■1 群（信号・システム）- 8 編（論理回路）

3 章 有限状態機械と順序回路

（執筆者：高木直史）［2010 年 2 月 受領］

■概要■

出力が現時点での入力だけでなく，過去の入力にも依存する論理回路を順序回路という．順序回路は，過去の入力に依存する内部状態をもつ．順序回路は，入力や内部状態がクロック信号に同期して一斉に変化する，同期式順序回路と，事象生起の因果関係を駆動原理とし，クロック信号が存在しない，非同期式順序回路に大別される．

順序回路を設計する場合，まず，動作を有限状態機械として表す．有限状態機械は有限個の状態をもち，入力とそのときの状態に応じて出力を生成し次の状態に遷移する．入力と現状態のすべての組合せに対して出力と次状態が定義されている有限状態機械を完全指定有限状態機械，そうでないものを不完全指定有限状態機械という．

入出力動作が同じである有限状態機械が複数存在する．入出力動作が同じで状態数が最小の有限状態機械を求めることを，有限状態機械の最小化という．完全指定有限状態機械の最小化は，等価な状態を併合することによって行える．不完全指定有限状態機械の最小化は，両立する状態の集合を求め，両立集合の集合による状態遷移に関して閉じた最小の被覆を求めることによって行える．

同期式順序回路は，状態を記憶するフリップフロップ群と，出力関数及び次状態関数を計算する組合せ回路部からなる．有限状態機械を同期式順序回路で実現するには，状態をフリップフロップ群で記憶するために 2 値系列で符号化する必要がある．これを状態割当てという．状態割当てにおいては，組合せ回路部を小さくするために，次状態関数の状態変数への依存ができるだけ小さくなるようにすることが望ましい．状態集合の分割による方法のほか，論理最小化に基づく方法が提案されている．

同期式順序回路に対し，初期状態にかかわらず出力系列から最終状態を決めることができる入力系列をホーミング系列，初期状態と出力にかかわらず最終状態を決めることができる入力系列を同期系列，出力系列から初期状態を決めることができる入力系列を識別系列と呼ぶ．これらの系列を求める方法が提案されている．

非同期式回路設計では，論理ゲートや配線に起こり得る遅延変動の振る舞いを表したモデル，すなわち遅延モデル（遅延仮定）が重要な役割を果たす．小規模な非同期式順序回路合成方法として，ハフマン式回路合成方法やマラー式回路合成方法が知られている．また，ディジタル回路の基本動作となるレジスタ間データ転送に関して，束データ方式や 2 線 2 相式などの非同期式回路実現方法が提案されている．

【本章の構成】

本章では，3-1 節で，有限状態機械とはどのようなものかを述べる．3-2 節では，完全指定及び不完全指定有限状態機械の最小化の手法を述べる．3-3 節では，同期式順序回路による有限状態機械の実現について説明する．3-4 節では，同期式順序回路の解析について述べる．3-5 節では，非同期式順序回路とその設計について述べる．

■1 群 - 8 編 - 3 章

3-1 有限状態機械

（執筆者：樋口博之）[2008 年 10 月 受領]

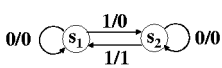
有限状態機械は，有限個の状態をもち，入力とそのときの状態に応じて出力を生成し次の状態に遷移する，一種の計算モデルである．有限状態機械は次のように定義される[1)]．

定義 1　有限状態機械（finite state machine; FSM）は五つ組 $M = (I, O, S, \delta, \lambda)$ である．ここで，I, O, S はそれぞれ入力記号，出力記号，状態を要素とする空でない有限集合である．$\delta : I \times S \to S$ は**状態遷移関数（state transition function）**，λ は**出力関数（output function）**である．$\lambda : I \times S \to O$ である有限状態機械を**ミーリー型有限状態機械（finite state machine of Mealy type）**と呼び，$\lambda : S \to O$ である有限状態機械を**ムーア型有限状態機械（finite state machine of Moore type）**と呼ぶ．

有限状態機械の動作を理解しやすくするために，その表による表現である**状態遷移表（state transition table）**あるいはグラフによる表現である**状態遷移グラフ（state transition diagram）**を用いる．状態遷移表は，各入力に対応した $|I|$ 個の列と各状態に対応した $|S|$ 個の行をもつ．現状態と入力記号の各組合せに対して，対応する行と列の欄にその次状態と出力を指定する．ここで，集合 A に対して $|A|$ は集合の要素数を表す．状態遷移グラフでは，その各節点が有限状態機械の状態に対応する．一つの節点から，各入力記号による状態遷移に対応して $|I|$ 本の有向枝が出る．有向枝にはその遷移を起こす入力記号と生成される出力記号をラベル付けする．例として，次の有限状態機械 M1-1 を考える：$I = \{0, 1\}$; $O = \{0, 1\}$; $S = \{s_1, s_2\}$; $\delta(0, s_1) = s_1$, $\delta(1, s_1) = s_2$, $\delta(0, s_2) = s_2$, $\delta(1, s_2) = s_1$; $\lambda(0, s_1) = 0$, $\lambda(1, s_1) = 0$, $\lambda(0, s_2) = 0$, $\lambda(1, s_2) = 1$．機械 M1-1 の状態遷移表と状態遷移グラフを図 3･1 の (a) と (b) にそれぞれ示す．これらから機械 M1-1 は，状態 s_1 から開始する場合には，入力系列の中の偶数番目の 1 のたびに 1 を出力する機械であるといえる．

現状態	次状態，出力	
	入力0	入力1
s_1	s_1, 0	s_2, 0
s_2	s_2, 0	s_1, 1

(a) 状態遷移表



(b) 状態遷移グラフ

図 3･1　機械 M1-1

次状態あるいは出力が指定されてない現状態と入力の組が存在してよい有限状態機械を**不完全指定有限状態機械（incompletely specified finite state machine）**と呼ぶ．これに対し，定義 3･1 で与えられる有限状態機械を**完全指定有限状態機械（completely specified finite state machine）**と呼ぶ．不完全指定は，ある状態において決して印加されることのない入力が存在する場合や，ある入力と状態の組に対して出力を外部で観測する必要のない場合などに用いられる．例えば，図 3･1 の機械 M1-1 において，状態 s_2 では入力 0 が決して入力されないことが分かっている場合，表 3･1 のような不完全指定機械で

表すことができる．表 3･1 において，「−」はその状態あるいは出力が指定されていないことを示す．未指定のことを**ドントケア（don't care）**とも呼ぶ．

表 3･1　機械 M1-2

現状態	次状態, 出力	
	入力 0	**入力** 1
s_1	$s_1, 0$	$s_2, 0$
s_2	−, -	$s_1, 1$

不完全指定有限状態機械が未指定の次状態をもつ場合は，**トラップ状態（trap state）**を用いた変換により出力にのみ不完全指定をもつ有限状態機械に置き換え可能である[1)]．その変換では，指定されていない次状態の欄すべてにおいてトラップ状態を指定し，トラップ状態からの遷移を，出力が未指定の終端状態となるよう追加する．トラップ状態 s_T を用いて，表 3･1 の機械 M1-2 を未指定の次状態をもたない機械 M1-2' に変換した結果を表 3･2 に示す．

表 3･2　出力のみの不完全指定に変換した機械 M1-2'

現状態	次状態, 出力	
	入力 0	**入力** 1
s_1	$s_1, 0$	$s_2, 0$
s_2	$s_T, -$	$s_1, 1$
s_T	$s_T, -$	$s_T, -$

ミーリー型機械はムーア型に変換することができる．この変換は，出力が異なる状態遷移は異なる次状態に至るよう，次状態を遷移の出力ごとに分割することで行える*．例えば，図 3･1 の機械 M1-1 をムーア型に変換すると図 3･2 となる．節点のラベルは「ミーリー型機械での状態／その状態に至る出力」を示す．図 3･1 に示すとおり，機械 M1-1 は s_1 に至る遷移に関し，0 を出力する遷移と 1 を出力する遷移があるため，それぞれに対応して s_1 が「$s_1/0$」と「$s_1/1$」に分割される．$s_1/0$ からの遷移と $s_1/1$ からの遷移は同一である．

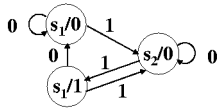


図 3･2　機械 M1-1 **のムーア型への変換**

■参考文献

1)　Z. Kohavi, "Switching and Finite Automata Theory (Second Edition)," McGraw-Hill, 1978.

* このとき，ムーア型機械での入力系列印加前の出力は無視するものとする．

■1 群 - 8 編 - 3 章

3-2 有限状態機械の最小化

（執筆者：樋口博之）[2008 年 10 月 受領]

有限状態機械の状態遷移表をつくると冗長な状態が含まれることが多い．このような機械は入出力動作を変えずに冗長状態を含まない最小の機械に変換することができる．以下では，完全指定有限状態機械と不完全指定有限状態機械それぞれに対する最小化を説明する．

3-2-1 完全指定有限状態機械の最小化

完全指定有限状態機械 M の状態 s_i と s_j について，入力系列印加前の状態が s_i か s_j かにより，異なる出力系列を生成するような有限長の入力系列が少なくとも一つ存在するとき，s_i と s_j は**識別可能（distinguishable）**であるという．s_i と s_j を識別する入力系列を状態対 (s_i, s_j) の**識別系列（distinguishing sequence）**と呼ぶ．ある二つの状態が識別可能ならば，それらは一つの状態にまとめられない．状態 s_i と s_j が識別可能でないならば，任意の入力系列に対して，入力系列印加前の状態が s_i か s_j かにかかわらず同一の出力系列が生成される．状態 s_i と s_j が識別可能でないとき，かつそのときのみ，s_i と s_j は**等価（equivalent）**であるという．

状態 s_i と s_j に対して長さ k の識別系列が存在するとき，s_i と s_j は k **識別可能（k-distinguishable）**であるという．k 識別可能でない状態の組を k **等価（k-equivalent）**であるという．二つの状態が等価であるとは，任意の自然数 k に対して k 等価であることである．例えば，表 3･3(a) の機械 M2-1 では s_1 と s_2 は 1 等価であり，s_1 と s_3 は 1 識別可能である．

表 3･3　機械 M2-1 と最小機械

現状態	次状態, 出力	
	入力0	入力1
s_1	s_5, 0	s_6, 0
s_2	s_4, 0	s_3, 0
s_3	s_1, 1	s_2, 0
s_4	s_2, 0	s_6, 0
s_5	s_1, 0	s_3, 0
s_6	s_3, 1	s_1, 0

(a) **機械**M2-1

現状態	次状態, 出力	
	入力0	入力1
$\{s_1,s_4\}\rightarrow A$	B, 0	D, 0
$\{s_2,s_5\}\rightarrow B$	A, 0	C, 0
$\{s_3\}\rightarrow C$	A, 1	B, 0
$\{s_6\}\rightarrow D$	C, 1	A, 0

(b) **最小機械**M2-1*

定理 1　機械 M の状態 s_i と s_j が k 等価（$k > 1$）である必要十分条件は次の二条件が成り立つことである．

1. 任意の入力に対して，s_i と s_j の出力が同一である．すなわち，s_i と s_j が 1 等価である．

2. 任意の入力 x に対し，s_i と s_j の x に対する次状態 $\delta(x, s_i), \delta(x, s_j)$ が $k-1$ 等価である．　□

状態の k 等価関係は同値関係であり，有限状態機械の状態集合は k 等価関係により同値類に分割できる．ある集合 A の分割とは，A の部分集合の集合で，各要素が互いに素であり，かつすべての要素の和が A になるものである．分割の要素をブロックと呼ぶ．状態集合の分

割 P_i と P_j に対し，P_i に含まれる任意のブロックが P_j のどれかのブロックに含まれるとき，P_i は P_j より小さいか等しいといい，$P_i \leq P_j$ と書く．完全指定有限状態機械 M の等価状態をすべて求める手続き[1)]を表 3･3(a) の機械 M2-1 を例として具体的に示す．

1. 機械 M のすべての状態を同一のブロックに入れた初期分割 P_0 をつくる．機械 M2-1 では $P_0 = \{\{s_1, s_2, s_3, s_4, s_5, s_6\}\}$ である．

2. 分割 P_0 内のブロックを分割して 1 等価な状態集合のブロックからなる分割 $P_1(\leq P_0)$ をつくる．これは，任意の入力に対して同じ出力をもつ状態のみを同一のブロックに入れることにより行える．機械 M2-1 では $P_1 = \{\{s_1, s_2, s_4, s_5\}, \{s_3, s_6\}\}$ である．

3. 分割 P_1 内のブロックを分割して 2 等価な状態のブロックからなる分割 $P_2(\leq P_1)$ をつくる．具体的には，ある状態対が P_1 の同じブロックに含まれ，かつその状態対の任意の入力に対する次状態対も P_1 のある一つのブロックに含まれるとき，かつそのときのみ，この状態対を P_2 の同じブロックに含める．機械 M2-1 では $P_2 = \{\{s_1, s_2, s_4, s_5\}, \{s_3\}, \{s_6\}\}$ となる．

4. ステップ 3 と同様に, 分割 P_k 内のブロックを分割して $k+1$ 等価な状態のブロックからなる分割 $P_{k+1}(\leq P_k)$ を k を 1 ずつ増やしながらつくっていく．ある k に対して $P_{k+1} = P_k$ になれば, 処理を終了し P_k を返す．機械 M2-1 では, $P_3 = \{\{s_1, s_4\}, \{s_2, s_5\}, \{s_3\}, \{s_6\}\}$, $P_4 = P_3$ となるため，P_3 を返す． □

上述の手続きの結果として得られる分割を**等価分割（equivalence partition）**と呼ぶ．等価分割では，同じブロックに含まれる状態対はすべて等価であり，異なるブロックに含まれる状態対はすべて識別可能である．またこの分割は唯一に定まる．次の定理から，分割 P_k はたかだか $k = n$ まで計算すればよいことが分かる．

定理 2　状態数 n の機械 M の二つの状態 s_i と s_j が識別可能ならば，その識別系列の長さは $n-1$ 以下である[1)]． □

定義 1　二つの機械 M_1 と M_2 について，M_1 の任意の状態に対して，それと等価な M_2 の状態が存在し，逆に，M_2 の任意の状態に対して，それと等価な M_1 の状態が存在するとき，かつそのときのみ，M_1 と M_2 は**等価（equivalent）**であるという．機械 M と等価で状態数最小の機械を求めることを M の**最小化（minimization）**と呼ぶ．

