JP-A No. 2004-287883

There is disclosed a processor provided with plural storage means, a data processing means receiving an input data from a single storage means for inputting among the plural storage means, and storing a process result in a single storage means for outputting with respect to an execution unit which is one of time sharing processing, and an activation priority deciding means determining an execution condition per execution unit from the information of the data storage amount of the storage means and deciding an activation priority per execution.

The storage means is provided at preceding and following of the data processing means, various image processing are executed by the individual data processing means, and the individual data processing means is structured such as to acquire the image data from the storage means in increments of an easily processed unit in correspondence to the kind and the contents of the executed image processing. In the image processing section constructed by coupling the plural image processing modules and the buffer module according to a pipe line aspect or a directed acyclic graph aspect, in the case that a desired image processing is executed by activating the respective modules in parallel, the individual image processing modules and the buffer module execute the process while acquirement and release of the memory necessary for executing the process is repeated, however, if a condition that the memory is not acquired is generated in a certain module, the process is not continued in the module. As a result, there is generated a problem that the entire image processing executed in the image processing section results in failure.

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2004-287883
(43)Date of publication of application : 14.10.2004

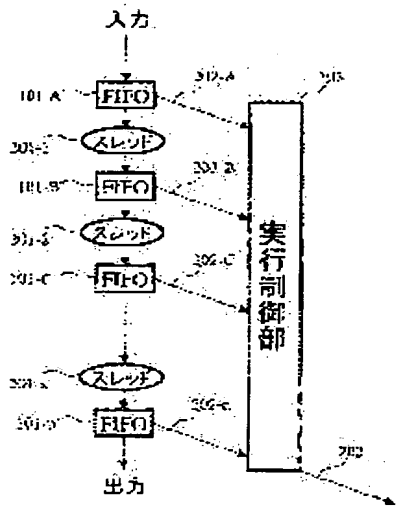
(51)Int.Cl. G06F 9/46

(21)Application number : 2003-079478
(22)Date of filing : 24.03.2003

(71)Applicant : TOSHIBA CORP
(72)Inventor : SUGANO SHINICHI
MIYAMOTO YUKIMASA

(54) PROCESSOR, COMPUTER AND PRIORITY DECISION METHOD

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a data flow type processor that can minimize loss of real-time performance, a computer that uses the processor, and a start priority decision method.
SOLUTION: The processor comprises a plurality of storing means, data processing means for receiving input data from one of the plurality of storing means about an execution unit as a single thread of time-divided processing and storing processing results in output storing means determined as output storing means, and start priority deciding means for computing execution statuses of execution units from information about data storage volumes of the storing means to decide start priority levels of the execution units.



(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2004-287883
(P2004-287883A)
(43)公開日　平成16年10月14日(2004.10.14)

(51)Int.Cl.[7]	F I			テーマコード（参考）
G06F 9/46	G06F	9/46	340B	5B098
	G06F	9/46	340D	

審査請求 有　請求項の数 17　OL　（全 13 頁）

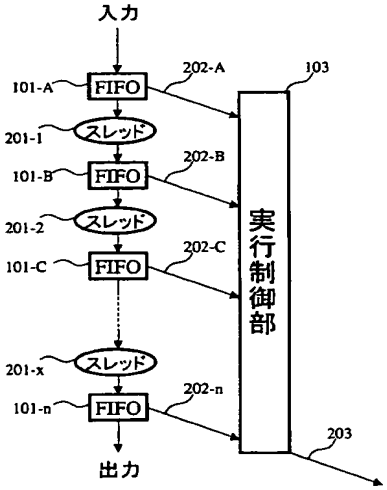
(21)出願番号　特願2003-79478(P2003-79478)
(22)出願日　平成15年3月24日(2003.3.24)

(71)出願人　000003078
株式会社東芝
東京都港区芝浦一丁目1番1号
(74)代理人　100083161
弁理士　外川　英明
(72)発明者　菅野　伸一
神奈川県川崎市幸区小向東芝町1番地　株式会社東芝研究開発センター内
(72)発明者　宮本　幸昌
神奈川県川崎市幸区小向東芝町1番地　株式会社東芝研究開発センター内

Fターム(参考)　5B098 AA03　CC04　GA05　GC03　GC05
GC09

(54)【発明の名称】プロセッサ、計算機及び優先度決定方法

(57)【要約】
【課題】本発明はリアルタイム性の損失を最小限に抑えることが可能なデータフロー型プロセッサ、このプロセッサを利用した計算機及び起動優先度決定方法を提供することを目的とする
【解決手段】複数の記憶手段と、時分割された処理の一つである実行単位について、前記複数の記憶手段の1つから入力データを受け取り、出力用記憶手段と定めた出力用記憶手段に処理した結果を記憶するデータ処理手段と、前記記憶手段のデータ記憶量の情報から実行単位の実行状況を求め、実行単位の起動優先度を決定する起動優先度決定手段とを備えたことを特徴とするプロセッサが提供される。
【選択図】　図2



