Journal of Engineering and Applied Sciences 13 (5): 1142-1147, 2018
ISSN: 1816-949X

Design and Development of Encryption Key Management Control Module Using Elliptic Curve Theory

[1]Sang-Hyun Lee and [2]Dong-Joon Chun
[1]Department of Computer Engineering, Honam University, Gwangju, Korea
[2]Department of Development and Planning and Management,
Gwangju Institute of Green-Car Advancement, Gwangju, Korea

Abstract: Recently, as the security problem for wireless internet environment has emerged, Elliptic Curve Crypto system (ECC) has emerged as a next generation cryptosystem that provides a solution to this problem. In this study, Simple Power Analysis (SPA) attack method which is one of the subchannel attacks is called public key-based password required for information security and authentication of smart home appliances included in smart grid system we design and develop a elliptic curve cryptosystem algorithm that can perform a given task independently.

Key words: ECC, SPA, public key, registers, accumulator, elliptic curve cryptosystem

INTRODUCTION

A hardware module for protecting the data of the information device in the smart home can provide the confidentiality and integrity of the data generated in the information device. Recently, personal information has been leaked in large quantities and with the enforcement of personal information laws such as the Personal Information Protection Act, it is urgent to develop a password technology for safely storing and managing important information such as personal information. The security of encrypted information is closely related to cryptographic algorithms as well as cryptographic key management. Recently with the emergence of security problems in the wireless internet environment, the Elliptic Curve Crypto system (ECC) has emerged as a next generation cryptosystem to solve this problem (Ah, 2017).

The use of elliptic curves in public key cryptosystems was independently proposed by Miller and Koblitz in 1985 (Anonymous, 1999). In elliptic curve discrete log, unlike factorization or discrete logarithm of finite sphere there is no applicable quasi-exponential time algorithm and parallelization of pollard-rho algorithm is known as the most efficient algorithm to solve elliptic curve discrete logarithm problem. To safely use ECCs to avoid these algorithms, elliptic curves defined in finite fields of about 160 bits or more should be used and curves known weakness with an ultra-specific curve, a non-regular curves and a curve with two traces must be avoided (Lopez and Dahab, 2000).

The most prominent advantages of elliptic curve cryptography is that it provides a similar level of safety while using short keys compared to RSA encryption method which is known as the first algorithm capable of encryption and digital signatures and existing public key cryptosystems method such as El-Gamal scheme developed by El-Gamal in 1982. These advantages have led to a lot of research in the academic world and it is a general opinion that it is suitable for environments where the throughput and computation volume are relatively poor such as wireless environment (Kwon, 2000).

However, the background theory is relatively complicated and it takes some time for the industry to widely use it because it requires a certain amount of expertise in the field to actually implement it. One of the reasons why the application was actually delayed to the industrial world is that the degree of recognition is relatively lower in comparison with RSA in terms of developers who do not have cryptographic expertise and policy makers. Table 1 compares the size of domain variables with the same security.

The public key cryptosystem can be computed within a theoretically finite time by computational complexity theory but it takes advantage of the intractability that computation takes too long in practice. The initial public key cryptography is based on the fact that it takes a long time to divide a very large integer into two or more prime numbers. Elliptic curve cryptography also designed that it takes a long time to find a discrete logarithm of a random elliptic curve for a certain known point (Johnson and Menezes, 1999).

Corresponding Author: Dong-Joon Chun, Department of Development Planning and Management, 55 Jingoksandanjuangang-ro, Gwangsan-gu, 62465 Gwangju, Korea

