IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 10, 2019, accepted May 28, 2019, date of publication June 17, 2019, date of current version July 22, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2923440

High Throughput Implementation of SMS4 on FPGA

JUN ZHAO, ZHICHUAN GUO, AND XUEWEN ZENG

National Network New Media Engineering Research Center, Institute of Acoustics, Chinese Academy of Sciences, Beijing 100190, China
School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100190, China

Corresponding author: Zhichuan Guo (guozc@dsp.ac.cn)

This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDC02010701.

ABSTRACT The SMS4 algorithm is a block cipher algorithm, which has the characteristics of high security and easy implementation. However, the optimization and implementation schemes proposed for FPGA platform currently use multi-channel parallel and pipelined architectures to improve performance, which results in a large consumption of resources, and the clock cycles taken to process a single data block is not reduced. This paper proposes a novel implementation scheme of SMS4 on FPGA. This scheme separates the generations of 32 round keys and encryption operations, 32 round keys are generated on the host computer in advance, and the encryption operations completed on the FPGA. At the same time, for the 32-round iterative structure of the SMS4, this paper proposes a dual-cascade implementation architecture that can compress 32 rounds of iterative operations from 32 clock cycles to 16 clock cycles. This greatly improves the performance of the SMS4. To compare with the previous works needing 32 cycles or more, which greatly reduces the clock cycles spent on processing each data block. The throughput achieves 1.9 Gbps at a frequency of 286 MHz on Xilinx FPGA.

INDEX TERMS SMS4, FPGA, dual-cascade, block cipher.

I. INTRODUCTION

The rapid development of network technology has also brought about a variety of information security issues [1]. In order to ensure information security, various solutions have been proposed. At present, the most widely used one is the cryptographic technology [2], which encrypts data through various cryptographic algorithms to ensure the security of information. There are three major types of cryptographic algorithms, asymmetric cryptographic algorithms [3], HASH algorithm [4], and symmetric cryptographic algorithms [5]. Asymmetric algorithms, also named public key cryptographic algorithm, are mainly used for identity authentication, such as ECC and RSA [6], and Hash algorithms are for ensuring message integrity, like SHA256 [7]. Symmetric algorithms are widely used to encrypt large data, which plays a vital role in the field of information security. AES [8] is one of the most widely used algorithms. This type of algorithm can achieve high performance when encrypting data, and it also has reliable security.

The SMS4 block cipher algorithm [9] is designed as commercial block cipher algorithm, which is officially released by the Office of State Commercial Cryptography Administrator in China in March 2012. The algorithm was originally applied to WLAN, and then for its properties of high security and easy implementation, it is widely used in fields of IoT, smart cards and others industries, too. Similar to AES128, the length of block and key also is 128 bits in SMS4. And the process of encryption, decryption and key expansion consists of 32 rounds nonlinear iteration. To meet the scenario of high throughput application, this algorithm has been implemented on various platforms, such as GPU, ASIC and FPGA, and achieves high performance. In this paper, we propose a novel design of SMS4 on FPGA, and the implementation of SMS4 based on this design reaches higher resource and time utilization. In this design, we optimize SMS4 from two aspects of 32 round keys' generation and 32-round iterative architecture of encryption operation. For round keys, 32 round keys are generated in host computer in advance, and then, they are transferred to round key registers on FPGA. For 32-round iterative structure of encryption process, we propose dual-cascade architecture, which can complete 32-round iteration in 16 cycles. That reduces the cycles required to process each data block.

The content of this paper is organized as follows: The second section reviews the related work about design

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Imran Tariq.

and optimization of SMS4. The third section details the SMS4 algorithm. The fourth section analyzes the rationality of the implementation architecture proposed in this paper. The fifth section implements the design on FPGA platform. In the sixth section, the performance of dual-cascade design is compared with other works, and the resource occupied by this design is analyzed. In seventh section, the conclusion about this design is drawn.

II. RELATED WORK

The optimization and implementation of SMS4 on various platforms has been researched. In [10], the mathematical theory of SMS4 is analyzed. In [11], the throughput of unrolling architecture implementation for SMS4 achieved 20736 Mbps on Stratix II FPGA, but 8373 ALMs are occupied. The pipelined architecture is introduced in [12], this architecture is similar to unrolling architecture proposed in [11], and the performance of SMS4 on Xilinx Virtex-4 FPGA reached 24320 Mbps, using 9500 CLB slices. The parallelizing design of SMS4 on UMC 180nm ASIC technology is proposed in [13], that can achieve 136.9 Gbps, it also uses a lot of resources, up to 11574 GEs. The low-cost reconfigurable VLSI implementation of SMS4 in [14] is implemented with SMIC 0.13um CMOS technology, the area is 22k equivalent gates, and the throughput is 800 Mbps. The design based on trade-off between area and speed is introduced on Altera Stratix IV in [15], the maximum throughput achieved 27 Gbps. The pipeline architecture for SMS4-GCM proposed in [16] reached 22 Gbps on Xilinx Virtex5xc5vlx50t, and that needs 16712 LUTs. By analyzing the implementation schemes mentioned above, it can be found that the main idea adopted to improve performance are multiple parallel processing and pipelined. That leads to a significant increase in the resources consumed, but the clock cycles taken to process a single block is not reduced. The implementation scheme of dual-cascade proposed in this paper can greatly reduce the cycle-consuming processing of each block on FPGA, and greatly improve the performance without occupying too much resources.

