使用SMP的超大点数FFT算法研究与实现

钱炳锋*，孙以泽

(东华大学机械工程学院 上海 松江区 201620)

【摘要】该文通过分析对称多处理器(SMP)并行处理系统的特点，提出了一种适用于SMP的超大点数FFT快速算法。该算法采取限定序列划分规则、改变铰链因子计算方法和优化数据分布及存储访问等手段，大大减少了对存储资源的依赖，并提升了FFT的执行性能。实测结果表明，该算法适用于SMP平台，有效地解决了单核处理器较难高效实现超大点数FFT的问题。

关 键 词 并行处理系统; 雷达信号处理; 对称多处理器; 存储优化; 超大点数FFT

中图分类号 TN911.7 **文献标志码** A doi:10.3969/j.issn.1001-0548.2019.01.005

Research and Implementation of VLFFT Algorithm Using SMP System

QIAN Bing-feng* and SUN Yi-ze

(School of Mechanical Engineering, Donghua University Songjiang Shanghai 201620)

Abstract Through analyzing the characteristics of symmetric multi-processor (SMP) parallel processing system, a very large fast fourier transform (FFT) algorithm is proposed. This algorithm significantly reduces the dependence on memory and improves FFT's performance by taking the limited rules of one-dimensional sequence split, changing the twiddle factor calculation method, and optimizing the data distribution and storage access. Experiment results show that the algorithm is suitable for SMP platform and can effectively solve the problem of very large FFT, which single-core processor harder to realize.

Key words parallel processing system; radar signal processing; SMP; storage optimizing; very large FFT

快速傅里叶变换(FFT)是雷达信号处理的关键技术，FFT的执行效率关系到系统的处理性能。近年来高分辨、大带宽相控阵雷达系统的快速发展，使得FFT计算点数大幅度的增加，带来了FFT计算面临处理器资源有限和执行效率降低这两大问题，直接影响了雷达系统的研制进度。

现有FFT算法大都基于单核处理器进行研究，很少有文献涉及适用于对称多处理器(symmetric multiprocessor, SMP)的超大点数FFT算法研究，但现有研究仍值得本文借鉴。文献[1]给出基于GPU分块的FFT算法，解决了图像容量较大引起的内存溢出问题，对于雷达系统中采用SMP计算FFT具有一定参考，但该算法很难移植到其他的多核处理器平台。文献[2]在DSP芯片上设计实现了一个基于矩阵转置操作的高能效FFT加速器，采用多种并行策略、混合旋转因子产生策略和“乒乓”结构多数据存储器来提升FFT加速性能和计算效能。文献[3]针对现代超标量结构处理器建模分析，最终实现了一种基于cache优化的高效FFT映射方法，该方法将FFT进行拆分，发挥了cache的作用进而提高了处理的性能，其cache优化策略和FFT拆分方法具有参考意义。文献[4]针对多核处理器实现了大点数FFT算法分解和并行优化，验证了多核DSP处理器的运算性能，但是该算法引入加权旋转因子，同时存在多次显性转置，降低了FFT计算实时性。目前芯片技术的发展已经不能适应摩尔定律和突破功率墙的限定，单核处理器性能已接近极限值，为了追求更高的处理性能，各芯片厂商采取在相同的面积上集成更多的处理器核。SMP汇集了一组处理器，它是应用十分广泛的并行技术，同时SMP芯片作为雷达信号处理芯片逐渐普及，因此，研究一种适合SMP的超大点数FFT算法，对于提升雷达系统的信号处理性能十分必要。

本文通过分析SMP并行处理系统架构特点，给出一种适用于SMP的超大点数FFT算法。该算法采取限定一维序列划分规则，最大限度降低复数乘加

收稿日期：2017 – 09 – 06；修回日期：2018 – 03 – 15
基金项目：上海市教委实验队伍建设计划(14SY08)
作者简介：钱炳锋(1982 –)，男，博士生，主要从事阵列天线方面的研究. E-mail: qbf314403@126.com

