FPGA Design and Implementation of Data Covering Based on MD5 Algorithm

Dr. Thamir R. Saeed
Electrical Engineering Department, University of Technology/Baghdad
Email:thaalwaily@gmail.com

Received on:4/4/2016 & Accepted on:29/12/2016

ABSTRACT

The protection of information leads to protection of individual privacy for everyone. This protection is performed using encryption. Many types of encryption may be utilized while the simplest one is the covering of information. In this paper, three novel algorithms have been presented for covering the information and for increasing the security. The degree of these security algorithms depends on three keys; MD5 core code, MD5 iteration, and mode of data hashing. The strengths of this work are the simplicity of the design and taken a long time for attack recovering the hashing data. Where, the sequence length for our proposed algorithms related to the MD5 output sequence length will be increased from 100% for 8-bit core code to 256% for scenarios E and F, while 1024% for scenario G with processing time is 20 nsec and 60 nsec depend on the scenarios. While the bit rate of the information data in transmitted data stream bits are different from 64-to- 32 bits for each transmitted stream bits depending on the scenario that selected. In this context, the maximum expected throughput is 806.596 Mbps. The implementation of algorithm circuits is built by using Xilinx Spartan 3-xc3s1400a-4fg484.
Keywords: VHDL, FPGA-Spartan-3, Hashing data, and MD5.

INTRODUCTION

The hide and distort, of data transformation, must be fulfilled in the communication system for considering it as a secured, with authentication as an associated factor [1]. Therefore, there are three types of strategies for security, cryptography, steganography, and covering information. Each strategy has many algorithms for providing high security, and each of them must satisfy some properties as [2,3].

One of these algorithms that have been widely used in cryptography protocols and internet communication is called Message Digest (MD)[4,5]. It is generating a unique message digest for the arbitrary message, while any change in the original data will reflect on the hash value of the respective data[6,7], where, it is named hash algorithm, and the most popular variants are the MD5 [8,9]. The message digest, which has been generated by MD5 algorithm has the following properties; irrevocable, non-counterfeit, and represent a fingerprint of the message. In this context, MD5 have the following specification: easy to compute, The computation of the message from the digest is very hard, and the similarity of the digest for the different message is impossible. Therefore, it provides a unique relationship between the input and the hash value [7,8,10].

In practical work, implementation is necessary and represented a vital point in the proposed algorithms. While, FPGA technology presents an excellent tool for modern digital circuits design. This technology has provided simple implementation and high reliability. It is so flexible that its library can be extended to include new circuits by using the available options [11,12].

In the present, many important related studies dealt with the data security have been reported.

Ref [1] has been presented a protocol that provides the primitives of the cryptography; these are; confidentiality and authentication. Where, it is based on Dual-RSA algorithm, Elliptic Curve Cryptography, and Message Digest MD5. The message digest from MD5 algorithm has

https://doi.org/10.30684/etj.34.14A.7
2412-0758/University of Technology-Iraq, Baghdad, Iraq

been calculated in [13] and then, regenerated the message digest at the receiver side to verify by comparing the two digest values.

In previous work [4] many covering algorithms in four scenarios have been presented to increase the degree of security. Where the use of MD5 has been served to increase the sequence length and the complexity that are used for covering information. In the present work, a novel usage of the MD5 algorithm will be presented for data or information covering, where, it can be considered as expanding of the previous work in [4]. Three scenario's algorithms have been presented, the information's bits rate in the transmitted bit stream and the combination sequences which mean the complexity are the main difference between these scenario's algorithms. The implementation has been carried out using Xilinx Spartan-3A DSP 3400A Platform [14].

The proposed Algorithm

The proposed approaches to data secrecy are based on the MD5 core algorithm that is shown in figure (1). Where, the output of each iteration (or final iteration) is used as a set of successive initial key code for covering the required information (data or message). The formula for the message(information) which can be used is [5];

$$\textit{Message (information)} = \big((a \to z) + (A \quad Z) + (! \quad ;)\big) \qquad (1)$$

The message digest (MD) should be a string of fixed length (128 bit).

$$\textit{Message Digest (MD)} = \big((0 \quad 9) + (A \quad F)\big) \qquad (2)$$

Where

n- MD5 iteration (1-64) and related to the input code.

m- some bits that will be representing the code.

