一种可重构的快速有限域乘法结构

袁丹寿 戎蒙恬
(上海交通大学电子工程系 上海 200030)

摘 要 在一种改进的串行乘法器的基础上，提出了一种可重构的快速有限域 GF(2^m) ($1<m\le M$)乘法器结构。利用一组配置信号和逻辑电路来改变有限域的度 m，使得乘法器可以重构和编程。同时采用"门控时钟"减小电路功耗。该乘法器结构具有可重构性、高灵活性和低电路复杂性等特点。与传统的移位乘法器相比，它将乘法器速度提高一倍。这种乘法器适合于变有限域，低硬件复杂度的高性能加密算法的 VLSI 设计。

关键词 VLSI，有限域，乘法器， 可重构， 椭圆曲线密码
中图分类号： TN918, TN47 **文献标识码**：A **文章编号**：1009-5896(2006)04-0717-04

Reconfigurable and Fast Finite Field Multiplier Architecture

Yuan Dan-shou Rong Meng-tian
(*Dept of Electronic Engineering, Shanghai Jiaotong Univ., Shanghai* 200030, *China*)

Abstract A reconfigurable and fast architecture over Galois field GF(2^m) ($1<m\le M$) is presented based on the improved serial multiplier. The value m, of the irreducible polynomial degree, can changed by adding a set of configuring signals and logic circuits, which results in that the multiplier architecture is reconfigurable and programmable without changing the hardware. The proposed multiplier architecture has high order of flexibility and low hardware complexity. Compared with the traditional serial multiplier, it can obtain twice speed-up. It suits high-security cryptographic applications with variable finite fields and low complexity requirements.

Key words VLSI, Finite field, Multiplier, Reconfigurable, Elliptic curve cryptosystems

1 引言

有限域在纠错码、高速数字通信、密码学等领域有着广泛的应用。在 RSA 密码和椭圆曲线密码[1-3]等算法设计中，有限域乘法是最频繁、最复杂的运算。因此，优化乘法器结构对提高加密解密运算速度非常重要。

目前已提出的乘法器可以归结为 3 类：比特串行乘法器[4]，具有 $O(m)$ 面积复杂度；并行乘法器[5]，具有 $O(m^2)$ 面积复杂度；混合乘法器[6]，通常用来均衡性能和面积复杂度，例如 LSD (最低数字先)乘法器和 MSD (最高数字先)乘法器。Moon 等人[7]利用算法"展开"技术，提出了一种改进的串行乘法器。它将乘数多项式按照 x 的奇偶次方分成奇偶两部分，然后同时对奇偶部分进行运算，以此将有限域乘法运算速度提高一倍。

以前关于有限域乘法器的研究主要集中在基于固定多项式 $f(x)$ 的乘法器结构设计上。这类乘法器不能满足变有限域的应用。在实际应用中，比如椭圆曲线加密，要求根据需要的安全度选择相应的有限域大小。能满足这种变有限域的有限域乘法有文献[8,9]等。但这些乘法器有的实现面积过大，有的延迟周期过长，它们不适合快速的 VLSI 设计。本文在一种改进串行乘法[7]的基础上，提出了一种可重构的乘法器结构。在此结构中，采用"门控时钟"关闭没有用到的寄存器(触发器)来降低电路功耗。与传统的串行乘法器相比，它具有高度的灵活性，同时能将速度提高一倍。与文献[9]中的乘法器相比，它有较低的硬件复杂度。

基 $\{1,a,a^2,\cdots,a^{m-1}\}$ 被称之为多项式基，其中 a 是不可约多项式 $f(x)=\sum_{i=0}^{m}f_ix^i(f_i\in\{0,1\}$ 的根。使用这种表示法，有限域 GF(2^m) 中的元素可以用阶数小于 m 的多项式表示，其中多项式的系数 $a_i\in\{0,1\}$。例如，元素 A 表示成为 $A=\sum_{i=0}^{m-1}a_ia^i$ ， $a_i\in\{0,1\}$ 。两个元素 $A=\sum_{i=0}^{m-1}a_ia^i$ 和 $\sum_{i=0}^{m-1}b_ia^i$ 的加法定义如式(1)所示，即两个多项式的系数模 2 相加。

$$A+B \bmod F(a)=\sum_{i=0}^{m-1}(a_i+b_i \bmod 2)a^i \tag{1}$$

两个元素的乘法如式(2)所示。

$$\begin{aligned}AB&=\left(A\sum_{i=0}^{m-1}b_ia^i\right)\bmod f(a)\\&=[b_0A+\ldots+b_{m-2}a^{m-2}A+b_{m-1}a^{m-1}A]\bmod f(a)\end{aligned}\tag{2}$$