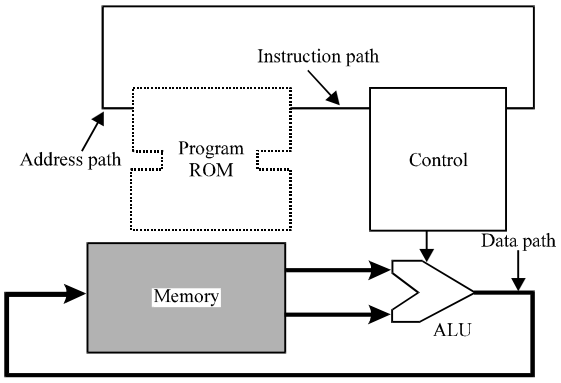

Fig. 1: Structure by module

Table 1: Comparison of the size of domain variables with the same security

Time to break in MIPS (Million Instraction Per Secound)	RSA/DSA (bits)	ECC (bits)	RSA vs. ECC key size ratio
10^4	512	106	5:1
10^8	768	132	6:1
10^{11}	1024	160	7:1
10^{20}	2048	210	10:1
10^{78}	21000	600	35:1

This study implements the elliptic curve cryptographic algorithm which corresponds to Simple Power Analysis (SPA) attack method which is one of the subchannel attacks and the public key-based cryptography required for information security and authentication of smart home appliances included in smart grid system and to design and develop the module that can perform the given task independently.

Literature review

Structure of process module: It is designed as a model suitable for ECC control by analyzing Intel 8051, Microchips PIC 1674 and Xilinx Picoblaze with the existing 8 bit CPU. First as shown in Fig. 1, the basic functions are divided into modules. An ALU module that receives an external key and performs basic operations required for processing, a memory module that stores results of operations or data and a ROM module that stores instructions can be largely divided.

To maximize functionality, all three of these modules were placed inside the FPGA to act as part of the ECC process. As all modules are inside, pins used outside the FPGA can be significantly reduced. To implement this, the program ROM uses the internal block RAM. Therefore, the number of pins to connect from inside is almost unlimited, so, it can experiment using the size of all cases that block RAM supports. For example, if a command width is small, a large amount of program memory can be obtained but the number of commands is reduced. On the contrary, if the command width is large, the number of commands increases but the program memory area becomes small. The available block RAM size is shown in Fig. 2.

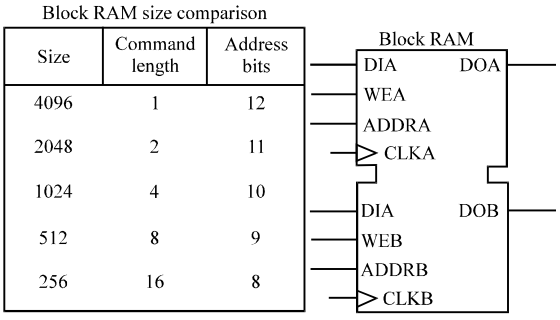

Fig. 2: Available size of internal block RAM

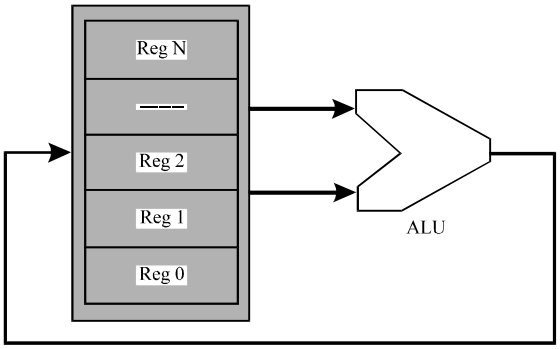

Fig. 3: Register architecture

Register structure: Registers can be selected by the program to temporarily store the data and the number of registers can be determined by considering the size of the entire processor as a temporary storage for operations or transmission to external devices. The number of bits required to represent a register is the same as the number of registers which also affects the size of the instruction. That is if 4 bits are allocated, 16 registers can be used.

To configure the registers, it will need the multiplexer logic to connect the ALU to the selected register by default. Figure 3 shows the simplified structure of the register.

