A massively parallel hardware for modular exponentiations using the m-ary method

M. S. Farias[1], N. Nedjah[2], L.M. Mourelle[2]
e-mail: msantana@ien.gov.br

[1] *Division of Nuclear Engineering - IEN*
[2] *UERJ*

Keywords: modular exponentiation, cryptographic systems, modular multiplication.

Introduction

Most of cryptographic systems are based on modular exponentiation. It is performed using successive modular multiplications. One way of improving the throughput of a cryptographic system implementation is reducing the number of the required modular multiplications. Existing methods attempt to reduce this number by partitioning, the exponent has constant or variable size windows. A simple strategy of raise text to n, or perform n-1 sequential multiplication, demands processing time that could be avoided by a concurrent approach: a (significantly large) series of modular exponentiations may be split into parallelized exponentiations upon pre-computed values, reducing notably total time to encrypt text. This report shows a proposed hardware implementation for computing modular exponentiations using the m-ary method and parallelized exponentiations. This study is of interest for applications in communication security and data transmission in the nuclear area [1].

Methodology

Modular multiplication is a very important operation for cryptography systems. The algorithm encrypts and decrypts information performing the operation $C = T^E \bmod M$, wherein E is called exponent and M modulus is the module chosen from the product of two primes. Note that the larger the prime numbers are, the more secure the process is. The solution developed seeks to parallel implementation of the exponentiation $T^E \bmod M$ using different multipliers. As in the m-ary method [2], the exponent E is divided into w partitions or windows with d bits. The computation can be described as in (1).

$$T^E = (T^{p_{w-1}})^{2^{(w-1)d}} \times \ldots \times (T^{p_i})^{2^{id}} \times \ldots \times (T^{p_1})^{2^d} \times T^{p_0}, \quad (1)$$

wherein exponent E is viewed as in (2).

$$E = p_{w-1} \times 2^{(w-1)d} + \ldots + p_i \times 2^{id} + \ldots + p_1 \times 2^d + p_0 \quad (2)$$

From (1), one can envision the computation of all $(T^{p_i})^{2^{id}}$ in parallel once the pre-computation of T^{pi} mod M has been completed.

Proposed architecture

The Figure 1 shows the macro-architecture of the proposed modular exponentiator. It includes the power memory (PMEM), a scalable number of modular multipliers (MMULTs), the main controller (MCTRL), wherein MMULTs receive their operands from PMEM via the shared data bus (DBUS). Memory PMEM stores the repository of the pre-computed powers of T. Modular multiplier MMULT implements a modular multiplication using the Montgomery algorithm. Each MMULT performs iteratively in order to provide a given power of one of the pre-computed values stored in PMEM. Binary-coded exponent is split into $w \geq 2$ partitions. Each partition comprises $d \geq 2$ bits. The number of MMULTs coincides with the number of partitions. The hardware has three processing stages: pre-computation, squaring and multiplication.

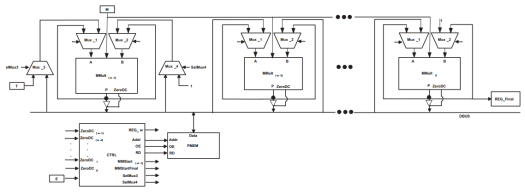

Fig. 1 –Macro-architecture of the parallel exponentiator

Results

A parametrized VHDL code was written and simulated on ModelSim XE III 6.4. The tests show that the modular exponentiator takes about 40 clock cycles to yield one single result [2]. It is a practical demonstration that parallelized modular exponentiation remarkably speeds up modular exponentiations, which is a critical stage on encryption/decryption related-computation in most crypto systems. The performance depends on parameters w, d and the number of bits in E. The best configuration should find a balance regarding hardware area and response time.

References

[1] Cordaro, Joseph V., et al. "Ultra secure high reliability wireless radiation monitoring system." *Instrumentation & Measurement Magazine, IEEE* 14.6 (2011): 14-18.

[2] Farias, M. S., Raposo, S. S. ; Nedjah, N., Mourelle, L. M. A Massively Parallel Hardware for Modular Exponentiations Using the m-ary Method. In: IEEE Latin American Symposium on Circuits and Systems (LASCAS 2011).