2012 IEEE 18th International Symposium on Asynchronous Circuits and Systems

Adding Temporal Redundancy to Delay Insensitive Codes to Mitigate Single Event Effects

Julian Pontes and Ney Calazans
Faculty of Informatics - FACIN - PUCRS
Porto Alegre, RS, Brazil
{julian.pontes, ney.calazans}@pucrs.br

Pascal Vivet
CEA-Leti, MINATEC
Grenoble, France
pascal.vivet@cea.fr

Abstract— **In advanced CMOS technology, Single Event Effects due to high energy particle may cause different types of electrical effects when crossing silicon: from small delay variations, to bit flips, until permanent damage. Quasi Delay Insensitive asynchronous circuits are the most immune to delay variations thanks to the use of Delay Insensitive codes, but can be very sensitive to bit flips since a Single Event Effect may corrupt the handshake protocol. This paper presents a design technique to mitigate Single Event Effect by adding temporal redundancy to Delay Insensitive codes. This multiple bit fault tolerant design technique is adaptable to any 1-of-N DI code, and is particularly well suited to asynchronous Networks-on-Chip. The proposed Temporally Redundant Delay Insensitive codes have been evaluated using a Single Event Effect digital fault characterization environment. The result shows better SEE tolerance and reduced area and performance impact.**

Keywords-component; asynchronous circuits; quasi delay insensitive; network on chip; single event upset; soft errors; radiation hardening

I. INTRODUCTION

In advanced deep submicron technologies, the aggressive scaling of the clock and associated high frequencies has now terminated. At the circuit top level, global clocking is not feasible anymore, which has led to the popularization of the Globally Asynchronous Locally Synchronous (GALS) paradigm, with local islands of clocked logic interconnected by asynchronous communication. By providing packet based communication, asynchronous Network-on-Chips have recently shown their benefits compared to their synchronous counterparts in terms of performance and power [1][2][3], to build future many-core architectures. One of the next challenges for such asynchronous architectures is reliability to Single Event Effects in case of particle radiation, since technology scaling continuously increase logic sensitivity [4].

By getting rid of the clock, the asynchronous logic brings many benefits, such as low power, low noise, modularity, but most of all delay robustness. Thanks to the characteristics of the Delay Insensitive Codes, like dual-rail or m-of-n codes, the Quasi Delay Insensitive (QDI) asynchronous logic presents a high level of robustness to almost any source of delay variations such as Process, Voltage, Temperature or even Crosstalk [15]. Nevertheless, as any kind of logic, QDI logic is also sensitive to Single Event Effects (SEEs). An SEE is caused by a high energy particle that may cause different types of electrical effects when crossing the silicon: from small delay variations, to bit flips, until permanent damage [4]. SEE induced delay variations will have no impact on the QDI logic, but in case of bit flips within the logic, the handshake of the asynchronous pipeline may be corrupted. The asynchronous pipeline corruption can take various forms with corrupted data value but also with handshake protocol deadlock due to erroneous tokens or erroneous bubbles.

In order to increase the robustness of asynchronous pipelines it is necessary to apply fault tolerance mechanisms that go beyond the synchronous design techniques such as parity bits or glitch filtering. To recover from SEE within the asynchronous logic it is necessary to use a modified Delay Insensitive code capable to offer more robustness to the whole design.

This paper proposes to add temporal redundancy to classical Delay Insensitive codes, such as 1-of-n, to mitigate SEEs. The temporal redundancy principle consists in encoding the current data token with its previous value, by using a higher order DI code more robust to SEE. The proposed temporal redundancy and transcoding scheme fits well point to point connections for on-chip and off-chip communication as well as for asynchronous Network-on-Chip. It offers robustness at reduced power and area cost when compared to existing solutions using spatial redundancy. The proposed scheme has been fully validated and characterized on asynchronous pipeline links, using an accurate SEE digital fault characterization environment.

The paper outline is as follows. Section II presents some related works on fault tolerant asynchronous design. Section III analyzes the impact of SEEs in asynchronous pipelines. Section IV presents the principle of temporal redundancy while Section V presents the details of its implementation. Section VI presents the used characterization environment. The subject of Section VII is to show comparative evaluations of robustness, area, timing and power between the proposed technique and other asynchronous pipelines links using the SEE fault simulation environment. Finally, Section VIII concludes the paper and points directions for future work.

II. RELATED WORKS

Bastos et al. [5] evaluate the implementation of asynchronous and synchronous versions of a DES crypto-processor. The results show that the asynchronous version has a better response when increasing SEE pulse widths. Rahbaran and Steininger [6] make the same comparison for a 16-bit processor using an FPGA as a target for SEE emulation. The asynchronous implementation shows the best results under several different SEE scenarios. Even if they display better robustness, QDI circuits are not free from soft errors, and in order to enable their use in hostile environments or even for new technologies, it is necessary to implement additional techniques to ensure low Single Effect Rates (SERs).

Agyekum and Nowick [7] propose an unordered DI code that enables two-bit error detection and one-bit error correction capabilities. However, this code is difficult to implement in a fully QDI way and becomes complex for several parallel bits due to the completion detection complexity. Bainbridge and Salisbury [8] present a set of techniques to apply to QDI Networks on Chip links, particularly for links based on m-of-n encoding, to reduce glitch sensitivity. The presented techniques are mostly sampling filtering techniques to reduce glitches in data signals and in some of them in acknowledge signals. However, none of these is adequate for SEE and none offers data error correction. Note also that data error correction and/or detection is required since the techniques are not able to completely eliminate the consequences of glitches or SEEs in QDI links.

