International Journal of
Scientific Engineering and Technology Research

www.ijsetr.com

ISSN 2319-8885
Vol.03,Issue.46
December-2014,
Pages:9381-9385

Implementation of DES Cryptography on UART Module

P. KONDALA RAO[1], M. RAVINDRA KUMAR[2]
[1]PG Scholar, Pydah College of Engineering & Technology., AP, India, E-mail: kondalarao79@gmail.com.
[2]Assistant Professor, Pydah College of Engineering & Technology., AP, India, E-mail: ravi.moningi@gmail.com.

Abstract: The UART (universal asynchronous receiver and transmitter) module provides asynchronous serial communication with external devices such as modems and other computers. The UART can be used to control the process of breaking parallel data from the PC down into serial data that can be transmitted and vice versa for receiving data. The UART allows the devices to communicate without the need to be synchronized. In the proposed design the communication is done securely by introducing DES Cryptography module by performing both s the encryption and decryption process for providing the security for the data which should be transmitted. So by this project we are able to transmit the data with low cost and with high security. Modern applications of DES cover a wide variety of applications, such as secure internet (ssl), electronic financial transactions, remote access servers, cable modems, secure video surveillance and encrypted data storage. The functionality is verified through ISE simulator and the synthesis is carried out by XILINX ISE 12.3i using verilog HDL.

Keywords: DES, UART, XILINX, Verilog.

I. INTRODUCTION

Serial communication is the process of sending data and receiving one bit of data at one time sequentially through a communications channel or computer bus. On the other hand, parallel communications is a process where all the bits of each symbol are sent together. In general, serial communication is used for all long communications and most computer networks where it is impractical to use parallel communications due to the cost of cable and synchronization. Nowadays computer buses or network communication using serial communications are becoming more common as improved technology enables them to transfer data at higher speeds. There are 2 types of serial communication, full duplex and half duplex. A full duplex device can send and receive data at the same time. Thus, a full duplex communication needs 2 different ports, one for serial in data while another for serial out data. On the other hand, half duplex serial devices support only one-way communications and therefore only able either receiving or transmitting data at a time. Normally half duplex devices share the same port for both serial in and out.

Universal asynchronous receive transmit (UART) is an asynchronous serial receiver/transmitter. It is a piece of computer hardware that commonly used in PC serial port to translate data between parallel and serial interfaces. The UART takes bytes of data and transmits the individual bits in a sequential fashion. At the receiving point, UART re-assembles the bits into complete bytes. Asynchronous transmission allows data to be transmitted without having to send a clock signal to the receiver. Thus, the sender and receiver must agree on timing parameters in advance and special bits are added to each word, which is used to synchronize the sending and receiving units. In general, UART contains of two main block, the transmitter and receiver block. The transmitter sends a byte of data bit by bit serially out from UART while UART receiver receives the serial in data bit by bit and converts them into a byte of data. The origins of DES go back to the early 1970s. In 1972, after concluding a study on the US government's computer security needs, the US standards body NBS (National Bureau of Standards) — now named NIST (National Institute of Standards and Technology) — identified a need for a government-wide standard for encrypting unclassified, sensitive information.

Accordingly, on 15 May 1973, after consulting with the NSA, NBS solicited proposals for a cipher that would meet rigorous design criteria. None of the submissions, however, turned out to be suitable. A second request was issued on 27 August 1974.On 17 March 1975, the proposed DES was published in the Federal Register. Public comments were requested, and in the following year two open workshops were held to discuss the proposed standard. There was some criticism from various parties, including from public-key cryptography pioneers Martin Hellman and Whitfield Diffie, citing a shortened key length and the mysterious "S-boxes" as evidence of improper interference from the NSA. The suspicion was that the algorithm had been covertly weakened by the intelligence agency so that they — but no-one else — could easily read encrypted messages. Alan Konheim (one of the designers of DES) commented, "We sent the S-boxes off to Washington. They came back and were all different. The United States Senate Select Committee on Intelligence reviewed the NSA's actions to determine whether there had been any improper involvement. In the unclassified summary of their findings, published in 1978, the Committee wrote:

In the development of DES, NSA convinced IBM that a reduced key size was sufficient; indirectly assisted in the development of the S-box structures; and certified that the final DES algorithm was, to the best of their knowledge, free from any statistical or mathematical weakness. However, it also found that NSA did not tamper with the design of the algorithm in any way. IBM invented and designed the algorithm, made all pertinent decisions regarding it, and concurred that the agreed upon key size was more than adequate for all commercial applications for which the DES was intended. Another member of the DES team, Walter Tuchman, stated "We developed the DES algorithm entirely within IBM using IBMers. The NSA did not dictate a single wire In contrast, a declassified NSA book on cryptologic history states.

