短期開発に対応したＬＳＩ設計環境の構築

Creation of an LSI Design Environment that Accommodates Short-Range Development

井　原　　渉　*Wataru Ihara*
伏　見　文　孝　*Fumitaka Fushimi*
伊　藤　隆　志　*Takashi Itoh*
前　田　恵　一　*Keiichi Maeda*

要　旨

LSIの回路規模は，プロセスの微細化に伴い，増加の一途を辿っている。一方で，市場の要求により，早期投入が重要度を増してきており，さらなる設計期間の短縮を求められている。

大規模LSIを効率よく，高品質に設計するには，多くのツールを活用する必要があり，また，いかにコストをかけずに設計生産性を上げるかも重要である。LSIを早期に製品化し市場投入するには，LSIの評価を効率化することも欠かせない。

本稿では，短期開発を実現するために当社で構築している設計環境，工夫点を中心に紹介する。

Abstract

As process miniaturization advances, the large-scale integration (LSI) of circuits continues to proceed. Moreover, market demand is increasing the importance of early introduction, and further shortening of the design period is being sought.

Designing large-scale integrated circuits efficiently and with high quality requires the use of many tools. It is also important to raise design productivity without increasing costs. In order to quickly commercialize and introduce LSI circuits to the market, more efficient LSI evaluation is also indispensable.

The main intent of this report is to introduce the features of design environments created by our company to realize short-range development.

1 はじめに

当社のIC開発は，モートロニクスで，1973年（昭和48年）にシートベルト装着強制装置用でスタートした。当時は，100素子にも満たない小さなカスタムICだった。その後，主力製品であるエミッションコントロール用を中心として開発を進め，1988年（昭和63年）には，IC設計専任の課が設置され，バイポーラLSI開発の設計環境が整備された。

一方，AVCでは，1980年(昭和55年)のチューナマイコンでスタートした。1988年(昭和63年)のDSP用チップでは，配線幅が1.2μmで，D/Aコンバータ外付でも，チップサイズが10×12mmと大きなものであった。

その後，モートロニクスとAVCのIC開発部門が集約されLSI開発部が誕生し，現在に至っている。

その間，プロセス技術は，3年で4倍というムーアの法則に則して微細化が進展し，現在では0.1μmによる設計が目前となっている。当然，回路規模は増大し，世間では1000万ゲートを超えるチップ開発も珍しくなくなっている。

しかし，LSIの大規模化にもかかわらず，市場はマーケットインが重要なため，ますます開発期間の短縮が必要になっている。

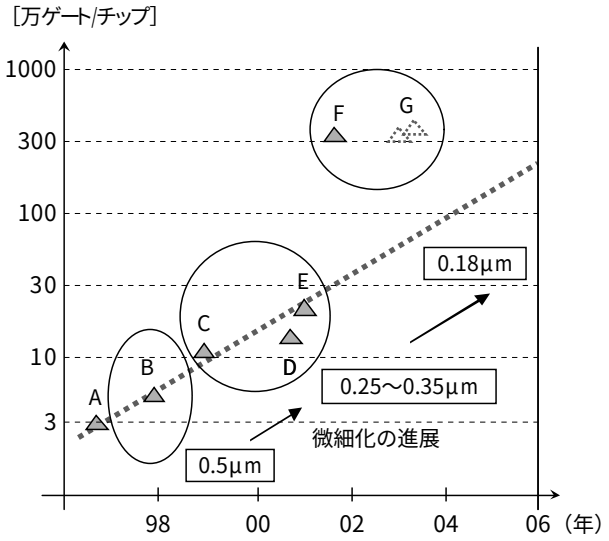

図-1　当社開発LSI（ディジタル）
Fig.1 Advances in Fujitsu Ten's LSI (digital)circuit developments

そこで，本論では，LSIの設計環境の視点から，ディジタル設計，アナログ設計，ディジアナ混載設計の変遷が，開発のスピードアップにどのように影響してきたかを論じる。加えて，設計の効率化への取り組み事例を他社とのベンチマークも含め，紹介する。

