FAST RSA-HARDWARE : DREAM OR REALITY ?

Frank Hoornaert[1] Marc Decroos[1] Joos Vandewalle[2] René Govaerts[2]

[1] CRYPTECH NV/SA
Av. Lloyd George 7
1050 Brussels, Belgium

[2] ESAT K.U.LEUVEN
K. Mercierlaan 94
3030 Heverlee Belgium

ABSTRACT

This paper describes a successful hardware implementation of the RSA algorithm. It is implemented as an 120-bit bit-slice processor, which may be interconnected without additional circuitry to obtain arbitrary word lengths. With 512-bit operands, exponentiation takes less than 30 milliseconds.

I. INTRODUCTION

The actual explosion of electronic data communication and manipulation creates a still growing need for cryptography. This need exists as well for secret-key systems (using e.g. DES [1]) as for public-key systems (using e.g. RSA [2]). While DES can be efficiently implemented in software and hardware, the implementation of RSA is a lot more difficult in order to obtain a reasonable speed, especially for a software implementation. This drawback is certainly the main reason why up to now the RSA system has not been used more frequently, in spite of its very interesting cryptographic properties. (e.g. authentication, electronic signature, key management, etc ...). Although a few RSA implementations already exist or have been announced [4,5,6,7], a real breaktrough has not been achieved yet, mainly due to practical or economical reasons.

C.G. Guenther (Ed.): Advances in Cryptology - EUROCRYPT '88, LNCS 330, pp. 257-264, 1988.

In this context, CRYPTECH has started a project in cooperation with the K.U.Leuven to study and build an RSA implementation. These chips should be used in a first pilot project, which is the BISTEL system of the Belgian government [8]. The system requirements were the following :

- fast enough to allow on line encryption.
- making use of secure key lengths (e.g. 512 bit).
- compact enough to allow integration in existing equipment.
- it must be in conformity with cryptographical principles.
- low cost to make an RSA solution economical.
- available for commercial applications.

Now the project is finished and a hardware implementation using ASIC's (Application Specific Integrated Circuit) has been designed and built with success. The tests of the prototypes show RSA calculations at 9600 bit/sec and faster using 672 bit (= 200 digit) modulus and exponent.

II. ALGORITHMS

During the first development phase, different calculation methods were analysed. Very soon it appeared that hardware knowledge had to be integrated in the algorithmic study in order to obtain an optimal calculation scheme. Therefore, cooperation with IMEC [3] was set up to get necessary input from hardware engineers. The result was that an arithmetically simple calculation scheme evolved into an arithmetically complex calculation scheme in order to allow faster and more compact hardware implementation.

The original simple calculation scheme partitions the exponentiation with the well-known square-and-multiply algorithm [9] into subsequent multiplications. Then these multiplications are divided by the shift-and-add algorithm in subsequent additions and shifts. The additions are done according to the carry-save principle so that the addition is not delayed by the length of the numbers. The entire exponentiation must be calculated modulo n and this is performed by doing after each shift-and-add operation a reduction modulo n. The basic principle of that reduction is summarised in following algorithm.

Reduction algorithm.

Given modulus n, multiplicand A, multiplier B, intermediate result R, intermediate quotient q.

repeat for all bits b of B (starting with msb)
begin
$R := 2 * R + b * A$;
$q := \lfloor R/n \rfloor$;
$R := R - q * n$;
end.

However, a direct implementation of this reduction algorithm is very inefficient because three of the operations require a lot of time and/or hardware due to the length of the numbers, namely

- division R/n
- multiplication $q * n$.
- subtraction $R - q * n$.

Therefore some modifications are applied in order to simplify and speed up the hardware implementation :

1. The subtraction is replaced by an addition combined with the subtraction of all the overflow bits appearing after the addition. This gives the correct result only if the added value equals the value of the overflow bits minus the intented value to subtract.

   Example : (supposing 2 digit arithmetic)
   $(xx - 33)$ is equivalent to $(xx + 37)$ only if the overflow equals "100". (e.g. $xx = 75$ is OK, but $xx = 16$ is not OK.)

2. The number of possible quotients will be limited to the values 0,1,2 and 3 so that the multiplication can be replaced by a small table of precomputed values [10].
3. The values in the table are not the values to subtract, but are the values to add corresponding the above described modification. This implies that with every table entry (= possible quotient) a needed overflow is associated.

4. The division is replaced by a sub-estimation of the quotient which uses only a very small part of the bits of R and n. This estimation function must at the same time take care of following problems :

   - In spite of all the imposed limitations on the quotient q, the estimation has to remain accurate enough to avoid a systematic increase of R which would create a divergence.
   - For every intermediate result R, the quotient q may only take those values which guarantee that the condition of the correct overflow after the addition of the table value will be fulfilled. This problem can be solved thanks to the limited carry propagation of the carry-save addition.

