RSA ALGORITHM OPTIMIZATION ON ASSEMBLER OF TI TMS320C54X SIGNAL PROCESSORS

Milan Marković[1], *Tomislav Unkašević*[1], *Goran Đorđević*[2]
[1]Institute of Applied Mathematics and Electronics
Kneza Miloša 37, 11000 Belgrade, Yugoslavia,
e-mail: mmarkov@beotel.yu, utom@Eunet.yu
[2]Military Technical Academy
Ratka Resanovića bb, 11000 Belgrade, Yugoslavia,
e-mail: djordjevicg@mail.ru

ABSTRACT

Possible optimization techniques for RSA algorithm realization on assembler of Texas Instruments TMS320C54x family of signal processors are considered. Obtained results justify the use of the proposed optimization methods and also show that the TMS320C54x family of signal processors is suitable for the RSA algorithm realization. Therefore, they are proposed for realization of some coprocessor module in computer networks with "client-server" architecture requiring some public-key cryptographic applications.

1 INTRODUCTION

RSA algorithm [1] is a typical representative and probably the most popular asymmetric cryptographic algorithm. The RSA algorithm is widely used in emerging e-commerce and e-business systems for creating "digital signature" and "digital envelope" according to PKCS#1 standard [2].

In this work, some aspects of the RSA public-key algorithm realization on assembler of Texas Instruments TMS320C54x family of signal processors are considered. A stress is posed to the optimization possibilities of the RSA algorithm assembler's implementation. First, a modified Karatsuba-Offman's algorithm [3] is considered as the multiplication procedure in the TMS320C54x assembler RSA algorithm realization. Also, a possibility of the application of the squaring procedure for improving the efficiency of the entire multiplication process is analyzed. Obtained experimental results justify the use of the proposed optimization techniques for the multiplication procedure in the RSA algorithm implementation, instead of using the standard multiplication algorithm. Additionally, the use of Chinese Remainder Theorem [4] for RSA private key operation is considered. The obtained results show that TMS320C54x family of signal processors is suitable for the RSA algorithm realization and, thus, they are proposed for the realization of some cryptographic coprocessor module in computer networks with "client-server" architecture requiring the asymmetric cryptographic applications.

The paper is organized as follows. Proposed optimization techniques for the RSA algorithm assembler implementation are considered in Section 2. Section 3 is dedicated to a brief description of the main characteristics of the signal processors of TI TMS320C54x family. Section 4 is dedicated to the experimental analysis while concluding remarks are given in Section 5.

2 OPTIMIZATION TECHNIQUES FOR THE RSA ALGORITHM IMPLEMENTATION

RSA (Rivest, Shamir, Adleman) algorithm, as a typical asymmetrical cryptographic algorithm, is described in details in [1]. In a sequel, a brief description of the RSA algorithm application is given.

Suppose that person A wants to establish a secure communication with person B by using the RSA algorithm. Person A determines the numbers n, e and d with following characteristics:

1. n is a product of two prime numbers, denoted as $n=p*q$.
2. a greatest common divisor: $gcd(e,(p-1)*(q-1))=1$
3. product of numbers: $e*d=1 \bmod ((p-1)*(q-1))$, i.e. $d=e^{-1} \bmod ((p-1)*(q-1))$.

A pair of numbers (e,n) is publicly available (it is called "public key") and serves to person B for encrypted messages intended to person A or to verify digital signatures of messages sent by person A. A pair (d,n) is called "private key" and it is only known by person A and serves for decryption or digital signing of the messages. RSA encryption and decryption work in the following way:

1. A message M, as a bit stream, which is to be sent from B to A is divided into corresponding number of parts:
   $$M=M_1M_2 \dots M_k,$$
   such as: M_i, $i=1,2,\dots, k$; is a number from the set $\{0,1,\dots, n-1\}$.
2. RSA encrypted message $C=C_1C_2\dots C_k$ is obtained by the following:
   $$C_i = M_i^e \pmod n,$$
   and it is sent to person A.
3. Person A decrypts message C and reconstructs message M by using its private key (d,n) and following operation:

$$M_i = C_i^d \pmod{n}.$$

In modern e-commerce and e-business systems, RSA algorithm is mainly used according to PKCS#1 standard. PKCS#1 standard [2] describes a method for encrypting data using the RSA public-key cryptosystem. Its intended use is in the construction of digital signatures and digital envelopes, according to the sintax described in PKCS#7 standard [5].

