

引用格式:宋宇鲲,曲双双,徐礼晗,等.混合基可重构 FFT 处理器的设计与实现[J].微电子学与计算机,2020,37(1):87-92.
[SONG Y K, QU S S, XU L H, et al. Design and implementation of mixed radix reconfigurable Fast Fourier Transform processor[J]. Microelectronics & Computer, 2020, 37(1): 87-92.]

混合基可重构 FFT 处理器的设计与实现

宋宇鲲,曲双双,徐礼晗,张多利

(合肥工业大学 电子科学与应用物理学院,安徽 合肥 230009)

摘 要:本文提出了一种新型混合基可重构 FFT 处理器,由支持基-2/3FFT 的新型可重构蝶形单元和多路并行无冲突的存储器组成,实现了 FFT 过程中多路数据并行性和操作的连续性.本设计在 TSMC28nm 工艺下的最高频率为 1.06GHz,同时在 Xilinx 的 XC7V2000T FPGA 芯片上搭建了混合基 FFT 处理器硬件测试系统.对混合基 FFT 处理器的 FPGA 硬件测试结果表明,本设计支持基-2、基-3 和基-2/3 混合模式 FFT 变换,且执行速度达到给定蝶乘器数量下的理论周期值,对单精度浮点数,混合基 FFT 处理器可提供 10^{-5} 的结果精度.

关键词:混合基;快速傅里叶变换;无冲突地址规则;连续数据流

中图分类号:TN47

文献标识码:A

文章编号:1000-7180(2020)01-0087-06

Design and implementation of mixed radix reconfigurable Fast Fourier Transform processor

SONG Yu-kun, QU Shuang-shuang, XU Li-han, ZHANG Duo-li

(School of Electronic Science and Applied Physics, Hefei University of Technology, Hefei 230009, China)

Abstract: This paper presents a novel mixed radix reconfigurable FFT processor, which consists of a new reconfigurable butterfly unit supporting radix-2/3 FFT and multi-channel parallel collision-free memory, which implements multi-channel data parallelism and the continuity of the operation for FFT operation. The maximum frequency of this design is 1.06 GHz under the TSMC28nm process. At the same time, the mixed radix FFT processor hardware test system based on the Xilinx's XC7V2000T FPGA chip completes performance verification. The results shows that the design supports FFT operations for radix-2 mode, radix-3 mode and radix-2/3 mixed mode, and the execution speed reaches the theoretical period value under the given number of butterfly multipliers. For single-precision floating-point numbers, mixed radix FFT processor provides a result accuracy of 10^{-5} .

Key words: mixed radix; FFT; conflict-free address rule; continuous flow of data

1 引言

在高精度和高分辨率雷达成像等系统中,快速傅里叶变换算法(FFT)因其具备较高的信号量化噪声比而得到了广泛应用^[1-2].近几十年来,人们提出了各种 FFT 算法的实现结构保证计算精度的同时减少面积、功耗和延迟^[3-7].FFT 处理器主要包括蝶

形运算单元、数据存储单元与蝶形因子生成单元.FFT 处理器的结构复杂度主要由蝶形运算单元的基和数量决定.高效的 FFT 处理器的性能核心在于蝶形运算单元和存储结构的设计^[8-9].基-2 算法实现结构较为简单得到了广泛应用^[10],但处理非 2 次幂的点数 FFT 时要进行补零,导致需要更多存储资源和处理时间,对于大点数 FFT 这种操作会造成极

大的资源浪费^[11],而在某些特定应用场景下补零操作是不被许可的,如合成孔径雷达聚束模式下信号处理^[12-13]是不能进行点数扩展的.部分FFT处理器将几个不同基FFT运算单元拼接起来,进行混合基的计算.每种基的蝶形运算单元(BF)的存储器只存储该基变换所需数据存储操作,这导致电路缺少灵活性,增加了设计规模^[14].因此本文设计了基-2和基-3混合基FFT处理器,可以根据每层运算需要将可重构蝶形运算单元(RCBF)配置为基-2FFT或基-3FFT工作模式.

2 算法原理

由文献[15]可知,基-2FFT蝶形运算公式如下:

$$\begin{cases} X_1(k) = X'_1(k) + X'_2(k) * W_N^k \\ X_2(k) = X'_1(k) - X'_2(k) * W_N^k \end{cases} \quad (1)$$

式中: $W_N^k = e^{-\frac{2\pi k}{N}}$, $(k = 0, 1, 2, \dots, \frac{N}{2} - 1)$

$X'_i(k)$ 和 $X_i(k)$ ($i = 1, 2$) 分别为蝶形计算单元输入操作数和输出结果数.

