

基于 SystemC 的可配置 FFT 周期精确模型

樊恩辰, 姚 馨, 何书专, 潘红兵

(南京大学 电子科学与工程学院, 江苏省光电信息功能材料重点实验室, 江苏 南京 210093)

摘 要: 基于 SystemC 实现了一款可配置 FFT(Fast Fourier Transformation)硬件加速器的周期精确模型. 该模型采用以基-8 为主的混合基、4 路并行和基于存储器结构的设计方法, 可以计算 16-4M 点单精度浮点复数 FFT, 并将信噪比稳定在 130 dB 以上. 与同类型的模型相比较, 它不仅可以实现与硬件周期的精确匹配, 同时可以保证远快于硬件的仿真速度, 达到了配合硬件设计以及验证的效果.

关键词: FFT 建模; SystemC; 混合基 FFT 算法

中图分类号: TN46

文献标识码: A

文章编号: 1000-7180(2014)11-0083-05

Cycle-Accurate Configurable FFT Modeling with SystemC

FAN En-chen, YAO Xin, HE Shu-zhuan, PAN Hong-bin

(College of Electronics Science and Engineering, Nanjing University, Jiangsu Provincial Key Laboratory of Advanced Photoic and Bectronic Materials, Nanjing 210093, China)

Abstract: This article is implements a configurable FFT (Fast Fourier Transformation) hardware accelerators cycle accurate model based on SystemC . The mixed-Radix8 FFT, 4 parallel computation and memory architecture-based design are used in this model. The model can calculate 16-4 m point single-precision floating-point complex FFT, the signal-to-noise ratio at the same time stable at more than 130 db. Compared with the model of same type, It can achieve an exact match with the hardware cycle, and guarantee much faster speed than hardware simulation, which achieves the effect of cooperating with hardware design and verification.

Key words: FFT modeling; SystemC; mixed radix FFT algorithm

1 引言

随着芯片设计复杂度的不断提高,传统直接使用硬件描述语言进行 RTL 级设计的方法就显得工作繁重并且耗时. 如果在早期建立芯片的高级模型,不但可以验证设计的正确性、分析芯片的各项性能指标,还可以提速与硬件配套的软件的开发. 如此就可以大大缩短芯片的研发周期,减少芯片的设计成本,实现软件与硬件的协同设计^[1].

SystemC 是一种广泛应用的硬件建模语言,它提供了从算法到 RTL 级的一整套解决方案^[2]. Sys-

temC 在 C++ 的基础上添加了并行性、时钟、端口、信号等概念以及相应的库,能够非常方便地对硬件进行建模^[3]. 因此本文选取 SystemC 作为建模语言.

2 FFT 算法建模

本文所介绍的 FFT 算法模块是可配置专用处理核模型的一部分. 本模型是基于 Synopsys 公司的 PA 平台的事务级模型,通过在功能模块中加入时序信息,实现周期精确级的建模. 它主要用于配合硬件模块的设计与调试,以及后期软件的开发^[3].

收稿日期: 2014-01-18; **修回日期:** 2014-03-07

基金项目: 国家自然科学基金项目(61176024, 61006018, 61370040, 61376075); 高等学校博士学科点专项科研基金(20120091110029); 江苏省产学研联合创新资金-前瞻性联合研究项目(BY2013072-05); 江苏高校优势学科建设工程资助项目

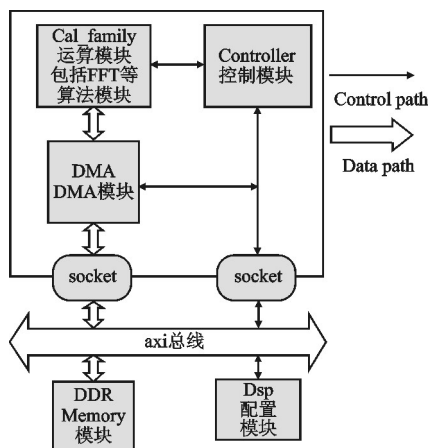


图 1 可配置处理核整体结构

2.1 总体框架

本文处理核的整体结构如图 1 所示,内部可以分为 3 个主要模块.其中,Cal_family 模块是主要的运算模块.除了 FFT,该运算模块还包括矩阵运算(加、减、乘、求逆等)和向量运算(FIR,自相关,互相关等).Controller 模块是处理核的控制模块,该模块的作用是与外界的控制信息进行交互,并且控制内部状态的转换. DMA 模块是主要的数据传输通道,用于接收从外界 DDR 搬运来的数据,并且将数据搬运到运算模块. DSP 配置模块的功能是使用 API 函数生成相应的配置信息(如表 1 所示).

