FACTA UNIVERSITATIS
Series: **Electronics and Energetics** Vol. 39, N° 1, March 2026, pp. 77 - 103
https://doi.org/10.2298/FUEE2601077H

Original scientific paper

FAULT DETECTION SCHEME FOR NEGATIVE CONTROL FLIPPING FAULTS IN REVERSIBLE CIRCUITS

Mousum Handique[1], Amrit Prasad[2], Rajeeb Dey[3], Valentina E. Balas[4,5]

[1]Department of Computer Science and Engineering, TSSOT, Assam University, Silchar, 788011, Assam, India
[2]Rapawalk Fashion Technologies, Bengaluru, 560025, Karnataka
[3]Department of Electrical Engineering, National Institute of Technology Silchar, Silchar, 788011, Assam, India
[4]Faculty of Engineering, Aurel Vlaicu University of Arad, Arad, Romania
[5]Academy of Romanian Scientists, Bucharest, Romania

ORCID iDs:	Mousum Handique	https://orcid.org/0000-0002-8290-6185
	Amrit Prasad	https://orcid.org/0009-0007-5547-4173
	Rajeeb Dey	https://orcid.org/0000-0002-6573-0114
	Valentina E. Balas	https://orcid.org/0000-0003-0885-1283

Abstract. *Today's the CMOS technologies-based electronic devices are designed to be comprised of numerously efficient microchips to meet the demands of fast computational power, and there is a need to perform loss-less information computation. In this context, the reversible logic way of computation is a rapidly growing research area for low-power circuit design and lossless information computation. The conceptual reversible computation is widely applied in current technologies, such as quantum computing. The analysis and efficient functioning of the reversible logic depended on the various domains, such as reversible logic synthesis, verification, testing, and debugging. This article focuses on the domain of testing reversible logic circuits, which then examines a fault model referred to as the negative control flipping fault (NCFF) under the control flipping fault (CFF) model. In this paper, the proposed work necessitates the utilization of an automatic test pattern generation (ATPG) algorithm to generate a complete test set for detecting NCFFs. Moreover, the present work shows the correlation of NCFF with the existing fault models in reversible circuits by encompassing single missing gate fault (SMGF), multiple missing gate fault (MMGF), and partial missing gate fault (PMGF). Finally, experimental results are performed on several benchmark circuits to verify our proposed algorithm for fault detection of NCFF. Additionally, we have assessed the fault coverage capabilities of the existing fault models in reversible circuits with the help of a generated complete test set for NCFF in reversible circuits.*

Key words: *k-CNOT circuit, Control flipping faults, Negative control flipping fault, Complete test set*

Received March 10, 2025; revised June 26, 2025, and July 25, 2025; accepted August 28, 2025
Corresponding author: Mousum Handique
Department of Computer Science and Engineering, TSSOT, Assam University, Silchar, 788011, Assam, India.
E-mail: mousum.handique@aus.ac.in

1. INTRODUCTION

In the electronics circuit design domain, advancing the fabrication process and integrating more ICs lead to fabricating an intricate circuit design that incorporates a greater number of components within a small silicon area. The nature of a complex circuit design with a smaller size increases the complexity, such as more power dissipation, probability of faults increases, etc. In connection with the traditional circuit, the gate operations are performed based on the different inputs and outputs; as a result, the information is lost and heat dissipation occurs. In 1961, Rolf Landauer [1] presented that the minimum quantity of energy $kTln2$ Joules is required to transmute every bit of information, with k representing the Boltzmann constant and T indicating the system's temperature in Kelvin. The Landauer principle shows the effect of energy dissipation during computation as a consequence of the loss of every bit information. It has been suggested by Bennett [2] in 1973 that $KTln2$ joules of energy dissipation would remain absent, provided that the system maintains logical reversibility, this consequently inspires researchers to delve into unconventional reversible logic as a viable alternative for circuit design. The synthesis of reversible circuits is suitable for large reversible function [3], which is also mapping in conventional circuit design approach. The Boolean function data can be used in traditional circuits, along with reversible circuits, where the redundancy data in compact form by applying various techniques like EXOR Sum of Products (ESOP) [4], transformation-based algorithm [5], Functional Decision Diagram (FDD) representation [6], etc. Along with providing low-power CMOS design technology, the reversible circuits as a specific instance within the broader category of quantum circuits. Therefore, researchers have been more focused on the utility of reversibility in the context of circuit design alternatives in recent days.

Within a reversible circuit, all the gate operations are reversible, i.e., maintain the one-to-one mapping operation and gates are arranged in a linear cascade sequence. Due to the bijective operation, the reversible circuit ensures retrieval of the output from the input [7]. The nature of reversible logic follows the two conditions: (1) the deterministic device performs reversible to ensure that input and output can be uniquely and mutually retrieved from one another, i.e., logically reversible, and (2) the device is physically reversible, i.e., it can be capable to run backward. Based on these conditions, the reversible circuit computes the lossless information while executing the operations within reversible logic. The utilization of the reversible logic circuit implementation is evident in various contemporary technologies like DNA computing [8,9,10], optical computing [11], quantum computing [12], and quantum cellular automata (QCA) [13,14], etc.

In circuit design technology, there may be a possibility that even a single fault occurrence leads to effects on the entire functional behavior of the system. Therefore, detection of faults requires attention to continue the correct operation of a system, such that the system can be capable of providing the expected performance. For this purpose, testing is the experimental process over the system to ensure the correct operational state is performed during and after the circuit's design process. Due to the complex circuit structure, the number of physical parameters affecting a gate is innumerable, it is impossible to individually examine each physical failure that causes significant deviation from the anticipated system performance. Therefore, we need some conceptual mathematical model designed to depict the physical defects of a system. To be more precise, a fault model is a mathematical representation employed to be used for representing the different abstraction levels of faults and aiding the

simplification of testing procedures for fault detection within a circuit [15]. Within reversible circuits, numerous fault models are identified in [16] to describe the structural level of the circuit and their connection of those fault models.

The testing process is performed by the sequence of inputs applied to the circuit such that the functional behavior of the faulty and fault-free circuit can be distinguished. The applied inputs in the circuit are called input test vectors and the collection of input test vectors is represented by the test set. The testing process is evaluated by producing efficient test vectors for identifying the faults that are present in the circuit, which is recognized as an NP-hard problem [15]. For fault-detecting purposes, when a test set can detect all faults present in a given circuit, it is referred to as a complete test set (CTS). Because of its reversible nature, the circuit demonstrates exceptional controllability and observability [17]. The concept of controllability states that the produced output test vector based on the applied reversible gate operation at any circuit's level can produce a unique test vector at the initial stage of the reversible circuit through a backtracking process. The observability property signifies that modifications at the intermediary level of the circuit result in changes to the final output vector.

In the present literature, many approaches [17, 18, 19, 20, 21, 22] have been contributed such that efficient test set generation for fault detection in reversible circuits relies on various fault models. Several existing methods have utilized deterministic approaches to establish fault detection logic for classical fault models such as stuck-at faults, bridging faults, and cell faults in k-CNOT reversible circuits. The fault detection logic [17] is designed using integer linear programming (ILP) to assess the detection mechanism for identifying stuck-at faults and cell faults. The use of ILP for generating an efficient and complete test set is practical and scalable when the circuit has a small number of input vectors or limited input wires. In contrast, the complete test set size is increased when the number of input wires is more in the circuit. The ILP-based experimental analysis has shown that the execution time for constructing the complete test set increases exponentially with the number of gates, i.e., circuit length. The authors in [18] have introduced a deterministic approach using the block division method for constructing the complete test set that can identify the intra-level single bridging faults. The experimental analysis presented in this study is limited to small benchmark circuits. The block division method requires a high computational cost for circuits with large input sizes. In other way, some of the existing works [19, 20, 21, 22] have discussed the logical fault models, like SMGF, PMGF, MMGF, and repeated-gate fault (RGF) that are explicitly used for the reversible circuits. Some of these existing methods [19, 20] have relied on specific tools like ILP, Boolean satisfiability (SAT), and pseudo-Boolean optimization (PBO) to explore the basic requirement of the fault detection logic for developing the test set. The performance of the applied tools for producing the complete test set to identify faults is significantly influenced by the size of the circuit. The work in [21] presented a design-for-testability (DFT) method for detecting SMGF, MMGF, detectable RGF and PMGF without using an ATPG. However, the hardware overhead cost is increased for the large circuits. The authors in [22] developed ATPG algorithms using exact approaches that are applied to detect the single stuck-at faults, multiple stuck-at faults, MMGF, SMGF, RGF, and PMGF in the reversible circuits. Though the ATPG algorithms of this existing method accurately construct the minimal complete test set, the performance of the computation degrades for the large input lines and gate counts of the circuits. As per the discussion of the above-mentioned existing works, it is observed that the determination of the fault detection logic and the process of test set generation are also contingent upon the specific structure of the employed fault model.

Moreover, the reversible gates are well decomposed in the quantum circuits, and the quantum circuits' real implementation is performed on various nanotechnologies [12] based on the electromagnetic pulses. Classical fault models like bridging fault, stuck-at fault, and cell fault, which are based on the wired-structured and fault detection logic of the classical fault models, may not be appropriate for reversible and quantum gate operations. As per the discussion of some of the existing works, it is noticed that the study or analysis of the fault detection logic connectivity from one fault model to another fault model is limited. Therefore, the mixing test set generation is required to cover various fault models in reversible circuits with less computation cost.

This article concentrates on constructing the complete test set (CTS) for the negative control flipping faults (NCFFs). For this purpose, the fault detection logic is developed for detecting the NCFFs in k-CNOT circuits. The CTS is generated using the fault detection logic for NCFFs within the framework of the Control Flipping Fault (CFF) model. The applicability of the NCFFs is well suited for the reversible circuits and the quantum circuits in the current developments in nanotechnology. Moreover, the proposed work establishes a correlation between the NCFF and other fault models, such as SMGF, MMGF, and PMGF, regarding fault coverage range within the context of reversible circuits.

The primary contributions of this study can be outlined as follows:

(1) The complete test set (CTS) is constructed using an ATPG algorithm for identifying the NCFF in *k*-CNOT based circuit.
(2) The CTS generated by NCFF is used with other fault models in the reversible circuit to establish the correlation in terms of the fault coverage range.

The paper is structured as follows: Section 2 provides the fundamentals, like reversible logic, reversible gate operation, the arrangement of the *k*-CNOT circuit, and an outline of the fault models relevant to reversible circuits. Also, we discussed some of the existing works in this section, which are pertinent to the work we have introduced. Section 3 describes the detailed structural overview of the control flipping fault (CFF) and specifically focuses on the NCFF in the reversible circuit. Section 4 includes the methodology for constructing the CTS for identifying the NCFFs in *k*-CNOT circuits. This section also converses the correlation between the fault coverage range of various fault models and the CTS produced by the introduced method. Section 5 presents the experimental results obtained by the CTS generation method. The experimental results regarding the fault coverage range in relation to other already available fault models are also presented in this section. Section 6 contains the concluding observations of our proposed work.