将式(2)改写成式(3)

$$\begin{aligned}AB&=[Ab_{m-1}a^{m-1}+Ab_{m-2}a^{m-2}+\cdots+Ab_1a+Ab_0]\bmod f(a)\\&=Ab_{m-1}a^{m-1}\bmod f(a)+Ab_{m-2}a^{m-2}\bmod f(a)+\cdots\\&\quad+Ab_1a\bmod f(a)+Ab_0\\&=[\ldots[[Ab_{m-1}]a\bmod f(a)+Ab_{m-2}]a\bmod f(a)+\cdots\\&\quad+Ab_1]a\bmod f(a)+Ab_0\end{aligned}\tag{3}$$

根据处理多项式 B 的系数 b_i 的顺序不同，可以导出两种不同的串行乘法器结构。一种是最低比特先(LSB)的乘法器，从系数 b_0 开始处理。另外一种是最高比特先(MSB)的乘法

2004-06-03 收到, 2004-09-09 改回
国家 863 计划 (2003AA141040)资助课题

器，从系数 b_{m-1} 开始处理。

2 一种改进的串行乘法器

Moon 等人[7]提出了一种有限域乘法器结构。将式(2)分成奇偶两部分如式(4)所示。修改奇数部分，将 x 提到括号外面。

$$\begin{aligned} Z_{\text{even}}(x) &= [b_0A(x)+\cdots+b_{m-3}A(x)x^{m-3}+b_{m-1}A(x)x^{m-1}] \bmod f(x) \\ Z_{\text{odd}}(x) &= [b_1xA(x)+b_3x^3A(x)+\cdots+b_{m-2}x^{m-2}A(x)] \bmod f(x) \\ &= x[b_1A(x)+b_3x^2A(x)+\cdots+b_{m-2}x^{m-3}A(x)] \bmod f(x) \end{aligned} \tag{4}$$

利用式(5)执行模约简运算。

$$\left.\begin{aligned} x^m &= f_0+f_1x+\cdots+f_{m-1} \bmod f(x) \\ x^{m+1} &= f_0x+f_1x^2+\cdots+f_{m-2}x^{m-1}+f_{m-1}x^m \bmod f(x) \end{aligned}\right\} \tag{5}$$

为了简化约简，令不可约多项的 f_{m-1} 等于 0。每次移位需要执行 $x^2A(x)$ 运算，其约简结果如式(6)所示。

$$\begin{aligned} x^2A(x) &= a_0x^2+a_1x^3+\cdots+a_{m-2}x^m+a_{m-1}x^{m+1} \\ &= a_{m-2}f_0+(a_{m-2}f_1+a_{m-1}f_0)a+(a_{m-2}f_2+a_{m-1}f_1+a_0)x^2 \\ &\quad +\cdots+(a_{m-2}f_{m-1}+a_{m-1}f_{m-2}+a_{m-3}) \end{aligned} \tag{6}$$

图 1 和图 2 分别给出了偶数部分和奇数部分的实现。在图中⊗表示“与”门，同时⊕表示“异或”门。在图 2 中，需要处理乘 x^2 运算，也需要乘 x 运算。用 $m+w_t-2$ 个多路选择器(MUX)来选择乘 x^2 运算和乘 x 运算，其中 w_t 是多项式 $f(x)=\sum_{i=0}^{m}f_ix^i$ 汉明重量。在图 1 和图 2 中，分别有两条反馈路径(path1 和 path2)，其中两个反馈位置是相邻的，分别是 $z[m-1]$ 和 $z[m-2]$。

