが処理を行っているように説明する。

【００７８】

通信制御スレッド２０１は、Ｒｅａｄｙ状態の通信制御処理のプロセスがあるか否かを判定する（ステップＳ２０１）。

【００７９】

Ｒｅａｄｙ状態のプロセスがない場合（ステップＳ２０１：否定）、通信制御スレッド２０１は、今回のＩｄｌｅルーチン以前に、自己の動作モードがＩｄｌｅであったか否かを判定する（ステップＳ２０２）。そして、以前の動作モードがＩｄｌｅでなかった場合（ステップＳ２０２：否定）、通信制御スレッド２０１は、メモリ領域１２１を更新し、アプリスレッド２００に対して自己の動作モードが変化したことを通知する（ステップＳ２０３）。これに対して、以前の動作モードがＩｄｌｅであった場合（ステップＳ２０２：肯定）、通信制御スレッド２０１は、ステップＳ２０４へ進む。

【００８０】

通信制御スレッド２０１は、自己の消費資源モードを省消費資源モードに設定する（ステップＳ２０４）。

【００８１】

その後、通信制御スレッド２０１は、アプリスレッド２００の動作モードの変化により自己に割り当てられたメモリ領域１２２が更新されたか又は割り込みが発生したかを判定

する（ステップＳ２０５）。メモリ領域１２２の更新及び割り込みのいずれも発生していない場合（ステップＳ２０５：否定）、通信制御スレッド２０１は、メモリ領域１２２が更新される又は割り込みが発生するまで待機する。これに対して、メモリ領域１２２の更新又は割り込みが発生した場合（ステップＳ２０５：肯定）、通信制御スレッド２０１は、ステップＳ２０１に戻る。

【００８２】

一方、Ｒｅａｄｙ状態のプロセスがある場合（ステップＳ２０１：肯定）、通信制御スレッド２０１は、動作モードがＩｄｌｅから通信制御処理を実行するモードに切り替わったことをアプリスレッド２００へ通知する（ステップＳ２０６）。具体的には、通信制御スレッド２０１は、メモリ領域１２１を更新することでアプリスレッド２００への通知を行う。その後、通信制御スレッド２０１は、スケジューラを呼び出し（ステップＳ２０７）、通信制御処理のプロセスの実行を待つ。

【００８３】

次に、図９を参照し、本実施例に係る通信制御スレッド２０１のｍｐｏｌｌルーチンにおける動作について説明する。図９は、実施例１に係る通信制御スレッドのｍｐｏｌｌルーチンにおける動作のフローチャートである。ここでは、説明を分かり易くするために、各スレッドが処理を行っているように説明する。

【００８４】

通信制御スレッド２０１は、ＣＰＵコア２０又は２１からのデータが到着しているか否かを判定する（ステップＳ３０１）。

【００８５】

データが到着していない場合（ステップＳ３０１：否定）、通信制御スレッド２０１は、今回のｍｐｏｌｌルーチン以前に、自己の動作モードがｍｐｏｌｌであったか否かを判定する（ステップＳ３０２）。そして、以前の動作モードがｍｐｏｌｌでなかった場合（ステップＳ３０２：否定）、通信制御スレッド２０１は、メモリ領域１２１を更新し、アプリスレッド２００に対して自己の動作モードが変化したことを通知する（ステップＳ３０３）。これに対して、以前の動作モードがｍｐｏｌｌであった場合（ステップＳ３０２：肯定）、通信制御スレッド２０１は、ステップＳ３０４へ進む。

【００８６】

通信制御スレッド２０１は、アプリスレッド２００の動作モードがＩｄｌｅか否かを判定する（ステップＳ３０４）。アプリスレッド２００の動作モードがＩｄｌｅの場合（ステップＳ３０４：肯定）、通信制御スレッド２０１は、自己の消費資源モードを通常モードに設定する（ステップＳ３０５）。これに対して、アプリスレッド２００の動作モードがＩｄｌｅでない場合（ステップＳ３０４：否定）、通信制御スレッド２０１は、自己の消費資源モードを省消費資源モードに設定する（ステップＳ３０６）。

【００８７】

その後、通信制御スレッド２０１は、アプリスレッド２００の動作モードの変化により自己に割り当てられたメモリ領域１２２が更新されたか又は割り込みが発生したかを判定する（ステップＳ３０７）。メモリ領域１２２の更新及び割り込みのいずれも発生していない場合（ステップＳ３０７：否定）、通信制御スレッド２０１は、メモリ領域１２２が更新される又は割り込みが発生するまで待機する。これに対して、メモリ領域１２２の更新又は割り込みが発生した場合（ステップＳ３０７：肯定）、通信制御スレッド２０１は、ステップＳ３０１に戻る。

【００８８】

一方、Ｒｅａｄｙ状態のプロセスがある場合（ステップＳ３０１：肯定）、通信制御スレッド２０１は、動作モードがＩｄｌｅから通信制御処理を実行するモードに切り替わったことをアプリスレッド２００へ通知する（ステップＳ３０８）。具体的には、通信制御スレッド２０１は、メモリ領域１２１を更新することでアプリスレッド２００への通知を行う。その後、通信制御スレッド２０１は、受信処理を実施する（ステップＳ３０９）。