2. PRELIMINARIES

2.1. Reversible Logic Function

When a function $f(x_1, x_2, \ldots, x_n)$ generates n Boolean variables and if f achieves the bijective function or one-to-one mapping, it is referred to as a reversible function. More precisely, the function $f: \mathbb{B}^n \Rightarrow \mathbb{B}^m$ is reversible, when the two conditions are satisfied: (a) The quantity of inputs, denoted as n, is equivalent to the number of outputs, represented as m, which is expressed as $n = m$ and (b) these inputs and outputs are uniquely retrievable from one another. Based on these concepts, an irreversible function can be converted into a reversible function by introducing additional outputs, commonly referred to as garbage line(s), in order to equalize the number of output lines with the number of

input lines while applying reversible operations [23]. To provide a brief illustration of the transformation from an irreversible to a reversible function, we consider the following example:

Example 1 *Let us contemplate the function f : (x, y) ⇒ (x ⊕ y) with 2-input and 1-output, as shown in Table 1 (a). Here, we observe that the function f is not reversible as the number of inputs and outputs is unequal and the function does not satisfy the properties of a bijective mapping* (*i.e.,* (0, 1 ⇒ (1) and (1, 0) ⇒ (1)). *The irreversible function f converts to the reversible function f' : (x, y) ⇒ (x, x ⊕ y) by adding extra output 'x' and follow the one-to-one mapping (bijective), which is depicted in Table 1 (b).*

Table 1 Illustration of Irreversible to Reversible function

(a) Irreversible function *f*

Input		Output
x	*y*	*x* ⊕ *y*
0	0	0
0	1	1
1	0	1
1	1	0

(b) Reversible function *f'*

Input		Output	
x	*y*	*x*	*x* ⊕ *y*
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

2.2. Reversible Logic Gates

To perform a reversible function, it's essential that the operations involved are themselves reversible, and these operations are achieved using reversible gates. A reversible gate characterized by having *n* input and output lines, denoting it as an $n \times n$ gate [17] and each input and output line establishes a one-to-one mapping with the others. Various reversible gates have been introduced for the synthesis of reversible circuits, including the NOT gate, CNOT (Controlled NOT) or FEYNMAN gate [24], Toffoli gate [25], MCT (Multiple-Control Toffoli) gate [25], FREDKIN gate [26,27], PERES gate [28], and SWAP gate [29]. Among these gates, the NOT, CNOT and Toffoli gates are regarded classical reversible gates and their generalized versions of these gates are represented by the *n*-bit MCT gate or *k*-CNOT gate. In general, *k*-CNOT gate can realize any arbitrary reversible functionality, thus, it is called as universal reversible gate [25]. Figure 1 illustrates the gate operation of classical reversible gates.

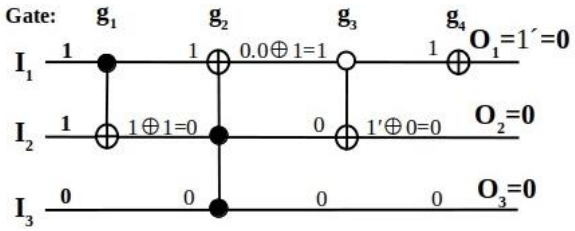

Fig. 1 Gate operation of classical reversible gates

2.3. Reversible Circuits

A reversible circuit consists exclusively of reversible gates, which are arranged in a sequential linear cascade. The reversible gate's interconnection does not permit the fanout and feedback connections in the circuit [12]. The level (L) is partitioned of each gate and the depth of the level is computed by the number of gates that are employed within a particular circuit. Contemplate a reversible circuit C=$\{g_1, g_2, \ldots, g_N\}$ is the formation of linear cascade sequence of reversible gates and N represents the overall count of gates utilized within circuit C. Here, each gate g_i lies between in the levels L_{i-1} and L_i, for $1 \leq i \leq N$ and the operation for gate g_i is executed only after the operation of gate g_{i-1} has been concluded. In another way, the output of the gate g_{i-1} functions as an input of the gate g_i. The evaluation of the performance cost metric for the reversible circuits depends on various parameters [22] like Quantum Cost (QC), Gate Count (GC), Garbage Outputs (GO), Ancilla Inputs (AI) and Delay (△).

Example 2 *Let us take the ex_1_166 reversible benchmark circuit that consists of four cascades of reversible gates {g_1, g_2, g_3, g_4} as depicted in Figure 2. Here, the circuit that comprises two CNOT gates g_1 and g_3, one Toffoli gate g_2 and one NOT gate g_4. The starting input vector* ⟨110⟩ *is utilized to the circuit at level L_0 and the final output vector* ⟨010⟩ *is generated at the level L_4.*

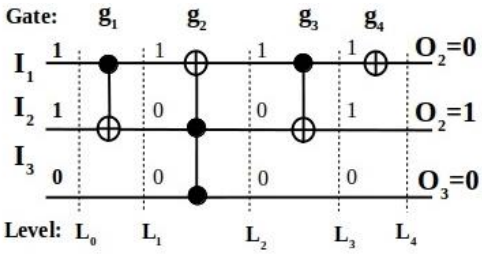

Fig. 2 *ex_1_166* benchmark circuit composed of **two** CNOT gates, **one** Toffoli gate and **one** NOT gate

2.4. Fault Models in Reversible Circuits

A fault model is constructed through a mathematical testing experiment, which elucidates various levels of fault abstraction, encompassing behavioral, functional, structural, and geometric aspects [15]. In this connection, a fault model is used to evaluate all the possible

faults based on their corresponding level of abstraction and reduce the effort of testing complexity in the circuit. This paper deals with the structural fault model related to the faults that appear at the gate level or within the interconnections between gates of circuits. The fault models namely Stuck-at Fault [17,30], Bridging Fault [19,31,32], and Cell Fault [17] are applied for both conventional logic circuits and reversible circuits. Additionally, certain fault models such as Partial Missing Gate Fault (PMGF) [19], Single Missing Gate Fault (SMGF) [33,19], Multiple Missing Gate Fault (MMGF) [33,19], Repeated Gate Fault (RGF) [19], and Crosspoint Fault [34] specifically address faults within reversible circuits. A comprehensive analysis of different fault models and their relationships in reversible circuits is presented in [16]. However, the work specifically concentrates on one type of fault, the negative control flipping fault (NCFF), which falls under the CFF model, a structural fault model for reversible circuits. A comprehensive discussion on the NCFF is provided in Section 3.

2.5. Generating the CTS for testing the reversible circuits

In this section, we focus only on offline testing approaches of reversible circuits that are relevant to our work. In 2011, R. Wille et al. [20] proposed three approaches to generate the ATPG for testing the reversible circuits, covering faults like single additional control fault (SACF), SMGF and single missing control fault (SMCF). The initial approach involves a simulation-based technique for generating test patterns, which is simple if the circuit is considered without additional constraints. The second approach is considered Boolean satisfiability (SAT) with the additional constraints of the circuit. Here, the ATPG is reformulated as an SAT instance, composed of circuit constraints to create the set of test patterns. In the PBO approach, the generated test pattern is the minimized form and handles the circuit with or without additional constraints. In 2012, M. Zamani et al. [35] proposed a Ping-Pong test technique for evaluating all SMGFs, RGFs and most of the MMGFs in reversible circuits. In this method, the final output test vector from the applied circuit is reused as the next test vector for fault detection. In 2013, J. Mondal et al. [36] introduced a design-for-test (DFT) approach, which is designed by adding two additional input lines, as well as incorporating a few k-CNOT gates in ($n \times n$) reversible circuits. The formulated testable design circuit can detect the SMGFs, PMGFs and detectable RGFs in the circuit by applying the test set size (n+1). In 2014, the authors in [37] introduced a Boolean difference method to derive a test set which is able of identify all SMGFs in k-CNOT circuits. In 2016, A. N. Nagamani et al. [22] developed an ATPG algorithm that uses an exact approach to generate a CTS for the single stuck-at faults, multiple stuck-at faults, RGFs, PMGFs, SMGFs and MMGFs in k-CNOT, Peres and FREDKIN based reversible circuits. In 2020, M. Handique et al. [38] presented a scheme to generate the complete test set for covering nearly 100% fault coverage of the SMGF and MMGF in k-CNOT reversible circuit. In 2021, J. Mondal et al. [39] introduced a DFT technique in reversible circuits, where the gates of a circuit are separated into different sets. In order to create a circuit design that is testable, an extra control input line is introduced, which is linked to the corresponding gate within each cluster. In 2023, the authors in [40] proposed a testable design technique for diagnosing the positive control flipping faults (PCFFs) under the control flipping fault (CFF) model in k-CNOT circuits. In this technique, the given circuit is augmented to make the testable design circuit by adding the duplicate copy of the original k-CNOT gate and preparing the blocks, where each block comprises the pair of original and duplicate copies of k-

CNOT gates. A single test vector is used to identify all PCFFs, and a parity test pattern helps locate the faulty gate by incorporating a parity-bit operation. The proposed work of the present article is the extended version of the existing work [40], where the CFF model maps to another control flipping fault, i.e., negative control flipping fault (NCFF). The fault detection logic for the NCFF is not identical to the fault detection logic in PCFF because of the logic variation of *k*-CNOT gate. The study of the extended PCFF to NCFF makes an accurate logical fault analysis of the CFF model.

As per the above discussion, it clearly indicates that current works are related to the heuristic approaches for generating efficient CTS to assess the identification of the various faults in the circuits. We have observed that the heuristic approaches have provided the solution of generating the CTS, but there is no exact minimal solution. Also, we have seen many fault detection approaches related to the DFT technique. Though the DFT technique efficiently detects all faults, the additional circuitry overhead cost is expensive. Moreover, the production of a complete test set depends on the specific fault model used in reversible circuits. There is limited research on creating test sets for one fault model that are also capable of detecting faults from another model. In our proposed work, we consider the fault model, labeled as the NCFF under the CFF model in *k*-CNOT circuits.

3. NCFF IN REVERSIBLE CIRCUITS

This section discusses an NCFF under the CFF model of the *k*-CNOT reversible circuit. The CFF model has been considered under the permanent faults, whose entire design is vulnerable to the designer, and any breakage or incorrect interconnection in design can be directed as permanent in CFF. Similarly, this can be classified as a structural fault, indicating that faults may arise at the gate level and within the connections between the gates [16]. CFF is a part of structural faults. In the k-CNOT gate structure, a control connection can be formulated as either a positive control (represented by •) or a negative control (represented by ◦), as illustrated in Figure 3 (a) and (b), respectively. A CFF takes place when a positive control is flipped to a negative control, or the other way around. Based on the control flipping, CFFs can occur in three possible ways- positive to negative control flipping, negative to positive control flipping and both are present in the *k*-CNOT gate structure. In this work, we consider only negative to positive control flipping, i.e., labeled as the NCFF in *k*-CNOT reversible circuit.

