

FFT 算法的 DSP 实现

刘美容^{1,2}

(1 湖南师范大学 物理与信息科学学院, 湖南 长沙 410081;

2 合肥工业大学 电气工程博士后流动站, 安徽 合肥 230009)

摘 要: 在介绍 FFT 算法基本原理的基础上, 提出一种基于 DSP 芯片实现 FFT 算法的方法. 算法程序研究表明该方法可实现 FFT 运算, 并能提高其运算速度和精度.

关键词: 快速傅里叶变换; 数字信号处理; TMS320C5402

中图分类号: TN911.7

文献标识码: A

文章编号: 1000-7180(2015)01-0076-04

DSP Implementation of FFT Algorithm

LIU Mei-rong^{1,2}

(1 College of Physics and Information Science, Hunan Normal University, Changsha 410081, China ;

2 Electric Engineering Postdoctoral Center, Hefei University of Technology, Hefei 230009, China)

Abstract: On the basis of analyzing the basic principle of FFT, the realization method of FFT based on DSP TMS320C5402 is presented. It is proved that the algorithm can realize FFT mathematical operation and improve its speed and precision of operation.

Key words: fast fourier transform; digital signal processing; TMS320C5402

1 引言

傅里叶变换是将信号从时域形式变换为频域形式表示的一种变换方法. 由于信号的某些特性在时域表现得不明显, 而通过傅里叶变换将其变换到频域形式后, 这些特性能清楚地表现出来, 因此在信号处理中它是分析信号时经常采用的一种方法.

随着信息技术的不断发展, 在计算机系统中, 信号的处理以数字形式来完成, 因此, 离散傅里叶变换(DFT)被提出. DFT 是将离散(数字)信号从时域变换为频域的一种变换方法, 但由于其运算量相当大, 因此在实际应用中常使用快速傅里叶变换(FFT). 数字计算系统的问世以及 FFT 的出现共同推动了 DFT 的应用, 能满足了工程中实时处理的需求, 因此使 DFT 在实际工程中得到了广泛的应用^[1-3]. 然而, 虽然 FFT 较之前的 DFT 在运算量方面减少很多, 但普通的单片机、微处理器、CPU 却均

难以实现 FFT 多点、实时运算的要求. 但是数字信号处理器(DSP)一般能在一个指令周期内完成一次乘法、加法处理, 并且有专门实现 FFT 的指令, 这大大提高了 FFT 算法在 DSP 上实现的速度^[4]. 由于 DSP 具有运算速度快和精度高等优点, 它的出现使 FFT 算法的实现变得尤为方便. 因此本文提出利用 DSP 来实现 FFT 的方法.

2 傅里叶变换原理

2.1 傅里叶变换

傅里叶(Fourier)变换是实现将信号从时域到频域的变换^[5], 它建立了信号时域(自变量为“时间”)与频域(自变量为“频率”)之间的对应关系. 因此, 当信号的自变量取值不同(连续值或离散值)时, 就会出现各种不同形式的傅里叶变换^[6]. 对于连续时间非周期信号 $x(t)$ 的傅里叶变换人们将之定义为 $X(j\omega)$, 则:

收稿日期: 2014-02-02; 修回日期: 2014-03-08

基金项目: 国家自然科学基金(60876022); 湖南省科技计划(2012FJ4347)

$$X(j\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt \quad (1)$$

式中, $X(j\omega)$ 称为 $x(t)$ 的频谱密度函数. 同时,

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(j\omega)e^{j\omega t} d\omega \quad (2)$$

2.2 离散傅里叶变换

当对信号 $x(t)$ 以抽样间隔 T 进行抽样, 则可得到离散时间信号 $x(kT)$, 则有

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(j\omega)e^{j\omega t} d\omega \quad (3)$$

当信号自变量“时间”和“频率”均离散时, 此时的傅里叶变换被称之为离散傅里叶变换, 即 DFT. DFT 变换对被定义为

$$X[m] = \sum_{k=0}^{N-1} x[k]e^{-j\frac{2\pi}{N}mk}, m = 0, 1, \dots, N-1 \quad (4)$$

$$x[k] = \frac{1}{N} \sum_{m=0}^{N-1} X[m]e^{j\frac{2\pi}{N}mk}, k = 0, 1, \dots, N-1 \quad (5)$$

式中, $W_N = e^{-j\frac{2\pi}{N}}$ 被称为旋转因子. 对于信号 $x(k)$ 和 $X(m)$, 它们都是离散信号, 而且都是有限长信号. 从计算量来讲, 从式(4)和式(5)可以看出计算 N 点序列的 DFT 和 IDFT 的复数乘法和复数加法次数分别为 N^2 和 $N(N-1)$.