The work of Bastos et al. [9] shows a comparison between different implementations of C-elements operating under SEEs. They also present a hardening technique based on electrical filtering by resizing these components. Resizing basically consists in increasing the capacitance of the nodes to filter small size pulses. Since C-elements are the most important building blocks of QDI circuits and the main component of memory elements in these circuits, the increase of their robustness may bring enhanced SEE tolerance to the whole circuit. However, increasing the size of transistors expands the die area susceptible to particle strikes and the area of the junctions, responsible for charge collection.

Monnet et al. [10] show three different techniques to mitigate SEEs in asynchronous logic. The first consists in the duplication of the asynchronous logic and a check at memory elements. This technique is able to remove errors in case of single events. It clearly involves a big area overhead, since the whole circuit is duplicated. The second technique consists in synchronizing different DI bits of the computation part. For example, a less significant bit of an AND applied bit per bit to a 16-bit word will be synchronized with the second less significant bit. This is done successively to the other bits. The synchronization is performed by a four input C-element in such a way that the output of the cell is set just when the AND computation in both bits has finished. This technique does not ensure the elimination of errors but increases the logic filtering property of the cell. The last technique adds a single rail signal for the validity of the data in the forward direction between memory blocks that is used to validate the data at the input of each memory block. This technique, as the previous one, only increases the logic filtering property.

Jang and Martin [11] propose an SEE hardening technique using spatial redundancy for asynchronous circuits called *double check*. The technique consists in duplicating the logic and adding a verification point after the computation, to ensure that the results are equivalent in both branches. Double check is capable of providing multi-bit error correction, and is accomplishable with simple, C-element-based circuits.

The work presented in this paper proposes an encoding technique able to implement double check but exchanging *spatial redundancy* by *temporal redundancy*. As a consequence, it decreases area and power of the whole circuit while keeping performance and robustness. Lastly, none of the related works presented in this section presents quantitative measurements at circuit level to validate the hardening technique propositions. In this paper, a digital characterization flow for SEE is proposed and used to validate and extract quantitative measurements about the robustness of the proposed temporal redundancy scheme.

III. SEE Impact in QDI Pipelines

A. Introduction to QDI Pipeline

Asynchronous QDI data links are built through pipelines using two main components: asynchronous registers and completion detectors (CD). The simplest asynchronous register implementation is the weak conditioned half buffer (WCHB). Figure 1 shows one possible and frequent implementation of an asynchronous three stage pipeline for the one-of-four encoding. The pipelines presented and evaluated here in this work are using the 1-of-4 / 4-phases handshake protocol.

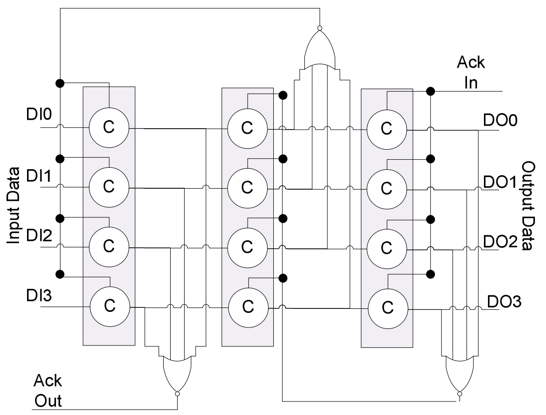

Figure 1 – 1-of-4 QDI pipeline using weak conditioned half buffer.

B. SEE impact on C-element

Asynchronous registers are implemented using the C-element for the forward logic (direct path), but the C-elements have also an important role in the completion detectors of the backward logic (acknowledge path). In order to analyze precisely the QDI pipelines behavior in case of SEE, it is firstly required to analyze the behavior of the C-element under radiation. This behavior is presented in Figure 2 by a state graph that describes the behavior of a simple 2 input C-element under the presence of radiation.

The C-element can present Single Event Transient (SET) or Single Effect Upset (SEU). The resulting effect depends on the state of the victim C-element when it is hit by a radiation. In the states 000 and 111, the C-element has a direct electrical path from the inputs to the output. In this case, a radiation hitting can generate a SET (Transient fault). In states 010, 100, 011 and 101, the C-element acts as a memory element. In these cases, the radiation causes a SEU (Upset or bit flip fault).

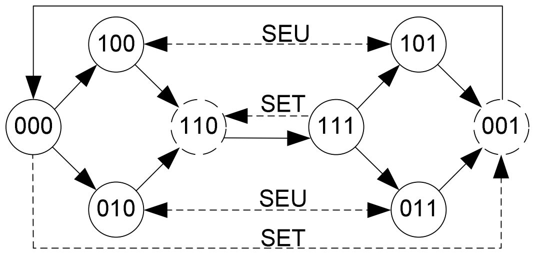

Figure 2 - State transition graph of a C-element under presence of SEE.

