Functional Testing of Boolean Systems with Unknown Functionality

Ariel Burg and Osnat Keren
Bar-Ilan University
burgariel@gmail.com, kereno@macs.biu.ac.il

Ilya Levin
Tel-Aviv University
ilia1@post.tau.ac.il

Abstract

The paper deals with functional testing of Boolean systems in cases when the functionality of the system is unknown. Constructions of optimal linear-checks for testing such systems that have an acceptable representation as low order polynomials are presented. The linear checks determine a set of binary test vectors that form a relatively small subgroup of $\mathcal{C}_2^n$ ($\mathcal{C}_2^n$ consists of all the n-bit binary vectors with addition modulo two). The paper shows that for Walsh-transform-based implementations it is possible to define a subgroup in $\mathcal{C}_2^n$ which does not depend on the actual functionality of the system. Moreover, the check set can be defined even without knowing neither the number of input variables nor their precision.

1 Introduction

Two alternative types of system testing are known: on-line and off-line testing. The on-line testing (usually called concurrent checking) requires introducing an additional circuitry for detecting faults during the normal operation of the system. This kind of testing protects the system from both permanent and transient faults that may occur during its operation. In contrast, the off-line testing is a procedure allowing to check the system before use. This kind of test protects the system from two types of faults: fabrication faults and fault that occurred before the test has been applied. The off-line testing is based on applying a predefined set of test vectors. Two types of off-line testing are used: a) testing by using external equipment, and b) self testing running on a *built-in* circuit. The present paper belongs to the area of built-in-self test.

There are two conceptual levels for testing which in turn define different testing methods: gate level and functional level testing. In gate level testing, the test vectors suit a specific implementation, while on the functional level, the testing is independent from the specific implementation and tests the correctness of the operation. Up to now, design of functional testing has been carried out only if the functionality of the system was known to the test designer. In our paper, we study the case when the test designer does not know the functionality. We address the following question: is it possible to design functional testing for a system whose functionality is unknown?

In our paper we apply the spectral approach to solve the above problem. The spectral approach to testing was studied in [4, 13], and in the papers collected in the compendium "*Spectral Techniques and Fault Detection*" [7]. Testing by verification of Walsh coefficients can be viewed as data compression of test responses. Although this approach eliminates the problem of check set (also called test vectors) generation and storage, it requires exhaustive application of all 2^n possible input patterns. An efficient approach to spectral testing, that does not require exhaustive application of all 2^n possible n-bit input patterns and at the same time eliminates the problem of check set generation, is the *testing by linear-checks* [6, 7, 8, 9]. Linear-checks are used in the context of detecting permanent stack-at faults. Linear-checks allow to define the check set analytically. For functions whose Walsh spectra contains sufficiently many zeros the check set forms a relatively small subgroup, and thus the implementation cost of the testing mechanism becomes negligible in respect to the cost of the overall system. Polynomials of low order belong to this class of functions.

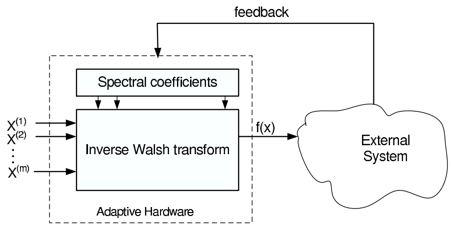

Figure 1: The architecture of a WbAH system

Denote by $K^{n_m,\dots,n_1}_{s_m,\dots,s_1}[\mathbf{x}_m,\dots,\mathbf{x}_1]$ the class of polynomials

$$f(\mathbf{x}_m,\dots,\mathbf{x}_1)=\sum_{i_m=0}^{s_m}\cdots\sum_{i_1=0}^{s_1}a(i_m,\dots,i_1)\mathbf{x}_m^{i_m}\dots\mathbf{x}_1^{i_1}$$

of m integer variables $\mathbf{x}_m,\dots,\mathbf{x}_1$ with known precision $n_m,\dots,n_1$. Methods for constructing linear (equality) checks for *a given polynomial* in $K^{n_m,\dots,n_1}_{s_m,\dots,s_1}[\mathbf{x}_m,\dots,\mathbf{x}_1]$ satisfying $s_t < n_t$ where s_t is the maximal degree of t'th variable, are presented in [6, 8]. In this paper, we address the problem of construction of linear-checks for cases where *no information is provided* on the system except the fact that it has an acceptable representation as a polynomial of order M. In particular, we provide linear-checks for Walsh-based Adaptive Hardware (WbAHs).

