設 計 自 動 化 84－2
（１９９７．５．２３）

論理合成を用いた低消費電力回路への自動変換手法

中木琢夫　柿沼義則
ＮＥＣテレコムシステム
水嶋紀子　河原林政道　河村一
ＮＥＣ

概要

ＬＳＩの低電力化のため、フリップフロップがデータを更新しない間、更新するか否かを制御する信号を用いてクロックを止める回路構成(以降　ゲーテッドクロックと呼ぶ)を用いている。今回我々は、この制御信号(以降 Enable と呼ぶ)に着目し、Enable オフ時には自フリップフロップの出力をデータ入力に帰還する回路(以降 Recirculation Logic と呼ぶ)を、クロック側に制御回路を挿入したゲーテッドクロック回路に自動変換する手法を提案する。

本手法により、ＨＤＬ等を用いたＲＴＬ設計段階では低電力化を意識せず、１相同期回路設計が可能であり、さらに複数の Recirculation Logic に使用されるセレクタが一つのクロック制御回路に変換されるため回路規模が削減できる。

An optimization technique to low power circuit using logic synthesis

Takuo Nakaki ,Yoshinori Kakinuma
NEC Telecom Systems Corporation
Noriko Mizushima　Masamichi Kawarabayashi　Hajime Kawamura
NEC Corporation

Abstract

The power consumption of the circuit can be reduced by the gated clock technique to stop the clock signal at each flipflop while it must keep the previous value.
We propose a new methodology which translates the recirculation logic circuits into the gated clock circuits automatically. Using our methodology, large-scale circuits can be designed at register transfer level without any consideration of power consumption. Furthermore the reduction of the selectors may cause the reduction of total size of the circuits.

１．開発の背景

１.１．ＬＳＩの大規模化と低電力化技術

ＬＳＩに搭載される論理回路規模はその微細化技術を背景に６年で１０倍の進歩を遂げている。一方チップを搭載するパッケージ熱抵抗の制限から、消費電力を押さえることを余儀なくされているとともに、装置の市場競争力において、ＬＳＩの低消費電力化が求められている。

一方では生産性向上のため VHDL、VerilogHDL などのハードウェア記述言語を用いて、より抽象度の高い設計を行い、論理合成システムを用いて論理回路を自動合成する事が一般的になっている。

ハードウェア記述言語を用いた抽象化設計において低消費電力回路を作るには、アーキ

ハードウェア記述言語を用いた抽象化設計において低消費電力回路を作るには、アーキテクチャレベルで共通化を図る、機能を削減するなどの一般的に行われている全般的な機能の削減と、論理合成システムを用いた局所的な最適化の２通りがある。

前者は基本的には設計対象ごとに手法が異なるが、共通的に用いられる一つの手法として、クロック線に制御回路を入れ、データ入力が変化しないときにクロックを止めるゲーテッドクロック技術がある［1,2］。

一方後者には、論理合成システム内の最適化技術として、ＣＭＯＳ回路の特長を生かし、同じ機能でありながらゲート出力を変化させないような論理ゲートを選択するものがある[3,4,5,6]。

ゲーテッドクロック技術は不必要なクロック線での電力消費がなくなり効果的である[7]。しかし、ゲートレベル回路図での人手設計においてはEnable信号の本数や対象となるフリップフロップの個数の問題があり、かなりの手間がかかる。一方、論理合成による回路の自動生成は不可能で、以下に示す記述例を合成すると、Recirculation Logic が生成される。

```
process(CLK, RSTB)
begin
if RSTB='0' then DFF <= '0';
elsif(CLK'event and CLK='1')then
  if Enable='0' then
    DFF <= INADATA+INBDATA;
  else NULL;
  end if;
end if;
end process;
```

この記述から作られる回路は、Enable が'0'のとき INADATA+INBDATA の結果を信号 DFF に出力し、Enable がそれ以外のときには前の値を保持する。

また、ゲーテッドクロック化を論理合成で実行可能になったとしても単純に行うと、後述するタイミング問題が発生する。

1.2. ゲーテッドクロック回路が持つ問題点

クロック線に制御回路を入れるゲーテッドクロック回路の例を図1に示す。クロック線に入れる制御回路は、入れないときと動作が同じになるようにしなければならず、かつ安全に動作させる以下のタイミング条件がある。

