声纳阵元域信号并行仿真系统设计

王希敏，蔡志明，邱 风，许任洲
(海军工程大学，武汉 430033)

摘要：采用由多片 DSP 和工控机为核心组成的 DSP 并行处理系统，进行声纳阵元域信号仿真。为满足实时性与通用性的应用需求，探讨基于并行 DSP 系统的仿真系统的软硬件结合设计方法。根据仿真算法，设计仿真系统任务流水线并行模式；基于软件工程方法和设计模式，设计便于流水线阶段映射的 DSP 进程框架和适当粒度的任务封装；采用时延隐藏方式设计 DSP 间的数据通信。使用表明，这些设计既满足了实时性能的要求，又解决了并行多任务实时系统通常难以进行适应性维护和功能重配置的问题。

关键词：信息处理；并行计算；流水线并行模式；TigerSHARC DSP；实时仿真；声纳信号
中图分类号：TP391.9 **文献标识码**：A **文章编号**：1000-3630(2009)-04-0550-06
DOI 编码：10.3969/j.issn1000-3630.2009.04.025

The design for parallel computing simulation of waveforms in the front end of sonar

WANG Xi-min, CAI Zhi-ming, QIU Feng, XU Ren-zhou
(*Naval University of Engineering*, *Wuhan* 430033, *China*)

Abstract: A commercial off-the-shelf (COTS) parallel processing system, consisted of multi-DSPs and a control computer, is used as a hardware platform for the online simulation of waveforms in the front end of sonar. The simulation system should keep open and run in real time, so the design issues for the simulation software in the multi-DSPs platform are addressed. The task parallel mode of the simulation system is designed. The program framework of DSP process is defined as an active object based on the object-oriented for the pipeline stage being smartly mapped onto DSP. Latency-tolerance technology is used for the design of data communications between DSPs. Practices show that the constructed simulation system runs in real-time, and can be easily modified and reconfigured which are usually difficult for the multi-task parallel DSP system.

Keyword: information processing; parallel computing; pipeline parallel mode; TigerSHARC DSP; real-time simulation; sonar signal

1 引 言

声纳阵元域信号实时仿真，是指在虚拟海洋环境中与虚拟作战态势下，实时产生声纳传感器阵所接收的信号。阵信号包括多个目标的辐射噪声或回波信号、海洋环境噪声、本舰噪声干扰、拖体流噪声干扰以及海洋混响干扰等声纳接收全信号。声纳阵元域仿真可用于声纳装备性能测试、模拟训练与作战效能研究等。

声纳阵元域仿真的用途决定了仿真系统必须具备实时、开放及可复用的特征。实时的含义是指声纳全信号仿真生成的速度要与声纳信号处理速度相匹配。这个速度与声纳信号频率或系统带宽相关联。因此，仿真系统必须是一个高效的并行处理系统，其整体吞吐率要能支持多达几百路的复杂信号计算，且最高信号频率可达 100kHz。目前，一般选用成熟的通用 DSP 板卡作为实时仿真系统的集成基础。总体硬件规模根据单片 DSP 的处理能力以及板卡上 DSP 的数量决定。这种硬件体系的设计要点是 DSP 之间的拓扑连接，使之最优地支持任务并行性。开放的含义是指仿真的主要对象(包括目标、声纳运载平台、声纳、海洋环境)的属性可由使用者在任务启动前选定，或由操作者在任务过程中修改。这就要求仿真系统是可重配置的，且能不断地进行适应性维护以满足新的需求。可复用的含义是指在保持各主要对象间复杂逻辑关系的前提下，可以方便地对它们的属性与方法进行扩展和修订。例如，随着装备技术及装备体制的发展，声纳的功能、性能参数以及波形不可避免地需要重定义。

收稿日期：2009-01-10；**修回日期**：2009-04-10
作者简介：王希敏(1963-)，女，副教授，研究方向为军用仿真系统体系结构。
通讯作者：王希敏，E-mail: zmcai@jlonline.com

针对仿真系统的开放性需求，采用面向对象的方法建立了仿真软件架构模型[1]。为了满足实时和硬件可扩展的需求，采用由多片 TigerSHARC DSP 和工控机为核心的 COST 并行处理系统硬件。因此如何设计功能可重配置的实时分布式并行 DSP 软件是仿真系统设计面临的主要挑战。

2　仿真系统硬件结构设计

仿真系统硬件组成包括：一台 CompactPCI 总线的工控机箱，主板上含有 Intel Pentium III 或更高的处理器；256MB 或更多内存；机箱中插入了 1 块 ADSP-TS101S DSP 板卡和多块 DA 板卡。如图 1 所示。