【特許請求の範囲】
【請求項１】
あるデータ処理を少なくとも１以上に時分割した処理の一つに当たる、１以上の実行単位が処理すべき処理データを記憶するための複数の記憶手段と、
ある実行単位に割り当てた前記複数の記憶手段の少なくとも１つからこの実行単位が処理すべきデータを受け取り、該実行単位に割り当てた前記複数の記憶手段の１つに該実行単位の実行結果を記憶するデータ処理手段と、
前記記憶手段のそれぞれから各記憶手段が記憶する処理データ量についての情報を受け取り、ある記憶手段が記憶する処理データ量から、該記憶手段をアクセスする実行単位の実行状況を求め、該実行状況から該実行単位の以降実行すべき順位を示す優先度を決定する優先度決定手段と、
を備えたことを特徴とするプロセッサ。
【請求項２】
前記優先度決定手段は、ある記憶手段が記憶する処理データ量がより多いほど、該記憶手段から処理すべきデータを受け取る実行単位に対し、より高い優先度を決定することを特徴とする、請求項１に記載のプロセッサ。
【請求項３】
前記優先度決定手段は、ある記憶手段が記憶する処理データ量がより多いほど、該記憶手段に実行結果を記憶する実行単位に対し、より低い優先度を決定することを特徴とする、請求項２に記載のプロセッサ。
【請求項４】
前記優先度決定手段は、ある記憶手段が記憶する処理データ量がより少ないほど、該記憶手段から処理すべきデータを受け取る実行単位に対し、より低い優先度を決定することを特徴とする、請求項１に記載のプロセッサ。
【請求項５】
前記優先度決定手段は、ある記憶手段が記憶する処理データ量がより少ないほど、該記憶手段に実行結果を記憶する実行単位に対し、より高い優先度を決定することを特徴とする、請求項４に記載のプロセッサ。
【請求項６】
前記優先度決定手段は、ある実行単位に対し、該実行単位が処理すべきデータを受け取る記憶手段が記憶している処理データ量が同等であっても、該記憶手段が記憶する処理データ量が増加傾向にあるときと比較して、該記憶手段が記憶する処理データ量が減少傾向にあるときの方により高い優先度を決定することを特徴とする、請求項２及び請求項４に記載のプロセッサ。
【請求項７】
前記優先度決定手段は、ある実行単位に対し、該実行単位が実行結果を記憶する記憶手段が記憶している処理データ量が同等であっても、該記憶手段が記憶する処理データ量が増加傾向にあるときと比較して、該記憶手段が記憶する処理データ量が減少傾向にあるときの方により低い優先度を決定することを特徴とする、請求項３及び請求項５に記載のプロセッサ。
【請求項８】
前記優先度決定手段は、ある記憶手段が記憶する処理データ量が該記憶手段の記憶容量の上限を越えると判断したときは、該記憶手段から処理すべきデータを受け取る実行単位に対し、もっとも高位の優先度を決定することを特徴とする、請求項１に記載のプロセッサ。
【請求項９】
前記優先度決定手段は、ある記憶手段が記憶する処理データ量が該記憶手段の記憶容量の上限を越えると判断したときは、該記憶手段に実行結果を記憶する実行単位に対し、もっとも低位の優先度を決定することを特徴とする、請求項１に記載のプロセッサ。
【請求項１０】

請求項１乃至請求項９のいずれかに記載のプロセッサを備えた計算機。
【請求項１１】
あるデータ処理を少なくとも１以上に時分割した処理の一つに当たる実行単位について、
ある実行単位に割り当てた複数ある記憶装置の少なくとも１つから該実行単位が処理すべきデータを受け取り、
該実行単位に割り当てた前記複数の記憶装置の１つに該実行単位の実行結果を記憶し、
前記記憶装置のそれぞれから各記憶装置が記憶する処理データ量についての情報を受け取り、ある記憶装置が記憶する処理データ量から、該記憶装置をアクセスする実行単位の実行状況を求め、該実行状況から該実行単位の以降実行すべき順位を示す優先度を決定することを特徴とする優先度決定方法。
【請求項１２】
前記優先度を決定するときは、ある記憶装置が記憶する処理データ量がより多いほど、該記憶装置から処理すべきデータを受け取る実行単位に対しより高い優先度を決定し、
かつ記憶する処理データ量がより少ないほど低い優先度を決定することを特徴とする、請求項１１に記載の優先度決定方法。
【請求項１３】
前記優先度を決定するときは、ある記憶装置が記憶する処理データ量がより多いほど、該記憶装置に実行結果を記憶する実行単位に対しより低い優先度を決定し、
かつ記憶する処理データ量がより少ないほど高い優先度を決定することを特徴とする、請求項１１に記載の優先度決定方法。
【請求項１４】
前記優先度を決定するときは、ある実行単位に対し、該実行単位から処理すべきデータを受け取る記憶装置が記憶している処理データ量が同等であっても、該記憶装置が記憶する処理データ量が増加傾向にあるときと比較して、該記憶装置が記憶する処理データ量が減少傾向にあるときの方により高い優先度を決定することを特徴とする、請求項１２に記載の優先度決定方法。
【請求項１５】
前記優先度を決定するときは、ある実行単位について、該実行単位が実行結果を記憶する記憶装置が記憶している処理データ量が同等であっても、該記憶装置が記憶する処理データ量が増加傾向にあるときと比較して、該記憶装置が記憶する処理データ量が減少傾向にあるときの方により低い優先度を決定することを特徴とする、請求項１３に記載の優先度決定方法。
【請求項１６】
前記優先度を決定するとき、ある記憶装置が記憶する処理データ量が該記憶装置の記憶容量の上限を越えると判断したときは、該記憶装置から処理すべきデータを受け取る実行単位に対し、もっとも高位の優先度を決定することを特徴とする、請求項１１に記載の優先度決定方法。
【請求項１７】
前記優先度を決定するとき、ある記憶装置が記憶する処理データ量が該記憶装置の記憶容量の上限を越えると判断したときは、該記憶装置に実行結果を記憶する実行単位に対し、もっとも低位の優先度を決定することを特徴とする、請求項１１に記載の優先度決定方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、データフロー型プロセッサに関するものであり、特に入力段と出力段にバッファメモリを備えたプロセッサ及びこのプロセッサにおける実行スレッドの選択方法に関するものである。
【０００２】
【従来の技術】

計算機等のプロセッサにデータを処理させる際にはより多くのデータ処理を効率よく行うために、複数のタスクを時分割に実行するマルチスレッド処理が一般的に採用されている（たとえば特許文献１を参照）。

【０００３】

データ処理のためのマルチスレッド化を実現する方法がいくつか提案されている。その中にデータフローに従ってスレッドの実行順序を決定するデータフロー型マルチスレッド処理がある。このようなプロセッサでは、処理すべきデータの処理過程で必要とする処理を、その処理順序に従って実行する仕組みが取られている。このようにすると、処理過程にあたらないスレッドは実行されることがなく、その時点で必要とする処理にプロセッサ能力を効率よく割り当てることが可能となる。

