デバイスの記事

論理合成のトラブル・シューティング(中編)

—“Zen of RTL”

Jack Marshall

前回に続いて今回も，論理合成ツールによって回路を合成した結果，所望のタイミング仕様を満たしていなかった場合の対処法について解説する．今回は，DesignWareコンポーネントや制御信号，ファンアウトに関係するクリティカル・パスを取り扱う．なお，本記事ではASIC設計で標準的に利用されている米国Synopsys社の論理合成ツール「Design Compiler」の場合を例に説明する．（編集部）

本記事の前編（本誌2003年5月号，pp.91-101を参照）では，タイミングに関するいくつかのトラブル事例について説明しました．引き続き今回も，タイミングに関する別のトラブル事例（DesignWare，制御信号，ファンアウトに関するトラブル）を紹介します．

1．DesignWareが効果的に使われているか？

Design Compilerのタイミング・レポートを見てください．クリティカル・パスの中にDesignWareコンポーネントが含まれていますか？ クリティカル・パスの信号遅延のうちのどの程度がDesignWareコンポーネントに起因していますか？ そこではどのタイプのDesignWareコンポーネントが使用されていますか？ その回路は，ほんとうにDesignWareコンポーネントが必要な構造になっていますか？ DesignWareコンポーネントは，どのようにしてその回路にマッピングされたのでしょうか？

●演算器に多様な回路を割り当てるDesignWare

図1はDesignWareコンポーネントが含まれているクリティカル・パスのタイミング・レポートの例です．どの

タイミング・レポート

Point		Incr	Path
ACU/ALU/sub_343/B[7] (ALU_DW01_sub_17_0)	← DesignWareコンポーネント	0.00	3.64 r
ACU/ALU/sub_343/U194/Y (NAND2BX4)		0.23	3.87 r
ACU/ALU/sub_343/U204/Y (NAND4BX4)		0.18	4.05 f
ACU/ALU/sub_343/U117/Y (NAND4BX2)		0.19	4.24 r
ACU/ALU/sub_343/U186/Y (NAND4BX4)	コンポーネント#1	0.21	4.45 f
ACU/ALU/sub_343/U49/Y (NOR2X4)		0.05	4.50 f
ACU/ALU/sub_343/U160/Y (NOR2X4)		0.09	4.59 r
ACU/ALU/sub_343/U143/Y (MXI2X4)		0.28	4.87 r
ACU/ALU/sub_343/DIFF[11] (ALU_DW01_sub_17_0)		0.00	4.87 r
ACU/ALU/lt_353/B[11] (ALU_DW01_cmp2_17_0)	← DesignWareコンポーネント	0.00	4.87 r
ACU/ALU/lt_353/U55/Y (NAND2BX4)		0.07	4.94 f
ACU/ALU/lt_353/U71/Y (NAND3X4)		0.10	5.04 r
ACU/ALU/lt_353/U67/Y (NAND2X4)	コンポーネント#2	0.07	5.11 f
ACU/ALU/lt_353/U52/Y (NOR2X4)		0.10	5.21 r
ACU/ALU/lt_353/U43/Y (AOI21X4)		0.09	5.30 f
ACU/ALU/lt_353/U48/Y (OAI21X4)		0.14	5.44 r
ACU/ALU/lt_353/LT_LE (ALU_DW01_cmp2_17_0)		0.00	5.44 r
...			

〔図1〕
DesignWareコンポーネントが使われた場合のタイミング・レポートの例
どのDesignWareコンポーネントを使用するかの選択は，論理合成の最適化処理のときに行われる．

DesignWareコンポーネントを使用するかの選択は，論理合成の最適化処理のときに行われます．

Design Compilerが設計データをコンパイルするとき，**図2**に示すようにRTL記述の中の演算器に対して等価な機能のDesignWareコンポーネントを割り当てる段階があります．Design Compilerのこの機能のおかげで，ASIC設計者は，単一機能の演算器（加算器，乗算器，比較器など）に対応した複雑な回路構成の問題に悩まされなくてすみます．

