Design and Implementation of A High-speed Reconfigurable Multiplier

Wei LI Zi-Bin DAI Tao MENG Qiao REN

(Institute of Electronic Technology, the PLA Information Engineering University, Zhengzhou 450004, China)

Email: try_1118@ 163.com

Abstract-On the basis of analyzing the theory of multiplication operation in block ciphers and modular multiplication algorithms of different operation width, this paper present a high-speed reconfigurable multiplier, which can be reconfigured to perform 16-bit, 32-bit multiplication and modulo $2^{16}+1$ multiplication operation, and then optimize each critical block. The design is realized using Altera's FPGA. Synthesis, placement and routing of reconfigurable multiplier have accomplished on 0.18μm SMIC technology. The result proves that the propagation time of the critical path is 2.84ns. The reconfigurable multiplier is able to achieve relatively high performance in the block cipher algorithms processing.

Key words: reconfigurable, multiplier, booth algorithm, leapfrog Wallace tree, Ling adder.

1. Introduction

Block ciphers are the core mechanism of data encryption, digital signature and key management due to the characteristic of high-speed, simplified standardization and hardware realization. With the development of cipher chip design technology, the realization method of cipher algorithms is increasing gradually. The cipher processor is widely approved due to the characteristic of high-speed and flexible realization. The design scheme of reconfigurable cipher processing unit is: the hardware circuit can be reconfigured to adapt to more cipher algorithms and have a tradeoff between flexibility and efficiency.

Based on the analysis of block cipher architectures, most designs of block cipher algorithms have a few similar hardware units. In the block cipher algorithms, the frequency of multiplication operation is very high. As for different block ciphers, there are large difference in the aspects of width and module. So the multiplication unit not only takes up large area but also pays more delay time. Basing on the analysis of multiplication processing, this paper presents a high-speed reconfigurable multiplier, which can be reconfigured to perform 16-bit, 32-bit multiplication and modulo $2^{16}+1$ multiplication operation.

This paper is organized as follows: section 2 presents a reconfigurable multiplier. Section 3 optimizes the 16-bit multiplier with booth algorithm[1], leapfrog Wallace tree[2] and Ling adder[3]. Section 4 introduces a high-speed module modification circuit. Section 5 analyzes conventional adders in delay and area. Section 6 shows the simulation result. Finally, we conclude with section 7.

2. Analysis and implementation of high-speed reconfigurable multiplier

Seven algorithms of block ciphers contain modular multiplication operation. Data width is separately 16- bit, 32-bit and 64-bit. The modular multiplication is usually: $C=(A\times B) \bmod N$. Based on the analysis of block cipher, if the width of two operands is n, module N is 2^n or 2^n+1, and n is usually 16 or 32. Table 1 shows the modular multiplication operation of published block cipher algorithms.

Table 1. Modular multiplication of block cipher

Cipher algorithms	Type
IDEA	$2^{16}+1$
RC6	2^{32}
E2	2^{32}
TWOFISH	2^{32}
MARS	2^{32}
MMB	$2^{32}-1$
DFC	2^{64}

In the block cipher algorithms, the frequency of modulo 2^{32} operation is very high, however, the frequency of modulo $2^{16}+1$, $2^{32}-1$, 2^{64} operation is lower, If we design the special circuit to implement these modular multiplication operation, the circuit will take up more areas and bring more delay time. But 32-bit multiplication can be divided into 16-bit multiplication based on mathematic deduction, We can utilize 16-bit multiplier, adder array, module modification circuit and multiplexers to perform 16-bit, 32-bit multiplication and modulo $2^{16}+1$ multiplication operation. This design scheme not only minimizes the area but also extends the function of circuit.

