ISSN 2186-5647

ループ数削減指向バインディングの データパスインテンシブ回路テスタビリティの評価

日大生産工(院)　○森田 菜留美
日大生産工　　細川　利典

1. はじめに

近年，半導体技術の急速な進歩により，大規模集積回路(Large Scale Integrated circuits :LSI)が大規模化し，回路が複雑化してきている．従来 LSI はレジスタ転送レベル(Register Transfer Level: RTL)での設計が主流であるが，LSI の大規模化に伴い， RTL での設計が困難になってきている[1]．それゆえ，RTL より抽象度の高い動作レベルで設計された記述を RTL 回路に変換する技術である動作合成[2]が注目されている．

一方，LSI の大規模化，複雑化に伴い，従来の LSI のスキャンテストのコストは劇的に増加している．そのため，動作合成の段階で非スキャンテストを基本とした順序回路のテスト容易性を考慮することが必要とされている[2]．

LSI は順序回路であり，順序回路中にはフィードバックループ[3]が存在するため，テスト生成が困難である[1]．順序回路のテスト容易化の設計手法として，一部のレジスタをスキャン可能なレジスタ(スキャンレジスタ)に置き換える部分スキャン設計[4]が提案されている．部分スキャン設計を指向したデータパスのテスト容易化動作合成法として，これまで多くの手法が提案されている[2][3]．

本論文では，文献[2]で提案された無閉路部分スキャン設計を指向したデータパスのバインディング手法[2]の実装を検討し，回路中のセルフループ数を削減した回路のテスト容易性の評価結果を報告する．また，文献[2]の手法をベースに，ループ段数削減指向バインディング法を提案し，回路中の段数の深いループ数を削減した回路のテスト容易性の評価結果を報告する．

2. 動作合成

動作合成とは，動作記述から RTL 回路を合成する技術である[2]．動作合成にはグラフ生成，スケジューリング，バインディング，RTL 回路記述生成の 4 つのステップがある(図 1)．

グラフ生成では与えられた動作記述をグラフで表現する．グラフにはデータフローグラフ(Data Flow Graph :DFG)[1]を用いる．スケジューリングでは各演算の依存性を保ちながら，各時刻に演算操作を割当てる．バインディングではスケジューリング済みの DFG (Schedule Data Flow Graph :SDFG)を基に各演算操作や変数に具体的な演算器やレジスタを割当てる．RTL 回路記述生成は割当てられた演算器やレジスタ間を接続し，RTL 回路を生成する．

2.1. スケジューリング

スケジューリングの基本的な手法として、ASAP(As Soon As Possible)[4]や ALAP(As Late As Possible)[4]がある．ASAP は可能な限り早い時刻に演算操作を割当てるスケジューリング手法である．ALAP は可能な限り遅い時間に演算操作を割当てるスケジューリング手法である．ASAP および ALAP では，使用リソース数を考慮していないため，多くの演算操作を同時刻に割当て，面積のオーバーヘッドが大きくなることがある．面積効率のよい SDFG を得るアルゴリズムとして FDS(Force Directed Scheduling) [5]が提案されている．本論文では，スケジューリング手法に FDS を用いる．図 2 に SDFG の例を示す．

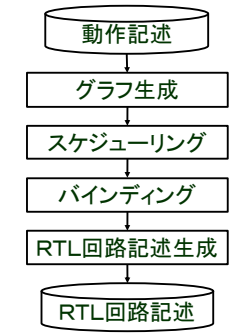


図 1. 動作合成の流れ

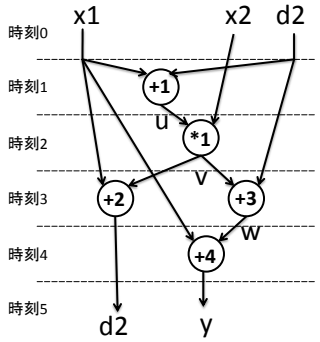


図 2. スケジューリング済み DFG

2.2. バインディング

SDFG において，各変数をレジスタに割当てる操作や，各演算を演算器に割当てる操作をバインディングという．SDFG から演算操作使用時刻や変数使用時刻を表すライフタイムを求め，ライフタイムが衝突しないように演算操作を演算器に，変数をレジスタに割当てる．

バインディングの基本的な手法として、レフトエッジアルゴリズム(Left Edge Algorithm LEA)[4]が提案されている．LEA は面積最小化を考慮したバインディング手法である．図 3 に図 2 の SDFG を LEA を用いてバインディングを行った RTL データパス回路を示す．

