ON OPTIMIZING RSA ALGORITHM IMPLEMENTATION ON SIGNAL PROCESSOR: INFLUENCE OF ASYMMETRIC KEY LENGTH

Milan Marković[1], Goran Đorđević[2], Tomislav Unkašević[3]

[1]Mathematical Institute SANU, Knez Mihailova 35, 11001 Belgrade, NetSeT, Karađorđeva 65/4, 11000 Belgrade
Serbia, Yugoslavia, e-mail: milan@netset.co.yu, mmarkov@beotel.yu
[2]Military Technical Academy, Ratka Resanovića bb, 11000 Belgrade, Serbia, Yugoslavia,
e-mail: djordjevicg@mail.ru
[3]Institute of Applied Mathematics and Electronics, Kneza Miloša 37, 11000 Belgrade, Serbia, Yugoslavia,
e-mail: utom@Eunet.yu

I. Introduction

RSA algorithm 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.

In this work, influences of asymmetric key length to possible optimization of the RSA algorithm realization on assembler of Texas Instruments TMS320C54x family of signal processors are considered. Proposed optimization techniques include multiplication, modular reduction and private key operation procedures.

At the end of the paper, some concluding remarks regarding the optimal combination of the considered optimization techniques related to different asymmetric key lengths are given. Obtained results show that the TMS320C54x family of signal processors is suitable for the RSA algorithm realization.

II. 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. Following parameters are adopted: message m, private key d and public key n are from 128 to 2048 bits long. Numbers of CPU time cycles for RSA private key operation regarding the use of three considered modular reduction procedures: standard dividing (stddiv), reciprocal value method (RVM) and Montgomery's procedure (Mont), with application of different combinations of ordinary multiplication, squaring and modified Karatsuba-Offman's procedures, with or without application of the Chinese Remainder Theorem (CRT), are given in Table 1, 2 and 3.

In this sense, we choose the RSA algorithm's parameters (n 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. Also, the length of the processed messages is the same as the applied RSA modulus n. Chinese Remainder Theorem could be implemented only for the RSA private key operation (digital signature and asymmetrical decryption) since requires the knowledge of the p and q numbers. Regarding the modular reduction procedure, experimental results, presented in these tables, justify that the best results are obtained by using the Montgomery's modular reduction approach. Also, the results presented in Tables 1, 2, and 3 show that we could achieved more that 2.5 times better results for RSA private key operation by applying the CRT. Based on the results, presented in Tables 1, 2, and 3, we can conclude that the best results could be achieved by using Montgomery's procedure as the modular reduction method with application of the Chinese Remainder Theorem in the RSA algorithm implementation. Also, presented results justify the use of combination of squaring and modified Karatsuba-Offman's algorithms for optimization of the RSA algorithm's multiplication procedure.

Table 1: Numbers of CPU cycles for RSA private key algorithm implementation using ordinary multiplication

m (bit)	d (bit)	n (bit)	Standard implementation			CRT application		
			Stddiv	RVM	Mont	stdiv	RVM	Mont
128	128	128	584 115	295 414	200 054	300 765	145 997	115 073
256	256	256	2 634 626	1 627 506	1 108 378	1 166 915	592 344	487 163
512	512	512	13 760 619	10 758 553	7 296 468	5 242 024	3 264 975	2 784 167
1024	1024	1024	83 851 837	79 797 911	53 721 043	27 700 654	21 811 434	18 891 235
2048	2048	2048	569 165 635	612 145 906	409 895 160	167 374 150	159 253 661	135 067 307

Table 2: Numbers of CPU cycles for RSA private key algorithm implementation using ordinary multiplication and squaring procedure

m (bit)	*d* (bit)	*n* (bit)	Standard implementation			CRT application		
			Stddiv	RVM	Mont	stdiv	RVM	Mont
128	128	128	579 919	291 343	195 544	302 633	147 990	116 945
256	256	256	2 539 576	1 532 708	1 013 340	1 158 355	584 033	478 217
512	512	512	12 748 643	9 747 084	6 281 778	5 058 876	3 082 330	2 598 601
1024	1024	1024	74 592 179	70 539 272	44 459 091	25 675 268	19 787 066	16 861 081
2048	2048	2048	490 060 269	533 042 586	330 790 736	148 849 515	140 731 066	116 480 526

Table 3: Numbers of CPU cycles for RSA private key algorithm implementation using ordinary multiplication, squaring and modified Karatsuba-Offman's procedure

m (bit)	*d* (bit)	*n* (bit)	Standard implementation			CRT application		
			Stddiv	RVM	Mont	stdiv	RVM	Mont
128	128	128	579 919	291 343	195 544	302 633	147 990	116 945
256	256	256	2 539 576	1 532 708	1 013 340	1 158 355	584 033	478 217
512	512	512	12 734 330	9 679 147	6 264 578	5 058 810	3 082 264	2 598 535
1024	1024	1024	73 470 518	65 180 458	43 321 105	25 649 088	19 596 840	16 791 452
2048	2048	2048	465 103 873	441 442 154	305 680 740	146 661 495	130 329 671	112 005 224

As for the possibilities of the RSA algorithm's realization on the 'C54x family signal processors, in Table 4, we give the CPU time for the realization of the RSA private key operation with the same values of *d* and *n*, depending of the cycle time of the particular 'C54x signal processors. The results obtained by the standard RSA algorithm implementation (ordinary multiplication and ordinary application of modular reduction process (by standard dividing algorithm)) and the results obtained by applying the all of the proposed RSA optimization techniques, including: squaring procedure and modified Karatsuba-Offman's algorithm for multiplication, Montgomery's method for modular reduction, as well as Chinese Remainder Theorem, are presented In Table 4.

Based on the presented experimental results in Table 4, 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 can conclude that about thirty-three 1024-bit or five 2048-bit RSA private key transactions could be realized per second by using the fastest DSP from 'C54x family. Namely, we could see that RSA private key operation based on 2048 asymmetric key length is about seven times slower then the same operation based on 1024-bit key. Additional accelerating could be achieved by application of the specialized hardware elements for multiplication of two large numbers. 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 or Internet (WEB) architecture.

Table 4: RSA private key realization in milliseconds depending of particular DSP from 'C54x family

m, d, n (bit)	Cycle (ns)	Standard implementation (ms)	Optimized algorithm with CRT application (ms)
1024	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
2048	25	14229.14	2800.13
	20	11383.31	2240.10
	15	8537.48	1680.08
	12.5	7114.57	1400.07
	10	5691.66	1120.05

	8.33	4741.15	933
	6.25	3557.28	700.03
	5	2845.83	560.03
	1.875	1067.17	210.01

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] T. Unkašević, M. Marković, G. Đorđević, “Optimization of RSA algorithm implementation on TI TMS320C54x Signal Processors Based on a Modified Karatsuba-Offman’s algorithm,” in *Proc. ECMCS’2001*, 11-13 September, 2001, Budapest.

[4] M. Marković, T. Unkašević, G. Đorđević, “RSA Algorithm Pptimization on Assembler of TI TMS320C54x Signal Processors,” in *Proc. of EUSIPCO 2002*, Sept. 4-7, Toulouse, France.

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

[6] J.-F. Dhem, *Design of an Efficient Public-Key Cryptographic Library for RISC-based Smart Cards*, Ph. Dissertation, University Catholique de Louvain, May 1998.

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

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

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