空时自适应处理的通用平台设计与实现

邵银波[①②] 李 强[①] 王永良[①] 陈 辉[①] 肖奚安[②]
[①](空军雷达学院雷达兵器运用工程全军重点实验室 武汉 430019)
[②](解放军理工大学通信工程学院 南京 210016)

摘 要 该文利用多个高性能数字信号处理器，结合 FPGA 和通用处理器，实现了一个空时自适应处理(STAP)的通用实时平台系统。借鉴 Valiant(1990)提出的 BSP 模型，采用多重流水线，提出一个空时自适应处理(STAP)计算模型。该模型可以弥补 STAP 算法和实际并行系统的差距，为开发提供了统一框架；同时，方便了对算法的性能评估。在基于该模型的具体开发过程中，选择可扩展簇式多处理机结构作为系统硬件架构，采用数据块静态分配方案进行算法的分解与映射，并采取一系列通信和程序优化的方法。结果表明，系统能满足实时要求，可扩展性好，方便类似系统的系列开发。

关键词 数字信号处理，并行处理，空时自适应处理，通用实时平台
中图分类号: TN911.72 **文献标识码**：A **文章编号**：1009-5896(2006)02-0317-05

A Universal Real-Time Platform for Space-Time Adaptive Processing

Shao Yin-bo[①②] Li Qiang[①] Wang Yong-liang[①] Chen Hui[①] Xiao Xi-an[②]
[①](*Key Research Lab., Wuhan Radar Academy, Wuhan* 430019, *China*)
[②](*Nanjing Institute of Communications Engineering, Nanjing* 210016, *China*)

Abstract A universal real-time platform for a Space-Time Adaptive Processing (STAP) is developed. The platform is composed of multi-DSPs，FPGA and a general-purpose processor. Refined from the Bulk Synchronous Parallel model (BSP) by Valiant(1990), an STAP computation model is brought forward. The model provides a bridge between STAP algorithms and real parallel systems. Moreover, it can be applied to performance evaluation. During the course of development, scalable cluster-organized multi-processors structure is adopted as hardware architecture. And data-block static allocating is taken as the mapping scheme. Afterwards, some optimization methods about communication and programming are introduced. This system can meet the real-time requirement, its scalability is good, and it facilitates the development of similar systems.

Key words DSP, Parallel processing, Space-time adaptive processing, Universal real-time platform

1 引言

空时二维自适应处理(STAP)技术是新一代星载、机载雷达的关键技术，能有效抑制强度大且分布广的地(海)杂波和多种干扰，从而大幅度提高空中与地(海)目标检测性能。然而，由于STAP运算量巨大，数据交互复杂，(每秒一般有 10^{10} 到 10^{14}次的浮点操作，需处理 0.1G到几百G的数据量)，因而导致实时性能较差，很难应用到实际系统上。随着现代并行处理和大规模集成电路(VLSI)设计技术的发展，使得STAP 有可能应用于实际系统。

当前，国外主要有两大主流开发技术： 基于UNIX/ NT的高性能商用服务器[1-3]和嵌入式多处理机[4-6]。而国内西安电子科技大学曾经用一百多片数字信号处理器(DSP)设计了STAP处理机。一般说来，基于UNIX/NT的高性能商用服务器更容易编程而且更容易移植。但星载、机载雷达的实时性、嵌入特性的特殊要求迫使人们采用嵌入式多处理器。

解决二者分歧进退两难，没有一个简单的答案，较有前景的方案是集成协处理器到商用机架，而同时保留他们的软件优势。我们采用该方案，同时考虑到国外对我们的限制以及系统限制，因此，利用多个高性能 DSP 结合 FPGA 和通用处理器，研制完成了 STA P 通用平台，以得到最优

2004-07-22 收到, 2005-04-18 改回
全国高等学校优秀青年教师教学科研奖励计划(TRAPOYT)和国家自然科学基金(60272086)资助课题

的 STAP 系统核，为 STAP 成功应用于实践奠定基础，同时，为下一系列开发提供通用的框架平台。

本文研究的重点是综合提高以下指标：(1)实时性能(performance): 当前主要是使完成时间最小化，吞吐量最大化。(2) 可扩展性(scalability)[7]: 主要指应用的可扩展性、资源的可扩展性、技术的可扩展性，初期目标是方便各种算法的选择、测试。(3)生产率(productivity): 目前，该设备原型系统已调试完毕，各项指标达到初步设计要求。

