DA−90−14

FPGA 上の組合せ回路および順序回路のための新しい論理関数表現法

安岡孝一
京都大学大型計算機センター
yasuoka@kudpc.kyoto-u.ac.jp

あらまし LUT 型 FPGA 上での組合せ回路の各結線の論理関数を記述するための表現法として、SPD (Sums of Products to be Distinguished) を提案する。また、LUT 型 FPGA 上での順序回路の状態割当を記述し、さらに各結線の論理関数を記述するための表現法として、SPSD (Sums of Products and States to be Distinguished) を提案する。

キーワード LUT 型 FPGA、順序回路、状態割当、最小化

New Algebraic Expressions for Combinational and Sequential Circuits on FPGAs

Koichi Yasuoka
Kyoto University Data Processing Center
yasuoka@kudpc.kyoto-u.ac.jp

Abstract In this paper, SPD (Sums of Products to be Distinguished) is proposed for a new algebraic expression of combinational circuits on LUT-based FPGA. Also SPSD (Sums of Products and States to be Distinguished) is proposed for a new algebraic expression of state assignments and logic functions on LUT-based FPGA realizing sequential circuits.

key words LUT-based FPGA, sequential circuit, state assignment, optimization

1 はじめに

同期式順序回路を設計する際の最大の問題は、各状態にどのように状態割当をおこなうかにある。状態割当を順序機械の分割によっておこなう方法は、1960年のHartmanisの論文[1]に端を発する。こののち、分割された順序機械間の依存関係を小さくするための研究が多くなされ[2, 3]、それらは最終的に分割対という形でKohaviによってまとめられた[4]。その後1980年代までに、状態割当に関して多くの研究がなされた[5, 6, 7, 8]が、これらはいずれも、状態数の対数程度のFlipFlopを用いて状態割当をおこなうものであった。

1990年代に入り、LUT型FPGAが実用化されるに至って、状況は一変した。Xilinx XC 3000シリーズに代表されるLUT型FPGAは、k-LUTとよばれるk入力1出力の万能組合せ回路と、D-FlipFlopとから構成されており、通常k-LUT 1個当たりD-FlipFlopが1あるいは2個という構造になっている。すなわち、これまでの論理回路からすると、D-FlipFlopの数が圧倒的に多く、非常に冗長な状態割当が可能となっているのである。この結果、FPGA上での状態割当に関して多くの研究がなされるようになってきている[9, 10, 11]が、このように大量のD-FlipFlopを含むアーキテクチャにおいて、実際にどのような状態割当が有効なのかは、今のところよくわかっていない。

本稿では、LUT型FPGA上での非常に冗長な状態割当を記述し、さらに各結線の表す次状態関数の部分関数を記述するための表現法として、SPSD (Sums of Products and States to be Distinguished)を提案する。SPSDは、LUT型FPGA上の結線の表す関数を全て表現することができる。またSPSDを用いれば、k-LUTの入出力が矛盾していないかどうか、あるいはD-FlipFlopの入出力が矛盾していないかどうかを、非常に簡単に判断することができる。さらにSPSDの特殊な形として、D-FlipFlopの出力関数を表現するSSD (Sums of States to be Distinguished)と、入力部などの内部状態に依存しない論理関数を表現するSPD (Sums of Products to be Distinguished)とを提案する。実際にはSSDは、Kohaviが[4]で提案した分割という概念のうち、ブロック数が2のものと等価である。またSPDは、山下らが[12]で提案したSPFDsを、積和形表現に拡張したものである。

次章ではまずSPDを、状態に依存しない回路すなわち組合せ回路上で定義し、その性質およびk-LUTとの関係について述べる。3章ではSPSDを順序回路上で定義し、SPDおよびSSDとの関係、さらにそれらの性質およびD-FlipFlopとの関係について述べる。4章では、SPSD式を用いて順序回路をLUT型FPGA上で設計する手法について述べる。

2 組合せ回路とSPD式

2.1 SPDとSPD式

不完全指定n変数論理関数fが、積項集合$P \subset \{0,1,-\}^n$によって$f: P \to \{0,1,-\}$という形で与えられたとする。どの積項にも含まれないビット組合せについては関数値は$-$ (ドントケア)とし、複数の積項に含まれるビット組合せについては関数値は矛盾しない(0と1の両方にはならない)ものとする。このときfの**SPD** (Sums of Products to be Distinguished)を以下のように記す。

$$\sum_{p \in P, f(p)=0} p \quad \bowtie \quad \sum_{p \in P, f(p)=1} p$$

すなわちSPDは、関数値が0であるような積項の論理和と、関数値が1であるような積項の論理和とを、$\bowtie$の左右にそれぞれ並べたものである。例えば

	$x_1x_2x_3x_4x_5$	f_1
p_1	0 0 − 0 −	0
p_2	0 1 − − −	1
p_3	1 0 1 1 −	1
p_4	1 1 0 − 0	0
p_5	1 − 0 0 −	0
p_6	− 0 − 1 1	1

