

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第3949764号

(P3949764)

(45) 発行日 平成19年7月25日(2007.7.25)

(24) 登録日 平成19年4月27日(2007.4.27)

(51) Int.Cl.

G06F 17/50 (2006.01)

F I

G06F 17/50 666G

請求項の数 12 (全 35 頁)

(21) 出願番号	特願平8-318062	(73) 特許権者	506076606
(22) 出願日	平成8年11月28日(1996.11.28)		アバゴ・テクノロジーズ・ジェネラル・ア
(65) 公開番号	特開平9-231261		イピー(シンガポール)プライベート・リ
(43) 公開日	平成9年9月5日(1997.9.5)		ミテッド
審査請求日	平成15年10月29日(2003.10.29)		シンガポール国シンガポール768923
(31) 優先権主張番号	566,233		, イーシュン・アベニュー・7・ナンバー
(32) 優先日	平成7年12月1日(1995.12.1)		1
(33) 優先権主張国	米国(US)	(74) 代理人	110000246
			特許業務法人オカダ・フシミ・ヒラノ
		(74) 代理人	100087642
			弁理士 古谷 聡
		(74) 代理人	100076680
			弁理士 溝部 孝彦
		(74) 代理人	100121061
			弁理士 西山 清春

最終頁に続く

(54) 【発明の名称】 F E Tモデル回路からゲートモデル回路を抽出する方法

(57) 【特許請求の範囲】

【請求項1】

F E Tモデル回路からゲートモデル回路を抽出するための方法であって、コンピュータが、

(a) 前記 F E Tモデル回路の接続状態を表すネットリストのデータを読み込むステップと、

(b) 前記 F E Tモデル回路の前記ネットリストから、電源およびグラウンドを検出して、該コンピュータの記憶装置に保存するステップと、

(c) 前記 F E Tモデル回路の前記ネットリストから、すべての論理信号を検出して、前記記憶装置に保存するステップと、

(d) 1つまたは複数の構造ベースのブールの部分木を生成するステップであって、

i) 前記保存された論理信号のそれぞれを、該部分木の根に設定し、

A) 前記根に設定した論理信号に、ソースまたはドレインを介して接続された F E Tを、前記ネットリストから識別して、該部分木の幹に設定し、

B) ソースおよび／またはドレインの接続を介して、前記幹に設定された F E Tに接続された F E Tを、前記ネットリストから識別して、該幹の枝に設定し、

C) 前記枝が、前記保存された論理信号、前記電源、または前記グラウンドに達したとき、該枝を終端させる、ことにより該部分木を生成し、

該生成された部分木は、A) 論理信号を低に駆動する F E Tを表す 0-木と、B) 論理信号を高に駆動する F E Tを表す 1-木と、を含む、ステップと、

10

20

(e) 前記生成された1つまたは複数の構造ベースのブールの部分木に対し、発見的に枝刈りするステップであって、該枝刈りは、該部分木によって表されるF E Tおよびポートの方向を見極めつつ、所定の規則に従って不要と判断される枝を、該部分木から削除することを含み、該部分木によって表されるすべてのF E Tおよびポートの方向を見極めたことに応じて、該枝刈りを完了する、ステップと、

(f) 前記枝刈りが完了した1つまたは複数の構造ベースのブールの部分木を、ブールの論理方程式に変換するステップと、

(g) 前記論理方程式に従ってゲートのプリミティブをマップすることにより、前記ゲートモデル回路のネットリストのデータを生成するステップと、

を含む方法。

10

【請求項2】

前記ステップ(d)に先だって、前記F E Tモデル回路のネットリストにおけるそれぞれのF E Tについて、前記電源またはグラウンドから開始して該F E Tのソースおよびドレインの接続を横断することによって、最初のF E T方向を推定するステップを含み、

該F E Tの駆動される端が、他の1つのF E Tのみに接続されていると共に、ポート、F E Tゲート、または非F E Tインスタンスに接続されていない限り、該横断は、該F E Tに続いて該他の1つのF E Tについても行われる、

請求項1に記載の方法。

【請求項3】

前記ステップ(d)に先だって、冗長なF E Tを削除するステップをさらに含む、

請求項1または2に記載の方法。

20

【請求項4】

さらに、

a) 前記部分木のそれぞれについて、最大の強さを計算するステップであって、該最大の強さは、該部分木において枝が直列になっているパスのうちの、最も低い抵抗を示すパスの抵抗に対応する、ステップと、

b) 前記ステップ(f)に先立ち、前記部分木において帰還木を検出することにより、該部分木を、弱くない部分木と弱い部分木とに分類するステップであって、該帰還木は、2つの交差結合される部分木であり、該2つの交差結合される部分木のそれぞれは、該2つの交差結合される部分木の他方の根を入力として有する、ステップと、

を含む、請求項1または2に記載の方法。

30

【請求項5】

前記保存された論理信号のそれぞれについて、

a') 弱い部分木であって1-木である部分木を併合するステップと、

b') 弱い部分木であって0-木である部分木を併合するステップと、

c') 弱くない部分木であって1-木である部分木を併合するステップと、

d') 弱くない部分木であって0-木である部分木を併合するステップと、

をさらに含み、該併合によって出力されるa')からd')の4つの構造ベースのブール木を、それぞれの該論理信号について生成し、

前記ステップ(f)は、該4つの構造ベースのブール木のそれぞれを、前記論理方程式に変換するステップを含む、

請求項4に記載の方法。

40

【請求項6】

前記ステップ(f)に先立ち、前記部分木に含まれるサブツリーの並列の組み替えを実行するステップであって、該組み替えは、該部分木における重複したノードの上方にある木構造を組み合わせてることによって達成される、

請求項5に記載の方法。

【請求項7】

共通の前記論理信号を根に持つ前記弱い部分木および弱くない部分木について、論理関数を決定するステップをさらに含む、

50

請求項 6 に記載の方法。

【請求項 8】

さらに、前記決定した論理関数を、対応するゲートのプリミティブにマップすることを含む、

請求項 7 に記載の方法。

【請求項 9】

前記ステップ (e) は、矛盾するゲート信号を有する F E T によって駆動される部分木の枝を削除することを含む、

請求項 1 または 2 に記載の方法。

【請求項 10】

前記ステップ (e) は、さらに、

a) 前記部分木に表されるポートのポート方向を見極めるステップと、

b) 前記部分木に表される F E T の F E T 方向を見極めるステップと、

c) 前記ポート方向または前記 F E T 方向を見極めることができないとき、ユーザに該ポート方向または該 F E T 方向の提供を促すステップと、

を含む請求項 1 に記載の方法。

【請求項 11】

さらに、前記 F E T モデル回路の前記ネットリストから、クロック信号を検出して、該コンピュータの記憶装置に保存するステップを含み、

前記ステップ (f) は、該検出されたクロック信号に少なくとも部分的に基づいて、該ブールの部分木を、該論理方程式に変換する、

請求項 1 に記載の方法。

【請求項 12】

さらに、前記 F E T モデル回路の前記ネットリストから、インバータを検出して、該コンピュータの記憶装置に保存するステップを含み、

前記ステップ (f) は、該検出されたインバータに少なくとも部分的に基づいて、該ブールの部分木を、該論理方程式に変換する、

請求項 1 に記載の方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

この発明は、F E T モデル回路からゲートモデル回路を抽出する方法と装置に関し、特に、F E T モデル回路と構造上の一致を保持するゲートモデル回路を抽出する方法と装置に関する。

【0002】

【従来の技術】

集積回路 (I C) は電界効果トランジスタ (F E T) を用いて設計され、組み立てられるが、多くのツールは I C のゲートレベル・モデル (または少なくとも非常に制限された数の F E T を用いるモデル) を必要とする。I C の F E T モデルは、F E T、ワイヤおよび他の構造素子 (非 F E T インスタンス (non-fet instances)) を含むが、ゲートモデルは、相互接続される論理ゲート (およびできる限り制限された数の F E T) を含む I C の機能的な抽象概念である。F E T モデルからゲートモデルを抽出することは時間がかかり、間違いやすいプロセスである。そこである設計者は、コンピュータを使用してプロセスを自動化しようと試みた。

【0003】

ゲートモデルを抽出する方法の一つの方法は、イグゾースティブ (exhaustive) ・ベクトルの使用による。イグゾースティブ・ベクトル組が回路の入力から導き出され、F E T レベル・シミュレータによって実行される。各々のベクトルは 1 組の期待される出力を作り出し、そこからゲートモデルが合成されうる。そのような方法の不利益は、1) およそ 14 より多い入力をもつ回路は非常に多くのベクトルを必要とするので、方程式が事実上扱いにくい

10

20

30

40

50

こと、および2) 状態素子(格納または帰還)をもつ回路はイグゾースティブ・ベクトルによって十分に特徴付けられることができないことを含む。これは、それらの回路が各々の状態について付加のベクトルを必要とし、付加のベクトルを導き出す現在の方法は扱いにくく、効果的でないからである。

【0004】

ゲートモデルのイグゾースティブ・ベクトル方法の別の不利益は、結果として生じるゲートモデルが、入力FETモデルと構造上の類似をもたないことである。それゆえ結果として生じるゲートモデルは、回路、入力および出力を保持するが、入力と出力の間にFETモデルの構造との類似点をもたない。したがって、ツールがゲートモデル回路で誤りを示すとき、FETモデル回路をデバッグすることは非常に困難になりうる。

10

【0005】

ゲートモデルを抽出する別の方法は、単純な回路の認識に基づく再モデル化変換を含む。この解法は、FET構造を相補形CMOS構造に置き換え、ゲートをもつ回路をプリチャージ(precharge)することを試みる。しかしこの方法は、少数の単純な回路タイプのみの認識に限られる。CPU(中央処理装置)またはASIC(特定用途向け集積回路)のような複雑なICには非常に多くの「規則の例外」が存在し、単純な回路の認識はもはや可能ではない。単純な回路の認識は、FETモデルの構造上の特性を保持するのに役立つが、大規模なICの回路図で使用するには全く実用的ではない。

【0006】

【発明が解決しようとする課題】

20

したがってこの発明の主な目的は、2つのモデル間の構造上の一致を保持しながら、FETモデル回路からゲートモデル回路を抽出する方法と装置を提供することである。

【0007】

この発明の別の目的は、ゲートモデル回路を抽出するために必要とされる時間を大幅に減少させる方法と装置を提供することである。

【0008】

またこの発明の別の目的は、正確で、実証可能な方法および装置を提供することである。

【0009】

さらにこの発明の別の目的は、大半の抽出プロセスから設計者を引き離すために使用される方法と装置を提供することである。

30

【0010】

【課題を解決するための手段】

上述の目的の達成において、発明者は、FETモデル回路からゲートモデル回路を抽出するコンピュータ実現の方法と装置を発明した。

【0011】

方法は、「エキスパートシステム」としてプログラムされるコンピュータによって実行される。エキスパートシステムは、コンピュータの典型的なアルゴリズム方法(解法に至る命令のシーケンスの実行)を、通常人間のエキスパートによって提供される知識ベースの方法と結びつけることを試みるシステムである。

【0012】

40

エキスパートシステムは一般に、知識ベース、推論エンジンおよびユーザー・インタフェースという3つの部分を含む。知識ベースは、典型的に人間のエキスパートによって供給される規則、事実および発見的手法(hueristics)を含む。知識ベースは、推論エンジンによって新しい事実(学習)を推論する(導き出す)ために使用される。推論エンジンが、必要とされる事実を知識ベースから推論することができないとき、システムのユーザーは欠けている情報を供給するように促される。

【0013】

開示される方法と装置(エキスパートシステム)がどのように機能するかということの完全な理解を助けるために、エキスパートシステムの概要およびProlog(開示されるシステムが実現されるコンピュータ言語)が後に続く。当業者であれば、すでにこれらの分野には

50

精通しているはずである。

【0014】

エキスパートシステムの知識ベースは、多数の規則グループ(またはクラス)を含む。Prologのターム(項)で、規則グループは次のステートメントによって表されることができる。

【0015】

【表1】

inference_control (Name, Type, Control)

ここで、Nameは規則グループの名前であり、Typeは推論タイプであり、Controlは規則のリストを含む構造である。規則推論タイプは、順次(基本エンジン)、前向きチェーン(前向きエンジン)、および後ろ向きチェーン(後ろ向きエンジン)を含む。

10

【0016】

順次規則グループが開かれると、その規則は規則グループ内に示される順序で一度に一つずつ発火(fire, 処理開始)する。前向きチェーン規則グループの規則は、グループが開かれて開始されると活動状態になり、規則グループが閉じられるまで任意の時間に(任意の順序で、規則自体の条件の合致が成功することによってのみトリガされて)発火することができる。後ろ向きチェーン規則グループはいつも活動状態である。しかし後ろ向きチェーン規則はデマンド駆動されるので、そうするように要求されるときにだけ発火する。後ろ向きチェーン規則は、「膨らんで」貴重なコンピュータ処理時間を浪費しうる計算を制御するために使用される。

【0017】

すべてのProlog規則は基本形式に従う。形式は以下の通りである。

20

【0018】

【表2】

rule Name if Conditions then Actions [because Notes]

Prolog規則の例は次のように表されることができる。

【0019】

【表3】

```

rule      'Power Connected Fet'
  if      database instance_type(Fet,_,_) and
          not database redundant_fet(Fet) and
          fet_source_drain_signals(Fet,VDD,Sig) and
          database power(VDD,_)
  then    assert signal_driven_to_power(Sig,Fet)
  because ['This signal is connected to power via a fet.'].