Based on the analysis of multiplication processing, the paper puts forward a high-speed reconfigurable multiplier, as shown in Figure 1.The circuit architecture consists of 16-bit multiplier, module modification circuit and adder array. The circuit can be reconfigured to perform 16-bit, 32-bit multiplication and modulo $2^{16}+1$ multiplication operation. We optimize module modification circuit and adder array with CSA[4], Moreover, We design high performance 16-bit multiplier with booth algorithm, leapfrog Wallace tree and Ling adder. Further illustration is shown in section 3 and 4. The theory of reconfigurable multiplier is as follows：

A and B are two 32 bit datum, AH and BH denote 16 most significant bits of A and B. AL and BL denote16 least significant bits of A and B. $S_{16\times16}$ is the 16 most significant bits result of multiplication. $\overline{S}_{16\times16}$ is 16 least significant bits result of multiplication. $S_{32\times32}$ is the 32-bit result of multiplication.

$$S_{16\times16} = AH \times BH$$

$$\overline{S}_{16\times16} = AL \times BL$$

$$S_{32\times32} = AH\times BH\times 2^{32} + AH\times BL\times 2^{16} + BH\times AL\times 2^{16} + AL\times BL$$

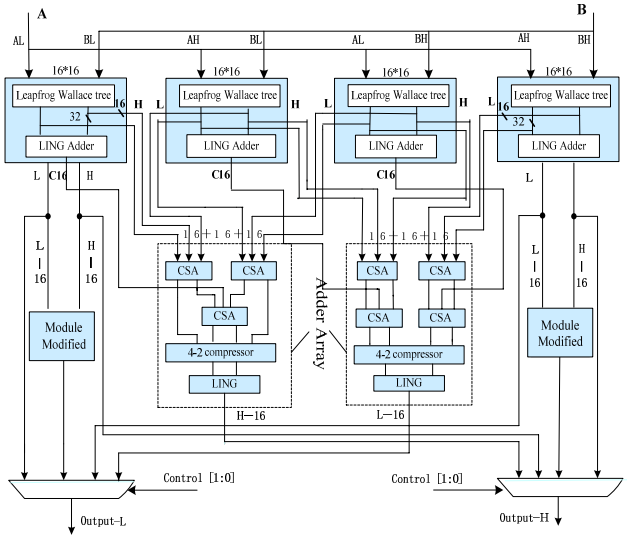

Figure 1. Architecture of reconfigurable multiplier

As shown in Figure 1, the core circuit is 16×16 multiplier. In the consideration of 32-bit multiplication operation, we design the adder array which is optimized with CSA circuit. With the wide application of IDEA algorithm, we design two module modification blocks, furthermore, 32-bit multiplication operation can be extended to implement modulo 2^{32}-1 and modulo 2^{64} multiplication operation.

Based on the analysis of the whole reconfigurable multiplier, we optimized 16-bit multiplier, module modification block and adder array block. We give detailed illustration about 16-bit multiplier and module modification block shown in section 3 and 4. The performance improvements made in adder array are as follows: we optimize the adder array with CSA and 4-2 compressor[5]. Considering the delay of adder in 16-bit multiplier, we divide the two 32-bit output signals of leapfrog Wallace tree into 16 most and least significant bits, and make them input to adder array block respectively according to the 32-bit multiplication expressions above. Moreover, we make the C16 signal (shown in Fig.1) which comes from final adder of 16-bit multiplier input to the second level of adder array. So the delay of C16 signal isn't added to the whole critical path of adder array. There is only a 16-bit adder in the adder array block. Beside that, all components are CSA. The kind of architecture has a high performance.

3 Analysis and realization of 16-bit multiplier

As the whole circuit architecture, 16-bit multiplier is the core block, but it has a long delay time. So we make the optimization to improve its performances in speed, area and power. The improved 16-bit multiplier we designed is shown in Figure 2. We consider the optimization of multiplier mainly from three parts: Firstly, compared with other booth algorithms, we adopt modified booth 2 algorithm which has a high performance of partial products generation. Secondly, the traditional Wallace tree is replaced by leapfrog Wallace tree which minimizes the delay time of partial products compression. Finally, we adopt the Ling adder based on improved CLA adder[6]. The detailed illustration about Ling adder is shown in section 5.

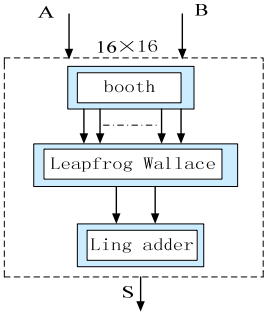

Figure 2. Structure of 16-bit multiplier

3.1 Analysis of booth algorithm

Recently, a recoding scheme introduced by Booth reduces the number of partial products by about a factor of two. Since the amount of hardware and the delay depends on the number of partial products to be added, this may reduce the hardware cost and improve performance. But there is different characteristic in various booth algorithms. In this paper, we analyze various booth algorithms based 16-bit multiplication operation. The analysis result is shown in table2.