等価分割の各ブロックに含まれる状態を一つの状態にまとめた機械は状態数最小の機械を与える．例えば，機械 M2-1 の等価分割 P_3 のブロック $\{s_1, s_4\}, \{s_2, s_5\}, \{s_3\}, \{s_6\}$ をそれぞれ A, B, C, D で表すと，表 3･3(b) の最小機械 M2-1*を得る．

以上の等価分割手順は，機械の状態数 n に対し $O(n^2)$ の計算量が必要であるが，分割の扱いを工夫することで $O(n \log n)$ に削減できる[2)]．

3-2-2 不完全指定有限状態機械の最小化

不完全指定有限状態機械では，次状態が未指定の場合，それ以降の動作すべてが未指定であるから，入力系列も次状態が未指定になるまでの入力系列のみを考慮する．状態 s_i の不完

全指定有限状態機械 M にある入力系列を印加し，最後の遷移を除いて未指定の次状態がないとき，その入力系列は M の状態 s_i に**印加可能（applicable）**であるという．

定義 2　機械 M の二つの状態 s_i と s_j に対し，s_i と s_j の両方に印加可能な任意の入力系列について，入力系列印加前の状態が s_i か s_j かにかかわらず，両方の出力が指定されているときはいつでも同じ出力が生成されるとき，かつそのときのみ，s_i と s_j は**両立的（compatible）**であるという．両立的な状態対を**両立対（compatible pair）**と呼ぶ．

定理 3　機械 M の二つの状態 s_i と s_j が両立的である必要十分条件は，任意の入力に対して，s_i と s_j の出力が両方とも指定されているときには出力が同一であり，かつ，s_i と s_j の次状態が両方とも指定されているときには次状態が両立的であることである．　□

表 3･4 の機械 M2-2 では，(s_1, s_2) と (s_2, s_3) はいずれも両立対だが，(s_1, s_3) は両立対ではない．すなわち，両立関係は等価関係と異なり推移律が成り立たず，同値関係でない．

表 3･4　機械 M2-2

現状態	**次状態, 出力**	
	入力 0	**入力** 1
s_1	$s_2, 0$	$s_3, 0$
s_2	$s_1, 0$	$s_2, -$
s_3	$-, 0$	$s_1, 1$

定義 3　状態の集合 $\{s_{i_1}, s_{i_2}, \ldots, s_{i_k}\}$ のすべての要素の対が両立的であるとき，その状態集合は両立的であるという．両立的な状態集合を**両立集合（compatible set）**と呼ぶ．

両立集合は，その要素すべてを併合して一つの状態にできることを意味する．例えば，表 3･5 の機械 M2-3 では，状態集合 $\{s_3, s_4, s_5\}$ は両立集合である．

表 3･5　機械 M2-3

現状態	**次状態, 出力**	
	入力 0	**入力** 1
s_1	$s_5, -$	$s_2, 0$
s_2	$s_3, 0$	$s_2, -$
s_3	$s_2, -$	$s_4, 1$
s_4	$s_2, -$	$s_3, 1$
s_5	$s_1, 1$	$s_4, -$

定義 4　機械 M_1 の状態 s_i と機械 M_2 の状態 s_j に対して，s_j に印加可能な任意の入力系列が s_i にも印加可能であり，M_1 と M_2 がそれぞれ最初に s_i と s_j にいるとき，s_j に印加可能な任意の入力系列を印加することにより，M_2 の出力が指定されているときは必ず M_1 も同一の出力を生成するとき，かつそのときのみ，s_i は s_j を**被覆（cover）**するという．機械 M_2 の任意の状態 s_j に対して，機械 M_1 のある状態 s_i が存在し，s_i が s_j を被覆するとき，かつそのときのみ，M_1 は M_2 を被覆するという．

両立集合に含まれる状態すべてを一つに併合した状態は元の各状態を被覆する．

不完全指定有限状態機械の最小化が完全指定の場合と違う点を三点述べる．第一の点は，**状態分割 (state splitting)** によって状態数を更に削減できることである．例えば，表 3･4 の機械 M2-2 では行「s_2」列「入力 1」のドントケアを 0 にしても 1 にしても等価状態は存在しない．しかし，表 3･6(a) に示すように，s_2 を s_2' と s_2'' に分割した機械 M2-2' を考えると，機械 M2-2 の状態数を削減できる．例えば，ドントケアに表 3･6(b) のように値を割り当てれば (s_1, s_2') と (s_2'', s_3) がそれぞれ等価になり，二状態にすることができる．$(s_1, s_2'), (s_2'', s_3)$ はそれぞれ，元の機械 M2-2 の両立対 $(s_1, s_2), (s_2, s_3)$ に対応する．不完全指定有限状態機械の最小化問題は，全状態の集合を両立集合により重なりを許して被覆する問題となる．

表 3･6　機械 M2-2 の状態分割

現状態	次状態, 出力	
	入力0	入力1
s_1	s_2', 0	s_3 , 0
s_2'	s_1, 0	s_2", -
s_2"	s_1, 0	s_2', -
s_3	- , 0	s_1 , 1

(a) **状態分割**M2-2'

現状態	次状態, 出力	
	入力0	入力1
s_1	s_2', 0	s_3 , 0
s_2'	s_1, 0	s_2", 0
s_2"	s_1, 0	s_2', 1
s_3	s_2', 0	s_1, 1

(b) **完全指定化**M2-2"

第二の点は，最小解が唯一とは限らないことである．例えば，表 3･5 の機械 M2-3 では，解となる両立集合の集合として $\{\{s_1, s_2\}, \{s_3, s_4, s_5\}\}$ と $\{\{s_1, s_5\}, \{s_2, s_3, s_4\}\}$ の二つが存在するため，二つの相異なる最小機械が存在する．また，解となる両立集合の選択が唯一でも，次状態としてどの両立集合を用いるかにより最小機械が複数できる場合もある．

第三の点は，元の状態がすべて被覆されることだけでなく，両立集合の次状態の集合が，選択した両立集合に含まれることも必要であることである．表 3･5 の機械 M2-3 を再び考えると，両立集合の集合 $\{\{s_1\}, \{s_2\}, \{s_3, s_4, s_5\}\}$ も元の状態をすべて被覆しているが，$\{s_3, s_4, s_5\}$ の入力 0 に対する次状態の集合 $\{s_1, s_2\}$ を被覆する要素が存在しないため解ではない．

以上を考慮して，不完全指定有限状態機械の最小化は次のような手順で行える[3, 4]．

1. 併合表をつくり，すべての両立対を求める．
2. すべての両立対をもとに極大な両立集合をすべて求める．
3. 最小化で考慮すべき両立集合である「プライムコンパティブル」をすべて求める．
4. 最小機械を構成する「プライムコンパティブル」の集合を選択する問題を「バイネイト被覆問題」に帰着して解く．
5. 4 で求めた解に対応する最小機械を求める．　□

以下では，「プライムコンパティブル」「バイネイト被覆問題」などの用語の定義も含め，それぞれのステップについて表 3･7 の機械 M2-4 を例に詳しく説明する．

ステップ 1 では，併合表をつくる．**併合表 (merger table)** は，行と列それぞれに状態を並べ，表中の各欄に，対応する状態対の両立性の情報を記入したものである．二つの状態 s_i と s_j の入力 x に対する次状態をそれぞれ s_p と s_q とすると，$s_p \neq s_q$ ならば，(s_i, s_j) が両立的になるためには (s_p, s_q) が両立的でなければならない．このとき，(s_p, s_q) は (s_i, s_j)

表 3･7　機械 M2-4

現状態	次状態, 出力			
	入力 x_1	**入力** x_2	**入力** x_3	**入力** x_4
s_1	$s_3, 0$	$s_4, 1$	$s_5, -$	$s_5, -$
s_2	$-, -$	$s_3, 1$	$s_4, 1$	$-, -$
s_3	$s_4, 0$	$-, -$	$-, -$	$s_4, -$
s_4	$s_5, 0$	$s_5, 0$	$-, -$	$s_6, -$
s_5	$-, -$	$s_6, -$	$s_1, -$	$-, -$
s_6	$s_1, -$	$-, 0$	$s_2, -$	$-, -$

に**含意（imply）**されるといい，(s_p, s_q) を**含意対（implied pair）**と呼ぶ．例えば，機械 M2-4 の状態対 (s_3, s_4) は (s_1, s_2) の含意対である．二つの状態が非両立的なら × 印を記入し，両立的なら，無条件に両立的であるときはチェック印を記入し，含意対が存在するときはその含意対を記入する．各欄に最初に記入した直後の機械 M2-4 の併合表を図 3･3(a) に示す．

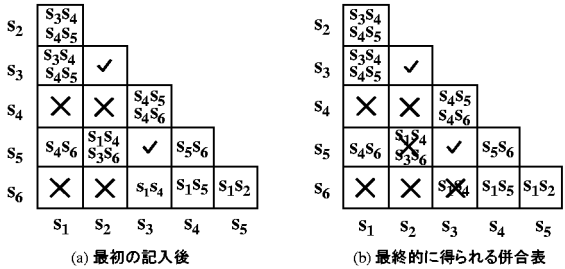


図 3･3　機械 M2-4 **の併合表**

状態対が両立的になるにはその含意対も両立でなければならないので，次に，含意対が非両立的でないかどうかを調べ，併合表を更新する．更新がなくなるまで繰り返した後の併合表を図 3･3(b) に示す．この併合表から，次の 9 個の両立対が得られる．

$$(s_1, s_2), (s_1, s_3), (s_1, s_5), (s_2, s_3), (s_3, s_4), (s_3, s_5), (s_4, s_5), (s_4, s_6), (s_5, s_6)$$

ステップ 2 では，併合表から次の (a)〜(c) の手順ですべての極大両立集合を生成する．

(a) 併合表の一番右から，両立対を含む列まで左に進み，その列のすべての両立対を列挙する．

(b) 一つ以上の両立対を含む次の列まで左に進む．その列に対応する状態 s_i が，既に生成されたある両立集合のすべての要素と両立的なら，s_i をその両立集合に追加する．s_i がすでに生成した両立集合の一部の要素とのみ両立的なら，その両立している要素すべてと s_i からなる新しい両立集合をつくる．最後に，その列の両立対で，すでに生成されたどの両立集合にも含まれないものをすべて列挙する．

(c) (b) をすべての列を処理するまで繰り返し，極大両立集合の集合を得る．　□

例えば，図 3･3(b) の併合表から次の極大両立集合の集合を得る．

$$\{\{s_1, s_2, s_3\}, \{s_1, s_3, s_5\}, \{s_3, s_4, s_5\}, \{s_4, s_5, s_6\}\}$$

ステップ 3 では，プライムコンパティブルをすべて生成する．極大両立集合の部分集合をすべて列挙すればすべての両立集合が求まるが，両立集合間の支配関係から，考慮すべき両立集合を絞り込める．ある両立集合に関して，ある入力 x に対する次状態の集合を，その両立集合の入力 x に対する**含意集合（implied set）**と呼ぶ．

定義 5　不完全指定有限状態機械 M の両立集合の集合 C に関し，その集合に含まれる任意の両立集合に対して，そのすべての含意集合も C のどれかの要素に状態の集合として含まれるとき，その両立集合の集合は**閉じている（closed）**という．M のすべての状態を含む両立集合の閉じた集合を**閉じた被覆（closed covering）**と呼ぶ．

機械 M の最小化は最小の閉じた被覆を求める問題に帰着できる．例えば，機械 M2-4 に対して，集合 $\{\{s_1, s_5\}, \{s_4, s_6\}\}$ は閉じた集合である．集合 $\{\{s_1, s_5\}, \{s_4, s_6\}, \{s_2, s_3\}\}$ は閉じた被覆である．ある両立集合を解に含めると，その含意集合も解の要素の両立集合のいずれかに含まれる必要がある，という含意集合に関する条件を**閉包条件（closure condition）**という．

定義 6　両立集合 C の**クラスセット（class set）** $CS(C)$ は，C の含意集合 D のうち，次の条件を満たすものすべてからなる集合である．

1. D は二つ以上の要素をもつ, かつ
2. $D \not\subseteq C$, かつ
3. $D \not\subset D'$ if $D' \in CS(C)$.

直感的に言えば，C のクラスセットは，C の含意集合のうち閉包条件に寄与する含意集合の集合である．例えば，機械 M2-4 において，両立集合 $\{s_3, s_4, s_5\}$ の含意集合は $\{s_4, s_5\}, \{s_5, s_6\}, \{s_1\}, \{s_4, s_6\}$ であり, クラスセット $CS(\{s_3, s_4, s_5\})$ は $\{\{s_4, s_6\}, \{s_5, s_6\}\}$ である．

定義 7　両立集合 C, C' が次の二条件を満たすとき，C は C' を**支配する（dominate）**という．

1. $C \supset C'$, かつ
2. $CS(C) \subseteq CS(C')$.

