

文章编号: 1000-5013( 2008) 04 0542- 03

# 一种 FFT 处理器的地址生成算法

张育钊, 邱应强

( 华侨大学 信息科学与工程学院, 福建 泉州 362021 )

**摘要:** 通过对 8 点基 2 按时间抽取-快速傅里叶变换(DIT-FFT)、按频率抽取-快速傅里叶变换 DIF-FFT 流程图的分析, 总结出连续参加蝶形单元运算结点数据和旋转因子的地址产生规律. 提出一种基 2 FFT 处理器中结点数据地址和旋转因子地址快速生成算法. 该算法只需通过对几个相关寄存器进行移位操作, 即可快速生成蝶形运算单元结点数据和旋转因子的地址.  
**关键词:** 快速傅里叶变换; 处理器; 蝶形运算; 地址生成  
**中图分类号:** TN 911. 21 **文献标识码:** A

快速傅里叶变换(FFT)算法在各种数字信号处理中起着核心的作用, 已广泛应用于通信、控制、信号处理、图像处理等许多领域<sup>[1]</sup>. 作为 FFT 算法的硬件实现, FFT 处理器主要由蝶形运算单元、地址生成单元、运算数据寄存器及旋转因子寄存器构成<sup>[2]</sup>. 地址生成单元负责产生蝶形运算单元两结点数据地址及该蝶形单元所用旋转因子地址<sup>[2-6]</sup>, 其速度快慢将直接影响处理器的性能. 本文提出了一种地址生成新算法.

## 1 基 2 FFT 算法

$N$  个样点的离散傅里叶变换(DFT)定义为

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{nk}, \quad k = 0, \dots, N-1; \quad W_N = e^{-j\frac{2\pi}{N}} \tag{1}$$

直接运算复杂度较高, 如运用基 2FFT 可通过将序列逐步分解到最后全是两个点的 DFT 运算, 利用  $W_N^{nk}$  的对称性和周期性, 大大简化 DFT 的运算量. 图 1(a) 为  $N=8$  按基 2 时间抽取法 FFT 运算流程图, 其参与运算的数据倒序输入, 原位同址运算得到结果数据顺序输出; 图 1(b) 为  $N=8$  按基 2 频率抽取法 FFT 运算流程图, 参与运算的数据顺序输入, 原位同址运算得到结果数据倒顺序输出.

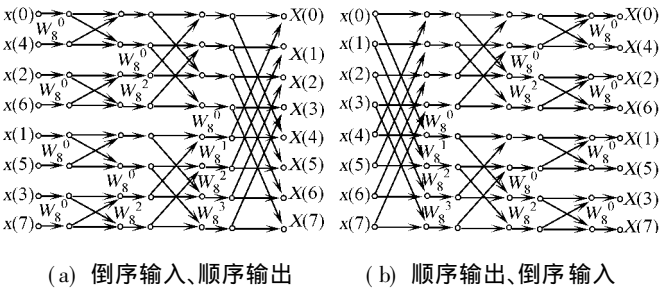


图 1 8 点基的 2 DIF-FFT 流图

## 2 基 2 FFT 算法的地址产生规律

Fig. 1 The flow process diagram of 8-points radix-2 DIT-FFT

### 2.1 8 点基 2 DIT-FFT 蝶形单元结点数据地址

图 1(a) 中各级蝶形运算单元的两结点数据地址, 如表 1 所示. 每个蝶形单元地址用二进制数  $(a_2 a_1 a_0 / b_2 b_1 b_0)_j$  表示,  $j$  ( $0 \leq j \leq 2$ ) 代表第  $j$  级蝶形运算. 每级蝶形运算的各蝶形单元的次序表示成二进制数  $(c_1 c_0)$ . 由于蝶形单元运算次序是任意的, 在此假设从上到下依次对蝶形单元排序. 不难发现, 蝶形单元次序位数比蝶形单元节点地址少一位. 为了更好地说明蝶形单元地址  $(a_2 a_1 a_0 / b_2 b_1 b_0)_j$ 、级数  $j$

收稿日期: 2008-07-12

作者简介: 张育钊 (1963-), 男, 讲师, 主要从事数字信号处理研究. E-mail: zyz@hqu.edu.cn.