III. OVERVIEW OF SMS4 ALGORITHM

SMS4 is a block cryptographic algorithm, the length of its block and key are 128bits. Both encryption algorithm and key expansion algorithm adopt a nonlinear iterative structure, and require 32 rounds of operations. The algorithm of decryption and encryption have the same structure, except that the usage order of round keys is reversed, the decryption round keys are the reverse order of the encryption round keys. Before the process, 32 round keys rk_i are generated by the key expansion algorithm, and then rk_i are sequentially input to the round function F in the 32 rounds of iterations. 32 round keys rk_i are generated by key expansion algorithm, the functions used by key expansion function are as follows.

Permutation function is shown in algorithm 1.

The permutation function T is composed of two operations: a nonlinear transformation τ, which consists of four parallel

Algorithm 1 Permutation Function T

Input: $A = (a_0, a_1, a_2, a_3) \in (Z_2^8)^4$;
Output: $C \in Z_2^{32}$;
1: $B = \tau(A) = (Sbox(a_0), Sbox(a_1), Sbox(a_2), Sbox(a_3))$;
2: $C = L(B) = B \oplus (B <<< 2) \oplus (B <<< 10) \oplus (B <<< 18) \oplus (B <<< 24)$;
3: return C;

S-box lookup tables, where S-box lookup table is a table of 8-bit input and 8-bit output. The primitives of this table are constant. S-box lookup table is shown in Figure 1.

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	d6	90	e9	fe	cc	e1	3d	b7	16	b6	14	c2	28	fb	2c	05
1	2b	67	9a	76	2a	be	04	c3	aa	44	13	26	49	86	06	99
2	9c	42	50	f4	91	ef	98	7a	33	54	0b	43	ed	cf	ac	62
3	e4	b3	1c	a9	c9	08	e8	95	80	df	94	fa	75	8f	3f	a6
4	47	07	a7	fc	f3	73	17	ba	83	59	3c	19	e6	85	4f	a8
5	68	6b	81	b2	71	64	da	8b	f8	eb	0f	4b	70	56	9d	35
6	1e	24	0e	5e	63	58	d1	a2	25	22	7c	3b	01	21	78	87
7	d4	00	46	57	9f	d3	27	52	4c	36	02	e7	a0	c4	c8	9e
8	ea	bf	8a	d2	40	c7	38	b5	a3	f7	f2	ce	f9	61	15	a1
9	e0	ae	5d	a4	9b	34	1a	55	ad	93	32	30	f5	8c	b1	e3
a	1d	f6	e2	2e	82	66	ca	60	c0	29	23	ab	0d	53	4e	6f
b	d5	db	37	45	de	fd	8e	2f	03	ff	6a	72	6d	6c	5b	51
c	8d	1b	af	92	bb	dd	bc	7f	11	d9	5c	41	1f	10	5a	d8
d	0a	c1	31	88	a5	cd	7b	bd	2d	74	d0	12	b8	e5	b4	b0
e	89	69	97	4a	0c	96	77	7e	65	b9	f1	09	c5	6e	c6	84
f	18	f0	7d	ec	3a	dc	4d	20	79	ee	5f	3e	d7	cb	39	48

FIGURE 1. S-box lookup table.

In Fig. 1, it can be seen that this is a 16-row, 16-column table, there are a total of 256 elements in this table. The possible values of row coordinates and column coordinates are from 0 to 15. The data in the S-box is expressed in hexadecimal, the top row and the leftmost column in S-box are the possible values for the inputs. In order to understand the search process, we give an example of search: when the input is 32, the output is the corresponding element of the 3rd row and the 2nd column in S-box, here is 1c.

The linear transformation L in algorithm 1 consists of two operations of shifting and exclusive-OR, where $<<<$ is defined as 32-bit loop left shift and $\oplus$ is 32-bit XOR. The output of τ transformation is used as the input of L transformation.

Round function is shown in algorithm 2.

Algorithm 2 Round Function F

Input: $X = (X_0, X_1, X_2, X_3) \in (Z_2^{32})^4$;
Output: $Y \in Z_2^{32}$;
1: $TMP = T(X_1 \oplus X_2 \oplus X_3 \oplus rk_i)$;
2: $Y = X_0 \oplus TMP$;
3: return Y;

The round function F is the most important transformation in SMS4, the properties of security and easy implementation of SMS4 is determined by F. It is used in key expansion

function and encryption function, these related algorithms are described as follows.

Key expansion function is shown in algorithm 3.

Algorithm 3 Key Expansion Function

Input: $MK = (MK_0, MK_1, MK_2, MK_3) \in (Z_2^{32})^4$;
Output: $rk_i \in Z_2^{32}, i \in (0, 1, \ldots, 31)$;
1: $(K_0, K_1) = (MK_0 \oplus FK_0, MK_1 \oplus FK_1)$;
2: $(K_2, K_3) = (MK_2 \oplus FK_2, MK_3 \oplus FK_3)$;
3: for $i = 0; i < 32; i ++$ do
4: $rk_i = K_{i+4} = K_i \oplus T'(K_{i+1} \oplus K_{i+2} \oplus K_{i+3} \oplus CK_i)$;
5: endfor
6: return rk_i;

In key expansion algorithm,FK_0, FK_1, FK_2 and FK_3 are four system parameters, and their values are set as follows:

$$FK_0 = A3B1BAC6 \quad (1)$$
$$FK_1 = 56AA3350 \quad (2)$$
$$FK_2 = 677D9197 \quad (3)$$
$$FK_3 = B27022DC \quad (4)$$

The function T' in algorithm 3 is similar to function T, the only difference is that L(B) in T is replaced by $L'(B)$ in T', which is shown in equation 5.

$$L'(B) = B \oplus (B <<< 13) \oplus (B <<< 23) \quad (5)$$

In key expansion function,$B \in Z_2^{32}$,$<<<$ is defined as 32-bit loop left shift, $\oplus$ is defined as 32-bit XOR.CK_i is a fixed set of parameters, the process of CK_i generation is shown in algorithm 4.

