(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2018年5月24日(24.05.2018)

WIPO | PCT

(10) 国際公開番号
WO 2018/092237 A1

(51) 国際特許分類:
G06F 9/44 (2006.01) *G06F 11/36* (2006.01)

(21) 国際出願番号： PCT/JP2016/084077

(22) 国際出願日： 2016年11月17日(17.11.2016)

(25) 国際出願の言語： 日本語

(26) 国際公開の言語： 日本語

(71) 出願人:三菱電機株式会社(**MITSUBISHI ELECTRIC CORPORATION**) [JP/JP]; 〒1008310 東京都千代田区丸の内二丁目7番3号 Tokyo (JP).

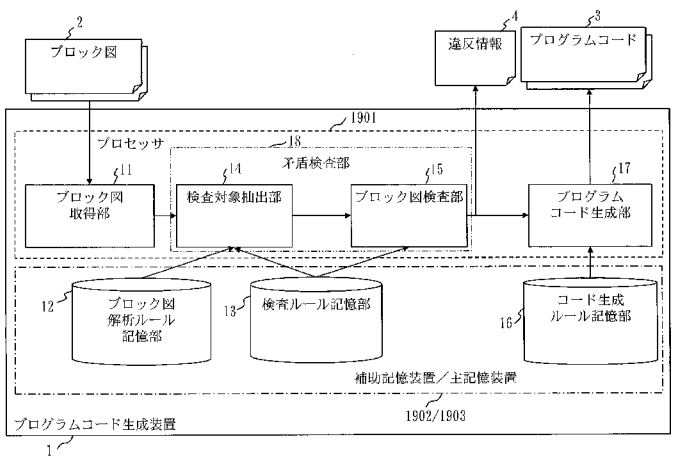
(72) 発明者:植田 武(**UEDA, Takeshi**); 〒1008310 東京都千代田区丸の内二丁目7番3号 三菱電機株式会社内 Tokyo (JP). 小林 信博(**KOBAYASHI, Nobuhiro**); 〒1008310 東京都千代田区丸の内二丁目7番3号 三菱電機株式会社内 Tokyo (JP).

(74) 代理人: 溝井 章司, 外 (**MIZOI, Shoji** et al.); 〒2470056 神奈川県鎌倉市大船二丁目17番10号 NTA大船ビル3階 溝井国際特許事務所 Kanagawa (JP).

(81) 指定国(表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS,

(54) **Title:** PROGRAM CODE GENERATION APPARATUS, PROGRAM CODE GENERATION METHOD, AND PROGRAM CODE GENERATION PROGRAM

(54) 発明の名称： プログラムコード生成装置、プログラムコード生成方法及びプログラムコード生成プログラム



1 Program code generation apparatus
2 Block diagram
3 Program code
4 Violation information
11 Block diagram acquisition unit
12 Block diagram analysis rule storage unit
13 Check rule storage unit
14 Check target extraction unit
15 Block diagram check unit
16 Code generation rule storage unit
17 Program code generation unit
18 Inconsistency check unit
1901 Processor
1902/1903 Auxiliary storage device/main storage device

(57) **Abstract:** A block diagram acquisition unit (11) acquires a block diagram (2) that defines, by connecting a plurality of blocks to one another, a data processing procedure to be reflected in a program code. By tracing the connection among the blocks in the block diagram (2), an inconsistency check unit (18) checks whether or not the data processing procedure defined by the block diagram (2) includes an inconsistency.

WO 2018/092237 A1

[続葉有]

RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) 指定国(表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

添付公開書類:

— 国際調査報告(条約第21条(3))

(57) 要約: ブロック図取得部(11)は、プログラムコードに反映させるデータ処理手順を複数のブロックの接続により定義するブロック図(2)を取得する。矛盾検査部(18)は、ブロック図(2)でのブロック間の接続を辿って、ブロック図(2)で定義されているデータ処理手順に矛盾があるかどうかを検査する。

明　細　書

発明の名称 ：

プログラムコード生成装置、プログラムコード生成方法及びプログラムコード生成プログラム

技術分野

[0001] 本発明は、プログラムコード生成装置、プログラムコード生成方法及びプログラムコード生成プログラムに関する。

背景技術

[0002] 事前に生成したプログラム仕様に基づいて、ソースコード等のプログラムコードを自動生成する方法がある。例えば、ブロック図からプログラムコードを生成する方法がある。ブロック図は、複数のブロックと、ブロック間を接続する接続線とで構成される。各ブロックには、データ処理手順の要素が定義されている。具体的には、各ブロックには、ひとまとまりのデータ処理が定義されている。接続線で複数のブロックを接続することで、ブロック図において、プログラムが入力値を取得してから演算結果が示される出力値を出力するまでのデータ処理手順の流れが記述される。

ブロック図からプログラムコードを生成する技術として、例えば、特許文献１に開示の技術がある。

先行技術文献

特許文献

[0003] 特許文献1：特開２０１１－１３８８７号公報

特許文献2：特開２０１６－５７７１５号公報

発明の概要

発明が解決しようとする課題

[0004] 特許文献１及び２では、生成されたブロック図と、不適切なブロック図のパターンとを比較し、生成されたブロック図が不適切なブロック図のパター

ンと一致する場合に、生成されたブロック図が不適切であることを通知する。

このように、特許文献１及び２の技術では、不適切なブロック図を検出するために、不適切なブロック図のパターンを予め設けておく必要があるという課題がある。

[0005] 本発明は、このような課題を解決することを主な目的とする。つまり、本発明は、不適切なブロック図のパターンを予め設けておくことなく、不適切なブロック図を検出できる構成を得ることを主な目的とする。

課題を解決するための手段

[0006] 本発明に係るプログラムコード生成装置は、

プログラムコードに反映させるデータ処理手順を複数のブロックの接続により定義するブロック図を取得するブロック図取得部と、

前記ブロック図でのブロック間の接続を辿って、前記ブロック図で定義されているデータ処理手順に矛盾があるかどうかを検査する矛盾検査部とを有する。

発明の効果

[0007] 本発明では、ブロック図でのブロック間の接続を辿ってデータ処理手順に矛盾があるかどうかを検査する。このため、本発明によれば、不適切なブロック図のパターンを予め設けておくことなく、不適切なブロック図を検出することができる。

図面の簡単な説明

[0008] [図1]実施の形態１に係るプログラムコード生成装置の機能構成例を示す図。

[図2]実施の形態１に係るプログラムコード生成装置の動作例を示すフローチャート。

[図3]実施の形態１に係るブロック図の例を示す図。

[図4]実施の形態１に係るＳｕｂＡの例を示す図。

[図5]実施の形態１に係るＳｕｂＢの例を示す図。

[図6]実施の形態１に係る解析ルールの例を示す図。

[図7]実施の形態１に係る検査ルールの例を示す図。

[図8]実施の形態１に係るコード生成ルールの例を示す図。

[図9]実施の形態１に係るコード生成ルールの例を示す図。

[図10]実施の形態１に係るプログラムコードの例を示す図。

[図11]実施の形態１に係るプログラムコードの例を示す図。

[図12]実施の形態１に係る違反情報の例を示す図。

[図13]実施の形態１に係る検査対象抽出部の動作例を示すフローチャート。

[図14]実施の形態１に係るブロック図検査部の動作例を示すフローチャート。

[図15]実施の形態１に係るプログラムコード生成部の動作例を示すフローチャート。

[図16]実施の形態１に係るプログラムコード生成装置のハードウェア構成例を示す図。

発明を実施するための形態

[0009] 以下、本発明の実施の形態について、図を用いて説明する。以下の実施の形態の説明及び図面において、同一の符号を付したものは、同一の部分または相当する部分を示す。

[0010] 実施の形態１.

＊＊＊構成の説明＊＊＊

図１は、本実施の形態に係るプログラムコード生成装置１の機能構成例を示す。

また、図１６は、本実施の形態に係るプログラムコード生成装置１のハードウェア構成例を示す。

先ず、図１６を参照してプログラムコード生成装置１のハードウェア構成例を説明し、その後に、図１を参照してプログラムコード生成装置１の機能構成例を説明する。

なお、プログラムコード生成装置１で行われる動作は、プログラムコード生成方法及びプログラムコード生成プログラムに相当する。

[0011] プログラムコード生成装置１は、コンピュータである。

プログラムコード生成装置１は、ハードウェアとして、プロセッサ１９０１、補助記憶装置１９０２、主記憶装置１９０３、通信装置１９０４及び入出力装置１９０５を備える。

補助記憶装置１９０２には、図１に示すブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコード生成部１７の機能を実現するプログラムが記憶されている。

そして、これらプログラムが補助記憶装置１９０２から主記憶装置１９０３にロードされ、プロセッサ１９０１がこれらプログラムを実行して、後述するブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコード生成部１７の動作を行う。

図１では、プロセッサ１９０１がブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコード生成部１７の機能を実現するプログラムを実行している状態を模式的に表している。

また、補助記憶装置１９０２は、図１に示すブロック図解析ルール記憶部１２、検査ルール記憶部１３及びコード生成ルール記憶部１６を実現する。なお、ブロック図解析ルール記憶部１２、検査ルール記憶部１３及びコード生成ルール記憶部１６の少なくともいずれかが主記憶装置１９０３により実現されてもよい。

通信装置１９０４は、プログラムコード生成装置１が外部装置と通信を行う場合に用いられる。プログラムコード生成装置１が外部装置との通信を行わないのであれば、通信装置１９０４は省略してもよい。

入出力装置１９０５は、プログラムコード生成装置１のユーザからの指示、ブロック図２等を取得する。また、入出力装置１９０５は、プログラムコード生成装置１にプログラムコード３又は違反情報４を提示する。

[0012] 次に、図１を参照して、プログラムコード生成装置１の機能構成例を説明する。

[0013] ブロック図取得部１１は、入出力装置１９０５を介して、ブロック図２

を取得する。

なお、ブロック図２とは、プログラムコードに反映させるデータ処理手順を複数のブロックの接続により定義するデータである。つまり、ブロック図２は、主に複数のブロックと各ブロックを接続する接続線から構成される。なお、ブロック図２の複数のブロックの各々にはデータ処理手順の要素である処理手順要素が割り当てられている。処理手順要素は、ひとまとまりのデータ処理である。

ブロック図取得部１１は、複数のブロック図２を取得することができる。

ブロック図取得部１１により行われる動作は、ブロック図取得処理に相当する。

[0014] ブロック図解析ルール記憶部１２は、解析ルールを記憶する。解析ルールは、検査対象抽出部１４がブロック図の構造を解析する際に用いられる。

なお、本実施の形態では、ブロック図解析ルール記憶部１２が解析ルールを保持することとしているが、ブロック図２に解析ルールが含まれていてもよい。

[0015] 検査ルール記憶部１３は、検査ルールを記憶する。検査ルールは、ブロック図検査部１５がブロック図２で定義されているデータ処理手順に矛盾があるかどうかを検査する際に用いられる。

検査ルール記憶部１３は、検査ルールを複数記憶してもよい。

[0016] 検査対象抽出部１４は、解析ルール及び検査ルールに基づいて、ブロック図２でのブロック間の接続を辿って、ブロック図２から検査対象となるブロック及び接続線を抽出する。

より具体的には、検査対象抽出部１４は、複数のブロックのうちのいずれかのブロックを起点ブロックとして設定する、そして、検査対象抽出部１４は、起点ブロックからブロック図２におけるブロック間の接続を辿って、起点ブロックが依存しているブロックを依存先ブロックとして抽出する。更に、検査対象抽出部１４は、抽出した依存先ブロックが依存しているブロック

を依存先ブロックとして抽出する動作を繰り返し、起点ブロックが直接的及び間接的に依存している依存先ブロックを抽出する。

[0017] ブロック図検査部１５は、検査対象抽出部１４により抽出されたブロック及び接続線に対して、検査ルールを適用して、ブロック図２で定義されているデータ処理手順に矛盾があるかどうかを検査する。つまり、ブロック図検査部１５は、ブロック図２に検査ルールに違反している箇所がないかどうかを検査する。

例えば、ブロック図検査部１５は、起点ブロックに割り当てられている処理手順要素と依存先ブロックに割り当てられている処理手順要素との間にデータタイプ、最大値及び最小値の少なくともいずれかにおいて矛盾があるかどうかを検査する。

また、ブロック図検査部１５は、ブロック図２で定義されているデータ処理手順に矛盾がある場合に、データ処理手順における矛盾を通知する違反情報４を入出力装置１９０５を介して出力する。

[0018] 検査対象抽出部１４とブロック図検査部１５をまとめて矛盾検査部１８という。

つまり、検査対象抽出部１４及びブロック図検査部１５は、矛盾検査部１８として、ブロック図２でのブロック間の接続を辿って、ブロック図２で定義されているデータ処理手順に矛盾があるかどうかを検査する。なお、矛盾検査部１８により行われる動作は、矛盾検査処理に相当する。

[0019] コード生成ルール記憶部１６は、コード生成ルールを記憶する。コード生成ルールは、プログラムコード生成部１７がブロック図２からプログラムコード３を生成する際に用いられる。

[0020] プログラムコード生成部１７は、コード生成ルールに基づき、ブロック図２からプログラムコード３を生成する。

つまり、プログラムコード生成部１７は、ブロック図２で定義されているデータ処理手順に矛盾がない場合に、ブロック図２から、ブロック図２で定義されているデータ処理手順が反映されたプログラムコードを生成する。プ

ログラムコードは、コンピュータ言語で記述されている。

[0021] ＊＊＊動作の説明＊＊＊

次に、図２のフローチャートを参照して、プログラムコード生成装置１の動作例を説明する。

[0022] 先ず、ブロック図取得部１１が、入出力装置１９０５を介して、ブロック図２を取得する（ステップＳ１）。

[0023] 次に、検査対象抽出部１４が、ブロック図２に、ブロック図解析ルール記憶部１２に記憶されている解析ルール、検査ルール記憶部１３に記憶されている検査ルールを適用して、ブロック図２から検査対象の情報を抽出する（ステップＳ２）。なお、ステップＳ２の詳細は後述する。

[0024] 次に、ブロック図検査部１５が、検査対象抽出部１４により抽出された検査対象に関する情報に、検査ルール記憶部１３に記憶されている検査ルールを適用して、ブロック図２で定義されているデータ処理手順に矛盾がないかどうかの検査を行う（ステップＳ３）。なお、ステップＳ３の詳細は後述する。

