

(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2014-106928

(P2014-106928A)

(43) 公開日 平成26年6月9日(2014.6.9)

(51) Int.Cl.
G06F 11/22 (2006.01)F I
G06F 11/22 310Aテーマコード (参考)
5B048

審査請求 未請求 請求項の数 6 O L (全 20 頁)

(21) 出願番号 特願2012-261729 (P2012-261729)
(22) 出願日 平成24年11月29日 (2012.11.29)(71) 出願人 000005223
富士通株式会社
神奈川県川崎市中原区上小田中4丁目1番
1号
(74) 代理人 100074099
弁理士 大菅 義之
(74) 代理人 100133570
弁理士 ▲徳▼永 民雄
(72) 発明者 西牧 静香
神奈川県川崎市中原区上小田中4丁目1番
1号 株式会社富士通コンピュータテクノ
ロジーズ内

最終頁に続く

(54) 【発明の名称】 試験方法、試験プログラム、および、試験制御装置

(57) 【要約】

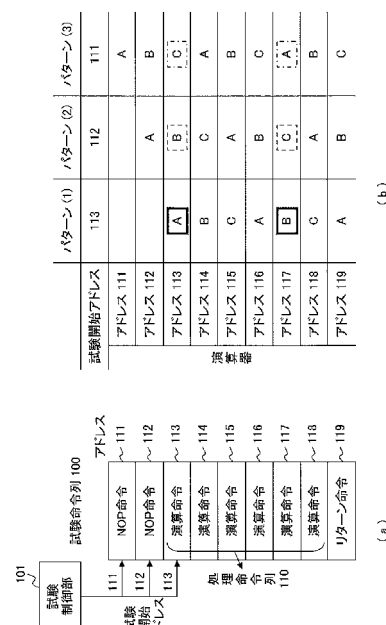
【課題】 複数の演算器の間を接続する回路の故障の検出を効率的に行う。

【解決手段】

実施形態にかかる方法では、第1～第3の演算器を有する装置の試験を行うことができる。実施形態にかかる方法では、第1の演算器と第2の演算器を接続する第1の回路を使用する処理を含む命令列である処理命令列を用いて、第2の演算器と第3の演算器を接続する第2の回路を使用する処理を含む試験命令列を生成する。ここで、試験命令列は、処理命令列の先頭に、処理命令列に含まれる命令の演算結果に変化を与えない命令列である追加命令列を追加したものである。処理命令列の先頭から試験命令列を装置に実行させることにより、前記第1の回路の故障の有無を判定する。また、追加命令列から試験命令列を装置に実行させることにより、第2の回路の故障の有無を検出する。

【選択図】 図1

試験方法の例



【特許請求の範囲】

【請求項 1】

第 1 ～ 第 3 の演算器を有する装置の試験方法であって

前記第 1 の演算器と前記第 2 の演算器を接続する第 1 の回路を使用する処理を含む命令列である処理命令列の先頭に、前記処理命令列に含まれる命令の演算結果に変化を与えない命令列である追加命令列を追加することにより、前記第 2 の演算器と前記第 3 の演算器を接続する第 2 の回路を使用する処理を含む試験命令列を生成し、

前記処理命令列の先頭から前記試験命令列を前記装置に実行させることにより、前記第 1 の回路の故障の有無を判定し、

前記追加命令列から前記試験命令列を前記装置に実行させることにより、前記第 2 の回路の故障の有無を判定する

処理を前記装置に行わせることを特徴とする試験方法。

【請求項 2】

前記装置は、

前記処理命令列の実行によって得られると予測される予測値を記憶し、

前記処理命令列の先頭から前記試験命令列を実行することにより得られる第 1 の演算結果が前記予測値と異なると、前記第 1 の回路に故障があると判定し、

前記追加命令列の先頭から前記試験命令列を実行することにより得られる第 2 の演算結果が前記予測値と異なると、前記第 2 の回路に故障があると判定する

ことを特徴とする請求項 1 に記載の試験方法。

【請求項 3】

前記追加命令列は、1 つ以上の命令を含み、

前記装置は、

前記試験命令列の実行を開始するアドレスである開始アドレスの初期値を、前記処理命令列の先頭に割り当てられたアドレスの番地に設定し、

前記試験命令列を実行すると、次に前記試験命令列を実行するときの開始アドレスを、前記試験命令列の実行のときに用いた開始アドレスに割り当てられた前記追加命令列中の 1 つの命令の格納に使用されるメモリサイズを減算して得られる番地に変更し、

前記試験命令列を実行した回数が、前記追加命令列に含まれている命令の数以下の所定の閾値に達すると処理を終了する

ことを特徴とする請求項 1 または 2 に記載の試験方法。

【請求項 4】

前記追加命令列は、1 つ以上の命令を含み、

前記装置は、

前記試験命令列の実行を開始するアドレスである開始アドレスの初期値を、前記試験命令列の先頭に割り当てられたアドレスの番地に設定し、

前記試験命令列を実行すると、次に前記試験命令列を実行するときの開始アドレスを、前記試験命令列の実行のときに用いた開始アドレスに割り当てられた前記追加命令列中の 1 つの命令の格納に使用されるメモリサイズを加算して得られる番地に変更し、

前記試験命令列を実行した回数が、前記追加命令列に含まれている命令の数以下の所定の閾値に達すると処理を終了する

ことを特徴とする請求項 1 または 2 に記載の試験方法。

【請求項 5】

第 1 ～ 第 3 の演算器を有する装置の試験プログラムであって

前記第 1 の演算器と前記第 2 の演算器を接続する第 1 の回路を使用する処理を含む命令列である処理命令列の先頭に、前記処理命令列に含まれる命令の演算結果に変化を与えない命令列である追加命令列を追加することにより、前記第 2 の演算器と前記第 3 の演算器を接続する第 2 の回路を使用する処理を含む試験命令列を生成し、

前記処理命令列の先頭から前記試験命令列を前記装置に実行させることにより、前記第 1 の回路の故障の有無を判定し、

前記追加命令列から前記試験命令列を前記装置に実行させることにより、前記第2の回路の故障の有無を判定する

処理を前記装置に行わせることを特徴とする試験プログラム。

【請求項6】

第1～第3の演算器を有する装置の試験制御装置であって

前記第1の演算器と前記第2の演算器を接続する第1の回路を使用する処理を含む命令列である処理命令列の先頭に、前記処理命令列に含まれる命令の演算結果に変化を与えない命令列である追加命令列を追加することにより、前記第2の演算器と前記第3の演算器を接続する第2の回路を使用する処理を含む試験命令列を生成する生成部と、

前記処理命令列の先頭から前記試験命令列を前記装置に実行させることにより、前記第1の回路の故障の有無を判定し、前記追加命令列から前記試験命令列を前記装置に実行させることにより、前記第2の回路の故障の有無を判定する制御部

を備えることを特徴とする試験制御装置。

【発明の詳細な説明】

【技術分野】

【0001】

この発明は、情報処理装置の試験方法に関する。

【背景技術】

【0002】

複数の演算器が並列に動作する情報処理装置で、先行して実行される演算命令のレジスタファイルへの書き込みが確定したあと、後続の演算命令の実行によりレジスタファイルを読み出す際に、レジスタ中の同じアドレスに対するアクセスが重複して発生する。レジスタ中の同じアドレスに対するアクセスが重複して行われる場合、レジスタの競合が発生し、後続の演算命令は、先行して実行される演算命令の書き込み処理が完了するのを待つので、並列処理の遅延が発生する。レジスタの競合を避けるため、情報処理装置にバイパス回路を採用することが知られている。バイパス回路によって、先行して実行される演算命令を実行する演算器がレジスタファイルへ演算結果を書き込む前に、後続の演算命令を実行する演算器が演算結果を使用できるようになる。

【0003】

近年の回路の微細化や複雑化により、ハードウェアの故障は微細な電気条件によって発生することがある。また、特異な命令シーケンスで故障が発生するケースもある。バイパス回路が故障すると、並列処理の遅延、データの読み出し故障などが起こりうる。演算器周辺の故障を検出するため、ランダムな試験命令列を生成し、微細な電気条件によるノイズや特異な命令シーケンスで発生する故障を特定する。