3. フィードバックループ

順序回路にはフィードバックループが存在する．あるレジスタから演算器や他のレジスタを通って再び同一のレジスタへ経路があるものをフィードバックループという．その経路内で通るレジスタの数をループ段数とする．フィードバックループはセルフループとグローバルループがある．セルフループはループ段数が 0 で，同一のレジスタの出力から入力までの経路が存在し，その経路中には他のレジスタが存在しないループである．グローバルループはループ段数が 1 以上で，同一のレジスタの出力から入力までの経路が存在し，その経路中に他のレジスタが存在するループである．

フィードバックループは順序回路のテスト生成を困難にする．フィードバックループが存在することで順序深度[7]が大きくなり，一般的な順序回路のテスト生成方法で

Evaluation of binding methods for testability
to reduce the number of loops of datapath intensive circuits

Narumi MORITA and Toshinori HOSOKAWA

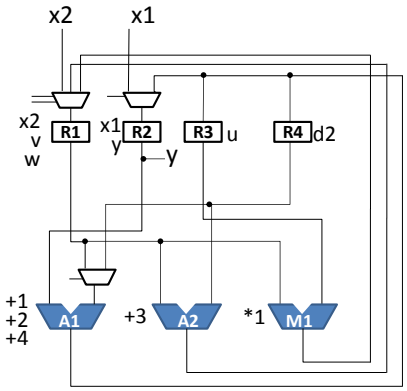
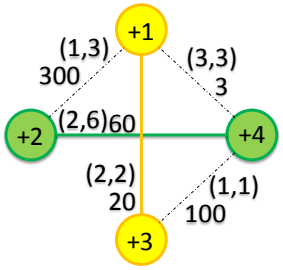


図 3. RTL データパス回路



(最短経路長,最短経路時間)

$K_1 = 100$, $K_2 = 10$, $K_3 = 1$

図 4. 演算両立グラフ(文献[2]の方法)

ある時間展開モデル[7]の時間展開数が有限化できない．順序回路を無閉路構造にすることで時間展開数を有限化でき，テスト容易化されることが知られている[7]．

図 3 の RTL データパス回路では下記の 4 個のセルフループが存在する．

・R1→A2→R1
・R1→M1→R1
・R2→A1→R2
・R4→A1→R4

また，グローバルループはループ段数が 1 段のループが

・R1→A1→R3→M1→R1
・R1→A1→R4→A2→R1

の 2 個，2 段のループが

・R1→A1→R4→A1→R3→M1→R1

の 1 個，3 段のループが

・R1→A1→R2→A1→R4→A1→R3→M1→R1
・R1→A1→R4→A1→R2→A1→R3→M1→R1

の 2 個存在する．図 3 の回路では最大のループ段数は 3 である．

4. テスト容易化バインディング

4.1. セルフループ削減指向バインディング(文献[2]の方法)

テスト容易化バインディングの手法として、無閉路部分スキャン設計に基づくデータパスのテスト容易化バインディング手法[2]がある．あるフィードバックループを削除するためには，そのフィードバックループ内のレジスタを少なくとも一つはスキャンレジスタに置き換える必要がある．セルフループは他のレジスタを経由しないループであるため，セルフループ内のレジスタはスキャンレジスタ[4]に置き換える必要がある．よって，セルフループが多い場合，スキャンレジスタ数も多くなる．手法[2]では，RTL データパス回路内に発生するセルフループをできる限り削減し，残ったセルフループやグローバルループに対してはスキャンレジスタに置き換えることで，少ないスキャンレジスタ数で無閉路構造を構成し，テスト容易化を実現する．

この手法では演算両立グラフ，レジスタ両立グラフでクリーク数最小のクリーク分割を行うことで面積最小化を実現する．また，両立グラフの各辺や各頂点に重みをつけ，重み和最小クリーク分割を行うことで，面積最小化かつテスト容易化を考慮したバインディングを行う．演算両立グラフは無向グラフであり，頂点は演算操作で，演算操作が演算器を両立可能な場合に頂点間に辺が存在する．同じ種類の演算でかつ演算操作のライフタイムが重なっていない場合，演算器を両立可能である．レジスタ両立グラフは無向グラフであり，頂点は変数で，変数がレジスタを両立可能な場合に頂点間に辺が存在する．変数のライフタイムが重なっていない場合，レジスタを両立可能である．