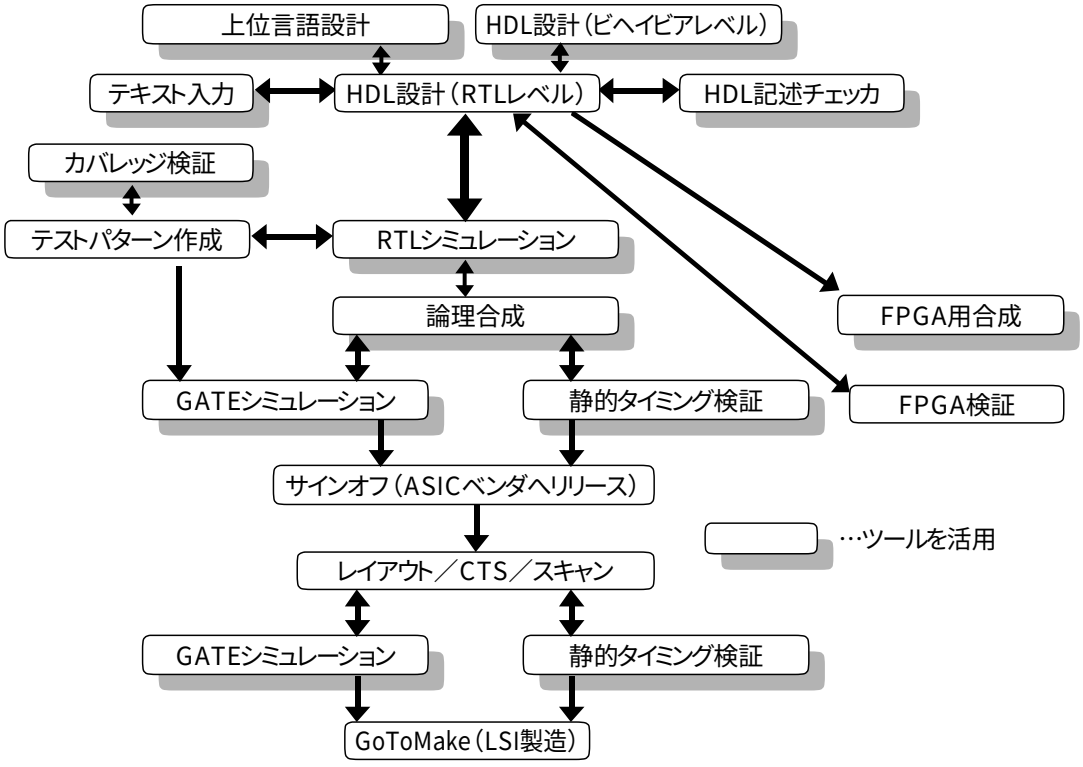

図-2　ディジタルLSI設計フロー
Fig.2 Digital LSI design process

2.1 ディジタルLSIの設計環境

ディジタル用LSIでは，高集積化の進歩が早く，それに伴い，設計規模も増大しているが，設計期間は従来と同じか，それ以下を要求されている。

この要求をクリアするために，短期開発を可能とする設計手法やツールが進歩している。

当社では現在，図2のような環境を構築している。

1）ハードウェア記述言語による設計

ディジタル設計では，ハードウェア記述言語(HDL：Hardware Description Language)を利用した設計手法を活用している。

RTL(Register Transfer Level)レベルのHDLは，回路の動作をクロックの概念が入った記述で表現するもので，古典的な回路図設計(GATE回路)に比べ，抽象度が格段に上がり，シミュレーションも高速なため，設計生産性を10倍以上向上させることができる。

当社では，1995年から標準化(IEEE1364)されたVerilog-HDLを設計言語として使用している。

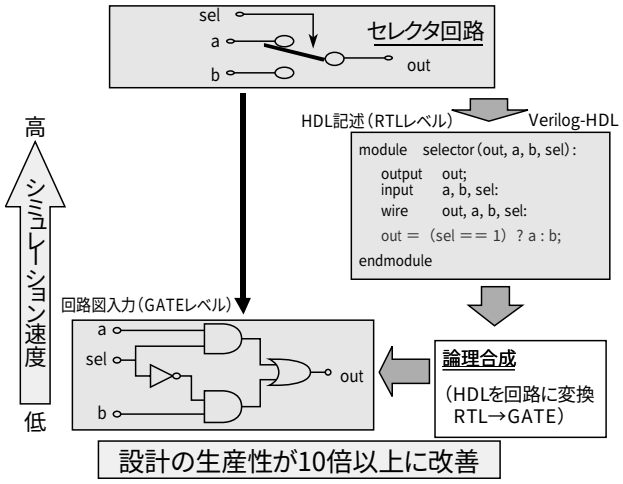

図-3　HDLを利用した回路設計
Fig.3 Circuit design using HDL

2）記述チェッカによる品質向上

設計したHDLに対し，当社の設計ノウハウを反映させた記述チェッカを適用する事で，文法チェックや記述ミスの低減，論理合成などの後工程に不向きな記述の摘出を行っている。これにより，早期にバグの発見／修正が可能になり，設計の効率化，回路品質の向上を図っている。

