

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第4348664号
(P4348664)

(45) 発行日 平成21年10月21日 (2009. 10. 21)

(24) 登録日 平成21年7月31日 (2009. 7. 31)

(51) Int. Cl.	F I
G 0 5 B 19/05 (2006. 01)	G O 5 B 19/05 N
G 0 6 F 9/50 (2006. 01)	G O 6 F 9/06 6 4 O H
G 0 6 F 12/06 (2006. 01)	G O 6 F 12/06 5 1 5 J
G 0 6 F 12/16 (2006. 01)	G O 6 F 12/16 3 4 O H

請求項の数 3 (全 9 頁)

(21) 出願番号	特願2000-195131 (P2000-195131)	(73) 特許権者	000002945
(22) 出願日	平成12年6月28日 (2000. 6. 28)		オムロン株式会社
(65) 公開番号	特開2002-14705 (P2002-14705A)		京都市下京区塩小路通堀川東入南不動堂町
(43) 公開日	平成14年1月18日 (2002. 1. 18)		8 0 1 番地
審査請求日	平成17年3月4日 (2005. 3. 4)	(74) 代理人	100069431
前置審査			弁理士 和田 成則
		(72) 発明者	大谷 修史
			京都府京都市右京区花園土堂町 1 0 番地
			オムロン株式会社内
		審査官	二階堂 恭弘
			最終頁に続く

(54) 【発明の名称】 プログラマブルコントローラの制御方法

(57) 【特許請求の範囲】

【請求項 1】

内蔵 R A M を持つ M P U と、電断時にバッテリーによってバックアップされる低速 R A M と、電断時にバックアップされない高速 R A M と、からなるプログラマブルコントローラの制御方法であって、

ソースプログラムおよび電断時に保持を必要とする保持変数を前記低速 R A M に格納する工程と、

電源 O N 時に前記低速 R A M 内のソースプログラムを実行形式であるオブジェクトプログラムに変換し、変換したオブジェクトプログラムを、電断時に保持を必要としない非保持変数とともに前記高速 R A M または前記内蔵 R A M に格納する工程と、

前記高速 R A M または内蔵 R A M 内のオブジェクトプログラムを読み出し実行する工程と、からなることを特徴とするプログラマブルコントローラの制御方法。

【請求項 2】

前記保持変数および非保持変数を定義するシステム設定として入力する工程をさらに含み、

前記システム設定にしたがって前記保持変数および前記非保持変数を格納することを特徴とする請求項 1 に記載のプログラマブルコントローラの制御方法。

【請求項 3】

前記オブジェクトプログラムおよび前記非保持変数を前記高速 R A M および前記内蔵 R A M のどちらかに格納するかをモード設定として入力する工程をさらに含み、

10

20

電源ON時に前記モード設定にしたがって前記オブジェクトプログラムおよび前記非保持変数を格納してなり、

前記モード設定は、前記オブジェクトプログラムの容量と前記内蔵RAMの容量の関係によって決定されることを特徴とする請求項1または2に記載のプログラマブルコントローラの制御方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

この発明は、ユーザ作成のプログラムおよびこのプログラムの実行過程で使用されたり、生成されたりする変数を格納するメモリを有し、ユーザの目的に応じてプログラム実行速度を切り替え使用でき、また、プログラム実行結果によるデータを保持することのできるプログラマブルコントローラの制御方法に関する。

10

【0002】

【従来の技術】

プログラマブルコントローラは、電源ユニット、CPUユニット、I/Oユニット等の多数のユニットで構成され、I/Oユニットの書き換え可能なメモリに、外部で組まれたユーザプログラム、例えば、ラダープログラムを書き込み、ユーザプログラムを実行するようになっている。

【0003】

上記I/Oユニットには、多数の制御対象機器が接続されるが、この際接続される制御対象機器に応じて、I/Oユニットには、任意の動作モードが設定される。

20

【0004】

具体的には、I/OユニットとCPUユニット間のデータリンク機能、I/Oユニット側からCPUユニットに対して任意の割り込みを発生させる機能、CPUユニット側と同期して動作する機能、CPUユニット側の故障を検出して、CPUユニットの故障時の処理をする機能、特殊入出力（パルス、アナログ、ID等）を扱う入出力ポートの制御機能などの動作モードが設定される。

【0005】

これによって、この発明では、ユーザはI/Oユニットに対して演算処理のためのユーザプログラムを入力して、ユーザオリジナルのユニット動作モードを作り上げ、制御対象、制御方式に最適な状態にカスタマイズして使用できるようにする。

