(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4051091号
(P4051091)

(45)発行日　平成20年2月20日(2008.2.20)　(24)登録日　平成19年12月7日(2007.12.7)

(51)Int.Cl.　F I
G06F 11/00 (2006.01)　G06F 9/06 630A

請求項の数 33 (全 19 頁)

(21)出願番号	特願平10-539543
(86)(22)出願日	平成9年4月3日(1997.4.3)
(65)公表番号	特表2002-516005(P2002-516005A)
(43)公表日	平成14年5月28日(2002.5.28)
(86)国際出願番号	PCT/US1997/005622
(87)国際公開番号	WO1998/040818
(87)国際公開日	平成10年9月17日(1998.9.17)
審査請求日	平成15年9月11日(2003.9.11)
(31)優先権主張番号	08/818,389
(32)優先日	平成9年3月13日(1997.3.13)
(33)優先権主張国	米国(US)

(73)特許権者
マクロニクス　インターナショナル　カンパニー　リミテッド
台湾　シン　チュ　サイエンス　ベースド　インダストリアル　パーク　クリエイション　ロード　サード　ナンバー３
(74)代理人
弁理士　中村　稔
(74)代理人
弁理士　大塚　文昭
(74)代理人
弁理士　熊倉　禎男
(74)代理人
弁理士　宍戸　嘉一

最終頁に続く

(54)【発明の名称】ＲＯＭとフラッシュメモリーを有する回路内プログラミング構造

(57)【特許請求の範囲】
【請求項１】
集積回路の回路内プログラミング用装置において、
命令を実行する前記集積回路上のプロセッサ、
外部ソースからデータを受取るための前記集積回路上の外部ポート、
前記集積回路上の複数の浮遊ゲートメモリーセルを含む不揮発性メモリーアレーセルを備える第１メモリーアレーであって、前記外部ソースから前記外部ポートを通って前記集積回路内への命令の転送を制御するための１組の命令を含む前記プロセッサにより実行される命令を記憶する第１メモリーアレー、及び、
前記集積回路上の複数のマスクＲＯＭセルを備える第２メモリーアレーであって、前記第１メモリーアレー内の命令を消去、プログラム、検証する回路内プログラミングステップを制御するための１組の命令を含む前記プロセッサにより実行される命令を記憶する第２メモリーアレー、を備え、
前記命令の転送を制御するための１組の命令によって転送されてきた命令が、前記第２メモリーアレー内のプログラムによって前記第１メモリーアレー内へ書き込まれることを特徴とする装置。
【請求項２】
前記第１メモリーアレーのための消去、プログラム、検証オペレーションの順序付けは、前記プロセッサにより実行される１組の命令により制御御される請求項１に記載した回路内プログラミング用装置。

【請求項３】
前記第１メモリーアレーのための消去、プログラム、検証オペレーションの順の順の順序付けは、前記プロセッサにより実行され、前記第２メモリーアレーのマスクＲＯＭセルに記憶される１組の命令により制御される請求項１に記載した回路内プログラミング用装置。
【請求項４】
前記プロセッサは、前記第１メモリーアレーのための消去、プログラム、検証オペレーションを、前記第１メモリーアレーに結合した制御レジスターにより制御する請求項２に記載した回路内プログラミング用装置。
【請求項５】
前記第１メモリーアレーのための前記消去、プログラム、検証オペレーションのタイミングは、前記プロセッサに含まれるタイマー機能により制御される請求項１に記載した回路内プログラミング用装置。
【請求項６】
前記第１メモリーアレーのための前記消去、プログラム、検証オペレーションのタイミングは、前記プロセッサにより実行される１組の命令により制御される請求項１に記載した回路内プログラミング用装置。
【請求項７】
前記第１メモリーアレーのための前記消去、プログラム、検証オペレーションのタイミングは、前記プロセッサにより実行され、前記第２メモリーアレーのマスクＲＯＭセルに記憶される１組の命令により制御される請求項１に記載した回路内プログラミング用装置。
【請求項８】
前記プロセッサに結合し、前記プロセッサによる前記回路内プログラミング命令実行中のエラーからの回復をトリガするウォッチドッグタイマーを備える請求項１に記載した回路内プログラミング用装置。
【請求項９】
前記第１メモリーアレーは、不揮発性メモリーセルの複数の別々に消去可能なブロックを備える請求項１に記載した回路内プログラミング用装置。
【請求項１０】
前記外部ポートは、直列ポートである請求項１に記載した回路内プログラミング用装置。
【請求項１１】
前記外部ポートは、並列ポートである請求項１に記載した回路内プログラミング用装置。
【請求項１２】
前記外部ポートは、並列ポート又は直列ポートとして構成できる請求項１に記載した回路内プログラミング用装置。
【請求項１３】
外部データソースへの複数のポートを備え、前記複数のポートのうち前記外部ポートのために使用されるポートは、回路内プログラミングステップを制御するための前記１組の命令内の命令により決められる請求項１に記載した回路内プログラミング用装置。
【請求項１４】
集積回路の回路内プログラミング用装置において、
命令を実行する前記集積回路上のプロセッサ、
外部ソースからデータを受取るための前記集積回路上の外部ポート、
前記集積回路上の浮遊ゲートメモリーセルを備える第１メモリーアレーであって、前記外部ソースから前記外部ポートを通って前記集積回路内への命令の転送を制御するための１組の命令を含む前記プロセッサにより実行される命令を記憶する第１メモリーアレー、
前記集積回路上のマスクＲＯＭセルを備える第２メモリーアレーであって、前記第１メモリーアレー内の命令を消去、プログラム、検証する回路内プログラミングステップを制御、この順の順序付け、タイミング合わせするための１組の命令を含む前記プロセッサにより実行される命令を記憶する第２メモリーアレー、及び、

前記第１メモリーアレーに結合し、前記プロセッサが前記第１メモリーアレー内の消去、プログラム、検証を制御するための制御レジスター、
を備え、
前記命令の転送を制御するための１組の命令により転送されてきた命令が、前記第２メモリーアレー内のプログラムによって前記第１メモリーアレー内へ書き込まれることを特徴とする装置。
【請求項１５】
前記プロセッサに結合し、前記プロセッサが前記回路内プログラミング命令を実行する間にエラーからの回復をトリガするウォッチドッグタイマーを備える請求項１４に記載した回路内プログラミング用装置。
【請求項１６】
前記第１メモリーアレーは、不揮発性メモリーセルの複数の別々に消去可能なブロックを備える請求項１４に記載した回路内プログラミング用装置。
【請求項１７】
前記外部ポートは、直列ボートである請求項１４に記載した回路内プログラミング用装置。
【請求項１８】
前記外部ポートは、並列ポートである請求項１４に記載した回路内プログラミング用装置。
【請求項１９】
前記外部ポートは、並列ポート又は直列ポートとして構成できる請求項１４に記載した回路内プログラミング用装置。
【請求項２０】
外部データソースへの複数のポートを備え、前記複数のポートのうち前記外部ポートのために使用されるポートは、回路内プログラミングステップを制御するための前記１組の命令内の命令により決められる請求項１４に記載した回路内プログラミング用装置。
【請求項２１】
プロセッサと外部ポートを備え、不揮発性メモリーセルを有する第１メモリーアレーと、複数のマスクＲＯＭセルを備える第２メモリーアレーを備える集積回路において、前記集積回路の回路内プログラミングの方法において、
前記プロセッサが、前記集積回路の外部のイニシエータから、回路内プログラムコマンドを受取り、
前記回路内プログラムコマンドに応答して、前記プロセッサが、前記第１メモリーアレー内の命令を消去、プログラム、検証する回路内プログラミングステップを制御するため前記第２メモリーアレー内の１組の命令を実行し、
前記プロセッサが、外部ソースから前記外部ポートを通って前記集積回路内への１組の命令の転送を制御するため、前記第１メモリーアレーからの１組の命令を実行する、ステップを備え、
前記命令の転送を制御するための１組の命令により転送されてきた命令が、前記第２メモリーアレー内のプログラムによって前記第１メモリーアレー内へ書き込まれることを特徴とする方法。
【請求項２２】
前記第１メモリーアレー内の命令を消去、プログラム、検証する回路内プログラミングステップを制御する前記１組の命令は、前記第２メモリーアレーに記憶される請求項２１に記載した集積回路の回路内プログラミング方法。
【請求項２３】
前記第１メモリーアレー内の前記不揮発性メモリーセルは、浮遊ゲートメモリーセルを備える請求項２１に記載した集積回路の回路内プログラミング方法。
【請求項２４】
前記プロセッサが、命令を消去、プログラム、検証する回路内プログラミングステップを