A WbAH is based on representing the system in the Walsh frequency domain (see Fig. 1) [10]. An (n, M) WbAH is an n-input bit circuit that can acquire the spectral coefficients of any polynomial of order M (and hence acquire the functionality of the system). As reported in [10], a WbAH provides better performance than conventional Multiply and Accumulate (MAC) based architectures in terms of its acquisition time and the average residual error (which reflects the difference between the target functionality of the system and the functionality that the hardware has converged to). However, its main advantage over a MAC-based architecture is that it acquires the functionality of the system even if neither

- the number of real (or complex) variables, nor
- the maximal degree of each variable, nor
- the precision of each variable, nor
- the order of the variables,

are known to the system designer.

In this paper, we present linear-checks for spectral testing of WbAH. As shown in the paper, the proposed check set is optimal - it is the smallest set that allows testing the WbAH without identifying the polynomial the system has converged to. The efficiency of the suggested approach in terms of implementation cost and in comparison to structural testing is demonstrated in [2].

The paper is organized as follows: mathematical background is given in Section 2. Section 3 introduces the concept of linear-checks for a given polynomial and reviews results presented in [8]. Section 4 introduces the main results of this paper, it defines linear-checks for arbitrary polynomial and shows that the check set is the best possible. Section 5 concludes the paper.

2 Preliminaries

Consider a system that has m real (or complex) inputs, and let the functionality of the system be represented by the function $f(\mathbf{x}_m,\dots\mathbf{x}_1)$. Without loss of generality, assume that the domain and the range of the function are $[-1,1)^m$ and $[-1,1)$, respectively. The inputs and the output of the system are quantized; each variable $\mathbf{x}_w$, $w=1,\dots,m$, is represented by a binary vector of n_w bits, and the value of the function is represented by a binary vector of length k. The mapping between the binary vectors and the numbers in interval $[-1,1)$ may be according to the 2's complement representation or any other weighted number system (refer to [11]).

To simplify the presentation we use bold letters for real variables, and Italic letters for Boolean variables. For example, the real variable $\mathbf{x}_w$ is represented by a binary vector $(x^{(w)}_{n_w-1}, x^{(w)}_1,\dots,x^{(w)}_0)$.

Similarly, a vector $\mathbf{x} = (\mathbf{x}_m, \ldots, \mathbf{x}_1)$ of m real variables, where each variable $\mathbf{x}_w$ is represented by n_w bits, can be referred to as a binary vector $x = (x_{n-1}, \ldots, x_0)$ of length $n = \sum_{w=1}^{m} n_w$.

Definition 1 (Order of polynomial) *Let $\mathbf{x} = (\mathbf{x}_m, \ldots, \mathbf{x}_1) \in \mathcal{C}^m$ and $D = (d_m, \ldots d_1)$, $d_i \in \{0\} \cup \mathcal{Z}^+$, and denote by $\mathbf{x}^D$ the monomial (product) $\prod_{i=1}^{m} \mathbf{x}_i^{d_i}$. Let $f(\mathbf{x}) = \sum_j a_j \mathbf{x}^{D_j}$ be a polynomial of m variables, $a_i \in \mathcal{C}$. The order M of the polynomial is $M = max_j L_1(D_j)$ where L_1 is the 1-norm, $L_1(D) = \sum_{i=1}^{m} d_i$.*

The class of polynomials $f(\mathbf{x})$ of order $\leq M$ is denoted by K_M. The class of polynomials $f(\mathbf{x}) = f(\mathbf{x}_m, \ldots, \mathbf{x}_1)$ of order $\leq M$ in $m \leq n$ quantized real variables is denoted by K_M^n. Indeed,

$$K_M^n \subset \bigcup_{m \leq n,\ \sum n_t \leq n,\ s_t \leq M} K_{s_m, \ldots, s_1}^{n_m, \ldots, n_1}[\mathbf{x}_m, \ldots, \mathbf{x}_1]. \tag{1}$$

A polynomial $f \in K_M^n$ can be referred to as a set of k switching functions of n binary variables $\{x_i\}_{i=0}^{n-1}$, or equivalently as a single multi-output function $f(x) = f(x_{n-1}, \ldots, x_0)$. The properties of a single multi-output function (or a set of switching functions) can be analyzed via the Walsh spectrum.