30

【0006】

すなわち、I/Oユニットに取り込まれ実行されるプログラムは、ユーザがI/Oユニットで実行させたいプログラムで、ユーザが別途作成するカスタマイズされたプログラムである。このプログラムは、プログラマブルコントローラのCPUユニットでなされる処理プログラム（システムプログラム、ユーザプログラム）とは独立した別のもので、I/Oユニット向けで独自のI/O制御ロジックや任意の演算処理などを行うプログラムである。外部ツールを用いて作成されるプログラムで、ユーザが新規作成または書き換え（変更または追加）でき、ユーザが自由にプログラムできるものである。

【0007】

40

プログラマブルコントローラには、また、電源、CPU、I/O等を一体にまとめた一体型のマイクロPLCが知られているが、この発明のプログラマブルコントローラは、このマイクロPLCにも勿論適用できるものである。また、上述の説明から理解できるように、プログラマブルなI/Oユニットにも適用できるものである。

【0008】

プログラマブルコントローラにおいては、図7に示すように、例えば、ラダープログラムのようなソースプログラムと、これをコンパイルして実行形式に変換したオブジェクトプログラム（実行用プログラム）と、プログラム実行に使用する変数を、バッテリ1にバックアップされる共通の低速RAM2に格納するようになっている。上記変数には、電断時等にバックアップを必要とする保持変数とバックアップを必要としない非保持変数とがあ

50

る。この非保持変数とソースプログラムから容易にコンパイルできるオブジェクトプログラムとは、電断等の非常の場合は保持しなくてもよいので、バックアップを行わない方がバッテリー寿命が長くなって好ましい。しかし、低速RAM2内のエリアを仕分けてこのような処理をするのは煩雑なので、低速RAM2全体をバッテリー1でバックアップするのが普通である。

【0009】

なお、図7において、3は、プログラム実行処理等を行うMPU、4は、システムプログラム等を格納したROM、5は、プログラマブルコントローラ外部との入出力用のI/O、6は、低速RAM2、MPU3、ROM4、I/O5を相互に接続するバスである。

【0010】

【発明が解決しようとする課題】

ところで、近年は、プログラマブルコントローラの高速処理の要求が高まっている。しかし、高速処理には、次のような課題があった。

【0011】

1 高速処理のためには、MPU3に内蔵のRAMにオブジェクトプログラムを展開して格納しておき、プログラムを実行するのが最も望ましいが、内蔵RAMの容量が小さく、ラダープログラムをソースとしたような容量の大きいプログラムは格納できない。

【0012】

2 大きいプログラムを格納するために外部RAMを使用する場合は、プログラムの実行速度を上げるために、外部RAMとして高速タイプのSRAMが必要となる。しかし、この高速タイプのSRAMの記憶内容を電断時にバックアップすると、リテンション電流が大きくなり、バックアップ用のバッテリーの消耗が速く、低速RAMを用いた場合ならば数年のバッテリーのバックアップ維持期間があるのに対して、高速タイプSRAMを用いる場合はバッテリーのバックアップ維持期間が数か月と短くなってしまう。

【0013】

3 上記1および2の理由から、従来、図7のように、外部RAMとして低速RAMを採用するしかなかった。この場合は、バックアップ時のリテンション電流が小さく、バッテリーの寿命が長い。しかし、プログラム実行速度の高速化を図ることはできない。

【0014】

4 また、図7の場合、電断時に保持を必要とする保持変数と保持を必要としない非保持変数とを同一のRAMの連続したエリアに混在させて格納しているため、電断時に保持変数エリアと非保持変数エリアとを分けて保持変数だけを保持することは困難である。

【0015】

この発明は、上述の問題を解決し、ユーザプログラムとその変数を従来同様バッテリーで長期間バックアップできて、しかも、プログラムの高速処理ができるプログラマブルコントローラの制御方法を提供するものである。

【0016】

【課題を解決するための手段】

上述の目的を達成するために、この発明は、内蔵RAMを持つMPUと、電断時にバッテリーによってバックアップされる低速RAMと、電断時にバックアップされない高速RAMと、からなるプログラマブルコントローラの制御方法であって、ソースプログラムおよび電断時に保持を必要とする保持変数を前記低速RAMに格納する工程と、電源ON時に前記低速RAM内のソースプログラムを実行形式であるオブジェクトプログラムに変換し、変換したオブジェクトプログラムを、電断時に保持を必要としない非保持変数とともに前記高速RAMまたは前記内蔵RAMに格納する工程と、前記高速RAMまたは内蔵RAM内のオブジェクトプログラムを読み出し実行する工程と、からなることを特徴とする。