表 1 FFT 配置参数表

位域	位域名称	位宽	描述
输入长度寄存器			
31:24	Reserved	8	保留
23:0	Input_Len1	24	点数 * 8
输出长度寄存器			
31:24	Reserved	8	保留
23:0	Output_Len	24	结果长度 * 8
参数寄存器			
31:27	Reserved	5	保留
26:3	Data_size	24	FFT 点数
2	Ifft	1	ifft 标识位
1:0	IO order	2	输入输出顺序

整个处理流程是:每次从 DSP 读取的配置信息和从 DDR 中搬运的数据都被封装为事务,通过端口(SC_PORT)与 AXI 总线进行数据交互,每次交互都由时钟信号触发.配置信息对控制模块中的配置寄存器进行配置,由控制模块对配置信息进行翻

译,传递给 DMA 模块和运算单元(DMA 模块负责配置数据的搬运方式).配置完成后,开始数据搬运.当配置和搬运都完成后,控制模块向运算模块发送 Start 信号,启动运算.运算完成之后,运算模块向控制模块发送 Finish 信号,由控制模块对 DMA 进行配置,由 DMA 把结果搬运到 DDR 相应的地址中,整个操作过程完成.

2.2 FFT 运算模块建模

FFT 模块是运算模块子模块之一,它采用频率抽取的基-8 FFT 算法、基-4 FFT 算法和基-2 FFT 算法^[4],同时根据配置点数的不同,灵活选取三种算法,最大可以实现 4M 点 FFT.整个模块使用 SRAM 进行存储,SRAM 一共分成 32 块(bank),每块大小是 8k word(1word=64 * 1 bit),片上可以存储 256k 单精度浮点复数数据.因此小点数运算模式可以计算 32-256k 点,大点数运算模式则计算 256k-4M 点 FFT,其中大点数 FFT 采用二维 FFT 算法,分多次搬运实现.整个 SRAM 总共有 8 个双向的读写端口,为了配合以上存储结构,模型采用了与硬件贴合的四路并行的设计.片上 SRAM 在模型被封装为模块,模块中用数组对其进行模拟.

2.2.1 算法描述

对于点数为 N 的复数序列 $x(n) \ n = 1, 2, \dots, N$,进行按照频率抽取的基-8FFT.由傅里叶变换定义得

$$X(k) = DFT[x(n)] = \sum_{n=0}^{N-1} x(n) W_N^{nk}$$

$$(k = 0, 1, 2, \dots, N-1)$$

利用 W_N^{nk} 的周期性和对称性,则上式化为

$$\begin{aligned} X(k) = & \sum_{m=0}^{N/8-1} x(8m) W_{N/8}^{nk} + W_N^{nk} \sum_{m=0}^{N/8-1} x(8m+1) W_{N/8}^k + \\ & W_N^{2k} \sum_{m=0}^{N/8-1} x(8m+2) W_{N/8}^{nk} + W_N^{3k} \sum_{m=0}^{N/8-1} x(8m+3) W_{N/8}^{nk} + \\ & W_N^{4k} \sum_{m=0}^{N/8-1} x(8m+4) W_{N/8}^{nk} + W_N^{5k} \sum_{m=0}^{N/8-1} x(8m+5) W_{N/8}^{nk} + \\ & W_N^{6k} \sum_{m=0}^{N/8-1} x(8m+6) W_{N/8}^{nk} + W_N^{7k} \sum_{m=0}^{N/8-1} x(8m+7) W_{N/8}^{nk} \\ & (m = 0, 1, 2, \dots, (N/8) - 1). \end{aligned}$$

由上式得出基-8 蝶形单元每计算出 1 个结果需要 7(8-1)次累加,一共有八组这样的运算^[5].而基-2、基-4 的蝶形单元都可以看作是基-8 蝶形单元的一部分,如此,基-2、基-4 和基-8 的混合基 FFT 在硬件上就可以实现复用.同时考虑到速度因素,在设计中尽可能使用基-8FFT^[6].

2.2.2 程序流程

如果 FFT 的运算点数小于 128k, 数据可以一次性搬运到 SRAM 中完成运算. 从 SRAM 中读取数据前, 先判断运算模式(点数 $N = A \times 8^n$). 如果 $A = 2$, 则首先要进行一级基 2 运算; 如果 $A = 4$, 则先进行一级基 4 运算; 如果 $A = 8$, 则直接进行基-8 运算. 整个运算过程需要完成 n 级基-8 运算. 基-2、基-4 和基-8 运算的蝶形运算单元都封装成相应的子函数形式, 再根据情况调用相关的运算.

