

(19) 日本国特許庁(JP)

## (12) 特許公報(B2)

(11) 特許番号

特許第4424760号  
(P4424760)

(45) 発行日 平成22年3月3日(2010.3.3)

(24) 登録日 平成21年12月18日(2009.12.18)

(51) Int.Cl.

F 1

G06F 11/26 (2006.01)

G06F 11/26

G06F 11/22 (2006.01)

G06F 11/22 320B

G06F 17/50 (2006.01)

G06F 17/50 664P

請求項の数 17 (全 51 頁)

(21) 出願番号

特願平11-500747

(86) (22) 出願日

平成10年5月18日(1998.5.18)

(65) 公表番号

特表2002-507294(P2002-507294A)

(43) 公表日

平成14年3月5日(2002.3.5)

(86) 国際出願番号

PCT/US1998/010171

(87) 国際公開番号

W01998/054664

(87) 国際公開日

平成10年12月3日(1998.12.3)

審査請求日

平成17年5月13日(2005.5.13)

(31) 優先権主張番号

08/865,741

(32) 優先日

平成9年5月30日(1997.5.30)

(33) 優先権主張国

米国(US)

(73) 特許権者

クリックターン・デザイン・システムズ・  
インコーポレイテッド  
アメリカ合衆国 95131 カリフォルニア州、サン・ジョゼ、ウエスト・トリニティ・ロード 55

(74) 代理人

弁理士 岡田 英彦

(74) 代理人

弁理士 池田 敏行

(74) 代理人

弁理士 岩田 哲幸

(74) 代理人

弁理士 中村 敦子

最終頁に続く

(54) 【発明の名称】時間多重相互接続を用いたエミュレーション・システム

## (57) 【特許請求の範囲】

## 【請求項 1】

デジタル・ロジックにより構成されるユーザ設計を実現するために構成される電気的に再構成可能なハードウェア・エミュレーション・システムであって、前記ユーザ設計は少なくとも1つのユーザ・クロック信号を利用するものであり、

ロジック・ボードを備え、

前記少なくとも1つのユーザ・クロック信号の各々と非同期の多重クロック信号を生成する多重クロック信号ソースを備え、

出力マルチプレクサ用回路及び入力デマルチプレクサ用回路を有する回路に電気的に接続された複数の入力／出力ピンを有する複数のプログラム可能なロジック装置を備え、前記出力マルチプレクサ用回路及び入力デマルチプレクサ用回路は前記多重クロック信号によって制御されるものであり、

出力マルチプレクサ用回路及び入力デマルチプレクサ用回路を有する回路に電気的に接続された複数の入力／出力ピンを有する複数のプログラム可能な相互接続装置を備え、前記出力マルチプレクサ用回路及び入力デマルチプレクサ用回路は前記多重クロック信号によって制御され、プログラム可能なロジック装置は複数のプログラム可能な相互接続装置の少なくとも1つのプログラム可能な相互接続装置を使用して相互接続されるものであり、前記ロジック・ボードは、パーシャル・クロスバー・インターフェクションの配置において、前記プログラム可能なロジック装置の入力／出力ピンを前記プログラム可能な相互接続装置の前記入力／出力ピンに接続する1組の固定された電気導体をさらに備え、前記導

体は、前記プログラム可能なロジック装置の前記入力／出力ピンと前記プログラム可能な相互接続装置の前記入力／出力ピンとの間に時間多重化信号を伝達する、ことを特徴とするエミュレーション・システム。

【請求項 2】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記再プログラム可能なロジック装置はプログラマブル・ゲート・アレイにより構成されていることを特徴とするエミュレーション・システム。

【請求項 3】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記再プログラム可能なロジック装置はフィールド・プログラマブル・ゲート・アレイにより構成されていることを特徴とするエミュレーション・システム。

10

【請求項 4】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記再プログラム可能な相互接続装置はプログラマブル・ゲート・アレイにより構成されていることを特徴とするエミュレーション・システム。

【請求項 5】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記再プログラム可能な相互接続装置はクロスポイント・スイッチのアレイが形成されている集積回路により構成されていることを特徴とするエミュレーション・システム。  
。

20

【請求項 6】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記複数のプログラム可能なロジック装置の前記入力デマルチプレクサは少なくとも2つのシーケンシャル・ロジック・エレメントにより構成されていることを特徴とするエミュレーション・システム。

【請求項 7】

請求項 6 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記少なくとも2つのシーケンシャル・ロジック・エレメントはフリップ・フロップにより構成されていることを特徴とするエミュレーション・システム。

【請求項 8】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記複数のプログラム可能なロジック装置の前記出力デマルチプレクサは少なくとも2つのシーケンシャル・ロジック・エレメントにより構成されており、前記少なくとも2つのシーケンシャル・ロジック・エレメントの各々のそれぞれの出力はマルチプレクサに入力されることを特徴とするエミュレーション・システム。

30

【請求項 9】

請求項 8 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記少なくとも2つのシーケンシャル・ロジック・エレメントはフリップ・フロップにより構成されており、前記フリップ・フロップの各々はデータ入力、イネーブル入力、クロック入力及び出力を有することを特徴とするエミュレーション・システム。

40

【請求項 10】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記複数の前記再プログラム可能な相互接続装置の前記入力デマルチプレクサは少なくとも2つのシーケンシャル・ロジック・エレメントにより構成されていることを特徴とするエミュレーション・システム。

【請求項 11】

請求項 10 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであつて、前記少なくとも2つのロジックエレメントはフリップ・フロップにより構成されていることを特徴とするエミュレーション・システム。

【請求項 12】

請求項 1 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであって、前記少なくとも 2 つのシーケンシャル・ロジック・エレメントは第二のクロック分周器の出力によってクロックされ、前記第二のクロック分周器は前記多重クロック信号を受信することを特徴とするエミュレーション・システム。

【請求項 1 3】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであって、前記複数のプログラム可能なロジック装置の前記出力マルチプレクサは第一のマルチプレクサにより構成されており、前記第一のマルチプレクサは少なくとも選択入力及び 2 つの入力から構成されていることを特徴とするエミュレーション・システム。

【請求項 1 4】

請求項 1 3 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであって、前記第一のマルチプレクサの前記選択入力は第一のクロック分周器の出力を受信するものであり、前記第一のクロック分周器は前記多重クロック信号を受信することを特徴とするエミュレーション・システム。

【請求項 1 5】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであって、前記出力マルチプレクサ用回路及び前記入力デマルチプレクサ用回路は、前記複数のプログラム可能なロジック装置の各々の前記入力 / 出力ピンと電気的に通信を行ってプログラム可能な回路にプログラムされることを特徴とするエミュレーション・システム。

【請求項 1 6】

請求項 1 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであって、前記出力マルチプレクサ用回路及び前記入力デマルチプレクサ用回路は、前記複数のプログラム可能なロジック装置の各々に形成されているとともに、前記複数のプログラム可能なロジック装置の前記入力 / 出力ピンと電気的に通信を行うことを特徴とするエミュレーション・システム。

【請求項 1 7】

請求項 7 に記載の電気的に再構成可能なハードウェア・エミュレーション・システムであって、前記少なくとも 2 つのシーケンシャル・ロジック・エレメントは第一のクロック分周器の出力によってイネーブルされるものであり、前記第一のクロック分周器は前記多重クロック信号を受信することを特徴とするエミュレーション・システム。

【発明の詳細な説明】

発明の技術分野

本発明は、一般的には電子回路設計を検証するための装置に関し、より詳細には複数の設計信号がプログラム可能なロジック・チップ (logic chip) 間で 1 つの物理的な結線により伝送されるハードウェア・エミュレーション・システム (hardware emulation system) に関する。

発明の背景技術

ハードウェア・エミュレーション・システムは、チップ又はプリント回路基板の製造に先立って電子回路設計を検証するために設計された装置である。これらのシステムは、一般的には、プログラム可能なロジック・チップ (ロジック・チップ) 及びプログラム可能な相互接続チップ (相互接続チップ) (interconnectchip) によって構築される。ここで、「チップ」という用語は、集積回路をいうものとする。ロジック・チップの例としては、フィールド・プログラマブル・ゲート・アレイ (field-programmable gate arrays) ("FPGAs") のような再プログラム可能な論理回路 (logic circuit) を含み、既製品と注文品が含まれる。相互接続チップの例としては、再プログラム可能な F P G A、マルチプレクサ・チップ (multiplexer chip)、クロスポイント・スイッチ・チップ (cross point switch chip) 等を含む。相互接続チップは、既製品又は注文設計された製品のいずれであってもよい。

従来のエミュレーション・システムは、一般的に、エミュレーションされる 1 つの電子回路設計内の各信号が 1 つのロジック・チップ内の一又は複数の物理的な金属配線 ("ワ

10

20

30

40

50

イヤ")にマップされるように設計されていた。ロジック・チップ間でやりとりされる信号は、1つのロジック・チップ上の1つ又は複数の物理的ピンと、ロジック・チップ及び相互接続チップを含むプリント回路基板上の1つ若しくは複数の物理的な経路にマップされていた。

従来のエミュレーション・システムにおける設計信号の物理的なピン及び経路に対する1対1のマッピングのため、エミュレーション・システムは、少なくともロジック・チップの間を接続する設計信号と同じ数のロジック・チップのピン及びプリント回路基板の経路が必要である。このような構成によれば、エミュレーション・システムを構成するために、非常に複雑で高価な集積回路のパッケージ、プリント回路基板及び回路基板コネクタが必要であった。これらの部品は高価であるために、ハードウェア・ロジック・エミュレーション・システムの価格も順次高価なものとなり、設計者の数が限定され、それにより、システムからの恩恵が限定され、ハードウェア・エミュレーション・システムの優位性が限定される要因となっていた。10

さらに、集積回路の製造技術の進展により、急速に小型化されたサイズのものが使用できるようになった。このため、ロジック・チップのロジック密度(すなわち、ロジック・チップの中に設けることができるロジック・ゲートの数)は劇的に増加した。しかしながら、1つのロジック・チップに設けられ又はエミュレートされるロジック・ゲートの数の増加は、チップのパッケージ(package)の入力、出力、クロック等に利用可能なピン(すなわち、リード線)の数の増加に結びつかない。集積回路パッケージのピンの数は、チップの利用可能な周辺の長さにより限定される。さらに、集積回路のダイス(dice)上の結合パッドをパッケージ上のピンに接続するために用いられるワイヤ・ボンディング組み立て装置の能力は、長い時間をかけてゆっくりと向上している。このため、1つのロジック・チップ上で利用可能なロジックの数と、これらのロジックを外部に接続するために利用可能なピンの数との間の食い違いは、ますます大きなものとなっている。このため、ロジック・チップの論理的な容量(logical capacity)の平均的な利用率は低く、任意の大きさの電子回路設計のエミュレーションに必要とされるハードウェア・エミュレーション・システムのコストを押し上げる結果となっている。20

時間多重(time-multiplexing)は、各物理的な接続のコストが非常に高いあるタイプのシステムにおいて、複数の論理信号の間で1つの物理的ワイヤ又はピンを共有するために用いられてきた技術である。このようなシステムには、遠距離通信システムが含まれる。しかしながら、従来の時間多重技術方法ではエミュレートされる回路の動作速度が大きく低下するため、時間多重は、Quickturn Design Systems, Inc.、Mentor Graphics Corporation、Aptix Corporationその他により提供されるハードウェア・エミュレーション・システムには一般的には用いられていなかった。さらに、従来の時間多重技術では、ハードウェア・エミュレーション・システムにおいて、組み込みまれた設計の非同期の動作を正しく保持することが困難である。30

前述したように、ハードウェア・エミュレーション・システムの1つの機能は、集積回路の機能を検証することにある。典型的には、回路の設計者又は技術者が集積回路を設計する時、その設計は、設計の"ネットリスト"記述の形で表わされる。ネットリスト記述(又は当業者によって呼ばれている"ネットリスト")は、集積回路の部品及びその部品の間の電気的な相互接続を記述したものである。これらの部品には、ロジック回路を供給するために必要なすべての回路要素(例えば、ゲート)及びシーケンシャル・ロジック(sequential logic)(例えば、フリップ・フロップ及びラッチ)が含まれる。従来のエミュレーション・システムでは、ネットリストをハードウェア・エミュレーション・システムに供給する前に、ユーザーの回路のネットリストの解析を行っていた。この解析には、その設計のさまざまな回路上の経路をクロック経路、クロック・クオリファイア(clock qualifier)及びデータ経路に分離するステップが含まれていた。この解析と分離を行う方法は、Chen等による米国特許No. 5,475,830号に開示されている。この特許は、本発明の譲受人と同一の譲受人に譲渡されている。ここで、参考のために、米国特許No. 5,475,830号の開示内容を完全な形で組み込む。米国特許No. 5,440  
50

75, 830号に開示されている技術は、Quickturn Design Systems, Inc., Mountain View, CaliforniaのSystem Realizer<sup>TM</sup>ブランドのハードウェア・エミュレーション・システムのような従来のエミュレーション・システムにおいて用いられてきた。しかしながら、そこに開示されている技術は、いかなる種類の時間多重とも組み合わせられて使用されていない。

Virtual Machine Works(現在のIKOS)、ARKOS(現在のSynopsis)及びIBMから提供されている従来のハードウェア・エミュレーション・システムでは、エミュレーションされる任意のサイズの電子的な設計についてハードウェアのコストの低減を図るべく、設計信号を単一の物理的なロジック・チップのピン及びプリント回路基板に時間多重する試みがなされていた。しかしながら、これらの従来のエミュレーション・システムでは、正確な回路の動作を維持するためにクロック経路を変更(alter)し又は再合成(re-synthesize)する。これらの変更又は再合成のプロセスは、同期的な設計については予測可能な動作を行う。しかしながら、非同期的な設計では、不正確又は誤ったエミュレーション結果をもたらす可能性がある。多くの回路設計では非同期クロック構造が採用されているため、クロック経路の変更又は再合成の必要性は大きな欠点となっている。10

これに加えて、時間多重を使用している従来のハードウェア・エミュレーション機器は、低い動作速度に悩んでいた。これは、クロック経路の再合成に起因する。これらの機器では、設計について1つのクロック・サイクルをエミュレートするために複数の内部マシン・サイクルが必要となる。このため、エミュレーションされる設計の実効的な動作速度は、通常は、エミュレーション・システム自体の最大クロック・レートよりもかなり遅いものとなる。エミュレーションされる設計に複数の非同期のクロックが含まれると、各組の入力クロックのエッジの間でエミュレートされる設計の状態を評価する必要が生じるため、通常、動作速度の低下はさらに深刻になる。20

時間多重を用いた従来のハードウェア・エミュレーション・システムでは、単一の物理的なロジック・チップ・ピン又はプリント回路基板の配線(trace)における多くの設計信号の流れを同期させるための複雑なソフトウェアが必要である。各設計信号は、システムの他の部分において、他の設計信号を計算するためにそれが必要とされる時点において正確な値を有するように時間的に管理される必要がある。このタイミング解析のソフトウェア(スケジューリング・ソフトウェアとも呼ばれている)は、エミュレータの複雑性を増加させ、回路設計をエミュレータの中に編集(compile)するのに必要な時間を増加させる。30

さらに、時間多重を用いた従来のハードウェア・エミュレーション機器は、最小限のハードウェアを必要とする簡単な形式の時間多重を使用するのみでよいが、大きな消費電力(例えば、電流)及び複雑なシステム設計が必要である。

以上の理由により、論理的容量が非常に大きく、編集時間が短く、複雑なソフトウェアではなく、簡略化された機械設計及び消費電力が小さいハードウェア・エミュレーション・システムが求められている。

### 発明の概要

複数の設計信号を複数の物理的なロジック・チップのピン及び複数のプリント回路基板の配線に時間多重化することによってハードウェアのコストを削減し、低い動作速度や低い非同期動作性等の制限がない新しい種類のハードウェア・エミュレーション・システムを開示し、特許請求の範囲に記載する。また、ハードウェア・エミュレーションに適しているが、大商品電力や複雑なシステム設計等の欠点のない複数の信号を単一の物理的相互接続に時間多重化する追加の方法も開示する。40

好ましい実施例では、時間多重化は、設計中のクロック・クロリファイア経路(クロック・クロリファイアとは、クロック信号をゲートするために使用されるすべての信号をいう)及びデータ経路で実行されるが、クロック経路(クロック経路とは、クロック信号とクロック信号を生成するクロック資源との間の経路をいう)では実行されない。

本発明の再構成可能(reconfigurable)なロジック・システムは、複数の再構成可能なロジック・デバイス(logic device)によって構成されている。各ロジック・デバイスは50

、少なくとも組み合わせロジック・エレメント (logic element) 及び格納エレメント (storage element) を提供するために再プログラム可能に構成することができる内部回路を有している。また、このプログラム可能なロジック・デバイスは、ロジック・デバイスの機能エレメント (functional element) の中から選択された 1 つの機能エレメントに再プログラム可能に相互接続することができる入力 / 出力端子を有している。再プログラム可能なロジック・デバイスは、各入力 / 出力端子に設けられた入力デマルチプレクサ (input demultiplexer) 及び出力マルチプレクサ (output multiplexer) を有している。入力デマルチプレクサは、時間多重化信号を受信して、これを 1 つ又は複数の内部信号に分割する。出力マルチプレクサは、1 又は複数の内部信号を単一の物理的な相互接続に結合する。

10

また、本発明は、複数の再プログラム可能な相互接続デバイスによって構成されている。各相互接続デバイスは、入力 / 出力端子と、選択された入力 / 出力端子の間に相互接続を提供するように再プログラム可能に構成することができる内部回路を有している。また、再プログラム可能な相互接続デバイスは、各入力 / 出力端子に設けられた入力デマルチプレクサ及び出力マルチプレクサを有している。入力デマルチプレクサは、時間多重化された入力信号を受信して、これを 1 つ又は複数のコンポーネント信号に分割する。出力マルチプレクサは、1 又は複数のコンポーネント信号を第二の単一の物理的な相互接続として結合する。

また、本発明は、再プログラム可能なロジック・デバイス上のプログラム可能な入力 / 出力端子を、再プログラム可能な相互接続デバイス上の入力 / 出力端子に接続する 1 組の固定された電気的導体を含む。

20

本発明の他の観点では、ロジック・アナライザが、エミュレーションを受けている設計の完全な視認性を提供するロジック・エミュレーション・システムに統合されている。本発明のロジック・アナライザは、その構成部分がエミュレーション・システムの多くの資源に統合される形で分配される。本発明のロジック・アナライザは、少なくとも、ロジック・ボードの各ロジック・チップにプログラムされるスキャン・チェーン (scan chains) を有している。スキャン・チェーンは、少なくとも 1 つのフリップ・フロップにより構成される。スキャン・チェーンは、エミュレーションを受けている設計のシーケンシャル・ロジック・エレメントの選択されたサブセットにプログラム可能な形で接続することができる。

30

ロジック・アナライザは、さらに、スキャン・チェーンとの通信を行うメモリ・デバイスを含む。このメモリ・デバイスは、エミュレーションを受けているロジック設計のシーケンシャル・ロジック・エレメントからのデータを格納する。制御回路は、エミュレーション・システムのロジック・チップと通信して、スキャン・チェーンのクロックとなるロジック・アナライザ・クロック信号を生成する。制御回路は、また、ロジック・チップの中で予め定められた信号の組み合わせが発生したときに、トリガ信号を生成する。

エレメントの供給及び組み合わせの種々の新規な詳細を含む、本発明の上記の及び他の好みしい特徴は、添付する図面を参照してより詳しく記載されており、また請求の範囲に指摘されている。発明の実施する特定の方法及び回路は、説明のために示されており、発明の範囲を制限するものではないと理解すべきである。当業者に理解されるように、本発明の原理及び特徴は、本発明の趣旨を逸脱することなく、多くの様々な、多くの実施例において用いることができる。

40

#### 【図面の簡単な説明】

本発明のさまざまな側面を解説するための実施例を示す添付図面を参照することにより、新規な特徴及び優位性が明らかとなる。

図 1 は、時間多重を組み込んだ部分的なクロスバー・ネットワークを示すブロック図である。

図 2 は、2 対 1 の時間多重における信号の関係を示すタイミング図である。

図 3 は、2 対 1 の時間多重を行うために F P G A 内に必要な回路を示すブロック図である。

50

図4は、多重を行うチップ内の等価回路を示すブロック図である。

図5は、4対1の時間多重を行うために必要な信号の関係を示すタイミング図である。

図6は、4対1の時間多重を行うためにFPGA内に必要なロジックを示すブロック図である。

図7は、多重を行うチップ内の等価回路を示すブロック図である。

図8は、ハードウェア・エミュレーション・システムに好適なパルス幅エンコーディング処理のための信号の関係を示すタイミング図である。

図9は、ハードウェア・エミュレーション・システムに好適な位相エンコーディング処理のための信号の関係を示すタイミング図である。

図10は、ハードウェア・エミュレーション・システムに好適なシリアル・データ・エンコーディング処理のための信号の関係を示すタイミング図である。 10

図11は、本発明の好ましい実施例のロジック・ボードのブロック図である。

図12は、本発明の好ましい実施例の種々の回路基板の間の相互接続を示すブロック図である。

図13は、本発明の好ましい実施例の物理的な構成を示す図である。

図14は、より少ないロジック容量を有する現時点での好ましいバージョンのエミュレーション・システムの種々の回路基板の間の相互接続のブロック図である。

図15は、1つのロジック・ボードと1つのI/Oボードを有する図14に記載のエミュレーション・システムの物理的な構成を示す図である。

図16は、I/Oボード及びコア・ボード(core board)のブロック図である。 20

図17は、マルチプレクサ・ボード(mux board)のブロック図である。

図18は、拡張可能なマルチプレクサ・ボードのブロック図である。

図19は、本発明の好ましいハードウェア・エミュレーション・システムにおけるユーザー・クロックの分配状態を示すブロック図である。

図20は、本発明の好ましいハードウェア・エミュレーション・システムの制御構造を示すブロック図である。

図20aは、本発明の好ましい実施例のロジック・アナライザのブロック図である。

図20bは、本発明の好ましい実施例のロジック・アナライザ信号のデータ経路を示すブロック図である。

図20cは、本発明の好ましい実施例におけるロジック・アナライザのイベント(event)のロジック・チップ内での分配状態を示すブロック図である。 30

図20dは、格納エレメント及び外部入力値からプローブ信号(probed signal)を計算する状態を示すロジック図である。

図21は、本発明の好ましいハードウェア・エミュレーション・システムをプログラムする状態を示すフロー・チャート図である。

図22は、本発明の好ましい実施例に対応する動作テストベンチ・コンパイラー(behavioral testbench compiler)により作成されたソフトウェア・ハードウェア・モデルの編集に必要なステップの順序を示すフロー図である。

図22aは、本発明の好ましい実施例のLCMメモリ・ジェネレータ(memory generator)によって生成され得るメモリ回路の例を示すブロック図である。 40

図23は、本発明の好ましい実施例に対応するマイクロプロセッサのイベント同期バス(microprocessor event synchronization)の共同シミュレーション・ロジック(co-simulation logic)の特別な接続を表現するために作成されたネットリスト構造のブロック図である。

図24aは、本発明の好ましい実施例におけるロジック・チップのI/Oピンの種別に応じて挿入される時分割多重セル(time-division-multiplexing cell)の概略図である。

図24bは、本発明の好ましい実施例におけるロジック・チップのI/Oピンの種別に応じて挿入される時分割多重セルの概略図である。

図24cは、本発明の好ましい実施例におけるロジック・チップのI/Oピンの種別に応じて挿入される時分割多重セルの概略図である。 50

図 24 d は、本発明の好ましい実施例におけるロジック・チップの I/O ピンの種別に応じて挿入される時分割多重セルの概略図である。

図 24 e は、本発明の好ましい実施例におけるロジック・チップの I/O ピンの種別に応じて挿入される時分割多重セルの概略図である。

図 24 f は、本発明の好ましい実施例におけるロジック・チップの I/O ピンの種別に応じて挿入される時分割多重セルの概略図である。

図 24 g は、本発明の好ましい実施例におけるロジック・チップの I/O ピンの種別に応じて挿入される時分割多重セルの概略図である。

図 24 h は、本発明の好ましい実施例におけるロジック・チップの I/O ピンの種別に応じて挿入される時分割多重セルの概略図である。 10

図 24 i は、本発明の好ましい実施例におけるロジック・チップの I/O ピンの種別に応じて挿入される時分割多重セルの概略図である。

図 24 j は、本発明の好ましい実施例におけるロジック・チップの I/O ピンの種別に応じて挿入される時分割多重セルの概略図である。

図 24 k は、本発明の好ましい実施例におけるロジック・チップの I/O ピンの種別に応じて挿入される時分割多重セルの概略図である。

図 25 は、本発明の好ましい実施例のイベント検出セル (event detection cell) のブロック図である。

図 26 は、本発明の一実施例に対応する特別なイベント多重セル (special event-multiplexing cell) を用いて 1 組ごとに時間多重化される AND ツリーの出力状態を示す概略図である。 20