The proposed work is based on the following requirements as shown in the Figure (2) for both transmitter and receiver sides:

- Store numbers of key codes (*K1*), up to 256 (can be stored more) 8-bit data, in the RAM memory, where it is representing the MD5 core Key code.
- After 64 iterations the MD5 core output is 128-bit. Also, every iteration MD5 has 128-bit as the temporary output.
- By using the MD5 output (final iteration), or (MD5 temporary output for any selected iteration), can make the required process on the information (according to the suitable scenario). Then transmits it in the certain stream as in the below scenarios.

Three scenarios represent the primary of the proposed work for covering and recovering sides, and this work can be considered as improve and extended to the previous work that has been presented in [4]. The scenarios are;

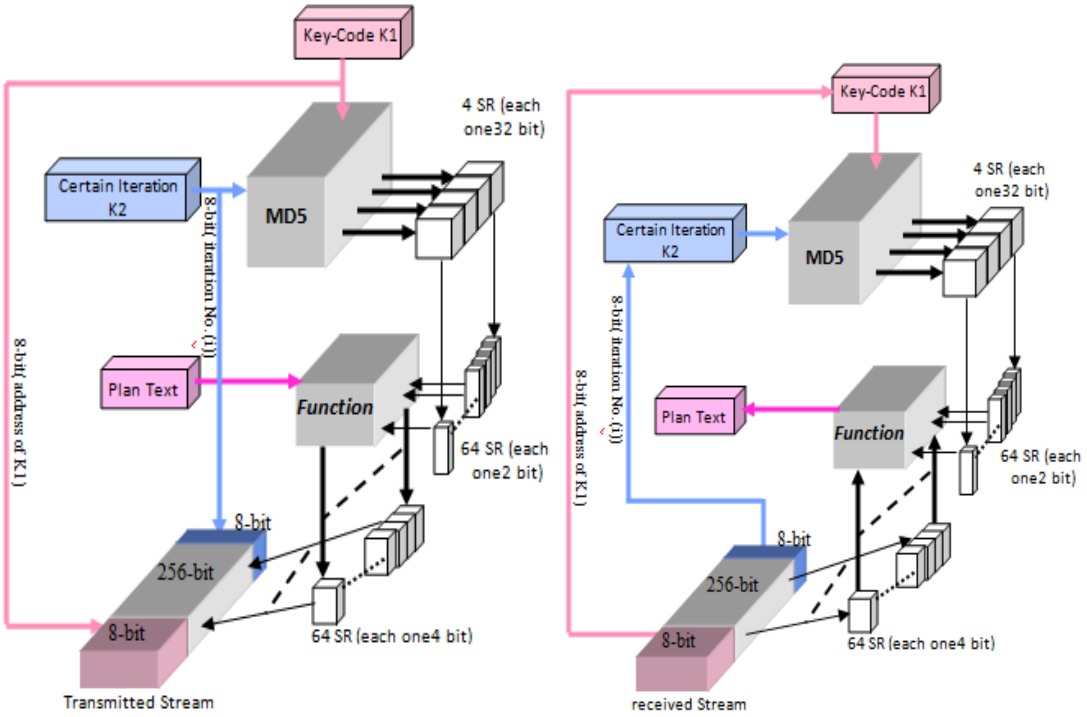
```
MD5 core
AA:=A;BB:=B;CC:=C;DD:=D;
foriin 0 to 63 loop
if(i>=0) and (i<=15)then
         F:=((B and C)or((not B) and D));
elsif(i>=16) and (i<=31)then
         F:=((B and D)or(Band (not D)));
elsif(i>=32) and (i<=47)then
         F:=(B xor C xor D);
elsif(i>=48) and (i<=63)then
         F:=(Cxor(B and (not D)));
endif;
      TS(31-S(i)downto 0):=T(i)(31 downto S(i));
      TS(31 downto (31-S(i)+1)):=T(i)((S(i)-1) downto 0);
      A:=B+((A+F)+X(K(i))+TS);
      D:=C;C:=B;B:=A;A:=D;
ifi=IT then
        At:=A+AA;Bt:=B+BB;Ct:=C+CC;Dt:=D+DD;
Git(31 downto0):=At;
        Git(63 downto32):=Bt;
        Git(95 downto64):=Ct;
        Git(127 downto96):=Dt;
endif;
endloop;
    A:=A+AA;B:=B+BB;C:=C+CC;D:=D+DD;
    G(31 downto0):=A;
    G(63 downto32):=B;
    G(95 downto64):=C;
    G(127 downto96):=D;
XOR Process to covering the information
    GREG:=DREG xorGit;
TxStream arrange
    DATA(127 downto0)<=GREG;
    DATA(135 downto128)<=CONV_STD_LOGIC_VECTOR(IT,8);
    DATA(143 downto136)<=CONV_STD_LOGIC_VECTOR(BLIind,8);
End Behavioral;
```

