IMPLEMENTATION OF THE TRIPLE-DES BLOCK CIPHER USING VHDL

Sai Praveen Venigalla, M. Nagesh Babu, Srinivas Boddu, G. Santhi Swaroop Vemana
Department of Electronics & Communications Engineering,
KL University, Vijayawada, A.P., India

ABSTRACT
This paper presents FPGA implementations of the DES and Triple-DES with improved security against power analysis attacks. The proposed designs use Boolean masking, a previously introduced technique to protect smart card implementations from these attacks. Triple DES was the answer to many of the shortcomings of DES. Since it is based on the DES algorithm, it is very easy to modify existing software to use Triple DES. It also has the advantage of proven reliability and a longer key length that eliminates many of the shortcut attacks that can be used to reduce the amount of time it takes to break DES. However, even this more powerful version of DES may not be strong enough to protect data for very much longer. The DES algorithm itself has become obsolete and is in need of replacement.DES encrypts data in 64-bit and it is a symmetric algorithm. The key length is 56-bits.

KEYWORDS: *DES, Encryption, Decryption, Cryptography, Simulation, Synthesis, TDES, Cipher.*

I. INTRODUCTION

DES (the Data Encryption Standard) is a symmetric block cipher developed by IBM. The algorithm uses a 56-bit key to encipher/decipher a 64-bit block of data. The key is always presented as a 64-bit block, every 8th bit of which is ignored. However, it is usual to set each 8th bit so that each group of 8 bits has an odd number of bits set to 1.[1] The algorithm is best suited to implementation in hardware, probably to discourage implementations in software, which tend to be slow by comparison. However, modern computers are so fast that satisfactory software implementations are readily available. DES is the most widely used symmetric algorithm in the world, despite claims that the key length is too short. Ever since DES was first announced, controversy has raged about whether 56 bits is long enough to guarantee security. The key length argument goes like this. Assuming that the only feasible attack on DES is to try each key in turn until the right one is found, then 1,000,000 machines each capable of testing 1,000,000 keys per second would find (on average) one key every 12 hours. Most reasonable people might find this rather comforting and a good measure of the strength of the algorithm[9]. Those who consider the exhaustive key-search attack to be a real possibility (and to be fair the technology to do such a search is becoming a reality) can overcome the problem by using double or triple length keys. In fact, double length keys have been recommended for the financial industry for many years. Section II discusses about Triple DES, Section III discusses about the algorithm of Triple DES, Section IV represents Simulation Results, Section V describes Conclusion and finally section VI describes Scope and Future Development.

II. TRIPLE DES

Use of multiple length keys leads us to the Triple-DES algorithm, in which DES is applied three times. Triple DES is simply another mode of DES operation. It takes three 64-bit keys, for an overall key length of 192 bits. In Private Encryption, you simply type in the entire 192-bit (24 character) key rather than entering each of the three keys individually[4]. The Triple DES DLL then breaks the user provided key into three sub keys, padding the keys if necessary so they are each 64 bits long. The procedure for encryption is exactly the same as regular DES, but it is repeated three times. Hence the

name Triple DES, The data is encrypted with the first key, decrypted with the second key, and finally encrypted again with the third key. Triple DES, also known as 3DES. Consequently, Triple DES runs three times slower than standard DES, but is much more secure if used properly. The procedure for decrypting something is the same as the procedure for encryption, except it is executed in reverse[2]. Like DES, data is encrypted and decrypted in 64-bit chunks. Unfortunately, there are some weak keys that one should be aware of: if all three keys, the first and second keys, or the second and third keys are the same, then the encryption procedure is essentially the same as standard DES. This situation is to be avoided because it is the same as using a really slow version of regular DES[4]. Note that although the input key for DES is 64 bits long, the actual key used by DES is only 56 bits in length. The least significant (right-most) bit in each byte is a parity bit, and should be set so that there are always an odd number of 1s in every byte. These parity bits are ignored, so only the seven most significant bits of each byte are used, resulting in a key length of 56 bits. This means that the effective key strength for Triple DES is actually 168 bits because each of the three keys contains 8 parity bits that are not used during the encryption process.

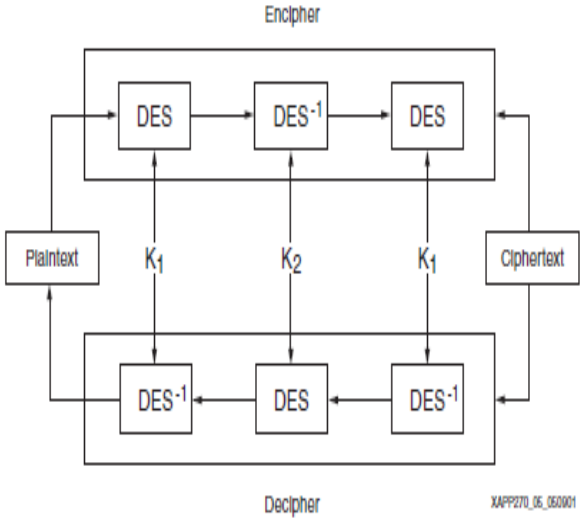

Figure 1. Triple DES-Block Diagram

If we consider a triple length key to consist of three 56-bit keys K1, K2, K3 then encryption is as follows:

- •Encrypt with K1
- •Decrypt with K2
- •Encrypt with K

Decryption is the reverse process:

- •Decrypt with K3
- •Encrypt with K2
- •Decrypt with K1

Setting K3 equal to K1 in these processes gives us a double length key K1, K2. Setting K1, K2 and K3 all equal to K has the same effect as using a single-length (56-bit key). Thus it is possible for a system using triple-DES to be compatible with a system using single-DES. DES operates on a 64 – bit block of plaintext[3]. After an initial permutation the block is broken into a right half and left half, each 32 – bits long. Then there are 16 rounds of identical operations, called Function f, in which the data are combined with the key. After the sixteenth round, the right and left halves are joined, and a final permutation (the inverse of the initial permutation) finishes off the algorithm. DES operates on a 64 – bit block of plaintext. After an initial permutation the block is broken into a right half and left half, each 32 – bits long. Then there are 16 rounds of identical operations, called Function f, in which the data are combined with the key. After the sixteenth round, the right and left halves are joined, and a final permutation (the inverse of the initial permutation) finishes off the algorithm. In each round the key bits are shifted, and then 48 – bits are selected from the 56 –bits of the key. The right half of the data is expanded to 48 – bits via an expansion permutation, combined with 48 –bits of a shifted and permuted key via an XOR, sent through 8 S- boxes producing 32- new bits, and permuted again.

These four operations make up Function f. The output of Function f is then combined with the left half via another XOR. The results of these operations become the new right half; the old right half becomes the new left half. These operations are repeated sixteen times, making 16 rounds of DES.

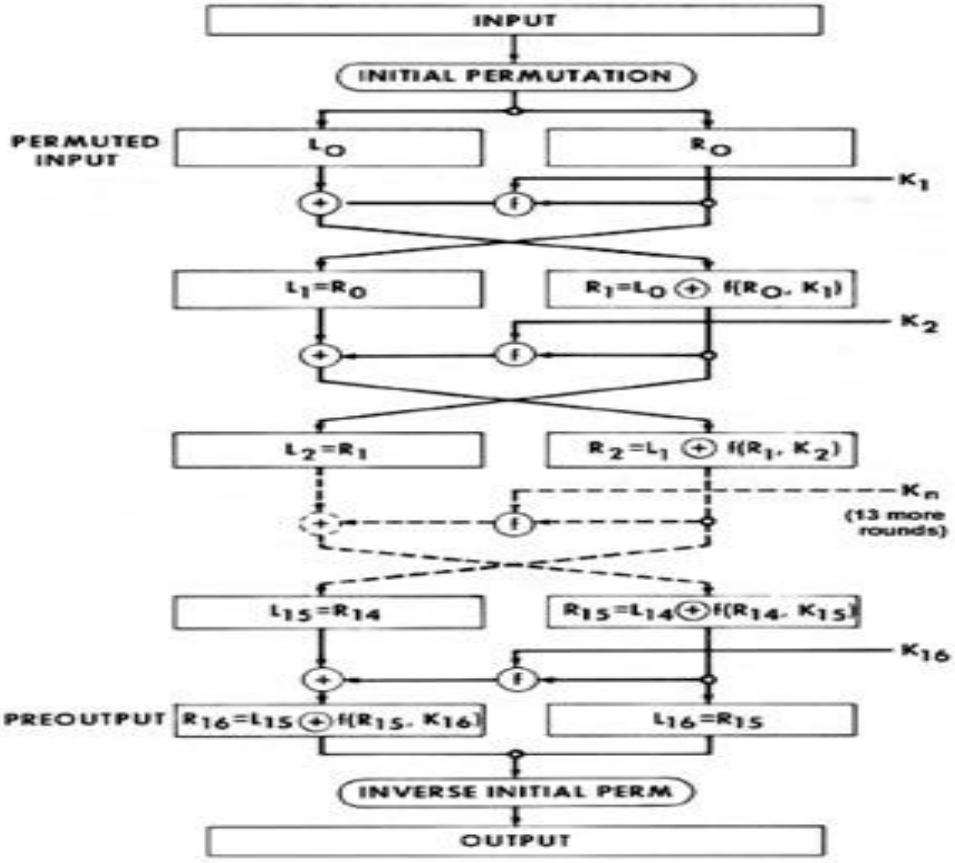

Figure 2. Enciphering computation

In each round the key bits are shifted, and then 48 – bits are selected from the 56 –bits of the key. The right half of the data is expanded to 48 – bits via an expansion permutation, combined with 48 – bits of a shifted and permuted key via an XOR, sent through 8 S- boxes producing 32- new bits, and permuted again. These four operations make up Function f. The output of Function f is then combined with the left half via another XOR. The results of these operations become the new right half; the old right half becomes the new left half. These operations are repeated sixteen times, making 16 rounds of DES.

III. ALGORITHM FOR TDES

3.1. Encryption

Step1: k1, K2, k3 are the keys in key expander with the selection function.
Step2: If selection function is active i.e. '1' then encryption process is activated with key k1. And this encryption output is given to input of the decryption i.e. selection function is '0' with key K2.
Step3: Decryption output is given to input of encryption i.e. if selection function is '1' with k3.

3.2. Decryption

Step4: It is the reverse process of encryption.