It is clear that these modifications result in an arithmetically more complex algorithm. The hardware mapping of the algorithm however is more optimal because only two additions on long numbers are needed every cycle. The subtraction of the overflow is performed by wiping out some bits and the subestimation of the quotient can be done by a small and fast combinatorial circuit (< 20 nsec.)

III. IMPLEMENTATION

The chip architecture is a direct mapping of the algorithm. A first adder stage performs the conditional addition of the multiplicand and a second adder stage performs the modulo reduction. The result is stored and simultaneously the decision is made about the most optimal quotient for the next modulo reduction.

Each chip contains a datapath for 120 bits and chips can be hardware concatenated to arbitrary datapath lengths. (e.g. 720 bit or longer). In any case, the datapath must be at least as long as the used modulus. Even after concatenating a fixed number of chips, the used keylength can still be changed arbitrarily between 32 bit and the maximum value imposed by the hardware.

Besides the arithmetic part, the general structure and behaviour has also been optimised in order to get a universally useful module. For instance :

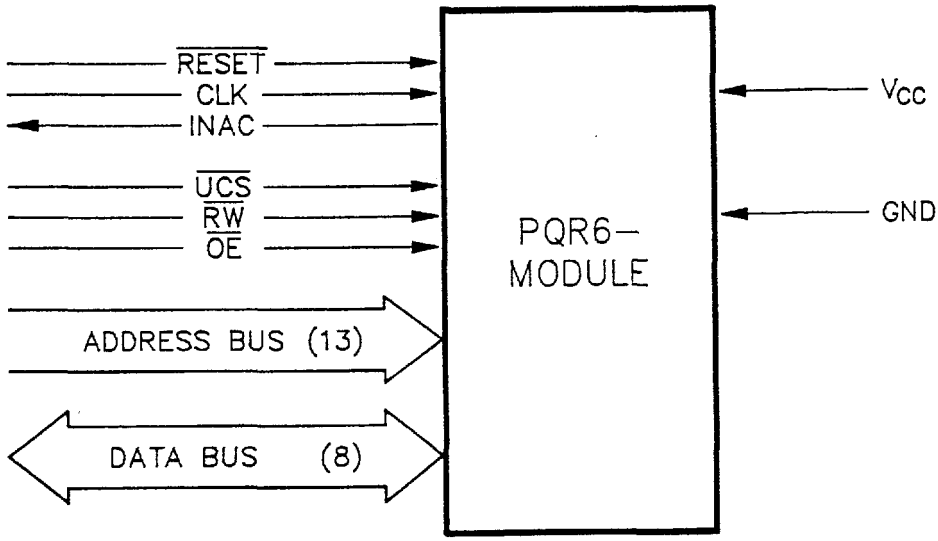

Figure 1: Architecture of the communciation.

1. Powerful tasks can be done without external aid. E.g. a complete RSA calculation.
2. A self-kill instruction destroys all internally stored keys in case of detection of an intruder.
3. Keys can be entered and during this process, the keys can be read out in order to check proper hardware functioning. After the entering is completed, the key can never again be read out or can't even be changed partially.
4. Up to 16 complete keys (e and n) can be memorised by the module.
5. The external interface of the module is very similar to the interface of a standard RAM. Therefore it can be coupled with almost every microprocessor bus (fig. 1).

IV. PERFORMANCE

The following table gives an overview of the datarates which have been achieved with the RSA hardware. The speed is linearly dependent on the exponent length, so that the use of very short exponents (e.g. 3, 65537) can boot up the speed [9,11]. By putting modules in parallel, a supplementary speed gain factor up to 10 is possible. The module is completely built in the latest CMOS technology ($1.5\mu m$) and consumes about 400 mA at maximum speed. A total of about 200,000 transistors are incorporated in a 6 chips module (maximum 712 bit modulus) which has the size of an actual pocketcalculator (13.9×6.4 cm^2) (fig. 2).

modulus length	256 bit	512 bit	672 bit
exponent = 65537	512 Kb/s	512 Kb/s	512 Kb/s
exponentlength = moduluslength	35 Kb/s	17 Kb/s	13 Kb/s

Table 1: Speed of a single module (14 MHz clock).

V. CONCLUSIONS

In the paper it is shown that a compact and fast (17 Kb/s for 512 bit) gate array chip design is feasible. The actual chip development is in the commercial phase. Testsamples are already tested and fully approved. Mass production quantities of the chips are available and the first RSA security systems using these production chips are actually under test (BISTEL [8]). An evaluation package including a 712-bit module, an interface card for the IBM PC, sources of driver software (C-language) and Hot-line problem support, is now available.

