IMPLEMENTATION OF SECURED HASH ALGORITHM -1 ON VERILOG

A. RAVICHANDRA, G.MANI KUMAR & N.SURESH BABU
Department of Electronics and Communication Engineering, Chirala Engineering College, Chirala, India

ABSTRACT

A hash (also called a hash code, digest, or message digest) can be thought of as the digital fingerprint of a piece of data. You can easily generate a fixed length hash for any text string using a one-way mathematical process. It is next to impossible to (efficiently) recover the original text from a hash alone. It is also vastly unlikely that any different text string will give you an identical hash - a 'hash collision'. These properties make hashes ideally suited for storing your application's passwords. Although an attacker may compromise a part of your system and reveal your list of password hashes, they can't determine from the hashes alone what the real passwords are. A cryptographic hash function is a deterministic procedure that takes an arbitrary block of data and returns a fixed-size bit string, the (cryptographic) hash value, such that an accidental or intentional change to the data will change the hash value. The data to be encoded is often called the "message," and the hash values are sometimes called the message digest or simply digest.

KEYWORDS: Secure Hash Algorithm, Cryptography, One way Encryption

INTRODUCTION

Increasingly, the world is turning to portable electronic devices with cryptographic capabilities to perform very secure network authentication, virtual private networking, vending system regulation, fare collection, and employee or citizen identification in digital electronics. The design of such a portable device (or token) requires careful study of the methods by which cryptographers authenticate and protect sensitive data and monetary information. Of course, an electronic or identification token must be portable, durable, and secure, but it quickly becomes obvious that there are several additional cryptographic, electrical, and physical requirements that a secure token must satisfy.

Nowadays, network is not only a way for us to get more information. It has already become a new lifestyle for people. For example, network bank, net-shopping, online chat, e-government, etc. All these need a high security network.

So networks security has become a very important problem in information age [1]. In cryptographic circles, truly secure authentication is handled by a method called challenge-and-response. This method involves a secret that is known only to valid tokens and hosts, and methods whereby tokens can prove that they know the secret, which proves that they are authentic. Of course, the secret must never be revealed in the process, and that is where a cryptographic concept called Zero Knowledge Proof comes into play.

The token must support a mechanism where it can prove that it knows a secret without revealing any information about it. At first glance, this may seem impossible, but it is a common exercise in secure cryptographic systems [4].

The most critical function for a truly secure token is that of authentication. A token must be able to prove that it is authorized by the issuer (the service provider), and that it is authentic. A token can prove that it is authorized simply by virtue of the fact that it contains encrypted information that only the issuing authority could have created.

Secure authentication, however, is a far more difficult issue. The device must be able to prove that it is not a fake or duplicate, and this requires some very special hardware functionality.
A device will not be trusted if its design cannot be freely examined, so secrets in design or function are all but impossible to keep. If we assume that the inner workings of the device are published and widely known, then we must also assume that anyone with sufficient skill could build an emulation of the device and bypass some of the special controls that the device hardware imposes. Any token design based on security by obscurity is doomed to failure. In this paper, message compress standard SHA-1 (Secure Hash Algorithm) will be implemented by FPGA (Field Programmable Gate Array) to cooperate the DSS. And it will be used in DSA (Digital Signature Arithmetic) that implemented by FPGA.

The ideal cryptographic hash function has four main or significant properties:

1. it is easy to compute the hash value for any given message,
2. it is infeasible to find a message that has a given hash,
3. it is infeasible to modify a message without hash being changed,
4. it is infeasible to find two different messages with the same hash.

Cryptographic hash functions have many information security applications, notably in digital signatures, message authentication codes (MACs), and other forms of authentication.

They can also be used as ordinary hash functions, to index data in hash tables, for fingerprinting, to detect duplicate data or uniquely identify files, and as checksums to detect accidental data corruption. Indeed, in information security contexts, cryptographic hash values are sometimes called (digital) fingerprints, checksums, or just hash values, even though all these terms stand for functions with rather different properties and purposes.

SHA-1 can be used in a variety of applications:

1. Security applications that require authentication
2. E-mail
3. Electronic funds transfer
4. Software distribution
5. Data storage

ARCHITECTURE OF THE SYSTEM

The above SHA-1 algorithm is implemented as an array of eighty 32-bit words. This is efficient from the standpoint of minimization of execution time. However, if the space is at a premium such as in the embedded consumer electronic system, an alternative is to implement it as a serial pattern.

Hash function is an important part of many crypto algorithms; there are 3 famous Hash Algorithm, SHA, MD (Message Digest) and RIPEMD-160 message compress algorithm[5]. In comparison, the security of SHA-1 is better than MD and RIPEMD-160.

SHA (Secure Hash Algorithm) is designed by National Security Agency of the U.S.A [5]. It is a message compress standard is used to cooperate DSS (Digital Signature Standard) that designed by NIST (National Institute of Standards and Technology). Though SHA is designed for DSS, it can be also used in many protocols or secure algorithm[1]. The original version of SHA is called SHA or SHA-0.