【0017】

内蔵RAMへの格納は、容量は小さいが高速処理に最適であるから、一般に、プログラム実行中頻繁に使用される非保持変数や比較的小さいプログラムをここに格納することが好ましい。

10

20

30

40

50

【0018】

保持変数エリアと非保持変数エリアとを区別することなく算出したオフセットアドレス（格納するメモリと関係なく最初の変数を原点として算出したアドレス）、にRAM毎のベースアドレスを加算して、低速RAMと高速処理用RAMとにともに保持変数のエリアと非保持変数エリアとを割り当てて使用すれば、ユーザは、格納場所を意識する必要がない。

【0019】

更に、システム設定部に、オブジェクトプログラムの実行時に使用する変数として、保持変数または非保持変数のいずれかを選択設定したり、高速処理モードとノーマル処理モードとを選択設定するようにしてもよい。このようにすると、アプリケーションプログラム 10 に応じてユーザが好適な処理速度を選択して実行することができる。

【0020】

また、上記システム設定部が、高速処理モードとノーマル処理モードとを選択設定するようになっており、高速処理モードのときは上記内蔵RAMのオブジェクトプログラムを、ノーマル処理モードのときは上記高速RAMのオブジェクトプログラムをオブジェクトプログラムの実行時に使用するようにしてもよい。このようにすると、高速モードの中でも、外部高速RAMを使用する普通高速モードと、MPU内蔵RAMを使用する超高速モードを選択できる。

【0021】

使用するプログラムがコンパイラでない場合は、ソース、オブジェクトの区別が存在しないので、プログラムを低速RAMに格納してバックアップに備える。なお、この場合、処理速度を上げるため、高速処理用RAMにも同一のプログラムを格納して、高速処理用RAM側のプログラムを用いて実行することになることは勿論である。

【0023】

この発明の「プログラマブルコントローラ」には、マイクロプログラマブルコントローラ（マイクロPLC）やプログラマブルI/Oユニットを含む。

【0024】

【発明の実施の形態】

この発明の実施の形態を、以下、図1～図6を参照して説明する。

【0025】

図1は、この発明の一実施の形態におけるプログラムおよびそのプログラムの変数の格納状態を示すブロック図である。図1において、既に説明した図7と同一の部分には、同一の符号を付して、その説明を省略する。7は、MPU3に内蔵された内蔵RAM、8は、高速処理用の高速RAM、9は、処理モード、プログラムと変数の格納場所を設定するシステム設定部である。

【0026】

ソースプログラム（ユーザ作成のラダープログラム）およびこのプログラム実行用の保持変数は低速RAM2に格納され、上記ソースプログラムをコンパイルしたオブジェクトプログラムおよびプログラム実行用の非保持変数は高速RAM8に格納されている。この高速RAM8は、この実施の形態では、32kBまでメモリエリアが使用でき、オブジェクトは2ステート実行できる。なお、オブジェクトプログラムのうち、比較的小さく、高速処理を必要とするものは、高速RAM8の代わりに内蔵RAM7に格納される。また、プログラム実行中に生成されたり、初期値として設定した非保持変数のうち、再使用頻度の高いものは内蔵RAM7に格納して、処理速度向上を図ることもできる。この内蔵RAM7は、この実施の形態では、12kBまでメモリエリアが使用でき、オブジェクトは1ステート実行できる。

【0027】

プログラムを実行するときは、MPU3からバス6を介して高速RAM8に格納されたオブジェクトプログラムを読み出し、MPU3でプログラムが実行される。

【0028】

10

20

30

40

50

電断時は、低速RAM2の保持電源がバッテリー1に切り替わり、低速RAM2の格納内容、すなわち、ソースプログラムと保持変数がバッテリー1により保持される。

【0029】

[メモリエリアの割り当て]