で与えられる5変数不完全指定論理関数f_1のSPDは、$p_1+p_4+p_5 \bowtie p_2+p_3+p_6$となる。

あるk-LUTの入力のSPDがそれぞれ$E_1, E_2, \cdots, E_k$であり、出力のSPDがFであるとき、これを

$$E_1 \vee E_2 \vee \cdots \vee E_k \geq F$$

と記し、**SPD式** (SPD Expression)と呼ぶ。また、このSPD式の左辺のように、SPDを$\vee$で繋いだものを、**$\vee$-SPD** (Union of SPDs)と呼ぶ。例えば上記の論理関数f_1を

	$x_3x_4x_5$	f_2
p_1	− 0 −	0
p_2	− − −	−
p_3	1 1 −	1
p_4	0 − 0	0
p_5	0 0 −	0
p_6	− 1 1	1

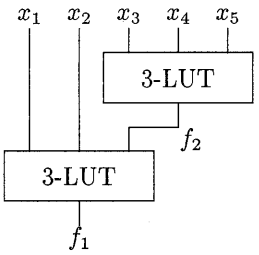

図 1: f_1 の 3-LUT での実現

という論理関数 f_2 を媒介にして、図 1のように 2 個の 3-LUT で実現した場合を考えると、右上の 3-LUT は

$$p_4{+}p_5 \bowtie p_3 \vee p_1{+}p_5 \bowtie p_3{+}p_6$$
$$\vee\ p_4 \bowtie p_6 \geq p_1{+}p_4{+}p_5 \bowtie p_3{+}p_6$$

という SPD 式で、左下の 3-LUT は

	$x_1x_2f_2$	f_1
p_1	0 0 0	0
p_2	0 1 −	1
p_3	1 0 1	1
p_4	1 1 0	0
p_5	1 − 0	0
p_6	− 0 1	1

という論理関数すなわち

$$p_1{+}p_2 \bowtie p_3{+}p_4{+}p_5$$
$$\vee\ p_1{+}p_3{+}p_6 \bowtie p_2{+}p_4$$
$$\vee\ p_1{+}p_4{+}p_5 \bowtie p_3{+}p_6$$
$$\geq p_1{+}p_4{+}p_5 \bowtie p_2{+}p_3{+}p_6$$

という SPD 式で表される。なお、ここでは外部入力も SPD で表しており、例えば入力 x_1 は $p_1{+}p_2 \bowtie p_3{+}p_4{+}p_5$ となる。

2.2 SPD 式の性質

k-LUT での実現という点から見ると、SPD 式の $\vee$ と $\geq$ には

$A \geq A$
$A \geq B$ ならば $A \vee C \geq B$
$A \vee B \geq C$ ならば $B \vee A \geq C$

という性質が存在する。これに加え、複数の k-LUT をまとめて

$A \geq B$ かつ $B \geq C$ ならば $A \geq C$
$A \geq B$ かつ $A \geq C$ ならば $A \geq B \vee C$

と記してもよいことにすると、SPD 式の $\geq$ は $\vee$ に関して閉じた半順序関係を成すと言える。さらに、$A \geq B$ かつ $B \geq A$ を $A = B$ という等価関係で記すと、$\vee$ と $=$ には

$$A = A \vee A$$
$$A \vee B = B \vee A$$

が常に成立すると言える。

次に SPD 式の $\bowtie$ について考えると、やはり k-LUT での実現という点から

$A \bowtie B \geq C$ ならば $B \bowtie A \geq C$
$A \bowtie B \geq C$ ならば $A \bowtie B{+}D \geq C$
$A \bowtie B \vee A \bowtie C \geq D$ ならば
$A \bowtie B{+}C \geq D$

という性質が存在する。これを、等価関係 $=$ を用いて書き直すと

$$A \bowtie B = B \bowtie A$$
$$A \bowtie B{+}C = A \bowtie B \vee A \bowtie C$$

が常に成立すると言える。

最後の性質は特に重要である。この性質によって $\vee$-SPD を、それと等価で $+$ を含まない形に変換できるのである。しかもそのような $+$ を含まない形は、SPD の並べ換えと $\bowtie$ の左右の入れ換えを同一視すれば、任意の $\vee$-SPD に対して一意に定まるため、$\vee$-SPD の標準形として用いることが可能である。すなわち、この $\vee$-SPD 標準形への変換によって、任意の 2 つの $\vee$-SPD の間に半順序関係 $\geq$ や等価関係 $=$ が成立するかどうか、判断可能になるのである。例として

$$E_1 = p_1{+}p_2 \bowtie p_3{+}p_4{+}p_5$$
$$E_2 = p_1{+}p_3{+}p_6 \bowtie p_2{+}p_4$$
$$E_4 = p_1{+}p_5 \bowtie p_3{+}p_6$$
$$F = p_1{+}p_4{+}p_5 \bowtie p_2{+}p_3{+}p_6$$