Algorithm 4 CK_i Generation Function

Input: i, j;$i \in (0, 1, \ldots, 31)$, $j \in (0, 1, 2, 3)$;
Output: $CK_i \in Z_2^{32}, i \in (0, 1, \ldots, 31)$;
1: $(K_0, K_1) = (MK_0 \oplus FK_0, MK_1 \oplus FK_1)$;
2: $(K_2, K_3) = (MK_2 \oplus FK_2, MK_3 \oplus FK_3)$;
3: for $i = 0; i < 32; i ++$ do
4: for $j = 0; j < 4; j ++$ do
5: $ck_{i,j} = (4 \times i + j) \times 7 (mod 256)$;
6: endfor
7: $CK_i = (ck_{1,0}, ck_{1,1}, ck_{1,2}, ck_{1,3})$
8: endfor
9: return CK_i;

Since the values of CK_i are constant, when CK_i are used in SMS4, they can be generated in advance and directly called during the process of key expansion function and encryption function. With the algorithm 3 and algorithm 4, 32 round keys rk_i can be generated, and then rk_i would be used in encryption function. The process of encryption is described in algorithm 5.

From algorithm 5, we can know that encryption function is composed of 32 rounds of iterations by round function with 32 round keys. As depicted in algorithm 5, the ciphertext is

Algorithm 5 Encryption Function

Input: $X = (X_0, X_1, X_2, X_3) \in (Z_2^{32})^4$, $rk_i \in Z_2^{32}, i \in (0, 1, \ldots, 31)$;
Output: $Y = (Y_0, Y_1, Y_2, Y_3) \in (Z_2^{32})^4$;
1: for $i = 0; i < 32; i ++$ do
2: $X_{i+4} = F(X_i, X_{i+1}, X_{i+2}, X_{i+3}, rk_i)$;
3: endfor
4: $Y = (Y_0, Y_1, Y_2, Y_3) = (X_{35}, X_{34}, X_{33}, X_{32})$;
5: return Y;

the reverse order of the output of last four rounds, and the process of decryption also requires the above 32 rounds of iterative operations, but the use order of round keys in which is reverse to the encryption process.

IV. ANALYSIS OF NOVEL ARCHITECTURE

From the implementation analysis of SMS4 mentioned in section II, we can find these following two phenomena. First, the mainstream optimization and implementation scheme are based on parallel and pipeline. Second, for the multiplexing of functional modules, both round key generation and encryption operation are completed in the same module, and the desired function is selected by a control signal. That leads to an increase in resource consumption and also makes the implementation more complicated. In view of the two problems above, we take the following two measures to optimize the implementation of SMS4 on FPGA. First, we separate modules of cryptographic operation and key expansion. Second, for cryptographic operation, we introduce the dual-cascade architecture. The details are as follow.

A. FAST GENERATION OF ROUND KEYS

Since the key used in SMS4 is fixed during the entire encryption or decryption process, 32 round keys generated with key are also fixed. When round keys are generated on FPGA, since that process also undergoes 32 rounds of iterative operation, this requires 32 or more cycles. And the FPGA clock frequency is low (only a few hundred megahertz), that causes a large delay, so we could transfer the key expansion function illustrated in section III to the host computer, and after 32 round keys are generated, these round keys can be saved to round key registers on FPGA. The design of key expansion in host computer is depicted in Figure 2.

In Fig. 2, FK and CKi are constant parameters, MK is key, they are all 128 bits. The Round function in the above figure is shown in Figure 3.

The operation process is described as follows.

Step 1, letting $MK \oplus FK$ to obtain a 128-bit intermediate result and assigning this result to K_0, K_1, K_2 and K_3.

Step 2, inputting $K_0K_1K_2K_3$ and CK_0 to Round0 and letting $K_1 \oplus K_2 \oplus K_3 \oplus CK_i$ to get a 128-bit intermediate value.

Step 3, inputting the value get in step 2 to 4 Sboxs and using the output of τ transformation as input of $L'(B)$ transformation in Fig. 3.

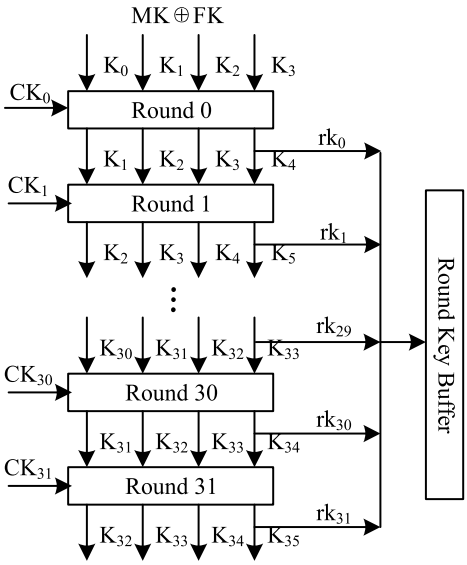


FIGURE 2. 32 Round keys generation.

