DA 27－2
（1975. 11. 19）

ＭＯＳゲート列レイアウトプログラム

吉沢　仁，藤波義忠，川西　宏，可児賢二
（日本電気株式会社）

（概　要）

ＭＯＳＬＳＩのレイアウトの一方法として，論理ゲートを縦の列，信号線を横の行により実現する方式（ゲート列構成レイアウト方式と呼ぶこととする）がある。本文ではこの方式によるＬＳＩのチップ面積をできるだけ少なくするように各ゲートの配列を定め，引続いて配線径路決定を行うプログラムＢＬＯＯＭ（ＢＬＯＣＫ　ＯＲＤＥＲＩＮＧ　ＡＮＤ　ＭＡＰＰＩＮＧ　ＰＲＯＧＲＡＭ）の構成および評価について述べる。本プログラムにより，人手設計に比べ平均１５％のチップ面積縮少が可能である。

1.　はじめに

論理回路をＭＯＳＬＳＩで実現する場合，図１に示すように（ａ）のＮＡＮＤゲートを（ｂ）のようなＭＯＳ回路で構成し，チップ上のレイアウトとしては，抽象化すれば（ｃ）のような基本単位を配列する手法がしばしば用いられる。[1) 2)]
すなわち，１列のレイアウト基本単位で１ゲートを構成するわけである。

このレイアウト方式は，チップの一部に図２のように単純に１ゲート分を１列に並べる，あるいは図３のように図２を向いあわせるようにして１列で２ゲートを構成して並べる，のように用いられることが多い。

ゲート列が占めるチップの面積を少なくするためには，図２において信号配線（同一系列の配線は直線で実現するものとする）に必要な行の数（ℓ）を少なくするように各ゲートの配列を定めることが必要である。例えば，図２の例はゲー

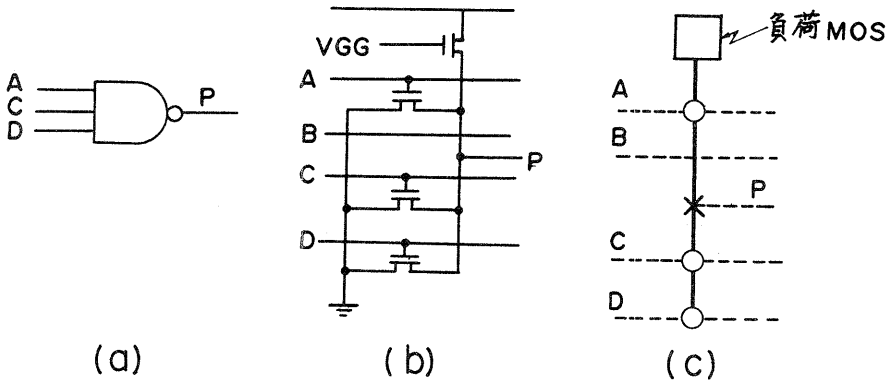

図１　１列で１ゲートを構成するＭＯＳＬＳＩの基本単位

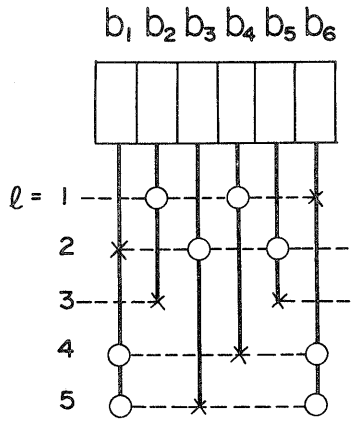

図２　１ゲート列構成

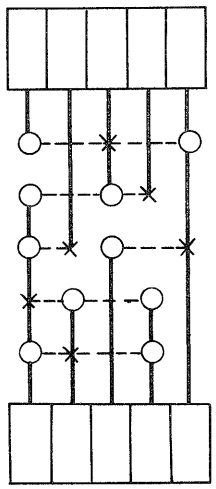

図３　２ゲート列構成

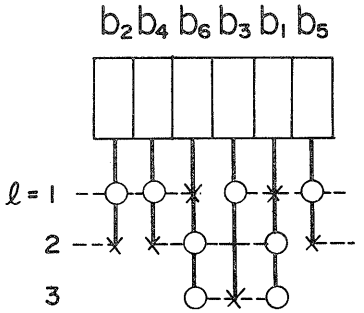

図４　列の入れ替え

トの配列を図４のように変更すれば，ℓ＝５からℓ＝３に減少させることができる。

2.　プログラム構成

ＢＬＯＯＭプログラムの概略構成を図５に示した。入力となる論理回路図の接続情報は論理シミュレータの入力をもとに，対象とする部分のみを抽出して得られる。ゲート配列の決定は，１ゲート列の場合と２ゲート列の場合に分け（前者

