doi:10.3969/j.issn.1672-1292.2019.01.004

基于 Lorenz 系统的离散分析与 FPGA 实现

金秋森,闵富红,吕晏旻

(南京师范大学南瑞电气与自动化学院,江苏 南京 210023)

[**摘要**] 以现场可编程门阵列(field-programmable gate array,FPGA)和 IEEE754 标准为基础,设计出能产生 Lorenz 混沌信号的数字电路. 首先,根据 Euler 法、Runge-Kutta 法,分别将系统方程离散化. 然后,利用 Verilog HDL 语言编写程序,运用 Xilinx 软件、Modelsim 软件将程序综合、编译、检测. 最终,将生成的 bit 文件烧录到 FPGA 中,通过示波器观测系统的混沌态与非混沌态. 对比论证不同算法的实现效果,得出二阶 Runge-Kutta 法是实现经典混沌系统的 FPGA 仿真的最优离散方法,为后续混沌信号在数字化领域的进一步发展提供参考和依据.

[**关键词**] Lorenz 系统,离散化,FPGA,Verilog HDL

[**中图分类号**]TP391.3　[**文献标志码**]A　[**文章编号**]1672-1292(2019)01-0029-11

Discrete Analysis and FPGA Implementation Based on Lorenz System

Jin Qiusen, Min Fuhong, Lü Yanmin

(School of NARI Electrical and Automation, Nanjing Normal University, Nanjing 210023, China)

Abstract: Based on the field-programmable gate array(FPGA) and IEEE754 standard, digital circuits capable of generating Lorenz chaotic signals are designed and demonstrated. Firstly, the system equations are discretized according to the Euler method and the Runge-Kutta method. Then, the program is written in Verilog HDL language, and the program is integrated, compiled and tested through Xilinx software and Modelsim software. Finally, the generated bit file is burned into FPGA to observe the chaotic and non-chaotic states of the system through an oscilloscope. By comparing and demonstrating the implementation effect of different algorithms, it is concluded that the second-order Runge-Kutta method is the optimal discrete method to realize the FPGA simulation of the classical chaotic system, providing reference and basis for the further development of the subsequent chaotic signal in the digital field.

Key words: Lorenz system, discretization, field-programmable gate array(FPGA), Verilog HDL

随着计算机信息技术、机械制造与自动化技术、人工智能科学的飞速发展,数字信号处理技术已广泛应用于通信、医学、勘测等多个学科领域[1]. 相较于模拟信号处理技术,数字信号处理技术灵活性强、精确度高,应用范围更为广泛,且具有开发成本低、周期短等优势. 目前数字信号处理技术主要包括 PLC、DSP、FPGA、CPLD,其中 FPGA 技术的应用最为广泛[2]. 1985 年,IEEE 制定了相关的 IEEE-754 标准,解决了利用 FPGA 实现浮点数四则运算的问题,弥补了 FPGA 本身只能做整体运算的缺点,为 FPGA 的数字信号处理奠定了实践基础. 在探索 FPGA 技术实现混沌系统的过程中,大部分学者采用的是类似于文献[3-6]中的方法,即首先通过 DSP-Builder 搭建电路,再由电路的模型转化成 HDL 语言输出,导入代码至 FPGA 开发板的方法. 相比而言,直接利用硬件语言对混沌系统进行编程的方法拥有系统模型占内存较小、时序控制性、可移植性和拓展性较强等优点,但却鲜有学者进行讨论研究[7].

Lorenz 作为最具代表的经典连续时间混沌系统,由美国气象学家 Lorenz 通过观察气象现象并进行大量数据实验得出,拥有丰富的动力学特性,在不同参数的条件下,系统的运动轨迹会产生不同的周期、混沌

收稿日期:2018-10-19.

基金项目:国家自然科学基金课题(61871230)、江苏省自然科学基金课题(BK2018021196)、中国江苏省研究生研究与创新实践项目(KYCX18_1220).

通讯联系人:闵富红,博士,教授,研究方向:非线性电路与系统. E-mail:minfuhong@ njnu.edu.cn

状态,揭示了一系列复杂混沌特性的本质. 因此,Lorenz 系统的非线性特性被广泛应用实践工程领域,众多研究和分析都在其基础上产生. 为了产生更复杂的吸引子,文献[8-9]通过在 Lorenz 方程添加新的控制器或者变量,获得了新的系统并完成了硬件电路实现. 文献[10-11]基于 Lorenz 系统对分数阶、时滞、同步做了大量研究. 文献[12-13]基于 Lorenz 系统改进算法,利用其产生的混沌秘钥和混沌序列,探索了非线性理论在图像加密、MIMO 雷达信号设计的应用前景.

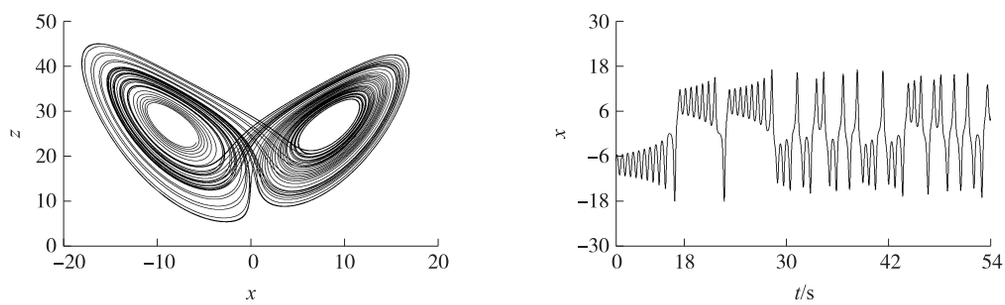
本文将以经典的 Lorenz 系统为研究对象,从设计和验证两方面论证了利用 Verilog HDL 语言和 FPGA 开发板对混沌系统进行数字信息化处理的实现. 设计包括 3 种不同离散算法的选择、程序功能的分割、模块的搭建. 验证包括不同离散算法的对比、示波效果的比较、程序功能的检验等. 通过对比不同离散算法的仿真效果,得出在兼顾精确率和资源调用情况下,二阶 Runge-Kutta 法是最优的离散方案.