a) Tx Side VHDL Prog.

```
Rx Stream Arrange
   DATA:in std_logic_vector(143 downto 0);
   MSG:out std_logic_vector(127 downto 0)
variableG,Git:std_logic_vector(127 downto 0);
variableX:y3;
BLIind:=CONV_INTEGER(DATA(143 downto 136));
   IT:=CONV_INTEGER(DATA(135 downto 128));
   DREG:=DATA(127 downto0);
   BL(L-1 downto0):=CONV_STD_LOGIC_VECTOR(BLI(BLIind),32);
   BL(L):='1';
   BL(n downto (n-63)):=CONV_STD_LOGIC_VECTOR(L,64);
foriin 0 to 15 loop
     X(i):= BL((i*32)+31 downto (i*32));
end loop;
Same MD5 Core
XOR Process Extract the information
135 GREG:=DREG xor Git;
136 MSG(127 downto 0)<=GREG;
```

b) Rx Side VHDL Prog.

Figure 1 Tx/Rx VHDL core Program For proposed algorithm



(a)Tx Side (b) Rx side

Figure 2 Proposed algorithm in both transmitter (Tx) and receiver (Rx) sides

2.1 Scenario E (Location Scenario LS) or (SE)

Figure 3a is described in this scenario's algorithm in the cover side, while the algorithm's operation is as follows;

- Divide the information data into blocks each one has 64-bits.
- Prepare 32 shift register; each one has 4-bit.
- K3 of this scenario is

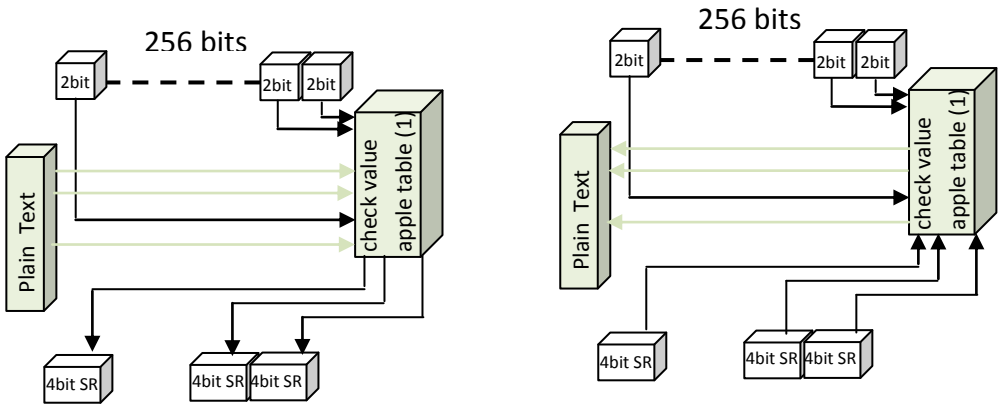
$$K3 = f(PT, K2(i)) \quad (3)$$

Where; PT is plain text.

f (PT,$K2$(i))- is a function of hashing the plaintext (information or message).

$K2(i)$ - MD5 output according to the iteration (i)

.