両立可能な 2 つの演算間に経路が存在すると，その演算の共有によってループができる．その経路上の少なくとも一つの変数はスキャンレジスタに割当てられる変数(スキャン変数)となる．経路の長さ(経路上の変数の数)が大きいほど，スキャン変数の選択肢は多くなる．よって，演算両立グラフの辺の重みは，最短経路長が長いほど小さな重みをつける．以下に演算両立グラフの重みのつけ方を記す．

（1）演算間に経路がないとき
　（辺の重み）＝0
（2）2 つの演算間に経路があるとき
　（辺の重み）＝ 最短経路時間×$K_{\text{最短経路長}}$
※$K_{\text{最短経路長}}$は最短経路長が大きいほど小さい値
　例　K_3=1，K_2=10，K_1=100

重みをつけた演算両立グラフに対し，重み和最小クリーク分割を行い，それを基に演算共有グラフ[2]を生成する．このとき，各クリークの重みはクリーク内の重みの総和とする．図 2 の SDFG から生成した演算両立グラフに対し，重み和最小クリーク分割した結果を図 4 に示す．

図 4 の演算両立グラフにおいて，(+1,+3)の辺の重みを考える．+1 と+3 の演算間には図 2 の SDFG から+1→*1→+3 の最短経路が存在することがわかる．この経路間には u，v の 2 つの変数が存在するため最短経路長は 2 となる．また，2 時刻に渡って経路があるので，最短経路時間が 2 となる．ここで先に記した(2)の重みを与えると，+1→+3 の辺の重みは $2\times K_2 = 2\times 10 = 20$ となる．同様にして他の辺にも重みを与え，重み和最小クリーク分割を解く．図 4 では，加算器 A1(+2,+4)，加算器 A2(+1,+3)と割当てた．図 4 の重み和最小クリーク分割結果から生成した演算共有グラフを図 5 に示す．演算共有グラフは有向グラフである．頂点は外部入出力と演算器で，辺は変数で外部入出力と演算器間の経路を示している．演算共有グラフを基に，レジスタ両立グラフの重みを与え，変数をレジスタへ割当てる．

演算共有グラフで両立可能な 2 つの変数間に経路が存在し，それらの変数を 1 つのレジスタで共有すると，変数間の経路は共有したレジスタを通るループとなる．よって，その経路内のいずれかの変数はスキャンレジスタに割当てなければならない．特に，経路が隣接している場合は，セルフループが生じるので，そのレジスタはスキャンレジスタに割当てなければならない．また，演算共有グラフでセルフループを構成している変数もスキャンレジスタに割当てる必要がある．レジスタ両立グラフの頂点の重みは演算共有グラフで変数がセルフループを構成しているときに大きな重みをつける．レジスタ両立グラフの辺の重みは演算共有グラフで変数間に経路が存在するときに大き

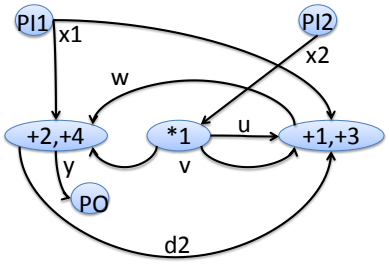


図 5.　演算共有グラフ(文献[2]の方法)

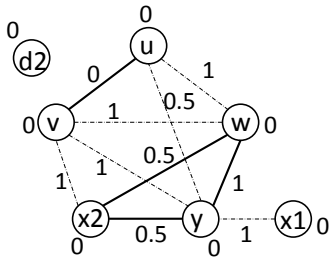


図 6.　レジスタ両立グラフ(文献[2]の方法)

な重みをつける．また，経路が隣接しているときはさらに大きな重みをつける．経路が隣接しているとは，ある 2 つの変数間に経路が存在しかつその変数間に他の変数がない場合をいう．以下にレジスタ両立グラフの重みのつけ方を記す．

《辺の重み》
(1)　演算共有グラフで対応する変数が隣接しているとき　　：1
(2)　隣接していないとき　　：0.5
(3)　変数間に経路が存在しない：0
《頂点の重み》
(1)演算共有グラフで対応する変数がセルフループを構成しているとき　　：1
(2)(1)以外のとき　　：0

演算両立グラフと同様に，重みをつけたレジスタ両立グラフに対し，重み和最小クリーク分割を行う．このとき，各クリークの重みはクリーク内の重みの最大値とする．図 2 の SDFG から生成したレジスタ両立グラフに対し，重み和最小クリーク分割した結果を図 6 に示す．