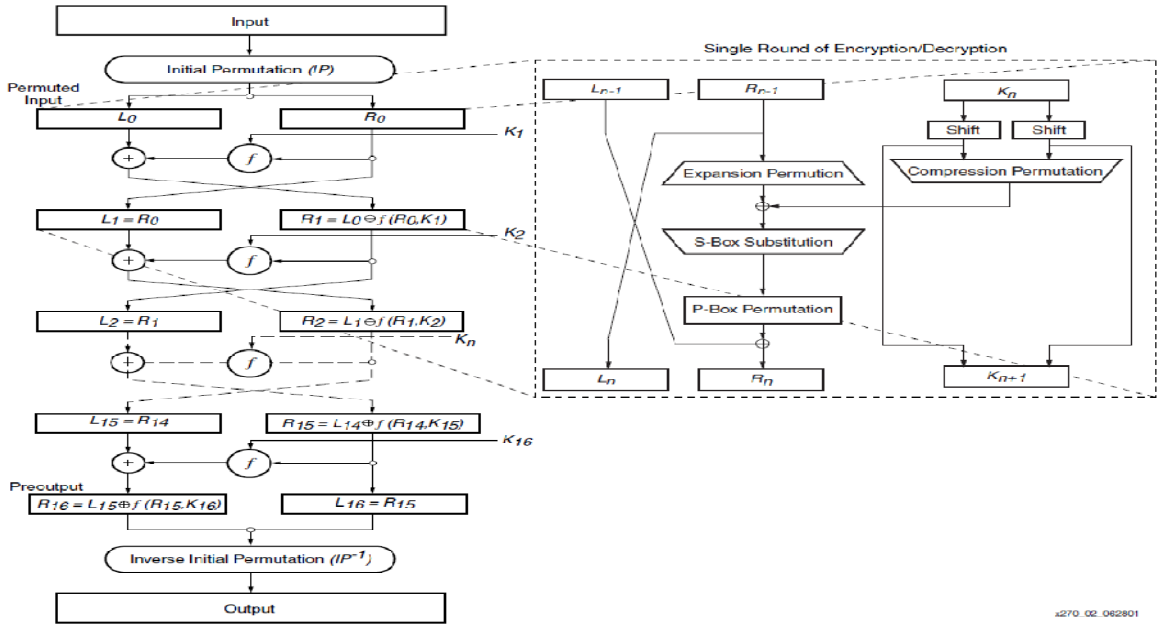

Figure 3. TDES Algorithm

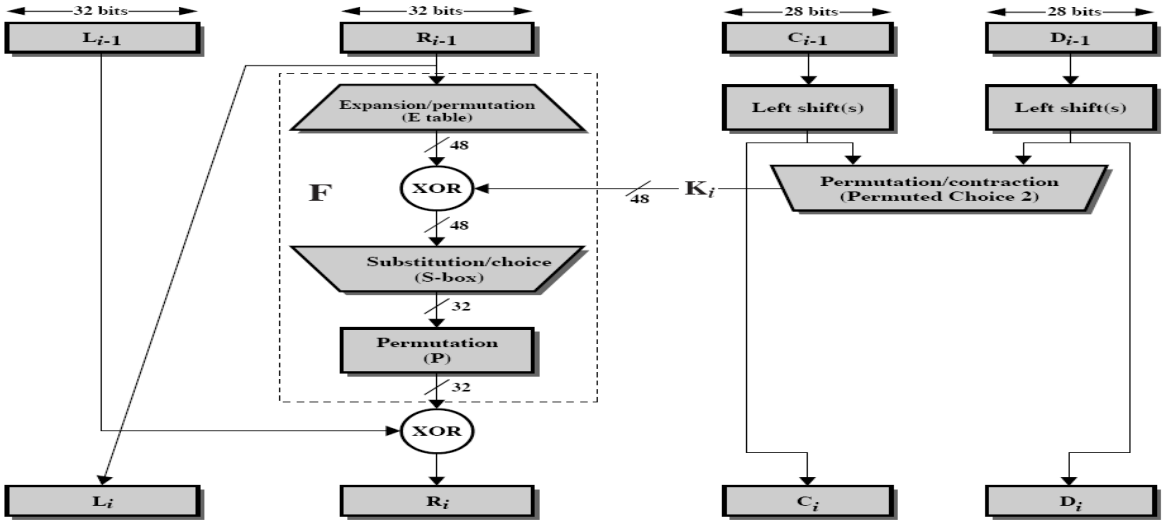

Figure 4. Single Round of DES

3.3. Initial permutation (IP)

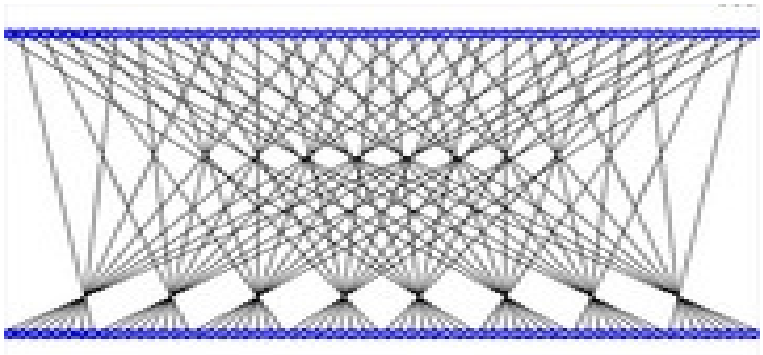

Figure 5. Initial permutation

Table 1. Initial permutation

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Table 1 specifies the input permutation on a 64-bit block. The meaning is as follows: the first bit of the output is taken from the 58th bit of the input; the second bit from the 50th bit, and so on, with the last bit of the output taken from the 7th bit of the input. The initial permutation occurs before round one; it transposes the input block as described in table 1 this table, like all the other tables in this chapter , should be read left to right, top to bottom. For example, the initial permutation moves bit 58 of the plaintext to bit position 1, bit 50 to bit position 2, and so forth. The initial permutation and the corresponding final permutation do not affect DES‘s security.

