Improving the Masked Division for the FALCON Signature

Pierre-Augustin Berthet *†, Justine Paillet *‡, Cédric Tavernier *, Lilian Bossuet ‡, Brice Colombier ‡
* *Hensoldt France SAS*, Plaisir, France
† *LTCI, Télécom Paris, Institut Polytechnique de Paris*, Palaiseau, France
‡ *Université Jean-Monnet Saint-Étienne, CNRS, Institut d'Optique Graduate School, Laboratoire Hubert Curien UMR 5516*, F-42023, Saint-Étienne, France

***Abstract*—FALCON is a post-quantum signature selected by the National Institute of Standards and Technology (NIST). Although its side-channel resilience has been studied and a masking countermeasure proposed, the division is a major performance bottleneck. This work proposes a different approach to the masked FALCON division. We use the Newton method and a convergent sequence to approximate this operation. The performance of the masked division is improved by a factor 6.7 for two shares and 6.98 for three shares. For the Gaussian sampler, the improvements are of a factor 1.45 for two shares and 1.43 for three shares. Formal security proofs using the MIMO-SNI criteria are also provided.**

***Index Terms*—Post-Quantum Cryptography, FALCON, Side-Channel Analysis, Masking, MIMO-SNI**

I. Introduction

Future quantum technologies will solve the Discrete Logarithm Problem (DLP) thanks to the Shor algorithm [1] in practicable times. The National Institute of Standards and Technology (NIST) launched a competition [2] to select the next post-quantum cryptography standards. Among them is FALCON [3], a lattice-based signature.

Another threat is Side-Channel Analysis (SCA), first published by Paul Kocher [4]. An opponent with physical access to the device under attack can correlate the device's environmental parameters (power consumption, execution timing, etc.) with its sensitive data. It is important to study the resilience of FALCON against such attacks.

A. Related works

Several generic countermeasures were applied to defend FALCON against a side-channel opponent. Chen and Chen [5] proposed a masking methodology for the addition and multiplication of floating numbers. Karabulut and Aysu [6] did a similar work on a hardware platform, specifically on multiplication. In a previous work [7], we completed their design and provided a masking methodology for the floor function and division, as well as the first performances of a completely masked FALCON on a laptop computer.

Supported by Agence de l'Innovation de Défense (AID), French MoD, thanks to the grants 2022156 and 2023151 from the Thèses CIFRE Défense program. This paper is also part of the ongoing work of Hensoldt France SAS for the Appel à projets Cryptographie Post-Quantique launched by Bpifrance for the France 2030 vision. In this, Hensoldt France SAS is part of the X7-PQC project in partnership with Secure-IC, Télécom Paris, and Xlim.

B. Contribution

This paper proposes a different method for computing a masked division within FALCON. This design relies on the Newton method to find a root for a smooth function. We provide a performance comparison with an optimisation of our previous design on a laptop computer along with formal proofs.

Section II introduces the background and notations. Section III presents a new masked design for the division in FALCON. Section IV analyses its formal security. Section V provides a comparison with the division in our previous work. Section VI concludes the paper.

II. Preliminaries

A. Notations

We denote by (α_i) a secret sharing of a sensitive datum α. $\alpha_k, k \in \mathbb{N}$, refers to the k-ith share of the sharing. An exception is made for the specific case of a_0, a_n, a_{n+1}, and b_0, b_n, b_{n+1}, which are the 0-ith, n-ith and $n+1$-ith terms of convergent sequences.

The constant values $\sigma_{min} = 1.277833697$ and $\sigma_{max} = 1.8205$ are parameters of the FALCON-512 signature. For FALCON-1024, $\sigma_{min} = 1.298280334$. Throughout the paper, calculations involving σ_{min} will take the FALCON-512 value. The design stays the same for FALCON-1024, but calculations are not remade for this higher-parameter set of FALCON in this work.

Notations in Algorithm 1 are from the FALCON reference paper [3].

B. Masking

1) Overall idea: Among the generic countermeasures most studied against SCA is *masking* [8]. The sensitive datum is separated into random *shares* to decorrelate it from its leakage. The shares form a *sharing* of the secret. Each share is processed independently from the others. Recombining the shares after sensitive computations are performed ensures correctness.

