SCLCRL: Shuttling C-elements based Low-Cost and Robust Latch Design Protected against Triple Node Upsets in Harsh Radiation Environments

Aibin Yan[1], Zhixing Li[1], Shiwei Huang[1], Zijie Zhai[1], Xiangyu Cheng[1], Jie Cui[1], Tianming Ni[2], Xiaoqing Wen[3] and Patrick Girard[4]

[1]*School of Computer Science and Technology, Anhui University, Hefei, China*
[2]*College of Electrical Engineering, Anhui Polytechnic University, Wuhu, China*
[3]*Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology, Fukuoka, Japan*
[4]*LIRMM, University of Montpellier / CNRS, Montpellier, France*

***Abstract*—As the CMOS technology is continuously scaling down, nano-scale integrated circuits are becoming susceptible to harsh-radiation induced soft errors, such as double-node upsets (DNUs) and triple-node upsets (TNUs). This paper presents a shuttle C-elements based low-cost and robust latch (namely SCLCRL) that can recover from any TNU in harsh radiation environments. The latch comprises seven primary storage nodes and seven secondary storage nodes. Each pair of primary nodes feeds a secondary node through one C-element (CE) and each pair of secondary nodes feeds a primary node through another CE, forming redundant feedback loops to robustly retain values. Simulation results validate all key TNUs' recoverability features of the proposed latch. Simulation results also demonstrate that the proposed SCLCRL latch can approximately save 29% silicon area and 47% D-Q delay on average at the cost of moderate power, compared with the state-of-the-art TNU-recoverable reference latches of the same-type.**

I. Introduction

With the ceaseless reduction of CMOS transistor feature sizes, the soft error issue in nano-scale integrated circuits has become one of the most serious reliability challenges in harsh (e.g., space) environments [1]. As a type of transient errors, soft errors are mainly caused by the strike of radiative particles, such as protons, neutrons, heavy ions, electrons, muons, and alpha particles [2, 3]. Soft errors can lead to severe reliability problems, causing potential malfunctions of circuits and systems in the worst case. To mitigate soft errors, the recently developed FinFET technology is a feasible approach [3]. However, effective and scalable solutions based on other approaches for soft error tolerance, e.g., *radiation-hardening-by-design* (*RHBD*), are still needed.

Single-node upsets (*SNUs*), *double-node upsets* (*DNUs*) and *triple-node upsets* (*TNUs*) are typical soft errors. In a storage module, the value change of a single node caused by the strike of a radiative particle is commonly known as an SNU. In advanced technologies, due to the shrinking of transistor sizes, the space between nodes is becoming smaller. Thus, due to the charge sharing mechanism [4], the strike of a radiative particle is likely to cause logic values of multiple nodes to change, which is known as a *multiple-node upset* (*MNU*). Common MNUs include DNUs and TNUs. Especially, with advanced nano-scale technology nodes, the phenomenon of MNUs has become more prominent and is considered as a severe threat to circuit reliability [4]. Therefore, to ensure high circuit-reliability for safety-critical applications, especially in harsh radiation environments, *integrated circuit* (*IC*) designers and technologists need to consider radiation hardening against MNUs (especially TNUs).

In the last decade, researchers have proposed many radiation hardened circuits, such as flip-flops [5-6], random-access-memory cells [7-9], and latches [10-20], to tolerate SNUs, DNUs and even TNUs. This paper mainly considers hardness for latch designs. Typical SNUs, DNUs and/or TNUs hardened latch designs include the SHC [10], LSEHv1 [11], RFC [12], LSEDUT [13], DeltaDICE [14], HRDNUT [15], HREETNU [16], TNUHL [17], TNURL [18] and LCTNURL [4]. However, these latch designs still suffer from severe limitations as described below. First, most of existing latch designs cannot self-recover from DNUs and TNUs. Among them, some latch designs, such as the SHC and LSEHv1, even cannot self-recover from SNUs. There are also some latch designs that can self-recover from SNUs, but they cannot tolerate DNUs, e.g., the RFC. The LSEDNT can tolerate any DNU, but cannot tolerate TNUs. Moreover, some latch designs, such as the HREETNU and TNUHL, can tolerate any TNU but cannot self-recover from DNUs. Although the TNURL and LCTNURL latches can self-recover from DNUs and TNUs, the latch suffers from large overhead especially in terms of silicon area and delay.

Yan et al's work in [19, 20] cannot provide complete TNU self-recoverability. Although their previous work in [4, 18] can provide complete TNU-recoverability, the area and delay penalty is quite high. In this paper, a Shuttle C-elements based Low-Cost and highly Robust Latch, namely SCLCRL, is proposed. The latch mainly comprises two groups (i.e., the left and the right groups) of storage nodes. Two outputs in each group of two *C-elements* (*CEs*) feed two inputs of another CE so that redundant feedback loops can be formed to robustly retain values. In other words, the interlocking mechanism of CEs enables the latch to recover from any TNU. Simulation results demonstrate the TNU-recoverability and low overhead of the proposed latch.