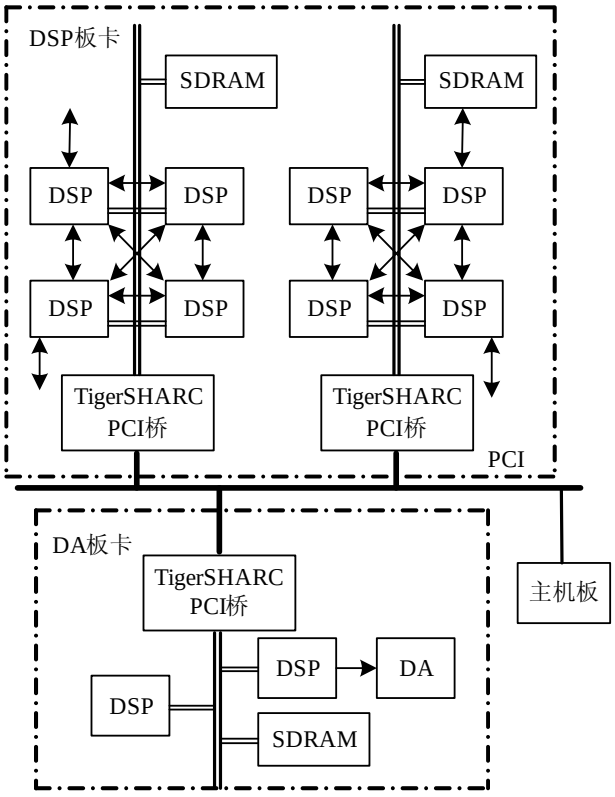

图 1　仿真系统硬件

Fig.1　Hardware of simulation system

DSP 板卡上共配置了 8 片 TigerSHARC (ADSP-TS101S)芯片。将这 8 片 DSP 芯片分为两个 DSP 簇，每个 DSP 簇的 4 个 DSP 通过 Cluster 总线互相连接，同时每簇的 DSP 间通过多条 Link 口互连，提供了分布式处理的系统结构。簇间也通过 Link 口进行互连。TigerSHARC DSP 这种处理器间的互联能力为构成不同的 DSP 硬件拓扑结构提供了基础。这种结构大大增加了 DSP 间数据交换的灵活性，且便于扩展。

另外多个 DSP 通过局部总线连接一个容量较大的全局共享外部存储器。这个共享存储器可以作为 DSP 间数据交换的大型缓冲区。

在与主机通信方面，所有 DSP 以及外部共享全局存储器都通过共享的局部总线及 PCI 接口控制器连接到 PCI 总线。这样主机可以直接访问任何一个 DSP 处理器的内部存储器及外部共享全局存储器。因此仿真系统硬件结构支持共享内存与消息传递的并行软件模型的实现。

DA 板卡上含有一个 DSP 和输出 32 路模拟信号的 DA 转换器；DSP 与 DAC 之间的接口采用 FPGA 实现，其中包括用于缓存 DSP 送往 DA 芯片的数据和地址信息的 16K 字的 FIFO。DSP 通过总线慢速设备协议向 FPGA 发送数模转换所需数据。每个 DA 通道都对应着一个地址映射。采用二维 DMA 控制方式，软件实现 DSP 片内存储器与 FIFO 之间的数据传输。DA 板卡与 DSP 板卡的 DSP 之间可通过 Link 口实现点对点通信。

3　仿真系统并行软件设计

在 COTS 并行 DSP 系统上，软件以构件化封装、合理构件粒度设计、负载平衡的任务映射和时延包容为主要技术手段，达到了实时、可复用和可配置的目标。

3.1　DSP 软件的可复用与可配置设计

3.1.1　软件构件化

按照计算与通信功能分离原则，以可复用与可配置为设计目标，仿真系统的构件设计为四大类：

(1) 仿真任务构件。是系统的核心，完成并行计算任务。又具体分为目标信号计算仿真、背景噪声计算仿真、混响计算仿真、反波束形成计算仿真等任务构件。

(2) 通信构件。面向并行 DSP 系统的消息传递构件。可屏蔽 DSP 之间数据传送的物理链路及物理拓扑的异构，便于计算任务在 DSP 节点上的配置与迁移。

(3) 管理配置构件。管理与配置各 DSP 处理器任务构件，其构件功能包括，根据不同的声纳阵型与采样频率，实例化每一 DSP 处理器上的计算仿真任务对象，并设置对象属性。

(4) 公共服务构件。设计与实现面向 DSP 应用的内存池，支持 DSP 节点任务的动态创建与配置[2]。

3.1.2　软件分层架构

软件由构件组合而成，构件之间的关系称为软件架构。从大粒度上看，架构为三层结构，如图 2

所示。底层为硬件抽象层，由设备(驱动)对象或函数库组成。中间层即为简易中间件，由内存池、数据通信构件组成，主要负责内存与各种通信接口资源的管理。顶层为应用层，由任务构件、信号处理库、以及管理构件组成。