2) Formal proofs: To verify the effectiveness of a masked gagdet, we rely on formal proofs. Ishai et al. [9] introduced the concept of *t*-probing security, formalising the resilience of a device against a side-channel opponent with measurement

probes. However, the composability of t-probing gagdets is not necessarily secure. To address this issue, Barthe et al. [10] proposed the concept of t-(Strong-)Non-Interference (t-NI, t-SNI):

Definition II.1 (t-Non Interference (t-NI) security [10])**.** *A gadget is said* t*-Non Interference (*t*-NI) secure if every set of* t *intermediate values can be simulated by no more than* t *shares of each of its inputs.*

However, *t-NI* gadgets composition does not imply *t-NI* security. Barthe et al. [10] propose a stronger definition for this:

Definition II.2 (t-Strong Non Interference (t-SNI) security [10])**.** *A gadget is said* t*-Strong Non-Interference (*t*-SNI) secure if for every set of t_I of internal intermediate values and t_O of its output shares with $t_I + t_O \leq$t, they can be simulated by no more than t_I shares of each of its inputs.*

Although the composition of t-SNI gadgets is t-SNI, it only applies to single-output gadgets. To prove the security of Multiple Inputs - Multiple Outputs (MIMO) gadgets, Cassiers and Standaert [11] introduced the t-MIMO-SNI model:

Definition II.3. *(*t-MIMO-SNI *security [11]). Let $\mathcal{O}_i$ be a set of shares indices for $i = 0, \ldots, d-1$. A gadget is* t-MIMO-SNI *if and only if for any set $\mathcal{I}$ of t_1 internal probes and any sets $\mathcal{O}_i$ such that there exists a t_2 that satisfies $t_1 + t_2 \leq t$ and $|\mathcal{O}_i| \leq t_2$ for $i = 0, \ldots, d-1$, the sets of probes $\mathcal{I} \cup y_{\mathcal{O}_0,0} \cup \cdots \cup y_{\mathcal{O}_{d-1},d-1}$ can be simulated with at most t_1 input shares.*

As we use composition proofs to formally verify the security of our design, a list of the sub-gadgets used for the masked division is the following:

- SecFprMul, which performs a masked multiplication on the floating-point masking of Chen and Chen [5],
- SecFprAdd, which performs a masked addition on the floating-point masking of [5].

Both gadgets have been proved t-SNI secure by Chen and Chen [5].

C. FALCON and division

FALCON [3], or FN-DSA, is a lattice-based signature selected by the NIST for standardisation. It relies on the NTRU and SIS problems over the GPV framework [12].

Its peculiarity is to use a Gaussian sampler and floating-point arithmetic. The sampler, named SamplerZ, is described in Algorithm 1. The function BaseSampler outputs an integer between 0 and 18. The function UniformBits(8) outputs 8 random bits uniformly. The function BerExp(x, ccs) outputs a single bit 1 with probability $ccs \cdot exp(-x)$.

The calculations addressed in this work are highlighted in red. Although there are two divisions, as they use the same datum, we can perform them once and simply refresh the result for its second use.

Algorithm 1: SamplerZ(μ,σ') [3]

Data: floating-point values μ,$\sigma' \in \mathbb{R}$ such that $\sigma' \in [\sigma_{\min}, \sigma_{\max}]$
Result: $z \in \mathbb{Z}$ sampled from a distribution very close to $D_{\mathbb{Z},\mu,\sigma'}$

$r \leftarrow \mu - \lfloor \mu \rfloor$;
$ccs \leftarrow \sigma_{min}/\sigma'$;
while *1* **do**
  $z_0 \leftarrow$ BaseSampler();
  $b \leftarrow$ UniformBits(8) And 0x1;
  $z \leftarrow b + (2 \cdot b - 1)z_0$;
  $x \leftarrow \frac{(z-r)^2}{2\sigma'^2} - \frac{z_0^2}{2\sigma_{\max}}$;
  if BerExp$(x, ccs) = 1$ **then**
    return $z + \lfloor \mu \rfloor$;

III. FALCON masked division

Instead of computing a masked division, we calculate a masked inversion and then perform a masked multiplication. In our previous work, the masked inversion used the naive Euclidean approach. This leads to a high complexity in terms of masked additions and multiplications, to a point where the inversion takes more than 50% of the overall cost of one call to the Gaussian sampler in [7]. FALCON performs either 1024 or 2048 calls to this sampler in one rejection loop, hence reducing the cost of the inversion improves the overall performance of the masked signature significantly. We use a different approach based on the Newton-Raphson method.