由文献[16]可知,基-3FFT蝶形运算公式:

$$\begin{cases} X_1(k) = X'_1(k) + X'_2(k) * W_N^k \\ \quad \quad \quad + X'_3(k) * W_N^{2k} \\ X_2(k) = X'_1(k) + X'_2(k) * W_N^k * W_3^1 \\ \quad \quad \quad + X'_3(k) * W_N^{2k} * W_3^2 \\ X_3(k) = X'_1(k) + X'_2(k) * W_N^k * W_3^2 \\ \quad \quad \quad + X'_3(k) * W_N^{2k} * W_3^1 \end{cases} \quad (2)$$

式中: $W_N^k = e^{-\frac{2\pi k}{N}}$, $(k = 0, 1, 2, \dots, \frac{N}{3} - 1)$, $W_3^1 =$

$\frac{1}{2} * (-1 - j * \sqrt{3})$, $W_3^2 = \frac{1}{2} * (-1 + j * \sqrt{3})$, $X'_i(k)$ 和 $X_i(k)$ ($i = 1, 2, 3$) 分别为蝶形单元输入操作数和输出结果数.

由文献[17]可知,混合基FFT算法公式如下:

当FFT点数为 $N = r_1 r_0$ 时,将 n 表示为:

$$n = n_1 r_1 + n_0 \begin{cases} n_0 = 0, 1, \dots, r_1 - 1 \\ n_1 = 0, 1, \dots, r_0 - 1 \end{cases} \quad (3)$$

这样将 N 点分解为 r_1 个 r_0 点 $x(n)$ 序列.

将频率变量 k 表示为:

$$k = k_1 r_1 + k_0 \begin{cases} k_1 = 0, 1, \dots, r_1 - 1 \\ k_0 = 0, 1, \dots, r_0 - 1 \end{cases} \quad (4)$$

将 n, k 带入DFT公式可得混合基公式:

$$X(k_1, k_0) =$$

$$\sum_{n_0=0}^{r_1-1} \left\{ \left[\sum_{n_1=0}^{r_0-1} x(n_1, n_0) W_{r_0}^{n_1 k_0} \right] W_N^{n_0 k_0} \right\} W_{r_1}^{n_0 k_1} \quad (5)$$

由公式可以推出 $N = r_1 r_0$ 点的混合基FFT算法:首先以 r_1 为进制将序列系数 n 表示为 $n = n_1 r_1 + n_0$ 的形式,这表明将序列 $x(n)$ 分解成 r_1 个 r_0 点的序列,然后将 r_1 个 r_0 点进行DFT计算得到 $X_1(k_0, n_0)$;将 $X_1(k_0, n_0)$ 乘以中间旋转因子 $W_N^{n_0 k_0}$ 后,再做 r_0 个 r_1 点DFT计算得到 $X_2(k_1, k_0)$,最后得到结果 $X(k_1, k_0) = X(k)$.以计算6点FFT为例,做混合基FFT流程图如图1.

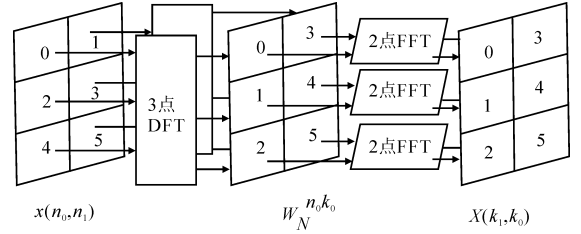


图1 6点混合基FFT流程图

3 系统框架

混合基可重构FFT系统包括:(1)控制单元(CU, Control Unit)是整个系统的工作核心,负责控制各模块相互协作;(2)地址生成单元(AGU, Address Generation Unit)负责生成蝶形运算中的操作数地址、结果数地址和旋转因子的读取地址;(3)可重构蝶形单元(RCBF, Reconfigurable Butterfly)负责在不同模式下相对应的FFT运算操作;(4)旋转因子生成单元(TFG, Twiddle Factor Generator)负责生成每次蝶形运算所需的旋转因子和混合基运算时的中间旋转因子;(5)存储块(MB, Memory Block)负责存储运算操作数据和结果数据.

图2为论文提出的可重构FFT处理器整体架构,CU将MB分别配置为操作数存储器和结果数存储器,运算时从操作数存储器中读取操作数,并且由TFG生产旋转因子,一同送入RCBF中进行计算,生成结果数存储到结果数存储器中,两组块存储器进行乒乓操作,完成8K点以内的 $2^m * 3^n$ 点数的FFT运算.其中,基-2模式下, $4 < m < 14$;基-3模式下, $2 < n < 8$;混合基模式下, $4 < m < 8, 2 < n < 5$.特别的,在进行混合基FFT运算时,先进行 $2m$ 个 $3n$ 点的基-3FFT运算,将中间结果乘以中间旋转因子之后,再进行 $3n$ 个 $2m$ 点的基-2FFT运算,最后得到FFT运算结果.

