解　説

ハードウェア記述言語

2. 主要なハードウェア記述言語の特徴と標準化状況

2.4 Verilog☆ HDL†

野　地　保††

1. まえがき

設計の基本を論理図面ベースからハードウェア記述言語 (HDL) ベースへと，設計の中心を論理設計レベルから機能設計/方式設計レベルへと，より上流主体に移行し，しかも上流からレイアウトの下流まで後戻りのないトップダウン設計手法が実現可能となってきている．この手法の要である言語 Verilog HDL は，米国，日本において商用ベースで広く使用され，実質の業界標準となりつつある[1]．**図-1** にその手法における Verilog HDL の位置付けを示す．

HDL と設計環境は切り離して考えることはできない．Verilog HDL は，ユーザである設計者の立場に重点を置き，①使いやすく書きやすい②シミュレーション，論理合成，テストなど，言語回りのツール群が豊富に揃っている③言語仕様が公開されだれでもオープンに使える，など設計言語として必要な事柄を備えている．言語仕様の公開後，言語の普及活動と標準化を目的に Verilog HDL のユーザ団体 (OVI: Open Verilog International) が設立され，標準言語仕様書第1版が昨年発行された．現在，この言語仕様書をベースに新しい言語仕様の機能拡張，標準化が推進されている．

2. 言語設計の基本思想

Verilog HDL はミックスレベルディジタル論理シミュレータ Verilog-XL 用として 1984 年にゲートウェイ・デザイン・オートメーション社（現ケイデンス・デザイン・システムズ社）によって HILO-2, LALSD-Ⅱ, OCCAM, C言語を参考に開発された言語である[2]．以下の3点が開発指針とされた．

①人間が読みやすいこと．すなわち記述されてから時間が経過しても理解できること．また他の設計者が見ても理解できること．

②HDL モデルが論理合成可能なこと．すなわち構造的なネットリストで記述可能な実際のハードウェアを表現できること．

③HDL モデルが高速にシミュレーション可能で，論理合成結果に対しても，高い品質が得られること．

以上の指針を達成するために Verilog HDL では，アーキテクチャ，アルゴリズム，レジスタ転送，ゲート，スイッチの各レベルの記述およびその混在を可能としている．ただし Verilog HDL では待ち行列のモデル化をアーキテクチャレベルと称し，ハードウェアとの対応が1対1でない記述レベルをアルゴリズムレベルと称している．Verilog HDL の記述レベルと各レベルでの記述例を**図-2** に示す．以下ではアーキテクチャ/アルゴリズム/レジスタ転送レベル (RTL) の記述を動作記述 (behavioral description)，ゲート/スイッチレベルの記述を構造記述 (structural description) と称する[2]．

3. 言語の主な特徴

Verilog HDL は，シミュレータ入力言語として開発された言語であり，他のハードウェア記述言語と比較して高速シミュレーション，ミックスレベルシミュレーションが可能，などの長所があるが，高位レベルの表現・合成が不得意，状態遷移記述言語として使用する場合に注意が必要，などの短所もある．

以下，構造記述と動作記述，それぞれの構文，記述法，両者を混合したミックスレベル記述，階

† Verilog HDL by Tamotsu NOJI (Mitsubishi Electric Corp., ADEC).
†† 三菱電機(株)・CL 研(情)
☆ Verilog は Cadence Design Systems, INC の登録商標である．

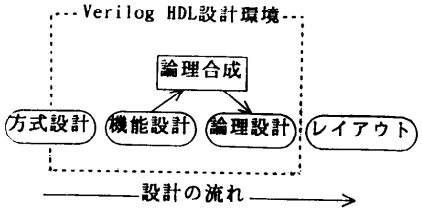


図-1　トップダウン設計手法での位置付け

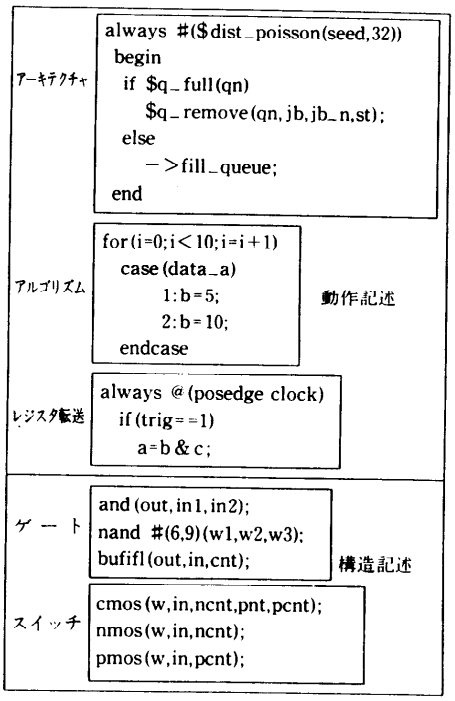


図-2　Verilog HDL の記述レベル

and	buf	nmos	tran	pullup
nand	not	pmos	tranif0	pulldown
nor	bufif0	cmos	tranif1	
or	bufif1	rnmos	rtran	
xor	notif0	rpmos	rtranif0	
xnor	notif1	rcmos	rtranif1	