(a) Scenario's function at transmitter side (b) Scenario's function at receiver side

Figure 3 Scenario's function according to Table (1)

The action of this function { f (PT,$K2$(i))} is as follow; Each value of two bits of the output of certain MD5 iteration output $K2$(i), represent the location of corresponding one bit of the information data in the 4-bit shift register as in Table 1.

Table1 One information bit location in 4-bit SR

value of two bits from MD5 output	Location number of the 4-bit SR that the information will be put
00	0
01	1
10	2
11	3

- Then put the content of the 4-bit shift register (SR) in the transmitted stream form with a code key $K1$ and MD5 iteration number $K2(i)$.
- Figure 3b represents the recovering side of the scenario's algorithm. The process of this side is reversed of the covering process, where it started from final stream of the covering side to the extract the information, where, the K3 will be;

$$K3 = f\,(K2(i),\mathrm{SI}) \qquad (4)$$

Where;

SI- spreading information (covered information).

Figure (4) represent the FPGA output waveform, where figure (4a) represents the output stream of covering side while Figure 4b represents the waveform of the recovering side.

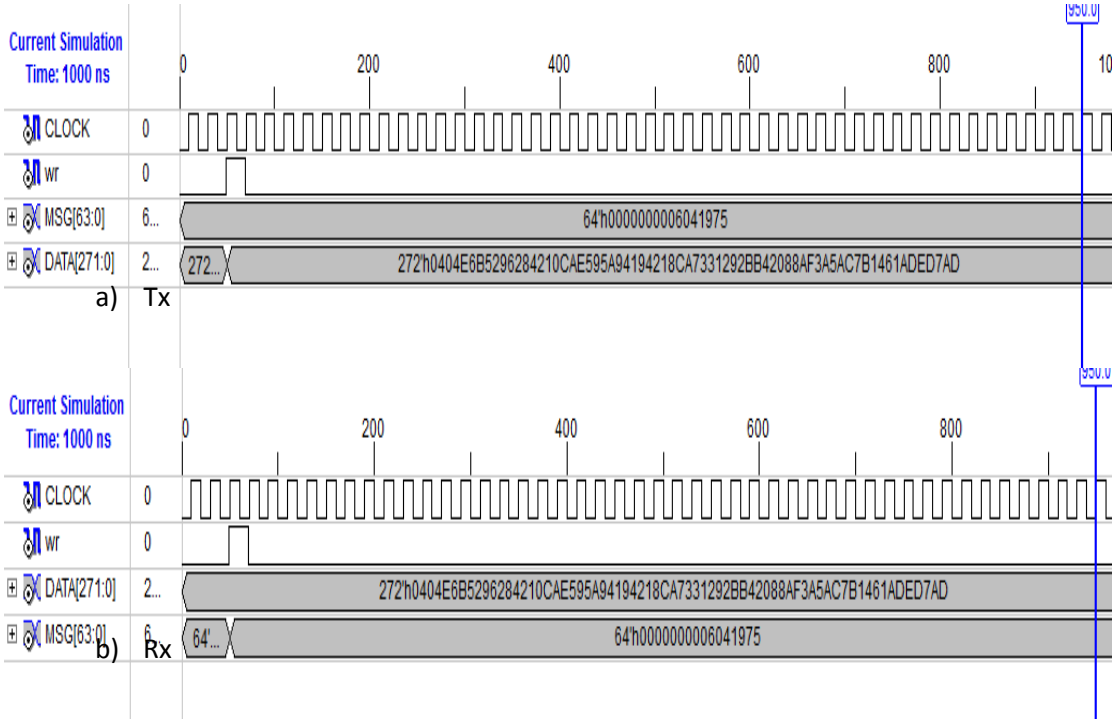


Figure 4 Covering/recovering Location scenario a) covering b) recovering

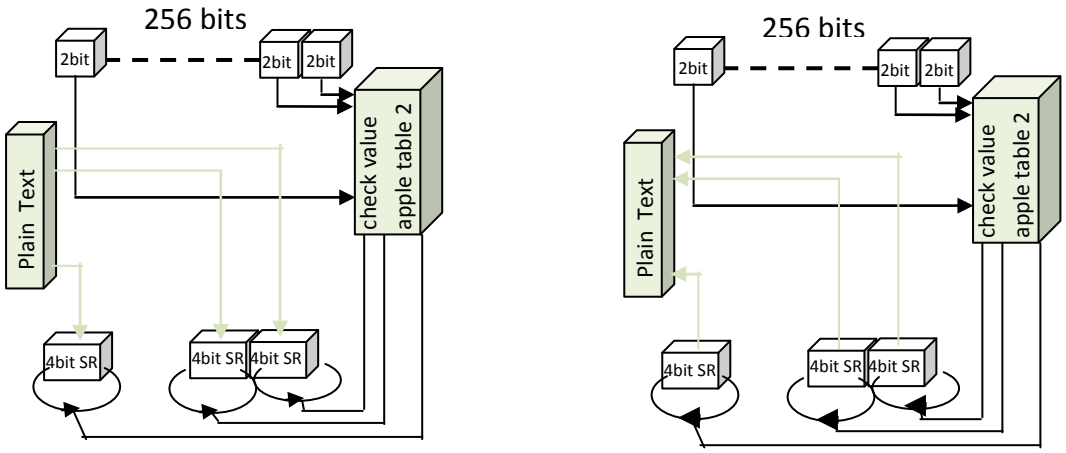
Scenario F (Rotation Scenario RS) or (SF):

The algorithm block diagram of this scenario is shown in Figure 5, where, Figure 5a represents the covering process while Figure 5b represents the recovering process. The algorithm's description is as follows;

- Divide the information data into a block each one with 64-bits.
- Prepare 32 shift register, each one has 4-bit and filling it with random data.
- *K3* of this scenario is;

$$K3 = f\,(\mathrm{PT}, K2(i)) \qquad (5)$$

The action of this function of that scenario is as follows; Put each bit of the information data in the first bit(LSB) in each shift register. Then, each two bits of the output of certain MD5 iteration K2(i), which corresponding to one bit of the information data, have been used to rotate the contain of the shift register by their value as in Table 2.



(a) Scenario's function at transmitter side (b) Scenario's function at receiver side

Figure 5 Scenario's function according to Table (2)

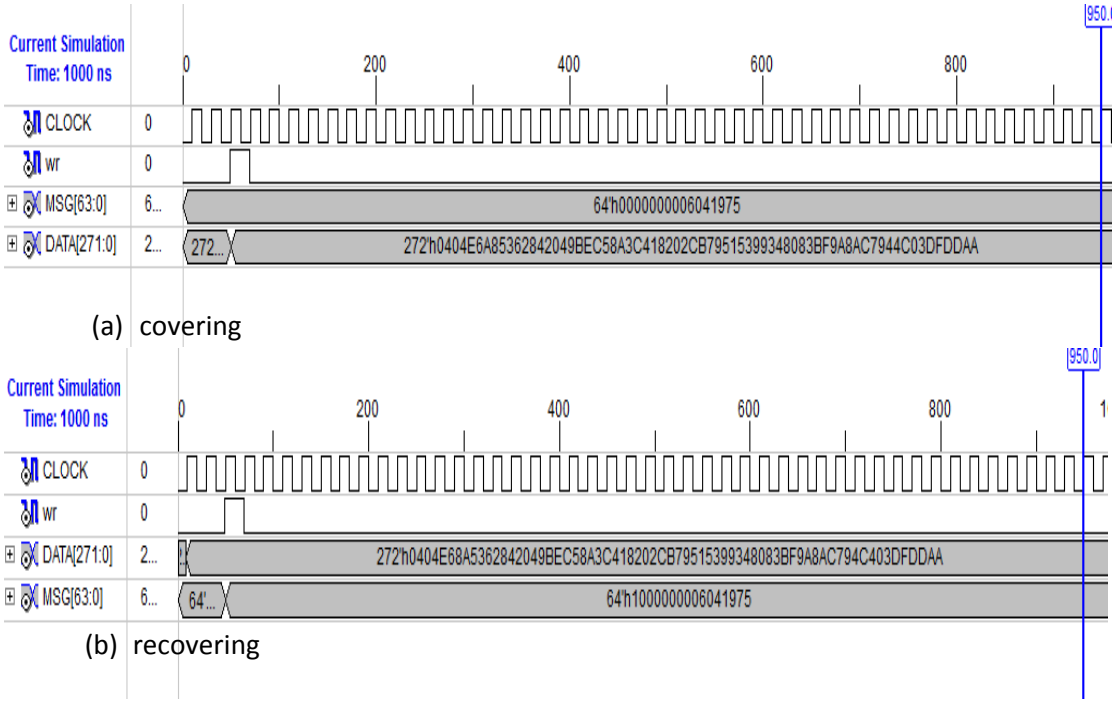
Table 2 One information bit rotates in 4-bit SR