运算量，并改变乘铰链因子方法，减少了对存储资源的依赖，同时优化数据分布和存储访问来隐藏显性转置，最后将算法映射到TMS320C6678多核DSP处理平台。实测结果表明，该算法适用于SMP平台，能够解决单核处理器较难实现超大点数FFT的问题，并在FFT存储资源利用和执行性能上均有提升。

1　基于SMP的并行处理系统

为满足相控阵雷达系统的高分辨和大带宽的应用需求，信号处理系统需要更大的存储资源和更低的处理时延。因此，将并行计算机理论应用于雷达实时处理系统，通过加大并行计算规模和提升并行效率来解决现有问题[5-6]。

目前，多核处理器已成为市场主流，SMP是一种应用广泛的多处理器系统[7-8]。SMP的每一个处理器，都可以同等权限访问I/O、外设、内存和中断等系统资源，这样可以使任务或数据对称地分布于多个处理器核之上。由于核之间的距离更近，它们之间通信的带宽更大，时延更小，性能更佳。图1为基于SMP的分布式存储结构并行处理系统示意图。

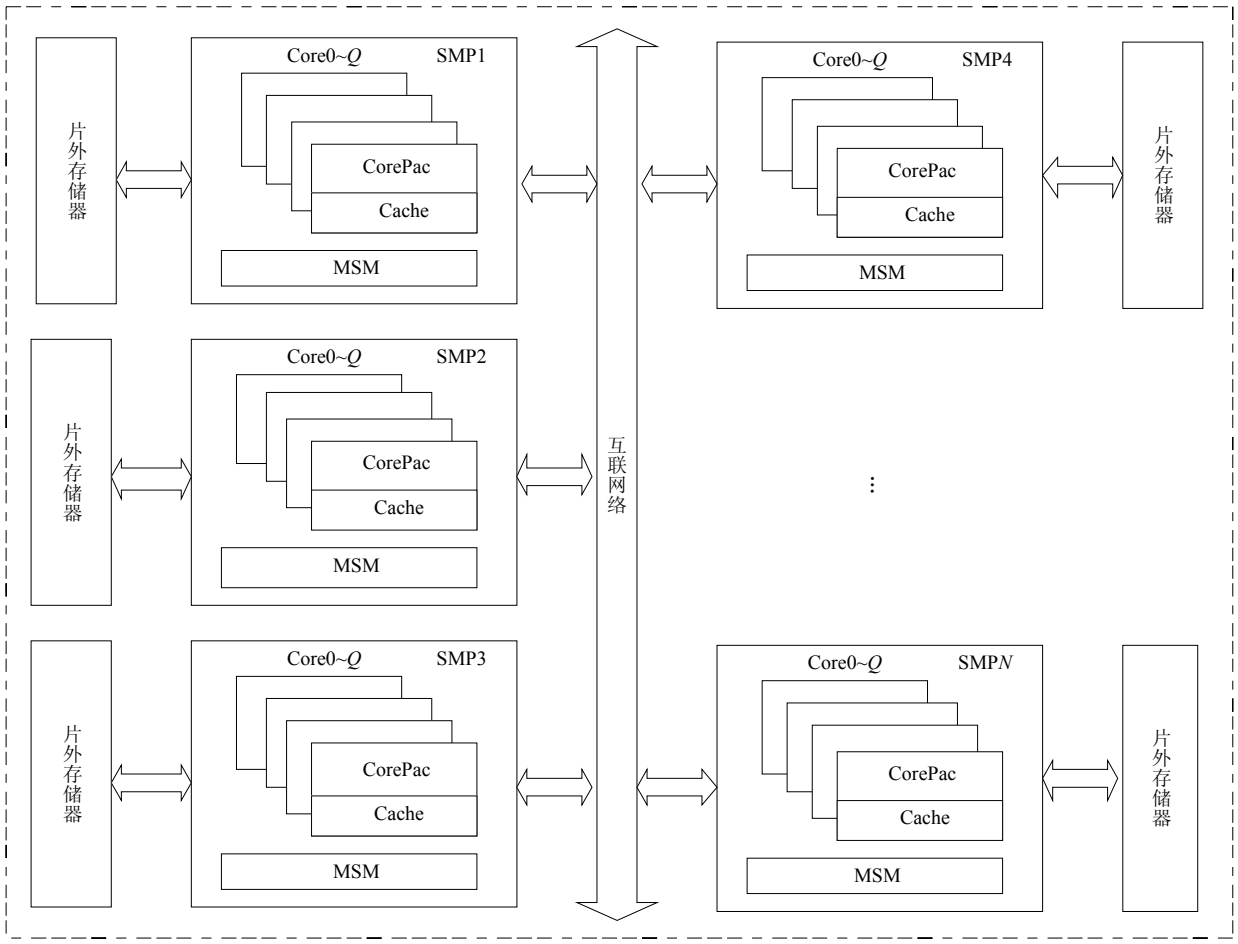

图1　基于SMP的分布式存储结构并行处理系统示意图

由图1可以看出，通过互联网络，每片SMP之间可以实现通信及数据交互功能，处理器核可以同等的权限访问每一片SMP，包括它的片外存储器和片内共享内存(MSM)，所有这些SMP组成了一个SMP系统。现有算法的工程实现大都基于单核处理器来完成，无法直接适用于SMP处理平台。例如，在空时自适应处理系统中，为了最大限度地提升FFT算法的效率，FFT的实现过程均要结合处理器的特性，针对特有平台来编写实现。针对SMP并行系统的特点，下面给出一种适用于SMP的改进型超大点数FFT算法。

2　改进型超大点数FFT算法

对于大点数FFT计算，文献[2-4]中将一维大点数序列FFT拆分成二维小点数来运算，该方法能够将FFT运算划分成更小的粒度来进行处理。其实现的主要流程为：1) 将一维序列拆分成二维矩阵并转置；2) 行方向FFT处理；3) 乘以铰链因子；4) 对3) 的结果转置存储；5) 列方向FFT处理；6) 将处理结果转置存储，得到序列的FFT结果。但是，针对

