A Quantum Algorithm for Finding Minimum Exclusive-Or Expressions. PRELIMINARY VERSION

M. Sampson, D. Voudouris, G. Papakonstantinou
Dept of Electrical and Computer Engineering
Division of Computer Science
Computing Systems Laboratory
National Technical University of Athens
157 80, Zografou Campus, Greece

Abstract

This paper presents a quantum algorithm for minimizing both Exlusive-or Sum of Complex Terms (ESCT) and Exlusive-or Sum of Products (ESOP) expressions. The proposed algorithm, QMin, takes advantage of the inherrent massive parallelism of quantum circuits. The ESCT expressions produced by QMin are presented in the related bibliography as an attractive architecture for implementing reversible and quantum circuits.

1 Introduction

Since the early 1980's Quantum Mechanics has begun to establish itself in the field of Computer Science. The promising aspects of Quantum Mechanics paved the way for an entirely novell approach of the computational model that is completely dominant nowadays. Even though the realization of quantum computers is in its early stages, the scientific community is already building up the software and the algorithms that will be used in such devices in the near future. The three algorithms that constitute, so far, the foundations of quantum algorithms are the Shor's [1], the Grover's [2] and the Quantum Fourier Transformation [3] algorithms. In particular the Shor's algorithm can find the periodicity of a function in polynomial time, providing exponential speedup which, in principle, renders RSA and related cryptography algorithms obsolete. Grover's algorithm is the optimal quantum searching algorithm even though it doesn't achieve the spectacular speedup of the previous one. Finally, the quantum Fourier Transform is actually the implementation of the Discrete Fourier Transform as a quantum circuit and has many applications in quantum algorithms as it provides the theoretical basis to the phase estimation procedure and is a key feature for many important quantum algorithms.

Another interesting problem which is intractable for a conventional computer is the mapping of an arbitrary switching function to a cellular architecture in an optimal way. Such architectures are the Maitra Cellular Architecture and the ESOP architecture which is in fact a subcase of the first one. Lately, the Maitra Cellular Architecture has attracted much scientific interest because it has been proved to be reversible and thus it may be useful for designing quantum circuits [4]. The problem of mapping an arbitrary function, in an optimal way, to such architectures, has been extensively researched in the past and non-heuristic solutions have been found for a small number of input variables [5, 6, 7, 8, 19, 18, 12]. Other heuristic approaches have been presented for more input variables [4, 9, 10, 18, 12, 17]. The above algorithms lead to optimal or near optimal circuits as far as the size is concerned, thus reducing the production cost of these circuits.

In this work, a quantum algorithm which deals with both the ESCT and ESOP minimization problem is presented. The proposed algorithm utilizes the quantum superposition to efficiently address these interesting problems. The produced ESCT expressions can be used for quantum circuit synthesis, since the underlying Maitra Cascades architecture can be directly mapped to quantum circuits. As it will be presented later, the proposed algorithm can also be used for ESOP (Exclusive-or sum of products) minimization, since an ESOP expression is a subcase of an ESCT expression.

In [15] a quantum algorithm for finding FPRM (Fixed Polarity Reed Muller) expressions with number of terms less than a specified threshold is described. It proposes the construction of a specialized quantum operator (oracle) which evaluates FPRM expressions. It then uses this oracle in conjunction with Grover's algorithm [2] in order to find the FPRM expressions with the desired characteristics.