MD5 two bits	Number of shift times of one-bit information in 4-bit SR
00	- (no shift)
01	> (one bit shift)
10	>> (two bit shift)
11	>>> (three bit shift)

Then put the content of the 4-bit shift registers (SRs) in the transmitted stream form with a key code *K1* number of MD5 and iteration number *K2*(i) for each stream. The recovering process is in Figure 9b, where, the *K3* is as in equation (6), and its process is the reverse the covering process.

$$K3 = f\,(\mathrm{SI}, K2(i)) \qquad (6)$$

Also, Figure (6) represent the FPGA output's waveform of this scenario. Where Figure 6a represents the stream code for covering process, while the recovering the information of this scenario is shown in Figure 6b.



Figurer 6 Covering/recovering rotation scenario code stream; a) covering b) recovering

Scenario G (Location Rotation Scenario LRS) or (SG):

The algorithm flowchart of this scenario is shown in Figure 7, and it represents the combination of the above two scenarios. The description of this scenario is as follows; The covering side algorithm flowchart is as in Figure 7a, where, its operation as follows;

- Divide the information data into blocks each one has 32-bits.
- Prepare 32 shift registers each one has 4-bit and then filling it with random data.
- *K3* of this scenario is;

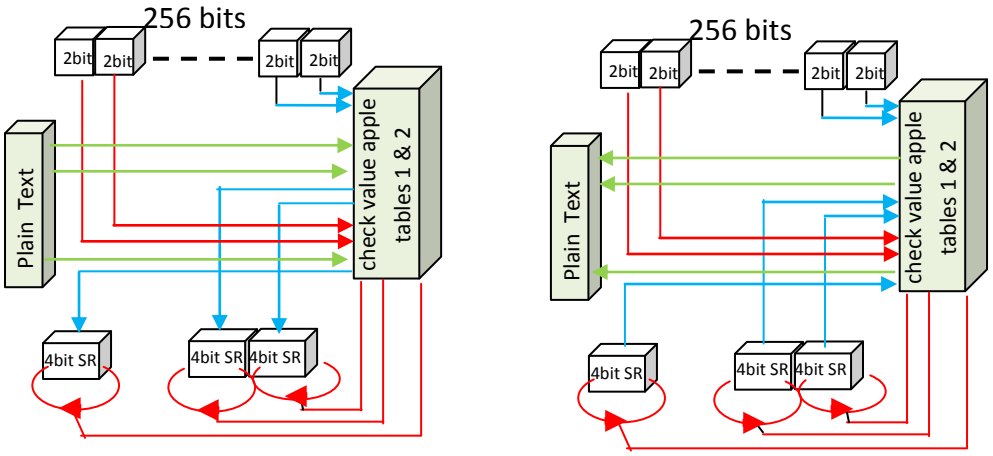
$$K3 = f\,(\mathrm{PT}, K2(i)) \qquad (7)$$

Where;

The action of this function is as follows; each two bits from least side of the certain MD5 output iteration output K2(in), which corresponding to one bit of the information data, have been used to determination the location of an information bit in the shift register that will be placed in it. Then, the content of that shift register will be rotated by the value of most two bit from the same MD5 iteration output. Then put the content of the 32 (4-bit) shift registers (SR) in the transmitted stream form with a key code (K1) number of MD5 and iteration number (*K2*(i)) for each stream.

