Multi-fault Attack Detection for RNS Cryptographic Architecture

Jean Claude Bajard*, Julien Eynard† and Nabil Merkiche‡
*Sorbonne Universités, UPMC, CNRS, LIP6, Paris, France.
†Department of Electrical and Computer Engineering, University of Waterloo.
‡DGA-MI, Rennes, France.
jean-claude.bajard@lip6.fr, jeynard@uwaterloo.ca, nabil.merkiche@intradef.gouv.fr

***Abstract*—Residue Number Systems (RNS) have been a topic of interest for years. Many previous works show that RNS is a good candidate for fast computations in asymmetric cryptography by using its intrinsic parallelization features. A recent result demonstrates that redundant RNS and modular reduction can fit together efficiently, providing an efficient RNS modular reduction algorithm owning a single-fault detection capability. In this paper, we propose to generalize this approach by protecting the classical Cox-Rower architecture against multi-fault attacks. We prove that faults occuring at different places and at different times can be detected with a linear cost for the architecture and a constant time for the execution.**

***Index Terms*—Public-Key Cryptography, Residue Number Systems, Side-Channel Attacks, Multi-Fault, FPGA**

I. Introduction

Residue Number Systems (RNS) have been proven to be a good candidate for achieving fast computation in finite fields [5], [6], [9], [15], [18], which is a critical issue for limiting latency of public key cryptography. In practice, most of the competitive hardware implementations rely on the so-called Cox-Rower architecture, introduced by Kawamura et al. [15], which is designed to fit with natural properties of RNS.

When critical applications, e.g. related to banks and credit cards, are implemented in an embedded system, they must be supplied with protections against side-channel attacks, such as timing attacks [12], or simple/differential analysis [7], [13]. For this purpose, prior works showed that RNS naturally provides a Leak Resistant Arithmetic [4]. Besides, fault attacks can remain a real threat despite protections against leaks. Thus, fault detection is an important security features which should be integrated into a crypto-chip. Again, RNS owns a natural detection solution through the use of redundancy [17]. But, due to the particular structure of RNS modular reduction, this kind of approach seemed not compliant with finite field RNS arithmetic. Recently [2], this issue has been settled thanks to a new redundant RNS modular multiplication algorithm which includes a single-fault detection ability. Nevertheless, it does not guarantee detection of multiple faults in time and/or space inside the Cox-Rower architecture, making such multi-fault attack potentially invisible and successful.

This work has been supported in part by the European Unions H2020 Programme under grant agreement number ICT-644209 and ANR ARRAND 15-CE39-0002-01.

In this work, we propose a solution for detecting a multi-fault attack in space and in time on a cryptographic device. By using a stronger fault model, we show how to accomplish the matching between redundant RNS and finite field arithmetic. The multi-fault model offers the possibility to model as well precise targeted attacks as hardware failures which could impact several RNS channels. In order to provide a very practical fault resistant modular reduction algorithm, the fault model is adapted to hardware constraints.

The paper is organised as follows. Section II gives background on standard/redundant RNS, required in the rest of the paper. Section III is the core part where Thm. III.3 and Thm. III.4 are the main novelties. It deals with the use of redundant RNS to detect multiple faults. In section IV, practical considerations about the integration of the solution in a hardware implementation relying on a Cox-Rower architecture are suggested. In particular, the area overhead is analysed. Finaly, some conclusions are drawn.

II. Background Overview

A. Residue Number Systems

Residue number systems [8] are non positional numeral systems, based on the Chinese Remainder Theorem (CRT). The CRT states the existence of a ring isomorphism $\varphi_{\mathcal{B}} : \mathbb{Z}_M \xrightarrow{\sim} \mathbb{Z}_{m_1} \times \ldots \times \mathbb{Z}_{m_n}$ (where $M = \prod_{i=1}^{n} m_i$) as soon as the "base" $\mathcal{B} = \{m_1, \ldots, m_n\}$ is composed of pairwise coprime moduli. Consequently, the arithmetic in the interval $[0, M)$ can be replaced by independant computations in the channels $\mathbb{Z}_{m_i}$. In practice, these concurrent channels are implemented in parallel arithmetic unit called Rowers. Given an integer x, we will denote x_i or $|x|_{m_i}$ its residue in $\mathbb{Z}_{m_i}$.

The inverse $\varphi_{\mathcal{B}}^{-1}(\boldsymbol{x}_{\mathcal{B}}) = x$ can be computed as follows:

$$\varphi_{\mathcal{B}}^{-1}(\boldsymbol{x}_{\mathcal{B}}) = |\sum_{i=1}^{n} \left|x_i M_i^{-1}\right|_{m_i} M_i|_M = \sum_{i=1}^{n} \xi_{i,x} M_i - \kappa_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}}) M. \quad (1)$$

M_i is $\frac{M}{m_i}$, $\left|M_i^{-1}\right|_{m_i}$ denotes its inverse modulo m_i, and $\xi_{i,x}$ refers to $\left|x_i M_i^{-1}\right|_{m_i}$. It is direct to notice that the integer $\kappa_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}})$ belongs $[0, n-1]$ and is equal to

$$\kappa_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}}) = \sum_{i=1}^{n} \frac{\xi_{i,x}}{m_i} - \frac{x}{M} = \lfloor \sum_{i=1}^{n} \frac{\xi_{i,x}}{m_i} \rfloor. \quad (2)$$

In the following, we may represent an integer through a vector of residues. Such vectors may be merged with integers inside a same formula to point out more easily certain properties.

1063-6889/16 $31.00 © 2016 IEEE
DOI 10.1109/ARITH.2016.16

B. Base conversions

While the Rowers are physical units implementing the basic concurrent RNS arithmetic, the Cox unit is dedicated to the compution of $\kappa_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}})$ during a base conversion. To switch between two bases, two main approaches are used in practice: by using (1) (similar to Lagrange's interpolation), or through a positional mixed radix system (similar to Newton's interpolation). However, the latter is a real bottleneck because it is intrinsically sequential. In practice, the Cox-Rower architecture is designed to efficiently compute the first variant.

In CRT based conversion, an approximation of the factor $\kappa_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}})$ is computed by the Cox unit [11]. To be precise, when $2^{r-1} < m_i < 2^r$, the Cox unit computes:

$$\tilde{\kappa}_{\mathcal{B}}\left(\boldsymbol{x}_{\mathcal{B}}\right) = \lfloor \sum_{i=1}^{n} \frac{\mathrm{trunc}_h\left(\xi_{i,x}\right)}{2^h} \rfloor \tag{3}$$

where the function trunc_h returns the h most significant bits. h is a parameter of approximation. If h matches well with parameters r and n, one may obtain either $\tilde{\kappa}_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}}) = \kappa_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}})$ or $\tilde{\kappa}_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}}) = \kappa_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}}) - 1$. The conversion $\mathtt{Bex}(\mathcal{B}, x)$ derived from this approach returns an almost reduced value: $\mathtt{Bex}(\mathcal{B}, x) = x$ or $x + M$.

The error of approximation can be bounded by a real number $0 \leqslant \Delta < 1$ (cf. [11] for more details). For any x in $[0, M)$, such Δ satisfies the following:

$$-\Delta \leqslant \sum_{i=1}^{n} \left(\frac{\mathrm{trunc}_h(\xi_{i,x})}{2^h} - \frac{\xi_{i,x}}{m_i}\right) \leqslant 0. \tag{4}$$

This error can be corrected by adding an offset $\alpha \in [\Delta, 1)$ to the sum (3) (inside the flooring) computed by the Cox. In the next discussions, this corrected conversion is denoted $\mathtt{Bex}_\alpha$. $\mathtt{Bex}_0$ is the previous uncorrected variant $\mathtt{Bex}$. Moreover, it will be useful to notice that, for $\mathtt{Bex}$ and any $x < M$, $0 \leqslant \mathtt{Bex}(\mathcal{B}, x) < (1 + \Delta)M$.

