(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4847036号
(P4847036)
(45)発行日　平成23年12月28日(2011.12.28)　(24)登録日　平成23年10月21日(2011.10.21)

(51)Int.Cl.　F I
G06F 13/362 (2006.01)　G06F 13/362 510E
G06F 13/362 510H

請求項の数 14　(全 17 頁)

(21)出願番号　特願2005-99420(P2005-99420)
(22)出願日　平成17年3月30日(2005.3.30)
(65)公開番号　特開2006-277620(P2006-277620A)
(43)公開日　平成18年10月12日(2006.10.12)
審査請求日　平成20年3月26日(2008.3.26)

前置審査

(73)特許権者　000001007
キヤノン株式会社
東京都大田区下丸子３丁目３０番２号
(74)代理人　100076428
弁理士　大塚　康徳
(74)代理人　100112508
弁理士　高柳　司郎
(74)代理人　100115071
弁理士　大塚　康弘
(74)代理人　100116894
弁理士　木村　秀二
(74)代理人　100130409
弁理士　下山　治
(74)代理人　100134175
弁理士　永川　行光

最終頁に続く

(54)【発明の名称】バスアクセスを調停する制御装置およびデータ処理装置の制御方法

(57)【特許請求の範囲】
【請求項１】
バスのアクセスを調停する制御装置であって、
前記アクセスの対象であるバスに接続されたメモリに対してリードアクセスを要求する第一のマスタと、前記メモリに対してライトアクセスを要求する第二のマスタを含む複数のマスタ、並びに、前記第二のマスタのライトアクセスの要求を、前記第一のマスタのリードアクセスの要求よりも優先して前記調停を行う第一のアービタを備え、前記アクセスの対象であるバスに接続された複数のバスマスタと、
前記バスにアクセスしたバスマスタを識別する識別手段と、
前記識別手段の識別結果に応じて前記複数のバスマスタの優先順位を設定する第二のアービタとを有することを特徴とする制御装置。
【請求項２】
前記第一のアービタは、前記複数のマスタそれぞれに対応するバッファの空き状態に応じて、前記複数のマスタのバスアクセスの要求を受け付けることを特徴とする請求項1に記載の制御装置。
【請求項３】
前記第一のアービタは、前記複数のマスタそれぞれのバッファの空き状態が前記複数のバスマスタごとに規定された条件を満たさない場合は、前記バスアクセスの要求の発生頻度が高いマスタのバスアクセスを優先して前記調停を行うことを特徴とする請求項1または請求項2に記載の制御装置。

【請求項４】
前記第一のアービタは、さらに、レイテンシの短いバスアクセスを要求するマスタのバスアクセスを優先して前記調停を行うことを特徴とする請求項2または請求項3に記載の制御装置。
【請求項５】
前記第一のアービタは、前記複数のマスタからリードアクセスまたはライトアクセスの種別のバスアクセス要求を受け付け、直前に受け付けたバスアクセス要求の種別と同じ種別のバスアクセス要求を優先して前記調停を行うことを特徴とする請求項1から請求項4の何れか一項に記載の制御装置。
【請求項６】
前記複数のマスタの少なくとも一つはライトアクセスによってDRAMに書き出すデータを保持するライトバッファに対応し、前記複数のマスタの少なくとも一つはリードアクセスによって前記DRAMから読み出すデータを保持するリードバッファに対応し、
前記ライトバッファが保持するデータ量が所定量以上の場合、前記第一のアービタは、当該ライトバッファに対応するマスタの要求を、前記リードバッファに対応するマスタのリードアクセスの要求よりも優先して前記調停を行うことを特徴とする請求項1から請求項4の何れか一項に記載の制御装置。
【請求項７】
前記ライトバッファが保持するデータは、前記リードバッファから読み出したデータに処理を施したデータであることを特徴とする請求項6に記載の制御装置。
【請求項８】
前記第二のアービタは、直前に前記バスにアクセスしたとして前記識別手段によって識別されたバスマスタを優先して前記調停を行うことを特徴とする請求項1に記載の制御装置。
【請求項９】
一つのバスマスタによる前記バスの連続アクセスの回数をカウントする計数手段を更に有し、
前記第二のアービタは、前記連続アクセスの回数が所定値に達したバスマスタの前記バスアクセスの要求を無視することを特徴とする請求項1または請求項8に記載の制御装置。
【請求項１０】
前記計数手段は、直前に前記バスにアクセスしたバスマスタとは異なるバスマスタのアクセスが許可された場合、または、前記バスアクセスの要求を出しているバスマスタの数が一つ以下の場合、前記カウントの値を零にリセットすることを特徴とする請求項9に記載の制御装置。
【請求項１１】
前記バスおよびCPUバスに接続されたバスブリッジを更に有し、
前記第二のアービタは、前記複数のバスマスタのバスアクセスの要求よりも、前記バスに対する前記バスブリッジのバスアクセスの要求を優先して前記調停を行うことを特徴とする請求項1、または、請求項8から請求項10の何れか一項に記載の制御装置。
【請求項１２】
前記複数のバスマスタのうち少なくとも一つは、画像データのデータパスに使用する少なくとも二つのマスタ、および、前記画像データの処理において発生した誤差のデータパスに使用する少なくとも二つのマスタを有する画像処理手段であることを特徴とする請求項1、または、請求項8から請求項11の何れか一項に記載の制御装置。
【請求項１３】
前記複数のバスマスタのうち少なくとも一つは、プリンタエンジンの同期信号に合わせて画像処理を開始する画像処理手段であり、
前記第二のアービタは、前記画像処理手段のバスアクセスの要求を最優先して前記調停を行うことを特徴とする請求項1、または、請求項8から請求項12の何れか一項に記載の制御装置。