直感的に言えば, C' が C より小さく, かつ, 閉包条件も等しいか C' の方が厳しいときのみ, C が C' を支配する．C' を含む解は C' を C で置き換えても必ず解となる．例えば, 機械 M2-4 では，両立集合 $\{s_1, s_2, s_3\}$ は $\{s_1, s_2\}$ を支配する．なぜなら，$\{s_1, s_2, s_3\} \supset \{s_1, s_2\}$，かつ $CS(\{s_1, s_2, s_3\}) = CS(\{s_1, s_2\}) = \{\{s_3, s_4\}, \{s_4, s_5\}\}$ であるからである．

定義 8　ほかのどの両立集合にも支配されない両立集合を**プライムコンパティブル（prime compatible）**という．

プライムコンパティブルを生成する手順を示す．

(a) 極大両立集合を大きさの降順に並べる．最大の極大両立集合のサイズを n_{max} とする．
(b) 極大両立集合のクラスセットを生成する．
(c) サイズ n_{max} の極大両立集合をプライムコンパティブルの集合 P に加える．
(d) サイズ k を $n_{max}-1$ から 1 まで 1 ずつ減らしながら (d.1) と (d.2) を繰り返す．
　(d.1) サイズ $k+1$ 以上の各極大両立集合の部分集合でサイズが k のもののうち，P のどの要素にも支配されないもの C を P に加え，C のクラスセットも生成する．
　(d.2) サイズ k のすべての極大両立集合を P に加える． □

例として，機械 M2-4 のプライムコンパティブルを生成する．まず，サイズ $n_{max}=3$ の極大両立集合とそのクラスセットは表 3･8 の番号 1 から 4 の行で示される．(d) でサイズ 2 から 1 まで順に生成したプライムコンパティブルを表 3･8 の番号 5〜14 の行に示す．

表 3･8　機械 M2-4 のプライムコンパティブルとクラスセット

番号	サイズ k	プライムコンパティブル	クラスセット
1	3	$\{s_1,s_2,s_3\}$	$\{\{s_3,s_4\},\{s_4,s_5\}\}$
2	3	$\{s_1,s_3,s_5\}$	$\{\{s_3,s_4\},\{s_4,s_5\},\{s_4,s_6\}\}$
3	3	$\{s_3,s_4,s_5\}$	$\{\{s_4,s_6\},\{s_5,s_6\}\}$
4	3	$\{s_4,s_5,s_6\}$	$\{\{s_1,s_2\},\{s_1,s_5\}\}$
5	2	$\{s_2,s_3\}$	$\emptyset$
6	2	$\{s_1,s_5\}$	$\{\{s_4,s_6\}\}$
7	2	$\{s_3,s_5\}$	$\emptyset$
8	2	$\{s_3,s_4\}$	$\{\{s_4,s_5\},\{s_4,s_6\}\}$
9	2	$\{s_4,s_5\}$	$\{\{s_5,s_6\}\}$
10	2	$\{s_4,s_6\}$	$\{\{s_1,s_5\}\}$
11	2	$\{s_5,s_6\}$	$\{\{s_1,s_2\}\}$
12	1	$\{s_1\}$	$\emptyset$
13	1	$\{s_4\}$	$\emptyset$
14	1	$\{s_6\}$	$\emptyset$

最小化のステップ 4 では，プライムコンパティブルのみからなる最小の閉じた被覆を求める．i 番目のプライムコンパティブルに論理変数 C_i を割り当て，C_i が 1 のときそのプライムコンパティブルを解の要素として選択するものとし，解が満たすべき条件を論理式で書き出す．この論理式は和積形であり，被覆条件に関する和項と閉包条件に関する和項からなる．

各状態が被覆されるとは，その状態を含むプライムコンパティブルが少なくとも一つ解に含まれることである．例えば，機械 M2-4 の状態 s_1 はプライムコンパティブル C_1,C_2,C_6,C_{12} に含まれるので，状態 s_1 の被覆条件を表す和項は $(C_1+C_2+C_6+C_{12})$ となる．

プライムコンパティブル C_i が閉包条件を満たすとは，C_i を選択したとき，その含意集合各々を包含するプライムコンパティブルも少なくとも一つは選択するということである．例えば，機械 M2-4 のプライムコンパティブル $C_1=\{s_1,s_2,s_3\}$ を考えると，C_1 を選択するときには，そのクラスセット $\{\{s_3,s_4\},\{s_4,s_5\}\}$ の各要素を包含するプライムコンパティ

ブルも選択しなければならない．例えば，含意集合 $\{s_3, s_4\}$ を包含するプライムコンパティブルは C_3 と C_8 であるから，その閉包条件は $C_1 \Rightarrow C_3 + C_8$ となる．これを論理式で表すと和項 $(\overline{C_1} + C_3 + C_8)$ となる．

被覆条件と閉包条件に関するすべての和項の積が，閉じた被覆が充足すべき論理式となる．その論理式を真にするプライムコンパティブルの選択が閉じた被覆を表し，最小の閉じた被覆が最小解となる．機械 M2-4 の例では，閉じた被覆が充足すべき論理式は次のようになる．6 番目までの和項が被覆条件に対応する．

$$(C_1+C_2+C_6+C_{12})\cdot(C_1+C_5)\cdot(C_1+C_2+C_3+C_5+C_7+C_8)\cdot(C_3+C_4+C_8+C_9+C_{10}+C_{13})\cdot(C_2+C_3+C_4+C_6+C_7+C_9+C_{11})\cdot(C_4+C_{10}+C_{11}+C_{14})\cdot(\overline{C_1}+C_3+C_8)\cdot(\overline{C_1}+C_3+C_4+C_9)\cdot(\overline{C_2}+C_3+C_8)\cdot(\overline{C_2}+C_3+C_4+C_9)\cdot(\overline{C_2}+C_4+C_{10})\cdot(\overline{C_3}+C_4+C_{10})\cdot(\overline{C_3}+C_4+C_{11})\cdot(\overline{C_4}+C_1)\cdot(\overline{C_4}+C_2+C_6)\cdot(\overline{C_6}+C_4+C_{10})\cdot(\overline{C_8}+C_3+C_4+C_9)\cdot(\overline{C_8}+C_4+C_{10})\cdot(\overline{C_9}+C_4+C_{11})\cdot(\overline{C_{10}}+C_2+C_6)\cdot(\overline{C_{11}}+C_1).$$

閉じた被覆が充足すべき論理式は各和項にたかだか一つ負のリテラルを含む．一般に，負のリテラルを含む和積形論理式を充足する最小コストの値割当てを求める問題を**バイネイト被覆問題**と呼ぶ．バイネイト被覆問題は分枝限定法により解くことができる．機械 M2-4 の場合，最小解は $C_5 = C_6 = C_{10} = 1$ であり，閉じた被覆は $\{\{s_1, s_5\}, \{s_2, s_3\}, \{s_4, s_6\}\}$ となる．

最小化のステップ 5 では，ステップ 4 で求めた閉じた最小被覆の各状態集合を 1 状態にまとめ，最小機械をつくる．機械 M2-4 では，$\{s_1, s_5\}, \{s_2, s_3\}, \{s_4, s_6\}$ をそれぞれ状態 A, B, C に置き換え，表 3･9 の最小機械を得る．

表 3･9　機械 M2-4 の最小機械

現状態	次状態, 出力			
	入力 x_1	入力 x_2	入力 x_3	入力 x_4
$\{s_1, s_5\} \to A$	$B, 0$	$C, 1$	$A, -$	$A, -$
$\{s_2, s_3\} \to B$	$C, 0$	$B, 1$	$C, 1$	$C, -$
$\{s_4, s_6\} \to C$	$A, 0$	$A, 0$	$B, -$	$C, -$

上述した最小化の厳密解法以外に，近似解法についても考案されている[5)]．二分決定グラフを用いた最小化の効率的方法については，文献 6) に詳しく述べられている．

■参考文献

1) Z. Kohavi, “Switching and Finite Automata Theory, (Second Edition),” McGraw-Hill, 1978.
2) J. Hopcroft, “An $n \log n$ Algorithm for Minimizing States in a Finite Automaton,” Theory of Machines and Computations, Z. Kohavi and A. Paz, editors, Academic Press, pp.189-196, 1971.
3) M.C. Paull and S.H. Unger, “Minimizing the Number of States in Incompletely Specified Sequential Switching Functions,” IRE Trans. Electronic Computers, vol.8, pp.356-367, Sep. 1959.

4) A. Grasselli and F. Luccio, “A Method for Minimizing the Number of Internal States in Incompletely Specified Sequential Networks,” IEEE Trans. Electronic Computers, vol.14, pp.350-359, Jun. 1965.
5) J.-K. Rho and G.D. Hachtel and F. Somenzi and R. Jacoby, “Exact and Heuristic Algorithms for the Minimization of Incompletely Specified State Machine,” IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, vol.13, no.2, pp.167-177, Feb. 1994.
6) T. Kam and T. Villa and R. Brayton and A. Sangiovanni-Vincentelli, “Synthesis of Finite State Machines: Functional Optimization,” Kluwer Academic Publishers, 1997.

■1 群 - 8 編 - 3 章

3-3 同期式順序回路による有限状態機械の実現

（執筆者：樋口博之）[2008 年 10 月 受領]

本節では，有限状態機械の実現として用いられる同期式順序回路について述べる．また，その実現のために必要な有限状態機械の状態割当てについて説明する．状態割当ては実現する回路の構造と複雑さに大きな影響を与える．

3-3-1 同期式順序回路

有限状態機械は状態をもち，その状態と入力により次状態と出力を計算するものであるから，状態を回路の内部に記憶できれば，記憶した値と入力から次状態と出力を計算する回路は組合せ回路により実現できる．組合せ回路と記憶回路からなる回路を**順序回路（sequential circuit）**と呼ぶ．ここでは，状態遷移をクロックパルスに同期させて行う**同期式順序回路（synchronous sequential circuit）**を考え，有限状態機械から同期式順序回路を得る一般的手順[1)]を説明する．順序回路には同期式順序回路のほかに，クロックパルスを用いない**非同期式順序回路（asynchronous sequential circuit）**もある．

同期式順序回路の一般的構成を図 3･4 に示す．$x_1, x_2, \ldots, x_p$ は入力変数，$z_1, z_2, \ldots, z_q$ は出力変数，$y_1, y_2, \ldots, y_n$ は現状態変数，$Y_1, Y_2, \ldots, Y_n$ は次状態変数である．各々の入力変数，出力変数，現状態変数，及び次状態変数は 0 か 1 のいずれかの値をとる．$\vec{x} = (x_1, x_2, \ldots, x_p)$ を入力ベクトル（あるいは単に入力），$\vec{z} = (z_1, z_2, \ldots, z_q)$ を出力ベクトル（あるいは単に出力），$\vec{y} = (y_1, y_2, \ldots, y_n)$ を現状態ベクトル（あるいは単に現状態），$\vec{Y} = (Y_1, Y_2, \ldots, Y_n)$ を次状態ベクトル（あるいは単に次状態）と呼ぶ．図 3･4 では明示していないが，記憶回路には同期のためのクロック信号も入力する．クロックパルスの時間間隔を時間の単位とすると，時刻 t に，入力 $\vec{x}$ と現状態 $\vec{y}$ の値が組合せ回路部に与えられ，それにより出力 $\vec{z}$ と次状態 $\vec{Y}$ が計算される．次のクロックパルス，すなわち時刻 $t+1$ に，$\vec{Y}$ の値が記憶回路に記憶されると同時にそれらの値は $\vec{y}$ から現状態として現れる．組合せ回路部での計算時間はクロックパルスの時間間隔より短くなければならない．

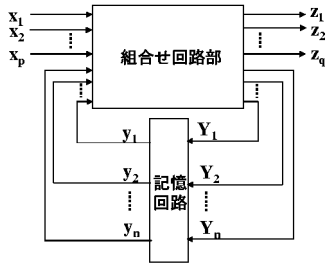


図 3･4　同期式順序回路

有限状態機械を順序回路で実現するには，記号で表された各状態を相異なる二値ベクトルに対応させる必要がある．この処理を**状態割当て（state assignment）**という．入力や出力も記号で与えられた場合は，これらにも同様に符号割当てが必要である．

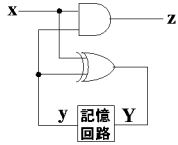
例として，図 3･5(a) の機械 M3-1 を考える．状態 s_1, s_2 にそれぞれ 0, 1 を割り当てた後の状態遷移表を図 3･5(b) に示す．状態遷移関数は $Y = x \oplus y$，出力関数は $z = xy$ である．得られる順序回路を図 3･5(c) に示す．

現状態	次状態，出力	
	入力0	入力1
s_1	s_1, 0	s_2, 0
s_2	s_2, 0	s_1, 1

(a) **機械**M3-1

y	Y,z	
	x=0	x=1
$s_1 \rightarrow 0$	0, 0	1, 0
$s_2 \rightarrow 1$	1, 0	0, 1

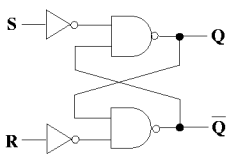
(b) **状態割当後**



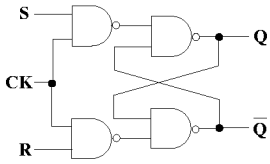
(c) **順序回路**

図 3･5　機械 M3-1 **の順序回路**

順序回路の記憶回路を構成する記憶素子として広く用いられているのは**フリップフロップ（flip-flop; FF）**である．FF は **SR ラッチ（SR latch）**と呼ばれる回路が基本要素になる．SR ラッチは $Q = 1$ と $Q = 0$ の二つの状態をもつ．出力 $\overline{Q}$ は出力 Q の否定である．入力 S と R はそれぞれセットとリセットの意味である．$(S, R) = (0, 0)$ のとき状態は保持される．$S = 1$ にすると状態は $Q = 1$ にセットされる．$R = 1$ にすると状態は $Q = 0$ にリセットされる．$(S, R) = (1, 1)$ は禁止入力である．SR ラッチの NAND ゲートを用いた回路図を図 3･6(a) に示す．**クロック付き SR ラッチ（clocked SR latch）**はクロック信号 CK を用いた同期式の SR ラッチである．その回路図を図 3･6(b) に示す．以下ではクロック付きのもののみを考える．



(a) SR**ラッチ**



(b) **クロック付き**SR**ラッチ**

図 3･6　SR **ラッチ**

ラッチの問題点は，クロックが 1 の間に入力が変化すると，状態も変化し得ることである．一つのクロックパルスの間に状態遷移が 2 回以上起きないようパルス幅の上限を設ける必要があり，回路設計が難しい．この点を解決した記憶素子が FF*である．FF には**マスタースレーブ型 FF（master-slave FF）**と**エッジトリガ型 FF（edge-triggered FF）**がある．

マススレーブ型 FF は二つの SR ラッチを直列につなげ，そのクロック入力が否定の関係になるようにしたものである．その回路を図 3･7 に示す．最初のラッチを**マスターラッチ（master latch）**，二つ目のラッチを**スレーブラッチ（slave latch）**と呼ぶ．クロック入力が 1 の間にマスターラッチが入力を取り込み，状態を変えるが，スレーブラッチは前

* 広義にはラッチも含めて FF と呼ぶこともある．

状態を保持したままになり，次にクロック入力が 0 になると逆に，マスターラッチの状態は固定され，スレーブラッチの状態が変化し，マスターラッチの状態と同じになる．したがって，マスタースレーブ型 FF を使うと入力が複数回変化しても FF 全体の状態はただ一度しか変化しない．

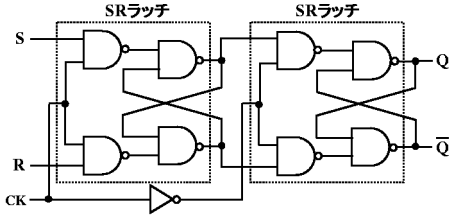


図 3·7　マスタースレーブ型 FF

エッジトリガ型 FF は，入力読込みと出力変化の両方をクロックの立ち上がりまたは立ち下がりのいずれかのみで行う FF で，それぞれ**ポジティブエッジトリガ型 FF（positive edge-triggered FF）**及び**ネガティブエッジトリガ型 FF（negative edge-triggered FF）**と呼ぶ．

