(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7473145号
(P7473145)

(45)発行日　令和6年4月23日(2024.4.23)

(24)登録日　令和6年4月15日(2024.4.15)

(51)国際特許分類

		FI		
G06F	*8/41 (2018.01)*	G06F	8/41	170
G05B	*19/042 (2006.01)*	G05B	19/042	

請求項の数　5　（全19頁）

(21)出願番号　特願2019-103138(P2019-103138)
(22)出願日　令和1年5月31日(2019.5.31)
(65)公開番号　特開2020-197866(P2020-197866A)
(43)公開日　令和2年12月10日(2020.12.10)
審査請求日　令和4年5月24日(2022.5.24)

(73)特許権者　514318600
コネクトフリー株式会社
京都府京都市下京区四条烏丸西入ル函谷鉾町８３番地
(74)代理人　110001195
弁理士法人深見特許事務所
(72)発明者　帝都 久利寿
京都府京都市下京区四条烏丸西入ル函谷鉾町８３番地 コネクトフリー株式会社内
審査官　金沢 史明

最終頁に続く

(54)【発明の名称】　ソフトウェア開発装置およびソフトウェア開発プログラム

(57)【特許請求の範囲】

【請求項１】

ソフトウェア開発装置であって、

ソースコードに含まれる制約の内容を指定する制約コードを抽出する処理と、

前記ソースコードのうち前記制約コードにより指定される制約が適用される範囲である第１のスコープを決定する処理と、

前記ソースコードのうち前記第１のスコープに関連付けられた範囲である第２のスコープを決定する処理と、

前記第１のスコープに適用される制約と前記第２のスコープに適用される制約とを決定する処理と、

前記第１のスコープおよび前記第２のスコープのソースコードが前記決定される制約に適合するか否かを評価する処理とを実行し、

前記決定される制約は、オブジェクトコードが実行時に利用するリソースに対する制限あるいは規則、前記オブジェクトコードの実行状態に対する制限あるいは規則、前記オブジェクトコードの実行手順に対する制限あるいは規則、ならびに、前記ソースコードに含まれる命令に対する制限あるいは規則のうち、いずれか１つを含む、ソフトウェア開発装置。

【請求項２】

前記ソースコードから前記オブジェクトコードを生成する処理を実行する、請求項１に記載のソフトウェア開発装置。

【請求項３】
前記ソースコードが前記制約に適合していないと評価すると、前記オブジェクトコードの生成を中止する、請求項２に記載のソフトウェア開発装置。
【請求項４】
前記第１のスコープのソースコードが呼び出し命令を含む場合に、当該呼び出し命令により呼び出される命令に対応する範囲が前記第２のスコープとして決定される、請求項１～３のいずれか１項に記載のソフトウェア開発装置。
【請求項５】
ソフトウェア開発プログラムであって、前記ソフトウェア開発プログラムはコンピュータに、
ソースコードに含まれる制約を指定する制約コードを抽出するステップと、
前記ソースコードのうち前記制約コードにより指定される制約が適用される範囲である第１のスコープを決定するステップと、
前記ソースコードのうち前記第１のスコープに関連付けられた範囲である第２のスコープを決定するステップと、
前記第１のスコープに適用される制約と前記第２のスコープに適用される制約とを決定するステップとを実行させ、
前記第１のスコープおよび前記第２のスコープのソースコードが前記決定される制約に適合するか否かを評価する処理とを実行し、
前記決定される制約は、オブジェクトコードが実行時に利用するリソースに対する制限あるいは規則、前記オブジェクトコードの実行状態に対する制限あるいは規則、前記オブジェクトコードの実行手順に対する制限あるいは規則、ならびに、前記ソースコードに含まれる命令に対する制限あるいは規則のうち、いずれか１つを含む、ソフトウェア開発プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本開示は、ソフトウェア開発装置およびソフトウェア開発プログラムに関する。
【背景技術】
【０００２】
近年の情報通信技術（Information and Communication Technology：ＩＣＴ）の進歩は目覚ましく、インターネットなどのネットワークに接続されるデバイスは、従来のパーソナルコンピュータやスマートフォンといった情報処理装置に限らず、様々なモノ（things）に広がっている。このような技術トレンドは、「ＩｏＴ（Internet of Things；モノのインターネット）」と称され、様々な技術およびサービスが提案および実用化されつつある。将来的には、地球上の数十億人と数百億または数兆のデバイスとが同時につながる世界が想定されている。このようなネットワーク化された世界を実現するためには、よりシンプル、より安全、より自由につながることができるソリューションを提供する必要がある。
【０００３】
ＩｏＴで利用されるデバイス（「エッジデバイス」とも称される。）のインテリジェント化に伴って、様々な種類のプログラムの作成が必要となっている。一方で、対象となるデバイスで利用可能なリソースは、パーソナルコンピュータなどに比較して制限されたものとなることが多い。
【０００４】
利用可能なリソースを考慮してプログラムを作成する方法の一例として、特開２００４－０３８９５６号公報は、様々なコンピューティングデバイスで利用可能なコンピューティングリソースを発見し示すための、またこうしたリソースをソフトウェアアプリケーションによってアドレス指定可能なサービスとして露出するためのシステムを開示する。
【先行技術文献】

【特許文献】
【０００５】
【文献】特開２００４－０３８９５６号公報
【発明の概要】
【発明が解決しようとする課題】
【０００６】
エッジデバイスで実行されるプログラムを作成するにあたっては、利用可能なリソースおよびセキュリティの観点から様々な点を考慮する必要がある。しかしながら、上記特許文献１は、対象のコンピューティングデバイスで利用可能なコンピューティングリソースを考慮してプログラムを作成することに着目するにすぎず、プログラムの作成にあたって、様々な点を考慮しなければならないといった課題に対するソリューションを提供するものではない。
【課題を解決するための手段】
【０００７】
本開示のある形態に従えば、ソースコードからオブジェクトコードを生成するソフトウェア開発装置が提供される。ソフトウェア開発装置は、ソースコードに設定される制約を抽出し、当該抽出した制約の適用範囲において、当該ソースコードが当該制約に適合しているか否かを評価する評価手段と、制約に適合するようにオブジェクトコードを生成する生成手段とを含む。
【０００８】
生成手段は、制約の適用範囲において、ソースコードが制約に適合していないと評価されると、オブジェクトコードの生成を中止するようにしてもよい。
【０００９】
生成手段は、制約の適用範囲において、ソースコードが制約に適合しているか否かを評価できない場合に、ソースコードに対応するオブジェクトコードに加えて、当該オブジェクトコードの実行中に、当該制約に適合しているか否かを評価するための別のオブジェクトコードを生成するようにしてもよい。
【００１０】
制約は、オブジェクトコードが実行時に利用するリソースに対する制限あるいは規則、オブジェクトコードの実行状態に対する制限あるいは規則、オブジェクトコードの実行手順に対する制限あるいは規則、ならびに、ソースコードに含まれる命令に対する制限あるいは規則のうち、いずれか１つを含んでいてもよい。
【００１１】
評価手段は、制約の適用範囲に呼び出し命令が含まれる場合に、当該呼び出し命令により呼び出される命令についても、制約に適合しているか否かを評価するようにしてもよい。
【００１２】
本開示の別の形態に従えば、ソースコードからオブジェクトコードを生成するソフトウェア開発プログラムが提供される。ソフトウェア開発プログラムはコンピュータに、ソースコードに設定される制約を抽出し、当該抽出した制約の適用範囲において、当該ソースコードが当該制約に適合しているか否かを評価するステップと、制約に適合するようにオブジェクトコードを生成するステップとを実行させる。
【発明の効果】
【００１３】
本開示によれば、エッジデバイスなどで実行されるプログラムに対して、様々な制約を自在に設定できる環境を提供できる。
【図面の簡単な説明】
【００１４】
【図１】本実施の形態に従うＩｏＴシステムの全体構成の一例を示す模式図である。
【図２】本実施の形態に従うソフトウェア開発装置のハードウェア構成例を示す模式図である。

