高吞吐率流水线结构的ZUC-256流密码硬件设计

刘云涛[1,2],申泽生[1],方 硕[1,2],王 云[3]
(1. 哈尔滨工程大学信息与通信工程学院,黑龙江哈尔滨 150001;
2. 先进船舶通信与信息技术工业和信息化部重点实验室,黑龙江哈尔滨 150001;
3. 广东省大湾区集成电路与系统应用研究院,广东广州 510535)

摘 要: ZUC-256是为提供5G应用环境256 bit安全性而设计的流密码算法,数据处理速率是其核心性能之一,为此本文提出一种具有高吞吐率特性的硬件设计方案. 该方案采用流水线拆分关键路径初步提升系统工作频率,并提出一种完成模$(2^{31}-1)$加算法的优化电路进一步缩短关键路径延迟,该模加结构相较于现有结构缩短了42%的逻辑延迟,能够显著提升系统工作频率和吞吐率. 本研究分别采用Xilinx公司的Virtex-5器件、Alter公司的DE2-115器件和TSMC 90 nm工艺实现了该流密码硬件结构. 实验测试结果表明,采用TSMC 90 nm工艺实现的ASIC系统工作频率最高达到1200 MHz,吞吐率可达38.4 Gbps,比现有研究成果提升71%.

关键词: 5G;祖冲之算法;知识产权核;高吞吐率;流水线
基金项目: 黑龙江省自然科学基金面上项目(No.JJ2018ZR1021);中央高校基本科研业务费专项资金(No.3072021CF0806)
中图分类号: TN431.2 **文献标识码:** A **文章编号:** 0372-2112(2023)02-0438-08
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20211327

A Hardware Design of ZUC-256 Stream Cipher of Pipelining Structure with High Throughput

LIU Yun-tao[1,2], SHEN Ze-sheng[1], FANG Shuo[1,2], WANG Yun[3]
(1. *College of Information and Communication Engineering, Harbin Engineering University, Harbin, Heilongjiang* 150001, *China*;
2. *Key Laboratory of Advanced Marine Communication and Information Technology, Ministry of Industry and Information Technology, Harbin, Heilongjiang* 150001, *China*;
3. *Guangdong Greater Area Institute of Integrated Circuit and System, Guangzhou, Guangdong* 510535, *China*)

Abstract: ZUC-256 is a stream cipher algorithm designed to provide 256-bit security in a 5G application environment. The data processing rate is one of ZUC-256 core performances. Therefore, a hardware design scheme with high throughput characteristics is proposed. This scheme uses pipeline to split the critical path to increase the operating frequency of the system, and proposes an optimized circuit that completes the modular$(2^{31}-1)$ addition algorithm to further shorten the critical path delay. Compared with the conventional structure, the modular addition structure can shorten the path delay by 42%, which can significantly improve the working frequency and throughput rate of the system. This research implements the structure based on Xilinx's Virtex-5 device, Alter's DE2-115 device and TSMC 90 nm technology respectively. The experimental results shows that the ASCI system implemented by TSMC 90 nm technology achieves the maximum operating frequency of 1200 MHz, and the throughput rate can reach 38. 4 Gbps, which is 71% higher than the existing research results.

Key words: 5G; ZUC-256; IP core; throughput rate; pipe-line
Foundation Item(s): National Natural Science Foundation of Heilongjiang Province of China (No.JJ2018ZR1021); Fundamental Research Funds for the Central Universities (No.3072021CF0806)

1 引言

ZUC流密码又称为祖冲之算法,是首个由我国自主研发并成为国际标准的密码算法[1],该算法分为2种:ZUC-128和ZUC-256. ZUC-128流密码是3GPP机

收稿日期:2021-09-28;修回日期:2022-05-01;责任编辑:孙瑶

密性和完整性算法规范128-EEA3和128-EIA3的核心算法[2]. ZUC-256是为满足5G应用需求而提出的ZUC-128的升级版本[3],而5G网络技术以高速率和低延迟为核心目标,要求ZUC-256流密码每一帧产生高达2^{32}数据量的密钥流[4,5]. 因此,ZUC-256流密码的数据处理速率是其关键问题之一,探讨ZUC-256流密码算法的高吞吐率硬件设计具有重要意义[3,6].

目前大多数研究[6~11]只针对ZUC-128版本或软件实现进行了结构优化和分析. Wang等人[12,13]针对ZUC-128分别使用S-box分时复用、进位保存加法器(Carry Save Adder,CSA)和流水线结构设计了提升工作频率3种硬件实现方案,分别从硬件资源占用和增加吞吐率这2个方面优化,实现高达7.1 Gbps的吞吐率,但该流水线结构硬件方案未包含初始化阶段. Kitsos等人[14]从减少资源占用的角度完成仅含385 Slice的ZUC-128的硬件结构,该结构由于吞吐率较低的特性,仅适用于低速低功耗的应用环境. Wang等人[15]采用超前进位加法器(Carry Look-ahead Adder,CLA)替换关键路径的普通加法器完成ZUC-256版本的流水线结构,但CLA性能提升有限且严重增加了硬件资源占用. Yang等人[16]通过将关键路径的多个加法器拓展为34位宽来降低路径延迟,同时提出一种Sbox的优化替代算法,能够在FPGA平台较为明显的增加吞吐率;但是该优化方案仅缩短了FPGA平台下的线网延迟(net delay),而逻辑门数量仍然需要多达5个31位宽以上的加法器,逻辑门延迟很高,该方案较为适用于FPGA平台,而针对ASIC工艺下的吞吐率有着自身的局限性.

