基于FPGA的通用FFT处理器的设计

张 裕,方康玲
(武汉科技大学 信息科学与工程学院,湖北 武汉 430081)

摘 要:介绍了一种通用的可以在低端或是高端的FPGA上实现$N(N=2M,M=2,3,4\cdots)$点FFT变换的方法。设计采用基4布斯编码算法和华莱士树算法设计完成了16X16位有符号数并行乘法器,并采用此并行乘法器为核心设计了FFT算法中的基-2蝶形运算单元,设计了串并转化模块、并串转换模块、移位选择模块、溢出检测模块和地址与控制模块等其它模块,并以这些模块和FPGA内部的双口RAM和ROM为基础组成了基-2FFT算法模块。整个模块采用基-2时域抽取,顺序输入,逆序输出的方法;利用Modelsim完成了FFT模块的前后仿真;利用Matlab编写了用于比较仿真结果和Matlab中FFT函数产生的结果的程序,从而验证了仿真结果的正确性。该模块最后能够在Cyclone EP1C6Q240C8型FPGA上稳定运行在60MHz。整个FFT模块能够在183μs左右完成1024点的16位定点复数FFT运算,能够满足一般工程的要求。该方法也可以用于实现更低点数或是更高点数的FFT运算。

关键词:FPGA;FFT;基2;时域抽取;块浮点
中图分类号:TN402 **文献标识码**:A **文章编号**:1673-629X(2010)08-0087-04

Design of General FFT Processor Based on FPGA

ZHANG Yu,FANG Kang-ling
(College of Information Science and Engineering,Wuhan University of Science and Technology,Wuhan 430081,China)

Abstract: Introduces a general method which can be realized in the low-end or high-end FPGA to achieve N ($N=2M,M=2,3,4\cdots$) point FFT. The 16X16-bits signed number parallel multiplier is completed with radix-4 Booth encoding algorithm and Wallace tree algorithm in the design. The parallel multiplier is used to design the radix-2 butterfly unit of the core design of the FFT algorithm, and the parallel-serial conversion module, serial-parallel conversion module, shift select module, overflow detection module, address and control module and other modules are designed. The radix-2 FFT module is formd based on these modules and the internal dual-port RAM and ROM of FPGA. The whole module uses the way of time-domain-based extraction, the order of input,output reverse. The pre-simulation and the post- simulation of the design is done by Modelsim. The code used to differ the simulation result from the FFT function in Matlab is created in Matlab and the simulation result is proved to be correct. Finally the design can be able to run at 60MHz in the Cyclone EP1C6Q240C8 stably. The FFT module can realize 1024 points,16-bit fixed-point complex FFT at about 183μs, and it can meet the general engineering requirements. This method can also be used to achieve a lower point or higher points FFT.

Key words: FPGA;FFT;radix-2;DIT;block-floating-point

0 引 言

随着电子技术的发展,数字信号处理(DSP)技术已经广泛地应用于数字通信、语音处理、图像处理、生物医学信号处理和机器人技术等许多领域。传统的数字信号处理一般都会使用通用的数字信号处理器来完成,比如Ti公司的TMS320C2000、TMS320C5000、TMS320C6000等。但是近年来,由于现场可编程门阵列(FPGA)技术的飞速发展,在很多场合,人们已经使用FPGA来替代DSP完成一些数字信号处理的算法了,并且取得了很好的效果。

快速傅里叶变换(FFT)是离散傅里叶(DFT)的一种快速的算法。FFT的运算速度比DFT提高了1~2个数量级。正是由于FFT的出现,使得信号的实时处理成为了可能。目前,FFT已经广泛地应用于生产生活的各个方面了。文中所介绍的就是一种利用FPGA实现FFT变换的方法。该方法主要是采用基于时域抽取的基2-FFT算法,并且采用了块浮点的计算方式防止在计算过程中可能产生的溢出。最后实现了1024点16位定点数的FFT运算。虽然最后实现的是1024点的运算,但是该方法同样适用于2^M($M=2,3,$

收稿日期:2009-12-22;修回日期:2010-02-08
作者简介:张 裕(1984-),男,湖北武汉人,硕士研究生,研究方向为嵌入式系统;方康玲,教授,博士生导师,研究方向为智能控制、图像处理、嵌入式系统等。