Table 2. Analysis of booth algorithms

Algorithm type	Partial products number	Required extra circuit
None	16	None
Booth3	6	Shifting, inverting, 3M.
Booth4	5	Shifting, inverting, 3M, 5M, 6M,7M.
Redundant Booth3	6+1（constant）+2（carry combining）=9	Shifting, inverting, 3M.
Modified Booth2	9	Shifting, inverting.

Based on the analysis above, booth algorithm has an obvious advantage in partial products compression. But it requires extra circuit which includes shifting, inverting and multiple generation circuit. All multiples with the exception of 3M,5M and 7M are easily obtained by simple shifting and complementing of the multiplicand. Make the pre-synthesis based on SMIC 0.18 technology, the synthesis result indicates the delay of shifting and complementing circuit can be ignored. Generation of the multiple 3M,5M and 7M generally require some kind of carry propagate adder to produce. This carry propagate adder may increase the latency and area.

As shown in table 2, booth 4 algorithm produces 4 partial products. But it requires more multiple generation circuit including 3M, 5M, 6M and 7M; Redundant booth 3 algorithm produces 6 partial products, and it requires 1 bias constant and two carry combining numbers. Moreover, it also requires multiple generation circuit; booth 3 algorithm products 6 partial products, and requires 3M generation circuit.

Modified booth 2 algorithm produces 9 partial products. But all of the multiples can be produced using simple shifting and complementing. So the modified booth 2 algorithm shows advantage in speed and area compared to other booth algorithms.

3.2 Analysis of leapfrog Wallace tree

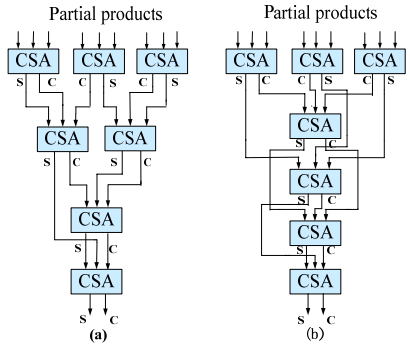

Figure 3. Structure of Wallace tree

As we know, Wallace tree architecture can enhance the compressing speed of partial products. The traditional Wallace tree architecture is shown in Figure 3.(a). In this paper, we have adopted the leapfrog Wallace tree (as shown in Figure 3.(b)) which has a high performance in speed. Furthermore, it reduces the spurious switching in the circuit to save power dissipation.

In the traditional architecture, there are different delay time between Carry (C) signal and Sum (S) signal. For this reason, only when S signal becomes a stable value, can C signal enter next level CSA circuit, which lead to invalid inversion. An improved method is proposed through adopting leapfrog Wallace tree architecture. The main idea of it is to make C and S signals enter next level CSA circuit separately. They can arrive at input port of every level CSA circuit at the same time. So the invalid inversion can be decreased. Compared with traditional Wallace tree architecture, there is not data waiting time in the leapfrog Wallace tree architecture, which accelerate compression of partial products.

4. Analysis and implementation of module modification circuit

In the consideration of wide application of IDEA cipher algorithm, we design the specific module modification circuit to implement modulo $2^{16}+1$ multiplication operation. There are two main realization methods: One is Ma algorithm[7], which can modify directly in the process of multiplication operation. Another method is Low—High algorithm scheme, which can modify the result after multiplication operation. In this paper, we adopt Low—High algorithm to realize module modification circuit. The modified scheme of modulo $2^{16}+1$ multiplication operation is:

a and b are two n-bit data, $c = ab \bmod(2^n+1)$ modulo 2^n+1 multiplication is:

$$c = ab \bmod(2^n+1) = (c_H.2^n + c_L) \bmod(2^n+1)$$

Where c_H is n most significant bits , c_L is n least significant bits.