【００８９】

次に、図１０を参照して、各スレッドの動作モード及び消費資源モードの遷移の一例について説明する。図１０は、実施例１に係る各スレッドの動作モード及び消費資源モードの遷移の一例を示すタイムチャートである。図１０の帯グラフ６０１はアプリスレッド２００の動作モードの遷移を表している。また、図１０の帯グラフ６０２は通信制御スレッド２０１の動作モードの遷移を表している。また、図１０の帯グラフ６０３はＣＰＵコア１０の省電力モードの遷移を表している。さらに、図１０の最下部の矢印は時間経過を表している。そして、帯グラフ６０１及び６０２におけるグレーアウトしている部分は、それぞれのスレッドの消費資源モードが通常モードであることを表している。また、帯グラフ６０３におけるグレーアウトしている部分は、ＣＰＵコア１０が省電力モードに入っていない状態、すなわちアクティブな状態を表している。ここでは、説明を分かり易くするために、各スレッドが処理を行っているように説明する。

【００９０】

時刻ｔ１までは、アプリスレッド２００の動作モードは、アプリケーション処理を行っている状態である。この状態では、アプリスレッド２００の省電力モードは、通常モードである。また、通信制御スレッド２０１の動作モードは、ｍｐｏｌｌの状態である。この状態では、通信制御スレッド２０１の省電力モードは、省消費資源モードである。この場合、アプリスレッド２００が通常モードであるので、ＣＰＵコア１０の省電力モードはアクティブである。

【００９１】

そして、時刻ｔ１でアプリスレッド２００におけるアプリケーション処理が終了する。これにより、アプリスレッド２００の動作モードはＩｄｌｅに遷移する。そして、アプリスレッド２００は、通信制御スレッド２０１がｍｐｏｌｌなので、自己の消費資源モードを省消費資源モードに設定する。通信制御スレッド２０１は、アプリスレッド２００がＩｄｌｅであり、自己がｍｐｏｌｌなので、自己の消費資源モードを通常モードに設定する。この場合、通信制御スレッド２０１が通常モードであるので、ＣＰＵコア１０の省電力モードはアクティブである。

【００９２】

次に、時刻ｔ２でＣＰＵコア２０又は２１からデータが到着する。これにより、通信制御スレッド２０１の動作モードは、受信処理などの通信制御処理を実行するモードへ遷移する。通信制御スレッド２０１の消費資源モードは既に通常モードであるので変化しない。また、アプリスレッド２００の消費資源モードも、省消費資源モードから変化しない。この場合、通信制御スレッド２０１が通常モードであるので、ＣＰＵコア１０の省電力モードはアクティブである。

【００９３】

次に、時刻ｔ３で通信制御処理が終了する、データ到着を待つため通信制御スレッド２０１の動作モードはｍｐｏｌｌに遷移する。ここで、アプリスレッド２００の動作モードがＩｄｌｅであるので、通信制御スレッド２０１は、省消費資源モードを通常モードのまま維持する。また、アプリスレッド２００も、消費資源モードを省消費資源モードのまま維持する。この場合、通信制御スレッド２０１が通常モードであるので、ＣＰＵコア１０の省電力モードはアクティブである。

【００９４】

次に、時刻ｔ４でアプリスレッド２００がアプリケーション処理を開始する。これにより、アプリスレッド２００は、消費資源モードを通常モードに設定する。これを受けて、動作モードがｍｐｏｌｌである通信制御スレッド２０１は、消費資源モードを省消費資源モードに設定する。この場合、アプリスレッド２００が通常モードであるので、ＣＰＵコア１０の省電力モードはアクティブである。

【００９５】

次に、時刻ｔ５でＣＰＵコア２０又は２１からデータが到着する。これにより、通信制御スレッド２０１の動作モードは、受信処理などの通信制御処理を実行するモードへ遷移する。そして、通信制御スレッド２０１は、消費資源モードを通常モードに設定する。こ

の時、アプリスレッド２００は、アプリケーション処理を実行しているので、消費資源モードを通常モードのまま維持する。この場合、アプリスレッド２００及び通信制御スレッドともに通常モードであるので、ＣＰＵコア１０の省電力モードはアクティブである。

【００９６】

次に、時刻ｔ６でアプリケーション処理が終了し、アプリスレッド２００の動作モードはＩｄｌｅに遷移する。この時、通信制御スレッド２０１が通信制御処理を実行中であるので、アプリスレッド２００は、消費資源モードを省消費資源モードに設定する。通信制御スレッド２０１は、消費資源モードを通常モードのまま維持する。この場合、通信制御スレッド２０１が通常モードであるので、ＣＰＵコア１０の省電力モードはアクティブである。

【００９７】

そして、時刻ｔ７で通信制御処理が終了し、通信制御スレッド２０１の動作モードはＩｄｌｅに遷移する。通信制御スレッド２０１は、アプリスレッド２００の動作モードがＩｄｌｅであるので、消費資源モードを省消費資源モードに設定する。アプリスレッド２００は、通信制御スレッド２０１の動作モードがＩｄｌｅであるので、消費資源モードを通常モードに設定する。この場合、アプリスレッド２００が通常モードであるので、ＣＰＵコア１０の省電力モードはアクティブである。

【００９８】

このように、アプリスレッド２００及び通信制御スレッド２０１のいずれか片方をなるべく省消費資源モードにすることで、通信制御処理又はアプリケーション処理における処理速度を向上させることができる。また、ＣＰＵコア１０は、常にアクティブとなり、省電力モードに遷移しないため、通信制御スレッド２０１の受信処理の遅延を軽減することができる。

