

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B1)

(11) 特許番号

**特許第3768516号
(P3768516)**

(45) 発行日 平成18年4月19日 (2006. 4. 19)

(24) 登録日 平成18年2月10日 (2006. 2. 10)

(51) Int. Cl.		F I			
G06F	9/42	(2006.01)	G06F	9/42	320A
G06F	9/32	(2006.01)	G06F	9/32	320A
G06T	1/20	(2006.01)	G06T	1/20	B

請求項の数 12 (全 14 頁)

<p>(21) 出願番号 特願2004-350702 (P2004-350702)</p> <p>(22) 出願日 平成16年12月3日 (2004. 12. 3)</p> <p>審査請求日 平成17年11月17日 (2005. 11. 17)</p>	<p>(73) 特許権者 395015319 株式会社ソニー・コンピュータエンタテインメント 東京都港区南青山二丁目6番21号</p> <p>(74) 代理人 100105924 弁理士 森下 賢樹</p> <p>(72) 発明者 野田 慎治 東京都港区南青山2丁目6番21号 株式会社ソニー・コンピュータエンタテインメント内</p> <p>(72) 発明者 河野 健 東京都港区南青山2丁目6番21号 株式会社ソニー・コンピュータエンタテインメント内</p>
--	--

最終頁に続く

(54) 【発明の名称】 マルチプロセッサシステムとそのシステムにおけるプログラム実行方法

(57) 【特許請求の範囲】

【請求項1】

第1のプロセッサと第2のプロセッサを含み、

第1のプロセッサは、実行中のメインルーチンにおいて所定のコール命令を実行したとき第2のプロセッサに対して割込を発生する割込発生部を備え、

第2のプロセッサは、割込発生部から割込を受けたとき、前記コール命令によって呼び出されるサブルーチンの処理が完了したときメインルーチンへ戻るための戻りアドレスを所定のメモリ領域に退避するアドレス退避部を備えることを特徴とするマルチプロセッサシステム。

【請求項2】

請求項1に記載のシステムにおいて、

前記割込発生部は、サブルーチンにおいて所定のリターン命令を実行したとき第2のプロセッサに対して再度割込を発生し、

第2のプロセッサは、再度発生された割込を受けたとき、前記戻りアドレスを第1のプロセッサへ通知するアドレス通知部をさらに備えることを特徴とするマルチプロセッサシステム。

【請求項3】

請求項2に記載のシステムにおいて、第1のプロセッサは、命令をフェッチするフェッチャーを備え、このフェッチャーによるアクセスの対象アドレスとして前記戻りアドレスが設定されることを特徴とするマルチプロセッサシステム。

10

20

【請求項4】

第1のプロセッサと第2のプロセッサを含み、

第1のプロセッサは、所定のコール命令またはジャンプ命令を実行したとき第2のプロセッサに対して割込を発生する割込発生部を備え、

第2のプロセッサは、

第1のプロセッサから割込を受けたとき、前記コール命令またはジャンプ命令と、その後には置かれる実行停止命令のフォーマット中に分割して格納されたコール先アドレスまたはジャンプ先アドレスを抽出するアドレス抽出部と、

取得したコール先アドレスまたはジャンプ先アドレスを第1のプロセッサに通知するアドレス通知部と、

を備えることを特徴とするマルチプロセッサシステム。

10

【請求項5】

請求項4に記載のシステムにおいて、第1のプロセッサは、命令をフェッチするフェッチャーを備え、このフェッチャーによるアクセスの対象アドレスとして前記コール先アドレスまたはジャンプ先アドレスが設定されることを特徴とするマルチプロセッサシステム。

【請求項6】

グラフィックプロセッサとメインプロセッサを含み、

グラフィックプロセッサは、

メモリからディスプレイリストとして記述される命令を順次読み出すDMACと、

読み出された命令を順次解読するデコーダと、

解読された命令がディスプレイリストのメインルーチンに含まれる所定のコール命令であるときメインプロセッサに対して移行用割込を発生し、かつ、解読された命令が前記コール命令によって呼び出されるサブルーチンに含まれるリターン命令であるときメインプロセッサに対して復帰用割込を発生する割込発生部とを備え、

メインプロセッサは、

割込発生部から移行用割込を受けたとき、前記サブルーチンの処理が完了したときメインルーチンへ戻るための戻りアドレスを所定のメモリへ退避するアドレス退避部と、

割込発生部から復帰用割込を受けたとき、前記戻りアドレスを前記所定のメモリから読み出してグラフィックプロセッサへ通知するアドレス通知部とを備え、

グラフィックプロセッサへ通知された戻りアドレスがDMACのアクセスの対象アドレスとして設定されることを特徴とするマルチプロセッサシステム。

20

30

【請求項7】

グラフィックプロセッサとメインプロセッサを含み、

グラフィックプロセッサは、

メモリからディスプレイリストとして記述される命令を順次読み出すDMACと、

読み出された命令を順次解読するデコーダと、

解読された命令がディスプレイリストに含まれる所定のコール命令またはジャンプ命令であるときメインプロセッサに対して割込を発生する割込発生部とを備え、

メインプロセッサは、割込発生部から割込を受けたとき、前記コール命令またはジャンプ命令と、その後には置かれる実行停止命令のフォーマット中に分割して格納されたコール命令またはジャンプ先アドレスを取得してグラフィックプロセッサへ通知するアドレス通知部を備え、

グラフィックプロセッサへ通知されたコール先アドレスまたはジャンプ先アドレスがDMACのアクセスの対象アドレスとして設定されることを特徴とするマルチプロセッサシステム。

40

【請求項8】

第1のプロセッサが、実行中のメインルーチンにおいてコール命令を実行したとき、そのコール命令によって呼び出されるサブルーチンの処理が完了したときメインルーチンへ戻るための戻りアドレスの退避を第2のプロセッサに委託することを特徴とするマルチ