C. RNS arithmetic in $\mathbb{F}_p$ within a Cox-Rower architecture

RNS modular reduction relies on Montgomery's approach [14]. Different variants are possible, depending on the base conversion methods which are used [1], [11], [16]. All these techniques involve two coprime RNS bases $\mathcal{B}$ and $\mathcal{B}'$. When using conversions described in previous part, the principle of computation of a modular reduction within a Cox-Rower architecture is briefly summarised in Alg. 1 (cf. Fig. 2 and 3 in [11] for detailed algorithms run in the Cox-Rower). The conversion Bex is used from $\mathcal{B}$ to $\mathcal{B}'$, because one does not need a complete conversion at this step. However, the second one has to be complete, so as to keep consistency between residues in the two bases. It is possible to do so as soon as $\boldsymbol{s}_{\mathcal{B}'} < (1-\alpha)M'$. This justifies the condition $2p \leqslant (1-\alpha)M'$, because the result verifies $s < 2p$.

Algorithm 1 $\mathtt{RNSMR}(x, p, \mathcal{B}, \mathcal{B}')$

Require: $\mathcal{B}$, $\mathcal{B}'$ with $\gcd(M, M'p) = 1$, $4p < (1-\Delta)M$, $2p \leqslant (1-\alpha)M' < M$; residues $\boldsymbol{x}_{\mathcal{B}\cup\mathcal{B}'}$ of $x < 4p^2$.
Ensure: residues $(\boldsymbol{s}_{\mathcal{B}}, \boldsymbol{s}_{\mathcal{B}'})$ of $s < 2p$, $s \equiv xM^{-1} \bmod p$.
1: $\boldsymbol{q}_{\mathcal{B}} \leftarrow \left|-xp^{-1}\right|_M$ ▷ in concurrent Rowers for $\mathcal{B}$
2: $\widehat{\boldsymbol{q}}_{\mathcal{B}'} \leftarrow \mathtt{Bex}(\mathcal{B}, \mathcal{B}', \boldsymbol{q}_{\mathcal{B}})$ ▷ Fig. 2 in [11] with $\alpha = 0$
3: $\boldsymbol{t}_{\mathcal{B}'} \leftarrow \left|x + \widehat{q}p\right|_{M'}$ ▷ in concurrent Rowers for $\mathcal{B}'$
4: $\boldsymbol{s}_{\mathcal{B}'} \leftarrow \left|tM^{-1}\right|_{M'}$ ▷ in concurrent Rowers for $\mathcal{B}'$
5: $\boldsymbol{s}_{\mathcal{B}} \leftarrow \mathtt{Bex}_\alpha(\mathcal{B}', \mathcal{B}, \boldsymbol{s}_{\mathcal{B}'})$ ▷ Fig. 2 in [11]
6: **return** $\boldsymbol{s}_{\mathcal{B}\cup\mathcal{B}'}$ ▷ $s \equiv \left|xM^{-1}\right|_p$, $0 \leqslant s < 2p$

D. Redundant RNS and fault detection capability

Within a context of fault attack, the crucial point is the independance of Rowers. A fault remains inside a same Rower as long as no base conversion occurs.

Definition II.1. *Given $\mathcal{B} = \{m_1, \ldots, m_n\}$ and $t \in [1, n]$, a t-fault on $\boldsymbol{x}_{\mathcal{B}}$ is a set $\mathcal{I}_t \subseteq [1, n]$ and t residues $e_i \in [1, m_i]$ for any $i \in \mathcal{I}_t$. If $\boldsymbol{x}_{\mathcal{B}}$ is supposed to be faulty, it is denoted $\overline{\boldsymbol{x}}_{\mathcal{B}}$. Moreover, $M_{\mathcal{I}_t}$ denotes $\prod_{j\in[1,n]\setminus\mathcal{I}_t} m_j$.*

A redundant RNS with a k-fault detection capability is composed by a main base $\mathcal{B} = \{m_1, \ldots, m_n\}$ and a coprime redundant base $\mathcal{B}_R = \{m_{R,1}, \ldots, m_{R,k}\}$ with $k \leqslant n$. The principle of fault detection in redundant RNS is based on a "consistency check" [17]. Some residues $(\boldsymbol{x}_{\mathcal{B}}, \boldsymbol{x}_{\mathcal{B}_R})$ are said to be consistent if they pass the following test:

$$\boldsymbol{x}_{\mathcal{B}_R} == \varphi_{\mathcal{B}_R} \circ \varphi_{\mathcal{B}}^{-1}\left(\boldsymbol{x}_{\mathcal{B}}\right). \tag{5}$$

The function $\varphi_{\mathcal{B}_R} \circ \varphi_{\mathcal{B}}^{-1}$ represents a complete conversion from $\mathcal{B}$ to $\mathcal{B}_R$. The dynamic range (i.e. used to represent the values modulo p) of this redundant RNS is by definition the one of $\mathcal{B}$, i.e. $[0, M)$. Hence, residues of any x in this range always passes the consistency check test.

In practice, adding a k-fault detection in a Cox-Rower architecture is simply achieved by adding k Rowers concurrently to the "regular" ones. The way to handle them along computations in $\mathbb{F}_p$ is the purpose of next section.

III. Multi-fault resistant RNS modular reduction

Without loss of generality, $\mathcal{B}$ and $\mathcal{B}'$ both own n moduli.

A. Preliminary remarks

Alg. 1 involves operations in $\mathcal{B}$ and $\mathcal{B}'$, a first conversion Bex possibly incomplete, and a second complete one Bex_α. So, attacks may occur in different parts of the process, affect different Rowers, and modify residues in both $\mathcal{B}$ and $\mathcal{B}'$ (the consequences of faults on the Cox are discussed in Section III-C). However, because of independancy of Rowers, an infected unit does not propagate a fault between conversions. In Fig. 1, we consider such types of fault: types 1 and 3 in $\mathcal{B}$, type 2 in $\mathcal{B}'$.

Fig. 2 illustrates the way a faulty Rower can propagate its error to other units during a conversion. We can notice that an error of type 1 will affect all the residue of the second base. The columns of the second base represent the evaluation of the summation in (1).

The goal is to find an easy way to detect if some faults occur during the reduction process. We notice that the 2nd conversion $\mathtt{Bex}_\alpha$ is complete, and can be used as is to perform a consistency check, as described in Thm. III.1.

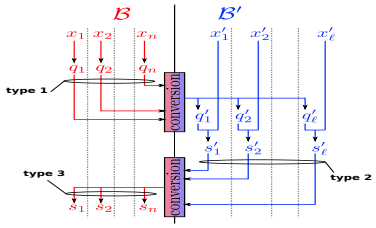

Fig. 1. Types of faults during the modular reduction

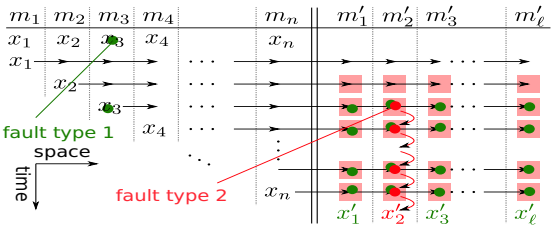

Fig. 2. Types of faults inside the conversion box

Theorem III.1. *Let $\mathcal{B}$ and $\mathcal{B}_R = \{m_{R,1}, \ldots, m_{R,k}\}$ ($k \leqslant n$) be a redundant RNS. For any $d \in [1,k]$, any $x \in [0, (1-\alpha)\,M)$ and any d-fault affecting $(\boldsymbol{x}_{\mathcal{B}}, \boldsymbol{x}_{\mathcal{B}_R})$, the consistency check $\overline{\boldsymbol{x}}_{\mathcal{B}_R} == \mathtt{Bex}_\alpha(\mathcal{B}, \mathcal{B}_R, \overline{\boldsymbol{x}}_{\mathcal{B}})$ always fails if for all $m_{R,z} \in \mathcal{B}_R$, $m_{R,z} > \max_{m\in\mathcal{B}} m$.*

Thus, if we consider Alg. 1, multiple faults in $\mathcal{B}'$ (i.e. type 2) can be detected by using the second conversion for checking, because $\mathtt{Bex}_\alpha$ is a complete reduction.

Thwarting type 3 faults: In the next section, we show how to deal with type 1 faults, possibly associated to type 2 faults and to faults occuring during a conversion. We won't mention faults in $\mathcal{B}$ after the second base conversion, i.e. type 3 faults. Either they can be reduced to type 1 (and then will be detected) when the output of the reduction is re-used as an input (e.g. as in a modular exponentiation), or residues in $\mathcal{B}'$ (whose integrity is checked by the detection process described afterwards) are sufficient to reconstruct the output of reduction in binary representation.

