(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4498167号
(P4498167)
(45)発行日　平成22年7月7日(2010.7.7)　(24)登録日　平成22年4月23日(2010.4.23)

(51)Int.Cl.　F I
G06F 17/50 (2006.01)　G06F 17/50 664Z

請求項の数 5　(全 15 頁)

(21)出願番号　特願2005-43143(P2005-43143)
(22)出願日　平成17年2月18日(2005.2.18)
(65)公開番号　特開2006-228065(P2006-228065A)
(43)公開日　平成18年8月31日(2006.8.31)
審査請求日　平成20年2月15日(2008.2.15)

(73)特許権者　000001007
キヤノン株式会社
東京都大田区下丸子３丁目３０番２号
(74)代理人　100076428
弁理士　大塚　康徳
(74)代理人　100112508
弁理士　高柳　司郎
(74)代理人　100115071
弁理士　大塚　康弘
(74)代理人　100116894
弁理士　木村　秀二
(72)発明者　清水　康世
東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内

最終頁に続く

(54)【発明の名称】プロパティ生成方法、検証方法及び検証装置

(57)【特許請求の範囲】
【請求項１】
論理システムを検証するためのプロパティを生成するプロパティ生成方法であって、
リスト生成手段が、論理システムが満たすべき仕様から対応する全事象のリストを生成するリスト生成工程と、
プロパティ生成手段が、前記全事象のリストに基づいて、前記仕様からユーザが予め定義したプロパティにおいて不足している事象を補うための補集合のプロパティを生成するプロパティ生成工程とを有し、
前記プロパティ生成工程は、前記全事象のリストから未定義の状態の事象を抽出し、抽出した未定義の状態の事象が成立しないことを表すプロパティを前記補集合のプロパティとして生成することを特徴とするプロパティ生成方法。
【請求項２】
論理システムが満たすべき仕様から作成されたプロパティを用いて前記論理システムを検証する検証方法であって、
プロパティ生成手段が、論理システムが満たすべき仕様に対応する全事象のリストに基づいて、前記仕様からユーザが予め定義したプロパティにおいて不足している事象を補うための補集合のプロパティを生成するプロパティ生成工程と、
検証実施手段が、前記補集合のプロパティを用いて前記論理システムの静的検証を実施する検証実施工程とを有し、
前記プロパティ生成工程は、前記全事象のリストから未定義の状態の事象を抽出し、抽

出した未定義の状態の事象が成立しないことを表すプロパティを前記補集合のプロパティとして生成することを特徴とする検証方法。

【請求項３】

論理システムが満たすべき仕様から作成されたプロパティを用いて前記論理システムを検証する検証装置であって、

論理システムが満たすべき仕様に対応する全事象のリストに基づいて、前記仕様からユーザが予め定義したプロパティにおいて不足している事象を補うための補集合のプロパティを生成するプロパティ生成手段と、

前記補集合のプロパティを用いて前記論理システムの静的検証を実施する検証実施手段とを有し、

前記プロパティ生成手段は、前記全事象のリストから未定義の状態の事象を抽出し、抽出した未定義の状態の事象が成立しないことを表すプロパティを前記補集合のプロパティとして生成することを特徴とする検証装置。

【請求項４】

コンピュータに、論理システムを検証するためのプロパティを生成するプロパティ生成手順を実行させるためのプログラムであって、

論理システムが満たすべき仕様から対応する全事象のリストを生成するリスト生成手順と、

前記全事象のリストに基づいて、前記仕様からユーザが予め定義したプロパティにおいて不足している事象を補うための補集合のプロパティを生成するプロパティ生成手順とをコンピュータに実行させ、

前記プロパティ生成手順は、前記全事象のリストから未定義の状態の事象を抽出し、抽出した未定義の状態の事象が成立しないことを表すプロパティを前記補集合のプロパティとして生成することを特徴とするプログラム。

【請求項５】

コンピュータに、論理システムが満たすべき仕様から作成されたプロパティを用いて前記論理システムを検証する検証手順を実行させるためのプログラムであって、

論理システムが満たすべき仕様に対応する全事象のリストに基づいて、前記仕様からユーザが予め定義したプロパティにおいて不足している事象を補うための補集合のプロパティを生成するプロパティ生成手順と、

前記補集合のプロパティを用いて前記論理システムの静的検証を実施する検証実施手順とをコンピュータに実行させ、

前記プロパティ生成手順は、前記全事象のリストから未定義の状態の事象を抽出し、抽出した未定義の状態の事象が成立しないことを表すプロパティを前記補集合のプロパティとして生成することを特徴とするプログラム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、論理システムを検証するためのプロパティを生成する技術及び論理システムを検証する技術に関する。

【背景技術】

【０００２】

近年、市場はより高機能な製品を要求している。これに伴い、ＬＳＩを含む論理システムの規模は年々増加している。そのため、論理システムの検証には膨大な時間が必要となってきている。また、市場には消費者が欲するときに製品を投入しなければならず、ＬＳＩをはじめとした論理システムの開発期間は短くなっている。そのため、論理システムの検証効率を向上させる必要性が生じている。

【０００３】

ＬＳＩを含む論理システムの検証には、従来から使用されている動的シミュレーション手法と併せて静的検証手法を用いるようになってきた。静的検証はプロパティチェックと

等価性チェックの二種類に分けられる。本発明において、静的検証はプロパティチェックを取り扱うこととする。このプロパティチェックは、プロパティと呼ばれる仕様を数式で表したものと、検証対象のＤＵＴ（Design Under Test）との一致性を数学的に判定することで検証を行う。そのため、プロパティチェックは、動的シミュレーションと比較して短時間で検証を終えることが可能である。例えば、特許文献１には、ＤＵＴを基に形式検証を使用して、動的シミュレーションのカバレージ率を向上させる検証方法が記載されている。