[0025] ステップＳ３の検査の結果、ブロック図２で定義されているデータ処理手順に矛盾がない場合（ステップＳ４でＹＥＳ）は、プラグラムコード生成部１７が、ブロック図２に、コード生成ルール記憶部１６に記憶されているコード生成ルールを適用して、プログラムコード３を生成する（ステップＳ５）。なお、ステップＳ５の詳細は後述する。

[0026] 一方、ブロック図２で定義されているデータ処理手順に矛盾がある場合（ステップＳ４でＮＯ）は、ブロック図検査部１５は、データ処理手順における矛盾を通知する違反情報４を入出力装置１９０５から出力する（ステップＳ６）。なお、違反情報４の形式については後述する。

[0027] このように、本実施の形態では、プログラムコード生成装置１は、ブロック図２が適切であるか否かを検査し、ブロック図２が適切であればプログラムコード３を生成する。

[0028] 次に、ブロック図２、ブロック図解析ルール記憶部１２で記憶されている

解析ルール、検査ルール記憶部１３で記憶されている検査ルール、コード生成ルール記憶部１６で記憶されているコード生成ルール、プログラムコード３及び違反情報４の詳細を説明する。

[0029] （ブロック図の例）

ブロック図２は、画像ファイル等のブロック図の外観のみを表す情報のみではなく、開発ツール上でプログラムをブロック図として描画するために必要となる情報や、シミュレーションやコード自動生成を行うための付加的な情報が含まれているファイルである。ブロック図２は、例えば、ＸＭＬ（Ｅｘｔｅｎｓｉｂｌｅ Ｍａｒｋｕｐ Ｌａｎｇｕａｇｅ）やＪＳＯＮ（ＪａｖａＳｃｒｉｐｔ（登録商標） Ｏｂｊｅｃｔ Ｎｏｔａｔｉｏｎ）等の階層的なデータ構造を表現可能なデータを含むファイルとして実現される。ただし、以降では、理解を容易にするために、ブロック図２は、ＸＭＬやＪＳＯＮの形式ではなく、複数のブロックが接続線で接続された形式で表現する。そして、必要に応じて吹き出し等で付加情報を表す。

[0030] 図３～図５は、ブロック図２の例を示す。

図３は、プログラムコードに反映させるデータ処理手順が定義されるブロック図２を示す。図４は、図３のブロック図２に含まれるサブルーチンであるＳｕｂＡの詳細を示す。また、図５は、図３のブロック図２に含まれるサブルーチンであるＳｕｂＢの詳細を示す。

[0031] 図３において、符号Ｂ１～Ｂ９が付されている要素がブロックである。また、符号Ｌ１２、Ｌ２３、Ｌ２４、Ｌ５３、Ｌ６３、Ｌ７４、Ｌ８４、Ｌ３４９が付されている要素が接続線である。

また、図４において、符号Ｂ３１～Ｂ３４が付されている要素がブロックである。また、符号Ｌ３１３、Ｌ３２３、Ｌ３３４が付されている要素が接続線である。

また、図５において、符号Ｂ４１～Ｂ４４が付されている要素がブロックである。また、符号Ｌ４１３、Ｌ４２３、Ｌ４３４が付されている要素が接続線である。

以下、図３のブロック及び接続線について説明するが、以下の説明は図４及び図５のブロック及び接続線にも適用される。

[0032] 各ブロックには内部データが設定されている。内部データには、例えば、ブロックのＩＤ（Ｉｄｅｎｔｉｆｉｅｒ）、ブロックの名前、ブロックの種別が含まれる。なお、各ブロックに設定されている内部データに含まれる内容が処理手順要素である。

例えば、ブロックＢ１のＩＤは１であり、ブロックＢ１の名前はＩｎ１であり、ブロックＢ１の種別はＩｎｐｕｔである。ＩＤは、各ブロックをユニークに識別するための番号や文字列である。本実施の形態では、ＩＤはユニークな番号とする。名前は、ブロック図２をディスプレイに表示した際にプログラムコード生成装置１のユーザが各ブロックを識別しやすくするために任意につけるものである。ブロックの種別は、各ブロックに割り当てられているデータ処理の特性を識別するために用いられる。例えば、ブロックの種別として、入力、出力、加算、ｉｆ分岐、サブシステム等がある。ブロックの種別を表す識別子として、複数の番号や文字列が予め定義されている。

[0033] 例えば、ブロックＢ１には種別として「Ｉｎｐｕｔ」が設定されており、ブロックＢ１には処理手順要素として入力処理が割り当てられている。以下、入力処理が割り当てられているブロックを入力ブロックという。

また、ブロックＢ９には種別として「Ｏｕｔｐｕｔ」が設定されており、ブロックＢ９には処理手順要素として出力処理が割り当てられている。以下、出力処理が割り当てられているブロックを出力ブロックという。

また、以下では、処理手順要素として加算演算処理が割り当てられているブロック（種別が「Ｓｕｍ」のブロック）をＳｕｍブロックという。なお、Ｓｕｍブロックは図３には示されていない。

また、処理手順要素としてｉｆ分岐処理が割り当てられているブロック（種別が「Ｉｆ」のブロック）をｉｆブロックという。

また、処理手順要素としてサブシステムが割り当てられているブロック（種別が「Ｓｕｂｓｙｓｔｅｍ」のブロック）をＳｕｂｓｙｓｔｅｍブロック

という。

[0034] また、図３では、図示していないが、内部データには、最終的にブロック図２からプログラムコード３を生成するために必要となる属性情報も含まれる。例えば、入力ブロックの内部データには、属性情報として、入力データのデータ型、入力データの最大値又は／及び最小値等が含まれる。また、ｉｆブロック等の制御構造を表すブロックでは、内部データに、その制御を行うための条件等や、条件に合致した場合の処理先等に関する情報が属性情報として含まれる。

[0035] 接続線は、任意の２つのブロック間を接続する線である。接続線は、処理やデータの流れを示すための有向線である。接続線には、内部データとして、接続線が接続する２つのブロックのＩＤと、処理又はデータの流れる方向に関する情報が設定されている。接続線の内部データには、接続元のブロックのＩＤと接続先のブロックのＩＤが含まれる。また、接続元のブロックに複数の出力ポートがある場合、又は接続先のブロックに複数の入力ポートが存在する場合は、接続線の内部データには、ブロックのポートのＩＤも含まれる。例えば、接続線Ｌ１２の内部データには、「接続元：１＃ｏｕｔ：１」と「接続先：２＃ｉｎ：１」と記述されている。これは、接続線Ｌ１２は、ブロックＢ１の出力ポート１とブロックＢ２の入力ポート１とを接続していることを意味する。

また、接続線の内部データには、接続元のブロックのＩＤと接続先のブロックのＩＤとが記述されているため、ブロック図検査部１５は、ブロック間のデータの流れや処理の流れも特定することができる。

例えば、ブロック図検査部１５は、接続線Ｌ１２の内部データを参照することで、入力ブロックＢ１から出力されたデータが、ＩｆブロックＢ２に入力されたことを特定することができる。

このように、ブロック図検査部１５は、ブロック図２に含まれる接続線の内部データを参照することで、ブロック図２内のブロック間の接続関係を特定することができる。

ブロック図検査部１５は、ブロック間の接続関係、各ブロックに割り当てられたデータ処理を解析して、ブロック図２で定義されているデータ処理手順に矛盾がないかどうかを検査する。

なお、接続線は接続元のブロックのＩＤと接続先のブロックのＩＤで識別可能であるため、本実施の形態では、接続線にはＩＤが設定されていない。接続線にＩＤを設定するようにしてもよい。

[0036]（解析ルールの例）

次に、ブロック図解析ルール記憶部１２に記憶される解析ルールを説明する。

図６は、解析ルールの例を示す。

[0037] 図６の解析ルール６００では、ブロックの種別ごとに、ブロックから得られる情報が定義される。

例えば図６の解析ルール６００には、種別６０１、データ入力数６０２、データ出力数６０３、制御入力数６０４、制御出力数６０５、属性６０６が含まれる。

種別６０１は、ブロックの種別を表す。

データ入力数６０２は、ブロックに入力されるデータの数を表す。データ入力数６０２に「１以上」と記載されている場合は、ブロックに入力されるデータの数は可変である。

データ出力数６０３は、ブロックから出力されるデータの数を表す。

制御入力数６０４は、処理の制御に関する入力数を表す。例えば、Ｓｕｂｓｙｓｔｅｍ６１４では制御入力数６０４が「１以上」と定義されている。これは、Ｉｆブロックでの制御結果がＳｕｂｓｙｓｔｅｍブロックに１以上入力されるということを意味する。なお、「１以上」と記載されている場合は、ブロックに入力される制御結果の数は可変である。

制御出力６０５は、処理の制御に関する出力数を表す。例えば、Ｉｆ６１３では制御出力数６０５が「２」と定義されている。これは、Ｉｆブロックでの２つの制御結果が出力されることを意味する。

属性６０６は、ブロックがサポートしている属性を示す。

[0038] なお、図６では、説明の簡明化のために自然言語で記述された解析ルール６００を示したが、解析ルール６００をプログラム言語で記述してもよい。

[0039] （検査ルールの例）

次に、検査ルール記憶部１３に記憶される検査ルールを説明する。

図７は、検査ルールの例を示す。

[0040] 図７の検査ルール７００には、ブロック図２に定義されているデータ処理手順における矛盾を検出するためのルールが記述されている。

例えば図７の検査ルール７００には、対象７０１、起点７０２、終点７０３、検査項目７０４が含まれる。

対象７０１は、検査対象の種類を表す。

Ｂｌｏｃｋ７１１は、各ブロックに対する検査項目を表す。

ＤａｔａＳｌｉｃｅ７１２は、データスライスに対する検査項目を表す。

Ｓｌｉｃｅ７１３は、スライスに対する検査項目を表す。

[0041] データスライスは、制御依存関係を考慮せずにデータ依存関係のみを考慮して抽出したスライスを指す。なお、本実施の形態では、あるブロック（ブロックαとする）に割り当てられているデータ処理の実行結果が他のブロック（ブロックβとする）に割り当てられているデータ処理の実行に影響を及ぼす場合に、ブロックβはブロックαと「制御依存関係」があるという。ブロックαはブロックβの依存先ブロックである。例えば、図３において、ブロックＢ３はブロックＢ２と制御依存関係がある。つまり、ブロックＢ２はブロックＢ３の依存先ブロックである。

また、「データ依存関係」とは、あるブロック（ブロックαとする）から出力されたデータが他のブロック（ブロックβとする）に到達することである。例えば、図３において、ブロックＢ１から出力されたデータはブロックＢ２に入力されるため、ブロックＢ２はブロックＢ１とデータ依存関係がある。つまり、ブロックＢ１はブロックＢ２の依存先ブロックである。

「スライス」とは、あるブロックと制御依存関係及びデータ依存関係があ

るブロックの集合（及びその接続関係）である。つまり、スライスは、あるブロックから、当該ブロックと制御依存関係があるブロックを辿っていって得られるブロックの集合（及びその接続関係）と、当該ブロックとデータ依存関係があるブロックを辿っていって得られるブロックの集合（及びその接続関係）である。換言すれば、スライスは当該ブロックと関係する全てのブロックの集合（及びその接続関係）である。

「制御依存関係を考慮せずにデータ依存関係のみを考慮して抽出したスライス」とは、制御依存関係を除いて構築できるスライスのことである。

つまり、データスライスは、あるブロックから、当該ブロックとデータ依存関係があるブロックのみを辿っていって得られたブロックの集合（及びその接続関係）である。

[0042] 起点７０２は、制御依存関係又はデータ依存関係を探索していく際に起点となるブロックの種別を表す。図７では、ＤａｔａＳｌｉｃｅ７１２及びＳｌｉｃｅ７１３に対して、起点７０２として「Ｏｕｔｐｕｔ」が記載されている。このため、ブロック図検査部１５は、種別が「Ｏｕｔｐｕｔ」のブロックから、種別が「Ｏｕｔｐｕｔ」のブロックとデータ依存関係にあるブロックを辿ってデータスライスを求める。また、ブロック図検査部１５は、種別が「Ｏｕｔｐｕｔ」のブロックから、種別が「Ｏｕｔｐｕｔ」のブロックとデータ依存関係及び制御依存関係にあるブロックを辿ってスライスを求める。

[0043] また、終点７０３は、制御依存関係又はデータ依存関係の探索を終了するブロックの種別を表す。ブロック図検査部１５は、通常は、制御依存関係又はデータ依存関係が探索できなくなるまで、つまり、入力ブロックにたどり着くまで制御依存関係又はデータ依存関係の探索を続ける。しかしながら、ブロック図検査部１５の探索を特定の種別のブロックまでで終える場合には終点７０３の欄に該当する種別が定義される。例えば、出力ブロックからＩｆブロックまでのスライスを取得する場合は、終点７０３に「Ｉｆ」が設定される。

[0044] 検査項目７０４は、抽出された検査対象に対する検査内容を示す。

例えば、Ｂｌｏｃｋ７１１では、ＤａｔａＴｙｐｅが設定できるブロックにＤａｔａＴｙｐｅが設定されていない場合に、ブロック図検査部１５はブロック図２に記載のデータ処理手順に矛盾があると判定する。

また、ＤａｔａＳｌｉｃｅ７１２では、ブロック図検査部１５はデータスライスのパスに含まれるいずれかのブロックのＤａｔａＴｙｐｅが起点のブロックのＤａｔａＴｙｐｅと異なる場合に、ブロック図検査部１５は、ブロック図２に記載のデータ処理手順に矛盾があると判定する。

また、Ｓｌｉｃｅ７１３では、１つのスライスに複数のパスがある場合に、パス間でＭｉｎ（最小値）及びＭａｘ（最大値）のいずれかが異なる場合には、ブロック図検査部１５は、ブロック図２に記載のデータ処理手順に矛盾があると判定する。