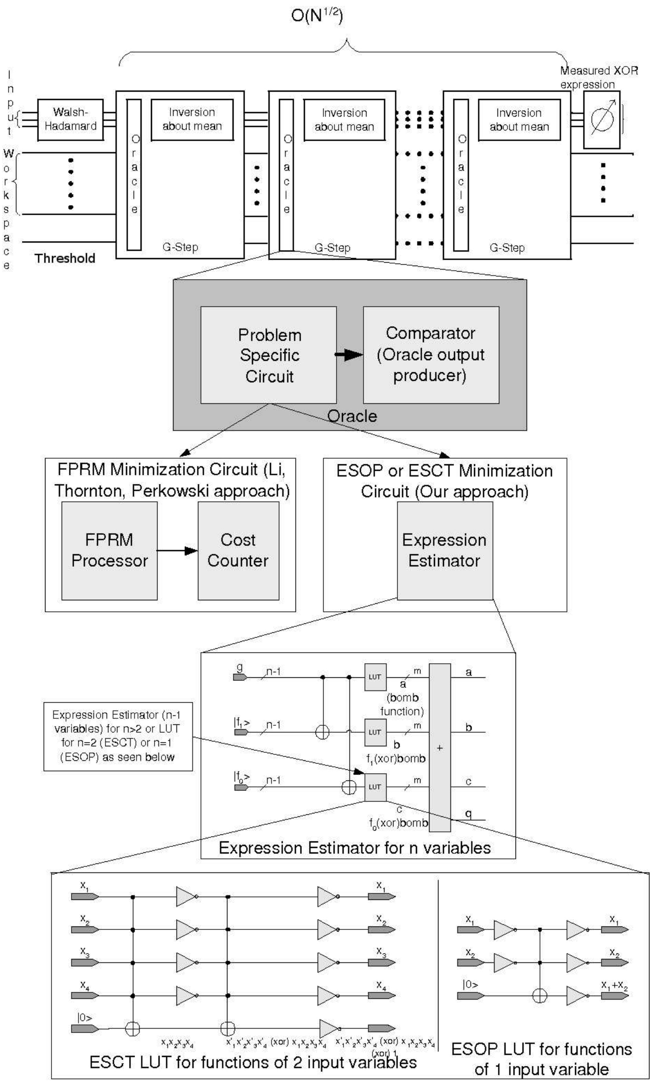

Figure 1. Algorithm hierarchy

Our proposed algorithm uses the same principle as this previous one but instead proposes a new oracle which evaluates either ESCT or ESOP expressions (which are a subset of ESCT expressions and a superset of FPRM expressions).Our implemented oracle has been simulated successfully, using the Fraunhofer Quantum Computing Simulator [14].

To the best of the authors' knowledge, this is the first quantum algorithm that addresses the interesting problems of ESCT and ESOP minimization.

2 Theoretical Background

2.1 Definitions

In this section we provide some background definitions.

Definition 1 *A complex Maitra term [11](complex term or Maitra term for simplicity) is recursively defined as follows:*

1. *Constant* 0 (1) *Boolean function is a Maitra term.*
2. *A literal is a Maitra term.*
3. *If* M_i *is a Maitra term,* a *is a literal, and* G *is an arbitrary two-variable Boolean function (Maitra cell), then* $M_{i+1} = G(a, M_i)$ *is a Maitra term.*

Additionally, it is required that each variable appears in each Maitra term only once.

As it was presented in [6], every complex term can be written as: $F_p(x_n, y)$, where y is a complex term, not depending on x_n, and $F_p(x_n, y)$ denotes a boolean function between x_n, y. In particular, $F_p(x_n, y)$ can be one of the following: $F_p(x_n, y) = \{F_1(x_n, y) = x_n + y, F_2(x_n, y) = \bar{x_n} + y, F_3(x_n, y) = \bar{x_n} y, F_4(x_n, y) = x_n y, F_5(x_n, y) = x_n \oplus y, F_6(x_n, y) = y\}$. Hence, $p = 1, 2, ..., 6$.

A wave cascade expression can be directly mapped to a special celllular architecture, called the reversible wave cascade cellular architecture.

Definition 2 *An ESCT (Exclusive-or Sum of Complex Terms) expression (some times also called reversible wave cascade or Maitra expression) for a switching function is an exlusive-OR sum of complex terms:*

$$Q = \sum_{i=1}^{m} \oplus M_i,$$

where M_i *are complex terms and* m *is their number inside the expression. The same variable ordering is used for every* M_i*. The size,* $s(Q)$*, of the expression* Q *is defined as the number of complex terms inside the expression.*

In the case where the Maitra cells of a complex term do not implement the logical OR and XOR functions, the complex term is also called a product term and an ESCT expression composed by product terms is reduced to an ESOP expression.

