(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2005-293342
(P2005-293342A)
(43)公開日　平成17年10月20日(2005.10.20)

(51)Int.Cl.[7]	F I	テーマコード（参考）
G06F 15/80	G06F 15/80	5B081
G06F 9/45	G06F 9/44 322E	

審査請求 未請求　請求項の数 6　ＯＬ　（全 33 頁）

(21)出願番号　特願2004-108827 (P2004-108827)
(22)出願日　平成16年4月1日 (2004.4.1)

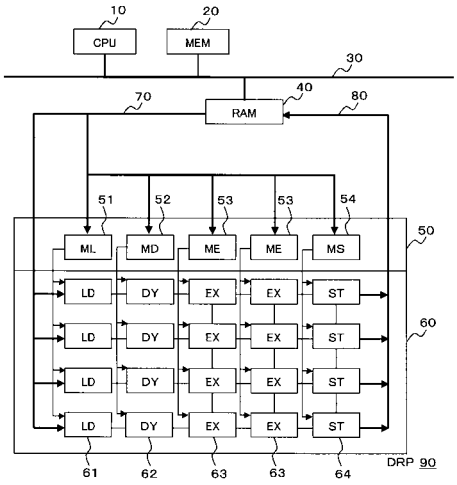
(71)出願人　503121103
株式会社ルネサステクノロジ
東京都千代田区丸の内二丁目４番１号
(74)代理人　110000198
特許業務法人湘洋内外特許事務所
(72)発明者　佐藤　真琴
神奈川県川崎市麻生区王禅寺１０９９番地
株式会社日立製作所システム開発研究所内

Ｆターム(参考)　5B081 CC32　CC41

(54)【発明の名称】動的再構成可能プロセッサおよびコンパイラ装置

(57)【要約】
【課題】動的再構成可能プロセッサ向けプログラムの開発にかかる負担を軽減する。
【解決手段】動的再構成可能プロセッサ９０は、M行N列（但し、M≧２、N≧１）に配列された、設定された構成情報に従って入出力先および演算内容を切り替え可能な演算ユニット６３と、１列目の演算ユニット６３毎に設けられた、当該演算ユニット６３へ入力するデータを局所メモリ４０からロードするロードユニット６１と、N列目の演算ユニット６３毎に設けられた、当該演算ユニット６３から出力するデータを局所メモリ４０にストアするストアユニット６３と、ロードユニット６１がロードしたデータの１列目の対応する演算ユニット６３へのデータ入力を遅延させる遅延ユニット６２と、を有する。
【選択図】図１

図1



【特許請求の範囲】
【請求項１】
設定された構成情報に従って入出力先および演算内容を切り替え可能な演算ユニットが、Ｍ行Ｎ列（但し、Ｍ≧２、Ｎ≧１）に配列されて構成された、動的再構成可能プロセッサであって、
設定された構成情報に従って１列目の演算ユニットへのデータ入力を遅延させる遅延ユニットを有すること
を特徴とする動的再構成可能プロセッサ。
【請求項２】
請求項１に記載の動的再構成可能プロセッサであって、
１列目の演算ユニット毎に設けられた、当該演算ユニットへ入力するデータをメモリからロードする、Ｍ個のロードユニットと、
Ｎ列目の演算ユニット毎に設けられた、当該演算ユニットから出力するデータを前記メモリにストアする、Ｍ個のストアユニットと、を備え、
少なくとも２つの前記ロードユニットは、同列の前記ストアユニットが互いに異なるサイクルで前記メモリにストアしたデータを、前記メモリからロードし、
前記遅延ユニットは、前記少なくとも２つのロードユニットがロードしたデータが前記異なるサイクルにより特定されるサイクル差で１列目の対応する演算ユニットに入力するように、当該対応する演算ユニットへのデータ入力を遅延させること
を特徴とする動的再構成可能プロセッサ。
【請求項３】
請求項１又は２記載の動的再構成可能プロセッサと、
前記動的再構成可能プロセッサに構成情報を設定すると共に、前記動的再構成可能プロセッサとデータを入出力するための局所メモリと、
前記動的再構成可能プロセッサに設定する構成情報および前記動的再構成可能プロセッサに投入するデータが記憶されたメモリと、
前記メモリに記憶された構成情報およびデータを前記局所メモリに転送するホストプロセッサと、を有すること
を特徴とする情報処理装置。
【請求項４】
動的再構成可能プロセッサを構成するＭ行Ｌ列の演算ユニットに設定する構成情報を含む入力プログラムを、請求項１又は２に記載の動的再構成可能プロセッサを構成する遅延ユニットおよびＭ行Ｎ（但しＮ＜Ｌ）列の演算ユニットに設定する構成情報を含む出力プログラムに変換するコンパイラ装置であって、
前記入力プログラムを記憶する入力プログラム記憶手段と、
前記入力プログラムに含まれる前記Ｍ行Ｌ列の演算ユニットの構成情報をＮ列毎に分割して、ｎｕｍ（但し、ｎｕｍはＭ/Ｎより大きい最小の整数）回分の前記Ｍ行Ｎ列の演算ユニットの構成情報（分割構成情報と呼ぶ）を生成する構成情報分割手段と、
ｕ回目の分割構成情報を前記Ｍ行Ｎ列の演算ユニットに設定した場合にＮ列目の演算ユニットから出力される各データの出力サイクルのサイクル差を算出する処理を、前記ｕが１からｎｕｍ-１となるまで繰り返す遅延解析手段と、
ｔ回目の分割構成情報が設定された前記Ｍ行Ｎ列の演算ユニットのＮ列目から出力される各データが、前記遅延解析手段により算出された当該各データの出力サイクルのサイクル差で前記Ｍ行Ｎ列の演算ユニットの１列目に入力されるように、前記遅延ユニットを制御するための構成情報を、Ｋ+１回目の分割構成情報に含める処理を、ｔが１からｎｕｍ-１となるまで繰り返して、ｎｕｍ回分の分割構成情報を含む出力プログラムを生成するプログラム出力手段と、を有すること
を特徴とするコンパイラ装置。
【請求項５】
コンピュータで読取り可能なプログラムであって、

前記プログラムは、動的再構成可能プロセッサを構成するＭ行Ｌ列の演算ユニットに設定する構成情報を含む入力プログラムを、請求項１又は２に記載の動的再構成可能プロセッサを構成する遅延ユニットおよびＭ行Ｎ（但しＮ＜Ｌ）列の演算ユニットに設定する構成情報を含む出力プログラムに変換するプログラムであり、

前記コンピュータに、

前記入力プログラムを記憶する入力プログラム記憶手段、

前記入力プログラムに含まれる前記Ｍ行Ｌ列の演算ユニットの構成情報をＮ列毎に分割して、ｎｕｍ（但し、ｎｕｍはＭ/Ｎより大きい最小の整数）回分の前記Ｍ行Ｎ列の演算ユニットの構成情報（分割構成情報と呼ぶ）を生成する構成情報分割手段、

ｕ回目の分割構成情報を前記Ｍ行Ｎ列の演算ユニットに設定した場合にＮ列目の演算ユニットから出力される各データの出力サイクルのサイクル差を算出する処理を、前記ｕが１からｎｕｍ-１となるまで繰り返す遅延解析手段、そして、

ｔ回目の分割構成情報が設定された前記Ｍ行Ｎ列の演算ユニットのＮ列目から出力される各データが、前記遅延解析手段により算出された当該各データの出力サイクルのサイクル差で前記Ｍ行Ｎ列の演算ユニットの１列目に入力されるように、前記遅延ユニットを制御するための構成情報を、Ｋ+１回目の分割構成情報に含める処理を、ｔが１からｎｕｍ-１となるまで繰り返して、ｎｕｍ回分の分割構成情報を含む出力プログラムを生成するプログラム出力手段として、機能させること

を特徴とするコンピュータで読取り可能なプログラム。

【請求項６】

コンパイラ装置が、動的再構成可能プロセッサを構成するＭ行Ｌ列の演算ユニットに設定する構成情報を含む入力プログラムを、請求項１又は２に記載の動的再構成可能プロセッサを構成する遅延ユニットおよびＭ行Ｎ（但しＮ＜Ｌ）列の演算ユニットに設定する構成情報を含む出力プログラムに変換する変換方法であって、

前記入力プログラムに含まれる前記Ｍ行Ｌ列の演算ユニットの構成情報をＮ列毎に分割して、ｎｕｍ（但し、ｎｕｍはＭ/Ｎより大きい最小の整数）回分の前記Ｍ行Ｎ列の演算ユニットの構成情報（分割構成情報と呼ぶ）を生成し、記憶手段に記憶する構成情報分割ステップと、

前記記憶手段に記憶された構成情報を読み出し、ｕ回目の分割構成情報を前記Ｍ行Ｎ列の演算ユニットに設定した場合にＮ列目の演算ユニットから出力される各データの出力サイクルのサイクル差を算出し、前記記憶手段に記憶する処理を、前記ｕが１からｎｕｍ-１となるまで繰り返す遅延解析ステップと、

前記記憶手段に記憶された構成情報およびサイクル差を読み出し、ｔ回目の分割構成情報が設定された前記Ｍ行Ｎ列の演算ユニットのＮ列目から出力される各データが、前記遅延解析手段により算出された当該各データの出力サイクルのサイクル差で前記Ｍ行Ｎ列の演算ユニットの１列目に入力されるように、前記遅延ユニットを制御するための構成情報を、Ｋ+１回目の分割構成情報に含める処理を、ｔが１からｎｕｍ-１となるまで繰り返して、ｎｕｍ回分の分割構成情報を含む出力プログラムを生成するプログラム出力ステップと、を行うこと

を特徴とする変換方法。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、動的再構成可能プロセッサおよび動的再構成可能プロセッサ向けプログラムのコンパイラ装置に関する。

【背景技術】

【０００２】

設定された構成情報に従って入出力先および演算内容を切り替え可能な演算ユニットが複数配列されて構成された動的再構成可能プロセッサ（ＤＲＰ：Dynamically Reconfigurable Processor）が知られている（例えば特許文献１）。

【０００３】
図２８は、従来の動的再構成可能プロセッサを有する情報処理装置の概略図である。
【０００４】
図示するように、情報処理装置は、ホストプロセッサ（ＣＰＵ）１００と、メモリ（ＭＥＭ）２００と、局所メモリ（ＲＡＭ）４００と、動的再構成可能プロセッサ（ＤＲＰ）９００と、を有する。ホストプロセッサ１００、メモリ２００および局所メモリ４００は、内部バス３００で互いに接続されている。また、局所メモリ４００および動的再構成可能プロセッサ９００は、局所メモリ４００から動的再構成可能プロセッサ９００へデータおよびプログラムを転送するための入力配線７００と、動的再構成可能プロセッサ９００から局所メモリ４００へデータを転送するための出力配線８００とを介して互いに接続されている。
【０００５】
メモリ２００には、ホストプロセッサ１００および動的再構成可能プロセッサ９００が使用するデータ、プログラムが格納される。ホストプロセッサ１００は、メモリ２００に記憶されたホストプロセッサ１００用のプログラムに従い、動的再構成可能プロセッサ９００の起動、終結、および、動的再構成可能プロセッサ９００へのデータ、プログラムの転送を制御する。局所メモリ４００は、動的再構成可能プロセッサ９００が使用するデータ、プログラムが格納される。動的再構成可能プロセッサ９００は、構成情報を格納する構成情報格納部５００と、演算部６００と、を有する。
【０００６】
演算部６００は、Ｍ行Ｌ列（図２８では４行４列）に配列された演算ユニット（ＥＸ）６３０と、１列目の演算ユニット６３０毎に設けられたＭ個（Ｍ行１列）のロードユニット（ＬＤ）６１０と、Ｌ列目の演算ユニット６３０毎に設けられたＭ個（Ｍ行１列）のストアユニット（ＳＴ）６４０と、を有する。
【０００７】
ロードユニット６１０は、設定された構成情報に従い、１列目の対応する（同行の）演算ユニット６３０へ入力するデータを局所メモリ４００からロードする。
【０００８】
演算ユニット６３０は、図示していないが、複数の入出力ポートと、四則演算等の各種演算を行う複数の演算回路と、入出力ポートおよび演算回路の接続を切替えるスイッチと、を有する。そして、演算ユニット６３０は、設定された構成情報に従い、スイッチを制御して、データの入力元、演算内容および演算結果の出力先を切り替える。
【０００９】
ストアユニット６４０は、設定された構成情報に従い、Ｎ列目の対応する（同行の）演算ユニット６３０から出力されるデータを局所メモリ４００へストアする。
【００１０】
構成情報格納部５００は、ロードユニット用構成情報メモリ（ＭＬ）５１０と、Ｍ行Ｌ列の演算ユニットの列毎に設けられたＬ個の演算ユニット用構成情報メモリ（ＭＥ）５３０と、ストアユニット用構成情報メモリ（ＭＳ）５４０と、を有する。
【００１１】
ロードユニット用構成情報メモリ５１０は、Ｍ個のロードユニット６１０各々に設定するロードユニット用構成情報（データをロードするアドレス等のパラメータ情報）を格納する。
【００１２】
演算ユニット用構成情報メモリ５３０は、対応する列のＭ個の演算ユニット６３０各々に設定する演算ユニット用構成情報（データ入出力先および演算内容を特定する情報）を格納する。
【００１３】
ストアユニット用構成情報メモリ５４０は、Ｍ個のストアユニット６４０各々に設定するストアユニット用構成情報（データをストアするアドレス等のパラメータ情報）を格納

する。
【００１４】
以上のような構成において、ホストプロセッサ１００は、動的再構成可能プロセッサ９００が使用するデータ、プログラムをメモリ２００から局所メモリ４００へ転送する。ここで、プログラムは、上述の、ロードユニット用構成情報、演算ユニット用構成情報、および、ストアユニット用構成情報を含む。
【００１５】
次に、ロードユニット用構成情報メモリ５１０は、局所メモリ４００からロードユニット用構成情報を読み込み、各ロードユニット６１０に構成情報を設定する。また、各演算ユニット用構成情報メモリ５３０は、対応する列の演算ユニット用構成情報を読込み、当該列の各演算ユニット６３０に構成情報を設定する。また、ストアユニット用構成情報メモリ５４０は、局所メモリ４００からストアユニット用構成情報を読み込み、各ストアユニット５４０に構成情報を設定する。
【００１６】
さて、以上のようにして演算部６００の各ユニットに構成情報が設定されると、先ず、ロードユニット６１０が、設定された構成情報に従い、局所メモリ４００からデータをロードし、これを１列目の対応する（同行の）演算ユニット６３０へ転送する。次に、演算ユニット６３０が、設定された構成情報に従い、データの入力元、演算内容および演算結果の出力先を切り替えて演算処理を行う。そして、ストアユニット６４０が、設定された構成情報に従い、Ｌ列目の対応する（同行）演算ユニット６３０から出力されたデータを局所メモリ４００にストアする。
【００１７】
このように、動的再構成可能プロセッサ９００では、プログラムによって複数の演算ユニット６３０各々の演算内容および演算順番を変更することができ、従って動的に機能を変更することができる。
【００１８】
【特許文献１】特開２００３-７６６６８号公報
【発明の開示】
【発明が解決しようとする課題】
【００１９】
ところで、動的再構成可能プロセッサ向けプログラムの開発は、動的再構成可能プロセッサ毎に行っている。つまり、Ｍ行Ｌ列の演算ユニットを有する動的再構成可能プロセッサ向けに開発されたプログラムを、演算ユニットの列数の少ないＭ行Ｎ列（但しＮ＜Ｌ）の演算ユニットを有する動的再構成可能プロセッサに使用することができず、このＭ行Ｎ列の演算ユニットを有する動的再構成可能プロセッサ向けに新たにプログラムを開発しなければならない。
【００２０】
本発明は上記事情に鑑みてなされたものであり、本発明の目的は動的再構成可能プロセッサ向けプログラムの開発にかかる負担を軽減することにある。
【課題を解決するための手段】
【００２１】
上記課題を解決するために、本発明の動的再構成可能プロセッサでは、１列目の各演算ユニットへのデータ入力を遅延する遅延ユニットを設けている。
【００２２】
例えば、本発明の動的再構成可能プロセッサは、設定された構成情報に従って入出力先および演算内容を切り替え可能な演算ユニットが、Ｍ行Ｎ列（但し、Ｍ≧２、Ｎ≧１）に配列されて構成された、動的再構成可能プロセッサであって、
設定された構成情報に従って１列目の演算ユニットへのデータ入力を遅延させる遅延ユニットを有する。
【００２３】