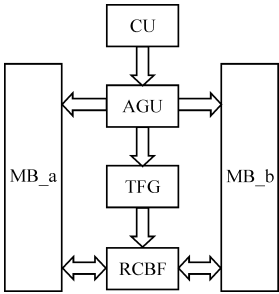


图 2 可重构 FFT 处理器架构

3.1 可重构蝶形运算单元

由公式(1)(2)可发现,基 2/3 运算公式中有公共部分如 $X'_2(k) * W_N^k$,即基-2/3 都有输入第二、三个操作数和旋转因子相乘的部分,将此处的电路复用可以节约乘法资源.在进行数据处理操作时,由控制单元配置寄存器开关,将可重构蝶形运算单元配置为两个并行的基-2 蝶形运算器或一个基-3 蝶形运算器进行 FFT 运算操作.

表 1 为图 3 中的符号说明,图 3 为混合基蝶形单元的结构图.图 4 和图 5 分别为基-2 蝶形单元的结构图和基-3 蝶形单元的结构图.运算时通过选择器控制输入端 mode 判断选择选通电路.图 4 和图 5 中实线表示选通的连线,虚线为不选通的连线.

表 1 符号说明

符号	说明
$X_{in}(i)(i=0,1,2,3)$	基-2: $X_{in}(i)(i=0,1,2,3)$ 为四个输入操作数; 基-3: $X_{in}(i)(i=0,1,2)$ 为三个操作数.
$X_{out}(i)(i=0,1,2,3)$	基-2: $X_{out}(i)(i=0,1,2,3)$ 为四个输出结果数; 基-3: $X_{out}(i)(i=0,1,2)$ 为三个输出结果数.
W_1	基-2:第一个蝶形运算器需要的旋转因子 基-3:第二个操作数需要的旋转因子
W_2	基-2:第二个蝶形运算器需要的旋转因子 基-3:第三个操作数需要的旋转因子
$2 * W_3^1$	基-3:第二、三个操作数的旋转因子系数
$2 * W_3^2$	基-3:第二、三个操作数的旋转因子系数
$\otimes$	2 输入复数浮点乘法操作
$\oplus$	2 输入复数浮点加法操作
$\ominus$	2 输入复数浮点减法操作
$\gg 1$	移位寄存器,将数据向右移动一位
$\neg$	反相器
$\swarrow \searrow$	选择器