【特許文献１】特開平１１－８５８２８号公報

【発明の開示】

【発明が解決しようとする課題】

【０００４】

このような性質を備えた静的検証は、一般に網羅検証が可能であるといわれているが、実際はプロパティを記述した部分のみの検証しかできていない。また、網羅検証ができているかどうかの確認方法は人手によるレビューの実施以外に確立されていない。

【０００５】

このように、静的検証で網羅検証ができているかどうかを客観的に判定する方法がない。その結果、検証未実施の部分が存在していることに気づかずに、十分な精度を持たない製品を市場に投入してしまう恐れがある。

【０００６】

また、本来、動的シミュレーションで検証する機能と静的検証で検証する機能とを切り分けて検証工数の削減を図ることが可能であるのに、実際は従来の動的シミュレーションに加えて静的検証を補完目的で使用しているので、検証工数が増大する一方であるという問題も生じている。

【０００７】

本発明は上記課題を解決するためになされたもので、静的検証における網羅率を客観的に判断し、網羅検証を可能にすることを目的とする。

【課題を解決するための手段】

【０００８】

本発明は、論理システムを検証するためのプロパティを生成するプロパティ生成方法であって、リスト生成手段が、論理システムが満たすべき仕様から対応する全事象のリストを生成するリスト生成工程と、プロパティ生成手段が、前記全事象のリストに基づいて、前記仕様からユーザが予め定義したプロパティにおいて不足している事象を補うための補集合のプロパティを生成するプロパティ生成工程とを有し、前記プロパティ生成工程は、前記全事象のリストから未定義の状態の事象を抽出し、抽出した未定義の状態の事象が成立しないことを表すプロパティを前記補集合のプロパティとして生成することを特徴とする。

【０００９】

また、本発明は、論理システムが満たすべき仕様から作成されたプロパティを用いて前記論理システムを検証する検証方法であって、プロパティ生成手段が、論理システムが満たすべき仕様に対応する全事象のリストに基づいて、前記仕様からユーザが予め定義したプロパティにおいて不足している事象を補うための補集合のプロパティを生成するプロパティ生成工程と、検証実施手段が、前記補集合のプロパティを用いて前記論理システムの静的検証を実施する検証実施工程とを有し、前記プロパティ生成工程は、前記全事象のリストから未定義の状態の事象を抽出し、抽出した未定義の状態の事象が成立しないことを表すプロパティを前記補集合のプロパティとして生成することを特徴とする。

【００１０】

更に、本発明は、論理システムが満たすべき仕様から作成されたプロパティを用いて前記論理システムを検証する検証装置であって、論理システムが満たすべき仕様に対応する全事象のリストに基づいて、前記仕様からユーザが予め定義したプロパティにおいて不足している事象を補うための補集合のプロパティを生成するプロパティ生成手段と、前記補

集合のプロパティを用いて前記論理システムの静的検証を実施する検証実施手段とを有し、前記プロパティ生成手段は、前記全事象のリストから未定義の状態の事象を抽出し、抽出した未定義の状態の事象が成立しないことを表すプロパティを前記補集合のプロパティとして生成することを特徴とする。

【発明の効果】

【００１１】

本発明によれば、静的検証において困難であったプロパティの網羅率の判断が客観的にできるようになり網羅検証が可能となる。これにより、十分な精度を持った製品を市場に投入することが可能になる。

【００１２】

また、プロパティの信頼性を客観的に判断することが可能となるため、従来実施していた人間のレビューによるプロパティのチェックが不要となり、検証工数の削減及びプロパティの品質を向上させることが可能となる。また、静的検証の網羅検証が可能になることで、静的検証と動的シミュレーションによる検証とを切り分けることができ、検証工数の削減が可能となる。従って、開発期間の短縮ができ、市場が欲するときに製品を投入することが可能となる。

【発明を実施するための最良の形態】

【００１３】

以下、図面を参照しながら発明を実施するための最良の形態について詳細に説明する。

【００１４】

図１は、本実施形態における静的検証に用いる不成立動作プロパティの生成手順を示す図である。図１において、１０１は仕様であり、論理システムの仕様が記載されたデータである。ここで、仕様１０１は書式に依存したものではなく、論理システムの入力信号と内部の信号及びそれらの取り得る値が抽出可能なように記載されていれば良く、自然言語、その他、如何なる言語や書式で記述されていてもかまわない。

【００１５】

また、本実施形態では、如何なる場合も仕様１０１が正しいものとし、後述するユーザ定義プロパティ、全事象のリスト及び検証対象のＤＵＴ（Design Under Test）等は全て仕様１０１のデータに基づいて作成されるものとする。

【００１６】

１０２はユーザ定義プロパティであり、仕様１０１に基づいてユーザが記述したもので、その記述内容に間違いはないものとする。但し、検証未実施の部分が存在している可能性は残っているものとする。また、このユーザ定義プロパティ１０２は、テストアイテムチェックリスト等から人手で記述されても自動で記述されても良い。

【００１７】

１１０は事象リスト生成モジュールであり、仕様１０１に基づいて論理システムの入力信号及び内部の信号が取り得る値を抽出し、仕様１０１が取り得る全事象のリストを生成する機能を有する。この事象リスト生成モジュール１１０の機能については図２を用いて更に後述する。

【００１８】

尚、本実施形態では、事象のリストを生成するアルゴリズムに依存するものではなく、如何なるアルゴリズムを用いてもかまわない。例えば、仕様１０１から必要な信号が取り得る値を全て自動で抽出しても、自動で抽出した後に不足の状態をユーザが付け加えてもかまわない。