A. Newton-Raphson Method

To find a root β of a smooth function $f : \mathbb{C} \to \mathbb{C}$ such as the derivative of f in β is not zero, we can use the iterative scheme

$$a_{n+1} = a_n - \frac{f(a_n)}{f'(a_n)}. \tag{1}$$

If the first term a_0 is chosen close enough to β, this scheme will converge quadratically to this root.

To compute the inverse, there are several possibilities in the choice of f. Several works [13], [14] propose the same sequence for the computation of the inverse. It is described in Equation (2):

$$a_{n+1} = a_n(2 - a_n x). \tag{2}$$

However, the sequence converges efficiently only if the first term a_0 is a good approximation of the final result. As we perform this inversion in a masked manner, we have to compute this first approximation using the knowledge we have on the masked value σ' without revealing its true value.

In the FALCON Gaussian sampler described in Algorithm 1, we have $\sigma' \in [\sigma_{\min}; \sigma_{\max}]$. To approximate the first term of the sequence, we use a first-order minimax polynomial [15]:

$$\begin{aligned} t(x) &= p + q(x - \sigma_{min}) = -0.4298677x + 1.3215798 \\ p &= \frac{1}{2\sigma_{min}} - \frac{1}{2\sigma_{max}} + \frac{1}{\sqrt{\sigma_{min}\sigma_{max}}} \\ q &= -\frac{1}{\sigma_{min}\sigma_{max}} \end{aligned}$$

The evaluation of t in σ' can be performed masked and gives us a good first approximation of the inverse of σ' without revealing its value. The first term is thus $a_0 = t(\sigma')$.

The number of iterations of the sequence to perform depends on the precision required by FALCON. The signature is tailored around the double precision for floating points, i.e. the tolerated margin of error is smaller than $1.11 * 10^{-16}$. To evaluate the number of iterations, we define the sequence $b_n = a_n - 1/x$. We have the following result:

$$\begin{aligned} a_{n+1} &= a_n(2 - a_n x) \\ &= (b_n + 1/x)(2 - (b_n + 1/x)x) \\ &= (b_n + 1/x)(1 - b_n x) \\ &= 1/x - b_n^2 x \\ b_{n+1} &= -b_n^2 x \\ b_n &= (-1)^n b_0^{2^n} x^{2^n - 1} \end{aligned}$$

As $b_0 = a_0 - 1/x = t(x) - 1/x$, we can rewrite the sequence b_n as follows:

$$b_n = (-1)^n x^{2^n - 1}(t(x) - 1/x)^{2^n} \tag{3}$$

Table I is obtained thanks to a computational model like Wolfram Alpha. It highlights the maximum of $|b_n|$ for several values of n. The optimum number of iterations for the sequence from Equation (2) in the case of FALCON is 4.

TABLE I
MAXIMA OF THE VALUES OF $|b_n|$ FOR $n \in [\![1; 5]\!]$ AND $b_0 = t(x)$, WITH $x \in [\sigma_{min}, \sigma_{max}]$

Number of terms n	1	2	3	4	5
Magnitude of $\lvert b_n \rvert$	10^{-4}	10^{-8}	10^{-15}	10^{-29}	10^{-58}

B. Masked Algorithm

Given the methods described in the previous section, a masked inversion using the sequence based on Equation (2) is described in Algorithm 2.

The inversion described in Algorithm 2 has a complexity of 9 masked multiplications and 5 masked additions, for a total of 14 masked operations. This beats the 55 masked additions required by SECFPRINV in our previous work.

Algorithm 2: Masked Inversion for FALCON

Data: 64-bit boolean sharing (x_i) of the secret x and an integer n number of iterations.
Result: 64-bit boolean sharing (out_i) for value $1/x$.

