Haar spectra-based entropy approach to quasi-minimisation of FBDDs

C.-H. Chang
B.J. Falkowski

Abstract: An information theoretic approach, to exploit the additional degree of freedom associated with don't cares of incompletely specified Boolean functions, is applied to quasi-minimisation of free binary decision diagrams (FBDDs). The concept of entropy and equivocation is formulated through paired Haar spectra of incompletely specified Boolean functions. The likelihood metric expressed in terms of selected spectral coefficients is used to simplify the process of don't care allocation. The approach is general and can be extended to other combinatorial decision problems.

1 Introduction

Spectral techniques have been widely applied to Boolean function classification, disjoint decomposition, parallel and serial linear decomposition, spectral translation synthesis (extraction of linear pre- and post-filters), multiplexer synthesis, prime implicant extraction, threshold logic synthesis, state assignment, testing and evaluation of logic complexity [1–8]. There are at least two transforms, based on square-wave like functions, that are well suited to Boolean functions: Walsh and Haar transforms. The Walsh functions are global like Fourier functions, and consist of a set of irregular rectangular waveforms with only two amplitude values +1 and –1 [1, 2, 6, 9]. All but two basis functions in the Haar transform consist of a square wave pulse located on an otherwise zero amplitude interval. When applied to logic design, an unnormalised Haar transform [3, 4, 10–14] is normally used. Computation of the fast Haar transform (FHT) requires order N (N is the number of spectral coefficients) additions and subtractions, which makes it much faster than the fast Walsh transform (FWT) [1, 2, 8–10]. Hardware-based fast Haar chips have also been developed [10]. Due to its low computing requirements, the Haar transform has been used mainly for pattern recognition and image processing [7–9]. Such a transform is also well suited in communication technology for data coding, multiplexing and digital filtering [7, 8]. The advantages of computational and memory requirements of the Haar transform make it of great interest to VLSI designers as well [1, 3, 4].

The local property of the Haar transform makes it useful in those applications in computer-aided design systems, where there are Boolean functions of many variables that have most of their values grouped locally. Such local functions frequently occur in logic design, and can be extremely well described by a few spectral coefficients from the Haar transform, while the application of the global Walsh transform would be quite cumbersome in such cases, and the locally grouped minterms would be spread throughout the whole Walsh spectrum. In most engineering design problems, incompletely specified functions have to be dealt with. The don't care sets derived from circuit structures represent an additional degree of freedom, and their effective utilisation often results in highly economical circuits [15–17]. To better deal with the mentioned cases with incompletely specified Boolean functions, the idea of a paired Haar transform was introduced [11, 13]. In the paired Haar transform, all the information about true and don't care minterms is kept separate, by what is available in different stages of CAD process. This paper further explores the properties of the paired Haar transform, and derives an elegant polynomial Haar expansion for incompletely specified Boolean functions. This polynomial expansion is used in conjunction with probability theory to generate heuristic solutions to classical logic minimisation problems for incompletely specified Boolean functions.

Finding the minimal realisations for logic functions is usually associated with the problem of optimising their reduced representations. For large digital circuits, free binary decision diagrams (FBDDs) [18, 19] are a more succinct representation than the cubical representation of a disjunctive sum-of-products expression for a given function in two levels. A special subset of FBDD is the ordered binary decision diagram (OBDD) [12, 19–25], which is a canonical representation of a Boolean function with a given ordering of variables. The main disadvantage of OBDDs is their sensitivity to the ordering of input variables [20–25]. The best published exact algorithm for the computation of an optimal variable ordering for the OBDD is a dynamic programming method by Friedman and Supowit [21] with a time complexity of $O(n^2 3^n)$. There exist important Boolean functions such as the FHS-function, integer multiplication, hidden weighted bit function or indirect storage access function for which the OBDD representations are exponential for all possible variable orderings. These functions, however, can be represented by polynomial or even quadratic size FBDDs [18, 19]. As a natural extension of the OBDD, the FBDD has inher-

IEE Proceedings online no. 19990247

DOI: 10.1049/ip-cdt: 19990247

Paper first received 9th January and in revised form 8th July 1998

C.-H. Chang is with the Electronics Design Centre, French Singapore Institute, Nanyang Polytechnic, 180 Ang Mo Kio Ave 8, Singapore 569830

B.J. Falkowski is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Blk S1, Nanyang Avenue, Singapore 639798

ited many useful properties of the OBDD that allow basic Boolean manipulations to be performed as efficiently as for the OBDD. Besides being more succinct, the reduced FBDD of a fixed complete type is also canonical [19]. Obviously, it is NP-hard to transform the general circuit topology with an NP-complete satisfiability test to an optimal OBDD [20]. This also holds for the computation of the more general form of the minimal FBDD. In this paper, a unified entropy approach operated on the paired Haar spectrum to the heuristic optimisations of the FBDD, with effective utilisation of the don't care sets for incompletely specified Boolean functions, has been developed.

The concept of entropy [17, 26, 27] in probability theory arose from an attempt to develop a theoretical model for the transmission of discrete information in noisy channels. Since the introduction of Shannon's theorem on channels with noise, in terms of a quantity known as equivocation [26, 27], the exposition of the theory of entropy and equivocation has appeared in various disciplines. In this paper, we exploit the general nature and theoretical significance of this mathematical apparatus to bridge the gap between communication theory and combinatorial decision problems. The concept of entropy and equivocation is applied to the generation of quasi-optimal FBDD of incompletely specified Boolean functions. We show that entropy and equivocation can be elegantly formulated by a paired Haar spectrum. Moreover, the unified and systematic entropy approach that has evolved from the presented theorems to this general decision problem is intuitively appealing. There is no need to generate an initial BDD with an arbitrary variable ordering, followed by improving the variable ordering with local search [22, 25], or simulated annealing [23, 24], in two steps. The algorithm for the FBDD minimisation can be used for multiplexer universal logic module network synthesis in tree type realisation [5, 6], by treating each vertex as a set of control variables with multiple children. The extension of the FBDD minimisation algorithm to multiplexer synthesis permits mixed control variables within each level, if it leads to early termination of more paths with constants or single variables.

2 Basic definitions

An n-variable Boolean function $F(x_1, x_2, ..., x_n)$ is a mapping $F:\{0, 1\}^n \rightarrow \{0, 1, -\}^k$, where the symbol '–' means a non specified value (a don't care), and k is the number of outputs. A Boolean function is completely specified if all its outputs contain only the set $\{0, 1\}$, and incompletely specified if any of its outputs is non specified.

