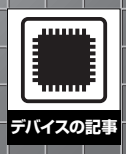
複数のプロセッサ・コアを搭載するシステムLSIを開発するための基礎知識

――ハード，ソフト，デバッグの包括的な理解が必要



Paul Kimelman

今日の組み込み機器は，ほんの数年前と比べて，はるかに厳しい要求を市場から突き付けられています．組み込み機器は，新しい特徴や機能の追加，変化し続ける標準規格や市場要求への対応，低消費電力化，製造コストの低減といった要求にこたえながら発展しています．その結果，さまざまな市場において，組み込みソフトウェアに対する需要が高まっています．また，ハイエンドのFPGA(field programmable gate array)が使用されるケースも増えています．こうした背景から，設計者の関心は複数のプロセッサ・コアを内蔵するASIC(application specific integrated circuit)やシステムLSI，あるいはまったく新しいLSI技術へ向かっています．（筆者）

システムLSIの技術を採用すれば，集積度が上がり，特定用途向けに最適化でき，消費電力も抑えられます．1チップ化することによって物理的なスペースが小さくなるため，最終製品を小型化できます．チップを製造するために顧客が独自に半導体製造工場を持つ必要はなく，（複数チップの場合と比べて）製造コストも抑えられます．

●複数プロセッサ・コア搭載へシフトするシステムLSI

システムLSIのおかげで，より高度な機能の統合を実現できるようになりました．しかしその一方で，コストを引き下げるために，システムLSIはプログラム可能な要素（プロセッサ・コアなど）を多く取り込むようになってきています．かつて，システムLSIの多くは「非常に特殊なASIC」でしたが，現在では，回路ブロックを寄せ集めてLSIを組み上げる「ビルディング・ブロック」が，ごくあたりまえの開発手法になっています．ビルディング・ブロック型の開発が増えている理由の一つとして，GSM（Global System for Mobile Communications），MP3，DivX/MPEG-4，DSL/ケーブル・モデム，ワイヤレスLANなどの標準規格や標準プロトコルが台頭してきたことが挙げられます．これらの機能がシステムLSIの中の大きな比重を占めるようになり，かつそれらの規格やプロトコルは猛烈な速さで変化しています．

ビルディング・ブロックが増えてくると，ソフトウェアに対する要求が大きくなり，その結果，ハードウェアとソフトウェアの間の連携動作（interaction）が増えることになります．このような連携動作が増えるにつれて，システムLSIの機能の統合に要する期間が長くなり，その結果，アプリケーションの多くの部分がハードウェアに依存するようになります．

ビルディング・ブロック型の設計が多くのシステムLSIに採用されるようになると，さらに用途を限定したプロセッサが使われるようになります．高速な汎用CPUを組み込めば，たいていの問題は解決できますが，部品コストや消費電力の面から考えると，この方法は効率が良いとは言えません．一方，連続的なストリーム・データの処理にはDSP（ディジタル信号処理プロセッサ）が用いられています．たとえローエンドのものであっても，DSPは音声や映像，モデム，ネットワークなどの標準的なストリーミング・プロトコルを，ほとんど電力を消費せずに，わずかなコストで処理します．一方，汎用CPUの場合，これと同等の処理を実行するために，はるかに多くの消費電力とコストを必要とする傾向にあります（ただし，プログラミングは汎用CPUのほうが容易）．

DSPの性能は専用の命令セットやメモリ/バス・アーキテクチャによって引き出されています．そのため，高級言語（C/C++やJavaなど）で適切なプログラムを作成することは容易ではありません．多くの部分はDSP固有のアセン

ブリ言語を用いて開発されています．DSPのソフトウェア開発では，アーキテクチャごとに専用の言語とモデルを使用します．したがって，あるDSPから別のDSPへコードを移植することは，一般には困難です．つまり，ここにソフトウェア開発のボトルネックが存在します．

