(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)特許出願公表番号
特表2006-515690
(P2006-515690A)
(43)公表日　平成18年6月1日(2006.6.1)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 9/46	(2006.01)	G06F	9/46	360B	5B045
G06F 15/177	(2006.01)	G06F	15/177	674A	5B098

審査請求 有　予備審査請求 未請求　（全 28 頁）

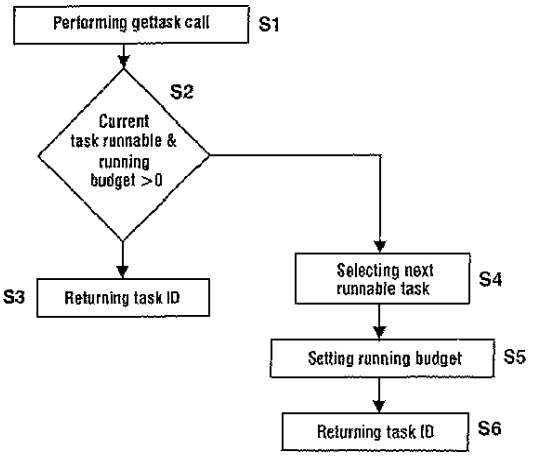
(21)出願番号　特願2003-553417 (P2003-553417)
(86)(22)出願日　平成14年12月5日 (2002.12.5)
(85)翻訳文提出日　平成16年6月11日 (2004.6.11)
(86)国際出願番号　PCT/IB2002/005199
(87)国際公開番号　WO2003/052597
(87)国際公開日　平成15年6月26日 (2003.6.26)
(31)優先権主張番号　01204882.3
(32)優先日　平成13年12月14日 (2001.12.14)
(33)優先権主張国　欧州特許庁 (EP)

(71)出願人　590000248
コーニンクレッカ　フィリップス　エレクトロニクス　エヌ　ヴィ
Ｋｏｎｉｎｋｌｉｊｋｅ　Ｐｈｉｌｉｐｓ　Ｅｌｅｃｔｒｏｎｉｃｓ　Ｎ．Ｖ．
オランダ国　５６２１　ベーアー　アインドーフェン　フルーネヴァウツウェッハ　１
Ｇｒｏｅｎｅｗｏｕｄｓｅｗｅｇ　１，５６２１　ＢＡ　Ｅｉｎｄｈｏｖｅｎ，Ｔｈｅ　Ｎｅｔｈｅｒｌａｎｄｓ
(74)代理人　100092048
弁理士　沢田　雅男

最終頁に続く

(54)【発明の名称】複数のプロセッサを有するデータ処理システムと、複数のプロセッサを有するデータ処理システムのためのタスクスケジューラと、タスクスケジューリングの対応する方法

(57)【要約】
【課題】　カーンスタイルのデータ処理システムの動作を改良すること。
【解決手段】　本発明は、複数のプロセッサを有するデータ処理システムにおいて分散型タスクスケジューリングを提供するための発想に基づいている。従って、データオブジェクトのストリームを処理するための第一プロセッサおよび少なくとも1つの第二プロセッサと、通信ネットワークと、メモリとを有するデータ処理システムであって、第一プロセッサが、データオブジェクトのストリームからのデータオブジェクトを第二プロセッサに渡す、データ処理システムが提供される。第二プロセッサは、第一および第二タスクをインターリーブ式に処理することのできるマルチタスキングプロセッサであり、第一および第二タスクは、それぞれ、データオブジェクトの第一および第二ストリームを処理する。このデータ処理システムは、第二プロセッサのそれぞれのためのタスクスケジューリング手段をさらに有し、タスクスケジューリング手段は、第二プロセッサと通信ネットワークの間に作動的に配置されており、第二プロセッサのタス



【特許請求の範囲】
【請求項１】
－ データオブジェクトのストリームを処理するための第一プロセッサおよび少なくとも1つの第二プロセッサであって、当該第一プロセッサが、データオブジェクトのストリームからのデータオブジェクトを前記第二プロセッサに渡すように構成されており、当該第二プロセッサが、第一および第二タスクをインターリーブ式に処理することのできるマルチタスキングプロセッサであり、当該第一および第二タスクが、それぞれ、データオブジェクトの第一および第二ストリームを処理する、前記第一プロセッサおよび少なくとも1つの第二プロセッサと、
－ 通信ネットワークと、
－ 当該第二プロセッサのそれぞれのためのタスクスケジューリング手段であって、当該タスクスケジューリング手段が、当該第二プロセッサと当該通信ネットワークの間に作動的に配置されている、前記タスクスケジューリング手段と、
を有する、データ処理システムであって、
当該第二プロセッサのそれぞれの前記タスクスケジューリング手段が、当該第二プロセッサの前記タスクスケジューリングを制御する、
データ処理システム。
【請求項２】
当該第二プロセッサが、タスクごとに複数の入力および出力ストリームおよび／または複数のストリームを処理するように構成されている、請求項1に記載のデータ処理システム。
【請求項３】
当該タスクスケジューリング手段が、当該第二プロセッサによって処理される次のタスクを、当該第二プロセッサからの要求を受け取った時点で確定し、かつ、当該次のタスクの識別情報を当該第二プロセッサに転送するように適合化されており、
当該第二プロセッサが、次のタスクを連続的な間隔で要求し、当該間隔が、当該第二プロセッサの前記処理ステップに相当する、
請求項1に記載のデータ処理システム。
【請求項４】
当該第二プロセッサとそれらの関連付けられているタスクスケジューリング手段の間の前記通信が、マスター／スレーブ通信であり、当該第二プロセッサが、マスターとして機能する、
請求項1に記載のデータ処理システム。
【請求項５】
当該第二プロセッサが、一連のパラメータ化されたストリーム処理機能を実行する、機能に固有な専用プロセッサである、
請求項1に記載のデータ処理システム。
【請求項６】
当該タスクスケジューリング手段が、
－ 前記関連付けられているプロセッサにマッピングされている前記タスクに関連付けられている各ストリームのパラメータを格納するためのストリームテーブルであって、当該ストリームテーブルが、ストリームごとの様々な管理データを含んでいる、前記ストリームテーブル、および／または、
－ 当該第二プロセッサに関連付けられている前記異なるタスクを管理するためのタスクテーブルであって、当該タスクテーブルが、どのストリームが当該タスクに関連付けられているかを示す前記ストリームテーブルへのインデックス、前記タスクが実行を許可されているかを示す各タスクのイネーブルフラグ、および／または、各タスクの利用可能な処理割当量を示す割当量カウンタを含んでいる、前記タスクテーブル、
を有する、請求項1に記載のデータ処理システム。
【請求項７】

当該ストリームテーブルが、読み取りのための有効なデータの量、書き込みのための利用可能な空間の量、実行中のタスクが当該ストリームに対する読み取りまたは書き込み時にブロックされているかに関する情報、および／または、当該ストリームをタスクに関係付ける構成設定情報、を含んでいる、
請求項6に記載のデータ処理システム。
【請求項８】
当該タスクスケジューリング手段が、当該ストリームテーブル内のすべてのストリームをチェックし、かつ、当該ストリームのうちのどのストリームがタスクの進行を許可するかを確定するように適合化されており、
a)前記ストリームが読み取りのための有効なデータまたは書き込みのための利用可能な空間を持つ場合、b)前記タスクが、前記ストリームにおいて利用可能である以上の有効なデータまたはスペースを要求しなかった場合、および／または、c)オプションa)およびb)がタスクの進行に無関係であるものとして構成設定されている場合、にストリームが進行を許可する、
請求項6に記載のデータ処理システム。
【請求項９】
当該タスクスケジューリング手段が、当該タスクテーブル内のすべてのタスクをチェックし、かつ、当該タスクのうちのどのタスクが実行を許可されるかを確定するように適合化されており、
当該タスクに関連付けられている前記ストリームのすべてがタスクの進行を許可し、かつ前記タスクが実行可能と構成設定されている場合に、タスクが実行を許可される、
請求項6または8に記載のデータ処理システム。
【請求項１０】
当該タスクスケジューリング手段が、構成設定されている複数のタスクから1つのタスクを、前記次に処理されるタスクとして選択するように適合化されている、
請求項6、7、8、または9に記載のデータ処理システム。
【請求項１１】
当該タスクスケジューリング手段が、前記現在のタスクの前記リソース割当量を制御するための割当量カウンタ手段を有する、
請求項1または9に記載のデータ処理システム。
【請求項１２】
当該タスクスケジューリング手段が、タスクごとのリソース割当量パラメータを利用するように適合化されており、当該リソース割当量パラメータが、プロセッサが前記関係付けられているタスクによって連続的に占有されている時間を制限する、
請求項1または11に記載のデータ処理システム。
【請求項１３】
当該タスクスケジューリング手段が、前記現在のタスクの後に次に処理されるタスクを、当該第二プロセッサからの要求を受け取った時点で選択するように適合化されており、
前記現在のタスクが依然として実行を許可されており、かつそのリソース割当量が使い果たされていない場合に、前記現在のタスクが続行を許可され、
これ以外の場合には、当該タスクスケジューリング手段によって確定された次のタスクが、新しい現在のタスクとして選択される、
請求項12に記載のデータ処理システム。
【請求項１４】
当該タスクスケジューリング手段が、実行を許可されている前記次のタスクをラウンドロビン順序にて選択するように適合化されている、
請求項13に記載のデータ処理システム。
【請求項１５】
当該タスクスケジューリング手段が、選択された次のタスクの識別情報を当該第二プロセッサにただちに戻すことができるように、当該第二プロセッサが前記次のタスクを要求