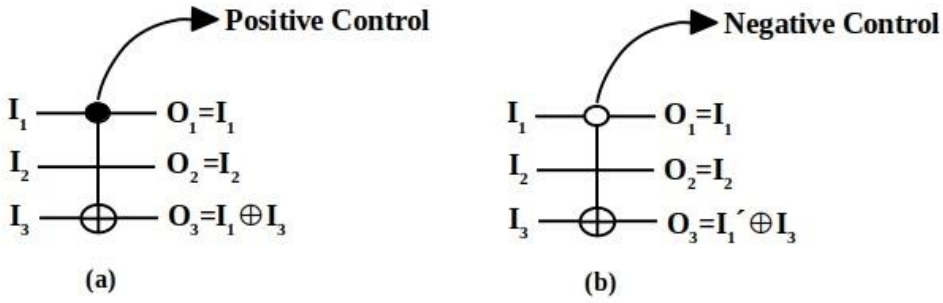

Fig. 3 Demonstration of (a) positive control connection (•) (b) negative control connection (◦) in CNOT gate

The NCFF occurs when negative control is switched to a positive control; this phenomenon is termed a negative control flipping fault (NCFF). An illustration of the NCFF is depicted in Figure 4, where the CNOT circuit contains only the negative control along with the target connection and a negative control is flipped to the positive control. As per the logical evidence of the NCFF in *k*-CNOT gate, the NCFF can be classified into two types: negative single control flipping fault (*n*-SCFF) and negative multiple control flipping fault (*n*-MCFF). In a *k*-CNOT reversible gate, one negative control is flipped to the positive control, which results in a design fault known as an *n*-SCFF. The single negative control lies at input line I_2 of the lefthand side of Figure 5 (a) and is flipped to the positive control at input line I_2, as shown in the righthand side of Figure 5 (a). In another way, more than one control connection is flipped from negative to positive within a gate, resulting in a design fault known as an *n*-MCFF. In Figure 5 (b), the lefthand side of the negative controls that occurred in the input lines I_1 and I_2 are flipped to the positive controls, respectively, as shown in the righthand side of Figure 5 (b).

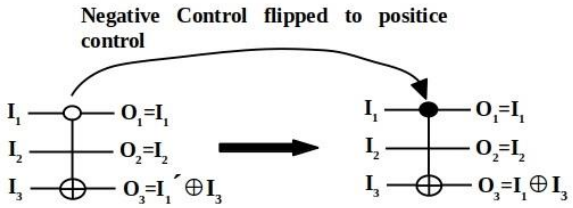

Fig. 4 Illustration of NCFF at input line I_1 in CNOT gate

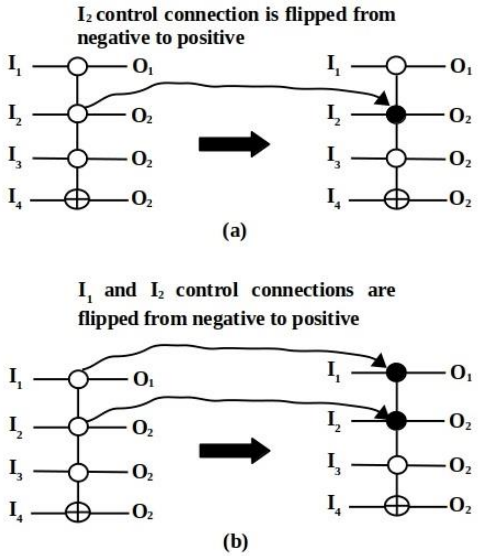

Fig. 5 Illustration of (a) *n*-SCFF at input line I_2 (b) *n*-MCFF at input lines I_1 and I_2

Example 3 *Here, we examine the reversible circuit to exhibit the effect of an n-SCFF and n-MCFF. For this purpose, a fault-free reversible circuit containing only negative control connection(s) is considered, as illustrated in Figure* 6 *(a). The initial input test vector* ⟨1000⟩ *at level* L_0 *is applied to the fault-free reversible circuit, as depicted in Figure* 6 *(a). As a result, the primary output* ⟨0100⟩ *generates at level* L_4 *after propagating each subsequent level of the circuit. Figure* 6 *(b) shows that an n-SCFF appears at the control connection that lies in the* I_1 *input line of gate* g_3*. Due to the occurrence of an n-SCFF, the faulty final output* ⟨0101⟩ *is generated at level* L_4 *in place of a fault-free final output* ⟨0100⟩ *by utilizing its identical initial input test vector* ⟨1000⟩ *at level* L_0*. Similarly, an n-MCFF appears at the control connections that are present in both* I_1 *and* I_2 *input lines of gate* g_3*, as illustrated in Figure* 6 *(c). However, the primary output test vector at level* L_4 *would be* ⟨0101⟩*, which represents a faulty final output caused by the presence of an n-MCFF.*

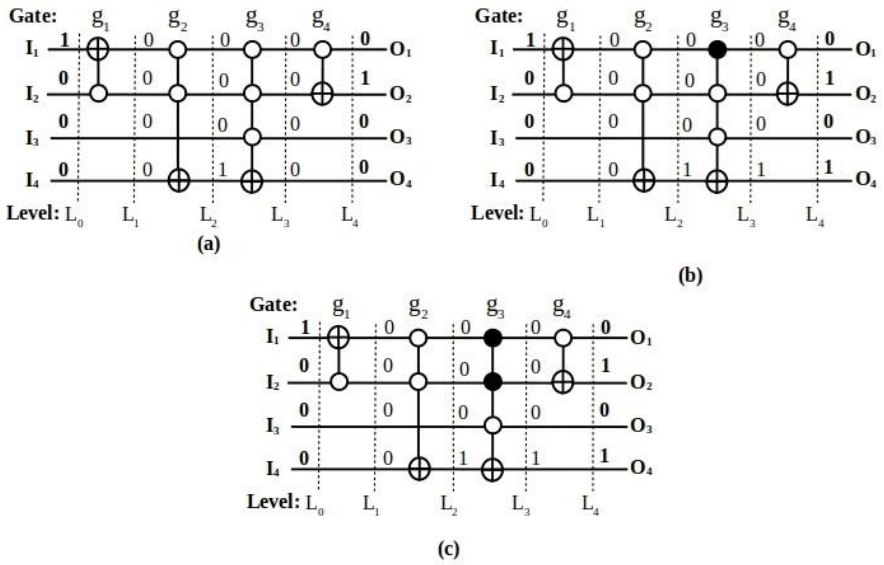

Fig. 6 Figurative of the reversible circuit: (a) the fault-free circuit (b) *n*-SCFF occurs at input line I_1 of the gate g_3 (c) *n*-MCFF occurs at both input lines I_1 and I_2 of the gate g_3

The evaluation of the total number of NCFFs that logically appeared in the *k*-CNOT circuits is discussed here. For this purpose, a general formula is formulated, which is given as: $\sum_{i=1}^{n-1}\left[K_i \sum_{j=1}^{i} ({}^{i}C_j)\right]$ where K_i denotes the number of *k*-CNOT gates that are occupied the same group of i^{th} control connections of total input lines *n* in a given circuit, for $1 \le i \le n$-1. Here, ${}^{i}C_j$ indicates the combination of $C(i, j)$, for $1 \le j \le i$.

Example 4 *In this example, we apply the above-generalized formula to evaluate the total count of faults for NCFF in the negative controlled based k-CNOT circuit. Therefore, the circuit is considered, which is shown in Figure 6 (a). The circuits' structure provides the number of* 1*-CNOT gates as* 2*, the number of* 2*-CNOT gates as* 1*, and the number of* 3*-CNOT gates as* 1*, with the input lines n*=4*. Thus, K*$_1$=2*, K*$_2$=1*, K*$_3$=1 *and n*-1=3*, in the circuit. Therefore,*

$$\sum_{i=1}^{n-1}\left[K_i \sum_{j=1}^{i}(i_{C_j})\right]$$

$$=\left[K_1\times({}^1C_1)+K_2\times({}^2C_1+{}^2C_2)+K_3\times({}^3C_1+{}^3C_2+{}^3C_3)\right]$$

$$=\left[2\times({}^1C_1)+1\times({}^2C_1+{}^2C_2)+1\times({}^3C_1+{}^3C_2+{}^3C_3)\right]$$

$$=2\times 1+1\times(2+1)+1\times(3+3+1)$$

$$=2+3+7=12$$

Hence, the circuit of Figure 6 (a) consists of a total 12 *number of NCFFs, where the occurrence of n-SCFF is* 7 *and n-MCFF is* 5*. The above-mentioned generalized formula can be used for other k-CNOT circuits to determine the total number of faults in NCFF.*

4. ATPG ALGORITHM FOR THE GENERATION OF THE CTS

This section provides the ATPG algorithm to generate the CTS for evaluating the detection of the NCFF. At first, we consider the *k*-CNOT circuits that contain only negative control connections, and the detection logic for NCFF has been formulated. The CTS has been generated using detection logic for the NCFF and an ATPG algorithm for the same has been discussed. Prior to delving into the details of the proposed work, let's establish some key definitions that will be utilized in the detection logic and CTS generation process within the proposed methodology. Moreover, Table 2 describes the specific notations that are used in our proposed work.

Definition 1 *T_{lp} is denoted as a local test pattern which consists of binary inputs ⟨$b_1b_2 \ldots b_n$⟩. These inputs are sequentially applied in the backward direction to each k-CNOT gate to facilitate the testing of faults within the gate. Let us consider, T_{lp}=⟨$b_1b_2 \ldots b_n$⟩, where b_i represents the i^{th} position of the bit that indicates to the i^{th} line and n is the total count of inputs for* $1 \le i \le n$*, $b_i \in \{0,1\}$. The T_{lp}=⟨$b_1b_2 \ldots b_n$⟩ is generated by the rule of detection logic for different faults under the CFF and extracts the necessary test vector of the circuit in the initial stage through a backpropagation process.*

Definition 2 *The test set TS_{ncf} is considered the CTS to identify all the NCFFs (both n-SCFF and n-MCFF) in the specified reversible circuit. Let us assume the CTS be TS_{ncf}=$\{TV_1, TV_2, \ldots, TV_l\}$, for* $1 \le i \le l$ *and TV_i=⟨$b_{1i}b_{2i} \ldots b_{ni}$⟩, where bji denotes the j^{th} bit of i^{th} test vector; $b_{ji} \in \{0, 1\}$.*

Table 2 Notations and their meaning for the proposed work

Notations	Meaning
k-CNOT gate	k number of control connections are present in the k-CNOT gate
k-CNOT circuit	comprises of linear cascading k-CNOT gates only
n-SCFF	negative single control flipping fault under NCFF model
n-MCFF	negative multiple control flipping fault under NCFF model
$\boldsymbol{I_i}$	i^{th} input line of the circuit
$\boldsymbol{L_i}$	i^{th} level of the given circuit
$\boldsymbol{g_i}$	i^{th} gate in the given circuit
$\boldsymbol{n}$	total count of input lines in a circuit
$\boldsymbol{N}$	total count of k-CNOT gates in a circuit
$\boldsymbol{T_{lp}}$	local test pattern that contains a set of binary inputs $\langle b_1 b_2 \ldots b_n \rangle$, where $\langle b_i \rangle \in \{0\}$, for $1 \leq i \leq n$
$\boldsymbol{TV_i}$	i^{th} count of test vectors in TS_{ncf} that contains a set of binary inputs $\langle b_1 b_2 \ldots b_n \rangle$, where $\langle b_i \rangle \in \{0,1\}$, for $1 \leq i \leq n$
$\boldsymbol{TS_{ncf}}$	complete test set for NCFFs in a k-CNOT circuit

4.1. Detection Logic for NCFF

The local test pattern T_{lp} is generated using the rule of detection logic for NCFF. The detection logic for NCFF is that the unflipped negative control connection(s) is designated by the logical digit 0, flipped negative control connection(s) is assigned by the logical digit of either 0 or 1, i.e., don't care (×), and the target connection and unconnected connection(s) are also assigned by don't-care (×). In proposed detection logic for NCFF, we consider the $T_{lp}=\langle b_1\ b_2\ .\ .\ .\ b_n \rangle$, for $1 \leq i \leq n$, $b_i \in \{0\}$ without violating the constraints of fault detection logic for NCFF. This indicates that the logic value '0' is assigned to the negative control connection(s), unconnected connection(s), and the target connection for each k-CNOT gate in the circuit. The extensive illustration of the fault identifying logic for NCFF is discussed in Example 5.