Figure 4 shows the structure of the accumulator. An accumulator that is most closely connected to a register or memory and that can store various variables can be used either as a designated operand or as an omitted operand that does not affect instruction size. Therefore, the instruction can be implemented by allocating a small amount of logic because it requires only a bit area for use with the specified operand. Accumulator is very efficient

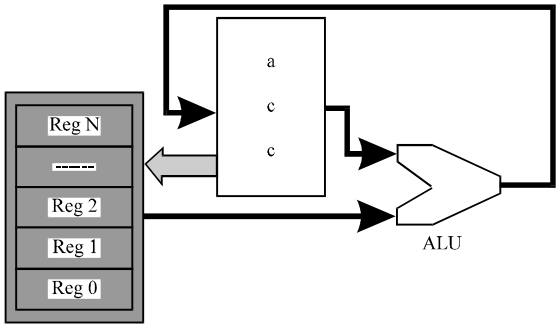

Fig. 4: Accumulator architecture

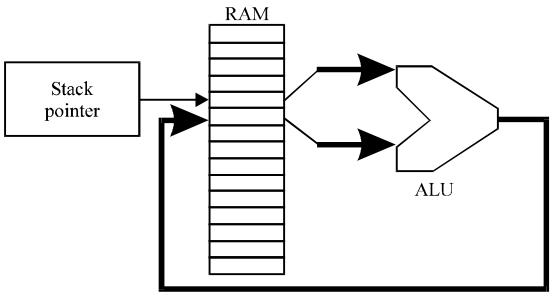

Fig. 5: Stack architecture

when used in a program that saves stored values for a long time but it slows down if it implements a program that frequently changes and stores values using an accumulator.

Because the stack is configured directly in memory there is no need for any logic other than the stack pointer. Since, the operand and the result of executing the instruction are always stored in the top of the stack, it can be applied to the instruction structure very efficiently. However, since, the stack pointer needs to be precisely moved and the overhead imposed on it, it is excluded from the POP or PUSH commands. Therefore, as shown in Fig. 5, the stack consists of only the logic that controls the PC (Program Counter) in the command branch (JP, CALL, RET) command.

MATERIALS AND METHODS

Design of proposed encryption key management control module

Design of registers: Assuming four 8 bit registers are implemented, 32 flip-flops are needed, plus a 4:1 multiplexer to select the operand and a clock enable signal supplied to the register (Fig. 6). Also, another multiplexer is needed for the instruction to select two operands at the same time (Table 2). However when the flip-flop and the multiplexer are used there is a time delay between the logic and the resources to be used are very large.

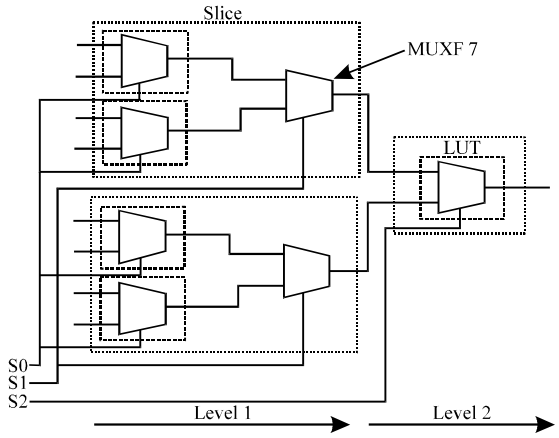

Fig. 6: Design of registers

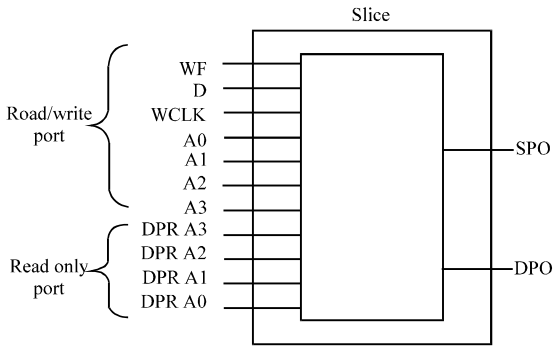

Fig. 7: Distributed RA

Table 2: Required resources of 8-bit registers (slice)

No. of registers	No. of flip flop	1 bit MUX	Select 2 operands	No. of gate	Size
4	16	1	16	2	18
8	32	2.5	40	4	44
16	64	5	80	8	88

Design of registers using distributed RAM: The cell configuration of the SRAM is generally implemented using a Look Up Table (LUT) of xilinx. Each LUT provides 16 bit distributed RAM and using two slices can implement 32 bit distributed RAM. Also, using 16 bit distributed RAM can create dual-port RAM for the register banks. As shown in Fig. 7, the 16 bit distributed RAM functions as a complete 1 bit, 16 register banks with write enable selection. Therefore, to implement 8 bit, 16 register banks, 8 slices are needed which are much smaller than the 88 slices mentioned above and the efficiency is greatly increased. Therefore, it designed 16 general-purpose registers of 8 bit size.