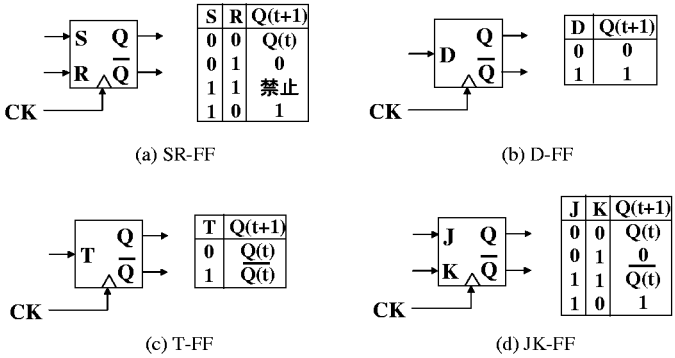
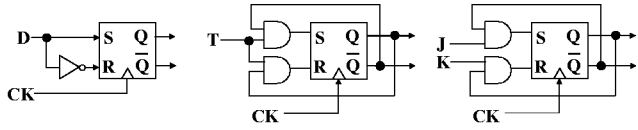


図 3·8　**FF **の記号と入出力動作表

FF には **SR-FF** のほかに，**D-FF**，**T-FF** 及び **JK-FF** がある．各 FF の記号と入出力動作表を図 3·8 に示す．D-FF は遅延素子と同じ動作をし，入力 D を次時刻の状態とし，その値を出力する．順序回路の次状態の値をそのまま入力できるため，順序回路の記憶回路として最もよく用いられる．クロック入力 CK の周期を時間単位とした時刻 t に対して，$Q(t+1) = D(t)$ となる．次に，T-FF は入力 T=1 で状態が変わる機能をもつ．最後に，JK-FF は SR-FF と T-FF を組み合わせた機能をもつ．入力 J, K は基本的に SR-FF の S, R にそれぞれ対応する．SR-FF では禁止入力であった $(J, K) = (1, 1)$ は禁止入力でなく，その入力に対しては T-FF として動作する．D-FF, T-FF, JK-FF の SR-FF による実現方法をそれぞれ図 3·9(a),(b),(c) に示す．



(a) D-FF　(b) T-FF　(c) JK-FF

図 3·9　D-FF,T-FF,JK-FF

表 3·10　FF **の励起要求**

状態遷移	SR-FF		D-FF	T-FF	JK-FF	
$Q(t) \to Q(t+1)$	S	R	D	T	J	K
$0 \to 0$	0	−	0	0	0	−
$0 \to 1$	1	0	1	1	1	−
$1 \to 0$	0	1	0	1	−	1
$1 \to 1$	−	0	1	0	−	0

各状態遷移を引き起こすための FF の入力，すなわち，**励起要求（excitation requirement）**は FF の種類ごとに異なる．表 3·10 に 4 種類の FF に対する励起要求を示す．用いる FF の種類を決めると，その FF の励起要求と状態割当て後の状態遷移表から，次状態の記憶のために組合せ回路部で計算すべき関数，すなわち**励起関数（excitation function）**が決まる．例えば，図 3·5(b) の状態遷移表を D-FF で実現する場合には，状態遷移関数がそのまま励起関数となるため，得られる順序回路は図 3·5(c) の記憶回路を D-FF に置き換えたものとなる．図 3·5(b) を JK-FF で実現する場合，例えば，現状態 y=0 で入力 $x = 1$ により次状態 $Y = 1$ になる遷移は Q が $0 \to 1$ となる遷移であるから，JK-FF の入力は $J = 1$，$K = -$（ドントケア）となる．同様に考えて，JK-FF の励起表は表 3·11 のようになり，励起関数は，例えば，$J = K = x$ である．

表 3·11　図 3·5(b) **に対する励起表**

y	$x = 0$ JK	$x = 1$ JK
0	0−	1−
1	−0	−1

有限状態機械から同期式順序回路を合成する一般的手順は次のようになる．

1. 与えられた有限状態機械の状態数を最小化し，冗長な状態をなくす．
2. 状態割当てを行い，状態を二進符号で置き換えた状態遷移表をつくる．
3. 用いる FF の種類を選び，その FF の励起関数と出力関数を求める．
4. 励起関数と出力関数を実現する組合せ回路をつくる．
5. 4 でつくった組合せ回路と FF の入出力を接続し，順序回路をつくる．　□

これ以降，FF は D-FF を仮定する．したがって，励起関数は状態遷移関数と同一である．

3-3-2 状態割当て

順序回路の実現に必須の処理である状態割当てについて，二つの枠組みによる手法を説明する．第一は状態集合の分割に基づく方法[1, 2]，第二は論理最小化に基づく方法[3, 4]である．前者は更に閉じた分割に基づく方法と分割対に基づく方法に分かれる．

(1) 状態集合の分割に基づく方法

(a) 閉じた分割に基づく方法

状態割当てでは，各状態変数 y_i は，状態集合 S を $y_i = 0$ の集合と $y_i = 1$ の集合に二分した分割 τ_i をつくっていると考えることができる．

分割 τ_1 と τ_2 に対して，和 $\tau_1 + \tau_2$ は，ある状態の系列 $s_{k_1}, s_{k_2}, \ldots, s_{k_a}$ が存在し，その任意の隣り合う要素 $s_{k_i}, s_{k_{i+1}} (1 \leq i \leq a-1)$ が τ_1 か τ_2 の少なくとも一方において同一のブロックに含まれるとき，かつそのときのみ，s_{k_1} と s_{k_a} を同一のブロックに含めるようにした分割である．積 $\tau_1 \cdot \tau_2$ は，τ_1 と τ_2 の両方において，それぞれ同一のブロックに含まれる状態対のみ同一のブロックに含めるようにした分割である．例えば，$\tau_1 = \{\{s_1, s_6\}, \{s_2, s_5, s_7\}, \{s_3\}, \{s_4, s_8\}\}$，$\tau_2 = \{\{s_1, s_4\}, \{s_2, s_3, s_5\}, \{s_6\}, \{s_7\}, \{s_8\}\}$ とすると，$\tau_1 + \tau_2 = \{\{s_1, s_4, s_6, s_8\}, \{s_2, s_3, s_5, s_7\}\}$, $\tau_1 \cdot \tau_2 = \{\{s_1\}, \{s_2, s_5\}, \{s_3\}, \{s_4\}, \{s_6\}, \{s_7\}, \{s_8\}\}$ となる．

ある状態割当てにおいて，状態変数 $y_1, y_2, \ldots, y_k$ がそれぞれ分割 $\tau_1, \tau_2, \ldots, \tau_k$ をつくるとき，その状態割当てがすべての状態に別々の符号を割り当てる必要十分条件は $\tau_1 \cdot \tau_2 \cdots \tau_k = \pi(0)$ である．ここで $\pi(0)$ は 0 分割，すなわち，各ブロックがただ一つの状態からなる分割を表す．

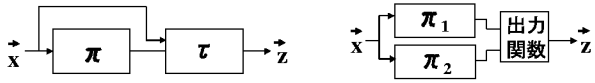
状態割当て問題は，積が 0 分割となる分割の集合を求める問題と考えることができる．各分割は必ずしも二つのブロックからなる必要はない．分割 τ のブロック数を $|\tau|$ とすると，これらのブロックを区別するために，$\lceil \log_2 |\tau| \rceil$ 個の状態変数を割り当て，τ のそれぞれのブロックに，これらの変数の相異なる値の組合せを割り当てる．簡単な回路構造につながる分解をもたらし，かつ，積が 0 分割になる分割の集合を求めるため，閉じた分割という概念を導入する．

定義 1　有限状態機械 M の状態集合に関する分割 π について，π の同じブロックに含まれるどの二つの状態 s_i, s_j に対しても，任意の入力 $x_k \in I$ に関し，次状態 $\delta(x_k, s_i)$ と $\delta(x_k, s_j)$ が π の同一のブロックに含まれるとき，分割 π は**閉じている（closed）**という．

閉じた分割は現在のブロックと入力から次のブロックが一意に決められることを意味する．

定理 1　機械 M の状態集合について，$\pi \cdot \tau = \pi(0)$ となる閉じた分割 π 及びある分割 τ が存在すれば，M は直列につながる二つの部分機械に分解できる．　□

定理 1 の分解を**直列分解（serial decomposition）**と呼ぶ．直列分解の構造を図 3･10(a) に示す．直列接続の一つ目の部分機械は π のブロックを区別するための状態変数に対応し，$\lceil \log_2 |\pi| \rceil$ 個の記憶素子をもつ．これらの状態変数の状態遷移関数は残りの状態変数に独立である．直列接続の二つ目の部分機械は τ のブロックを区別するための状態変数に対応し，$\lceil \log_2 |\tau| \rceil$ 個の記憶素子をもつ．



(a) **直列分解**　　(b) **並列分解**

図 3·10　直列分解と並列分解

例えば，表 3·12(a) の機械 M3-2 は閉じた分割 $\pi_1 = \{\{s_1, s_4\}, \{s_2, s_3\}\}$ をもち，直列分解できる．$\tau = \{\{s_1, s_3\}, \{s_2, s_4\}\}$ とし，τ, π_1 をそれぞれ y_1, y_2 に対応させると，表 3·12(b) の状態割当てが得られる．表 3·12(b) に示す状態遷移関数から Y_2 は x と y_2 のみに依存することが分かる．

表 3·12　機械 M3-2 **と状態割当て**

現状態	次状態，出力	
	入力0	入力1
s_1	s_4, 0	s_2, 0
s_2	s_3, 0	s_4, 1
s_3	s_3, 0	s_1, 0
s_4	s_1, 0	s_3, 1

(a) **機械**M3-2

y_1y_2	Y_1Y_2, z	
	x=0	x=1
$s_1 \rightarrow 00$	10, 0	11, 0
$s_2 \rightarrow 11$	01, 0	10, 1
$s_3 \rightarrow 01$	01, 0	00, 0
$s_4 \rightarrow 10$	00, 0	01, 1

$Y_1 = \bar{y}_1\bar{y}_2 + xy_1y_2$
$Y_2 = \bar{x}y_2 + x\bar{y}_2$
$z = xy_1$

(b) **状態割当と次状態・出力関数**

定理 2　機械 M の状態集合について，$\pi_1 \cdot \pi_2 = \pi(0)$ となる二つの閉じた分割 π_1, π_2 が存在すれば，M は互いに独立に並列に動作する二つの部分機械に分割できる．　□

定理 2 の分解を**並列分解（parallel decomposition）**と呼ぶ．並列分解の構造を図 3·10(b) に示す．例えば，表 3·13(a) の機械 M3-3 は二つの閉じた分割 $\pi_1 = \{\{s_1, s_2\}, \{s_3, s_4\}\}$, $\pi_2 = \{\{s_1, s_3\}, \{s_2, s_4\}\}$ をもつので並列分解できる．現状態変数 y_1 を π_1 に，現状態変数 y_2 を π_2 にそれぞれ対応させると表 3·13(b) の状態割当てを得る．表 3·13(b) に示す状態遷移関数から，Y_1 は x と y_1 にのみ依存し，Y_2 は x と y_2 にのみ依存することが分かる．

表 3·13　機械 M3-3 **と状態割当て**

現状態	次状態，出力	
	入力0	入力1
s_1	s_4, 1	s_1, 0
s_2	s_3, 0	s_2, 1
s_3	s_2, 0	s_1, 0
s_4	s_1, 0	s_2, 1

(a) **機械**M3-3

y_1y_2	Y_1Y_2, z	
	x=0	x=1
$s_1 \rightarrow 00$	11, 1	00, 0
$s_2 \rightarrow 01$	10, 0	01, 1
$s_3 \rightarrow 10$	01, 0	00, 0
$s_4 \rightarrow 11$	00, 0	01, 1

$Y_1 = \bar{x}\bar{y}_1$
$Y_2 = \bar{x}\bar{y}_2 + xy_2$
$z = xy_1\bar{y}_2 + xy_2$

(b) **状態割当と次状態・出力関数**

定理 3　機械 M の状態集合に関する二つの閉じた分割 π_1, π_2 の和 $\pi_1 + \pi_2$ と積 $\pi_1 \cdot \pi_2$ は状態遷移に関して閉じた分割である．　□

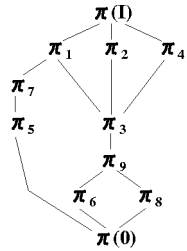
機械 M の状態集合に関する，閉じた分割すべての集合は，和と積の二項演算について閉じており，任意の閉じた分割の組 π_1, π_2 に対して $\pi_1 + \pi_2$ が上限，$\pi_1 \cdot \pi_2$ が下限に対応する．したがって，閉じた分割の集合と和と積の二項演算は束をなす．この束を π **束（π-lattice)** という．π 束は分割全体からなる集合の束の部分束である．π 束は次の二つのステップで生成できる．

1. 各状態対 (s_i, s_j) に対して，(s_i, s_j) を 1 ブロックに含む最小の閉じた分割 $\pi_{s_i s_j}$ を求める．
2. 1 で求めた $\pi_{s_i s_j}$ のすべての可能な和を求める（$i, j = 1, 2, \ldots, n; i \neq j$）． □

ステップ 1 で生成する $\pi_{s_i s_j}$ を基本分割と呼ぶ．$\pi_{s_i s_j}$ を求めるには，まず最初に，$\pi(0)$ から s_i と s_j を同一のブロックに併合する．この分割を閉じた分割にするには，任意の入力 x に対する s_i と s_j の次状態 $\delta(x, s_i)$ と $\delta(x, s_j)$ も併合する必要がある．この併合手順を推移的に行うことにより閉じた分割 $\pi_{s_i s_j}$ が求まる．

現状態	次状態，出力	
	入力0	入力1
s_1	s_2, 0	s_3, 0
s_2	s_1, 1	s_7, 1
s_3	s_7, 0	s_5, 0
s_4	s_7, 1	s_5, 0
s_5	s_8, 1	s_4, 0
s_6	s_4, 1	s_2, 0
s_7	s_4, 1	s_2, 1
s_8	s_5, 1	s_6, 0

(a) **機械** M3-4



(b) π **束**

図 3·11　機械 M3-4 **と** π **束**

例として，表 3·11(a) の機械 M3-4 を考える．まず基本分割を求める．例えば，状態 s_1 と s_2 の初期併合 (s_1, s_2) から始めて，その次状態対を推移的にすべて求めていくと，$(s_1, s_2), (s_3, s_7), (s_4, s_7), (s_2, s_5), (s_1, s_8), (s_3, s_6)$ となる．これらの状態対について，例えば，$(s_1, s_2), (s_1, s_8)$ から推移的に併合して $\{s_1, s_2, s_8\}$ を得るなどしていくと，$\pi_{s_1 s_2} = \{\{s_1, s_2, s_5, s_8\}, \{s_3, s_4, s_6, s_7\}\}$ を得る．

