

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号

特開2004-252585

(P2004-252585A)

(43) 公開日 平成16年9月9日(2004.9.9)

(51) Int.Cl.⁷

G06F 11/22

G06F 11/28

F I

G06F 11/22 340A

G06F 11/22 360B

G06F 11/28 L

テーマコード (参考)

5B042

5B048

審査請求 未請求 請求項の数 7 O L (全 20 頁)

(21) 出願番号 特願2003-40177 (P2003-40177)

(22) 出願日 平成15年2月18日 (2003.2.18)

(71) 出願人 000005821

松下電器産業株式会社

大阪府門真市大字門真1006番地

(74) 代理人 110000040

特許業務法人池内・佐藤アンドパートナーズ

(72) 発明者 藤田 充典

大阪府門真市大字門真1006番地 松下電器産業株式会社内

(72) 発明者 鶴本 克己

大阪府門真市大字門真1006番地 松下電器産業株式会社内

Fターム(参考) 5B042 GA13 HH06 HH07 HH17 HH49

5B048 AA12 DD01 DD17

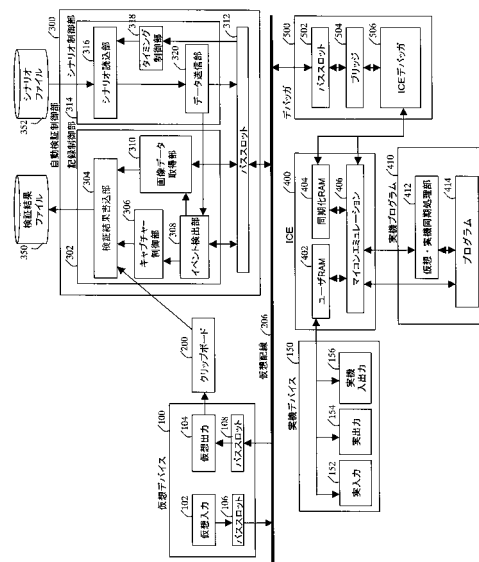
(54) 【発明の名称】 プログラム検証システム

(57) 【要約】

【課題】 実機デバイスの完成を待たずにプログラム検証が可能であって、かつ、動作フリーズ等を起こさずにシステムの自動評価を長時間安定して行えるプログラム検証システムを提供する。

【解決手段】 ICE400と、実機デバイス150に対応した検証動作を行うようICEを制御する検証制御部300と、仮想デバイス100からの制御信号等を検証制御部300に適合させるブリッジ504と、実機デバイス150に対するプログラム410の入出力動作に同期させて、同期化RAM404と実機用ユーザRAM402との間のデータ転送を制御する仮想・実機同期処理部412とを備え、ブリッジ504内に、検証制御部300からの検証動作指示が正常に処理されているかを判断する監視部を備えたプログラム検証システム。

【選択図】 図1



【特許請求の範囲】

【請求項 1】

マイコンの周辺デバイスとして実機デバイスおよび仮想デバイスを用いて、前記マイコン用プログラムの動作を検証するプログラム検証システムであって、
前記マイコン上の前記プログラムの動作をエミュレーションするエミュレータと、
前記実機デバイスに対応した検証動作を行うよう前記エミュレータを制御する検証制御部と、
前記検証制御部と前記仮想デバイスとの間に設けられ、前記仮想デバイスからの制御信号およびデータを前記検証制御部に適合させるブリッジ部と、
前記実機デバイスと前記プログラムとの間の入出力データを格納する実機用 R A M と、
前記実機用 R A M と前記仮想デバイスとの間に設けられた同期化 R A M と、
前記実機デバイスに対する前記プログラムの入出力動作に同期させて、前記同期化 R A M と前記実機用 R A M との間のデータ転送を制御する、仮想・実機同期処理部とを備えると共に、
前記ブリッジ部内に、前記検証制御部からの検証動作指示が正常に処理されているかを判断する監視部を備えたことを特徴とするプログラム検証システム。

【請求項 2】

検証動作の手順を記載したシナリオファイルを記憶するシナリオファイル記憶部をさらに備え、
前記検証制御部が、前記シナリオファイル記憶部から読み出したシナリオファイルに基づいて前記エミュレータを制御する、請求項 1 に記載のプログラム検証システム。

【請求項 3】

前記検証動作指示がリスタートコマンドであり、
前記監視部が、前記リスタートコマンドが所定時間以内に処理されたか否かを判断する、請求項 1 または 2 に記載のプログラム検証システム。

【請求項 4】

前記検証動作指示がキー実行コードであり、
前記監視部が、前記キー実行コードが所定時間以内に処理されたか否かを判断する、請求項 1 または 2 に記載のプログラム検証システム。

【請求項 5】

前記検証動作指示がシーケンス実行コードであり、
前記監視部が、前記シーケンス実行コードが所定時間以内に処理されたか否かを判断する、請求項 1 または 2 に記載のプログラム検証システム。

【請求項 6】

前記仮想デバイスに対する前記エミュレータの入出力データを取得するデータ取得部をさらに備え、
前記ブリッジ部内に、前記データ取得部によるエミュレータ入出力データの取得タイミングを制御するデータ取得タイミング制御部をさらに備えた、請求項 1 または 2 に記載のプログラム検証システム。

【請求項 7】

前記ブリッジ部内に、前記監視部により異常動作が発見された場合、検証動作の手順を、前記シナリオファイル記憶部からではなく前記仮想デバイスから入力するよう切り替える切替設定部をさらに備えた、請求項 2 に記載のプログラム検証システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、I C E を用いてマイコンプログラムの動作検証を行うプログラム検証システムであって、特に、プログラム実行に必要な周辺デバイスとして実機デバイスと仮想デバイスとを用いるプログラム検証システムに関する。

【0002】

【従来の技術】

図14に、従来の実機デバッグセットの一例を示す。図14に示すように、シングルチップマイコン制御用アプリケーションプログラムの評価は、ターゲットとなるシングルチップマイコンの代わりに、シングルチップマイコンの全機能をエミュレートできるインサーキットエミュレータ（以下ICEと称する）400をデバッグボード18に組み込んだものを、ハードウェアで構成された実機デバッグセット（周辺デバイス120）に接続して行われていた。

【0003】

しかし、このようなデバッグセットを用いる場合、プログラムのテストを行うためには、システムを動作させるハードウェアが先行完成していることが必須条件であるため、ソフトウェア開発の工程がハードウェア開発の進捗状況に大きく左右されるという問題があった。 10

【0004】

この問題を解決するために、ターゲットとなるシングルチップマイコンのシミュレーションおよび周辺デバイスのシミュレーションを行うことで、実機を使わずに、プログラムを評価する方法が提案されている。

【0005】

このようなシミュレーション方法の一例が、例えば特許文献1（特開平9-114700号公報）に開示されている。特許文献1に開示されたシステムは、テスト／デバッグの対象となる組込みソフトウェアと、マイクロプロセッサユニット（MPU）およびメモリを模擬するMPUシミュレータと、入出力を模擬する入出力装置模擬部と、MPUと1つ以上の入出力装置を接続する信号線を模擬する入出力装置実行制御機構と、シミュレータの起動順序を記述した起動定義ファイルと、I/OモニタとI/Oモデルのプロセスを起動／終了する手段と、ユーザが全てのI/Oモデルのリソース情報を参照・変更できる手段と、I/OモデルとI/Oモニタの接続関係を表示する手段を持つ入出力模擬部操作環境とで構成されている。それぞれの構成要素は、複数のプロセスで構成され、仮想配線にて接続されている。このシミュレーション方法では、図15に示す通り、実機周辺デバイス120を全て仮想デバイス100にて実現する。 20

【0006】

また、実機デバッグ方法とシミュレータデバッグ方法の両方の良いところを合わせた実機・シミュレータ混載システムの一例が、特許文献2（特開平8-6810号公報）に開示されている。特許文献2に開示されたシステムは、図16に示す通り、ICE64を利用した実機デバッグセットと、仮想デバイス（シミュレーション部50）との混載システムである。このシステムでは、シミュレーション部50とICE64との入出力データの受渡しは、専用の実機インターフェースボード60を用いて実現されている。 30

