疑似3次元サブシステム分割冗長VLSIの救済可否判定アルゴリズム

苫米地　宣　裕

An Algorithm for Checking Survival of Virtually 3-Dimensional Subsystem-Divided VLSIs with Redundancy

Nobuhiro TOMABECHI

Abstract

In the design of VLSI with redundancy, subsystem-dividing results in many merits. The subsystem-dividing is usually performed toward one direction. The virtually 3-dimensional subsystem-dividing has been presented in which a system is divided toward row direction, column directions, and a diagonal direction. This paper presents an algorithm for checking the survival of the virtually 3-dimensional subsystem-divided VLSIs.

Key words: algorithm/survival/virtually 3-dimension/subsystem-dividing/VLSI/redundancy

1. まえがき

ULSI/WSIの製作には，チップ面積の増大に伴う欠陥の増加が最大の問題であり，冗長構成を導入して欠陥救済を行うことが不可欠と考えられる[1-4]。冗長構成の導入に当たって，システム全体をいくつかのサブシステムに分割して，サブシステムごとに冗長化を行うと，種々の利点を生ずる[5,6]。

通常，サブシステム分割は，システムを一つの方向に分割する方法がとられる。これに対して，本研究者は，サブシステム分割を行方向と列方向に2次元に行う方法，さらには，一つの対角線方向を加えて3次元に行う方法（疑似3次元サブシステム分割）を提案してきた[6]。そして，2次元サブシステム分割，あるいは，疑似3次元サブシステム分割によれば，一つの方向に分割する場合よりも，VLSIの歩留りが向上することを示した。

しかし，これまでの報告では，疑似3次元サブシステム分割を行ったVLSIが救済可能か否かを判定する手続きの定式化が完全には行われていなかった。本論文では，この疑似3次元サブシステム分割VLSIの救済可否判定アルゴリズムを提案する。

2. 疑似3次元サブシステム分割VLSIのモデル

対象とするVLSIシステムのモデルを次のように設定する。

[条件1]　システムは，いくつかの非冗長な基本回路（以下，基本回路を単に回路という）から構成される。回路の機能，および，チップ面積はみな等しい。

[条件2]　システムは，非冗長な回路にいくつかの冗長な回路を付加して構成される。冗長な回路と非冗長な回路は，機能とチップ面積が等

平成8年12月10日受理
八戸工業大学 電気工学科　教授

しい。

[条件3]　冗長構成に必要なハードウェアは，冗長な回路以外は充分小さい。

[条件4]　システムは，行方向，列方向，および，対角線方向の三つの方向のサブシステムに分割される。冗長な回路は，一つの方向については，接続されているサブシステム内の任意の非冗長な回路と置き換えることができるが，他のサブシステム内の回路と置き換えることはできない。また，一つの回路は，どの方向の冗長な回路に置き換えるかを選択できる。一つの方向のサブシステムに接続される冗長な回路の個数は，サブシステムによらずみな等しい。

なお，「行方向にサブシステム分割する」とは，行方向の長さは変えないで列方向を細分化することを意味する。「列方向にサブシステム分割する」とは，列方向の長さは変えないで行方向を細分化することを意味する。

以下，列方向を x 方向，行方向を y 方向，対角線方向を z 方向とよぶ。x 方向に分割されるサブシステムの数を m_x，y 方向に分割されるサブシステムの数を m_y，z 方向に分割されるサブシステムの数を m_z と表わす。x 方向のサブシステム（x 方向に長いサブシステム）を S^x_1, S^x_2, ..., S^x_{mx}，y 方向のサブシステムを S^y_1, S^y_2, ..., S^y_{my}，z 方向のサブシステムを S^z_1, S^z_2, ..., S^z_{mz} と表わす。x 方向，y 方向，z 方向の各々のサブシステム1個に接続される冗長な回路の数を，それぞれ，R_x, R_y, R_z と表わす。

