(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2008-226010
(P2008-226010A)
(43)公開日　平成20年9月25日(2008.9.25)

(51)Int.Cl.			F I			テーマコード（参考）
G06F	9/45	(2006.01)	G06F	9/44	320F	5B081
G06F	9/44	(2006.01)	G06F	9/44	322A	5B176
			G06F	9/06	620K	

審査請求 未請求　請求項の数 5　OL　（全 13 頁）

(21)出願番号　特願2007-65330(P2007-65330)
(22)出願日　平成19年3月14日(2007.3.14)

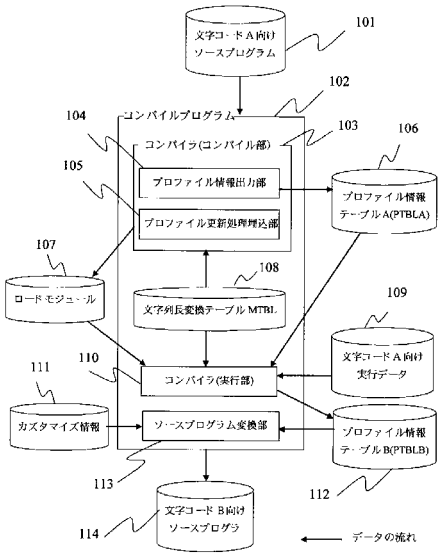
(71)出願人　000005108
株式会社日立製作所
東京都千代田区丸の内一丁目６番６号
(74)代理人　110000442
特許業務法人　武和国際特許事務所
(72)発明者　下坂　健則
神奈川県横浜市戸塚区戸塚町５０３０番地
株式会社日立製作所ソフトウェア事業部内
(72)発明者　小西　正洋
神奈川県横浜市戸塚区戸塚町５０３０番地
株式会社日立製作所ソフトウェア事業部内

Fターム(参考)　5B081 AA02　CC01　CC13
5B176 EA19

(54)【発明の名称】コンパイル方法及びコンパイル装置

(57)【要約】
【課題】文字コードが異なることに起因する実行データの性質の差異を気にすることなく、文字コードの異なるソースプログラム間でのプログラムの変更を行うこと。
【解決手段】固定長文字コードによるソースプログラム101を可変長文字コードによるソースプログラム114に変換するコンパイル装置であり、プロファイル情報を解析するプロファイル情報出力部104と、ロードモジュール107を生成する過程で、ロードモジュールの実行時に必要なプロファイル更新処理を埋め込むプロファイル更新処理埋込部105を含み、プロファイル情報出力部104と、ロードモジュール107の実行時にプロファイル更新処理によって更新されたプロファイル情報テーブル106をもとに、異なる文字コードとターゲットとする実行データ群とに最適に対応したソースプログラム114に変換する。
【選択図】図１

図１



【特許請求の範囲】
【請求項１】
第１の文字コードに対応したソースプログラムを第２の文字コードに対応したソースプログラムに変換する情報処理装置におけるコンパイル方法において、
前記情報処理装置内に、プロファイル情報出力手段と、プロファイル更新処理埋込手段と、コンパイル実行手段とを構成し、
前記プロファイル情報出力手段は、前記第１の文字コードに対応したソースプログラムのプロファイル情報を解析して第１のプロファイル情報を生成して出力し、
前記プロファイル更新処理埋込手段は、ロードモジュールの実行時に必要なプロファイル更新処理を、前記第１の文字コードに対応したソースプログラムからロードモジュールを生成する過程で埋め込み、
前記コンパイル実行手段は、前記ロードモジュールを実行し、該ロードモジュールの実行時に実行されるプロファイル更新処理は、前記第１のプロファイル情報を更新して第２のプロファイル情報を生成して出力することを特徴とするコンパイル方法。
【請求項２】
前記情報処理装置内に、ソースプログラム変換手段をさらに備え、該ソースプログラム変換手段は、前記第２のプロファイル情報に基づいて、前記第１の文字コードに対応したソースプログラムを前記第２の文字コードに対応したソースプログラムに変換することを特徴とする請求項１記載のコンパイル方法。
【請求項３】
前記ソースプログラム変換手段は、前記第２のプロファイル情報をカスタマイズすることにより、前記第１の文字コードに対応したソースプログラムを前記第２の文字コードに対応したソースプログラムに変換することを特徴とする請求項２記載のコンパイル方法。
【請求項４】
第１の文字コードに対応したソースプログラムを第２の文字コードに対応したソースプログラムに変換する情報処理装置において、
情報処理装置内に、プロファイル情報出力手段と、プロファイル更新処理埋込手段と、コンパイル実行手段と、ソースプログラム変換手段とを備え、
前記プロファイル情報出力手段は、前記第１の文字コードに対応したソースプログラムのプロファイル情報を解析して第１のプロファイル情報を生成して出力し、
前記プロファイル更新処理埋込手段は、ロードモジュールの実行時に必要なプロファイル更新処理を、前記第１の文字コードに対応したソースプログラムからロードモジュールを生成する過程で埋め込み、
前記コンパイル実行手段は、前記ロードモジュールを実行し、該ロードモジュールの実行時に実行されるプロファイル更新処理は、前記第１のプロファイル情報を更新して第２のプロファイル情報を生成して出力し、
前記ソースプログラム変換手段は、前記第２のプロファイル情報に基づいて、前記第１の文字コードに対応したソースプログラムを前記第２の文字コードに対応したソースプログラムに変換することを特徴とするコンパイル装置。
【請求項５】
第１の文字コードに対応したソースプログラムを第２の文字コードに対応したソースプログラムに変換するために情報処理装置に実行させるコンパイルプログラムであって、
前記第１の文字コードに対応したソースプログラムのプロファイル情報を解析して第１のプロファイル情報を生成して出力するステップと、
ロードモジュールの実行時に必要なプロファイル更新処理を、前記第１の文字コードに対応したソースプログラムからロードモジュールを生成する過程で埋め込むステップと、
前記ロードモジュールを実行し、該ロードモジュールの実行時に実行されるプロファイル更新処理が、前記第１のプロファイル情報を更新して第２のプロファイル情報を生成して出力するステップと、
前記第２のプロファイル情報に基づいて、前記第１の文字コードに対応したソースプロ