Example 5 *Figure 7 demonstrates the detection logic for NCFF. The detection logic is applied to the output line* O_1 *(i.e., the process starts from the output to input due to the backpropagation) that is assigned by the logic value* 0 *because the input line labeled as* I_1 *is linked to the negative* (◦) *control connection. Similarly, the output line* O_2 *and* O_3 *are set by the logical value* 0 *for the input line* I_2 *and* I_3*, respectively, as depicted in Figure 7* (a). *The fourth line of input* I_4 *is connected to the unconnected connection n* (*i.e., no negative control* (◦) *and target connections* (⊕)) *that is assigned by a don't care* (×) *logic value* 0 *or* 1. *As per our earlier discussion of fault detection logic for NCFF, we consider the output line* $O_4=0$. *The fifth line of input* I_5 *relates to the target connection* (⊕) *and output line* O_5 *is assigned by the don't-care logic value, i.e., either* 0 *or* 1. *Here, we consider the output line* $O_5=0$. *Finally, we formulate the* $T_{lp}=\langle O_1O_2O_3O_4O_5 \rangle=\langle 0\ 0\ 0\ 0\ 0 \rangle$ *for the NCFF using the corresponding logic values of the output lines* O_1, O_2, O_3, O_4 *and* O_5, *as depicted in Figure 7 (a). The generated* $T_{lp}=\langle 0\ 0\ 0\ 0\ 0 \rangle$ *is applied to the k-CNOT gate; it back propagates toward the input side and perform the k-CNOT gate operation to retrieve the initial input test vector* $TV_i=\langle O_1O_2O_3O_4O_5 \rangle=\langle 0\ 0\ 0\ 0\ 1 \rangle$ *in the fault-free gate, where* $O_1=I_1=0$, $O_2=I_2=0$, $O_3=I_3=0$, $O_4=I_4=0$ *and* $O_5=\bar{O}_1.\bar{O}_2.\bar{O}_3 \oplus O_5=\bar{0}.\bar{0}.\bar{0} \oplus 0=I_5=1$. *However, the input test vector* $TV_i=\langle 0\ 0\ 0\ 0\ 1 \rangle$ *that extracts from the* $T_{lp}=\langle 0\ 0\ 0\ 0\ 0 \rangle$ *using backpropagation is applied to the faulty k-CNOT gate, where an n-SCFF* (*i.e., negative*

control (◦) *flips to positive control* (•)) *occurs at the input line* I_1, *as shown in Figure 7 (b). Due to the occurrence of an n-SCFF, the faulty k-CNOT gate generates the output test vector* $\langle O_1O_2O_3O_4O_5\rangle$=⟨0 0 0 0 1⟩, *where* I_1=O_1=0, I_2=O_2=0, I_3=O_3=0, I_4=O_4=0 and I_5=$I_1.\bar{I}_2.\bar{I}_3 \oplus I_5$=$0.\bar{0}.\bar{0} \oplus 1$=$O_5$=1, *instead of the fault-free output vector* ⟨0 0 0 0 0⟩. *In Figure 7 (c), the n-MCFF* (*i.e., more than one negative controls* (◦) *flips to positive control connections* (•)) *is occurred at the input line* I_1 *and* I_2. *Due to the presence of the n-MCFF, the output vector* ⟨0 0 0 0 1⟩ *is generated by applying the same input test vector* TV_i=⟨0 0 0 0 1⟩ *in the similar process, as discussed above. Based on these analysis, it ensures that the generated* T_{lp}=⟨0 0 0 0 0⟩ *by the rule of detection logic for NCFF can differentiate the faulty and fault-free behavior of NCFF in given k-CNOT gate.*

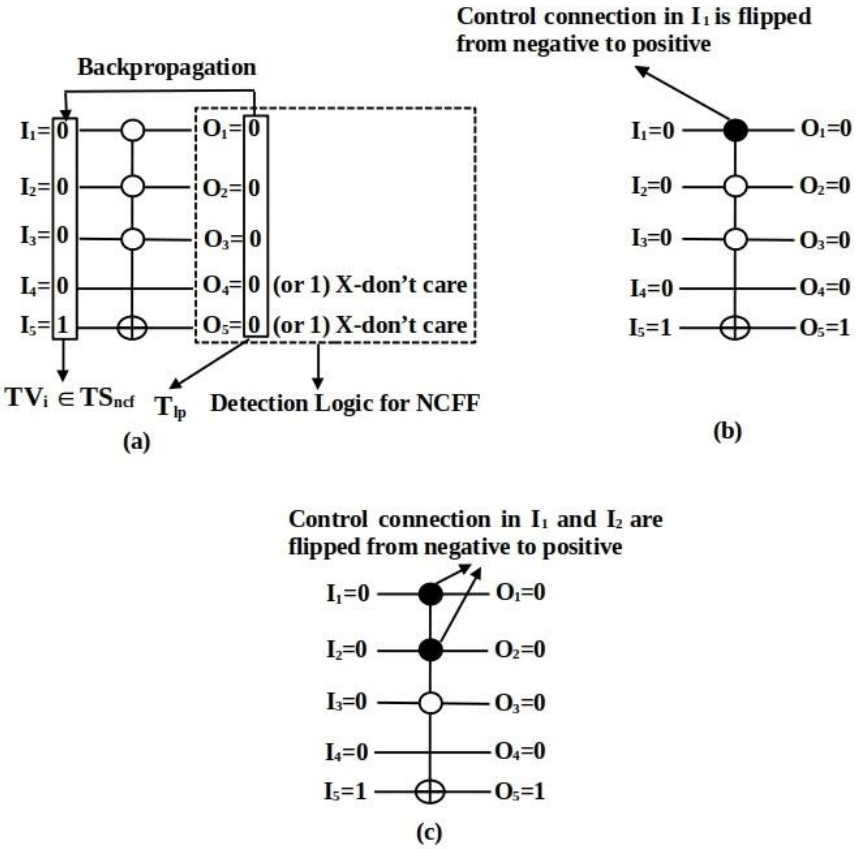

Fig. 7 (a) Detection logic of NCFF (b) Detection logic is applied in *n*-SCFF (c) Detection logic is applied in *n*-MCFF

4.2. Generation of Complete Test Set TS_{ncf} for NCFF

This section briefly discusses the TS_{ncf} generation using the fault detection method for NCFF in k-CNOT circuit. For this purpose, we consider k-CNOT circuits that include negative control connection(s), along with the target connection and unconnected connection(s). As per our earlier discussion, a reversible k-CNOT gate may have a negative control flipping at any (or more) control connection(s) that leads to the occurrence of n-SCFF or n-MCFF in NCFF. In this context, an ATPG algorithm has been introduced to generate the complete test set TS_{ncf}, aimed at identifying all possible Negative Control Flipping Faults (NCFFs) in a given circuit. To perform the testing process for NCFF, we have used positive controlled based k-CNOT circuits made to negative controlled based k-CNOT circuits. Also, the constant input and garbage output lines are assumed to be normal input and output lines, respectively.

According to the fault detection technique for NCFF, we apply the local test pattern $T_{lp}=\langle b_1\ b_2\ \ldots\ b_n\rangle$, for $1 \le i \le n$, $b_i \in \{0\}$ to each signal line of k-CNOT gate in circuit. Once T_{lp} is assigned, the backpropagation process begins from the current level and proceeds toward the circuits initial input level to obtain the TS_{ncf}. The construction of TS_{ncf} is outlined in Algorithm 1.

ALGORITHM 1 TS_{NCF} GENERATION FOR NCFFS

Input: Reversible Circuit comprises of negative-controlled based k-CNOT gates.
Output: Generate the $\boldsymbol{TS_{ncf}}$ for all NCFFs.
1 Extract $\boldsymbol{n}$ and $\boldsymbol{N}$ in a given circuit
2 $\boldsymbol{T_{lp}} \leftarrow \langle b_1\ b_2\ \ldots\ b_n\rangle$, $\langle b_i\rangle \in \{0\}$, for $1 \le i \le n$
3 **for** $i \leftarrow 1$ to $\boldsymbol{N}$ **do**
4 Applied $\boldsymbol{T_{lp}}$ for each gate $\boldsymbol{g_i}$.
5 $\boldsymbol{T_{lp}}$ performs back propogation for gate $\boldsymbol{g_i}$ to retrieve $\boldsymbol{TV_i}$ at initial level $\boldsymbol{L_0}$.
6 $\boldsymbol{TS_{ncf}} \leftarrow \boldsymbol{TV_i}$
7 Generate complete test set $\boldsymbol{TS_{ncf}}$

In Algorithm 1, the formation of the local test pattern $T_{lp}=\langle b_1 b_2\ \ldots\ b_n\rangle$ needs constant time $O(1)$ for assigning the logic value $b_i=0$ for n number of input lines of the circuit, where the value of n is provided in the early stage of the input of the algorithm. The test vector TV_i is produced by back propogate fixed $T_{lp}=\langle b_1 b_2\ \ldots\ b_n\rangle$ for each corresponding gate g_i. Therefore, the back propagation process is dependent on the N number of gates that are used in the circuit, which takes linear time. Consequently, the computational efficiency of Algorithm 1 is O(N) in terms of the time complexity in the worst-case. In the context of the space complexity, the Algorithm 1 requires $O(n \times N)$.

Example 6 *The benchmark ham3_tc circuit is illustrated to elucidate Algorithm* 1 *for generating the step-by-step processes for generating the CTS* TS_{ncf} *with the help of Figure* 8. *From Step* 1*, the inputs* n=3 *and* N=4 *are retrieved from negative-controlled based ham3_tc circuit. In Step* 2*, each of the input line* I_i *of the ham3_tc circuit is assigned by the bit value* $\langle b_i\rangle$=0*, as per the fault detection logic of NCFF for constructing the* T_{lp}*, which is discussed in the previous section. The constructed* $T_{lp}=\langle 0\ 0\ 0\rangle$*, for* $1 \le i \le 3$. *In Step* 3*, we are considering every gate* g_i*, for* $1 \le i \le 5$ *(i.e.,* N=5*), that is present in the ham3_tc circuit, as shown in Figure* 8. *In Step* 4*, the* $T_{lp}=\langle 0\ 0\ 0\rangle$ *is applied to every gate* g_i. *In Step* 5*,* $T_{lp}=\langle 0\ 0\ 0\rangle$ *is back propogated for each gate* g_i *to extract the* TV_i *at the* L_0 *in*

the ham3_tc circuit. The test vectors TV_1=⟨1 0 0⟩, TV_2=⟨0 1 0⟩, TV_3=⟨0 0 1⟩, TV_4=⟨0 1 0⟩ *and* TV_5=⟨0 1 1⟩ *are generated at level* L_0 *by the back propogation for the gates* g_1, g_2, g_3, g_4 *and* g_5, *respectively, as depicted in Figure* 8. *In Step* 6, *the generated test vectors* TV_1, TV_2, TV_3, TV_4 *and* TV_5 *are assigned to the test set* TS_{ncf}. *Finally,* TS_{ncf}= {100, 010, 001, 011} *is generated to detect all possible NCFFs that are appeared in ham3_tc circuit, which is shown in Step* 7 *of Figure* 8.