図1に疑似3次元サブシステム分割の例を示している。図は $m_x=m_y=m_z=3$ としている。z 方向のサブシステムは，メッシュの中にサブシステムの番号を記入して表わしている。

m_z の値は m_x と m_y に依存し，次のような関係が成り立つ。

$$m_x \geqq m_y \text{ ならば } m_z=m_x$$
$$m_x < m_y \text{ ならば } m_z=m_y$$

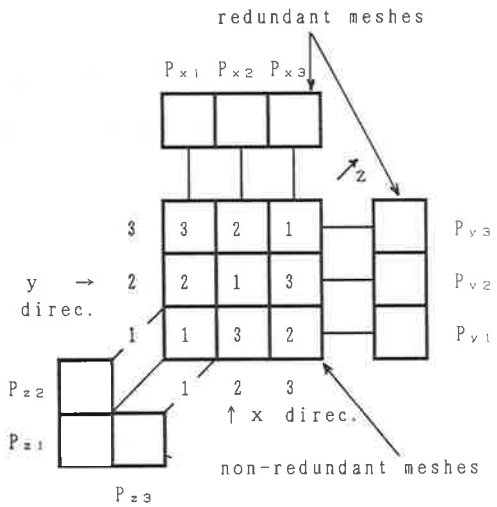

図1　疑似3次元サブシステム分割
（分割数は各方向とも3の例）

3. 救済可否判定法

まず，以下の論議に用いる用語を定義する。

[定義1]　メッシュ

x 方向と y 方向のサブシステムに共通に含まれるすべての回路の集合をメッシュとよぶ。x 方向のあるサブシステム S^x_p と，y 方向のあるサブシステム S^y_q により指定されるメッシュを $M_{p,q}$ と表わす。

[定義2]　冗長なメッシュ

一つのサブシステムに付加されるすべての冗長な回路の集合を冗長なメッシュとよぶ。x 方向の冗長なメッシュを P_{x1}, P_{x2}, ..., P_{xmx}，y 方向の冗長なメッシュを P_{y1}, P_{y2}, ..., P_{ymy}，z 方向の冗長なメッシュを P_{z1}, P_{z2}, ..., P_{zmz} と表わす。

[定義3]　ブロック

x 方向のいくつかのサブシステム，y 方向のいくつかのサブシステム，および，z 方向のいくつかのサブシステムに共通に含まれるすべてのメッシュの集合をブロックとよぶ。x 方向のサブシステム，S^x_{i1}, S^x_{i2}, ..., S^x_{ip}，y 方向のサブシステム，S^y_{j1}, S^y_{j2}, ..., S^y_{jq}, ...，z 方向のサブシステム S^z_{k1}, S^z_{k2}, ..., S^z_{kr}, で指定されるブロック

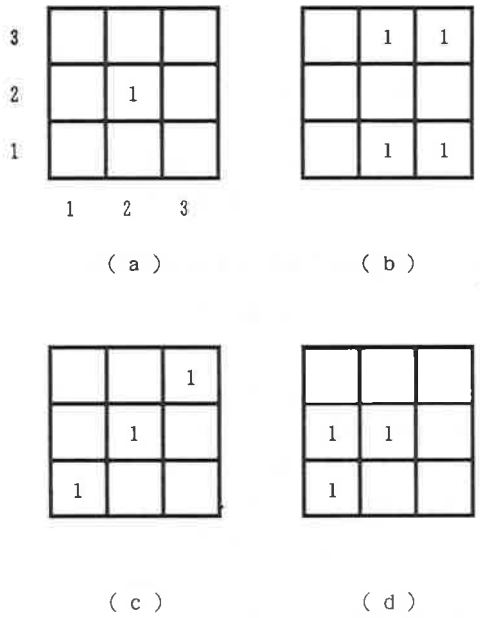

図2　ブロックの例（3×3の場合）

を $B_{i1, i2, \ldots, ip\,;\ j1, j2, \ldots, jq\,;\ k1, k2, \ldots, kr}$ と表わす。

図2に，3行3列の場合のブロックの例を4個示している。(a) は $B_{2\,;\ 2\,;\ 1}$ を，(b) は $B_{2,3\,;\ 1,3\,;\ 1,2,3}$ を，(c) は $B_{1,2,3\,;\ 1,2,3\,;\ 1}$ を，(d) は $B_{1,2\,;\ 1,2\,;\ 1,2}$ を，それぞれ表わしている。

