(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3562215号
(P3562215)
(45)発行日　平成16年9月8日(2004.9.8)　(24)登録日　平成16年6月11日(2004.6.11)

(51)Int.Cl.[7]　F I
G06F 1/04　G06F 1/04　301B
G06F 9/38　G06F 9/38　310E
G06F 9/38　350A

請求項の数 10　(全 26 頁)

(21)出願番号　特願平9-137710
(22)出願日　平成9年5月13日(1997.5.13)
(65)公開番号　特開平10-312222
(43)公開日　平成10年11月24日(1998.11.24)
審査請求日　平成14年12月12日(2002.12.12)

(73)特許権者　000002369
セイコーエプソン株式会社
東京都新宿区西新宿２丁目４番１号
(74)代理人　100090479
弁理士　井上　一
(74)代理人　100090387
弁理士　布施　行夫
(74)代理人　100090398
弁理士　大渕　美千栄
(72)発明者　久保田　哲
長野県諏訪市大和３丁目３番５号　セイコーエプソン株式会社内

審査官　山崎　慎一

最終頁に続く

(54)【発明の名称】マイクロコンピュータ及び電子機器

(57)【特許請求の範囲】
【請求項１】
各パイプラインステージの処理のための制御信号の状態が格納されるパイプラインレジスタを有し、発行された各命令のパイプラインステージをパイプライン制御により並列処理する中央処理ユニットと、
前記パイプラインレジスタに供給される第１のクロックを制御するクロック制御回路とを含み、
前記クロック制御回路が、
並列処理されるパイプラインステージの中の少なくとも１つが所与の要因によりストールした場合に、前記第１のクロックを停止する制御を行い、前記中央処理ユニットのパイプライン処理を停止させることを特徴とするマイクロコンピュータ。
【請求項２】
請求項１において、
前記クロック制御回路が、
命令のフェッチ要求に対する第１のアクノリッジ信号が少なくとも１クロックサイクル遅延した場合、命令データの第１のバリッド信号が少なくとも１クロックサイクル遅延した場合、データのアクセス要求に対する第２のアクノリッジ信号が少なくとも１クロックサイクル遅延した場合、データの第２のバリッド信号が少なくとも１クロックサイクル遅延した場合、前記中央処理ユニットが内蔵する汎用レジスタのインターロックが発生した場合、前記中央処理ユニットのバスアクセス要求よりも高い優先順位を持つバスアクセス要

求が発生した場合の少なくとも１つの場合に、前記第１のクロックを停止する制御を行うことを特徴とするマイクロコンピュータ。

【請求項３】

請求項２において、

前記クロック制御回路が、

前記第１のアクノリッジ信号の未返却数が０であることを表すステート、１であることを表すステート及び２であることを表すステートを有する第１のステートマシーン、前記第１のバリッド信号の未返却数が０であることを表すステート、１であることを表すステート及び２であることを表すステートを有する第２のステートマシーン、前記第２のアクノリッジ信号の未返却数が０であることを表すステート、１であることを表すステート及び２であることを表すステートを有する第３のステートマシーン、前記第２のバリッド信号の未返却数が０であることを表すステート、１であることを表すステート及び２であることを表すステートを有する第４のステートマシーンの少なくとも１つを含むことを特徴とするマイクロコンピュータ。

【請求項４】

請求項１乃至３のいずれかにおいて、

前記クロック制御回路が、

前記中央処理ユニットを動作させる第２のクロックを、パイプラインステージが前記所与の要因によりストールした場合にも停止しないように制御することを特徴とするマイクロコンピュータ。

【請求項５】

請求項１乃至４のいずれかにおいて、

前記中央処理ユニットが、

前記クロック制御回路に出力され且つ１つのバスアクセス要求について１クロックサイクル期間だけアクティブになる第１のリクエスト信号と、前記中央処理ユニットからのバスアクセス要求を受け付けるバス制御ユニットに対して出力され且つ前記第１のリクエスト信号がアクティブになってから前記バス制御ユニットからのアクノリッジ信号がアクティブになるまでの期間アクティブになる第２のリクエスト信号とを生成するインターフェース回路を含むことを特徴とするマイクロコンピュータ。

【請求項６】

請求項１乃至５のいずれかにおいて、

前記クロック制御回路が、

前記中央処理ユニットの制御の下で動作する所与の回路ユニットを動作させる第３のクロックを、パイプラインステージが前記所与の要因によりストールした場合及び前記回路ユニットを使用しない命令を処理する場合のいずれかの場合に停止する制御を行うことを特徴とするマイクロコンピュータ。

【請求項７】

請求項１乃至６のいずれかにおいて、

前記中央処理ユニットからのバスアクセス要求を受け付けるバス制御ユニットと、

前記バスアクセス要求に対する前記バス制御ユニットの応答のウェイトサイクル数を制御するウェイト制御回路とを含むことを特徴とするマイクロコンピュータ。

【請求項８】

各パイプラインステージの処理のための制御信号の状態が格納されるパイプラインレジスタを有し、発行された各命令のパイプラインステージをパイプライン制御により並列処理する中央処理ユニットと、

前記パイプラインレジスタに供給される第１のクロックを制御するクロック制御回路と、

前記中央処理ユニットからのバスアクセス要求を受け付けるバス制御ユニットと、

前記バスアクセス要求に対する前記バス制御ユニットの応答のウェイトサイクル数を制御するウェイト制御回路と、

前記バス制御ユニットの応答が遅延した場合に、前記第１のクロックを停止する制御を行い、前記中央処理ユニットのパイプライン処理を停止させるクロック制御回路とを含むことを特徴とするマイクロコンピュータ。
【請求項９】
請求項７又は８において、
前記ウェイト制御回路が、
所与の設定値に基づき決定される周期でアクティブになる周期信号を生成する回路と、
前記中央処理ユニットからのリクエスト信号がアクティブとなり且つ前記周期信号がアクティブとなった場合にアクティブになるアクノリッジ信号を生成する回路とを含むことを特徴とするマイクロコンピュータ。
【請求項１０】
請求項１乃至９のいずれかのマイクロコンピュータと、
前記マイクロコンピュータの処理対象となるデータの入力源と、
前記マイクロコンピュータにより処理されたデータを出力するための出力装置とを含むことを特徴とする電子機器。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、マイクロコンピュータ及び電子機器に関する。
【０００２】
【背景技術及び発明が解決しようとする課題】
近年、家庭用のゲーム装置、カーナビゲーションシステム、プリンタ、携帯情報端末などの電子機器に組み込まれ、高度な情報処理を実現できるマイクロコンピュータに対する需要が高まっている。
【０００３】
このようなマイクロコンピュータでは、処理の高速化を図ることが重要な課題となっており、これを達成するものとしてパイプライン処理と呼ばれる技術が知られている。パイプライン処理では、命令の処理を複数のステージ（パイプラインステージ）に分け各命令のステージを並列処理する。即ち命令の処理を、例えば命令フェッチ、命令デコード、レジスタリード、アドレス計算、命令実行、レジスタライトのステージに分ける。そして例えば第１の命令によるレジスターリードやアドレス計算を行っている際に、第２の命令のデコードやフェッチを行う。このように命令をパイプラインにより処理することで、命令の処理のスループットを格段に向上できる。
【０００４】
しかしながら、このパイプライン処理においては、データの依存関係等に起因するストールの問題がある。この問題は例えば、第１の命令でメモリからロードしたデータを直後の第２の命令で使用する場合等に生じる。パイプラインを採用して処理の高速化を図りながらも、回路の小規模化を図るためには、ストール発生時におけるパイプライン制御をできる限り簡易化することが望まれる。
【０００５】
一方、マイクロコンピュータは携帯用の電子機器などに使用されることが多く、このような電子機器においては電池駆動による長時間動作への要求が高まっている。従って、このような電子機器に組み込まれるマイクロコンピュータは、できる限り低消費電力であることが望まれる。
【０００６】
本発明は、以上のような技術的課題に鑑みてなされたものであり、その目的とするところは、ストールが発生した場合にも簡易にパイプライン制御できると共に、省電力を実現できるマイクロコンピュータ及び電子機器を提供することにある。
【０００７】
【課題を解決するための手段】

上記課題を解決するために本発明に係るマイクロコンピュータは、発行された各命令のパイプラインステージをパイプライン制御により並列処理する中央処理ユニットと、前記中央処理ユニットを動作させる第１のクロックを制御するクロック制御回路とを含み、前記クロック制御回路が、並列処理されるパイプラインステージの中の少なくとも１つが所与の要因によりストールした場合に、前記第１のクロックを停止する制御を行うことを特徴とする。

【０００８】

本発明によれば、パイプラインが所与の要因によりストールした場合に、中央処理ユニットを動作させる第１のクロックが停止する。そしてストールの解消後に第１のクロックの停止が解除され、中央処理ユニットの動作が再開する。本発明によれば、パイプライン制御される回路間の同期を保つことができるため、ストール時のパイプライン制御を簡易化できる。またストール時に中央処理ユニットの大部分の動作を停止できるため、省電力化を図れる。

【０００９】

また本発明は、前記クロック制御回路が、命令のフェッチ要求に対する第１のアクノリッジ信号が少なくとも１クロックサイクル遅延した場合、命令データの第１のバリッド信号が少なくとも１クロックサイクル遅延した場合、データのアクセス要求に対する第２のアクノリッジ信号が少なくとも１クロックサイクル遅延した場合、データの第２のバリッド信号が少なくとも１クロックサイクル遅延した場合、前記中央処理ユニットが内蔵する汎用レジスタのインターロックが発生した場合、前記中央処理ユニットのバスアクセス要求よりも高い優先順位を持つバスアクセス要求が発生した場合の少なくとも１つの場合に、前記第１のクロックを停止する制御を行うことを特徴とする。例えば命令フェッチ要求に対する第１のアクノリッジ信号が遅延した場合には、第１のクロックを停止し、第１のアクノリッジ信号がアクティブになるのを待つことでストールを解消できる。またレジスタ・インターロックが生じた場合には、第１のクロックを停止し、レジスタ・インターロックが解除されるのを待つことでストールを解消できる。またＤＭＡリクエストや外部バスマスタによるリクエストなどの高い優先順位を持つバスアクセス要求が発生した場合には、第１のクロックを停止し、そのバスアクセス要求の処理が完了するのを待つことでストールを解消できる。

【００１０】

また本発明は、前記クロック制御回路が、前記第１のアクノリッジ信号の未返却数が０であることを表すステート、１であることを表すステート及び２であることを表すステートを有する第１のステートマシーン、前記第１のバリッド信号の未返却数が０であることを表すステート、１であることを表すステート及び２であることを表すステートを有する第２のステートマシーン、前記第２のアクノリッジ信号の未返却数が０であることを表すステート、１であることを表すステート及び２であることを表すステートを有する第３のステートマシーン、前記第２のバリッド信号の未返却数が０であることを表すステート、１であることを表すステート及び２であることを表すステートを有する第４のステートマシーンの少なくとも１つを含むことを特徴とする。このように未返却数が０、１、２であることを表すステートを持たせることで、例えばリクエスト信号が連続して出され且つアクノリッジ信号の返却が遅延した場合にも、正常な動作を保証できることになる。

【００１１】

また本発明は、前記クロック制御回路が、前記中央処理ユニットを動作させる第２のクロックを、パイプラインステージが前記所与の要因によりストールした場合にも停止しないように制御することを特徴とする。このような第２のクロックを、ストール時にも動作する必要がある回路ユニットに供給することで、マイクロコンピュータの正常動作を保証できる。

【００１２】

また本発明は、前記第２のクロックが、前記中央処理ユニットが内蔵する汎用レジスタへのデータアクセスを制御するレジスタ制御回路及び前記クロック制御回路の少なくとも一

方を動作させるクロックであることを特徴とする。このようにレジスタ制御回路を第２のクロックにより動作させることで、レジスタへのデータ書き込み時の誤動作等を防止できる。またクロック制御回路を第２のクロックにより動作させることで、第１のクロックの停止処理を適正に行うことが可能となる。

【００１３】

また本発明は、前記中央処理ユニットが、前記クロック制御回路に出力され且つ１つのバスアクセス要求について１クロックサイクル期間だけアクティブになる第１のリクエスト信号と、前記中央処理ユニットからのバスアクセス要求を受け付けるバス制御ユニットに対して出力され且つ前記第１のリクエスト信号がアクティブになってから前記バス制御ユニットからのアクノリッジ信号がアクティブになるまでの期間アクティブになる第２のリクエスト信号とを生成するインターフェース回路を含むことを特徴とする。このようにすることで、クロック制御回路は、パイプラインがストールしたか否かを第１のリクエスト信号を用いて判断できる。またバス制御ユニットは、中央処理ユニットからバスアクセス要求があったか否かを第２のリクエスト信号を用いて判断できる。このように２種類のリクエスト信号を用意することで、クロック制御回路を正常に動作させることができる共に、バス制御ユニットとのインターフェース処理を適正化できる。