[0045] なお、図７では、説明の簡明化のために自然言語で記述された検査ルール７００を示したが、検査ルール７００をプログラム言語で記述してもよい。

[0046] （コード生成ルール）

次に、コード生成ルール記憶部１６に記憶されるコード生成ルールを説明する。

図８及び図９は、コード生成ルールの例を示す。

[0047] 図８及び図９のコード生成ルール８００、９００には、ブロック図２に含まれる情報に基づいて、プログラムコード３を生成するために必要な情報が記載される。本実施の形態では、コード生成ルール８００とコード生成ルール９００という、役割が異なる２つのコード生成ルールが存在している。より具体的には、コード生成ルール８００は、ブロック図２の表現には関わらない一般的なコード生成ルールである。コード生成ルール９００は、ブロック図２の表現に関わるコード生成ルールである。

図８のコード生成ルール８００では、プログラムコードに関する一般的な項目８０１とその変換ルール８０２の対が記述されている。また、コード生成ルール８００では、例えば「関数名」という項目については、変換ルール

８０２に拡張子は除いたブロック図のファイル名を利用することが記述されている。

また、図９のコード生成ルール９００では、各ブロックの種別を表すブロック種別９０１とその変換ルール９０２の対が記述されている。また、コード生成ルール９００では、ブロック種別６１１が「Ｉｎｐｕｔ」であるブロックに対する変換ルールとして、言語上の種別はＩＮ構造体のメンバで、メンバ名はｎａｍｅ属性の値であり、データ型はｔｙｐｅ属性の値とすることが記述されている。

このようなコード生成ルール８００、９００に基づいて、プログラムコード生成部１７はブロック図２からプログラムコード３を生成する。

[0048] なお、図８及び図９では、説明の簡明化のために自然言語で記述されたコード生成ルール８００、９００を示したが、コード生成ルール８００、９００をプログラム言語で記述してもよい。

[0049] （プログラムコードの例）

次に、プログラムコード生成部１７で生成されるプログラムコード３を説明する。

図１０及び図１１は、プログラムコード３の例を示す。

[0050] 図１０に示すプログラムコードをプログラムコード１０００といい、図１１に示すプログラムコードをプログラムコード１１００という。プログラムコード１０００は、ヘッダファイルであり、プログラムコード１１００は、ソースファイルである。

プログラムコード１０００及びプログラムコード１１００は、ともにＣ言語で記述されている。図１０のプログラムコード１０００では、すべての入力を扱うＩＮ構造体が定義されている（図１０の符号１００１）。また、出力を表すＯＵＴ構造体も定義されている（図１０の符号１００２）。また、関数ｓａｍｐｌｅのプロトタイプ宣言も定義されている（図１０の符号１００３）。また、プログラムコード１１００では、ブロック図２から生成されたコードが記述されている。

[0051] （違反情報の例）

次に、ブロック図検査部１５が出力する違反情報４を説明する。

図１２は、違反情報４の例を示す。

[0052] 違反情報４には、データ処理手順における矛盾、すなわち違反が発生しているブロック図２内の箇所を通知するための情報が含まれる。

例えば、図１２に示すように、違反情報４には、検査ルール１２０１、違反ブロック１２０２、関連スライス１２０３、関連パス１２０４が存在する。

検査ルール１２０１には、検査ルール７００が複数存在する場合に、違反を検知した際に用いられた検査ルール７００の番号が示される。

違反ブロック１２０２には、違反が含まれるブロックのＩＤが示される。

関連スライス１２０３には、検査ルール７００に含まれるＳｌｉｃｅ７１３の検査において違反が検知された場合に、違反が検知されたスライスに含まれるブロックのＩＤが列挙される。

また、関連パス１２０４には、検査ルール７００に含まれるＳｌｉｃｅ７１３の検査において違反が検知された場合に、スライスに含まれるパスが示される。１つの｛｝に囲まれる番号が１つのパスを意味する。

なお、違反情報４は、図１２に示す形式とは異なり、検査ルール７００の検査項目７０４ごとに、違反なし、違反ありを一覧表示するようにしてもよい。例えば、違反のない検査項目７０４には「ＹＥＳ」、違反のある検査項目７０４には「ＮＯ」を表示するようにしてもよい。

また、プログラムコード生成装置１のユーザにより分かりやすく違反内容を通知するために、違反情報４に、違反内容を説明する説明文を付加してもよい。

[0053] 次に、図２に示したステップＳ２、Ｓ３、Ｓ５の詳細を説明する。

最初に、図１３を参照して、ステップＳ２の詳細を説明する。

[0054] まず、検査対象抽出部１４が、ブロック図取得部１１からブロック図２を取得する（ステップＳ２１）。

[0055] 次に、検査対象抽出部１４は、ブロック図解析ルール記憶部１２から解析ルールを読み込む（ステップＳ２２）。

[0056] 次に、検査対象抽出部１４は、検査ルール記憶部１３から検査ルールを読み込む（ステップＳ２３）。

[0057] 次に、検査対象抽出部１４は、ステップＳ２３で読み込んだ検査ルールを用いてブロック図２から検査対象を抽出する（ステップＳ２４）。ステップＳ２３で複数の検査ルールを読み込んでいる場合は、検査対象抽出部１４は、検査ルールごとにステップＳ２４を行う。

図７の検査ルール７００を用いて検査対象を抽出する場合は、検査対象抽出部１４は、例えば以下のように動作する。

[0058] 図７のＢｌｏｃｋ７１１では、起点７０３が「任意」であるため、検査対象抽出部１４は、すべてのブロックを検査対象として抽出する。ただし、検査対象抽出部１４は、ブロック間の関係性は解析しない。

[0059] また、ＤａｔａＳｌｉｃｅ７１２では、起点７０３が「Ｏｕｔｐｕｔ」で、終点７０４が「任意」であるため、ブロックの種別が「Ｏｕｔｐｕｔ」であるブロックを起点に、検査対象抽出部１４は、データ依存関係のみを考慮してスライスを抽出する。

つまり、検査対象抽出部１４は、ブロック種別が「Ｏｕｔｐｕｔ」のブロックに割り当てられているデータ処理で扱われるデータに影響を与えるブロックの集合を抽出する。例えば、図３のブロックＢ９とデータ依存関係のあるブロックとして、検査対象抽出部１４は、ブロックＢ３とブロックＢ４を抽出する。つまり、ブロックＢ３とブロックＢ４は、ブロックＢ９の依存先ブロックである。

[0060] より具体的には、検査対象抽出部１４は、図６の解析ルールを参照して、ブロックＢ３とブロックＢ４を抽出する。まず、検査対象抽出部１４は、図６で種別６０１が「Ｏｕｔｐｕｔ」の行６１２を参照する。次に、検査対象抽出部１４は、行６１２のデータ入力数６０２を参照する。図６では「１以上」となっているので、種別が「Ｏｕｔｐｕｔ」のブロックは他のブロック

から出力データに対する影響を受ける。このため、種別が「Ｏｕｔｐｕｔ」のブロックと接続線で接続されているブロックは、種別が「Ｏｕｔｐｕｔ」のブロックとデータ依存関係にある。

このため、図３では、ブロックＢ９と接続線で接続されているブロックＢ３とブロックＢ４が、ブロックＢ９とデータ依存関係のあるブロックとして抽出される。つまり、ブロックＢ３とブロックＢ４は、ブロックＢ９の依存先ブロックである。

[0061] 次に、検査対象抽出部１４は、ブロックＢ３とブロックＢ４のそれぞれに対するデータ依存関係がないかを調べる。

ブロックＢ３は、種別がＳｕｂｓｙｓｔｅｍなので、検査対象抽出部１４は、図６の行６１４を参照する。行６１４では、データ入力数６０２が「１以上」なので、ブロックＢ３は、ブロックＢ３にデータを入力しているブロックとデータ依存関係があることが分かる。但し、Ｓｕｂｓｙｓｔｅｍの場合には、実体は別のブロック図にあるので、別のブロック図を確認する必要がある。ブロックＢ３の場合は、図４のＳｕｂＡに実体が示される。検査対象抽出部１４は、図４に対しても同様の解析を行う。

図４のブロックＢ３４と図３のブロックＢ９はＩＤが同じであるため、同じブロックである。図４においてブロックＢ３４はブロックＢ３３に接続している。

ブロックＢ３３は、種別が「Ｓｕｍ」である。図６の解析ルール６００の行６１５では、データ出力数６０３が「１」である。このため、ブロックＢ３３は、ブロックＢ３４とデータ依存関係がある。つまり、ブロックＢ３３はブロックＢ３４（ブロックＢ９）の依存先ブロックである。

また、ブロックＢ３３は、ブロックＢ３１とブロックＢ３２に接続している。ブロックＢ３１とブロックＢ３２はそれぞれ種別が「Ｉｎｐｕｔ」であり、図６の解析ルール６００の行６１１では、データ出力数６０３が「１」である。このため、ブロックＢ３１及びブロックＢ３２は、ブロックＢ３３とデータ依存関係がある。つまり、ブロックＢ３１とブロックＢ３２は、ブ

ロックＢ３３の依存先ブロックである。

以上で、図３のブロックＢ３の内部構成である図３のブロックＢ３１～Ｂ３４が解析できたため、検査対象抽出部１４は、図３に戻って解析を続ける。

[0062] 図３のブロックＢ３と接続されているブロックとしては、ブロックＢ２、Ｂ５、Ｂ６の３つがある。ブロックＢ２の種別は「Ｉｆ」であり、図６の解析ルール６００の行６１３では、データ出力数６０３が「０」である。このため、ブロックＢ３はブロックＢ２とはデータ依存関係を持たない。つまり、ブロックＢ２はブロックＢ３の依存先ブロックではない。また、ブロックＢ５とブロックＢ６はそれぞれ種別が「Ｉｎｐｕｔ」であり、図６の解析ルール６００の行６１１では、データ出力数６０３が「１」である。このため、ブロックＢ５及びブロックＢ６は、ブロックＢ３とデータ依存関係がある。つまり、ブロックＢ５及びブロックＢ６は、ブロックＢ３の依存先ブロックである。なお、図３のＩｎ２と図４のＩｎ２はＩＤが同じであるため、同じブロックである。同様に、図３のＩｎ３と図４のＩｎ３はＩＤが同じであるため、同じブロックである。このため、検査対象抽出部１４は、抽出結果をマージする。

検査対象抽出部１４は、図３のブロックＢ９の他方の依存先ブロックであるブロックＢ４についても、同様の手順でデータ依存関係を解析する。

[0063] 検査対象抽出部１４は、結果的に図３のブロックＢ９のデータスライスとして、ブロックＢ９、ブロックＢ３、ブロックＢ５、ブロックＢ６、ブロックＢ７、ブロックＢ８、図４のブロックＢ３３及び図５のブロックＢ４３を抽出する。

なお、ブロックＢ９、ブロックＢ３、ブロックＢ５、ブロックＢ６、ブロックＢ３３の組と、ブロックＢ９、ブロックＢ４、ブロックＢ７、ブロックＢ８と図５のブロックＢ４３の組は、それぞれ影響を及ぼさないので、検査対象抽出部１４は、それぞれを別のパスとして管理する。

[0064] 次に、検査対象抽出部１４は、図７の検査ルール７００のＳｌｉｃｅ７１

３に従って、種別が「Ｏｕｔｐｕｔ」のブロックからスライスを抽出していく。

この場合には、検査対象抽出部１４は、データ依存関係に加えて、制御依存関係も考慮してブロックを抽出する。あるブロックに対する制御依存関係とは、あるブロックの実行に影響を与えるブロックの出力の集合のことである。

例えば、図３のブロックＢ３及びブロックＢ４と制御依存関係のあるブロックは、Ｉｆブロックである図３のブロックＢ２である。検査対象抽出部１４は、前述したデータスライスとして抽出したブロックの集合に、図３のブロックＢ２とブロックＢ１とを加えて、検査対象のスライスとする。

[0065] 以上の手順により、検査対象抽出部１４は、検査対象のブロック、検査対象のスライス及び検査対象のデータスライスを抽出することができる。

[0066] 次に、図１４を参照して、図２のステップＳ３の詳細を説明する。

[0067] まず、ブロック図検査部１５は、すべての検査ルールを未検査にマークする（ステップＳ４１）。

[0068] 次に、ブロック図検査部１５は、未検査の検査ルールがあるか否かを確認する（ステップＳ４２）。

[0069] 未検査のルールが存在する場合（ステップＳ４２でＹＥＳ）は、ブロック図検査部１５は、検査ルール記憶部１３から未検査の検査ルールを一つ読み込む（ステップＳ４３）。

[0070] 次に、ブロック図検査部１５は、読み込んだ検査ルールに従ってブロック図２を検査する（ステップＳ４４）。

具体的には、ブロック図検査部１５は、検査対象抽出部１４により抽出されたブロック、データスライス、スライスに対して、検査ルールを適用して、ブロック図２に定義されているデータ処理手順に矛盾がないかを検査する。

つまり、ブロック図検査部１５は、図７の検査ルール７００の行７１１に従い、検査対象抽出部１４により抽出された全てのブロックにおいてＤａｔ

ａＴｙｐｅが設定されていないブロックが存在するかどうかを検査する。

また、ブロック図検査部１５は、図７の検査ルール７００の行７１２に従い、検査対象抽出部１４により抽出されたデータスライスのパスごとに、起点ブロックのＤａｔａＴｙｐｅと異なるＤａｔａＴｙｐｅのブロックがパスに含まれていないかどうかを検査する。

また、ブロック図検査部１５は、図７の検査ルール７００の行７１３に従い、検査対象抽出部１４により抽出されたスライスのパスの間で、Ｍｉｎ又はＭａｘが異なるかどうかを検査する。

[0071] 次に、ブロック図検査部１５は、検査ルール違反が検知されたか否かを確認する（ステップＳ４５）。

[0072] 検査ルール違反が検知された場合（ステップＳ４５でＹＥＳ）は、ブロック図検査部１５は、検査ルール違反の詳細を違反検出情報として主記憶装置１９０３等に一時的に記憶させる（ステップＳ４６）。

例えば、ブロック図検査部１５は、違反が検知された検査ルールのＩＤ、違反が検知されたブロックのＩＤ、違反が検知されたスライス、違反が検知されたパス等を違反検出情報として主記憶装置１９０３等に記憶させる。