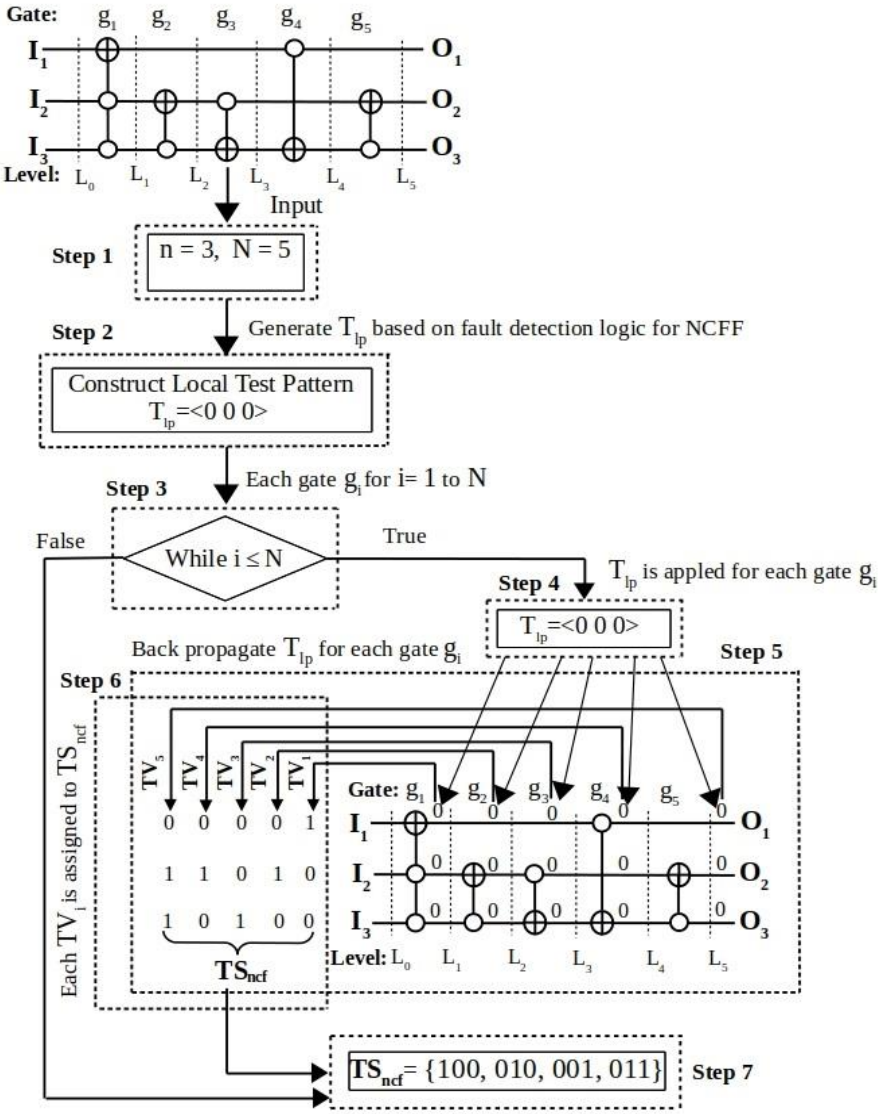

Fig. 8 Demonstration of Algorithm 1 for NCFF with the help of the negative controlled based *ham*3_*tc* circuit

Example 7 *Let us consider, the negative-controlled based k-CNOT full adder circuit to demonstrate and verify our fault detection method for NCFF, which is illustrated in Figure* 9*. Figure* 9 *(a) depicts the fault-free full adder circuit; the input test vector* ⟨0 1 0 0⟩ *is applied and the final output vector* ⟨0 0 1 1⟩ *at level* L_4*. Figure* 9 *(b) is demonstrated that an n-SCFF transpires at the gate* g_3 *in the input line* I_2 *(denoted as a dotted box), where negative control* (◦) *flips to the positive control* (•)*. According to our proposed method, the local test pattern* T_{lp}=⟨0 0 0 0⟩ *is applied on gate* g_3*, as shown in Figure* 9 *(b) and input vector* ⟨0 1 0 0⟩ *is extracted at* L_0 *by using the backpropagation. Here, it is observed that the input vector* ⟨0 1 0 0⟩ *is applied to both fault-free (Figure* 9 *(a)) and faulty (Figure* 9 *(b)) circuits. As a result, the primary output* ⟨0 0 1 1⟩ *of the fault-free circuit is not matched with the primary output* ⟨0 0 1 0⟩ *of the faulty circuit due to the presence of an n-SCFF in the negative-controlled based k-CNOT full adder circuit. Therefore, the generated test vector* ⟨0 1 0 0⟩ *at the initial level of the circuit, which is derived from the proposed method, can detect the n-SCFF. Similarly, the input vector* ⟨0 0 0 0⟩ *is extracted by using back propagates* T_{lp}=⟨0 0 0 0⟩ *at gate* g_1 *by our proposed method, which is shown in Figure* 9 *(d). Now, the input test vector* ⟨0 0 0 0⟩ *is applied to the fault-free circuit ((Figure* 9 *(c)) and faulty circuit ((Figure* 9 *(d)); the faulty primary output is generated* ⟨0 1 0 0⟩ *in faulty circuit as contrasted to the fault-free primary output* ⟨0 1 0 1⟩ *of the fault-free circuit. This takes place due to the occurrence of an n-MCFF at gate* g_1 *of the input lines* I_1 *and* I_2 *(denoted as dotted box), as shown in Figure* 9 *(d). Therefore, the* TV_i *generated in the* TS_{ncf} *by the proposed method are adequate for detecting the NCFFs (i.e., both n-SCFF and n-MCFF).*

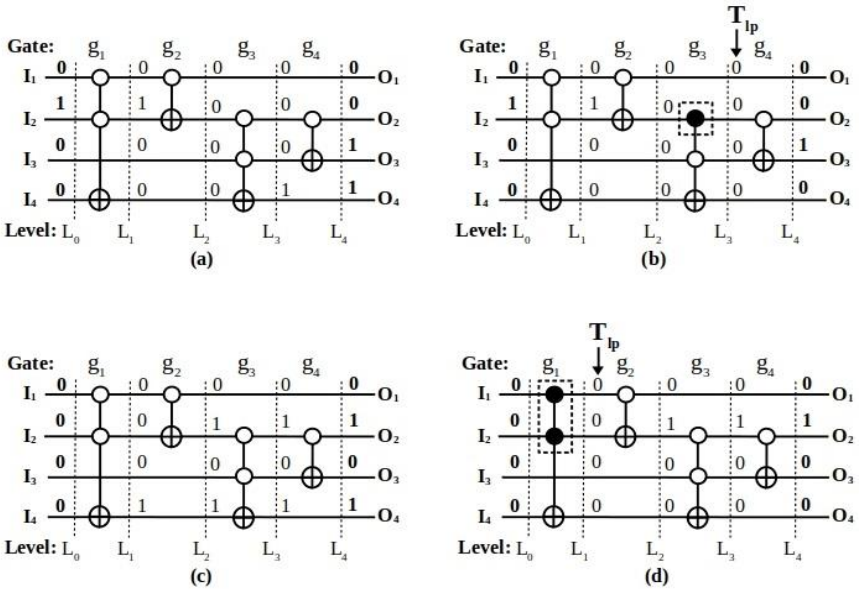

Fig. 9 Demonstrate of NCFFs: (a) fault-free full adder circuit to illustrate *n*-SCFF (b) an *n*-SCFF occurs at gate g_3 of I_2 line (c) fault-free full adder circuit to illustrate *n*-MCFF (d) an *n*-MCFF occurs at gate g_1 of I_1 and I_2 lines

Lemma 1 *The complete test set TS_{ncf} produced by the introduced method can detect all the NCFFs for an n-input k-CNOT circuit.*

Proof: According to the fault detection technique for NCFF, we consider the T_{lp}=⟨b_1 b_2 . . . b_n⟩, where $1 \le j \le n$; $bj \in 0$ and apply to each signal line of gate g_i, where $1 \le i \le N$ and N is the total number of gates that are appeared in a given k-CNOT circuit. Suppose, we consider the gate g_i is formulated with one target connection (⊕) at line In, one unconnected connection at line I_{n-1} and the remaining connection lines $\{I_1, I_2, \ldots, I_n\} \setminus \{I_{n-1}, I_n\}$ are negative control connections (◦). Let us consider the gate g_i is fault-free from NCFFs, then the T_{lp}=⟨b_1 b_2 . . . b_n⟩ generates the input test vector of g_i as ⟨b_1 b_2 . . . b_{n-1}($\bar{b}_1$ $\bar{b}_2$ . . . $\bar{b}_{n-2}$ ⊕ b_n)⟩=⟨b_1 b_2 . . . b_{n-1}(1 ⊕ b_n)⟩=⟨b_1 b_2 . . . $\bar{b}_n$⟩ with the help of backpropagation. This indicates that the target connection undergoes inversion only when all negative control connections are assigned the logic value 0. Due to this reason, the input test vector ⟨b_1 b_2 . . . $\bar{b}_n$⟩ generates the output test vector ⟨b_1 b_2 . . . b_n⟩ of the fault free gate g_i. In contrast, if there are NCFFs (i.e., an n-SCFF or n-MCFF) at the gate g_i, any of the negative control connection line(s) $I_k \in \{I_1, I_2, \ldots, I_n\} \setminus \{I_{n-1}, I_n\}$, where $1 \le k \le n - 2$, is switched negative (◦) to positive control connection (•). Now, the same input test vector ⟨b_1 b_2 . . . $\bar{b}_n$⟩ that extracts from the T_{lp}=⟨b_1 b_2 . . . b_n⟩ is applied to the faulty gate g_i and generates the output test vector as ⟨b_1 b_2 . . . b_k . . . b_{n-1}($\bar{b}_1$ $\bar{b}_2$ . . . b_k . . . $\bar{b}_{n-2}$ ⊕ $\bar{b}_n$)⟩= ⟨b_1 b_2 . . . b_k . . . b_{n-1}(0 ⊕ $\bar{b}_n$)⟩=⟨b_1 b_2 . . . $\bar{b}_n$⟩. Therefore, the generated output test vector ⟨b_1 b_2 . . . $\bar{b}_n$⟩ of the faulty gate g_i is different as compared to output test vector ⟨b_1 b_2 . . . b_n⟩ of the fault-free gate g_i . By applying the T_{lp}=⟨b_1 b_2 . . . b_n⟩ for each individual k-CNOT gate to obtain the test set TS_{ncf} , we extract all input test vectors at the circuit's initial level. Consequently, TS_{ncf} serves as a comprehensive test set for the detection of all potential NCFFs, including n-SCFF and n-MCFF, within the given circuit.

4.3. Correlation of NCFF with the Existing Fault Models in Reversible Circuits

To establish the correlation and evaluate the fault coverage range with the existing fault models in reversible circuits, we have correlated the NCFF in the CFF fault model with other fault models, like the SMGF, MMGF and PMGF. The following sections discuss the correlation of different fault models with the NCFF.