【請求項１４】
バスのアクセスを調停するデータ処理装置の制御方法であって、
前記アクセスの対象であるバスに接続されたメモリに対してリードアクセスを要求する第一のマスタと、前記メモリに対してライトアクセスを要求する第二のマスタを含む複数のマスタのうち、前記第二のマスタのライトアクセスの要求を、前記第一のマスタのリードアクセスの要求よりも優先して前記調停を行う第一の調停工程と、
前記アクセスの対象であるバスに接続され、前記第一、第二のマスタを備える複数のバスマスタのうち、前記バスにアクセスしたバスマスタを識別する識別工程と、
前記識別工程の識別結果に応じて前記複数のバスマスタの優先順位を設定する調停工程とを有することを特徴とする制御方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、バスアクセスを調停する制御装置およびデータ処理装置の制御方法に関する。
【背景技術】
【０００２】
バスマスタのバスアクセスを調停するアービタは、DRAMなどのメモリバスをアクセスする複数のバスマスタからのバス使用要求に対して、そのうちの一つのバスマスタに対してバス使用権を与えるバス使用権の管理制御（調停）を行う。各バスマスタに対するバス使用権の優先順位はハードウェア的に決められているので、複数のバスマスタから同時にバス使用要求があった場合、バスアービタは、予め決められている優先順位に従い、必ず、優先順位が高いバスマスタにバス使用権を与えるようにバス使用許可信号を出力する（例えば、特許文献1参照）。
【０００３】
従って、高い優先順位に位置付けされたバスマスタから頻繁にバス使用要求が出力されると、そのバスマスタのバス使用権の獲得割合が多くなり、低い優先順位に位置付けされたバスマスタがバス使用権を獲得するのが困難になるという問題点がある。
【０００４】
また、バス使用要求の受け付けを制限し、一度受け付けたバス使用要求のすべてに対してバス使用権を与えてから、次のバス使用要求を受け付けるようにすることができる。しかし、多数のバスマスタからバス使用要求が出る場合、たとえ高い優先順位に位置付けされたバスマスタでも、低い優先順位に位置付けされたバスマスタと同程度のアクセスしか実行できないという問題がある。
【０００５】
さらに、バースト転送がサポートされたバスや、DRAMなどのメモリバスにおいて、バス使用権を獲得したバスマスタが頻繁に入れ替わると、アドレス設定のためのオーバヘッドが増加し、バスの使用効率が低下するという問題点がある。
【０００６】
また、一つのアービタにバス使用権の調停を集中させると、バスマスタ数の増加に伴い調停処理が複雑化して回路の増大を招くだけでなく、バスの高速性も損われる欠点がある。
【０００７】
【特許文献１】特開平09-062579号公報
【発明の開示】
【発明が解決しようとする課題】
【０００８】
本発明は、バス使用権を動的に管理して、バスの使用効率を向上することを目的とする。
【０００９】
また、バス使用権の調停を分散処理して、バスマスタ数の増加に伴う回路規模の増大を

防ぎ、バスの高速性を維持することを他の目的とする。

【課題を解決するための手段】

【００１０】

本発明は、前記の目的を達成する一手段として、以下の構成を備える。

【００１１】

本発明にかかる制御装置は、バスのアクセスを調停する制御装置であって、前記アクセスの対象であるバスに接続されたメモリに対してリードアクセスを要求する第一のマスタと、前記メモリに対してライトアクセスを要求する第二のマスタを含む複数のマスタ、並びに、前記第二のマスタのライトアクセスの要求を、前記第一のマスタのリードアクセスの要求よりも優先して前記調停を行う第一のアービタを備え、前記アクセスの対象であるバスに接続された複数のバスマスタと、前記バスにアクセスしたバスマスタを識別する識別手段と、前記識別手段の識別結果に応じて前記複数のバスマスタの優先順位を設定する第二のアービタとを有することを特徴とする。

【発明の効果】

【００１５】

本発明によれば、バス使用権を動的に管理することで、バスの使用効率を向上することができる。

【００１６】

また、バス使用権の調停を分散処理することで、バスマスタ数の増加に伴う回路規模の増大を防ぎ、バスの高速性を維持することができる。

【発明を実施するための最良の形態】

【００１７】

以下、本発明にかかる実施例を図面を参照して詳細に説明する。勿論、以下の実施例は、本発明の技術分野における当業者による実施を容易にするための開示を提供するものであり、特許請求の範囲によって確定される本発明の技術的範囲に含まれるほんの一部の実施例に過ぎない。従って、本明細書に直接的に記載されていない実施例であっても、技術思想が共通する限り、本発明の技術的範囲に包含されることは、当業者にとって自明である。

【００１８】

また、便宜上、複数の実施例を記載するが、これらは個別に発明として成り立つだけではなく、勿論、複数の実施例を適宜組み合わせることでも発明が成り立つことは、当業者であれば容易に理解できる。

【実施例１】

【００１９】

［装置の構成］

まず、各種画像処理を行った後、画像を出力する画像処理装置を例に説明する。図1は実施例1の画像処理装置の構成例を示すブロック図である。

【００２０】

図1において、CPU 1は、ROM 2に格納されたプログラムに従い、DRAM 7をワークメモリとして、画像処理装置全体を制御する。また、CPU 1は、CPUバス1aを介してROM 2、バスブリッジ3およびI/0ポート9に接続する。

【００２１】

アービタ5は、バスブリッジ3を介したCPU 1によるDRAM 7のアクセス、および、n（n≧1、整数）個の画像処理モジュール4によるDRAM 7のアクセスを調停する。また、DRAM 7は、DRAMインタフェイス(I/F) 6を有する。

【００２２】

また、画像処理モジュール1の一つ（図1ではモジュール4n）は、例えばインクジェットプリンタのプリントヘッドに接続するためのヘッドI/F 8が備わる。

【００２３】

なお、図1には、DRAM 7がCPU 1および複数の画像処理モジュール4に共有される例を示

したが、パフォーマンスの維持または向上のために、CPU 1用のRAMをCPUバス1aに接続する構成としてもよい。

【００２４】

［処理動作］

CPU 1は、ROM 2に格納されたプログラムに従い、I/Oポート9より処理すべき画像データを受け取り、バスブリッジ3、アービタ5、DRAM I/F 6を経てDRAM 7へ格納する。次に、CPU 1は画像処理モジュール4aのコンフィグレーションレジスタを設定し、画像処理モジュール4aを動作させる。

【００２５】

画像処理モジュール4aは、所定の処理を実行し、コンフィグレーションレジスタに設定された処理すべきデータの読出処理が終了するか、あるいは、コンフィグレーションレジスタに設定された処理データの書込処理が終了すると、割り込みを発生して、処理の終了をCPU 1へ通知する。

【００２６】

CPU 1は、割り込みを受けると、その要因を解析し、画像処理モジュール4aの読出処理が終了した場合は、次に処理すべきデータの設定を行い、画像処理モジュール4aに処理を続行させる。また、画像処理モジュール4aの書込処理が終了した場合は、次の処理データの格納先を設定し、画像処理モジュール4aに処理を続行させるとともに、次の画像処理モジュール4bのコンフィグレーションレジスタを設定して、画像処理モジュール4bを動作させる。

【００２７】

画像処理モジュール4bは、所定の処理を実行し、コンフィグレーションレジスタに設定された処理すべきデータの読出処理が終了するか、あるいは、コンフィグレーションレジスタに設定された処理データの書込処理が終了すると、割り込みを発生して、処理の終了をCPU 1へ通知する。

【００２８】

CPU 1は、割り込みを受けると、その要因を解析し、画像処理モジュール4bの読出処理が終了した場合は、次に処理すべきデータの設定を行い、画像処理モジュール4bに処理を続行させる。また、画像処理モジュール4bの書込処理が終了した場合は、次の処理データの格納先を設定し、画像処理モジュール4bに処理を続行させるとともに、次の画像処理モジュール4cのコンフィグレーションレジスタを設定して、画像処理モジュール4cを動作させる。

【００２９】

このように、前の処理が終わった直後に、次の画像処理モジュールを起動し、処理データを次々と画像処理モジュールに受け渡すことで、画像処理モジュール単位のパイプラインを構成することができる。

【００３０】

そして、画像処理モジュール4mまでの処理が進み、所定量以上のビットマップデータを生成すると、CPU 1は図示しないプリンタエンジンを起動し、プリンタエンジンの同期信号に合わせて画像処理モジュール4nに処理を開始させ、ヘッドI/F 8を介してビットマップデータをプリンタエンジンに送り、プリントさせる。