という 4 つの SPD に対し、$E_1 \vee E_2 \vee E_4$ と F の間の半順序関係について考えてみよう。

$E_1 \vee E_2 \vee E_4$ を標準形に変換すると

$$\begin{aligned} &E_1 \vee E_2 \vee E_4 \\ &\quad = p_1 \bowtie p_2 \vee p_1 \bowtie p_3 \vee p_1 \bowtie p_4 \vee \\ &\qquad p_1 \bowtie p_5 \vee p_1 \bowtie p_6 \vee p_2 \bowtie p_3 \vee \\ &\qquad p_2 \bowtie p_4 \vee p_2 \bowtie p_5 \vee p_2 \bowtie p_6 \vee \\ &\qquad p_3 \bowtie p_4 \vee p_3 \bowtie p_5 \vee p_4 \bowtie p_6 \vee \\ &\qquad p_5 \bowtie p_6 \end{aligned}$$

となり、F を標準形に変換すると

$$F = p_1 \bowtie p_2 \vee p_1 \bowtie p_3 \vee p_1 \bowtie p_6 \vee p_2 \bowtie p_4 \vee p_2 \bowtie p_5 \vee p_3 \bowtie p_4 \vee p_3 \bowtie p_5 \vee p_4 \bowtie p_6 \vee p_5 \bowtie p_6$$

となる。これらを比較すると、F の標準形中の SPD は全て $E_1 \vee E_2 \vee E_4$ の標準形に含まれていることから、$E_1 \vee E_2 \vee E_4 \geq F$ が成立することがわかる。逆に $E_1 \vee E_2 \vee E_4$ の標準形中の SPD のうち 4 つは F の標準形に含まれていないことから、$F \not\geq E_1 \vee E_2 \vee E_4$ である。

なお、$E_1 \vee E_2 \vee E_4 \geq F$ すなわち

$$p_1+p_2 \bowtie p_3+p_4+p_5 \vee p_1+p_3+p_6 \bowtie p_2+p_4 \vee p_1+p_5 \bowtie p_3+p_6 \geq p_1+p_4+p_5 \bowtie p_2+p_3+p_6$$

を、前節の論理関数 f_1 に当てはめると、x_1, x_2, x_4 を入力とする 3-LUT によって

	$x_1x_2x_4$	f_1
p_1	0 0 0	0
p_2	0 1 –	1
p_3	1 0 1	1
p_4	1 1 –	0
p_5	1 – 0	0
p_6	– 0 1	1

という形で f_1 が実現可能である、という結論が導き出せる。

3 順序回路と SPSD 式

3.1 SPSD と SPSD 式

不完全指定 Mealy 型順序機械 M の遷移関数 λ が、状態集合 S と積項集合 $P \subset \{0,1,-\}^n$ によって λ: $S \times P \rightarrow S \cup \{\perp\}$ という形で与えられたとする。さらに、順序機械 M を実現する論理回路のある結線 (入出力を含む) の論理関数 δ が δ: $S \times P \rightarrow \{0,1,-\}$ という形で与えられたとする。ただし、どの積項にも含まれないビット組合せについては関数値は $\perp$ あるいは – とし、複数の積項に含まれるビット組合せについては関数値は矛盾しないものとする。このとき δ の **SPSD** (Sums of Products and States to be Distinguished) を以下のように記す。

$$\sum_{s \in S, p \in P, \delta(sp)=0} sp \quad \bowtie \quad \sum_{s \in S, p \in P, \delta(sp)=1} sp$$

すなわち SPSD は、関数値が 0 となるような状態と積項の積の論理和と、関数値が 1 となるような状態と積項の積の論理和とを、$\bowtie$ の左右にそれぞれ並べたものである。例えば

λ, y_1y_2	p_1 00–	p_2 010	p_3 1–0	p_4 111
s_1	s_1,0–	s_3,––	s_5,00	s_2,11
s_2	s_4,0–	$\perp$,––	$\perp$,––	s_4,1–
s_3	s_5,––	s_4,–0	s_1,––	$\perp$,––
s_4	$\perp$,––	s_3,11	s_2,–0	s_1,–1
s_5	$\perp$,––	s_5,1–	s_4,10	$\perp$,––

