·信号与信息处理·

基于FPGA的FFT算法设计与实现

刘宝军,王中训,钟 强,张 珉,娄 阳

(烟台大学 光电信息科学技术学院,山东 烟台 264005)

摘 要:提出一种基于FPGA实现32点实序列FFT的设计方法,采用基二时间抽取算法以及并行迭代处理方案。选择Altera公司Cyclone IV系列芯片,并用Verilog HDL完成设计输入。最后通过Signal Tap与Matlab数据对比发现,该设计可以实现对高速A/D采样数据的实时、准确处理。

关键词:现场可编程门阵列;快速傅里叶变换;硬件描述语言;逻辑分析仪

中图分类号:TN911.23 **文献标识码**:A **文章编号**:1673-1255(2016)-03-0046-04

Design and Implementation of FFT Algorithm Based on FPGA

LIU Bao-jun, WANG Zhong-xun, ZHONG Qiang, ZHANG Min, LOU Yang

(School of Opto-electronic Information Science and Technology, Yantai University, Yantai 264005*, China)*

Abstract: A design method of 32-point real sequence fast Fourier transform (FFT) based on field programmable gate array (FPGA) is presented. Base-2 DIT algorithm and parallel iterative processing scheme are adopted. Cyclone IV series chip of Altera company is chosen and Verilog HDL input method is used to implement design input. It is found that the design can achieve high speed A/D sampling data processing accurately and in real time through contrast of signal tap and Matlab data.

Key words: field programmable gate array (FPGA); fast Fourier transform (FFT); hardware description language; signal tap

快速傅里叶变换(FFT)是数字信号处理中的基本变换,它广泛应用于通信、雷达以及电子对抗等领域。目前采用硬件实现FFT的方法很多,例如ASIC、DSP等,但是随着FPGA技术的不断发展,采用FPGA实现超高速、大点数、高精度的FFT逐渐成为人们研究的热点。现场可编程门阵列(FPGA)直接针对硬件进行设计,具有实时性好、灵活性强、支持在线可编程、可充分进行设计开发和验证等优势[1]。高速FFT运算需要乘法器、大量存储器、寄存器,适合用FPGA实现[2-3]。国内越来越多的学者都参与到FFT的FPGA实现工作中,并且取得了令人欣喜的成果。但是由于起步较晚、基础薄弱的FFT处理器还没能追赶上国外的水平。

1 实序列FFT实现原理

FFT是离散傅里叶变换(DFT)的快速算法。DFT是将时域序列变换为长度相同的频域序列,从理论上讲,输入序列为复序列。对于N点离散有限长时间序列$x(n)$,离散傅里叶变换为[4-5]

$$X(K)=\sum_{n=0}^{N-1}x(n)W_N^{nk} \qquad k=0,1,2,....,N-1 \qquad (1)$$

但在实际工程实践中,输入序列$x(n)$通常是实序列。如果直接按照FFT运算流图计算,就是把$x(n)$看成一个虚部为0的复序列进行计算,这必将增加存储量和运算时间。根据数字信号处理理论[6-7],采用一种更加优良的方法实现32点实序列的

收稿日期:2016-05-06

基金项目:山东省科技攻关计划项目(2012J0030009)

作者简介:刘宝军(1991-),男,山东省莱西市人,烟台大学硕士,主要研究方向为LDPC码.

FFT运算。把32点实序列DFT计算转换为一个16点复序列DFT计算,然后将16点复序列的输出进行适当的运算组合就可得到原始序列的DFT。$x(n)$为N=32点实序列,取$x(n)$的偶数点$f(n)$和奇数点$g(n)$分别作为序列$h(n)$的实部和虚部构造新序列,有下式

$$h(n)=f(n)+jg(n)\quad n=0,1,...,15 \tag{2}$$

则$h(n)$就是一个16点复数序列,用FFT计算$h(n)$的16点离散傅里叶变换$H(k)$,就可通过$H(k)$推导得出$F(k)$和$G(k)$分别为

$$\begin{cases}F(k)=\dfrac{1}{2}[H(k)+H^*(\dfrac{N}{2}-k)]\\ G(k)=-\dfrac{j}{2}[H(k)-H^*(\dfrac{N}{2}-k)]\end{cases}\quad k=0,1,...15 \tag{3}$$

