Designing Robust Checkers in the Presence of Massive Timing Errors

Frédéric Worm, Patrick Thiran, Paolo Ienne
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne (EPFL)
{Frederic.Worm,Patrick.Thiran,Paolo.Ienne}@epfl.ch

ABSTRACT

So far, performance and reliability of circuits have been determined by worst-case characterization of silicon and environmental noise. As new deep sub-micron technologies exacerbate process variations and reduce noise margins, worst-case design will eventually fail to meet an aggressive combination of objectives in performance, reliability, and power. In order to circumvent these difficulties, researchers have recently proposed a new design paradigm: self-calibrating circuits. Design parameters (e.g., operating points) of self-calibrating circuits are set by monitoring correctness of their operation, thus enabling to dynamically trade reliability for power or performance, depending on actual silicon capabilities and noise conditions.

In this paper, we study the problem of detecting errors caused by self-calibration of the supply voltage and frequency of an on-chip link. These errors are caused by operation at sub-critical voltage and may be numerous. We attack the problem with a coding technique. We also discuss an alternative approach using double sampling flip-flops. We stress the complementarity of the two approaches and show how they can be combined. Finally, we consider extending our work to computation. We give preliminary research directions on the detection of errors induced by self-calibration for an adder.

1. INTRODUCTION

With the advance of CMOS technologies, designers are facing increasingly large difficulties in meeting simultaneous objectives in performance, reliability, and power. For this reason, self-calibration is gaining momentum, both at the device [6] and circuit level [1, 12].

The main focus of this paper is on a self-calibrating link, which we depict in Figure 1. We consider designing such a system without worst-case characterization of the link itself. Yet, the encoder, decoder and controller are designed traditionally, i.e., with worst-case assumptions. Metastability is avoided by inserting two flip-flops in a row before the decoder. Alternatively, metastability tolerant flip-flops can be used [1]. The voltage swing of a self-calibrating link is set dynamically to handle uncertainties about process and noise conditions. Likewise, dynamic voltage scaling techniques handle uncertainties about workload by matching power consumption to the actual performance requirement. Correct operation is verified by a *checker*. Furthermore, a controller reacts to detected errors by increasing reliability at the expense of performance (e.g., by decreasing frequency) or power (e.g., by increasing voltage). Although

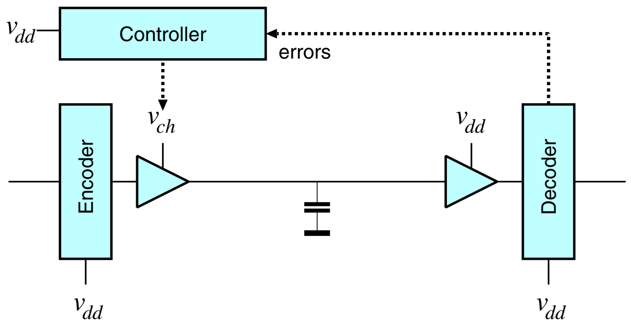

Figure 1: An error detecting code provides the controller with a feedback about link operation. The link supply voltage is adjusted depending on the occurrence of detected errors.

the controller should not request operating points where the checker reliability is poor, self-calibration offers a large flexibility to exploit design trade-offs. Example of control algorithms include *tracking* a target error rate. A more conservative control consists in increasing voltage as soon as an error is reported and periodically decreasing voltage. In any case, a self-calibrating link may be operated temporarily at sub-critical voltage resulting in a bit error rate as large as 1.

The requirement of detecting errors even under a large bit error rate renders the encoding problem very original. This particularity excludes readily the use of error correcting codes since none offers sufficient correcting capabilities at a tolerable overhead. Similarly, techniques such as triple modular redundancy cannot be used because errors in individual systems are strongly correlated with each others. Although asynchronous codes do not require any worst-case knowledge about the link delay, the paper focuses on synchronous implementations, which have less wiring overhead.