Combining the above equation, we can obtain:

$$c = (c_H \cdot 2^n + c_L + c_H - c_H) \bmod(2^n+1) = (c_L - c_H) \bmod(2^n+1)$$

If $c_L - c_H \geq 0$, then：$c = c_L - c_H$

If $c_L - c_H \leq 0$, then：$c = c_L - c_H + 1 + 2^n$

According to the Low—High algorithm theory, the conventional module modification circuit is shown in Figure 4.(a). It consists of three adders, which are connected in series. This architecture is easy to implement, but it also leads to a long delay time. The paper presents architecture of module modification circuit, which includes CSA, Ling adder and Multiplexers. Compared with conventional module modification circuit, the delay of two adders is replaced by the delay of two levels of CSA circuit. This improved

structure minimized the delay time obviously.

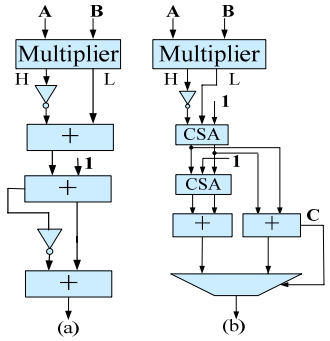

Figure 4. Structure of module modification circuit

5 Analysis of adder unit

In the reconfigurable multiplier 16-bit multiplier, module modification circuit and adder array all include the adder circuit. So the fast adder is important to high performance multiplier design. Compared with conventional adder, we adopt Ling adder which is designed based on CLA adder. Detailed illustration is shown in references[3].In the Ling scheme, the group carry generate and propagate (G and P) are replaced by similar functions (called H and I respectively) which can be produced in fewer stages than the group G and P. we make the comparison between Ling adder and other conventional adders in area and speed performance. Synthesize adders with Synopsys Design Compiler, the result is shown in table 3.

Table 3. Analysis of adder

Adder type	Delay ns	Area μm^2
CPA	1.21	4896
BK	1.27	5121
KS	1.13	6433
CLA	1.19	5255
Synopsys Adder	1.19	3998
LING	1.07	5264

BK=brent-kung adder[8]

CLA=carry increment adder

KS=kogge-stone adder[9]

Based on the analysis above, Ling adder shows an obvious advantage in speed. The area of Ling adder is similar with BK, CPA and Synopsys adder. What is more, the size of Ling adder is smaller than KS and CLA adder. So Ling adder is best choice in the consideration of speed and area.

6. Performance Evaluation of high-speed reconfigurable multiplier

Synthesize the reconfigurable multiplier with Synopsys Design Compiler, based on SMIC 0.18μm CMOS technology, without regard to wire load, at worst case, the comparison in area and speed performance of two kinds of design are list in table 4, where design A is conventional implementation, design B is optimized architecture.

Table 4. comparison in area and speed

Multiplier	Delay(ns)	Area(μm^2)
Design A	3.5	180897
Design B	2.8	182752

7. Conclusion

In this paper, we present a high-speed reconfigurable multiplier, which can achieve both short critical path and low power dissipation. Furthermore, we optimize 16-bit multiplier, module modification block and analyze conventional adder. Synthesize design with Synopsys Design Compiler, Simulation results show achievements on both high speed and small size.

References

[1] A. D. Booth. A Signed Binary Multiplication Technique. Quarterly Journal of Mechanics and Applied Mathematics, 4(2):236–240, June 1951.

[2] Nan Y S ,Chen O T .Low-power multipliers by minimizing switching activities of partial products. IEEE International Symposium on Circuits and Systems[C]. Arizona,USA. IEEE Circuits and Systems Society ,2002.

[3] H. Ling. High-Speed Binary Adder. IBMJournal of Research and Development, 25(2and 3):156–166,May 1981.

[4] L. Dadda. Some Schemes for Parallel Multipliers. Alta Frequenza, 1965.

[5] Jessani R M, Putrino M. Comparison of single and dual-pass multiply-add fused floating-point units [J] .IEEE Transactions on Computers , 1998 , 47(9) : 927?937.

[6] A. Weinberger and J. L. Smith. A One-Microsecond. Adder Using One-Megacycle Circuitry. IRE Transactions on Electronic Computers, EC-5:65–73,1956.

[7] Yutai Ma．A Simplified Architecture for Modulo (2n+1) Multiplication[J] IEEE TRANSACTIONS ON COMPUTERS, 1998.

[8] Brent P R，Kung T H．A regular layout for parallel adders．IEEE Trans Comput，1982.

[9] Kogge P，Stone H．A parallel algorithm for the efficient solu—tion．IEEE TransComput，1973.