(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4558879号
(P4558879)
(45)発行日　平成22年10月6日(2010.10.6)　(24)登録日　平成22年7月30日(2010.7.30)

(51)Int.Cl.　F I
G06F 15/82 (2006.01)　G06F 15/82 610Q

請求項の数 28 (全 31 頁)

(21)出願番号　特願2000-36874(P2000-36874)
(22)出願日　平成12年2月15日(2000.2.15)
(65)公開番号　特開2001-229172(P2001-229172A)
(43)公開日　平成13年8月24日(2001.8.24)
審査請求日　平成19年2月2日(2007.2.2)

（出願人による申告）国等の委託研究の成果に係る特許出願（平成１１年度通産省委託事業「エネルギー使用合理化電子計算機技術開発」委託研究、産業活力再生特別措置法第３０条の適用を受けるもの）

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100074099
弁理士　大菅　義之
(74)代理人　100067987
弁理士　久木元　彰
(72)発明者　陣崎　明
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

審査官　漆原　孝治

最終頁に続く

(54)【発明の名称】テーブルを用いたデータ処理装置および処理システム

(57)【特許請求の範囲】
【請求項１】
入力されるデータをメモリ検索用データに変換する入力変換手段と、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理の状態に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理の状態に対応するデータが格納されている状態遷移テーブルへアクセスするためのアクセス情報とを含むデータを表わす状態語が１つのエントリに格納されている状態遷移テーブル内の状態語を読み出すメモリ検索手段と、
該読み出された状態語の内容に基づいて、前記アクセス情報を得ると同時に、該読み出された状態語の内容に応じた演算を入力されたデータに対して実行する演算手段とを備えﾝ
前記入力変換手段は、前記演算手段が演算を実行した出力データに基づいて、入力データをメモリ検索用データに変換する方法を決定することを特徴とするテーブルを用いたデータ処理装置。
【請求項２】
入力されるデータをメモリ検索用データに変換する入力変換手段と、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理の状態に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理の状態に対応するデータが格納されている状態遷移テーブルへアクセスするためのアクセス情報とを含むデータを表わす状態語が１つのエントリに格納されている状態遷移テ

ーブル内の状態語を読み出すメモリ検索手段と、
該読み出された状態語の内容に基づいて、前記アクセス情報を得ると同時に、該読み出された状態語の内容に応じた演算を入力されたデータに対して実行する演算手段と、
前記メモリ内の状態遷移テーブルの内容をデータ処理前、あるいは処理中に変更する状態遷移テーブル書き換え手段と、
を備えることを特徴とするテーブルを用いたデータ処理装置。
【請求項３】
入力されるデータをメモリ検索用データに変換する入力変換手段と、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理の状態に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理の状態に対応するデータが格納されている状態遷移テーブルへアクセスするためのアクセス情報とを含むデータを表わす状態語が１つのエントリに格納されている状態遷移テーブル内の状態語を読み出すメモリ検索手段と、
該読み出された状態語の内容に基づいて、前記アクセス情報を得ると同時に、該読み出された状態語の内容に応じた演算を入力されたデータに対して実行する演算手段と、
を備え、
前記状態語が、該状態語に対応する入力に関するデータを格納する領域、前記状態遷移先へのアクセス情報を格納する領域、前記演算の内容を格納する領域、演算結果の出力に関するデータを格納する領域、および該状態語にこれらの４つの領域のいずれが含まれるかを示すタグ領域のうちで少なくとも該４つの領域の中の１つ以上の領域を含み、１つ以上の領域を備えることを特徴とするテーブルを用いたデータ処理装置。
【請求項４】
前記状態語が前記タグ領域を必ず備えることを特徴とする請求項３記載のテーブルを用いたデータ処理装置。
【請求項５】
前記状態語が、前記４つの領域のうちでいずれの領域が備えられるかによって、それぞれ一定の長さを、持つことを特徴とする請求項３記載のテーブルを用いたデータ処理装置。
【請求項６】
前記状態語が備える１つ以上の領域の種類によって決定される状態語の領域構成毎にそれぞれ対応する前記タグを格納する複数のタグレジスタを更に備え、前記メモリ検索手段が状態後のアクセスに際して、該状態語のタグがどのタグレジスタに格納されているかによって該状態語の領域構成を認識することを特徴とする請求項３記載のテーブルを用いたデータ処理装置。
【請求項７】
前記入力変換手段が、前記入力されるデータを、前記状態語を読み出すためのメモリ検索用データであって、該入力されるデータよりビット数の少ないメモリ検索用データに変換することを特徴とする請求項１記載のテーブルを用いたデータ処理装置。
【請求項８】
前記入力変換手段が、前記入力されるデータ内で任意の位置にある複数のビットを抽出し、該入力データに対応するビット配置上で任意の連続した位置に抽出されたビットをまとめるマスクアンドギャザー処理を実行することを特徴とする請求項７記載のテーブルを用いたデータ処理装置。
【請求項９】
入力されるデータをメモリ検索用データに変換する入力変換手段と、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理の状態に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理の状態に対応するデータが格納されている状態遷移テーブルへアクセスするためのアクセス情報とを含むデータを表わす状態語が１つのエントリに格納されている状態遷移テーブル内の状態語を読み出すメモリ検索手段と、

該読み出された状態語の内容に基づいて、前記アクセス情報を得ると同時に、該読み出された状態語の内容に応じた演算を入力されたデータに対して実行する演算手段と、
を備え、
前記メモリ検索手段が、前記入力変換手段の出力と前記メモリのベースアドレスとを合成して前記読み出すべき状態語のアドレスを求めると共に、該アドレスに格納されている状態語にメモリアドレスが含まれている時、該メモリアドレスを次に行うべきメモリ検索におけるメモリベースアドレスとして使用することを特徴とするテーブルを用いたデータ処理装置。
【請求項１０】
前記演算手段が、
演算に必要なデータを格納するレジスタ手段と、
該レジスタ手段に格納されているデータと前記入力されたデータとを用いて演算を実行する演算処理手段とを備えることを特徴とする請求項１記載のテーブルを用いたデータ処理装置。
【請求項１１】
前記演算処理手段が、前記入力されたデータを演算処理する間は演算結果の出力を遅延させるためのＦＩＦＯメモリ手段を更に備えることを特徴とする請求項１０記載のテーブルを用いたデータ処理装置。
【請求項１２】
前記演算手段が、入力データに対応する演算の結果を格納するテーブルを更に備え、
前記演算処理手段が入力データに対応して該テーブルを検索し、演算結果を出力することを特徴とする請求項１０記載のテーブルを用いたデータ処理装置。
【請求項１３】
前記演算手段が、
前記入力されたデータを一時的に格納する入力ＦＩＦＯメモリ手段と、
前記演算処理手段の演算結果を一時的に格納する出力ＦＩＦＯメモリ手段とを更に備えることを特徴とする請求項１０記載のテーブルを用いたデータ処理装置。
【請求項１４】
入力されるデータをメモリ検索用データに変換する入力変換手段と、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理の状態に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理の状態に対応するデータが格納されている状態遷移テーブルへアクセスするためのアクセス情報とを含むデータを表わす状態語が１つのエントリに格納されている状態遷移テーブル内の状態語を読み出すメモリ検索手段と、
該読み出された状態語の内容に基づいて、前記アクセス情報を得ると同時に、該読み出された状態語の内容に応じた演算を入力されたデータに対して実行する演算手段と、
を備え、
前記演算手段が、
演算に必要なデータを格納するレジスタ手段と、
該レジスタ手段に格納されているデータと前記入力されたデータとを用いて演算を実行する演算処理手段と、
を備え、
前記演算処理手段が、入力されたデータのデータ構造に対応する複数の領域を有するレジスタ手段を備え、
入力されたデータのデータ構造に対応して該入力されたデータを分割して該複数の領域に格納し、該格納結果に対応して該格納されたデータの全部、または一部に対して所定の処理を実行することを特徴とするテーブルを用いたデータ処理装置。
【請求項１５】
前記所定の処理が、前記格納されたデータがあらかじめ定められたデータ構造のパターンに一致するか否かを判定する処理であることを特徴とする請求項１４記載のテーブルを

用いたデータ処理装置。
【請求項１６】
前記所定の処理が、前記複数の領域のうちの１つ以上の領域の格納内容を抽出し、該抽出されたデータをあらかじめ定められた形式のデータとしてまとめ、入力されたデータの特徴を示す情報を求める処理であることを特徴とする請求項１４記載のテーブルを用いたデータ処理装置。
【請求項１７】
前記演算手段が、前記入力されたデータを、演算処理に必要なデータを格納しているメモリを検索するためのデータであって、該入力されたデータよりビット数の少ないメモリ検索用データに変換する演算処理用データ格納メモリ検索データ作成手段を更に備えることを特徴とする請求項１０記載のテーブルを用いたデータ処理装置。
【請求項１８】
前記演算処理用データ格納メモリ検索データ作成手段が、前記入力されたデータ内で任意の位置にある複数のビットを抽出し、該入力データに対応するビット配置上で任意の連続した位置にまとめるマスクアンドギャザー処理を実行することを特徴とする請求項１７記載のテーブルを用いたデータ処理装置。
【請求項１９】
直接、あるいはバッファメモリを介して順次直列に接続された複数のデータ処理装置から構成されるデータ処理システムにおいて、
該複数のデータ処理装置のそれぞれが入力されるデータをメモリ検索用データに変換する入力変換手段と、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理の状態に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理の状態に対応するデータが格納されている状態遷移テーブルへアクセスするためのアクセス情報とを含むデータを表わす状態語が１つのエントリに格納されている状態遷移テーブル内の状態語を読み出すメモリ検索手段と、
該読み出された状態語の内容に基づいて、前記アクセス情報を得ると同時に、該読み出された状態語の内容に応じた演算を入力されたデータに対して実行する演算手段とを備え
ゝ
前記入力変換手段は、前記演算手段が演算を実行した出力データに基づいて、入力データをメモリ検索用データに変換する方法を決定することを特徴とするテーブルを用いたデータ処理システム。
【請求項２０】
前記データ処理システムにおいて、
前記複数のデータ処理装置の一部、または全部のデータ処理装置からの処理状態表示信号を受けて、複数のデータ処理装置間の処理を同期させるための信号を該一部、または全部のデータ処理装置に与える処理同期手段を更に備えることを特徴とする請求項１９記載のテーブルを用いたデータ処理システム。
【請求項２１】
入力データが直接、あるいはバッファメモリを介してそれぞれ与えられ、並列に接続された複数のデータ処理装置によって構成されるデータ処理装置システムにおいて、
該複数のデータ処理装置のそれぞれが、入力されるデータをメモリ検索用データに変換する入力変換手段と、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理の状態に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理の状態に対応するデータが格納されている状態遷移テーブルへアクセスするためのアクセス情報とを含むデータを表わす状態語が１つのエントリに格納されている状態遷移テーブル内の状態語を読み出すメモリ検索手段と、
該読み出された状態語の内容に基づいて、前記アクセス情報を得ると同時に、該読み出された状態語の内容に応じた演算を入力されたデータに対して実行する演算手段とを備え