【００１９】

１０３は全事象のリストであり、上述の仕様１０１に基づいて生成され、仕様１０１が取り得る全事象が記載される。ここで、全事象のリスト１０３は後述する抽出モジュールが解釈できる書式であれば良く、どのような書式であってもかまわない。

【００２０】

１１１は抽出モジュールであり、ユーザ定義プロパティ１０２と全事象のリスト１０３

とからユーザ未定義状態の事象を抽出する。この抽出モジュール１１１の機能については図３及び図４を用いて更に後述する。

【００２１】

１０４はユーザ未定義状態の事象であり、全事象のリスト１０３に記載された事象中、ユーザ定義プロパティ１０２に記載されていない事象で、ユーザ定義プロパティ１０２の不足を補うための補集合である。

【００２２】

１１２は不成立動作プロパティ変換モジュールであり、ユーザ未定義状態の事象１０４を、常に成立しないとする不成立動作プロパティに変換するものである。この不成立動作変換モジュールの機能とこれが必要である理由は更に後述する。

【００２３】

１０５は不成立動作プロパティであり、不成立動作プロパティ変換モジュール１１２により変換されたプロパティである。この不成立動作プロパティ１０５を使用して静的検証等を実施することにより、ユーザ定義プロパティ１０２に不足のプロパティを追加することが可能となる。

【００２４】

１００は不成立動作プロパティ生成モジュールであり、上述したように、仕様１０１とユーザ定義プロパティ１０２とに基づいて不成立動作プロパティ１０５を生成する。

【００２５】

次に、上述した事象リスト生成モジュール１１０において、仕様１０１に基づいて事象リストを生成する方法について説明する。

【００２６】

図２は、本実施形態における事象リスト生成モジュール１１０が生成する全事象リストの一例を示す図である。図２に示す仕様書２０１は、上述した仕様１０１の一例を示すもので、ここでは、信号名、その種別及びそれらが取り得る値が表形式で記載されている。本発明は、仕様の書式に関するものではないので、表形式に限ったものではない。

【００２７】

全事象リスト２０２は、仕様書２０１に基づいて抽出されたもので、この例では仕様書２０１の入力信号及び内部の信号と、それらが取り得る値の範囲が記載されている。

【００２８】

尚、全事象リスト２０２は、仕様書２０１から自動的に抽出されても良いし、ユーザが手動で抽出しても良い。本発明は、抽出方法に関するものではないので、仕様１０１から全ての状態の事象が抽出できれば、如何なる方法を用いてもかまわない。

【００２９】

次に、上述した抽出モジュール１１１において、全事象のリスト１０３とユーザ定義プロパティ１０２とに基づいてユーザ未定義状態の事象１０４を抽出する方法について説明する。

【００３０】

図３は、本実施形態における抽出モジュール１１１の処理を示すフローチャートである。まず、ステップＳ３０１において、ユーザ定義プロパティ１０２からプロパティを一つずつ取り出す。次に、ステップＳ３０２において、ステップＳ３０１で取り出したプロパティと全事象のリスト１０３とを比較し、ステップＳ３０３において、結果が一致したか否かを判定する。その結果、一致したならばステップＳ３０５へ進むが、一致しなければステップＳ３０４へ進み、一致しなかったものについて差分を抽出し、不足状態リストに追加する。そして、ステップＳ３０５において、全事象のリスト１０３内で一致したもの及び差分を抽出したものに印を付ける。

【００３１】

次に、ステップＳ３０６において、ユーザ定義プロパティ１０２に残りのプロパティがあるか否かを判定する。ここで、残りがあればステップＳ３０１に戻り、上述の比較処理を繰り返し、残りがなくなるとステップＳ３０７へ進む。

【００３２】
このステップＳ３０７では、全事象のリスト１０３の中に未検査の信号があるかを否かを判定する。ここで、未検査の信号がなければ、この処理を終了するが、未検査の信号があればステップＳ３０８へ進み、未検査信号の状態をユーザ未定義状態１０４のリストに追加する。
【００３３】
ここで、図４を用いて上述した未検査信号の状態をユーザ未定義状態に追加する具体的な処理について説明する。
【００３４】
図４は、ユーザ定義プロパティ、ユーザ未定義状態及び比較後の全事象リストの一例を示す図である。図４において、４００は図２に示す仕様書２０１から生成したユーザ定義プロパティである。この例では、４つのプロパティ４０１～４０４が記載されているものとする。４２０はユーザ未定義状態のリストであり、ユーザ定義プロパティ４００に記載されていない未定義状態の事象が追加される。４３０は全事象のリストである。
【００３５】
上述したように、ステップＳ３０１で、ユーザ定義プロパティ４００から取り出されるプロパティは仕様書２０１からユーザによって生成されたものであり、記載内容に間違いはないが、不足がある恐れはある。本実施形態では、ユーザ定義プロパティ４００は信号の範囲及び表で記述されているが、本発明はユーザ定義プロパティの書式に関するものではないので、この書式に限定されるものではない。
【００３６】
次に、ステップＳ３０２で、ステップＳ３０１で取り出したプロパティが全事象リスト２０２に一致するものがあるか否かを比較する。例えば、ユーザ定義プロパティ４００から信号Ａについて記述されているプロパティ４０１を取り出した場合には、全事象リスト２０２の信号Ａの部分と比較する。
【００３７】
ステップＳ３０３では、ステップＳ３０２の比較が一致したかどうか、つまり、ユーザ定義プロパティ４００の内容が全事象リスト２０２の内容を満たしているかを判定する。一致しているならば、ユーザ定義プロパティ４００は全事象リスト２０２を網羅しているので、プロパティの不足はない。一方、一致しないならば、ユーザ定義プロパティ４００と全事象リスト２０２の間に差が生じており、ユーザ定義プロパティ４００に不足が存在することになる。
【００３８】
また、ユーザ定義プロパティ４００は仕様書２０１に基づいて記述されており、仕様書２０１の範囲を超えたプロパティが記述されていることはない。従って、ユーザ定義プロパティ４００に存在し、全事象リスト２０２に存在しない信号はないものとして良い。
【００３９】
上述のステップＳ３０３で不一致と判定された場合、ステップＳ３０４で、ユーザ定義プロパティ４００と全事象リスト２０２との差分を抽出する。例えば、プロパティ４０１と全事象リスト２０２とを比較する。ここで、信号Ａについての記述に注目すると、プロパティ４０１ではＡが取り得る範囲を０から１００までと記述しているが、全事象リスト２０２では、Ａは０から２の３２乗－１の範囲を取り得る。つまり、この差分を抽出すると、Ａが１００より大きく２の３２乗－１まではユーザ定義プロパティ４００には未記述であることが分かる。よって、この結果をユーザ未定義状態４２０へ追加する。また同様にして、プロパティ４０３についてもユーザ未定義状態４２０に信号ａが０の状態を抽出、追加する。
【００４０】
ステップＳ３０５では、全事象リスト２０２の中で判定し終わった信号に印を付ける。図４に示す例では、ユーザ定義プロパティ４００が全事象リスト２０２に記載の信号Ａ、信号Ｂ、信号ａ及び信号ｂを全て網羅しているため、全事象リスト４３０のように全てに