グラムを前記第２の文字コードに対応したソースプログラムに変換するステップとを、前記情報処理装置に実行させることを特徴とするコンパイルプログラム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、コンパイル方法及びコンパイル装置に係り、特に、特に、マルチバイト文字を固定長で表す文字コード（以下、固定長文字コード）のデータによるソースプログラムを、マルチバイト文字を可変長で表す文字コード（以下、可変長文字コード）のデータによるソースプログラムに変換するコンパイル方法及びコンパイル装置に関する。

【背景技術】

【０００２】

近年、官公庁、民間企業の情報システムは、文字コードとして、国際標準になりつつあるＵｎｉｃｏｄｅを採用する例が増加している。従来、日本国内の情報システムは、文字コードとしてシフトＪＩＳ（マルチバイト文字を固定長文字コードで表す）を採用することが多かった。そのため、国内の情報システムにおけるシフトＪＩＳに対応したプラットフォームからＵｎｉｃｏｄｅを採用したプラットフォームに移行させること事例が増加している。

【０００３】

Ｕｎｉｃｏｄｅの代表的なエンコーディングに、ＵＴＦ－８と呼ばれるものがあるが、ＵＴＦ－８のマルチバイト文字は、２バイト固定長のシフトＪＩＳと異なり、２バイトから６バイトの可変長で表現される。例えば、ＣＯＢＯＬ等で書かれたマルチバイト文字を多用する業務プログラムは、従来のシフトＪＩＳ対応からＵＴＦ－８対応に移行する場合、ソースプログラムに多量の修正（変換）を必要とする。このソースプログラムの修正方法、修正量は、業務プログラムが実行に使うデータ群に依存する。

【０００４】

ソースプログラムを効率よく編集、修正する方法として、コンパイラ要素技術を利用することがあるが、多くの場合、構文解析をベースとした方法が利用されている。この種の方法に関する従来技術として、例えば、特許文献１等に記載された技術が知られている。この従来技術は、ソフトウェアベンダーが規定しているコーディング規則等の規約情報に基づいて、ソースプログラムに対して構文解析を実施し、その解析結果をもとにソースプログラムを修正するというものである。

【特許文献１】特開２００６－２３６０４２号公報

【発明の開示】

【発明が解決しようとする課題】

【０００５】

前述した従来技術は、実行に使用するデータ群により、修正方法、修正量が変わる場合には、対応することができないという問題点を有している。そして、文字コードが異なるプラットフォーム間の移行では、実行するデータ群によっては、ほとんど修正がいらないケースもあるため、ソースプログラムを効率的に修正するためには、実行データの性質を有効に利用する必要がある。

【０００６】

本発明の目的は、前述したような点に鑑み、シフトＪＩＳからＵＴＦ－８というような文字コードが異なるプラットフォーム間の移行において、ソースプログラムを効率的に修正（変換）することができるコンパイル方法及びコンパイル装置を提供することにある。

【課題を解決するための手段】

【０００７】

本発明によれば前記目的は、第１の文字コードに対応したソースプログラムを第２の文字コードに対応したソースプログラムに変換する情報処理装置におけるコンパイル方法において、前記情報処理装置内に、プロファイル情報出力手段と、プロファイル更新処理埋込手段と、コンパイル実行手段とを構成し、前記プロファイル情報出力手段は、前記第１

の文字コードに対応したソースプログラムのプロファイル情報を解析して第１のプロファイル情報を生成して出力し、前記プロファイル更新処理埋込手段は、ロードモジュールの実行時に必要なプロファイル更新処理を、前記第１の文字コードに対応したソースプログラムからロードモジュールを生成する過程で埋め込み、前記コンパイル実行手段は、前記ロードモジュールを実行し、該ロードモジュールの実行時に実行されるプロファイル更新処理は、前記第１のプロファイル情報を更新して第２のプロファイル情報を生成して出力することにより、さらに、前記情報処理装置内に、ソースプログラム変換手段をさらに備えし、該ソースプログラム変換手段は、前記第２のプロファイル情報に基づいて、前記第１の文字コードに対応したソースプログラムを前記第２の文字コードに対応したソースプログラムに変換することにより達成される。