このような問題に対する解決策の一つは，ソフトウェアを二つに分割し，高い性能が必要な部分だけをDSPで実行し，残りを汎用CPUで実行することです（**図1**）．そのため，システムLSIは複数のプロセッサ・コアを搭載することになります（以下，複数のプロセッサ・コアのことを「マルチコア」と呼ぶことにする）．

●「DSPコア＋CPUコア」の構成が採用されるわけ

マルチコアを採用している典型的な設計では，アプリケーションの処理は主に「制御＋ユーザ・インターフェース処理」と「連続処理」に分割されます．制御＋ユーザ・インターフェース処理は，汎用CPUコア（例えばARMコアなど）上で実行されます．一方，連続処理はDSPコア（または2番目のARMコア）上で実行されます（**図2**）．

このような方式を採用する理由として，以下のようなことが挙げられます．

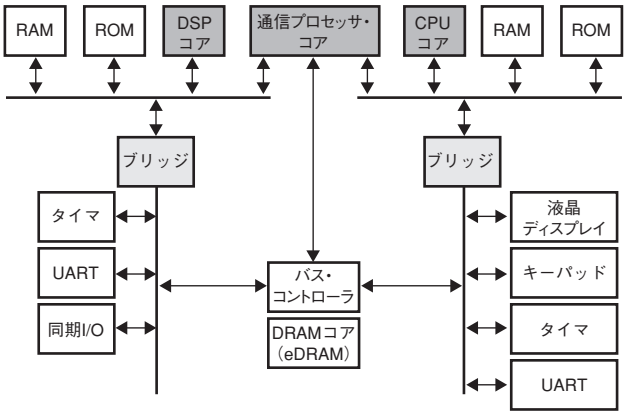
- 高級言語を用いて汎用CPUのプログラミングを行うことができる．
- 汎用CPU上でOSまたはリアルタイムOSを稼働させることができる．これによって，プロセッサの長所（合理的な割り込み，大容量メモリ，あるいは仮想メモリや保護メモリ，コンテキスト・スイッチなど）を利用できる．
- DSPに連続処理を実行させることにより，すべての処理を汎用CPUで行う場合と比べて，消費電力を引き下げることができる（割り込みのオーバヘッドが生じない）．
- 汎用CPUとDSPは互いに独立に稼働させられる．長時間処理を行わないときはそれぞれをスリープ状態にできるので，消費電力を抑えられる．さらに，汎用CPUとDSPを異なるクロック速度で動かすこともできる．そのため，各プロセッサは，それぞれの要件を満たす最低の速度で動作すればよい．これにより，廉価で消費電力の小さいチップを実現できる．
- 二つのプロセッサに対して，それぞれ異なる種類のバス・アーキテクチャを適用できる．例えば，内部RAMや内部ROM（フラッシュ・メモリ），主要な周辺回路にアクセスするためにDSPに密結合のバスを使用すると，データ・フロー処理の機能が強化される．一方，システムLSIのCPUまわりのバスにはAMBAバスがよく利用される．
- 二つのプロセッサの間の通信には，共有メモリや割り込み，メールボックス，シリアル・ポート，FIFOメモリを利用できる．

マルチコア方式の主な欠点は，ソフトウェアどうしが連携動作する部分の設計や最終的なアプリケーションのデバッグが複雑になってしまうことです．

●システムLSI開発の三つの領域

設計に取りかかる際にはすべての要件をどのようにして満足するかを決定し，（できれば）将来製造される改良版の製品も考慮して計画を立てる必要があります．

この過程では，以下のような三つの主要な領域に取り組む必要があります．



〔図1〕システムLSIの例

DSPコアでは画像処理を行い，CPUコアでは液晶ディスプレイへの表示やキーパッドからの入力を受け付ける．

〔図2〕汎用CPUとDSPの役割分担

汎用CPUは「制御＋ユーザ・インターフェース」の処理を，DSPは連続処理を担当している．