Definition 2 (Walsh functions) *Let $x = (x_{n-1}, \ldots, x_0)$ and $i = (i_{n-1}, \ldots, i_0)$ be two binary vectors of length n. The Walsh function $W_i(x)$ is defined as $W_i(x) = (-1)^{<x,i>} = exp(j\pi \sum_{m=0}^{n-1} x_m i_m)$.*

Denote by $\mathcal{C}_2^n$ the group of all binary n-vectors with respect to the operation $\oplus$ of componentwise addition mod 2.

Definition 3 (Walsh spectrum) *The coefficients vector of the Walsh spectrum is $S = (s_{2^n-1}, \ldots, s_0)$, where, $s_i = \sum_{x \in \mathcal{C}_2^n} W_i(x) f(x)$, and $f(x) = 2^{-n} \sum_{i \in \mathcal{C}_2^n} W_i(x) s_i$.*

The Walsh spectrum of $f \in K_M^n$ has the following property:

Theorem 1 *[10] Let $f \in K_M^n$ be a switching function in n binary variables that corresponds to a polynomial of order $M < n$. Then, the spectral coefficient s_i $(i = 0, \ldots, 2^n - 1)$ equals zero if the Hamming weight of i is greater than M.*

The correctness of the theorem follows from the linearity of the Walsh transform and from the fact that the polynomial can be represented as a sum of products of up to M Boolean variables.

Theorem 1 provides an upper bound on the number of non-zero spectral coefficients of any polynomial in K_M^n. The bound does not depend on the number of real (or complex) inputs nor on their precision. In this sense, a WbAH based on this bound is more robust than a conventional MAC implementation of a system that has an acceptable representation as a low order polynomial, since it acquires its target functionality even in cases where almost no information about the system is available.

3 Spectral testing of a given polynomial by linear-checks

Linear-checks is a method for off-line self testing that avoids the exhaustive application of all input patterns [8]. The method is based on the fact that for any given multi-valued function f in n Boolean variables there exists a subgroup T of $\mathcal{C}_2^n$ and a constant d such that $\sum_{\boldsymbol{\tau} \in T} f(x \oplus \boldsymbol{\tau}) = d$. Construction of optimal linear-checks for a given f involves finding a minimal check set (the subgroup T) and computation of d.

Denote by $V(n_t, s_t + 1)$ a maximal linear code $[n_t, k_t, s_t + 1]$ in $\mathcal{C}_2^{n_t}$ of length n_t, dimension k_t and Hamming distance $s_t + 1$. The dual code of $V(n_t, s_t + 1)$ is a linear subgroup,

$$V^{\perp}(n_t, s_t + 1) = \{\boldsymbol{\tau}_t = (\tau_{t,n_t-1}, \ldots, \tau_{t,0}) | \bigoplus_{j=0}^{n_t-1} \tau_{t,j} y_{t,j} = 0, \forall \mathbf{y}_t = (y_{t,n_t-1}, \ldots, y_{t,0}) \in V(n_t, s_t + 1)\}$$

of dimension $n_t - k_t$. Define by $V^{\perp}$ a linear code of length n that is the Cartesian product of m codes,

$$V^{\perp} \quad = \quad \prod_{t=1}^{m} V^{\perp}(n_t, s_t+1) = \{\boldsymbol{\tau} = (\boldsymbol{\tau}_m, \ldots, \boldsymbol{\tau}_1) | \boldsymbol{\tau}_t \in V^{\perp}(n_t, s_t+1)\}. \tag{2}$$

The following theorem presents linear equality checks for testing *a given polynomial*:

Theorem 2 ([8]) *Let $f \in K^{n_m,\ldots,n_1}_{s_m,\ldots,s_1}[\mathbf{x}_m, \ldots, \mathbf{x}_1]$ and $s_t < n_t$ for all $1 \le t \le m$. Then, the code $V^{\perp}$ is the check set for f, that is*

$$\sum_{\boldsymbol{\tau} \in V^{\perp}} f(\mathbf{x} \oplus \boldsymbol{\tau}) = \sum_{\boldsymbol{\tau}=(\boldsymbol{\tau}_m,\ldots,\boldsymbol{\tau}_1) \in V^{\perp}} f(\mathbf{x}_m \oplus \boldsymbol{\tau}_m, \ldots, \mathbf{x}_1 \oplus \boldsymbol{\tau}_1) = d$$

where

$$d = \prod_{t=1}^{m} |V(n_t, s_t+1)|^{-1} \sum_{\mathbf{x}_1,\ldots,\mathbf{x}_m} f(\mathbf{x}_m, \ldots, \mathbf{x}_1) = \prod_{t=1}^{m} |V(n_t, s_t+1)|^{-1} s_0.$$

Notice that for constructing $V(n_t, s_t+1), t = 1, \ldots, m$ one has to know of the number of variables (m) and their precision (n_t). Furthermore, in Theorem 2, s_t must be smaller than n_t, so it is impossible to use this method to construct a check set other than the trivial check set ($\mathcal{C}_2^{n_t}$) in cases where $s_t \ge n_t$. The following example illustrates the difficulty in using Th. 2 when the number of variables and their precision are unknown.

Example 1 *Consider three polynomials of order $M = 3$,*

$$f_1 \in K_3^{62}[\mathbf{x}_1],\ f_2 \in K_{3,3}^{31,31}[\mathbf{x}_2, \mathbf{x}_1],\ f_3 \in K_{1,\ldots1,3,3}^{1,\ldots,1,15,15}[\mathbf{x}_{34}, \ldots, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1].$$

According to Th. 2, the linear-checks for each polynomial can be constructed by defining a specific code for each one of its variables. For the cases where $s_t = 3 < n_t$ the linear-checks can be obtained by shortening the extended Hamming code [12]. The parameters of the shortened code are $[n = n_t, k = n_t - \lceil log_2(n_t) \rceil - 1, 4]$. For $s_t = n_t$ the trivial test set which contains all the binary vectors of length n_t must be used. That is,

1. *For the polynomial f_1 we choose the code $V_1(62, 4)$. The code is of dimension $62 - (6+1)$. The dual code $V^{\perp}$ is a code of dimension 7. In other words, $V^{\perp}$ is a check set of size 2^7 for f_1.*

2. *For the polynomial f_2 we choose two identical codes $V(31, 4)$. The codes are of dimension $31 - (5+1)$. The dual code $V^{\perp}$ is a Cartesian product of the two codes $V^{\perp}(31, 4)$, it is a code of length 62 and of dimension $2 \cdot (5+1) = 12$. Namely, the dual code $V^{\perp}$ is a check set of size 2^{12} for f_2.*

3. *The polynomial f_3 does not fulfill the requirements of Th. 2 since $s_i = n_i$ for $i > 2$. Nevertheless, it is possible to construct linear-checks for f_3 by using the trivial checks for the variables $\mathbf{x}_3, \mathbf{x}_4, \ldots, \mathbf{x}_{34}$, and two identical codes $V(15, 4)$ of length 15 and dimension $15 - (4+1)$ for the variables $\mathbf{x}_1$ and $\mathbf{x}_2$. The dual code $V^{\perp}$ is a Cartesian product of all the $32 + 2$ codes, it is a linear code of length 62 and dimension $32 + 2 \cdot (4+1)$. The dual code is a check set of size 2^{42} for f_3. The check set is defined by a generator matrix*

$$G = \begin{pmatrix} I_{32\times32} & 0 & 0 \\ 0 & G_{5\times15} & 0 \\ 0 & 0 & G_{5\times15} \end{pmatrix}$$

where I is the identity matrix and $G_{5\times15}$ is the (5×15) generator matrix of the dual code $V^{\perp}(15, 4)$. The structure of G depends on the order of the variable - if we change the order of the variables to $(\mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_{34}, \ldots, \mathbf{x}_4, \mathbf{x}_3)$ then we must change G accordingly.