SMP系统，该算法在实现过程中存在3个问题：1) 一维序列行列如何拆分才能得到最优化的运算性能；2) 如何均衡乘以铰链因子引入额外的存储空间和增加乘法的运算次数；3) 3次显性转置对内存空间需求较大，且点数较大时因无法在片内存储空间计算导致性能大幅降低。针对上述3个问题，本节分别给出具体的优化方法。

2.1 序列划分规则优化

设FFT一维序列长度为L，划分成为二维序列$L=MN$，M为列数，N为行数，细粒度行列FFT运算采取基2-FFT算法，依据其计算量公式(1)，可以得出实现FFT运算所需运算量为：

$$\begin{cases} C_{\text{mul}} = N\dfrac{M}{2}\log_2 M + M\dfrac{N}{2}\log_2 N + C_f \\ C_{\text{sum}} = NM\log_2 M + MN\log_2 N \end{cases} \tag{1}$$

式中，C_{mul}为乘法次数；C_{sum}为加法次数；C_f为额外引入的铰链因子复数乘法次数，序列长度为L。

由柯西不等式可得：

$$\begin{cases} C_{\text{mul}} \geqslant \dfrac{L^2}{4}\log_2 2\sqrt{L} + C_f \\ C_{\text{sum}} \geqslant L^2\log_2 2\sqrt{L} \end{cases} \tag{2}$$

当$M=N=\sqrt{L}$时，式(2)取最小值：

$$\begin{cases} C_{\text{mul}} = \dfrac{L^2}{4}\log_2 2\sqrt{L} + C_f \\ C_{\text{sum}} = L^2\log_2 2\sqrt{L} \end{cases} \tag{3}$$