また、本発明のコンパイラ装置では、本発明の動的再構成可能プロセッサよりも演算ユニットの列数が多い他の動的再構成可能プロセッサ向けに開発された入力プログラムを、複数回に分割して本発明の動的再構成可能プロセッサに実行させる出力プログラムに変換する。ここで、出力プログラムは、ある回の実行により、本発明の動的再構成可能プロセッサを構成するＭ行Ｎ列の演算ユニットのＮ列目から出力された各データが、次の回の実行のときに、当該各データの出力サイクルのサイクル差で前記Ｍ行Ｎ列の演算ユニットの１列目に入力されるように、遅延ユニットを制御するための構成情報を含む。

【００２４】

例えば、本発明のコンパイラ装置は、動的再構成可能プロセッサを構成するＭ行Ｌ列の演算ユニットに設定する構成情報を含む入力プログラムを、本発明の動的再構成可能プロセッサを構成する遅延ユニットおよびＭ行Ｎ（但しＮ＜Ｌ）列の演算ユニットに設定する構成情報を含む出力プログラムに変換するコンパイラ装置であって、

前記入力プログラムを記憶する入力プログラム記憶手段と、

前記入力プログラムに含まれる前記Ｍ行Ｌ列の演算ユニットの構成情報をＮ列毎に分割して、ｎｕｍ（但し、ｎｕｍはＭ/Ｎより大きい最小の整数）回分の前記Ｍ行Ｎ列の演算ユニットの構成情報（分割構成情報と呼ぶ）を生成する構成情報分割手段と、

ｕ回目の分割構成情報を前記Ｍ行Ｎ列の演算ユニットに設定した場合にＮ列目の演算ユニットから出力される各データの出力サイクルのサイクル差を算出する処理を、前記ｕが１からｎｕｍ-１となるまで繰り返す遅延解析手段と、

ｔ回目の分割構成情報が設定された前記Ｍ行Ｎ列の演算ユニットのＮ列目から出力される各データが、前記遅延解析手段により算出された当該各データの出力サイクルのサイクル差で前記Ｍ行Ｎ列の演算ユニットの１列目に入力されるように、前記遅延ユニットを制御するための構成情報を、Ｋ+１回目の分割構成情報に含める処理を、ｔが１からｎｕｍ-１となるまで繰り返して、ｎｕｍ回分の分割構成情報を含む出力プログラムを生成するプログラム出力手段と、を有する。

【発明の効果】

【００２５】

本発明の動的再構成可能プロセッサによれば、遅延ユニットにより、Ｎ列目（最終列目）の演算ユニットから出力された各データを、１列目の対応する（同行の）演算ユニットに、当該各データのＮ列目の演算ユニットからの出力サイクル差を与えて入力することができる。したがって、本発明の動的再構成可能プロセッサよりも演算ユニットの列数が多い他の動的再構成可能プロセッサ向けに開発された入力プログラムを、複数回に分割して実行できる。

【００２６】

また、本発明のコンパイラ装置によれば、本発明の動的再構成可能プロセッサよりも演算ユニットの列数が多い他の動的再構成可能プロセッサ向けに開発された入力プログラムを、本発明の動的再構成可能プロセッサ向けの出力プログラムに変換することができる。

【００２７】

このように、本発明によれば、動的再構成可能プロセッサ向けプログラムの開発にかかる負担を軽減することができる。

【発明を実施するための最良の形態】

【００２８】

以下、本発明の実施の形態を説明する。

【００２９】

<<第１実施形態>>

先ず、本発明の第１実施形態として動的再構成可能プロセッサを説明する。

【００３０】

図１は、本発明の第１実施形態が適用された動的再構成可能プロセッサを有する情報処理装置の概略図である。

【００３１】

図示するように、本実施形態の情報処理装置は、ホストプロセッサ（ＣＰＵ）１０と、メモリ（ＭＥＭ）２０と、局所メモリ（ＲＡＭ）４０と、動的再構成可能プロセッサ（ＤＲＰ）９０と、を有する。ホストプロセッサ１０、メモリ２０および局所メモリ４０は、内部バス３０で互いに接続されている。また、局所メモリ４０および動的再構成可能プロセッサ９０は、局所メモリ４０から動的再構成可能プロセッサ９０へデータおよびプログラムを転送するための入力配線７０と、動的再構成可能プロセッサ９０から局所メモリ４０へデータを転送するための出力配線８０とを介して互いに接続されている。

【００３２】

メモリ２０には、ホストプロセッサ１０および動的再構成可能プロセッサ９０が使用するデータ、プログラムが格納される。ホストプロセッサ１０は、メモリ２０に記憶されたホストプロセッサ１０用のプログラムに従い、動的再構成可能プロセッサ９０の起動、終結、および、動的再構成可能プロセッサ９０へのデータ、プログラムの転送を制御する。局所メモリ４０は、動的再構成可能プロセッサ９０が使用するデータ、プログラムが格納される。動的再構成可能プロセッサ９０は、構成情報を格納する構成情報格納部５０と、演算部６０と、を有する。

【００３３】

演算部６０は、Ｍ行Ｎ列（図１では２行２列）に配列された演算ユニット（ＥＸ）６３と、１列目の演算ユニット６３毎に設けられたＭ個（Ｍ行１列）のロードユニット（ＬＤ）６１と、行毎にロードユニット６１と１列目の演算ユニット６３との間に設けられたＭ個（Ｍ行１列）の遅延ユニット（ＤＹ）６２と、Ｎ列目の演算ユニット６３毎に設けられたＭ個（Ｍ行１列）のストアユニット（ＳＴ）６４と、を有する。

【００３４】

ロードユニット６１は、設定されたロードユニット用構成情報に従い、１列目の対応する（同行の）演算ユニット６３へ入力するデータを局所メモリ４０からロードする。

【００３５】

遅延ユニット６２は、対応する（同行）ロードユニット６１から出力されたデータを、設定された遅延ユニット用構成情報が示す遅延サイクル数分遅延させ、１列目の対応する（同行の）演算ユニット６３へ出力する。

【００３６】

演算ユニット６３は、図示していないが、複数の入出力ポートと、四則演算等の各種演算を行う複数の演算回路と、入出力ポートおよび演算回路の接続を切替えるスイッチと、を有する。そして、演算ユニット６３は、設定された演算ユニット用構成情報に従い、スイッチを制御して、データの入力元、演算内容および演算結果の出力先を切り替える。なお、演算ユニット６３同士の配線は上下左右の４方向であり、どの方向にも双方向にデータ（例えば８ビットデータ）を転送可能である。

【００３７】

図２は、演算ユニット用構成情報メモリ５３に格納される演算ユニット用構成情報の一例を示す図である。図示するように、演算ユニット用構成情報は、演算ユニット６３の列番号（Ｘ座標）を登録するフィールド５３１と、演算ユニット６３の行番号（Ｙ座標）を登録するフィールド５３２と、演算ユニット６３に行わせる演算種別を登録するフィールド５３３と、第１オペランドがどの方向の演算ユニット６３または遅延ユニット６２から来るかを示す情報を登録するフィールド５３４と、第２オペランドがどの方向の演算ユニット６３または遅延ユニット６２から来るかを示す情報を登録するフィールド５３５と、第３オペランドがどの方向の演算ユニット６３または遅延ユニット６２から来るかを示す情報を登録するフィールド５３６と、を有する。ここでは、フィールド５３４～５３６に方向を示す情報として、自演算ユニットのアドレス（列番号、行番号）を（Ｘ，Ｙ）とした場合に、上（Ｘ-１，Ｙ）を「００」、右（Ｘ，Ｙ+１）を「０１」、下（Ｘ+１，Ｙ）を「１０」、そして、左（Ｘ，Ｙ-１）を「１１」としている。なお、フィールド５３４～５３６のいずれにも登録されていない方向が、演算結果の出力先となる。また、図２に示す例では、演算ユニット用構成情報が、演算ユニット６３のアドレス（１列目、２行目

）、演算種別：加算（ａｄｄ）、第１オペランド入手先：左、第２オペランド入手先：上、第３オペランド入手先：右であることを表している。この演算ユニット用構成情報は、演算部６０に含まれている演算ユニット６３の数分存在する。

【００３８】

ストアユニット６４は、設定されたストアユニット用構成情報に従い、Ｎ列目の対応する（同行の）演算ユニット６３から出力されるデータを局所メモリ４０へストアする。

【００３９】

構成情報格納部５０は、ロードユニット用構成情報メモリ（ＭＬ）５１と、遅延ユニット用構成情報メモリ（ＭＤ）５２と、Ｍ行Ｎ列の演算ユニットの列毎に設けられたＮ個の演算ユニット用構成情報メモリ（ＭＥ）５３と、ストアユニット用構成情報メモリ（ＭＳ）５４と、を有する。

【００４０】

ロードユニット用構成情報メモリ５１は、Ｍ個のロードユニット６１各々に設定するロードユニット用構成情報（データをロードするアドレス等のパラメータ情報）を格納する。遅延ユニット用構成情報メモリ５２は、Ｍ個の遅延ニット６２各々に設定する遅延ユニット用構成情報（遅延サイクル数）を格納する。演算ユニット用構成情報メモリ５３は、対応する列のＭ個の演算ユニット６３各々に設定する演算ユニット用構成情報（データ入出力先および演算内容を特定する情報）を格納する。そして、ストアユニット用構成情報メモリ５４は、Ｍ個のストアユニット６４各々に設定するストアユニット用構成情報（データをストアするアドレス等のパラメータ情報）を格納する。

【００４１】

以上のような構成において、ホストプロセッサ１０は、動的再構成可能プロセッサ９０が使用するデータ、プログラムを、メモリ２０から局所メモリ４０へ転送する。ここで、プログラムは、上述の、ロードユニット用構成情報、遅延ユニット用構成情報、演算ユニット用構成情報、および、ストアユニット用構成情報を含む。

【００４２】

次に、ロードユニット用構成情報メモリ５１は、局所メモリ４０からロードユニット用構成情報を読み込み、各ロードユニット６１に構成情報を設定する。また、遅延ユニット用構成情報メモリ５２は、局所メモリ４０から遅延ユニット用構成情報を読み込み、各遅延ユニット６２に構成情報を設定する。また、各演算ユニット用構成情報メモリ５３は、対応する列の演算ユニット用構成情報を読込み、当該列の各演算ユニット６３に構成情報を設定する。また、ストアユニット用構成情報メモリ５４は、局所メモリ４０からストアユニット用構成情報を読み込み、各ストアユニット６４に構成情報を設定する。

【００４３】

さて、以上のようにして演算部６０の各ユニットに構成情報が設定されると、先ず、ロードユニット６１が、設定された構成情報に従い、局所メモリ４０からデータをロードし、これを対応する（同行の）遅延ユニット６２に転送する。次に、遅延ユニット６２が、ロードユニット６１から送られてきたデータを、設定された構成情報が示す遅延サイクル分だけ遅延させ、それから１列目の対応する（同行の）演算ユニット６３へ転送する。次に、演算ユニット６３が、設定された構成情報に従い、データの入力元、演算内容および演算結果の出力先を切り替えて演算処理を行う。そして、ストアユニット６４が、設定された構成情報に従い、Ｎ列目の対応する（同行）演算ユニット６３から出力されたデータを局所メモリ４０にストアする。

【００４４】

このように、本実施形態の動的再構成可能プロセッサ９０では、プログラムによって複数の演算ユニット６３各々の演算内容および演算順番を変更することができ、従って動的に機能を変更することができる。また、遅延ユニット６２により、Ｎ列目の演算ユニット６３から出力される各データを、１列目の対応する（同行の）演算ユニット６３に、当該各データのＮ列目の演算ユニット６３からの出力サイクル差を与えて入力できる。つまり、Ｍ行Ｎ列の演算ユニット６３から出力される複数のデータを、局所メモリ４０を介して

、Ｍ行Ｎ列の演算ユニット６３に再投入する場合に、局所メモリ４０によって吸収されてしまう出力サイクルのサイクル差を再現して、当該複数のデータをＭ行Ｎ列の演算ユニット６３に再投入できる。したがって、本実施形態の動的再構成可能プロセッサ９０よりも演算ユニット６３の列数が多い他の動的再構成可能プロセッサ（遅延ユニット６２を有さない既存の動的再構成可能プロセッサ）向けに開発された入力プログラムを、複数回に分割して実行できる。この場合、１回目の実行で使用する遅延ユニット用構成情報は、各遅延ユニット６２での遅延サイクル数を「０」にするものとする。また、２回目以降の実行で使用する遅延ユニット用構成情報は、１つ前の実行でＮ列目の演算ユニット６３から出力される各データのサイクル差を再現するために必要な、各遅延ユニット６２での遅延サイクル数とする。

【００４５】

なお、動的再構成可能プロセッサ９０は、遅延ユニット６２および遅延ユニット用構成情報メモリ５２が追加されている点を除き、図２８に示す動的再構成可能プロセッサ９００と同様である。そこで、以下では、遅延ユニット６２および遅延ユニット用構成情報メモリ５２について説明し、その他の構成の説明は省略する。

【００４６】

図３は、図１に示す遅延ユニット６２の回路構成例を示す図である。この例では最大で３サイクルまで遅延することができる回路を示している。ここで、符号６２０１は、対応する（同列の）のロードユニット６１から出力されたデータ（例えば８ビットデータ）を取り込むためのデータ入力線（ＩＮ）であり、符号６２０２は、データを１列目の対応する（同列の）演算ユニット６３へ転送するためのデータ出力線（ＯＵＴ）であり、そして、符号６２１５は、クロック信号線（ＣＬＫ）である。また、符号６２０３～６２０５は、遅延ユニット用構成情報メモリ５２から遅延ユニット用構成情報を取り込むための構成情報入力線である。構成情報入力線６２０３はデータ遅延を行うか否かを決定するためのイネーブル信号線（Ｅ）であり、そして、構成情報入力線６２０４、６２０５は、遅延サイクル数（１サイクル～３サイクル）を決定するためのビット線（Ｄ１、Ｄ２）である。

【００４７】

図示するように、遅延ユニット６２は、複数のアンド回路６２０６～６２０９と、複数のフリップフロップ回路６２１０～６２１４と、を有する。

【００４８】

フリップフロップ回路（ＦＦＢ１）６２１０には、イネーブル信号線６２０３およびビット線６２０４が入力されている。そして、イネーブル信号線６２０３がＯＮの場合、ビット線６２０４がＯＮならば、出力結果（Ｑ）をＯＮ、出力結果の否定値（ＮＯＴ Ｑ）をＯＦＦとし、一方、ビット線６２０４がＯＦＦならば、出力結果をＯＦＦ、出力結果の否定値をＯＮとする。

【００４９】

アンド回路（Ａ１）６２０６には、データ入力線６２０１およびフリップフロップ回路６２１０の出力結果が入力されている。フリップフロップ回路６２１０の出力結果がＯＮのときにデータ入力線６２０１のデータを出力し（ＯＮ）、ＯＦＦのときに出力しない（ＯＦＦ）。