In the instruction structure design, the instruction basic structure for using 16 registers can be configured as follows. In the case of 2 operands and one register, 4 bits remain in the instruction code and 16 kinds can be expressed. In addition, since, there are no bits that can be

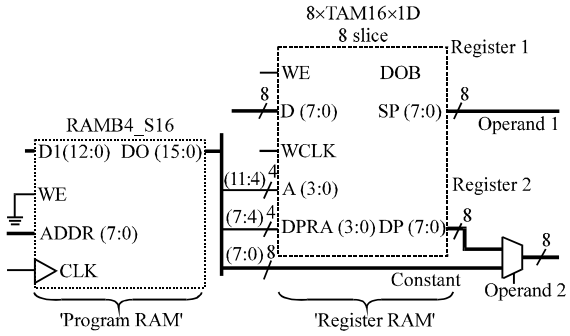

Fig. 8: Program memory design

allocated to the instruction in the case of two registers and an 8 bit constant value, if the result storage register is designed to be the same as the first operand register, four bits can be allocated to the instruction code. Thus, the instruction structure can be designed with a minimum size of 16 bits and a maximum size of 4096 bytes in a program ROM size of 256 bytes. In this task, 256 bytes were decided.

Program memory design: The block RAM simulates the program ROM and operates as a controller. Variables are stored in registers and are implemented as shown in Fig. 8 using the distributed cost-effective dual-port RAM as mentioned earlier.

In this study, since, the processor executes one instruction at a time sequentially from address 0, it is designed to increase the PC (Program Counter) after each instruction and it is easily implemented by using simple plus counter in Xilinx's Vertex 6.

In order to refer to 256 bytes of memory, only 8 bit program counter is required. However, since, there is a command to move control such as JP in the instruction, it is not enough to increase logic of simple PC (Program Counter). Therefore, the JP command requires a new 8-bit address to be inserted into the PC (Program Counter). When a JP command is encountered, logic must be prepared to prevent the program address from being loaded into the Program Counter (PC) and select a new address. This logic was designed by using the LUT as shown in Fig. 9.

The VALID-JUMP signal is active when a JP command is encountered and it is inverted to enter the carry logic to prevent the next address from being loaded. (Fig. 10) The processing of the CALL command or the RET command is almost the same as that of the JP command. When the CALL command is encountered, the next program address must be prevented from being loaded into the PC (Program Counter) a new address must be selected and the address to be returned from the end of the branching subroutine must be loaded into the PC. Also, unlike the JP and CALL commands, the RET