【００１４】

また本発明は、前記クロック制御回路が、前記中央処理ユニットの制御の下で動作する所与の回路ユニットを動作させる第３のクロックを、パイプラインステージが前記所与の要因によりストールした場合及び前記回路ユニットを使用しない命令を処理する場合のいずれかの場合に停止する制御を行うことを特徴とする。このようにすることで、中央処理ユニットと所与の回路ユニットとの同期関係を維持できると共に、この回路ユニットの無駄な電力の消費を解消できる。

【００１５】

また本発明は、前記回路ユニットが、積和演算回路、乗算器、加算器、除算器及びバレルシフタの少なくとも１つであることを特徴とする。これらの回路ユニットは回路規模が大きく高速度で動作する。従って、ストール時や、これらの回路ユニットを使用しない命令の処理時に、これらの回路ユニットの動作を停止することで、消費電力の大幅な低減化を期待できる。

【００１６】

また本発明は、前記中央処理ユニットからのバスアクセス要求を受け付けるバス制御ユニットと、前記バスアクセス要求に対する前記バス制御ユニットの応答のウェイトサイクル数を制御するウェイト制御回路とを含むことを特徴とする。

【００１７】

本発明によれば、バス制御ユニットの応答のウェイトサイクル数を制御することで、第１のクロックの停止期間が制御される。これにより中央処理ユニットの実行速度や消費電力を自動的に制御できることになる。

【００１８】

また本発明は、前記ウェイト制御回路が、所与の設定値に基づき決定される周期でアクティブになる周期信号を生成する回路と、前記中央処理ユニットからのリクエスト信号がアクティブとなり且つ前記周期信号がアクティブとなった場合にアクティブになるアクノリッジ信号を生成する回路とを含むことを特徴とする。このような周期信号を用いてアクノリッジ信号を生成すれば、少ない回路規模で簡易にウェイトサイクル数を制御できるようになり、中央処理ユニットの実行速度や消費電力を制御できるようになる。

【００１９】

また本発明に係る電子機器は、上記のいずれかのマイクロコンピュータと、前記マイクロコンピュータの処理対象となるデータの入力源と、前記マイクロコンピュータにより処理されたデータを出力するための出力装置とを含むことを特徴とする。このようにすれば、小規模で省電力のマイクロコンピュータを電子機器に組み込むことが可能となり、電子機器の低コスト化、コンパクト化、低消費電力化を図ることができる。

【００２０】
【発明の実施の形態】
以下、本発明の好適な実施形態について図面を用いて詳細に説明する。
【００２１】
（実施例１）
１．構成
実施例１は、パイプラインステージがストールした場合に、ＣＰＵ（中央処理ユニット）の動作クロックを停止する実施例である。
【００２２】
図１にマイクロコンピュータ１００のブロック図の例を示す。このブロック図は、実施例１、２、３に共通に使用されるブロック図である。
【００２３】
ここでマイクロコンピュータ１００は、発行された命令を処理するＣＰＵ１０２、バスの制御を行うＢＣＵ（バス制御ユニット）１０８、命令やデータを記憶する内部メモリ１１０を含む。なおマイクロコンピュータ１００の外部には外部アドレスバス、外部データバスが設けられており、これらの外部アドレスバス、外部データバスには、外部メモリ１１４などの各種の周辺装置が接続される。
【００２４】
ＢＣＵ（バス制御ユニット）１０８は、各種のバスの制御を行うものであり、これにより、例えばＣＰＵ１０２と、内部メモリ１１０、外部メモリ１１４との間でのデータ転送等が可能となる。
【００２５】
Ｉ＿ＡＤＤＲ＿ＢＵＳは命令アドレスバスであり、Ｉ＿ＤＡＴＡ＿ＢＵＳは命令データバスである。これらのバスを用いて内部メモリ１１０や外部メモリ１１４から命令が読み出される。またＤ＿ＡＤＤＲ＿ＢＵＳはデータアドレスバスであり、Ｄ＿ＤＡＴＡ＿ＢＵＳはデータバスである。これらのバスを用いて内部メモリ１１０や外部メモリ１１４から命令処理に必要なデータが読み出される。このように本実施例のマイクロコンピュータではいわゆるハーバードアーキテクチャのバス構成を採用している。
【００２６】
ＣＰＵ１０２は、命令デコーダ１２０、ＰＣ（プログラムカウンタ）１２６、汎用レジスタ１３０、レジスタ制御回路１３１、ＡＬＵ１３６、積和演算回路１４０、クロック制御回路１０、インターフェース回路３０などを含む。
【００２７】
ここで命令デコーダ１２０は、Ｉ＿ＤＡＴＡ＿ＢＵＳからＩＲＩＮを介して入力された命令を受け付けると共に解析し、命令の実行に必要な種々の制御信号を出力する。例えば命令に応じた種々の指示を、即値生成器１２２を介してＣＰＵ１０２の各部に与える。
【００２８】
即値生成器１２２は、命令に含まれる即値に基づき、命令の実行時に使用する３２ビットの即値データを生成したり、各命令の実行に必要な０、±１、±２、±４のｃｏｎｓｔａｎｔデータを生成したりする。ＰＣインクリメンタ１２４は、１つの命令を実行する毎にＰＣ１２６の値をインクリメントする処理を行う。アドレス加算器１２８は、各種レジスタに格納されている情報や即値生成器１２２で生成される即値データを用いて加算処理を行い、内部メモリ１１０や外部メモリ１１４からの読み出し処理に必要なアドレスを生成する。
【００２９】
汎用レジスタ１３０は、１６本の３２ビットのレジスタＲ０～Ｒ１５を含んでおり、読み出しポート１１８、１１９を有する。レジスタ制御回路１３１は、汎用レジスタ１３０へのデータアクセス（データ書き込み・読み出し）を制御する。ＳＰ１３２は、スタックポインタ専用の３２ビットのレジスタであり、スタックの先頭番地を指すスタックポインタを格納する。ＰＳＲ（プロセッサステータスレジスタ）１３４は、各種のフラグを格納す

る３２ビットのレジスタである。
【００３０】
ＡＬＵ（算術論理演算ユニット）１３６は、ＣＰＵ１０２の制御の下で、加減演算などの算術演算や論理和、論理積、論理シフトなどの論理演算を行うものである。ゼロディテクタ１３８は、ＡＬＵ１３６の演算結果が零か否かを検出するものである。積和演算回路１４０は、バスマルチプレクサ１３９、ＡＵＸ＿ＢＵＳを介して入力されるデータに基づき、ＣＰＵ１０２の制御の下で積和演算を実行するものであり、図示しない乗算器、加算器、積和結果レジスタを含む。この積和演算回路１４０は乗算器や除算器としても機能する。バスバイパス１４２は、所与の期間にバスのバイパスを行う回路である。
【００３１】
クロック制御回路１０は、ＣＰＵ１０２を動作させるクロックＣＬＫ１、ＣＬＫ２の制御を行う。インターフェース回路３０は、各種のインターフェース信号（リクエスト信号、アクノリッジ信号、データバリッド信号等）を用いて、ＢＣＵ１０８との間のデータ転送のインターフェースを行う。
【００３２】
なおＰＡ＿ＢＵＳ、ＰＢ＿ＢＵＳ、ＷＷ＿ＢＵＳ、ＸＡ＿ＢＵＳは、ＣＰＵ１０２の内部バスである。ＩＡ、ＤＡは、各々、ＣＰＵ１０２からＩ＿ＡＤＤＲ＿ＢＵＳ、Ｄ＿ＡＤＤＲ＿ＢＵＳにアドレスを出力するためのものである。ＤＩＮは、Ｄ＿ＤＡＴＡ＿ＢＵＳからのデータをＣＰＵ１０２に入力するためのものであり、ＤＯＵＴは、ＣＰＵ１０２からのデータをＤ＿ＤＡＴＡ＿ＢＵＳに出力するためのものである。
【００３３】
２．本実施例の特徴及びパイプラインのストール
さて本実施例の特徴は、パイプラインステージがストールした場合に、ＣＰＵ１０２の一部を除く大部分を動作させるクロックであるＣＬＫ１（第１のクロック）を停止する点にある。ＣＬＫ１を停止する制御は図１のクロック制御回路１０が行う。クロック制御回路１０には、マイクロコンピュータ１００の内部に設けられる図示しない発振回路或いは外部から、クロックＣＬＫが供給される。そしてパイプラインが後述する所与の要因によりストールした場合にＣＬＫを所与の期間だけマスクして、これをＣＬＫ１として出力する。これによりストールが生じた場合にもパイプラインを適正に制御できる。またＣＬＫ１が停止することにより無駄な電力の消費を回避できる。
【００３４】
なお本実施例のクロック制御回路１０は、ストールが発生した場合にも停止しないクロックであるＣＬＫ２（第２のクロック）も出力する。ＣＰＵ１０２の回路ユニットの中で、ストールが発生した場合にも動作させる必要がある回路ユニットは、このＣＬＫ２により動作させる。
【００３５】
次にパイプラインのストールについて説明する。ストールは、ＣＰＵ１０２のバスアクセス（メモリアクセス）要求が待たされた場合（図２（Ａ））や、レジスタのインターロック（データ・ハザード）が生じた場合（図２（Ｂ））などに発生する。なお図２（Ａ）、（Ｂ）に示すように本実施例では、各命令の処理は、Ｆ（命令フェッチ）、Ｄ（命令デコード）、Ｒ（レジスタリード）、Ａ（アドレス計算）、Ｅ（命令実行）、Ｗ（レジスタライト）というように６段のステージに分けられる。また各命令の処理は３クロックサイクルで完了する。
【００３６】
図２（Ａ）のパイプライン処理では、レジスタＲ８の内容で指定されるアドレス［％Ｒ８］によりメモリからデータを読み出しレジスタＲ０に書き込む第１のロード命令が、まず処理される。次にレジスタＲ９の内容で指定されるアドレス［％Ｒ９］によりメモリからデータを読み出しレジスタＲ１に書き込む第２のロード命令が処理される。次に上記第１、第２のロード命令によりレジスタＲ０、Ｒ１に書き込まれたデータを加算し、加算結果をレジスタＲ０に書き込む加算命令が処理される。この時、アドレス［％Ｒ８］は外部メ

モリ１１４のアドレスであるため、上記第２のロード命令と異なり上記第１のロード命令によるバスアクセス要求は例えば１クロックサイクルだけ待たされる。

【００３７】

本実施例では、このようにバスアクセス要求が待たされパイプラインがストールした場合に、図２（Ａ）のＤ１に示すようにＣＰＵ１０２の大部分を動作させるクロックであるＣＬＫ１を停止する。このようにすることで、アドレス［％Ｒ８］から読み出されたデータをレジスタＲ０に適正に書き込めると共に（Ｄ２）、第２のロード命令や加算命令を適正に実行できるようになる（Ｄ３）。即ち、ストールが生じた場合にもパイプラインを適正に制御できる。またＣＰＵ１０２の大部分を動作させるクロックであるＣＬＫ１が停止することで、無駄な電力の消費を回避できる。またパイプラインがストールした場合に、自動的に省電力モードにすることが可能となる。

【００３８】

図２（Ｂ）のパイプライン処理では、レジスタＲ３の内容で指定されるアドレス［％Ｒ３］によりメモリからデータを読み出しレジスタＲ１に書き込むロード命令が、まず処理される。次にレジスタＲ５に書き込まれたデータと上記ロード命令によりレジスタＲ１に書き込まれたデータを加算して、加算結果をレジスタＲ５に書き込む加算命令が処理される。このように第１の命令でメモリからレジスタにデータをロードし、その直後の第２の命令でそのレジスタのデータを使用すると、いわゆるレジスタ・インタロックが発生する。即ち加算命令を適正に実行するためにはＤ４の時点でレジスタＲ１にデータが書き込まれている必要があるが、図２（Ｂ）ではＤ５の時点でデータが書き込まれるため加算命令を適正に実行できない。

【００３９】

そこで本実施例では、このようなレジスタ・インタロックが生じパイプラインがストールした場合に、Ｄ６に示すようにＣＬＫ１を停止する。これにより加算命令のＥステージがＤ７に示す位置にシフトするため、加算命令を適正に実行できると共に、省電力化も図れる。

【００４０】

バスアクセス要求が待たされた場合のパイプライン制御について図３（Ａ）、（Ｂ）、（Ｃ）を用いて更に詳しく説明する。図３（Ａ）に示すように、ＣＰＵ１０２（インターフェース回路３０）とＢＣＵ１０８は、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌ、ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌ、ＣＰＵ＿ＩＲ＿ＶＬＤ＿１Ｌ、ＣＰＵ＿ＤＡＴＡ＿ＲＥＱ＿１Ｌ、ＣＰＵ＿ＤＡＴＡ＿ＡＣＫ＿１Ｌ及びＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌを用いてデータ転送を制御している。なお上記の”１Ｌ”の表記は、クロックの立ち上がりで信号がアクティブになることを意味する。

【００４１】

図３（Ｂ）に、命令のフェッチ要求の際のタイミングチャートを示す。命令フェッチを要求する場合には、ＣＰＵ１０２（インターフェース回路３０）は、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌ（命令リクエスト）をアクティブ（＝１）にし（図３（Ｂ）のＥ１）、その１／２クロックサイクル後にＩＡ及びＩ＿ＡＤＤＲ＿ＢＵＳを介してＢＣＵ１０８に命令アドレスを出力する（Ｅ２）。

