第 38 卷 第 2 期 计 算 机 工 程 2012 年 1 月
Vol.38 No.2 Computer Engineering January 2012

• 工程应用技术与实现 • 文章编号：1000—3428(2012)02—0242—03 文献标识码：A 中图分类号：TP303

高性能并行 FFT 处理器的设计与实现

石长振，杨 雪，王贞松

(中国科学院计算技术研究所，北京 100190)

摘 要：提出一种高性能并行快速傅里叶变换(FFT)处理器的设计方案，采用 4 个蝶形单元进行并行处理，利用改进的无冲突操作数地址映射方式，保证每个周期同时读取和写入 16 个数据。给出该处理器的 FPGA 实现，性能评测结果表明，与其他 FFT 处理器相比，该并行 FFT 处理器的性能较优，能满足实际应用需求。

关键词：快速傅里叶变换；并行处理；流水线；块浮点；蝶形单元

Design and Realization of High Performance Parallel FFT Processor

SHI Chang-zhen, YANG Xue, WANG Zhen-song

(Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China)

【Abstract】 This paper proposes a design of high performance parallel Fast Fourier Transform(FFT) processor. It uses four butterfly units in parallel processing. It uses an improved conflict free memory addressing method, and 16 data can be read, processed and written in one cycle simultaneously. It gives the FPGA implementation of the processor, performance evaluation results show that the high performance parallel FFT processor is superior to other FFT processors, and can meet the application needs.

【Key words】 Fast Fourier Transform(FFT); parallel processing; pipeline; block floating point; butterfly unit

DOI: 10.3969/j.issn.1000-3428.2012.02.081

1 概述

快速傅里叶变换(Fast Fourier Transform, FFT)是离散傅里叶变换(Discrete Fourier Transform, DFT)的快速实现形式[1]。在很多实时信号处理领域中，高性能的 FFT 处理器成为系统的核心部件，有些系统的处理性能就取决于 FFT 的处理速度。

提高 FFT 处理器的处理性能通常采用并行技术、流水线技术和高基算法。但是随着运算部件并行度的提高，数据的读写速度成为提高处理性能的一个瓶颈。文献[2]提出一种基 r 的无冲突的地址映射方法，实现在一个时钟周期内蝶形运算的 r 个数据的无冲突读写，但没有针对实现多个蝶形单元并行处理时所需要更高程度的数据并行读写问题。文献[3]设计了混合基方案提高处理的并行度，但是随着基数的加大，蝶形单元的结构变得复杂，占用了较多逻辑资源。

为此，本文设计一种并行结构的 FFT 处理器，采用 4 个基 4 蝶形单元，处理速度为单蝶形的 4 倍。在文献[2]地址映射算法的基础上进行改进，每个周期可以同时实现 16 个数据的读取和写入。

2 FFT 数学模型

FFT 的基本思想是利用旋转因子的周期性、对称性和可约性将一个长度为 N 的序列的 DFT 逐次分解为较短的 DFT 来计算，而总的运算次数比直接 DFT 运算要少得多，达到提高速度的目的。FFT 变换可以分为时域和频域抽取算法，也可分为基 2、基 4、分裂基等算法。当 $N=4^M$ 时，对 N 点 DFT 按如下方法做频率抽取[1]：

$$X(k)=\sum_{n=0}^{N/4-1}x(n)W_N^{nk}+\sum_{n=N/4}^{N/2-1}x(n)W_N^{nk}+\sum_{n=N/2}^{3N/4-1}x(n)W_N^{nk}+\sum_{n=3N/4}^{N-1}x(n)W_N^{nk} \tag{1}$$