Table I provides the normalized SEE critical charge that is required in order to generate an SEE or SET on the C-element according to its state. This result has been extracted by precise spice level simulation as presented section VI. As expected, the SEE amplitude required to have the C-element flip is smaller when the C-element is holding its output compared to when the C-element is driving its output (000 and 111 state).

Table 1 : SEE critical charge to generate a fault according to C-element state when driving a charge of 8.1fF.

	C-element States					
	000	010	011	100	101	111
Normalized SEE Critical Charge	0.72	0.088	0.12	0.097	0.1	1

C. *SEE impact on 1-of-n pipeline*

Data links implemented with 1-of-n encoding are always in an excited state. This means that when the asynchronous register holds a spacer, an SEE↑ is able to generate a new data. When the register holds a data, a SEE↓ is able to clear the data token and generate a bubble. Figure 3 shows the SEE and timing behavior of a 4-phase protocol for the 1-of-n weak conditioned half buffer. The *Data Delay* is the delay between the Ack Out Rising Transition (see Figure 1) and the generation of a new data transition. The *Ack Delay* is the delay between data propagation through next stage and consecutive detection. The *Spacer Delay* is similar to *Data Delay.* It starts with the Falling Transition of the Ack Out signal and stops when the previous state removes the data from the input. The possible SEE within each timing window is related to the C-element states in the timing window. In the *Data Delay* timing window, the C-elements of the asynchronous register are in the retention state, since one of its inputs is '0' (Input Data Signal) while the Ack Signal is '1'. In this state, all the cells in the register are susceptible to a SEU↑ (considering that all the outputs of the register are '0'), that can generate a *Valid Corrupted Data* (VCD). A VCD is a data token generated by the SEE that has a valid 1-of-n encoding. The VCD can be detected and propagated from side to side of the pipeline. Depending on the sequence of the events, this new token can be:

1. An early data indication: If the real data arrives in the same step of the 4-phase protocol (before the propagation delay of the next register stage delay plus the next stage completion detection delay) and the data wire is the same victim wire.
2. Incomplete Corrupted Data (ICD): real data arrives in the same step of the 4-phase protocol but in a different wire. The occurrence of an ICD means that the bit that carries the data is still present but one new bit was flipped resulting in an invalid 2-of-n symbol. This erroneous data can be easily detected by the receiver but it is not possible to determine which wire carries the correct data.
3. If the data arrives after the next register stage delay plus the next stage completion detector delay then the register will be closed and the register will preserve the VCD.

A SEE that occurs inside the *Ack Delay* timing window has the possible effects: an Invalid *Corrupted Data* (ICD) in the case of a SEU↑ and an *Unexpected Spacer* (US) in the case of a SET↓. The US means that the data disappears generating a spacer before the currently 4-phase protocol finishes. As presented Table 1, this occurs in the state where the C-element presents the highest critical charge value. Besides this, this is a transient event. This means that after some amount of time, corresponding to the amount of charge injected, the correct data will return in the pipeline stage. The *Unexpected Spacer* (US) and the *Unexpected Data* (UD) presented in the *Ack Delay* timing windows can change the sequence of the protocol and the result can be a stall in the pipeline.

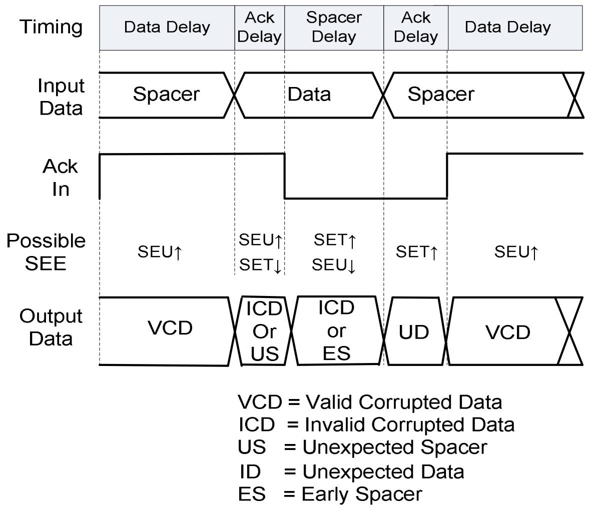

Figure 3 – SEE and timing diagram of the 1-of-n pipeline 4-phase protocol.

The completion detection of 1-of-n circuits is done by a NOR gate. In this way, the only SEE possible is a SET. In this work, the robustness in the Ack signals is not treated. However, one way to increase the robustness of the 1-of-n encoding is to add the detection based on several 1-of-n encodings by using a C-element tree [11]. In this case, the data becomes valid just after all the 1-of-n encodings involved on the detection become valid. This simple modification strongly changes the timing behavior of the pipeline. Figure 4 shows the completion detector of 4 dual-rail encodings based on a tree of C-elements. Applying this model of completion detection, the timing properties of the pipeline are closer to the 4-of-8 encoding than the dual-rail. Another impact is that the generation of a SEE in any cell of the completion detection apart of the last C-element in the tree will be just propagated to the output if the completion detection is already switching. This means that only early events can be generated. The only susceptible cell is the last C-element of the tree. This type of detection increases the robustness of individual 1-of-n but adds delay on the detection circuit and consequently increase the cycle time of the circuit.