図-3　あらかじめ用意されているプリミティブ

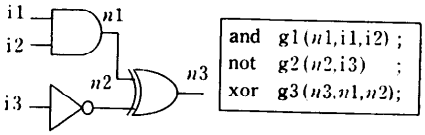


図-4　構造記述例

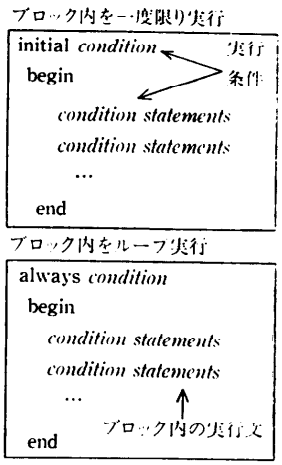


図-5　手続きブロックの実行概念図

層設計，典型的な Verilog HDL の長所，短所，制約などについて述べる．

3.1 構造記述 (structural description)

構造記述（ゲート/スイッチレベルの設計）は，言語仕様にあらかじめ用意されている 26 種のプリミティブ（図-3 に示す）を，物理的な配線に相当する net で接続する構造記述により行う．簡単な記述例を図-4 に示す．

3.2 動作記述 (behavioral description)

動作記述（アーキテクチャ/アルゴリズム/RTL の設計）は，手続きブロック (procedural block) を用いて行う．アーキテクチャ/アルゴリズム/RTL 各レベルでの記述上の区別はなく，always, initial などの手続きブロックを用いて全て表現される．図-2 で示すように，表現される内容が待ち行列のモデル化の場合をアーキテクチャレベルと称し，ハードウェアとの対応が 1 対 1 でない記述レベルをアルゴリズムレベル，クロックと対応可能な記述レベルを RTL と称している．図-5 に実行概念図を示す．initial ブロックはブロック内を 1 度限りしか実行しないのに対し，always ブロックはブロック内をループして実行する．

（1）手続きブロック内では以下の記述ができる．

①if, forever, repeat, while, for, case などの制御構造．

②begin～end による逐次実行．

③fork～join による並列実行．

④function, task で定義された動作，イベント制御など．

（2）手続きブロックの実行タイミングはユーザが指定した実行条件によって制御される．手続

きブロックと実行条件の関係を**図-6** に示す．同様の実行条件を手続きブロック文内の実行タイミングの制御に用いることもできる．全ての手続きブロックはシミュレーション時刻0で自動的に活性化 (activate) され並列動作を行う．構造記述では，net がメインのデータ型であるが，動作記述では reg がメインのデータ型となる．reg はハードウェアにおけるレジスタを抽象化したものであり，新たな値がロードされるまで値を保持する．

3.3 ミックスレベル設計

構造記述と動作記述を自由に混在させることが可能である．手続きブロックで生成された値で，ゲートやスイッチをドライブすることや，反対にゲートやスイッチで生成された値を手続きブロックで使用することが可能である．ゲートレベル (wire, reg, and のブロック) と RTL (always ブロック) 混在のミックスレベル設計の概念図を**図-7** に示す．

3.4 階層設計

階層の基本ユニットはモジュールである．モジュールは予約語である module と endmodule によって囲まれた部分で，全ての機能はモジュール内に記述する．モジュールはモジュール外と，入力，出力，双方向のポートを通じて信号値のやり取りを行う．ポートを通じた信号値はモジュール内のゲート，スイッチ，手続きブロックで使用することができる．モジュール内で他のモジュールを呼び出し，接続することによりモジュールのインスタンス生成を行い階層設計をすることができる．