本文设计混合基 FFT 处理器支持三类操作模式:只采用基-2 模式的 2 次幂点数 FFT,只采用基-3 模式的 3 次幂点数 FFT 和采用了基-3 和基-

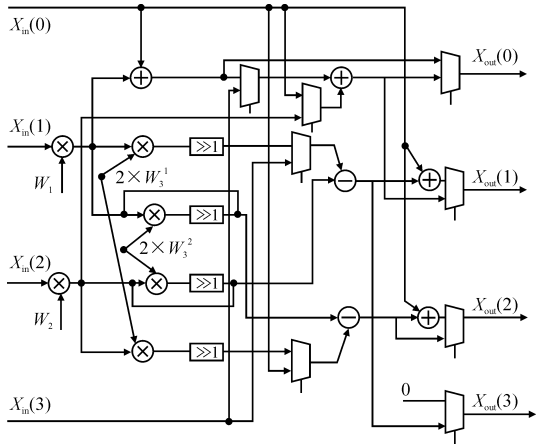


图 3 混合基蝶形单元电路图

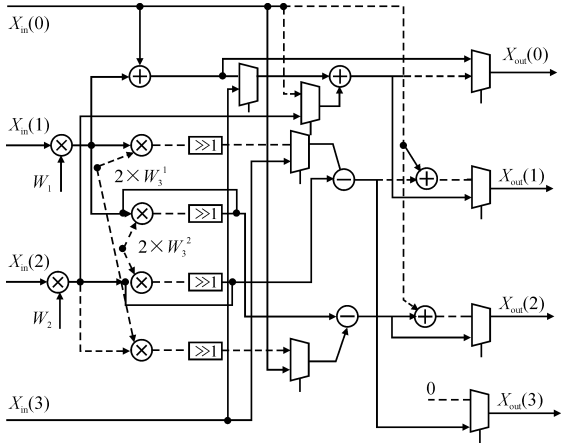


图 4 基-2 蝶形单元电路图

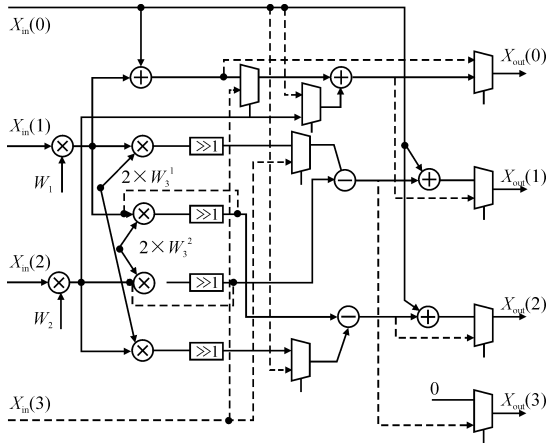


图 5 基-3 蝶形单元电路图

2 混合模式的 $2m * 3n$ 点数 FFT,在基-3 模式转换至基-2 模式时,需对基-3 运算的结果数据乘以中间旋转因子作为基-2 运算的源数据.

3.2 无冲突地址跳变规则

在进行 FFT 运算操作时,记每层运算的当前运算次数为 X , N 为一次 FFT 运算点数,记基-2FFT

运算为一次 FFT 运算,基-3FFT 运算为一次 FFT 运算,混合基 FFT 运算为 $n+m$ 次 FFT 运算. 在基-2 模式下, X 最大值为 $N/4$,在基-3 模式下, X 的最大值为 $N/3$. 每一次运算中,FFT 蝶形运算所需的数据的读取过程和生成的结果数据的存储过程与当前运算次数 X 有关,规律如下.

3. 2. 1 运算数据读取过程

在进行基-2FFT 运算时,两个并行的基-2 蝶形运算单元每次需要从 Memory0,1,2,3 中读取 4 个操作数,对应的 Memory 读取顺序和数据读取地址顺序规则如表 2.

表 2 基-2 模式数据读操作地址规则

Memory	数据读操作地址规则
Memory0	$0,1,\cdots,\frac{N}{4}-2,\frac{N}{4}-1$
Memory1	$0,1,\cdots,\frac{N}{4}-2,\frac{N}{4}-1$
Memory2	$0,1,\cdots,\frac{N}{4}-2,\frac{N}{4}-1$
Memory3	$0,1,\cdots,\frac{N}{4}-2,\frac{N}{4}-1$
当前运算次数 X	$1,2,3,4,\cdots,\frac{N}{4}-1,\frac{N}{4}$

从表 3 中可以看出,在当前运算次数小于 $N/8$ 时,既在进行前 $N/4$ 个蝶形运算时,按照 Memory0,Memory1,Memory2,Memory3 顺序读取出操作数送入蝶形单元进行运算操作;在运算次数大于 $N/8$,既在进行后 $N/4$ 个蝶形运算时,按照 Memory3,Memory2,Memory1,Memory0 顺序读取出操作数送入蝶形单元进行运算操作.

表 3 基-2 模式蝶形单元输入规则

输入	蝶形单元输入规则	
$X_{in}(0)$	Memory0	Memory3
$X_{in}(1)$	Memory1	Memory2
$X_{in}(2)$	Memory2	Memory1
$X_{in}(3)$	Memory3	Memory0
当前运算次数 X	$1,2,\cdots,\frac{N}{8}$	$\frac{N}{8}+1,\frac{N}{8}+2,\cdots,\frac{N}{4}$

在进行基-3FFT 运算时,一个基-3 蝶形运算单元每次从 Memory0,1,2 中读取 3 个操作数,对应的数据读取地址顺序和 Memory 读取顺序规则如表 4.

从表 5 中可以看出,在当前次数小于 $N/9$ 时,既在进行前 $N/3$ 个蝶形运算时,按照 Memory0,Memory1,Memory2 顺序读取出操作数送入蝶形运算单元进行运算操作;在运算次数在 $N/9+1$ 和 $2N/9$ 之间,既在进行 $N/3+1$ 和 $2N/3$ 之间时,按

表 4 基-3 模式数据读操作地址规则

Memory	数据读操作地址规则
Memory0	$0,\frac{N}{9},\frac{2N}{9},1,\frac{N}{9}+1,\frac{2N}{9}+1,\cdots,\frac{N}{3}-3,\frac{N}{3}-2,$ $\frac{N}{3}-1$
Memory1	$0,\frac{N}{9},\frac{2N}{9},1,\frac{N}{9}+1,\frac{2N}{9}+1,\cdots,\frac{N}{3}-3,\frac{N}{3}-2,$ $\frac{N}{3}-1$
Memory2	$0,\frac{N}{9},\frac{2N}{9},1,\frac{N}{9}+1,\frac{2N}{9}+1,\cdots,\frac{N}{3}-3,\frac{N}{3}-2,$ $\frac{N}{3}-1$
当前运算次数 X	$1,2,3,4,\cdots,\frac{N}{3}-1,\frac{N}{3}$