Definition 3 *A minimal (or exact) ESCT (ESOP) expression of a switching function* $f(x_1, \ldots, x_n)$ *of* n *variables is defined as the one which has fewer number of terms comparing to every other ESCT (or ESOP) expression for this function.*

Definition 4 *The ESCT (ESOP) weight $w(f)$ (or simply weight) of a switching function $f(x_1, \ldots, x_n)$ of n variables is defined as the number of complex terms (product terms) in a minimal ESCT (ESOP) expression of f.*

Every two-variable switching function has ESCT weight equal to 1 [11].

Definition 5 *The subfunctions f_1, f_0, f_2 of a switching function $f(\boldsymbol{x})$ ($\boldsymbol{x}$ is the vector of its variables) are defined as $f_1 = f(x_1, x_2, \ldots, x_{n-1}, 1)$, $f_0 = f(x_1, x_2, \ldots, x_{n-1}, 0)$, $f_2 = f_1 \oplus f_0$, regarding variable x_n.*

Definition 6 *The minterm representation (MT) of a switching function f with n variables is a bitvector of size 2^n where the i-th bit is 1 if the i-th minterm of f is 1.*

For the rest of this paper, the minterm representation of a switching function will be enclosed in brackets. It is very easy to prove that the upper half of the MT representation of a switching function corresponds to its f_1 subfunction, while the lower half corresponds to its f_0 subfunction.

2.2 ESOP and ESCT Minimization

The following theorems provide the necessary theoretical background in order to produce minimal expressions for an arbitrary switching function of n input variables.

Theorem 1 *Each minimal ESCT expression of a switching function f can always be written in one of the following compact forms:*

$$1st : f = F_p(x_n, y) \qquad (1)$$

(with one subfunction constant, ie. 0 or 1)

OR

$$2nd : f = F_p(x_n, y) \oplus F_q(x_n, z) \qquad (2)$$

OR

$$3rd : f = F_p(x_n, y) \oplus F_q(x_n, z) \oplus F_r(x_n, g) \qquad (3)$$

where the valid combinations of cell indices and their corresponding inputs have been presented in [6].

The proof of this theorem is presented in [6].

In [12, 19] it was proved that we can find a minimal ESOP expression for a function f of n input variables, by merging (xor summing) every possible function of $n - 1$ variables with its subfunctions f_0, f_1. This is because f can be written as: $f(x_1, \ldots, x_n) = \bar{x_n} V_1 \oplus x_n V_2 \oplus (\bar{x_n} \oplus x_n) V_3 == \bar{x_n}(V_1 \oplus V_3) \oplus x_n(V_2 \oplus V_3) = \bar{x_n} f_0 \oplus x_n f_1$. Hence, $V_1 = f_0 \oplus V_3$, $V_2 = f_1 \oplus V_3$ and $w(f) = w(V_1) + w(V_2) + w(V_3)$.

In the following theorem it is proved that an almost identical algorithm can be used for finding minimal ESCT expressions for an arbitrary switching function.

Theorem 2 *Every minimal ESCT expression for f of n variables can be found by merging every possible function of $n - 1$ variables with subfunctions f_0, f_1 of f.*

Proof. A minimal ESCT expression of f will be in one of the three compact forms of Theorem 1.