【発明の効果】

【０００８】

本発明によれば、文字コードが異なる新しいプラットフォームへの移行に際して、文字コードが異なることに起因する実行データの性質の差異を気にすることなく、ターゲットとする実行データ群に対応して、文字コードの異なるソースプログラム間でのプログラムの変更を行うことができる。

【発明を実施するための最良の形態】

【０００９】

以下、本発明によるコンパイル方法及びコンパイル装置の実施形態を図面により詳細に説明する。

【００１０】

図１は本発明の一実施形態によるコンパイル装置の機能構成を示すブロック図である。ここで説明する本発明の実施形態は、固定長文字コード（図面では、文字コードＡ）によるプログラムで書かれたソースプログラム１０１を、可変長文字コード（図面では、文字コードＢ）によるソースプログラムに変換するものとしている。また、ここでは、ＣＯＢＯＬ言語を用いたソースプログラムを例とする。なお、本発明の実施形態でのコンパイルプログラムは、文字コードＡ向けソースプログラム１０１だけを取り扱うことが可能なものであるとする。

【００１１】

本発明の実施形態によるコンパイル装置は、図１に示すように、コンパイル装置としての機能を実現するためのコンパイルプログラム１０２により構成される。このコンパイルプログラム１０２は、コンパイラ（コンパイル部）１０３と、コンパイラ（実行部）１１０と、ソースプログラム変換部１１３と、文字列長変換テーブル（ＭＴＢＬ）１０８とを有して構成されている。コンパイラ（コンパイル部）１０３は、プロファイル情報出力部１０４とプロファイル更新処理埋込部１０５とを有している。

【００１２】

そして、コンパイルプログラム１０２は、全体として、文字コードＡ向けソースプログラム１０１、文字コードＡ向け実行データ１０９及びカスタマイズ情報１１１を入力とし、プロファイル情報テーブルＢ１１２と文字コードＢ向けソースプログラム１１４とを出力する処理を実行する。

【００１３】

コンパイラ（コンパイル部）１０３は、文字コードＡ向けソースプログラム１０１と文字列長変換テーブル１０８とを入力とし、プロファイル情報テーブルＡ１０６とロードモジュール１０７とを出力する。詳細には、プロファイル情報出力部１０４は、文字コードＡ向けソースプログラム１０１を入力とし、プロファイル情報テーブルＡ１０６を出力する。また、プロファイル更新処理埋込部１０５は、コンパイラ（コンパイル部）１０３が、文字コードＡ向けソースプログラム１０１からロードモジュール１０７を生成する際に、プロファイル解析可能な処理を文字コードＡ向けソースプログラム１０１に埋め込んで、ロードモジュール１０７を生成する。本発明の本実施形態では、ソースプログラム中の手続き処理で更新されている変数（以下、「更新変数」と呼ぶ）またはファイル読み込み

を含む処理の直後に、「更新変数」またはファイル変数に対応したプロファイル解析処理（図６により後述する処理）を埋め込むこととしている。すなわち、ファイル読み込みを含む処理（図７の例では１８行目の処理）の場合、埋め込まれるファイル変数全てに対応したプロファイル解析処理を埋め込む。

【００１４】

図９はコンパイルプログラム１０２が文字コードＡ向けソースプログラム１０１をコンパイルした直後に生成したプロファイル情報テーブルＡ１０６の構成例を示す図である。このプロファイル情報テーブルＡ１０６は、文字コードＡ向けソースプログラム１０１を文字コードＢ向けソースプログラム１１４に変換するのに必要な情報を持つ。図９には、入力項目の一例を示しており、各項目はいずれも、文字コードＡ向けソースプログラム１０１に現れるものとする。

【００１５】

図９に示すプロファイル情報テーブル１０６は、項番、変数名、行番号、属性、定義長、更新レベル、最大長、ファイル名、レコード名の各項目からなる複数のレコードを有して構成される。このテーブルにおいて、「項番」は、テーブル１０６に格納されるレコードに順に付与された番号であり、「変数名」は、文字コードＡ向けソースプログラム１０１から抽出された変数名、「行番号」は、その変数が存在する文字コードＡ向けソースプログラム１０１の行番号である。

【００１６】

また、図９の項目のうち、「属性」は、変数の属性を示す。図９に示す例では、「Ｘ」は、英数字項目、図示していないが「Ｎ」は日本語項目を示す。「定義長」は、変数が定義されている長さ（単位はバイト）を示す。「更新レベル」は、コンパイラ（コンパイル部）１０３とコンパイラ（実行部）１１０とのどちらで当該変数の修正内容を決めるかを判別するフラグである。図示例では、「０」のときは、コンパイラ（コンパイル部）１０３で、「１」のときは、コンパイラ（実行部）１１０で、当該変数の修正内容を決めることを示している。「最大長」は、当該変数を更新する文字列群の中で文字コードＢに変換したときに最大となる長さを示す。「ファイル」は、ファイル変数が属するプログラム上のファイル名を示し、図９に示す例では、後述する文字コードＡ向けソースプログラム１０１の９行目で定義された「ＦＯＯ」が該当する。「レコード」は、ファイル変数が属するプログラム上のレコード名を示し、図９に示す例では、文字コードＡ向けソースプログラム１０１の１０行目で定義された「ＦＯＯＲ」が該当する。