【０００４】

【特許文献１】

特開２００３－０２９９８４公報

【０００５】

【発明が解決しようとする課題】

しかしながら、各処理単位の処理後の出力結果に依存して次の実行スレッドを選択する従来のデータフロー型では、次のような場合に問題となる。

【０００６】

たとえばリアルタイム性を要求する処理対象にあっては、処理の流れ（データフロー）に則さない処理が必要となる場合が多い。データフローから逸脱する処理は別途割り込み処理や作業状態の退避処理等のオーバヘッドを必要とする。また、このような割り込み処理はデータフローに乱れを生じさせ、処理効率を低下させてしまう。

【０００７】

本発明はこのような問題に鑑みてなされたものであり、リアルタイム性の損失を最小限に抑えることが可能なデータフロー型プロセッサ、このプロセッサを利用した計算機及び優先度決定方法を提供することを目的とする。

【０００８】

【課題を解決するための手段】

本発明によれば、

あるデータ処理を少なくとも１以上に時分割した処理の一つに当たる、１以上の実行単位が処理すべき処理データを記憶するための複数の記憶手段と、

ある実行単位に割り当てた前記複数の記憶手段の少なくとも１つからこの実行単位が処理すべきデータを受け取り、該実行単位に割り当てた前記複数の記憶手段の１つに該実行単位の実行結果を記憶するデータ処理手段と、

前記記憶手段のそれぞれから各記憶手段が記憶する処理データ量についての情報を受け取り、ある記憶手段が記憶する処理データ量から、該記憶手段をアクセスする実行単位の実行状況を求め、該実行状況から該実行単位の以降実行すべき順位を示す優先度を決定する優先度決定手段と、

を備えたことを特徴とするプロセッサが提供される。

【０００９】

及び上記プロセッサを備えた計算機が提供される。

【００１０】

また、

あるデータ処理を少なくとも１以上に時分割した処理の一つに当たる実行単位について、ある実行単位に割り当てた複数ある記憶装置の少なくとも１つから該実行単位が処理すべきデータを受け取り、

該実行単位に割り当てた前記複数の記憶装置の１つに該実行単位の実行結果を記憶し、

前記記憶装置のそれぞれから各記憶装置が記憶する処理データ量についての情報を受け取り、ある記憶装置が記憶する処理データ量から、該記憶装置をアクセスする実行単位の実行状況を求め、該実行状況から該実行単位の以降実行すべき順位を示す優先度を決定する

ことを特徴とする優先度決定方法が提供される。

【００１１】

【発明の実施の形態】

本発明の実施形態について、図を用いながら説明する。

【００１２】

（第１の実施形態）

図１は第１の実施形態におけるプロセッサのブロック図の一例である。図１には、複数のＦＩＦＯ（Ｆｉｒｓｔ－Ｉｎ　Ｆｉｒｓｔ－Ｏｕｔ：先入先出型メモリ）１０１、選択部１０２、実行制御部１０３、キャッシュ１０４、切替部１０５、複数のレジスタセット１０６、メモリバス１０７、記憶部１０８及び演算処理部１０９が示されている。

【００１３】

ＦＩＦＯ１０１はメモリから構成されており、先に記憶したものから先に読み出すことが可能な記憶装置である。それぞれのＦＩＦＯ１０１は１からｎ（ｎは任意）の複数備えている。それぞれのＦＩＦＯ１０１の記憶容量は必ずしも同じである必要は無いが、複数のスレッドを平等に処理するためには同一の構成である方が好ましい。

【００１４】

ここでいうスレッドとは、プロセッサが実行すべき一つの処理を複数のある時間間隔に分割して割り当てた後の処理単位またはその処理単位の実行を指す。スレッドは実行制御部１０３が行う各部の制御と、この制御下で、予め与えられた所定のプログラムコードを演算処理部１０９で実行することで実現する。本実施形態のプロセッサはデータフロー型なので、分割した複数のスレッドを１つの処理データに対して順次処理してゆくことで一つの処理が完了する。スレッドは並列処理されるようになっており、非同期に複数の処理データのデータフロー型処理を行うことができる。このように一つの処理をスレッド化するのは、主にリアルタイムＯＳとったマルチスレッド処理を実現するために行われている。

【００１５】

選択部１０２は複数備えたＦＩＦＯ１０１の中から実行制御部１０３が指定するＦＩＦＯを選択する機能を備えている。ここで選択したＦＩＦＯ１０１は演算処理部１０９によって読み書きが成される。

【００１６】

実行制御部１０３は、本実施形態のプロセッサが行う処理全般を制御する機能を備えている。ここには、どのスレッドを優先的に実行するかを示す優先度を保持するテーブルも含まれる。実行制御部１０３はＦＩＦＯ１０１などの情報をもとにこのテーブルを設定し、テーブルの内容にしたがってどの処理データを処理するのどのスレッドの実行を行うかを各部に指示する機能を有する。

【００１７】

キャッシュ１０４は演算処理部１０９がスレッドに対応するプログラムコードを実行する際に、読み書き速度の遅い記憶装置へ逐次アクセスすることを回避するために設けられている。一般に演算処理部１０９の処理速度と比較して、メインメモリや磁気ディスク装置といった大容量記憶装置のアクセス速度は遅い。キャッシュ１０４は大容量記憶装置程の記憶容量は無いものの、アクセス速度の速い記憶装置を用いている。頻繁にアクセスするデータおよびプログラムコードについては一旦キャッシュ１０４に記憶することで読み書きの時に演算処理部１０９が待たされるのをなくし、できる限り演算処理部１０９の処理量を稼ぐために用いられる。

【００１８】

切替部１０５は複数のレジスタセット１０６の内から処理に必要な１つ以上のレジスタセットをアクセス可能に選択するものである。演算処理部１０９は、この切替部１０５を介して処理に必要なデータが格納されているレジスタセットをアクセスする。