【００５０】

アンド回路（Ａ２）６２０７には、データ入力線６２０１およびフリップフロップ回路６２１０の出力結果の否定値が入力されている。フリップフロップ回路６２１０の出力結果の否定値がＯＮのときにデータ入力線６２０１のデータを出力し（ＯＮ）、ＯＦＦのときに出力しない（ＯＦＦ）。

【００５１】

フリップフロップ回路（ＦＦＢ２）６２１１には、イネーブル信号線６２０３およびビット線６２０５が入力されている。そして、イネーブル信号線６２０３がＯＮの場合、ビット線６２０５がＯＮならば、出力結果（Ｑ）をＯＮ、出力結果の否定値（ＮＯＴ Ｑ）をＯＦＦとし、一方、ビット線６２０５がＯＦＦならば、出力結果をＯＦＦ、出力結果の

否定値をＯＮとする。
【００５２】
アンド回路（Ａ３）６２０８には、アンド回路６２０８の出力信号線およびフリップフロップ回路６２１１の出力結果が入力されている。フリップフロップ回路６２１１の出力結果がＯＮのときにアンド回路６２０７の出力信号線のデータを出力し（ＯＮ）、ＯＦＦのときに出力しない。
【００５３】
アンド回路（Ａ４）６２０９には、アンド回路６２０７の出力信号線およびフリップフロップ回路６２１１の出力結果の否定値が入力されている。フリップフロップ回路６２１１の出力結果の否定値がＯＮのときにアンド回路６２０７の出力信号線のデータを出力し（ＯＮ）、ＯＦＦのときに出力しない。
【００５４】
フリップフロップ回路（ＦＦＡ１）６２１２には、アンド回路６２０６の出力信号線が入力されており、クロック信号線６２１５のクロック信号に従い、入力されたデータを１サイクル遅延して出力する。
【００５５】
フリップフロップ回路（ＦＦＡ２）６２１３には、フリップフロップ回路６２１２の出力信号線が入力されており、クロック信号線６２１５のクロック信号に従い、入力されたデータを１サイクル遅延して出力する。ここで、フリップフロップ回路６２１４の出力信号線は、アンド回路６２０７の出力信号線に接続されている。
【００５６】
フリップフロップ回路（ＦＦＡ３）６２１４には、アンド回路６２０８の出力信号線が入力されており、クロック信号線６２１５のクロック信号に従い、入力されたデータを１サイクル遅延して出力する。ここで、フリップフロップ回路６２１４の出力信号線は、アンド回路６２０９の出力信号線に接続されている。
【００５７】
さて、以上のような構成を有する遅延ユニット６２の動作は、イネーブル信号線６２０３がＯＮの場合、次のようになる。
【００５８】
（１-１）ビット信号線６２０４：ＯＦＦ　ビット信号線６２０５：ＯＦＦ
遅延サイクル数＝０
アンド回路６２０６、６２０８がＯＦＦとなり、アンド回路６２０７、６２０９がＯＮＮとなる。したがって、データ入力線６２０１のデータは、フリップフロップ回路６２１２～６２１４を経由することなく、データ出力線６２０２から出力されるため、遅延サイクル数は０となる。
【００５９】
（１-２）ビット信号線６２０４：ＯＦＦ　ビット信号線６２０５：ＯＮ
遅延サイクル数＝１
アンド回路６２０６、６２０９がＯＦＦとなり、アンド回路６２０７、６２０８がＯＮＮとなる。したがって、データ入力線６２０１のデータは、フリップフロップ回路６２１４を経由して、データ出力線６２０２から出力されるため、遅延サイクル数は１となる。
【００６０】
（１-３）ビット信号線６２０４：ＯＮ　ビット信号線６２０５：ＯＦＦ
遅延サイクル数＝２
アンド回路６２０６、６２０９がＯＮとなり、アンド回路６２０７、６２０８がＯＦＦとなる。したがって、データ入力線６２０１のデータは、フリップフロップ回路６２１２、６２１３を経由して、データ出力線６２０２から出力されるため、遅延サイクル数は２となる。
【００６１】
（１-４）ビット信号線６２０４：ＯＮ　ビット信号線６２０５：ＯＮ

遅延サイクル数＝３

アンド回路６２０６、６２０８がＯＮとなり、アンド回路６２０７、６２０９がＯＦＦとなる。したがって、データ入力線６２０１のデータは、フリップフロップ回路６２１２、６２１３、６２１４を経由して、データ出力線６２０２から出力されるため、遅延サイクル数は３となる。

【００６２】

図４は、図１に示す遅延ユニット用構成情報メモリ５２の回路構成例を示す図である。この例では４個（４行１列）の図２に示す遅延ユニット６２に遅延ユニット用構成情報を設定することができる回路を示している。ここで、符号５２０１は、ホストプロセッサ１０から出力されるフリップフロップのセット信号を取り込むためのセット信号線（ＳＥＴ）であり、符号５２０２は、ホストプロセッサ１０から出力されるフリップフロップのリセット信号を取り込むためのリセット信号線（ＲＥＳＥＴ）であり、符号５２０３は、クロック信号線（ＣＬＫ）である。また、符号５２０４は、局所メモリ４０から出力された遅延ユニット用構成情報（１つの遅延ユニット６２につき２ビット）を取り込むための構成情報入力線である。また、符号５２１４、５２１５は、４個の遅延ユニット６２へのイネーブル信号線（Ｅ）６２０３_{1}～６２０３_{4}を順番にＯＮにするための駆動信号線である。そして、符号５２１２は、４個の遅延ユニット６２へのビット線（Ｄ１）６２０４_{1}～６２０４_{4}上に順番にビットデータを出力するためのデータ信号線であり、符号５２１３は、４個の遅延ユニット６２へのビット線（Ｄ２）６２０５_{1}～６２０５_{4}上に順番にビットデータを出力するためのデータ信号線である。

【００６３】

図示するように、遅延ユニット用構成情報メモリ５２は、送りレジスタを構成する複数のフロップフロップ回路（ＦＦ）５２０５_{1}～５２０５_{4}と、遅延ユニット６３のアドレスを記憶するフリップフロップ回路（ＹＣ）５２０６_{1}～５２０６_{4}と、フリップフロップ回路５２０６_{1}～５２０６_{4}毎に設けられた２つのアンド回路５２０７_{1}～５２０７_{4}、５２０６８～５２０８_{4}と、遅延ユニット６３に設定する構成情報（２ビット）を記憶するフリップフロップ回路（ＤＣ）５２０９_{1}～５２０９_{4}と、フリップフロップ回路５２０９_{1}～５２０９_{4}毎に設けられた２つのアンド回路５２１０_{1}～５２１０_{4}、５２１１_{1}～５２１１_{4}と、遅延ユニット６２のイネーブル信号線６２０３_{1}～６２０３_{4}毎に設けられたアンド回路５２１７_{1}～５２１７_{4}と、アンド回路５２１７_{1}の２つの入力の両方に設けられたノット回路５２１８_{1}、５２１６_{1}と、アンド回路５２１７_{2}の２つの入力の一方に設けられたノット回路５２１８_{2}と、アンド回路５２１７_{3}の２つの入力の一方に設けられたノット回路５２１６_{3}と、を有する。

【００６４】

ここで、アンド回路５２０７_{1}、５２０８_{1}各々の２つの入力の一方は、フリップフロップ回路５２０５_{1}の出力に接続され、他方は、フリップフロップ回路５２０６_{1}の出力に接続されている。なお、フリップフロップ回路５２０６_{1}の２つの出力は、ＯＦＦ（ビット値＝０）、ＯＦＦである。このため、フリップフロップ回路５２０６_{1}からアンド回路５２０７_{1}への出力はＯＦＦであり、また、フリップフロップ回路５２０６_{1}からアンド回路５２０８_{1}への出力はＯＦＦである。

【００６５】

また、アンド回路５２０７_{2}、５２０８_{2}各々の２つの入力の一方は、フリップフロップ回路５２０５_{2}の出力に接続され、他方は、フリップフロップ回路５２０６_{2}の出力に接続されている。なお、フリップフロップ回路５２０６_{2}の２つの出力は、ＯＦＦ、ＯＮ（ビット値＝１）である。このため、フリップフロップ回路５２０６_{2}からアンド回路５２０７_{2}への出力はＯＦＦであり、また、フリップフロップ回路５２０６_{2}からアンド回路５２０８_{2}への出力はＯＮである。

【００６６】

また、アンド回路５２０７_{3}、５２０８_{3}各々の２つの入力の一方は、フリップフロップ回路５２０５_{3}の出力に接続され、他方は、フリップフロップ回路５２０６_{3}の出力に

接続されている。なお、フリップフロップ回路５２０６$_3$の２つの出力は、ＯＮ、ＯＦＦである。このため、フリップフロップ回路５２０６$_3$からアンド回路５２０７$_3$への出力はＯＮであり、また、フリップフロップ回路５２０６$_3$からアンド回路５２０８$_3$への出力はＯＦＦである。

【００６７】

また、アンド回路５２０７$_4$、５２０８$_4$各々の２つの入力の一方は、フリップフロップ回路５２０５$_4$の出力に接続され、他方は、フリップフロップ回路５２０６$_4$の出力に接続されている。なお、フリップフロップ回路５２０６$_4$の２つの出力は、ＯＮ、ＯＮである。このため、フリップフロップ回路５２０６$_4$からアンド回路５２０７$_4$への出力はＯＮであり、また、フリップフロップ回路５２０６$_4$からアンド回路５２０８$_4$への出力はＯＮである。

【００６８】

アンド回路５２０７$_1$～５２０７$_4$の出力は駆動信号線５２１５に接続されており、また、アンド回路５２０８$_1$～５２０８$_4$の出力は駆動信号線５２１４に接続されている。駆動信号線５２１４は、ノット回路５２１８$_1$の入力、ノット回路５２１８$_2$の入力、アンド回路５２１７$_3$の入力、そして、アンド回路５２１７$_4$の入力に接続されており、駆動信号線５２１５は、ノット回路５２１６$_1$の入力、アンド回路５２１７$_2$の入力、ノット回路５２１６$_3$の入力、そして、アンド回路５２１７$_4$の入力に接続されている。

【００６９】

アンド回路５２１０$_1$、５２１１$_1$各々の２つの入力の一方は、フリップフロップ回路５２０５$_1$の出力に接続され、他方は、フリップフロップ回路５２０９$_1$の出力に接続されている。また、アンド回路５２１０$_2$、５２１１$_2$各々の２つの入力の一方は、フリップフロップ回路５２０５$_2$の出力に接続され、他方は、フリップフロップ回路５２０９$_2$の出力に接続されている。また、アンド回路５２１０$_3$、５２１１$_3$各々の２つの入力の一方は、フリップフロップ回路５２０５$_3$の出力に接続され、他方は、フリップフロップ回路５２０６$_3$の出力に接続されている。また、アンド回路５２１０$_4$、５２１１$_4$各々の２つの入力の一方は、フリップフロップ回路５２０５$_4$の出力に接続され、他方は、フリップフロップ回路５２０６$_4$の出力に接続されている。

【００７０】

なお、アンド回路５２１０$_1$～５２１０$_4$の出力はデータ信号線５２１３に接続されており、アンド回路５２１１$_1$～５２１１$_4$の出力はデータ信号線５２１２に接続されている。また、フリップフロップ回路５２０９$_1$には、構成情報入力線５２０４を介して１行目の遅延ユニット６２に設定する遅延ユニット用構成情報が格納され、フリップフロップ回路５２０９$_2$には、構成情報入力線５２０４を介して２行目の遅延ユニット６２に設定する遅延ユニット用構成情報が格納され、フリップフロップ回路５２０９$_3$には、構成情報入力線５２０４を介して３行目の遅延ユニット６２に設定する遅延ユニット用構成情報が格納され、そして、フリップフロップ回路５２０９$_4$には、構成情報入力線５２０４を介して４行目の遅延ユニット６２に設定する遅延ユニット用構成情報が格納される。

【００７１】

さて、以上のような構成を有する遅延ユニット用構成情報メモリ５２は、セット信号５２０１にパルス信号が入力され、これにより、フロップフロップ回路５２０５$_1$～５２０５$_4$が、１サイクルずつずれて順番にパルス信号を出力することにより、次のように動作する。

【００７２】

（２-１）フロップフロップ回路５２０５$_1$がパルス出力

１行目の遅延ユニット６２に遅延ユニット用構成情報が設定される。アンド回路５２０７$_1$、５２０８$_1$の出力が駆動信号線５２１５、５２１４上に出力され、これにより、駆動信号線５２１４、５２１５が共にＯＦＦとなる。したがって、アンド回路５２１７$_1$の出力つまり１行目の遅延ユニット６２へのイネーブル信号線６２０３$_1$のみがＯＮとなる。また、アンド回路５２１０$_1$、５２１１$_1$の出力がデータ信号線５２１３、５２１２上

に出力され、これにより、フリップフロップ回路５２０９$_1$に記憶されている１行目の遅延ユニット６２用の遅延ユニット用構成情報が、データ信号線５２１２、５２１３を介して、１行目の遅延ユニット６２へのビット線６２０４$_1$、６２０５$_1$に出力される。

【００７３】

（２-２）フロップフロップ回路５２０５$_2$がパルス出力

２行目の遅延ユニット６２に遅延ユニット用構成情報が設定される。アンド回路５２０７$_2$、５２０８$_2$の出力が駆動信号線５２１５、５２１４上に出力され、これにより、駆動信号線５２１４、５２１５がＯＦＦ、ＯＮとなる。したがって、アンド回路５２１７$_2$の出力つまり２行目の遅延ユニット６２へのイネーブル信号線６２０３$_2$のみがＯＮとなる。また、アンド回路５２１０$_2$、５２１１$_2$の出力がデータ信号線５２１３、５２１２上に出力され、これにより、フリップフロップ回路５２０９$_2$に記憶されている２行目の遅延ユニット６２用の遅延ユニット用構成情報が、データ信号線５２１２、５２１３を介して、２行目の遅延ユニット６２へのビット線６２０４$_2$、６２０５$_2$に出力される。

【００７４】

（２-３）フロップフロップ回路５２０５$_3$がパルス出力

３行目の遅延ユニット６２に遅延ユニット用構成情報が設定される。アンド回路５２０７$_3$、５２０８$_3$の出力が駆動信号線５２１５、５２１４上に出力され、これにより、駆動信号線５２１４、５２１５がＯＮ、ＯＦＦとなる。したがって、アンド回路５２１７$_3$の出力つまり３行目の遅延ユニット６２へのイネーブル信号線６２０３$_3$のみがＯＮとなる。また、アンド回路５２１０$_3$、５２１１$_3$の出力がデータ信号線５２１３、５２１２上に出力され、これにより、フリップフロップ回路５２０９$_3$に記憶されている３行目の遅延ユニット６２用の遅延ユニット用構成情報が、データ信号線５２１２、５２１３を介して、３行目の遅延ユニット６２へのビット線６２０４$_3$、６２０５$_3$に出力される。

【００７５】

（２-４）フロップフロップ回路５２０５$_4$がパルス出力

４行目の遅延ユニット６２に遅延ユニット用構成情報が設定される。アンド回路５２０７$_4$、５２０８$_4$の出力が駆動信号線５２１５、５２１４上に出力され、これにより、駆動信号線５２１４、５２１５が共にＯＮとなる。したがって、アンド回路５２１７$_4$の出力つまり４行目の遅延ユニット６２へのイネーブル信号線６２０３$_4$のみがＯＮとなる。また、アンド回路５２１０$_4$、５２１１$_4$の出力がデータ信号線５２１３、５２１２上に出力され、これにより、フリップフロップ回路５２０９$_4$に記憶されている４行目の遅延ユニット６２用の遅延ユニット用構成情報が、データ信号線５２１２、５２１３を介して、４行目の遅延ユニット６２へのビット線６２０４$_4$、６２０５$_4$に出力される。