図 27 は、本発明の好ましい実施例のイベント検出ダウンロード回路 (event detector download) のブロック図である。

#### 画面の詳細な説明

以下に、画面を参照しながら、現時点において好ましい本発明の装置及び方法を説明する。

図 1 は、本発明のハードウェア・エミュレーション・システムの好ましい実施例のパーシャル・クロスバー・インターフェクション (partial crossbar interconnect) の一部を示したものである。パーシャル・クロスバー・インターフェクションのアーキテクチャの実施例は、本発明の譲受人と同一の譲受人に譲渡された Butts 等による米国特許 N o . 5 , 0 3 6 , 4 7 3 号、5 , 4 4 8 , 4 9 6 号、5 , 4 5 2 , 2 3 1 号に開示されている。米国特許 N o . 5 , 0 3 6 , 4 7 3 号、5 , 4 4 8 , 4 9 6 号、5 , 4 5 2 , 2 3 1 号の開示内容を完全な形での参照により組み込む。パーシャル・クロスバー・インターフェクションでは、各ロジック・チップの入力 / 出力ピンは、各ロジック・チップで同一の分割手法を用いて適当なサブセットに分割される。各マルチプレクサ・チップ (Mux chip) (クロスバー・チップとも呼ばれる) は、各ロジック・チップの同一のサブセットに接続される。このためクロスバー・チップ 'n' は、各ロジック・チップのピンのサブセット 'n' に接続される。サブセットと同じ数のクロスバー・チップが用いられ、各クロスバー・チップは、サブセットのピンの数にロジック・チップの数を乗算した数のピンを有する。各ロジック・チップ / クロスバー・チップの組は、各サブセットのピンと同じ数の、経路と呼ばれるワイヤによって相互接続される。 30

図 1 のパーシャル・クロスバー・インターフェクションは、多くの再プログラム可能な相互接続ブロック 12、好ましい実施例ではマルチプレクサ・チップ (Mux chip) により構成される。図 1 のパーシャル・クロスバー・インターフェクションは、さらに、複数の再プログラム可能な構成ロジック・チップ (reprogrammable configurable chip) 10、現時点での好ましい実施例ではフィールド・プログラマブル (field-programmable) ・ゲート・アレイ (FPGA) により構成される。各マルチプレクサ・チップ 12 は、各ロジック・チップ 10 に対して 1 つ又は複数の接続を有する。Butts が開示している好ましい実施例では、ロジック・チップからマルチプレクサ・チップに向かう各設計信号は、1 つの物理的相互接続を必要とする。換言すれば、ロジック・チップ 10 の 1 つのピン 40

上の 1 つの信号は、マルチプレクサ・チップ 1 2 の 1 つのピンに相互接続される。本発明の実施例では、パーシャル・クロスバー・アーキテクチャにおける各物理的相互接続は、1 つ又は複数の設計信号を表わす。

各マルチプレクサ・チップ 1 2 は、クロスバー 2 2、複数の入力デマルチプレクサ 2 4 及び複数の出力マルチプレクサ 2 6 を有する。入力デマルチプレクサ 2 4 は、時間多重入力信号を 1 つ又は複数のコンポーネント信号に分割する。コンポーネント信号は、クロスバー 2 2 を介して分離して伝送される。これらの信号は、出力マルチプレクサ回路 2 6 によって、同一又は異なる組み合わせにおいて再び多重化される。好ましい実施例では、時間多重信号は、マルチプレクサ・チップ・クロスバー 2 2 を経由して伝送されない。時間多重信号をマルチプレクサ・チップ、クロスバー 2 2 を経由して伝送しないことによって、マルチプレクサ・チップに対する入力信号及び出力信号が異なる組み合わせで組み合わされるため、パーシャル・クロスバー・ネットワークをルーティング (routing) する柔軟性が向上する。これは、時間多重周波数がコンポーネント信号の平均スイッチング比よりも格段に高いため、マルチプレクサ・チップの消費電力を削減する。

ロジック・チップ 1 0 は、また、複数の入力デマルチプレクサ 3 4 及び出力マルチプレクサ 3 6 を有する。出力マルチプレクサ 3 6 は、1 つ又は複数のロジック・チップ 1 0 の信号を 1 つの物理的相互接続に組み合わせる。入力デマルチプレクサ 3 4 は、時間多重信号を 1 つ又は複数の内部ロジック・チップ 1 0 の信号に分割する。現時点で好ましい実施例では、これらのマルチプレクサ 3 6 及びデマルチプレクサ 3 4 は、商業的に入手可能な既製の F P G A の内部的に構成可能なロジック・ブロックを使用して構成される。しかしながら、これらは、エミュレーションのために注文設計された再プログラム可能なロジック・チップの入力 / 出力ブロックを用いて構成することもできる。

図 1 は、専ら説明のためのものであるが、それぞれ 4 つのピンを有する 2 つのマルチプレクサ・チップ 1 2 及びそれぞれ 2 つのピンを有する 4 つのロジック・チップ 1 0 を示している。好ましいハードウェア・エミュレーション・システムの現実的な実施例では、より多くの種類のチップを有し、各チップはより多くのピンを有するであろう。マルチプレクサ・チップ 1 2、ロジック・チップ 1 0 の実際の数及び各チップのピンの数は、純粹に設計上の選択の問題であり、達成すべき望ましいゲート容量に依存する。現時点における好ましい実施例では、各プリント回路基板は、5 4 個のマルチプレクサ・チップ及び 3 7 個のロジック・チップ 1 0 を含む。現時点において好ましいロジック・チップ 1 0 は、4 0 3 6 X L F P G A (“ロジック・セル・アレイ”とも呼ばれる) であり、Xilinx Corporation によって製造されたものである。しかしながら、Altera Corporation、Lucent Technologies、Actel Corporation 等によって提供される他の再プログラム可能なロジック・チップも使用可能であることに注意すべきである。現時点における好ましい実施例では、3 6 個のロジック・チップ 1 0 が 5 4 個のマルチプレクサ・チップ 1 2 の各々に 5 つの接続を持っている。これは、これらの 3 6 個のロジック・チップ 1 0 の各々の 5 つのピンが 5 4 個のマルチプレクサ・チップ 1 2 の各々の 5 つのピンに対する物理的、電気的な接続を有している。3 7 番目のロジック・チップ 1 0 は、5 4 個のマルチプレクサ・チップのそれぞれに対して 3 つの接続を持っている。これは、3 7 番目のロジック・チップ 1 0 の 3 つのピンが、5 4 個のマルチプレクサ・チップ 1 2 の各々の 3 個のピンに物理的、電気的に接続されていることを意味する。

図 2 は、2 対 1 の時間多重エミュレーション・システムのタイミング図であり、内部ロジック・チップ信号 A 4 0 及び内部ロジック・チップ信号 B 4 2 が单一の出力信号 4 6 に多重化される。多重クロック信号 4 4 は、分周クロック信号 5 0 を生成するために 2 分周される。同期信号 4 8 は、多重クロック分周器 6 8 (図 3 参照) を同期させる。すなわち、同期信号 4 8 がローレベルの時、多重クロック 4 4 の立下りエッジによって分周クロック信号 5 0 をゼロに設定する。分周クロック 5 0 は、各立ち上がりエッジで内部信号 A 4 0 をサンプリングする。このサンプルは、フリップ・フロップ又はラッチ (図 3 参照) のような格納エレメントに格納される。同じ分周クロック信号 5 0 は、各立ち下りエッジで内部信号 B 4 2 をサンプリングするために使用される。このサンプルは、他のフリップ・フ

10

20

30

40

50

ロップ又はラッチ（図3参照）に格納される。好ましい実施例では、多重クロック信号44は、信号A40及び信号B42に対して非同期であってもよい。分周クロック信号50の値がハイレベルの時には、先にサンプリングされた信号A40は出力信号46に転送される。分周クロック信号50がローレベルの時には、先にサンプリングされた信号B42は出力信号46に転送される。

図3を参照すると、現時点で好ましい実施例のFPGA10内に設けられ、図2に示す信号のタイミングを作成するロジックの詳細が示されている。2対1のクロック分周器68は、多重クロック信号44を分周して分周クロック信号50を生成する。クロック分周器68は、フリップ・フロップ68a、ANDゲート68bとインバータ68cを有する。分周クロック信号50は、出力マルチプレクサ66（図1のマルチプレクサ36参照）及び入力デマルチプレクサ64（図1のデマルチプレクサ34参照）に入力される。クロック分周器68は、システム内のすべてのクロック分周器が同期化されるのを補償するために、同期信号48によって周期的にリセットされる。入力デマルチプレクサ64は、2つのフリップ・フロップ65a及び65bを有する。フリップ・フロップ65a及び65bは、多重クロック44が供給される。1つのフリップ・フロップ（例えば65b）は、分周クロック信号50がハイレベルの時にイネーブルされ、他のフリップ・フロップ（例えば65a）は、分周クロック信号50がローレベルの時にイネーブルされる。分周クロック信号50は、FPGA10の低スキュー・ライン（low-skew line）を維持するために、入力デマルチプレクサ64及び出力マルチプレクサ66のフリップ・フロップ65a、65bに対するクロックとしては直接に使用されない。フリップ・フロップ65a又は65bのいずれかの出力は、静的にデマルチプレクスされた設計信号をFPGA10のコア62（FPGA10のコア62は、ユーザー設計のロジック機能を実現するために使用される構成可能なエレメントを有する。）に供給する。出力マルチプレクサ66は、多重クロック44が供給される2つのフリップ・フロップ67a、67bを有する。出力マルチプレクサ66は、2対1のマルチプレクサ67cも有する。一方のフリップ・フロップ（例えば、フリップ・フロップ67b）は、分周クロック信号50がハイレベルの時にイネーブルされ、他のフリップ・フロップ（例えば、フリップ・フロップ67a）は、分周クロック信号50がローレベルの時にイネーブルされる。2対1のマルチプレクサ67cは、フリップ・フロップ67a又はフリップ・フロップ67bのいずれかの出力Qを選択し、出力ピンに出力する。

マルチプレクサ・チップ12に対応する回路を図4に示す。ロジック・チップ10の回路とは異なり、マルチプレクサ・チップ12の出力マルチプレクサ76（図1のマルチプレクサ26参照）は、フリップ・フロップを備えてなく、そのため、2対1のマルチプレクサ76aを有している。これは、好ましい実施例では、マルチプレクサ・チップ12を介した遅延が小さいので実現可能である。付加的なロジックを省略するために、入力デマルチプレクサ74（図1のデマルチプレクサ24参照）のフリップ・フロップ74a、74bは、イネーブル入力を持たない。その代わり、分周クロック信号50がフリップ・フロップ74a、74bに直接に供給される。クロック分周器78は、好ましくは、フリップ・フロップ78a、ANDゲート78b及びインバータ78cを有する。クロック分周器78、入力デマルチプレクサ74、出力マルチプレクサ76は、図3の対応するエレメントと同様に動作する。

マルチプレクサ・チップ12の入力／出力（“I/O”）ピンが与えられた設計に対して入力又は出力となるかは前もって分からぬいため、マルチプレクサ・チップ12の全てのI/Oピンは、入力デマルチプレクサ74及び出力マルチプレクサの両方を含んでいる。本発明の概念を利用することにより、ピンが、ある時点では入力となり、ある時点では出力となる、4対1の時間多重を行うことが可能である。図5は、4対1の時間多重のタイミング図を示すものである。2対1の時間多重の場合と同様に、多重クロック信号44及び同期信号48が存在する。多重クロック信号44は、分周クロック信号50を生成するために2分周される。分周器は、同期信号がローレベルで、多重クロック信号44が立ち下がった時にリセットされる。これらに加えて、分周クロック信号50をさらに2分周し

10

20

30

40

50

た追加の方向信号 8 0 が存在する。方向信号 8 0 は、各時点においてピンが入力であるか出力であるかを制御する。4つのイネーブル信号 E 0 - 9 0 、 E 1 - 9 2 、 E 2 - 9 4 及び E 3 - 9 6 は、後述するように、ロジック・チップ 1 0 の各フリップ・フロップをイネーブルするために使用される。これらの4つの信号は、分周クロック信号 5 0 及び方向信号 8 0 から形成される。

分周クロック信号 5 0 は、外部信号 9 8 をサンプリングし、方向信号 8 0 がローレベルの時に、内部入力信号 E - 8 6 及び F - 8 8 を生成する。方向信号 8 0 がローレベルの時には、ピンが入力方向で動作していることを示す。内部入力信号 E - 8 6 は、分周クロック信号 5 0 の立ち上がりエッジでサンプリングによって生成される。内部入力信号 F - 8 8 は、分周クロック信号 5 0 の立ち下りエッジでサンプリングによって生成される。方向信号 8 0 がハイレベルの時には、これを受信したピンは出力として動作する。内部出力信号 C - 8 2 は、分周クロック信号がローレベルの時に外部信号 9 8 に出力され、分内部出力信号 D ( 8 4 ) は、分周クロック信号 5 0 がハイレベルの時に出力される。

図 6 を参照すると、図 5 のタイミング信号を生成するためにロジック・チップ 1 0 に設けられたロジックの詳細が示されている。クロック分周器 1 0 4 は、多重クロック信号 4 4 を分周し、分周クロック信号 5 0 及び方向信号 8 0 を生成する。クロック分周器 1 0 4 は、フリップ・フロップ 1 0 4 a 、 1 0 4 b 、 AND ゲート 1 0 4 c 、 1 0 4 d 、インバータ 1 0 4 e 、イクスクルーシブ OR ゲート (exclusive-or gate) 1 0 4 f 、 AND ゲート 1 0 4 g ~ 1 0 4 j を有する。クロック・分周器 1 0 4 は、システム内の全てのクロック分周器 1 0 4 が同期されるのを補償するために、同期信号 4 8 によって周期的にリセットされる。これらに加えて、クロック分周回路 1 0 4 は、また、イネーブル信号 E 0 - 9 0 、 E 1 - 9 2 、 E 2 - 9 4 及び E 3 - 9 6 を生成する。これらの信号は、入力 / 出力マルチプレクサ回路 1 0 0 及び 1 0 2 においてイネーブルとして使用される。

入力 / 出力マルチプレクサ回路 1 0 0 は、図 5 に対応するタイミングを有している。外部信号 9 8 は、方向信号 8 0 がローレベルの時に入力となる。信号 E - 8 6 及び信号 F - 8 8 は、イネーブル信号 E 0 - 9 0 及びイネーブル信号 E 1 - 9 2 がアクティブである時に外部信号 9 8 からサンプリングされ、それぞれフリップ・フロップ 1 0 0 a 及び 1 0 0 b に置かれる。信号 D - 8 4 及び C - 8 2 は、イネーブル信号 E 2 - 9 4 及び E 1 - 9 2 がアクティブである時に、フリップ・フロップ 1 0 0 c 及び 1 0 0 d にセーブされる。好ましい入力 / 出力マルチプレクサ回路 1 0 0 は、また、マルチプレクサ 1 0 0 e 及びバッファ 1 0 0 f を有する。これらの構成により、フリップ・フロップ 1 0 0 c 及び 1 0 0 d に予めセーブされた信号 D - 8 4 及び C - 8 2 は、方向信号 8 0 がハイレベルの時に外部信号 9 8 に連続して出力される。

入力 / 出力のマルチプレクサ回路 1 0 2 は、方向信号 8 0 がローレベルの時に信号 1 0 6 が出力となるようにタイミングが変更されている点を除いて、同様の構成である。入力 / 出力マルチプレクサ 1 0 2 は、好ましくは、フリップ・フロップ 1 0 2 a ~ 1 0 2 d 、マルチプレクサ 1 0 2 e 、バッファ 1 0 2 f を有する。ここで、入力 / 出力マルチプレクサ 1 0 0 を“インアウト”マルチプレクサ (inout multiplexer) と呼び、マルチプレクサ 1 0 2 を“アウトイン”マルチプレクサ (outin multiplexer) と呼ぶものとする。ピンがシステム内で互いに接続されている時、一方のピンがドライブ (drive) し、他方のピンが読み取る（即ち、信号の受信準備ができている又は信号を受信中）ことができるように、インアウト・ピンは常にアウトイン・ピンに接続されている必要がある。

対応する4線 (4-way) の時間多重回路を、図 7 にマルチプレクサ・チップ 1 2 として示す。クロック分周器 1 3 2 は、分周クロック信号 5 0 及び方向信号 8 0 を生成する。クロック分周器 1 3 2 は、フリップ・フロップ 1 3 2 a 、 1 3 2 b 、 AND ゲート 1 3 2 c 、 1 3 2 d 、インバータ 1 3 2 e 及びイクスクルーシブ OR ゲート 1 3 2 f を有する。ロジック・チップ 1 0 と同様に、インアウト・マルチプレクサ 1 2 0 及びアウトイン・マルチプレクサ 1 2 2 が存在する。インアウト・マルチプレクサ 1 2 0 は、好ましくは、フリップ・フロップ 1 2 0 a 、 1 2 0 b 、 2 対 1 マルチプレクサ 1 2 0 c 及びバッファ 1 2 0 d を有する。インアウト・マルチプレクサ 1 2 0 は、図 5 に示すタイミングを有する。外部

10

20

30

40

50

信号 9 8 は、方向信号 8 0 がローレベルの時に入力となり、方向信号 8 0 がハイレベルの時に出力となる。内部信号 1 2 4 及び 1 2 6 は、方向信号 8 0 がローレベルの時に外部信号 9 8 からサンプリングされる。内部信号 1 2 8 及び 1 3 0 は、方向信号 8 0 がハイレベルの時に外部信号 9 8 に出力される。

アウトイン・マルチプレクサ回路 1 2 2 は、信号 1 3 4 が、方向信号 8 0 がハイレベルの時に入力となり、方向信号 8 0 がローレベルの時に出力となるようにタイミングが変更されている点を除いて、同様の構成である。アウトイン・マルチプレクサ 1 2 2 は、好ましくは、フリップ・フロップ 1 2 2 a、1 2 2 b、2 対 1 のマルチプレクサ 1 2 2 c 及びバッファ 1 2 2 d を有する。マルチプレクサ・チップ 1 2 のアウトイン・ピンは、他のマルチプレクサ・チップ 1 2 又はロジック・チップ 1 0 のインアウト・ピンに接続する必要がある。付加的な構成ビット (configuration bit) (図 7 に図示していない) により、マルチプレクス・チップ 1 2 の任意のピンを、多重化 (non-multiplexed) されていない入力又は出力、2 対 1 に多重化された入力又は出力としても、4 対 1 に多重化されたインアウト・ピン又はアウトイン・ピンとしてもプログラム可能に構成することができる。これは、選択的に、方向信号 8 0 を常にローレベル (2 対 1 入力に対応)、常にハイレベル (2 対 1 出力に対応)、非反転状態 (1 2 0 のようなアウトイン・4 対 1 ピンに対応) とし、又は反転状態 (1 2 2 のようなアウトイン・4 対 1 ピンに対応) とすることによって実現される。これらに加えて、外部信号 9 8 は、非多重化入力に対応して、コア信号 1 2 4 に直接に接続することができる。コア出力 1 2 8 及び 1 3 0 は、非多重化入力に対応して、バッファ 1 2 0 d の入力及びイネーブル・ピンに直接に接続することができる。

好ましい実施例では 2 対 1 及び 4 対 1 の時間多重を採用しているが、ここに開示された技術は、設計者によって選択されるあらゆる他の要因による多重化についても拡張して適用することができる。一般的には、高度に多重化を行うと、エミュレーション速度は低下するが、物理的なワイヤ及びピンをより多くのロジック設計信号間で共有することができるため、ハードウェアを簡略化することができ、低コストとなる。

さらに、エミュレーション・システムに使用可能な、情報の複数のビットを 1 つの物理的ワイヤに多重化する多くの他の方法が存在する。これらの技術の例としては、パルス幅変調、位相変調及びシリアル・データ・エンコーディング等がある。特定の実施例どの方法を選択するかは、設計者の選択の問題であり、動作速度、コスト、消費電力及び必要とされるロジックの複雑さの間での相対関係に依存する。

ハードウェア・エミュレーション・システムにおいて重要なこれらのさらに複雑なエンコーディング手法の一側面は、消費電力を低減することができるることである。ハードウェア・エミュレーション・システムは、通常、数千の相互接続経路を有している。システムを通じた遅延を最小化するために、これらの相互接続経路を異なるロジック設計信号の間でできるだけ速く切り替えることが望ましい。しかしながら、システムの消費電力は、相互接続経路が切り替えられる速度に大きく依存している。大規模なシステムでは、電力の生成及び分配と共に伴う発熱の除去のために、システムの複雑さとコストが大きく増大する。このため、高速で動作するが、大きな電力を必要としない多重方式を採用することが望ましい。消費電力を最小化する方法の 1 つとして、設計信号を連続して転送するのではなく、設計データが変更された時にのみ設計信号情報を転送するものがある。この方法は、現時点での好ましい実施例で採用されている。

エンコーディング方法を選択する時に考慮すべき他の重要な観点は、相互に非同期で又はマスター・多重クロック (master multiplexing clock) に非同期で動作する相互接続を有することができるることである。前記した、現時点での好ましい実施例に対応する時間多重の簡単な形式では、マスター・多重クロックを、低スキュー (low skew) でシステム内の全てのロジック・チップ 1 0 及びマルチプレクス・チップ 1 2 に分配する必要がある。これに加えて、マスター・多重クロックは、システム内の最も長い相互接続上を信号が通過可能であるように十分に低い速度とする必要がある。同時に、システム内の最も短い相互接続においてホールドタイム違反 (hold-time violation) が発生してはならない。ホールドタイム違反は、受信装置がデータ信号をフリップ・フロップ又はラッチに適切にセー

10

20

30

40

50

ブする前に、送信装置がデータ信号を除去することによって生じ得る。低スキーの、マスタ・クロックに対する要求によって、エミュレーション・システムの複雑さとコストは大幅に増大する。これに加えて、想定可能な最短データ経路上でのホールドタイム違反を防止するとともに、想定可能な最長データ経路上を信号が通過するのに十分な時間を確保することは、多重クロックが比較的低速で動作しなければならないことを意味する。すでに説明したとおり、これは、エミュレーション・システムの実効的な動作速度を限定するため、好ましくない。

前記した、時間多重の最も簡単な形式に関する発明の概念は、より複雑なエンコーディング方法にも同様に適用可能である。以下に、それを説明する。パルス幅変調、位相シフト変調及びシリアル・エンコーディングを用いたエンコーディング方法は、消費電力を低減し、最も簡単な形式の時間多重に本質的な問題である比較的低い動作速度を増加させることができる。これらのすべての手法の欠点（簡単な時間多重に関して）は、これらが格段に多くのエンコーディング・ロジック及びデコーディング・ロジックを必要とすることである。このため、現時点での好ましい実施例では、簡単な時間多重が用いられていた。デジタル・ロジックのコストが物理的なピンや回路基板経路（circuit board trace）のコストに比較して減少するにつれ、将来的には、1つ又は複数のこれらより複雑なエンコーディング手法が使用されることとなろう。