3.4. Final permutation (IP^{-1})

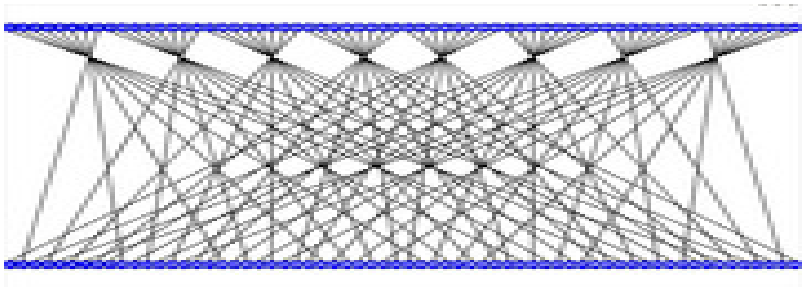

Figure 6. Final permutation

Table 2. Final permutation

IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

The final permutation is the inverse of the initial permutation; the table is interpreted similarly. This is shown in table 2

3.5. Expansion permutation (E)

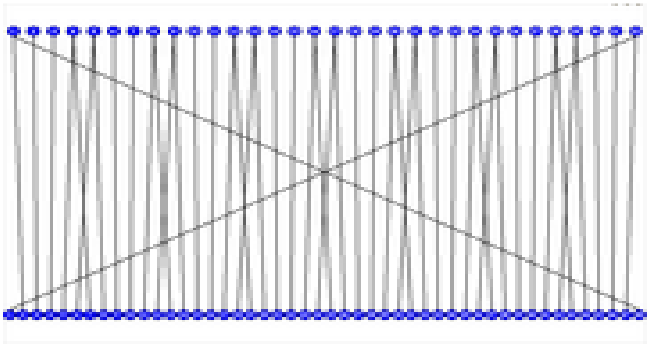

Figure 7. Expansion permutation

Table 3. Expansion permutation

E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

The expansion permutation is interpreted as for the initial and final permutations. Note that some bits from the input are duplicated at the output; e.g. the fifth bit of the input is duplicated in both the sixth and eighth bit of the output. Thus, the 32-bit half-block is expanded to 48 bits. This operation expands the right half of the data, RI, from 32-bits to 48 bits. Because this operation changes the order of the bits as well as repeating certain bits, it is known as an expansion permutation. This operation has two purposes: it makes the right half the same size as the key for the XOR operation and it provides a longer result that can be compressed during the substitution operation. However, neither of those is its main cryptographic purpose. By allowing one bit to affect two substitutions, the dependency of the output bits on the input bits spreads faster. This is called an avalanche effect. This is shown in table 3

3.6. Permutation (P)

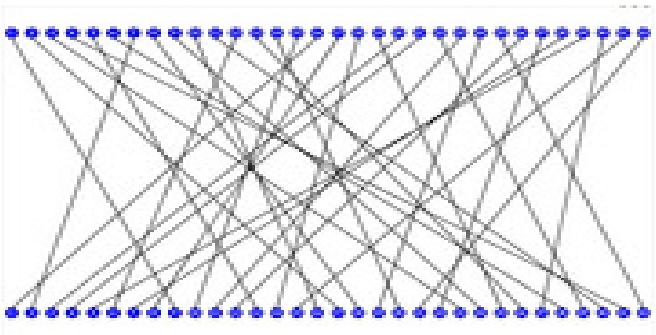

Figure 8. Permutation

Table 4. permutation

P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

The 32 – bit output of the S –box substitution is permuted according to a P –box. This permutation maps each input bit to an output position; no bits are used twice and no bits are ignored. This is called a straight permutation or just a permutation. This is shown in table 4.

3.7. Substitution boxes (S-boxes)

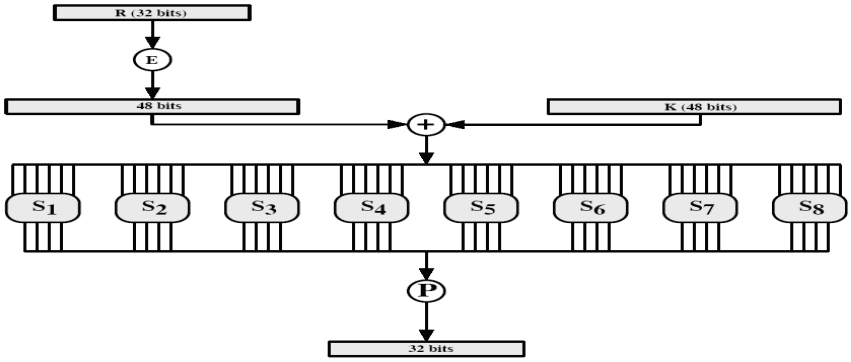

Figure 10. Calculation of f(R, k)

After the compressed key is XORed with expanded block, the 48 – bit result moves to a substitution operation. The substitutions are performed by eight substitution boxes, or S-boxes. Each S – box has a 6-bit input and a 4-bit output, and there are eight different S-boxes. The total memory requirements for the eight DES S-boxes are 256 bytes. The 48 bits are divided into eight 6-bit sub-blocks. Each separate block is operated on by a separate S-box: The first block is operated on by S-box 1; the second block is operated on by S-box 2, and so on.