【００７６】

<<第２実施形態>>

次に、本発明の第２実施形態として、上記の第１実施形態で説明した動的再構成可能プロセッサ向けプログラムのコンパイラ装置を説明する。本実施形態のコンパイラ装置は、第１実施形態で説明した動的再構成可能プロセッサ９０よりも演算ユニットの列数が多い既存の動的再構成可能プロセッサ（図２８に示すような、遅延ユニット６３および遅延ユニット用構成情報メモリ５２を有さない動的再構成可能プロセッサ９００）向けに開発されたプログラムを、第１実施形態で説明した動的再構成可能プロセッサ９０向けのプログラムに変換する装置である。

【００７７】

図５は、本発明の第２実施形態が適用されたコンパイラ装置の概略構成図である。図示するように、コンパイラ装置１０００は、演算部２０００と、記憶部３０００と、プログラムの入出力を行う入出力部４０００と、を有する。

【００７８】

記憶部３０００は、コンパイル対象のプログラムである入力プログラムを記憶する入力プログラム記憶部３１００と、構成情報のコード生成に利用する辞書を記憶する辞書記憶部３２００と、演算部６０の各ユニット６１～６４に設定する構成情報を管理するための

配置テーブルを記憶する配置テーブル記憶部３３００と、入力プログラムのコンパイル結果である出力プログラムを記憶する出力プログラム記憶部３４００と、を有する。

【００７９】

図６は、辞書記憶部３２００に記憶される辞書の一例を示している。辞書は入力プログラム中に記述されている変数（配列）毎に生成される。図示するように、別の変数に対する辞書へのポインタを格納するフィールド１５０１と、変数名を表す文字列へのポインタを格納するフィールド１５０２と、変数の型を格納するフィールド１５０３と、配列の大きさがコンパイル時に固定であるか否かを表す情報を格納するフィールド１５０４と、配列添字の下限値を格納するフィールド１５０５と、配列添字の上限値を格納するフィールド１５０６、配列が初期設定される場合の初期値へのポインタを格納するフィールド１５０７と、を有する。ここで、フィールド１５０４に格納されている情報が"fixed"の場合、配列の大きさは固定である。一方、該情報が"dynamic"の場合、配列の大きさはプログラム実行時に決まる。また、フィールド１５０７にポインタが格納されている場合、当該ポインタで示される位置には、配列に設定される初期値が格納される。

【００８０】

なお、図６に示す例では１次元配列に対する辞書を示しているが、一般の次元の場合も同様である。

【００８１】

図７は、配置テーブル記憶部３３００に記憶される配置テーブルの一例を示している。図示するように、配置テーブルは動的再構成可能プロセッサを構成するユニット毎に設けられたエントリ１６４０を有する。エントリ１６４０には、対応するユニットの設定情報が格納される。図７において、左端のエントリ１６４０の列は入力列１６１０であり、列１６１０の各行のエントリ１６４０が各行のロードユニットに対応する。右端のエントリ１６４０の列は出力列１６３０であり、列１６３０の各行のエントリ１６４０が各行のストアユニットに対応する。そして、入力列１６１０および出力列１６３０で挟まれたエントリ１６４０の複数の列が配置列１６２０であり、各行各列のエントリ１６４０が各行各列の演算ユニットに相当する。

【００８２】

図８は、図７に示す配置テーブルのエントリ１６４０に格納される設定情報を説明するための図である。図示するように、設定情報は、演算種別を登録するフィールド１６４１と、図７において上側に位置するエントリ１６４０に対するオペランド入出力情報（入力オペランド"in"および出力オペランド"out"のいずれか）を登録するフィールド１６４２と、右側に位置するエントリ１６４０に対するオペランド入出力情報を登録するフィールド１６４３と、下側に位置するエントリ１６４０に対するオペランド入出力情報を登録するフィールド１６４４と、左側に位置するエントリ１６４０に対するオペランド入出力情報を登録するフィールド１６４５と、上側に位置するエントリ１６４０に対するオペランド指定情報（指定"connect"および未指定"open"のいずれか）を登録するフィールド１６４８と、右側に位置するエントリ１６４０に対するオペランド指定情報を登録するフィールド１６４９と、下側に位置するエントリ１６４０に対するオペランド指定情報を登録するフィールド１６５０と、左側に位置するエントリ１６４０に対するオペランド指定情報登録するフィールド１６５１と、対応するユニットでの処理の実行タイミング（サイクル数）を登録するフィールド１６４６と、対応するユニットがロードユニットあるいは出力ユニットである場合に、ロードあるいはストアすべきデータの辞書へのポインタを格納するフィールド１６４７と、を有する。

【００８３】

なお、動的再構成可能プロセッサに処理を続けて複数回実行させる場合、処理回数と同じ数の配置テーブルが配置テーブル記憶部３３００に記憶される。このため、配置テーブル記憶部３３００には、配置テーブル毎に、配置テーブル同士を接続するための配置テーブルリストも記憶される。図９は、配置テーブル記憶部３３００に記憶される配置テーブルリストの一例を示している。図示するように、配置テーブルリストは、次の配置テーブ

ルへのポインタを格納するフィールド１７１０と、配置テーブルへのポインタを格納するフィールド１７２０と、配置テーブルに利用する辞書へのポインタを格納するフィールド１７３０と、を有する。

【００８４】

図５に戻って説明を続ける。演算部２０００は、構文解析部２１００と、構成情報分割部２２００と、遅延情報解析部２３００と、コード生成部２４００と、を有する。

【００８５】

構文解析部２１００は、入力プログラムの構文解析を行って辞書および配置テーブルを生成する。構文情報分割部２２００は、入力プログラムに含まれている構成情報を当該プログラムが対象とする既存の動的再構成可能プロセッサよりも演算ユニットの列数が少ない第１実施形態の動的再構成可能プロセッサ向けの構成情報に分割し、その結果を辞書および配置テーブルに反映させる。遅延解析部２３００は、分割された構成情報を第１実施形態の動的再構成可能プロセッサに設定した場合における当該動的再構成可能プロセッサから出力される各データの出力サイクルを解析し、出力サイクルのずれ（遅延）を配置テーブルに反映する。そして、コード生成部２４００は、辞書および配置テーブルを用いて、第１実施形態の動的再構成可能プロセッサ向けの出力プログラムを生成する。ここで、出力プログラムは、第１実施形態の動的再構成可能プロセッサ９０に処理を複数回実行させるプログラムである。

【００８６】

なお、図５に示すコンパイラ装置１０００は、例えば図１０に示すような、ホストプロセッサ１１１と、メモリ１１２と、ＨＤＤ等の外部記憶装置１１３と、キーボード、マウス等の入力装置１１４と、ＣＲＴ、ＬＣＤ等の表示装置１１５と、ＮＩＣ等の通信装置１１６と、これらを接続するバス１１８と、を有する一般的なコンピュータシステムにおいて、ホストプロセッサ１１１がメモリ１１２上にロードしたプログラム（コンパイラプログラム）を実行することにより実現できる。この場合、入出力部４０００には入力装置１１４、表示装置１１５および/または通信装置１１６が、そして、記憶部３０００にはメモリ１１２および/または外部記憶装置１１３が用いられる。

【００８７】

以下に、構文解析部２１００、構文情報分割部２２００、遅延解析部２３００、および、コード生成部２４００の動作について、図１１に示す入力プログラムが入力プログラム記憶部３１００に記憶されている場合を例にとり説明する。なお、図１１に示す入力プログラムは、Ｃ言語で記述されている。

【００８８】

図１１に示す入力プログラムにおいて、１行目の記述２００１は、関数名および引数の宣言である。ここで、引数ならびに３つの配列"ａ", "ｂ", "ｃ"は、宣言された関数への入力であり、配列"ｘ"は該関数からの出力である。２行目の記述２００２は、８行目の記述２００８および１１行目の記述２０１１で記述されている関数に対して不定個数の引数を指定するための引数配列の宣言である。３行目の記述２００３は、図２８に示すような既存の動的再構成可能プロセッサ向けの構成情報を配列"conf1"へ初期値設定するための記述である。

【００８９】

また、４行目の記述２００４は、引数配列"args"を初期化するための記述である。５行目の記述２００５は、引数配列"args"に、配列"a"の先頭アドレスをaとし、要素数を１００とし、そして、入力座標０から配列"a"の値を入力することを設定するための記述である。ここで、入力座標０は１行目のロードユニットに対応する。６行目の記述２００６は、引数配列"args"に、配列"b"の先頭アドレスをbとし、要素数を１００とし、そして、入力座標２から配列"b"の値を入力することを設定するための記述である。ここで、入力座標２は３行目のロードユニットに対応する。７行目の記述２００７は、引数配列"args"に、配列"c"の先頭アドレスをcとし、要素数を１００とし、そして、入力座標３から配列"c"の値を入力することを設定するための記述である。ここで、入力座標３は４行目のロー

ドユニットに対応する。

【００９０】

また、８行目の記述２００８は、以上のようにして設定された配列"a","b","c"に関する情報をロードユニット用構成情報メモリに設定するための関数呼出しである。設定された配列"a","b","c"に関する情報は、配列毎にロードユニット用構成情報としてロードユニット用構成情報メモリに設定される。ここで、ロードユニット用構成情報は、対象となるロードユニットの行番号（Ｙ座標）、配列のＬＳ（Logical Space）アドレス先頭および配列のＬＳ回数を有する。したがって、配列"a","b","c"のロードユニット用構成情報は、それぞれ、行番号が０，２，３、ＬＳアドレス先頭がａ,ｂ,ｃ、ＬＳ回数が１００，１００,１００となる。

【００９１】

また、９行目の記述２００９は、引数配列"args"を初期化するための記述である。１０行目の記述２０１０は、引数配列"args"に、配列"x"の先頭アドレスをxとし、要素数を１００とし、そして、出力座標１から配列"x"の値を出力することを設定するための記述である。ここで、出力座標１は２行目のストアユニットに対応する。

【００９２】

また、１１行目の記述２０１１は、以上のようにして設定された配列"x"に関する情報をストアユニット用構成情報メモリに設定するための関数呼出しである。設定された配列"x"に関する情報は、配列毎にストアユニット用構成情報としてストアユニット用構成情報メモリに設定される。ここで、ストアユニット用構成情報は、対象となるストアユニットの行番号（Ｙ座標）、配列のＬＳアドレス先頭および配列のＬＳ回数を有する。したがって、配列"x"のストアユニット用構成情報は、それぞれ、行番号が１、ＬＳアドレス先頭がx、ＬＳ回数が１００となる。これにより、入力プログラムを実行した場合に、２行目のストアユニットから出力されたデータの１００要素（ＬＳ）分が、局所メモリ上の配列x[0]からx[99]に格納される。

【００９３】

また、１２行目の記述２０１２は、記述２００３で初期値設定された配列"conf1"に関する情報を演算ユニット用構成情報メモリに設定するための関数呼出しである。設定された配列"conf1"に関する情報は、図２に示すフォーマットを有する演算ユニット用構成情報として、同じ列番号（Ｘ座標）毎に演算ユニットの行数分（Ｍ個）並べて演算ユニット用構成情報メモリに設定される。

【００９４】

また、１３行目の記述２１０３は、動的再構成可能プロセッサをホストプロセッサから起動するための関数呼出しである。そして、１４行目の記述２０１４は動的再構成可能プロセッサの実行の終了をホストプロセッサ側から待つための関数呼出しである。ホストプロセッサは、動的再構成可能プロセッサの実行が終了し、この関数がリターンするまで、この関数呼出し以降の処理を実行することができない。

【００９５】

図１２は、構文解析部２１００の辞書作成処理を説明するためのフロー図である。

【００９６】

先ず、構文解析部２１００は、入力プログラム記憶部３１００に記憶されている入力プログラムから宣言されている未注目の配列に注目し、当該配列の型を特定する（Ｓ７０１）。例えば図１１に示す入力プログラムにおいて、配列"a"に注目した場合、その型は"char"となる。

【００９７】

次に、構文解析部２１００は、注目配列の要素（初期値）あるいは要素数を設定する記述に基づいて、該配列の大きさが固定である否かを調べると共に、配列の大きさに基づいて配列添字の下限値および上限値を決定する（Ｓ７０２）。例えば図１１に示す入力プログラムにおいて、記述２００５により配列"a"の要素数は１００であるので、配列"a"の大きさは固定と判断され、配列添字の下限値が０、上限値が要素数-１＝９９に決定される

。また、記述２００２により、配列"args"の大きさは固定でないと判断され、配列添字の下限値および上限値が共に０に決定される。

【００９８】

次に、構文解析部２１００は、注目配列に初期値が設定されているか否かを調べ、初期値が設定されている場合は、その初期値へのポインタを特定する（Ｓ７０３）。例えば図１１に示す入力プログラムにおいて、記述２００３により配列"conf1"に初期値が設定されているので、その初期値へのポインタが格納される。

【００９９】

以上のようにして、注目配列について、型、配列の大きさの固定の有無、配列添字の下限値および上限値、そして、初期値へのポインタを特定したならば、これらの情報を有する辞書（図６参照）を作成し、辞書記憶部３２００に記憶する（Ｓ７０４）。ここで、当該辞書の１つ前に作成した辞書があるならば（Ｓ７０５）、この１つ前に作成した辞書のフィールド１５０１に、今回作成した辞書へのポインタを格納する（Ｓ７０６）。

【０１００】

その後、構文解析部２１００は、入力プログラムで宣言されている全ての配列に着目したならば、このフローを終了し、そうでないならばＳ７０１に戻る（Ｓ７０７）。

【０１０１】

図１３は、図１１に示す入力プログラムに対して、図１２に示すフロー（辞書作成処理）を実行した結果、作成された辞書を説明するための図である。図示するように、入力プログラムに含まれている配列"a","b","c","x","args","conf1"のそれぞれについて辞書１５１０～１５６０が作成される。配列"conf1"には、記述２００３により配列"conf1"に初期値（演算ユニット用構成情報）１５６１が設定されているので、配列"conf1"の辞書１５６０には、初期値１５６１へのポインタが格納される。

【０１０２】

図１４は、構文解析部２１００の配置テーブル作成処理を説明するためのフロー図である。

【０１０３】

なお、配置テーブル記憶部３３００には、図７に示すような、入力プログラムが対象とする動的再構成可能プロセッサに対応する配置テーブルの雛形（各エントリ１６４０が空欄の状態、既存雛形テーブルと呼ぶ）が予め登録されているものとする。

【０１０４】

まず、構文解析部２１００は、入力プログラム記憶部３１００に記憶されている入力プログラムから、ロードユニット用構成情報メモリに設定する配列の内容を判定する（Ｓ７５１）。例えば図１１に示す入力プログラムでは、記述２００４～記述２００８により配列"a"が１行目のロードユニットに、配列"b"が３行目のロードユニットに、そして、配列"c"が４行目のロードユニットに設定されることを判定する。それから、構文解析部２１００は、ロードユニットのエントリ１６４０各々の設定情報のフィールド１６４７（図９参照）に、上述の辞書作成処理（図１２のフロー）で作成された当該ロードユニットに設定される配列の辞書へのポインタを登録する（Ｓ７５２）。

【０１０５】

次に、構文解析部２１００は、入力プログラムから、ストアユニット用構成情報メモリに設定する配列の内容を判定する（Ｓ７５３）。例えば図１１に示す入力プログラムでは、記述２００９～記述２０１１により配列"x"が２行目のストアユニットに設定されることを判定する。それから、構文解析部２１００は、ストアユニットのエントリ１６４０各々の設定情報のフィールド１６４７に、上述の辞書作成処理（図１２のフロー）で作成された当該ストアユニットに設定される配列の辞書へのポインタを登録する（Ｓ７５４）。