- 制御信号がクロックより遅れていること。
- その制御信号の遅延時間はクロックの半周期区間を越えないこと。

この条件を満たさないとクロックにスパイク信号を乗せたり、クロック自体が細くなってしまい、そのクロックで駆動するフリップフロップを誤動作させてしまう(図2)。

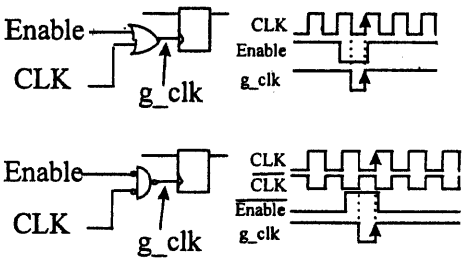


図1　ゲーテッドクロック回路例

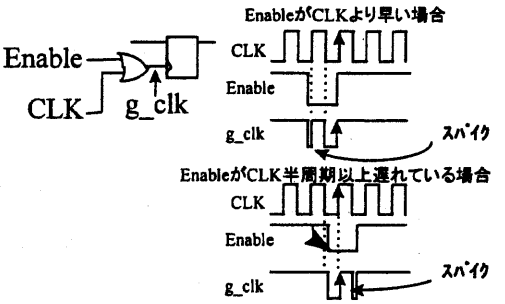


図2　Enable のタイミングミスで誤ったクロックを生成する例

1.3. 対象回路例

我々の設計する伝送装置用ＬＳＩの回路には以下の構成が多く存在する(図3)。

- Enable 信号を複数本生成する部分回路(Pulse Generator)。
- Enable 信号を受け取り、Enable=On 時に動作する部分回路。

伝送路上を流れる情報データに含まれる警報/状態情報を、その位置を示した Enable 信号を用いて処理する構成である。それぞれのフリップフロップは Enable 信号が On 時に論理演算等の処理後のデータを保持し、Off 時にはなにもしない。

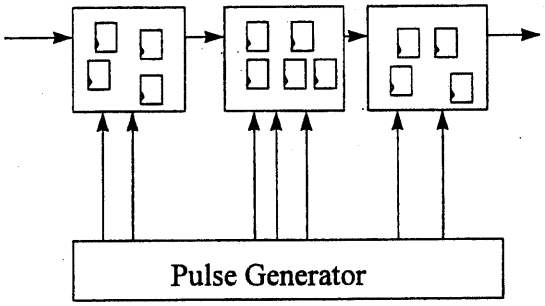


図3　伝送装置回路例

1.4. フリップフロップの消費電力

フリップフロップの消費電力を Spice シミュレータを用いて調べた(表1)。クロックが動作したときに消費する電力は、データが変化していない場合でも最大 145.8 μWと大きい。これを等価なゲーテッドクロック回路に置き換え、クロックを止めればフリップフロップ一つあたり最大 77.0 μW削減できる。

表. 1 FlipFlop(1個当たり)の消費電力

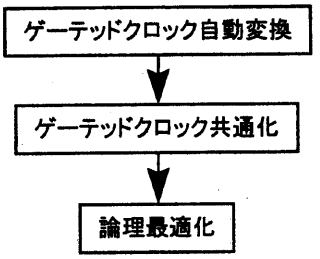
	ｸﾛｯｸ	ﾃﾞｰﾀ	消費電力 (μW)	
			slow条件	fast条件
ﾃﾞｰﾀ変動なし	L→H	L	80.1	135.0
	H→L	H	94.8	145.8
ｸﾛｯｸ停止	H	L→H	18.3	53.6
	H	H→L	40.5	68.8

:クロック総配線容量　63.33pF
:Technology　CMOS
:slow条件
　ﾌﾟﾛｾｽ:worst　温度　:125℃
　電圧　:3.0V
:fast条件
　ﾌﾟﾛｾｽ:best　温度　:−40℃
　電圧　:3.6V

2. 提案手法

2.1. 処理構成

今回我々が提案する手法は、データの更新を制御する Enable 信号に着目し Recirculation Logic をゲーテッドクロック回路へ自動変換し、フリップフロップのクロック線で消費している電力を抑えるものである。以下の順序で変換を行っていく。