プログラムとその変数は、このように各RAM2、8、7に格納されるのであるが、次に、各RAM2、8、7内へのプログラムとその変数のエリア割り付けの仕方を説明する。

【0030】

ROM4、低速RAM2、高速RAM8の各メモリエリアのアドレス設定のために、図2に示すように、例えば、ROM4はエリア1、高速RAM8はエリア2、低速RAM2はエリア3のように定めてあり、エリア1にはベースアドレスH'00000、エリア2にはベースアドレスH'X0000、エリア3にはベースアドレスH'Y0000のようにベースアドレスで分けられ、エリア1には予めシステムプログラム等常駐のデータがエリア内に割り当てられて格納されている。エリア2およびエリア3には、図3に示すように、ユーザプログラムとその変数のエリアを割り当てる。この割り当ては、ユーザが使用するアプリケーションのユーザプログラムに応じて格納先をシステム設定部において設定できるようにしてある。

10

【0031】

図3において、ユーザ作成のソースプログラム格納用のソースプログラムエリア20、これをコンパイルしたオブジェクトプログラム格納用のオブジェクトプログラムエリア80は、それぞれ低速RAM2および高速RAM8内の先頭からその容量分だけ割り当てられる。

20

【0032】

すなわち、まず、ユーザが作成するソースプログラムは、低速RAMのソースプログラムエリア20に作成される。このソースプログラムはMPU3でコンパイルされ、そのオブジェクトプログラムが高速RAMのオブジェクトプログラムエリア80に展開される。また、このコンパイルの過程でこのプログラムで使用される変数のオフセットアドレス（エリア1、2、3に関係なく最初の変数の第1ビットを原点とした相対アドレス）が算出される。

【0033】

この変数には、既に説明したように、保持変数と非保持変数に区分けして使用する。この区分けは、後に説明するように、変数毎に予めシステム設定部9で設定しておく。

30

【0034】

保持変数と非保持変数とを格納するための変数エリアは、それぞれの変数に必要な容量分だけ順次、低速RAM2には21、22、・・・、高速RAM8には81、82、・・・と割り当てられていく。より具体的に説明すれば、低速RAM2では、ベースアドレスH'Y0000にソースプログラムエリア20の分を加算した実アドレスに最初の変数エリア21を、更に変数エリア21のオフセットアドレスを加算した実アドレスにその次の変数エリア22を、・・・というように、また、高速RAM8では、ベースアドレスH'X0000にオブジェクトプログラムエリア80の分を加算した実アドレスに最初の変数エリア81を、更に変数エリア81のオフセットアドレスを加算した実アドレスにその次の変数エリア82を、・・・というように変数エリアが割り当てられていく。

40

【0035】

図4は、保持変数、非保持変数の実際に使用する変数エリアの割り当てのフローを示す。保持変数、非保持変数の別は、予めシステム設定部9で設定しておき、このシステム設定部9の変数の保持、非保持の別を順次呼び出す（ステップ400）。呼び出した設定値をステップ401で判定し、設定値が「保持」である場合は、ステップ402に進んで低速RAM2のベースアドレスをセットし、設定値が「非保持」である場合は、ステップ403に進んで高速RAM8のベースアドレスをセットする。

【0036】

このように、保持変数と非保持変数とを区別することなく算出したオフセットアドレスと

50

、RAM 2、8 毎のベースアドレスH'X0000、H'Y0000とを加算して、低速RAM 2と高速RAM 8とにともに保持変数のエリアと非保持変数エリアとを割り当て、これらのエリアにシステム設定部に設定された定義に応じて、低速RAM 2の保持変数のエリア23、25、・・・と高速RAM 8の非保持変数エリア81、82、84、86、・・・とにアクセスして、変数の書き込み、読み出しを行うことができるようにする。なお、低速RAM 2の非保持変数エリア21、22、24、26、・・・と高速RAM 8の保持変数エリア83、85、・・・は、「空き」領域となる。

【0037】

〔処理モードの設定〕

図5は、システム設定部においてシステム設定をする設定画面の例を示し、通常10
の速度でプログラムを実行するノーマル処理モードと高速実行する高速処理モードとを設定したものである。図5では、ノーマル処理モードでは、4キロワードまでのメモリを、高速処理モードでは1キロワードまでのメモリを、それぞれ使用できることが備考欄に表示され、作業者は、この備考を参照しながら使用するプログラムが高速処理可能か否かを判断して、いずれかの処理を設定する。ここで、「高速処理モードで1キロワードまでのメモリ可」ということは、少なくともオブジェクトプログラムエリアが、望ましくは、これに加えて更に使用頻度の高い非保持変数エリアが、(MPU内蔵RAM 7内で使用できる) 1キロワード以下であれば高速処理モードを採用できる、ということを意味している。