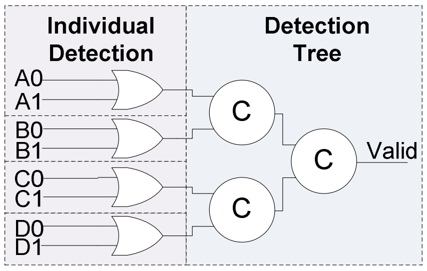

Figure 4 – Completion Detection of a dual rail encoding.

D. *SEE impact on m-of-n pipeline*

In this section, we use the term m-of-n encoding to refer to the subclass of this DI encodings where m > 1. The m-of-n codes data encoding are more robust to SEE since the data encoding are always at *m* bits distant from the spacer encoding. This means that a SEE is not able to create a data or clear a data from the data link if the data is not already switching. This property is obtained due to the completion detection circuit that needs at least *m* inputs to switch. Figure 5 shows the completion detector of an individual 2-of-3 encoding.

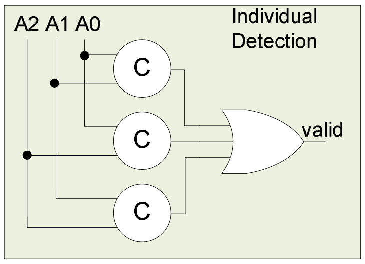

Figure 5 – Completion Detection for a 2-of-3 data encoding

Figure 6 shows the timing and SEE analysis of a 4-phase protocol for an m-of-n weak conditioned half buffer. For each phase of the protocol, the timing description is shown on the top of the figure. The *Best Case Data Delay* indicates the fastest path between the previous stage and the register input. The *Data Skew* is the timing difference between the fastest and the slowest bit in the data-path. The Acknowledge Delay (*Ack Delay*) is the timing needed from data propagation to the completion detection plus the detection delay. The SEEs that are observed in each window are the results of the C-element behavior.

In the *Best Case Delay* timing window, the only possible SEE is the occurrence of a SEU↑. This event will generate an *Incomplete Data* (ID). This is the main difference with the timing behavior of 1-of-n encoding (Figure 3) where a VCD is generated and a new token is erroneously created.

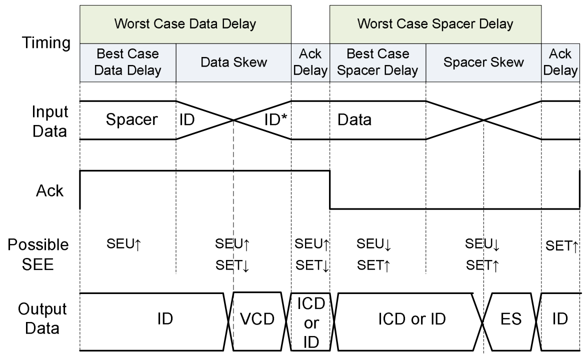

Figure 6 – Timing and SEE details of m-of-n four phase protocol.

The *Data Skew* timing window is subdivided in two new windows. These news windows are delimited by the Input Data State. If the *m-1* bits of the Input Data already switched (ID*), then the encoding reaches an excited state otherwise the inertial property of the code will keep filtering any event at the Input Data. The *Data Skew ID** timing window is the only window where a VCD can happen. Since the data is already switching during this window, it is expected that the remaining data wire switches before the Ack-In signal arrives (event that closes the register for the data propagation). If the data wire arrives before the acknowledge signal, the result will be an ICD instead of a VCD. In this way, the VCD opportunity can be removed at design time by guaranteeing that the data skew is smaller than the Ack In propagation. Data skews are usually rather small since all data signals are generated in the Rise Edge of the Acknowledge signal. The environment must always detect and sample the data before acknowledge it. Looking inside the pipeline, it is possible to note that the Acknowledge propagation will be the timing to cross the next stage register plus the completion detection time. This gives enough time for the missing data wire to be generated.

The timing property of m-of-n enlarges the ICD window inside the protocol. An ICD presents the correct data information plus an extra bit that turns the DI code invalid (m+1-of-n). The invalid code presented by a corrupted data is able to cross the pipeline stages and arrives to the final data receiver since it is detected by a regular completion detector implementation.

By comparing Figures 3 and 6, one can finally note that the unexpected data and spacer (UD and UE from Figure 3) are not presented in the m-of-n protocol. Even if m-of-n is presenting better filtering properties than 1-of-n, the ICD fault is still present. The main advantage is that this class of errors can be easily detected by the receiver, but no correction can be done at pipeline level.

IV. Temporal Redundancy in DI Codes

In order to increase the robustness of the QDI data links, we propose to translate from the initial 1-of-n DI encoding to a higher order m-of-n DI encoding. The m-of-n encoding has timing properties that enable a better SEE filtering. However, as shown in the previous section, the m-of-n encoding still suffers with the Invalid Corrupted Data sequence. In order to overcome this problem, a Temporally Redundant Delay Insensitive Code

(TRDIC) is introduced. The proposed solution can be applied to any 1-of-n encoding and is capable to:

1. Keep the filtering property and robustness of the n-of-m encoding
2. Correct multi-bit corrupted data (ICDs)
3. Keep the throughput close to the 1-of-n encoding by keeping the same number of tokens in the overall architecture

Figure 7 details the conversion process from 1-of-4 to 2-of-5 temporally redundant code for all possible encoding combinations. The Data[i-1] represents the previous data that has already been sent. The Data[i] is the new data that will be sent for the first time. During the reset phase, it is mandatory that both encoder and decoder use the same initial values as starting point in the previous register. To complete a transmission, it is required to send the last data plus a fake data to carry the second occurrence of the real last data. The TRDIC conversion always uses all possible 2-of-n+1 encodings. Code conversion can be implemented by simply a logic OR function (Data[i] OR Data[i-1]) between successive data tokens – where *i* denotes the actual data and *i-1* the previous one. The additional most significant bit is necessary to indicate when two identical tokens are sent in sequence. This last bit can be easily generated by a sum of products and implemented using DIMS logic.

