HAL
open science

# Improving Euclidean Division and Modular Reduction for Some Classes of Divisors

Jean-Claude Bajard, Laurent Imbert, Thomas Plantard

## To cite this version:

Jean-Claude Bajard, Laurent Imbert, Thomas Plantard. Improving Euclidean Division and Modular Reduction for Some Classes of Divisors. Asilomar Conference on Signals, Systems and Computers, Nov 2003, Asilomar, CA, United States. pp.2218-2221, 10.1109/ACSSC.2003.1292374 . lirmm-00269572

**HAL Id: lirmm-00269572**

**https://hal-lirmm.ccsd.cnrs.fr/lirmm-00269572**

Submitted on 7 Jun 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Improving Euclidean Division and Modular Reduction for some Classes of Divisors

Jean-Claude Bajard, Laurent Imbert, and Thomas Plantard

LIRMM, CNRS, Université Montpellier II
161 rue Ada, 34392 Montpellier cedex 5, FRANCE

## Abstract

*Modular arithmetic is becoming an area of major importance for many modern applications; RNS is widely used in digital signal processing, and most public-key cryptographic algorithms require very fast modular multiplication, and exponentiation. When such an arithmetic is required, specific values such as Fermat or Mersenne numbers are often chosen since they allow for very efficient implementations. However, there are cases where only very few of those numbers are available. We present an algorithm for the Euclidean division with remainder and we give the classes of divisors for which our algorithm is particularly efficient compared to commonly used method.*

## 1 Introduction

With no doubt, modular multiplication is the most important arithmetic operation of today's public-key cryptographic algorithms [6]. During the last three decades, many solutions have been proposed to speed-up modular arithmetic. Although modular multiplication can be performed by interleaving multiplications and modular reductions [2], evaluating the product and then reducing is often a preferred option since one can take advantage of fast multiplication algorithms (see [3, 4, 9] for more details).

Modular multiplication and reduction algorithms are thus very closely related. They can be classified in many different ways. For instance, a consideration that can be taken into account for the classification is the requirement precomputed values [1] or look-up tables[8, 5]. We can also distinguish between those which do not depend on any specific modulus, like the widely used Montgomery's algorithm [7], and those that only consider specific ones, like Fermat or Mersenne numbers. Of course, taking advantage of specific modulus provides very efficient solutions but there are cases where only very few of those specific numbers are available. In this paper, we propose an intermediate approach by defining new classes of modulus with wider size and high scalability. Our solution seems very efficient compared to commonly used techniques.

## 2 Problem and Notations

Given the two positive integers $D$ and $0 \leq X < D^2$, we compute the results of the Euclidean division $X/D$ ; i.e. the integers $Q, R$ which satisfy the following equation:

$$X = QD + R, \text{ with } R < D. \tag{1}$$

Throughout the paper we consider the following notations: $D$ is a $n$-digit integer in base $\beta$:

$$\beta^{n-1} \leq D < \beta^n, \tag{2}$$

$X$ is a $2n$-digit number in base $\beta$ ; for instance, the result of a multiplication of two $n$-digit numbers, such that:

$$0 \leq X < D^2. \tag{3}$$

We use $a$ for the difference between $\beta^n$ and $D$

$$D = \beta^n - a, \tag{4}$$

and denote $k$ its size in base $\beta$: $|a| = k$. We define

$$A = \frac{a\beta^n}{D}. \tag{5}$$

## 3 Algorithm

Our algorithm proceeds in two steps. Instead of evaluating the quotient $Q = \lfloor X/D \rfloor$, we first compute an approximation of the quotient $\hat{Q} = Q - e$, with $e \in \{0, 1, 2\}$, and we deduce an approximation of the

remainder $\hat{R} = X - \hat{Q}D$. Since $e \in \{0, 1, 2\}$, the correct remainder is obtained with at most two subtractions.

Using (4), the evaluation of $Q = \lfloor X/D \rfloor$ can be rewritten as

$$Q = \left\lfloor \frac{X\,(D+a)}{D\,\beta^n} \right\rfloor = \left\lfloor \frac{X + \frac{X\,a}{D}}{\beta^n} \right\rfloor.$$

Using the same trick, i.e. by introducing $\beta$ in $\frac{X\,a}{D}$, and eq. (5), we obtain

$$Q = \left\lfloor \frac{X + \frac{X}{\beta^n} \times A}{\beta^n} \right\rfloor. \tag{6}$$

At this point, note that the divisions by $\beta^n$ reduce to simple shifts in base $\beta$.