B. Multi-fault injected during a modular reduction

In this part and as a first step, we consider "theoretical" faults, which affect values modulo some m_i's. In other words, we do not consider yet faults at the binary representation level. For example, if x_i is an r-bit word, a faulty residue $\overline{x}_i$ is assumed to belong to $[0, m_i)$. The other case ($\overline{x}_i \in [m_i, 2^r)$) is tackled in next section. To simplify the study in the context of a Cox-Rower architecture, faults on the Cox are not considered for the moment. This issue will be considered later.

The detection capability k determines the size of $\mathcal{B}_R$. The k-fault resistant modular reduction is described by Alg. 2. Steps 1 to 5 are the same than for Alg. 1 but they additionally take into account $\mathcal{B}_R$. Next, steps 6 to 9 implement the consistency check. A multi-fault during a base conversion can be modeled by a multi-fault infecting pre- and/or post-conversion residues. Thus, we can restrain to the faults affecting $\boldsymbol{q}_{\mathcal{B}}$ due to perturbations at steps 1 and 2, and those affecting $\boldsymbol{s}_{\mathcal{B}'}$ and $\boldsymbol{s}_{\mathcal{B}_R}$ after faults at steps 2, 3, 4 and 5.

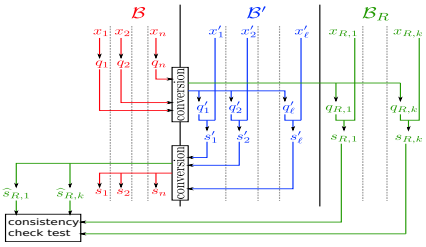

Fig. 3. Scheme of the consistency test.

The structure of a multi-fault injected during the execution of Alg. 2 is formally described in the following definition.

Definition III.2. *Let $d \leqslant k$. A d-fault appearing during a redundant RNS modular reduction is described by $d = u + v + w$ index numbers $\mathcal{I}_u, \mathcal{J}_v \subset [1,n]$ and $\mathcal{Z}_w \subset [1,k]$ with $|\mathcal{I}_u| = u$, $|\mathcal{J}_v| = v$, $|\mathcal{Z}_w| = w$, and d integers: $e_i \in \mathbb{Z}/m_i\mathbb{Z}$ for all $i \in \mathcal{I}_u$, $e'_j \in \mathbb{Z}/m'_j\mathbb{Z}$ for all $j \in \mathcal{J}_v$, and $e_{R,z} \in \mathbb{Z}/m_{R,z}\mathbb{Z}$ for all $z \in \mathcal{Z}_w$. This fault modifies the residues of $\boldsymbol{q}_{\mathcal{B}}$, $\boldsymbol{s}_{\mathcal{B}'}$ and $\boldsymbol{s}_{\mathcal{B}_R}$ computed during the execution of Alg. 2 as follows: $\forall (i,j,z) \in \mathcal{I}_u \times \mathcal{J}_v \times \mathcal{Z}_w$,*

$$\overline{q}_i = |q_i + e_i|_{m_i}, \overline{s}'_j = \left|s'_j + e'_j\right|_{m'_j}, \overline{s}_{R,z} = |s_{R,z} + e_{R,z}|_{m_{R,z}}. \tag{6}$$

Algorithm 2 $\mathtt{R\_RNSMR}(x, p, \mathcal{B}, \mathcal{B}', \mathcal{B}_R)$

Require: Same hyp. than Alg. 1, $\mathcal{B}_R = \{m_{R,1}, \ldots, m_{R,k}\}$, $\gcd(M_R, pMM') = 1$; residues $\boldsymbol{x}_{\mathcal{B}\cup\mathcal{B}'\cup\mathcal{B}_R}$ of $x < 4p^2$.
Ensure: res. $(\boldsymbol{s}_{\mathcal{B}}, \boldsymbol{s}_{\mathcal{B}'}, \boldsymbol{s}_{\mathcal{B}_R})$ of $s < 2p$, $s \equiv xM^{-1} \bmod p$.
1: $\boldsymbol{q}_{\mathcal{B}} \leftarrow \left|-xp^{-1}\right|_M$ ▷ // in $\mathcal{B}$
2: $(\widehat{\boldsymbol{q}}_{\mathcal{B}'}, \widehat{\boldsymbol{q}}_{\mathcal{B}_R}) \leftarrow \mathtt{Bex}(\mathcal{B}, \mathcal{B}' \cup \mathcal{B}_R, \boldsymbol{q}_{\mathcal{B}})$
3: $(\boldsymbol{t}_{\mathcal{B}'}, \boldsymbol{t}_{\mathcal{B}_R}) \leftarrow (x + \widehat{q}p) \bmod M', M_R$ ▷ // in $\mathcal{B}' \cup \mathcal{B}_R$
4: $(\boldsymbol{s}_{\mathcal{B}'}, \boldsymbol{s}_{\mathcal{B}_R}) \leftarrow (tM^{-1}) \bmod M', M_R$ ▷ // in $\mathcal{B}' \cup \mathcal{B}_R$
5: $(\boldsymbol{s}_{\mathcal{B}}, \widehat{\boldsymbol{s}}_{\mathcal{B}_R}) \leftarrow \mathtt{Bex}_\alpha(\mathcal{B}', \mathcal{B} \cup \mathcal{B}_R, \boldsymbol{s}_{\mathcal{B}'})$
6: **if** $\boldsymbol{s}_{\mathcal{B}_R} == \widehat{\boldsymbol{s}}_{\mathcal{B}_R}$ **then** ▷ Consistency check.
7: **return** $(\boldsymbol{s}_{\mathcal{B}}, \boldsymbol{s}_{\mathcal{B}'}, \boldsymbol{s}_{\mathcal{B}_R})$
8: **else**
9: **return** ''Corrupted data''
10: **end if**

Fig. 3 depicts the principle of an architecture implementing Alg. 2. The next theorem ensures the correctness of Alg. 2.

Theorem III.3. *Let's consider Alg. 2. If $m_R > m$ for all $(m_R, m) \in \mathcal{B}_R \times (\mathcal{B} \cup \mathcal{B}')$, then any d-fault as in Def. III.2 is detected by the consistency check at steps 6-9.*

Proof: Let's consider a $(u+v+w)$-fault as defined by (6). The w faults in redundant Rowers do not play any role. Indeed, the detection capability k ensures that at least $u+v$ redundant channels are not corrupted between two consistency checks. Then, if $u = v = 0$, the faults only affect redundant residues, and the context of single-fault in [2] is enough to ensure the detection (just consider the system with only one

faulty redundant channel). From now on we denote $M_{R,\mathcal{Z}_w}$ the product of $k-w$ safe redundant channels. The proof is mainly based on the fact that it suffices to study effect of the $(u+v)$ faults on the integer $t = x + \widehat{q}p$ at line 3.

First, type 1 faults modify q at line 1. We then denote it $\overline{q} \in [0, M)$. After the first conversion, we denote $\tilde{t}^{(1)} \stackrel{def}{=} x + \widehat{\overline{q}}p$ the corresponding corrupted t ($\widehat{\overline{q}}$ denotes the output of $\texttt{Bex}(\overline{q})$). In particular, $\tilde{t}^{(1)}$ is known in each uncorrupted redundant channel, i.e. modulo $M_{R,\mathcal{Z}_w}$.

Second, type 2 faults modify $\tilde{t}^{(1)}$ in $\mathcal{B}'$ before the second conversion. We denote $\widehat{\overline{\boldsymbol{s}}}_{\mathcal{B}'} \stackrel{def}{=} \texttt{Bex}_\alpha(\overline{\boldsymbol{s}}_{\mathcal{B}'})$ the value sent by the conversion to $\mathcal{B}$ to recover the ouput in the full base, and to $\mathcal{B}_R$ for the consistency check. So the associated "t" is defined by $\tilde{t}^{(2)} = M\widehat{\overline{\boldsymbol{s}}}_{\mathcal{B}'}$.

