論理合成によるFPGA設計に関する検討

指導教官 市川周一　　学籍番号 941708　瀧川正史

1　背景と目的

市川研究室ではField Programmable Gate Array (FPGA) を用いて再構成論理回路の研究を行ってきた [1]．論理設計にはハードウェア記述言語 VHDL と Synopsys 社の論理合成システム Design Compiler による自動化設計を利用するが，FPGA には特有の構造があるため現状では論理合成で良い結果を得ることが難しい．本研究の目的は，FPGA 設計に論理合成を適用して得られる論理回路の品質を評価し，さらに品質の良い回路を得る利用方法を調査することである．

2　FPGA の構造と設計・実装方法

FPGA の内部構造は各社異なるが，基本構造に関しては類似している．そこで本研究では，OPERL ボード [1] で採用している Lucent 社の OR2C シリーズ [2] を例として評価を行う．

OR2C FPGA 内部は，プログラム可能な論理セル (PLC) の 2 次元配列になっている．各 PLC は機能ユニット (PFU) と配線資源からなり，PFU は 19 本の入力と 6 本の出力を持つプログラム可能な論理回路になっている．PFU は多くの動作モードを持ち，例えば 4 ビット加算器として利用したり，16 × 2 ビットの SRAM として利用することができる．SRAM を Look Up Table(LUT) として利用すれば，任意の 4 入力組合せ論理回路として利用可能である．一般に論理回路は基本論理回路 (TTL 技術では NAND) を組み合せて構成するが，FPGA では，PLC という複雑な論理単位を組合せて構成することが特徴である．FPGA の特徴を生かした設計手法をとらなければ，良い結果は得られない．

実際の論理設計には幾つかの方法がある．最も簡単な方法は VHDL を用いて機能記述を行い，実装を論理合成に依存する方法である．論理ゲートで実装する方法 (方法 1) と FPGA 向きの LUT で実装する方法 (方法 2) がある．別な方法としては，FPGA に依存したライブラリを直接指定して論理を記述する方法 (方法 3) がある．これは論理図を書くのと同じで，人間が実装まで立ち入った指定を行う場合に有効である．

こうして論理が生成された後，実際の FPGA チップの品種に合わせて写像 (テクノロジ・マッピング) を行い，チップ上の配置と配線を決定する (P&R)．マッピングまで行うと FPGA チップ上での回路規模が確定し，P&R が終了すると回路の動作速度が判明する．

3　実験内容

本研究ではビット計数回路を評価対象とし，8 ビットの入力と 4 ビットの出力，16 ビットの入力と 5 ビットの出力，32 ビットの入力と 6 ビットの出力を持つ回路を設計する．ビット計数回路は入力ビットのうちの 1 の数を出力ビット幅の 2 進数で出力する回路である．この 3 通りのビット計数回路を以下の方法で実装し，回路規模と最大遅延を評価する．

まず，2章で述べた方法 1 で設計した回路を BEBC(LOG)，方法 2 の結果を BEBC(LUT) とする．また，ビット計数回路は 1 ビット全加算器を組み合わせることにより実現可能である．Lucent 社から提供された全加算器ライブラリを利用して方法 3 で設計した回路を以降 FABC と呼ぶ．FABC はビット計数の定義に忠実にしたがって Carry Save Adder 方式で設計された回路である．OR2C FPGA は構造上 4 ビット単位の処理回路を実装するのに適しているので，それを利用して方法 3 で設計した回路 LUT_ADD4_BC(図 1) も考える．この回路では入力を 4 ビット単位に分割し，それぞれを 4 入力 3 出力のビット計数回路の入力とし，その出力を 4 ビット加算器で加える．LUT_ADD4_BC は FPGA の特性を最大限考慮しているため最も良い評価が得られると予想される．

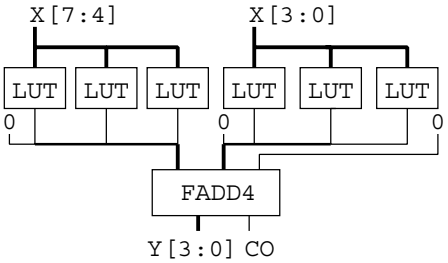

図 1：LUT_ADD4_BC (8 入力)