もちろん，あなたが独自の加算器を使用したければ，それでもかまいません（それぞれの演算器に対して，さまざまな構成の回路があらかじめ用意されているので，設計者が独自に設計することはあまりないが…）．例えば，特別なライセンスがなくても，二つの基本的な加算器アーキテクチャ（リプル・キャリ，キャリ・ルックアヘッド）は利用できます．

あなたの所属するグループがDesignWare Foundationのライセンスを持っていれば，さらにいくつかの加算器アーキテクチャ（高速キャリ・ルックアヘッド加算器，Brent-Kung加算器，条件付きのsum加算器，リプル・キャリ・セレクト加算器）を利用できます．

●DesignWare部品はどのように選択されるのか

DesignWareコンポーネントを選択するプロセスは2段階からなっています．第1段階では，RTL記述から推定される演算器（加算器や減算器，加減算器など）が認識され，これに対して適切なDesignWareコンポーネントが選ばれます．第2段階では，それぞれの演算器に対して，ライブラリのスタンダード・セルを用いてさまざまなアーキテクチャの回路が合成されます．また，それらがタイミング要件を満たしているかどうかテストします．

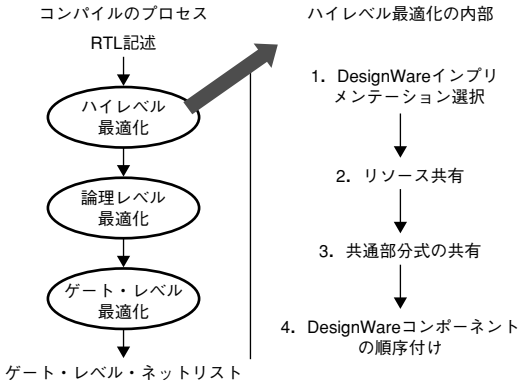
DesignWareコンポーネントの選択は最初に，つまりゲート・レベルの最適化処理が行われるより前に実施されます．そのため，不完全な（ややあいまいな）情報に基づいて選択の決定が行われます．デフォルトの状態では，タイミング要件を満たすもっとも小さい回路が選択されがちです．このため，加算器ではリプル・キャリがよく利用されます．

この選択のプロセスについては，ユーザが最適化パラメータを設定することで，無効にすることもできます．これらのパラメータには“constraint_driven（制約優先）”，“area_driven（面積優先）”，“none（優先なし）”があります．デフォルトの状態では“constraint_driven”になっています．

●DesignWare部品が遅延の原因となった場合の対策

クリティカル・パスが所望のタイミング要件を満たしておらず，そのパスに含まれるDesignWareコンポーネントが信号遅延の大きな原因となっている場合，以下のようないくつかの対策が存在します．

- **DesignWareコンポーネントのインプリメンテーション（合成後の回路）をより速いものに変更する．**



〔図2〕Design Compilerの最適化処理

Design Compilerが設計データをコンパイルするとき，RTL記述の中の演算器に対して等価な機能のDesignWareコンポーネントを割り当てる段階がある．

```
current_design alu
report_resources
```

```
*********************************
Report : resources
Design : alu
Version: 2001.08
Date   : Wed Jan  1 14:33:56 2003
*********************************
Resource Sharing Report for design alu
============================================================================
|          |           |             | Contained |                         |
|Resource  | Module    | Parameters  | Resources | Contained Operations    |
============================================================================
| r70      | DW01_add  | width=8     |           | add_1 add_2 add_3       |
| r79      | DW01_add  | width=8     |           | add_8 add_9 add_10      |
============================================================================
Implementation Report
============================================================================
|          |           | Current     | Set                               |
| Cell     | Module    | Implementation | Implementation                 |
============================================================================ ==
| r70      | DW01_add  | rpl         |                                   |
| r79      | DW01_add  | rpl         |                                   |
============================================================================
```

もっとも遅い回路

〔図3〕DesignWareのリソース・レポートの例

report_resourcesを実行した例．このレポートの内容は，current_designに対してのみ有効である．