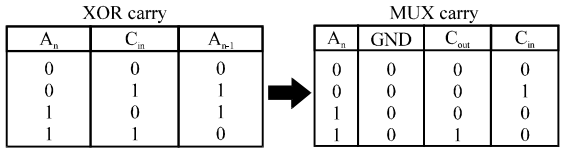

Fig. 9: Behavior of XOR carry and Mux carry

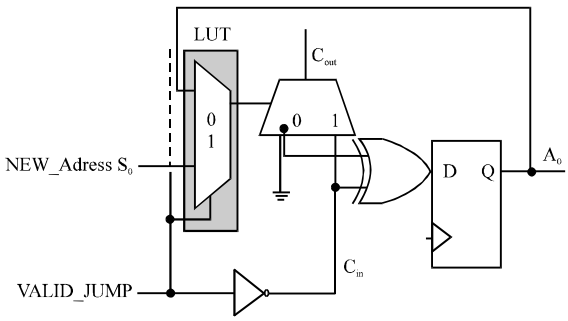

Fig. 10: Logic design using LUT

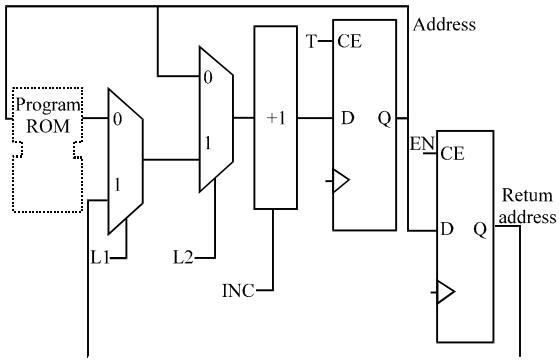

Fig. 11: Processing of the CALL command

command does not have a second operand to define a new address. Therefore, the CALL instruction must save the current PC (Program Counter) (Fig. 11).

If the register connected to the PC (Program Counter) encounters the CALL instruction, it is designed to store the address value when the RET instruction is encountered at the end of the subroutine, the address stored in the register is increased and loaded into the PC (Program Counter) again. This problem was solved by using two multiplexers.

Next, the distributed RAM was designed in single-port mode to create a stack for the program address in the processing of the program stack.When the CALL instruction is encountered, the current program address is stored on the stack and the stack counter is incremented. Conversely, the RET instruction decrements the stack counter and reads the program address stored in distributed RAM and loads it into the program counter (Fig. 12).

Next, the JP or CALL instraction in the clock determines the next clock address to be executed in the first cycle and the PC (Program Counter) is incremented at the next clock edge or a new address is loaded. Therefore, all instructions are designed to operate in two cycles as shown in Fig. 13.

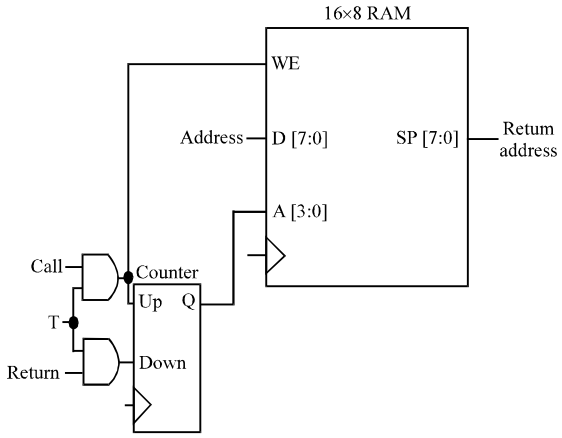

Fig. 12: Program counter stack

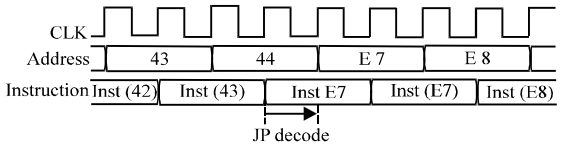

Fig. 13: Program counter stack

RESULTS AND DISSCUSSION

The interface design method of this study is divided into an external interface and an internal ECC module interface. The external interface is designed by loading the key and data using serial communication via. RS232 and returning the encrypted data. The internal ECC module interface has 3 bits allocated to the ECC module and the interface for controlling the ECC module as shown in Table 3, 8 commands are defined and the processing status of the ECC module can be monitored using 3 bits Fig. 14 and 15.

Both the key and data are processed in units of 32 bits, processed using a single data bus and decoded to

Table 3: Command definition for ECC module control

Command codes	Functions	Status codes	Functions
000	Reset key and data	000	Idle
001	Key road	001	Busy
010	Data road	101	Error
011	Perform an operation	010	Done