【００４２】

ノーウェイトで命令フェッチ要求をアクノリッジできる場合には、ＢＣＵ１０８は、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌがアクティブになってからクロックサイクル後にＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌ（命令アクノリッジ）をアクティブにする（Ｅ３）。更にノーウェイトで命令データをバリッドにできる場合には、ＢＣＵ１０８は、ＣＰＵ＿ＩＲ＿ＶＬＤ＿１Ｌ（命令バリッド）をＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌと同時にアクティブにし（Ｅ４）、その１／２クロックサイクル後にＩ＿ＤＡＴＡ＿ＢＵＳ及びＩＲＩＮを介してＣＰＵ１０２に命令データを出力する（Ｅ５）。

【００４３】

ＣＰＵ１０２が、内部メモリ１１０に対してハーバードアーキテクチャのバスを用いてア

クセスできる時には、このようにノーウェイトで命令をフェッチできる。そしてこの時にはＣＬＫ１は停止しない。

【００４４】

一方、ＢＣＵ１０８が、優先度の高いバス要求を処理している場合等には、例えばＤＭＡリクエストやＤＲＡＭリフレッシュを処理している場合には、命令フェッチ要求に対するアクノリッジ信号が遅延する。即ちＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌがアクティブになってから例えば２クロックサイクル後にＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌがアクティブになる（Ｅ６）。このようにＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌがアクティブになるのが遅延すると、クロック制御回路１０は、ＣＬＫ１を例えば１クロックサイクルだけ停止する（Ｅ７）。これにより、アクノリッジ信号が遅延した場合にもパイプラインを適正に制御することが可能となる。なおＣＰＵ１０２は、ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌがアクティブになるまで、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌのレベル及びＩＡの値を保持する。

【００４５】

図３（Ｃ）に、データのリード要求の際のタイミングチャートを示す。ＣＰＵ１０２は、ＣＰＵ＿ＤＡＴＡ＿ＲＥＱ＿１Ｌ（データリクエスト）をアクティブにしてデータのリード要求を行い、ＤＡ及びＤ＿ＡＤＤＲ＿ＢＵＳを介してアドレスを出力する。ＢＣＵ１０８は、ＣＰＵ＿ＤＡＴＡ＿ＡＣＫ＿１Ｌ（データアクノリッジ）、ＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌ（データバリッド）をアクティブにして、バスアクセスのアクノリッジ信号やデータのバリッド信号をＣＰＵ１０２に出力し、Ｄ＿ＤＡＴＡ＿ＢＵＳ及びＤＩＮを介してデータをＣＰＵ１０２に出力する。そしてアクノリッジ信号やバリッド信号が遅延した場合には、クロック制御回路１０がＣＬＫ１を停止し、これに対処する。

【００４６】

なおデータライトを行う場合には、ＣＰＵ１０２は、図３（Ｃ）のＤＡと同じタイミングでＤＯＵＴにデータを出力する。またデータライト時には、ＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌは使用されない。

【００４７】

ＢＣＵ１０８が、メモリアクセスに時間を要する外部メモリ１１４にアクセスする場合や、１６ビットのデータバスを用いバスサイクルを２回起動し３２ビットのデータをリードする場合（１６ビットの外部データバスを用いてマイクロコンピュータの周辺装置にアクセスする場合）等には、データのバリッド信号が遅延する。即ちＣＰＵ＿ＤＡＴＡ＿ＡＣＫ＿１Ｌがアクティブになってから（Ｅ８）、例えば１クロックサイクル後にＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌがアクティブになる（Ｅ９）。このようにＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌがアクティブになるのが遅延すると、クロック制御回路１０は、ＣＬＫ１を例えば１クロックサイクルだけ停止する（Ｅ１０）。これにより、データのバリッド信号が遅延した場合にもパイプラインを適正に制御できることになる。

【００４８】

さてストール時にパイプラインを適正に制御する手法の１つとして、各ステージの処理に必要な各種の制御信号（フェッチ信号、デコード信号、レジスタリード信号、メモリリード信号、演算開始信号、レジスタライト信号等）の状態をパイプラインレジスタに格納し、ストールが解消するまでこれらの制御信号の状態が変化しないように保持しておく手法が考えられる。

【００４９】

しかしながら、この手法によると、上記制御信号を格納するためにパイプラインレジスタが大規模化し、消費電力の増加を招く。また制御信号の状態が変化しないように、各ステージ毎に制御回路（ステートマシーン）を設ける必要があり、回路設計が複雑化する。また、ＣＰＵを動作させるクロック、特にＣＰＵ内のパイプラインレジスタや各ステージ毎に設けられた上記制御回路を動作させるクロックは、パイプラインのストール時にも停止していないため、電力が無駄に消費されてしまう。

【００５０】

本実施例によれば、ＣＰＵを動作させるクロックをストール時に停止することでストール

時のパイプライン制御の適正化を図っている。従って、上記手法で問題となった回路規模や消費電力の増加、回路設計の複雑化、電力の無駄な消費などの問題を解消できることになる。

【００５１】

３．クロック制御回路

図４にクロック制御回路１０の回路図の例を示す。命令アクノリッジウェイト回路１２（第１のステートマシーン）、命令バリッドウェイト回路１４（第２のステートマシーン）、データアクノリッジウェイト回路１６（第３のステートマシーン）、データバリッドウェイト回路１８（第４のステートマシーン）、インターロックウェイト回路２０及び外部リクエストウェイト回路２２の出力のいずれかがアクティブになると、ＣＰＵ＿ＳＴＯＰ＿０Ｌがアクティブになる。するとこのＣＰＵ＿ＳＴＯＰ＿０ＬによりＣＬＫがマスクされ、ＣＰＵ＿ＳＴＯＰ＿０Ｌがアクティブになる期間、ＣＬＫ１が停止する。一方、パイプラインのストール時にも動作させる回路ユニットに対しては、ＣＬＫをバッファリングした第２のクロックＣＬＫ２が供給される。

【００５２】

各回路１２～２２の出力は、以下に示す要因ＦＣ１～ＦＣ６が生じた場合にアクティブになる。

【００５３】

▲１▼ＦＣ１

ＣＰＵからＢＣＵへの命令フェッチ要求に対するアクノリッジ信号が遅延した場合（アクノリッジ信号がアクティブになるのを待つ場合）

▲２▼ＦＣ２

命令フェッチ要求に対するアクノリッジ信号をＢＣＵがＣＰＵに返したが命令データのバリッド信号が遅延した場合（バリッド信号がアクティブになるのを待つ場合）

▲３▼ＦＣ３

ＣＰＵからＢＣＵへのデータアクセス要求に対するアクノリッジ信号が遅延した場合（リード及びライト時）

▲４▼ＦＣ４

データアクセス要求に対するアクノリッジ信号をＢＣＵがＣＰＵに返したがデータのバリッド信号が遅延した場合（リード時のみ）

▲５▼ＦＣ５

データリード時にレジスタ・インタロックが生じた場合

▲６▼ＦＣ６

ＣＰＵ以外のバスアクセス要求にＢＣＵが応答するためにＣＰＵのメモリアクセスが待たされた場合（周辺装置が外部バスマスタになる場合等）

３．１　命令アクノリッジウェイト回路（第１のステートマシーン）

図５（Ａ）、（Ｂ）に、命令アクノリッジウェイト回路１２の状態遷移図及びこの回路の動作を説明するためのタイミングチャートを示す。

【００５４】

図５（Ａ）に示すように、命令アクノリッジウェイト回路１２は、”００”、”０１”、”１１”の３つのステート（ＳＴＡＴＥ＿ＷＡＩＴ＿ＩＲ＿ＡＣＫ）を持ち、これらの”００”、”０１”、”１１”は、各々、アクノリッジ信号の未返却数が０、１、２であることを示す。また命令アクノリッジウェイト回路１２の入力信号はＲＥＱ（＝ＰＲＩＭＥ＿ＩＲ＿ＲＥＱ）及びＡＣＫ（＝ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌ）であり、出力信号はＷＡＩＴ＿ＩＲ＿ＡＣＫ＿０Ｌである。なお”０Ｌ”の表記は、クロックの立ち下がりで変化する信号であることを意味する。

【００５５】

ここでＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌは、ＣＰＵがＢＣＵに命令のフェッチ要求をした時にアクティブになる信号であり、ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌは、ＣＰＵからの命令フェッチ要求をＢＣＵが受け付けた時にアクティブになる信号である。またＰＲＩＭＥ＿ＩＲ＿

ＲＥＱは、１つの命令の実行を終了した場合に次の命令をフェッチするための信号である。ＰＲＩＭＥ＿ＩＲ＿ＲＥＱは、ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌがアクティブになったか否かに関わらず、１つの命令の実行に対して１クロックサイクルのみアクティブになるのに対し、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌは図５（Ｂ）のＦ１、Ｆ２に示すように、対応するＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌがアクティブになるまでアクティブ状態を保持する。命令アクノリッジウェイト回路１２の状態を変化させる信号として、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌではなく ＰＲＩＭＥ＿ＩＲ＿ＲＥＱを使用することで、必要な数だけの命令リクエストをステートマシーンに入力できるため、回路を正常動作させることができる。

【００５６】

図５（Ａ）の状態遷移図について説明する。ＲＥＱが１（アクティブ）になると（Ｆ３）、ステート（ＳＴＡＴＥ＿ＷＡＩＴ＿ＩＲ＿ＡＣＫ）が”０１”に移行する（Ｆ４）。そして次にＡＣＫが１になると”００”に戻る（Ｆ５、Ｆ６）。

【００５７】

一方、ＲＥＱが１になり”０１”に移行した後（Ｆ７、Ｆ８）、ＲＥＱ及びＡＣＫが共に０であるとステートが”０１”に留まり、ウェイト状態になる（Ｆ９、Ｆ１０）。そして次にＲＥＱが０のままでＡＣＫが１になると”００”に戻る（Ｆ１１、Ｆ１２）。

【００５８】

ＲＥＱ３が出された後、ＡＣＫ３が返される前にＲＥＱ４が出された場合には（Ｆ１３、Ｆ１４）、ステートは、”０１”、”１１”、”０１”、”００”というように変化する（Ｆ１５）。ＲＥＱ３が出された後にＣＬＫ１はすぐに停止しないため（Ｆ１６）、次のＲＥＱ４が出されてしまう。従って、これらのＲＥＱ３、ＲＥＱ４に対応するＡＣＫ３、ＡＣＫ４が返されるのを待つ必要がある。本実施例によれば、アクノリッジ信号の未返却数が２であることを表すステート”１１”を用意することで、このような場合にも適正に対処できる。即ちＡＣＫ３及びＡＣＫ４が返された後に、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌがインアクティブになるように制御できる（Ｆ２）。

【００５９】

なおＲＥＱに対してＡＣＫがノーウェイトで返されない場合には、ＲＥＱ４が出された後にＣＬＫ１は停止する（Ｆ１８）。これによりＣＰＵの動作も停止するため、ＲＥＱ４の次のリクエストがＣＰＵから連続して出ることはない。従って、アクノリッジ信号の未返却数が３以上あることを表すステートを用意する必要はない。

【００６０】

命令アクノリッジウェイト回路１２は、ステートが”０１”又は”１１”で且つＡＣＫ（ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌ）がインアクティブの時にアクティブになる信号ＷＡＩＴ＿ＩＲ＿ＡＣＫ＿１Ｌを生成し、このＷＡＩＴ＿ＩＲ＿ＡＣＫ＿１Ｌを１／２クロックサイクル遅延させた信号であるＷＡＩＴ＿ＩＲ＿ＡＣＫ＿０Ｌを出力する。ＷＡＩＴ＿ＩＲ＿ＡＣＫ＿０Ｌがアクティブになると、そのアクティブ期間にＣＬＫがマスクされるため（図４参照）、ＣＬＫ１が停止することになる（Ｆ１７～Ｆ２０）。

【００６１】

３．２　命令バリッドウェイト回路（第２のステートマシーン）

図６（Ａ）、（Ｂ）に、命令バリッドウェイト回路１４の状態遷移図及びこの回路の動作を説明するためのタイミングチャートを示す。

【００６２】

図６（Ａ）に示すように命令バリッドウェイト回路１４は、前述の命令アクノリッジウェイト回路１２と同様に”００”、”０１”、”１１”の３つのステート（ＳＴＡＴＥ＿ＷＡＩＴ＿ＩＲ＿ＶＬＤ）を持つ。これらのステートは、各々、バリッド信号の未返却数が０、１、２であることを表すものである。また命令バリッドウェイト回路１４の入力信号はＡＣＫ（＝ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌ）及びＶＬＤ（＝ＣＰＵ＿ＩＲ＿ＶＬＤ＿１Ｌ）であり、出力信号はＷＡＩＴ＿ＩＲ＿ＶＬＤ＿０Ｌである。ここでＣＰＵ＿ＩＲ＿ＶＬＤ＿１Ｌは、ＢＣＵがＣＰＵに対して命令データを出力する時にアクティブになる信号である。