4.3.1. Correlation between NCFF with SMGF

An SMGF model structure exists when one k-CNOT gate is completely missing or disappears in the circuit due to the operating gate signal being short, missing, misaligned, or mistuned [33, 19]. The measurement of the total count of SMGFs is dependent on the count of gates N of a circuit. Let us consider *ham*3*_tc* circuit, as depicted in Figure 8, where total number of k-CNOT gate is 5 (i.e., N=5). Therefore, the total count of SMGFs is 5 in *ham*3*_tc* circuit.

As per our previous discussion, NCFF occurs in the negative controlled based k-CNOT circuits. In this context, if any SMGF occurs in negatively controlled k-CNOT circuits, the fault detection logic dictates that all negative control connections must be assigned the logic value 0. The unconnected connection(s) and the target connection are assigned by the arbitrary logical value of either 0 or 1 (i.e., don't-care condition). Similarly, our proposed work applies the same fault detection logic to the NCFF. Thus, the constructed complete test set TS_{ncf} for NCFF can identify all the possible SMGFs in a

given negative controlled based *k*-CNOT circuit. It is noted that the *k*-CNOT circuits do not contain the general form of any complete negative control *k*-CNOT gates. However, in order to check the validity of NCFF, all positive controlled based *k*-CNOT circuits are converted into the negative controlled based *k*-CNOT circuits. To analyze the fault coverage range for an SMGF, we considered the TS_{ncf} for NCFF that can detect all possible SMGFs that appeared in a given negative controlled based *k*-CNOT circuit, and it is shown in Section 5.

4.3.2. Correlation between NCFF with MMGF

As per the definition of MMGF, if two or more succeeding numbers of *k*-CNOT gates disappear in the circuit, then it is considered an MMGF [33,19]. The authors in [19] mentioned that SMGFs are a subset of MMGFs. Therefore, the total count of MMGFs can be calculated as $N(N+1)/2$ [19], where N denotes the total number of *k*-CNOT gates that appeared in a given circuit. If we measure only the more than two consecutive missing *k*-CNOT gates (excluding the SMGFs), it can be evaluated as $N(N-1)/2$. However, the complete test set for SMGF does not guarantee coverage of all MMGFs [19]. As a result, supplementary test vectors are required alongside the SMGF test set to detect MMGFs. While the detection logic for an NCFF can identify all potential SMGFs in negatively controlled *k*-CNOT circuits, the TS_{ncf} may not be sufficient to detect all possible MMGFs

4.3.3. Correlation between NCFF with PMGF

A PMGF manifests in *k*-CNOT circuits when one or more than one control connections are absent or disappear [19]. If a single control connection is missing in a *k*-CNOT gate within the circuit, it is classified as a first-order PMGF. [19]. In contrast, if more than one control connection is absent, it denotes a higher-order PMGF. The work in [19] showed that any test vector capable of detecting the first-order PMGF can also detect higher-order PMGF. Therefore, we consider the first-order PMGF in our work. The total number of fault measuring of the PMGFs can be evaluated as the total count of control connections present in the *k*-CNOT circuit. For example, the total count of control connections is 6 in the *ham*3_*tc* circuit. Thus, the total count of PMGFs is 6 in the *ham*3_*tc* circuit.

In a negatively controlled *k*-CNOT circuit, the detection logic for NCFF is as follows: a logic value of 1 is assigned to the missing control connection(s), a logic value of 0 is assigned to every other control connections, and the rest of the unconnected connection(s) and target connection are arbitrarily set to a logical value of either 0 or 1 (i.e., don't-care). In our proposed work, the fault detection logic for NCFF provides the detection rule, where the negative control connection(s) (flipped or unflipped) is given by the logic value 0, and the target connection and unconnected connection(s) are assigned by don't-care (×). It means that the generated test vector TV_i in TS_{ncf} has not achieved 100% fault coverage for the PMGF in certain *k*-CNOT gates or entire *k*-CNOT gate that depends on the structure of the *k*-CNOT circuits.

Example 8 *Let us consider the negative controlled based benchmark circuit peres_9, as shown in Figure* 10 *(a), where all the possible PMGFs are injected as per the definition* [19] *of the PMGF model in k-CNOT circuits. Figure* 10 *(a) shows the fault-free peres_9 circuit. Here, Figure* 10 *(b) illustrates that the negative control* (◦) *connection at* I_2 *input line of gate* g_1 *is missing, which is denoted as dotted box. The other PMGF occurs in the*

negative control (◦) *connection at* I_3 *input line of gate* g_1 *is missing (denoted as a dotted box), which is shown in Figure* 10 *(c). Finally, the last PMGF of the peres_*9 *circuit occurs at* I_3 *line of* g_2 *gate (denoted as a dotted box), as shown in Figure* 10 *(d). Therefore, the total number of faults in PMGF is* 3 *for the peres_*9 *circuit. As per our proposed Algorithm* 1*, the complete test set* TS_{ncf}={⟨0 1 0⟩, ⟨1 0 0⟩} *for NCFF is applied to evaluate the fault coverage of PMGFs in peres_*9 *circuit. Table 3 shows the fault coverage of all available PMGFs in the peres_*9 *circuit by the complete test set* TS_{ncf}*. In Table 3, it is noticed that the test vector* ⟨0 1 0⟩ *in* TS_{ncf} *is distinguished the fault-free and faulty output (denoted as bold text) of the PMGF occurs in Figure* 10 *(b). The remaining PMGFs are unable to detect by the test set* TS_{ncf}={⟨0 1 0⟩, ⟨1 0 0⟩}*. Therefore, the fault coverage percentage is* 33.33% *in the peres_*9 *circuit, which is achieved by* TS_{ncf} *of our proposed method.*

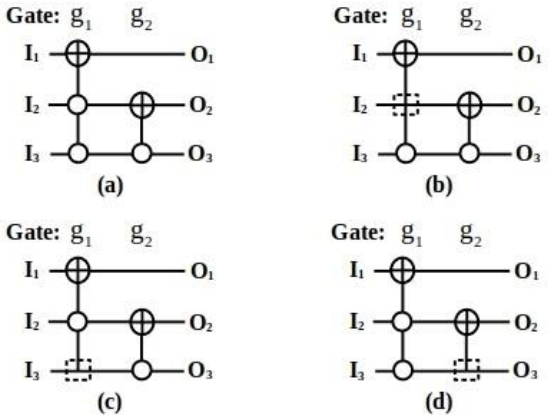

Fig. 10 Illustration of PMGF fault coverage in *peres*_9 circuit: (a) fault-free circuit (b) PMGF at I_2 in gate g_1 (c) PMGF at I_3 in gate g_1 (d) PMGF at I_3 in gate g_2

Table 3 PMGF fault coverage by the TS_{ncf} in *peres*_9 circuit

TS_{ncf}	Fault-free output	Faulty outputs		
	Figure 10 (a)	PMGF in Figure 10 (b)	PMGF in Figure 10 (c)	PMGF in Figure 10 (d)
010	000	100	000	000
100	010	010	010	010
Fault Coverage		1	0	0

5. Experimental Results and Discussions

This section offers experimental results and analysis of the introduced approach to generate a complete test that detects NCFFs in *k*-CNOT circuit are discussed. Also, we have depicted the fault coverage range in comparison to other existing fault models. The reversible benchmark circuits in our experimental results are taken from [41,42]. For implementation purposes, we have considered the machine-readable version (i.e., *tfc* file

format) posted on [41] for each benchmark circuit. The proposed ATPG algorithm is implemented in Python 3.12. The programs were executed on a system with an Ubuntu 24.04.1 LTS, Processor: AMD Ryzen™ 7 8845HS w/ Radeon™ 780M Graphics×16 with 16.0 GB RAM. As per our prior discussion, the standard benchmark *k*-CNOT circuits do not contain the general form of complete negative control *k*-CNOT gates. Here, all positive controlled based *k*-CNOT circuits are converted into the negative controlled based *k*-CNOT circuits to implement the proposed TS_{ncf} generation algorithm for the NCFF and verify the experimental evaluation. The experimental results for the TS_{ncf} to identify the NCFFs are demonstrated in Table 4, where the columns 1 to 6 furnish information, including the circuit name, the total number of input lines (*n*), the total number of gates (*N*), the total number of *n*-SCFFs, the total count of *n*-MCFFs and the total count of faults of both *n*-SCFFs and *n*-MCFFs, respectively. Columns 7 and 8 of Table 4 represent the total count of test vectors in TS_{ncf} used for detecting the NCFFs and the CPU time taken for producing the complete test set TS_{ncf}, respectively. On comparing the circuit *peres*_9 with the circuit *ham*7_21_69, the test set size of *peres*_9 and *ham*7_21_69 circuits are 2 and 15, respectively. For the *peres*_9 circuit, the total count of gates and faults (both *p*-SCFF and *p*-MCFF) are 2 and 4, respectively, whereas the *ham*7_21_69 circuit are 21 and 37. Thus, the size of the TS_{ncf} for the *peres*_9 circuit is less as compared to the size of the TS_{ncf} for the *ham*7_21_69. Similar observations are seen in other benchmark circuits also. Moreover, the CPU time, as shown in Column 8 for the generation of TS_{ncf} in NCFF, has been found to be dependent mainly on the gates count (*N*) and faults in the circuit. It means that a large count of gates and count of faults in the circuit has increased the CPU time in the maximum cases presented in Table 4.

Table 5 demonstrates the fault coverage results of the currently established fault models, including SMGF, MMGF and PMGF within reversible circuits. In general, fault coverage is calculated as the ratio of the number of faults detected by the testing approach to the total number of faults present in the given circuit. In our proposed work, the fault coverage range is calculated as the total count of faults within existing fault models that can be covered by the generated complete test set TS_{ncf}. Let us consider the benchmark circuit *ex*_1_166, the total number of PMGFs is 4, which is shown in column 6 of Table 5. As per our proposed method, the test set TS_{ncf}={⟨0 1 0⟩, ⟨0 0 0⟩, ⟨1 0 0⟩} for the *ex*_1_166 circuit is detected only 3 number of actual faults in PMGF. Therefore, the fault coverage percentage of PMGF is 75% for the *ex*_1_166 circuit by the generated test set TS_{ncf}, as shown in column 9 of Table 5. In Table 5, columns 1 to 3 are presented with the various benchmark circuit names, the number of input (*n*), and the number of gates (*N*), respectively. The total count of possible faults, like SMGF, MMGF, and PMGF, is represented by columns 4, 5, and 6 of Table 5, respectively. The constructed complete test set TS_{ncf} for the NCFF is applied to perform the fault coverage of the SMGF, MMGF and PMGF, which are shown in columns 7, 8 and 9, respectively. By observing the experimental results, we found that TS_{ncf} by our proposed method, is capable of covering 100% fault coverage for the SMGF. In our introduced method, the local test pattern T_{lp} is derived using the fault detection logic of NCFF for each *k*-CNOT gate. The back propagates T_{lp} for each *k*-CNOT gate to extract the required test vector at the initial input level of the circuit. Therefore, each T_{lp} covers every *k*-CNOT gate and their corresponding test vector at the initial input level covers all possible SMGFs. Hence, the test set TS_{ncf} acquires 100% fault coverage on an average for SMGF.