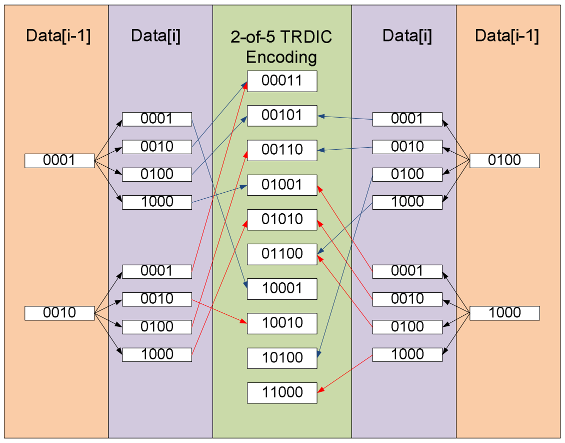

Figure 7 – Conversion from 1-of-4 to 2-of-5 redundant code.

V. Temporal Redundancy Implementation

Figure 8 shows the general structure of the TRDIC communication system. Communication starts using a regular 1-of-n encoding. This DI code is converted by the TRDIC encoder in a temporally redundant 2-of-n+1 encoding. The data is sent by a regular 2-of-n+1 QDI link to the receiver. Before reception of the data, it is decoded in the TRDIC Decoder and the necessary error corrections are done by a double check scheme [11]. In this way, the temporal redundancy can be used jointly with the spatial redundancy proposed by [11]: computation based designs are more adapted to spatial redundancy while communication based designs can be implemented with the proposed temporal redundancy mechanism.

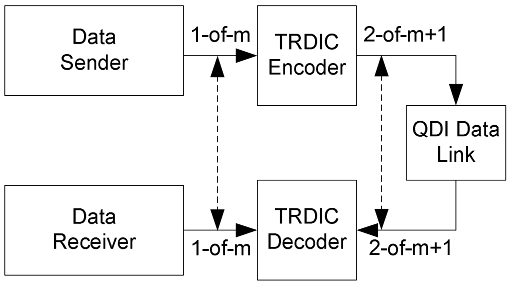

Figure 8 –Overview of the proposed temporal redundancy communication system.

Since the TRDIC proposal is well adapted to communication based designs, it is a good candidate to provide SEE robustness to QDI asynchronous Networks-on-Chip that use the 1-of-n encoding such as [1][2][3].

A. *Temporal Redundancy Encoder*

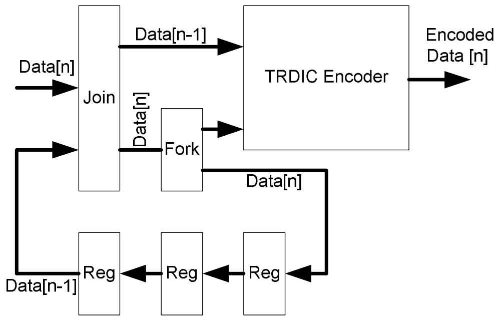

Figure 9 – TRDIC encoder.

Figure 9 shows the architecture of a TRDIC encoder. It consists of a circuit responsible to combine previous and current values of a data token, using the conversion scheme presented in Figure 7. It contains a feedback loop to keep track of the previous token value. At reset phase, the register that keeps the previous data is initialized with the same value used in the decoder while the other two registers in the loop keep a bubble with two consecutive spacers.

B. *Temporal Redundancy Decoder*

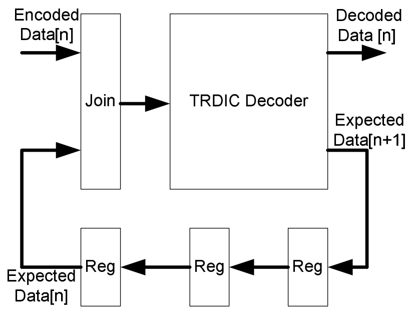

Figure 10 – TRDIC decoder.

Figure 10 shows the architecture of the decoder. Decoding is performed by C-elements. As for the encoder, a feedback loop is necessary to keep track of the expected next data. The last register of the loop that feeds the join components must be initialized with the same predefined value of the encoder, while the other two registers are holding a spacer bubble. After the reset phase, the first data decoded has no meaning and is discarded.

Figure 11 shows a temporal double check example for a 2-of-5 TRDIC. The circuit is implemented using a C-element between the redundant data with the expecting data extracted from the previous decoding step. Bit D4 is not used after the decoding process, since it is just used to represent the two consecutive values in TRDIC encoding.