【００６３】
図６（Ａ）の状態遷移図について説明する。ＡＣＫが１（アクティブ）になった時に、ＶＬＤもノーウェイトで１になるとステートは”００”に留まる（Ｇ１、Ｇ２、Ｇ３）。
【００６４】
一方、ＡＣＫは１になったがＶＬＤが０（インアクティブ）のままであるとステートは”０１”に移行する（Ｇ４、Ｇ５、Ｇ６）。即ちＶＬＤがアクティブになるのを待つ状態になる。その後、ＡＣＫ及びＶＬＤが共に０のままである場合にはステートは”０１”に留まり（Ｇ７、Ｇ８、Ｇ９）、ＶＬＤが１になると、ステートは”００”に戻る（Ｇ１０、Ｇ１１、Ｇ１２）。
【００６５】
ＡＣＫ３を受け付けた後、ＶＬＤ３を受け付ける前にＡＣＫ４を受け付けた場合には（Ｇ１３、Ｇ１４、Ｇ１５）、ステートは、”０１”、”１１”、”０１”、”００”というように変化する（Ｇ１６）。ＲＥＱ３が出された後にＣＬＫ１はすぐに停止しないため、次のＲＥＱ４が出されてしまい、これらのＲＥＱ３、ＲＥＱ４に対応するＶＬＤ３、ＶＬＤ４がアクティブになるのを待つ必要がある。本実施例によれば、バリッド信号の未返却数が２であることを表すステート”１１”を用意することで、このような場合にも適正に対処できる。
【００６６】
命令バリッドウェイト回路１４は、ステート（ＳＴＡＴＥ＿ＷＡＩＴ＿ＩＲ＿ＶＬＤ）が”０１”又は”１１”で且つＣＰＵ＿ＩＲ＿ＶＬＤ＿０Ｌ（ＣＰＵ＿ＩＲ＿ＶＬＤ＿１Ｌを１／２クロックサイクル遅延させた信号）がインアクティブの時にアクティブになる信号ＷＡＩＴ＿ＩＲ＿ＶＬＤ＿０Ｌを出力する。ＷＡＩＴ＿ＩＲ＿ＶＬＤ＿０Ｌがアクティブになると、そのアクティブ期間にＣＬＫがマスクされるため（図４参照）、ＣＬＫ１が停止することになる（Ｇ１７～Ｇ２２）。
【００６７】
３．３　データアクノリッジウエイト回路（第３のステートマシン）、データバリッドウェイト回路（第４のステートマシーン）
データアクノリッジウェイト回路１６の状態遷移図及びタイミングチャートは、
図５（Ａ）、（Ｂ）とほぼ同様であり、データバリッドウェイト回路１８の状態遷移図及びタイミングチャートは図６（Ａ）、（Ｂ）とほぼ同様にあるため、詳しい説明を省略する。
【００６８】
なおデータアクノリッジウェイト回路１６の入力信号はＲＥＱ（＝ＰＲＩＭＥ＿ＤＡＴＡ＿ＲＥＱ）及びＡＣＫ（＝ＣＰＵ＿ＤＡＴＡ＿ＡＣＫ＿１Ｌ）であり、出力信号はＷＡＩＴ＿ＤＡＴＡ＿ＡＣＫ＿０Ｌである。またデータバリッドウェイト回路１８の入力信号はＡＣＫ（＝ＣＰＵ＿ＤＡＴＡ＿ＡＣＫ＿１Ｌ）及びＶＬＤ（＝ＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌ）であり、出力信号はＷＡＩＴ＿ＤＡＴＡ＿ＶＬＤ＿０Ｌである。ここでＣＰＵ＿ＤＡＴＡ＿ＲＥＱ＿１Ｌは、ＣＰＵがＢＣＵにデータのアクセス要求をした時にアクティブになる信号であり、ＣＰＵ＿ＤＡＴＡ＿ＡＣＫ＿１Ｌは、ＣＰＵからのデータアクセス要求をＢＣＵが受け付けた時にアクティブになる信号である。またＰＲＩＭＥ＿ＤＡＴＡ＿ＲＥＱは、ＣＰＵがメモリアクセスをする命令を実行した時にＲＡフェーズでアクティブになる信号である。またＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌは、ＢＣＵがＣＰＵに対しデータを出力する時にアクティブになる信号である。
【００６９】
３．４　インターロックウェイト回路
まずレジスタ・インタロックについて詳細に説明する。レジスタ・インタロックは、第１の命令でメモリからレジスタにデータをロードし、その直後の第２の命令でそのレジスタのデータを使用する場合に発生する。例えば第１、第２の命令が下記に示すものである場合を考える。
【００７０】

ＬＤ　　％Ｒ１，［％Ｒ３］・・・（第１の命令）
ＡＤＤ　％Ｒ５，％Ｒ１　　・・・（第２の命令）
上記の命令の実行前にレジスタＲ１、Ｒ３、Ｒ５及びアドレス［Ｋ］で指定されるメモリ領域に格納されているデータ、並びに上記の命令実行後にこれらのレジスタ及びメモリ領域に格納されるべきデータを、図７（Ａ）に示す。図中の”－”はドントケアである事を表し、ここではレジスタＲ１にデータＪが格納されていたとする。

【００７１】

図７（Ｂ）に、クロック停止及びバスバイパスを行わない場合のタイミングチャートを示す。まず第１の命令のＲＡフェーズにより、汎用レジスタ１３０の読み出しポート１１８（図１参照）からデータＫが読み出される（Ｈ１）。このデータＫは、ＤＡを介してＤ＿ＡＤＤＲ＿ＢＵＳに出力され（Ｈ２）、これによりメモリからデータＭが読み出され、Ｄ＿ＤＡＴＡ＿ＢＵＳ、ＤＩＮを介してＣＰＵにデータＭが入力される（Ｈ３）。

【００７２】

一方、第２の命令のＲＡフェーズにより、読み出しポート１１８、１１９からデータＪ、Ｌが読み出され、ＰＡ＿ＢＵＳ、ＰＢ＿ＢＵＳに出力される（Ｈ４、Ｈ５）。次に第２の命令のＥＷフェーズにより、ＰＡ＿ＢＵＳ、ＰＢ＿ＢＵＳ上のデータＪ、Ｌが加算され（Ｈ６）、レジスタＲ５に加算結果データＪ＋Ｌが書き込まれる（Ｈ７）。

【００７３】

このように、クロック停止及びバスバイパスを行わない図７（Ｂ）の例では、レジスタＲ５に誤ったデータＪ＋Ｌ（正しいデータはＭ＋Ｌ）が書き込まれてしまう。加算処理のオペランドが、メモリから読み出したデータＭではなくＪになってしまうからである。

【００７４】

そこで本実施例では、図８に示すように、レジスタ・インタロックが発生した場合にクロックＣＬＫ１を停止すると共にバスのバイパスを行うことで、この問題を解消している。

【００７５】

まず第１、第２の命令のＦＤフェーズ（Ｉ１、Ｉ２）で得られたデコード情報に基づいて、レジスタ・インタロックが発生するか否かを命令デコーダ１２０が判断する。即ち第１の命令でメモリからレジスタにデータをロードし、第２の命令でそのレジスタのデータを使用する場合にレジスタ・インタロックが発生すると判断する。レジスタ・インタロックが発生すると判断すると、命令デコーダ１２０はＤＩＮ＿ＩＮＴ＿１Ｌをアクティブにする（Ｉ３）。するとクロック制御回路１０の中のインタロックウェイト回路２０（図４）がこのＤＩＮ＿ＩＮＴ＿１Ｌを受け、これを１／２クロックサイクル遅延させた信号ＤＩＮ＿ＩＮＴ＿０Ｌを出力する。このＤＩＮ＿ＩＮＴ＿０ＬによりＣＬＫがマスクされ、ＣＬＫ１が停止する（Ｉ４）。これにより第１の命令のＥＷフェーズ、第２の命令のＲＡフェーズの期間が延長され（Ｉ５、Ｉ６）、第２の命令のＥＷフェーズが１クロックサイクルだけ遅延する。即ち加算処理の実行を１クロックサイクルだけ遅延できる。

【００７６】

一方、この時、図１のバスバイパス１４２が、ＷＷ＿ＢＵＳとＰＡ＿ＢＵＳの間をバイパスする。これにより、Ｄ＿ＤＡＴＡ＿ＢＵＳ、ＤＩＮを介してＷＷ＿ＢＵＳに入力されたデータＭが（Ｉ８）、ＰＡ＿ＢＵＳに出力される（Ｉ９）。この結果、ＰＡ＿ＢＵＳ、ＰＢ＿ＢＵＳ上のデータＭ、Ｌが、第２の命令のＥＷフェーズで加算され（Ｉ１０）、レジスタＲ５に正しいデータＭ＋Ｌが書き込まれることになる（Ｉ１１）。

【００７７】

以上のように本実施例によれば、レジスタ・インタロックが生じた場合にも、パイプラインを適正に制御でき、正しい結果を得ることができる。

【００７８】

３．５　外部リクエストウェイト回路

ＢＣＵ１０８は、バスのアクセス要求に対してバスの使用権の調停を行っている。そして例えばＤＭＡリクエスト、外部バスマスタからのリクエスト、ＤＲＡＭリフレッシュのリクエスト等は、ＣＰＵ１０２のバスアクセスリクエストよりも優先順位が高い。従って、

このようにＢＣＵ１０８が優先順位の高いリクエストに応答している場合には、図４の外部リクエストウェイト回路２２がＥＸ＿ＲＥＱ＿０Ｌをアクティブにする。これによりＥＸ＿ＲＥＱ＿０Ｌがアクティブとなる期間、ＣＬＫ１が停止することになり、ＣＰＵ１０２の動作が停止する。

【００７９】

４．停止しないクロックＣＬＫ２

上述したように、例えばパイプラインレジスタなど、ＣＰＵ１０２の大部分の回路ユニットは、パイプラインのストール時に停止するクロックＣＬＫ１により動作する。しかしながら、回路ユニットによっては、パイプラインのストール時に停止しないクロックＣＬＫ２により動作させることが望ましいものもある。

【００８０】

本実施例では第１に、クロック制御回路１０については、ＣＬＫ１ではなくＣＬＫ２により動作させている。クロック制御回路１０は、ＣＬＫ１を停止させるか否かを判断する回路だからである。

【００８１】

また本実施例では第２に、ＤＩＮを介してＣＰＵ１０２に入力されるデータを汎用レジスタ１３０に書き込むための制御等を行うレジスタ制御回路１３１（図１参照）を、ＣＬＫ２により動作させている。図９に、レジスタ制御回路１３１の構成例（書き込み回路の部分のみ）を示し、図１０、図１１にそのタイミングチャートを示す。

【００８２】

図９において、ＲＥＧ＿ＷＲ＿ＳＥＬはレジスタＲ０～Ｒ１５のいずれかを選択するための４ビットの信号であり、この信号はレジスタセレクトデコーダ４０によりデコードされる。

【００８３】

ＤＩＮ２ＷＷ＿１Ｌは、命令をデコードした結果、メモリから読み出したデータを汎用レジスタ１３０に書き込む命令であると判断された場合に、アクティブになる信号である。ＲＥＧ＿ＷＲ＿１Ｌは、命令をデコードした結果、ＡＬＵ１３６などのメモリ以外の回路ユニットからのデータを汎用レジスタ１３０に書き込む命令であると判断された場合に、その命令のＥＷフェーズの期間にアクティブになる信号である。

【００８４】

汎用レジスタ１３０のラッチ信号であるＲＥＧ＿ＬＡＴＣＨ＿ＳＩＧは、ＲＥＧ＿ＷＲ＿１Ｌ、ＤＩＮ２ＷＷ＿１Ｌ、ＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌ及びストール時に停止しないクロックＣＬＫ２に基づき論理回路４２が生成する。またＤＩＮからのデータをＷＷ＿ＢＵＳに伝える３ステートバッファ４２のイネーブル信号ＤＩＮ＿ＥＮＢは、ＤＩＮ２ＷＷ＿１Ｌ、ＣＬＫ２に基づきＡＮＤ回路４４が生成する。

【００８５】

図１０は、ノーウェイトでメモリアクセスする場合のタイミングチャートである。まず第１の命令のＷステージで、ＲＥＧ＿ＬＡＴＣＨ＿ＳＩＧがアクティブになりデータ０ｘ７４がレジスタＲ１に書き込まれる（Ｊ１）。この際、ＲＥＧ＿ＷＲ＿ＳＥＬに基づき、データを書き込むレジスタが決められる（Ｊ２）。次に、レジスタＲ１に格納されるデータ０ｘ７４が、メモリの読み出しアドレスとしてＤＡに出力され（Ｊ３）、これによりメモリからＤＩＮを介してデータ０ｘ１２３４が読み出される（Ｊ４）。そしてこのデータ０ｘ１２３４は第２の命令のＷステージでレジスタＲ２に書き込まれる（Ｊ５）。次に、データ０ｘ０４と、レジスタＲ１に格納されるデータ０ｘ７４とが第３の命令により加算され、加算結果がレジスタＲ１に書き込まれる（Ｊ６）。