II. DATA ENCRYPTION STANDARD

The selective application of technological and related procedural safeguards is an important responsibility of every Federal organization in providing adequate security to its electronic data systems. This publication specifies two cryptographic algorithms, the Data Encryption Standard (DES) and the Triple Data Encryption Algorithm (TDEA) which may be used by Federal organizations to protect sensitive data. Protection of data during transmission or while in storage may be necessary to maintain the confidentiality and integrity of the information represented by the data. The algorithms uniquely define the mathematical steps required to transform data into a cryptographic cipher and also to transform the cipher back to the original form. The Data Encryption Standard is being made available for use by Federal agencies within the context of a total security program consisting of physical security procedures, good information management practices, and computer system/network access controls as shown in fig.1.

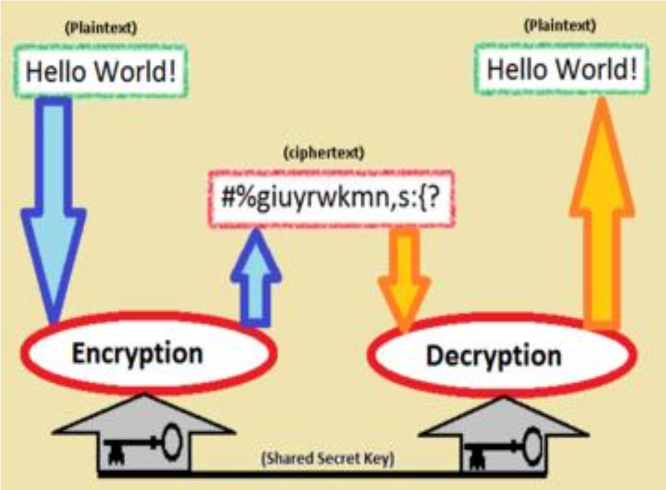


Fig.1. Data Encryption Standard (DES).

The Data Encryption Standard (DES) specifies two FIPS approved cryptographic algorithms as required by FIPS 140-1. When used in conjunction with American National Standards Institute (ANSI) X9.52 standard, this publication provides a complete description of the mathematical algorithms for encrypting (enciphering) and decrypting (deciphering) binary coded information. Encrypting data converts it to an unintelligible form called cipher as shown in Fig.2. Decrypting cipher converts the data back to its original form called plaintext. The algorithm described in this standard specifies both enciphering and deciphering operations which are based on a binary number called a key. A DES key consists of 64 binary digits ("0"s or "1"s) of which 56 bits are randomly generated and used directly by the algorithm. The other 8 bits, which are not used by the algorithm, may be used for error detection. The 8 error detecting bits are set to make the parity of each 8-bit byte of the key odd, i.e., there is an odd number of "1"s in each 8-bit byte. A TDEA key consists of three DES keys, which is also referred to as a key bundle. Authorized users of encrypted computer data must have the key that was used to encipher the data in order to decrypt it.

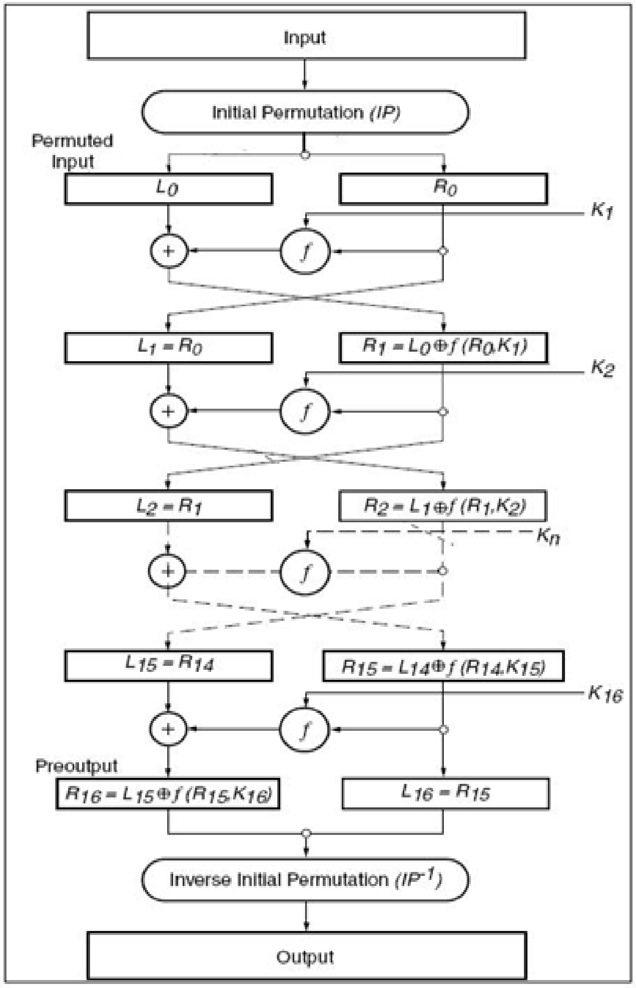
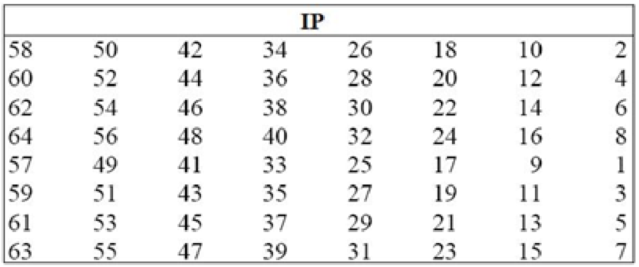