同様にして，基本分割をすべて求めると次のようになる．

$$\pi_1 \equiv \{\{s_1, s_2, s_5, s_8\}, \{s_3, s_4, s_6, s_7\}\} \quad (= \pi_{s_1 s_2} = \pi_{s_5 s_8})$$
$$\pi_2 \equiv \{\{s_1, s_3, s_4, s_5\}, \{s_2, s_6, s_7, s_8\}\}$$
$$(= \pi_{s_1 s_3} = \pi_{s_1 s_4} = \pi_{s_2 s_6} = \pi_{s_2 s_7} = \pi_{s_3 s_5} = \pi_{s_4 s_5} = \pi_{s_6 s_8} = \pi_{s_7 s_8})$$
$$\pi_3 \equiv \{\{s_1, s_5\}, \{s_2, s_8\}, \{s_3, s_4\}, \{s_6, s_7\}\} \quad (= \pi_{s_1 s_5} = \pi_{s_2 s_8})$$
$$\pi_4 \equiv \{\{s_1, s_5, s_6, s_7\}, \{s_2, s_3, s_4, s_8\}\}$$
$$(= \pi_{s_1 s_6} = \pi_{s_1 s_7} = \pi_{s_2 s_3} = \pi_{s_2 s_4} = \pi_{s_3 s_8} = \pi_{s_4 s_8} = \pi_{s_5 s_6} = \pi_{s_5 s_7})$$
$$\pi_5 \equiv \{\{s_1, s_8\}, \{s_2, s_5\}, \{s_3, s_6\}, \{s_4, s_7\}\}$$
$$(= \pi_{s_1 s_8} = \pi_{s_2 s_5} = \pi_{s_3 s_6} = \pi_{s_4 s_7})$$

$$\pi_6 \equiv \{\{s_1\},\{s_2\},\{s_3,s_4\},\{s_5\},\{s_6\},\{s_7\},\{s_8\}\} \quad (= \pi_{s_3 s_4})$$
$$\pi_7 \equiv \{\{s_1,s_8\},\{s_2,s_5\},\{s_3,s_4,s_6,s_7\}\} \quad (= \pi_{s_3 s_7} = \pi_{s_4 s_6})$$
$$\pi_8 \equiv \{\{s_1\},\{s_2\},\{s_3\},\{s_4\},\{s_5\},\{s_6,s_7\},\{s_8\}\} \quad (= \pi_{s_6 s_7})$$

第二のステップで，基本分割の組の和により第二レベルの分割を生成し，更にその第二レベルの分割の組の和により第三レベルの分割を生成する．この操作を新たな閉じた分割が生成されなくなるまで繰り返す．機械 M3-4 に対しては，次の一つの閉じた分割が追加される．

$$\pi_9 \equiv \pi_6 + \pi_8 = \{\{s_1\},\{s_2\},\{s_3,s_4\},\{s_5\},\{s_6,s_7\},\{s_8\}\}.$$

したがって，機械 M3-4 の π 束は図 3･11(b) のようになる．図 3･11(b) の π 束から，例えば，$\pi_2 \cdot \pi_5 = \pi(0)$ という関係が成り立つことが分かる．これは π_2 と π_5 を使って並列分解が可能であることを示す．また，$\pi_1 > \pi_5$ なので，$\pi_5 = \pi_1 \cdot \tau$ を満たす分割 τ が存在することも分かる．例えば，$\tau = \{\{s_1,s_3,s_6,s_8\},\{s_2,s_4,s_5,s_7\}\}$ とすればよい．これは，π_5 に対応する部分機械が直列分解できることを表す．

(b) 分割対に基づく方法

有限状態機械は必ずしも閉じた分割をもつとは限らない．そのような場合でも，閉じた分割を一般化した分割対を考えることにより，状態変数の依存関係を削減できることがある．

定義 2　有限状態機械 M の状態集合に関する**分割対（partition pair）** (τ,τ') は，二つの状態 s_i と s_j が τ の同じブロックに含まれるなら，任意の入力 $x \in I$ に対して次状態 $\delta(x,s_i)$ と $\delta(x,s_j)$ が τ' の同じブロックに含まれるような，分割の順序付き対である．

分割対 (τ,τ') において，τ のブロックの遷移先は τ' のブロックに含まれる．したがって，τ の各ブロックを識別するために状態変数の集合を割り当て，τ' の各ブロックを識別するために別の状態変数の集合を割り当てると，τ' の状態遷移関数は入力と τ に割り当てた状態変数の集合のみに依存する．すなわち，分割対に基づいて状態割当てを行うと，状態変数の依存関係を削減できる．$\tau' = \tau$ の場合は閉じた分割を表す．

例えば，表 3･14(a) の機械 M3-5 は閉じた分割をもたないが，二つの分割対 (τ_1,τ_2) と (τ_2,τ_1) をもつ．ここで，$\tau_1 = \{\{s_1,s_4\},\{s_2,s_3\}\}, \tau_2 = \{\{s_1,s_3\},\{s_2,s_4\}\}$ である．τ_1,τ_2 にそれぞれ状態変数 y_1,y_2 を割り当て，入力変数を x とすると，τ_1 の状態遷移関数は x と y_2 にのみ依存し，τ_2 の状態遷移関数は x と y_1 にのみ依存する．

表 3･14　機械 M3-5 と状態割当て

現状態	次状態，出力	
	入力0	入力1
s_1	s_1, 0	s_3, 0
s_2	s_3, 0	s_4, 1
s_3	s_1, 0	s_2, 0
s_4	s_3, 1	s_1, 1

(a) 機械M3-5

y_1y_2	Y_1Y_2, z	
	x=0	x=1
$s_1 \rightarrow 00$	00, 0	10, 0
$s_2 \rightarrow 11$	10, 0	01, 1
$s_3 \rightarrow 10$	00, 0	11, 0
$s_4 \rightarrow 01$	10, 1	00, 1

$Y_1=\bar{x}y_2+x\bar{y}_2$
$Y_2=xy_1$
$z=\bar{y}_1y_2+xy_2$

(b) 状態割当と次状態･出力関数

分割対を求める方法を考える．分割対 (τ_1, τ_1') と (τ_2, τ_2') に対し，$\tau_1 \geq \tau_2$ かつ $\tau_1' \geq \tau_2'$ のとき，かつそのときのみ，$(\tau_1, \tau_1') \geq_P (\tau_2, \tau_2')$ と定義する．関係 $\geq_P$ は半順序関係である．(τ_1, τ_1') と (τ_2, τ_2') が機械 M の状態に関する分割対ならば，$(\tau_1 \cdot \tau_2, \tau_1' \cdot \tau_2')$ と $(\tau_1 + \tau_2, \tau_1' + \tau_2')$ もまた M の分割対である．$(\tau_1 \cdot \tau_2, \tau_1' \cdot \tau_2')$ と $(\tau_1 + \tau_2, \tau_1' + \tau_2')$ は (τ_1, τ_1') と (τ_2, τ_2') の下限及び上限である．したがって，すべての分割対の集合は半順序関係 $\geq_P$ のもとで束をなす．

分割対の定義から，(τ, τ') が分割対であるとき，$\tau_a \leq \tau$ となる任意の分割 τ_a について，(τ_a, τ') も分割対であり，$\tau_b' \geq \tau'$ となる任意の分割 τ_b' について，(τ, τ_b') も分割対であることは明らかである．この性質から，すべての分割対を求める問題を，分割対全体の集合より十分小さい，次で定義される Mm ペアという分割対を生成する問題に帰着できる．

定義 3　機械 M の状態集合の分割 τ に対して，(τ, τ_i') が分割対であるようなすべての分割 τ_i' の積 $\Pi\tau_i'$ を分割 $m(\tau)$ と定義する．また，分割 τ' に対して，(τ_i, τ') が分割対であるようなすべての分割 τ_i の和 $\sum \tau_i$ を分割 $M(\tau')$ と定義する．分割対 (τ, τ') が，$\tau = M(\tau')$ かつ $\tau' = m(\tau)$ のとき，かつそのときのみ，(τ, τ') を **Mm ペア（Mm pair）**という．

$M(\tau')$ を M 分割，$m(\tau)$ を m 分割と呼ぶ．すべての Mm ペアの生成は次の手順で行える．

1. すべての状態対 (s_i, s_j) に対して分割 $m(\tau_{s_i s_j})$ を求める．ここで，$\tau_{s_i s_j}$ は，ブロック $\{s_i, s_j\}$ を含み，ほかの各状態を別々のブロックにした分割を表す．
2. 1 で求めた分割のすべての可能な和を求める．これによりすべての m 分割を生成できる．
3. あらゆる m 分割 τ' に対応する M 分割 $M(\tau')$ は，$m(\tau_{s_i s_j}) \leq \tau'$ となるすべての $\tau_{s_i s_j}$ の和 $\sum \tau_{s_i s_j}$ により得られる．　□

例として，表 3･14(a) の機械 M3-5 の Mm ペアを計算する．まず，すべての状態対 (s_i, s_j) に対して $m(\tau_{s_i s_j})$ を計算する．

$$
\begin{aligned}
m(\tau_{s_1 s_2}) &= \{\{s_1, s_3, s_4\}, \{s_2\}\}\ (=\tau_3 \text{ とする}) \\
m(\tau_{s_1 s_3}) &= \{\{s_1\}, \{s_2, s_3\}, \{s_4\}\}(=\tau_4 \text{ とする}) \\
m(\tau_{s_1 s_4}) &= \{\{s_1, s_3\}, \{s_2\}, \{s_4\}\}\ (=\tau_5 \text{ とする}) \\
m(\tau_{s_2 s_3}) &= \{\{s_1, s_3\}, \{s_2, s_4\}\} = \tau_2 \\
m(\tau_{s_2 s_4}) &= \{\{s_1, s_4\}, \{s_2\}, \{s_3\}\}\ (=\tau_6 \text{ とする}) \\
m(\tau_{s_3 s_4}) &= \{\{s_1, s_2, s_3\}, \{s_4\}\}\ (=\tau_7 \text{ とする})
\end{aligned}
$$

次に，$m(\tau_{s_i s_j})$ の任意の和を求め，すべての分割に対する m 分割を生成する．

$$\tau_4 + \tau_6 = \{\{s_1, s_4\}, \{s_2, s_3\}\} = \tau_1$$

更に，生成したすべての m 分割に対する M 分割を求める．$M(\tau_i')$ は，$m(\tau_{s_a s_b}) \leq \tau_i'$ となるすべての $\tau_{s_a s_b}$ の和 $\sum \tau_{s_a s_b}$ であるから，求める M 分割は次のようになる．

$$M(\tau_1) = \tau_{s_1 s_3} + \tau_{s_2 s_4} = \tau_2$$

$$
\begin{aligned}
&M(\tau_2) = \tau_{s_1 s_4} + \tau_{s_2 s_3} = \tau_1\\
&M(\tau_3) = \tau_{s_1 s_2} + \tau_{s_1 s_4} + \tau_{s_2 s_4} = \{\{s_1, s_2, s_4\}, \{s_3\}\}\\
&\quad (=\tau_8 \text{ とする})\\
&M(\tau_4) = \tau_{s_1 s_3} = \tau_5\\
&M(\tau_5) = \tau_{s_1 s_4} = \tau_6\\
&M(\tau_6) = \tau_{s_2 s_4} = \{\{s_1\}, \{s_2, s_4\}, \{s_3\}\}\ (=\tau_9 \text{ とする})\\
&M(\tau_7) = \tau_{s_1 s_3} + \tau_{s_1 s_4} + \tau_{s_3 s_4} = \tau_3
\end{aligned}
$$

したがって，機械 M3-5 の Mm ペアは次の七つである．

$$
\begin{aligned}
&(\tau_2, \tau_1) = (\{\{s_1, s_3\}, \{s_2, s_4\}\},\ \{\{s_1, s_4\}, \{s_2, s_3\}\})\\
&(\tau_1, \tau_2) = (\{\{s_1, s_4\}, \{s_2, s_3\}\},\ \{\{s_1, s_3\}, \{s_2, s_4\}\})\\
&(\tau_8, \tau_3) = (\{\{s_1, s_2, s_4\}, \{s_3\}\},\ \{\{s_1, s_3, s_4\}, \{s_2\}\})\\
&(\tau_5, \tau_4) = (\{\{s_1, s_3\}, \{s_2\}, \{s_4\}\},\ \{\{s_1\}, \{s_2, s_3\}, \{s_4\}\})\\
&(\tau_6, \tau_5) = (\{\{s_1, s_4\}, \{s_2\}, \{s_3\}\},\ \{\{s_1, s_3\}, \{s_2\}, \{s_4\}\})\\
&(\tau_9, \tau_6) = (\{\{s_1\}, \{s_2, s_4\}, \{s_3\}\},\ \{\{s_1, s_4\}, \{s_2\}, \{s_3\}\})\\
&(\tau_3, \tau_7) = (\{\{s_1, s_3, s_4\}, \{s_2\}\},\ \{\{s_1, s_2, s_3\}, \{s_4\}\})
\end{aligned}
$$

Mm ペアを用いて状態変数依存度の少ない状態割当てを行える．機械 M3-5 は状態数が 4 なので，二つの状態変数の状態割当てを求める．機械 M3-5 では 2 状態ずつ二つのブロックに分かれる m 分割は τ_1 と τ_2 しかない．$\tau_1 \cdot \tau_2 = \pi(0)$ かつ $M(\tau_1) = \tau_2, M(\tau_2) = \tau_1$ であるから，表 3･14(b) のとおり，Y_1 が x と y_2 にのみ，Y_2 が x と y_1 にのみ依存する状態割当てを得る．

(2) 論理最小化に基づく方法

分割に基づく方法とは異なるアプローチとして，状態割当てを多値入力多値出力の論理最小化の枠組みの中で考える方法[3, 4]を紹介する．

説明を簡単にするため，出力関数は考えず，状態遷移関数のみ考える．有限状態機械の状態遷移関数 $\delta : I \times S \to S$ を二段組合せ回路で実現することを考えると，状態割当ては関数 δ の二段論理式を最小にする状態割当てを求める問題となる．ここでは，「最小」とは積和形論理式の積項数を最小にすることをいうものとする．また，$I = \{0,1\}^p, |S| = n$ とする．関数 δ は入力と出力の両方に状態記号をもつため，状態割当ての現状態への影響と次状態への影響の両方を考える必要がある．

