Keywords you should know. 第105回

知っておきたいキーワード

C言語ベースハードウェア設計技法

原　祐子†

† 東京工業大学 大学院理工学研究科 通信情報工学専攻

"C-based Hardware Design Technology" by Yuko Hara (Department of Communications and Computer Engineering, Tokyo Institute of Technology, Tokyo)

キーワード：高位合成，FPGA，設計自動化，組込みシステム

高位合成

従来は，専用回路（ASIC）やFPGA（Field Programmable Gate Array）上に実現するハードウェア（LSI回路）は，Verilog HDL言語やVHDL言語などのハードウェア記述言語を用いたRegister Transfer Level（RTL）設計により行ってきました．RTL記述では，「いつ（時間）・どのような処理（機能）を何（どのようなハードウェアモジュール）によって行うか」と「ハードウェアモジュールの階層構造」を記述し，ハードウェアの構成を明示的に定義します．設計するハードウェアの大規模・複雑化に伴い，2000年代からは，徐々にC言語やC++言語などのソフトウェアプログラムからRTL記述に変換するC言語ベースハードウェア設計が適用され始め，2010年代には普及の段階に入っています．この変換技法は，高位合成（High Level Synthesis：あるいは，動作合成（Behavioral Synthesis）や機能合成（Functional Synthesis））と呼ばれます．高位合成への入力となるソフトウェアプログラムには，実現すべきハードウェアの振る舞い（アルゴリズム）のみを定義し，高位合成ツールは，面積・性能・電力などの設計制約に応じて，ハードウェアの時間と構造を定義します（**図1**）．

高位合成では，ソフトウェアプログラムを再利用できる，記述量を削減できる（C/C++プログラムはRTL記述に比べて1/5～1/10程度），同一のソフトウェアプログラムから面積・性能・電力などの異なる複数のハードウェアを設計できるなどにより，設計生産性を大幅に改善できるメリットがあります．多くの高位合成ツールでは，設計制約オプションや入力プログラム中に記述するプラグマなどにより，生成されるハードウェアを変更できます．高位合成はさまざまなアプリケーションドメインに適用可能ですが，特に，データ処理が中心となる映像信号処理は高位合成が得意とする分野で，映像系専門のLSI設計者の減少という背景もあり，多用されています．

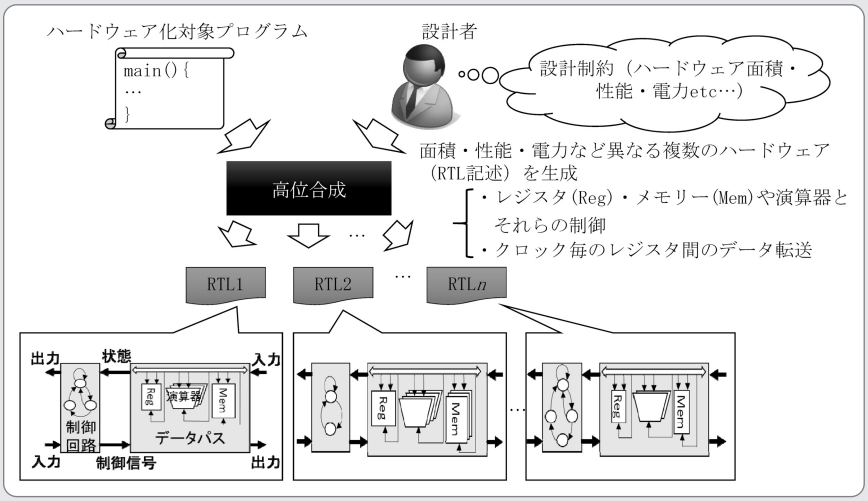

図1　高位合成による設計フロー

映像情報メディア学会誌 Vol. 69, No. 5, pp. 442～444（2015）

ハードウェア構造を意識したソフトウェアプログラム記述

多くの人が懸念するのは，「果たして高位合成によって，従来の人手によるRTL設計に適うハードウェア設計ができるのか」というところでしょう．確かに，いくらソフトウェアプログラムをそのまま流用できると言っても，ハードウェアを意識した記述でなければ，面積や性能の良いハードウェア設計はできません．

