CD-ROM 関連情報

—200MHz動作のSH-4整数演算ブロックを最適化

データパス回路設計の勘どころ

阪口幸雄

高性能なLSIを開発する際に，一筋縄でいかないのがデータパス回路の設計である．制御系のランダム論理の場合と異なり，論理合成ツールまかせにできないことが多い．設計者はアーキテクチャ設計，論理設計，トランジスタ・レベルの回路設計，レイアウト設計など，広範囲の知識を駆使して，パズルを解くように回路を組み上げていく．ここでは，日立製作所のマイクロプロセッサ「SH-4」の設計に関与した筆者が，データパス回路の設計手法について解説する．高性能のデータパス回路の設計では，トライ・アンド・エラーの回数を減らすことは難しい．そこで，設計した回路の性能をできるだけ短時間に見積もれる環境を用意することが重要になる．（編集部）

1. データパス回路を取り巻く状況

近年，LSIの高性能化の要求は非常に厳しくなってきている．とくにマルチメディア関連のLSIでは，音声や動画などの大容量データをリアルタイムに処理し，結果を遅れることなく出力することが求められる．こうした用途で，各種のディジタル信号処理プロセッサ（DSP）やメディア・プロセッサ，マルチメディア命令を追加したマイクロプロセッサなどが，半導体メーカから続々発売されている．こうしたマルチメディア応用の拡大にともなって，チップのなかのデータパス回路の割合が増えている．

●データパス回路とは？

データパス回路とは，あるデータ幅（32ビットや64ビットなど）で演算処理を行う回路の総称である．

標準的なデータパス回路としては，ALU（arithmetic logic unit），レジスタ・ファイル，シフタ，乗算器，浮動小数点演算器などがある．マイクロプロセッサやDSPの構成要素としては，制御回路，入出力回路，メモリと並んでもっとも重要な機能ブロックである．

マイクロプロセッサやDSPでは，ロード/ストア命令を通じてメモリからレジスタ・ファイルにデータを取り込む．このデータをレジスタ・ファイルから読み込み，演算処理（論理演算，加減乗除の算術演算，シフト演算など）が行われ，結果が再度レジスタ・ファイルに書き込まれる．

こうした演算処理は，実際にはパイプライン処理されることが多い．つまり，演算回路のなかにいくつかのフリップフロップ（FF）が挿入され，フリップフロップではさまれた処理は1クロック・サイクルで完了することが要求される．また，パイプラインが乱れたときの処理なども必要になる．

スーパ・スカラ・アーキテクチャのマイクロプロセッサでは演算ユニットが複数用意されており，同時に演算を行う．しかし，リソースを共有しているため，競合が起こったときの処理は複雑になりがちである．

●データパス設計がボトルネック

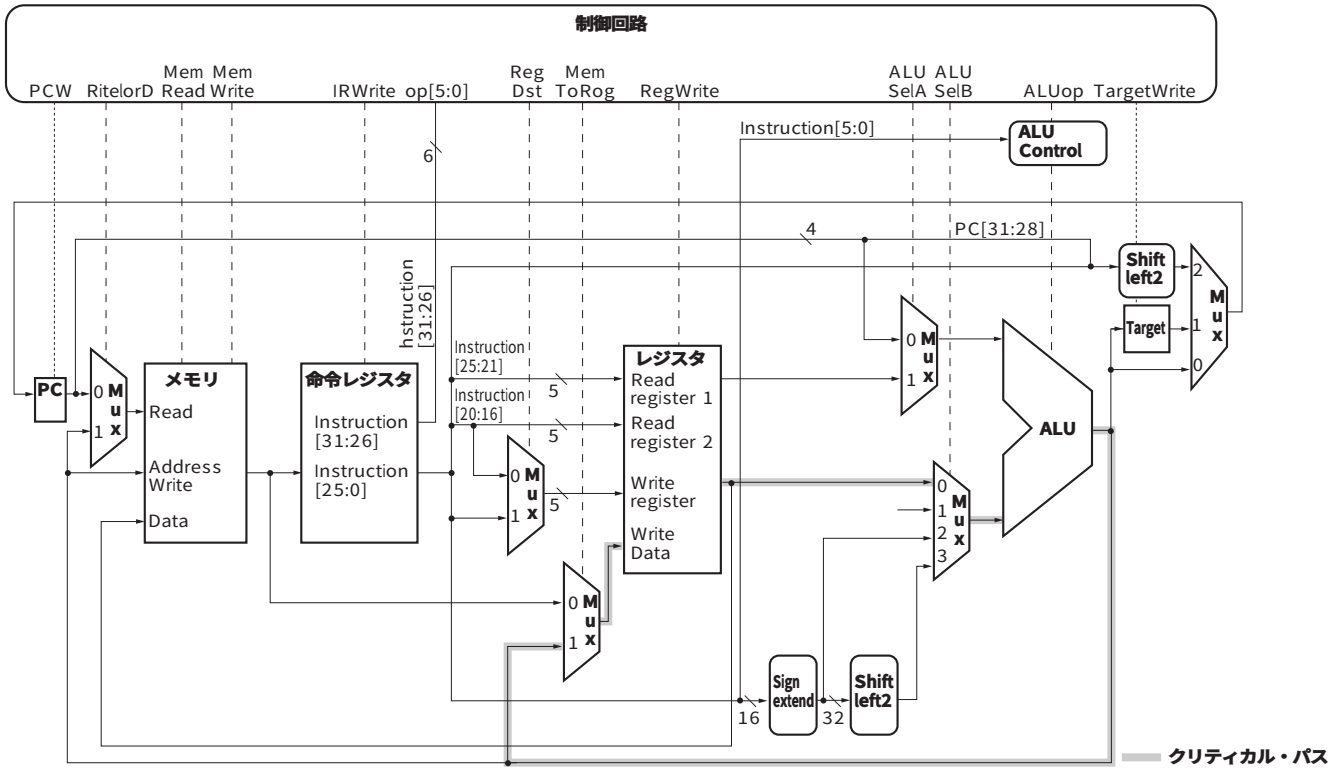
最近になって，上記のようなデータパス回路の設計がLSI開発のボトルネックになってきている．これは，おもに以下の四つの理由による．