［定義4］　方向交換

あるメッシュ内の一つの方向の冗長な回路に置き換えられている不良な回路を他の方向の冗長な回路に置き換えを変更する操作を方向交換という。

［定義5］　救済不能，救済可能

あるブロック（または，メッシュの集合，以下，同じ）に含まれるすべての不良な回路を，そのブロックに接続されている正常な冗長な回路に置き換えることができないとき，そのブロックは救済不能であるという。また，あるブロックに含まれるすべての不良な回路を，そのブロックに接続されている正常な冗長な回路に置き換えることができるとき，そのブロックは救済可能であるという。

対象とするVLSIシステムについて，以下のような性質が成り立つ。

［定理1］　不良な回路の置き換え

あるブロック内の不良な回路は，そのブロックに接続されている同じ数の冗長な回路に置き換えることができる。

(証明)　ブロック内の不良な回路を，任意に，そのブロックに接続されている冗長な回路に置き換える。その結果，いくつかの不良な回路とそれと同数の冗長な回路が置き換えられずに残ったとする。その中の一つの不良な回路 $C1$ と一つの冗長な回路 $C2$ をとる。$C2$ に接続されているブロック内の不良な回路に方向交換を行って，$C2$ を他の方向の冗長な回路 $C3$ に置き換える。再び，方向交換を行って，$C3$ を他の方向の冗長な回路 $C4$ に置き換える。この操作を，冗長な回路の方向が $C1$ の方向に一致するまでくり返す。これによって，$C1$ を冗長な回路に置き換えることができる。以上の操作をくり返すことによって，ブロック内のすべての不良な回路を，それと同数の冗長な回路に置き換えることができる。　□

［定理2］　ブロックの救済可能，救済不能の判定

あるブロック $B_{i1, i2, \ldots, ip\,;\ j1, j2, \ldots, jq\,;\ k1, k2, \ldots, kr}$ に含まれる不良な回路の数 D が，$D > pR_x + qR_y + \ldots + rR_z - B$ ならば，そのブロックは救済不能

$D \leqq pR_x + qR_y + \ldots + rR_z - R$ ならば，そのブロックは救済可能である。

ただし，B は，そのブロックに接続されているすべての冗長なメッシュ（すなわち，P_{xi1}, P_{xi2}, …, P_{xip}, P_{yj1}, P_{yj2}, …, P_{yjq}, P_{zk1}, P_{zk2}, …, P_{zkr}）の中の不良な冗長な回路の数を表わしている。

(証明)　ブロック $B_{i1, i2, \ldots, ip\,;\ j1, j2, \ldots, jq\,;\ k1, k2, \ldots, kr}$ に接続される冗長な回路の数は $pR_x + qR_y + \ldots + rR_z$ 個であり，この中で正常な回路は，$pR_x + qR_y + \ldots + rR_z - B$ 個なので，この数より多い不良な回路が救済不能なのは自明で

ある。また，ブロック内に最大で $pR_x+qR_y+\ldots+rR_z-B$ 個の不良な回路が存在するときは，定理1に基づき，この不良な回路すべてをブロックに接続されている $pR_x+qR_y+\ldots+rR_z-B$ 個の正常な冗長な回路に置き換えることができる。従って，$D \leqq pR_x+qR_y+\ldots+rR_z-B$ ならばそのブロックは救済可能である。　□

結論として，あるVLSIシステムが救済可能か否かの判定は，次のように行うことができる。

[定理3]　システムの救済可能の判定

システムに救済不能なブロックが存在しないとき，そのシステムは救済可能である。

(証明)　システム内に救済不能なメッシュの集合が存在すると仮定すると，そのメッシュの集合を含むすべての方向のすべてのサブシステムで指定されるブロックも救済不能となる。しかし，システム内に救済不能なブロックが存在しないならば，そのような救済不能なメッシュの集合も存在しないことになる。救済不能なメッシュの集合が存在しなければ，そのシステムは救済可能となる。　□