The rest of the paper is organized as follows. Section II describes the schematics, working principles and validations of the proposed latch design. Section III presents comparisons with the state-of-the-art hardened latch designs. Section IV concludes the paper.

II. Proposed Robust Latch Design

A. Working Principles

Fig. 1 shows the proposed so-called SCLCRL latch design. It can be seen that the storage part of the SCLCRL latch mainly comprises seven primary CEs (CE1 to CE7) and seven secondary CEs (CE8 to CE14) so that the latch has seven primary storage nodes (N1 to N7) and seven secondary storage nodes (N8 to N14). Clearly, the CEs are seemingly shuttling between the primary and the secondary nodes so that the latch is said to be based on shuttling CEs. In the latch, a pair of primary nodes feed the inputs of a CE, which outputs a value to a secondary node and a pair of secondary nodes feed the inputs of a CE, which outputs a value to a primary node, so that redundant feedback loops can be constructed to robustly retain values. The latch also comprises four *transmission gates* (*TGs*) as shown at the bottom of Fig. 1. D is the input of the latch, N4(Q) is the output of the latch, and CLK and NCK are the system clock and negative system clock signals, respectively. Note that CE14 is controllable by CLK/NCK signals to reduce current competition on Q and hence reduce D-Q delay. Fig. 2 shows the layout of the proposed SCLCRL latch. In the following, we consider D = N4(Q) = 1 as example to describe the working principle of the latch.

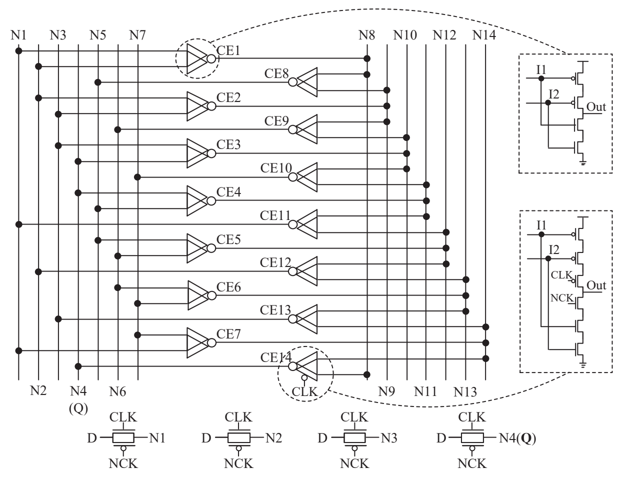

Fig. 1. Circuit structure of the proposed SCLCRL latch design.

When CLK is high and NCK is low, the latch works in transparent mode, and all TGs are ON. At this time, nodes D = N1 = N2 = N3 = N4(Q) = 1 so that CE1, CE2 and CE3 can output the reversed value of D to N8, N9 and N10, respectively. Thus, N8 = N9 = N10 = 0. Although N1 and N4 feed one input of CE7 and CE4, respectively, CE7 and CE4 cannot output a value during initialization. However, the value of N8, N9 and N10 can continue to propagate to N5 and N6 through CE8 and CE9, respectively. Thus, N5 = N6 = 1 so that CE5 can output 0 at N12. Note that N4 = N5 = 1 so that CE4 can output 0 at N11. Then, CE10 can output a reversed value of N10 and N11 at N7. Therefore, all primary nodes are initialized (i.e., the value of N1 to N7 is 1) so that the secondary nodes can be initialized (i.e., the value of N8 to N14 is 0) through CE1 to CE7. In other words, the proposed SCLCRL latch can be initialized correctly and can output the initialized value to the output of the latch.

When CLK is low and NCK is high, the latch switches into hold mode and all TGs are OFF; hence, nodes N1, N2, N3 and N4(Q) cannot be driven by D through TGs. At this time, the value of N1, N2, N3 and N4 can be driven by the initialized inputs of CE11, CE12, CE13 and CE14, respectively. It can be seen from Fig. 1 that any two adjacent primary nodes feed a secondary node through a primary CE and any two adjacent secondary nodes feed a primary node through a secondary CE (Note that we can consider N1 to N7 as nodes of a singly linked circular list, which means that N7 and N1 are also adjacent so does that for N14 and N8). In this manner, the primary and the secondary nodes are feeding back to each other through CEs so that redundant feedback loops can be constructed to retain values for the latch. Therefore, the proposed SCLCRL latch can correctly store values and can output the stored values to the output of the latch in hold mode.

Let us now discuss the TNU-recoverability of the proposed SCLCRL latch in hold mode. Due to the symmetric structure of the latch, we only need to consider two cases. ***Case T1***: Three primary nodes are affected by a TNU (this is similar to the case where three secondary nodes are affected by a TNU - we only take primary nodes as example to introduce TNU recoverability). ***Case T2***: Two primary nodes and one secondary node are affected by a TNU (this is similar to the case where one primary node and two secondary nodes are affected by a TNU - we only take two primary nodes and one secondary node as example to introduce TNU recoverability). Before analyzing the TNU-recoverability of the latch, four important properties of a 2-input CE are introduced as follows.