で与えられる順序機械の外部出力 y_2 すなわち

y_2	p_1	p_2	p_3	p_4
s_1	–	–	0	1
s_2	–	–	–	–
s_3	–	0	–	–
s_4	–	1	0	1
s_5	–	–	0	–

という論理関数 y_2 の SPSD は

$$s_1p_3+s_3p_2+s_4p_3+s_5p_3 \bowtie s_1p_4+s_4p_2+s_4p_4$$

となる。

論理関数 δ が P に依存しない場合、すなわち δ が δ: $S \rightarrow \{0,1,-\}$ という形で与えられうる場合には、δ の SPSD の特殊な形として、δ の **SSD** (Sums of States to be Distinguished) を以下のように記す。

$$\sum_{s \in S, \delta(s)=0} s \quad \bowtie \quad \sum_{s \in S, \delta(s)=1} s$$

また、論理関数 δ が S に依存しない場合、すなわち δ が δ: $P \rightarrow \{0,1,-\}$ という形で与えられうる場合には、δ の SPSD の特殊な形として、δ の **SPD** (Sums of Products to be Distinguished) を以下のように記す。

$$\sum_{p \in P, \delta(p)=0} p \quad \bowtie \quad \sum_{p \in P, \delta(p)=1} p$$

同様に SPSD 中でも、$\sum_{p \in P} sp$ を s と、$\sum_{s \in S} sp$ を p と略記する。

ある k-LUT の入力の SPSD がそれぞれ E_1, E_2, $\cdots$, E_k であり、出力の SPSD が F であるとき、これを

$$E_1 \vee E_2 \vee \cdots \vee E_k \geq F$$

と記し、**SPSD 式** (SPSD Expression) と呼ぶ。また、この SPSD 式の左辺のように、SPSD を $\vee$ で繋いだものを **$\vee$-SPSD** (Union of SPSDs) と呼ぶ。$\vee$-SPSD のうち、含まれる SPSD が全て SSD であるものを、特に **$\vee$-SSD** と呼ぶ。

ある D-FlipFlop の入力の SPSD が E であり、出力の SSD が F であるとき、これを遷移関数 λ を用いて

$$\lambda(E) \geq F$$

と記し、**SPSD 遷移式** (SPSD Transition) と呼ぶ。例えば先の順序機械の外部出力 y_2 を

δ_1	p_1	p_2	p_3	p_4
s_1	1	0	1	1
s_2	1	–	1	1
s_3	1	1	1	1
s_4	1	0	1	1
s_5	1	–	1	1

という論理関数 δ_1 を媒介にし

$$s_1 \rightarrow 1,\ s_2 \rightarrow 1,\ s_3 \rightarrow 0,\ s_4 \rightarrow 1,\ s_5 \rightarrow -$$

という状態割当をおこなった D-FlipFlop と 2 個の 3-LUT を用いて、図 2のように実現した

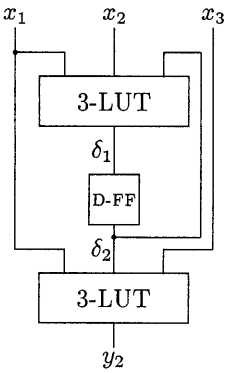

図 2: y_2 の 3-LUT での実現

場合を考えると、図 2の上の 3-LUT は

$$\begin{aligned} &p_1{+}p_2 \bowtie p_3{+}p_4 \vee p_1 \bowtie p_2{+}p_4 \\ &\vee\ s_3 \bowtie s_1{+}s_2{+}s_4 \\ &\geq s_1p_2{+}s_4p_2 \bowtie s_3p_2{+}p_1{+}p_3{+}p_4 \end{aligned}$$

という SPSD 式で、真ん中の D-FlipFlop は

$$\begin{aligned} &\lambda(s_1p_2{+}s_4p_2 \bowtie s_3p_2{+}p_1{+}p_3{+}p_4) \\ &\geq s_3 \bowtie s_1{+}s_2{+}s_4 \end{aligned}$$

という SPSD 遷移式で、下の 3-LUT は

$$\begin{aligned} &p_1{+}p_2 \bowtie p_3{+}p_4 \vee s_3 \bowtie s_1{+}s_2{+}s_4 \\ &\vee\ p_2{+}p_3 \bowtie p_4 \\ &\geq s_1p_3{+}s_3p_2{+}s_4p_3{+}s_5p_3 \\ &\qquad\qquad \bowtie s_1p_4{+}s_4p_2{+}s_4p_4 \end{aligned}$$