図8を参照すると、ハードウェア・エミュレーション・システムに適すると思われるパルス幅変調の1つの形式が示されている。外部信号146は、通常はローレベルである。設計信号140又は142が変化すると、パルスが外部信号146に出力される。高速の非同期クロック信号144は、システム内の全てのチップに分配される。図2を参照して述べた多重クロック44とは異なり、非同期クロック信号144は、システム内の任意の2つのチップ間、又は同じチップの2つのピンの間であっても同期化する必要がない。そのため、図2を参照して述べた同期信号48は必要ない。また、非同期クロック144は、外部信号144に生成される最小パルス幅が品質の劣化を伴わずに相互接続を介して通過することができる限り、任意の速度で動作させることができる。外部信号146に出力されるパルスは、2つの設計信号140及び142の信号変化時点における値が00、01、10又は11であるか否かに応じて、1つ、2つ、3つ又は4つのクロックの幅を有する。しかしながら、非同期クロック信号144は、情報が失われることがないように、設計信号140及び142の連続するエッジの間に常に5つのクロック・サイクルが経過するよう十分に高速でなければならない。データ信号140及び142は、外部信号146がハイレベルとなるたびに発生する非同期クロック144のサイクル数をカウントすることにより、外部信号146から再生することができる。現実の実施例では、非同期クロック144は、再生された信号が明確に判別できるように、ここに示す速度の2倍又は3倍の速度で動作することになるであろう。また、付加的な回路は、設計信号が変化しない場合でも設計を適正に初期化できるようにするために、周期的にデータを転送するための附加回路が加えられる。

設計信号140及び142は、非同期クロック144と平均的に比較して比較的頻繁には変化しないため、消費電力は、前記した連続的な時間多重に比べて低くなる。それに加えて、このエンコーディング手法は、送信回路及び受信回路の間の遅延量の変化の影響を受けない。

このパルス幅エンコーディング手法を実現するために必要なロジック回路は、回路設計の分野における当業者によって設計が可能である。ここでは、これ以上論じない。しかしながら、当業者は、同じ機能を実現する、多くの異なる種類のロジック回路を設計することができることに注意すべきである。例えば、2つではなく、3つの設計信号を1つの外部信号146にエンコードすることができる。また、設計信号140及び142の異なるエンコード手法を用いることができ、又は外部信号146のデフォルト値を0ではなく1とすることもできる。

図8に示したパルス幅変調エンコーディング手法は、以下の制約がある。パルス幅変調エンコード手法では、パルス幅は、外部信号146の立ち上がりエッジから外部信号146

10

20

30

40

50

の立ち下りエッジの間で測定される。しかしながら、信号がルーティング・チップ (routing chips) の多くのレベルを通過する時、立ち上がりエッジは、しばしば、立ち下りエッジとは異なる遅延が発生する。このため、信号多重の速度は、ルーティング・チップの多くのレベルを通過した後でも信号の値を判別することができるよう、低速化する必要がある。また、図 8 の変調手法は、避けることのできない一時的な信号の変化又は外部信号 146 のグリッチに敏感である。このため、誤った信号の値が送信される可能性がある。

図 9 を参照すると、ハードウェア・エミュレーション・システムに適していると思われる位相変調の一方式を示す。内部の位相ロックループ (“PLL”) 回路は、入力として非同期クロック 144 を用い、連続的に 0 から 3 までカウントする（図 9 の PLL カウント 150 参照）。この PLL 回路は、一般的にデジタル位相ロックループ (“DPLL”) と呼ばれるもので、相補型の金属酸化半導体 (“CMOS”) 集積回路技術を用いて比較的簡単に構成することができる。設計信号の 140 又は 142 に変化が生じると、外部信号 152 は、設計信号 140 及び 142 の値に依存した時間に変化する。例えば、信号 A - 140 の最初の変化の後では、信号 A - 140 及び信号 B - 142 の両方がハイレベルとなる。そのため、外部信号 152 は、PLL がカウント 3 の状態 (A, B = 11) の時に変化する。その後、信号 B - 142 の変化の後で、信号 A - 140 がハイレベルとなり、信号 B - 142 がローレベルとなる。このため、外部信号 152 は、PLL がカウント 2 の状態 (A, B = 10) の時に変化する。

受信回路は、転送すべきデータがない時に定期的に送信される同期パルスによって送信 PLL との同期状態を保持される、マッチング PLL を有する。同期パルスは、送信 PLL の時間 0 及び時間 2 において発生する 2 つの変化により構成される。1 つの PLL サイクル内で外部信号 152 に 2 つの変化が生ずるのは同期パルスが送信される時のみであるため、同期パルスは、受信 PLL によって認識することができる。同期パルスによって受信 PLL のカウントが除々に調整され、いくつかの同期パルスが発生した後に受信 PLL は送信 PLL と同期状態となる。同期パルスは、信号 A - 140 及び信号 B - 142 の変化に比べて比較的少ない頻度で発生させれば充分であるため、消費電力は大きくは増加しない。現実の実施例では、非同期クロック 144 は、外部信号 152 の異なるエッジ変化時間の間を明確に区別するのに十分な分解能を持たせるために、図 9 に示す速度の 2 倍又は 3 倍の相対速度で動作させることになるであろう。代わりに、分解能を向上させるために、位相ロックループを非同期クロック 144 の周波数の数倍の速度で動作させることもできる。また、設計信号 A - 140 及び設計信号 B - 142 が変化していない場合でもその値を定期的に送信する回路を付加し、設計が適正に初期化されるようにすることもできる。

この位相エンコード手法で用いられる、デジタル位相ロックループを構成するために必要な回路、送信及び受信回路は、回路設計の技術分野における当業者によって設計可能であり、ここでは、これ以上論じない。

前記した位相変調エンコード手法は、これまでに論じたパルス幅変調手法（図 8 参照）に対していくつかの利点を有する。図 8 の外部信号 146 の場合に比べて、信号 A - 140 及び信号 B - 142 の値を送信するために必要な外部信号 152 の変化が少ない。これにより、システムによって消費される電力が低減される。また、故障 (glitch) パルス又は短パルスは同期パルスとして扱われ、PLL カウント 150 は除々に影響を受けるのみであるため、ノイズに対する回路の感度を引き下げることができる。これに加えて、同期パルスは常に 1 つの立ち上がりエッジと 1 つの立ち下りエッジを有しているため、立ち上がりエッジと立ち下りエッジを計時するために別個の PLL カウンタを使用することができる。立ち上がりエッジと立ち下りエッジを個別に計時することにより、非同期クロック 144 は、非常に高い周波数で動作することができるとともに、信号 A - 140 及び信号 B - 142 の再生の信頼性に影響を与えることなく、外部信号 152 を中間のルーティング・チップを介して通過させることができる。しかしながら、位相変調の主な欠点は、比較的大量のデジタル・ロジックを組み込む必要があることである。

10

20

30

40

50

ここに開示した位相変調エンコード手法については、本発明の趣旨を逸脱することなく様々な変更を加えることができる。例えば、PLLは、4つ以外の8又は16の変化時点を認識することができる。また、設計信号に変化が発生する毎に外部信号152に複数のエッジを作成することによって、付加的な設計信号を送信することができる。例えば、設計信号A及びBは、外部信号152の最初のエッジで送信することができ、設計信号C及びDは、外部信号152の2番目のエッジで送信することができる。これにより、外部信号152でより多くのデータを送信できるという効果を生じるが、速度は低くなる。

図10を参照すると、ハードウェア・エミュレーション・システムに使用可能と思われる、変調の他の形式が示されている。この技術は、シリアル・データ・エンコーディングと呼ばれている。RS232C等の多くの一般的なプロトコルでは、シリアル・データ・エンコーディングの変形を使用している。設計信号A-140又は設計信号B-142が変化すると、データのシリアル・ストリング(serial string)が外部信号162に送信される。常に0であるスタート・ビットは、送信が開始されることを示す。次に、信号A-140及び信号B-142の値が連続して送信される。最後に、常に1であるストップ・ビットが送信される。受信回路は、信号A-140を再生するために外部信号162をサンプリングする前に、非同期クロック信号144を使ってスタート・ビットの立ち下りエッジから1及び1.5クロックだけ遅延させる。そして、信号B-142を再生するために外部信号162をサンプリングする前に、さらに1クロック遅延させる。現実の実施例では、信号A-140及び信号B-142が送信されている時、外部信号162をその中央点で正確にサンプリングするために、非同期クロック144は、図10に示す周波数より数倍高い周波数で動作することになる。

シリアル・データ・エンコード手法を構成するために必要な回路は、回路設計の技術分野における当業者によって設計可能であり、ここでは、これ以上論じない。

シリアル・データ・エンコーディングは、比較的簡単なデジタル・ロジックを使用できるという利点がある。しかしながら、設計信号A-140及び設計信号B-142に対する変化を送信する毎に外部信号162に複数のエッジが必要であるという欠点がある。これは、データ・レートが比較的低く、他の技術に比較して消費電力が比較的に高いことを意味する。

ここで開示したシリアル・データ・エンコード手法については、本発明の趣旨を逸脱することなく様々な変更を加えることができる。例えば、設計信号が変化した時毎に、2つ以上の設計信号の値を送信することもできる。

図8～10に示す各エンコード手法は、いくるかの形式のエラー・チェック技術を追加することによってさらに改良することができる。設計データは設計信号が変化したときにのみ送信されるため、送信エラーが発生すると、受信回路によって誤ったデータ値がラッチされることになり、またエミュレーションが誤動作する可能性がある。パリティ・チェックや巡回型冗長性チェック(CRC)等の一般的なエラー検出を用いることができる。

好ましい実施例のシステム面をさらに詳細に開示する。図11を参照すると、ロジック・チップ(現時点での好ましい実施例ではFPGA)及びマルチプレクサ・チップ12を組み込んだ、好ましい実施例のロジック・ボード200のブロック図が示されている。ロジック・ボード200は、Buttsによって開示されたものに類似するパーシャル・クロスバー・インターフェクションを有している。主な相違点は、本発明の現時点での好ましい実施例のパーシャル・クロスバーは、後述するロジック・チップ204に接続されているマルチプレクサの数が他のロジック・チップより少ないため、完全に同型(uniform)になっていないことである。現時点での好ましい実施例では、それぞれが260個のI/Oピンを有する54個のマルチプレクサ・チップ12及びそれぞれが270個のI/Oピンを有する36個のロジック・チップ(FPGA)がある。現時点での好ましい実施例では、Xilinx Corporation, San Jose, California, U.S.A.により製造された部品番号XC4036XLのFPGAをロジック・チップ10として使用する。36個のロジック・チップ10の各々は、54個のマルチプレクサ・チップ12の各々との間に5つの接続を有している。37番目のロジック・チップ204は、ここでは共同シミュレーション(co

10

20

30

40

50

-simulation) (CoSim) として知られているが、54個のマルチプレクサ・チップの各々との間に3つの接続を有している。現時点での好ましい実施例では、37番目のロジック・チップ204も、Xilinxによって製造された部品番号4036XLのFPGAである。マルチプレクサ・チップ12及びロジック・チップ10、204の追加のピン(図示していない)は、ダウンロード、クロック分配及び他のシステム機能のために確保されている。以下に、CoSimロジック・チップの目的を説明する。任意のマルチプレクサ・チップ12からロジック・チップ10への接続は、マルチプレクサ・チップ12及びロジック・チップ10を適当にプログラムすることにより、非多重、2対1の多重、4対1の多重とすることができます。

前記した相互接続に加え、CoSimロジック・チップ204も、プロセッサ206と電気的に通信を行う。現時点での好ましい実施例では、プロセッサ206は、IBM Corporationから入手可能なPowerPC 403GCチップである。プロセッサ206は、共同シミュレーションのために使用される。これは、Sampleらによる「Method And Apparatus For Design Verification using Emulation And Simulation」という名称の米国特許第5,841,967号を開示されている。参考のため、米国特許第5,841,967号の開示内容をここに組み入れる。プロセッサ206は、診断機能や、マルチプレクサ・チップ12、ロジック・チップ10、204、RAM208(以下に説明する)及びSGRAM210(以下に説明する)への情報のダウンロードに使用される。プロセッサ206は、VMEインターフェース(図示していない)を介してバックプレーン・コネクタ(back plane connector)220に接続される。また、12個のロジック・チップ10は、32個の静的・ランダム・アクセス・メモリ(static randomaccess memory)(RAM)・チップ208によって32Kに接続される。このRAMチップ208は、エミュレートされる回路の一部となり得る大きなメモリを構成するために使用される。また、RAM208は、ロジック・チップ10をマルチプレクサ・チップ12に接続している線の一部に取り付けられる。このようにして、RAMが使用されていない場合には、ロジック・チップ10からマルチプレクサ・チップ12への接続は、通常の相互接続機能として使用することが可能であり、これらが失われることはない。RAM208が特定のネットリストの一部のメモリを構成するために必要とされる場合には、それと通信するロジック・チップ10にはRAMコントローラ機能がプログラムされる。

マルチプレクサ・チップ12は、バック・プレーン・コネクタ220及びターボ・コネクタ(turbo connector)202との間にも接続を有している。バック・プレーン及びターボとの接続は、非多重、2対1の多重あるいは4対1の多重のいずれでもよい。ターボ・コネクタ202は、2つのロジック・ボード200をサンドウィッチの形で電気的に接続するために使用される。2つのロジック・ボードを1組として直接に接続することにより、特定の設計について必要とされるバックプレーン接続が軽減される。バックプレーン・コネクタは、ロジック・ボードの一辺に沿って適合するものでなければならず、可能なバックプレーン接続の数は利用可能なコネクタの種類によって限定される。不十分なバックプレーン接続が存在する場合には、パーティショニング・ソフトウェア(partitioning software)は、効率的に動作することができず、ボードのロジック容量は低下する。サンドwich状に接続された2つのエミュレーション・ボードを図13に示す。2つより少ないエミュレーション・ボードから成る小規模なエミュレーション・システムが要求される場合には、ロジック回路が配設されていない特別のターボ・ループバック・ボード(turbo loopback board)が使用される。このようなシステムでは、この特別なターボ・ループバック・ボードは、ターボ・コネクタ202からバック・プレーン・コネクタ220に信号をルーティング(route)する。ターボ・ループバック・ボードを使用した構成の例を図15に示す。

これに加えて、マルチプレクサ・チップ12は、1組の同期グラフィックRAM(SGRAM)210に対する8つの接続を有する。これらのSGRAM210は、分配されたロジック・アナライザのデータ経路を形成するために使用される。設計信号は、ロジック・チップ10及びCoSimロジック・チップ204でサンプリングされ、マルチプレクサ・チ

10

20

30

40

50

ツップ12を介してルーティングされ、そしてユーザによる将来的な解析のためにSGRA M210にセーブされる。ロジック・アナライザについて、以下で説明する。

ロジック・チップ10及びCosimロジック・チップ204は、また、イベント・バス212及びクロック入力・バス214が設けられている。イベント・バスは、イベント信号をロジック・チップ10及びCosimロジック・チップ204内からロジック・アナライザ制御回路（以下に説明する図20a参照）にルーティングする。イベント・バスは、4つの信号により構成され、8つの信号を供給するために2対1に時間多重化される。イベント・バス212の信号は、バッファに格納され、そしてバックプレーン・コネクタ220上の付加ピン（図示していない）にルーティングされる。

バス214のクロックは、すべてのロジック・チップ10及びCosimロジック・チップ204（以下に説明する）にルーティングされる8つの低スキューレ特別目的のクロック・ネット（clock net）により構成される。バス214のクロックは、米国特許No. 5,475,830号に説明されているように、クロック信号を分配するために使用される。バス214のクロックからのクロックは、バック・プレーン・コネクタ220上の付加ピン（図示していない）に接続される信号218からバッファ216を介して直接的に送られ、又は主クロック信号をCosimロジック・チップ204内のロジックと組み合わせることによって作成される。Cosimロジック・チップ204がクロック・ロジックを供給するために使用されている場合には、米国特許No. 5,475,830号説明されているように、“クロック生成FPGA”として動作する。

図12参照すると、ボード間の相互接続が示されている。ロジック・ボードは、ターボ・コネクタ202を介して接続される一対として組み立てられる。ロジック・ボードは、また、バックプレーン・コネクタ220（図11参照）介してスイッチング・バックプレーン420に接続される。スイッチング・バックプレーン420は、ロジック・ボードに直角に配設されたマルチブレクサ・ボード400により構成される。ロジック・ボード及びスイッチング・ボードの配置は、本発明の譲受人と同一の譲受人であるSampleらに譲渡された米国特許No. 5,352,123号に開示されている。米国特許No. 5,352,123号は、参考のために、ここに完全な形で組み込まれる。スイッチング・バックプレーン420は、また、I/Oボード300に接続されている（図12では、1つのI/Oボードのみが示されている。しかしながら、複数のI/Oボード300を使用することも、本発明の一部であると考えられる。）。I/Oボード300は、コア・ボード500又は外部システム540に含まれる外部装置からの信号をルーティングするとともにバッファする機能を提供する。また、これらは、すべての外部ピンに対して刺激信号（stimulus signal）を提供する能力を有している。それによって、エミュレートされる設計は、外部装置又は外部システムが無い場合でも動作可能である。

I/Oボード300は、コア・ボード500及びリピータ・ポッド520を介して外部システム540に接続される。簡略化のため、図12では、現実のボード及び接続の数より低減されている。現時点における好ましい実施例では、22個のマルチブレクサ・ボード400、1から10対のロジック・ボード200及び最大8個のI/Oボード300が存在する。現時点における好ましい実施例では、2以上のI/Oボードが使用されると、一対のロジック・ボード200は、追加のI/Oボード300の対の各々について失われる。現時点における好ましい実施例では、各I/Oボード300は、ケーブルに取り付けられる、最大7つのリピータ・ポッド520を有するコア・ボード500を1つ有している。現時点における好ましい実施例における各リピータ・ポッドは、88の双方向信号をバッファする。

図13は、好ましい実施例のシステムの物理的な構成を示したものである。マルチブレクサ・ボード400は、ロジック・ボード200及びI/Oボード300に対して直角に設けられている。バックプレーン800は、一方側にマルチブレクサ・ボード400に対するコネクタを有し、他方側にロジック・ボード200又はI/Oボード300に対するコネクタを有する。図面の簡略化のため、1つのマルチブレクサ400及び3対のロジック・ボード200のみを示している。しかしながら、現時点における好ましい実施例では、

10

20

30

40

50

現実には、22個のマルチプレクサ・ボード400及び最大7対までのロジック・ボード200又はI/Oボード300が存在する。I/Oボード300は、コネクタ330を介してコア・ボード500に取り付けられている。コア・ボード500は、外部コネクタ510を有している。外部コネクタ510は、ケーブルを介してリピータ・ポッド520及び外部システム540(図13に示していない)に装着される。電源ボード240は、48VのDC主電源を、ロジック・ボードの電源供給のために必要な3.3Vに変換する。この種の分配電源変換は、時間多重回路の電源に対する高い要求によって必要となる。これに加えて、システムは、コントロール・ボード600及びCPUボード700(図20参照)を含む。現時点での好ましい実施例では、CPUボード700は、Themis Computerやその他入手可能なVMEバス対応のPower-PCプロセッサ・ボードである。他の同様なプロセッサボードを利用するこども可能である。使用する特定のプロセッサボードの選択は、コスト、速度、RAM容量及びその他の要因の間での相対関係に依存する。CPUボード700は、ネットワーク・インターフェースを提供し、エミュレーション・システムを総合的に制御する。コントロール・ボード600は、クロック分配、ダウンロード及び他のボードのテスト機能を提供するとともに、ロジック・アナライザ及びパターン・ジェネレータ(構造及び機能は以下で説明する)の集中化機能を提供する。

また、現時点において好ましいエミュレーション・システムの縮小バージョンを構成することもできる。このシステムのブロック図を図14に示す。縮小されたシステムは、スイッチング・バックプレーン420を持たない。その代わり、ロジック・ボード200は、互いに直接に接続され、またI/Oボード300に接続される。これは、システムのサイズが二対のロジック・ボード200及び一対のI/Oボード300に限定されたため、可能である。バックプレーン接続は、図14の上部に示される。各バックプレーン・コネクタ220からのピン(図11に示されている)は、4つの等しいグループに分割される。各グループは、バックプレーンを介して、同一の対に含まれない2つのロジック・ボード200の1つ及び各I/Oボード300にルーティングされる。接続は、ロジック・ボード200の場合にはターボ・コネクタ202を介して提供され、I/Oボード300の場合には接続を必要としないため、同一の対に含まれる他のロジック・ボード200又はI/Oボードに対してはバックプレーンを介して接続する必要はない。図14に示す接続パターンは、ボード間の十分に余裕のあるルータビリティ(routability)を提供するが、スイッチング・バックプレーン420が高価となるのを避ける。大規模なシステムでは、I/Oボードは、コア・ボード500及びリピータ・ポッド520を通じて外部システム540に接続される。図面の簡略化のために、第2のボードに対して、コア・ボード500、リピータ・ポッド520及び外部システム540が現実には存在するが、示していない。追加のボードの組を、そうしなければ使われることのないバックプレーン及びターボ・コネクタの間を接続するために使用することにより、小規模なシステムのバージョンは、1から4個のロジック・ボード200及び1個又は2個のI/Oボード300のいずれかによって構成することができる。これらの追加ボードは、図15に示されているターボ・ループバック・ボード260及びバックプレーン・ループバック・ボード280である。これらのボードのいずれもデジタル・ロジックを有していない。これらは、コネクタ間で信号をルーティングするに過ぎない。

1つのロジック・ボード及び1つのI/Oボードを有する小規模なシステムの物理的な図を図15に示す。バックプレーン802は、これまでに図14を参照しながら説明した接続を提供する。I/Oボードは、コネクタ330を介してコア・ボード500に接続される。コア・ボード500は、ケーブルを介してリピータ・ポッド520及び外部システム540(図15には図示していない)を装着する外部コネクタ510を有する。電源ボードは240、48VのDC主電源をロジック・ボードの電源供給のために必要な3.3Vに変換する。これに加え、小規模なシステムは、これまでに図13を参照して説明した大規模システムのように、コントロール・ボード600及びCPUボード700を含む。4個以下のロジック・ボードが使用される時、ルーティング接続を確保するために、ターボ・ループバック・ボード260は、ロジック・ボード200の使用されていないターボ・

コネクタ 202 (図 11 に示されている) からの信号をバック・プレーン 802 に接続する。ターボ・ループバック・ボード 260 は、システム内の 1つ又は 3つのロジック・ボード 200 が存在する場合に使用される。追加のバックプレーン・ループバック・ボード 280 の対は、使用されていないロジック・ボード・スロットがある時に、バックプレーンを介したルーティング接続を確保するために使用される。これは、システム内に 1つ又は 2つのロジック・ボードが存在する場合に発生する。バックプレーン・ループバック・ボード 280 は、バック・プレーン信号 (図 4 に示されている) のグループを互いに接続して、空のバックプレーン・コネクタが存在する時に信号が失われないようにする。

