KRISNA: CRYPTOPROCESSOR ARCHITECTURE FOR NATIONAL SMART CARD

Muhammad Husni Santriaji
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Indonesia
santriaji@s.itb.ac.id

Arif Sasongko
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Indonesia
asasongko@stei.itb.ac.id

Abstract—**Smart card technology has already widely used in Indonesia. Security threat on smart card is growing rapidly, therefore more security is needed. Ironically, Indonesia as one of the smart card most users very dependent on foreign technology and haven't built its own smart card architecture. This paper presents KRISNA, a cryptoprocessor architecture for national smart card. KRISNA has an ability to maintain the runtime of the smart card operation system by using 8051 architectural system as the main control. KRISNA uses a selection of lightweight implementation of cryptograpyc algorithm that can be implemented in smart card environment system. KRISNA has BC3 64-bit as symmetric cryptography, ECC 233-bit as asymmetric cryptography, WELL 2^{512} as pseudo random number generator and also SHA256 as hash function. KRISNA also accelerates the communication between the main controller and the coprocessor by implementing instruction buffer and shared memory controller.**

Keywords— **smart card; BC3; ECC; PRNG WELL; SHA256; Cryptoprocessor Architecture**

I. INTRODUCTION

A smart card is a device that includes an embedded integrated circuit that can be either a secure microcontroller or equivalent intelligence with internal memory or a memory chip alone[1]. Smart card has been used in many aspect of human life, such as in banking system and in government system.

Smart card security system must acommodate the need of various application requirement such as symetric algorithm, asymetric algorithm, hash function and pseudo random number generator. Due to the resource limitation in the smart card, almost all of the smart card didn't provide a complete cryptography resources. Some of them used simetric algorithm to implements RNG, authentication and PIN matching. This implementation of course can be harmful for security.

Indonesia as one of the most users very dependent on foreign smart card technology. Even for a sensitive government system application such as e-KTP, the country imported smart card from the foreign country.

Prior work already researchead about the codesign between 8-bit microcontroller and the cryptocoprocessor. However, those implementation doesn't cover all of the cryptography algorithm that needed on smart card security. A design of cryptoprocessor was presented on [2] but only consist of DES algorithm. A design of cryptoprocessor also presented on [3] but too complicated to be implemented for smart card. A design of cryptoprocessor on [4] doesn't have hash algorithm.

The architectural design for cryptoprocessor presented in this paper differs from previous work in two important aspects. First, we focus our implementation in smart card environment. Second, we pay attention on data transfer between microcontroller and coprocessor which always become a bottleneck in the previous works. Some works also already try to improve the data transfer between XRAM microcontroller and the cryptography coprocessor hardware. While those researchs show promising results, a relation between microcontroller and coprocessor in smart card application needs a unique treatment. Most of the time, the data transfer is not between XRAM memory and the cryptography coprocessor but among the cryptography coprocessor itself.

Data transfer between coprocessor cryptography and microcontroller RAM impacts the performance of a cryptography operation. Moreover a smart card runtime require an authentication operation which produces an intermediate value between cryptography component. Another approach is to used DMA (Direct Memory Addressing) to bypass the data transfer. We combine this two approach in lightweight ways so the data transfer rate is reduced and the resource area is preserved.

This paper introduces KRISNA a hardware implementation of a cryptoprocessor that implements private-key algorithms made in Indonesia (BC3), one standard public-key algorithm (ECC), hash algorithm(SHA256) and pseudo random number generator (WELL) in a single design that satisfies the functionality of smart card processor that not only can secure the data but also handle the smart card operating system.

II. CRYPTOGRAPHY ALOGRITHM

A. BC3 SYMMETRIC ALGORITHM.

Symmetric key encryption is a cryptography technique that uses a shared secret key to encrypt and decrypt data. BC3[5] algorithm is implemented in KRISNA architecture as symmetric algorithm because this algorithm is made in Indonesia. The BC3 algorithm is improvement of BC2[6] and AE3[7]. The characteristics of BC3 are quiet similar with AES [8] and Camellia [9]. It is developed with two considerations: projected robustness, and implementation efficiency. It means that this algorithm is designed to survive against various attacks or crypto analysis. At the same time, it is also designed to be implemented at various platforms efficiently and fast.