制御するための１組の命令を実行する前記ステップは、消去、プログラム、検証オペレーションの順の順序付けの制御を含む請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項２５】

前記プロセッサは、前記第１メモリーアレーのための消去、プログラム、検証オペレーションを、前記第１メモリーアレーに結合した制御レジスターにより制御する請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項２６】

前記第１メモリーアレーのための前記消去、プログラム、検証オペレーションのタイミングは、前記プロセッサに含まれるタイマー機能により制御される請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項２７】

前記プロセッサが、命令を消去、プログラム、検証する回路内プログラミングステップを制御するための１組の命令を実行する前記ステップは、消去、プログラム、検証オペレーションのタイミングの制御を含む請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項２８】

前記プロセッサに結合し、前記プロセッサが前記回路内プログラミング命令を実行する間にエラーからの回復をトリガするウォッチドッグタイマーを備える請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項２９】

前記第１メモリーアレーは、不揮発性メモリーアレーの複数の別々に消去可能なブロックを備える請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項３０】

前記外部ポートは、直列ポートである請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項３１】

前記外部ポートは、並列ポートである請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項３２】

前記外部ポートは、並列ポート又は直列ポートとして構成できる請求項２１に記載した集積回路の回路内プログラミング方法。

【請求項３３】

前記集積回路は、外部データソースへの複数のポートを備え、前記複数のポートのうち前記外部ポートのために使用されるポートを、前記プロセッサが、回路内プログラミングステップを制御するための前記１組の命令内の命令を実行することにより決めるステップを備える請求項２１に記載した集積回路の回路内プログラミング方法。

【発明の詳細な説明】

37C.F.R.1.71(e)による著作権

関連出願の参照

本出願は、1996年10月28日に出願された「回路内プログラミング構造が組込まれたプロセッサー」という題の国際出願No.PCT/US/96/17302に関連する。この出願は、米国外ではマクロニクスインターナショナルが出願人で、米国内ではアルバートC.サン、チーH.リー、チャングL.チェンが出願人である。

背景

発明の技術分野

本発明は、集積回路上のマイクロコントローラにより実行する命令のシーケンスを記憶す

るための不揮発性メモリーを有する集積回路に関し、より詳しくは、本発明は、記憶した命令のシーケンスを更新し修正するため回路内プログラミングを実行する技術に関する。

関連技術

マイクロコントローラにより実行される命令のシーケンスを記憶するため、集積回路上に不揮発性メモリーのアレーを備える集積回路マイクロコントローラが、開発されてきた。命令のシーケンスは、読み出し専用メモリー（ＲＯＭ）に記憶され、それはデバイスの製造中にプログラムされなければならず、更新することができなかった。他のアプローチでは、命令はＥＰＲＯＭアレーに記憶された。しかし、これらのデバイスは、デバイスを回路内に配置する前にＥＰＲＯＭアレーをプログラムするため、特別のハードウェアを必要とした。更に他のシステムでは、命令を記憶するのにＥＥＰＲＯＭメモリーが使用された。ＥＥＰＲＯＭは、ＥＰＲＯＭよりずっと速くプログラムすることができ、進行中に修正することができ、有利である。更に他のアプローチでは、命令を記憶するのにフラッシュメモリーが使用され、それにより不揮発性メモリーのより高密度でより高速のプログラミングを行うことができる。デバイスが、ＥＥＰＲＯＭ又はフラッシュメモリー等の再プログラム可能な不揮発性メモリーをマイクロコントローラ結合するとき、デバイスは、回路内で再プログラム可能であり、対話形アルゴリズムに基づき回路内プログラミングが可能になる。

命令の組とデータを遠隔のデバイスに対話的にダウンロードする能力は、ネットワーク環境では非常に価値がある。例えば、顧客が機器をサービスセンターに持ってこなくても、会社は、顧客の機器にサービスを提供することができる。むしろ、会社は、インターネット又は電話線等の通信回線を通り、顧客の機器の回路内プログラミング能力を使用して、診断機能を実行することができる。このように、ソフトウェアのフィックスは、顧客の機器にダウンロードすることができ、機器は訂正した即ち更新したコードで再度可能にされる。

この能力を備える従来のデバイスには、カリフォルニア州サンジョーズのアトメルにより製造されるAT89S8252マイクロコントローラ、及びオランダ、アインドホーベンのフィリップスセミコンダクターズにより製造されるP89CE558シングルチップマイクロコントローラがある。フィリップスP89CE558マイクロコントローラのアーキテクチャに従うと、命令の回路内プログラミング（ＩＰＣ）の組のため、マスクＲＯＭが使用され、その命令は、チップ上のフラッシュメモリーを更新するため、マイクロコントローラにより使用される。従って、フィリップスのマイクロコントローラは、個々の環境でフィックスされたＩＰＣコードを記憶するため、専用マスクＲＯＭを必要とする。ＩＰＣコードを個々の環境に適合させるためには、マスクＲＯＭが適当にコード化できるようにデバイスの製造が完了する前に環境を知らなければならない。さらに、ＩＰＣ通信チャンネルは、フィリップスのマイクロコントローラの直列RS232ポートに固定されている。このため、マイクロコントローラの使用は、比較的狭い用途範囲に限られ、ダイナミック通信環境では、直列ポートは更新されたソフトウェアが供給される通信チャンネルと良く適合せず、ＩＰＣ機能を使用するのは困難になる。

アトメルAT89S8252マイクロコントローラのアーキテクチャによれば、チップ上の専用直列周辺インターフェース（ＳＰＩ）が、ポートフラッシュメモリーの更新に使用される。このＳＰＩポートは、柔軟性のないプログラムロジックで実行され、ＳＰＩポートの柔軟性がないため、回路内プログラミングの修正技術を実行できないという欠点がある。アトメルのチップは、更に次の欠点がある。ＩＰＣイニシエータとのハンドシェークと、フラッシュメモリー用に消去/プログラム/検証の波形をエミュレートするため、ＩＣＰチップに複雑なハードウェアを追加しなければならない。ＳＰＩバスは、必ずしも色々のシステムの用途に最適の選択ではない。回路内プログラミングアルゴリズムにより使用されるオリジナルのリセット回路を修正するのに、特別のシステムロジックを必要とする。複雑なＳＰＩドライバーとレシーバーのロジックをチップに取り付けなければならない。以上の欠点である。

回路内プログラミングプロセスで柔軟性を維持しながら、回路内プログラミング機能を実

行するため使用されるフラッシュメモリーにより占められるシリコンの面積量を最小にする、回路内プログラミングのためのアーキテクチャが必要である。

発明の概要

本発明は、回路内プログラミングをサポートする集積回路上のマイクロコントローラシステムのアーキテクチャを提供する。このシステムは、回路内プログラミング（ＩＣＰ）コードをフラッシュメモリーに記憶して、回路内プログラミングプロセスを使用して容易に変更することができるようにし、また修正する必要のないＩＣＰコードの部分を集積回路上のよりスペース効率の良いマスクＲＯＭセル内に記憶することにより、回路内プログラミングプロセスにおける柔軟性を維持する。特に、回路内プログラミングプロセスに含まれる通信を取扱うＩＰＣコードは、フラッシュメモリーに記憶されるので、幾つかの異なる通信フォーマットとプロトコルに適合するように容易に修正することができる。回路内プログラミングプロセスの消去、プログラム、検証の部分を実行するコードは、集積回路上のスペース効率の良いマスクＲＯＭセル内に保持される。

マスクＲＯＭに記憶されるソフトウェア内のフラッシュメモリーの消去、プログラム、検証機能のためのステートマシンを実行することにより、本発明のフラッシュメモリーアレーは、更に簡単にされる。一般にはハードウェアで実行される消去、プログラム、検証機能のタイミング機能は、マスクＲＯＭに記憶されるソフトウェア内で実行される。本発明のフラッシュメモリーの設計は、簡単にされ、その結果、フラッシュメモリーにより占められるシリコンの面積は減少する。このように、本発明の回路内プログラミングアーキテクチャは、回路内プログラミングシステムの柔軟性を効果的に維持し、回路内プログラミングシステムにより占められるシリコンの面積の要求を減少させる。