```

30

条件は、論理積「and」または論理和「or」によって区分される一つまたは複数の節である。条件節は、エキスパートシステムのデータベースにアクセスすることも、ユーザ定義の述語を呼び出すこともできる。後ろ向きチェーン規則照会は、条件に表される「?」演算子によって行われることができる。一般に規則の条件とは、規則の左側を言う。

40

【0020】

規則条件は、結論または決定(規則の右側を言う)に従える。順次または前向きチェーン規則の右側は、エキスパートシステム・データベースに情報をアサート(assert)し(以下の記述でデータベースとはゲートモデル・データベースのこと)、データベースから情報を取り消し、任意の他のユーザ定義の関数を実行することができる1つまたは複数の動作である。後ろ向きチェーン規則の右側は、後ろ向きチェーン照会の結果を定義する一つのタームである。

【0021】

50

because節は任意であるので、エキスパートシステムの演算に入らない。それは、規則の関数に関する文書化(documantation)に使用される。

【0022】

例えば上述の規則「Power Connected Fet」は次のように機能する。データベースで冗長としてマークされず、そのソース/ドレイン(Prolog述語fet_source_drain_signals/3)が指示される電源信号に接続されるデータベースの各々のインスタンス(instance_type/3)について、エントリが「signal_driven_to_Power(Sig,Fet)」のPrologのデータベースにアサートされる(加えられる)。ここで、変数FetはF E Tのインスタンス名であり、Sigは電源信号と反対のF E Tの信号名である。

【0023】

ここで開示されるエキスパートシステムを含む方法および装置は、電源、グラウンド(接地)およびクロック信号を認識するステップと、インバータを認識するステップと、F E Tモデル回路に含まれるすべての論理信号を認識し、保存するステップと、認識される論理信号の各々について、一つまたは複数の構造ベースのブールの部分的トリーを組み立てるステップと、一つまたは複数のブールの部分的トリーを発見的に枝刈り(pruning)するステップと、一つまたは複数のブールの部分的トリーから論理方程式を組み立てるステップを含む。

【0024】

この方法は、抽出されるゲートモデルの構造上の整合性を保持する際の論理信号の重要性和、構造ベースのブール・トリーの使用を認識する。構造ベースのブール・トリーは、以下の記述で開示される新しい概念である。部分的トリーは、論理方程式を簡略化し、ゲートモデルをより高いレベルに抽象化することを助ける。以前のゲートモデル抽出ツールは、大規模でこれらの仕事を達成することができなかった。

【0025】

構造ベースのブールの部分的トリーの生成によると(ここで部分的トリーの枝はF E Tを表す)、上述の方法によって作り出されるゲートモデルは、そのF E Tモデルと構造上の一致をもち、テストツールがゲートモデルで誤りを示すときにデバッグするのがより簡単なF E Tモデルを作る。

【0026】

前もって人間に任せられる多くの推論プロセスを通ることによって、発明はゲートモデル回路を抽出するに必要とされる時間を著しく減少させる。さらに、エキスパートシステムの方法および装置は保守的(conservative)であって、それが適度に信頼できるときにF E T方向、ポート方向または回路関数を推論するにすぎない。

【0027】

F E T方向、ポート方向、または回路関数がいくぶん曖昧であるとき、エキスパートシステムは、最も正しく効率的なモデルを作り出すように、選択肢の間で選ぶようにユーザーを促す。しかし開示される方法は、ユーザープロンプトを少しだけ用いて、多くのゲートモデルを抽出することができる。

【0028】

【発明の実施の形態】

F E Tモデル回路からゲートモデル回路を抽出するコンピュータ実現される方法が以下に記述され、概して図1に示される。方法は、電源、グラウンドおよびクロック信号を認識するステップ(44)と、インバータを認識するステップ(46)と、F E Tモデル回路のすべての論理信号を認識し、保存するステップ(48)と、認識される論理信号の各々について、一つまたは複数の構造ベースのブールの部分的トリーを組み立てるステップ(50)と、一つまたは複数のブールの部分的トリーを発見的に枝刈りするステップ(52)と、一つまたは複数のブールの部分的トリーの各々から論理方程式を組み立てるステップ(54)とを含む。

【0029】

方法は、プログラマブル・コンピュータ40(またはコンピュータ40をプログラムすることができる物理記憶媒体42に格納されるエキスパートシステムによって実現されることが好

10

20

30

40

50

ましく、エキスパートシステムは一般に F E T モデル回路入力、推論エンジン、規則ベース、ユーザー入力およびゲートモデル回路出力を含む。図 3 4 および図 3 5 を参照されたい。推論エンジンは規則を処理し、ゲートレベル・モデルを(出力ネットリストの形式で)推論する際に、F E T モデル回路入力を(入力ネットリストの形式で)使用する。

【0030】

このように方法と装置を大体説明したが、それらをさらに詳細に説明する。ここで開示される方法と装置は、Prolog コンピュータ・プログラミング言語を使用して開発されている。しかし特定のプログラミング言語の使用は、開示される方法と装置を使用するために必ずしも必要ではなく、当業者であれば異なったプログラミング言語を使用して開示される方法と装置を実現することを選択することができる。

10

【0031】

好ましい実施例で、方法は図 2 に示される拡張されたステップのシーケンスを含む。方法は、F E T モデル回路に含まれる情報が入力ネットリストの形式でエキスパートシステムに提供されることを必要とする。ネットリストは、回路の各々のネット(ノード)に接続される端子のリストを提供する。

【0032】

システムの外側層(またはシェル)は、引数を確認し、ファイルの存在を確認し、入力ネットリストをデータベースに読み込み、それを Prolog タームに変換し、任意に出力ネットリストを交互にフォーマットされるネットリストに変換する役割を果たす。外側層が入力ネットリストを Prolog タームに変換した後に、推論エンジンが始まる。方法のどのステップがいつ実行されるかを実際に決定するのは推論エンジンである。規則が推論エンジンによって発火されると、ゲートモデルの作成を助けるという事実が、エキスパートシステムのデータベースにアサートされる。

20

【0033】

推論エンジンは、最初に一連の順次規則グループを開く。これらの第 1 のものは、電源、グラウンドおよびクロック信号を認識する役割を果たす規則グループである。電源、グラウンド、およびクロック信号に使用される信号の命名規則は次の通りである。

【0034】

【表 4】

NEG: *GND, GND, GND*, gnd*, gnd, *gnd, VSS, vss
 POS: *VDD, VDD, VDD*, *vdd, vdd, vdd*, *VCC, VCC,
 VCC*, *vcc, vcc, vcc*, *VGG, VGG, VGG*, *vgg,
 vgg, vgg*
 CLK: CK, CK*, ck, ck*, NCK, NCK*, nck, nck*, *CLK,
 CLK, CLK*, *clk, clk, clk*, *CKH, (CKH), *ckh,
 (ckh), *CKL, (CKL), *ckl, (ckl)
 (*は一つまたは複数の文字を表す)

30

40

この方法はこれらの規則を認識して、エキスパートシステムのデータベースにこれらの信号について事実をアサートする(44)。しかし規則は、入力ネットリストに(またはネットリストに関連する個別のファイルに)含まれる特性によって無視されることがありえる。

【0035】

次に推論エンジンは、冗長な F E T の除去規則グループに引き続きチェーンする。強制的ステップではないが、冗長な F E T を除去するプロセスは後の計算と抽象化ステップをかなり簡略化する。

【0036】

冗長な並列 F E T の除去は非常に簡単である。2 つまたはそれ以上の F E T が共通のソー

50

ス、ドレインおよびゲート接続を共有する場合、それらの1つを除いて全てが除去される。図4を参照されたい。しかし方法は、さらに一つのステップに進み、冗長な「FET構造」を除去する。冗長な「FET構造」の除去は、それほど簡単ではない。FET構造を等価性(または冗長性)について検査する疑似符号で実現する発見的手法は、次のように与えられる。

【0037】

【表5】

For each signal, S, in circuit

For each pair of fets, F1, F2, connected to S

10

IF F1 & F2 of same type

Put F1 (F2) in tree Tree1 (Tree2)

IF redundant_trees_from_fets(F1,F2,Tree1,Tree2)

mark tree Tree2 as redundant with Tree1

redundant_trees_from_fets(F1,F2,Tree1,Tree2):

IF gate signals of F1 & F2 not same return FALSE

20

Locate signals on other side of fets:S1,S2

IF S1 = S2 return TRUE

IF either S1 or S2 is a port return FALSE

IF either S1 or S2 power or ground return FALSE

IF seen S1 (S2) in Tree1 (Tree2) before return FALSE

For each instance, I1, connected to S1

IF I1 in Tree1 or Tree2 return FALSE

30

IF S1 is connected to the gate of I1 return FALSE

IF I1 is not a fet return FALSE

IF Instance I2, connected to S2, & I2 same type as I1

& I2 not in Tree1 nor Tree2

Add I1 (I2) to Tree1 (Tree2)

IF NOT redundant_trees_from_fets(I1,

I2,Tree1,Tree2)

40

return FALSE

ELSE return FALSE

return TRUE

簡略化のために、上述の冗長除去プロセスは、1)一つのFET構造のトリーのすべてのFETゲート信号が、冗長なFET構造の対応するFETゲート信号と同一であること、および2)FET構造が帰還ループを含まないこと、を要求する。

【0038】

除去プロセスは、それらのソースおよび/またはドレインを通して共通のノードに接続さ

50

れる2つのFETで開始する(図5のノードSを参照)。接続されているFETは同じタイプ(両方ともp-FETか両方ともn-FET)でなければならない。さらに、接続されるFETのゲート信号が同一でなければならない。FET構造トリーは、共通のノードに接続されるFETの各々について生成される。トリーは、FETソース/ドレイン接続を通して、2つのトリーが共通のノード(図5のノードT)で再集結するまでパスを横断することによって生成される。

【0039】

回路の注記または特性が入力ネットリストに加えられて、インスタンスを冗長であると言明したり(その除去を強制する)、または冗長なFETの除去プロセスがFETを除去しないようにすることもできる。

10

【0040】

続いて推論エンジンは、方法のこの時点で2つの付加の規則グループにチェーンすることができる。第1に、キャパシタとして使用されるFETが発見的に認識され、これらのFETに関する事実がエキスパートシステムのデータベースにアサートされることができる。FETは、1)そのソースとドレインが両方とも電源に(または両方ともグラウンドに)接続される(図64を参照)、あるいは2)それが永久に動作不能である(つまりゲートがグラウンドに配線されたn-FET、またはゲートが電源に配線されたp-FET)(図65を参照)場合、キャパシタであるとアサートされる。第2に、レールに配線されるソースまたはドレイン(電源またはグラウンド)のどちらかを有するすべてのFETは、データベースにそのように注記される。これらのステップは両方とも、後の発見的な部分的トリーの枝刈りのステップを簡略化する。

20

【0041】

方法の次のステップはインバータの認識(46)である。再び推論エンジンは、このステップの規則グループに引き続きチェーンする。最も基本の論理ゲートとして、インバータを認識して、抽象化が行われ、ゲートモデルの論理が最小限にされることができることが重要である。インバータは、1)FETの各々が共通の論理信号へのソース/ドレイン接続をもち、2)一方のFETのp-FETが、電源に接続されるソース/ドレインをもち、他方のFETのn-FETが、グラウンドに接続されるソース/ドレインをもち、3)FETのゲート信号が共通の論理信号に接続される、という一対のFETのパターンの合致によって認識される。また反転の関係は、信号名N<sig>をもつネットリストの入力ポートは入力ポート<sig>の反転であるという取り決めによって認識される。

30

【0042】

インバータを認識した後に(46)、できる限り多くのFETについて最初のFET方向が決定される(58)。最初のFET方向は、電源またはグラウンド信号から、該電源またはグラウンド接続に直列に接続されそこから離れる方へ進み、一つまたは複数のFETのソース/ドレイン接続を横断することによって決定される。一端(ソース/ドレイン)が電源またはグラウンドに接続されているFETは、その他端(ソース/ドレイン)で信号を駆動するものとして、データベースにアサートされる。FETのソース/ドレイン接続は、FETの駆動される端が他の一つのFETのみに接続され、ポート、FETゲート、または非FETインスタンスが遭遇されない限り、横断される(横断される各々のFETが電源またはグラウンドから遠ざかる方へ駆動することをデータベースにアサートする)。最初のFET方向を決定するために使用される規則グループは、前向きチェーン規則グループでなければならない。