という SPSD 式で表される。

3.2 SPSD 式と SPSD 遷移式の性質

2.2節での議論と同様、SPSD 式においても $\geq$ は $\vee$ に関して閉じた半順序関係を成す。また、$A{\geq}B$ かつ $B{\geq}A$ を $A{=}B$ と記すと

$$\begin{aligned} &A = A \vee A \\ &A \vee B = B \vee A \geq A \\ &A \bowtie B = B \bowtie A \\ &A \bowtie B{+}C = A \bowtie B \vee A \bowtie C \end{aligned}$$

が常に成立すると言える。すなわち $\vee$-SPD と同様に、$\vee$-SPSD に対しても $\vee$-SPSD 標準形を考えることができる。これに加え、SSD と SPSD との間あるいは SPD と SPSD との間で、以下の性質が存在する。

$$\begin{aligned} sp \bowtie s'p' &\leq s \bowtie s' \\ sp \bowtie s'p' &\leq p \bowtie p' \end{aligned}$$

$s{=}\sum_{p \in P} sp$ および $p{=}\sum_{s \in S} sp$ であることから、自明の性質だといえる。ただし $s{=}s'$ の場合には $sp \bowtie s'p' \leq s \bowtie s'$ は成立しないことから、$sp \bowtie sp'$ に対し $p \bowtie p'$ を**必須な SPD** と呼ぶことにする。同様に、$p{=}p'$ の場合には $sp \bowtie s'p' \leq p \bowtie p'$ は成立しないことから、$sp \bowtie s'p$ に対し $s \bowtie s'$ を**必須な SSD** と呼ぶことにする。

SPSD 遷移式 $\lambda(E) \geq F$ の E, F の論理関数を、δ_E: $S{\times}P \rightarrow \{0,1,-\}$, δ_F: $S \rightarrow \{0,1,-\}$ とする (ただし $\delta_F(\perp){=}-$ とみなす)。このとき δ_E と δ_F には以下の関係が成立する。

$$\begin{aligned} &{}^\forall s{\in}S,\ {}^\forall p{\in}P \text{ に対し} \\ &\quad \delta_F(\lambda(sp)){=}0 \text{ ならば } \delta_E(sp){=}0 \\ &\quad \delta_F(\lambda(sp)){=}1 \text{ ならば } \delta_E(sp){=}1 \end{aligned}$$

この関係を、SPSD 遷移式の入出力関係と呼ぶ。なお $\lambda(E) \geq F$ が特に

$$\begin{aligned} &{}^\forall s{\in}S,\ {}^\forall p{\in}P \text{ に対し} \\ &\quad \delta_F(\lambda(sp)){=}0 \Leftrightarrow \delta_E(sp){=}0 \\ &\quad \delta_F(\lambda(sp)){=}1 \Leftrightarrow \delta_E(sp){=}1 \\ &\quad \delta_F(\lambda(sp)){=}- \Leftrightarrow \delta_E(sp){=}- \end{aligned}$$

を満たす時、$\lambda(E){=}F$ あるいは $E{=}\lambda^{-1}(F)$ と記すことにする。

SPSD 遷移式の入出力関係が意味するところを、図 2中の D-FlipFlop を例に考えてみよう。この D-FlipFlop の状態割当は

	δ_2
s_1	1
s_2	1
s_3	0
s_4	1
s_5	–

であることから、このD-FlipFlopの入力には、次状態がs_1ならば1、s_2ならば1、s_3ならば0、s_4ならば1が入力されなければならない。言い換えると、論理関数δ_1は

$\lambda(sp)=s_1$ ならば $\delta_1(sp)=1$
$\lambda(sp)=s_2$ ならば $\delta_1(sp)=1$
$\lambda(sp)=s_3$ ならば $\delta_1(sp)=0$
$\lambda(sp)=s_4$ ならば $\delta_1(sp)=1$

を満たさなければならない。これと論理関数δ_2とを見比べると、上記の条件は

$\delta_2(\lambda(sp))=1$ ならば $\delta_1(sp)=1$
$\delta_2(\lambda(sp))=0$ ならば $\delta_1(sp)=0$

という形に書き直すことができる。これがSPSD遷移式の入出力関係の意味である。ちなみに、このD-FlipFlopの入力が満たすべき条件、すなわち$\lambda^{-1}(s_3 \bowtie s_1+s_2+s_4)$は