求出$F(k)$和$G(k)$后就可利用下面的公式计算出$x(n)$的离散傅里叶变换$X(k)$为

$$\begin{cases}X(k)=F(k)+W_N^kG(k)\\ X(k+\dfrac{N}{2})=F(k)-W_N^kG(k)\end{cases}\quad k=0,1,...15 \tag{4}$$

这种算法比相同长度的复序列FFT算法减少约一半的运算量,运算效率提高了近一倍。

2　实序列FFT硬件实现

2.1　系统总体设计

目前FFT硬件实现结构主要分为三类:递归结构、流水线结构以及并行迭代结构。递归结构又叫作内存共享结构,这种结构的主要优点是只有一个运算单元,占用硬件资源少,缺点是运算时间较长。流水线结构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,因此速度上有所提高。并行迭代结构每级都采用N/2个蝶形单元进行并行运算,每一列中的运算并行进行,一列迭代至另一列是顺序进行的。这种方法处理速度快,但设备量较大需要采用大规模集成电路实现。考虑到本设计点数较少,对处理速度要求较高,因此采用并行迭代结构实现。设计主要包括以下几个模块:实/复序列转换模块、输入数据串并转换模块、基2-DIT算法实现模块、输出数据并串转换模块以及还原实序列DFT模块。实/复序列转换模块主要完成的功能为:将32点实序列转换为16点复序列,然后再进行FFT运算。还原实序列DFT模块是将16点复序列$h(n)$的FFT运算结果根据式(3)、式(4)求出32点实序列$x(n)$的FFT运算结果。基2-DIT算法实现模块是整个系统实现的核心模块,因此主要介绍基2-DIT算法实现模块。因为输入的32点实序列通过实/复转换模块将变成16点复序列,所以基2-DIT实现模块主要是针对16点复序列设计完成的。

2.2　基2-DIT算法硬件实现

基2-DIT算法实现的核心是蝶型运算单元的实现,蝶型运算单元的结构框图如图1所示。蝶型运算单元的设计主要包括以下几个模块:复数乘法模块、延时D触发模块、截取模块、缩小模块以及实数加减模块,文中主要介绍复数乘法模块、截取模块、缩小模块。

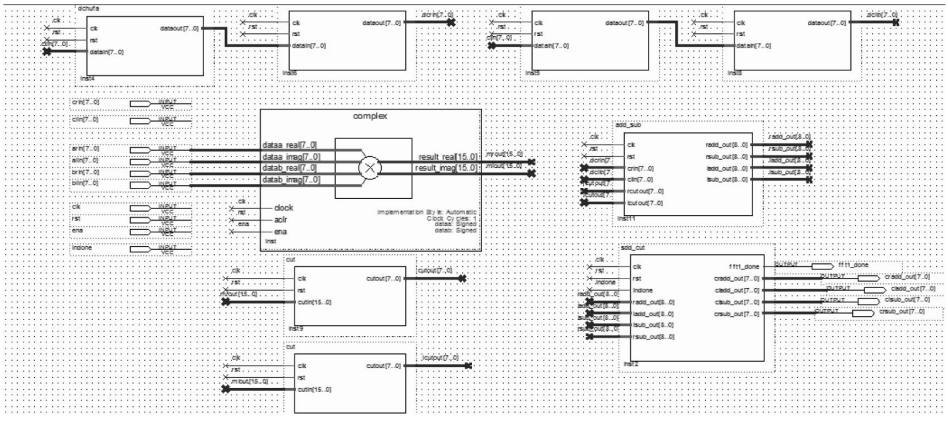


图1　蝶型运算单元结构框图

复数乘法模快采用IP核ALTMULT-COMPLEX实现，主要完成旋转因子与一个输入数据的复数乘法。考虑到16点FFT设计只需8个旋转因子，因此可以先通过Matlab计算出所需的旋转因子并存储在ROM里，按照每级运算的需求直接读取降低运算量。由于旋转因子取值范围在[-1,1]之间，为了方便定点运算，将旋转因子扩大2^6并取整数，这样复数乘法器的输出结果也扩大2^6。由于FPGA内部运算采用定点处理，因此每次复乘之后输出结果的位宽都要扩大一倍。并且，相乘之后的数据还要相加减，为了保证不溢出，输出结果位宽还要扩展一位。所以随着FFT运算点数的增加，输出结果的位宽将以2的指数次幂增加，占用的资源太多以致无法实现。因此，必须增加截取模块。

