基于 FPGA 的 CORDIC 算法的改进及实现

习 玉[1],宋泽琳[2],马令坤[2]
(1.中国人民解放军第四三二八工厂,山西 长治 046011;
2.陕西科技大学 电气与信息工程学院,陕西 西安 710021)

摘 要:介绍了 CORDIC 算法的基本原理,分析了其具体计算方法。针对利用 CORDIC 流水线实现 FFT 蝶形运算耗费资源多的问题,依据 CORDIC 计算迭代系数的方法改进了 CORDIC 流水线的结构形式,使其适应 FFT 算法。选用 ALTERA 公司 CycloneII 系列的 EP2C35F672C6 来实现整个 FFT 处理器,并对设计进行了时序仿真和硬件仿真。通过比较,计算结果与设计基本一致。

关键词:CORDIC;FFT;Cyclone II;流水线
中图分类号:TP274 **文献标识码**:A **文章编号**:1674-7720(2011)16-0078-04

The improvement and implementation of CORDIC algorithm based on FPGA

Diao Yu[1], Song Zelin[2], Ma Lingkun[2]
(1.The 4328th factory of Chinese People's Liberation Army, Changzhi 046011, China;
2.School of Electric and Communication Engineering, Shanxi University of Science and Technology, Xi'an 710021, China)

Abstract: Introduce the basic principle of CORDIC algorithm, and analyse the calculation method of CORDIC algorithm. For use of CORDIC streamline realize FFT wing operation cost more resources, according to CORDIC iterative method of calculating coefficient to improve CORDIC pipeline structure form, in order to accommodate FFT algorithm. Choose ALTERA company CycloneII series of EP2C35F672C6 to realize the whole FFT processor. And timing simulation and hardware emulation are designed. Through comparing the both, the calculation results are basically the same.

Key words: CORDIC; FFT; Cyclone II; streamline

坐标旋转计算机 CORDIC(The Coordinate Rotational Digital Computer)算法是一种用于计算一些常用的基本运算函数和算术操作的循环迭代算法。其基本思想是用一系列与运算基数相关的角度的不断偏摆来逼近所需旋转的角度,从广义上讲它是一个数值型计算逼近的方法。由于这些固定的角度与计算基数有关,运算只有移位和加/减。若用传统的乘、除等计算方法,需要占用大量的硬件资源,甚至算法是难以实现的,这样就不能满足设计者的要求。CORDIC 算法正是由此产生的,它仅在硬件电路上用到了移位和加/减,大大节约了硬件资源,使得这些算法在硬件上可以得到较好地实现,从而满足设计者的要求。根据它的迭代原理,CORDIC 单元可以用流水线结构表示,使向量旋转并行处理,大大加快了蝶形运算的速度[1]。但是 CORDIC 运算单元的多级迭代也占用了大量的芯片资源,尤其是在使用多个蝶形进行 FFT 处理时,使用的资源是非常巨大的,为了尽量降低资源占用,对 CORDIC 流水线进行了结构上的改进。

1 CORDIC 算法原理

1959 年,VOLDER 开发了一类计算三角函数、双曲函数的算法,其中包括指数和对数运算。此算法的基本思想是用一系列固定的与运算基数相关的角度不断偏摆从而逼近所需的角度。从广义上讲它是提供一个数值计算的逼近方法。由于这些固定的角度只与计算基数有关,运算只有移位和加减。CORDIC 算法虽然可以实现很多基本函数,但一开始并没有引起人们很大的注意,只是 CAGGETT 用它来实现二进制和十进制的转换。整个 60 年代没什么进展,直到 1971 年 WALTHER 提出统一的 CORDIC 算法,加上 VLSI 技术的不断发展,CORDIC 算法才越来越受到人们的重视,并展示出广泛的应用前景[2]。CORDIC 算法已被广泛用作现代信号处理各种算