【0004】

関連する技術に、演算器のパイプライン機構を検証するために、パイプラインのインターロックを引き起こす命令列を生成する方法がある。インターロックを引き起こす試験命令列を自動的に生成するため、先行して実行される演算命令が使用するレジスタを、後続の演算命令に通知し、後続の演算命令が使用するレジスタを、先行して実行される演算命令が使用するレジスタと重複して使用させるように生成する技術が知られている。（例えば、特許文献1を参照）

【先行技術文献】

【特許文献】

【0005】

【特許文献1】特許公開2001-222442号

【発明の概要】

【発明が解決しようとする課題】

【0006】

上述した従来の試験方法には、以下のような問題がある。

特定の試験命令列は、特定の演算器の間を接続する回路の故障を検出することができる

10

20

30

40

50

ものの、他の演算器の間を接続する回路の故障を検出できないという問題がある。そこで、他の演算器間を接続する回路に対して、故障の検出を行うためには、演算器の数に応じた複数の新しい試験命令列を作成しなければならない。新しく作成された試験命令列は、故障の検出に有効であるかどうかを判断しなければならない、検証に多くの人手と時間を費やしていた。

【0007】

本発明は、1つの側面では、複数の演算器の間を接続する回路の故障を効率的に検出することを目的とする。

【課題を解決するための手段】

【0008】

実施形態にかかる方法では、第1～第3の演算器を有する装置の試験を行うことができる。実施形態にかかる方法では、第1の演算器と第2の演算器を接続する第1の回路を使用する処理を含む命令列である処理命令列を用いて、第2の演算器と第3の演算器を接続する第2の回路を使用する処理を含む試験命令列を生成する。ここで、試験命令列は、処理命令列の先頭に、処理命令列に含まれる命令の演算結果に変化を与えない命令列である追加命令列を追加したものである。処理命令列の先頭から試験命令列を装置に実行させることにより、前記第1の回路の故障の有無を判定する。また、追加命令列から試験命令列を装置に実行させることにより、第2の回路の故障の有無を検出する。

【発明の効果】

【0009】

複数の演算器の間を接続する回路の故障を効率的に検出することができる。

【図面の簡単な説明】

【0010】

【図1】試験方法の例である。

【図2】情報処理装置の例を示す図である。

【図3】情報処理装置のハードウェア構成の例を示す図である。

【図4】試験命令列を使用した第1の試験で行われる処理の例を説明するフローチャートである。

【図5A】試験開始アドレスとしてメモリアドレス401を指定した試験の例を示す図である。

【図5B】試験開始アドレスとしてメモリアドレス402を指定した試験の例を示す図である。

【図5C】試験開始アドレスとしてメモリアドレス403を指定した試験の例を示す図である。

【図5D】試験開始アドレスとしてメモリアドレス404を指定した試験の例を示す図である。

【図6】試験命令列を使用した第2の試験で行われる処理の例を説明するフローチャートである。

【図7】試験命令列の生成例を説明するフローチャートである。

【図8】試験命令列を使用した試験制御方法の例を説明するフローチャートである。

【図9】試験命令列の演算結果と予測値を比較する処理を説明するフローチャートである。

【図10】試験結果データの例である。

【発明を実施するための形態】

【0011】

以下、本実施形態について、図面を参照しながら詳細に説明する。

図1は、試験方法の例である。図1(a)は、試験命令列100と、試験制御部101とを示す。試験命令列100は、メモリアドレス111、112にNOP命令を備え、メモリアドレス113～118の各々に演算命令を備える処理命令列110と、メモリアドレス119にリターン命令とを備える命令列である。NOP命令は、何も処理を実行しな

10

20

30

40

50

い (No Operation) 命令であり、また、他の演算命令に何ら影響を与えない演算命令である。メモリアドレス 113～118 に格納された演算命令は、特定の演算器間で発生した故障を検出する演算命令列である。簡略化のため、特定の演算器間で発生した故障を検出する演算命令列のことを、「処理命令列」と称す。

【0012】

パイプライン構造の複数の演算器は、互いにバイパス回路で接続されており、演算命令がレジスタ中の同じアドレスに対するアクセスを重複して行っても並列処理を円滑に行える。以下、処理命令列 110 の、メモリアドレス 113 とメモリアドレス 117 に格納されている演算命令の組み合わせで、レジスタ中の特定のアドレスへのアクセスが重複する場合を例として述べる。この場合、処理命令列 110 を含む命令列が実行されると、メモリアドレス 113 に格納された演算命令を実行する演算器と、メモリアドレス 117 に格納された演算命令を実行する演算器の間を接続する回路が使用される。

10

【0013】

試験命令列は、処理命令列の先頭に、少なくとも 1 つの NOP 命令を備えている。また、NOP 命令は、処理命令列に干渉しない演算命令であって、処理命令列が使用しているレジスタと競合せず、処理命令列の演算結果に影響を与えない演算命令であれば、他の演算命令に代替してもよい。リターン命令は、処理命令列の演算結果を試験制御部に返し、また、処理を試験制御部に返す演算命令である。

【0014】

試験制御部 101 は、試験命令列を複数の演算器に実行させる命令として、試験開始命令を出力する。試験開始命令は、試験命令列内の特定のメモリアドレスを、試験を開始するアドレスに指定する分岐命令を含む。以下、試験を開始するアドレスを「試験開始アドレス」と記載することがある。分岐命令は、試験命令列内の各 NOP 命令のメモリアドレスや、処理命令列の先頭のメモリアドレスを試験開始アドレスとして指定することができる。例えば、試験制御部 101 が試験開始アドレスとしてメモリアドレス 113 を指定すると、試験命令列 100 は、処理命令列 110 の先頭の演算命令から複数の演算器に実行される。試験制御部 101 が試験開始アドレスとしてメモリアドレス 112 を指定すると、メモリアドレス 112 に格納されている NOP 命令以降の命令が複数の演算器に実行される。試験制御部 101 が、試験開始アドレスとしてメモリアドレス 111 を指定すると、メモリアドレス 111 に格納されている NOP 命令以降の命令が複数の演算器に実行される。

20

30

【0015】

図 1 の例では、試験命令列 100 は、分岐命令によって試験開始アドレスを指定することにより、1 つの試験命令列を、以下の 3 つのパターンの命令列として使用できる。

(1) メモリアドレス 113 に格納されている処理命令列 110 の先頭の演算命令から複数の演算器に実行されるパターン

(2) メモリアドレス 112 に格納されている NOP 命令から複数の演算器に実行されるパターン

(3) メモリアドレス 111 に格納されている NOP 命令から複数の演算器に実行されるパターン

40

パターン (1) ～ (3) の命令列において、処理命令列は、命令シーケンスを変更されない。

【0016】

図 1 (b) に、演算器の割り当て状態の例を示す。演算器の割り当て状態は、試験命令列 100 を (1) ～ (3) のパターンで実行した場合に、各メモリアドレスに格納されている演算命令が、パイプライン処理によって、演算器 A ～ C のどの演算器に割り当てられたかを示した例である。

【0017】

図 1 (b) のパターン (1) の列では、試験開始アドレスとしてメモリアドレス 113 が指定されている。このため、各メモリアドレスの演算命令は、以下の演算器に割り当て

50

られる。パターン（１）の例では、メモリアドレス１１１，１１２に格納されているNOP命令は、演算器で実行されない。

メモリアドレス１１３の演算命令：演算器A
 メモリアドレス１１４の演算命令：演算器B
 メモリアドレス１１５の演算命令：演算器C
 メモリアドレス１１６の演算命令：演算器A
 メモリアドレス１１７の演算命令：演算器B
 メモリアドレス１１８の演算命令：演算器C
 メモリアドレス１１９のリターン命令：演算器A

【００１８】

10

パターン（２）の列では、試験開始アドレスとしてメモリアドレス１１２が指定されている。そこで、各メモリアドレスの演算命令は、以下の演算器に割り当てられる。パターン（２）の例では、メモリアドレス１１１に格納されているNOP命令は、演算器で実行されない。