Property 1 (Recovery): If all inputs of a CE are correct, no matter whether its output is impacted or not, then its output will provide the correct value.

Property 2 (Valid-retention): If one input of a CE is impacted but its output is not impacted, then it will provide the correct output value, i.e., the error is masked.

Property 3 (Corruption): If all inputs of a CE are affected, it will provide a flipped output value. At this time, the inputs need recovery.

Property 4 (Invalid-retention): If at least one input along with the output are simultaneously impacted, the output will keep the flipped value. At this time, the inputs need recovery.

In ***Case T1***, there are four sub-cases. ***<u>Case T1-1</u>***: Three adjacent primary nodes are affected by a TNU. Let us denote as λ the distance between two adjacent nodes. The maximum distance between any two primary nodes is 3λ only. This is because, as mentioned above, we can consider N1 to N7 as nodes of a singly linked circular list, which means that N7 and N1 are also adjacent. Clearly, the key node list is <N1,N2,N3> only, and this sub-case belongs to the $<\lambda,\lambda>$ type because N2 is an adjacent node of N1(N3). Note that, the node lists, such as <N6,N7,N1> and <N7,N1,N2>, are equivalent to <N1,N2,N3>.

The TNU-recoverability for <N1,N2,N3> is discussed. When this node list is affected by a TNU, all inputs of CE1 and CE2 are affected so that the error can propagate to the output N8 and N9 of CE1 and CE2, respectively (Property 3). Note that the error at N1 cannot propagate to the output of CE7 and the error at N3 cannot propagate to the output of CE3 because only single inputs of the CEs are affected (Property 2). In other words, CE7 and CE3 can mask the error at their single inputs. As mentioned above, the error propagates to N8 and N9 through CE1 and CE2 so that the error can propagate to N5 through CE8 (Property 3). Note that the error at N8 cannot propagate to the output of CE14 and the error at N9 cannot propagate to the output of CE9 because only single inputs of the CEs are affected (Property 2). In other words, CE14 and CE9 can mask the error at their single inputs. As mentioned above, the error propagates to N5. However, the error at N5 cannot propagate to the output of CE4 and CE5 because only single inputs of the CEs are affected (Property 2). In other words, CE4 and CE5 can mask the error at their single inputs. Hence, the error at N1, N2 and N3 can be blocked by CEs. This means that nodes N11, N12, N13 and N14 are still correct so that N1, N2 and N3 can self-recover by these nodes through CE11, CE12 and CE13, respectively (Property 1). Hence, the error at N8 and N9 can be removed through CE1 and CE2, and the error at N5 can be removed through CE8 (Property 1). Therefore, node list <N1,N2,N3> is self-recoverable from a TNU.

Case T1-2: Two adjacent primary nodes and a close but not adjacent primary node are affected by a TNU. The key node list is <N1,N2,N4> only and this sub-case belongs to the <λ,2λ> or <2λ,λ> type. Clearly, node lists, such as <N4,N6,N7>, <N7,N1,N3> and <N6,N1,N2>, are equivalent to <N1,N2,N4>.

Let us discuss the TNU-recoverability for <N1,N2,N4>. When this node list is affected by a TNU, all inputs of CE1 only are affected so that the error can propagate to the output of CE1 (i.e., N8). Note that the error at N1 cannot propagate to the output of CE7, the error at N2 cannot propagate to the output of CE2 and the error at N4 cannot propagate to the output of CE3 and CE4 because only single inputs of the CEs are affected. As mentioned above, the error propagates to N8 through CE1. The error at N8 cannot propagate to the output of CE8 because only one input of CE8 is affected. However, the error at N8 can impact CE14 because input N8 and output N4 of CE14 have errors (Property 4). Hence, N8 needs recovery so that N1 and N2 need recovery to correct N8 through CE1. It can be seen from the above discussions that the errors cannot propagate to the inputs of other CEs, such as CE11 and CE12, so that N1 (N2) can recover through CE11 (CE12). Then, N8 can recover through CE1. Hence, N4 can recover through CE14. Therefore, node list <N1,N2,N4> is self-recoverable from a TNU.

Case T1-3: Two adjacent primary nodes and another far primary node are affected by a TNU. The key node list is <N1,N2,N5> only and this sub-case belongs to the <λ,3λ> or <3λ,λ> type. Note that, the node lists, such as <N3,N6,N7>, <N7,N1,N4> and <N1,N4,N5>, are equivalent to <N1,N2,N5>.

Let us discuss the TNU-recoverability for <N1,N2,N5>. When this node list is affected by a TNU, all inputs of CE1 only are affected so that the error can propagate to the output of CE1. Note that the error at N1 cannot propagate to the output of CE7, the error at N2 cannot propagate to the output of CE2 and the error at N5 cannot propagate to the output of CE4 and CE5. As mentioned above, the error propagates to N8 through CE1. At this time, the error at N8 can impact CE8 because input N8 and output N5 of CE8 have errors (Property 4). Hence, N8 needs recovery so that N1 and N2 need recovery to correct N8. It can be seen from the above discussions that the errors cannot propagate to the inputs of other CEs, such as CE11 and CE12, so that N1 (N2) can recover through CE11 (CE12). Then, N8 can recover. Hence, N5 can recover through CE8. Therefore, node list <N1,N2,N5> is self-recoverable from a TNU.