2.3 快速傅里叶变换

DFT 虽然在理论上论述了如何利用数字化方法进行信号的分析与处理, 但其计算量大, 计算效率非常低, 这大大地限制了 DFT 的应用. 因此 20 世纪中期以来, 信号处理快速算法研究取得了进展. 1965 年 Cooley 和 Tukey 提出了一种 DFT 的快速算法, 即快速傅里叶算法 (Fast Fourier Transform, FFT), 这使得 DFT 的快速运算随着电子技术的发展得到有效解决. 由于 FFT 算法充分利用了 W_N 的特性, 因此 FFT 算法将计算 N 点序列的 DFT 和 IDFT 的复数乘法和复数加法次数分别降为 $(N/2) \log_2 N$ 和 $N \log_2 N$. FFT 与 DFT 所需乘法次数如图 1 所示.

基于时间抽取 FFT 算法的原理是在时域对序列 $x[k]$ 逐次分解为两个子序列, 利用 W_N 的特性, 将两个子序列的 DFT 逐渐合成以得到整个序列的 DFT, 提高 DFT 的运算速度^[2]. 由 DFT 的定义可知,

$$\begin{aligned} X[m] &= \text{DFT}\{x[k]\} = \sum_{k=0}^{N-1} x[k]W_N^{km} \\ &= \sum_{k=0}^{N-1} x[k]W_N^{km} + \sum_{k=N}^{2N-1} x[k]W_N^{km} \end{aligned}$$

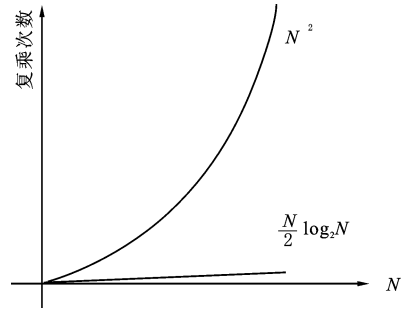


图 1 FFT 与 DFT 所需乘法次数比较

$$\begin{aligned} &= \sum_{r=0}^{N/2-1} x[2r]W_N^{2rm} + \sum_{r=0}^{N/2-1} x[2r+1]W_N^{(2r+1)m} \\ &= \sum_{r=0}^{N/2-1} x_1[r]W_N^{2rm} + \sum_{r=0}^{N/2-1} x_2[r]W_N^{(2r+1)m} \end{aligned} \quad (6)$$

由于旋转因子 $W_N^{2km} = W_N^{km}$, 即其具有可约性, 故式(6)可表示成如下形式:

$$\begin{aligned} X[m] &= \sum_{r=0}^{N/2-1} x_1[r]W_N^{rm} + \sum_{r=0}^{N/2-1} x_2[r]W_N^{rm} \\ &= X_1[m] + W_N^m X_2[m] \end{aligned} \quad (7)$$

式中, $X_1[m]$ 和 $X_2[m]$ 分别为 $N/2$ 点 DFT, 即

$$\begin{aligned} X_1[m] &= \sum_{r=0}^{N/2-1} x_1[r]W_{N/2}^{rm} \\ &= \sum_{r=0}^{N/2-1} x[2r]W_{N/2}^{rm}, \\ m &= 0, 1, \dots, \frac{N}{2} - 1 \end{aligned} \quad (8)$$

$$\begin{aligned} X_2[m] &= \sum_{r=0}^{N/2-1} x_2[r]W_{N/2}^{rm} \\ &= \sum_{r=0}^{N/2-1} x[2r+1]W_{N/2}^{rm}, \\ m &= 0, 1, \dots, \frac{N}{2} - 1 \end{aligned} \quad (9)$$

另外由于限长序列的 DFT $X_1[m]$ 和 $X_2[m]$ 都隐含有周期性, 且均以 $N/2$ 为周期, 因此有 $X_1[m + N/2] = X_1[m]$ 和 $X_2[m + N/2] = X_2[m]$, 并且 $W_N^{m+N/2} = e^{-j\pi} W_N^m = -W_N^m$, 所以式(7)可表示成式(10)的形式:

$$\begin{aligned} X[m + N/2] &= X_1[m + N/2] - W_N^{m+N/2} X_2[m + N/2] \\ &= X_1[m] - W_N^m X_2[m] \end{aligned} \quad (10)$$