1　系统模型

Lorenz 数学模型是基于纳维-斯托克斯方程、热传导方程和连续性方程简化得出,属于耗散系统. 作为一个典型的自治系统,其代数方程主要由线性和简单的二次非线性组成[14],具体如下:

$$\begin{cases}\dot{x}=-a(x-y),\\ \dot{y}=cx-xz-y,\\ \dot{z}=xy-bz.\end{cases}\tag{1}$$

式中,a、b、c 为系统的控制参数,决定了系统不同的运动状态. 当 $a=10$,$b=8/3$,$c=28$,$x(0)=0.12$,$y(0)=0.1$,$z(0)=0.15$ 时,利用 MATLAB 仿真可得如图 1 所示 Lorenz 混沌系统的 x-z 相图和 x 时序波形图.



(a) x-z相图　　(b) x时序图

图 1　MATLAB 数值仿真图

Fig. 1　MATLAB numerical simulation

2　FPGA 及开发环境介绍

本文采用直接编写 Verilog HDL 语言的方法实现 FPGA. 其中,Verilog HDL 语言是对硬件的描述,用该语言实现的结果不是程序而是电路. 此次,FPGA 所用芯片为 Xilinx 厂商的 Spartan-6 系列芯片,具体型号是 XC6SLX16. 其片内资源主要包括:①14 579 个逻辑单元(Logic Cell);②2 278 个 Slices,每个 Slices 包含 4 个六输入查找表(LUT)8 个触发器(Flip-Flops);③18 224 个触发器(Flip-Flops);④32 个 DSP48A1 Slices,每个 DSP48A1 能进行 18×18 的乘法运算,不过该运算只能是整数运算;⑤18 kb×32 个 Block RAM Blocks.

本实验设计的开发环境是 Xilinx 提供的 ISE(integrated software environment)套件,其功能强大,集成度高,能对程序进行合理的优化并发出警告. 具有强大的布局布线、在线调试仿真能力,能加快项目的进展减少电路设计者的负担.

为了节约设计成本和开发时间,本次设计各模块中的浮点乘法器与加法器将直接使用集成开发环境提供的 IP 核. 浮点运算 IP 核输入输出浮点数皆遵循 IEEE754 标准. 使用 IP 核时,需要把定点数转为 IEEE754 标准的二进制浮点数进行计算,该标准定义的浮点数分为单精度和双精度两种类型,双精度占用资源较多,本次设计只使用单精度.

3 利用 Euler 算法分析

3.1 方程离散化

FPGA 由于是半定制集成电路,其功能需要使用者定义. 与 DSP 不同,FPGA 只能进行简单的整数运算,既不支持小数运算,也不支持积分与微分运算. 若要在 FPGA 实现 Lorenz 系统需要把该方程离散化,把微分转为积与和的运算. 离散化的方法不同精度也会不同,首先采用精度较低的 Euler 法进行离散. 为了使 x,y,z 轴不超过 DA 转换的±5 V 范围,x,y,z 同时除以参数 $E=0.1$,取时间步长 $h=0.005$,参数 $a=10$,$b=8/3$,$c=28$,初始值 $x(0)=0.12$,$y(0)=0.1$,$z(0)=0.15$,将系统离散化可得:

$$\begin{cases} x_{i+1}=0.95x_i+0.05y_i, \\ y_{i+1}=0.14x_i+0.995y_i-0.05x_iz_i, \\ z_{i+1}=0.05x_iy_i-0.986\ 67z_i. \end{cases} \tag{2}$$

3.2 程序的设计与仿真

用 Verilog 编写基于 Euler 离散化算法的模块. 模块的整体连接图如图 2 所示,由 Module_Lorenz 模块、Module_change 模块两部分组成. 左边的模块为 Module_Lorenz 模块、右边为 Module_change 模块. 其中,Module_Lorenz 模块有 2 个输入,4 个输出,输出口与 DA 转换(Module_change 模块)连接.

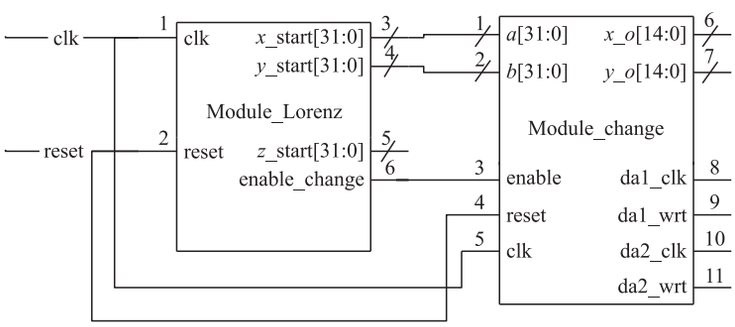


图 2 Euler 法模块整体连接图

Fig. 2 Overall module connection diagram of Euler method

(1)Module_Lorenz 模块计算的流程图如图 3 所示,一共用到了 4 个乘法器,2 个加法器,5 个状态,每个状态进行不同的运算. 程序按照流程图所示顺序依次使用乘法器和加法器进行计算,把得到的结果重新赋值给 x_start,y_start,z_start,再次进行循环计算.

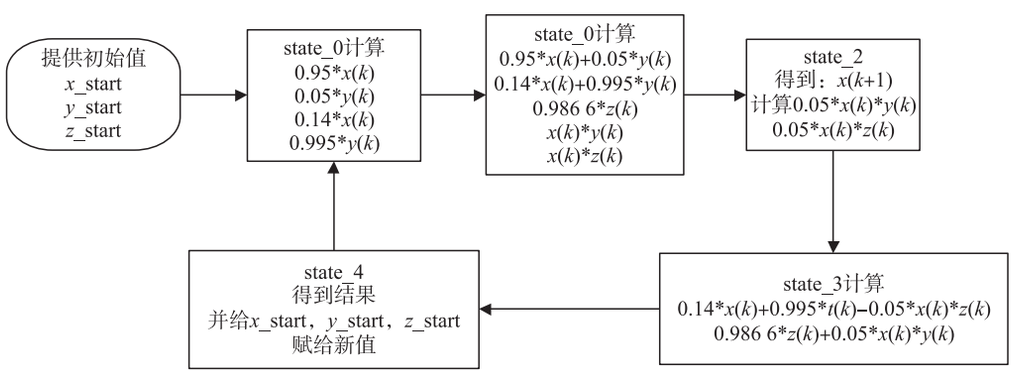


图 3 Module_Lorenz 模块流程图

Fig. 3 Flow chart of Module_Lorenz module

(2)Module_change 模块为高速 DA 转换服务的,DA 采用的芯片是 AD9767. 该模块包含了把浮点数转为定点数的 IP 核,在转换之前要先把 Module_Lorenz 输出的结果计算成 DAC 芯片的输入值,得到这些值之后再用浮点转定点的 IP 进行转换和输出. Module_change 模块占用的资源较少,每个乘法器和加法器只需要完成对应的一个特定运算即可,运行流程简单,没有多个状态的先后次序,因此它们之间的连接采用线型,相当于电路加电后同时运算一样,要保持输出不变只要输入不变即可.