Case T1-4: Three non-adjacent primary nodes are affected by a TNU. The key node list is <N1,N3,N5> only and this sub-case belongs to the <2λ,2λ> type. Note that the <2λ,2λ> type is the same as the <2λ,3λ> type. For example, <N1,N3,N5> belongs to the <2λ,2λ> type, but <N5,N1,N3> belongs to the <3λ,2λ> type. Since <N1,N3,N5> equals to <N5,N1,N3>, the <2λ,2λ> type is the same as the <2λ,3λ> type.

Let us discuss the TNU-recoverability for <N1,N3,N5>. When this node list is affected by a TNU, only single inputs of primary CEs are affected. Note that the error at N1 cannot propagate to the output of CE1 and CE7, the error at N3 cannot propagate to the output of CE2 and CE3, and the error at N5 cannot propagate to the output of CE4 and CE5. Clearly, the errors cannot propagate to the inputs of other CEs, such as CE11, CE13 and CE8, so that N1, N3 and N5 can self-recover through CE11, CE13 and CE12, respectively. Therefore, node list <N1,N3,N5> is self-recoverable from a TNU. From the above discussions we can see that the proposed SCLCRL latch can self-recover from any possible TNU in ***Case T1***.

In ***Case T2***, there are three sub-cases. ***Case T2-1***: Two adjacent primary nodes (the node distance is λ) and one secondary node are affected by a TNU. Note that, if <N1,N2> is affected, N8 will be immediately affected, because CE1 can output the wrong value of <N1,N2> to N8. Thus, <N1,N2,N8> suffering from a TNU is similar to <N1,N2> suffering from a DNU. In ***Case T1***, N1, N2 and another primary node can be impacted by a TNU but the latch can self-recover from the TNU so that the latch can also self-recover from a DNU at <N1,N2>. Therefore, <N1,N2,N8> is self-recoverable from a TNU. Note that if ***Case T1*** can pass simulations, this sub-case (T2-1) can also pass simulations so <N1,N2,N8> will not be selected as key node list for simulations in the next sub-section.

Let us discuss the TNU-recoverability for <N1,N2,N9>. When this node list is affected by a TNU, all inputs of CE1 are affected so that the error can propagate to the output of CE1. Note that the error at N1 cannot propagate to the output of CE7 and the error at N9 cannot propagate to the output of CE9. As mentioned above, the error propagates to N8 through CE1. At this time, the error at N9 can impact CE8 because all inputs of CE8 have errors so that the error propagates to N5 through CE8. Note that the error at N5 cannot propagate to the output of CE4 and CE5. Meanwhile, the error at N2 and N9 can impact CE2 so that N2 needs recovery. Clearly, the errors cannot propagate to the inputs of other CEs, such as CE11 and CE12, so that N1 and N2 can recover through CE11 and CE12, respectively. Then, N8 can recover through CE1. Hence, N9 can recover through CE2 and then N5 can recover through CE8. Therefore, node list <N1,N2,N9> is self-recoverable from a TNU. Note that <N1,N2,N9> and <N1,N2,N14> are of the same type. This is because, the errors at N1 and N2 can propagate to single inputs of CE8 and CE14 through CE1, and meanwhile the errors at N9 and N14 can impact the other single inputs of CE8 and CE14, respectively. Therefore, node list <N1,N2,N14> is also self-recoverable from a TNU.

Let us discuss the TNU-recoverability for <N1,N2,N10>. When this node list is affected by a TNU, all inputs of CE1 are affected so that the error can propagate to the output of CE1. Note that the error at N1 cannot propagate to the output of CE7 and the error at N2 cannot propagate to the output of CE2. As mentioned above, the error propagates to N8 through CE1. Note that the error at N8 cannot propagate to the output of CE8 and CE14. Clearly, the errors cannot propagate to the inputs of other CEs, such as CE11, CE12 and CE3, so that N1, N2 and

N10 can recover through CE11, CE12 and CE3, respectively. Hence, N8 can recover through CE1. Therefore, node list <N1,N2,N10> is self-recoverable from a TNU.

Let us discuss the TNU-recoverability for <N1,N2,N11>. When this node list is affected by a TNU, all inputs of CE1 are affected so that the error can propagate to the output of CE1. Note that the error at N1 cannot propagate to the output of CE7 and the error at N2 cannot propagate to the output of CE2. As mentioned above, the error propagates to N8 through CE1. Note that the error at N8 cannot propagate to the output of CE8 and CE14. However, the error at N1 and N11 can impact CE11 (Property 4). Hence, N11 needs recovery. Clearly, the errors cannot propagate to the inputs of other CEs, such as CE4 and CE12, so that N11 (N2) can recover through CE4 (CE12). Hence, N1 can recover through CE11. Then, N8 can recover through CE1. Therefore, node list <N1,N2,N11> is self-recoverable from a TNU. Note that <N1,N2,N11> and <N1,N2,N13> are of the same type because the errors at N11 and N1 can impact CE11 and the errors at N13 and N2 can impact CE12, respectively (Property 4). Therefore, node list <N1,N2,N13> is also self-recoverable from a TNU. Moreover, <N1,N2,N11> and <N1,N2,N12> are also the same type because the errors at N12 and N1 can impact CE11 and the errors at N12 and N2 can impact CE12, respectively (Property 4) but N12 can recover through CE5. Therefore, node list <N1,N2,N12> is also self-recoverable from a TNU.