An n-bit string is a vertex of an object called a 0-cube. An n-variable Boolean function is represented as an n-dimensional space (n-hypercube) in which each vertex represents a minterm. A collection of 2^i, $i \in \{0, 1, ..., n\}$ adjacent minterms is called an i-cube. A cube can be represented by an n-tuple string of 0, 1, and –, where 0 corresponds to the complemented value of the variable, 1 to the affirmative value, and – to the vacuous variable in the cube. The cardinality of an i-cube is 2^i, where i is the number of vacuous variables of the cube. The ON, OFF and DC cubes are cubes corresponding to the product terms of ON, OFF and DC minterms, respectively. The sets of ON, OFF and DC cubes are called ON, OFF and DC arrays, respectively. A cube is an implicant of a function F if its intersection with the OFF array is empty. A prime implicant is an implicant that has the largest cardinality, such that removal of any of its literals results in a cube that is not an implicant.

A binary decision diagram (BDD) [12, 18–25] is a rooted directed acyclic graph representation with vertex set V and edge set E. The vertex set consists of two types of vertices, the nonterminal and terminal vertices. A nonterminal vertex $v \in V$ has as attributes an index, denoted by $index(v)$, to identify an input variable of a function, and two children (or successors), $low(v)$ and $high(v) \in V$. A terminal vertex (or terminus) $u \in V$ has no child and it has a value, denoted by $value(u)$. $value(u)$ = 0, 1 or 0.5 for the functional value of logical zero, one or don't care, respectively. The edge set consists of two types of edges. A 0-edge is a link from a node v to its low child $low(v)$, and a 1-edge is one that connects v to $high(v)$. A root is the topmost or the first non-terminal vertex in the BDD. A path from a vertex v_1 to a vertex v_2 is a set of vertices and edges traversed from v_1 to v_2. A free binary decision diagram (FBDD) [18, 19] is a BDD for which each variable of the function represented by it is encountered at most once along any path from the root to a terminal vertex. An ordered binary decision diagram (OBDD) [12, 19–23, 25] is a special subset of the FBDD in which the input variables in all paths appear in a fixed order, and there exists an index function for every nonterminal vertex $v \in V$ such that $index(low(v)) < index(v)$ and $index(high(v)) < index(v)$.

Property 2.1: A path with k vertices represents a $(n-k)$-cube, where $k = 1, 2, ..., n$, since an absent vertex corresponds to a vacuous variable in a product term, or '–' in a cube notation.

Definition 2.2: Let X be a finite space with elementary events X_i and the probability distribution $p(X_i)$ for $1 \leq i \leq n$ and $\Sigma_{i=1}^{n} p(X_i) = 1$. The entropy $H(X)$ of the finite space X is defined as [27]:

$$H(X) = -\sum_{i=1}^{n} p(X_i) \log_2 p(X_i) \qquad (1)$$

where the expression $p(X_i)\log_2 p(X_i)$ is taken to be 0 if $p(X_i) = 0$.

Entropy is a reasonably good measure of the amount of uncertainty associated with a given finite scheme. The entropy $H(X) = 0$ if and only if $p(X_i) = 1$ for only one value of i, and $p(X_i) = 0$ for all other values of i. This is an extreme situation where the outcome of an experiment can be predicted with complete certainty. For a fixed finite number of events n, maximum entropy occurs when each event X_i is equally likely, i.e. $p(X_i) = 1/n \ \forall \ i = 1, 2, ..., n$.

Definition 2.3: Let X and Y be two finite spaces with elementary events X_i, Y_j and their probability distributions $p(X_i)$ and $p(Y_j)$, respectively. $1 \leq i \leq n$, $1 \leq j \leq m$, $\Sigma_{i=1}^{n} p(X_i) = 1$ and $\Sigma_{j=1}^{m} p(Y_j) = 1$. X_i and Y_j may be dependent. The conditional entropy $H(Y \mid X_i)$ of space Y, based on the assumption that event X_i has occurred in space X, is given by [27]:

$$H(Y|X_i) = -\sum_{j=1}^{m} p(Y_j|X_i) \log_2 p(Y_j|X_i) \qquad (2)$$

Since the occurrence of each event X_i results in a specific value of $H(Y \mid X_i)$, the conditional entropy $H(Y \mid X_i)$ can be regarded as a random variable defined on the space X. The mathematical expected value of this random variable leads to the definition of equivocation.

Definition 2.4: The equivocation $H(Y \mid X)$ [27] is defined as the conditional entropy of the finite space Y averaged over the space X. Mathematically

$$H(Y|X) = \sum_{i=1}^{n} p(X_i)H(Y|X_i) \tag{3}$$

3 Polynomial Haar expansion and paired Haar spectrum

Definition 3.1: The unnormalised Haar transform T_N [1–4, 10–13] of order $N = 2^n$ can be defined recursively as

$$T_N = \begin{bmatrix} T_{\frac{N}{2}} \otimes [1 \quad 1] \\ I_{\frac{N}{2}} \otimes [1 \quad -1] \end{bmatrix} \text{ and } T_1 = 1 \tag{4}$$

where $I_{N/2}$ is an identity matrix of order $N/2$, and the symbol $\otimes$ denotes the right-hand Kronecker product.

For an n-variable Boolean function $F(x_1, x_2, ..., x_n)$, the Haar spectrum is given by $R = T_N \times F$, where R is Haar spectrum (a column vector of dimension $2^n \times 1$), and F is the R-coded vector of the Boolean function $F(X)$ [1, 2, 11–13]. In R-coding, the false minterms are coded as 0, the true minterms as 1, and the don't care (DC) minterms as 0.5.

Besides the first two Haar spectral coefficients r_{dc} (the so called *dc* coefficient corresponding to the *dc* function) and $r_0^{(0)}$, which are globally sensitive to $F(X)$, the remaining 2^n–2 Haar spectral coefficients are only locally sensitive. A spectral coefficient $r_l^{(k)}$ is characterised by its degree l and order k, where $0 \le l \le n-1$ and $0 \le k \le 2^l-1$.

Property 3.2: For a Haar spectrum of an n-variable Boolean function F, there are 2^l spectral coefficients of degree l, and each measures a correlation of a different set of 2^{n-l} neighbouring minterms, where l = 1, 2, ..., n–1. The value of the *dc* coefficient r_{dc} is proportional to the number of true and don't care minterms of F, and the coefficient $r_0^{(0)}$ describes the difference between the number of truth and don't care minterms in the subfunctions $\bar{x}_n$ and x_n.

Definition 3.3: A standard trivial function, denoted by u_I, $I = 2^l + k$ and $I \in \{1, 2, ..., 2^n - 1\}$, associated with each Haar spectral coefficient r_{dc} or $r_l^{(k)}$, describes some set of 2^{n-l} neighbouring minterms on a Karnaugh map that has an influence on the value of a spectral coefficient r_{dc} or $r_l^{(k)}$, where $0 \le l \le n - 1$ and $0 \le k \le 2^l - 1$. For each index I of u_I, there exists a unique value of l and k. Formally, u_I can be expressed as a product term:

$$u_0 = u_1 = 1 \text{ and } u_I = \prod_{i=1}^{l} x_{n-l+i}^{k_i} \quad \forall l, k \in Z;$$
$$1 \le l \le n-1 \text{ and } 0 \le k \le 2^l - 1 \tag{5}$$

where Z is the set of integers, and k_i is the ith bit value when k is expressed as a binary l-tuple $k_l k_{l-1} ... k_1$.