、
前記入力変換手段は、前記演算手段が演算を実行した出力データに基づいて、入力データをメモリ検索用データに変換する方法を決定することを特徴とするテーブルを用いたデータ処理システム。
【請求項２２】
前記データ処理システムに対して複数のデータが多重化されて入力データとして与えられる時、該入力データをあらかじめ規定された方法で分離し、前記複数のデータ処理装置の中であらかじめ定められた２つ以上のデータ処理装置に入力させる多重化入力データ分離手段を更に備えることを特徴とする請求項２１記載のテーブルを用いたデータ処理システム。
【請求項２３】
前記データ処理システムに対して複数のデータがアドレスによって領域分割されたバスを介して多重化されて入力される時、入力されるデータが経由してきたバス内の領域のアドレスに対応して該多重化された入力データをあらかじめ規定された方法で分離し、前記複数のデータ処理装置の中であらかじめ定められた２つ以上のデータ処理装置に入力させる多重化入力データ分離手段を更に備えることを特徴とする請求項２１記載のテーブルを用いたデータ処理システム。
【請求項２４】
入力されるデータをメモリ検索用のデータに変換する入力変換手段と、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理に対応するデータが格納されているテーブルへアクセスするためのアクセス情報とを含むデータが１つのエントリに格納されているテーブル内のデータを読み出すメモリ検索手段と、
該読み出されたデータの内容に基づいて、前記アクセス情報を得ると同時に、該読み出されたデータの内容に応じた演算を入力されたデータに対して実行する演算手段とを備え、
前記入力変換手段は、前記演算手段が演算を実行した出力データに基づいて、入力データをメモリ検索用データに変換する方法を決定することを特徴とするテーブルを用いたデータ処理装置。
【請求項２５】
入力されるデータをメモリ検索用データに変換する入力変換手段と、
該メモリ検索用データを用いてメモリ内のテーブルに格納されているデータを読み出すメモリ検索手段と、
該読み出されたデータに基づいて、入力データに対する演算を実行する演算手段とを備え、
前記入力変換手段は、前記演算手段が演算を実行した出力データに基づいて、入力データをメモリ検索用データに変換する方法を決定することを特徴とするテーブルを用いたデータ処理装置。
【請求項２６】
入力されるデータをメモリ検索用データに変換し、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理の状態に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理の状態に対応するデータが格納されている状態遷移テーブルへアクセスするためのアクセス情報とを含むデータを表わす状態語が１つのエントリに格納されている状態遷移テーブル内の状態語を読み出してメモリ検索し、
該読み出された状態語の内容に基づいて、前記アクセス情報を得ると同時に、該読み出された状態語の内容に応じた演算を入力されたデータに対して実行し、
前記入力データを変換する際は、前記演算を実行した出力データに基づいて、入力データをメモリ検索用データに変換する方法を決定することを特徴とするテーブルを用いたデ

ータ処理方法。
【請求項２７】
入力されるデータをメモリ検索用のデータに変換し、
該メモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理に対応するデータであって、該処理の内容と次の入力データに対して行われるべき処理に対応するデータが格納されているテーブルへアクセスするためのアクセス情報とを含むデータが１つのエントリに格納されているテーブル内のデータを読み出してメモリ検索し、
該読み出されたデータの内容に基づいて、前記アクセス情報を得ると同時に、該読み出されたデータの内容に応じた演算を入力されたデータに対して実行し、
前記入力データを変換する際は、前記演算を実行した出力データに基づいて、入力データをメモリ検索用データに変換する方法を決定することを特徴とするテーブルを用いたデータ処理方法。
【請求項２８】
入力されるデータをメモリ検索用データに変換し、
該メモリ検索用データを用いてメモリ内のテーブルに格納されているデータを読み出してメモリ検索し、
該読み出されたデータに基づいて、入力データに対する演算を実行し、
前記入力データを変換する際は、前記演算を実行した出力データに基づいて、入力データをメモリ検索用データに変換する方法を決定することを特徴とするテーブルを用いたデータ処理方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明はプロセッサなどのデータ処理装置に係り、さらに詳しくは、例えばサンプリング周期毎に発生する時系列データなど一連のデータ（ストリームデータ）を高速に処理するために、命令列を読み込む必要を無くし、入力されるデータに対応してメモリを検索し、検索された内容に応じて入力データに対する処理を実行する、テーブルを用いたデータ処理装置に関する。
【０００２】
【従来の技術】
ネットワーク通信で用いられる通信パケット、ビデオ／音声データ、各種センサからサンプリング周期毎に発生する時系列データディスクをリードライトするデータ、データフロープロセッサの演算データ、並列計算機のプロセッサ間通信データなどの一連のデータ（ストリームデータ）を処理する場合は非常に多い。ここでいうストリームデータ処理とは概ね以下のような特性の一つあるいは複数をもつ。
Ａ）一定のあるいは間欠的な転送速度で一定の語長のデータが処理装置に入力される
Ｂ）ストリームデータは複数の種類のデータが多重化される場合がある
Ｃ）処理出力は新たなストリームデータになりうる
Ｄ）処理出力はメモリにバッファリングされることがありうる
Ｅ）入力ストリームデータおよび出力（データストリーム）は複数でありうる
Ｆ）処理シーケンスを有限状態機械で構成できる
Ｇ）処理機能の一つとしてテーブル検索があり、ストリームデータ語をキーとしてテーブル検索をおこなう必要がある
Ｈ）処理機能の一つとして特殊な演算があり、ストリームデータ語に対して演算する必要がある
ここで有限状態機械（Finite State Machine）は形式言語理論で定義された意味では最も能力の低いクラスのオートマトンの名称でもある。本発明では敢えて有限状態機械という言葉を使用するが、これは一般的な意味で有限の状態と状態遷移で定義される状態機械を意味する。
【０００３】

ストリームデータはネットワーク、バスなどの伝送路を介して計算機などの処理装置に転送され、処理される。
このようなストリームデータの速度はデバイス速度の向上に従って年々高速化しており、通信パケットを例にとると現在でも１Ｇbps(125 MB/sec）～４Ｇbps(500MB/sec)が要求されており、将来はさらに高速化する見込みである。例えば転送速度が１Ｇbps のストリームデータをバイト処理すると８ns(125 MHz）、４バイト単位で処理しても32ns(31.25 MHz) で処理しなければならない。高速のストリームデータを処理しようとするとこの性能が問題となる。また機能的にも画像処理、通信処理など複雑な処理が要求されるだけでなく、処理内容を柔軟に変更できることが求められている。

【０００４】

本発明はストリームデータ処理をはじめとする汎用データ処理を目的とし、特に処理内容を変更可能な処理装置（プロセッサ）の構成方式にかかわる。ストリームデータを処理するための従来技術はハードウェア方式とソフトウェア方式に大別される。ストリームデータの処理は論理的にはハードウェア方式でもソフトウェア方式でも実現可能であるが、考慮しなければならないのは処理性能と機能変更の容易さである。

【０００５】

ハードウェア方式は処理機能を専用のハードウェアで実現する方式で広く用いられている。専用ハードウェア処理ではストリームデータを入出力速度で処理できるよう専用ハードウェアを構成するので、ストリームデータをバッファリングすることなく一語入力する毎に逐次的に処理する方式（逐次処理方式）が可能である。バッファリングしないといってもある程度のエラスティックバッファを介することで転送レートと処理レートの調整をする場合はあるが、逐次処理方式では一連のストリームデータを全部メモリに格納してから処理する一括処理方式に比べて一般に処理遅延が小さくなる長所がある。

【０００６】

現在のＣＭＯＳデバイスの性能は250MHz程度であるから、処理する語長を適切に調整することで逐次処理方式を用いた低遅延、高性能を実現可能であるが、こんどは機能の変更可能性が問題となる。この問題を解決するために従来より用いられる方式としてはＦＰＧＡ（Field Programmable Gate Array), ＰＬＤ（Programmable Logic Device)などの再構成可能なデバイスを用いる方法がある。再構成可能なデバイスを用いる方式は一部のインターネットルータなどで用いられているが、現在のプログラマブルデバイスでは実現できる回路量が限られること、性能が押さえられることから機能を限定した分野のみで使用される。今後テクノロジの進歩によって大規模かつ高性能のデバイスが利用可能になるとしても、同じテクノロジを用いたストリームデータの転送速度も同様に向上すると考えられるので再構成可能なデバイスの利用分野は常に性能の低い分野に限定されると考えられる。

【０００７】

次にソフトウェア方式は汎用または専用のプロセッサおよびプロセッサが実行するソフトウェアを用いて処理を行う方式である。ソフトウェアによって機能を実現するため、機能変更が容易という利点がある。また、計算機システムにおいてはプロセッサが存在しているので、最小限のハードウェアで実現でき、低コスト化がはかれるという利点もある。

【０００８】

しかしながら性能的にはいくつかの問題がある。一般に一つのストリームデータを処理するためには複数の命令を実行しなければならないため、プロセッサはストリームデータの転送速度の複数倍の速度で動作しなければならない。仮に一つのストリームデータを処理するために１０命令を実行しなければならないとすると、１Ｇbps のストリームデータを４Ｂ処理するためには312.5MHz以上で動作するプロセッサを100 %動作させる必要がある。結局のところストリームデータの転送速度が低速であればソウトウェア方式は有効であるが、プロセッサの動作周波数に近い高速なストリームデータを処理するのは困難である。

【０００９】

また計算機は一般にオペレーティングシステムなどの管理システムのもとで動作するもの

であって、ストリームデータが発生したからといって直ちに処理を開始できない場合がある。そこで一連のストリームデータをメモリに格納し、まとまった量がたまってから一括処理し、結果データを得たり、再度別の場所に転送する。このような一括処理方式は通常計算機システムで採用される代表的な方式であって、ストリームデータをＩ／Ｏバスを介してメモリに貯え、一連のデータが格納されたところで計算機がソフトウェアによってデータを処理し、処理が終わると結果をＩ／Ｏバスを介して他に転送する。具体的には多くの計算機ネットワーク処理、画像処理、インターネットルータなどがこの方式を採用している。この方式ではデータをメモリに格納するため遅延を生じるのが問題である。このため処理能力的には十分であっても処理が間欠的になり、低遅延の逐次処理方式は採用できない。これはリアルタイム（実時間）性の問題として広く知られている。

【００１０】

以上をまとめると、従来技術において、ハードウェア方式は高速処理可能な反面、機能変更が困難であり、ソフトウェア方式は柔軟に機能変更できる反面、性能に限りがある。そこで、機能変更可能かつ逐次処理可能な処理方式が求められている。

【００１１】

従来のプロセッサは、図２７に示されるように、ノイマン型プロセッサと呼ばれるストアドプログラム方式であって、ハードウェアとしては基本的な要素となる演算装置とプログラムを実行するための機構からなる。プログラムはこれらのハードウェアを利用して処理機能を実現するもので、プログラムを変更することによって機能を変更できる点が特徴である。ストアドプログラム方式ではデータを処理するために、処理機能を実現するプログラムを構成する命令（インストラクション）を獲得（フェッチ）し、これを解読（デコード）し、実行（イグゼキュート）するというハードウェア操作が必要である。処理内容が複雑になると一個のデータを処理するために複数の命令を処理する必要がある。従って一般的にストアドプログラム方式のデータ処理能力はプロセッサの命令処理能力に比例し、かつデータ処理性能は命令処理性能より低い。いいかかえると命令処理性能より高速のデータ処理性能は得られない。