From the above discussions we can see that the key node lists are <N1,N2,N9>, <N1,N2,N10> and <N1,N2,N11> for this sub-case. In the next sub-section, TNUs will be injected into all key node lists to verify the TNU-recoverability for the proposed latch.

Case T2-2: Two near but non-adjacent primary nodes (the node distance is 2λ) and another secondary node are affected by a TNU. The key node list is <N1,N3,N8> only. Note that <N1,N3,N9>, <N1,N3,N10>, <N1,N3,N11>, <N1,N3,N12>, <N1,N3,N13> and <N1,N3,N14> are of the similar type compared to <N1,N3,N8>, because each of them suffering from a TNU can affect one input and the output of CE1, CE2, CE3, CE11, CE11, CE13 and CE13/CE7, respectively.

Let us discuss the TNU-recoverability for <N1,N3,N8>. When this node list is affected by a TNU, the error at N1 cannot propagate to the output of CE7, the error at N3 cannot propagate to the output of CE2 and CE3 and the error at N8 cannot propagate to the output of CE8 and CE14. However, the error at N1 and N8 can impact CE1. Hence, N1 needs recovery. Clearly, the errors cannot propagate to the inputs of other CEs, such as CE11 and CE13, so that N1 (N3) can recover through CE11 (CE13). Then, N8 can recover through CE1. Therefore, node list <N1,N3,N8> is self-recoverable from a TNU. Note that, for <N1,N3,N14>, although a TNU can impact one input and the output of both CE13 and CE7, N1 can recover through CE11 and then N14 can recover through CE7 so that N3 can recover through CE13.

Case T2-3: Two far primary nodes (the node distance is 3λ) and another secondary node are affected by a TNU. The key node list is <N1,N4,N8> and <N1,N4,N9>.

Let us discuss the TNU-recoverability for <N1,N4,N8>. When this node list is affected by a TNU, the error at N1 cannot propagate to the output of CE7, the error at N4 cannot propagate to the output of CE3 and CE4 and the error at N8 cannot propagate to the output of CE8. However, the error at N1 and N8 can impact CE1 and the error at N4 and N8 can impact CE14. Hence, N1 needs recovery for CE1 and N8 needs recovery for CE14. Clearly, the errors cannot propagate to the inputs of other CEs, such as CE11, so that N1 can recover through CE11. Then, N8 can recover through CE1. Thus, N4 can recover through CE14. Therefore, node list <N1,N4,N8> is self-recoverable from a TNU. Note that, <N1,N4,N8>, <N1,N4,N10>, <N1,N4,N11>,<N1,N4,N12> and <N1,N4,N14> are of the same type because one or two CEs suffer from errors at both its input and output and the nodes needing recovery can still recover.

Let us discuss the TNU-recoverability for <N1,N4,N9>. When this node list is affected by a TNU, the error at N1 cannot propagate to the output of CE1 and CE7, the error at N4 cannot propagate to the output of CE3 and CE4 and the error at N9 cannot propagate to the output of CE8 and CE9. Clearly, the errors cannot propagate to the inputs of other CEs, such as CE11, CE14 and CE2, so that N1, N4 and N9 can recover through CE11, CE14 and CE2, respectively. Therefore, node list <N1,N4,N9> is self-recoverable from a TNU. Note that, <N1,N4,N9> and <N1,N4,N13> are of the same type because the errors cannot propagate through CEs and all nodes can still recover. From the above discussions we can see that the proposed SCLCRL latch can self-recover from any possible TNU in ***Case T2***. In summary, the proposed SCLCRL latch can provide complete TNU-recoverability from all key TNUs so that the proposed latch can also provide complete DNU/SNU-recoverability from all key DNUs/SNUs.

B. Simulations

The SCLCRL latch was implemented in a 22 nm CMOS technology from GlobalFoundries and extensive simulations using HSPICE from Synopsys were performed. The simulation parameters were as follows: the supply voltage was set to 0.8V, the working temperature was set to the room temperature, the PMOS transistors of CEs had the ratio W/L = 32/22nm, and the NMOS transistors had the ratio W/L = 28/22nm.

Fig. 3 shows the error-free simulation results for the proposed latch to show correct operations. It can be seen that Q can correctly change along with D when the latch works in transparent mode (CLK = 1) and can correctly store sampled D when the latch works in hold mode (CLK = 0).