2.2. 処理詳細

(1)ゲーテッドクロック回路自動変換

以下の順序で変換処理を行う。

- 論理合成内の最適化で共通化されている組み合わせ回路をフリップフロップ毎に論理展開する。
- フリップフロップのデータ入力に自フリップフロップの出力信号が入力されているかどうかを次の論理関数で表す。

$$f = e \cdot h + \overline{e} \cdot d \quad \cdots\cdots(1)$$
$$f = \overline{e} \cdot h + e \cdot d \quad \cdots\cdots(2)$$

f：フリップフロップのデータ入力
e：Enable 信号
$\overline{e}$：Enable 信号の反転
h：フリップフロップの出力
d：データ信号

フリップフロップのデータ入力に h が含まれていた場合、次のコファクタの計算を行う。以下の式は、(1)及び(2)式の e と $\overline{e}$ に論理‘1’及び論理‘0’を入力することで算出できる。

$$f(e) = 1h + 0d \quad \cdots\cdots(3)$$
$$f(\overline{e}) = 1h + 0d \quad \cdots\cdots(4)$$

まず、（1）式に対する（3）式の判定を行い、ここでf(e)＝hの条件を満たせばゲーテッドクロック化された論理回路の生成を行うが、条件を満たさなかった場合、（2）式に対する（4）式の関係を判定する。ここでf($\overline{e}$)＝hの条件を満たせばゲーテッドクロック化された論理回路の生成を行う。

・Recirculation Logicのセレクタを削除しデータ入力をフリップフロップに直接接続する。

・生成された回路図例を図4に示す。

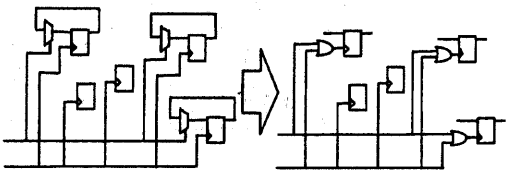


図4 Recirculation Logicからゲーテッドクロック回路への変換例

このORはEnable信号の極性によって他の論理になることもあるが、変換前との動作が等しくかつ1．2．で説明した条件を満たさなければならない。Enable信号が'0'でOnの場合ORしかない。

(2)ゲーテッドクロック共通化

フリップフロップごとに置かれたクロック制御回路を、Enable信号が等しいもので共通化し（図5）、さらに階層をまたいで引き上げ、共通化させる（図6）。

尚、この処理は階層をまたがって最上位階層まで再帰的に処理され、ゲーテッドクロックの個数は極小化される。

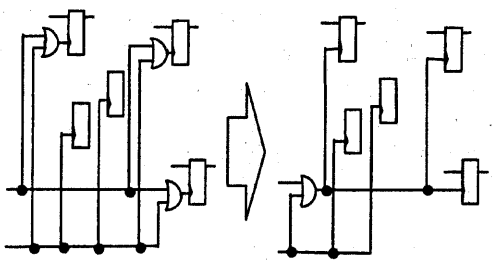


図5 ゲーテッドクロック共通化

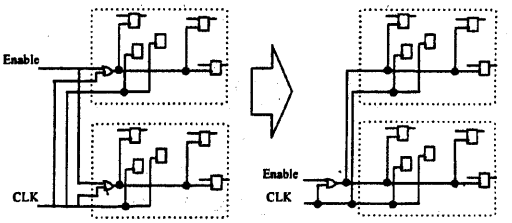


図6 ゲーテッドクロック階層引き上げ共通化

(3)最適化

(1)で論理展開したフリップフロップのデータ入力前段の論理回路を再度最適化し論理圧縮する。

2.3. その他回路に対する機能

各種Enableの条件に対処するため以下の機構を内包している。

・多bitフリップフロップ展開機構

同じEnableでRecirculation Logic構成になっている複数のフリップフロップとEnable制御されていないフリップフロップが混在して入っている多bitフリップフロップが使われていた場合、多bitフリップフロップを展開して認識処理することによりゲーテッドクロック化できる(図7)。本機構は同じEnable信号でRecirculation Logic構成になっているにも関わらず、複数のクロック制御回路になることを抑制できる。