本文以提高ZUC算法吞吐率为目标,针对线性反馈移位寄存器LFSR设计了一种包含初始化和工作阶段的6级流水线结构,并提出一种快速完成模$(2^{31}-1)$加算法结构,2种结构相结合能够极大降低各寄存器之间的逻辑电路延迟,从而提高整体工作频率. 该高吞吐率ZUC流密码在FPGA平台完成原型验证,并对其性能做出评价.

2　ZUC-256算法描述

2.1　算法结构

本节完成ZUC-256算法的原理、结构和执行过程的详细描述[3]. ZUC-256是一种面向字的流密码算法,其输入数据为256 bit的密钥key和向量vector,经过33次迭代处理后每个循环输出一个32位密钥流.

ZUC的结构由线性反馈移位寄存器(Linear Feedback Shift Register,LFSR)、比特重组(Bit Reorganization,BR)和有限状态自动机(Finite State Machine,FSM)这3个模块组成(图1).

线性反馈移位寄存器LFSR用于计算和存储每次迭代操作的数据,包含16个31 bit的寄存器:$S_{15},S_{14},\cdots,S_1,S_0$. 首先LFSR完成$S_{16}$的计算,在工作模式下,$S_{16}$由式(1)得到,在初始化模式执行式(2)可得$S_{16}$,即

$$\begin{aligned}S_{16}=&\left(2^{15}S_{15}+2^{17}S_{13}+2^{21}S_{10}+2^{20}S_4\right.\\&\left.+2^8S_0+S_0\right)\bmod\left(2^{31}-1\right)\end{aligned}\tag{1}$$

$$\begin{aligned}S_{16}=&\left((W\gg1)+2^{15}S_{15}+2^{17}S_{13}+2^{21}S_{10}\right.\\&\left.+2^{20}S_4+2^8S_0+S_0\right)\bmod\left(2^{31}-1\right)\end{aligned}\tag{2}$$

其中,输入数据W是非线性变换函数FSM的32 bit输出数据,模$(2^{31}-1)$加运算的计算过程如式(3)和式(4)所示:

$$S=(A+B)\bmod\left(2^{31}-1\right)\tag{3}$$

$$\begin{cases}\{C,R\}=A+B\\ \text{if }(C=0)\quad S=R\\ \text{else }(C=1)\quad S=R+1\end{cases}\tag{4}$$

执行31 bit数据A和B相加运算,得到31 bit的结果数据R和1 bit的进位数据C. 如果进位等于0,模$(2^{31}-1)$加运算结果为R. 否则,C等于1,则结果为R+1. 如果S_{16}的计算结果等于零,则设置S_{16}为$(2^{31}-1)$,反之S_{16}数据不变. S_{16}的最终结果用于LFSR的移位操作:将$S_{16},S_{15},\cdots,S_2,S_1$的数据赋值给寄存器$S_{15},S_{14},\cdots,S_1,S_0$.

比特重组BR提取LFSR中的8个31 bit数据并进行线性重组,得到4个32位字X_0,X_1,X_2和X_3,式(5)~式(8)描述了线性重组过程,其中,S_{1H}表示数据S_1的高16位,S_{2L}是S_2的低16位数据.

$$X_0=S_{15H}\|S_{14L}\tag{5}$$

$$X_1=S_{11L}\|S_{9H}\tag{6}$$

$$X_2=S_{7L}\|S_{5H}\tag{7}$$

$$X_3=S_{2L}\|S_{0H}\tag{8}$$

有限状态自动机FSM对BR的输出数据X_0,X_1和X_2进行非线性变换,得到32位输出字W,计算过程如式(9)~式(13)所示:

$$W=\left[(X_0\oplus R_1)+R_2\right]\bmod\left(2^{32}\right)\tag{9}$$

$$W_1=\left[R_1+X_1\right]\bmod\left(2^{32}\right)\tag{10}$$

$$W_2=R_2+X_2\tag{11}$$

$$R_1=S_1\left[L_1\left(W_{1L}\|W_{2H}\right)\right]\tag{12}$$

$$R_2=S_2\left[L_2\left(W_{2L}\|W_{1H}\right)\right]\tag{13}$$

该非线性变换包括2个32 bit寄存器R_1和R_2,并涉及加法运算、异或运算、线性移位和S盒变换. ZUC的输出W用于LFSR的输入和32位输出密钥流的生成.