【００１９】

レジスタセット１０６は実行制御部１０３がプログラムコードの実行に必要となる各種レジスタ（一時記憶装置）をまとめたものである。レジスタセット１０６には、たとえば計

算用レジスタ、アドレッシング用レジスタ、スタックポインタといったものが含まれている。本実施形態ではこのレジスタセット１０６が１からｍ（ｍは任意）の複数備えられている。それぞれのレジスタセット１０６は、必ずしも同じ構成である必要は無いが、それぞれのスレッドがどのレジスタセットを使用しても処理できるためには同一の構成であることが好ましい。

【００２０】

メモリバス１０７は、キャッシュ１０４、複数のレジスタセット１０７及び記憶部１０８の間でデータの授受が行えるように設けられたものである。それぞれに記憶されたデータは、メモリバス１０７を通じてデータの移動が行える。データの移動は実行制御部１０３があたる。

【００２１】

記憶部１０８は、本実施形態のプロセッサが処理のために実行するプログラムコードや、処理の対象となる処理データを記憶する。場合によってはキャッシュ１０４及びレジスタセット１０７に記憶したデータの一時退避場所として使用される。。

【００２２】

演算処理部１０９は、記憶部１０８またはキャッシュ１０４が記憶するプログラムコードを実行する機能を有する。プログラムコードの実行に際しては、実行制御部１０３の指示によりどのＦＩＦＯ１０１、どのレジスタセット１０６を使用して、どのスレッドを処理するのかを決定する。

【００２３】

図２に本実施形態におけるプロセッサのスレッド処理の概要を示す。

本実施形態のプロセッサはデータフロー型であるから、処理データは入力の形でもたらされ、基本的に一つの道筋（フロー）を経て出力が得られる。入力された処理データは、演算処理部１０９が予め記憶部１０８に記憶したプログラムコードに従って逐次実行する処理（各スレッド２０１）を行い、出力結果として出力される。次段の処理がある場合には、出力された出力結果は次段の入力データとなる。

【００２４】

入力データである処理データが入力されると、実行制御部１０３が指示した複数あるＦＩＦＯ１０１の一つであるＦＩＦＯ１０１－Ａがこれを記憶する。ＦＩＦＯ１０１－Ａは実行制御部１０３などからの求めに応じて、あるいは自発的にＦＩＦＯ１０１－Ａ内のＦＩＦＯ蓄積量を実行制御部１０３に対して状態報告２０２－Ａとして通知する。

【００２５】

実行制御部１０３の指示により演算処理部１０９がスレッド２０１－１を実行すると、ＦＩＦＯ１０１－Ａ内のデータを入力データとした処理を開始する。このときＦＩＦＯ１０１－Ａ内のデータは、先に記憶したものから順に読み出される。このスレッドが行った処理結果は実行制御部１０３の指示によりＦＩＦＯ１０１－Ｂに記憶される。ＦＩＦＯ１０１－ＢもまたＦＩＦＯ１０１－Ａと同様に、実行制御部１０３に対してＦＩＦＯ蓄積量を含む状態報告２０２－Ｂを通知する。

【００２６】

上記の処理がスレッド２０１－２からスレッド２０１－ｘ（ｘは任意）まで処理を終えると、処理結果が出力として出力される。

【００２７】

演算処理部１０９がどのスレッドを実行状態とするかは、実行制御部１０３が実行指示２０３によって指示している。実行制御部１０３は、ＦＩＦＯ１０１－ＡからＦＩＦＯ１０１－ｎが通知する状態報告２０２－Ａから状態報告２０２－ｎの情報をもとに自らが備える実行優先度テーブルに優先度を設定し、この優先度から実行指示２０３を決定する。実行指示２０３の決定に際しては、複数の状態報告２０２のうち一部だけを使用して決定した優先度に従っても良いし、すべての状態報告２０２を考慮して決定した優先度から最優先のスレッド２０１を決定してもよい。好ましくはすべての状態報告２０２からそれぞれのスレッド２０１の実行順位を決定するようにし、処理全体の効率化を図るようにした方

が良い。

【００２８】

以降、実行制御部１０３が上記スレッドの実行順位を決定するための方法について説明する。

【００２９】

図３に本実施形態におけるプロセッサの実行制御部１０３が、あるスレッドについての入力側ＦＩＦＯ蓄積量に対するスレッドの実行優先度を決定する方法の一例を示す。図３に示す縦軸に当たるＦＩＦＯ蓄積量は入力側ＦＩＦＯであるから、たとえば図２に示すところのスレッド２０１－１に対するＦＩＦＯ１０１－Ａ、スレッド２０１－２に対するＦＩＦＯ１０１－Ｂとなる。

【００３０】

あるスレッドの入力側ＦＩＦＯに処理待ちデータが多く蓄積されているということは、すなわちそのスレッドの処理が遅れていることを意味する。よってそのスレッドの優先度を上げ、処理待ちデータを速やかに処理する必要がある。このため図３に示すようにＦＩＦＯ蓄積量が増えるに従って実行され易くなる（優先度が高くなる）よう設定する。交点３０２に示すように、ＦＩＦＯ蓄積量が高くなるほど実行制御部１０３が保持する各スレッドの優先度を示すテーブルにより高い優先度が設定される。

【００３１】

図３に示す例ではＦＩＦＯ蓄積量とスレッド実行優先度の関係を比例直線３０１で示しているが、必ずしも直線である必要はない。たとえばＦＩＦＯ蓄積量が上限に近づくに連れて、優先度の上がり方が大きくなるようにした比例曲線３０３であっても良い。この場合にはＦＩＦＯが溢れることを効果的に防止することができる。また、曲線や直線に依らずプロセッサの設計の上で実装が簡易となるように優先度が段階的あるいは階段状に高くなるようにしても良い。一例として、ＦＩＦＯ蓄積量にあるしきい値を設け、このしきい値を越えるまではスレッドの実行をしない、またはしきい値を切るまでは実行しつづけるといった操作も可能である。