【図３】本実施の形態に従うコントローラのハードウェア構成例を示す模式図である。
【図４】本実施の形態に従うソフトウェア開発装置において設定可能な制約について説明するための図である。
【図５】本実施の形態に従うソフトウェア開発装置において設定可能な制約のスコープについて説明するための図である。
【図６】本実施の形態に従うソフトウェア開発装置において設定可能な制約の適合を実行時に判断しなければならない場合について説明するための図である。
【図７】本実施の形態に従うソフトウェア開発装置が提供する機能構成を示すブロック図である。
【図８】本実施の形態に従うソフトウェア開発装置における呼び出しに対する制約への適合を実現する方法を説明するための図である。
【図９】本実施の形態に従うソフトウェア開発装置におけるソースコードからオブジェクトコードを生成する処理手順を示すフローチャートである。
【図１０】本実施の形態に従うソフトウェア開発装置が提供するユーザインターフェイス画面の一例を示す図である。
【発明を実施するための形態】
【００１５】
本開示に係る実施の形態について、図面を参照しながら詳細に説明する。なお、図中の同一または相当部分については、同一符号を付してその説明は繰り返さない。
【００１６】
以下の説明においては、典型例として、本実施の形態に従うソフトウェア開発装置１００をＩｏＴシステムに適用した場合について説明するが、本開示はＩｏＴシステムに限らず任意のシステムおよびコントローラに適用可能である。
【００１７】
＜Ａ．ＩｏＴシステム１＞
まず、本実施の形態に従うソフトウェア開発装置１００およびエッジデバイス２を含むＩｏＴシステム１の全体構成について説明する。
【００１８】
図１は、本実施の形態に従うＩｏＴシステム１の全体構成の一例を示す模式図である。図１を参照して、ＩｏＴシステム１においては、典型的には、ソフトウェア開発装置１００においてエッジデバイス２で実行されるプログラム（オブジェクトコード）が生成される。生成されたプログラムは、ソフトウェア開発装置１００からエッジデバイス２に含まれるコントローラ２００へ転送される。
【００１９】
ソフトウェア開発装置１００には、統合開発環境（ＩＤＥ：Integrated Development Environment）が提供されており、ユーザは統合開発環境上で任意のプログラムを作成できる。すなわち、ソフトウェア開発装置１００は、ユーザが任意に作成するソースコードからオブジェクトコードを生成する。
【００２０】
エッジデバイス２としては、どのようなデバイスであってもよいが、典型的には、工場設備、家庭内の各種装置、社会インフラ設備、車両などの移動体、任意の携帯デバイスなどが想定される。後述するように、コントローラ２００は、プロセッサを有しており、ソフトウェア開発装置１００からのプログラムを実行可能になっている。
【００２１】
ＩｏＴシステム１における処理手順の一例について説明する。まず、ユーザがソフトウェア開発装置１００を用いてソースコードを作成する（（１）ソースコード作成）。そして、作成されたソースコードは、ソフトウェア開発装置１００においてコンパイルされ、オブジェクトコードが生成される（（２）オブジェクトコード生成）。生成されたオブジェクトコードは、エッジデバイス２のコントローラ２００へ転送される（（３）オブジェクトコード転送）。転送されたオブジェクトコードは、コントローラ２００で実行される

（（４）オブジェクトコード実行）。

【００２２】

このような手順によって、ソフトウェア開発装置１００において開発された任意のプログラムをコントローラ２００において実行させることができる。

【００２３】

後述するように、本実施の形態に従うソフトウェア開発装置１００は、コントローラ２００で実行されるプログラムに対して、様々な制約を自在に設定できる環境を提供する。典型的には、プログラムが実行されるコントローラ２００が有しているリソース、エッジデバイス２の種類および用途、実行されるプログラムの重要性および確保すべきセキュリティなどに応じて、プログラムに対して様々な制約を任意に設定できる。このような制約の設定によって、限られたリソースを利用したアプリケーションの実現や、アプリケーションの意図しない挙動の防止などを実現できる。

【００２４】

＜Ｂ．ハードウェア構成例＞

次に、本実施の形態に従うＩｏＴシステム１に含まれるデバイスのハードウェア構成例について説明する。

【００２５】

（ｂ１：ソフトウェア開発装置１００）

ソフトウェア開発装置１００は、典型的には汎用コンピュータで実現される。

【００２６】

図２は、本実施の形態に従うソフトウェア開発装置１００のハードウェア構成例を示す模式図である。図２を参照して、ソフトウェア開発装置１００は、主たるコンポーネントとして、プロセッサ１０２と、メインメモリ１０４と、入力部１０６と、ディスプレイ１０８と、ハードディスク１１０と、通信インターフェイス１２２とを含む。これらのコンポーネントは、内部バス１２０を介して接続されている。

【００２７】

プロセッサ１０２は、例えば、ＣＰＵ（Central Processing Unit）やＧＰＵ（Graphics Processing Unit）などで構成される。複数のプロセッサ１０２が配置されてもよいし、複数のコアを有するプロセッサ１０２を採用してもよい。

【００２８】

メインメモリ１０４は、ＤＲＡＭ（Dynamic Random Access Memory）やＳＲＡＭ（Static Random Access Memory）などの揮発性記憶装置で構成される。ハードディスク１１０は、プロセッサ１０２で実行される各種プログラムや各種データを保持する。なお、ハードディスク１１０に代えて、ＳＳＤ（Solid State Drive）やフラッシュメモリなどの不揮発性記憶装置を採用してもよい。ハードディスク１１０に格納されたプログラムのうち、指定されたプログラムがメインメモリ１０４上に展開され、プロセッサ１０２は、メインメモリ１０４上に展開されたプログラムに含まれるコンピュータ可読命令（computer-readable instructions）を順次実行することで、後述するような各種機能を実現する。

【００２９】

典型的には、ハードディスク１１０には、ユーザが任意に作成するソースコード１１２と、統合開発環境を実現するためのソフトウェア開発プログラム１１４と、ソースコード１１２から生成されるオブジェクトコード１１６とが格納される。ソフトウェア開発プログラム１１４は、ユーザが任意に作成するソースコード１１２からオブジェクトコード１１６を生成するものであり、プログラムの開発環境を提供するモジュールを含む。

【００３０】

入力部１０６は、ソフトウェア開発装置１００を操作するユーザの入力操作を受け付ける。入力部１０６は、例えば、キーボード、マウス、表示デバイス上に配置されたタッチパネル、ソフトウェア開発装置１００の筐体に配置された操作ボタンなどであってもよい。