3.3　实验结果

修改上述程序的顶层模块,添加 test 文件将上述模块综合、编译之后,点击 Xilinx 软件中的 Modelsim Simulator 选项,可以调出 Modelsim 软件,来验证程序是否正常运行. 查看 Module_Lorenz 运行的时序图如图 4 所示,可以看出该模块中的乘法器和加法器随着 clk 时钟信号按顺序依次进行计算,最终把得到的结果重新赋值给 x_start,y_start,z_start 进行循环计算. 当模块的变量较多时候,可以先从 state 开始检查,不同的 state 对应着不同的计算,整个程序是根据状态的转移,按顺序运行的.

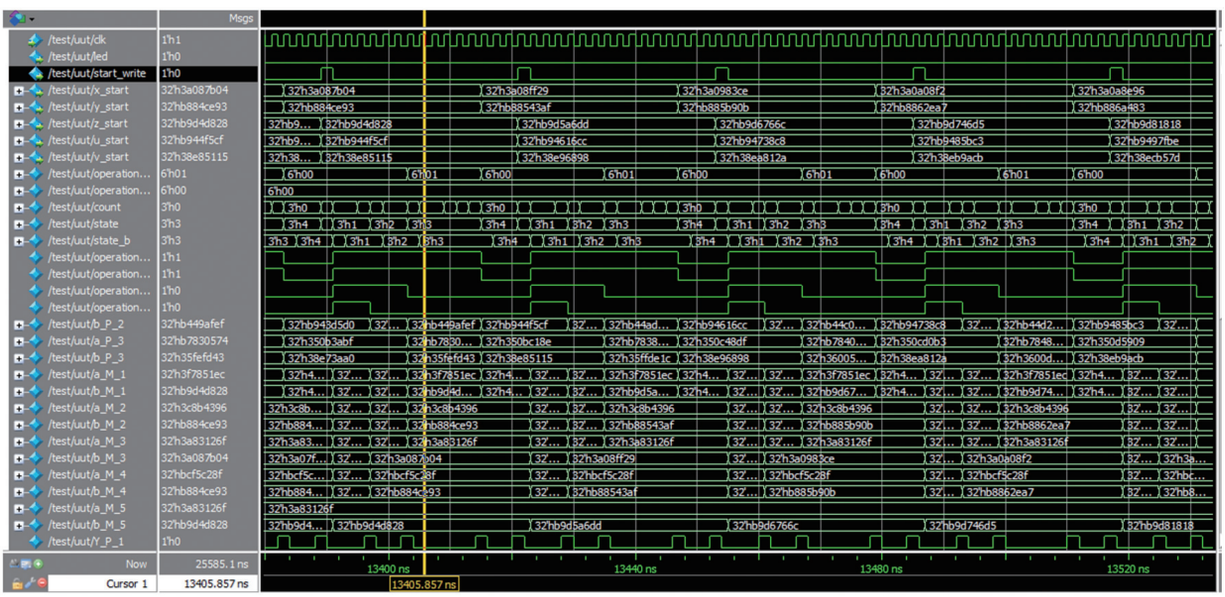


图 4　Modelsim 联合调试时序图

Fig. 4　Time sequence diagram of Modelsim joint debugging

将 FPGA 开发板分别连接至电脑和示波器的两端,如图 5 所示. 将顶层模块更改为原程序,添加用户约束文件(ucf)定义程序对应的引脚,再次将程序综合、翻译、映射、布局布线,点击 Xilinx 软件中的 Configure Target Device 选项,生成 bit 文件. 点击 Programme Download,将 bit 文件烧录至 FPGA 开发板,点击开发板 Reset 键,示波器上将会显现完整的数值仿真波形. 调整参数,将数据 $a=10$,$b=8/3$,$c=28$,$x(0)=0.12$,$y(0)=0.1$,$z(0)=0.15$,通过 IEEE-754 标准转化为 32 位单精度浮点数后,给 Module_Lorenz 模块中的变量重新赋值,编译文件,重复上述烧录工作,调整示波器可得如图 6 所示 Lorenz 系统相轨图和时序图.

对比图 1 中由 MATLAB 仿真出的数值拓扑图,可见 Lorenz 系统经过 Euler 算法离散化后,利用 FPGA 开发板,可以获得类似 MATLAB 仿真图的效果. 但是在参数相同情况下,图 6 中的吸引子没有图 1 上的吸引子光滑,这是由于 Euler 算法比较简单,精确度不是很高所致.

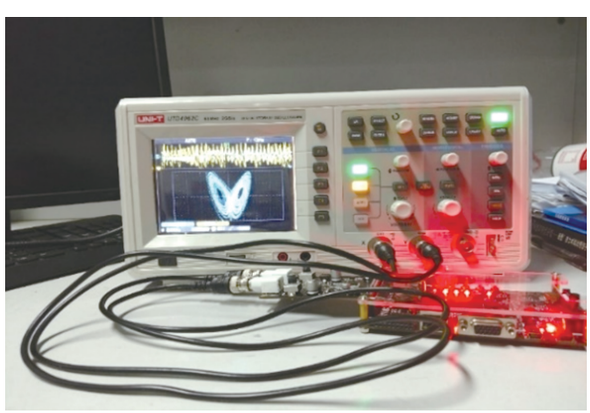
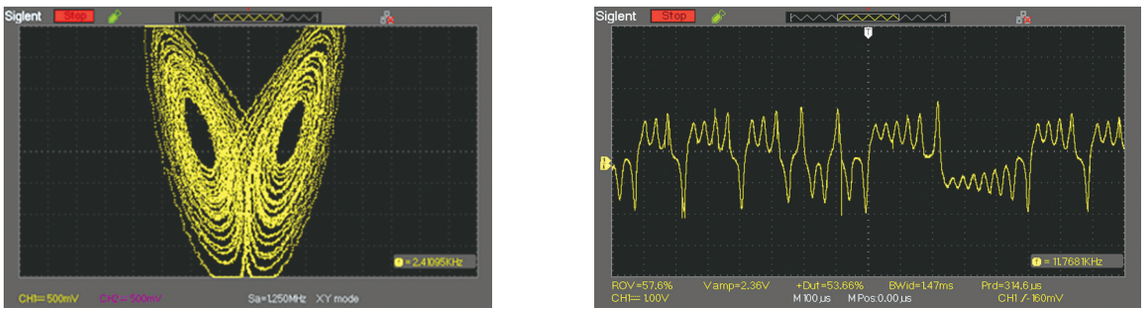