4,…)点的 FFT 运算。文中还给出了利用此法进行 8,16,32,64,128,256,512,1024 点的 FFT 变换所花费的时间。

1 基 2 - FFT 算法

1.1 概 述

傅里叶变换的理论在一百多年前就已经提出,它提供了频域分析信号的方法,用较精确的数字方法,即离散傅里叶变换(DFT)进行频谱分析,因为其计算量太大而难以实现,直接用 DFT 进行谱分析是不切实际的。直到 1965 年发现了 DFT 的一种快速算法以后,情况才发生了根本的改变。

自从 1965 年图基(J. W. Tuky)和库利(T. W. Coody)在《计算机数学》(Math. Computation, Vol. 19, 1965)杂志上发表了著名的《机器计算傅里叶级数的一种算法》论文后,桑德(G. Sand)—图基等快速算法相继出现,又经人们的改进,很快形成一套高效运算方法,这就是现在的快速傅里叶变换,简称 FFT。

FFT 算法并不是一种新的理论算法,它只是用来计算 DFT 的快速算法,所以它是以 DFT 为基础的。常用的 FFT 算法有基 2 算法、基 4 算法和混合基算法。因为基 2 算法可以计算出 $2^M(M=2,3,4\cdots)$点的 FFT,通用性最强,所以文中使用的是基 2 - FFT 算法。而且基 2 算法也便于硬件实现,这也是它很重要的一个优点。

1.2 算法基本原理介绍

基 2 - FFT 算法分为基于时域(DIT)抽取和基于频域(DIF)抽取两种。这两种的算法的复杂度是一样的。文中采用的是基于时域抽取的这一方法。

设序列 $x(n)$ 的长度为 N,且满足 $N=2^M$,M 为整数。将 N 点 DFT 先分成 2 个 $N/2$ 点 DFT,再是 4 个 $N/4$ 点 DFT,进而 8 个 $N/8$ 点 DFT,直至 $N/2$ 个 2 点 DFT。每分一次,称为一级运算。因为 $M=\log_2 N$,所以 N 点 DFT 可分成 M 级。FFT 的信号流图如图 1 所示,以 8 点的为例。图中有大量蝶形运算单元,它是 FFT 算法的基本运算单位[1]。

蝶形运算单元的计算公式如式(1)所示。

$$X(k)=X_1(k)+W_N^k X_2(k)\quad k=0,1,\cdots\frac{N}{2}-1$$

$$X(k+\frac{N}{2})=X_1(k)-W_N^k X_2(k)\quad k=0,1,\cdots\frac{N}{2}-1 \qquad (1)$$

其中,$W_N=e^{-j\frac{2\pi}{N}}$ 就称为旋转因子。

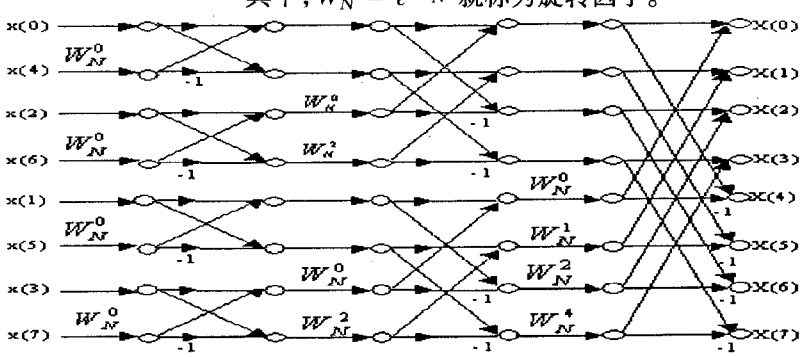

图 1 8 点 DIT - FFT 信号流图

2 FFT 的 FPGA 实现

2.1 整体结构

常用的利用 FPGA 实现 FFT 算法的系统结构有递归型和级联型两种。递归型耗费的硬件资源少,但是速度比级联型的慢。级联型耗费的硬件资源多,但是速度比递归型的快。由于本设计希望在低端和高端的 FPGA 上都能够实现,而低端的 FPGA 的资源有限,所以本设计最后选择了递归型的结构。虽然递归型结构的实现方式比级联型的要慢,但是只要设计合理,最后的运算速度也能够满足大多数工程的需要了[2~5]。

本设计整个系统的组成框图如图 2 所示。由图中可以看出,系统由移位模块、串并转换模块、蝶形运算模块、并串转换模块、溢出检测模块、地址及控制信号产生模块、dpram 和 rom 这些模块所构成。