对于点数大于 128k 的情况, 将采用二维 FFT 的计算方法^[7]. 以 1M 点 FFT 为例, 在进行第一轮运算时, 将数据按照列(搬数间隔为 256(总点数/4096))的顺序搬运到运算模块中, 每次搬运 128k, 运算完毕之后搬运回 DDR 中, 一共搬运 8(总点数/128k)次; 第二轮运算时, 将数据按照行(搬数间隔为 4096(固定))的顺序搬运到运算模块中, 每次搬运 128k 数据到运算模块中进行运算(该轮运算需要乘以中间旋转因子), 运算完毕之后再搬运回 DDR 中. 第一轮运算将 128k 点分成 512(128k/256)组的 256(总点数/4096)点 FFT, 第二轮运算将 128k 点分成 32(128k/4096)组的 4096 点(固定)FFT. 搬运时采用乒乓的搬运方式, 运算与搬运同时进行, 奇数次搬运使用 bank0 ~ bank15, 偶数次搬运使用 bank16 ~ bank31.

2.2.3 模型各模块描述

算法模型的顶层模块是时序模块 FFT. 顶层模块的作用包括: 控制函数与存储器之间的数据交换; 乘法函数(蝶形运算单元的结果与旋转因子相乘)、基-2、基-4、基-8 蝶形运算函数的调用; 时序的控制(调用运算 IP 产生的延时). 模型主要包括蝶形运算、乘法运算、旋转因子产生、地址生成等子函数(整体结构互联如图 2 所示), 它们都是组合逻辑模块, 当输入发生变化时输出立刻变化, 同时将延时记录下来, 在顶层模块中对其进行延时.

FFT 模块包含有端口、信号、子模块、进程及构造函数等 SystemC 的基本元素. 通过在该模块中添加 sc_clock 类和 sc_method 进程, 完成时序的控制. FFT 模块(包括模型中的其他算法模块和控制模块)敏感列表仅为时钟, 只有在时钟上升沿到来时, 对时钟敏感的内部信号和输出才会更新. 模块之间的互联采用 sc_in 和 sc_out 端口. 模块内部使用 sc_signal 信号, 它在赋值-更新之间有一个很小的延迟, 这样就在仿真中实现了赋值和更新的分离. 设计中比较重要的包括以下三部分:

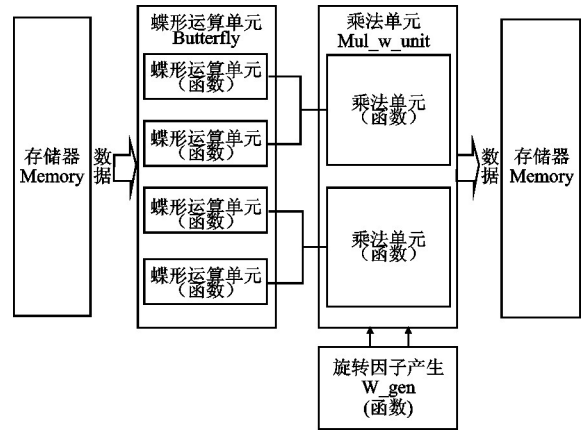


图 2 FFT 模型整体函数结构图

(1) 蝶形运算单元

由于 SRAM 模块有 8 个输入输出双向端口, 而在每一级运算时需要流水进数和出数, 蝶形运算单元采用了四路并行的设计(图 3 所示为第一路结构). 基-8 运算中一共有 8 组运算, 每一组运算都是 7(8-1)次乘累加. 四路并行中, 1、2 路从第 1 组开始运算, 这样一次可以算出一组中的两个数, 然后进行累加, 第一组经过 1、2 路分别运算四次依次累加得出第一组的结果; 3、4 路相似的从第 5 组开始运算, 分别运算四次依次累加后得出第 5 组的结果. 得出结果后, 再乘以相应的旋转因子, 最后存入存储器原位置. 由于基-2 运算、基-4 运算的蝶形运算单元可以看成是基-8 蝶形运算单元的一部分, 因此可以复用基-8 的运算单元, 仅仅在抽取的间隔上做出调整即可.

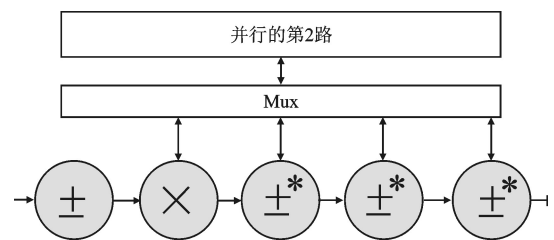


图 3 基-8 蝶形运算单元示意图(第 1 路)

蝶形运算单元包含两个子函数 Butterfly1 和 Butterfly2. 其中, Butterfly2 用来描述蝶形运算单元中的乘法运算, Butterfly1 则负责累加 Butterfly2 中的结果. 所有涉及到的运算(实数加、实数乘、复数加和复数乘)都加入了与硬件的运算 IP 相应的延时. Butterfly1 和 Butterfly2 都是组合逻辑函数, Butterfly2 封装在 Butterfly1 内.