法实现中的运算单元，诸如离散傅里叶变换、矩阵的分解、矩阵特征值的求解、场分解、线性预测参数的求解等。

如图1所示，一对直角坐标轴顺时针旋转角度A（点M相对于坐标轴逆时针旋转），点M的坐标从(x_0,y_0)变为(x,y)[3-6]。

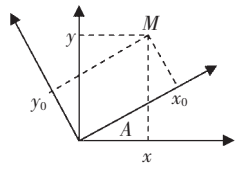

图1 坐标旋转

旋转方程为：

$$\begin{bmatrix}x\\y\end{bmatrix}=\begin{bmatrix}\cos A & -\sin A\\ \sin A & \cos A\end{bmatrix}\begin{bmatrix}x(0)\\y(0)\end{bmatrix} \tag{1}$$

旋转角度A可通过若干步实现（对绝大多数角都要无穷多步才能无误差地实现），每一步旋转一个较小的角，完成旋转的一部分。单独的一步旋转为：

$$\begin{bmatrix}X_{n+1}\\Y_{n+1}\end{bmatrix}=\cos A_n\begin{pmatrix}1 & -\tan A_n\\ \tan A_n & 1\end{pmatrix}\begin{bmatrix}X_n\\Y_n\end{bmatrix} \tag{2}$$

显然，上述角A是有符号的，所以A_n之和为A，但方便起见，A_n的大小和符号分开考虑。θ_n表示大小，S_n表示符号，则A_n可表示为S_nA_n。每一步要旋转的角可这样选取：

$$\theta_n=\arctan\left(\frac{1}{2^n}\right) \tag{3}$$

所有旋转的角度A_n之和即为所需旋转的角度A：

$$\sum_{n=0}^{\infty}A_n=\sum_{n=0}^{\infty}S_n\theta_n=A \tag{4}$$

其中，S_n为$\{-1,1\}$。$S_n=-1$表示第n步坐标轴要逆时针旋转，$S_n=+1$则表示顺时针旋转。当$\tan A_n$满足$\tan A_n=S_n2^{-n}$时，结合上式，每一步的旋转方程变为：

$$\begin{bmatrix}X_{n+1}\\Y_{n+1}\end{bmatrix}=\cos A_n\begin{pmatrix}1 & -S_n2^{-n}\\ S_n2^{-n} & 1\end{pmatrix}\begin{bmatrix}X_n\\Y_n\end{bmatrix} \tag{5}$$

上式除$\cos A_n$外，整个算法就只是移位和加法运算了。由式(5)知：

$$\cos A_n=\frac{1}{1+S_n^2\,2^{-2n}} \tag{6}$$

因$S_n\neq0$（实际上S_n是会等于零的，这点将在后面讨论），故：

$$\cos A_n=\frac{1}{1+2^{-2n}} \tag{7}$$

显然，$\cos A_n$是一个与A无关的因子，$\cos A_n$的乘积可看作一个常数$P(N)$，令$K(N)$为这个常数的倒数，有：

$$K(N)=\frac{1}{P(N)}=\prod_{n=0}^{N-1}\frac{1}{\cos A_n}=\prod_{n=0}^{N-1}(1+2^{-2n}) \tag{8}$$

当$N=13$时，$P(N)\approx0.607\,253$。既然$\cos A_n$已经提出，则式(5)可以简化为：

$$\begin{cases}X_{n+1}=X_n-S_nY_n\times2^{-n}\\ Y_{n+1}=X_n+S_nX_n\times2^{-n}\end{cases} \tag{9}$$

在完成所有N次迭代之后，得到X_N、Y_N

$$\begin{cases}X_N=K(N)(X_0\cos(Z_0)-Y_0\sin(Z_0))\\ Y_N=K(N)(Y_0\cos(Z_0)+X_0\sin(Z_0))\end{cases} \tag{10}$$

于是(10)式所要的旋转的坐标值为：

$$\begin{cases}X=X_N\times P(N)\\ Y=Y_N\times P(N)\end{cases} \tag{11}$$

令$Z_0=\theta$，$Y_0=(P_{re}-Q_{re})$，$X_0=(P_{im}-Q_{im})$得：

$$\begin{cases}X_k=K(N)((P_{im}-Q_{im})\cos(\theta)-(P_{re}-Q_{re})\sin(\theta))\\ Y_k=K(N)((P_{re}-Q_{re})\cos(\theta)+(P_{im}-Q_{im})\sin(\theta))\end{cases} \tag{12}$$