50

ロセッサシステムにおけるプログラム実行方法。

【請求項 9】

請求項 8 に記載の方法において、

第 1 のプロセッサ内部のスタック領域に空きがある場合は第 1 のプロセッサが自らそのスタック領域へ戻りアドレスを退避する一方、

前記スタック領域に空きがない場合は第 2 のプロセッサに戻りアドレスの退避を委託することを特徴とするマルチプロセッサシステムにおけるプログラム実行方法。

【請求項 10】

請求項 8 に記載の方法において、

前記コール命令が、戻りアドレスの退避を第 2 のプロセッサへ委託する旨を明示しない命令であるとき、第 1 のプロセッサは、自身内蔵するスタック領域へ戻りアドレスを退避する一方、

前記コール命令が、戻りアドレスの退避を第 2 のプロセッサへ委託する旨を明示する命令であるとき、第 1 のプロセッサは、第 2 のプロセッサに戻りアドレスの退避を委託することを特徴とするマルチプロセッサシステムにおけるプログラム実行方法。

【請求項 11】

第 1 のプロセッサがコール命令またはジャンプ命令を実行したとき、コール先アドレスまたはジャンプ先アドレスのビット数が第 1 のプロセッサによって取得可能なビット数を超えているとき、コール先アドレスまたはジャンプ先アドレスのアドレスの取得を第 2 のプロセッサに委託することを特徴とするマルチプロセッサシステムにおけるプログラム実行方法。

【請求項 12】

第 1 のプロセッサがコール命令またはジャンプ命令を実行したとき、前記コール命令またはジャンプ命令が、コール先アドレスまたはジャンプ先アドレスの取得を第 2 のプロセッサへ委託する旨を明示する命令であるとき、コール先アドレスまたはジャンプ先アドレスのアドレスの取得を第 2 のプロセッサに委託することを特徴とするマルチプロセッサシステムにおけるプログラム実行方法。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、コンピュータのプロセッサにマルチプロセッサ構成を用いたシステムと、そのシステムにおいてプログラムを実行する方法に関する。

【背景技術】

【0002】

リアルタイムマルチメディアアプリケーションの重要性は、より一層高まってきている。こうしたアプリケーションは、マルチプロセッサシステムで処理されることが多くなってきている。例えば、ゲーム機など高速かつ高精細な画像表示が要求される装置では、グラフィックプロセッサ (GPU; Graphics Processing Unit) とメインプロセッサとが連携して、高速性が要求されるグラフィック処理を実行する。

【0003】

一方、ゲーム機、携帯電話、個人情報端末 (PDA; Personal Digital Assistance) など、携帯型の電子装置は、ますます小型化する傾向にあり、プロセッサやメモリなどを実装する制御回路の小面積化、高集積化が要請される。

【発明の開示】

【発明が解決しようとする課題】

【0004】

制御回路が小面積化、高集積化されてくると、プログラムの実行などを円滑に行うために実装されるレジスタなどのハードウェア資源を実装する面積も縮小していく。このようなハードウェア資源を少なくしていくと、コマンドによっては、円滑に実行することが難しくなる場合も生じてくる。

10

20

30

40

50

【 0 0 0 5 】

本発明は、このような状況に鑑みてなされたものであり、その目的は、小規模な回路で柔軟に処理を実行することができるマルチプロセッサシステムとそのシステムにおけるプログラム実行方法を提供することにある。

【 課題を解決するための手段 】

【 0 0 0 6 】

上記課題を解決するために、本発明のある態様は、マルチプロセッサシステムにおけるプログラム実行方法である。この方法は、第1のプロセッサが、実行中のメインルーチンにおいてコール命令を実行したとき、そのコール命令によって呼び出されるサブルーチンの処理が完了したときメインルーチンへ戻るための戻りアドレスの退避を第2のプロセッサに委託する。

10

【 0 0 0 7 】

本発明のさらに別の態様もまた、マルチプロセッサシステムにおけるプログラム実行方法である。この方法は、第1のプロセッサがコール命令またはジャンプ命令を実行したとき、コール先アドレスまたはジャンプ先アドレスのフルアドレスの取得を第2のプロセッサに委託する。

【 0 0 0 8 】

なお、以上の構成要素の任意の組合せ、本発明の表現を方法、装置、システム、記録媒体、コンピュータプログラムなどの間で変換したものもまた、本発明の態様として有効である。

20

【 発明の効果 】

【 0 0 0 9 】

本発明によれば、小規模な回路で柔軟に処理を実行することができる。

【 発明を実施するための最良の形態 】

【 0 0 1 0 】

まず、実施形態を詳細に説明する前に、概要を以下に示す。

ある態様は、第1のプロセッサと第2のプロセッサを含む。第1のプロセッサは、実行中のメインルーチンにおいて所定のコール命令を実行したとき第2のプロセッサに対して割込を発生する割込発生部を備える。第2のプロセッサは、当該割込発生部から割込を受けたとき、コール命令によって呼び出されるサブルーチンの処理が完了したときメインルーチンへ戻るための戻りアドレスを所定のメモリ領域に退避するアドレス退避部を備える。ここで、「メインルーチン」および「サブルーチン」とは、コール元のルーチンとコール先のルーチンとの関係を示した記載表現であり、例えば、特定のルーチンがコール元のルーチンとの関係では「サブルーチン」と、コール先のルーチンとの関係では「メインルーチン」と表現されてもよい。