【０１０６】

次に、構文解析部２１００は、入力プログラムから、演算ユニット用構成情報メモリに設定する配列の情報を特定する（Ｓ７５５）。例えば図１１に示す入力プログラムでは、記述２００３、記述２０１２により演算ユニット用構成情報メモリに設定する配列"conf1

"の内容が"0x0a11b7…"であることを特定する。それから、図２に示すフォーマットに従い、特定した配列の情報に含まれている演算ユニット用構成情報各々を認識し、演算ユニットのエントリ１６４０各々に設定情報を登録する（Ｓ７５６）。

【０１０７】

具体的には、演算ユニット用構成情報各々について、該演算ユニット用構成情報のフィールド５３１、５３２に登録されている列番号および行番号に対応する演算ユニットのエントリ１６４０を特定し、該エントリ１６４０の設定情報のフィールド１６４１に、該演算ユニット用構成情報のフィールド５３３に登録されている演算種別を、そして、該設定情報のフィールド１６４２～１６４５、１６４８～１６５１に、該演算ユニット用構成情報のフィールド５３４～５３６の登録内容から特定されるオペランドの入力先および演算結果の出力先を登録する。なお、該エントリ１６４０の設定情報のフィールド１６４６に登録するサイクル数はＮＵＬＬとしておく。

【０１０８】

図１５は、図１１に示す入力プログラムに対して、図１４に示すフロー（配置テーブル作成処理）を実行した結果、作成された配置テーブルを説明するための図である。ここでは、入力プログラムの記述２００３で配列"conf1"に初期設定された演算ユニット用構成情報"0x0a11b7…"が、動的再構成可能プロセッサに次式で表される処理を行わせるものである場合を想定している。

【０１０９】

x[i] = a[i]*b[i] + b[i]*c[i] + c[i]

但し、x[i]は配列添字iで特定される配列"x"の構成要素であり、a[i]は配列添字iで特定される配列"a"の構成要素であり、b[i]は配列添字iで特定される配列"b"の構成要素であり、c[i]は配列添字iで特定される配列"c"の構成要素である。

【０１１０】

図１５において、入力列１６１０のエントリ１６４０に記述された記号ａ，ｂ，ｃは、対応するロードユニットに入力する配列"a","b","c"の辞書へのポインタである。また、出力列１６３０のエントリ１６４０に記述された記号xは、対応するストアユニットから出力する配列"x"の辞書へのポインタである（図１３参照）。

【０１１１】

また、配置列１６２０の各エントリ１６４０に記述された記号は、対応する演算ユニットに行わせる演算の演算種別を示している。記号Ｔは入力したデータを１サイクル後にそのまま出力するスルー命令を、記号＊は乗算した結果を１サイクル後に出力する乗算命令を、記号＋は加算した結果を１サイクル後に出力する加算命令を、そして、記号Ｄは入力したデータを２サイクルの遅延後にそのまま出力する遅延命令を、それぞれ示している。

【０１１２】

また、エントリ１６４０間の矢印は、データの入出力を示している。例えば、配置列１６２０の４行１列目のエントリ１６４０は、左側のエントリ１６４０（入力列１６１０の４行目）からデータを入力し、右側のエントリ１６４０（配置列１６２０の４行２列目）に出力することを示している。この場合、図９に示す設定情報は、左側のエントリ１６４０との結線の状態を示すフィールド１６５１が"connect"、その種類を示すフィールド１６４５が"in"となり、かつ、右側のエントリ１６４０との結線の状態を示すフィールド１６４９が"connect"、その種類を示すフィールド１６４３が"out"となる。

【０１１３】

なお、図１５において遅延命令を用いているのは、各演算で複数のオペランドが同じタイミングで入力できるように調整するためである。この調整によって、入力列１６１０の１行目のエントリ１６４０、３行目のエントリ１６４０および４行目のエントリ１６４０に、それぞれ、配列"a"の構成要素a[i]、配列"b"の構成要素b[i]および配列"c"の構成要素c[i]が同時入力されたときに、上述の式（x[i] = a[i]*b[i] + b[i]*c[i] + c[i]）が正しく計算され、７サイクル後に、出力列１６３０の２行目のエントリから演算結果である配列"x"の構成要素x[i]が出力される。

【０１１４】
図１６は、構成情報分割部２２００の処理を説明するためのフロー図である。
【０１１５】
先ず、構成情報分割部２２００は、予め登録された情報から、入力プログラムが対象とする動的再構成可能プロセッサの配置列１６２０の列数（演算ユニットの列数）Ｌを取得すると共に、出力プログラムが対象とする動的再構成可能プロセッサの配置列１６２０の列数Ｎを取得する。ここで、入力プログラムが対象とする動的再構成可能プロセッサが図２８に示す動的再構成可能プロセッサ９００であるのでＬ＝４となり、出力プログラムが対象とする動的再構成可能プロセッサが図１に示す第１実施形態の動的再構成可能プロセッサ９０であるのでＮ＝２となる。また、構成情報分割部２２００は、配列"conf1"に初期設定されている演算ユニット用構成情報へのポインタ"tmp0"をポインタ"tmp"の初期値とすると共に、配列"conf1"に対する辞書１５６０（図１３参照）を削除する。このとき、配列"conf1"に対する辞書１５６０のフィールド１５０３、１５０４の内容を保持しておく。さらに、カウント値ｐ、ｑを共に0に設定する（Ｓ１２０１）。
【０１１６】
次に、構成情報分割部２２００は、ｐ*NがＬより小さいか否かを調べることで、入力プログラムに含まれているＭ行Ｌ列の演算ユニットの構成情報から、Ｍ行Ｎ列の演算ユニットの構成情報を、新たに切出すことが可能か否かを判断する（Ｓ１２０２）。ｐ*NがＬより小さい場合は、Ｍ行Ｎ列の演算ユニットの構成情報を新たに切出すことが可能としてＳ１２０３に進む。一方、ｐ*NがＬ以上の場合はＳ１２０４に進む。
【０１１７】
Ｓ１２０３において、構成情報分割部２２００は、カウント値ｐを１つインクリメントする。それから、配列"comf_p"に対する辞書を作成する。そして、作成した辞書のフィールド１５０３、１５０４に削除した配列"conf1"に対する辞書のフィールド１５０３、１５０４の内容を登録する。また、構成情報分割部２２００は、１列当たりの演算ユニット用構成情報の大きさＫを特定し、このＫに基づいて配列添字の下限値（０）および上限値（０+N*Kバイトに相当する値）を決定して、作成した辞書のフィールド１５０５、１５０６に登録する。本実施形態では、１つの演算ユニットに対する演算ユニット用構成情報を２バイトとしている。また、１列につき４つの演算ユニットが配置されている。したがって、Ｋ＝８バイトとなる。次に、構成情報分割部２２００は、削除した配列"conf1"に対する辞書のフィールド１５０７にポイントされていた演算ユニット用構成情報のうち、ポインタ"tmp"が指示す位置からN*Kバイト分をコピーし、該コピーへのポインタを、作成した辞書のフィールド１５０７に登録する。それから、ポインタ"tmp"をN*Kバイト分インクリメントし、Ｓ１２０２に戻る。
【０１１８】
本実施形態では、Ｌ＝４、Ｎ＝２であるので、２つの配列"conf_1","conf_2"に対する辞書が作成される。
【０１１９】
次に、Ｓ１２０４において、構成情報分割部２２００は、入力プログラムの分割数を示す値numをカウント値ｐとする。また、ポインタ"tmp0"が指示す位置にある演算ユニット用構成情報（削除した配列"conf1"に対する辞書のフィールド１５０７にポイントされていた演算ユニット用構成情報）を削除する。次に、構成情報分割部２２００は、配置テーブルリストをnum個作成して順番を付す。また、配置列１６２０の列数をＮとする配置テーブル（分割配置テーブルと呼ぶ）をnum個作成し、順番を付して接続する。そして、配置テーブルリストのフィールド１７２０に同じ順番が付された分割配置テーブルへのポインタを登録すると共に、配置テーブルリストのフィールド１７１０に次の順番が付された分割配置テーブルへのポインタを登録する。また、作成した各分割配置テーブルの入力列１６１０の各エントリ１６４０のサイクル数を-1に初期化する。さらに、カウント値ｕを１に設定する。
【０１２０】

次に、構成情報分割部２２００は、ｕ番目の配置テーブルリストのフィールド１７３０に、配列"conf_u"に対する辞書へのポインタを登録する。また、構文解析部２１００が配置テーブル作成処理（図１４のフロー）により作成し、配置テーブル記憶部３３００に記憶した配置テーブルの配置列１６２０の第Ｎ*ｑ+1列目から第Ｎ*（ｑ+1）列目までを、ｕ番目の分割配置テーブルの配置列１６２０にコピーする（Ｓ１２０５）。

【０１２１】

次に、構成情報分割部２２００は、ｑが０か否かを調べることで、ｕ番目の分割配置テーブルが、構文解析部２１００が作成した配置テーブルの配置列１６２０の先頭列を含むものであるか否かを判断する（Ｓ１２０６）。ｑが０の場合は、先頭列を含むとしてＳ１２０７に進む。一方、ｑが０でない場合は、先頭列を含まないとしてＳ１２０８に進む。

【０１２２】

Ｓ１２０７において、構成情報分割部２２００は、構文解析部２１００が作成した配置テーブルの入力列１６１０を、ｕ番目の分割配置テーブルの入力列１６１０にコピーする。その後、Ｓ１２１０に進む。

【０１２３】

Ｓ１２０８において、構成情報分割部２２００は、ｑが値numと一致するか否かを調べることで、Ｕ番目の分割配置テーブルが、構文解析部２１００が作成した配置テーブルの配置列１６２０の最終列を含むものであるか否かを判断する。ｑが値numと一致する場合は、最終列を含むとしてＳ１２０９に進む。一方、ｑが値numと一致しない場合は、最終列を含まないとしてＳ１２１０に進む。

【０１２４】

Ｓ１２０９において、構成情報分割部２２００は、構文解析部２１００が作成した配置テーブルの出力列１６３０を、ｕ番目の分割配置テーブルの出力列１６３０にコピーする。この際、入力プログラムが対象とする動的再構成可能プロセッサの演算ユニットの列数Ｌが、出力プログラムが対象とする動的再構成可能プロセッサの演算ユニットの列数Ｎの倍数であるか否かを調べる。倍数である場合は、Ｓ１２１０に進む。一方、倍数でない場合は、出力列１６３０のエントリ１６４０のうち、配列が設定されているエントリ１６４０を特定する。次に、これと同行の、配置列１６２０のＳ+1列目からＮ列目までの各エントリ１６４０に、右側のエントリ１６４０（１列前の同行のエントリ１６４０）を入力、左側のエントリ１６４０（１列後の同行のエントリ１６４０）を出力、そして、演算種別をスルー命令とする設定情報を設定する。ここで、Ｓ＝ＬｍｏｄＮである。それから、Ｓ１２１０に進む。

【０１２５】

Ｓ１２１０において、構成情報分割部２２００は、カウント値ｑを１つインクリメントし、それから、カウント値ｕが値num未満であるか否かを調べる（Ｓ１２１１）。カウント値ｕが値num未満の場合は、未処理の配置テーブルリストが存在することを意味するので、カウント値uを１つインクリメントし（Ｓ１２１２）、その後、Ｓ１２０５に戻る。一方、カウント値ｕが値num以上の場合は、このフローを終了する。

【０１２６】

図１７は、図１３に示す辞書および図１５に示す配置テーブルに対して、図１６に示すフローを実行した結果、新たに作成された辞書を説明するための図である。図示するように、配列"conf1"に対する辞書１５６０が削除され、配列"conf_1","conf_2"に対する辞書１５６０ａ、１５６０ｂが追加されている。また、配列"conf1"からポイントされていた３２バイトの演算ユニット用構成情報１５６１は、前半１６バイトの演算ユニット用構成情報１５６１ａおよび後半１６バイトの演算ユニット用構成情報１５６１ｂに分割され、それぞれ、配列"conf_1","conf_2"からポイントされている。

【０１２７】

図１８は、図１３に示す辞書および図１５に示す配置テーブルに対して、図１６に示すフローを実行した結果、新たに作成された分割配置テーブルおよび配置テーブルリストを説明するための図である。出力プログラムが対象とする第１実施形態の動的再構成可能プ

ロセッサ９０の配置列数Ｎが２であるので、図１３に示す辞書および図１５に示す配置テーブルから、２つの配置テーブルリスト１７０１、１７０２と、２つの分割配置テーブル１６０１、１６０２が生成される。配置テーブルリスト１７０１、１７０２のフィールド１７１０には、それぞれ、分割配置テーブル１６０１、１６０２の先頭を指示すポインタが格納される。また、各分割配置テーブル１６０１、１６０２の入力列１６１０、出力列１６３０の各エントリ１６４０には、サイクル数「-１」が設定されている。

【０１２８】

分割配置テーブル１６０１の配置列１６２０は、図１５に示す配置テーブルの配置列１６２０の前半の２列を並べたものである。また、分割配置テーブル１６０１の出力列１６３０の各エントリ１６４０には、１列前（配置列１６２０の最終列）の同行のエントリ１６４０の出力が入力されるように、設定情報のフィールド１６５１に"connect"が登録され、フィールド１６４５に"in"が登録される（図９参照）。

【０１２９】

分割配置テーブル１６０２の配置列１６２０は、図１５に示す配置テーブルの配置列１６２０の後半の２列を並べたものである。また、分割配置テーブル１６０２の入力列１６１０の各エントリ１６４０には、順番上１つ前に位置する分割配置テーブル１６０１の配置列１６２０の最終列の各エントリ１６４０から、設定情報のフィールド１６４３、１６４９に登録されている１列後の同行のエントリ１６４０との配線状況をコピーしたものである。

【０１３０】

図１９は、遅延解析部２３００の処理を説明するためのフロー図である。

【０１３１】

先ず、遅延解析部２３００は、構成情報分割部２２００が作成した各分割配置テーブルの配置列１６２０中の各エントリ１６４０のサイクル数を０に初期化する。また、カウント値ｐを０に設定すると共に、カウント値ｕを１に設定する。また、構成情報分割部２２００が作成したｕ番目の配置テーブルリストが指示す分割配置テーブルをＵとする（Ｓ１３０１）。

【０１３２】

次に、遅延解析部２３００は、カウント値ｐを１つインクリメントする。それから、分割配置テーブルＵの配置列１６２０中の１行１列目のエントリ１６４０をＥ１とする（Ｓ１３０３）。そして、エントリＥ１の設定情報（フィールド１６４５、１６５１の情報）に基づいて、当該エントリＥ１の１列前（入力列１６１０）の同行のエントリ１６４０との配線状況が"connect"且つ"in"であるか否かを調べる（Ｓ１３０４）。

【０１３３】

Ｓ１３０４において、エントリＥ１の１列前の同行のエントリ１６４０との配線状況が"connect"且つ"in"でない場合、Ｓ１３０８に進む。一方、エントリＥ１の１列前の同行のエントリ１６４０との配線状況が"connect"且つ"in"である場合、カウント値ｐが１でないならば（Ｓ１３０５でＮｏ）、Ｓ１３０７に進み、カウント値ｐが１であるならば（Ｓ１３０５でＹｅｓ）、エントリＥ１の１列前の同行のエントリ１６４０のサイクル数を０に変更し（Ｓ１３０６）、それからＳ１３０７に進む。