Property 3.4: The degree l of a Haar coefficient indicates the number of literals present in a standard trivial function u_I for I = 2, 3, ..., 2^n – 1.

Property 3.5: The order k of a Haar spectral coefficient $r_l^{(k)}$ is the decimal equivalence of the binary l-tuple, formed by writing a 1 or 0 for each variable in its standard trivial function u_I (I = 2, 3, ..., 2^n – 1), according to whether this literal appears in affirmation or negation. When k is expressed as a binary l-tuple, the most significant bit (MSB) corresponds to the literal $\dot{x}_n$, and the least significant bit (LSB) corresponds to the literal $\dot{x}_{n-l+1}$.

Based on the recursive definition of an unnormalised Haar transform in eqn. 4, a polynomial Haar expansion of an n-variable Boolean function F can be derived.

Lemma 3.6:

$$F(X) = \frac{1}{2^n}\left\{ r_{dc} + (-1)^{x_n} r_0^{(0)} + \sum_{l=1}^{n-1} 2^l (-1)^{x_{n-l}} \sum_{k=0}^{2^l-1} r_l^{(k)} \prod_{i=n-l+1}^{n} x_i^{k_{i-n+l}} \right\} \tag{6}$$

where $k_i \in \{0, 1\}$ is the ith bit in the binary l-tuple of order k; $x_i^j = x_i$ if $j = 1$ and $x_i^j = \bar{x}_i$ if $j = 0$.

Proof: The inverse unnormalised Haar transform T_N^{-1} of order of $N = 2^n$ can be defined in a similar recursive form as in eqn. 4:

$$T_N^{-1} = \frac{1}{2^n} G_N^T \tag{7}$$

where

$$G_N = \begin{bmatrix} G_{\frac{N}{2}} \otimes [1 \quad 1] \\ \frac{N}{2}\mathbf{I}_{\frac{N}{2}} \otimes [1 \quad -1] \end{bmatrix} \text{ and } G_1 = 1$$

Comparing eqns. 4 and 7, G_N is generated from T_N by incorporating a scaling factor of $N/2$. For $n \ge 1$, since each iteration of eqn. 7 generates an additional degree of inverse Haar functions, the scaling factor of 2^l can be applied to every forward Haar function of the same degree l to obtain the corresponding row in G_N.

Hence, $T^{-1}(0) = (1/2^n)T^T(0)$ and $T^{-1}(I) = (1/2^n) \times 2^l \times T^T(I)$, where I = 1, 2, ..., 2^n – 1, the superscript T denotes matrix transpose, and $T(I)$ is the row vector corresponding to row I of the matrix T, where $I = 2^l + k$.

$$\begin{aligned} F(X) &= T^{-1} \times R \\ &= [T^{-1}(0)]^T r_{dc} + [T^{-1}(1)]^T r_0^{(0)} \\ &\quad + [T^{-1}(2)]^T r_1^{(0)} + [T^{-1}(3)]^T r_1^{(1)} \\ &\quad + \cdots + [T^{-1}(2^n - 1)]^T r_{n-1}^{(2^{n-1}-1)} \\ &= \frac{1}{2^n}\left\{ T(0) r_{dc} + \sum_{l=0}^{n-1}\sum_{k=0}^{2^l-1} 2^l T(2^l + k) r_l^{(k)} \right\} \end{aligned}$$

The forward Haar functions representing each row of the transform T are given by: $T(0) = 1$, $T(1) = (\bar{x}_{n-l} - x_{n-l})$ and $T(I) = T(2^l + k) = u_I(\bar{x}_{n-l} - x_{n-l})$, where u_I is the standard trivial function corresponding to the Haar function $T(I)$. Since $\bar{x}_{n-l} - x_{n-l} = 1$ if $x_{n-l} = 0$, and –1 if $x_{n-1} = 1$, $\bar{x}_{n-l} - x_{n-l} = (-1)^{x_{n-l}}$. The standard trivial function u_I can be represented by the product $\prod_{i=n-l+1}^{n} x_i^{k_{i-n+l}}$, where $k_{i-n+l} \in \{0, 1\}$ is the $(i - n + l)$th bit in the binary l-tuple of order k, $x_i^{k_{i-n+l}} = x_i$ if $k_{i-n+l} = 1$, and $x_i^{k_{i-n+l}} = \bar{x}_i$ if $k_{i-n+l} = 0$. Thus,

$$F(X) = \frac{1}{2^n}\left\{ r_{dc} + (-1)^{x_n} r_0^{(0)} + \sum_{l=1}^{n-1}\sum_{k=0}^{2^l-1} 2^l(-1)^{x_{n-l}} r_l^{(k)} \prod_{i=n-l+1}^{n} x_i^{k_{i-n+l}} \right\}$$
$$= \frac{1}{2^n}\left\{ r_{dc} + (-1)^{x_n} r_0^{(0)} + \sum_{l=1}^{n-1} 2^l(-1)^{x_{n-l}} \sum_{k=0}^{2^l-1} r_l^{(k)} \prod_{i=n-l+1}^{n} x_i^{k_{i-n+l}} \right\}$$

Example 3.7: Consider the four-variable incompletely specified Boolean function $F(X) = \Sigma_{ON}(0, 2, 5, 6, 10, 11) + \Sigma_{DC}(1, 4, 14)$, where the numbers enclosed in $\Sigma_{ON}(...)$ and $\Sigma_{DC}(...)$ indicate the truth and don't care minterms, respectively. The Haar spectrum calculated from the R-coded vector of F is given by:

$$R = \left[r_0 \quad r_0^{(0)} \quad r_1^{(0)} \quad r_1^{(1)} \quad r_2^{(0)} \quad \cdots \quad r_2^{(3)} \quad r_3^{(0)} \quad \cdots \quad r_3^{(7)} \right]^T$$
$$= [7.5 \quad 2.5 \quad 0 \quad 1.5 \quad 0.5 \quad 0.5 \quad -2 \quad -0.5 \quad 0.5 \quad 1 \quad -0.5 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0.5]^T$$

From lemma 3.6, the Haar expansion of this function is given by:

$$F(X) = \frac{1}{16}\{7.5 + (-1)^{x_4}(2.5) + (-1)^{x_3}(3x_4) + (-1)^{x_2}(2\overline{x}_4\overline{x}_3 + 2\overline{x}_4 x_3 - 8x_4\overline{x}_3 - 2x_4x_3) + (-1)^{x_1}(4\overline{x}_4\overline{x}_3\overline{x}_2 + 8\overline{x}_4\overline{x}_3 x_2 - 4\overline{x}_4 x_3\overline{x}_2 + 8\overline{x}_4 x_3 x_2 + 4x_4x_3x_2)\}$$

Consider the input assignment $X = 1 = 0001_2$, i.e. $x_4 = x_3 = x_2 = 0$ and $x_1 = 1$. The R-coded value of F under this input assignment can be calculated from the above expansion: $F(1) = (1/16)(7.5 + 2.5 + 0 + 2 - 4) = 0.5$.

