基于DSP的实数FFT算法研究与实现

陈恒亮 蒋 勇
(深圳技师学院电气工程系,深圳 518040)

摘要 介绍了一种实数快速傅里叶变换(FFT)的设计原理及实现方法,利用输入序列的对称性,将$2N$点的实数FFT计算转化为N点复数FFT计算,然后将FFT的N点复数输出序列进行适当的运算组合,获得原实数输入的$2N$点FFT复数输出序列,使FFT的运算量减少了近一半,很大程度上减少了系统的运算时间,解决了信号处理系统要求实时处理与傅里叶变换运算量大之间的矛盾.同时,给出了在TMS320VC5402 DSP上实现实数FFT的软件设计,并比较了执行16,32,64,128,256,512,1024点实数FFT程序代码与相同点数复数FFT的程序代码运行时间.经过实验验证,各项指标均达到了设计要求.

关键词 数字信号处理器,快速傅里叶变换,蝶形运算

引言

数字信号处理器(DSP)是一种可编程的高性能处理器,近年来发展很快.它不仅适用于数字信号处理,而且在图像处理、语音处理、通信等领域得到了广泛的应用.通用的微处理器在运算速度上很难适应信号实时处理的要求.DSP处理器中集成有高速的乘法器硬件,能快速地进行大量数据的乘法和加法运算,TMS320VC5402[1,2](以下简称C5402)就是TI公司近年推出的一款高性能DSP,它能在一个指令周期内完成一次乘法和一次加法,已经成为当今应用最为广泛的DSP芯片之一.

傅里叶变换是一种将信号从时域变换到频域的变换形式,是声学、图像、电信和信号处理等领域中一种重要的分析工具.其中,离散傅里叶变换(DFT)更是数字信号处理领域不可缺少的工具之一[3,4],特别是快速傅里叶变换(FFT)的出现使得DFT在实际应用中得到了广泛的应用.通常情况下,都是假设输入的数字序列是复数的,但是,输入信号是纯实数的更符合实际情况,如果利用输入序列的对称性,将$2N$点的实数FFT计算转化为N点复数FFT计算,然后将FFT的N点复数输出序列进行适当的运算组合,获得原实数输入的$2N$点FFT复数输出序列,这样使得FFT的运算量减少了近一半,效率可比一般的FFT提高近一倍.通过这种变换处理,可以很大程度上减少了系统的运算时间,满足信号处理系统对实时处理和运算精度的要求.

2005-02-25收到第1稿,2005-04-20收到修改稿.

1 实数FFT算法构成

FFT的实质是将较长序列的DFT运算逐级分解为较短序列的DFT运算.序列$f(n)$的DFT定义为

$$F(k)=\sum_{n=0}^{N-1}f(n)e^{-j2\pi nk/N}=\sum_{n=0}^{N-1}f(n)W^{nk} \quad (1)$$

其中$W=e^{-j2\pi/N}$为旋转因子.

假设$g(n)(n=0,1,\cdots,2N-1)$是输入为$2N$点的实数序列,作如下变换将$2N$点的实数DFT变为N点的复数DFT

取:$x_1(n)=g(2n),x_2(n)=g(2n+1)$.

设:$x(n)=x_1(n)+jx_2(n)$,则$X(k)=X_1(k)+jX_2(k)$.

由傅里叶变换的特点能推出

$$X_1(k)=\frac{1}{2}\{X(k)+X^*(N-k)\}$$

$$X_2(k)=\frac{1}{2j}\{X(k)-X^*(N-k)\}$$

$$k=0,1,\cdots,N-1 \quad (2)$$

其中$*$是复共轭算子.

用2个N点DFT表示一个$2N$点DFT

$G(k)=\mathrm{DFT}[g(n)]=\mathrm{DFT}[g(2n)]+$

$$\mathrm{DFT}[g(2n+1)] = X_1(k) + W_{2N}^k X_2(k) \tag{3}$$

将式(2)代入式(3),有

$$G(k) = \frac{1}{2}\{X(k) + X^*(N-k)\} + W_{2N}^k \frac{1}{2\mathrm{j}}\{X(k) - X^*(N-k)\} = X(k)A(k) + X^*(N-k)B(k) \tag{4}$$

其中 $A(k) = \frac{1}{2}(1 - \mathrm{j}W_{2N}^k)$,$B(k) = \frac{1}{2}(1 + \mathrm{j}W_{2N}^k)$,$k = 0,1,\cdots,N-1$.

如果输入为纯实数,其 DFT 结果具有复共轭对称的特点,同时,由于 DFT 具有周期性,$X(k+N) = X(k)$,则 $X(N) = X(0)$.根据这些特点,能推导出输入为实数的 DFT 另一半结果.将式(4)的乘法运算展开,并使计算结果实部和虚部分离,得到如下结果

$$\begin{cases} Gr(k) = Xr(k)Ar(k) - Xi(k)Ai(k) + Xr(N-k)Br(k) + Xi(N-k)Bi(k) \\ Gi(k) = Xi(k)Ar(k) + Xr(k)Ai(k) + Xr(N-k)Bi(k) - Xi(N-k)Br(k) \end{cases} \tag{5}$$

其中 $k = 0,1,\cdots,N-1$;$X(N) = X(0)$;

$$\begin{cases} Gr(k) = Xr(0) - Xi(0) \\ Gi(k) = 0 \end{cases} \tag{6}$$

其中 $k = N$;

$$\begin{cases} Gr(2N-k) = Gr(k) \\ Gi(2N-k) = -Gi(k) \end{cases} \tag{7}$$

其中 $k = 0,1,\cdots,N-1$.