3）テストパターンの精度向上による回路品質の向上

シミュレーション時に重要なのは，シミュレーション自体の高速化はもちろんのこと，いかに短時間で全ての機能を検証できるかである。

当社では，カバレッジツールを用いることで，テストパターンによってどれくらいの記述や機能を検証できているかを把握し，必要最小限のテストパターンで効率良く検証している。

また，LSI出荷時に使用する故障検出用のテストパターンに関しても，故障シミュレーションを実施することで，高い故障検出率を確認し，不良品の市場流出を防いでいる。

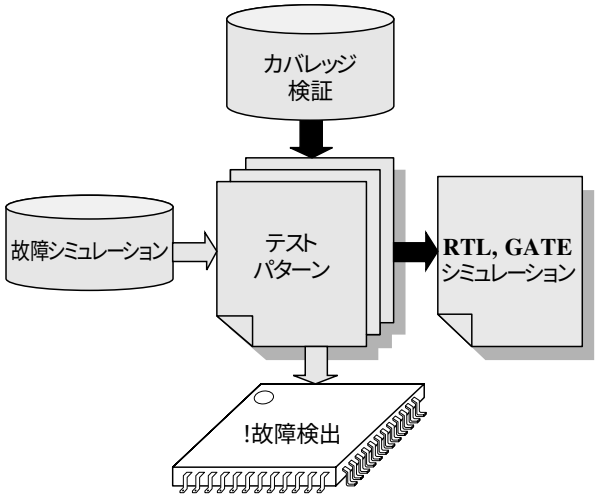

図-4　テストパターンの精度向上
Fig.4 Improvement of test pattern accuracy

4）静的検証による設計効率化

論理合成後のGATE回路は，論理検証の他に，温度変化や電源電圧変動があっても要求動作周波数で回路が動作するか (タイミング検証)も検証する必要がある。

しかし，GATEシミュレーションは，RTLシミュレーションに比べ数倍～数十倍遅くなり，さらに同一テストパターンでBestCase(温度低，電圧高)，WorstCase（温度高，電圧低）について検証しなければならず，非常に効率が悪い。

そこで，テストパターンで回路を動作させて検証する動的な検証ではなく，回路内の各フリップフロップ間の遅延を計算し，それが規格時間(セットアップ／ホールド時間)を満たしているかどうかで判断する静的検証ツールを用いている。

静的検証は，シミュレーションに比べ1000倍以上高速で，テストパターンを使わず，機械的に全ての回路の遅延を計算するため，回路を100%検証できるのが特徴である。

5）FPGAによる並列検証

シミュレーションによる検証だけでは，膨大な時間がかかり，データ量も多いため，動作不具合の発見漏れが生じる恐れがある。そこで，容易に回路の書換えが可能な，FPGA(Field Programable Gate Array)を用いた実ボードによる検証を実施している。

FPGAを利用する事で，シミュレーションに比べ1000～数10万倍高速に論理・機能検証できる。また，システム全体での動作確認が可能となるため，動作不具合を発見しやすいという利点がある。

ただし，FPGAは動作周波数が遅く，回路規模に制限もあるため，FPGA設計に時間がかかることが多い。

そこで当社では，FPGA専用に配置配線を考慮した論理合成ツールを用いる事で，動作周波数と回路規模の最適化を図り，早期設計を可能にしている。

このように，ディジタルLSI設計では，多くのツールを利用し，いろいろな検証手法を使うことで，高品質のLSIを短期間で開発できる環境を構築している。

2.2 アナログLSIの設計環境

アナログLSIでも高集積化は進んでいるものの，設計手法そのものに大きな変化はなく，回路図を作成してSPICE(Simulation Program with Integrated Circuit Emphasis)系シミュレータで回路検証すると言う，古典的な手法が一般的である。

当社では，1987年に初めてSPICEを導入した。

以後，高速なシミュレータやPC上で動作するシミュレータを導入することにより，設計期間の短縮を図っている。

マスクレイアウトでは，DRC(Design Rule Check)で線幅や間隔の自動チェックを行い，LVS(Layout vs Schematic)で回路図とレイアウトを自動照合することにより，ミスの防止を図っている。