【0007】**【特許文献1】**

特開平9-114700号公報

【0008】**【特許文献2】**

特開平8-6810号公報

【0009】**【発明が解決しようとする課題】**

しかし、従来の実機・シミュレータ混載システムでは、図16に示したように、仮想デバイスとICEとの入出力データの受渡しを実現するために、専用のインターフェースボード60を必要とする。このため、周辺デバイスの仕様変更等が発生した場合は、それに合わせてインターフェースボードを設計変更するために費用と時間が必要となるという問題があった。 40

【0010】

本発明は、このような問題に鑑み、実機のハードウェア開発よりもプログラム開発の方が 50

先行した場合でも実機デバイスの完成を待たずにプログラム検証が可能であって、かつ、動作フリーズ等を起こさずにシステムの自動評価を長時間安定して行えるプログラム検証システムを提供することを目的とする。

【0011】

【課題を解決するための手段】

上記の目的を達成するために、本発明にかかるプログラム検証システムは、マイコンの周辺デバイスとして実機デバイスおよび仮想デバイスを用いて、前記マイコン用プログラムの動作を検証するプログラム検証システムであって、前記マイコン上の前記プログラムの動作をエミュレーションするエミュレータと、前記実機デバイスに対応した検証動作を行うよう前記エミュレータを制御する検証制御部と、前記検証制御部と前記仮想デバイスとの間に設けられ、前記仮想デバイスからの制御信号およびデータを前記検証制御部に適合させるブリッジ部と、前記実機デバイスと前記プログラムとの間の入出力データを格納する実機用RAMと、前記実機用RAMと前記仮想デバイスとの間に設けられた同期化RAMと、前記実機デバイスに対する前記プログラムの入出力動作に同期させて、前記同期化RAMと前記実機用RAMとの間のデータ転送を制御する、仮想・実機同期処理部とを備えると共に、前記ブリッジ部内に、前記検証制御部からの検証動作指示が正常に処理されているかを判断する監視部を備えたことを特徴とする。

【0012】

この構成は、実機のハードウェア開発よりもプログラム開発の方が先行した場合でも、実機デバイスの完成を待たずにプログラム検証を可能とするために、既存の実機デバイスの仕様に対応した検証制御部と、新規プログラムに対応する実機デバイスの仕様をソフトウェアによりシミュレーションする仮想デバイスとの間に、この仮想デバイスからの制御信号やデータを検証制御部に適合させるべく例えばコマンドやデータの変換等を行うブリッジ部を設けたものである。これにより、実機デバイスの完成を待たずに、新規プログラムの検証を行うことが可能となる。また、実機デバイスと仮想デバイスとの動作速度の違いを吸収するために、プログラムが使用する実機用RAMとは別に、同期化を行うための同期化RAMを設ける。さらに、同期化RAMと実機用RAMとのデータ転送を同期制御するために仮想・実機同期処理部を設ける。

【0013】

これにより、例えば、主機能が共通であるがユーザインターフェース周り（操作と表示に関する部分）の仕様にバリエーションを有する、いわゆる展開機種の開発を行うような場合に、新たにデバッグ用の実機セットをハードウェアで作らずとも、既存の実機セットを利用しつつ仮想デバイスでシミュレーションすることが可能となる。また、従来のように、仮想デバイスとICEとの間で入出力データを受け渡しするための、専用のインターフェースボードを作成する必要もない。以上のように、本発明によれば、実機デバッグセットの完成を待たずして、効率良くプログラムの開発を行うことが可能なプログラム検証システムを提供できる。

【0014】

さらに、検証制御部からの検証動作指示が正常に処理されているかを判断する監視部を備えたことにより、異常動作が生じた場合にこれを検知することができ、動作フリーズ等を起こさずにシステムの評価を長時間安定して行うことが可能となる。

【0015】

本発明にかかるプログラム検証システムは、検証動作の手順を記載したシナリオファイルを記憶するシナリオファイル記憶部をさらに備え、前記検証制御部が、前記シナリオファイル記憶部から読み出したシナリオファイルに基づいて前記エミュレータを制御する構成とすることが好ましい。これにより、検証動作手順を手入力する手間が省け、入力ミスも防止できるので、検証作業を効率的にかつ高精度に行うことが可能となる。

【0016】

本発明にかかるプログラム検証システムは、前記検証動作指示がリスタートコマンドであり、前記監視部が、前記リスタートコマンドが所定時間以内に処理されたか否かを判断す

る構成とすることが好ましい。

【0017】

この構成によれば、例えば、検証制御部からリスタートコマンドが発行された後、エミュレータやデバッグ対象プログラムに対して確実にリセット処理がされたかを確認することが可能になる。

【0018】

本発明にかかるプログラム検証システムは、前記検証動作指示がキー実行コードであり、前記監視部が、前記キー実行コードが所定時間以内に処理されたか否かを判断する構成とすることが好ましい。

【0019】

このように、各キーイベントに対してキー処理時間制限を付けることで、デバッグ対象プログラムのバグ等により、デバッグ対象プログラムが無限ループ等に陥った場合に、次の処理が行えなくなることを回避できる。例えば、制限時間内にキー実行コードの処理が行われなければ、次の検査項目の処理を行う様に回避策を入れることにより、検査項目全てを実行することが可能になる。

【0020】

本発明にかかるプログラム検証システムは、前記検証動作指示がシーケンス実行コードであり、前記監視部が、前記シーケンス実行コードが所定時間以内に処理されたか否かを判断することが好ましい。

【0021】

これにより、各シーケンス実行コードに対してシーケンスが実行完了したかどうかの判断を行うことにより、シーケンスの検証動作が停止した場合でも、例えば次のシーケンスの検証を行う等の回避策を講ずることが可能となる。これにより、さらに効率良く検証を行えると共に、自動検証作業の信頼性の向上が可能となる。

【0022】

本発明にかかるプログラム検証システムは、前記仮想デバイスに対する前記エミュレータの入出力データを取得するデータ取得部をさらに備え、前記ブリッジ部内に、前記データ取得部によるエミュレータ入出力データの取得タイミングを制御するデータ取得タイミング制御部をさらに備えたことが好ましい。

【0023】

この構成によれば、検証制御部から送られるキーのメッセージに対してエミュレータ入出力データの取得タイミングの同期がとれるため、例えば、仮想デバイスで模擬される実機の表示状態においてフラッシング等の表示の点滅（オン／オフ）がある場合に、表示画像が「オフ」の時にデータ取得せず、「オン」の時に確実にデータ取得できる。これにより、自動検証精度を向上できる。

【0024】

本発明にかかるプログラム検証システムは、上記のシナリオファイル記憶部を備えた構成において、前記ブリッジ部内に、前記監視部により異常動作が発見された場合、検証動作の手順を、前記シナリオファイル記憶部からではなく前記仮想デバイスから入力するよう切り替える切替設定部をさらに備えたことが好ましい。

【0025】

これにより、シナリオファイルを用いた自動検証作業の途中で異常動作を発見した場合等に、自動検証を一時中断して、仮想デバイスからの入力による手動検証に切り替えることが可能になる。

【0026】

【発明の実施の形態】

最初に、以下の各実施形態で共通する構成および用語について、図1を参照しながら説明する。

【0027】

プログラム414は、マイコン上で動作するプログラムとして開発されたものであり、本

10

20

30

40

50

発明にかかるプログラム検証システムにおいてデバッグ対象となるプログラムである。

【0028】

実機プログラム410とは、デバッグ対象のプログラム414と、同期化プログラム412とを含む。同期化プログラム412は、実機のマイコンには搭載されず、プログラム検証システムにおいてのみ用いられ、後に詳述するが、仮想デバイス100からユーザRAM402へのデータ書き込みを、実機デバイス150からのデータ書き込みと同期させる処理を行う。

【0029】

リスタートとは、シナリオファイル352や仮想入力102から取得するシーケンスキーがリセットのコードであれば、マイコン制御をリセットさせ、自動検証システムを再スタートさせることである。 10