截取模块实现的功能是将复数乘法器的16位输出crout[15:0]、ciout[15:0]截取8位。通过比较，多种截取方式找到一种误差比较低的截取方式：首先判断crout[6]、ciout[6]是否为1，如果是1，则输出为{crout[15], crout[13:7]}+1、{ciout[15], ciout[13:7]}+1；如果不是1，则输出为{crout[15], crout[13:7]}、{ciout[15], ciout[13:7]}。截取后数据输出结果缩小了2^7倍，因为做复数乘法之前旋转因子扩大2^6倍，所以复数乘法器的最终输出结果缩小了2倍。因此也应该将另一个输入数据缩小2倍，才能进行后续的加减操作。因此每级蝶型运算的输出都缩小两倍，这样可以有效的避免输出结果的溢出，FFT最后的输出结果缩小了16倍。因为对于FFT运算，所关心的主要是输出数据的相对幅度，所以这种固定的缩小比例对FFT运算结果影响不大[8]。

缩小模块实现的功能是将不与旋转因子相乘的输入数据缩小2倍。通过移位的方式可以近似实现缩小两倍的功能，这种方式比除法占用更少的逻辑资源。具体的实现方式是：首先判断输入数据datain的符号位datain[7]是0还是1（即判断输入数据的正负），如果是0则输出为{1'b0,datain[7:1]}；如果是1则输出为{1'b1,datain[7:1]}。

整个基2-DIT算法的硬件实现分为4级，每级用8个蝶形运算单元实现。前一级运算结束的标志作为下一级运算以及读取旋转因子的启动信号，上一级运算结果直接送到下一级作为输入数据。基2-DIT算法实现模块的输出经过并串转换后送入还原实序列DFT模块进行组合运算，输出结果即为32点实序列FFT运算的最终结果。基2-DIT模块输出结果缩小16倍，还原实序列DIT模块输出结果缩小2倍，所以整个系统的输出缩小了32倍。

3 仿真与分析

系统设计完成后用Signal Tap观察仿真结果。当串行输入32点实序列$x(n)$=[21 39 58 76 97 119 126 117 96 78 53 36 18 −10 −35 −66 −38 −8 26 38 62 40 23 −14 −30 −52 −68 −76 −89 −101 −118 −125]时，经过实/复转换模块输出为16点复序列$y(n)$=[21+39i 58+76i 97+119i 126+117i 96+ 78i 53+36i18−10i −35−66i −38−8i26+38i62+40i23−14i −30 −52i −68−76i −89−101i −118−125i]，将$y(n)$串并转换后送入基2-DIT实现模块得到16点复序列$y(n)$的FFT运算结果，仿真波形如图2所示。