1.1 Structure of the Paper

Section 2 describes the type of errors we consider, i.e., timing errors and their associated communication channel. In Section 3, we describe a coding scheme obtained by augmenting standard CRC codes with time redundancy. In Section 4, we discuss a few techniques related to the detection of timing errors. We focus especially on double sampling flip-flops. Section 5 emphasizes the complementarity of double sampling flip-flops to the coding scheme introduced in Section 3. Moreover, we show how the two approaches can be combined into a more robust checker. Then, Section 6 gives perspectives towards self-calibrating computation. Finally, Section 7 summarizes our achievements.

2. ERROR MODEL

When operating the link at a sub-critical voltage, the decoder is likely to sample incorrect data, because transitions from the previous piece of data have not completed. We call this type of error—i.e., the failure of bit transitions—timing errors. Moreover, *additive* errors—i.e., bit flips—can corrupt transmitted data. Self-calibrating communication has thus the following features: it is affected by timing errors of very large error rate, and additive errors of low error rate. We now define in more detail timing errors. We denote by x_k the k^{th} input data bit and by y_k the k^{th} data bit sampled by the decoder. Moreover, we denote by $e_k \in \{0,1\}$ the failure of sampling the k^{th} data bit: $e_k = 1$ indicates that the sampling was too early, whereas $e_k = 0$ indicates a correct sampling. A timing error occurs whenever $y_k \neq x_k$, which requires the two following conditions to be met:

- a transition has to occur on the channel, i.e., $x_k \neq x_{k-1}$, and
- the sampling has been too early, i.e., $e_k = 1$.

We model the correctness of sampling with a Bernouilli random variable. More precisely, e_k is described by a sequence of i.i.d.[1] Bernouilli random variables

$$e_k = \begin{cases} 1 & \text{with probability } \varepsilon_{\text{t}}, \\ 0 & \text{with probability } (1-\varepsilon_{\text{t}}), \end{cases} \tag{1}$$

where ε_{t} is the timing error rate, i.e., the probability that a bit transition is sampled too early. In this context, this probability is also referred to as bit error rate. It holds thus $P\left(y_k \neq x_k \mid x_k \neq x_{k-1}\right) = \varepsilon_{\text{t}}$.

For a N-bit wide data, x_k, y_k, and e_k constitute N-bit vectors. The relation between the channel output, y_k and the channel input, x_k, is then given by

$$y_k = x_k \oplus e_k \cdot (x_k \oplus x_{k-1}). \tag{2}$$

Eq. (2) models the fact that $y_k = x_k$ if $e_k = \vec{0}$ or $x_k \oplus x_{k-1} = \vec{0}$. We call a channel described by Eq. (2) a *timing error channel*. Figure 2 shows the graphic model of a timing error channel.

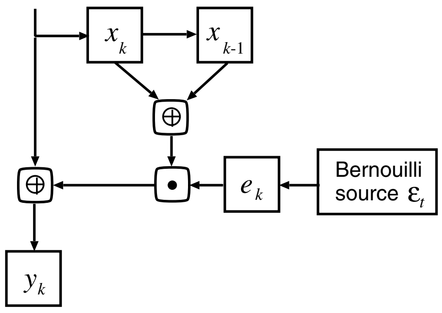

Figure 2: A timing error channel.

[1]Independent and identically distributed.

In the next section, we show how the detection of timing errors is enhanced by adding redundancy not only in space—such as CRCs do—but also in time.

3. ALTERNATING-PHASE ENCODING

Alternating-phase encoding has been proposed and studied as a low wiring overhead scheme offering a strong robustness to both timing and additive errors [11]. We describe it briefly.

