Tiny encryption algorithm tea computer science

Today, security is an issue concern by everyone. Many ways of implementing encryption algorithms have been investigated in order to achieve better performance in terms of security level, speed, power consumption and cost. This project will discuss about implementing Tiny Encryption Algorithm (TEA) using Field Programmable Gate Array (FPGA). FPGA are reconfigurable chips that the integrated circuit is designed meant for reconfigurable architecture. A FPGA chips is programmed using Hardware Description Language (HDL). TEA is an encryption algorithm or block cipher that consider fast, easy and used for many application. In this project, TEA will be implemented on Altera Cyclone II FPGA using Altera DE1 Board. Keyboard using PS2 or the SWITCH on the DE1 will be used as input. The output of the encryption and decryption data will be show on VGA monitor. The encrypted data will be store in memory.

Specific Objectives

In order to complete this project, there are few objectives have to be archieve.

Program the Tiny Encryption Algorithm (TEA) using verilog HDL (Hardware Description Language)

Verifying the functionality of the implementation of the encryption in FPGA

Perform simulation for timing analysis and the encryption process on the implementation of Tiny Encryption Algorithm (TEA) in FPGA

Experiment and test the project in practical

Literature ResearchCryptography

Before the modern era, security communication is the primary concern in Government and Military[2]. Security communication become more important today as a result of the increasing use of the electronic communication for many daily activities such as internet banking, online shopping. Cryptography is a practical way of conveying information securely [1]. The main aim of cryptography is to allow authorized person to receive the message correctly while preventing eavesdroppers understanding the content of the message [1]. The original message is called plaintext t[1]. Plaintext will be encrypted using certain algorithms in the secure system in order to hide the meaning[1]. The output of this reversible mathematical process is called ciphertext and the algorithm used in this process is called cipher [1]. Ciphertext can be transmitted securely because ideally eavesdroppers that access to the ciphertext won’t understand what the meaning is behind [1]. The reverse of this mathematical process is to decrypt the ciphertext back to plaintext and this only can be done by the original recipients [1]. The processes of encryption and decryption are shown in Figure 1.

Eavesdropper

Plaintext

Encryption

Ciphertext

Plaintext

Decryption

Figure 1

Encryption

There are two types of encryption or cipher depends on the key used: Asymmetric key and Symmetric key.

Symmetric key – The encryption and decryption process use the same key [1]. The major problems and drawback of this key both sender and receiver must know the key prior to the transmissions [1]. If the key is transmitted then it will compromise the system’s security [1]. The advantages of symmetric key is the process of encryption and decryption will be faster compare to asymmetric key, in another words it can encrypt or decrypt more data in shorter period of time [1].

Asymmetric key – The encryption and decryption process use different key but both of the key are related mathematically [1]. It is very hard to obtain one from the other although they are mathematically related [1]. The public key is used for the encryption process and the private key is used for the decryption process [1]. The security of the system won’t be compromised even though the public key is made available but the corresponding private key cannot be revealed to anyone [1].

Symmetric key

Symmetric key is further divided into two types: Symmetric Cipher and Block Cipher.

Stream Cipher – Stream cipher that generates a keystream (a sequence of bits used as a key) [4]. The encryption process is usually done by combining the keystream with plaintext using bitwise XOR operation [4]. Keystream that generated is independent of the plaintext and ciphertext is called synchronous stream cipher while keystream that is generated is depent of plaintext is called self-synchronizing stream cipher [4].

Block Cipher – Stream cipher that generates a keystream encrypt fixed length block of plaintext into block ciphertext that is same length [3]. The fix length is called block size. Block Cipher using same secret key for the encryption and decryption process [3]. Usually, the size of block cipher is 64 bits [3]. By increasing the size of block cipher to 128 bits will make the processors become more sophisticated [3].

Stream Cipher vs Block Cipher

Stream cipher is a type of symmetric encryption algorithm that can be designed to be exceptionally fast and even much faster compare to block cipher [4]. Stream ciphers normally process on less bits while block ciphers can process large blocks of data [4]. Plaintext that encrypted using block cipher will result in the same ciphertext when the same key is used [4]. With a stream cipher, the transformation of thse smaller plaintext units will vary depending on when they are encountered during the encryption process [4].

Stream Cipher

Block Cipher

Block Size

Depends

Fixed

Encryption/Decryption Speed

Fast

Slower

Size of block data can be process

Small

Larger

Figure 2: Comparison of Stream Cipher and Block Cipher

Figure 3 below shows different type of algorithm

table. jpgFigure 3 : Different type of encryption algorithm

Tiny Encryption Algorithm is implemented in this project because it is one type of cipher encryption algorithm that encrypt 64 bits of plaintext using a 128 bits of key into a 64 bits ciphertext.