【００３１】

●画像処理モジュールの構成

図2は画像処理モジュールの詳細な構成例を説明するブロック図で、リードバッファ10、m（m≧1、整数）個のサブモジュール11、ライトバッファ12、アービタ13、リードアドレス発生器14、割込コントローラ15、ライトアドレス発生器16を備える。

【００３２】

CPU 1は、画像処理モジュール4のコンフィグレーションレジスタの設定により、リードアドレス発生器14にリードの開始および終了アドレスを設定し、リードイネーブル信号Renをセットする。また、ライトアドレス発生器16にライトの開始および終了アドレスを設

定し、ライトイネーブル信号Wenをセットする。

【００３３】

アービタ13はリードバッファ10の空き容量Rpおよびリードアドレス発生器14のイネーブル信号Renを検出し、リードアドレスが有効（Ren=‘１’）で、リードバッファ10にデータが格納可能（Rp≧Rn）であれば、アービタ5へリードリクエスト（PREQ=‘１’、PNRW=‘０’、PNUM=Rn、PADD=Rad）を発行する。

【００３４】

一方、ライトバッファ12のデータ蓄積数Wpが所定のワード数以上（Wp≧Wn）になると、アービタ13は、ライトアドレス発生器16のイネーブル信号Wenを検出し、ライトアドレスが有効（Wen=‘１’）であれば、アービタ5へライトリクエスト（PREQ=‘１’、PNRW=‘１’、PNUM=Wn、PADD=Wad）を発行する。

【００３５】

アービタ5は、画像処理モジュール4からリクエスト信号PREQを受け取ると、PNRWでリード/ライトを判別し、PNUMでワード数を、PADDでアドレスを検知する。CPU 1および他の画像処理モジュール4からのリクエストがなければ、アービタ5は、DRAM I/F 6を通じてDRAM 7の該当データのアクセスを開始する。DRAM I/F 6にリクエストが受け付けられると、アービタ5は、受領信号PACKを要求元の画像処理モジュール4に返す。一方、CPU 1および他の画像処理モジュール4からリクエストがある場合は、優先順位に従って、リクエストを受け付ける。

【００３６】

アービタ13は、受領信号PACKを受け取ると、リードリクエストの場合は受領信号Rackを要求元のリードアドレス発生器14に返し、ライトリクエストの場合は受領信号Wackを要求元のライトアドレス発生器16に返す。

【００３７】

受領信号Rackを受け取ったリードアドレス発生器14は、次のアドレスを生成するが、リクエストしたアドレスがリード終了アドレスの場合は、リードイネーブル信号Renをリセットし、リード終了信号Rendを割込コントローラ15に出力する。また、受領信号Wackを受け取ったライトアドレス発生器16は、次のアドレスを生成すが、リクエストしたアドレスがライト終了アドレスの場合は、ライトイネーブル信号Wenをリセットし、ライト終了信号Wendを割込コントローラ15に出力する。

【００３８】

割込コントローラ15は、コンフィグレーションレジスタによってリード終了割込マスクおよびライト終了割込マスクの設定が可能で、各割込マスクの設定が割込イネーブルとなっている場合は、リード終了信号Rendまたはライト終了信号Wendによって割込信号INTを生成してCPU 1へ通知する。

【００３９】

CPU 1は、割り込みを受付けると、割込コントローラ15のステータスを読み取り、割り込みの要因がリード終了の場合は、リード終了割込マスクをリセットして割り込みを解除する。さらに処理を続行する場合は、リード開始および終了アドレスを再設定し、リードイネーブル信号Renのセットなどを行った後、リード終了割込マスクをセットする。同様に、割り込み要因がライト終了の場合は、ライト終了割込マスクをリセットして割り込みを解除する。さらに処理を続行する場合は、ライト開始および終了アドレスを再設定し、ライトイネーブル信号Wenのセットなどを行った後、ライト終了割込マスクをセットする。

【００４０】

次に、DRAM 7よりデータが読み出されるとアービタ5は、DRAMデータ有効信号PVALIDを要求元の画像処理モジュール4に返す。要求元の画像処理モジュール4のアービタ13は、リードバッファ10へデータ有効信号Rvalidを返す。リードバッファ10は、データ有効信号Rvalidがセットされている期間、DRAMデータ出力信号PDIN上のデータを格納する。この操作により、DRAM 7のデータがリードバッファ10へ格納される。

【００４１】
一方、DRAM 7にデータを書込む場合は、DRAM 7の書込タイミングに合わせて、アービタ5は、DRAMデータ有効信号PVALIDを要求元の画像処理モジュール4に返す。要求元の画像処理モジュール4のアービタ13は、ライトバッファ12へデータ有効信号Wvalidを返す。ライトバッファ12は、データ有効信号Wvalidがセットされている期間、DRAMデータ入力信号PDOUTとしてDRAM 7に書き込むデータを出力する。この操作により、ライトバッファ12のデータがDRAM 7へ格納される。
【００４２】
リードバッファ10は、サブモジュール11aの処理に必要なデータが揃うと有効信号valid_0をセットし、サブモジュール11aの処理に必要なデータが揃っていなければ有効信号valid_0をリセットする。
【００４３】
そして、サブモジュール11aからの保持要求信号stall_0がセットされていない場合、リードバッファ10は、格納したデータをクロックに同期して出力する。もし、保持要求信号stall_0がセットされている場合は、データを更新しない。
【００４４】
サブモジュール11aは、有効信号valid_0がセットされているデータのみを受け取る。なお、データの受け取りが不可能な場合は、保持要求信号stall_0をセットし、リードバッファ10の出力をホールドする。なお、入力データの並べ替えが不要な場合、リードバッファ10はFIFO (First In First Out)メモリでよい。同様に、出力データの並べ替えが不要な場合、ライトバッファ12はFIFOメモリでよい。
【００４５】
画像処理モジュール4の内部は、画像処理を行う一つ以上のサブモジュール11によって構成され、各サブモジュール間では、上記と同様の動作（有効信号valid_xと保持要求信号stall_xによるハンドシェーク）によってデータdata_xの受け渡しを行う。
【００４６】
●サブモジュール間のデータ転送
図3はサブモジュール11間のデータ転送を説明するタイミングチャートである。
【００４７】
データ送信側のサブモジュール11は、データが出力可能であればクロックclkの立ち上がりに同期してデータ信号d1および有効信号validをセットする(T1)。次のクロックの立ち上がりで受信側から保持要求信号stallがセットされていなければデータ信号d1が受信されたとみなし、次のデータが出力可能であればデータ信号d2および有効信号validをセットする(T2)。もし、次のデータが出力不能であれば有効信号validをリセットする(T3)。また、次のクロックの立ち上がりで受信側から保持要求信号stallがセットされていた場合は、データ信号が受信されなかったとみなし、データ信号d5および有効信号validをホールドする(T7)。なお、受信側から保持要求信号stallがセットされていても有効信号validをセットしていなければ(T8)、無効データであるからデータ信号および有効信号validをホールドせずに、次の有効データとしてデータ信号d6を出力し、有効信号validをセットする(T9)。言い換えれば、有効信号validをセットしていない場合の保持要求信号stallは無視する。
【００４８】
データ受信側のサブモジュール11は、受信可能であれば有効信号validがセットされているデータ信号をクロックclkの立ち上がりに同期して受け取る(T1、T2、T4、T5)。また、受信不能であれば保持要求信号stallをセットし、送信側にデータ信号d5および有効信号validを保持させる(T6)。そして、受信可能になると保持要求信号stallをリセットし、データ出力d5を受信する(T7)。
【００４９】
また、ライトバッファ12は、バッファに空きがあれば、サブモジュール11からの有効信号valid_nがセットされたときのデータ信号data_nをバッファに格納する。もし、バッフ