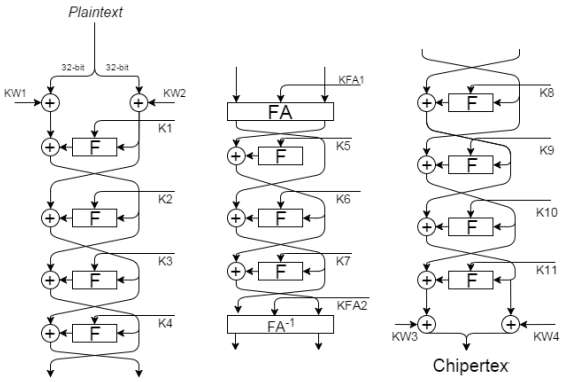

Fig. 1. Datapath of BC3.

The main features of BC3 algorithm are:

1. The width of input and output block is 64 bits.
2. The main key length is 128 bits.
3. Key expansion is done in two steps. Regular round is the same F function used in the randomizing stage.
4. Randomizing stages in encryption and decryption processes are applied in eleven regular rounds. Each regular round consists of the same F function.
5. In addition to the regular round, special function FA and FA^{-1} are used in every encryption and decryption process.

FA function is defined by:

$$Y_R = \left((X_R \cup KFA_1) \ll_1\right) \oplus X_R$$
$$Y_L = X_L \oplus ((\gg_1) \cap KFA_2)$$

FA^{-1} function is defined by:

$$X_R = \left((X_L \cup KFA_1) \ll_1\right) \oplus Y_R$$
$$X_L = Y_L \oplus ((Y_R \gg_1) \cap KFA_2)$$

where

$$\gg_1 \ shift\ right\ 1\ bit$$
$$\ll_1 \ shift\ left\ 1\ bit$$
$$\cap\ AND$$
$$\cup\ OR$$
$$\oplus\ XOR$$

BC3 implementation in VHDL only requires 3854 logic elements. Which is proven lightweight and suitable for smart card environment.

B. ELLIPTIC CURVE CRYPTOGRAPHY.

Elliptic curve cryptography (ECC) is a public-key cryptography. It based on the algebraic structure of elliptic curves over finite fields. ECC requires smaller keys compared to non-ECC cryptography to provide equivalent security.

Our ECC is designed to work on Koblitz Curve,

$$y^2 + xy = x^3 + x^2 + 1$$

at GF(2) and an extention field L=GF(2^{233}) with irreducible polynomial f(z)=z^{233}+z^{74}+1. Koblitz curve can eliminates the point doubling operation thus reducing the area and increasing the speed of ECC operation.

The arithmatic multiplication in GF(2^m) is the main operation in ECC operation. To reduce more resources, an interleaved multiplication is implemented. Interleaved multiplication is the most simple multiplication algorithm in GF(2^m).

KRISNA implements a τ-ary representation of k to accelerate the operation of ECC from [10]. VHDL implementation of ECC requires only 5,615 logic cells. Which is lightweight and suitable for smart card environment.

C. SHA256 SECURE HASH ALGORITHM.

Cryptographic hash function is a mathematical operation that runs on digital data to compare the computed "hash" (the output from execution of the algorithm) to a known and expected hash value. Cryptographic hash function is used to determine the data's integrity. KRISNA implements SHA256 [11] as cryptographic hash function because it has a small area and high energy efficiency. Hash function on smart card application is required to save the PIN password in safe mode.

SHA256 requires 512-bit input and produces 256-bit output. We use a VHDL implementation of SHA256 from opencores[12]. VHDL implementation of SHA256 only requires 2,381 logic elements.