TEA

Tiny Encryption Algorithm (TEA) is a Feistel type routine designed by David J. Wheeler and Roger M. Needham. It used addition and subtraction as the reversible operators [5]. XOR and ADD alternately used in the routine provide nonlinearity [5]. The Dual bit shifting in the routine cause all the bits

and data mixed repeatedly [5]. The three XOR, ADD and SHIFT operation will provide Shannon’s properties of diffusion and confusion necessary for a secure block cipher without the need for P-boxes and S-boxes [6]. TEA is a feistel cipher that split the plaintext into halves [7]. A sub key will be applied to the one half of plaintext in the round function, F [8]. Then the output of the F will be XOR with other half before the two halves are swapped [8]. All same patterns applied to the entire round except the last round where there is often no swap [8]. Figure 2 below show a Feistel cipher diagram where 64 bits of plaintext is divided into halves which are equally 32 bits each part. 128 bits of key is used for the encryption and decryption process and it is spitted into 32 bits subkey [7].

TEA. png

Figure 4: Two Fiestal round(one cycle) of TEA

The encryption and decryption routine of Tiny Encryption Algorithm (TEA) written in C language [5].

“

```
void encrypt (uint32_t* v, uint32_t* k, uint32_t* v1) {

uint32_t v0= v[0], sum= 0, i; /* set up */

uint32_t delta= 0x9e3779b9; /* a key schedule constant */

uint32_t k0= k[0], k1= k[1], k2= k[2], k3= k[3]; /* cache key */

for (i= 0; i < 32; i++) { /* basic cycle start */
```

```
sum += delta;

v0 += ((v1 <<4) + k0) ^ (v1 + sum) ^ ((v1>> 5) + k1);

v1 += ((v0 <<4) + k2) ^ (v0 + sum) ^ ((v0>> 5) + k3);

} /* end cycle */

v[0]= v0; v[1]= v1;

}

void decrypt (uint32_t* v, uint32_t* k, uint32_t* v1) {

uint32_t v0= v[0], sum= 0xC6EF3720, i; /* set up */

uint32_t delta= 0x9e3779b9; /* a key schedule constant */

uint32_t k0= k[0], k1= k[1], k2= k[2], k3= k[3]; /* cache key */

for (i= 0; i <32; i++) { /* basic cycle start */

v1 -= ((v0 <<4) + k2) ^ (v0 + sum) ^ ((v0>> 5) + k3);

v0 -= ((v1 <<4) + k0) ^ (v1 + sum) ^ ((v1>> 5) + k1);

sum -= delta;

} /* end cycle */

v[0]= v0; v[1]= v1;

}
```

“[5]

delta is derived from the golden number where

delta =

Architectures

Untitled. jpg

Figure 5: TEA architectures

TEA is implemented using three different architectures. The first architecture (Figure 3a) is a multiple 32 bit adders that simultaneously perform operations needed for one encryption cycle [6]. This parallel form structure should be quite large in terms of hardware area but will perform faster [6]. On the other hands, in order to reduce the area, the second architecture (Figure 3b) performs operations sequentially using a single 32 bit adder [6]. The last design (Figure 3c) is a 8 bit digit-serial adders that use advance architecture offered by application-specific hardware solution [6]. The latter two design are meant for low area solutions but in terms of control and data selection, the effectiveness remain confirmed [6].

Software vs Hardware Implementation of Encryption

Implementation of encryption using software is easier to design and upgrade, it also portable and flexible [7]. One of the major problems of software implementation is in most typical personal computer have external memory out from the processor, the external memory is used to store raw data or instruction in unencrypted form so if an attacker gain access to the

system, the key can be easier obtained [7]. One of the most common way used by the attacker is bruteforce, a special program can be easily design to bruteforce the algorithm. Besides this, reverse engineering method easier to apply on software implementation. So it can be concluded that software implementation is lack of physical security[7].

Implementation of encryption using hardware by naturally is physically more secure as they are hard to read and view by attacker [7]. Another advantage of hardware implementation is all the data in the encryption process is correlated according to an algorithm which usually perform operation on same data [7]. This will prevent computer technique such as out of order execution and cause hang to the system [7]. Hardware implementation also tend to be more parallel so more orders of magnitudes can be done at certain period of time [7].

Hardware implementation is will be better choice for encryption in terms of performance but the cost of implementation is higher compare to software implementation. Higher security level and better performance is the main concern in this project, so the encryption will be implemented on FPGA, one of the hardware implementation method.

Microcontroller, Microprocessor, DSP Processor and FPGAMicroprocessor

The first microprocessors invented in the 1970s [10]. This is the first time where such an amazing devices put a computer CPU onto a single IC [10]. The significant processing was available at rather low cost, in comparatively small space [10]. At beginning stage, all other functions, like input/output interfacing and memory were outside the microprocessor [10]. Gradually all

the other functions in embedded into a single chip [10]. At the same time, microprocessor becoming more powerful in terms on the speed, power consumption and so on [10]. Microprocessor is moving rapidly from 8 bits to 32 bits [10].