基金项目: 福建省自然科学基金资助项目 (2006J0404); 华侨大学科研基金资助项目 (06HZR25)

表 1 各级蝶形运算单元两结点数据地址

Tab.1 Two operands' addresses of every butterfly calculation

$c_1c_0$	第 0 级蝶形运算	第 1 级蝶形运算	第 2 级蝶形运算
00	$(000/001)_0$	$(000/010)_1$	$(000/100)_2$
01	$(010/011)_0$	$(001/011)_1$	$(001/101)_2$
10	$(100/101)_0$	$(100/110)_1$	$(010/110)_2$
11	$(110/111)_0$	$(101/111)_1$	$(011/111)_2$

与蝶形单元次序( $c_1c_0$ )之间的关系, 可在蝶形单元次序( $c_1c_0$ )添加一位二进制数  $c_{-1}c_{-1} \in (0, 1)$ . 事实上, 当  $c_{-1} = 0$  时,  $(a_2a_1a_0)_0 = (c_1c_0c_{-1})$ ,  $(a_2a_1a_0)_1 = (c_1c_{-1}c_0)$ ,  $(a_2a_1a_0)_2 = (c_{-1}c_1c_0)$ . 当  $c_{-1} = 1$  时,  $(b_2b_1b_0)_0 = (c_1c_0c_{-1})$ ,  $(b_2b_1b_0)_1 = (c_1c_{-1}c_0)$ ,  $(b_2b_1b_0)_2 = (c_{-1}c_1c_0)$ . 因此, 在图 1(a) 所示的 8 点基 2 DIT-FFT 运算流程中, 将序列( $c_1c_0c_{-1}$ )中的  $c_{-1}$  位依次向左移  $j$  位. 其中,  $c_{-1}$  分别取 0, 1 值, 即可得到第  $j$  级蝶形运算中次序为( $c_1c_0$ )的蝶形单元两结点数据地址( $a_2a_1a_0/b_2b_1b_0$ ) $_j$ .

2.2  $N = 2^M$  点基 2 DIT-FFT 蝶形单元结点数据地址

对于任意  $N = 2^M$  的基 2 FFT 运算, 需要进行  $M$  级蝶形运算, 每级蝶形运算中包含  $N = 2^{M-1}$  蝶形运算单元. 如采用时间抽取法顺序输入、倒序输出实现 FFT 运算, 可用  $(a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0)_j$  表示第  $j$  ( $0 \leq j \leq M-1$ ) 级蝶形运算中蝶形运算单元的两结点地址,  $(c_{M-2} \dots c_0)$  表示各级蝶形运算中蝶形运算单元的次序. 在二进制序列( $c_{M-2} \dots c_0$ )添加一个二进制位  $c_{-1}$ , 并将  $c_{-1}$  依次在序列中左移  $j$ ,  $c_{-1}$  分别取 0, 1 值即得到地址  $(a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0)_j$ . 即  $(a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0)_0 = (c_{M-2} \dots c_00/c_{M-2} \dots c_01)$ ,  $\dots$ ,  $(a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0)_j = (c_{M-2} \dots c_j0c_{j-1} \dots c_0/c_{M-2} \dots c_j1c_{j-1} \dots c_0)$ ,  $\dots$ ,  $(a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0)_{M-1} = (0c_{M-2} \dots c_0/1c_{M-2} \dots c_0)$ .

2.3  $N = 2^M$  点基 2 DIF-FFT 蝶形单元结点数据地址

对比图 1(a), (b) 可以看出, 将按倒序输入、顺序输出的 DIT-FFT 运算流图加以转置, 即可以得到顺序输入、倒序输出的 DIF-FFT 运算流图. 因此, 从倒序输入、顺序输出 DIT-FFT 地址生成规律中, 不难得到在各级蝶形运算中次序为( $c_{M-2} \dots c_0$ )的蝶形运算单元前加二进制位  $c_{M-1}$  组成序列  $c_{M-1}c_{M-2} \dots c_0$ . 将  $c_{M-1}$  依次在  $c_{M-1}c_{M-2} \dots c_0$  依次右移  $j$  位, 分别取 0, 1 值, 即可得到第  $j$  级蝶形运算中( $c_{M-2} \dots c_0$ )个蝶形运算单元的两结点数据地址( $a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0$ ) $_j$ . 有  $(a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0)_0 = (0c_{M-2} \dots c_0/1c_{M-2} \dots c_0)$ ,  $\dots$ ,  $(a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0)_j = (c_{M-2} \dots c_{M-j-1}0c_{M-j-2} \dots c_0/c_{M-2} \dots c_{M-j-1}1c_{M-j-2} \dots c_0)$ ,  $\dots$ ,  $(a_{M-1} \dots a_1a_0/b_{M-1} \dots b_1b_0)_{M-1} = (c_{M-2} \dots c_00/c_{M-2} \dots c_01)$ .