分别令 $k=4r$、$k=4r+2$、$k=4r+1$、$k=4r+3$，$r=0,1,\cdots,\frac{N}{4}-1$，得到：

$$X(4r)=\sum_{n=0}^{N/4-1}\left\{[(x(n)+x(n+\frac{N}{2})]+[x(n+\frac{N}{4})+x(n+3\frac{N}{4})]\right\}W_{N/4}^{nr}$$

$$X(4r+2)=\sum_{n=0}^{N/4-1}\{[(x(n)+x(n+\frac{N}{2})]-[x(n+\frac{N}{4})+x(n+3\frac{N}{4})]\}W_N^{2n}W_{N/4}^{nr}$$

$$X(4r+1)=\sum_{n=0}^{N/4-1}\{[(x(n)-x(n+\frac{N}{2})]-\mathrm{j}[x(n+\frac{N}{4})-x(n+3\frac{N}{4})]\}W_N^{n}W_{N/4}^{nr}$$

$$X(4r+3)=\sum_{n=0}^{N/4-1}\{[(x(n)-x(n+\frac{N}{2})]+\mathrm{j}[x(n+\frac{N}{4})-x(n+3\frac{N}{4})]\}W_N^{3n}W_{N/4}^{nr} \tag{2}$$

式(2)是 FFT 基 4 频域抽取算法的基本运算单元，一般称为蝶形运算单元，简称蝶算单元。

3 关键技术

设计高性能 FFT 处理器的关键在于用尽可能少的资源取得尽可能快的处理速度和尽可能高的处理精度，以满足系统对 FFT 处理器的需求。通常通过计算量来衡量 FFT 算法的性能，但是在用硬件设计高性能 FFT 处理器时，往往硬件资源和结构复杂度是更需要考虑的问题。高性能 FFT 处理器的设计一般包括以下 4 个关键问题。

基金项目：国家部委基金资助项目

作者简介：石长振(1976－)，男，助理研究员，主研方向：计算机体系结构，实时信号处理；杨 雪，硕士研究生；王贞松，研究员、博士生导师

收稿日期：2011-07-08 **E-mail**：shicz@189.cn

3.1　处理器结构

硬件结构实现 FFT 的常用形式有 4 种：递归结构，流水线结构，并行迭代结构和全并行结构[4]。递归结构的主要好处在于所占硬件资源少，但是由于只有一个运算单元，运算的时间较长。流水线结构一般在 FFT 实现的每一级均采用一个运算单元，前一级运算结果直接用于下一级运算而无需等到本级运算全部完成，因此，可提高运算速度。并行迭代结构是指在每一级蝶形运算中采用多个蝶形单元并行处理，这样可以成倍提高处理能力，但是对数据存取带宽要求很高。全并行结构一般是指在 FFT 每一级根据 FFT 的点数设置呈正比的运算单元，该结构虽然速度快，但资源消耗过大。

3.2　存储器的组织

并行运算是提高 FFT 速度一种有效解决方法，如采用多个蝶算单元并行处理。随着 FFT 处理器的处理并行度的提高，数据访问速度成为系统的瓶颈，需要提高数据存取的并行性。如何将操作数无冲突的组织到多个存储体中是关键。

3.3　旋转因子生成及实现

在实时处理中旋转因子生成方式最常用的有查表方式和坐标旋转数字计算机(Coordinated Rotation Digital Computer, CORDIC)方式[5]。查表方式是提前将旋转因子计算好存放在 ROM 中，在复数乘运算时读取。这种方式比较简单，缺点是当 FFT 的点数较大和精度要求较高时，ROM 容量会很大。CORDIC 算法是基于向量旋转的迭代算法，是一种广义上逐次逼近的数值计算方法，把复乘转化为移位相加运算，不仅能减小芯片占用面积，还能提高单次蝶算速度。

3.4　数据表示格式

FFT 处理器通常采用定点、浮点和块浮点 3 种数据表示格式。定点数运算电路简单，但由于小数点位置固定、动态范围小、易溢出。浮点数运算动态范围大，不会发生溢出现象，但电路较复杂，消耗较多资源。块浮点介于浮点和定点之间，块浮点数格式是指一个处理的数据段共用一个指数，增加了数据的动态范围。采用定点运算和块浮点 2 种数据格式都要采取一定的防溢出处理机制。

4　FFT 处理器的结构设计

本文设计的高性能 FFT 处理器结构如图 1 所示，主要包括蝶形运算模块、存储单元模块及地址生成模块、旋转因子及实现模块、扩展检测与移位控制模块、数据输入输出模块。

