On Exploiting PSOP Decomposition for Quantum Synthesis

Anna Bernasconi
Dipartimento di Informatica
Università di Pisa, Italy
anna.bernasconi@unipi.it

Valentina Ciriani, Gianmarco Cuciniello, Asma Taheri Monfared
Dipartimento di Informatica
Università degli Studi di Milano, Italy
{valentina.ciriani, asma.taheri}@unimi.it

***Abstract*—The synthesis strategy for quantum oracles is based on a reversible logic synthesis and a quantum compilation step. In reversible logic synthesis it is important to obtain a compact reversible circuit in order to minimize the size of the final quantum circuit. Projected Sum Of Product, PSOP, decomposition is an EXOR based technique that can be applied to any Boolean function as a very fast pre-processing step for further minimizing the circuit area in standard logic synthesis. In this paper, we exploit PSOP decomposition in quantum synthesis. In particular, we describe a new technique for the quantum synthesis of PSOP decomposed functions. The experimental results validate the proposed pre-processing method in quantum synthesis, showing an interesting gain in area, within the same time limit.**

***Index Terms*—Circuit decomposition, reversible logic, quantum circuits**

I. Introduction

The recent technological improvement in quantum architectures has led to renewed and growing interest in quantum computing and in the design of secure cryptographic protocols. Therefore, the research in the field of quantum logic synthesis has attracted considerable new attention. In particular, many quantum algorithms, including Grover's search algorithm, usually require computing *oracles* [13], i.e., subroutines given as classical logic functions. The standard method for synthesizing quantum oracles generally consists of two steps: reversible logic synthesis and quantum compilation. This is due to the fact that, in general, the evolution of quantum systems is described by reversible unitary operators.

Recently, new techniques and tools have been proposed for quantum synthesis [10], [14], [18]. Moreover, several pre-processing methods have been proposed for further enabling reversible synthesis and quantum compilation. Indeed, such methods exploit structural regularities of the input function [1], [4]. Therefore, just "regular functions" can benefit from this pre-processing strategies (i.e., autosymmetric and D-Reducible functions). In this paper we propose a new pre-processing strategy that have the following characteristics:

1) The method can be exploited for *any* Boolean function (not just regular ones);
2) The method consists in a decomposition of the original function f and a re-composition after the quantum compilation;
3) The decomposition procedure has a linear-time complexity and the re-composition phase is constant in time.

These characteristics make the proposed method a possibly useful and fast pre-processing strategy before reversible synthesis and quantum compilation. The method is based on a structural non-disjoint decomposition of the input function called *Projected Sum of Products* (PSOP), which is an EXOR based non-disjoint decomposition. PSOP forms are a generalization of the standard Shannon decomposition. We consider this particular decomposition since it is EXOR based (i.e., the reconstruction has a very low quantum cost), and the decomposition process is vary fast (i.e., exhibits a linear time complexity). Figure 3 shows the standard quantum synthesis methods and the new proposed one.

The theoretical part of this paper describes the proposed pre-processing method and the reconstruction strategy. In particular, we show that after the PSOP decomposition and the quantum synthesis of the components, it is possible to reconstruct the original function f adding a constant number of Toffoli gates (2 or 3) that correspond to 8 or 12 T-gates [10].

Finally, we test the quantum synthesis of PSOP decomposed functions and we compare the results with the ones obtained by the classical quantum compilation. The experimental results show that the proposed pre-processing phase gives better results for the 61% of the benchmarks with an average gain of about 22% in terms of T-gates, using the XAG-based quantum compilation described in [10].

The paper is organized as follows: the preliminary concepts of projections of functions, PSOP forms, along with reversible circuits and quantum compilations, are outlined in Section II. Our proposed new pre-processing strategy for quantum reversible synthesis is presented in Section III. We discuss the evaluation of the proposed method and report our experiments on a set of benchmarks in Section IV. Finally, the conclusion of this work is given in Section V.

II. Preliminaries

A. Projections of Functions

PSOP decomposition, originally introduced for logic synthesis in CMOS technology, can be exploited to enable quantum compilation, as proposed and discussed in Section III. In this preliminary section, we review some basic concepts of Boolean space partitioning and we present the projections exploited in this particular decomposition.

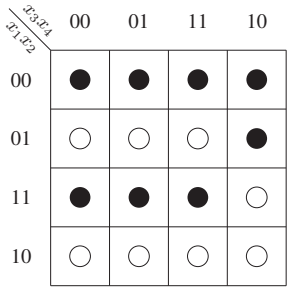

Fig. 1. The Boolean space $\{0,1\}^4$ partitioned into the two distinct sets $B_{x_1=p}$ (black points) and $B_{x_1\neq p}$ (white points), with $p = x_2 \cdot (\overline{x}_3 + x_4)$.

Let us consider the Boolean space with variables $x_1, x_2, \ldots, x_n$ and let x_i be one of these variables. Let p represent a function over a subset of variables excluding x_i, denoted by $\{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n\}$. The Boolean space $B = \{0,1\}^n$ can be partitioned into two distinct subsets: the set $B_{x_i=p}$, where x_i equals the function p, and the set $B_{x_i\neq p}$, where x_i is not equal to the function p. Formally, we have $B_{x_i=p} = \{(v_1, \ldots, v_n) \in \{0,1\}^n \mid v_i = p(v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n)\}$, and $B_{x_i\neq p} = \{(v_1, \ldots, v_n) \in \{0,1\}^n \mid v_i \neq p(v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n)\}$, respectively.

Example 1: Let us consider the function $p = x_2 \cdot (\overline{x}_3 + x_4)$ and the variable x_1 in the Boolean space $\{0,1\}^4$, which can be partitioned into two sets. The first set consists the subspace where $x_1 = p$ and the second one consists of the subspace where $x_1 \neq p$. Figure 1 depicts a Karnaugh map illustrating these two sets, the black points correspond to $B_{x_i=p} = \{0000, 0001, 0010, 0011, 0110, 1100, 1101, 1111\}$, while the white points correspond to $B_{x_i\neq p} = \{0100, 0101, 0111, 1000, 1001, 1010, 1011, 1110\}$.