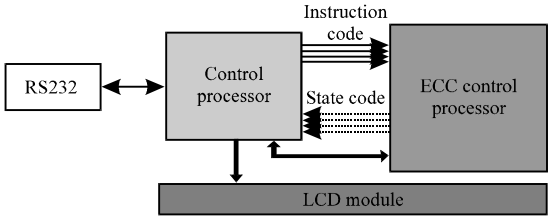

Fig. 14: Internal interface structure

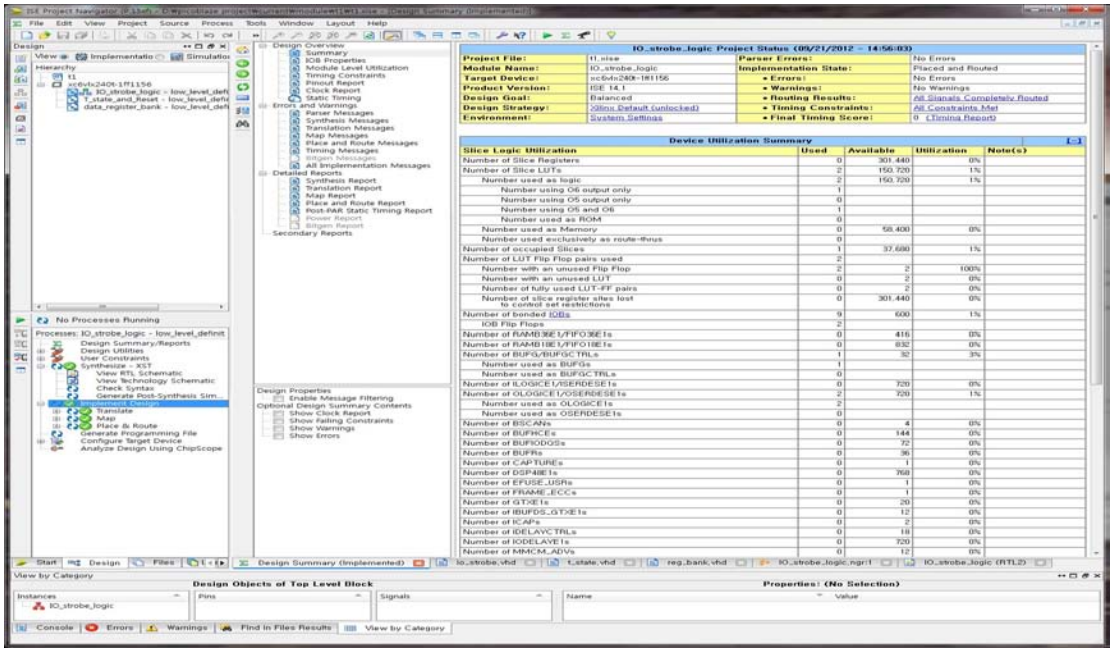

Fig. 15: Place and route result screen

define the control signal and the result signal to be processed subsequently. In this study, additional 16×2 LCD module was added to enhance the visual effects of debugging and processing results at the development stage (Fig. 15).

CONCLUSION

Since, the home network is used in various environments such as various wired and wireless networks and protocols, it may cause additional security vulnerabilities to be considered in addition to the security vulnerabilities that have occurred in the existing internet.

This study implements the elliptic curve cryptographic algorithm which corresponds to Simple Power Analysis (SPA) attack method which is one of the subchannel attacks and the public key-based cryptography required for information security and authentication of smart home appliances included in smart grid system and to design and develop the module that can perform the given task independently.

REFERENCES

Ah, S.K., 2017. [Information protection elliptic curve cryptosystem emergence]. Information and Communication Technology (ICT), Uganda, East Africa. (In Korean)

Anonymous, 1999. Recommended elliptic curves for federal government use. National Institute of Standards and Technology, Gaithersburg, Maryland.

Johnson, D. and A. Menezes, 1999. The elliptic curve digital signature algorithm. MSc Thesis, Department of C&O, University of Waterloo, Waterloo, Ontario.

Kwon, T., 2000. Authentication and key agreement via memorable password. International Association for Cryptologic Research (IACR), Mumbai, India. https://eprint.iacr.org/2000/026.pdf.

Lopez, J. and R. Dahab, 2000. Performance of elliptic curve cryptosystems. Technical Report IC-00-08, Institute of Computing, Sate University of Campinas, Brazil, May, 2000.