At this point, we know that, in particular, the consistency check will detect the multi-fault if $(\tilde{t}^{(1)}M^{-1} - \widehat{\overline{\boldsymbol{s}}}_{\mathcal{B}'}) \not\equiv 0 \bmod M_{R,\mathcal{Z}_w}$. Due to coprimality of M and M_R, this inequality holds iff $(\tilde{t}^{(1)} - \tilde{t}^{(2)}) \not\equiv 0 \bmod M_{R,\mathcal{Z}_w}$. Thus, we will show that this inequality holds.

Basically, the strategy is to prove that $|\tilde{t}^{(1)} - \tilde{t}^{(2)}|$ is an integer fully representable in $\mathcal{B} \cup \mathcal{B}'$ (i.e. it is $< MM'$), and has non zero residues in, and only in, channels of $\mathcal{B}$ affected by the u type 1 faults (and index-numbered by $\mathcal{I}_u$), and channels of $\mathcal{B}'$ affected by the v type 2 faults (and index-numbered by $\mathcal{J}_v$). In other words, this means that we need to prove that $|\tilde{t}^{(2)} - \tilde{t}^{(1)}| = e_t M_{\mathcal{I}_u} M'_{\mathcal{J}_v}$, with $0 < e_t < \prod_{i\in\mathcal{I}_u, j\in\mathcal{J}_v} m_i m'_j$. Hence, the hypothesis that $M_{R,\mathcal{Z}_w} > \prod_{i\in\mathcal{I}_u, j\in\mathcal{J}_v} m_i m'_j$ will ensure $e_t \neq 0 \bmod M_{R,\mathcal{Z}_w}$.

A first step is to establish that:

(i) $\tilde{t}^{(1)} \in [0, (1-\alpha)MM')$.

Point (i) is due to the fact that, despite the type 1 faults on q, giving $\overline{q} \in [0, M)$, the first conversion still ensures that $\widehat{\overline{q}} \in [0, (1+\Delta)M)$. Indeed, from hyp. on M and M' in Alg. 2, it comes $x + \widehat{\overline{q}}p < 4p^2 + (1+\Delta)Mp < 2Mp \leqslant (1-\alpha)MM'$.

Now, we highlight crucial properties about $\overline{\boldsymbol{s}}_{\mathcal{B}'}$ and $\widehat{\overline{\boldsymbol{s}}}_{\mathcal{B}'}$:

(ii) $|\overline{\boldsymbol{s}}_{\mathcal{B}'} - |\tilde{t}^{(1)}M^{-1}|_{M'}| = e_s M'_{\mathcal{J}_v}, 0 < e_s < \prod_{j\in\mathcal{J}_v} m'_j$;

(iii) $\widehat{\overline{\boldsymbol{s}}}_{\mathcal{B}'} \in \{\overline{\boldsymbol{s}}_{\mathcal{B}'}, \overline{\boldsymbol{s}}_{\mathcal{B}'} - M'\}$, so $\widehat{\overline{\boldsymbol{s}}}_{\mathcal{B}'} \in [-\alpha M', M')$.

The point (ii) is just a consequence of definition of $\overline{\boldsymbol{s}}_{\mathcal{B}'}$ which is $|\tilde{t}^{(1)}M^{-1}|_{M'}$ modified by the type 2 faults. Moreover, because of the faults, $\overline{\boldsymbol{s}}_{\mathcal{B}'}$ may not belong to $[0, (1-\alpha)M')$ anymore. This implies that $\texttt{Bex}_\alpha$ may provide the incomplete reduced integer $\overline{\boldsymbol{s}}_{\mathcal{B}'} - M'$. However, it can only happen when $\overline{\boldsymbol{s}}_{\mathcal{B}'} \in [(1-\alpha)M', M')$. This justifies point (iii).

We continue with following properties about $\tilde{t}^{(2)} = M\widehat{\overline{\boldsymbol{s}}}_{\mathcal{B}'}$:

(iv) $0 < |\tilde{t}^{(2)} - \tilde{t}^{(1)}| < MM'$;

(v) $|\tilde{t}^{(2)} - \tilde{t}^{(1)}| = e_t M_{\mathcal{I}_u} M'_{\mathcal{J}_v}, 0 < e_t < \prod_{i\in\mathcal{I}_u, j\in\mathcal{J}_v} m_i m'_j$.

Point (iv) is directly deduced from (i) and (iii). To prove (v), we first notice that, viewed in $\mathcal{B}\cup\mathcal{B}'$, the two integers $\tilde{t}^{(1)}$ and $M|\tilde{t}^{(1)}M^{-1}|_{M'}$ only differ in channels of $\mathcal{B}$ index-numbered by $\mathcal{I}_u$. So, because of the definition of $\tilde{t}^{(2)} = M\widehat{\overline{\boldsymbol{s}}}_{\mathcal{B}'} = M \times \texttt{Bex}_\alpha(|\tilde{t}^{(1)}M^{-1}|_{M'})$ and of (ii), we deduce that $\tilde{t}^{(1)}$ and $\tilde{t}^{(2)}$ only differ by a non zero multiple of $M_{\mathcal{I}_u} M'_{\mathcal{J}_v}$. And because of (iv), this multiple verifies (v).

To end the proof, it just remains to notice that, by hypothesis on the number of faults and on the size of redundant moduli, we have $M_{R,\mathcal{Z}_w} > \prod_{i\in\mathcal{I}_u, j\in\mathcal{J}_v} m_i m'_j$. So, this and (v) imply that $|e_t|_{M_{R,\mathcal{Z}_w}} \neq 0$, or in other words that $(\tilde{t}^{(1)} - \tilde{t}^{(2)}) \not\equiv 0 \bmod M_{R,\mathcal{Z}_w}$: the consistency check detects the multi-fault, at least in all the uncorrupted channels of $\mathcal{B}_R$.

■

C. Treating faults on the Cox

A fault on a Rower corresponds to a type 1 or 2 in Fig. 1. We have to address the effects of the faults made on the Cox part (not described in Fig. 1). The Cox evaluates the value $\kappa_{\mathcal{B}}(\boldsymbol{x}_{\mathcal{B}})$ in (1). If the Cox is common to all the Rowers then it will affect all the output by the same way. In other words, that means that the reduction could be not complete. For the first conversion in Alg. 1 if we are under the conditions given in [1] then the faults will not affect the result modulo p. Here we assume that, among the k possible faults injected during a Montgomery reduction, τ of them can affect the Cox. In the proof of Thm III.3, a key point is that $\tilde{t}^{(1)}$ remains in $[0, (1-\alpha)MM')$. Thus, to counteract possible negative multiples of M due to faults on the Cox, one should automatically add $(k-1)M$ to the value $\widehat{q}$ obtained in $\mathcal{B}'\cup\mathcal{B}_R$. Consequenlty, one should also increase the size of M' to keep $\tilde{t}^{(1)} + (k+1)p < (1-\alpha)MM'$.

For the second base conversion, it may seem more complex because we expect a complete reduction modulo M' to ensure the validity of the checking. In fact, as we check the consistency between the result in $\mathcal{B}'$ and $\mathcal{B}_R$, if multiples of M' appear because of consecutive faults on the Cox (one fault would at most add a single multiple of M'), it would affect the result, and could avoid a multiple fault to be detected. Actually, in such a case, the check would verify if a quantity of the form $e_{\mathcal{I}_u,\mathcal{J}_v} M_{\mathcal{I}_u} M'_{\mathcal{J}_v} + \tau MM'$ is zero, or not, modulo M_R ($\tau M'$ coming from the faulty Cox, with τ lower than the number consecutive faults affecting the Cox, and M coming from the multiplication by M in the final reasoning in the proof of Thm III.3). But by hypothesis on the detection capability, there are at least $u+v+\tau$ safe redundant moduli. Thus, to detect such sets of faults, it suffices to have the product $M_{R,u+v+\tau}$ greater than $(\tau+1)\prod_{i\in\mathcal{I}_u} m_i \prod_{j\in\mathcal{J}_v} m'_j$. For instance, if the size of redundant moduli is constrained by some architecure considerations, one could simply add an extra redundant modulus verifying $m_{R,k+1} > k > \tau$.

Finally, such kind of faults on a single Cox shared between all Rowers can easily be defeated, basically by adding a small extra modulus to $\mathcal{B}_R$ for instance.