【００９９】

図１１は、ポーリングによるアプリケーション処理への影響を説明するための図である。図１１は縦軸で時間を表し、横軸でアプリケーション処理に対応するベンチマークの種類を表している。横軸の各ベンチマークにおける左端のグラフがポーリング処理を行わない場合の各ベンチマークの処理時間を表している。また、横軸の各ベンチマークにおける中央のグラフが単純なビジーポーリングを行った場合の、各ベンチマークの処理時間を表している。単純なビジーポーリングとは、アプリケーション処理が実行されている場合にポーリングを数クロック停止するポーズ命令を使用してポーリングを行った場合である。また、横軸の各ベンチマークにおける右端のグラフが省消費資源モードでポーリング処理を行った場合の、各ベンチマークの処理時間を表している。そして、縦軸の時間は、ポーリング処理を行わない場合の各ベンチマークの処理時間により各処理時間を正規化した値である。

【０１００】

図１１に示すように、単純なビジーポーリングを行った場合には、アプリケーション処理の処理時間が非常に長くなり、アプリケーション処理の性能が低下している。これに対して、省消費資源モードでポーリングを行った場合には、単純なビジーポーリングを行った場合に比較していずれのベンチマークにおいても処理時間が短くなっている。すなわち、本実施例のように、アプリスレッド２００がアプリケーション処理を実行している場合にポーリングを行うのであれば、通信制御スレッド２０１を省消費資源モードにすることで、アプリケーション処理の性能低下を軽減することができる。

【０１０１】

以上に説明したように、本実施例に係る情報処理装置は、他のスレッドが処理を実行している場合、ポーリングを行うスレッドを実行するＣＰＵコア又は実行部は省消費電力モードに遷移する。これにより、あるスレッドがポーリングを行っている場合に、他のスレッドが実行している処理の速度を向上させることができる。

【０１０２】

また、常にいずれかのスレッドを実行するＣＰＵコア又は実行部の消費資源モードを通

常モードにしておくことで、ＣＰＵコアが省電力モードに遷移することを防止でき、受信処理における遅延を軽減することができる。

【０１０３】

さらに、あるスレッドを実行するＣＰＵコア又は実行部が処理を実行している場合、なるべく他のスレッドを実行するＣＰＵコア又は実行部を省消費資源モードに遷移させることで、その処理の実行速度を向上させることができる。例えば、通信制御スレッドを実行するＣＰＵコア又は実行部がポーリングを行っている場合に、アプリスレッドを実行するＣＰＵコア又は実行部をなるべく省消費資源モードに遷移させることで、受信処理の遅延を軽減できる。

【０１０４】

このように、本実施例に係る情報処理装置は、高い通信性能を維持しつつアプリケーションの処理性能の低下を抑制することができる。

【実施例２】

【０１０５】

次に、実施例２について説明する。本実施例に係る情報処理装置は、アプリスレッドを実行するＣＰＵコア又は実行部の消費資源モードを常に通常モードにしておくことが実施例１と異なる。そこで、以下では、アプリスレッド及び通信制御スレッドを実行するＣＰＵコア又は実行部の消費資源モードの遷移について主に説明する。ここでは、実施例１と同じ機能を有する各部については説明を省略する。

【０１０６】

図１２は、実施例２に係る各スレッドの消費資源モードの設定の一例を説明するための図である。アプリスレッド２００及び通信制御スレッド２０１を実行するＣＰＵコア又は実行部は、図１２のテーブル５１０を満たすように、それぞれの動作モードに応じて消費資源モードを変更する。

【０１０７】

まず、テーブル５１０におけるグレーアウトされている欄５１１～５１５は、対応するスレッドが処理を行っている状態であるため、そのスレッドの消費資源モードが通常モードに設定される場合を表している。

【０１０８】

さらに、本実施例の場合、欄５１６～５１８に示すように、アプリスレッド２００がＩｄｌｅであっても、通信制御スレッド２０１の動作モードにかかわらず、アプリスレッド２００を実行するＣＰＵコア又は実行部は、消費資源モードを通常モードに設定する。このように、アプリスレッド２００を実行するＣＰＵコア又は実行部を常に通常動作モードにすることで、ＣＰＵコア１０は、省電力モードに遷移することはなくなり、常にアクティブとなる。

【０１０９】

そこで、通信制御スレッド２０１を実行するＣＰＵコア又は実行部は、ＣＰＵコア１０の省電力モードへの遷移を気にせずに、省消費資源モードに遷移できる場合には、消費資源モードを省消費資源モードに設定することができる。すなわち、通信制御スレッド２０１を実行するＣＰＵコア又は実行部は、動作モードがＩｄｌｅ又はｍｐｏｌｌの場合には、消費資源モードを省消費資源モードに設定する。

【０１１０】

そして、アプリスレッド２００及び通信制御スレッド２０１を実行するＣＰＵコア又は実行部は、動作モードが通信制御処理を実行するモード又はｍｐｏｌｌからＩｄｌｅの状態に遷移した場合、ＯＳを利用してＩｄｌｅルーチンを走らせて、消費資源モードの切替えを行う。また、通信制御スレッド２０１を実行するＣＰＵコア又は実行部は、動作モードが通信制御処理を実行するモード又はＩｄｌｅの状態からｍｐｏｌｌの状態に遷移した場合、ＯＳを利用してｍｐｏｌｌルーチンを走らせて、消費資源モードの切替えを行う。これにより、図１２で表される消費資源モードの状態が実現される。