例えば，**図2**上部のプログラムは，インデックスiの値によって実行すべき処理が異なる二つのループ文（LOOP L1およびLOOP L2）から構成されています．この記述は，処理内容を理解しやすく，可読性の高いプログラムであると言えますが，高位合成により生成されるハードウェアには二つのループ構造の重複（i=100～9898間）ができ，非効率です．図2下部のプログラムは，このような重複処理を極力減らすように改変（LOOP L1およびLOOP L2を併合）したもので，合成されるハードウェアの性能を大幅に改善できます．このように，高位合成では，入力のソフトウェアプログラム上での記述がどのようなハードウェア構造に変換されるかを意識したC/C++プログラム設計が重要になります[1]．

```
void filter(int a[], int c[]){
   int b[10000], i;
   for (i=1, i<9999,  i++)          // LOOP L1
      b[i] = a[i-1] -2*a[i] + a[i+1];
   for (i=100, i<9899,  i++)      // LOOP L2
      c[i] = b[i-100] -2*b[i]+b[i+100];
}
```

ハードウェア構造を意識しないソフトウェア記述

```
void filter(int a[], int c[]){
   int b[10000], i;
   for (i=1, i<9999,  i++){      // LOOP L1+L2
      b[i] = a[i-1] -2*a[i] + a[i+1];
      if (i >= 200)
         c[i-100] = b[i-200] -2*b[i-100]+b[i];
   }
}
```

ハードウェア構造を意識したソフトウェア記述（ループ併合）

図2　高位合成向きソフトウェアプログラムへの改変例

高位合成を使いこなすには

より良いハードウェアを高位合成するための入力ソフトウェアプログラムは，使用する高位合成ツールの特徴によって異なることがありますが，ツールによらず，共通して言える点を以下に紹介します[2]．

変数のサイズ（ビット幅）と符号の定義は，生成されるハードウェア規模に直結します．変数のサイズは，各高位合成ツールが入力のソフトウェア言語に施している言語拡張により，明示的に定義できます．

ソフトウェアプログラム中の配列等のデータ構造は，高位合成されるハードウェア中では，それぞれ独立の幅と深さを持ったメモリーモジュールによって実現されます．したがって，単一の配列を多用すると，メモリーモジュールのアクセスに時間を要し，実行時間が増加します．複数のメモリーモジュールへ並列してアクセスできるように，データ構造の分散の検討が必要です．

ソフトウェアプログラムでは逐次処理で記述されますが，ハードウェアは並列処理が可能です．演算や関数レベルの依存関係を考慮して，並列化が可能な記述をすることが重要です．高位合成ツールによっては，複数のステートメントの並列性を明示的に記述できるように入力のソフトウェア言語を拡張しているものがあります．その他，多くの高位合成ツールで，ソフトウェアプログラム中のプラグマや制約ファイルで定義するオプション（ループアンローリングなど）により，並列化をサポートしています．

多くの高位合成ツールで，ポインタ演算やポインタ変数の使用には制限があります．使用できる場合でも，効率の良いハードウェア構成に変換されないこともあります．上述の通り，複数メモリーモジュールを利用し，サイズを管理することで，ポインタの使用は自然と制限することが考えられます．

ハードウェア化の対象プログラムが大規模である場合には，あらかじめモジュール分割しておく必要があります．高位合成ツールに入力できるアルゴリズムの規模には限界があり，一定量を超えると最適化処理に非常に長い時間を要するためです．さらに，生成されるハードウェアの制御部が複雑になり，性能（クロック周波数）が上がらないことが多くあります．モジュール分割時には，上述の通り，モジュールの並列性を考慮した分割をすることで，ハードウェアのさらなる性能改善に繋がります．

上記の他，留意すべき点は，C/C++言語記述のうち高位合成できない記述がある点です．再帰関数，浮動小数点，関数ポインタ，メモリーの動的割当て（malloc，freeなど），および，ライブラリーの呼び出し（printf，ファイル入出力など）です．

FPGAと高位合成

FPGAとは，製造後に設計者や購入者が構成を変更（再構成）できる集積回路のことを言います．**図3**の概略図に示すように，論理と配線をつなぐスイッチを自由にプログラムでき，所望のハードウェアをFPGA上に実現できます．論理情報と配線情報を変更可能であるため，修正・仕様変更が容易に行えるというメリットがあります．FPGAが登場した当初は，主にASICの試作用途に使われていましたが，FPGAの大容量化が進むにつれ，より大規模なアプリケーション用ハードウェア（アクセラレータなど）も実現可能になっています．さらに，最近では，高位合成技術の発展も相まって，高位合成を用いたFPGA設計環境が充実しています（**図4**）．このような開発環境の整備の流れは，FPGAベンダーのみでなく，システムベンダーでも見られます．