【００３１】

ディスプレイ１０８は、プロセッサ１０２での処理結果などを表示する。ディスプレイ１０８は、例えば、ＬＣＤ（Liquid Crystal Display）や有機ＥＬ（Electro-Luminescence)ディスプレイなどであってもよい。

【００３２】

通信インターフェイス１２２は、コントローラ２００とのデータ交換を担当する。通信インターフェイス１２２は、例えば、ＵＳＢ（Universal Serial Bus）ポート、ＩＥＥＥ１３９４などのシリアルポート、レガシーなパラレルポートといった有線接続端子を含む。あるいは、通信インターフェイス１２２は、イーサネット（登録商標）ポートを含んでいてもよい。

【００３３】

なお、ソフトウェア開発装置１００の全部または一部は、コンピュータ可読命令に相当する回路が組み込まれたＡＳＩＣ（Application Specific Integrated Circuit）などのハードワイヤード回路を用いて実現してもよい。さらにあるいは、ＦＰＧＡ（field-programmable gate array）上にコンピュータ可読命令に相当する回路を用いて実現してもよい。また、プロセッサ１０２およびメインメモリ、ＡＳＩＣ、ＦＰＧＡなどを適宜組み合わせて実現してもよい。

【００３４】

ソフトウェア開発装置１００は、コンピュータ可読命令を含むソフトウェア開発プログラム１１４を格納する非一過性（non-transitory）のメディアから、当該格納しているプログラムなどを読み出すためのコンポーネントをさらに有していてもよい。メディアは、例えば、ＤＶＤ（Digital Versatile Disc）などの光学メディア、ＵＳＢメモリなどの半導体メディアなどであってもよい。

【００３５】

なお、ソフトウェア開発プログラム１１４は、メディアを介してソフトウェア開発装置１００にインストールされるだけではなく、ネットワーク上の配信サーバから提供されるようにしてもよい。

【００３６】

（ｂ２：コントローラ２００）

コントローラ２００は、汎用コンピュータを用いて実現してもよいし、処理を実現するために必要なコンポーネントを含む半導体基板を用いて実現してもよい。

【００３７】

図３は、本実施の形態に従うコントローラ２００のハードウェア構成例を示す模式図である。図３を参照して、コントローラ２００は、主たるコンポーネントとして、演算処理部２１０と、無線通信モジュール２１２と、ＵＳＢコントローラ２１４と、通信コントローラ２１６と、１または複数のパッド２２０と電気的に接続されたＩＯドライバ２１８とを含む。

【００３８】

演算処理部２１０は、プログラムを実行する演算部であり、主たるコンポーネントとして、プロセッサ２０２と、メインメモリ２０４と、フラッシュメモリ２０６とを含む。プロセッサ２０２は、例えば、ＣＰＵやＧＰＵなどで構成される。複数のプロセッサ２０２が配置されてもよいし、複数のコアを有するプロセッサ２０２を採用してもよい。メインメモリ２０４は、ＤＲＡＭやＳＲＡＭなどの揮発性記憶装置で構成される。フラッシュメモリ２０６は、プロセッサ２０２で実行されるプログラムや必要なデータを保持する不揮発性記憶装置である。フラッシュメモリ２０６に格納されたプログラムのうち、指定されたプログラムがメインメモリ２０４上に展開されて、プロセッサ２０２により実行されることで、各種機能が実現される。

【００３９】

無線通信モジュール２１２は、他の任意のデバイスとの間の無線によるデータ交換を担当する。無線通信モジュール２１２は、デバイス、ルータ、移動体基地局などと無線通信するための処理回路およびアンテナなどを含んでもよい。無線通信モジュール２１２が対

応する無線通信は、例えば、Ｗｉ－Ｆｉ（登録商標）、Ｂｌｕｅｔｏｏｔｈ（登録商標）、ＺｉｇＢｅｅ（登録商標）、ＬＰＷＡ（Low Power Wide Area）、ＧＳＭ（登録商標）、Ｗ－ＣＤＭＡ、ＣＤＭＡ２００、ＬＴＥ（Long Term Evolution）、第５世代移動通信システム（５Ｇ）のいずれであってもよい。

【００４０】

ＵＳＢコントローラ２１４は、ソフトウェア開発装置１００とのデータ交換を担当する。通信コントローラ２１６は、他の任意のデバイスとの間の有線によるデータ交換を担当する。通信コントローラ２１６は、例えば、シリアル通信、パラレル通信、ＧＰＩＯ（General-purpose input/output）などの公知のデータ交換方式に対応するようにしてもよい。

【００４１】

ＩＯドライバ２１８は、パッド２２０を介して電気的に接続された任意のデバイスとの間の電気信号の遣り取りを担当する。ＩＯドライバ２１８は、演算処理部２１０からの指令に従って電気信号を出力する。また、ＩＯドライバ２１８は、パッド２２０を介して与えられる電気信号を検知し、その検知結果を演算処理部２１０へ出力する。より具体的には、ＩＯドライバ２１８は、信号生成回路、信号検知回路、バッファ回路などで構成される。

【００４２】

コントローラ２００は、図示しないバッテリからの電力により駆動されてもよい。

＜Ｃ．制約＞

次に、本実施の形態に従うソフトウェア開発装置１００において設定可能な制約について説明する。

【００４３】

本明細書において、「制約」は、ソースコード１１２から生成されるオブジェクトコード１１６（アセンブラコード）の実行において遵守すべき規則を包含する。「制約」としては、オブジェクトコード１１６が実行時に利用するリソースに対する制限あるいは規則、オブジェクトコード１１６の実行状態に対する制限あるいは規則、オブジェクトコード１１６の実行手順に対する制限あるいは規則、ならびに、ソースコード１１２に含まれる命令に対する制限あるいは規則などを含み得る。

【００４４】

図４は、本実施の形態に従うソフトウェア開発装置１００において設定可能な制約について説明するための図である。図４（Ａ）に示すソースコード１１２は、予め定義されたファンクションｆｎ１（）により決定される出力値を指定されたアドレスに書き込むためのコード例である。

【００４５】

より具体的には、ソースコード１１２は、出力値変数の定義１１２１と、出力値変数の定義１１２１とを含む。ファンクションｆｎ１（）の戻り値が出力値ＯｕｔＶａｌｕｅにセットされる（命令１１２３）。そして、出力値ＯｕｔＶａｌｕｅの値がアドレス「０ｘ１０００」に書き込まれる（命令１１２４）。

【００４６】

図４（Ａ）に示すソースコード１１２に対して、制約コード１１２５を追加することで、制約を設定できる。図４（Ｂ）に示すソースコード１１２においては、アクセス可能なメモリ範囲を指定するための制約コード１１２５が追加されている。例えば、「$allowedAddressRange = 0x0000 ... 0x0FFF」は、「０ｘ００００」～「０ｘ０ＦＦＦ」のアドレスに対してのみアクセスが可能であることを意味する。

【００４７】

このような制約が設定された場合において、出力値ＯｕｔＶａｌｕｅの値をアドレス「０ｘ１０００」に書き込むための命令１１２４は、制約に適合しないことになる。すなわち、命令１１２４は実行不可となる。