【０１１１】

次に、図１３を参照し、本実施例に係るアプリスレッド２００のＩｄｌｅルーチンにおける動作について説明する。図１３は、実施例２に係るアプリスレッドのＩｄｌｅルーチンにおける動作のフローチャートである。ここでは、説明を分かり易くするために、各スレッドが処理を行っているように説明する。

【０１１２】

アプリスレッド２００は、Ｒｅａｄｙ状態のアプリケーション処理のプロセスがあるか否かを判定する（ステップＳ４０１）。

【０１１３】

Ｒｅａｄｙ状態のプロセスがない場合（ステップＳ４０１：否定）、アプリスレッド２００は、割り込みが発生したか否かを判定する（ステップＳ４０２）。割り込みのいずれも発生していない場合（ステップＳ４０２：否定）、アプリスレッド２００は、割り込みが発生するまで待機する。これに対して、割り込みが発生した場合（ステップＳ４０２：肯定）、アプリスレッド２００は、ステップＳ４０１に戻る。

【０１１４】

一方、Ｒｅａｄｙ状態のプロセスがある場合（ステップＳ４０１：肯定）、アプリスレッド２００は、スケジューラを呼び出し（ステップＳ４０３）、アプリケーション処理のプロセスの実行を待つ。

【０１１５】

次に、図１４を参照し、本実施例に係る通信制御スレッド２０１のＩｄｌｅルーチンにおける動作について説明する。図１４は、実施例２に係る通信制御スレッドのＩｄｌｅルーチンにおける動作のフローチャートである。

【０１１６】

通信制御スレッド２０１は、Ｒｅａｄｙ状態の通信制御処理のプロセスがあるか否かを判定する（ステップＳ５０１）。

【０１１７】

Ｒｅａｄｙ状態のプロセスがない場合（ステップＳ５０１：否定）、通信制御スレッド２０１は、自己の消費資源モードを省消費資源モードに設定する（ステップＳ５０２）。

【０１１８】

その後、通信制御スレッド２０１は、自己に割り当てられたメモリ領域１２２が更新されたか又は割り込みが発生したかを判定する（ステップＳ５０３）。メモリ領域１２２の更新及び割り込みのいずれも発生していない場合（ステップＳ５０３：否定）、通信制御スレッド２０１は、メモリ領域１２２が更新される又は割り込みが発生するまで待機する。これに対して、メモリ領域１２２の更新又は割り込みが発生した場合（ステップＳ５０３：肯定）、通信制御スレッド２０１は、ステップＳ５０１に戻る。

【０１１９】

一方、Ｒｅａｄｙ状態のプロセスがある場合（ステップＳ５０１：肯定）、通信制御スレッド２０１は、スケジューラを呼び出し（ステップＳ５０４）、通信制御処理のプロセスの実行を待つ。

【０１２０】

次に、図１５を参照し、本実施例に係る通信制御スレッド２０１のｍｐｏｌｌルーチンにおける動作について説明する。図１５は、実施例２に係る通信制御スレッドのｍｐｏｌｌルーチンにおける動作のフローチャートである。

【０１２１】

通信制御スレッド２０１は、ＣＰＵコア２０又は２１からのデータが到着しているか否かを判定する（ステップＳ６０１）。

【０１２２】

データが到着していない場合（ステップＳ６０１：否定）、通信制御スレッド２０１は、消費資源モードを省消費資源モードに設定する（ステップＳ６０２）。

【０１２３】

その後、通信制御スレッド２０１は、メモリ領域１２２が更新されたか又は割り込みが

発生したかを判定する（ステップＳ６０３）。メモリ領域１２２の更新及び割り込みのいずれも発生していない場合（ステップＳ６０３：否定）、通信制御スレッド２０１は、メモリ領域１２２が更新される又は割り込みが発生するまで待機する。これに対して、メモリ領域１２２の更新又は割り込みが発生した場合（ステップＳ６０３：肯定）、通信制御スレッド２０１は、ステップＳ６０１に戻る。

【０１２４】

一方、Ｒｅａｄｙ状態のプロセスがある場合（ステップＳ６０１：肯定）、通信制御スレッド２０１は、受信処理を実施する（ステップＳ６０４）。

【０１２５】

次に、図１６を参照して、各スレッドの動作モード及び消費資源モードの遷移の一例について説明する。図１６は、実施例２に係る各スレッドの動作モード及び消費資源モードの遷移の一例を示すタイムチャートである。図１６の帯グラフ６１１はアプリスレッド２００の動作モードの遷移を表している。また、図１６の帯グラフ６１２は通信制御スレッド２０１の動作モードの遷移を表している。また、図１６の帯グラフ６１３はＣＰＵコア１０の省電力モードの遷移を表している。さらに、図１６の最下部の矢印は時間経過を表している。そして、帯グラフ６１１及び６１２におけるグレーアウトしている部分は、それぞれのスレッドの消費資源モードが通常モードであることを表している。また、帯グラフ６１３におけるグレーアウトしている部分は、ＣＰＵコア１０が省電力モードに入っていない状態、すなわちアクティブな状態を表している。ここでは、説明を分かり易くするために、各スレッドが処理を行っているように説明する。

【０１２６】

アプリスレッド２００は、動作モードがどのモードに遷移しても、帯グラフ６１１に示すように消費資源モードを通常モードに維持する。そのため、ＣＰＵコア１０は、帯グラフ６１３に示されるように常にアクティブの状態になっている。