【００８６】

一方、図１１は、メモリアクセスにウェイトがある場合のタイミングチャートである。図１１では、アクノリッジ信号ＣＰＵ＿ＤＡＴＡ＿ＡＣＫ＿１Ｌが１クロックサイクル遅延し（Ｋ１）、バリッド信号ＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌが２クロックサイクル遅延している（Ｋ２）。まず第１の命令のＷステージでデータ０ｘ７４がレジスタＲ１に書き込

まれる（Ｋ３）。次にアクノリッジ信号の遅延によりＷＡＩＴ＿ＤＡＴＡ＿ＡＣＫ＿０Ｌがアクティブになり（Ｋ４）、バリッド信号の遅延によりＷＡＩＴ＿ＤＡＴＡ＿ＶＬＤ＿０Ｌがアクティブになる（Ｋ５）。これによりＣＰＵ＿ＳＴＯＰ＿０Ｌがアクティブになり（Ｋ６）、ＣＬＫ１が停止する（Ｋ７）。

【００８７】

さて、第２の命令によりメモリから読み出されたデータ０ｘ１２３４をレジスタＲ２に書き込むためには、図１１のＫ８においてＲＥＧ＿ＬＡＴＣＨ＿ＳＩＧをアクティブにすることが望ましい。ＤＩＮ上のデータ０ｘ１２３４が有効になるのは、ＣＰＵ＿ＤＡＴＡ＿ＶＬＤ＿１Ｌがアクティブになってから１／２クロックサイクル後を起点として１クロックサイクル期間だけだからである（Ｋ９）。

【００８８】

しかしながら、例えば図９に示すラッチ信号ＲＥＧ＿ＬＡＴＣＨ＿ＳＩＧやイネーブル信号ＤＩＮ＿ＥＮＢを、ストール時に停止するクロックＣＬＫ１に基づき生成すると、データ０ｘ１２３４をレジスタＲ２に適正に書き込めないという問題が発生する。

【００８９】

そこで本実施例では、これらのＲＥＧ＿ＬＡＴＣＨ＿ＳＩＧやＤＩＮ＿ＥＮＢを、ストール時に停止しないクロックＣＬＫ２に基づき生成することで、上記問題を解決している。

【００９０】

５．リクエスト信号の生成

本実施例のインターフェース回路３０は、クロック制御回路１０に出力され１つのバスアクセス要求について１クロックサイクル期間だけアクティブになる第１のリクエスト信号（ＰＲＩＭＥ＿ＩＲ＿ＲＥＱ、ＰＲＩＭＥ＿ＤＡＴＡ＿ＲＥＱ）と、ＢＣＵ１０８に出力され第１のリクエスト信号がアクティブになってからＢＣＵ１０８からのアクノリッジ信号がアクティブになるまでの期間アクティブになる第２のリクエスト信号（ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌ、ＣＰＵ＿ＤＡＴＡ＿ＲＥＱ＿１Ｌ）とを生成している。このように２種類のアクノリッジ信号を用意することで、パイプラインにストールが生じた場合にも、ＢＣＵ１０８との間のデータ転送を適正に制御できると共にクロック制御回路１０が含むステートマシーンを適正に動作させることが可能になる。

【００９１】

図１２に、ＰＲＩＭＥ＿ＩＲ＿ＲＥＱ（第１のリクエスト信号）、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌ（第２のリクエスト信号）を生成する回路の例を示し、図１３にその動作を説明するためのタイミングチャートを示す。この回路は、図１のインターフェース回路３０に含まれるものである。

【００９２】

ＩＲ＿ＲＥＱ＿１Ｌは、１つの命令の実行を終了した時に次の命令をフェッチするための信号である。ＩＲ＿ＲＥＱ＿１Ｌは命令の最後のＲＡフェーズでアクティブになる（Ｌ１、Ｌ２）。このＩＲ＿ＲＥＱ＿１ＬとＣＰＵ＿ＳＴＯＰ＿１Ｌ（ＣＰＵ＿ＳＴＯＰ＿０Ｌを１／２クロックサイクル遅延させた信号）の反転信号の論理積をとることで、ＰＲＩＭＥ＿ＩＲ＿ＲＥＱが得られる。このＰＲＩＭＥ＿ＩＲ＿ＲＥＱは、クロック制御回路１０に含まれる命令アクノリッジウェイト回路１２（第１のステートマシーン）に出力され、命令アクノリッジウェイト回路１２は、このＰＲＩＭＥ＿ＩＲ＿ＲＥＱに基づいてステートを変化させる。この時、ＰＲＩＭＥ＿ＩＲ＿ＲＥＱは、１つのバスアクセス要求（１つの命令の実行）について１クロックサイクル期間だけアクティブになる（Ｌ３～Ｌ６）。このようなＰＲＩＭＥ＿ＩＲ＿ＲＥＱを使用することで、命令アクノリッジウェイト回路１２が１つのリクエストを２つ以上のリクエストと誤認することを防止できる。即ちＰＲＩＭＥ＿ＩＲ＿ＲＥＱの代わりにＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌを使用すると、命令アクノリッジウェイト回路１２に入力されるリクエスト信号が例えば図１３のＬ７の期間においてもアクティブであると判断されてしまい、命令アクノリッジウェイト回路１２が誤動作するが、ＰＲＩＭＥ＿ＩＲ＿ＲＥＱを使用することでこの問題を解消できる。

【００９３】

論理回路４６は、ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌ、ＳＴＡＴＥ＿ＷＡＩＴ＿ＩＲ＿ＡＣＫ（図５参照）に基づいて、ＥＸＴＥＮＤ＿ＩＲ＿ＲＥＱ＿１Ｌを生成する。このＥＸＴＥＮＤ＿ＩＲ＿ＲＥＱ＿１Ｌは、▲１▼ＳＴＡＴＥ＿ＷＡＩＴ＿ＩＲ＿ＡＣＫが”０１”又は”１１”且つＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌが０の場合或いは▲２▼ＳＴＡＴＥ＿ＷＡＩＴ＿ＩＲ＿ＡＣＫが”１１”の場合に、アクティブになる。このＥＸＴＥＮＤ＿ＩＲ＿ＲＥＱ＿１ＬとＰＲＩＭＥ＿ＩＲ＿ＲＥＱの論理和をとることで、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌが得られる。即ちＰＲＩＭＥ＿ＩＲ＿ＲＥＱのアクティブ期間を、ＥＸＴＥＮＤ＿ＩＲ＿ＲＥＱ＿１Ｌにより延長したものがＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌになる。そして、このＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌは、対応するアクノリッジ信号がアクティブになるまでの期間、アクティブになる（Ｌ８～Ｌ１１）。これにより、パイプラインがストールした場合にも、ＢＣＵ１０８との間の適正なデータ転送制御が可能となる。即ちＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌの代わりにＰＲＩＭＥ＿ＩＲ＿ＲＥＱを使用すると、ＢＣＵ１０８からのアクノリッジ信号が返される前（例えばＬ１２）にＢＣＵ１０８に出力されるリクエスト信号がインアクティブになってしまうが、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌを使用することでこの問題を解消できる。

【００９４】

（実施例２）

実施例２は、積和演算回路１４０（所与の回路ユニット）を動作させる第３のクロックＣＬＫ３を、パイプラインステージがストールした場合又は積和演算回路１４０を使用しない命令を処理する場合に停止する実施例である。

【００９５】

図１４（Ａ）に実施例２のマイクロコンピュータに使用されるクロック制御回路の構成例を示す。図４との相違点は、第３のクロックＣＬＫ３を生成するための論理回路５０を新たに設けた点である。ここでＭＡＣ＿ＩＮＳＴは、積和演算命令を処理するときにアクティブになる信号であり、命令デコーダ１２０から入力される。このＭＡＣ＿ＩＮＳＴは、ラッチ５２によりラッチされ、インバータ回路５４により反転される。そして、この反転された信号とＣＰＵ＿ＳＴＯＰ＿０ＬがＯＲ回路５６に入力され、ＭＡＣ＿ＳＴＯＰ＿０Ｌが生成される。このＭＡＣ＿ＳＴＯＰ＿０ＬがアクティブになるとＣＬＫ３は停止する。

【００９６】

図１４（Ｂ）のタイミングチャートに示すように、ＣＬＫ３は、ＣＰＵ＿ＳＴＯＰ＿０Ｌがアクティブになった場合、即ちパイプラインがストールした場合に停止する（Ｍ１、Ｍ２）。また積和演算命令以外の命令が処理されている場合にも停止する（Ｍ３～Ｍ６）。

【００９７】

積和演算回路１４０はＣＰＵ１０２の制御の下で動作しており、例えば積和演算回路１４０の入力データはＣＰＵ１０２の制御の下で供給される。従って、積和演算回路１４０は、ＣＰＵ１０２の動作と同期をとる必要がある。しかしながら、パイプラインがストールした場合には、実施例１で説明したようにＣＰＵ１０２の大部分を動作させるクロックＣＬＫ１も停止する。これによりＣＰＵ１０２の動作との同期がとれなくなってしまう。

【００９８】

そこで本実施例では、パイプラインがストールした場合にＣＬＫ１のみならずＣＬＫ３も停止する。これにより、ＣＰＵ１０２の動作と同期をとることが可能となり、パイプラインの適正な制御が可能となる。

【００９９】

また積和演算命令以外の命令が処理されている場合に、積和演算回路１４０を動作させると無駄な電力を消費することになる。特に積和演算回路１４０は高速動作すると共に回路規模が大きいため、積和演算回路１４０の無駄な動作に起因する電力の消費は非常に大きなものとなる。

【０１００】

そこで本実施例では、積和演算命令以外の命令が処理されている場合に、積和演算回路１

４０を動作させるＣＬＫ３を停止する。これにより、消費電力を格段に低減できる。
【０１０１】
なおＣＬＫ３により動作する回路ユニットとしては、積和演算回路のみならず、乗算器、加算器、除算器、或いはＡＬＵに含まれるバレルシフタ等、種々のものを考えることができる。
【０１０２】
（実施例３）
実施例３は、ＣＰＵからのバスアクセス要求に対するＢＣＵの応答のウェイトサイクル数を制御することで、ＣＰＵのクロック周波数を変更することなく、ＣＰＵの命令実行速度及び消費電力を制御する実施例である。
【０１０３】
図１５に実施例３のマイクロコンピュータのブロック図の例を示す。ここでＢＣＵ１０８が含むバスインターフェース回路６０は、バスのインターフェースを行う回路であり、各種のバス（Ｉ＿ＡＤＤＲ＿ＢＵＳ等）や各種のインターフェース信号（ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌ等）が接続される。そしてバスインターフェース回路６０は、バスアクセス要求に対するＢＣＵの応答のウェイトサイクル数を制御するウェイト制御回路６８を含む。
【０１０４】
第１のウェイト制御レジスタ６２は、メモリマップの各エリア毎にウェイトサイクル数を設定するためのレジスタである。例えば内部メモリ１１０はメモリアクセスに時間をあまり要しないため、内部メモリ１１０が割り当てられるエリアには小さなウェイトサイクル数が設定される。一方、外部メモリ１１４はメモリアクセスに時間を要するため、外部メモリ１１４が割り当てられるエリアには大きなウェイトサイクル数が設定される。
【０１０５】
第２のウェイト制御レジスタ６４は、ウェイトサイクル数の制御により消費電力を制御するためのレジスタであり、本実施例では２ビットのレジスタになっている。第２のウェイト制御レジスタ６４に”００”、”０１”、”１０”、”１１”が設定されると、図１６（Ａ）、（Ｂ）、（Ｃ）に示す信号ＲＥＱ＿ＷＩＮ（周期信号）のデューティが、各々、１／１デューティ、１／２ディーティ、１／４デューテイ、１／８デューティに設定される。この信号ＲＥＱ＿ＷＩＮの生成は、ウェイト制御回路６８が行う。
【０１０６】
ＲＥＱ＿ＷＩＮが１／１デューティに設定されている場合には、図１６（Ａ）に示すように、ＲＥＱ＿ＷＩＮは常にアクティブになり、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌに対してノーウェイトでＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌが返されることになる。
【０１０７】
ＲＥＱ＿ＷＩＮが１／２デューティに設定されている場合には、図１６（Ｂ）に示すように、ＲＥＱ＿ＷＩＮは２クロックサイクルに１回アクティブになる。そしてアクノリッジ信号ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌは、ＣＰＵ１０２からのリクエスト信号ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌがアクティブであり且つ周期信号ＲＥＱ＿ＷＩＮがアクティブとなった場合に、アクティブになる（Ｎ１、Ｎ２、Ｎ３）。これによりＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌのウェイトサイクル数が、ほぼ１に設定されることになる。
【０１０８】
ＲＥＱ＿ＷＩＮが１／４デューティに設定されている場合には、図１６（Ｃ）に示すように、ＲＥＱ＿ＷＩＮは４クロックサイクルに１回アクティブになる。そしてアクノリッジ信号ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌは、ＣＰＵ＿ＩＲ＿ＲＥＱ＿１Ｌがアクティブであり且つＲＥＱ＿ＷＩＮがアクティブとなった場合に、アクティブになる（Ｎ４、Ｎ５、Ｎ６）。これによりＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌのウェイトサイクル数が、ほぼ３に設定されることになる。
【０１０９】
このように本実施例では、第２のウェイト制御レジスタ６４の設定により、アクノリッジ信号ＣＰＵ＿ＩＲ＿ＡＣＫ＿１Ｌのウェイトサイクル数を制御できる。そしてアクノリッ