【００３２】

図４は本実施形態におけるプロセッサの、あるスレッドににについて出力側ＦＩＦＯ蓄積量に対するスレッドの優先度決定方法の一例を示す。図４に示す縦軸に当たるＦＩＦＯ蓄積量は出力側ＦＩＦＯであるから、たとえば図２に示すところのスレッド２０１－１に対するＦＩＦＯ１０１－Ｂ、スレッド２０１－２に対するＦＩＦＯ１０１－Ｃとなる。

【００３３】

あるスレッドの出力側ＦＩＦＯに処理待ちデータが多く蓄積されているということは、すなわちその後続のスレッドの処理が遅れていることを意味する。このままスレッドを実行すると、出力側のＦＩＦＯが溢れてしまう可能性がある。そこでそのスレッドの優先度を下げ、出力側のＦＩＦＯが溢れないように処理を抑制する必要がある。このため図４に示すように出力側のＦＩＦＯ蓄積量が増えるに従って実行され難くなる（優先度が低くなる）よう設定する。交点４０２に示すように、ＦＩＦＯ蓄積量が高くなるほど実行制御部１０３が保持する各スレッドの優先度を示すテーブルにより低い優先度が設定される。

【００３４】

図４に示す例ではＦＩＦＯ蓄積量とスレッド実行優先度の関係を比例直線４０１で示しているが、必ずしも直線である必要はない。たとえば出力側のＦＩＦＯ蓄積量が上限に近づくに連れて、優先度の下がり方が大きくなるようにした比例曲線４０３であっても良い。この場合には出力側ＦＩＦＯが溢れることを効果的に防止することができる。また、曲線や直線に依らずプロセッサの設計の上で実装が簡易となるように優先度が段階的あるいは階段状に低くなるようにしても良い。一例として、ＦＩＦＯ蓄積量にあるしきい値を設け、このしきい値を切るまではスレッドの実行をしない、またはしきい値を越えるまで実行しつづけるといった操作も可能である。

【００３５】

上記したように実行制御部１０３は入力側ＦＩＦＯから得られた優先度と出力側ＦＩＦＯ

から得られた優先度の両方を総合した優先度、あるいは入力側ＦＩＦＯに基づく優先度に従って、スレッドの優先度を決定する。

【００３６】

このとき入力側ＦＩＦＯ蓄積量から得られた優先度と出力側ＦＩＦＯ蓄積量から得られた優先度とが相反する優先度を示す場合も考えられる。たとえば入力側出力側の両方が蓄積量の上限に近い場合が考えられる。この場合には出力側のＦＩＦＯ蓄積量の空き領域に、入力側ＦＩＦＯに記憶された処理データを処理した後の出力結果が格納できるだけの領域があることを条件に、入力側ＦＩＦＯ蓄積量に基づく優先度を優先する。この場合を除けば、出力側ＦＩＦＯが溢れるのを防止するため出力側ＦＩＦＯ蓄積量に基づく優先度を優先する、あるいはこのスレッドの実行を一時的に禁止する。

【００３７】

図５に本実施形態におけるプロセッサの、次に実行するスレッドを決定するためのフローの一例を示す。

【００３８】

まず、全ＦＩＦＯの蓄積量を取得する（Ｓ１）。取得の方法については、各ＦＩＦＯに問い合わせても良いし、各ＦＩＦＯから自主的に報告されたものであっても構わない。取得の方法については限定しない。

【００３９】

次に、各ＦＩＦＯについて上限の蓄積量を超えそうなＦＩＦＯがあるかどうかを検査する（Ｓ２）。検査の結果超えそうなＦＩＦＯがある場合には、そのＦＩＦＯに関連するスレッドについて緊急処理を行う（Ｓ３）。

【００４０】

緊急処理には、たとえば次のような処理がある。一つには該当するＦＩＦＯを入力側に持つスレッドを割り込みにより最優先で処理させるものである。割り込みとは通常の処理の順序を強制的に変更させる手段を言う。他のスレッドに優先して処理することにより、当該ＦＩＦＯの蓄積量を減らすことができる。もう一つは該当するＦＩＦＯを出力側に持つスレッドの実行を一時的に禁止するものである。当該スレッドの実行を禁止することで、当該ＦＩＦＯに追加されるデータを止め、蓄積量に余裕ができるようにする。

【００４１】

緊急処理の内容は処理状況や処理データの特性により柔軟に決定すべきものであり、上記した例に限定されるものではない。

【００４２】

オーバフローしそうなＦＩＦＯが無いときは、次に全スレッドの優先度を求めたかどうかを判断する（Ｓ４）。

【００４３】

優先度を求めたスレッドがすべでない場合は、処理中あるいは待機中のあるスレッドの一つを選択する（Ｓ５）。そして選択したスレッドが使用する入力側および出力側ＦＩＦＯの蓄積量から、上述した方法で当該スレッドの優先度を決定する（Ｓ６）。決定後、再び全スレッドの優先度を求めたかを判断する（Ｓ４）。

【００４４】

すべてのスレッドについて優先度を求めたと判断すると、優先度を求めたスレッドの中からもっとも高位の優先度を持つスレッドを選択する（Ｓ７）。

【００４５】

上記のようにすると処理中あるいは待機中のスレッドから、次に実行すべきスレッドを選択することができる。

【００４６】

上記したように構成すると、データフロー型プロセッサの処理能力を効率よく必要とする処理に割り当てられるとともに、入力側及び出力側のＦＩＦＯ蓄積量に応じて実行すべきスレッドを選択することが可能となる。