【０１２７】

一方、時刻Ｔ１までは、通信制御スレッド２０１の動作モードは、ｍｐｏｌｌの状態である。この状態では、通信制御スレッド２０１の省電力モードは、省消費資源モードである。

【０１２８】

そして、時刻Ｔ１でアプリスレッド２００におけるアプリケーション処理が終了する。これにより、アプリスレッド２００の動作モードはＩｄｌｅに遷移する。これに対して、通信制御スレッド２０１は、アプリスレッド２００の動作モードに関らず、自己がｍｐｏｌｌなので、自己の消費資源モードを省消費資源モードに維持する。

【０１２９】

次に、時刻Ｔ２でＣＰＵコア２０又は２１からデータが到着する。これにより、通信制御スレッド２０１の動作モードは、受信処理などの通信制御処理を実行するモードへ遷移する。これにより、通信制御スレッド２０１は、消費資源モードを通常モードに設定する。

【０１３０】

次に、時刻Ｔ３で通信制御処理が終了し、データ到着を待つため通信制御スレッド２０１の動作モードはｍｐｏｌｌに遷移する。これにより、通信制御スレッド２０１は、消費資源モードを省消費資源モードに設定する。

【０１３１】

次に、時刻Ｔ４でアプリスレッド２００がアプリケーション処理を開始する。このときも、動作モードがｍｐｏｌｌである通信制御スレッド２０１は、消費資源モードを省消費資源モードのまま維持する。

【０１３２】

次に、時刻Ｔ５でＣＰＵコア２０又は２１からデータが到着する。これにより、通信制御スレッド２０１の動作モードは、受信処理などの通信制御処理を実行するモードへ遷移する。そして、通信制御スレッド２０１は、消費資源モードを通常モードに設定する。

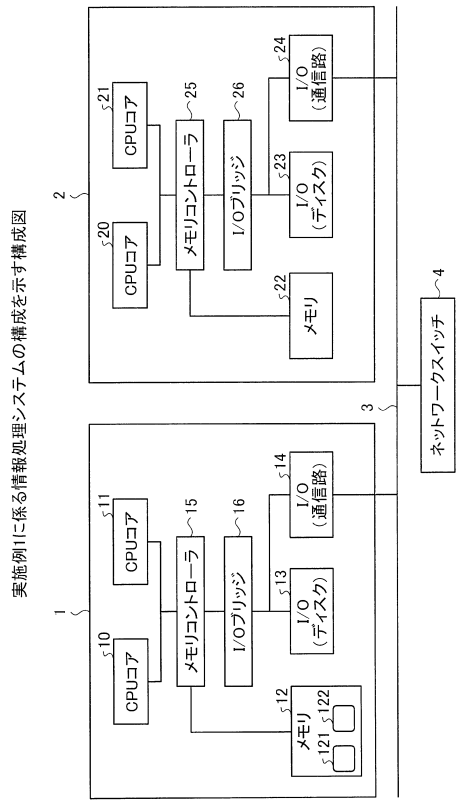
【０１３３】
次に、時刻Ｔ６でアプリケーション処理が終了し、アプリスレッド２００の動作モードはＩｄｌｅに遷移する。これに対して、通信制御スレッド２０１は、通信制御処理を実行中であるので、消費資源モードを通常モードのまま維持する。
【０１３４】
そして、時刻Ｔ７で通信制御処理が終了し、通信制御スレッド２０１の動作モードはＩｄｌｅに遷移する。通信制御スレッド２０１は、アプリスレッド２００の動作モードに関らず、消費資源モードを省消費資源モードに設定する。
【０１３５】
以上に説明したように、本実施例に係る情報処理装置においても、他のスレッドを実行するＣＰＵコア又は実行部が処理を実行している場合、ポーリングを行うスレッドは省消費電力モードに遷移する。これにより、あるスレッドを実行するＣＰＵコア又は実行部がポーリングを行っている場合に、他のスレッドが実行している処理の速度を向上させることができる。また、常にアプリスレッドを実行するＣＰＵコア又は実行部の消費資源モードを通常モードにしておくことで、ＣＰＵコアが省電力モードに遷移することを防止でき、受信処理における遅延を軽減することができる。
【０１３６】
また、アプリスレッドを常に通常モードにし、通信制御スレッドが省消費資源モードに遷移できる場合には、通信制御スレッドを省消費資源モードに設定するので、実施例１に比べて制御が容易になる。
【０１３７】
さらに、以上の実施例では、受信処理の遅延をなるべく軽減するため、ＣＰＵコアを常にアクティブの状態にするように、いずれかのスレッドを実行するＣＰＵコア又は実行部の消費資源モードが通常モードとなるように設定した。ただし、受信処理の遅延の許容できる程度によっては、全てのスレッドを実行するＣＰＵコア又は実行部が省消費資源モードになることを許容して制御してもよい。その場合も、他のスレッドを実行するＣＰＵコア又は実行部が処理を実行している場合、ポーリングを行うスレッドを実行するＣＰＵコア又は実行部は省消費電力モードに遷移する。これにより、あるスレッドを実行するＣＰＵコア又は実行部がポーリングを行っている場合に、他のスレッドを実行するＣＰＵコア又は実行部が実行している処理の速度を向上させるという効果を得ることはできる。
【０１３８】
また、以上では、図１に示すシステム構成を有する情報処理システムを例に説明したが、システム構成はこれ以外の構成を用いることもできる。図１７は、情報処理システムの他の構成を示す構成図である。図１７では、図１におけるメモリコントローラ１５及びＩ／Ｏブリッジ１６が、ＣＰＵコア１０及び１１に内蔵されている。また、メモリコントローラ２５及びＩ／Ｏブリッジ２６が、ＣＰＵコア２０及び２１に内蔵されている。
【０１３９】
この場合、ＣＰＵコア１０には、メモリ１２ａが接続されている。そして、ＣＰＵコア１０に内蔵されたメモリコントローラによりメモリ１２ａに対するデータの読み書きが行われる。また、ＣＰＵコア１１には、メモリ１２ｂが接続されている。そして、ＣＰＵコア１１に内蔵されたメモリコントローラによりメモリ１２ｂに対するデータの読み書きが行われる。
【０１４０】
また、ＣＰＵコア１０及び１１は、内蔵されているＩ／ＯバスによりＩ／Ｏ１３及びＩ／Ｏ１４と接続されており、ＣＰＵコア１０及び１１は、内蔵されているＩ／ＯバスによりＩ／Ｏ１３及びＩ／Ｏ１４との通信を行う。
【０１４１】
このように、ＣＰＵコアにメモリコントローラやＩ／Ｏバスが内蔵されているシステム構成の場合でも、上述した各機能を有することができ、同様の効果を発揮することができる。