由以上推导分析可得，当一维序列按照行和列相等的原则来划分时，理论上可以保证FFT的复数乘法和加法的运算量达到最小值，使其拆分性能达到最优。对FFT计算来说，序列的长度为$N=2^n$，实际操作中M，N的值取：

$$\begin{cases} M = N = 2^{\frac{n}{2}} & n\text{为偶数} \\ M = 2^{\frac{n-1}{2}},\ N = 2^{\frac{n+1}{2}} & n\text{为奇数} \end{cases} \tag{4}$$

2.2 铰链因子计算优化

由前面的处理步骤可知，在步骤3)时对行方向FFT的处理结果乘以加权旋转因子，其值如下：

$$Z(n_0,k_0) = \mathrm{e}^{\frac{-\mathrm{j}2\pi n_0 k_0}{L}} \tag{5}$$

式中，$n_0=0,1,\cdots,M-1;\ k_0=0,1,\cdots,N-1$。

因为处理器的资源有限，需要设法通过有限的资源最大限度的节省内存资源，这对实现超大点数FFT十分重要。因此将式(5)进行展开分解形成一个$M\times N$的二维矩阵，矩阵每一行的值可以由第1行计算得到。由此，铰链因子可以只存储第1行数据，可由第1行的铰链因子重复计算得出其他行的铰链因子。但是，这个处理方法引入了额外的复数乘法运算，需要评估该运算对超大点数FFT执行效率的影响，下面对该复数乘法运算对整个计算量的影响作进一步分析。

假设只考虑采用复数乘法来衡量FFT的运算时间，按照划分原则，新引入的额外复数乘法占总复数乘法的比重为：

$$D = \frac{C_{\text{amul}}}{C_{\text{mul}} + C_{\text{amul}}} = \frac{N(M-1)}{\dfrac{L^2}{4}\log_2 2\sqrt{L} + C_f + N(M-1)} \tag{6}$$

式中，C_{amul}为额外增加的复数乘法数量，取$M=N=\sqrt{L}$，式(6)推导为：

$$D < \frac{1}{\dfrac{L^2}{4(L-\sqrt{L})}\log_2 2\sqrt{L} + 2} \tag{7}$$

从式(7)和图2可以看出，新引入的额外复数乘法在FFT总复数乘法所占比重随着点数增加而减小，当FFT点数大于256 000时，所占复数乘法比重系数D趋于0。

因此，对于超大点数FFT而言，通过优化铰链因子计算方法，使得新引入的额外复数乘法运算量可以忽略不计，能够节省处理器片内存储资源。

2.3 数据分布和存储访问优化

由于超大点数FFT点数较大，且3次显性转置需要至少一倍于序列长度的缓冲空间，无法将FFT的原始数据、中间缓冲转置数据和FFT处理结果数据全部放在处理器的片内存储空间，需要选择片外存储器(DDR)作为大数据的存储空间，但由处理器直接访问DDR将降低执行效率。下面对均衡FFT数据分布和存储访问性能之间的关系进行说明。

根据SMP并行处理系统的特点，单片SMP作为一个基本处理单元，由多个处理器核共同完成一个超大序列的FFT运算。如果直接将原始数据、中间转换数据、结果数据分布在片外存储空间进行访问计算，会引起处理性能的下降；又因为FFT被拆分成二维序列来处理，行或列FFT的处理点数变得很小，可以读到片内存储空间进行计算，所以可以采取将原始序列数据存储在DDR，然后将数据搬移到片内存储空间进行行或列FFT计算，最终再将结果写回DDR。设SMP的核处理器个数为Q，序列长度$L=M\times N$，则单片SMP共同处理一个超大序列的FFT流程如图3所示。