上面乘法器是一种改进的串行乘法器，使用算法“展开”来提高一倍速度，但它所需要的硬件资源比传统的串行乘法器多 50%。

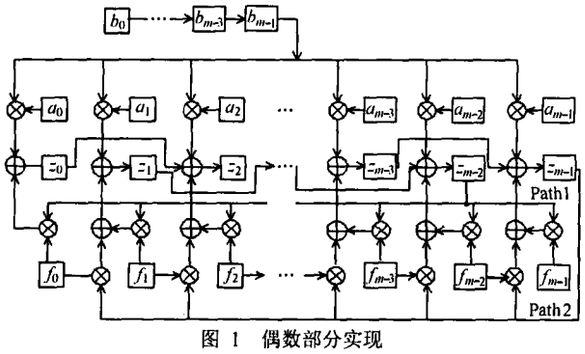

图 1 偶数部分实现

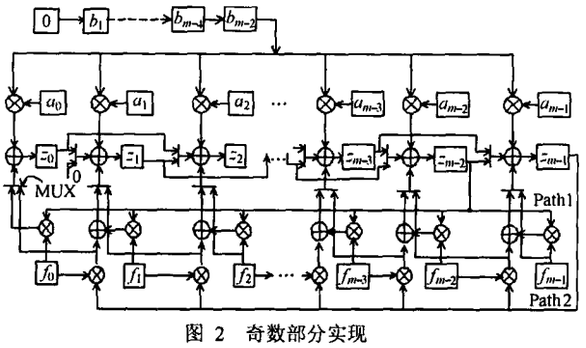

图 2 奇数部分实现

3 一种可重构的乘法器结构

本文在 Moon 等人[7]提出来的乘法器结构的基础上，设计了一种可重构的乘法器结构。使其奇偶部分都可重构。图 3 和图 4 分别给出了可重构的偶数部分和奇数部分。为了实现重构，多项式 $f(x)$ 和有限域都必须是可变的。为了满足这些要求，添加一组配置信号和逻辑电路使不可约多项式 $f(x)$ 和有限域都可变。有限域 GF(2^m) 的度可在[1, M]之间变化，其中 M 是此乘法器支持的最大有限域的度，由需要的安全度确定。这种乘法器结构与上面的乘法器结构相似。奇偶部分各需要 M 个选择逻辑(SL)单元 (如图 5 所示)来确定每个 $z(i)(0\le i\le M-1)$ 寄存器的反馈状态。为了产生 SL 单元，需要添加 M+2 个配置信号 $C[M+1,0]$，表 1 给出了 SL 单元的真值表。如果有限域的度为 m，则反馈位置由 z_{m-1} 和 z_{m-2} 确定，$C[M+1,0]$ 由式(7)确定。