It is noteworthy that the Boolean space divides evenly into these two sets, demonstrating the following general property [3]: When x_i is a Boolean variable and p is a function represented as $p : \{0,1\}^{n-1} \rightarrow \{0,1\}$ on variables $\{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n\}$, the sets $B_{x_i=p}$ and $B_{x_i\neq p}$ are such that:

1) $B_{x_i=p} \cup B_{x_i\neq p} = \{0,1\}^n$
2) $|B_{x_i=p}| = |B_{x_i\neq p}| = 2^{n-1}$
3) $B_{x_i=p} \cap B_{x_i\neq p} = \emptyset$

In the Boolean space $B = \{0,1\}^n$, the simplest partitioning occurs when p equals 1 (or 0). In this scenario, $B_{x_i=1}$ and $B_{x_i\neq 1}$ represent the subspaces of B where x_i equals 1 and x_i equals 0, respectively. The characteristic functions x_i and $\overline{x}_i$ can represent these subspaces [3]. Observe that $B_{x_i=1}$ and $B_{x_i\neq 1}$ represent the basic partitions of the *classical Shannon decomposition*

$$f = x_i f|_{x_i=1} + \overline{x}_i f|_{x_i\neq 1}$$

where $f|_{x_i=1}$ and $f|_{x_i\neq 1}$ denote the two cofactors obtained from f replacing x_i with 1 and 0, respectively.

In [6], [8], a Boolean functional decomposition method is presented, generalizing the classical Shannon decomposition: $f = (\overline{x}_i \oplus p) f|_{x_i=p} + (x_i \oplus p) f|_{x_i\neq p}$. This method projects the function f onto the two complementary subsets $B_{x_i=p}$ and $B_{x_i\neq p}$ of the Boolean space $B = \{0,1\}^n$. The expressions $\overline{x}_i \oplus p$ and $x_i \oplus p$ denote the characteristic functions of $B_{x_i=p}$ and $B_{x_i\neq p}$, respectively. It should be noted that the Shannon decomposition is a specific case of this partition, where $x_i \oplus 1$ equals $\overline{x}_i$ and $\overline{x}_i \oplus 1$ equals the variable x_i.

Example 2: Figure 2 illustrates the Karnaugh map of the function $f = \{0000, 0001, 0111, 1011, 1100, 1101, 1111\}$, in the right side. Consider $B_{x_1=p}$ and $B_{x_1\neq p}$ as two projecting sets with $p = x_2 \cdot (\overline{x}_3 + x_4)$. As shown in the left side of this figure, the function f can be projected onto the two spaces $B_{x_1=p}$ and $B_{x_1\neq p}$. The resulting projected functions depend on x_2, x_3, x_4 and can be represented by $f|_{x_1=p} = \{000, 001, 100, 101, 111\}$ and $f|_{x_1\neq p} = \{011, 111\}$, respectively.

It is important to point out that *Hamming distances* can change when points are projected onto different subspaces. As a result, they may be combined into larger terms and may reveal new implications not present in the original function. For instance, consider the two points in Example 2 represented by the minterms $\overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4$ and $x_1x_2\overline{x}_3\overline{x}_4$. We can notice that their Hamming distance is equal to 2. After the projection onto the space where $x_1 \neq p$, their Hamming distance is reduced to 1 as they become more similar; thus it is possible to merge them into the larger product $(\overline{x}_3)$ in $f|_{x_1\neq p}$.

B. PSOP forms

The PSOP decomposition and synthesis approach involves constructing a circuit for f by exploiting p and the two projected functions $f|_{x_i=p}$ and $f|_{x_i\neq p}$. The synthesis for the projected functions is simpler, compared to f, as they involve at least one fewer variable, leading often to more compact circuits. The functions p, $f|_{x_i=p}$ and $f|_{x_i\neq p}$ can be synthesized using any logic minimization methodologies, including SOP synthesis and also *quantum* synthesis. When represented as sums of products, they form the *Projected Sum of Products* form, abbreviated as PSOP(f) and defined as follows [3].

Definition 1: Let $f|_{x_i=p}$ and $f|_{x_i\neq p}$ indicate the projections of f onto $B_{x_i=p}$ and $B_{x_i\neq p}$, respectively. The *PSOP* of f with respect to p is expressed as

$$\text{PSOP}(f) = (\overline{x}_i \oplus p) f|_{x_1=p} + (x_i \oplus p) f|_{x_1\neq p}\,,$$

where p, $f|_{x_i=p}$, and $f|_{x_i\neq p}$ are expressed as SOP forms.

It is worth mentioning that in order to minimize the overall form, it is possible to further minimize the SOP for p as well as the two projected SOP forms $f|_{x_i=p}$ and $f|_{x_i\neq p}$ after projection.

Definition 2: Let $\tilde{p}$ be a minimal SOP form for the function p and let the *minimal SOP expressions* for the projections of f onto the sets $B_{x_i=p}$ and $B_{x_i\neq p}$ be represented by $\tilde{f}|_{x_i=p}$ and $\tilde{f}|_{x_i\neq p}$, respectively. The *minimal PSOP* of f with respect to p is expressed as:

$$\text{PSOP}(f) = (\overline{x}_i \oplus \tilde{p}) \tilde{f}|_{x_1=p} + (x_i \oplus \tilde{p}) \tilde{f}|_{x_1\neq p}\,.$$

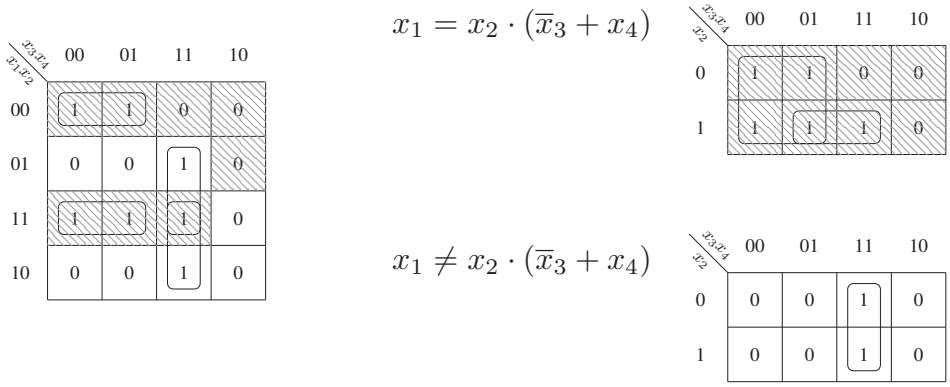

Fig. 2. Karnaugh maps of a function f (left) and its corresponding projections onto $f|_{x_1=p}$ and $f|_{x_1\neq p}$ (right), with $p = x_2 \cdot (\overline{x}_3 + x_4)$..