照 Memory1,Memory2,Memory0 顺序读取出操作数送入蝶形运算单元进行运算操作;在运算次数在 $N/9+1$ 和 $2N/9$ 之间,既在进行 $N/3+1$ 和 $2N/3$ 之间时,按照 Memory1,Memory2,Memory0 顺序读取出操作数送入蝶形运算单元进行运算操作,在运算次数大于 $2N/9+1$,既在进行后 $N/3$ 个蝶形运算时,按照 Memory2,Memory0,Memory1 顺序读取出操作数送入蝶形运算单元进行运算操作.

表 5 基-3 模式蝶形单元输入规则

运算单元输入序列		蝶形单元输入规则	
$X_{in}(0)$	Memory_0	Memory_1	Memory_2
$X_{in}(1)$	Memory_1	Memory_2	Memory_0
$X_{in}(2)$	Memory_2	Memory_0	Memory_1
当前运算次数 X	$1,2,\cdots,\frac{N}{9}$	$\frac{N}{9}+1,\frac{N}{9}+2,\cdots,\frac{2N}{9}$	$\frac{2N}{9}+1,\frac{2N}{9}+2,\cdots,\frac{N}{3}$

3. 2. 2 结果数据存储过程

在进行基-2FFT 运算时,两个并行的蝶形运算单元每次生成得 4 个运算结果数存储到另一组相对应的 Memory0,1,2,3 中,对应的结果数存储地址序列和 Memory 存储顺序规则如表 6.

由表 6 可以看出,基-2 数据存储地址序列规则为,当 X 为奇数时,Memory0、Memory1 存储地址为 $0,1,2,\cdots,N/8-1$;当 X 为偶数时,Memory0、Memory1 存储地址为 $N/8,N/8+1,N/8+2,\cdots,N/4-1$. Memory2、Memory3 的规则正好与之相反. 当 X 为奇数时,Memory2、Memory3 存储地址为 $N/8,N/8+1,N/8+2,\cdots,N/4-1$;当 X 为偶数时,Memory2、Memory3 存储地址为 $0,1,2,\cdots,N/8-1$.

表 6 基-2 模式数据写操作地址规则

Memory	数据写操作地址规则
Memory0	$0, \frac{N}{8}, 1, \frac{N}{8} + 1, \dots, \frac{N}{8} - 1, \frac{N}{4} - 1$
Memory1	$0, \frac{N}{8}, 1, \frac{N}{8} + 1, \dots, \frac{N}{8} - 1, \frac{N}{4} - 1$
Memory2	$\frac{N}{8}, 0, \frac{N}{8} + 1, 1, \dots, \frac{N}{4} - 1, \frac{N}{8} - 1$
Memory3	$\frac{N}{8}, 0, \frac{N}{8} + 1, 1, \dots, \frac{N}{4} - 1, \frac{N}{8} - 1$
X	$1, 2, 3, 4, \dots, \frac{N}{4} - 1, \frac{N}{4}$

由表 7 可以看出,基-2 模式运算单元数据输出序列规则为,当前运算次数 X 为奇数时,运算结果数据按照 Memory0、Memory3、Memory2、Memory1 的顺序分别存入存储块中;当前运算次数 X 为偶数时,运算结果数据按照 Memory2、Memory1、Memory3、Memory0 的顺序分别存入到存储块中。

表 7 基-2 模式蝶形单元输出规则

	运算单元输出序列	蝶形单元输出规则
$X_{out}(0)$	Memory0	Memory2
$X_{out}(1)$	Memory3	Memory1
$X_{out}(2)$	Memory1	Memory3
$X_{out}(3)$	Memory2	Memory0
当前运算次数 X	奇数	偶数

在进行基-3FFT 运算时,一个的蝶形运算单元每次生成得 3 个运算结果数存储到另一组相对应的 Memory0,1,2 中,对应的结果数存储地址序列和 Memory 存储顺序规则如表 8。

