快速傅里叶变换的DSP实现及代码优化

楼天良

（浙江义乌工商学院计算机系　浙江义乌 322000）

摘　要　介绍了时间抽取基2FFT算法的基本原理和特点，并详细分析了FFT算法的DSP实现及程序优化，最后采用MATLAB软件对算法进行了仿真。仿真结果证明，该方法具有精度高、运算速度快等特点。

关键词　快速傅立叶变换，数字信号处理器件，程序优化

Realization and Code Optimization of Fast Fourier Transform Based on DSP

LOU Tian-liang

(Department of Computer , Yiwu Industrial & Commercial College, Yiwu Zhejiang 322000, China)

Abstract　The algorithm theory and features of the DIT radix-2 Fast Fourier Transform is introduced. And the author then study the Fast Fourier Transform process and program optimization with Digital Signal Processor in detail. Finally, the thesis uses MATLAB to do many simulations and research on this algorithm. Simulation results shown that this method has many special features such as high precision and fast.

Keywords　Fast fourier transform, Digital signal processor, Program optimization

1　引言

数字信号处理器(DSP)是一种可编程的高性能处理器，它提供了低成本、低功耗、高性能的处理能力，以DSP为核心是智能化制造技术中智能控制器的发展方向。离散傅里叶变换(DFT)是数字信号处理领域的工具之一，但由于DFT的计算量太大，使其应用受到限制，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应用。虽然FFT比DFT的计算量减少了很多，但用普通单片机来实现FFT多点、实时运算还是比较困难的，若把DSP用于FFT的实现，就可充分发挥两者的优势，从而很好地解决此类问题。

2　快速傅里叶变换原理

离散傅里叶变换为[1]

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

令DFT的长度$N=2^M$，M为正整数。现在将时域序列$x(n)$分为两组，n为偶数是一组，n为奇数是另一组，也即令$n=2r$及$n=2r+1$，$r=0,1,0,1,\cdots,N/2-1$。

$$A(k)=\sum_{r=0}^{N/2-1}x(2r)W_{N/2}^{rk}\ k=0,1,0,1,\cdots,N/2-1\quad(2)$$

$$B(k)=\sum_{r=0}^{N/2-1}x(2r+1)W_{N/2}^{rk}\ k=0,1,0,1,\cdots,N/2-1\quad(3)$$

故(1)式可表示为

$$X(k)=A(k)+W_N^kB(k)\quad(4)$$

$A(k)$和$B(k)$都是$N/2$点的DFT，$X(k)$是N点DFT，因此单用$A(k)$和$B(k)$表示$X(k)$并不完全。但

$$X(k+N/2)=A(k)-W_N^kB(k)\quad(5)$$

这样，可以用$A(k)$和$B(k)$完整表示$X(k)$。图1是$N=8$时$A(k)$，$B(k)$及$X(k)$的关系。综上所述，就将一个N点的DFT分解成了两个$N/2$点的DFT，而且仅最后求和的符号不同，就可同时算出$X(k)$和$X(k+N/2)$。

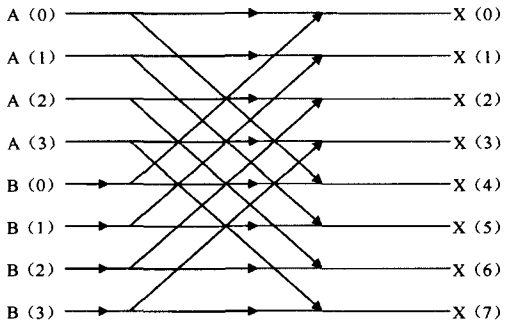

图1　$N=8$时$A(k)$，$B(k)$及$X(k)$的关系图

由于DFT的运算量与其点数的平方成正比，因此将N点DFT分解为两个$N/2$点的DFT会使运算量减少。但并不到此为止，还应该将每一个$N/2$点的DFT再分解为两个$N/4$点的DFT。并且，由于N为2的正整数次幂，因此还可以继续分解下去，直到分解为2点的DFT为止，总共需要进行$\log_2N-1$次分解。

3　快速傅里叶变换方法

FFT采用基二按时间抽取的方法包括两大部分：一部分是倒序重排；另一部分是用3层嵌套的循环来完成$M=\log_2N$次迭代。

①倒序重排：由上面的讨论可知，将N点的DFT分解为两个$N/2$点的DFT时，需将输入序列按照奇偶分为两组。再分解时，又要将每组再按奇偶重排。这样下去，直到最后分解为2点的DFT，输入数据才不再改变顺序。这样做的结果，使得做FFT算法时，输入序列的次序要按其序号的倒序进行重新排列。下面以8点FFT为例，说明输入序列按倒序