【００１２】

また、一個のデータを処理するのに複数の命令を実行しなければならないので、データ処理性能は命令処理性能の１／ｎになる。ここでｎはプロセッサのアーキテクチャや処理内容に依存する数値であるが、一般に単純なコード変換でも５～10，複雑な通信パケット処理では100 ～1000程度になる場合がある。すなわち、ある周波数のストリームデータを処理するためには、その周波数の５～1000倍の命令処理性能をもつプロセッサが必要である。

【００１３】

従来技術では命令処理性能を向上させる面とｎを小さくする面の両面から改善が行われている。キャッシュ、パイプラインなどは命令処理性能面、マルチメディア処理用命令セットであるＭＭＸ命令などはｎを小さくする面からの改善である。また並列処理は両面に寄与する改善策といえる。しかし、以上に説明したように、ストアドプログラム方式のプロセッサでは本質的に「命令処理性能＞データ処理性能」の制限を免れることはできない。通信ネットワークなどストリームデータを供給する側は専用ハードウェアで構成されるため、同じ半導体テクノロジを用いた時、「ストリームデータ性能＝命令処理性能＞データ処理性能」の関係が常に成り立ち、アドレス処理方式では永久にストリームデータを処理することは不可能ということになる。

【００１４】

ストアドプログラム方式のプロセッサは命令ストリームを高速に処理するように最適化された有限状態機械である。プロセッサが処理する命令の形式を変更する方式は従来から提案されており、そのような方式を備えたプロセッサアーキテクチャをダイナミックアーキテクチャとよぶ。ダイナミックアーキテクチャを実現する一般的な方式としては主にＣＩＳＣ（Complex Instruction Set Computer）で用いられるマイクロプログラム方式がある。

【００１５】
マイクロプログラム方式の詳細な説明は省略するが、概略は以下のようなものである。まず命令をデコードした結果、その命令に対応するマイクロ命令のアドレスを得る。マイクロ命令は制御記憶に格納されたプログラムで、これを実行することによってもともとの命令の機能を実現する。マイクロ命令には様々な実現方式があるが、一般的にプロセッサハードウェアの資源を制御するためのビット列からなっており、この命令を順次読み出し、ハードウェアに適用することによって目的の機能を得る。マイクロプログラムプロセッサの処理の基本的な手順は以下のようになる。
【００１６】
第１手順：命令を読み込む。
第２手順：命令に対応して定義される処理（マイクロ命令）を選択する。
第３手順：選択された処理を実行し、第１手順に戻る。
【００１７】
そこて、この制御記憶に格納されるマイクロ命令を変更することでプロセッサの命令を変更することができる。但し、従来のマイクロ命令の形式はプロセッサハードウェアが持つ資源に応じて固定的であり、変更はあくまでもプロセッサアーキテクチャの範囲内に限定される。このため任意のデータを処理できるほどの柔軟さはないし、仮に複数のマイクロ命令を用いて処理を実現できたとしても、これはプロセッサのプログラムレベルで処理するのと本質的に同じであるため処理性能は低下する。ＲＩＳＣ（Reduced Instruction Set Computer）はマイクロプログラム処理の性能制限を克服するために考案された方式であるが、このことはマイクロプログラム方式が場合によって性能的に問題があることを示している。
【００１８】
以上のようにマイクロプログラム方式にはアーキテクチャ的な制限の問題と性能の問題があり、マイクロプログラム方式のプロセッサを用いて一般的なストリームデータ処理を実現することは、従来提案されたことがないと考えられる。
【００１９】
【発明が解決しようとする課題】
以上説明したように、例えばストリームデータ処理をハードウェア方式で実現する場合には、処理速度の点については高い性能を実現できるが、機能変更が容易でないという問題点があった。
【００２０】
またソフトウェア方式を用いる場合には、データ処理性能が常に命令処理性能より低いという制限を免れることができず、ダイナミックアーキテクチャ実現のためのマイクロプログラム方式においても機能変更に柔軟さがないという問題点があった。
【００２１】
本発明の課題は、上述の問題点に鑑み、マイクロプログラム方式に類似したダイナミックアーキテクチャを用いて、プロセッサに命令ではなく、ストリームデータを直接処理させることによって、ストリームデータ処理を始めとする汎用データ処理を高速に実行することができ、また処理機能を容易に変更することができるデータ処理装置を提供することである。
【００２２】
【課題を解決するための手段】
図１は本発明の原理構成ブロック図である。同図は、例えば有限状態機械として、ストリームデータを処理するストリームプロセッサを実現するための、テーブルを用いたデータ処理装置１の構成ブロック図である。
【００２３】
図１において、入力変換手段２は入力されるデータをメモリ検索用データ、例えば入力データに含まれるメモリ検索用データを更にビット数の少ないメモリ検索用データに変換するものである。ただし発明の実施の形態によっては入力されるデータそのものをメモリ検

索用データとして用いることもできる。
【００２４】
メモリ検索手段３はそのメモリ検索用データを用いてメモリ４を検索し、入力データに対して行われるべき処理の状態に対応して、その処理の内容を含むデータとしての状態語が１つのエントリに格納されている状態遷移テーブル内の状態語を読み出すものである。
【００２５】
演算手段５は読み出された状態語の内容に応じて、次の入力データに対応して行われるべき処理の状態に対応する状態遷移テーブルへのアクセス情報を得ると同時に、その状態語の内容に応じた演算を入力されたデータに対して実行するものである。
【００２６】
本発明の実施の形態においては、データ処理装置の内部のメモリの状態遷移テーブルの内容をデータ処理前、あるいは処理中に変更する状態遷移テーブル書き換え手段を更に備えることもできる。
【００２７】
本発明の実施形態においては、前述の状態語は入力に関するデータを格納する領域、状態遷移先へのアクセス情報を格納する領域、演算の内容を格納する領域、結果の出力に関するデータを格納する領域、およびこれら４つの領域のいずれが状態語に含まれるかを示すタグ領域のうちで、少なくとも前述の４つの領域の中の１つ以上を含み、１つ以上の領域を備えるものである。この場合、前述の４つの領域のうちでいずれが備えられるかによって、状態語はそれぞれ一定の長さを持つこともできる。
【００２８】
また発明の実施の形態によっては、状態語はタグ領域を必ず備えるようにすることもできる。この場合タグ領域を状態語そのものに備えるのではなく、状態語の領域構成毎にそれぞれ対応するタグを格納する複数のタグレジスタが備えられ、タグがどのタグレジスタに格納されているかによって状態語の領域構成をメモリ検索手段３が認識することもできる。
【００２９】
発明の実施の形態においては、入力変換手段２は入力されるデータ内で任意の位置にある複数のビットを、入力データに対応するビット配置上で任意の連続した位置にまとめるマスクアンドキャザー処理を実行することもできる。
【００３０】
またメモリ検索手段３は入力変換手段２の出力とメモリのベースアドレスとを合成して状態語のアドレスを求めると共に、そのアドレスに格納されている状態語に含まれているメモリアドレスを次に行うべきメモリ検索におけるベースアドレスとして使用することもできる。
【００３１】
本発明の実施の形態においては、演算手段５は演算に必要なデータを格納するレジスタ手段と、レジスタ手段に格納されているデータと入力データとを用いて演算を実行する演算処理手段とを備えることもできる。
【００３２】
この場合演算処理手段が、入力されたデータの演算処理の間は演算結果の出力を遅延させるためのＦＩＦＯメモリ手段を更に備えることもできる。
実施の形態においては、演算手段は入力データに対応する演算結果を格納するテーブルを更に備えて、演算処理手段が入力データに対応してそのテーブルを検索することによって演算結果を出力することもできる。
【００３３】
また演算手段５は、入力されたデータを一時的に格納する入力ＦＩＦＯメモリ手段と、演算処理手段の出力を一時的に格納する出力ＦＩＦＯメモリ手段とを更に備えることもできる。
【００３４】

また実施形態においては、演算処理手段が入力データのデータ構造に対応して複数の領域を有するレジスタ手段を備え、入力データを分割してその複数の領域に格納し、格納結果のデータの全部、または一部に対して所定の処理を実行することもできる。

【００３５】

この場合所定の処理は、格納されたデータがあらかじめ定められたデータ構造のパターンに一致するか否かを判定する処理であることも、また１つ以上の領域に格納内容を抽出して、あらかじめ定められた形式のデータとしてまとめ、入力データの特徴を示す情報を求める処理であることもできる。

【００３６】

実施の形態においては、演算手段５は演算処理に必要なデータを格納するメモリ検索用のデータとして、入力データをビット数の少ないメモリ検索用データに変換する手段を備えることもできる。この場合、この手段は前述のマスクアンドキャザー処理を実行することもできる。

【００３７】

本発明の実施形態においては、図１で説明したデータ処理装置が直接、あるいはバッファメモリを介して順次直列に接続されたデータ処理システムを構成することもできる。

【００３８】

この場合一部、または全部のデータ処理装置からの処理状態表示信号を受けて、データ処理装置間の処理を同期させるための信号をデータ処理装置に与える処理同期手段を更に備えることもできる。

【００３９】

本発明の実施の形態においては、入力データが直接、あるいはバッファメモリを介してそれぞれ与えられ、並列に接続された、図１で説明した構成を有するデータ処理装置によって、データ処理システムを構成することもできる。

【００４０】

この場合、このシステムに対して複数のデータが多重化されて入力される時、その入力データを所定の方法で分離し、複数のデータ処理装置の中であらかじめ定められた２つ以上のデータ処理装置に入力させる手段を更に備えることもできる。

【００４１】

またこのシステムに対して、複数のデータがアドレスによって領域分割されたバスを介して多重化されて入力される時、バス内の領域のアドレスに対応して多重化入力データを所定の方法で分離し、あらかじめ定められた２つ以上のデータ処理装置に入力させる手段を備えることもできる。

【００４２】

本発明の実施の形態においては、図１で説明したデータ処理システムを単純化して、入力されるデータをメモリ検索用データに変換する入力変換手段と、そのメモリ検索用データを用いてメモリを検索し、入力データに対して行われるべき処理に対応して、その処理の内容を含むデータが１つのエントリに格納されているテーブル内のデータを読み出すメモリ検索手段と、読み出されたデータの内容に対応して、次の入力データに対して行われるべき処理に対応するテーブルへのアクセス情報を得ると同時に、読み出されたデータの内容に応じた演算を入力データに対して実行する演算手段とを備えるデータ処理装置を用いることもできる。

【００４３】

また実施の形態においては、データ処理装置の構成を更に単純化し、入力されるデータをメモリ検索用データに変換する入力変換手段と、メモリ検索用データを用いてメモリ内のテーブルに格納されているデータを読み出すメモリ検索手段と、読み出されたデータに対応して入力データに対する演算を実行する演算手段とを備えるデータ処理装置を用いることもできる。

【００４４】

以上説明した本発明のテーブルを用いたテーブル処理装置では、データ処理装置に命令で

はなく、例えばストリームデータを直接処理させることになる。ストリームデータ処理の基本的な手順は次のようになる。

【００４５】

第１手順：入力データを読み込む。

第２手順：入力データに対応して定義される処理を選択する。

第３手順：選択された処理を実行し、第１手順に戻る。

【００４６】

この手順はマイクロプログラム方式のプロセッサが命令を実行する場合と本質的に同様の処理である。本発明では従来のマイクロプログラム方式を一歩進めて、命令ストリームを読み込む必要をなくし、例えばストリームデータ処理を実現するダイナミックアーキテクチャを構築するものである。このようなデータ処理装置を実現するために入力データを解析し、処理を行う有限状態機械が構成される。処理機能を変更可能とするために、メモリに格納されている状態遷移表の内容を処理前、あるいは処理の途中で更新することにより、有限状態機械の構成を変更可能とすることができる。