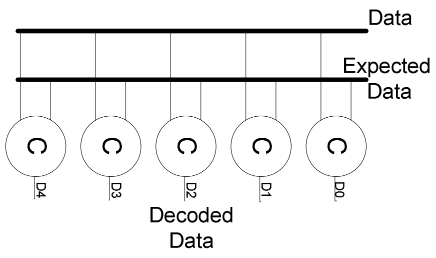

Figure 11 – Double Check between consecutive data.

The double check decoder works as a filtering element applied to two consecutive data. The TRDIC code can be viewed as a two stage trellis. To pass from one stage to another there is always four possible paths, one for each possible encoding using 2 bits. It is important to note that the double checker is not capable to filter errors that result in a VCD. However, the timing properties of the 2-of-n encoding pipeline almost eliminate the occurrence of VCDs as shown in Figure 6. Since the ICD data contains the data plus an invalid bit flip, the combination of the timing filtering of the 2-of-n encoding and the TRDIC based in a double checker is able to filter most possible erroneous data.

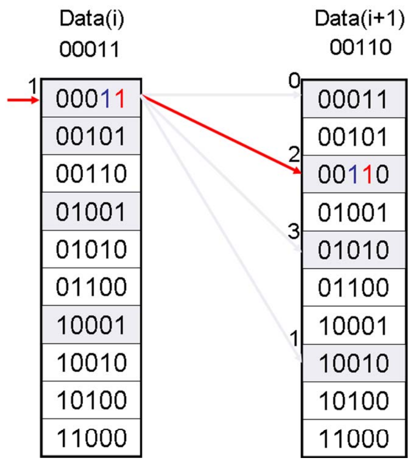

Figure 12 – Representation of decoder using a two stage trellis.

Figure 12 shows an example of data transmission. In token Data(i), the 1-of-4 “0001” value is decoded. The expected next data must include the “00010” data. Token Data(i+i) in the example arrives without error and the 1-of-4 “0010” value is decoded. The expected data for the next transmission is the “00100”. If Data(i+1) was an ICD with encoding “01110”, the erroneous data would be filtered just by applying double check. It is important to note that a simple double check is capable to filter just some portion of the ICD in the data. Double check alone is not capable to use all the power of the TRDIC encoding. A decoder based on a three stage trellis would present better results and solve even some VCD errors, what is left as future work. In this work, we only present the decoder based on double check.

VI. Hardening Evaluation Environment

In order to evaluate SEE impacts, the asynchronous QDI circuits were firstly evaluated analytically using State Transition Graphs [11] (as done in [8]), or using a simulation environment with analysis of the token propagation and progression under the presence of SEU [12]. However, in these analyses, the electrical filtering and the SEE timing pulse width are not taken in account. In order to create an environment for evaluation of the SER in real case circuits, this work has extensively used the simulation environment proposed in [13]. Using this simulation environment, it is possible to make use of commercial simulation tools and in this way have a digital flow capable to accurately characterize complex circuits.

Figure 13 presents the overall SEE characterization flow. The second and fourth column represents the regular design flow. The standard-cell library is firstly adapted to permit the synthesis of QDI circuits using commercial tools following the flow proposed in [14]. The first and the third column represent the steps necessary to perform the SEE evaluation at circuit level.

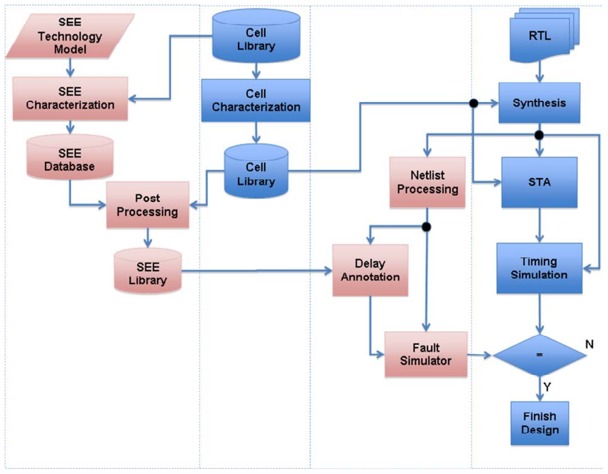

Figure 13 – SEE flow for circuit level characterization.

In a preliminary step, the SEE characterization of the standard-cell libraries is done. Compared to the initial flow proposed in [13], applied only to standard combinational + FF cells, the SEE characterization has been extended with the required C-elements to support asynchronous logic. The characterization is done by electrical simulations using an accurate SEE current injection model based on a double exponential current source. The cell characterization extracts the following information for each cell:

1. SEE critical charge. The results of critical charge are converted to time, as shown in [13], to enable the digital simulator to find the electrical filtering property of the whole circuit.
2. Pulse width at the cell output, when the pulse is presented (SET), for several different levels of charge insertion.
3. The maximum and the minimum voltage reached at the victim node. This is necessary to limit the charge tested during the circuit level evaluation. If the voltage during the characterization pass the maximum voltage tolerated by the transistor than the SEE it is not more a non permanent event.

The characterization is done for each cell, for each cell state, and for different output loads. The standard-cell library models (.lib, .v,) are then enriched by adding an additional SEE pin to each cell to model the SEE impact effect in terms of timing and functional behavior.