楼天良　高级讲师。

重排的情况。现在将图中输入序号以及重排后的序号按二进制写出，如表 1(注：下标“2”表示二进制数)。

表 1　8 点 FFT 输入序列倒序重排表

正序		倒序	
0	000_2	000_2	0
1	001_2	100_2	4
2	010_2	010_2	2
3	011_2	110_2	6
4	100_2	001_2	1
5	101_2	101_2	5
6	110_2	011_2	3
7	111_2	111_2	7

②三层嵌套循环：最内层循环是控制一类蝶形运算，中间层是控制各类蝶形运算，外层是控制级数的运算。三层循环的功能是：最里的一层循环完成蝶形运算，中间的一层循环完成因子 W_N^k 的变化，而最外的一层循环则是完成 M 次迭代过程。对于各次迭代运算，蝶形的跨度是各不相同的，W_N^k 因子中指数 k 的递增数以及它相乘的位置变化也是各不相同的，这些都要用在最外的一层循环中引入几个变量来进行控制。对于 W_N^k，可将全部的系数计算出来，放在固定的存储单元中，形成一个 W_N^k 表，用时查表。这种方法的优点是节约时间，缺点是占用内存较多。

4　TMS320C6000 系列 DSP 的特点

TMS320C6000 系列 DSP 是 32 位浮点 DSP 处理器。它包含 3 个子系列[2]：用于定点计算的 TMS320C62X 系列、TMS320C64X 系列和用于浮点计算的 TMS320C67X 系列、TMS320C6000 系列。时钟频率最高可达到 250MHz。该系列 DSP 包含两个通用的寄存器组 A 和 B，每组有 16 个 32 位寄存器。芯片内含 8 个运算功能单元：两个乘法器(. M1 和. M2)；六个算术逻辑单元(. L1. L2. S1. S2. D1. D2)。所有单元都能独立并行操作。以 TMS320C6701 为例，它的工作频率最高为 167MHz，最快速度可达 8×167＝1336MIPS。实际上，要实现这个速度存在很多瓶颈，主要有下面几种限制：

①功能模块的限制。8 个功能模块能够执行的指令不尽相同。在实际程序中，由于程序流程的限制，指令的位置不能随便调换，因此不可能在每一个时钟周期都让 8 个模块同时工作。程序优化的主要手段就是要提高指令的并行程度，即平均每一周期内同时执行的指令数。

②交叉路径(Cross Path)的限制。每一个功能模块都只能对其所属的寄存器组中的寄存器进行直接操作。例如，. L1只能将结果直接写入寄存器组 A。如果要对另一个寄存器组执行读或写操作，需要用到“交叉路径”，而整个 CPU 中只有两条交叉路径。也就是说，一个周期内至多能同时容纳两个相反方向的交叉读写。

③多周期指令的限制。LD 命令的功能是将数据从存储器读到寄存器中，由. D 模块执行。但执行 LD 命令后必须等待 4 个周期才能得到需要的数据。类似这样的需要多个周期才能完成的命令(例如跳转指令 B)，都成为提高指令并行处理程度的障碍。

④对长数据操作的限制。C6000 指令集只能以 8 比特、16 比特、32 比特或者 40 比特为单位对数据进行操作。

⑤对长循环操作的限制。

5　FFT 在 TM320C6701 上的实现及优化

本文采用 TMS320C6701 实现 FFT 算法。图 2 描述了 FFT 在 TMS320C6701 上的实现流程，其中 FFT 蝶形运算是整个算法中运算量最大的部分。因此，优化的核心任务就是减少每个蝶形运算所消耗的运算周期数。每个蝶形运算包括：三次加载数据操作(load)，因为可以证明一个蝶形中的 4 条支路的度量具有相同的绝对值，所以每次只需要加载一个由 BMU 预先计算的结果；4 次加法操作；2 次比较操作；比较之后的 4 次存储操作。其中，4 次加法操作可以在一个周期内同时完成。针对前面提到的几个障碍，可以用以下的方法分别加以解决[3]：