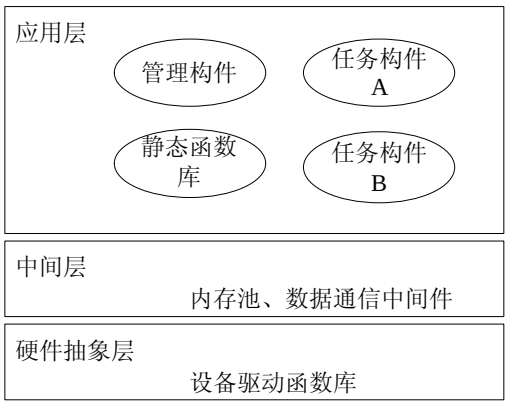

图 2 分层架构模型图
Fig.2 Layered structure diagram

3.1.3 仿真任务构件封装

为了便于任务对象在 DSP 节点上的分布，实现 DSP 软件功能的静态重配置，对仿真任务进行了合适粒度封装，任务实现代码独立于任务在 DSP 进程中的分配。一个仿真计算任务是由一组耦合非常强的类对象实例通过互相协作完成，因此仿真任务构件是一类簇构件[3]。仿真任务构件封装是实现系统重配置的基础。

任务粒度是一个任务执行时间和内存的度量。小粒度便于任务并发，但有较大的管理与通信开销。并行处理任务封装粒度与平行度、通行量等问题有关；内存需求、执行时间、处理器的个数是粒度设计的主要影响因素。对于流水线并发多任务，采取了先按任务并行划分任务并封装，再按数据并行分布任务对象到 DSP 进程。

例如，图 3 中的目标信号计算仿真任务构件中封装了完成该任务所需的实体对象成员和任务执行的主控制流程，是实体类对象的组合体，其中目标信号仿真类是该类簇对应的主控类。

3.2 DSP 软件的高性能并行与分布式设计

影响分布式并行系统实时性的主要因素是任务平行度和数据的通信时延。提高平行度的关键是发现任务和数据的并发性。

数据通信时延与任务分布及数据通信模式等因素有关。从软件功能重配置的角度看，如何实现并行任务灵活地向处理器节点映射也是一个关键问题。由此可见，DSP 软件设计是一个软/硬件的协同设计问题，综合了硬件结构、计算仿真算法、分

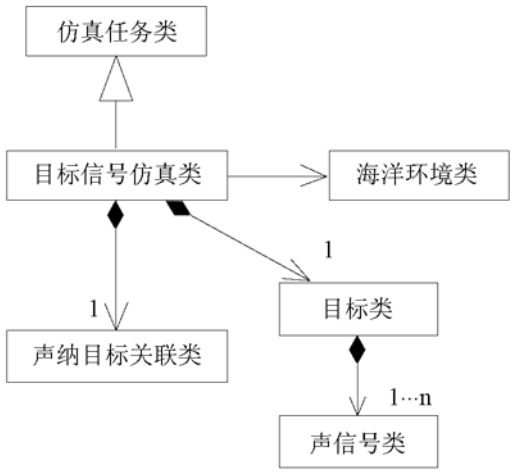

图 3 目标信号仿真构件类图
Fig.3 Classification diagram for target signal simulation components

布式并行处理方法等多方面问题。这些问题也是高性能分布式计算、软硬件任务分配[4]、任务映射与调度[5]及嵌入式软件架构[6]等领域的研究热点。但是，目前还没有一个有效工具和系统的理论方法能够解决实际工程中的应用问题。

在声纳阵元域计算仿真的软/硬件结合设计中，从工程应用出发，成功解决了如下关键设计：

(1) 根据声纳阵元域计算仿真功能与数据流，设计 DSP 软件的任务并行模式，将计算与数据并行化；(2) 结合 TigerSHARC DSP 硬件结构和计算仿真任务的构件粒度(以计算负载与数据存储需求为衡量)，设计任务的映射方法；(3) 为了减小系统时延和提高数据吞吐，综合考虑 DSP 输入/输出特性和并行软件的流水线并行模式，设计时延隐藏的 DSP 间数据通信，提出 DSP 内核运算与数据传输并发的实现方法。

3.2.1 流水线模式用于任务并发的设计

首先根据计算功能划分任务，再由任务数据流确定任务与数据的并发模式。声纳阵元域信号仿真计算过程从数据流上分为三阶段：(1) 多个目标信号仿真。功能是根据作战态势、海洋环境、声纳频段及目标声特性，输出目标信号波形。(2) 多个目标信号反波束计算并求和，并将其结果与海洋环境噪声、混响仿真信号相加形成多个阵元的声纳基阵信号输出。(3) 通过 D/A 转换，输出多路声纳基阵模拟信号。每一阶段的输入数据是上一阶段输出的结果；各阶段的计算任务是顺序相关，且以固定周期同步地更新数据；第一阶段中的任务并发地计算多个目标，第二阶段中的任务并发地计算多路信号，如图 4 所示。可以看出，仿真任务在时间顺序上是并发的，以流水线模式并行完成。