图 5　实物连接仿真图

Fig. 5　Physical connection simulation diagram



(a) x–z相图　　(b) x时序图

(c) x–y相图　　(d) y–z相图

图 6　Euler 法 FPGA 仿真所得相轨图和时序图

Fig. 6　Phase and time sequence diagram of the result of Euler method FPGA simulation

4　利用 Runge-Kutta 算法分析

不同于 Euler 算法最原始的显式单步法,即利用普通的差商运算来逼近求导数值对系统方程进行求解,该方法简单取切线端点作为下一步的起点进行计算,当迭代次数增加时,误差会因为累积而越来越大. Runge-Kutta 法采用稍复杂的多步骤运算方法,借助中值定理思想和泰勒级数展开法,在一个步长区间(x_i,x_{i+1})内,多取几个点,计算其斜率的大小,并进行加权、平均,作为导数的近似,经过多次迭代求解系统方程的解. 其中,根据单个步长内所取点数的不同,可以将 Runge-Kutta 法划分为一阶、二阶、三阶 Runge-Kutta 法等.

4.1　方程离散化

类似于前面的操作流程,首先,需将系统方程进行离散化处理. 本文将以二阶、四阶 Runge-Kutta 法为例,分别将 Lorenz 系统方程进行离散化,可得:

(1)二阶离散化方程

$$\begin{cases} K11=-a(x_i-y_i), \\ K21=cx_i-x_iz_i/E-y_i, \\ K31=x_iy_i/E-bz_i. \end{cases} \tag{3}$$

$$\begin{cases} K12=-a((x_i+T*K11)-(y_i+T*K21)), \\ K22=c(x_i+T*K11)-(x_i+T*K11)(z_i+T*K31)/E-(y_i+T*K21), \\ K32=(x_i+T*K11)(y_i+T*K21)/E-b(z_i+T*K31). \end{cases} \tag{4}$$

$$\begin{cases} x_{i+1}=T*(K11+K12)/2, \\ y_{i+1}=T*(K21+K22)/2, \\ z_{i+1}=T*(K31+K32)/2. \end{cases} \tag{5}$$

(2)四阶离散化方程

$$\begin{cases} K11=-a(x_i-y_i), \\ K21=cx_i-x_iz_i/E-y_i, \\ K31=x_iy_i/E-bz_i. \end{cases} \tag{6}$$

$$\begin{cases} K1n=-a\left(\left(x_i+\dfrac{T}{2}*K1(n-1)\right)-\left(y_i+\dfrac{T}{2}*K2(n-1)\right)\right), \\ K2n=c\left(x_i+\dfrac{T}{2}*K1(n-1)\right)-\left(x_i+\dfrac{T}{2}*K1(n-1)\right)\left(z_i+\dfrac{T}{2}*K3(n-1)\right)\Big/E-\left(y_i+\dfrac{T}{2}*K2(n-1)\right), \\ K3n=\left(x_i+\dfrac{T}{2}*K1(n-1)\right)\left(y_i+\dfrac{T}{2}*K2(n-1)\right)/E-b\left(z_i+\dfrac{T}{2}*K3(n-1)\right). \end{cases} \tag{7}$$

式中,$n=2,3$.

$$\begin{cases} K14=-a((x_i+T*K13)-(y_i+T*K23)), \\ K24=c(x_i+T*K13)-(x_i+T*K13)(z_i+T*K33)/E-(y_i+T*K23), \\ K34=(x_i+T*K13)(y_i+T*K23)/E-b(z_i+T*K33). \end{cases} \tag{8}$$

$$\begin{cases} x_{i+1}=x_i+T*(K11+2*K12+2*K13+K14)/6, \\ y_{i+1}=y_i+T*(K21+2*K22+2*K23+K24)/6, \\ z_{i+1}=z_i+T*(K31+2*K32+2*K33+K34)/6. \end{cases} \tag{9}$$

4.2　程序的设计与仿真

Runge-Kutta 离散化算法需要的运算量较大,如果在一个模块实现会导致状态过多,实现其他系统时修改不易. 因此分为四个可以实现不同功能的模块,分别为顶层模块,Module_*K*1_*K*2 模块,Module_XB_Re 模块和 Module_change 模块. 二阶 Runge-Kutta 与四阶 Runge-Kutta 法在求取 *K* 值的方程式是一样的. 因此,二阶与四阶的整体模块连接结构相同,底层 3 个模块通过顶层模块进行联系,顶层模块控制 Module_*K*1_ *K*2,Module_XB_Re 以及 Module_change 运行,具体连接方式如图 7 所示. 4 个模块中 Module_*K*1_*K*2 模块、Module_change 模块相同,变化较大的是 Module_XB_Re 模块和顶层模块,以下将对顶层模块和

