文章编号:1671-4598(2018)02-0252-03 DOI:10.16526/j.cnki.11—4762/tp.2018.02.062 中图分类号:TH873.7 文献标识码:A

# 基于 FPGA 的 RC4 加密算法设计及实现

苗三立[1]，左金印[2]，宋宇飞[1]
(1. 华北计算机系统工程研究所，北京 100083；2. 北京大学，北京 100083)

**摘要**：针对通信安全问题，采用自顶向下的设计方法，设计了一种 RC4 算法基于 FPGA 的实现方式，实现了通信数据的加密传输；根据 RC4 加密算法的原理和设计流程，使用 Verilog HDL 编程语言，采用有限状态机（FSM）的编程方式实现算法，通过 Modelsim SE 10.1a 仿真软件进行仿真，并在 FPGA 开发板上进行验证；采用文章提出的 FPGA 设计方法实现的 RC4 加密算法相比软件加密方式和已有的 FPGA 实现方式速度有明显提高。

**关键词**：数据传输；Verilog HDL；FPGA；RC4；有限状态机

# Design and Achieve of FPGA—based RC4 Encryption Algorithm

Miao Sanli[1], Zuo Jinyin[2], Song Yufei[1]
(1. North China Computer System Engineering Research Institute, Beijing 100083, China;
2. Peking University, Beijing 100083, China)

**Abstract**: According to the communication security problem, proposes a top—down design method of Top—Down FPGA RC4, hardware encryption method based on the realization of data encryption communication, this paper introduces the principle and design process of the RC4 encryption algorithm, using Verilog HDL programming language, using finite state machine (FSM) programming algorithm is simulated by Modelsim SE 10.1a the simulation software, and verified in the FPGA version of the development. The encryption algorithm FPGA encryption software design compared to the rate increased significantly, compared to other hardware encryption clock was significantly reduced.

**Keywords**: data transmission; Verilog HDL; FPGA; RC4; FSM

## 0 引言

随着计算机网络和通信技术的发展，信息作为一种重要的资源已经成为促进社会进步和经济增长的重要力量。但是，信息的存储、传递、处理等过程通常需要通过公开的计算机网络进行操作，容易遭受到窃听、篡改、伪造等多种攻击方式的威胁，因此信息安全是计算机网络领域中的一个重要的课题方向[1]。众多加密算法也应运而生，典型的有 DES，RC4，RSA，AES，IDEA 等，其中加密算法 RC4 以其实现容易、加密速度快、安全性较高，得到广泛的认可与应用。该算法的速度可以达到 DES 加密算法的 10 倍左右，且具有较高级别的非线性[2]。随着计算机网络的普及，传统的软件加密已经逐渐不能满足日常工作的需求，传统软件加密的局限性也逐渐暴露出来，在众多计算机信息安全系统中，各种高速数据密码机、密码卡、智能卡等硬件加密手段被应用到各种系统设备中，用来提高密码运算速度和系统的安全性、可靠性。当前，以密码设备为核心的硬件加密系统已成为构筑信息安全平台的重要屏障。本文给出了一种基于 FPGA 的 RC4 加密算法实现方案，相比软件实现，该方案加密速度更快，安全性更高，可以应用于多种不同的信息传输通信系统[3]。

## 1 RC4 算法介绍

RC4 是一种流密码算法，是现代通信领域通用的基于非线性变换的算法，该算法被广泛用于 SSL/TLS 协议、WEP 协议和 WPA 协议，Lotus Notes、Windows、Oracle Secure SQL、Apple（AOCE）等多种通讯协议，现如今 RC4 算法也被用作蜂窝数字数据包规范[4]。RC4 加密算法是由 Ron Rivest，一名美国的密码学家于 1987 年设计完成实现，起初 RC4 算法被用于数据库（MySQL）安全以及网络安全等领域上，一直到 1994 年该算法被发布到了互联网上，开源公布给大家[5]。对 RC4 算法的改进分别是 Paul 和 Prenecl 于 2005 年提出的 RCA 算法和 Bartosz 于同年提出的 VMPC 算法[6]。

该算法是一种流密码加密算法，是基于非线性变换的算法，主要包括两个方面的内容：

1）密钥编制算法（Key Schedule Algorithm）：用变长的密钥进行随机排序产生密钥流的初始状态；

2）伪随机序列生成算法 PRGA（pseudo random generation algorithm）：参照初始状态生成相应的密钥流序列，然后与明文相异或生成相应的密文。