For efficient synthesis of incompletely specified Boolean functions, instead of operating on a single spectrum from the R-coded vector, a paired Haar transform has been introduced [11, 13].

Definition 3.8: A paired Haar transform (PHT) for an incompletely specified n-variable Boolean function F is a mapping χ: $(F_{ON}, F_{DC}) \to (R_{ON}, R_{DC})$, where $R_{ON} = T \times F_{ON}$ and $R_{DC} = T \times F_{DC}$. F_{ON} is obtained by replacing all don't care outputs of F by zeros, and F_{DC} is obtained from F by replacing all true outputs by zeros and don't care outputs by ones. T is the unnormalised Haar transform defined in eqn. 4. The tuple (R_{ON}, R_{DC}) is known as the paired Haar spectrum. Spectral coefficients from spectra R_{ON} and R_{DC} are indicated by lower case letters accordingly.

In R coding, the unnormalised Haar spectrum is related to the paired Haar spectrum as follows:

$$R = R_{ON} + 0.5 \times R_{DC} \tag{8}$$

Example 3.9: For the four-variable incompletely specified Boolean function from example 3.7, the paired Haar spectrum $(R_{ON}, R_{DC}) = [((r_{ON})_{dc}, (r_{DC})_{dc})$ $((r_{ON})_0^{(0)}, (r_{DC})_0^{(0)})$ $((r_{ON})_1^{(0)}, (r_{DC})_1^{(0)})$ ... $((r_{ON})_3^{(7)}, (r_{DC})_3^{(7)})]^T$ = [(6, 3) (2, 1) (0, 0) (2, –1) (0, 1) (0, 1) (–2, 0) (0, –1) (1, –1) (1, 0) (–1, 1) (1, 0) (0, 0) (0, 0) (0, 0) (0, 1)]^T.

Definition 3.10: Let X_i be an input assignment covered by a cube C whose cardinality is equal to $|C|$, then the output signal probability $p(C)$ under the set of input assignments X_i ∀ i = 1, 2, ..., $|C|$ is given by:

$$p(C) = \frac{1}{|C|}\sum_{i=1}^{|C|} F(X_i) \tag{9}$$

where $F(X_i)$ is the R-coded functional value of the input assignment X_i.

The value of $p(C)$ lies between 0 and 1, which indicates the likelihood of the cube C being an implicant of the function F. $p(C) = 1$ if C is an ON cube, and $p(C) = 0$ if C is an OFF cube.

When applying the statistical decision theory to logic synthesis problems, we are often interested in comparing the equivocations for different input assignments, and selecting one with the maximum likelihood. Depending on the formulation of the problem, the maximum likelihood decision corresponds to either the maximum or minimum equivocation. Since the conditional entropies $p\log_2 p$ and $p\log_2 p + (1 - p)\log_2(1 - p)$ encountered in the decision problems are monotonic increasing for $0 \le p \le 0.5$, and monotonic decreasing in the range of $0.5 \le p \le 1$, an appropriate metric to describe the equivocation for an input assignment C would be a number that is proportional to $|p(C) - 0.5|$, where | | denotes the absolute value. We call this number the likelihood metric.

Theorem 3.11: Let $\rho_i(C)$ be the cube resulting from a logical right shift of the cube C by i bits, and $\gamma_i(C)$ be the number of '–' in C between bit 1 and bit i inclusive. Then, the likelihood metric $M(C)$ can be expressed as the summation of selected paired Haar coefficients:

$$M(C) = \left| M_{ON}(C) + \frac{1}{2}M_{DC}(C) - 2^{n-1}|C| \right| \tag{10}$$

where $M_{ON}(C) = |C|(r_{ON})_{dc} + \Sigma_{l=0}^{n-1}\,\delta_{ON}(l)$, where

$$\delta_{ON}(l) = \begin{cases} 0 & \text{if } x_{n-l} = \text{'–'} \\ 2^{l+\gamma_{n-l}(C)}(-1)^{x_{n-l}} \sum\limits_{X\in\rho_{n-l}(C)} (r_{ON})_l^{(X)} & \text{if } x_{n-l} \neq \text{'–'} \end{cases} \tag{11}$$

and $M_{DC}(C) = |C|(r_{DC})_{dc} + \Sigma_{l=0}^{n-1}\,\delta_{DC}(l)$, where

$$\delta_{DC}(l) = \begin{cases} 0 & \text{if } x_{n-l} = \text{'–'} \\ 2^{l+\gamma_{n-l}(C)}(-1)^{x_{n-l}} \sum\limits_{X\in\rho_{n-l}(C)} (r_{DC})_l^{(X)} & \text{if } x_{n-l} \neq \text{'–'} \end{cases} \tag{12}$$

In the above equations, $X \in \rho_{n-l}(C)$ denotes the set of input assignments (in decimal number representation) covered by the cube $\rho_{n-l}(C)$.

Proof: From eqn. 6, for any l, $0 \le l \le n-1$, the product $\Pi_{i=n-l+1}^{n}\, x_i^{k_{i-n+l}} = 1$ iff $k = \lfloor X/2^{n-l} \rfloor$, and = 0 for any other k in the range of $0 \le k \le 2^l-1$, where X is the decimal number representation of an input assignment, and the symbol $\lfloor z \rfloor$ denotes the largest integer not exceeding z. Substituting the second summation in eqn. 6 by the non vanishing coefficient $r_l^{(k)}$ of each degree l, where $k = \lfloor X/2^{n-l} \rfloor$, we have

$$F(X) = \frac{1}{2^n}\left\{ r_{dc} + \sum_{l=0}^{n-1} 2^l(-1)^{x_{n-l}} r_l^{\lfloor \frac{X}{2^{n-l}} \rfloor} \right\} \tag{13}$$

It should be noted that the value $\lfloor X/2^{n-l} \rfloor$ can also be obtained by shifting $n-l$ bits in X to the right.