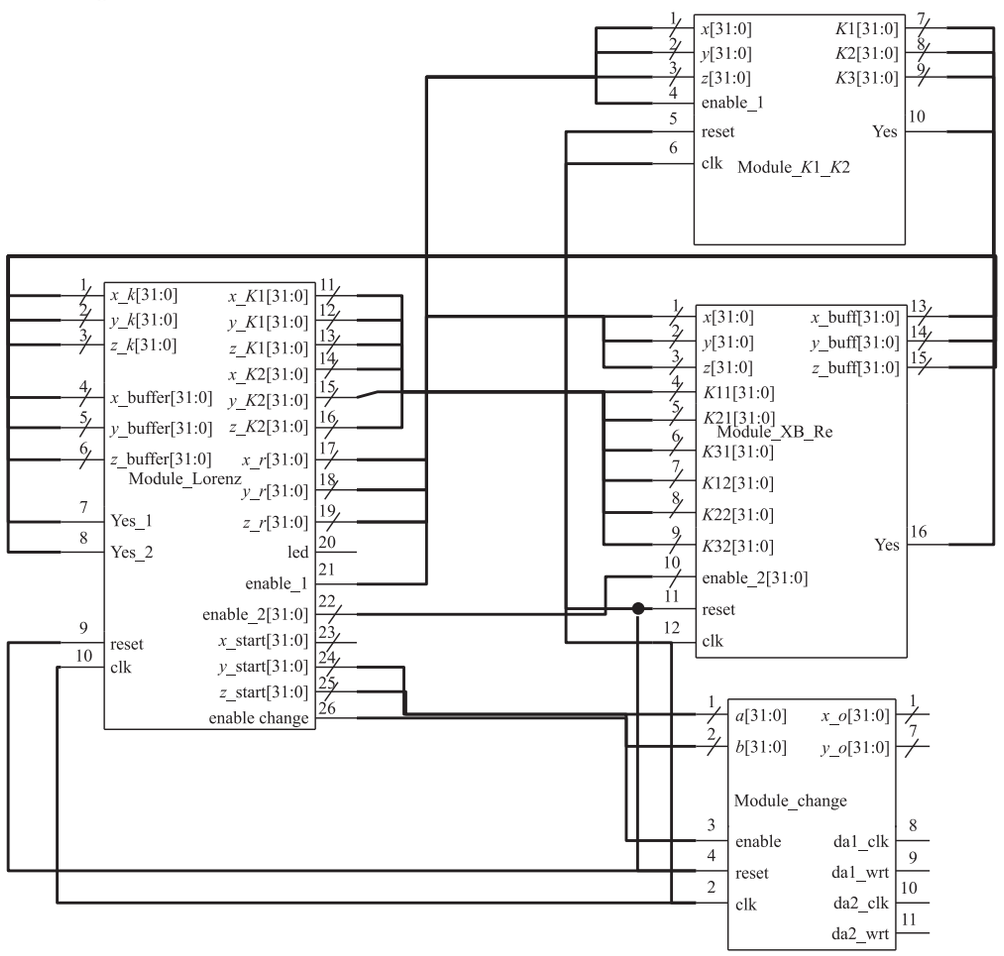


图 7　Runge-Kutta 法模块整体连接图

Fig. 7　Overall module connection diagram of Runge-Kutta method

Module_XB_Re 模块的流程图作具体分析.

(1)顶层模块调用和支配其他 3 个模块,主要作用是控制计算的顺序. 将二阶顶层模块命名为 Module_Lorenz,四阶顶层模块名称为 Module_Lorenz_4_R_K,程序编写综合出的硬件有 10 个输入,16 个输出,与其他两个模块连接,还有 x_start,y_start,z_start 输出方便与 DA 转换模块连接.

①二阶顶层模块计算流程为:首先给 x_start,y_start,z_start 状态赋初值,调用 Module_K1_K2 模块求取 K1,再执行 Module_XB_Re 模块获得求取 K2 的中间值 x_buff,y_buff,z_buff,然后再次调用 Module_XB_Re 模块,利用中间值求得 K2,最后再次执行 Module_XB_Re 模块获得新的 x_start,y_start,z_start,依次循环. 其中新的 x_start,y_start,z_start 将通过调用 Module_change 模块 DA 转换输出. 其具体计算流程图如图 8 所示.

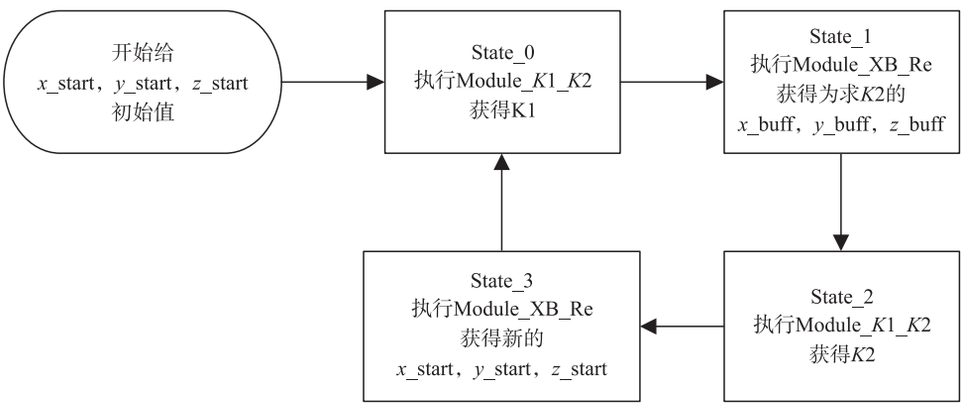


图 8　二阶顶层模块流程图

Fig. 8　Flow chart of the second-order top-level module

②四阶顶层模块综合成的电路与二阶类似,输入输出只需要在二阶的基础上把各相的 K1,K2 改成 K1,K2,K3,K4. 该模块计算流程图相较于二阶 Runge-Kutta 法多了求解 K2、K3 的过程,其过程与求解 K1、K2 过程相同. 由图 9 模块流程图可以看出,该模块一共有 9 个状态,每个状态对应着调用一个底层模块的运行,程序运行一个周期时长对比二阶 Runge-Kutta 法的 5 个状态大很多.

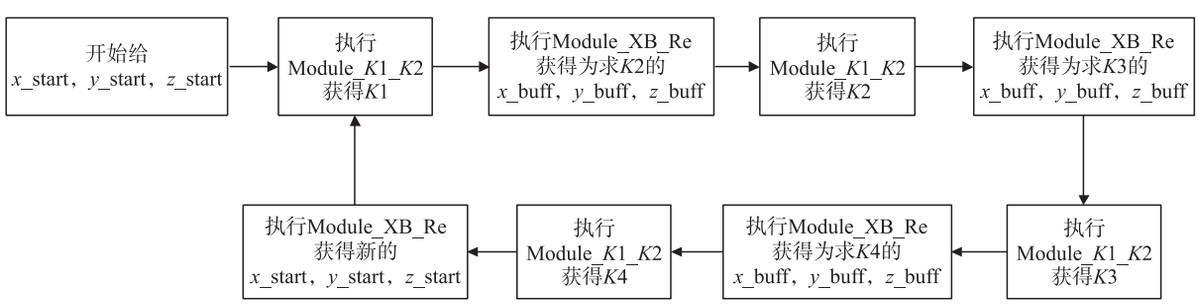


图 9　四阶顶层模块流程图

Fig. 9　Flow chart of the fourth-order top-level module

(2)Module_XB_Re 模块主要作用是计算新的 x,y,z 结果值(x_{i+1},y_{i+1},z_{i+1}),和求取为计算 K2 的中间值(x_buffer,y_buffer,z_buffer). 这两个计算过程并不一样,但是所用的乘法器与加法器数量相同,且数量只与方程组的微分方程数有关.