2.4 旋转因子地址

在  $N = 2^M$  的基 2 FFT 蝶形运算中, 还将用到  $2^{M-1}$  个形如  $W_N^k$  ( $0 \leq k \leq 2^{M-1} - 1$ ) 的旋转因子. 假设这些选转因子按  $k$  大小顺序存放在 ROM 中, 地址可用  $M-1$  位二进制数  $d_{M-2} \dots d_1d_0$ , 文[3] 分析了旋转因子地址  $d_{M-2} \dots d_1d_0$  与蝶形单元次序  $c_{M-2} \dots c_1c_0$  之间的关系. 在此, 可进一步总结得到这样的规律, 对于 DIT-FFT, 有  $(d_{M-2} \dots d_1d_0)_0 = (00 \dots 00)$ ,  $(d_{M-2} \dots d_1d_0)_1 = (c_00 \dots 00)$ ,  $\dots$ ,  $(d_{M-2} \dots d_1d_0)_j = (c_{j-1}c_{j-2} \dots 00)$ ,  $\dots$ ,  $(d_{M-2} \dots d_1d_0)_{M-2} = (c_{M-3}c_{M-4} \dots c_00)$ ,  $(d_{M-2} \dots d_1d_0)_{M-1} = (c_{M-2}c_{M-3} \dots c_1c_0)$ ; 而对于 DIF-FFT,  $(d_{M-2} \dots d_1d_0)_0 = (c_{M-2}c_{M-3} \dots c_1c_0)$ ,  $(d_{M-2} \dots d_1d_0)_1 = (c_{M-3}c_{M-4} \dots c_00)$ ,  $\dots$ ,  $(d_{M-2} \dots d_1d_0)_j = (c_{M-j-2}c_{M-j-3} \dots 00)$ ,  $\dots$ ,  $(d_{M-2} \dots d_1d_0)_{M-2} = (c_00 \dots 00)$ ,  $(d_{M-2} \dots d_1d_0)_{M-1} = (00 \dots 00)$ .

因此, 用硬件生成 DIT-FFT 算法旋转因子时, 可以通过采用  $M-1$  个移位寄存器, 设置初值均为 0, 根据当前运算级数  $j$ , 将  $c_{M-2} \dots c_1c_0$  中的低  $j$  位依次顺序从高位左移至寄存器中.  $M-1$  个寄存器的值即为第  $j$  级蝶形运算第  $c_{M-2} \dots c_1c_0$  个蝶形单元所用旋转因子对应的地址. DIF-FFT 算法旋转因子的硬件实现就更加简单, 只需采用  $M-1$  个移位寄存器, 设置初值  $c_{M-2} \dots c_1c_0$ . 第  $j$  级蝶形运算第  $c_{M-2} \dots c_1c_0$  个蝶形单元所用旋转因子对应的地址, 可通过为将寄存器左移  $j$  并且低位补零来生成.

3 基 2 FFT 地址生成算法

在  $N = 2^M$  的基 2 FFT 算法中, 需要  $M-1$  位蝶形运算单元次序计数器  $CNT(c_{M-2} \dots c_0)$ , 其进位位为  $c$ ,  $\lceil \log_2 M \rceil$  ( $\lceil \cdot \rceil$  代表向上取整) 位蝶形运算级数计数器  $CNT(j)$ , 两个  $M-1$  位移位寄存器作为蝶形

单元两结点数据地址发生器( $a_{M-1} \dots a_1 a_0$ ), ( $b_{M-1} \dots b_1 b_0$ ), 以及  $M-1$  位移寄存器( $d_{M-2} \dots d_1 d_0$ )用于生成蝶形单元所用旋转因子的地址. 基 2 倒序输入、顺序输出 DIT-FFT 地址生成算法有如下 5 个实现步骤.