- If it is in the third compact form then according to Theorem 1 and the proof in [8], there must be at least another minimal ESCT expression of f in the following form: $F_3(x, Y) \oplus F_4(x, Z) \oplus F_6(x, G) = xY \oplus \bar{x}Z \oplus G$, where Y, Z, G are minimal ESCT expressions of functions y, z, g. It, also, holds: $f_1 = y \oplus g, f_0 = z \oplus g$. We obviously don't know y, z, g functions, but we do know the f_0, f_1 subfunctions. In order to find y, z, g functions, we can merge every possible g function (in essence we check every possible g function to find the appropriate one(s)) with f_0, f_1 in order to produce y and z functions (the same process as in ESOP minimization). The minimal expression will be: $f = xY \oplus \bar{x}Z \oplus G$, where Y, Z are minimal ESCT expressions of functions $y = g \oplus f_1, z = g \oplus f_0$. The minimal ESCT expressions of f are produced by those combinations of y, z, g that give the least number of complex terms and $w(f) = w(f_0 \oplus g) + w(f_1 \oplus g) + w(g)$.
- If it is in the second form then there must be at least another minimal ESCT expression of f in one of the forms corresponding to the Shannon or Davio expansions: $f = \bar{x} f_0 \oplus x f_1$ or $f = f_0 \oplus x f_2$ or $f = f_1 \oplus \bar{x} f_2$ [8]. We can find these forms by xor summing f_0, f_1 with the constant 0 function (Shannon expansion), f_1 subfunction (negative Davio expansion) or f_0 subfunction (positive Davio expansion). The minimal ESCT expressions will be: $f = x(f_1 \oplus 0) \oplus \bar{x}(f_0 \oplus 0) \oplus 0$ or $f = x(f_1 \oplus f_1) \oplus \bar{x}(f_0 \oplus f_1) \oplus f_1$ or $f = x(f_1 \oplus f_0) \oplus \bar{x}(f_0 \oplus f_0) \oplus f_0$.
- If it is in the first form then one of f_0, f_1, f_2 will be constant (0 or 1). The appropriate minimal ESCT expression can easily be found by xor summing constant function 0 with f_0, f_1. For instance if $f_2 = 1$ then the minimal ESCT expression will be: $f = x(f_0 \oplus 0) \oplus \bar{x}(f_1 \oplus 0) \oplus 0$ and since $f_2 = 1 \Leftrightarrow f_0 = \bar{f_1}$, it holds $f = x\bar{f_1} \oplus \bar{x} f_1 = x \oplus f_1 = F_5(x, f_1)$.

It is obvious that in every possible case (both for ESCT and ESOP), we can merge any possible g function (including the constant 0 function) with f_0, f_1 and we will acquire at least one minimal ESCT (ESOP) expression for function f in the form: $x(f_1 \oplus g) \oplus \bar{x}(f_0 \oplus g) \oplus g$. The only difference is that in the case of ESCT expressions we must stop the recursion at the 2-variable level (since every non constant 2-variable function has ESCT weight equal to 1), instead of the 1-variable level in the ESOP case. Moreover, the constant function 1 does not add to the ESCT weight, although it does so in the ESOP case [8].

2.3 Grover's Algorithm

In [2], L. Grover presented a quantum algorithm for finding a specific element in an unsorted database in $O(\sqrt(N))$ steps (N is the number of elements in the database). This result is much better than its conventional analogue ($O(N)$). The initial state of the database is the superposition of all possible N elements. This is accomplished using the Walsh-Hadamard gate. In each step, Grover's algorithm increases the amplitude of the marked states (the states of the elements we are searching for) and decreases those of the unmarked ones by $O(1/\sqrt(N))$. After $O(\sqrt(N))$ steps the probability of the marked states will be almost 1 and those of the unmarked states will be almost 0. At that point, we perform a measurement and find one of the elements we are searching for. The actual behavior of Grover's algorithm depends on a specific quantum operator called the Oracle. This operator decides which states are the marked ones and which are not.

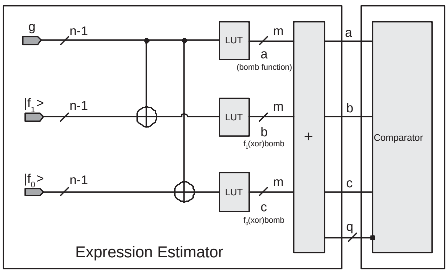

Figure 2. QMin-Oracle Circuit

In [15] Li, Thornton and Perkowski presented a quantum algorithm based on Grover's, that could find FPRM expressions for a specific function with number of terms less than a specific threshold. Their algorithm is actually Grover's algorithm using a special Oracle operator. In particular, this operator is composed of three distinct parts: the FPRM processor, the cost counter and the comparator (Fig. 1). The FPRM processor calculates all possible FPRM expressions for a specific function. The cost counter determines the size of each expression (number of coefficients in FPRM expression). Finally, the comparator checks if the determined size is less or equal to the specified threshold and outputs 1, otherwise 0. This number is the output of the oracle operator. In general, they proposed a generic platform for the development of quantum algorithms that is suitable for solving several intractable problems. All you have to do is to provide a problem-specific circuit to evaluate the cost function (cost counter).