【００４８】

図４（Ｂ）に示すような制約コード１１２５は、例えば、コントローラ２００において、「０ｘ００００」～「０ｘ０ＦＦＦ」のメモリ範囲はノンセキュア領域として設定されており、「０ｘ１０００」～「０ｘ１ＦＦＦ」のメモリ範囲はセキュア領域として設定されているような場合に有効である。

【００４９】

制約に適合しない命令の実行を禁止する典型的な手法として、（１）ソースコード１１２からオブジェクトコード１１６を生成する過程において判断する方法、および、（２）オブジェクトコード１１６の生成時に判断する方法が想定される。（１）の方法については、プリプロセッサ、コンパイラ、オプティマイザなどのオブジェクトコード１１６を生成するための機能が制約への適合を評価する。一方、（２）の方法については、ソースコード１１２から生成されるオブジェクトコード１１６に加えて、制約への適合を評価するためのオブジェクトコード（以下、「適合評価用オブジェクトコード」とも称す。）を生成するようにしてもよい。適合評価用オブジェクトコード１１８は、オブジェクトコード１１６の一部に組み込まれてもよいし、オブジェクトコード１１６とは独立して存在してもよい。このような実装例の詳細については後述する。

【００５０】

次に、本実施の形態に従う制約の適用範囲（以下、「スコープ」とも称す。）について説明する。

【００５１】

図４（Ｂ）に示すように、基本的には、制約コード１１２５の記述に引き続く部分に対して、制約コード１１２５に規定された制約が適用される。すなわち、制約コード１１２５の記述に引き続く部分が制約のスコープとなる。この制約のスコープの終了位置は、任意に設定できるが、基本的には、制約コード１１２５を含むカッコの範囲内を制約のスコープとすることができる。

【００５２】

また、プロシージャやファンクションが呼び出された場合には、それらの呼び出されたプロシージャやファンクションについても制約のスコープとしてもよい。

【００５３】

図５は、本実施の形態に従うソフトウェア開発装置１００において設定可能な制約のスコープについて説明するための図である。図５に示すように、ファンクションｆｎ１（）を呼び出して、ファンクションｆｎ１（）の戻り値を出力値ＯｕｔＶａｌｕｅにセットするという命令１１２３については、呼び出されるファンクションｆｎ１（）の部分ソースコード１１２６についても制約のスコープに含まれるようにしてもよい。

【００５４】

このように、ソフトウェア開発装置１００は、制約のスコープに呼び出し命令が含まれる場合に、当該呼び出し命令により呼び出される命令（プロシージャやファンクション）についても、制約に適合しているか否かを評価する。このような制約のスコープを順次継承することで、呼び出されるプロシージャやファンクションを規定するソースコードについても制約への適合を評価することで、制約を確実に遵守できる。

【００５５】

図６は、本実施の形態に従うソフトウェア開発装置１００において設定可能な制約の適合を実行時に判断しなければならない場合について説明するための図である。図６（Ａ）には、図４（Ｂ）に示すソースコード１１２と同様のソースコード１１２を示す。図６（Ａ）に示すソースコード１１２においては、出力値ＯｕｔＶａｌｕｅの値を書き込むアドレスが「０ｘ１０００」に固定されており（命令１１２４）、ソースコード１１２を語句解析および構文解析することで、制約に適合していないと評価できる。

【００５６】

これに対して、図６（Ｂ）に示すソースコード１１２は、出力値ＯｕｔＶａｌｕｅの値を書き込むアドレスは出力アドレス初期値ＩｎｉＰｔｒを用いて決定されるので、書き込むアドレスをソースコード１１２の解析だけでは一意に決定できない。

【００５７】
より具体的には、図６（Ｂ）に示すソースコード１１２においては、出力アドレス初期値変数の定義１１２７に加えて、出力アドレス初期値ＩｎｉＰｔｒを決定する処理１１２８が規定されている。そして、出力アドレスＯｕｔＡｄｄｒｓは、出力アドレス初期値ＩｎｉＰｔｒに「０Ｆ００」を加算して決定することを規定する出力アドレスを決定する命令１１２９がソースコード１１２に規定されている。そして、出力値ＯｕｔＶａｌｕｅの値が出力アドレスＯｕｔＡｄｄｒｓにより示されるアドレスに書き込まれる（命令１１３０）。
【００５８】
図６（Ｂ）に示されるソースコード１１２においては、出力アドレスＯｕｔＡｄｄｒｓの値は、出力アドレス初期値ＩｎｉＰｔｒの値に依存することになり、対応するオブジェクトコード１１６が実行される時点で、動的に決定されることになる。
【００５９】
本実施の形態に従うソフトウェア開発装置１００は、対応するオブジェクトコード１１６の実行中においても、制約への適合を評価できる仕組みを提供する（詳細については後述する）。
【００６０】
＜Ｄ．制約の種類＞
本実施の形態に従う制約としては、以下のような種類のものを採用してもよい。
【００６１】

【表１】

No	制約コード	設定値	説明
1	$allowedAddressRange	0x0000 ... 0xFFFF	アクセス可能なメモリ範囲の指定
2	$allowedScopeDepth	usize (default: none)	スコープが有効なデプス指定 $allowSelfRecursion は False に設定される
3	$allowedStackDepth	usize (default: none)	スコープが有効なスタックのデプス指定 $allowSelfRecursion は False に設定される
4	$enableRuntimeSafety	True | False (default: True)	ランタイムのセーフティ実行の有効/無効
5	$enforceStrictSwitches	True | False (default: True)	()ステートメント内でのすべてのオプション指定を強制/不強制
6	$allowPanics	True | False (default: False)	パニック発生の許容/不許容
7	$allowUndefinedVariables	True | False (default: False)	未定義変数の使用の可否
8	$allowUnreachable	True | False (default: False)	指定された宛先に到達できない状態の許容/不許容
9	$allowNoReturns	True | False (default: False)	戻り値が得られない状態の許容/不許容
10	$clearMemoryUponEnd	true | false (default: false)	スコープが終了するとメモリを完全クリアするか否か
11	$allowInputToOutput	True | False (default: True)	指定されたメモリのスコープからリターンの許容/不許容
12	$allowConditionalBranches	True | False (default: True)	条件分岐の使用の可否
13	$allowSelfRecursion	True | False (default: True)	再帰呼び出しの使用の可否
14	$allowPtrToZero	True | False (default: False)	0x0 を示すポインタの使用の可否
15	$allowRawPointers	True | False (default: False)	物理値を示すポインタの使用の可否
16	$inlineFunctionCalls	True | False (default: False)	呼び出す function のすべてをインライン展開することの強制/不強制
17	$floatMode = @import("builtin").FloatMode	Strict/ Optimized (default:.Strict)	Strict: IEEE コンプライアンスに沿った浮動小数点演算の実行 Optimized: 高速な数学処理への最適化
18	$evalBranchQuota	Usize (default: 1000)	バックワード可能な最大数
19	$allowSubWithOverflow	True | False (default: True)	オーバーフローを伴う減算処理の許容/不許容
20	$allowMulWithOverflow	True | False (default: True)	オーバーフローを伴う掛算処理の許容/不許容
21	$allowRem	True | False (default: True)	剰余処理の許容/不許容
22	$allowPtrCast	True | False (default: True)	型変換処理の許容/不許容
23	$allowMemset	True | False (default: True)	メモリブロック設定の許容/不許容