メモリアドレス１１２のNOP命令：演算器A
 メモリアドレス１１３の演算命令：演算器B
 メモリアドレス１１４の演算命令：演算器C
 メモリアドレス１１５の演算命令：演算器A
 メモリアドレス１１６の演算命令：演算器B
 メモリアドレス１１７の演算命令：演算器C
 メモリアドレス１１８の演算命令：演算器A
 メモリアドレス１１９のリターン命令：演算器B

20

【００１９】

パターン（３）の列では、試験開始アドレスとしてメモリアドレス１１１が指定されている。そこで、各メモリアドレスの演算命令は、以下の演算器に割り当てられる。

メモリアドレス１１１のNOP命令：演算器A
 メモリアドレス１１２のNOP命令：演算器B
 メモリアドレス１１３の演算命令：演算器C
 メモリアドレス１１４の演算命令：演算器A
 メモリアドレス１１５の演算命令：演算器B
 メモリアドレス１１６の演算命令：演算器C
 メモリアドレス１１７の演算命令：演算器A
 メモリアドレス１１８の演算命令：演算器B
 メモリアドレス１１９のリターン命令：演算器C

30

【００２０】

なお、試験開始アドレスを指定する分岐命令は、演算器に試験命令列を実行させる前に、実行中の演算命令を全てキャンセルする命令を含む。更に、パイプラインフラッシュをする命令を含む。これにより、試験命令列は毎回規定の演算器から処理を順に割り当てることができる。図１（b）の例では、パターン（１）～（３）の先頭の演算命令は、演算器Aに割り当てられている。演算器Aのあとは、演算器B、Cの順に命令が割り当てられる。

40

【００２１】

メモリアドレス１１３とメモリアドレス１１７に格納されている演算命令は、パターン（１）の場合、太線枠で示すように演算器Aと演算器Bに割り当てられる。このため、パターン（１）の命令列を用いると、演算器A－B間のバイパス回路が使用される。バイパス回路に故障がない場合、処理命令列は正常に処理を終了し、特定の演算結果を試験制御部に返す。しかし、バイパス回路に故障が発生した場合、処理命令列は正常に処理を終了できず、異常値やエラー値を返す。例えば、演算器A－B間のバイパス回路が故障し、処理命令列を正常に処理できない場合に、処理命令列１１０の演算結果として特定のエラー値が出力される。処理命令列１１０の演算結果として予測される予測値と異なる演算結果

50

が返ってきた場合に、演算器 A－B 間のバイパス回路が故障していると判断する。

【0022】

次に、試験開始アドレスとしてメモリアドレス 112 が指定された場合、パターン (2) で説明したように、試験に使用される命令列は、処理命令列 110 の先頭に、メモリアドレス 112 に格納された NOP 命令を備える。メモリアドレス 113 とメモリアドレス 117 に格納されている演算命令は、パターン (2) の場合、点線枠で示すように演算器 B と演算器 C に割り当てられる。このため、パターン (2) の命令列が実行されると、演算器 B－C 間のバイパス回路が使用される。なお、パターン (2) は、パターン (1) と比べて、NOP 命令が先頭に設置されているが、NOP 命令は、処理命令列の演算結果に影響を与えない演算命令である。そのため、パターン (2) は、演算器 B－C 間のバイパス回路の故障を検出するのに有効な演算命令列である。

10

【0023】

試験開始アドレスとしてメモリアドレス 111 が指定された場合、パターン (3) で説明したように、試験に使用される命令列は、処理命令列 110 の先頭に、メモリアドレス 111、112 に格納された NOP 命令を備える。メモリアドレス 113 とメモリアドレス 117 に格納されている演算命令は、パターン (3) の場合、一点鎖点枠で示すように演算器 C と演算器 A に割り当てられる。このため、パターン (3) の命令列が実行されると、演算器 C－A 間のバイパス回路が使用される。

【0024】

図 2 は、情報処理装置の例を示す。図 2 は、試験制御部 101、データベース 260、複数の演算器 270、レジスタ 280 を備える。試験制御部 101 は、試験命令列生成部 210、試験開始アドレス設定部 220、試験データ設定部 230、試験命令列制御部 240、試験結果出力部 250 を備える。複数の演算器は、バイパス回路 271a～271c を備える。なお、図 2 の例では、演算器の数は 3 であるが、情報処理装置に含まれる演算器の数は、実装に応じて任意に変更されるものとする。また、情報処理装置に含まれる演算器の数に応じて、演算器の間を接続するバイパス回路の数も変動するものとする。

20

【0025】

試験命令列生成部 210 は、試験命令列を生成する。試験命令列は、処理命令列の先頭に少なくとも 1 つの NOP 命令を備え、次に処理命令列と、リターン命令とを備える。試験命令列生成部 210 は、処理命令列をデータベース 260 より取得する。データベース 260 は、複数の処理命令列と、各処理命令列に対応した、回路が正常な場合の予測される演算結果である予測値とを保持している。また、データベース 260 は、予測値の代わりに、故障時の各処理命令列に対応した演算結果としてのエラー値を保持していてもよく、予測値とエラー値の両方を保持してもよい。予測値とエラー値は、処理命令列を演算器に実行させて得られた実際の演算結果と、予測値又はエラー値とを比較することで、回路が正常かどうかを判断するために使用される。また、データベース 260 は、処理命令列を実行する際の、試験に使用する所定の試験環境情報を保持してもよい。処理命令列と、予測値と、エラー値、試験環境情報は、試験に応じてオペレータによって設定されてもよい。

30

【0026】

さらに、試験命令列生成部 210 は、情報処理装置の備える演算器の数を取得する。演算器の数は、NOP 命令の最適な数を算出するために用いられる。試験命令列は、少なくとも 1 つの NOP 命令を備えればよいが、演算器の総数よりも小さい数の NOP 命令を有することが望ましい。例えば、図 1 (a) に示す試験命令列 100 からメモリアドレス 111 の NOP 命令を削除し、試験命令列 100' がメモリアドレス 112 の NOP 命令と処理命令列の場合を考えたい。試験命令列 100' は、NOP 命令を 1 つしか有しないため、パターン (1) とパターン (2) の試験での命令列として、使用できるが、パターン (3) の試験には使用できない。そのため、図 1 (b) の演算器の割り当て状態 120' は、パターン (3) の列を持たない。すると、試験命令列 100' は、パターン (1) のように、演算器 A－B 間のバイパス回路の故障の検出に有効であり、また、パターン (2

40

50

）のように、演算器 B－C 間のバイパス回路の故障の検出に有効である。しかしながら、演算器 C－A 間のバイパス回路の故障の検出に有効ではない。よって、演算器間を網羅して故障を検出するためには、試験命令列 1 0 0 は、演算器 A～C の 3 つの演算器に対してパターン（1）～（3）の試験を行うのが好ましい。そのためには、2 つの NOP 命令を備えた試験命令列が好ましい。同様に、演算器が n 個のケースは、パターン（1）～（n）の試験を行うのが好ましい。ここで、n は、2 以上の整数である。すると、n－1 個の NOP 命令を備えた試験命令列を使用するのが好ましい。しかしながら、試験時間を短縮したい場合や、既に特定の演算器間には故障がないことが判明している場合は、演算器間を網羅して試験を行わない。この場合、実施したい試験に応じて、NOP 命令の数を調整した試験命令列を使用すればよい。

10

【0027】

情報処理装置が備える演算器の数、及び、データベース 2 6 0 は、メモリ装置又は記憶装置に記憶される。情報装置が備える演算器の数は、データベース 2 6 0 に含まれてもよい。また、試験制御部 1 0 1 も、メモリ装置又は記憶装置に記憶される。

【0028】

試験開始アドレス設定部 2 2 0 は、試験開始アドレスの初期値としてメモリアドレスを設定する。試験開始アドレスの初期値は、分岐命令で指定される試験開始アドレスのメモリアドレスを指し、所定のメモリや記憶装置に記憶される。