$(cstm_i) \longleftarrow$ (0xBFDB82F3D046D4D1, ..., 0)
$(csta_i) \longleftarrow$ (0x3FF52530DC40DE17, ..., 0)
$(cst_i) \longleftarrow$ (0x4000000000000000, $\cdots$, 0)
`// cstm=-0.4298677, csta=1.3215798, cst = 2`
$(tmp_i) \longleftarrow$ SECFPRMUL$((x_i), (cstm_i))$
$(tmp_i) \longleftarrow$ SECFPRADD$((tmp_i), (csta_i))$
for `i from 1 to n` **do**
  $(xa_i) \longleftarrow$ SECFPRMUL$((x_i), (tmp_i))$
  $xa_0 \longleftarrow xa_0$ XOR $(1 << 2^{63})$
  $(xa_i) \longleftarrow$ SECFPRADD$((xa_i), (cst_i))$
  $(tmp_i) \longleftarrow$ SECFPRMUL$((tmp_i), (xa_i))$
return $(out_i) = (tmp_i)$

IV. SECURITY

To formally verify the security of our design, we use the methodology described by Cassiers and Standaert [11]. They model a gadget as a Directed Acyclic Graph (DAG), or computation graph, where each vertex is a *t*-NI secure subgadget and the edges are the wires connecting the subgadgets to form the main gadget. Every subgadget has one output. A duplication vertex, denoted $Split_j$ is used if a datum is reused by j subgadgets. If a subgadget is known to be *t*-SNI secure, it is shown as a *t*-NI secure subgadget followed by a *t*-SNI Refresh vertex.

This graph model highlights how the sensitive datum's information propagates through the gadget, and thus which information a specific probe placement can reveal. Cassiers and Standaert [11] define the Simplified Computation Graph as follows:

Definition IV.1 (Simplified Computation Graph (SCG), [11])**.** *The simplification of the computation graph G is the graph obtained from G by removing all the* t*-SNI Refresh vertices and their incident edges.*

According to [11] (Proposition 7, page 2549), to prove the *t*-NI security of a gadget, it is sufficient to verify that its SCG is a Single Path - NI Built gadget (SP-NIB):

Definition IV.2 (*Single Path - NI Built gadget* (SP-NIB), [11])**.** *A composite gadget G is SP-NIB if it is implemented with only NI gadgets and SNI refreshes, and if for any pair of vertices u, v in the corresponding simplified computation graph there exists at most one path from u to v.*

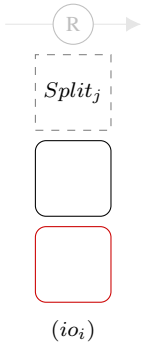
The conditions required for an SCG to verify *t*-MIMO-SNI security are as follows:

Proposition IV.1 ([11])**.** *A composite gadget G is* t*-MIMO-SNI if it satisfies the three following conditions. (i) G is SP-NIB. (ii) For any pair of output nodes u_1, u_2 there is no node*

v such that there is a path from v to u_1 and a path from v to u_2. (iii) For any pair of input nodes u_1, u_2 there is no node v such that there is a path from u_1 to v and a path from u_2 to v. (iv) There is no path from any input node to an output node.

To prove the *t*-SNI security of our inversion, we perform a composition proof using the *t*-MIMO-SNI criteria and Proposition IV.1 for the sub-gadgets of Algorithm 2. We use the legend described in Table II for the SCG used in the proofs. The *t*-SNI Refresh vertices are depicted as grey edges for clarity. They are not considered edges or vertices in the SCG when performing the proofs.

TABLE II
SCG LEGEND



: *t-SNI* Refresh represented as an edge.

: $Split_j$ vertex (has j outgoing edges).

: *t-NI* vertex.

: *t-MIMO-SNI* vertex.

: Input/output nodes.

Lemma IV.1. *The For loop from Algorithm 2 is* t*-MIMO-SNI secure when we take into account that the input of the first iteration is fed by a* t*-SNI gadget.*

Proof. The For loop is depicted in Figure 1. We choose to depict the edge from the input $(temp_i)$ to its $Split_2$ node as a *t*-SNI Refresh. The justification is twofold:

- The first iteration of the For Loop takes as input $(temp_i)$ which is outputted by a $SecFprAdd$ gadget. This is a *t*-SNI gadget, thus its representation in the SCG is a *t*-NI node followed by a *t*-SNI Refresh. This Refresh is thus represented in Figure 1 when we consider the first iteration of the For loop.
- The output $(temp_i)$ of the For loop is preceded by a *t*-SNI Refresh. As this output will serve as an input to the next iteration of the For loop, we can once again depict the edge from the input $(temp_i)$ to $Split_2$ as a *t*-SNI Refresh. This Refresh after the input thus represents the Refresh before the output from the previous iteration of the For loop.