4. ブロック生成アルゴリズム

疑似3次元サブシステム分割VLSIが救済可能か否かの判定は，定理3に基づいて，あらゆる種類のブロックを求めて，各ブロックについて救済可否を判定すればよい。あらゆる種類のブロックを求めるには，原理的には，x 方向サブシステム，y 方向サブシステム，z 方向サブシステムのすべての組み合わせを求めればよい。この組み合わせの数 B_1 は次のようになる。

$$B_1=(2^{mx}-1)(2^{my}-1)(2^{mz}-1) \qquad (1)$$

しかし，その組み合わせの中にはメッシュを全く含まないブロックや，重複したブロックが含まれる。これは，x 方向と y 方向については，サブシステムは互いに独立に選択できるが，z 方向サブシステムは，x 方向，y 方向と相関関係を有するためである。この相関関係は，一般には，簡単な式で表現できないので，これまでは，図表を用いて経験的に求めるという方法をとってきた。従って，無駄なブロックの発生ができるだけ少なく，かつ，定式化されたブロック発生アルゴリズムが望まれる。本論文では，以下のようなアルゴリズムを提案する。

4.1　用語の定義とパターンの性質

はじめに，用語を定義する。

[定義6]　パターン

メッシュの集合をパターンという。パターンは，$m_x \times m_y$ の格子図に，1または0を記入して表わす。格子図の x 軸は x 方向サブシステムの番号を，y 軸は y 方向サブシステムの番号を表わす。従って，格子図の座標 (x, y) はメッシュ $M_{x,y}$ に対応する。パターンに含まれるメッシュに対応する座標には1を記入する。格子図は，$m_x \times m_y$ の配列 P で表わすこともできる。このとき，メッシュ $M_{x,y}$ は配列要素 $P(x, y)$ に対応し，パターンは，値が1となる配列要素の集合 $\{P(x, y) \mid P(x, y)=1, x=1\sim m_x, y=1\sim m_y\}$ で表わされる。

[定義7]　方形パターン

連続したいくつかの x 方向サブシステム，$S_i^x, S_{i+1}^x, \ldots, S_{i+p}^x$ と連続したいくつかの y 方向サブシステム，$S_j^y, S_{j+1}^y, \ldots, S_{j+q}^y$ に共通に含まれるすべてのメッシュの集合 $\{M_{x,y} \mid x=i\sim i+p, y=j\sim j+q\}$ を方形パターンという。

[定義8]　z 削除方形パターン

方形パターンから，いくつかの z 方向サブシステムに含まれるすべてのメッシュを除いたパターンを z 削除方形パターンという。

[定義9]　スプリット

パターンを x 座標のある位置 i において，i 以降のすべてのメッシュの x 座標を一つずらす操作，あるいは，同様の操作を y 座標について行う操作をスプリットという。x 座標のスプリットは，次のような配列要素間のデータの移動で表わされる。

$P(j,y)\rightarrow P(j+1,\ y)$ $(j=1\sim m_x-1,\ y=1\sim m_y)$

y 座標のスプリットも同様に表わされる。

[定義 10]　準方形パターン

方形パターンにいくつかのスプリットを行って得られるパターンを準方形パターンという。

[定義 11]　対称スプリット

スプリットを x 座標のある位置と y 座標のある位置で同時に行う操作を対称スプリットという。

[定義 12]　z 削除準方形パターン

z 削除方形パターンにいくつかの対称スプリットを行って得られるパターンを z 削除準方形パターンという。

[定義 13]　循環シフト

x 座標の循環シフトは，次のような配列要素間のデータの移動で表わされる。ただし，a mod b は，a を b で割った剰余を表わしている。

$$P(i,\ y)\rightarrow P((i+1) \bmod m_x,\ y)$$
$$(i=1\sim m_x\ \ y=1\sim m_y)$$

y 座標の循環シフトも同様に表わされる。　□

パターンについて，次のような性質が成り立つ。

[定理 4]　方形パターンの性質

$P(1,\ 1)=1$ となる方形パターン $\{M_{x,y}\mid x=1\sim p,\ y=1\sim q\}$ はブロックであり，$B_{1,2,\ \ldots,\ p\ ;\ 1,\ 2,\ \ldots,\ q\ ;\ 1,\ 2,\ \ldots,\ p,\ mz-q+1,\ mz-q,\ \ldots,\ mz}$ と表わされる。

(証明)　方形パターンの定義より自明である。　□

図3に，$P(1,\ 1)=1$ となる方形パターンの例を示している。

一般にブロックの場合は，ブロックを構成する x 方向サブシステム，y 方向サブシステムと z 方向サブシステムの関係は簡単には表現できないが，方形パターンの場合は，定理4のように簡単となる。

[定理 5]