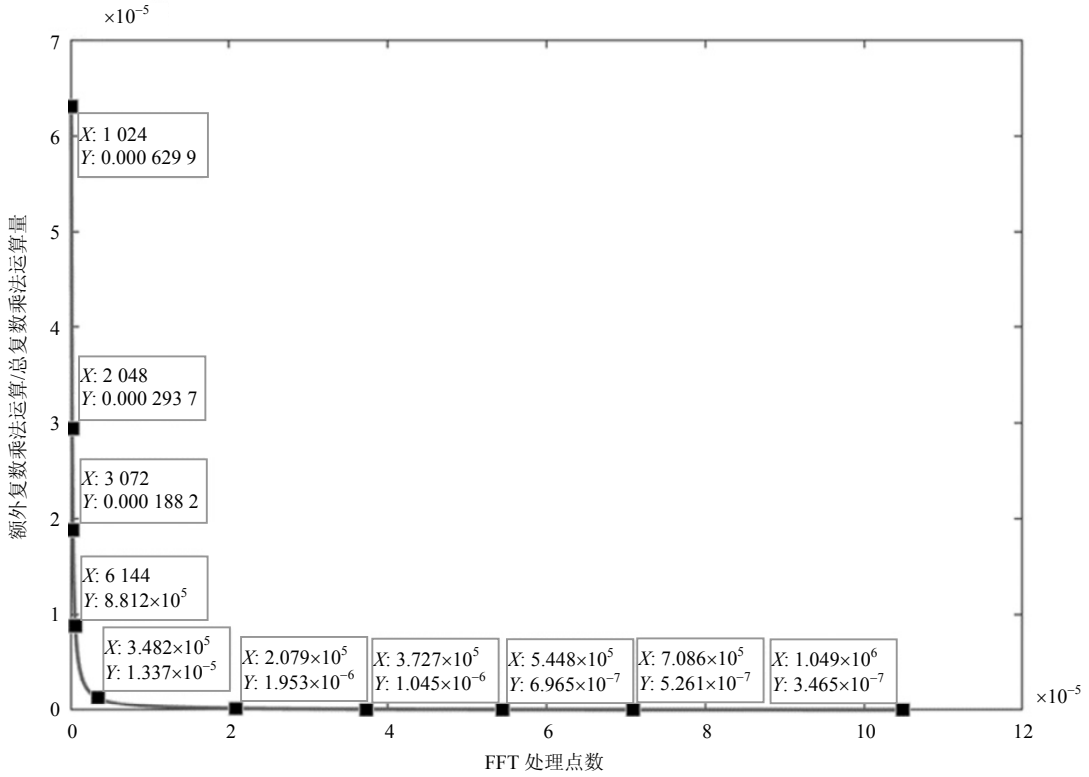

图2　随FFT处理点数增加D的曲线图

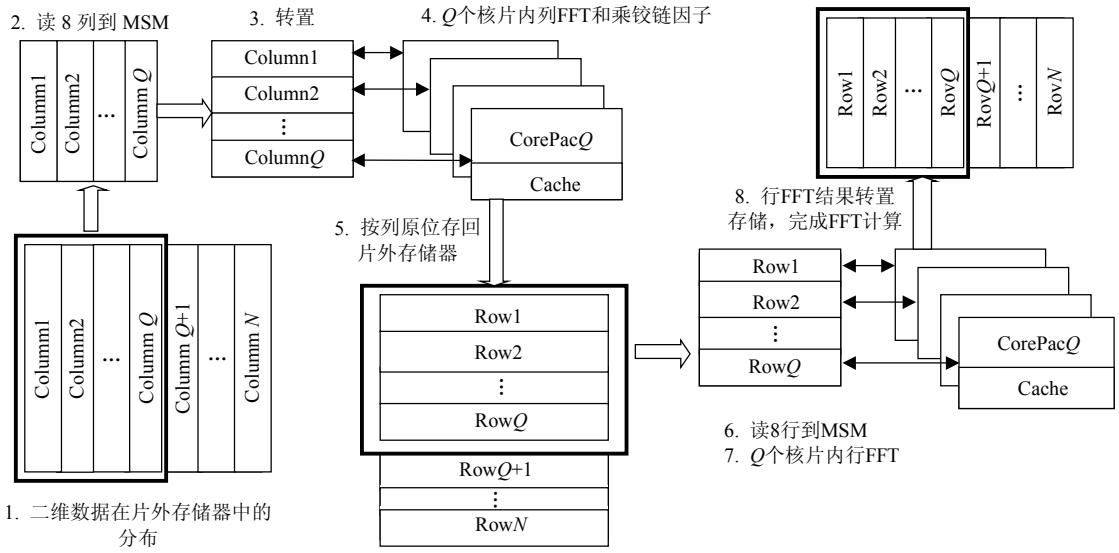

图3　单片SMP并行FFT处理流程及中间数据分布

采取数据分布和矩阵访问优化，将占用存储空间较大的原始数据放在片外存储空间DDR，按照Q为访问粒度来读取相应的行或列数据到MSM，由SMP处理器核1、2、…、Q分别完成各自对应行和列的FFT计算。行列FFT运算可以划分为4个任务：读数据、转置、FFT和写数据，对4个任务流水进行合理安排，行列FFT的循环流水时序如图4、图5所示。

由此可知，数据读写及矩阵转置均被隐藏在处理流程中，减少额外的DDR存储空间进行显性转置，节省了存储空间，提升了矩阵转置效率。表1给出了该处理方法对存储资源需求。

