基于 FPGA 的 apFFT 算法实现 *

孙 林，黄晓红，蔡江利

（河北理工大学，河北 唐山 063000）

摘 要：全相位频谱分析(apFFT)是传统 FFT 的一种改进算法，能改善 FFT 的栅栏效应和截断效应，具有频谱泄露少、相位不变的特性。介绍采用 FPGA 器件实现 apFFT 算法，精度高于模拟式测量，并且适用性强、成本低，所得到的 QuratusII 仿真结果与 Matlab 软件仿真结果一致。

关键词：现场可编程门阵列；全相位快速傅里叶变换；频谱泄露；相位不变性

中图分类号：TN77 **文献标识码**：A **文章编号**：1674-7720(2010)18-0021-04

Implementation of the apFFT algorithm based on FPGA

SUN Lin, HUANG Xiao Hong, CAI Jiang Li

(Hebei Polytechnic University, Tangshan 063000, China)

Abstract: All Phase spectrum analysis(apFFT) is an improved algorithm of the traditional fast Fourier transform, it can improve the fence effect and truncation effect, and has excellent performance in suppressing spectral leakage and the property of phase invariant. This paper introduces the use of FPGA devices to implement apFFT algorithm which is composed of three parts: adress generate modular, data memory modular and FFT modular. The precision is higher than analogue measurement, the simulation result of QuratusII is similar to the simulation result of Matlab.

Key words: FPGA; apFFT; spectral leakage; phase invariant

全相位频谱分析 apFFT(all phase FFT)是近几年提出的频谱分析方法，该方法具有比传统 FFT 更优良的频谱抑制性能；具有“相位不变性”，该性质意味着即使是在“不同步采样的情况”，无需借助任何附加的校正措施即可精确提取出信号相位信息，因而在相位计的设计、激光测距、雷达等多个应用领域具有较高的实用价值。对于 apFFT 的理论研究已有一些论文，但是硬件实现此算法的研究还很少见。本文对此新型频谱分析算法用 FPGA 来实现，并进行了仿真和分析。

目前，通常采用两种途径通过硬件方式实现 FFT 算法：(1)使用 DSP 器件实现；(2)通过 FPGA 器件实现。一般来说，DSP 器件多用于数字信号处理领域，而且开发过程相对简单，易于实现，但速度较慢，无法完成对速度要求较高的算法。而 FPGA 器件由于内部嵌入了硬件乘法器、可编程寄存器和 M4K 内存块，在速度上具有明显的优势。考虑到 apFFT 对速度要求较高，故选用 FPGA 器件作为硬件开发平台。

1 全相位频谱分析

apFFT 理论推导详见参考文献[1,2]，本文以 FFT 点数 $N=3$ 点为例简化此频谱分析图如图 1。其中的卷积窗 $wc=[wc(-N+1),\cdots, wc(-1), wc(0), wc(1),\cdots, wc(N-1)]$ 由两个长度为 N 的对称窗卷积而来，用这个长为 $(2N-1)$ 的卷积窗 wc 对输入样本加窗后，再将间隔为 N 的两数据平移相加生成 N 个数据 $y(n)$ $(n=0,1,\cdots,N-1)$，最后对 $y(n)$ 进行 FFT 即得谱分析结果。

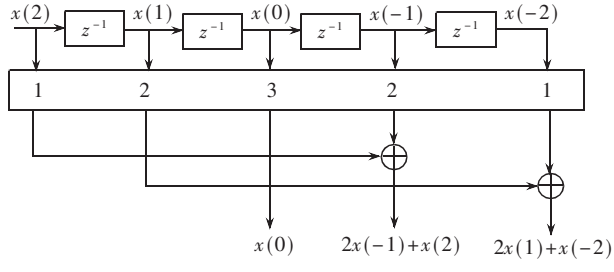

图 1 全相位频谱分析框图

2 软硬件简介

在 FPGA 开发过程中，常用的是 VHDL 和 Verilog HDL 语言。VHDL 语言比较适合做大型的系统级设计，而 Verilog HDL 则适合逻辑级、门级设计。所以，考虑到两种语言各自特点，本文选用 VHDL 语言完成设计。

* 基金项目：河北省自然科学基金资助(F2008000415)

采用 FPGA 实现 apFFT 算法，对硬件资源要求较高，故开发芯片选择 Altera 公司的 EP2C35F672C8。该芯片内部包含有 33 216 个逻辑单元，105 个 M4K RAM 模块，以及 18 bit×18 bit 嵌入式乘法器。