Let $k \bmod 2$ be the *phase* of data x_k. The phase is thus a binary information about the sequencing of data. Consider a systematic encoding. The input data is augmented with a phase bit. The phase bit is not transmitted; instead, it is generated locally and synchronously by the encoder and decoder. However, redundant bits contain information about the phase bit. On the decoder side, the latter is included into the received data before verifying the parity checks, as depicted in Figure 3. Such an encoding scheme actu-

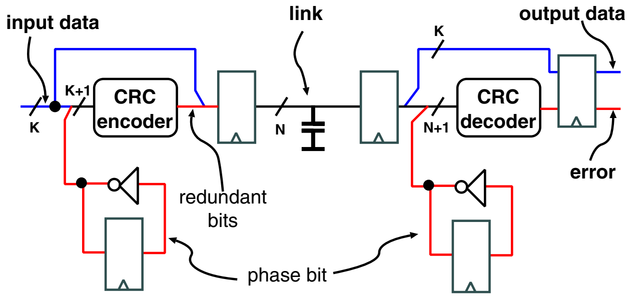

Figure 3: The input (respectively received) data is augmented with a phase bit that is not transmitted but generated by the encoder (respectively decoder).

ally alternates transmission with codewords belonging to two codes, C_0 and C_1, where C_0 (respectively C_1) is the set of codewords obtained by encoding the input data with a 0 (respectively 1) phase bit. For this reason, this encoding is called *alternating-phase* encoding.

The reliability of a given encoding is measured by its *residual* word error rate, which is the probability of an undetected word error. We have bounded and approximated the residual word error rate of linear error detecting codes (under some assumptions about the code) over the timing error channel [11]. We have also upper bounded and approximated the residual word error rate of alternating-phase encoding using linear codes. In Figure 4, we plot these approximations for the 8-bit CRC generated by the polynome $x^8 + x^2 + x + 1$ used to encode 32 information bits. With small timing error rates (roughly, less than 10^{-2}), the detection capability of both the normal and the alternating-phase CRC are satisfactory. The residual word error rate is maximum for a bit error rate equal to 0.5. In such a case, data received by the decoder is completely mixed between the previous one and the correct one, which is the worst situation for the decoder. However, as the bit error rate further increases, the residual word error rate of alternating-phase encoding eventually reaches 0. This is due to the decoder phase bit that does not match the phase of the received data. If the bit error rate becomes as large as 1, the normal CRC does not detect any error since the same codeword is

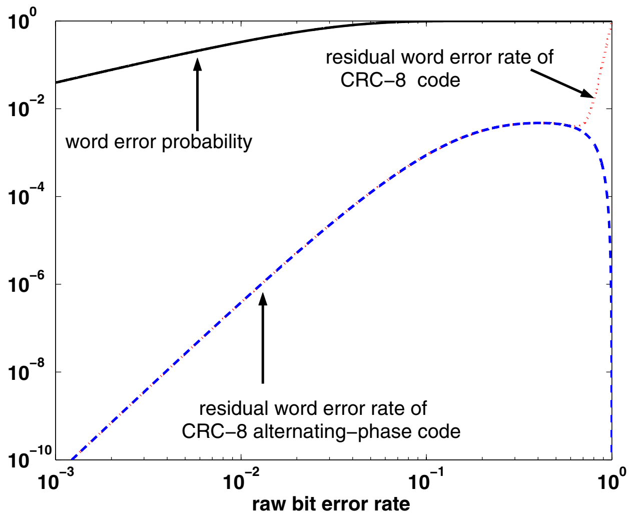

Figure 4: Residual word error rate of the normal 8-bit CRC generated by the polynome $x^8 + x^2 + x + 1$ and of the alternating-phase encoding obtained with the same CRC.

sampled twice consecutively: this situation is actually mistaken for the correct transmission of, indeed, twice the same information. On the contrary, alternating-phase encoding detects deterministically the error as the phase bit added by the decoder causes a single bit error, which is always detected by codes like CRCs.

During the self-calibration process of an on-chip link, we believe that it would be unlikely to operate in the region of moderate bit error rates where reliability is minimum. The reason is that, as voltage is lowered to sub-critical values, we expect the transition from low to high bit error rates to be very steep.