式(12)中K_1为旋转增益的极限值，因为：

$$K(N)=\prod_{n=0}^{N-1}\frac{1}{\cos A_n}=\prod_{n=0}^{N-1}(1+2^{-2n}) \tag{13}$$

通过求极限可以求出$K_1\approx1.646\,759>1$，为了防止数据溢出，利用CORDIC算法的计算结果右移一位（即乘以0.5）。

$$\begin{cases}R_{re}=K(N)'(P_{re}-Q_{re})\\ R_{im}=K(N)'(P_{im}+Q_{im})\\ S_{re}=K(N)'((P_{re}-Q_{re})\cos(\theta)+(P_{im}-Q_{im})\sin(\theta))\\ S_{im}=-K(N)'((P_{re}-Q_{re})\sin(\theta)+(P_{im}-Q_{im})\cos(\theta))\end{cases} \tag{14}$$

其中$K(N)'=K(N)/2$。

2 CORDIC的流水线结构及其改进

采用流水线结构，能够在执行进程的同时输入数据，从而极大地提高程序的运行效率。在该结构中，每一个移位都是固定的深度，而且旋转角度集的各个值作为常数直接联到了累加器上，不需要存储空间和读取时间。整个CORDIC简化为加减法器的直接相连，而且在大多数器件的每个单元中都有寄存器，便于采用流水线技术。如图2所示。

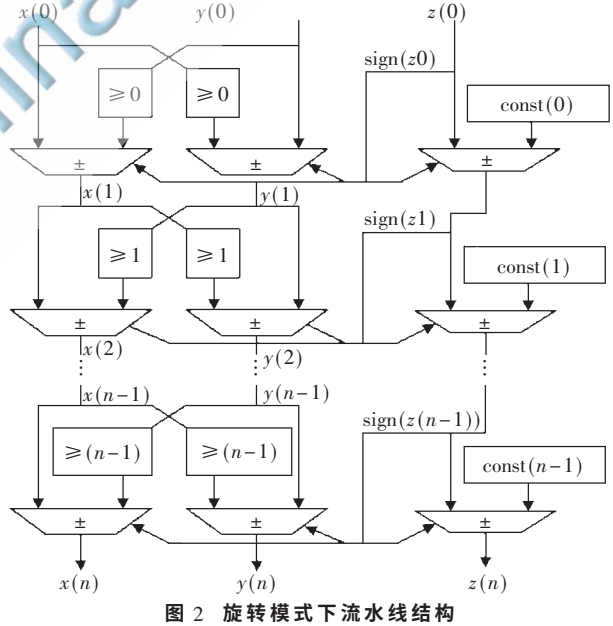

图2 旋转模式下流水线结构

为了满足FFT在速度上的要求，CORDIC可以设计成流水线的形式。将需要旋转的角度加到Z_0数据通道，通过Z_1与固定角度相加减产生所取的值。需要旋转的(x_0,y_0)向量在各级迭代中旋转方向。Z_n通过多次迭代，趋近于零，向量旋转到相应角度。如果在FFT的蝶形单

元中用 CORDIC 代替复乘单元，只需要将数据的实部和虚部分别加到 x_0 和 y_0 通道，将复乘系数作为角度从 Z_0 处输入，达到了乘以的目的。实际应用中将 FFT 使用的角度值存储在 ROM 中，由地址发生器控制，在计算时将相应的旋转角度读入 CORDIC 中即可。使用 CORDIC 算法可以方便快捷地计算 FFT 蝶形，但是由于迭代次数多，导致耗费资源也比较多。

将 CORDIC 流水线形式进行改进，如图 3 所示，需要旋转的向量的实部和虚部分别加到 X_0 和 Y_0 数据通道上，系数输入到 D 触发器中与向量保持同步，用来控制向量在各级迭代中旋转的方向。向量经多次迭代旋转到相应角度[7-8]。