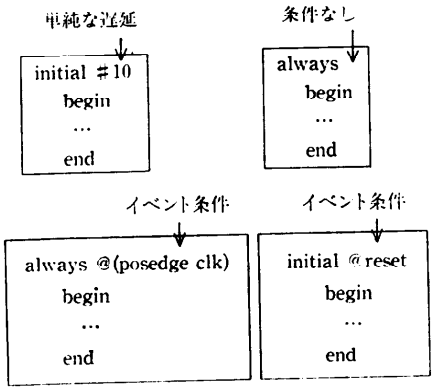


図-6 手続きブロックの実行条件

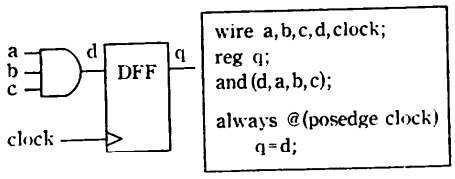


図-7 ミックスレベル設計の概念図

3.5 長　所

Verilog HDL の長所を以下に示す．

①アーキテクチャ，アルゴリズム，レジスタ転送，ゲート，スイッチの各レベルの記述およびその混在が可能である．

②言語仕様が単純であり，書きやすく読みやすい．

③高速のゲートレベルシミュレータが存在する．

④ライブラリが豊富である．

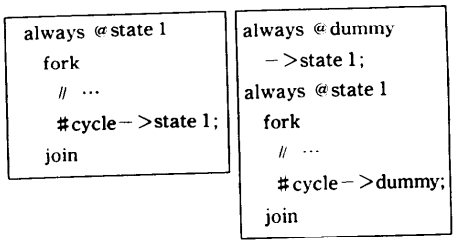
⑤言語の使用実績が多い．

3.6 短所・制約

状態遷移記述言語として使用するとき次のような注意が必要である．

①状態遷移をイベント制御文を用いて記述する場合，Verilog HDL ではイベント文の中で自分自身のイベントを発生させても認識されないので，次サイクルでも同一の状態に留まる（現状態を維持する）ためには，ダミーのイベントが必要になる．たとえば複数サイクルにわたって一つの状態に留まるためには，**図-8** (a)ではなく**同図** (b) のように記述しなければならない．

②状態遷移を階層的に記述することができない．ハードウェアの階層構造は module を用いて記述することができるが，動作を階層的に記述することができない．ハードウェアに依存しないイベント制御文として唯一用意されている "@"，"->" は，それ自身記憶機能をもたないので，状態遷移記述には不向きである．したがってたとえば複数の状態遷移（オートマトン）を連動させて，



(a) 留まらない例　　(b) 留まる例

図-8 状態遷移記述例

並列動作，パイプライン動作を記述する場合も，1階層のイベント制御のみで記述しなければならず，結局 Verilog HDL による動作記述では，動作ではなくハードウェアの階層構造でこれらを記述することになる．したがってデータパス用のリソース（レジスタ，ALU，メモリなど）だけでなく，制御パス用のリソース（状態遷移レジスタ）も常に意識しなければならない．

4. 処理系の開発状況

論理合成ツール，シミュレータ，テスト，その他のツールが豊富である．

（1） ユーティリティ

Verilog HDL から他ツール（シミュレータ，タイミング検証ツール，テスト装置など）用フォーマットへの変換や他のツールから Verilog HDL への変換ソフトウェアが開発されている．

また，Verilog HDL ソースの高機能エディタやパーザが開発されている．

（2） 論 理 合 成

論理合成ツールの多くは独自の入力フォーマット（ブール式，真理値表，状態遷移表）のほかに，Verilog HDL によるレジスタ転送レベルの機能記述もサポートしている．

（3） シミュレータ

シミュレータの多くは Verilog HDL を自フォーマットに変換し使用している．Verilog HDL の動作記述，構造記述による機能および論理シミュレーションの直接処理が可能な Verilog-XL コンパチィブルシミュレータ (CAD Artissans 社 AuSIM-XL) なども，提供されてきている．

（4） テ ス ト

テスト用の記述に Verilog HDL を使用したり，LSI の動作記述，構造記述から LSI チップ評価用のテストプログラムを自動的に生成することができる．

（5） そ の 他

機能ライブラリ，グラフィカルな状態遷移図入力による Verilog HDL モデルの生成および解析，Verilog HDL から FPGA に自動変換し，LSI または装置のプロトタイプを製作するなどユニークなツールがある．また，Verilog HDL を自社の CAE システムやツールキットに組み込み活用しているものもある．

5. 標準化の状況

HDL の普及にともない，ユーザ（設計者）側の使いやすさ，高度な機能への要求が発生してくる．付随するライブラリやシミュレータ，論理合成ツールなどの設計環境も重要である．ユーザは，これらの設計環境を自由に構築できたり，不特定多数のベンダが提供してくれることを期待している．