表 8 基-3 模式数据读操作地址规则

Memory	数据读操作地址规则		
Memory0	$0, \dots, \frac{N}{9} - 1$	$\frac{2N}{9}, \dots, \frac{N}{3} - 1$	$\frac{N}{9}, \dots, \frac{2N}{9} - 1$
Memory1	$\frac{N}{9}, \dots, \frac{2N}{9} - 1$	$0, \dots, \frac{N}{9} - 1$	$\frac{2N}{9}, \dots, \frac{N}{3} - 1$
Memory2	$\frac{2N}{9}, \dots, \frac{N}{3} - 1$	$\frac{N}{9}, \dots, \frac{2N}{9} - 1$	$0, \dots, \frac{N}{9} - 1$
X	$0, \dots, \frac{N}{9} - 1$	$\frac{N}{9}, \dots, \frac{2N}{9} - 1$	$\frac{2N}{9}, \dots, \frac{N}{3} - 1$

由表 8 可以看出,基-3 数据存储地址序列规则为,将 0 至 $N/9-1$ 视为第一段序列,将 $N/9$ 至 $2N/9$ 视为第二段序列,将 $2N/9$ 至 $N/3-1$ 视为第三段序列。Memory0 存储地址序列规则为第一、三、二序列,Memory1 存储地址序列规则为第二、一、三序列,Memory2 存储地址序列规则为第三、二、一序列。

由表 9 可以看出,基-3 模式运算单元数据输

出序列规则为,当前运算次数 X 小于 $N/9$ 时,运算结果数据按照 Memory0、Memory1、Memory2 的顺序分别存入存储块中;当前运算次数 X 大于 $N/9$ 小于 $2N/9$ 时,运算结果数据按照 Memory1、Memory2、Memory0 的顺序分别存入存储块中;当前运算次数 X 大于 $2N/9$ 时,运算结果数据按照 Memory2、Memory0、Memory1 的顺序分别存入存储块中。

表 9 基-3 模式蝶形单元输出规则

	运算单元输出序列	蝶形单元输出规则		
$X_{out}(0)$	Memory0	Memory1	Memory2	
$X_{out}(1)$	Memory1	Memory2	Memory0	
$X_{out}(2)$	Memory2	Memory0	Memory1	
X	$0, 1, \dots, \frac{N}{9} - 1$	$\frac{N}{9}, \frac{N}{9} + 1, \dots, \frac{2N}{9} - 1$	$\frac{2N}{9}, \frac{2N}{9} + 1, \dots, \frac{N}{3} - 1$	

4 设计实现与性能分析

本文使用 Verilog 语言完成混合基可重构 FFT 处理器电路设计,在 TSMC28nm 工艺下,本文的设计最大频率为 1.06 GHz,面积约为 1.65 mm²。

论文以 Xilinx XC7V2000T FPGA 芯片为载体搭建了混合基可重构 FFT 处理器的硬件原型测试环境,表 10 给出了 FPGA 的资源消耗。将 10000 组 512 点、729 点和 864 点的数据分别输入混合基可重构 FFT 硬件设计和 MATALB 软件,对两者的结果进行的误差分析,结果如表 11 所示。

表 10 FPGA 中 FFT 综合结果

Resource	Estimation	Available	Utilization/%
LUT	4 009	362 800	1.11
LUTRAM	137	141 600	0.10
FF	1 840	725 600	0.25

表 11 FFT 误差分析

点数	最大误差	平均误差
512	0.001 94	4.4e-5
729	0.000 16	7.1e-6
864	0.001 74	8.17e-5

表 11 中 864 点 FFT 实测最大误差 0.001 74,平均误差 8.17e-5,符合 FFT 处理器计算精度要求,适用于更多点数的 FFT 运算。图 6 为 864 测试结果与 MATLAB 计算结果的相对误差(绝对值)波形图,并且在表 12、13 中给出了硬件实现的部分数据的实虚部测试数据对比。

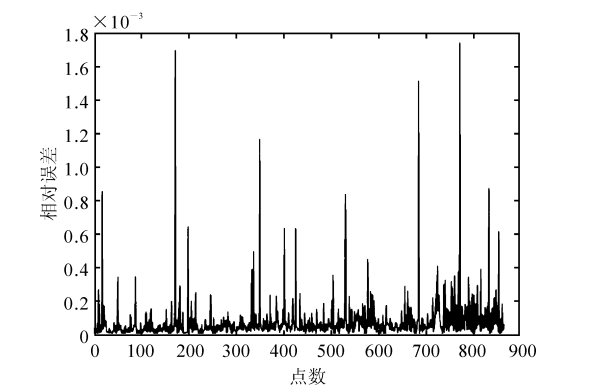


图 6 基于本设计的 864 点 FFT 结果与 Matlab 计算结果的相对误差

表 12 硬件实现的部分数据实部的测试数据对比

MATLAB 仿真结果	硬件测试结果	相对误差
−36 073.524 848	−36 073.7	4.86e−6
1 041.557 523	1 041.56	2.38e−6
−1 469.870 392	−1 469.87	2.7e−7
−15 851.498 197	−15 851.5	1.14e−7
1 151.742 5	1 151.73	1.08e−5
1 285.550 65	1 285.55	5.1e−7

