ストリームFIFO方式に基づくベクトル・プロセッサ『順風』の構成

7X-3

弘中哲夫　久我守弘　村上和彰　富田眞治
(九州大学大学院総合理工学研究科)

1. はじめに

我々は，シングル・ユーザ向けデスクトップ・スーパーコンピュータ『新風』DTS-edition開発の一環として，スカラ・プロセッサ『新風』に付加する形でベクトル・プロセッサ『順風』を開発中である．『順風』は我々が提案しているストリームFIFO方式[1]に基づく最初の試作機であり，ストリームFIFO方式の有効性を検証することを目的としている．本稿では『順風』プロセッサの構成について述べる．

2. 命令パイプライン構成[2]

『順風』の基本的な命令パイプライン構成としては，それ専用の命令フェッチ（IF）ステージ，命令解読（D）および命令発行（I）ステージを備えずに，実行（E）ステージのみから成る．すなわち，『順風』は『新風』の命令発行ステージからストリーム命令を受け取りベクトル演算を行なう．これにより，Crayタイプの一般のベクトル・プロセッサ同様，単一命令流で『新風』と『順風』の両プロセッサを駆動することになる（これをcoupled動作モードと呼ぶ）．

図1にEステージの構成を示す．これは，次の2つに大別される．

①ウェイティング・バッファ（WB：Waiting Buffer）
②ストリーム処理ユニット

以下，これらについて述べる．

3. ウェイティング・バッファ

『新風』から『順風』へのインターフェースであり，『新風』のEステージに設けられたWRB（Waiting and Reorder Buffer）のWB側に相当する．『新風』のIステージでは命令の依存関係の解析を行ない，スカラ命令であれば『新風』のWRBに対し命令発行を行なう．一方，ストリーム命令であれば『順風』のWBに対して命令を発行する．本WBは，『新風』のWB同様，毎サイクルごとに命令を調べて発火可能な命令が存在するか否か判断する．もし，存在すればその命令を（後述する）仮想パイプラインにディスパッチする．この時の発火条件は，『新風』のそれとは異なり[3]，以下のようになる．

①制御依存関係が解決済みである；

『新風』では制御依存関係が未解決でも発火可能だが，『順風』では解決されてなければならない．すなわち，分岐命令に後続するストリーム命令は，その実行が真に必要であると判明するまで発火されない．これは，スカラ命令と異なり，ストリーム命令の実行結果は直にFIFOレジスタに格納され，分岐命令後の復元処理が困難であるからである．

②スカラ・レジスタに起因するデータ依存関係が解決済みである；

ソース・レジスタとしてスカラ・レジスタが指定されている場合，ソース・オペランドのスカラ・データが揃っていなければならない．つまり，データ依存関係が解決されている必要がある．これは，スカラ・レジスタおよびFIFOレジスタが『新風』および『順風』双方のプロセッサからアクセス可能となっていることによる．『新風』においては命令のout-of-order実行を行なっているため，『順風』が単純にスカラ・レジスタを読出しに行っただけでは，求めるバージョンの値が得られるとは限らない．したがって，書込み予約されているスカラ・レジスタがソース・レジスタの場合，『新風』から求める演算結果がコミットされるのを待つ必要がある．

一方，FIFOレジスタのみ指定されている，つまりベクトル・データのみの場合はこの限りでない．これは，ベクトル・データに関しては，次に述べるストリーム制御ステーションにおいてデータ依存関係の解決を待つためである．

③空きの仮想パイプラインが存在する．

4. ストリーム処理ユニット

4. 1 仮想パイプライン

ストリームFIFO方式は，動的命令流中の個々の命令をノード，そのオペランドであるFIFOレジスタを枝とするデータフロー・グラフを処理の対象とする．このグラフが大きければ大きいほど処理効率が良い．しかし，命令をそのまま演算パイプラインにディスパッチすると，パイプライン本数でグラフの大きさが限定されてしまう．そこで，少ないハードウェア資源でより大きなグラフを処理するために，演算およびロード／ストア・パイプラインの仮想化を図り，“仮想パイプライン”という概念を導入する．したがって，ストリーム命令のディスパッチ対象は実パイプラインではなく，仮想パイプラインとなる．この仮想パイプラインの実体は，実パイプライン，および，ディスパッチされたストリーム命令の実行環境を保持しつつ演算発火制御を行なうストリーム制御ステーション（SS：Streaming Station）である．1本の実パイプラインは複数の仮想パイプラインにより共有され，時分割使用される．仮想パイプラインに

【dʒúmpu:】: A Vector Processor Based on Stream FIFO Architecture
Tetsuo HIRONAKA, Morihiro KUGA, Kazuaki MURAKAMI, and Shinji TOMITA
Kyushu University

対する実パイプラインの割付け／解放は，全SSが協調して動的に行う．

仮想パイプラインの導入により，以下の効果が得られる．

①演算パイプラインの使用効率向上：

イタレーション間に依存関係がある場合，一般に演算パイプラインに遊びが生じ使用効率が低下する．しかし，仮想演算パイプラインの導入により，実演算パイプラインが1個の命令に排他的に独占されることがなくなり，使用効率の向上が図れる．