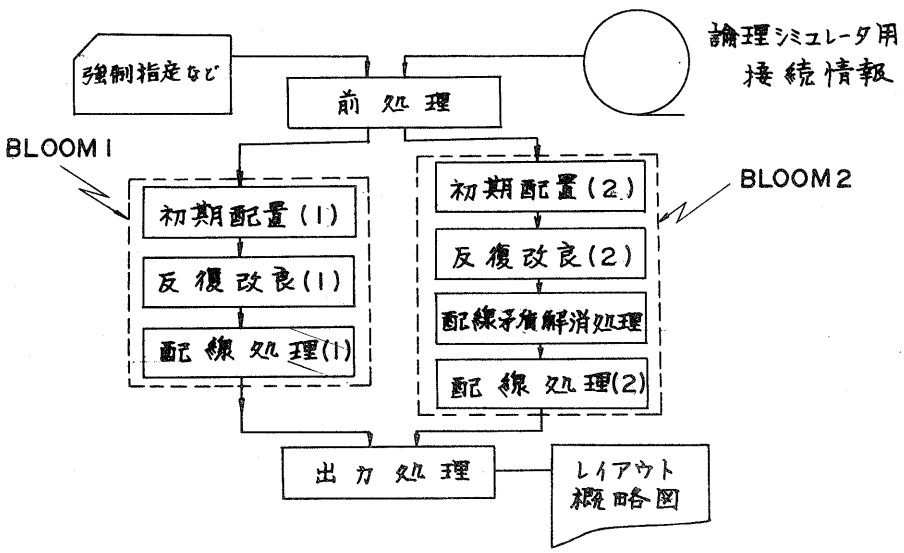

図５　プログラム構成概略

をＢＬＯＯＭ１，後者をＢＬＯＯＭ２と名付けた），どちらも初期配置と反復改良を基本としている。ゲート配列決定の後，配線処理により信号線の位置を決定するが，２ゲート列の場合には配線処理の前に，後で述べるような配線矛盾解消処理を必要とする。出力結果はラインプリンタリストとして得られる。

使用計算機はＡＣＯＳ７７シリーズ・モデル６００又はＮＥＡＣ２２００／５００，言語はＦＯＲＴＲＡＮ，プログラムステートメント数は約１１０００である。

3. サブプログラムの機能と算法

以下，図５に示した各サブプログラムについて機能と算法の概略を述べる。

3.1 前処理

論理回路図の接続情報をもつ論理シミュレータ用入力ファイルから，ゲート列構成でレイアウトしようとする部分を抽出するとともに，ゲートの強制位置指定，上下左右にとり出す配線系列の指定などの入力データを読みとり，各データをチェックの後，以下の処理で用いるデータ構造に格納する。

3.2 ＢＬＯＯＭ１

１ゲート列構成の場合のレイアウトを行う部分で，次に述べる３個のサブプログラムから構成されている。主な制限は，ゲート数≦２００，配線系列数≦400である。算法の詳細については文献3)で述べたので，ここでは要約して述べる。

3.2.1 初期配置（１）…… （ｉ）乱数を利用したランダムな初期配置の発生，又は（ii）クラスタリング法による初期配置の発生，を選択することができる。（ii）の手法は次に述べるクラスタリング処理とゲート配列発生処理とから成る。クラスタリング処理は文献4)の手法をもとに，クラスタリング値を改良したものである。

（Ａ）クラスタリング処理 …… ゲートの集合S_i，S_jと非負の整数uに対してクラスタリング値$T^u(S_i, S_j)$を定義する。これは「不足数u以内でS_i，S_jを結ぶ配線数」のことで，例えば図６において$u=1$（S_iとS_jのすべてのゲート