【0030】

周辺デバイス120とは、プログラム414の動作検証に必要な入出力デバイスであり、実機デバイス150と仮想デバイス100とを含む。実機デバイス150とは、プログラム414が搭載される実機の入出力デバイス（ハードウェアそのもの）である。これに対して、仮想デバイス100は、プログラム414が搭載される実機のハードウェアをソフトウェアでシミュレーションするものである。

【0031】

ICE400は、プログラム414を実行検証するために、プログラム414が搭載される実機の組み込みマイコンをエミュレートするインサーキットエミュレータ（i n c i r c u i t e m u l a t o r）であり、マイコンエミュレータ406、ユーザRAM402、および同期化RAM404を含む。ユーザRAM402は、実機デバイス150とプログラム414との間の入出力データを格納するRAMであり、同期化RAM404は、後に詳述するが、仮想デバイス120からユーザRAM402へ書き込むべきデータを一時的に格納するRAMである。 20

【0032】

ICE400の動作は、デバッガ500により制御される。デバッガ500は、ブリッジ504およびICEデバッガ506（検証制御部）を含む。ICEデバッガ506は、実機デバイス150の仕様に対応した検証動作をICE400に実行させるように設計されている。これに対して、ブリッジ504は、実機デバイス150の代わりに仮想デバイス100を入出力デバイスとして用いた検証動作を可能とするために、仮想デバイス100とICE506との間で制御コマンドの変換等を行うものであり、ソフトウェアで実現される。 30

【0033】

すなわち、本実施形態にかかるプログラム検証システムは、実機のハードウェア開発よりプログラム開発の方が先行した場合でも、実機デバイスの完成を待たずにプログラム検証を可能とするために、既存の実機デバイス150およびICEデバッガ506を用いながら、新規プログラム（プログラム414）に対応する実機デバイスの仕様をソフトウェアによりシミュレーションする仮想デバイス100と、この仮想デバイス100をICEデバッガ506に適合させるブリッジ504とを設けたものである。 40

【0034】

仮想配線206とは、仮想デバイス120とデバッガ500、並びに後の実施形態で説明する自動検証制御部等を、仮想的に接続する配線である。仮想デバイス100およびデバッガ500等を複数のパーソナルコンピュータで実現した場合は、この仮想配線206として、ネットワーク、USB、RS232C、IEEE1394を使用できる。また、仮想デバイス100およびデバッガ500等を、一台のパーソナルコンピュータ内で単独実行可能な複数のアプリケーションまたはプロセスとして実現した場合は、仮想配線206は、アプリケーションまたはプロセス間にて通信可能なDDE、COM等のデータ通信機能を利用して実現できる。なお、仮想配線206の具体的な実施形態は、本発明の発明者らによってなされた発明にかかる先の出願（特願2001-219887号）に、詳しく 50

開示されている。

【0035】

仮想配線206による通信方法は、非同期通信または同期通信のどちらでもよい。非同期通信とは、データを通信相手に一方的に送信するだけで、返信がない通信方法である。同期通信とは、データを通信相手に送信した後、通信相手から返信データが返ってくるまで待っている通信方法である。

【0036】

HTAとは、HTML Applicationsの略で、Webページで実行可能なさまざまな機能（HTML、CSS、スクリプト言語（Java（登録商標、以下略）Script、VBA等）、Behavior）をサポートする。

10

【0037】

ActiveXとは、Internet対応のOLEドキュメントオブジェクトであるActiveXドキュメントや、Internetアクセス機能を持ったクライアントアプリケーションを開発するためのAPIである。

【0038】

OLEとは、Object Linking and Embeddingの略で、アプリケーションが別のアプリケーションの機能と呼出して自分自身の機能のように利用したりできる。

【0039】

（第1実施形態）

20

以下、本発明の一実施形態について、図1を用いて説明する。

【0040】

本実施形態にかかるプログラム検証システムは、長時間動作を安定させ、検証システムの停止やデータ取得失敗を起こさないで、プログラム自動検証を行うことを目的とする。このため、図1に示すとおり、本プログラム検証システムは、検証結果ファイル350および記録制御部302と、シナリオファイル352と、これに対する読み書きを制御するシナリオ制御部314を備えている。なお、記録制御部302およびシナリオ制御部314により、プログラム自動検証を実現するための自動検証制御部300が構成されている。

【0041】

シナリオファイル352とは、プログラム414を自動的に検証するための操作手順（シナリオ）を記録したファイルであり、検証結果ファイル350と同様に、例えば、Microsoft社のExcel等を利用して実現することができる。自動検証制御部300の制御プログラムは、HTAを利用して記述されている。また、検証結果ファイル350およびシナリオファイル352をExcelで作成する場合、これらの読み書きを自動検証制御部300から行うために、OLE機能を利用する。また、自動検証制御部300と仮想配線206との間のデータ通信は、ActiveXを利用して実現できる。

30

【0042】

以下、本プログラム検証システムの構成および動作について、具体的な説明を行う。

【0043】

本プログラム検証システムは、ICE400内の表示用イメージデータの変化に基づいて、仮想出力104の表示を更新する。仮想出力104の表示画像は、クリップボード200および検証結果書込部304を経由して検証結果ファイル350に記録され、予め検証結果ファイル350に記録されていた検証用の期待値データと自動比較される構成となっている。

40

【0044】

＜シナリオからICEへのデータ転送の自動シーケンス＞

以下、本プログラム検証システムの自動検証処理について説明する。自動検証処理は、自動検証制御部300が、シナリオファイル352からキーデータ等を読み出し、ICE400へ自動的に転送して実行させ、その実行結果を検証結果ファイル350へ書き込んで評価することにより行う。

50

【0045】

シナリオファイル352には、動作検証に必要な操作情報（キーデータ等）が、検証手順に従ってあらかじめ記録されている。シナリオファイル352は、Excelファイルを利用して、例えば、図12に示すような状態遷移表として作成される。本プログラム検証システムでは、シナリオファイルに記述されている動作シーケンスに従って画面モードを所定の状態にした後、イベントであるキーの発行を実行して、各キー入力に対するアクション（有効または無効）の確認を行う。例えば、図12に示すシナリオファイル352の場合、図12中に示した実行順番に従い、まず、画面モードを"MDPWRON"にした状態で、"KPOWER"、"KCH UP"、"KVOL UP"、"KMENU"、"KSET"のキー入力を順次行い、これらの各キー操作に対する仮想デバイスおよび実機デバイスの動作を検証する。そして、次に、画面モードを"MDMENU"とした状態で、上述と同様に"KPOWER"、"KCH UP"、"KVOL UP"、"KMENU"、"KSET"のキー入力を順次行い、仮想デバイスおよび実機デバイスの動作を検証する。以降、"MDTIMER"、"MDSET"、"MDTEST"の各画面モードについて、同様に検証を行う。

【0046】

ここで、図12に示すシナリオファイル352を用いた自動検証処理の詳細について、画面モード"MDMENU"の状態での各キー入力に対する動作検証を行う場合を例にとり、さらに具体的に説明する。自動検証制御部300は、以下の(1)～(9)の手順により、画面モード"MDMENU"の状態でのキー入力に対する動作検証を行う。

【0047】

(1) 同期化RAM400内のリセット完了モニタ用フラグRAMに"1"をセットする。

【0048】

(2) ICE400にプログラム414を初期化させるために、シナリオ読込部316からデータ送信部320へ、リセットGOコマンド発行を指示する。なお、"リセットGO"コマンドとは、ICE400に実機プログラム410の実行を停止させ初期化する"リセット"コマンドと、ICE400に実機プログラム410を最初から実行開始させる"GO"コマンドとを一連に実行させるものである。データ送信部320は、同期通信により、ブリッジ504を経由してICEデバッガ506に対してリセットGOのメッセージを送る。ICEデバッガ506が、ICE400をリセットGO実行させることにより、実機プログラム410がリセットスタートされる。実機プログラムリセットスタート実行の先頭プログラムで、前記のリセット完了モニタ用フラグRAM値を"0"にする。ICE400がリセットGO完了すれば、ブリッジ504からデータ送信部320を経由して、シナリオ読込316部にリセットGO完了通知を行う。