表 13 硬件实现的部分数据虚部的测试数据对比

MATLAB 仿真结果	硬件测试结果	相对误差
38 013.714 3	38 013.7	3.8e−7
−20.123 28	−20.131	3.84e−4
−862.796 643	−862.799	2.73e−6
47 089.223 706	470 899.3	1.62e−6
−543.454 952	−543.452	5.43e−6
−2 274.546 888	−2 274.54	3.03e−6

在 100 MHz 系统时钟下,本设计 FPGA 版本的基-3 蝶形运算器整体运算流水级为 25 级,运算结果传输延时为 250 ns,小于文献[16]中的输出延时310 ns,因此本设计的 FFT 大大减少了运算延时,提高了运算速度,本设计完全满足精度需求较高的实时信号处理系统的要求。

5 结束语

本文给出了一种有效的单精度浮点数混合基 2/3FFT 设计,可以完成 8k 点以内的部分 $2^m \times 3^n$ 点数的 FFT 运算,相比传统的 FFT 处理器,本设计扩大了可执行点数的范围,同时与文献[16]相比减少了输出延时,保证了执行效率.试验结果表明本设计完全满足精度需求较高的实时信号处理系统的要求。

参考文献:

[1] 吕倩,苏涛. 基于改进型快速双线性参数估计的复杂运动目标 ISAR 成像[J]. 电子与信息学报, 2016, 38(9): 2301-2308.

[2] HUANG S J, CHEN S. A high-throughput radix-16 FFT processor with parallel and normal input/output ordering for IEEE 802. 15. 3c systems [J]. IEEE Transactions on Circuits and System I: Regular Papers, 2012,59(8):1752-1765.

[3] 陈杰男,费超,袁建生. 超高速全并行快速傅里叶变换器[J]. 电子与信息学报, 2016, 38(9): 2410-2414.

[4] JONEWOOK S, EARL E. Improved architectures for a floating-point fused dot product unit [C]//IEEE Symposium on Computer Arithmetic, Austin. USA, 2013: 41-48.

[5] 王旭东,刘渝. 一种基于 FFT 的超宽带数字正交变换算法及实现 [J]. 仪器仪表学报, 2008, 29(8): 1714-1718.

[6] LUZ A G DA, COSTA E A DA, GHISSONI S. Reducing power consumption in FFT architectures by using heuristic-based algorithms for the ordering of the twiddle factors [C]//Proc . IEEE 3rd Latin Amer. Symp. Circuits Syst. (LASCAS) 2012, [S. L]. 2012: 1-4.

[7] BHAGAT D, MERGLER H W. A high-performance microprocessor-based FFT processor[J]. IEEE Trans. Very Large Scale Integration. (VLSI) Syst. ,2016,24(9):3008-3012.

[8] 杨军,郭跃东,丁俊. 一种高速实时浮点蝶形运算单元的设计与实现 [J]. 仪器仪表学报, 2010, 31(3): 519-524.

[9] 朱冰莲,孔杰. 高效复数流水蝶形计算单元的 FPGA 实现[J]. 电子测量与仪器学报, 2005,19(4): 77-80.

[10] ABER M A, MASSICOTTE D. A new FFT concept for efficient VLSI implementation: Part I-Butterfly processing element [C]//16th International Conference on Digital Signal Processing. Santorini-Hellas, 2009.

[11] SUZUKIY, SONET, KIDOKI. New FFT algorithm of radix-3, 6, AND 12[J]. IEEE Transactions on Acoustics, Speech and Signal Processing, 1986, 34(2): 380-383.

[12] 高祥武,雷宏,杨汝良. 合成孔径雷达聚束模式合成孔径长度研究 [J]. 电子与信息学报, 2004, 26(6): 953-958.

[13] 宗皓. 聚束模式弹载 SAR 成像及抗干扰[D]. 成都:电子科技大学,2014.

拟机数据交换和带宽分配中存在的不足,从硬件层面出发对以太网控制器的系统架构进行研究,并提出了对虚拟机友好的 VFN 架构. 基于此,提出了硬件支持的多虚拟机数据交换方法以及动态的带宽分配方法. 实验结果表明,本文提出的方法不仅有效减小了多虚拟机之间数据交换和带宽分配的 CPU 开销,而且整个处理过程对上层的虚拟机管理程序保持透明.