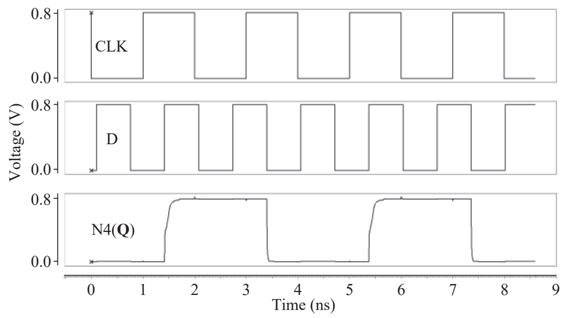

Fig. 3. Error-free simulation results for the proposed SCLCRL latch design.

Fig. 4 shows the simulation results for the complete key TNU injections of the proposed SCLCRL latch. Note that, the lighting marks in Fig. 4 denote the injected TNUs (we use three simultaneous SNUs to mimic a TNU). It can be seen that the

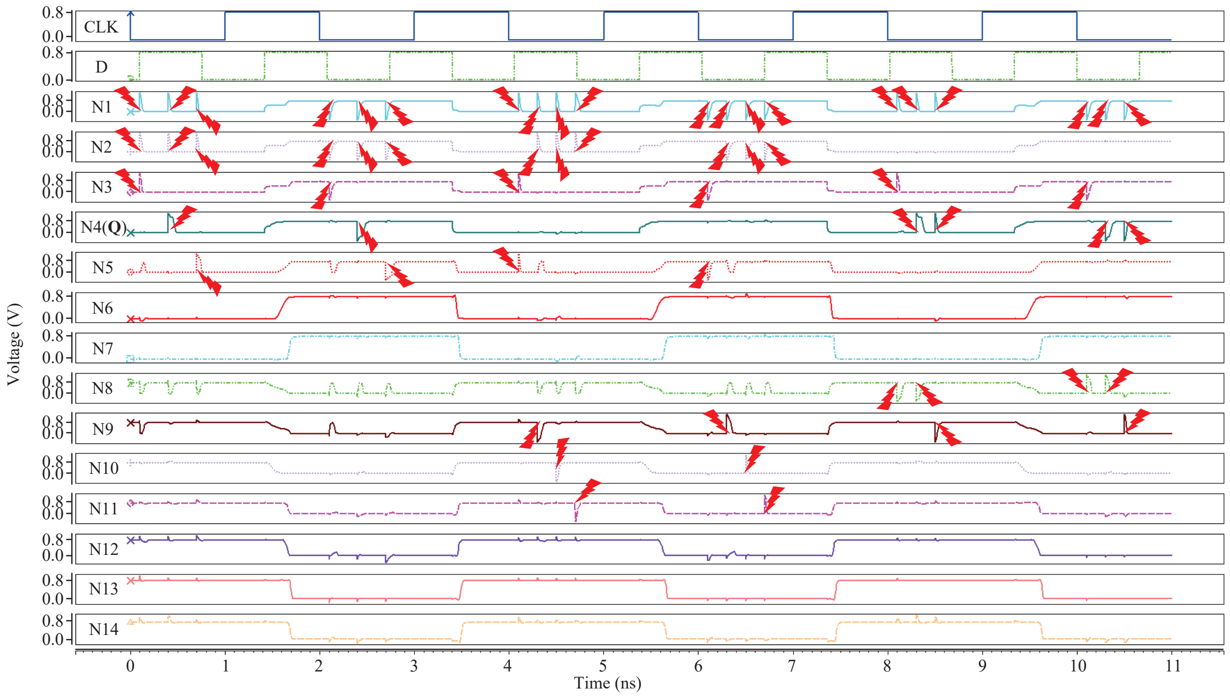

Fig. 4. Simulation results for the complete key TNU injections of the proposed SCLCRL latch design.

SCLCRL latch can self-recover from any key TNU (only causing narrow pulses). In summary, simulation results validate the fact that the SCLCRL latch can provide complete TNU recoverability (and thus SNU/DNU recoverability).

III. Comparison and Evaluation Results

To make a fair comparison, typical latch designs and the proposed SCLCRL latch were implemented with the same parameters listed in the previous sub-section (22 nm CMOS technology from GlobalFoundries, 0.8V supply voltage and room temperature).

Table I shows the reliability comparisons among the SNU, DNU, and/or TNU hardened latch designs. Note that “resilient” means “self-recoverable”. It can be seen from Table I that, the SHC and LSEHv1 latches are only SNU-tolerant, but the RFC latch can additionally provide SNU recoverability. The LSEDUT latch is not only SNU-resilient but also DNU-tolerant, but the DeltaDICE and HRDNUT latches can additionally provide DNU recoverability. The HREETNU and TNUHL latches can provide SNU/DNU/TNU-tolerance and SNU recoverability, but the TNURL, LCTNURL and the proposed SCLCRL latches can additionally provide DNU and TNU recoverability. Therefore, the bottom three latches in Table I are more reliable (but the proposed SCLCRL latch has small silicon area and delay as it will be discussed below).