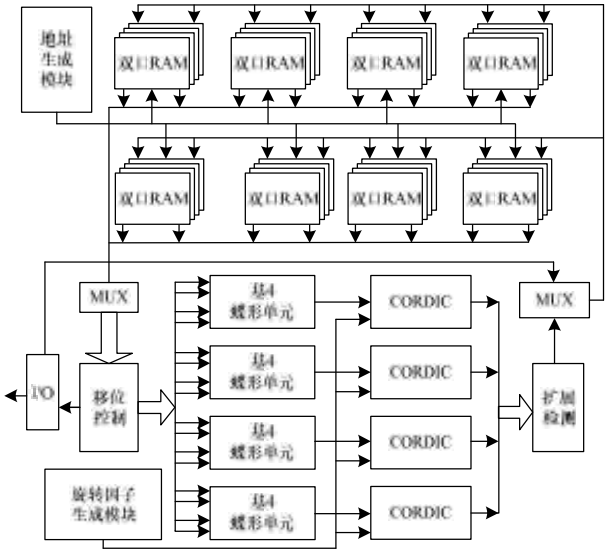

图 1　FFT 处理器结构框图

为了提高 FFT 处理器的处理性能，采用 4 个基 4 的蝶形并行处理。根据处理数据对精度的要求和存储资源的情况，数据采用块浮点格式。由于 FFT 处理器的长度较长，而且每个周期需要 16 个旋转因子参与计算，因此用 CORDIC 完成角度旋转。由于采用流水线结构，每个周期要从存储器中读取 16 个数据作为蝶形单元的输入，同时将 16 个运算结果保存到存储器中，因此将存储器配置成 16 个双端口 RAM 实现同时读写。4 个蝶算单元并行处理缩短了运算时间，使 FFT 运算时间和从存储器中输入输出操作数的时间相当，因此本设计采用 2 个存储器组，一个用于 FFT 运算，另外一个用于数据的输入输出，2 个存储器组以乒乓模式工作。

4.1　蝶形加法单元

蝶形运算单元作为 FFT 处理器的核心，其运算速度直接决定整个 FFT 处理器的速度。蝶形加法运算的数学模型见式(2)，根据该式设计的蝶形运算单元结构如图 2 所示。

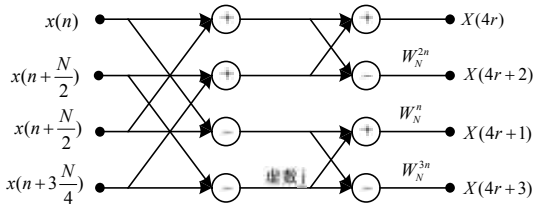

图 2　蝶形运算单元结构

为了提高运算速度，蝶形单元设计采用并行和流水线技术。由于采用 4 个蝶形单元并行，因此处理速度为单个蝶形单元的 4 倍。其中每个蝶形单元采用流水线技术设计，每个蝶形需要 8 个复数加法器(或减法器)。蝶形运算单元启动后每个周期处理 4 个复数点数据。即每个周期需要 4 个数据输入到蝶形单元，同时输出 4 个蝶形运算结果。4 个蝶形单元共 16 个数据，对于 N 点的 FFT，则每一级蝶形运算约需 $N/16$ 个时钟周期。

4.2　存储器地址生成模块

本文设计的 FFT 处理器采用 4 个蝶形运算并行处理来提高 FFT 运算的速度。4 个蝶形运算并行执行，在同一个时钟周期要完成 16 个数据同时读取和写存。因此这里将 FFT 处理器内部的存储器例化为双口 RAM 的形式，每个存储器组配置为 16 个双口 RAM，每个周期可同时写入和读出 16 个数据。通过采用改进的无冲突操作数地址变换方式，使得每级 FFT 运算中并行的 4 个蝶形单元所需的 16 个操作数在存储上不冲突。