【００１７】

図４はコンパイルプログラム１０２がアクセスする文字列変換テーブル１０８の構成例を示す図である。文字列長変換テーブル１０８は、コンパイルプログラム１０２で取り扱い可能な文字に対して、文字コードＡの場合の長さと文字コードＢの場合の長さとを示している。図４に示す例では、「項番」、「文字」、「長さ（文字コードＡ）」、「長さ（文字コードＢ）」の各項目からなるレコードにより、文字毎の文字コードＡの場合の長さと文字コードＢの場合の長さとを示している。

【００１８】

コンパイラ（実行部）１１０は、プロファイル情報テーブルＡ１０６、ロードモジュール１０７、文字列長変換テーブル１０８を入力とし、プロファイル情報テーブルＢ１１２を出力する。また、ソースプログラム変換部１１３は、文字コードＡ向けソースプログラム１０１、カスタマイズ情報１１１、プロファイル情報テーブルＢ１１２を入力として、文字コードＢ向けソースプログラム１１４を出力する。カスタマイズ情報１１１は、プロファイル情報テーブルＢ１１２の内容を加工する情報を有するもので、よく知られているように、定義長を設定し直すために利用される。

【００１９】

図２はプロファイル情報出力部１０４の構成を示すブロック図である。プロファイル情報出力部１０４は、データ定義処理部２０１と手続き処理部２０２とを有して構成されている。データ定義処理部２０１は、文字コードＡ向けソースプログラム１０１と文字列長

変換テーブル１０８とを入力として、プロファイル情報テーブルＡ１０６を出力する。手続き処理部２０２は、文字コードＡ向けソースプログラム１０１、データ定義処理部２０１から出力されたプロファイル情報テーブルＡ１０６と文字列長変換テーブル１０８とを入力として、プロファイル情報テーブルＡ１０６を更新する。

【００２０】

図７は文字コードＡ向けソースプログラム１０１の例を示す図、図８は文字コードＡ向けの実行データの例１０９を示す図であり、ここで、本発明の実施形態で変換しようとしている文字コードＡ向けソースプログラム１０１について説明する。

【００２１】

文字コードＡ向けソースプログラム１０１は、図７に示すように、各行に順に行番号が付与され、行番号とその行での処理の内容等を示す情報とを１つの行として、多数の行が行番号順に記述されて構成されている。そして、このソースプログラム１０１は、７行目から１５行目までのデータ定義部７０１と、１６行目から２５行目までの手続き部７０２とを持って構成されている。

【００２２】

このようなソースプログラム１０１に含まれる変数により決まる実行データは、図８に示すように、「data.txt」というファイル名で定義されており、図８に示す例では、ソースプログラム１０１内の変数Ａ１、Ａ２のそれぞれが、実行データとして、「ＸｐＸｓＸｕ」、「ＸｐＸｕＸｕ」という文字列で定義され、ソースプログラム１０１は、「data.txt」というファイル名で、この実行データ１０９を読み込む。

【００２３】

図３はデータ定義処理部２０１での処理動作を説明するフローチャートであり、次に、これについて説明する。

【００２４】

（１）データ定義処理部２０１は、処理を開始すると、文字コードＡ向けソースプログラム１０１のデータ定義部７０１の変数を行単位に検索していく。変数の検索で、変数がヒットしたら、その変数がファイル変数であるか否かを判別する。変数の検索及びその変数がファイル変数であるか否かの判別は、各行を構文解析することにより、その行に含まれる変数を抽出して、判別する処理である（ステップ３０１、３０２）。

【００２５】

（２）ステップ３０２の判別で、変数がファイル変数であった場合、その変数に依存した内容である「行番号」、「属性」、「定義長」をプロファイル情報テーブルＡ１０６に登録すると共に、対応する「ファイル」、「レコード」が存在するので、それらの名前を記入する。また、「更新レベル」には「１」、「最大長」には「０」を入力する（ステップ３０３）。

【００２６】

（３）ステップ３０２の判別で、変数がファイル変数でなかった場合、ステップ３０３での処理の場合と同様に、その変数に依存した内容である「行番号」、「属性」、「定義長」をプロファイル情報テーブルＡ１０６に登録すると共に、「ファイル」、「レコード」には「ＮＵＬＬ」を入力し、「更新レベル」には「０」、「最大長」には、初期値がある場合、文字列長変換テーブル１０８を参照して、文字コードＢに変換したときの文字列長を計算してセットする。また、初期値がない場合、「最大長」には、「０」をセットする（ステップ３０４）。

【００２７】

（４）ステップ３０３の処理、または、ステップ３０４の処理により、当該変数のプロファイル情報テーブル１０６への登録が完了した後、文字コードＡ向けソースプログラム１０１のデータ定義部７０１の処理が終了しているか否かを判別し、終了していなかった場合、次の変数を検索して、ステップ３０２からの処理に戻って処理を続け、終了していた場合、ここでの処理を終了する（ステップ３０５、３０６）。

【００２８】

図５は手続き処理部２０２での処理動作を説明するフローチャートであり、次に、これについて説明する。

【００２９】

（１）手続き処理部２０２は、処理を開始すると、文字コードＡ向けソースプログラム１０１の手続き部７０２の先頭行（図７では、１６行目）から順に処理を進めることとし、変数Ｋを手続き部７０２の先頭行Ｊに設定する（ステップ５０１）。