【００４７】

【発明の実施の形態】

図２は本発明の有限状態機械、例えばストリームプロセッサの基本構成ブロック図である。一般に本発明における有限状態機械、例えばストリームプロセッサは入力、状態、状態遷移、および出力の４つの要素によって定義される。まず入力はそれぞれの状態において定義される入力データであり、具体的には入力されるストリームデータの全部、あるいは一部、さらに例えば演算に必要な有限状態機械の内部情報などである。

【００４８】

状態は有限状態機械の内部状態であり、処理の過程で状態の推移が行われる。ある状態において定義される入力のそれぞれに対して、次の状態遷移と出力とが定義される。

【００４９】

状態遷移は状態から状態への遷移規則を示すものであり、一般に有限状態機械内でメモリに状態遷移表（テーブル）の形式で格納される。

出力は状態に対応して定義される出力データであり、具体的には出力されるストリームデータの全部、あるいは一部、ハードウェアを制御するための制御語などである。

【００５０】

有限状態機械は、例えば前述のようなストリームデータを処理するストリームプロセッサであるが、より一般的には順序回路と同様の動作を行うものである。すなわちあるサイクルにおいて１つの状態に対応して順序回路の最初の段の動作が行われ、次のサイクルで状態遷移先の状態に対応して順序回路の次の段の動作が行われ、以下同様の動作が繰り返されるものと考えることができる。

【００５１】

図２は有限状態機械の最も基本的な実施形態を示すものであり、有限状態機械１０は入力変換機構１１、状態遷移表（テーブル）を格納するメモリ１２、検索機構１３、および演算・出力機構１４によって構成されている。入力変換機構１１は入力データに含まれる、例えばメモリ検索用のデータを、例えばビット数のより少ないメモリ検索値に変換し、検索機構１３、および必要に応じて演算・出力機構１４に与えるものである。

【００５２】

検索機構１３は、入力変換機構１１から与えられたメモリ検索値を用いて、メモリ１２に格納されている状態遷移表を検索する。後述するように、状態遷移表は複数のエントリから構成され、各エントリには入力データに対して行われるべき演算処理などの状態に対応する状態語が格納されている。この状態語には、入力データに対する処理の内容を規定する、例えば制御語に加えて、次の状態遷移表の先頭アドレス、すなわち状態遷移先のアドレスが格納されている。

【００５３】

演算・出力機構１４は、検索された状態語の内容によって入力データに対して必要な演算

などの処理を実行し、必要に応じて出力データを有限状態機械１０の外部に出力するものである。

【００５４】

本発明の実施形態においては、有限状態機械の構成として図２の構成をより一般化、単純化したものを使用することもできる。例えばメモリ１２に状態遷移表という有限状態機械特有の表を格納することなく、メモリ１２に格納された任意のデータを、各サイクルで入力されるデータをパラメータとしてアクセスし、メモリのアクセス結果や入力データを用いて演算を行って出力を得たり、次のメモリアクセス先を求めたりするデータ処理装置とすることもできる。

【００５５】

また更に有限状態機械を一般化、単純化したデータ処理装置として、単にメモリにアクセスし、メモリに格納された任意のデータを用いて順次入力されるデータに対する演算などを行って出力を得るデータ処理装置を実現することもできる。この場合には、各サイクルにおいて順次入力される入力データをパラメータとして、メモリのアクセス先が決定されるだけであり、次のメモリアクセス先の獲得、すなわち状態遷移に相当する動作は行われない。すなわちこの場合には、固定されたメモリデータを用いるだけで、入力データに対して必要な演算を実行できるような比較的単純な動作を実行するデータ処理装置が実現される。

【００５６】

以下の説明では、本発明の内容を図２に示した最も基本的な構成を用いて説明することにし、有限状態機械を一般化、単純化したデータ処理装置の動作については特に記述しない。

【００５７】

図２においてメモリ１２は、ランダムアクセス可能なメモリであれば、読み出し専用であっても、また書き換え可能なメモリであってもよい。メモリ１２として書き換え可能なメモリを用いて、処理の前、あるいは処理の途中に格納されている状態遷移表の内容を変更することによって、処理の内容を変更することが可能となる。

【００５８】

図３は入力ストリームデータにあらかじめ定められた処理を実行して処理結果を出力ストリームデータとして出力する有限状態機械としてのストリームプロセッサを示す。この有限状態機械１０はその構成を変更可能なものである。すなわち、その内部のメモリに格納されている状態遷移表（テーブル）の内容を処理の前、あるいは処理の途中で更新することにより、有限状態機械１０の構成を静的、または動的に変更することができる。

【００５９】

次に有限状態機械の実行サイクルについて図４を用いて説明する。図４で有限状態機械、例えばストリームプロセッサは基本的には単一のクロックに同期して動作する。１つのサイクルの動作は入力、状態遷移、処理、および出力の４段階からなる。まず１つのサイクルの始まりで入力データが確定し、図２の入力変換機構１１によってメモリ検索用の検索値が求められ、状態遷移表の検索、すなわち状態遷移が開始される。状態遷移は状態遷移表の検索、すなわちメモリアクセスによって状態語が確定し、その内容によって状態遷移先、すなわち次の状態が確定することを意味する。

【００６０】

次の状態が確定すると同時に入力データに対する処理が開始され、処理が完了して出力、すなわち演算結果が確定すると、必要に応じてその結果が出力されると共に、次のサイクルに対応する次のデータの入力動作が開始され、次のサイクル開始時点では入力データが確定し、以下同様の動作が繰り返される。

【００６１】

図４において有限状態機械の処理時間は入力に必要な時間、状態遷移表を検索するメモリアクセスタイム、演算・出力機構の処理時間の合計となる。これらの時間の中で最も長いものはメモリアクセスタイムであり、現在のテクノロジではＬＳＩ内蔵メモリで４～５ns

程度であり、単純な演算しか行わない場合のサイクルタイムは10ns以下になると考えられる。すなわち100MHz程度の動作が可能と期待される。

【００６２】

図４で説明したように基本的には１つのサイクルで入力データに対応する１つの処理が実行されることを基本とするが、例えば演算処理に時間が必要な場合には、その処理の完了を待って同期をとる必要がある。図５はこのような処理の同期を説明する図である。

【００６３】

例えば有限状態機械が入力動作可能になっても、入力データがない場合にはそのまま待機する必要があり、また入力データに対する処理の時間が長い場合にも同期をとる必要がある。図５は１つのサイクルにおいて開始された処理が次のサイクルの開始時点までに完了しなかった場合の説明図である。処理が完了し、出力が確定するのは次のサイクルの途中であり、それまで次の入力データの入力動作の開始は遅らされる。

【００６４】

このような制御は、状態遷移表の中の状態語に格納されている制御語の内容によって、例えば図２の演算・出力機構１４が起動された後に演算・出力機構１４から出力される処理完了の有無を示す信号を検査することによって実現できる。但し後述するように演算の種類によっては、演算・出力機構による処理が実行されている状態で状態遷移を完了させ、次の状態に対応する処理を開始することもできる。

【００６５】

ストリームデータは入力、出力共に外部のクロック、例えばネットワークのクロックに同期して転送されることが多い。この場合、有限状態機械はこのような入力・出力クロックの２倍以上の平均周波数で動作する必要がある。これはシャノンの標本化定理による。また演算の内容によっては図５で説明したように処理に時間がかかる場合があり、この場合有限状態機械は処理の完了を待つ必要があり、その間に新しいストリームデータが到着しても処理を行うことができない。

【００６６】

図６はこのような処理速度の違いを吸収するための入出力同期の説明図である。同図に示すように、有限状態機械１０の入力側、および出力側にＦＩＦＯメモリを設けることによって、処理速度の違いを吸収することができる。有限状態機械１０は入力ＦＩＦＯメモリ１６が空の時には待機状態となり、新たなデータが到着するまで動作を停止する。また出力ＦＩＦＯメモリ１７が一杯になった場合には、そのメモリに空きができるまで有限状態機械１０、例えばストリームプロセッサは処理を停止する。

【００６７】

続いて図２の有限状態機械の各構成要素の動作について更に詳細に説明する。図７は入力変換機構１１の構成例の説明図である。同図において入力変換機構１１は入力データ、または演算・出力機構１４から与えられるデータを選択するためのセレクタ２１、変換のために必要なパラメータを格納する複数のパラメータメモリ２２ａ，２２ｂ，・・・、およびセレクタ２１の出力とパラメータメモリの格納内容を用いて入力データの変換を実行する変換回路２３から構成されている。

【００６８】

入力変換機構１１は、メモリ１２に格納されている状態遷移表内の状態語の検索処理を効率化するために、入力データに含まれるメモリ検索用のデータを、例えばビット数の少ないメモリ検索値に変換する機構である。例えばストリームプロセッサにおいて、一般にストリームデータの形式は多種多様であり、例えば４バイト入力のストリームデータによって状態遷移表を作ると、その状態遷移表には2^{32}（４Ｇ）のエントリが必要となる。このような巨大な状態遷移表をチップ内の高速アクセス可能なメモリにおくのは困難であり、また実際には全ての入力パターンが有効でないことが多いため、入力されたストリームデータからその状態に応じて検索に必要なデータ、すなわち検索値への変換を行うことによって、状態遷移表の検索処理が高速化される。

【００６９】

入力変換機構１１は、入力データと共に演算・出力機構１４の出力を用いて、メモリの検索値を生成する。演算・出力機構によって行われる演算としては、入力データに対応してエンディアンやビット配置を変換する演算，ハッシュ方式によってデータを検索するためのハッシュ値を計算するハッシュ演算などがある。また演算に相当してテーブル、例えばルックアップテーブル（ＬＵＴ）を用いることもできる。このようなテーブルは本質的にメモリ１２に格納されている状態遷移表と同様のものであるが、状態遷移表とは独立にテーブルを設けることによって処理がパイプライン化され、スループットを向上させる効果がある。

【００７０】

図８は入力変換機構によるビット配置変換演算としてのマスクアンドギャザー（ＭＡＧ）処理の説明図である。この処理においては、入力データから任意のビットや領域が抽出され、あらかじめ定められた方法によってメモリの検索値が作成される。すなわち入力データに対してＭＡＧパターンが指定され、ＭＡＧ処理を行うことによってＬＳＢ側に集約されたメモリ検索値が処理結果として得られている。処理結果はＭＳＢ側に集約してもよく、どちらに集約するかを指定可能としてもよい。更に集約する位置を１ワードの端ではなく、途中にすることも可能である。このように処理結果をあるビット数にまとめることによって、状態遷移表としてのテーブルあたりのエントリの数が一定でない場合にも検索を高速化することのできる検索値を作成することができる。