软件选用 Altera 公司开发的 QuartusII 平台。该软件提供了丰富的开发工具供用户使用，可以完成代码输入、编译、仿真以及下载到芯片的全部功能。

3 apFFT 模块设计

本文所设计的 apFFT 模块由三部分构成，分别为：地址发生模块、数据存储模块和 FFT 运算模块。各个模块间的关系如图 2 所示。

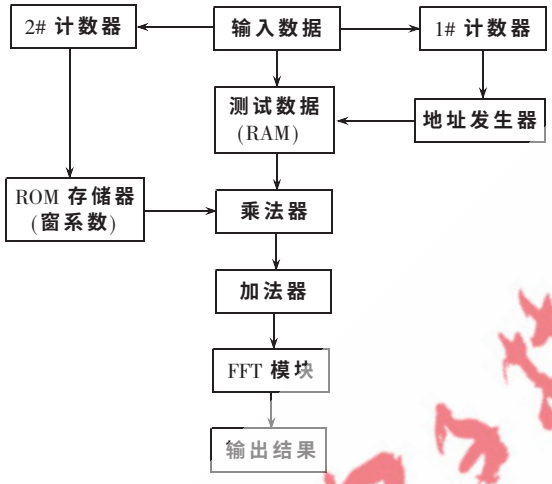

图 2 apFFT 硬件模块图

3.1 地址发生模块

为了保证测试数据能够完整无误地输入到 EP2C35F672C8，需要选择合适的存储地址来保存数据。本文以做 8 点 FFT 为例，所涉及的所有数据总线宽度均为 8 bit，序列长度取 15 bit。为了保证 15 bit 的存储数据都能够及时存储到寄存器中，需要至少 4 bit 的地址总线才能满足设计需求。

地址发生模块的结构体部分程序如图 3 所示。

```
10 ARCHITECTURE BEHAV OF count IS
11   signal CQI : STD_LOGIC_VECTOR(3 DOWNTO 0);
12 BEGIN
13   P_REG: PROCESS(CLK,RST,ENA)
14     BEGIN
15       IF RST='1' THEN  CQI<= "0000";
16        ELSIF CLK'EVENT AND CLK= '1' THEN
17            IF ENA='1' THEN
18                IF CQI<"1110" THEN CQI<=CQI+1;
19                ELSE  CQI<="0000";
20                END IF;
21            END IF;
22        END IF;
23     END PROCESS P_REG;
24       OUTY <= CQI;
25 END BEHAV;
```

图 3 地址发生模块

编译通过之后，得到的 Symbol 文件如图 4 所示。

3.2 存储器设计

3.2.1 数据存储器

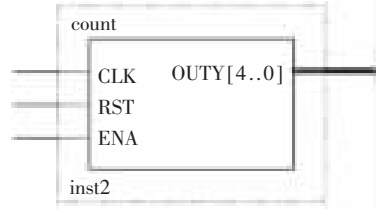

图 4 地址发生模块

由于输入数据由总线宽度为 8 bit 的实部和虚部两部分构成，所以需要双口 RAM 对数据进行存储。这样设计的优势在于能够很好地将输入数据按其顺序输入到 FFT 核当中，而且方便对不同地址的数据进行实时调用。

在对模块设计过程中，可以直接调用 QuratusII 里的 MegaWizard Plus-In Manager 工具定制 RAM。定制过程中，需要对 RAM 的控制线、地址线和数据线进行选择，这里选择地址线宽度为 4 bit，输入、输出数据线宽度为 8 bit，读取时钟信号 rdclock 同时控制读地址和 RAM 的输出。

在该存储器中，时钟信号 wrclock 和 rdclock 分别控制随机存储器的写、读状态，均为高电平有效。同时，wrclock 和 rdclock 作为写、读数据的地址发生器工作。即对 wrclock 和 rdclock 的上升沿进行计数，并根据计数结果产生相应的地址位。生成的数据存储器如图 5 所示。