【００４７】

また本実施形態におけるプロセッサを搭載した、データフロー型の処理を行う計算機とすることも可能である。

【００４８】

（第２の実施形態）

第２の実施形態におけるプロセッサのブロック図の一例を示す図、およびスレッド処理の概要を示す図は、第１の実施形態における図１および図２と同様である。

【００４９】

図６に本実施形態におけるプロセッサの、あるスレッドについての入力側ＦＩＦＯ蓄積量に対するスレッドの優先度決定方法の一例を示す。図６に示す縦軸に当たるＦＩＦＯ蓄積量は入力側ＦＩＦＯであるから、たとえば図２に示すところのスレッド２０１－１に対するＦＩＦＯ１０１－Ａ、スレッド２０１－２に対するＦＩＦＯ１０１－Ｂとなる。

【００５０】

あるスレッドの入力側ＦＩＦＯに処理待ちデータが多く蓄積されているということは、すなわちそのスレッドの処理が遅れていることを意味する。よってそのスレッドの優先度を上げ、処理待ちデータを速やかに処理する必要がある。このため図６に示すようにＦＩＦＯ蓄積量が増えるに従って実行され易くなる（優先度が高くなる）よう設定する。

【００５１】

第１の実施形態との相違は、ＦＩＦＯ蓄積量の増加時と減少時で異なるスレッド実行優先度とするところである。同じＦＩＦＯ蓄積量のところを見ても、たとえばＦＩＦＯ蓄積量が増加する際の交点６０１に当たる優先度は、ＦＩＦＯ蓄積量が減少する際の交点６０２に当たる優先度に比べて低くなる。

【００５２】

入力側ＦＩＦＯの蓄積量とスレッド処理との関係を見たとき、スレッドが実行されれば入力側ＦＩＦＯの蓄積量が減ってゆき、逆に実行されなければ増加する傾向にある。このとき交点６０２があるＦＩＦＯ蓄積量の減少時について考えると、入力側ＦＩＦＯの蓄積量が減っていっても、そのスレッドの優先度の下がり方は緩やかになっている。逆に交点６０１があるＦＩＦＯ蓄積量の増加時について考えると、入力側ＦＩＦＯの蓄積量が増えていっても、そのスレッドの優先度の上がり方は緩やかである。つまり入力側ＦＩＦＯの蓄積量が変化しても、一旦処理を始めたスレッドは優先度が高いまま維持されることから連続して実行される可能性が高い。逆に優先度が低くなり実行され難くなったスレッドは、入力側ＦＩＦＯの蓄積量がより高くならないと実行され難いことを意味している。

【００５３】

図１に示したキャッシュ１０４は、前述したようにスレッド処理に必要とする処理データを記憶容量は少ないものの高速にアクセスできるように設けられている。記憶容量が少ないことから、一旦読み込んだあるスレッドの処理データは他のスレッドが実行されると上書きされて消滅する可能性が高い。消滅した処理データは再びアクセス速度の遅い記憶装置からキャッシュ１０４上に読み込まねばならない。入力側ＦＩＦＯの蓄積量の変動により実行されるスレッドが頻繁に入れ替わると、必然的に他の記憶装置からキャッシュ１０４に読み込まねばならないデータの量及び回数が増える。

【００５４】

図１に示したレジスタセット１０６についても同様に、実行中及び待機中のスレッドがレジスタセット１０６の数を上回る場合には、他の記憶装置に一時的に退避する必要がある。これはある時点にレジスタセット１０６に割り当てられなかったスレッドを実行するために、他のスレッドが占有していたレジスタセット１０６をこのスレッドのために開放する必要があるからである。レジスタセット１０６の内容を記憶装置間で移動しなければならないことから、頻繁なスレッドの切替は、キャッシュ１０４と同様にオーバヘッドを発生させる。

【００５５】

本実施形態のプロセッサではあるスレッドが頻繁に実行されやすい状態を実現し、キャッシュ１０４へのデータ読み込みのオーバヘッドを削減する。よってプロセッサの持つ処理

能力を、本来のデータ処理に振り向けることが可能となる。
【００５６】
図６に示す例ではＦＩＦＯ蓄積量とスレッド優先度の関係を楕円で示しているが、必ずしも楕円である必要はない。たとえばプロセッサの設計の上で実装が簡易となるように優先度が段階的あるいは階段状に高くなるようにしても良い。
【００５７】
図７は本実施形態におけるプロセッサの、あるスレッドにについて出力側ＦＩＦＯ蓄積量に対するスレッドの優先度決定方法の一例を示す。図７に示す縦軸に当たるＦＩＦＯ蓄積量は出力側ＦＩＦＯであるから、たとえば図２に示すところのスレッド２０１－１に対するＦＩＦＯ１０１－Ｂ、スレッド２０１－２に対するＦＩＦＯ１０１－Ｃとなる。
【００５８】
あるスレッドの出力側ＦＩＦＯに処理待ちデータが多く蓄積されているということは、すなわちその後続のスレッドの処理が遅れていることを意味する。このままスレッドを実行すると、出力側のＦＩＦＯが溢れてしまう可能性がある。そこでそのスレッドの優先度を下げ、出力側のＦＩＦＯが溢れないように処理を抑制する必要がある。このため図７に示すように出力側のＦＩＦＯ蓄積量が増えるに従って実行され難くなる（優先度が低くなる）よう設定する。
【００５９】
第１の実施形態との相違は、ＦＩＦＯ蓄積量の増加時と減少時で異なるスレッド実行優先度とするところである。同じＦＩＦＯ蓄積量のところを見ても、たとえばＦＩＦＯ蓄積量が増加する際の交点７０２に当たる優先度は、ＦＩＦＯ蓄積量が減少する際の交点７０１に当たる優先度に比べて高くなる。
【００６０】
出力側ＦＩＦＯの蓄積量とスレッド処理との関係を見たとき、スレッドが実行されれば出力側ＦＩＦＯの蓄積量が増えてゆき、逆に実行されなければ減少する傾向にある。このとき交点７０２があるＦＩＦＯ蓄積量の増加時について考えると、出力側ＦＩＦＯの蓄積量が増えていっても、そのスレッドの優先度の下がり方は緩やかになっている。逆に交点７０１があるＦＩＦＯ蓄積量の減少時について考えると、出力側ＦＩＦＯの蓄積量が減っていっても、そのスレッドの優先度の上がり方は緩やかである。つまり出力側ＦＩＦＯの蓄積量が変化しても、一旦処理を始めたスレッドは優先度が高いまま維持されることから連続して実行される可能性が高い。逆に優先度が低くなり実行され難くなったスレッドは、出力側ＦＩＦＯの蓄積量がより低くならないと実行され難いことを意味している。
【００６１】
この効果については、上述した図６の説明について参照されたい。
【００６２】
図７に示す例ではＦＩＦＯ蓄積量とスレッド実行優先度の関係を楕円で示しているが、必ずしも楕円である必要はない。たとえばプロセッサの設計の上で実装が簡易となるように優先度が段階的あるいは階段状に高くなるようにしても良い。
【００６３】
上記したように実行制御部１０３は入力側ＦＩＦＯから得られた優先度と出力側ＦＩＦＯから得られた優先度の両方を総合した優先度、あるいは入力側ＦＩＦＯに基づく優先度に従って、スレッドの優先度を決定する。
【００６４】
このとき入力側ＦＩＦＯ蓄積量から得られた優先度と出力側ＦＩＦＯ蓄積量から得られた優先度とが相反する優先度を示す場合も考えられる。たとえば入力側ＦＩＦＯ及び出力側ＦＩＦＯの両方に処理データが蓄積されていないときである。この場合は優先的にスレッドの実行をする必要はないので入力側ＦＩＦＯ蓄積量に基づく優先度を優先する。
【００６５】
一方、入力側出力側の両方が蓄積量の上限に近い場合も考えられる。この場合には出力側のＦＩＦＯ蓄積量の空き領域に、入力側ＦＩＦＯに記憶された処理データを処理した後の