Notice that each one of the three polynomials is associated with a different check set. The size of the check set and its structure depend on the number of variables, their precision and their order. Moreover, the size of check set grows as the number of variables increases. Yet, all the three polynomials can be represented as functions in 62 Boolean variables in K_3^{62}. As such, a Walsh based adaptive hardware that has 62 inputs may converge to each of them. Indeed, the worst case scenario (from the point of view of the test designer) happens, for example, when there are 31 variables each represented by two bits. In this case the size of the test is 2^{62}.

4 Universal Linear-checks

Since no information about the polynomial (except the fact that it is in K_M^n) is available, we are interested in a check set that will be suitable for any polynomial in K_M^n. There are two options: a) preparing in advance a check set for each polynomial in K_M^n and selecting the proper one in real time, and b) preparing a fixed check set that allows applying the test set without identifying the polynomial. The latter case is called *blind testing.* In blind testing, the check set is universal, it is applicable to any polynomial in K_M^n.

First, notice that it is impossible for an n_t-bit word to have Hamming weight of $n_t + 1$ and therefore,

Lemma 1 *If $s_t \geq n_t$ then $|V^\perp(n_t, s_t + 1)| = 2^{n_t}$.*

Consequently, the worst case scenario (in terms of test duration) happens when a WbAH system has converged to a polynomial for which $s_t \geq n_t$ for all t. For these polynomials , the size of the check set is 2^n.

The following lemma shows that the best case scenario happens when $m = 1$ (and thus $n_1 = n$).

Lemma 2 *Denote by $V_{(w)}^\perp$ the linear-checks for a polynomial of order M in w quantized variables in K_M^n. Then for $1 \leq m \leq n$ we have $|V_{(1)}^\perp| \leq |V_{(m)}^\perp|$.*

From Lemmas 1 and 2, in the worst case scenario the linear-checks are spanned by n (linearly independent) vectors, and in the best case scenario the linear-checks are spanned by $log_2(|V_{(1)}^\perp|)$ vectors of length n. In all other cases the linear-checks are determined by a Cartesian product of m codes of different lengths and dimensions. Indeed, it is impossible to aggregate the proper codes and construct the linear-checks without knowing the function in advance. Since it is impossible to extract information about the type of a polynomial from its spectral coefficients. The question is then, how to construct a non-trivial check set that can diagnose the health of the system without knowing the function it has converged to. The following theorem answers this question,

Theorem 3 *Let V be a subgroup of $\mathcal{C}_2^n$, and $V^\perp = \{\boldsymbol{\tau} | \boldsymbol{\tau} \in \mathcal{C}_2^n, \sum_{s=0}^{n-1} \boldsymbol{\tau}_s i_s = 0, \forall i \in V\}$. Then for any $f(\mathbf{x})$ defined on $\mathcal{C}_2^n$: $\sum_{\boldsymbol{\tau} \in V^\perp} f(\mathbf{x} \oplus \boldsymbol{\tau}) = \frac{1}{|V|} \sum_{i \in V} W_i(\mathbf{x}) s_i$.*

Notice that the set of test vectors, $\{\mathbf{x} \oplus \boldsymbol{\tau}\}_{\boldsymbol{\tau} \in V^\perp}$, to be applied is not predefined - it depends on the value of the inputs at the time the test is activated. This allows testing different propagation paths in the hardware.

Corollary 1 *Let $f \in K_M^n$ and let $V = V(n, \delta)$ be a maximal linear code of length n and minimum distance δ, and $V^\perp = V^\perp(n, \delta) = \{\boldsymbol{\tau} | \boldsymbol{\tau} \in \mathcal{C}_2^n, \sum_{s=0}^{n-1} \boldsymbol{\tau}_s i_s = 0, \forall i \in V\}$. Then,*

$$\sum_{\boldsymbol{\tau} \in V^\perp} f(\mathbf{x} \oplus \boldsymbol{\tau}) = \frac{1}{|V|} \left(s_0 + \sum_{i \in V, \delta \leq wt(i) \leq M} W_i(\mathbf{x}) s_i \right). \tag{3}$$

Note that if we choose $\delta = M + 1$, we get the test set $V_{(1)}^\perp$ (from Lemma 2). Recall that $V_{(1)}^\perp$ is considered as the *best-case-scenario* when applying Theorem 2 for constructing linear-checks for a given polynomial. Yet, the check set $V_{(1)}^\perp$ is optimal since it is the smallest set that covers all the scenarios including the *worst-case scenario* ($s_t \geq n_t$ for all t).