印を付ける。

【００４１】

尚、ユーザ定義プロパティ４００に記述されている信号数が全事象リスト２０２の信号数を満たしていない場合には、全事象リスト４３０の中には印が付かない箇所が存在することになり、印の付いていない信号については未抽出の信号であることがわかる。即ち、未抽出の信号を知らせるために、印を付けていくものである。

【００４２】

ステップＳ３０６では、ユーザ定義プロパティ４００に残りがないか否かを判定する。残りがない場合はそのままステップＳ３０７へ進む。また、残りがある場合は残りがなくなるまで、ステップＳ３０１からステップＳ３０６までの処理を繰り返す。

【００４３】

ステップＳ３０７では、全事象リスト４３０に未検証の信号があるか否かを判定する。つまり、印の付いていない信号がないか否かを判定する。例えば、ユーザ定義プロパティ４００と全事象リスト２０２とを比較した結果、全事象リスト２０２にあってユーザ定義プロパティ４００にない信号名はないため、全事象リスト４３０のように全て印が付く。これは、全事象リスト２０２に未検査の信号が存在しないことになる。この場合、全ての信号についての検査が終了したと判定し、全ての処理を終了して良い。

【００４４】

一方、全事象リスト４３０に印の付いていない信号があった場合には、未検査の信号が存在するため、次の処理に進む。

【００４５】

ステップＳ３０８では、ステップＳ３０７で未検査の信号があると判定された場合に、全事象リスト２０２に記載されている状態をそのまま未検査信号の事象として、ユーザ未定義状態リスト４２０に追加する。このようにすることにより、仕様１０１の全ての事象についてユーザ未定義状態リスト４２０に追加することができる。

【００４６】

ここでは、全事象リストにおけるユーザ定義プロパティの不足を補う補集合を、全事象リストとユーザ定義プロパティの比較によって求めたが、補集合を求める方法は本発明に依存するものではなく、如何なる方法でもかまわない。

【００４７】

また、図３及び図４に示す全事象リスト２０２、ユーザ定義プロパティ４００、ユーザ未定義状態４２０、ここで説明した比較処理、抽出処理、印の付けかた等は本発明に依存するものではなく、如何なる方法でもかまわない。

【００４８】

次に、上述の不成立動作プロパティ変換モジュール１１２において、ユーザ未定義状態１０４を不成立動作プロパティ１０５に変換する処理について説明する。

【００４９】

図５は、不成立動作プロパティ変換モジュール１１２の処理を示すフローチャートである。まず、ステップＳ５０１において、ユーザ未定義状態１０４から状態を一つずつ取り出し、ステップＳ５０２において、ステップＳ５０１で取り出した状態を不成立動作プロパティに変換する。次に、ステップＳ５０３において、ユーザ未定義状態１０４に残りがあるか否かを判定し、ユーザ未定義状態１０４に残りがあればステップＳ５０１に戻り、上述の処理を繰り返す。また、残りがなければ、この処理を終了する。

【００５０】

ここで、本実施形態における不成立動作プロパティ１０５を生成する理由について説明する。ユーザ定義プロパティ１０２を用いてＤＵＴを検証する場合に、仕様１０１に記載されている内容の全てがユーザ定義プロパティ１０２として表現されていなければ、網羅検証が可能とは言えない。

【００５１】

そこで、ユーザ定義プロパティ１０２は全ての仕様１０１を表現しており、記述されて

いない状態は仕様１０１で取り得る範囲ではないと仮定して、「ユーザ定義プロパティに記述されていない状態は起こりえない」というプロパティを生成する。これが不成立動作プロパティ１０５である。このプロパティを生成することにより、不成立動作プロパティ１０５を用いて検証を実施した場合に「起こりえないはずの状態が起こり得る」となれば、それはユーザ定義プロパティ１０２に不足していたプロパティであることがわかる。