【０１３４】

Ｓ１３０７において、遅延解析部２３００は、エントリＥ１のサイクル数を、エントリＥ１の１列前の同行のエントリ１６４０のサイクル数+1とし、Ｓ１３０８に進む。

【０１３５】

Ｓ１３０８において、遅延解析部２３００は、配置列１６２０の１列目に未処理（Ｅ１としていない）のエントリ１６４０が存在するか否かを調べ、存在する場合は、この未処理のエントリ１６４０をＥ１とし（Ｓ１３０９）、Ｓ１３０４に戻る。一方、存在しない場合は、Ｓ１３１０へ進む。

【０１３６】

図２０は、図１９のＳ１３０３～Ｓ１３０９の処理により、図１８に示す第１番目の分

割配置テーブル１６０１において、配置列１６２０の第１列目および入力列１６１０の各エントリ１６４０のサイクル数が更新された様子を示している。図示するように、配置列１６２０の１列目の１行目、３行目、４行目に、入力列からの入力がある。このため、これらのエントリ１６４０のサイクル数が０から１に更新され、これらと同行の入力列のエントリ１６４０のサイクル数が-１から０に更新される（更新部分Ａ）。

【０１３７】

さて、Ｓ１３１０において、遅延解析部２３００は、分割配置テーブルＵの配置列１６２０中の１行１列目のエントリ１６４０をＥ２とする（Ｓ１３０３）。次に、遅延解析部２３００は、エントリＥ２のサイクル数が０より大きいか否かを調べる（Ｓ１３１１）。０より大きい場合はＳ１３１２に進み、０以下の場合はＳ１３１３に進む。

【０１３８】

Ｓ１３１２において、遅延解析部２３００は、エントリＥ２の設定情報に基づいて、当該エントリＥ２に隣接するエントリ１６４０のうち、配線状況が"connect"且つ"状況がout"となるエントリ１６４０を特定し、特定した全てのエントリ１６４０のサイクル数をエントリＥ２のサイクル数+1に変更する。但し、特定したエントリ１６４０の演算種別が遅延命令の場合は、このエントリ１６４０のサイクル数を、エントリＥ２のサイクル数+１+遅延命令による遅延サイクル数に変更する。それから、エントリＥ２のサイクル数を０に変更する。その後、Ｓ１３１３に進む。

【０１３９】

Ｓ１３１３において、遅延解析部２３００は、配置列１６２０にエントリＥ２に続く未処理（Ｅ２としていない）のエントリ１６４０が存在するか否かを調べ、存在する場合は、この未処理のエントリ１６４０をＥ２とし（Ｓ１３１４）、Ｓ１３１１に戻る。一方、存在しない場合は、配置列１６２０に、サイクル数が０より大きいサイクル数を持つエントリ１６４０が存在するか否かを調べる（Ｓ１３１５）。存在する場合は、Ｓ１３１０に戻り、存在しない場合はＳ１３１６に進む。

【０１４０】

図２１は、図１９のＳ１３１０～Ｓ１３１４の処理により、図２０に示す第１番目の分割配置テーブル１６０１において、配置列１６２０の１列目の各エントリ１６４０が処理された様子を示している。配置列１６２０の１列目において、１行目、３行目、４行目のエントリ１６４０のサイクル数が１から０に変更され、２行目のエントリ１６４０のサイクル数が０から２へ変更される。また、配置列１６２０の２列目の２行目、３行目、４行目のエントリ１６４０のサイクル数がそれぞれ４、３、２に変更される。なお、２列目の２行目、３行目のエントリ１６４０は、演算種別が遅延命令であるので、１列目の同行のエントリの更新前のサイクル数+1に、遅延命令によるサイクル数が加えられる（更新部分Ｂ）。

【０１４１】

図２２は、図１９のＳ１３１０～Ｓ１３１５の処理により、図２０に示す第１番目の分割配置テーブル１６０１において、配置列１６２０の各エントリ１６４０が処理され、その結果、出力列１６３０のエントリ１６４０が更新された様子を示している。出力列１６３０の各エントリ１６４０のサイクル数は、１行目から順番に、-１、５、４、３となる。また、配置列１６２０中の全てのエントリ１６４０のサイクル数が０となる（更新部分Ｃ）。

【０１４２】

さて、Ｓ１３１６において、遅延解析部２３００は、分割配置テーブルＵの出力列１６３０の各エントリ１６４０のうち、-１以外のサイクル数を持つ各エントリ１６４０のサイクル数の最小値をｍに設定する。そして、前記-１以外のサイクル数を持つ各エントリ１６４０のサイクル数からｍを減算する。

【０１４３】

次に、遅延解析部２３００は、カウント値ｐが分割配置テーブルのテーブル数numと一致するか否かを調べ（Ｓ１３１７）。一致する場合は、分割配置テーブルＵの出力列１６

３０の内容を、次の分割配置テーブル（ｕ+1番目の分割配置テーブル）の入力列１６１０のコピーし（Ｓ１３１８）、それから、Ｓ１３１９に進む。一致しない場合は、直ちにＳ１３１９に進む。

【０１４４】

Ｓ１３１９において、遅延解析部２３００は、カウント値ｕが分割配置テーブルのテーブル数num未満であるか否かを調べる。カウント値ｕがテーブル数num未満の場合は、未処理の配置テーブルリストが存在することを意味するので、カウント値uを１つインクリメントして（Ｓ１３２０）、その後、Ｓ１３０３に戻る。一方、カウント値ｕがテーブル数num以上の場合は、このフローを終了する。

【０１４５】

図２３は、図１９のＳ１３１６～Ｓ１３１８の処理により、図２２に示す分割配置テーブルが処理された様子を示している。先ず、Ｓ１３１６により第１番目の分割配置テーブル１６０１の出力列１６３０の各エントリ１６４０のサイクル数が１行目から順に、-１、２、１、０に変更され（更新部分Ｄ）、まら、第２番目の分割配置テーブル１６０２の入力列１６１０に、第１番目の分割配置テーブル１６０１の出力列１６３０がコピーされている（更新部分Ｅ）。

【０１４６】

図２４は、図１９の処理により、図２２に示す第２番目の分割配置テーブル１６０２が処理された様子を示している。図示するように、第２番目の分割配置テーブル１６０２が対象となる場合、カウント値ｐ＝２であるので、Ｓ１３０６の処理は行われない。このため、第１番目の分割配置テーブル１６０１の出力列１６３０からコピーした内容はそのまま残る。それ以外は、第１番目の分割配置テーブル１６０１の場合と同様に処理される。その結果、配置列の各エントリ１６４０のサイクル数は０となり、出力列１６３０の各エントリ１６４０のサイクル数は、１行目から順に、-１、５、-１、-１となる（更新部分Ｆ）。

【０１４７】

図２５は、コード生成部２４００の処理を説明するためのフロー図である。

【０１４８】

先ず、コード生成部２４００は、遅延解析部２３００が作成した各分割配置テーブルの出力列１６３０において、０以上のサイクル数を持つエントリ１６４０のエントリ数の最大値を求め、これをｔｎとする。次に、一時配列"t_1"～"t_tn"各々に対する辞書（ｔｎ個の辞書）を作成する。これらの辞書の要素数data_sizeは、第１番目の分割配置テーブルの入力列１６１０に設定される配列の要素数（図１１に示す入力プログラムの場合は要素数１００）と同じに設定する。次に、辞書記憶部３２００に記憶されている構成報分割部２２００で生成された辞書から、関数名および引数の宣言を生成し、出力プログラムに追加する。また、該辞書にある引数以外の全ての変数および一時配列"t_1"～"t_tn"に対する宣言を生成し、出力プログラムに追加する。この際、配列"conf_i"に対する初期値として、配列"conf_i"に対する辞書からポイントされている構成情報を設定する（Ｓ１４０１）。

【０１４９】

例えば、構成報分割部２２００が生成した辞書が図１３に示すものであり、遅延解析部２３００が作成した分割配置テーブルが図２４に示すものである場合、値ｔｎは３となり、したがって一時配列"t_1"～"t_3"に対する辞書が作成され、その要素数data_sizeは１００となる。そして、Ｓ１４０１での処理の結果、出力プログラムとして、図２６に示す記述３００１～３００５が作成される。ここで、記述３００１は関数名および引数の宣言、記述３００２は引数配列の宣言、記述３００３は一時配列"t_1"～"t_3"の宣言、そして、記述３００４、３００５は、配列"conf_1","conf_2"各々に構成情報を初期設定するための記述である。

【０１５０】

次に、コード生成部２４００は、カウント値ｐ、ｔを共に１に初期化する。そして、構

成報分割部２２００が作成したｔ番目の配置テーブルリストが指示す分割配置テーブルをＴとする（Ｓ１４０２）。次に、コード生成部２４００は、カウント値ｐ＝１ならば（Ｓ１４０３でＹＥＳ）、引数配列"args"を初期化するための記述を生成し（図２６の記述３００６参照）、出力プログラムに追加する。また、分割配置テーブルＴの入力列１６１０の第１行目のエントリ１６４０をＧ１とする（Ｓ１４０４）。それから、Ｓ１４０５に進む。一方、カウント値ｐ＝１でないならば、Ｓ１４０９に進む。

【０１５１】

Ｓ１４０５において、コード生成部２４００は、エントリＧ１のサイクル数が０以上であるか否かを調べる。そして、０以上であるならば、構成報分割部２２００が生成した辞書からエントリＧ１にポイントされている配列の先頭アドレスn、要素数ｓを取得する。そして、引数配列"args"に、先頭アドレスを"n"とし、要素数を"ｓ"とし、入力座標ｌ（＝エントリＧ１の行番号）からエントリＧ１にポイントされている配列を入力し、且つ、エントリＧ１と同行の遅延ユニットに、エントリＧ１に設定されているサイクル数分遅延させて出力することを設定するための記述を生成し、出力プログラムに追加する（Ｓ１４０６）。その後、Ｓ１４０７に進む。一方、エントリＧ１のサイクル数が０未満の場合は、直ちにＳ１４０７に進む。

【０１５２】

Ｓ１４０７において、コード生成部２４００は、分割配置テーブルＴの入力列１６１０にエントリＧ１の次の行があるか否かを調べ、あるならば、この次の行のエントリ１６４０をＧ１とし（Ｓ１４０８）、Ｓ１４０５に戻る。

【０１５３】

例えば、構成報分割部２２００が生成した辞書が図１３に示すものであり、遅延解析部２３００が作成した分割配置テーブルが図２４に示すものである場合、Ｓ１４０５～Ｓ１４０８の処理により、図２６に示す記述３００７～３００９が作成される。ここで、記述３００７は、引数配列"args"に、配列"a"の先頭アドレスをaとし、要素数を１００とし、入力座標０から配列"a"の値を入力し、且つ、同行の遅延ユニットに０サイクル数分遅延させて出力することを設定するための記述である。記述３００８は、引数配列"args"に、配列"b"の先頭アドレスをbとし、要素数を１００とし、入力座標２から配列"b"の値を入力し、且つ、同行の遅延ユニットに０サイクル数分遅延させて出力することを設定するための記述である。そして、記述３００９は、引数配列"args"に、配列"c"の先頭アドレスをcとし、要素数を１００とし、入力座標３から配列"c"の値を入力し、且つ、同行の遅延ユニットに０サイクル数分遅延させて出力することを設定するための記述である。

【０１５４】

一方、Ｓ１４０７において、分割配置テーブルＴの入力列１６１０にエントリＧ１の次の行がない場合、コード生成部２４００は、Ｓ１４０５～Ｓ１４０８で設定した各配列に関する情報をロードユニット用構成情報メモリおよび遅延ユニット用構成情報メモリに設定するための関数呼出しを生成し（図２６の記述３０１０参照）、出力プログラムに追加する。また、引数配列"args"を初期化するための記述を生成し（図２６の記述３０１１参照）、出力プログラムに追加する。それから、分割配置テーブルＴの出力列１６３０の第１行目のエントリ１６４０をエントリＧ２とする（Ｓ１４０９）。その後、Ｓ１４１０に進む。

【０１５５】

Ｓ１４１０において、コード生成部２４００は、エントリＧ２のサイクル数が０以上であるか否かを調べる。そして、０以上であるならば、配列"t_w"の先頭アドレスn、要素数ｓを取得する。ここで、値wは、サイクル数が０以上であるエントリＧ２の出現回数である。次に、コード生成部２４００は、引数配列"args"に、先頭アドレスを"n"とし、要素数を"ｓ"とし、出力座標ｌ（＝エントリＧ２の行番号）から配列"t_w"が、エントリＧ２のサイクル数分遅延して出力することを設定するための記述を生成し、出力プログラムに追加する（Ｓ１４１１）。その後、Ｓ１４１２に進む。なお、分割配置テーブルＴに次の分割配置テーブルがある場合、その入力列１６１０には、分割配置テーブルＴの出力列１

６３０の内容がコピーされている。したがって、分割配置テーブルＴの出力列１６３０のエントリＧ２に対する記述が、次の分割配置テーブルの入力列１６１０のエントリＧ２と同行のエントリ１６４０に対する記述としても機能する。つまり、次の分割配置テーブルの入力列１６１０のエントリＧ２と同行のエントリ１６４０に対しては、引数配列"args"に、先頭アドレスを"n"とし、要素数を"ｓ"とし、入力座標ｌ（＝エントリＧ２の行番号）から配列"t_w"を入力し、エントリＧ２と同行の遅延ユニットに、当該エントリＧ２のサイクル数分遅延させて出力することを設定するための記述として機能する。一方、エントリＧ２のサイクル数が０未満の場合は、直ちにＳ１４１２に進む。

【０１５６】

Ｓ１４１２において、コード生成部２４００は、分割配置テーブルＴの出力列１６３０にエントリＧ２の次の行があるか否かを調べ、あるならば、この次の行のエントリ１６４０をＧ２とし（Ｓ１４１３）、Ｓ１４１０に戻る。

【０１５７】

例えば、遅延解析部２３００が作成した分割配置テーブルが図２４に示すものである場合、Ｓ１４１０～Ｓ１４１３の処理により、図２６に示す記述３０１２～３０１４が作成される。ここで、記述３０１２は、引数配列"args"に、配列"t_1"の先頭アドレスをt_1とし、要素数を１００とし、出力座標０から配列"t_1"の値が２サイクル数分遅延して出力することを設定するための記述である。記述３０１３は、引数配列"args"に、配列"t_2"の先頭アドレスをt_2とし、要素数を１００とし、出力座標２から配列"t_2"の値が１サイクル数分遅延して出力することを設定するための記述である。そして、記述３０１４は、引数配列"args"に、配列"t_3"の先頭アドレスをt_3とし、要素数を１００とし、出力座標３から配列"t_3"の値が０サイクル数分遅延して出力することを設定するための記述である。

【０１５８】

一方、Ｓ１４１２において、分割配置テーブルＴの出力列１６３０にエントリＧ２の次の行がない場合、コード生成部２４００は、Ｓ１４１０～Ｓ１４１３で設定した各配列に関する情報をストアユニット用構成情報メモリに設定するための関数呼出しを生成し（図２６の記述３０１５参照）、出力プログラムに追加する。また、初期値設定された配列"conf_t"に関する情報を演算ユニット用構成情報メモリに設定するための関数呼出しを生成し（図２６の記述３０１６参照）、出力プログラムに追加する。さらに、動的再構成可能プロセッサをホストプロセッサから起動するための関数呼出しと、動的再構成可能プロセッサの実行の終了をホストプロセッサ側から待つための関数呼出しとを生成し（図２６の記述３０１７、３０１８参照）、出力プログラムに追加する（Ｓ１４１３）。

【０１５９】

次に、コード生成部２４００は、カウント値ｔが分割配置テーブルのテーブル数num未満であるか否かを調べる（Ｓ１４１４）。カウント値ｔがテーブル数num未満の場合は、未処理の配置テーブルリストが存在することを意味するので、カウント値ｐ、ｔをそれぞれ１つインクリメントして（Ｓ１４１５）、その後、Ｓ１４０３に戻る。一方、カウント値ｔがテーブル数num以上の場合は、このフローを終了する。