- In the recovering side as in Figure 7b reverse process will be made for recovering the information and *K3* will be as;

$$K3 = f\,(\mathrm{SI}, K2(i)) \qquad (8)$$



(a) Scenario's function at transmitter side (b) Scenario's function at receiver side

Figure 7 Scenario's function according to Table (1) and (2)

Figure 8 Represent the FPGA output waveform of this scenario, where Figure 8a is a code stream for covering side while the recovering information stream is as in Figure 8b.

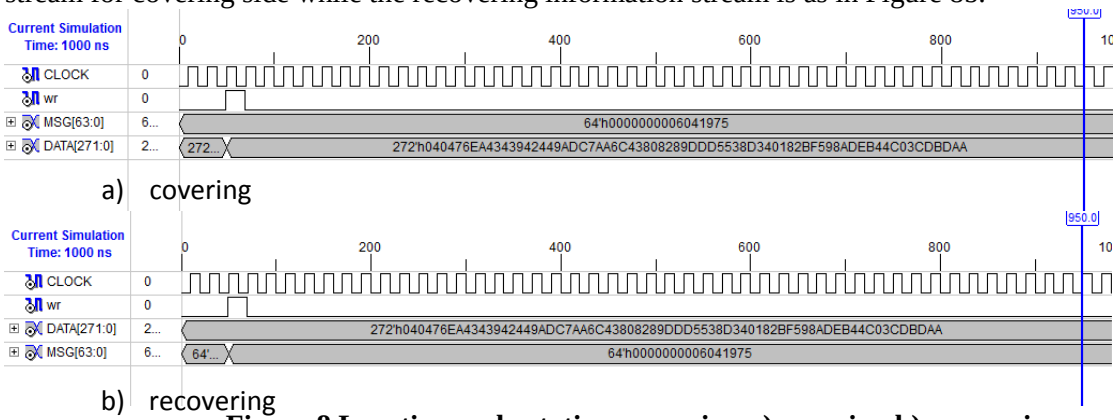


Figure 8 Location and rotation scenario; a) covering b) recovering

This algorithm has tested for many cases of Key-code K1 and MD5 output K2 for different iterations *i*. The encouraging results have been gained and without error. The bit rate and processing time with the number of combination sequences (complexity) gained are shown in the Table (3). From this table, it is clear that the number of information bits in the transmitted bit stream (bit rate) is inversely proportional to the number of combination sequences and these depends on the scenarios algorithms.

Table 3 Proposed scenarios specification

Scenario		Combination	% with respect to MD5 (MD5 combination = 265)	Percentage of information bits (bit rate) in transmitted (128 bit) bit stream	Process time for covering information	Number of occupied slices
A	**Ref. 4**	256 [**Ref. 15**]	Equal	100%	20 nsec	22 % [66%]
B		16384	64% over	100%	20 nsec	22%
C		16384	64% over	100%	20 nsec	22%
D		49152	192% over	100%	60 nsec	25%
E		65536	256% over	50%	20 nsec	28%
F		65536	256% over	50%	20 nsec	28%
G		262144	1024% over	25%	20 nsec	31%

Performance Evaluation

Expected Throughput

The comparison of the presents work with previous work from the point of throughput is shown in Table (4)

Table (4) throughput comparison between present and previous work

Ref.	Scenario	Expected Throughput
4,15	A	0.787 Mbps
4	B,C,D	50.41 Mbps
9	A	165Mbps
	B	354 Mbps
Present work	E, F	201.649 Mbps
	G	806.596 Mbps

Attack Evaluation

For attacking this scenario's algorithm, it must be trying many times, and this takes a long time. The number of attempts for each scenario to attack it is;

If $K1<448$

MD5 output = 128^A

First scenario = $128^A * 4 * 64 * t_{oi}$

Second scenario = $128^A * 4 * 64 * t_{oi}$

Third scenario = $128^A * 4 * 4 * 32 * t_{oi}$

$A = 2^{448} * 64$

Where; 448- length input code possible.

64 – code value length bit.

t_{oi} – a time for MD5 one iteration.

This calculation for MD5 output, while when to take the MD5 output from individual iteration, and then the above calculation must be multiply by the number of iteration. Therefore, the required time represents the above calculations multiply by the processor time that is used.