3.8. Rotations in the key-schedule

Before the round subkey is selected, each half of the key schedule state is rotated left by a number of places. This table specifies the number of places rotated. Triple DES has two attractions that assure its widespread use over the next few years[6]. First, with its 168-bit key length, it overcomes the vulnerability to brute-force attack of DES. Second, the underlying encryption algorithm in Triple DES is the same as in DES. This algorithm has been subjected to more scrutiny than any other encryption algorithm over a longer period of time, and no effective cryptanalytic attack based on the algorithm rather than brute-force has been found[5]. Accordingly, there is a high level of confidence that 3DES is very resistant to cryptanalysis. If security were the only consideration, then 3DES would be an appropriate choice for a standardized encryption algorithm for decades to come.

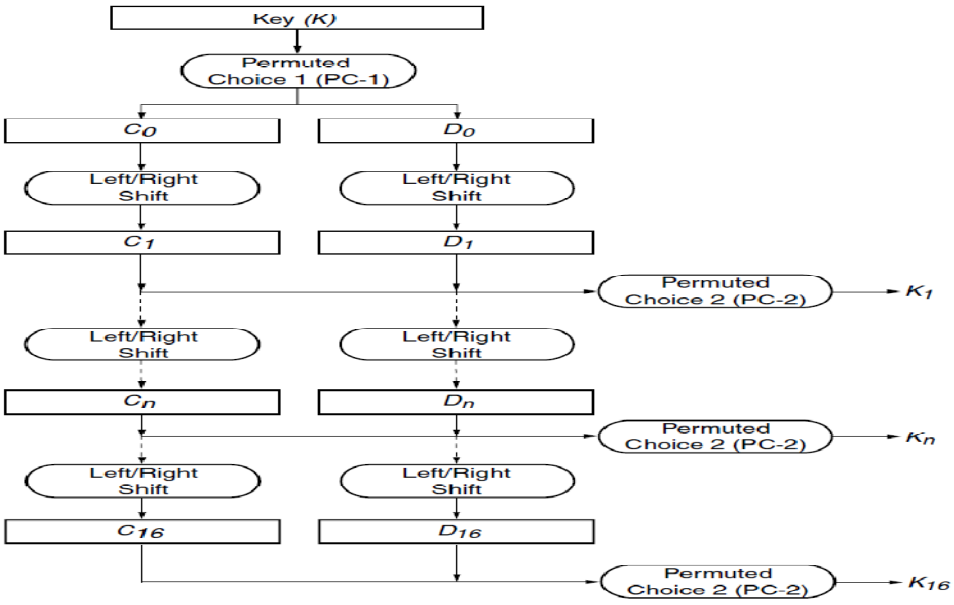

Figure 11. Key schedule calculation

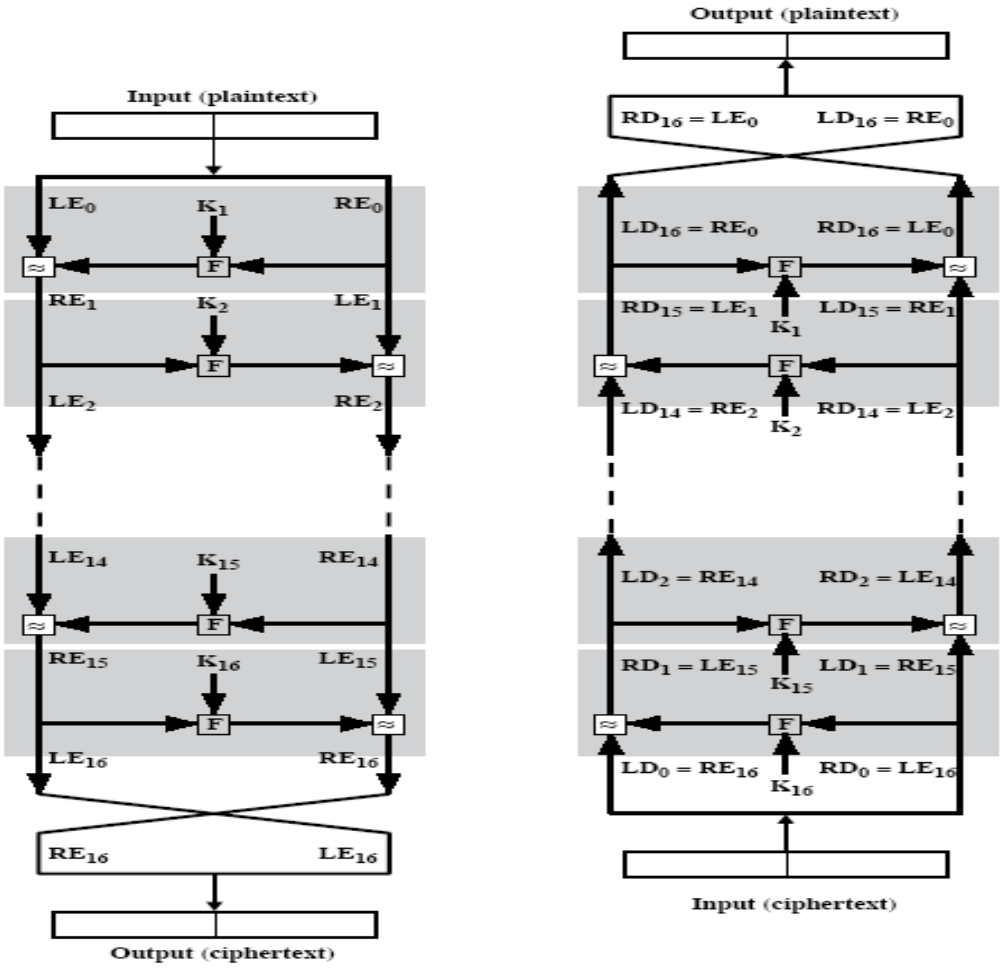

Figure 12. Feistel Decryption Algorithm

3.9. DES Decryption

i) Use same function

ii) Key is the key…
Used in reverse order (K1,…, K16 becomes K16,…, K1)
Right circular shift of 0-2 bits

0 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
(1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1)

With DES it is possible to use the same function to encrypt or decrypt a block. The only difference is that the keys must be used in the reversed order. That is , if the encryption keys for each round are K1,K2,K3,…K16, then the decryption keys are K16, K15, K14, …,K1.The algorithm that generates the key used for each round is circular as well. The key shift is shown above.

3.10. Applications

The DES3 core can be utilized for a variety of encryption applications including:

- Secure File/Data transfer
- Electronic Funds Transfer
- Encrypted Storage Data
- Secure communications

3.11. Features

- FIPS 46-3 Standard Compliant
- Encryption/Decryption performed in 48 cycles(ECB mode)
- Up to 168 bits of security
- For use in FPGA or ASIC designs
- Verilog IP Core

3.11.1. Non Pipelined Version

- Small gate count shared DES

3.11.2. Pipelined Version

- Pipelined for maximum performance
- Encryption/Decryption performed in 1 cycle (ECB mode) after an initial latency of 48 cycles