する前に、次に処理されるタスクを選択するように適合化されている、
請求項1に記載のデータ処理システム。
【請求項１６】
当該割当量カウンタが、リアルタイムクロックに基づいてイベントによって更新される、
請求項12、13、または14に記載のデータ処理システム。
【請求項１７】
当該タスクスケジューリング手段が、次のタスクが前記現在のタスクとなるように選択されたときに当該次のタスクの前記割当量を補充するように適合化されている、
請求項12、13、または14に記載のデータ処理システム。
【請求項１８】
当該第二プロセッサが、一連のプログラム可能なパラメータ化されたストリーム処理機能を実行するプログラム可能なプロセッサである、
請求項1に記載のデータ処理システム。
【請求項１９】
データ処理システムのためのタスクスケジューラであって、当該システムが、
－ データオブジェクトのストリームを処理するための第一プロセッサおよび少なくとも1つの第二プロセッサであって、当該第一プロセッサが、データオブジェクトのストリームからのデータオブジェクトを前記第二プロセッサに渡すように構成されている、前記第一プロセッサおよび少なくとも1つの第二プロセッサと、
－ 通信ネットワークと、
－ メモリと、
を有する、データ処理システムのためのタスクスケジューラであって、
－ 前記タスクスケジューラが、当該第二プロセッサのうちの1つに関連付けられるように適合化されており、
－ 前記タスクスケジューラが、当該第二プロセッサと当該通信ネットワークの間に作動的に配置されているように適合化されており、かつ、
－ 前記タスクスケジューラが、当該関連付けられている第二プロセッサのタスクスケジューリングを制御するように適合化されている、
データ処理システムのためのタスクスケジューラ。
【請求項２０】
当該タスクスケジューラが、当該第二プロセッサによって処理される次のタスクを、当該第二プロセッサからの要求を受け取った時点で確定し、かつ、当該次のタスクの識別情報を当該第二プロセッサに転送するように適合化されており、
当該第二プロセッサが、次のタスクを所定の間隔で要求し、当該間隔が、当該第二プロセッサの前記処理ステップに相当する、
請求項19に記載のタスクスケジューラ。
【請求項２１】
－ 前記関連付けられているプロセッサにマッピングされている前記タスクに関連付けられている各ストリームのパラメータを格納するためのストリームテーブルであって、当該ストリームテーブルが、ストリームごとの様々な管理データを含んでいる、ストリームテーブル、および／または、
－ 当該第二プロセッサに関連付けられている前記異なるタスクを管理するためのタスクテーブルであって、当該タスクテーブルが、どのストリームが当該タスクに関連付けられているかを示す前記ストリームテーブルへのインデックス、前記タスクが実行を許可されているかを示す各タスクのイネーブルフラグ、および／または、各タスクの利用可能な処理割当量を示す割当量カウンタを含んでいる、タスクテーブル、
をさらに有する、請求項19に記載のタスクスケジューラ。
【請求項２２】
当該ストリームテーブルが、読み取りのための有効なデータの量、書き込みのための利

用可能な空間の量、実行中のタスクが当該ストリームに対する読み取りまたは書き込み時にブロックされているかに関する情報、および／または、当該ストリームをタスクに関係付ける構成設定情報、を含んでいる、
請求項19に記載のタスクスケジューラ。
【請求項２３】
当該ストリームテーブル内のすべてのストリームをチェックし、かつ、当該ストリームのうちのどのストリームがタスクの進行を許可するかを確定するように適合化されており、
a)前記ストリームが読み取りのための有効なデータまたは書き込みのための利用可能な空間を持つ場合、b)前記タスクが、前記ストリームにおいて利用可能である以上の有効なデータまたはスペースを要求しなかった場合、および／または、c)オプションa)およびb)がタスクの進行に無関係であるものとして構成設定されている場合、にストリームが進行を許可する、
請求項21に記載のタスクスケジューラ。
【請求項２４】
当該タスクテーブル内のすべてのタスクをチェックし、かつ、当該タスクのうちのどのタスクが実行を許可されるかを確定するように適合化されており、
当該タスクに関連付けられている前記ストリームのすべてがタスクの進行を許可し、かつ前記タスクが実行可能と構成設定されている場合に、タスクが実行を許可される、
請求項21または23に記載のタスクスケジューラ。
【請求項２５】
前記現在のタスクの後に次に処理されるタスクを、当該第二プロセッサからの要求を受け取った時点で選択するように適合化されており、
前記現在のタスクが依然として実行を許可されており、かつ当該タスクテーブル内の割当量カウンタが0でない場合に、前記現在のタスクが続行を許可され、
これ以外の場合には、当該タスクスケジューリング手段によって確定された次のタスクが現在のタスクとして選択され、かつ、前記割当量カウンタがリセットされる、
請求項24に記載のタスクスケジューラ。
【請求項２６】
構成設定されている複数のタスクから1つのタスクを、前記次に処理されるタスクとして選択するように適合化されている、
請求項21、22、23、または24に記載のタスクスケジューラ。
【請求項２７】
前記現在のタスクの前記リソース割当量を制御するための割当量カウンタ手段、
を有する、請求項19または24に記載のタスクスケジューラ。
【請求項２８】
タスクごとのリソース割当量パラメータを利用するように適合化されており、当該リソース割当量パラメータが、プロセッサが前記関係付けられているタスクによって連続的に占有されている時間を制限する、
請求項19または27に記載のタスクスケジューラ。
【請求項２９】
前記現在のタスクの後に次に処理されるタスクを、当該第二プロセッサからの要求を受け取った時点で選択するように適合化されており、
前記現在のタスクが依然として実行を許可されており、かつそのリソース割当量が使い果たされていない場合に、前記現在のタスクが続行を許可され、
これ以外の場合には、当該タスクスケジューリング手段によって確定された次のタスクが、新しい現在のタスクとして選択される、
請求項28に記載のタスクスケジューラ。
【請求項３０】
当該タスクスケジューリング手段が、実行を許可されている前記次のタスクをラウンド

ロビン順序にて選択するように適合化されている、
請求項29に記載のタスクスケジューラ。
【請求項３１】
次のタスクが前記現在のタスクとなるように選択されたときに当該次のタスクの前記割当量を補充するように適合化されている、
請求項28、29、または30に記載のタスクスケジューラ。
【請求項３２】
データ処理システムにおけるタスクスケジューリングのための方法であって、当該システムが、
－ データオブジェクトのストリームを処理するための第一プロセッサおよび少なくとも1つの第二プロセッサであって、当該第一プロセッサがデータオブジェクトのストリームからのデータオブジェクトを前記第二プロセッサに渡すように構成されている、前記第一プロセッサおよび少なくとも1つの第二プロセッサと、
－ 通信ネットワークと、
を有する、タスクスケジューリングのための方法であって、
当該システムが、当該第二プロセッサのそれぞれのためのタスクスケジューラを有し、これによって、
前記タスクスケジューラが、当該第二プロセッサの前記タスクスケジューリングを制御する、
タスクスケジューリングのための方法。
【請求項３３】
－ 当該第二プロセッサによって処理される次のタスクを、当該第二プロセッサからの要求を受け取った時点で確定するステップと、
－ 当該次のタスクの識別情報を当該第二プロセッサに転送するステップと、
をさらに有する、請求項32に記載のタスクスケジューリングのための方法であって、
当該第二プロセッサが、次のタスクを連続的な間隔で要求し、当該間隔が、当該第二プロセッサの前記処理ステップに相当する、
タスクスケジューリングのための方法。
【請求項３４】
当該第二プロセッサとそれらの関連付けられているタスクスケジューリング手段の間の前記通信が、マスター／スレーブ通信であり、当該第二プロセッサが、マスターとして機能する、
請求項32に記載のタスクスケジューリングのための方法。
【請求項３５】
－ 前記関連付けられているプロセッサにマッピングされている前記タスクに関連付けられている各ストリームのパラメータをストリームテーブルに格納するステップであって、当該ストリームテーブルが、ストリームごとの様々な管理データを含んでいる、ステップ、および／または、
－ 当該第二プロセッサに関連付けられている前記異なるタスクをタスクテーブルによって管理するステップであって、当該タスクテーブルが、どのストリームが当該タスクに関連付けられているかを示す前記ストリームテーブルへのインデックス、前記タスクが実行を許可されているかを示す各タスクのイネーブルフラグ、および／または、各タスクの利用可能な処理割当量を示す割当量カウンタを含んでいる、ステップと、
をさらに有する、請求項32に記載のタスクスケジューリングのための方法。
【請求項３６】
当該ストリームテーブルが、読み取りのための有効なデータの量、書き込みのための利用可能な空間の量、実行中のタスクが当該ストリームに対する読み取りまたは書き込み時にブロックされているかに関する情報、および／または、当該ストリームをタスクに関係付ける構成設定情報、を含んでいる、
請求項35に記載のタスクスケジューリングのための方法。

【請求項３７】
当該ストリームテーブル内のすべてのストリームをチェックし、かつ、当該ストリームのうちのどのストリームがタスクの進行を許可するかを確定するステップ、
をさらに有する、請求項35に記載のタスクスケジューリングのための方法であって、
a)前記ストリームが読み取りのための有効なデータまたは書き込みのための利用可能な空間を持つ場合、b)前記タスクが、前記ストリームにおいて利用可能である以上の有効なデータまたはスペースを要求しなかった場合、および／または、c)オプションa)およびb)がタスクの進行に無関係であるものとして構成設定されている場合、にストリームが進行を許可する、
タスクスケジューリングのための方法。
【請求項３８】
当該タスクテーブル内のすべてのタスクをチェックし、かつ、当該タスクのうちのどのタスクが実行を許可されるかを確定するステップ、
をさらに有する、請求項35または37に記載のタスクスケジューリングのための方法であって、
当該タスクに関連付けられている前記ストリームのすべてがタスクの進行を許可し、かつ前記タスクが実行可能と構成設定されている場合に、タスクが実行を許可される、
タスクスケジューリングのための方法。
【請求項３９】
構成設定されている複数のタスクから1つのタスクを、前記次に処理されるタスクとして選択するステップ、
をさらに有する、請求項35、36、37、または38に記載のタスクスケジューリングのための方法。
【請求項４０】
前記現在のタスクの前記リソース割当量を制御するステップ、
をさらに有する、請求項32または39に記載のタスクスケジューリングのための方法。
【請求項４１】
タスクごとのリソース割当量パラメータを利用するステップであって、当該リソース割当量パラメータが、プロセッサが前記関係付けられているタスクによって連続的に占有されている時間を制限する、前記ステップ、
をさらに有する、請求項32または40に記載のタスクスケジューリングのための方法。
【請求項４２】
前記現在のタスクの後に次に処理されるタスクを、当該第二プロセッサからの要求を受け取った時点で選択するステップ、
をさらに有する、請求項41に記載のタスクスケジューリングのための方法であって、
前記現在のタスクが依然として実行を許可されており、かつそのリソース割当量が使い果たされていない場合に、前記現在のタスクが続行を許可され、
これ以外の場合には、当該タスクスケジューリング手段によって確定された次のタスクが、新しい現在のタスクとして選択される、
タスクスケジューリングのための方法。
【請求項４３】
実行を許可されている前記次のタスクをラウンドロビン順序にて選択するステップ、
をさらに有する、請求項42に記載のタスクスケジューリングのための方法。
【請求項４４】
選択された次のタスクの識別情報を当該第二プロセッサにただちに戻すことができるように、当該第二プロセッサが前記次のタスクを要求する前に、次に処理されるタスクを選択するステップ、
をさらに有する、請求項32に記載のタスクスケジューリングのための方法。
【請求項４５】
当該割当量カウンタをリアルタイムクロックに基づいてイベントによって更新するステ