Example 3: Consider the function $f = \overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4 + \overline{x}_1\overline{x}_2\overline{x}_3x_4 + \overline{x}_1x_2x_3x_4 + x_1\overline{x}_2x_3x_4 + x_1x_2\overline{x}_3\overline{x}_4 + x_1x_2\overline{x}_3x_4 + x_1x_2x_3x_4$ described in Example 2. The minimal SOP form of this function is $\overline{x}_1\overline{x}_2\overline{x}_3 + x_2x_3x_4 + x_1x_2\overline{x}_3 + x_1x_3x_4$. As shown in Figure 2, the function f is projected onto the two sets. The minimal SOP forms for the sets $B_{x_1=p}$ and $B_{x_1\neq p}$, can be represented by $\tilde{f}_{x_1=p} = \overline{x}_3 + x_2x_4$ and $\tilde{f}|_{x_1\neq p} = x_3x_4$, respectively. As the minimal SOP form of p is $\tilde{p} = x_2\overline{x}_3 + x_2x_4$, the overall minimal PSOP form for f is then $(\overline{x}_1 \oplus (x_2\overline{x}_3 + x_2x_4))(\overline{x}_3 + x_2x_4) + (x_1 \oplus (x_2\overline{x}_3 + x_2x_4))(x_3x_4)$.

Another useful form is the *Pr-SOP* for the function f, which is also known as the *PSOP with remainder* [3]. It includes a *remainder*, containing all products in the SOP expression of f that intersect both projection sets. These products are called *crossing products*, whereas products entirely included in one of the two projection sets are called *non-crossing products*.

Definition 3: Let $f|_{x_i=p}$ and $f|_{x_i\neq p}$ denote the projections of all non-crossing products in a SOP representation of f, and let r denote the sum of all crossing products. The *Pr-SOP* of f with respect to p is expressed as:

$$\text{Pr-SOP}(f) = (\overline{x}_i \oplus p)f|_{x_i=p} + (x_i \oplus p)f|_{x_i\neq p} + r\,.$$

Minimizing all SOP expressions, we derive a *minimal PSOP with remainder*.

Definition 4: Let $\tilde{p}$ and $\tilde{r}$ be minimal SOPs form for the function p and the remainder r, respectively. Let the minimal SOP expressions for the projections of all non-crossing products of f onto $B_{x_i=p}$ and $B_{x_i\neq p}$ be represented by $\tilde{f}|_{x_i=p}$ and $\tilde{f}|_{x_i\neq p}$, respectively. The *minimal Pr-SOP* of f with respect to p is expressed as:

$$\text{Pr-SOP}(f) = (\overline{x}_i \oplus \tilde{p})\tilde{f}|_{x_1=p} + (x_i \oplus \tilde{p})\tilde{f}|_{x_1\neq p} + \tilde{r}\,.$$

Algorithms and heuristic methods for minimizing PSOP expressions have been proposed and analyzed in [3], [5].

C. Reversible Circuits and Quantum Compilation

Reversible circuits have one-to-one correspondence between their inputs and outputs, ensuring that the number of outputs is always equal to the number of inputs. In order to assess the efficiency of such circuits, metrics like *the number of ancilla inputs and garbage outputs* are crucial. The number of ancilla inputs refers to the number of additional input bits required to make logic gate irreversibles. The number of garbage outputs signifies the outputs generated to maintain one-to-one mappings but contain unimportant values. Decreasing these important features enhances the efficiency of designing reversible circuit.

Reversible circuits are normally based on *Mixed-polarity Multiple Control (MPMC) Toffoli gates*. All functions can be implemented with these reversible gates. In Figure 6, the realization of three *MPMC Toffoli gates* are illustrated, the notation $\oplus$ indicates the target line, while the notations $\bullet$ and $\circ$ denote positive and negative control connections, respectively. This gate is known as a *NOT gate* when there are no control connections. It is classified as a *Controlled-NOT (CNOT) gate* when there is only one positive connection, and as a *Multiple-Control Toffoli gate* when there are only positive connections [13].

The synthesis of quantum circuits begins with the design of reversible circuits. An additional quantum compilation step is then required to transform reversible circuits, which utilize gates like *MPMC Toffoli gates*, into quantum circuits with functionally equivalent gates.

This process involves decomposing each reversible gate into elementary quantum gates, based on *standard quantum gate libraries* [9], [12]. Adding a quantum compilation step to a reversible circuit with *MPMC Toffoli gates* can transform it into a functionally equivalent quantum circuit that can be implemented on quantum hardware. A quantum circuit as a result of this transformation maintains the same logical operations as the original reversible circuit while using quantum gates.

In this work, this mapping will be based on the *Clifford+T library*. It includes *Pauli, Hadamard, and CNOT gates* as well as *T-gate* [13]. Since the T-gate is considered as the most expensive quantum gate, the cost efficiency of quantum circuits is evaluated in terms of the number of T-gates required.

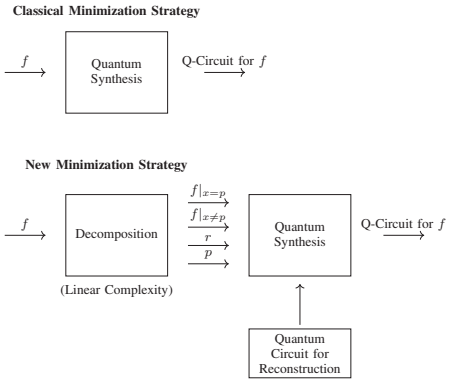

Fig. 3. Classical and new minimization strategies *without* SOP synthesis.