However, due to small area cost of a Cox, it may be more interesting to integrate a Cox inside each Rower. In the next parts, we select this variant.

D. Integrating binary representation within the fault model

The previous fault model takes into account the appearance of multiple faults in time and space in a Cox-Rower architecture, but it does not fit with the way that data are represented into hardware. A residue was always considered like an integer in $[0, m)$. But such value needs to be handled through a binary representation. During a conversion, each rower computes and

stores the $\xi_{i,q}$ or $\xi'_{j,s}$ in r-bit registers before sending them towards other rowers. In practice, the size of moduli is chosen such that $2^{r-1} < m < 2^r$. For further discussion, we consider that any modulus verifies $2^r - m < 2^c$.

Thereafter, we analyse the consequences of this more precise model, as it is done in [2]. Without loss of generality, a fault on a residue q_i (or similarly s'_j) can be modeled by a fault modifying the corresponding $\xi_{i,q}$ (or $\xi'_{j,s}$). As an r-bit value, the faulty residue $\overline{\xi}_{i,q}$ can be anywhere in $[0, 2^r)$.

Obviously, new circumstances appear when some $\overline{\xi}_{i,g}$'s verify $m_i \leqslant \overline{\xi}_{i,q} < 2^r$. In this case, we write $\overline{\xi}_{i,q} = m_i + \tilde{\xi}_{i,q}$ with $\tilde{\xi}_{i,q} \in [0, m_i)$. $\tilde{\xi}_{i,q}$ may represent another fault, but the point is that the affected underlying residue q_i is still the same. Such overflows have two consequences in the subsequent base conversion. First, a positive multiple of M can come from the sum $\sum_{i=1}^n \overline{\xi}_{i,q} M_i$. Second, a negative multiple of M can be due to the sum $\sum_{i=1}^n \frac{\mathrm{trunc}_h(\overline{\xi}_{i,q})}{2^h}$ when computing $\tilde{\kappa}_{\mathcal{B}}(\overline{\boldsymbol{q}})_{\mathcal{B}}$.

The following theorem provides sufficient conditions which ensure that positive and negative multiples of M for the first base conversion (and M' for the second one) almost cancel each other out. This is essentially what is proven in the proof of Thm. III.4.

The main consequence of the theorem is that the redundant moduli are not required to be greater than 2^r to ensure the detection of these specific overflowing multi-faults.

Theorem III.4. *Assume that any modulus $m \in \mathcal{B} \cup \mathcal{B}'$ satisfies $2^{r-1} < m < 2^r$. Let $\varepsilon = \min\{t \in \mathbb{Z} \mid \forall m \in \mathcal{B} \cup \mathcal{B}', 2^r - m < 2^t\}$. Let's consider Alg. 3 with following hypothesis:*

(i) $h \leqslant r - \varepsilon$
(ii) $\alpha = \frac{n+k}{2^{h-1}}$ *and* $\alpha + \frac{k}{2^h} < 1$
(iii) $M > \frac{9p}{1-\alpha}$
(iv) $M' > \frac{3p}{1-\alpha-\frac{k}{2^h}} (> \frac{3p}{1-\alpha})$.

If $m_R > m$ for all $(m_R, m) \in \mathcal{B}_R \times (\mathcal{B} \cup \mathcal{B}')$, then any hardware d-multifault is detected for any $d \in [1, k]$.

Algorithm 3 R_RNSMR_HW$(x, p, \mathcal{B}, \mathcal{B}', \mathcal{B}_R)$ (adapted to hardware multi-fault model)

Require: Hyp. of Thm III.4 for $\mathcal{B}$, $\mathcal{B}'$ and $\mathcal{B}_R$ with $\gcd(M_R, pMM') = 1$; $\boldsymbol{x}_{\mathcal{B}\cup\mathcal{B}'\cup\mathcal{B}_R}$ with $x < 9p^2$.
Ensure: $(\boldsymbol{s}_{\mathcal{B}}, \boldsymbol{s}_{\mathcal{B}'}, \boldsymbol{s}_{\mathcal{B}_R})$, $s < 3p$, $s \equiv xM^{-1} \bmod p$.
1: $\boldsymbol{q}_{\mathcal{B}} \leftarrow \left|-xp^{-1}\right|_M$ ▷ // in $\mathcal{B}$
2: $(\widehat{\boldsymbol{q}}_{\mathcal{B}'}, \widehat{\boldsymbol{q}}_{\mathcal{B}_R}) \leftarrow \mathtt{Bex}(\mathcal{B}, \mathcal{B}' \cup \mathcal{B}_R, \boldsymbol{q}_{\mathcal{B}})$
3: $(\boldsymbol{t}_{\mathcal{B}'}, \boldsymbol{t}_{\mathcal{B}_R}) \leftarrow (x + (\widehat{q} + M)p) \bmod M', M_R$ in $\mathcal{B}' \cup \mathcal{B}_R$
4: $(\boldsymbol{s}_{\mathcal{B}'}, \boldsymbol{s}_{\mathcal{B}_R}) \leftarrow (tM^{-1}) \bmod M', M_R$ ▷ in $\mathcal{B}' \cup \mathcal{B}_R$
5: $(\boldsymbol{s}_{\mathcal{B}}, \widehat{\boldsymbol{s}}_{\mathcal{B}_R}) \leftarrow \mathtt{Bex}_\alpha(\mathcal{B}', \mathcal{B} \cup \mathcal{B}_R, \boldsymbol{s}_{\mathcal{B}'})$
6: **if** $\boldsymbol{s}_{\mathcal{B}_R} == \widehat{\boldsymbol{s}}_{\mathcal{B}_R}$ **then** ▷ Consistency check.
7: **return** $(\boldsymbol{s}_{\mathcal{B}}, \boldsymbol{s}_{\mathcal{B}'}, \boldsymbol{s}_{\mathcal{B}_R})$
8: **else**
9: **return** ''Corrupted data''
10: **end if**

Preliminary remarks: First, the offset α (ii) which corrects the value output by the Cox during the second conversion fits in the h-bit register of the Cox unit. Moreover, we claim that $\Delta = \frac{n}{2^{h-1}}$ is an upper bound of the error appearing in the sum (4) computed by the Cox, when the $\xi_{i,q}$ (or $\xi'_{i,s}$) are in $[0, m_i)$. Indeed, because $\mathrm{trunc}_h(\xi) = \frac{\xi - |\xi|_{2^{r-h}}}{2^{r-h}}$ and $m_i = 2^r - c_i$ with $c_i < 2^\varepsilon$, then we have:

$$\frac{\xi_i}{m_i} - \frac{\mathrm{trunc}_h(\xi_i)}{2^h} \leqslant \frac{2^r\xi_i - m_i\xi_i + m_i|\xi_i|_{2^{r-h}}}{2^r m_i} < \frac{2^\varepsilon}{2^r} + \frac{2^{r-h}}{2^r}.$$

Since $h \leqslant r - \varepsilon$ (i), it comes $\frac{n}{2^{r-\varepsilon}} + \frac{n}{2^h} \leqslant \frac{n}{2^{h-1}} = \Delta$. So Δ is greater than the full error for both Bex and $\mathtt{Bex}_\alpha$. And the value of α (ii) enables to correct this error (i.e. $\alpha \geqslant \Delta$).

The following discussion aims to show that, under conditions of Thm. III.4, the new context is actually identical to the one about "theoretical" faults. To do so, we show that the key points in proof of Thm. III.3 are still relevant. Precisely, we prove that $|\tilde{t}^{(1)} - \tilde{t}^{(2)}| < MM'$ and it owns non zero residues only in u channels of $\mathcal{B}$ and v of $\mathcal{B}'$. We also prove that the new conditions still imply a complete second base conversion when no fault occurs.

a) About first conversion: In the following, $\overline{q}$ may denote as well a correct q as q affected by u type 1 faults. The first conversion is not required to provide a full reduction modulo M. The crucial point is to guarantee that $\tilde{t}^{(1)} \in [0, (1-\alpha)MM')$. So, we need the converted value $\widehat{\overline{q}}$ to be positive, and M' to be such that $\tilde{t}^{(1)} = x + \widehat{\overline{q}}p < (1-\alpha)MM'$.