【0049】

(3) リセット完了モニタ用フラグRAM値が"0"になったことを確認した後、画面モードを"MDMENU"にするために、シナリオファイル352において当該画面モードの動作シーケンスに記述されているキーをシナリオ読込部316により順次読み込み、発行する。発行されたキーは、データ送信部320、バススロット312、仮想配線206、デバッガ500を介して、ICE400へ渡される。図12の例では、"MDMENU"の動作シーケンスの最初の動作（図12のシナリオファイルにおける動作1の欄）として"KPOWER"というキーが記述されているので、このキーが最初に発行される。

【0050】

(4) "KPOWER"キーの処理を実機プログラム410が完了したかを、ユーザRAM402内の特定RAMにおける値の変化をチェックすることにより、確認する。特定RAMの変化が確認できると、次に、動作シーケンスの動作2に記述されている"KMENU"キーを発行し、上記と同様に、ユーザRAM402内の特定RAMの変化をチェックすることにより、"KMENU"キーの処理が完了したかを確認する。以上の処理を、シナリオファイル352の動作シーケンスに記述されている最後のキーまで、繰り返し実行

する。図 1 2 に示す " M D M E N U " の場合、" K P O W E R " キーおよび " K M E N U " キーの実行により、目的の画面モードである " M D M E N U " になる。

【 0 0 5 1 】

(5) 目的の画面モードになったら、まず、ユーザ R A M 4 0 2 内の画像データを同期化 R A M 4 0 4 に取り込む。

【 0 0 5 2 】

(6) 次に、シナリオファイル 3 5 2 に記述されている発行キーを、順次発行し、動作を検証する。図 1 2 の例では、最初に、" K P O W E R " キーを発行する。" K P O W E R " キーの処理が完了したことを、ユーザ R A M 4 0 2 内の特定 R A M 値の変化で確認する。

10

【 0 0 5 3 】

(7) " K P O W E R " キーの処理が完了したことが確認できたら、この " K P O W E R " キーの処理結果を検証する。仮想出力 1 0 4 は、画面変化の有無を、イベント検出部 3 0 8 に伝える。イベント検出部 3 0 8 は、画面変化の有無が、シナリオファイル 3 5 2 に記述されている有効・無効仕様に合致するかを判断する。

【 0 0 5 4 】

例えば、図 1 2 に示すシナリオファイルの場合、○記号は、当該画面モードにおける当該キー発行が有効であることを表し、×記号は、当該画面モードにおける当該キー発行が無効であることを表す。なお、「キー発行が有効である」とは、そのキー発行により画面上に何らかの変化が生じることをいい、「キー発行が無効である」とは、そのキー発行によっても画面に変化が生じないことをいう。

20

【 0 0 5 5 】

ここでは、" M D M E N U " の列と " K P O W E R " の行との交点にあたる欄には○記号が記述されているので、画面モードが " M D M E N U " である場合に " K P O W E R " キーを発行した場合、このキーが正しく処理されたならば、何らかの画面変化が生じるべきである。従って、" K P O W E R " キーの発行により画面変化が生じていれば、イベント検出部 3 0 8 は、" M D M E N U " 画面モードにおける " K P O W E R " キーの有効・無効仕様は満たされているものと判断する。

【 0 0 5 6 】

イベント検出部 3 0 8 は、有効・無効仕様が満たされているか否かの判断結果に応じて、シナリオファイル 3 5 2 の有効・無効仕様欄に色付けする。すなわち、イベント検出部 3 0 8 が、シナリオファイル 3 5 2 に記述されている有効・無効仕様が満たされていると判断すれば、検証結果書込部 3 0 4 を介して、シナリオファイル 3 5 2 の当該有効・無効仕様欄の表示属性を例えば青色に設定し、満たされていないと判断した場合は例えば赤色に設定することにより異常動作であることが分かるようにする。これにより、デバッグ作業者は、シナリオファイル 3 5 2 の有効・無効仕様欄の表示色により、目標仕様が満たされているかを容易に識別することが可能となる。また、ここでは、有効・無効仕様の検証結果をシナリオファイル 3 5 2 に記録することとしたが、検証結果ファイル 3 5 0 に、有効・無効仕様の検証結果を記録する欄を別途設けるようにしてもよい。

30

【 0 0 5 7 】

(8) また、検証結果書込部 3 0 4 は、画像データ取得部 3 1 0 によりユーザ R A M 4 0 2 から取得した画面データ（実行画像データ）を、検証結果ファイル 3 5 0 へ書き込む。さらに、検証結果書込部 3 0 4 は、前記実行画像データと、検証処理の実行に先立って予め作成されている基準画像データとを比較し、比較結果を検証結果ファイル 3 5 0 へ書き込む。基準画像データとは、キーが正しく処理された結果として表示されるべき画像データのダンプデータであり、図 1 3 に示すように検証結果ファイル 3 5 0 へ書き込まれている。例えば図 1 3 に示した例にかかる検証結果ファイル 3 5 0 では、基準画像データと実行画像データとの比較結果が一致すれば○記号、一致しなければ×記号が、「判定」の欄に書き込まれている。これにより、デバッグ作業者は、検証結果ファイル 3 5 0 の判定欄を見るだけで、キー発行により所定の画面データが表示されたか否かを容易に検証できる

40

50

。

【0058】

(9) また、検証結果ファイル350には、基準画像データに基づく画面イメージ（基準画面）が予め貼り付けられている。検証結果書込部304は、図13に示すように、この基準画面の隣の列に、仮想出力104に出力された画面（実行画面）を、クリップボード200を介して貼り付ける。これにより、デバッグ作業者は、検証結果ファイル350を見るだけで、基準画面と実行画面との違いを容易に識別することが可能となる。

【0059】

” K P O W E R ” キーについて上記の(9)までの処理が終了したら、(5)へ戻り、(6)においてシナリオファイルに記述されている次の発行キー” K C H U P ”を発行し、以降の処理を行う。同様にして、シナリオファイルに記述されている全ての発行キーについて、(5)～(9)の処理を繰り返すことにより、図13に示すような検証結果ファイルが完成する。 10

【0060】

また、本実施形態では、自動検証処理中の何らかの要因により I C E 4 0 0 が異常動作になった場合に、自動検証処理が続行できなくなることを防ぐため、既存デバッガ562（図2参照）が I C E 4 0 0 の挙動を監視する。そして、I C E 4 0 0 の動作に変化が発生した場合は、図11に示すようなメッセージを、I C E デバッガ506が、ブリッジ504内の I C E 状態モニタ558（図2参照）に伝える。さらに、このメッセージは、I C E 状態送信部556から、仮想配線206を介して、シナリオ制御部314内のタイミング制御部318に伝えられる。これにより、タイミング制御部318は、I C E 4 0 0 の動作変化に合わせて、シナリオ読込部316にシナリオファイル352のデータを再発行させる等の処理を行う。 20

【0061】

なお、自動検証処理において、シナリオファイル352として前記の状態遷移表を用いる代わりに、シナリオファイル352の記述方法や自動検証制御部300の制御方法を適宜変更することで、任意の自動検証処理を実現可能である。

【0062】

＜具体的動作＞

以下、プログラム自動検証を行う本プログラム検証システムの動作について、さらに具体的に説明する。 30

【0063】

＜初期化処理＞

本実施形態のプログラム検証システム（図1に示す構成）は、起動すると、まず、初期化処理を行う。仮想デバイス100、デバッガ500、及び自動検証制御部300は、仮想配線206に接続するための接続処理を行う。また、ブリッジ504は、予め、データ送信部320および仮想出力104からのメッセージ受信を行うために、受信割込み許可しておく。

【0064】

＜シナリオ読込から仮想配線へリセット G O コマンドデータを流す＞ 40

シナリオ読込部316から、I C E 4 0 0 の実行を初期化するために、リセット G O コマンドの送信をデータ送信部320に指示する。データ送信部320は、図3に示すようなリセット G O コマンドのフォーマットに従って作成したメッセージを、ブリッジ504に宛てて、バススロット312から仮想配線206に同期通信にて送出する。