对于 $N=4^M$ 点的 FFT，$M=\log_4 N$。N 个操作数的地址编号记为 A，$A=0,1,\cdots,N$，A 的 4 进制表示为：

$$A=\sum_{i=2}^{M-1}4^i\cdot A_i+4\cdot A_1+A_0 \tag{3}$$

构造等式：

$$B=16\sum_{i=2}^{r-1}4^{i-2}A_i+4((\sum_{i=0}^{r-1}A_i)\bmod 4)+(\sum_{i=1}^{M-1}A_i)\bmod 4$$

$$a=\sum_{i=2}^{r-1}4^{i-2}\cdot A_i,\quad b_1=(\sum_{i=0}^{r-1}A_i)\bmod 4$$

$$b_0=(\sum_{i=1}^{r-1}A_i)\bmod 4,\quad b=4b_1+b_0 \tag{4}$$

则 $B=16a+b=16a+4b_1+b_0$，这里 A 和 B 有一对一的对应关系，这样就把操作数地址 A 映射到 2 维存储体。存储体分为 16 个，b 表示体号，a 示体内地址。对于式(3)和式(4)给出的 A 到 B 的映射，4 个基 4 并行蝶算过程中所需要的 16 个操作数，能够映射到 16 个不同的存储体中，运算过程中存取

操作数不存在冲突。

对于 $N=4^M$ 的基 4 运算第 m 级，$m=0,1,\cdots,M-1$，蝶形运算的地址可以表示为：

$$A=\sum_{i=M-m}^{M-1}4^iA_i+4^{M-m-1}\cdot A_{M-m-1}+\sum_{i=1}^{M-m-2}4^iA_i+A_0$$
$$A_i=0,1,2,3 \tag{5}$$

在同一个蝶形运算中，4 个数据的地址的不同体现在 A_{M-m-1} 上，4 组并行蝶形运算的数据上的不同体现在 A_0 上。第 m 级一个基 4 蝶算的 4 个数据地址映射后得到不同的 4 个 b_1。根据 $b_1=(b_0+A_0)\bmod 4$ 的关系，映射到同一个 b_1 的 4 个数，所对应的 b_0 一定为 4 个不同值，如果相同则为同一个数。这说明这种映射方式能使参与运算的 16 个操作数处于不同的存储体中。

存储体的选择值 b 可以采用半加器来完成。由于采用四进制的格式表示，A_i 为 2 位。存储体选择单元如图 3 所示，其中，加法器为 2 位半加器，将 b_1 左移 2 位加上 b_0 得到 b 值，a 值可以由地址值 A 直接右移 4 位得到。

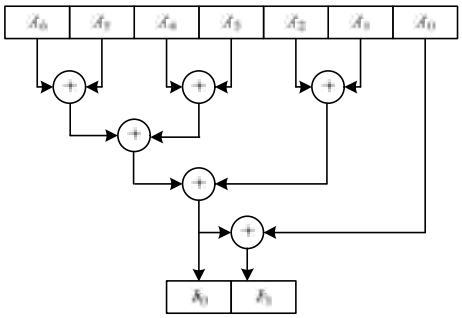

图 3 存储体选择单元

4.3 旋转因子生成模块

由图 2 可知，蝶形运算要用到大量的复数乘法运算，而乘法运算占用的面积大。另外旋转因子也会占用大量 ROM，而且每个周期要读取多个旋转因子值也会使结构变得复杂。在本文中采用 CORDIC 算法。相比查表法，CORDIC 算法具有不依赖三角函数查找表的大小、支持高点数的 FFT 处理以及不需要乘法器的特点。

CORDIC 的原理为：当把向量 (X,Y) 旋转 θ 度，得到新的向量 (X',Y') 后，根据三角函数运算可以表述为下式：

$$X'=X\cos\theta-Y\sin\theta=\cos\theta(X-Y\tan\theta)$$
$$Y'=Y\cos\theta+X\sin\theta=\cos\theta(Y+X\tan\theta) \tag{6}$$

如果取 $\tan(\theta)=\pm2^{-i}$，就可将式(6)中的运算简化为一简单移位操作。事实上对于一个 $\theta(0\leqslant\theta$ ，可以用 $\theta=\sum_{i=0}^{\infty}S_i\cdot\arctan(2^{-i})$ 去逼近，将 θ 角度的旋转分解成一系列小角度的旋转。FFT 算法中的复数乘法用 CORDIC 表示如下：

$$X_{i+1}=X_i-S_i2^{-i}Y_i,Y_{i+1}=Y_i+S_i2^{-i}X_i$$
$$Z_{i+1}=Z_i-S_i\arctan(2^{-i})$$
$$S_i=\begin{cases}-1 & Z_i<0\\ 1 & Z_i\geqslant 0\end{cases} \tag{7}$$