従って、本発明は、命令を実行する集積回路上のプロセッサを備える集積回路の回路内プログラミング用の装置として特徴付けられる。集積回路は外部ソースからデータを受取る外部ポートを備える。集積回路はまた、プロセッサーにより実行される命令を記憶するための不揮発性メモリーセルを有する第１メモリーアレーを備え、この命令は、外部ソースから外部ポートを通って集積回路内へ別の命令を転送することを制御するための命令を含む。集積回路は又、プロセッサにより実行される命令を記憶する第２メモリーアレーを備え、この命令は、第１メモリーアレーの命令を消去、プログラム、検証する回路内プログラミングステップを制御するための命令を含む。

本発明の１態様によれば、第２メモリーアレーは、複数の不揮発性マスクＲＯＭセルを備える。

本発明の他の態様によれば、回路内プログラミングプロセスの間の消去、プログラム、検証のオペレーションの順序付けは、第２メモリーアレー内のマスクＲＯＭセルに記憶されるソフトウェアにより達成される。

本発明の他の態様によれば、回路内プログラミングプロセスの消去、プログラム、検証のオペレーションのタイミングは、第２メモリーアレー内のマスクＲＯＭセルに記憶されるソフトウェアを実行するプロセッサにより達成される。

本発明のさらに他の態様によれば、プロセッサは、第１メモリーアレーに結合する制御レジスターに命令を書き込むことにより、回路内プログラミングプロセスに含まれる消去、プログラム、検証のオペレーションを制御する。

本発明の他の態様によれば、回路内プログラミング用装置はさらに、プロセッサに結合したウォッチドッグタイマーを備え、それがプロセッサが回路内プログラミング命令を実行している間に起こるデッドロックエラーからの回復をトリガする。

本発明の他の態様によれば、外部ソースからデータを受取る外部ポートは、並列又は直列ポートとして動作するように構成することができる。

本発明の他の態様によれば、第１メモリーアレーは、不揮発性メモリーアレーの複数の別々に消去可能なブロックを備える。

本発明のさらに他の態様によれば、集積回路は、１つ又はそれ以上の直列ポートと、１つ又はそれ以上の並列ポートと、１つ又はそれ以上の特別の通信ポート（任意）等の外部データソースへの複数のポートを備える。複数のポートのうち外部ソースから回路内プログ

ラミング命令を受取るために使用されるポートは、回路内プログラミングコード自体内の命令により決められ、従って動的に変更することができる。
本発明の他の態様によれば、集積回路は、第１メモリーアレーをプログラムし、検証するためのデータパスを備え、また任意に回路内プログラミング命令と独立の第２メモリーアレーを備える。従って、多重化されたＩ/Ｏピン等を使用して、製造中又はチップをシステムに取りつける前に、オリジナルのソフトウェアをデバイスにロードすることができる。
本発明の他の態様と利点は、図面、発明の詳細な説明、請求の範囲を見れば、分かる。
図面の説明
図１は、本発明の１態様による集積回路の回路内プログラミング用システムのブロック線図である。
図２は、本発明の１態様によるマイクロコントローラ100により実行されるアドレス空間を含む命令の線図である。
図３は、本発明の１態様による本発明が通信チャンネルと回路内プログラミングイニシエータ340を含むシステムに一体化されるかを示す線図である。
図４Ａは、本発明の１態様による回路内プログラミングプロセスのフローチャートの第１部分である。
図４Ｂは、本発明の１態様による回路内プログラミングプロセスのフローチャートの第２部分である。
図４Ｃは、本発明の１態様による回路内プログラミングプロセスのフローチャートの第３部分である。
詳細な説明
次の説明は、当業者が本発明を実施できるように記載され、また特定の用途とその要求の関係において記述される。好適な実施例の色々の修正は当業者には明らかであり、ここに定義する一般原則は、発明の精神と範囲から離れずに、他の実施例及び用途に適用することができる。従って、本発明はここに示す実施例に限定することを意図していず、ここに開示した原則と態様と矛盾しないも広い範囲を与えられる。
図１は、本発明の１態様による集積回路の回路内プログラミングのためのシステムのブロック線図である。図１に示す全ての構成要素は、集積回路上にある。マイクロコントローラ100は、マスクＲＯＭモジュール160と、複数回プログラミングモジュール140からの命令を実行する。複数回プログラミングモジュール140は、簡単にされたフラッシュメモリーアレーであり、フラッシュメモリーセルの複数の独立に消去可能なブロックに分割されている。複数回プログラミングモジュール140はまた、マイクロコントローラ100に直接結合した状態ビットとブートベクトル146を備える。複数回プログラミングモジュール140はまた、ＩＣＰ通信ハンドラ142を実行するコードと、マイクロコントローラ100にユーザーが定義した機能を実行させるためのユーザーコード144とを含む。マイクロコーントローラ100は、Ｉ/Ｏポート120、割込みライン122、他の信号124を通って、集積回路外のデバイスへの接続を備える。割込みライン122、他の信号124は、一般にＩ/Ｏポート120を通る通信を協働させ、同期させるため使用される。マイクロコントローラ100はさらに、タイマー105を備え、それがマイクロコントローラ100のタイミング機能を実行する。ウォッチドッグタイマー110がマイクロコントローラ100に結合し、マイクロコントローラ100の動作のデッドロックを検出するため使用される。
マイクロコントローラ100は、命令アドレス126を発行し、それはマスクＲＯＭモジュール160と複数回プログラミングモジュール140の両方のアドレス入力へ送られる。命令アドレス126は、マスクＲＯＭ160と複数回プログラミングモジュール140内の命令に索引を付ける。マスクＲＯＭモジュール160と複数回プログラミングモジュール140からの命令は、これらの命令をマイクロコントローラ100に選択的に切替えるマルチプレクサ（ＭＵＸ）130を通って送られる。さらに、複数回プログラミングモジュール140の一部である状態ビットとブートベクトル146は、マイクロコントローラ100に送られる。
本発明の好適な実施例では、ブートベクトルは、複数回プログラミングモジュール140内

のフラッシュメモリーの一部ではない。そうではなく、複数回プログラミングモジュール140内の別のレジスターであり、特定のアドレスと特定の制御信号が複数回プログラミングモジュール140へ入力されるとき、複数回プログラミングモジュール140から選択的に出力される。

マイクロコントローラ100は、フラッシュ制御レジスター150により、複数回プログラミングモジュール140の動作を制御する。フラッシュ制御レジスター150は、制御レジスター152と、タイミングレジスター154とを含む。マイクロコントローラ100は、フラッシュ制御レジスター150に制御コードを書き込み、複数回プログラミングモジュール140内のコードのため、回路内プログラミングプロセスの消去、プログラム、検証を制御する。本発明の好適な実施例の制御コードを、表１と２に示す。

表１は、複数回プログラミングモジュール140の消去、プログラム、検証機能を制御するため、制御レジスター152へ送られるビットパターンのリスティングを含む。

```
©1996 Macronix International, Co., Ltd.
FMCON: Flash Module Control register
  FMCON[7:4]: Reserved
  FMCON[3:0]: i.e. MS[3:0] of MTPG2 module
    0000:  normal read MTP module at DPTR/PA[15:0] location; this is the default value after reset
    0001:  erase 0-64KB as well as LOCK bits
    0010:  block erase;  erase 0-16K if DPTR [2:0] = 000
                         erase 16-32K if DPTR [2:0] = 010
                         erase 32-48K if DPTR [2:0] = 100
                         erase 48-56K if DPTR [2:0] = 110
                         erase 56-64K if DPTR [2:0] = 111
    0011:  program byte at DPTR/PA [15:0] with data = FMDATA/DQ[7:0]
    0100:  verify erased byte at DPTR/PA [15:0]
    0101:  verify programmed byte at DPTR/PA [15:0]
    0110:  program lock bits,  program LOCK[1] to be 1 if DPTR [1:0] = 00
                               program LOCK[2] to be 1 if DPTR [1:0] = 01
                               program LOCK[3] to be 1 if DPTR [1:0] = 1x
    0111:  verify three programmed lock bits
    1001:  erase status bits as well as boot vector
    1010:  program status bits or boot vector;
                               program SBIT[1:0] if DPTR[0] = 0 with FMDATA/DQ[1:0]
                               program BVEC[7:0] if DPTR[0] = 1 with FMDATA/DQ[7:0]
    1011:  verify programmed status bits or boot vector;
                               verify SBIT[1:0] if DPTR[0] = 0
                               verify BVEC[7:0] if DPTR[0] = 1
    1111:  read Manufacture ID or Device ID
```