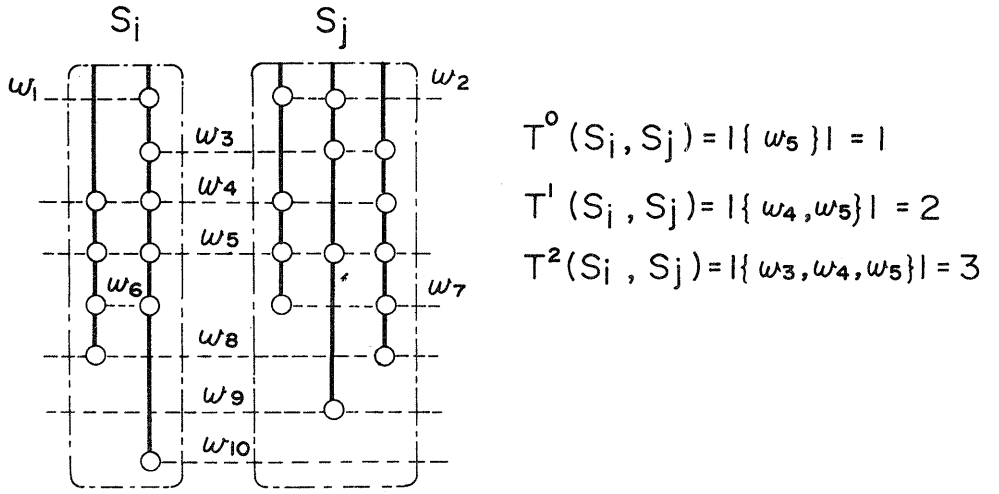

図６ クラスタリング値・$T^u(S_i, S_j)$

に端子をもつか又は1ゲートだけ欠けている）のときの配線数（＝クラスタリング値；$T^1(S_i, S_j)$）は $|\{w_4, w_5\}| = 2$ である（集合Aに対して $|A|$ でAの元の数をあらわす）。より正確には次のように定義する。

$$T^u(S_i, S_j) = \begin{cases} = \left|\{ w \in W ; g(w) \cap S_i \neq \phi,\ g(w) \cap S_j \neq \phi,\ |(g(w) \cap (S_i \cup S_j))| \geq |S_i| + |S_j| - u \}\right| \\ \quad , \ |S_i| + |S_j| > u \text{ のとき} \\ = 0 , \text{ それ以外} \end{cases}$$

ここで，$g(w)$ は配線 $w \in W$ が端子をもつゲートの集合をあらわす。クラスタリング処理の手順は次のように定めた。

C 1. $u \leftarrow 0$，$S \leftarrow B$（ゲートの集合）

C 2. Sの中のすべての S_i，S_j の組に対して $T^u(S_i, S_j)$ を計算し，最大値をとる組を S_k，S_ℓ とする。

C 3. $T^u(S_k, S_\ell) = 0$ ならば，$u \leftarrow u + 1$ としてC 2へ。それ以外なら S_k と S_ℓ をクラスタリングし，Sを更新（Sから S_k，S_ℓ を削除した後，S_k と S_ℓ を新しいクラスタとして登録したものをあらためてSとおく）してC 4へ。

C 4. $|S| = 1$ ならば終了。それ以外ならC 2へ。

（B）ゲート配列発生処理 …… クラスタリング処理によってゲートがクラスタされていく過程は図7のような2進木で表現できる。図7において各節点の番号はクラスタされた順序をあらわしている。ゲート配列はこの2進木の $n-1$ 番目の節点（n は全ゲート数）からはじめて，クラスタされた順序と逆順にクラスタを分解していくことによって得られる。処理手順を以下に示す。

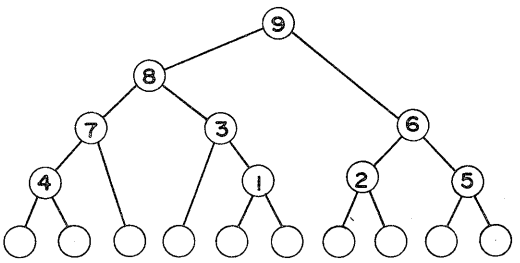

図7　2進木

P 1. $n-1$ の節点に対応するクラスタを2つのクラスタに分解し，左右を適当に決める。

P 2. 分解されてないクラスタがなければ終了。あれば番号最大のものを分解する（Sが S_1，S_2 に分解されたとする）。

P 3. Sの左に置かれているゲート集合を L，Sの右に置かれているゲート集合をRとする（R，Lは空集合のこともある）。配列〔L，S_1，S_2，R〕および〔L，S_2，S_1，R〕のうち，ℓ の下限値の最大が小さくなる方を選び，S_1，S_2 の配列を定め，P 2へ。