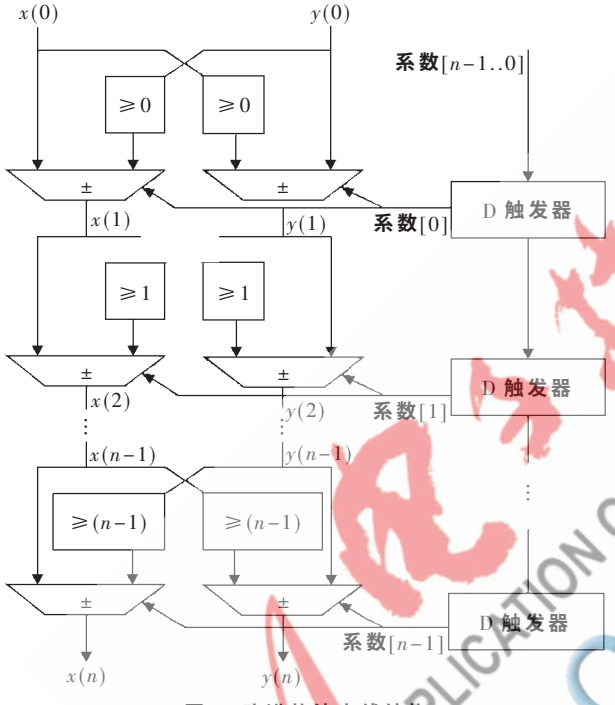

图 3 改进的流水线结构

3 CORDIC 的旋转系数

按照改进后 CORDIC 的结构，需要事先求出 CORDIC 的旋转系数。根据 CORDIC 算法的迭代原理以及此结构的具体情况，使用 MATLAB 语言编写程序求出各级旋转系数，存在 ROM 中。时序仿真结果如图 4 所示。

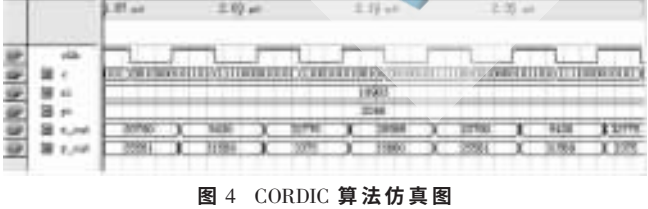

图 4 CORDIC 算法仿真图

图 4 是利用改进的 CORDIC 算法计算的结果。从仿真图可以看出：当输入不同的迭代系数 C 时，就可以计算出不同的结果。

4 FFT 处理器的仿真和测试

在完成了 FFT 的整体设计和具体模块设计之后，选用 ALTERA 公司 CycloneII 系列的 EP2C35F672C6 来实现整个 FFT 处理器，并对设计进行了时序仿真和硬件仿真[9-10]。

(1)直流信号的测试

使用 MATLAB 产生一个直流信号：i=1:256，$x(i)$=50。输出计算结果和 MATLAB 的计算结果比较如图 5 所示，MATLAB 计算的直流信号 FFT 结果虚部全为 0，实部只有一个值即：$X(0)$=14 000；FFT 处理器的计算结果也一样：$X(0)$=6 000，如图 6 所示。因为运用 CORDIC 算法一方面使运算每一级的结果扩大了 1.65 倍，另一方面为了减少误差和防止溢出，在 FFT 处理器的各级都进行了 1/2 的截尾处理。但是这样的处理不会影响频谱的显示。

图 5 MATLAB 计算结果

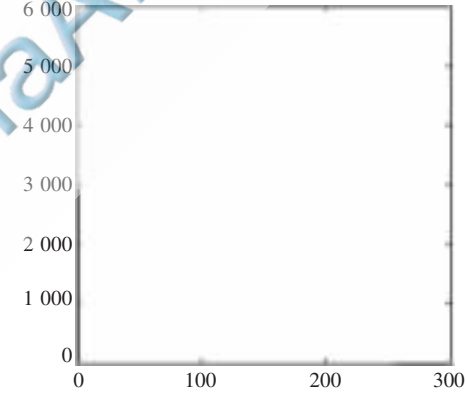

图 6 FFT 处理结果

(2)三角波信号的测试

仍然使用 MATLAB：i=1:128，$x(i)$=i；(28+i)=128，产生一个三角波信号。MATLAB 的计算结果见图 7 和图 8，FFT 处理器中输出计算结果见图 9 和图 10。由于 FFT 处理器对各级进行了放大和截尾处理，所以为了便于比较，将 FFT 处理的结果进行了还原(即除以 1.65^3/16)，通过图对实部和虚部进行比较，可以证明计算结果基本一致。