2 STAP 基本原理

空时二维自适应处理技术基本原理见图 1 所示[8]，其中N表示天线子阵数或空间通道数，K为时间脉冲数。图中$\{w_{nk}\}$，$n=[1,N], k=[1,K]$为空时二维权系数。

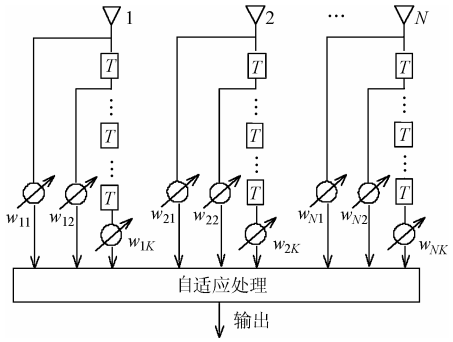

图 1 最优处理器原理图

图 1 的最优处理器就是对空域N个单元和时域K个单元全部作自适应控制，从而可以达到处理器的最佳性能。但实际机载雷达一般采用平面阵，其单元数往往为几十、几百甚至上千，如果在阵元级作自适应处理，则控制数目NK将很大，作全自适应所需的运算量相当庞大，而且杂波数据协方差矩阵也难以构成。因此，为了实现STAP技术必须考虑采用降维处理。通常降维处理方法主要有[9]：时空二维Capon法(也用MTI-1DT-SAP 表示，简称MTI-1DT)、多通道联合自适应处理方法(MDT-SAP，简称MDT)以及和差通道(ΣΔ-STAP)法。当然，还有其它类型处理算法，如特征相消器(EC)、交叉谱(CSM)和多级维纳滤波(MWF)。在具体设计与开发过程中，我们可以综合考虑各种具体要求，通过理论评估、仿真以及在线测试进行相应算法选择。

3 STAP 并行计算模型

为了向开发人员提供统一框架模型，统一各种开发实现活动，在综合考虑了 STAP 各种算法的特点后，我们首先建立 STAP 并行计算模型，以弥补 STAP 算法和实际并行系统的差距。

关于各种STAP算法计算性能的比较，我们已经具体分析过[9]，然而，当时并没有考虑到实际系统的种种限制，如数据交互、同步等对性能的重要影响。因此，该模型的建立同时也为算法研究人员进行性能评估提供了必要的工具。

根据STAP算法特点，借鉴Valiant提出的BSP模型[10]，我们提出的STAP系统的并行计算模型如图 2。该模型为多重流水线结构，系统每隔一段时间就输入一批数据进行处理。整个处理过程是由并行初始化过程以及一系列的串行的超级步(Superstep)组成。在下一个超级步开始之前，当前超级步的所有操作必须结束。每一个超级步由计算阶段和紧随其后的数据交互阶段构成。其中，计算阶段指各处理器并行地对预取到当地的数据集进行运算处理；而交互阶段则是指各种各样的通信操作，如点对点通信、全局数据交互、路障同步等等。交互阶段所交互的数据，主要来自本批数据处理前一阶段的结果，也有可能来自上批数据处理的前一阶段结果。

模型有以下特点：(1)充分反映了 STAP 处理流程的特点。(2)考虑到算法有大量的数据交换，且往往是系统的性能瓶颈，因此我们在设计初期时就进行数据交互的设计。(3)为减少不必要的同步和通信，合并了一些算法阶段，形成相应的超级步。(4)流水线内部数据交互少，故在实现时，应考虑尽量将单条流水线在一块子板内运行，多条流水线对应多块子板。

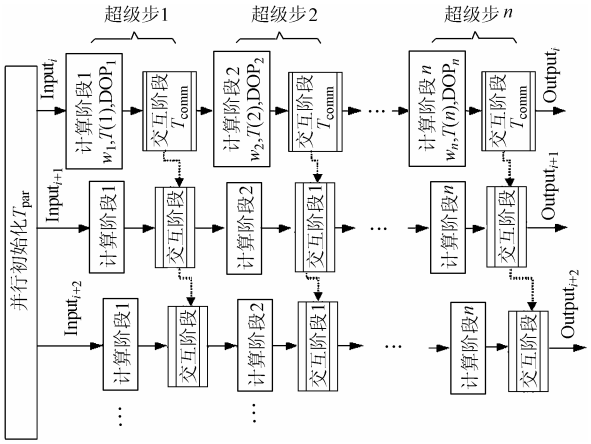

图 2 STAP 并行计算模型

下面根据该计算模型分析系统的实时性能。

在上述模型中，不妨设T_{par}为并行初始化时间，T_{comm}为

数据交互时间，T_{comp} 为计算时间，不难得到，系统延时 T_{latency} 为

$$T_{\text{latency}} = T_{\text{par}} + \sum_{1\le j\le n} T_{\text{comm}}(j) + \sum_{1\le j\le n} T_{\text{comp}}(j) \tag{1}$$