Alternating-phase encoding can be considered as a generalization of the LEDR code [4]. LEDR adds one redundant bit to each information bit. The redundant bit is obtained by *xor*-ing the phase with the information bit, i.e., it is alternatively equal and opposite to the information bit. For 2-bit symbols, LEDR alternates two codes: $\{(00), (11)\}$ and $\{(01), (10)\}$. That is, LEDR alternates the parity of the transmitted codeword. As shown in Figure 5, a K-bit LEDR encoder is a particular alternating-phase encoding where each information bit is involved in a single parity check relation with the phase bit. Indeed, each *XOR* gate can be considered as a 1-bit CRC encoder. The wiring overhead amounts thus to 100%. Although having a poor detection capability of additive errors, LEDR detects all timing errors. On the contrary, the proposed scheme does not detect all timing errors, but has a much reduced wiring overhead and a better detection of additive errors.

4. RELATED WORK

We describe briefly a few techniques targeted at the detection of timing errors, other than alternating-phase encoding. However, we do not discuss traditional fault-tolerant techniques since they aim at recovering from relatively rare and uncorrelated faults. On the contrary, errors occurring during self-calibration are strongly correlated and may be numerous. In such a particular situation, modular redundancy (i.e., duplicating or triplicating an identical circuit) is impractical because the failures of each of these circuits are strongly correlated.

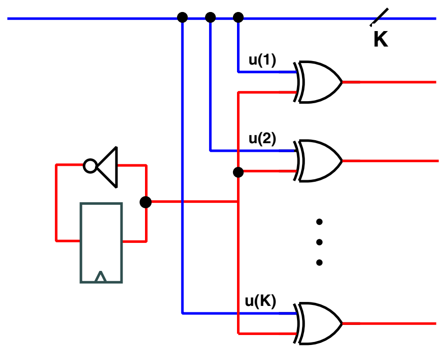

Figure 5: Synchronous implementation of a K-bit LEDR encoder.

4.1 Coding Techniques for Timing Errors

Unordered codes [3] have the property that for any codeword p, there exists no other codeword q satisfying $q^i = 1$ for all bit positions i where $p^i = 1$. For example, constant weight codes (such as 1-of-N codes or the Sperner codes [5]) and the Berger code [2] are unordered. The unordering property ensures the detection of all possible timing errors: unordered codes are also referred to as self-synchronising or delay-insensitive codes. However, most of them either incur a large wiring overhead, or require complex encoders, or detect poorly additive errors—contrary to alternating-phase encoding.

Rossi et al. have recently proposed to optimize the wiring of error correcting codes in order to mitigate crosstalk effects [10]. While their scheme is able to correct single errors, it is not meant to detect the numerous errors that would occur during operation at sub-critical voltage.

4.2 Razor Flip-Flops

Double sampling data—i.e., adding intra-cycle time redundancy—has been proposed by Nicolaidis [8] as a mean of recovering from transient faults, which exploits their locality in time. More recently, double sampling flip-flops have been applied to correct timing errors resulting of self-calibration [1]. As Figure 6 shows, a double sampling flip-flop consists of a main flip-flop—fed by a normal clock–and a so-called *shadow-latch* fed by a delayed clock. Data at the input is first latched by the main flip-flop. Then, it is latched after some delay by the shadow latch. The setup and hold constraints of the latter are ensured by worst-case assumptions. That is, variations in arrival time of the input signal are subject to both best-case and worst-case assumptions. Provided these assumptions are met, the data held by the shadow latch acts a "reference". Next, a metastability-tolerant comparator validates the data latched in the main flip-flop by comparing it with the one of the shadow latch. In case a timing error—i.e., a mismatch—is detected, the data held by the shadow latch is re-issued at the main flip-flop input, while at the same time asserting an error signal. The timing error is thus corrected at the expense of a one cycle latency (in a processor pipeline, error recovery actually

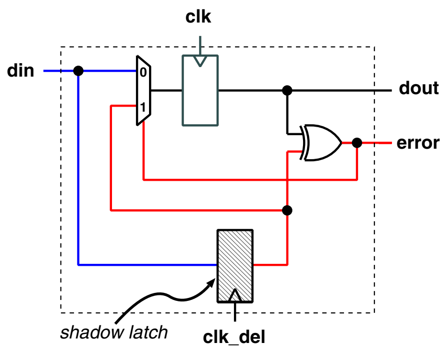

Figure 6: A Razor flip-flop.