For digital signatures, the content to be signed is first reduced to a message digest with a message-digest algorithm (such as MD5 or SHA-1), and then an octet string containing the message digest is encrypted with the RSA private key operation of the signer of the content. The content and the encrypted message digest are represented together according to the syntax in PKCS#7 to yield a digital signature.

For digital envelopes, the content to be enveloped is first encrypted under a symmetric encryption key with a symmetric encryption algorithm (such as DES, 3DES, IDEA, AES, ...), and then the symmetric encryption key is encrypted with the RSA public key of the recipient of the content. The encrypted content and the encrypted symmetric encryption key are represented together according to the syntax in PKCS#7 to yield a digital envelope.

A security achieved by using the RSA algorithm is based on a fact that the private key could be compromised only by factorization of RSA modulus *n*. This way, by proper choosing the number *n*, possible factorization could be very complex, uncertain and long-term problem. In this sense, the number *n* should be more than 1024 bits long.

Having in mind the size of the number *n*, it is obvious that some multiprecise calculation methods have to be used for realization of RSA algorithm. A complexity of RSA algorithm realization is traditionally given in equivalent number of ordinary multiplication, which are necessary for obtaining the final result. Exponentiation is commonly realized as a number of successive multiplications, but it is not the most efficient way. Some optimal techniques with limitations according to particular situations or some sub-optimal methods, on the other side, are designed for solving this problem. In this realization, we have used a technique of non-zero variable-length windows, see [6].

On the other side, the standard division algorithm with remainder is not used for realization of modular reduction. It is shown that modular reduction could be realized by some of the methods, which not requires dividing. Montgomery's approach, as one of such method, is described in [7] and considered in this paper.

The multiplication is a critical operation in the RSA algorithm realization. In this paper, instead of standard multiplication operation, we propose the modified Karatsuba-Offman's multiplication algorithm. A detailed description of Karatsuba-Offman's algorithm could be finding in [3]. Additionally, since the original Karatsuba-Offman's algorithm is recursive, its implementation on 'C54x DSPs architecture is not efficient. Thus, four variants of the Karatsuba-Offman's algorithm modifications regarding the number of levels of recursions are considered in this paper. Besides, the application of the squaring procedure is considered in order to improve the efficiency of the multiplication process in the RSA assembler's algorithm realization.

In order to additionally optimize the RSA algorithm realization, in this paper, the application of the Chinese Remainder Theorem (CRT) [4] for the RSA private key operation is elaborated.

3 TI TMS320C54X FAMILY OF SIGNAL PROCESSORS

TMS320C54x ('C54x) [8] family of digital signal processors (DSP) offers the optimal combination of high performanse, peripheral options, small packaging and low power dissipation to give designers an edge in today's wireless and wireline communication markets. The 'C54x DSPs deliver the performanse demanded by wireless communications aplications like cellular phones, pagers, PCS terminals, as well as emerging applications like Internet Protocol (IP) phones and portable information appliances. The main characteristics of the 'C54x family of signal processors are:

- 40-532 MIPS performance,
- 40-bit ALU and two independent 40-bit accumulators,
- 17 x 17-bit multiplier,
- 1.5, 1.6, 1.8, 2.5, 3.3, and 5 volts versions are available,
- extended addressing mode for 8M x 16-bit maximum addressable external space,
- up to 640 K words on-chip RAM ('C5441),
- up to 24-channel direct memory access (DMA) controller,
- 8/16-bit host port interface (HPI),
- different types of serial ports are available: full-duplex serial port to support 8 or 16-bit transfers, time-divison multiplexed (TDM) serial port, buffered serial port (BSP) and multi-channel BSP (McBSP), and
- ultra-thin packaging (100, 128, and 144-pin TQFPs; 144 and 176 BGAs (Bal Grid Array)).

4 EXPERIMENTAL ANALYSIS

This chapter is dedicated to the experimental analysis of the 'C54x assembler's RSA algorithm realization. In order to experimentally show the efficiency of the proposed optimization procedures, first we consider multiplication optimizations based on the squaring and modified Karatsuba-Offman's procedures. In the given experimental analysis, the following parameters are adopted: two multipliers *a* and *b* are from 64 to 2048 bits long, and have three different values 0x55…55 (same numbers of 1 and 0), 0xff...ff (maximum values) and 0x80...01 (minimum values for *a* and *b*). For the purpose of this analysis, *a* and *b* have the same values.