This being clarified, we now verify that the conditions of Proposition IV.1 are met by the SCG in Figure 1.

(i) For every couple of nodes u, v, there is at most one path from u to v. The SCG is thus SP-NIB and we have *t*-NI security.
(ii) As there is only one output, the second condition of Proposition IV.1 is verified.
(iii) As explained previously, the input $(temp_i)$ is always refreshed and thus not "connected" to the remainder of the graph. As the input (in_i) is the only other input, the third condition of Proposition IV.1 is verified.
(iv) As there is a *t*-SNI Refresh before the only output of the For loop, there is no path from any input to the output. The final condition of Proposition IV.1 is verified.

Hence, according to Proposition IV.1, each iteration of the For loop from Algorithm 2 is *t*-MIMO-SNI secure, when taking into account that the input of the first iteration is fed by a *t*-SNI gadget. By composition of *t*-MIMO-SNI gadgets, the entire For loop is *t*-MIMO-SNI secure. □

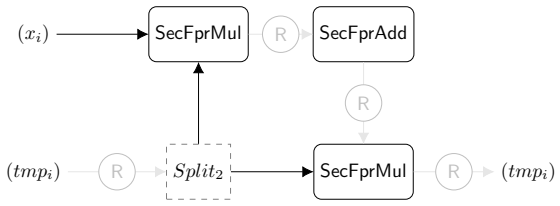


Fig. 1. SCG of the For loop of Algorithm 2

Theorem IV.1. *The inversion described in Algorithm 2 is* t*-SNI secure.*

Proof. The SCG of the algorithm is shown in Figure 2. For clarity, the n edges from node $Split_{n+1}$ to each iteration of the For loop are represented as one edge. As the For loop is *t*-MIMO-SNI secure according to Lemma IV.1. The SCG verifies the conditions of Proposition IV.1:

(i) The SCG depicted in Figure 2 is SP-NIB.
(ii) There is only one output.
(iii) There is only one input.
(iv) As the output is preceded by a *t*-SNI Refresh, there is no path in the SCG between the input and the output.

A single-input single-output *t*-MIMO-SNI gadget is a *t*-SNI gadget. The inversion described in Algorithm 2 is *t*-SNI secure. □

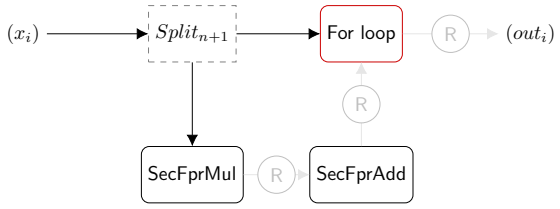


Fig. 2. SCG of Algorithm 2

V. PERFORMANCES

The performances are performed on a laptop computer. It is equipped with an Intel Core i7-11800H CPU. The compiler is *gcc version 9.4.0*. A comparison between the two inversion

methods and their impact on the Gaussian sampler SamplerZ is provided in Table III.

TABLE III
COMPARISON IN MICROSECONDS BETWEEN THE TWO INVERSIONS METHODS AND THEIR IMPACT ON SAMPLERZ

Algorithm	Paper	2 shares	3 shares
SecFprInv	[7]	268	740
Algorithm 2	This work	40	106
SamplerZ	[7]	704	1833
SamplerZ	This work	486	1280

An estimate of the performance of a masked signature of FALCON is given in Table IV.

TABLE IV
COMPARISON IN SECONDS BETWEEN THE TWO ESTIMATIONS OF THE COST OF PERFORMING A MASKED FALCON SIGNATURE

Algorithm	Paper	2 shares	3 shares
FALCON-512	[7]	3.198545	6.367758
FALCON-512	This work	1.281485	2.955044
FALCON-1024	[7]	6.908620	12.926950
FALCON-1024	This work	2.847457	5.431982

VI. CONCLUSION

In this paper, we present a different methodology to perform the masked division within the FALCON post-quantum signature compared to our previous masked implementation of FALCON [7]. We use the Newton-Raphson method and a convergent sequence to approximate the value of the inverse of a masked value. The performance of the masked division is improved by a factor 6.7 for two shares and 6.98 for three shares. The performance of the Gaussian sampler is improved by a factor 1.45 for two shares and 1.43 for three shares.