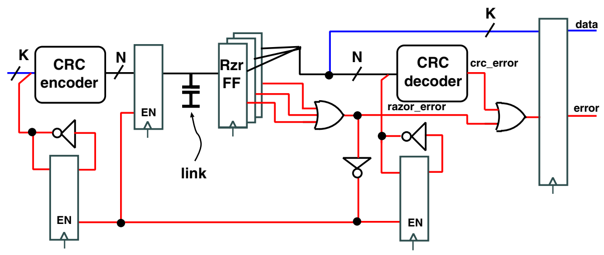

Figure 7: A self-calibrating link combining Razor flip-flops with alternating-phase encoding.

incurs a larger penalty).

As mentioned before, a Razor flip-flop operates reliably as long as the setup and hold constraints of the shadow latch are met. These constraints are ensured by both best-case and worst-case assumptions about the data arrival time. The worst-case constraint is somewhat relaxed, because the shadow latch is fed with a delayed clock. More specifically, a too late data arrival results in an undetected timing error since both the main flip-flop and shadow latch hold the same corrupted data: no mismatch is detected. On the contrary, a too early data arrival causes a short path error, whereby the data held in the main flip-flop—although correct—is invalidated due to the early arrival of the the next data held in the shadow latch. In such a situation, a timing error is reported although none actually occurred.

In the next section, we explain why Razor flip-flops and codes constitute complementary checkers and show how they can be combined.

5. A COMBINED CHECKER

When considering checker for a self-calibrating on-chip link, Razor flip-flops and alternating-phase codes are complementary in many aspects, namely:

1. **Reliability.** Razor flip-flops correct all timing errors provided the link propagation delay lies in an interval such that both short-path errors (too short delay) and undetected errors (too long delay) are avoided. Contrary to Razor flip-flops, the residual error rate of alternating-phase code is maximal when the link propagation delay lies in a specific interval: according to Figure 4, it is approximately in the range 0.1 to 0.9. In addition, alternating-phase code provides a reliable error signal under both large and small delay, which is exactly in situations where the error signal of Razor flip-flops is unreliable.
2. **Correction/Detection Capability.** Differently from Razor flip-flops, alternating-phase encoding cannot correct timing errors as efficiently as the former enable since correction is achieved by retransmission.
3. **Control.** Used as a checker, Razor flip-flops require unsafe operating points to be a-priori forbidden by worst-case assumptions. Such unsafe operating points cannot be automatically discovered by the controller because they may violate the timing constraints of the shadow latch. On the other hand, controlling a circuit with alternating-phase encoding does not require the a priori exclusion of unsafe operating points. However, the control algorithm needs to react immediately to reported errors in order to make sure the checker is not operated where its reliability is low. Indeed, the controller has to interpret the feedback it receives about detected word errors as a correlated feedback about undetected word errors.

Figure 7 depicts how to combine simply the two checkers and obtain the benefits of each one. The two checkers are used in a serial combination. The data sent through the link is augmented with redundant bits computed from information bits and the phase. At the link output, data is first sampled by Razor flip-flops before the decoding occurs. The phase of the decoder is preserved each time at least one Razor flip-flop reports an error in order to keep consistency between the decoder phase and the phase of the decoded data. While most errors are expected to be corrected by Razor flip-flops, undetected or falsely reported timing errors result in data being input to the decoder with a phase mismatch, which is likely to raise the *crc_error* signal. That is, the efficient correction capability of Razor flip-flops is preserved, while retransmissions are triggered only in situations that would have resulted in an undetected error had Razor flip-flops been used as a single checker.

5.1 Reliability

In order to estimate the reliability of the combined checker, we have simulated in VHDL the transfer of over $500'000$ 8-bit data words. We have measured the word error rate and residual word error rate. The residual word error rate is computed as the ratio of the number of words delivered and corrupted to the total number of word transfers. We have modelled a 130nm technology.