对超大点数FFT运算的数据分布和矩阵存储访问进行优化，以SMP处理器核Q为数据访问粒度，采用片内临时“乒乓”缓冲空间，使得FFT计算存取速度比直接访问DDR快，同时对任务进行并行流水处理，将数据读写和矩阵转置隐藏在FFT处理流

程中，进而节省系统存储资源，降低了超大点数FFT处理延时。因此，通过优化数据分布和矩阵访问，在有限的处理器资源下，为高效地实现更大点数的FFT提供了可行方法。

基于以上分析，改进后的超大点数FFT算法实现流程如图6所示。

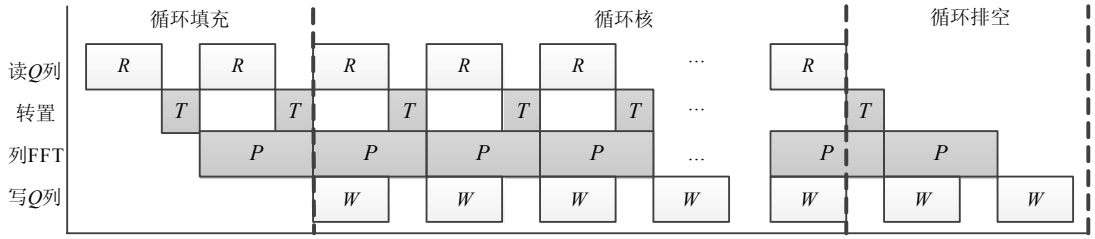

图4 列FFT循环流水时序

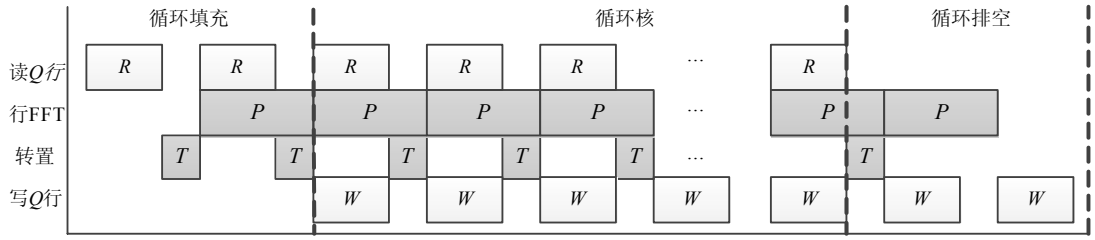

图5 行FFT循环流水时序

表1 本文方法实现FFT对资源的需求

存储空间	存储资源(复数点)	存储类型
输入/输出数据空间	L	DDR
旋转因子空间	N	片内SRAM
铰链因子空间	N	片内SRAM
“乒乓”缓存空间	$4QN$	MSM
总需求存储资源	$L+(4Q+2)N$	——