On the one side, the sum $\sum_{i=1}^n \overline{\xi}_{i,q} M_i$ can contain an extra term ℓM with $\ell \leqslant k$. ℓ is the exactly the number of erroneous $\overline{\xi}_{i,q}$ which are overflowing their respective m_i. We denote by $(\xi_{i,q})_{i\in[1,n]}$ the set of all the $\overline{\xi}_{i,q}$, but where $\tilde{\xi}_{i,q} = \overline{\xi}_{i,q} - m_i$ for each $\overline{\xi}_{i,q}$ overflowing m_i. In particular, the new coefficients $\tilde{\xi}_{i,q}$ are those of q affected by a theoretical multi-fault, and represent an integer denoted $\overline{q}$ in $[0, M)$. Consequently, $\sum_{i=1}^n \overline{\xi}_{i,q} M_i = \sum_{i=1}^n \tilde{\xi}_{i,q} M_i + \ell M$. So, the conversion outputs $\overline{q}$ plus a multiple of M. On the other side, we check if the value (3) computed by the Cox can cancel ℓM. This computation involves the terms $\mathrm{trunc}_h(\overline{\xi}_{i,q})$. In particular, we have ℓ such following quantities (where $\delta_i \in \{0, 1\}$): $\mathrm{trunc}_h(m_i + \tilde{\xi}_{i,q}) = \mathrm{trunc}_h(m_i) + \mathrm{trunc}_h(\tilde{\xi}_{i,q}) + \delta_i$.

Under conditions of Thm. III.4, the ℓ terms $\mathrm{trunc}_h(m_i) + \delta_i$ cancel at least $(\ell - 1)M$. Indeed, we can write (for brievity's sakeness, details are omitted), with $\Delta = \frac{n}{2^{h-1}}$:

$$\begin{cases} \sum_{i=1}^n \frac{\mathrm{trunc}_h(\overline{\xi}_{i,q})}{2^h} \leqslant \kappa_{\mathcal{B}}(\overline{\boldsymbol{q}}_{\mathcal{B}}) + \frac{\overline{q}}{M} + \ell + \frac{\ell}{2^h} \\ \sum_{i=1}^n \frac{\mathrm{trunc}_h(\overline{\xi}_{i,q})}{2^h} \geqslant \kappa_{\mathcal{B}}(\overline{\boldsymbol{q}}_{\mathcal{B}}) + \frac{\overline{q}}{M} + \ell - \Delta - \frac{\ell}{2^{r-\varepsilon}} - \frac{\ell}{2^h} \end{cases} \quad (7)$$

Moreover, we notice that (i) implies $\Delta + \frac{\ell}{2^{r-\varepsilon}} + \frac{\ell}{2^h} \leqslant \Delta + \frac{k}{2^{h-1}} = \alpha$. Since $\alpha < 1$ (ii), the Cox outputs:

$$\left\lfloor \sum_{i=1}^n \frac{\mathrm{trunc}_h(\overline{\xi}_{i,q})}{2^h} \right\rfloor = \kappa_{\mathcal{B}}(\overline{\boldsymbol{q}}_{\mathcal{B}}) + \ell + \beta, \text{ with } \beta \in \{-1, 0, 1\}.$$

Finally, the 1st conversion outputs $\widehat{\overline{q}} = \overline{q} - \beta M$. As previously explained, although an incomplete reduction is allowed, $\widehat{\overline{q}}$ has to be non negative. To prevent the case $\beta = 1$, the value returned by the Cox must be decreased by 1. This justifies the term $\widehat{q} + M$ at line 3 of Alg. 3, and the new definition of $\tilde{t}^{(1)} \overset{def}{=} x + (\widehat{\overline{q}} + M)p$. Consequently, $\widehat{\overline{q}} + M \in \{\overline{q}, \overline{q} + M, \overline{q} +$

$2M\}$, and we can deduce from (7) that $\widehat{\overline{q}} = \overline{q} + 2M$ only if $\overline{q} < (1 + \Delta + \frac{\ell}{2^{r-\varepsilon}} + \frac{\ell}{2^h})M \leqslant (1+\alpha)M$. Hence,

$$0 \leqslant \widehat{\overline{q}} + M < (2+\alpha)M. \tag{8}$$

Besides, cond. (iii) about M states that $x < 9p^2 < (1-\alpha)Mp$. This, together with (8) and condition (iv) about M', implies:

$$0 \leqslant \tilde{t}^{(1)} = x + (\widehat{\overline{q}} + M)p < 3Mp < (1 - \alpha - \tfrac{k}{2^h})MM'. \tag{9}$$

b) About second conversion: The analysis is similar for conversion of $\overline{\boldsymbol{s}}_{\mathcal{B}'}$. We yet consider ℓ overflows. Then, the offset α (ii) enables to correct the error $\Delta + \frac{\ell}{2^{r-\varepsilon}} + \frac{\ell}{2^h} \leqslant \frac{n+k}{2^{h-1}} = \alpha$. Hence, the version of second inequality in (7) for this case is greater than $\kappa_{\mathcal{B}}(\overline{\boldsymbol{s}}_{\mathcal{B}'}) + \ell$. Since $\overline{s}$ denotes an integer in $[0, M')$, and since, by hypothesis, $\alpha + \frac{k}{2^h} < 1$ (ii), the first upper bound in (7) becomes, in this case:

$$\begin{aligned}\kappa_{\mathcal{B}}(\overline{\boldsymbol{s}}_{\mathcal{B}'}) + \tfrac{\overline{s}}{M'} + \ell + \alpha + \tfrac{\ell}{2^h} &\leqslant \kappa_{\mathcal{B}}(\overline{\boldsymbol{s}}_{\mathcal{B}'}) + \tfrac{\overline{s}}{M'} + \ell + \alpha + \tfrac{k}{2^h} \\ &< \kappa_{\mathcal{B}}(\overline{\boldsymbol{s}}_{\mathcal{B}'}) + \ell + 2.\end{aligned} \tag{10}$$

Finally, the Cox outputs either $\kappa_{\mathcal{B}}(\overline{\boldsymbol{s}}_{\mathcal{B}'}) + \ell$ or $\kappa_{\mathcal{B}}(\overline{\boldsymbol{s}}_{\mathcal{B}'}) + \ell + 1$. In particular, the conversion outputs $\overline{\boldsymbol{s}}_{\mathcal{B}'} - \delta M'$ with $\delta \in \{0, 1\}$. And, from (10), we deduce that $\delta = 1$ only if $\overline{\boldsymbol{s}}_{\mathcal{B}'} \geqslant (1 - \alpha - \frac{k}{2^h})M'$. Consequently, we obtain the following bounds:

$$-(\alpha + \tfrac{k}{2^h})MM' \leqslant \mathtt{Bex}_{\alpha}(\overline{\boldsymbol{s}}_{\mathcal{B}'}) \times M < MM'. \tag{11}$$

By gathering (9) and (11), and the definition $\tilde{t}^{(2)} = \mathtt{Bex}_{\alpha}(\overline{\boldsymbol{s}}_{\mathcal{B}'})M$, we obtain the key point $\left|\tilde{t}^{(1)} - \tilde{t}^{(2)}\right| < MM'$. The second key point in proof of Thm III.3, i.e. $|\tilde{t}^{(1)} - \tilde{t}^{(2)}|$ owning u non zero residues in $\mathcal{B}$ and v in $\mathcal{B}'$, is obviously verified. Indeed, the only difference with Thm III.3 is due to a possible presence of multiples of MM'. Thus, due to faulty residues index numbered by $\mathcal{I}$ on q in $\mathcal{B}$ and $\mathcal{J}$ on s in $\mathcal{B}'$, we have yet that $\left|\tilde{t}^{(1)} - \tilde{t}^{(2)}\right| = e_t M_{\mathcal{I}} M'_{\mathcal{J}}$ with $0 < e_t < \prod_{i\in\mathcal{I}, j\in\mathcal{J}} m_i m'_j$. This guarantees the detection.