TABLE 1

表２は、タイミングレジスター154内の色々のビットにより実行される機能のリスティングを含む。これらのビットを処理することにより、マイクロコントローラ100は、フラッシュメモリーの消去、プログラム、検証、及び読取りオペレーションに要求される波形を生じる。

```
FMTIM:  Flash Module Timing register, used by software to emulate the waveform needed
        for flash operations.
    FMTIM[7]:  VPP Enable bit.
    FMTIM[6]:  Module Enable bit
    FMTIM[5]:  Read Enable bit
    FMTIM[4]:  Write Enable bit
    FMTIM[3:0]:  Reserved
```

TABLE 2

マスクＲＯＭモジュール160は、消去、プログラム、検証ハンドラ162を実行するためのコードを含む。これは、回路内プログラミングに含まれる消去、プログラム、検証動作の順序付けとタイミングを実行するコードを含む。本発明の好適な実施例のこの種類のコードのリスティングを表３に示す。表３は、回路内プログラミングシステムの消去、プログラ

ム、検証機能に含まれるサブルーチンの8051アセンブリコードのリスティングである。

```
;©1996 Macronix International Co., Ltd.
;
; SFR define
        .EQU    FMCON, 0x40H                ; Flash Module CONtrol register
        .EQU    FMTIM, 0x41H                ; Flash Module TIMing register
                                            ; used by s/w to emulate the emulate
                                            ; the waveform needed for flash operations
        .EQU    delay1, 0xFAH               ; Tck = 1/Fosc (ns)
                                            ; delay1 = (256 - 2000/(Tck*12) + 1)
        .EQU    delay2, 0x1AH               ; delay2 = (500000000/ (Tck* 12 * 65536)
; Subroutine: Erase full chip
ERASE:
        MOV     FMCON, #00010010b           ; chip erase
        MOV     FMTIM, #11000000b           ; VPP in and CEB is active
        LCALL   DELAY2us                    ; delay tVPS or tCES (2us)
        MOV     FMTIM, #11010000b           ; WEB is active
        LCALL   DELAY1s                     ; delay tEW (erase time)
        MOV     FMTIM, #11000000b           ; disable WEB
        LCALL   DELAY500ms                  ; delay tER (0.5s)
        MOV     FMTIM, #00000000b           ; clear FMTIM
        MOV     FMCON,#00000000b            ; clear FMCON
        RET
; Subroutine: Verify erased byte
ERASE_VERIFY:
        MOV     FMCON, #00010010b           ; verify chip erase
        MOV     FMTIM, #11000000b           ; VPP in, CEB is active
        MOV     DPTR, #00H                  ; Address
```

```
        LCALL   DELAY2us                ; delay tMS (2us)
        MOV     FMTIM, #11100000b       ; OEB is active
        MOV     R1, #00H
        MOV     R0, #00H
        NOP
        NOP
LOOP_E:
;   MOVX        A,@DPTR                 ; read Flash
        CJNE    A,#0xFFH, ERASE_FAIL
        INC     DPTR
        DJNZ    R0, LOOP_E
        DJNZ    R1, LOOP_E
        MOV     R7, #0xFFH              ; erase verify passed
        SJMP    END_EV
ERASE_FAIL:
        MOV     R7, A                   ; erase verify fail
END_EV:
        MOV     FMTIM, #0000000b        ; clear FMTIM
        MOV     FMCON, #0000000b        ; clear FMCON
        RET

; Subroutine:  Program byte
; DPTR: program address
; A: program data
PROGRAM_B:
        MOV     FMCON, #00010001b       ; program byte
        LCALL   PROGRAM
        MOV     FMCON, #00000000b       ; clear FMCON
        RET

; Subroutine Verify Byte
; DPTR: verify address
; A: verify data
VERIFY_B:
        MOV     FMCON, #00010011b       ; verify byte
        MOV     FMTIM, #110000000b      ; VPP in, CEB active
        MOV     B, A
        LCALL   DELAY2us                ; delay tPV(2us)
        MOV     FMTIM, #11100000b       ; OEB is active
        NOP                             ; delay 4 NOP
        NOP
        NOP
        NOP
        CJNE    A, B, VERIFY_B_FAIL
        MOV     R7, #0xFFH              ; verify passed
        SJMP    END_PV
VERIFY_B_FAIL
        MOV     R7, A                   ; program verify fail
END_PV:
        MOV     FMTIM, #00000000b       ; clear FMTIM
        MOV     FMCON, #00000000b       ; clear FMCON
        RET

; Subroutine: Program LOCK bits
; which lock bit being programmed depends on DPTR [2:0]
PROGRAM_L:
        MOV     FMCON, #00010100b       ; program LOCK bit
        LCALL   PROGRAM
```

```
        MOV     FMCON, #00000000b           ; clear FMCON
        RET

; Subroutine: Verify LOCK bits
VERIFY_L:
        MOV     FMCON, #00010101b           ; verify three programmed lock bits
        LCALL   VERIFY_L_S
        MOV     FMCON, #00000000b           ; clear FMCON
        RET

; Subroutine: Program Status bit
PROGRAM_S:
        MOV     FMCON, #00010110b           ; program State bit
        LCALL   PROGRAM
        MOV     FMCON, #00000000b           ; clear FMCON
        RET

; Subroutine: Verify programmed status bit
VERIFY_S:
        MOV     FMCON, #00010111b           ; verify state bit
        LCALL   VERIFY_L_S
        MOV     FMCON, #00000000b           ; clear FMCON
        RET

; Subroutine: PROGRAM
PROGRAM:
        MOV     FMTIM, #11000000b           ; VPP in, CEB active
        LCALL   DELAY2us                    ; delay tVPS or tCES or tMS (2us)
        MOV     FMTIM, #11010000b           ; WEB is active
        LCALL   DELAY100us                  ; delay tPW(100us)
        MOV     FMTIM, #11000000b           ; disable WEB
        LCALL   DELAY2us                    ; delay tPR (2us)
        MOV     FMTIM, #00000000b           ; clear FMTIM
        RET

; Subroutine: Verify LOCK bits or State bit
VERIFY_L_S:
        MOV     B, A
        MOV     FMTIM, #01000000b           ; VPP = 5V, CEB active
        LCALL   DELAY2us                    ; delay tVPS or tCES or tMS (2us)
        MOV     FMTIM, #11100000b           ; OEB is active
        NOP
        NOP
        NOP
        NOP
        CJNE    A, B, VERIFY_LS_FAIL
        MOV     R7, #0xFFH                  ; verify LOCK or State bit passed
        SJMP    END_VLS
VERIFY_LS_FAIL:
        MOV     R7, A                       ; verify LOCK or state fail
END_VLS:
        MOV     FMTIM, #00000000b           ; clear FMTIM
        RET

; Subroutine: DELAY2us
DELAY2us:
```

```
        PUSH    IE
        CLR     EA
        MOV     TH1, #0xFFH
        MOV     TL1, #delay1
        MOV     TCON, #00H
        MOV     TMOD, #10H                  ; TIMER1, MODE 1
        SETB    TR1                         ; start TIMER1
LOOP1:  JNB     TF1, LOOP1
        MOV     TCON, #00H                  ; clear TF1 and TR1
        POP     IE
        RET

; Subroutine: DELAY100us
DELAY100us
        MOV     R7, #0x32H
LOOP2:  LCALL   DELAY2us
        DJNZ    R7, LOOP2
        RET

; Subroutine: DELAY500ms
DELAY500ms:
        PUSH    IE
        CLR     EA
        MOV     R7, #delay2
        MOV     TCON, #00H
        MOV     TMOD, #10H                  ; TIMER1, MODE 1
LOOP4:  MOV     TH1, #00H
        MOV     TL1, #00H
        SETB    TR1
LOOP3:  JNB     TF1, LOOP3
        MOV     TCON, #00H                  ; clear TF1 and TR1
        DJNZ    R7, LOOP4
        POP     IE
        RET

; Subroutine: DELAY1s
DELAY1s:
        LCALL   DELAY500ms
        LCALL   DELAY500ms
        RET
        .END
```