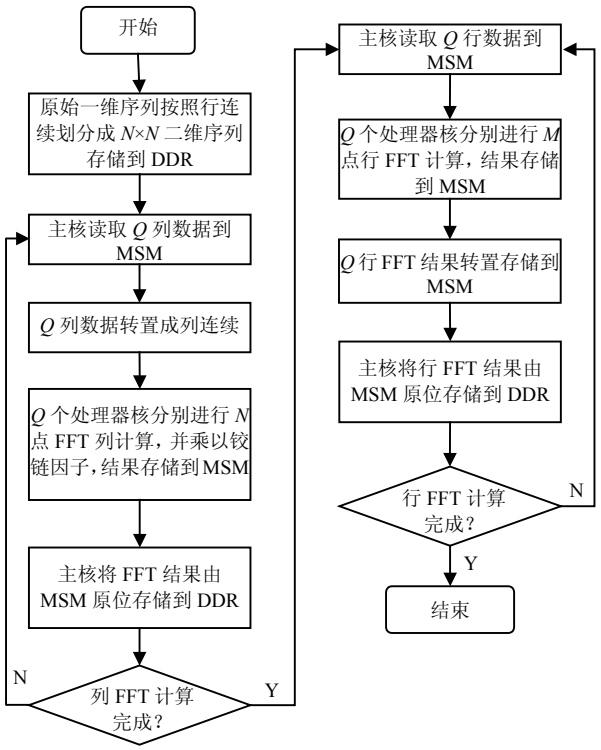

图6 超大点数FFT流程图

3 实验验证与性能对比

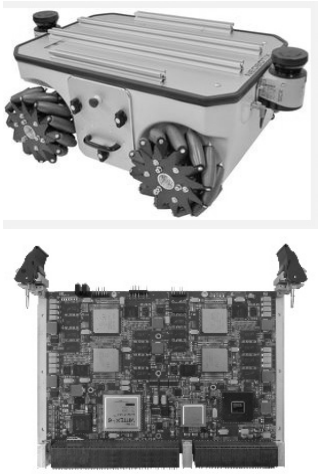

图7 系统构建实物图

实验依托AS-R机器人项目，实验结果在多DSP综合雷达信号板卡上测试得到，系统构建实物如图7所示。该综合处理系统由一片Virtex7 FPGA和4片TMS320C6678的组成，每片DSP最大支持外挂8 GB的DDR3片外存储器，DSP之间通过高速串行总线Rapid IO互联形成SMP系统架构；TMS320C6678是TI公司新一代浮点DSP，由8个同构处理器核共同组成SMP系统结构；每个处理器核最高支持1.4 GHz的内核时钟，内部集成512 KB的L2存储空间，8个核共享4 MB片内存储空间，并且拥有强大的处理能力和系统互联能力[10]。实验验证条件和测试方法：1) DSP内核工作主频为1 GHz；2) 通过读取超大点

数FFT函数执行前后时钟计数周期之差得到执行时间。基于SMP的超大点数FFT实测结果如表2所示。

由以上对比结果可知，相对于已有处理方法，改进型超大点数算法能够适用于SMP处理器架构，且随着FFT点数的增加，其对存储资源的节省和对性能的提升也越明显。

表2　浮点FFT实测结果对比　　ms

点数	本文	文献[4]
8 192	0.062 330	——
16 384	0.129 746	0.131
32 768	0.189 187	0.198
65 536	0.313 758	0.315
131 072	0.629 758	0.641
262 144	1.186 272	1.386
524 288	2.836 328	3.103
1 048 576	4.664 122	6.507