由式(7)和式(10)可得 $X[m]$ 为:

$$\begin{cases} X[m] = X_1[m] + W_N^m X_2[m] \\ X[m + N/2] = X_1[m] - W_N^m X_2[m] \end{cases}, m = 0, 1, \dots, N/2 - 1 \quad (11)$$

由式(11)可知, 将 $x[k]$ 分解为偶数序列 $x_1[k]$ 与奇数序列 $x_2[k]$ 之和, 就可以由这两个子序列对

应的 DFT 综合为序列 $x[k]$ 的 DFT, 且该过程相当简单, 只需要作 $N/2$ 次复数乘法 $W_N^m X_2[m]$ 即可, 因此能有效提高求取 $x[k]$ 的 DFT 的运算速度.

基于时间抽取 FFT 的算法的分解过程可用信号流程图来表示, 如图 2 所示^[2], 因为这个图形呈蝶形, 所以被称为蝶形运算结构. 依此类推, 经过 $\log_2 N$ 次分解, 最后将 N 点 DFT 分解为 $N/2$ 个两点 DFT.

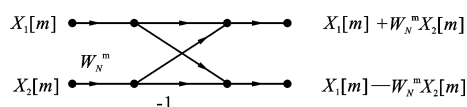


图 2 基于时间抽取 FFT 蝶形运算流程图

当 $N = 2$ 时, 2 点序列的 DFT 为

$$X[0] = x[0] + x[1], X[1] = x[0] - x[1],$$

因此其 DFT 运算流程图可用图 3 表示. 同理, 可推出当 $N = 4$ 时 DFT 运算流程图如图 4 所示.

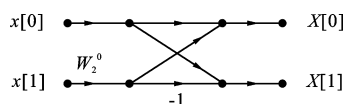


图 3 2 点序列 DFT 运算流程图

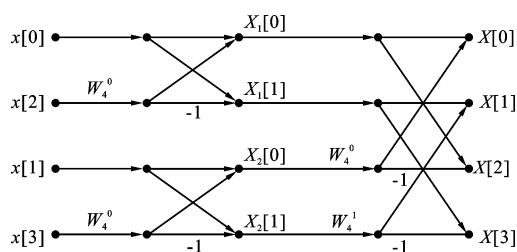


图 4 4 点序列 DFT 运算流程图

因此, FFT 算法的原理是通过短序列的 DFT 逐步合成长序列的 DFT, 即利用许多小的容易实现的变换来完成大规模的变换, 以降低了运算次数, 从而提高其运算速度. 所以, 序列的 FFT 运算就是 DFT 运算, 它们的运算结果完全是相同的, 均为序列 $x[k]$ 的频谱, 且 FFT 的运算速度有了很大的提高.

3 基于 DSP TMS320C5402 的 FFT 实现

由以上分析表明, FFT 算法的主要特点是多次乘积、累加运算, 及位码倒序、间接寻址. 因此, 普通处理器和单片机都很难实现 FFT 算法, DSP 作为专门的数字信号处理器, 它能在一个指令周期内完成一次乘积、累加运算, 而且具有间接寻址和位倒序寻址功能. 因此相对其他的处理器而言, 在数字信号

处理方面, DSP 具有绝对的优势. 本文采用 TI 公司 C5000 系列 TMS320C5402 芯片来实现 FFT 算法.

TI 公司开发的 TMS320C54x 芯片是为实现低功耗、高性能而专门设计的定点 DSP 芯片, 主要应用在无线通信等应用型领域. 由于 C54x 具有单周期乘法加法指令、重复指令, 并且具有位倒序寻址等功能, 因此其最高运算速度可达 40MIPS^[7]. 以 DSP 芯片为核心处理器时, TI 公司专门为其开发的集成开发环境 (CCS), 开发语言可以是汇编语言编程, 也可以是 C/C++ 编程和二者混合编程. 为了充分利用 DSP 芯片特有的优点, 如片内 RAM 和比特反转寻址方式等. 本文中, FFT 运算程序采用汇编语言编程, 而主程序部分采用 C 语言. 由于整个程序相当复杂, 文中只给出利用汇编语言编写的 FFT 运算中位倒序程序及模拟输入数据文件的 C 语言程序.