図 16 に、I/O ボード 300 及びコア・ボード 500 のブロック図を示す。マルチブレクサ・チップ 12 の第 1 列 301 は、I/O ボード 300 のバックプレーン・コネクタ 320 に装着される。図の簡略化のため、3つのマルチブレクサ・チップ 12 のみが第 1 列 301 に示されている。しかしながら、現時点における好ましい実施例では、最初の列 301 には 14 個のマルチブレクサ・チップ 12 が存在する。マルチブレクサ・チップ 12 の第 2 列 303 は、マルチブレクサ・チップ 12 の第 1 列に接続されるとともに、電界効果型トランジスタ (FET) 308 及びロジック・チップ 304 に接続される。ここでも、図面は、簡略化されて、2つのマルチブレクサ・チップ 12 だけが示されている。現時点における好ましい実施例では、第 2 列 303 には 12 個のマルチブレクサ・チップ 12 が存在する。マルチブレクサ・チップ 12 の 2つの列 301、302 は、ルーティングにおける十分な柔軟性を達成することが要求される。それにより、任意の外部信号を、コア・ボード 500 のリピータ・ケーブル・コネクタ 510 の任意のピンに接続することができる。ロジック・チップ 304 も、同期グラフィック RAM (SGRAM) 302 に装着される。現時点における好ましい実施例では、ロジック・チップ 304 は、FPGA である。1つのロジック・チップ 304 と 1つの SGRAM 302 のみが示されているが、現時点における好ましい実施例では、I/O ボード 300 には 6 個のロジック・チップ 304 及び 3 個の SGRAM 302 が存在する。ロジック・チップ 304 及び SGRAM 302 は、刺激ベクトル (stimulus vector) を外部接続ピン上のエミュレータにドライブする能力を提供する。刺激ベクトルをドライブする時には、FET 308 はターンオフされる (即ち、開放される)。これにより、刺激が、リピータ・ポッド 520 を介してコネクタ 510 に装着することのできる外部システムからの信号と競合を生じないようにする。刺激ベクトルをドライブしない時には、ロジック・チップ 304 のピンは、トライステートされ (tristate)、FET 308 がターンオンされる (即ち、閉じられる)。これにより、コネクタ 510 の信号は、第 2 列のマルチブレクサ・チップ 303 からの信号をドライブ又は受信することができる。現時点における好ましい実施例では、ロジック・チップ 304 は、Xilinx Corporation, San Jose, California から入手可能な XC5215 であるが、他のプログラム可能なロジック・チップを使用することによっても満足すべき結果を得ることができる。図 16 に示す部品に加えて、I/O ボード 300 は、VME インターフェースを通じてバックプレーン・コネクタ 320 に接続されるプロセッサ・チップ (図示していない) を含む。現時点における好ましい実施例では、このプロセッサ・チップは、IBM Corporation から入手可能な PowerPC 403GC であるが、他のマイクロプロセッサ・チップを用いることによっても満足すべき結果を得ることができる。プロセッサのチップは、プロセッサ・バス 310 を介してロジック・チップ 304 に装着される。プロセッサ・バス 310 は、刺激情報を SGRAM 302 にアップロードする働きをする。プロセッサは、診断機能のため、マルチブレクサ・チップ 12、ロジック・チップ 304 及び SGRAM 302 からの情報をアップロード及びダウンロードするために使用される。

コネクタ 330 は、コア・ボード 500 を I/O ボードに装着する。FET 308 から送られるロジック信号に加えて、このコネクタ 330 は、JTAG 信号を受信し、電気的に VME バスに接続される。JTAG 信号は、コネクタ 510 に差し込むことのできるリピーター・ポッド 520 をダウンロード及びテストするためのものである。現時点における好ましい実施例では、VME バスは、コア・ボードと共に使用されていない。しかしながら、VME バスは、コネクタ 330 に差し込むことができる他の種類のボードとともに使

10

20

30

40

50

用することが考えられる。例えば、大容量のメモリ・ボードをコネクタ330に差し込んで、RAM208に収納可能なメモリよりも大きなメモリをエミュレートする能力を提供することが考えられる(図11に示されている)。

図17を参照すると、マルチプレクサ・ボードのブロック図が示されている。マルチプレクサ・チップ12は、バックプレーン・コネクタ420に対して分配される形で装着される。図17の簡略化のために、4つのマルチプレクサ・チップ12のみが示されている。しかしながら、現時点における好ましい実施例では、実際には、マルチプレクサ・ボード400には7個のマルチプレクサ・チップが存在する。さらに、マルチプレクサ12への接続は、図17に示されている数よりはるかに多く存在する。これらの追加の接続は、図示されたものと同様に配置される。図17に示すマルチプレクサ・チップ12に加えて、マルチプレクサ・ボード400は、バックプレーン・コネクタ420に装着され、マルチプレクサ・チップ12にダウンロードし又はテストするJTAGインターフェース(図示していない)を含んでいる。

図17のマルチプレクサ・ボードは、拡張性のないエミュレーション・システムに適している。しかしながら、いくつかのエミュレーション・システムを互いに接続することによって、大容量のエミュレーション・システムを構成することが、しばしば要望される。この場合には、拡張性のあるマルチプレクサ・ボード400が使用される。拡張可能なマルチプレクサ・チップ・ボード402のブロック図を図18に示す。マルチプレクサ・チップ12の第1列404は、バックプレーン・コネクタ420に電気的に接続されている。

図面の簡略化のため、第1列404には、4つのマルチプレクサ・チップ12のみが示されている。しかしながら、現時点における好ましい実施例では、第1列404には、10個のマルチプレクサ・チップ12が存在する。マルチプレクサ・チップ12の第1列404は、マルチプレクサ・チップ12の第2列406及びターボ・コネクタ430に電気的に接続される。マルチプレクサ・チップ12の第2列406は、また、ターボ・コネクタ430及び外部コネクタ440に電気的に接続される。図18には、第2列には、2つのマルチプレクサ・チップ12のみが示されており、そして2つの外部コネクタ440のみが示されている。しかしながら、現時点における好ましい実施例では、第2列406には、5個のマルチプレクサ・チップ12が存在する。さらに、現時点における好ましい実施例では、6個の外部コネクタ440が存在する。現時点における好ましい実施例の各外部コネクタ440は、92個のI/Oピンを有している。マルチプレクサ・ボード402は、ターボ・コネクタ430を介して互いに結合された対に組み立てられる。ターボ・コネクタ430は、マルチプレクサ・ボード402の対とロジック・ボード200の対の間の有効な交差領域(intersection area)を拡張する役割を果たす。ターボ・コネクタ430が無い場合には、交差領域が小さくなりすぎて、外部コネクタ440とロジック・ボード200の間での有効なルーティング性能(routability)を確保することができない。

図19を参照すると、エミュレーション・システム内におけるユーザ・クロックの分配方法が示されている。ユーザ・クロックの分配は、エミュレーション・システムの設計において重要である。米国特許No.5,475,830号に開示されているように、ユーザ・クロックは、ユーザー・クロック及びデータ信号が外部システム540(外部システムは、図12及び14に示されている。)内で同時に変化すると仮定すると、データ信号の前にエミュレーション・ボード200のロジック・チップ10に到達する必要がある。この要求は、データ信号を遅延させることにより満足させることができる。しかしながら、この解決方法では、エミュレーション・システムの最大動作速度が低下する。より好ましい代替案は、ユーザ・クロックの分配ネットワークをできる限り高速とし、データ信号に遅延を付加する必要があるとしても、遅延を最小限とすることである。

図19に、好ましいハードウェア・エミュレーション・システムに対するクロックの分配を示す。クロックは、コントロール・ボード600のクロック・コネクタ620を介して、コントロール・ボード600のマルチ・ボックス・クロック・コネクタ630を介して、又はコア・ボード500のコネクタ510の通常の信号としてのいずれかによってシステムに入力することができる。前述したように、コア・ボード500は、I/Oボード3

10

20

30

40

50

00に装着されている。簡略のため、図19には、1つのコネクタのみが示されている。しかしながら、現時点における好ましい実施例では、各コア・ボード500には7つのコネクタ510が存在する。また、システムは、複数のI/Oボードとコア・ボードの組み合わせを含めてもよい。これまでに図12及び14を参照して説明したように、コネクタ510は、外部システム540に接続されるリピータ・ポッド520に装着されている。クロック・コネクタがクロックを入力するために使用される場合には、コネクタ620も、ケーブルを通じて外部システム540に装着されるであろう。クロック・コネクタ620は、クロックをエミュレーション・システムに入れるための速い方法を提供し、コア・ボード500のコネクタ510は、より簡単な方法をユーザーに提供する。

コア・ボード500のコネクタ510は、これまでに図16を参照して説明したように、コネクタ330及びFET308を介してI/Oボード300上のマルチプレクサ・チップ12の第2列に接続される。マルチプレクサ・チップ12の第2列303は、これまでに説明した他の接続に加えて、バックプレーン・コネクタ320の専用クロック・ピンに接続される。現時点における好ましい実施例では、これらのピンは16本存在する。I/Oボードのバックプレーン・コネクタ320から、クロックは、バックプレーン800又は802を介してコントロール・ボード(図13参照)に接続する。コントロール・ボード600では、マルチプレクサ・チップ12は、異なる可能性を有する全てのソースの中から1つのクロックの組み合わせを選択するために用いられる。システムは、最大32まで別個のクロック・ソースを有することができる。これらの中の任意の8つを、一対のエミュレーション・ボード200上で使用することができる。これにより、必要に応じ、例えば、複数のチップの設計が1つのハードウェア・エミュレーション・システム内でエミュレートされる場合には、エミュレーション・ボード200の異なる対が異なるクロックを持つことができる。クロックは、プログラム可能な遅延エレメント604及びバッファ614を介し、そしてバックプレーン800又は802を介してエミュレーション・ボード200にルーティングされる。これまでに図11を参照しながら説明したとおり、エミュレーション・ボード200のクロックは、ロジック・チップ10に到達する前に、バッファ216又はクロック生成ロジック・チップ204(即ち、Cosimロジック・チップ)のいずれを介してルーティングされてもよい。

コントロール・ボード600のロジック・アナライザのクロック生成ロジック・チップ602によりクロックを生成してもよい。これは、一般的には、システムをテスト・ベクトルで動作させる場合に起こる。クロックRAM612からのデータは、ロジック・アナライザのクロック生成ロジック・チップ602内にプログラムされ、リターン・トゥ・ゼロ(return-to-zero)、ノン・リターン・トゥ・ゼロ(non-return-to-zero)、2位相ノン・オーバーラッピング(two-phase non-overlapping)等の異なるクロック・パターンを生成するステート・マシン(state machine)に入力される。このようなステート・マシンの設計は、制御ロジック設計の当業者によって十分に理解することができるため、ここではこれ以上説明しない。ロジック・アナライザのクロック生成ロジック・チップ602から生成された32種のクロックは、クロック選択・マルチプレクサ・チップ12に通信される。現時点における好ましい実施例では、ロジック・アナライザのクロック生成ロジック・チップは、Xilinx Corporationによって製造されたXC4036XLを使用しているが、他のプログラム可能なロジック・デバイスを用いても満足できる結果を得ることができる。

マルチ・ボックス・クロック・コネクタ(multi-box clock connector)630は、入力クロックに対しても、出力クロックに対しても動作することができる。方向は、バッファ608によって制御される。マルチボックス・エミュレーション・システム、即ち、複数の孤立した(stand alone)エミュレーション・システムによって構成されるエミュレーション・システムでは、1つのボックスがマスターとして指定され、他のボックスはスレーブ(slave)として指定される。マスター・ボックスは、そのマルチ・ボックス・クロック・コネクタ630でクロックを生成し、次にこれがマルチボックス・クロック・コネクタ630を通じて全ての他のスレーブ・エミュレーション・システムに入力される。マルチ・ボックス・システムでは、マスター・ボックス内に遅延エレメント604がプログラム

10

20

30

40

50

され、マスタ・ボックスとスレーブ・ボックスの間での避けることのできないケーブルによる遅延を補償する。

当業者であれば、図19は、明瞭にするために大幅に簡略化されており、多くの相互接続及び構成部品が示されていないことを認識できる。これらの追加の構成部品及び相互接続に対する要求は、設計上の選択の問題である。

図20を参照すると、ハードウェア・エミュレーション・システムの制御構造が示されている。従来のハードウェア・エミュレーション・システムでは、一般的に不十分な処理能力が問題となっていた。このため、システムに又はシステムからデータを転送する時、設計データをシステムにロードする時、ハードウェア診断を動作させる時に、長い遅延を生じていた。本発明の好ましい実施例では、この問題を軽減するために、2つのレベルのプロセッサ・アーキテクチャが使用される。メイン・プロセッサ700は、コントロール・ボード600に装着される。現時点における好ましい実施例では、プロセッサ700は、Themis Computerから入手可能な、Power PCのVME準拠プロセッサ・カードであるが、他の類似のカードを用いても満足な結果を得ることができる。プロセッサ700は、イーサネット及びコントロール・ボード600のVMEバス650に電気的に接続される。VMEバス650は、インターフェース(図20には図示していない)を介してバックプレーン800又は802に電気的に接続され、次いでロジック・ボード200及びI/Oボード300に接続される。VMEバスは650は、また、コントロール・ボード600のJTAGインターフェース660及びバックプレーン800を介してマルチプレクサ・ボード400に接続される。

各ロジック・ボード200及びI/Oボード300は、VMEインターフェース及びメモリを備えたローカル・プロセッサを有している。類似する回路が各I/Oボード300に存在するが、ロジック・ボード200を参照しながら、この回路を論じる。プロセッサ206(これまでに図11に示した)は、VMEインターフェース222を通じてバックプレーン800又は802のVMEバス650に電気的に接続される。これは、また、コントローラ221にも電気的に接続される。好ましい実施例では、コントローラ221は、Xilinx Corporationの複数個のpXC5215 FPGAによって構成される。コントローラ221は、JTAGのテスト信号をロジック・ボード200の他の部品に供給する。これに加えて、フラッシュEEPROM224やダイナミックRAM226等のような種々の装置がプロセッサ206に接続される。プロセッサ206は、ボードレベルでの診断を実行する時、ロジック・チップ10内に構成データをロードする時、メモリ208及び210(これまでに図11に示した)に又はメモリ208及び210からデータを伝送する時に、独立して動作する。

図20aを参照しながら、好ましい実施例のシステムのロジック・アナライザ回路を詳細に論じる。ロジック・アナライザは、分配される。これは、ロジック・アナライザの部分が各ロジック・ボード200に含まれ、集中化された機能がコントロール・ボード600に含まれることを意味する。イベント、即ち、エミュレーションを受けている設計内の信号ステート(signal state)の組み合わせは、ロジック・ボード200のロジック・チップ10及び204の内部で生成される。これらは、一対に結合されて信号236上に出力され、そして特別なイベント・ロジック・チップ232(図20aでANDゲート232として示されている)で共にAND(論理積)される。この結果得られる結合イベント信号は、フリップ・フロップ230(簡略のため、図20aには2つのフリップ・フロップのみが示されている)によって8つの信号に分離される。分離されたイベント信号240は、次に、バックプレーン800又は802(図20aには示されていない)を介してコントロール・ボード600に送られ、そこで他のボード又は他のボックスからのイベントと再びANDゲート678(ロジック・チップの一部)でAND(論理積)される。コネクタ670は、他のエミュレーション・ボックスからのイベント情報を与えてもよい。最後のイベント信号は、トリガ条件及び条件付き読み込み条件(conditional acquisition condition)を計算し、またロジック・ボード200上のデータの読み込み(acquisition)を制御する読み込みイネーブル信号238を生成するためのコントロール・ボード

10

20

30

40

50

600上のトリガ・生成ロジック・チップに674に伝達される。トリガ生成ロジック・チップ674の出力は、バッファ671を介してコネクタ672に送出され、また遅延エレメント676を通る。遅延エレメント676の出力は、バッファ673によりバッファされ、バックプレーン800又は802を介してロジック・ボード200上のロジック・アナライザのメモリ・コントローラ234に送られる。コントロール・ボード600は、また、経路、機能テスト・クロック及び他のロジック・アナライザ/パターン生成信号を発生する。

図20bを参照すると、ロジック・アナライザのデータ経路が示されている。データ信号は、ロジック・チップ10及び204内にラッチされ、エミュレーション・ボード200の同期グラフィックRAM(SGRAM)210にスキャン出力される。ロジック・アナライザのデータ経路は、すべてのロジック・ボード200に渡って分配される。ロジック・ボード200の各マルチプレクサ・チップ12は、256K×32のSGRAMに接続された8つのピンを持っている。SGRAM210は、エミュレーションがロジック・アナライザのデータをセーブしている時に高速で動作する。データは、望ましいロジック解析速度、チャンネルの深さ及びプローブ信号(probed signal)の数に応じて、以下の表に示すように、2対1から64対1のいずれかによって時間多重化される。

#### ロジック・アナライザ・トレードオフ

最高速度	深さ	チャンネル/ロジック・ボード	時間多重因子
16MHz	128K	864	2-1
8MHz	64K	1,728	4-1
4MHz	32K	3,456	8-1
2MHz	16K	6,912	16-1
1MHz	8K	13,824	32-1
0.5MHz	4K	27,648 (全信号)	64-1

上で示した最高速度は概略値であり、ロジック・アナライザの設計及びマルチプレクスのクロック速度に応じて変化する。0.5MHzのレートでは、十分な数のチャンネルが利用できるため、同時に、エミュレートされる設計の各フリップ・フロップ又はラッチをプローブ(probe)することが可能である。信号が“プローブ”されると、そのエレメント又はノードにおける信号の値が読み取られる。一般的に、この値はメモリ・エレメント(SGRAM-210)に格納される。組み合わせ信号(combational signal)をソフトウェアで再構築することにより、ユーザは、信号の任意の組み合わせを、トリガ条件の前後の数千クロックにわたって、プローブ(probes)を移動させることなく又はエミュレータの再スタートをも必要とせずに見ることができる。組み合わせ信号をプローブする必要がある場合には、ソフトウェアは、設計のネットリストを検査する。ロジックのコーン(cone of logic)が抽出され、その中で意図する信号に続く各組み合わせロジック経路が後ろ向きに、プローブ格納エレメント(probed storage element)(例えばフリップ・フロップ又はラッチ)又は設計の外部入力までトレース(trace)される。意図する信号についてのロジック機能は、次に、すべての格納エレメント又はこれに与えられる外部入力から求められる。最後に、意図する信号の値は、全ての格納ノード及び外部入力について予めセーブされた値を用いてロジック機能を評価することによって、各瞬間毎に算出される。ロジック機能は、ロジック・コーンに対する入力の1つが変化する各時点において評価される。これは、設計デバッグ・ソフトウェアの一部として実行される。

例えば、図20dでは、プローブされた信号Eは、格納エレメントB、C、D及び設計入力Aで終了する組み合わせロジック・コーンを抽出することにより算出される。信号Eの式は、信号A、B、C、Dが変化したときに評価される。信号Eの波形は、物理的なプロ

10

20

30

40

50

ープが置かれているかのように、正確に表示される。この完全な視認性により、複雑な設計上の問題に対するデバッグが格段に高速化される。この完全な視認性は、ロジック・チップ10又は204あたりのフリップ・フロップの数が限定されている場合には、より高い周波数でも確保される。

より高い速度、即ち、0.5MHzより高い速度では、ユーザは、プローブする信号を指定しなければならない。しかしながら、各ロジック・ボード200は、それぞれ自身のロジック・アナライザ・メモリ210を有しているため、プローブされている信号の切り替えは速い。この理由は、従来のエミュレーション・システムのように、プローブがバックプレーンの上をルーティングされる必要がないことによる。

図20bを再び参照すると、各ロジック・チップ10又は204の内部には、追加のロジック回路2000が、ロジック・チップ10又は204内にプログラムされているユーザの設計に追加される。注文設計のロジック・チップが使用される場合には、このロジック回路2000をチップの中に設計（即ち、ハード・ワイヤ（hard-wired））することができる。専用スキャン・レジスタの数は、プローブすべき信号の数に応じて追加される。スキャン・レジスタは、スキャン・チェーン（scan chain）として知られている。各スキャン・フリップ・フロップ2004の間には、2対1のマルチプレクサ2005が配置されている。各マルチプレクサ2005の出力は、これに続くスキャン・フリップ・フロップ2004のD入力に与えられる。各マルチプレクサ2005への最初の入力は、ユーザの設計内ノードによって供給される。各マルチプレクサ2005への2番目の入力は、その前のスキャン・フリップ・フロップ2004によって供給される。マルチプレクサ2005への選択入力は、トレース・クロック2002であり、その機能は以下に説明する。スキャン・フリップ・フロップ2004は、多重クロック信号44によってクロックされる。一連のスキャン・フリップ・フロップ2004及びマルチプレクサ2005は、全体としてスキャン・レジスタ又はスキャン・チェーンを構成する。スキャン・チェーンの長さ及びプローブすべき信号の数に応じて、各ロジック・チップ10又は204は、0、1又は複数のスキャン・チェーンを有する。任意のチップ内のスキャン・チェーンの数は、プローブすべきフリップ・フロップ又信号の数に依存する。後に説明するように、ソフトウェアは、スキャン・チェーンに信号を割り当てるにより、チェーンの数を最小化し、チップのルーティングを簡素化する。好ましい実施例では、エミュレートされる設計内のすべてのフリップ・フロップ又はラッチをプローブするために、ロジック・チップ10又は204の1個当たり、最大12個のスキャン・チェーン及び12個のI/Oピンが必要である。ロジック・アナライザの動作速度を最大限に高めるため、スキャン・チェーン及びSGRAM210は、時間多重周波数の2倍の速度で動作する。時間多重クロックの各サイクルごとに、各スキャン出力ピン2006に1ビットのデータが出力される。

図20cを参照すると、ロジック・アナライザのイベントは、ロジック・ボード200の上に分配される。これにより、イベントに寄与する設計信号をバックプレーン800又は802にルーティングする必要がない。イベントは、ロジック・ボード200上の各ロジック・チップ10又は204に挿入された追加専用のロジック2000を用いて検出される。

イベントに寄与する信号（Signal Contributing to event）は、ロジック・アナライザのデータについて使用され、またこれまでに図20bに示した、同一のスキャン・フリップ・フロップ2004によってラッチされる。これらの信号は、CLBメモリ2018（CLBメモリは、ロジック・チップ10、204において利用可能）を構成するJTAGプログラマブル・エッジ検出器にルーティングされ、次いで、8つのイベント信号を形成するためにワイドなエッジ・デコーダ（wide edge decoder）2012を使って一緒にANDされる。各ロジック・チップ10、204内の8つのイベント信号は、マルチプレクサ2020を使って2つずつ1つのピンに組み合わされ、エミュレーション・ボードに対してイベント信号236（同じく図20aに示されている）として出力される。これらは、そこで他のFPGAからのイベント信号と再びANDされる。ボード・レベルのイ

10

20

30

40

50

イベント信号は、バックプレーンを渡ってコントロール・ボードに送信される。これらは、そこで他のエミュレーション・ボード又は他のボックスからのイベント信号とANDされる。結果として得られるシステムのワイドなイベント信号(wide event signal)は、コントロール・ボードのトリガ・ロジック・チップ674に送られ、読み取りイネーブル(acquisition enable)その他のロジック・アナライザ制御信号を生成するために利用される。

イベントに寄与する信号は、ワークステーションがエミュレーション・システムに接続されるのに先だってユーザに対して表示される形式を整理することにより、編集の前にエミュレーション・システムのユーザによって定義される。これが行われると、ロジック・チップ10、204内の十分に構成可能なロジック・ブロックは(configurable logic blocks)(CLB)(CLBは、ロジック・チップ10、204内の機能性を実現するために用いられる論理構築ブロック(logical building block)である)、全ての必要なイベント・ロジックを適合させるための編集処理の間予約される。容量にほとんど影響を与えることなく、任意の数の信号を予め定義することができる(信号あたり、概略4つのCLB)。コンパイルが全て完了した後でも、新しい信号を加えることが可能である。このためには、インクリメンタルな再編集を行って、再びダウン・ロードして追加のエッジ検出器を作成して新しい信号をルーティングする必要がある。イベントに寄与する信号がすべて定義されると、ユーザは、エミュレーションの実行中に素早くイベントの条件を変更することのできる総合的な柔軟性を得ることができる。ブレーキ・ポイント、トリガ条件及び条件付き読み取り条件は修正が可能であり、エミュレーションを停止することなくロジック・アナライザを再スタートさせることができる。これは、イベント・ロジックの設定にJTAGプログラミングを用いることによって可能となる。

図20cに、すべてのイベント及びスキャン・ロジックを挿入したロジック・チップ10又は204を示す。設計は、スキャン・フリップ・フロップ2004及びマルチプレクサ2005によって構成されるスキャン・レジスタと、フリップ・フロップ2010、JTAGインターフェース2016及び2014、1組のエッジ検出器2018及びワイドなエッジ・デコーダ2012により構成されるイベント・レジスタとに分割される。

イベント信号は、ロジック・アナライザのデータがシフト出力される時に内容が変化するため、スキャン・フリップ・フロップ2004内にはセーブすることができない。このため、イベント・フリップ・フロップ2010を使って、イベントに寄与するすべての信号についての現在及び過去の状態(state)を記憶する。イベント・レジスタ2010は、スキャン・レジスタ2004がトレース・クロック信号2002(以下で説明する)によってロードされた後の次のスキャン・クロックによって1回クロックされる。代わりに、スキャン・レジスタ2004は、平行なシャドウ・レジスタ(parallel shadow register)として使用することができ、トライステート・バッファは、スキャン・データをスキャン出力ピンにロードするために使用することができるであろう。