その後、ブロック図検査部１５は、検査が完了した検査ルールを検査済みとマークし（ステップＳ４７）、ステップＳ４２に戻る。

[0073] 違反が検知されることなく、一つの検査ルールに対する処理が完了した場合（ステップＳ４５でＮＯ）は、ブロック図検査部１５は、検査が完了した検査ルールを検査済みとマークし（ステップＳ４７）、ステップＳ４２に戻る。

[0074] ステップＳ４２で未検査の検査ルールが存在しないと判定した場合（ステップＳ４２でＮＯ）は、ブロック図検査部１５は、主記憶装置１９０３等に違反検出情報が存在しているかどうかを確認する（ステップＳ４８）。

[0075] 違反検出情報が存在する場合（ステップＳ４８でＹＥＳ）は、ブロック図検査部１５は、違反検出情報を違反情報４として入出力装置９０５のディスプレイ等に出力し、処理を終了する。

また、違反検出情報が存在しない場合（ステップＳ４８でＮＯ）は、ブロック図検査部１５は、プログラムコード生成部１７にプログラムコードの生成を指示する。

[0076] 次に、図１５を参照して、図２のステップＳ５の詳細を説明する。

[0077] まず、プログラムコード生成部１７は、コード生成ルール記憶部１６からコード生成ルールを読み込む（ステップＳ５１）。

[0078] 次に、プログラムコード生成部１７は、ブロック図検査部１５からブロック図２を取得する（ステップＳ５２）。

[0079] 次に、プログラムコード生成部１７は、コード生成ルールを使ってブロック図２からプログラムコード３を生成する（ステップＳ５３）。

[0080] ステップＳ５３では、例えば、読み込んだコード生成ルールが図８のコード生成ルール８００及び図９のコード生成ルール９００である場合には、プログラムコード生成部１７は、コード生成ルール８００に従って、ヘッダファイルとソースファイルを生成する。

ヘッダファイルのファイル名はブロック図２のファイル名に「．ｈ」を加えたものであり、ソースファイルのファイル名はブロック図２のファイル名に「．ｃ」を加えたものである。このため、ブロック図２のファイル名が「ｓａｍｐｌｅ．ｂｌｋ」であれは、プログラムコード生成部１７は、「ｓａｍｐｌｅ．ｈ」と「ｓａｍｐｌｅ．ｃ」というファイルを生成する。

この際に「ｓａｍｐｌｅ．ｃ」に含ませる関数の関数名は、「ｓａｍｐｌｅ」である。

次に、プログラムコード生成部１７は、ヘッダファイル「ｓａｍｐｌｅ．ｈ」の入力データの構造体と出力データの構造体を定義する。

プログラムコード生成部１７は、入力データの構造体（ＩＮ構造体）を、ブロックの種別が「Ｉｎｐｕｔ」のブロックの内部データに基づいて生成する。

また、プログラムコード生成部１７は、出力データの構造体（ＯＵＴ構造体）を、ブロックの種別が「Ｏｕｔｐｕｔ」のブロックの内部データに基づ

いて生成する。

プログラムコード生成部１７は、図３～図５のブロック図から、例えば、図１０に示すプログラムコード１０００（ヘッダファイル）を生成する。

また、プログラムコード生成部１７は、ソースファイル「ｓａｍｐｌｅ．ｃ」の内部を、図９のコード生成ルール９００に従って生成する。

プログラムコード生成部１７は、図３～図５のブロック図から、例えば、図１１に示すプログラムコード１１００（ソースファイル）を生成する。

[0081] ＊＊＊実施の形態の効果の説明＊＊＊

このように、本実施の形態では、ブロック図２でのブロック間の接続を辿ってデータ処理手順に矛盾があるかどうかを検査する。このため、本実施の形態によれば、不適切なブロック図のパターンを予め設けておくことなく、不適切なブロック図を検出することができる。

つまり、本実施の形態では、ブロック図２からプログラムコード３を生成する前に、ブロック図２から抽出したブロック同士の接続関係や設定された変数等の構造を比較するようにしているため、構造上矛盾する箇所を検出することができる。

このため、ソフトウェア開発者が修正の必要な箇所を探しだす必要がなくなり、ソフトウェア開発に要する労力及び時間を削減することができる。

[0082] 実施の形態２．

実施の形態１では、ブロック図２におけるデータ依存関係及び制御依存関係を抽出し、検査ルールに基づいてデータ処理手順における矛盾がないかどうかを検査する。

本実施の形態では、ブロック図検査部１５は、プログラムコード生成部１７がブロック図２からプログラムコード３を生成するために参照するコード生成ルール８００、９００を参照して、ブロック図２で定義されているデータ処理手順に矛盾があるかどうかを検査する。

本実施の形態でも、プログラムコード生成装置１の機能構成例は図１に示した通りである。また、プログラムコード生成装置１のハードウェア構成例

は図１６に示した通りである。

また、ブロック図検査部１５が検査ルール７００に加えてコード生成ルール８００、９００を参照する点を除けば、プログラムコード生成装置１の動作は、実施の形態１と同じである。

[0083] 本実施の形態によれば、生成対象のプログラムコードの言語仕様に依存した不具合を、プログラムコードを生成する前に検出することができる。このため、本実施の形態によれば、生成したプログラムコードを、プログラム言語に対応する静的コード解析ツールを用いて検査をする必要がなくなり、開発者の手間を省くことができる。

[0084] 実施の形態３．

実施の形態１及び２では、ブロック図取得部１１を介して外部からブロック図２が入力される。

これに代えて、ブロック図２をプログラムコード生成装置１内で生成するようにしてもよい。

本実施の形態に係るプログラムコード生成装置１では、外部からのブロック図２の入力の代わりに図１のブロック図取得部１１がブロック図を生成する。つまり、本実施の形態では、ブロック図取得部１１はブロック図を生成することによりブロック図を取得する。

プログラムコード生成装置１のハードウェア構成例は、図１６に示す通りである。

また、本実施の形態では、外部からのブロック図２の入力の代わりにブロック図取得部１１がブロック図２を生成する。これ以外の動作は、実施の形態１に示したものと同じである。

[0085] 以上、本発明の実施の形態について説明したが、これら２つの実施の形態を組み合わせて実施しても構わない。

あるいは、これら２つの実施の形態のうち、１つを部分的に実施しても構わない。

あるいは、これら２つの実施の形態を部分的に組み合わせて実施しても構

わない。

なお、本発明は、これらの実施の形態に限定されるものではなく、必要に応じて種々の変更が可能である。

[0086] ＊＊＊ハードウェア構成の説明＊＊＊

最後に、プログラムコード生成装置１のハードウェア構成の補足説明を行う。

図１６に示すプロセッサ１９０１は、プロセッシングを行うＩＣ（Ｉｎｔｅｇｒａｔｅｄ　Ｃｉｒｃｕｉｔ）である。

プロセッサ１９０１は、ＣＰＵ（Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）、ＤＳＰ（Ｄｉｇｉｔａｌ　Ｓｉｇｎａｌ　Ｐｒｏｃｅｓｓｏｒ）等である。

補助記憶装置１９０２は、ＲＯＭ（Ｒｅａｄ　Ｏｎｌｙ　Ｍｅｍｏｒｙ）、フラッシュメモリ、ＨＤＤ（Ｈａｒｄ　Ｄｉｓｋ　Ｄｒｉｖｅ）等である。

主記憶装置１９０３は、ＲＡＭ（Ｒａｎｄｏｍ　Ａｃｃｅｓｓ　Ｍｅｍｏｒｙ）である。

通信装置１９０４は、データを受信するレシーバー及びデータを送信するトランスミッターを含む。

通信装置１９０４は、例えば、通信チップ又はＮＩＣ（Ｎｅｔｗｏｒｋ　Ｉｎｔｅｒｆａｃｅ　Ｃａｒｄ）である。

入出力装置１９０５は、例えば、マウス、キーボード、ディスプレイ等である。

[0087] また、補助記憶装置１９０２には、ＯＳ（Ｏｐｅｒａｔｉｎｇ　Ｓｙｓｔｅｍ）も記憶されている。

そして、ＯＳの少なくとも一部が主記憶装置１９０３にロードされ、プロセッサ１９０１により実行される。

プロセッサ１９０１はＯＳの少なくとも一部を実行しながら、ブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコ

ード生成部１７の機能を実現するプログラムを実行する。

プロセッサ１９０１がＯＳを実行することで、タスク管理、メモリ管理、ファイル管理、通信制御等が行われる。

また、プログラムコード生成装置１は、プロセッサ１９０１を代替する複数のプロセッサを備えていてもよい。これら複数のプロセッサは、ブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコード生成部１７の機能を実現するプログラムの実行を分担する。それぞれのプロセッサは、プロセッサ１９０１と同じように、プロセッシングを行うＩＣである。

また、ブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコード生成部１７の処理の結果を示す情報やデータや信号値や変数値が、記憶装置１９０２、プロセッサ１９０１内のレジスタ及びキャッシュメモリの少なくともいずれかに記憶される。

また、ブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコード生成部１７の機能を実現するプログラムは、磁気ディスク、フレキシブルディスク、光ディスク、コンパクトディスク、ブルーレイ（登録商標）ディスク、ＤＶＤ等の可搬記憶媒体に記憶されてもよい。

[0088] また、ブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコード生成部１７の「部」を、「回路」又は「工程」又は「手順」又は「処理」に読み替えてもよい。

また、プログラムコード生成装置１は、ロジックＩＣ（Ｉｎｔｅｇｒａｔｅｄ Ｃｉｒｃｕｉｔ）、ＧＡ（Ｇａｔｅ Ａｒｒａｙ）、ＡＳＩＣ（Ａｐｐｌｉｃａｔｉｏｎ Ｓｐｅｃｉｆｉｃ Ｉｎｔｅｇｒａｔｅｄ Ｃｉｒｃｕｉｔ）、ＦＰＧＡ（Ｆｉｅｌｄ－Ｐｒｏｇｒａｍｍａｂｌｅ Ｇａｔｅ Ａｒｒａｙ）といった電子回路により実現されてもよい。

この場合は、ブロック図取得部１１、検査対象抽出部１４、ブロック図検査部１５及びプログラムコード生成部１７は、それぞれ電子回路の一部として実現される。

なお、プロセッサ及び上記の電子回路を総称してプロセッシングサーキットリーともいう。

符号の説明

[0089] １　プログラムコード生成装置、２　ブロック図、３　プログラムコード、４　違反情報、１１　ブロック図取得部、１２　ブロック図解析ルール記憶部、１３　検査ルール記憶部、１４　検査対象抽出部、１５　ブロック図検査部、１６　コード生成ルール記憶部、１７　プログラムコード生成部、１８　矛盾検査部、６００　解析ルール、７００　検査ルール、８００　コード生成ルール、９００　コード生成ルール、１０００　プログラムコード、１１００　プログラムコード。

請求の範囲

[請求項1] プログラムコードに反映させるデータ処理手順を複数のブロックの接続により定義するブロック図を取得するブロック図取得部と、

前記ブロック図でのブロック間の接続を辿って、前記ブロック図で定義されているデータ処理手順に矛盾があるかどうかを検査する矛盾検査部とを有するプログラムコード生成装置。

[請求項2] 前記矛盾検査部は、

前記ブロック図で定義されているデータ処理手順に矛盾がある場合に、前記データ処理手順における矛盾を通知する違反情報を出力する請求項１に記載のプログラムコード生成装置。

[請求項3] 前記プログラムコード生成装置は、更に、

前記ブロック図で定義されているデータ処理手順に矛盾がない場合に、前記ブロック図からプログラムコードを生成するプログラムコード生成部を有する請求項１に記載のプログラムコード生成装置。

[請求項4] 前記ブロック図取得部は、

前記複数のブロックの各々に前記データ処理手順の要素である処理手順要素が割り当てられているブロック図を取得し、

前記矛盾検査部は、

前記複数のブロックのうちのいずれかのブロックを起点ブロックとして設定し、前記起点ブロックから前記ブロック図におけるブロック間の接続を辿って、前記起点ブロックが依存しているブロックを依存先ブロックとして抽出し、抽出された依存先ブロックが依存しているブロックを更に依存先ブロックとして抽出する動作を繰り返し、前記起点ブロックが直接的及び間接的に依存している依存先ブロックを抽出し、

前記起点ブロックに割り当てられている処理手順要素と、抽出された依存先ブロックに割り当てられている処理手順要素との間に矛盾があるかどうかを検査する請求項１に記載のプログラムコード生成装置

。

[請求項5] 前記矛盾検査部は、

前記起点ブロックに割り当てられている処理手順要素と、抽出された依存先ブロックに割り当てられている処理手順要素との間にデータタイプ、最大値及び最小値の少なくともいずれかにおいて矛盾があるかどうかを検査する請求項４に記載のプログラムコード生成装置。

[請求項6] 前記矛盾検査部は、

前記起点ブロックに割り当てられている処理手順要素で扱われるデータについて前記起点ブロックが依存しているブロックを依存先ブロックとして抽出し、抽出された依存先ブロックに割り当てられている処理手順要素で扱われるデータについて当該依存先ブロックが依存しているブロックを更に依存先ブロックとして抽出する動作を繰り返す請求項４に記載のプログラムコード生成装置。

[請求項7] 前記矛盾検査部は、

前記起点ブロックに割り当てられている処理手順要素での制御について前記起点ブロックが依存しているブロックを依存先ブロックとして抽出し、抽出された依存先ブロックに割り当てられている処理手順要素での制御について当該依存先ブロックが依存しているブロックを更に依存先ブロックとして抽出する動作を繰り返す請求項４に記載のプログラムコード生成装置。