Table II shows the overhead comparisons among the SNU, DNU, and/or TNU hardened latch designs. In Table II, "Area" denotes the silicon area that is calculated based on extracted layouts, "Delay" denotes D to Q transmission delay (i.e., the average of rise and fall delays from D to Q), "Power" denotes the average power dissipation (dynamic and static) and the *area-delay-power product* (*ADPP*) means a comprehensive metric that is calculated by multiplying area, delay and power.

It can be seen from Table II that, compared with the SNU/DNU tolerant/resilient latches, the TNURL, LCTNURL as well as the proposed SCLCRL latch consume more silicon area, because redundant transistors are used to provide SNU/DNU/TNU tolerance/recoverability. However, compared with the TNURL and LCTNURL latches, the proposed SCLCRL latch consumes less silicon area, delay, and ADPP.

In terms of delay, for the LSEHv1 and TNUHL latches, their delay is large due to the use of many logic devices from D to Q. In contrast, the RFC, LSEDUT, HRDNUT, HREETNU as well as the proposed SCLCRL latch have a small delay since they use high-speed transmission paths from D to Q. However, the other latches have a moderate delay. Note that, although the TNURL and LCTNURL latches also use a high-speed path from D to Q, there is current competition on Q, and thus their delay is not small.

In terms of power, among the latches in Table II, the LSEDUT latch consumes the largest power mainly because the feedback loop at the output is always active. Due to the current competition between nodes, the HRDNUT latch also consumes more power. The power dissipation of the LSEHv1, DeltaDICE, TNUHL, TNURL, LCTNURL and SCLCRL latches is not small, mainly because of their large area. The SHC latch consumes less power mainly due to the small silicon area. The RFC and HREETNU latches consume less power due to the small silicon area and the use of clock-gating technologies.

In terms of ADPP, the LSEHv1 and TNUHL latches have very high ADPP values, mainly because of their large delay. Conversely, the SHC, RFC, and HREETNU latches have low ADPP values since their area, delay, and/or power are small. The other latches have moderate ADPP values. However, compared to the TNU-tolerant latches, the ADPP of the proposed SCLCRL latch is still low.

TABLE I
Reliability Comparisons among the SNU/DNU/TNU Hardened Latch Designs

Latch	SNU Tolerant	SNU Resilient	DNU Tolerant	DNU Resilient	TNU Tolerant	TNU Resilient
SHC	√	×	×	×	×	×
LSEHv1	√	×	×	×	×	×
RFC	√	√	×	×	×	×
LSEDUT	√	√	√	×	×	×
DeltaDICE	√	√	√	√	×	×
HRDNUT	√	√	√	√	×	×
HREETNU	√	√	√	×	√	×
TNUHL	√	√	√	×	√	√
TNURL	√	√	√	√	√	√
LCTNURL	√	√	√	√	√	√
SCLCRL	√	√	√	√	√	√

TABLE II
Overhead Comparisons among the SNU/DNU/TNU Hardened Latch Designs

Latch	Area (μm^2)	Delay (ps)	Power (μW)	ADPP
SHC [10]	1.69	17.79	0.05	1.50
LSEHv1 [11]	4.32	89.66	0.44	170.43
RFC [12]	3.79	3.00	0.13	1.48
LSEDUT [13]	6.38	1.67	1.69	18.01
DeltaDICE [14]	5.77	16.70	0.42	40.47
HRDNUT [15]	6.12	3.46	1.19	25.20
HREETNU [16]	5.59	1.67	0.20	1.87
TNUHL [17]	10.28	101.86	0.63	659.69
TNURL [18]	13.52	5.20	0.41	28.82
LCTNURL [4]	10.97	5.77	0.40	25.32
SCLCRL	8.57	2.91	0.48	11.97

Let us discuss the quantitative overhead comparison results. Compared with the TNU-recoverable TNURL latch, the SCLCRL latch can save about 36.61% silicon area, 44.04% delay and 58.47% ADPP but at the cost of 17.07% additional power. Compared with the TNU-recoverable LCTNURL latch, the SCLCRL latch can save about 21.88% silicon area, 49.57% delay and 52.73% ADPP but at the cost of 20.00% power. Therefore, compared with the TNU-recoverable latches, the proposed SCLCRL latch can save about 29.25% silicon area, 46.80% delay and 55.60% ADPP but at the cost of 18.54% power on average. In summary, the proposed SCLCRL latch can provide TNU recoverability (and thus DNU and SNU recoverability) with low cost in terms of area, delay and ADPP.

IV. Conclusions

CMOS technology scaling can lead to the occurrence of soft errors (e.g., TNUs) in harsh radiation environments. Existing latches suffer from severe limitations such as non-tolerance to TNUs, and/or large overhead. To address this issue, this paper has proposed a shuttling C-elements based low-cost and robust latch design recoverable from triple node upsets. Simulation results have demonstrated the TNU self-recoverability and low overhead especially in terms of silicon area, D-Q delay and ADPP for the proposed latch.