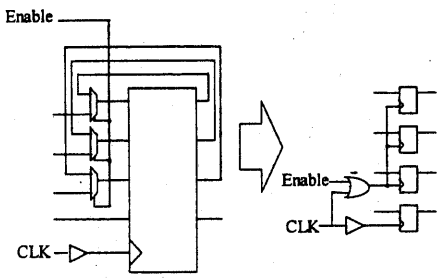


図7　多bitフリップフロップ展開機構

・多段Enable時クロックバッファ制御機構
この制御は複数のEnableでRecirculation Logic構成をとっていたとき、一つ目のEnableで本システムの処理を行ったあとで、2つめのEnableで処理をしてもゲーテッドクロックが誤った構成にならないようにする機構である(図8)。

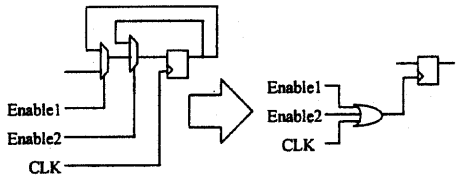


図8　多段Enableクロックバッファ制御例

・Enable異極性制御機構
'0'でOnするRecirculation Logicと'1'でOnするものがあった場合、一方のEnable線にインバータを挿入し1．2．で述べた正しいゲーテッドクロック回路に変換する機構である(図9)。

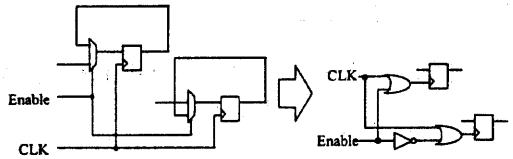


図9　Enable異極性制御例

3. 評価

3.1. 評価回路

本提案を論理合成システムに組み込み、以下の特徴を持つ実開発したデータに対して評価を行い、効果を測定した。

最上位階層：18個のサブモジュールが存在。そのうち13個のサブモジュールをゲーテッドクロック対象回路(サブモジュール内にメモリを含む)とした。

全体の回路規模：約957Kgate。このうち搭載しているメモリの規模は全体規模の3／4である。

メモリの処置方法：メモリにもゲーテッドクロック処理を行い低消費電力化を図る。具体的にはデュアルポートタイプはWriteEnableとWriteCLK、シングルポートタイプはChipSelectとRead/Write用CLKの論理をとって、ゲーテッドクロック回路を構成した。

ランダムロジック部：ランダムロジック部はEnable信号の特定をした後、提案システムを使いゲーテッドクロック回路へ自動変換した。
評価用データは、提案システム処理前は1相同期回路だった。そのため各フリップフロップに到達するデータ線及びEnableはCLKに同期して生成された信号であり、CLKの変化タイミングよりも遅く、かつ、CLKの半周期よりも遅れることはない。

3.2. 評価結果

上記回路の処理前と処理後の比較を行った結果を表2に示す。この表からわかるように、Enable信号によってゲーテッドクロック回路に変換した各サブモジュールは、消費電力が削減できており、さらに回路規模も削減されていることがわかる。また、メモリはメモリ内部の回路構成を変換していないので回路規模の削減はされていないが、EnableないしChipSelectのディセーブル状態によって消費電力の削減が図れていることがわかる。

表 2　提案システム評価結果

Module	処理部分		処理前	処理後	削減率	処理前	処理後
	logic	RAM	消費電力(mW)	消費電力(mW)	(%)	回路規模(Kgate)	回路規模(Kgate)
M1×2		○	126.16	92.75	26.5	32822	32282
M2×2	○	○	226.16	128.18	43.3	84920	82554
M3×2	○	○	181.45	131.64	27.5	43815	42513
M4×2			29.31	29.31	0.0	1536	1536
M5×2	○	○	126.93	87.75	30.9	37941	35988
M6×2			353.38	353.38	0.0	85904	85904
M7×2		○	530.74	278.15	47.6	184866	184866
M8×2	○	○	1134.35	657.05	42.1	347018	342334
M9×2		○	572.00	448.70	21.6	115509	115509
M10×2			26.34	26.34	0.0	22370	22370
M11×2			0.43	0.43	0.0	759	759
			———	———		957460	946615