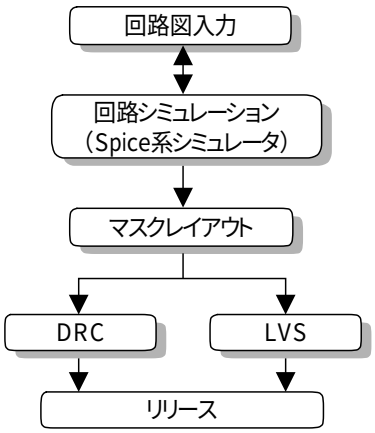

図-5　アナログLSI設計フロー
Fig.5 Analog LSI design process

2.2.1 ライセンスの有効利用

アナログ設計では前述のような手法のため，設計規模の拡大にともない，設計人員の増加，およびライセンスを増やす必要がある。しかし費用が高いため，簡単には対応できない。

そこで設計作業中のライセンスの必要性に着目した。設計時に使用するライセンスは，大きく分けて2種類ある。一つは回路図の編集用。もう一つはシミュレータそのもののライセンスである。

シミュレータ用ライセンスは，シミュレーションが流れている間，常に必要となるため，ライセンスを増やす以外に対処方法はない。

しかし，回路図編集用ライセンスは，作業時以外は必要ない。現状では一度ライセンスを掴むと，回路図を閉じるまで保持している。

そこで，編集時以外はライセンスを開放するようなプログラムを作成することで，ライセンスを時分割で複数の設計者が同時に利用できるよう工夫し，費用発生を抑えている。

2.2.2 AHDLの活用

アナログ設計においても，ディジタルでのHDL設計と同様の，アナログ記述言語(AHDL:Analog Hardware Description Language)を活用した設計手法が提案されている。

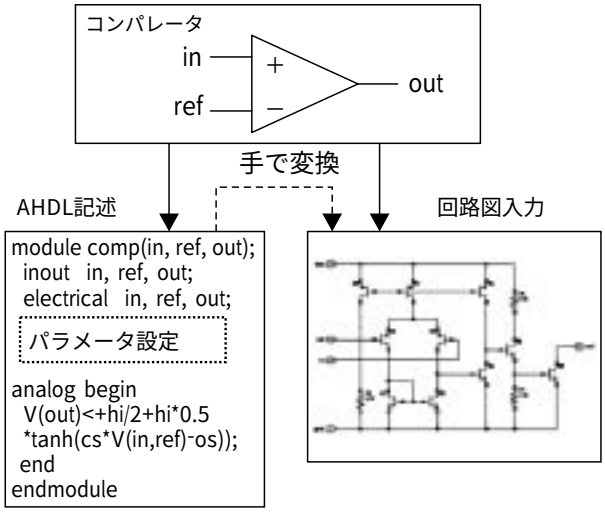

図-6　AHDLを利用した回路設計
Fig.6 Circuit design using AHDL

AHDL記述された回路は，トランジスタで構成された回路をSpiceでシミュレーションするのに比べ，(モデルのレベルにもよるが)10～100倍以上高速にシミュレーションできる。

また，設計初期段階から全体の検証を行うトップダウン設計に適しており，設計の最適化や，I/F等の不具合を早期に発見できるという，大きな成果が期待できる。

当社では現在，企画段階のICをAHDL化し，動作検証を行うことでICの実現性を見極め，製品開発への判断材料となるよう活動中である。

現状では，AHDLからトランジスタレベルへの自動変換ツールが，まだ実用レベルに達していないという問題があるが，今後，市場動向を見極めて対応していく予定である。

2.3 ディジアナ混載設計環境の構築

ディジアナ混載のICは，1996年にリニアソレノイド用ICから本格的に採用を始めた。当初の設計手法は，ディジタル部とアナログ部を個別に設計し，各々が完成した時点で接続していた。

接続した回路を検証するには，ディジタル回路を全てトランジスタに置き換え，アナログ回路と共にSPICE系シミュレータで検証するか，個別のシミュレータを用い，通信プロトコルで双方のシミュレータを接続して検証する方法がある。

しかし，これらの方法はシミュレーションに多大な時間がかかる場合が多く，操作性も悪いため，実設計では活用されていないのが現状であった。

その結果，ディジアナ接続部の単純な結線ミスやI/F部の動作不具合が発生し，改版を余儀なくされていた。

そこで，ディジタルとアナログが混在した状態でシミュレーションが可能なミックスドシミュレータを導入した。

このシミュレータはさまざまなレベル(トランジスタ，アナログ言語記述，ディジタル言語記述，システム言語記述 等)が混在した回路を，一つのシミュレータでシミュレーションが可能なため，従来方式よりも高速に検証できるようになった。

当社のディジアナ混載ICは，回路の大半をアナログ部が占める場合が多い。この場合，トランジスタレベルのシミュレーションが大部分を占めるため，ミックスドシミュレータと言えど，大幅な時間短縮は望めない。