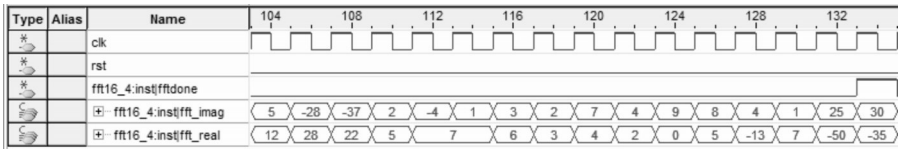


图2 16点复序列FFT运算结果仿真图

最后通过输出数据并串转换模块以及还原实序列DFT模块得到最终的FFT运算结果，Signal Tap仿真结果如图3所示。

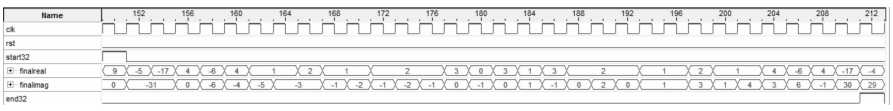


图3 32点实序列FFT运算结果仿真图

为了验证系统设计的准确性,用Matlab计算$x(n)$　　的FFT理论值并缩小32倍后的结果如图4所示。

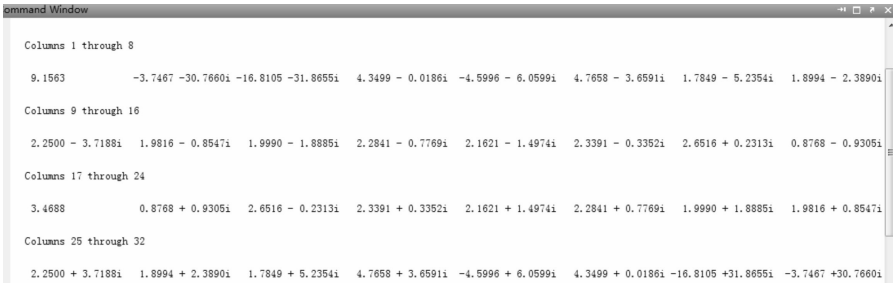
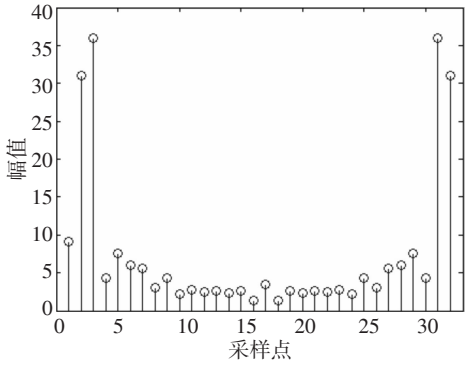
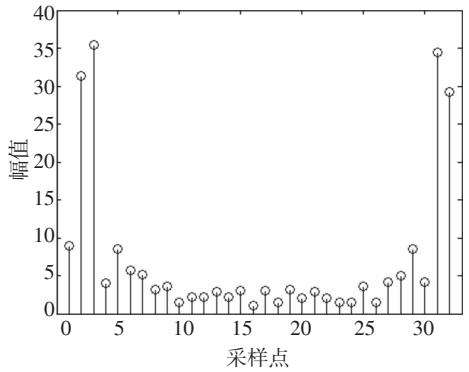


图4　32点实序列FFT计算理论结果(缩小32倍)

Matlab计算理论结果与Signal Tap输出实际结果幅值比较如图5所示。通过对比可以发现,本设计仿真结果与Matlab计算得到的理论值大致相同,误差较小。FFT运算过程主要的误差有舍入误差、溢出误差、截取误差,由于本设计采用定点运算,所以主要误差为截取误差。如果想要提高精度,可以采用浮点或块浮点运算,但是硬件实现的复杂度也会相对提高。



(a)Matlab计算理论结果



(b)Signal Tap计算实际结果

图5　理论结果与实际结果幅值比较

4　结　论

设计采用并行迭代处理方案实现了32点8位实序列FFT,通过Signal Tap与Matlab数据对比发现,该设计可以实现预期的功能。从串行数据全部输入开始到最后开始输出数据总共需要42个时钟周期,本设计在一些数字信号处理领域(比如信道化接收机)具有良好的应用前景。

参考文献

[1] Sanchez M A, Garrido M, Lopez-Vallejo M, et al. Implementing FFT based digital channelized receivers on FPGA platforms [J]. Aerospace and Electronic Systems, 2008, 44(4): 1567-1585.

[2] 刘美容.FFT算法的DSP实现[J]. 微电子学与计算机,2015,32(1):76-79,84.

[3] 万红星,陈禾,韩月秋.一种高速并行FFT处理器的VLSI结构设计[J]. 电子技术应用,2005,31(50):45-48.

[4] 高西全,丁玉美.数字信号处理[M]. 3版.西安:西安电子科技大学出版社,2008:110-123.

[5] 吴大正.信号与线性系统[M]. 4版.北京:高等教育出版社,2009:115-187.

[6] 郑南宁.数字信号处理[M]. 西安:西安交通大学出版社,1991:67-69.

[7] Alan V O, Ronald W S. Discrete-time digital signal processing [M]. Beijing: Tsinghua University Press, 2005.

[8] 王旭东,刘渝.全并行结构FFT的FPGA实现[J]. 南京航空航天大学学报,2006,38(2):96-100.

[9] 马春燕. 基于投影的图像识别方法研究[J]. 光电技术应用,2016,30(6):51-55.