D. WELL PSEUDO RANDOM NUMBER GENERATOR.

A pseudo random-number generator (PRNG) is a computational algorithms designed to generate a sequence of numbers or symbols that cannot be reasonably predicted better than by a random chance. While PRNG can be implemented entirely by software, the resource-restricted environmentand FIPS 140-2 requirements compelled the model to be partially based on hardware implementations. WELL algorithm [13] is used in KRISNA because it offers longer periode than a commonly used Linear-feedback shift register (LFSR) and a lighter resource than Marsenne Twister[14].

WELL algorithm has a periodic value of 2^{512}, long enough for cryptography application. FPGA implementation of WELL algorithm produces 8 bit random value every clock.

VHDL implementation of WELL PRNG only requires 1,330 logic elements.

III. CRYPTOPROCESSOR SPECIFICATIONS

A. System Overview.

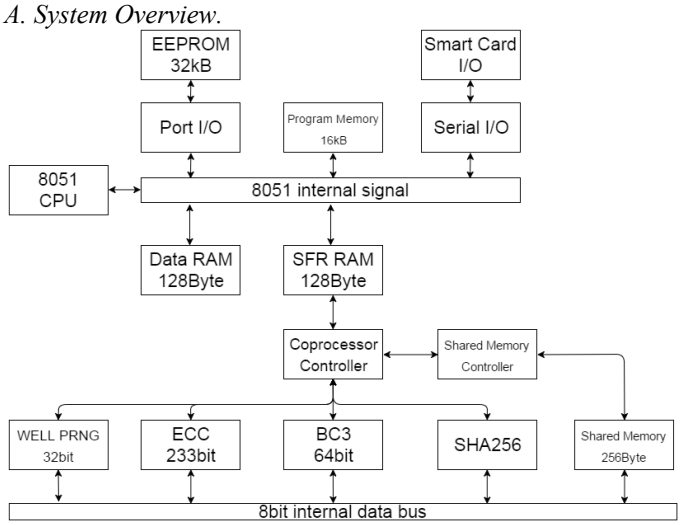

Fig. 2. System Overview.

KRISNA is a cryptoprocessor architecture system that consists of microcontroller and coprocessor like in figure 2. KRISNA implements 8051 microcontroller architecture from Oregano [15]. This Microcontroller is used to run the smart card operating system Pintar OS[16]. Coprocessor is controlled by microcontroller using the SFR register in 8051 RAM. The coprocessor consists of ECC 233bit as asymetric cryptography algorithm. ECC was designed as simple as possible to reserve the area. Therefore, an interleaved multiplication was used as the multiplication operator in GF(2^{233}). BC3 64bit is used as symetric cryptography. BC3 is a symetric cryptography algorithm which developt by Indonesia. BC3 is lighter than AES but gives more security. SHA256 is used as hash algorithm. WELL algorithm is used as PRNG algorithm.

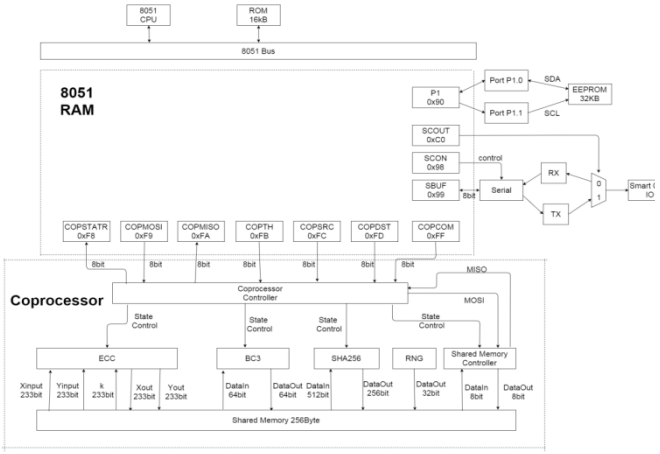

Fig. 3. Schematic of KRISNA.