References

[1] Y. Li, J. Han, X. Zeng, et al., “TRIGON: A Single-phase-clocking Low Power Hardened Flip-Flop with Tolerance to Double-Node-Upset for Harsh Environments Applications,” in *Proc. Design, Automation & Test in Europe Conference & Exhibition (DATE),* pp. 88-93, 2021

[2] M. Ebara, K. Yamada, K. Kojima, et al., “Process Dependence of Soft Errors Induced by α Particles, Heavy Ions, and High Energy Neutrons on Flip Flops in FDSOI,” *IEEE Journal of the Electron Devices Society*, vol. 7, no. 1, pp. 817-824, 2019

[3] B. Narasimham, S. Gupta, D. Reed, et al., “Scaling Trends and Bias Dependence of the Soft Error Rate of 16 nm and 7 nm FinFET SRAMs,” *IEEE International Reliability Physics Symposium*, pp. 1-4, 2018

[4] A. Yan, Y. Hu, J. Cui, et al, "Information Assurance through Redundant Design: A Novel TNU Error-Resilient Latch for Harsh Radiation Environment," *IEEE Transactions on Computers*, vol. 69, no. 6, pp. 789-799, 2020

[5] K. Kobayashi, K. Kubota, M. Masuda, et al., “A Low-Power and Area-Efficient Radiation-Hard Redundant Flip-Flop, DICE ACFF, in a 65 nm Thin-BOX FD-SOI,” *IEEE Transactions on Nuclear Science*, vol. 61, no. 4, pp. 1881-1888, 2014

[6] Y. Li, H. Wang, R. Liu, et al., “A Quatro-Based 65 nm Flip-Flop Circuit for Soft-Error Resilience,” *IEEE Transactions on Nuclear Science*, vol. 64, no. 6, pp. 1554-1561, 2017

[7] L. Dang, J. Kim, and I. Chang, “We-Quatro: Radiation-Hardened SRAM Cell with Parametric Process Variation Tolerance,” *IEEE Transactions on Nuclear Science*, vol. 64, no. 9, pp. 2489–2496, 2017

[8] S. Pal, S. Bose, W. Ki, et al., “Characterization of Half-Select Free Write Assist 9T SRAM Cell,” *IEEE Transactions on Electron Devices*, vol. 66, no. 11, pp. 4745-4752, 2019

[9] R. Rajaei, B. Asgari, M. Tabandeh, et al., “Design of Robust SRAM Cells Against Single-Event Multiple Effects for Nanoscale Technologies,” *IEEE Transactions on Device and Materials Reliability*, vol. 15, no. 3, pp. 429-436, 2015

[10] S. Tajima, N. Togawa, M. Yanagisawa, et al., ”A Low Power Soft Error Hardened Latch with Schmitt-Trigger-Based C-Element,” *IEICE Transactions on Fundamentals*, vol. E101–A, no. 7, pp. 1025-1034, 2018

[11] R. Rajaei, M. Tabandeh, M. Fazeli, “Low cost soft error hardened latch designs for nano-scale CMOS technology in presence of process variation,” *Microelectronics Reliability*, vol. 53, pp. 912-924, 2013

[12] A. Yan, H. Liang, Z. Huang, et al., “A self-recoverable, frequency-aware and cost-effective robust latch design for nanoscale CMOS technology,” *IEICE Transactions no Electronics*, vol. E98, no. 12 pp. 1171-1178, 2015

[13] J. Jiang, Y. Xu, J. Ren, et al., “Low-cost single event double-upset tolerant latch design,” *Electronics letters*, vol. 54, no. 9, pp. 554-556, 2018

[14] N. Eftaxiopoulos, N. Axelos, G. Zervakis, K. Tsoumanis, et al., “Delta DICE: A double node upset resilient latch,” in *Proc. IEEE International Midwest Symposium on Circuits and Systems*, pp. 1-4, 2015

[15] A. Watkins and S. Tragouodas, “A Highly Robust Double Node Upset Tolerant Latch,” in *Proc. International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, pp. 15-20, 2016

[16] C. I. Kumarand and B. Anand, “A Highly Reliable and Energy Efficient Triple Node-Upset Tolerant Latch Design,” *IEEE Transactions on Nuclear Science*, vol. 66, no. 10, pp. 2196-2206, 2019

[17] A. Watkins and S. Tragoudas, “Radiation Hardened Latch Designs for Double and Triple Node Upsets,” *IEEE Transactions on Emerging Topics in Computing*, vol. 8, no. 3, pp. 616-626, 2020

[18] A. Yan, X. Feng, Y. Hu, et al., “Design of a Triple-Node-Upset Self-Recoverable Latch for Aerospace Applications in Harsh Radiation Environments,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 2, pp. 1163-1171, 2020

[19] A. Yan, C. Lai, Y. Zhang, et al., “Novel Low Cost, Double-and-Triple-Node-Upset-Tolerant Latch Designs for Nano-Scale CMOS,” *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 1, pp. 520-533, 2021

[20] A. Yan, Y. Ling, J. Cui, et al., “Quadruple Cross-Coupled Dual-Interlocked-Storage-Cells based Multiple-Node-Upset-Tolerant Latch Designs,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 3, pp. 879-890, 2020