ァに空きがなければ、保持要求信号stall_nをセットし、サブモジュール11に出力をホールドさせる。

【００５０】

●アービタ13のアルゴリズム

図4はアービタ13のアルゴリズムを説明するフローチャートである。なお、以下の説明においては、リクエストキューに蓄積されたリクエストの数をPp、リクエストキューに蓄積されたリクエストが実行された場合のライトバッファ12のデータ蓄積数（評価値）をPw、リクエストキューに蓄積されたリクエストが実行された場合のリードバッファ10の空き容量（評価値）をPrとする。なお、リクエストキューに蓄積されたリクエストの数Ppはアービタ5がリクエストを受け付ける（PACK=‘１’）と一つ減じられる。また、以下の説明では、（リードリクエストの頻度）＞（ライトリクエストの頻度）と仮定し、バスアクセスのリクエスト発生頻度が低いモジュールの順に、後述するバッファの空き状態の検出を行う。

【００５１】

ライトリクエストWreqは、ライトバッファ12の蓄積数の評価値Pwが所定のワード数以上Pw≧Wnになり、ライトアドレスが有効Wen=‘１’であるならばWreq=‘１’になり、リードリクエストRreqは、リードバッファ10の空き容量の評価値Prが所定のワード数以上Pr≧Rnになり、リードアドレスが有効Ren=‘１’であるならばRreq=‘１’になるものとする。

【００５２】

まず、ライトリクエストWreq=‘１’で、ライトバッファ12のデータ蓄積数の評価値Pwと所定値Wthの関係がPw≧Wth、または、直前にリクエストキューに格納したリクエストがライトリクエスト（ID1=IDw）か否かを判定し(S201)、そうであればライトリクエストをリクエストキューに蓄積する(S205)。

【００５３】

上記の条件が成立しない場合は、リードリクエストRreq=‘１’で、リードバッファ10の空き容量の評価値Prと所定値Rthの関係がPr≧Rth、または、直前にリクエストキューに格納したリクエストがリードリクエスト（ID1=IDr）か否かを判定し(S202)、そうであればリードリクエストをリクエストキューに蓄積する(S206)。

【００５５】

上記の二つの条件が成立せず、リードリクエストRreq=‘１’の場合は(S203)、リードリクエストをリクエストキューに蓄積し(S206)、ライトリクエストWreq=‘１’の場合は(S204)、ライトリクエストをリクエストキューに蓄積する(S205)。

【００５６】

リクエストキューへのライトリクエストの蓄積(S205)は、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDをライトリクエスト識別コードIDwに更新する。同時に、ライトバッファ12の蓄積数の評価値Pwからライトデータ数Wnを減算して評価値Pwを更新し、リクエストキューに蓄積されているリクエスト数Ppを一つ増やす。

【００５７】

同様に、リクエストキューへのリードリクエストの蓄積(S206)は、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDをリードリクエスト識別コードIDrに更新する。同時に、リードバッファ10の空き容量の評価値Prからリードデータ数Rnを減算して評価値Prを更新し、リクエストキューに蓄積されているリクエスト数Ppを一つ増やす。

【００５８】

上記処理が終了すると、処理をステップS201に戻し、上記の処理を繰り返す。

【００５９】

なお、ここではバッファの容量、アービタ5のシーケンスなどによりリクエストキューに蓄積されるリクエスト数Ppの上限が決まるため、リクエスト数Ppを制限しないが、システムによって制限が必要な場合は、リクエスト数Ppが最大値となった時点で、Rreq=Wreq=

0とすればよい。

【００６０】

●アービタ5のアルゴリズム

図5はアービタ5のアルゴリズムを説明するフローチャートである。なお、以下の説明では、三つの画像処理モジュールM1、M2、M3と、エンジン処理モジュールM4、バスブリッジB0が接続されているとする。エンジン処理モジュールM4には、リアルタイム制御のために、最高の優先順位を設定する。また、バスブリッジB0は、エンジン処理モジュールM4に次いで高い優先順位に設定する。三つの画像処理もジュールの優先順位は同じである。従って、各モジュールの優先順位は下記のようになる。

M4 ＞ B0 ＞ M1, M2, M3

【００６１】

アービタ5は、まず、優先順位の一番高いエンジン処理モジュールM4からバス使用権のリクエストreq4がある（つまりreq4=‘１’）か否かを判定する(S211)。req4=‘１’ならばリクエストreq4を受け付け、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDをエンジン処理モジュールM4のリクエスト識別コードID_4に更新し、エンジン処理モジュールM4に受領信号PACKを返し(S216)、処理をステップS211に戻す。

【００６２】

エンジン処理モジュールM4のリクエストがなければ、バスブリッジB0のリクエストreq0がある（つまりreq0=‘１’）か否かを判定する(S212)。req0=‘１’ならばリクエストreq0を受け付け、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDをバスブリッジB0のリクエスト識別コードID_0に更新し、バスブリッジB0に受領信号PACKを返し(S217)、処理をステップS211に戻す。

【００６３】

バスブリッジB0のリクエストもなければ、画像処理モジュールM1のリクエストreq1がある（つまりreq1=‘１’）か否かを判定する(S213)。req1=‘１’で、さらに、リクエストキューに蓄積されている画像処理モジュールM1のリクエスト数P1が三つの画像処理モジュールM1、M2、M3の中で一番多い（つまりP1=Pmax）、あるいは、直前に受け付けたリクエストが画像処理モジュールM1のリクエスト（つまりID1=ID_1）であれば、リクエストreq1を受け付け、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDを画像処理モジュールM1のリクエスト識別コードID_1に更新し、画像処理モジュールM1に受領信号PACKを返し(S218)、処理をステップS211に戻す。

【００６４】

また、req1=‘０’あるいはP1≠PmaxかつID1≠ID_1ならば、他の画像処理モジュールM2、M3のリクエストreq2、req3がある（つまりreq2=‘１’、req3=‘１’）か否かを判定する(S214、S215)。リクエストがあり、さらに、リクエストキューに蓄積されている画像処理モジュールが三つの画像処理モジュールM1、M2、M3の中で一番多い（つまりP2=PmaxまたはP3=Pmax）、あるいは、直前に受け付けたリクエストが画像処理モジュールM2またはM3のリクエスト（つまりID1=ID_2またはID1=ID_3）であれば、リクエストreq2またはreq3を受け付け、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDを画像処理モジュールM2またはM3のリクエスト識別コードID_2またはID_3に更新し、画像処理モジュールM2またはM3に受領信号PACKを返し(S219またはS220)、処理をステップS211に戻す。