【0038】

図6は、図5の処理モードの設定状態に応じてオブジェクトプログラムがいずれかの高速20
処理用RAMに格納する処理を示すフローチャートである。システム設定部9で設定された処理モード(ステップ500)は、ステップ501で判定され、処理モードが高速処理に設定されているときは、ステップ502へ進んでコンパイルされたオブジェクトプログラムの先頭アドレスと最終アドレスがMPU内蔵RAM 7にセットされ、処理モードがノーマル処理に設定されているときは、ステップ503へ進んでコンパイルされたオブジェクトプログラムの先頭アドレスと最終アドレスが高速RAM 8にセットされる。

【0039】

システム設定部9では、更に、使用目的に応じて、オブジェクトプログラムの実行時に使用する変数として、保持変数または非保持変数を選択設定することもできる。保持変数を使用する場合は、非保持変数を使用する場合よりも処理速度は遅くなるが、実行時に使用する変数の電断時の保持ができる。30

【0040】

以上、システム設定部における設定による代表的な各メモリの使い方を整理すると、次のようになる。

【0041】

(1) ノーマル処理モード： オブジェクトプログラムと非保持変数を高速RAMに格納。変数は、非保持変数／保持変数の両方または一方を使用。

【0042】

(2) 高速処理モード： オブジェクトプログラムを内蔵RAMに格納。変数は、内蔵RAM／高速RAMの非保持変数の両方または一方を使用。40

【0043】

(3) 低速処理モード： オブジェクトプログラムを高速処理用RAMに格納。変数は、保持変数を使用。

【0044】

この発明は、上記(1)～(3)の他、種々の変形が可能である。例えば、高速RAMとMPU内蔵RAMとの使い分け方として、サブルーチンプログラムを含む複数のプログラムを実行させる場合、プログラム実行中使用頻度が高くて比較的小さいプログラムと使用頻度が高い非保持変数とをMPU内蔵RAMに格納するようにシステム設定部で選択設定して、内蔵RAMのメモリを節約しながら高速処理を実現したり、保持変数と非保持変数の仕分け方として、保持が必要で、かつ、プログラム実行中使用頻度の高い変数を、低速50

R A Mと高速処理用 R A Mとに重複して格納して、高速処理用 R A Mの保持変数を使用して高速処理しながら低速 R A Mの保持変数を電断時保持して、高速処理と保持とを両方実現したりすることもできる。

【 0 0 4 5 】

【発明の効果】

以上の説明から理解されるように、この発明によれば、バックアップに要する電流が小さい低速 R A Mに、ソースプログラムと電断時に保持を要する保持変数とを格納し、高速 R A Mに、保持を要しないオブジェクトプログラムと非保持変数を格納するようにしたから、プログラム実行速度を上げながら、バックアップ時のバッテリーのリテンション電流が少なく、バックアップ用バッテリーの寿命が長くなって、ソースプログラムと保持変数を長期間保持できる。

10

【 0 0 4 6 】

また、使用するアプリケーションのユーザプログラムに応じて個別に保持変数、非保持変数を定義したり、実行処理モードを指定するようにしたから、使用目的に応じて、プログラム実行速度を一層上げたり、プログラマブルコントローラが制御する工場内の機器の作動速度に合わせて適度の処理速度を選択したりすることができる。