40

【0043】

上述のステップは、順次または前向きのチェーン規則グループを開くことを含んでいる。次のいくつかのステップは、後ろ向きチェーン規則グループを開くを含む。

【0044】

この方法の重要なステップは、回路の中のすべての論理信号の認識および保存(48)である。論理信号は、論理が生成される信号のサブセットを表す。他のすべての回路信号は中間物と考えられ、ゲートモデル抽出プロセスの間に包含されうる。論理信号は、ポート(電

50

源およびグラウンドのポートを除く)、F E Tゲート、あるいは非F E Tインスタンスに接続される任意の信号として定義される。非F E Tインスタンスは、設計者がF E T実現の代わりの抽象化を与えるインスタンスをF E Tモデル回路に含む。例えば加算機または比較器への接続は、非F E Tインスタンスへの接続である。図6で、認識される論理信号は、A、B、C、D、WおよびZである。V、XおよびYは中間信号と考えられ、抽出プロセスの中に包含されることができる。

【0045】

論理信号を認識して、結果として生じるゲートモデルがそのF E Tモデルと構造上の一致をもつことが重要である。他のゲートモデル抽出方法では論理信号を認識せず、その結果全てではないとしてもほとんどの論理信号がゲートモデルから外される。他のゲート抽出モデルは、パス/失敗の原理で回路のテストを可能にしたが、F E T回路は一度誤りが注意されると、テストされるゲートモデルへの構造上の類似がないために、デバッグするのが難しかった。

10

【0046】

方法の次のステップは、認識される論理信号の各々について、一つまたは複数の構造ベースのブールの部分的トリー（部分木、以下同様）を組み立てることを含む(50)。部分的トリーの構造の基礎(basis)は、入力ネットリストのソース/ドレインF E T接続である。部分的ブール・トリー（ブール部分木、以下同様）は、論理信号に接続される各々のF E Tについて構築される。トリーのルート（根、以下同様）は論理信号である。接続しているF E Tは幹と考えることができる。幹の他方の側に接続される全ての他のF E Tは、別の論理信号または電源/グラウンド信号が到達されるまで(リーフ(leaf))、(F E Tの端と端を接続して)扇形に広がる枝を開始する。したがって特定の論理信号は、それに関連する(各々のF E Tのトリーはそれに接続される)いくつかの部分的トリー(森)をもつことができる。

20

【0047】

部分的トリーは構造ベースであるので、枝は単なる論理パスではなく、F E TモデルにおけるF E Tを表す。構造ベースのブールの部分的トリーは、BDD(boolean decision diagrams、ブールの決定図)に類似しているが、構造の基礎をトリーに加える際にさらに一つのステップへ進む。以前のトリーはこれを行わなかった。BDDは論理ベースであり、いかなる構造とも一致しない。

30

【0048】

部分的トリーは他の論理信号で終わるので、それらは比較的小さく、従って管理し、推理するのが容易である。一方、BDDおよびその他の同様の方法は成長して非常に大きく扱いにくくなり、時間および空間の両方で管理するのに費用がかかる。

【0049】

いくつかの部分的トリーは1-トリー（1-木、以下同様）であり、(電源に対し)高い論理信号を駆動することができるF E Tを表す。トリーのいくつかは0-トリー（0-木、以下同様）であり、(グラウンドに対し)低い論理信号を駆動することができるF E Tを表す。したがって、N O Rゲートは2つの1-部分的トリーをもち、N A N Dゲートは2つの0-部分的トリーをもち。

40

【0050】

図7の論理信号Zは、図8で説明されるような3つの部分的トリーに関連する。

【0051】

部分的トリーは、論理信号で始まり、その論理信号に接続されるF E Tを(部分的トリーの幹になる)そのソースまたはドレインによって識別し、ソースおよび/またはドレイン接続を介して端から端まで幹に接続されるF E Tを部分的トリーに加わえることによって組み立てられる。部分的トリーの枝は、電源、グラウンドまたは論理信号のいずれかに直面したときに終わる。

【0052】

この段階で、部分的トリーはF E Tまたはポートの方向を考慮せずに組み立てられるので

50

、トリーの枝は様々な部分的トリーに複製されることが可能である。例えば、図15のFET F1を含み、論理信号Yで終わる論理信号Zの部分的トリーは、Zで終わるYの部分的トリーのようにも見える。

【0053】

部分的トリーは組み立てられるとき、単純な規則によって枝刈りされることができる。矛盾するゲート信号がトリーの枝に見つけられる(論理反転の以前の識別によって)場合、その枝は枝刈りされる。例えばゲート信号AをもつFETがゲート信号 $\sim A$ (Aの否定またはAの反転)をもつFETと直列である場合、その枝は論理的に不可能を表すので枝刈りされる。

【0054】

各々の部分的トリーについて、最大の強さを計算することができる。計算はレール(電源およびグラウンド)で強さ63から始まり、そして直列の枝パスの各々のFETについて計算されるFET「抵抗」を減ずる。強さは1以下に下がることができない。抵抗は $R=\max(K \cdot L/W, L)$ として計算され、ここでWおよびLはそれぞれのFETの幅および長さである。Kは、p-FETおよびn-FETについてユニークな値をもつプロセスに特有の係数である。より低い抵抗をもつゲートは、より大きい強さをパスに加える。部分的トリーの最大の強さは、部分的トリーの最強の枝パスの強さに等しい。強さの計算は、帰還パスの識別のような後の規則に関連して使用されうる。

【0055】

構造ベースの部分的トリーが組み立てられたあと(50)、多数の付加の後ろ向きチェーン規則グループを使用して発見的に枝刈りされる(52)(図3)。規則グループは、第1および第2フェーズのグループに小分割される。枝刈りは第1フェーズの規則から始まって、必要に応じて第2フェーズの規則へ進む。最後に(また必要に応じて)、ユーザーは情報について促される。第1および第2フェーズの規則グループの各々はさらに、部分的トリー枝刈り規則68、74、FET方向規則70、76、およびポート方向規則72、78に小分割される。FETおよびポート方向規則は、後ろ向きチェーンによって部分的トリー枝刈り規則からアクセスされる。規則フェーズの背後の理論は、方法がユーザを促すことなく、できる限り多く成し遂げなければならないということである。エキスパートシステムが十分な情報を入力ネットリストまたはそれ自身の推論規則から得ることができない場合は、情報をユーザから得ることができる。

【0056】

第1フェーズの部分的トリー枝刈り規則(68)が発火されるとき、ゲートモデルに関する事実がデータベースにアサートされる。事実は、FET方向、ポート方向および論理的フロー・パスに関する情報を含む(つまり実現不可能なパスは枝刈りされる)。

【0057】

入力、出力あるいは双方向のいずれのポート方向も、必ずしも入力ネットリストで指示されるとは限らない。その結果、方法はしばしばこの情報を回路から推論しなければならない。同様に通常全てのFETの方向は与えられないので、決定されなければならない。これは、トリー枝刈り規則が多くの情報を必要とするので、方法のこの時点でなされる。

【0058】

第1フェーズの部分的トリー枝刈り規則は、必要とされる場合はいつでもFETまたはポートの方向について照会する。照会が行われるとき、推論エンジンはその照会が以前に行われたかどうか決定し、もし行われたのであればFETまたはポート方向がデータベースに存在するかどうか決定する。もし存在しないならば、推論エンジンは第1フェーズのFET 70またはポート72の方向規則グループに後ろ向きにチェーンする。FETまたはポート方向が決定できないならば、照会は満足されずに戻される。付加の情報がデータベースにアサートされるとき、再び質問が行われ、満足される。さもなければFETまたはポート方向の決定は、第2フェーズの規則が開かれて、第2フェーズの規則またはユーザーの入力のどちらかが照会を満足するまで待たなければならない。

【0059】

10

20

30

40

50

第1フェーズの規則が可能な限り多く達成すると、第2フェーズの部分的トリー枝刈り規則74が推論エンジンによって開かれる。第2フェーズ規則74は、第1または第2フェーズのFET70、76またはポート72、78の方向規則をどちらも照会することができる。第2フェーズでFETまたはポート方向が規則グループを使用して推論されることができないとき、ユーザーは方向について促される。方向が提供されると、規則は、新しい情報を使用して発火しうる全ての規則が発火するまで、重ねて発火する。任意に、ユーザーが方法の実行に随伴することができないとき、方法は非相互作用モードで実行させるように命令されうる。非相互作用のモードで実行するとき、方法は未知のポートまたはFET方向が双方向であるとアサートしなければならない。

【0060】

部分的トリーの構造の基礎によって、推論エンジンは部分的トリーの枝刈りがいつ完了するかを決定することができる。部分的トリーの枝刈りは、すべてのFETの方向(部分的トリーの枝)が知られ、それ以上のトリー枝刈り規則68、74が発火し得ないとき完了する。

【0061】

論理的トリー構造(あるモデル化プロセスによって使用される従来のBDDのような)を用いる場合、双方向のFETおよびポート方向を分解できないことは、結果として生じるゲートモデルに付加の複雑さを加える。論理的トリーを使用するとき、FETまたはポート方向の決定を強制することは不可能である。これはFET構造が表されず、多数の論理信号がゲートモデル化プロセスに包含されるからである。

【0062】

構造ベースのブール・トリーの使用は、ゲートモデルに構造上の一致を加えるだけでなく、ゲートモデルの論理を簡略化する助けになる。

【0063】

発見的な部分的トリー枝刈り規則は次の通りである。

【0064】

1. 第1フェーズの部分的トリー枝刈り規則

部分的トリーは、そのルートの論理信号が入力ポートであり、部分的トリーの全ての枝が共通の論理信号で終わる場合、その全体を枝刈りされる。第1フェーズのポート方向規則は、部分的トリーのルートが入力ポートであるかどうかを決定する後ろ向きチェーン照会によって発火されることができる。この規則の背後にある論法は、入力ポートと一つの論理信号の間の流れが、入力ポートから遠ざかるものでなければならないということである。部分的トリーによって示される流れの方向は入力ポートに向いているので、入力ポートでルートした(rooted、入力ポートを根に持つ)トリーは間違った論理パスを表しており、枝刈りされなければならない。図9は、この規則の適用を示している。入力ポートAは、枝刈りされる論理信号Sに基づく部分的トリーを持つ。

【0065】

ある論理信号の部分的トリーが論理信号の「安定した(sound)」トリーのグループよりも弱い場合、その全体が枝刈りされる。安定したトリーとは、完全な相補形CMOSゲート(例えばNANDゲート)のような常に駆動しているゲートを表す。この規則の理由は、安定したゲートより弱い部分的トリーは、そのルートの論理信号の値に影響を与えることができないことである。例えば図10で、YからZまで組み立てられた部分的トリーは、ZがYからZへのパスよりも安定したNANDゲートによって駆動されると決定される場合に枝刈りされる。

【0066】

部分的トリーの枝は、それがFETでありえるとき枝刈りされる。部分的トリーで示されるFETの方向は、それが部分的トリーのルートに向かって駆動するということである。従って、FETが事実上部分的トリーのルートから遠ざかる方へ駆動すると決定される場合、枝は論理パスではなく、枝刈りされなければならない。第1フェーズのFET方向規則は後ろ向きチェーン照会によって呼び出される。図11で、FETのF2を含むZの部

10

20

30

40

50

分的トリートは、F 2 が B に向かって(質問における部分的トリートのルートである Z ではなく)事実上駆動すると決定されると、枝刈りされる。

【0067】

部分的トリートは、同じ論理信号にルートした別の部分的トリートがそれを包含する場合にその全体を枝刈りされる。この枝刈りステップは、構造上の部分的トリートの論理的な枝刈りを含む。例えば、ある部分的トリートが「A and B」という論理関数を表し、別の部分的トリートが「A and B and C」という論理関数を表す場合、後者の部分的トリートは論理的に前者によって包含されるので枝刈りされる。図 12 を参照されたい。

【0068】

最後に、部分的トリートの枝は、その FET のゲート信号の 2 つが互いに相反する場合に枝刈りされる。例えば、ある枝の FET がゲート信号 A をもち、同じ枝のもう一つの FET がゲート信号 NA (A の否定)をもつ場合、そのようなパスは実現することが不可能であるので、枝は枝刈りされなければならない。Z の部分的トリートを含む図 13 の FET の場合、FET F1 で始まる枝は相反するゲート信号 A および NA によって枝刈りされる。

【0069】

2. 第 1 フェーズの FET 方向規則

以下の規則は後ろ向きチェーンであり、特定の FET の方向が質問の中にあるときにのみ発火される。

【0070】

FET が特別なタイプの方向性のない FET である場合、それは方向性のない FET のライブラリの FET と比較されうる。その方向がライブラリから引き出され、データベースにアサートされる。

【0071】

FET の方向が FET モデル・ネットリスト(入力ネットリスト)に含まれる場合、FET モデル・ネットリストで提供される方向がデータベースにアサートされる。

【0073】

電源またはグラウンドに接続される FET は、電源またはグラウンドの接続から遠ざかる方へ駆動する。図 14 を参照されたい。

【0074】

上述したように、示される FET 方向は、それが部分的トリートのルートに向かって駆動するということである。示される FET 方向が、それが属する全ての部分的トリートで一貫している場合、その示される方向はその実際の方向であるとアサートされる。図 15 で、FET F1 は論理信号 Y および Z の部分的トリートの一部である。F1 が Y の部分的トリートから枝刈りされる場合、この規則の適用は、F1 が Z に向かって駆動することを示す(他のいかなる部分的トリートも相反する方向を示唆しない)。