ここで，〔L，S_1，S_2，R〕に対する ℓ の下限値の最大値は次のようにして求める。ゲート b が S_1 に含まれるとき：b の端子数＋Lと（S_2 又はR）を結び，b に端子をもたない配線数。ゲート b が S_2 に含まれるとき：b の端子数＋Rと（S_1 又はL）を結び，b に端子をもたない配線数。下限値は S_1 および S_2 に含まれるゲートの数だけ求められ，その最大値がとられる。

3. 2. 2　反復改良（1）…… 次の2つの単位操作により，3. 2. 1で得られたゲート配列の反復改良を行う（図8参照）。

〔単位操作1〕　ある配列とゲート b_i , b_j に対して，b_i を b_j を跳び越して b_j のとなりに移動することによって新しい配列を得る。

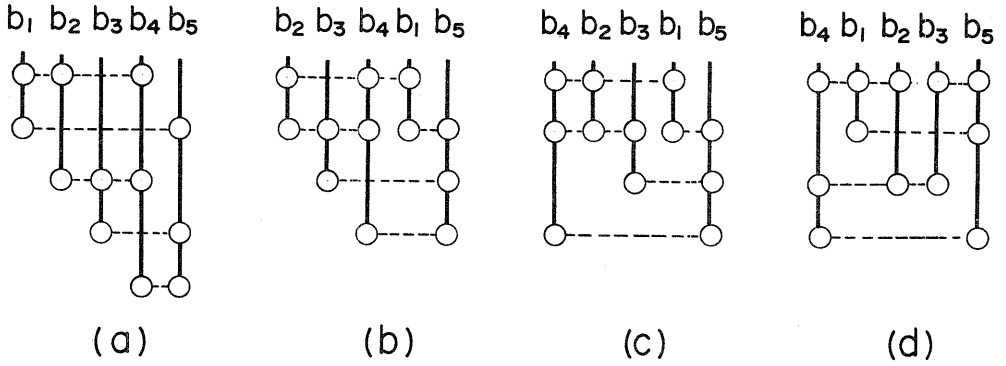

図8　単位操作1および2の説明図。a)　元の配列，b)　b_1 , b_4 に単位操作1をほどこした結果，c) b_1 , b_4 に単位操作2をほどこした結果，d)　b_4 , b_1 に単位操作1をほどこした結果。

〔単位操作2〕　ある配列とゲート b_i , b_j に対して，b_i と b_j の位置を交換することによって新しい配列を得る。

反復改良処理は与えられたゲート配列に対し，単位操作1，2をほどこし，解が改良されれば配列を更新するという手順で行う。ここで「解の改良」とは，（イ）目的関数（max.ℓ）が減少するか又は（ロ）目的関数の値は変らないが，その値をもつゲート数が減少する，のどちらかが成立つことを言う。これら（イ）（ロ）の2つの判定基準を設定することにより，（イ）だけの場合に比べて，より良い極小解が得られる。

3.2.3　配線処理（1）…… 3.2.1，3.2.2により求められたゲート配列のもとで信号配線の位置を定める。基本的には左端端子位置が最も左に位置する配線から順次空区間を上からつめていくようにして定める[7]。下側への引き出し端子をもつ配線は，引き出し線が他のゲート端子と同一列内でオーバラップしないように下方にくるように考慮する。

3.3　BLOOM2

2ゲート列構成の場合のレイアウトを行う部分で，次に述べる4個のサブプログラムから構成されている。主な制限は，ゲート数≦200，配線系列数≦400である。

3.3.1　初期配置（2）…… 全ゲートをゲート数の等しい2つの集合に分割し，各々を上の列，下の列に割当てる。その場合に，上下列の双方に端子をもつような配線を少なくするために，〝クラスタリング法による初期分割の後，ゲートの対交換を行う[5]。対交換の各段階では，改良に寄与する可能性の大きいゲート対から順次選びつつ，改良をもたらす i（≦5）対目が現われた時点で i 対の同時交換を行う。次に，2分割されたゲートの集合それぞれについて3.2.1の手法により配列を決定する。