【００５２】

例えば、ツールとしてこのような動作を組み込めば、「記述されていないプロパティの状態がＤＵＴ内で起こっているが、プロパティは正しいか？」というようなエラー表示が吐き出され、プロパティの不足又はＤＵＴのバグが発見されることになる。

【００５３】

ここで、図４に示すユーザ未定義状態４２０及び図６を用いて、不成立動作プロパティ１０５を生成する方法について説明する。ここでは、不成立動作プロパティ１０５の生成方法として、「ユーザ未定義状態が常に成り立たない」というプロパティを生成する一例を示すが、不成立動作を生成するならば、この方法に限られるものではない。

【００５４】

図６は、ユーザ未定義状態４２０が常に成立しない不成立動作プロパティの一例を示す図である。図６において、６００はユーザ未定義状態４２０が成立しないことを証明するためのプロパティである。ここで、プロパティ６０１、プロパティ６０２は、共にユーザ未定義状態４２０に「forever」及び「！」が付けられている。それぞれの意味は以下の通りである。

【００５５】

「forever」==「永遠に」

「！」　　==「否定（起こらない）」

即ち、プロパティ６０１は、「信号Ａが１００より大きく、２の３２乗－１より小さい値をとることは永遠に起こりえない。」という意味を示し、プロパティ６０２は「信号ａは永遠に０となることはありえない。」という意味を示している。

【００５６】

これにより、「ユーザ未定義状態に記述されている状態は永遠に起こることはない」というプロパティが追加される。静的検証を実施する際に、ユーザ定義プロパティ１０２に不成立動作プロパティ１０５を使用することにより、仕様１０１の取り得る全事象を検証することが可能となる。

【００５７】

ここで、追加した不成立動作プロパティ１０５を用いて静的検証を実施し、ユーザ定義プロパティ１０２の不備を指摘する例について説明する。

【００５８】

図７は、不成立動作プロパティを用いて静的検証を実施し、追加プロパティを生成する手順を示す図である。ここで、図７に示す仕様７０１及びユーザ定義プロパティ７０２は図１に示す仕様１０１及びユーザ定義プロパティ１０２に相当し、不成立動作プロパティ生成モジュール７２０及び不成立動作プロパティ７０５は、図１に示す不成立動作プロパティ生成モジュール１００及び不成立動作プロパティ１０５に相当するものである。

【００５９】

図７に示すＤＵＴ７０６は、仕様７０１に基づいて生成されたVerilog或いはＶＨＤＬ（VHSIC Hardware Description Language）などのハードウェア記述言語により記述された論理システムの設計データである。これは仕様７０１から、自動で生成されても、人手で作成されてもかまわない。

【００６０】

静的検証実施モジュール７２１は、不成立動作プロパティ７０５とＤＵＴ７０６を用いて静的検証を実施するものである。尚、この静的検証実施モジュール７２１は、本発明に依存しないため、如何なるツールを用いても、またその他の如何なる方法で実施されてもかまわない。

【００６１】
この静的検証実施モジュール７２１で検証した結果、フェイルしたプロパティ７０７及びパスしたプロパティ７０８のログを取る。ここで、フェイルしたプロパティ７０７には、不成立動作プロパティ７０５の中で静的検証がフェイルしたプロパティが記録される。そして、不成立動作プロパティ７０５がフェイルした状態とは、図１に示したユーザ未定義状態がＤＵＴ７０６内で存在することを意味する。即ち、ユーザが記述していない状態にも関わらず、ＤＵＴ７０６でその状態が起こり得るため、ユーザ定義プロパティ７０２の不備、若しくはＤＵＴ７０６のバグである可能性を含む。
【００６２】
一方、パスしたプロパティ７０８には、不成立動作プロパティ７０５の中で静的検証がパスしたプロパティが記録される。そして、不成立動作プロパティ７０５がパスする状態とは、図１に示したユーザ未定義状態がＤＵＴ内には存在しないことを意味する。即ち、元々仕様に記載されていたものではなく、そのような状態は本来あり得ない状態である。そのため、パスしたプロパティ７０８に記録されたプロパティは、ユーザ定義プロパティ７０２の不備ではない。しかし、その状態は起こり得ないという確認のために不成立動作プロパティの形のまま追加プロパティ７０９に記録する。
【００６３】
次に、比較モジュール７１０は、フェイルしたプロパティ７０７と仕様７０１とを比較し、ユーザ定義プロパティ７０２の不備のものなのか、或いはＤＵＴ７０６のバグによりフェイルしたものなのかを判定する。この比較の方法は、本発明に依存するものではないので、如何なる方法で実施してもかまわない。
【００６４】
プロパティ変換モジュール７１１は、フェイルしたプロパティ７０７の中でユーザ定義プロパティ７０２の不備だと分かったプロパティを、不成立動作から成立するプロパティに変換し、追加プロパティ７０９へ記録する。これは、不成立動作プロパティ７０５の、「ユーザ未定義の状態は起こり得ない」という形のままではプロパティとＤＵＴ７０６が矛盾するので、「ユーザ未定義の状態は起こる」というプロパティに変換する必要があるためである。
【００６５】
デバッグ７２２は、比較モジュール７１０により仕様７０１とフェイルしたプロパティ７０７とを比較した結果、ＤＵＴ７０６のバグであると判定したプロパティに対して行う。ここでバグとは、仕様７０１とＤＵＴ７０６とが矛盾を起こしていることをいう。このデバッグの方法は、本発明に依存するものではないので、如何なる方法で実施してもかまわない。
【００６６】
図８は、静的検証を実施してユーザ定義プロパティの不備を追加する処理を示すフローチャートである。
【００６７】
まず、ステップＳ８０１において、不成立動作プロパティ７０５からプロパティを一つずつ取り出し、ステップＳ８０２において、ステップＳ８０１で取り出したプロパティとＤＵＴ７０６とを用いて静的検証を実施する。この静的検証の実施方法については本発明に関するものではないので、言及しないこととする。
【００６８】
次に、ステップＳ８０３において、ステップＳ８０２で実施した静的検証の結果を判定する。ここで、静的検証がパスした場合はステップＳ８０７へ進み、フェイルした場合はステップＳ８０４へ進む。
【００６９】
このステップＳ８０４では、不成立動作プロパティ７０５を成立動作プロパティに変換する。図６を用いて変換の一例を示す。プロパティ６０１が、ステップＳ８０３において「静的検証がフェイルした」と報告されたならば、