TABLE 3

本発明の背後にあるキーとなる考えは、回路内プログラミングコードは、２つの部分に分割できることである。回路内プログラミン通信ハンドラ等の通常修正されるコードの部分は、各異なる通信プロトコルのため再構成しなければならず、複数回プログラミングモジュール140内のフラッシュメモリーに記憶される。修正する必要のないＩＣＰに部分、特に複数回プログラミングモジュール140のアーキテクチャに特別に適合させた消去、プログラム、検証機能は、スペース効率の良いマスクＲＯＭモジュール160に記憶される。

図１を参照すると、回路内プログラミングプロセスは、次のように動作する。マイクロコントローラ100が、ＩＣＰ通信ハンドラ142からのコードを実行し、ＩＣＰ通信ハンドラは、Ｉ/Ｏポート120の１つを通って遠隔サイトてＩＣＰイニシエータと通信する。複数回プログラミングモジュール140のユーザーコード部分144内へロードすべき新しい命令は、Ｉ/Ｏポート120の１つを通って、マイクロコントローラ100内へ転送される。マイクロコントローラ100は、マスクＲＯＭモジュール160内の消去/プログラム/検証ハンドラ162内でコードを実行し、複数回プログラミングモジュール140のユーザーコード部分144へ新しいコードをロードする。マイクロコントローラ100は、ウォッチドッグタイマー110に関連して動作し、これは、回路内プログラミングコードのマイクロコントローラ100の実行のデッドロックを検出するために使用される。複数回プログラミングモジュール140内に新し

い命令をプログラムするためには、マイクロコントローラ100は、最初に、制御レジスター152とタイミングレジスター154に書き込まれた命令のシーケンスにより、複数回プログラミングモジュール140の一部を消去する。マイクロコントローラ100は、次に、制御レジスター152とタイミングレジスター154に書き込まれた別の命令により、複数回プログラミングモジュール140に新しいコードをプログラムする。最後に、マイクロコントローラ100は、制御レジスター152とタイミングレジスター154に書き込まれた命令のシーケンスにより、複数回プログラミングモジュール140の新しいコードのプログラミングを検証する。

図２は、マイクロコントローラ100により見られるアドレス空間を示す。アドレス空間は、複数の独立に消去可能なフラッシュメモリーのブロックと、マスクＲＯＭメモリーとに分けられる。ユーザーブロック＃１（260）は、アドレス0からアドレス16Ｋまで延びる。ユーザーブロック＃２（250）は、アドレス16Ｋからアドレス32Ｋまで延びる。ユーザーブロック＃３（240）は、アドレス32Ｋからアドレス48Ｋまで延びる。ユーザーブロック＃４（230）は、アドレス48Ｋからアドレス56Ｋまで延びる。新しいユーザーコード220をダウンロードするための１次ブートコードは、アドレス56Ｋと63Ｋの間にある。この１次ブートコードは、複数回プログラミングモジュール140のユーザー部分144内へ新しい命令をダウンロードするため、システムのブートアップの間に使用される。この１次ブートコードの空間が不十分であれば、２次ブートコードを記憶するため、ユーザーブロック＃４（230）が使用される。

63Ｋと64Ｋの間のアドレス空間は、消去/プログラム/検証のサブルーチン210を備える。アドレス空間200のこの部分は、マスクＲＯＭモジュール160内に位置する。0から63Ｋのアドレス空間200の他の部分は、複数回プログラミングモジュール140のフラッシュメモリーセル内に位置する。アドレス空間200のこれら２つの部分は、異なるメモリーモジュール内にあるが、マイクロコントローラ100のための１つのアドレス空間200の部分である。マスクＲＯＭモジュール160と複数回プログラミングモジュール140からの命令は、ＭＵＸ130を通ってマイクロコントローラ100へ選択的に切替えられる。

図１に示す実施例では、１つのマスクＲＯＭモジュール160と複数回プログラミングモジュール140が示される。他のシステムでは、１つ以上のマスクＲＯＭモジュールと１つ以上の複数回プログラミングモジュールを備え、回路内プログラミング命令の設計と実行により柔軟性がある。

図３は、発明の用途の環境を示す。本発明は、システム300内に配置された集積回路310で実行される。システム300は、プリント回路基板又は他のシステムの器具を備える。集積回路310は、マイクロコントローラ100と、マスクＲＯＭモジュール160と、複数回プログラミングモジュール140と、図１に示すＩＣＰシステムの他の構成要素を備える。マイクロコントローラ100は、複数の集積回路（ＩＣ）312,314,316に結合される。通信チャンネルへのブリッジ320が、チャンネルを与え、そこを通って回路内プログラミングコードが転送される。通信チャンネルへのブリッジ320は、１つのネットワークポートを含んでも良く、又はＩＰＣシステムが非ＩＰＣコードに透過形とするため、別の接合ロジックを含んでも良い。通信チャンネルへのブリッジ320の機能を修正し、通信チャンネルへのブリッジ320が、異なるレベルのデータ速度と、エラー速度と、複雑性とを有する別のＩＣＰ通信チャンネルに結合できるようにすることもできる。例えば１実施例では、通信チャンネル330は、インターネット通信プロトコルを実行するインターネットを備える。

通信チャンネルへのブリッジ320は、通信チャンネル330を通って、パーソナルコンピュータ又はワークステーション等のＩＣＰイニシエータ340に結合される。ＩＣＰイニシエータ340は、通信チャンネル345により、大規模記憶装置350に結合する。ＩＣＰイニシエータ340は、色々の方法でマイクロコントローラ100に結合することができる。１実施例では、ＩＣＰイニシエータ340は、通信チャンネル330をインターネットを通ってアクセスしたワールドワイドウェブサイトである。又は、ＩＣＰイニシエータ340は、ダイヤル呼出しモデムリンクを通るイニシエータとして作用する。更に他の実施例では、通信チャンネル330は、パーソナルコンピュータシステムの通信バスであり、回路内ソフトウェアは、バス330を通ってロードされる。この実施例では、システム300のアップグレードは、フロッ

ピーディスクでエンドユーザーへ分配し、又はＩＣＰイニシエータ340を通って他の方法でロードすることができる。
ある用途では、通信チャンネルへのブリッジ320は必要でない。図１を参照すると、ある用途では、ＩＣＰ通信ハンドラ142内に含まれるソフトウェアは、１つのＩ/Ｏポート120経由で通信チャンネル330を通ってＩＣＰ通信のための適当なプロトコルを実行するのにそれ自体で十分である。このため１つのＩ/Ｏポート120が、直接通信チャンネル330に接続することができ、それにより通信チャンネル320へのブリッジの必要性がなくなる。
図４Ａ、４Ｂ、４Ｃは、回路内プログラミングプロセスに含まれるオペレーションのフローチャートである。図４Ａ、４Ｂ、４Ｃの各々は、４つの列に分けられる。第１列は、「ＩＣＰイニシエータ340と名称を付けられ、図３に示す回路内プログラミングイニシエータ340の活動を表す。ＩＣＰイニシエータ340は、集積回路310の外部のデバイスであり、通信チャンネル330を通って、回路内プログラミングシステム300に接続される。ＩＣＰイニシエータ340は、回路のプログラミングプロセスを開始し、制御する。
図４Ａ、４Ｂ、４Ｃの残りの列は、マスクＲＯＭモジュール160と、複数回プログラミングモジュール140内に記憶されるコードの異なる本体を実行するマイクロコントローラ100のアクションを表す。「ユーザーコード144」と名前を付けた行は、複数回プログラミングモジュール140内のユーザーコード144を実行する間のマイクロコントローラ100の活動を表す。「ＩＣＰ通信ハンドラ142」と名前を付けた行は、複数回プログラミングモジュール140内のＩＣＰ通信ハンドラ142を実行する間のマイクロコントローラ100の活動を表す。「消去/プログラム/検証ハンドラ162」と名前を付けた行は、マスクＲＯＭモジュール160内の消去/プログラム/検証ハンドラ162を実行する間のマイクロコントローラ100のアクションを表す。
図４Ａ、４Ｂ、４Ｃに回路内プログラミングプロセスは、次のように動作する。システムをリセットし、又はウォッチドッグタイマー110がタイムアウトになると、マイクロコントローラ100は、状態430に入りそこで状態ビットがチェックされる。状態ビットがゼロであれば、マイクロコントローラ100は、ユーザーコード144内のステップ420へ入る。状態ビットが１にセットされれば、マイクロコントローラ100は、ブートベクトルにより示される位置へジャンプし、ＩＣＰ通信ハンドラ142内でステップ440を実行する。
ステップ420で、マイクロコントローラ100は、ＩＣＰイニシエータ340から次のコマンドを得るのを待つ。ＩＣＰイニシエータ340がステップ400を実行するとき、マイクロコントローラ100へ更新コマンドを送信する。この更新コマンドは、マイクロコントローラ100に受取られ、ステップ421へ進む。ステップ421で、マイクロコントローラ100は、受取ったコマンドは更新コマンドかどうか尋ねる。そうでなければ、マイクロコントローラ100は、ステップ426へ進み、そこで通常のマイクロコントローラ100のオペレーションが行われ、マイクロコントローラ100は非ＩＣＰユーザーコードを実行する。次にシステムは、次のコマンドを得るため、ステップ420へ戻る。もしコマンドが更新コマンドであれば、マイクロコントローラ100は、ステップ422へ進み、そこで「確認要求」がＩＣＰイニシエータ340へ送られる。ステップ400を実行した後、ＩＣＰイニシエータ340は、ステップ401へ行き、そこでＩＣＰイニシエータ340は、確認要求を待つ。確認要求コマンドが受取られるとき、ＩＣＰイニシエータ340は、ステップ402へ進み、ステップ402へ進み、そこで確認コマンドがマイクロコントローラ100へ送信される。ステップ423で、マイクロコントローラ100は、確認コマンドを受取り、ＩＣＰ通信ハンドラ142内でステップ445へ進み、そこでマイクロコントローラ100は、ＩＣＰイニシエータ340へＩＣＰ準備完了（ready to ICP）コマンドを送信する。確認コマンドを送出後、ＩＣＰイニシエータ340は、ステップ403へ進み、そこでＩＣＰイニシエータ340は、マイクロコントローラ100からのＩＣＰ準備完了コマンドを待つ。
ステップ430で、状態ビットが１にセットされると、マイクロコントローラ100は、１組のステップを実行して、ＩＣＰ通信ハンドラ142内のＩＣＰ要求を確認する。これらのステップは、状態ビットがゼロにセットされたとき、ユーザーコード144内でＩＣＰコードを確認するのに使用されるステップを殆ど正確に反映する。ステップ440で、マイクロコン