【００６２】

【表２】

No	制約コード	設定値	説明
24	$allowIntToPtr	True	False (default: True)	ポインタへの整数設定の許容/不許容
25	$allowBreakpoint	True	False (default: True)	ブレークポイント使用の許容/不許容
26	$allowBitCas	True	False (default: True)	ビットキャスト演算子の許容/不許容
27	$allowExternalCalls	True	False (default: True)	外部ファンクションのコールの許容/不許容
28	$allowExternalVariables	True	False (default: True)	外部変数の許容/不許容
29	$allowVarArgs	True	False (default: True)	変数を引数する使用の許容/不許容
30	$allowErrorTermination	True	False (default: True)	スコープから生じるエラーストリームの許容/不許容
31	$allowGlobalVariables	True	False (default: True)	グローバル変数使用の許容/不許容
32	$allowGlobalConstants	True	False (default: True)	グローバル定数使用の許容/不許容
33	$allowIntermediateVariables	True	False (default: True)	中間変数/中間定数の使用の許容/不許容 *中間変数/中間定数は、直前のスコープおよび後続のグローバルスコープで使用される構造体および列挙型の集合を意味する
34	$allowIntermediateConstants	True	False (default: True)	
35	$allowSuperVariables	True	False (default: True)	超変数/超定数の使用の許容/不許容 *超変数/超定数は、現在のスコープでのみ使用される構造体および列挙型の集合を意味する
36	$allowSuperConstants	True	False (default: True)	
37	$allowLocalVariables	True	False (default: True)	ローカル変数使用の許容/不許容
38	$allowLocalConstants	True	False (default: True)	ローカル定数使用の許容/不許容
39	$allowThreadLocalVariables	True	False (default: True)	ローカル変数のスレッド使用の許容/不許容
40	$allowInlineAssembly	True	False (default: True)	インライン展開されたアセンブリのスコープへの適用可否
41	$coldScope	True	False (default: False)	オプティマイザにスコープの実行頻度が低いことの通知有無
42	$secureScope	True	False	True に設定されることは以下を意味する $allowInputToOutput を False に設定 $clearMemoryOnExit を True に設定 $allowConditionalBranches を False に設定 $enableRuntimeSafety を True に設定 $allowUnreachablePaths を False に設定 $allowUndefinedVariables を False に設定 $allowPanics を False に設定 $allowFunctionRecursion を False に設定 $floatMode を Strict に設定 $allowErrorTermination を False に設定

【００６３】

上述したような制約は、典型的には、オブジェクトコード１１６が実行時に利用するリソースに対する制限あるいは規則、オブジェクトコード１１６の実行状態に対する制限あるいは規則、オブジェクトコード１１６の実行手順に対する制限あるいは規則、ならびに、ソースコード１１２に含まれる命令に対する制限あるいは規則などを含み得る。

【００６４】

なお、上表に示す制約コードのすべてを実装する必要はなく、要求される仕様に応じてその一部のみを実装するようにしてもよい。また、上表に示す制約コード以外の制約コードを採用してもよい。

【００６５】
＜Ｅ．制約への適合を評価する仕組み＞
次に、本実施の形態に従うソフトウェア開発装置１００が制約への適合を評価する仕組みの一例について説明する。
【００６６】
図７は、本実施の形態に従うソフトウェア開発装置１００が提供する機能構成を示すブロック図である。図７に示す各機能は、典型的には、ソフトウェア開発装置１００のプロセッサ１０２がソフトウェア開発プログラム１１４を実行することで実現される。
【００６７】
図７を参照して、ソフトウェア開発プログラム１１４は、ソースコード１１２の入力を受けて、オブジェクトコード１１６（アセンブラコード）を生成する。より具体的には、ソフトウェア開発プログラム１１４は、プリプロセッサ１１４１と、コンパイラ１１４２と、オプティマイザ１１４３と、コードジェネレータ１１４４とを含む。
【００６８】
プリプロセッサ１１４１は、ソースコード１１２に対する語句解析および構文解析を実行するとともに、コンパイラ１１４２、オプティマイザ１１４３およびコードジェネレータ１１４４の挙動を制御する。
【００６９】
コンパイラ１１４２は、ソースコード１１２に対する語句解析および構文解析の結果に基づいて、オブジェクトコードに生成する。オプティマイザ１１４３は、生成されたオブジェクトコードを最適化する。コードジェネレータ１１４４は、オプティマイザ１１４３による最適化の結果に基づいて、最終的なオブジェクトコード１１６を出力する。
【００７０】
上述した制約への適合を評価するにあたって、プリプロセッサ１１４１、コンパイラ１１４２およびオプティマイザ１１４３は、ソースコード１１２に規定された制約を抽出するとともに、抽出した制約への適合を評価する（ステップＳ１）。なお、設定された制約の内容に応じて、オプティマイザ１１４３がオブジェクトコードを修正するようにしてもよい。このように、ソフトウェア開発装置１００は、ソースコード１１２に設定される制約を抽出し、当該抽出した制約のスコープにおいて、当該ソースコード１１２が当該制約に適合しているか否かを評価する。
【００７１】
コードジェネレータ１１４４は、抽出した制約への適合をオブジェクトコード１１６の実行時でなければ評価できない場合に、制約への適合を評価するためのアセンブラコードである適合評価用オブジェクトコード１１８を生成する（ステップＳ２）。
【００７２】
このように、ソフトウェア開発装置１００は、制約に適合するようにオブジェクトコード１１６を生成する。
【００７３】
図８は、本実施の形態に従うソフトウェア開発装置１００における呼び出しに対する制約への適合を実現する方法を説明するための図である。図８に示す処理は、ソフトウェア開発プログラム１１４により提供されてもよいし、オブジェクトコード１１６の実行時に適合評価用オブジェクトコード１１８により提供されてもよい。
【００７４】
図８を参照して、呼び出し元のプロシージャに設定された制約スコープ（以下、「親スコープ」とも称す。）を示すデータセット１５０が生成される。親スコープのデータセット１５０は、変数、制約、プロシージャなどの情報を含む。親スコープのデータセット１５０に対応付けて、管理オブジェクト１５２も生成される。
【００７５】
呼び出し先のプロシージャに継承されるべき制約スコープ（以下、「子スコープ」とも称す。）を示すデータセット１５４は、親スコープのデータセット１５０に関連付けられ

る。子スコープのデータセット１５４に対応付けて、管理オブジェクト１５６も生成される。

【００７６】

例えば、プロシージャやファンクションが呼び出されると、親スコープのデータセット１５０に関連付けられる管理オブジェクト１５２が参照される（ステップＳ１１）。そして、親スコープのデータセット１５０に基づいて子スコープのデータセット１５４が生成される（ステップＳ１２）。子スコープのデータセット１５４から管理オブジェクト１５２への参照（ステップＳ１３）に応答して、管理オブジェクト１５２から管理オブジェクト１５６が生成される（ステップＳ１４）。生成された管理オブジェクト１５６は、子スコープのデータセット１５４と関連付けられる（ステップＳ１５）。