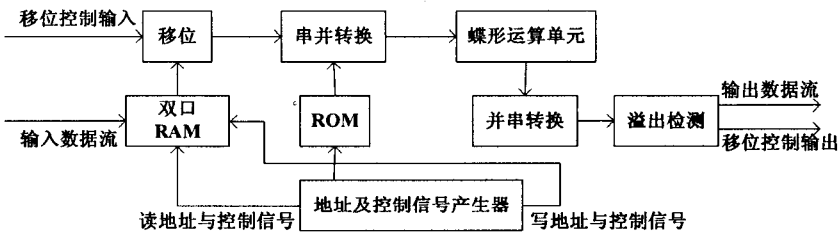

图 2 系统组成框图

2.2 dpram 数据存储模块

dpram 是双口 ram。双口 ram 不同于一般的 ram,它可以同时进行读写。altera 的 FPGA 中有 M4K RAM 存储模块,它可以被配置成单口 ram,简单双口 ram,真正双口 ram 和 rom。本设计中是将 M4K RAM 配置成简单双口 ram。简单双口 ram 有两个口,一个口只能读,另一个口只能写。但是注意,不能同时对同一个存储单元进行读写操作,这样可能会引起读写的错误。

本设计中有两个 dpram,其中一个为输入 dpram,另一个为输出 dpram。这两个 dpram 的数据宽度是 36 位的(高 18 位存放蝶形运算结果的实部,低 18 位存放

蝶形运算的虚部),数据深度是 1024。一开始,原始序列会被按顺序写入输入 dpram 中,然后进行 1～9 级蝶形运算时,都是从输入 dpram 中读取数据,然后再将数据以读取时地址为写入的地址写入到输入 dpram 中。到了第 10 级蝶形运算,就是将蝶形运算的结果写入到输出 dpram 中,最后从输出 dpram 中读出的即是一帧 1024 点的 FFT 的结果了。

2.3　旋转因子 ROM 存储单元

由式(1)可知,1024 点的 FFT 变换有 512 个旋转因子。旋转因子的实部和虚部的绝对值都是小于 1 的。将旋转因子的实部和虚部都用 16 位的有符号 2 进制数表示。这里是无法将小数直接表示出来,因此需要指定小数点的位置,这一过程就称作定点数的定标。数的定标有两种表示方法,即 Q 表示法和 S 表示法[6]。本设计中采用的就是 Q15 定标法。16 位有符号数的定标方法如表 1 所示。

表 1　16 位有符号数的定标表示法

Q 表示	S 表示	十进制数表示范围
Q15	S0.15	$-1\leqslant x\leqslant 0.9999695$
Q14	S1.14	$-2\leqslant x\leqslant 1.9999390$
Q13	S2.13	$-4\leqslant x\leqslant 3.9998779$
Q12	S3.12	$-8\leqslant x\leqslant 7.9997559$
Q11	S4.11	$-16\leqslant x\leqslant 15.9995117$
Q10	S5.10	$-32\leqslant x\leqslant 31.9990234$
Q9	S6.9	$-64\leqslant x\leqslant 63.9980469$
Q8	S7.8	$-128\leqslant x\leqslant 127.9960938$
Q7	S8.7	$-256\leqslant x\leqslant 255.9921875$
Q6	S9.6	$-512\leqslant x\leqslant 511.9804375$
Q5	S10.5	$-1024\leqslant x\leqslant 1023.96875$
Q4	S11.4	$-2048\leqslant x\leqslant 2047.9375$
Q3	S12.3	$-4096\leqslant x\leqslant 4095.875$
Q2	S13.2	$-8192\leqslant x\leqslant 8191.75$
Q1	S14.1	$-16384\leqslant x\leqslant 16383.5$
Q0	S15.0	$-32768\leqslant x\leqslant 32767$

在本设计中,将 M4K RAM 模块配置成两个数据宽度为 16 位,数据深度为 512 的 ROM。一个就是用来存储旋转因子的实部的,另一个就是用来存储旋转因子的虚部的。旋转因子实部的数据可以通过 Matlab 7.1 来产生。Matlab 7.1 当中增加了定点数运算的工具箱,可以很方便地将浮点数转换成定点数并且完成各种定点数的运算。