トローラ100は、ＩＣＰイニシエータ340からの次のコマンドを待つ。ＩＣＰイニシエータ340がステップ400を実行するとき、マイクロコントローラ100へ更新コマンドを送信する。ステップ440で、この更新コマンドがマイクロコントローラ100に受取られるとき、マイクロコントローラ100は、ステップ441へ進み、そこでマイクロコントローラ100は、コマンドが更新コマンドかどうか求める。更新コマンドでなければ、マイクロコントローラ100は、ステップ442へ進み、そこではオペレーションが起こらない（ＮＯＯＰ）。マイクロコントローラ100は次に、他のコマンドを受取るためステップ440へ戻る。

もし、ステップ441で、マイクロコントローラ100が更新コマンドを受取ると、ステップ443へ行き、そこでマイクロコントローラ100は、確認要求コマンドをＩＣＰイニシエータ340へ送信する。ＩＣＰイニシエータ340がステップ400を実行した後、ステップ401へ行き、そこで確認要求コマンドを待つ。マイクロコントローラ100からのＩＣＰイニシエータ340は、ステップ402へ行き、そこで確認コマンドをマイクロコントローラ100へ送信する。ステップ443を実行した後、マイクロコントローラ100は、ステップ444へ行き、そこでＩＣＰイニシエータ340からの確認コマンドを待つ。確認コマンドを受取ると、マイクロコントローラ100は、ステップ445へ行き、そこでマイクロコントローラ100は、「ＩＣＰ準備完了」コマンドをＩＣＰイニシエータ340へ送信する。ステップ402を実行した後、ＩＣＰイニシエータ340は、ステップ403へ行き、そこでマイクロコントローラ100からのＩＣＰ準備完了コマンドを待つ。

この点で、ＩＣＰコマンドの確認プロセスが完了し、新しいユーザーコードのダウンロードが起こる。ステップ403を実行した後、ＩＣＰイニシエータ340はステップ404へ進み、そこでＩＣＰイニシエータ340は、暗号化した形式の新しいユーザーコードを、通信チャンネルを通ってマイクロコントローラ100へダウンロードする。ステップ446で、マイクロコントローラ100は、新しいユーザーコードを受取り、データ形式を復号化する。ステップ446を実行後、マイクロコントローラ100は、ステップ447へ行き、そこでＩＣＰイニシエータ340へチェックサムを送る。ステップ444を実行した後、ＩＣＰイニシエータ340は、ステップ405へ行き、そこでチェックサムを待つ。チェックサムを受取った後、ＩＣＰイニシエータ340は、ステップ406へ行き、そこでチェックサムがマイクロコントローラ100へ送られたコードのチェックサムと一致するか検証する。チェックサムが一致しなければ、ＩＣＰイニシエータ340は、ステップ408へ行き、それはエンド状態であり、エラーが知らされる。チェックサムが一致すれば、ＩＣＰイニシエータ340は、ステップ407へ行き、そこで進行（go ahead）コマンドを送出する。ステップ447でチェックサムを送った後、マイクロコントローラ100は、ステップ448へ行き、そこで進行コマンドを待つ。

ＩＣＰイニシエータ340からの進行コマンドを受取ると、マイクロコントローラ100は、ステップ449へ行き、そこで回路内プログラミングが開始する。ステップ449で、マイクロコントローラ100は、ブートベクトルを検証して、状態ビットを１にセットし、ＩＰＣオペレーションが行われることを示す。次にマイクロコントローラ100は、状態ビットがセットされたことを検証し、ウォッチドッグタイマー110をセットしスタートする。次にマイクロコントローラ100は、ステップ450へ行き、そこで消去/プログラム/検証ハンドラ162から消去サブルーチンを呼び出す。次にマイクロコントローラ100は、消去/プログラム/検証ハンドラ162のステップ460へ進み、そこでマイクロコントローラ100は、複数回プログラミングモジュール140内の特定のブロックを消去する。次にマイクロコントローラ100は、ＩＣＰ通信ハンドラ142内のステップ451へ進む。

ステップ451で、マイクロコントローラ100は、消去/プログラム/検証ハンドラ162から検証バイトサブルーチンを呼び出す。次にマイクロコントローラは、ステップ461へ進み、そこで検証バイトサブルーチンが実行される。次にマイクロコントローラ100は、ＩＣＰ通信ハンドラ142内のステップ452へ進む。ステップ452で、マイクロコントローラ100は、消去オペレーションの検証が完了したかどうか求める。完了していなければ、マイクロコントローラ100は、次のバイトを検証するため、ステップ451へ戻る。完了していれば、マイクロコントローラ100は、ステップ453へ進み、そこでマイクロコントローラ100は、消去ＯＫコマンドをＩＣＰイニシエータ340へ送信する。ステップ407を実行した後、ＩＣＰ

イニシエータ340は、ステップ409へ進み、そこでＩＣＰイニシエータ340は、消去ＯＫコマンドを待つ。消去ＯＫコマンドを受取った後、ＩＣＰイニシエータ340は、ステップ410へ行き、そこでＩＣＰイニシエータ340は、マイクロコントローラ100からのプログラムＯＫコマンドを待つ。

ステップ453で消去ＯＫコマンドを送出した後、マイクロコントローラ100は、ステップ454へ行き、そこでマイクロコントローラ100は、消去/プログラム/検証ハンドラ162からバイトプログラムサブルーチンを呼び出す。マイクロコントローラ100は次に、ステップ462へ行き、そこでバイトプログラムサブルーチンが実行される。マイクロコントローラ100は次に、ＩＣＰ通信ハンドラ142内のステップ455へ進む。ステップ455で、マイクロコントローラ100は、プログラミングが完了したかどうか求める。完了していなければ、マイクロコントローラ100は、ステップ454へ戻り、プログラムされる次のバイトのためバイトプログラムコマンドを呼び出す。プログラミングが完了していれば、マイクロコントローラ100は次に、ステップ456へ行き、そこで検証バイトサブルーチンが呼び出される。マイクロコントローラ100は次に、ステップ463で、消去/プログラム/検証ハンドラ162内の検証バイトサブルーチンへ進む。ステップ463で、検証バイトサブルーチンが実行される。マイクロコントローラ100は次に、ＩＣＰ通信ハンドラ142内のステップ457へ進む。ステップ457で、マイクロコントローラ100は、検証オペレーションが完了したかどうか求める。完了していなければ、マイクロコントローラ100は、ステップ456へ戻り、そこで検証する次のバイトのため検証バイトサブルーチンが呼び出される。検証オペレーションが完了していれば、マイクロコントローラ100は、ステップ458へ進む。