ップ、
をさらに有する、請求項41、42、または43に記載のタスクスケジューリングのための方法。
【請求項４６】
次のタスクが前記現在のタスクとなるように選択されたときに当該次のタスクの前記割当量を補充するステップ、
をさらに有する、請求項41、42、または43に記載のタスクスケジューリングのための方法。
【請求項４７】
前記タスクスケジューラをプログラム可能な第二プロセッサ上に実装するステップ、
をさらに有する、請求項1に記載のタスクスケジューリングのための方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、複数のプロセッサを有するデータ処理システムと、複数のプロセッサを有するデータ処理システムのためのタスクスケジューラと、タスクスケジューリングの対応する方法とに関する。
【背景技術】
【０００２】
データ依存性のハイパフォーマンスメディア処理、例えば高品位(high－definition) MPEG復号化のための異種マルチプロセッサアーキテクチャは公知である。メディア処理アプリケーションは、単方向のデータストリームによってのみ情報を交換する一連の同時実行タスクとして指定することができる。G. Kahnは、このようなアプリケーションの形式モデルを、「パラレルプログラミング用の単純な言語のセマンティクス（The Semantics of a Simple Language for Parallel Programming）」（Proc. of the IFIP congress 74、8月5～10日、スウェーデン、ストックホルム、North－Holland publ. Co, 1974年、p.471～475）ですでに1974年に紹介し、その後、1977年に、KahnおよびMacQueenが、「パラレルプログラミングのコルーチンとネットワーク（Co－routines and Networks of Parallel Programming）」（Information Processing 77、B. Gilchhirst (Ed.)、North－Holland publ、1977年、p.993～998）で動作を説明した。この形式モデルは、現在では一般にKahnプロセスネットワークと呼ばれている。
【０００３】
一連の同時実行可能タスクとしてのアプリケーションは、公知である。情報は、タスク間で単方向のデータストリームによって交換しかできない。タスクは、事前定義されたデータストリームに関する読み取りおよび書き込みアクションによって決定論的にのみ通信する必要がある。データストリームは、FIFO挙動に基づいてバッファリングされる。このバッファリングによって、ストリームを通じて通信する2つのタスクは、個々の読み取りまたは書き込みアクションにおいて同期する必要がない。
【０００４】
ストリーム処理においては、データストリームに対する連続的な操作が、異なるプロセッサによって実行される。例えば、第一ストリームは、イメージのピクセル値から成り、これらが第一プロセッサによって処理されて、ピクセルの8×8ブロックのDCT（離散コサイン変換）係数のブロックの第二ストリームが生成される。第二プロセッサは、このDCT係数のブロックを処理して、DCT係数の各ブロックに対する、選択および圧縮された係数のブロックのストリームを生成することができる。
【０００５】
図1は、先行技術から知られている、アプリケーションからプロセッサへのマッピングの図解を示している。データストリーム処理を実現するために、多数のプロセッサが設けられており、各プロセッサは、特定の操作を繰り返し実行することができ、各回に、データオブジェクトのストリームからの次のデータオブジェクトからのデータを使用する、お

よび／または、そのようなストリームにおける次のデータオブジェクトを生成する。ストリームは1つのプロセッサから別のプロセッサに渡され、従って、第一プロセッサによって生成されたストリームを第二プロセッサによって処理することができ、以下同様である。第一プロセッサから第二プロセッサにデータを渡す1つのメカニズムは、第一プロセッサによって生成されたデータブロックをメモリに書き込むことによる。

【０００６】

ネットワーク内のデータストリームは、バッファリングされる。各バッファは、FIFOとして実現されており、1つのみのライターと1つ以上のリーダーとを有する。このバッファリングによって、ライターとリーダーは、チャネル上の個々の読み取りおよび書き込みアクションを相互に同期させる必要がない。利用可能なデータが十分にないチャネルからの読み取りは、それに起因して読み取りタスクが停止する。プロセッサは、弱くのみプログラム可能な専用ハードウェアの機能装置でよい。すべてのプロセッサは平行に動作し、それぞれ自身の制御のスレッドを実行する。これらのプロセッサは、まとまって、Kahn型アプリケーションを実行し、Kahn型アプリケーションでは、各タスクは1つのプロセッサにマッピングされる。これらのプロセッサは、マルチタスキングを可能とし、すなわち、複数のKahnタスクを1つのプロセッサにマッピングすることができる。

【０００７】

【非特許文献１】G. Kahn「パラレルプログラミング用の単純な言語のセマンティクス（The Semantics of a Simple Language for Parallel Programming）」（Proc. of the IFIP congress 74、8月5～10日、スウェーデン、ストックホルム、North－Holland publ. Co, 1974年、p.471～475）

【非特許文献２】Kahn、MacQueen「パラレルプログラミングのコルーチンとネットワーク（Co－routines and Networks of Parallel Programming）」（Information Processing 77、B. Gilchhirst (Ed.)、North－Holland publ、1977年、p.993～998）

【発明の開示】

【課題を解決するための手段】

【０００８】

本発明の目的は、カーン型データ処理システムの動作を改良することである。

【０００９】

この目的は、請求項1によるデータ処理システムと、請求項19によるタスクスケジューラと、請求項32によるタスクスケジューリングの対応する方法とによって解決される。

【００１０】

本発明は、複数のプロセッサを有するデータ処理システムにおいて分散型タスクスケジューリングを提供するための発想に基づいている。従って、データオブジェクトのストリームを処理するための第一プロセッサおよび少なくとも1つの第二プロセッサであって、当該第一プロセッサが、データオブジェクトのストリームからのデータオブジェクトを前記第二プロセッサに渡す、前記第一プロセッサおよび少なくとも1つの第二プロセッサと、通信ネットワークとを有する、データ処理システムが提供される。当該第二プロセッサは、第一および第二タスクをインターリーブ式に処理することのできるマルチタスキングプロセッサであり、当該第一および第二タスクが、それぞれ、データオブジェクトの第一および第二ストリームを処理する。当該データ処理システムは、当該第二プロセッサのそれぞれのためのタスクスケジューリング手段をさらに有し、当該タスクスケジューリング手段は、当該第二プロセッサと当該通信ネットワークの間に作動的に配置されており、当該第二プロセッサの前記タスクスケジューリングを制御する。

【００１１】

第二プロセッサそれぞれが自身のタスクスケジューラを有する分散型タスクスケジューリングは、有利である。なぜなら、スケーラブルシステムの前提条件である、第二プロセッサを自立させることが可能となるためである。

【００１２】

本発明の1つの観点においては、当該タスクスケジューリング手段は、当該第二プロセ

ッサによって処理される次のタスクを、当該第二プロセッサからの要求を受け取った時点で確定し、かつ、当該次のタスクの識別情報を当該第二プロセッサに転送する。当該第二プロセッサは、次のタスクを所定の間隔で要求し、当該間隔は、当該第二プロセッサの前記処理ステップに相当する。従って、ノンプリエンプティブタスクスケジューリングを実現することができる。

【００１３】

本発明の好ましい観点においては、当該タスクスケジューリング手段は、ストリームテーブルとタスクテーブルとを有する。当該ストリームテーブルは、前記関連付けられているプロセッサにマッピングされている前記タスクに関連付けられている各ストリームのパラメータを格納するために使用される。当該パラメータは、読み取りのための有効なデータの量、書き込みのための利用可能な空間の量、実行中のタスクが当該ストリームに対する読み取りまたは書き込み時にブロックされているかに関する情報、および／または、当該ストリームをタスクに関係付ける構成設定情報を含む。当該タスクテーブルは、当該第二プロセッサに関連付けられている前記異なるタスクを管理するために使用され、当該タスクテーブルは、どのストリームが当該タスクに関連付けられているかを示す前記ストリームテーブルへのインデックス、前記タスクが実行を許可されているか否かを示す各タスクのイネーブルフラグ、および／または、各タスクの利用可能な処理割当量(budget)を示す割当量カウンタを含む。第二プロセッサに関連付けられているタスクスケジューリング手段にストリームテーブルとタスクテーブルとを設けることにより、データ処理システムのローカル制御および管理能力が向上する。

【００１４】

本発明のさらに別の観点においては、当該タスクスケジューリング手段は、当該ストリームテーブル内のすべてのストリームをチェックし、かつ、当該ストリームのうちのどのストリームがタスクの進行を許可するかを確定する。a)前記ストリームが読み取りのための有効なデータまたは書き込みのための利用可能な空間を持つ場合、b)前記タスクが、前記ストリームにおいて利用可能である以上の有効なデータまたはスペースを要求しなかった場合、および／または、c)オプションa)およびb)がタスクの進行に無関係であるものとして構成設定されている場合に、ストリームは進行を許可する。

【００１５】

本発明のさらなる観点においては、当該タスクスケジューリング手段は、当該タスクテーブル内のタスクをチェックし、当該タスクのうちのどのタスクが実行を許可されるかを確定する。タスクは、当該タスクに関連付けられている前記ストリームのすべてが実行を許可されており、かつ当該タスクの前記イネーブルフラグが設定されている場合に、実行が許可される。

【００１６】

本発明のさらに別の観点においては、当該タスクスケジューリング手段は、前記現在のタスクの後に次に処理されるタスクを、当該第二プロセッサからの要求を受け取った時点で選択し、前記現在のタスクは、前記現在のタスクが依然として実行を許可されておりかつ当該タスクテーブル内の割当量カウンタが0でない場合に、続行が許可される。これ以外の場合には、当該タスクスケジューリング手段によって確定された前記次のタスクが、現在のタスクとして選択され、かつ、前記割当量カウンタがリセットされる。このように、第二プロセッサにマッピングされている各タスクが、第二プロセッサ上で実行される機会を定期的に得ることが、保証される。