【図面の簡単な説明】

【図 1】 この発明の一実施の形態におけるプログラムおよびデータの格納状態を示すブロック図。

【図 2】 図 1 におけるメモリマップを説明する説明図。

20

【図 3】 図 1 における高速 R A Mと低速 R A Mの内容を説明する説明図。

【図 4】 図 1 における変数メモリのアドレス設定処理を示すフローチャート。

【図 5】 この発明の一実施の形態におけるシステム設定例を示す説明図。

【図 6】 この発明の一実施の形態におけるオブジェクトプログラム格納メモリのアドレス設定処理を示すフローチャート。

【図 7】 従来のプログラマブルコントローラにおけるプログラムおよびデータの格納状態を示すブロック図。

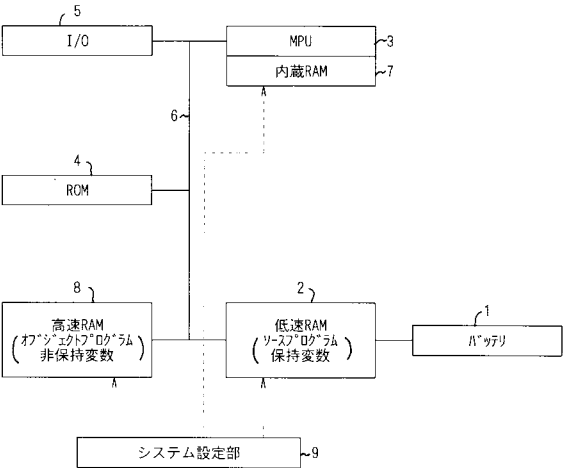
【符号の説明】

- 1 バッテリ
- 2 低速 R A M
- 3 M P U
- 4 R O M
- 5 I / O
- 6 バス
- 7 内蔵 R A M（高速処理用 R A M）
- 8 高速 R A M（高速処理用 R A M）
- 9 システム設定部
- 2 0 ソースプログラムエリア
- 2 1、2 2、・・・ 変数エリア
- 8 0 オブジェクトプログラムエリア
- 8 1、8 2、・・・ 変数エリア

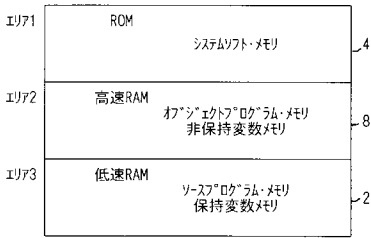
30

40

【図 1】



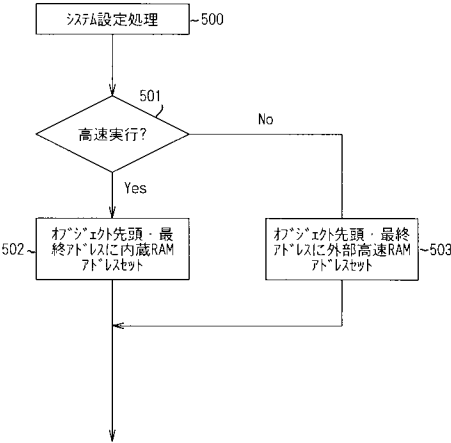
【図 2】



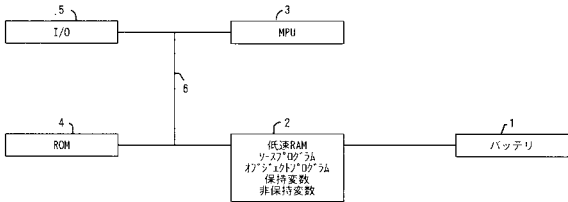
【図 5】

CH No.	Bit	機能	備考
DM m	00-15	高速演算指定	0000:ノーマルモード UM4kwまで可能
			5A5A:高速モード UM1kwまで可能

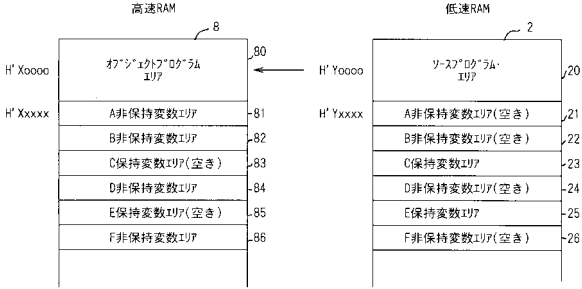
【図 6】



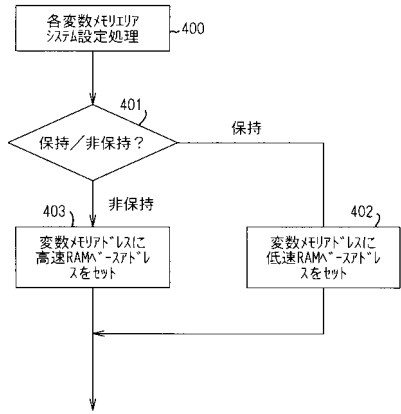
【図 7】



【図 3】



【図 4】



フロントページの続き

- (56)参考文献 特開平04-118705 (JP, A)
特開平08-006616 (JP, A)
特開平05-165510 (JP, A)
特開平04-280324 (JP, A)
特開平06-175916 (JP, A)
特開2000-122877 (JP, A)
特開2000-029783 (JP, A)
特開平06-051810 (JP, A)
特開平11-003106 (JP, A)

(58)調査した分野(Int.Cl., DB名)

G05B 19/04-19/05

G06F 9/50

G06F 12/06

G06F 12/16