Input
Initial Permutation (IP)
Permuted Input
L_0
R_0
K_1
+
f
$L_1 = R_0$
$R_1 = L_0 \oplus f(R_0,K_1)$
K_2
+
f
$L_2 = R_1$
$R_2 = L_1 \oplus f(R_1,K_2)$
K_n
+
f
$L_{15} = R_{14}$
$R_{15} = L_{14} \oplus f(R_{14},K_{15})$
K_{16}
+
f
Preoutput
$R_{16} = L_{15} \oplus f(R_{15},K_{16})$
$L_{16} = R_{15}$
Inverse Initial Permutation (IP^{-1})
Output

Fig. 2. DES Encryption Process.

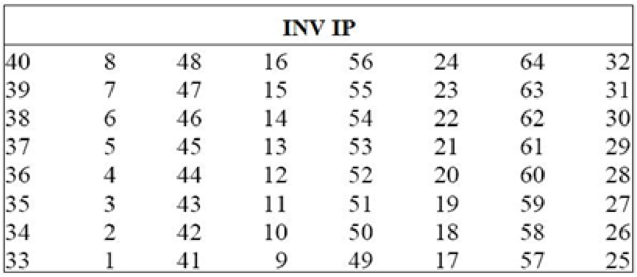
The encryption algorithms specified in this standard are commonly known among those using the standard cryptographic. Sometimes keys are generated in an encrypted form. A random 64-bit number is generated and defined to be the cipher formed by the encryption of a key using a key encrypting key. In this case the parity bits of the encrypted key cannot be set until after the key is decrypted. Security of the data depends on the security provided for the key used to encipher and decipher the data. Data can be recovered from cipher only by using exactly the same key used to encipher it. Unauthorized recipients of the cipher who know the algorithm but do not have the correct key cannot derive the original data algorithmically. Also, anyone who does have the key and the algorithm can easily decipher the cipher and obtain the original data. A standard algorithm based on a secure key thus provides a basis for exchanging encrypted computer data by issuing the key used to encipher it to those authorized to have the data.

Initial Permutation and its inverse are defined, as shown in fig3 and fig4. The input to a table consists of 64 bits numbered from 1 to 64 entries in the permutation table contain a permutation of the numbers from 1 to 64. Each entry in the permutation table indicates the position of a numbered input bit in the output, which also consists of 64 bits.



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

Fig.3. Initial Permutations.



INV IP							
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

Fig.4. Inverse Initial Permutations.

The left and right halves of 64-bit intermediate value are treated as separate 32-bit quantities, labeled L (left) and R (right). The overall processing at each round can be summarized in the following formulas:

$$L_i = R_{i-1} \quad \textbf{(1)}$$

$$R_i = L_{i-1} * F(R_{i-1}, K_i) \quad \textbf{(2)}$$

The round key K_i is 48 bits. The R input is 32 bits. This R input is first expanded to 48 bits by using a table that defines a permutation plus an expansion that involves duplication of 16 of the R bits. The resulting 48 bits are XORed with K_i. This 48-bits result passes through a substitution function that produces a 32 bit output. The role of S-box in the function F is shown fig.5 below. The substitution consists of a set of eight S-boxes, each of which accepts 6 bits as input and produces 4 bits as output. These transformations are defined in table 2-5. The first and last bits of the input to box S_i form a 2-bit binary number to select one of four substitutions defined by four rows in the table for S_i. The middle four bits select one of the sixteen columns. The decimal value in the cell selected by the row and column is then converted to its 4-bit representation to produce the output. Calculation of F(R,K).The decryption of the encrypted information undergoes the same procedure that is used for encryption of the data , that is first it goes for initial permutation and then 16 rounds of complex key dependent calculations followed by it the final permutation which gives the decrypted data, but with a slight change i.e. for encryption process the key given to round1 till round 16 are from key1 to key 16 but for the decryption process the keys are given in the reverse process i.e. Key16 is given to round1, key15 is given to round2 and soon till round 16.

