Fault Detection in Parity Preserving Reversible Circuits

Nils Przigoda*†  Gerhard Dueck‡  Robert Wille¶  Rolf Drechsler*†

*Group of Computer Architecture  †Cyber-Physical Systems  ‡Faculty of Computer Science  ¶Johannes Kepler University Linz
University of Bremen  DFKI GmbH  University of New Brunswick  Institute for Integrated Circuits
28359 Bremen, Germany  28359 Bremen, Germany  28359 Bremen, Germany  4040 Linz, Austria

email: *{przigoda,drechsle}@informatik.uni-bremen.de  †gdueck@unb.ca  ¶robert.wille@jku.at

Abstract—Motivated by its variety of applications in several (emerging) technologies, the design of reversible circuits received significant attention in the recent past. With the emergence of physical realizations, also the consideration of faults and fault-tolerance became important. It has been suggested that parity preserving circuits would be ideal for fault detection, since here the parity of the inputs is the same as the parity of the outputs. Hence, if there is a fault on any single output, the parity should be flipped which would make the fault easy to detect. This paper however shows that this is not always the case. In fact, we provide and discuss examples showing that it is not sufficient to have parity preserving circuits when considering established fault models for reversible logic. As a result of our investigations, we can conclude that, even if a reversible circuit is parity preserving, it has to be checked against a particular fault model.

I. INTRODUCTION

Reversible circuits represent an emerging technology based on a computation paradigm which significantly differs from conventional circuits. In fact, they allow bijective operations only, i.e., n-input n-output functions that map each possible input vector to a unique output vector. Reversible computation enables several promising applications and, indeed, surpasses conventional computation paradigms in many domains including but not limited to quantum computation (see, e.g., [1]), certain aspects of low-power design (as experimentally observed, e.g., in [2]), the design of adiabatic circuits (see, e.g., [3]), encoding and decoding devices (see, e.g., [4]), or verification (see, e.g., [5]).

Accordingly, also the consideration of the design of reversible circuits received significant interest. In comparison to conventional circuit design, new concepts and paradigms have to be considered here. For example, fanout and feedback are not directly allowed. This affects the design of reversible circuits and requires alternative solutions. To this end, several approaches ranging from synthesis (see, e.g., [6], [7], [8], [9], [10]), optimization (see, e.g., [11], [12]), verification (see, e.g., [13], [14], [15]), and debugging (see, e.g., [16]) have been introduced. An overview of that is, e.g., provided in [17], [18].

In parallel, how to physically build reversible and quantum circuits is investigated and led to first promising results (see, e.g., [19], [20]). With this, also the question of how to prevent and detect faults in the physical realization became relevant. In particular for quantum computation, this is a crucial issue: Quantum systems are much more fault-prone than conventional circuits, since the phenomenon of quantum decoherence forces the qubit states to decay — resulting in a loss of quantum information which, eventually, causes faults.

Besides the work on testing and test pattern generation (see, e.g., [21], [22], [23]), this also triggered the design of fault-detecting or fault-tolerant circuits — leading to the development of fault-tolerant libraries (see, e.g., [24], [25]) and corresponding design methods. The later includes the design of parity preserving reversible circuits which gained significant attention in the recent years and yielded several contributions such as [26], [27], [28], [29], [28], [30], [31], [32]. This development was motivated by the benefits of parity preservation in conventional circuits and aimed for adapting this to reversible circuits as well (this is discussed in more detail later in Section III).

However, no real investigation or discussion has been performed yet tackling the question whether parity preserving is indeed useful for reversible/quantum circuits. In this work, we are conducting such an investigation. To this end, we are considering two faults models that rank amongst the mostly considered models for reversible and quantum circuits: the single missing control fault and the single missing gate fault [22]. We explicitly apply previously proposed design methods for parity preserving reversible circuits and evaluate the resulting netlists with respect to faults from these fault models.

Our evaluations unveil that, although previously proposed techniques indeed guarantee parity preservation, this property is often not helpful with respect to the commonly assumed faults. In fact, parity preserving reversible/quantum circuits are frequently not capable of detecting or even tolerating missing control as well as missing gate faults. If they are, the respective
method to make a circuit parity preserving yields a significant increase in the costs of the circuit. In this sense, our work motivates a re-evaluation of the need and usefulness of parity-reversing reversible and quantum circuits.

The remainder of this work is structured as follows: The next section reviews the basics on reversible circuits and the fault models considered here, while, Section III provides an overview on recent work which has been published on parity preserving reversible circuits. Based on that, Section IV includes the evaluation and discussion on the resulting circuits with respect to fault detection and correction. The findings of the work are, eventually, summarized in the conclusions in Section V.

II. PRELIMINARIES

To keep the remainder of the paper self-contained, this section introduces reversible circuits and provides a brief overview on the fault models considered in this work.