We propose the following approximation:

$$\hat{Q} = \left\lfloor \frac{X + \varphi(\frac{X}{\beta^n})\,\psi(A)}{\beta^n} \right\rfloor, \tag{7}$$

or more exactly different approximations with increasing accuracy which depend on the level of accuracy of $\varphi(\frac{X}{\beta^n})$ and $\psi(A)$.

### 3.1 Evaluation of $\varphi(\frac{X}{\beta^n})$

We consider two cases depending on the error we can afford on $X/\beta^n$. As we shall see further, this error is controlled by the size of $a$.

In the first case, $\varphi(\frac{X}{\beta^n})$ is just the integer part of $\frac{X}{\beta^n}$. Clearly, we have

$$\varphi\left(\frac{X}{\beta^n}\right) = \frac{X}{\beta^n} - e, \tag{8}$$

with $e \in [0; 1)$.

In the second case, we only consider the $k+1$ most significant digits of the integer part of $\frac{X}{\beta^n}$. We compute:

$$\varphi\left(\frac{X}{\beta^n}\right) = \beta^{n-(k+1)} \left\lfloor \frac{X}{\beta^{2n-(k+1)}} \right\rfloor, \tag{9}$$

which lead to an error $e \in [0; \beta^{n-(k+1)})$.

### 3.2 Evaluation of $\psi(A)$

We first remark that $A$ can be written as

$$A = \frac{a\beta^n}{\beta^n - a} = \frac{a}{1 - a/\beta^n}.$$

Thus, we obtain different approximations of $A$ with increasing accuracy by evaluating the following series at increasing orders:

$$A = a + \frac{a^2}{\beta^n} + \frac{a^3}{\beta^{2n}} + \cdots + \frac{a^k}{\beta^{(k-1)n}} + \cdots \tag{10}$$

For example, if $a^2 < \beta^n$, we define $\psi(A) = a$, and we evaluate

$$Q' = \left\lfloor \frac{X + \left\lfloor \frac{X}{\beta^n} \right\rfloor \times a}{\beta^n} \right\rfloor.$$

In any cases, we are able to get can an approximation of $A$ s.t. $|A - \psi(A)| \leq 1$. Table 1 gives different approximations of $A$ depending on the size of $a$.

| Size of $a$ | Approximation of $A$ |
|---|---|
| $\lvert a\rvert \leq \frac{n}{2}$ | $\psi(A) = a$ |
| $\lvert a\rvert \leq \frac{2n}{3}$ | $\psi(A) = a + \lfloor \frac{a^2}{\beta^n} \rfloor$ |
| $\lvert a\rvert = k$ | $\psi(A) = \sum_{i=1}^{\frac{n}{n-k}} \frac{a^i}{\beta^{(i-1)n}}$ |

**Table 1. Different approximations of $A$.**

### 3.3 Bounds on $Q$

The approximations on $\frac{X}{\beta^n}$ and $A$ lead to an error on $Q$ such that:

$$\hat{Q} = Q - e \quad \text{with} \quad e \in \{0, 1, 2\}.$$

Thus, the correct result is obtained with at most two subtractions by $D$.

*Proof:* Let us denote $e_1$ the error on $\varphi(X/\beta^n)$:

$$\varphi\left(\frac{X}{\beta^n}\right) = \frac{X}{\beta^n} - e_1,$$

and $e_2$ the error on $\psi(A)$:

$$\psi(A) = A - e_2, \quad \text{with } e_2 \in [0, 1). \tag{11}$$

Using (7), we get

$$\begin{aligned}
\hat{Q} &= \left\lfloor \frac{X + \left(\frac{X}{\beta^n} - e_1\right)(A - e_2)}{\beta^n} \right\rfloor \\
&= \left\lfloor \frac{X + \frac{X}{\beta^n}A - e_1 A - e_2 \frac{X}{\beta^n} + e_1 e_2}{\beta^n} \right\rfloor \\
&\geq \left\lfloor \frac{X + \frac{X}{\beta^n}A}{\beta^n} \right\rfloor + \left\lfloor -\frac{e_1 A + e_2 \frac{X}{\beta^n} - e_1 e_2}{\beta^n} \right\rfloor
\end{aligned}$$