【0029】

つぎに、試験データ設定部 2 3 0 は、試験に使用するデータを設定する。試験データ設定部 2 3 0 は、試験命令列生成部 2 1 0 で生成された試験命令列を所定のメモリや記憶装置に格納する。また、試験データ設定部 2 3 0 は、演算器の数、予測値、所定の試験環境情報、試験番号などを所定のメモリや記憶装置やレジスタに設定する。また、後述するが、試験データ設定部 2 3 0 は、試験開始アドレスが変更された場合、変更後の試験開始アドレスを設定する。

20

【0030】

試験命令列制御部 2 4 0 は、試験開始命令を出力し、試験命令列を複数の演算器 2 7 0 に実行させる。その後、複数の演算器 2 7 0 から演算結果を受信する。更に、試験命令列制御部 2 4 0 は、受信した演算結果を予測値と比較し、比較結果を試験結果出力部 2 5 0 に出力する。また、詳細は後述するが、試験命令列制御部 2 4 0 は、次に試験命令列を実行するときの試験開始アドレスを算出する。試験開始命令は、分岐命令を含む。試験開始命令を受けた複数の演算器は、試験命令列を分岐命令で指定された試験開始アドレスより、各演算器に処理を割り当て、演算命令を実行する。次に、試験命令列制御部 2 4 0 は、試験命令列が各演算器で実行された演算結果を受信し、データベース 2 6 0 より取得した予測値と比較する。受信した演算結果が、予測値と一致した場合、所定の演算器間の回路は正常に動作していると判断できる。また、受信した演算結果が予測値と一致しない場合は、所定の演算器間の回路は故障していると判断される。試験命令列制御部 2 4 0 は、回路が正常かどうか判断した比較結果を試験結果出力部 2 5 0 に出力する。試験結果出力部 2 5 0 は、試験命令列制御部 2 4 0 から比較結果を受信し、試験結果を出力する。

30

【0031】

複数の演算器 2 7 0 は、演算器 A～C を備え、演算器 A～C の各々はバイパス回路 2 7 1 a～2 7 1 c で接続されている。演算器 A－B 間は、バイパス回路 2 7 1 a で接続されており、演算器 B－C 間は、バイパス回路 2 7 1 b、演算器 C－A 間は、バイパス回路 2 7 1 c で接続されている。試験命令列制御部 2 4 0 の試験開始命令によって、各演算器は、試験命令列 1 0 0 のうちの割り当てられた命令を処理する。例えば、図 1 (a) に示す試験命令列 1 0 0 がパターン（1）で実行されると、バイパス回路 2 7 1 a に故障が発生しているかが判定されることになる。同様に、パターン（2）の実行により、バイパス回路 2 7 1 b の故障の有無が判定され、パターン（3）の実行により、バイパス回路 2 7 1 c の故障の検出が行われる。レジスタ 2 8 0 は、試験命令列が実行される際に、適宜使用される。

40

50

【0032】

図3は、情報処理装置のハードウェア構成の例である。図3は、情報処理装置300と、ディスプレイ装置307を有する。情報処理装置300は、プロセッサメモリ302、入力装置インターフェース303、バス304、記憶装置305、出力装置インターフェース306を備える。図3の例では、プロセッサは、CPU (Central Processing Unit) 301であるが、情報処理装置は、実装に応じた任意の制御回路を備えていても良い。CPU 301は、バス304を介して、メモリ302、入力装置インターフェース303、記憶装置305、出力装置インターフェース306と相互に接続されている。CPU 301は、記憶装置305に格納されたプログラム、又は、メモリ302にロードされたプログラムを実行する。記憶装置305は、磁気ディスク装置、光ディスク装置、光磁気ディスク装置あるいは半導体記憶装置を含み、コンピュータ読み取り可能な記録媒体を扱うドライブ装置も含む。ドライブ装置が扱う記録媒体としては、磁気テープ、メモリカード、光ディスク (CD-ROM、DVD-ROMなど)、光磁気ディスク (MO、MDなど) といった、任意のコンピュータ読み取り可能な記録媒体を使用することができる。記憶装置305は、実施例に係る各処理を行うプログラム、及び、実施例において使用する各種データを格納している。

10

【0033】

次に、図3のハードウェア構成を、図2の構成に基づいて説明する。複数の演算器270は、CPU 301に該当し、レジスタ280は、CPU 301に含まれる。試験制御部101の処理を実現するために使用されるプログラムは、メモリ302又は記憶装置305に記憶される。更に、試験制御部101は、試験命令列生成部210、試験開始アドレス設定部220、試験データ設定部230、試験命令列制御部240、試験結果出力部250を備え、CPU 301によって実現される。試験命令列生成部210は、試験命令列を生成する際に、データベース260から処理命令列や予測値などの各種情報を取得する。データベース260は、記憶装置に記憶されていてもよく、入力装置インターフェース303を介した不図示の外部記憶装置であってもよい。また、処理命令列や予測値などの各種情報を、キーボードやマウスといった入力装置インターフェース303を使用して、オペレータによって入力されてもよい。試験結果出力部250は、演算結果を出力する際に、出力装置インターフェース306を介して、ディスプレイ装置307にその演算結果を表示する。

20

30

【0034】

図4は、試験命令列を使用した第1の試験で行われる処理の例を説明するフローチャートである。第1の試験は、1つの試験開始アドレスを指定して試験命令列を処理する試験である。第1の試験で行われる一連の処理の例を、ステップS101～S107に示す。試験制御部が、ステップS101～S104と、ステップS106～S107とを行い、複数の演算器が、ステップS105を処理する。

【0035】

試験命令列生成部210は、試験命令列を生成する (ステップS101)。試験開始アドレス設定部220は、試験開始アドレスの初期値として用いるメモリアドレスを設定する (ステップS102)。試験データ設定部230は、試験に使用するデータを設定する (ステップS103)。試験に使用するデータは、試験命令列、演算器の数、予測値、所定の試験環境情報、試験番号などである。試験命令列制御部240は、試験開始アドレスを指定し、試験開始命令を出力する (ステップS104)。第1の試験では、試験開始アドレスとして、S102で設定された試験開始アドレスの初期値が使用される。複数の演算器270は、試験開始アドレスから試験命令列を実行する (ステップS105)。なお、試験命令列が複数の演算器で実行されている間は、他の演算命令の割り込みがないことが好ましい。そのため、試験開始命令を受信すると、複数の演算器は実行中の演算命令をキャンセルし、試験命令列の実行を優先して行う。また、試験制御部は、試験開始命令を出力した後は、演算結果を受信するまで処理を行わない。試験命令列制御部240は、演算結果を受信し、演算結果を予測値と比較する (ステップS106)。試験結果出力部2

40

50

50は、ステップS106から比較結果を取得し、比較結果を試験結果として出力する（ステップS107）。

【0036】

第2の試験の例を、図5A～Dと図6で説明する。第2の試験の説明では、1つの試験命令列が出力されてから、試験命令列から得られる複数のパターンの命令列が実行され、試験結果が複数のパターンについてまとめて得られる場合について述べる。また、第2の試験の説明では、演算器A～Dの4つの演算器を備える情報処理装置での試験を例として説明する。

【0037】

以下の説明では図5Aに示す試験命令列が生成されるものとする。試験命令列生成部210は、処理命令列をデータベース260から取得して、メモリアドレス404～411に格納したとする。さらに、試験命令列生成部210は、演算器の数が4であることから、処理命令列の先頭に付加するNOP命令の数を3に決定したとする。すると、個々で生成される図5Aに示すとおりになる。図5Aの例では、試験命令列400は、メモリアドレス401～403にNOP命令と、処理命令列である演算命令をメモリアドレス404～411に備え、メモリアドレス412にリターン命令を備える。図5Aに示す試験命令列400では、メモリアドレス404とメモリアドレス409に格納された命令を実行する演算器が、レジスタ中の同じアドレスに対するアクセスを重複して行うものとする。また、分岐命令は、試験命令列内の各NOP命令のメモリアドレスや、処理命令列の先頭のメモリアドレスを試験開始アドレスとして指定することができる。よって、試験命令列400は、試験開始アドレスとして、メモリアドレス401～404を指定することができる。第2の試験では、メモリアドレス401～404が試験開始アドレスとして一度ずつ指定されることにより、以下の4つのパターンの命令列を用いた試験が行われる。