Microcontroller

A microcontroller is an inexpensive single-chip computer [9]. The entire computer system lies within the confines of the integrated circuit chip, so it is called a single chip computer [9]. The microcontroller on the encapsulated sliver of silicon has features similar to those personal computers [9]. Mainly, the microcontroller is able to store and run a program [9]. The microcontroller contains a CPU (central processing unit), ROM (random-access memory), RAM (random-access memory), Input/Output lines, and oscillator, serial and parallel ports [9]. Some more advanced microcontroller also have other built in peripherals such as A/D (analog-to-digital) converter [9].

DSP (Digital Signal Processing) Processor

DSP processor is a specialized microprocessor optimized to process digital signal [12][13]. Most of the DSP processors are commonly designed to have basic features such as high performance, repetitive and numerically intensive tasks so DSP processor often have advantage in terms of speed, cost and energy efficiency [11]. DSP processor have the avility to perform one or more multiply accumulate operations (often called “ MACs”) in a single instruction cycle [14].

FPGA (Field Programmable Gate Array)

Xilinx Co-Founders, Ross Freeman and Bernard Vonderschmitt, invented the first commercially viable field programmable gate array in 1985 – the XC2064. FPGA is integrated circuit for reconfigurable purposes by user after manufacturer. FPGA is generally specified using Hardware Description language (HDL). FPGA can be programmed to perform logic function and due to this ability, FPGA become more popular. Using FPGA for design can lower non recurring Engineering cost and apply on many application.

Hardware Architectures comparison

The figure 6 below show the comparison of different architectures used for hardware implementation on encryption.

Architecture

Efficiency

Performance

Non recurring Engineering Cost

Unit Cost

Microprocessor

Low

Low

Low

Low

Microcontroller

Low

Low

Low

Low

DSP processor

Moderate

Moderate

Low

Moderate

FPGA

High

High

Low

High

Figure 6: Architectures Comparison

Comparing the four architectures above, FPGA have the advantage in terms of the efficiency Performance but the unit cost is high. Since costing is not a major concern in this project, so FPGA is better choice for implementing Tiny Encryption Algorithm.

Altera DE1 Development and Education Board

Altera DE1 is a FPGA Development and Education Board that will be used for this project [17]. Below is the features of this board:

DE1_intro_500x. png

Figure 7: Altera DE1 Board

Altera Cyclone II 2C20 FPGA with 20000 LEs

Altera Serial Configuration deivices (EPCS4) for Cyclone II 2C20

USB Blaster built in on board for programming and user API controlling

JTAG Mode and AS Mode are supported

8Mbyte (1M x 4 x 16) SDRAM

4Mbyte Flash Memory

512Kbyte(256Kx16) SRAM

SD Card Socket

4 Push-button switches

10 DPDT switches

8 Green User LEDs

10 Red User LEDs

4 Seven-segment LED displays

50MHz oscillator , 24MHz oscillator , 27MHz oscillator and external clock sources

24-bit CD-Quality Audio CODEC with line-in, line-out, and microphone-in jacks

VGA DAC (4-bit R-2R per channel) with VGA out connector

RS-232 Transceiver and 9-pin connector

PS/2 mouse/keyboard connector

Two 40-pin Expansion Headers

DE1 Lab CD-ROM which contains many examples with source code

Size? s153*153 mm

There are few features of DE1 Board will be used for this project.

PS/2 mouse/keyboard connector

PS/2 keyboard is used as input for the plaintext

4 Push button switches

used as a reset button

VGA DAC (4-bit R-2R per channel) with VGA out connector

VGA monitor is connected to the DE1 board to show the input of plaintext and the output of the encryption, cipher text

4Mbyte Flash Memory

Used to store the ciphertext

VGA controller

IBM introduce video display standard called VGA (video graphics array) in the late 1980s that widely supported by PC graphics hardware and monitors [18].

Figure 8: Simplified Block Diagram of VGA Controller

The vga_sync circuit generates timing and synchronization signals [18]. The hsync and vsync signals are connected to the VGA port to control the horizontal and vertical scans of the monitor [18]. Two signals which are pixel_x and pixel_y are decoded from the internal counters [18]. The pixel_x and pixel_y signals indicate the relative positions of the scans and essentially specify the location of the current pixel [18]. Videl_on signal is generated from vga_sync to check whether the display is enable or disable [18]. The pixel generation circuit generate three video signal which is RGB signal [18]. The current coordinates of the pixel (pixel_x and pixel_y), external control and data signals determine the color value [18].