III. IMPLEMENTATION OF UART USING DES ALGORITHM

The UART (universal asynchronous receiver and transmitter) module provides asynchronous serial communication with external devices such as modems and other computers. The UART can be used to control the process of breaking parallel data from the PC down into serial data that can be transmitted and vice versa for receiving data. UART includes three kernel modules which are the baud rate generator, receiver and transmitter. By implementing the DES algorithm to uart serial communication module then it forms below fig 4-2.For modern society, the completeness of data is particularly important. This problem is solved by using the DES algorithm. It can provide security for communication devices. The input data which is given should be applied to UART tx. The UART tx transmits a byte of data at a time using UART protocol. The obtained serial data is passed through SIPO. The output of the SIPO block is collected by DES cipher unit and generates the cipher text. The obtained cipher text is decrypted by DES decryption unit.

The corresponding output is received by the UART receiver. UART starts the data transmission by asserting a bit called the "Start Bit" to the beginning of each data that is to be transmitted. The Start Bit is also used to inform the receiver that a byte of data is about to be sent. After the Start Bit, the individual bits of the “byte” of data are sent, with the Least Significant Bit (LSB) being sent first. Each bit in the transmission is transmitted for exactly the same amount of time as all of the other bits. On the other, UART the receiver will need to sample the logic value that being received at approximately halfway through the period assigned to each bit to determine if it is logic 1 or logic 0. When a byte of data has been sent, the transmitter may add a "Parity Bit". The receiver to perform simple error checking may use the Parity Bit. In this project, parity bit is not being implemented. After this, a "Stop Bit" is sent by the transmitter to indicate the transmitter has completed the data transmission. If another byte of data is to be transmitted, the Start Bit for the new data can be sent as soon as the Stop Bit for the previous word has been sent.

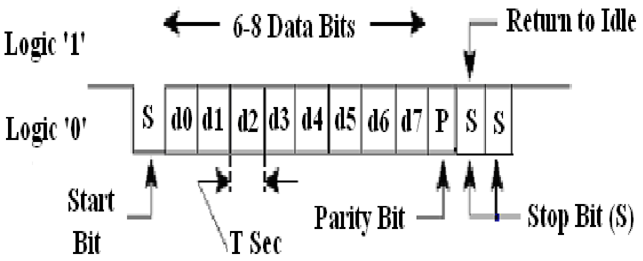


Fig. 5. Uart Data Frame Format.

IV. RESULTS

Results of this paper is shown in bellow Figs.6, 7 and 8.

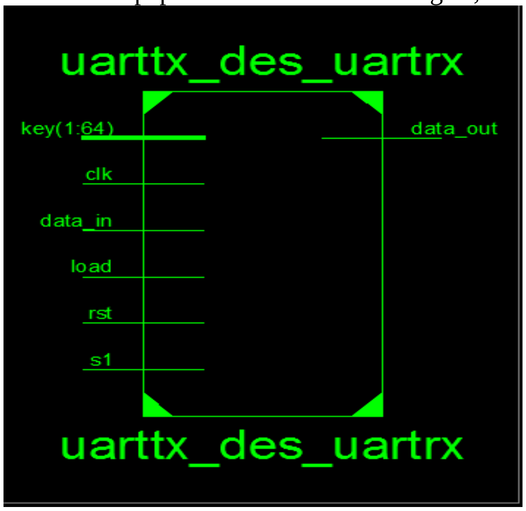


Fig.6. Schematics.

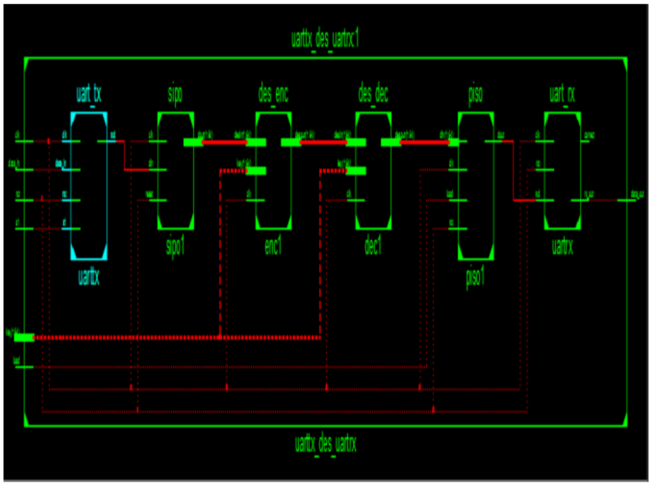


Fig.7. RTL Schematic.