【００１７】

本発明の別の観点においては、当該タスクスケジューリング手段は、選択された次のタスクの識別情報を当該第二プロセッサにただちに戻すことができるように、当該第二プロセッサが前記次のタスクを要求する前に、次に処理されるタスクを選択する。従って、データ処理システムの処理速度が向上する。

【００１８】

本発明のさらに別の観点においては、当該タスクスケジューリング手段は、前記現在の

タスクの前記割当量カウンタを制御する割当量カウンタ手段を有する。各タスクに対して割当量カウンタを設けることにより、相異なるタスクの処理における正当性(justice)の実施が確保される。

【００１９】

本発明は、データ処理システムのためのタスクスケジューラにも関する。当該システムは、データオブジェクトのストリームを処理するための第一プロセッサおよび少なくとも1つの第二プロセッサであって、当該第一プロセッサが、データオブジェクトのストリームからのデータオブジェクトを前記第二プロセッサに渡すように構成されている、前記第一プロセッサおよび少なくとも1つの第二プロセッサと、通信ネットワークと、メモリとを有する。前記タスクスケジューラは、当該第二プロセッサのうちの1つに関連付けられており、当該第二プロセッサと当該通信ネットワークの間に作動的に配置されており、かつ、当該関連付けられている第二プロセッサのタスクスケジューリングを制御する。

【００２０】

本発明は、データ処理システムにおけるタスクスケジューリングのための方法にも関する。当該システムは、データオブジェクトのストリームを処理するための第一プロセッサおよび少なくとも1つの第二プロセッサであって、当該第一プロセッサがデータオブジェクトのストリームからのデータオブジェクトを前記第二プロセッサに渡すように構成されている、前記第一プロセッサおよび少なくとも1つの第二プロセッサと、通信ネットワークとを有する。当該システムは、当該第二プロセッサのそれぞれのためのタスクスケジューラを有する。前記タスクスケジューラは、当該第二プロセッサのタスクスケジューリングを制御する。

【００２１】

本発明の観点においては、前記タスクスケジューラは、プログラム可能な第二プロセッサ上に実装されている。

【００２２】

本発明のさらなる実施例は、従属請求項に記載されている。

【００２３】

本発明の上記およびその他の観点は、図面を参照しながら以下により詳細に説明されている。

【発明を実施するための最良の形態】

【００２４】

図2は、本発明の好ましい実施例による、データオブジェクトのストリームを処理する処理システムを示す。このシステムは、異なる層、すなわち計算層1と、通信サポート層2と、通信ネットワーク層3とに分割することができる。計算層1は、CPU 11と、2個のプロセッサまたはコプロセッサ12a，12bとを含む。これは単なる一例であり、明らかに、より多くのプロセッサをシステムに含めることができる。通信サポート層2は、CPU 11に関連付けられているシェル21と、コプロセッサ12a，12bにそれぞれ関連付けられているシェル22a，22bとを有する。通信ネットワーク層3は、通信ネットワーク31とメモリ32とを有する。

【００２５】

プロセッサ12a，12bは、好ましくは専用プロセッサであり、それぞれが、ストリーム処理の限られた範囲を実行するように特殊化されている。各プロセッサは、ストリームの連続するデータオブジェクトに同じ処理操作を繰り返し適用するように構成されている。プロセッサ12a，12bは、それぞれが、異なるタスクまたは機能、例えば、可変長復号化、ランレングス復号化、動き補償、イメージスケーリング、またはDCT変換の実行などを行うことができる。動作時、各プロセッサ12a，12bは、1つ以上のデータストリームに対する操作を実行する。この操作は、例えば、ストリームを受信して別のストリームを生成する、新しいストリームを生成せずにストリームを受信する、ストリームを受信せずにストリームを生成する、または受信したストリームを修正することを含むことができる。プロセッサ12a，12bは、別のプロセッサ12b，12aによって、またはCPU 11によって生成されたデ

ータストリーム、あるいは自身が生成したストリームを処理することができる。ストリームは、メモリ32を介してプロセッサ12a，12bから／に転送される一連のデータオブジェクトを有する。

【００２６】

シェル22a，22bは、通信層である通信ネットワーク層の方への第一インタフェースを有する。この層は、すべてのシェルに対して均一または包括的である。さらに、シェル22a，22bは、シェル22a，22bがそれぞれ関連付けられているプロセッサ12a，12bの方への第二インタフェースを有する。第二インタフェースは、タスクレベルのインタフェースであり、関連付けられているプロセッサ12a，12bの特定のニーズを処理することができるように、これらのプロセッサ12a，12b用にカスタマイズされている。従って、シェル22a，22bは、第二インタフェースとしてプロセッサに固有なインタフェースを持つが、システムアーキテクチャ全体の中でシェルの再利用を容易にする一方で、特定のアプリケーションのパラメータ化との適合化を行うことができるように、これらのシェルの全体的なアーキテクチャは、すべてのプロセッサに対して包括的かつ均一である。

【００２７】

シェル22a，22bは、データ転送のための読み取り／書き込み装置と、同期装置と、タスク切替え装置とを有する。これらの3つの装置は、関連付けられているプロセッサとマスター／スレーブベースで通信し、プロセッサはマスターとして機能する。従って、3つの装置それぞれは、プロセッサからの要求によって初期化される。プロセッサとこれら3つの装置の間の通信は、引数値を渡して要求値が戻されるのを待つようにするために、要求／アクノリッジハンドシェークメカニズムによって実施されることが好ましい。従って、通信はブロッキング式、すなわちそれぞれの制御スレッドがその完了を待つ。

【００２８】

読み取り／書き込み装置は、2つの異なる操作、すなわちプロセッサ12a，12bがメモリからデータオブジェクトを読み取ることができるようにする読み取り操作と、プロセッサ12a，12bがメモリ32にデータオブジェクトを書き込むことができるようにする書き込み操作とを実施することが好ましい。各タスクは、データストリームの付加ポイント(attachment point)に対応する事前定義された一連のポートを持つ。これらの操作の引数は、それぞれのポートのID「port_id」と、読み取り／書き込みが行われるオフセット位置「offset」と、データオブジェクトの可変長「n_bytes」である。ポートは、引数「port_id」によって選択される。この引数は、現在のタスクのみのローカルスコープを保持する、負でない小さな数である。

【００２９】

同期装置は、空のFIFOからの読み取り、または満杯のFIFOへの書き込み時のローカルなブロッキング条件を扱うための同期を目的とする2つの操作を実施する。第一操作、すなわちgetspace操作は、FIFOとして実装されているメモリ内のスペースの要求であり、第二操作、すなわちputspace操作は、FIFO内のスペースを解放するための要求である。これらの操作の引数は、「port_id」と可変長「n－bytes」である。

【００３０】

getspace操作とputspace操作は、リニアテープまたはFIFOの同期順に実行されるが、この操作によって取得されるウィンドウの内側では、ランダムアクセスの読み取り／書き込みアクションがサポートされる。

【００３１】

タスク切替え装置は、プロセッサのタスク切り替えをgettask操作として実施する。この操作の引数は、「blocked」、「error」、および「task_info」である。

【００３２】

引数「blocked」は、入力ポートまたは出力ポート上のgetspace呼び出しが偽を戻したために、最後の処理ステップを正常に完了することができなかった場合に設定されるブーリアンである。従って、ブロックされているポートに対して新しい「space」メッセージが到着しない限りはそのタスクを再スケジューリングしない方がよいことが、タスクスケ

ジューリング装置にただちに通知される。この引数値は、スケジューリングの向上につながるアドバイスとして考慮されるにすぎず、機能性に影響することはない。引数「error」は、最後の処理ステップの間にプロセッサの内部で致命的エラーが起きた場合に設定されるブーリアン値である。mpeg復号化の例では、例えば、未知の可変長符号または不正なモーションベクトルが現れた場合である。その場合、シェルは、タスクテーブルのイネーブルフラグをクリアしてさらなるスケジューリングを防止し、システムの状態を修復するための割り込みが主CPUに送られる。現在のタスクは、CPUがソフトウェアを通じて対話するまでは絶対にスケジューリングされない。

【００３３】

シェル22とプロセッサ12の間のタスクレベルのインタフェースに関して、シェル22とプロセッサ12の間の境界は、以下のいくつかの点を念頭に置いて引かれる。すなわち、すべてのプロセッサは、シェルのマイクロアーキテクチャを再利用することができる。また、シェルは、機能に固有な問題に関する意味論的な知識を持たない。シェルは、グローバルな通信システム上の抽象部分を形成する。（プロセッサの観点から）異なるタスクは、互いを認識しない。

【００３４】

上述されている操作は、プロセッサからのread呼び出し、write呼び出し、getspace呼び出し、putspace呼び出し、またはgettask呼び出しによって開始される。

【００３５】

図2によるシステムアーキテクチャは、マルチタスキングをサポートし、すなわち、いくつかのアプリケーションタスクを1つのプロセッサにマッピングすることができる。マルチタスキングのサポートは、一連のアプリケーションを構成設定することと、データ処理システムにおける異なる場所に同じハードウェアプロセッサを適用することに対してのアーキテクチャの柔軟性を達成するうえで重要である。明らかに、マルチタスキングでは、アプリケーションを正しく進行させるためにプロセッサがどのタスクをどの時点で実行しなければならないかを決定するプロセスとして、タスクスケジューリング装置が必要となる。好ましい実施例のデータ処理システムは、データ依存のストリームの不定期な処理と動的な作業負荷とが目標とされており、タスクスケジューリングはオフラインではなくオンラインで実行され、実際の状況を考慮することができる。タスクスケジューリングは、固定的なコンパイル時スケジュールとは対照的に実行時に実行される。

【００３６】

プロセッサ12は、タスクが実行されている間で、実行中のタスクに割り込むことができる時間的な瞬間を明示的に決定することが好ましい。このようにして、このハードウェアアーキテクチャは、任意の時点におけるコンテキストを保存するための方策を必要としない。プロセッサは、状態をほとんど、またはまったく持たない時点まで処理を続行することができる。これらの時点は、プロセッサがタスク切り替えを最も容易に実行することができる瞬間である。