(1) LSIの性能はデータパス回路の性能で決まることが多い．
(2) 古い設計ではデータパス回路が人手で設計されている場合が多いため，新しいプロセスへのポーティング（リターゲット）に工数がかかる．
(3) チップ全体に占めるデータパス回路の面積が増大している．
(4) 高性能のデータパス回路を設計するには，アーキテクチャ設計，論理設計，トランジスタ・レベルの回路設計，レイアウト設計，プロセス技術などの総合的な知識が必要になる．

このうち，(1)のような問題が発生するのは，データパス回路の以下のような性質に起因する．

- パイプラインの段数を増やすことが難しい．
- 一つのパイプにたくさんの論理を詰め込まなければならない．
- 年々，データ幅のビット数が増える傾向にあるが（最近では32ビットから64ビットへ移行している），パイプラインの段数は抑えなければならない．
- データをレジスタから読み込んで演算処理した後，レジスタに1サイクルで書き込まなければならない場合が多い．

これを，MIPSアーキテクチャを例に説明してみたい．**図1**にMIPS R3000の標準的なデータパス・ブロックを示した．**図1**のなかで，タイミングがもっとも厳しいのは，レジスタ・ファイルからデータを取り込み，ALUで演算処理した後，レジスタ・ファイルに書き込むところである．多くの場合，演算処理の後でCC（condition code；条件コード）を生成す



〔図1〕データパス回路の例(MIPS R3000)

MIPSアーキテクチャのデータパス回路のブロック図．32ビット幅のデータおよび命令が，このデータパス回路で処理される．レジスタ・ファイルから読み込んだデータをALUで処理し，再度，レジスタ・ファイルに書き込むところがクリティカル・パスとなる．この図は，“Computer Organization & Design(David Patterson, John Hennessy)”に掲載されているR3000のブロック図を参考にして作成した．

る必要があり，このための時間も考慮に入れておかなければならない．また，ALUに次のアドレスを計算をさせる場合も，同じようにタイミングが厳しくなる．

このとき，後述するような設計上の工夫を行って，性能を引き上げる必要がある．

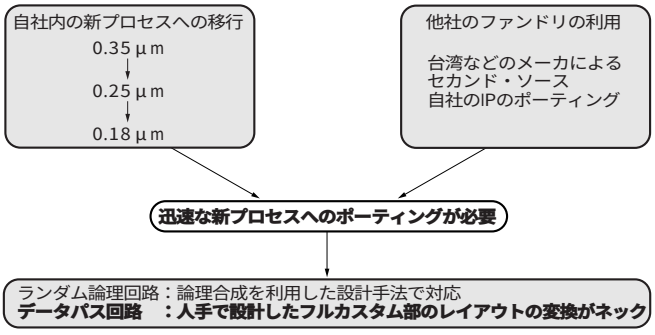
●プロセス移植性が課題に

次に，プロセス技術のポータビリティの重要性について，考えてみたい(**図2**)．近年のプロセス技術の移行スピードは驚くほど速い．極端な場合，毎年のように0.5μmから0.35μm，0.25μm，0.18μmへと，新しいプロセスに移行している．これは設計者にとって，たいへんなことである．また，LSI開発の国際分業が行われるようになり，日本の半導体メーカが自社の製造ラインだけでなく，台湾などのファウンドリを使うケースも増えている．

このとき，ランダム論理については論理合成技術を利用した設計手法が普及しているため，比較的容易に新しいプロセスに移行できる．しかし，データパス回路の場合，人手で設計したフルカスタムのレイアウトの変換がネックになる．このため，ポータビリティを考慮したデータパス回路の設計手法が必要となってきている．

〔図2〕プロセス・ポタビリティが重要に

LSIのポータビリティが，近年，たいへん重要になってきている．このとき，データパス回路を新しいプロセス技術へ迅速にポーティングすることが，必要になる．



2. データパス回路設計手法の変遷

さて，ここで，データパス回路ブロックの設計手法の変遷を振り返ってみたい(**図3**)．歴史的にみると，三つのフェーズがあった．

●第1期：ポリゴン・レベルで設計(1970年代～1980年代前半)

設計ルールでは5μm～1μmの時代である．LSI設計の黎明期にあたるこの時期，データパス回路はすべて人手で設計されていた．設計手順としては，まず論理設計者が論理機能を考える．その後，回路設計者がすべての回路をトランジス