参考文献:

[1] 金顺福, 郝闪闪, 王宝帅. 融合双速率和工作休眠的虚拟机调度策略及参数优化[J]. 通信学报, 2017, 38(12):10-20.

[2] 叶可江, 吴朝晖, 姜晓红, 等. 虚拟化云计算平台的能耗管理[J]. 计算机学报, 2012, 35(6):1262-1285.

[3] ADAMS K, AGESEN O. A comparison of software and hardware techniques for x86 virtualization[J]. ACM SIGOPS Operating Systems Review, 2006, 40(5): 2-13.

[4] GUAN H, MA R, LI J. Workload-aware credit scheduler for improving network I/O performance in virtualization environment[J]. IEEE transactions on cloud computing, 2014, 2(2): 130-142.

[5] SHEA R, LIU J. Network interface virtualization: challenges and solutions[J]. IEEE Network, 2012, 26(5): 28-34.

[6] DONG Y, YANG X, LI J, et al. High performance network virtualization with SR-IOV[J]. Journal of Parallel and Distributed Computing, 2012, 72(11): 1471-1480.

[7] FEI Z, LIU G, FU X, et al. A Survey on Virtual Machine Migration: Challenges, Techniques and Open Issues[J]. IEEE Communications Surveys & Tutorials, 2018(99):1.

[8] GUMASTE A, DAS T, KHANDWALA K, et al. Network Hardware Virtualization for Application Provisioning in Core Networks[J]. IEEE Communications Magazine, 2017, 55(2):152-159.

[9] AZODOLMOLKY S, WIEDER P, YAHYAPOUR R. Cloud computing networking: challenges and opportunities for innovations[J]. IEEE Communications Magazine, 2013, 51(7): 54-62.

[10] 张洪素. 基于 Xen 的虚拟交换技术的研究与设计[D]. 天津:天津大学, 2012.

[11] CHERKASOVA L, GARDNER R. Measuring CPU Overhead for I/O Processing in the Xen Virtual Machine Monitor[C]//USENIX Annual Technical Conference, General Track, 2005: 50.

[12] 郭军, 闫永明, 马安香, 等. 云环境下基于冷点虚拟机迁移的热点消除方法[J]. 清华大学学报(自然科学版), 2016(11):1232-1236.

[13] LIAO G, BHUYAN L. Performance measurement of an integrated nic architecture with 10gbe[C]// 17th IEEE Symposium on High Performance Interconnects. New York, NY, USA ,IEEE, 2009: 52-59.

作者简介:

李龙飞, 男,(1988-),博士,工程师. 研究方向为高性能网络芯片、网络硬件加速、网络安全等.
E-mail: longfeisos@163.com.

王剑峰 男,(1971-),研究员. 研究方向为高性能系统总线、计算机网络及控制、交换芯片等.

刘欢 女,(1983-),高级工程师. 研究方向为高性能系统总线及控制芯片等.

史阳春 男,(1983-),高级工程师. 研究方向为高性能网络控制、交换芯片等.

(上接第 92 页)

[14] SHI X Y, LIU Y Q, CHOU H R. 48-Mode Reconfigurable Design of SDF FFT Hardware Architecture Using Radix-32 and Radix-23 Design Approaches[J]. IEEE Transactions On Circuits And Systems-I: Regular Papers, 2017, 64(4): 1456 - 1467. DOI: 10. 1109/ TCSI. 2017. 2654451.

[15] 杨学鹏. 异构多核 SoC 中大点数 FFT 加速单元的实现[D]. 安徽:合肥工业大学, 2013.

[16] 禹霁阳, 李阳, 黄丹, 等. 一种高性能单精度浮点基-3 蝶形运算单元的设计与实现[J]. 仪器仪表学报, 2010, 31(12): 2676-2681.

[17] 王靖琰, 曹新民. 混合基 FFT 处理器的设计与 Verilog 实现[J]. 中国科技论文在线精品论文, 2008, 1(10):

1128-1135. <http://www.paper.edu.cn/releasepaper/content/200802-267>.

[18] 方波. 基于改进 CORDIC 算法的 FFT 处理器设计与研究[D]. 广州:华南理工大学, 2013.

作者简介:

宋宇鲲 男,(1975-),博士,副研究员. 研究方向为面向数据密集与计算机密集应用的 SoC/MPSoC 体系结构与实现.

曲双双 女,(1993-),硕士研究生. 研究方向为 SoC 设计.

徐礼晗 男,(1994-),硕士研究生. 研究方向为 SoC 设计.

张多利(通讯作者) 男(1975-),博士,研究员. 研究方向为多核处理器体系结构与设计方法、多媒体专用集成电路设计实现. E-mail: zhangduoli@hfut.edu.cn.