A. Reversible Circuits

A logic function $f: \mathbb{B}^n \to \mathbb{B}^m$ over inputs $X = \{x_1, ..., x_n\}$ is reversible iff (1) its number of inputs is equal to its number of outputs (i.e. $n = m$) and (2) it maps each input pattern to a unique output pattern. That is, reversible functions represent bijections. Reversible circuits are realizations of reversible functions. A reversible circuit $G$ is a cascade of reversible gates $g_i$, i.e., $G = g_1g_2...g_d$, where no fanout and feedback is allowed [1]. In this work, we consider the most widely used reversible gate, the Toffoli gate [33]

**Definition 1.** A Toffoli gate over the set of inputs $X = \{x_1, ..., x_n\}$ has the form $g(C, t)$, where $C \subset X$ is the set of control lines and $t \in X \setminus C$ is the target line. In the following the target line has the index $k$, i.e., $t = x_k$. A single Toffoli gate $g(C, t)$ realizes the bijective function

$$(x_1, \ldots, x_n) \mapsto (x_1, \ldots, x_{k-1}, x_k \oplus \bigwedge_{x_c \in C} x_c, x_{k+1}, \ldots, x_n).$$

That is, if all control line variables $x_c$ are assigned 1, the target line $t$ is inverted. Under this assignment the gate is called activated. All other input values $x \in X \setminus \{t\}$ pass the gate unaltered. Note that the set of control lines may be empty. In this case, the gate works as a NOT gate, i.e., the target line is always inverted. Gates with a single control line are termed CNOT. When the term “Toffoli gate” is used in this paper, it is implied that it has two controls, unless otherwise noted.

**Example 1.** Fig. 1(a) shows a reversible circuit including three circuit lines and four Toffoli gates, i.e., $n = 3$ and $d = 4$. Control lines are denoted by a $\bullet$, while the target line is denoted by $\oplus$. The annotated values demonstrate the computation of the respective gates for a certain input pattern. In this case, gates $g_1$ and $g_3$ are activated.

Note that reversible circuits usually provide a “blueprint” for quantum circuits. That is, in order to realize the logic design of a quantum circuit, the respective function is first synthesized as a reversible circuit and, afterwards, mapped to a quantum circuit structure (using methods such as [34], [35]). As the underlying fault models (covered in the next section) are similarly applied for both, reversible and quantum circuits, we omit an explicit introduction of quantum circuits and perform the discussion on reversible circuits only.

B. Fault Models

As in conventional circuits, faults may occur in reversible and quantum circuits (either caused by production or degradation). In order to abstract from the physical faults and, hence, allow for a logical consideration of the corresponding effects, usually discrete fault models are utilized. For reversible and quantum circuits, the single missing control fault and the single missing gate fault rank amongst the mostly considered models [22]. They are defined as follow:

**Definition 2.** Let $g(C, t)$ be a Toffoli gate of a circuit $G$. Then,

1) a Single Missing Control Fault (SMCF) appears if instead of $g(C, t)$, a gate $g'(C', t)$ is executed, whereas $C' = C \setminus \{x_i\}$ with $x_i \in C$ (i.e., a control line is removed).

2) a Single Missing Gate Fault (SMGF) appears if instead of $g$ no gate is executed (i.e., $g$ completely disappears).

In order to detect a fault, the respective gates have to be activated so that the faulty behavior can be observed at the outputs of the circuit. Depending on the considered fault, this requires certain input assignments [22]. More precisely,

1) to detect an SMCF at gate $g(C, t)$, all control lines in $C$ (except the missing one) have to be assigned 1, while the missing control line $x_i \in C$, respectively $x_i \notin C'$, has to be assigned 0. The assignment of the remaining lines can be arbitrarily chosen.

2) to detect an SMGF of gate $g(C, t)$, i.e., the disappearance of $g$, all control lines in $C$ have to be assigned 1, i.e., $g$ simply has to be activated. The assignment of the remaining lines can be arbitrarily chosen.

**Example 2.** Fig. 1 illustrates an SMCF 1(b) and an SMGF 1(c), respectively, which can occur in the reversible circuit previously introduced in Fig. 1(a). The respective assignment needed to detect these faults are also given.

III. PARITY PRESERVING REVERSIBLE CIRCUITS

Parity bits have been used in electronic data transmission form the early days of computers, since it is easy to detect a fault in a single bit. Thornton [36] has shown that a parity bit can be efficiently embedded in adder circuits. Parhami [37] coined the term parity preserving in 2002. None of the papers considered the fault coverage of such a circuit. Furthermore, both papers consider conventional Boolean circuits that are not necessarily reversible.
A multiple-input multiple-output function $f(X)$ is parity preserving if $\bigoplus X = \bigoplus f(X)$, for all assignments of the input variables in $X$.