状態割当ての現状態への影響は，関数 δ に多値入力論理式の最小化を適用することで考慮できる．例として，図 3･12(a) の機械 M3-6 を考える．現状態部分と次状態部分にそれぞれ状態数と同じ n 個のビットの二値ベクトルを割り当て，i 番目のビットが 1 のとき状態 s_i を含むものとする．図 3･12(a) の各遷移をそれぞれ二値ベクトルで表したものを図 3･12(b) に示す．関数の積項をこのようなベクトル形式で表現したものを**キューブ（cube）**と呼ぶ．関数はキューブの集合により表される．図 3･12(b) の最左列の四角で囲まれた数字は各キューブの番号を示す．キューブ c=[c_{IN} c_{PS} c_{NS}] は，入力，現状態，次状態を表す部分キューブ c_{IN}, c_{PS}, c_{NS} からなる．入力部分キューブ c_{IN} のみは 0,1 以外にドントケア値 ('-') も取り得る．例えば，図 3･12(b) の一番目のキューブ [0 1000 0001] は，現状態 s_1 で 0 が入

力されると次状態が s_4 になる遷移に対応する．一般に，c_{IN}, c_{PS}, c_{NS} はそれぞれ入力集合，現状態集合，次状態集合を表す．図 3･12(b) のキューブのように，入力部分キューブにドントケアビットを含まず，すなわち一つの入力のみを含み，現状態部分と次状態部分もそれぞれただ一つの状態を含むキューブを**最小項（minterm）**と呼ぶ．キューブ c が表す入力集合，現状態集合，次状態集合が最小項 m の入力，現状態，次状態をそれぞれ含むとき c は m を**被覆する（cover）**という．

現状態	次状態, z	
	x=0	x=1
s_1	s_4,0	s_2,0
s_2	s_4,0	s_4,1
s_3	s_1,0	s_1,0
s_4	s_1,0	s_3,1

(a) 機械M3-6

1	0 1000 0001	0 1100 0001
2	1 1000 0100	1 1000 0100
3	0 0100 0001	- 0100 0001
4	1 0100 0001	- 0010 1000
5	0 0010 1000	0 0011 1000
6	1 0010 1000	1 0001 0010
7	0 0001 1000	
8	1 0001 0010	

(b) 各遷移を表す最小項　(c) 最小化後のキューブ

図 3･12　機械 M3-6 と多値入力論理最小化

図 3･12(b) の最小項のキューブ集合に積和形論理式の最小化を適用すると，図 3･12(c) のキューブ集合を得る．例えば，図 3･12(c) の一番目のキューブは，現状態 s_1 も s_2 も入力 0 に対して s_4 に遷移することを意味する．したがって，s_1, s_2 にハミング距離 1 の符号，例えば，$s_1 \rightarrow 00, s_2 \rightarrow 01$ を割り当てれば，現状態について一つの積項 $\overline{y_1}$ で表せ，キューブに対する積項も，入力部分もあわせて $\bar{x}\ \overline{y_1}$ の一つですむ．一般的には，現状態に関する状態割当ての制約は，各キューブ c の現状態部分 c_{PS} が表す状態集合 S_{PS} に含まれるすべての状態の符号に対して真になり，S_{PS} に含まれない S のすべての状態の符号に対して偽になるような，現状態に対する二値論理のある積項が存在することである．この制約を**面埋め込み制約（face embedding constraint）**と呼ぶ．

次に，状態割当ての次状態への影響について考える．例えば，図 3･12(b) の最小項 1 と最小項 2 は，次状態部分は異なるが，入力部分と現状態部分のみを見た場合，一つにまとめて [− 1000] とできる．最小項 1,2 の次状態 s_4, s_2 にそれぞれ 11,01 の符号を仮に割り当てると，両方の符号で 1 になるビット，今の場合は最下位ビット，に限り，一つにまとめた部分キューブ [− 1000] をそのビットの**内項（implicant）**として用いることができる．このような状態割当てを後で求められるよう，図 3･12(b) の最小項 1 と 2 を一つにまとめて [− 1000 0101] というキューブをつくることとする．このキューブの次状態部分に含まれる状態の符号のビットごとの論理積をとると，値が 1 になるビットが入力部分 [− 1000] を内項とするビットを示す．更に，与えられたキューブ集合 C に対して，最小項 m を被覆するキューブが複数存在する場合，それらのキューブのうちのいずれかが m の次状態の符号ビットを 1 にすればよい．したがって，あるキューブの集合 C における最小項 m の次状態への状態割当ての制約は次のようになる：最小項 m を被覆するキューブの集合 $C_m \subseteq C$ について，各キューブ $c \in C_m$ の次状態集合 S_{NS} の状態の符号のビット積をとり，更にそれらのビット積のビット和をとったものが，m の次状態の符号と一致する．この制約を**次状態符号制約（next-state encoding constraint）**と呼ぶ．

多値入出力関数 δ の論理を最小化しつつ, 状態割当てに対する二種類の制約, すなわち, 面埋め込み制約と次状態符号制約を導出するため, 二値出力関数に対する**主項（prime implicant）**を，次状態記号を出力する関数が扱えるように拡張した，以下のような一般化主項を考える．

定義 4　状態の集合 $S_{PS} \subseteq S$ と符号化関数 $e : S \to \{0,1\}^l$ に対し，$s \in S_{PS}$ を符号化した状態 $e(s)$ を $e(s)^{(1)}e(s)^{(2)}\cdots e(s)^{(l)}$ とする．このとき，次を満たす符号化した状態部分キューブ $b^{(1)}b^{(2)}\cdots b^{(l)}$ を $\biguplus e(S_{PS})$ と表す．

$$b^{(i)} = \begin{cases} -, & \text{if } \exists s, \tilde{s} \in S_{PS}.\ e(s)^{(i)} = 1 \text{ かつ } e(\tilde{s})^{(i)} = 0 \\ 0, & \text{if } \forall s \in S_{PS}.\ e(s)^{(i)} = 0 \\ 1, & \text{if } \forall s \in S_{PS}.\ e(s)^{(i)} = 1. \end{cases}$$

定義 5　次状態関数 $\delta : I \times S \to S$ のキューブ c=$[c_{IN}\ c_{PS}\ c_{NS}]$ と符号化関数 e に対し，$[c_{IN} \quad \biguplus e(S_{PS}) \quad \bigcap e(S_{NS})]$ を符号化キューブ $e(c)$ と呼ぶ．ここで，S_{PS}, S_{NS} はそれぞれ c_{PS}, c_{NS} が表す現状態集合と次状態集合とし，$\bigcap e(S_{NS})$ は S_{NS} に含まれる状態の符号のビットごとの論理積とする．

定義 6　次状態関数 δ のキューブ c に関して，ある整数 l と符号化関数 $e : \{0,1\}^l$ が存在し，$e(c)$ が δ の状態 $s \in S$ を $e(s)$ で置き換えた関数 $e(\delta)$ の主項となるとき，c を f の**一般化主項（generalized prime implicant; GPI）**という．

二つのキューブ c_i, c_j の距離 d は，c_i と c_j の入力部分キューブの相異なる値をもつビットの数を d_{IN} とすると，現状態部分が同一のときは $d = d_{IN}$，異なるときは $d = d_{IN} + 1$ とする．距離 1 の二つのキューブ c_i, c_j を**併合する**とは次のようなキューブ c_k をつくることである：(1) 入力部分は，c_i, c_j とも値 v をとるビットでは v，値が異なるビットではドントケア値をとり，(2) 現状態部分と次状態部分はそれぞれ c_i, c_j の現状態部分と次状態部分のビット和をとる．

次状態関数 δ から一般化主項をすべて生成する方法として，主項を最小項から生成する方法を拡張した方法を以下に示す．

1. レベル $lev = 0$ とする．関数 δ のすべての最小項の集合をレベル 0 の集合 P_0 とする．
2. レベル lev の集合 P_{lev} の距離 1 の任意のキューブ対 (c_i, c_j) について，その併合キューブ c_k の次状態部分がすべての状態を含む（すなわち次状態部分が $11\cdots1$）場合を除き，c_k を集合 P_{lev+1} に追加する．c_i, c_j のうち，c_k と現状態部分も次状態部分も同一のものに印をつける．
3. レベル $lev + 1$ のキューブが存在すれば，レベル $lev = lev + 1$ とし，ステップ 2 に戻る．
4. 集合 $P_i(i = lev, \ldots, 1, 0)$ の印のない要素で，現状態部分がすべての状態を含む（すなわち現状態部分が $11\cdots1$ の）キューブ c それぞれについて，次の条件を満たすキューブ $c_j \in P_k(k = 0, 1, \ldots, lev)$ に印をつける：(a) c の入力部分が表す入力集合が c_j の入力部分が表す入力集合を含み，かつ，(b) c と c_j の次状態部分が同一である．
5. 集合 $P_i(i = 0, 1, \ldots, lev)$ の要素で印のないものの集合を一般化主項の集合とする．□

図 3･12(b) の最小項から一般化主項を生成する様子を図 3･13 に示す．図中，横線は印のついたキューブを示す．丸で囲んだ数字は一般化主項の番号を示す．例えば，レベル 0 の一番目と三番目のキューブを併合して，レベル 1 の二番目のキューブを得る．現状態部分が同一でないためこの併合では印はつかない．生成した一般化主項を列に，最小項を行にした被覆表を図 3･14 に示す．i 行 j 列に 1 をもつとき，最小項 i が GPI_j に被覆されることを示す．

レベル0	レベル1	レベル2	レベル3
① 0 1000 0001	⑤ - 1000 0101	⑭ - 1100 0101	㉑ - 1110 1101
② 1 1000 0100	⑥ 0 1100 0001	⑮ - 1010 1101	㉒ 0 1111 1001
~~0 0100 0001~~	~~0 1010 1001~~	~~0 1110 1001~~	㉓ - 0111 1011
~~1 0100 0001~~	~~0 1001 1001~~	~~0 1101 1001~~	
~~0 0010 1000~~	~~1 1100 0101~~	~~0 1111 1001~~	
~~1 0010 1000~~	⑦ 1 1010 1100	~~0 1011 1001~~	
③ 0 0001 1000	⑧ 1 1001 0110	~~1 1110 1101~~	
④ 1 0001 0010	⑨ - 0100 0001	⑯ 1 1101 0111	
	~~0 0110 1001~~	⑰ 1 1011 1110	
	~~0 0101 1001~~	⑱ - 0110 1001	
	~~1 0110 1001~~	⑲ - 0101 1011	
	⑩ 1 0101 0011	~~0 0111 1001~~	
	⑪ - 0010 1000	~~1 0111 1011~~	
	⑫ 0 0011 1000	⑳ - 0011 1010	
	~~1 0011 1010~~		
	⑬ - 0001 1010		

図 3･13　一般化主項生成

最小項 ＼ 一般化主項	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	1				1	1								1	1						1	1	
2		1			1		1	1						1	1	1	1				1		
3						1			1					1				1	1		1	1	1
4									1	1				1		1		1	1		1		1
5											1	1			1			1		1	1	1	1
6							1				1				1		1	1		1	1		1
7			1									1	1						1	1		1	1
8				1				1		1			1			1	1		1	1			1

図 3･14　一般化主項による被覆表

符号ビット数を l とすると，状態割当て問題は，すべての最小項を被覆し，かつ，選択した一般化主項が課すすべての符号制約を満たすビット数 l の状態割当てが存在するような，最小サイズの一般化主項の集合を求める問題となる．状態数 n に対して，符号ビット数 l は最小値 $\lceil log_2 n \rceil$ からはじめ，制約が満たせない場合は満たすまでビット数 l を増やしていく．

機械 M3-6 に対して，被覆条件と符号制約を満たす最小サイズの一般化主項集合を選択すると，例えば，$\{GPI_5, GPI_6, GPI_{10}, GPI_{20}\}$ となる．選択した一般化主項が課す面埋め込み制約は $\{s_1, s_2\}, \{s_2, s_4\}, \{s_3, s_4\}$ の三つである．各最小項ごとの次状態符号制約は次のようになる．

（最小項 1）　$e(s_4) = e(s_2)e(s_4) + e(e_4)$

（最小項 2）　$e(s_2) = e(s_2)e(s_4)$

（最小項 3）　$e(s_4) = e(s_4)$

（最小項 4）　$e(s_4) = e(s_3)e(s_4)$

（最小項 5,6,7）　$e(s_1) = e(s_1)e(s_3)$

（最小項 8）　$e(s_3) = e(s_3)e(s_4) + e(s_1)e(s_3)$

これらの次状態符号制約をまとめると次の四つになる．

(1) $e(s_2) < e(s_4)$ （最小項 2 の制約より）
(2) $e(s_4) < e(s_3)$ （最小項 4 の制約より）
(3) $e(s_1) < e(s_3)$ （最小項 5,6,7 の制約より）
(4) $e(s_3) = e(s_1) + e(s_4)$ （最小項 8 の制約と (2),(3) より）

機械 M3-6 の面埋め込み制約と次状態符号制約を満たす状態割当ては，符号ビット数が 2 のとき，$s_1 \rightarrow 01, s_2 \rightarrow 00, s_3 \rightarrow 11, s_4 \rightarrow 10$ となる．状態割当て後のキューブは次の四つで済む．

-	01	-0	（GPI_5 に対応）
0	0-	10	（GPI_6 に対応）
1	-0	1-	（GPI_{10} に対応）
-	1-	-1	（GPI_{20} に対応）

状態遷移関数は $Y_1 = \bar{x}\ \overline{y_1} + x\ \overline{y_2}, Y_2 = y_1$ となる．一番目のキューブ $[-\ 01\ -0]$ は関数出力部分に 1 のビットがないため不要であり，最終的に必要な積項は $\bar{x}\ \overline{y_1}, x\ \overline{y_2}, y_1$ の三つである．

文献 5) では，面埋め込み制約と次状態符号制約を満たす状態割当てを求めるために dichotomy という概念を用いる方法が提案されている．多段回路の最小化を考慮した状態割当て方法は文献 6) などで示されている．

■参考文献

1) Z. Kohavi, “Switching and Finite Automata Theory, (Second Edition),” McGraw-Hill, 1978.
2) J. Hartmanis, “On the State Assignment Problem for Sequential Machines I,” IRE Trans. Electronic Computers, vol.EC-10, pp.157-165, Jun. 1961.
3) S. Devadas and A.R. Newton, “Exact Algorithms for Output Encoding, State Assignment, and Four-Level Boolean Minimization,” IEEE Trans. Computer-Aided Design, vol.10, no.1, pp.13-27, Jul. 1991.
4) T. Villa, T. Kam, R. Brayton, and A. Sangiovanni-Vincentelli, “Synthesis of Finite State Machines: Logic Optimization,” Kluwer Academic Publishers, 1997.
5) A. Saldanha, T. Villa, R. Brayton, and A. Sangiovanni-Vincentelli, “A framework for satisfying input and output encoding constraints,” Proc. Design Automation Conf., pp.170-175, Jun. 1991.
6) S. Malik, L. Lavagno, R. Brayton, and A. Sangiovanni-Vincentelli, “Symbolic Minimization of Multilevel Logic and the Input Encoding Problem,” IEEE Trans. Computer-Aided Design, vol.11, no.7, pp.825-843, Jul. 1992.