方形パターン，準方形パターン，z 削除方形パターン，および，z 削除準方形パターンはいずれ

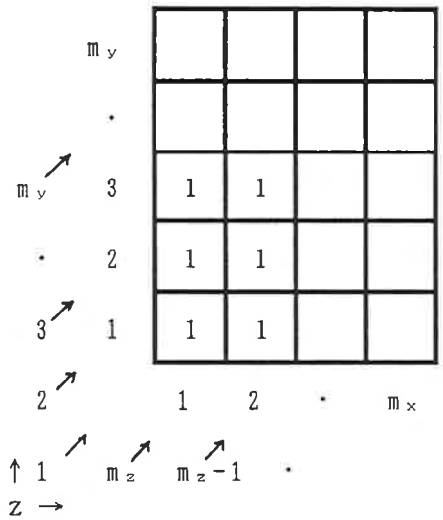

図3　方形パターンの例

もブロックである。

(証明)　方形パターンと準方形パターンがブロックであることは自明である。z 削除方形パターンは，元の方形パターンを形成するサブシステムからいくつかの z 方向サブシステムに含まれるすべてのメッシュを除いたものなので，元の方形パターンがブロックであるのと同様にブロックであることが自明である。また，z 削除準方形パターンは，対称スプリットする前の z 削除方形パターンに比較して，x 方向サブシステムと y 方向サブシステムの中でいくつかのサブシステムが入れ替わるだけであり，かつ，z 方向サブシステムは変わらないのでブロックである。　□

[定理 6]

方形パターン，準方形パターン，z 削除方形パターン，z 削除準方形パターンの各パターンを循環シフトして得られるパターンはブロックである。

(証明)　循環シフトによって，x 方向，y 方向，z 方向ごとに，サブシステムが入れ替わるだけで，各方向のサブシステムの数やメッシュの数は変わらないので，循環シフトする前のパターンがブロックならば，循環シフトした後のパ

ターンもブロックである。

［定理7］

方形パターン，準方形パターン，z削除方形パターン，z削除準方形パターン，および，以上のパターンを循環して得られるパターンがブロックのすべてである。

（証明）　一つの方形パターンP_0とP_0から得られる準方形パターン，z削除方形パターン，z削除準方形パターン，および，それらのパターンを循環シフトして得られるパターンは，P_0を形成するx方向，y方向，z方向それぞれのサブシステムの個数を一定として組み合わせを変えたすべてのブロックを含んでいる。従って，任意の方形パターンとその方形パターンから得られる準方形パターン，z削除方形パターン，z削除準方形パターン，および，それらのパターンを循環シフトして得られるパターンは，生じ得るすべてのブロックを含んでいる。　□

4.2　提案するブロック発生アルゴリズム

あらゆるブロックを効率的に発生するアルゴリズムは，次のように記述される。

［アルゴリズム1］　ブロック発生

［手順1-1］　$P(1, 1)=1$となる一つの方形パターンを選ぶ。これをPとする。

［手順1-2］　P自身，または，Pから得られる一つの準方形パターンを選ぶ。これが一つのブロックとなる。これをP_1とする。

［手順1-3］　P_1を循環シフトしてブロックを作成する。この手続きの詳細は，アルゴリズム2に示す。

［手順1-4］　Pから得られるすべての準方形パターンを作り終わったならば手順1-5に行く。さもなければ手順1-2に戻る。

［手順1-5］　Pから得られる一つのz削除方形パターンを選ぶ。これをP_2とする。

［手順1-6］　P_2から得られる一つのz削除準方形パターンを選ぶ。これが一つのブロックとなる。これをP_3とする。

［手順1-7］　P_3を循環シフトしてブロックを作成する。この手続きはアルゴリズム2に等しい。

［手順1-8］　P_2から得られるすべてのz削除準方形パターンを作り終わったならば手順1-9に行く。さもなければ手順1-6に戻る。

［手順1-9］　P_1から得られるすべてのz削除方形パターンを作り終わったならば手順1-10に行く。さもなければ，手順1-5に戻る。

［手順1-10］　$P(1, 1)=1$となるすべての方形パターンを作り終わったならば終了とする。さもなければ手順1-1に戻る。　□

循環シフトによるブロックの作成は次のようになる。

［アルゴリズム2］　循環シフト

［手順2-1］　対象となるパターンP_sのx座標を1座標循環シフトする。これがひとつのブロックとなる。これをあらためてP_sとする。循環シフトした結果，重複したパターンが得られたら終了とする。