図 6 のレジスタ両立グラフにおいて，(u,w)の辺と頂点の重みを考える．図 5 の演算共有グラフより，u と w は演算共有グラフ内でセルフループを構成していない．よって，u と w の頂点の重みを 0 と与える．また，u と w の間には最短経路 u→+1,+3→w の経路が存在する．u と w は隣接しているので，(u,w)の辺の重みは 1 となる．同様にして，すべての頂点と辺に重みを与え，重み和最小クリーク分割を解く．図 6 では，レジスタ R1(x2,w,y)，レジスタ R2(x1)，レジスタ R3(u,v)，レジスタ R4(d2)と割当てた．

図 7 にセルフループ削減指向バインディングを行った結果の RTL データパス回路を示す．図 7 の RTL データパス回路内のセルフループは R1→A1→R1 ，R3→A2→R3 の 2 個に削減された．グローバルループはループ段数が 1 段のループが
・R1→A1→R4→A2→R1
・R1→M1→R3→A1→R1
・R1→M1→R3→A2→R1
・R3→A1→R4→A2→R3

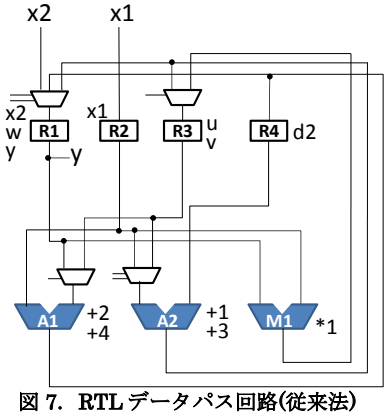
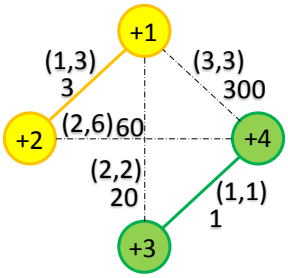


図 7.　RTL データパス回路(従来法)



(最短経路長,最短経路時間)
K₁ = 1 , K₂ = 10 , K₃ = 100

図 8.　演算両立グラフ(提案手法)

の 4 個，2 段のループが
・R1→A1→R4→A2→R3→A1→R1
・R1→A1→R4→A2→R3→A2→R1
・R1→M1→R3→A1→R4→A2→R1
の 3 個存在する．よって 2 段のループが最大のループ段数となった．

4.2.　ループ段数削減指向バインディング(提案手法)

文献[2]の方法は無閉路パーシャルスキャン設計を前提としているため，セルフループ数に着目し，それを削減することをテスタビリティ向上の指針としてバインディングを行うものであった[2]．セルフループを削減した回路において，隣り合う変数を別のレジスタに割当てることで，ループ段数が大きくなると考えられる．しかしながら，パーシャルスキャン設計を前提としない場合，回路のループ段数が大きい回路はテスタビリティが低いと考えられる[3]．そこで，文献[2]の重み和最小クリーク分割問題の重みのつけ方において，ループ段数を削減できるように重みを与えることで，ループ段数を削減する手法を提案する．

従来の両立グラフの重みのつけ方はセルフループを削減するため，セルフループができる場合の重みを大きくしていた．演算両立グラフでは最短経路長が 1 の場合セルフループができ，それより大きい場合は経路長に伴ってループ段数が大きくなると考えられるので，4.1.で記した演算両立グラフの重みの(2)を以下のように変更する．

（2）2 つの演算間に経路があるとき
（辺の重み）＝ 最短経路時間×$K_{\text{最短経路長}}$
※$K_{\text{最短経路長}}$は最短経路長が小さいほど小さい値
　例　K_1=1，K_2=10，K_3=100

演算両立グラフの各辺にこの重みを与える．図 8 に演算両立グラフの重み和最小クリーク分割，図 9 に演算共有グ

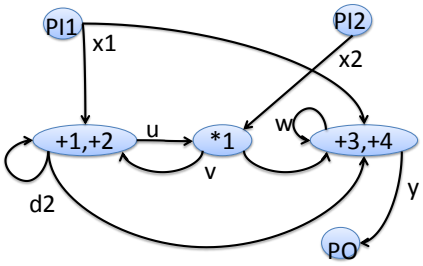


図 9.　演算共有グラフ(提案手法)