In this work, we propose a different oracle implementation in order to find minimal ESOP or ESCT expressions for an arbitrary switching function. In our case, the oracle is modified in the same principle as before. It is composed of two distinct parts: The Expression-Estimator and the Comparator (Fig. 2). The first part calculates all possible ESOP or ESCT expressions and determines the number of terms (complex or product) in each one (cost function). The Comparator is the same as in [15]. Our proposed algorithm also detects the expressions that have number of terms (product or complex) less or equal to a certain threshold. In Fig. 1, the aforementioned generic framework and its variations for the FPRM and the ESOP/ESCT expressions are presented.

3 Algorithms

The proposed quantum algorithm (QMin) is based on Grover's algorithm [2]. We propose a special oracle (QMin-Oracle) as input for Grover's algorithm, which is based on Theorem 2.

In a conventional ESCT (or ESOP) minimization algorithm like XMin6 [8], the main computational overhead comes from the for-loop where the $(n-1)$-input variable function is merged with f's subfunctions in order to produce ESOP or ESCT expressions for the input function. From Theorem 2 it is obvious that some of them will be minimal. In QMin-Oracle all the iterations of this previous loop are performed in one step, utilizing the quantum superposition, inside Grover's oracle. This, enables a significant speedup, allowing to use the same brute-force technique as with XMin without developing any further minimization theory.

In contrast to the classical XMin algorithm, the merging process in QMin is performed using any possible $(n-1)$ variable function (since all these functions can be represented by a quantum register of $n-1$ qbits in superposition). Furthermore, in QMin, it is sufficient to use only one pair of subfunctions (here f_0, f_1) since we merge with every possible $(n-1)$ variable function (Theorem 2).

QMin stages are presented in Fig. 1. As it can be observed QMin is essentially Grover's algorithm with modified oracle.

Every oracle for Grover's algorithm should return 1 or 0 depending on the input. If the input is a marked state (an input that satisfies the search criteria) then it should return 1 otherwise it should return 0. QMin-Oracle receives two inputs, our input function and a threshold. It returns 1 if the produced expression has number of terms (product or complex) less or equal to Threshold. Otherwise it returns 0.

The QMin-Oracle is composed of two distinct components. The first one called the Expr-Estimator is the one that implements Theorem 2 and produces ESOP or ESCT expressions for our input function. The second one is the

Comparator which compares the number of terms (complex or product) of every produced expression with a given Threshold. It returns 1 if this number is less or equal to Threshold and 0 otherwise.

The Expr-Estimator component can be seen in Fig. 2. It is composed of three main buses. The first is initialized by the Walsh-Hadamard gates of Fig. 1 (they reside outside the Oracle operator) and stands for the g function of Theorem 2. The second and the third one are initialized as the minterm representation of subfunctions f_1 and f_0 of our input function, respectively. The CNOT gates produce the bitwise XOR sum of the g function (first bus) with f_1 and f_0, respectively, thus producing the $g \oplus f_1$ and $g \oplus f_0$ functions of Theorem 2.

The LUT operators that follow, can be considered as black boxes which produce the weight of their input function. One way to implement them, is to use the Expr-Estimator circuit recursively for $n-1, n-2, \ldots, 2, 1$ variables.

If QMin is designed to produce ESCT expressions, then our recursion stops at the 2-variable level, and the corresponding LUT circuit used can be seen at the bottom of Fig. 1 (leftmost part). This circuit implements the Boolean function $(x_1 + x_2 + x_3 + x_4) \oplus x_1x_2x_3x_4$, where x_1, x_2, x_3, x_4 are the bits corresponding to the MT formulation of the input function (not to be confused with the input variables of the function). It results to 0 when all its inputs are either 0 or 1 (constant functions have ESCT weight equal to 0) and 1 in all other cases. Obviously, its result corresponds to the weight of the function with MT formulation composed of bits x_1, x_2, x_3, x_4.