To end the proof, when no faults in $\mathcal{B}$ and $\mathcal{B}'$ occur (except maybe overflows $\overline{\xi}_{i,q} = m_i + \xi_{i,q}$ and $\overline{\xi}'_{j,s} = m'_j + \xi'_{j,s}$), then $s = \frac{t}{M} < 3p < (1 - \alpha - \frac{k}{2^h})M' < (1-\alpha)M'$. It means that the offset α guarantees an exact conversion: $\mathtt{Bex}_{\alpha}(\boldsymbol{s}_{\mathcal{B}'}) = s$. And the output in $\mathcal{B} \cup \mathcal{B}' \cup \mathcal{B}_R$ is nothing but $s \in [0, 3p)$ with $s \equiv xyM^{-1} \bmod p$.

E. Example of parameters

To show that the conditions of Thm. III.4 are realistically reachable despite their apparent restrictiveness, we consider the parameters calibrated for 521-bit ECC with $n = 31$ and $r = 17$. For instance, we set the detection capability to $k = 6$.

The set $[2^{17} - 2^9, 2^{17}]$ contains 68 prime numbers (i.e. enough for $\mathcal{B}$, $\mathcal{B}'$ and $\mathcal{B}_R$), so $\varepsilon = 9$. Consequently, $h \leqslant 8$ and it has to be set such that $\alpha + \frac{k}{2^h} = \frac{2n+3k}{2^h} < 1$ is true. Setting $h = 7$ is then sufficient. These parameters are such that conditions (iii) and (iv) about M and M' are satisfied (we remind $\log_2(p) = 521$). Indeed, we have in this case that $\log_2(M), \log_2(M') > \log_2\left((2^{17} - 2^9)^n\right) \sim 526.8$ and $\log_2(\frac{9p}{1-\alpha}) \sim 525.4$, $\log_2(\frac{3p}{1-\alpha-\frac{k}{2^h}}) \sim 524$.

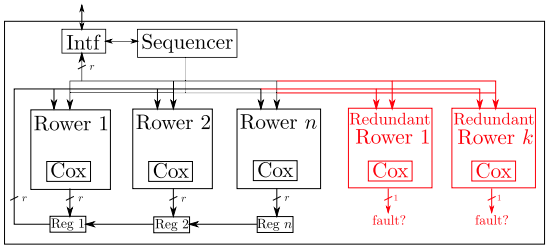

Fig. 4. Cox-Rower Architecture, with detection capability

IV. Multi-fault resilient hardware implementation

In this section, we provide some practical guidelines to re-use the redundant Rowers and about hardening parts of Cox-Rower architecture which could not be covered by the detection approach. In III-C, we discussed about the advantage of using one Cox per Rower. This is the choice we make.

A. Cox-Rower architecture

The Cox-Rower architecture has been intensively used for implementing RNS cryptoprocessors [3], [6], [9], [15]. It is composed of a Sequencer unit, a Cox unit and n Rower units. It is illustrated by the black parts in Fig. 4.

The Cox unit implements the trunc_h function so as to return the $\tilde{\kappa}$ coefficient (3). Each Rower unit is dedicated to computations like $(\sum_{i=1}^{n} a_i b_i) \bmod m$, where m is a modulus of $\mathcal{B}$ or $\mathcal{B}'$. This architecture shows a good trade-off between area and performance, and can take advantage of DSP blocks available in FPGA [3].

The n registers below rowers are in charge of sending ξ_i's coefficients (cf. (1)) from each rower to all others during a base conversion. To avoid burdensome architecture, the sending can be done through shiftings between these registers.

In the previous section, redundant residues were used to detect multi-fault attacks during Montgomery reduction computations and did not play any role during the Montgomery reduction. The modifications set to the Cox-Rower architecture to include the detection capability are shown in red in Fig. 4.

B. Redundant Rower unit

Although the architecture of a redundant Rower unit is quite identical to a regular one, the depth of the memories containing the precomputed values for the reduction level is divided by two. The extra features is that it implements the consistency check. This can be achieved at the accumulator/reduction level by comparing the result with zero.

Because redundant residues are not involved as part of pre-conversion base during a modular reduction, they are not subject to Cox-Rower conditions. In order to keep available the range of moduli under Cox-Rower conditions for the two bases $\mathcal{B}$ and $\mathcal{B}'$, we could choose to use $m_R \geqslant 2^r$ for all $m_R \in \mathcal{B}_R$. However, we have shown (Thm. III.4) that such condition is not necessary for detecting specific hardware overflowing faults. Thus redundant rowers could

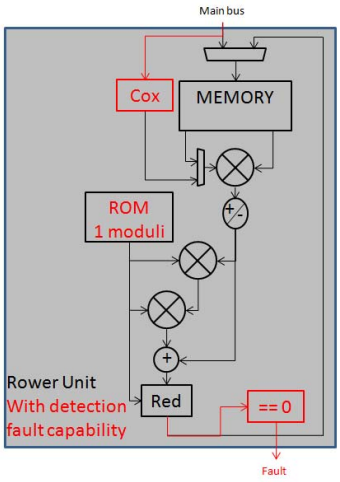

Fig. 5. Redundant Rower with detection capability.

contain strictly identical arithmetic and logic units (i.e. over r bits). Those modifications are highlighten in red in Fig. 5.

In previous works [9], [15], several RNS to binary radix conversions have been introduced so as to achieve a final conversion from RNS to binary radix. The principle is to recover coefficients of the result s in base 2^r, from the least significant digit to the most significant one, by using formula $\lfloor \frac{s}{2^r} \rfloor = \frac{s-|s|_{2^r}}{2^r}$. Consequently, we propose here to share the functionality of one redundant Rower using moduli $m = 2^r$ in order to detect one single fault and to compute RNS to binary radix conversion. Furthermore, this redundant Rower is obviously smaller than other ones because computing $\bmod 2^r$ is just truncating binary representations.

On FPGA, this solution is quite easy to implement as DSP blocks embed multiply and accumulate features and also a test value features dedicated to zero value detection and which can be used for consistency checks.

Last, because one standard rower is usually dedicated to two moduli, one of $\mathcal{B}$ and one of $\mathcal{B}'$, it is worth noticing that multi-fault model enables preventing hardware failures, which then could affect residues two by two. In this case, k is advantageously chosen being even.

C. Hardening Cox unit and registers

In III, it is shown that multi-fault attack can be detected when it affects computations done inside Rower units. Faults affecting Cox unit or Registers used during base conversions are not covered in the hypothesis of Alg. 3 and Thm. III.4. In order to prevent and to detect a fault in Cox unit, we suggest to harden it by using lockstep fault-tolerant methodology. Indeed, Cox is only composed by an $h + \lceil \log_2(n) \rceil$-bit adder/accumulator together with an $h + \lceil \log_2(n) \rceil$-bit register. Thus it is quite small compared to the Rower units. In previous example III-E, the size of the Cox unit implemented in FPGA would be only 13 LUTs and registers.

In the case of Register units dedicated to distribution of ξ_i's coefficients during a base conversion, the problem is that a permanent fault could corrupt much more than k of these coefficients because of shifting. Thus the fault-model would not fit anymore. For instance a fault which would permanently affect the 1st register could modify the set of whole ξ_i's passing through this register, and may be responsible for appearance of more than k faults.

To prevent this phenomenon, each ξ_i can be memorised in its associated rower before it is sent into output shift registers. When ξ_i is distributed to all rowers, the i^{th} rower can perform a comparison with respect to previously memorised value so as to check that the distribution step has been fair. The extra cost is then nr registers. The modified rower unit is presented in Fig. 6 using red color for modifications.

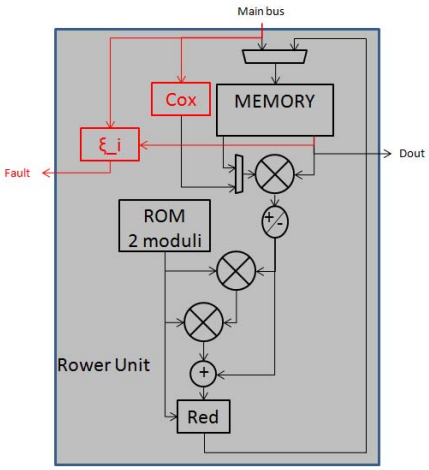

Fig. 6. Modified Rower.

D. Extra cost of detection capability

The detection procedure has the advantage to not increase execution time of modular reduction as soon as enough extra surface can be used to implement the whole detection process. Indeed, redundant residues are never involved in the emitting side during a base conversion. Furthermore, any computation concerning consistency check or protection of output registers can be done independently from main computation flow.