出力結果が格納できるだけの領域があることを条件に、入力側ＦＩＦＯ蓄積量に基づく優先度を優先する。この場合を除けば、出力側ＦＩＦＯが溢れるのを防止するため出力側ＦＩＦＯ蓄積量に基づく優先度を優先する、あるいはこのスレッドの実行を一時的に禁止する。

【００６６】

本実施形態におけるプロセッサの、次に実行するスレッドを決定するためのフローの一例を示す図は、第１の実施形態における図５と同様である。

【００６７】

なお、本発明は上記実施形態そのままに限定されるものではなく、実施段階ではその要旨を逸脱しない範囲で構成要素を変形して具体化できる。また、上記実施形態に開示されている複数の構成要素の適宜な組み合わせにより、種々の発明を形成できる。例えば、実施形態に示される全構成要素から幾つかの構成要素を削除してもよい。さらに、異なる実施形態にわたる構成要素を適宜組み合わせてもよい。

【００６８】

【発明の効果】

本発明によれば、リアルタイム性の損失を最小限に抑えることが可能なデータフロー型プロセッサ、このプロセッサを利用した計算機及び優先度決定方法を提供することができる。

【図面の簡単な説明】

【図１】本発明の第１の実施形態にかかるプロセッサのブロック構成の一例を示す図である。

【図２】本発明の第１の実施形態にかかるプロセッサのスレッド処理の概要を示す図である。

【図３】本発明の第１の実施形態にかかるプロセッサのあるスレッドについての入力側ＦＩＦＯ蓄積量に対するスレッドの起動優先度決定方法の一例を示す図である。

【図４】本発明の第１の実施形態にかかるプロセッサのあるスレッドについての出力側ＦＩＦＯ蓄積量に対するスレッドの起動優先度決定方法の一例を示す図である。

【図５】本発明の第１の実施形態にかかるプロセッサの次に起動するスレッドを決定するためのフローの一例を示す図である。

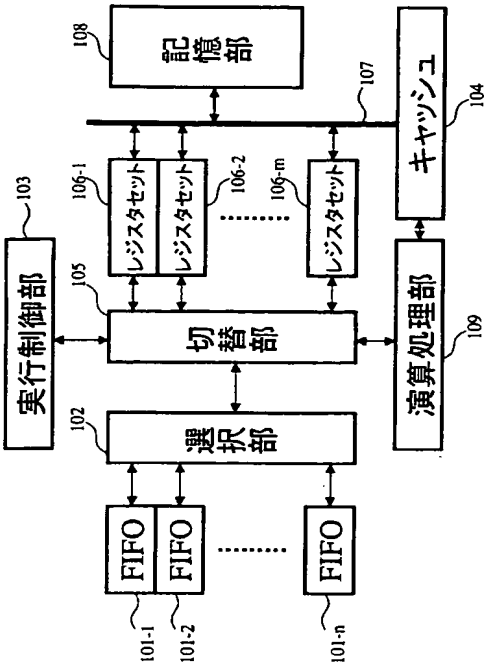
【図６】本発明の第２の実施形態にかかるプロセッサのあるスレッドについての入力側ＦＩＦＯ蓄積量に対するスレッドの起動優先度決定方法の一例を示す図である。

【図７】本発明の第２の実施形態にかかるプロセッサのあるスレッドについての出力側ＦＩＦＯ蓄積量に対するスレッドの起動優先度決定方法の一例を示す図である。

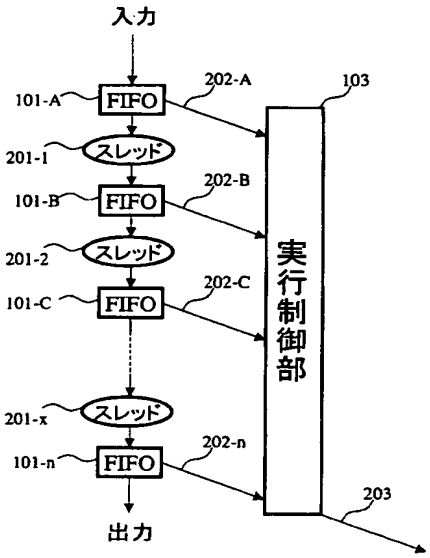
【符号の説明】

１０１　ＦＩＦＯ（Ｆｉｒｓｔ－Ｉｎ　Ｆｉｒｓｔ－Ｏｕｔ）
１０２　選択部
１０３　実行制御部
１０４　キャッシュ
１０５　切替部
１０６　レジスタセット
１０７　メモリバス
１０８　記憶部
１０９　演算処理部
２０２　状態報告
２０３　実行指示