CONCLUSION

The use of MD5 was supported to satisfy many of properties of security mechanism, such as confidentiality (privacy), authenticity, and dynamical data. The bit rate (number of information bits) in the transmitted stream are from 201.649 Mbps to 806.596 Mbps, while, the processing time are from 20 to 60 nsec depending on the selected scenario. Also, the combination sequences longer (complexities) are from 65536 to 262144 and has been improved four times of the previous work as in ref. [4], with reducing in information bit rate. These differences in combination and time depend on the selected scenario's algorithms. The attack requirements are described, and it takes a long time. The use of Xilinx technology which implementing all scenarios algorithm has given the strength of the design because this technology has many advantages as; a flexibility of changing or improving the design, speed of operation and low cast. The hashing or covering information based on MD5 and implementation by Xilinx has been given important two factors; the simplicity and ability to improving.

REFERENCES

[1] Subasree S.,Sakthivel K.Design a New Security Protocol Using Hybrid Cryptography Algorithm. IJRRAS, Vol. 2 , No. 2, 2010.

[2] Sivaraman V., Ostry D., Shaheen J., Hianto A., Jha S. Broadcast Secrecy via Key-Chain-Based Encryption in Single-Hop Wireless Sensor Networks. EURASIP Journal on Wireless Communications and Networking, 2011.

[3] Wolf M. , Weimerskirch A., Wollinger T. State of the Art: Embedding Security in Vehicles. EURASIP Journal on Embedded Systems, 2007.

[4] Thamir R. Saeed, Hashmea S. Dakel, Najmah A. beeb, Ivan A. Hashim and Prof. Jawad K. Ali, " VHDL Processor for Covering Information Using MD5 Algorithm I", Journal: The islamic college university journal/ Najaf, ISSN: 62081997, Year: 2014 Issue: 29 Pages: 9-25.

[5] Dongjing H. , Zhi X.Multi-parallel Architecture for MD5 Implementations on FPGA with Gigabit-level Throughput. 2010 International Symposium on Intelligence Information Processing and Trusted Computing.

[6] M. Thangavel, P. Varalakshmi, and M. Kuthalingam, "A Novel Secured Cryptographic Hash (NSCH) Algorithm", International Journal of Engineering Research & Technology (IJERT), Vol. 3 Issue 3, March – 2014.

[7] Shadab Ahmad Khan, " FPGA Implementation of MD5 Algorithm for Password Storage", International Journal of Science and Research (IJSR), Volume 4 Issue 6, June 2015.

[8] Kimmo J., Matti T. , Jorma S. Hardware Implementation Analysis of the MD5 Hash Algorithm. Proceedings of the 38th Hawaii International Conference on System Sciences, 2005.

[9] J. Deepakumara; H. M. Heys; R. Venkatesan, " FPGA implementation of MD5 hash algorithm", IEEE Conference Publications, Canadian, Year: 2001, Volume: 2, Pages: 919 – 924.

[10] Janaka D., Howard M. H. , Venkatesan R.FPGA Implementation of MD5 Hash Algorithm. IEEE, Electrical and Computer Engineering Canadian Conference,IVSL 2001.

[11] Ziad T. Yassen, Thamir R. Saeed and Jawad K.Ali, " An FPGA Based Implementation of CA-CFAR Processor", Medwell online, Asain Journal of information Technology, Vol. 6, No. 4, 2007

[12] Thamir R. Saeed, "Tx/Rx: Generation and Correlation of a Costas Array FM Code Using FPGA Spatran-3 Technology", Eng. & Tech. Journal, Vol.30, No.8 ,2012.

[13] Mohanty R. , Sarangi N. , Bishi S. A Security Cryptographic Hashing Algorithm. arXiv, 2010.

[14] Spartan-3A/3AN FPGA Starter Kit Board Web Page http://www.xilinx.com/s3astarter and http://www.xilinx.com/s3anstarter

[15] Mohammed A. Noaman, " A VHDL Model for Implementation of MD5 Hash Algorithm", Eng. & Tech. Journal, Vol.31,Part (A), No.6, 2013.