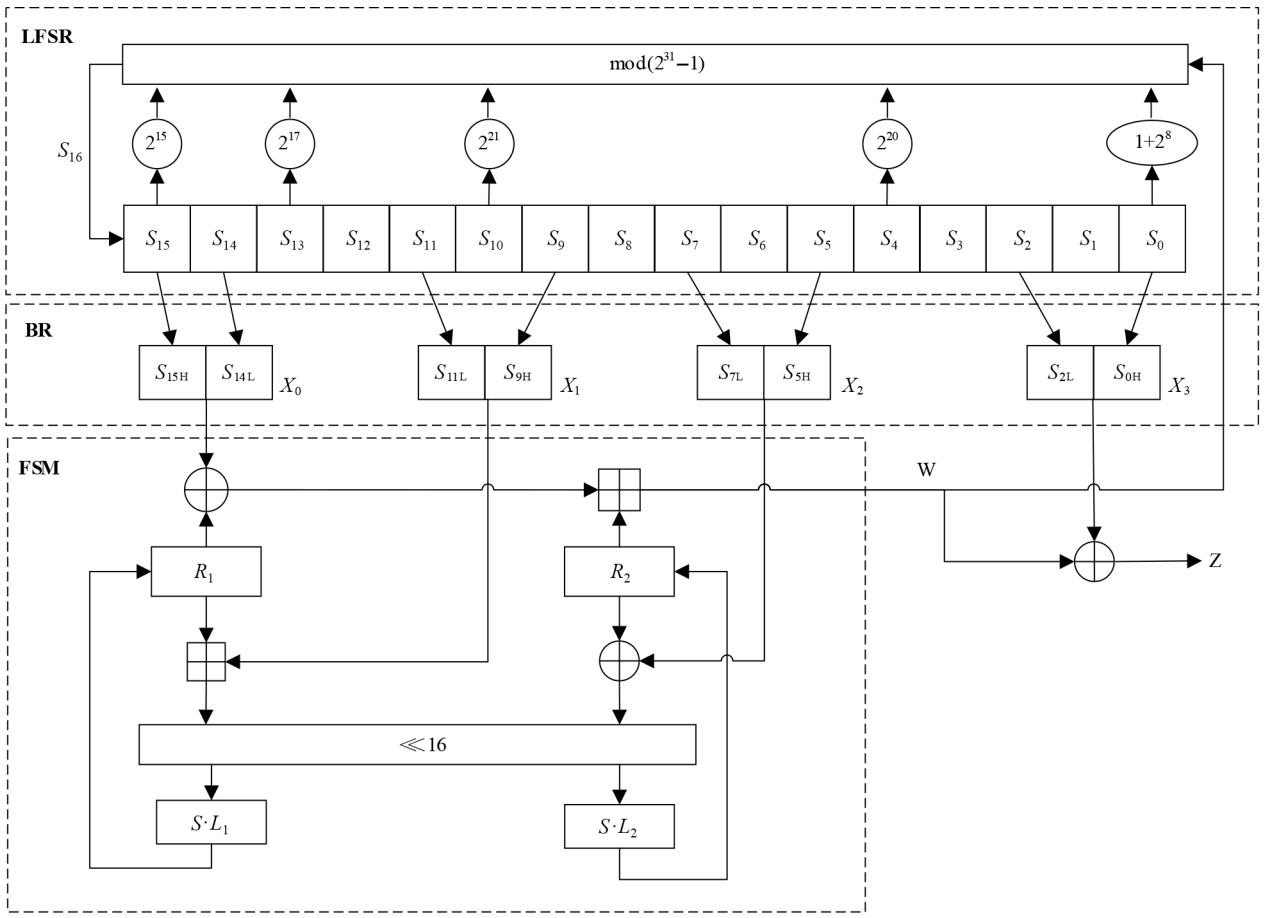


图1 ZUC-256流密码结构

2.2 算法执行流程

ZUC-256的执行过程分为2个阶段:初始化阶段和工作阶段.

初始化阶段流程如下:

(1)对LFSR的16个寄存器以及FSM中寄存器R_1和R_2赋初值[3];

(2)执行LFSR初始化模式,经过BR和FSM后,将输出W返回值LFSR;

(3)循环迭代32次过程(2);

(4)执行LFSR工作模式,同时执行一次BR和FSM过程.

工作阶段循环迭代执行LFSR工作模式、BR和FSM,并将FSM的输出W与BR的输出X_3进行异或运算,得到32位的密钥流输出.

3 硬件优化设计

3.1 流水线结构

根据上一节算法描述,该加密系统要求每个时钟周期输出32 bit的密钥流,因此系统的最高工作频率决定该算法的吞吐率,从降低关键路径延迟出发,需要提高工作频率来增加系统吞吐率.

在普通结构的LFSR中,其内部运算式(1)和式(2)严重限制了系统工作频率上限,因此对该关键路径的运算过程进行优化,采用多级流水线结构实现,算法整体硬件结构如图2所示.LFSR流水线包括16个31 bit的移位寄存器$S_0,S_1,\cdots,S_{15}$,5个31 bit流水线寄存器A,B,C,D,E,以及若干组合逻辑电路和时序控制电路.其中移位寄存器逐次相连,由移位信号shift控制移位赋值操作,末端寄存器S_{15}通过S_{16}连接至多路选择器MUX,由控制信号Mode完成移位寄存器的初始化阶段和工作阶段不同的数据更新.流水线寄存器组A,B,C,D,E通过5个模$(2^{31}-1)$加运算电路(ModA,ModB,ModC,ModD,ModE)和1个加法运算电路ADDE依次相连,分布完成式(1)和式(2)的运算,并缩短各寄存器之间的关键路径,其中模$(2^{31}-1)$的内部电路将在下一小节详细介绍.有限状态自动机FSM完成初始化阶段所需的数据W以及密钥流的生成过程,包括位宽为32 bit的寄存器R_1和R_2,辅以异或运算XOR、加法器CLA、线性变换电路($\ll 16$)和S盒移位变换SL_1,SL_2等组合电路,实现式(10)~式(13)的运算.R_1和R_2数据经过异或运算和加法器ADDW后传递至寄存器W,完成式(9)运

算.W经过异或运算后存入寄存器Z得到密钥流数据.从理论分析,本文提出的结构与普通结构相比,以增加5个31 bit寄存器资源为代价,最高能够将系统工作频率提升5倍.