【0075】

FET がポートへの唯一の接続を含み、そのソースまたはドレインを介してポートに接続されている場合、それが入力ポートであるならばポートから遠ざかる方へ、それが出力ポートであるならばポートに向かって FET は駆動する。ポートが双方向であるとき、FET 方向に関する結論は(この規則によって)導かれない。ポートから遠ざかる方へまたはポートに向かって駆動するとアサートされる FET の例は、図 16 および図 17 に示されている。

【0076】

ある部分的トリートが、論理信号を一方向に(つまり電源へまたはグラウンドへ)引くことができる唯一の部分的トリートである場合、その部分的トリートの FET はその論理信号を駆動するはずであり、FET の示される方向はそれらの実際の方向であるとアサートされる。この発見の手法は、ポートに接続されないすべての論理信号は、それらを駆動する FET によって、両方のレール(電源とグラウンド)へ駆動されることが可能でなければならないと仮定する。ポートに接続される論理信号は、回路の外の論理がそれらに影響を与えることができるので、この規則から除外される。この規則は、FET が双方向でありうるとい

10

20

30

40

50

う事実を無視する。しかし双方向 F E T はまれであるために、それはめったに失敗しない。図 18 で、F E T F 1 は Z を「1」に駆動することができる唯一の F E T であり、従って F 1 は Z に向かって駆動するとアサートされる。

【0077】

F E T が、それに対向して駆動することができる他のいかなる可能なトリーよりも強い「安定した」論理ゲート(インバータすなわち完全な相補形 N A N D ゲートのような)によって駆動される場合、F E T はより大きい論理ゲートから遠ざかる方へ駆動するとアサートされる。例えばマスタースレーブ・レジスタで、マスターインバータは、スレーブの帰還インバータよりもずっと大きいので、マスターとスレーブの間の転送 F E T の方向はスレーブに向かって駆動すると決定される。図 19 の転送 F E T を参照されたい。

10

【0078】

3. 第 1 フェーズのポート方向規則

ポート方向特性が入力ネットリストに存在する場合、ポート方向は指示される通りである。

【0079】

電源およびグラウンドは常に入力の方向を持つ。

【0080】

ポートがパススルー・ポートである場合、その方向は入力である。パススルー・ポートは回路への接続を持たず、整合性のために F E T モデルからゲートモデルへ単にコピーされる。

20

【0081】

駆動されないポート(単に F E T ゲートに接続されるポート)の方向は、図 20 の A のように入力である。

【0082】

ポートを駆動する論理が安定している場合、ポートは帰還パスと関連せず、ポートの方向は図 21 のポート Z のように出力である。

【0083】

4. 第 2 フェーズの部分的トリー枝刈り規則

F E T の実際の方向がその部分的トリーの枝によって示される方向と反対であるならば、その F E T を含む枝は枝刈りされる。この規則は、第 1 フェーズの枝刈り規則と同一である。F E T およびポートの方向は、第 2 フェーズの F E T (76) およびポート (78) 方向規則への後ろ向きチェーン照会から得られる。F E T またはポート方向に関する新しい情報を得た後、後ろ向きチェーン照会が任意の第 1 フェーズの規則グループに対して行われる。

30

【0084】

5. 第 2 フェーズの F E T 方向規則

第 2 フェーズの F E T 方向規則は、F E T 方向についてユーザーを単に促すことである。ユーザーは、F E T が特定の信号を駆動する、または F E T が本当に双方向であると応答することができる。

【0085】

促すことについてユーザーが利用可能でない場合、F E T は双方向であると仮定される。

40

【0086】

6. 第 2 フェーズのポート方向規則

ポートの部分的トリーのすべてが排除される場合、駆動されないポートは入力の方向を持つ(図 22)。

【0087】

ポートがある部分的トリーのルートに現れるが、他の全ての部分的トリーから「リーフ」として排除される場合、ポートの方向は出力または双方向のどちらかでなければならない。ポート(図 23 のポート Y)が回路によって両方のレールに駆動される場合、その方向は出力であるとアサートされる。そうでない場合、それは図 24 のポート Z のように双方向であるとアサートされる(この双方向の発見的手法は、テストパターン・ジェネレータが

50

ポートをより試験可能にする)。

【0088】

ポートの方向が依然として未知である場合、ユーザーは方向について促される。

【0089】

エキスパートシステム装置の代替実施例で、システムは、所与の「努力(effort)」を F E T またはポート方向を得ることに注ぐように命令されうる。例えば非常に大きい F E T モデルからゲートモデルを抽出するとき、ユーザーが F E T またはポート方向について促される前に、システムはある程度の仕事しか出さないように命令されうる。こうして、コンピュータで実行される計算の数を低減させることができ、方法の実行時間を減らすことができる。

10

【0090】

推論エンジンによってアクセスされる規則グループの残りは(論理関数の決定がなければ)順次の規則グループである。

【0091】

この方法の重要な特徴は、弱い部分的トリートを弱くない(強い/支配的な)部分的トリートから分割する(60)能力である。分割は、ゲートモデルの論理関数を抽出する際の助けとなる。例えば、ラッチを駆動する強い部分的トリートから、ラッチの弱い帰還インバータを表す弱い部分的トリートを切り離すことによって、ラッチがそのままの状態では抽出されることができ、従ってゲートモデルでラッチをより本来的にモデル化する。他の方法は、多数の F E T を使用して実現される状態インスタンスを捕獲するが、オリジナルの F E T 構造または回路の記憶信号との一致を保持しない。

20

【0092】

上述したように、部分的トリートの最大の強さは、部分的トリートを通る最も強いパスの強さに等しい。弱いおよび弱くないトリートは、多様な部分的トリートの最大の強さの比較によって切り離される。

【0093】

弱い部分的トリートは、論理信号の弱くない1および0-トリートから切り離される。部分的トリートが弱いとマークされるとき、「全体の」部分的トリートは弱い。部分的トリートは、弱い部分と弱くない部分にさらに小分割されるべきではない。

【0094】

帰還が存在しないならば、弱い部分的トリートをつくる必要はないと仮定する。この仮定は、弱くないトリートから弱いトリートを分割する仕事を大いに簡略化する。仮定は必ずしも正しくはないが(例えば、帰還は、モデル化された回路の外に存在することがありえる)、その利益がその損失よりはるかに重要である場合には正しい。

30

【0095】

切り離しプロセスを開始するために、帰還トリートが認識されなければならない。部分的トリートは、交差結合(cross-coupled)される2本のトリートについて吟味される。すなわち各々の部分的トリートの論理信号は、他のトリートへの入力である。これは2路帰還の例である。図25で、F E T F 1、F 4およびF 5は信号Aのトリートにあり、F E T F 2、F 3およびF 6は信号Bのトリートにある。したがって、AとBの間には帰還が存在する。また3路および4路帰還は、この方法(2本のトリートの間の帰還が実際に他のトリートを通過する)を使用して認識される。図26で、論理信号AとBの間の帰還は、信号Cの部分的トリートを通過する。

40

【0096】

より強い「反対側」に別の部分的トリートが存在し、2本の部分的トリートが「解体されて(disjoint)」いない場合、帰還と関係する部分的トリートは弱い部分的トリートである。「反対側」という用語は、反対のルールへ駆動する部分的トリート(1-トリート対0-トリート)を言う。

「解体」は、部分的トリートが(トリートの論理によって決定される)同じ時間に駆動することが不可能であることを意味する。解体の制約は、フィードフォワード・インバータのn-F E Tが、同じインバータのp-F E Tの足のように強くないからといって弱いと類別さ

50

れることを防ぐ。図 2 5 で、部分的トリ－ F 4 および F 5 は、信号 A をいずれのレールにも駆動することができる F 1 よりも弱いので、弱いとマークされる。

【0097】

また、反対側に部分的トリ－が存在せず、論理信号(すなわち帰還トリ－のルート)がポートであるかまたは非 F E T インスタンスによって駆動されるかのどちらかである場合、帰還と関係する部分的なトリ－は弱い部分的トリ－である。図 2 7 で F E T F 1 は、ポート Z および信号 Y の間の帰還に含まれているので、この規則によって弱いとマークされる。

【0098】

部分的トリ－が、(電源またはグラウンドからでなく別の論理信号から来る)転送トリ－であり、転送信号が反対のレールへ駆動されることが出来る帰還信号である場合、弱い。図 2 6 で F E T F 7 は、この規則によって弱い部分的トリ－としてマークされる。

【0099】

最終的に部分的トリ－は、弱い部分的トリ－が他方の側に存在し、部分的トリ－の入力が他の部分的トリ－の入力のサブセットであれば、弱いトリ－である。この規則は、強い F E T を一方の側にのみ駆動する弱い論理の他方の側を拾い上げる。一方の側にだけ強い F E T があると、他の弱い側は、この規則なしでは認識されない。図 6 6 の例を参照されたい。F E T F 2 は、F E T F 3 がすでに弱いとマークされているので、この規則によって弱いと認識される。すなわち F 6 よりも弱いということである。

【0100】

部分的トリ－が枝刈りされ、弱いおよび弱くないグループに切り離されたので、それらは、完全な構造ベースのブール・トリ－を形成するように併合される。この点に先立ち、方法は部分的トリ－によって厳格に機能している。識別される各々の論理信号は、最高 4 本のトリ－を持つことができる。論理信号について可能なトリ－は、強い 1-トリ－、強い 0-トリ－、弱い 1-トリ－および弱い 0-トリ－を含む。1-トリ－は論理信号を電源へ引く役割を果たし、0-トリ－は信号をグラウンドまで引く役割を果たす。

【0101】

完成されるブール・トリ－は、各々の論理信号についてブール論理方程式を組み立てるために使用される(54)。単純な変換を使用して、各々のブール・トリ－を論理方程式に変える。直列の F E T はブールの A N D 節「*節」を形成し、枝(またはサブ・トリ－)を分岐させることは、ブールの O R 節(「+」節)を形成する。例えば、C M O S N O R ゲートは、2 つのサブ・トリ－(ブールの O R 節)をもつ 0-トリ－を含み、それぞれのサブ・トリ－はゲート出力をグラウンドに引くことができる。N O R ゲートの 1-トリ－は、ゲートの出力を電源に引くことができる 2 つの直列の F E T (ブールの A N D 節)を含む。論理方程式は、ド・モルガン(反転)形式で 1-トリ－について組み立てられる。したがって、N O R ゲートの方程式(入力 A および B を仮定する)は以下で表される。

【0102】

【数 1】

0-方程式 = $A+B$

1-方程式 = $\sim(\sim A \sim B)$ (ド・モルガン形式)

$A+B$ (最終の簡略化された形式)

反転は、トリ－の中の F E T のタイプによって命令される。p - F E T は反転を引き起こすが、n - F E T は反転を引き起こさない。ド・モルガン形式を使用する理由は、安定した論理の検出(0-および 1-論理方程式は両方とも同一になる)を簡略化するためである。

【0103】

上で述べたように、F E T 構造と厳密に一致するゲートモデルを抽出することがこの方法の目的である。これは時々最適より少し劣る論理方程式を与えるが、それは回路の問題をデバッグする際にユーザーを大いに助ける。例えば、方程式 $(A+B) \cdot (C+D)$ をもつ C M O S 相補形 A N D - O R ゲートを考える。ゲートの下半分が図 2 8 に示されている。ブール論理を使用すると、この方程式は、 $A \cdot C + B \cdot C + A \cdot D + B \cdot D$ という同等の方程式に操作され

10

20

30

40

50

ることができる。しかし、F E T 構造が第 1 の方程式を直接に与える場合、その方程式を保持することが好ましい (F E T A および B が並列で、F E T C および D が並列であり、一つの並列構造は他方と直列である)。操作される方程式は、F E T 構造を「逆に設計する」こと、および F E T 構造を含む回路をデバッグすることを困難にする。

【0104】

F E T モデルの物理的構造をたどるために、トリーから方程式への変換にはわずかな修正が必要である。これは、トリーが格納されている態様のためである。図 28 に示される A N D - O R ゲートの格納されているトリーが、図 29 に表されている。方程式の変換に対する率直なトリーは、方程式 $A*(C+D)+B*(C+D)$ をもたらす。この方程式は、部分的トリーの並列の組換えを行っていないので、方程式 $(A+B)*(C+D)$ とは形式が異なる。並列の組換えを実行するために、サブ・トリーを開始するノード 80 の信号名がデータベースに格納される。図 29 で、ノード 80 の名前は X 1 である。ノード名は、(C+D) サブ・トリー構造とともに格納される。したがって、並列のサブ・トリーを組み換えるために、第 1 のステップはトリーのノードのリストを作り、複製 (重複) を保有することである。複製は、それらのノードの下方の構造が 1 またはそれ以上の倍数で重複していることを意味する。この情報を用いて、複製される (重複した、以下同様) ノードの上方のトリーが組み合わせられる。