30

【 0 0 1 1 】

上記割込発生部は、サブルーチンにおいて所定のリターン命令を実行したとき第2のプロセッサに対して再度割込を発生し、第2のプロセッサは、再度発生された割込を受けたとき、戻りアドレスを第1のプロセッサへ通知するアドレス通知部をさらに備えてもよい。第1のプロセッサは、命令をフェッチするフェッチャーを備え、このフェッチャーによるアクセスの対象アドレスとして戻りアドレスが設定されてもよい。

40

【 0 0 1 2 】

別の態様も、第1のプロセッサと第2のプロセッサを含む。第1のプロセッサは、所定のコール命令またはジャンプ命令を実行したとき第2のプロセッサに対して割込を発生する割込発生部を備える。第2のプロセッサは、第1のプロセッサから割込を受けたとき、コール命令またはジャンプ命令と、その後に置かれる実行停止命令のフォーマット中に分割して格納されたコール先アドレスまたはジャンプ先アドレスを抽出するアドレス抽出部と、取得したコール先アドレスまたはジャンプ先アドレスを第1のプロセッサに通知するアドレス通知部とを備える。ここで、「ジャンプ命令」は、条件付ジャンプ命令、無条件ジャンプ命令のいずれであっててもよい。第1のプロセッサは、命令をフェッチするフェッ

50

チャーを備え、このフェッチャーによるアクセスの対象アドレスとしてコール先アドレスまたはジャンプ先アドレスが設定されてもよい。

【 0 0 1 3 】

別の態様は、グラフィックプロセッサとメインプロセッサを含む。グラフィックプロセッサは、メモリからディスプレイリストとして記述される命令を順次読み出すDMAC (Direct Memory Access Controller) と、読み出された命令を順次解読するデコーダと、解読された命令がディスプレイリストのメインルーチンに含まれる所定のコール命令であるときメインプロセッサに対して移行用割込を発生し、かつ、解読された命令がコール命令によって呼び出されるサブルーチンに含まれるリターン命令であるときメインプロセッサに対して復帰用割込を発生する割込発生部とを備える。メインプロセッサは、割込発生部から移行用割込を受けたとき、サブルーチンの処理が完了したときメインルーチンへ戻るための戻りアドレスを所定のメモリへ退避するアドレス退避部と、割込発生部から復帰用割込を受けたとき、戻りアドレスを所定のメモリから読み出してグラフィックプロセッサへ通知するアドレス通知部とを備える。グラフィックプロセッサへ通知された戻りアドレスがDMACのアクセスの対象アドレスとして設定される。

10

【 0 0 1 4 】

別の態様も、グラフィックプロセッサとメインプロセッサを含む。グラフィックプロセッサは、メモリからディスプレイリストとして記述される命令を順次読み出すDMACと、読み出された命令を順次解読するデコーダと、解読された命令がディスプレイリストに含まれる所定のコール命令またはジャンプ命令であるときメインプロセッサに対して割込を発生する割込発生部とを備える。メインプロセッサは、割込発生部から割込を受けたとき、コール命令またはジャンプ命令と、その後置かれる実行停止命令のフォーマット中に分割して格納されたコール先アドレスまたはジャンプ先アドレスを取得してグラフィックプロセッサへ通知するアドレス通知部を備える。グラフィックプロセッサへ通知されたコール先アドレスまたはジャンプ先アドレスがDMACのアクセスの対象アドレスとして設定される。

20

【 0 0 1 5 】

第1のプロセッサ内部のスタック領域に空きがある場合は第1のプロセッサが自らそのスタック領域へ戻りアドレスを退避してもよい。一方、スタック領域に空きがない場合は第2のプロセッサに戻りアドレスの退避を委託してもよい。コール命令が、戻りアドレスの退避を第2のプロセッサへ委託する旨を明示しない命令であるとき、第1のプロセッサは、自身内蔵するスタック領域へ戻りアドレスを退避してもよい。一方、コール命令が、戻りアドレスの退避を第2のプロセッサへ委託する旨を明示する命令であるとき、第1のプロセッサは、第2のプロセッサに戻りアドレスの退避を委託してもよい。

30

【 0 0 1 6 】

コール先アドレスまたはジャンプ先アドレスのビット数が第1のプロセッサによって取得可能なビット数を超過しているとき、コール先アドレスまたはジャンプ先アドレスのフルアドレスの取得を第2のプロセッサに委託してもよい。コール命令またはジャンプ命令が、コール先アドレスまたはジャンプ先アドレスの取得を第2のプロセッサへ委託する旨を明示する命令であるとき、コール先アドレスまたはジャンプ先アドレスのフルアドレスの取得を第2のプロセッサに委託してもよい。

40

【 0 0 1 7 】

以下、本発明の実施形態を詳細に説明する。

図1は、本発明の実施形態におけるマルチプロセッサシステム100の構成を示す図である。本実施形態におけるマルチプロセッサシステム100は、同一バス32上にGPU12、CPU14、およびメインメモリ16が接続された構成である。なお図1のブロック図では、GPU12、CPU14、およびメインメモリ16に関し、本実施形態の説明に必要な範囲で、それぞれの機能やデータ構造を記載しており、それらの一般的な機能やデータ構造は省略してある。

【 0 0 1 8 】

50

GPU12は、主に、3次元グラフィック表示に必要な計算やレンダリングを行うグラフィック処理用のプロセッサであり、ワンチップ化された半導体集積回路で構成されてもよい。GPU12は、DMAC20、デコーダ22、および割込発生部24を備える。DMAC20は、メインメモリ16との間で直接データ転送を行う。本実施形態では、メインメモリ16からコマンドをフェッチするフェッチャーとして機能する。