Future actions are on one hand the support of these RSA chips and derived products (PC-encryptors, key generators, high-speed encryptors, ...). On the other hand the availability of fast RSA implementations should stimulate the research and development of public key cryptography, which was forced too long in the past to proceed without actual fast hardware.

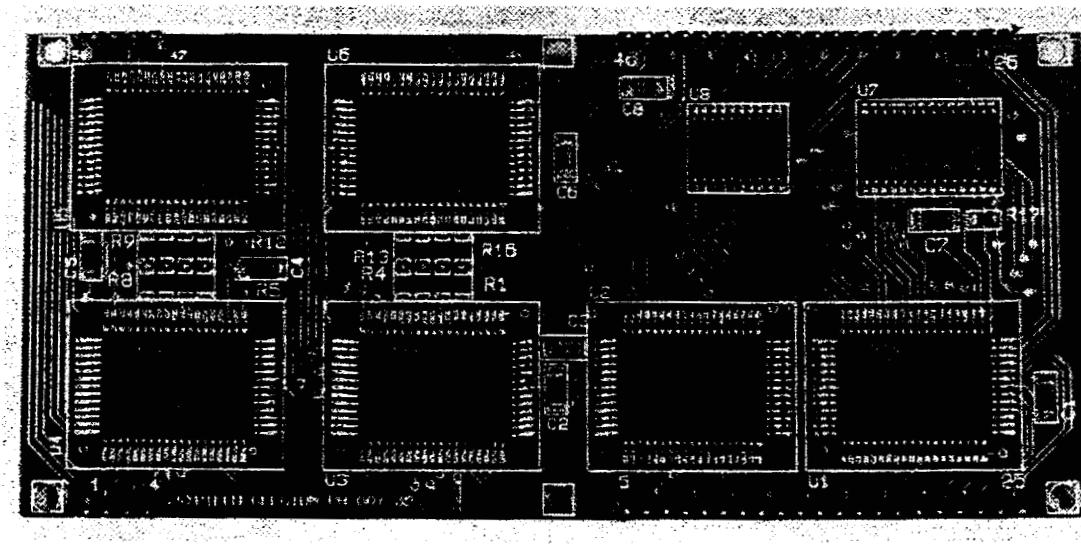

Figure 2: Photograph of the RSA module for keys up to 712 bit.

References

[1] National Bureau of Standards, *Data Encryption Standard*, U.S. Department of Commerce, FIPS Pub. no. 46, January 1977.

[2] R.L.Rivest, A.Shamir and L.Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Commun. ACM*, vol. 21, pp. 120-126, February 1978.

[3] IMEC (Interuniversitair Micro Electronica Centrum), Kapeldreef 75, B-3030 Heverlee Belgium, Tel. 32-(0)16-281211.

[4] R.L. Rivest, "A Description of a Single Chip Implementation of the RSA Cipher", *LAMBDA Magazine*, Vol. 1, No.3 (Fourth Quarter 1980), pp.14-18.

[5] M. Kochanski, "Split Key", *Systems International*, October 1986.

[6] S. Miyaguchi, "Fast encryption algorithm for the RSA cryptographic system", *Proceedings COMPCON 1982 - Twenty-fifth IEEE Computer Society International Conference*, September 1982.

[7] J.C.Pailles and M.Girault, "The Security Processor CRIPT", Pre-prints of the Fourth IFIP Conference on Information System Security, Monte-Carlo, December 1986.

[8] J.Vandewalle, R. Govaerts, W. De Becker and M.Decroos, "Implementation study of public key cryptography protection in an existing electronic mail and document handling system", *Advances in Cryptology, Proc. of EUROCRYPT '85 (Lecture Notes in Computer Science ; 219)*, F. Pichler, Ed., Springer-Verlag, Berlin, 1986, pp. 43-49.

[9] D.E.Knuth, *The art of computer programming. Vol. 2 : Seminumerical algorithms*, Addison-Wesley, Reading, MA, 1981.

[10] E.F. Brickell,"A Fast Modular Multiplication Algorithm with Application to Two Key Cryptography", *Advances in Cryptology, Proc. of CRYPTO '82*, D. Chaum, R.L. Rivest and A.T. Sherman, Eds., Plenum, New York, 1983, pp. 51-60.

[11] H. Sedlak, "The RSA Cryptography Processor", *Advances in Cryptology, Proc. of EUROCRYPT '87 (Lecture Notes in Computer Science ; 304)*, D. Chaum and W.L.Price, Eds., Springer-Verlag, Berlin, 1988, pp. 95-105.