(2) 旋转因子产生单元

旋转因子产生的模块 W_gen, 首先预存了 W_{4k}^0

$\sim W_{4k}^{1k-1}$ 、 $W_{4M}^0 \sim W_{4M}^{1k-1}$ 2 组总共 $2k$ 旋转因子(为以后大点数的设计留出余量),根据三角函数的对称性可以推算出 $W_{4k}^0 \sim W_{4k}^{4k-1}$ (初值 1), $W_{4M}^0 \sim W_{4M}^{4k-1}$ (初值 2). 这样,用插值的方法(公式如下)就可以推算出 $4M$ 点的旋转因子^[8].

$$W_{4M}^{k_1} = W_{4M}^{k_2} \times W_{4M}^{k_3}$$

$$k_1 = 0, 1, 2, 3, \dots, 4M-1;$$

$$k_2 = 0, 1, 2, 3, \dots, 4k;$$

$$k_3 = 0k, 1k, 2k, 3k, \dots, 4M;$$

旋转因子由 DMA 搬入常数 Memory (常数 Memory = 8bank * 1k), 前 1k 点初值 1 顺序放置在 bank0 中, bank2、bank4、bank6 与 bank0 中放置相同数据. 后 1k 点初值 2 顺序放置在常数 Memory 的 bank1 中, bank3、bank5、bank7 与 bank1 中放置相同数据, 这样就可以并行的产生旋转因子.

(3) 地址产生单元

地址产生单元在每次搬运数据时先计算相应的序列号, 再根据序列号和地址的映射关系从相应的地址搬运数据. 从 DMA 搬入数据时, 128k 以下用到 0~15bank (Memory = 32bank * 8k), 128k 以上是大点数, 采用乒乓的搬运方式, 每次搬运 128k 数据, 奇数次搬运使用 0~15bank, 偶数次用到 16~32bank. 由于运算结果放回 Memory 原位置, 因此输出顺序与输入顺序相同. 以小点数为例, 如果是 128k 点则需要 17 位地址 (4 位表示 bank 号, 13 位表示 bank 深度), 序号与 bank 以及 bank 深度的映射关系如下:

$$\text{bank}[2:0] = \text{index}[2:0]$$

$$\text{bank}[3] = \text{index}[3] \oplus \text{index}[4] \dots \oplus \text{index}[16]$$

$$\text{addr}[12:0] = \text{index}[16:4]$$

以上映射方式可以保证在 FFT 运算时不会发生地址的冲突, 即并行的四路每次都从不同的 bank 中取出数据.

3 FFT 模型的测试与结果分析

本模型使用 Synopsys 公司开发的 PA 平台中的 SystemC IDE 进行仿真和测试, 其中全部计算使用单精度浮点复数. 相应硬件设计同样全部使用单精度浮点复数, 其中 128k 以下小点数使用 VCS 进行测试, 其余的大点数使用 FPGA 进行测试 (Xilinx Virtex-6 FPGA). 表 2 中信噪比计算中使用的信号功率是在 Matlab R2012b 中使用相同输入数据计算得到的.

由测试结果可以看出, 信噪比随着点数的增加

不断下降. 与基-2 和基-4 运算相比较, 基-8 运算对信噪比的影响稍大. 这主要是由于基-8 运算相乘的系数中有很多, 如此就增加了误差. 软件与硬件测试出的信噪比十分相近, 并且有着相同的变化趋势, 模型达到了在硬件设计以前对性能指标进行验证的效果.

表 2 两种平台测试的信噪比

点数	SystemC IDE 测试数据	VCS 测试数据
256	135.2	135
512	135	134.7
1024	134.8	134.6
4096	134.1	134
16k	133.3	133.3
32k	133.1	133.3
64k	134	134
128k	132.5	132.6
256k	132.7	132.6
512k	132.7	132.5

从表 3 可以看出, VCS 由于需要仿真硬件电路以及波形需要比较长的时间, 而 SystemC 模型在仿真时间上有着非常大的优势. 表 4 所示为两种平台测试的周期数.