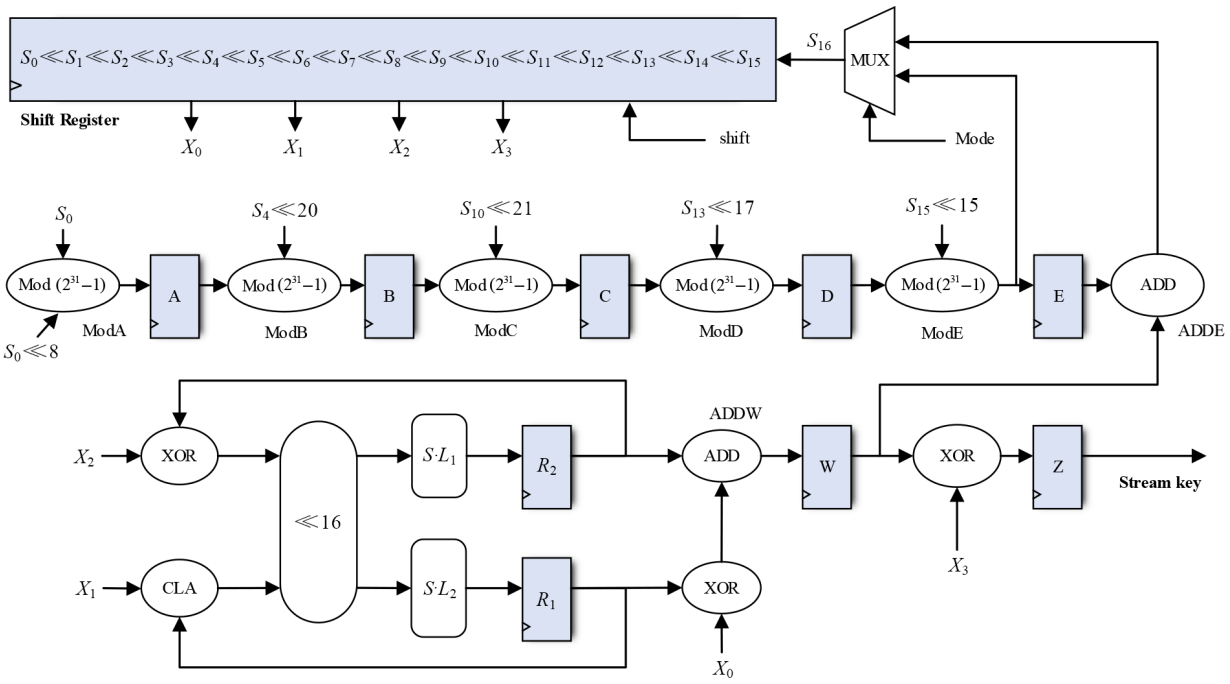


图2　硬件实现流水线结构

上述电路的时序转换由一个简单的状态机控制,根据ZUC算法加密流程,共包含以下4个状态:空状态(IDLE)、流水线装载状态(LOAD)、LFSR初始化状态(INIT)和LFSR工作状态(WORK).各状态的详细功能和转换过程如表1所示.

表1　内部状态转换过程

状态	跳转条件	状态	周期	主要功能
IDLE	启动信号	LOAD	-	复位各寄存器
LOAD	加载完成	INIT	5	初始化流水线组寄存器
INIT	初始化完成	WORK	63	完成ZUC初始化阶段
WORK	密钥输出完成	IDLE	-	完成ZUC工作阶段

在空状态IDLE下,系统不执行任何进程并持续复位内部寄存器,等待启动信号有效后进入流水线装载状态.LOAD状态主要完成对流水线寄存器A,B,C,D,E的数据初始化,由3位宽的计数器控制装载.图3为流水线装载和初始化阶段的部分时序转换图,其中各级寄存器的装载过程为LOAD的时钟周期T_0至T_4阶段.在T_0上升沿配置寄存器A,T_1上升沿更新寄存器B,依次同理在T_4上升沿更新寄存器E.T_4上升沿之前需要将寄存器R_1和R_2初始化为0,同时在T_4上升沿配置寄存器W的数据,该状态下不会对LFSR的移位寄存器S_0~S_{15}产生有效数据.

INIT状态完成第2.2节所述的初始化阶段,该状态下产生有效地模式信号Mode和移位信号Shift等,Mode控制MUX输入寄存器E和W之和,并按照式(2)产生新的S_{15}.初始化模式下的6级流水线结构具有反馈通路,由于LFSR需要输入31 bit的数据W完成线性移位,而FSM输出数据W又需要读取LFSR中寄存器数据,故执行一次初始化迭代过程需要两个时钟周期完成,其初始化模式下寄存器装载时序如图3中INIT阶段所示.移位信号Shift用于控制LFSR的移位操作,在LOAD状态T_4期间其高电平有效,在INIT状态T_0上升沿将寄存器E和W的计算结果更新至S_{15},此时即完成了一次初始化迭代过程.第二次初始化过程由LOAD状态T_2上升沿的寄存器A开始,经过5个周期后在INIT状态的T_2的上升沿更新至寄存器S_{15},同理LOAD状态T_4开始的流水线将在INIT状态的T_4得到有效数据.按照上述过程执行32次迭代运算后,移位寄存器组$S_0,S_1,\cdots,S_{15}$初始化完毕并得到正确的数据.

在INIT状态的末尾周期内,将同时完成对WORK状态的流水线寄存器A,B,C,D的重装载过程,以节省流水线的装载时间,不需要额外的增加时钟周期,直接由LOAD状态的时钟T_{62}进入WORK状态的T_0.INIT状态至WORK状态的流水线重装载如表2所示.寄存器A在LOAD状态T_{60}完成最后一次迭代的数据运算后,在T_{60}转换为ZUC工作阶段的流水线装载,此时将存储数据转换为S_3和2^8S_3的模$(2^{31}-1)$加运算结果.