FPGAベンダーでは，XilinxはC/C++/SystemC（SystemCはC++を拡張したハードウェア記述言語）プログラムから同社の最新FPGA向けハードウェアを合成可能なVivado HLSを，Alteraは同社FPGA向けのOpenCL（OpenCLはCPU/GPU/Accelerator等のさまざまな並列コンピューティング向けのクロスプラットフォームなフレームワーク）開発環境をそれぞれ提供しています．システムベンダーが提供する高位合成ツールの多くは，ASICとFPGAの両方をサポートし，さらに，FPGA向けの最適化機能なども盛り込んでいます．

このようなFPGAのプログラム開発環境の発展により，高位合成，および，FPGAの利用は，特に通信・画像処理分野で高く評価されています．さらに，最近では，金融アプリケーションの分析・演算の高速化という実用例もあり，ますますアプリケーションの幅が広がっています[3]．

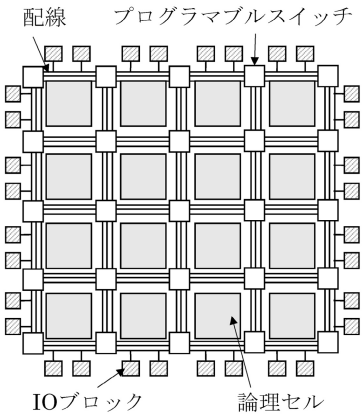

図3　FPGAの構成の概略
（ブロックRAMやDSP等は省略）

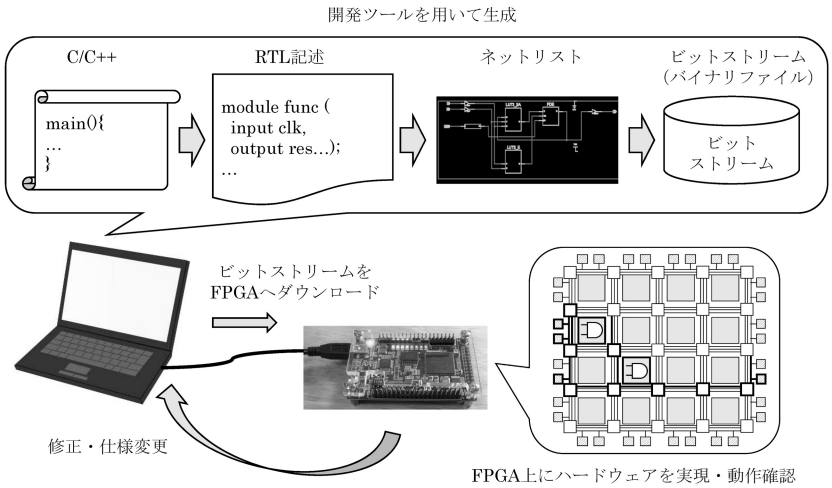

図4　高位合成を用いたFPGA開発フロー

むすび

C言語ベースハードウェア設計技法として，高位合成の概要とその活用方法について紹介してきました．近年のアプリケーションの多様化に加え，今後はInternet of Things社会の普及により，アプリケーションはますます大規模化・多様化するであろうことは想像に難くありません．上述の例に見るように，今後は，ハードウェアの設計指標（面積・性能・電力など）のみではなく，設計期間が製品・サービスの優劣を決めるアプリケーションも増えることが考えられます．高位合成とFPGA設計環境を活用することにより，ハードウェア設計を効率化することはもちろん，ハードウェア設計の効率化による新たなアプリケーションの創出など，多岐にわたる発展が期待されます．

（2015年1月30日受付）

参　考　文　献

1）Kaiyi Mao，天野英晴，堤聡，Vasutan Tunbunheng：“ハードコンシャス記述:ハードウェア化を目的としたC言語記述スタイル”，信学技報，107，419，pp.101-106（Jan. 2008）
2）瀬戸謙修，田中輝明，佐々木俊介，小松聡，藤田昌宏：“ループ融合を利用した複数のforループからのパイプラインハードウェア合成”，情処学研報，2010-SLDM-146，4，pp.1-6（Sep. 2010）
3）NEC，変動する株価データなどをリアルタイムに分析できるハードウェア設計技術を開発，http://www.nec.co.jp/press/ja/1109/2801.html

原　祐子（はら　ゆうこ）　2006年，名古屋大学工学部卒業．2008年，同大学大学院情報科学研究科博士課程前期課程修了．2010年，同大学同研究科博士課程後期課程修了．同年，日本学術振興会特別研究員(PD)．立命館大学，米国・カリフォルニア大学アーバイン校，および，ドイツ・カールスルーエ工科大学客員研究員．2012年，奈良先端科学技術大学院大学助教．2014年，東京工業大学大学院理工学研究科准教授に就任し，現在に至る．主に，組込みシステムの設計自動化に関する研究に従事．博士（情報科学）．