そこで，検証内容に応じ，関係ないブロックをAHDLに置換えて，抽象度を上げることにより，シミュレーション時間の短縮を図った。その結果，全てトランジスタレベルで検証するよりも，約3分の1の時間に短縮することができた。また，結線ミスやI/F部に動作不具合がないことを確認でき，不安なくIC製造へ移ることができた。

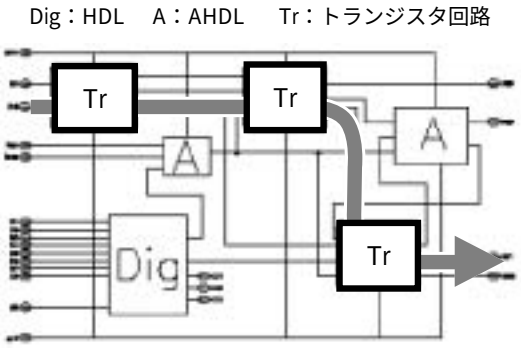

図-7　ディジアナ混載回路
Fig.7 Digital-analog mixed circuit

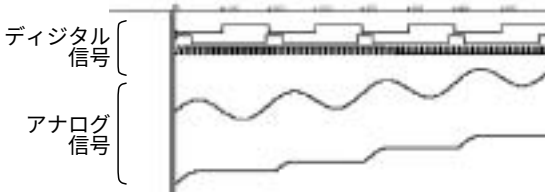

図-8　シミュレーション結果
Fig.8 Simulation results

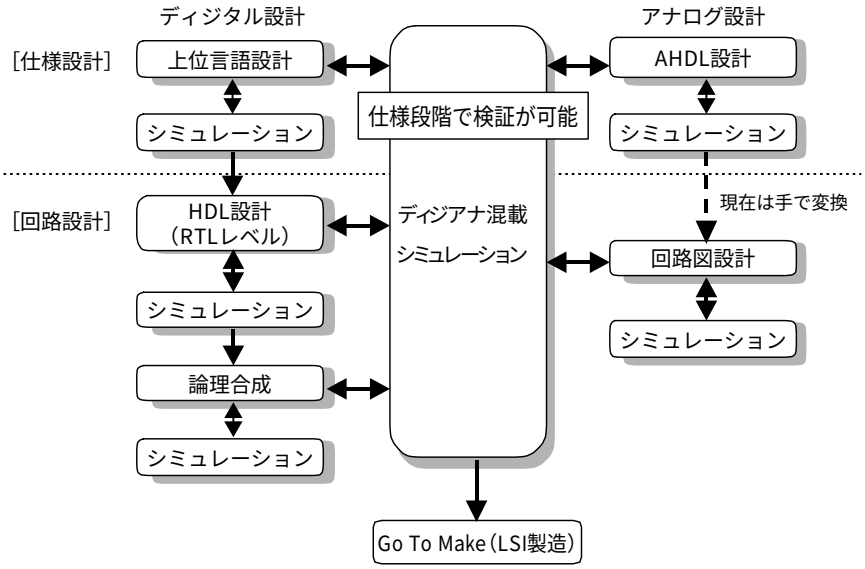

図-9　ディジアナ混載ICでのトップダウン設計フロー
Fig.9 Top-down design process with digital-analog mixed IC

3 設計の効率化を促進させる取組み

前章で述べたように，設計期間の短縮を図るため，効果的なツールを活用できる環境を構築してきた。しかし，設計規模増大はさらに進んでおり，さらなる対策が必要となっている。そこで当社では，費用を抑えながら，効果的な設計環境を構築するために，以下の3つの活動を推進している。

①LSI設計設備の有効活用
②設計資産の再利用
③自動評価の推進

3.1 負荷分散ソフト(LSF)による設計効率化

LSF(Load Sharing Facility)とは，ネットワーク上のコンピュータを一元管理し，ツールを最適なコンピュータ上で実行したり(負荷の分散)，使用状況を管理するソフトのことで，当社では，1999年からLSFを導入し運用している。

3.1.1 LSFのメリット

1）ハードウェアの有効利用

設計用のコンピュータ(EWS)の性能は，年々向上している。そのため，導入時期により社内のEWSにも性能にバラツキがある。

設計者が自分で実行EWSを決めてシミュレーション等を実行する場合，どうしても性能の良いEWSに実行が集中してしまう。すると，せっかくの高速なEWSも，処理能力が半減以下になり，短時間で検証ができなくなるという問題が発生していた。

そこで，LSFを用いて"最も負荷が軽く，最も効率的な検証が可能なEWS"で自動実行できるようにすることで，実行時間を最小限にし，また，EWSの資産を最大限に活用することが可能となった。