【００３７】

このような瞬間において、プロセッサ12は、自身が次に処理を実行すべきタスクをシェル22に尋ねる。この問い合わせは、gettask呼び出しを通じて行われる。このような問い合わせの間の間隔は、処理ステップとみなされる。一般的には、処理ステップには、1つ以上のデータパケットを読み取ることと、取得されたデータに何らかの操作を実行することと、1つ以上のデータパケットを書き込むことが含まれる。

【００３８】

タスクスケジューリング装置は、シェル22に属しており、gettask機能を実装している。プロセッサ12は、各処理ステップの前にgettask呼び出しを実行する。戻り値は、タスクのコンテキストを識別する負でない小さな数であるタスクIDである。このように、プロセッサ12の要求時、スケジューラは、次の最適なタスクをプロセッサ12に伝える。この手順構成は、切り替えポイントがプロセッサ12によって提供されるノンプリエンプティブスケジューリングとみなすことができる。スケジューリング装置は、プロセッサ12に割り込

むことはできず、プロセッサ12が処理ステップを終了して新しいタスクを要求するのを待つ。

【００３９】

本発明によるタスクスケジューリングのアルゴリズムは、次のようなアプリケーション、すなわち、作業負荷が動的であり、一時的な過負荷状況時の挙動が予測可能であり、次のタスクが数クロックサイクル内に選択され、かつアルゴリズムが単純であって各シェルにハードウェアを高い費用効果で実装するのに適したアプリケーションに効果を発揮するはずである。

【００４０】

マルチタスキングアプリケーションは、マルチタスキングプロセッサ上に適切なタスクをインスタンス化することによって実施される。あるタスクの挙動が、同じプロセッサを共有している別のタスクの挙動に悪い影響を与えてはならない。従って、スケジューラは、割り当てられている以上のリソースを要求して別のタスクの進行を妨げるタスクを防止する。

【００４１】

典型的な場合においては、メディアデータストリームのリアルタイムスループットを可能にするために、すべてのタスクの作業負荷の合計がプロセッサの計算能力を超えないことが好ましい。データ依存挙動のタスクの場合に、最悪の条件時における一時的な過負荷状況は起きてもよい。

【００４２】

ラウンドロビン式のタスク選択は、この場合のリアルタイムパフォーマンス要件に適している。なぜなら、この方式では、処理ステップの所要時間が短い場合、各タスクが十分に高い頻度で処理されることが保証されるためである。

【００４３】

システム設計者は、構成設定時にリソース割当量を各タスクに割り当てる。タスクスケジューリング装置は、割当量を確実に保護するためにポリシー方式をサポートしている必要がある。スケジューラは、タスクの正確な実行時間に割当量を関係付けることによって、リソース割当量のポリシー化を実施する。スケジューラは、測定単位としてタイムスライス、すなわち、（代表的には処理ステップの長さのオーダーである）所定の固定数のサイクルを使用する。タスク割当量は、多数のタイムスライスとして与えられる。タスクスケジューラは、運用中の割当量を、新しく選択されたタスクの割当量に初期化する。シェルは、各タイムスライスの後に、アクティブタスクの運用中の割当量をデクリメントする。このようにして、割当量は、処理ステップの長さとは無関係であり、スケジューラは、アクティブタスクを、その割当量によって与えられるタイムスライスの数に制限する。

【００４４】

このようなタスクあたりの割当量の実施は、二重の用途がある。すなわち、プロセッサを共有するタスクの相対的な割当量の値が、タスク間での計算リソースの分割を制御し、絶対的な割当量の値が、状態の保存と回復の相対的なオーバーヘッドに影響するタスク切り替え頻度を制御する。

【００４５】

運用中の割当量は、アクティブタスクが通信をブロックすると破棄される。このブロックしているタスクがスケジューリング上の割当量に戻ると、ただちに次のタスクが始まる。このようにして、作業負荷が十分なタスクは、その割当量をより頻繁に使用することによって余分な計算時間を使用することができる。

【００４６】

プロセッサにおけるタスクの絶対的な割当量は、これらのタスクの実行時間、従ってプロセッサのタスク切り替え速度を決定する。そして、プロセッサのこのタスク切り替え速度は、タスクの全ストリームのためのバッファサイズに関連する。タスク切り替え速度が小さいことは、タスクのスリープタイムが長いことを意味し、バッファ要件が大きくなる。従って、タスク切り替え速度はかなり高いことが好ましく、従って、長いタスク切り替

え時間は容認されない。理想的には、プロセッサのタスク切り替え時間は、毎回のタスク切り替えを可能にするために、1つの処理ステップと比較して短くすべきである。これにより、最低の絶対的割当量と最小のストリームバッファとを割り当てることが可能になる。

【００４７】

本発明によるタスクは、動的な作業負荷を持つ。これらのタスクは、実行時間、ストリームの選択、および／またはパケットサイズに関して、データに依存することがある。このデータ依存性は、スケジューラの設計に影響する。なぜなら、タスクが進行できるか否かを事前に確定することができないためである。本発明による実施例として、「最良の推測」を実行するスケジューリング装置が説明されている。このタイプのスケジューラは、ほとんどの場合に正しいタスクを選択し、そうでない場合には限られたペナルティによって回復することによって、有効となり得る。スケジューラの目的は、プロセッサの稼働率を向上させて、タスクができるだけ進行できるようにスケジューリングすることである。タスクの操作がデータに依存するため、スケジューラは、選択されたタスクが処理ステップを完了できることを保証することができない。

【００４８】

タスクは、タスクに少なくともいくらかの利用可能な作業負荷がある場合に実行可能である。タスクがアクティブであるように構成設定時に設定されている場合、タスクのイネーブルフラグが設定される。スケジュールフラグも、構成設定パラメータであり、タスクが実行可能であるためにはスケジューラがストリームの利用可能なスペースを考慮する必要があるか否かをストリームごとに示す。スペースパラメータは、putspace操作を介して実行時に更新された、ストリーム内の利用可能なデータまたは空間を保持する。これに代えて、タスクの最後のgetspace問い合わせ時に不十分なスペースが存在していた場合には、ブロック状態フラグが実行時に設定される。

【００４９】

スペースが不十分であることに起因してタスクが進行できない場合には、そのタスクのストリームの1つに対するgetspace問い合わせが偽を戻したはずである。シェル22a，22bは、ストリームごとに、最後のgetspace問い合わせの否定の結果値と共にブロック状態フラグを維持する。

【００５０】

このようなブロック状態フラグが設定されると、そのタスクは以降は実行可能ではなく、タスクスケジューリング装置は、ブロック状態フラグがリセットされるまでは、以降のgettask要求においてそのタスクを再び発行することはない。このメカニズムにより、タスクスケジューリング装置は、プロセッサのストリームI／O選択またはパケットサイズがデータ依存でありスケジューラよって予測することができない場合に進行することのできるタスクを選択することができる。

【００５１】

getspace要求が失敗した後、そのアクティブタスクは、より少ない数のバイトの第二のgetspace問い合わせを発行して、それによってブロック状態フラグをリセットすることができる。ブロックされているストリーム用のスペースが外部の「putspace」によって増すと、シェルがブロック状態フラグをクリアする。

【００５２】

タスクの実行可能性は、そのタスクに対する利用可能な作業負荷に基づく。少なくとも1つの処理ステップを完了させることができるためには、タスクに関連付けられているすべてのストリームが、十分な入力データまたは出力空間を持つ必要がある。シェルは、タスクスケジューリング装置を含めて、メディアデータを解釈せず、データパケットを認識しない。データパケットのサイズは、タスクごとに可変であり、パケットサイズがデータ依存である場合がある。従って、スケジューラは、getspaceアクションの成功を保証するための十分な情報を持たない。なぜなら、そのタスクがどのストリーム上にどのくらいのスペースを要求するかが不明であるためである。

【００５３】
スケジューリング装置は、タスクの実行に対してどのくらいの量のスペースが利用可能であるかまたは必要であるかに関係なく、関連付けられているすべてのストリームに対して少なくともいくらかの利用可能な作業負荷を有するタスク（すなわちスペース > 0）を選択することによって、「最良の推測」を発行する。1つの処理ステップを完了させるのに十分な量には関係なく、いくらかのデータ、またはバッファ内の利用可能な空間があるかをチェックすることは、－消費側タスクと生産側タスクが、同じ粒度において同期する－場合には、少なくとも1つの処理ステップの実行には充分である。従って、データまたは空間が利用可能である場合、これは、少なくとも、1つの処理ステップの実行に必要なデータまたは空間の量となる。消費側タスクと生産側タスクは、操作の同じ論理単位、すなわち処理ステップの同じ粒度で動作する。例えば、バッファ内にいくらかの、しかし不十分なデータがある場合、このことは、生産側タスクが現在アクティブであることと、タスク切り替えを実行する代わりに消費側タスクが待つことができるだけ十分に早く不足データが到着することとを示している。

【００５４】
入力または出力ストリームの選択は、処理されているデータに依存することがある。このことは、タスクに関連付けられているストリームのいくつかに対してスペース = 0である場合でも、そのタスクがこれらのストリームにアクセスしなければ、そのタスクは依然として実行可能であることを意味する。従って、スケジューラは、各ストリームのスケジュールフラグを考慮する。偽のスケジュールフラグは、タスクがそのストリームにアクセスするか否かが不明であることと、スケジューラはそのストリームに対してスペース > 0の実行可能性テストを省略しなければならないこととを示す。しかしながら、そのタスクが選択されてその後にそのストリームにおける利用できないデータまたは空間によってブロック状態になる場合には、ブロック状態フラグが設定される。ブロック状態フラグを設定することにより、ブロック状態のストリームも少なくともいくらかの利用可能なスペースを持つまでは、スケジューリング装置がそのタスクを再び選択しないことが保証される。