【００３０】

（２）次に、処理対象となっている行に変数が存在しているか否かを判別し、変数が存在していた場合、当該変数が参照だけの変数であるかを判定し、参照だけの変数であれば何もしない（ステップ５０２、５０３）。

【００３１】

（３）ステップ５０２の判断で、変数が存在していて、参照だけの変数ではなかった場合、その変数が変数で更新されるものであるかを判定し、変数で更新されるものであれば、プロファイル情報テーブルＡ１０６の更新レベルに「１」をセットする（ステップ５０４）。

【００３２】

（４）また、ステップ５０２の判断で、変数が存在していて、参照だけの変数ではなく、変数で更新されるものもなかった場合、次に、当該変数が定数で更新されるものであるかを判定し、定数で更新されるものであれば、文字列長変換テーブル１０８を参照して、定数を文字コードＢに変換したときの文字列長を計算し、プロファイル情報テーブルＡ１０６の当該変数の最大長欄よりも大きい値の場合、最大長欄の値を更新する（ステップ５０５）。

【００３３】

（５）ステップ５０５での処理の終了後、あるいは、ステップ５０２の判断で、該当行に変数が存在していなかった場合、処理を行った手続き部の行が最終行であるか否かを判定し、最終行でなかった場合、変数ＫをＫ＋１として次の行を処理することとして、ステップ５０２からの処理に戻って処理を続け、最終行であった場合、ここでの処理を終了する（ステップ５０６、５０７）。

【００３４】

図３により説明したデータ定義処理部２０１での処理と、図５により説明した手続き処理部２０２での処理から判るように、最初に、データ定義処理部２０１の処理で作成されたプロファイル情報テーブルＡ１０６は、手続き処理部２０２での処理により更新されることになり、図９に示しているプロファイル情報テーブルＡ１０６は、手続き処理部２０２での処理により更新された後のものである。ちなみに、データ定義処理部２０１の処理で作成されたプロファイル情報テーブルＡ１０６は、項番「３」の更新レベルが「０」となっており、手続き処理部２０２での処理により「１」に更新されている。

【００３５】

また、ステップ５０３～５０５の各処理において、変数がどのようなものかを判断しているが、この判断は、各行の構文解析の結果から行うことができる。

【００３６】

図６はプロファイル更新処理埋込部１０５で文字コードＡ向けソースプログラム１０１に埋め込まれるプロファイル更新処理の動作を説明するフローチャートであり、次に、これについて説明する。ここでの処理は、文字コードＡ向けソースプログラム１０１からロードモジュール１０７を生成する際に、文字コードＡ向けソースプログラム１０１の手続き部７０２の中で、プロファイル情報テーブル１０６の「更新レベル」が「１」となっている変数を含む処理の直後に埋め込まれるプロファイル更新処理であり、コンパイラ（実行部）１１０がロードモジュール１０７を処理するときに実行される処理である。