Table 1: Ordinary multiplication and squaring procedure, values of $a=b$=0x5555...5555

a (bit)	b (bit)	Ordinary multiplication					Squaring procedure				
		Stand.	1Lr	2Lr	3Lr	4Lr	Stand.	1Lr	2Lr	3Lr	4Lr
64	64	167	467	1210	x	x	163	414	981	x	x
128	128	467	809	1770	4103	x	405	757	1557	3327	x
256	256	1547	1853	2944	5941	13042	1129	1623	2712	5185	10564
512	512	5627	5381	6372	9779	18882	3537	4075	5562	8904	16396
1024	1024	21467	18197	17548	20695	31048	12193	11859	13422	17962	28063
2048	2048	83867	66869	57180	55487	65100	44865	38947	37782	42558	56257

Table 2: Ordinary multiplication and squaring procedure, values of $a=b$=0xffff...ffff

a (bit)	b (bit)	Ordinary multiplication					Squaring procedure				
		Stand.	1Lr	2Lr	3Lr	4Lr	Stand.	1Lr	2Lr	3Lr	4Lr
64	64	167	559	1498	x	x	187	525	1264	x	x
128	128	467	921	2098	4917	x	477	902	1898	4115	x
256	256	1547	2005	3352	6885	15342	1297	1868	3157	6145	12796
512	512	5627	5613	6940	10983	21582	3897	4520	6311	10178	19142
1024	1024	21467	18589	18436	22419	34548	12937	12704	14779	20152	31753
2048	2048	83867	67581	58708	58251	70200	46377	40592	40355	46580	62699

Table 3: Ordinary multiplication and squaring procedure, values of $a=b$=0x8000...0001

a (bit)	b (bit)	Ordinary multiplication					Squaring procedure				
		Stand.	1Lr	2Lr	3Lr	4Lr	Stand.	1Lr	2Lr	3Lr	4Lr
64	64	117	452	1134	x	x	108	373	904	X	x
128	128	197	634	1609	3687	x	180	525	1295	2938	x
256	256	357	998	2263	5180	11474	324	829	1839	4135	9148
512	512	677	1726	3571	7358	16089	612	1437	2927	5943	12803
1024	1024	1317	3182	6187	11714	23055	1188	2653	5103	9559	18579
2048	2048	2597	6094	11419	20426	36987	2340	5085	9455	16791	30131

Table 4: Numbers of CPU cycles for RSA private key algorithm implementation ($n=d$=1024 bits)

Multiplication option	Standard implementation	CRT application
Ordinary multiplication	53 721 043	18 891 235
Squaring procedure	44 459 091	16 861 081
Squaring with modified Karatsuba=Offman's algorithm	43 321 105	16 701 452

Table 5: RSA private key realization in milliseconds depending of particular DSP from ‘C54x family ($n=d$=1024 bits)

Cycle (ns)	Standard implementation (ms)	Optimized algorithm with CRT application (ms)
25	2096.30	419.79
20	1677.04	335.83
15	1257.78	251.87
12.5	1048.15	209.89
10	838.52	167.91
8.33	698.49	139.87
6.25	524.07	104.95
5	419.26	83.96
1.875	157.22	31.48

Numbers of CPU time cycles for realization of the standard multiplication and the four variants of the Karatsuba-Offman's algorithm modification on 'C54x signal processors, with or without application of the squaring procedure, for different bit lengths of multipliers *a* and *b*, and for abovementioned three typical values of *a* and *b* are given in Table 1, 2 and 3. In these tables, Lr means levels of recursions of the modified Karatsuba-Offman's algorithm. All the results presented in Tables 1, 2 and 3 are obtained by including the modular reduction procedure based on Montgomery's approach. Based on the results, presented in Table 1, 2 and 3, we can conclude that better results could be achieved if the squaring procedure is included in the RSA algorithm multiplication process. Also, based on the results presented in Tables 1 and 2, we can conclude that the modified Karatsuba-Offman's algorithm show better results, compared to the standard multiplication procedures, only for multipliers of 512 bits long and more for ordinary multiplication and for multipliers of 1024 bits long and more for the squaring procedure. Results presented in Table 3, obtained for minimum values of *a* and *b*, show that better results could not be achieved by using the modified Karatsuba-Offman's algorithm, but this is not the most usual case.