【００７１】

次に状態遷移表（テーブル）について説明する。図９は状態遷移表の基本構造の例である。図２のメモリ１２の内部には一般に複数個の状態に対応する複数の状態遷移表が格納される。状態遷移表はある状態を定義するテーブルであり、その状態で定義される入力に対応する個数のテーブルエントリ（状態語）から構成され、この状態語によって特定の入力データに対応する状態遷移と、その入力データに対する処理の内容が定義される。　１つの状態語、すなわちエントリは１個、または複数の領域からなる。領域としては、タグ、入力、状態遷移、制御語、出力の５つが考えられる。タグはその状態語の種類や構成を示す識別子であり、入力はその状態語を選択するための検索データパターンである。この検索データパターンとは入力変換機構１１によって変換されたメモリ検索値でもよく、また入力データのうちのメモリアクセスのためのパターンそのものであってもよい。このようなパターンを１つの領域として状態語に格納しておく理由は、一般的に比較によって入力データを検索する場合があるためである。

【００７２】

状態遷移は状態遷移先、すなわち次の状態遷移表の先頭アドレスであり、制御語は入力データに対して行われるべき演算の内容などを示すものである。また出力はハードウェアに対応する制御語、すなわち従来のマイクロ命令と同様のもの、あるいは出力データそのものを示すものなどである。

【００７３】

状態語はこの５つの領域を全て備えるとは限らない。例えば１個の状態遷移表だけを備えた有限状態機械においては、状態遷移先は不必要であり、状態語に状態遷移の領域を定義する必要はない。また入力の検査が不必要な場合には入力の領域も不要であり、同様に制御語、出力の領域も不必要な場合があり、このような場合にはメモリの節約のためにこのような領域が省略される。ただしタグは状態語を解読するために必ず必要な領域である。

【００７４】

状態語の領域のうちでタグは状態語の種類や構成を示す識別子であり、タグの内容に対応して状態語の長さが設定されることになる。すなわちタグの内容、例えば状態遷移の領域が含まれるか否かによって、状態語の長さは可変となる。そこで状態遷移表を格納するメモリ１２に対するアクセスは様々な語長で行えることが必要となり、メモリ１２は様々なバスサイズでアクセスできるものでなければならない。例えばメモリ空間毎にあらかじめサイズを定義しておくことにより、状態遷移表のアドレスによって、メモリ１２に対するアクセスサイズを切り替えることが可能となる。

【００７５】
前述のように状態語の中でタグ領域は必ず必要なものではあるが、状態語の長さに応じて状態語の領域構成をあらかじめ決めておき、例えばメモリ空間毎に格納されている状態語の長さを変えることによって、アクセスされたメモリ空間によって状態語の領域構成が判明するようにしておくことにすると状態語にタグ領域を含める必要がなくなるため、状態語の情報量を削減することができる。
【００７６】
更にバイトアクセス、例えば１バイトのアドレスによってアクセスされた状態語は必ずバイトデータを出力するものであるという定義を作っておけば、エントリあたり１バイトの状態遷移表を用いてコード変換処理を実現することができる。この場合、状態語にタグ領域を含めるとすると、タグ領域と出力するバイトデータとが状態語の内容として必要となるため、エントリあたり１バイトではすまなくなる。
【００７７】
このようにタグ領域自体を不必要とすることもできるが、またタグ領域を状態語に含めるのではなく、タグを格納するタグレジスタを図２のメモリ１２と別に設け、そこにタグ領域の内容を格納することによって状態語の領域構成を知ることも可能である。このようなタグレジスタを状態語の長さ毎に設けておき、その内容を可変にすることによって状態語の領域構成を変更可能とすることもできる。状態語の長さが異なることによって、状態語に含まれる領域構成が異なってくるため、タグの内容も異なり、異なる長さの状態語に対応して複数のタグレジスタが必要となる。
【００７８】
図２において状態遷移表は有限状態機械１０の内部のメモリ１２に格納されている。有限状態機械の性能は状態遷移表のアクセスタイムに依存するため、このアクセスは高速に実行されることが望ましい。しかしながら複雑な有限状態機械では大きな状態遷移表が必要となり、状態遷移表を一般に容量の大きい外部メモリに格納し、一部分を有限状態機械内部にキャッシュすることによって動作の高速化が図られる。
【００７９】
状態遷移表へのアクセスパターンは有限状態機械によって行われる動作に依存するが、ある状態から始まる状態遷移の可能性は過去の動作経験によってあらかじめある程度予測可能であり、この予測に基づいて必要な状態遷移表の内容を事前にキャッシングすることも可能である。更にこの事前キャッシングの動作を制御語の内容として組み込むことによって、意図的なスケジューリングも可能となる。
【００８０】
次に検索機構の動作について説明する。図２の検索機構１３は、入力変換機構１１から与えられたメモリ検索値を用いてメモリ１２の内部の状態遷移表を検索するものであり、この検索のための方法としては最も単純な方法として、状態遷移表の先頭アドレスに、例えば演算・出力機構１４による演算結果、すなわち前述のハッシュ値の計算結果などを加算することによって入力データに対応するエントリ、すなわち状態語のアドレスを求めることができる。また例えば状態遷移表の先頭アドレスを２の冪乗となるようにしておき、これをベースアドレス、検索値をオフセットアドレスとしてベースアドレスとオフセットアドレスとの論理和をとる方法もある。
【００８１】
図１０は検索機構の動作説明図である。同図において現状態は現在の状態に対応する状態遷移表の先頭アドレスを指しており、入力データ、または演算・出力機構１４の出力する演算結果が与えられることによって、例えば状態遷移表の先頭アドレスとの単純な加算（入力合成）によって入力データに対応する状態語のエントリを指すアドレスが求められ、その状態語の内部の状態遷移が次の状態に対応する状態遷移表の先頭アドレス、すなわち次状態を示す値として出力されると共に、検索された状態語の内部に格納されている制御語などが出力される。次の状態遷移表の先頭アドレス、次の入力データに対する現状態の値として用いられる。

【００８２】
一般にメモリ検索の手法は多種多様である。高速性を図るために連想メモリ（コンテントアドレッサブルメモリ、ＣＡＭ）などの手法や、検索のためのハードウェアを用いる方法もあるが、本発明においてはどのようなメモリ検索方法を用いることもできる。
【００８３】
図１１は状態遷移表の検索例の説明図である。同図において最初の入力データに対応する状態遷移表は２４であり、また入力データ中に含まれるメモリ検索用のデータＸが８ビットであるとすると、状態遷移表２４は８ビットに対応する256 個のエントリを持つことになる。そしてその８ビットの内容が例えばそのまま状態遷移表２４の先頭アドレスに加算され、１つのエントリ、すなわち状態語２５が検索される。この状態語には図９で説明したように状態遷移、すなわち次の状態遷移表の先頭アドレスが格納されており、このアドレスによって第２の状態遷移表２６が確定され、また制御語の内容に対応して入力データに対する演算などの処理が実行される。
【００８４】
次の入力データが与えられるとメモリ検索用のデータ、例えばＹに応じて前述と同様にして１つのエントリ、すなわち状態語２７が検索されて、入力データに対して同様に処理が実行される。
【００８５】
なお図１１では入力データのうちのメモリ検索用のデータ、例えば８ビットがそのまま状態遷移表の中の１つの状態語の検索に用いられるものとしたが、前述のように入力変換機構１１によって例えば４ビットへの変換を行うことによって、より少ないエントリ、例えば16個のエントリを持つ状態遷移表の検索が行われる。
【００８６】
ここで本実施形態におけるデータ処理と状態遷移について具体例を用いて説明する。例えば３つのデータＡ，Ｂ、およびＣを用いて（Ａ・Ｂ）＋Ｃの計算を行うものとし、第１～第３のサイクルにおいて入力データとしてＡ，Ｂ、およびＣがこの順序で与えられるものとする。
【００８７】
第１のサイクルでは、入力データ内のメモリアクセス用のデータから入力変換機構１１によって、例えばビット数の少ないメモリ検索値が求められ、この検索値を用いて第１の状態遷移表から状態語が読み出される。この状態語には入力データとしてのＡを、例えばデータ処理装置内のレジスタに格納する動作が規定されており、入力データＡは第１のサイクルにおいてレジスタに格納されると共に、次のサイクル、すなわち第２サイクルで入力されるデータＢとの積を求めるための第２の状態遷移表の先頭アドレスが得られる。
【００８８】
第２のサイクルでは、一般的には入力データのうちメモリアクセス用のデータが同様に検索値に変換され、第２の状態遷移表の先頭アドレスと、例えば加算されて状態語が読み出されることによって、レジスタに格納されているデータＡと入力データＢとの積が求められ、この結果がレジスタに再び格納される。但しこのような具体例では各サイクルで行う処理が確定されており、実行順序も固定であるため、第２の状態遷移表の先頭アドレスがそのまま読み出すべき状態語のアドレスとして用いられる。
【００８９】
第２のサイクルでは、同時に次の第３サイクルで入力されるデータＣとの加算を行うための状態語が格納されている第３の状態遷移表の先頭アドレスが求められる。そして第３のサイクルで第１、第２サイクルと同様にして第３の状態遷移表の中の状態語が読み出され、レジスタに格納されている値Ａ・Ｂと入力データＣとの加算が行われることになる。
【００９０】
図２の各構成要素の説明に戻り、続いて演算・出力機構１４の動作について説明する。演算・出力機構１４は、状態遷移表に格納されている状態語の中の制御語の領域の格納内容に従って、入力データに対する演算を行ったり、演算結果を出力したりするものである。

演算・出力機構１４を、例えばストリームプロセッサに備える理由は、純粋な有限状態機械だけでは実現できないような処理を実現したり、性能を高速化するためである。例えば浮動小数点演算などはその最もよい例であるが、ストリーム処理を考慮すると、他にも様々な演算・出力処理が考えられる。

【００９１】

演算・出力機構１４としては、同時に複数の演算回路を動作させることもできる。例えば入力データの数をカウントしながら、エラー検出処理としてのサイクリックリダンダンシーチェック（ＣＲＣ）計算を実行し、同時に出力データを外部に転送するような並列的な処理を実行することによって、ソフトウェアで逐次手続き的な処理を行うよりも、処理の高速化を実現することができる。

【００９２】

図１２は演算・出力機構の構成例のブロック図である。同図において演算・出力機構１４は、複数の演算機構３０ａ，３０ｂ，．．．と、これらの演算機構からの出力を選択して、有限状態機械１０の出力とするための出力セレクタ３１とから構成される。各演算機構３０ａ，３０ｂ，．．．、および出力セレクタ３１の制御は、状態語が格納されている状態語レジスタ３２から出力される制御データ（結合されていない矢印）によって行われる。各演算機構への入力としては、入力データ、状態語レジスタ３２から出力される状態語の内容としての例えば制御語、および各演算機構の内部に含まれる演算回路の出力などがある。このような構成、および各部の接続は必要に応じて変更されるものであり、図１２の構成に限定されるものでないことは当然である。

【００９３】

図１２において各演算機構３０ａ，３０ｂ，．．．は、それぞれ入力セレクタ３４、レジスタ３５、および演算回路３６から構成されている。レジスタ３５は演算に必要とされる複数の演算データを格納するメモリであり、必要に応じて入力データ、出力データなどを格納する。レジスタとしては単純なメモリの他に、ＦＩＦＯメモリ、カウンタ、アキュミュレータ、コンパレータ、シフタ、エンデアン変換、ビット配列変換、コード変換などの機能を持つものを用いる場合がある。しかしながら、これらの機能をレジスタによって実現するか、演算回路として実現するかは実施上の問題である。本発明として特に特定するものではない。