- (1) 初始化设置  $CNT(c_{M-2} \dots c_0) = 0, CNT(j) = 0$ .
- (2) 设置寄存器的初值( $a_{M-1} \dots a_1 a_0$ ) = ( $c_{M-2} \dots c_0 0$ ), ( $b_{M-1} \dots b_1 b_0$ ) = ( $c_{M-2} \dots c_0 1$ ), ( $d_{M-2} \dots d_1 d_0$ ) = ( $0 \dots 00$ ).
- (3) 根据级数计数器计数值  $j$ , 将两个生成结点数据地址寄存器低  $j+1$  位进行循环左移  $j$  位, 即可得( $a_{M-1} \dots a_1 a_0 / b_{M-1} \dots b_1 b_0$ ) $_j = (c_{M-2} \dots c_j 0 c_{j-1} \dots c_0 / c_{M-2} \dots c_j 1 c_{j-1} \dots c_0)$ ,. 同时, 将  $CNT(c_{M-2} \dots c_0)$  低  $j$  位从高位顺序移入生成旋转因子的寄存器中, 可得( $d_{M-2} \dots d_1 d_0$ ) $_j = (c_{j-1} c_{j-2} \dots 00)$ .
- (4) 根据所得地址读取数据及旋转因子, 完成一次蝶形运算.
- (5)  $CNT(c_{M-2} \dots c_0) = CNT(c_{M-2} \dots c_0) + 1, CNT(j) = CNT(j) + c$ . 如果  $j = M$ , 表示 FFT 运算结束; 否则, 转入步骤(2).

实现  $N = 2^M$  的基 2 顺序输入、倒序输出 DIF-FFT 地址生成算法, 只是在步骤(2) 生成结点数据及旋转因子的移位寄存器设置不同初值, 以及在步骤(3) 中对这些移位寄存器作不同的操作.

4 结束语

通过对 8 点基 2 DIT-FFT, DIF-FFT 流图的分析, 总结出 FFT 算法地址产生规律, 提出了一种地址生成新算法, 为 FFT 处理器的设计提供了一定的参考价值.

参考文献:

[ 1 ] 陆旦前, 陈建平, 陈晓勇. FFT 算法的 FPGA 设计[J]. 现代电子技术, 2007, 245(6): 178-180.  
[ 2 ] 马余泰. FFT 处理器无冲突地址生成方法[ J ]. 计算机学报, 1995, 18( 11): 875-880.  
[ 3 ] 罗文哲, 徐葭生. 一种新型的 FFT 地址发生器集成电路的设计[ J ]. 电子学报, 1994, 22(5): 32-38.  
[ 4 ] 杨 靓, 黄 巾, 刘红侠, 等. 一种高效的 FFT 处理器地址快速生成方法[ J ]. 信号处理, 2004, 20(3): 251-257.  
[ 5 ] 黄 巾, 杨 靓, 刘红侠, 等. 高基 FFT 处理器高效地址生成算法[ J ]. 信号处理, 2005, 21(1): 74-78.  
[ 6 ] 陆 涛, 何佩琨, 陈 禾, 等. 一种快速 FFT 处理器的地址生成方法[ J ]. 北京理工大学学报, 2006, 26(1): 68-71.

Address Generation Algorithm for a FFT Processor

ZHANG Yu-zhao, QIU Ying-qiang

( College of Information Science and Engineering, Huaqiao University, Quanzhou 362021, China)

**Abstract:** By analyzing the flow process through diagram radix-2 of 8 points decimation-in-time fast Fourier transform (DIT-FFT), decimation-in-frequency fast Fourier transform (DIF-FFT) and summarizing the rule of address generation for butterfly calculation and twiddle factor, a fast address generation algorithm of operands and twiddle factors for radix-2 FFT processor is presented, the address of operands and twiddle factors for butterfly calculation will be generated quickly by shift operation of some related register.  
**Keywords:** fast Fourier transform; processor; butterfly calculation; address generation

(责任编辑: 黄仲一 英文审校: 吴逢铁)