IV. SIMULATED RESULTS

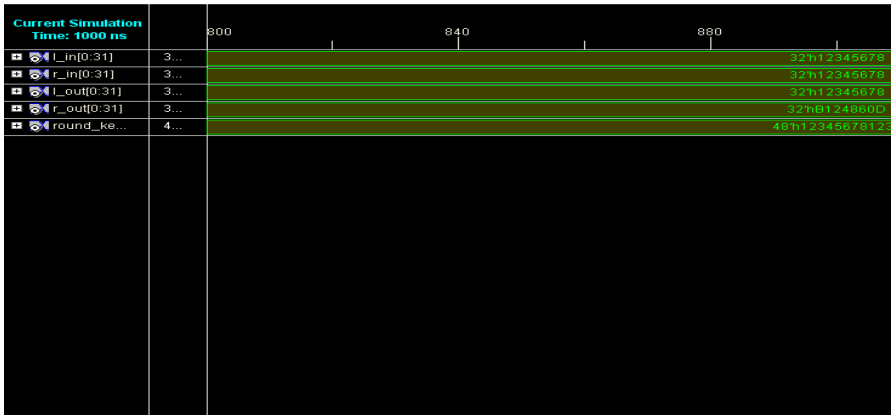

Figure 13. Waveform of DES Block

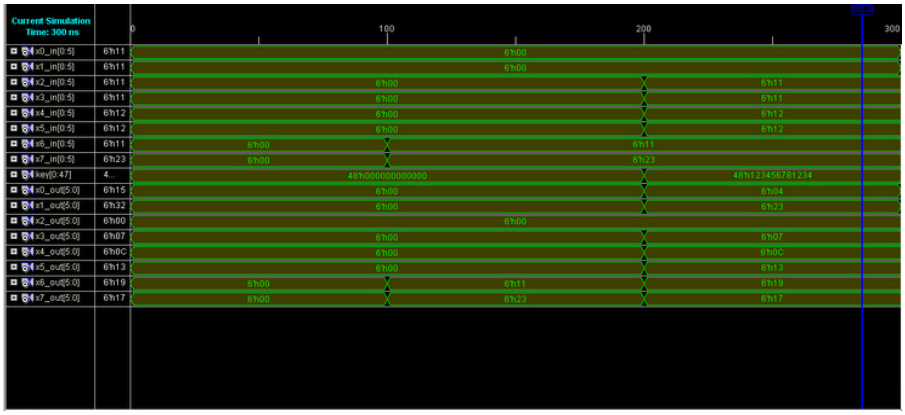

Figure 14. Waveform of Add Key

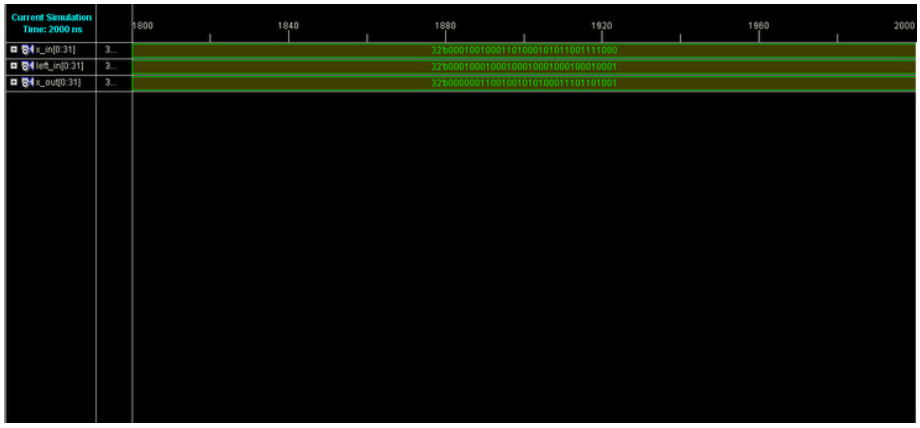

Figure 15. Waveform of Add left

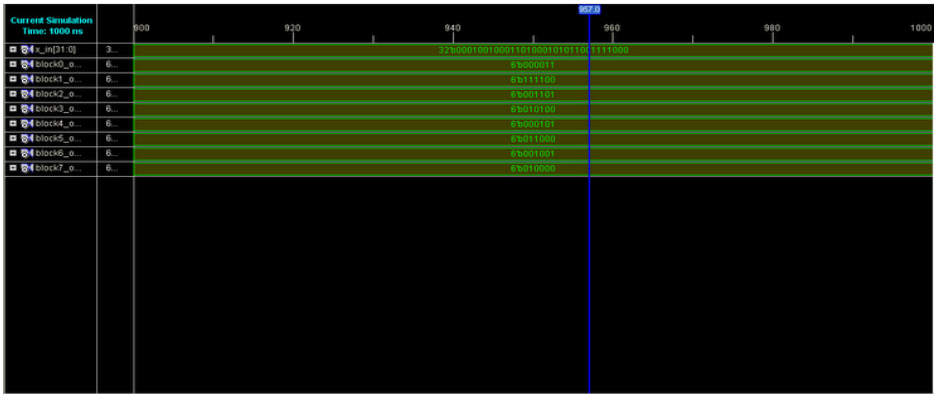

Figure 16. Waveform of Expansion Table

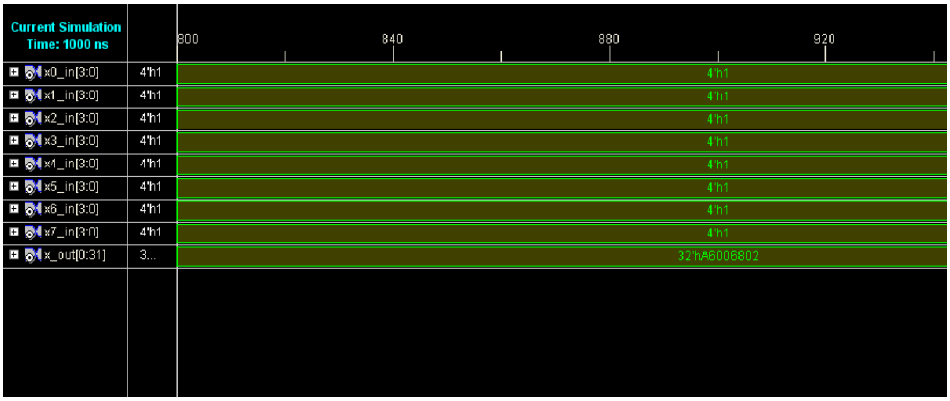

Figure 17. Waveform of Pbox

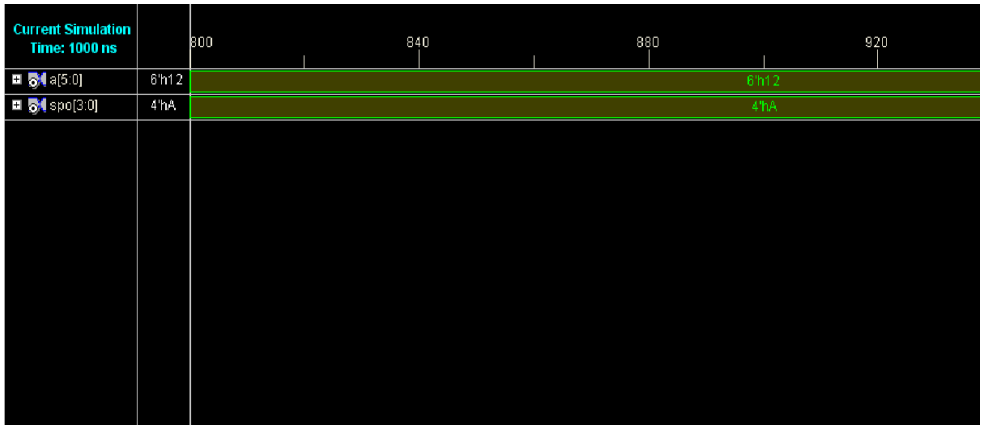

Figure 18. Waveform of S1box

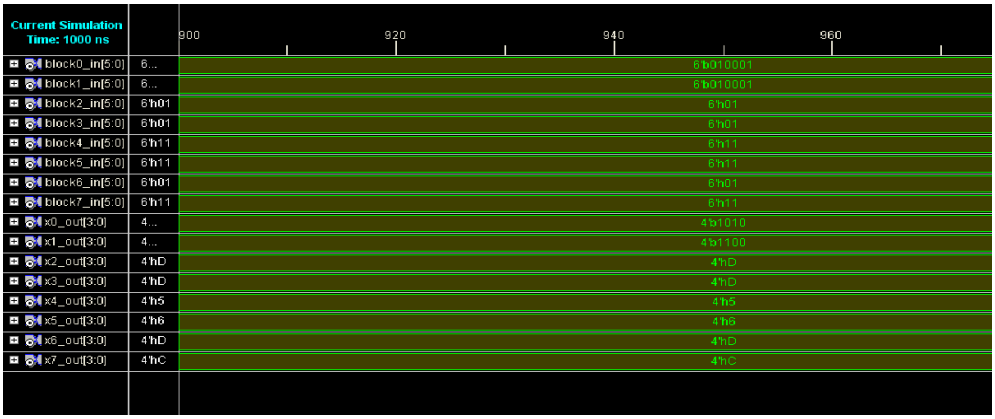

Figure 19. Waveform of Sbox

V. CONCLUSIONS

As DES will run through 16 iterations to achieve its desired cipher text (final output).With Triple DES, it will Encrypt-Decrypt-Encrypt the block and a completely different output is generated with a final combination. It's said that the security is 192 bit encryption, but also argued that regardless of the keys, the security is only 168 bit. This debate is clearly beyond the scope of this article/writer. If you wish to participate with the scientists in their discussions, it's your humility at stake. It's a safe but that Triple DES is exponentially stronger than the previous DES. After that, AES may supplant Triple DES as the default algorithm on most systems if it lives up to its expectations. But Triple DES will be kept around for compatibility reasons for many years after that. So the useful lifetime of Triple DES is far from over, even with the AES near completion.

VI. SCOPE AND FUTURE DEVELOPMENT

For the foreseeable future Triple DES is an excellent and reliable choice for the security needs of highly sensitive information. The AES will be at least as strong as Triple DES and probably much faster. It's the industry mandate from Visa and MasterCard that's requiring ATM deployers to upgrade and/or replace their legacy terminals. In a nutshell, it's all about three waves of encryption, and it's designed to make ATM transactions more secure.