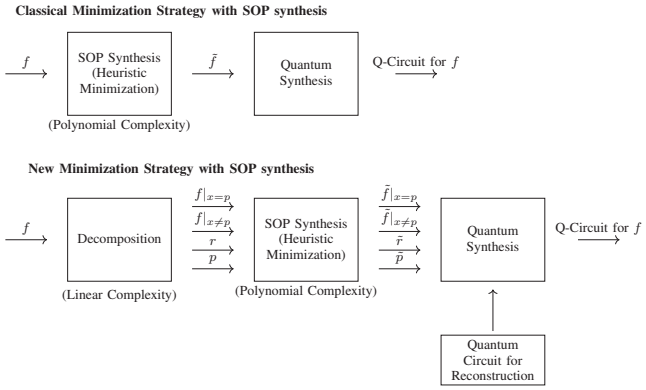

Fig. 4. Classical and new minimization strategies *with* SOP synthesis.

The classical cost in terms of *T-gates* for the realization of a 2-*controlled MPMC Toffoli gates* is 4, in accordance with the algorithm described in [10].

A detailed information on reversible circuits as well as an overview of efficient quantum compilation methods can be found in [10], [18].

III. Quantum circuits synthesis based on PSOP decomposition

In this section, we describe how the PSOP decomposition of a Boolean function f can be exploited to ease its quantum compilation. More precisely, we show how to combine quantum circuits for the two projected functions $f_{|x_i=p}$ and $f_{|x_i\neq p}$, the function p, and the remainder r, if present, in order to derive a quantum circuit for the original function f, following the strategies depicted schematically in Figure 3. Potential benefits of this approach are a reduced compilation time, and a final quantum circuit of reduced area with respect to the quantum circuit derived compiling directly the function f without leveraging its PSOP decomposed forms.

As already observed, this new quantum compilation strategy can be applied to any Boolean function, after the fast PSOP decomposition step, whose cost is linear in the initial SOP representation of the target function.

Let f be a Boolean function depending on n binary variables, and let PSOP(f) and Pr-SOP(f) denote its PSOP forms, without and with remainder r:

$$\text{PSOP}(f) = (\overline{x}_i \oplus p) f_{|x_i=p} + (x_i \oplus p) f_{|x_i\neq p}\,,$$
$$\text{Pr-SOP}(f) = (\overline{x}_i \oplus p) f_{|x_i=p} + (x_i \oplus p) f_{|x_i\neq p} + r\,,$$

where x_i is one input variable, p is a function that does not depend on x_i, $f_{|x_i=p}$ and $f_{|x_i\neq p}$ are the two projections of the SOP expression of f, and r is the remainder. Recall that both forms can be derived in linear time.

Before the quantum synthesis step, a heuristic SOP minimization step could be performed to facilitate quantum compilation and possibly derive more compact circuits, as shown in Figure 4. This step can be performed by applying polynomial time SOP heuristics on all components of the PSOP(f) and Pr-SOP(f) expressions, which are generally smaller functions, that depend on fewer variables and contain fewer minterms than the target function f. Notice that a similar step in the standard quantum compilation flow, not based on decomposition, would require the more costly heuristic SOP minimization of the whole function f. This preliminary minimization is not mandatory and can be avoided in case of large benchmarks, whose SOP minimization could result too time demanding.

After the optional SOP minimization step, quantum compilation is applied independently onto the subfunctions p, $f_{|x_i=p}$, $f_{|x_i\neq p}$, and the remainder r (if present).

Finally, we derive a quantum circuit for the overall function f using the quantum circuits for p, $f_{|x_i=p}$, $f_{|x_i\neq p}$, and the remainder r as building blocks, as shown in Figures 5 and 6.

Before describing how to derive a quantum circuit for a function f from PSOP decomposition, we state and prove a proposition that allows to ease the reconstruction strategy.

Proposition 1: Let f be a Boolean function depending on n binary variables, and let PSOP(f) and Pr-SOP(f) be its PSOP decomposition without and with remainder, respectively. The disjunction between the first two terms in both algebraic expressions can be replaced with an Exclusive Or:

$$\text{PSOP}(f) = (\overline{x}_i \oplus p) f_{|x_i=p} \oplus (x_i \oplus p) f_{|x_i\neq p}\,,$$
$$\text{Pr-SOP}(f) = \big((\overline{x}_i \oplus p) f_{|x_i=p} \oplus (x_i \oplus p) f_{|x_i\neq p}\big) + r\,.$$

Proof. Observe that the first two terms in both PSOP(f) and Pr-SOP(f) represent disjoint sets of points. Indeed, the two subspaces $B_{x_i=p}$ and $B_{x_i\neq p}$ do not intersect, and the product of their characteristic functions $(\overline{x}_i \oplus p)$ and $(x_i \oplus p)$ is the zero function. This immediately implies that the disjunction can be replaced with an exclusive OR. ■

This result is important for the reconstruction procedure since an EXOR can be easily implemented in a quantum circuit using a CNOT instead of a Toffoli gate.

We now describe the reconstruction procedure of a quantum circuit for f, considering first the case of PSOP decomposition without remainder.

The overall quantum circuit for f in this case is obtained concatenating the two quantum subcircuits for the projections, that depend on all variables but x_i, with the quantum circuit for $(x_i \oplus p)$, possibly depending on all input variables. The

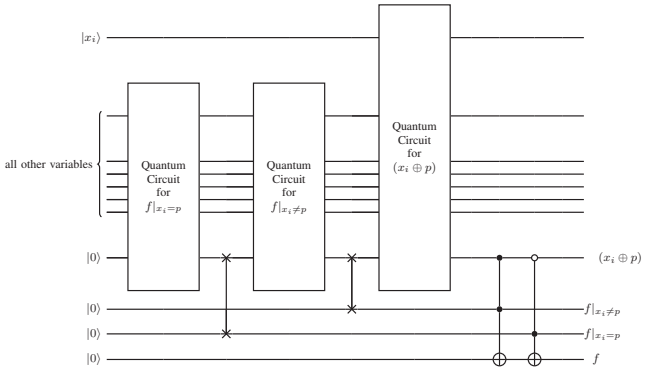

Fig. 5. Quantum circuit based on PSOP without remainder.