【符号の説明】

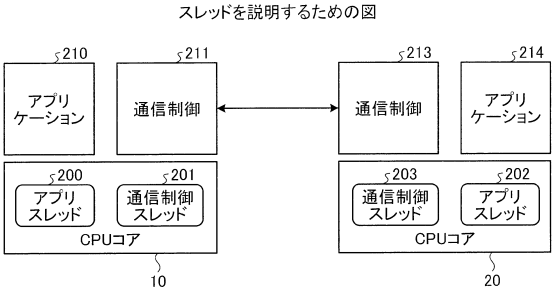
【０１４２】

１，２　サーバ
３　バス
４　ネットワークスイッチ
１０，１１　ＣＰＵコア
１２　メモリ
１３，１４　Ｉ／Ｏ
１５　メモリコントローラ
１６　Ｉ／Ｏブリッジ
２０，２１　ＣＰＵコア
２２　メモリ
２３，２４　Ｉ／Ｏ
２５　メモリコントローラ
２６　Ｉ／Ｏブリッジ
２００，２０２　アプリスレッド
２０１，２０３　通信制御スレッド

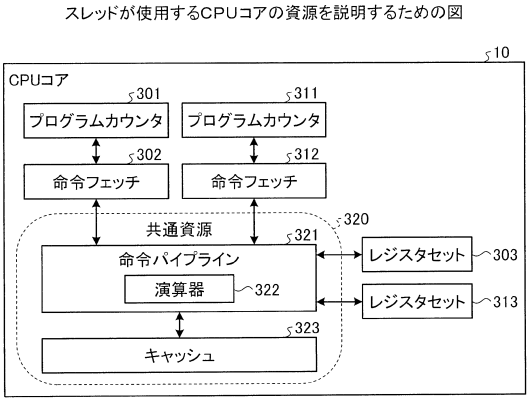
【図１】



【図２】



【図３】



【図４】

消費資源モードとCPUコアの省電力モードとの関係を示す図

	アプリスレッド 通常モード	アプリスレッド 省消費資源モード
通信制御スレッド 通常モード	アクティブ	アクティブ
通信制御スレッド 省消費資源モード	アクティブ	省電力

400

【図５】

各消費資源モードでの受信処理及びアプリケーション処理の関係を説明するための図

通信制御スレッドの消費資源モード	アプリスレッドの消費資源モード	CPUコアの省電力モード	遅延（μsec）	アプリスレッドの処理能力の状態
通常モード	通常モード	アクティブ	0.77	アプリケーション処理の性能低下
	省消費資源モード	アクティブ	0.73	アプリスレッドが通常モードの場合に比べて受信処理の遅延は短い
省消費資源モード	通常モード	アクティブ	1.03	受信処理は遅れるが、アプリケーション処理の性能維持
	省消費資源モード	省電力	1.89	受信処理の遅延大

401

【図６】

実施例1に係る各スレッドの消費資源モードの設定の一例を説明するための図

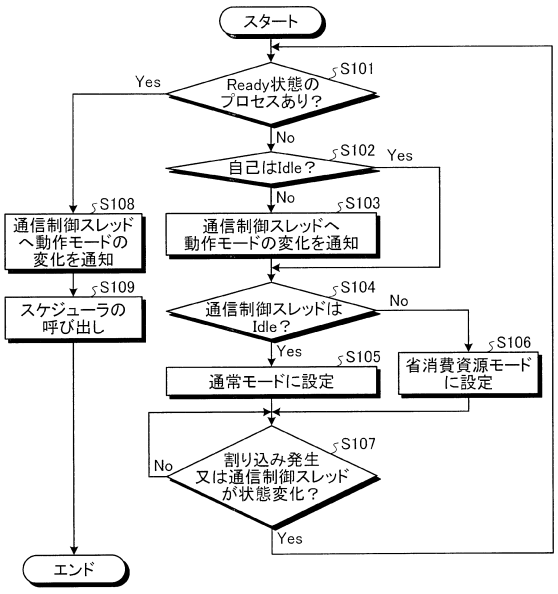
500

アプリスレッドの動作モード	通信制御スレッドの動作モード	アプリスレッドの消費資源モード	通信制御スレッドの消費資源モード	CPUコアの状態
Idle	Idle	通常モード	省消費資源モード	アクティブ
	mpoll	省消費資源モード	通常モード	アクティブ
	通信制御処理	省消費資源モード	通常モード	アクティブ
アプリケーション処理	Idle	通常モード	省消費資源モード	アクティブ
	mpoll	通常モード	省消費資源モード	アクティブ
	通信制御処理	通常モード	通常モード	アクティブ