■1 群 - 8 編 - 3 章

3-4 同期式順序回路の解析

（執筆者：樋口博之）[2008 年 10 月 受領]

順序回路の入力端子から適当な入力系列を印加したときの出力系列だけを観測して，その回路の状態を同定する問題について考える．

3-4-1 実験による順序回路の解析

入力系列を順序回路に印加し，それに対する出力系列を観測することにより，内部動作の解析を行うことを**実験（experiment）**と呼ぶ[1)]．実験開始時の回路の状態を**初期状態（initial state）**と呼び，実験終了時の回路の状態を**最終状態（final state）**と呼ぶ．実験は**適応実験（adaptive experiment）**と**既定実験（preset exepriment）**に分類される．適応実験は，各時刻での入力をそれ以前の出力に依存して変えられる実験である．既定実験は，入力系列全体が実験の出力とは独立にあらかじめ決められている実験である．本節では，順序回路の動作を表す有限状態機械は完全指定，最小，かつ強連結，すなわち任意の状態対が到達可能であると仮定する．

3-4-2 ホーミング系列と同期系列

定義 1　入力系列 X に対して，初期状態にかかわらず X に対する機械 M の出力系列から最終状態を決めることができるとき，X を M の**ホーミング系列（homing sequence）**という．

表 3･15 の機械 M4-1 に対し，系列 10 はホーミング系列である．系列 10 に対して，初期状態が s_1, s_2, s_3, s_4 の場合，出力系列と最終状態の組はそれぞれ $(11, s_3)$, $(01, s_1)$, $(11, s_3)$, $(00, s_2)$ であるので，出力系列が 00, 01, 11 のとき最終状態はそれぞれ s_2, s_1, s_3 に特定できる．

表 3･15　機械 M4-1

現状態	次状態, 出力	
	入力 0	入力 1
s_1	$s_2, 0$	$s_4, 1$
s_2	$s_4, 0$	$s_3, 0$
s_3	$s_1, 1$	$s_4, 1$
s_4	$s_3, 1$	$s_1, 0$

ホーミング系列の求め方を考える．有限状態機械が初期状態としてとり得る状態の集合を**初期不確定性（initial uncertainty）**と呼ぶ．機械 M が最初にどの状態にいる可能性もあるとすると，初期不確定性は M の状態すべてからなる．最終状態を同定するには，初期不確定性から始めて，その不確定性を削減していき，状態が一意に確定できればよい．例えば，機械 M4-1 に入力 1 を印加すると，出力が 0 か 1 かに依存して次状態の不確定性はそれぞれ $(s_1 s_3)$ か $(s_4 s_4)$ になる．このとき，$(s_1 s_3)(s_4 s_4)$ は不確定性 $(s_1 s_2 s_3 s_4)$ の 1 遷移先（1-successor）であるという．遷移先の不確定性は一般的に不確定性の集合となる．この集

合を**不確定性ベクトル（uncertainty vector）**と呼ぶ．個々の不確定性では遷移元の各状態に対応して要素が一つ存在するため，$(s_4 s_4)$ のように同一の状態が複数出てくることもある．不確定性ベクトルに含まれるすべての不確定性が一つの状態のみからなる不確定性ベクトルを**自明な不確定性ベクトル（trivial uncertainty vector）**と呼ぶ．不確定性ベクトルに含まれるすべての不確定性について，その要素が一つの状態か同一の状態の繰り返しのいずれかである不確定性ベクトルを**均質不確定性ベクトル（homogeneous uncertainty vector）**と呼ぶ．

遷移先不確定性ベクトルを次々に求めるために，**遷移木（successor tree）**と呼ばれる図を用いる．遷移木は 0 からはじまる連続したレベルが付いた節点からなる．レベル i の節点からは機械の入力記号に応じて $x_1, x_2, \ldots x_p$ のラベルのついた p 本の枝が出る．レベル i から出た枝はレベル $i+1$ の節点につながる．遷移木の各節点に不確定性ベクトルを対応付ける．一番上のレベル 0 の節点に初期不確定性を対応付け，レベル 1 の p 個の節点の各々に初期不確定性の遷移先の不確定性ベクトルを対応付ける．レベル 0 の節点からレベル i の節点までの i 本の枝の系列を経路と呼ぶ．各経路はある入力系列を表す．遷移木でホーミング系列を求めるために，節点が次の二条件のいずれかを満たすときに終端とする．

1. レベル i の節点 v の不確定性ベクトルのすべての非均質不確定性が，i より小さいレベルの，ある一つの節点の不確定性ベクトルに含まれるなら，節点 v を終端とする．
2. レベル i のある節点の不確定性ベクトルが自明な不確定性ベクトルか均質不確定性ベクトルのいずれかのとき，レベル i の全節点を終端とする．

上の終端条件を適用した遷移木を**ホーミング木（homing tree）**と呼ぶ．この木の経路で，初期不確定性から始まり，均質不確定性ベクトルで終わる経路が，最短のホーミング系列に対応する．機械 M4-1 の初期不確定性 $(s_1 s_2 s_3 s_4)$ に対するホーミング木を図 3･15 に示す．機械 M4-1 は三つの最短のホーミング系列 00,10,11 をもち，図 3･15 の太線がそれらに対応する．

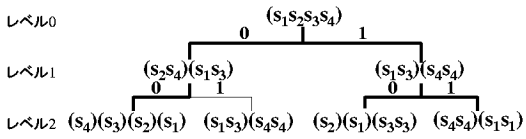


図 3･15　機械 M4-1 のホーミング木

定理 1　n 状態の有限状態機械は，長さがたかだか $(n-1)^2$ のホーミング系列をもつ．□

定義 2　入力系列 X に対して，初期状態と出力にかかわらず有限状態機械 M の最終状態を決めることができるとき，X は M の**同期系列（synchronizing sequence）**であるという．

同期系列がホーミング系列と異なる点は，出力を見ずに最終状態を同定できることである．同期系列のための遷移木では，各節点の不確定性は不確定性ベクトルでなく，出力を無視し

て i 番目のレベルの節点にレベル 0 からその節点までの経路の i 個の入力記号を印加した後の状態の集合となる．節点が次の二条件のいずれかを満たすとき終端とする．

1. レベル i の節点 v の状態集合が，i より小さいレベルの，ある一つの節点の状態集合と同じかあるいは包含していれば，v を終端とする．
2. レベル i のある節点の状態集合の要素が一つのとき，レベル i の全節点を終端とする．

この終端条件を適用してつくられた遷移木を**同期木（synchronizing tree）**と呼ぶ．同期木の経路が最短の同期系列を表す必要十分条件は，その経路が初期不確定性から始まり，単一の要素からなる状態集合の節点で終わることである．機械 M4-1 の同期木を図 3･16 に示す．図 3･16 から，機械 M4-1 の最短の同期系列は 10101，最終状態は s_4 であることが分かる．

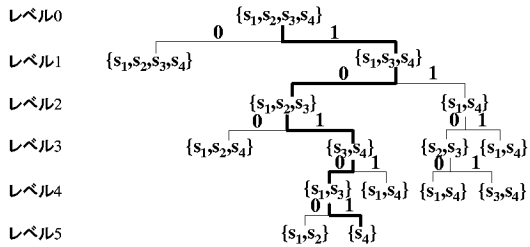


図 3･16　機械 M4-1 の同期木

ホーミング系列と異なり，同期系列は必ずしも存在するとは限らない．

定理 2　n 状態機械が同期系列をもつなら，その長さはたかだか $(n-1)^2 n/2$ である．□

3-4-3 識別系列

定義 3　機械 M に入力系列 X を印加したときの出力系列が各々の初期状態に対して異なるとき，X は M の**識別系列（distinguishing sequence）**であるという．

識別系列は初期状態を同定する．識別系列は必ずホーミング系列である．しかし，逆に，ホーミング系列は必ずしも識別系列であるとは限らない．識別系列を生成するために用いる遷移木を識別木という．識別木は，節点について次の三条件のいずれかを満たすとき，終端になる遷移木である．

1. レベル i の v の不確定性ベクトルに含まれるすべての非均質不確定性が，i より小さいレベルの，ある一つの節点の不確定性ベクトルに含まれるなら，v を終端とする．
2. 節点 v の不確定性ベクトルが同じ状態を二つ以上含む不確定性を要素としてもつなら，節点 v を終端とする．
3. レベル i のある節点が自明な不確定性ベクトルをもつときレベル i の全節点を終端とする．

初期不確定性から始まり，自明な不確定性ベクトルに対応する節点で終わる識別木の経路が，最短の識別系列を表す．例えば，機械 M4-1 に対する識別木を図 3･17 に示す．図 3･17 より，機械 M4-1 の最短の識別系列は 00 であることが分かる．識別系列 00 による出力が 00, 01, 10, 11 のとき，それぞれ初期状態は s_1, s_2, s_3, s_4（最終状態は s_4, s_3, s_2, s_1）である．

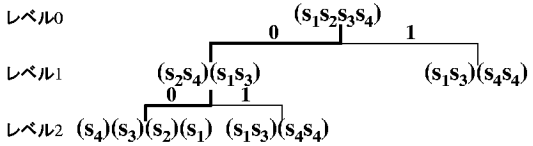
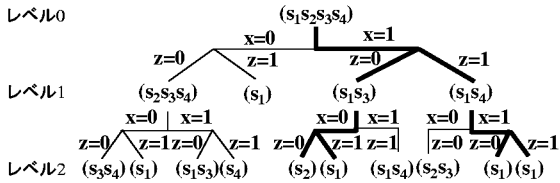


図 3･17　機械 M4-1 の識別木

適応実験のための識別系列の生成では，遷移木を拡張した**適応木（adaptive tree）**を用いる．適応実験では出力に依存して入力を選択できるため，遷移木での不確定性を出力に応じて個々の不確定性に分けて考えることができる．したがって，適応木では入力に対する枝分かれだけでなく出力に対する枝分かれも存在する．適応木では，各々の水平の線はそのラベルで示される入力記号の印加を表す．下あるいは斜め下への線はそのラベルで示される出力記号の応答を表す．例として，表 3･18(a) の機械 M4-2 に対する適応木を図 3･18(b) に示す．

現状態	次状態，出力	
	入力0	入力1
s_1	s_2, 0	s_1, 1
s_2	s_4, 0	s_3, 0
s_3	s_1, 1	s_4, 1
s_4	s_3, 0	s_1, 0

(a) **機械**M4-2



(b) **機械**M4-2**の適応木**

図 3･18　機械 M4-2 と適応木

図 3･18(b) の太線が最短の適応識別系列に対応する．1 を入力し，その出力に応じて次の (1) か (2) を行う：(1) 出力が 0 なら 0 を入力し，(2) 出力が 1 なら 1 を入力する．(1) のとき，出力が 0 なら初期状態は s_4（最終状態は s_2），出力が 1 なら初期状態は s_2（最終状態は s_1）である．(2) のとき，出力が 0 なら初期状態は s_3（最終状態は s_1），出力が 1 なら初期状態は s_1（最終状態は s_1）である．

本節で述べた，実験による順序回路の解析は文献 2),3) に詳しく述べられている．

■参考文献

1) E.F. Moore, “Gedanken-experiments on Sequential Machines,” Automata Studies, Annals of Mathematics Studies, no.34, pp.129-153, Princeton University Press, 1956.
2) Z. Kohavi, “Switching and Finite Automata Theory, (Second Edition),” McGraw-Hill, 1978.
3) 当麻喜弘・内藤祥雄・南谷崇, “ 順序機械, ” 岩波書店，1982.

■1 群 - 8 編 - 3 章

3-5 非同期式順序回路とその設計

（執筆者：今井　雅）[2008 年 10 月 受領]

非同期式回路は，グローバルクロックを用いず，事象生起の因果関係を駆動原理とする回路である．非同期式回路設計では，システムの生涯を通じて論理ゲートや配線に起こり得る遅延変動の振る舞いを表したモデル，すなわち遅延モデル（遅延仮定）が重要な役割を果たし，これまで様々な遅延モデルとそのモデルに基づいた回路設計方式が提案されてきた．本節では，はじめに代表的な遅延モデルについて紹介し，次に小規模な非同期式順序回路合成方法としてハフマン（Huffman）式回路合成方法とマラー（Muller）式回路合成方法を紹介する．更に，ディジタル回路の基本動作となるレジスタ間データ転送に関して，代表的な非同期式回路実現方法を紹介する[1, 2, 3]．

3-5-1 代表的な遅延モデル

（1）Huffman Model（ハフマンモデル）

素子にも配線にも遅延が存在し，上限値は既知とする．このモデルのもとでは，遅延の上限値で決まる一定時間内に回路は必ず安定するため，入力変化の間隔をこれより大きく取ることで，すべての入力変化は回路が安定しているときに起きるように設計することができる．この「すべての入力変化は回路が安定しているときに生ずる」という仮定は基本モード（Fundamental Mode）と呼ばれる．

また，ハフマンモデルの発展として，多入力変化動作がグループ化され，ある一定時間内に一気に変化する限り，その多入力変化がいつ起こってもよいと仮定するものをバーストモード（Burst mode:BM）モデルと呼び，BM モデル及び拡張バーストモード（Extended Burst Mode:XBM）モデルに基づいた回路設計方式が提案されている．

（2）Delay Insensitve Model（DI モデル）

素子の出力と配線に，有限であるが上限は未知の遅延が存在すると仮定する．DI モデルは最も悲観的な遅延モデルであり，単一出力論理ゲートを素子とする場合，DI モデルのもとでは意味のある回路を構成することはできない．

（3）Speed Independent Model（SI モデル）

素子の出力にのみ，有限であるが上限は未知の遅延が存在すると仮定する．すなわち，配線の遅延は無視できると仮定するモデルである．

（4）Quasi Delay Insensitive Model（QDI モデル）

DI モデルに対して，配線の分岐のうち，必要な場合には分岐点からその各分岐先への到達時間差は無視することができるという等時分岐（Isochronic Fork）の仮定を加えたモデルである．QDI モデルのもとでは任意の機能を持つ回路を設計することができる．

3-5-2 ハフマン（Huffman）回路設計方式

ハフマン回路は遅延の上限値が既知とするハフマンモデルに基づいて設計される回路であり，その基本モデルは図 3･19（b）に示すように，外部入力，外部出力，及びフィードバック状態変数と組合せ回路をもつ．状態はフィードバックループパスに保持され，状態変化が速く起こりすぎる場合には，遅延素子が挿入される．入力変化は基本モードの仮定に基づいて