【0019】

DMAC20は、アドレススタック26、アドレスカウンタ28、およびメモリ制御部30を含む。アドレススタック26は、GPU12が現在実行中の描画処理を割込の発生により一時中断し、コールバック関数などを実行する場合、当該コールバック関数に係るサブルーチンを終了し、上記描画処理を再開する際に戻るべき戻りアドレスを格納する。本実施形態では、GPU12の小規模化の要請により、小さな領域とする。例えば、レジスタを2個設け、2つの戻りアドレスを格納可能な設計にしてもよい。この設計では、2回コール命令が実行され、2段のサブルーチンが構築された場合も、このアドレススタック26に2つの戻りアドレスを格納することができる。それ以上のコール命令が実行された場合、その他の領域に戻りアドレスを格納する必要が生じるが、その領域の確保する手法については後述する。アドレスカウンタ28は、アクセス先のアドレス、本実施形態ではメインメモリ16のアドレスを保持し、データ読み出しタイミングにしたがい、そのアドレスをインクリメントしていく。メモリ制御部30は、メインメモリ16との間のデータ転送を制御し、具体的には、アドレスカウンタ28の指示するアドレスからデータを讀み出す。

【0020】

デコーダ22は、DMAC20にフェッチされたコマンドを順次解読する。その解読にしたがい、図示しない加算器などを備える実行ユニットは、コマンドを実行し、図示しないフレームメモリなどに演算結果を書き出す。デコーダ22は、解読したコマンドが、割込を発生させて、後述するディスプレイリスト50のコール命令であった場合、割込ハンドラ52を讀み出す。割込ハンドラ52が実行されると、デコーダ22は、現在実行中の描画処理を一時中断して、その処理に復帰するための戻りアドレスをアドレススタック26に格納する。なお、描画処理を一時中断しない場合は、戻りアドレスを格納しなくてもよい。また、アドレススタック26に空き領域がない場合、割込発生部24に指示して、CPU14に戻りアドレスの格納を依頼する。

【0021】

戻りアドレスの格納と共に、デコーダ22は、初期設定によりフックされているコールバック関数を讀み出す。例えば、解読したコマンドフォーマットの下位16ビットに設定された引数を用いて、当該コールバック関数を呼び出す。この際、現在実行中の描画処理を一時中断させてもよいし、現在実行中の描画処理を継続してもよい。また、描画処理を一時中断した場合、コールバック処理から戻ってきた際、その中断していた処理を再開してもよいし、描画処理の再開を指示する別のコマンドを解読した際に、その処理を再開してもよい。これらは、予めコマンドフォーマットに、コマンドタイプとして設定することができる。また、ホルト命令などの実行停止命令の有無によって、設定してもよい。

【0022】

また、デコーダ22は、解読したコマンドが割込を発生させて、後述するディスプレイリスト50のコール命令またはジャンプ命令を発生させるものであり、かつ、アクセスすべき対象アドレスの一部しか設定されていないタイプであった場合、デコーダ22が割込発生部24に指示して、CPU14にディスプレイリスト50の対象アドレスの抽出を依頼する。なお、コール命令であった場合、上記と同様に戻りアドレスを格納する。本実施形態では、この対象アドレスを、1つのコマンドフォーマット中に設定しきれない場合、複数コマンドに跨って設定する。例えば、ディスプレイリスト50に割込を発生させるためのコール命令やジャンプ命令のコマンドと、その後ホルト命令やエンドマーカが設定されたコマンドとがセットで記述されるコマンド体系を用いている場合、それら2つのコマンドに分割して設定することができる。そして、上記対象アドレスが32ビットで指定

10

20

30

40

50

すべきものであり、1つのコマンドにアドレスを設定可能な領域が24ビット以下の場合、上記対象アドレスを16ビットずつに分割して、2つのコマンドフォーマット中に設定されてもよい。

【0023】

なお、当該アドレスが1つのコマンドに設定可能なデータ長であり、1つのコマンドに設定されている場合、デコーダ22は、アドレスの抽出をCPU14に依頼しない。通常通り、デコーダ22は、そのアドレスをアドレスカウンタ28に設定すればよい。対象アドレスが複数のコマンドに分割されて設定されているか否かは、コマンドタイプとして、予めコマンドのフォーマット中に設定しておき、それを解読して判断してもよいし、実行停止コマンドの有無で判断してもよい。

10

【0024】

また、デコーダ22は、解読したコマンドが割込を発生させて、上記コール命令に対応したリターン命令を発生させてるものであり、そのコール命令に対応する戻りアドレスをアドレススタック26または後述するメインメモリ16の補助スタック領域54から読み出す。その戻りアドレスが補助スタック領域54に格納されている場合、割込発生部24に指示して、CPU14に戻りアドレスの読み出しを依頼する。

【0025】

割込発生部24の以上の動作をまとめると、以下のイベントが発生した場合、CPU14に割込信号を発生させる。それは、戻りアドレスの格納をCPU14に依頼する場合である。また、複数のコマンドに分割して設定されたアドレスの抽出をCPU14に依頼する場合である。さらに、サブルーチンにおいて、コール元に戻るためのリターン命令が実行された際、戻りアドレスがメインメモリ16の補助スタック領域54に格納されている場合、その戻りアドレスの転送をCPU14に依頼する場合である。以下、本明細書中では、ディスプレイリスト50のメインルーチンに含まれる所定のコール命令を解読、実行したとき、CPU14に発生させる割込信号を移行用割込と呼ぶ。また、コール命令によって呼び出されたサブルーチンに含まれるリターン命令を解読、実行したとき、CPU14に発生させる割込信号を復帰用割込と呼ぶ。なお、割込を発生させるイベントには、キーボードからの入力など、種々存在するが、ここでは、本実施形態に関連するGPU12における描画処理実行中に、コール命令やジャンプ命令を解読、実行した際に発生する割込に限定して説明している。