イベント・フリップ・フロップ2010からの出力は、エッジ検出器2018への入力として使用される。エッジ検出器2018は、デュアル・ポートCLBメモリにより構成される。各CLBメモリは、ロードされて、2つの入力信号に対して所望のレベル/エッジ検出を行い、1つのイベント出力を生成する。1つのイベントに属するすべてのCLBメモリからの出力は、組み付けのワイドなデコーダ2012を使ってANDされ、このロジック・チップ10についての1つのイベント信号を形成する。次に、イベント信号はマルチプレクサ2020を使って結合され、I/Oピンの場所のトライステート・バッファ2022に出力される。ユーザの信号がいずれかのイベントで必要とされるたびに、8つのイベントのすべてについてこれが添付され、イベントの定義を稼動中に変更することができるようになっている。

エッジ検出器2018の内部で使用されるCLBメモリは、JTAGバス上でプログラムされる。これは、好ましい実施例のロジック・チップ10、204のデュアル・ポート・メモリの性質を使ってカウンタ2016及びデコーダ2014によって実現される。イベント回路の数が多い場合には、デコーダ2014からの選択信号の作成及びルーティング

に、ロジック・チップ10のゲート容量のかなりの部分を費やすことになる。代わりに、すべてのエッジ検出器メモリ2018を含むシフト・レジスタを作成する方法がある。しかしながら、この代替案では、ランダムなアクセスを行うことができない。

イベントに寄与する各信号は、概略4つのCLBとJTAGインターフェースのための小さなオーバーヘッドを必要とする。信号が追加されるたびに、8つのイベントのすべて又は一部として使用できるように必要なロジックが挿入されることが想定される。ユーザーが、どのイベントについてその信号が使用されるのかを正確に指定した場合には、CLBの2分の1が必要とされるに過ぎない。しかしながら、この方法では、エミュレーションの実行中にイベントの条件に変更を加える能力を著しく制限することとなる。

各信号 / イベントの組み合わせに対するエッジ検出メモリ2018は、次の条件のうちの1つを検出するようにプログラムされる。

#### イベント条件

式	二進ノック	説明
A = 0	0	0 レベル
A = 1	1	1 レベル
A = 0 & B = 1	F	立下りエッジ
A = 1 & B = 0	R	立ち上がりエッジ
A xor B	E	すべてのエッジ
A = 0 & B = 0	S0	0 で安定
A = 1 & B = 1	S1	1 で安定
A xor B	S	1 又は 0 で安定
0	—	信号を使用しない

ロジック・アナライザのサイクルは、トレース・クロック信号2002によってスタートする。トレース・クロック2002は、厳密に制御された信号ではない。これは、多重クロック信号(MUXCLK)の立ち上がりエッジでのみ有効であることが保証されている。トレース・クロック2002は、データの同期サンプルをすべてのスキャン・チェーンにおいてセーブさせる。これは、また、イベントの計算を開始する。ボード・レベルのイベントは、コントロール・モジュール600に送出され、そこで、一緒にANDされ、トリガ生成・ステート・マシン674(trigger generator state machine)を制御するために使用される。いくつかのトレース・クロック期間の後に、トリガ生成器は、ロジック・ボード200のSGRAM210へのデータの書き込みを制御する読み取りイネーブル信号(acquire enable signal)238を生成する。そして、回路は、次のトレース・クロック2002まで非アクティブを保持する。

ロジック・アナライザのデータは、各エミュレーション・ボードのRAM内に格納される。これまでに述べたように、各ロジック・ボード200は、54個のマルチプレクサ・チップ12を有している。また、各マルチプレクサ・チップは、SGRAM210に接続される8本のピンを有している。このため、RAMには $54 \times 8 = 432$ のデータ・チャンネルが存在する。ロジック・アナライザのデータは、フレームと呼ばれる基本単位に格納される。1つのフレームは、各トレース・クロック2002の後に生成され、ロジック・チップ10又は204のスキャン・チェーンから一度シフト出力(shift out)されたすべてのデータにより構成される。フレームは、2から64までのRAM位置を満たし、また生成のために2から64の多重クロック信号(MUXCLK)を必要とする。典型的なフレームは、以下のようなものである。

10

20

30

40

50

データ・チャンネル(432)

フレーム0 データ0

データ1

データ2

データ3

フレーム1 データ0

データ1

データ2

データ3

最小のフレームでは、2つのRAM位置だけを必要とする。フレームの長さは常に2の倍数である。このため、有効な長さは、2、4、8、・・・64のRAM位置である。SGRAM210のタイミング要求を満たすために、1つのフレーム内のシーケンシャルな書き込みは、メモリの反対側のバンクに行われる。最小サイズのフレームの場合、データの1つのワードが低いRAMのバンクに格納され、また1つのワードが高いRAMのバンクに格納される。

ロジック・ボードのメモリは、256Kワードの容量(deep)を有している。メモリは、32の独立したブロックに等しく分割される。各ブロックは、8192のワードの大きさであり、フレームの長さに応じて4096から128のフレームを含むことができる。ブロックは、固定した長さを有しており、常に8Kワードの境界から開始する。1つのブロック内では、フレームは、ランダムな順序で格納されるが、ブロック間でのフレームの重なりは発生しない。後のブロックにある全てのフレームは、前のブロックにある全てのフレームよりも新しいタイムスタンプ値(timestamp value)を有している。

ロジック・ボードのメモリ210の深さは、設計者の選択及び利用可

ロジック・ボードのメモリ210の深さは、設計者の選択及び利用可能なメモリ・チップの深さに依存する。将来的には、より深いメモリが使用可能となり、より大きなSGRAMが利用可能となろう。

タイムスタンプ値は、フレームがロジック・ボード200にセーブされる毎に、コントロール・ボード600のクロックRAM612(図19に示されている)にセーブされる。ロジック・アナライザは、条件付き読み取りのオプションをサポートする。これは、個々のフレームは、1つのイベント信号の値及び/又はトリガ・ステート・マシンの現在の状態(current state)に応じてメモリに書き込まれ又は書き込まれないということを意味している。条件付き読み取りによって、重要なデータだけをセーブすることができるため、メモリをより効率的に利用することができる。条件付き読み取りは、コントロール・ボード600で生成される読み取りイネーブル信号238によって制御される。トレース・クロック2002の後で読み取りイネーブル信号が生成されるまでに、概略トレース・クロック4個のパイプライン遅延(pipeline delay)が存在する。

遅延された読み取りイネーブル信号のために、データが利用可能になった時点では、そのデータをセーブすべきであるか否かを決定することができない。このため、常にデータをメモリにセーブし、遅延された読み取りイネーブル信号が、それが正しい信号でないことを示している場合には、後で上書きする。これにより、データは、基本的にメモリ内にランダムな順序でセーブされる。正しいデータの順序は、ロジック・アナライザが停止した後に、クロックRAM612にセーブされたタイムスタンプをソートし、そして各ロジック・ボード・プロセッサ206に対する1組のポインタをソートすることによって回復される。ポインタは、各シーケンシャル・データ・サンプルの物理的なメモリ位置を示す。メモリ・アドレス・カウンタの重ね合わせ(wraparound)を処理する必要があるため、規定外(out-of-order)のデータは、メモリの1つのブロックに限定される。アドレス・カウンタがブロックの先頭位置に再び書き込みを行うと、直ちに最も古いブロックのデータを破棄する必要がある。

コントロール・ボードのロジック・アナライザ・コントロール・ロジック・チには、各ブロックが書き込まれた後に、データの5ビットがセーブされる(合計160ビット)。こ

10

20

30

40

50

これらのビットのうち 4 ビットは、最後に書き込まれた 4 つのフレームの各々についての読み取りイネーブル信号の値である。特定の 1 ビットは、そのブロックがソートされた順に書き込まれたかどうかを示す。これは、実質的に、ブロック中の各トレース・クロックについての読み取りイネーブルが有効であったことを示す。

ブロックがオーバーラップされないようにするために、読み取りイネーブル信号の状態に関係なく、各ブロック内の最後の 4 つのフレームは常に書き込まれ。これらの最後の 4 つのフレームは、有効なデータを含む場合と含まない場合がある。コントロール・モジュールのプロセッサは、ブロック・レジスタ内の対応する読み取りイネーブル・ビットを調べて、データが有効であるか否かを判断する。このため、1 つのブロック内の実際のデータ・フレームの数は、4 つまでの誤差がある。10

エミュレーション・ボードについてのポインタのセットを作成する場合には、これを考慮する必要がある。ロジック・アナライザが停止する前の最後の 4 つのデータのワードは、有効なデータを含んでいる場合と含んでいない場合とがある。これは、ロジック・アナライザが停止した後、読み取りイネーブル・パイプラインをブロック・レジスタに流し込むことによって決定される。

コントロール・モジュール・プロセッサ 700 は、ロジック・アナライザが停止する前に最後に格納したフレームのアドレスを、ロジック・アナライザのコントロール・チップ 674 から読み取ることができる。これは、最後に書き込まれたデータ・ブロックを決定するために使用される。最初のデータ・ブロックは、アドレス・カウンタがオーバー・ランしていない場合にはブロック 0 であり、そうでない場合には次の高いブロックである。アドレス・カウンタが初めてオーバーランする時にセットされる追加のステータス・ビットが必要である。20

ロジック・アナライザが停止した時に書き込まれている最後のデータ・ブロックは、多分、アドレス・カウンタのその前のラップアラウンドの間に書き込まれたいくつかの古いフレームを含んでいるであろう。これらは、破棄する必要がある。破棄すべきフレームは、タイムスタンプ値をソートし、そして最初のデータ・ブロック内の最も古いタイムスタンプ値より前のタイムスタンプを有するフレームを破棄することによって決定される。

例えば、フレーム長が 1 (2 から 64 ではなく) であると仮定すると、ブロック当り 8 つのフレーム (4096 ではない) が存在し、メモリは 24 の深さ (262, 144 ではない) を有することになる。ロジック・ボード及びコントロール・ボードのメモリは、ロジック・アナライザが停止した後では以下のデータを有する可能性がある。30

	<u>ロジック・ボード</u>	<u>制御ボード</u>	<u>ブロック・レジスタ</u>	
<u>アドレス</u>	<u>データ・メモリ</u>	<u>タイムスタンプ</u>	<u>ソート</u>	<u>読み取りイネーブル</u>
<u>イネーブル</u>				
0	2 8	4 3	0	0 1 1 1
1	1 8	4 7		
2 <-カウンタ	1 7	4 5		
3	9 2	4		10
4	9 3	5		
5	9 4	6		
6	9 5	7		
7	9 6	8		
8	3	1 3	0	0 1 0 1
9	1	1 0		20
1 0	2	1 2		
1 1	5	2 7		
1 2	7	2 9		
1 3	1 4	3 0		
1 4	8	3 1		
1 5	2 7	3 2		
1 6	3	3 3 -	1	0 1 1 1
1 7	9	3 4		
1 8	1 0	3 5		
1 9	1 1	3 7		
2 0	1 2	3 9		
2 1	1 3	4 0		
2 2	1 4	4 1		40
2 3	1 7	4 2		

アドレス・オーバーフロー・ビット=1

アドレス・カウンタは位置 2 で停止しており、アドレス・オーバーフロー・ビットがセットされている。これは、位置 0 から 7 のブロックが最後のブロックであり、8 から 1 5 のブロックが最初のブロックであることを意味する。最初のブロックに対して格納された読み取りイネーブル・ブロックを調べることによって、最初のブロックの終わりの部分の位置 1 2 及び 1 4 にあるフレームは有効であり、最初のブロックの終わりの部分の位置 1 3 及び 1 5 にあるフレームは無効であることを決定する。このブロック内の全ての他のフレームは有効である。それは、もしそうでなければ、アドレス・カウンタは、次のブロック

にインクリメントされていないはずだからである。タイムスタンプによってソートし、そして無効なデータを除去した後は、最初のブロックは以下のようになる。

	エミュレーション・ボード	コントロール・ボード
<u>アドレス</u>	<u>データ・メモリ</u>	<u>タイムスタンプ</u>
9	1	1 0
1 0	2	1 2
8	3	1 3
1 1	5	2 7
1 2	7	2 9
1 4	8	3 1

注意：あるブロック内の最後の4つのフレームは常にソートされた順序となっているため、タイムスタンプによるソートの前後で、無効なフレームが除去されている場合がある。

次に、2番目のブロックが処理される。アドレス16から開始する2番目のブロックでは、アドレス23のフレームが無効である。このブロックに対してソートされたビットはブロック・レジスタにセットされているため、このブロックは、ソートする必要がない。無効なフレームを除去した後、ブロックは、以下のようになる。

	エミュレーション・ボード	コントロール・ボード
<u>アドレス</u>	<u>データ・メモリ</u>	<u>タイムスタンプ</u>
1 6	3	3 3
1 7	9	3 4
1 8	1 0	3 5
1 9	1 1	3 7
2 0	1 2	3 9
2 1	1 3	4 0
2 2	1 4	4 1

最後のブロックは、アドレス0から開始する処理が実行される。まず、フレームは、タイムスタンプによってソートされて以下のようになる。

	<u>エミュレーション・ボード</u>	<u>コントロール・ボード</u>
<u>アドレス</u>	<u>データ・メモリ</u>	<u>タイムスタンプ</u>
3	9 2	4
4	9 3	5
5	9 4	6
6	9 5	7
7	9 6	8
0	2 8	4 3
2	1 7	4 5
1	1 8	4 7

次に、最初のタイムスタンプより前のタイムスタンプを有する全てのフレームで最初のブロック（10）にあるものは破棄される。これにより、そのブロックには3つのフレームだけが残される。

	<u>エミュレーション・ボード</u>	<u>コントロール・ボード</u>
<u>アドレス</u>	<u>データ・メモリ</u>	<u>タイムスタンプ</u>
0	2 8	4 3
2	1 7	4 5
1	1 8	4 7

最後のフレームについてのブロック・レジスタの読み取りイネーブル・ビットは、読み取りイネーブル・パイプラインからの最後の値を含んでいる。このブロックについてのレジスタの内容は0111である。これは、アドレス1にある最後のフレームが無効であり、アドレス0及び2にある他の2つのフレームは有効であることを意味する。このブロックには3つのフレームしか書き込まれていないため、低いオーダー（order）のビットは意味がない。最後のブロックは以下のようになる。

	<u>エミュレーション・ボード</u>	<u>コントロール・ボード</u>
<u>アドレス</u>	<u>データ・メモリ</u>	<u>タイムスタンプ</u>
0	2 8	4 3
2	1 7	4 5

再生されたデータの完全なセットは、以下のようになる。

<u>アドレス</u>	<u>エミュレーション・ボード</u> <u>データ・メモリ</u>	<u>コントロール・ボード</u> <u>タイムスタンプ</u>
9	1	1 0
1 0	2	1 2
8	3	1 3
1 1	5	2 7
1 2	7	2 9
1 4	8	3 1
1 6	3	3 3
1 7	9	3 4
1 8	1 0	3 5
1 9	1 1	3 7
2 0	1 2	3 9
2 1	1 3	4 0
2 2	1 4	4 1
0	2 8	4 3
2	1 7	4 5

好ましい実施例のシステムをプログラムするために必要なソフトウェアを以下に説明する。ソフトウェアは、アップデートされている。そのため、これまでに米国特許No. 5,109,353号、5,036,473号、5,448,496号、5,452,231号、5,475,830号に開示されているソフトウェアとは異なっている。これらの特許の開示を、参考のためにすべてここに組み込む。フロー図を図21に示す。

ソースのネットリストは、ネットリスト・インポータ1000によって直接にインポート(import)することができ、Quickturn Design Systems, Inc. から入手可能なHDL-ICE™ブランドのロジック合成ソフトウェアのようなロジック合成プログラム1002又は動作テストベンチ・コンパイラー(behavioral testbench compiler)1004によって生成することができる。ネットリスト・インポータ1000は、ゲート・レベルのテキスト形式のネットリストをEDIFやVerilogのような種々のフォーマットに取り入れができる。また、ネットリストを、階層的に定義されたセル、一般的なセル、特別なハードウェア・セルを含むデータ・ベース・ロジカル・ライブラリ(data base logical library)によって表現する内部データベース・ネットリスト・フォーマットに変換することができる。特別なハードウェア・セルには、メモリ仕様セル(memory specification cells)、マイクロプロセッサ・セル、部品アダプタ・セルが含まれる。階層的に定義されたセルの一部には、チップ間をルーティングする時のタイミング問題を避けるために、これらのロジック・チップの間で平坦化されまた分割されるのを防止するフラグを持っている。ネットリストのインポート・ソフトウェアの選択と設計は、設計上の選択の問題であり、これ以上論じない。前述したように平坦化されたセルは、階層構造のセルを含んでいない。これは、簡単なロジック・ゲートのような最も基本的な部品のみを含む。

HDL-ICE™ブランドのロジック・シンセサイザ1002は、現時点における好ましいロジック・シンセサイザ1002であり、レジスタ転送レベル(register-transfer-level)50

( RTL ) の Verilog 又は VDHL ネットリストを受け取り、それらをロジック合成プロセスを通じてネットリスト・インポータ又は他のコンパイル・ステップ ( compilation step ) で使用されるデータベース・フォーマットに変換する。他の好適な合成のための製品が Synopsys Corporation 及びその他から商業的に入手可能であるが、HDL-ICE™ ブランドのロジック・シンセサイザは、より良い集積化やより高い動作速度などのいくつかの利点を有している。

動作テストベンチ・コンパイラ 1004 は、Verilog 又は VHDL で記述された動作テストベンチをエミュレートすることができる。1 又は複数のロジック・ボード 200 のプロセッサ 206 で並列に実行されるコードは、共同ミュレーションのロジック・チップ 204 を通じて、ネットリスト・インポート・プログラム 1000 又は HDL-ICE™ ブランドのロジック・シンセサイザに対応可能な他のロジックチップと緊密に結合される。プロセッサ 206 で実行中のコードは、ロジック設計の動作（合成不可能）表現であるかもしれないが、他のロジックは、ゲート・レベルの（合成可能）RTL 表現である。

ロジック・セル・メモリ (LCM) ・ジェネレータ 1006 は、ロジック・チップ内に組み込まれたメモリを使用するように供給されるユーザ設計からのメモリ仕様セルを、ロジック・チップ 10 内の構成可能なロジック・ブロック及び各ロジック・チップ内の相対的な位置へのマッピングを含むメモリ・セルの供給を定義する階層的に定義されたセル（ハード・マクロ）に置き換える。

ユーザ・データ入力プログラム 1008 により、設計のコンパイルに必要な、クロック情報、プローブ情報、特別のネット情報等を入力することができる。この情報は、特別な方法で処理しない時、エミュレーション中に問題を起こす可能性のある任意の条件を処理する際にエミュレーション・システムを支援する。

データ・クオリフィケーション・プログラム (data qualification program) 1010 は、ネットリスト及びユーザ・データが正確であるか否かを確認する。これは、ドライブされない入力又は複数の出力がネットに割り当てられている場合などの一般的なネットリストのエラーを検出する。

クロック・ツリー抽出プログラム 1012 は、階層的なネットリストからクロックのツリーを抽出し、設計階層の全てのレベルのクロックの末端 (terminal) を特定する。このステップの動作は、米国特許 No. 5,475,830 号に詳細に開示されている。

階層的パーティション計画プログラム (Hierarchical partition planning program) (HPP) 1014 は、物理的なモジュール・チップのパーティショニングのアルゴリズムである。これは、各ロジック・ボード 200 に割当てられる (map) 設計の部分を特定する。

パーティション DB セットアップ 1016 は、HPP 1014 によって特定された各部分について、チップ・パーティショニング・プログラムを並列に動作させるためのデータベースを準備する。

チップ・パーティショニング・プログラム 1018 は、別個のロジック・チップ 10 の各々において実現されるロジックのクラスタ (cluster) を特定する。

NGD 出力プログラム 1020 は、チップの分割 (パーティショニング) の結果に基づいて、各チップに対応する NGD ファイルを作成する。NGD は、Xilinx Corporation から入手可能な種々のソフトウェア・プログラムに共通するファイル・フォーマットである。NGD ファイルは、ロジック・チップ内にロジック設計を組み込むために必要なロジック及びルーティングの情報を含む。前述したように、現時点における好ましい実施例では、Xilinx のロジック・チップが用いられている。NGD 出力プログラム 1020 は、データ・ベースの情報を NGD フォーマットに翻訳する。また、NGD 出力プログラムは、任意のピン割り当てがなされた個々のロジック・チップ 10 に対して、並列パーティショニング、配置及びルーティング (PPR) ジョブ 1022 を開始する。PPR プログラム 1022 は、Xilinx Corporation から商業的に入手可能なプログラムであり、Xilinx が製造する FPGA についてのプログラミング・ファイルを生成する。

物理的 DB 生成プログラム 1026 は、ボード・パーティショニング・プログラムで用い

10

20

30

40

50

られる物理的データ・ベースを準備する。物理的データ・ベースは、システム中の各ボードのロジック・チップ10及びマルチプレクサ・チップ12の間での物理的接続についての情報を含む。

ボード・パーティショニング・プログラム1028は、各ロジック・ボード内のロジック・チップ10へのロジック・ゲートの配置を特定する。これは、各ロジック・ボード200で実現可能なメモリ・インスタンス(memory instance)の制約、ロジック・アナライザのプローブ・チャンネルの制約、ボード当りのマイクロプロセッサの制約とともに、バックプレーン及びターボ・コネクタの制約を考慮する。

E B Mコンパイル・プログラム1030は、同一のロジック・ボード200に割り当てられたメモリ仕様セルをR A M 2 0 8(これまでに図11に示されている)に対応する最大12のグループに結合する。S R A Mチップ208に接続するI / O信号208は、対応するピン番号のマークが付されている。

システム・ルーティング・モジュール1032は、物理的なネット及び時分割多重(TDM)位相を選択して、チップの境界(boundary)を超えた論理的なネットを実現する。これは、ピン番号及びT D M位相を全てのチップI / Oピンに割り当てる。これは、また、マルチプレクサ・チップ12及びリピータ・ポッド520のプログラミング・データを生成する。

N G D更新プログラム1034は、各ロジック・チップに対する最終的なインクリメンタルP P Rジョブ1036を開始し、T D Mロジック及びI / O割り当ての最終的な接続性を提供する。ジョブが正常に完了すると、コンパイルは終了する。

種々のプログラムの機能性の詳細をさらに説明する。

図22を参照すると、動作テストベンチ・コンパイラ1004によって作成されるソフトウェア・ハードウェア・モデルのコンパイルに必要なステップのシーケンスが示されている。コンパイルは、V e l i o g又はV H D Lのユーザのソース・コードから開始する。インポート・プロセス1100の結果として、動作データベース表現1102が作成される。モデルのコンパイルが終了すると、エミュレーション・モデル1114の論理表現及びこれまでに図20に示したロジック・モジュール・プロセッサD R A M 2 2 6にダウンロード可能な実行可能形式(executable)1112のセットが生成される。

動作テストベンチ・コンパイラ・ソフトウェア1004は、4つの実行可能形式及び1つのランタイム・サポート・ライブラリ(runtime support library)を含む。

インポータ1100は、ユーザのVerilog又はVHDLソース・ファイルを処理し、動作データ・ベース・ライブラリ1102を作成する。これは、ソース・ファイル名、位置及びライブラリのファイル名のリストを受け入れ、未定義のモジュール・リファレンスの問題を解決する。ソース・ファイル名は、V e l i o g又はV H D Lで使用されるファイル名である。

プリプロセッサ(preprocessor)1104は、インポータ1100によって作成された動作データベース・ライブラリ1102を新しい動作データベース・ライブラリ1106に変換する。これは、動作コードを各利用可能なプロセッサ206での実行を目的とするクラスタ(パーティションとも呼ばれる)にパーティショニング(分割)し、コード・ragment(code fragment)の実行順序及びパーティションにおける変数の場所を決定する。コード・fragmentは、プロセッサ206で並列に実行できる独立したコードの片(piece)をいう。また、プリプロセッサは、ホールド・タイム違反がないモデルを作成するために必要なすべての変換を行う。例えば、Priceらによる米国特許N o . 5 , 2 5 9 , 0 0 6号を参照されたい。参考のために、これによる開示をここに完全な形で組み込む。

