A Compact and Efficient FPGA Implementation of the DES Algorithm

Nazar A. Saqib, Francisco Rodríguez-Henriquez, and Arturo Díaz-Pérez

Computer Science Section, Electrical Engineering Department
Centro de Investigación y de Estudios Avanzados del IPN
Av. Instituto Politécnico Nacional No. 2508, México D.F.
{nabbas@computacion.cs.cinvestav.mx, {francisco, adiaz}@cs.cinvestav.mx}

Abstract. In this paper we present an efficient and compact reconfigurable hardware implementation of the Data Encryption Standard (DES) algorithm. Our design was implemented on a VirtexE XCV400e device. As a strategy to reduce the associated design critical path, we utilized a parallel structure that allowed us to compute all the eight DES S-boxes simultaneously. Our DES round design achieved a data encryption/decryption rate of 274 Mbits/s occupying only 117 CLB slices. These results are quite competitive when compared with other reported reconfigurable hardware implementations of DES.

Key Words: DES, FPGA, Parallel structure.

1 Introduction

Cryptography is the main mechanism to secure digital information data. In recent years due to the heavy increase in the volume of information data, secure and rapid cryptographic algorithms were developed to combat security threats and security measures were considered to be essential wherever, digital data transactions have to be performed. The elevated diversity seen on security applications posed an additional challenge since highly secure algorithms were not the only requirement but rather high performance for some applications and for others, less space. In that scenario, cryptographic designers have explored not only realizations on software platforms, but also on classic hardware or reconfigurable hardware platforms as well.

Implementing cryptographic algorithms on reconfigurable hardware provides major benefits over VLSI (very large scale integrated circuits) and software platforms since they offer high speed similar to VLSI and high flexibility similar to software. VLSI implementations are fast but must be designed all the way from behavioral description to the physical layout. They have to follow

an expensive and time consuming fabrication process. Software implementations offer high flexibility but they are not fast enough for the applications where time factor is vital. On the other hand, reconfigurable devices are attractive since the time and costs of VLSI design and fabrication can be reduced. Moreover, they offer high potential for reprogramming and experimenting on multiple architectures or several revisions of the same architecture.

Among the different cryptographic algorithms, the most popular example in the field of symmetric ciphers is the Data Encryption Standard (DES) algorithm [1, 2], which was developed by IBM in the mid-seventies. The DES algorithm is organized in repetitive rounds composed of several bit-level operations such as logical operations, permutations, substitutions, shift operations, etc. Although those features are naturally suited for efficient implementations on reconfigurable devices FPGAs, DES implementations can be found on all platforms: software [1–5], VLSI [6–8] and reconfigurable hardware using FPGA devices [9–13, 8, 14].

In this paper we present an efficient and compact DES architecture especially designed for reconfigurable hardware platforms. The DES implementation presented in this paper differs from other previous works in the following aspect: It makes use of an eight DES S-Boxes parallel structure, resulting in a significant reduction of the critical path for encryption/decryption.

The rest of this paper is organized as follows: Section 2 describes the DES algorithm. Our proposed DES architecture and its implementation on a reconfigurable hardware device is presented in Section 3. Section 4 compares the achieved results with the previous DES implementations. Conclusions and future work are drawn in Section 5.

2 The DES Algorithm

On August, 1974, IBM submitted a candidate (under the name LUCIFER) for cryptographic algorithm in response to the second call from National Bureau of Standards (NBS), now the National Institute of Standards & Technology (NIST)[15], to protect data during transmission and storage. NBS launched an evaluation process with the help of National Security Agency (NSA) and finally adopted on July 1977 a modification of LUCIFER algorithm as the new Data Encryption Standard (DES). The Data Encryption Standard [16], known as Data Encryption Algorithm (DEA) by the ANSI [17] and the DEA-1 by the ISO [18] remained a worldwide standard for a long time and was replaced by the new Advanced Encryption Standard (AES) on October 2000. However, it is expected that DES will remain in the public domain for a number of years. It provides a basis for comparison for new algorithms and it is also used in

IPSec protocols, ATM cell encryption, the secure socket layer (SSL) protocol and in TripleDES. A detailed description of the DES algorithm can be found at [19–21].