論理合成システムを使って回路規模と最大遅延を見積った結果は図 2，図 3 の通りである．論理合成後の回路規模は FPGA 内の PLC 使用数の見積り値，最大遅延は論理合成システムが算出した見積り値である．マッピング後の回路規模 (図 4) は，実際に FPGA 中で使用される PLC 数である．回路規模の見積り値と現実が異なるのは，1 つの PLC 内で利用できる 4 入力 LUT は 4 つまでといった実装上の制限があるからである．

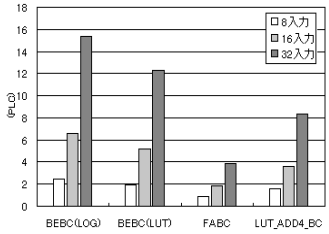

図 2：論理合成後の回路規模

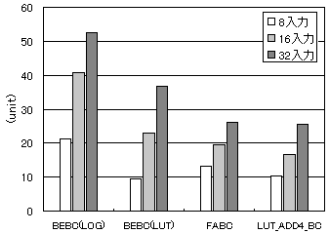

図 3：論理合成後の最大遅延

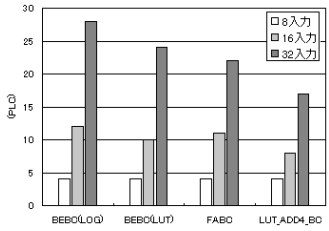

図 4：マッピング後の回路規模

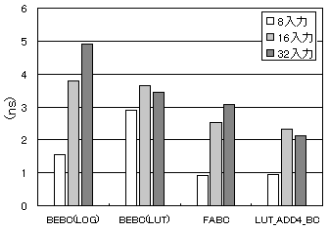

図 5：P&R 後の最大遅延

4　まとめと今後の課題

BEBC(LOG) は回路規模と最大遅延が最大になっている (図 4，図 5)．また BEBC(LUT) は BEBC(LOG) と実装方法が異なるだけだが，回路規模と最大遅延が改善されている (図 4，図 5)．FPGA 設計に方法 1 が適しておらず，実装に LUT を用いると良い結果が得られることがわかる．標準的な方法で設計された回路の FABC は BEBC(LUT) と回路規模では大きな差がない．最大遅延も 8 入力で 0.32 倍，16 入力で 0.70 倍，32 入力で 0.90 倍と回路の規模が大きくなるにしたがって差がなくなっている．また FABC は LUT_ADD4_BC より回路規模，最大遅延が共に劣っている．これらことから通常の設計方法は FPGA 設計に適していないことがわかる．予想通り LUT_ADD4_BC は最も良い評価が得られた．しかし 32 入力の BEBC(LUT) と LUT_ADD4_BC では回路規模で 0.70 倍，最大遅延で 0.62 倍の程度の改善である．回路の規模と遅延に余裕のない場合は FPGA の特性を考慮し方法 3 で設計しなければならないが，規模と遅延に余裕がある場合は方法 2 を用いて設計労力を減らすことが有効であろう．

上述のように方法 1，2 は設計労力が小さいかわりに回路品質が下がる．方法 3 では回路品質が良いかわりに多大の設計労力を要する．しかし方法 3 において一部の回路を方法 2 で設計すれば，大きく回路品質を下げることなく設計労力を低減できると期待できる．例えば，LUT_ADD4_BC(図 1) の LUT 部分 (4 入力ビット計数回路) は case 文を利用 (場合分け) して機能記述できる．また論理合成で 4 ビット加算器の部分を “Y[3:0] <= (’0’ & LUTOUT[2:0]) + (’0’ & LUTOUT[2:0])” のように機能記述することも可能である．実際に 8 入力の LUT_ADD4_BC をこの方法で実装し，回路規模と最大遅延が方法 3 と同様になることを確認した．

今後の課題は，実際に本研究室で利用する各種の回路を，今回の結果を生かして設計することである．

参考文献

[1] 市川周一，島田俊夫：パーソナルコンピューティング指向の動的再構成可能 PCI カード，第 5 回 FPGA/PLD Design Conference & Exhibit，pp. 269-277(1997) .

[2] ルーセント・テクノロジー半導体販売株式会社：$ORCA^{TM}$ (Optimized Reconfigurable Cell Array) OR2CxxA(5.0V) と OR2TxxA(3.3V) シリーズ・フィールド・プログラマブル・ゲートアレイ　データ・シート，東京 (1997).