【００９４】

演算回路３６はあらかじめ決められた演算を行う回路であり、演算の種類は想定されるアプリケーションに依存する。例えば数値計算では整数演算、浮動小数点演算、などの数値演算である。信号処理では積和計算や、フーリエ変換などであり、通信処理ではＣＲＣ計算、ＩＰチェックサム計算などである。ここでＩＰチェックサム計算とは、インターネットプロトコルヘッダのチェックサムの計算である。チェックサム計算では、ヘッダが１６ビットワードの並びとみなされ、それぞれの和が１の補数によって計算され、その結果の１の補数をもってチェックサムとされる。

【００９５】

また演算の種類としての暗号処理ではＤＥＳ演算、ＭＤ５演算などがある。ここでＭＤ５演算は、インターネットにおけるデータにセキュリティのためにデータの証明をつけるものであり、ハッシュデータを利用して送信データに対応して特殊な数式の計算によって求められるコードをデータと一緒に送信し、受信側で同じ手法で取り出されたハッシュデータと比較されることによって、データのセキュリティを確保する方式である。本発明では、演算回路によって行われる演算は特に特定されず、どのようなタイプの演算も実現できるような一般的な構成が用いられる。

【００９６】

図１３はそのような一般的な演算回路の構成例の説明図である。同図において、演算回路３６は一般に複数のレジスタ３８ａ，３８ｂ，．．．、および論理回路３９によって構成されている。演算回路３６への制御入力は、図１２における状態語レジスタ３２から直接に与えられ、出力はレジスタ３５に格納される。また演算結果出力は、例えば図２で説明

した入力変換機構１１に与えられるデータであり、入力データからメモリ検索値への変換に必要な、例えばハッシュ値である。

【００９７】

演算回路の制御は、マイクロプログラム方式と同様に、制御語の内容によって行われる。従って複数の演算を同時に制御できるような制御語を用いる場合には、１個の入力データに対して複数の処理を同時に実行することができる。図１４はそのような複数の処理を同時に実行する演算回路としてのカウンタ回路の例である。

【００９８】

同図において演算回路は複数のカウンタ４０ａ，４０ｂ，．．．、およびセレクタ４１から構成されている。各カウンタは制御語の内容に対応して、入力データによってプリセットされたり、カウント動作を行ったり、任意のカウント値をセレクタ４１を介して出力したりすることができる。制御語の内容を適切に構成することによって、複数のカウンタの同時制御が実行される。

【００９９】

ＦＩＦＯメモリも演算回路の一種として使用することができる。例えばインターネットプロトコルバージョン４，ＩＰｖ４のパケットをＩＰｖ６のパケットの中に埋め込んで通信を行うためのパケットカプセリングや、エラーコードチェックなどの処理ではパケット、例えばストリームデータを全部チェックしないと結果を出力できない場合があり、このような場合に入力ストリームデータを外部に出力するまでの間、１パケット分のデータを保存しておく必要がある。そこでＦＩＦＯメモリがバッファメモリとして用意され、出力可能となるまで１パケット分のデータが一時的に保存される。図１５はそのようなバッファメモリの構成を示し、複数のＦＩＦＯメモリ４２ａ，４２ｂ，．．．、およびセレクタ４３によって構成されている。

【０１００】

演算回路、例えば論理回路をテーブル、例えばルックアップテーブル（ＬＵＴ）によって実現する方法は広く知られている。このようなテーブルは、本発明における状態遷移表と同様の演算テーブルとして実現することが可能である。この場合、状態語はタグと出力とからなるものと考えられ、また入力データは演算対象データである。

【０１０１】

このような場合、演算回路専用のハードウェアで実現するか、テーブルを用いて実現するかは発明の実施方法や目的に依存する。目的によっては、例えば通信処理に用いられるルーティングテーブルのように非常に大きなテーブルを必要とする場合がある。このような場合には、演算回路として大きなテーブルを検索する機構を設けることが必要となる。

【０１０２】

図１６はそのようなテーブル検索機構の構成例の説明図である。このテーブルの検索方法は、図１０で説明した状態遷移表の検索方法とほぼ同様であり、外部メモリ上の１つのテーブルの先頭アドレスを指す現アドレスと、入力データに対応するメモリ検索値との、例えば加算によって、テーブル内の１つのエントリが検索され、また必要に応じて次のテーブルの先頭アドレス（次アドレス）を求めることもできる。

【０１０３】

前述のように、有限状態機械においては、ある状態に対応して開始された演算が終了するまでは次の状態への状態遷移は行われないのが原則であり、演算回路の処理時間は有限状態機械の性能に大きな影響を与える。しかしながら処理内容によっては、演算回路の処理結果を直ちに必要としない場合があり、このような場合は演算が完了しなくても、次の状態に遷移する方が効率的である。

【０１０４】

例えば暗号処理において、入力ストリームデータに対してＤＥＳ処理を行って結果を出力するような場合においては、有限状態機械の内部でのその結果を利用するわけではないため、連続するサイクルにおいてＤＥＳ演算の対象となる入力データを次々と与えて演算を実行すればよい。但しこの場合は、ＤＥＳ演算は複数の連続するデータに対して行われる

ことになり、ＤＥＳ演算の次の状態は同じＤＥＳ演算である。
【０１０５】
図１７は、このようにＤＥＳ演算を行うために、データ入力部にＦＩＦＯメモリを設けたＤＥＳ演算回路の構成ブロック図である。同図においてＤＥＳ演算回路は、入力データを書き込むためのＦＩＦＯメモリ４５、ＤＥＳキーを格納するレジスタ４６、ＤＥＳモードを格納するレジスタ４７、ＤＥＳコア演算部４８、および出力側のＦＩＦＯメモリ４９から構成されている。入力側のＦＩＦＯメモリ４５は、このメモリに空きがある限り入力データを待ち時間なく書き込むためのものであり、また出力側のＦＩＦＯメモリ４９は、このメモリにデータが格納されたことを検出して、有限状態機械がデータの出力を行うためのものである。
このような演算回路の構成はＤＥＳに限らず、処理スループットが一定ではないが、平均的には入力ストリームデータの速度と同程度か、あるいは速い場合に、有限状態機械のスループットを低下させない効果がある。
【０１０６】
また演算回路の例として、パターンメモリを用いたパターン処理回路がある。例えば通信パケットのヘッダ部分はいくつかの領域で構成されており、それらの値は相互に関連性を持っている。図１８はＩＰｖ４パケットのヘッダ部分を示す。同図においてディスティネーションアドレスまでの部分がヘッダ部分である。本発明の有限状態機械では、このように例えばインターネットプロトコルのように広く用いられ、形式が確定しているようなデータのデータ構造の解析を、パターン処理回路によって実行することができる。
【０１０７】
図１９はそのようなパターン処理回路の構成例のブロック図である。同図においてパターン処理回路は、複数のレジスタ５０ａ，５０ｂ，．．．、およびパターン処理部５１によって構成されている。パターン処理回路は確定したデータ構造のデータに対する処理を行うものであり、入力データを例えば分割して複数のレジスタに逐次格納した後に、入力データが想定されているデータのデータ構造に一致するかを検査したり、データ構造の中の特定の領域の内容を変更したりすることができる。パターン処理部５１の構成は目的に依存するが、基本的には論理回路によって実現できる。
【０１０８】
パターン処理回路の変形例としてプロファイル作成回路、すなわちプロファイラがある。プロファイラは一連のストリームデータの特徴を表わす情報としてのプロファイルを作成するものである。例えば図１８で示したように、ＩＰｖ４パケットのヘッダは複数の語から構成されるデータ構造を持っており、全ての領域を調べることによってＩＰｖ４パケットであることが認識できる。そして一度データ構造をチェックした後では、あらかじめ定義されたＩＰｖ４を示す、例えば１バイトのコードをプロファイルとして付与することにより、その後はそのコードによってデータ構造の識別が可能となる。
【０１０９】
このようなプロファイル処理は、一般的な有限状態機械としての順序回路を用いて実現することも可能であるが、例えばＩＰｖ４パケットのように頻繁に使われるデータ構造についてはあらかじめ専用の演算回路を用意しておくことによって、高速な処理が可能となる。
【０１１０】
演算・出力機構の内部には、例えば図８で説明したようなＭＡＧ処理部を備えることも可能である。入力変換機構の内部のＭＡＧ処理部は、状態遷移表が格納されたメモリを検索するためのメモリ検索値を求めるために使用されるが、演算・出力機構では図１６で説明したように、例えば論理回路の代わりにＬＵＴが用いられることがあり、このようなテーブルに対するアクセスを行うために、演算処理の段階でビット数の少ない検索値を使用することによって、テーブルへのアクセスを高速化することができ、このような場合にＭＡＧ処理は有効である。
【０１１１】

次に複雑な処理を実現するための有限状態機械の直列、または並列処理について説明する。例えばパケット処理は一般に複雑な処理である。例えばＩＰチェックサム計算は、ＩＰヘッダの内容を確定した後に計算する必要があり、例えば前述のパケットカプセリングにおいては、ＩＰヘッダの内容を変更しながら、同時にＩＰチェックサムの計算を行う必要がある。このような場合には、作業用レジスタを中間的に使用して、複数のサイクルを用いて必要な処理を実現することもできるが、このような処理では遅延の増加、スループットの低下を招くことになる。

【０１１２】

このような複雑な処理を効率的に行うための１つの解決策が、有限状態機械を直列に接続して、パイプライン構成を用いることである。図２０はそのようなパイプライン構成の説明図である。同図においては複数の有限状態機械５３ａ，５３ｂ，．．．が、それぞれＦＩＦＯメモリ５４ａ，５４ｂ，．．．を介して直列に接続されている。前述のパケット処理では、第１段目の有限状態機械５３ａによってＩＰヘッダの変換が行われ、第２段目の機械５３ｂによってＩＰチェックサム計算が行われる。各有限状態機械は全く独立に動作するため、機械的に有限状態機械を直列に接続することもできる。ハードウェアの容量が許すならば、このように有限状態機械のパイプライン構成をとることが望ましい。

【０１１３】

次に有限状態機械を複数個並列に動作させることによって、データ処理能力を向上させることができる。図２１はこのような並列構成の例を示す。この例では、複数個の有限状態機械５３ａ，５３ｂ，．．．が全て同じ処理速度で同期して動作するものとし、そのため並列に接続された有限状態機械の入力側にＦＩＦＯメモリ５５ａ、出力側にＦＩＦＯメモリ５５ｂがそれぞれ１個だけ備えられる。

【０１１４】

また複数の並列接続有限状態機械のうちで、データ処理速度が異なるものが一部存在するような場合には、処理の同期をとるために相互に信号をやり取りする必要がある。また有限状態機械の間での動作の同期を実現するために、一部の有限状態機械の入力側、または出力側にＦＩＦＯメモリを備えて、例えば処理時間の異なる演算機構の間での動作の同期をとるのと同様の動作を行わせる必要がある。