DES is a block cipher: It encrypts/decrypts data in 64-bit blocks using a 64-bit key (although its effective key length is in reality only 56-bit). DES is a symmetric algorithm: The same algorithm and key are used for both encryption and decryption. DES is an iterative cipher: the basic building block (a substitution followed by a permutation) called a *round* is repeated 16 times. For each DES round, a sub-key is derived from the original key using an algorithm called *key schedule*. Key schedule for encryption and decryption is the same except for the minor difference in the order (reverse) of the sub-keys for decryption. A basic algorithm flow for encrypting/decrypting one block of data is shown in Fig. 1. Encryption begins with an *initial permutation* (IP),

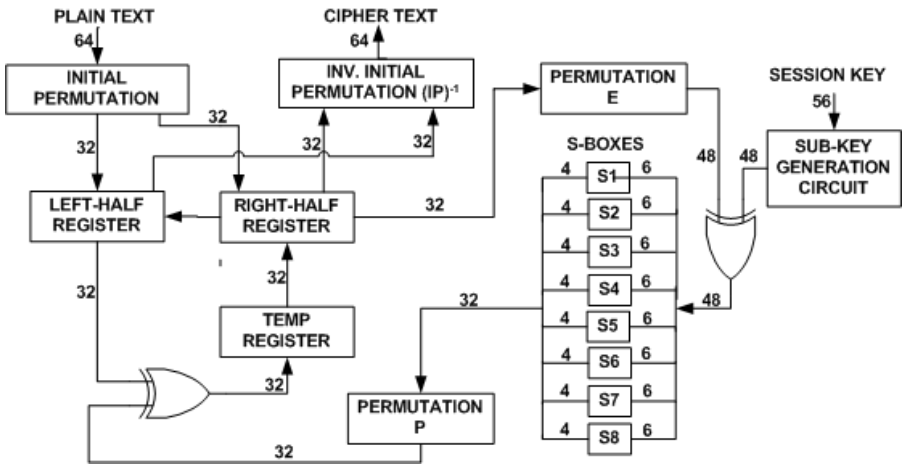

Fig. 1. DES Algorithm

which scrambles the 64-bit plain-text in a fixed pattern. The result of the initial permutation is sent to two 32-bit registers, called the *right half register* and *left half register*. Those registers hold the two halves of the intermediate results through successive 16 iterations. The contents of the right half register are permuted (*permutation E*) and sent to an exclusive-OR unit along with the sub-key for each iteration. Note that some bits are selected twice, allowing the 32-bit register to expand to 48 bits. The 48-bit output of the exclusive-OR

block is divided into eight groups (6-bits each) to address eight substitution memories (S-boxes). A *permutation P* is applied to 32-bit output from S-boxes and then feed into an exclusive-OR block along with the contents of the left half register. The output of this block is written into a temporary register, concluding the first iteration.

At the next clock cycle, the contents of the temporary registers are written into the right half register and previous contents of the right half register are written into left half register. This process is repeated through the whole 16 DES iterations. After 16 iterations, the right half and left half register contents are subjected to a final permutation IP^{-1}, which is the inverse of the initial permutation. The output of IP^{-1} is the 64-bit cipher-text.

3 A Reconfigurable Hardware DES Implementation

Referring to Fig. 1, the 16 iterations of the identical operations are repeated which come under the name of a function $f(R,K)$. DES combines first permutation, function $f(R,K)$, second permutation, and key schedule for one encryption as shown in Fig. 2. As it was mentioned before, DES encryption and decryption routines are the same. Only the order of the sub-keys is reversed in case of decryption.

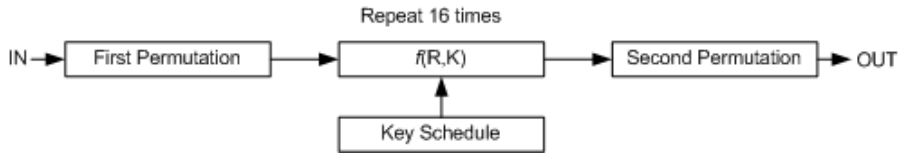

Fig. 2. DES algorithm

Key schedule algorithm is simple and consumes a little time. Moreover the keys generated once, are used for the whole session. However, in our FPGA implementation, pre-computed sub-keys are stored in memories.

In the rest of this section, specific details of the architectural arrangement for DES leading to its FPGA implementation are presented.

3.1 Block Architecture of a DES Round

Fig. 3 represents a block diagram architecture for a round of the DES algorithm. That architecture was specifically tailored for its efficient implementation on a reconfigurable hardware platform.

