文章编号:1674-2974(2022)02-0106-05 DOI:10.16339/j.cnki.hdxbzkb.2022265

一种优化的MD5算法与硬件实现

王镇道 ,李妮 †
(湖南大学 物理与微电子科学学院 湖南 长沙 410082)

摘 要:MD5算法是应用非常广泛的一种Hash算法,在数字签名和验签中占有重要地位,算法的效率会直接影响到签名和验签的速度.本文提出一种优化的MD5算法,采用三级加法器替代四级加法器、优化循环移位操作的方式缩短MD5算法单步运算的关键路径,并用VERILOG HDL语言进行硬件实现.通过仿真和FPGA验证,结果表明该设计功能正确,硬件资源消耗少,数据吞吐量大.该设计应用于一款密码安全芯片,采用0.18 μm工艺进行MPW流片,芯片面积为6 mm^2.时钟频率为150 MHz,电压3.3V时,功耗约为10.7 mW.

关键词:MD5算法;hash算法;签名和验签;散列函数
中图分类号:TN335 **文献标志码**:A

An Optimized MD5 Algorithm and Hardware Implementation

WANG Zhendao, LI Ni†
(School of Physics and Electronics, Hunan University, Changsha 410082, China)

Abstract: The MD5 algorithm is a widely used Hash algorithm, which occupies an important position in digital signatures and signature verification. The efficiency of the algorithm will directly affect the speed of signature and signature verification. This paper proposes an optimized MD5 algorithm, which uses a three-stage adder to replace a four-stage adder, optimizes the cyclic shift operation to shorten the critical path of the single-step operation of the MD5 algorithm, and implements the hardware in VERILOG HDL language. Through simulation and FPGA verification, the results show that the design function is correct and consumes fewer hardware resources and has a large data throughput. The design is applied to a cryptographic security chip, which uses a 0.18 μm process for MPW tape-out with a chip area of 6 mm^2. When the clock frequency is 150 MHz and the voltage is 3.3 V, the power consumption is about 10.7mW.

Key words: MD5 algorithm; Hash algorithm; signature and signature verification; Hash functions

互联网技术的高速发展给人们带来了许多便利的同时,也带来了诸多的问题.人们的生产和社会活动与网络密切相关,随时随地都在利用网络进行数据交互.提供一个高效性、隐私性和完整性的信息生

* **收稿日期**:2021-03-22
基金项目:湖南省战略性新兴产业科技攻关与重大科技成果转化项目(2017GK4008),Transformation Project of Major Scientific and Technological Achievements in Strategic Emerging Industries in Hunan Province
作者简介:王镇道(1974—),男,湖南沅江人,湖南大学副教授
† 通信联系人,E-mail: 15200343925@163.com

存环境是时代的需求,是迫切需要解决的问题.

在信息安全的研究领域中,密码学以及相关技术发挥着越来越关键的作用.加密哈希函数在密码学中扮演着至关重要的角色.它广泛应用于电子商务、信息安全和电子政务等安全性要求比较高的领域中[1],同时也是实现数字签名、消息的完备性和消息可认证性的重要工具.

MD5算法是MD结构的典型代表,是密码学中应用广泛的一种哈希函数[2].由Ronald Rivest在1991年提出[3].MD5算法可以将任意长度的数据输入压缩成128 bit的输出,具有不可逆性、数据完整性、不可抵赖性等特点[4],可以防止数据被篡改和数据丢失.

MD5算法可通过软件和硬件的方式实现.软件实现的算法极其依赖计算机硬件平台,算法运算时间长,效率低,不能满足物联网高速发展的需求.传统的MD5算法一般采用软件或计算机硬件平台的方式实现,算法效率难以满足物联网高速发展的需求.本文提出一种优化的MD5算法,在循环迭代模式上利用三级加法器替代四级加法器、优化循环移位操作方式缩短关键路径,在流水线模式下采用32级流水线设计去搭建算法实现架构.最后通过VERILOG HDL语言进行硬件实现.

1　算法介绍

MD5算法以任意位的消息作为输入,将消息经过系列处理后以512位分组形式来处理输入消息,且每一分组会被分割成16个32位子分组,经过一系列的计算处理得到新的四个32位分组,将这四个32位分组级联后生成一个128位散列值就是所求的MD5算法加密的摘要值[5].

1.1　消息扩展

首先对输入消息进行填充,使其消息长度对512求余数的值为448.故消息长度扩展至N*512+448 bits(N为正整数)[6-7].