【００３７】

（１）この処理が開始されると、埋め込まれたプロファイル更新処理は、まず、文字列長変換テーブル１０８を参照して、更新対象変数のデータを文字コードＢに変換したときの

長さを計算する（ステップ６０１）。
【００３８】
（２）ステップ６０１の処理で計算した文字列長が、プロファイル情報テーブルＡ１０６の当該変数に対応した最大長欄の値より大きいか否かを判別し、最大長欄の値より大きかった場合、最大長欄の数値を更新し、最大長欄の値より大きくなかった場合、なにもせずに、そのまま終了する（ステップ６０２、６０３）。
【００３９】
次に、図４に示す文字列変換テーブル１０８の例と、図７に示す文字コードＡ向けソースプログラム１０１の例と、図８に示す実行データ１０９の例と、図９に示すプロファイル情報テーブルＡ１０６の項目とを用いて、コンパイルプログラム１０２の一連の処理の流れを具体的に説明する。なお、以下では、前述したフローにおけるステップの番号をも付与して説明する。
【００４０】
まず、プロファイル情報出力部１０４のデータ定義処理部２０１は、文字コードＡ向けソースプログラム例１０１と文字列長変換テーブル１０８とを入力する。そして、データ定義処理部２０１は、文字コードＡ向けソースプログラム１０１の７行目から変数検索を開始し（ステップ３０１）、１１、１２、１４、１５行目にそれぞれ変数Ａ１、Ａ２、Ｂ１、Ｂ２が見つかるので、順にプロファイル情報テーブルＡ１０６へその内容を登録していく。４つの変数のうち、Ａ１、Ａ２は、ファイル変数であることがわかるので（ステップ３０２）、プロファイル情報テーブルＡ１０６の「ファイル」、「レコード」の欄にそれぞれ「ＦＯＯ」、「ＦＯＯＲ」をセットする。また、「更新レベル」に「１」、「最大長」に「０」をセットする（ステップ３０３）。
【００４１】
Ｂ１、Ｂ２は、ファイル変数ではないと判断できるので、「ファイル」、「レコード」の欄には「ＮＵＬＬ」を登録する。また、Ｂ１は、初期値が定義されていないため、「最大長」の欄に「０」を登録する（ステップ３０４）。Ｂ２は、初期値「ＸｓＸｓＸｕＸｕ」が定義されているため、文字列長変換テーブル１０８を参照して、文字コードＢに対応した長さ（３＋３＋４＋４＝１４）を計算し、その計算結果の値を「最大長」の欄にセットする（ステップ３０４）。
【００４２】
次に、手続き処理部２０２は、文字コードＡ向けソースプログラム１０１とデータ定義処理部２０１で作成したプロファイル情報テーブルＡ１０６とを入力とし、文字コードＡ向けソースプログラム１０１の１６行目から処理を開始する（ステップ５０１）。そして、図５に示した処理を実行することにより、データ定義処理部２０１で作成したプロファイル情報テーブルＡ１０６の項番「３」の更新レベルが「０」から「１」に更新される。
【００４３】
一方、プロファイル更新処理埋込部１０５は、１８行目の直後にＡ１、Ａ２に対応した図６で示すプロファイル更新処理を埋め込み、１９行目の直後にＢ１に対応した図６で示すプロファイル更新処理を埋め込み、２１行目の直後にＢ１に対応した図６で示すプロファイル更新処理を埋め込む。
【００４４】
コンパイラ（コンパイル部）１０３での処理が完了すると、コンパイラ（実行部）１１０へ処理が遷移する。コンパイラ（実行部）１１０は、ロードモジュール１０７の実行を開始し、図８に示す文字コードＡ向け実行データ１０９をロードモジュール１０７に読み込む。最初に、Ａ１に実行データ「ＸｐＸｓＸｕ」が入り、プロファイル更新処理埋込部１０５で埋め込まれた処理により、文字列長変換テーブル１０８を参照して「ＸｐＸｓＸｕ」を文字コードＢに変換したときの長さを（１＋３＋４＝８）として計算する（ステップ６０１）。計算値「８」は、プロファイル情報テーブルＡ１０６の項番「１」の変数名Ａ１の最大長の欄にセットしてある値「０」よりも大きいので（ステップ６０２）、最大長の欄を「８」に更新する（ステップ６０３）。同様に、Ａ２の最大長の欄を「９」に更

新する。次に、Ｂ１のプロファイル更新処理を実行する。Ｂ１には、Ａ１の文字列「ＸｐＸｓＸｕ」が入ってくるので、Ａ１の処理と同様となり、Ｂ１の最大長の欄を「８」に更新する。２１行目では、Ｂ１を再度、Ａ２で更新しているため、Ｂ１のプロファイル更新処理を実行し、Ｂ１の最大長欄を「９」に更新する。以上で、文字コードＡ向けソースプログラム例１０１のプロファイル更新処理が完了する。

【００４５】

図１１はプロファイル更新処理完了後のプロファイル情報テーブルＢ１１２の構成例を示す図である。この図１１から判るように、各変数の最大長の欄に、文字コードＢに変換したときの長さの値が格納される。

【００４６】

その後、ソースプログラム変換部１１３は、カスタマイズ情報１１１、プロファイル情報テーブルＢ１１２をもとに、文字コードＡ向けソースプログラム１０１を文字コードＢ向けソースプログラム１１４に変換する。カスタマイズ情報１１１には、定数Ｔが格納されているものとする。ソースプログラム変換部１１３は、プロファイル情報テーブルＢ１１２の更新レベル欄「１」の変数の最大長欄の数値に対して、定数Ｔ（説明している例の場合、Ｔ＝１とする）を掛けて、プロファイル情報テーブルＢ１１２の最大長欄を更新する。

【００４７】

プロファイル情報テーブルＢ１１２の例は、変数Ａ１、Ａ２、Ｂ１、Ｂ２の４つの変数の全てにおいて、「定義長」よりも「最大長」が大きい結果となっているので、文字コードＡ向けソースプログラム１０１で定義されている長さを「最大長」の値に変換する。この変換後の結果が文字コードＢ向けソースプログラム１１４となる。以上で、コンパイルプログラム１０２の処理が完了する。

【００４８】

図１０は文字コードＢ向けソースプログラム１１４の例を示す図である。この文字コードＢ向けソースプログラム１１４では、文字コードＡ向けソースプログラム１０１における各変数の属性Ｘとして記述されている変数の定義長が変更されたものとなっている。

【００４９】

図１２は本発明の実施形態によるコンパイル装置が構築される情報処理装置のハードウェア構成を示すブロック図である。図１２に示すように、本発明の実施形態によるコンパイル装置を構築する情報処理システム１２０１は、プロセッサ１２０２、ハードディスクドライブ１２０３、メインメモリ１２０４を備えて構成される。メインメモリ１２０４には、コンパイルプログラム１０２が格納され、プロセッサ１２０２により実行されて、前述で説明したような処理を行う。ハードディスクドライブ１２０３には、ソースファイル１０１、プロファイル情報テーブル１０６、文字列長変換テーブル１０８が格納されている。

【００５０】

前述した本発明の実施形態での各処理は、すでに説明したように、プログラムにより構成し、本発明が備えるＣＰＵに実行させることができ、また、それらのプログラムは、ＦＤ、ＣＤＲＯＭ、ＤＶＤ等の記録媒体に格納して提供することができ、また、ネットワークを介してディジタル情報により提供することができる。