We can substitute all input assignments covered by the cube C into the simplified expression of eqn. 13 for the term $\Sigma_{i=1}^{|C|} F(X_i)$ in eqn. 9. Since the MSB and LSB of the product term representing the standard trivial function are x_n and x_{n-l+1}, respectively, the value of $\lfloor X/2^{n-l} \rfloor$ in eqn. 13 is identical for all input assignments with the same bit values x_i for all $n-l+1 \le i \le n$. Thus, $\lfloor X/2^{n-l} \rfloor = X$ for all X covered by $\rho_{n-l}(C)$. As there are $2^{\gamma_{n-l}}(C)$ input assignments covered by $\rho_{n-l}(C)$, the summation in eqn. 13 is given by:

$$\sum_{l=0}^{n-1} \delta(l) = \sum_{l=0}^{n-1} 2^{l+\gamma_{n-l}(C)}(-1)^{x_{n-l}} \sum_{X \in \rho_{n-l}(C)} r_l^{(X)}$$

When $x_{n-l} = $ '–', there is an equal number of positive and negative $r_l^{(X)}$ for all X covered by $\rho_{n-l}(C)$ due to the multiplier $(-1)^{x_{n-l}}$. Therefore, $\delta(l) = 0$ for $x_{n-l} = $ '–'. In addition, the term r_{dc} is summed $|C|$ times as there are $|C|$ input assignments covered by C. Hence, $\Sigma_{i=1}^{|C|} F(X_i) = \frac{1}{2}\{|C| r_{dc} + \Sigma_{l=0}^{n-1} \delta(l)\}$, where

$$\delta(l) = \begin{cases} 0 & \text{if } x_{n-l} = \text{`}-\text{'} \\ 2^{l+\gamma_{n-l}(C)}(-1)^{x_{n-l}} \sum\limits_{X \in \rho_{n-l}(C)} r_l^{(X)} & \\ & \text{if } x_{n-l} \ne \text{`}-\text{'} \end{cases} \quad (14)$$

From Definition 3.10, $|p(C) - 0.5|$ is proportional to $|\Sigma_{i=1}^{|C|} F(X_i) - |C|/2|$. Hence, $M(C) = |\,|C| r_{dc} + \Sigma_{l=0}^{n-1} \delta(l) - 2^{n-1}|C|\,|$. Since $r_{dc} = (r_{ON})_{dc} + 0.5 \times (r_{DC})_{dc}$ and $r_l^{(k)} = (r_{ON})_l^{(k)} + 0.5 \times (r_{DC})_l^{(k)}$, $p(C)$ can be divided into $p_{ON}(C)$ and $p_{DC}(C)$, with $r_l^{(k)}$ replaced by $(r_{ON})_l^{(k)}$ in $\delta_{ON}(l)$ from eqn. 11, and by $(r_{DC})_l^{(k)}$ in $\delta_{DC}(l)$ from eqn. 12, respectively.

Lemma 3.12: When $M_{ON}(C) + M_{DC}(C) = 2^n|C|$, the cube C is an implicant of F, provided that all the don't care minterms covered by C are assigned as 1.

Lemma 3.13: When $M_{ON}(C) = 0$, the cube C is an implicant of $\bar{F}$, provided that all the don't care minterms covered by C are assigned as 0.

Example 3.14: Consider the four variable incompletely specified Boolean function from Example 3.9. The calculation of $M(C)$ for the cube $C = $ —10 by theorem 3.11 is shown below.

From eqns. 11 and 12, $\delta_{ON}(0) = \delta_{ON}(1) = \delta_{DC}(0) = \delta_{DC}(1) = 0$ since $x_4 = x_3 = -$. $\rho_2(C) = 00$—, $\gamma_2(C) = 0$. $\delta_{ON}(2) = 2^{2+0}(-1)^1((r_{ON})_2^{(0)} + (r_{ON})_2^{(1)} + (r_{ON})_2^{(2)} + (r_{ON})_2^{(3)}) = 8$. $\delta_{DC}(2) = -4((r_{DC})_2^{(0)} + (r_{DC})_2^{(1)} + (r_{DC})_2^{(2)} + (r_{DC})_2^{(3)}) = -4$. $\rho_1(C) = 0$—1, $\gamma_1(C) = 0$. $\delta_{ON}(3) = 2^{3+0}(-1)^0((r_{ON})_3^{(1)} + (r_{ON})_3^{(3)} + (r_{ON})_3^{(5)} + (r_{ON})_3^{(7)}) = 8(2) = 16$. $\delta_{DC}(3) = 8((r_{DC})_3^{(1)} + (r_{DC})_3^{(3)} + (r_{DC})_3^{(5)} + (r_{DC})_3^{(7)}) = 8$. Since $|C| = 4$, we have $M_{ON}(C) = 4 \times 6 + 8 + 16 = 48$ and $M_{DC}(C) = 4 \times 3 - 4 + 8 = 16$. From eqn. 10, $M(C) = |48 + 0.5 \times 16 - 8 \times 4| = 48$. The likelihood metric $M(C)$ is relatively low, indicating a high probability that C is an implicant of the function or its complement. In fact, from Lemma 3.12, since $M_{ON}(C) + M_{DC}(C) = 16 \times 4 = 64$, C is an implicant of the function by assigning the don't care minterm 1110 to '1'.

4 Generation of quasi-optimal FBDD and OBDD through paired Haar spectrum

The notions of the free binary decision diagram and ordered binary decision diagram have been discussed in Section 2. It should be noted that the choice of the decision variable at each vertex of the FBDD and OBDD has a strong influence on the size of the resulting decision diagram, in terms of the number of vertices. However, optimal selection of decision variables is an NP-complete problem [18–25]. Particularly, the allocations of don't care minterms are to be considered together when the BDD is to be optimised for an incompletely specified Boolean function. In the worst case, the decision diagram has $2^n - 1$ nonterminal vertices. The depth of a path is the number of nonterminal vertices in it. A path with a depth d eliminates a complete subtree of $2^{n-d} - 1$ nonterminal vertices from the worst case decision diagram. Hence, our aim is to minimise the depth of as many paths as possible in the selection of decision variables for each level.

From property 2.1, for each vertex v in a path η, if its 1-edge is also contained in η, then the variable x_i is present in the cube, where $i = index(v)$. Otherwise, the variable $\bar{x}_i$ is present. The logical value of the cube follows the functional value ε of the terminal vertex in the path. The cubes obtained from any two paths of an FBDD or an OBDD are disjoint. Since each path of an FBDD or an OBDD that leads to a 1- or a 0-terminus is a disjoint cube of the Boolean function or its complement, our primary goal is to search for disjoint cubes with cardinalities as large as possible that completely cover a given function or its complement.

Let X and Y denote the random variables associated with the decision variables and the terminal value of a path, respectively. Then, the conditional entropy $H(Y = \varepsilon | X_i)$ is the likelihood or expectancy that the children of the vertex are terminal vertices with value ε, given that a decision variable x_i has been selected. Therefore, a quasi-optimal FBDD can be generated by recursively seeking a maximum likelihood metric for each path of an FBDD in a depth first traversal. Let $C = \langle c_n\, c_{n-1} \ldots c_1 \rangle$ be the cube associated with a vertex v of the FBDD, where c_i is the edge value of the decision variable x_i being traversed from the root to v, and the vacuous variables in C, denoted by '–', are all the possible decision variables for the vertex v. Then, we have the following propositions:

Proposition 4.1: Let C denote the cube associated with the path from the root of the FBDD to a vertex v. At the root of the FBDD, C is an n-cube. During depth first traversal of the FBDD, a candidate vacuous variable x_i of C is selected for the vertex v, such that $M_{\max}(C \cap \dot{x}_i) = \max_{x_s \in \Omega}(\max(M(C \cap \bar{x}_s), M(C \cap x_s)))$, where Ω is the set of m vacuous variables of C. The candidate variable x_i is used to decompose C into two $(m-1)$-cubes $C_0 = C \cap \bar{x}_i$ and $C_1 = C \cap x_i$, associated with the children $low(v)$ and $high(v)$, respectively. When $M_{ON}(C) + M_{DC}(C) = 2^n|C|$, the vertex associated with the cube C can be replaced by a 1-terminus. When $M_{ON}(C) = 0$, the vertex associated with the cube C can be replaced by a 0-terminus.