The complexity $N(\delta)$ of the linear-checks as derived from a code V of Hamming distance δ can be measured as the number of additions required for the computation of the two sums in Eq. 3. That is,

$$N(\delta) = |V^\perp(n, \delta)| + \sum_{i=\delta}^{M} \binom{n}{i}.$$

The following theorem says that for even values of M the best solution (in terms of computation time) is when $\delta = M + 1$.

Theorem 4 **(M even)** *Let M be an even integer. Define $p = \lceil log(n) \rceil$. Then $N(M+1) \leq N(M)$ for $M \leq 2^{\frac{p}{2}-1}$.*

Theorem 5 **(M odd)** *Let M be an odd integer. Define $p = \lceil log(n) \rceil$. Then $N(M+1) \leq N(M)$ for $M \leq 2^{\frac{p}{2}-2}$ and $M > \frac{p}{2}$.*

Remark: The fault detection capability of the linear-checks depends on the actual implementation of the circuit. In this paper we assume that implementation is based on the inverse Walsh transform. Initial experimental results indicate that a plurality of stuck-at faults in the combinatorial part of the system can be detected.

5 Conclusions

The paper deals with functional testing of Boolean systems whose functionality is unknown. The functional testing is performed off-line and is based on applying linear-checks. The suggested linear-checks are optimal for testing systems that have an acceptable representation as low order polynomials. In contrast to existing methods which require some information about the functionality of the system for constructing the tests, our method allows to construct a check set, which does not depend on: a) the actual functionality of the system, and b) the number of input variables and their precision.

References

[1] L.A.M. Bennett, "Synthesis of multi-output logic networks using spectral techniques," *IEE Proc. Comput. Digit. Tech.*, vol. 142, vol. 4, pp. 241-248, 1995.

[2] A. Burg, O. Keren, "Functional level embedded self-testing for Walsh transform based adaptive hardware," *IEEE International On-Line Testing Symposium*, 2012.

[3] N. Goel, M.G. Karpovsky, "Functional Testing of Computer Hardware Based on Minimization of Magnitude of Undetected Errors," *IEE Journal on Computer and Digital Techniques*, Vol. 129, No. 5, pp. 169-181, 1982.

[4] T.C. Hsiao and S.C. Seth, "An analysis of the use of Rademacher-Walsh spectrum in compact testing," *IEEE Trans. on Computers*, Vol. C-33, No. 10, pp. 934 - 937, 1984.

[5] M.G. Karpovsky, *Finite Orthogonal Series in the Design of Digital Devices*, Wiley, 1976.

[6] M.G. Karpovksy, "Error Detection for Polynomial Computations," IEE Journal on Computer and Digital Techniques, C-26, No. 6, pp. 523-528, 1980.

[7] M.G. Karpovsky, L. B. Levitin, "Universal Testing of Computer Hardware," *Spectral Techniques and Fault Detection*, M. G. Karpovsky, Editor, Academic Press, 1985.

[8] M.G. Karpovsky, R.S. Stankovic, J.T. Astola, *Spectral Logic and Its Applications for the Design of Digital Devices*, John Wiley&Sons, 2008.

[9] M.G. Karpovsky, E.A. Trachtenberg, "Linear Checking Equations and Error-Correcting Capability for Computation Channels," *Proc. IFIP*, pp. 619-624, 1977.

[10] O. Keren, "Adaptive Hardware Based on the Inverse WalshTransform," *Reed-Muller workshop*, Finland, pp. 21-26, 2011.

[11] I. Koren, *Computer Arithmetic Algorithms*, A. K. Peters, Natick, MA, 2002.

[12] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, 1977.

[13] D.M. Miller and J.C. Muzio, "Spectral fault signatures for single stuck-at faults in combinational networks," *IEEE Trans. on Computers*, Vol. C-33, pp. 765-769, 1984.

[14] C. Moraga, R. Heider, "Tutorial review on applications of the Walsh transform in switching theory," *TICSP Series*, no. 1, pp. 494-512, 1998.