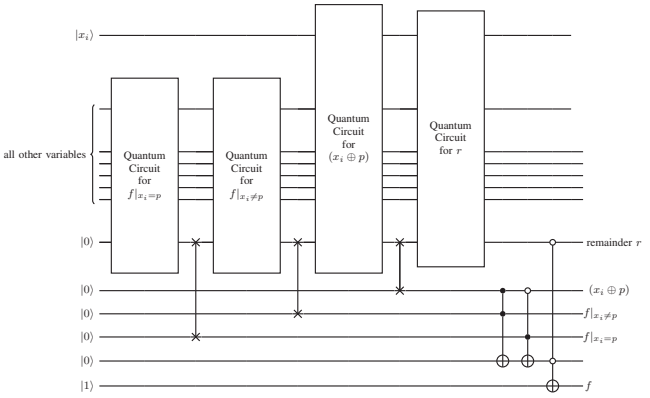

Fig. 6. Quantum circuit based on PSOP with remainder.

quantum circuit for $(x_i \oplus p)$ can be derived inserting a CNOT, controlled by x_i, on the output line of a quantum circuit for p. Note that four additional lines (and therefore four new qubits) are needed: one for $f_{|x_i=p}$, one for $f_{|x_i\neq p}$, one for $(x_i \oplus p)$ and finally one output line for f. The overall circuit structure is shown in Figure 5, where two swap gates are used to bring the qubits for the intermediate results closer to the corresponding subcircuits. Eventually, two Toffoli gates are inserted for computing the AND between the projections and the corresponding subspaces, one described by the subcircuit for $(x_i \oplus p)$ and the other by its complement. Both Toffoli gates act on the output line for f, thanks to the fact that the OR operator in the PSOP expression has been replaced with an EXOR. The overall methodology is summarized in the algorithm in Figure 7, and its cost in terms of elementary quantum T-gates is discussed in the following proposition.

Proposition 2: The number of T-gates required to synthesize the PSOP-based quantum circuit for f is given by the overall number of T-gates occurring in the subcircuits for $f_{|x_i=p}$, $f_{|x_i\neq p}$, and $(x_i \oplus p)$, plus 8 additional T-gates.

Proof. Observe from Figure 5 that the three quantum subcircuits for $f_{|x_i=p}$, $f_{|x_i\neq p}$, and $(x_i \oplus p)$ are combined using only two additional swap gates and two Toffoli gates. Since swap gates are implemented using CNOTs, only 8 additional T-gates are required, four for each Toffoli gate [10]. ■

Figure 6 shows the circuit for f based on the PSOP decomposition with remainder. As already noted in Proposition 1, the first disjunction can be replaced by an EXOR. Moreover, using De Morgan's laws, we can replace the remaining OR with a NAND. Thus the form becomes

$$\text{Pr-SOP}(f) = \overline{\left((\overline{x}_i \oplus p) f_{|x_i=p} \oplus (x_i \oplus p) f_{|x_i\neq p}\right)} \wedge \overline{r}$$

The overall quantum circuit for f is thus obtained concatenating the subcircuits for the projections, for the characteristic function $(x_i \oplus p)$ of the projection subspace, and for the remainder r, possibly depending on all input variables.

This time, six additional lines are used: two for $f_{|x_i=p}$ and $f_{|x_i\neq p}$, one for $(x_i \oplus p)$, one for the remainder, one for storing the intermediate result $(\overline{x}_i \oplus p) f_{|x_i=p} \oplus (x_i \oplus p) f_{|x_i\neq p}$, and one as output line for f. As before, swap gates are used to bring the qubits for the intermediate results closer to the corresponding subcircuits.

Two Toffoli gates, both acting on the same line, are then used for computing the EXOR of the products between the projections and the corresponding subspaces. A third Toffoli gate on the output line for f, inizialized with a qubit in state $|1\rangle$, is finally used to compute the NAND between the complement of the EXOR of the two products on the second to last line, and the complement of the remainder r.

The overall methodology, summarized in the algorithm in Figure 8, requires a constant number of additional T-gates for combining the four quantum subcircuits, as stated and proved in the following proposition.

Proposition 3: The number of T-gates required by the quantum circuit based on PSOP decomposition with remainder is given by the overall number of T-gates occurring in the subcircuits for $f_{|x_i=p}$, $f_{|x_i\neq p}$, $(x_i \oplus p)$, and r, plus 12 additional T-gates.

Proof. Observe from Figure 6 that the four quantum subcircuits for $f_{|x_i=p}$, $f_{|x_i\neq p}$, $(x_i \oplus p)$, and r are combined using three additional swap gates, implemented using CNOT gates only, and three Toffoli gates. Thus, only 12 additional T-gates are required, four for each Toffoli gate [10]. ■

The overall computational cost of the proposed approach includes the cost of the projections (linear in the initial SOP of f), the cost of the optional heuristic SOP minimization of $f_{|x_i=p}$, $f_{|x_i\neq p}$, p, and r (polynomial), the cost of their quantum compilation, and the (constant) cost for combining the quantum subcircuits into a quantum circuit for f.

The cost of the standard quantum compilation would include the cost of the optional heuristic SOP minimization of f and the cost of its quantum compilation.

IV. Experimental Results

In this section we evaluate the effectiveness of the proposed method for the quantum synthesis of PSOP-decomposed functions. We, then, present the computational results achieved by constructing PSOP expressions for Boolean functions, and comparing these expressions to their standard quantum synthesis forms. In order to assess reversible circuits derived

INPUT
f /* Function in SOP form depending on n variables $\{x_1, \ldots, x_n\}$ */
x_i /* An input variable */
p /* Function in SOP form depending on all input variables, but x_i */
$f_{|x_i \neq p}$ /* Projection of f onto the subspace $(x_i \oplus p)$ */
$f_{|x_i = p}$ /* Projection of f onto the subspace $(\overline{x}_i \oplus p)$ */

OUTPUT
Q /* Quantum circuit for f */

OPTIONAL: Heuristic SOP minimization of p, $f_{|x_i \neq p}$, $f_{|x_i = p}$;
$Q_{f_{\neq}}$ = QuantumSynthesis($f_{|x_i \neq p}$);
$Q_{f_{=}}$ = QuantumSynthesis($f_{|x_i = p}$);
Q_p = QuantumSynthesis($x_i \oplus p$);
Q = Toffoli($Q_{f_{\neq}}$, Q_p) $\oplus$ Toffoli($Q_{f_{=}}$, $\overline{Q}_p$);
return Q

Fig. 7. Quantum synthesis based on PSOP decomposition without remainder.