［手順2-2］　P_sのy座標を1座標循環シフトする。これが一つのブロックとなる。これをあらためてP_sとする。循環シフトした結果，重複したパターンが得られたら終了とする。

［手順2-3］　y座標の循環シフトを(m_y-1)回行ったら手順2-4に行く。さもなければ手順2-2に戻る。

［手順2-4］　x座標の循環シフトを(m_x-1)回行ったら終了とする。さもなければ手順2-1に戻る。

以上提案したブロック生成アルゴリズムにおいて生ずるブロックの数B_2は次のようになる。

$$\begin{aligned} B_2=&(\text{準方形パターンの数}\\ &+z\text{削除方形パターンの数}\\ &\times\text{対称スプリットの回数})\\ &\times\text{循環シフトの回数}\\ \fallingdotseq&(2^{mx+my}+B_p m_z)(m_x-1)(m_y-1)\\ B_p=&\sum_{i=1}^{mz-1}\sum_{j=1}^{mz-i} {}_{mz-i}C_j \end{aligned} \qquad (2)$$

ただし，B_pはz削除方形パターンの数を，${}_aC_b$

は，a 個の中から b 個を選ぶ組み合わせの数を表わしている。

式(1)で示される従来の方法によるブロックの数 B_1 と提案したアルゴリズムのブロック数 B_2 を比較すると次のようになる。

$m_x=m_y=3$ の場合　$B_1=343$,　$B_2=304$
$m_x=m_y=4$ の場合　$B_1=3375$,　$B_2=2592$
$m_x=m_y=5$ の場合　$B_1=29791$,　$B_2=18464$

サブシステム分割数が大きくなるに従い，提案した方法のブロック数の方が相対的に少なくなることが分かる。

4.3 救済可否判定アルゴリズム

疑似3次元サブシステム分割を行ったVLSIの救済可否判定アルゴリズムは次のようになる。

[アルゴリズム3] VLSIの救済可否判定
[手順3-1] 一つのブロックを選ぶ。詳細は，アルゴリズム1に従う。
[手順3-2] そのブロックの救済可否の判定を行う。救済可能ならば，手順3-3に行く。救済不能ならば，VLSIが救済不能と判定し終了する。
[手順3-3] すべてのブロックについて判定を終わったならば，VLSIが救済可能と判定し終了する。調べていないブロックがあるならば，手順3-1に戻る。

5. むすび

本論文では，疑似3次元サブシステム分割を行ったVLSIが救済可能か否かを判定するアルゴリズムを提案した。

疑似3次元サブシステム分割は，分割数が多くなると救済可否判定に要する時間が飛躍的に大きくなるという問題点を有しており，その短縮が今後の課題となる。

なお，本研究は，平成8年度文部省科学研究費（基盤研究C）の補助を受けたことを付記する。

文　献

1) W.R. Moore, "A review of fault-tolerant techniques for the enhancement of integrated circuit yield," Proc. IEEE, vol. 74, pp. 684-698, May 1986.
2) I. Koren and D.K. Pradhan, "Yield and performance enhancement through redundancy in VLSI and WSI multiprocessor systems," Proc. IEEE, vol. 74, pp. 699-711, May 1986.
3) J.H. Kim and S.M. Reddy, "On the design of fault-tolerant two-dimensional systolic arrays for yield enhancement," IEEE Trans. on Comp., vol. 38, no. 4, pp. 515-525, April 1989.
4) 苫米地宣裕，"剰余数系に基づくULSI/WSI算術演算システムの欠陥救済方法，"電子情報通信学会論文誌D-I, vol. J73-D-I, no. 1, pp. 72-78, Jan. 1990.
5) 苫米地宣裕，金沢正治，"WSI規模高速FFTプロセッサの冗長化，"電子情報通信学会論文誌D-I, vol. J80-D-I, no. 1, pp. 110-120, Jan. 1997.
6) 苫米地宣裕，"多次元サブシステム分割冗長VLSI，"電子情報通信学会技術研究報告，vol. FTS96-51, pp. 25-32, Dec. 1996.