From (6), we obtain

$$\hat{Q} \geq Q - e,$$

with

$$e = \left\lceil \frac{e_1 A + e_2 \frac{X}{\beta^n} - e_1 e_2}{\beta^n} \right\rceil \geq 0.$$

Let us look at $e$ in more details. Clearly, from (5), replacing $A$ by its value yields

$$e \leq \left\lceil \frac{e_1 \frac{a\beta^n}{D} + e_2 \frac{X}{\beta^n}}{\beta^n} \right\rceil$$

Since $X \leq D^2$, we have

$$e \leq \left\lceil \frac{e_1 \frac{a\beta^n}{D} + e_2 \frac{D^2}{\beta^n}}{\beta^n} \right\rceil \leq \left\lceil e_1 \frac{a}{D} + e_2 \frac{D^2}{\beta^{2n}} \right\rceil.$$

Replacing $e_1$ and $e_2$ by their respective maximum value given by (9) and (11), we obtain:

$$e \leq \left\lceil \beta^{n-(k-1)} \frac{a}{D} + \frac{D^2}{\beta^{2n}} \right\rceil$$

Since $D < \beta^n$, the second term $\frac{D^2}{\beta^{2n}}$ is less than 1. For the first term we have

$$\beta^{n-(k-1)} \frac{a}{D} \leq \beta^{n-(k-1)} \frac{\beta^k}{D} < \frac{\beta^{n-1}}{D} < 1.$$

Taking the ceil function gives $0 \leq e \leq 2$, and since $e$ is an integer, we have $e \in \{0, 1, 2\}$. □

## 4 Example

Let us consider the following example in radix $\beta = 10$.

| Data: | |
|---|---|
| $D \quad (n = \lvert D \rvert = 10)$ | 9995566778 |
| $a \quad (k = \lvert a \rvert = 7)$ | 4433222 |
| $X$ | 56789098765432101234 |
| Evaluation of $\psi(A)$: | |
| $a^2$ | **19653**457301284 |
| $\left\lfloor \frac{a^2}{10^{10}} \right\rfloor$ | **1965** |
| $\psi(A) = a + \left\lfloor \frac{a^2}{10^{10}} \right\rfloor$ | 4435187 |
| Evaluation of $\hat{Q}$: | |
| $X$ | **5678909**8765432101234 |
| $\varphi(\frac{X}{10^n}) = 10^{n-k} \left\lfloor \frac{X}{10^{2n-k}} \right\rfloor$ | **5678909000** |
| $\varphi(\frac{X}{10^n})\psi(A)$ | 25187023370983000 |
| $X + \varphi(\frac{X}{10^n})\psi(A)$ | **5681428578**8803084234 |
| $\hat{Q} = \left\lfloor \frac{X + \varphi(\frac{X}{10^n})\psi(A)}{10^n} \right\rfloor$ | **5681428578** |
| Evaluation of $\hat{R}$: | |
| $X$ | **56789098**765432101234 |
| $\hat{Q}D$ | **56789098**745836581684 |
| $\hat{R} = X - \hat{Q}\,D$ | **00000000**19595519550 |
| Final correction: | |
| $R = \hat{R} - D$ | 9599952772 |

## 5 Complexity

We analyze the computational complexity of our algorithm by counting the number of elementary operations, i.e. the number of multiplication of single digits in radix $\beta$. We do not consider the evaluation of $\psi(A)$ in our complexity analysis since it can be precomputated if necessary, and for some values of $D$ does not requires any computations at all (the case $\psi(A) = a$).

The evaluation of $\hat{Q}$ requires the multiplication $\varphi(X/\beta^n)\psi(A)$, where the two operands are of size at most $k+1$ digits. Thus the cost is $(k+1)^2$.

The complexity for the evaluation of $\hat{R}$ is more tricky. Since $\hat{R} < 3D$, we only need to compute the $|3D|$ less significant digits of $\hat{Q}D$, i.e. $n+2$ in base 2, and $n+1$ for all base $\beta$ greater than 2. One can also remark that since $D = \beta^n - a$, it is more interesting to evaluate $\hat{R} = X - \hat{Q}\beta^n + \hat{Q}a$. Thus, we only

consider the cost of the product $\hat{Q}a$, which requires $kn - \sum_{i=1}^{k-3} i = kn - \dfrac{(k-3)(k-2)}{2}$ elementary multiplications. The total cost is then