2.4　蝶形运算模块

蝶形运算模块就是式(1)的硬件实现。在这里,为了方便说明,令:

$$X_1(k) = a_1 + b_1 * i$$

$$X_2(k) = a_2 + b_2 * i$$

$$W_N^k = c + d * i$$

那么式(1) 就转换成为式(2):

$$X(k) = (a_1 + c * a_2 - d * b_2) + (b_1 + c * b_2 + d * a_2) * i$$

$$X(k + N/2) = (a_1 - c * a_2 + d * b_2) + (b_1 - c * b_2 - d * a_2) * i \quad (2)$$

由式(2)可以看出,蝶形运算就是进行乘法运算、加法运算和减法运算。乘法运算的速度是影响蝶形运算速度的关键所在,所以蝶形运算模块中乘法器的设计就显得尤为重要了。高端的 FPGA 中都是含有硬线乘法器的,它们的运算速度很快。所以如果使用高端的 FPGA 来完成 FFT 变换,建议采用片上乘法器模块。但是低端的 FPGA 当中是没有乘法器的模块的,这样就需要自己设计乘法器了。目前流行的做法是使用改进型的 booth 编码器 + 3 - 2 压缩器 + wallace 的结构。文献[7～9]中详细介绍了这种结构,这里就不加赘述了。本设计中所使用的 16X16 位乘法器就是采用这种结构完成的。

蝶形运算模块的输入为 16 位有符号数的 2 进制补码。在蝶形运算模块内部,由于定点数有限字长效应的影响,采取了舍入的手段。与截位运算相比,舍入运算后的结果精度更高[10]。

2.5　串并转换模块

串并转换模块的作用是将串行数据转换成并行数据。因为每次是从 dpram 中读取一个数据,所以要将先读出来的这一个数据延时一个节拍,和后读出来的这个数据一起送往蝶形运算模块进行蝶形运算。

2.6　并串转换模块

并串转换模块的作用是将并行数据转换成串行数据。因为蝶形运算后是同时输出了两组数据,而每一个时钟节拍到来时只能往 dpram 中写入一组数据。所以要将两组数据中的其中一组数据先写入 dpram,将另外一组数据延时一个节拍后再写入到 dpram 中。

2.7　移位模块和溢出检测模块

移位模块和溢出检测模块就是为了实现块浮点的运算思想。块浮点算法的速度比浮点算法的速度高,其精度比定点算法的精度高,所以采用块浮点算法可以很好地在速度与精度之间取得平衡。块浮点部分由两个功能单元组成:指数检测单元和指数累加单元。假设蝶形运算模块的输入数据为 n 位,且每进行一次加减法就扩展一位,则蝶形运算模块的输出数据应为 $n+2$ 位。溢出检测模块检测这 $n+2$ 位数据。如果它的高三位为 000 或 111,则溢出为零;如果为 001 或 110,则溢出为 1;如果为 $01x$ 或 $10x$(x 表示无关),则溢出为 2。指数累加单元将每级蝶形运算的溢出相加,构成

指数部分。移位模块就是根据溢出的结果来进行数据移位，将移位后的数据送到蝶形运算模块当中的[11,12]。

2.8 地址及控制信号产生模块

地址及控制信号产生模块是整个系统的控制核心。各个功能模块之间的地址、数据传递均通过地址及控制信号产生模块协调工作。当外部转换开始信号到来的时候，地址及控制信号产生模块就将输入的原始序列数据按顺序写入到输入 dpram 中。当所有数据写入完毕以后，该模块又产生正确的地址和控制信号将数据从输入 dpram 中读出，送入串并转换模块、移位模块、蝶形运算模块、并串转换模块等模块处理。处理完毕后，地址及控制信号产生模块又产生控制信号将数据回写入输入 dpram。蝶形运算的前 1～9 级都是这样的。到了第 10 级蝶形运算，所有运算完的数据都是写入输出 dpram 的，最后从输出 dpram 中读出的就是最终蝶形运算的结果了。地址及控制信号产生模块是基于计数器的。蝶形运算的两个操作数的地址和旋转因子的地址与计数器关系在文献[5]中有详细的介绍。这里就不加赘述了。

3 系统仿真与验证

本设计采用自顶向下的设计思路，完成系统模块划分和各个功能模块的 Verilog 代码编写。选用 Quartus－Ⅱ7.1 作为软件开发平台，利用该软件宏模块的调用功能产生 RAM、ROM 以节省设计时间，提高系统工作效率。整个工程经过 quartusII 7.1 全编译后，编译结果如图 3 所示。