【０１１５】

更に並列に動作する有限状態機械の処理速度が、個々の有限状態機械毎にまちまちの場合には、個々の有限状態機械の入力側、および出力側にそれぞれＦＩＦＯメモリを設ける必要がある。このような並列構成を図２２に示す。同図においては入力データの読み込みと、データ出力とを処理の同期をとるために制御する必要があるが、この制御も前述と同様に動作時間の異なる演算機構の間での演算処理を同期させるのと同じ方法を用いて実現できる。

【０１１６】

前述のように直列接続、または並列接続された複数の有限状態機械を備えるシステムでは、有限状態機械の間で動作の同期をとる必要がある。図２３はそのための同期回路を持つ有限状態機械の並列構成の例を示す。同図においてシステム構成自体は図２１と同様であるが、複数の有限状態機械５３ａ，５３ｂ，．．．の間で処理の同期をとるために、各有限状態機械の処理状態が同期回路に集められ、その出力を各有限状態機械への入力として与えることによって、各有限状態機械が他の有限状態機械と同期して処理を行うことができる。

【０１１７】

次に多重ストリームデータの処理について説明する。例えば計算機バスのように、複数のデータの転送を共有バスを用いて行うような場合には、複数の互いに独立なストリームデータが多重化されて転送されることが多い。そこで本発明のストリームプロセッサにこのようなデータを処理させる場合には、多重化されたストリームデータを独立なデータに分類する機能が必要である。そのような分別機能を用いることによって、複数の互いに独立なストリームデータが多重化されて転送される場合に、そのストリームデータを分類して

互いに独立なストリームデータに分割した後に、例えば図２２で説明した並列に接続された各有限状態機械にそれぞれ入力させることによって、多重化されたストリームデータの処理を実行することができる。

【０１１８】

また例えば計算機バスのアドレス空間があらかじめ複数の領域に分割され、複数の互いに独立なストリームデータがその複数の領域を用いて転送されるような、アドレス付きで計算機バス内のアドレス空間を多重化されてストリームデータが転送される場合には、転送されるデータのアドレスがその複数の領域のどの領域に属するかによって、互いに独立なストリームデータの分類を行うことができる。

【０１１９】

最後に本発明の有限状態機械、例えばストリームプロセッサの具体的な応用例について説明する。第１の応用例としてのアスキー（ＡＣＳＩＩ）コード変換処理を図２４に示す。この例では入力ストリームのバイトデータがアスキーコードとして扱われ、大文字が小文字に変換されるものとする。この場合ストリームプロセッサは大文字を小文字に変換するアスキーコード変換表６０を備えており、入力される１文字毎にその変換表が検索されて、結果が出力される。アスキーコードは７ビットであり、変換表６０の大きさは128 エントリである。

【０１２０】

図２５はこの例における状態遷移のステートマシンを示す。アスキーコード変換ステートマシンは状態を１つだけ持ち、入力と出力がそれぞれアスキーコードであるような有限状態機械である。アスキーコード変換表は１つの状態を表し、変換表の各々のエントリが入力文字に対応するステートマシンの動作を示す。このようなステートマシンは、単純なＬＵＴを用いて、処理内容を変更可能なものとして構成可能である。ストリームプロセッサにおいては、アスキーコード変換表を１個の状態遷移表として備えることになる。この状態遷移表、すなわちＬＵＴは128 エントリであり、出力は７ビットのアスキーコードである。状態が１個なので、状態遷移は行われない。ストリームプロセッサの処理性能は基本的にＬＵＴのアクセス性能に依存する。ＬＵＴは小規模、かつ高速なメモリによって構成することができる。

【０１２１】

次に第２の応用例として、パケットのラベリングについて説明する。パケットラベリングは、コード変換より若干複雑な処理であり、入力されるパケットの内容を解析し、その結果に基づいてラベルの内容を決定し、そのラベルをパケットの先頭に付け加えて出力することであり、タギングということもある。当然その逆変換処理として、ラベルやヘッダの一部、または全部を削除する処理もあるが、ここではこれらをまとめてパケットラベリング処理と呼ぶ。パケットラベリングは、ほとんど全てのパケット通信処理において必要不可欠な機能である。

【０１２２】

図２６は、パケットラベリング処理の説明図である。この処理では、ヘッダの内容を解析する時間と、新しいデータとしてのラベルを生成し、入力パケットの先頭に挿入するための時間を作る必要がある。ここでヘッダの内容を解析するには、一般に入力パケットの転送レートよりも長い時間が必要である。このため図２６の解析／ラベル生成処理部７０においては、テーブル検索機構などを用いて、解析とラベルの生成を行い、その間は入力パケットはＦＩＦＯメモリ７１に格納され、ラベル生成が終了するとラベルが出力され、その後に引き続いてＦＩＦＯメモリ７１に格納されている入力パケットを出力して、全体としての出力パケットを得るような動作が実行される。

【０１２３】

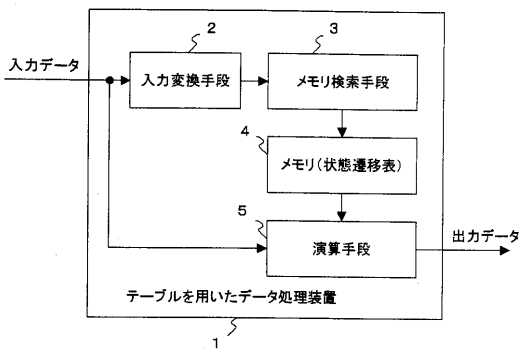
このようなパケットラベリングは、パケット通信処理における基本的な機能を全て含むものである。例えばインターネットプロトコルにおけるパケットのルーティングでは、パケットのヘッダが解析され、ＴＴＬ領域が書き換えられて、次の転送先アドレスがラベルに書き込まれ、パケットの転送が行われる。

【０１２４】
ここでＴＴＬ領域はタイムツーライブ（生存時間）であり、パケットの送信側で、パケットが受信側で受信できるまでの時間を予測して、ＴＴＬの設定が行われる。途中ゲートウェイのＩＰモジュールでは、処理に費やした時間をこの値から減らし、パケットを転送する。処理時間を計測することができない場合でも、少なくともＴＴＬの値から１だけ減算する。途中のＩＰモジュールにおいてＴＴＬの値が０となったパケットが検出された時には、そのパケットは廃棄される。
【０１２５】
更に複雑なパケットラベリングの例としては、セキュリティアーキテクチャフォーインターネットプロトコルＩＰＳｅｃがある。例えばデータをカプセル化してトンネリングする手法を規定するＥＳＰ（エンカプシュレーテッドセキュリティペイロード）では、入力ヘッダの一部を書き換えると共に、それを暗号化／復号化したうえで新しいヘッダ（ラベル）を付け加えたり、逆変換したりする。当然ヘッダの書き換えによってＩＰチェックサムの変更も必要となる。このような複雑な処理を実現するためには、図２６で示したように単純なＦＩＦＯメモリを用いるだけでは不充分であり、複数のストリームプロセッサがパイプライン方式で動作するような構造も必要となる。
【０１２６】
【発明の効果】
以上詳細に説明したように、本発明によれば有限状態機械、例えばストリームプロセッサを状態遷移規則を格納したメモリを備える形式で構成することができ、またこのメモリの格納内容を変更することによって有限状態機械の動作を変更することが可能となる。そしてＦＩＦＯメモリ、レジスタ、各種演算回路を用いることによって、多様な処理を実現することが可能となり、汎用のプロセッサとしての使用が可能である。
【０１２７】
次に本発明によれば、状態遷移表の１つの状態語のアクセスによって、データ入力、解析、状態遷移、複数の演算の実行、およびデータ出力が可能となり、処理の速度は状態遷移表のアクセス性能によって規定される。最近のＶＬＳＩテクノロジを用いれば、100MHz以上の処理性能を実現することができ、ソフトウェアと高価な高速プロセッサとを用いる場合に比較して、低コストで、高速な処理を実現することができ、データ処理装置の基本的な性能向上に寄与するところが大きい。
【図面の簡単な説明】
【図１】本発明の原理構成ブロック図である。
【図２】本発明における有限状態機械の基本構成を示すブロック図である。
【図３】ストリームデータを処理する有限状態機械の説明図である。
【図４】有限状態機械の実行サイクルを説明する図である。
【図５】有限状態機械内部での処理の同期を説明する図である。
【図６】有限状態機械とデータ入出力との同期を説明する図である。
【図７】入力変換機構の構成例を示す図である。
【図８】入力変換処理としてのマスクアンドギャザー機能を説明する図である。
【図９】状態遷移表の基本構造を説明する図である。
【図１０】検索機構の動作を説明する図である。
【図１１】状態遷移表の検索例を説明する図である。
【図１２】演算・出力機構の構成例を示す図である。
【図１３】一般的な演算回路の構成例を示す図である。
【図１４】演算回路としてのカウンタ回路の構成例を示す図である。
【図１５】演算回路としてのバッファメモリの構成例を示す図である。
【図１６】演算回路としてのテーブル検索機構の動作を説明する図である。
【図１７】データ入力部にＦＩＦＯメモリを設けたＤＥＳ演算回路の構成例を示す図である。
【図１８】ＩＰｖ４パケットのヘッダ部分を示す図である。

【図１９】演算回路としてのパターン処理回路の構成例を示す図である。
【図２０】有限状態機械が直列に接続されたパイプライン構成を説明する図である。
【図２１】複数の有限状態機械が並列に接続されたシステムの構成例を示す図である。
【図２２】入力側、および出力側にＦＩＦＯメモリがそれぞれ接続された有限状態機械の並列構成を示す図である。
【図２３】複数の有限状態機械の間での動作の同期をとるための同期回路を持つシステムの構成例を示す図である。
【図２４】有限状態機械の第１の応用例としてのアスキーコード変換処理の説明図である。
【図２５】アスキーコード変換処理における状態遷移のステートマシンを説明する図である。
【図２６】有限状態機械の第２の応用例としてのパケットラベリングを説明する図である。
【図２７】ストアードプログラム方式の計算機におけるストリームデータ処理の説明図である。
【符号の説明】
１　テーブルを用いたデータ処理装置
２　入力変換手段
３　メモリ検索手段
４　メモリ（状態遷移表）
５　演算手段
１０　有限状態機械
１１　入力変換機構
１２　メモリ（状態遷移表）
１３　検索機構
１４　演算・出力機構
１６　入力ＦＩＦＯメモリ
１７　出力ＦＩＦＯメモリ

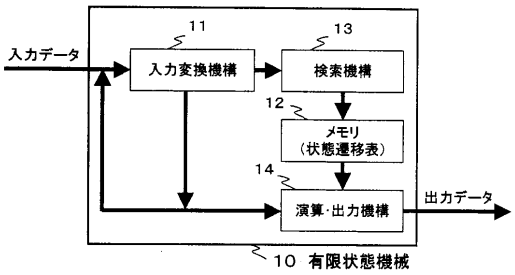
【図１】

本発明の原理構成ブロック図



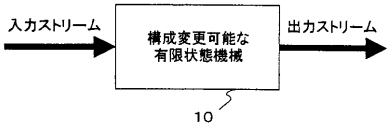
【図２】

本発明における有限状態機械の基本構成を示すブロック図



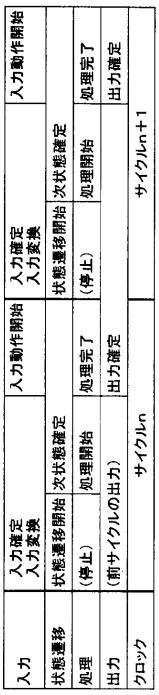
【図３】

ストリームデータを処理する有限状態機械の説明図



【図４】

有限状態機械の
実行サイクルを説明する図



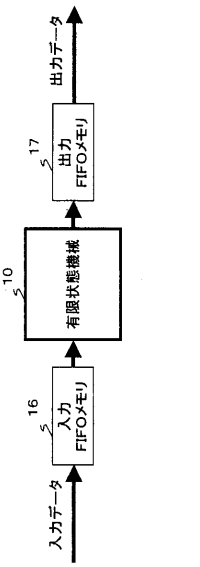
【図５】

有限状態機械内部での
処理の同期を説明する図

入力	入力確定、入力変換					入力動作開始
状態遷移	状態遷移開始	次状態確定				
処理	（停止）	処理開始			処理完了	
出力	（前サイクルの出力）				出力確定	
クロック	サイクルn			サイクルn+1		