B. Microcontroller.

The system main controller is based on 8051 architecture. Architecture 8051 has 256bytes RAM. To control the cryptograhy hardware, eight special function registers (SFRs) are added to the RAM. COPSTATR is the status register provides feedback about the current operation status of the cryptography component. COPMOSI is the data register to send data from microcontroler to coprocessor. COPMISO is data register that contains data from criptoprocessor. COPTH is the threshold register to control the iteration data transfer. COPSRC is address register to point the source of data in coprocessor. COPDST is address register to point the destination address of data in coprocessor. COPCOM is command register to command the cryptography component in the coprocessor. Table 1 shows all the command that should be put in the COPCOM register. This command can be easily added to smart card operating system.

Table 1. Command of KRISNA.

No	COMMAND	Notes
1	0x01	Memory Copy
2	0x02	Read Memory
3	0x03	Write memory
4	0x04	Memory Copy Block
5	0x05	HASH_START
6	0x06	BC3_Encrypt
7	0x07	BC3_Decrypt
8	0x08	ECC_START
9	0x09	BC3 Encrypt with previous key
10	0x0A	BC3 Decrypt with previous key
11	0xFF	Command run

C. Smart Card Operating System.

Smart card operating system is an operating system that embedded in the ROM of the smart card. Smart card operating system is not dependent on any particular application, but are frequently used by most applications. KRISNA is designed to run on smart card environment system, therefore a smart card operating system is needed. We implements PintarOS, an open source smart card operating system. PintarOS is written in C and supports ISO7816 smart card standard. PintarOS responsible for managing the interchanges between the card and the outside world, management of the files and data, access control to information and functions and management of card security and the cryptographic algorithm procedures.

D. Shared Memory Controller.

The main challenge in cryptoprocessor design is the data transfer between 8051 architecture 8-bit microcontroller and the coprocessor. The main purpose of the shared memory is to reduce the rate of data transfer between the coprocessor and the microcontroller. To reduce this problem, in this paper we reduce the transfer data rate by doing all of the data operation in the coprocessor memory without the microcontroller software involvement. Transfer data between IO of

cryptografphic component can be done by the shared memory controller by moving the data inside the shared memory. Figure 4 shows the inside of the shared memory controller. Each of cryptography component has its own address in shared memory. The data transfer between shared memory and the cryptography component can be done in one cycle.