The next evolution of this work is to port the implementation on an embedded target, either microcontroller, FPGA or ASIC, to perform some experimental validation of the security of the design and prepare it for deployment.

REFERENCES

[1] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," *Proceedings 35th Annual Symposium on Foundations of Computer Science*, Santa Fe, NM, USA, 1994, pp. 124-134, doi: 10.1109/SFCS.1994.365700 .

[2] L. Chen, et al., "Report on post-quantum cryptography," Vol. 12, Gaithersburg, MD, USA: US Department of Commerce, National Institute of Standards and Technology, 2016.

[3] T. Prest, P-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Ricosset, G. Seiler, W. Whyte and Z. Zhang, "FALCON," Technical report, National Institute of Standards and Technology, 2020. [Online] Available: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions

[4] P. Kocher, "Differential power analysis," *Proc. Advances in Cryptology (CRYPTO'99)*, 1999.

[5] K.-Y. Chen, and J.-P. Chen, "Masking floating-point number multiplication and addition of Falcon: first- and higher-order implementations and evaluations," IACR Transactions on Cryptographic Hardware and Embedded Systems, 2024(2), 276-303, doi: 10.46586/tches.v2024.i2.276-303

[6] E. Karabulut, and A. Aysu, "Masking FALCON's floating-point multiplication in hardware," IACR Transactions on Cryptographic Hardware and Embedded Systems, 2024(4), 483-508, doi: 10.46586/tches.v2024.i4.483-508

[7] P.-A. Berthet, J. Paillet, C. Tavernier, L. Bossuet, and B. Colombier, "Masked computation of the floor function and its application to the FALCON signature," IACR Communications in Cryptology, vol. 1, no. 4, Jan 13, 2025, doi: 10.62056/ay73zl7s

[8] S. Chari, C.S. Jutla, J.R. Rao, P. Rohatgi, "Towards sound approaches to counteract power-analysis attacks," In: Wiener, M. (eds) Advances in Cryptology — CRYPTO' 99. CRYPTO 1999. Lecture Notes in Computer Science, vol 1666. Springer, Berlin, Heidelberg, doi: 10.1007/3-540-48405-1_26

[9] Y. Ishai, A. Sahai, and D. Wagner, "Private circuits: securing hardware against probing attacks," In: Boneh, D. (eds) Advances in Cryptology - CRYPTO 2003. CRYPTO 2003. Lecture Notes in Computer Science, vol 2729. Springer, Berlin, Heidelberg, doi: 10.1007/978-3-540-45146-4_27

[10] G. Barthe et al., "Strong non-interference and type-directed higher-order masking," In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16). Association for Computing Machinery, New York, NY, USA, 116–129, doi: 10.1145/2976749.2978427

[11] G. Cassiers and F. -X. Standaert, "Trivially and efficiently composing masked gadgets With probe isolating non-interference," in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 2542-2555, 2020, doi: 10.1109/TIFS.2020.2971153

[12] C. Gentry, C. Peikert and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in Proceedings of the fortieth annual ACM symposium on Theory of computing (STOC '08), Association for Computing Machinery, New York, USA, 2008, pp. 197-206, doi: 10.1145/1374376.1374407

[13] P. Soderquist and M. Leeser, "Division and square root: choosing the right implementation," in IEEE Micro, vol. 17, no. 4, pp. 56-66, July-Aug. 1997, doi: 10.1109/40.612224

[14] J. D. Blanchard, and M. Chamberland, "Newton's Method Without Division," The American Mathematical Monthly, 130(7), pp. 606–617, doi: 10.1080/00029890.2022.2093573

[15] M. P. Vestias and H. C. Neto, "Revisiting the Newton-Raphson Iterative Method for Decimal Division," 2011 21st International Conference on Field Programmable Logic and Applications, Chania, Greece, 2011, pp. 138-143, doi: 10.1109/FPL.2011.33