表 3 两种平台测试的仿真时间

点数	SystemC IDE 仿真时间/s	VCS 仿真时间/s
2048	5.51	19.24
64k	7.73	234.11
128k	11.44	1036
512k	60.13	4306.34

表 4 两种平台测试的周期数
(从 Start 信号到 Finish 信号)

点数	SystemC IDE 仿真周期数	VCS/FPGA 仿真周期数
4 096	8 710	8 710
16k	41 606	41 606
128k	393 990	393 990
512k	4 014 657	4 543 213
1024k	5 917 818	6 706 563

128k 以下小点数模型和硬件的仿真周期相同, 而大点数周期相差较大. 分析原因是: 大点数 FFT 使用乒乓搬运 (如图 4 所示), 运算间隔的等待时间与 DDR 搬数的周期数有关, 而在实际测试中, DDR 读取和写入的时间并不确定. 如果 DDR 搬运时间或响应时间较长, 则搬运时间完全可以掩盖掉运算

时间.为此抓取了模型和硬件实际的运算周期数(如表5所示),由上表格可以看出,两者的运算周期数完全相同.



图4 乒乓运算示意图

表5 大点数纯运算周期数

点数	SystemC IDE 仿真周期数	FPGA 仿真周期数
512k	3 490 304	3 490 304
1024k	5 717 818	5 717 818

4 结束语

针对在 Soc 设计中实现软硬件协同的要求,本文提出了一种可配置的周期精确的 FFT 模型.通过与硬件测试结果的比对,模型实现了小点数周期精确,达到了完善硬件设计、验证各项性能指标、缩短硬件研发时间的目的.

参考文献:

- [1] Ali Ahmadinia, Balal Ahmad, Ahmet Erdogan, et al. System C-based reconfigurable IP modelling for system-on-chip design[C]// IEEE NASA/ESA Conference on Adaptive Hardware and Systems. Piscataway, NJ: Adaptive Hardware and Systems, 2008:362-367.
- [2] 何卫强,杨靓,卢强.基于 SystemC 的周期精确级 DSP

处理器建模[J].微电子学与计算机,2012,29(4):107-110.

- [3] SystemC Version 2.0 User's Guide [EB/OL]. [2012-01-13]. Inc. <http://www.systemc.org>.
- [4] Bidet E, Castelain D, Joanblanc C, et al. A fast single-chip implementation of 8192 complex point FFT[J]. IEEE Journal of Solid-State Circuits, 1995, 30(3): 300-304.
- [5] Hongjiang He, Hui Gu. The realization of FFT algorithm based on FPGA Co-Processor[C]// Second International Symposium on Intelligent Information Technology Application, Beijing, 2008: 239-243
- [6] Lihong Jia, Yonghong Gao, Hannu Tenhunen. Efficient VLSI implementation of radix-8 FFT algorithm [C]// IEEE Pacific Rim Conference on Communications. Piscataway, NJ: Computers and Signal Processing, 1999: 468-471.
- [7] 邓波,戎蒙恬,汤晓峰.可配置高速高精度 FFT 的硬件实现[J].计算机工程,2006(17):254-256.
- [8] Lu Shyuekung, Yeh Chienhung, Lin Hanwen. Efficient built-in self-test techniques for memory-based FFT processors [C]// Proceedings of the 10th IEEE Pacific Rim International Symposium on Dependable Computing. Piscataway, NJ: Computer and Signal Processing, 2004: 468-471.

作者简介:

樊恩辰 男,(1990-),硕士研究生.研究方向为 ASIC 设计、多核处理器.

(上接第 82 页)

据传输接口、多功能的芯片控制逻辑和易于修改的驱动程序,可以满足 SM234 密码芯片的 FPGA 验证与测试要求.本系统降低密码芯片的流片风险,提高了芯片测试效率,减少了密码芯片的开发时间和开发成本.

参考文献:

- [1] 王齐. PCI Express 体系结构导读 [M]. 北京:机械工业出版社,2010.
- [2] Xilinx. LogiCore IP endpoint block plus V1.9 for PCIeExpress [s]. USA:California:Xilinx Corp., 2008.

[3] 何妍,李树国.一种具有双 PCI 接口的 ASIC 验证系统[J].微电子学,2006(4):36-38.

- [4] Xilinx. Virtex-5 FPGA integrated endpoint block for PCI express designs [s]. USA: California: Xilinx Corp., 2008.
- [5] 武安河. Windows 设备驱动程序 WDF 开发 [M]. 北京:电子工业出版社,2009.

作者简介:

刘学光 男,(1989-),硕士研究生.研究方向为信息安全.
李树国 男,(1963-),教授,博士生导师.研究方向为信息安全算法及其大规模集成电路设计与实现