【０１６０】

例えば、遅延解析部２３００が作成した分割配置テーブルが図２４に示すものである場合、２番目の配置テーブルリスト１７０２が指示す分割配置テーブル１６０２を処理するべく、Ｓ１４０３～Ｓ１４１５が２回実行される。しかし、２回目の処理では、Ｓ１４０３でｐ＝２でるため、Ｓ１４０９以降のみが実行され、Ｓ１４０４～Ｓ１４０８は実行されない。その結果、図２６に示すように、記述３０１０～３０１８に相当する記述３０１９～３０２５のみが出力プログラムに追加され、記述Ｓ３００６～Ｓ３００９に相当する記述は追加されない。

【０１６１】

図２７（Ａ）は、図２６に示す出力プログラムを、図１に示す第１実施形態の動的再構成可能プロセッサ９０に実行させた場合のタイムチャートを示しており、図２７（Ｂ）は

、図１１に示す入力プログラムを図２８に示す既存の動的再構成可能プロセッサ９００に実行させた場合のタイムチャートを示している。

【０１６２】

入力プログラムの場合、図１５に示す構文解析直後の配置テーブルから分かるように、入力データが７サイクルで出力される。このため、図２７（Ｂ）に示すように、配列"a","b","c"の各要素a[1], ｂ[1], c[1]は、第７サイクルで配列"x"の要素x[1]に格納される（Ｔ４０００）。同様にして、a[100],b[100],c[100]は第１０６サイクルでx[100]に格納される（Ｔ４００５）。

【０１６３】

一方、出力プログラムの場合、図２２に示す遅延解析中の配置テーブルから分かるように、１番目の分割配置テーブルの出力列におけるサイクル数の最大値が５なので、図２７（Ａ）に示すように、配列"a","b","c"の各要素a[1], ｂ[1], c[1]は、第５サイクルで各々配列"t_1","t_2","t_3"の要素t_1[1],t_2[2],t_3[3]（局所メモリ４０）に格納される（Ｔ４０１０）。同様にして、a[100],b[100],c[100]は第１０４サイクルでt_1[100],t_2[100],t_3[100]に格納される（Ｔ４０１５）。そして、この直後に、動的再構成可能プロセッサの構成情報が、１番目の分割配置テーブルに基づくものから２番目の分割配置テーブルに基づくものに変更される（Ｔ４１００）。図２７（Ａ）に示す例では、第１０５サイクルで開始して、第１０６サイクルで完了したことを表わしている。

【０１６４】

次に、t_1[1], t_2[1], t_3[1]は１サイクルで局所メモリ４０からロードされ、第１０７サイクルでロードユニット６１に入力される。図２４に示すように、２番目の分割配置テーブルの入力列１６１０の第４行目に０サイクルの遅延で入力されたデータが５サイクルで出力列１６３０の第２行目に到達する。したがって、第１０７サイクルに入力されたt_1[1], t_2[1], t_3[1]、は第１１２サイクルで配列"x"の要素x[1]に格納される（Ｔ４２００）。同様にして、t_1[100],t_2[100],t_3[100]は第２１１サイクルでx[100]に格納される（Ｔ４２０５）。

【０１６５】

このように、本実施形態のコンパイラ装置によれば、第１実施形態の動的再構成可能プロセッサよりの演算ユニットの列数の多い既存の動的再構成可能プロセッサ向けのプログラムを有効利用して、第１実施形態の動的再構成可能プロセッサ向けのプログラムを自動的に得ることができる。

【図面の簡単な説明】

【０１６６】

【図１】図１は本発明の第１実施形態が適用された動的再構成可能プロセッサを有する情報処理装置の概略図である。

【図２】図２は演算ユニット用構成情報メモリ５３に格納される演算ユニット用構成情報の一例を示す図である。

【図３】図３は図１に示す遅延ユニット６２の回路構成例を示す図である。

【図４】図４は図１に示す遅延ユニット用構成情報メモリ５２の回路構成例を示す図である。

【図５】図５は本発明の第２実施形態が適用されたコンパイラ装置の概略図である。

【図６】図６は辞書記憶部３２００に記憶される辞書の一例を示す図である。

【図７】図７は配置テーブル記憶部３３００に記憶される配置テーブルの一例を示す図である。

【図８】図８は図７に示す配置テーブルのエントリ１６４０に格納される設定情報を説明するための図である。

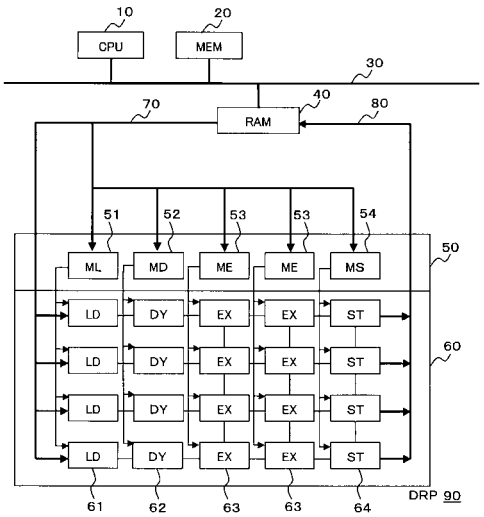
【図９】図９は配置テーブル記憶部３３００に記憶される配置テーブルリストの一例を示す図である。

【図１０】図１０は図５に示すコンパイラ装置１０００のハードウエア構成例を示す図である。

【図１１】図１１は入力プログラムの一例を示す図である。
【図１２】図１２は構文解析部２１００の辞書作成処理を説明するためのフロー図である。
【図１３】図１３は図１１に示す入力プログラムに対して、図１２に示すフローを実行した結果、作成された辞書を説明するための図である。
【図１４】図１４は構文解析部２１００の配置テーブル作成処理を説明するためのフロー図である。
【図１５】図１５は図１１に示す入力プログラムに対して、図１４に示すフローを実行した結果、作成された配置テーブルを説明するための図である。
【図１６】図１６は構成情報分割部２２００の処理を説明するためのフロー図である。
【図１７】図１７は図１３に示す辞書および図１５に示す配置テーブルに対して、図１６に示すフローを実行した結果、新たに作成された辞書を説明するための図である。
【図１８】図１８は図１３に示す辞書および図１５に示す配置テーブルに対して、図１６に示すフローを実行した結果、新たに作成された分割配置テーブルおよび配置テーブルリストを説明するための図である。
【図１９】図１９は遅延解析部２３００の処理を説明するためのフロー図である。
【図２０】図２０は図１９のＳ１３０３～Ｓ１３０９の処理により、図１８に示す第１番目の分割配置テーブル１６０１において、配置列１６２０の第１列目および入力列１６１０の各エントリ１６４０のサイクル数が更新された様子を示す図である。
【図２１】図２１は図１９のＳ１３１０～Ｓ１３１４の処理により、図２０に示す第１番目の分割配置テーブル１６０１において、配置列１６２０の１列目の各エントリ１６４０が処理された様子を示す図である。
【図２２】図２２は図１９のＳ１３１０～Ｓ１３１５の処理により、図２０に示す第１番目の分割配置テーブル１６０１において、配置列１６２０の各エントリ１６４０が処理され、その結果、出力列１６３０のエントリ１６４０が更新された様子を示す図である。
【図２３】図２３は図１９のＳ１３１６～Ｓ１３１８の処理により、図２２に示す分割配置テーブルが処理された様子を示す図である。
【図２４】図２４は図１９の処理により、図２２に示す第２番目の分割配置テーブル１６０２が処理された様子を示す図である。
【図２５】図２５はコード生成部２４００の処理を説明するためのフロー図である。
【図２６】図２６は出力プログラムの一例を示す図である。
【図２７】図２７（Ａ）は、図２６に示す出力プログラムを、図１に示す第１実施形態の動的再構成可能プロセッサ９０に実行させた場合のタイムチャートであり、また、図２７（Ｂ）は、図１１に示す入力プログラムを図２８に示す既存の動的再構成可能プロセッサ９００に実行させた場合のタイムチャートである。
【図２８】図２８は従来の動的再構成可能プロセッサを有する情報処理装置の概略図である。
【符号の説明】
【０１６７】
１０・・・ホストプロセッサ、２０・・・メモリ、３０・・・バス、４０・・・局所メモリ、５０・・・構成情報格納部、５１・・・ロードユニット用構成情報メモリ、５２・・・遅延ユニット用構成情報メモリ、５３・・・演算ユニット用構成情報メモリ、５４・・・ストアユニット用構成情報メモリ、６０・・・演算部、６１・・・ロードユニット、６２・・・遅延ユニット、６３・・・演算ユニット、６４・・・ストアユニット、２０００・・・演算部、２１００・・・構文解析部、２２００・・・構成情報分割部、２３００・・・遅延解析部、２４００・・・コード生成部、３０００・・・記憶部、３１００・・・入力プログラム記憶部、３２００・・・辞書記憶部、３３００・・・配置テーブル記憶部、３４００・・・出力プログラム記憶部、４０００・・・入出力部

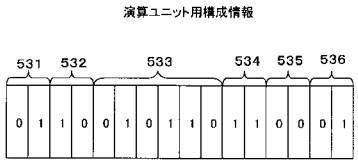
【図１】

図1



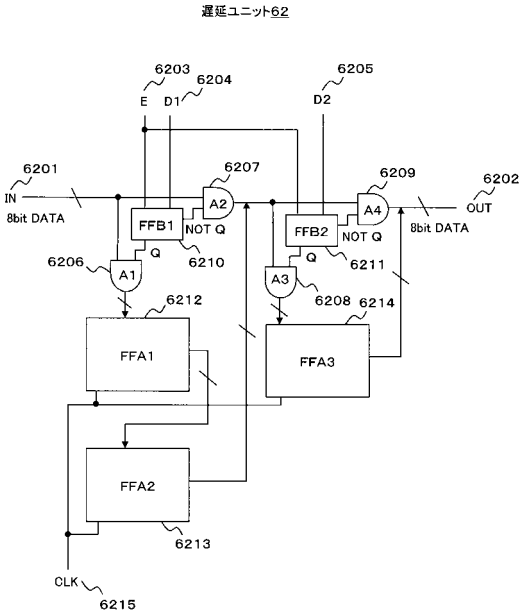
【図２】

図2



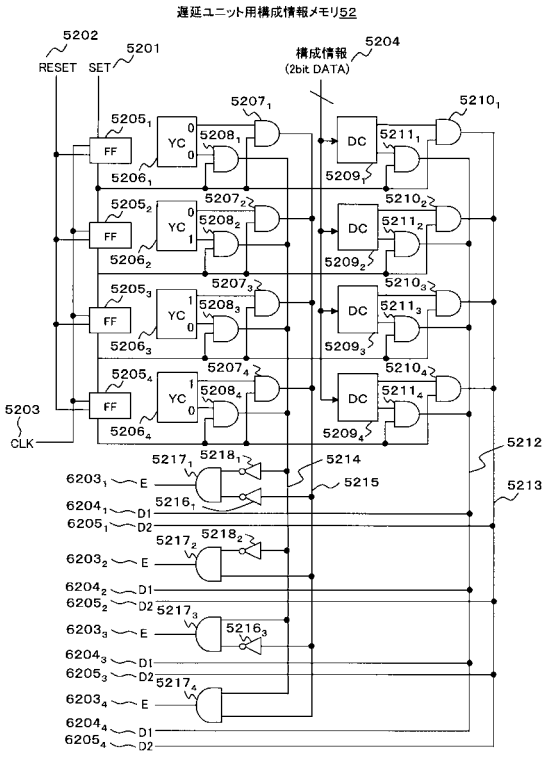
【図３】

図3



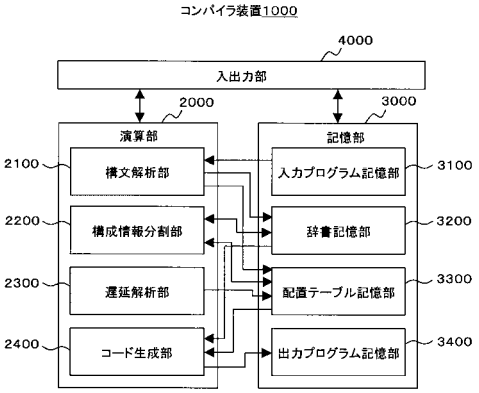
【図４】

図4



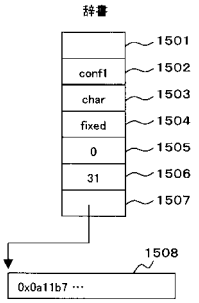
【図５】

図5



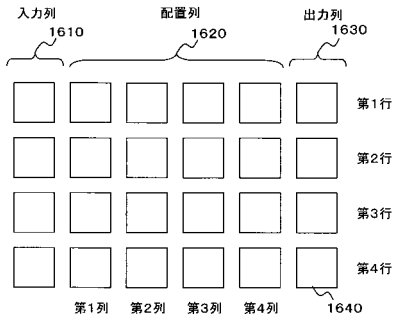
【図６】

図6



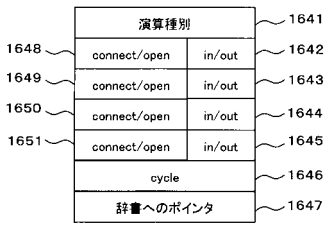
【図７】

図7



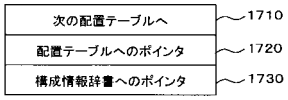
【図８】

図8



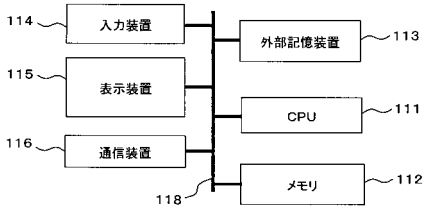
【図９】

図9



【図１０】

図10



【図１１】

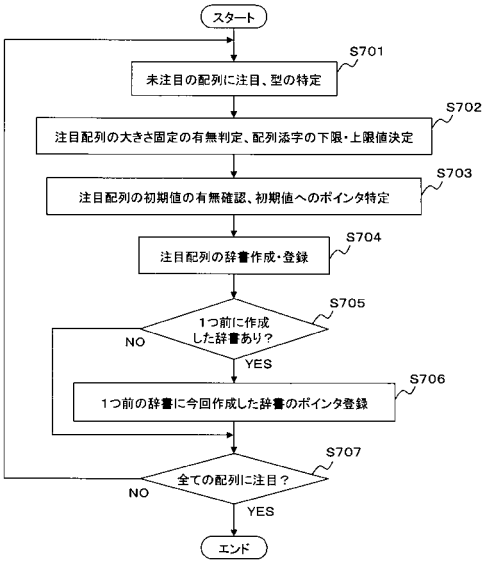
図11

入力プログラム

```
2001 void fnrp (char a[100], char b[100], char c[100], char x[100]) {
2002 void *args;
2003 char conf1={0x0a11b7…};
2004 initarg (args);
2005 setarg (args, a, 100, 0);
2006 setarg (args, b, 100, 2);
2007 setarg (args, c, 100, 3);
2008 setload (args);
2009 initarg (args);
2010 setarg (args, x, 100, 1);
2011 setstore (args);
2012 setconf (conf1);
2013 execrp ();
2014 waitrp (); }
```