Table 4 Generation of Complete test set TS_{ncf} for detecting the NCFFs

Benchmark Circuit	n	N	Total No. of Faults	Total No. of Faults	Total No. of Faults (NCFFs)	Size of TS_{ncf}	CPU Time (sec)
			n-SCFFs	n-MCFF	(n-SCFFs+n MCFFs)	(n-SCFFs+n-MCFFs)	TS_{ncf}
peres_9	3	2	3	1	4	2	0.00000
3_17_14	3	6	7	2	9	4	0.00001
3_17_13	3	6	7	2	9	4	0.00001
ex-1-166	3	4	4	1	5	3	0.00001
fredkin_6	3	3	6	3	9	3	0.00001
ham3	3	5	6	1	7	4	0.00013
miller_11	3	5	8	3	11	4	0.00011
nth_prime3_inc	3	4	5	1	6	4	0.00011
4b15g_1	4	14	19	9	28	9	0.00023
4b15g_2	4	15	21	14	35	7	0.00071
4b15g_3	4	15	20	12	32	8	0.00046
4b15g_4	4	15	19	12	28	9	0.00037
4b15g_5	4	15	19	7	26	8	0.00039
4_49d3	4	12	15	5	20	8	0.00028
hwb4-11-21	4	11	14	3	17	8	0.00019
hwb4d1	4	17	27	14	41	8	0.00042
hwb4d3	4	11	36	25	61	8	0.00172
mspk_4b15g_1	4	15	20	8	28	10	0.00132
mspk_4b15g_2	4	15	19	5	24	9	0.00091
mspk_4b15g_3	4	13	18	5	23	11	0.00082
mspk_hwb4_12	4	12	15	3	18	10	0.00025
mspk_hwb4_13	4	13	16	3	19	9	0.00066
mspk_nth_primes4_11	4	11	20	13	33	7	0.00137
mspk_nth_primes4_12	4	12	18	6	24	7	0.00067
mspk_nth_primes4_13	4	13	19	6	25	10	0.00102
nth_primes5_inc_29_91	5	29	22	13	35	18	0.00432
hwb5_31_91	5	31	42	17	59	17	0.00639
ham_7_21_69	7	21	24	13	37	15	0.01232
ham_7_25_49	7	25	31	6	37	24	0.01784
hwb8-637	8	637	2214	18361	20575	160	1.29176
hwb9-1544	9	1544	5812	60944	66756	358	2.60100
ham15-70	15	70	70	208	278	53	1.38220
ham15-109-214	15	109	126	301	427	94	3.02353
ham15tc1	15	132	354	2462	2816	68	3.08315

Column 8 of Table 5 designates the range of fault coverage of MMGF by the generated test set TS_{ncf}. Here, it is noticed that the test set TS_{ncf} by our proposed method has achieved 100% fault coverage for 63% of the benchmark circuits, and the remaining 37% benchmark circuits achieved 81.77% fault coverage on an average for the MMGF. The lowest fault coverage range of the MMGF is 63.12% for the circuit *miller*_11, based on the experimental results. In the last row of Table 5, it is perceived that the percentage of fault coverage difference between SMGF and MMGF is 6.72%. The coverage of fault of an PMGF by the test set TS_{ncf} of our work is shown in column 9 of Table 5. Here, TS_{ncf} attains 100% fault coverage of PMGF for the circuits 3_17_13, *nth_prime*3_*inc*, *hwb4*_11_21 and *mspk_nth_primes*_4_11. It is also observed that the more than or equal

to 50% fault coverage of MMGF gains by the 63% benchmark circuits used in our experimental results. As per the experimental results, the lowest fault coverage range, like 33.33%, 27.32% are found for the circuits peres_9 and 4*b*15*g*_1, respectively, for the MMGF. The overall fault coverage range is 74.12% on average for the PMGF, which is achievable by the TS_{ncf}. After detailed analyses of the fault coverage for the existing fault models, it is evident that the TS_{ncf} of the NCFF is covered 100% fault coverage for the SMGF, whereas the average number of fault coverage range is 93.28% for the MMGF and 74.12% for the PMGF by the test set TS_{ncf}. Therefore, fine-tuning is required for the TS_{ncf} to achieve complete fault coverage for MMGF and the PMGF.

Table 5 Evaluation of Fault Coverage range of SMGF, MMGF and PMGF Fault Models by the TS_{ncf}

Benchmark Circuit	*n*	*N*	No. of Faults	No. of Faults	No. of Faults	Fault Coverage(%) by TS_{ncf} [Proposed]	Fault Coverage(%) by TS_{ncf} [Proposed]	Fault Coverage(%) by TS_{ncf} [Proposed]
			SMGF	MMGF	PMGF	SMGF	MMGF	PMGF
peres_9	3	2	2	1	3	100	100	33.33
3_17_14	3	6	6	7	5	100	100	57.14
3_17_13	3	6	6	7	5	100	100	100
ex_1_166	3	4	4	4	4	100	100	75
fredkin_6	3	3	3	6	0	100	100	50
miller_11	3	5	5	8	2	100	63.12	75
nth_prime3_inc	3	4	4	5	3	100	100	100
4b15g_1	4	14	14	28	6	100	100	27.32
4b15g_2	4	15	15	35	24	100	88.32	65.71
4b15g_3	4	15	15	43	25	100	100	77.21
4b15g_4	4	15	15	19	26	100	72.43	84.21
hwb4_11_21	4	11	11	14	19	100	100	100
mspk_4b15g_2	4	15	15	17	26	100	84.61	88.13
mspk_hwb4_12	4	12	12	15	18	100	100	100
mspk_nth_primes_4_11	4	11	11	20	13	100	93.12	100
nth_primes4_inc_15_51	5	15	15	29	23	100	100	56.71
nth_primes5_inc_29_91	5	29	29	41	29	100	82.27	73.11
hwb5d3	5	24	24	38	27	100	100	67.33
hwb5_31_91	5	31	31	44	22	100	88.52	78.14
Average						100	93.28	74.12

The experimental analysis of our proposed method in terms of the total count of faults, test set size and fault coverage for the SMGF are compared with the existing work in [37], which is shown in Table 6. The authors in [37] presented a Boolean difference method to generate a CTS for detecting all possible SMGFs in *k*-CNOT circuits. Columns 4, 5 and 6 within Table 6 indicate the total count of faults, test set size and fault coverage by the method presented in [37], respectively. In comparison, columns 7, 8 and 9 of Table 6 represent the same parameters by our proposed method, respectively. The method in [37] focused only on detecting SMGFs; the total count of faults is 7, on average, as shown in column 4.

Table 6 Comparison of total number of faults, test set size and fault coverage with existing work [37]

Benchmark Circuit	n	N	Work in [37]			Proposed Work		
			SMGF			NNCF+SMGF+MMGF+PMGF		
			Total Faults	No. of Test Vectors	Fault Coverage (%)	Total Faults	No. of Test Vectors in TS_{ncf}	Fault Coverage (%)
ham3tc	3	5	5	3 or 5	100	28	4	95.83
rd32	4	4	4	6	100	24	4	100
xor4d1	5	4	4	2	100	18	4	100
3_17tc	3	6	6	2	100	37	4	100
mod5d1	5	8	8	2	100	59	5	100
4_49d3	4	12	12	4	100	113	8	100
hwb4d1	4	17	17	7	100	221	7	100
mod5d2	5	9	9	2	100	71	8	100
rd32d1	4	4	4	6	100	26	4	100
mod5d4	5	5	5	4	100	28	4	100
Average			7	4	100	63	6	99.58

In our proposed method, the test set TS_{ncf} covers SMGF, MMGF and PMGF, including NNCF, which leads to an average increase in the total number of faults 63, as shown in column 7. In this context, the number of faults is increased by 9X in our proposed method when contrasted to the approach in [37]. However, the test set size acquired by the proposed technique is more than that of the work in [37]. On average, the test set size in our method has increased by 50%, as compared to the work in [37], which is shown in columns 5 and 8. This occurs due to the large number of faults covered by our proposed method. Columns 6 and 9 show the fault coverage for the existing and proposed methods, respectively. On the average percentage of the fault coverage, the method in [37] and the proposed method are 100% and 99.58%, respectively. In the proposed method, the test set TS_{ncf} is explicitly constructed for NCFF, which is also tested for the fault coverage of other already available fault models in reversible circuits. Therefore, the proposed method cannot cover all the faults for some of the circuits. The test set TS_{ncf} covers only 83.33% of PMGFs for the *ham3tc* circuit; due to this, the overall fault coverage is obtained only 95.83%, as shown in column 9 of Table 6.

Table 7 tabulates the comparative analysis of our proposed method with the already available ATPG method in [19]. For the purpose of comparative analysis, we consider the same parameters, like total number of faults, test set size and fault coverage that are mentioned in the previous discussion. The work in [19] proposed an exact ATPG method to extract the complete test set for identifying SMGF, MMGF and PMGF using an ILP. Columns 4, 5 and 6 of Table 7 are provided the total number of faults, number of test vectors, and percentage in fault coverage, respectively, which are generated by the ATPG method [19]. As per the work in [19], the total count of faults is extracted by combining SMGFs, MMGFs and PMGFs that occur in a corresponding benchmark circuit, as tabulated in column 4 within Table 7.

Table 7 Comparison of total number of faults, test set size and fault coverage with existing work [19]

Benchmark Circuit	*n*	*N*	Work in [19]			Proposed Work		
			SMGF+MMGF+PMGF			NNCF+SMGF+MMGF+PMGF		
			Total Faults	No. of Test Vectors	Fault Coverage (%)	Total Faults	No. of Test Vectors in TS_{ncf}	Fault Coverage (%)
2of5d1	6	18	195	11	100	272	9	100
2of5d2	7	12	97	4	100	123	8	100
3_17tc	3	6	28	3	100	37	4	100
4_49tc1	4	16	160	5	100	197	6	100
5mod5tc	6	17	191	7	100	292	7	100
6symd2	10	20	243	4	100	289	4	91.54
9symd2	12	28	454	-	-	522	13	100
ham3tc	3	5	21	3	100	28	4	95.83
ham7tc	7	24	334	4	100	381	19	100
hwb4tc	4	17	180	6	100	221	7	100
hwb5tc	5	56	1697	9	100	1852	20	93.66
hwb6tc	6	126	8134	16	100	9973	23	95.30
mod5adders	6	21	258	8	100	321	5	100
mod5d1	5	8	46	4	100	59	5	100
mod5d2	5	9	56	3	100	71	8	100
rd32	4	4	16	3	100	24	4	100
rd53d1	7	12	106	8	100	182	12	100
rd53d2	8	12	98	4	100	126	7	100
rd53rcmg	7	30	506	10	100	599	8	88.46
rd73d2	10	20	244	4	100	292	6	100
rd84d1	15	28	455	-	-	525	14	100
xor5d1	5	4	14	2	100	18	4	100
Average			615	6	100	745	9	98.39

In contrast, the total count of faults is determined based on the NCFFs, SMGFs, MMGFs and PMGFs in our proposed work, which is shown in column 7 of Table 7. By comparative analysis of the total count of faults for both methods it is observed that the percentage of total detectable faults increase on average is 21.13% in our proposed method, as compared to the total count of faults in an existing ATPG method [19]. Columns 5 and 8 of Table 7 tabulated the test set size which is implemented to detect the faults for the ATPG method [19] and the proposed method, respectively, where dashed indicates no result is generated for the particular parameter of their corresponding circuit. Here, it noticed that the average percentage of the test set size is 50% decreased of the existing ATPG method [19], as compared with the proposed method. It indicates that more test vectors in TS_{ncf} are required to detect the NCFFs, along with the existing faults SMGFs, MMGFs and PMGFs in our proposed method. The fault coverage is shown 100% by the existing method in [19], as shown in column 6 of Table 7. On average, the fault coverage of the percentage difference is 1.61% for both methods, i.e., the fault coverage range is 98.39% of the proposed method, which is slightly lower than the existing ATPG method [19]. It happens due to the fault coverage range of PMGFs by the proposed method is 66.17%, 83.33%, 74.67%, 81.21% and 53.87% for the benchmark circuits *6symd2*, *ham3tc*, *hwb5tc*, *hwb6tc* and *rd53cmg*, respectively.