【００７７】

このような一連の処理が繰り返されることで、プロシージャやファンクションが呼び出しに応じて、設定された制約のスコープが継承される。

【００７８】

＜Ｆ．処理手順＞

次に、本実施の形態に従うソフトウェア開発装置１００におけるソースコードからオブジェクトコードを生成する処理手順について説明する。

【００７９】

図９は、本実施の形態に従うソフトウェア開発装置１００におけるソースコードからオブジェクトコードを生成する処理手順を示すフローチャートである。図９に示す各ステップは、典型的には、プロセッサ１０２がソフトウェア開発プログラム１１４を実行することで実現される。

【００８０】

図９を参照して、ソフトウェア開発装置１００は、入力されたソースコード１１２に対して語句解析および構文解析を実行する（ステップＳ１００）。ソフトウェア開発装置１００は、解析結果に基づいて、制約が設定されているか否かを判断する（ステップＳ１０２）。すなわち、ソフトウェア開発装置１００は、ソースコード１１２に設定される制約を抽出し、当該抽出した制約のスコープにおいて、ソースコード１１２が当該制約に適合しているか否かを評価する。ここで、制約が設定されていなければ（ステップＳ１０２においてＮＯ）、途中のステップがスキップされて、ステップＳ１１８以下の処理が実行される。

【００８１】

制約が設定されていれば（ステップＳ１０２においてＹＥＳ）、ソフトウェア開発装置１００は、設定されている制約のうち１つを選択し（ステップＳ１０４）、選択された制約のスコープに含まれるソースコードが当該制約に適合するか否かを判断する（ステップＳ１０６）。

【００８２】

選択された制約のスコープに含まれるソースコードのうち当該制約に適合しない部分があれば（ステップＳ１０６においてＮＯ）、ソフトウェア開発装置１００は、制約に適合しない旨のメッセージを出力し（ステップＳ１０８）、オブジェクトコード１１６を生成する処理を中断する（ステップＳ１１０）。そして、処理は終了する。このように、ソフトウェア開発装置１００は、制約のスコープにおいてソースコード１１２が当該制約に適合していないと評価されると、オブジェクトコード１１６の生成を中止する。

【００８３】

図１０は、本実施の形態に従うソフトウェア開発装置１００が提供するユーザインターフェイス画面の一例を示す図である。図１０を参照して、ユーザインターフェイス画面３００は、ソースコードを作成するための編集領域３０２と、編集領域３０２において作成されたソースコードのコンパイルを開始するためのコンパイルボタン３０４と、エラーメッセージを表示するためのメッセージ表示領域３０６とを含む。制約に適合しない旨のメッセージは、メッセージ表示領域３０６に表示されてもよい。

【００８４】
再度図９を参照して、選択された制約のスコープに含まれるソースコードが当該制約に適合するか否かを判断できなければ（ステップＳ１０６において「不明」）、ソフトウェア開発装置１００は、ソースコードが当該制約に適合するか否かを判断できない部分をマークする（ステップＳ１１２）。そして、処理はステップＳ１１４へ進む。
【００８５】
選択された制約のスコープに含まれるすべてのソースコードが当該制約に適合していれば（ステップＳ１０６においてＹＥＳ）、ソフトウェア開発装置１００は、設定されているすべての制約についての評価が完了したか否かを判断する（ステップＳ１１４）。設定されている制約のうち評価が完了していないものが残っていれば（ステップＳ１１４においてＮＯ）、ソフトウェア開発装置１００は、未評価の制約のうち１つを選択し（ステップＳ１１６）、ステップＳ１０６以下の処理を実行する。
【００８６】
設定されているすべての制約についての評価が完了していれば（ステップＳ１１４においてＹＥＳ）、ソフトウェア開発装置１００は、オブジェクトコード１１６を生成する（ステップＳ１１８）。
【００８７】
続いて、ソフトウェア開発装置１００は、ソースコードが当該制約に適合するか否かを判断できない部分がマークされているか否かを判断する（ステップＳ１２０）。すなわち、上述のステップＳ１１２においていずれかの部分がマークされているか否かが判断される。
【００８８】
ソースコードが当該制約に適合するか否かを判断できない部分がマークされていなければ（ステップＳ１２０においてＮＯ）、ステップＳ１２２の処理はスキップされる。これに対して、ソースコードが当該制約に適合するか否かを判断できない部分がマークされていれば（ステップＳ１２０においてＹＥＳ）、ソフトウェア開発装置１００は、マークされた部分が実行時において設定された制約に適合するか否かを判断するための適合評価用オブジェクトコード１１８を生成する（ステップＳ１２２）。このように、ソフトウェア開発装置１００は、制約のスコープにおいて、ソースコード１１２が制約に適合しているか否かを評価できない場合に、ソースコード１１２に対応するオブジェクトコード１１６に加えて、オブジェクトコード１１６の実行中に、当該制約に適合しているか否かを評価するための別のオブジェクトコード（適合評価用オブジェクトコード１１８）を生成する。
【００８９】
最終的に、ソフトウェア開発装置１００は、生成したオブジェクトコードを出力する（ステップＳ１２４）。すなわち、ソフトウェア開発装置１００は、ソースコード１１２に含まれる制約に適合するようにオブジェクトコードを生成する。そして、処理は終了する。
【００９０】
＜Ｇ．変形例＞
上述の説明においては、説明の便宜上、１つの制約を設定する場合について例示したが、これに限らず複数の制約を重ねて設定することもできる。また、複数の制約をそれぞれのスコープを互いに一部重複させて設定することもできる。
【００９１】
また、上述の説明においては、ソースコード１１２に制約コードを埋め込む構成例について例示したが、これに限らず、ソースコード１１２とは別に制約を規定する定義ファイルを用意にしてもよい。この場合には、定義ファイルには、制約のスコープとなるプロシージャ名やファンクション名を特定する情報と、適用される制約の内容とを関連付けて含めるようにしてもよい。
【００９２】
このように、制約の設定方法については、任意の方法を採用できる。
＜Ｈ．利点＞

本実施の形態に従うソフトウェア開発装置１００によれば、エッジデバイスなどで実行されるプログラムに対して、様々な制約を自在に設定できる環境を提供できる。これによって、利用可能なリソースおよびセキュリティの観点から様々な点を考慮した上で、エッジデバイスで実行されるプログラムを作成できる。

【００９３】

今回開示された実施の形態はすべての点で例示であって制限的なものでないと考えられるべきである。本発明の範囲は上記した説明ではなくて請求の範囲によって示され、請求の範囲と均等の意味および範囲内でのすべての変更が含まれることが意図される。