填充的方法如下:在消息后面填充一个1和无数个0,直至满足对512求余得448才停止对信息进行填充.然后再在其后加上一个以64位二进制表示的填充前的消息长度.经过这两步的处理,填充后的消息长度为N*512+448+64=(N+1)*512 bits,长度恰好是512的整数倍,以便后续分组[8-9].

1.2　消息分组

将每512-bit的消息划分成16组,每组32-bit,同时给出MD5中四个32位作为链接变量(Chaining Variable)的整数参数,分别为:A=0x01234567,B=0x89$abcdef$,C=0$xfedcba$98,D=0x76543210[10].本文中填充好的消息数据以及链接变量均采用小端字节序的方式进行计算处理.

1.3　循环运算

首先定义好每轮运算的逻辑函数,即:

FF(a,b,c,d,M_j,s,t_i)为

$$a=b+((a+F(b,c,d)+M_j+t_i)<<<s) \tag{1}$$

GG(a,b,c,d,M_j,s,ti)为

$$a=b+((a+G(b,c,d)+M_j+t_i)<<<s) \tag{2}$$

HH(a,b,c,d,M_j,s,t_i)为

$$a=b+((a+\mathrm{H}(b,c,d)+M_j+t_i)<<<s) \tag{3}$$

H(a,b,c,d,M_j,s,t_i)为

$$a=b+((a+I(b,c,d)+M_j+t_i)<<<s) \tag{4}$$

其中包含了四个非线性基本函数为:

$$\mathrm{F}(x,y,z)=(xy)|(\sim x\&z) \tag{5}$$

$$\mathrm{G}(x,y,z)=(xy)|(\sim x\&z) \tag{6}$$

$$\mathrm{H}(x,y,z)=(x\hat{}y\hat{}z) \tag{7}$$

$$\mathrm{I}(x,y,z)=y\hat{}(x|\sim z) \tag{8}$$

t_i表示4294967296*abs(sin(i))的整数部分[11-12];M_j表示512-bit的消息的第j个子分组;s表示循环左移s位.

当MD5算法进行运算时,先将链接变量A,B,C,D寄存到a,b,c,d中,再进入四轮循环,每轮操作非常相似.其中每轮操作就是计算对应轮的逻辑函数的值且每轮的循环次数为16次.

1.4　摘要输出

四轮计算完后,将A,B,C,D加上对应的a,b,c,d得到新的A,B,C,D.如果还有512-bit分组,则可作为下一分组数据运算的链接变量,最终输出的A,B,C和D,A是低位,D为高位,$DCBA$组成128位输出结果,即MD5算法计算得出的消息的摘要值.

2　优化MD5算法设计

硬件实现MD5算法有两种模式,其一为循环迭代模式,其二为流水线设计[13].首先在循环迭代模式上对64步循环运算的单步运算关键路径进行优化,利用三级加法器替代四级加法器、优化循环移位操作的方式缩短MD5算法单步运算的关键路径.其次采用流水线设计实现64步循环运算并行化,一个时钟周期内实现2步运算,实现缩短单步运算关键路径,提高时钟频率,而且32个通道可同时进行MD5

运算,提高数据处理速度,增大数据吞吐量.具体优化设计如下.

2.1 缩短关键路径

在循环迭代方面将逻辑函数FF、GG、HH、II的数据计算流程优化设计如图1所示.

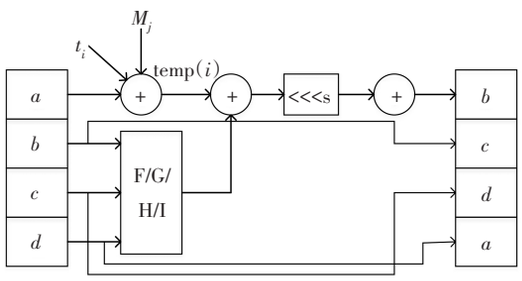

图1 MD5算法的逻辑函数计算流程图

Fig.1 MD5 algorithm logic function calculation flow chart

1)利用一个加法器替代实现两个加法器的操作,四级加法器操作优化成三级加法器.

以单步函数FF计算为例:算法的第1、2步在创建激励时将任意位宽的消息拓展成所需的数个512-bit数据.每组512-bit输入数据划分为16个子分组数据,M_j视为常数,t_j表示4294967296*abs(sin(i))的整数部分视为已知常数[14-15],故可先预处理两组中间值

$$\mathrm{temp}(i) = c(i)+M(i+2)+t(i+2)$$

$$\mathrm{temp}(i+1) = b(i)+M(i+2)+t(i+2)$$

利用流水线设计实现一个时钟周期内输出逻辑函数运算的结果,并对应地输出给紧接的两组FF数据.其余轮的运算与此类似,得到的a,b,c,d的值向右轮换输出给下一轮计算使用.