6. CONCLUSION

This paper presents a fault detection approach that identifies the negative control flipping fault (NCFF) in *k*-CNOT circuits. According to our proposed methodology, the constructed complete test set TS_{ncf} has the capability to identify all types of NCFFs in *k*-CNOT-based circuits. For this motive, a fault detection logic is developed based on the logical operation of the negative control(s) in *k*-CNOT gates and constructed the TS_{ncf} for detecting the NCFF by the proposed algorithm. Furthermore, a fault coverage analysis has been performed with the existing fault models to verify the efficiency of our proposed complete test set construction method for the NCFF. Initially, the complete test set TS_{ncf} is constructed only for the detection of NCFF, and later, the TS_{ncf} is applied to analyze the fault coverage range for the SMGF, MMGF, and PMGF. By observing the experimental results, TS_{ncf} obtained an average of 100%, 93.28%, and 74.12% fault coverage for the SMGF, MMGF, and PMGF, respectively. The results show that the complete test set for NCFF ensures full fault coverage for SMGF in *k*-CNOT circuits with negative controls. In contrast, the fault coverage percentage (74.12%) for the PMGF is less as compared to the fault coverage percentage (93.28%) for the MMGF by the TS_{ncf}. Thus, there is needed to add some additional constraints that depend on the fault detection logic for MMGF and PMGF are incorporated into the TS_{ncf} for gaining complete fault coverage.

In this paper, the presented fault detection logic is developed based on the occurrence of negative controlled flipping in the *k*-CNOT circuit structure. To verify the fault detection logic for NCFF, we perform the circuit computation in classical bits (i.e., classical implementation), as the proposed work in this paper is based on reversible circuits. However, the application of the *k*-CNOT gate is widely used in quantum circuits, which is leading to future work for correlating the NCFF model with other emerging quantum faults, like decoherence faults, leakage faults, or quantum crosstalk faults on IBM Quantum hardware. Also, the proposed NCFF model may be extended to examine the absence of noise and physical error behaviors in quantum circuits.

REFERENCES

[1] R. Landauer, "Irreversibility and Heat Generation in the Computing Process", *IBM J Res. Dev.*, vol. 5, no. 3, pp. 183–191, 1961.

[2] C. H. Bennett, "Logical Reversibility of Computation", *IBM J. Res. Dev.*, vol. 17, no. 6, pp. 525–532, 1973.

[3] N. Alhagi, M. Hawash and M. Perkowski, "Synthesis of Reversible Circuits for Large Reversible Functions", *FU: Elec. Energ.*, vol. 23, no. 3, pp. 273–286, 2010.

[4] K. Fazel, M. A. Thornton and J. E. Rice. "ESOP-Based Toffoli Gate Cascade Generation" In Proceedings of the 2007 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, 2007, pp. 206–209.

[5] 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.

[6] S. Stojković, M. Stanković and C. Moraga, "Complexity Reduction of Toffoli Gates Realization of Boolean Functions Based on FDD", *FU: Elec. Energ.*, vol. 28, no. 2, pp. 251–262, 2015.

[7] C. H. Bennett, "Notes on the History of Reversible Computation", *IBM J. Res. Dev.*, vol. 44, no. 1/2, pp. 270, 2000.

[8] M. Sarkar, P. Ghosal and S. P. Mohanty, "Minimal Reversible Circuit Synthesis on a DNA computer", *Nat. Comput.*, vol. 16, no. 3, pp. 463–472, 2017.

[9] M. Rofail and A. Younes, "Synthesis Strategy of Reversible Circuits on DNA Computers", *Symmetry*, vol. 13, no. 7, pp. 1242, 2021.

[10] A. Sarker, H. M. Hasan Babu and S. M. M. Rashid, "Design of a DNA Based Reversible Arithmetic and Logic Unit", *IET Nanobiotechnol.*, vol. 9, no. 4, pp. 226–238, 2015.
[11] C. Taraphdar, T. Chattopadhyay and J. N. Roy, "Mach–Zehnder Interferometer-Based All-Optical Reversible Logic Gate", *Opt. Laser Technol.*, vol. 42, no. 2, pp. 249–259, 2010.
[12] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, 10th Anniversary Edition, 10th Edition, Cambridge University Press, New York, NY, USA, 2011.
[13] X. Ma, J. Huang, C. Metra and F. Lombardi, "Reversible Gates and Testability of One Dimensional Arrays of Molecular QCA", *J. Electron. Test.*, vol. 24, no. 1-3, pp. 297–311, 2008.
[14] R. Faraji and A. Rezai, "A Novel Reversible Multilayer Full Adder Circuit Design in QCA Technology", *Facta Universitatis: Series Electronics and Energetics,* vol. 37, no. 3, pp. 437–453, 2024.
[15] N. K. Jha and S. Gupta, *Testing of digital systems*, Cambridge University Press, 2003.
[16] J. Rice, "An Overview of Fault Models and Testing Approaches for Reversible Logic", In Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), 2013, pp. 125–130.
[17] K. N. Patel, J. P. Hayes and I. L. Markov, "Fault Testing for Reversible Circuits", *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 23, no. 8, pp. 1220–1230, 2004.
[18] H. Rahaman, D. K. Kole, D. K. Das and B. B. Bhattacharya, "Optimum Test Set for Bridging Fault Detection in Reversible Circuits", In Proceedings of the 16th Asian Test Symposium (ATS 2007), 2007, pp. 125–128.
[19] I. Polian, T. Fiehn, B. Becker and J. P. Hayes, "A Family of Logical Fault Models for Reversible Circuits", In Proceedings of the 14th Asian Test Symposium (ATS'05), pp. 422–427, 2005.
[20] R. Wille, H. Zhang and R. Drechsler, "ATPG for Reversible Circuits Using Simulation, Boolean Satisfiability, and Pseudo Boolean Optimization", In Proceedings of the 2011 IEEE Computer Society Annual Symposium on VLSI, 2011, pp. 120–125.
[21] H. Rahaman, D. K. Kole, D. K. Das and B. B. Bhattacharya, "Fault Diagnosis in Reversible Circuits Under Missing-Gate Fault Model", *Comput. Electr. Eng.*, vol. 37, no. 4, pp. 475–485, 2011.
[22] A. Nagamani, S. Ashwin, B. Abhishek and V. K. Agrawal, "An Exact Approach for Complete Test Set Generation of Toffoli-Fredkin-Peres Based Reversible Circuits", *J. Electron. Test.*, vol. 32, no .2, pp. 175–196, 2016.
[23] D. A. Maslov, "Reversible Logic Synthesis", Ph.D. Dissertation, Fredericton, N.B., Canada, Canada, aAINQ98874, 2003.
[24] R. P. Feynman, "Quantum Mechanical Computers", *Found. Phys.*, vol. 16, no. 6, pp. 507–531, 1986.
[25] T. Toffoli, "Reversible Computing", In Proceedings of the International Colloquium on Automata, Languages, and Programming, Springer, 1980, pp. 632–644.
[26] C. Moraga and F. Z. Hadjam, "The Fredkin Gate in Reversible and Quantum Environments", *FU: Elec. Energ.*, vol. 36, no. 2, pp. 253-266, 2023.
[27] E. Fredkin and T. Toffoli, "Conservative Logic", *Int. J. Theor. Phys.*, vol. 21, no. 3–4, pp. 219–253, 1982.
[28] A. Peres, "Reversible Logic and Quantum Computers", *Phys. Rev. A*, vol. 32, no. 6, p. 3266, 1985.
[29] D. Loss and D. P. DiVincenzo, "Quantum Computation with Quantum Dots", *Phys. Rev. A*, vol. 57, no. 1, pp. 120, 1998.
[30] M. Ibrahim, A. R. Chowdhury and H. M. H. Babu, "Minimization of CTS of K-Cnot Circuits for SSF and MSF Model", In Proceedings of the 2008 IEEE International Symposium on Defect and Fault Tolerance of VLSI Systems, 2008, pp. 290–298.
[31] M. Bubna, N. Goyal and I. Sengupta, "A DFT Methodology for Detecting Bridging Faults in Reversible Logic Circuits", In Proceedings of the TENCON 2007-2007 IEEE Region 10 Conference, 2007, pp. 1–4.
[32] A. Nagamani, B. Abhishek and V. K. Agrawal, "Deterministic Approach for Bridging Fault Detection in Peres-Fredkin and Toffoli Based Reversible Circuits", In Proceedings of the 2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2015, pp. 1–6.
[33] J. P. Hayes, I. Polian and B. Becker, "Testing for Missing-Gate Faults in Reversible Circuits", In Proceedings of the 13th Asian Test Symposium, 2004, pp. 100–105.
[34] J. Zhong and J. C. Muzio, "Analyzing Fault Models for Reversible Logic Circuits", In Proceedings of the 2006 IEEE International Conference on Evolutionary Computation, 2006, pp. 2422–2427.
[35] M. Zamani, M. B. Tahoori and K. Chakrabarty, "Ping-Pong Test: Compact Test Vector Generation for Reversible Circuits", In Proceedings of the 2012 IEEE 30th VLSI Test Symposium (VTS), 2012, pp. 164–169.
[36] J. Mondal, D. K. Das, D. K. Kole and H. Rahaman, "A Design for Testability Technique for Quantum Reversible Circuits", In Proceedings of the IEEE EastWest Design & Test Symposium (EWDTS 2013), 2013, pp. 1–4.

[37] B. Mondal, D. K. Kole, D. K. Das and H. Rahaman, "Generator for Test Set Construction of SMGF in Reversible Circuit by Boolean Difference Method", In Proceedings of the 2014 IEEE 23rd Asian Test Symposium, 2014, pp. 68–73.

[38] M. Handique, J. K. Deka and S. Biswas, "An Efficient Test Set Construction Scheme for Multiple Missing-Gate Faults in Reversible Circuits", *J. Electron. Test.*, vol. 33, no. 1, pp. 105–122, 2020.

[39] J. Mondal, A. Deb, and D. K. Das, "An Efficient Design for Testability Approach of Reversible Logic Circuits", *J. Circuit. Syst. Comput.*, vol. 30, no. 6, p. 2150094, 2021.

[40] M. Handique, and H. K. D. Sarma, "Testable Design for Positive Control Flipping Faults in Reversible Circuits", *Indones. J. Electr. Eng. Inform. (IJEEI)*, vol. 11, no. 2, pp. 416–430, 2023.

[41] D. Maslov, Reversible logic synthesis benchmarks page, 2015. [Online]. Available at: http://webhome.cs.uvic.ca/dmaslov

[42] R. Wille, D. Große, L. Teuber, G. W. Dueck and R. Drechsler, "Revlib: An Online Resource for Reversible Functions and Reversible Circuits", In Proceedings of the IEEE 38th International Symposium on Multiple Valued Logic (ISMVL 2008), 2008, pp. 220–225.