【００６５】

上記のアルゴリズムは、一部固定の優先順位を用いているので、高い優先順位に位置付けされているバスマスタ（エンジン処理モジュールM4やバスブリッジB0）から頻繁にバス使用権が要求されると、そのバスマスタにバスが占有されてしまう可能性がある。とくに、上記の例では、バスブリッジB0よりもエンジン処理モジュールM4の優先順位が高く設定されているため、タイミングによってCPU 1のリクエストに対する応答性が悪化する。ま

た、画像処理モジュールM1、M2、M3においても、直前に受け付けたリクエストを優先するため、特定の画像処理モジュールにバスを占有されてしまう可能性がある。そこで、特定のバスマスタがバスを占有しないように、連続アクセス数に制限を付ける。

【００６６】

図6は連続したバスアクセスをカウントするアルゴリズムを説明するフローチャートである。

【００６７】

まず、現在リクエストを出しているモジュール数Nrを検出し(S221)、現在のリクエスト識別コードIDと直前のリクエスト識別コードレジスタID1の値を比較し(S222)、Nr≦1またはID1≠IDであればカウンタCを零にリセットする(S223)。また、Nr＞1かつID1=IDであれば、ステップS224の判定により、対象モジュールに受領信号PACKを返す度にカウンタCをインクリメントする(S224)。

【００６８】

図6の処理を繰り返すことで、カウンタCには同一モジュールによる連続したバスアクセスの回数（以下「連続バスアクセス数」と呼ぶ）がカウントされる。そして、アービタ5は、カウンタCが所定値になる（C＜Cth）と(S226)、直前のリクエスト識別コードレジスタID1が示すモジュールのリクエストをマスクする(S227)。

【００６９】

このように、次には必ず異なるモジュールのリクエストが受け付けられるため、連続バスアクセス数は制限されることになる。なお、一度、異なるモジュールにバスの使用権が移れば、ステップS221～S223によってカウンタCがリセットされるので、リクエストのマスクは解除される(S228)。

【００７０】

なお、図6には、連続バスアクセス数をカウントするカウンタを全モジュールで共通に使用する例を説明したが、モジュールごとにカウンタを設けてもよい。この場合、モジュールごとに連続バスアクセス数を制限することができる。例えばバスブリッジB0の制限値をCPU 1のキャッシュのラインサイズに合わせておけば、効率的にキャッシュを更新することができる。

【００７１】

また、各モジュールがDRAM 7のそれぞれ異なるバンクをアクセスする場合、バスアクセスの連続性を考慮する必要はない。従って、現在のリクエスト識別コードIDと直前のリクエスト識別コードレジスタID1の比較は不要になる。

【００７２】

このように、バッファの空き状態、直前のアクセス、連続アクセス数に応じてバスアクセスの優先順位を動的に変更（制御）することで、バスアクセスの連続性を向上させつつ、バスマスタのバス使用頻度に合わせた調停が可能になる。

【００７３】

また、アービタ5だけではなく各モジュール内で調停を実行する（調停の分散処理）ため、モジュールに最適な調停が可能である。例えば、上述したように、各バスマスタのリクエスト発生頻度が異なる場合は、発生頻度の低い順にポインタ評価と連続アクセス数の評価を行う。これは、発生頻度の低い方を優先することで、発生頻度の低い方のバス使用権の獲得確率を高め、処理の平準化を図るためである。ただし、発生頻度がほぼ同じ場合は、ライト側を優先する。これは、一般にライト処理の方がレイテンシ（待ち時間）が少なく、メモリバスが解放されるまでの時間が短いためである。

【実施例２】

【００７４】

以下、本発明にかかる実施例2の画像処理を説明する。なお、実施例2において、実施例1と略同様の構成については、同一符号を付して、その詳細説明を省略する。

【００７５】

図7は実施例2の画像処理モジュールの詳細な構成例を説明するブロック図で、図2に示

す画像処理モジュールとの違いは、リードバッファ10、ライトバッファ12、リードアドレス発生器14およびライトアドレス発生器16がそれぞれ二つずつあることである。

【００７６】

アービタ13は、リードバッファ10aのバッファの空き容量R0pおよびリードアドレス発生器14aのイネーブル信号R0enを検出し、リードアドレスが有効（R0en=‘１’）でリードバッファ10aにデータが格納可能（R0p≧R0n）であれば、アービタ5へリードリクエスト（PREQ=‘１’、PNRW=‘０’、PNUM=Rn、PADD=Rad）を発行する。同様に、リードバッファ10bのバッファの空き容量R1pおよびリードアドレス発生器14bのイネーブル信号R1enを検出し、リードアドレスが有効（R1en=‘１’）でリードバッファ10bにデータが格納可能（R1p≧R1n）であれば、アービタ5へリードリクエスト（PREQ=‘１’、PNRW=‘０’、PNUM=Rn、PADD=Rad）を発行する。

【００７７】

一方、ライトバッファ12aのデータ蓄積数W0pが所定のワード数以上（W0p≧W0n）になると、アービタ13は、ライトアドレス発生器16bのイネーブル信号W0enを検出し、ライトアドレスが有効（W0en=‘１’）であれば、アービタ5へライトリクエスト（PREQ=‘１’、PNRW=‘１’、PNUM=Wn、PADD=Wad）を発行する。同様に、ライトバッファ12bのデータ蓄積数W0pが所定のワード数以上（W1p≧W1n）になると、ライトアドレス発生器16bのイネーブル信号W1enを検出し、ライトアドレスが有効（W1en=‘１’）であれば、アービタ5へライトリクエスト（PREQ=‘１’、PNRW=‘１’、PNUM=Wn、PADD=Wad）を発行する。

【００７８】

図8は実施例2のアービタ13のアルゴリズムを説明するフローチャートである。実施例1と同様に、リクエストキューに蓄積されているリクエストが実行された場合のライトバッファ12aのデータ蓄積数（評価値）をPw0、ライトバッファ12bのデータ蓄積数（評価値）をPw1、リクエストキューに蓄積されているリクエストが実行された場合のリードバッファ10aの空き容量（評価値）をPr0、リードバッファ10bの空き容量（評価値）をPr1とする。なお、リクエストキューに蓄積されているリクエスト数Ppは、アービタ5がリクエストを受け付ける（PACK=‘１’）と一つ減じられる。以下の説明では（リードバッファ10aのリクエストの頻度）＞（リードバッファ10bのリクエストの頻度）=（ライトバッファ12bのリクエストの頻度）＞（ライトバッファ12aリクエストの頻度）であると仮定し、バスアクセスのリクエスト発生頻度が低いモジュールの順に、バッファの空き状態の検出を行う。

【００７９】

ライトバッファ12aのリクエストW0reqは、ライトバッファ12aの蓄積数の評価値Pw0が所定のワード数以上（Pw0≧W0n）になり、ライトアドレスが有効（W0en=‘１’）ならば、W0req=‘１’になる。同様に、ライトバッファ12bのリクエストW1reqは、ライトバッファ12bの蓄積数の評価値Pw1が所定のワード数以上（Pw1≧W1n）になり、ライトアドレスが有効（W1en=‘１’）ならば、W1req=‘１’になる。