①解决功能模块的限制，可以用不同的命令相互替代。例如赋值操作 MV 只能用. L，. S 和. D 功能模块完成。如果这些模块都被其它的并行指令占用，可以用乘 1 的方法实现赋值，而乘法指令 MPY 是用. M 单元实现的。类似地，也可以用加零或减零的指令代替 MV 指令。

②解决交叉路径的限制，需要依靠寄存器的分配和倒换，让同一指令涉及到的寄存器尽量处在同一个寄存器组中，减少需要用到交叉路径的机会。

③解决多周期指令的限制，加载数据的结果需要在 4 个周期以后才能得到。为了有效地利用等待的这段时间，在程序设计中把加载数据的指令放在前面的蝶形运算中执行。当进入本次蝶形运算时，就能立即使用加载的新数据。同样，本次蝶形运算也要执行为下一个蝶形运算加载数据的指令。B 指令(跳转指令)的问题可以用类似的方法来处理。

④解决对长数据操作的限制。但 TMS320C6701 不能直接对 64 比特长的数据进行读写操作，所以把 PM 分成两个相同的 32 位数组(PMO 和 PMI)。前者用来存储状态 0-31 位，后者存储状态 32-64 位。当数据长度更大时，也可以用同样的办法来分开存储[4]。

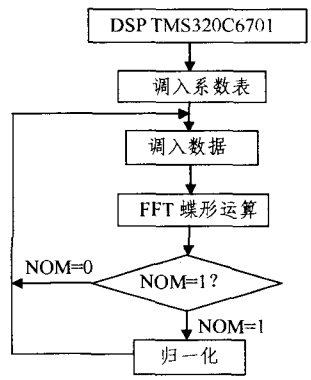

图 2　FFT 在 TMS320C6701 上的实现流程图

⑤运用软件流水方式。软件流水线是在循环语句中调用指令的方法，即安排循环中的多个迭代运算并行执行。在编译 C 代码时，可以选择编译器的-o2 或-o3 选项，则编译器将根据程序尽可能地安排软件流水线。在 FFT 算法中存在大量的循环操作，因此充分地运用软件流水线方式，能极大地提高程序的运行速度。软件流水线在程序编译时自动安排，但如果循环算法编程不合理编译时，将不能运用软件流水线。如何充分利用软件流水线是提高编程效率的关键。因此在编写 FFT 中循环语句时应注意以下原则：1)可以调用内联函

(下转第 268 页)

[11] Cousot P. Automatic Verification by Abstract Interpretation// Fourth International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI'03). LNCS, 2003 (2566):85-108
[12] Comini M, Levi G, Vitiello G. Abstract Debugging of Logic Program // Proceedings of International Worksho Pon Meta-Programming in Logic (META), 1994:440-450
[13] Comini M, Levi G, Vitiello G. Efficient Detection of Incompleteness Errors in the Abstract Debugging of Logic Programs// Proceedings of AADEBUG. 1995:159-174
[14] Cousot P, Cousot R. An abstract interpretation - based framework for software watermarking. In Principles of Programming Languages 2003, POPL'03, 2003:311-324
[15] Zhao L, Gu T, Qian J. Goal-independent Semantics for Path Dependent Analysis of Prolog Programs// Proceedings of 1st IEEE & IFI PInternational Symposium on Theoretical Aspects of Software Engineering (TASE 2007). 2007:261-270
[16] 赵岭忠,古天龙,钱俊彦. 目标独立的 Prolog 程序路径依赖分析语义. 计算机科学,2008,35(2)
[17] Cousot P. Abstract Interpretation Based Formal Methods and Future Challenges. LNCS, 2000:138-156
[18] Clarke E M, Grumberg O, Peled D. Model Checking. Massachusetts: MIT Press, 1999
[19] Clarke E M, Grumberg O, Jha S, et al. Counterexample-guided Abstraction Refinement// Proceedings of Computer Aided Verification. 2000:154-169
[20] Lakhnech Y, Bensalem S, Berezin S, et al. Incremental Verification by Abstraction // Proceedings of TACAS 2001. LNCS, 2001, 2031:98-112
[21] Deransart P. Proof methods of declarative properties of definite programs. Theoretical Computer Science, 1993(118):99-166
[22] Hogger C. Introduction to Logic Programming. Academic London, 1984
[23] Bossi A, Cocco N. Verifying correctness of logic programs// Proceedings of Tapsoft '89. 1989:96-110
[24] Apt K R, Marchiori E. Reasoning about prolog programs: From modes through types to assertions. Formal Aspects of Computing, 1994, 6(6A):743-764
[25] Drabent W. It is declarative// A. Bossi, ed. ILPS post-conference Worksho Pon Verification, Model Checking and Abstract Interpretation. Port Jefferson, Long Island N. Y., USA, 1997
[26] Puebla G, Bueno F, Hermenegildo M. An Assertion Language for Constraint Logic Programs. Analysis and Visualization Tools for Constraint Programming, 2000:23-62
[27] Jaffar J, Maher M J. Constraint Logic Programming: A Survey. Journal of Logic Programming, 1994, 19-20:503-581
[28] Spoto F, Operational and Goal-independent Denotational Semantics for Prolog with Cut. The Journal of Logic Programming, 2000(42):1-46
[29] 陆钟万. 面向计算机科学的数理逻辑(第二版). 北京:科学出版社,2002
[30] Tarski A. A Lattice Theoretical Fixpoint Theorem and its Applications. Pacific Journal of Mathematics, 1955(5):285-310
[31] Spoto F, Levi G. Abstract Interpretation of Prolog Programs// A. M. Haeberer ed. Proceedings of the Seventh International Conference on Algebraic Methodology and Software Technology, AMAST'98. Lecture Notes in Computer Science, vol. 1548, Amazonia, Manaus, Brazil, Berlin Springer, January, 1999:455-470