【図面の簡単な説明】

【００５１】

【図１】本発明の一実施形態によるコンパイル装置の機能構成を示すブロック図である。

【図２】プロファイル情報出力部の構成を示すブロック図である。

【図３】データ定義処理部での処理動作を説明するフローチャートである。

【図４】コンパイルプログラムがアクセスする文字列長変換テーブルの構成例を示す図である。

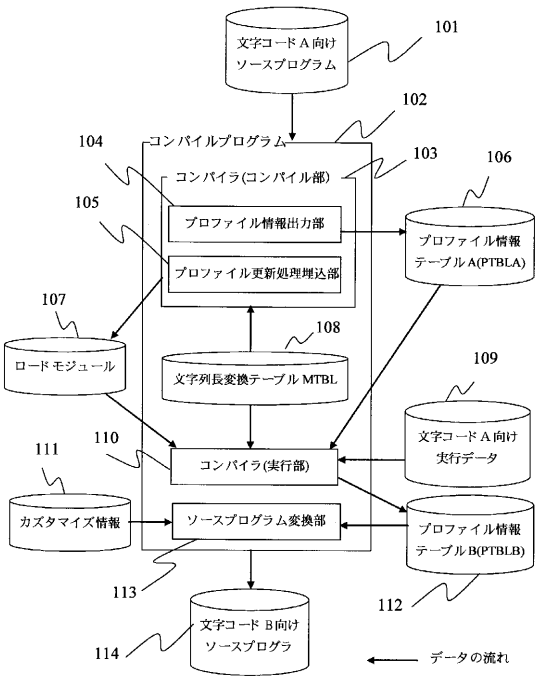
【図５】手続き処理部での処理動作を説明するフローチャートである。

【図６】プロファイル更新処理埋込部で文字コードＡ向けソースプログラムに埋め込まれ

るプロファイル更新処理の動作を説明するフローチャートである。
【図７】文字コードＡ向けソースプログラムの例を示す図である。
【図８】文字コードＡ向けの実行データの例を示す図である。
【図９】コンパイルプログラムが文字コードＡ向けソースプログラムをコンパイルした直後に生成したプロファイル情報テーブルＡの構成例を示す図である。
【図１０】文字コードＢ向けソースプログラムの例を示す図である。
【図１１】プロファイル更新処理完了後のプロファイル情報テーブルＢの構成例を示す図である。
【図１２】本発明の実施形態によるコンパイル装置が構築される情報処理システムのハードウェア構成を示すブロック図である。
【符号の説明】
【００５２】
１０１　文字コードＡ向けソースプログラム
１０２　コンパイルプログラム
１０３　コンパイラ（コンパイル部）
１０４　プロファイル情報出力部
１０５　プロファイル更新処理埋込部
１０６　プロファイル情報テーブルＡ
１０７　ロードモジュール
１０８　文字列長変換テーブル
１０９　文字コードＡ向け実行データ
１１０　コンパイラ（実行部）
１１１　カスタマイズ情報
１１２　プロファイル情報テーブルＢ
１１３　ソースプログラム変換部
１１４　文字コードＢ向けソースプログラム
２０１　データ定義処理部
２０２　手続き処理部
１２０１　情報処理システム
１２０２　プロセッサ
１２０３　ハードディスクドライブ
１２０４　メインメモリ