Flow Status	Successful - Sun Sep 13 15:24:11 2009
Quartus II Version	7.1 Build 156 04/30/2007 SJ Full Version
Revision Name	fft_1024_top
Top-level Entity Name	fft_1024_top
Family	Cyclone
Device	EP1C6Q240C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	3,103 / 5,980 (52 %)
Total pins	74 / 185 (40 %)
Total virtual pins	0
Total memory bits	90,112 / 92,160 (98 %)
Total PLLs	0 / 2 (0 %)

图 3 编译结果

由图 3 可以看出，该工程占用了 52% 的 FPGA 的逻辑单元和 98% 的片上 RAM 资源。

功能仿真和时序仿真都是使用 Modelsim 6.2b 完成的。由仿真结果知，系统的频率可达 60M，完成一帧 1024 点 16 位定点数的 FFT 变换所需的时间约为 182.5μs。

算法验证时利用 Matlab 7.1 完成的。测试用的激励函数由以下 Matlab 语句产生。

```
n=1024; %Number of points
Fs=4; %Sampling frequency in Hz
t=(0:(n-1))/Fs; %Time vector
f=linspace(0,Fs,n); %Frequency vector
f0=.2; f1=.5;
x0=cos(2*pi*f0*t) + 0.55*sin(2*pi*f1*t);
```

图 4 为将 Modelsim 仿真所得数据与 Matlab 计算所得数据进行对比的结果。

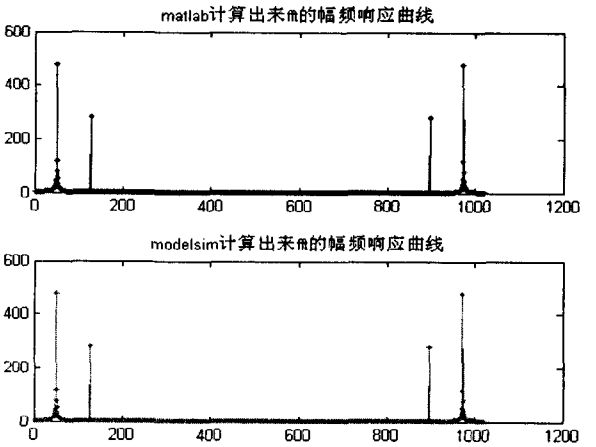

图 4 算法验证

图 4 即验证了该算法的正确性。

笔者还用此方法完成了 8 点、16 点、32 点、64 点、128 点、256 点、512 点的 16 位定点数的 FFT 变换，各点数 FFT 所耗费的时间如表 2 所示。

表 2 各点数 FFT 变换所需时间

变换点数	耗费时间(μs)
8	1.2
16	2.2
32	4.2
64	8.5
128	17.8
256	38.5
512	83.8

4 结束语

文中介绍了一种在低成本的 FPGA 上实现 FFT 的通用的方法。利用该方法当系统频率为 60MHz 时完成一帧 1024 点 16 位定点数的 FFT 变换所需的时间为 182.5μs。这样的性能已经能够满足大多数工程的需要了。相信随着 FPGA 技术的不断发展，FPGA 会在越来越多的场合替代 DSP 处理器来完成数字信号处理的工作。

(下转第 95 页)

ta_pre=11111111,用 d3=1 来描述用户没有进入普适环境,用 d7=1 来描述教授角色拥有享受个性化饮品服务。当 d3 所对应的按键按下时(模拟系统检测到某用户进入普适环境系统,即用户位置信息发生变化),系统检测到这个变化后,会相应地改变 data 的值(模拟给用户赋某个角色,比如给某个用户赋教授角色),并在 LED 上显示。当 d7 所对应的按键按下时(模拟环境中水资源不足,即环境上下文信息发生变化),系统检测到这个变化后,会相应地改变 data 的值(模拟改变角色的权限,这里为 data 中 d7 位变为 0,假设为教授角色没有权限享受个性化饮品服务),并在 LED 上显示;同样,当工作人员添加足够的水资源后(用 d7 按键不按下来表示),系统检测到这个变化,会相应改变 data 值的 d7 为 1 来体现教授角色拥有享受个性化饮品服务的权限。

本实验的上下文信息的变化检测,在普适环境中可以用各种传感器来实现,比如位置传感器可以用来检测用户是否进入环境中等等。

4　结束语

访问控制是安全访问数据必要的环节,在普适计算的智能化环境中尤其重要。文中在介绍桌面计算模式下的典型访问控制策略基础上,提出在普适环境下访问控制的特殊需求,并对普适环境下上下文相关的动态访问控制策略(DRBAC)作详细描述。文中也通过实验模拟了系统检测上下文环境变化并做出反应的过程。最后,提出了 DRBAC 相关的数据描述设计。

参考文献:

[1] Weiser M. The Computer for the 21st Century[J]. Scientific American,1991,265(3):94－104.