INPUT
f /* Function in SOP form depending on n variables $\{x_1, \ldots, x_n\}$ */
x_i /* An input variable */
p /* Function in SOP form depending on all input variables, but x_i */
$f_{|x_i \neq p}$ /* Projection of the non-crossing products of f onto the subspace $(x_i \oplus p)$*/
$f_{|x_i = p}$ /* Projection of the non-crossing products of f onto the subspace $(\overline{x}_i \oplus p)$*/
r /* Sum (OR) of the crossing products of f */

OUTPUT
Q /* Quantum circuit for f */

OPTIONAL: Heuristic SOP minimization of p, $f_{|x_i \neq p}$, $f_{|x_i = p}$, r;
$Q_{f_{\neq}}$ = QuantumSynthesis($f_{|x_i \neq p}$);
$Q_{f_{=}}$ = QuantumSynthesis($f_{|x_i = p}$);
Q_p = QuantumSynthesis($x_i \oplus p$);
Q_r = QuantumSynthesis(r);
Q_1 = Toffoli($Q_{f_{\neq}}$, Q_p) $\oplus$ Toffoli($Q_{f_{=}}$, $\overline{Q}_p$);
Q = 1 $\oplus$ Toffoli($\overline{Q_r}$, $\overline{Q_1}$);
return Q

Fig. 8. Quantum synthesis based on PSOP decomposition with remainder.

from PSOP decomposition and compare them with standard synthesis derived circuits, we have measured their number of qubits and also evaluated their cost in terms of elementary quantum T-gates. Specifically, we mapped each MPMC Toffoli gate into elementary quantum gates. This mapping was performed based on the Clifford+T library and the algorithm detailed in [10]. Since the T-gate is considered the most expensive gate in the library, usually the cost of a Toffoli gate is expressed in the number of T-gates needed for its realization. For this reason we report the number of T-gates in the tables.

All computational experiments have been run on a Intel i7-8550U CPU of 1.80GHz with 16GB of RAM. The benchmarks utilized are classical benchmarks in PLA form (classical Espresso and LGSynth'89 benchmark suite [19]). This choice is due to the fact that the computation of the function p described in [2] derives from a statistical analysis of the initial SOP (or PLA) form. The benchmarks in other classical sets (such as EPFL benchmark suite [16], [17]) are, unfortunately, not given in PLA form. We further discuss this point in the concluding section. As representative indicators of our experiments, we report only a significant subset of the functions.

The experiments has been conduct using the SOP minimization as described in the strategy depicted in Figure 4, using ESPRESSO [7] in the heuristic mode for the SOP synthesis. The experimental results are obtained by applying the XAG-based quantum compilation heuristic proposed in [11]. In particular, we are interested in evaluating experimentally whether this recent technique could benefit from the PSOP decomposition of the target function.

The decomposition phase is extremely fast, coherently with the linear time complexity of the corresponding algorithm [2]. Therefore, the computational times of the standard minimization and the decomposed one are extremely similar. For this reason the comparison of computational times is not interesting, and we do not report them in the tables.

In Table I, the names of a significant set of benchmarks, included in our experiments, are listed in the first column. The following four columns provide details on the number of T-gates, which determine the cost, and the number of qubits required for the quantum circuits obtained from standard synthesis and PSOP expressions of the benchmarks. As can be seen, we investigate three scenarios for PSOP expressions: the projection of f with respect to p is first explored as an AND of two variables (PSOP with AND), secondly as a simple Boolean variable (PSOP with variable), and lastly as an EXOR of two variables (PSOP with EXOR). In each scenario, we examine PSOP expressions both with and without remainder.

As shown in Table I, it is clear that some benchmarks experience significant advantages from PSOP expressions in terms of the number of T-gates and the number of qubits compared to standard synthesis. For instance, the benchmarks *rd73* and *sym10* achieve a significant reduction in T-gates and qubits when utilizing PSOP with EXOR (with remainder) and PSOP with EXOR (without remainder), respectively. Specifically, *rd73* shows a 49% reduction in T-gates and a 45% reduction in qubits, while *sym10* shows a 47% reduction in T-gates and an 46% reduction in qubits. However, in some cases, standard synthesis results in circuits with fewer T-gates and qubits compared with PSOP-based synthesis. For example, the *addm4* benchmark.

Overall, we can note that the best strategy seems to be the one where p is a single variable (with or without remainder). Moreover, the one that uses p as an AND gate is less useful. This is probably due to the fact that an AND gate has an expensive (in terms of T-gates) quantum representation. Meanwhile, the single variable or the EXOR gates require less expensive quantum gates.

Table II reports a subset of all the benchmarks used in our experiments. The first column lists the name of each benchmark. The next group of 2 columns detail the cost and the number of qubits for the the quantum circuits derived from standard synthesis and the best PSOP expressions of the benchmarks. Finally, the last column reports the gain in the number of T-gates.

According to the results shown in Table II, it is evident that some benchmarks benefit greatly from the proposed strategy. For instance, the benchmarks *newapla* and *newxcpla1* achieve a 71% and 75% reduction in T-gates, respectively. However, the gain is much less significant for some benchmarks, such as *in0* and *in2*. In some cases, the best PSOP strategy results in circuits with a higher number of T-gates, for example *adr4*

TABLE I
COMPARISON BETWEEN THE COMPILATION HEURISTIC PROPOSED IN [11] (STANDARD SYNTHESIS) APPLIED AFTER ESPRESSO IN THE HEURISTIC MODE, AND THE PROPOSED STRATEGY WITH DIFFERENT OPTIONS OF p WITH AND WITHOUT REMAINDER.