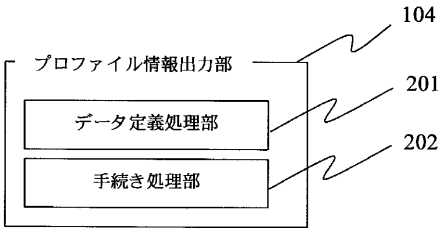
【図１】

図1



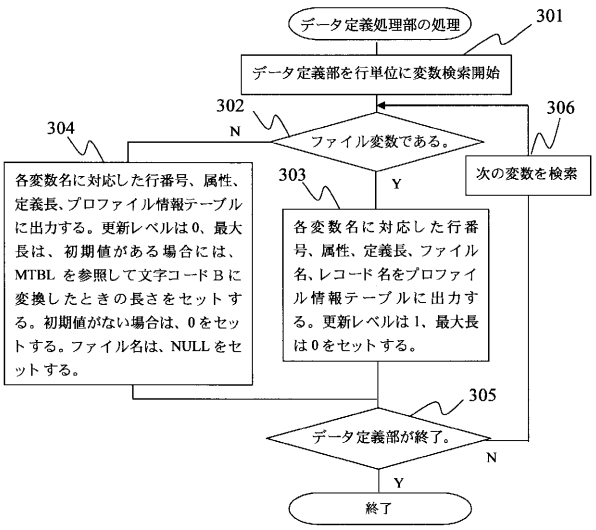
【図２】

図2



【図３】

図3



【図４】

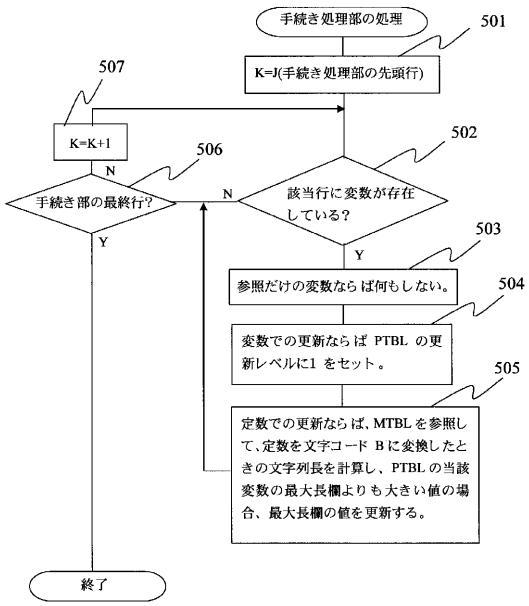
図4　文字列長変換テーブル MTBL

108

項番	文字	長さ(文字コードＡ、単位：バイト)	長さ(文字コードＢ、単位：バイト)
1	X_1	1	1
・	・	・	・
・	・	・	・
p	X_p	1	1
・	・	・	・
・	・	・	・
q	X_q	1	1
q+1	X_{q+1}	2	2
・	・	・	・
・	・	・	・
r	X_r	2	2
r+1	X_{r+1}	2	3
・	・	・	・
・	・	・	・
s	X_s	2	3
・	・	・	・
・	・	・	・
t	X_t	2	3
t+1	X_{t+1}	2	4
・	・	・	・
・	・	・	・
u	X_u	2	4
・	・	・	・
・	・	・	・
v	X_v	2	4

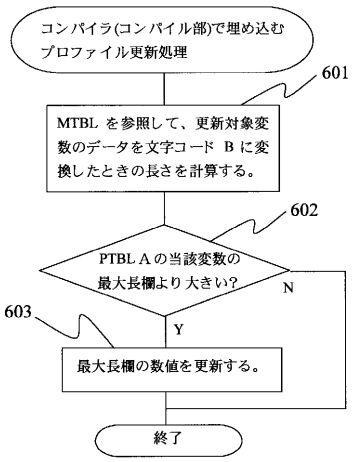
【図５】

図5



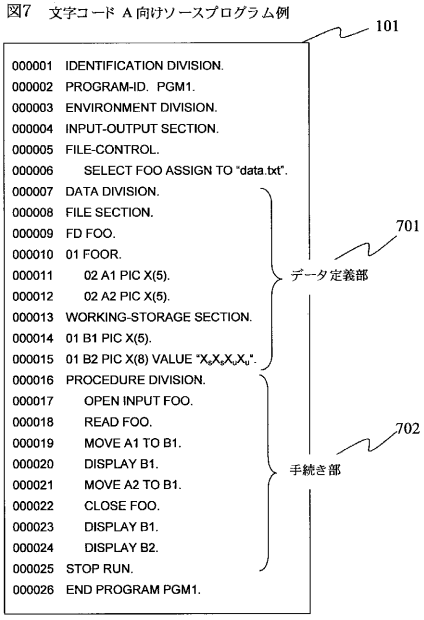
【図６】

図6



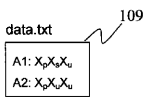
【図７】

図7　文字コード A 向けソースプログラム例



【図８】

図8　実行データ例



【図９】

図9　プロファイル情報テーブル A(PTBLA)(コンパイル直後の例)

106

項番	変数名	行番号	属性 (*1)	定義長	更新レベル (*2)	最大長	ファイル	レコード
1	A1	11	X	5	1	0	FOO	FOOR
2	A2	12	X	5	1	0	FOO	FOOR
3	B1	14	X	5	1	0	NULL	NULL
4	B2	15	X	8	0	14	NULL	NULL

(*1) ：X 英数字項目
：N 日本語項目

(*2) ：0 更新なし、もしくは定数での更新
：1 変数更新、ファイルデータでの更新

【図１０】

図1０ 文字コードＢ向けソースプログラム例

114

```
000001  IDENTIFICATION DIVISION.
000002  PROGRAM-ID.  PGM1.
000003  ENVIRONMENT DIVISION.
000004  INPUT-OUTPUT SECTION.
000005  FILE-CONTROL.
000006      SELECT FOO ASSIGN TO "data.txt".
000007  DATA DIVISION.
000008  FILE SECTION.
000009  FD FOO.
000010  01 FOOR.
000011      02 A1 PIC X(8).
000012      02 A2 PIC X(9).
000013  WORKING-STORAGE SECTION.
000014  01 B1 PIC X(9).
000015  01 B2 PIC X(14) VALUE "XuXuXuXu".
000016  PROCEDURE DIVISION.
000017      OPEN INPUT FOO.
000018      READ FOO.
000019      MOVE A1 TO B1.
000020      DISPLAY B1.
000021      MOVE A2 TO B1.
000022      CLOSE FOO.
000023      DISPLAY B1.
000024      DISPLAY B2.
000025  STOP RUN.
000026  END PROGRAM PGM1.
```

【図１１】

図1 1プロファイル情報テーブルB(PTBLB)(実行後の例)

112

項番	変数名	行番号	属性	定義長	更新レベル	最大長	ファイル	レコード
1	A1	11	X	5	1	8	FOO	FOOR
2	A2	12	X	5	1	9	FOO	FOOR
3	B1	14	X	5	1	9	NULL	NULL
4	B2	15	X	8	0	14	NULL	NULL

【図１２】

図１２