Table 5.1 compares the residual error rate of different checkers: *C3*, i.e., alternating-phase encoding with the CRC generated by $x^3 + x + 1$, *RZR*, i.e., 8 Razor flip-flops, and *RZR+C1*, i.e., a checker complementing the 8 Razor flip-flops with a single parity bit computed over the information bits and the phase bit. The word error rates observed for the three checkers are very similar, essentially because words send through the link have comparable size. We give thus only the word error rate for *C3*. *RZR+C1*, a minimal combined checker, reduces the residual error rate of two orders of magnitude compared to RZR when the word error rate is less than 100%, which is where the checker is expected to be operated most of the time. The maximum residual error rate of *C3* occurs at 0.85V. Actually, we could not measure residual errors for $C3$ below 0.8V. The maximum residual error rate of RZR+C1 occurs at 0.65V. In comparison to

voltage	0.8V	0.85V	0.9V	0.95V	1V
WER C3	0.99	0.94	0.58	0.10	$7.8 \cdot 10^{-3}$
RES.WER C3	$8.0 \cdot 10^{-2}$	$8.9 \cdot 10^{-2}$	$1.9 \cdot 10^{-2}$	$4.3 \cdot 10^{-4}$	$3.3 \cdot 10^{-6}$
RES.WER RZR	0.13	$7.7 \cdot 10^{-3}$	$1.5 \cdot 10^{-4}$	N.M.	N.M.
RES.WER RZR+C1	$2.3 \cdot 10^{-2}$	$7.0 \cdot 10^{-5}$	N.M.	N.M.	N.M.

Table 1: Word error rate (WER) and residual word error rate (RES.WER) of different checkers as a function of the link supply voltage. N.M. means that no residual error could be measured during the whole simulation.

$C3$, the overall reliability is increased as much more detected errors can warn the operating point controller before reaching this highly sub-critical voltage.

We see the opportunity of relaxing the coding problem due to the presence of Razor flip-flops as a remarkable property of the combined checker: the reliability of alternating-phase encoding is important only from the point on when timing errors are not detected any more by Razor flip-flops. This is a particular situation of large error rate where the detection of errors essentially relies on the notion of phase and not on complex spatial redundancy.

Next, we briefly discuss how error recovery is performed in a system using a combined checker such as described in this section.

5.2 Error Recovery

The combined checker is organised as a 3-stage pipeline. We point out that this fact in itself does not constitute an additional overhead since coding techniques are already deployed over on-chip links—see for instance the implementation of practically all internal buses of the Itanium 2 processor [7].

While timing errors may either be corrected by Razor flip-flops or by retransmission, data should be delivered at the decoder output only if it is correct and in sequence. For example, data output after a retransmission is not delivered even though it may have no error, since the next data to be delivered in-sequence is the retransmitted one. In summary, a piece of data is delivered only if no error has been detected during its transfer and it is in-sequence. To do so, an additional entity, namely, an *Automatic Repeat reQuest* (ARQ) controller, decides which piece of data should be delivered. Such a controller has a very low hardware cost. A synthesis of this controller shows that it consists of only 6 flip-flops and about 20 gates. Actually, a very similar controller is required in any system that uses codes to detect errors.

6. EXTENSION TO COMPUTATION

In this section, we consider extending the combination of Razor flip-flops with “relaxed” codes to an adder. We give thereby preliminary research directions towards self-calibrating adder circuits independent of worst-case assumptions about the adder itself. The discussion expands on prior work about parity prediction for adders and ALUs [9].

We denote respectively by A, B, C, and S the two input operands, internal carries and sum of an N-bit adder. C contains the carry-in c_{in} and the $N-1$ internal carries. Moreover, we denote respectively by $p(A)$, $p(B)$, $p(C)$, and $p(S)$ the parity of the inputs, internal carries, and sum. The bitwise relation between A, B, C, and S is $S_i = A_i \oplus B_i \oplus C_i$, with $1 \leq i \leq N$ and $c_1 = c_{\text{in}}$. It follows that $p(S) = p(A) \oplus p(B) \oplus p(C)$. By reordering