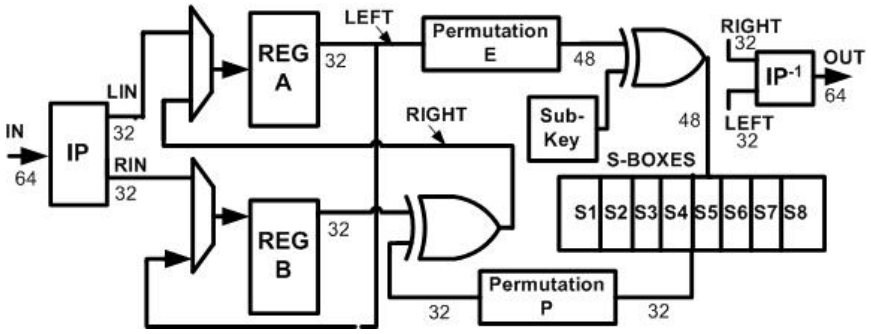

Fig. 3. DES algorithm implementation on FPGA

DES algorithm is mainly based on two operations: fixed permutations and substitution. Both operations can be proficiently implemented on an FPGA device. DES makes use of 8 S-Boxes (each of 64×4) occupying a total of 2Kbits. This relatively small amount of memory can be implemented by using the distributed memory resources in FPGAs. Fixed permutations in fact do not occupy FPGA resources as they can be implemented by just changing the wires. Those useful features were exploited to get an efficient implementation of DES as shown in Fig 3.

Three inputs: Chip Enable (CE), Clock (CLK), input data (IN) and the only output (OUT) are the four pins of DES chip. Chip enable (CE) activates the timing logic as well as the rest of the circuitry, in its low state (when it is '0'). The external clock CLK is the master clock for the whole circuit that is used to generate all the control signals to synchronize the data flow.

When CE enables the circuit, The 64-bit at the input are permuted and divided into two halves RIN and LIN. At the first rising edge of the clock, both halves are being transferred to the output of the two registers REGA and REGB. The right halves (REGA output) go through a number of operations: Permutation E; addition with sub-key; substitution (through S-Boxes); Permutation P and; addition with the original left half (REGB output). Before the next clock comes, the old right half (RIGHT) is the input of the register

REGB and the new left half (LEFT) is the input of the register REGA. The sixteen iterations are then executed. After 16th clock cycles the two halves RIGHT and LEFT are concatenated and the resulting block goes through the inverse permutation (IP^{-1}) resulting one encryption for a 64-bit input block. Notice that the usage of an eight DES S-Boxes parallel structure, results in a significant reduction of the critical path for encryption/decryption.

3.2 Implementation summary

FPGA implementation of DES algorithm was accomplished on a VirtexE device XCV400e-8-bg560 using Xilinx Foundation Series F4.1i as synthesis tool. The design was coded using VHDL language. It occupied 165 (3%) CLB slices, 117 (1%) slice Flip Flops and 129 (41%) I/Os. The design achieves a frequency of 68.05 MHz (14.7 ηS). It takes 16 clock cycles to encrypt one data block (64-bits). Therefore, the achieved throughput is $(68.05 \times 64)/16$=274 Mbits/s.

4 Performance comparison

Table 1 shows the performance figures for some representative DES hardware implementations. Notice that the achieved results are competitive with the existing implementations.

A VLSI implementation of DES on static 0.6 micron CMOS technology at [8] is the fastest implementation of DES reported in the literature. Using a pipeline approach, the encryption can be performed at the rate of $\geq$6.7 Gbs. Several FPGA implementations of DES have been reported in the liter-

Author	Device used	CLB slices (A)	Allowed Freq. (MHz)	Throughput (Mbits/s)(T)	T/A Factor
Wong et al. [10]	XC4020E	438	10	26.7	0.06
Kaps and Paar [11]	XC4028EX	741	25.18	402.7	0.54
Free-DES [12]	XCV400	5263	47.7	3052	0.57
McLoony, McCanny [13]	XCV1000	6446	59.5	3808	0.59
Sandia Laboratories [8]	ASIC	—	—	9280	—
Patterson (Jbits) [14]	XCV150	1584	168	10752	6.78
This work (FPGA)	XCV400e	117	68.05	274	2.34

Table 1. Recent DES reconfigurable hardware implementations

ature achieving throughput ranges from 26 to 10752 Mbits/s using different design strategies. A DES implementation at [12] is a free DES cores which uses pipeline approach in ECB mode and achieves a data rate of 3052 Mbits/s. A java-based (Jbits) DES implementation at [14] achieves the fastest encryption rate of 10752 Mbits/s. DES implementation at [11] implement both 2-stage and 4-stage pipeline approaches obtaining throughput of 183.8 Mbits/s and 402.7 Mbits/s respectively. Almost all FPGA architectures for DES implement use partially or fully pipeline approaches. Only the design in [10] is a one round DES implementation in FPGAs. A fair comparison is possible with this design only. The design was implemented on XC4020E occupying 438 CLB slices. It takes 24 cycles to complete encryption for one single data block achieving a throughput of 26.7 Mbits/s. Hence the Throughput/Area factor is 0.06. Our DES implementation improves both the area and throughput factors consuming only 165 CLB slices on XCV400 and showing a throughput of 274 Mbits/s. The Throughput/Area factor for our design is 2.34. Comparing our architecture with the design in [10], we get a speedup improvement of almost 10 times in throughput occupying four times less CLB slices. In fact our design ranks second considering as a figure of merit the Throughput/Area Factor is really convincing.