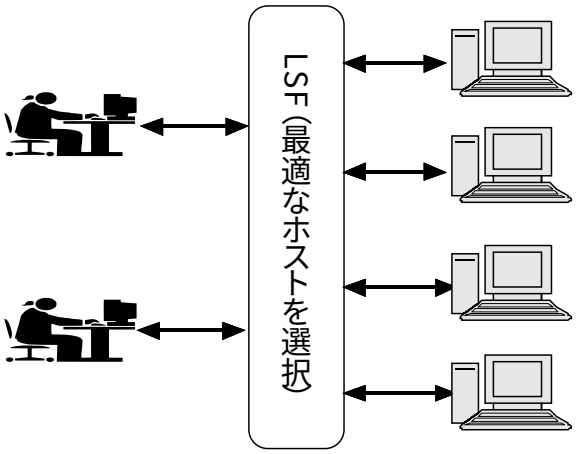

図-10　LSFを活用したEWSの有効活用
Fig.10 Effective utilization of LSF-utilizing EWS

2）設計ツールの有効利用

LSI設計に使用するLSI設計用ツールは高価なものが多い。そこで，いかに必要最小限の設計ツールで最大限の効果を出すかも重要なポイントになる。

LSFでは，各設計ツールの使用頻度を容易に把握できるため，ライセンス増減の指標とすることが出来る。

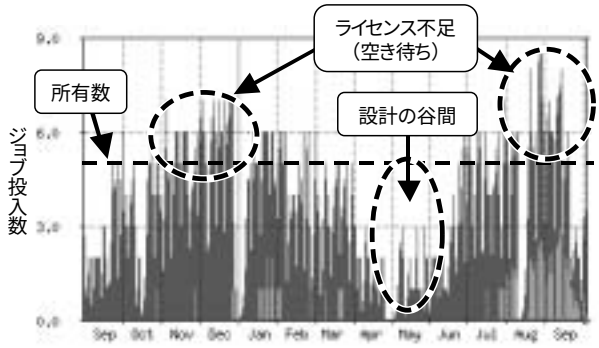

図-11　ツール使用頻度の把握
Fig.11 Monitoring of tool use frequency

当社では，このLSFを利用し，以下のような工夫をしている。

①CPU負荷がかかるツールは，1CPUで1ジョブしか流れないようにした。
②CPU負荷を常に必要とせず，回路表示や波形表示など，表示をメインにしたツールは，ややCPU能力が劣る端末を中心に実行するようにした。
③異なるOS同士でもディレクトリ構造を全く同じにすることで，OSの違いを気にすることなくジョブを実行できるようにした。

この結果，ツールにより若干差があるが，約20%～40%の実行時間短縮を図ることが出来た。

3.2 設計資産の再利用

3.2.1 IPとしての登録

大規模化した回路を，すべて一から設計していたのでは短納期に対応するのは難しい。

この問題に対応する手段として，過去の設計資産(IP：Intellectual Property)を回路設計時に再利用する手法が広まっている。

IPを利用することで，新規に設計するブロックを減らすことができる。また，同じプロセスの場合，再利用した部分は動作が保証されているため，検証する必要がない。その結果，大規模LSIでも短期間で開発することが可能となる。

当社では，ディジタル設計においては，通信や演算系のモジュールをIP化している。アナログ設計においては，オ

ペアンプやコンパレータといった回路をIP化している。

3.2.2 IP活用度向上のための取り組み

これらのIPは，各プロジェクト毎にまとめられていたため，全てのIPを効率よく活用できていなかった。

そこで，IPに対するデータベースを構築し，全てのIPをweb上で確認できるようにした。登録するIPにはルールを制定し，レベルを統一。また，仕様や特性，使用履歴情報を登録することにより，設計者が不安なくIPを利用できるよう工夫している。

現在のIP利用率は，ディジタルにおいては約20%，アナログではプロセス毎に10～60%とバラツキがあるものの平均約30%の利用率となっている。

今後もシステムの改善，IPの登録を推進することで，さらに利用率が上昇し，設計期間短縮が図れるものと考える。

図-12　IP管理データベース
Fig.12 IP control database

3.3 LSI評価環境の効率化推進

LSIの短期開発を実現するには，設計の効率化と同時に，そのテスト・評価をいかに効率的に実施するかも重要となる。特に近年のシステムLSI化に伴い，そのテスト内容も多様化，複雑化し，効率的な評価を行うための環境整備が重要になってきている。