$$C(i)=\begin{cases}1, & 0\le i<m-1\\ 0, & m\le i\le M+1\end{cases} \tag{7}$$

第 m 个 SL 单元确定第 1 个反馈位置，第 $m-1$ 个 SL 单元确定第 2 个反馈位置。反馈路径由线性“或”门组成。

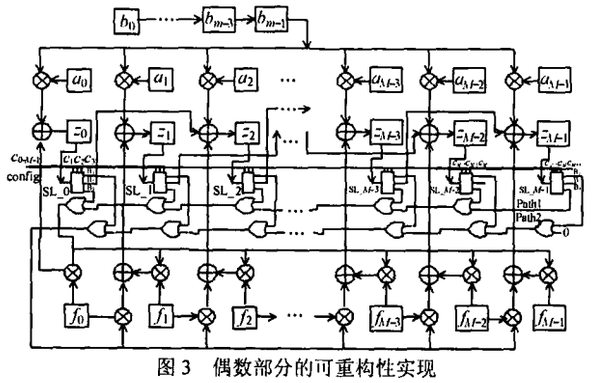

图 3 偶数部分的可重构性实现

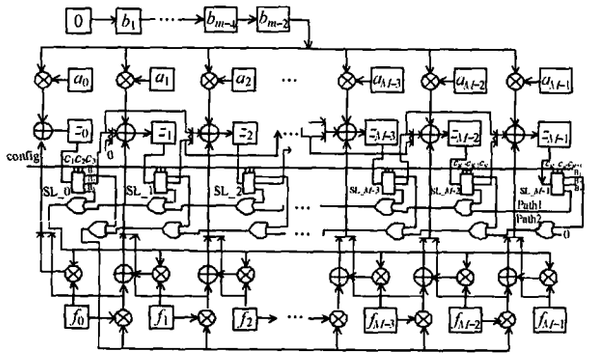

图 4 奇数部分的可重构性实现

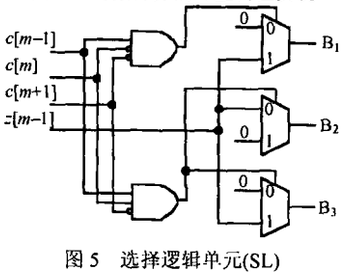

图 5 选择逻辑单元(SL)

当 $i>m-1$ 时，第 i 个 SL 单元的输出(B_1, B_2, B_3)全是 0，因此这些输出不会对反馈造成影响，使得反馈工作正常。

乘法器的关键路径延迟主要由反馈路径(m 个“或”门)和 SL 单元组成。SL 单元的延迟由“非”门的延迟、3 输入-“与”门的延迟和多路选择器的延迟等组成，即 $T_{\mathrm{SEL}}=T_{\mathrm{NOT}}+T_{3\text{-AND}}+T_{\mathrm{Mux}}$。因此乘法器路径延迟为 $mT_{\mathrm{OR}}+T_{\mathrm{XOR}}+T_{\mathrm{Mux}}+T_{\mathrm{SEL}}=mT_{\mathrm{OR}}+T_{\mathrm{XOR}}+2T_{\mathrm{Mux}}+T_{\mathrm{NOT}}+T_{3\text{-AND}}$。这里 T_{OR}，T_{XOR}，T_{Mux}，T_{NOT}，$T_{3\text{-AND}}$ 和 T_{SEL} 分别表示“或”门“异或”门、多路选择器、3 输入-“与”门和选择逻辑(SL)的延迟。

表 1　选择逻辑单元真值表

$c_{[m+1,m-1]}$	B_1	B_2	B_3
001	z_{m-1}	0	0
011	0	0	z_{m-1}
111	0	z_{m-1}	0
其它	0	0	0

由于有限域是可以变的，为了选择图 4 中的 x^2- 和 $x-$ 运算，需要 $2(m-1)$ 个多路选择，而不是 $m+w_t-2$ 个多路选择器。这与文献[7]中乘法器要求不同。

奇偶部分在 $m/2$ 个时钟周期内完成递归运算。其运算结果保存在各自的寄存器 $z[i]$ ($0<i<m-1$)中。最后将 Z_{even} 和 Z_{odd} 模 2 相加(“异或”)得到有限域乘法结果。因此计算有限域乘法需要 $m/2+1$ 时钟周期。

当配置乘法器时，将多项式 $f(x)$ 的系数装载到寄存器 $f(i)$ 中。有限域的度由配置信号控制 $c[i]$ ($0\le i\le M+1$)。

为了降低功耗，采用门控时钟将没有用到的寄存器(触发器)关闭。将时钟与配置信号相“与”，然后连接到相应寄存器的时钟端口上，这样可以实现门控时钟。这是一种减少功耗的常用方法。

4　性能分析与比较

表 2 说明了几种乘法器的比较结果。文献[9]是一种具有可变有限域的乘法器。它的实现需要大量的硬件资源，如 m^2 寄存器，当 m 比较大时，这是一个很大的开销。同时它具有较长的延迟周期($3m$ 个时钟周期)。这种乘法器结构不适合高速的、低面积的 VLSI 设计。文献[8]也是一个可重构的乘法器，它具有 m 个延迟周期。根据图 3 和图 4 的结构，在表中的 Prop(0)栏给出了这种乘法器结构的分析结果。此乘法器比文献[4, 8, 9]中的乘法器都要快，比文献[9]的硬件复杂度低。与文献[7]中的乘法器比较，可以得出可重构性的实现是以牺牲硬件资源为代价的。