2　软件实现

软件设计的具体过程可以归纳为:将 $2N$ 点实数 FFT 输入序列进行适当的组合以形成 N 点复数序列;计算复数序列的 FFT;将 FFT 的 N 点复数输出序列按式(5)～式(7)进行适当的运算组合,恢复为原来实数输入的 $2N$ 点 FFT 复数输出序列.

2.1　软件设计需要解决的问题

要实现高效的实数 FFT 运算,需要解决好以下几个问题:定标、制表、即位运算.

2.1.1　定标

由于C5402是16 bits定点型DSP,因此必须采用定标算法防止计算过程中的溢出.如果要求 N 点 FFT 的结果不发生溢出,则要求时域信号的幅度上限小于 $1/N$,因此定标算法就是对时域信号进行除以 N 的运算,这主要有以下 3 种实现方式:

1) 预定标:数据输入端除以 N,以使 FFT 的结果不发生溢出;

2) 逐级定标:由于 FFT 的基本运算单元是碟形运算,每个基本的碟形运算需要一次复数乘法和两次复数加法,在每个碟形运算之后除以 2 就可以防止溢出.

3) 块浮点定标:只对发生溢出的碟形运算进行除以 2 的处理.

块浮点定标的精度最高,但它需要逐级进行溢出和移位选择,因此实现最复杂;预定标只在数据输入的一级上进行移位,实现最简单,但性能最差;逐级定标不需要判断和选择,同时,由于 C5402 有专门对运算进行定标的滚筒移位器,它能在进行乘加运算的同时完成定标,不需要额外占用 CPU 时间,其性能比块浮点定标略差,在对 FFT 运算精度要求不是特别高时,逐级定标是最佳选择,所以,这里选择逐级定标方法,在计算完 FFT 后再对结果恢复原值.

2.1.2　制表

从FFT的第4级开始,将每一级运算所需的旋转因子事先计算出来,并按照实数 FFT 运算要求的顺序存放好,形成一个旋转因子表,在计算到 FFT 的特定级时到旋转因子表的对应位置查表.这种方法的优点是节约时间,缺点是占用内存较多,在对实时性要求比较高的场合,以占用一定内存来提高运行效率是一种比较可取的方法.

2.1.3　即位运算

即位运算是指当把数据存入存储器后,每一级运算的结果都存在相应的输入存储器中,直到计算出最终结果.系统中,使用时间抽选奇偶分解 FFT 算法,将 $2N$ 点实数 FFT 输入序列进行适当的组合以形成 N 点复数序列,将这 N 点复数序列进行倒位序排列,结果还存在原 N 点复数序列所在的存储单元中,随后逐级进行碟形运算,后一级的结果存入前一级对应数据所在的存储单元中,直至算得最后结果.

2.2　具体软件编程

在进行实数 FFT 软件设计时,首先将 $2N$ 点实数 FFT 输入序列进行适当的组合以形成 N 点复数序列,然后调用倒位序子程序 cbrev 将 N 点复数序

列进行倒位序排序，以使得最后结果是实数 FFT 要求的正常序列.

图 1 为实数 FFT 系统程序流程图.

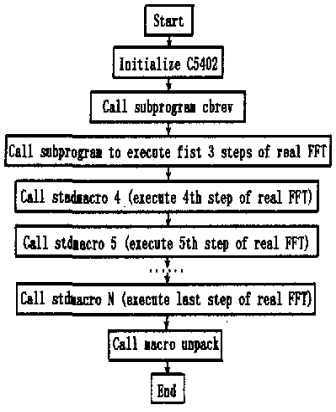

图 1 实数 FFT 系统程序流程图

Fig. 1 Program flow chat of real FFT system

由于 FFT 的前 3 级运算的旋转因子比较简单，相应的运算量比较小，为了最大限度地提高系统的运行效率，采用前 3 级运算与 FFT 第 4 级以后的运算分开设计：将 FFT 的前 3 级运算放在一起单独组成子程序；由于 FFT 第 3 级以后的碟形运算有相同的规律，将其以宏(stdmacro stage,s1,s2,idx,sin,cos) 的形式编程，在进行相应级的运算时传递当前级 FFT 运算的参数(当前执行 FFT 的第 stage 级、该级所含群的个数 s1、该级运算每个群所含碟形运算的个数s2、该级运算每个碟形运算两个输入端数据间的偏移量 idx、该级每个群在计算中所需旋转因子的首地址 sin/cos) 执行 FFT 计算.