[請求項8] 前記プログラムコード生成部は、

前記ブロック図から前記プログラムコードを生成するためのコード生成ルールを参照して、前記ブロック図から前記プログラムコードを生成し、

前記矛盾検査部は、

前記コード生成ルールを参照して、前記ブロック図で定義されているデータ処理手順に矛盾があるかどうかを検査する請求項３に記載のプログラムコード生成装置。

[請求項9] 前記ブロック図取得部は、

前記ブロック図を生成する請求項１に記載のプログラムコード生成装置。

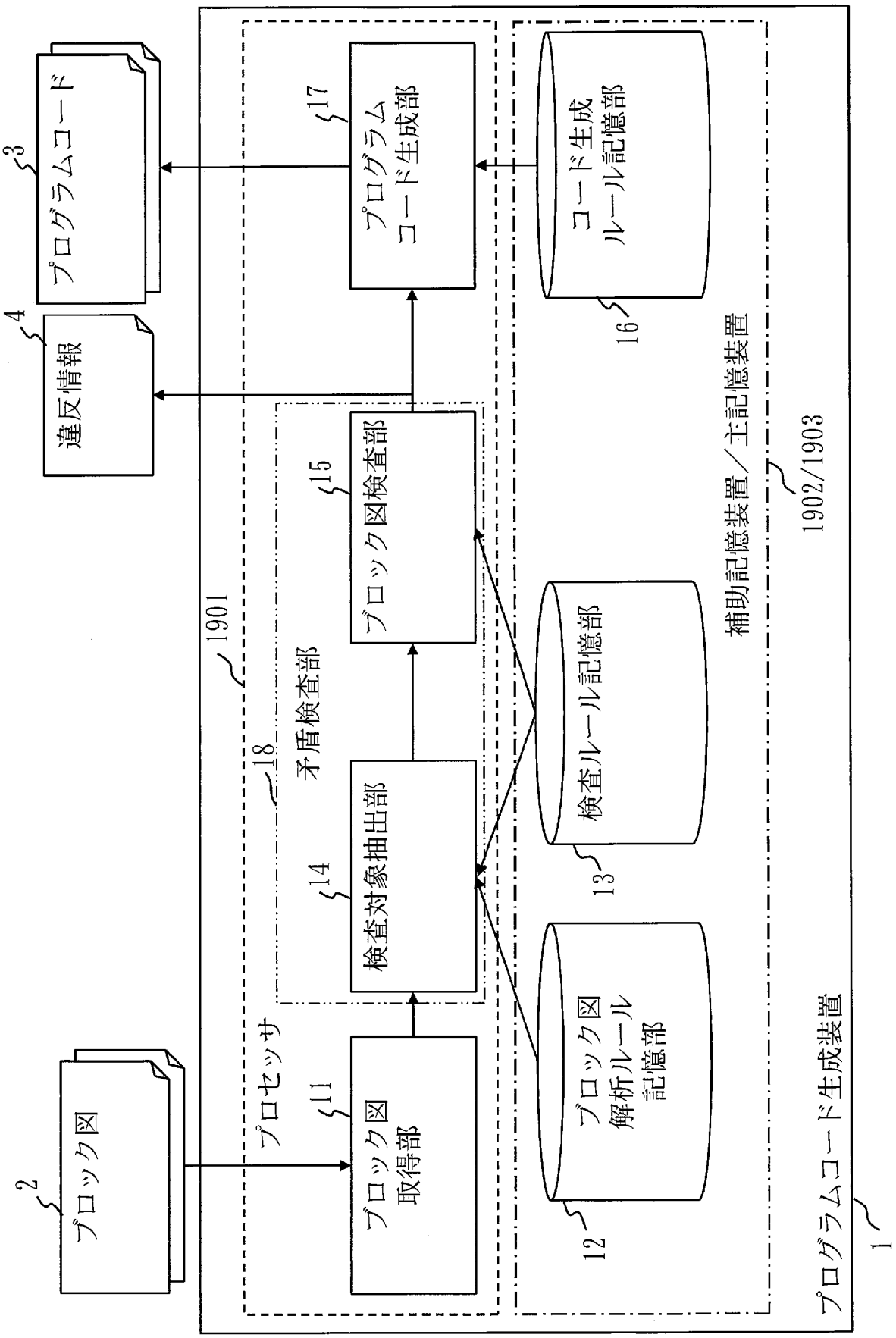
[請求項10] コンピュータが、プログラムコードに反映させるデータ処理手順を複数のブロックの接続により定義するブロック図を取得し、

前記コンピュータが、前記ブロック図でのブロック間の接続を辿って、前記ブロック図で定義されているデータ処理手順に矛盾があるかどうかを検査するプログラムコード生成方法。

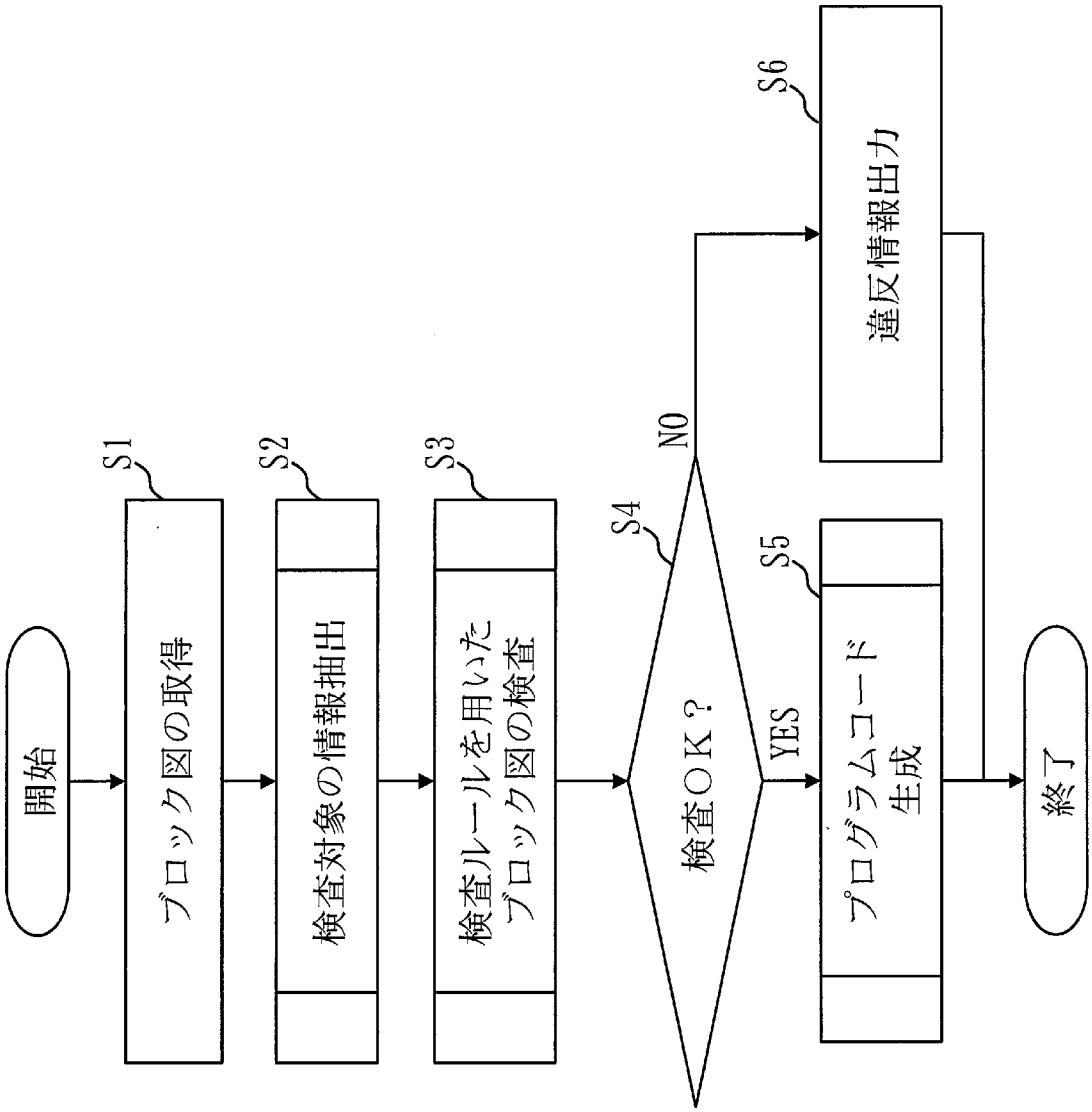
[請求項11] プログラムコードに反映させるデータ処理手順を複数のブロックの接続により定義するブロック図を取得するブロック図取得処理と、

前記ブロック図でのブロック間の接続を辿って、前記ブロック図で定義されているデータ処理手順に矛盾があるかどうかを検査する矛盾検査処理とをコンピュータに実行させるプログラムコード生成プログラム。

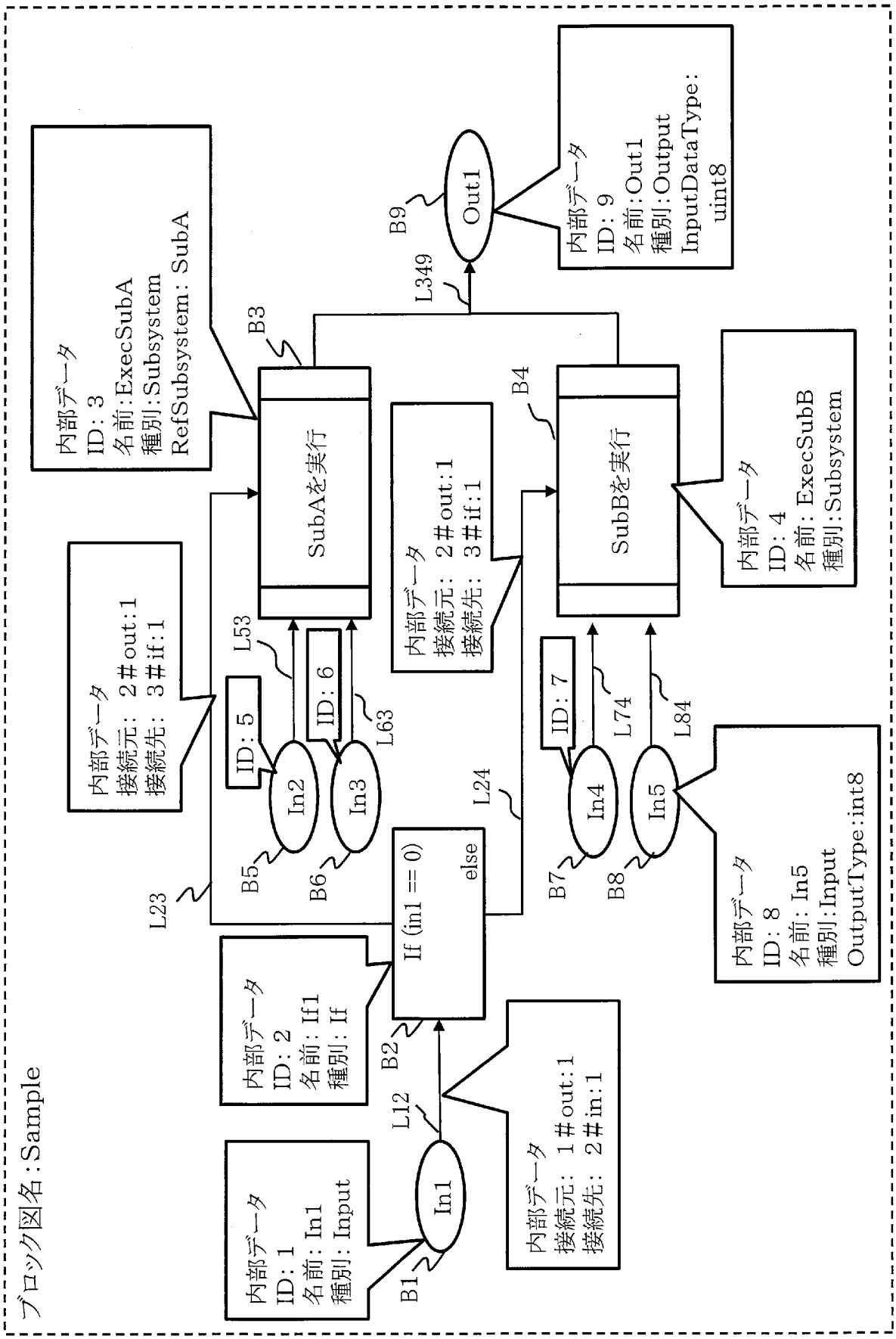
[図1]


プログラムコード生成装置
1
ブロック図
2
プログラムコード
3
違反情報
4
プロセッサ
1901
ブロック図
取得部
11
矛盾検査部
18
検査対象抽出部
14
ブロック図検査部
15
プログラム
コード生成部
17
補助記憶装置／主記憶装置
1902/1903
ブロック図
解析ルール
記憶部
12
検査ルール記憶部
13
コード生成
ルール記憶部
16

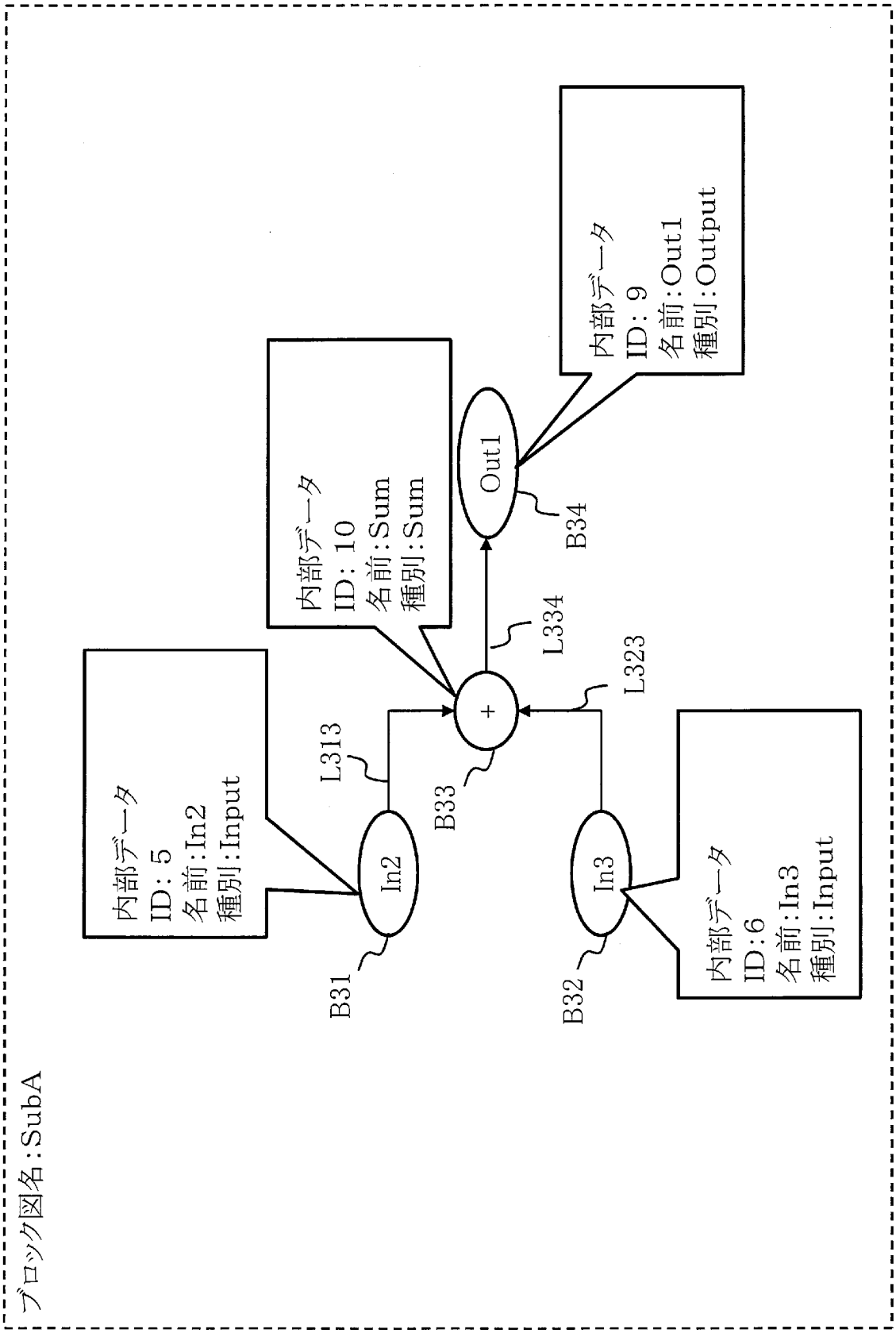
[図2]