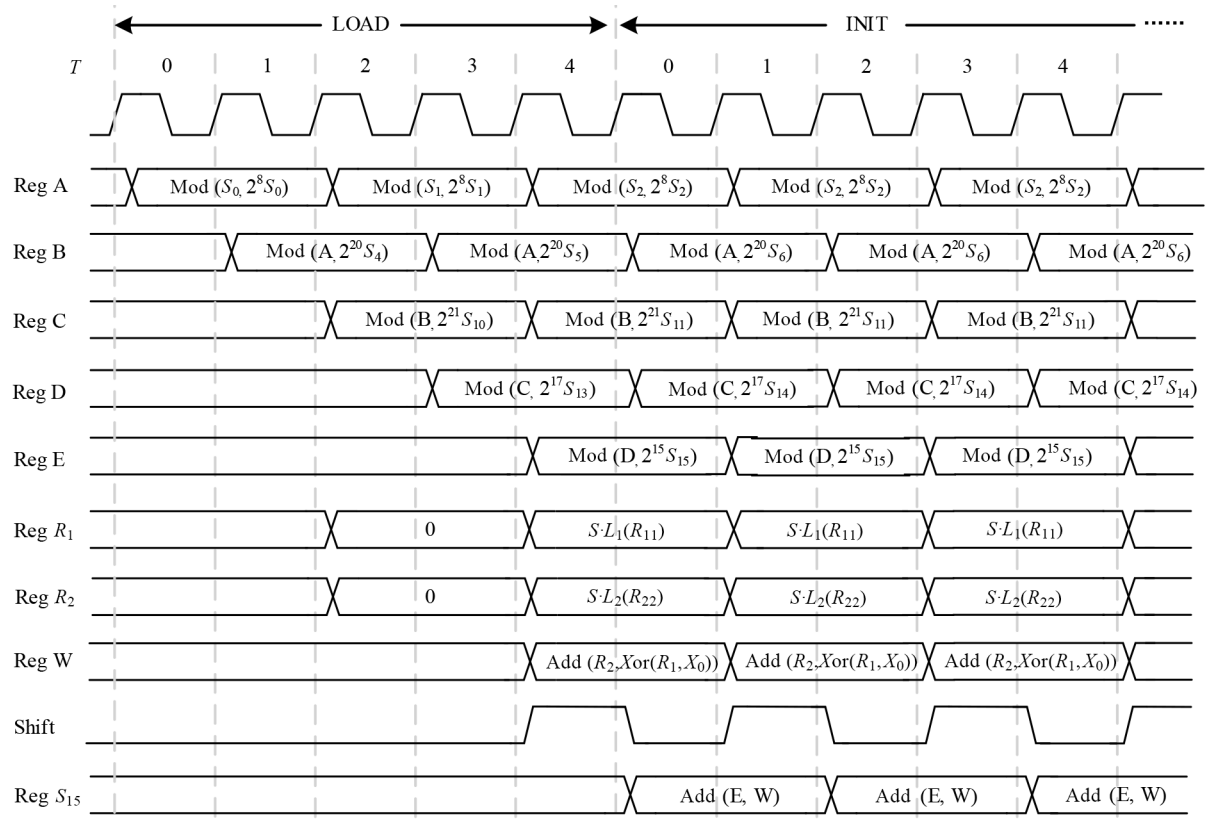


图3 初始化模式的时序转换图

表2 WORK状态与INIT状态流水线重装载过程

状态	LOAD			WORK			
时钟	T_{60}	T_{61}	T_{62}	T_0	T_1	T_2	T_3
寄存器A	S_2, 2^8S_2	S_3, 2^8S_3	S_3, 2^8S_3	S_4, 2^8S_4	S_4, 2^8S_4	S_4, 2^8S_4	S_4, 2^8S_4
寄存器B	A, $2^{20}S_6$	A, $2^{20}S_6$	A, $2^{20}S_6$	A, $2^{20}S_7$	A, $2^{20}S_7$	A, $2^{20}S_7$	A, $2^{20}S_7$
寄存器C	B, $2^{21}S_{11}$	B, $2^{21}S_{11}$	B, $2^{21}S_{11}$	B, $2^{21}S_{12}$	B, $2^{21}S_{12}$	B, $2^{21}S_{12}$	B, $2^{21}S_{12}$
寄存器D	C, $2^{17}S_{14}$	C, $2^{17}S_{14}$	C, $2^{17}S_{14}$	C, $2^{17}S_{14}$	C, $2^{17}S_{14}$	C, $2^{17}S_{14}$	C, $2^{17}S_{14}$

WORK状态下完成ZUC工作阶段的密钥流生成过程,根据算法规则在时钟T_0进行一次数据缓冲,T_2开始得到密钥流数据. MUX将ModE的运算结果赋值给寄存器S_{15},由于没有流水线寄存器的回路,Shift信号持续将输出高电平,确保在每个时钟周期都会完成移位寄存器组数据更新,并输出32 bit的密钥流数据.

3.2 模$(2^{31}-1)$加运算的优化结构

在上述流水线结构中,关键路径为各寄存器之间的模$(2^{31}-1)$加运算. 为进一步缩短关键路径,本文提出一种高速完成模$(2^{31}-1)$加运算的结构,该结构如图4(a)所示,由2个16位加法器、2个15位加法器和5个多路选择器MUX组成,该结构将自上而下分为3级.

