特集1 動かないLSIを動かす方法

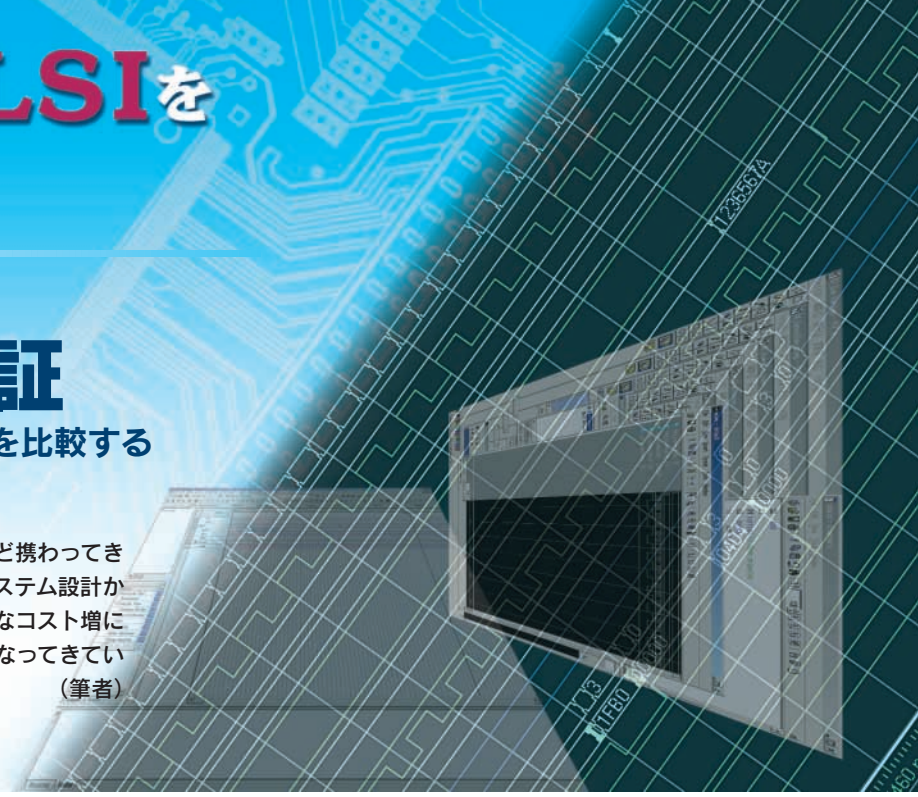
第1章

LSIの設計と検証

―― FPGAとASICの設計ルーチンを比較する

浅田朋範

ここでは，ASIC/FPGA設計に関する業務に約10年ほど携わってきた筆者が，みずからの経験をもとに，LSI設計手法とシステム設計から実機テストまでの作業工程について解説します．膨大なコスト増につながる作り直しを避けるために，「検証工程」が重要になってきています．（筆者）



システムに使用するFPGAの割合がかなり高くなってきました．FPGAに実装する設計では，ASICのときと比べて，設計ルーチンや，スケジューリング方法が変わってきていると思います（図1）．

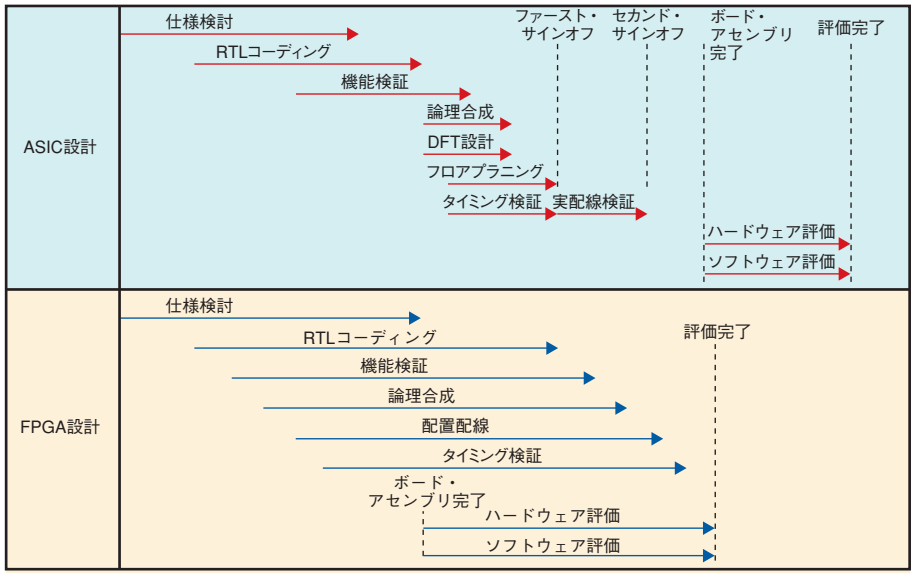
効率的なLSI設計/検証手法

まず，LSIの設計フローに沿って，ASIC設計とFPGA設計の違いを解説します．FPGAの特徴を生かすことで，効率的に設計できるようになります．しかし効率が上がる反面，注意しなければならない点もあります．