評価の効率化を図るために，当社ではLSIテスタによる評価の自動化を進めている。

3.3.1 ディジタルLSIの評価環境

ディジタルLSIの機能試験(ファンクションテスト)をLSIテスタで実現するには，テストパターンが必要となる。

このテストパターンを作成するために，2つの方法を実施している。

①シミュレーション結果の波形データから，直接テストパターンを生成できるツールを活用

②LSIテスタの独自言語で自社開発した変換ツールの活用

これらを活用することで評価準備工数の削減を図っている。

3.3.2 アナログLSIの評価環境

アナログLSIにおいては，ディジタルでの波形データに相当するものがないため，流用することができない。評価のためには，試験仕様を基にプログラム作成作業が必要となる。

当社では，アナログ専用に，試験仕様書からテストプログラムを自動生成するツールを開発した。

これにより，設計者が複雑なプログラミング作業から開放され，簡単にLSIテスタを利用できる環境となっている。

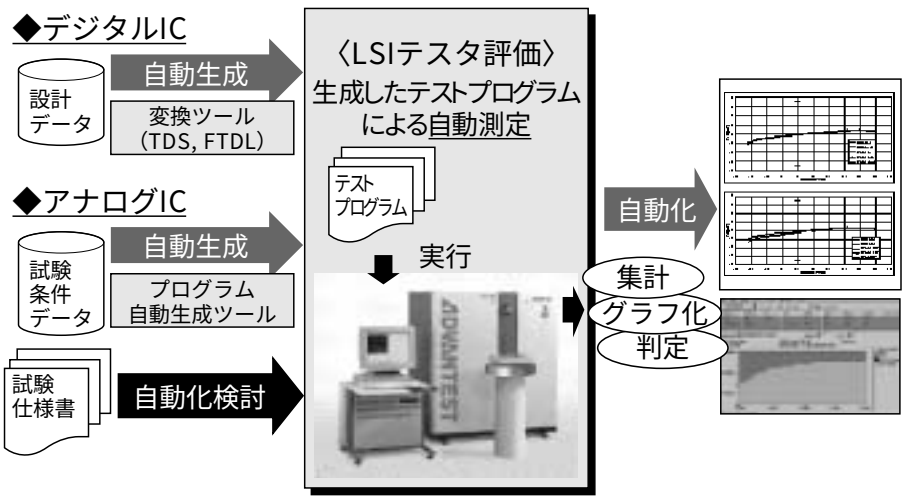

図-13　LSI評価環境
Fig.13 Outline of LSI evaluation

4 他社との設計環境比較

前述の設計の効率化を進める中で，当社の環境が同業他社と比べ，どの程度のレベルかを確認した。

調査項目として，①設計の生産性，②設計ノウハウ，③ツール数，④EWS数，⑤設計品質を挙げた。

①設計の生産性

一人当たり1ヶ月に何ゲート，何素子の設計能力があるかを比較。

②設計ノウハウ

開発ICのIP使用率を比較。

③ツール数

ツール1本に対する設計者の比率を比較。

④EWS数

EWS1台に対する設計者の比率を比較。

⑤設計品質

開発ICの一発完成度を比較。

表-1　ディジタル設計環境の比較
Table 1 Comparison of digital design environments

	A社	B社
①設計生産性	―	―
②設計ノウハウ	○	○
③EWS	―	―
④CADツール	―	×
⑤設計品質	―	―

表-2　アナログ設計環境の比較
Table 2 Comparison of analog design environments

	C社	D社
①設計生産性	―	―
②設計ノウハウ	△	×
③EWS	○	―
④CADツール	―	―
⑤設計品質	―	○

○：当社より少し勝る
―：当社と同等レベル
△：当社より少し劣る
×：当社より劣る

これらを比較した結果，アナログのIP活用で当社の優位性を確認できたが，ディジタルのIP活用では少し遅れが見られた。その他の項目に対しては，他社(ディジタル環境：2社，アナログ環境：2社)と遜色ないレベルの環境，能力を保持していることが確認できた。