ジ信号にウェイトサイクルが挿入されると、実施例１で説明したように、ＣＰＵ１０２の大部分を動作させるクロックであるＣＬＫ１がウェイトサイクルの期間停止する。従って、ウェイトサイクル数を大きくすると、ＣＬＫ１の停止期間も長くなる。これにより、クロック周波数を低くすることなく、ＣＰＵ１０２の実行速度を低くし、省電力化を図ることができる。

【０１１０】

ＣＰＵの消費電力を制御する１つの手法として、ＣＰＵのクロック周波数を制御する手法がある。しかしながら、この手法を採用すると、クロックを分周する回路が必要となり回路規模が増加する。また高速で動作する分周回路の付加は消費電力の増加を招く。更に分周比の切り替えタイミングの制御が難しいという問題もある。

【０１１１】

本実施例によれば、ウェイトサイクル数を制御することで、ＣＬＫ１の停止期間が制御され、ＣＰＵの実行速度及び消費電力が自動的に制御されるため、分周回路の付加に伴い生じる上記問題点を解消できる。特に本実施例は、パイプラインのストール時にＣＬＫ１を停止するために設けられたハードウェア資源を利用して、ＣＰＵの実行速度及び消費電力の制御を実現している点に特徴がある。

【０１１２】

なおウェイトサイクル数を制御する手法としては、図１６（Ａ）～（Ｃ）で説明したＲＥＱ＿ＷＩＮを利用する手法が、回路及び制御の簡易性の観点から特に望ましいが、これに限られるものではない。

【０１１３】

（実施例４）

実施例４は、本発明のマイクロコンピュータの詳細例を示す実施例である。

【０１１４】

図１７に示すように実施例４のマイクロコンピュータ７００は、３２ビットのマイクロコンピュータであり、ＣＰＵ７１０、ＲＯＭ７２０、ＲＡＭ７３０、高周波発振回路９１０、低周波発振回路９２０、リセット回路９３０、プリスケーラ９４０、１６ビットプログラマブルタイマ９５０や８ビットプログラマブルタイマ９６０やクロックタイマ９７０などのタイマ回路、インテリジェントＤＭＡ９８０や高速ＤＭＡ９９０などのデータ転送制御回路、割り込みコントローラ８００、シリアルインターフェース８１０、ＢＣＵ７４０、Ａ／Ｄ変換器８３０やＤ／Ａ変換器８４０などのアナログインターフェース回路、入力ポート８５０や出力ポート８６０やＩ／Ｏポート８７０などのＩ／Ｏ回路、及びそれらを接続する各種バス７５０、７６０、各種端子８９０を含む。

【０１１５】

１チップの半導体基板上に形成されるこのマイクロコンピュータ７００は、３２ビットのデータを処理できるＲＩＳＣ方式のマイクロコンピュータである。そしてパイプライン方式及びロード・ストア方式のアーキテクチャーを採用し、ほとんど全ての命令を１クロックの期間で実行する。全ての命令は１６ビットの固定長で記述されており、これにより極めて小さい命令コードサイズを実現している。

【０１１６】

（実施例５）

実施例５は、実施例１～４で説明したマイクロコンピュータを含む電子機器に関する実施例である。実施例１～４のマイクロコンピュータによれば、適正なパイプライン制御を行いながら、マイクロコンピュータの小規模化、省電力化を実現できる。これによりこのマイクロコンピュータが組み込まれる電子機器の低コスト化、コンパクト化、低消費電力化を図ることが可能となる。

【０１１７】

例えば図１８（Ａ）に電子機器の１つであるカーナビゲーションシステムの内部ブロック図を示し、図１９（Ａ）にその外観図を示す。カーナビゲーションシステムの操作はリモコン５１０を用いて行われ、ＧＰＳやジャイロからの情報に基づいて位置検出部５２０が

車の位置を検出する。地図などの情報はＣＤＲＯＭ５３０（情報記憶媒体）に格納されている。画像メモリ５４０は画像処理の際の作業領域になるメモリであり、生成された画像は画像出力部５５０を用いてドライバーに表示される。マイクロコンピュータ５００は、リモコン５１０、位置検出部５２０、ＣＤＲＯＭ５３０などのデータ入力源からデータを入力し、種々の処理を行い、処理後のデータを画像出力部５５０などの出力装置を用いて出力する。

【０１１８】

図１８（Ｂ）に電子機器の１つであるゲーム装置の内部ブロック図を示し、図１９（Ｂ）にその外観図を示す。このゲーム装置では、ゲームコントローラ５６０からのプレーヤの操作情報、ＣＤＲＯＭ５７０からのゲームプログラム、ＩＣカード５８０からのプレーヤ情報等に基づいて、画像メモリ５９０を作業領域としてゲーム画像やゲーム音を生成し、画像出力部６１０、音出力部６００を用いて出力する。

【０１１９】

図１８（Ｃ）に電子機器の１つであるプリンタの内部ブロック図を示し、図１９（Ｃ）にその外観図を示す。このプリンタでは、操作パネル６２０からの操作情報、コードメモリ６３０及びフォントメモリ６４０から文字情報に基づいて、ビットマップメモリ６５０を作業領域として、印刷画像を生成し、プリント出力部６６０を用いて出力する。またプリンタの状態やモードを表示パネル６７０を用いてユーザに伝える。

【０１２０】

なお本発明のマイクロコンピュータを適用できる電子機器としては、上記以外にも例えば、携帯電話（セルラーフォン）、ＰＨＳ、ページャ、オーディオ機器、電子手帳、電子卓上計算機、ＰＯＳ端末、タッチパネルを備えた装置、プロジェクタ、ワードプロセッサ、パーソナルコンピュータ、テレビ、ビューファインダ型又はモニタ直視型のビデオテープレコーダなど種々のものを考えることができる。

【０１２１】

なお、本発明は上記実施例１～５に限定されるものではなく、本発明の要旨の範囲内で種々の変形実施が可能である。

【０１２２】

例えばマイクロコンピュータの構成、バスや命令やデータのビット数、使用する汎用レジスタの数、パイプラインステージの構成等は、上記実施例で説明したものに限らず種々の変形実施が可能である。

【０１２３】

またパイプラインのストール要因も上記実施例で説明したものに限られるものではない。またクロック制御回路、インターフェース回路、バス制御回路の構成も上記実施例で説明したものに限られるものではない。

【０１２４】

また上記実施例３は実施例１との組み合わせにおいて特に効果を奏するが、これに限られるものではない。

【０１２５】

【図面の簡単な説明】

【図１】マイクロコンピュータの構成例を示す図である。

【図２】図２（Ａ）、（Ｂ）は、パイプラインのストール及びストール時のクロック停止について説明するための図である。

【図３】図３（Ａ）、（Ｂ）、（Ｃ）は、バスアクセス要求が待たされた場合のパイプライン制御について説明するための図である。

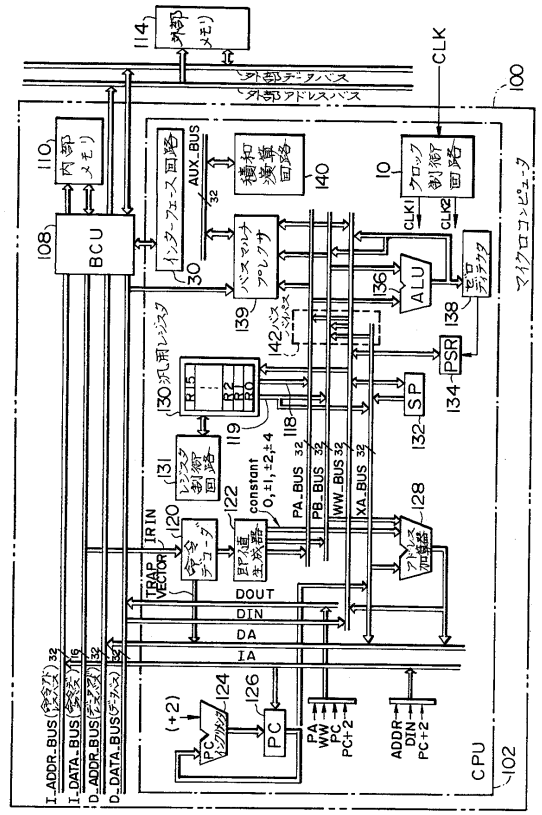
【図４】クロック制御回路の構成例を示す図である。

【図５】図５（Ａ）、（Ｂ）は、命令アクノリッジウェイト回路の状態遷移図及びその動作を説明するためのタイミングチャートである。

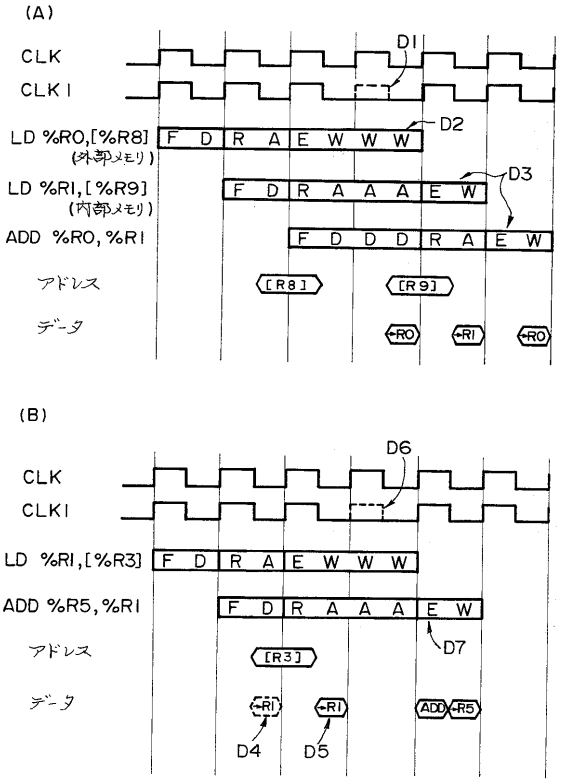
【図６】図６（Ａ）、（Ｂ）は、命令バリッドウェイト回路の状態遷移図及びその動作を説明するためのタイミングチャートである。

【図７】図７（Ａ）、（Ｂ）は、レジスタ・インターロック時のパイプライン制御について説明するための図である。
【図８】レジスタ・インターロック時のパイプライン制御について説明するための図である。
【図９】レジスタ制御回路の構成例を示す図である。
【図１０】パイプラインのストール時に停止しないクロックについて説明するための図である。
【図１１】パイプラインのストール時に停止しないクロックについて説明するための図である。
【図１２】リクエスト信号を生成回路の構成例を示す図である。
【図１３】２種類のリクエスト信号の生成について説明するための図である。
【図１４】実施例２に係るクロック制御回路の構成例を示す図である。
【図１５】実施例３に係るマイクロコンピュータの構成例を示す図である。
【図１６】実施例３の動作について説明するための図である。
【図１７】実施例４に係るマイクロコンピュータの構成例である。
【図１８】図１８（Ａ）、（Ｂ）、（Ｃ）は、種々の電子機器の内部ブロック図の例である。
【図１９】図１９（Ａ）、（Ｂ）、（Ｃ）は、種々の電子機器の外観図の例である。
【符号の説明】
１０　クロック制御回路
３０　インターフェース回路
６０　バスインターフェース回路
６２　第１のウェイト制御レジスタ
６４　第２のウェイト制御レジスタ
６８　ウェイト制御回路
１００　マイクロコンピュータ
１０２　ＣＰＵ
１０８　ＢＣＵ
１１０　内部メモリ
１１４　外部メモリ
１２０　命令デコーダ
１２２　即値生成器
１２４　ＰＣインクリメント
１２６　ＰＣ
１３０　汎用レジスタ
１３１　レジスタ制御回路
１３２　ＳＰ
１３４　ＰＳＲ
１３６　ＡＬＵ
１３８　ゼロディテクタ
１３９　バスマルチプレクサ
１４０　積和演算回路