【0038】

(i) メモリアドレス401に格納されているNOP命令から複数の演算器に実行されるパターン

(ii) メモリアドレス402に格納されているNOP命令から複数の演算器に実行されるパターン

(iii) メモリアドレス403に格納されているNOP命令から複数の演算器に実行されるパターン

(iv) メモリアドレス404に格納されている処理命令列の先頭の演算命令から複数の演算器に実行されるパターン（メモリアドレス401、402、403に格納されているNOP命令は、演算器で実行されない）

【0039】

図6は、試験命令列を使用した第2の試験で行われる処理の例を説明するフローチャートである。第2の試験は、パターン(i)、パターン(ii)、パターン(iii)、パターン(iv)の順番で、試験を連続して行う。

【0040】

<1回目の試験：パターン(i)>

最初の処理として、試験命令列生成部210は、試験命令列400を生成する（ステップS201）。試験命令列400は、パターン(i)～(iv)で共通して使用する試験命令列である。そのため、2回目以降の試験では、試験命令列400を再度生成する処理を有さない。試験開始アドレス設定部220は、試験開始アドレスの初期値としてメモリアドレス401を設定する（ステップS202）。試験データ設定部230は、試験に使用するデータを設定する（ステップS203）。試験に使用するデータは、試験命令列400、演算器の数、予測値、所定の試験環境情報、試験番号などである。試験命令列制御部240は、試験開始アドレスとしてメモリアドレス401を指定し、試験開始命令を出力する（ステップS204）。複数の演算器270は、試験開始アドレスから試験命令列を実行する（ステップS205）。パターン(i)の試験では、試験命令列400の各演算命令は、図5Aに示すように演算器A～Dに割り当てられる。メモリアドレス404とメ

モリアドレス409に格納された命令を実行する演算器が、レジスタ中の同じアドレスに対するアクセスを重複して行う。そのため、パターン(i)の試験は、演算器D-A間のバイパス回路を使用している。試験命令列制御部240は、パターン(i)の演算結果を受信し、演算結果を予測値と比較処理を行い、演算器D-A間のバイパス回路が故障しているかを判断する(ステップS206)。試験命令列制御部240は、次に実行する試験命令列の試験開始アドレスを算出する(ステップS207)。試験命令列制御部240は、次に実行する試験命令列の試験開始アドレスをメモリアドレス402とする。試験命令列制御部240は、演算器数と同じ回数の試験を実行したかを判断する(ステップS208)。パターン(i)は1回目の試験であるため、演算器数と同じ4回の試験を実行しておらず、ステップS208では、NOと判断され、処理をステップS203に返す。

10

【0041】

<2回目の試験：パターン(ii)>

試験データ設定部230は、試験に使用するデータを設定する(ステップS203)。ここで設定されるデータは、1回目の試験のステップS207で算出された、次に実行する試験命令列の試験開始アドレスである。1回目の試験のステップS207の算出の結果、試験データ設定部230は、次に実行する試験命令列の試験開始アドレスとして、メモリアドレス402を設定する。なお、1回目の試験のステップS203で設定した試験命令列400、演算器の数、予測値、所定の試験環境情報は、2回目の試験でも同じものを使用する。試験命令列制御部240は、試験開始アドレスとしてメモリアドレス402を指定し、試験開始命令を出力する(ステップS204)。複数の演算器270は、試験開始アドレスから試験命令列を実行する(ステップS205)。パターン(ii)の試験では、試験命令列400の各演算命令は、図5Bに示すように、演算器A~Dに割り当てられる。試験命令列400のメモリアドレス401に格納されているNOP命令は、演算器で実行されない。メモリアドレス404とメモリアドレス409に格納された命令を実行する演算器が、レジスタ中の同じアドレスに対するアクセスを重複して行う。そのため、パターン(ii)の試験は、演算器C-D間のバイパス回路を使用している。試験命令列制御部240は、パターン(ii)の演算結果を受信し、演算結果を予測値と比較することにより、演算器C-D間のバイパス回路が故障しているかを判断する(ステップS206)。試験命令列制御部240は、次に実行する試験命令列の試験開始アドレスを算出する(ステップS207)。試験命令列制御部240は、次に実行する試験命令列の試験開始アドレスをメモリアドレス403とする。試験命令列制御部240は、演算器数と同じ回数の試験を実行したかを判断する(ステップS208)。パターン(ii)は2回目の試験であるため、演算器数と同じ4回の試験を実行していない。このため、ステップS208では、NOと判断され、処理がステップS203から繰り返される。

20

30

【0042】

<3回目の試験：パターン(iii)>

試験データ設定部230は、試験に使用するデータを設定する(ステップS203)。ここで設定されるデータは、2回目の試験のステップS207で算出された、次に実行する試験命令列の試験開始アドレスである。2回目の試験のステップS207の算出の結果、試験データ設定部230は、次に実行する試験命令列の試験開始アドレスとして、メモリアドレス403を設定する。なお、1回目の試験のステップS203で設定した試験命令列400、演算器の数、予測値、所定の試験環境情報は、3回目の試験でも同じものを使用する。試験命令列制御部240は、試験開始アドレスとしてメモリアドレス403を指定し、試験開始命令を出力する(ステップS204)。複数の演算器270は、試験開始アドレスから試験命令列を実行する(ステップS205)。パターン(iii)の試験では、試験命令列400の各演算命令は、図5Cに示すように、演算器A~Dに割り当てられる。試験命令列400のメモリアドレス401、402に格納されているNOP命令は、演算器で実行されない。メモリアドレス404とメモリアドレス409に格納された命令を実行する演算器が、レジスタ中の同じアドレスに対するアクセスを重複して行う。そのため、パターン(iii)の試験は、演算器B-C間のバイパス回路を使用している。試験

40

50

命令列制御部 240 は、パターン (iii) の演算結果を受信し、演算結果を予測値と比較することにより、演算器 B-C 間のバイパス回路が故障しているかを判断する (ステップ S 206)。試験命令列制御部 240 は、次に実行する試験命令列の試験開始アドレスを算出する (ステップ S 207)。試験命令列制御部 240 は、次に実行する試験命令列の試験開始アドレスをメモリアドレス 404 とする。試験命令列制御部 240 は、演算器数と同じ回数の試験を実行したかを判断する (ステップ S 208)。パターン (iii) は 3 回目の試験であるため、演算器数と同じ 4 回の試験を実行していない。このため、ステップ S 208 では、NO と判断され、処理をステップ S 203 から繰り返される。

【0043】

< 4 回目の試験：パターン (iv) >

試験データ設定部 230 は、試験に使用するデータを設定する (ステップ S 203)。ここで設定されるデータは、3 回目の試験のステップ S 207 で算出された、次に実行する試験命令列の試験開始アドレスである。3 回目の試験のステップ S 207 の算出の結果、試験データ設定部 230 は、次に実行する試験命令列の試験開始アドレスとして、メモリアドレス 404 を設定する。なお、1 回目の試験のステップ S 203 で設定した試験命令列 400、演算器の数、予測値、所定の試験環境情報は、4 回目の試験でも同じものを使用する。試験命令列制御部 240 は、試験開始アドレスとしてメモリアドレス 404 を指定し、試験開始命令を出力する (ステップ S 204)。複数の演算器 270 は、試験開始アドレスから試験命令列を実行する (ステップ S 205)。パターン (iv) の試験では、試験命令列 400 の各演算命令は、図 5 B に示すように、を演算器 A~D に割り当てられる。試験命令列 400 のメモリアドレス 401、402 に格納されている NOP 命令は、演算器で実行されない。メモリアドレス 404 とメモリアドレス 409 に格納された命令を実行する演算器が、レジスタ中の同じアドレスに対するアクセスを重複して行う。そのため、パターン (iv) の試験は、演算器 A-B 間のバイパス回路を使用している。試験命令列制御部 240 は、パターン (iv) の演算結果を受信し、演算結果を予測値と比較することにより、演算器 A-B 間のバイパス回路が故障しているかを判断する (ステップ S 206)。試験命令列制御部 240 は、次に実行する試験命令列の試験開始アドレスを算出する (ステップ S 207)。試験命令列制御部 240 は、演算器数と同じ回数の試験を実行したかを判断する (ステップ S 208)。パターン (iv) は 4 回目の試験であるため、演算器数と同じ 4 回の試験を実行している。そのため、ステップ S 208 では、YES と判断され、処理をステップ S 209 に移す。