2)优化循环移位操作.

由于64步循环步骤中分成F、G、H、I四轮操作,且每轮中循环移位的位数s按固定的4个数字进行循环移位,一共循环4次,各占16步操作.其中F轮操作中循环移位的位数s为7、12、17、22;G轮操作中循环移位的位数s为5、9、14、10;H轮操作中循环移位的位数s为4、11、16、23;I轮操作中循环移位的位数s为6、10、15、21;由于位数固定,取消32-bit数据进行循环移位的操作,替代的是直接将移位的结果作为加法器的加数相加.

利用以上两种方式缩短单步运算的关键路径,有效提高MD5算法运算速度.

2.2 流水线设计

MD5算法的运算过程是串行执行,每一个逻辑函数计算的结果都要作为下一步逻辑函数计算的初始值,寄存器大部分时间都处于等待状态,故运算速度慢,数据吞吐量小.文献[5]以并行计算为基础,优化MD5算法实现的硬件架构,采用三级、四级流水线进行模块设计.文献[7]首先分析了1、4、32级多种流水线设计的吞吐性能,最终利用32级流水线设计,尽管实现了数据吞吐量达到32 Gbps,但整个设计只对流水线结构进行扩展,电路结构非常复杂.文献[6]对数据流进行优化,提出多种方案实现MD5算法,最终实验结果表明:最佳方案数据吞吐量达到了66.56 Gbps;但资源占用率较高,实现MD5算法对硬件要求高,不易实现.本文在算法实现结构上进行优化,构建新的32级流水线设计,极大提升了运算速度,并且面积相对较小.

在设计过程中将两步运算作为流水线的一级,构建一个32级的流水线,来完成MD5的64步运算.在占用相对较少资源的基础上提高算法的运算速度,35个时钟周期可以完成一次消息摘要值的计算.

MD5算法的架构设计如图2所示.顶层模块名为MD5_Accelerate,功能是可同时计算32个通道消息分组(512-bit)的MD5摘要值.由于输入消息数据长度不一,算法第1、2步填充数据创建激励在验证平台实现,输入数据是按照512-bit位宽进行输入.同一个消息可能有多个消息分组(512-bit)组成,对属于同一个消息的不同消息分组,利用输入该消息分组(512-bit)的通道编号(tid),以及开始(隐含)和结束(tlast)的标志来进行判别.整个模块包括2个子模块md5_pipe_ctrl和md5_pipe_core,他们的主从关系如图3所示.

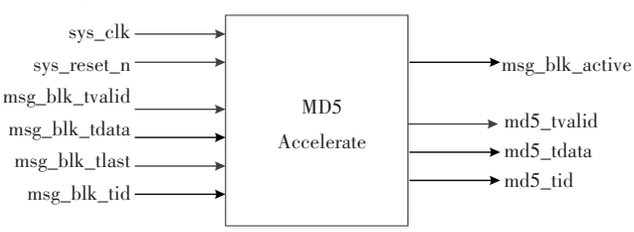

图2 顶层模块示意图

Fig.2 Top-level module diagram

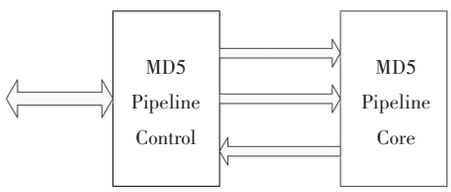

图3 MD5算法硬件结构图

Fig.3 MD5 algorithm hardware structure diagram

2.2.1　MD5 Pipeline Core子模块

子模块MD5 Pipeline Core实现消息分组(512-bit)的Hash运算,共包括4轮,每轮包括16步运算,共64步.为了兼顾吞吐量与延时,将2步作为流水线的一级,构建一个32级流水线,完成MD5的64步运算.32级流水线可以同时处理32个消息分组,对于属于同一个消息的不同消息分组,比如M0和M1,由于M1的Hash运算依赖于M0的Hash结果,所以32个流水级中不允许出现同一个消息的不同消息分组.