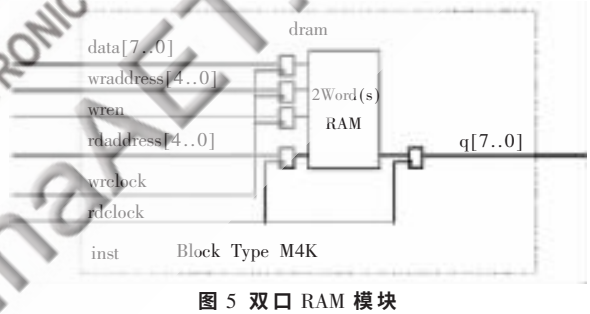

图 5 双口 RAM 模块

需要注意的是，由于 EP2C35F672C8 属于 CycloneII 器件，在调用 RAM 模块时，必须做如下设置：选择 Assignments→Setting 命令，在弹出的对话框中选择 Analysis & Synthesis Settings 下的 Default Parameters 选项，并在该选项的 Name 文本框中输入 CYCLONEII_SAFE_WRITE；在 Default Setting 文本框中输入 VERIFIED_SAFE，并分别点击 Add 和 OK 按钮关闭 Settings 窗口。这样才能在最后综合以及仿真时，得到正确的结果。

3.2.2 窗函数存储器

apFFT 相比传统 FFT，最大的区别在于其 FFT 运算模块输入数据是经过预处理的数据，而非采集电路直接采集到的数据。在进行数据预处理的过程中，非常重要的部分就是窗系数的选择。以 N=8 点 FFT 为例，全相位输入数据是 $2N-1$=15 个，采集余弦函数的 15 个数据为：−0.173 65，−0.990 27，−0.438 37，0.719 34，0.882 95，−0.173 65，−0.990 27，−0.438 37，0.719 34，0.882 95，−0.173 65，−0.990 27，−0.438 37，0.719 34，0.882 95。

按照参考文献[1,2]选择的窗函数为：0.013 684，0.096 665，

0.346 18, 0.846 66, 1.590 8, 2.431 7, 3.111 8, 3.375, 3.111 8, 2.431 7, 1.590 8, 0.846 66, 0.346 18, 0.096 665, 0.013 684。将窗函数转换为 8 bit 二进制的形式，并存储到只读存储器当中以方便运算。如图 6 所示。

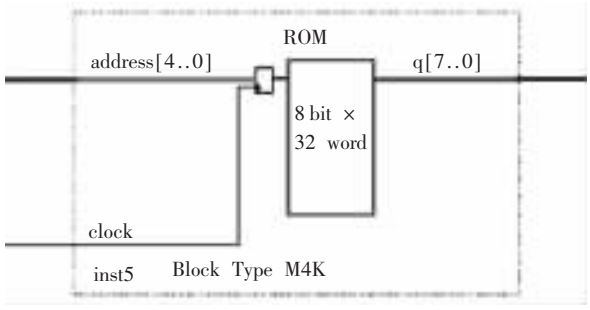

图 6 窗系数 ROM 模块

将输入数据经加窗处理并叠加后，在 matlab 中得到的结果为：-1.479 5, 2.236 1, 2.051 3, -0.428 0, -0.229 4, 1.252 9, -0.352 7, -3.069 4。此时，在 QuartusII 中得到的结果为 -1.236 8, 2.339 7, 2.004 9, -0.4029, -0.1803, 1.118 6, -0.348 5, -2.985 6。可以看出两者有一定的误差，其原因是在 QuartusII 中得到的结果是以二进制形式表示，在转换过程中存在一定的量化误差。

3.2.3 量化误差

在 FPGA 中实现算法，一般要对十进制的小数进行量化，即将十进制的小数转换为二进制数，并运用二进制补码表示，兼顾舍入误差，由于将十进制小数转换为二进制比较繁琐，现编写 matlab 程序进行转换：下面是将整数部分不为零的十进制的小数转换为二进制小数的部分程序：

```
function [num,numint,numf]=dectobin1(innum,N);
%clc;clear;close all;
%十进制数转换为二进制数
%输入为十进制数 innum,以及小数部分的位数 N
%输出为三个参数 num,numint,numf
%num 为输出的二进制形式
%numint 为整数部分的二进制表达式
%numf 为小数部分的二进制表达式
sep=5;%整数和小数部分的分隔符
if(mod(innum,1)==0)%判断输入是否为整数,mod 为取余函数
    numint=dec2bin(innum);
    numint=double(numint)-48;
    numf=zeros(1,N);
    num=[numint,sep,numf];
    return
end;
%输入为非整数的情况
nint=floor(innum);%整数部分
nf=innum-nint;%小数部分
res_nint=dec2bin(nint);
res_nint=double(res_nint)-48;
res_nf=dectobin(nf,N);
numint=res_nint;
numf=res_nf;
num=[numint,sep,numf];
```