If QMin is designed to produce ESOP expressions, then the recursion stops at the 1-variable level and the LUT circuit appears at the bottom of Fig. 1 (rightmost part). It implements function $x_1 + x_2$, where x_1, x_2 are the bits corresponding to the MT formulation of the input function. This results to 0 only if x_1, x_2 are both 0 and 1 in all other cases and corresponds, accordingly, to the weight of the input function.

At this point the number of terms for each possible ESCT or ESOP expression of functions $g, g \oplus f_1, g \oplus f_0$ is calculated. According to Theorem 2, the number of terms in an expression of our initial function is their sum. Therefore, we use a quantum adder to perform this task. Quantum adders have been presented in the related bibliography [13, 15] and any of these can be used.

The other component of QMin-Oracle is the Comparator. The Comparator we use, has been presented in [15]. A 2-qubit Comparator can be seen in Fig. 3. It should be noted that the Comparator is used only once and not within the recursion performed by the Expression-Estimator.

As an example we present the QMIN-Oracle circuit for finding ESCT expressions with number of complex terms

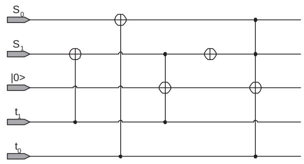

Figure 3. A 2-qubit Comparator, comparing s_0s_1 to t_0t_1, implementing function: $(s_1 \oplus t_1)t_1 \oplus \overline{(s_1 \oplus t_1)}(s_0 \oplus t_0)t_0$

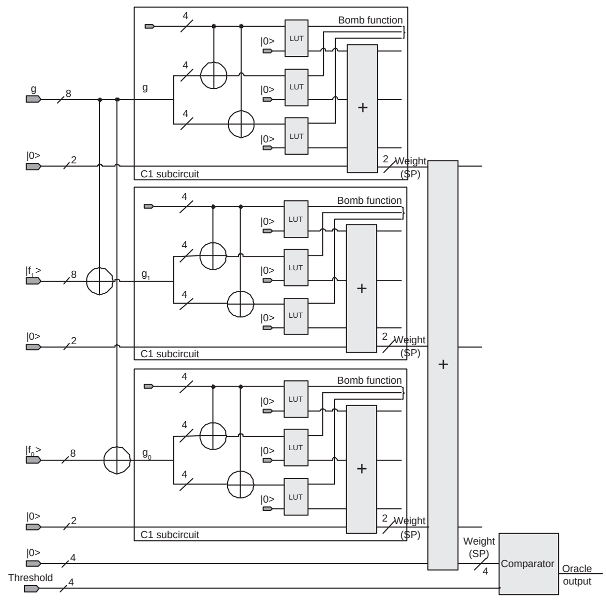

Figure 4. Four Variable Oracle Circuit

less or equal than a specified Threshold for switching functions of 4-input variables. This circuit can be seen in Fig. 4. It is noted that the QMin-Oracle circuit has been succesfully simulated using the Fraunhofer Quantum Computing Simulator [14] for the ESOP case and for functions of up to 2-input variables.

It must be noted that the q output of the QMin Oracle operator in Fig. 2 is the input to the Comparator operator. The a, b, c outputs correspond to the $g, g \oplus f_1, g \oplus f_0$ functions of Theorem 2 and are considered outputs of our quantum minimization circuit (Fig. 1).

The actual quantum minimization algorithm is, of course, Grover's algorithm, using the previously described QMin Oracle circuit as the specialized oracle operator. It takes as input (Fig. 1) an arbitrary switching function f and a Threshold. Its outputs are the a, b, c, q outputs of the QMin Oracle circuit. At the end of the execution, those expressions of f that have number of terms (either complex or product, depending on the type of expressions we are searching for) less or equal to the Threshold, will have probability almost 1 (marked states), while all the others will have probability almost 0 (unmarked states). Upon measuring one of its outputs (for instance the a output), all the outputs will collapse to those corresponding to one of the marked states. The weight of the output expression will be given by q, while the actual expression can be reconstructed from outputs a, b, c according to Theorem 2.