【００８０】

また、リードバッファ10aのリクエストR0reqは、リードバッファ10aの空き容量の評価値Pr0が所定のワード数以上（Pr0≧R0n）になり、リードアドレスが有効（R0en=‘１’）ならば、R0req=‘１’になる。同様に、リードバッファ10bのリクエストR1reqは、リードバッファ10bの空き容量の評価値Pr1が所定のワード数以上（Pr1≧R1n）になり、リードアドレスが有効（R1en=‘１’）ならば、R1re=‘１’になる。

【００８１】

まず、ライトリクエストW0req=‘１’で、ライトバッファ12aのデータ蓄積数の評価値P0wと所定値W0thの関係がP0w≧W0th、または、直前にリクエストキューに格納したリクエストがライトバッファ12aのライトリクエスト（ID1=IDw0）か否かを判定し(S231)、そうであればライトバッファ12aのライトリクエストをリクエストキューに蓄積する(S239)。

【００８２】

上記の条件が成立しない場合は、ライトリクエストW1req=‘１’で、ライトバッファ12b

のデータ蓄積数の評価値P1wと所定値W1thの関係がP1w≧W1th、または、直前にリクエストキューに格納したリクエストがライトバッファ12bのライトリクエスト（ID1=IDw1）か否かを判定し(S232)、そうであればライトバッファ12bのライトリクエストをリクエストキューに蓄積する(S240)。

【００８３】

上記の二つの条件が成立せず、リードリクエストR1req=‘１’で、リードバッファ10bの空き容量の評価値Pr1と所定値R1thの関係がPr1≧R1th、または、直前にリクエストキューに格納したリクエストがリードバッファ10bのリードリクエスト（ID1=IDr1）か否かを判定し(S233)、そうであればリードバッファ10bのリードリクエストをリクエストキューに蓄積する(S242)。

【００８４】

上記の三つの条件が成立せず、リードリクエストR0req=‘１’で、リードバッファ10aの空き容量の評価値Pr0と所定値R0thの関係がPr≧R0th、または、直前にリクエストキューに格納したリクエストがリードバッファ10aのリードリクエスト（ID1=IDr0）か否かを判定し(S234)、そうであればリードバッファ10aのリードリクエストをリクエストキューに蓄積する(S241)。

【００８５】

上記の四つの条件が成立せず、リードリクエストR0req=‘１’の場合は(S235)、リードバッファ10aのリードリクエストをリクエストキューに蓄積し(S241)、リードリクエストR1req=‘１’の場合は(S236)、リードバッファ10bのリードリクエストをリクエストキューに蓄積し(S242)、ライトリクエストW1req=‘１’の場合は(S237)、ライトバッファ12bのライトリクエストをリクエストキューに蓄積し(S240)、ライトリクエストW0req=‘１’の場合は(S238)、ライトバッファ12aのライトリクエストをリクエストキューに蓄積する(S239)。

【００８６】

リクエストキューへのライトバッファ12aのライトリクエストの蓄積(S239)は、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDをライトリクエスト識別コードIDw0に更新する。同時に、ライトバッファ12aの蓄積数の評価値Pw0からライトデータ数W0nを減算して評価値Pw0を更新し、リクエストキューに蓄積されているリクエスト数Ppを一つ増やす。

【００８７】

同様に、リクエストキューへのライトバッファ12bのライトリクエストの蓄積(S240)は、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDをライトリクエスト識別コードIDw1に更新する。同時に、ライトバッファ12bの蓄積数の評価値Pw1からライトデータ数W1nを減算して評価値Pw1を更新し、リクエストキューに蓄積されているリクエスト数Ppを一つ増やす。

【００８８】

同様に、リクエストキューへのリードバッファ10aのリードリクエストの蓄積(S241)は、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDをリードリクエスト識別コードIDr0に更新する。同時に、リードバッファ10aの空き容量の評価値Pr0からリードデータ数R0nを減算して評価値Pr0を更新し、リクエストキューに蓄積されているリクエスト数Ppを一つ増やす。

【００８９】

同様に、リクエストキューへのリードバッファ10bのリードリクエストの蓄積(S242)は、現在のリクエスト識別コードIDを直前のリクエスト識別コードレジスタID1に格納し、現在のリクエスト識別コードIDをリードリクエスト識別コードIDr1に更新する。同時に、リードバッファ10bの空き容量の評価値Pr1からリードデータ数R1nを減算して評価値Pr1を更新し、リクエストキューに蓄積されているリクエスト数Ppを一つ増やす。

【００９０】

上記処理が終了すると、処理をステップS231に戻し、上記の処理を繰り返す。

【００９１】
上記のモジュールにはデータパスが二つ存在する。例えば、誤差拡散回路の誤差バッファをDRAM 7上に構成するようなモジュールは、画像データのデータパスと誤差バッファのデータバスが存在する。この場合、バスマスタの数は二個から四個に増加し、このままではメモリバスに接続することができない。このような場合、モジュール内で調停を実行して接続すれば、上層の回路を一切変更することなしに、モジュールとメモリバスを接続することが可能になる。
【００９２】
また、上述したように、アービタ5だけではなく各モジュール内で調停を実行する（調停の分散処理）ため、モジュールに最適な調停が可能である。例えば、上述したように、各バスマスタのリクエスト発生頻度が異なる場合は、発生頻度の低い順にポインタ評価と連続アクセス数の評価を行う。これは、発生頻度の低い方を優先することで、発生頻度の低い方のバス使用権の獲得確率を高め、処理の平準化を図るためである。ただし、発生頻度がほぼ同じ場合は、ライト側を優先する。これは、一般にライト処理の方がレイテンシ（待ち時間）が少なく、メモリバスが解放されるまでの時間が短いためである。
【００９３】
また、各モジュール内で、図6に示した連続アクセス数をカウントするカウンタを用いて、連続アクセス数を制限するようにしてもよい。この場合、上記の発生頻度を考慮しなくとも、処理の平準化を図ることができる。
【００９４】
さらに、上記評価値（Pw0、Pw1、Pr0、Pr1）との比較のための閾値（W0th、W1th、R0th、R1th）を複数設定し、より細やかな調停を行うことも可能である。この場合は、閾値の大きさ順に優先順位を設定し、最も大きな閾値との比較時に連続アクセス数を判定すればよい。このように、モジュール単位の最適化が可能になるので、容易にメモリバスの使用効率を向上させることができる。
【００９５】
以上では、メモリバスのアクセスを調停する例を実施例として説明したが、本発明は、メモリバスに限らず、様々なバスの使用権のアービトレーションに適用可能である。
【００９６】
［他の実施例］
なお、本発明は、複数の機器（例えばホストコンピュータ、インタフェイス機器、リーダ、プリンタなど）から構成されるシステムに適用しても、一つの機器からなる装置（例えば、複写機、ファクシミリ装置など）に適用してもよい。
【００９７】
また、本発明の目的は、前述した実施例の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体（または記録媒体）を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはCPUやMPU）が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、達成されることは言うまでもない。この場合、記憶媒体から読み出されたプログラムコード自体が前述した実施例の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施例の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム(OS)などが実際の処理の一部または全部を行い、その処理によって前述した実施例の機能が実現される場合も含まれることは言うまでもない。
【００９８】
さらに、記憶媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によって前述した実施例の