Verilog HDL の言語仕様は全ての人に公開され，OVI (Open Verilog International) により標準化も推進されている[3]．

5.1 公 開

公開は，1989 年のケイデンス社とゲートウェイ・デザイン・オートメーション社の合併後，実現され，これ以後特定の会社（ケイデンス社）専用の言語から，ユーザ，ベンダが自由に利用できる言語となった．

（1） 公開/標準化のメリットを以下にあげる．

①設計ツール間の互換性/流用性の向上
標準インタフェースでの開発が可能である．

②ライブラリの共通化
互換性のある HDL レベルでのライブラリの提供が容易である．

③設計ツール統合化の容易性
より多くのベンダから自システムにあったツール，ライブラリの選択が可能となる．

④設計者の生産性の向上
設計データ，HDL 記述モデルを自由に活用できる．

（2） 公開のサポート内容

現在，次のサポートがケイデンス社より提供されている．将来，これらの標準マニュアルは OVI からサポートされる予定である．

- Verilog HDL リファレンスマニュアル
- HDL モデリングハンドブック
- ホットラインサポート
- 開発者用ツールキット

5.2 OVI (Open Verilog International)

Verilog HDL が公開され，ケイデンス社の束縛から一般ユーザに解放され，だれでも自由に使用することが可能になった．このことは反面 Verilog HDL の方言化を招く恐れや，新しい機能向上を標準仕様として盛り込むことの必要性も出てくる．

このような背景のもとに，Verilog ユーザ，ASIC ベンダ，ツール開発のサードベンダが中心となってケイデンス社とは独立した非営利団体である OVI が米国で設立された．日本においても OVI 日本 WG（ワーキンググループ）として活動を開始している．現在 WG はヨーロッパでも準備中である．

（1） 活動目的は，①機能向上の推進と②Verilog HDL, PLI (Programming Language Interface), SDF (Standard Delay File Format) の標準化とその維持である．

（2） 活動内容は，①Verilog HDL, PLI, SDF リファレンスマニュアルの作成，改訂，配布②Verilog HDL への準拠および互換性のテスト方法の確立③ユーザと EDA ベンダへの普及/促進④VHDL, UDL/I 委員会との連絡/調整などである．

（3） 技術小委員会の活動

OVI には，技術検討組織として技術協議委員会 (TCC) が設置されており，現在以下の6つの技術小委員会 (TSC) がある．

①互換性および準拠の検査
Verilog HDL 準拠および検証手順の開発．

②テ ス ト
テスト関連の改良要求の評価．

③アーキテクチャ・モデリング
高位レベル/アーキテクチャ・モデリングに関する検討．

④ロジックモデリング
SDF リファレンスマニュアルの作成．

⑤言語サポート
言語リファレンスマニュアルの作成．言語機能拡張プロポーザルの作成．

⑥論理設計手法
PLI リファレンスマニュアルの作成．設計手法/合成/テスト容易化設計．

これらの技術小委員会 (TSC) は，必要に応じて内容の見直しが図られている．

（4） OVI 日本 WG

OVI 日本 WG は 1991 年初に発足し，1992 年5月現在メンバは企業 16 社，大学3校より構成されている．米国 OVI の活動状況のフォロを中心に，言語マニュアルに焦点をあて，レビュ結果を米国技術小委員会 (TSC) にフィードバックする作業を進めており，現言語仕様での使いにくい点や，追加して欲しい点をまとめ，OVI に提言する第一ステップ作業が完了した．逐次，言語の特定機能（たとえば論理合成やシミュレーションなど）にポイントを絞り，ユーザの立場でどうあるべきかの方向を検討していく予定である．

5.3 標 準 化

1991 年6月 OVI より，Verilog HDL の標準言語仕様書となる，リファレンスマニュアルのドラフト第1版 (Draft Release 0.1 July 1, 1991) が発行された[4]．

第1版の内容はケイデンス社より公開された Verilog HDL のリファレンスマニュアルをベースに，シミュレータ Verilog-XL[5] 固有の言語仕様の削除，ケイデンス社ロゴマークの削除，その他ケイデンス社に関する記述の削除などを行ったものである．マニュアル全体の構成も多少変更されているが，言語仕様に関しては従来のケイデンス社発行のものと大差ない．この OVI 発行のドラフト第1版により，言語仕様そのものは，ケイデンス社とは独立して発展/標準化されることになった．

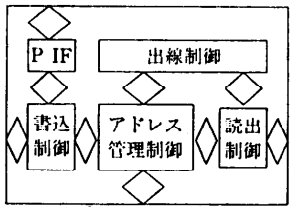
業界標準を狙った OVI の Verilog HDL 標準化活動は，始まったばかりで，言語機能の技術的な改版の狙いや改良/拡張の具体的項目に関しては，明らかではない．ユーザ/ベンダからの要求に基づいて今後決められていくことになる．