3.3.2　反復改良（2）…… 3.2.2で述べた手法と同様にしてゲート配列の改良をはかる。ただし，〔単位操作1〕〔単位操作2〕ともに，上のゲート列内，下のゲート列内，上下同一列に位置するゲート対，を対象にして行う。

3. 3. 3　配線矛盾解消処理 …… 3. 3. 1，3. 3. 2で決定されたゲート配置では配線不可能の場合がある。ここでは実現不可能のときその矛盾を解消する。

i) 位置矛盾の解消 …… 同一位置にある上下ゲートの両方に端子をもつ配線系列は配線が不可能である。そこで上下ゲートの一方をとなりあうゲートと交換するなどの手段により矛盾の解消を行う。

ii) 閉路矛盾の解消 …… 同一位置にある上ゲートおよび下ゲートのそれぞれに接続される配線系列n_1，n_2があるとき，n_1はn_2より上の位置に配線されなければならない。一方，別の位置でn_2が上ゲートに，n_1が下ゲートに接続されていると配線が不可能である（図9（a）参照）。ここでは図9（b）又は（c）のように，対象となるゲートの隣りの列での折り曲げ，又は1列分の余分のスペースを挿入することにより矛盾の解消を行っている。

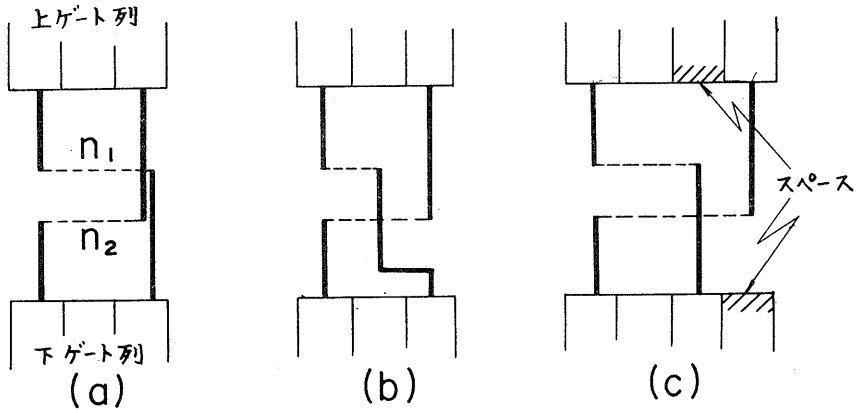

図9　閉路矛盾の解消

3. 3. 4　配線処理（2） …… 3. 2. 3で述べた1ゲート列の場合と異なって3. 3. 3（ii）で述べたような配線間の上下制約が生じるので，文献6)で述べたマッピング処理ステップと同様に，配線の形状による優先度を用いてヒューリスティックに配線位置を定めている。

3. 4　出力処理

レイアウト概略図をプリンタ出力する。BLOOM1の場合の例を図10，BLOOM2の場合の例を図11に示した。§1で述べたようにこの方式のレイアウトはチップの一部分に適用されるので，周辺との関連から修正されることもあり，直接アートワークテープを作ることは行っていない。

4.　ラン結果

2ゲート列の場合（BLOOM2）の結果の評価は現在のところ未だ不充分であるので，1ゲート列の場合（BLOOM1）についてラン結果を表1に示した表1から信号配線に要する行の数は，人手設計に比べ平均して15％改良されており，本プログラムが実用的であると言えよう。初期配置の2つの手法（ランダムとクラスタ）の差は顕著でない（ランダム法では10組の初期解について反復改良を行い最良のものを選んでいる）。なお表1の処理時間はNEAC2200