機能が実現される場合も含まれることは言うまでもない。

【００９９】

本発明を上記記憶媒体に適用する場合、その記憶媒体には、先に説明したフローチャートに対応するプログラムコードが格納されることになる。

【図面の簡単な説明】

【０１００】

【図１】実施例1の画像処理装置の構成例を示すブロック図、

【図２】画像処理モジュールの詳細な構成例を説明するブロック図、

【図３】サブモジュール間のデータ転送を説明するタイミングチャート、

【図４】アービタ13のアルゴリズムを説明するフローチャート、

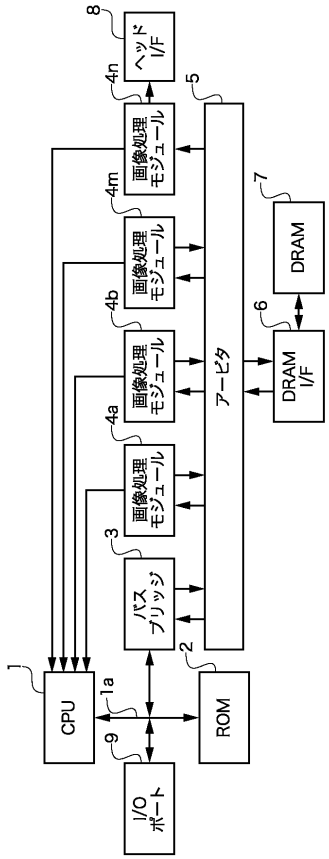
【図５】アービタ5のアルゴリズムを説明するフローチャート、

【図６】連続したバスアクセスをカウントするアルゴリズムを説明するフローチャート、

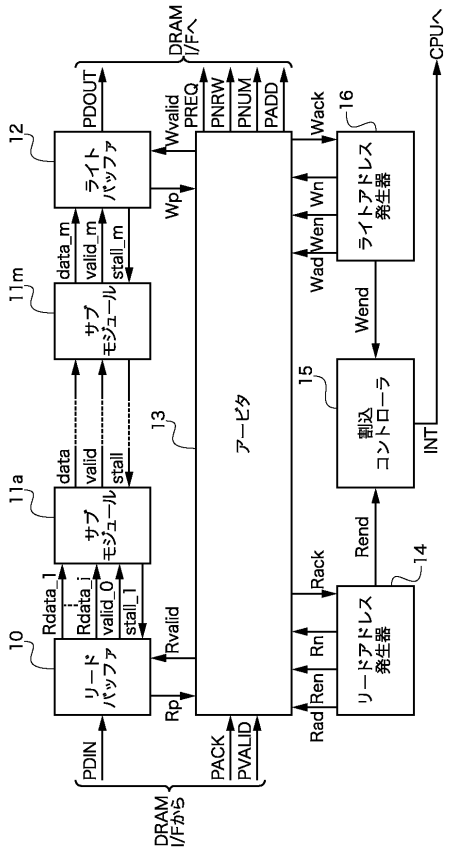
【図７】実施例2の画像処理モジュールの詳細な構成例を説明するブロック図、

【図８】実施例2のアービタ13のアルゴリズムを説明するフローチャートである。

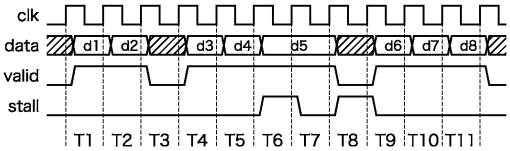
【図１】



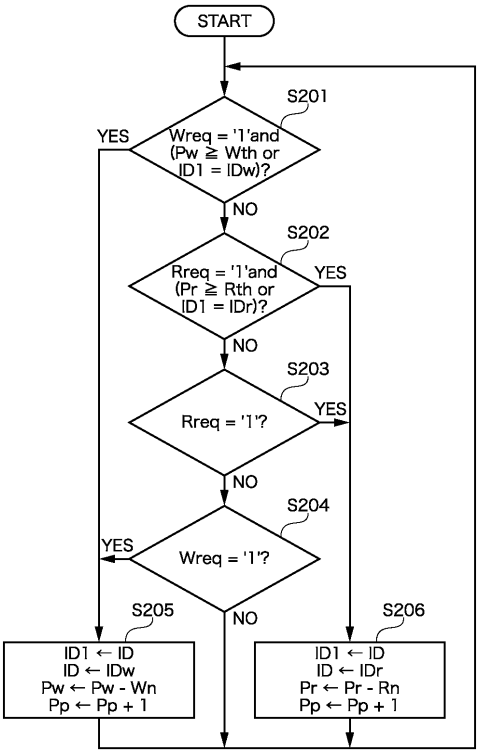
【図２】



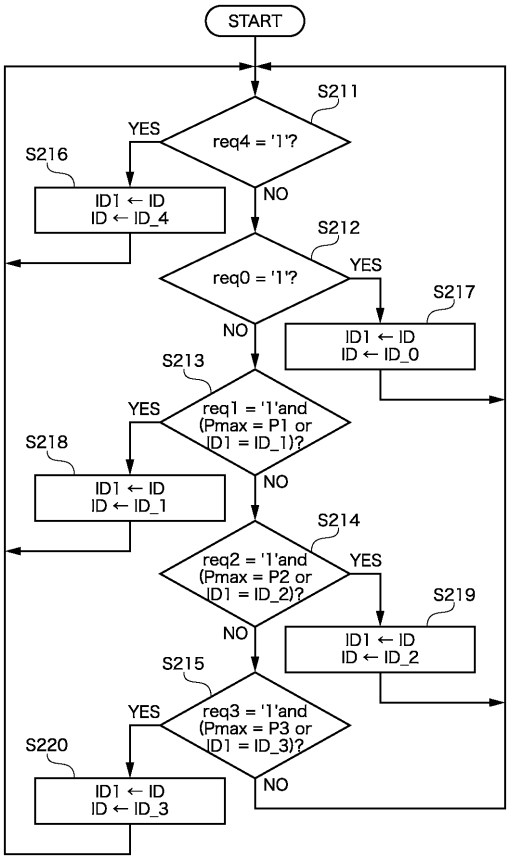
【図３】


clk
data
valid
stall
d1
d2
d3
d4
d5
d6
d7
d8
T1
T2
T3
T4
T5
T6
T7
T8
T9
T10
T11

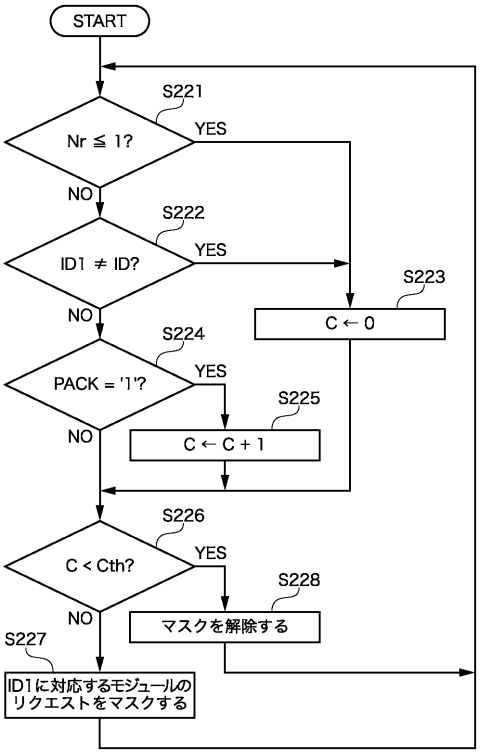
【図４】


START
S201
Wreq = '1'and (Pw ≧ Wth or ID1 = IDw)?
YES
NO
S202
Rreq = '1'and (Pr ≧ Rth or ID1 = IDr)?
YES
NO
S203
Rreq = '1'?
YES
NO
S204
Wreq = '1'?
YES
NO
S205
ID1 ← ID
ID ← IDw
Pw ← Pw - Wn
Pp ← Pp + 1
S206
ID1 ← ID
ID ← IDr
Pr ← Pr - Rn
Pp ← Pp + 1

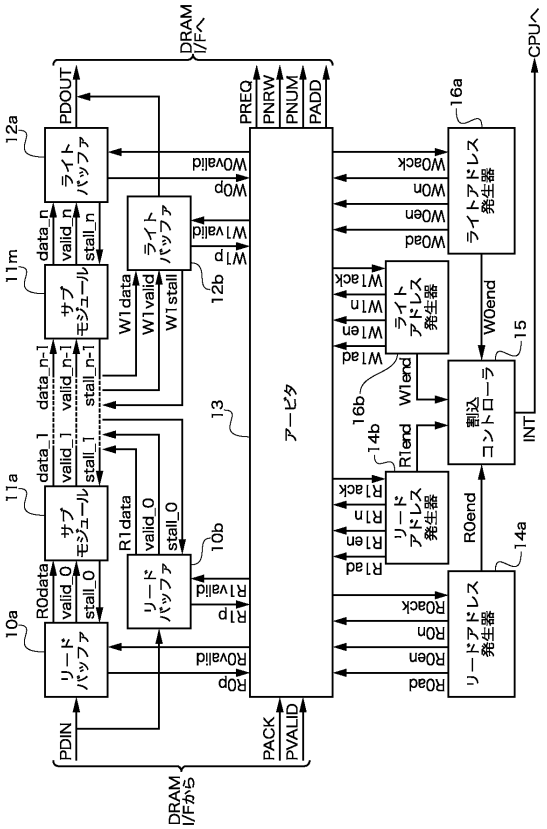
【図５】


START
S211
req4 = '1'?
YES
NO
S216
ID1 ← ID
ID ← ID_4
S212
req0 = '1'?
YES
NO
S217
ID1 ← ID
ID ← ID_0
S213
req1 = '1'and (Pmax = P1 or ID1 = ID_1)?
YES
NO
S218
ID1 ← ID
ID ← ID_1
S214
req2 = '1'and (Pmax = P2 or ID1 = ID_2)?