開始
S1
ブロック図の取得
S2
検査対象の情報抽出
S3
検査ルールを用いた
ブロック図の検査
S4
検査ＯＫ？
NO
S6
違反情報出力
YES
S5
プログラムコード
生成
終了

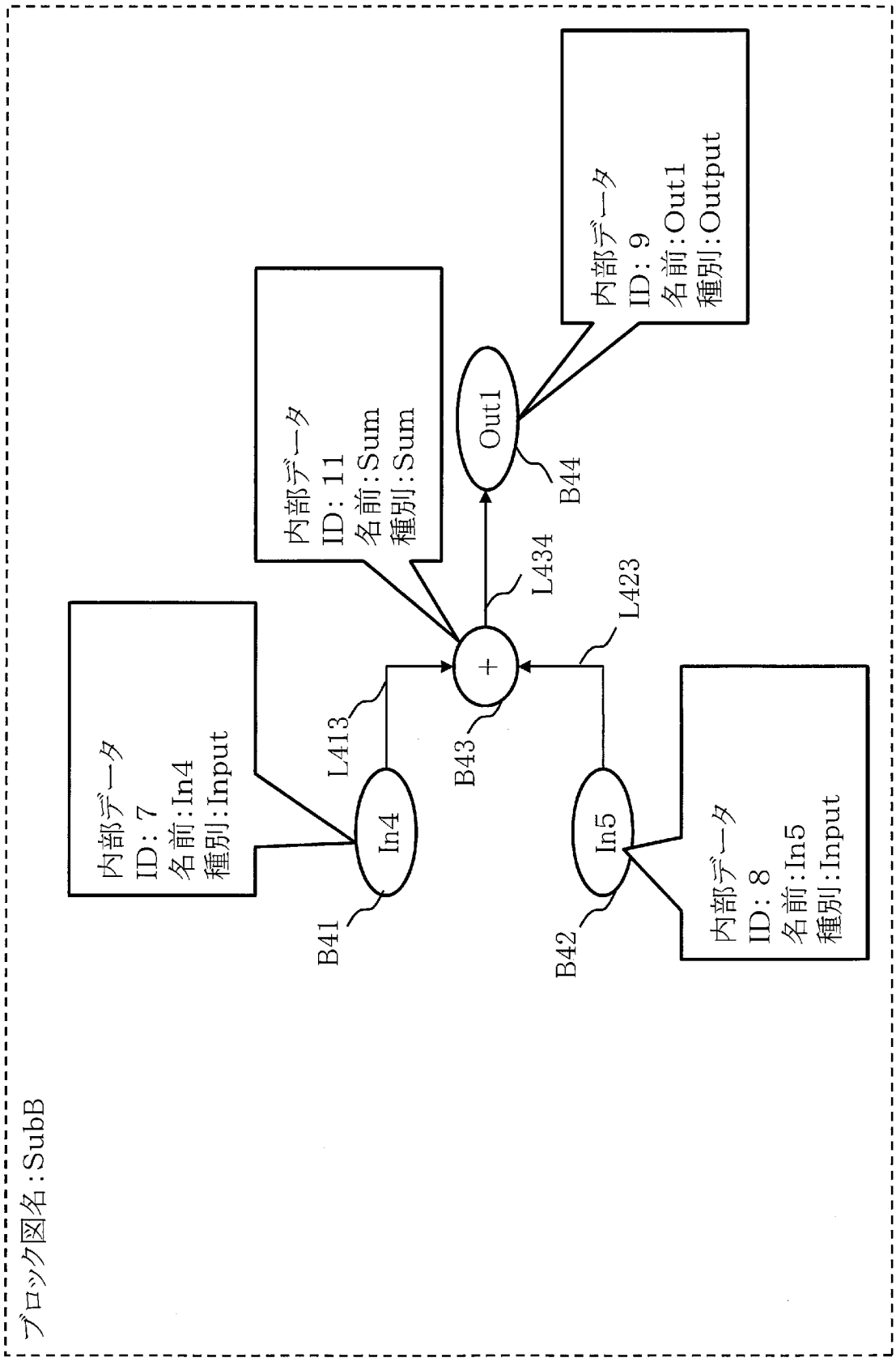
[図3]


ブロック図名:Sample
内部データ
ID: 1
名前: In1
種別: Input
In1
B1
L12
内部データ
ID: 2
名前: If1
種別: If
If (in1 == 0)
else
B2
内部データ
接続元: 1#out:1
接続先: 2#in:1
L23
L24
内部データ
接続元: 2#out:1
接続先: 3#if:1
ID: 5
ID: 6
In2
In3
B5
B6
L53
L63
SubAを実行
B3
内部データ
ID: 3
名前:ExecSubA
種別:Subsystem
RefSubsystem: SubA
L349
B9
Out1
内部データ
ID: 9
名前:Out1
種別:Output
InputDataType:
uint8
B4
SubBを実行
内部データ
ID: 4
名前: ExecSubB
種別:Subsystem
ID: 7
In4
In5
B7
B8
L74
L84
内部データ
ID: 8
名前: In5
種別:Input
OutputType:int8

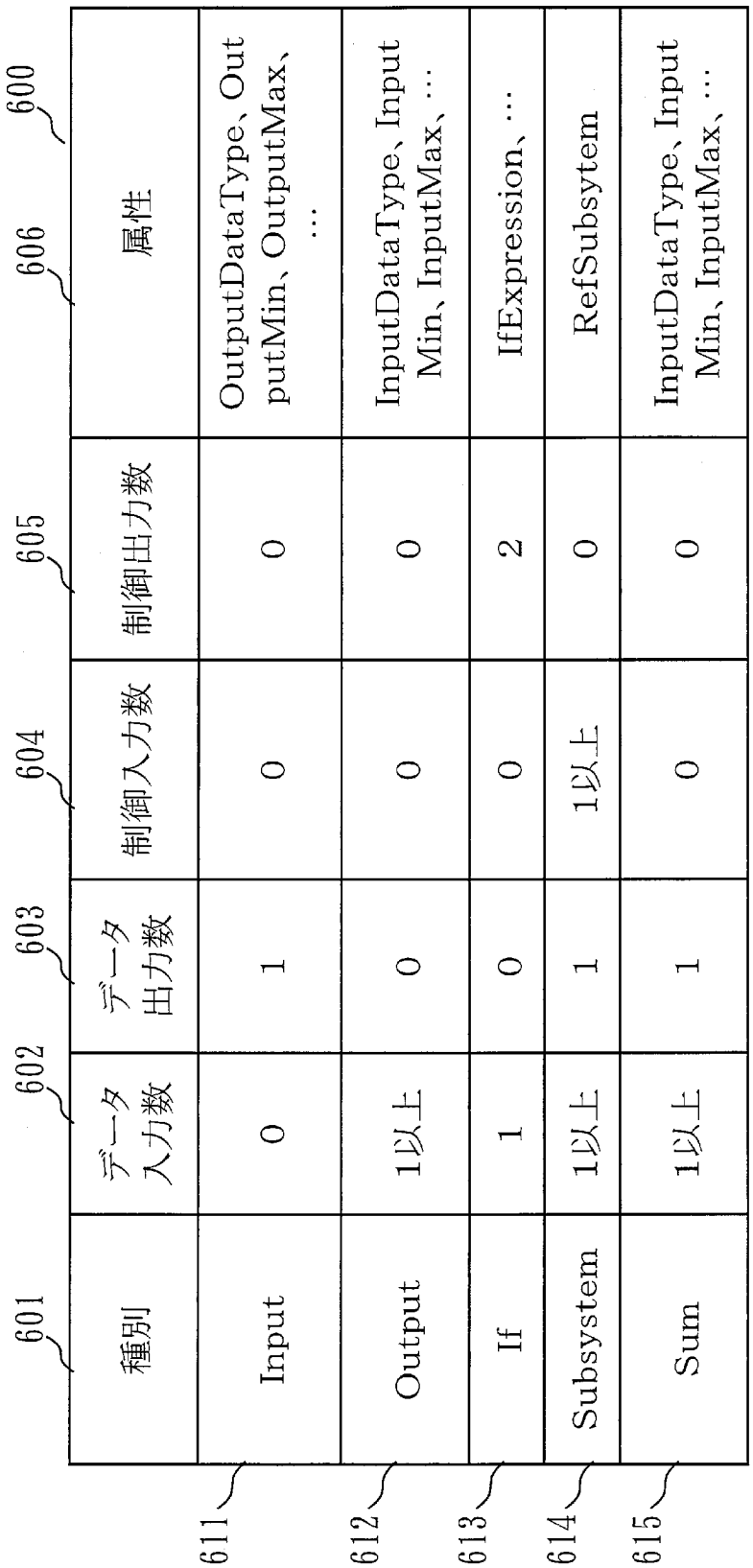
[図4]


ブロック図名:SubA
In2
In3
+
Out1
B31
B32
B33
B34
L313
L323
L334
内部データ
ID: 5
名前:In2
種別:Input
内部データ
ID: 10
名前:Sum
種別:Sum
内部データ
ID: 9
名前:Out1
種別:Output
内部データ
ID:6
名前:In3
種別:Input

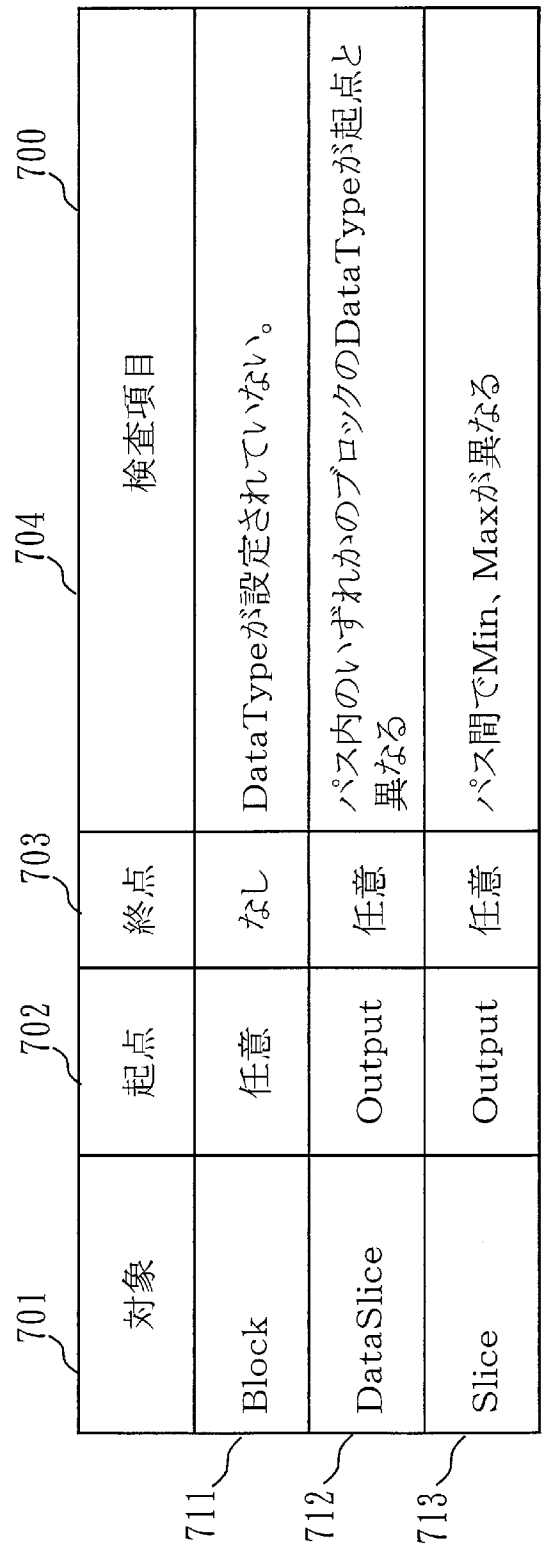
[図5]


ブロック図名:SubB
内部データ
ID: 7
名前:In4
種別:Input
内部データ
ID: 11
名前:Sum
種別:Sum
内部データ
ID: 9
名前:Out1
種別:Outputt
内部データ
ID: 8
名前:In5
種別:Input
In4
In5
+
Out1
B41
B42
B43
B44
L413
L423
L434