●仕様検討

ASICで設計するかぎり，必ずマスクを作成しなくてはなりません．マスク作成には多くの費用と時間がかかります．そのため，作り直しを避ける設計，すなわち質の高い仕様書の作成が求められます．仕様書を早く完成させないと，コーディングを開始する時期が遅れ，後工程のすべてに影響が及ぶことになります．

それではマスクを作成する必要がない，すなわち再コンフィグレーションが可能なFPGAの場合，仕様の決定をどの段階まで送らせることができるのでしょうか．もちろん理想的なルーチンとしては，ASICと同様にコーディング



〔図1〕
ASICとFPGAの設計ルーチンの比較
FPGAに実装する設計では，ASICのときと比べて，設計ルーチンやスケジューリングの方法が異なっている．

を始める前までに決定していれば問題ありません．一方，最悪のケースを考えた場合，ハードウェアの評価を始める前までに決定すればよいことになります．

仕様の検討とコーディングを並行して進められるというのは，FPGAを使う最大の利点の一つです．厳しい市場競争に勝つためには，このメリットを有効に使うべきだと思います．ただし，仕様書の作り込みがASICほどシビアではないという甘えから，仕様の質や完成度が下がる可能性があります．そのため，細かいフェーズでレビューを行うなど，ASICとは違った取り組みを行う必要があります．

●RTLコーディング

同じ機能，規模の設計を行うのであれば，基本的にはASICもFPGAもコーディングに必要となる時間に差はありません．開発期間に差が出てくる要素としては，デバイス固有の性質やルールに合わせてコーディングしなくてはならない点と，設計条件に合ったIPコアを利用できるかどうかといった点が挙げられます．

FPGA向けの設計で仕様書の作り込みが一部遅れる場合は，それに合わせてコーディングを行う必要があります．そのため，ASICに比べて開発期間を長くとる必要があります．また，仕様書に対するコーディングの進捗状況をこまめに管理する必要があります．

●機能検証

ASICの場合，設計のやり直しを避けるためにすべてのケースをシミュレーションする必要があります．またシミュレーションで使用したパターンを出荷テスト時の印加パターンとしても使用するので，活性化率の高いシミュレーションが要求されます．

しかし，現実には膨大なゲート規模の回路のすべての状態をカバーするシミュレーション・パターンを作り出すことは不可能に近くなっています．また，画像処理装置のように大量のデータを処理/制御する回路の場合，すべての状態をシミュレーションすること自体が非現実的です．

良いのか悪いのか，FPGAの場合にはボード評価の段階でバグが見つかっても，再コンフィグレーションにより修正できてしまいます．シミュレーションに漏れがあれば手戻りにはなりますが，開発期間にそれほど影響を与えずに処置できることもあります．

筆者のしごとは現在FPGA設計が主体ですが，ASIC設計を行っていたころと比べると，シミュレーションにかける時間が減っていると感じます．シミュレーションであるレベルの動作を確認できたら，あとは実機による検証に入っています．同様な状況にある読者の方もいらっしゃるのではないでしょうか．

ところで，シミュレーションと実機による検証をうまく使い分けているうちはよいのですが，シミュレーションをほとんど行わずにボード評価に挑むという，無謀なルーチンが確立されてしまうのでは問題です．これでまともに動くLSIが作れるとは思えません．ASICであろうとFPGAであろうと，シミュレーションは項目を明確にして進めるべきです(図2)．

●論理合成，DFT設計

論理合成ツールが優秀になるにしたがって，従来と比べて論理合成にかける時間が大幅に減ってきています．

例として論理合成によって所望のタイミングが得られなかったときのことを考えてみましょう．従来は設計者が自前のノウハウを生かしてRTLの段階でタイミングを調整していました．しかし，最近は論理合成ツールの能力でカバーできてしまい，速度優先に設定するだけで解決するケースが増えています．タイミングの調整のために，長い時間をかける必要がなくなってきました．

ASICの場合に限り，スキャン・セルの挿入など，DFT(design for testability)設計のための工数が必要になりま

〔図2〕機能検証