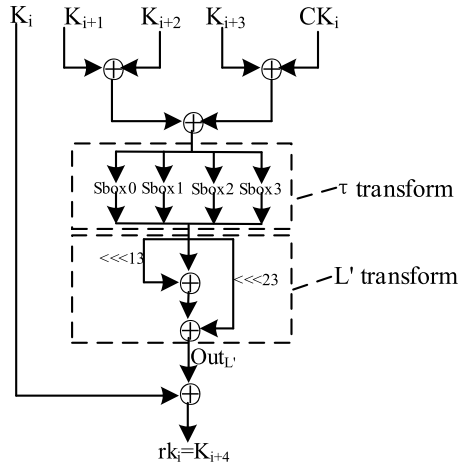


FIGURE 3. Round function in key expansion.

Step 4, letting $\mathrm{Out}_{L'} \oplus K_0$ and assigning the result of output K_4 to rk_0, then storing rk_0 to round key buffers, where $\mathrm{Out}_{L'}$ is the output of $L'(B)$ transformation in Fig. 3.

Step 5, inputting $K_1K_2K_3K_4$ and CK_1 into Round1 and assigning output K_5 to rk_1, then storing rk_1 to round key buffers.

The other process to generate rk_2 to rk_{31} is similar to rk_0. After 32 round keys are generated, they are transferred to the round key registers on FPGA. For the clock frequency of the host computer is much higher (up to several gigahertz) than that on FPGA, the time to generate 32 round keys in host computer is far less than that needed if round keys are generated on FPGA. More importantly, it will not overload the host computer in the meanwhile.

B. DUAL-CASCADE SCHEME OF CRYPTOGRAPHIC OPERATION

The most time consuming step in the SMS4 algorithm is the 32 iterations of the round function. Currently, in the mainstream design scheme, the round function is implemented by a sequential logic circuit or a combinational logic circuit on FPGA. The former one splits one operation of the round function into multiple cycles, and resources required are a few. The latter one can complete one round function in one clock cycle, but resources required are a little more. For improving the performance, combinational logic circuits are used in our design.

Although the input signal and output signal of the combinational logic circuit are theoretically changed at the same time, in the actual circuit, routine and the logic device itself have delays. The combinational logic circuit does always need a delay from receiving a valid input signal to generating a stable output signal, although this delay is very small. Compared with the delay of the combinational logic circuit, one clock cycle on FPGA is much longer than the delay. What's more, in some complicated system designs, there are many functional modules which may cause the system clock to perform operations at a lower frequency, so one clock cycle will take longer time. In this case, if only one round function is performed in one cycle, the design scheme could not reach high performance.

In our design, the plaintext and ciphertext required by SMS4 algorithm need to interact with the host computer through the AXI4 and PCIe bus, for improving the overall throughput of SMS4, we use the clock signal of AXI4 bus as the driving clock of SMS4. The engineering block diagram is shown in Figure 4.

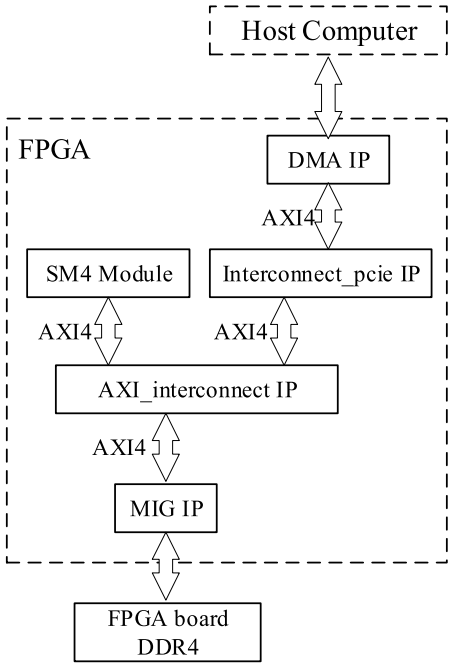


FIGURE 4. Overall engineering block diagram.

In Fig. 4, the clock signal of AXI4 bus is 286 MHz, which is 3.50ns for one clock cycle. In the scenario mentioned above, for the sake of analysis, we introduce the following variables to represent various delays, where $D_{total-1}$ is used

to represent the total delay of single round function from receiving input signals to generating valid and stable output signals, $D_{total-2}$ is used to represent the total delay of dual round functions, T_{period} represents the clock cycle of the system, here which means the cycle of AXI4's clock signal. After synthesis and implementation by Vivado, we get that the two delays for single and dual round functions. Therefore $D_{total-1}$, $D_{total-2}$ and T_{period} have the following relationships:

$$T_{period} = 3.50ns \tag{6}$$
$$D_{total-1} = 2.14ns \tag{7}$$
$$D_{total-2} = 3.31ns \tag{8}$$

From the above equation (6), (7) and (8), the following inequation can be derived:

$$D_{total-1} < D_{total-2} < T_{period} \tag{9}$$

The above inequation (9) means that although the delay of the dual round functions is slightly larger than the delay of single round function, the two delays $D_{total-2}$ and $D_{total-1}$ are smaller than AXI4's clock cycle T_{period}. So both schemes could work under the current clock.

However, due to the round function completed by dual-cascade architecture is twice that of single-cascade, under the current clock, the overall performance is twice as much as that reached by single-cascade scheme.

The single-cascade implementation architecture of SMS4 is illustrated in Figure 5.