REFERENCES

[1]. Data Encryption Standard, Federal Information Processing Standard (FIPS) 46, National Bureau of Standards, 1977.
[2]. Federal Information Processing Standards Publication 140-1, "Security Requirements for Cryptographic Modules", U.S. Department of Commerce/NIST, Springfield, VA: NIST, 1994

[3]. B. Schneier, "Applied Cryptography, Protocols, Algorithms, and Source Code in C", John Wiley & Sons 1994.
[4]. D. C. Feldmeier, P. R. Karn, "UNIX Password Security – Ten Years Later," CRYPTO'89, Santa Barbara, California, USA, pp. 44-63, 1989.
[5]. Xilinx, San Jose, California, USA, Virtex, 2.5 V Field Programmable Gate Arrays, 2001, www.xilinx.com.
[6]. NIST Special Publications 800-20, "Modes of Operation Validation System for the Triple Data Encryption Algorithm", National Institute of Standard and Technology, 2000.
[7]. Ohjun KWON, Hidenori SEIKE, Hirotsugu KAJISAKI and Takakazu KUROKAWA, "Implementation of AES and Triple-DES cryptography using a PCI-based FPGA board", Proc. of the International Technical Conference On Circuits/Systems Computers and Communications 2002, ITC-CSCC-2002, Phuket, Thailand, July 16-19, 2002.
[8]. Pawel Chodowiec, Kris Gaj, Peter Bellows, and Brian Schott, "Experimental Testing of the Gigabit IPSec- Compliant Implementations of Rijndael and Triple DES Using SLAAC-1V FPGA Accelerator Board", Proc. Information Security Conference, Malaga, Spain, October 1-3, 2001, pp. 220-234.