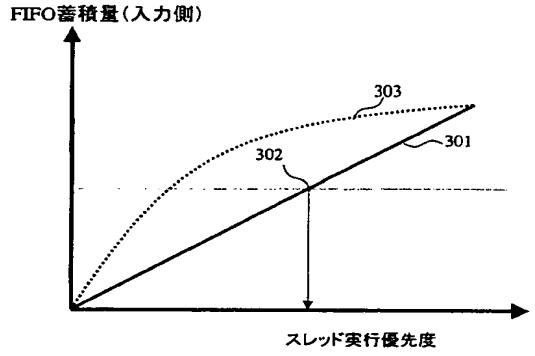
【図１】


記憶部
108
107
キャッシュ
104
レジスタセット
106-1
106-2
106-m
103
実行制御部
105
切替部
109
演算処理部
102
選択部
FIFO
101-1
101-2
101-n

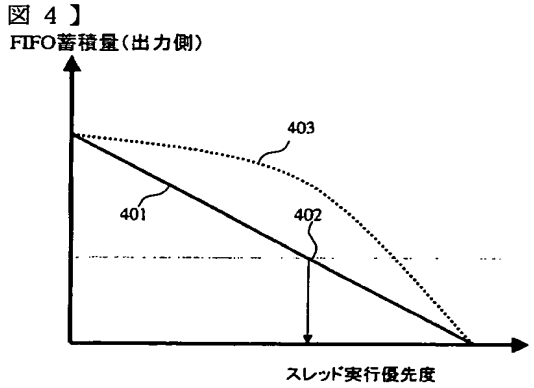
【図２】


入力
FIFO
101-A
202-A
103
スレッド
201-1
101-B
202-B
201-2
101-C
202-C
実行制御部
201-x
101-n
202-n
203
出力

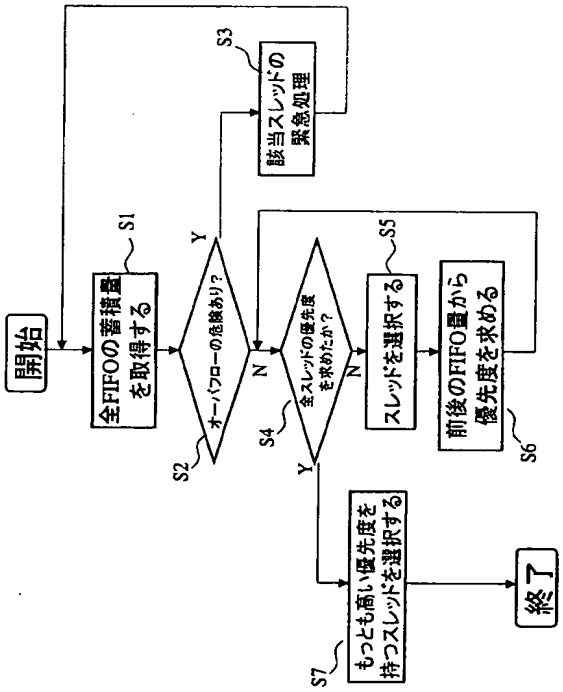
【図３】


FIFO蓄積量(入力側)
303
301
302
スレッド実行優先度

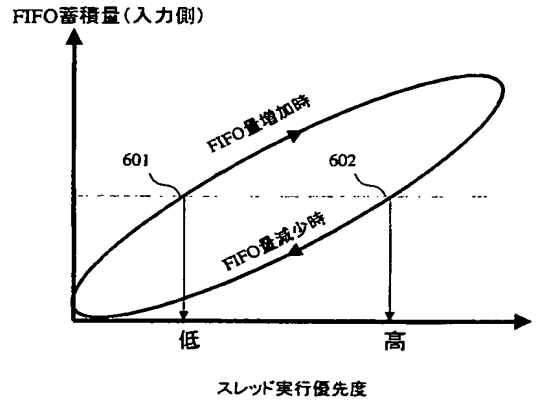
【図４】


FIFO蓄積量(出力側)
403
401
402
スレッド実行優先度

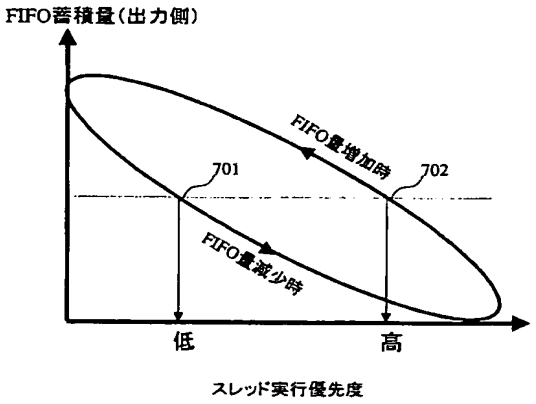
【図５】


開始
S1
全FIFOの蓄積量を取得する
S2
オーバフローの危険あり?
Y
N
S3
該当スレッドの緊急処理
S4
全スレッドの優先度を求めたか?
S5
スレッドを選択する
S6
前後のFIFO量から優先度を求める
S7
もっとも高い優先度を持つスレッドを選択する
終了

【図６】


FIFO蓄積量(入力側)
FIFO量増加時
601
602
FIFO量減少時
低
高
スレッド実行優先度

【図７】


FIFO蓄積量(出力側)
FIFO量増加時
701
702
FIFO量減少時
低
高
スレッド実行優先度