式中 n 为系统的超级步总数。

实际估算中，在并行初始化阶段，当去除数据分配所消耗时间 $T_{\text{comm}}(0)$ 后往往远低于其他时间，因此，式(1)可以简化为

$$T_{\text{latency}} = \sum_{0\le j\le n} T_{\text{comm}}(j) + \sum_{1\le j\le n} T_{\text{comp}}(j) \tag{2}$$

式(1)，(2)中，$T_{\text{comm}}(j)$ 可以由下式进一步估算出

$$T_{\text{comm}}(j) = t_0(p) + m_j / B(P) \tag{3}$$

式中 $t_0(p)$ 为数据传输的启动时延，$B(p)$ 为系统的数据传输带宽，他们都仅仅与系统的硬件规模，尤其是处理器数目 p 多少有关。m_j 为各处理器发送或接收数据块的最大长度，而 $T_{\text{comp}}(j)$ 则为

$$T_{\text{comp}}(j) = T(j)/\text{Min}(\text{Dop}_j, p_j) \tag{4}$$

其中 $T(j)$ 代表用单个处理器处理其工作量所需要的时间，Dop_j 代表算法固有的可并行度，p_j 代表实际处理使用的处理器个数。

结合式(1)－式(4)，系统的延时性能可以由下式估算出:

$$T_{\text{latency}} = (n+1)\text{to}(p) + \sum_{0\le j\le n} m_j / B(P) + \sum_{1\le j\le n} T(j)/\text{Min}(\text{Dop}_j p_j) \tag{5}$$

但是，仍有一种情况，式(4)估算不是很精确。举例说明，当 $\text{Dop}_j = 5,\ p_j = 4$ 时，若用式(4)估算，有一个前提假设：系统 4 个处理器全并行工作。实际上，系统在完成前 4 部分任务时，是可以达到这个速度的，而完成 4 个任务以后，已经只剩下 1 个任务了，此时，系统实际速度是 1 个处理器工作。显然，此时的计算性能估算误差较大。

观察发现，该计算阶段其实可以看成两个阶段，前一阶段是 p_j 个处理器全并行，而后一阶段是前一阶段余下任务数，即 $(\text{Dop}_j \bmod p_j)$ 个处理器的全并行，因此，式(4)可以表示为

$$T_{\text{comp}}(j) = \frac{T(j)}{p_j} \cdot \frac{\text{ceil}(\text{Dop}_j / p_j) - 1}{\text{ceil}(\text{Dop}_j / p_j)} + \frac{T(j)}{(\text{Dop}_j \bmod p_j)} \cdot \frac{1}{\text{ceil}(\text{Dop}_j / p_j)} \tag{6}$$

式(6)中，ceil 表示向上取整函数，mod 代表求余函数。

显然，当 $\text{ceil}(\text{Dop}_j / p_j)$ 不够大时，应该用式(6)估算，否则，用式(4)即可。

必须指出的是，上述估算并没有考虑负载不平衡，因为，STAP 算法非常规则且具有很高的可并行度，实际应用中不难达到相对负载平衡。

通过上述公式可以方便地估算出各种算法的延时性能。关于吞吐量的性能分析，由于它需要考虑具体的系统设计、硬件配置和部署方案，故在此我们不做统一讨论。

4　系统设计与实现

为确保系统综合性能的提高，在设计与实现过程中，我们参考 STAP 计算模型，紧密结合算法以及底层硬件特点，充分协调各种要求，有效折衷，采用多种方法。

4.1　系统硬件架构选择可扩展簇式多处理机结构

系统主要由主机 HOST 和多片 DSP 扩展板组成。而每片 DSP 扩展板选用多片 DSP(暂时为 4 片 TS101 结合 FPGA 组成。系统采用 cPCI 总线标准，该标准既具有 PCI 总线的高性能，又具有 VME 欧规卡结构的高可靠性。

多片 DSP 间架构我们采用一种可扩展簇式多处理机结构：子板间通过总线互连；子板内 DSP 之间通过 TS101 的 link 口全互连结构，同时又有高速共享存储器总线相连共享 SDRAM，每个 DSP 各有一 link 口连到 FPGA。该结构兼有数据流式或者波前阵列多处理机结构的高吞吐量，而同时又有共享存储总线结构的高灵活性，非常适合作为通用处理平台。