【0065】

＜ブリッジデバイスの書込み処理＞

データ送信部320からデバッガ500のバススロット502経由でブリッジ504に送られて来たメッセージは、図2に示す通り、受信割込み処理を行う同期受信部550に受け取られる。コマンド変換部552は、図3に示すようなテーブルを参照し、受け取ったメッセージが、1バイトメモリライト、1バイトメモリリード、I C E リセット、I C E 50

実行等のどのメッセージであるかを分析し、その結果をコマンド発行部 554 から I C E デバッガ 506 内の V i r t u a l L i n k 560 に渡す。

【0066】

V i r t u a l L i n k 560 は、既存デバッガ 562 を制御して、I C E 400 にリセットコマンドを発行した後、実行開始コマンドを発行して、実行完了通知を同期受信部 550 の返信データとして、メッセージの送信元であるデータ送信部 320 に返信する。データ送信部 320 は、シナリオ読込部 316 にリセット G O コマンド実行完了を伝え、シナリオファイル 352 の内容に従った実行を行う。

【0067】

＜シナリオファイルから仮想配線へのデータを流す＞

10

シナリオ読込部 316 は、シナリオファイル 352 からキーコードを読込んで、データ送信部 320 に指示して、図 3 に示すようなメッセージ一覧表内の 1 バイトメモリライトコマンド発行フォーマットに従って作成したメッセージを、ブリッジ 504 に宛てて、バススロット 106 から仮想配線 206 へ同期通信にて送出する。

【0068】

＜ブリッジの書き込み処理＞

データ送信部 320 から、バススロット 502 経由でブリッジ 504 に送られて来たメッセージは、図 2 に示す通り、受信割込み処理を行う同期受信部 550 に受け取られる。そして、コマンド変換部 552 が、図 3 に示すようなテーブルに基づき、受け取ったメッセージが、1 バイトメモリライト、1 バイトメモリリード、I C E リセット、I C E 実行等のどのメッセージであるかを分析する。その分析結果は、コマンド発行部 554 から I C E デバッガ 506 内の V i r t u a l L i n k 560 に渡される。

20

【0069】

V i r t u a l L i n k 560 は、既存デバッガ 562 を制御して、プログラム 414 を実行中の I C E 400 内の同期化 R A M 404 に、仮想入力 102 から渡された指定 R A M アドレスの R A M データをオンザフライにて書込んだ後、書き込み完了通知を、同期受信部 550 の返信データとして、メッセージの送信元であるデータ送信部 320 へ返信する。返信を受けると、キーデータを配信したことを、実行状況配信部 322 からイベント検出部 308 に伝える。

【0070】

30

＜リセットコードの判定およびコマンド変換処理＞

自動検証制御部 300 から、仮想配線 206 およびバススロット 502 経由でブリッジ 504 に送られて来たリセットコードは、図 2 に示す通り、ブリッジ 504 内で受信割込み処理を行う同期受信部 550 で受け取られる。ここで、リセットコードを受け取ったコマンド変換部 552 が行うコマンド変換処理の内容を、図 6 に示す。

【0071】

図 6 に示すように、コマンド変換部 552 は、受け取ったリセットコードの種類を判断し（ステップ S 60）、リセットコードの種類に応じた処理を行う。

【0072】

（1）リスタートコードの場合

40

受け取ったリセットコードがリスタートコードであった場合、コマンド変換部 552 は、リスタート初期化処理（ステップ S 611）を行い、コマンド変換部 552 内に設けられたリスタート判定部（図示せず）により、リスタートの完了確認フラグが一定時間以内にセットされたか否かを判定するリスタート判定処理（ステップ S 612）を行う。リスタート判定処理の結果は、「受付異常」または「リスタート O K」のいずれかを表す返信データとして、コマンド発行部 554 へ返信される（ステップ S 69）。

【0073】

リスタート判定処理の内容を、図 7 に示す。リスタート初期化処理（ステップ S 611）の後、図 7 に示すように、リスタートの完了確認フラグがセットされたか判断し（ステップ S 612 a）、フラグがセットされていれば、返信データの内容に、「リスタート O K

50

」を表すデータをセットする（ステップS 6 1 2 b）。フラグがセットされていなければ、予め設定された所定の時間以内であるか否かを判断し（ステップS 6 1 2 c）、時間内であればステップS 6 1 2 aへ戻る。前記所定の時間を越えた場合は、返信データの内容に、「受付異常」を表すデータをセットする（ステップS 6 1 2 d）。

【0074】

（2）キー実行コードの場合

受け取ったリセットコードがキー実行コードであれば、コマンド変換部552は、イベントキーの発行初期化処理（ステップS 6 2 1）を行い、コマンド変換部522内に設けられたキー実行判定部（図示せず）により、イベントキー発行が一定時間以内に行われたか否かを判定するキー実行判定処理（ステップS 6 2 2）を行う。キー実行判定処理の結果は、コマンド発行部554へ返信される（ステップS 6 9）。 10

【0075】

キー実行判定処理の内容を、図8に示す。イベントキーの発行初期化処理（ステップS 6 2 1）の後、図8に示すように、イベントキーが発行されたか判断し（ステップS 6 2 2 a）、イベントキーが発行されていれば、コマンド発行部554への返信データとして表示画像を取得し（ステップS 6 2 2 b）、コマンド変換部522内に設けられたフラグセット部（図示せず）によってフラッシングフラグをセットする（ステップS 6 2 2 c）。ステップS 6 2 2 aにおいてイベントキーが発行されていないと判断されれば、予め設定された所定の時間以内であるかを判断し（ステップS 6 2 2 d）、時間内であればステップS 6 2 2 aへ戻る。前記所定の時間を越えた場合は、コマンド発行部554への返信データの内容に、「応答なし」を表すデータをセットする（ステップS 6 2 2 e）。 20

【0076】

なお、上述のフラッシングフラグは、検証プログラムにフラッシング表示仕様があれば、表示をフラッシングできるようにするためのフラグであり、図10に示すように、実機プログラム内のリセット時の初期化処理の時に、フラグクリア部によってクリアされる。なお、フラグクリア部は、図2に示すコマンド変換部552内に設けられる（図示省略）。

【0077】

（3）シーケンス実行コードの場合

受け取ったリセットコードがシーケンス実行コードであれば（ステップS 6 3 0）、コマンド変換部552は、キー発行処理（ステップS 6 3 1）を行い、コマンド変換部522内に設けられたシーケンス実行判定部（図示せず）により、キー発行が一定時間以内に行われたか否かを判定するシーケンス実行判定処理（ステップS 6 3 2）を行う。シーケンス実行判定処理の結果は、「シーケンス実行異常」または「シーケンス実行OK」のいずれかを表す返信データとして、コマンド発行部554へ返信される（ステップS 6 9）。 30

【0078】

シーケンス実行判定処理の内容を、図9に示す。キー発行処理（ステップS 6 3 1）の後、図9に示すように、キーが発行されたか判断し（ステップS 6 3 2 a）、キーが発行されていれば、返信データの内容に、「シーケンス実行OK」を表すデータをセットする（ステップS 6 3 2 b）。キーが発行されていなければ、予め設定された所定の時間以内であるか否かを判断し（ステップS 6 3 2 c）、時間内であればステップS 6 3 2 aへ戻る。前記所定の時間を越えた場合は、返信データの内容に、「シーケンス実行異常」を表すデータをセットする（ステップS 6 3 2 d）。 40

【0079】

以上の（1）～（3）に説明したように、コマンド変換部552によりリセットコードに応じた返信データが作成され、コマンド発行部554へ返信される。コマンド発行部554は、その分析結果を、ICEデバッガ506内のVirtual Link 560に渡す。

【0080】

Virtual Link 560は、実機デバイス150の仕様に対応した検証動作を行うように設計されている既存デバッガ562を制御して、プログラム実行中のICE 400 50

内の同期化RAM 404に、仮想入力102から渡された指定RAMアドレスのRAMデータを、オンザフライにて書込む。その後、同期受信部550が、前記メッセージへの返信としての書込み完了通知を、メッセージの送信元である仮想入力102へ返信する。なお、「オンザフライ」とは、プログラム414が実行中でも、ICE 400内のユーザRAM 402と同期化RAM 404のRAMデータ、およびマイコンエミュレータ406内のレジスタの読書きを可能とする、ICE 400の機能である。