RC4 加密算法加密原理如下：第一步，按照既定的算法初始化两个向量寄存器 S 和 T；第二步，对寄存器 S 进行重排列（乱序排列）；第三步，按照既定的算法产生相应的密钥流 k，然后存储并参与之后的明文加密操作[7]。加密流程如图 1 所示。

## 2 硬件设计

本文采用 Verilog HDL 语言以同步有限状态机（FSM）的设计方法，实现了基于 FPGA 的 RC4 加密处理流程。在计算机信息安全系统中，硬件加密手段被应用到设备中来提高密码

**收稿日期**:2017-08-17； **修回日期**:2017-09-15。

**作者简介**:苗三立(1991-),男,山东潍坊人,硕士,主要从事遥测遥控系统等方向的研究。

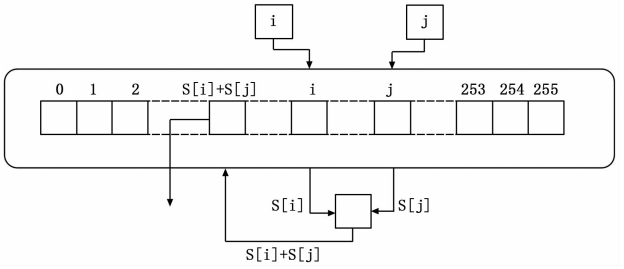


图 1　加密流程

运算速度和系统的安全性。RC4 加密算法的 FPGA 实现方案，相比软件实现方案，硬件加密方案速度更快，安全性更高[8]。

系统设计由 3 部分组成，分别是时序逻辑控制模块、运算控制模块及存储模块三部分组成，其中存储模块包括密钥流存储模块和内部寄存器存储模块。明文数据由外部输入，暂存于明文寄存器，与存储模块存储的密钥流共同参与运算模块的处理。运算模块产生的密文暂存于密文寄存器中，供后续模块的使用。其中，系统内使用的时钟由时序处理模块产生，通过 FPGA 配置 AD9516 芯片输出低抖动、稳定的时钟[9]。首先将密钥信息 K 存储在密钥存储模块（RAM）中，然后根据地址信息从密钥存储模块（RAM）中提取密钥信息 K，本文中规定明文数据为 3 字节（24 bit），经过 RC4 加密处理模块产生相应的密钥流 k 信息，完成密钥流的生成，用以生成相应的密文信息[10]。逻辑框图如图 2 所示。

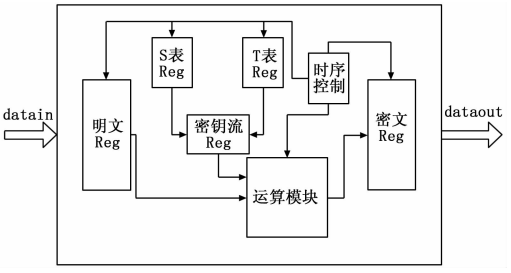


图 2　逻辑框图

1）时序逻辑控制模块：时序逻辑控制模块采用 AD9516－3 芯片，AD9516－3 是 ADI 公司发布的 AD9516 系列时钟产品，可以提供高质量、低抖动的时钟信号，适用于无线和有线的基础设施、WIMAX 基地、光网站、自助测试设备、医学图像处理等仪器仪表装置。AD9516 系列集成了 1 个整数 N 分频的频率合成器、1 个轧空振荡器（VCO）、2 个参考输入端、可编程驱动器、可调延迟线和 14 个时钟驱动器，包括 LVPECL、LVDS 和 CMOS 输出。AD9516 芯片由系统 FPGA 芯片通过串行控制端口来进行相关的参数配置及输入[11]。模块结构图如图 3 所示。

2）运算控制模块：采用有限状态机的方式对各个逻辑状态进行控制和操作，时钟采用 AD9516 芯片输出的低抖动、高质量的时钟信号。详细的状态流程才加软件设计部分。

3）存储模块：存储模块采用 Block RAM 存储，相较与 FIFO 存储而言，优势在于可以通过访问地址选取不同的数据信息。为解决并行系统高效进行大量数据传输和交换的实时性要求，本文采用了一种基于 FPGA 的四口 RAM，四口 RAM 由 1 个双口 RAM 模块、2 个控制模块和 4 个缓存模块组成[12]。总体设计架构如图 4 所示。