Numbers of CPU time cycles for RSA private key operation based on standard multiplication, squaring procedure and joint squaring and modified Karatsuba-Offman's algorithm, with or without application of the Chinese Remainder Theorem (CRT), are given in Table 4. In this sense, we choose the RSA algorithm's parameters (*n*, *e*, and *d*) from real application conditions. Namely, we choose standard low-length *e* ($e=2^{16}+1$), while *d* and *n* are of the same length in bits (1024 bits). Also, the length of the processed messages is the same as the applied RSA modulus *n*.

Experimental results, presented in Table 4, regarding the chosen multiplication procedure, show that the best results are obtained by using the squaring procedure and modified Karatsuba-Offman's algorithm for multiplication optimization. Also, the results presented in Table 4 show that we could achieved more that 2.5 times better results for RSA private key operation by applying the CRT. All results presented in Table 4 are obtained with application of the Montgomery's approach for the modular reduction.

As for the possibilities of the RSA algorithm's realization on the 'C54x family signal processors, in Table 5, we give the calculated CPU time for the realization of the RSA private key operation with the same values of *d* and *n* (as in the case of experiments presented in Table 4) depending of the cycle time of the particular 'C54x signal processors. In table 5, we present the results obtained by the standard RSA algorithm implementation (ordinary multiplication and ordinary application of modular reduction process (by standard division algorithm)) and by the all proposed RSA optimization techniques, including: squaring procedure, modified Karatsuba-Offman's algorithm, Montgomery's approach and Chinese Remainder Theorem. Based on the presented experimental results in Table 5, we could conclude that assembler's realization of the RSA private key operation could be accelerated by about five times by using the set of optimization techniques, proposed in this paper. Also, we could conclude that about 30 1024 bits RSA private key transactions could be realized per second by using the fastest DSP from 'C54x family. Additional accelerating could be achieved by application of the specialized hardware elements for two large numbers multiplication.

Based on the entire experimental analysis, we could conclude that 'C54x signal processors represent a good basis for realization of the cryptographic coprocessor module for secure computer networks based on the client-server architecture.

5 CONCLUSION

In this paper, optimization possibilities for RSA algorithm realization on TI TMS320C54x signal processors are considered. The application of squaring procedure, modified Karatsuba-Offman's multiplication algorithm, Montgomery's modular reduction algorithm and Chinese Remainder Theorem (CRT) for RSA private key operation are elaborated. Presented experimental results justify the use of the proposed optimization procedures and show that the 1024 bits RSA private key operation could be accelerated about five times by applying them. This way, by applying the fastest 'C54x signal processor, about 30 1024 bits RSA private key operations per second could be realized. The presented results justify also that TI TMS320C54x signal processors are suitable for hardware implementation of RSA algorithm, and they are proposed for realization of some cryptographic coprocessor module in computer systems, which requires applications of the asymmetric cryptographic techniques.

REFERENCES

[1] R. Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Commun. of the ACM*, Vol. 21, No. 2, pp. 120-126, Feb. 1978.

[2] RSA Laboratories, *PKCS#1: RSA Encryption Standard*, Version 2, 1999.

[3] D. E. Knuth, *The Art of Computer Programming, Vol. II, Seminumerical algorithms*, Addison-Wesley, 1997.

[4] J. J. Quisquater, C. Couvreur, "Fast dechiperment algorithm for RSA public-key cryptosystem," *Electronic letters*, 18(21), pp. 905-907, Oct. 1982.

[5] RSA Laboratories, *PKCS#7: Cryptographic Message Syntax Standard*, Version 1.5, November 1993.

[6] C. K. Koc, "Analysis Of Sliding Window Techniques For Exponentiation," *Computers and Mathematics with Applications*, 30(10): 17-24,1995.

[7] P. L. Montgomery, "Modular Multiplication Without Trial Division," *Mathematics of computation*, 44(170): 519-521, April 1985.

[8] Texas Instruments, *TMS320C54x Digital Signal Processors*, Product Bulletin.