$$C = (k+1)^2 + kn - \frac{(k-3)(k-2)}{2}.$$

In table 2 we roughly estimate $k$, the size of $a$, which yield to some given complexities. We consider the cases 1 $1/2, 1, 3/4$, and $1/2$ multiplications. In figure 1 we

| Cost | Exact bounds on $k$ |
|---|---|
| $C < \frac{3n^2}{2}$ | $k \leq -\frac{9}{2} - n + \sqrt{\frac{89}{4} + 9n + 4n^2}$ |
| $C < n^2$ | $k \leq -\frac{9}{2} - n + \sqrt{\frac{89}{4} + 9n + 3n^2}$ |
| $C < \frac{3n^2}{4}$ | $k \leq -\frac{9}{2} - n + \sqrt{\frac{89}{4} + 9n + \frac{5n^2}{2}}$ |
| $C < \frac{n^2}{2}$ | $k \leq -\frac{9}{2} - n + \sqrt{\frac{89}{4} + 9n + 2n^2}$ |

**Table 2. Bounds on $k$ for some given costs.**

have plotted the relative size of $a$ according to $D$ which allows us to reach the costs considered in the previous table. For example, the lowest curve means that when $k$ represent about 40% of $n$, the cost of our algorithm is less that half a multiplication.

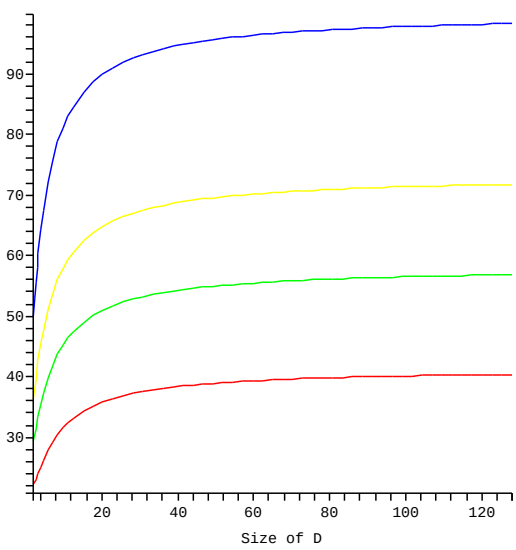


**Figure 1. Complexity and size of $a$ based on the size of $D$.**

## 6 Conclusions

The proposed algorithm is interesting if we can not use the usual specific divisors (Fermat, Mersenne), or when we need more than those available in the dynamic range. If the size of $a$ is less than 70% the size of $D$, the cost of our reduction is less than one multiplication. So, in this case, the cost of our modular multiplication is less than two multiplications which is the best we can have with Montgomery multiplication. Also, when $|a|$ is about 50% $|D|$, we do not need to perform any precomputation since the approximation $\psi(A) = a$ holds.

## References


[1] P. D. Barret. Implementing the rivest shamir adleman pubic key encryption algorithm on a standard digital signal processor. In *Advances in Cryptology – CRYPTO'86*, number 263 in LNCS, pages 311–323. Springer-Verlag, 1986.

[2] S. R. Dussé and J. B. S. Kaliski. A cryptographic library for the motorola DSP56000. In *Advances in Cryptoloy – Eurocrypt 90*, number 473 in LNCS, pages 230–244, New York, 1990. Springer-Verlag.

[3] A. Karatsuba and Y. Ofman. Multiplication of multidigit numbers on automata. *Soviet Physics—Doklady*, 7(7):595–596, Jan. 1963.

[4] D. E. Knuth. *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*. Addison-Wesley, Reading, MA, third edition, 1997.

[5] C. H. Lim, H. S. Hwang, and P. J. Lee. Fast modular reduction with precomputation. In *Proceedings of Korea-Japan joint Workshop on Information Security and Cryptology*, Seoul, Korea, October 1997.

[6] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone. *Handbook of applied cryptography*. CRC Press, 2000 N.W. Corporate Blvd., Boca Raton, FL 33431-9868, USA, 1997.

[7] P. L. Montgomery. Modular multiplication without trial division. *Mathematics of Computation*, 44(170):519–521, April 1985.

[8] N. Takagi and S. Yajima. Modular multiplication hardware algorithms with a redundant representation and their application to RSA cryptosystem. *IEEE Transactions on Computers*, 41(7):887–891, July 1992.

[9] D. Zuras. More on squaring and multiplying larges integers. *IEEE Transactions on Computers*, 43(8):899–908, 1994.