	p_1	p_2	p_3	p_4
s_1	1	0	–	1
s_2	1	–	–	1
s_3	–	1	1	–
s_4	–	0	1	1
s_5	–	–	1	–

であるが、前節にもある通りδ_1は、確かにこれを満たしていると言える。

4 SPSD式を用いた順序回路設計

4.1 基本アルゴリズム

不完全指定Mealy型順序機械の外部入出力と遷移関係が与えられた時、この順序機械をLUT型FPGA上で実現することを考える。基本的には、実現すべき関数のSPSDをFとするとき、$E_1 \vee E_2 \vee \cdots \vee E_k \geq F$を満たす$E_1$, E_2, $\cdots$, E_kのSPSDを求める、という作業がk-LUTの合成にあたる。ここで、E_iがSSDであればD-FlipFlopで実現し、E_iが外部入力に含まれるSPDであればその外部入力で置き換える。それ以外ならば、E_1をさらにあらたなk-LUTで合成することになる。また、SSDをD-FlipFlopで実現した場合、そのD-FlipFlopの入力のSPSDは、やはりあらたなk-LUTで合成することになる。このように、外部出力のSPSDから始めて、実現すべきSPSDが全て外部入力、D-FlipFlopあるいはk-LUTによって実現できたとき、与えられた順序機械はLUT型FPGA上で実現されたことになる。

この手順をおおまかなアルゴリズムで示すと、以下のようになる。

Step 1. 実現すべき順序機械の外部出力のSPSDの集合を$\mathcal{S}$とおく。

Step 2. $\mathcal{S}$が空集合ならば終了。

Step 3. $\mathcal{S}$からSPSDを1つ取り出し、Fとおく。FがSSDでなければStep 7へ。

Step 4. Fがすでに合成したD-FlipFlopに含まれるならば、Fをそれで置き換えてStep 2へ。

Step 5. Fを出力とするD-FlipFlopを合成し、その入力すなわち$\lambda^{-1}(F)$を$\mathcal{S}$に加える。

Step 6. Fがすでに合成したD-FlipFlopの出力を含んでいるならば、それをStep 5で合成したD-FlipFlopで置き換える。Step 3へ。

Step 7. Fが外部入力もしくはすでに合成したk-LUTの出力に含まれるならば、Fをそれで置き換えてStep 2へ。

Step 8. $E_1 \vee E_2 \vee \cdots \vee E_k \geq F$を満たす$E_1$, E_2, $\cdots$, E_kを求める。ただしE_1, E_2, $\cdots$, E_{k-1}はSSDもしくはSPDとし、E_kはSPSDとする。

Step 9. E_1, E_2, $\cdots$, E_kを入力とするk-LUTでFを合成する。E_1, E_2, $\cdots$, E_kを$\mathcal{S}$に加えて、Step 3へ。

このアルゴリズムを実際におこなう場合には、Step 8で$E_1, E_2, \cdots, E_k$を導出する際に、必須なSPDを優先的に導出する方が良い結果が得られるようである。さらに、SPDを導出する際には、それを含む外部入力で置き換えることを考え、最初から外部入力のSPDを導出する方が良い結果が得られるようである。

例として、3.1節の順序機械の外部出力y_1を、3-LUT型FPGAで合成することを考えてみよう。y_1のSPSDをFとおくと、$F=s_1p_1+s_1p_3+s_2p_1 \bowtie s_1p_4+s_2p_4+s_4p_2+s_5p_2+s_5p_3$より、$F$に必須なSPDは$p_1+p_3 \bowtie p_4$である。このSPDを含む外部入力を考えると、$p_1 \bowtie p_4$は入力$x_1$すなわち$p_1+p_2 \bowtie p_3+p_4$で、$p_3 \bowtie p_4$は入力$x_3$すなわち$p_2+p_3 \bowtie p_4$で置き換えることができる。これより、$F$を3-LUTで合成するならば$p_1+p_2 \bowtie p_3+p_4 \vee p_2+p_3 \bowtie p_4 \vee F' \geq F$という形で合成するのが良い、と考えられる。ここで$F'$は$F' \geq s_1p_1+s_2p_1 \bowtie s_4p_2+s_5p_2 \vee s_1p_3 \bowtie s_5p_3$を満たさねばならないので、例えば$F'=s_1p_1+s_1p_3+s_2p_1 \bowtie s_4p_2+s_5p_2+s_5p_3$として、さらに$F'$を3-LUTで合成することを考える。$F'$には必須なSPDはなく、$s_1 \bowtie s_5$が必須なSSDである。$F'$中で$s_1 \bowtie s_5$に含まれない部

分は、$p_1 \bowtie p_2$ あるいは $p_3 \bowtie p_2$ に含まれる。これらを含む外部入力を考えると、$p_1 \bowtie p_2$ は入力 x_2 すなわち $p_1 \bowtie p_2+p_4$ で、$p_3 \bowtie p_2$ は入力 x_1 すなわち $p_1+p_2 \bowtie p_3+p_4$ で置き換えることができる。これより F' は $s_1 \bowtie s_5 \vee p_1 \bowtie p_2+p_4 \vee p_1+p_2 \bowtie p_3+p_4 \geq F'$ という形で合成できるといえる。$s_1 \bowtie s_5$ は D-FlipFlop で合成することになるので、アルゴリズムではさらに $\lambda^{-1}(s_1 \bowtie s_5)$ の 3-LUT 合成を求めていくことになる。