【００５５】
スケーラブルなシステムのためには、プロセッサはできるだけ自立的である必要がある。この目的のため、非同期型かつ分散型のタスクスケジューリング装置が採用され、各プロセッサシェルは自身のタスクスケジューリング装置を持つ。プロセッサは弱く結合されており、すなわち、バッファが橋渡しすることのできるタイムスケールの中では、1つのプロセッサ上のタスクのスケジューリングが、別のプロセッサ上のタスクの瞬間的なスケジューリングと無関係である。バッファが橋渡しできる以上のタイムスケールでは、異なるプロセッサ上のタスクのスケジューリングは、共有されているバッファにおけるデータストリームの同期によって結合されている。

【００５６】
図2によるシステムアーキテクチャは、相対的に高いパフォーマンスの高データスループットのアプリケーションをサポートする。ストリームのFIFOバッファを含むオンチップメモリのサイズが限られているため、高いデータ同期速度とタスク切り替え速度とが必要である。プリエンプティブスケジューリングの割り込み駆動式のタスク切り替えを使用しない場合、十分に粒度の細かいタスク切り替えを可能にするためには、処理ステップの持続時間を短く維持する必要がある。プロセッサとシェルの間のインタフェースは、これらの要件に対応できるだけの非常に高いタスク切り替え速度を可能とし、主CPUからの介入の必要なしにローカルかつ自立的に実装することができる。gettask呼び出しは、10～1000クロックサイクルごとに1回の割合で実行されることが好ましく、これはマイクロ秒のオーダーの処理ステップの持続時間に対応する。

【００５７】
図3は、図2によるデータ処理システムに基づく、好ましい実施例によるタスクスケジューリングプロセスのフローチャートである。しかしながら、この実施例においては、シェ

ル22の中に読み取り／書き込み装置と同期装置が存在している必要はない。

【００５８】

このタスクスケジューリングプロセスは、ステップS1において、プロセッサ12aがそのシェル22a内のスケジューリング装置に送られるgettask呼び出しを実行することによって開始される。シェル22aのスケジューリング装置は、このgettask呼び出しを受け取り、タスクの選択を開始する。ステップS2において、タスクスケジューリング装置は、現在のタスクが依然として実行可能であるか、すなわち実行することができるかを確定する。タスクを実行することができるのは、入力ストリーム内にデータがあり、かつ利用可能な出力ストリーム内に空間があるときである。タスクスケジューリング装置は、現在のタスクの運用中の割当量が0より大きいかをさらに確定する。現在のタスクが実行可能であり、かつその運用中の割当量が0より大きい場合には、タスクスケジューリング装置は、ステップS3において、現在のタスクのtask_IDを、関連付けられているプロセッサ12aに戻し、プロセッサ12aが現在のタスクの処理を続行すべきことを示す。次いで、プロセッサ12aは、次のgettask呼び出しの発行まで、現在のタスクの処理を続行する。

【００５９】

しかしながら、運用中の割当量が0である場合、または、例えば、入力ストリーム内にデータの不足に起因して現在のタスクが実行可能でない場合は、フローはステップS4に飛ぶ。このステップにおいては、タスクスケジューリング装置は、プロセッサ12aによって次に処理されるべきタスクを選択しなければならない。タスクスケジューリング装置は、実行可能タスクのリストからラウンドロビンの順序で次のタスクを選択する。ステップS5において、次のタスクの運用中の割当量が、タスクテーブルからの対応する設定パラメータに設定され、ステップS6において、そのタスクのtask_IDがプロセッサ12aに戻される。次いで、プロセッサ12aは、次のgettask呼び出しの発行まで、次のタスクの処理を開始する。

【００６０】

次のタスクの実際の選択について、以下にさらに詳しく説明する。このタスク選択は、スケジューリング装置がプロセッサ12aからgettask呼び出しを受け取った時点でただちに行うか、または、スケジューリング装置がgettask呼び出しを受け取った時点で、選択の結果、すなわち次のタスクがすでに用意されているように、従ってプロセッサがgettask呼び出しの戻り値を待つ必要がないように、次のgettask呼び出しを受け取る前にスケジューリング装置が開始することができる。後者が可能であるのは、プロセッサ12aがgettask呼び出しを定期的な間隔で発行し、この間隔が処理ステップであるためである。

【００６１】

シェル22a，22bのスケジューリング装置は、ストリームテーブルとタスクテーブルとを有することが好ましい。スケジューリング装置は、関連付けられているプロセッサ12a，12bにマッピングされている異なるタスクを構成設定および管理するためにタスクテーブルを使用する。これらのローカルテーブルには、高速にアクセスすることができる。このテーブルは、各タスクの一連のフィールドを含んでいる。好ましくは、このテーブルは、タスクに関連付けられている第一ストリームへのストリームテーブル内のインデックスと、タスクが実行を許可されておりかつ必要な利用可能リソースを持つかを示すイネーブルビットと、タスクスケジューリング装置をパラメータ化するためとタスク間で処理上の公正さを確保するための割当量フィールドとを含む。

【００６２】

タスクスケジューリング装置は、ストリームテーブル内の全ストリームを1つずつ繰り返し検査して、これらが実行可能かを確定する。ストリームが実行を許可されている、すなわち実行可能であるとみなされるのは、ストリームが0でないスペースを含んでいる場合か、または、そのスケジュールフラグが設定されておらずかつブロック状態フラグが設定されていない場合である。この後、タスクスケジューリング装置は、タスクテーブル内の全タスクについて、これらが実行可能であるか1つずつ検査する。タスクが実行可能であるとみなされるのは、関連付けられているストリームすべてが実行可能であり、かつタ

スクのイネーブルフラグが設定されている場合である。タスクスケジューリング装置の次のステップは、このタスクテーブルから実行可能タスクのうちの1つを選択することであり、このタスクはプロセッサ12aによって次に処理されるべきタスクである。

【００６３】

個々のプロセスでは、シェル22a，22b内のクロック分周器によって定義される各タイムスライスごとに、運用中の割当量がデクリメントされる。

【００６４】

シェルは、タスクスケジューリング装置を専用ハードウェアの中に実装している。なぜならソフトウェア実装にはタスク切り替え速度が高すぎるためである。タスクスケジューリング装置は、gettask要求への回答を数クロックサイクル以内に供給しなければならない。

【００６５】

タスクスケジューリング装置は、新しいタスクの提案をバックグラウンドプロセスにおいて準備しておき、gettask要求が到着したときにただちにこの提案を利用可能とすることもできる。さらに、各タスクがプロセッサ上にスケジューリングされている持続時間を制御するために、タスクスケジューリング装置は「運用中の割当量」カウンタを追跡する。

【００６６】

タスク選択は、バッファの実際のステータスに関して後ろに遅れることができる。アクティブタスクのみがストリームバッファ内のスペースを減少させ、外部のすべての同期putspaceメッセージがバッファ内のスペースを増大させる。従って、実行する準備ができているタスクは、外部の同期メッセージがバッファスペースの値を更新している間、実行可能のままである。従って、スケジューラをプルメカニズム(pull mechanism)として実装することができる。この場合、スケジューラはストリームテーブル内を周期的にループし、入力される同期メッセージに関係なく各タスクの実行可能性のフラグを更新する。スケジューリングと同期とがこのように分離されていることにより、タイムクリティカル性の低いスケジューラを実装し、同時に、同期コマンドの待ち時間を最小にすることができる。

【００６７】

gettask要求は、データのブロッキングに起因して処理ステップが時期尚早に終了したときにプロセッサによって設定される「active_blocked」フラグを含むこともできる。このフラグが設定されると、アクティブタスクの「実行可能」ステータスがただちにクリアされる。この迅速なフィードバックは、スケジューラのプロセスにおける待ち時間を補正し、スケジューラは異なるタスクにただちに応答することができる。

【００６８】

本発明の好ましい実施例によるシステムアーキテクチャは、リアルタイム挙動と動的挙動とを合わせ持つ一連のメディアアプリケーション間で計算ハードウェアを再利用するための、費用効果の高いスケーラブルな解決策を提供する。各プロセッサシェル内のタスクスケジューリング装置は、利用可能な作業負荷を観察し、データ依存性の挙動を認識し、その一方で、各タスクに最小の計算割当量と最大のスリープタイムとを保証する。シェルのハードウェア実装によって、非常に高いタスク切り替え速度がサポートされる。スケジューリングは分散される。各プロセッサのタスクは、それぞれのシェルによって独立してスケジューリングされる。

【００６９】

図4は、読み取りおよび書き込み操作とそれに関連付けられている同期操作のプロセスの図解を描いている。プロセッサの観点からは、データストリームは、現在のアクセスポイントを持つ無限のデータテープのように見える。プロセッサから発行されたgetspace呼び出しは、図4aの小さな矢印によって描かれている現在のアクセスポイントより先の特定のデータスペースへのアクセスの許可を求める。この許可が認められる場合、プロセッサは、要求されたスペース、すなわち図4bにおける枠型ウィンドウの内側で、引数n_bytesによって示されている可変長データを使用して、引数offsetによって示されているランダ

ムなアクセス位置において、読み取りおよび書き込みアクションを実行することができる。

【００７０】

許可が認められない場合には、この呼び出しは偽を戻す。1つ以上のgetspace呼び出し（およびオプションとしていくつかの読み取り／書き込みアクション）の後、プロセッサは、その処理、またはデータスペースのいくらかの部分について完了することを決定し、putspace呼び出しを発行することができる。この呼び出しにより、アクセスポイントが、特定のバイト数だけ、すなわち図4dにおいてはn_bytes2だけ前に進み、このサイズは前に認められたスペースによって制約されている。

【００７１】

図4は、循環的なFIFOメモリの図解を描いている。データストリームを伝達するにはFIFOバッファが必要であり、FIFOバッファは有限かつ一定のサイズであることが好ましい。好ましくは、このバッファは、メモリ内に事前に割り当てられており、リニアメモリのアドレス範囲内で正しいFIFO挙動を得るため、循環的なアドレス指定メカニズムが適用される。

【００７２】

図4の中央における回転矢印50は、プロセッサからのgetspace呼び出しが、読み取り／書き込み用に認められたウィンドウを確認する方向を描いており、これは、putspace呼び出しによってアクセスポイントが先に移動する方向と同じである。小さな矢印51，52は、タスクAおよびBの現在のアクセスポイントを示す。この例においては、Aはライターであり、従って適切なデータを後に残すのに対し、Bはリーダーであり、空のスペース（または意味のない屑）を後に残す。各アクセスポイントより先の斜線領域（A1，B1）は、getspace操作を通じて取得されたアクセスウィンドウを示す。