			PSOP with AND				**PSOP with variable**				**PSOP with EXOR**			
	Standard synthesis		**Without remainder**		**With remainder**		**Without remainder**		**With remainder**		**Without remainder**		**With remainder**	
Benchmark	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**
addm4	**1680**	429	2048	521	2156	548	2240	569	2244	570	2352	597	2352	597
adr4	**108**	35	264	74	120	38	216	62	216	62	352	96	440	118
amd	1244	325	**1096**	288	1128	296	1124	295	1132	297	1140	299	1140	299
apla	**404**	111	776	204	824	216	724	191	748	197	616	164	632	168
b3	1220	338	1320	363	**1200**	333	1304	359	1340	368	1272	351	1272	351
b10	1520	396	**1424**	372	1428	373	1436	375	1468	383	1552	404	1524	397
b12	280	85	344	101	260	80	260	80	**236**	74	408	117	256	79
bench	**228**	63	280	76	296	80	264	72	260	71	332	89	332	89
br1	504	138	580	157	496	136	**444**	123	504	138	524	143	524	143
br2	**348**	99	**348**	99	388	109	360	102	368	104	432	120	436	121
co14	192	62	244	75	224	70	192	62	192	62	**172**	57	180	59
dc2	**328**	90	424	114	364	100	**328**	90	**328**	91	424	114	424	115
exp	**1132**	292	1280	329	1284	330	1192	307	1208	311	1208	311	1272	327
f51m	454	121	412	111	**400**	108	508	135	508	135	420	113	416	112
fout	**820**	211	932	239	928	238	916	235	916	235	900	231	900	231
gary	1716	444	1792	463	1488	387	1612	418	1592	413	1692	438	**1440**	375
in0	1720	445	**1656**	429	1668	432	1712	443	1708	442	1720	445	1744	451
in2	1352	357	1632	427	1340	354	**1316**	348	1352	357	1328	351	1332	352
in3	1284	356	**1164**	326	1256	349	1224	341	1272	353	1280	355	1240	345
in4	1344	368	1384	378	1288	355	1372	375	1380	377	1344	368	**1320**	363
in5	1216	328	1160	314	1072	293	1052	287	**956**	264	1168	316	1088	297
in7	480	146	536	160	348	114	592	174	332	110	432	134	**316**	106
inc	**364**	98	444	118	444	118	380	102	384	103	388	104	388	104
m3	1024	264	936	242	948	245	1008	260	1052	271	**888**	230	912	236
m4	2128	540	1692	431	**1592**	406	1680	428	1836	467	1944	494	1932	491
max128	1392	356	**1196**	307	1232	316	1220	313	1260	323	1496	382	1496	382
mlp4	**1264**	324	1336	342	1308	335	1388	355	1388	355	1316	337	1316	337
newapla	136	46	196	61	72	31	140	47	**40**	23	200	62	76	32
newcpla1	336	93	464	125	248	72	280	79	320	90	364	100	**260**	75
newcpla2	228	64	216	61	**148**	45	236	66	168	49	240	67	152	46
newxcpla1	508	136	528	141	184	56	368	101	**128**	42	584	155	184	56
p3	632	167	736	193	720	189	**584**	155	592	157	628	166	620	164
p82	**292**	78	328	87	320	85	320	85	328	87	368	97	372	98
rckl	520	162	864	248	568	175	544	168	**296**	107	528	164	568	175
rd73	352	95	344	93	344	93	388	104	312	85	300	82	**180**	52
root	516	137	520	138	524	139	**452**	121	**452**	121	468	125	**452**	121
spla	**2536**	650	2720	696	2828	723	3356	855	3464	882	3308	843	3484	887
sqr6	404	108	408	109	416	111	**392**	105	**392**	105	440	117	428	114
sym10	1420	365	1080	280	1080	280	804	211	804	211	**748**	197	756	199
t1	**568**	163	792	219	792	219	572	164	592	169	816	225	816	225
t3	252	75	244	73	**228**	69	276	81	272	80	320	92	272	80
tms	744	194	828	215	840	218	704	184	744	194	**680**	178	692	181
vg2	444	136	440	135	360	116	512	153	364	116	348	112	**292**	99
x6dn	1112	317	1224	345	1180	334	**1080**	309	1092	312	1248	351	1308	366
x9dn	408	129	428	134	348	115	436	136	348	115	320	107	**256**	92
Z5xp1	456	121	380	102	392	105	512	135	520	137	356	96	**352**	95
Z9sym	828	216	804	210	788	206	692	182	692	182	**680**	179	**680**	179

and *apla*. Overall, the T cost of the best PSOP-based quantum circuit is significantly lower than that of circuits derived from standard synthesis.

It is also crucial to minimize the number of qubits in quantum circuit design. As can be observed in Table II, it is clear that the best PSOP strategy has a significant effect on some benchmarks in terms of the number of qubits. For example, the benchmark *newapla* and *newxcpla1* experiences more that 50% reduction in qubit numbers. However, the improvement is much smaller for some benchmarks like *b3* and *sqr6*. In some instances, the best PSOP strategy leads to circuits with a higher number of qubits, such as in the case of *adr4*. In general, we can see that the number of qubits in quantum circuits based on the best PSOP is notably fewer compared to the circuits obtained through standard synthesis.

In summary, we have that the proposed strategy gives better results for the 61% of the benchmarks with an average gain of about 22% in terms of T-gates, within the same time limit. Some benchmarks particularly benefit from this strategy, since their cost gain is more than the 70%.

V. CONCLUSION

This paper has described a pre-processing procedure and a reconstruction method to ease quantum synthesis. The proposed strategy is given by a PSOP decomposition based on the expression $x_i \oplus p$. Moreover, the algorithms have been experimentally tested on decompositions where p is a variable, an AND of variables and an XOR of variables, validating the proposed approach.

The synthesis method based on PSOP decomposition gives interesting results. Nevertheless, the decomposition is applied to SOP forms, since the PSOP decomposition is based on some statistics on the variables appearing in the starting SOP. This means that the input Boolean function must be represented with a PLA or any 2 level logic representation.

The future works on this topic should study new methods for deriving PSOP forms starting from other representations as AND inverter graphs or ROBBDs. Moreover, another interesting new direction is the study of the use of other possible decompositions for easing quantum compilation. For example, it would be interesting to study "Projected Exclusive Sum of Products" forms as a starting point for reversible

TABLE II
COMPARISON BETWEEN THE COMPILATION HEURISTIC PROPOSED IN [11] (STANDARD SYNTHESIS) APPLIED AFTER ESPRESSO IN THE HEURISTIC MODE, AND THE BEST SOLUTION OF THE PROPOSED STRATEGY.