在实际的椭圆曲线加密设计中，为了安全性，要求有限域的度m大于163[10]。根据实际应用，令重构的起始地址为 s ($s\le M-1$)。定义可配置的长度为 $\mathrm{Len}=M-s$。这样可以减少s个SL单元，同时可以减少s个“或”门延迟。在Prop(0)中的关键路径延迟中有一项 mT_{OR}，其平均反馈延迟为 $T_{\mathrm{ave}}=([1+\mathrm{Len}]/2)T_{\mathrm{OR}}$。如果把线性“或”门反馈路径改成二进制树”或”门反馈路径，则其延迟为 $[\log_2(\mathrm{Len}+1)]T_{\mathrm{OR}}$。当Len>3时，$[\log_2(\mathrm{Len}+1)]T_{\mathrm{OR}}<T_{\mathrm{ave}}$。图6给出了第1条反馈路径。第2条反馈路径延迟与图6相似。这种方法需要的“或”门近似为 $\dfrac{\mathrm{Len}}{2}+\dfrac{\mathrm{Len}}{2^2}+\cdots+\dfrac{\mathrm{Len}}{2^{\log_2\mathrm{Len}}}=\mathrm{Len}\left(1-\dfrac{1}{2^{\log_2\mathrm{Len}}}\right)$。

表 2　乘法器性能比较

乘法器	文献[4]	文献[7]	文献[8]	文献[9]	Prop (0)	Prop(1)
“与”门	m	$2m$	$2m$	$3m$	$6m$	$6m$
“异或”门	m	$3m$	m	$2m$	$3m$	$3m$
寄存器	$2m$	$3m$	$3m$	m^2	$3m$	$3m$
2:1 选择器	0	$m+w_t-2$	m	$m(m{:}1)$	$8m-2$	$2m$+6Len
延迟周期	m	$m/2$+1	m	$3m$	$m/2$+1	$m/2$+1
1:2 解分器	0	0	m	0	0	0
关键路径延迟	$T_{\mathrm{AND}}+T_{\mathrm{XOR}}$	$T_{\mathrm{AND}}+T_{\mathrm{Mux}}+T_{\mathrm{XOR}}$	*	**	***	****
3 输出-与门	0	0	0	0	$4m$	4Len
“非”门	0	0	0	0	$6m$	6Len
“或”门	0	0	m	0	$4m-2$	$4\mathrm{Len}\left(1-1/2^{\log_2\mathrm{Len}}\right)$
可重构性	No	No	Yes	Yes	Yes	Yes

* $2T_{\mathrm{AND}}+T_{\mathrm{XOR}}+T_{\mathrm{NOT}}+(m+1)T_{\mathrm{OR}}$.

** $3m(T_{\mathrm{XOR}}+[\log_2 m](T_{\mathrm{NOT}}+T_{\mathrm{AND}}+T_{\mathrm{OR}}))$.

*** $mT_{\mathrm{OR}}+T_{\mathrm{XOR}}+T_{\mathrm{Mux}}+T_{\mathrm{SEL}}=mT_{\mathrm{OR}}+T_{\mathrm{XOR}}+2T_{\mathrm{Mux}}+T_{\mathrm{NOT}}+T_{3\text{-AND}}$.
$T_{\mathrm{SEL}}=T_{\mathrm{NOT}}+T_{3\text{-AND}}+T_{\mathrm{Mux}}$

**** $[\log_2(\mathrm{Len}+1)]T_{\mathrm{OR}}+T_{\mathrm{XOR}}+T_{\mathrm{Mux}}+T_{\mathrm{SEL}}=[\log_2(\mathrm{Len}+1)]+T_{\mathrm{XOR}}+2T_{\mathrm{Mux}}+T_{\mathrm{NOT}}+T_{3\text{-AND}}$