将 FFT 的 N 点复数输出序列按照式(5) ~ 式(7) 所示实数 FFT 算法构成进行适当的运算组合，恢复为原来实数输入的 $2N$ 点 FFT 复数输出序列. 这一级运算是以宏(unpack sin,cos) 的形式编程，sin 和 cos 为这一级运算所需要的旋转因子首地址.

3 实验分析

表 1 给出了一组使用所提方法得到的 16 点实数 FFT 与 MATLAB 的内部函数 fft(x) 算得结果的对比关系.

由于在要求对信号进行实时处理的实际运用中，FFT 运算的点数通常小于或等于 1024 点，基于 C5402 DSP(C5402 的工作时钟频率为 100MHz)，表 2 给出了应用文中方法实现 16,32,64,128,256,512,1024 点实数 FFT 与对应点复数 FFT 的程序代码长度及运行时间的比较.

从表 1 和表 2 中可以看出，文中提出的基于 DSP 的实数 FFT 算法在处理 1024 点实数 FFT 时不到 600μs 即可完成，同时，该算法还具有很高的精度(数据误差小于 0.5%)，并且如果将该算法扩展为 32bits 实数 FFT，将会能满足更高的精度要求. 结果显示基于 DSP 的实数 FFT 算法完全可以满足信号处理系统对实时处理和数据精度的要求，有很大的应用价值.

表 1 实验数据

Table 1 Experimental data

No.	Raw data(Q15)	Data from MATLAB Q(15)	Data from DSP Q(15)
1	21061	85544.0000000000	85552
2	- 3624	- 33222.6878097975 - 51863.8120423701i	- 33232 - 51872i
3	7564	4615.3342594059 - 73761.8437743444i	4608 - 73776i
4	19130	- 38774.4120260345 + 88354.1125079538i	- 38784 + 88352i
5	27641	47276.0000000000 - 23036.0000000000i	47264 - 23040i
6	15609	- 54211.8923457129 - 64882.8175987058i	- 54224 - 64896i
7	- 21215	97240.6657405941 - 69005.8437743444i	97232 - 69008i
8	- 6180	96308.9921815449 + 21351.2578509703i	96304 + 21344i
9	28536	12968.0000000000	12960
10	27319	96308.9921815449 - 21351.2578509703i	96304 - 21344i
11	- 5880	97240.6657405941 + 69005.8437743444i	97232 + 69008i
12	25795	- 54211.8923457129 + 64882.8175987058i	- 54224 + 64896i
13	- 28972	47276.0000000000 + 23036.0000000000i	47264 + 23040i
14	- 9642	- 38774.4120260345 - 88354.1125079538i	- 38784 - 88352i
15	20521	4615.3342594059 + 73761.8437743444i	4608 + 73776i
16	- 32119	- 33222.6878097975 + 51863.8120423701i	- 33232 + 51872i

表 2　实数 FFT 与一般复数 FFT 程序代码长度及运行时间比较

Table 2　Comparison of code length and operation time between real FFT and common complex FFT

Points of FFT	Code length of real FFT(word)	Operation time of real FFT(μs)	Code length of complex FFT(word)	Operation time of complex FFT(μs)
16	211	3.64	191	5.22
32	253	7.41	239	12.33
64	301	17.6	287	27.72
128	349	39.16	335	67.95
256	397	85.7	383	151.42
512	445	208.81	431	380.49
1024	493	557.16	479	920.98

参 考 文 献

1　TMS320C54x DSP Reference Set Volume 1: CPU and Peripherals. Texas Instrument Inc, 1999

2　TMS320C54x Assembly Language Tools User's Guide. Texas Instrument Inc, 1999

3　[美]布莱赫特.数字信号处理的快速算法.北京:电子工业出版社,1992(Rright Hert. Rapid algorithms of digital signal processing. Beijing: Electronic Industrial Press, 1992 (in Chinese))

4　程佩青.数字信号处理教程.北京:清华大学出版社,2001(Cheng Peiqing. Tutorial of digital signal processing. Beijing: Qsinghua University Press, 2001(in Chinese))

DESIGN AND REALIZATION OF REAL FFT BASED ON DSP

Chen Hengliang　Jiang Yong

(*Department of Electrical Engineering, Shenzhen Technician College, Shenzhen*　518040, *China*)

Abstract　The design principle and realization of a real FFT algorithm were proposed. In the proposed algorithm, real FFT computation of $2N$ points were transformed to complex FFT computation of N points according to the symmetry of input serial at first. Then after properly processing the N complex outputs of FFT, the $2N$ FFT complex outputs of original real inputs were gotten. The proposed algorithm can almost half reduce the operation time of FFT and resolve the conflict between the request of real-time operation in digital signal processing system and the lots of accounting in FFT. The software design of real FFT based on TMS320VC5402 DSP was given. The code length and operation time of the real FFT with 16, 32, 64, 128, 256, 512, 1024 points were compared with that of complex FFT. The experiment results proved the correctness of the method.

Key words　digital signal processor, FFT algorithm, butterfly operation

Received 25 February 2005, revised 20 April 2005.