In 2006, Parhami used the parity preserving property for reversible logic circuits [38]. He noted that any change in a single output will be detected.

Paul et al. [26] have shown that there are $(2n-1)2^n$ reversible parity preserving function with $n$ variables while the number of all reversible circuits is $(2^n)!$. This means that less than one percent of all reversible circuits with four or more lines are parity preserving. But any reversible function can be made parity preserving by adding one more input and setting the corresponding output to preserve the parity of the input (a proof is also given in [26] fmtoreover, in [26] it is shown how parity preserving functions can be generated with exponential complexity. It is not known how the complexity increase of the function will affect the cost of the circuit.

Another way to embed a non parity preserving reversible circuit into an parity preserving reversible circuit is depicted in Fig. 2. By adding one line, with constant zero input, to the original circuit and calculating the parity before and after the original circuit on the added line, the new overall circuit will be parity preserving. Unfortunately, this circuit will always produce a parity preserving input-output relation independently of the original circuit $C$ (assuming that all CNOTs gates before and after the original are fault-free). Hence, no fault in the original circuit $C$ will be detected which does not motivate further studies.

The main idea of many papers dealing with parity preserving functions/circuits is to build circuits with elementary building blocks in order to detect if a circuit under test realizes a parity preserving function. For example, Haghparast and Navi [27] presented the design of a new reversible fault tolerant gate (NTF) that is parity preserving. It should be noted that in [27] the term fault tolerant is used in a misleading fashion. A system is said to be fault tolerant if it continues to work correctly, even with the presence of some faults. Clearly, a parity preserving circuit will, at best, detect a fault, but will not be able to correct it. Since the fault cannot be corrected, the system will not be able to continue to operate correctly and thus is not fault tolerant.
concerning the circuits cost, it should be mentioned that if the
of both realizations is only used to preserve the parity. Con-
needed, because the value of the bottom line in the figure
are applied.

are shown in Fig. 4. Here, either
pendently from the number of control lines) parity preserving
gates into a parity preserving block. In nearly all
models will not be detected in circuits with parity preserving
elementary building block/gates do not help
in detecting a fault according to the the missing gate fault
model.

2) Single Missing Control Fault:
Let us now consider the missing control fault model. As the
elementary building blocks/gates in other papers are presented
as black boxes, we cannot really apply a missing control fault
anywhere. Hence, it remains unknown how to model a missing
control fault within a building block. In fact, the realization
of the block must be known before the missing control faults can
be analyzed. One exception holds for [29], where the authors
do not only describe the elementary parity preserving building
blocks, but also present a quantum realization of an embedded
Toffoli gate. A detailed analysis of this block will later be
conducted in Sect. IV-C.

B. Parity Preserving Building Blocks

As shown in the last subsection, faults of the considered
models will not be detected in circuits with parity preserving
elementary building blocks. Thus, we have to analyze the
quantum realizations of such building blocks. Unfortunately, of
the standard gates, only the Fredkin gate is parity preserving.
The NOT, CNOT and Toffoli gate are not parity preserving.
However, an additional line is needed to embed any of the
last-named gates into a parity preserving block. In nearly all
of the papers presented in the last section, exactly one line
with a constant zero input is added to implement these gates.

Two straightforward methods to make a Toffoli gate (inde-
dependently from the number of control lines) parity preserving
are shown in Fig. 4. Here, either

- a double gate as shown in Fig. 4(a) or
- two additional CNOT gates as shown in Fig. 4(b)
are applied.

Note that for either method only one additional line is
needed, because the value of the bottom line in the figure
of both realizations is only used to preserve the parity. Con-
cerning the circuits cost, it should be mentioned that if the
first method is used, twice as many gates are needed and the
quantum cost doubles. While the second method will add twice
as many CNOT gates as the number of used Toffoli gates. This
also means, that the quantum costs will increase by two times
the number of Toffoli gates, since a CNOT gates has quantum
cost 1.

Furthermore, note that we will ignore faults that will cause
a non-Boolean values at any control or output.

1) Double gate method:
Let us now assume that the double gate (see Fig. 4(a)) method
has been used. If exactly one of the two Toffoli gates is
missing, the building block will not be parity preserving. Thus,
the whole circuit will also be not parity preserving and it is
theoretically possible to detect this fault provided that we know
a corresponding input/output pattern.

Besides a missing gate, it is also possible that exactly
one control is missing. Fortunately, in this case, we can also
find an input/output pattern that breaks the parity preserving
feature/property. For example, if the first control (on line \(a\))
of the first Toffoli gate is missing, the input \((1,1,0,0)\) will be
mapped to the output \((1,1,0,1)\). For all 4 missing control cases,
such a pattern can be found.

Hence, this design detects all faults of the two considered
fault models, but—as stated above—at the expense of signifi-
cantly increased costs.