2.2.2　MD5 Pipeline Control子模块

子模块MD5_Pipeline_Control接收至多32个通道的消息分组(512-bit),发送到MD5 Pipeline Core子模块进行Hash运算,同时接收MD5 Pipeline Core子模块的Hash结果进行输出.该模块提供MD5 Pipeline Core处理的消息分组(512-bit)的工作状态msg_blk_active[31:0],msg_blk_active[i]表示通道i消息分组(512-bit)的工作状态,'1'表示被处理中,'0'表示空闲状态.由于MD5算法的运算依赖关系,消息分组(512-bit)的输入需要依据msg_blk_active[31:0],即如果msg_blk_active[i]等于'0',那么可以输入通道i的消息分组(512-bit).对于同一个消息有多个消息分组(512-bit)组成的情况,如果输入数据的tid相同、且上一个tid对应的tlast不为0,即上一个输入的数据不是对应tid的最后一个消息分组,此时输入数据依然是同一消息的一个分组.如果该消息分组(512-bit)的结束标志(tlast)为'1',那么经过35个流水线延时后,输出该消息的MD5摘要值.具体时序如图4所示.

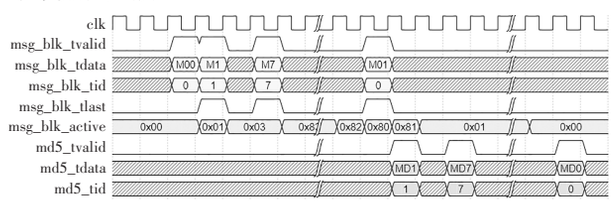

图4　多个消息分组时序图

Fig.4　Multiple message grouping sequence diagram

图4表示3个消息分别输入通道0,1和7,第一个消息包含2个消息分组:M00和M01,直到msg_blk_active[0]等于0,通道0才能输入下一个消息分组M01:msg_blk_tvalid有效并且msg_blk_tlast等于1,表示该消息分组是最后一个分组.从输入消息分组M01到输出摘要MD0的时间间隔是35个时钟周期.

3　实验结果与分析

通过ModelSim对该电路进行功能仿真,如图5所示.MD5运算完后拉高md5_tvalid信号,对应的md5_tdata即为Hash值,仿真结果表明该设计功能正确.

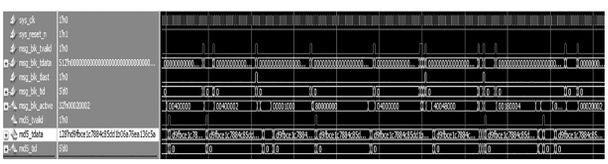

图5　功能仿真图

Fig.5　Functional simulation diagram

使用Arria10 FPGA开发板进行了板级验证,验证了该设计的正确性.该设计使用了1 1903个ALUTs,最大的时钟频率173 MHz,占用寄存器为12 883个,数据吞吐量最大为81.31 Gbps.表1列出了同类设计的资源消耗和性能比较.

表1　FPGA结果对比

Tab.1　FPGA result comparison

设计	频率/MHz	资源/个	吞吐量/Mbps
文献[1]	48	30 134	13 010
文献[6]	64.45	5 912	493.2
文献[7]	130	27 050	66 560
文献[16]	111	7 253	56 863
本设计	173.7	11 903	81 311

该设计应用于一款密码安全芯片,图6为该芯片的版图,采用0.18 μm工艺进行MPW流片,该芯片面积为6 mm^2,工作电压3.3 V,时钟频率为150 MHz时,功耗约为10.7 mW,测试结果表明,该设计功能正确.

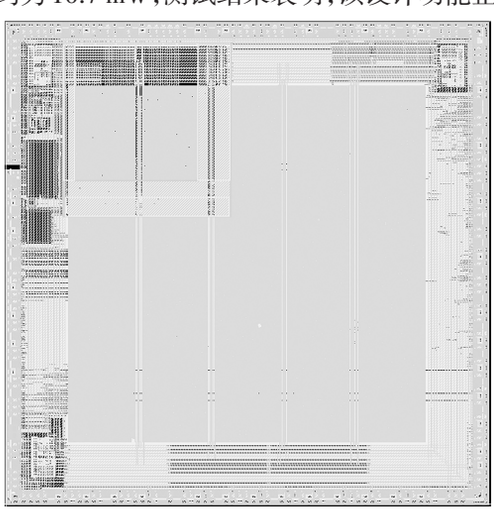

图6　芯片版图

Fig.6　Chip layout

4 结 语

MD5算法广泛应用于数字签名和验签,传统的软件或计算机硬件平台实现方法难以满足互联网时代对算法性能的要求.本文提出了一种通过优化加法器设计、循环移位操作缩短单步运算的关键路径,减少MD5运算的时钟周期的硬件实现方法;通过流水线设计,可同时计算32个通道消息的摘要值,且满载时每个时钟周期都可输出数据,大幅度提高了数据吞吐量.该设计使用VERILOG HDL语言实现,最后使用0.18 μm工艺进行流片.