5 Conclusions

In this work, an efficient and compact DES implementation on reconfigurable hardware platforms was presented. VLSI or FPGA implementations achieve ultra high throughputs depending on the design strategy; design resources and optimization work both at algorithm and design level. From Table 1, it can be seen that our design achieved a competitive performance when compared with other reported reconfigurable hardware implementations of DES.

Our architecture can be improved to offer even better results in terms of achieved throughput. The most obvious extension is to design a fully pipelined architecture in order to obtain a higher throughput at the price of area.

References

1. Davio, M., Desmedt, Y., Goubert, J., Hoornaert, F., Quisquater, J.J.: Efficient hardware and software implementations for the DES. In: Proc. of Crypto' 83. (1984) 144–146
2. Feldmeier, D.C. A high speed crypt program (1989) Technical Memo TM-ARH-013711.
3. Karn, P.R. (Karns DES implementation source code)

4. Bishop, M.: An application of a fast data encryption standard implementation. In: Computing Systems, 1(3). (1988) 221–254
5. Biham, E.: A fast new DES Implementation in Software. In: 4th Int. Workshop on Fast Software Encryption, FSE97, Haifa, Israel, Springer-Verlag, 1997 (1997) 260–271
6. Eberle, H., Thacker, C.: A 1 Gbit/second GaAs DES chip. In: Proc. IEEE 1992 Custom Integrated Circuits Conference, New York,USA, Springer-Verlag, 1992 (1992) 19.7/1–4
7. Eberle, H.: A high speed DES implementation for network applications. In: Advances in Cryptology-CRYPTO'92, Lecture Notes in Computer Science, Berlin, Germany, Springer-Verlag, 1992 (1992) 521–539
8. Wilcox, D., Pierson, L., Robertson, P., Witzke, E.L., Gass, K.: A DES asic suitable for network encryption at 10 Gbs and beyond. In: CHESS 99, LNCS 1717 (1999) 37–48
9. Leonard, J., Magione-Smith, W.: A case study of partially evaluated hardware circuits: key specific des. In: Proc. Field-Programmable Logic and Applications, FPL' 97, London, UK, Springer-Verlag, 1997 (1997) 234–247
10. Wong, K., Wark, M., Dawson, E.: A Single-Chip FPGA Implementation of the Data Encryption Standard (des) Algorithm. In: IEEE Globecom Communication Conf., Sydney, Australia (1998) 827–832
11. Kaps, J., Paar, C.: Fast DES implementations for FPGAs and its application to a Universal key-search machine. In: Proc. 5th Annual Workshop on selected areas in cryptography-Sac' 98, Ontario, Canada, Springer-Verlag, 1998 (1998) 234–247
12. Core(2000), F.D.: (2000) URL: http://www.free-ip.com/DES/.
13. McLoone, M., McCanny, J.: High-performance FPGA implementation of DES using a novel method for implementing the key schedule. IEE Proc.: Circuits, Devices & Systems **150** (2003) 373–378
14. Patterson, C.: High Performance DES Encryption in Virtex FPGAs using Jbits. In: Field-programmable custom computing machines, FCCM' 00, Napa Valley, CA, USA, IEEE Comput. Soc., CA, USA, 2000 (2000) 113–121
15. NIST: Announcing the ADVANCED ENCRYPTION STANDARD(AES). Federal Information Standards Publication (2001)
16. X9.62, A. Federal Information Processing Standard (FIPS) 46, National Bureau Standards (1977)
17. (Revised):, A.X. National Standards for financial institution key management (wholesale), American Bankers Association (1986)
18. 8732:, I.D. Banking-key management (wholesale), Association for Payment Clearing Services (1987)
19. Schneier, B.: Applied Cryptography: Protocols, Algorithms, and Source Code in C. John Wiley & Sons, New York (1996)
20. Menezes, A., Oorschot, P.V., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton, FL (1997)
21. Trappe, W., Washington, L.: Introduction to Cryptography with Coding Theory. Prentice Hall, Inc., Upper Saddle River, NJ 07458 (2002)