①二阶 Module_XB_Re 模块运行的时候有一个判断过程,enable = 2′b01 时执行计算 K2 中间值的过程,enable = 2′b10 时执行计算 x,y,z 值的过程,enable = 2′b00 时停止计算,所有输出为 0. 其中二阶的 Module_XB_Re 模块流程图为图 10.

②四阶比二阶多了两个 K 值计算,计算每个 K 用不同的步长,计算 x,y,z 结果时每个 K 值有不同的权重,因此四阶的 Module_XB_Re 模块变化较大,模块中判断的状态也有所增加,enable = 4′b0001 时计算为求 K2 的中间值,enable = 4′b0010 时计算为求 K3 的中间值,enable = 4′b0100 时计算为求 K4 的中间值,enable = 4′b1000 时计算新的 x,y,z 值. 绘制四阶 Module_XB_Re 流程图如图 11.

(3)Module_*K*1_*K*2 模块主要作用是计算 *K*1,*K*2,二阶和四阶 Runge-Kutta 法计算过程相同,一共 5 个状态,迭代计算较为简单,能用一个模块实现,只是在顶层模块中输入不同的值就可以获得 *K*1,*K*2.

(4)Module_change 模块与 Euler 法使用的模块相同,即将模块最终的运算数据进行 DA 转换输出,具体工作流程不再介绍.

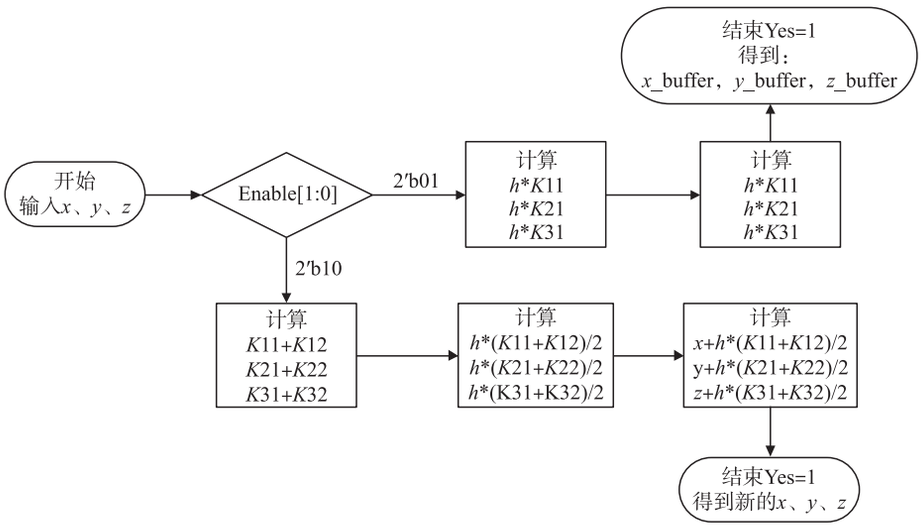


图 10　二阶 Module_XB_Re 模块流程图

Fig. 10　Flow chart of the second-order Module_XB_Rel module

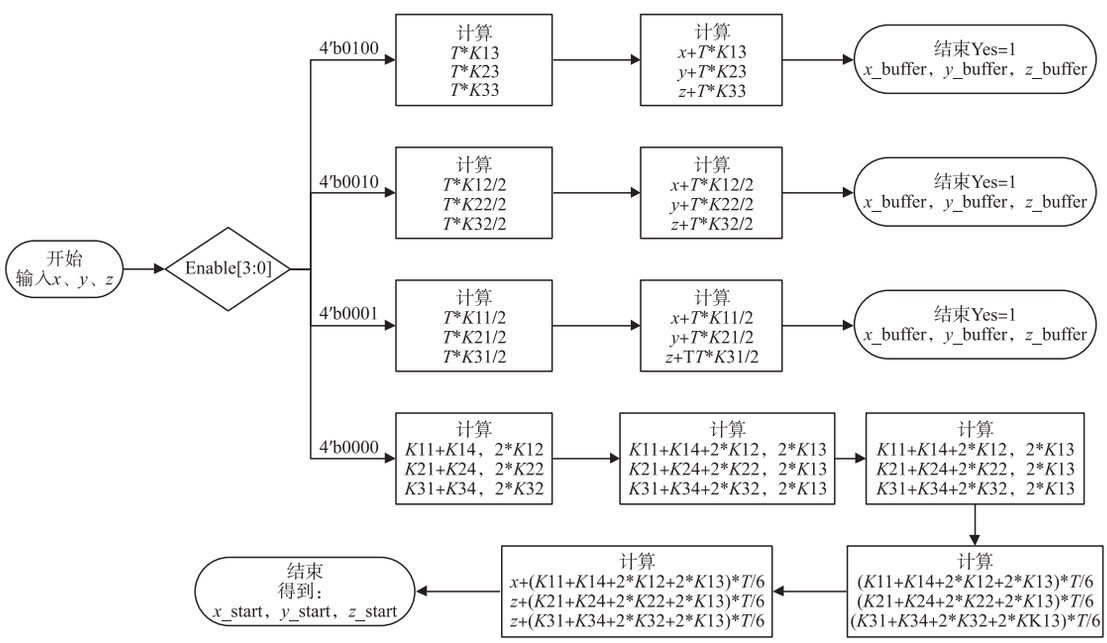
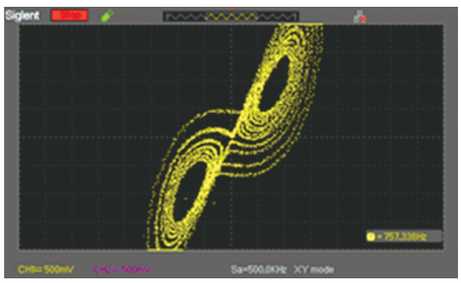


图 11　四阶 Module_XB_Re 模块流程图

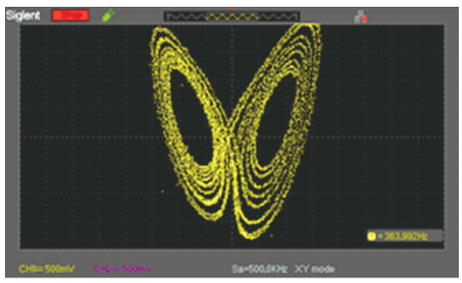
Fig. 11　Flow chart of the fourth-order Module_XB_Re module