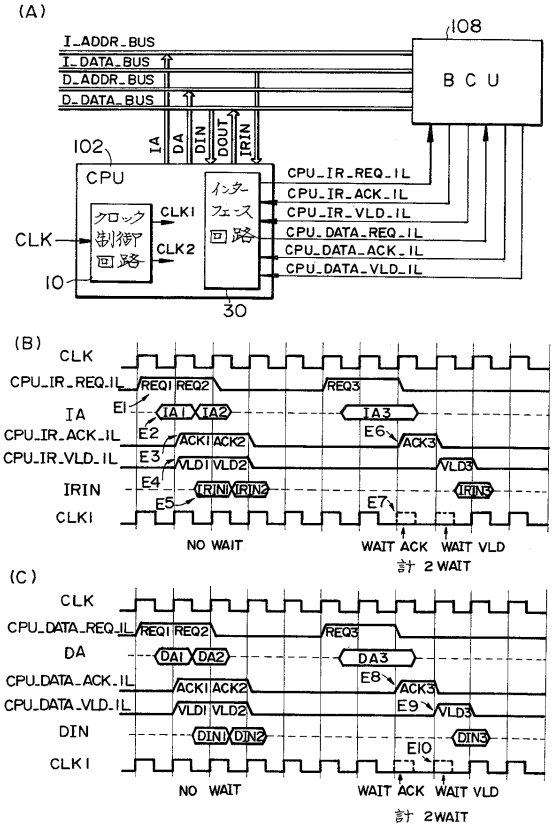
【図１】


外部メモリ
外部データバス
外部アドレスバス
CLK
100
内部メモリ
BCU
インターフェース回路
AUX_BUS
補助演算回路
クロック制御回路
CLK1
CLK2
バスマルチプレクサ
ALU
桁シフタ
PSR
SP
汎用レジスタ
R15
R2
R0
レジスタ制御回路
constant 0,±1,±2,±4
PA_BUS
PB_BUS
WW_BUS
XA_BUS
IRIN
TRAP VECTOR
命令デコーダ
即値生成器
アドレス加算器
DOUT
DIN
DA
IA
I_ADDR_BUS
I_DATA_BUS
D_ADDR_BUS
D_DATA_BUS
PC
CPU
マイクロコンピュータ

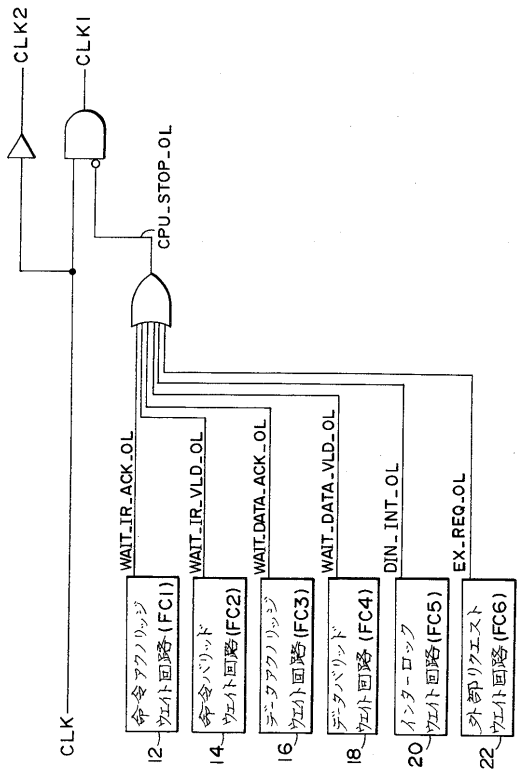
【図２】


(A)
CLK
CLK1
LD %R0,[%R8]
(外部メモリ)
LD %R1,[%R9]
(内部メモリ)
ADD %R0,%R1
アドレス
データ
F D R A E W W W
F D R A A A E W
F D D D R A E W
D1
D2
D3
[R8]
[R9]
←R0
←R1
(B)
D6
LD %R1,[%R3]
ADD %R5,%R1
[R3]
D7
ADD
←R5
D4
D5

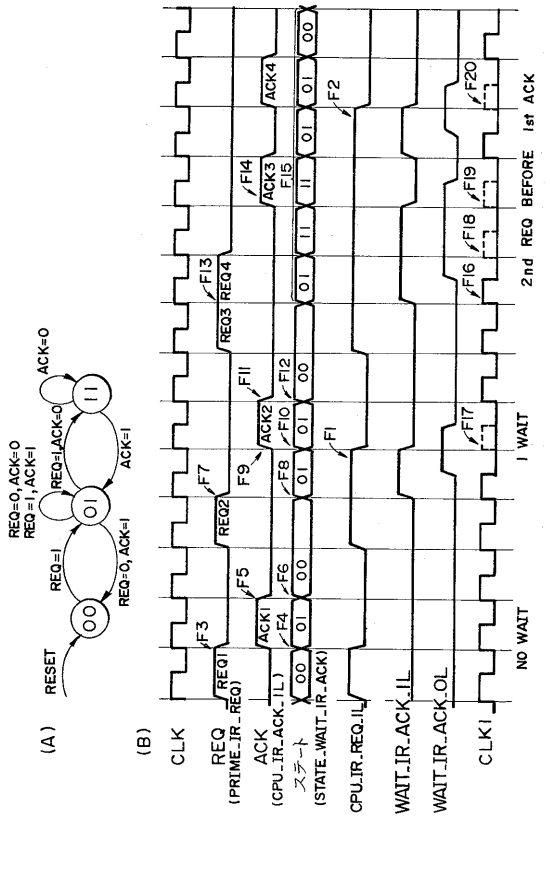
【図３】


(A)
I_ADDR_BUS
I_DATA_BUS
D_ADDR_BUS
D_DATA_BUS
108
BCU
102
CPU
IA
DA
DIN
DOUT
IRIN
クロック制御回路
インターフェース回路
CLK1
CLK2
CLK
10
30
CPU_IR_REQ_IL
CPU_IR_ACK_IL
CPU_IR_VLD_IL
CPU_DATA_REQ_IL
CPU_DATA_ACK_IL
CPU_DATA_VLD_IL
(B)
REQ1 REQ2 REQ3
IA1 IA2 IA3
ACK1 ACK2 ACK3
VLD1 VLD2 VLD3
IRIN1 IRIN2 IRIN3
E1 E2 E3 E4 E5 E6 E7
NO WAIT
WAIT ACK
WAIT VLD
計 2 WAIT
(C)
DA1 DA2 DA3
DIN1 DIN2 DIN3
E8 E9 E10
NO WAIT
WAIT ACK
WAIT VLD
計 2WAIT

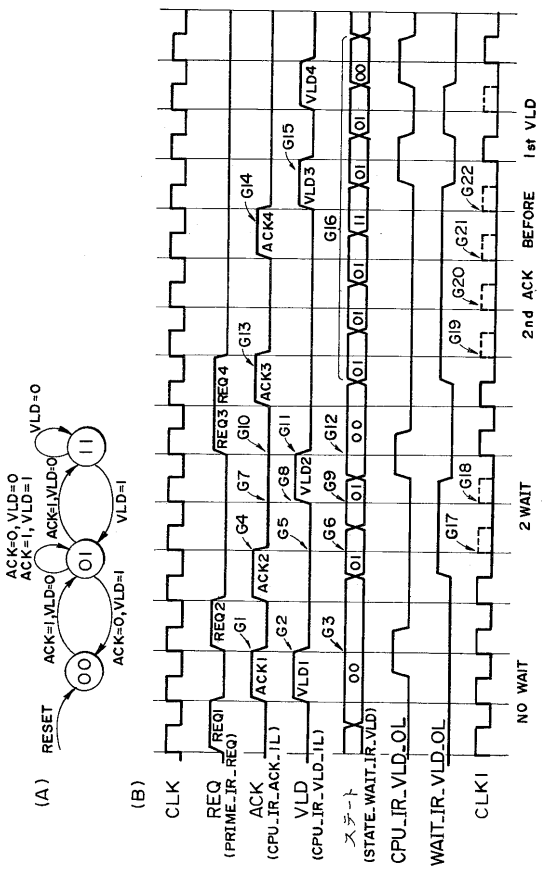
【図４】


CLK2
CLK1
CPU_STOP_OL
CLK
WAIT_IR_ACK_OL
WAIT_IR_VLD_OL
WAIT_DATA_ACK_OL
WAIT_DATA_VLD_OL
DIN_INT_OL
EX_REQ_OL
命令アクノリッジウェイト回路(FC1)
命令バリッドウェイト回路(FC2)
データアクノリッジウェイト回路(FC3)
データバリッドウェイト回路(FC4)
インターロックウェイト回路(FC5)
外部リクエストウェイト回路(FC6)
12
14
16
18
20
22

【図５】



【図６】

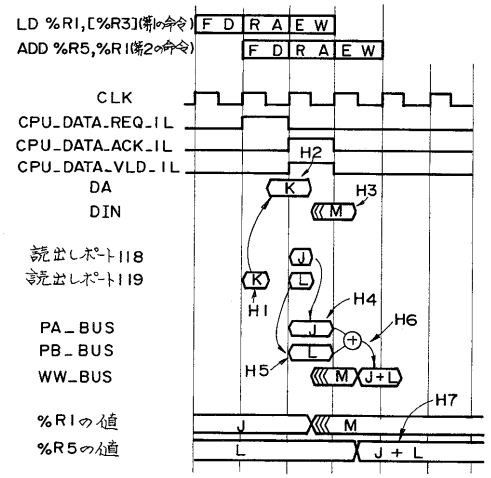


【図７】

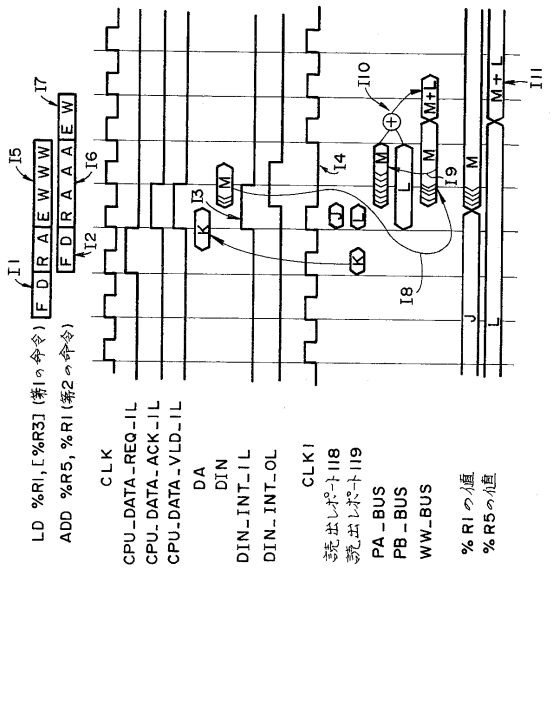
(A)

	命令実行前	命令実行後
%R1	−(J)	M
%R3	K	K
%R5	L	M+L
[K]	M	M

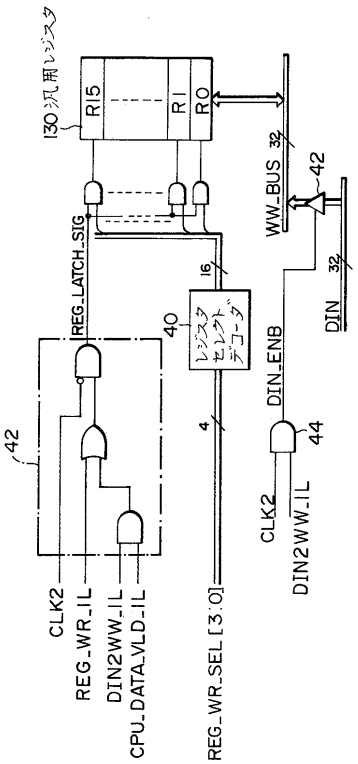
(B)