YES
NO
S219
ID1 ← ID
ID ← ID_2
S215
req3 = '1'and (Pmax = P3 or ID1 = ID_3)?
YES
NO
S220
ID1 ← ID
ID ← ID_3

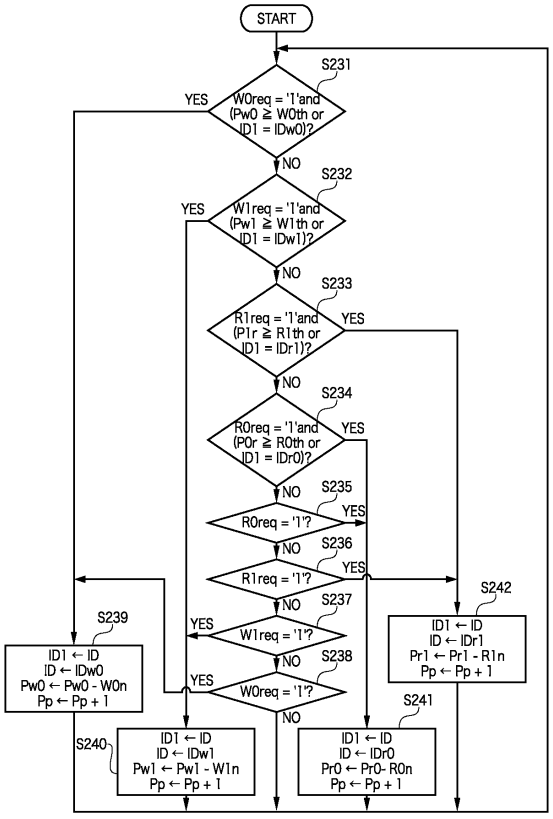
【図６】


START
S221
Nr ≦ 1?
YES
NO
S222
ID1 ≠ ID?
YES
NO
S223
C ← 0
S224
PACK = '1'?
YES
NO
S225
C ← C + 1
S226
C < Cth?
YES
NO
S228
マスクを解除する
S227
ID1に対応するモジュールのリクエストをマスクする

【図７】


DRAM I/Fへ
PDOUT
PREQ
PNRW
PNUM
PADD
CPUへ
ライトバッファ
ライトアドレス発生器
サブモジュール
リードバッファ
リードアドレス発生器
アービタ
割込コントローラ
INT
W0end
W1end
R0end
R1end
PDIN
PACK
PVALID
DRAM I/Fから

【図８】


START
S231
W0req = '1'and (Pw0 ≧ W0th or ID1 = IDw0)?
S232
W1req = '1'and (Pw1 ≧ W1th or ID1 = IDw1)?
S233
R1req = '1'and (P1r ≧ R1th or ID1 = IDr1)?
S234
R0req = '1'and (P0r ≧ R0th or ID1 = IDr0)?
S235
R0req = '1'?
S236
R1req = '1'?
S237
W1req = '1'?
S238
W0req = '1'?
S239
ID1 ← ID
ID ← IDw0
Pw0 ← Pw0 - W0n
Pp ← Pp + 1
S240
ID1 ← ID
ID ← IDw1
Pw1 ← Pw1 - W1n
Pp ← Pp + 1
S241
ID1 ← ID
ID ← IDr0
Pr0 ← Pr0- R0n
Pp ← Pp + 1
S242
ID1 ← ID
ID ← IDr1
Pr1 ← Pr1 - R1n
Pp ← Pp + 1
YES
NO

フロントページの続き

(72)発明者　石川　尚
東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内

審査官　木村　貴俊

(56)参考文献　特開２００２－１３２７０７（ＪＰ，Ａ）
特開平１１－３３８８２０（ＪＰ，Ａ）
特開２００３－２２８５１２（ＪＰ，Ａ）
特開平０５－２５０３１３（ＪＰ，Ａ）
特開平０８－２５５１２６（ＪＰ，Ａ）
特開昭６４－０８２２４８（ＪＰ，Ａ）
特開平１０－１８７６０１（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　３／０６－　３／０８
Ｇ０６Ｆ　１３／２０－１３／４２