4.3　实验结果

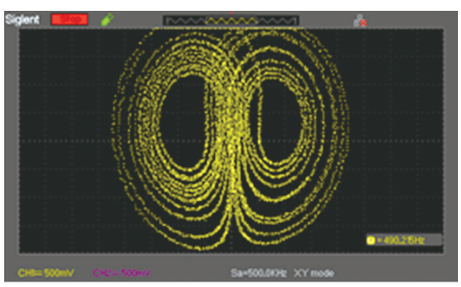
重复上一章节的程序调试工作,将布局布线生成 bit 文件烧录到 FPGA 开发板用示波器进行观察. 取参数 $a=10$,$b=8/3$,$c=28$,$x(0)=0.12$,$y(0)=0.1$,$z(0)=0.15$,$E=0.1$,步长 $T=0.002$,可得 Lorenz 系统相轨图如图 12 所示. 其中,图 12(a)~图 12(c)为二阶 Runge-Kutta 法实现,图 12(d)~图 12(f)为四阶法实现. 对比可以发现由四阶 Runge-Kutta 法实现的图像点之间的间隔较小,并且色彩较深,原因在于四阶离散使多个点在一个区域内显示. 运用二阶 Runge-Kutta 法离散时候,在单位时间计算的点数相同情况下,点与点之间跨距减少,这使得计算过程未能多次遍历整个混沌区域,可能只遍历了一次,因此显得图像比较稀疏. 但是相较于 Euler 法而言,可以看到图像轨道分明,毛刺较少,显得较为光滑,混沌特性还原度较高.



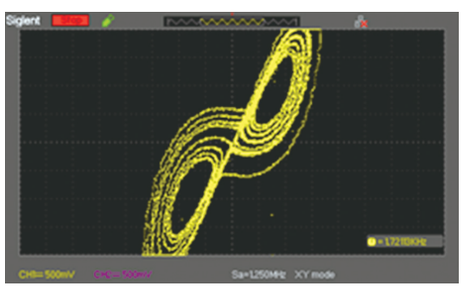
(a) x–y相图，二阶



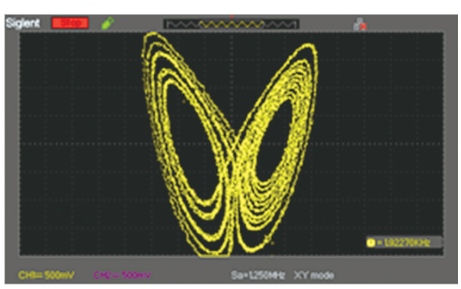
(b) x–z相图，二阶



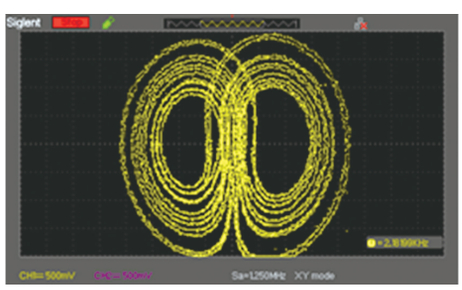
(c) y–z相图，二阶



(d) x–y相图，四阶



(e) x–z相图，四阶



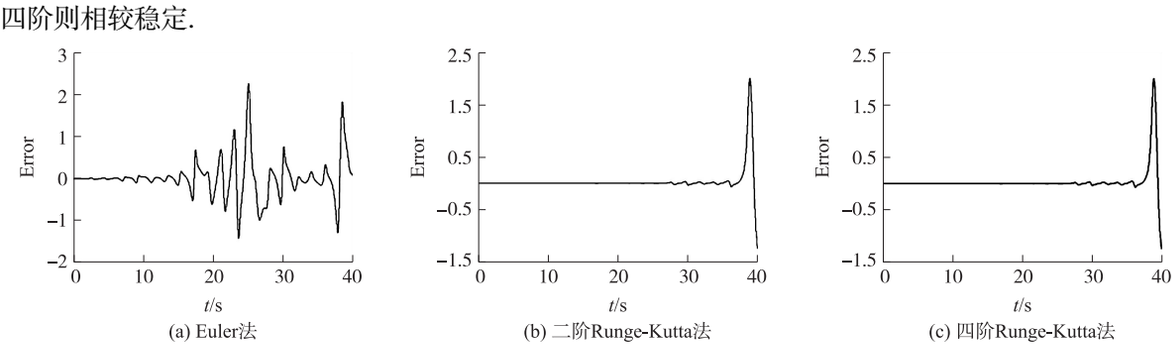
(f) y–z相图，四阶

图 12　Runge-Kutta 法 FPGA 仿真所得相轨图

Fig. 12　Phase diagram of the result of Runge-Kutta method FPGA simulation

4.4　误差分析与算法比较

将 FPGA 仿真中 3 种离散方案所得的 x 时序图数据通过 MATLAB 转换,并将转换后的数据分别与原 MATLAB 仿真数据进行误差拟合,可以得出 3 种离散方案的误差分析曲线如图 13 所示. 由图可见采用 Euler 法时的系统误差比采用 Runge-Kutta 法更为明显,且稳定时间较短,仅有 15 s 左右,随着程序迭代,15 s 之后误差便会不断地累积放大,波动也更加明显. 由图 12(b)、图 12(c)可知,相比较 Euler 法,Runge-Kutta 法的稳定性要好很多,精度也更高,二阶 Runge-Kutta 法在 22 s 后误差逐渐增大,而四阶算法大约在 25 s 出现误差. 这两种 Runge-Kutta 法之间的误差分析曲线差别不大,但二阶误差曲线在 17 s 和 21 s 时有微小误差存在,而四阶则相较稳定.



(a) Euler法　(b) 二阶Runge-Kutta法　(c) 四阶Runge-Kutta法

图 13　FPGA 与 MATLAB 仿真输出数据 x 的误差曲线

Fig. 13　Error curve of FPGA and MATLAB output data x

判断一个算法实现的优劣不但要依靠数据的误差度,还需要考虑 FPGA 实现时开发板的资源使用情况,资源调用少,算法运行简易,运行速率快. 反之,程序运行冗长,易出错,消耗资源,在有效时间内示波器不易

显示完整波形图. ISE 提供的浮点乘法器及浮点加法器,FPGA 中内置 32 个 DSP48A1,一个乘法器使用完整的 4 个 DSP 加上 100 个 LUT,或者直接使用 648 个 LUT. 本次系统 3 种算法实现及 DA 转换模块占用的 FPGA 资源如表 1 所示.