このようにアルゴリズムを繰り返していった結果、y_1 および y_2 を出力する回路は、図 3のように合成される。なお図中、上の D-FlipFlop

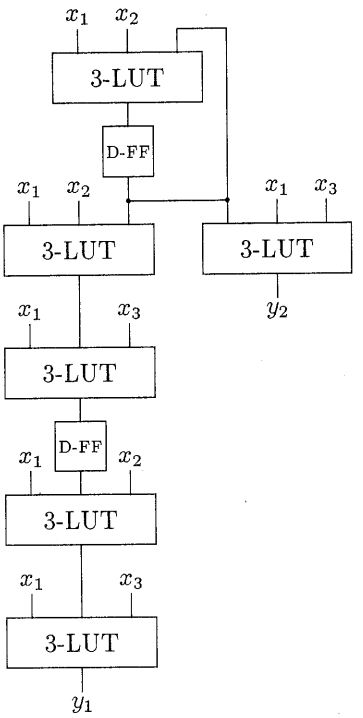

図 3: y_1, y_2 の 3-LUT での実現

は $s_3 \bowtie s_1+s_2+s_4$、下の D-FlipFlop は $s_1 \bowtie s_5$ となっている。すなわち、状態割当は

$$s_1 \rightarrow 10,\ s_2 \rightarrow 1-,\ s_3 \rightarrow 0-,\ s_4 \rightarrow 1-,\ s_5 \rightarrow -1$$

となっており、10 は状態 s_1 と s_2 と s_4 を、11 は s_2 と s_4 と s_5 を、01 は s_3 と s_5 を、同時に表すようになっている。

4.2 実験結果

前節のアルゴリズムに従って、MCNC ベンチマークの順序機械 bbara (KISS2 フォーマットで状態数 10、入力数 4、出力数 2、積項×状態数 60) を 5-LUT 型 FPGA で合成したところ、9 個の 5-LUT と 6 個の D-FlipFlop で合成できた。合成結果を図 4に示す。なお、図中の D-FlipFlop は

	D_1	D_2	D_3	D_4	D_5	D_6
st0	0	0	0	0	0	0
st1	0	0	1	0	–	–
st2	0	1	–	0	0	0
st3	1	–	–	0	0	0
st4	0	0	0	0	0	1
st5	0	0	0	0	1	–
st6	0	0	0	1	–	–
st7	0	0	0	0	0	0
st8	0	0	0	0	0	0
st9	0	0	0	0	0	0

という状態割当になっていた。すなわち、st0, st7, st8, st9 の 4 状態は、実は等価な状態であったことがわかった。

図 4の回路を Xilinx XC 3000 シリーズで実現した場合、必要な CLB (Configurable Logic Blocks) の数は、単純に 5-LUT を 1 個ずつ CLB で実現した場合には 9 個、図中左上の 4 入力の 5-LUT とそのすぐ下の 4 入力の 5-LUT を 1 個の CLB にまとめると 8 個となる。これまでの報告では bbara を XC 3000 シリーズ上で実現するのに、[9] の one-hot では 15 個の CLB、[10] の sis-pga では 11 個の CLB、[11] の LAX では 14 個の CLB が必要であったことから、本稿の結果は非常に画期的であるといえる。

蛇足ながら、図 4の回路を手作業で検証した結果、右上 2 つ目の 5-LUT とそのすぐ下の 5-LUT に D_3 から来ている結線は、D_5 と D_6 を

	D_1	D_2	D_3	D_4	D_5	D_6
st0	0	0	0	0	0	0
st1	0	0	1	0	0	0
st2	0	1	–	0	0	0
st3	1	–	–	0	0	0
st4	0	0	0	0	0	1
st5	0	0	0	0	1	–
st6	0	0	0	1	–	–
st7	0	0	0	0	0	0
st8	0	0	0	0	0	0
st9	0	0	0	0	0	0

という形にすれば、不要となることがわかった。すなわちこれらの 5-LUT を 1 個の CLB にまとめられるので、bbara の CLB 数は 7 個に減らすことができると考えられる。このような状態割当を優先的に導出するアルゴリズムについては、今後の課題としたい。