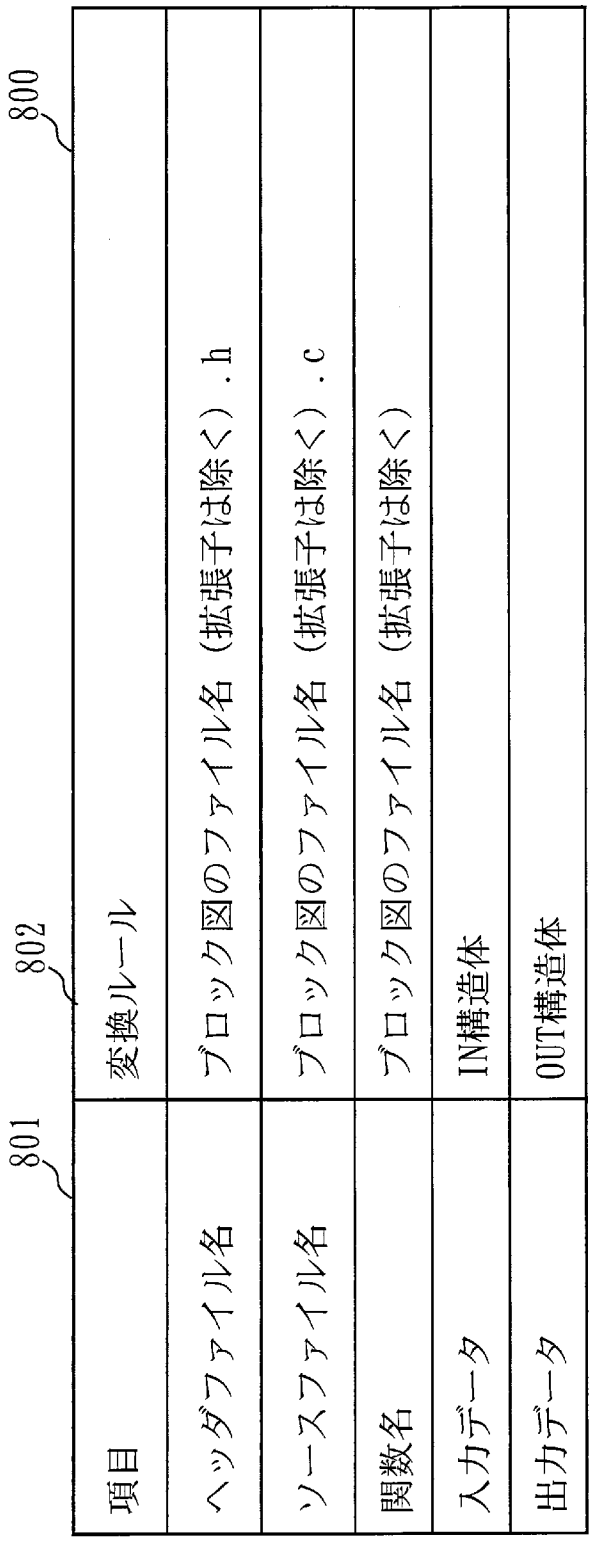
[図6]


600
601
602
603
604
605
606
種別
データ入力数
データ出力数
制御入力数
制御出力数
属性
611
Input
0
1
0
0
OutputDataType、OutputMin、OutputMax、…
612
Output
1以上
0
0
0
InputDataType、InputMin、InputMax、…
613
If
1
0
0
2
IfExpression、…
614
Subsystem
1以上
1
1以上
0
RefSubsytem
615
Sum
1以上
1
0
0
InputDataType、InputMin、InputMax、…

[図7]


700
701
702
703
704
対象
起点
終点
検査項目
711
Block
任意
なし
DataTypeが設定されていないこと。
712
DataSlice
Output
任意
パス内のいずれかのブロックのDataTypeが起点と異なる
713
Slice
Output
任意
パス間でMin、Maxが異なる

[図8]

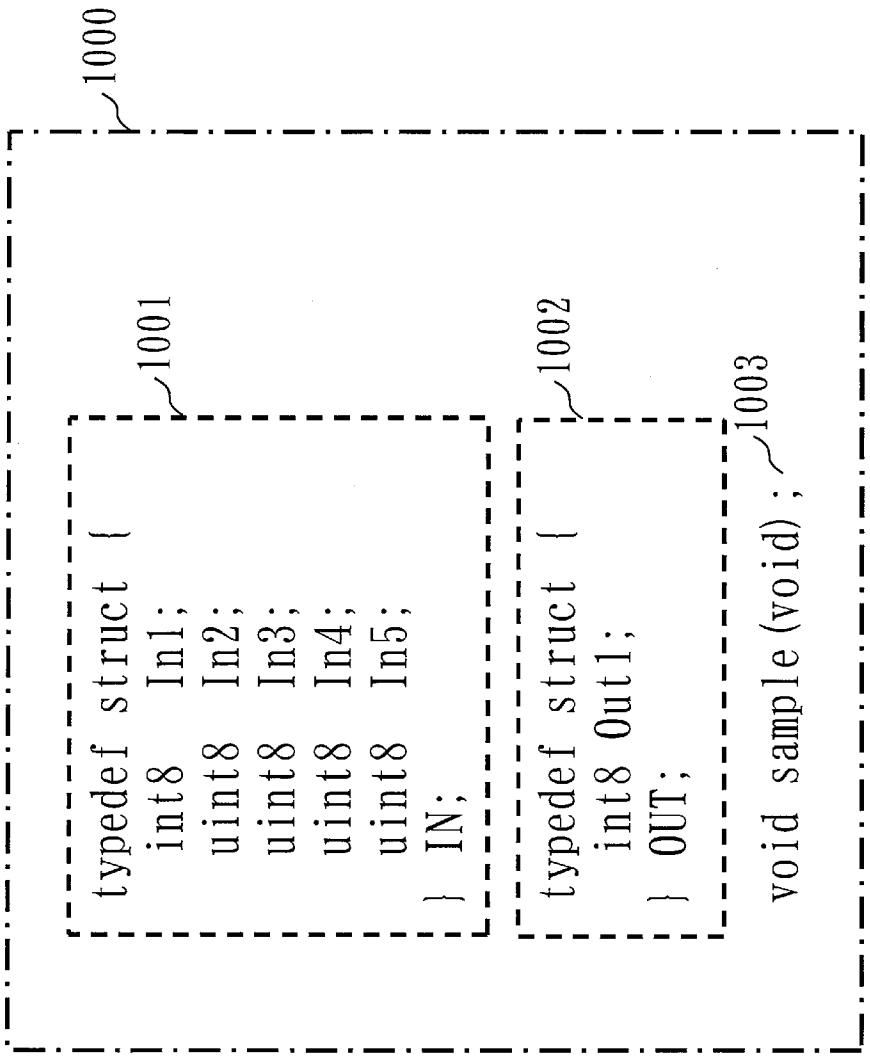

800
801
802
項目
変換ルール
ヘッダファイル名
ブロック図のファイル名（拡張子は除く）.h
ソースファイル名
ブロック図のファイル名（拡張子は除く）.c
関数名
ブロック図のファイル名（拡張子は除く）
入力データ
IN構造体
出力データ
OUT構造体

[図9]

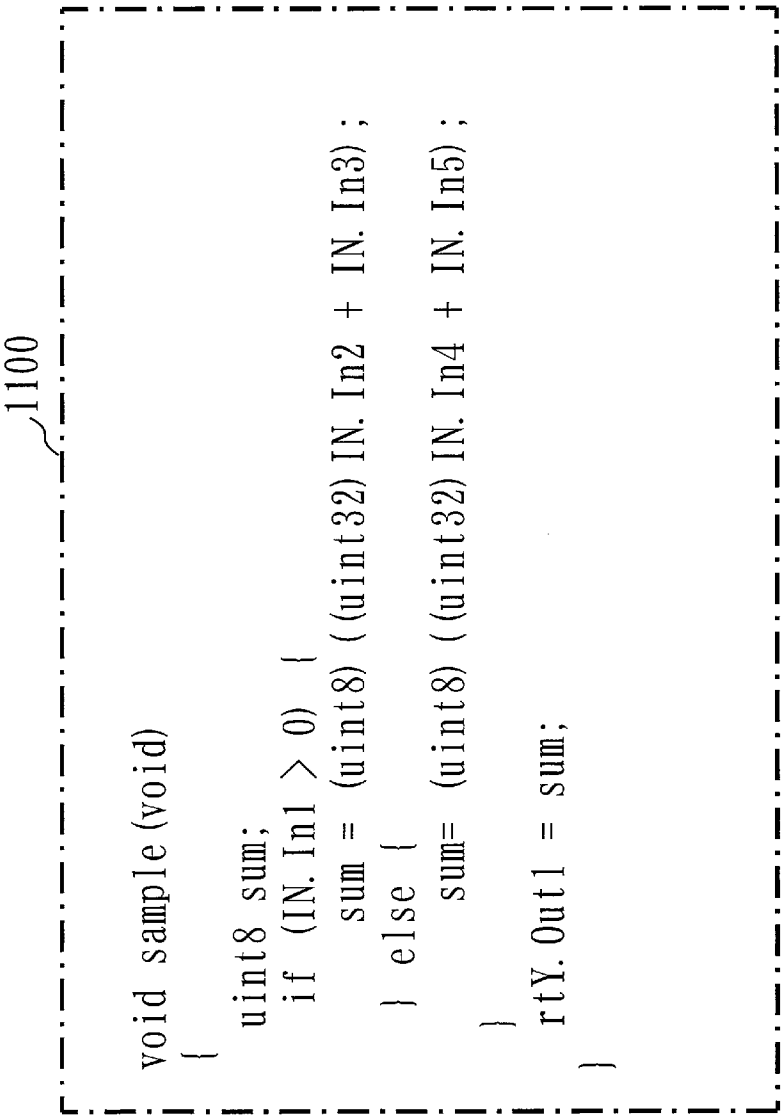
900

ブロックの種別 (901)	変換ルール (902)
Input	・言語上の種別：IN構造体のメンバ ・メンバ名：name属性の値（重複した場合には、末尾に「_」とID属性を追加） ・データ型：type属性の値
Output	・言語上の種別：OUT構造体のメンバ ・メンバ名：name属性の値（重複した場合には、末尾に「_」とID属性を追加） ・データ型：type属性の値
Sum	・言語上の種別：加算演算 〈OUT〉＝〈IN1〉＋〈IN2〉

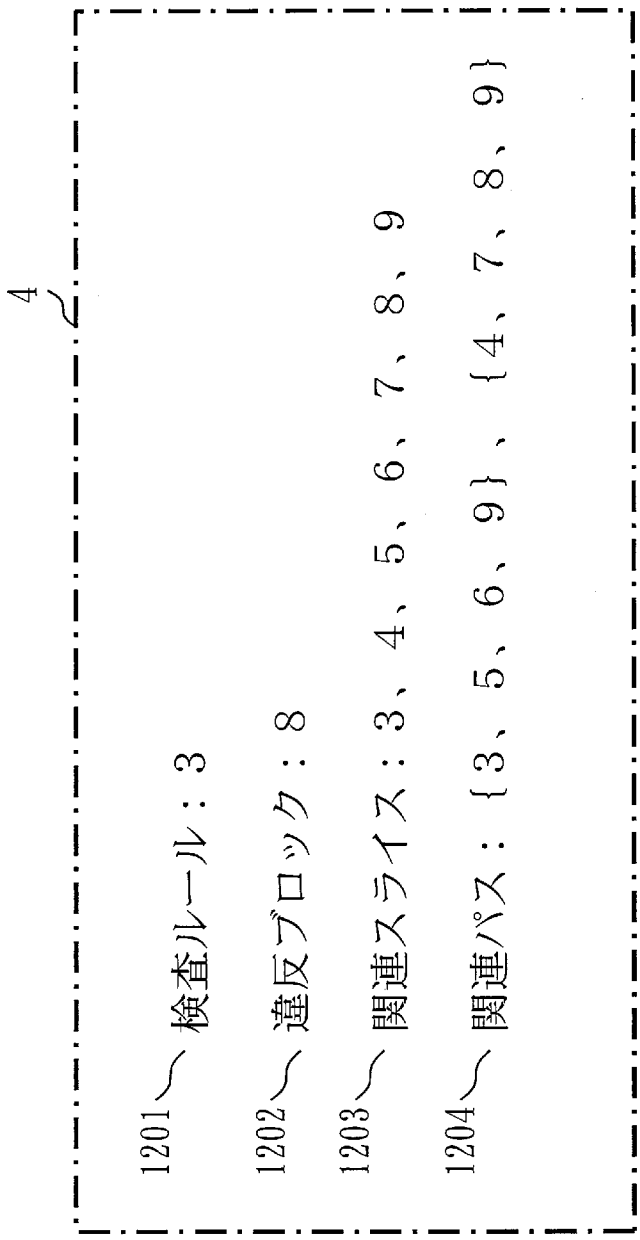
[図10]


1000
1001
typedef struct {
int8 In1;
uint8 In2;
uint8 In3;
uint8 In4;
uint8 In5;
} IN;
1002
typedef struct {
int8 Out1;
} OUT;
1003
void sample(void);