参考文献

[1] HE D J, XUE Z. Multi-parallel architecture for MD5 implementations on FPGA with gigabit-level throughput[C]//2010 International Symposium on Intelligence Information Processing and Trusted Computing. Huanggang, China: IEEE, 2010: 535-538.

[2] IGNATIUS MOSES SETIADI D R, FAISHAL NAJIB A, RACHMAWANTO E H, *et al*. A comparative study MD5 and SHA1 algorithms to encrypt REST API authentication on mobile-based application[C]//2019 International Conference on Information and Communications Technology (ICOIACT). Yogyakarta, Indonesia: IEEE, 2019: 206-211.

[3] MOHAMMED ALI A, KADHIM FARHAN A. A novel improvement with an effective expansion to enhance the MD5 hash function for verification of a secure E-document[J]. IEEE Access, 2020, 8: 80290-80304.

[4] JARVINEN K, TOMMISKA M, SKYTTA J. Hardware implementation analysis of the MD5 hash algorithm[C]//Proceedings of the 38th Annual Hawaii International Conference on System Sciences. Big Island, HI, USA: IEEE, 2005: 298a.

[5] HOANG A T, YAMAZAKI K, OYANAGI S. Multi-stage pipelining MD5 implementations on FPGA with data forwarding[C]// 2008 16th International Symposium on Field-Programmable Custom Computing Machines. Stanford, CA, USA: IEEE, 2008: 271-272.

[6] WANG Y L, ZHAO Q X, JIANG L H, *et al*. Ultra high throughput implementations for MD5 hash algorithm on FPGA[C]//High Performance Computing and Applications, shanghai, China. lncs, 2010: 433-441.

[7] 韩津生,林家骏,叶建武,等. 基于FPGA的MD5高速处理模型设计[J]. 北京理工大学学报,2012,32(12):1258-1261.
HAN J S, LIN J J, YE J W, *et al*. Design of MD5 high-speed models on FPGA[J]. Transactions of Beijing Institute of Technology, 2012, 32(12): 1258-1261. (In Chinese)

[8] KHATRI V, AGARWAL V. Modified MD5 algorithm for low end IoT Edge devices[C]//2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT). Kanpur, India: IEEE, 2019: 1-6.

[9] 王孟钊. 安全散列算法的应用研究与实现[J]. 信息技术,2018,42(7):159-161.
WANG M Z. Application research and implementation of secure hash algorithm[J]. Information Technology, 2018, 42(7): 159-161. (In Chinese)

[10] KAREEM S M, RAHMA A M S. A new hybrid (MD5 and RC4) cryptography algorithm using multi-logic states[C]//2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS). Cairo, Egypt: IEEE, 2019: 285-292.

[11] SHADAB AHMAD Khan. FPGA implementation of MD5 algorithm for password storage[J]. International Journal of Science and Research, 2013, 4(6): 136-139

[12] 王波涛,韩国栋,张效军. 基于FPGA的MD5算法设计与实现[J]. 通信技术,2010,43(1):69-71.
WANG B T, HAN G D, ZHANG X J. Design and implementation of MD5 algorithm based on FPGA[J]. Communications Technology, 2010, 43(1): 69-71. (In Chinese)

[13] SUN Y H, WEI L F, LI P. Fast hardware implementation of MD5 algorithm[J]. Computer and Information Technology, 2007(5): 14-15.

[14] TAN J, ZHOU Q L. Implementation and improvement of MD5 algorithm in mimicry computer based on full pipeline architecture[J]. Small and Micro Computer System, 2017, 38(6): 1216-1220.

[15] WANG Y L, ZHAO Q X, JIANG L H, *et al*. Ultra high throughput implementations for MD5 hash algorithm on FPGA[M]//Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010: 433-441.

[16] 王臣,袁焱. 超高吞吐量MD5算法的FPGA实现[J]. 信息技术,2011,35(9):55-58.
WANG C, YUAN Y. A super-high throughput implementation of MD5 algorithm on FPGA[J]. Information Technology, 2011, 35(9): 55-58. (In Chinese)