第1版以降，ユーザ/ベンダから TCC に 49 の言語仕様改良要求がだされ，最終的に 32 の項目に整理され，現在検討されている．そのうちの一部が第2版（発行予定 1992 年末）に盛り込まれる予定である．

その主な改良検討項目は，RTL 論理合成機能の強化，状態遷移機能の改良，VHDL との親和性機能（列挙型新データタイプの追加，VHDL の GENERATE 文の追加），C との親和性機能（C型代入文の追加），アーキテクチャレベルの機能強化，マクロ機能改良（C型パラメータ追加）などである．

なお，SDF, PLI のリファレンスマニュアル第2版もリリースされる予定である．

OVI では新たに，現状商用レベルでの業界標準である Verilog HDL の IEEE 標準化を推進する活動が発足した．




```
記述例
always@ (posedge clk)
  if(rst)
    for(k=0; k<=127; k=k+1)
        mem[k]=1'b0;
        else
    if(ffclk)
        mem[address]=~rau;
```

図-9 通信制御機能チップ概略図

6. 応 用 例

データベースプロセッサ，画像処理用 DSP，BUS コントローラ，計算機/通信関連機器用 LSI，などのレジスタ転送レベルの機能設計に多く応用されている．

ここでは，組み合わせ回路，順序回路を含む通信制御機能をもつ約 60 KG の 1 チップ LSI の全体を機能記述し，論理合成ツールと接続した例[6]を述べる．**図-9** にその概略図を示す．LSI は制御回路，データパス回路，非同期回路からなり，目標動作周波数は 100 MHz である．

図-9 の機能記述例は，レジスタファイルの入力部（デコードとレジスタ）をモデル化したもので，レジスタ転送レベルで記述している．合成を考慮して 5 KG 程度のモジュールに分割して一階層もしくは多階層設計した．機能記述と合成ツールを使用すると人手設計に比較して 47% の期間で機能設計から論理検証までが完了した．

現在の合成ツールは，言語仕様を 100% カバーするものでなく，シミュレーション用機能記述を合成用に書き換える必要がある．

7. あ と が き

機能設計/シミュレーション完了後，論理合成からレイアウトをトップダウンに実行できる設計環境が要求されており，HDL の機能もこの手法に対応して進化する．『設計ニーズにあった標準言語』が，今求められている．Verilog HDL はミックスレベルディジタル論理シミュレータ用言語として，設計者にとって書きやすい特長をもっている．これらの特長を生かせる設計分野では，今後も積極的に使用されることになる．標準化では，言語の互換性を考慮しつつ新しい機能を拡張していかなくてはならない．Verilog HDL も IEEE 標準化に向けて動き出した段階である．

謝辞 貴重なコメントをいただいた査読者の方々ならびに，Verilog HDL の標準化に関して日頃より，貴重なご意見をいただいている OVI 日本 WG のメンバの方々特に事務局の村山順一氏に深く感謝します．

参 考 文 献

1) ME 知的ソフトウェア環境に関する調査報告書・知的ソフトウェア応用/ソフトウェア生産性尺度・ASIC 利用/処理系言語，92-パ-7，日本電子工業振興協会 (1992)．
2) 野地，清水，濱田：AD 7-2 設計言語の標準化 Verilog HDL，工業調査会，ADEE，pp. 25-35 (1992)．
3) 仕様公開や業界団体の設立で普及を狙うハードウェア記述言語 Verilog HDL，日経データプロ・マイクロプロセッサ，11 月号，pp. MC 8-000531～536.
4) Verilog Hardware Description Language Reference Manual Draft Release 0.1 July 1, 1991, Open Verilog International.
5) VERILOG-XL Reference Manual, Cadence Design Systems, INC.
6) 野地，小山，清水：機能設計からの論理合成評価，電子情報通信学会研究報告 VLD 91-136，ICD 91-222，pp. 59-66 (1992)．

（平成 4 年 6 月 1 日受付）

野地 保（正会員）
1947 年生．1970 年長崎大学工学部電気工学科卒業．同年三菱電機(株)計算機製作所入社．同情報電子研究所を経て，現在同 CL 研(情)所属．1992 年より長崎大学大学院後期博士課程在学中．コンピュータ・アーキテクチャ，トップダウン設計手法，論理合成などの研究に従事．OVI 日本 WG 委員，電子情報通信学会会員．