【0044】

試験結果出力部 250 は、全試験におけるステップ S 206 の比較結果を受信し、試験結果として出力する (ステップ S 209)。ステップ S 209 が終了すると、試験制御部は処理を終了する。

【0045】

図 6 は、第 2 の試験の例として、パターン (i) を 1 回目の試験とし、パターン (ii)、パターン (iii)、パターン (iv) を順番に実行するように試験を行っているが、実施例は上記のものに限定されない。パターン (iv) の試験を 1 回目に行い、パターン (iii)、パターン (ii)、パターン (i) の順番に試験を実行してもよい。

【0046】

図 7 は、試験命令列の生成例を説明するフローチャートである。図 7 では、図 6 のステップ S 201 の処理を詳しく説明している。試験命令列生成部 210 は、試験を行う演算器の数を指す arithmetic_no を、データベース 260 より取得する (ステップ S 301)。取得した arithmetic_no は、所定のメモリや記憶装置に記憶される。試験命令列生成部 210 は、試験命令列の先頭のメモリアドレスを指す start_address 番地を取得する (ステップ S 302)。取得する試験命令列の先頭のメモリアドレスは、試験命令列を生成するために使用される任意のメモリアドレスである。取得した start_address 番地は、所定のメモリや記憶装置に記憶される。

【0047】

ステップ S 3 0 3 ~ S 3 0 5 は、試験命令列に NOP 命令を書き込む処理を示す。試験命令列生成部 2 1 0 は、arithmetic_no から 1 を減算し、減算された値が 0 より大きいかを判断する（ステップ S 3 0 3）。arithmetic_no から 1 を減算した値が 0 より大きい場合、試験命令列生成部 2 1 0 は、start_address 番地に、NOP 命令を書き込む（ステップ S 3 0 3 で YES、ステップ S 3 0 4）。試験命令列制御部 2 4 0 は、start_address 番地に 1 つの NOP 命令の格納に使用されるメモリサイズを加算する。また、試験命令列制御部 2 4 0 は、arithmetic_no から 1 を減算する。試験命令列制御部 2 4 0 は、1 つの NOP 命令の格納に使用されるメモリサイズを加算した start_address 番地と、減算された arithmetic_no を、ステップ S 3 0 3 に返し、ステップ S 3 0 3 以降の処理を繰り返す。

10

【0048】

ステップ S 3 0 6 ~ S 3 0 8 は、試験命令列に処理命令列とリターン命令を書き込む処理を示す。arithmetic_no から 1 を減算した値が 0 になると、試験命令列制御部 2 4 0 は、データベース 2 6 0 から、処理命令列を取得する（ステップ S 3 0 3 で NO、ステップ S 3 0 6）。試験命令列生成部 2 1 0 は、ステップ S 3 0 5 で最後に算出された start_address 番地に、処理命令列を書き込む（ステップ S 3 0 7）。試験命令列生成部 2 1 0 は、処理命令列の末尾にリターン命令を書き込む（ステップ S 3 0 8）。ステップ S 3 0 8 が終了すると、試験命令列生成部 2 1 0 は処理を終了する。

【0049】

図 8 は、試験命令列を使用した試験制御方法の例を説明するフローチャートである。図 8 では、図 6 のステップ S 2 0 2 ~ S 2 0 8 を詳しく説明している。

20

試験開始アドレス設定部 2 2 0 は、試験番号を指す Test_No に 0 を設定する（ステップ S 4 0 1）。試験番号は、試験開始アドレスに対応づけて付与される番号である。試験開始アドレス設定部 2 2 0 は、試験開始アドレスの初期値としてメモリアドレスを指す test_address を設定する（ステップ S 4 0 2）。test_address は、以下の数式 1 で算出される。

【数 1】

$test_address = start_address + (arithmetic_no - 1) \times NOP\text{命令のメモリサイズ}$

【0050】

上記の数式で使用される start_address は、ステップ S 3 0 2 で取得された start_address 番号である。また、上記の数式で使用される arithmetic_no は、ステップ S 3 0 1 で取得された arithmetic_no である。

30

【0051】

試験データ設定部 2 3 0 は、試験に使用するデータを設定する（ステップ S 4 0 3）。ここで設定されるデータは、演算器の数、予測値、所定の試験環境情報、試験番号である。試験命令列制御部 2 4 0 は、試験開始アドレスを指定し、試験開始命令を出力する（ステップ S 4 0 4）。複数の演算器 2 7 0 は、試験開始アドレスから試験命令列を実行する（ステップ S 4 0 5）。複数の演算器 2 7 0 は、ステップ S 4 0 5 の演算結果を試験命令列制御部 2 4 0 に送る。試験命令列制御部 2 4 0 は、ステップ S 4 0 5 の演算結果を複数の演算器 2 7 0 から受信する（ステップ S 4 0 6）。試験命令列制御部 2 4 0 は、受信した演算結果と予測値を比較し、比較結果を出力する（ステップ S 4 0 7）。試験命令列制御部 2 4 0 は、test_address に 1 つの NOP 命令の格納に使用されるメモリサイズを加算する（ステップ S 4 0 8）。試験命令列制御部 2 4 0 は、Test_No に 1 を加算する（ステップ S 4 0 9）。試験命令列制御部 2 4 0 は、試験の回数が、演算器の数以下であるかを判断する（ステップ S 4 1 0）。試験データ設定部 2 3 0 は、試験に使用するデータを設定する（ステップ S 4 1 0 が YES、ステップ S 4 0 3）。ステップ S 4 1 0 で YES と判断されると、ステップ S 4 0 8 で算出された test_address と、ステップ S 4 0 9 で算出された Test_No とが、試験データ設定部 2 3 0 で設定される。ステップ S 4 0 3 の処理が実行されると、ステップ S 4 0 4 ~ S 4 1 0 の処理が繰り返される。一方、ステップ S 4 1 0 で NO と判断されると、試験命令列制御部は、処理を終了する（ステップ S 4 1 0 が NO、

40

50

END)。

【0052】

図9は、試験命令列の演算結果と予測値を比較する処理の例を説明するフローチャートである。このフローチャートは、図6のステップS206の処理を詳しく説明している。試験命令列制御部240は、試験命令列の演算結果を取得し、所定のメモリに格納する(ステップS501)。試験命令列制御部240は、データベース260から取得した予測値を所定のメモリより読み出す(ステップS502)。演算結果と予測値を比較する(ステップS503)。予測値は、情報処理装置の環境に応じた、レジスタ毎の予測値を含む。試験命令列制御部240は、ステップS503の比較によって、演算結果と予測値が一致したかを判断する(ステップS504)。演算結果と予測値が一致したと判定されると、試験命令列制御部240は、所定のメモリに、演算結果と予測値が一致したことを書き込む(ステップS504がYES、ステップS505)。一方、演算結果と予測値が一致なかった場合、試験命令列制御部240は、所定のメモリに演算結果と予測値が一致なかったことを書き込む。また、不一致となったレジスタに対応した予測値と演算結果とを所定のメモリに書き込む(ステップS505がNO、ステップS506)。

10

【0053】

図10は、試験結果データの例である。図10は、ステップS505と、ステップS506で所定のメモリに書き込まれた比較結果を、試験結果出力部250が、試験結果の例とする表示の一例である。図10は、最上段の行に試験番号を備え、2行目に各試験番号に対応した試験結果を示す。併せて、図10は、各試験番号に対応して、レジスタ毎にステップS504の比較結果を表している。

20

【0054】

ステップS401において、Test_Noは、初期値として0が設定されている。図6で例示したパターン(i)～パターン(iv)の連続した試験では、1回目の試験のパターン(i)は、Test_Noが0の列に、比較結果が表示される。次に、2回目の試験のパターン(ii)は、Test_Noが1の列に、3回目の試験のパターン(iii)は、Test_Noが2の列に、4回目の試験のパターン(iv)は、Test_Noが3の列に比較結果が表示されている。