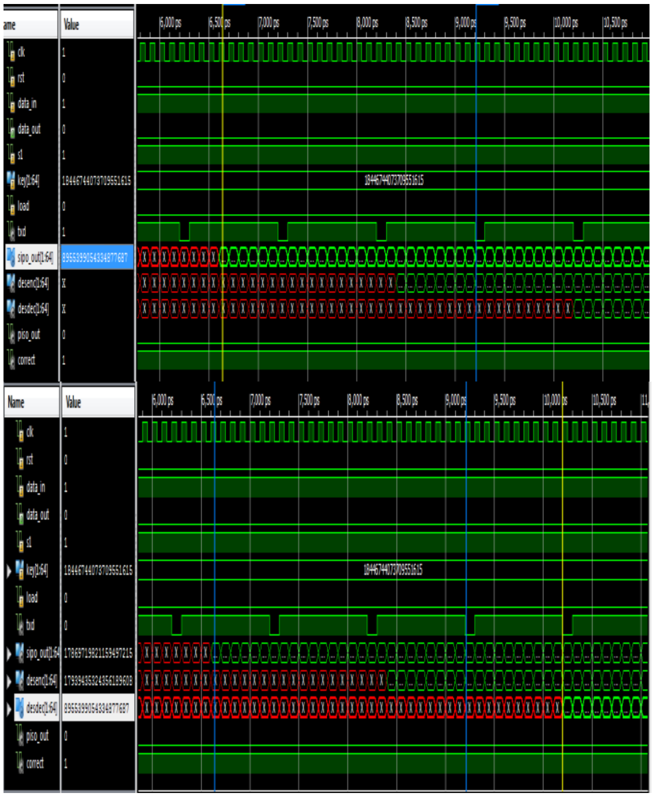


Fig.8. Simulated Waveform.

V. CONCLUSION

The input data which is given should be applied to UART tx. The UART tx transmits a byte of data at a time using UART protocol. The obtained serial data is passed through SIPO. The output of the SIPO block is collected by DES cipher unit and generates the cipher text. The obtained cipher text is decrypted by DES decryption unit. The corresponding output is received by the UART receiver. The functionality is verified through ISE simulator and the synthesis is carried out by XILINX ISE 12.3i using verilog HDL. From the results it can be concluded that the area required to integrate the chip is 3615 LUTS with power consumption of 58.56 mw and delay of 0.567ns.

VI. REFERENCES

[1] Zou,Jie Yang,Jianning。Design and Realization of UART Controller based on FPGA.

[2] Liakot Ali , Roslina Sidek , Ishak Aris , Alauddin Mohd. Ali , Bambang Sunaryo Suparjo. Design of a micro - UART for SoC application [J].In: Computers and Electrical Engineering 30 (2004) 257–268.

[3] HU Hua, BAI Feng-e. Design and Simulation of UART Serial Communication Module Based on Verilog -HDL[J]. J ISUANJ I YUXIANDA IHUA 2008 Vol. 8.

[4] Frank Durda Serial and UART Tutorial. uhclem @FreeBSD.org.

[5] Kamoun N, Bossuet L, Ghazel A. Correlated Power Noise Generatoras A Low Cost DPA Countermeasures to Secure Hardware AES Ciper[C].IEEE Signals Circuits and Systems, 010:1-6.

[6] Alioto M, Poli M, et al. A General Model of DPA Attacks to Precharged Buses in Symmetric-key Cryptographic Algorithms[C]. International Conference on Circuit Theory and Design, 2007:368- 371.

[7] Zafar Y., P. Jihan, et al. Random Clocking Induced DPA Attack Immunity in FPGA[C]. IEEE International Conference on Industrial Technology, 2010: 1068 – 1070.

[8] Guiley S, Sauvage L, et al. Security Evaluation of WDDL and Seclib Countermeasures against Power Attacks[J]. IEEE TRANSACTIONS ON COMPUTERS, 2008, Vol.57:1482-1497.

[9] Yoshikawa, M, et al. Efficient Random Number for the Masking Method against DPA Attacks[C].IEEE Conference on Systems Engineering, 2011:16-18.

[10] M.-L. Akkar, C. Giraud. An Implementation of DES and AES Secure against Some Attacks[C]. LNCS, 2001:309-318.