[9]. Herbert Leitold, Wolfgang Mayerwieser, Udo Payer, Karl Christian Posch, Reinhard Posch, and Johannes Wolkerstorfer, "A 155 Mbps Triple-DES Network Encryptor", Proc. Cryptographic Hardware and Embedded Systems - CHES 2000, USA, August 17-18, 2000.

Authors

Sai Praveen Venigalla was born in A.P, India. He received the B.Tech degree in Electronics & Communications Engineering from Jawaharlal Nehru Technological University in 2009. Presently he is pursuing M.Tech VLSI Design in KL University. His research interests include FPGA Implementation, Low Power Design.

M. Nagesh Babu was born in Kurnool, Kurnool (Dist.), AP, India. He received B.Tech in Electronics & Communication Engineering from JNTU Anantapur, AP, India and M.Tech from Hyderabad Institute of Technology and Management, R.R (Dist), AP, India. He is working as Associate Professor in Department of Electronics & Communication Engineering, K L University, Vijayawada, AP, India. He has 9 years of Industry experience and 9 years of teaching experience. He presented 2 papers in National conferences.

Srinivas Boddu was born in A.P, India. He received the B.Tech degree in Electronics & communications Engineering from Jawaharlal Nehru Technological University in 2009. Presently he is pursuing M.Tech VLSI Design in KL University. His research interests include FPGA Implementation, Low Power Design.

G Santhi Swaroop Vemana was born in A.P, India. He received B.Tech degree in Electronics and Communication Engineering from Jawaharlal Nehru technological university in 2008. He worked as OFC engineer at united telecoms ltd at GOA during 2009-2010. Presently he is pursuing M.Tech VLSI Design in KL University. His research interests include FPGA Implementation, Low Power Design.