第1级由4个加法器组成. 首先16位加法器Adder0计算进位为1时A和B的低16位(A_L,B_L)之和,得到1位进位信号C_{L1}和16位数据S_{L1},如式(14)所示. 同理16位加法器Adder1计算进位为0时A和B的低16位(A_L,B_L)之和,得到1位进位信号C_{L0}和16位数据S_{L0},如式(15)所示. 15位加法器Adder2和Adder3分别计算进位为0和进位为1时A和B的高15位(A_H,B_H)之和,得到1位进位信号C_{H0}和16位数据S_{H0},如式(16)和式(17)所示.

$$C_{L1}\| S_{L1}=A_L+B_L+1 \tag{14}$$

$$C_{L0}\| S_{L0}=A_L+B_L+0 \tag{15}$$

$$C_{H1}\| S_{H1}=A_H+B_H+1 \tag{16}$$

$$C_{H0}\| S_{H0}=A_H+B_H+0 \tag{17}$$

第2级由3个MUX组成. 第1个MUX由C_{L0}选择C_{H1}和C_{H0}可得到的A+B的进位C,如式(18)所示. 第2个MUX由16位加法器1的进位信号C_{L0}选择出高15位数据之和S_{HL0},如式(19)所示,该数据即总进位为0时模$(2^{31}-1)$加法运算结果的高15位数据S_{HL0}. 同理第3个MUX选择出总进位为1时模$(2^{31}-1)$加法运算结果的

高15位数据S_{HL1},如式(20)所示.

$$C=C_{L0}\cdot C_{H1}+\overline{C_{L0}}\cdot C_{H0} \tag{18}$$

$$S_{HL0}=C_{L0}\cdot S_{H1}+\overline{C_{L0}}\cdot S_{H0} \tag{19}$$

$$S_{HL1}=C_{L1}\cdot S_{H1}+\overline{C_{L1}}\cdot S_{H0} \tag{20}$$

第3级由2个MUX组成. 第1个MUX由总进位信号C选择模$(2^{31}-1)$加运算结果的高15位数据S_H. 第2个MUX由总进位信号C选择模$(2^{31}-1)$加运算结果的低16位数据S_L.

文献[13]、文献[14]和文献[17]设计的模$(2^{31}-1)$加结构如图4(b)所示,该结构由两个31 bit加法器和1个MUX级联而成;文献[15]采用图4(c)所示的加法器结构,该方案下2个31 bit加法器分别计算进位为0和进位为1时的结果,然后使用1级MUX得到模$(2^{31}-1)$加运算结果. 其中文献[15]将两个31位加法器采用CLA结构实现,文献[17]采用一种优化后的CLA结构完成来降低模$(2^{31}-1)$运算的资源占用. 本文方案没有对15位加法器和16位加法器内部结构进行详细规定,普通逐位进位加法器、CSA或CLA等均适用于本文结构,采用CLA结构同样能够显著提升运算速度. 在此统一使用逐位进位加法器结构对比,31位加法器需要31个全加器级联,1个全加器需要2级门电路延迟,1级MUX需要3级门电路延迟. 故图4(b)结构在采用逐位进位加法器时逻辑延迟为127级;图4(c)结构的门电路延迟为65级. 本文结构的最长延迟路径为16个全加器和2级MUX,门级逻辑总延迟为38级;本文结构相较于文献[13, 14, 17]逻辑延迟缩短了70%,相较于文献[15]逻辑延迟缩短了42%,且硬件资源只需额外增加4个多路选择器.

4　算法实现与性能分析

根据本文提出的结构使用Verilog硬件描述语言完成RTL级设计,使用ISE design suite软件,在Xilinx公司的Virtex-5系列器件环境下进行与综合,硬件仿真环境如图5所示. 基于Synopsis公司的VCS工具完成功能验证,使用高级程序语言完成算法验证平台,通过ZUC-256的高级算法模型与硬件模型进行输入输出对比,以确保功能正确性. 图6描述了电路内部关键节点的数据变换过程,输入密钥K_i=0xff, $0\leqslant i\leqslant 31$,输入向量IV_i=0xff, IV_j=0x3f, $0\leqslant i\leqslant 16$, $17\leqslant j\leqslant 24$,经过装载状态LOAD和初始化状态INIT后,进入工作状态WORK,在每个时钟周期得到32位宽的密钥流keystream:3356cbae,d1a1c18b,6baa4ffe,…,测试结果正确.