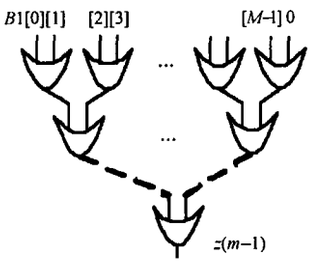

图 6 反馈通道的“或”门二进制树结构

这种方法能够减少反馈路径的平均延迟，同时也可以减少“或”门。选择配置的起始地址为s，同时采用 “或”门二进制树确定反馈路径。表2中的Prop(1)栏给出了这种改进后的分析结果。

令最大多项式的度为$M=193$。对于Prop(1) 栏对应的乘法器结构。功能和时序仿真验证了电路的正确性。用0.25 μm CMOS工艺综合了这种乘法器结构。延迟由数据到达时间(the data arrival time)确定。图7给出了电路关键路径延迟(delay)与配置的起始位置(s)的关系。图8给出了电路面积(Area)与配置的起始位置(s)的关系，没有考虑连线面积。两张图说明了电路延迟与电路面积随着s的增加而减小。此综合结果与上面的理论分析相吻合。

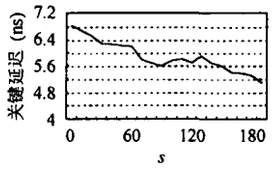

图 7 关键路径延迟与配置起始位置s的关系

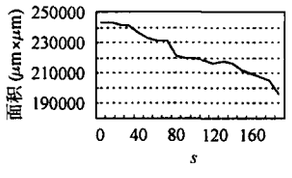

图 8 面积与配置起始位置s的关系

5 结束语

本文针对加密算法的有限域乘法运算提出了一种可重构的快速的乘法器结构。这种乘法器结构支持任意的度小于M的多项式。它具有可重构性、高灵活性和低硬件复杂度等特点。功能仿真和DC综合验证了此乘法器结构的正确性。它适合变有限域、低硬件复杂度和高速等要求的智能卡的VLSI设计。

参 考 文 献

[1] Menezes A J, Oorscho P C V, Vanstone S A. Handbook of Applied Cryptography, Boca Raton, FL, CRC Press,1997.

[2] Orlando G. Efficient elliptic curve processor architectures for field programmable logic [PhD thesis]. Dept.of Electrical Eng., Worcester Polytechnic Institute, America, 2002.

[3] Bednara M, Daldrup M, Gathen J V Z. Reconfigurable implementation of elliptic curve crypto algorithms. parallel and distributed processing symposium. Proceedings International, IPDPS, Fort Lauderdale, Florida 2002: 157 – 164.

[4] Mastrovito E. VLSI architectures for computation in Galois fields [PhD thesis]. Dept.of Electrical Eng., Linkoping Univ., Sweden, 1991.

[5] Lee C Y, Lu E H, Sun L F. Low-complexity bit-parallel systolic architecture for computing AB^2+C in a class of finite field GF(2^m). *IEEE Trans. on Circuits and Systems* II, 2001, 48(5): 519 – 523.

[6] Paar C, Fleischmann P, Rordriguez P S. Fast arithmetic for public-key algorithms in Galois fields GF(2^m) with composite exponents. *IEEE Trans. on Computers*, 1999, 38(7): 796 – 800.

[7] Moon S, Park J, Lee Y. Fast VLSI arithmetic algorithms for high-security cryptographic application. *IEEE Trans. on Consumer Electronics*, 2001 47(3): 700 – 708.

[8] Kitsos P, Theodoridis G, Koufopavlou O. An efficient reconfigurable multiplier architecture for GF(2^m). *Microelectronic Journal*, 2003, 34(10): 975 – 980.

[9] Hasan M A, Ebtedaei M. Efficient architectures for computations over variable dimensional Galois fields. *IEEE Trans. on Circuits and Systems* I, 1998, 45(11): 1205 – 1211.

[10] Beth T, Gollman D. Algorithm engineering for public key algorithms. *IEEE J. on Selected Areas in Communications*, 1989, 7 (4): 466 – 485.

袁丹寿： 男，1977年生，博士生，研究方向为信息安全 VLSI 设计、图像传输.

戎蒙恬： 男，1952年生，教授、博士生导师，研究方向为数字与模拟集成电路芯片设计、集成电路设计自动化、通信系统性能分析与优化.