Proposition 4.2: To improve the quality of the results obtained from proposition 4.1, if there is more than one vacuous variable x_i of C with the maximum likelihood metric $M_{max}(C \cap \dot{x}_i)$, any variable among them with the maximum likelihood metric of $M_{DC_{\max}}(C \cap \dot{x}_i) = \max_{\dot{x}_s \in \Omega'} M_{DC}(C \cap \dot{x}_s)$ is selected as a candidate variable, where Ω' is the set of literals with $M(C \cap \dot{x}_i) = M_{\max}(C \cap \dot{x}_i)$.

In proposition 4.2, when two variables lead to the same conditional entropy, we select one that leads to the decomposition with more allocable don't care out-

Table 1: Calculation of likelihood metrics

Vertex	C_{j0}	C_{j1}	$M_{ON}(C_{j0})$	$M_{DC}(C_{j0})$	$M(C_{j0})$	$M_{ON}(C_{j1})$	$M_{DC}(C_{j1})$	$M(C_{j1})$	$\max(M(C_{j0}), M(C_{j1}))$
a : {––––}	–––0	–––1	64	32	16	32	16	24	24
	––0–	––1–	32	32	16	64	16	8	16
	–0––	–1––	64	16	8	32	32	16	16
	0–––	1–––	64	32	16	32	16	24	24
b : {–––0}	––00	––10	16	16	8	48	16	24	24
	–0–0	–1–0	48	0	16	16	32	0	16
	0––0	1––0	48	16	24	16	16	8	24
c : {–––1}	––01	––11	16	16	8	16	0	16	16
	–0–1	–1–1	16	16	8	16	0	16	16
	0––1	1––1	16	16	8	16	0	16	16
d : {––00}	–000	–100	16	0	0	0	16	8	8
	0–00	1–00	16	16	8	0	0	16	16
e : {––01}	–001	–101	0	16	8	16	0	0	8
	0–01	1–01	16	16	8	0	0	16	16
f : {––11}	–011	–111	16	0	0	0	0	16	16
	0–11	1–11	0	0	16	16	0	0	16
g : {–011}	0011	1011	0	0	8	16	0	8	16

puts for the children, i.e. one that maximises the entropy of $H(Y = 0.5|X_i)$. The algorithm for the selection of a good decision variable for a vertex v for an incompletely specified Boolean function from its paired Haar spectrum, (R_{ON}, R_{DC}) is shown below:

Procedure Select_var(R_{ON}, R_{DC}, C) {

for (each $x_i \in \Omega$, the set of vacuous variables of C) {

Calculate $M_{ON}(C \cap \bar{x}_i)$, $M_{DC}(C \cap \bar{x}_i)$ $M_{ON}(C \cap x_i)$ and $M_{DC}(C \cap x_i)$ from (R_{ON}, R_{DC});

$M(C \cap \bar{x}_i) = |M_{ON}(C \cap \bar{x}_i) + \frac{1}{2}M_{DC}(C \cap \bar{x}_i) - 2^{n-1}|C \cap \bar{x}_i||$;

$M(C \cap x_i) = |M_{ON}(C \cap x_i) + \frac{1}{2}M_{DC}(C \cap x_i) - 2^{n-1}|C \cap x_i||$;

$M(C \cap \dot{x}_i) = \max(M(C \cap \bar{x}_i), M(C \cap x_i))$;

}

$V = \{x_i \in \Omega | M_{\max}(C \cap \dot{x}_i) = \min_{x_s \in \Omega} M(C \cap \dot{x}_s)\}$;

if $(|V| \neq 1)$ Select any variable $x_i \in V$ such that $M_{DC\max}(C \cap \dot{x}_i) = \max_{\dot{x}_s \in \Omega'} M_{DC}(C \cap \dot{x}_s)$;

return index of the selected variable;

}

In the procedure **Select_var**, the likelihood metric $M(C \cap \dot{x}_i)$ can be calculated from selected coefficients of the paired Haar spectrum by eqn. 10. A computational cache may be used to cache the previously calculated likelihood metrics. Based on propositions 4.1 and 4.2, the algorithm for the generation of a quasi-optimal FBDD for an incompletely specified Boolean function is shown below:

FBDD_MIN(R_{ON}, R_{DC}) {

Initialize(*C*, *fbdd*, *unique_table*);

fbdd→*root* = **FBDD_MIN_AUX**$(R_{ON}, R_{DC}, C,$ *unique_table*);

return *fbdd*;

}

FBDD_MIN_AUX$(R_{ON}, R_{DC}, C,$ *unique_table*) {

p = **probability**$(R_{ON}, R_{DC}, C, \&p_{ON}, \&p_{DC})$;

if $(p_{ON} = 0)$ **return** *FBDD_ZERO*;

if $(p_{ON} + p_{DC} = 1)$ **return** *FBDD_ONE*;

i = **Select_var**(R_{ON}, R_{DC}, C);

$C_0 = C \cap \bar{x}_i$; $C_1 = C \cap x_i$;

low = **FBDD_MIN_AUX**$(R_{ON}, R_{DC}, C_0,$ *unique_table*);

high = **FBDD_MIN_AUX**$(R_{ON}, R_{DC}, C_1,$ *unique_table*);

if (*low* = *high*) **return** *low*;

return unique_table_find(*unique_table*, x_i, *low*, *high*);

}

The procedure **Initialize** sets up the FBDD structure *fbdd* and a unique node table *unique_table*, that keeps only unique vertices generated by the algorithm. The cube C is initialised to be an n-cube, where n is the number of input variables. The procedure **FBDD_MIN_AUX** is a recursive routine that generates the vertices of the minimal FBDD by depth first traversal. In **FBDD_MIN_AUX**, *FBDD_ZERO* and *FBDD_ONE* are the 0- and 1-termini, respectively. The procedure **Select_var** in the first algorithm is used to determine the best top variable x_i for the present vertex. The variables *low* and *high* are the pointers to the low and high children of the present vertex, respectively. The procedure **unique_table_find** searches in the *unique_table* for the vertex with the specified top variable and children. If found, it returns the pointer to the targeted vertex. Otherwise, a new vertex with the specified top variable and children is inserted in *unique_table* and returned.

Example 4.3: Consider the incompletely specified Boolean function from example 3.7. Fig. 1 shows the FBDD generated by the procedure **FBDD_MIN**, and the vertex variables are decided based on the maximum likelihood metrics calculated in Table 1. The minimised FBDD consists of six nonterminal vertices. Incidentally, it is also an OBDD of variable ordering <1, 2, 3, 4>.