2) Add 2 CNOTs method:
Assuming that all buildings blocks are made of 2 CNOTs
around the original Toffoli gate (see Fig. 4(b)), three different
gates can be missing: one of the two CNOTs or the original
gate. In the first case, i.e., one of the CNOTs is missing,
the whole block will lose its parity preserving feature. In the
latter case, i.e., the original gate is missing, the two remaining
CNOTs together are realizing the identity function and, thus,
the whole block will still be parity preserving. So, in 2 of 3
cases a missing gate would be detected.

If the missing control fault model is applied, there are
exactly two good cases: One of the CNOT controls is missing.
Then the whole block will lose its parity preserving feature.
On the other side, if one control of the main Toffoli gate
is missing, the whole block will still be parity preserving,
even if the overall circuit will not realize the desired function.
If multiple-controlled Toffoli gates are considered, then this
design provides very poor fault coverage. To summarize this,
method 2 will obtain only 2 of \(|C| + 2\) SMCF instances.

It can be concluded, that the additional cost is low, but the
fault coverage is not satisfactory.
C. Quantum Realization for a Parity Preserving Toffoli Gate

Instead of only using parity preserving building blocks, the authors of [29] have also proposed a quantum realization of the parity preserving Toffoli gate. We now investigate whether such a quantum realization can be used to detect faults. To this end, consider the quantum realization of a Toffoli gate as proposed in [29] and shown in Fig. 5. Both of the introduced fault models can be applied to the realization. First, the SMCF model is considered, afterwards the SMGF model.

All possible SMCFs for CNOT gates that do not result in entanglement, i.e., do not trigger a control line with a non-Boolean value, are labeled $a, b, c, d, e$ in Fig. 5. Those resulting in non-Boolean intermediate states are labeled $w, x, y, z$. Let us briefly consider the first set of faults:

- $a$: Since the value of $B$ will be always flip at the beginning, the parity is not preserved.
- $b$: Since this fault will only flip $B$, it will not preserve the parity.
- $c$: Similar to above.
- $d$: Since only $R$ is inverted, the parity is not preserved.
- $e$: Since only $Q$ is inverted, the parity is not preserved.

For all these SMCF instances, Tab. I gives one respective input/output pattern for the correct circuit as well as the output for the faulty circuit. This shows that the circuit is not parity preserving for any of the considered faults.

If one of the controls of $w, x, y, z$ is missing, non-Boolean outputs can appear which require a measurement of the qubits. Unfortunately, these measurements are non-deterministic and we can not make any safe statement. To be more precise, if none of the controls of the gates are missing, a measurement it not necessary because all values will be pure Boolean. However, if any control is missing and the measurement states that one calculation is parity preserving, we can make any statement, but if the measurement shows us an input/output pattern which is not parity preserving, we can know for sure that the circuit is faulty.

Analyzing possible SMGFs has produced the results as shown in Tab. II. While a missing first gate does not have any effect on the overall circuit, a missing gate at the position 6, 7, 8, or 9 would destroy the parity preservation. If one of the gates 2–5 is missing, again, non-Boolean values can be found at controls, thus, they are again omitted.

As the tables show, using the quantum realization of Islam gives us a good fault detection rate by checking the parity preserving property. However, for each Toffoli gate one additional constant zero line has to be added which is impractical for larger circuits. Since the first CNOT gate of the quantum realization will never be activated because the control is on a constant zero input (unless the circuit has no faults), the quantum cost for the realization can be quantified with 8. Compared to the quantum cost of 5 of the normal Toffoli gate, the quantum cost of a final circuit would increased by a factor of $8/5 = 1.6$.

To the best of our knowledge, no method has been published that describes how, with only a fixed number of constant lines, more parity preserving Toffoli gates can be combined. In Fig. 6, we present such an embedding of a Toffoli gate. Unfortunately, all possible SMCF as well as SMGF would again cause non-Boolean signals, but without the usage of at least some $V$ or $V^\dagger$ gates, it would be impossible to realize the functionality.

V. Conclusions

In this work, we considered the effectiveness of parity preserving reversible circuits with respect to established fault models. Our investigations show that parity preserving techniques are inadequate to cover single missing gate faults and single missing control faults. Deficiencies can be observed in terms of both, costs (i.e., they introduce large costs) as well fault coverage. Examples have been provided which show that barely half of the faults are covered.
The lesson that we can learn from this investigation is that parity preserving circuits do not automatically ensure a reasonable fault coverage which always should be validated with a specific fault model. In the future, parity preserving might be beneficial when new (more realistic) faults models become available. But thus far, their benefit to reversible and quantum circuits is rather limited.

ACKNOWLEDGMENTS
This work has partially been supported by the Siemens AG as well as the EU COST Action IC1405.

REFERENCES