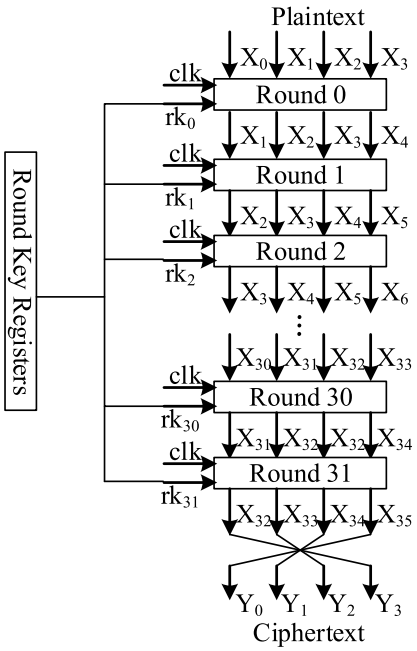


FIGURE 5. Single-cascade architecture of SMS4.

In Fig. 5, the plaintext input and ciphertext output are 128bits. In this design, permutation function T and S-box LUT mentioned in section III are all implemented in the form of combinational logic circuits instead of sequential logic circuits, which can simultaneously produce a useful output signal when the input signal is valid and stable. The Round function in Fig. 5 is implemented according to Figure 6.

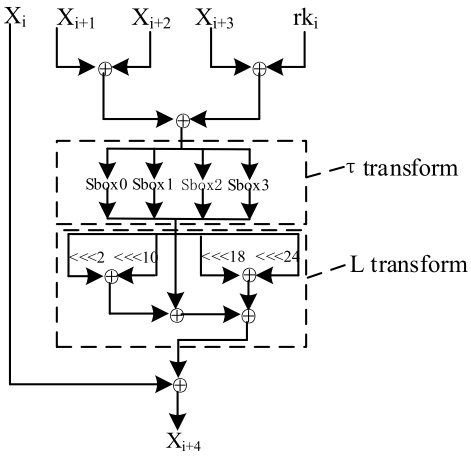


FIGURE 6. Round function in encryption process.

In Fig. 6, the delay of round function includes 6-level gate and 1-level lookup table. Before starting encryption process, the 32 round keys are generated in advance and transmitted to round key registers from host computer. The single-cascade architecture performs round function once in one clock cycle, such as Round0 completed in the first cycle, Round1 in the second cycle, Round2 in the third cycle and so on. For the sake of understanding, we give an example to illustrate the specific implementation process. When the rising edge of the clock arrives, the 128-bit plaintext data $X_0X_1X_2X_3$ and the round key rk_0 are input to Round0. Since round function is implemented in the form of combinational logic circuits, the output X_4 can be obtained when next rising edge of the clock arrives. At the same time, $X_1X_2X_3X_4$ would be used as Round1's inputs. And then $X_2X_3X_4X_5$ are used as Round2's inputs when the third rising edge of the clock arrives. After 32 clock cycles, 128 bits cipher text can be obtained. Based on the analysis above, this single-cascade scheme requires 32 clock cycles to process one block, so its performance is low.

Therefore, in order to improve the performance of SMS4, we introduce the dual-cascade implementation scheme, which can perform another round function in one cycle after the first round function is completed. The dual-cascade implementation architecture of SMS4 is illustrated in Figure 7.

In Fig. 7, the plaintext input is 128bits and cipher text output is 128bits, too. In order to allow the operation of round function to be completed in one clock cycle, as in Fig. 6, the round function is also implemented by a combinational logic circuits. The two round functions in the dashed box in Fig. 7 are completed in one clock cycle, and the implementation scheme is shown in Figure 8.

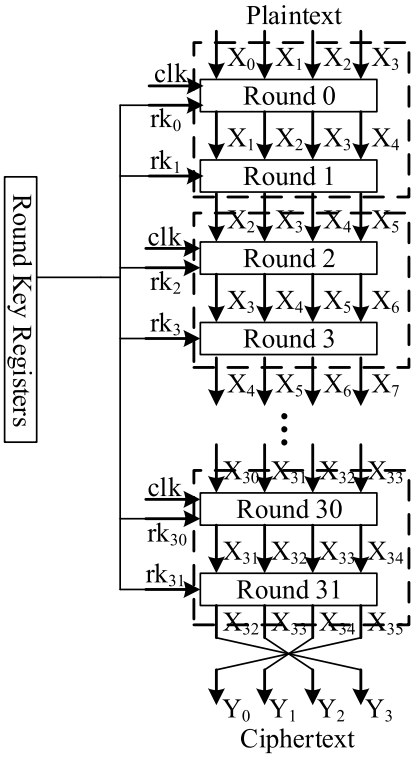


FIGURE 7. Dual-cascade architecture of SMS4.

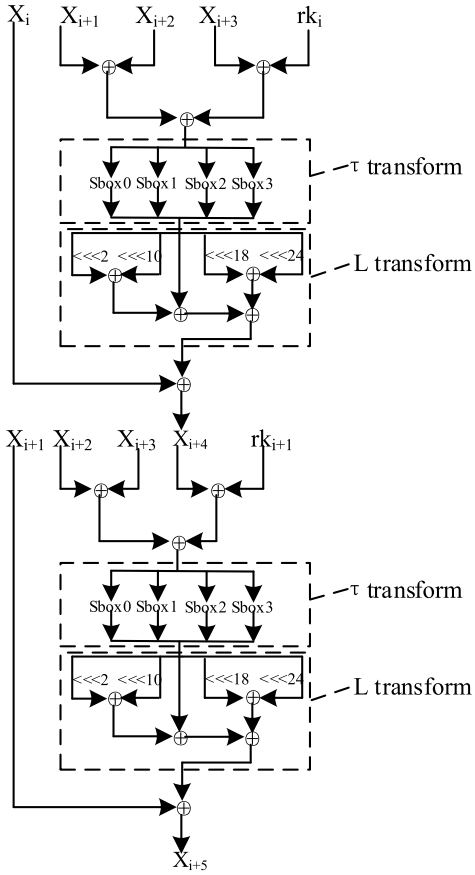


FIGURE 8. Dual-round function in encryption process.