【符号の説明】

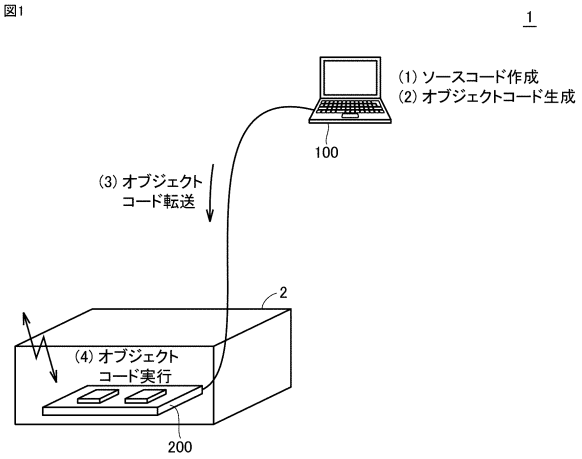
【００９４】

１　ＩｏＴシステム、２　エッジデバイス、１００　ソフトウェア開発装置、１０２，２０２　プロセッサ、１０４，２０４　メインメモリ、１０６　入力部、１０８　ディスプレイ、１１０　ハードディスク、１１２　ソースコード、１１４　ソフトウェア開発プログラム、１１６　オブジェクトコード、１１８　適合評価用オブジェクトコード、１２０　内部バス、１２２　通信インターフェイス、１５０，１５４　データセット、１５２，１５６　管理オブジェクト、２００　コントローラ、２０６　フラッシュメモリ、２１０　演算処理部、２１２　無線通信モジュール、２１４　ＵＳＢコントローラ、２１６　通信コントローラ、２１８　ドライバ、２２０　パッド、３００　ユーザインターフェイス画面、３０２　編集領域、３０４　コンパイルボタン、３０６　メッセージ表示領域、１１２１，１１２７　定義、１１２３，１１２４，１１２９，１１３０　命令、１１２５　制約コード、１１２６　部分ソースコード、１１２８　処理、１１４１　プリプロセッサ、１１４２　コンパイラ、１１４３　オプティマイザ、１１４４　コードジェネレータ。