在 FPGA 中只能进行定点运算，根据对系数的量化误差及有效字长效应，对加卷积窗的系数进行量化，所有的系数均采用二进制补码的形式表示，也就是采用有符号的八位二进制补码表示，在量化过程中，由于计算相对复杂，工作量比较大，如果采用手工计算来进行量化显然是不可取的，而且也容易出现错误，为此利用前面为量化误差编写的程序进行量化，这样大大减少了工作量，提高了工作效率。而且在设计中系数还具有线性相位的特性，利用这一特性更加减少计算的工作量。

对系数进行量化的数值如表 1 所示。

表 1 系数转换

系数	数值	量化值
$w(-7)$	0.013 684	3
$w(-6)$	0.096 665	12
$w(-5)$	0.346 18	44
$w(-4)$	0.846 66	108
$w(-3)$	1.590 8	50
$w(-2)$	2.431 7	77
$w(-1)$	3.111 8	99
$w(0)$	3.375 0	108
$w(1)$	3.111 8	99
$w(2)$	2.431 7	77
$w(3)$	1.590 8	50
$w(4)$	0.846 66	108
$w(5)$	0.346 18	44
$w(6)$	0.096 665	12
$w(7)$	0.013 684	3

3.3 FFT 运算模块

这里的 FFT 模块，可以通过两种方式得到。

第一种是自己编写一个 FFT 算法的子程序，编译通过后将该子程序打包成一个 Symbol 文件，并在最后的顶层文件中进行调用。这种方法的好处在于对 FFT 的算法能够很好表达，并根据需要进行灵活修改，缺点是开发周期较长，硬件资源利用率不是太高。

第二种设计方法是安装 Altera 公司提供的 IP 核，并对其进行相应的参数设定。这种开发方法的好处在于简单易用，并且能够很好利用硬件资源。缺点是由于该核包含知识产权，商用时需缴纳一定版权费用。考虑到本设计尚处于研究阶段，故选择后一种开发方式，也便于减少硬件资源的消耗。在使用 IP 核的过程中需要对 FFT 核的参数进行设置，过程分为三步：参数设定(Parameterize)，仿真设定(Set Up simulation)以及产生 FFT 核

(Generate)。

4 编译及仿真

对最终的顶层文件进行编译,并对其进行时序仿真。其仿真结果如图 7 所示。

在图 7 中,可看到最终仿真之后得到的波形情况,图中所有值均以二进制形式显示。

本设计所得到的硬件仿真结果与 Matlab 软件仿真得到的结果基本一致,说明 apFFT 的 FPGA 的可行性。下一步将对此设计进行改进,可以根据 FFT 点数实时地对 apFFT 模块进行参数化设置。

图 7 时序仿真波形图

最终设计的 FFT 模块使用了 2 755 个逻辑单元,仅占硬件资源的 8%。可见该设计的资源耗用与直接对数据进行 FFT 运算的资源耗用大体相当,apFFT 和 FFT 计算效率分别是 $N\log N+2N$ 和 $N\log N+N$。因为 apFFT 相对于传统的 FFT,虽然采样点数多了 $N-1$,但最终都用一个 N 阶 FFT 实现,而计算量主要体现在 FFT 中。在不增加 FFT 点数情况下,硬件资源耗用没有明显增加,但相对于传统的 FFT 可以降低频谱泄露,并且用 apFFT 测相位不用任何校正,所以在后续开发做频谱分析或者相位计时计算量会很小,有利于实时实现。

参考文献

[1] 王兆华, 侯正信, 苏飞. 全相位 FFT 频谱分析[J]. 通信学报, 2003, 24(11A): 16-19.

[2] 黄晓红,王兆华. 一种减少泄漏的新型谱估计方法[J]. 信号处理, 2007, 23(1): 144-147.

[3] 胡广书. 数字信号处理理论、算法与实现.第 2 版[M]. 北京:清华大学出版社, 2003.

[4] UWE M B. Digital signal processing with field programmable gate arrays[M].New York:Springer-Verlag Berlin Heidelberg, 2003.

[5] (美)阿森顿(Ashenden P J.).VHDL 设计指南.第二版[M].葛红,译.北京:机械工业出版社, 2005.

[6] 曾繁泰,陈美金.VHDL 程序设计[M]. 北京:清华大学出版社, 2007: 5-22.

(收稿日期:2010-05-21)

作者简介:

孙林,男,1981 年生,讲师, 硕士,主要研究方向:信号与信息处理技术及应用。

黄晓红,女,1973 年生,副教授,博士,主要研究方向:信号处理。