In Fig. 8, the delay of dual round functions includes 12-level gate and 2-level lookup table. It is shown that X_{i+4}, the output of 1st round function, is directly used as the input of 2nd round function. When X_{i+5}, output of 2nd round function, is generated, it would be stored to register.

This dual-cascade architecture can perform round function twice in one clock cycle, such as Round0 and Round1 completed in the first cycle, Round2 and Round3 in the second cycle and so on. The following is an example of execution process in the first clock cycle, when the rising edge of the clock arrives, the 128-bit plaintext data $X_0X_1X_2X_3$ and the round key rk_0 are input to Round0, And then $X_1X_2X_3X_4$ and rk_1 are directly input into Round1 simultaneously. From inequation (9) and our analysis above, it is conclude that when X_5 are generated stably, the next rising edge of the clock does not arrive. When the next rising edge of clock arrives, $X_2X_3X_4$ and X_5 are restored to registers. And Round2 uses $X_2X_3X_4$ and X_5 stored in registers as Round2's inputs at the same time, after 16 cycles, $X_{32}X_{33}X_{34}$ and X_{35} are obtained. Then $X_{32}X_{33}X_{34}$ and X_{35} are assigned to $Y_3Y_2Y_1$ and Y_0 respectively, here the cipher text is obtained. In this case, this dual-cascade scheme requires 16 clock cycles to process one block, so its performance is twice as much as single-cascade, which need 32 cycles to process one block.

C. COMPETITION AND RISK ANALYSIS OF CASCADED LOGIC CIRCUITS

Since the transmission delay of signals cannot be avoided, the adventure and competition phenomenon must also be considered in the design of the combinational logic circuit. The following is the feasibility analysis of the dual-architecture proposed in this paper under the conditions of competition and adventure.

In our implementation, the input signals are driven by registers, it can continuously provide a stable signal. So even if the arrival time of each signal is inconsistent, and the output signal will be unstable and jittered before the next rising edge of the clock arrives, during this period, the output signals will not be sampled and stored. From the delay relationship (9) analyzed above, the output signal has been valid and stable when the rising edge comes. At this time, storing these output results not affect the next round of operations. Since the cascaded architecture of round function in this implementation has only two cascades, inputs of round function are driven by register, and output signals are stored in registers when the rising edge of the clock arrives, the problem of erroneous transmission and accumulation due to competition and adventure of the combinational circuit can be effectively avoided.

D. PERFORMANCE OF SMS4 BASED ON DUAL-CASCADE DESIGN

The encryption process requires 32 clock cycles originally, but in our architecture, it takes only 16 cycles, the cycles

TABLE 1. Performance comparison between this design and other implementations on Xilinx FPGA.

Design	[12]	[16]	[18]	Our work			
Device	Virtex-4	Virtex-5	Virtex-6	Virtex-4	Virtex-5	Virtex-6	Xcku 115
Resources	380 CLB slices	-	-	3179 LUTs 1352 Regs	1253 LUTs 1351 Regs	1350 LUTs 1350 Regs	1365 LUTs 1351 Regs
NC	36	32	32	16	16	16	16
Frequency (MHz)	185	174	332	123	155	214	286
Throughput (Mbps)	740	695.25	1330	817	1030	1422	1900

TABLE 2. Performance comparison between this design and other implementations on Altera FPGA.

Design	[11]	[15]	[17]	[19]	Our work
Device	Stratix II	Stratix IV	Stratix II	Stratix II	Xcku 115
Resources	1552 ALMs	687 LUs 448 Regs	1150 ALMs	946 ALUTs	1365 LUTs 1351 Regs
NC	33	33	33	32	16
Frequency (MHz)	139	210.26	189	141	286
Throughput (Mbps)	539	815.6	733	564	1900

consumed is reduced by half. The throughput of SMS4 is calculated according to the following formula

$$throughput = \frac{B \times f}{N} \tag{10}$$

where *throughput* stands for the throughput of SMS4, B stands for the size of one block in SMS4, f stands for the frequency of system clock, and N stands for the number of clock cycles required to process each block. It can be deduced from the analysis and formula above that, in the case where B and f are unchanged, *throughput* is inversely proportional to N, so when N is 16, the performance of SMS4 is two times of that when N is 32.

V. IMPLEMENTATION OF SMS4 ON FPGA

The FPGA selected in this work is Xilinx Kintex UltraScale XCKU115-flva1517-2-e, and the software for simulation, synthesis and download are Xilinx Vivado 2018.3 and Xilinx ISE 14.7, the design of SMS4 described in this paper are coded using Verilog HDL.

The integrated result of design proposed is depicted in table 1 and 2. Table 1 listed the main implementations and comparison with prior works on Xilinx FPGA. Table 2 listed other implementations and comparison on Altera FPGA. In these two tables, Design is scheme proposed in related literatures, Device is FPGA platform, Resources are the logic resource on FPGA, NC is the number of clock cycles required to process a block, Frequency is the frequency of clock cycle, and Throughput is the throughput of SMS4.