【0055】

図10では、Test_Noが1の列において、試験結果がNGと判断されている。そして、Test_Noが1の列を確認すると、レジスタ3においてNGとなっている。更に、レジスタ3には、予測値及び演算結果が表示されている。オペレータは、表示されている予測値及び演算結果を元に、故障が発生している回路を特定する。予めTest_Noが1の列が、演算器C-D間のバイパス回路の故障の検出を行うパターン(ii)の試験であると分かっている場合は、オペレータはTest_Noを参照することにより、故障が発生している回路を特定できる。また、演算器C-D間のバイパス回路が正常にも関わらず、レジスタ3がNGとある場合は、レジスタ3が破損していることも考えられる。

30

【0056】

以上説明したように、実施形態にかかる方法では、特定の演算器間を接続するバイパス回路の試験に使用することができる命令列を処理命令列として使用する。さらに、処理命令列と、処理命令列の演算に影響を与えない命令列を合わせて得られる命令列が、試験命令列として生成される。また、1つの試験命令列を使用するときに、試験を開始するアドレスを変動させることにより、試験対象とするバイパス回路を変更することができる。このため、実施形態にかかる方法を用いると、効率的に、演算器間のバイパス回路の故障の検出を行うことができる。

40

【0057】

<その他>

なお、実施形態は上記に限られるものではなく、様々に変形可能である。以下にその例をいくつか述べる。

【0058】

図6のフローチャートに示す、パターン(i)～(iv)の連続した試験を、パターン(

50

i) から順番に実施しているものの、パターン (i) ~ (iv) の試験をランダムに 1 度ずつ試験するようにしてもよい。

【0059】

図 5 では、演算器 A-B 間のバイパス回路、演算器 B-C 間のバイパス回路、演算器 C-D 間のバイパス回路、演算器 D-A 間のバイパス回路の故障の検出を行う図である。しかし、演算器は他の組み合わせでも、互いに回路で接続されており、図 5 の例は、故障の検出を行う演算器間の回路を限定するものではない。例えば、演算器 A-C 間のバイパス回路、演算器 B-D 間のバイパス回路についての故障の検出を行うようにしてもよい。その場合は、例えば、図 5 A における、メモリアドレス 405 とメモリアドレス 411 に格納された命令を実行する演算器が、レジスタ中の同じアドレスに対するアクセスを重複して行うものとすればよい。

10

【0060】

図 6 のステップ S209 や、図 8 のステップ S410 では、試験の回数を演算器の数と比較して、処理を次のステップに移すことを判断している。しかしながら、図 6 のステップ S209 や、図 8 のステップ S410 の判断基準は、試験の回数と演算器の数の比較に限定されない。図 6 のステップ S209 や、図 8 のステップ S410 は、試験の回数と、予め決定した所定の閾値とを比較するようにしてもよい。この場合、例えば、閾値は、追加された NOP 命令の数とすることができる。

【0061】

図 1、図 5 の例は、演算器の数を限定するものではない。同様に、図 1、図 5 の例は、処理命令列が含む演算命令の数を限定するものではない。

20

【符号の説明】

【0062】

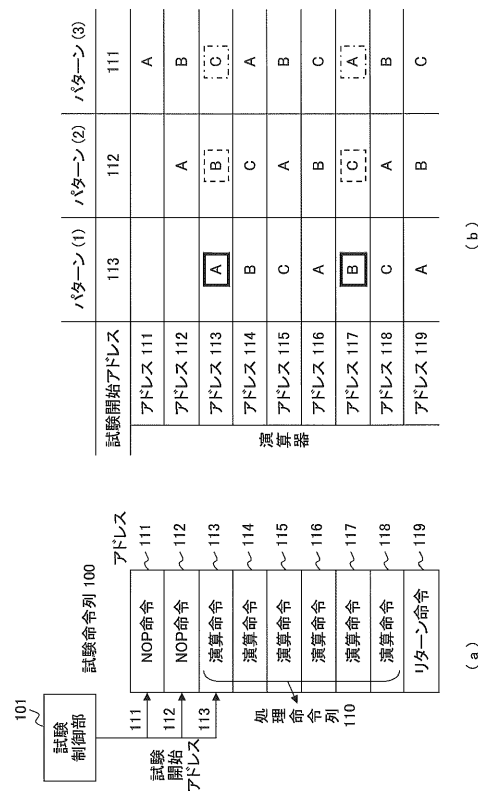
- 100 試験命令列
- 101 試験制御部
- 110 処理命令列
- 111 ~ 119 メモリアドレス
- 210 試験命令列生成部
- 220 試験開始アドレス設定部
- 230 試験データ設定部
- 240 試験命令列制御部
- 250 試験結果出力部
- 260 データベース
- 270 複数の演算器
- 271 a ~ 271 c バイパス回路
- 280 レジスタ
- 300 情報処理装置
- 301 CPU
- 302 メモリ
- 303 入力装置インターフェース
- 304 バス
- 305 記憶装置
- 306 出力装置インターフェース
- 307 ディスプレイ装置
- 400 試験命令列
- 401 ~ 412 メモリアドレス