【0081】

このようにすれば、シーケンス実行時に、万一なんらかの理由で動作がフリーズした場合でも検証システムが止まることなく、長時間安定した動作検証することができ、検証精度を向上させることができる。

10

【0082】

また、各テスト項目を実行する前にリセットを掛ければ、前テスト項目においてデバッグ対象プログラムのバグ等にて無限ループに陥った場合でもリスタートが可能となり、各項目のテストを行う事が可能になる。

【0083】

＜同期化RAMからユーザRAMへの転送処理＞

同期化プログラム412は、同期化RAM 404の変化を、プログラム414に必要なタイミングで、マイコンエミュレータ406にチェックさせている。

【0084】

同期化プログラム412は、例えばキー入力の場合は、プログラム414のキー入力処理の直後に呼ばれ、同期化RAM 404内のキーデータRAMの変化を、マイコンエミュレータ406経由でチェックし、もし変化があれば、キーデータをユーザRAM 402のキーデータ確定エリアに転送する。この際、仮想入力102のキーデータと実機デバイス150からのキーデータとが同時に入力された場合は、仮想入力102からのキーデータが優先される。

20

【0085】

同期化プログラム412内のキー入力処理は、図4に示すように、仮想キー入力、実機キー入力より優先されて処理される。

【0086】

＜ユーザRAMから仮想出力へのデータ転送＞

シナリオファイル352から読み込んだキーコードによってプログラム414がユーザRAM 402内の出力RAMデータ（仮想出力104への出力データ）を変更した場合、この出力RAMデータを仮想出力104へ転送する方法は、以下の(c1)～(c4)のとおりである。

30

【0087】

(c1) プログラムのユーザRAM変更；

プログラム414により、ユーザRAM 402内の仮想出力104に関係するRAMエリアが変更された直後に、ユーザRAM 402から同期化RAM 404内の特定RAMに変化を伝えるための処理を行う。具体的には、図5に示す通り、プログラム414から仮想同期化ルーチン呼び出すことにより、処理を開始する(ステップS420)。実機デバイス150への出力処理(ステップS422)を行った直後に、実機デバイス150への出力RAMデータの変化を検出し(ステップS424)、変化があれば、出力RAMデータをユーザRAM 402から同期化RAM 404内に転送する(ステップS426)。

40

【0088】

(c2) 仮想出力から仮想配線へデータを流す；

次に、仮想出力104から、同期化RAM 404内の出力RAMデータの変化イベントを読込むために、仮想出力104からデバッグ500へのメッセージ転送の指定とイベントを読込むためのRAMアドレスとを一緒に、バススロット108経由で、仮想配線206に同期通信により送出する。

【0089】

50

(c3) ブリッジデバイスの読み込み処理；

仮想出力104からバススロット502経由でブリッジ504に送られて来たメッセージは、図2に示す通り、受信割込み処理を行う同期受信部550で受け取られる。その後、コマンド変換部552が、図3に示すようなメッセージ一覧表に従い、受け取ったメッセージが、1バイトメモリライト、1バイトメモリリード、ICEリセット、ICE実行等のどのメッセージであるかを分析する。コマンド発行部554は、その分析結果を、ICEデバッガ506内のVirtual Link 560に渡す。Virtual Link 560は、既存デバッガ562を制御して、プログラム実行中にICE400内の同期化RAM404内の出力RAMデータをオンザフライにて読み込む。その後、読み込んだ出力RAMデータを、同期受信部550が、前記メッセージの送信元である仮想出力104へ返信する。これにより、出力RAMデータの仮想出力104への転送が、完了する。

10

【0090】

(c4) 仮想出力の表示処理；

仮想出力104では、出力RAMデータを元に、パソコンモニタに、実機液晶画面を模した仮想表示等を行う。また、同期化RAM404またはユーザRAM402から仮想出力104に取得した画像データの変化があれば、仮想出力104は、画像データ取得部310に対して、画像変化のメッセージおよび仮想出力104の表示ダンプデータを転送する。

【0091】

また、前記イベントを使わずに、仮想入力102から一定周期にて、デバッガ500にユーザRAM402のデータを読み込み、仮想出力104にて実機液晶画面を模した仮想表示等を行う方法でも良い。

20

【0092】

＜データ転送からイベント検出に転送＞

データ送信部320から前記キーデータが送られてくると、そのキーデータは、イベント検出部308から画像データ取得部310に渡される。また、同期化RAM404またはユーザRAM402から仮想出力104に取得した画像データの変化があれば、仮想出力104を経由して、画像データ取得部310に対して画像変化のメッセージおよび、仮想出力の表示ダンプデータを転送する。画像データ取得部310は、前記キーデータを検証結果書込部304から検証結果ファイル350に書込む。

30

【0093】

また、画像データ取得部310は、仮想出力104から表示変化のメッセージを受けると、仮想出力104の表示画像を、クリップボード200にキャプチャする。クリップボード200にキャプチャされた画像は、検証結果書込部304により、検証結果ファイル350に対して書込まれる。

【0094】

具体的には、図13に示すように、キーデータは操作キーの欄に書込み、シナリオファイル352の画面モードは、画面モードの欄に書込む。

【0095】

＜基準画像データと実行画像データとの比較＞

40

検証結果ファイル350において、「基準画像データ」と「実行画像データ」とのデータの相違を判定し、判定結果を「判定」の欄に「○」と「×」として書込む。なお、検証結果ファイル350をExcelで作成する場合は、IF関数を用いて「基準画像データ」と「実行画像データ」の相違の判定を行うことができる。

【0096】

＜基準画像データの作成＞

尚、自動検証処理を行うための基準画像およびその表示ダンプデータの作成については、前述したように画像データおよび表示ダンプデータを図13に示す基準画面および実行画像データに書込むことで作成できる。

【0097】

50

＜表示の出力抑制処理＞

また、本実施形態のプログラム検証システムでは、図10（b）に示すように、表示同期判定プログラムによって、実機プログラム内の出力処理でフラッシングフラグがセットされていれば仮想出力104へ出力しないようにする。なお、表示同期判定プログラムは、実機プログラム内に検証用プログラムとして埋め込む。このようにすれば、フラッシングフラグのタイミングにより表示出力するため、画像取得のタイミングが同期化される。このことにより、自動検証制御部300から来るキーのメッセージに対して同期がとれるため、フラッシング等の表示の点滅がある場合に、仮想出力104の画像表示の記録時と完全に同期がとれるために、表示画像を「滅」の時にキャプチャーせず、「点」の時に確実にキャプチャーできるため、自動検証精度を向上できる。

10

【0098】

＜仮想入力からの表示の出力抑制処理＞

なお、仮想デバイス100からの入力からも画像キャプチャーの同期を行えるようにするために、図6に示すように、コマンド変換部552において通常のキーコードを受理した場合は、キー発行処理（ステップS641）およびキー受理処理（ステップS642）を行い、仮想デバイス100からの入力がリセットキーであれば（ステップS643）、コマンド変換部552に設けられた表示同期化フラグクリア部（図示せず）によってフラッシングフラグをクリアする（ステップS644）。なお、ステップS642のキー受理処理とは、キー受付をしたかどうかを判定する処理である。これにより、仮想デバイス100からのリセット時にも画像取得を同期化することができる。

20

【0099】

このようにすれば、自動検証途中に異常動作を発見したときなど、自動検証を一時中断して手動検証する場合に、仮想デバイスからの入力に対して、通常の動作に切り替えて手動検査を行う事が可能になり、検証精度を向上させることができる。

【0100】

以上のとおり、本実施形態によれば、長時間安定した動作が可能なプログラム自動検証システムを実現することができる。また、表示画像取得をキーと同期させることにより、画像のキャプチャーずれを防止することができる、これにより、検証精度および作業効率が向上する。

【0101】

【発明の効果】

以上のように、本発明によれば、万一なんらかの理由で動作がフリーズした場合でも自動検証システムが停止することなく、長時間安定して動作検証を行えると共に、検証精度を向上させることができる。