表3　浮点FFT资源需求对比　　kB

点数	本文	文献[4]
8 192	98	162
16 384	162	290
32 768	324	580
65 536	580	1 092
131 072	1 160	2 184
262 144	2 184	4 232
524 288	4 368	8 464
1 048 576	8 464	16 656

4　结 束 语

本文针对现有FFT算法不适用SMP并行处理平台的问题进行分析，提出了改进型超大点数FFT算法。通过优化行列划分规则来最大程度的减少了复数乘加运算量，并改变了铰链因子计算方法，优化了数据分布和存储访问，降低了存储资源开销，给出了改进后的算法实现流程。实验对比结果表明，本映射算法适用于SMP系统，能够节省约1/2的存储资源，提升了大点数FFT处理速度，可用于改善高分辨率大带宽雷达信号处理系统的实时性。

参 考 文 献:

[1] 杨雪, 李学友, 李家国, 等. 基于GPU和分块技术的巨幅影像快速傅里叶变换算法研究[J]. 光谱学与光谱分析, 2014 (2): 498-504.
YANG Xue, LI Xue-you, LI Jia-guo, et al. Research on fast fourier transforms algorithm of huge remote sensing image technology with GPU and partitioning technology [J]. Spectroscopy and Spectral Analysis, 2014 (2): 498-504.

[2] 雷元武, 陈小文, 彭元喜. DSP芯片中的高能效FFT加速器[J]. 计算机研究与发展, 2016, 7: 1438-1446.
LEI Yuan-wu, CHEN Xiao-wen, PENG Yuan-xi. A high energy efficiency FFT accelerator on DSP chip[J]. Journal of Computer Research and Development, 2016, 7: 1438-1446.

[3] 高立宁, 朱亮, 刘腾飞, 等. 基于超标量处理器的高效FFT映射方法[J]. 北京理工大学学报, 2016, 9: 940-946.
GAO Li-ning, ZHU Liang, LIU Teng-fei, et al. An efficient FFT-mapping method based on superscalar processor[J]. Transactions of Beijing Institute of Technology, 2016, 9: 940-946.

[4] 袁琪, 杨康, 周建江, 等. 大点数FFT算法C6678多核DSP的并行实现[J]. 电子测量技术, 2015, 38(2): 74-80.
YUAN Qi, YANG Kang, ZHOU Jian-jiang, et al. Implementation of large points FFT based on C6678 multi-core DSP[J]. Electronic Measurement Technology, 2015, 38(2): 74-80.

[5] VINOGRADOVV I. Advanced high-performance computer system architectures[J]. Nuclear Instruments & Methods in Physics Research, 2007, 571(1-2): 429-432.

[6] NADER B, HAMID S. High-performance computing system architectures: design and performance [Editorial][J]. Iet Computers & Digital Techniques, 2012, 6(5): 257-258.

[7] ZHU Yong-zhi, ZHANG Dan-dan, CAO Bao-xiang, et al. Research of parallel programming techniques of hierarchical model based on SMP clusters[J]. Acta Electronica Sinica, 2012, 40(11): 2206-2210.

[8] TAM D, AZIMI R, STUMM M. Thread clustering: Sharing-aware scheduling on SMP-CMP-SMT multiprocessors[J]. Acm Sigops Operating Systems Review, 2007, 41(3): 47-58.

[9] 王世一. 数字信号处理[M]. 北京: 北京理工大学出版社, 1997: 123-131.
WANG Shi-yi. Digital signal processing[M]. Beijing: Beijing Institute of Technology Press, 1997: 123-131.

[10] Texas Instruments. TMS320C6678 multicore fixed and floating-point digital signal processor[EBOL]. [2017-10-11]. http://www.ti.com.cn/lsds/ti_zh/processors/dsp/c6000_dsp/c66x/overview.page, 2016.

编　辑　刘飞阳