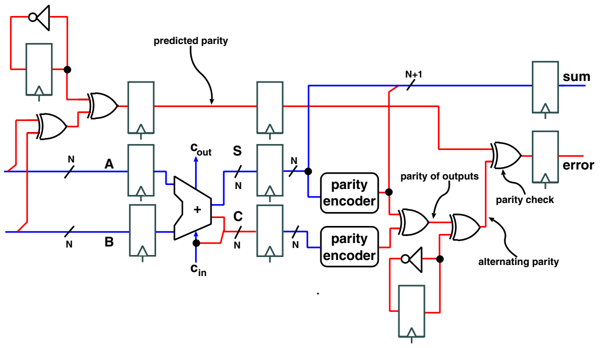

Figure 8: An adder with an alternating parity prediction scheme. A and B are the two parity-encoded operands. S and C are respectively the sum and internal carries.

the terms, we obtain the following input-output relation: $p(A) \oplus p(B) = p(C) \oplus p(S)$. That is, one can consider the adder as a single element with a $2N$-bit input, $A \mid B$, producing a $2N$-bit output $C \mid S$, where $\cdot \mid \cdot$ denotes the concatenation operator. From this point of view, the adder preserves parity, i.e., $p(A \mid B) = p(C \mid S)$.

Figure 8 is a natural translation of Figure 3 in the context of an adder. The encoding also includes the notion of phase. In Section 3, we have mentioned that LEDR alternates the parity of transmitted codewords. Likewise, the parity predicted in the encoding of Figure 8 alternates at each new addition between the parity $p(A) \oplus p(B)$ and its opposite. While such a scheme prevents from delivering twice the same adder output in case of over-aggressive operation, it is expected to have a poor detection capability under moderate timing bit error rate, since any even number of errors leaves the parity unchanged. Two non-mutually exclusive options are available to increase the reliability of this scheme. The first one consists in using Razor flip-flops to increase the detection of timing errors, as already discussed in Section 5 for the case of a link. The second option focuses on increasing the code error detection capability.

The addition of Razor flip-flops is, in turn, a natural translation of Figure 7 to an adder, as depicted in Figure 9. The resulting checker is expected to be much more robust than the one described in Figure 8, since its reliability relies on the single parity check only when all Razor flip-flops do not detect errors any more, i.e., in a situation of large timing error rate. This is the same reason as already invoked in Section 5 to justify why the coding requirements may be relaxed when combined with Razor flip-flops.

On the other hand, we see two ways of improving reliability of the coding scheme.

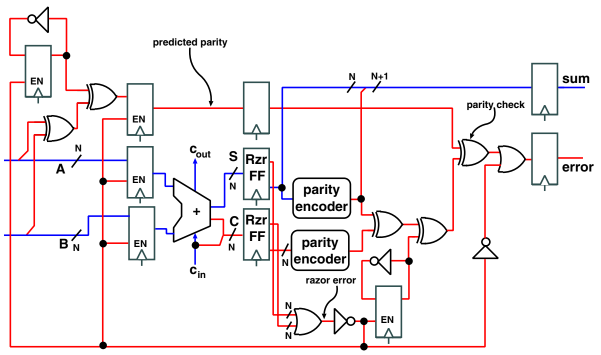

Figure 9: An adder combining an alternating parity prediction scheme with Razor flip-flops. *A* **and** *B* **are the two parity-encoded operands.** *S* **and** *C* **are respectively the sum and internal carries.**

1. Including more than a single parity check. Since parity is preserved by addition if it is computed on consecutive bits, different parity checks can be performed as long as each of them bears on a set of consecutive bits. The supports of the parity checks need not be disjoint.
2. Using other codes preserved by addition, such as the Berger code [2] or low-cost residue codes where the redundant bits are the remainder of the division of the information bits by $2^n - 1$, e.g., 3 or 7.

Both items involve trade-off between overhead and reliability. As a future work, we will investigate the reliability of the schemes presented in Figures 8 and 9 under the presence of numerous timing errors.

7. CONCLUSION

Self-calibrating circuits rely on a checker to ensure reliability. By their dynamic nature, they may be temporarily exposed to error rates as large as 100% due to operation at sub-critical voltage. Yet, and especially under these conditions, the checker should provide a reliable feedback about circuit operation.