②マスク付き演算と収集操作（compress）の複合化：

マスクFIFOレジスタを用いて，一般のベクトル・プロセッサと同様のマスク付き演算を行う．このとき，さらに収集操作を行なうことも可能である．これは，SSがマスク・データを先読みして，マスクをかけられた（演算が不要な）ベクトル要素に対しては演算を発火しないように制御することで実現する．従来方式では収集→演算→拡散の3命令を要する条件文の処理が，本方式ではマスク付き演算（＋収集）→拡散の2命令で済む．

4. 2構成要素

(1) ストリーム・ロード／ストア・ユニット（SLSU：Stream Load／Store Unit）

ストリーム・ロード／ストア命令に従い，メモリ上の指定されたアドレスから／へ，ベクトル・ストライド・レジスタ（VSR：Vector Stride Register）に指定された増分に従って，ベクトル・レングス・レジスタ（VLR）で指定された回数だけ指定のFIFOレジスタへ／から，ベクトル・データをロード／ストアする．本ユニットでは16本の仮想ロード／ストア・パイプラインを提供する．なお，実ロード／ストア・パイプラインの実装本数は4本を予定している．

(2) ストリーム演算ユニット（SFU：Stream Functional Unit）

ストリーム演算命令に従って，VLRで指定された回数だけ指定のFIFOレジスタをオペランドとした演算を繰り返す．本ユニットでは16本の仮想演算パイプラインを提供する．なお，実演算パイプラインにはTI社のSN74ACT8847を用い，実装本数は4本を予定している．

(3) FIFOレジスタ・ファイル

16本のFIFOレジスタを備える．各FIFOレジスタは，クロスバー・スイッチにより実演算パイプラインおよび実ロード／ストア・パイプラインに相互接続される．スイッチの設定は，全SSが協調して動的に行う．

5. おわりに

以上，筆者らが考案したストリームFIFO方式に基づく試作ベクトル・プロセッサ『順風』について述べた．現在，『順風』プロセッサの開発と並行してソフトウェア・シミュレーションを行ない，実装予定の実パイプライン本数について検証を行なっている．

参考文献

〔1〕弘中ほか：ストリームFIFO方式の構想－SIMP方式のベクトル処理への適応－，情処38全大論文集，5T-10（1989年3月）

〔2〕弘中ほか：ストリームFIFO方式に基づくベクトル・プロセッサ『順風』，信学技報（CPSY）「並列処理に関する『指宿』ミニシンポジウム」（1989年8月）

〔3〕久我ほか：SIMP（単一命令流／多重命令パイプライン）方式に基づく『新風』プロセッサの低レベル並列処理アルゴリズム，情処「並列処理シンポジウムJSPP'89」論文集，pp.163-170（1989年2月）

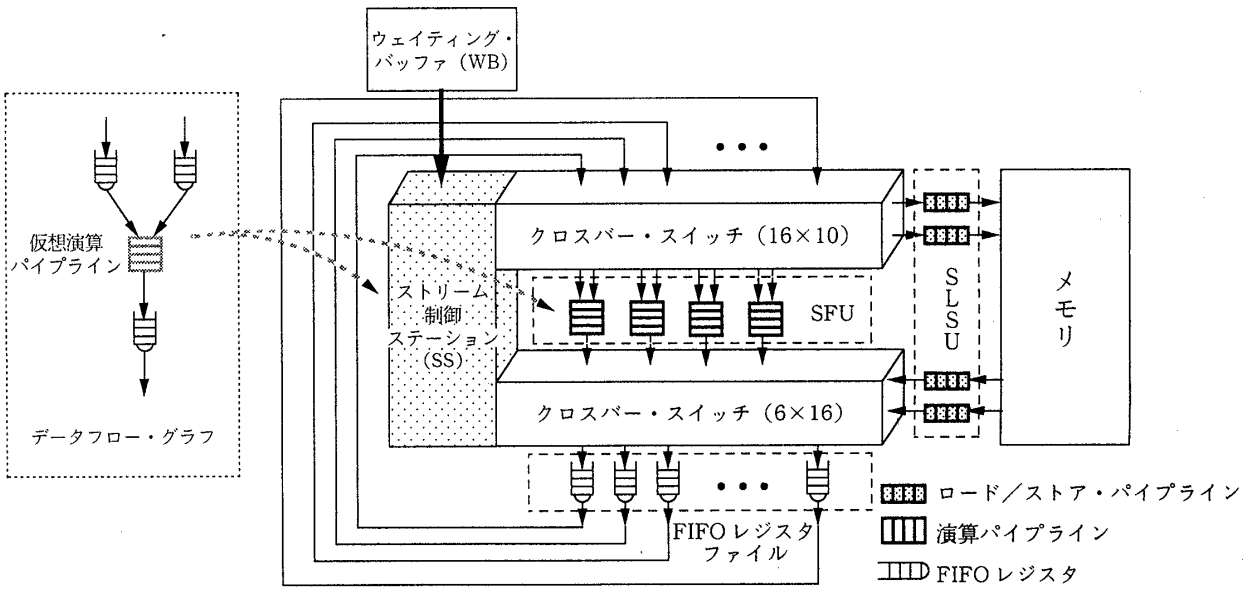

図1 Eステージの構成