PS/2 Controller

IBM introduced PS2 port in personal computers [18]. It is a widely used interface for keyboard and mouse to communicate with the host [18]. PS2

port consists of two wires for communication purposes [18]. One wire for transmitting data in serial stream while another wire is for the clock information which determine when the data is valid and can be retrieved [18]. The data is transmitted in 11 bit packet that contains 8 bits of data, an odd parity bit and stop bit [18].

Figure 9: Timing Diagram of a PS/2 port

Quartus II Web Edition

Quartus II Web Edition design software is a comprehensive environment available for system-on-a-programmable-chip (SOPC) design developed by Altera [19]. This software is used in this project to program and implement the Tiny Encryption Algorithm (TEA) on Altera DE1 Cyclone II FPGA [19]. This program also can be used for the simulation and timing analysis [19].

Hardware Description Language (HDL)

Hard description language (HDL) is a type of programming languages used to program and describe digital logic or electronic circuits [20]. It can describe circuit operation, its design and organization [20]. Figure 10 below shows different type of Hardware Description Language commonly used.

HDL

Syntax Similarity

AHDL

Ada programming Language

VHDL

Ada Programming Language

JHDL

Java

Verilog

C Programming Language

Figure 10 : Different type of HDL

Verilog Hardware Description Language (HDL) is used to program the FPGA in this project because it is a widely used HDL and it syntax is similar the C programming language.

Methodology

Block Diagram

VGA Monitor

PS/2 Keyboard

VGA Controller

Plaintext

TEA Encryption Core

Flash Memory

64 Bits

Ciphertext

PS/2 Controller

Key

128 Bits

64 Bits

Encryption/Decryption

Acknowledge

Key Update Request

Busy

Asynchronous Reset

Clock

Figure 11: Core Module

The Blog Diagram above explains the design of this project. PS/2 keyboard used as input for the plaintext. All the data from the PS/2 keyboard will be sent into PS/2 controller to process. The processed data, 128 Bits or key or 64 Bits of plaintext will sent into the TEA encryption core for encryption. The output of the encryption, ciphertext will store inside the flash memory. All the plaintext and cipher text will send into VGA controller to process and

show on the CRT monitor. The encryption/decryption will be connected to the DPDT switch to switch between encryption or decryption mode. Key Update Request also connected to the DPDT switch for the purpose of updating the key when the switch is on. Asynchronous reset is connected to the push button for the reset purpose. There are internal clock inside the DE1 board so no external clock is needed for this project.

Algorithm and Implementation Verification

The original Tiny Encryption Algorithm C source code by the author will be compiled or get a compiled executable program from other source to analyze the encryption of plaintext to ciphertext and decryption of ciphertext back to plaintext. A set of plaintext, ciphertext and key can generated from the program as a reference and compare with the encryption and decryption output implemented on FPGA.

Figure 12 is an example of compiled executable program of Tiny Encryption Algorithm by Andreas Jonsson

TEA. jpg

Figure 12

Costing Estimation

Components

Quantity

Price

Altera De1 Board [17]

1

RM 512. 84

Used 15? Samsung SyncMaster CRT monitor

1

RM50. 00

Used PS/2 Keyboard

1

RM10. 00

Total

RM572. 84

Gantt Chart

ganchart. jpg

Research analysis will be start from week 6 till week 8. Verilog coding on the implementation of TEA and module and test bench verification this 2 task must perform parallel because after finish a certain module, it should be test and simulate. If simulation or test is done after finish the whole coding, there will be a big problem in debugging the error. The synthesis of PS/2 keyboard, VGA monitor and FPGA start week 20 just before finish the coding. The

functionality verification task also runs parallel with the synthesis optimization task.

References and FiguresFigures

Figure 4: Tiny Encryption Algorithm . Available at:

http://en. wikipedia. org/wiki/Tiny_Encryption_Algorithm (Accessed: 30 October 2009)

Figure 5: Israsena. P, ‘ Design and Implementation of Low Power Hardware Encryption for Low Cost Secure RFID Using TEA’ . Information, Communications and Signal Processing, 2005 Fifth International Conference on 0-0 0 Page(s): 1402 – 1406, DOI 10. 1109/ICICS. 2005. 1689288. Available at

http://ieeexplore. ieee. org/stamp/stamp. jsp? tp=&arnumber= 1689288&isnumber= 35625 (Accessed : 26 October 2009)

Figure 7: Available at:

http://www. terasic. com. tw/cgi-bin/page/archive. pl? Language= English&No= 83

(Accessed : 28 October 2009)

Figure 8: Pong P. Chu (2008) FPGA Prototyping by Verilog Examples : John Wiley & Sons

Figure 9: Pong P. Chu (2008) FPGA Prototyping by Verilog Examples : John Wiley & Sons