モデル５００の場合であり，配線処理と出力処理時間は含めていない（含めた場合の例はEx．6で１６分５６秒，Ex．7で２分５０秒である）。

例題番号	ゲート数	配線数	信号配線の行数			処理時間（秒）	
			人手設計	クラスタ＋反復改良	ランダム＋反復改良	クラスタ＋反復改良	ランダム＋反復改良
Ex．1	61	87	36	34	33	733	634
2	80	55	22	19	21	1223	683
3	52	72	27	22	21	493	551
4	53	55	24	17	19	330	325
5	80	81	24	23	26	1028	398
6	45	31	23	23	—	339	—
7	31	38	16	14	—	100	—

表１　ＢＬＯＯＭ１ラン結果の例

謝辞　例題の提供他御指導載いた当社集積回路事業部第２回路技術部遠藤課長，有泉氏，有賀氏，算法に関し御討論戴いた東京大学伊理教授，当社中央研究所大附主任に深く感謝致します。

文献

1) A. Weinberger, "Large scale integration of MOS complex logic: a layout method," IEEE J. Solid-State Circuits, vol. SC-2, no.4, pp 182-190, December 1967

2) R. P. Larsen, "Computer-aided preliminary layout design of customized MOS arrays," IEEE Trans. on Computers, vol. C-20, no.5, pp 512-523, May 1971

3) H. Yoshizawa, H. Kawanishi and K. Kani, "A heuristic procedure for ordering MOS arrays," Proc. 12th D.A. Conf., pp 384-393, June 1975

4) D. M. Schuler and E.G. Ulrich, "Clustering and linear placement," Proc. 9th D.A. Workshop, pp 50-56, June 1972

5) B. W. Kerninghan and S. Lin, "An efficient heuristic procedure for partitioning graphs," B. S. T. J., vol. 49, pp 291-307, Feb., 1970

6) 川西，後藤，小山田，加藤，可児，"ビルディングブロック方式ＬＳＩ配線プログラムの一算法"，電子通信学会回路とシステム理論研究会資料 CT73-19，１９７３年７月

7) A. Hashimoto and J. Stevens, "Wire routing by optimizing channel assignment within large apertures," Proc. 8th DA Workshop, pp 155-169, June 1971