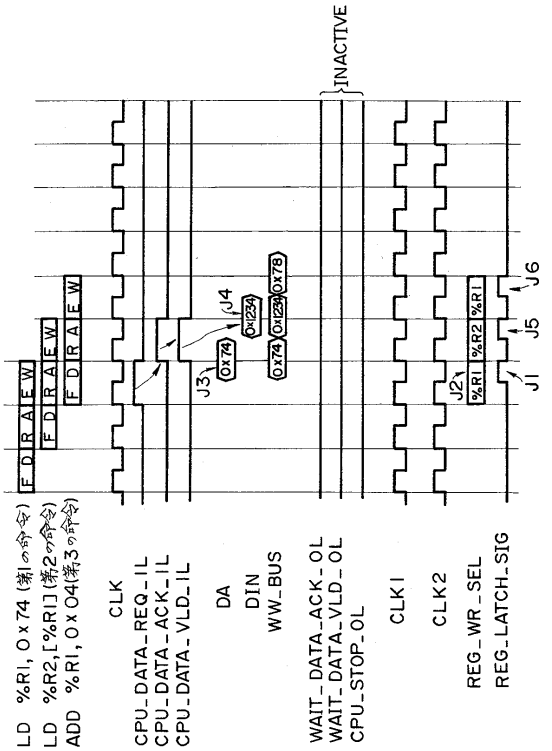
【図８】



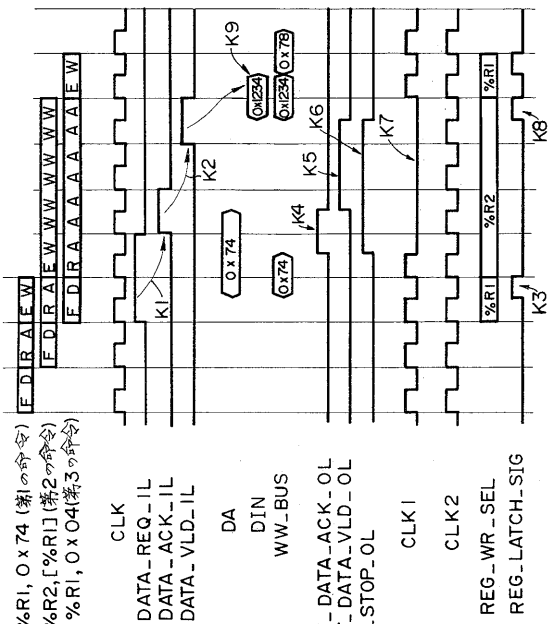
【図９】



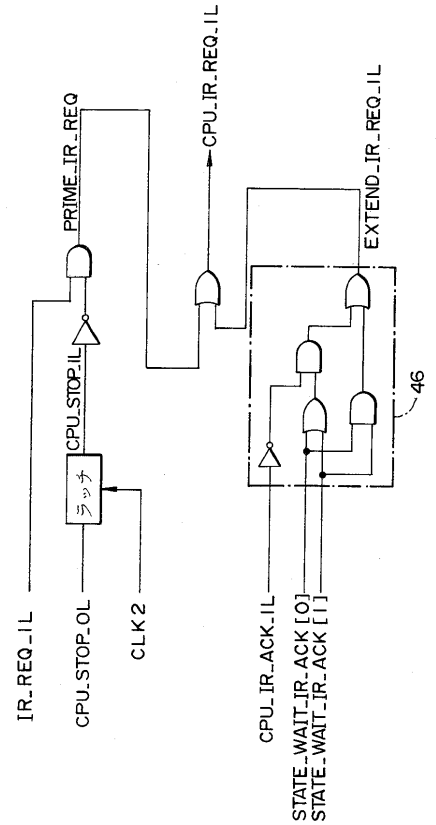
【図１０】



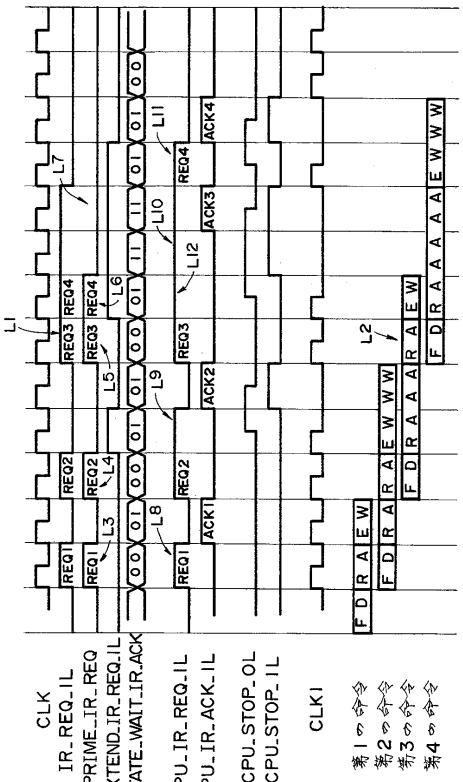
【図１１】



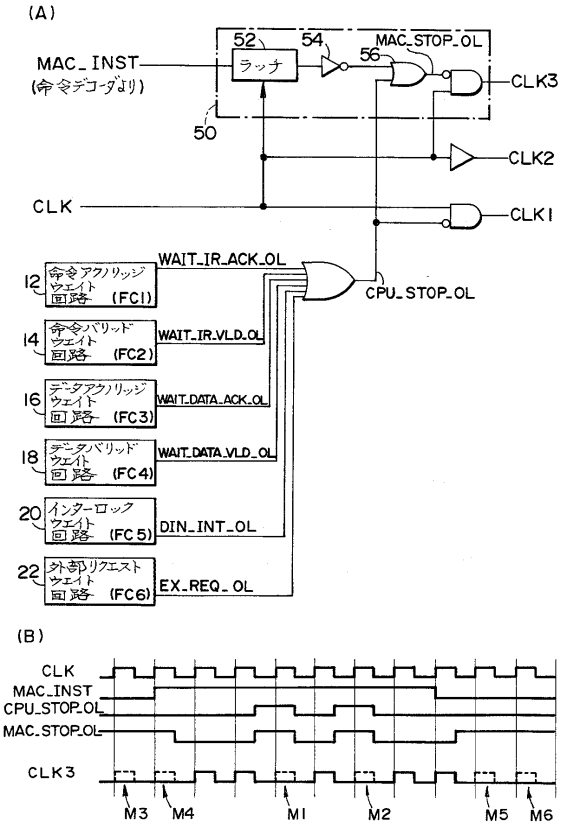
【図１２】



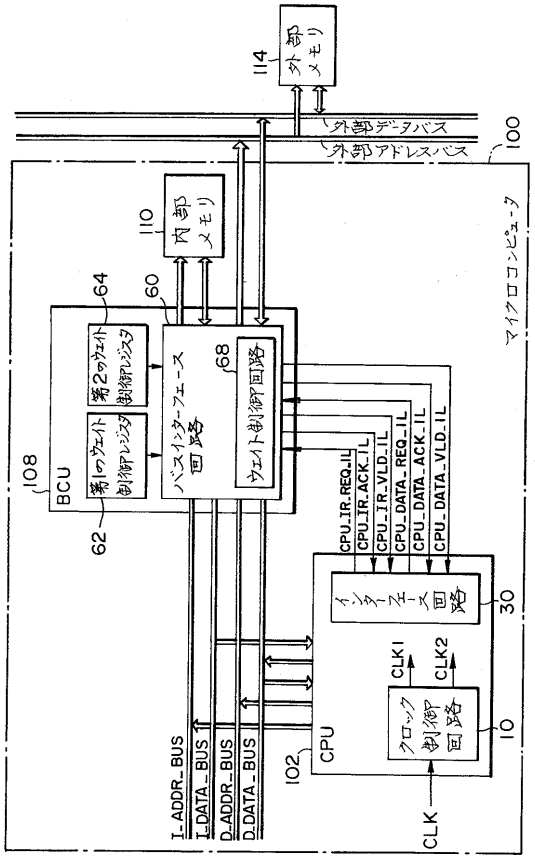
【図１３】



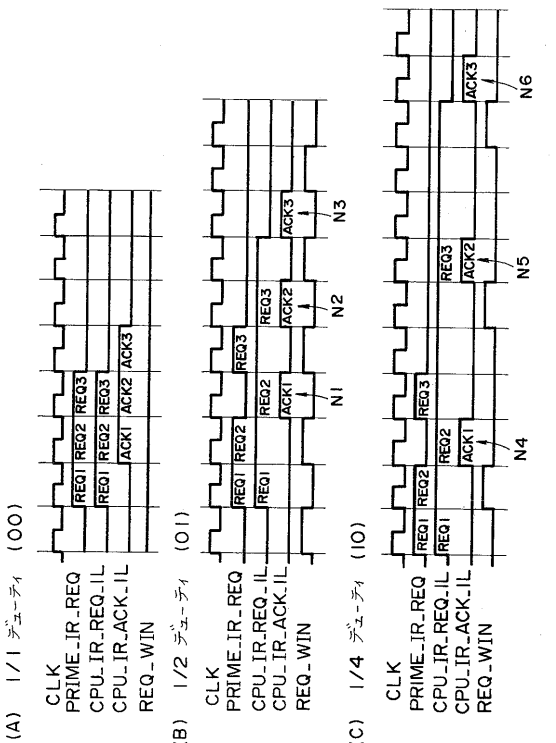
【図１４】



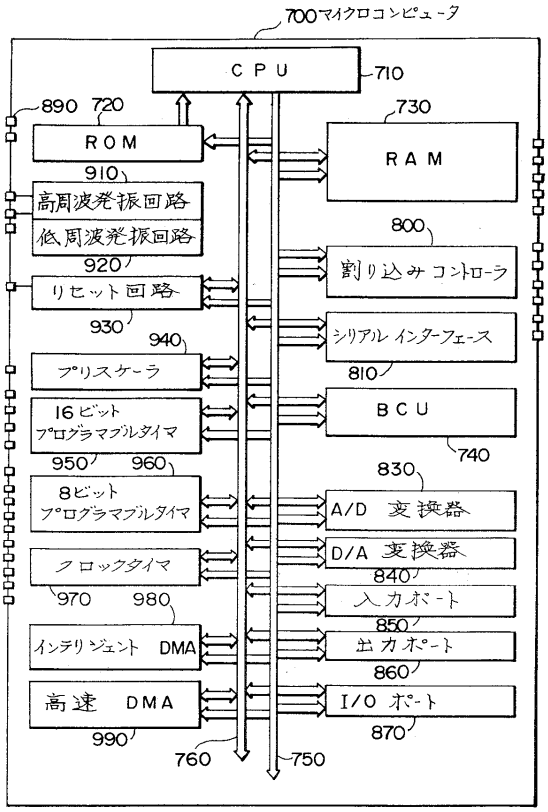
【図１５】



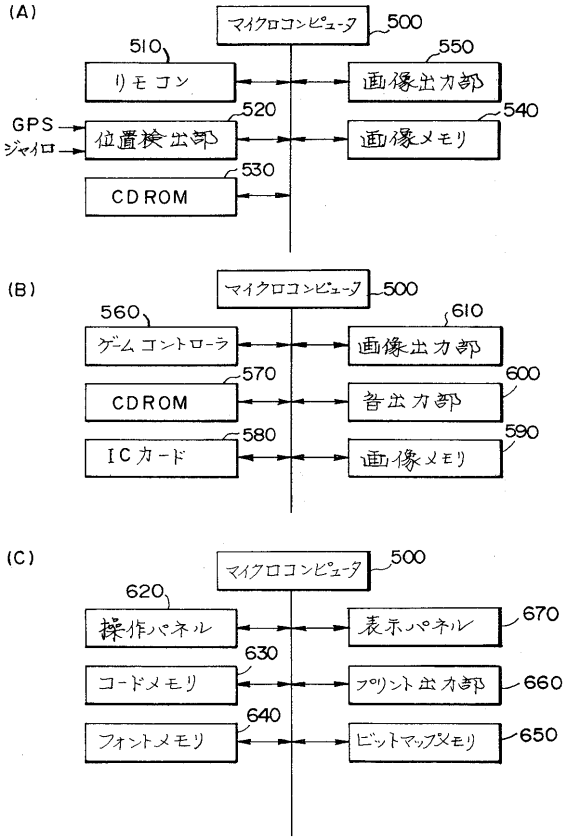
【図１６】



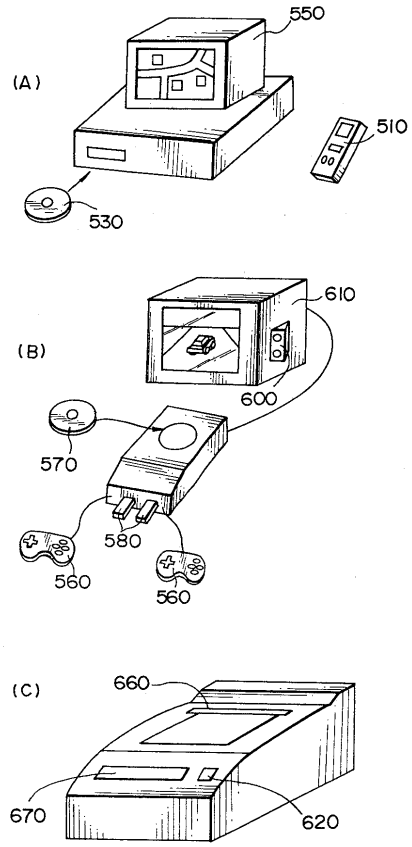
【図１７】


700マイクロコンピュータ
CPU
710
890
720
ROM
730
RAM
910
高周波発振回路
低周波発振回路
920
800
割り込みコントローラ
リセット回路
930
940
シリアルインターフェース
810
プリスケーラ
16ビット
プログラマブルタイマ
BCU
740
950
960
830
8ビット
プログラマブルタイマ
A/D 変換器
D/A 変換器
840
クロックタイマ
970
980
入力ポート
850
インテリジェント DMA
出力ポート
860
高速 DMA
I/O ポート
990
870
760
750

【図１８】


(A)
マイクロコンピュータ
500
510
リモコン
550
画像出力部
GPS
ジャイロ
520
位置検出部
540
画像メモリ
530
CDROM
(B)
マイクロコンピュータ
500
560
ゲームコントローラ
610
画像出力部
570
CDROM
600
音出力部
580
ICカード
590
画像メモリ
(C)
マイクロコンピュータ
500
620
操作パネル
670
表示パネル
630
コードメモリ
660
プリント出力部
640
フォントメモリ
650
ビットマップメモリ

【図１９】


(A)
550
510
530
(B)
610
600
570
580
560
560
(C)
660
670
620

フロントページの続き

(56)参考文献　特開平０６－２９５２４３（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)

G06F 1/04 301
G06F 9/38 310
G06F 9/38 350