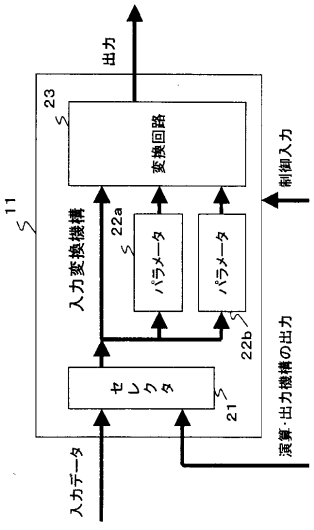
【図６】

有限状態機械とデータ入出力との
同期を説明する図



【図７】

入力変換機構の構成例を示す図



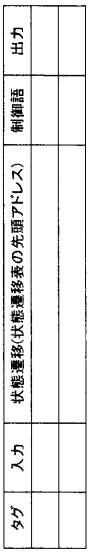
【図８】

入力変換処理としてのマスク
アンドギャザー機能を説明する図

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
入力データ								
MAGパターン	1	0	0	1	1	0	1	0
処理結果	0	0	0	0	bit7	bit4	bit3	bit1

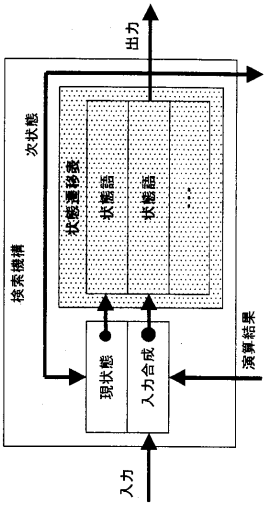
【図９】

状態遷移表の
基本構造を説明する図



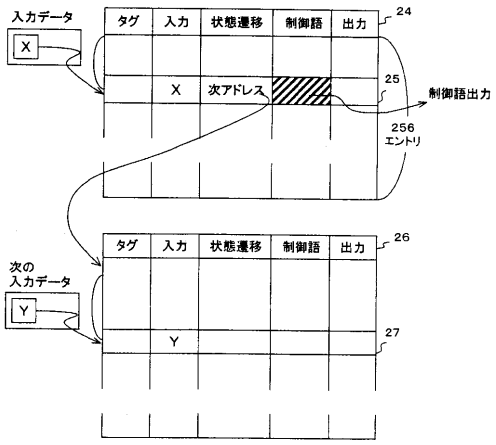
【図１０】

検索機構の動作を説明する図



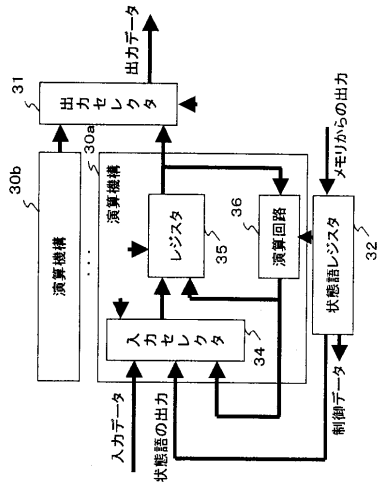
【図１１】

状態遷移表の検索例を説明する図



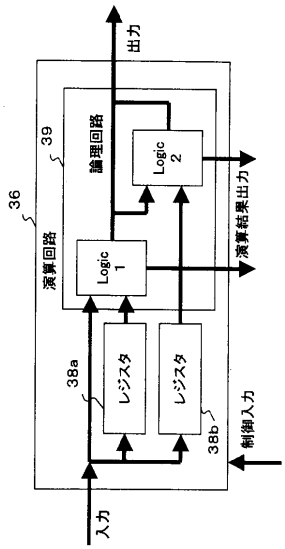
【図１２】

演算・出力機構の構成例を示す図



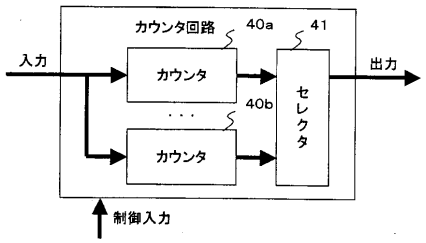
【図１３】

一般的な演算回路の構成例を示す図



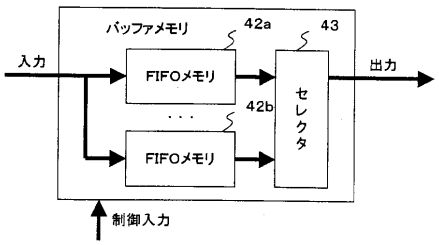
【図１４】

演算回路としてのカウンタ回路の構成例を示す図



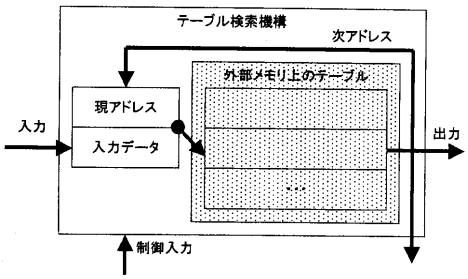
【図１５】

演算回路としてのバッファメモリの構成例を示す図



【図１６】

演算回路としてのテーブル検索機構の動作を説明する図



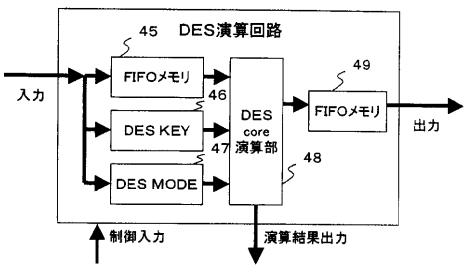
【図１８】

ＩＰｖ４パケットのヘッダ部分を示す図

Ver.	IHL	Type of Service	Total Length (In Octet)
Identification		Flags	Fragment Offset
Time to Live	Protocol	Header Checksum	
Source Address			
Destination Address			
Option (if any)			

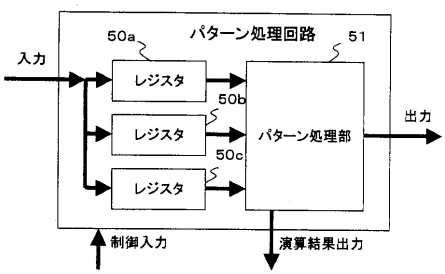
【図１７】

データ入力部にFIFOメモリを設けた
DES演算回路の構成例を示す図



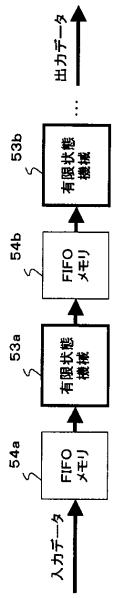
【図１９】

演算回路としてのパターン処理回路の構成例を示す図



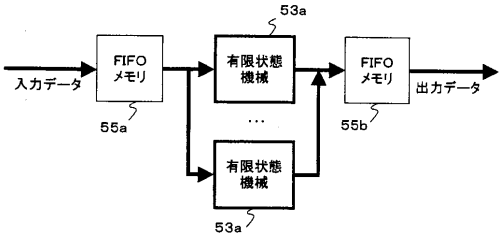
【図２０】

有限状態機械が直列に接続された
パイプライン構成を説明する図



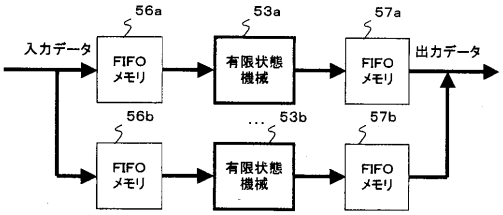
【図２１】

複数の有限状態機械が
並列に接続されたシステムの構成例を示す図



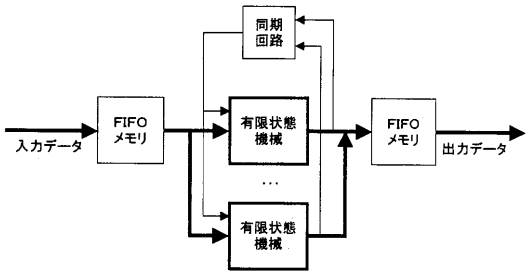
【図２２】

入力側、および出力側にFIFOメモリがそれぞれ接続された
有限状態機械の並列構成を示す図



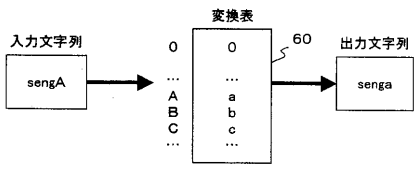
【図２３】

複数の有限状態機械の間での動作の同期をとるための
同期回路を持つシステムの構成例を示す図



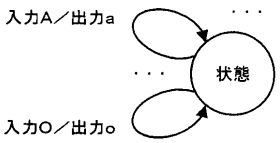
【図２４】

有限状態機械の第1の応用例としての
アスキーコード変換処理の説明図



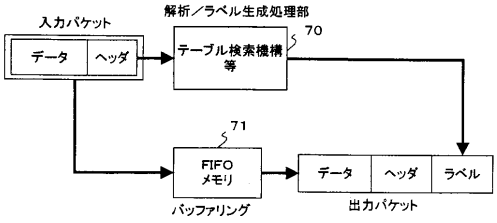
【図２５】

アスキーコード変換処理における
状態遷移のステートマシンを説明する図

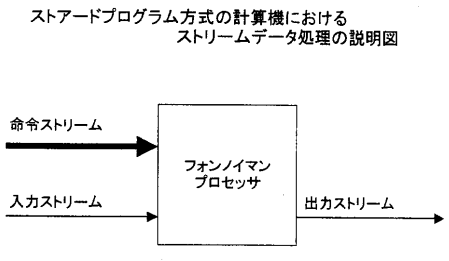


【図２６】

有限状態機械の第2の応用例としての
パケットラベリングを説明する図



【図２７】


ストアードプログラム方式の計算機における
ストリームデータ処理の説明図
命令ストリーム
入力ストリーム
フォンノイマン
プロセッサ
出力ストリーム

フロントページの続き

(56)参考文献 特開平０１－２３４９３０（ＪＰ，Ａ）
大規模広域並列分散システムの実現を目指す超高速インターネットの構想，電子情報通信学会技術研究報告，日本，電子情報通信学会，１９９７年 ８月２０日，Vol.97,No.226，PP.78-80

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 15/82