【００７３】

タスクAおよびBは、マルチタスキングに起因して、異なる速度で進む、および／または、ある期間について処理されないことがある。シェル22a，22bは、AおよびBが実行されるプロセッサ12a，12bに、AおよびBのアクセスポイントがそれぞれの順序付けを確実に維持するための、より厳密には、認められるアクセスウィンドウが絶対に重ならないようにするための情報を提供する。シェル22a，22bによって提供されたこの情報を使用して、この機能全体としての正確性が達成されるようにすることは、プロセッサ12a，12bの責任である。例えば、シェル22a，22bが、プロセッサからのgetspace要求に対して、例えば、バッファ内の利用可能なスペースが不十分であるために、場合によっては偽と回答することがある。その場合、プロセッサは、拒否されたアクセス要求に従って、そのバッファへのアクセスを控えるべきである。

【００７４】

シェル22a，22bは分散されており、従って、それぞれのシェルは、自身に関連付けられているプロセッサ12a，12bの近くに実装することができる。各シェルは、そのプロセッサにマッピングされているタスクに付帯するストリームの構成設定データをローカルに含んでおり、このデータを正しく扱うための制御ロジックすべてをローカルに実装している。従って、各ストリームの、言い換えれば各アクセスポイントの一連のフィールドを含んでいるローカルストリームテーブルが、シェル22a，22bの中に実装されている。

【００７５】

図4の手順構成を処理するために、タスクAおよびBのプロセッサシェル22a，22bのストリームテーブルそれぞれは、自身のアクセスポイントからこのバッファ内の別のアクセスポイントの方向への（おそらくは悲観的な）距離を含む「space」フィールドと、このバッファ内のその別のアクセスポイントのタスクとポートを持つリモートシェルを表すIDとを保持している、1行を含んでいる。さらに、このローカルのストリームテーブルは、現在のアクセスポイントに対応するメモリアドレスと、アドレスのインクリメントをサポートするための、バッファのベースアドレスおよびバッファのサイズのコーディングを含むことができる。

【００７６】
これらのストリームテーブルは、シェル22のそれぞれの中の小さなメモリ内にマッピングされている（レジスタファイルに似た）メモリであることが好ましい。従って、getspace呼び出しに対し、要求されたサイズとローカルに格納されている利用可能なスペースとを比較することによって、ただちにかつローカルに回答することができる。putspace呼び出し時、このローカルスペースのフィールドは、指示された量だけデクリメントされ、前のアクセスポイントを保持する別のシェルには、そのスペース値をインクリメントする目的でputspaceメッセージが送られる。これに対応して、シェル22は、リモート発信元からのこのようなputメッセージを受け取った時点で、ローカルフィールドをインクリメントする。シェル間でのメッセージの転送には時間がかかるため、両方のスペースフィールドの合計が全体のバッファサイズに達する必要がなく、かつ悲観的な値を瞬間的に含む場合が起こることがある。しかしながら、このことは同期の安全性に違反しない。例外的な状況においては、複数のメッセージが現時点で送り先までの途中にあり、正しくない順序で処理されることがあるが、たとえこの場合にも、同期は正しいままである。
【００７７】
図5は、各シェル内でローカルスペース値を更新して「putspace」メッセージを送るメカニズムを示す。この配置構成においては、プロセッサ12a，12bからのgetspace要求、すなわちgetspace呼び出しに対して、関連付けられているシェル22a，22bにおいて、要求されたサイズとローカルに格納されているスペースの情報とを比較することによって、ただちにかつローカルに回答することができる。putspace呼び出し時、ローカルシェル22a，22bは、スペースのフィールドを、指示された量だけデクリメントし、putspaceメッセージをリモートシェルに送る。リモートシェル、すなわちもう1つのプロセッサのシェルは、別のアクセスポイントを保持しており、そのスペース値をインクリメントする。これに対応して、ローカルシェルは、リモートの発信元からのこのようなputspaceメッセージを受け取った時点で、そのスペースフィールドをインクリメントする。
【００７８】
アクセスポイントに属すスペースフィールドは、2つの発信元によって修正される、すなわち、ローカルなputspace呼び出し時にデクリメントされ、putspaceメッセージの受け取り時にインクリメントされる。このようなインクリメントまたはデクリメントが自動的な操作として実施されない場合、これによって誤った結果につながることがある。このような場合には、それぞれが1つの発信元のみによって更新される、個別のローカルスペースフィールドおよびリモートスペースフィールドを使用することができる。この場合には、ローカルなgetspace呼び出し時、これらの値が減じられる。シェル22は、自身のローカルテーブルの更新を常に制御しており、これらの更新を自動的に実行する。このことは、明らかにシェルの実装の問題に過ぎず、外部の機能からは見えない。
【００７９】
getspace呼び出しが偽を戻す場合には、プロセッサは、それに対する挙動を自由に決定することができる。可能な挙動は、a)プロセッサは、より小さい引数n_bytesを使用して新しいgetspace呼び出しを発行する、b)プロセッサは、しばらく待ってから、再試行する、c)プロセッサは、現在のタスクを終了して、自身の別のタスクを進行させる、である。
【００８０】
これにより、タスク切り替えの決定を、より多くのデータが到着する予測時刻と、状態を保存するためのコストをかけて内部的に蓄積される状態の量とに依存して行うことができる。プログラム可能でない専用ハードウェアプロセッサの場合には、この決定は、アーキテクチャの設計プロセスの一部である。状態の保存と回復は、タスクスケジューラではなくプロセッサの責任である。プロセッサは、状態の保存と回復を様々な方法において実装することができ、例えば、次のとおりである。
【００８１】
－　プロセッサは、自身にローカルな各タスク用の明示的な状態メモリを持つ。
【００８２】

－　プロセッサは、getspace、read、write、およびputspaceプリミティブを使用して、共有されているメモリに状態を保存および回復する。

【００８３】

－　プロセッサは、プロセッサとシェルの間のインタフェースとは別のインタフェースを介して、外部のメモリに状態を保存および回復する。

【００８４】

シェル22の実装と操作によって、読み取りポートと書き込みポートの間に違いが生じることはないが、特定の具体化によってこの違いが生じることがある。シェル22によって実施される操作は、実装上の側面、例えば、FIFOバッファのサイズ、メモリ内のその位置、メモリバウンド的な循環的FIFOのアドレスに関するラップアラウンドメカニズム、キャッシング方式、キャッシュコヒーレンシ、グローバルなI／O配列の制限、データバスの幅、メモリ配列の制約、通信ネットワークの構造、メモリ編成などを実質的に隠す。

【００８５】

シェル22a，22bは、未フォーマットのバイト列に対して動作する。データストリームを伝達するライターとリーダーによって使用される同期パケットのサイズの間に、何らの相関性は必要ない。データの内容の意味論的な解釈は、プロセッサに任される。タスクは、アプリケーショングラフに付帯する構造、例えば、自身が通信している相手のタスク、自身がマッピングされているプロセッサ、同じプロセッサにマッピングされている他のタスクなどは認識しない。

【００８６】

シェル22の高パフォーマンスの実装においては、read呼び出し、write呼び出し、getspace呼び出し、およびputspace呼び出しは、シェル22a，22bの読み取り／書き込み装置と同期装置とを介して平行に発行することができる。シェル22の異なるポートに作用する呼び出しには、相互順序の制約がないが、シェル22の同じポートに作用する呼び出しは、呼び出し側のタスクまたはプロセッサに従って順序付けされなくてはならない。このような場合、プロセッサからの次の呼び出しは、前の呼び出しが戻るときに、すなわちソフトウェア実装においては機能呼び出しから戻ることによって、ハードウェア実装においてはアクノリッジ信号を供給することによって、開始することができる。

【００８７】

read呼び出しにおけるサイズの引数、すなわちn_bytesの0値は、メモリからシェルキャッシュへの、引数port_IDとoffsetによって示される位置におけるデータのプリフェッチを実行するために、予約しておくことができる。このような操作は、シェルによって実行される自動的なプリフェッチのために使用することができる。同様に、write呼び出しにおける0値は、キャッシュフラッシュ要求用に予約しておくことができるが、自動的なキャッシュフラッシュは、シェルの責任である。

【００８８】

オプションとして、5つのすべての操作は、さらなる最後の引数task_IDを受け取る。これは、通常、それより前のgettask呼び出しからの結果の値として取得される小さな正の数である。この引数の0値は、タスクに固有ではなくプロセッサ制御に関連する呼び出し用に予約される。

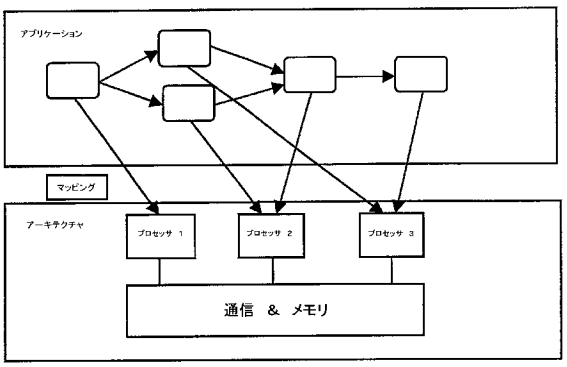
【００８９】

図2と図3による好ましい実施例に基づく別の実施例においては、機能に固有な専用プロセッサを、プログラム可能なプロセッサに置き換えて、好ましい実施例のそれ以外の特徴は同じままにすることができる。このプログラム可能なプロセッサ上に実装されているプログラムによって、各プロセッサは、ストリーム処理の限られた範囲を実行するように特殊化される。各プロセッサは、そのプログラミングによって、ストリームの連続的なデータオブジェクトに同じ処理操作を繰り返し適用するように構成されている。タスクスケジューラも、関連付けられているプロセッサ上で実行することのできるソフトウェアに実装されていることが好ましい。