表 1　3 种算法 FPGA 资源使用情况对比表

Table 1　Comparison table of FPGA resource usage of three algorithms

资源占用率	算法		
	Euler	二阶	四阶
寄存器模块	7%	19%	23%
LUT 单元	35%	73%	82%
逻辑单元	41%	82%	82%
未使用的触发器(Flip FlopH)	59%	55%	51%
成套 LUT-FF 单元	32%	41%	42%
输入输出单元(IOBs)	15%	15%	15%
DSP48A1s 模块	75%	93%	100%

由表 1 可知,在 FPGA 内部资源调动方面,Euler 算法一共用到了 4 个乘法器和 2 个加法器,LUT 占用 35%,可见 Euler 算法占用的系统资源较少,二阶 Runge-Kutta 算法一共用到了 6 个乘法器和 5 个加法器,LUT 占用率达到 73%,而使用四阶算法时 LUT 的占用率达到 82%. 无论采用哪种 Runge-Kutta 算法进行系统实现时,系统资源均已占用大半资源.

根据综合实现效果和资源调用情况分析可知,Runge-Kutta 法的精确度整体优于 Euler 法,在 Runge-Kutta 法中四阶精确度稍稍优于二阶. 但是,四阶 Runge-Kutta 对开发板资源的要求最大,故而在实现经典混沌系统方面,二阶 Runge-Kutta 法是最优选择.

5　结语

本文主要基于 FPGA 数字技术,利用 Verilog HDL 语言编写程序,进行调试与仿真. 讨论了 Euler 法、二阶和四阶 Runge-Kutta 法 3 种离散算法. 结合这 3 种离散算法,利用直接编写 Verilog HDL 语言程序的方式对经典 Lorenz 混沌系统进行 FPGA 仿真实现,得出在兼顾仿真精确度和开发板资源占用率这两项指标情况下,二阶 Runge-Kutta 法为非线性系统方程最优离散方法的结论.

[参考文献](References)

[1] PRICE D C,KOCZ J,BAILES M,et al. Introduction to the special issue on digital signal processing in radio astronomy[J]. Journal of astronomical instrumentation,2017,5(4):1602002.

[2] MILIK A. On hardware synthesis and implementation of PLC programs in FPGAs[J]. Microprocessors and microsystems,2016,44(C):2-16.

[3] 张钰. 基于 FPGA 技术的多涡卷混沌信号发生器的研究[D]. 广州:广东工业大学,2007.
ZHANG Y. The research on multi-scroll chaotic signal generator based on FPGA technology[D]. Guangzhou:Guangdong University of Technology,2007.(in Chinese)

[4] 王忠林,王光义. 基于 FPGA 的混沌系统设计与实现[J]. 计算机工程与设计,2009,30(14):3365-3366.
WANG Z L,WANG G Y. Design and implementation of chaotic system based on FPGA[J]. Computer engineering and design,2009,30(14):3365-3366.(in Chinese)

[5] 刘师彤. 混沌同步保密通信的研究及 DSP-Builder 实现[D]. 长春:长春理工大学,2010.
LIU S T. Secure communication research and DSP-Builder implementation based on chaos synchromization theory[D]. Changchun:Changchun University of Science and Technology,2010.(in Chinese)

[6] 袁泽世,李洪涛,朱晓华. 类 Chen 系统设计及其 FPGA 实现[J]. 南京理工大学学报(自然科学版),2015,39(3):323-329.
YUAN Z S,LI H T,ZHU X H. Design of Chen-like system and its FPGA implementation[J]. Journal of Nanjing university of science and technology(natural science edition),2015,39(3):323-329.(in Chinese)

[7] 李登辉,闵富红,马美玲. 基于 FPGA 技术的混沌系统设计与实现[J]. 南京师范大学学报(工程技术版),2015,15(1):8-14.

LI D H,MIN F H,MA M L. The design and implementation of chaotic system based on FPGA technology[J]. Journal of Nanjing normal university(engineering and technology edition),2015,15(1):8-14.(in Chinese)

[8] 王忠林,刘树堂. 一个切换 Lorenz 混沌系统的特性分析[J]. 重庆邮电大学学报(自然科学版),2017,29(1):68-74.
WANG Z L,LIU S T. Analysis of properties of a switched Lorenz type chaotic system[J]. Journal of Chongqing university of posts and telecommunications(natural science edition),2017,29(1):68-74.(in Chinese)

[9] GAO T G,CHEN G R,CHEN G R,et al. The generation and circuit implementation of a new hyper-chaos based upon Lorenz system[J]. Physics letters A,2016,361(1):78-86.(in Chinese)

[10] 张昊. 基于不同耦合的复杂系统同步研究[D]. 大连:大连理工大学,2016.
ZHANG H. Synchronization studies of complex system with different couplings[D]. Dalian:Dalian University of Technology,2016.(in Chinese)

[11] NI J K,LIU L,LIU C X,et al. Fractional order fixed-time nonsingular terminal sliding mode synchronization and control of fractional order chaotic systems[J]. Nonlinear dynamics,2017,89(3):2065-2083.

[12] 肖锋,张丽丽,冯飞. 混合混沌系统的并行多通道彩色图像加密[J]. 微电子学与计算机,2016,33(8):76-81.
XIAO F,ZHANG L L,FENG F. Parallel multi-channel color image encryption Based on hybrid chaotic system[J]. Microelectronics & computer,2016,33(8):76-81.(in Chinese)

[13] 张民,李宗浩,高明. 基于 Lorenz 混沌的 MIMO 雷达信号设计及性能分析[J]. 系统工程与电子技术,2017,40(1):58-64.
ZHANG M,LI Z H,GAO M. Design and performance of MIMO radar signal based on Lorenz chaos[J]. Systems engineering and electronics,2017,40(1):58-64.(in Chinese)

[14] 禹思敏. 混沌系统与混沌电路:原理、设计及其在通信中的应用[M]. 西安:西安电子科技大学出版社,2011.
YU S M. Chaotic systems and chaotic circuits:principle design and its application in communications[M]. Xi'an:Xidian University Press,2011.(in Chinese)

[责任编辑:陈　庆]