4.2　算法的分解与映射采取数据块静态分配方案

算法的分解首先要识别足够的并行性，考虑到 STAP 算法以及系统硬件特点，不太适合细粒度并行，我们将注意力集中在粗粒度并行分析上，即主要从问题以及算法本身来考虑。

考察 STAP 计算模型，我们发现算法是对大量的数据分别施行一系列有数据交互的计算处理，然后输出结果。因此，在算法的各个阶段，我们将数据集进行分解，并分配给不同的处理器，被分配了数据的处理器进程然后负责和那些数据有关的计算，即所谓拥有者计算。

关于分解后如何映射到处理器上，有两种方式：静态分配和动态分配。动态分配的负载平衡特性较好但实现复杂，需要专门的调度算法，这无疑进一步增加内存较小的 DSP 的负担，而且，执行调度非 DSP 专长。本系统算法结构清晰简单，即使采用静态分配方式分配，同样具有很好的负载平衡特性。具体操作过程是：(1)处理器结点分为预处理节点和 STAP 处理节点。预处理节点可以由 FPGA 或 DSP 构成。(2)针对 STAP 处理节点，将数据进行划分分配到各 DSP。算法分解和映射的结果见图 3。分配过程中，我们将相邻的行分配给同一个进程(块分配)，充分利用时间和空间局部性，从而减少了通信。

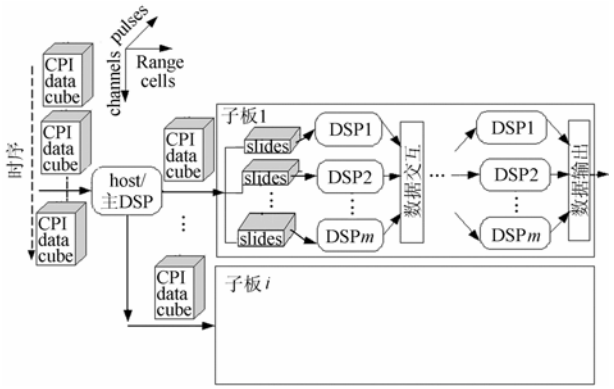

图 3 STAP 的数据分解与映射

4.3 通信优化

并行程序的主要执行开销是由处理节点间的通信和同步所引起的。系统编程模型、处理节点间网络结构、数据处理和路由选择以及相关协议都对性能有重大影响，这使得通信优化需要综合考虑各个方面。

幸运的是，STAP 算法的规则性使得分解和映射到具体多处理器系统后，可以得到很好的负载平衡特性；也正是由于这种规则性，使得访问和通信模式可以预测，因此，可以有很大的优化空间从而达到实时要求，而且，系统可以采取离线优化方式。具体实施时采取查表法，即事先将数据传输路径最优化，将结果存储于路径表中，程序按照查表所得的结果运行。

关于多个 DSP 同时访问某一存储器。大多推荐采用代理方式，其基本思想是，同一存储器的诸请求推选一个代表，代表与存储器打交道，然后利用数据分布算法迅速分布代表所获的结果。然而，本系统中，各 DSP 内存有限且必须装载程序，若再让其承担数据转发任务，很难胜任，且所取数据往往各不相同、节点数有限，故数据分布算法并无优势。因此，我们充分利用了硬件提供的信号量 semaphore，采取令牌环网的控制策略，可以达到让多片 DSP 分别处理部分数据最终完成全部数据的处理。

4.4 程序优化

关于程序的优化，资料比较多，我们主要把握了以下原则：面向算法，面向编译器，面向系统结构。

面向算法原则。算法中涉及到大量的各种各样的矩阵运算，我们针对 STAP 算法的特殊特点，对其进行了优化。如将大矩阵分解成多个小矩阵进行运算，从而使运算复杂度由 $O(n^2)$ 降到 $O(n\log n)$ 。又如系统提供了矩阵相乘的函数，但我们大量用到 $A\times A^{\mathrm{H}}$，根据其结果矩阵的对称性，只需算上三角或下三角，从而节省了大约一半的计算时间。

面向编译器原则。程序必须具有很好的可读性，不仅仅限于方便程序员理解，在实时系统开发过程中，方便编译器理解同样重要，甚至对系统的性能有着较大的影响。如直接利用系统支持函数、数据结构，避免循环内判断、避免循环依赖、不要人为展开循环等等。