コード・ジェネレータ1110は、プリプロセッサ1104で変換された動作データベース・ライブラリ1106を読み込み、プリプロセッサ1104によって特定される各クラスタに対してダウンロード可能な実行可能形式を生成する。これらの実行可能形式は、プロセッサ206の実行のためD R A M 2 2 6にダウンロードされる。

ネットリスト・ジェネレータ1108は、プリプロセッサ1104によって変換された動作データベース・ライブラリを読み込み、他のコンパイラ・プログラム1006～103

10

20

30

40

50

6によるさらなる処理のために論理的データベース・ライブラリ 1 1 1 4 を生成する。共同シミュレーション・ロジック・チップ 2 0 4 のマイクロプロセッサのバス及びイベント同期バス(図 1 1 参照)に対する特別な接続を表現するために、ネットリスト・ジェネレータ 1 1 0 8 は、図 2 3 に示すようなネットリスト構造を作成する。MP セル 1 2 0 0 は、プロセッサ 2 0 6 に対応する特別なセルであり、パーティショニング・プログラム 1 0 1 8 (LBM セル・インスタンスに類似する) によってはクラスタに分割されない。ペリフェラル・コントローラ・セル 1 2 0 2 は、ライブラリ・コンポーネントのインスタンスを含む通常のセルであり、共同シミュレーション・ロジック・チップ 2 0 4 内に配置される。マイクロプロセッサのバスと直接にやりとりを行う、このセル 1 2 0 2 には、最小限のロジックのみが配置される。ペリフェラル・コントロール・セル 1 2 0 2 内に最小限のロジックを配置することにより、待ち状態のプログラミング(wait state programming)の必要性がなくなる。ペリフェラル・コントロール・セル 1 2 0 2 は、チップ・パーティショニング・プログラム 1 0 1 8 がそれをいくつかのロジック・チップ 1 0 に分割するのを防止するためのフラグが付されている。このセルの容量が 1 つのロジック・チップ 2 0 4 を超えないように、またこのセルとネットリストの他の部分の間の接続の数が共同シミュレーション・ロジック・チップ 2 0 4 及びマルチプレクサ・チップ 1 2 の間の接続の数を超えないように、ネットリスト・ジェネレータ 1 1 0 8 の責任によって管理される。これまでに開示したように、共同シミュレーション・ロジック・チップ 2 0 4 は、54 個のマルチプレクサ・チップ 1 2 の各々と電気的な通信を行う 3 つのピンを持っている。これは、図 1 1 に示すように、共同シミュレーション・ロジック・チップ 2 0 4 とマルチプレクサ・チップ 1 2 との間に 1 6 2 ( $3 \times 54 = 162$ ) の接続が可能であることを意味する。ネットリスト・ジェネレータ 1 1 0 8 は、また、特別なネットをマーク(mark)する。このネットは、共同シミュレーション・ロジック・チップ 2 0 4 に対する正しい物理的接続を生成するようにシステム・ルータ(system router)をガイドする、対応するピン番号を有する MP セル 1 2 0 0 に接続する。これは、プロセッサ 2 0 6 と共同シミュレーション・ロジック・チップ 2 0 4 との間の接続がロジック・チップ 2 0 4 の特定のピンに接続されるために必要である。

動作テストベンチ・コンパイラ 1 0 0 4 は、Sample らによる、名称が「Method And Apparatus For Design Verification Using Emulation And Simulation」である、米国特許第 5,841,967 号に十分に開示されている。参考のために、これをここに完全な形で組み込む。

ロジック・チップ・メモリ(LCM)・ジェネレータ 1 0 0 6 は、Xilinx の relationally placed macros(rpm's) を用いて、浅いが高度にポートされた(highly ported) メモリを実現する。これは、最大 14までの書き込みポート、任意の数の読み取りポート及びデバッグ時のアクセスのための追加の 1 つの読み取り / 書き込みポートをサポートする。これは、ロジック・チップ 1 0 の一部として利用可能な同期型のデュアル・ポート RAM の基本要素(primitives)を使用している。

図 2 2 a に、ロジック・チップ 1 0 内に配置するために LCM メモリ・ジェネレータ 1 0 0 6 によって生成されるメモリ回路の例を示す。図 2 2 a のメモリ回路は、以下の部品によって構成される。

書き込みイネーブルのサンプラー及びアービトレータ(arbitrator) 1 0 5 0 は、書き込みイネーブル信号を高速なクロックに同期化する。また、いくつかのポートからのリクエストが同時に発生した場合に、メモリ回路の書き込み動作の優先順位を決定する。書き込みイネーブル・サンプラー及びアービトレータ 1 0 5 0 は、書き込みアドレス / データ、マルチプレクサ選択及び書き込みイネーブル信号を出力する。書き込みイネーブル・サンプラー及びアービトレータのセルは、2 から 16 までの種々の書き込みポート構造を有するハード・マクロ(hard macros) の形式でリファレンス・ライブラリにプリコンパイル(prec ompile) されている。

図 2 2 a のメモリ回路は、また、読み取りカウンタ 1 0 5 2 を有する。読み取りカウンタ 1 0 5 2 は、実現されるメモリの読み取りポートを循環(cycle)するために使用される

。これらのカウンタは、また、種々のカウント長を持ったハード・マクロ・セルとしてリファレンス・ライブラリにプリコンパイルされる。

図22aのメモリ回路は、また、読み取りカウンタ1052又は書き込みイネーブル・サンプラー及びアービトレータ1050のいずれかをその出力に配置するマルチプレクサ1053を有している。マルチプレクサ1053の出力は、スロット選択信号SLOT\_SELであり、16のスロット（又はポート）のうちの任意の1つを選択することができる4つのワイヤによって構成されている。

図22aのメモリ回路は、また、アドレス・マルチプレクサ及びデータ・マルチプレクサ1056を有する。アドレス・マルチプレクサ及びデータ・マルチプレクサ1056は、適当なスロット又はポート時間になると、ポートの書き込み／読み取りのアドレス・データ及びポートの書き込みデータを選択するために使用される。スロット選択信号SLOT\_SELは、この機能を実行するために、アドレス・マルチプレクサ及びデータ・マルチプレクサ1056の選択入力に入力される。  
10

図22aのメモリ回路は、また、メモリ1058を有する。メモリ1058は、1つ又は複数のXilinxの、構成可能なロジック・ブロック（CLB）部分として利用可能なスタティックRAMメモリである。

図22aのメモリ回路は、また、読み取りスロット・デコーダ1054を有する。読み取りスロット・デコーダ1054は、スロット選択信号SLOT\_SELを（ここでは4つ存在する）を最大16の別個のワイヤにデコードする。ワイヤは、出力レジスタ1060に対するクロック・イネーブル入力として使用される。  
20

図21を参照すると、LCMメモリ生成プログラム1006によって生成されるポートの幅、深さ及び数は、ネットリスト・インポート・プログラム1000、HDL-ICE<sup>TM</sup>ブランドの合成プログラム1002又は動作テストベンチ・プログラム1004によって生成されるネットリストの要求に依存する。Xilinxのrelationally placed macros（RPMS）は、一般的なセル・インスタンス及びロジック・チップ10の特定のロジック・モジュールへのメモリ回路のマッピングを制御する特別なFMAPとHMAPのセルのインスタンスによって定義されたデータベース・セルを用いて作成される。FMAP及びHMAPのセルは、Xilinx PPRプログラム1022の動作を制御するための特別の基本部品である。これまでに論じたように、現時点における好ましい実施例では、これらは、XilinxのFPGA内のCLBである。これらのインスタンスは、また、ロジックを配置するロジック・モジュール（現時点における好ましい実施例ではCLB）の相対的な位置を指定するLOC特性を持っている。  
30

RPMセルには、チップ・パーティショニング・プログラム1018がそれらをいくつかのロジック・チップ間に分割するのを防ぐために、フラグ（現時点における好ましい実施例では、このフラグは“NOFLAT”と呼ばれる）を付加しなければならない。RPMセルは、また、予め計算された容量値及びその大きさ（垂直及び水平方向に用いられるロジック・モジュール、すなわちCLBの数）を含む特性を持っていなければならない。

FMAP及びHMAPの基本要素（プリミティブ）とロジック基本要素との間のパラレル接続は設計ルール違反を生じることがあるため、データ判定プログラム1010は、RAMセル内のネットリストの確認を行わない。NGD出力プログラム1020は、各RAMインスタンス内の全ての基本要素のLOCの値を保持する。これにより、PPR1022は、LOCの特性によって定義される制約を満たすような形でRPMをチップ内に配置することができる。  
40

ユーザデータ入力プログラム1008は、ユーザによるクロック及び他の設計情報の入力を可能とするのみならず、グローバルなプローブ多重係数を計算する。プローブは、設計をデバッグ中に観察される、ネットリスト内部の点である。プローブ多重係数は、ロジック・チップ10に付加されるスキャン・チェーンの長さを決定する。ユーザは、プローブを表にするか、全体表示モードをリクエストすることができる。全体を表示する場合には、多重係数は64となる。ユーザが、信号の指定リストのみを目で見たい場合には、多重係数は、以下のように計算される。  
50

( プローブの数 ) × ( 偏差係数 ) / ( 432 × ロジック・ボードの数 )

ロジック・ボード 200 の数は、計算が行われるときに知られていなければならない。偏差係数は、実験的に決定される係数であり、プローブされる信号がロジック・ボード 200 の間で均一に分配されない場合に考慮されるものである。確率的な理論を考慮に入れれば、大きいシステムに対する 1.4 と 2 枚のボードのシステムに対する 1.7 の間となる。B 枚のボードを有するシステムに対しては、概略、 $1 / (1 - 0.29) \sqrt{B} / (B - 1)$  である。この係数をさらに大きくすることによって、複数のロジック・ボード 200 のリコンパイルを必要とせずにプローブを增加的に加算するためのルーム (room) を提供する。

好ましい実施例のシステムにおけるロジック・アナライザのイベントは、ロジック・ボード 200 のロジック・チップ 10 内で、プログラム可能なロジックによって計算される。それゆえ、イベントの計算のために、ロジック・チップ 10 内に容量を確保する必要がある。この結果として、ユーザが、信号及びイベントの定義を設計のコンパイル後まで遅らせると、影響を受けるチップのインクリメンタルな再コンパイルを行う必要が生ずる。あるチップについて確保された容量が十分でない場合には、図 20c で示したように、他のロジック・チップ 10 でイベント検出器を構成するに十分な容量があるものに信号をルートしなければならない。この場合には、より長いコンパイル時間を要する。イベントを作成するために使用される全ての信号をコンパイル前に指定することによって、コンパイル時間が長くなるのを防ぐことができる。この段階では、実際にイベント又はトリガを定義する必要はない。なぜなら、このような定義は容量に影響を与えないためである。イベント・ロジック機能自体は、コントローラ 221 に接続された JTAG バス (図 20 及び 20c に示す) を使って動作中にロジック・チップ 10 にダウンロードすることができる。最後に、このユーザデータ入力ステップ 1008 の間では、ユーザーは、クリティカルでない (non-critical) 信号に対する時間多重係数を選択しなければならない。前述したように、時間多重係数は、1、2 又は 4 のいずれにも設定できる。

チップ・パーティショニング・プログラム 1016 及び 1018 は、クラスタリングの基礎アルゴリズム (a clustering based algorithm) を使用する。類似するアルゴリズムの例としては、Quick Turn Design Systems, Inc. の System Realizer™ 等のハードウェア・エミュレーション・システムがある。しかしながら、現時点における好ましい実施例では、種々の相違点がある。これらの相違点を、以下に詳しく説明する。

1) 特定の種類のセルでは、不適正なパーティショニングやクラスタリング等を防止するために特別な注意を払わなければならない。ノー・タッチ ("No-touch") セルは、ロジックと一緒にクラスタすることができない特定のセルである。ノー・タッチ・セルの例としては、図 23 に示す MP セルがある。ノー・フラット ("No-flat") セルは、いくつかのチップの間に分割することができないセルである。ノー・フラット・セルの例としては、ラッチ及びハード・マクロがある。これらでは、分割によってタイミングの問題を生ずる。

2) ある特別なネットは、ドライバを持ってなく、任意に切断することができない。POWER 及び GROUND に加え、ロジック・ゲートが接続される特別のネットのような例が多重クロック信号 44 である。特に、動作テストベンチ・コンパイラ 1004、EBM コンパイラ 1030 及び LCM コンパイラ 1006 は、時間多重クロックに接続されるロジックを作成する。

3) ピン出力の制約によって、クラスタが有するネットの最大数が制御される。ロジックのクラスタが RILEGULAR 外部入力ネット、ROLLEGULAR 外部出力ネット、CNクリティカル外部ネット、P プローブ信号 (P probed signals) を有し、プローブに対する時分割多重係数が T であると仮定すると、このクラスタを 1 つのチップの上に実現するために必要なピンの数は、以下のように計算される (すべての分割動作は、四捨五入のない純粋な整数型の分割である)。

a. ロジック信号について時間多重を行わない場合には、ピンの数は、 $R_I + R_O + CN + (P + T - 1) / T$  で与えられる。

10

20

20

30

40

50

b . ロジック信号について 2 対 1 の時間多重を行う場合には、ピンの数は、 $(R_I + 1) / 2 + (R_O + 1) / 2 + CN + (P + T - 1) / T$  で与えられる。

c . ロジック信号について 4 対 1 の時間多重を行う場合には、ピンの数は、 $\max((R_I + 1) / 2, (R_O + 1) / 2) + CN + (P + T - 1) / T$  で与えられる。

注 ) ユーザーによって完全表示モードが選択された場合には、プローブ P の数は、フリップ・フロップ及びラッチの数と等しいものとされる。

4 ) クラスタに許される最大サイズは、特定のロジック・チップ 10 のゲート容量に依存する。このロジック・ゲートに加え、時分割多重、プローブ及びイベント検出回路について、追加の容量が必要となる。ロジックのクラスタは、RN 個のレギュラー（ノン・クリティカル）外部ネット（RN = RI + RO）、P 個のプローブ信号、E 個のイベント検出で使用される検出信号を仮定すると、時間分割マルチプレクシング、プローブ及びイベント検出の回路における追加される容量は以下のように与えられる。  
10

a . ロジック信号について時間多重を行わない場合には、ロジック・アナライザにおける追加の容量は以下で与えられる。

$$\text{フリップ・フロップ: } P + 2 \times E + \log E$$

$$\text{ゲート: } C_1 \times P + C_2 \times ((E + 1) / 2) \times 8$$

現時点における好ましい実施例では、定数は、C1 = 2、C2 = 4 である。これらは、実験の結果に応じて後で修正される。

b . 任意の種類の時間多重（2 : 1、4 : 1 又は他の処理）を有し、ロジック・アナライザで必要とされるものに加えて RN 個のフリップ・フロップの追加が必要となる。  
20

5 ) 米国特許 No. 5,475,830 号に説明されているように、クロック・ツリーを正確に実現する必要性から、パーティショニングが制御される。設計内の各ネットは、CLKMASK と呼ばれる 16 ビット整数型のプロパティを割り当てられる。CLKMASK のビット i は、ユーザー・クロック i が直接（非反転）の位相でネットに到達する場合にセットする必要がある。ビット 8 + i は、ユーザー・クロック i が反転された位相でこのネットに到達する場合にセットする必要がある。この情報は、PPR プログラム 1022 に渡されて、必要な遅延の調整を行う。

NGD 出力プログラム 1020 は、PPR プログラム 1022 での処理に適したフォーマットでネットリストを出力する。これに加え、これは、いくつかのロジックの修正に関する、時間領域マルチプレクス又はロジックのデバッグを行う。これらの関数としては、以下のものがある。  
30

・ 関連配置 (RP) マクロの保存 :

データベース内の関連配置マクロは、PPR に渡される NGD ファイルに保存される。RP マクロは、Xilinx の FPGA の中の固定的な CLB のパターンにマップされたロジック・ゲートのマクロである。RP マacro は、後続のソフトウェアのステップで再びパーティションされることなく、そのタイミング特性は保存される。

・ TDM セル挿入 : 時分割多重セルは、マルチプレクサ・チップ 12 に接続される各ロジック・チップ 10 の境界（バウンダリ）に追加される。予め定められたセルは、多重化されている I/O ピンの組に対して相対的に配置されて使用される。図 24a ~ 24k は、この I/O ピンの種類に応じて挿入される種々の異なる TDM セルを示したものである。時分割多重のために、ロジック・チップ 10 及びマルチプレクサ・チップ 12 の端子は、特別な RPM セルを使用して図 24a ~ 24k に示すように、4 つのグループに分割される。残りの端子、2 つのグループは使用され、又はレギュラーな非多重 I/O はすでにロジック・チップ 10 又はマルチプレクサ・チップ 12 で使用されている。非多重 I/O は、常に、クリティカルなネットに対して使用される。  
40

・ TDM 制御ロジック挿入 : TDM 制御ロジックは、TDM 制御信号である MC、MS、MT、E0、E1、E2 及び E3 を生成し、図 24a ~ 24k に示す回路に分配する。これらの信号は、図 24a ~ 24k に示すロジックに加えて、各ロジック・チップ 10 に挿入される 3 つの特別な制御セルの中の 1 つによって生成される。これらの信号の生成は、図 6 に示すロジック 104 又は図 3 に示すロジック 68 によって行われる。MC はマルチ

プレクサ・クロック信号 4 4 であり、 M S は分周クロック信号 5 0 であり、 M T は方向信号 8 0 であり、 E 0 ~ E 3 はそれぞれイネーブル信号 9 0 、 9 2 、 9 4 、 9 6 である。特別なセルは、ロジック・チップ 1 0 の固定入力ピンに接続される 2 つの入力 MUXCLK 及び同期 4 8 を有している。ある種類の制御セル（図示していない）は、 T D M を使用しないが、多重クロック信号（ MUXCLK ） 4 4 に接続されるチップに対して使用される。このセルは、マルチプレクサ・クロック信号（ MUXCLK ） 4 4 のみを出力する。第 2 の種類は（図 3 に示すロジック 6 8 ）は、 2 対 1 の T D M を有する設計に対して使用される。これは、多重クロック信号（ MUXCLK ） 4 4 及び M S （分周クロック）信号 5 0 を出力する。第三の種類の制御セル（図 6 のロジック 1 0 4 ）は、 4 対 1 の時間多重に使用される。これは、マルチプレクサ・クロック信号（ MUXCLK ） 4 4 、 M S （分周クロック） 5 0 、 M T （方向） 8 0 、 E 0 9 0 、 E 1 9 2 、 E 2 9 4 、 E 3 9 6 を生成する。  
10

- ・プローブされた信号へのスキャン・セル挿入：プローブされた信号の各々は、プローブ・セルのデータ入力に接続する必要がある。プローブ・セルは出力を持っておらず、他の 2 つの入力を持っている。これらの入力のうちの 1 つは、多重クロック信号（ MUXCLK ） 4 4 である。他の入力は、チップの入力からのトレース・クロック信号 2 0 0 2 に接続されている。プローブ・セルは、図 2 0 b 及び図 2 0 c に示すように、フリップ・フロップ 2 0 0 4 及びマルチプレクサ 2 0 0 5 を有している。

- ・スキャン・チェーン指定ファイルの生成：プローブ・セルのすべてのインスタンスは、スキャン・チェーン指定ファイルにリストされている。チップのスキャン出力 2 0 0 6 （図 2 0 b 参照）も、リストに加えておく必要がある。これらの出力は、チップ・ロジックのクラスタのデータベース・モデルに挿入される。これにより、システムのルータは、これを参照して、適切な接続を生成する。出力の数は、 ( P + T - 1 ) / T で与えられ、ここで、 P はプローブ・セルの数、 T はプローブ・信号の時分割多重係数である。  
20

- ・イベントに寄与する信号についてのイベント検出セルの挿入：イベントに寄与する信号は、一対ごとに分割される。各対は、図 2 5 に示すように、イベント検出セル 1 3 0 0 の 8 つのコピー（ copy ）のうちの I 0 及び I 1 に接続される。イベント検出セルの好ましい実施例は、図 2 0 c に示されている。イベント検出セル 1 3 0 0 は、 4 つのフリップ・フロップ 2 0 1 0 及び C L B メモリ 2 0 1 8 によって構成される。 4 つのマルチプレクサ 2 0 2 0 及び 4 つの出力バッファ 2 0 2 2 は、 4 つの多重化されたイベント信号 2 3 6 を生成するために使用される（図 2 0 c 及び 2 0 a に示されている）。信号の数が奇数の場合には、各イベント検出セルに対する入力のうちの 1 つは、対応する 8 つのセルについて使用されないままとなる。  
30

- ・イベント検出器の出力に対する 8 つのバランスされた A N D ツリーの生成及び 8 つの A N D ツリーの出力を 4 つの指定されたイベントピンに接続する T D M ロジック：イベント検出セル 1 3 0 0 の出力は、直前のステップで作成された 8 つのツリーの 1 つのコピーが各ツリーに存在するように、 8 つのバランスされた A N D ツリーを用いて結合される。ツリーの出力は、図 2 6 に示されているように、特別なイベント多重セルを用いて一対ごとに時間多重化される。この回路は、図 2 0 c を参照して説明されている。 A N D ゲート 2 0 1 2 は、図 2 0 c に示されているように、ワイド・エッジ・デコーダ 2 0 1 2 を用いて構成される。図 2 6 に、この回路をさらに詳細に示す。  
40

- ・イベント検出器のダウンロード経路の生成及びバウンダリ・スキャン・コントローラ：イベント検出器ダウンロード回路 1 5 0 0 を図 2 7 に示す。これは、カウンタ 2 0 1 6 、シフト・レジスタ 2 0 1 4 及び J T A G コントローラ 1 1 5 0 により構成される。 J T A G コントローラ 1 1 5 0 は、 Xilinx のロジック・チップ 1 0 の標準的な部分として利用可能である。この回路は、また、スキャン・レジスタ及びイベント検出器とともに図 2 0 c に示されている。イベント検出器のダウンロード回路 1 5 0 0 は、 W A 1 5 0 2 、 W E 1 5 0 4 、 D R C L K 1 5 0 8 及び T D I 1 5 0 6 信号をすべてのイベント検出器に対して生成する（図 2 0 c にも示されている）。イベント検出器のカウンタ 2 0 1 6 は、 W A 信号 1 5 0 2 及びシフト・レジスタ 2 0 1 4 に対するクロックを生成する。シフト・レジスタ 2 0 1 4 の長さは、イベント・デコーダ回路の数に依存する。この回路を、図 2 0 c  
50

及び 27 に示されている。好ましい実施例では、シフト・レジスタ 2014 は、イベント検出器の数に応じて作成される。しかしながら、これは、チップあたりのイベント検出器の最大数を限定し、シフト・レジスタ 2014 の設計を決定するものであり受容される。PPR プログラム 1022 は、使用されないロジックのほとんどを整理する。

図 21 を参照して、ボード・パーティショニングのステップ 1024 を説明する。ボード・パーティショニングのステップ 1024 では、1 つのロジック・ボード 200 (37 チップ) 又は一対のロジック・ボード (74 チップ) のロジック・チップ 10 及び 204 のチップの数を超えない最大のチップ数を持ったチップ・クラスタチップのクラスタ (1 つのクラスタは、相互接続された部品の集合である) を、以下の条件で検索する。

1. 入力 (ingoing) 又は出力 (outgoing) のネットの総数は、図 11 に示す一対のロジック・ボード 200 に対する 2 つのバックプレーン・コネクタ 220 の I/O 接続の数にバックプレーン利用係数 (backplane utilization coefficient) を乗じた数を超えてはならない。ターゲットのバックプレーン利用係数は、実験的に決定され、システムのルーティング・プログラム 1032 が平均的にどの程度成功するかに依存する。ターゲットのバックプレーン利用係数は、90% 程度になると予想される。  
10

2. ロジック・アナライザのチャンネルとしてマークされたチップの出力の総数は、864 を超えてはならない (54 個のマルチプレクサ・チップ 12 の数に 8 個の SGRAM 210 のピンの数 8 を乗じ、その総数は、1 つのモジュール内の 2 つのロジック・ボード 200 の数 2 を乗じたものである。)。