【図面】

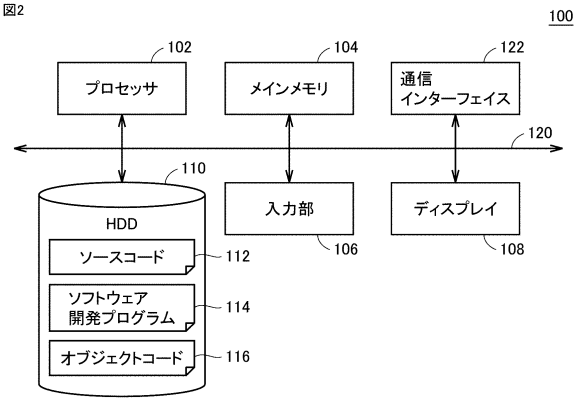
【図１】

図1



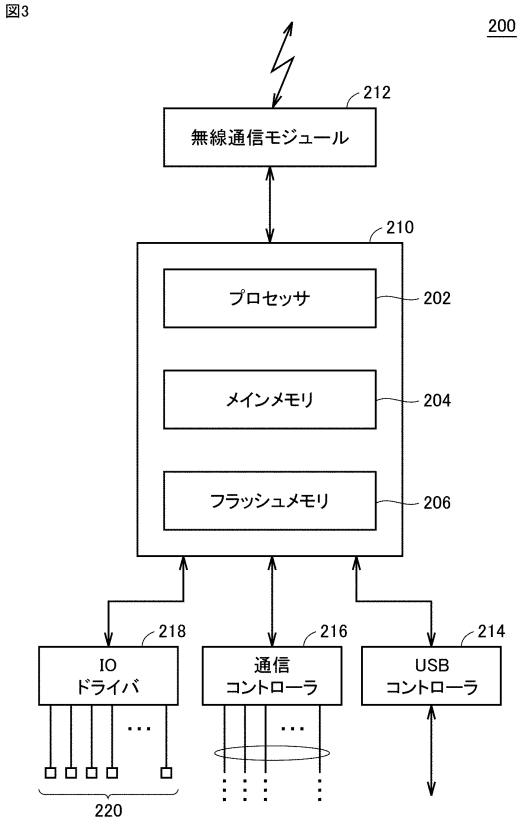
【図２】

図2



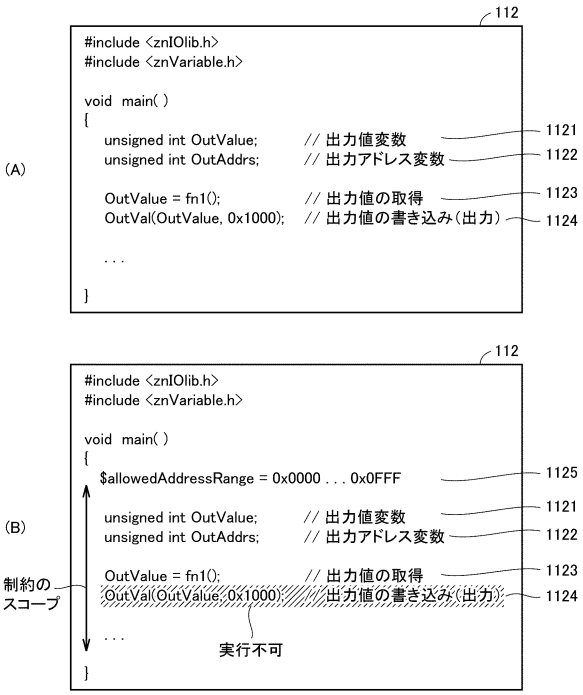
【図３】

図3



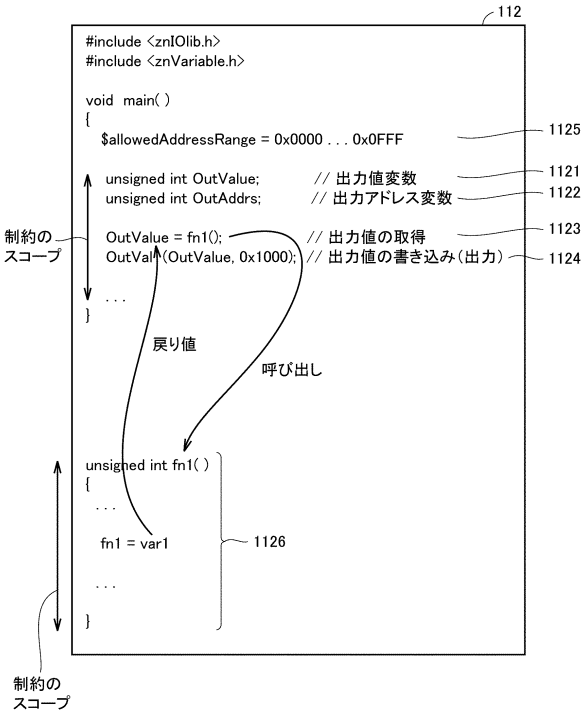
【図４】

図4



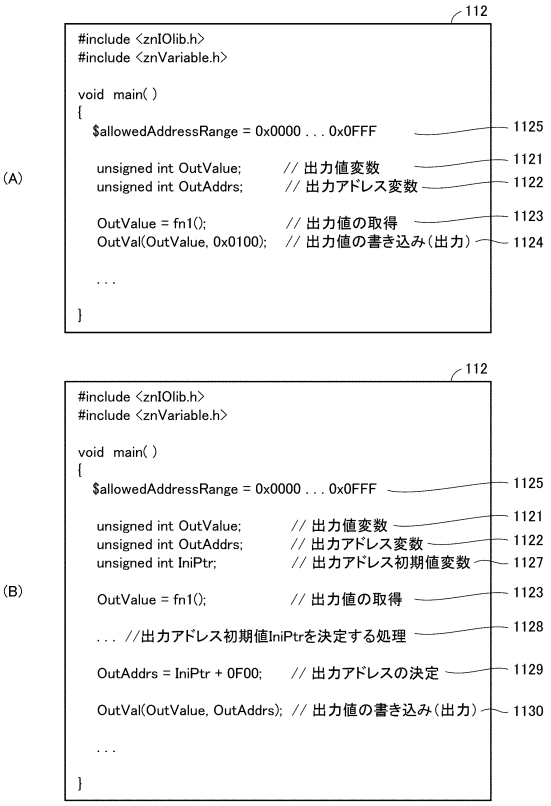
【図５】

図5



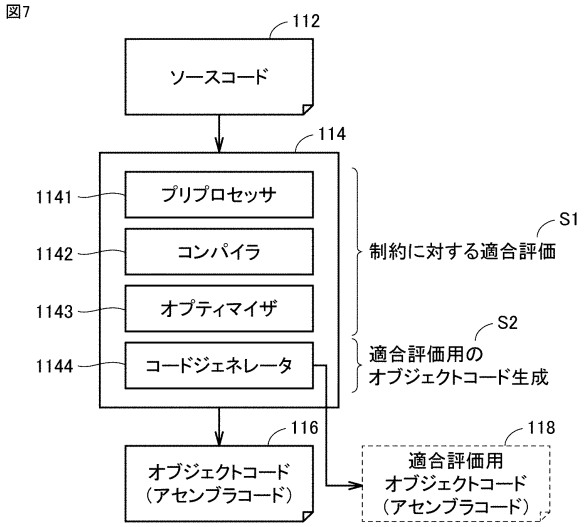
【図６】

図6



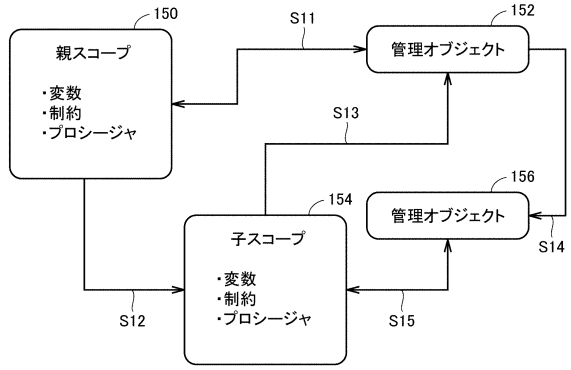
【図７】

図7



【図８】

図8



【図９】

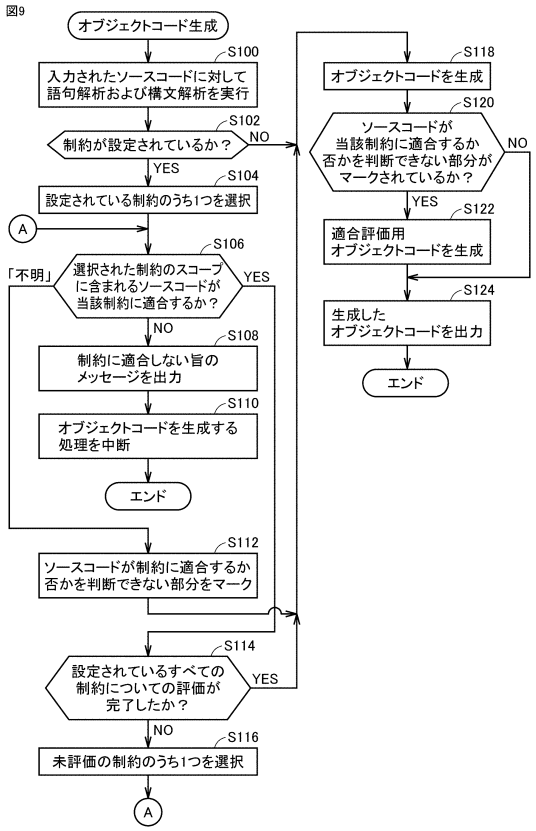


図9
オブジェクトコード生成
S100
入力されたソースコードに対して語句解析および構文解析を実行
S102
制約が設定されているか？
NO
YES
S104
設定されている制約のうち1つを選択
A
S106
「不明」
選択された制約のスコープに含まれるソースコードが当該制約に適合するか？
YES
NO
S108
制約に適合しない旨のメッセージを出力
S110
オブジェクトコードを生成する処理を中断
エンド
S112
ソースコードが制約に適合するか否かを判断できない部分をマーク
S114
設定されているすべての制約についての評価が完了したか？
YES
NO
S116
未評価の制約のうち1つを選択
A
S118
オブジェクトコードを生成
S120
ソースコードが当該制約に適合するか否かを判断できない部分がマークされているか？
NO
YES
S122
適合評価用オブジェクトコードを生成
S124
生成したオブジェクトコードを出力
エンド

【図１０】

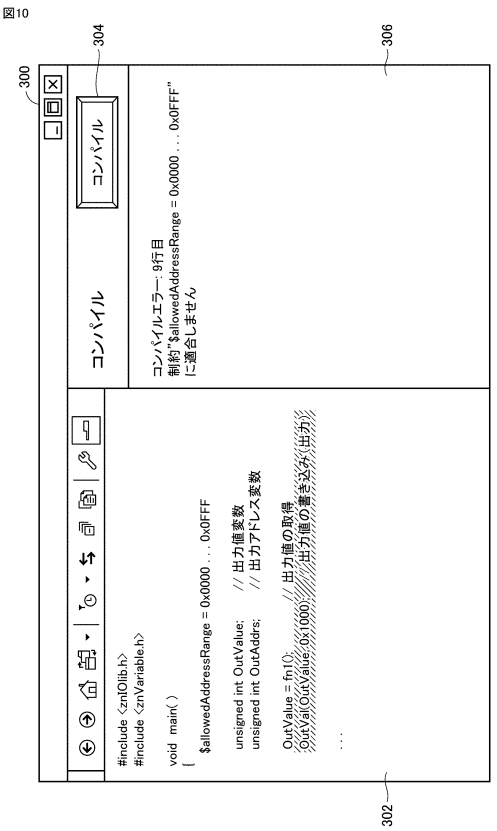


図10
300
304
306
コンパイル
コンパイル
コンパイルエラー: 9行目
制約"$allowedAddressRange = 0x0000 . . . 0x0FFF"
に適合しません
#include <znIOlib.h>
#include <znVariable.h>
void main( )
{
$allowedAddressRange = 0x0000 . . . 0x0FFF
unsigned int OutValue; // 出力値変数
unsigned int OutAddrs; // 出力アドレス変数
OutValue = fn1();
OutVal(OutValue,0x1000); // 出力値の書き込み(出力)
. . .
302

フロントページの続き

(56)参考文献　特開２００６－２８５５８２（ＪＰ，Ａ）
特開２０１３－１４０５１３（ＪＰ，Ａ）
特開２００７－１２２６３１（ＪＰ，Ａ）
米国特許出願公開第２０１６／０１４７５１１（ＵＳ，Ａ１）
米国特許出願公開第２０１７／０３１５９０３（ＵＳ，Ａ１）

(58)調査した分野　(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　９／４４
Ｇ０６Ｆ　８／３０，　８／４１，　８／７５
Ｇ０５Ｂ　１９／０４２