10

【0105】

図 29 で、複製されたノード 80 は X 1 である。従ってノード 80 の上方のトリー、すなわち A および B が組み合わせられることができる。これは、図 30 のトリーと図 28 の並列の F E T 構造を模倣する方程式 $(A+B)*(C+D)$ を与える。

20

【0106】

最後のケースは、並列の組換えに関してよく考えられる必要がある。図 31 の F E T 構造を考える。この F E T 構造について組み立てられる構造ベースのブール・トリーが最初に図 32 に表されている。複製されるノード 82、Y 1 の存在に注意されたい。しかし部分的トリー枝刈りプロセスの間に、相補的な枝が枝刈りされる (上述の部分的トリー枝刈り規則を参照されたい)。これは、複製されるノード (Y 1) を持つが、複製されるサブ・トリーをもたない図 33 のトリーをもたらす。このケースでは並列の組換えの試みを防ぐために、複製されるノード 82 の一つが名前を付け替えられる。例えば複製されるノード 82 は、Y 2 として名前を付け替えられる。

30

【0107】

トリーは方程式に変換されると (54)、発見的に簡略化される。方程式は、行われるべき簡略化が残らなくなるまで、一連の象徴的な簡略化を通る。これらの簡略化は、方程式の構造上の一致に重要な影響を与えない。好ましい簡略化規則グループは、以下の通りである。

【0108】

【数 2】

(注: 「op」はOR(「+」)またはAND(「*」)演算のいずれもありうる)

1. $X * (X \text{ op } B) \Rightarrow X * (1 \text{ op } B)$
 $X * (\sim X \text{ op } B) \Rightarrow X * (0 \text{ op } B)$
 $X + (X \text{ op } B) \Rightarrow X + (0 \text{ op } B)$
 $X + (\sim X \text{ op } B) \Rightarrow X + (1 \text{ op } B)$
2. $X \text{ op } (Y \text{ op } Z) \Rightarrow X \text{ op } Y \text{ op } Z$
3. $X * \sim X \Rightarrow 0$
 $X + \sim X \Rightarrow 1$
4. $X \text{ op } X \Rightarrow X$
5. $0 + X \Rightarrow X$
 $1 + X \Rightarrow 1$
 $0 * X \Rightarrow 0$
 $1 * X \Rightarrow X$
6. $X * Y + Y \Rightarrow Y$
 $(X + Y) * Y \Rightarrow Y$

10

20

上記の構文の簡略化に加えて、関数の簡略化も同様に行われる。例えばNAがAの反転である場合、NAが $\sim A$ に置き換えられ、Aが $\sim NA$ に置き換えられることができる。さらに、BAがNAの反転であるならば、BAおよびAは同等であり、入れ替えられる($X \text{ op } X \Rightarrow X$ 規則を使用して、 $A + BA \Rightarrow A$)。

【0109】

すべての論理信号について論理方程式(弱い方程式と弱くない方程式の両方)を決定した(54)後、ゲートモデル・ネットリストが抽出されうるように、十分な情報がデータベースにアサートされる。しかし、さらに論理関数を決定すること(62)が有利である。論理関数の決定(62)は、抽出されるゲートモデルを大いに簡略化する抽象化である。

30

【0110】

単純な相補形CMOS論理の場合、論理信号の1-および0-方程式が同等であり、ゲートが信号1-または0-方程式のどちらからも導き出されうることを理解することは容易である。プリチャージ・モデル(後述される)を抽出するときのように、ある信号についてゲートモデルが使用される他のケースもある。しかし、ゲート変換がより複雑であるケース(すなわち、論理信号がチャージを蓄えることができる時)もある。これらのケースで、簡略化されたFET実現が行われる。簡略化されたFET実現は、全体の0-方程式を、一つのn-FET(引き下げFET)を駆動する1つまたは複数のゲートと置き換えること、および全体の1-方程式を、一つのp-FET(引き上げFET)を駆動する1つまたは複数のゲートと置き換えることを含む。

40

【0111】

方法はいくつかの発見的手法を実行して、ある信号について弱い論理関数と弱くない論理関数の両方を決定する際に、1-方程式、0-方程式、あるいは両方の方程式(簡略化されたFETの実現)のどれが使用されるかを決定する。以下の規則は、前向きチェーンまたは後ろ向きチェーンのどちらでもありえる。

【0112】

1-および0-方程式が同一であるならば、0-方程式が論理関数を決定するために使用される。1-方程式は、完全に相補形のゲートが0-および1-方程式の合致をもつように、ド・モルガン形式で保有されることを思い出されたい。図36を参照して、信号Zについての方程

50

式は以下の通りである。

【0 1 1 3】

【数 3】

0-方程式 : $A * B$

1-方程式 : $A * B$

ルートした1-および0-方程式が同一である場合、0-方程式が論理関係を決定するために使用される。典型的な論理方程式は、他の論理信号に関して定義される。しかしルートした方程式は、他の論理信号ではなくポートに関して、その方程式を拡張しようとするものである。例えば、図 3 7 の信号GTOUTについて以下の方程式を考える。

【0 1 1 4】

【数 4】

0-方程式 : $NGTIN * EQ + B * NA$

1-方程式 : $(NGTIN + NEQ) * (B + NA)$

$NA = \sim A$ 、 $NB = \sim B$ および $NEQ = \sim EQ$ がわかっているとしても、2つの方程式が合致することは明らかではない。しかし方程式が拡張され、EQについての論理が方程式に挿入される場合($EQ = A * B + \sim A * \sim B$)、上記の方程式は次のようになる。

【0 1 1 5】

【数 5】

ルートした0-論理 : $NGTIN * A * B + NGTIN * NA * NB + B * NA$

ルートした1-論理 : $NGTIN * B + NGTIN * NA + B * NA$

ルートした方程式は同一であり、同じ方程式に簡略化する。証明は与えられないが、ターム $B * NA$ を用いて、ターム $NGTIN * A * B$ が $NGTIN * B$ に簡略化することに注意されたい。

【0 1 1 6】

F E T および非 F E T インスタンスが論理信号を駆動している場合、簡略化される F E T の実現は両方の論理方程式を使用して行われる。この規則が適用可能な状況は図 3 8 に示されており、ここで信号 Z は非 F E T インスタンス、例えば ADD4 によって駆動されることができる。

【0 1 1 7】

1-方程式がまさに(NMOS設計のような)引き上げ F E T を表すならば、論理が抽出され、0-方程式がゲートとして実現される。引き上げ F E T は、常に 0N である弱い p - F E T (ゲート接地を持つ)として認識される。図 3 9 で、信号 Z についての1-方程式は引き上げ F E T を含み、従って0-方程式($A * B$)がゲートとして実現される。

【0 1 1 8】

論理信号が、Andされるプリチャージ(Anded-Precharge)またはAndされる放電(Anded-DischARGE)のケースであり、クロックされるプリチャージ/チャージをもつポートではなく、また記憶ノードではない(すなわち弱い帰還インバータが存在する場合、ゲートは、プリチャージ/チャージの他方の側の方程式を使用して抽出される(例えばAndされるプリチャージ(Anded-Precharge)のケースでは、0-方程式を使用する)。Andされるプリチャージ(Anded-Precharge)のケースでは、0-方程式は、余分なタームがANDされることを除いて1-方程式と同じである。例えば、図 4 0 のAndされるプリチャージ(Anded-Precharge)回路の信号 Z は次の方程式を持つ。

【0 1 1 9】

【数 6】

1-方程式 : PRE

0-方程式 : $A * B * PRE$

回路は、入力 A、B および P R E、および出力 Z をもつ N A N D ゲートとして抽出される。論理信号がポートである場合、この規則は発火しないようにされる。リーフセル(leaf cell)設計者は、その値に影響を与えることができる信号に外部 F E T をフック(hook)することが知られているからである。論理信号が記憶ノードの一部分である場合、ゲートは抽出されない。その代わり方程式はラッチのセット/リセット論理の一部分になる。

10

20

30

40

50

【0120】

論理信号が、プリチャージ(Precharge)または放電(Discharge)のケースであり、ポートではなく、また記憶ノードではない(すなわち弱い帰還インバータが存在する)場合、ゲートはプリチャージ(Precharge)/放電(Discharge)の他方の側の方程式を使用して抽出される(例えばプリチャージ(Precharge)のケースでは、0-方程式を使用する)。この規則は、プリチャージ(Precharge)/放電(Discharge)のケースがプリチャージャー(precharger)/ディスチャージャー(discharger)にクロック信号になることを要求し、他方の側でANDされることを要求されるタームはないことを除いて、上記のAndされるプリチャージ(Anded-Precharge)/Andされる放電(Anded-discharge)規則と同様である。例えば、プリチャージ(Precharge)のケースは図41に示されており、そこで信号Zについて方程式は次式で示される。

10

【0121】

【数7】

0-方程式： $A * B$

1-方程式： NCK

またこの規則は、別の論理信号からの転送FETとともにクロック・プリチャージャーのケースを支持する。例えば図42について考える。そこでCKはZをプリチャージしており、信号Bは、AからZまで転送n-FETをゲートしている。これは、結果として生じる実現がNANDゲートであるプリチャージ(Precharge)のケースである($Z = \sim (X * \sim A)$)。

【0122】

20

論理信号が3状態(tristate)のドライバであって単純な転送FETではない場合、3状態のゲート実現が使用される。3状態のドライバは図43で示されるように、論理信号についてルートした方程式が次の形式で示されるものである。

【0123】

【数8】

ルートした0-方程式： $\sim DATA * EN$

ルートした1-方程式： $\sim DATA + \sim EN$

(ここで、ENはイネーブル信号である)

論理信号が3状態のCMOSドライバ(図44を参照)である場合、CMOS FETの実現が行われる。この規則は、2つのイネーブル(E1およびE2)が相補的である必要がないという事実がなければ、3状態のドライバに類似している。この規則の条件での間違った合致を制限するために、論理信号は一つの引き上げおよび引き下げFETによって駆動される必要がある。

30

【0124】

論理信号が「クロックされる論理」であって、3状態のドライバでない場合、実現は転送FETに従うゲートを使用して行われる。クロックされる論理では、他の方程式で相補されるタームのすべてが削除される場合に1-および0-方程式が安定する。例えば図45で、以下の信号Zの方程式はクロックされる論理を表す。

【0125】

【数9】

0-方程式： $D * (EN1 + EN2)$

1-方程式： $D + (\sim EN1 * \sim EN2)$

EN1およびEN2は0-方程式と1-方程式の間で相補的であり、(EN1およびEN2の全てのタームが取り除かれる)残りの方程式は、自明であるが安定している。クロックされる論理は転送FETとして実現され、除去される「相補形式」論理(上記の例のEN1およびEN2の形式)によってゲートされ、(相補形式が除去される)残りの安定した論理によって駆動される。図45の例で、結果として生じる実現は(EN1+EN2)によってゲートされる転送FETであり、Dのインバータ(図46)によって駆動される。

40

【0126】

論理信号がシングルエンドのクロックされる論理のケースであり、それが記憶入力ノード

50

(すなわちラッチへの入力)であり、入力がシングルエンドであるならば、適当な実現は、クロックされる論理を実現する論理ゲートからの転送 F E T である。シングルエンドのクロックされる論理で、方程式は次の形式である。

【0127】

【数10】

1-方程式：A

0-方程式：CK + A

図47の信号Zはこれらの方程式から導き出される。両方の方程式は、一方の側の直列のクロック・タームがなければ、同じものである(クロック・タームが1-方程式にあれば、類似の状況が存在する)。論理信号は、Aを生成する1つまたは複数のゲート(Aがただ一つの信号である場合はインバータ)を用いて実現され、CKによって記憶ノードにゲートされる転送 F E T に従う。図47についての実現が、図48に示されている。入力はシングルエンドであることが要求され、そうでないとこの規則は発火されない。シングルエンドとは、クロック・タームがoffであるので、全部の信号が(両側で)offである間、「非活動に」なることが保証されていることを意味する。クロック・タームがonである間、実現は相補形論理ゲートと全く同様であることに注意されたい。

【0128】

論理信号が示される記憶ノードであるならば、それはF E Tとして実現される。示される記憶ノードは、記憶入力ノードでないポートであり(すなわち、交差結合されたインバータのような定義されたラッチではない)、ただ一つの方程式をもち(他方の側はF E Tトリ

【0129】

論理信号が記憶入力ノードではないが、3状態であると知られているポートからの転送ゲートを持つ場合には、それはF E Tを用いて実現される。主要な3状態が可能なポート・インスタンスは、それが入力または双方向のポートであり、それを駆動すると知られているF E Tが存在する時である(データベースのF E T方向から取られる)。図50の論理信号Zは、信号Yが3状態可能なドライバであると決定されるとき、この規則の必要条件を満足する。このケースでは(および論理関数上の適用可能な規則がないとき)、信号が3状態になるように、論理信号への転送ゲートをF E Tとして実現することが好ましい。

【0130】

論理信号がポートであり、一つのレールにのみ引くF E Tがあり、ポートは厳密に出力であるかまたは他の可能なレシーバをもたない双方向であるかのどちらかである場合、F E T実現が行われる。この規則の発火を引き起こすケースについて図51を参照されたい。ポートがレシーバと双方向である場合、ユーザは、F E Tが一体必要とされるかを理解するように促されることがありえる。