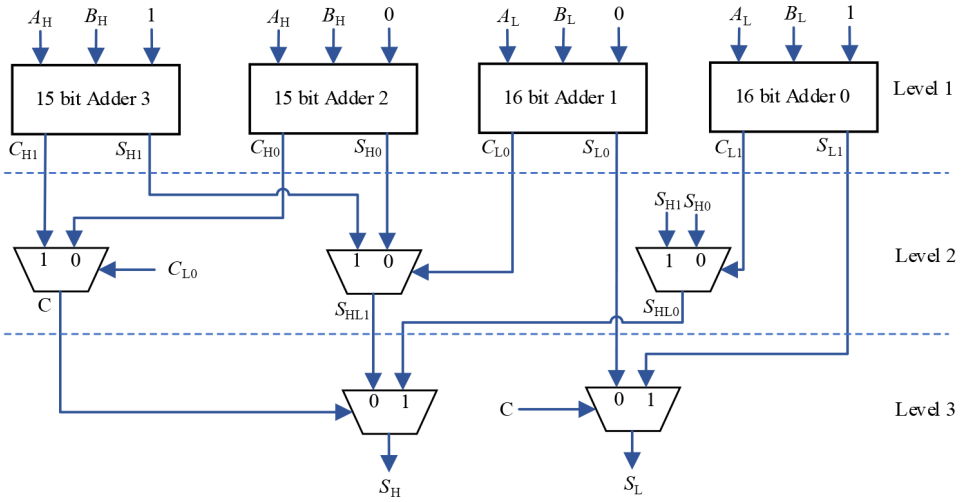
表3对本文的实验成果与已发表的研究成果从路径延迟、最高工作频率、资源占用与吞吐率方面进行了详细对比. ZUC流密码于每个时钟周期输出32 bit密钥流,其系统吞吐率为最高工作频率的32倍. 首先从系统的关键路径式(1)的门电路延迟对比. 文献[14]采用了3级模$(2^{31}-1)$加运算完成式(1),其中每一级采用图4(b)的结构,共需要381级的门电路延迟. 文献[15]基于图4(c)采用流水线结构,共需要35级门电路延迟. 文献[13]的方案2采用3级31位CSA加法器和1级图4(b)模$(2^{31}-1)$加运算结构级联,共需要313级门电路延迟,而文献[16]关键路径由5级34位加法器和1级31位加法器级联而成,共需要201级门电路延迟. 本文结构关键路径为图4(c)所示结构,共包含38级门电路,因此本文结构的关键路径延迟相较于文献[15]缩短了42%.

本文提出的结构在吞吐率性能上与文献[14]相比有较大优势,相较于文献[14]提升153%. 与文献[13]的全硬件方案2对比,本文方案在吞吐率性能上提升了52%. 为进行更细致准确的对照实验,本文使用Alter公司的DE2-115器件进行了仿真和实现,与文献[15]相比性能提升21%. 值得注意的是,文献[13]的方案3的吞吐率达到了7.1 Gbps,这是因为该方案的硬件结构不包含初始化阶段,其加密过程需要软硬件结合实现,故而降低了关键路径的延迟,使其吞吐量数值较高. 而本文结构能够全硬件实现ZUC-256的整个加密过程,硬件集成度更高.

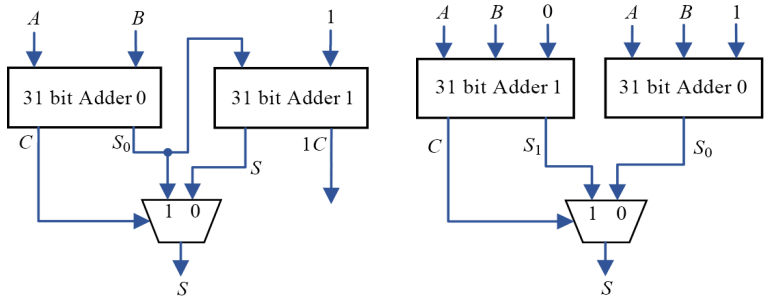
由于FPGA平台自身局限性,在本系统的关键路径延迟中,线网延迟(net delay)占比高达95%,门电路的自身延迟仅占比5%. 为更好地体现本文方案的优越性,基于TSMC 90 nm工艺下采用Synopsis公司的EDA工具对该系统进行设计与分析,在所有加法器采用普

表3　ZUC流密码的实现方案性能对比

方案	器件	关键路径/级	频率/MHz	面积/slice	吞吐率/Gbps
文献[14]	Virtex-5	381	65	385	2.08
文献[15]	DE2-115	65	115	—	3.68
文献[13] -方案3	Virtex-5	127	222.4	575	7.10
文献[13]-方案2	Virtex-5	313	108	356	3.46
文献[16]	TSMC 90 nm	201	700	—	22.40
本文	Virtex-5	38	164.4	847	5.26
本文	DE2-115	38	139.1	—	4.45
本文	TSMC 90 nm	38	1 200	—	38.40



(*a*) 本文的优化结构



(*b*) 文献[13,14,17]结构 (*c*) 文献[15]的结构

图4 模($2^{31}-1$)加运算硬件结构

图5 ZUC-256硬件实现环境

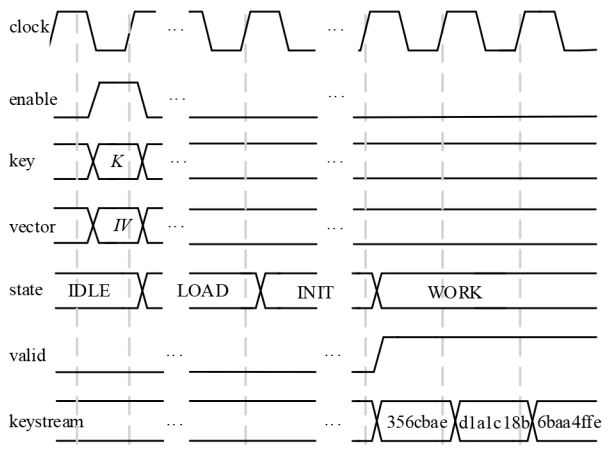


图6 ZUC-256输入输出系统验证模型

通加法器的条件下,整个系统的关键路径延迟为0.82 ns,分析可知最高频率约为1.2 GHz,吞吐率可达38.4 Gbps.在同样的实验条件下,对文献[16]提出的方案进行复现实验,其关键路径为寄存器S_0至寄存器S_{15},路径延迟为1.40 ns,最高频率约为0.7 GHz,吞吐率为22.4 Gbps.对比可知,在ASIC设计工艺下,本文设计方案相较于文献[16]提升了71%.