「forever(!(100 ＜ A ≦ 2^{32}-1))」
は成立しない、つまり、
「100 ＜ A ≦ 2^{32}-1」…Ｐ１
がＤＵＴ７０６内で起こってしまっていることになる。
【００７０】
よって、追加プロパティ７０９へ上記プロパティＰ１を記録するべきである。そのため、一度不成立動作に変換したプロパティを成立する形に変換する。ここでは、変換の方法を唯単純に「forever」と「！」を取り外すこととしたが、変換の方法は本発明に依存しないので、如何なる方法で変換されてもかまわない。
【００７１】
次に、ステップＳ８０５において、ステップＳ８０４で変換したプロパティＰ１を仕様７０１と比較する。なぜならば、ステップＳ８０３において静的検証がフェイルする原因としては単なるプロパティの漏れだけではなく、ＤＵＴ７０６のバグにより本来起こってはいけない現象が起こっている可能性があるためである。ＤＵＴ７０６のバグを見逃さないためにも、ここで一度仕様７０１と比較する必要がある。比較の方法は、如何なる方法でもかまわない。
【００７２】
そして、ステップＳ８０６において、ステップＳ８０５で比較した結果、仕様７０１とプロパティが一致したか否かを判定する。一致したならばステップＳ８０７へ進み、一致しないならばステップＳ８０９へ進む。
【００７３】
このステップＳ８０７では、ステップＳ８０３で静的検証の結果パスしたプロパティであれば、不成立動作のままのプロパティを、ステップＳ８０６でプロパティが仕様７０１と一致していたならば、ステップＳ８０４で変換したプロパティを、ステップＳ８０６でＤＵＴ７０６のバグが発覚し、ＤＵＴ７０６を直した後ならばステップＳ８０４で変換したプロパティを追加プロパティ７０９として記録すればよい。
【００７４】
例えば、ステップＳ８０３でパスしたならばプロパティＰ１を、それ以外の場合はプロパティ６０１を追加する。
【００７５】
次に、ステップＳ８０８において、不成立動作プロパティ７０５に残りがあるか否かを判定する。残りがあった場合はステップＳ８０１へ、残りがない場合はこの処理を終了とする。
【００７６】
また、ステップＳ８０９では、ステップＳ８０５で比較した結果、ステップＳ８０４で生成したプロパティが仕様７０１と一致しなかったことが分かったので、仕様７０１とプロパティの情報に基づいてＤＵＴ７０６のバグを取り除く。
【００７７】
次に、ステップＳ８１０において、ステップＳ８０８で正しくバグを取り除いたことを確認するために、デバッグを実施したＤＵＴ７０６とステップＳ８０４で生成したプロパティを使用して静的検証を実施する。このとき、ステップＳ８０５において、プロパティと仕様７０１とは既に比較しているため、ここで静的検証に用いられるプロパティは必ず正しいとしてよい。
【００７８】
そして、ステップＳ８１１において、ステップＳ８１０の結果を判定する。ステップＳ８１０の静的検証の結果がパスしたならばステップＳ８０７へ進み、ここで使用したプロパティを追加プロパティ７０９として記録する。フェイルしたならばステップＳ８０９に戻り、同プロパティを使用してＤＵＴ７０６のバグがなくなり静的検証がパスするまで、ステップＳ８０９からステップＳ８１１までの処理を繰り返す。
【００７９】

以上説明したように、本実施形態によれば、全事象リスト及び不成立動作プロパティを生成することにより、静的検証における網羅率を客観的に判断することが可能となり、従来困難であった静的検証の網羅検証が可能となる。

【００８０】

また、静的検証の網羅性を客観的に判断できるので、従来レビューにかかっていた時間が不要となり、検証工数の削減を図ることが可能となる。加えて、静的検証の網羅検証が可能になることにより、静的検証と動的シミュレーションの検証すべき機能を切り分けることが可能となり、検証工数の削減を図ることが可能になる。

【００８１】

尚、上述した各モジュールは、一般的なコンピュータによって実施されても良く、また専用の機器によって実施されても良い。

【００８２】

本発明は複数の機器（例えば、ホストコンピュータ，インターフェース機器，リーダ，プリンタなど）から構成されるシステムに適用しても、１つの機器からなる装置（例えば、複写機，ファクシミリ装置など）に適用しても良い。

【００８３】

また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記録媒体を、システム或いは装置に供給し、そのシステム或いは装置のコンピュータ（ＣＰＵ若しくはＭＰＵ）が記録媒体に格納されたプログラムコードを読出し実行することによっても、達成されることは言うまでもない。

【００８４】

この場合、記録媒体から読出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記録媒体は本発明を構成することになる。

【００８５】

このプログラムコードを供給するための記録媒体としては、例えばフロッピー（登録商標）ディスク，ハードディスク，光ディスク，光磁気ディスク，ＣＤ－ＲＯＭ，ＣＤ－Ｒ，磁気テープ，不揮発性のメモリカード，ＲＯＭなどを用いることができる。