Thus, the only cost is in terms of area and power consumption. Theorem III.4 proves that redundant moduli just have to be greater than standard ones. Thus, a redundant rower owns the same structure than a classical one, plus a negligible area overhead due to the consistency check. For instance, Table I provides area of a Rower design from [3], which is useable for implementing a 521-bit ECC (as considered in [3]). Moreover, section IV-C has also emphasized that protecting Cox through lockstep fault-tolerant methodology remains reasonably cheap because of the simple and small structure of this unit.

To resume, a relevant estimation of area overhead is $k \times$ (Rower Area), with k the detection capability. Because rowers represent main part of architecture, a first estimation is given by $\sim +\frac{k}{n}\%$ of initial area, where n is the number of standard rowers. Table II highlights orders of magnitude

Rower type	Slices	Logic LUTs	FFs	BRAM18	DSP48E
Non-modified	67	170	182	2	4
Modified	75	197	214	2	4
Redundant	56	86	180	2	4

TABLE I
ROWER AREA FOR KINTEX-7 WITH $r = 17$.

of area overhead in the case of $n = 31$ standard rowers for implementing a 521-bit ECC [3]. Those values are also compared with P&R results for area overhead (given in slices) as well as power overhead @200MHz which are less than the first estimation. The reason for this gap is mainly due to the fact that a redundant rower is smaller than a modified rower.

To finish, we also notice that the total number of slices in Tab. II for $k = 0$ is a bit greater than in [3] (2565 slices for the same parameters). This is because we use a deeper pipeline within the present modified rowers.

Detection cap. k	0	2	4	6	8
Estimated overhead $+\frac{k}{n}\%$	0%	+6.5%	+12.9%	+19.3%	+25.8%
Slices	2864	2985	3120	3196	3204
overhead	0%	+4.2%	+8.9%	+11.5%	+11.9%
Power	691	730	779	801	847
overhead (in mW @200MHz)	0%	+5.6%	+12.7%	+15.9	+22.6%

TABLE II
AREA OVERHEAD DUE TO REDUNDANT ROWERS (FOR COX-ROWER ARCHITECTURE OWNING $n = 31$ STANDARD ROWERS).

a) Validation of the methodology of fault detection: We have validated our fault detection design by using hardware simulator tool. To inject fault into our design, we forced erroneous values (type 1 and 2 as described above) during the computation in the simulator. Transient faults as well as permanent faults were simulated in our modified rowers. The fault detection mechanism was always triggered when multi-fault simulation was running. We also compared the faulted design with its golden model (simulation without fault) to observe the differences in the results.

V. CONCLUSION

In [2], an RNS modular reduction algorithm defeating single-fault attacks was proposed. Since fault attacks can be devastating, it really matters to know if this approach is efficiently scalable so as to defeat multiple attacks, which could cause several RNS units being corrupted. Moreover, a multi-fault model also allows to model hardware failures. Indeed, in a Cox-Rower architecture, a single Rower is generally dedicated to, at least, one channel of $\mathcal{B}$ and one of $\mathcal{B}'$.

For that purpose, an RNS modular reduction algorithm supplied with multi-fault detection capability has been presented. It has been shown that the approach in [2] is extendable to any degree of detection as proven by Thm III.3 and Thm III.4. Beyond pure theoretical analysis, a hardware multi-fault model adapted to Cox-Rower architecture has also been considered. Despite some specific constraints due to binary representation of RNS data, it has been proven that the redundant moduli do not have to own any extra specific features in hardware context comparing to theoretical model. So, the detection process is flexible, since any rower can be dedicated either to a standard RNS channel, or to a redundant one without significant changes.

As for the single-fault detection, defeating multiple-attacks can be done without any time overhead. Furthermore, area overhead is mainly limited to the one of redundant rowers, and is roughly estimated to be $+\frac{k}{n}\%$ of initial area, with k being the detection capability, and n the size of both bases $\mathcal{B}$ and $\mathcal{B}'$. As a consequence, for a given k, the smaller the regular moduli, the smaller the area overhead.

REFERENCES

[1] J.-C. Bajard, L.-S. Didier, and P. Kornerup. Modular multiplication and base extensions in residue number systems. In *Computer Arithmetic, 2001. Proc. 15th IEEE Symposium on*, pp 59–65, 2001.

[2] J.-C. Bajard, J. Eynard, and F. Gandino. Fault Detection in RNS Montgomery Modular Multiplication. In *Computer Arithmetic (ARITH), 2013 21st IEEE Symposium on*, pp 119–126, April 2013.

[3] J.-C. Bajard and N. Merkiche. Double Level Montgomery Cox-Rower Architecture, New Bounds. In *13th Smart Card Research and Advanced Application Conference, Paris, France*, 2014 Nov. 5-7.

[4] J.-C. Bajard, L. Imbert, P.-Y. Liardet, and Y. Teglia. Leak Resistant Arithmetic. *Cryptographic Hardware and Embedded Systems - CHES 2004*, vol. 3156 of *LNCS*, pp 62–75, 2004.

[5] K. Bigou and A. Tisserand. Improving Modular Inversion in RNS Using the Plus-Minus Method. In *CHES*, pp 233–249. Springer, 2013.

[6] R. C. C. Cheung, S. Duquesne, J. Fan, N. Guillermin, I. Verbauwhede, and G. X. Yao. FPGA Implementation of Pairings Using Residue Number System and Lazy Reduction. In *Proc. of the 13th Intern. Conf. on Crypto. Hard. and Embed. Systems*, CHES'11, pp 421–441, 2011.

[7] K. Gandolfi, C. Mourtel, and F. Olivier. Electromagnetic Analysis: Concrete Results. *Cryptographic Hardware and Embedded Systems, CHES 2001*, vol. 2162 *LNCS*, pp 251-261, 2001.

[8] H. L. Garner. The Residue Number System. In *Papers Presented at the the March 3-5, 1959, Western Joint Computer Conference*, IRE-AIEE-ACM '59 (Western), pp 146–153, New York, NY, USA, 1959.

[9] N. Guillermin. A High Speed Coprocessor for Elliptic Curve Scalar Multiplications over $\mathbb{F}_p$. *Cryptographic Hardware and Embedded Systems, CHES 2010*, vol. 6225 *LNCS*, pp 48–64, 2010.

[10] M. Joye and S.-M. Yen. The Montgomery Powering Ladder. *Cryptographic Hardware and Embedded Systems, CHES 2002*, vol.2523 *LNCS*, pp 291–302, 2003.

[11] S. Kawamura, M. Koike, F. Sano, and A. Shimbo. Cox-Rower Architecture for Fast Parallel Montgomery Multiplication. In *the 19th International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'00, pp 523–538, 2000.

[12] P. C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In Neal Koblitz, editor, *Advances in Cryptology, CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pp 104–113. Springer Berlin Heidelberg, 1996.

[13] P. C. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In *Proc. of the 19th Annual Intern. Cryptology Conf. on Advances in Cryptology*, CRYPTO '99, pp 388–397, London, UK, UK, 1999. Springer-Verlag.

[14] P. L. Montgomery. Modular Multiplication without Trial Division. *Math. of Computation*, 44(170): pp 519–521, 1985.

[15] H. Nozaki, M. Motoyama, A. Shimbo, and S. Kawamura. Implementation of RSA Algorithm Based on RNS Montgomery Multiplication. In *Cryptographic Hardware and Embedded Systems - CHES 2001*, vol. 2162 of *LNCS*, pp 364–376, 2001.

[16] K.C. Posch and R. Posch. Modulo reduction in residue number systems. *Parallel and Distributed Syst., IEEE Trans. on*, 6(5): pp 449–454, May 1995.

[17] R.W. Watson and C.W. Hastings. Self-checked computation using residue arithmetic. *Proc. of the IEEE*, 54(12): pp 1920–1931, Dec 1966.

[18] G. X. Yao, J. Fan, R. C.C. Cheung, and I. Verbauwhede. Faster Pairing Coprocessor Architecture. In *Proc. of the 5th Intern. Conf. on Pairing-Based Cryptography*, Pairing'12, pp 160–176, 2013.