30

【図面の簡単な説明】

【図1】本発明の一実施形態にかかる実セット・シミュレータ混載のプログラム自動検証システムを示すブロック図

【図2】デバッガの内部構成を示すブロック図

【図3】仮想デバイスからブリッジへのメッセージ一覧の説明図

【図4】仮想から実機同期化フローチャート

40

【図5】実機から仮想同期化フローチャート

【図6】本発明の一実施形態にかかるプログラム自動検証システムにおけるコマンド変換処理の内容を示すフローチャート

【図7】図6のリスタート判定処理の内容を示すフローチャート

【図8】図6のキー実行判定処理の内容を示すフローチャート

【図9】図6のシーケンス実行判定処理の内容を示すフローチャート

【図10】（a）および（b）は本実施形態のプログラム自動検証システムにおける入出力処理のフローチャート

【図11】ICEから仮想デバイスへ向けてのイベント一覧を示す説明図

【図12】シナリオファイルの一例を示す説明図

50

【図 1 3】本検証結果ファイルの一例を示す説明図

【図 1 4】従来のプログラム検証システムの一部を示すブロック図

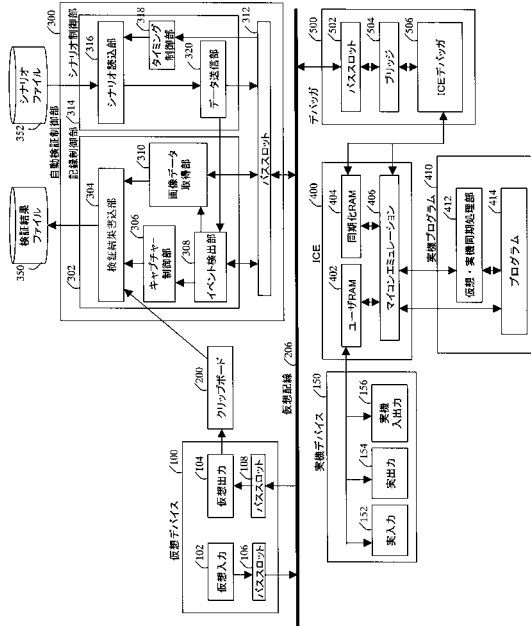
【図 1 5】従来のプログラム検証システムの一部を示すブロック図

【図 1 6】従来のプログラム検証システムの一部を示すブロック図

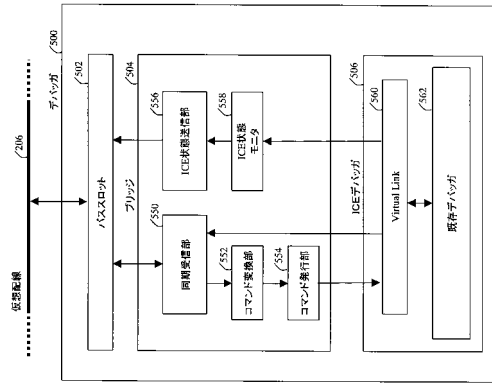
【符号の説明】

1 0 0	仮想デバイス	
1 0 2	仮想入力	
1 0 4	仮想出力	
1 2 0	周辺デバイス	
1 5 0	実機デバイス	10
1 5 2	実入力	
1 5 4	実出力	
1 5 6	実機入出力	
2 0 0	クリップボード	
2 0 6	仮想配線	
3 0 0	自動検証制御部	
3 0 2	記録制御部	
3 0 4	検証結果書込部	
3 0 8	イベント検出部	
3 1 0	画像データ取得部	20
3 1 2	バススロット	
3 1 4	シナリオ制御部	
3 1 6	シナリオ読込部	
3 1 8	タイミング制御部	
3 2 0	データ送信部	
3 5 0	検証結果ファイル	
3 5 2	シナリオファイル	
4 0 0	I C E	
4 0 2	ユーザ R A M	
4 0 4	同期化 R A M	30
4 1 0	実機プログラム	
4 1 2	同期化プログラム	
5 0 0	デバッガ	
5 0 2	バススロット	
5 0 4	ブリッジ	
5 0 6	I C E デバッガ	
5 2 2	コマンド変換部	
5 2 4	コマンド発行部	

【図 1】



【図 2】

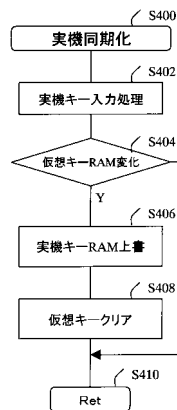


【図 3】

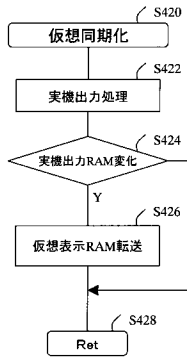
メッセージ一覧表

コマンド名	iMsg	Send Val	Send Len	Rev Val	Rev Len	RetVal
1バイトのメモライト	1	address	4	0	0	WrData
複数バイトのメモライト	2	nWriteDat	WriteValSiz	0	0	0
1バイトのメモリード	3	address	4	0	0	ReData
複数バイトのメモリード	4	nReadDat	12	nReadDat	0	0
レジスタ書き込み	5	RegName	RegNameSize	0	0	WrData
レジスタ読み出し	6	RegName	RegNameSize	0	0	ReData
リセット	7	0	0	0	0	0
リセットGO実行	8	0	0	0	0	0
強制ブレーク	9	0	0	0	0	0
ICEの状態チェック	10	0	0	0	0	Stat
ソフトウェアブレーク設定	11	0	0	0	0	BoAd
ソフトウェアブレーク解除	12	0	0	0	0	BoNo
シンボルアドレス取得	13	Symbol	0	0	0	SymbolAdrs
アドレスシンボル変換	14	0	0	Symbol	0	SymbolAdrs
実行通知コールバック設定	15	RevFunc	0	0	0	0
デバッグコマンド実行	16	IceCmd	IceCmdSize	0	0	0