其中 FFT 运算中位倒序程序如下:

```
. title "fft.asm"
. mmregs
. copy "coeff.inc"
. defstart
sine;. usect "sine", 512
cosine;. usect "cosine", 512
fft_data;. usect "fft_data", 1024
d_input;. usect "d_input", 1024
fft_out;. usect "fft_out", 512
STACK. usect "STACK", 10
K_FFT_SIZE .set 32
K_LOGN .set 5
PA0. set 0
PA1. set 1
. asgAR2, REORDER
. asgAR3, ORI_INPUT
. asgAR7, DATA_PROC_BUF
start:
SSBXFRCT
STM# STACK+10, SP
STM# d_input, AR1;从 PA1 口输入数据
RPT# 2*K_FFT_SIZE-1
PORTPA1, *AR1+
STM# sine, AR1;将正弦系数从程序存储器传送到数
据存储器
RPT# 511
MVPDsine1, #AR1+
STM# cosine, AR1;将余弦系数从程序存储器传送到
数据存储器
RPT# 511
MVPDcosin1, #AR1+
```

```

STM# d_input, ORI_INPUT
STM# fft_data, DATA_PROC_BUF
MVMM# DATA_PROC_BUF, REORDER
STM# K_FFT_SIZE-1, BRC
RPTBDbit_rev_end-1
STM# K_FFT_SIZE, AR0
MVDD* ORI_INPUT+, * REORDER+
MVDD* ORI_INPUT-, * REORDER+
MAR* ORI_INPUT+0B
bit_rev_end:

```

正弦和余弦系数表由 coeff. inc 文件给出, 可通过伪指令. copy 将系数和程序代码进行汇编. 在以上系数表文件中共有 1 024 个复数点 FFT 的正弦、余弦系数各 512 个, 因此利用此系数表对 8~1 024 点 FFT 进行运算. 根据点数 N 值不同时, 需修改 fft. asm 文件中的两个常数: K_FFT_SIZE 和 K_LOGN , 即可改变不同点数的 FFT 运算, 如当 $N=128$, 则

```

K_FFT_SIZE .set128;
K_LOGN .set7

```

当 N 大于 1 024 时, 除了修改以上两个常数外, 还需要增加系数表和修改命令文件.

用 C 语言程序建立 FFT 算法的模拟输入数据, 具体程序如下:

```

/* 文件名: sindatagen. c */
#include "stdio. h"
#include "math. h"
main( )
{
    inti;
    floatf[256];
    FILE* fp;
    if((fp=fopen("c:\\tms320c54\\sindata", "wt"))==
    NULL)
    {
        printf("can't openfile !\n");
        exit(0);
    }
    for(i=0; i<=255; i++)
    {
        f[i]=sin(2*3. 14159265*i/256. 0);
        fprintf(fp, "    word    %ld\n", (log)(f[i]*
        16384));
    }
    fclose(fp);
}

```

若输入信号振幅设为 1 000 左右, 信号频率是 300 Hz, 根据公式 $f=kf_s/N$, f 是原始信号的频率, k 表示峰值出现的位置, f_s 是采样频率, N 是 FFT 的点数. 图 5 说明 1 024 点时如何观察收到的数据

和显示的图像, 通过 FFT 计算后得到信号幅度谱如图 6 所示.

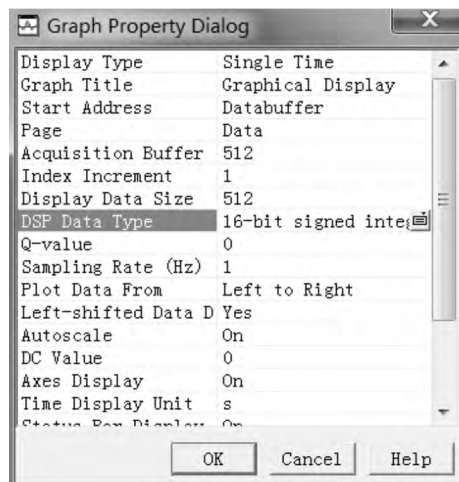


图 5 设置数据和显示的图像形式

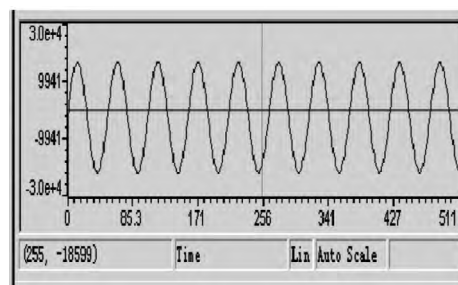


图 6 信号幅度谱

4 结束语

实例研究表明, 此程序在定点 DSP 芯片 TMS320C5402 中运行良好, 速度快且运算结果正确无误, 能满足一般的信号处理和工业控制对精度和实时的要求, 具有较高的学术价值和良好的应用前景.