	Standard synthesis		Best PSOP		Gain
Benchmark	**T -count**	**# qubits**	**T -count**	**# qubits**	**(T-gates)**
addm4	**1680**	429	2048	521	–
adr4	**108**	35	216	62	–
amd	1244	325	**1096**	288	12%
apla	**404**	111	616	164	–
b3	1220	338	**1200**	333	2%
b10	1520	396	**1424**	372	6%
b12	280	85	**236**	74	16%
bench	**228**	63	260	71	–
br1	504	138	**444**	123	12%
br2	**348**	99	**348**	99	–
co14	192	62	**172**	57	10%
dc2	**328**	90	**328**	90	–
exp	**1132**	292	1192	307	–
f51m	454	121	**400**	108	12%
fout	**820**	211	900	231	–
gary	1716	444	**1440**	375	16%
in0	1720	445	**1656**	429	4%
in2	1352	357	**1316**	348	3%
in3	1284	356	**1164**	326	9%
in5	1216	328	**956**	264	21%
in7	480	146	**316**	106	34%
inc	**364**	98	380	102	–
m3	1024	264	**888**	230	13%
m4	2128	540	**1592**	406	25%
max128	1392	356	**1196**	307	14%
mlp4	**1264**	324	1308	335	–
newapla	136	46	**40**	23	71%
newcpla1	336	93	**260**	75	23%
newcpla2	228	64	**148**	45	35%
newxcpla1	508	136	**128**	42	75%
p3	632	167	**584**	155	8%
p82	**292**	78	320	85	–
rckl	520	162	**296**	107	43%
rd73	352	95	**180**	52	49%
root	516	137	**452**	121	12%
spla	**2536**	650	2720	696	–
sqr6	404	108	**392**	105	3%
sym10	1420	365	**748**	197	47%
t1	**568**	163	572	164	–
t3	252	75	**228**	69	10%
tms	744	194	**680**	178	9%
vg2	444	136	**292**	99	34%
x6dn	1112	317	**1080**	309	3%
x9dn	408	129	**256**	92	37%
Z5xp1	456	121	**352**	95	23%
Z9sym	828	216	**680**	179	18%

logic synthesis, instead of Exclusive Sum of Products (ESOP) expressions [15].

ACKNOWLEDGMENT

This study was carried out within the National Centre on HPC, Big Data and Quantum Computing - SPOKE 10 (Quantum Computing) and received funding from the European Union Next-GenerationEU - National Recovery and Resilience Plan (NRRP) – MISSION 4 COMPONENT 2, INVESTMENT N. 1.4 – CUP N. I53C22000690001.

This work was supported in part by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the Italian MUR. Neither the European Union nor the Italian MUR can be held responsible for them.

The first two authors are members of the Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM), which provided partial support for this work.

REFERENCES

[1] A. Bernasconi, A. Berti, V. Ciriani, G. M. D. Corso, and I. Fulginiti, "XOR-AND-XOR logic forms for autosymmetric functions and applications to quantum computing," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 42, no. 6, pp. 1861–1872, 2023.

[2] A. Bernasconi, V. Ciriani, and R. Cordone, "On Projecting Sums of Products," in *11th Euromicro Conference on Digital Systems Design: Architectures, Methods and Tools*, 2008.

[3] ——, "The optimization of kEP-SOPs: Computational complexity, approximability and experiments," *ACM Trans. Design Autom. Electr. Syst.*, vol. 13, no. 2, 2008.

[4] A. Bernasconi, V. Ciriani, A. T. Monfared, and S. Zanoni, "Compact quantum circuits for dimension reducible functions," in *26th Euromicro Conference on Digital System Design, DSD 2023, Golem, Albania, September 6-8, 2023*. IEEE, 2023, pp. 776–781.

[5] A. Bernasconi, V. Ciriani, G. Trucco, and T. Villa, "On decomposing boolean functions via extended cofactoring," in *2009 Design, Automation & Test in Europe Conference & Exhibition*. IEEE, 2009, pp. 1464–1469.

[6] J. C. Bioch, "The complexity of modular decomposition of boolean functions," *Discrete Applied Mathematics*, vol. 149, no. 1-3, pp. 1–13, 2005.

[7] R. Brayton, G. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*. Kluwer Academic Publishers, 1984.

[8] P. Kerntopf, "New generalizations of shannon decomposition," in *Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design*, 2001, pp. 109–118.

[9] D. Maslov, G. W. Dueck, and D. M. Miller, "Synthesis of fredkin-toffoli reversible networks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 6, pp. 765–769, 2005.

[10] G. Meuli, M. Soeken, E. Campbell, M. Roetteler, and G. De Micheli, "The role of multiplicative complexity in compiling low t-count oracle circuits," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2019, pp. 1–8.

[11] G. Meuli, M. Soeken, E. Campbell, M. Roetteler, and G. D. Micheli, "The role of multiplicative complexity in compiling low t-count oracle circuits," in *Proceedings of the International Conference on Computer-Aided Design, ICCAD*, D. Z. Pan, Ed. ACM, 2019, pp. 1–8.

[12] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *Proceedings of the 40th annual Design Automation Conference*, 2003, pp. 318–323.

[13] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge university press, 2010.

[14] B. Schmitt, F. Mozafari, G. Meuli, H. Riener, and G. D. Micheli, "From boolean functions to quantum circuits: A scalable quantum compilation flow in C++," in *Design, Automation & Test in Europe Conference & Exhibition*. IEEE, 2021, pp. 1044–1049.

[15] B. Schmitt, M. Soeken, G. D. Micheli, and A. Mishchenko, "Scaling-up ESOP synthesis for quantum compilation," in *2019 IEEE 49th International Symposium on Multiple-Valued Logic (ISMVL), Fredericton, NB, Canada, May 21-23, 2019*. IEEE, 2019, pp. 13–18.

[16] E. Testa, M. Soeken, H. Riener, L. Amaru, and G. D. Micheli, "A logic synthesis toolbox for reducing the multiplicative complexity in logic networks," in *2020 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2020, pp. 568–573.

[17] E. Testa, M. Soeken, L. G. Amarù, and G. D. Micheli, "Reducing the multiplicative complexity in logic networks for cryptography and security applications," in *Proceedings of the 56th Annual Design Automation Conference 2019, DAC 2019, Las Vegas, NV, USA*, 2019, p. 74.

[18] R. Wille, S. Hillmich, and L. Burgholzer, "Efficient and correct compilation of quantum circuits," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[19] S. Yang, "Logic synthesis and optimization benchmarks user guide version 3.0," Microelectronic Center, User Guide, 1991.