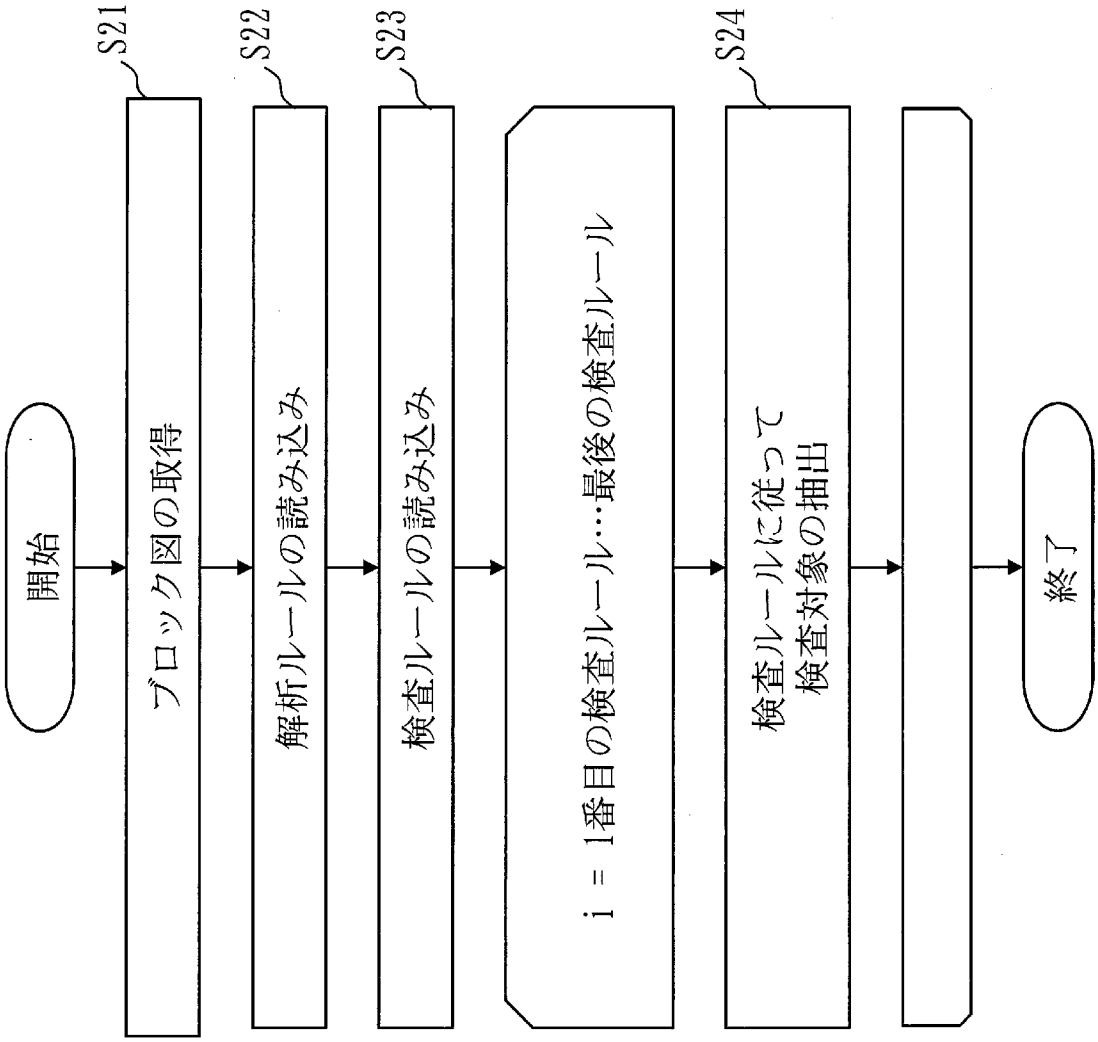
[図11]


1100
void sample(void)
{
uint8 sum;
if (IN.In1 > 0) {
sum = (uint8) ((uint32) IN.In2 + IN.In3);
} else {
sum= (uint8) ((uint32) IN.In4 + IN.In5);
}
rtY.Out1 = sum;
}

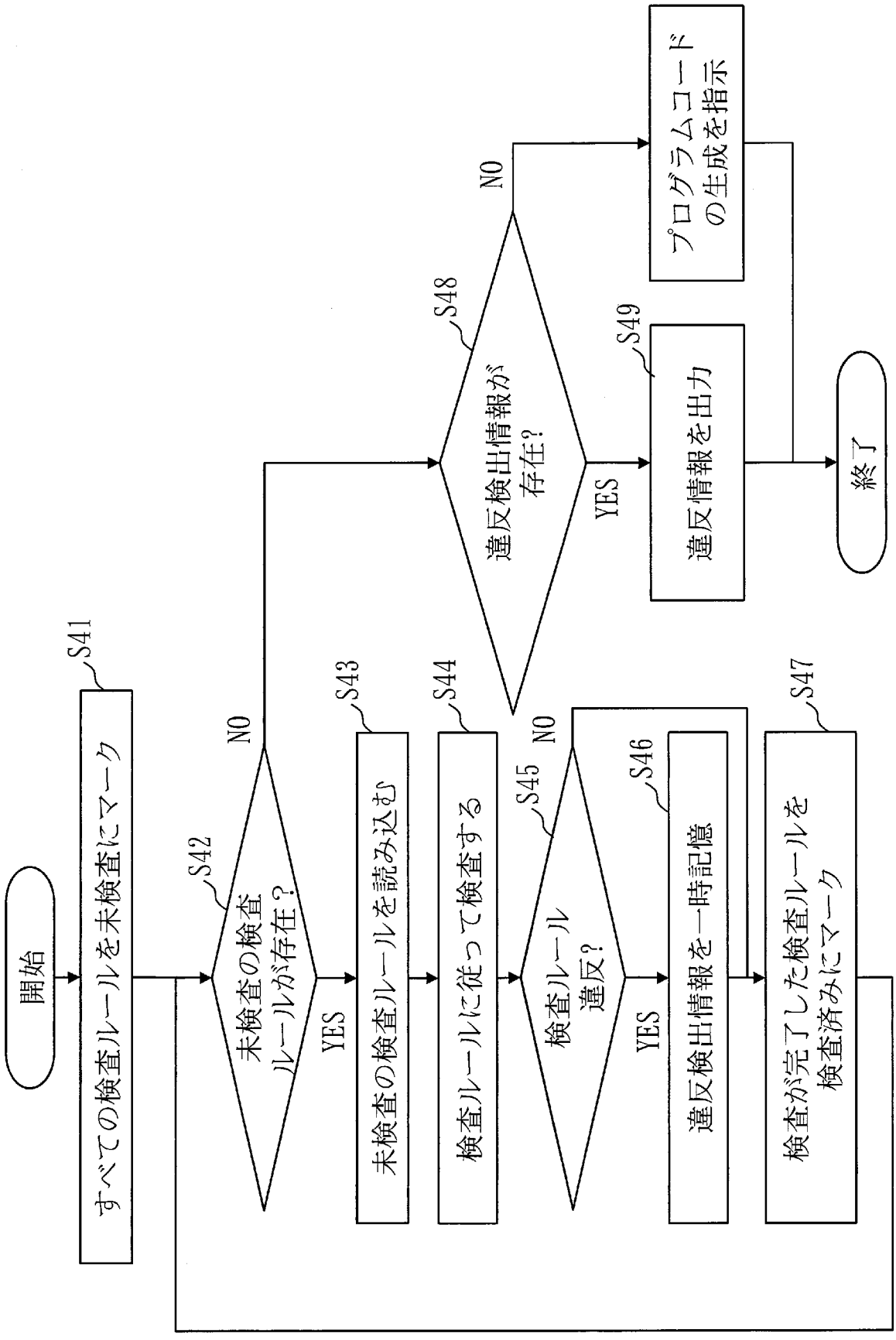
[図12]


4
1201 検査ルール：3
1202 違反ブロック：8
1203 関連スライス：3、4、5、6、7、8、9
1204 関連パス：{3、5、6、9}、{4、7、8、9}

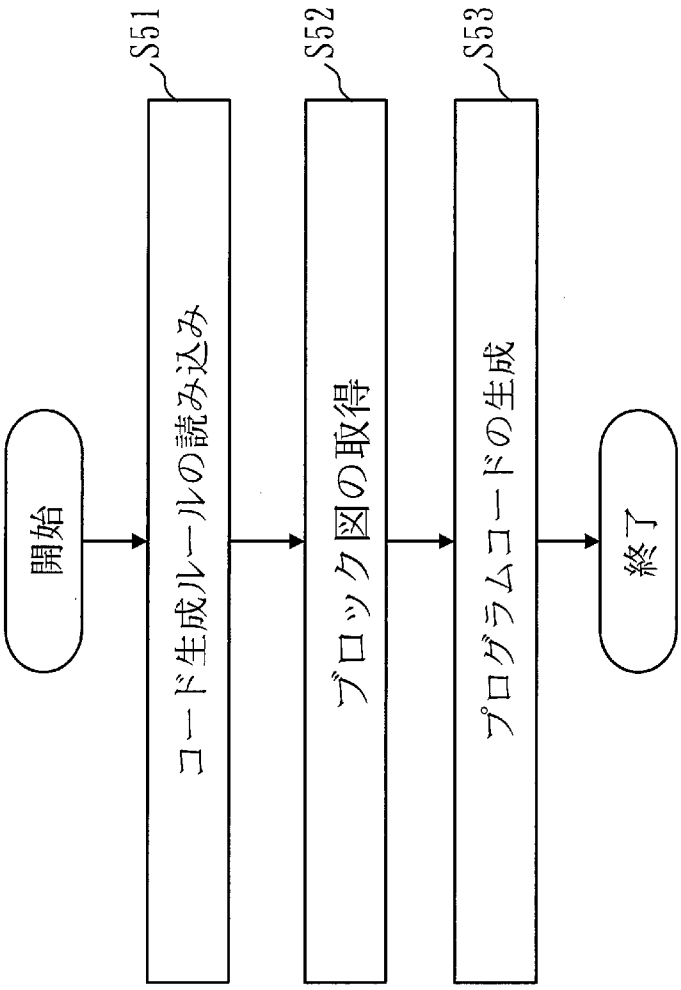
[図13]


開始
S21
ブロック図の取得
S22
解析ルールの読み込み
S23
検査ルールの読み込み
i = 1番目の検査ルール…最後の検査ルール
S24
検査ルールに従って
検査対象の抽出
終了

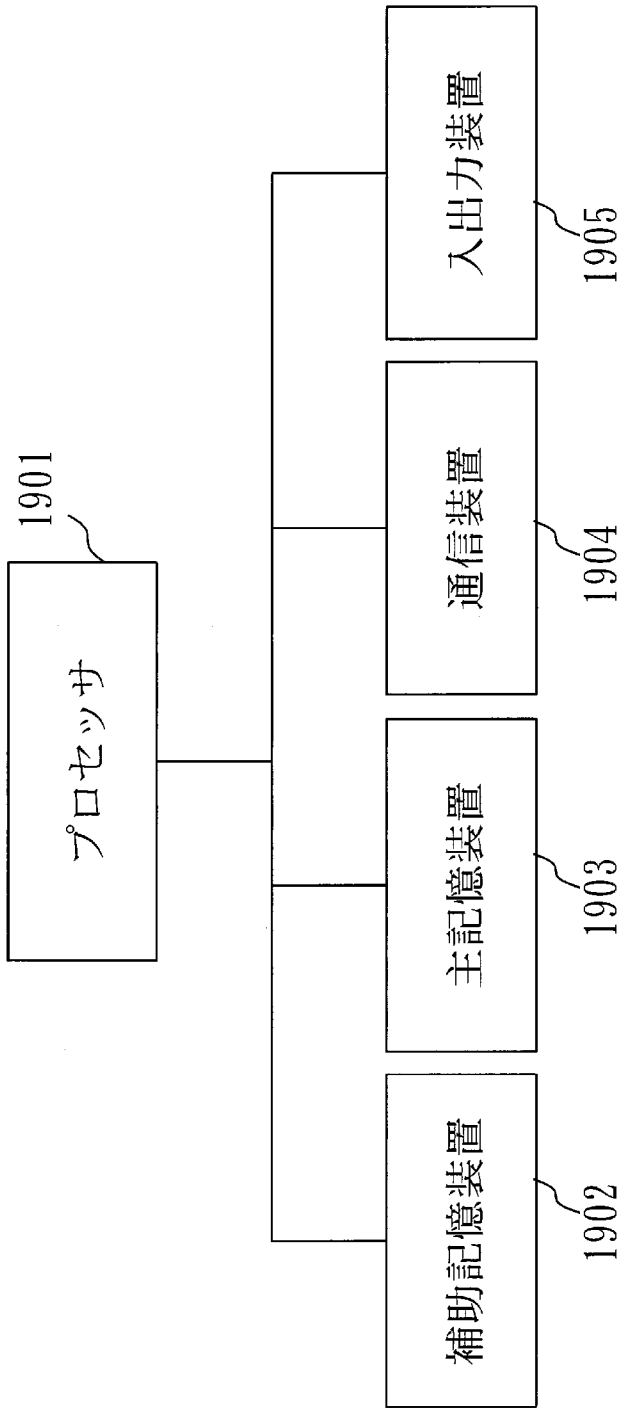
[図14]


開始
すべての検査ルールを未検査にマーク
S41
未検査の検査ルールが存在？
S42
NO
YES
未検査の検査ルールを読み込む
S43
検査ルールに従って検査する
S44
検査ルール違反？
S45
NO
YES
違反検出情報を一時記憶
S46
検査が完了した検査ルールを検査済みにマーク
S47
違反検出情報が存在？
S48
NO
YES
違反情報を出力
S49
プログラムコードの生成を指示
終了

[図15]



[図16]



INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2016/084077

A. CLASSIFICATION OF SUBJECT MATTER
G06F9/44(2006.01)i, *G06F11/36*(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F9/44, G06F11/36

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2017
Kokai Jitsuyo Shinan Koho	1971-2017	Toroku Jitsuyo Shinan Koho	1994-2017

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X A	JP 2010-102362 A (Meidensha Corp.), 06 May 2010 (06.05.2010), paragraphs [0036] to [0050]; fig. 1 to 5 (Family: none)	1-4,9-11 5-8
A	JP 2016-57715 A (Fuji Electric Co., Ltd.), 21 April 2016 (21.04.2016), entire text; all drawings (Family: none)	1-11

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
19 January 2017 (19.01.17)	31 January 2017 (31.01.17)

Name and mailing address of the ISA/	Authorized officer
Japan Patent Office 3-4-3,Kasumigaseki,Chiyoda-ku, Tokyo 100-8915,Japan	Telephone No.

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１６／０８４０７７

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F9/44(2006.01)i, G06F11/36(2006.01)i

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.　G06F9/44, G06F11/36

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２０１７年
日本国実用新案登録公報	１９９６－２０１７年
日本国登録実用新案公報	１９９４－２０１７年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
X	JP 2010-102362 A (株式会社明電舎) 2010.05.06, [0036]-[0050], 図 1-5 (ファミリーなし)	1-4, 9-11
A		5-8
A	JP 2016-57715 A (富士電機株式会社) 2016.04.21, 全文, 全図 (ファミリーなし)	1-11

☐　Ｃ欄の続きにも文献が列挙されている。

☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
１９．０１．２０１７	３１．０１．２０１７

国際調査機関の名称及びあて先
日本国特許庁（ＩＳＡ／ＪＰ）
郵便番号１００－８９１５
東京都千代田区霞が関三丁目４番３号

特許庁審査官（権限のある職員）　５Ｂ　３３５１
石川　亮
電話番号　０３－３５８１－１１０１　内線　３５４５