[2] 徐光佑,史元春,谢伟凯.普适计算[J].计算机学报,2003,26(9):1042－1050.

[3] Sandhu R, Samarati P. Access Control: Principles and Practice[J]. IEEE Communications,1994,32(9):40－48.

[4] Snyder L. Formal Models of Capability－Based Protection Systems[J]. IEEE Transactions on Computers, 1981,30(3):172－181.

[5] 孙尚辉,曹宝象,王廷蔚.扩展 RBAC 模型在文档管理中的应用[J].计算机技术与发展,2007,17(3):210－213.

[6] 孟庆荣.协同编辑中访问控制模型的设计与实现[J].计算机技术与发展,2007,17(2):72－74.

[7] Feinstein H, Sandhu R, Coyne E, et al. Role－based access control models[J]. IEEE Computer,1996,29(2):38－47.

[8] 邓集波,洪　帆.基于任务的访问控制模型[J].软件学报,2003,14(1):76－81.

[9] 郭　慧,李阳明,王丽芬.基于角色和任务的访问控制模型的设计与研究[J].计算机工程,2006,32(16):143－145.

[10] 景栋盛,杨季文.一种基于任务和角色的访问控制模型及其应用[J].计算机技术与发展,2006,16(2):212－214.

[11] Zhang G, Parashar M. Context－Aware Dynamic Access Control for Pervasive Applications[C]//Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS). [s.l.]:[s.n.],2004:219－221.

[12] 李　蕊.上下文感知计算若干关键技术研究[D].长沙:湖南大学,2007.

(上接第 90 页)

参考文献:

[1] 丁玉美,高西全.数字信号处理[M].第 2 版.西安:西安电子科技大学出版社,2000.

[2] 钱文明,刘新宁,张艳丽.基于 Cyclone 系列 FPGA 的 1024 点 FFT 算法的实现[J].电子工程师,2007,33(2):12－14.

[3] Chu Chao, Zhang Qin, XIE Yingke, et al. Design of a high performance FFT processor based on FPGA[C]//Proceedings of Design Automati on Conference, Asia and South Pacific. Shanghai, China:[s.n.],2005:920－923.

[4] Sung C H, Lee K B, Jen C W. Design and implementation of a scalable fast Fourier transform core[C]//Proceedings of 2002 IEEE ASIA－Pacific Conference on ASIC, Aug 2－8,2002, Taipei, China. Piscataway, NJ, USA: IEEE,2002:295－298.

[5] 梁曦捷,肖　璋.一种基于 FPGA 的顺序迭代 FFT 设计[J].微计算机信息,2005,21(12):135－137.

[6] 褚振勇,齐　亮,田红心,等.FPGA 设计及应用[M].第 2 版.西安:西安电子科技大学出版社,2006.

[7] 李　磊,赵建明.高速可重组 16×16 乘法器的设计[J].微电子学与计算机,2007,24(6):120－122.

[8] 王　定,余宁梅,张玉伦.改进型 booth 华莱士树的低功耗高速并行乘法器的设计[J].电子器件,2007,30(1):252－255.

[9] Hsiao S－F, Jiang M－R. Efficient Synthesiser for Generation of Fast Parallel Multiplier[J]. Computers and Digital Technology, IEEE Proceedings,2000,147(1):49－52.

[10] 贾玉臣,吴嗣亮.快速傅里叶变换的误差分析[J].北京理工大学学报,2005,25(8):739－742.

[11] 段小东,顾立志.高性能基 4 快速傅里叶变换处理器的设计[J].计算机工程,2008,34(24):238－240.

[12] 张傲华,张正鸿,尧德中.一种基于 FPGA 的高性能 FFT 处理器设计[J].电子对抗技术,2005,20(4):44－47.