In Tab. 1 and 2, the implementations mentioned in other related literatures improve performance by multi-stage pipeline or multiple parallel. In order to facilitate analysis and comparison, we normalize them to a one-stage pipeline or a single-channel work mode. Our work is based on dual-cascade architecture, this work is optimized using fast generation of round keys, and encryption process takes dual-cascade architecture, which uses 16 cycles to complete 32 iterations of round function. As can been seen from Tab. 1 that the proposed scheme in this paper achieves higher performance than the other schemes on Virtex-4, 5 and 6 series. At the same time, the resources occupied on different platforms are similar, since there are only 4-input LUTs on Virtex4, the resources used are higher than other devices with 6-input LUTs. In addition, as shown in Tab. 2, compared with other implementations on Altera FPGA, the scheme proposed in this paper is much higher in performance than others.

We focus on the number of cycles required to process each block and the number of bits processed in each cycle, the comparison are shown in Figure 9 and 10.

In Fig. 9, we can find that the implementations proposed in the other literatures require at least 32 cycles to process

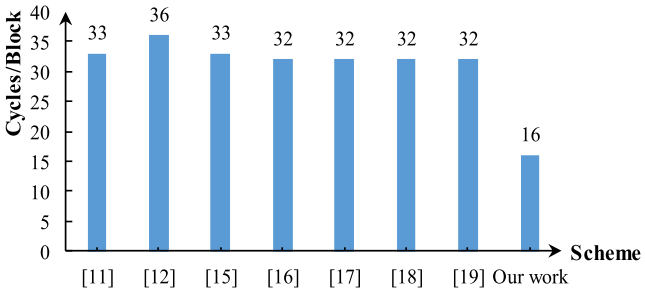


FIGURE 9. Number of clocks required for one block.

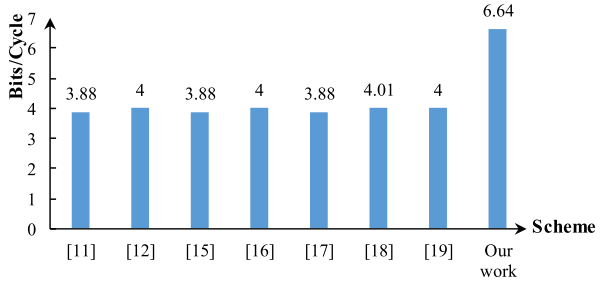


FIGURE 10. Number of bits processed per cycle.

each block. In our work, due to the dual-cascade architecture is introduced, only 16 cycles are required during process of each block.

The number of bits that can be processed in each cycle of each implementation is shown in Figure 10.

In Fig. 10, we can find that our work can process 6.64 bits in each cycle, which is much higher than the 4 bits processed by other implementations. Therefore, higher performance can be achieved in our dual-cascade scheme.

As can be seen from the above tables and figures that under the condition of similar working frequency, the throughputs achieved by each of other implementations are similar. The design in [18] with higher performance is due to its higher working frequency. By analyzing the resource utilization report, we find that the LUT required for dual-cascade architecture is increased by 182 compared to single-cascade scheme, and the register is increased by 28. In the case that the required resources are not increased much, the throughput is doubled. It can be seen that the dual-cascade architecture proposed in this paper is feasible and effective to improve performance.

VI. CONCLUSION

Based on the FPGA platform, this paper proposes a high-throughput implementation architecture of SMS4 algorithm. In our single-cascade scheme, the key expansion operation completed by the FPGA before is transfer to the host computer. After the host computer generates the round keys, they are set to the round key registers on FPGA. This design fully utilizes the high frequency of the host computer and effectively shortens the time generating round keys without occupying too much resources of the host computer. On the basis of this scheme, the dual-cascade architecture is introduced for 32 rounds of iterative operations involved in the SMS4 algorithm. This architecture works by performing two round functions in one cycle, and there are very few additional resources that need to be added (less than 10%). In this case, 32 rounds of iterative operations in the cryptographic operation can be compressed from 32 clock cycles required previously to 16 cycles needed now. The scheme greatly improves the performance of the SMS4 algorithm.

Compared with the implementation schemes mentioned in other literatures, the optimization scheme proposed in this paper greatly increases the number of bits that can be processed per cycle, so that the higher throughput is achieved.

REFERENCES

[1] R. E. Crossler, A. C. Johnston, P. B. Lowry, Q. Hu, M. Warkentin, and R. Baskerville, "Future directions for behavioral information security research," *Comput. Secur.*, vol. 32, pp. 90–101, Feb. 2013. doi: 10.1016/j.cose.2012.09.010.

[2] F. Piper and S. Murphy, "Uses of cryptography," in *Cryptography: A Very Short Introduction*, 1st ed. New York, NY, USA: OUP, 2002, pp. 85–107.

[3] I. N. Kovalenko and A. I. Kochubinskii, "Asymmetric cryptographic algorithms," *Cybern. Syst. Anal.*, vol. 39, no. 4, pp. 549–554, Jul. 2003.

[4] X. Y. W and H. B. Y, "Survey of hash functions," *J. Inf. Secur. Res.*, vol. 1, no. 1, pp. 19–30, Oct. 2015.

[5] G. J. Simmons, "Symmetric and asymmetric encryption," *Comput. Surv.*, vol. 11, no. 4, pp. 305–330, Dec. 1979.