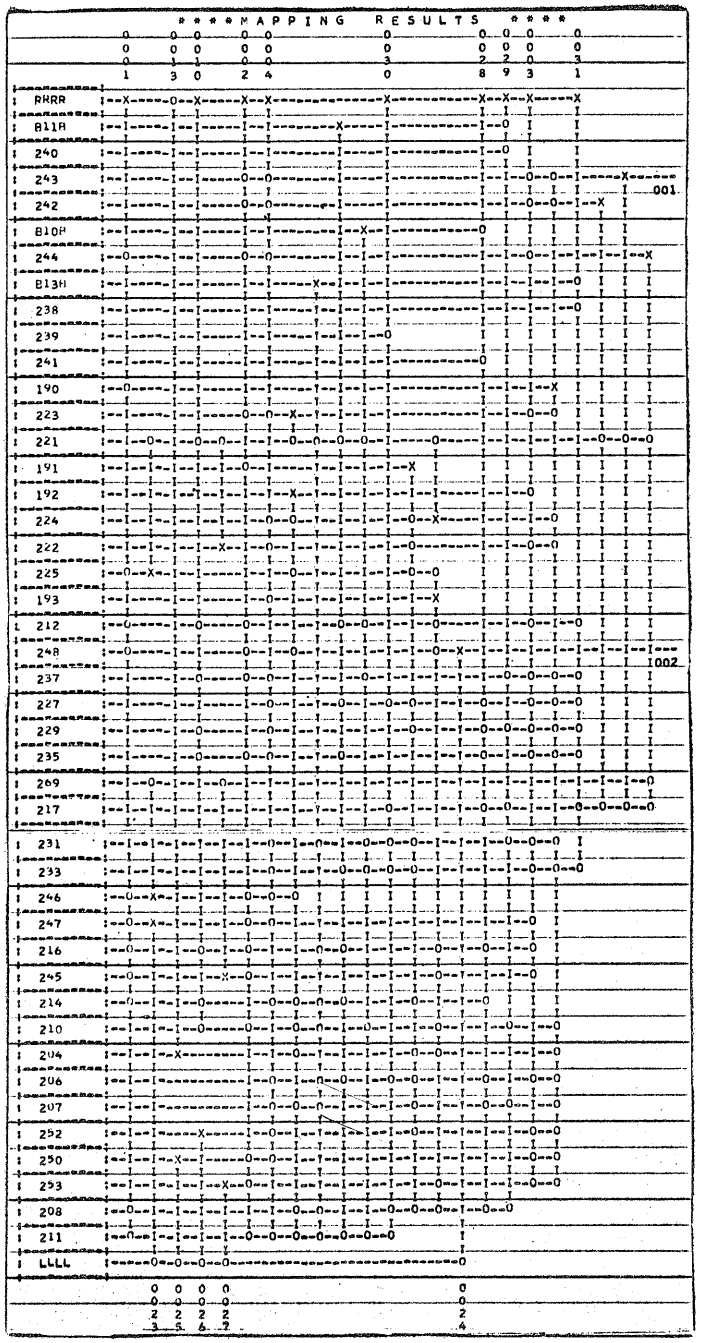

図１０　ＢＬＯＯＭ１の出力例

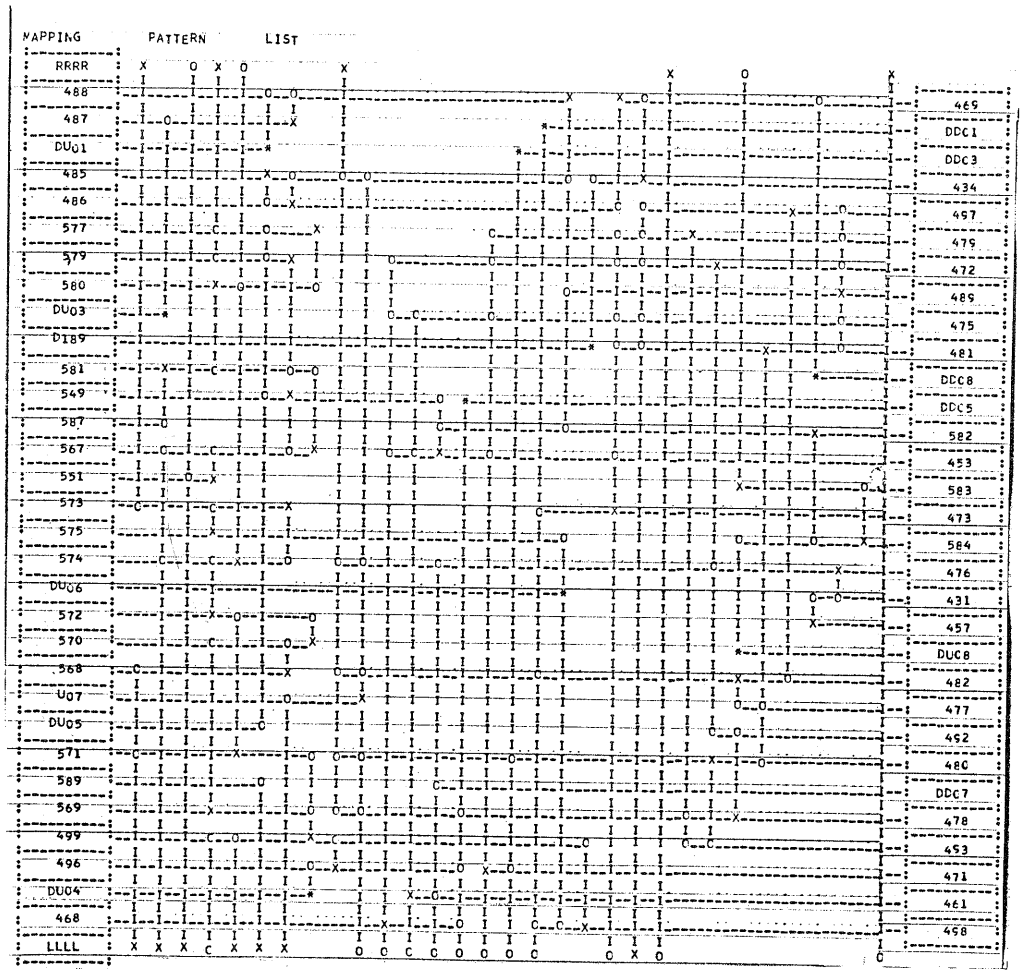

図１１　ＢＬＯＯＭ２の出力例