20

30

【0026】

CPU14は、割込制御部40、アドレス抽出部42、アドレス退避部44、アドレス読出部46、およびアドレス通知部48を備える。これらは、後述する割込ハンドラ52の機能を実現する機能ブロックを示す。割込制御部40は、GPU12の割込発生部24から移行用割込および復帰用割込を受け付ける。アドレス抽出部42は、移行用割込により、複数のコマンドに跨る対象アドレスの抽出を依頼されたとき、それぞれのコマンドからアドレスの一部を抽出する。例えば、コール命令またはジャンプ命令などのコマンド、およびその後には置かれるホルト命令などのコマンドのフォーマット中に分離して設定されたアドレスを抽出する。この対象アドレスは、メインメモリ16内のコール先アドレスまたはジャンプ先アドレスなどを指す。

40

【0027】

CPU14は、GPU12より図示しないレジスタ群を多く備える。シフトレジスタを備えてもよい。例えば、1つのコマンドフォーマットが32ビット仕様で、アドレスの設定可能ビット数が24ビットまでと規定される場合、2つのコマンドのそれぞれの下位16ビットを利用して、32ビットで規定されるアドレスを16ビットずつに分割して、2つのコマンドに設定する。そして、当該アドレスの上位16ビットを先のコマンドに設定し、その下位16ビットを後のコマンドに設定する。アドレス抽出部42は、先のコマンドからアドレスの上位16ビットを抽出し、シフトレジスタに入れる。そのシフトレジスタでMSB (Most Significant Bit) 側に16桁分シフトさせ、それと共に、後のコマンドからアドレスの下位16ビットを抽出し、シフトレジスタ内のデータと組み合わせるこ

50

とにより、分割前のアドレスに復元することができる。

【0028】

アドレス通知部48は、アドレス抽出部42が生成した対象アドレスをGPU12に転送し、アドレスカウンタ28にそれを設定する。アドレスカウンタ28は、この処理により保持データが更新され、以降、この設定されたアドレスからインクリメントしていく。

【0029】

アドレス退避部44は、移行用割込により、上記戻りアドレスの格納を依頼された場合、後述するメインメモリ16内の補助スタック領域54に、その戻りアドレスを格納する。複数の戻りアドレスを格納する場合、LIFO (Last-In First-Out) 方式で補助スタック領域54に格納していくとよい。

10

【0030】

アドレス読出部46は、復帰用割込により、戻りアドレスの転送を依頼された場合、メインメモリ16内の補助スタック領域54から、その戻りアドレスを読み出す。アドレス読出部46は、補助スタック領域54がLIFO方式で制御されている場合、最後に格納された戻りアドレスを読み出す。この読み出す順番は、GPU12からの戻りアドレス転送の依頼順にも合致する。

【0031】

アドレス通知部48は、アドレス読出部46が読み出した戻りアドレスをGPU12に転送し、アドレスカウンタ28にそれを設定する。アドレスカウンタ28は、この処理により保持データが更新され、以降、この設定されたアドレスからインクリメントしていく。

20

【0032】

メインメモリ16は、各種のコマンドやデータを格納する。本実施形態では、主に、ディスプレイリスト50、割込ハンドラ52を格納する。その他、テクスチャデータなどが格納されてもよい。また、メインメモリ16には、補助スタック領域54が確保される。ディスプレイリスト50とは、描画実行時に効率的に処理できるよう、描画に関するコマンド群をひとまとめにリスト化したものである。本実施形態では、GPU12にフェッチされて実行される。割込ハンドラ52は、割込を処理、制御するために待機しているプログラムである。本実施形態では、主に、ディスプレイリスト50の実行中にコール命令やジャンプ命令が実行される場合の、GPU12およびCPU14の動作を制御するものである。

30

【0033】

補助スタック領域54は、本実施形態では、上記戻りアドレスを格納するために確保される領域である。例えば、メインメモリ16の最終の領域から先頭アドレスに戻る方向に、転送されてきた戻りアドレスを順々に格納していく。例えば、コール命令により実行されるサブルーチンの段数に対応して、補助スタック領域54を確保しておけばよい。

【0034】

図2は、ディスプレイリスト50の一例を示す図である。DMAC20は、メインメモリ内のディスプレイリスト50の先頭アドレスから順番にフェッチしていく。図2において、コマンド"cmd"は、描画コマンドを総称したものであり、プリミティブを規定するためのワイヤー・フレームやポリゴンの指定、色の指定などを行うコマンドが該当する。

40

【0035】

GPU12は、デコーダ22がアドレス#1に格納されたコマンド"call id"を解釈すると、引数iDを利用して、予めフックしてあるコールバック関数を呼び出す(処理s)。ここで、#記号は16進数表記を示す。図2では、予めアドレス#1000にコールバック関数が設定されている。そして、アドレス#1000から始まるサブルーチンを実行していき、デコーダ22がその終了を示すコマンド"return"を解釈すると、戻りアドレス#2に移行し、コール元のメインルーチンへ復帰する(処理t)。なお、この戻りアドレスは、アドレススタック26またはメインメモリ16内の補助スタック領域54に格納されている。上記サブルーチンをコールした際に格納されたものである。

50

【 0 0 3 6 】

次に、GPU 12は、デコーダ22がアドレス# 100に格納されたコマンド"cpu call #20"を解読すると、そのコマンドと次のアドレス# 101に格納されたコマンド"halt #FF"に分割されて設定されているアドレス# 20FFをCPU 14に抽出してもらい、そのアドレス# 20FFから始まるサブルーチン呼び出し(処理u)。そして、そのアドレス# 20FFから始まるサブルーチンを実行していき、デコーダ22がその終了を示すコマンド"return"およびそれとセットのコマンド"halt"を解読すると、戻りアドレス# 102に移行し、メインルーチンに復帰する(処理v)。