In a second step, once the standard cell libraries are fully characterized and modeled for SEE, it is then possible to use classical and efficient digital flow tools: timing analysis for .sdf generation, back-annotation, and gate level simulation, including precise glitch filtering. Lastly, a digital fault simulator in SystemC has been developed in order to generate random or driven SEE faults, to compare simulation results with simulation golden models and finally to collect SEE results. As a result, by using the characterization and digital simulation environment proposed using commercial tools, it is possible to accurately and rapidly characterize SEE on complex circuits.

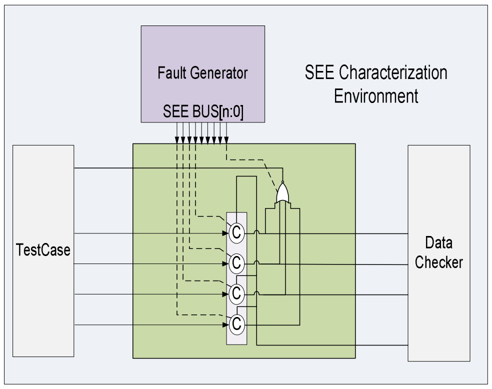

Figure 14 – SEE characterization environment.

Figure 14 shows the SEE characterization environment for the QDI pipelines. The environment is based on a fault generator and simulator developed in SystemC. The fault generator has access to all the instances in the design and generates randomly a SEE indication. The injected charge level the SEE rate can be precisely controlled.

VII. TRDIC EVALUATION

The results presented in this section are related to comparisons of the following QDI pipeline implementations:

- 1-of-4;
- 2-of-5;
- 2-of-5 with temporal redundancy (TRDIC).

The evaluated pipelines are 16 stages of 32 data bits (16 elements of asynchronous pipeline of 1-of-4 or 2-of-5 encoding). The register used is a weak conditioned half buffer. For this work, the conventional C-element implementation is considered since this implementation shows a good SEE tolerance compared to the symmetric version and is less susceptible to variation than the weak-feedback implementation [9]. The completion detection is done using trees in all the stages and for all the implementations.

The synthesis of the pipeline was done using the flow proposed in [14] for the CMOS 32nm technology. The results are related to the following operation conditions: 1Volt, 25C and typical transistor models. The evaluations were done for different levels of SEE insertion charge and different SEE insertion rates. Results presented in this work are related only to the pipelines and do not include analysis (SEE, timing, area and power) of the TRDIC Encoder and Decoder. This analysis is an ongoing work.

Figure 15 shows the comparison of the three implementations for a fixed level of injected charge. The results were obtained increasing the SEE injection rate from 1e^{6}SEEs/second until 10e^6 SEEs/second. The results show that the 2-of-5 reduces the failure rate on the QDI data link, as expected. When the temporal redundancy is applied, this rate is further reduced: 36 times in average when compared to 1-of-4 and around 11 times in average when compared to 2-of-5.

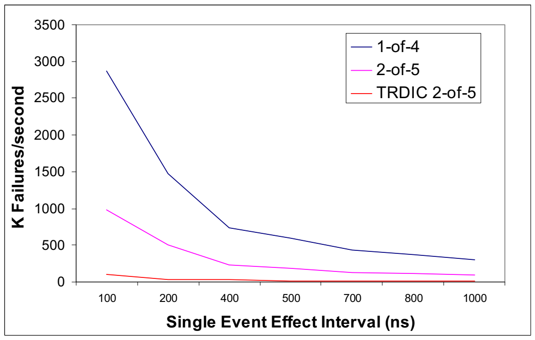

Figure 15 – Comparison between different encodings and TRDIC in function of SEE rate.

Figure 16 shows the behavior of the QDI data links when the injected rate is kept constant (5e^{6}SEEs/second) while the normalized injection charge is increased. The injected charge is normalized with the biggest charge evaluated. In the first point, no failure can be observed since this value is below the electrical filtering of the cells that compose both libraries. Starting from 0.03 point of normalized injected charge, failures start to manifest in the QDI pipeline. It is interesting to note that the QDI pipelines failure rates do not change after reaching point 0.07. This value can be used to describe the electrical filtering property of the circuit.

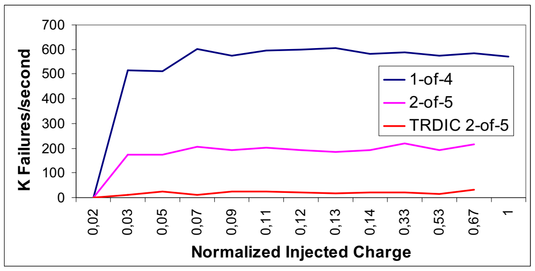

Figure 16 – Comparison between different encodings and TRDIC in function of the SEE normalized insertion charge.

The following tables give performance results (area, power and timing) of the two 1-of-4 and 2-of-5 encoding, without including the cost of the TRDIC encoder/decoder. The area increase for 2-of-5 (Table 1) is explained by a more complex encoding and its associated completion detection circuit. The design was synthesized using a standard-cell library plus an asynchronous cell library. The asynchronous library is mainly composed by C-elements and does not have adequate cells for direct mapping of m-of-n decoding. The completion detector for instance, uses 10 C-elements to detect individual 2-of-5 encodings. This could be further reduced by using a specific NCL cell to detect 2-of-5 codes.