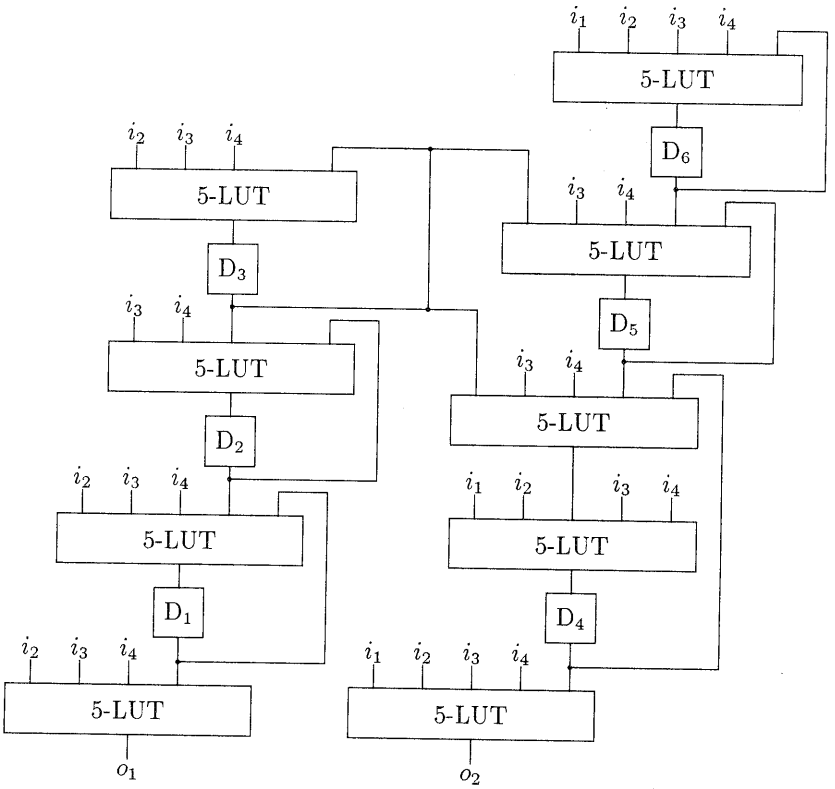

図 4: bbara の 5-LUT 実現

5 おわりに

LUT 型 FPGA 上での状態割当を記述し、各結線の表す論理関数を記述するための表現法として、SPSD を提案した。また、SPSD を用いて、LUT 型 FPGA 上で順序回路を合成する手法について述べた。さらに本稿の手法を MCNC ベンチマークの順序機械に適用し、有効性を検証した。今後、本稿の手法をさらに押し進めて、CLB 数の最小化を目指した順序回路の合成手法を完成したい。

参考文献

[1] J. Hartmanis: Symbolic Analysis of a Decomposition of Information Processing Machines. Information and Control, vol. 3 (June 1960), pp. 154-178.

[2] A. Gill: Cascaded Finite-state Machines, IRE Transactions on Electronic Computers. vol. EC-10 (September 1961), pp. 549-562.

[3] M. Yoeli: Cascade-Parallel Decompositions of Sequential Machines. IEEE Transactions on Electronic Computers, vol. EC-12 (June 1963), pp. 322-324.

[4] Z. Kohavi: Secondary State Assignment for Sequential Machines, IEEE Transactions on Electronic Computers, vol. EC-13 (June 1964), pp. 193-203.

[5] G. De Micheli, R. K. Brayton, and A. Sangiovanni-Vincentelli: Optimal State Assignment of Finite State Machines, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. CAD-4 (July 1985), pp. 269-285.

[6] S. Devadas, H-K. Ma, A. R. Newton, and A. Sangiovanni-Vincentelli: MUSTANG: State Assignment of Finite State Machines Targeting Multi-level Logic Implementations. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 7 (December 1988), pp. 1290-1300.

[7] J. L. Huertas and J. M. Quintana: Efficiency of State Assignment Methods for PLA-based Sequential Circuits. IEE Proceedings Part E, vol. 136 (July 1989), pp. 247-253.

[8] S. Devadas and A. R. Newton: Decomposition and Factorization of Sequential Finite State Machines, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 8 (November 1989), pp. 1206-1217.

[9] M. Schlag, P. K. Chan, and J. Kong: Empirical Evaluation of Multilevel Logic Minimization Tools for a Lookup-Table-Based Field-Programmable Gate Array Technology, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 12 (May 1993), pp. 713-722.

[10] R. Murgai, R. K. Brayton, and A. Sangiovanni-Vincentelli: Sequential Synthesis for Table Look Up Programmable Gate Arrays. Proceedings of 30th ACM/IEEE Design Automation Conference (June 1993), pp. 224-229.

[11] L. Burgun, N. Dictus, E. Prado Lopes, and C. Sarwary: A Unified Approach for FSM Synthesis on FPGA Architectures, IEEE Proceedings of the 20th EUROMICRO Conference (September 1994), pp. 660-668.

[12] S. Yamashita, H. Sawada, and A. Nagoya: A New Method to Express Functional Permissibilities for LUT Based FPGAs and Its Applications, IEEE Proceedings of ICCAD'96 (November 1996), pp. 254-261.