【 0 0 3 7 】

次に、GPU 12は、デコーダ22がアドレス# 1に格納されたコマンド"jmp #220"を解読すると、アドレス# 220にジャンプする(w)。この場合、同ルーチン内の分岐などによるジャンプであり、サブルーチン呼び出し、その終了後にコール元に復帰させるという処理ではないため、戻りアドレスの格納は必要ない。ここでは、アドレスが24ビット以内で規定できる場合、複数のコマンドに分割して設定する必要はなく、1つのコマンドでアドレスを設定することができる。この処理、一般のジャンプ命令の処理と同様である。

10

【 0 0 3 8 】

次に、GPU 12は、デコーダ22がアドレス# 300に格納されたコマンド"cpu jmp #40"を解読すると、そのコマンドと次のアドレス# 301に格納されたコマンド"halt #AA"に分割されて設定されているアドレス# 40AAをCPU 14に抽出してもらい、そのアドレス# 40AAにジャンプする(処理x)。なお、コマンド中の"cpu"は、CPU 14を介在させる処理であることを示す。

20

【 0 0 3 9 】

図3は、本実施形態におけるGPU 12がCPU 14にアドレス抽出を指示する第1コマンド例のフォーマットを示す。コマンド"cpu call"は、"halt"とセットで使用され、ディスプレイリスト50のコール命令を実行するものである。図3では、それぞれのコマンドが32ビットで規定され、上記8ビットすなわち31~24桁にコマンドコードが、そこから下位に8ビットすなわち23~16桁にコマンドタイプが、下位16ビットすなわち15~0桁にデータフィールドとして、コール先アドレスの上位または下位16ビット分が記述されている。ここで、コマンドコードは、各コマンドをコード化したものである。コマンドタイプは、割込を発生させた場合に実行中の処理を中断するか否か、戻りアドレスの保存が必要か否か、GPU 12単独の処理か否か、換言すればCPU 14と協働する処理か否か、といったコマンドの実行に関する付随的情報を指定するものである。なお、これらのフォーマットは一例であり、デコーダ22が正確にコマンドを解読できれば、任意にフィールドを設定し、命令やデータを記述することができる。例えば、上位8ビットでコマンドの命令部を記述できる場合、残りの24ビットをデータフィールドとして、そこにアドレスや引数を記述してもよい。

30

【 0 0 4 0 】

図4は、本実施形態におけるGPU 12がCPU 14にアドレス抽出を指示する第2コマンド例のフォーマットを示す。コマンド"cpu jmp"は、"halt"とセットで使用され、ディスプレイリスト50のジャンプ命令を実行するものである。このジャンプ命令には、分岐によるジャンプ命令や無条件のジャンプ命令などが含まれる。図4も、図3と同様に、それぞれのコマンドの下位16ビット分のフィールドに、ジャンプ先アドレスの上位または下位16ビット分が記述されている。

40

【 0 0 4 1 】

図5は、本実施形態におけるコール命令を実行した際のフローチャートである。GPU 12は、デコーダ22がコマンド"cpu call"を解読すると、そのコマンドを実行する(S10)。次に、デコーダ22がその次のコマンド"halt"を解読し、現在実行中の処理を一時中断する(S12)。GPU 12は、これに平行して、CPU 14に対して移行割込を発生させる(S14)。

50

【 0 0 4 2 】

C P U 1 4 は、この移行割込に応じて、G P U 1 2 が実行していたコール元のメインルーチンに復帰するための戻りアドレスをメインメモリ 1 6 内の補助スタック領域 5 4 に退避させる (S 1 6)。それと共に、上記コマンド " cpu call" および コマンド " halt" に分割して格納されているコール先アドレスを抽出する (S 1 8)。そして、そのコール先アドレスを G P U 1 2 に通知する (S 2 0)。

【 0 0 4 3 】

G P U 1 2 は、コール先アドレスから始まるサブルーチンに移行する (S 2 2)。そのサブルーチンを実行していき、デコーダ 2 2 がコマンド " return" を解読すると、そのコマンドを実行する (S 2 4)。デコーダ 2 2 がその次のコマンド " halt" を解読し、現在 10
実行中の処理を終了する (S 2 6)。G P U 1 2 は、これに平行して、C P U 1 4 に対して復帰割込を発生させる (S 2 8)。

【 0 0 4 4 】

C P U 1 4 は、この復帰割込に応じて、補助スタック領域 5 4 に退避させていた戻りアドレスをメインメモリ 1 6 内の補助スタック領域 5 4 から読み出す (S 3 0)。そして、その戻りアドレスを G P U 1 2 に通知する (S 3 2)。G P U 1 2 は、その戻りアドレスから、コール元のメインルーチンに復帰し、処理を続行する (S 3 4)。

【 0 0 4 5 】

なお、以上説明した処理は、G P U 1 2 のアドレススタック 2 6 に空き領域がないことが前提の処理である。この点、アドレススタック 2 6 に空き領域がある場合は、戻りアドレスをそこに退避してもよい。また、G P U 1 2 内にフラグレジスタなどを設け、アドレススタック 2 6 に空き領域があるか否かを示すフラグをそのレジスタに立ててもよい。この場合、デコーダ 2 2 は、そのフラグを参照して、C P U 1 4 に戻りアドレスの退避を依頼するか否かを判断することができる。戻りアドレスを G P U 1 2 のアドレススタック 2 6 に格納する場合、図 5 のステップ S 1 6、S 2 8 ~ S 3 2 の処理は不要となり、G P U 1 2 内での処理となる。 20

【 0 0 4 6 】

図 6 は、本実施形態におけるジャンプ命令を実行した際のフローチャートである。G P U 1 2 は、デコーダ 2 2 がコマンド " cpu jmp" を解読すると、そのコマンドを実行する (S 5 0)。デコーダ 2 2 がその次のコマンド " halt" を解読し、現在実行中の処理を終了 30
する (S 5 2)。G P U 1 2 は、これに平行して、C P U 1 4 に対して割込を発生させる (S 5 4)。