[6] A. S. Wander, N. Gura, H. Eberle, V. Gupta, and S. C. Shantz, "Energy analysis of public-key cryptography for wireless sensor networks," in *Proc. 3rd IEEE Int. Conf. Pervasive Comput. Commun.*, Kauai Island, HI, USA, Mar. 2005, pp. 324–328.

[7] S.-H. Lee and K.-W. Shin, "An efficient implementation of SHA processor including three hash algorithms (SHA-512, SHA-512/224, SHA-512/256)," in *Proc. ICEIC*, Honolulu, HI, USA, Jan. 2018, pp. 1–4.

[8] X. Zhang and K. K. Parhi, "High-speed VLSI architectures for the AES algorithm," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 9, pp. 957–967, Sep. 2004. doi: 10.1109/TVLSI.2004.832943.

[9] *SM4 Block Cipher Algorithm*, document GM/T 0002-2012, 2012.

[10] F. Liu, W. Ji, L. Hu, J. Ding, S. Lv, A. Pyshkin, and R.-P. Weinmann, "Analysis of the SMS4 block cipher," in *Proc. ACISP*, Townsville, QLD, Australia, 2007, pp. 158–170.

[11] X. Gao, E. Lu, L. Xian, and H. Chen, "FPGA implementation of the SMS4 block cipher in the Chinese WAPI standard," in *Proc. Int. Conf. Embedded Softw. Syst. Symposia*, Sichuan, China, Jul. 2008, pp. 104–106.

[12] Y. Jin, H. Shen, and R. You, "Implementation of SMS4 block cipher on FPGA," in *Proc. 1st Int. Conf. Commun. Netw. China*, Beijing, China, Oct. 2006, pp. 1–4.

[13] H. Yap, K. Khoo, and A. Poschmann, "Parallelizing the Camellia and SMS4 block ciphers," in *Proc. AFRICACRYPT*, Stellenbosch, South Africa, 2010, pp. 387–406.

[14] W. Yan, K. You, J. Han, and X. Zeng, "Low-cost reconfigurable VLSI implementation of the SMS4 and AES algorithms," in *Proc. IEEE 8th Int. Conf. ASIC*, Changsha, China, Oct. 2009, pp. 135–138.

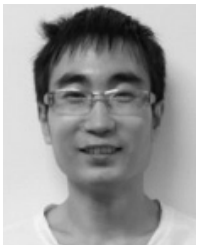
[15] Z. Guan, Y. Li, T. Shang, J. Liu, M. Sun, and Y. Li, "Implementation of SM4 on FPGA: Trade-off analysis between area and speed," in *Proc. IEEE Int. Conf. Intell. Saf. Robot. (ISR)*, Shenyang, China, Aug. 2018, pp. 192–197.

[16] M. Zhao, G. Shou, Y. Hu, and Z. Guo, "High-speed architecture design and implementation for SMS4-GCM," in *Proc. 3rd Int. Conf. Commun. Mobile Comput.*, Qingdao, China, Apr. 2011, pp. 15–18.

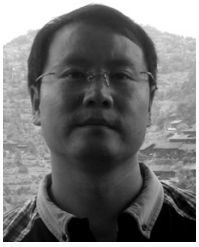
[17] W. Husen and L. Shuguo, "High performance FPGA implementation for SMS4," in *Proc. Int. Conf. High Perform. Netw., Comput. Commun. Syst.* Singapore: Singapore, 2011, pp. 469–475.

[18] A. F. Martínez-Herrera, C. Mancillas-López, and C. Mex-Perera, "GCM implementations of Camellia-128 and SMS4 by optimizing the polynomial multiplier," *Microprocessors Microsyst.*, vol. 45, pp. 129–140, Aug. 2016. doi: 10.1016/j.micpro.2016.04.006.

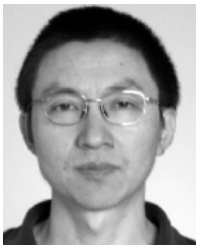
[19] X. Gao, E. Lu, L. Li, and K. Lang, "LUT-based FPGA implementation of SMS4/AES/Camellia," in *Proc. 5th IEEE Int. Symp. Embedded Comput.*, Beijing, China, Oct. 2008, pp. 73–76.



JUN ZHAO was born in Dezhou, Shangdong, China, in 1991. He received the B.S. degree from Shandong Normal University, in 2015. He is currently pursuing the Ph.D. degree with the School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, China. His current research interest includes information security.



ZHICHUAN GUO received the B.Sc. degree from Wuhan University, in 1996, the M.S. degree from the Huazhong University of Science and Technology, in 2001, and the Ph.D. degree from the University of Science and Technology of China, in 2006. From 1996 to 2003, he was an Electronics Engineer with the 13th Research Institute of China Electronics Technology Group Corporation and as a SDH Hardware R&D System Engineer of Optical Network with Huawei Company. In 2006, he joined the Institute of Acoustics, Chinese Academy of Sciences, where he is currently a Professor. He is currently involved in FPGA-based code acceleration, operation systems, and security.



XUEWEN ZENG received the B.S. degree in automatic control from Shanghai Jiao Tong University, China, in 1989, and the M.S. and Ph.D. degrees in signal and information processing from the Institute of Acoustics, Chinese Academy of Sciences, 1992, and 1997, respectively, where he is currently a Professor. He is also a Research Fellow of IACAS and a Professor with the Chinese Academy of Sciences. His research interests include multimedia communication and pattern recognition.