ステップ458で、回路内プログラミングの消去、プログラム、検証オペレーションが完了する。マイクロコントローラ100は、状態ビットを0にセットし、0にセットされたことを検証する。マイクロコントローラ100は次に、ステップ459へ進み、そこでＩＣＰイニシエータ340へプログラムＯＫコマンドを送出する。ＩＣＰイニシエータ340は、ステップ410で、プログラムＯＫコマンドを受取り、ステップ411へ行き、そこでＩＣＰイニシエータ340は、マイクロコントローラ100へリセットコマンドを送信する。ステップ459で、プログラムＯＫコマンドを送出した後、マイクロコントローラ100は、ユーザーコード144内のステップ424へ進み、そこでリセットコマンドを待つ。ＩＣＰイニシエータ340からリセットコマンドを受取ると、マイクロコントローラ100は、エンド状態であるステップ425へ進む。ステップ411でリセットコマンドを送出した後、ＩＣＰイニシエータ340は、同じくエンド状態のステップ412へ進む。この点で、回路内プログラミングプロセスは完了する。ＩＣＰイニシエータ340により新しい回路内プログラミングプロセスが開始されるとき、このプロセスは繰り返される。

図１に示すアーキテクチャを使用して、システムの設計者は、回路内プログラミングコードを特定の環境に適合させることができる。従って、図１に示すように、製造者は、彼らの回路で実行するため集積回路を選択する。回路内プログラミングコードが実行可能でなければ、マイクロコントローラ100が使用され、チップ300上の色々の通信ポートを使用して、システムを特定の回路内プログラミング環境に合わせるために、回路ボード上に必要な特別のロジックを最小限にする。第１に、回路内プログラミングのための適当な接続とプロトコルが、設計者により選択される。次に、選択した環境のためのＩＣＰコードが作られ、改良される。ＩＣＰコードは次に、システムの通常のオペレーションの間に実行されるプログラムと一体とされる。次に、一体にしたＩＣＰコードとユーザーコードは、複数回プログラミングモジュール140のフラッシュメモリー内に記憶される。次に、消去、プログラムオペレーションが検証される。次に、一体のＩＣＰコードを含むマイクロコントローラ100が、システム内に配置される。システムがよく作用すれば、システムが大量に製造される。もしＩＣＰコードを修正する必要があれば、ＩＣＰコードを最適化するため、ＩＣＰプロセスを繰り返す。同様に、同じプログラミング技術を使用して、システムコードが最適化される。従って、システムのエンドユーザーは、マイクロコントローラ100に組込まれた頑強な回路内プログラミングコードを得て、本発明の対話形回路内プログラミング技術を使用して、それを進行中に更新し修正することができる。

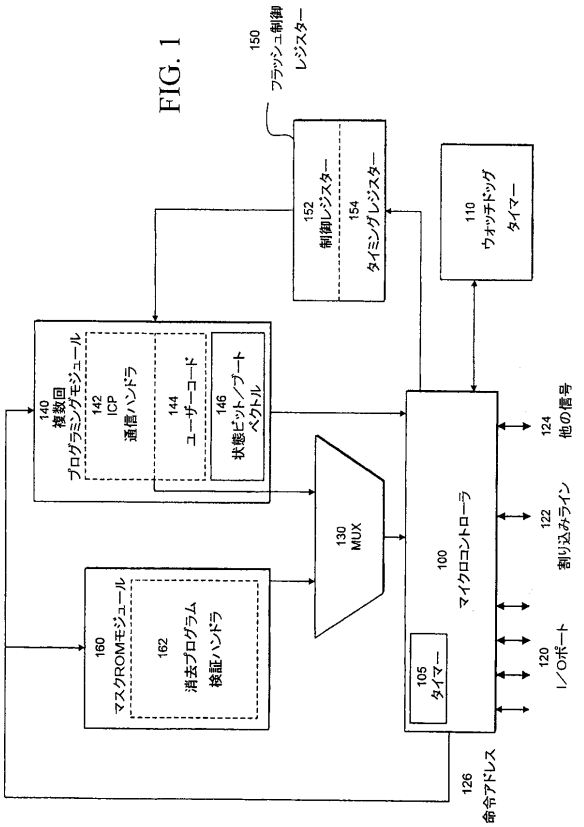
結論

従って、本発明は、多くの回路内プログラミングの用途に合う柔軟性のあるフラッシュメモリーベースのマイクロコントローラのアーキテクチャを与える。例えば、テレビ、又はビデオモニター、デジタルビデオディスク、又はＣＤ－ＲＯＭ、遠隔制御デバイス、又は携帯電話は、本発明による回路内プログラミング構造を有するマイクロコントローラを備えることができる。次に本発明の柔軟性のあるアーキテクチャを素養して、更新されたＩＣＰコードの色々のソースを、個々のデバイスにロードすることができる。それゆえ、本発明は、特定のようとの環境に修正し、又は適合させることができる。回路内プログラミング構造をサポートするのにグルーロジックは非常に少なくて良いか又は全く必要ない。さらに、回路内プログラミングと組み合わさるマイクロコントローラのパワーは、回路内プログラミングシステムのフラッシュメモリーの設計を簡単にするため、変更することができる。

ＩＣＰコードの集積回路に独自の部分は、変化しないと考えられるので、スペース効率の良いマスクＲＯＭに記憶することにより、集積回路上のシリコンの面積を節約することができる。通信ハンドラ等のＩＣＰコードの他の部分は、頻繁に修正されると考えられるので、フラッシュメモリーに保持される。このように、シリコンの面積を節約し、広い範囲の異なる用途に回路内プログラミングシステムを適合させる柔軟性を維持することができる。

本発明の好適な実施例のこの記述は、例示と説明の目的である。本発明をここに開示した正確な形がすべてではなく、これらに限定する意図ではない。明らかに、多くの修正と改変が当業者には明らかであろう。本発明の範囲は、請求の範囲とその均等範囲より限定される。