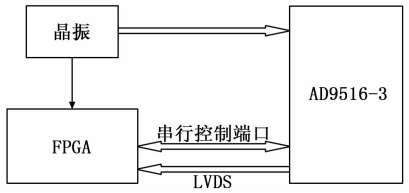


图 3　时序逻辑控制模块结构图

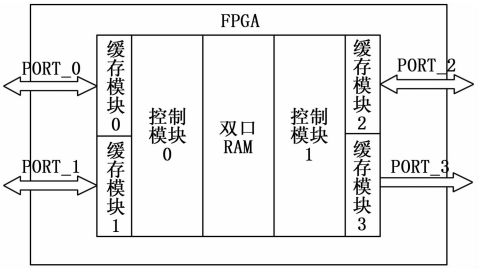


图 4　四口 RAM 存储模块中体设计架构图

## 3　软件设计

在本设计系统中，流程设计采用状态机的规范进行编写，总体流程为：消息信息进入设计系统，首先进行 CRC 编码，经过编码后的消息信息进行 RC4 编码加密处理，然后进行消息信息的传递。

### 3.1　CRC 校验设计

循环冗余校验码（cyclic redundancy check，CRC）又称 CRC 校验，是一种能力非常强的检错、纠错码，相比于更加稳定、精确的 RS 校验算法，具有实现编码和检码的电路比较简单、编码易于实现等优势，通常用于串行传输的辅助存储器与主机的信息数据通信和计算机网络中。CRC 检验是一种在数据通信系统和其它串行传输系统中广泛使用的错误检测手段。通用的 CRC 标准有 CRC－8、CRC－16、CRC－32、CRC－CCIT，其中在设备信息传输通信系统中应用最广泛的是 CRC－16 标准。传统手段是采用串行的编码方式进行 CRC 编码，速度比较慢，处理并行数据电路较为复杂；本系统将采用一种新的 CRC－16 并行编码方式，采用并行处理的方式进行对并行信息数据进行 CRC 编码。相比于串行 CRC 编码方式，消耗时钟减少，硬件电路简化等优势。

进行 CRC 编码要采用模 2 运算法则，具体描述一下模 2 运算法则。首先模 2 运算是一种二进制运算法则，同时也包括相应的模 2 加、模 2 减、模 2 乘、模 2 除 4 种运算。模 2 运算用“＋”表示加法运算，用“－”、“×”或“.”、“/”分别表示减法、乘法和除法运算。因此，模 2 运算是类似于四则运算的一种二进制算法。但是模 2 运算不同于四则运算的是，模 2 减法为不带借位的二进制减法，模 2 加法是不带进位的二进制加法。同理，模 2 乘法在中间的累加运算时也是不带进位的二进制加法；模 2 除法在中间结果的减法时也是不带借位的二进制减法。由此看来，模 2 运算的加减法相当于两个二进制数的按位异或运算操作。同理，模 2 除法运算的，如果余数的首位为 1，那么商就为 1，其他操作按照模 2 减法运算；如果余数的首位为 0，那么商就为 0。

CRC 码校验的基本思路是：利用线性反馈移位寄存器（linear feedback shift register，LFSR）和线性编码的原理。LFSR 是指，首先给定前一状态的输出，将该输出的线性函数再用作输入的移位寄存器。其中文章中采用的异或运算是最常见的单比特线性函数：对寄存器中的某些位进行异或运算后作为后面状态的输入端，然后再对整个寄存器进行移位操作。具体运算如下：再输出端按照要发送的 $k$ 位二进制数据序列，采用模 2 运算的法则产生一个 $r$ 位的监督码，即校验用的 CRC 码，然后将其添加到 $k$ 位二进制信息码的低位处，组成一个 $k+r$ 位的新的二进制数据序列，此产生二进制数据序列的过程便是 CRC 编码。在检验时，使用 CRC 编码与生成多项式 $G(x)$ 进行模 2 除法运算，运算结果如果余数为 0，那么说明在传输过程中 $k$ 位二进制数据序列没有发生错误或丢失；如果运算结果不为 0，说明在传输过程中 $k$ 位二进制数据序列发生错误或产生丢失。因此，采用余数是否为 0 来判断数据在传输过程中是否发生变化。

CRC 码的检错纠错原理：在接收端收到数据后若如果有一位数据出现错误，结果则为余数不为 0，并且不同位出错，其余数不但不为 0 并且也不同。假设循环码有一位出错，用 $G(x)$ 作模 2 除将得到一个不为 0 的余数。如果对余数补 0 继续除下去，将出现一个有趣的现象：余数会按照 001，010，100，011，111，101 循环下去，分别对应出错位第 1，2，3，4，5，6，7 位。例如第一位出错，余数将为 001，余数后面补 0 后再除（补 0 后如果数据的最高位为 1，则用除数做模 2 减法然后取余数；如果最高位为 0，则其最低 3 位就是余数），计算后得到第二次余数为 010。之后继续补 0 作模 2 除，依次得到余数为 100，011，111，101，循环下去。