【 0 0 4 7 】

C P U 1 4 は、この割込に応じて、上記コマンド " cpu jmp" および コマンド " halt" に分割して格納されているジャンプ先アドレスを抽出する (S 5 6)。そして、そのジャンプ先アドレスを G P U 1 2 に通知する (S 5 8)。G P U 1 2 は、ジャンプ先アドレスに移行し (S 6 0)、そのアドレスから読み出しを再開する。

【 0 0 4 8 】

以上説明したように本実施形態によれば、コール命令が割込として発生した場合、割込が発生したコール元のメインルーチンに復帰するための戻りアドレスを G P U 1 2 外のメインメモリ 1 6 に退避することができることから、G P U 1 2 内の戻りアドレスを格納する領域を縮小させることができる。すなわち、コール先のルーチンでさらにコール命令が発生するという処理が複数回実行され、複数段のサブルーチンが構築される場合、コール元のルーチンに復帰するための戻りアドレスを複数、保持する必要がある。これを G P U 1 2 内にすべて格納しようとする、それだけレジスタを多く設ける必要が生じ、G P U 1 2 の小規模化を難しくする。これに対し、G P U 1 2 のレジスタを削減し、小規模化を図ると、複数段のサブルーチンを構築することが難しくなり、処理の柔軟性が低下してしまう。この点、本実施形態では、戻りアドレスをメインメモリ 1 6 に退避することができることから、G P U 1 2 の小規模化と処理の柔軟性を両立することができる。具体的には、ハードウェア資源的に簡素な G P U 1 2 でも、多数段のコール命令を処理することがで 40
50

きる。

【0049】

また、1つのコマンドフィールド中に、コール先アドレスやジャンプ先アドレスが入り
 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
 きらず、複数のコマンドに渡ってそれらのアドレスが設定される場合、そのアドレスの抽出をCPU14に委託することにより、GPU12のレジスタを削減することができる。すなわち、複数のコマンドに跨るアドレスを抽出するには、それぞれのコマンドに含まれるアドレスの一部を抽出し、それをシフトさせたり、合成させたりするためのレジスタが必要となる。そのレジスタは、1つのコマンドに設定された引数やアドレスを抽出する場合に必要なレジスタより大きな規模となる。この点、本実施形態では、複数のコマンドに分割して設定されたアドレスを抽出するなど、シフトや加算などの論理演算が必要なコマンドの解釈をメインのCPU14に委託する。これにより、GPU12は、1コマンド単位で解釈可能なコマンドを処理するに必要なレジスタを備えればよいことになり、回路面積を削減し、小規模化を図ることができる。また、GPU12のハードウェア資源では解釈が難しいコマンドは、CPU14が代わりに解釈することにより、様々な記述体系のコマンドに対応することもできる。よって、GPU12の小規模化と処理の柔軟性を両立することができる。GPU12が小規模化されれば、マルチプロセッサシステム全体を小規模化することができる。

【0050】

以上、実施形態をもとに本発明を説明した。なお本発明はこれらの実施形態に限定されることなく、そのさまざまな変形例もまた、本発明の態様として有効である。例えば、実施形態では、GPU12にアドレススタック26を備えた構成を説明した。この点、GPU12にアドレススタック26を設けずに、すべての戻りアドレスをメインメモリ16のスタック領域に格納してもよい。これによれば、さらにGPU12を小規模化することができる。

【0051】

また、実施形態では、コール先アドレスおよびジャンプ先アドレスを2つのコマンドに分割して設定した。この点、1つのコマンドのデータフィールドが小さい場合や、設定するアドレスのデータ長がさらに長い場合などには、3つ以上のコマンドに分割して設定してもよい。これによれば、コマンドの記述体系を柔軟に規定することができる。また、1つのコマンドフォーマットを短くした場合、さらにGPU12のハードウェア資源を小規模化することも可能になる。

【図面の簡単な説明】

【0052】

【図1】本発明の実施形態におけるマルチプロセッサシステムの構成を示す図である。

【図2】本実施形態におけるディスプレイリストの一例を示す図である。

【図3】本実施形態におけるGPUがCPUにアドレス抽出を指示する第1コマンド例のフォーマットを示す図である。

【図4】本実施形態におけるGPUがCPUにアドレス抽出を指示する第2コマンド例のフォーマットを示す図である。

【図5】本実施形態におけるコール命令を実行した際のフローチャートである。

【図6】本実施形態におけるジャンプ命令を実行した際のフローチャートである。

【符号の説明】

【0053】

12 GPU、 14 CPU、 16 メインメモリ、 20 DMAC、 22 デコーダ、 24 割込発生部、 26 アドレススタック、 28 アドレスカウンタ、 30 メモリ制御部、 40 割込制御部、 42 アドレス抽出部、 44 アドレス退避部、 46 アドレス読出部、 48 アドレス通知部、 50 ディスプレイリスト、 52 割込ハンドラ、 54 補助スタック領域、 100 マルチプロセッサシステム。