4 Conclusions and future work

In this work a quantum algorithm (QMin) for producing ESOP or ESCT expressions with number of terms (complex or product) less than a specified threshold is presented. QMin is a quantum algorithm that receives as input an arbitrary switching function and detects its ESOP or ESCT expressions with number of terms (complex or product) less than a specified threshold. It is obvious that by repeativelly executing QMin and updating the Threshold as necessary we can find minimal expressions for a specific function. An initial estimation for the Threshold can be obtained from conventional heuristic minimizers such as exorcism-mv2 [16] or QuiXor [18] (ESOP case) or EMin1 [7] (ESCT case).

Future work will focus on extending the algorithm in order to address the multi-output switching function minimization problem. Another interesting aspect would be to find a more efficient algorithm with complexity less than exponential.

5 Acknowledgments

This work has been funded by the project PENED 2003. This project is part of the OPERATIONAL PROGRAMME "COMPETITIVENESS" and is co-funded by the European Social Fund (75%) and National Resources (25%).

References

[1] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer", SIAM J. Computing 26, pp. 1484-1509 (1997).

[2] L.K. Grover, "A fast quantum mechanical algorithm for database search", Proc. 28th Ann. ACM Symp. on Theory of Comput., 212219, 1996.

[3] D. E. Knuth "The Art of Computer Programming", Vol. 2: Seminumerical Algorithms, Second ed., Addison-Wesley, 1981.

[4] A. Mishchenko, M. Perkowski, "Logic Synthesis of Reversible Wave Cascades",International Workshop on Logic And Synthesis 2002, New Orleans, Louisiana, June 4-7, 2002.

[5] G. Papakonstantinou, "Synthesis of cutpoint cellular arrays with exclusive-OR collector row", Electronic Letters, 13(1977).

[6] D. Voudouris, S. Stergiou, G. Papakonstantinou "Minimization of reversible wave cascades", IEICE Trans. on Fund., Vol E88-A, No. 4, pp. 1015-1023, 2005/04.

[7] D. Voudouris, G. Papakonstantinou "Maitra Cascade Minimization", 6th IWSBP, 2005, Freiberg (Sachsen), Germany.

[8] D. Voudouris, M. Sampson, G. Papakonstantinou "Exact ESCT Minimization for functions of up to six input variables", submitted for publication to Integration, The VLSI Journal, Elsevier (under minor revision).

[9] D. Voudouris, M. Kalathas, G. Papakonstantinou "Decomposition of Multi-output Boolean Functions", HERCMA 2005, Athens, Hellas.

[10] G. Lee "Logic synthesis for celullar architecture FPGA using BDD", ASP-DAC 97, pp 253-258 Jan 1997.

[11] K.K. Maitra "Cascaded switching networks of two-input flexible cells" IRE Trans. Electron. Comput., pp, 136-143, 1962.

[12] A. Gaidukov, "Algorithm to derive minimum esop for 6variable function", 5th IWSBP, September 2002.

[13] Phil Gossett, "Quantum Carry Save Arithmetic", quant-ph/980861 (1998)

[14] http://www.qc.fraunhofer.de/

[15] Lun Li, Mitch Thornton and Marek Perkowski "A Quantum CAD Accelerator based on Grover's algorithm for finding the minimum Fixed Polarity Reed-Muller form", ISMVL'06, Proc. of the ISMVL'06 vol. 00, pp. 33- 33, 17-20 May 2006.

[16] Song, N., Perkowski, M.A., "EXORCISM-MV-2: minimization of exclusive sum of products expressions for multiple-valued input incompletely specified functions", ISMVL1993, Proc. of ISMVL93, pp.132-137, 24-27 May 1993.

[17] A. Mishchenko, M. Perkowski "Fast Heuristic Minimization of Exclusive-Sums-of-Products", 5th International Reed-Muller Workshop, Starkville, Mississippi, August, 2001

[18] Stergiou S., Voudouris D., Papakonstantinou G., "Multiple-Value Exclusive-Or Sum-Of-Products Minimization Algorithms", IEICE Trans. on Fund., 2004, vol 87, part 5, pp. 1226-1234.

[19] T. Hirayama, Y. Nishitani, T. Sato "A Faster Algorithm of Minimizing AND-EXOR Expressions", IEICE Trans. on Fund., Vol E85-A, No. 12, pp. 2708-2714, 2002/12.