3. EBM メモリのインスタンスのフル・セットは、24 チップ (サイズが半分のモジュールでは 12) 以下でなければならず (図 11 を参照して説明したように、1 つのロジック・ボード 200 には 12 の RAM 208 があり、また一対のロジック・ボードには 24 個存在する)、そして、EBM メモリについて必要とされるロジック・チップの数は総数である 74 (サイズが半分のモジュールでは 37) から減算される。  
20

4. CPUT セルのインスタンス (即ち、ユーザの設計からの CPUT インスタンスの数) は 2 を超えてはならない (サイズが半分のモジュールでは 1) (図 11 を参照して説明したように、ロジック・ボード 200 当り 1 つのプロセッサ 206 があり、一対のロジック・ボードには 2 個存在する。)。

5. 74 の (サイズが半分のモジュールでは 37) チップのうち 2 (サイズが半分のモジュールでは 1) つのチップ 204 は、クロック・ジェネレーション・ロジック・チップとして使用でき又はマイクロプロセッサのセルに装着して使用することができる。マイクロプロセッサのセルが存在する場合には、クロック・ジェネレーション・チップは使用することができず、その逆も成立する。これは、CoSim ロジック・チップ 204 が、チップでは同時に 1 つの機能にしか使用できないためである。しかしながら、どちらの機能も使用されない場合がある。このような場合には、72 (1 つのロジック・ボード 200 では 36) の最大容量のロジック・チップ 10 しか使用できない。クラスタの I/O ピンの数が 162 以下のときには、2 つの追加の CoSim ロジック・チップ 204 (サイズが半分のモジュールでは 1 つ) を追加されるユーザー・ロジックを実現するために使用することができる (図 11 参照)。  
30

適当なクラスタが特定されると、フル・サイズのクラスタは、ボード間の相互接続が 1868 (ターボ・コネクタ 202 のピンの数) 以内の条件で、2 つのエミュレーション・ボードに分割される。各ボードは、クリティカルなクラスタのリソースの総数の半分 (1804 入力又は出力ネット、12 の EBM メモリ、432 のロジック・アナライザ・チャンネル、37 のロジック・チップ 10、204) を超えてはならない。  
40

EBM のコンパイル・ステップ 1030 では、エミュレーション・ブロック・メモリ (EBM) として実現されるメモリ・セルのインスタンスを作成する。これらは、チップのパーティショニングの間は、どのロジック・クラスタにも含められない特別なセルとして作成される。評価サブルーチンによって、任意のメモリ・インスタンスのセットとして、いくつの EBM チップ 208 (図 11 参照) が必要となるかが評価される。このサブルーチンは、階層パーティション・プランニング・プログラム (hierarchical partition plan  
50

ning program) (HPP) 1014 (この接続は図21に表示しない) からコールされ、ボード・パーティショニング・プログラム1028が、1つのボードに実現される1組のメモリ・インスタンスを適当に決定し、また、メモリ制御回路で使用されるロジック・チップ10の数を決定する。ボード・パーティショニング・プロセスが完了すると、EBMメモリ・コンパイラ1030は、ロジック・ボード200の各RAMチップ208に関連するロジック・クラスタを作成する。RAMチップ208に接続される配線には、NGD出力プログラム1020がこれらに時間多重を組み込まないように、すべて、“クリティカル”の表示が付される。これらは、また、システム・ルータ1032が正しいI/Oの制約を生成できるよう、それぞれのロジック・チップ10のピン番号を含むプロパティを持っている。EBMロジック・クラスタは、ユーザーによってはアクセスできない自動的に生成されるロジックを含んでいるため、EBMロジック・クラスタにプローブ信号を含めることができず、EBMロジック・クラスタは、イベントを生成することができない。  
好ましい実施例では、EBMロジック・クラスタは、プリコンパイルされる。これにより、これらのクラスタ配置とルーティングの時間がセーブされる。EBMメモリ・コンパイラ1030は、米国特許第5,841,967号に完全に開示されている。

システム・ルータ1032は、ロジック・チップ10、204、マルチブレクサ・チップ12及びロジック・ボード200に物理的なワイヤをロジック・ネット(又はエミュレートされる設計の信号)、一対のロジック・ネット(2対1マルチブレクス)及び4つのネットからなるグループ(4対1マルチブレクス)に割り当てる。次いで、ロジック・チップ10のピン及び時分割多重(TDM)の位相をロジック・チップ10及び204に入り又はこれらから出る信号に割り当てる。

システムのルーティングを行う時には、信号の遅延を最小限に抑えるように、時間多重信号を最適にルーティングすることが重要である。このためのアルゴリズムを以下に示す。

#### 1. 2対1時分割多重(2-1TDM)

最適なルートは、各マルチブレクス・チップ12内のTDM位相をスイッチするが、物理的ネットのソースから物理的ネットの目的地(destination)にはルーティングしない。

最適なルートの例としては以下のものがある。

```
alpha/output/even-beta/input/even-beta/output/odd-alpha/input/odd, 又は、  
alpha/output/even-beta/input/even-beta/output/odd-muxbeta/input/odd-muxbeta/output/evenbeta/input/even-beta/output/odd-alpha/input/odd
```

この記述では、alphaチップはロジック・チップ10又は204に相当し、betaチップはマルチブレクサ・チップ12に等価となる。これにより、2つのロジック・チップ10又は204の間での遅延は、最小値である1サイクルとなる。この遅延を、図3及び図4の回路について調べれば、2分の1サイクルになるとも思える。実際には、これはちょうど1サイクルかかる。これは、ロジック・チップ10、204内のデマルチブレクサ34は、2分の1サイクルの終わりに近い部分で信号をクロックするため、データが受信されても2分の1サイクルにわたって、そのまま変化しないからである。ルータ1032によって最適なルートが見つからない場合には、即ち、適当なマルチブレクサ出力が利用可能でないか、適当な位相のマルチブレクサ出力が利用可能でないか適当なフェーズでのロジック・チップ10又は204の入力が利用可能でないときは、信号はさらに2分の1サイクルの遅延を失う。ルータは、可能な限り、同じネットについてのミスを重ねないように努める。クリティカルなネットは、遅延を最小化するため、マルチブレクスされない。

#### 2. 4対1時分割多重(4-1TDM)

各物理的ネットは、常に、1つのインアウト・ピン(IIOOシーケンス)及び1つのアウトイン・ピン(OOIISシーケンス)を有している。ここでも、最適なルートは、各マルチブレクス・チップ12内の時分割多重(TDM)位相をスイッチするが、物理的ネットのソースから物理的ネットの目的値にはルートしない。最適なルートの例としては以下のものがある。

```
alpha/01/01-beta/10/11-beta/01/02-alpha/10/2alpha/01/02-beta/10/12-beta/10/03-alpha/01/13alpha/01/01-beta/10/11-beta/01/02-muxbeta/10/12-muxbeta/10/03-beta/01/13
```

10

20

30

40

50

-beta/10/04-alpha/01/14

これにより、最小限の2分の1サイクルの遅延alpha-to-alphaが得られる。しかしながら、4対1時分割多重(4-1TDM)の2分の1サイクルは、2対1時分割多重(2-1TDM)の1サイクルと同じ期間となる。このため、全てのネットが最適にルーティングされたと仮定すれば、4対1時分割多重(4-1TDM)では、2対1時分割多重(2-1TDM)に比べて速度が損なわれることはない。しかしながら、4対1時分割多重(4-1TDM)におけるミス(すなわち、前記した、最適ルートの検索に失敗する)は、2対1時分割多重(2-1TDM)における場合よりも重大な影響を及ぼす。例えば、経路:alpha/01/01-beta/10/11-beta/01/01-alpha/10/11は、信号を1.25[4対1時分割多重(4-1TDM)サイクル](又は2.5[2対1時分割多重(2-1TDM)サイクル]遅延させる。これは、最適な遅延に比べ1.5倍悪い。マルチプレクサ・チップ12、ルータ1032を通じた行程ごとに、ルータ1032が選択した入力・出力対に応じて4対1時分割多重サイクルの0、1/4、1/2又は3/4倍の遅延が生じる。このように、クリティカルなネットは、遅延を最小限とするために多重化すべきではない。

一部のロジック・チップ10又は204は、特定のピンに固定された入力/出力のネットを有している。例としては、多重クロック信号(MUXCLK)44、トレース・クロック信号2002、共同シミュレーション・ロジック・チップ204とプロセッサ206(図11参照)との間の接続、メモリ・コントロール・ロジック・チップ10とRAMチップ208との間の接続、イベント信号出力236等がある。これらの接続は、ルートする必要はないが、ロジック・チップ10、204のピンの制約のデータに含める必要がある。

制御モジュール600(図19に示されている)のクロック分配回路(マルチプレクサ・チップ12)については、追加のプログラムが必要とされる。これは、各ロジック・モジュールに到達する最大8つまでのユーザ・クロックを選択するためのクロック回路の一部である。

NGDアップデート・プログラム1034は、システム・ルータ1032によって生成された実際のピンI/O割り当てについての情報を、最終のパラレルパーティション・配置・ルート(PPR)ソフトウェア1036に供給する。これは、非時分多重化されない設計に対しては、I/Oパッドへの信号の割り当てと同じである。時分多重化される設計に対しては、ロジック・チップ10、204及びマルチプレクサ・チップ12の周辺にTDMロジックも追加される。

最終のパラレルパーティション・配置及びルート(PPR)プログラム1036は、PPRプログラムをインクリメンタル・モードに戻してチップの周辺においてI/Oピンを再ルートする。これまでに述べたとおり、PPRプログラムはXilinx Corporationから入手可能である。この再ルーティングにより、ロジック・チップ10、204の構成ファイル(configuration file)は、その前に準備PPRステップ1022で作成されたロジック・チップ10、204のコンフィギュレーション・ファイルが変更され、ピン出力はシステム・ルーティング・ステップ1032で決定されて固定される。

以上のように、集積回路をエミュレートし、検証し、解析するための好ましい方法及び装置について説明した。本発明の実施例及び応用例を示し、説明したが、当業者であれば、ここに開示した発明の概念を逸脱することなく種々の実施例及び応用例が可能であることは明らかであろう。それゆえ、本発明は、添付した請求の範囲の精神を除いては限定されべきではない。

10

20

30

40

【図1】

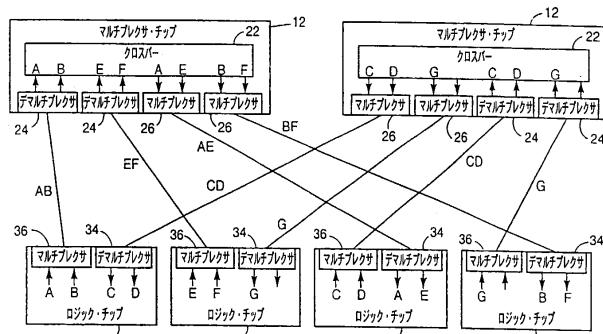


FIG. 1.

【図2】

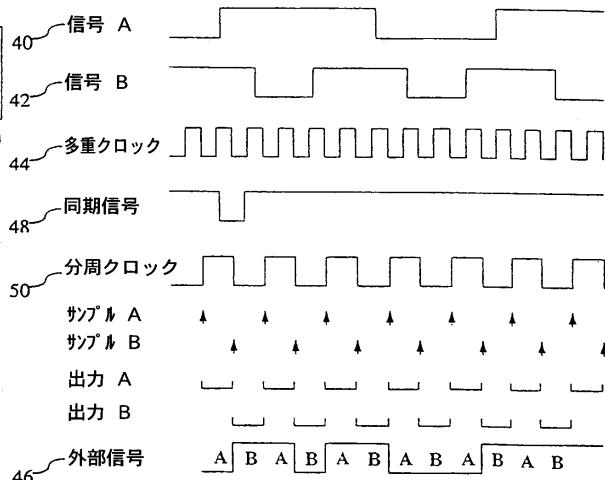


FIG. 2.

【図3】

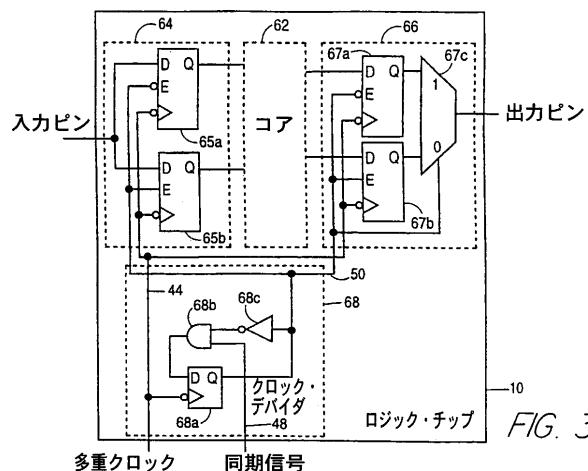


FIG. 3.

【図4】

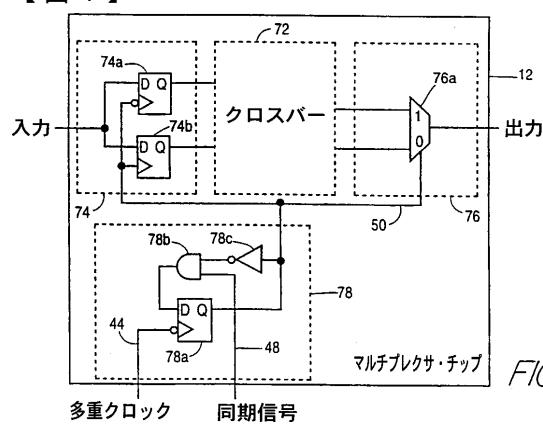
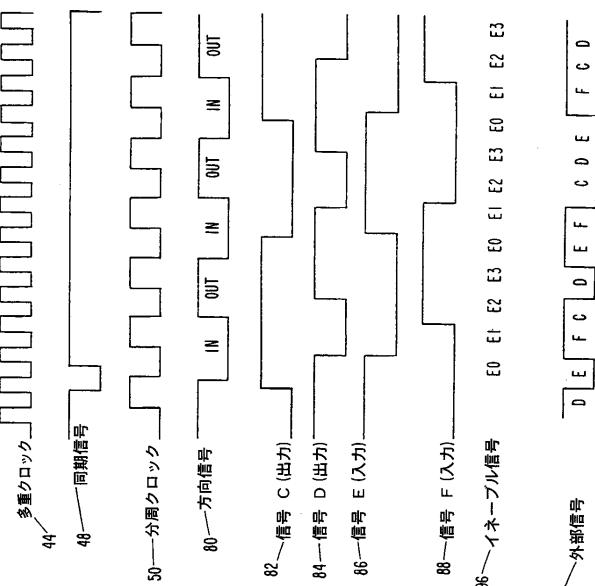


FIG. 4.

【図5】

外部信号  
98

D E F C D E F C D E F C D

【図6】

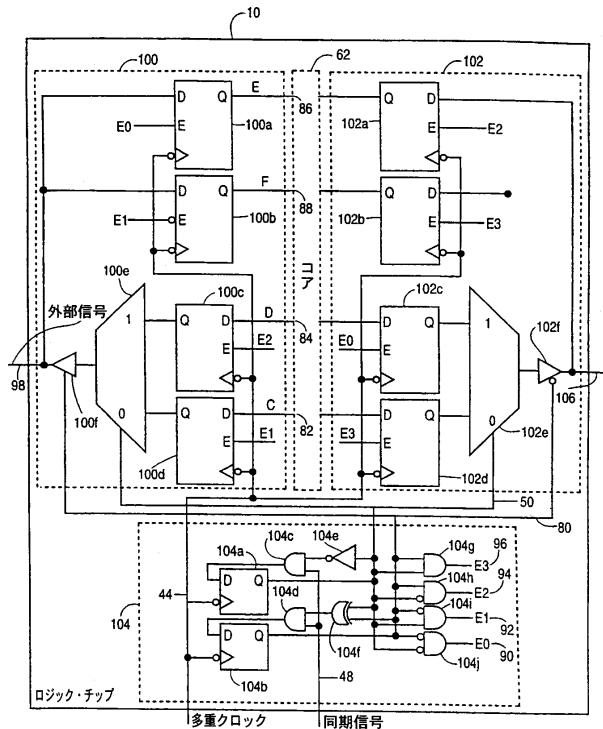


FIG. 6.

【図7】

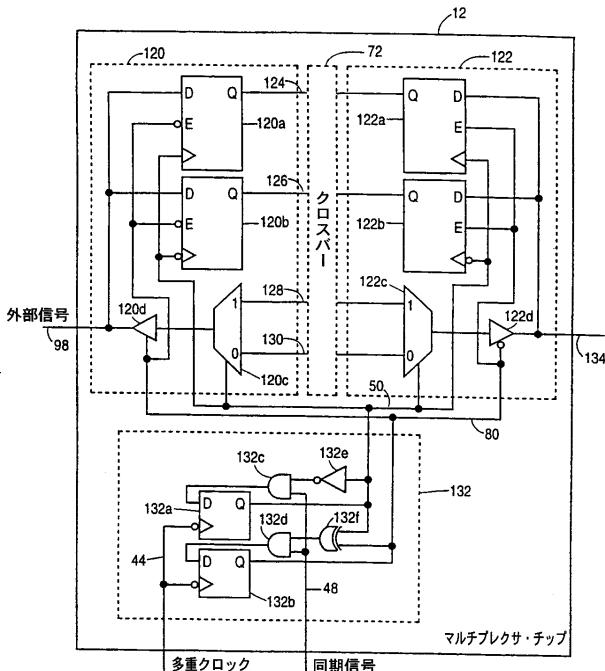
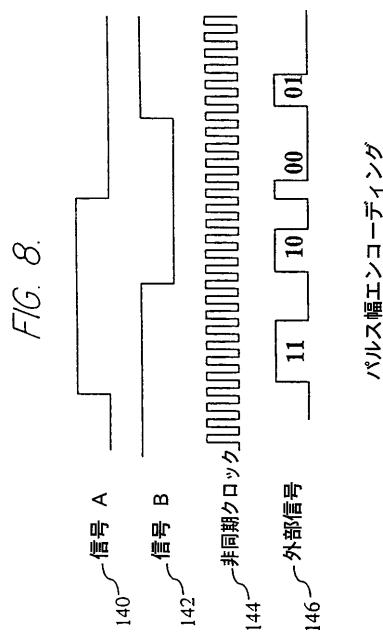
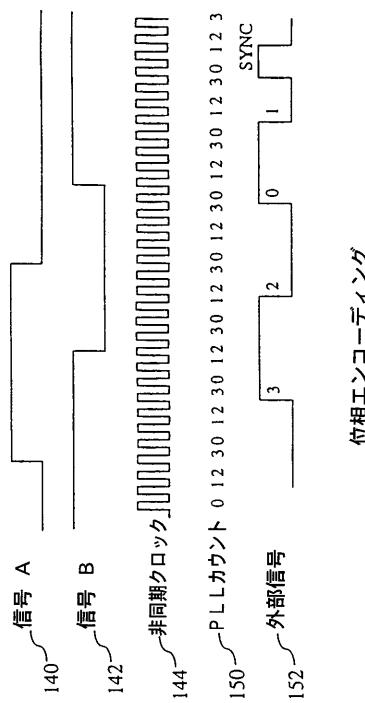


FIG. 7.

【図8】



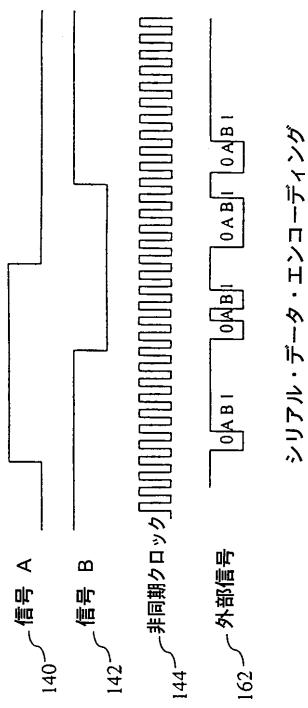
【図9】



位相エンコーディング

【図 10】

FIG. 10.



【図 11】

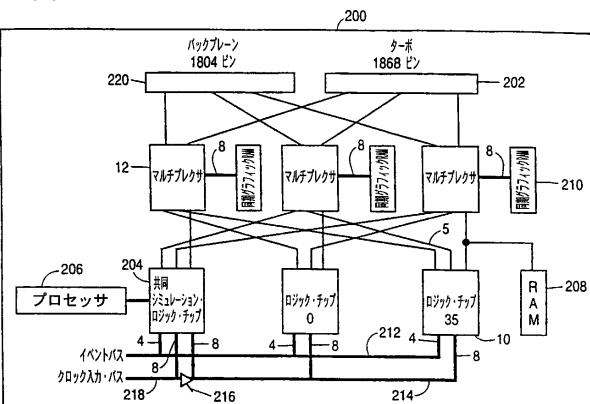


FIG. 11.

【図 12】

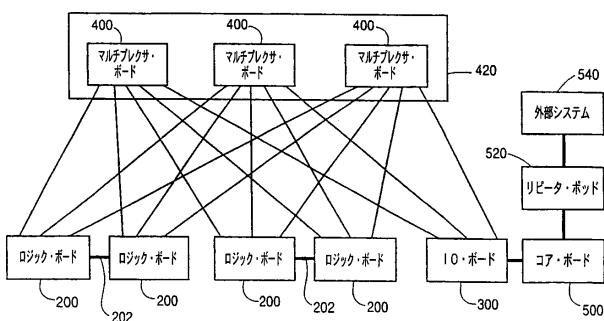
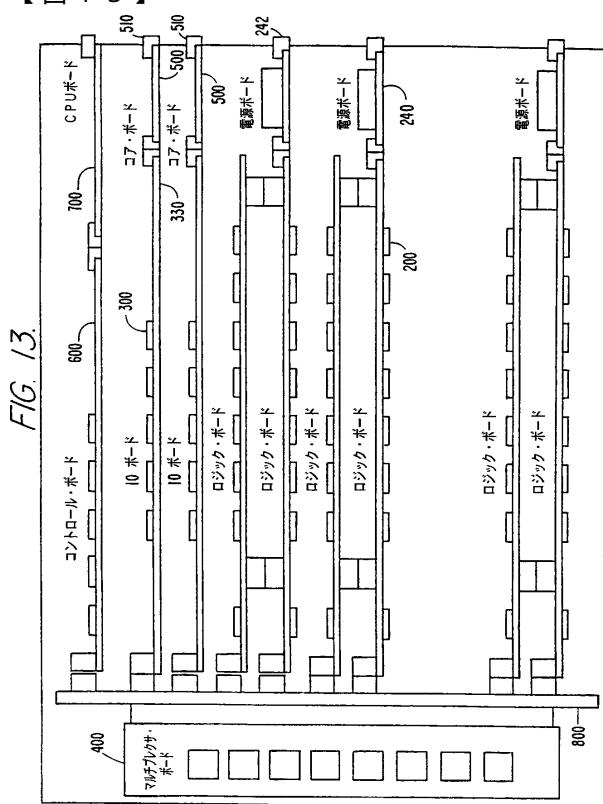


FIG. 12.