5 Experimental results

The algorithm **FBDD_MIN** is implemented in C, and the minimal or near minimal FBDDs are generated on

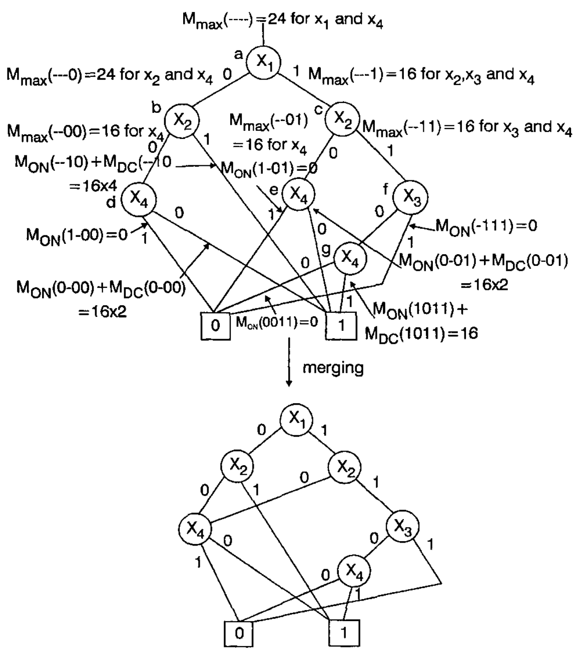

Fig. 1 *Generation of quasi-optimal FBDD*

an HP Apollo Series 715 workstation for some benchmark functions from the two-level examples of the MCNC benchmark suite. Since our present algorithm is designed to work with a single output, incompletely specified function, a simplified and direct extension of **FBDD_MIN** is adopted to generate a mult-root FBDD for multiple output functions by treating each output independently. The results are compared with the minimal OBDD generated by the dynamic variable reordering technique [22, 25] in Table 2. In the dynamic variable reordering technique, each variable is moved throughout the order to find an optimal position for that variable, assuming all other variables are fixed. Since the method [22, 25] does not deal with incompletely specified functions, the don't care outputs, if any, are randomly allocated before their OBDDs are generated. In Table 2, the columns labelled '#inputs' and '#outputs' are the numbers of input variables and outputs for each system of functions, respectively. The columns labelled 'Size (OBDD)' and 'Size (FBDD)' denote the number of non-terminal vertices of the multi-root OBDD obtained by the dynamic variable reordering algorithm, and the FBDD obtained by our algorithm, respectively. The column labelled 'Time' is the system execution time of our algorithm in seconds. Due to the temporary exponential size blow up in reshuffling the variables in the midst of the minimisation, an overflow occurs for the benchmark function ex1010 in the dynamic variable reordering algorithm. For single output functions, our algorithm always generates better results, and for multiple output functions, the quality of the results is comparable. The less favourable results for functions with a large number of outputs can be explained by the fact that our local minimisation effort of individual output does not always contribute to a global minimisation effectively. When different selected outputs of the functions apex4 and bw are tested separately, our algorithm always generates a smaller FBDD than the minimal OBDD generated by the dynamic variable reordering algorithm. In order to achieve such a global minimisation of the FBDD for all functions at once, a statistical test or measure of the percentage isomorphic subgraphs among different outputs for all possible decisions has to be formulated. At present, a good likelihood metric for the information theoretic approach to this problem has not been derived.

An $M(k)$ universal logic module (ULM) is a multiplexer with k control inputs and 2^k data inputs. By modifying the FBDD vertex structure such that it contains k vertex variables and 2^k edges corresponding to 2^k possible minterms of k variables, the same algorithm **FBDD_MIN** can be used for $M(k)$ ULM network synthesis. Besides constant termini, additional exit conditions are introduced to check whether the data input

Table 2: Benchmark results for FBDD_MIN and multiplexer synthesis

	#inputs	#outputs	Size (OBDD)	Size (FBDD)	Time, s	#$M(2)$	Time, s	#$M(3)$	Time, s
9sym	9	1	35	33	0.06	15	0.07	11	0.08
Z9sym	9	1	35	33	0.04	15	0.09	11	0.15
5xp1	7	10	81	104	0.04	52	0.09	38	0.07
Z5xp1	7	10	71	69	0.04	49	0.07	38	0.12
sao2	10	4	134	130	0.11	75	0.31	40	0.67
apex4	9	19	1118	1465	0.54	884	0.73	720	1.47
bw	5	28	126	139	0.05	79	0.07	29	0.03
clip	9	5	177	207	0.10	108	0.24	77	0.40
con1	7	2	20	21	0.03	11	0.04	7	0.03
inc	7	9	97	89	0.06	51	0.08	36	0.11
misex1	8	7	51	54	0.08	31	0.16	20	0.19
sqrt8	8	4	53	41	0.02	22	0.11	14	0.12
ex1010	10	10	–	1231	0.86	767	1.49	512	2.63
squar5	5	8	42	48	0.03	24	0.01	12	0.02
xor5	5	1	11	9	0.05	3	0.03	1	0.06
rd53	5	3	25	23	0.02	9	0.02	2	0.03
rd73	7	3	45	43	0.05	19	0.04	13	0.03
rd84	8	4	61	59	0.10	30	0.07	8	0.09

can be terminated with a single variable. With the modified algorithm, the incompletely specified function from example 3.7 can be synthesised by a 2-1 multiplexer network consisting of four $M(1)$ ULMs. The result is shown in Fig. 2. The same benchmark functions are optimised by a network implemented with $M(2)$ and $M(3)$ multiplexers, and the results are shown in the rightmost four columns of Table 2, where the columns labelled '$\#M(2)$' and '$\#M(3)$' are the number of $M(2)$ and $M(3)$ ULMs required, respectively. Unlike the level-by-level technique used in [5], our algorithm permits mixed control variables within each level, if this would result in more data paths being terminated with a shorter depth. However, owing to the same simplified approach towards minimisation of multiple output functions, and the fact that input inverters between modules are not used in our implementation, the quality of our results appears to be less attractive than the published results of [5] for the same benchmark functions.