501 502 503 504 505

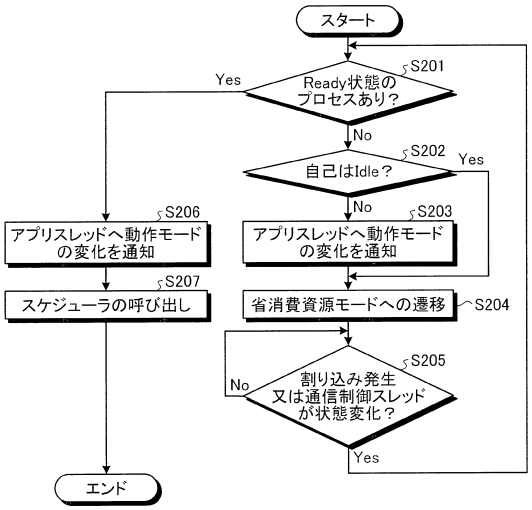
【図７】

実施例1に係るアプリスレッドのIdleルーチンにおける動作のフローチャート



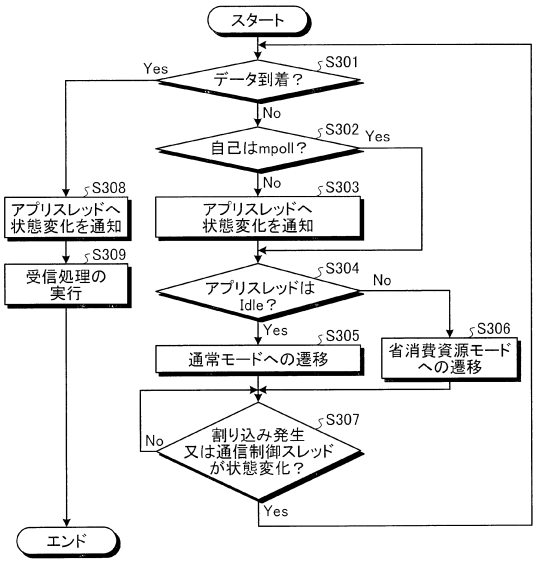
【図８】

実施例1に係る通信制御スレッドのIdleルーチンにおける動作のフローチャート

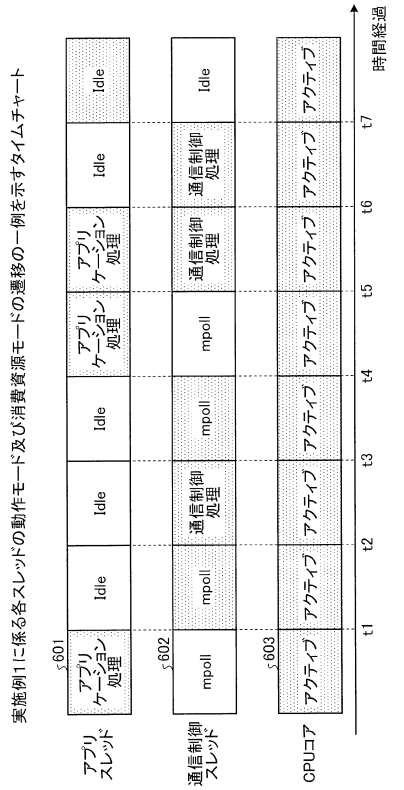


【図９】

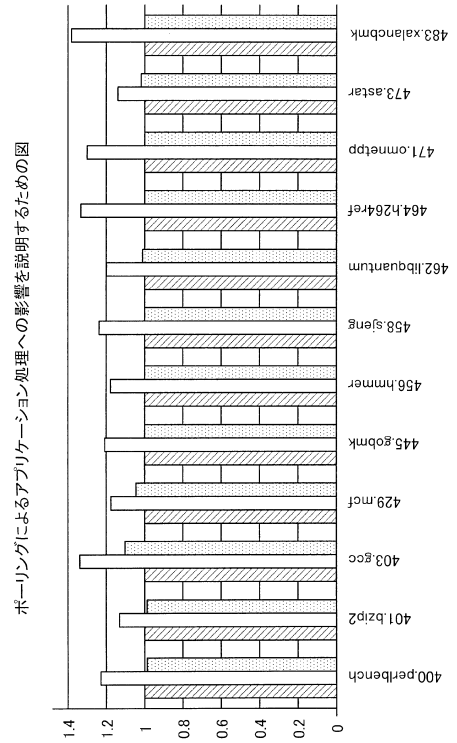
実施例1に係る通信制御スレッドのmpollルーチンにおける動作のフローチャート



【図１０】

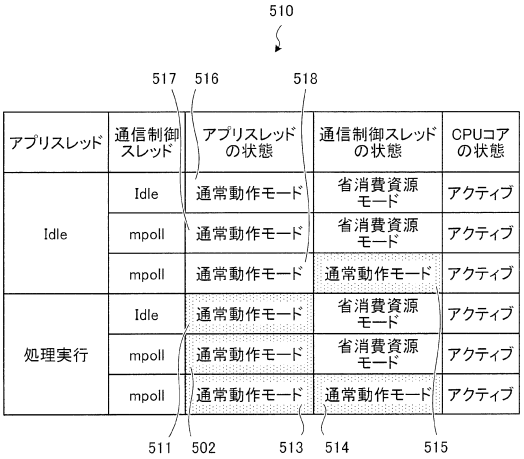


【図１１】



【図１２】

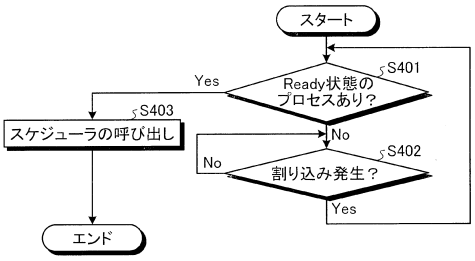
実施例2に係る各スレッドの消費資源モードの設定の一例を説明するための図



アプリスレッド	通信制御スレッド	アプリスレッドの状態	通信制御スレッドの状態	CPUコアの状態
Idle	Idle	通常動作モード	省消費資源モード	アクティブ
	mpoll	通常動作モード	省消費資源モード	アクティブ