【００８６】

また、コンピュータが読出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているＯＳ（オペレーティングシステム）などが実際の処理の一部又は全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００８７】

更に、記録媒体から読出されたプログラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるＣＰＵなどが実際の処理の一部又は全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【図面の簡単な説明】

【００８８】

【図１】本実施形態における静的検証に用いる不成立動作プロパティの生成手順を示す図である。

【図２】本実施形態における事象リスト生成モジュール１１０が生成する全事象リストの一例を示す図である。

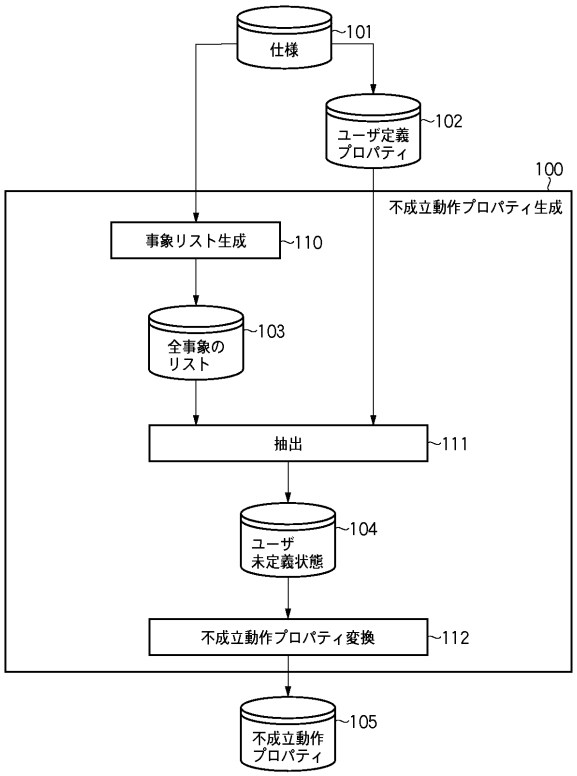
【図３】本実施形態における抽出モジュール１１１の処理を示すフローチャートである。

【図４】ユーザ定義プロパティ、ユーザ未定義状態及び比較後の全事象リストの一例を示す図である。

【図５】不成立動作プロパティ変換モジュール１１２の処理を示すフローチャートである

。
【図６】ユーザ未定義状態４２０が常に成立しない不成立動作プロパティの一例を示す図である。
【図７】不成立動作プロパティを用いて静的検証を実施し、追加プロパティを生成する手順を示す図である。
【図８】静的検証を実施してユーザ定義プロパティの不備を追加する処理を示すフローチャートである。
【符号の説明】
【００８９】
１００　不成立動作プロパティ生成モジュール
１０１　仕様
１０２　ユーザ定義プロパティ
１０３　全事象のリスト
１０４　ユーザ未定義状態
１０５　不成立動作プロパティ
１１０　事象リスト生成モジュール
１１１　抽出モジュール
１１２　不成立動作プロパティ変換モジュール
２０１　仕様書
２０２　全事象リスト
４００　ユーザ定義プロパティ
４２０　ユーザ未定義状態
４３０　全事象リスト
６００　不成立動作プロパティ
７０１　仕様
７０２　ユーザ定義プロパティ
７０５　不成立動作プロパティ
７０６　ＤＵＴ
７０７　フェイルしたプロパティ
７０８　パスしたプロパティ
７０９　追加プロパティ
７１０　比較モジュール
７１１　プロパティ変換モジュール
７２０　不成立動作プロパティ生成モジュール
７２１　静的検証実施モジュール
７２２　デバッグ

【図１】



【図２】

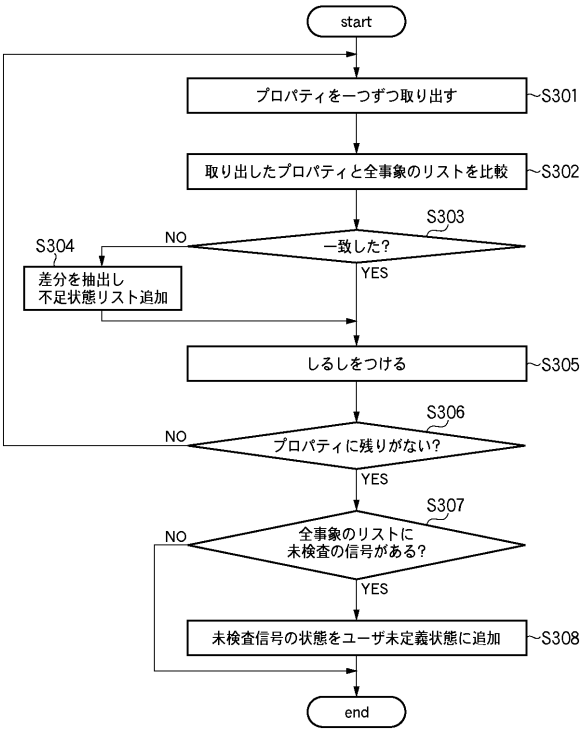
Signal name	type	value
A[31:0]	input	$0 \leqq A \leqq 2^{32}-1$
B	input	$0 \leqq B \leqq 1$
a	signal	$0 \leqq a \leqq 3$
b	signal	$0 \leqq b \leqq 1$
C	output	・
D	output	・
E	output	・

201

Signal name	value
A[31:0]	$0 \leqq A \leqq 2^{32}-1$
B	$0 \leqq B \leqq 1$
a	$0 \leqq a \leqq 3$
b	$0 \leqq b \leqq 1$