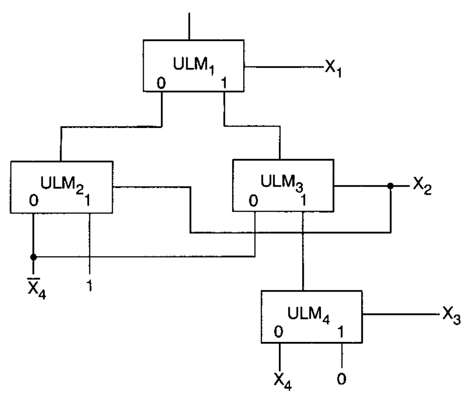

Fig. 2 *Synthesis of 2-1 multiplexer network*

6 Conclusions

The paired Haar transform has been introduced as an extension of the unnormalised Haar transform, to specially deal with the added complexity in allocating the don't care sets of incompletely specified Boolean functions [11, 13]. In the applications of the paired Haar spectrum to logic minimisation, the free binary decision diagrams have been considered. Since exact minimisation of the FBDDs has been proven to be NP-hard, the algorithms proposed for their optimisation are heuristic. By treating them as a general combinatorial decision problem, the concept of entropy and equivocation are adopted and re-formulated in terms of the paired Haar spectrum. Although we have demonstrated the minimisation methods by examples of single output, incompletely specified Boolean functions, the presented theorems can easily be extended to the paired Haar spectrum for a system of incompletely specified functions. The minimisation problems can also be easily extended to other decision based applications in logic synthesis, for example, multiplexer synthesis [5], where the level-by-level heuristics can be modelled as a special free decision diagram with k-variables for each vertex, and 2^k edges corresponding to the different polarities of k data select variables. In addition, the equivocation or conditional entropy computation can be biased to suit different cost functions or priorities of conflicting criteria. A good example is in the case of OBDD optimisation: if there exist many vertices at the same level, which belong to some classes of functions that have small OBDD sizes under some specific orderings of variables, the conditional entropies can be weighted so that those orderings of variables have greater precedence.

7 Acknowledgments

The authors wish to thank the referees for numerous helpful comments.

8 References

1 HURST, S.L., MILLER, D.M., and MUZIO, J.C.: 'Spectral techniques in digital logic' (Academic Press, London, 1985)
2 KARPOVSKY, M.G.: 'Finite orthogonal series in the design of digital devices' (John Wiley, New York, 1976)
3 RUIZ, G., MICHELL, J.A., and BURON, A.: 'Fault detection and diagnosis for MOS circuits from Haar and Walsh spectrum analysis: on the fault coverage of Haar reduced analysis', *in* MORAGA, C. (Ed.): 'Theory and applications of spectral techniques' (University Dortmund Press, 1988), pp. 97–106
4 RUIZ, G., MICHELL, J.A., and BURON, A.: 'Switch-level fault detection and diagnosis environment for MOS digital circuits using spectral techniques', *IEE Proc., Comput. Digit. Tech.*, 1992, **139**, (4), pp. 293–307
5 SCHAEFER, I., and PERKOWSKI, M.A.: 'Synthesis of multilevel multiplexer circuits for incompletely specified multioutput Boolean functions with mapping to multiplexer based FPGAs', *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 1993, **12**, (11), pp. 1655–1664
6 SASAO, T.: 'Logic synthesis and optimisation' (Kluwer Academic, Boston, 1993)
7 SHORE, J.E.: 'On the applications of Haar functions', *IEEE Trans. Commun.*, 1973, **21**, pp. 206–216
8 ZALMANZON, L.A.: 'Fourier, Walsh and Haar transforms and their application in control, communication and other fields' (Nauka, Moscow, 1989) (in Russian)
9 AHMED, N., and RAO, K.R.: 'Orthogonal transforms for digital signal processing' (Springer–Verlag, Berlin, 1975)
10 BURON, M., MICHELL, J.A., and SOLANA, J.M.: 'Single chip fast Haar transform at megahertz rates', *in* MORAGA, C. (Ed.): 'Theory and applications of spectral techniques' (University Dortmund Press, 1988), pp. 8–17
11 FALKOWSKI, B.J., and CHANG, C.H.: 'A novel paired Haar based transform: algorithms and interpretations in Boolean domain'. 36th Midwest Symposium on *Circuits and systems*, Detroit, Michigan, August 1993, pp. 1101–1104
12 FALKOWSKI, B.J., and CHANG, C.H.: 'Forward and inverse transformations between Haar spectra and ordered binary decision diagrams of Boolean functions', *IEEE Trans. Comput.*, 1997, **46**, (11), pp. 1272–1279
13 FALKOWSKI, B.J., and CHANG, C.H.: 'Properties and applications of paired Haar transform'. 1st IEEE international conference on *Information, communications and signal processing*, Singapore, September 1997, Vol. 1, pp. 48–51
14 ROESER, P.R., and JERNIGAN, M.E.: 'Fast Haar transform algorithms', *IEEE Trans. Comput.*, 1982, **31**, (2), pp. 175–177
15 BARTLETT, K., BRAYTON, R., HACHTEL, G., JACOBY, R., MORRISON, C., RUDELL, R., SANGIOVANNI-VINCENTELLI, A., and WANG, A.: 'Multi-level logic minimization using implicit don't cares', *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 1988, **7**, (6), pp. 723–740
16 BRAYTON, R.K., HACHTEL, G., MCMULLEN, C., and SANGIOVANNI-VINCENTELLI, A.: 'Logic minimisation algorithms for VLSI synthesis' (Kluwer Academic, Boston, 1985)
17 KABAKCIOGLU, A.M., VARSHNEY, P.K., and HARTMANN, C.R.P.: 'Application of information theory to switching function minimisation', *IEE Proc., Comput. Digit. Tech.*, 1990, **137**, (5), pp. 389–393
18 CHAKRAVARTY, S.: 'A characterization of binary decision diagrams', *IEEE Trans. Comput.*, 1993, **42**, (2), pp. 129–137
19 GERGOV, J., and MEINEL, C.: 'Efficient Boolean manipulation with OBDDs can be extended to FBDDs', *IEEE Trans. Comput.*, 1994, **43**, (10), pp. 1197–1209
20 BOLLING, B., and WEGENER, I.: 'Improving the variable ordering of OBDDs is NP-complete', *IEEE Trans. Comput.*, 1996, **45**, (9), pp. 993–1001
21 FRIEDMAN, S.J., and SUPOWIT, K.J.: 'Finding the optimal variable ordering for binary decision diagrams', *IEEE Trans. Comput.*, 1990, **39**, (5), pp. 710–713
22 FUJITA, M., MATSUNAGA, Y., and KAKUDA, T.: 'On the variable ordering of binary decision diagrams for the application of multi-level logic synthesis'. European *Design automation* conference, February 1991, pp. 50–54

23 ISHIURA, N., SAWADA, H., and YAJIMA, S.: 'Minimizations of binary decision diagrams based on exchanges of variables'. IEEE international conference on *Computer aided design*, 1991, pp. 472–475
24 MERCER, M.R., KAPUR, R., and ROSS, D.E.: 'Functional approaches to generating orderings for efficient symbolic representations'. 29th ACM/IEEE *Design automation* conference, June 1992, pp. 624–627
25 RUDELL, R.: 'Dynamic variable ordering for ordered binary decision diagrams', IEEE international conference on *Computer aided design*, 1973, pp. 42–47
26 HELLMAN, M.E.: 'An extension of the Shannon theory approach to cryptography', *IEEE Trans. Inf. Theory*, 1978, **23**, (5), pp. 289–294
27 KHINCHIN, A.I.: 'Mathematical foundations of information theory' (Dover Publications Inc., New York, 1957)