電圧：3.6[V]時
TECHNOLOGY：CMOS
電圧：3．6［V］
消費電力は各モジュールのピーク値を算出しており、ＬＳＩチップの合計ではない

3.3. 考察

本提案システムによる自動変換後に行った、論理検証及びタイミング検証では機能の等価性、同等なタイミング性能を確認した。従って本提案システムの自動変換手法が問題なく動作していることがわかった。

ただし今回はあらかじめ Enable 信号を意識した理想的な回路を対象にしているが、例えばすでに設計済みの回路を対象にした場合、２．３．で述べた付随機能だけで誤りのないゲーテッドクロック回路ができるかが検討課題である。

例えば、ゲーテッドクロック処理できたフリップフロップとそうでないフリップフロップが接続されていた場合、微小とはいえ遅延差があることから、フリップフロップ間に遅延差分のホールドタイムを保証する遅延バッファが必要になってしまう（図 10）。

この遅延バッファを抑制するため、より網羅的に Recirculation Logic をゲーテッドクロック回路に変換する、もしくは設計段階から Enable による制御構成を考慮するなどの準備が必要である。

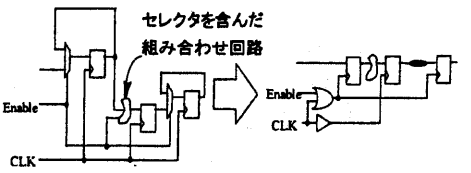


図 10　ゲーテッドクロック化が不可能な回路例

4. まとめ

本報告ではまずＬＳＩの大規模化から回路の低電力化が必要であることに触れ、ゲーテッドクロック技術についてその問題点を整理した。その後扱う伝送装置の回路例と、その中で使われる Recirculation Logic が導かれ

るHDL記述について説明し、フリップフロップの消費電力のうちクロック線で消費される電力が大きいことを示した。

以上をふまえRecirculation Logicをゲーテッドクロック回路に自動変換する手法を提案した。この中で自動変換時にRecirculation Logicを制御するEnable信号に着目して変換し、ゲーテッドクロックの共通化、階層をまたいで引き上げ、最後にCTS用のクロックバッファを配置する手法を説明した。最後に実開発のデータを用いて評価を行い、消費電力が削減されていることから本提案の有効性を示した。

5. 今後の課題

2．2．処理詳細の(1)で述べた一度論理展開する際、論理展開前の回路構造を記憶しておき、ゲーテッドクロック化できない部分回路は論理展開前の状態に戻す処理が必要である。この処理が実現できれば本提案の処理前の最適化済み回路を保存することができ、再度最適化するという手戻りがなくなる。大規模回路の最適化時間を削減できるため効果は大きい。また3．3．考察で述べたゲーテッドクロック処理の網羅性向上も今後の課題である。

参考文献

[1]:C. Papachristou, M. Spinning, and M. Nourani. An Effective Power Management Schame for RTL Design Based on Multiple Clocks. *In Proceedings of the 33th Design Automation Conference Proceedings 1996,Page337-341,1996*

[2]:L. Bonini, P. Sigel, and G. De. Micheli. Saving Power by Syntesizing Gated Clocks for Sequential Circuits. *In Proceedings of IEEE J. of Design and Test of Computers, 1994*

[3]:V. Tiwari, P. Ashar, and S. Malik. Technology Mapping for LowPower. *In Proceedings of the 30th Design Automation Conference,Page74-79, 1993*

[4]:R. Muragi, R. K. Brayton, and A. S. Vincentelli. Decomposition of Logic Functions for Minimum Transition Activity. *In Proceedings of IWLPD,Page33-38,1994*

[5]:C. Y. Tsui, M. Pedram, and A. M. Despain. Technology Decomposition and Mapping Targeting Low Power Description. *In Proceeding of the 30th Design Automation Conference, Page68-73,1993*

[6]:A. Chandrakasan, M. Potkonjak, J. Rabaey, and R. Broderson. HYPER-LP:A System for Power Minimization Using Architectual Transformations. *In Proceeding of the International Conference Computer-Aided Design,Page300-303,1992*

[7]:M. Kawarabayashi, N. Shenoy, and A. S. Vincentelli. A Verification Technique for Gated Clock. *In Proceedings of IEEE 30th Design Automation Conference Proceedings 1993*