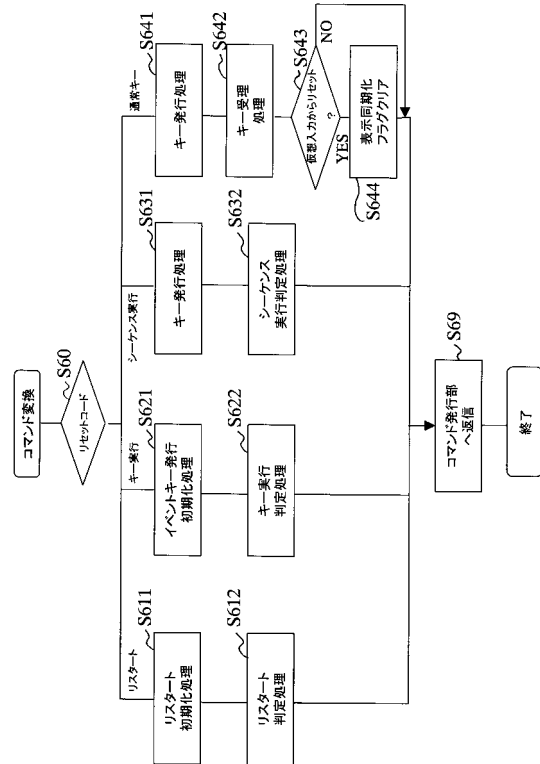
【図 4】



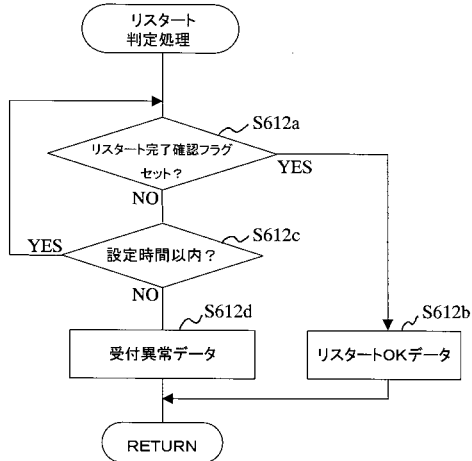
【図 5】



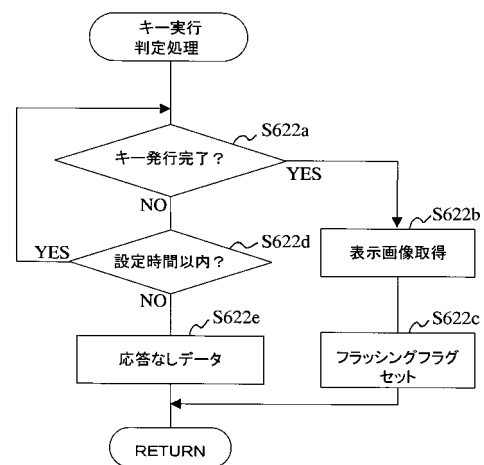
【図 6】



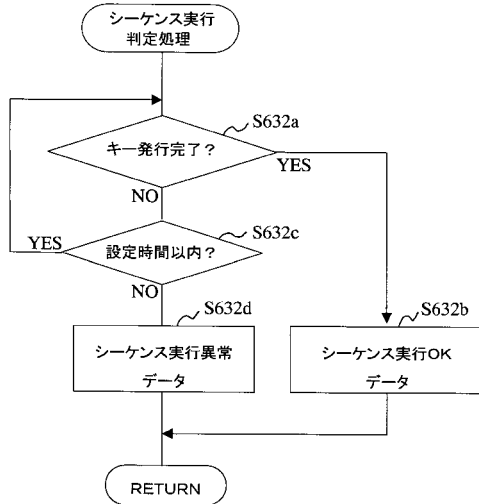
【図 7】



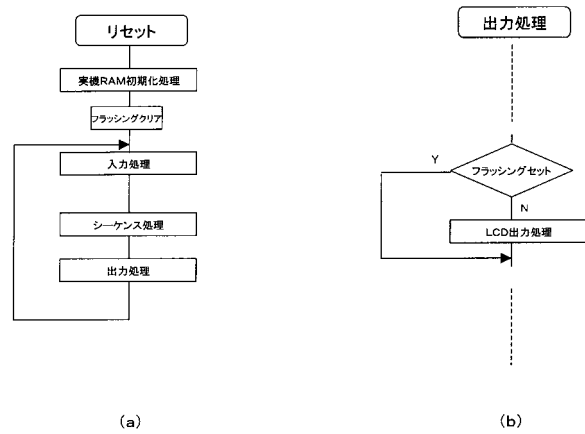
【図 8】



【図 9】



【図 10】

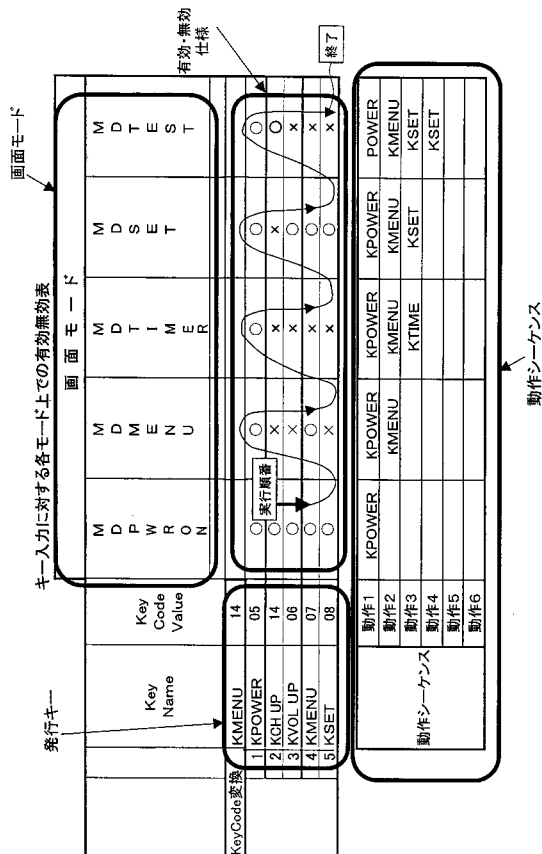


【図 11】

イベント一覧表

iMsg	コマンド名
1	ICE実行開始直前
2	ICE実行開始直後
3	ICEブレーク直後
4	ICEリセット直後
5	ICEモジュールのロード直後

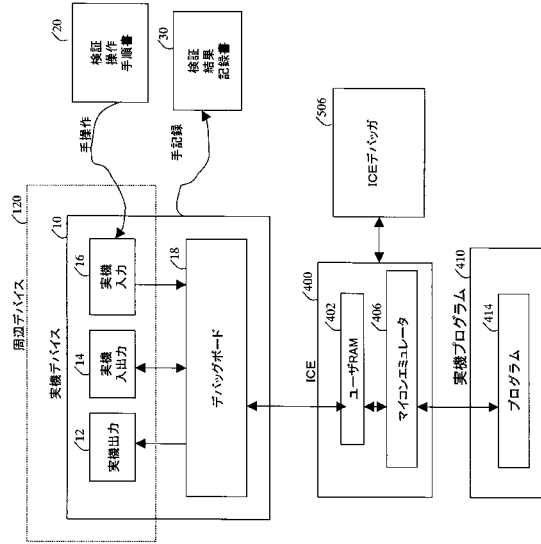
【図 12】



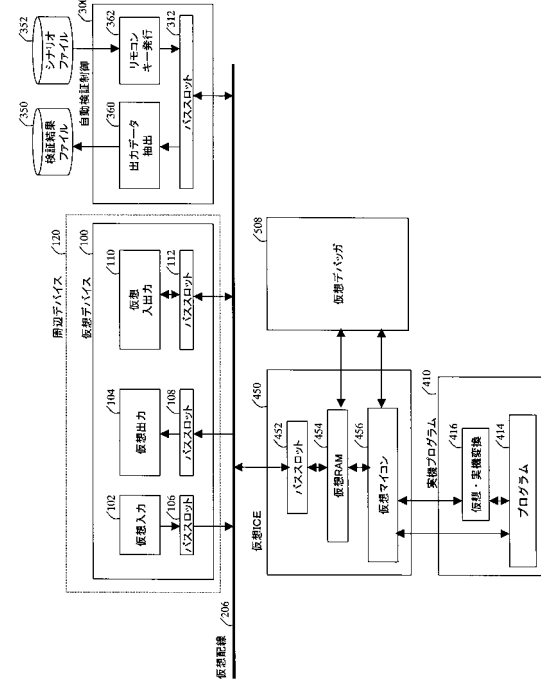
【図 13】

画面モード	操作キー	判定	基準画面	実行画面	基準画像データ	実行画像データ
MDPWON	KPOWER	○			E3FE3FE3FE018003FE3 3C463C503FE3C423C4A3C6 C343C503FE3C423C4A3C6B3 C5E3FE3C6B3C4A3C6B3FE3	E3FE3FE3FE018003FE3 3C463C503FE3C423C4A3C6 C343C503FE3C423C4A3C6B3 C5E3FE3C6B3C4A3C6B3FE3
MDPWON	KCH UP	○	CH03	CH03	FE3C6B3 3C523C423C5C3FE3C4A3C5C C343C503FE3C423C4A3C6B3 FE3C343C5E3C4C3C4C3FE3	FE3C6B3 3C523C423C5C3FE3C4A3C5C C343C503FE3C423C4A3C6B3 FE3C343C5E3C4C3C4C3FE3
MDPWON	KVOL UP	○	CH03	CH03 Vol 24	FE3C6B3 3C523C423C5C3FE3C4A3C5C C343C503FE3C423C4A3C6B3 FE3C343C5E3C4C3C4C3FE3	FE3C6B3 3C523C423C5C3FE3C4A3C5C C343C503FE3C423C4A3C6B3 FE3C343C5E3C4C3C4C3FE3
MDPWON	KMENU	x	MENU 1.CLOCK SET 2.CHANNEL SET	MENU 1.CLOCK SET 2.CHANNEL SET	FE0118 0718003FE3C4A3C6B3FE3 FE3C463C503FE3C423C4A3C6 FE3C343C5E3C4C3C4C3FE3	FE0118 0718003FE3C4A3C6B3FE3 FE3C463C503FE3C423C4A3C6 FE3C343C5E3C4C3C4C3FE3

【図 14】



【図 15】



【図 16】