(上接第 256 页)

数,但不能调用包含函数;2)不能使用"break"语句,不能嵌套"if"语句;3)循环变量不能跟循环体内的语句有关;4)循环体代码不能太长,否则需要的寄存器数会超过 C67x 提供的 32 个,应简化代码或将其分成多个小循环;5)循环体条件代码不能太复杂,否则需要的条件寄存器数会超过 C67x 提供的 5 个,应消除或精简条件代码。

6 仿真

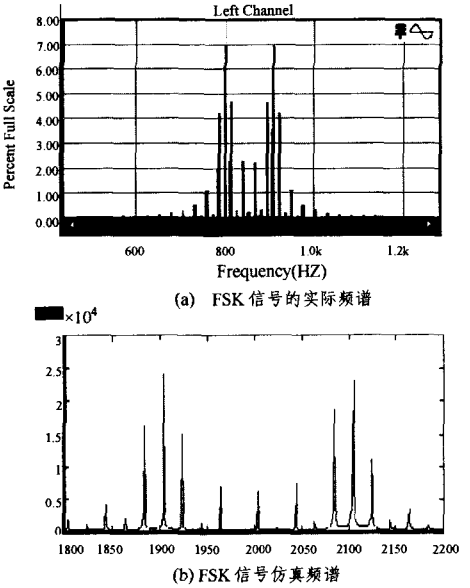

图 3 信号频谱图

为验证上述理论,采用著名的工具软件 MATLAB 进行 1024 点 FSK 信号的 FFT 算法仿真。MATLAB 有强大的 DSP 工具箱,其中有丰富的相关算法的函数。MATLAB 提供了 2 个内部函数,用于计算 DFT 和 IDFT,它们分别是:$fft(x)$, $fft(x,N)$。

(1)$fft(x)$计算 M 点的 DFT。M 是序列 x 的长度。

(2)$fft(x,N)$计算 N 点的 DFT。若 $M>N$,则将原序列截短为 N 点序列,再计算其 N 点的 DFT;若 $M<N$,则将原序列补零至 N 点,然后计算其 N 点的 DFT。

图 3(a)和图 3(b)分别是 FSK 信号实际的频谱结构图和仿真频谱结构图。通过仿真可知,采用此 FFT 算法和 DSP 技术,1024 点的 FFT 算法运算约需 3.75ms,且运算结果准确度高,误差率< 1%。

结束语 本文利用数字信号处理器实现基二按时间抽取快速傅里叶变换,并经证明,所实现的两种算法比采用的方法速度快且运算结果十分可靠,其用于一般的信号处理和工业控制都能满足精度和实时的要求,具有较高的学术价值和良好的市场应用前景。

参 考 文 献

[1] 吴美娟,岳俭. 数字处理器中的 FFT 实现[J]. 电子质量,2004(5)
[2] 胡广书. 数字信号处理[M]. 北京:清华大学出版社,2003:189-256
[3] 黄文梅,熊桂林,等. 信号分析与处理[M]. 长沙:国防科技大学出版社,1999:74-95
[4] 邹彦. DSP 原理及应用[M]. 北京:电子工业出版社,2005:279-316