Table 1 – Area comparison between the two encodings. The design is described in number of cells/area (gates/area in μm^2).

	Asynchronous Cells	Combinational Cells	Total
1-of-4	1264/1919.7	482/1007.7	1746/2927.4
2-of-5	4080/6120.3	1280/2142.0	5226/8338.0

Table 2 shows the power comparison between the two different QDI links implementation. The comparison was done with a data rate equal to 5.33 Gbits/second. The power increase is a consequence of the area increase necessary to implement the 2-of-5 encoding. The power, as well as the area, can be reduced with a more optimized implementation using a more complete asynchronous cell library.

Table 2 – Power comparison between the two encodings.

	Leakage(μW)	Dynamic(μW)	Total(μW)
1-of-4	134.7	2578.8	2713.5
2-of-5	317.4	5335.4	5652.8

The timing characteristics of the 2-of-5 encoding are presented in Table 3. The results show that 1-of-4 presents the best performance compared to 2-of-5. However the throughput and the latency of the 2-of-5 encoding also suffers with the lack of suitable cells.

Table 3 – Performance comparison between the two encodings.

	Maximum Throughput(Gbits/sec)	Latency(ns)
1-of-4	40.8	1.21
2-of-5	32.5	1.37

VIII. Conclusions

The proposed temporally redundant code translates a 1-of-n encoding to a 2-of-n+1 encoding, drastically reducing the probability of SEEs in QDI pipelines. The encoding also provides multi-bit correction on the receiver side while keeping the pipeline throughput close to the original 1-of-n encoding since it does not add any token on the communication link. The work presented also a precise SEE evaluation of the proposed technique through an accurate SEE characterization flow in an advanced CMOS 32nm technology. Ongoing works include an optimization of the TRDIC Encoder and Decoder designs, an adaptation of the proposed TRDIC encoding to an existing asynchronous NoC architecture, as well as library extensions to provide better support for the required cells.

➢ References

[1] J. Bainbridge and S. Furber, "Chain: A Delay-Insensitive Chip Area Interconnect," IEEE Micro, 22(5), pp. 16-23, Sep-Oct, 2002.

[2] E. Beigné, F. Clermidy, P. Vivet, A. Clouard, and M. Renaudin, "An Asynchronous NoC Architecture Providing Low Latency Service and its Multi-level Design Framework," In: IEEE Int. Symp. on Asynchronous Circuits and Systems (ASYNC'05), pp. 54-63, 2005.

[3] J. Pontes, M. Moreira, F. Moraes, and N. Calazans, "HERMES-A - An Asynchronous NoC Router with Distributed Routing." In: Int. Work. on Power and Timing Modeling, Optimization and Simulation (PATMOS'10), pp. 150-159, 2010.

[4] R. Baumann, "Radiation induced soft errors in advanced semiconductor technologies," IEEE Transactions on Device and Materials Reliability, 5(3), pp.305-316, September 2005.

[5] R. P. Bastos, G. Sicard, F. Kastensmidt, M. Renaudin, R. Reis. "Asynchronous circuits as alternative for mitigation of long-duration transient faults in deep-submicron technologies," Microeletronics Reliability, 50, pp. 1241-1246, November 2010.

[6] B. Rahbaran, A. Steininger. "Is asynchronous logic more robust than synchronous logic?" IEEE Transaction on Dependable and Secure Computing, 6(4), pp. 282-294, December 2009.

[7] M. Y. Agyekum and S. M. Nowick, "An error-correcting unordered code and hardware support for robust asynchronous global communication," In: Design, Automation & Test in Europe Conference (DATE'11), pp. 765-770, 2011.

[8] W. J. Bainbridge, S. J. Salisbury. "Glitch sensitivity and defense of quasi delay insensitive network-on-chip links," In: Asynchronous Circuits and Systems (ASYNC'09), pp. 35-44, 2009.

[9] R. P. Bastos, G. Sicard, F. Kastensmidt, M. Renaudin, R. Reis. "Evaluating transient-fault effects on traditional c-element's implementation," In: International On-Line Testing Symposium (IOLTS'10), pp. 35-40, 2010.

[10] Y. Monnet, M. Renaudin, R. Leveugle. "Hardening techniques against transient faults for asynchronous circuits, " In: International On-Line Testing Symposium (IOLTS'05), pp. 129-134, 2005.

[11] W. Jang, A. Martin. "SEU-tolerant QDI circuits," In: Asynchronous Circuits and Systems (ASYNC'05), pp. 156-165, 2005.

[12] Y. Monet, M. Renaudin, R. Leveugle. "Formal analysis of quasi delay insensitive circuits behavior in the presence of SEUs," In: International On-Line Testing Symposium (IOLTS'07), pp. 113-120, 2007.

[13] J. Pontes, P. Vivet, N. Calazans, "An Accurate Single Event Upset Digital Design Flow for Reliable System Level Design.", In: Design, Automation and Test in Europe, Dresden (DATE'12), 2012.

[14] Y. Thonnart, P. Vivet, F. Clermidy. "A fully-asynchronous low-power framework for GALS NoC integration." (DATE'10), pp. 33-38, 2010.

[15] J. Sparsø and S. Furber, "Principles of Asynchronous Circuit Design – A Systems Perspective". Kluwer Academic Publishers, Boston, 354p. 2001.