【0131】

論理信号が両側の論理方程式をもつ入力または双方向ポートであって、記憶ノードではない場合、F E T実現が行われる。この規則は、ゲートが入力ポートに対向して駆動するように(それはある試験装置に問題を引き起こす)使用されないようにする。図52でこの規則は、ゲートAおよびBが入力ポートINに対向して駆動することを防ぐ。

【0132】

論理信号が他の論理信号から転送F E Tへ接続されるだけであり、それが記憶ノードでないならば、F E T実現が行われる。これは、図53の信号Zについて示されるような変換関数を表すための最も自然な方法である。

【0133】

論理信号の一方の側の方程式が1または0のどちらかであり、他方の方程式が反対(ドライブファイトを示す)でなければ、ゲート実現(例えばバッファ)は常に1(「通電している(hot)」側)である方程式を使用して行われる。図54は、VDDの反転である信号Zの実現を示す。信号Zの0-方程式は、VDDに結びつけられるゲートをもつF E Tのために常に「

10

20

30

40

50

通電している」。

【0134】

弱い帰還がゲート実現をもつ論理信号について存在するならば(上述の規則の1つからの実現による)、その弱い帰還は、ゲートモデル・ネットリストから切り捨てられる。方法によって弱いと認識される図55のFET F1は、実現から切り捨てられる。この規則は、単に電気的理由のためにだけ存在する帰還を切り捨てる。

【0135】

ルートした方程式が論理信号の一方の側について生成されることができないならば、その側は切り捨てられ、論理関数はその側について生成されない。図56の信号Zの0-方程式は、Yを論理信号にもってくる道がないので切り捨てられる。標準的な方程式が論理信号について存在するが、ルートした方程式が生成されることができない場合、論理的な矛盾が見つけれられる($A * \sim A$ のように)。

10

【0136】

弱い帰還がインバータの一方の側だけに一つのFETを含む場合、弱い帰還は無視される(切り捨てられる)。図57の実現で、弱い帰還インバータであるF1は無視される。弱い帰還は、単に電気的目的(例えば信号を「支える」)のために存在する。

【0137】

記憶ノード(交差結合されたインバータのような)が、FET論理によって前後両方で設定され(図58の記憶ノードのFET F1およびF2のように両方のインバータの入力で設定され)、設定方程式の一方が他方を包含する場合、包含される方程式は切り捨てられる

20

【0138】

論理信号が記憶ノードであり、弱い帰還が無視されなかった場合、セット/リセット・ラッチを示すFET実現が行われる。

【0139】

上述の論理関数規則が論理信号を実現することができない場合、ユーザーが促される。ユーザーは、1)実現のために1-または0-方程式のどちらかを選択し、それによってゲート実現をもたらす、2)FET実現を選択する、3)方程式を切り捨て、論理関数の実現を全体で不要にする、という選択を与えられることができる。これらの選択は、弱い論理信号と弱くない論理信号の両方に関連して与えられる。

30

【0140】

特性を入力ネットリストに加えて、上で述べられた規則の実現を受け付けないようにしてもよい(この方法の全ての規則によくあることである)。

【0141】

論理関数が決定され(62)、各々の論理信号について使用される実現のタイプがわかると、論理関数がゲートとしてマップされる(64)。

【0142】

1-または0-方程式だけを使用してその関数を決定する各々の論理信号の場合、ゲートは示されるゲート出力の反転があること記憶して、直接にマップされることができる。例えば、0-方程式 $Z = A * B$ を使用する実現は、入力AおよびBをもつNANDゲートを生成する。弱い方程式から実現されるゲートは、そういうものとしてマークされる(出力ネットリストの強さの指定を含む)。

40

【0143】

FETを使用して実現される各々の論理信号の場合には、1-および0-方程式が、それぞれp-FETおよびn-FETに置き換えられる。そしてゲートが方程式から生成されて、各々のFETゲート信号を駆動する。転送FETの場合には、転送FETが来る各々の論理信号の方程式が分離され、別々に実現される。この規則を使用して行われる模範的な変換が図59に示されており、そこで信号Zは、p-FETおよびn-FETを駆動するゲートを用いて実現される。

【0144】

50

(実現される)論理関数をゲートとしてマップした後、ゲートは以下の規則を使用して最適化される。

【0145】

ゲートの実現 $(A * \sim B) + (\sim A * B)$ は X O R ゲートとして認識され、そのようなものによって置き換えられる。

【0146】

F E T がラッチの両端を駆動する(すなわち、帰還インバータの両端を駆動する F E T がある)場合、それが生じる信号を反転させることによって一端から他端に F E T を動かす試みがなされる。これを行うときの制約は、1) その駆動 F E T が動かされると考えられる弱い帰還ゲートは、インバータであるか、または帰還グループの外のゲートによってその出力を受け取らせるかのどちらかである(そうでないと、その関数はそれを動かすことによって改悪されうる)、および 2) ラッチの端を駆動する F E T トリーは双方向でありえない(RAMセルによくあることである)、ということである。F E T を動かすことによって、ラッチの端を駆動する F E T トリーに冗長をつくる場合、それは取り除かれる。この規則を使用して実行される模範的な最適化が、図 60 に示されている。F E T F 2 の動きに注意されたい。

10

【0147】

チェーンが奇数の長さであるならば、インバータのチェーンは一つのインバータに納められる。チェーンが偶数の長さであるならば、インバータのチェーンは 1 つのバッファに納められる。

20

【0148】

インバータが全ての入力を単純な論理ゲートに駆動するならば、ゲートおよびインバータは、全ての入力の反転された意味を使用して、それらの論理「二重(dual)」によって置き換えられる。例えば、N O R ゲートへの全ての入力が反転されるならば、N O R ゲートおよびインバータは反転された入力を使用して A N D ゲートによって置き換えられる(図 61)。この規則を使用することに対する制約は、どの入力もゲートを通る帰還と関係し得ないということである。

【0149】

バッファは削除され、以下の一つが真であることが与えられる(それによって、バッファは、両側の可能な 3 状態のノードの目前で保持されるようになる)。1) 出力はポートでなく、F E T によって受け取られず、記憶ノードでない。2) 入力ポートでなく、F E T によって受け取られず、記憶ノードでない。

30

【0150】

ゲートおよび以下のインバータ(中間信号を受け取るゲートをもたない)は、相補的なゲートによって置き換えられる。例えば、インバータ(N O T ゲート)に続く N A N D ゲートは、A N D ゲートによって置き換えられる。

【0151】

ゲート最適化の後に使われないで残されるゲートは、出力ネットリストから除去される。

【0152】

多くの特性を、出力ネットリストのインスタンスおよび信号に加えることは有用である。特性は、弱いゲート、記憶ノード、相補的入力のような信号に関わる仮定(上述された N < s i g > 取り決め)、プリチャージ、その他の識別を含むことができる。

40

【0153】

上で実現されるゲート(ゲート・プリミティブ)は、当業者によって他のゲート・プリミティブにマップされてもよい(66)。

【0154】

上で開示される方法の保守的な性質は、非常に正確なゲートモデル(すなわち出力ネットリスト)の抽出をもたらす。通常 F E T およびポートの方向は推論されうるが、そうすることができない場合、それらの方向はユーザーからリクエストされる。方法は、大胆な仮定を作るのではなく、事実を推論するように設計される。

50

【0155】

F E Tモデル、および上述の方法を使用してそこから抽出されるゲートモデルは、図6 2および図6 3にそれぞれ示されている。

【0156】

本発明は例として次の実施態様を含む。

(1) a) F E Tモデル回路の電源、グラウンドおよびクロック信号を認識するステップ(44)と、

b) F E Tモデル回路のインバータを認識するステップ(46)と、

c) F E Tモデル回路のすべての論理信号を認識し、保存するステップ(48)と、

d) 認識される論理信号の各々について、一つまたは複数の構造ベースのブールの部分的トリーを組み立てるステップ(50)と、 10

e) 一つまたは複数のブールの部分的トリーを発見的に枝刈りするステップ(52)と、

f) 一つまたは複数のブールの部分的トリーから論理方程式を組み立てるステップ(54)と、を含む、F E Tモデル回路からゲートモデル回路を抽出する方法。

【0157】

(2) さらに、一つまたは複数のブールの部分的トリーを組み立てるに先立ち、各々のF E Tについて最初のF E T方向を決定するステップ(58)を含む、上記(1)の方法。

(3) さらに、一つまたは複数のブールの部分的トリーを組み立てるに先立ち、冗長なF E Tを削除するステップ(56)を含む、上記(2)の方法。

(4) さらに、一つまたは複数のブールの部分的トリーから論理方程式を組み立てるに先立ち、弱い部分的トリーを弱くない部分的トリーから分けるステップ(60)を含む、上記(3)の方法。 20

(5) さらに、共通の論理信号にルートした一つまたは複数の弱いおよび弱くないサブセットのそれぞれについて、論理関数を決定するステップ(62)を含む、上記(4)の方法。