采用 CORDIC 算法以较少的资源实现了快速乘法器，通过增加 CORDIC 运算单元的处理位数，减少了算法的误差。设计使用 16 位宽，CORDIC 单元的误差不大于 2^{-14}，有效位数为 13 位。FFT 有限字长效应长生量化

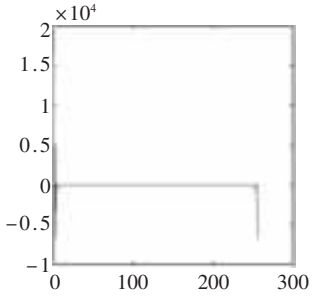

图 7 MATLAB 实部计算结果

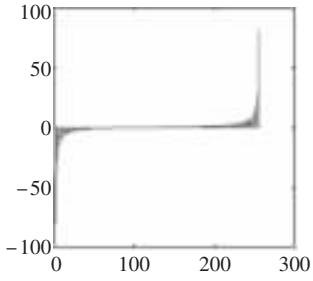

图 8 MATLAB 虚部计算结果

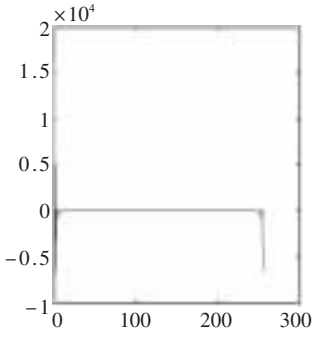

图 9 FFT 实部计算结果

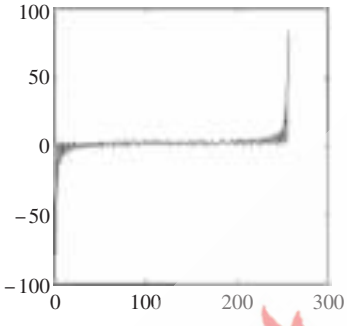

图 10 FFT 虚部计算结果

误差，主要来自输入量化误差、系统量化误差和运算量化误差。从仿真实验可以看出，达到了预期目的。

参考文献

[1] 胡国荣，孙允恭.CORDIC 算法及其应用[J].信号处理，1991(12)：229-242.

[2] VOLDER J E.The CORDIC trigonometric computing technique[J].IRE TlectronComputers，1959(9)：330-334.

[3] 于效宇，宋立新，刘艳.CORDIC 流水线结构在 FFT 设计中的改进[J].哈尔滨理工大学学报，2005，10(1)：55-57.

[4] 韩芳，初建朋，赖宗声.一种 CORDIC 算法的精度分析及其在 FFT 中的应用[J].微电子学与计算机，2004，7(7)：14-16.

[5] 于效宇.基于 FPGA 的 FFT 处理器的实现[D].哈尔滨：哈尔滨理工大学，2005.

[6] 李成诗，初建朋，李新兵.基于 CORDIC 的一种高速实时定点 FFT 的 FPGA 实现[J].微电子学与计算机，2004，21(4)：88-91.

[7] 杨宏，李国辉，刘立新.基于 FPGA 的 CORDIC 算法的实现[J].西安邮电学院学报，2008，3(1)：75-77.

[8] 杨宇，毛志刚，来逢昌.一种改进的流水线 CORDIC 算法结构[J].微处理机，2006(4)：10-13.

[9] 张俊涛，王红仓.基于 FPGA 的 CORDIC 算法通用 IP 核设计[J].微计算机信息，2008，24(7-3)：238-240.

[10] 刘桂华，傅佑麟，严平.FFT 实时谱分析系统的 FPGA 设计和实现[J].集成电路应用，2005(4)：65-67.

(收稿日期：2011-02-27)

作者简介：

习玉，男，1981年生，助理工程师，主要研究方向：电子信息和计量检测。

宋泽琳，男，1984年生，硕士研究生，主要研究方向：数字信号处理与 FPGA 技术。

马令坤，男，1967年生，博士，教授，主要研究方向：信号与信息处理。