SHA-1 is the improved version of SHA-0. Using SHA-1, a message which is no longer than bit can be generated a 160bit message abstract. Message abstract is much shorter than the message itself, so it will spend less time to generate a digital signature.
The more important is that the digital signature generate by message abstract has the same security as generate by message.[1]

SHA-1 DESIGN AND SYNTHESIS ANALYSIS

Message compress standard SHA is designed for DSS. The input of SHA is a message which is no longer than 264 bit, and it can generate a 160 bit message abstract[2]. If a message no longer than 264 bit, it needs to be added zeros to make the message become a 264 bit one. And if a message longer than 264 bit, it need to be separated into several groups. Every group contains 264 bit. Then the message groups will be converted into message abstract groups by SHA algorithm.[2,3] When message abstract is generated, five 32 bit initial values A, B, C, D, E will be used.

\[ \begin{align*}
A &= 0x67452301 \\
B &= 0xefc8db89 \\
C &= 0x98badcfe \\
D &= 0x10325476 \\
E &= 0x85d231f9 
\end{align*} \]

Every time SHA-1 operates, non-linear function \( F_t \), constant \( W_t \) and \( K_t \) are different if \( t \) is different value.

According to parameter \( t \), the non-linear function \( F_t \) is

\[ \begin{align*}
F_t(\overline{x,y,z}) &= (x \land y) \lor (\overline{x} \land z) \quad (t=0-19) \\
F_t(x,y,z) &= x \lor y \lor z \quad (t=20-39) \\
F_t(\overline{x,y,z}) &= (x \land y) \lor (x \land z) \lor (y \land z) \quad (t=40-59) \\
F_t(x,y,z) &= x \land y \land z \quad (t=60-79) 
\end{align*} \]

The symbol ‘ \( \land \) ’ means and logic. ‘ \( \lor \) ’ means or logic. means the opposite number of \( x \). Constant \( K_t \) is different according to parameter \( t \).

\[ \begin{align*}
K_t &= 0x5a82799 \quad (t=0-19) \\
K_t &= 0x6e19eba1 \quad (t=20-39) \\
K_t &= 0x8fb2b3d \quad (t=40-59) 
\end{align*} \]

Fig.1 Show SHA-1 Operated Flowchart in One Time.

According to the flowchart, the SHA-1 arithmetic can be list as follow.

The SHA-1 message abstract module can be designed according to the algorithm. The SHA-1 message abstract \( H(m) \) is a 160 bit abstract[3] combined with \( a_n, b_n, c_n, d_n, e_n \).
SHA-1 arithmetic:
Input: message \( m \) (\( m \) is the message no longer than \( 2^{64} \) bit)
Output: SHA-1 message abstract \( H(m) \)

S1. According to the rule, we can find the right \( W_t, K_t, K_0, K_3 \), when \( t \) are different.

S2. assign the initial value:
\[ a_0 = A, \quad b_0 = B, \quad c_0 = C, \quad d_0 = D, \quad e_0 = E. \]

S3. While \( t \) from 1 to 79, loop the follow step S3.1

S3.1. \( e = F_t(x, y, z) + W_t + K_t + e_{t-1} \) \((\text{if } t < 5)\)
\[ a_t = e_t, \quad b_t = a_{t-1}, \quad c_t = \text{if } (b_{t-3} \neq 0), \quad d_t = c_{t-1}, \quad e_t = d_{t-1} \]

Return: SHA-1 message abstract \( H(m) \)

SHA-1 MODULE DESIGN FLOW

There are two ways to design a system in FPGA or HDL[3]. One is top-to-down, another is down-to-top. Top-to-
down designs the system or module first, then design the subsystem, sub-module or gate circuit. Down-to-top is the
opposite way. It design gate circuit first, then design the subsystem or sub-module, and design the system or module at last.
Top-to-down is used in this design.[4,5,6] Fig.3 is SHA-1 module pin assignment diagram.

There are 6 input signals and 3 output signals in this module. The input signals are clock(clk), reset(rst),
enable(en), ready(rdy), message-in(msgin), message-in control(msg_in). And the output signals are sha-out, sha-out
control(sha_out), finish(fnsh). All the control signals are high level enable.

There are 4 sub-modules in SHA-1 module, input/output sub-module, memory sub-module, message abstract
operate sub-module[3].

Fig. 3: SHA-1 Module Pin Assignment Diagram
Input/output sub-module controls the data in and out. Memory sub-module just like the cache in CPU, it is used to store the temporary data.

Message abstract operate sub-module is the core in SHA-1 module, it is used to carry out the SHA-1 arithmetic.

Besides, the bus of the module is set 64 bits, so before the long message is inputted into the module, it should be separated into several groups. Input signals msg_in can know the number of the groups.

The block diagram of the SHA-1 processor is as shown in Fig.5.

The simulation results of SHA-1 are as shown in Fig.6. It shows us the timing and the function of the module. The clock period is 20.0ns. The message abstract which is no longer than $2^{64}$ bit can be generated in about 100ms.
CONCLUSIONS

An implementation of SHA-1 hash function is proposed in this paper. SHA is a famous message compress standard used in computer cryptography. Its improved version SHA-1 algorithm has been analyzed in this paper, and implied by HDL. The design was captured using VHDL hardware description language and implemented on Xilinx FPGA. The correctness of the functionality has been verified using simulation tools and the test vectors.
ACKNOWLEDGEMENTS
The authors would like to thank the anonymous reviewers for their comments which were very helpful in improving the quality and presentation of this paper.

REFERENCES


AUTHORS PROFILE

Aleti Ravichandra is pursuing M.Tech in VLSI &ES at Chirala Engineering College, Chirala.

G. Mani Kumar, M.Tech in Communications and signal processing from Bapatla engineering college & he is working as Assoc. prof in ECE Dept. CEC, Chirala. He has 4 years of teaching experience.

Prof. N. Suresh Babu is Vice-Principal & HOD of ECE Dept in CEC, Chirala. He got his M.Tech in Microwave Engineering from Birla Institute of Technology, Ranchi. He has 13 years of teaching experience and 2 years of industrial experience in various organisations.