おり，ハフマン回路は基本モード回路とも呼ばれる．また，正しいハフマン回路を設計するためには，入力変化に関して基本モードに加えて更に制約を課す必要があり，これまで，いくつかの制約とそれに応じた回路合成手続きが提案されてきた．代表的な制約として，同時にはただ 1 本の入力しか変化しない，すなわち，1 本の入力が変化してから次の入力が変化するまでにはある時間間隔を必要とする「単一入力変化」があげられる．この仮定は非常に厳しいため，多入力変化及び多出力変化を仮定するアプローチとしてバーストモード（BM）などが提案されている．

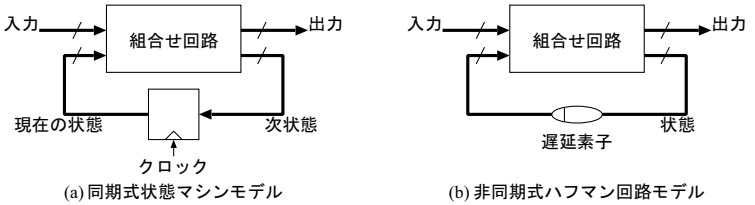


図 3･19　同期式状態マシンモデルと非同期式ハフマン回路モデル

ハフマン回路の設計では，非同期式状態マシン（Asynchronous Finite State Machine: AFSM），あるいは BM マシン，XBM マシンなどから得られるフローテーブルの形式で与えられる仕様から設計を行う．図 3･19 に示すように，ある「状態」に基づいて次の値を生成するという点では同期式状態マシンモデルと同様に考えることができ，合成手続きも同期式状態マシンと同様のものを用いることができる．ただし，非同期式タイミングモデルのもとで正しく動作する回路を得るためには，回路に期待しない信号変化，すなわちグリッチが生じないようにしなければならない．非同期式回路では，グリッチが生ずるとそれを受け取った回路が有効な信号遷移であると解釈し，誤った振る舞いをする可能性があるためである．単一入力変化の仮定に基づいたハフマン回路の典型的な合成手続きと，非同期式回路設計として考慮しなければならない点を以下に示す．

1. ハフマンフローテーブルにより仕様を記述する．

2. 両立的状態（Compatible state）をまとめることにより，フローテーブルを簡単化する．

3. 状態割当（State assignment）を行い，各状態に 2 値符号を割り当てる．

   このとき，現在の状態と次状態を表す符号が 2 ビット以上異なるとき，素子の遅延値によっては本来の状態と異なる安定状態に到達することがないように状態を割り当てなければならない．例えば，図 3･20(a) のようなフローテーブルが与えられたとき，状態割当てとして図 3･20(b) に示すものを用いるとする．状態 b において入力値が 1 から 0 に遷移するとき，割当て 1 では，「01」から「10」への遷移が起こり，2 ビットの状態変数が同時に遷移しない限り，一時的に状態「00」（状態 a）あるいは「11」（状態 d）を通ることになる．状態 a，状態 d ともに入力が 0 のときは状態 a への遷

移が生ずるため，結果として本来とは異なる状態に到達する．したがって，割当て 1 は用いることができない．一方，割当て 2 では，「011」から「110」への遷移が起こる．このとき，「111」や「010」の値を取り得るが，S_1 は 1 で安定であり，「000」や「101」への遷移は起こらない．したがって，割当て 2 を用いることができる．

	入力	
	0	1
a	a	b
b	c	b
c	c	d
d	a	d

(a) フローテーブル

	割当て1	割当て2	
	S_1 S_0	S_2 S_1 S_0	
a	0 0	0 0 0	
b	0 1	0 1 1	
c	1 0	1 1 0	
d	1 1	1 0 1	

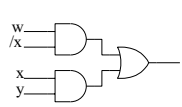
(b) 状態割当て

図 3·20　フローテーブルと状態割当ての例

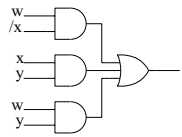
4. 論理簡単化（Logic minimization）を行い，2 値符号化されたフローテーブルから出力値と状態値の最適なネットリストを求める．

   ここで 2 レベル論理簡単化を行う際，積和形式の実現が静的 1 ハザードを生じないようにしなければならない．例えば，図 3·21 に示すようなカルノー図が与えられたとき，回路実現としては図 3·21(a)，(b) が考えられる．ここで，回路実現 (a) では，$(w, y) = (1, 1)$ で x が 0 から 1（あるいはその逆）に遷移するとき，瞬間的に 0 を出力し得るが，回路実現 (b) は積項 wy をもつため出力が 1 に保たれ，静的 1 ハザードが生じない．

yz \ wx	00	01	11	10
00	0	0	0	1
01	0	-	0	1
11	0	1	1	1
10	-	1	1	1



(a) 静的 1 ハザードをもつ回路



(b) 静的 1 ハザードをもたない回路

図 3·21　積和形式による回路実現例

このように得られたハフマン回路は，論理ゲートや配線の遅延が予測された範囲内の値を取る限り，どんな値を取っても正しく動作することが保証された回路となる．また，上記は単一入力変化を仮定した場合であるが，多入力変化，多出力変化を仮定する BM モデルや XBM モデルに基づいた回路設計方式が提案されており，回路設計支援環境として ACK システム[4)]などが実装されている．

3-5-3 マラー（Muller）回路設計方式

マラー回路はマラーモデル（Muller model）とも呼ばれる SI モデルに基づいた回路設計方式であり，仕様記述の段階では内部状態という概念をもたない．その基本モデルは図 3·22 に示すように，外部入力，外部出力，フィードバックループと組合せ回路からなる．ハフマン回路が基本モード回路と呼ばれるのに対して，マラー回路はインプット−アウトプットモード（Input-Output Mode）回路とも呼ばれる．

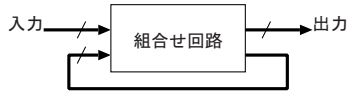


図 3·22　非同期式マラー回路モデル

通常，マラー回路設計法では，ハフマン回路設計法などと異なり，回路の上位レベル仕様を信号遷移グラフ（Signal Transition Graph: STG）などのグラフ表現から得られる状態グラフ（state graph: SG）によりモデル化する．STG を用いた合成は以下のステップからなる．

1. 以下の条件を満たすように仕様を STG で記述する．
   (a) 選択は排他的な入力信号により制御されること
   (b) プレース内のトークンはたかだか一つであること
   (c) デッドロックしないこと
   (d) どのように動作しても，ある信号の遷移は立ち上がりと立ち下がりが必ず交互に行われること
   (e) 任意の枝 $a* \rightarrow b*$ に対し，$a*$ の反対の遷移が起こる前に $b*$ が確実に起こることを保証する枝が存在すること
2. STG から状態グラフを生成し，任意の状態の組が完全状態コード（Complete State Code:CSC）をもつようにする．CSC とは，二つの状態が異なる 2 値ベクトルでラベル付けされたユニーク状態コード（Unique State Code: USC）をもつか，もしユニーク状態コードをもたなくてもそれぞれの状態で変化し得る出力の集合が同じであることをいう．もし状態グラフが CSC をもたないと，その原因となる出力において，それが取るべき値を信号線の値の組から一意に決めることができず，回路が生成できない．仕様を変更する手法としては，プロトコルをシャッフルする方式や，状態変数を挿入する手法などがある[2)]．
3. CSC をもつ状態グラフに対して，論理最適化を行い，ハザードなし回路を生成する．この論理最適化は使用するテクノロジーに依存する．代表的なものとして，複合ゲート（Complex atomic gates），一般化 C 素子（Generalized C elements）及び標準 C 素子（Standard C elements）を用いた方式がある[2)]．
4. 最後に，与えられたゲートライブラリを用いてテクノロジーマッピングを行い，回路を得る．

このように得られたマラー回路は遅延の上限値は未知と仮定した遅延モデルのもとで設計された回路であり，信号線の遅延は無視できるという仮定のもとで，ゲート遅延がいくら大きくなっても正しく動作することが保証された回路となる．マラー回路の設計支援ツールとして，STG を入力とし，テクノロジーマッピングされた回路を出力する Petrify[5)]が広く利用されている．

3-5-4 レジスタ間データ転送の非同期式実現

非同期式回路では，データ転送が必要かつ可能になったら転送元レジスタからデータが出力され，組合せ回路を経て転送先レジスタへ書き込まれる．次にその完了を示す応答信号が転送先レジスタから生成され，この応答信号が次のデータ転送を要求するトリガとなる．このように要求−応答ハンドシェイクに基づいてデータ転送が行われる．また，クロックなしで n ビット幅のデータを転送先レジスタに書き込むために，データに何らかのタイミング情報が付加される．その代表的な方式として，以下の二つがあげられる．

（1）束データ方式

n ビットのデータ信号線に対して，信号線が安定したことを表す 1 ビットのストローブ信号を付加する方式である．束データ方式は，遅延の上限値を既知とするハフマンモデル，BM モデル，XBM モデルのもとで用いることができる．

束データ方式の組合せ回路は同期式組合せ回路と同様の回路を用いることができる．転送を行うたびにデータパスの初期化を行う必要はなく，要求−応答ハンドシェイクプロトコルとして，図 3･23（a）に示す 2 フェイズ（2-Phase）プロトコルと，図 3･23（b）に示す 4 フェイズ（4-Phase）プロトコルのいずれも用いることができる．

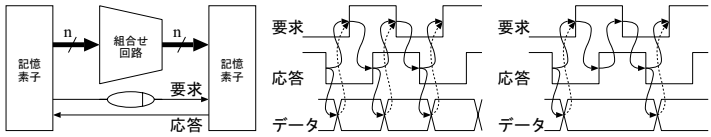


図 3･23　束データ方式回路モデルとハンドシェイクプロトコル

2 フェイズプロトコルでは，応答信号の 0 から 1 への遷移と 1 から 0 への遷移の両方を区別することなく，次のデータの処理を始める．4 フェイズプロトコルでは，応答信号の 1 から 0 への遷移（あるいはその逆）のみがトリガとなり，次のデータの処理を始める．応答信号が立ち上がってから応答信号が立ち下がるまでは回路の初期化を行う期間であり，このオーバーヘッドがシステムのサイクルタイムにそのまま現れると十分な速度性能を得ることが困難となる．プロトコルだけ比較すると 2 フェイズプロトコルの方がオーバーヘッドが小さいと考えられるが，記憶素子には立ち上がりと立ち下がりの両エッジで値の更新を行う素子を用いなければならず，レベル論理ではなく信号遷移論理で制御回路を実現しなければならないなどの制約もあり，4 フェイズプロトコルが広く使われている．

束データ方式の利点は，単純で，同期式回路の組合せデータパス構成要素を用いることができること，更に，動作条件の変化にある程度対応できることである．一方，欠点としては，データに依存した速度向上が得られないことである．

（2）2 線 2 相式

もう一つの代表的なデータパス実現方式として，各データ信号線を 2 本の信号線対 (x_p, x_n) で符号化する方式がある．$(x_p, x_n) = (1,0)$ を論理 1 に，$(x_p, x_n) = (0,1)$ を論理 0 に対応させて符号語とし，状態 $(x_p, x_n) = (0,0)$ で論理 0 でも論理 1 でもないスペーサと呼ば

れる中立的状態を表し，2 線式符号とスペーサを図 3･24 に示すように交互に送ることでデータ転送を繰り返す．転送元レジスタから出力される信号がスペーサから符号語に遷移し，組合せ回路で演算が行われて転送先レジスタに書き込まれるまでの期間を稼働相，転送元からスペーサが出力されて回路の初期化が行われる期間を休止相と呼ぶ．

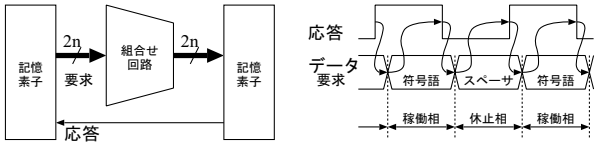


図 3･24　2 線 2 相式回路モデル

最も簡単な 2 線式回路の設計方式は，はじめに 1 線式回路を合成した後，図 3･25 に示すように各論理ゲートを対応する 2 線式回路に変換し，仮定する遅延モデルに従って回路の安定を監視する完了信号生成回路を付加する方式である．

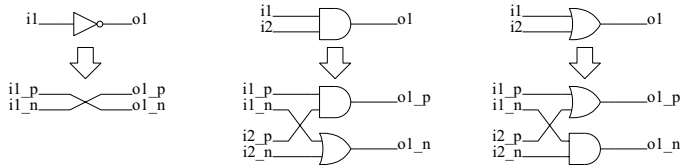


図 3･25　1 線式回路実現と 2 線式回路実現

束データ方式は図 3･23 に示すように，n ビットのデータ線に対して 1 ビットの要求信号が付加され，それに対する応答信号を加えて $n+2$ 本の信号線からなるのに対し，2 線 2 相式は図 3･24 に示すように n ビットのデータ線が 2 線式符号化され，それに対する応答信号を加えた $2n+1$ 本の信号線からなる．そのため，回路規模が大きくなりやすい．また，稼働相と休止相を繰り返す方式であるため，信号遷移数が多く，消費電力も大きくなるのが欠点である．しかし，ビットごとにタイミング情報を保持するため，遅延変動が大きくなっても正しく動作することが保証できる．また，様々な変動要因やデータの値に依存して遅延時間が大きく異なる場合，同期式回路や束データ方式実現ではその中の最大遅延により速度性能が制約されるのに対し，2 線 2 相式回路ではビットごとにデータが到着したことを認識できるため処理速度に合わせて動作させることができ，平均遅延性能を享受することができる．

■参考文献

1) Editted by Jens Sparso and Steve Furber, “Principles of Asynchronous Circuit Design –A Systems Perspective,” Kluwer Academic Publishers, 2001.
2) Chris J.Myers 著, 米田友洋訳, “非同期式回路の設計,” 共立出版株式会社, 2003.
3) Scott Hauck, “Asynchronous design methodologies: An overview.,” Proceedings of the IEEE, vol.83, no.1, pp.69-93, Jan. 1995.
4) Hans Jacobson, et al., “High-level asynchronous system design using the ACK framework,” In Proc. Async 2000, IEEE Computer Society Press, pp.93-103, Apr. 2000.

5) Jordi Cortadella, et al., “Technology mapping of speed-independent circuits based on combinational decomposition and resynthesis,” In Proc. European Design and Test Conference, pp.98-105, 1997. http://www.lsi.upc.es/~jordicf/petrify/.