In this paper, we have first reviewed a particular checker for self-calibrating links based on a coding technique as well as an error model taking into account that the link output may be sampled while individual bit transitions have not yet completed. We call such errors timing errors. The coding scheme we have discussed in Section 3, alternating-phase encoding, combines spatial redundancy with temporal redundancy obtained from a binary information about data sequencing. It can be considered as a significant extension from LEDR, a particular asynchronous code detecting any timing error at the expense of a 100% wiring overhead. By an analytical approximation, we have illustrated the detection capability of alternating-phase encoding. In particular, we have emphasized that a high level of reliability is ensured both under small and very large bit error rates.

Then, we have contrasted the detection capabilities of alternating-phase encoding with the one of double sampling Razor flip-flops. Reliable operation of Razor flip-flops is limited by the necessity of meeting the timing constraints of the shadow latch. In particular, reliability is not guaranteed under large timing error rates, while errors may be falsely reported due to short-path errors under small error rate. To alleviate these limitations, we have shown that Razor flip-flops can be combined with alternating-phase encoding. In case of an undetected timing error or short-path error, data output by Razor flip-flops does not match the phase of the decoder, which is likely to cause a detected error. From simulations of the residual error rate of the combined checker, we have observed that codes with minimal redundancy (e.g., a single parity check over 8-bit data) enable a significant decrease in the residual error rate.

Finally, expanding on the opportunity to relax coding requirements, we have given preliminary directions on how to combine low-cost codes with double sampling flip-flops on an adder operated at sub-critical voltage.

8. REFERENCES

[1] T. Austin, D. Blaauw, T. Mudge, and K. Flautner. Making typical silicon matter with Razor. *Computer*, 37(3):57–65, Mar. 2004.

[2] J. M. Berger. A note on error detecting codes for asymmetric channels. *Information and Control*, 4(1):68–71, Mar. 1961.

[3] B. Bose. On unordered codes. *IEEE Transactions on Computers*, C-40(2):125–31, Feb. 1991.

[4] M. E. Dean, T. E. Williams, and D. L. Dill. Efficient self-timing with level-encoded two-phase dual-rail (LEDR). In *Proceedings of the 1991 University of California/Santa Cruz Conference on Advanced Research in VLSI*, pages 55–70. MIT Press, Mar. 1991.

[5] C. V. Freiman. Optimal error detecting codes for asymmetric binary channels. *Information and Control*, 5(1):64–71, Mar. 1962.

[6] C. Kim, S. Hsu, R. Krishnamurthy, S. Borkar, and K. Roy. Self-calibrating circuit design for variation tolerant vlsi systems. In *11th IEEE International On-Line Testing Symposium (IOLTS'05)*, pages 100–105, Saint-Raphael, France, July 2005.

[7] C. McNairy and D. Soltis. Itanium 2 processor microarchitecture. *IEEE Micro*, 23(2):44–55, Mar.–Apr. 2003.

[8] M. Nicolaidis. Time-redundancy based soft-errors tolerance to rescue nanometer technology. In *17th IEEE VLSI Test Symposium*, pages 86–94, Apr. 1999.

[9] M. Nicolaidis. Carry checking/parity prediction adders and alus. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, VLSI-11(1):121–128, Feb. 2003.

[10] D. Rossi, A. K. Nieuwland, A. Katoch, and C. Metra. Exploiting ECC redundancy to minimize crosstalk impact. *IEEE Design and Test of Computers*, 22(1):57–70, Jan.–Feb. 2005.

[11] F. Worm, P. Ienne, and P. Thiran. Soft self-synchronising codes for self-calibrating communication. In *Proceedings of the International Conference on Computer Aided Design*, San Jose, Calif., Nov. 2004.

[12] F. Worm, P. Ienne, P. Thiran, and G. De Micheli. A robust self-calibrating transmission scheme for on-chip networks. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, VLSI-13(1):126–39, Jan. 2005.