30

40

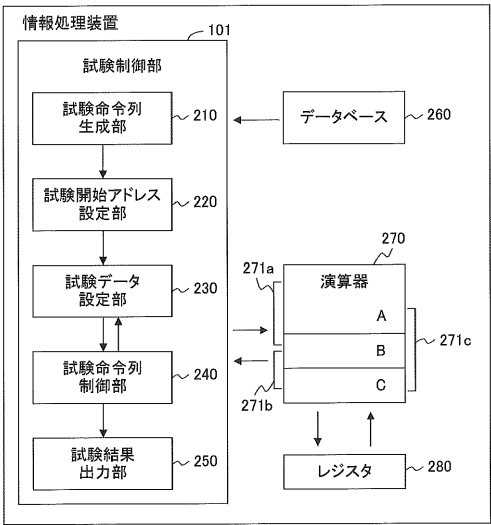
【図 1】

試験方法の例



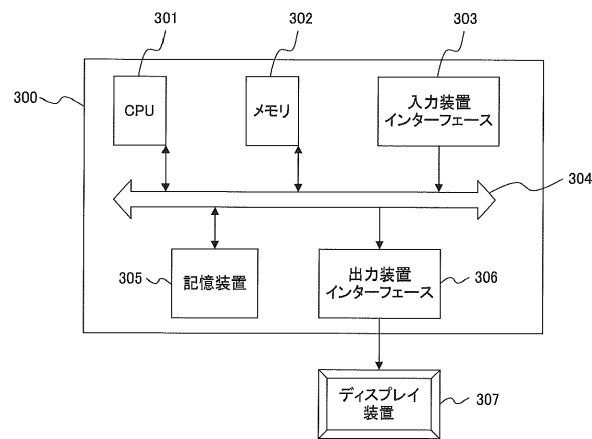
【図 2】

情報処理装置の例



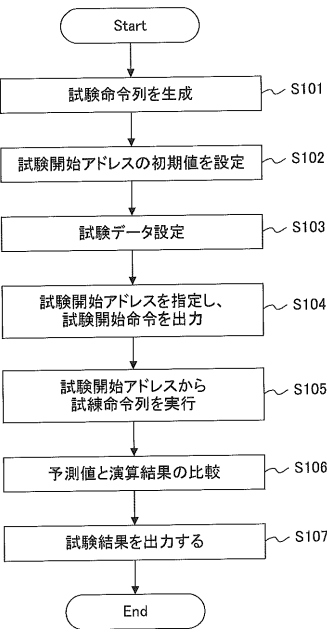
【図 3】

ハードウェア構成例



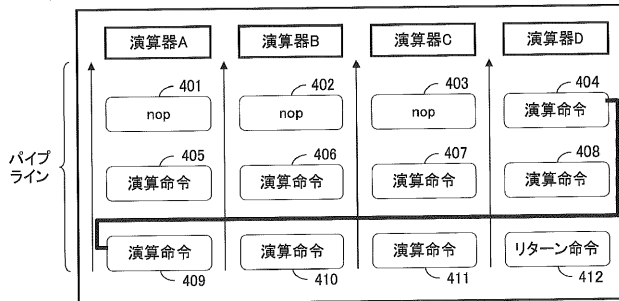
【図 4】

試験命令列を使用した第1の試験で行われる
処理の例を説明するフローチャート



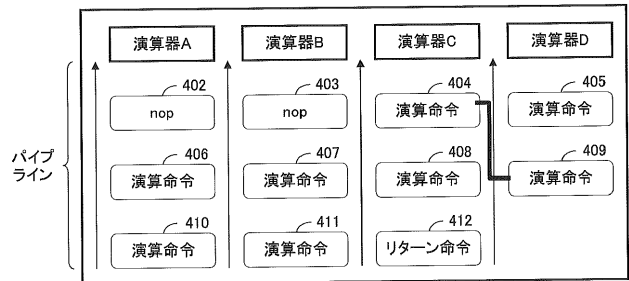
【図 5 A】

試験開始アドレスとして
メモリアドレス401を指定した試験の例



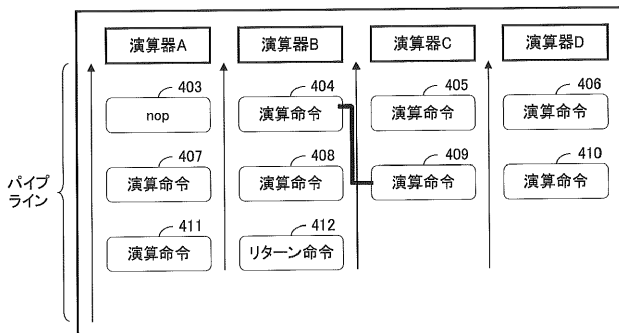
【図 5 B】

試験開始アドレスとして
メモリアドレス402を指定した試験の例



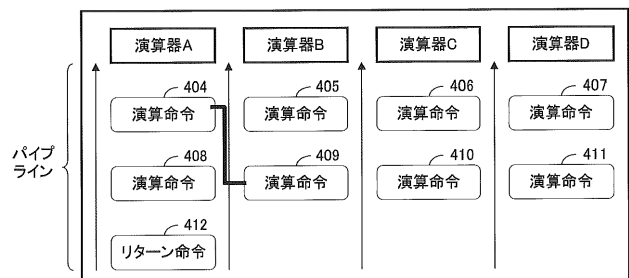
【図 5 C】

試験開始アドレスとして
メモリアドレス403を指定した試験の例



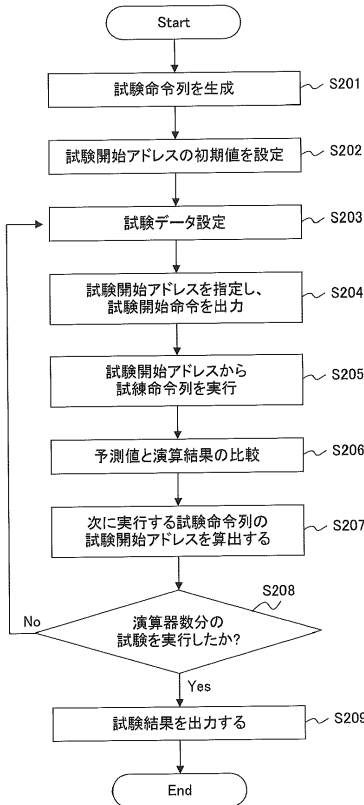
【図 5 D】

試験開始アドレスとして
メモリアドレス404を指定した試験の例



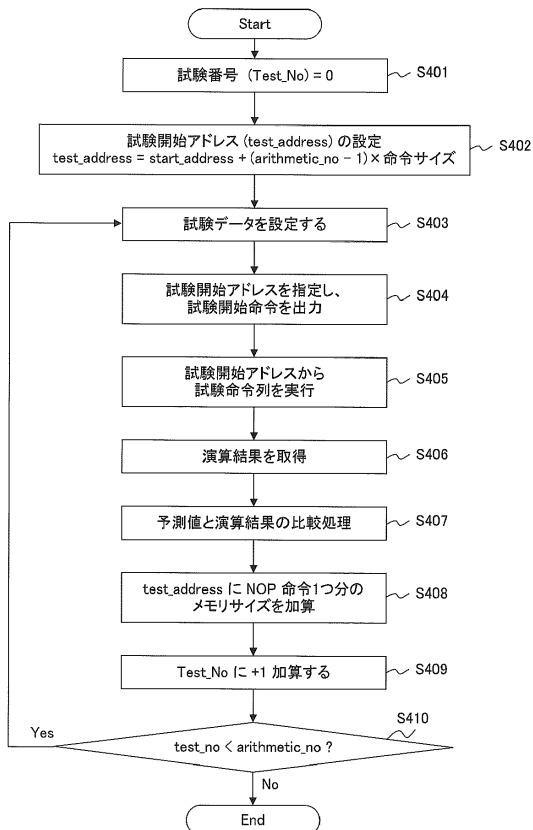
【図 6】

試験命令列を使用した第2の試験で行われる
処理の例を説明するフローチャート



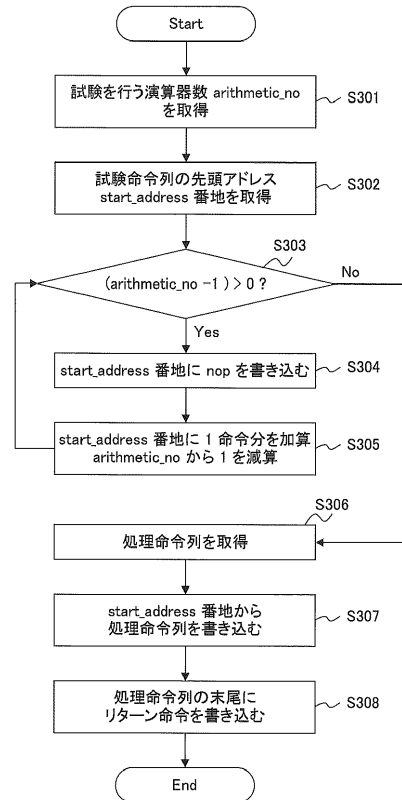
【図 8】

試験制御方法



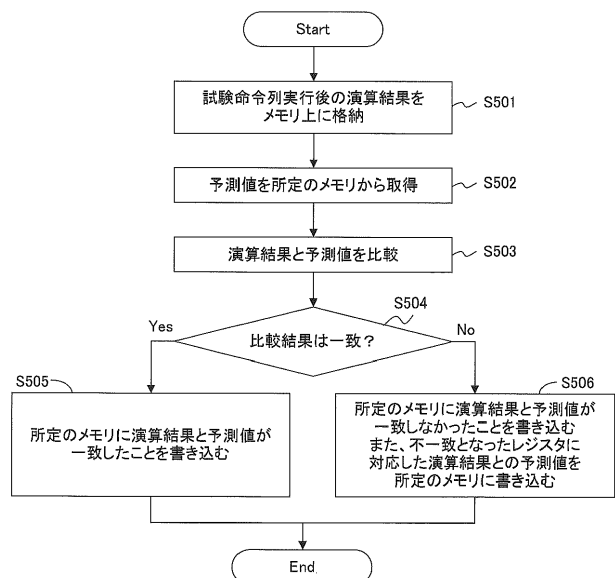
【図 7】

試験命令列の生成方法



【図 9】

予測値と演算結果の比較処理



【図 10】

試験結果データの例

Test_No	0	1	2	3
試験結果	OK	NG	OK	OK
レジスタ1	OK	OK	OK	OK
	-	-	-	-
	-	-	-	-
レジスタ2	OK	OK	OK	OK
	-	-	-	-
	-	-	-	-
レジスタ3	OK	NG	OK	OK
	-	予測値	-	-
	-	演算結果	-	-
レジスタ4	OK	OK	OK	OK
	-	-	-	-
	-	-	-	-
...				
...				
...				

フロントページの続き

(72)発明者 寺西 信輔

神奈川県川崎市中原区上小田中4丁目1番1号 株式会社富士通コンピュータテクノロジーズ内

Fターム(参考) 5B048 AA03 DD03 FF02