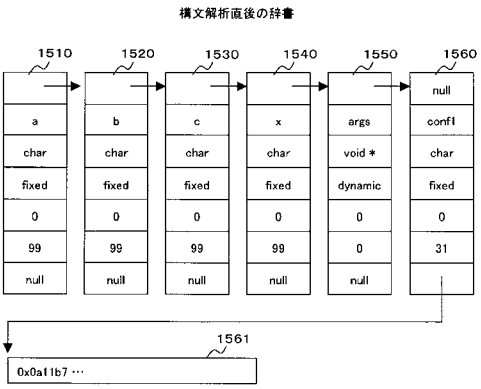
【図１２】

図12



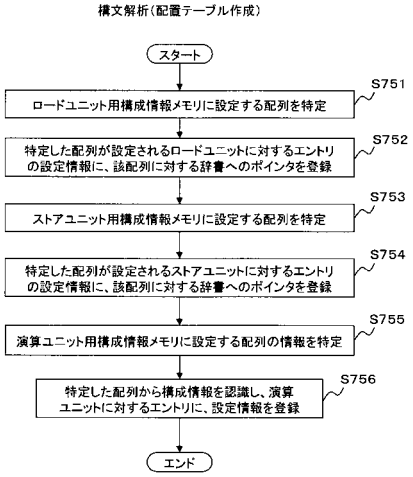
【図１３】

図13



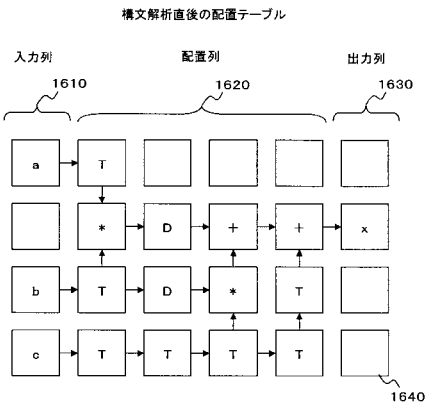
【図１４】

図14



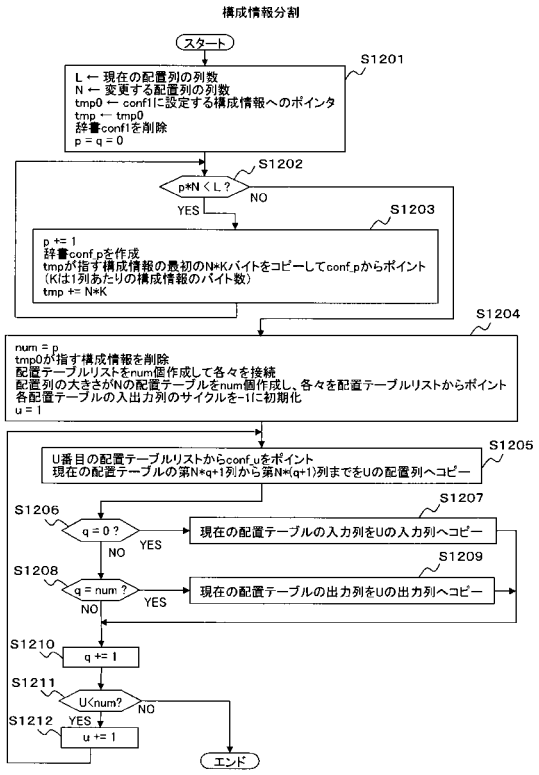
【図１５】

図15



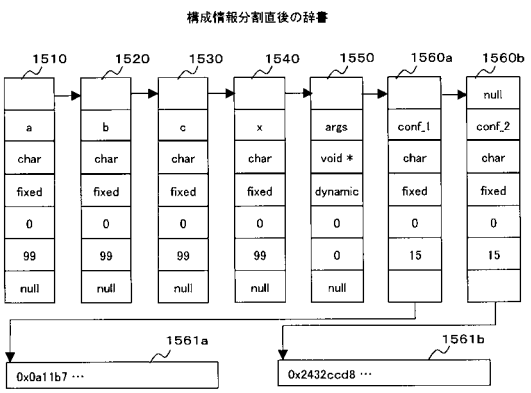
【図１６】

図16



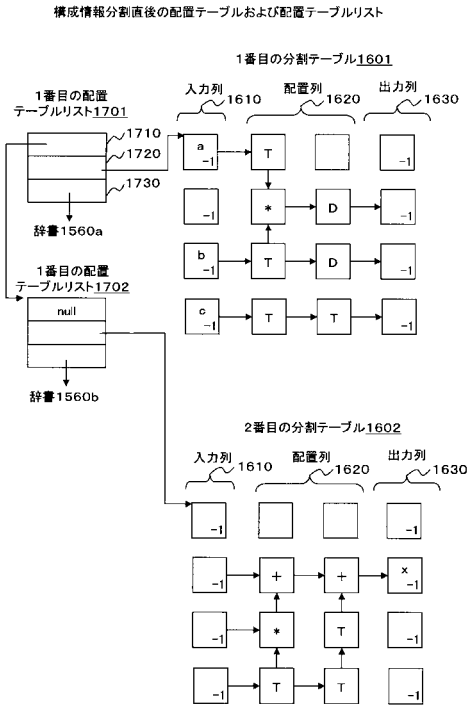
【図１７】

図17



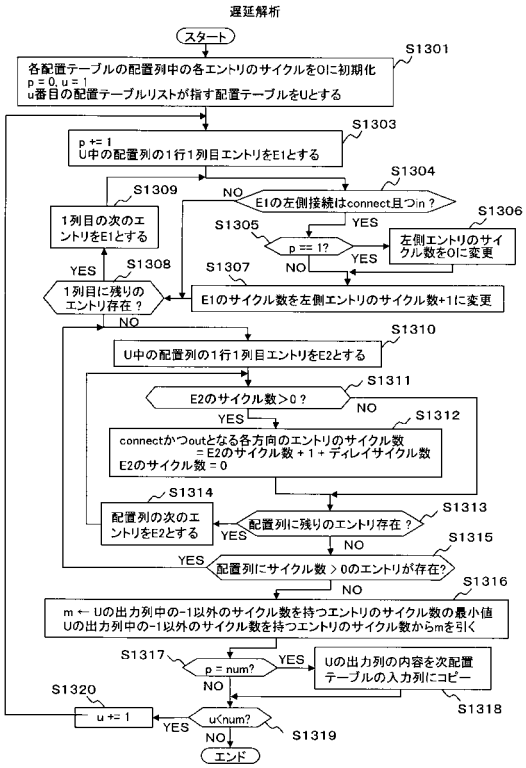
【図１８】

図18



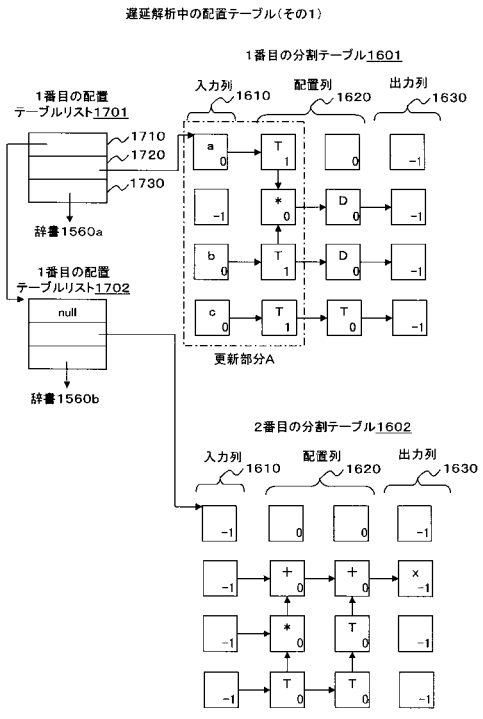
【図１９】

図19



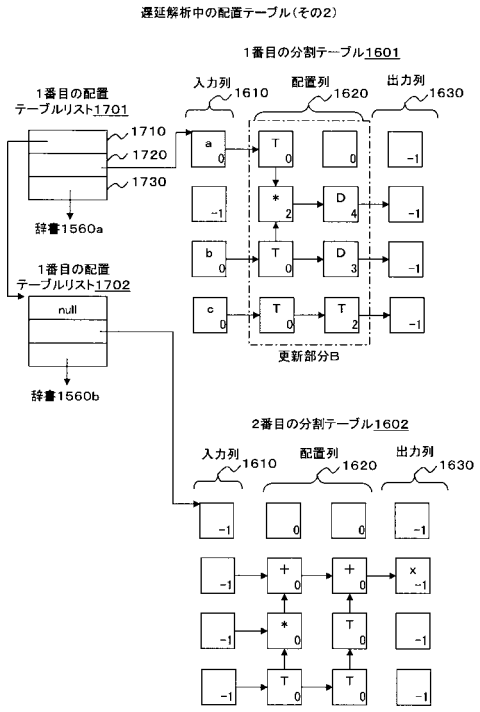
【図２０】

図20



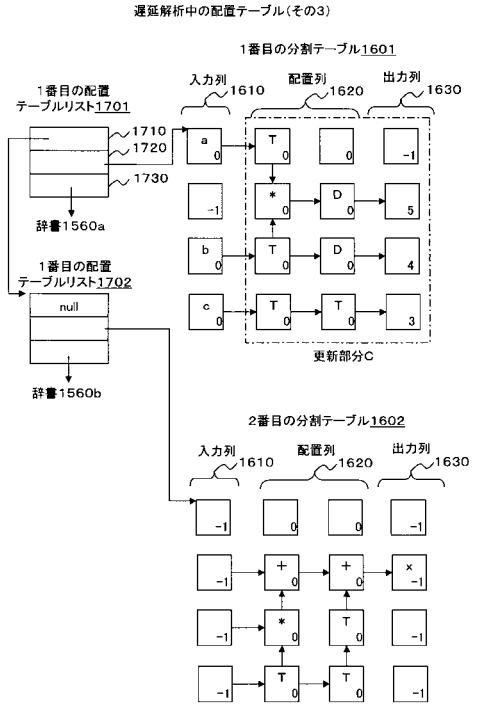
【図２１】

図21



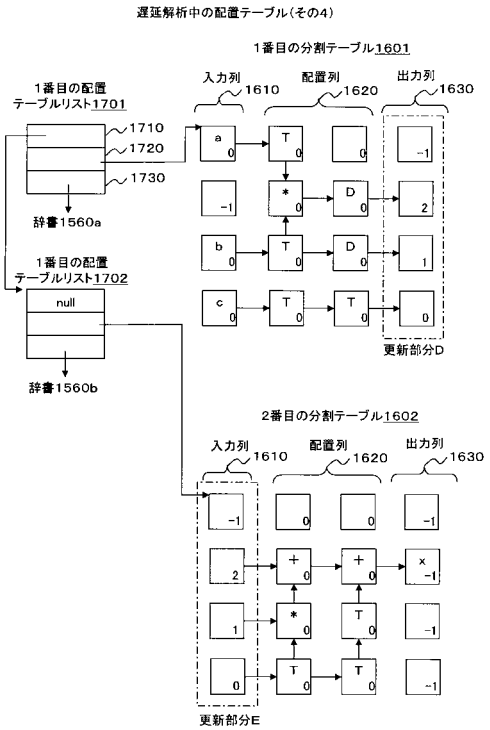
【図２２】

図22



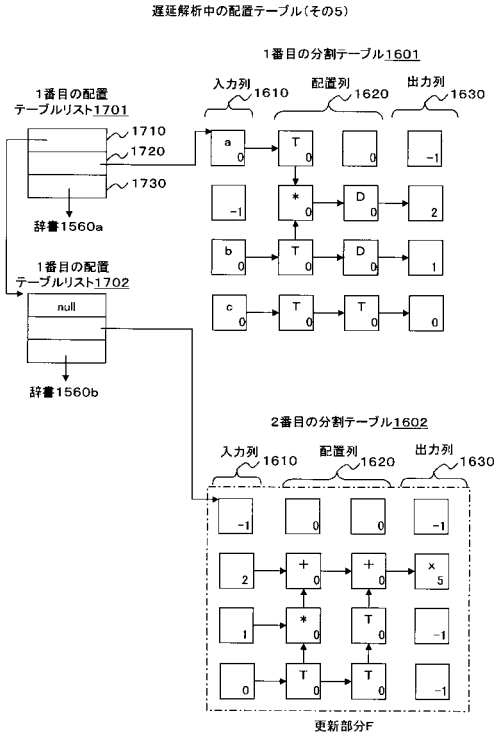
【図２３】

図23



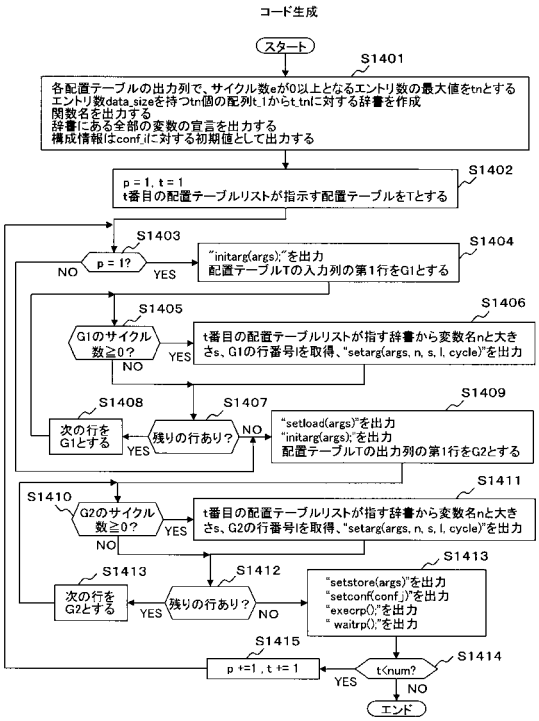
【図２４】

図24



【図２５】

図25



【図２６】

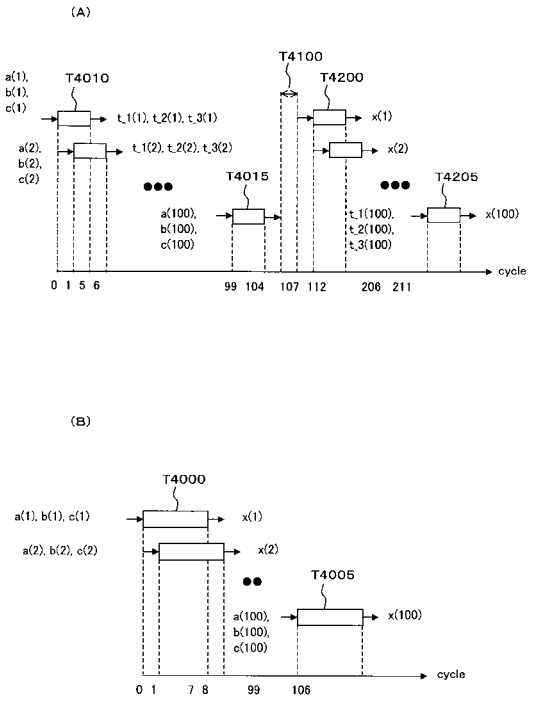
図26

出力プログラム

```
3001 void fnrp (char a[100], char b[100], char c[100], char x[100]) {
3002 void *args;
3003 char t_1[100], t_2[100], t_3[100];
3004 char conf_1 = {0x000a11b7…};
3005 char conf_2 = {0x2432ccd8…};
3006 initarg (args);
3007 setarg (args, a, 100, 0, 0);
3008 setarg (args, b, 100, 2, 0);
3009 setarg (args, c, 100, 3, 0);
3010 setload (args);
3011 initarg (args);
3012 setarg (args, t_1, 100, 1, 2);
3013 setarg (args, t_2, 100, 2, 1);
3014 setarg (args, t_3, 100, 3, 0);
3015 setstore (args);
3016 setconf (conf_1);
3017 execrp ();
3018 waitrp ();
3019 setload (args);
3020 initarg (args);
3021 setarg (args, x, 100, 1, 0);
3022 setstore (args);
3023 setconf (conf_2);
3024 execrp ();
3025 waitrp (); }
```

【図２７】

図27



【図２８】

図28