参考文献:

- [1] 陈学, 徐建华, 叶甜春, 等. 基于 FFT 的微弱 GPS 信号捕获算法[J]. 微电子学与计算机, 2010, 27(3): 98-101.
- [2] 潘邦淦, 毛志刚, 谢憬. 基于本地码插值修正的 GPS 信号捕获方法[J]. 微电子学与计算机, 2010, 27(11): 94-98.
- [3] 桑红石, 高伟. 高效可配置浮点 FFT 处理器设计[J]. 微电子学与计算机, 2012, 29(4): 36-40.
- [4] 刘艳萍. DSP 技术原理及应用教程[M]. 3 版. 北京: 北京航空航天大学出版社, 2012.

(下转第 84 页)

使其支持嵌入式 C 语言的语法特性的方法. 经过测试验证, 这种设计方法可降低移植 LCC 的难度和出错的概率, 且能快速开发目标处理器的编译工具, 是一种行之有效的 LCC 移植方案.

参考文献:

- [1] Malinowski A, Hao Y. Comparison of embedded system design for industrial applications[J]. Industrial Informatics, IEEE Transactions on, 2011, 7(2): 244-54.
- [2] Chen H C, Huang Y M, Su C H, et al. Implementation of SOPC based telecom & datacom for monitoring wireless sensor networks[J]. Telecommunication Systems, 2013, 52(4): 2325-2333.
- [3] Scharwaechter H, Youn J M, Leupers R, et al. A code-generator generator for multi-output instructions [C]//Proceedings of the 5th IEEE/ACM international conference on Hardware/software codesign and system synthesis. Salzburg, Austria: ACM, 2007: 131-137.
- [4] Povazan I, Popovic M, Djukic M, et al. A retargetable C compiler for embedded systems[C]//2013 IEEE 3rd Eastern European Regional Conference on the Engineering of Computer Based Systems, ECBS-EERC 2013. Budapest, Hungary: IEEE Computer Society, 2013: 48-54.
- [5] 周志雄, 何虎, 杨旭, 等. ASIP 处理器的可重定位编译器的设计[J]. 微电子学与计算机, 2007, 24(1): 38-41.
- [6] 杨庆庆, 周晓方, 杨鸿. 自定义指令集处理器及其工具链设计[J]. 小型微型计算机系统, 2011(2): 333-341.
- [7] Yue H, Lai M C, Dai K, et al. Design of a configurable embedded processor architecture for DSP functions[C]//11th International Conference on Parallel and Distributed Systems Workshops, ICPADS 2005. Fukuoka, Japan: Institute of Electrical and Electronics Engineers Computer Society, 2005: 27-31.
- [8] Fraser C W, Hanson D R. Simple register spilling in a retargetable compiler[J]. Software: Practice and Experience, 1992, 22(1): 85-99.

作者简介:

孙桂玲 女, (1964-), 教授, 博士生导师. 研究方向为无线传感器网络、物联网等.

范孟秋(通讯作者) 男, (1989-), 硕士研究生. 研究方向为无线传感器网络. E-mail: fanmengqiu@mail.nankai.edu.cn.

杨长锐 男, (1988-), 硕士研究生. 研究方向为无线传感器网络.

毕晓朕 男, (1992-). 研究方向为无线传感器网络.

(上接第 79 页)

- [5] 陈后金. 数字信号处理[M]. 2 版. 北京: 高等教育出版社, 2008.
- [6] 赵祥军. 莫尔斯信号实时检测和识别方法研究[D]. 黑龙江: 哈尔滨工程大学, 2007.

- [7] TI Corp. TMS320C5X user's guide[M]. USA: Texas Instruments, 1991.

作者简介:

刘美容 女, (1977-), 博士后, 讲师. 研究方向为信号处理、故障诊断等. E-mail: liumr12@aliyun.com.