【要約】

10

20

30

40

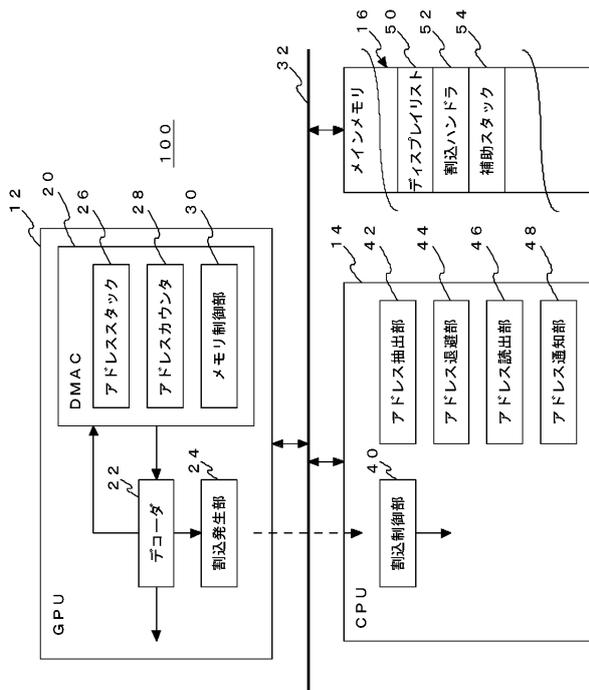
50

【課題】 小規模回路で柔軟に処理を実行する。

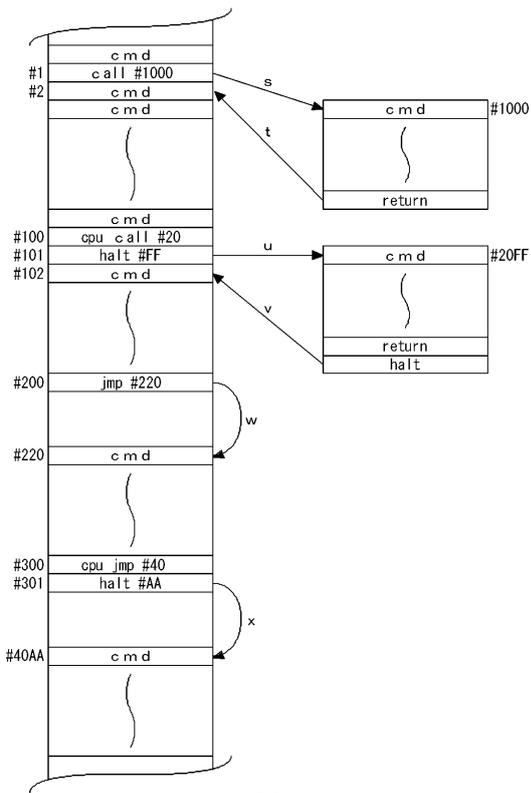
【解決手段】 マルチプロセッサシステム100において、第1のプロセッサの割込発生部24は、実行中のメインルーチンにおいて所定のコール命令を実行したとき第2のプロセッサに対して割込を発生させる。第2のプロセッサのアドレス退避部44は、割込発生部24から割込を受けたとき、コール命令によって呼び出されるサブルーチンの処理が完了したときメインルーチンへ戻るための戻りアドレスを所定のメモリ領域に退避する。割込発生部24は、サブルーチンにおいて所定のリターン命令を実行したとき第2のプロセッサに対して再度割込を発生させ、第2のプロセッサのアドレス通知部48は、再度発生された割込を受けたとき、戻りアドレスを第1のプロセッサへ通知する。

【選択図】 図1

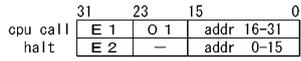
【図1】



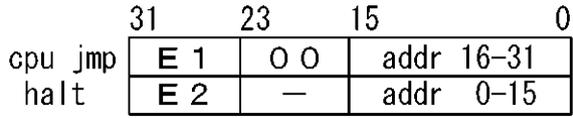
【図2】



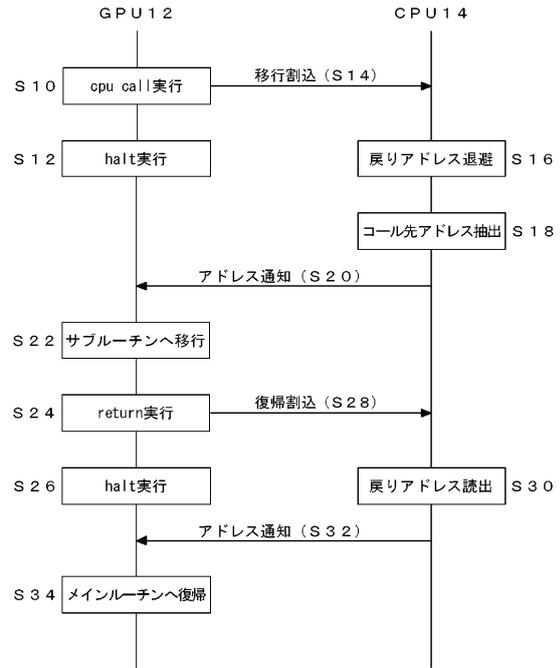
【図3】



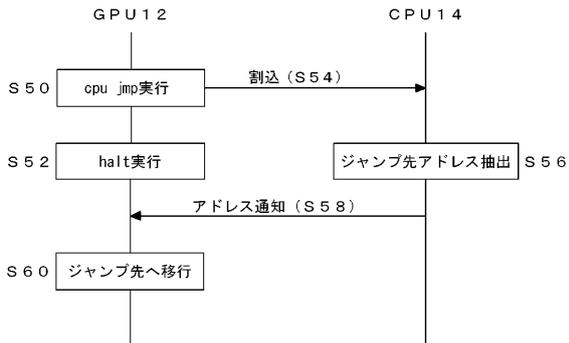
【図4】



【図5】



【図6】



フロントページの続き

審査官 後藤 彰

(56)参考文献 米国特許出願公開第2003/0120897(US, A1)

特開2003-36169(JP, A)

特開2002-202964(JP, A)

特開昭57-143642(JP, A)

(58)調査した分野(Int.Cl., DB名)

G06F 9/30 - 9/42

G06F 15/16 - 15/177

G06T 1/20