从资源占用方面对比,本文结构占用芯片面积较大,这是因为采用的LFSR和FSM流水线结构造成了资源使用量的增加.但是,相较于资源上的一定损耗,本文结构能够大幅降低关键路径的门级延迟从而提高系统的工作频率,在吞吐率性能上具有更为明显的优势.

5 结束语

本文从提高吞吐率出发,完成了流水线结构的ZUC-256流密码的硬件实现,并提出一种快速实现模$(2^{31}-1)$加运算的结构. 该结构相较于现有的模$(2^{31}-1)$加运算结构逻辑延迟缩短了42%,极大地减小了关键路径延迟,提升了系统吞吐率,使ZUC-256算法的吞吐率达到5.26 Gbit/s,相较于现有研究成果提升21%,能够更加适用于如5G等对处理速度要求较高的环境中.

参考文献

[1] 冯秀涛. 3GPP LTE国际加密标准ZUC算法[J]. 信息安全与通信保密, 2011, 9(12): 45-46.
FENG X T. ZUC algorithm: 3GPP LTE international encryption standard[J]. Information Security and Communications Privacy, 2011, 9(12): 45-46. (in Chinese)

[2] ETSI. Specification of the 3GPP confidentiality and integrity algorithms 128-EEA3 & 128-EIA3(Document 2: ZUC Specification): TS 35.222-2011[S]. Sophia Antipoli: SAGE, 2011.

[3] YANG J, JOHANSSON T. An overview of cryptographic primitives for possible use in 5G and beyond[J]. Science China Information Sciences, 2020, 63(12): 1-22.

[4] ZUC算法研制组. ZUC-256流密码算法[J]. 密码学报, 2018, 5(2): 167-179.
Team Design. ZUC-256 stream cipher[J]. Journal of Cryptologic Research, 2018, 5(2): 167-179. (in Chinese)

[5] ANSI. Information technology - security techniques - IT network security - Part 4: securing remote access: ANSI/INCITS/ISO/IEC 18033-4:2005[S]. Washington: ANSI, 2005.

[6] DRUCKERN,GUERONS.Fastconstanttimeimplementations of ZUC-256 on x86 CPUs[C]//2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC). Las Vegas: IEEE, 2019: DOI:10.1109/CCNC.2019.8651851.

[7] 李建鹏, 张艳硕, 董秀则, 等. ZUC-256算法的快速软件实现[J]. 计算机应用研究, 2019, 36(12): 3797-3800.
LI J P, ZHANG Y S, DONG X Z, et al. Fast software implementation of ZUC-256 algorithm[J]. Application Research of Computers, 2019, 36(12): 3797-3800. (in Chinese)

[8] MADANI M, TANOUGAST C. Combined and robust SNOW-ZUC algorithm based on chaotic system[C]//2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security). Glasgow: IEEE, 2018: 1-7.

[9] HUO Y H, LIU D K. High-throughput area-efficient processor for cryptography[J]. Chinese Journal of Electronics, 2017, 26(3): 514-521.

[10] WEI M M, YANG G Q, KONG F Y. Software implementation and comparison of ZUC-256, SNOW-V, and AES-256 on RISC-V platform[C]//2021 IEEE International Conference on Information Communication and Software Engineering. Chengdu: IEEE, 2021: 56-60.

[11] KRISHNA P L. Optimization of ZUC stream cipher to attain high throughput (Gbps) [J]. International Journal of Innovative Technology and Exploring Engineering, 2019, 8(8): 707-710.

[12] LIU Z, ZHANG L, JING J, et al. Efficient pipelined stream cipher ZUC algorithm in FPGA[C]//The First International Workshop on ZUC Algorithm. Beijing: [s.n.], 2010: 1-5.

[13] WANG L, JING J W, LIU Z B, et al. Evaluating optimized implementations of stream cipher ZUC algorithm on FPGA[M]//Information and Communications Security. Berlin: Springer, 2011: 202-215.

[14] KITSOS P, SKLAVOS N, SKODRAS A N. An FPGA implementation of the ZUC stream cipher[C]//2011 14th Euromicro Conference on Digital System Design. Oulu: IEEE, 2011: 814-817.

[15] WANG Y K, WU L J, ZHANG X M, et al. A hardware implementation of ZUC-256 stream cipher[C]//2020 IEEE 14th International Conference on Anti-counterfeiting, Security, and Identification. Xiamen: IEEE, 2020: 94-97.

[16] YANG Y T, ZHAO W C, XIONG L Q, et al. Optimized implementations for ZUC-256 on FPGA[J]. Wireless Personal Communications, 2021, 116(3): 2615-2632.

[17] KHARADKAR R D, HULLE N B. FPGA implementation of modulo $(2^{31}-1)$ adder[C]//2015 7th International Conference on Emerging Trends in Engineering & Technology (ICETET). Kobe: IEEE, 2015: 85-90.

作者简介

刘云涛　男,1980年出生,黑龙江鹤岗人. 工学博士. 副教授,博士生导师. 主要研究方向为模拟/数模混合集成电路设计、CMOS图像传感器.
E-mail: lyt@hrbeu.edu.cn

申泽生　男,1997年出生,山东泰安人. 硕士. 主要研究方向为数字集成电路设计、硬件加密算法、数字信号处理.
E-mail: fateszs@163.com