	mpoll	通常動作モード	通常動作モード	アクティブ
処理実行	Idle	通常動作モード	省消費資源モード	アクティブ
	mpoll	通常動作モード	省消費資源モード	アクティブ
	mpoll	通常動作モード	通常動作モード	アクティブ

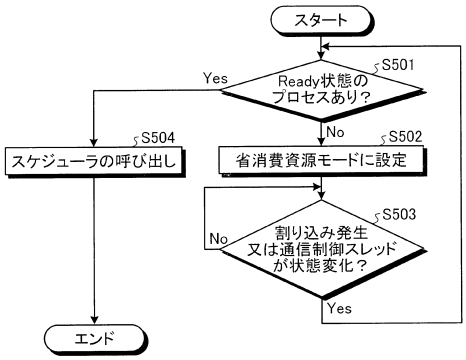
【図１３】

実施例2に係るアプリスレッドのIdleルーチンにおける動作のフローチャート



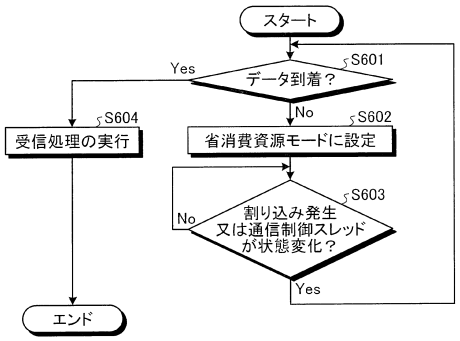
【図１４】

実施例2に係る通信制御スレッドのIdleルーチンにおける動作のフローチャート

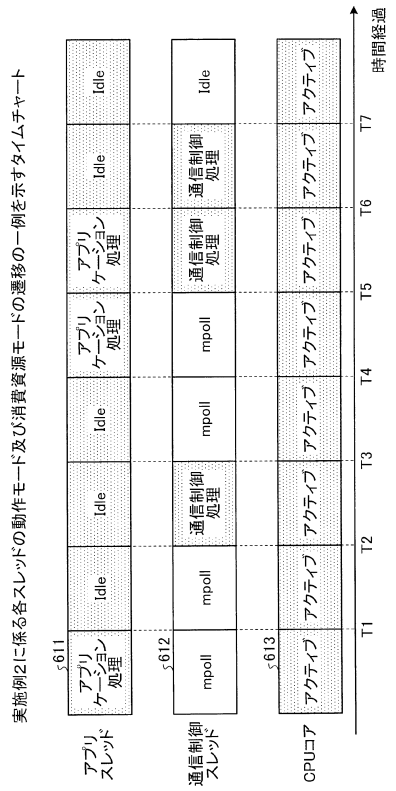


【図１５】

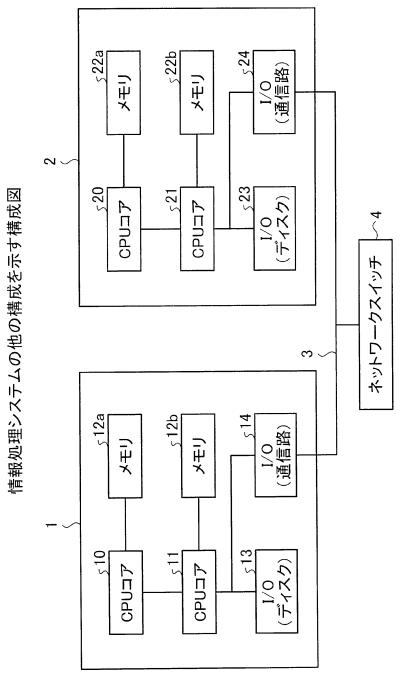
実施例2に係る通信制御スレッドのmpollルーチンにおける動作のフローチャート



【図１６】



【図１７】



フロントページの続き

(56)参考文献 国際公開第０３／０４０９４８（WO，Ａ１）
特表２０１０－５２４０８７（ＪＰ，Ａ）
特開２００８－１２９７６７（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／５０