【図 13】



【図 14】

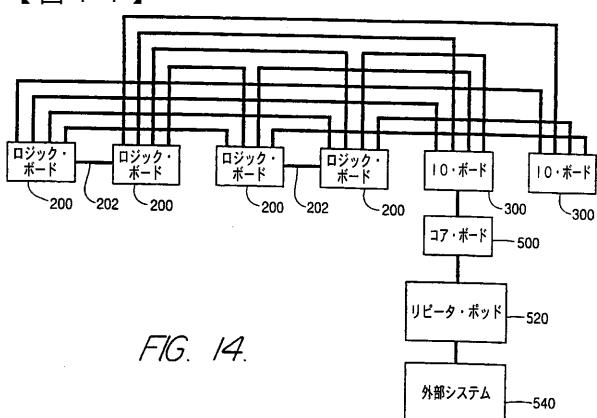
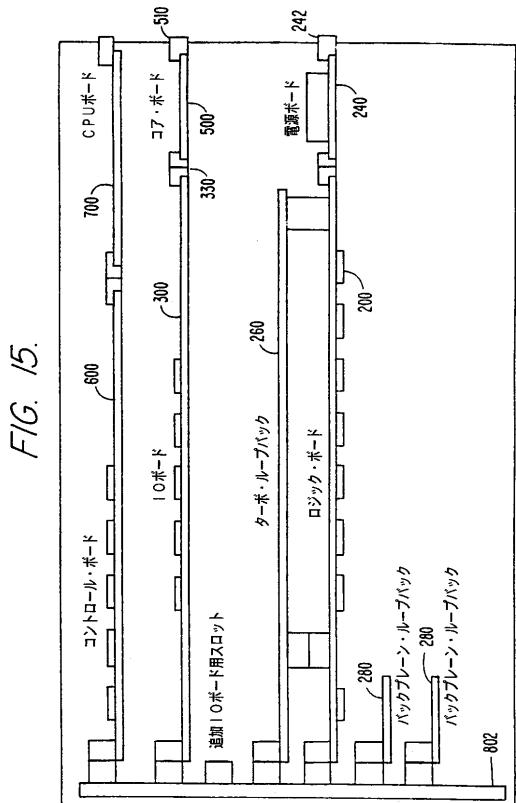
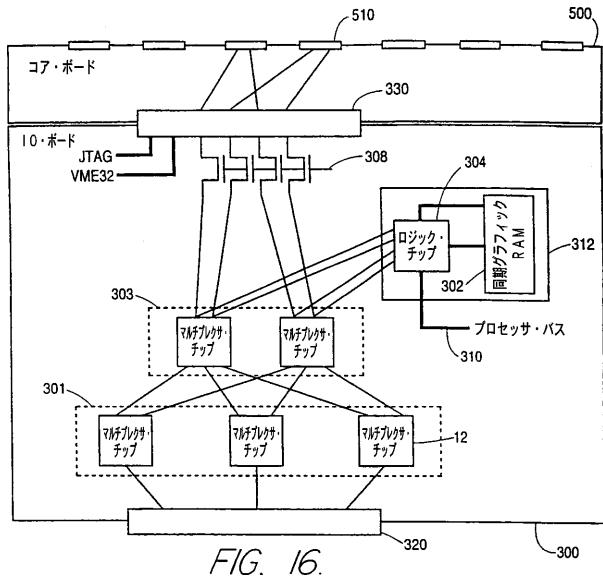


FIG. 14.

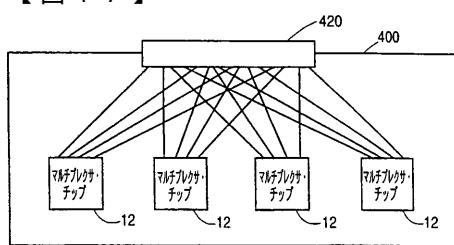
【図 15】



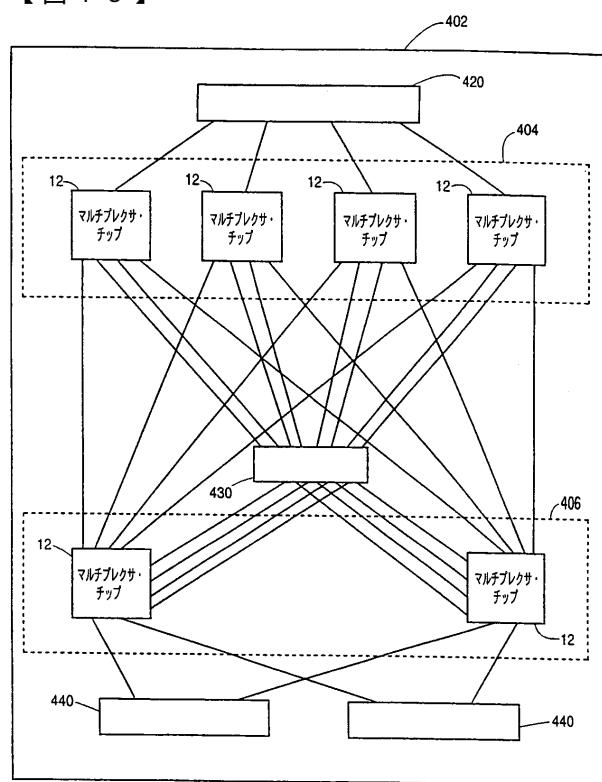
【図 16】



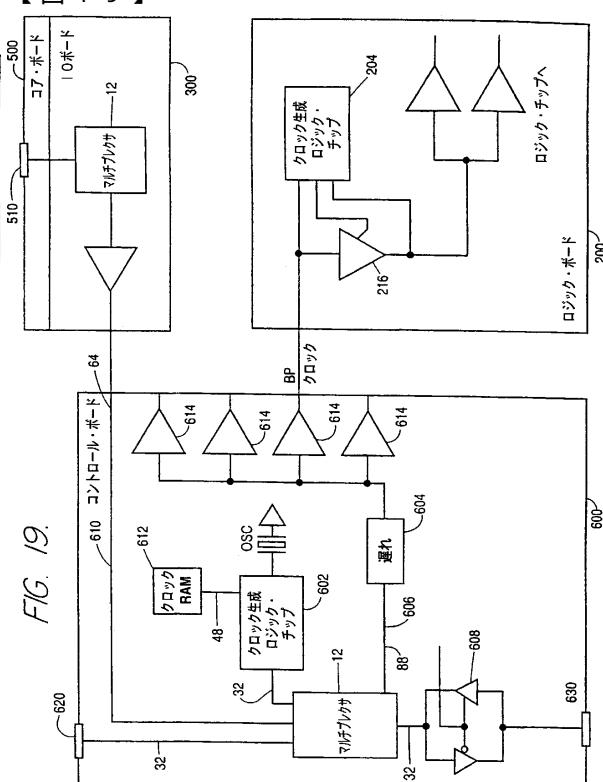
【図 17】



【図 18】



【図 19】



【図20】

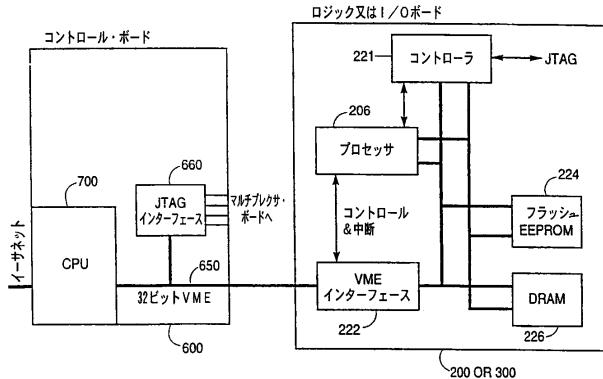


FIG. 20.

【図20a】

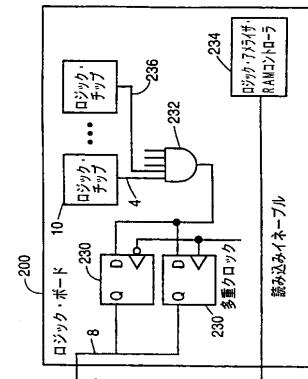
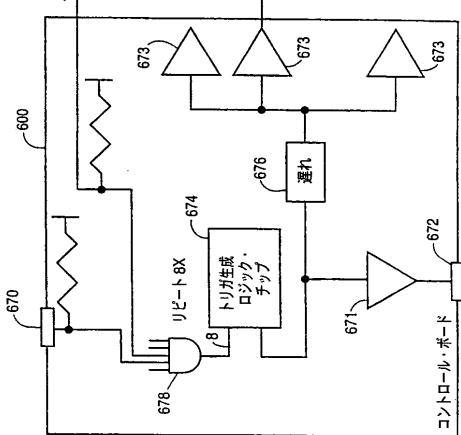


FIG. 20a.



【図20b】

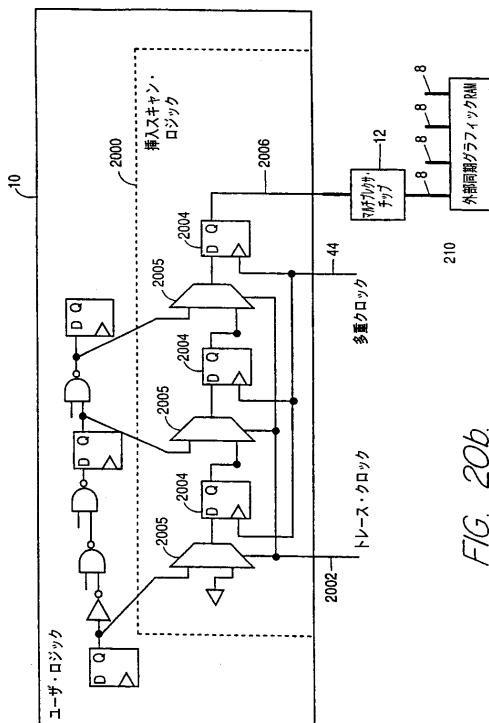
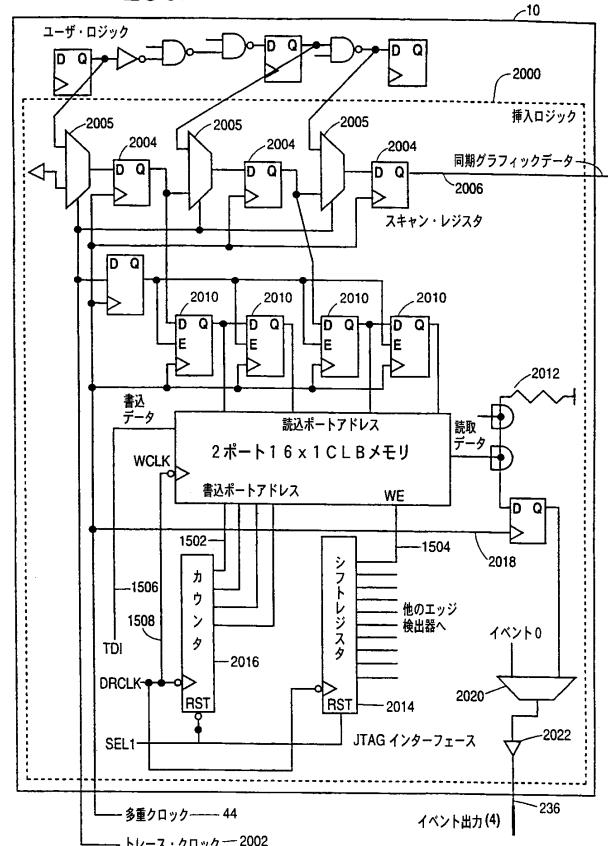


FIG. 20b

【図20c】

FIG. 20c.



ト雷斯・クロック—2002

多重クロック—44

イベント出力(4)—236

( 2 0 d )

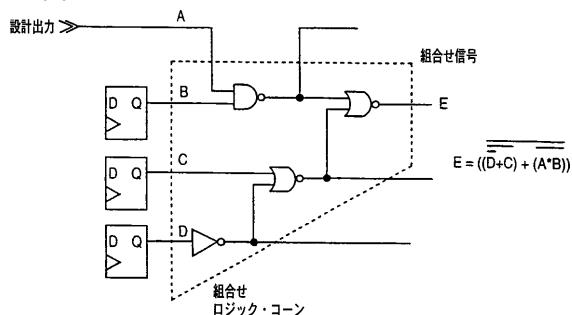


FIG. 20d.

【 図 2-1 】

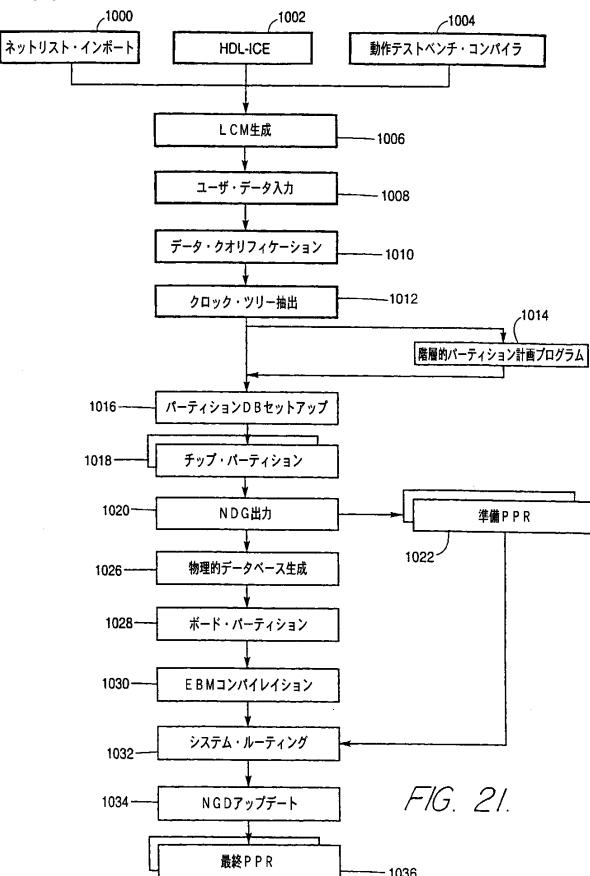
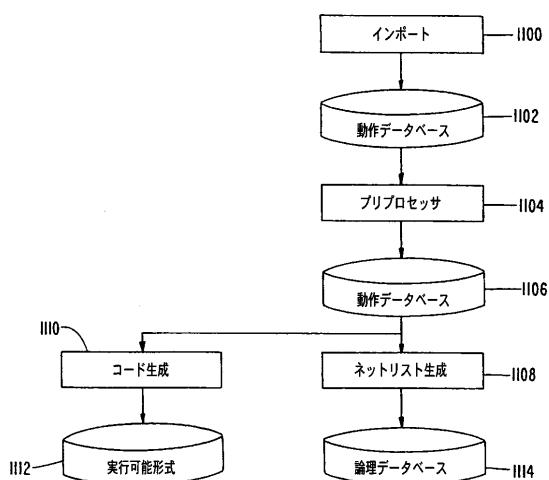


FIG. 21.

【 図 2 2 】

FIG. 22.



【図22a】

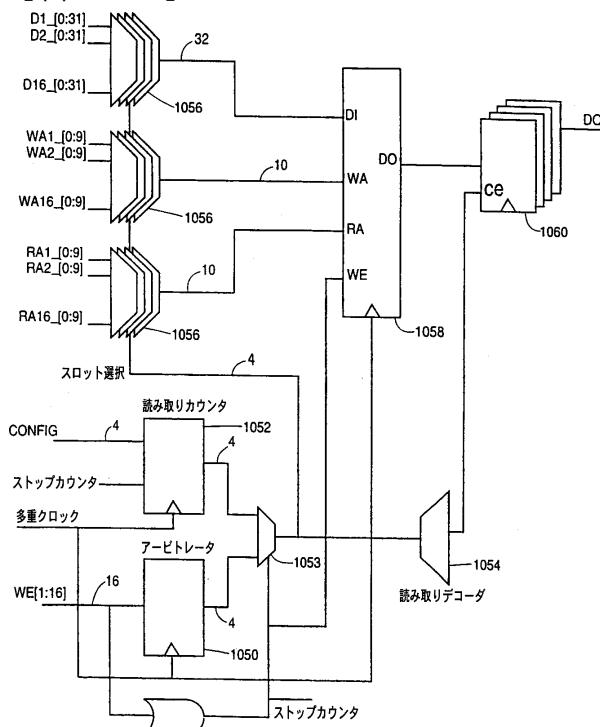
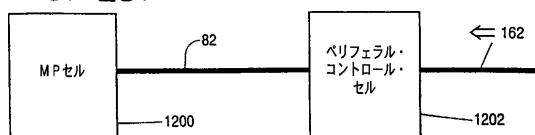


FIG. 22a.

【図 23】

FIG. 23.



【図 24 a】

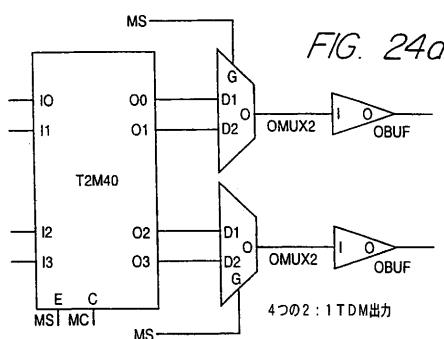


FIG. 24a.

【図 24 b】

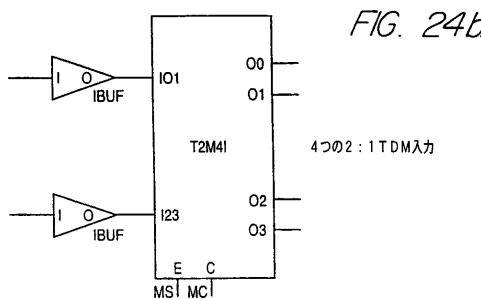


FIG. 24b.

【図 24 c】

【図 24 c】

FIG. 24c.

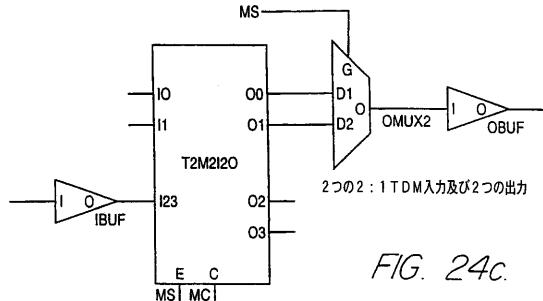


FIG. 24c.

【図 24 d】

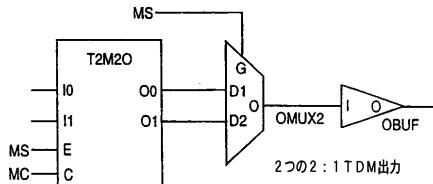


FIG. 24d.

【図 24 e】

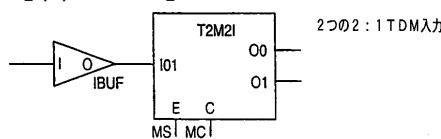


FIG. 24e.

【図 24 f】

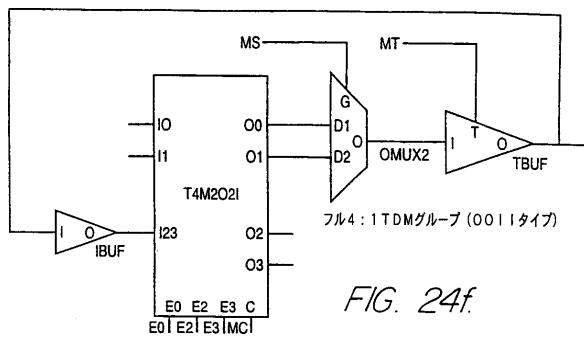


FIG. 24f.

【図 24 g】

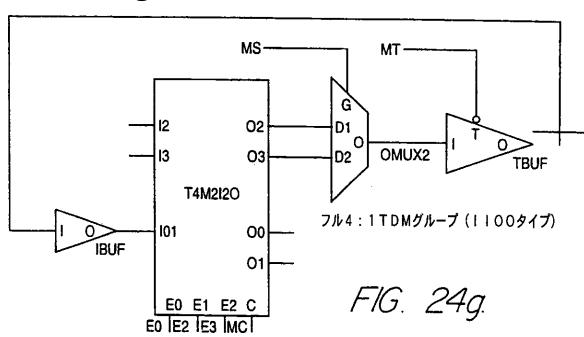


FIG. 24g.

【図 24 h】

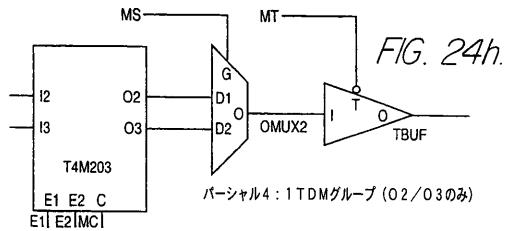


FIG. 24h.

【図 24 i】

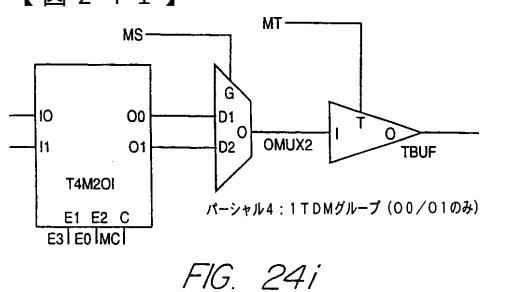


FIG. 24i.

【図 24 j】

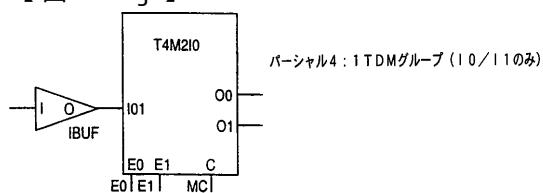


FIG. 24j.

【図24k】

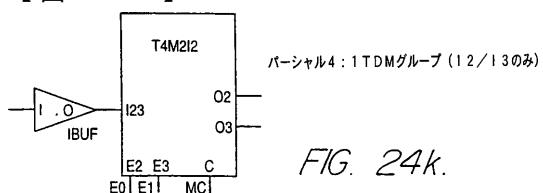


FIG. 24k.

【図25】

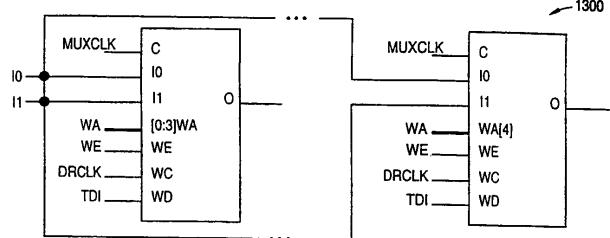


FIG. 25.

【図26】

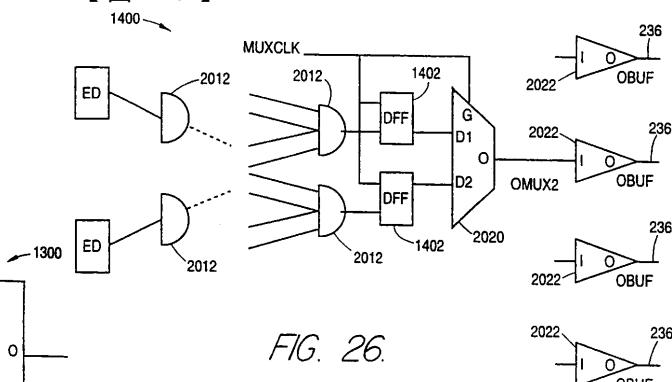


FIG. 26.

【図27】

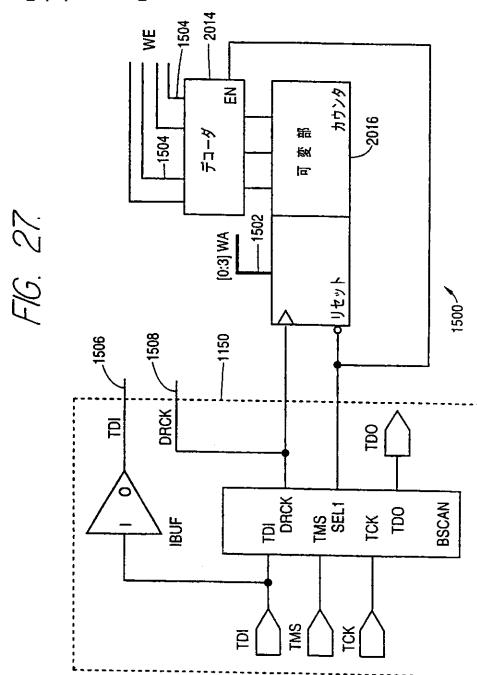


FIG. 27

---

フロントページの続き

(72)発明者 サンブル , ステファン・ピー  
アメリカ合衆国 95070 カリフォルニア州 , サラトガ , ボールマン・ロード 15027  
(72)発明者 バーシュティーン , ミハイル  
アメリカ合衆国 95008 カリフォルニア州 , キャンベル , ヴァレリエ・ドライブ 4168  
(72)発明者 パツ ,マイケル・アール  
アメリカ合衆国 97213 オレゴン州 , ポートランド , フォーティーナインス・アヴェニュー  
, ノースイースト 2734

審査官 久保 正典

(56)参考文献 特開平06-334368 (JP, A)  
特表平08-508599 (JP, A)  
特開平08-030653 (JP, A)

(58)調査した分野(Int.Cl. , DB名)

G06F 11/22 - 11/277  
G01R 31/28 - 31/3193  
G06F 17/50