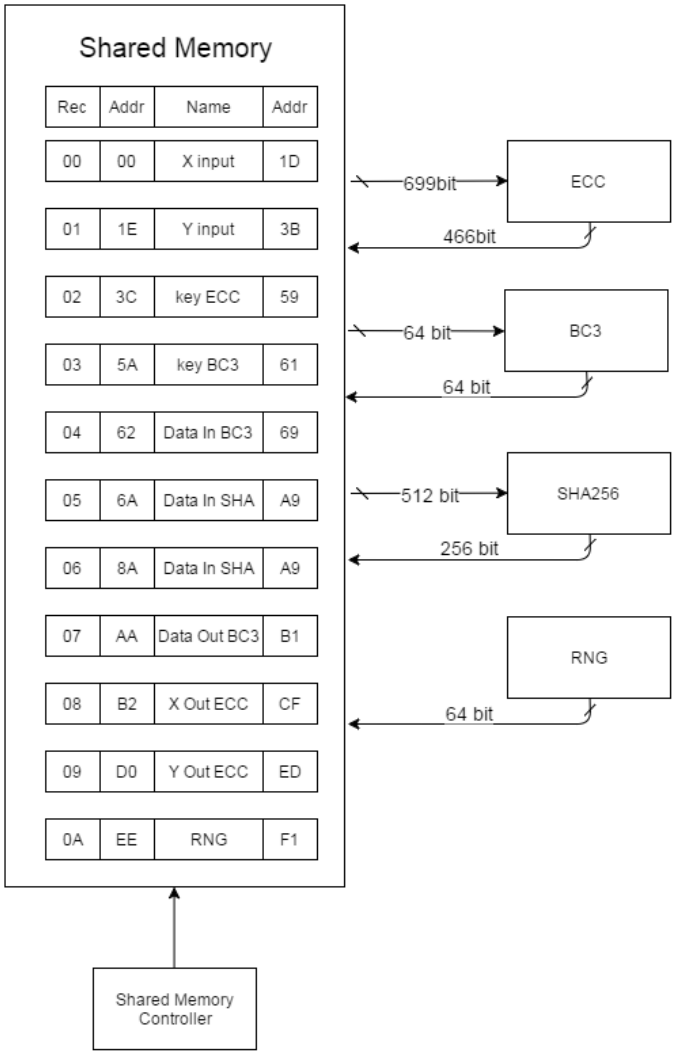

Fig. 4. Schematic of Shared Memory.

IV. RESULTS

Figure 5 shows KRISNA implementation in DE2-115 FPGA board. KRISNA requires 22.271 logic elements.

Flow Status	Successful - Thu Jan 07 14:22:26 2016
Quartus II 64-Bit Version	11.1 Build 216 11/23/2011 SP 1 SJ Full Version
Revision Name	smartcard_top
Top-level Entity Name	smartcard_top
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	22,271 / 114,480 (19 %)
Total combinational functions	19,615 / 114,480 (17 %)
Dedicated logic registers	9,567 / 114,480 (8 %)
Total registers	9567
Total pins	7 / 529 (1 %)
Total virtual pins	0
Total memory bits	951,951 / 3,981,312 (24 %)
Embedded Multiplier 9-bit elements	1 / 532 (< 1 %)
Total PLLs	0 / 4 (0 %)

Fig. 5. Implementation of KRISNA in DE2-115.

Slow 1200mV 85C Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	18.89 MHz	18.89 MHz	clk_reader	
2	81.81 MHz	81.81 MHz	altera_reserved_tck	

Figure 6. Timing Summary.

Figure 6 shows the fmax summary of the system architecture. KRISNA fmax of 18.89MHz meets the smart card requirements which is 4MHz.

We perform a verification on cryptography performance and functionality on KRISNA. Table 2 shows the cycles required to perform a single operation of cryptography algorithm.

Table 2. Cycles Required to Perform Cryptography Operation.

Name	Cycles (average)
WELL RNG	1
BC3 64-bit	12
ECC 233-bit	110,051
SHA256	67

KRISNA implements a shared memory to decrease the cycles required to move data between each coprocessor. Figure 6 compares the scenario of ECC application between architecture without a shared memory and KRISNA with shared memory. ECC application requires a random generated key from RNG coprocessor. An architecture without shared memory will transfer the data from RNG output to the ECC via microcontroller RAM. Meanwhile KRISNA has a shared memory that would bypass the transfer data. KRISNA requires less cycles because the data will move directly from RNG to ECC without transitting in microcontroller RAM. Table 3 shows the cycle required to transfer the 233-bit data from RNG output to ECC key input.

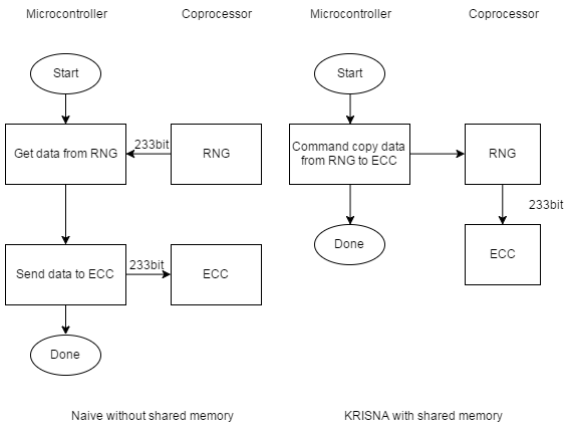

Figure 6. Shared Memory Application on ECC.