(6) さらに、上記(5)で決定される論理関数を、対応するゲート・プリミティブにマップするステップ(64)を含む、上記(5)の方法。

【0158】

(7) 一つまたは複数のブールの部分的トリーを発見的に枝刈りする上記ステップ(52)は、

a) F E Tモデル回路のポートのポート方向を分解するステップ(72、78)と、 30

b) F E Tモデル回路のF E TのF E T方向を分解するステップ(70、76)と、

c) ポートとF E Tの両方の方向を分解するために、必要なところで、方法のユーザーを促すステップと、

を含む上記(1)の方法。

【0159】

(8) プログラマブル・コンピュータ(40)において、F E Tモデル回路をゲートモデル回路に変えるエキスパートシステムであって、

a) F E Tモデル入力ネットリストと、

b) 推論エンジンと、

c) 規則ベースと、 40

d) ユーザー入力と、

e) ゲートモデル出力ネットリストと、

を含む、上記エキスパートシステム。

【0160】

(9) 上記規則ベースが、

a) 少なくとも一つの発見的なトリー枝刈り規則ベース(68、74)と、

b) 少なくとも一つのポート方向規則ベース(72、78)と、

c) 少なくとも一つのF E T方向規則ベース(70、76)と、

を含む、上記(8)のエキスパートシステム。

【0161】

(10) 上記FETモデル入力ネットリストが、ネットリスト・インタープリタを含む、上記(8)のエキスパートシステム。

【0162】

【発明の効果】

本発明によると、FETモデル回路からゲートモデル回路を抽出する方法において、2つのモデル間の構造上の一致を保持し、抽出する時間を減少させることができる。

【図面の簡単な説明】

【図1】 FETモデルからゲートモデルを抽出する方法のフローチャートである。

【図2】 図1に示されるFETモデルからゲートモデルを抽出する方法のより詳細なフローチャートである。

【図3】 図1および図2の方法で使用される部分的トリートの枝刈り、FET方向、およびポート方向規則グループの間で起こるチェーンのフローチャートである。

【図4】 冗長な並列のFET Aを示す回路図である。

【図5】 FET A、B、CおよびDの複雑なFETの冗長を示す回路図である。

【図6】 多様な論理および中間信号を示す回路図である。

【図7】 論理信号Zに接続される多様なFETを示す回路図である。

【図8】 図1または図2の方法によって図7の論理信号Zについて生成される部分的トリートを示す図である。

【図9】 唯一の論理信号Sに接続されている入力ポートへのすべての接続を示す回路図である。

【図10】 安定したNANDゲートによって駆動される信号を示す回路図である。

【図11】 示される方向と反対の方向をもつFET F2を示す回路図である。

【図12】 代替の論理経路によって包含される論理パスを示す回路図である

【図13】 一致しないゲート信号をもつFET AおよびNAをもつ論理パスの回路図である。

【図14】 グラウンドに接続されたFET Dを示す回路図である。

【図15】 論理信号YおよびZの部分的トリートに含まれるFET F1を示す回路図である。

【図16】 入力ポートから遠ざかる方へ駆動するFETを示す回路図である。

【図17】 出力ポートの方へ向かって駆動するFETを示す回路図である。

【図18】 信号Zを論理1へ駆動することが可能な唯一のFETであるFET F1を示す回路図である。

【図19】 マスター/スレーブ・レジスタの転送FET F1を示す回路図である。

【図20】 駆動されないポートAを示す回路図である。

【図21】 ポートZを駆動する安定した論理を示す回路図である。

【図22】 駆動されない入力ポートZを示す回路図である。

【図23】 出力ポートYを示す回路図である。

【図24】 双方向ポートZを示す回路図である。

【図25】 2路帰還を示す回路図である。

【図26】 3路帰還を示す回路図である。

【図27】 ポートZと信号Yの間の帰還に含まれる弱いFET F1を示す回路図である。

【図28】 CMOS相補形AND-ORゲートの下半分を示す回路図である。

【図29】 図28のゲート部分のブール(boolean)・トリートを示す図である。

【図30】 並列の組換えが図1および図2の方法によって実行されたあとの、図28のゲート部分のブール・トリートを示す図である。

【図31】 並列の組換えを受けないブール・トリートと関連する図28と同様のFET構造を示す回路図である。

【図32】 図31の回路図のブール・トリートを示す図である。

【図33】 図1および図2の方法によって枝刈りが実行されたあとの、図32のブール・

10

20

30

40

50

トリーを示す図である。

【図 3 4】図 1 および図 2 の方法が実現されうるコンピュータを示す図である。

【図 3 5】図 1 および図 2 の方法を実現するために図 3 4 のコンピュータをプログラムすることができる物理記憶媒体を示す図である。

【図 3 6】同一の 1-および 0-論理方程式をもつ信号 Z を示す回路図である。

【図 3 7】同一のルートした 1-および 0-論理方程式をもつ信号 Z を示す回路図である。

【図 3 8】非 F E T インスタンスによって駆動される信号 Z を示す回路図である。

【図 3 9】引き上げ F E T である 1-論理方程式をもつ信号 Z を示す回路図である。

【図 4 0】And されるプリチャージ (Anded-Precharge) 回路を示す回路図である。

【図 4 1】プリチャージ (Precharge) 回路を示す回路図である。

10

【図 4 2】クロック・プリチャージャーを示す回路図である。

【図 4 3】F E T モデルの 3 状態のドライバに対応するゲートモデルを示す図である。

【図 4 4】F E T モデルの 3 状態の C M O S ドライバに対応するゲートモデルを示す図である。

【図 4 5】クロックされる論理を示す回路図である。

【図 4 6】図 1 および図 2 の方法によって抽出される、図 4 5 のクロックされる論理のゲートモデルである。

【図 4 7】シングルエンドのクロックされる論理を示す回路図である。

【図 4 8】図 1 および図 2 の方法によって抽出される、図 4 7 の回路図のゲートモデルである。

20

【図 4 9】示される記憶ノードを示す回路図である。

【図 5 0】既知の 3 状態可能なポートからの転送ゲートをもつ信号 Z を示す回路図である。

【図 5 1】一つのレールのみへ引く F E T に接続されたポートを示す回路図である。

【図 5 2】1-および 0-論理方程式をもつ、記憶ノードでない論理信号を示す回路図である。

【図 5 3】転送 F E T のみに接続される論理信号 Z を示す回路図である。

【図 5 4】図 1 および図 2 の方法によって「熱い」側をもつ論理信号について抽出されるゲートモデルを示す図である。

【図 5 5】弱い帰還 F E T F 1 を示す回路図である。

30

【図 5 6】ルートした 1-論理方程式が導き出されることができない信号 Z を示す回路図である。

【図 5 7】別の弱い帰還 F E T F 1 を示す回路図である。