5 今後の取り組み

これまで取り組んできたことをさらに進めながら，今後は以下の視点からも設計期間の短縮，設計品質の向上を図っていく。

5.1 C言語設計への対応

設計規模の増大に伴って，HDL設計でも設計期間が増加して苦労する場面が出てきている。

そこで近年注目されているのがHDLよりさらに抽象度の高いC言語による設計である。

C言語設計により，以下のようなメリットが期待できる。

①シミュレーションの高速化

C言語で記述することで，さらに抽象度があがる。その結果，記述量で約25分の1，シミュレーション速度では約50倍の高速化を図ることができる。

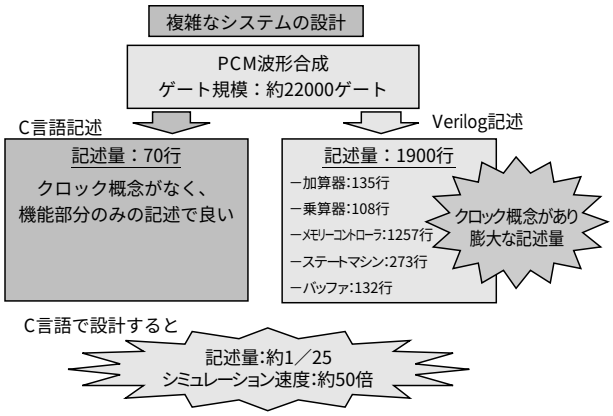

図-14　C言語での回路記述
Fig.14 Circuit description in C language

②ハード／ソフト協調検証

通常，ソフトデバッグは，ハードウェアが完成しないと実施できないが，バーチャルなハードウェアを用いることでソフトデバッグをハードウェアが完成する前に実施することが可能となる。

近年，ハードウェアをC言語で設計することで，より高速な環境下でデバッグが可能となってきており，ハード／ソフトの協調検証が注目されている。

既に大手ASICベンダでは，社内的に設計言語をC言語に統一する動きもある。

しかし，実際にはHDLからC言語へ完全に移行できていないというのが現状である。

その主な理由として，

・標準言語が制定されていない

・実用レベルのC→HDL変換ツールがない

が挙げられる。

また，HDL設計者にとってみると，新しい言語を覚えなければならない事，現状では，C言語からHDLへは手で変換する部分が存在することから，戸惑っているのが現状である。

5.2 UML言語への対応

UML(Unified Modeling Language)とは，統一モデリング言語と呼ばれる，仕様記述言語である。

仕様をUMLで表現することで，仕様の曖昧さが排除されるため，従来の設計手法で発生していた仕様ミスによる不具合を低減できる。

UMLは，今までソフトウェア開発の分野でメジャーな言語として知られていたが，システムLSI化によるハードウェア／ソフトウェア協調検証が注目されている現在では，C言語と共にハードウェア設計においても注目されている。

当社としては，これらの現状を踏まえ，世の中の動向を見ながらC言語やUMLを用いた設計に対応していく。

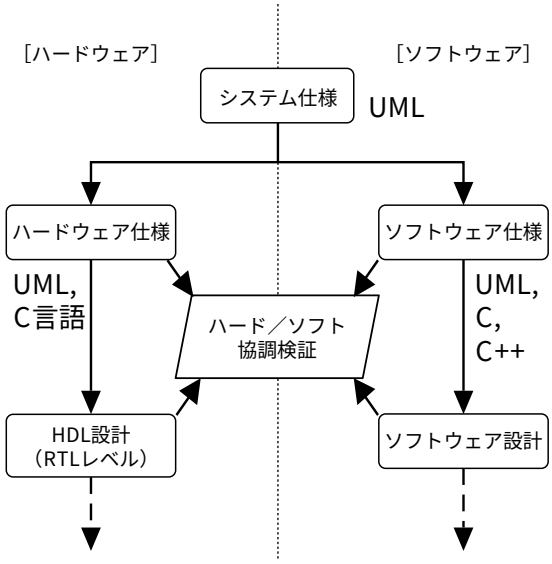

図-15　UML，C言語を用いた設計フロー
Fig.15 Design process using UML and C language

6　おわりに

今後，ますますIC開発における開発スピードと設計品質の向上が求められる。それに対処するには，本論で述べた課題達成が重要である。C言語設計，ディジアナ混載設計における高速化，設計資産の更なる有効活用などに重点をおいて，引き続き取り組んでいく。

筆者紹介

井原　渉
（いはら　わたる）
1993年入社。以来，車載用LSIの開発，LSI設計環境の開発に従事。
現在，開発統括部LSI開発部に在籍。

伏見　文孝
（ふしみ　ふみたか）
1993年入社。以来，車載用LSIの開発，LSI設計環境の開発に従事。
現在，開発統括部LSI開発部に在籍。

伊藤　隆志
（いとう　たかし）
1992年入社。以来，車載用LSIの検査および評価技術開発に従事。
現在，開発統括部LSI開発部に在籍。

前田　恵一
（まえだ　けいいち）
1974年入社。以来，移動通信関連の開発を経て，1995年よりICの開発に従事。
現在，開発統括部 LSI開発部長。