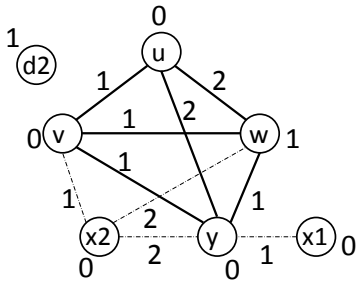


図 10.　レジスタ両立グラフ(提案手法)

ラフの例を示す．図 8 では，加算器 A1(+1,+2)，加算器 A2(+3,+4)を割当てた．

また，レジスタ両立グラフでは演算共有グラフで変数が隣接しているときにセルフループができるため，その場合の重みを大きくしていた．ここでは経路間の変数の数が多い場合，ループ段数が大きくなると考えられるので，4.1. で記したレジスタ両立グラフの辺の重みの(2)を以下のように変更する．

《辺の重み》
(2)　隣接していないとき：経路間の演算器数

レジスタ両立グラフの各辺にこの重みを与える．図 10 にレジスタ両立グラフの重み和最小クリーク分割の例を示す．図 10 では，レジスタ R1(x1)，レジスタ R2(x2)，レジスタ R3(u,v,w,y)，レジスタ R4(d2)と割当てた．

図 11 に提案手法のループ段数削減指向バインディングを行った結果の RTL データパス回路を示す．図 11 の RTL データパス回路内のセルフループは
・R3→A1→R3
・R3→A2→R3
・R3→M1→R3
・R4→A1→R4
の 4 個となった．グローバルループはループ段数が 1 段のループが
・R3→A1→R4→A1→R3
・R3→A1→R4→A2→R3
の 2 個存在する．よって 1 段のループが最大のループ段数となり，文献[2]の方法と比較し，ループ段数とループ数を削減することができた．

5. 実験結果

本論文では，図 2 の SDFG に対して，LEA と文献 [2] の方法と提案手法でバインディングを行い，RTL データパス回路を生成した．

表 1 に LEA と従来法[2]と提案手法のバインディングで生成した 3 つの回路のセルフループ数，グローバルループ数，最大ループ段数を示す．従来法は，LEA と比較しセルフループが削減できたのに対し，グローバルループ数が増加した．提案手法では，セルフループ数は LEA に比較

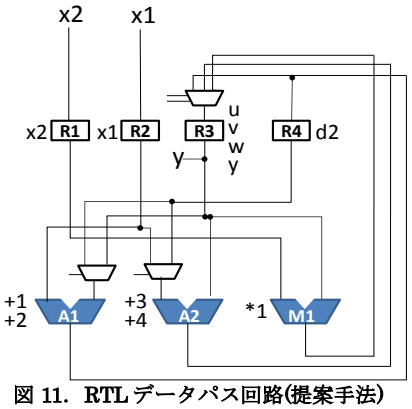


図 11.　RTL データパス回路(提案手法)

表 1.　ループ数

	セルフループ数	グローバルループ数	最大ループ段数
LEA	4	5	3
文献[2]	2	7	2
提案手法	4	2	1

し削減はできなかったが，グローバルループ数と最大ループ段数を削減することができた．文献[2]の方法と比較すると，セルフループ数は増加したが，グローバルループ数と最大ループ段数を削減することができた．

6. おわりに

本論文では一つの回路を対象にループ段数削減バインディング手法を提案した．本論文で使用した回路では，ループ段数を削減することはできたが，他の回路においても検証が必要である．今後の予定は，大規模な回路を対象に実験を行い，更に，ループ段数を削減することで，テスタビリティを評価する．

参考文献

1) 藤原 秀雄,"ディジタルシステムの設計とテスト", 工学図書株式会社, 2004.
2) 高崎智也,井上智生,藤原秀雄,"無閉路部分スキャン設計に基づくデータパスのテスト容易化高位合成におけるバインディング手法",電子情報通信学会論文誌(DI),Vol.J83-D-I No.2 ,2000.
3) V.Femandez and P.Shchez,"Partial Scan High-Level Synthesis", 1996 European Design and Test Conference (ED&TC '96),1996.
4) Giovanni De Micheli, "SYNTHESIS AND OPTIMIZATION OF DIGITAL CIRCUITS", McGraw-Hill,inc, 1994.
5) Daniel D, Gajski, "High-Level Synthesis", Kluwer Academic Pub, 1992.
6) R.Camposano and W.H.Wolf, "High-Level VLSI Synthesis", Kluwer Academic Publishers, 1991.
7) P.Muth, "A Nine-Valued Circuit Model for Test Generation," IEEE Trans. on Computers, vol. C-25,no.6, pp.630-636, June 1976.