面向系统结构原则。如充分利用 DSP 的哈佛结构，调整数据分布到代码存储器和数据存储器，便于数据并行读取。

4.5 一个例子

系统输入采用三维数据，帧脉冲数 40，距离门数 500，通道数 $N=12$。算法采用时空二维 Capon 法。

初步测试表明：(1)系统基本可满足实时要求。(2)在一定规模下，可扩充性非常好。再加一块子板，即使不进行系统优化，吞吐量性能加倍(如表 1)。(3)DMA 传输比直接读写更快，但仍占时间延时的较大比重，成为一关键性能瓶颈。但系统仍有较大的优化空间，如 link 口的带宽有待进一步利用。必须指出的是，本文计算模型完全可以用来预测各种算法的实时性能，而表 1 理论值与实测有一定差距，是因为用于预测的数据传输性能函数暂时采用的是点对点，而实际运行时却往往采用多播形式，但这并不妨碍算法的选择以及系统的设计。

表 1 算法实现性能(数据传输采用 DMA 方式)

子板数	延时(ms)		吞吐率(帧/s)	
	实测值	理论值	实测值	理论值
1	23	27	43	37
2	23	27	87	74

5 结束语

本文建立了一种 STAP 计算模型，既方便对算法的实时性能进行评估，同时又为系统开发提供了统一框架模型；系统硬件架构选择为可扩展簇式多处理机结构；算法的分解与映射采取数据块静态分配方案；运用了一系列通信和程序优化的方法。目前，该设备原型系统已调试完毕，各项指标达到初步设计要求。

参 考 文 献

[1] Hwang Kai, Wang Choming, Wang Cho-li, Xu Zhiwei. Resource scaling effects on MPP performance: The STAP benchmark implications [J]. *IEEE Trans. on Parallel and Distributed*

Systems, 1999, 10(5): 509－527.

[2] Choudhary A, Liao Wei-keng, Weiner D, Varshney P, Linderman R, Linderman M, Brown R. Design, implementation and evaluation of parallel pipelined STAP on parallel computers [J]. *IEEE Trans. on Aerospace and Electronic Systems*, 2000, 36(4): 528－548.

[3] Liao Wei-keng, Choudhary A, Weiner D, Varshney P. Multi-threaded design and implementation of parallel pipelined STAP on parallel computers with SMP nodes [A]. Proceedings of the 13th International Parallel Processing Symposium[EB/OL]. http://citeseer.ist.psu.edu /liao99multithreaded.html, 2003: 10－12.

[4] Hamlet E. Embedded and Real-Time Application of High-performance Scalable Computing [R]. Syracuse NY: Lockheed Martin, 1999: 1－30.

[5] Liu Wenheng , Prasanna K. Design of application software for embedded signal processing[EB/OL]. http://citeseer.ist.psu.edu /270331.html, 2003: 10－12.

[6] McMahon S, Teitelbaum K. Space-time adaptive processing on the mesh synchronous processor [A]. IEEE Proceedings of IPPS ’96 [C]. Hawaii , 1996: 734－740.

[7] Hwang Kai, Xu Zhiwei. Scalable parallel computers for real-time signal processing [J]. *IEEE Signal Processing Magzine*, 1996, 13(4): 50－66.

[8] 王永良，彭应宁. 空时自适应信号处理 [M]. 北京：清华大学出版社. 2000, 9: 47－48.

[9] 陈建文,王永良. 空时自适应处理典型方法综合性能评估 [J]. 系统工程与电子技术, 2000, (1): 42－46.

[10] Valiant G. A bridging model for parallel computation [J]. *Comm. ACM*, 1990, 33(8): 103－111

.

邵银波： 男，1978 年生，博士生. 从事高速实时信号处理技术及其应用.

李　强： 男，1969 年生，教授，博士，研究领域为高速 DSP 技术、多媒体技术.

王永良： 男，1965 年生，教授，博士生导师. 已发表论文 150 多篇。出版专著《空时自适应信号处理》、《空间谱估计理论与算法》，曾获省部级科技进步一等奖两项，二等奖、三等奖各一项. 曾获教育部“全国青年教师奖”、“全国优秀教师”和人事部“中国优秀博士后奖”，入选“新世纪百千万人才工程”国家级人选。研究领域为雷达技术、阵列信号处理、自适应信号处理等.

陈　辉： 男，1974 年生，讲师，已发表论文 20 余篇. 研究方向为超分辨谱估计、阵列信号处理.