【図１】



【図２】

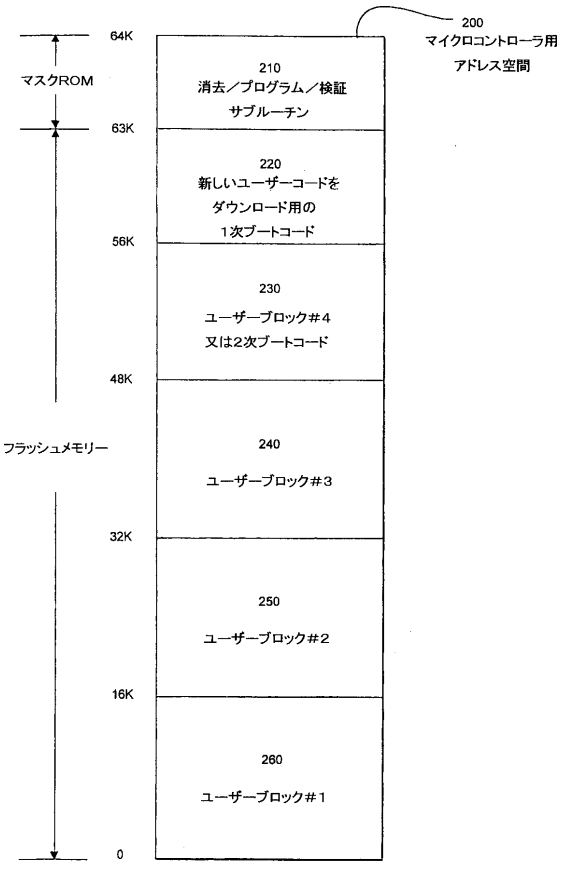


FIG. 2

【図３】

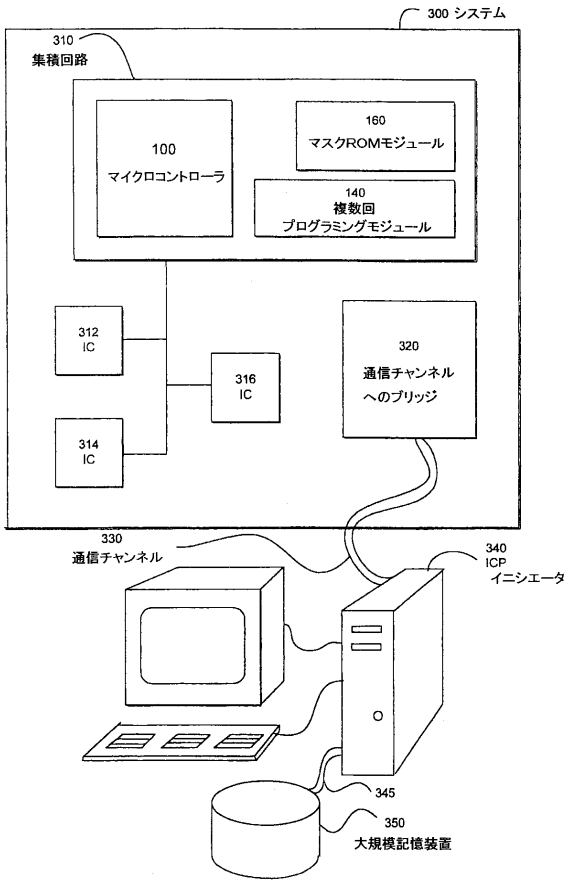


FIG. 3

【図４Ａ】

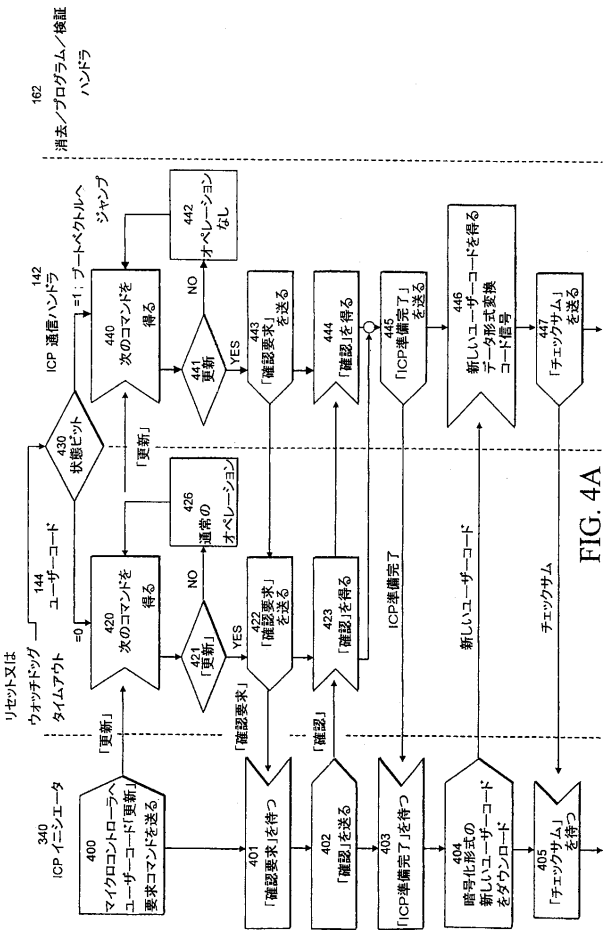


FIG. 4A

【図４Ｂ】

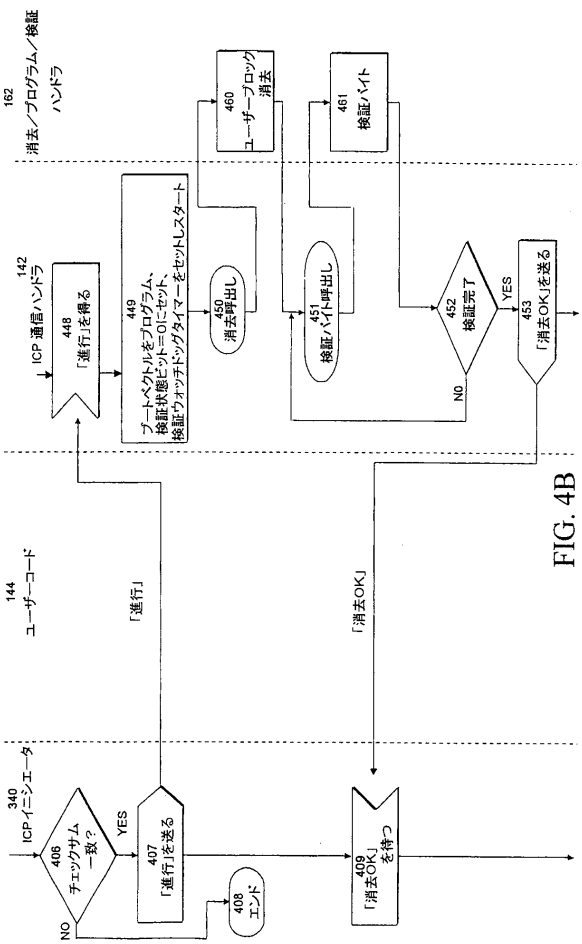


FIG. 4B

【図４Ｃ】

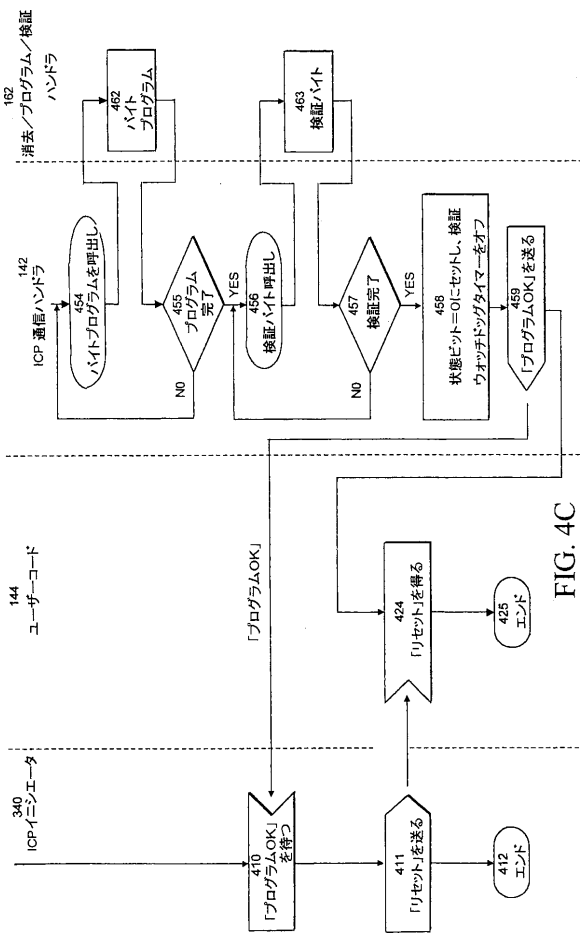


FIG. 4C

フロントページの続き

(74)代理人
弁理士 竹内 英人
(74)代理人
弁理士 今城 俊夫
(74)代理人
弁理士 小川 信夫
(74)代理人
弁理士 村社 厚夫
(74)代理人
弁理士 西島 孝喜
(74)代理人
弁理士 箱田 篤
(72)発明者 サン アルバート シー
台湾 タイペイ ネイヒュ ネイヒュ ロード セクション １ レーン ２８５ アレイ ６９ ナンバー４０ ４エフ
(72)発明者 リー チェー ホルン
台湾 タイペイ サン チュン チュン チェン ノース ロード ナンバー２９３ 2エフ
(72)発明者 チェン チャン ルン
台湾 シンチュ ドン ナルン ストリート レーン ２１０ アレイ １８ ナンバー５６ ６エフ

審査官 久保 光宏

(56)参考文献 特開昭６２－１１９６５３（ＪＰ，Ａ）
特開平５－２６６２１９（ＪＰ，Ａ）
特開昭６３－２６６６９８（ＪＰ，Ａ）
特開昭６３－２０６８５２（ＪＰ，Ａ）
特開平４－２７６８３８（ＪＰ，Ａ）
特開平７－６４７７０（ＪＰ，Ａ）
特開平６－４５９９８（ＪＰ，Ａ）
特開平１０－２２２３６２（ＪＰ，Ａ）
特開平１０－５００８６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 11/00
G06F 9/06
G06F 9/445
G06F 13/00