【図 5 8】F E T 論理によってその前後で設定される記憶ノードを示す回路図である。

【図 5 9】1-および 0-論理方程式の F E T を使用して実現される信号 Z について生成されるゲートモデルを示す。

【図 6 0】第 1 のゲート最適化プロセスを示す図である。

【図 6 1】第 2 のゲート最適化プロセスを示す図である。

【図 6 2】F E T モデル回路の回路図である。

【図 6 3】図 1 および図 2 の方法によって抽出される、図 3 6 の F E T モデル回路に対応するゲートモデル回路である。

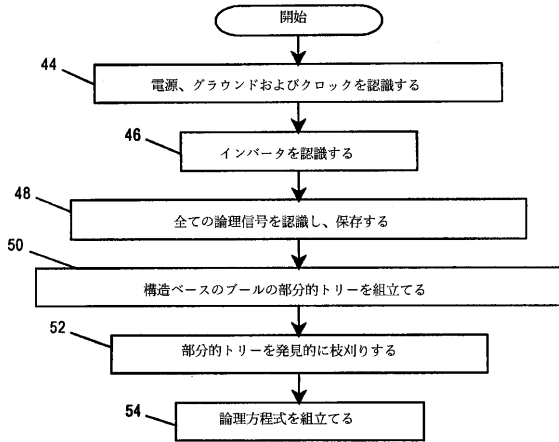
40

【図 6 4】第 1 の対の F E T 実現されるキャパシタを示す図である。

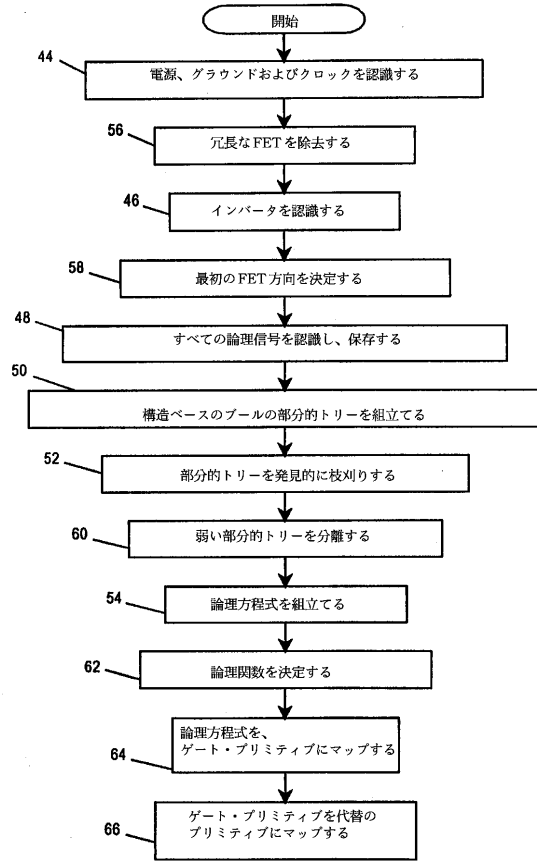
【図 6 5】第 2 の対の F E T 実現されるキャパシタを示す図である。

【図 6 6】2 路帰還を示す別の回路図である。

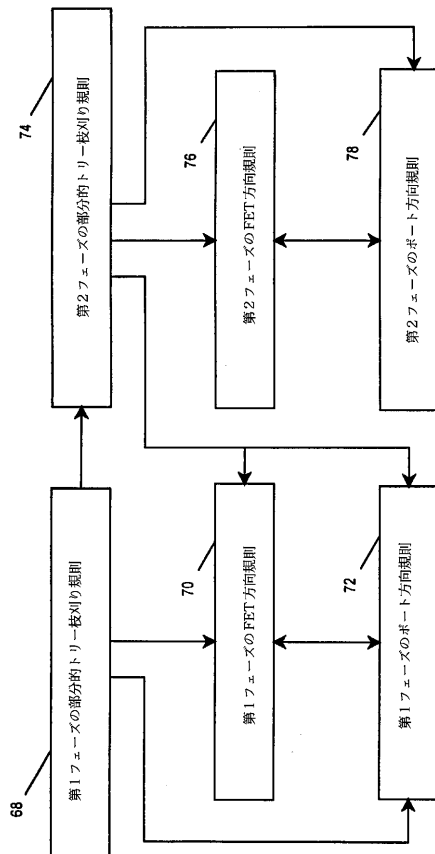
【図 1】



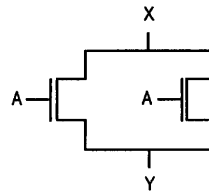
【図 2】



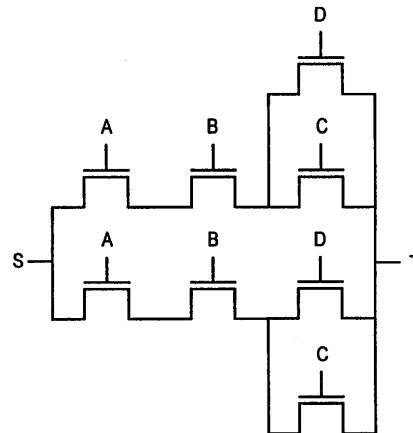
【図 3】



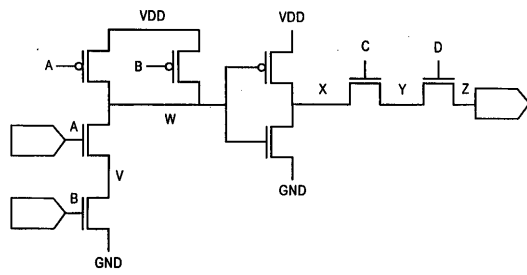
【図 4】



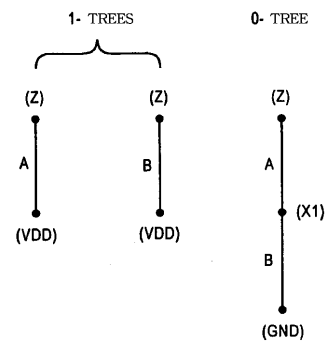
【図 5】



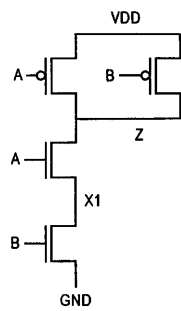
【図 6】



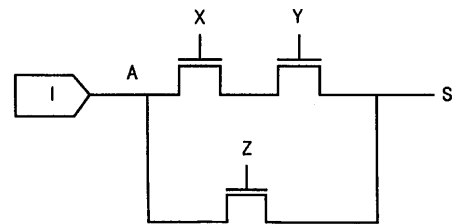
【図 8】



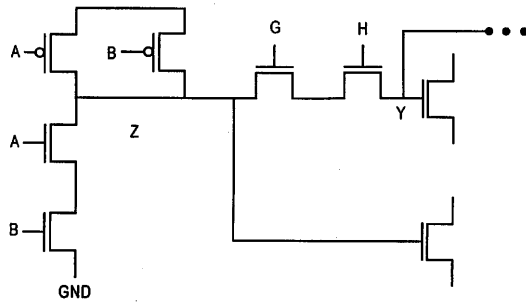
【図 7】



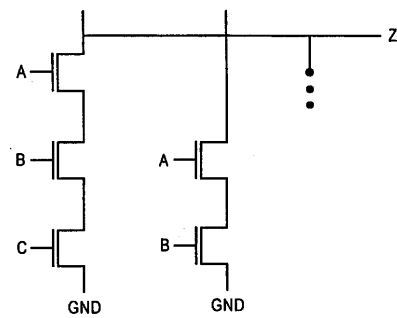
【図 9】



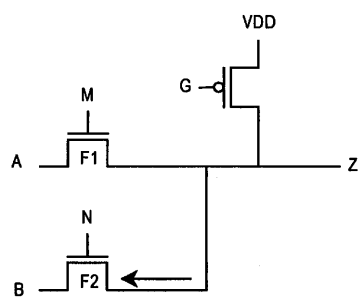
【図 10】



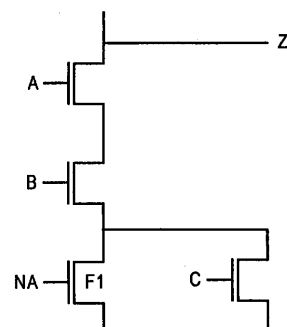
【図 12】



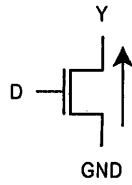
【図 11】



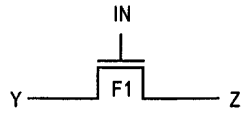
【図 13】



【図 14】



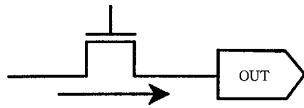
【図 15】



【図 16】



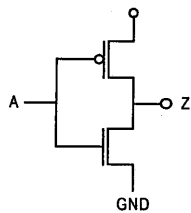
【図 17】



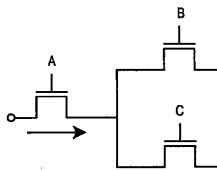
【図 20】



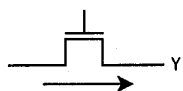
【図 21】



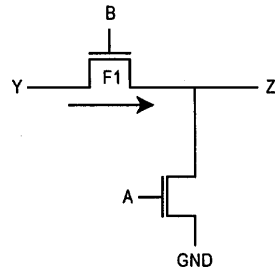
【図 22】



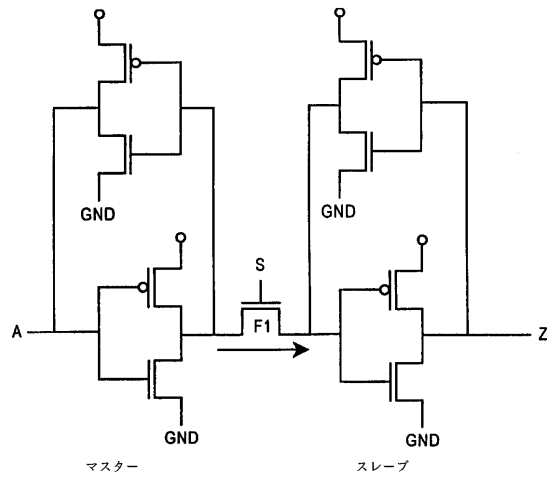
【図 23】



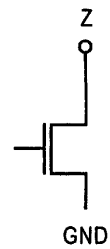
【図 18】



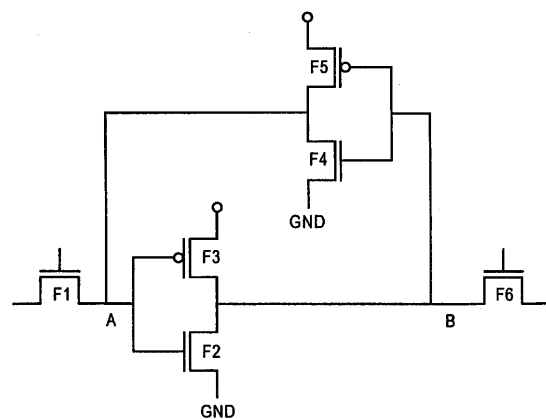
【図 19】



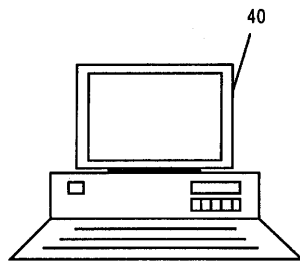
【図 24】



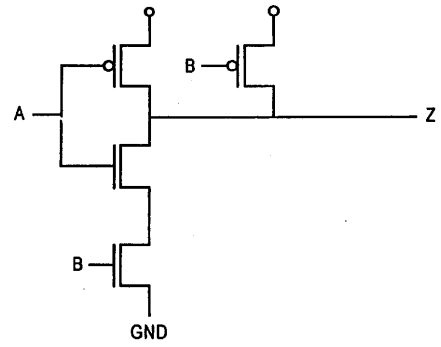
【図 25】



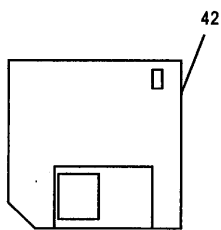
【図 3 4】



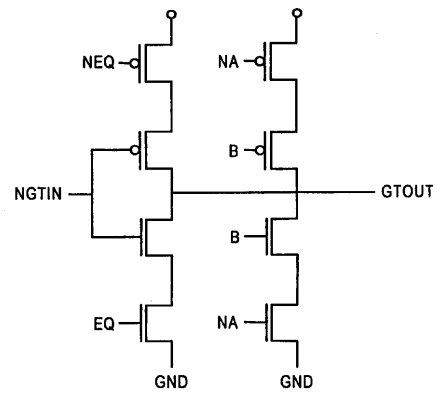
【図 3 6】



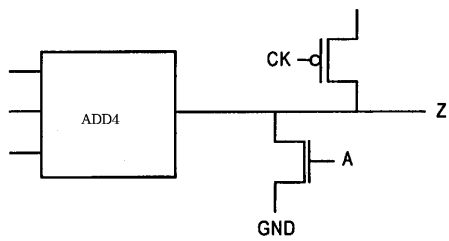
【図 3 5】



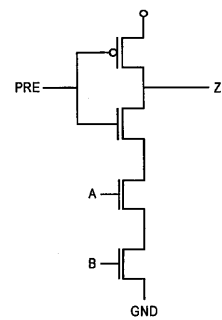
【図 3 7】



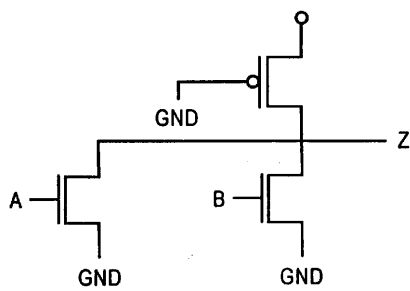
【図 3 8】



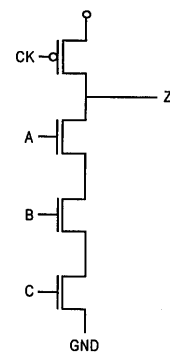
【図 4 0】



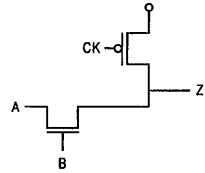
【図 3 9】



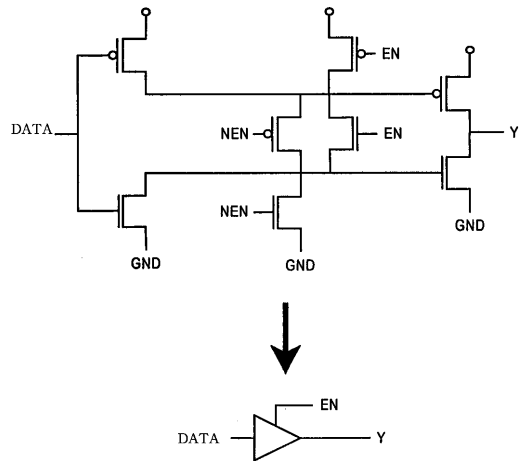
【図 4 1】



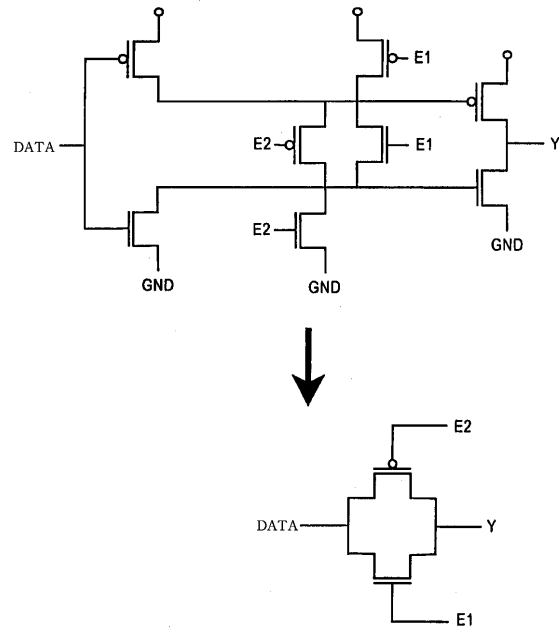
【図 4 2】



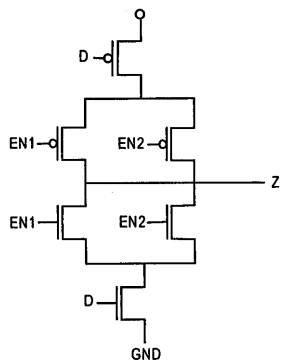
【図 4 3】



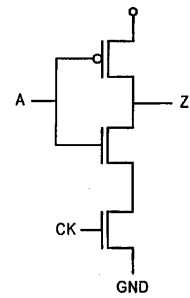
【図 4 4】



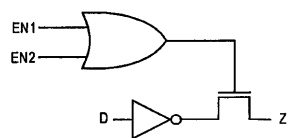
【図 4 5】



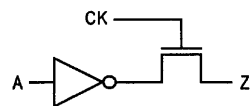
【図 4 7】



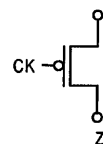
【図 4 6】



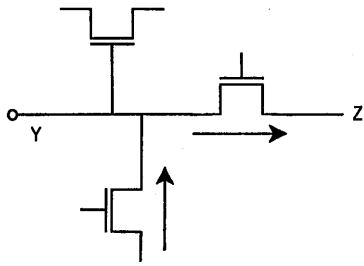
【図 4 8】



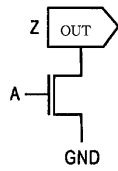
【図 4 9】



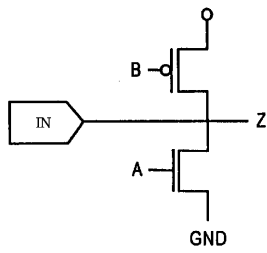
【図 50】



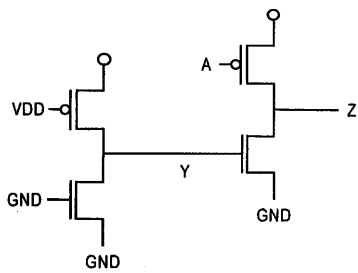
【図 51】



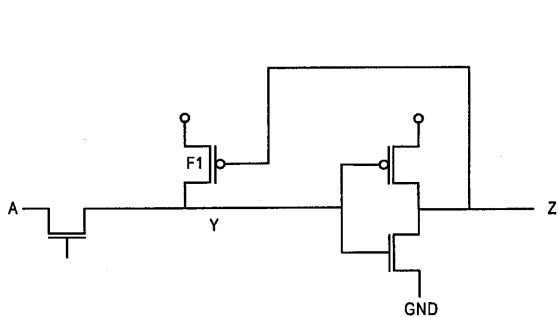
【図 52】



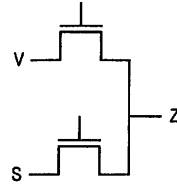
【図 56】



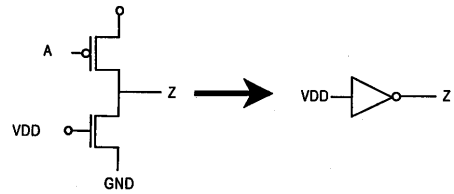
【図 57】



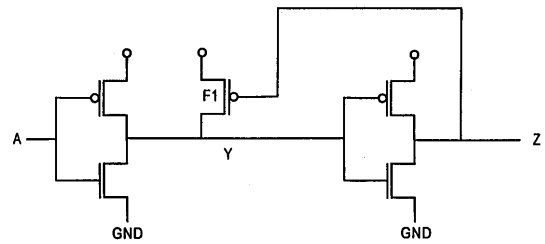
【図 53】



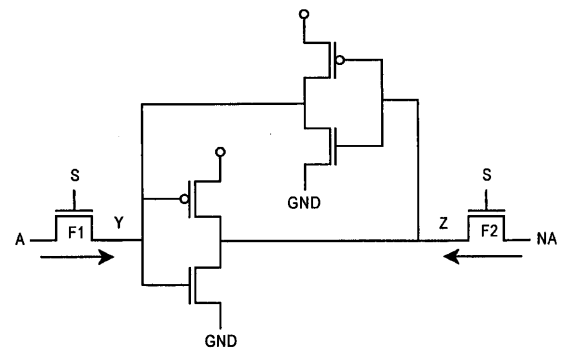
【図 54】



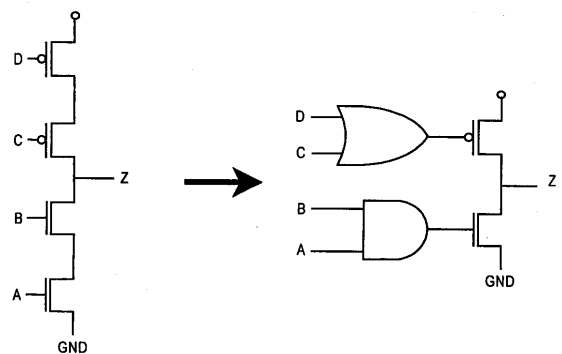
【図 55】



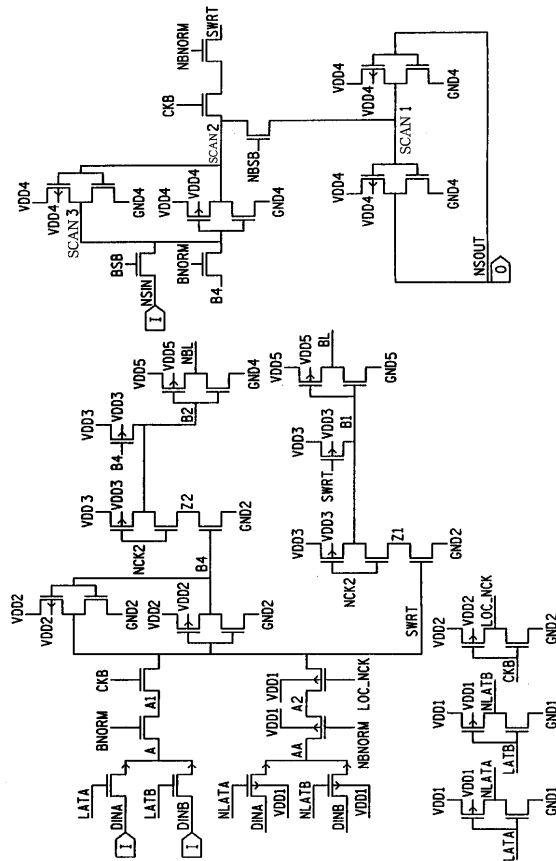
【図 58】



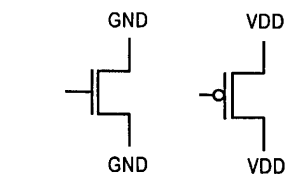
【図 59】



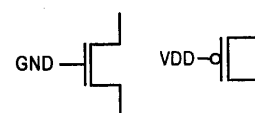
【図 6 2】



【図 6 4】



【图 6 5】



フロントページの続き

(72)発明者 ダリル・アルレッド

アメリカ合衆国80521コロラド州フォート・コリンズ、ウェコンダ・ドライブ 2602

審査官 早川 学

(56)参考文献 特開平03-092970 (JP, A)

(58)調査した分野(Int.Cl., DB名)

G06F 17/50