其中，X_i,Y_i 为第 i 次旋转的输入值，根据角度 Z_n 的符号分别加上或减去 Y_i 和 X_i 的右移 i 位的值就可得到 i 级的输出 X_{i+1} 和 Y_{i+1}。

在本文的设计中，CORDIC 运算单元的处理分粗粒度和细粒度 2 步进行，首先粗粒度旋转将输入数据通过简单的取反或交换，完成旋转因子从区间[0, 2π)到收敛区间[0, π/2)的转换，θ 为旋转角度，X、Y 为输入数据的实部和虚部，X'、Y' 为粗粒度旋转后数据的实部和虚部，粗粒度旋转关系如表 1 所示。

表 1 粗粒度旋转关系

角度范围	X'	Y'	θ'
$0\leqslant\theta<$	X	Y	θ
π $\leqslant\theta<$	Y	$-X$	$\theta-\pi$
$\pi\leqslant\theta<$	$-X$	$-Y$	$\theta-\pi$
3 $\leqslant\theta<$	$-Y$	X	$\theta-3$

经过粗粒度旋转后的数据和规格化的角度值输入到移位运算部件再进行细粒度旋转。由于数据经过叠加运算后是用 19 位表示的，细粒度旋转进行 19 次即达到收敛状态，因此 $\arctan(2^{-i})$ 的值的查找表 i 的范围为 0~19 即可。每一级移位累加部件的结构如图 4 所示。

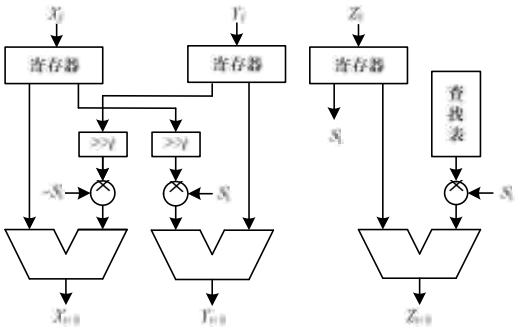

图 4 CORDIC 移位累加部件结构

4.4 溢出检测与移位控制模块

从设计复杂性、运算精度、运算速度和占用逻辑资源折中考虑，处理数据采用块浮点算法。在 FFT 处理器中采用块浮点数的格式是指每个处理的一组数据共用指数位。根据式(2)可知：

$$|X(\cdot)|\leqslant|(x(A)+x(B)+x(C)+x(D)|$$
$$|x(A)|+|x(B)|+|x(C)|+|x(D)|\leqslant$$
$$4\max\{|x(A)|,|x(B)|,|x(C)|,|x(D)|\} \tag{8}$$

因此，对于基 4 蝶算单元当输入为 16 位有符号数，蝶形运算单元的输出的最极端情况不会超出 19 位有符号数的表示范围。

在处理过程中检测每一级蝶算结果的最大扩展值。根据前一级蝶算的最大扩展值决定下一级蝶算从双口 RAM 中读取数据的移位值，保证移位后每个数据的高 4 位为符号位，使得蝶算结果的扩展不会超出 19 位的范围。把所有级的移位值进行累加，由此可确定最后结果的指数位。蝶形加法单元输出数据的高 4 位的值与扩展位数关系如表 2 所示。

表 2 蝶形单元运算扩展情况

检测值		扩展情况
正	负	
0000	1111	无扩展
0001	1110	扩展 1 位
001X	110X	扩展 2 位
01XX	10XX	扩展 3 位

5 性能评测

本文设计的高性能并行 FFT 处理器在 Xilinx 的 FPGA 器件 XC4VFX140-11 上实现，已经应用于某合成孔径雷达实时成像器中，满足系统设计指标。在实际的应用中，系统需要处理 16 384 点和 65 536 点 FFT 变换，为了节约资源，其存

(下转第 247 页)