Table 3. Shared Memory Impact.

Naive	Krisna
616 cycle	126 cycle

Altough KRISNA is designed to work on smart card environment, it can be implemented in any other environment as well. The data transfer in this research is faster because the data transfer is controlled by the hardware that doesn't required a decode, fetch and execute cycle but only done by putting some value to the register. KRISNA results also in par with the dedicated cryptoprocessor that only has one coprocessor because KRISNA implemented record based memory address .

V. CONCLUSION

In this paper, we propose KRISNA a cryptoprocessor architecture that has a complete cryptographic resources. KRISNA has ECC-233 bit as asymmetric algorithm, BC3 as symetric algorithm, SHA256 as hash algorithm and WELL as PRNG algorithm. All of those cryptographic component has been tested and worked well in smart card operating system.

REFERENCES

[1] Rankl, Wolfgang. "Smart Card Handbook, 4th Edition." New York: John Wiley and Sons, Inc, 2010.

[2] G. Selimis, N. Sklavos and O. Koufopavlou, "Crypto processor for contactless smart cards," Electrotechnical Conference, 2004. MELECON 2004. Proceedings of the 12th IEEE Mediterranean, 2004, pp. 803-806 Vol.2.

[3] Ho Won Kim and Sunggu Lee, "Design and implementation of a private and public key crypto processor and its application to a security system," in IEEE Transactions on Consumer Electronics, vol. 50, no. 1, pp.214-224,Feb2004.

[4] Y. Eslami, A. Sheikholeslami, P. G. Gulak, S. Masui and K. Mukaida, "An area-efficient universal cryptography processor for smart cards," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 1, pp. 43-56, Jan. 2006.

[5] Sasongko, A., Hidayat, Kurniawan, Y., Sutikno, S., "Architecture for The Secret-Key BC3 Cryptography Algorithm" ITB Journal, 2011.

[6] Kurniawan, Y., Suwandi, A., Mardianto, M. S., Supriana, I., Sutikno, S. "The New Block Cihper: BC2", International Journal of Network Security, 8(1), pp. 16-24, 2009.

[7] Kurniawan, Y., "Analisis Sandi Diferensial AE3", Proceeding of Seminar Nasional IC2007 BINUS University, Indonesia, 2007.

[8] Satoh, A., Morioka, S., Takano, K., & Munetoh, S., "A Compact Rijndael Hardware Architecture with S-Box Optimization", ASIACRYPT 2001, LNCS 2248. pp. 239-254, 2001.

[9] Matsui, M., Nakajimi, J., "A Description of Camellia Encryption Algorithm", from http://www.ipa.go.jp/security/rfc/RFC3713EN.html, Accessed on July, 2016.

[10] Deschamps, Jean. "Hardware Implementation of Finite-field Arithmatic". San Fransisco: Mc-Graw Hill, 2009.

[11] Stevens, Marc. "Attacks on Hash Functions and Applications." PhD Thesis, Leiden: Leiden University, 2011.

[12] De-la-Piedra, A. " sha256core", from http://opencores.org/project,sha256core,Overview. Accessed on November, 2015.

[13] Panneton, Francois and L'Ecuyer, Pierre and Matsumoto, Makoto. "Improved Long-period Generators Based on Linear Recurrences Modulo 2." ACM Trans. Math. Softw., 2006: 1-16.

[14] Makoto Matsumoto and Takuji Nishimura. 1998. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. 8, 1 (January 1998), 3-30. DOI=http://dx.doi.org/10.1145/272991.272995.

[15] Oregano System, "8051 IP Core", from http://www.oreganosystems.at/?page_id=96. Accessed on May 2015.

[16] Hariady, Ricky, dan Arif Sasongko. "PintarOS: A Reconfigurable Multi-Purpose MultiApplication Smart Card Operating System." IEEE International Conference on Electronics Technology and Industrial Development. Bali: IEEE, 2013.