202

【図３】



【図４】

400

$0 \leqq A \leqq 100$
$0 \leqq B \leqq 1$
$0 < a \leqq 3$
$0 \leqq b \leqq 1$

401 402 403 404

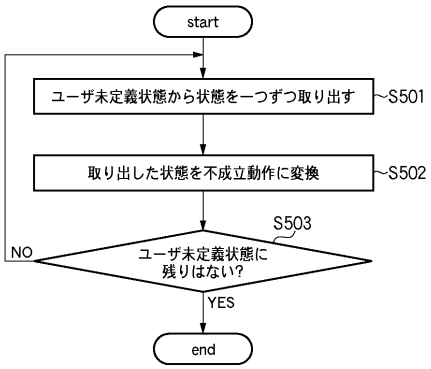
$100 < A \leqq 2^{32}-1$
$a == 0$

420

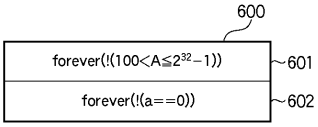
Signal name	value
A[31:0]	$0 \leqq A \leqq 2^{32}-1$
B	$0 \leqq B \leqq 1$
a	$0 \leqq a \leqq 3$
b	$0 \leqq b \leqq 1$

430

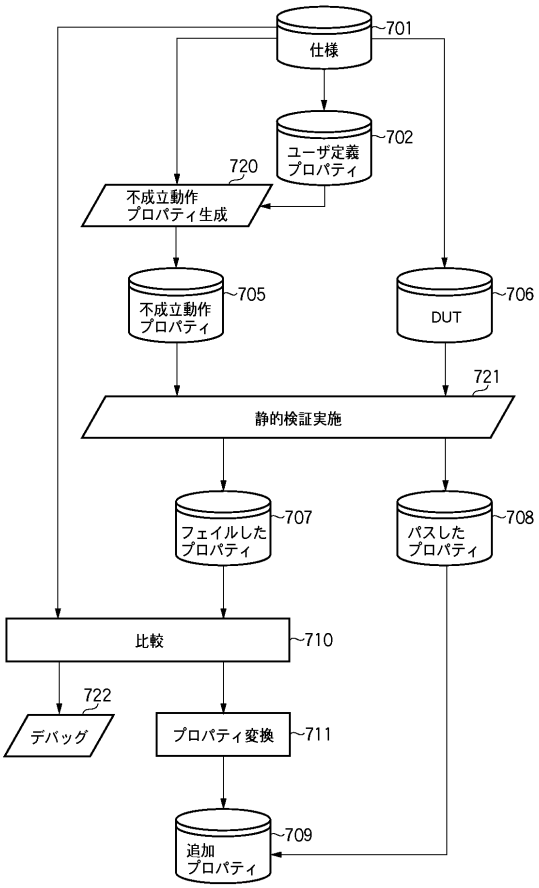
【図５】


start
ユーザ未定義状態から状態を一つずつ取り出す
S501
取り出した状態を不成立動作に変換
S502
S503
ユーザ未定義状態に
残りはない?
NO
YES
end

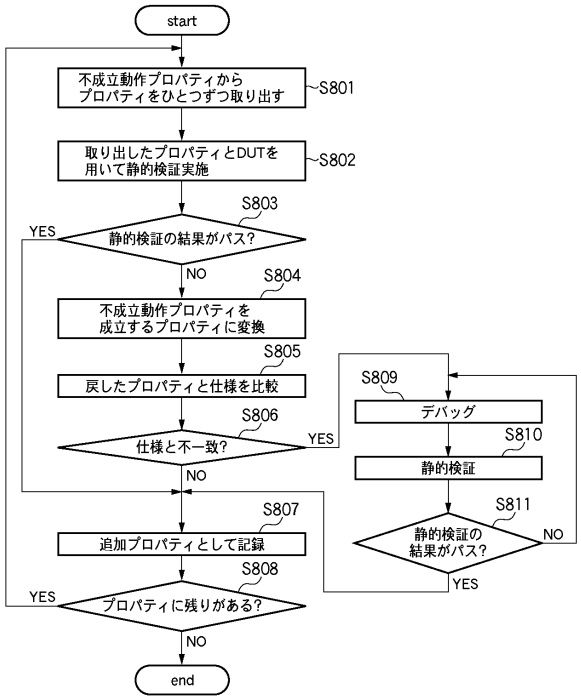
【図６】


600
forever(!(100<A≦2^32−1))
601
forever(!(a==0))
602

【図７】


701
仕様
702
ユーザ定義
プロパティ
720
不成立動作
プロパティ生成
705
不成立動作
プロパティ
706
DUT
721
静的検証実施
707
フェイルした
プロパティ
708
パスした
プロパティ
比較
710
722
デバッグ
プロパティ変換
711
709
追加
プロパティ

【図８】


start
不成立動作プロパティから
プロパティをひとつずつ取り出す
S801
取り出したプロパティとDUTを
用いて静的検証実施
S802
S803
静的検証の結果がパス?
YES
NO
S804
不成立動作プロパティを
成立するプロパティに変換
S805
戻したプロパティと仕様を比較
S806
仕様と不一致?
YES
NO
S807
追加プロパティとして記録
S808
プロパティに残りがある?
YES
NO
end
S809
デバッグ
S810
静的検証
S811
静的検証の
結果がパス?
NO
YES

フロントページの続き

審査官　平野　崇

(56)参考文献　特開２００１－３１８９５９（ＪＰ，Ａ）
特開平１１－０８５８２８（ＪＰ，Ａ）
特開２０００－１８１９３９（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１７／５０
ＣｉＮｉｉ
ＪＳＴＰｌｕｓ（ＪＤｒｅａｍＩＩ）