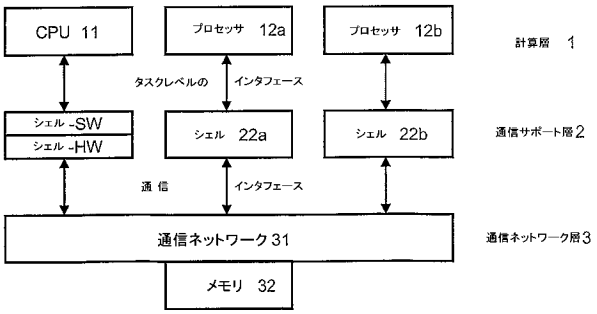
【図面の簡単な説明】

【００９０】
【図１】先行技術による、アプリケーションからプロセッサへのマッピングの図解である。
【図２】ストリームベースの処理システムのアーキテクチャの概略的なブロック図である。
【図３】好ましい実施例によるタスク切り替えプロセスのフローチャートである。
【図４】図2のシステムにおける同期操作とI／0操作の図解である。
【図５】図2による各シェルにおいてローカルスペース値を更新するメカニズムである。
【符号の説明】
【００９１】
1　計算層
2　通信サポート層
3　通信ネットワーク層
11　CPU
12a、12b　プロセッサ
21、22a、22b　シェル
31　通信ネットワーク
32　メモリ

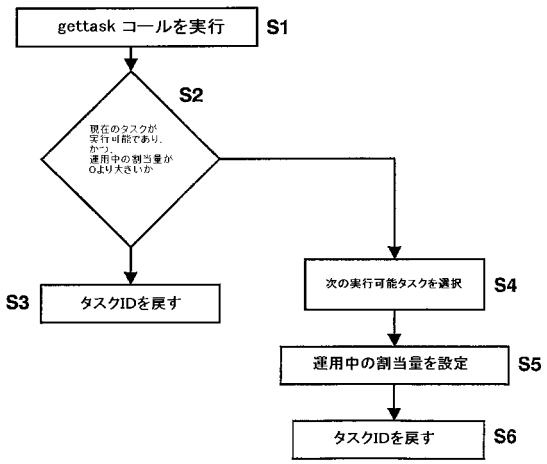
【図１】



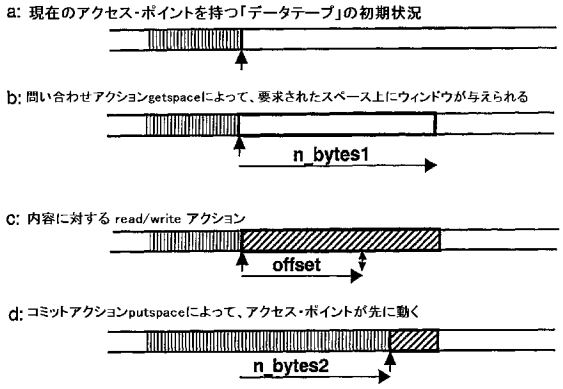
【図２】



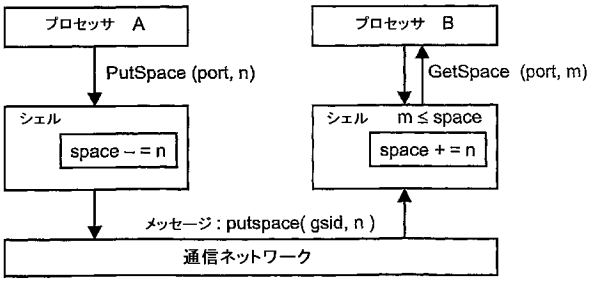
【図３】



【図４】


a: 現在のアクセス-ポイントを持つ「データテーブ」の初期状況
b: 問い合わせアクションgetspaceによって、要求されたスペース上にウィンドウが与えられる
n_bytes1
c: 内容に対する read/write アクション
offset
d: コミットアクションputspaceによって、アクセス・ポイントが先に動く
n_bytes2

【図５】


プロセッサ A
PutSpace (port, n)
シェル
space − = n
プロセッサ B
GetSpace (port, m)
シェル m ≤ space
space + = n
メッセージ : putspace(gsid, n)
通信ネットワーク

【国際調査報告】

INTERNATIONAL SEARCH REPORT

PCT/IB 02/05199

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F9/46 G06F9/48 G06F9/50

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	LIAO G ET AL: "A DYNAMICALLY SCHEDULED PARALLEL DSP ARCHITECTURE FOR STREAM FLOW PROGRAMMING" JOURNAL OF MICROCOMPUTER APPLICATIONS, LONDON, GB, vol. 17, no. 2, 1 April 1994 (1994-04-01), pages 171-196, XP000672380	1,2,4,5, 18-20, 32,34,47
A	figure 1 page 171, line 1-6 page 172, line 40 -page 173, line 30 page 177, line 1 -page 184, line 20	3,10,14, 15,26, 30,33, 39,43,44
	-/--	

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

'A' document defining the general state of the art which is not considered to be of particular relevance

'E' earlier document but published on or after the international filing date

'L' document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

'O' document referring to an oral disclosure, use, exhibition or other means

'P' document published prior to the international filing date but later than the priority date claimed

'T' later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

'X' document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

'Y' document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

'&' document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
24 February 2004	04/03/2004

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Kusnierczak, P

2

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

PCT/IB 02/05199

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	EP 0 806 730 A (SUN MICROSYSTEMS INC) 12 November 1997 (1997-11-12) claim 10 column 8, line 20 -column 9, line 13	1,2,4,5, 18-20, 32,34,47
A	RATHNAM S ET AL: "An architectural overview of the programmable multimedia processor, TM-1" DIGEST OF PAPERS OF COMPCON (COMPUTER SOCIETY CONFERENCE) 1996 TECHNOLOGIES FOR THE INFORMATION SUPERHIGHWAY. SANTA CLARA, FEB. 25 - 28, 1996, DIGEST OF PAPERS OF THE COMPUTER SOCIETY COMPUTER CONFERENCE COMPCON, LOS ALAMITOS, IEEE COMP. SOC. PRESS, , vol. CONF. 41, 25 February 1996 (1996-02-25), pages 319-326, XP010160916 ISBN: 0-8186-7414-8 abstract page 319, paragraph 1.0 -page 324, paragraph 4.6	1-5,10, 14,15, 18-20, 26,30, 32-34, 39,43, 44,47
A	COOLING J E: "TASK SCHEDULING IN HARD REAL-TIME EMBEDDED SYSTEMS USING HARDWARE CO-PROCESSORS" MICROPROCESSORS AND MICROSYSTEMS, IPC BUSINESS PRESS LTD. LONDON, GB, vol. 18, no. 10, 1 December 1994 (1994-12-01), pages 571-578, XP000488045 ISSN: 0141-9331 the whole document	1-5,10, 14,15, 18-20, 26,30, 32-34, 39,43, 44,47
A	EP 0 905 618 A (MATSUSHITA ELECTRIC IND CO LTD) 31 March 1999 (1999-03-31) column 1, line 32 -column 2, line 48 column 5, line 26 -column 6, line 21 column 6, line 50 -column 9, line 9	1-6,9, 10, 19-21, 24,26, 32-35, 38,39,47
A	DOLEV S ET AL: "Non-preemptive real-time scheduling of multimedia tasks" , COMPUTERS AND COMMUNICATIONS, 1998. ISCC '98. PROCEEDINGS. THIRD IEEE SYMPOSIUM ON ATHENS, GREECE 30 JUNE-2 JULY 1998, LOS ALAMITOS, CA, USA,IEEE COMPUT. SOC, US, PAGE(S) 652-656 XP010295203 ISBN: 0-8186-8538-7 the whole document	1,2,4,5, 11,12, 16,17, 19,27, 28,31, 32,34, 40,41, 45-47

-/--

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

PCT/IB 02/05199

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WONG C ET AL: "Task concurrency management methology to schedule the MPEG4 IM1 player on a highly parallel processor platform" PROCEEDINGS OF THE 9TH. INTERNATIONAL WORKSHOP ON HARDWARE/SOFTWARE CODESIGN. CODES 2001. COPENHAGEN, DENMARK, APRIL 25 - 27, 2001, PROCEEDINGS OF THE INTERNATIONAL WORKSHOP ON HARDWARE/SOFTWARE CODESIGN, NEW YORK, NY: ACM, US, 25 April 2001 (2001-04-25), pages 170-175, XP010543436 ISBN: 1-58113-364-2 the whole document	1,4,19, 32,34,47

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

PCT/IB 02/05199

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0806730 A	12-11-1997	US 5826081 A	20-10-1998
		EP 0806730 A2	12-11-1997
		JP 10055284 A	24-02-1998
EP 0905618 A	31-03-1999	CN 1210306 A ,B	10-03-1999
		EP 0905618 A2	31-03-1999
		JP 3007612 B2	07-02-2000
		JP 11134203 A	21-05-1999
		US 6243735 B1	05-06-2001

Form PCT/ISA/210 (patent family annex) (July 1992)

フロントページの続き

(81)指定国 AP(GH,GM,KE,LS,MW,MZ,SD,SL,SZ,TZ,UG,ZM,ZW),EA(AM,AZ,BY,KG,KZ,MD,RU,TJ,TM),EP(AT,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR,GB,GR,IE,IT,LU,MC,NL,PT,SE,SI,SK,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AT,AU,AZ,BA,BB,BG,BR,BY,BZ,CA,CH,CN,CO,CR,CU,CZ,DE,DK,DM,DZ,EC,EE,ES,FI,GB,GD,GE,GH,GM,HR,HU,ID,IL,IN,IS,JP,KE,KG,KP,KR,KZ,LC,LK,LR,LS,LT,LU,LV,MA,MD,MG,MK,MN,MW,MX,MZ,NO,NZ,OM,PH,PL,PT,RO,RU,SC,SD,SE,SG,SK,SL,TJ,TM,TN,TR,TT,TZ,UA,UG,US,UZ,VC,VN,YU,ZA,ZM,ZW

(72)発明者 ルッテン マーティーン ヨット
オランダ国 ５６５６ アー アー アインドーフェン プロフホルストラーン ６
(72)発明者 ファン アインドーフェン ヨゼフス テー ヨット
オランダ国 ５６５６ アー アー アインドーフェン プロフホルストラーン ６
(72)発明者 ポル エヴァート ヨット
オランダ国 ５６５６ アー アー アインドーフェン プロフホルストラーン ６

Ｆターム(参考) 5B045 GG02
5B098 GA04 GA05 GA08 GC01 GC16

【要約の続き】
クスケジューリングを制御する。