本系统采用的是并行处理的 CRC 校验，相比串行 CRC 校验消耗的时钟减少，电路结构简化。输入数据为十六进制数据 A7，串行处理结果图 5 所示，经过 8 个时钟周期后结果为十六进制数据 83D1。并行处理结果如图 6 所示，系统经过 1 个时钟周期后输出结果为十六进制数据 83D1。

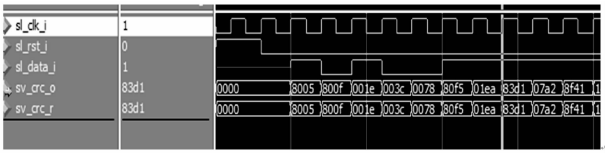


图 5 串行 CRC 编码时序仿真图


图 6 并行 CRC 编码时序仿真图

### 3.2 算法设计流程

在本系统设计中，FPGA 芯片采用 Xilinx 厂家生产的 Spartan6 XC6SLX45T 型号，RC4 算法程序采用 Verilog 编写。RC4 算法程序流程图如图 7 所示。

RC4 算法具体流程为，首先从 RAM 中提取密钥 k，然后初始化数组 S、T，系统内设定数组容量为 256 位；然后对数组 S 进行重排序，进而产生 32 位的密钥，存入 RAM 模块，与明文进行异或操作产生密文。

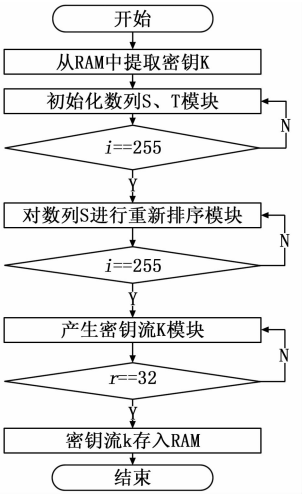


图 7 RC4 算法程序流程图

在设计中采用同步有限状态机（FSM）实现，其状态转移图如图 4 所示。idle 为初始化数组 S 和 T 状态，arrange 为重新排列数组 S 状态，produce 为产生密钥流状态；remainder 为对 j 取余状态，swap _ 1 为交换数组 S 状态；remainder _ i 为对 i 取余状态，remainder _ j 为对 j 取余状态，remainder _ t 为对 t 取余状态，swap _ 2 为交换数组 S 状态，myflow 为产生密钥流状态。

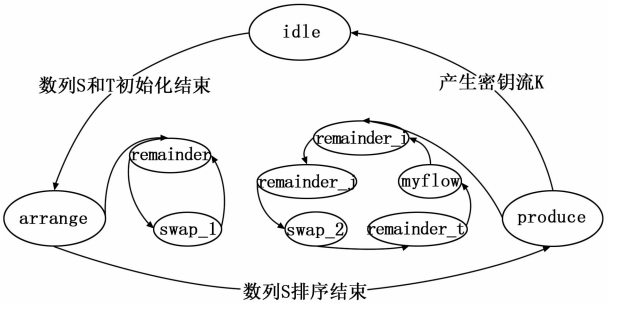


图 8 RC4 算法状态转移图

当明文数据有效时，程序 RC4 算法加密模块控制信号有效，程序采用异步复位信号控制，当复位信号为无效信号（即低电平）时，在 FPGA _ CLK 时钟上升沿的控制下，进入 idle 状态，定义两个位宽为 8 位，深度为 256 的一维数组 S 和 T，在该状态下进行对一维数组 S 和 T 的初始化。对一维数组 S 按照顺序进行初始化赋值，对一维数组 T 按照给定的密钥 K 来进行初始化赋值，此处需要 256 个时钟周期。当下一个 FPGA _ CLK 时钟上升沿到来时，state 进入 arrange 状态，进行对一维数组 S 进行重排序，如图 4 所示。注意进入该状态时，要对寄存器 j 进行清零操作。state 信号进入同步有限状态机（FSM）state1，首先进入 remainder 状态，对寄存器 j 进行取余操作；然后随着 FPGA _ CLK 上升沿的到来，进入 swap _ 1 状态，对一维数组 S 进行“任意”的交换处理操作，保证每一个一维数组 S 都进行交换处理操作，在该状态下需要 256 次循

（下转第 263 页）