由于每一阶段的任务在给定时间内所能处理的

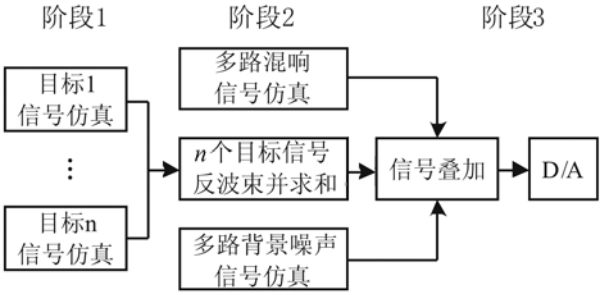

图 4　阵元域计算仿真流水线

Fig.4　Pipeline of sonar signal simulation

数据量要满足声纳实时处理的要求，数据吞吐量和数据计算时延是流水线设计中的重要指标。因此流水线模式的仿真计算设计问题主要包括[7-9]：(1) 负载平衡下的阶段任务分配与映射，其中最慢的阶段是总吞吐量的瓶颈；(2) 由于流水线上各阶段任务在时间顺序上是紧耦合并发，阶段间的数据流与控制流紧密耦合，一个阶段的修改或扩展即会影响全局。因此，须分离流水线阶段间的数据流与控制流。

针对上述问题，将基于流水线模式的并行计算进行如下设计：(1) 采用硬定时进行任务间同步。即系统定时产生外部硬中断，在每一 DSP 节点上的中断服务线程中处理分布在该节点上的任务，这样每一阶段的任务的调用或驱动在时间上没有依赖，只要求分布在各 DSP 节点上的计算服务线程满足定时要求；(2) 为了易于流水线任务的映射与扩展，设计可复用的 DSP 进程实现框架，将计算、通信与管理配置分离；(3) 为了提高数据吞吐量，采用计算与通信重叠方式隐藏通信时延。

3.2.2　可复用的 DSP 进程设计

一个或多个流水线阶段任务可映射为一个或多个 DSP 进程。一个 DSP 进程内部拥有自己的管理控制线程和服务线程。遵照计算方法的执行与调用分离原则，DSP 进程实现结构如图 5 所示。

DSP 进程可看作是主动对象的组合体，是由主线程和中断服务线程组成。主线程是 DSP 节点的控制线程，最基本的操作是根据不同的声纳阵型与采样频率，实例化 DSP 节点上中断服务线程及仿真任务对象，并设置对象属性。主线程的控制管理功能受控于运行在工控机 CPU 节点上的主进程。共享存储用于主线程与主进程之间的信息交换。

中断服务对象最基本的行为是查询任务队列中的仿真任务对象并执行，即先读输入消息队列中的数据，然后执行任务队列中仿真任务并存放结果数据到输出消息队列。输入/输出消息队列用于连接各个 DSP 进程。数据通信的基本操作是完成进程间的数据通信，即数据的编组/解组、路由与传输。中断

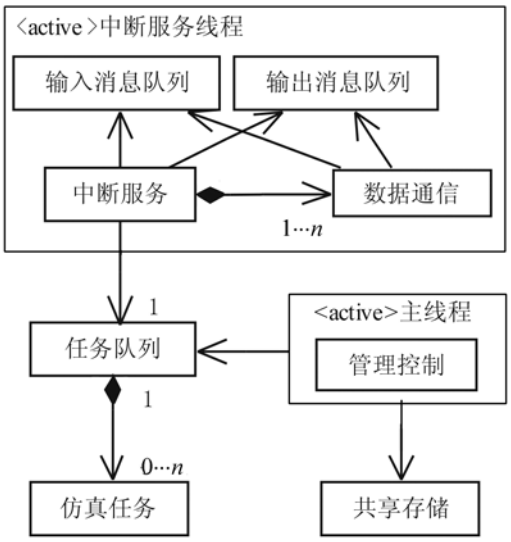

图 5　可复用的 DSP 进程实现

Fig.5　Implementation for reusable DSP process

服务与数据通信对象运行在由硬定时触发的中断服务线程中。

3.2.3　负载平衡的任务映射

一个通用并行 DSP 硬件平台须面对不同的声纳类型，根据负载平衡、通信时延及数据吞吐的性能要求，设计映射方案。由于 DSP 速度与存储空间的限制，任务对象到 DSP 的映射必须综合考虑任务需求与硬件资源属性。映射原则是计算负载与存储资源的均衡，尽量提高并行度，满足数据吞吐要求。

一方面要平衡每一节点上的任务运算量。要先测试每一任务的运算时间，然后按同等运算量的原则将流水线上同一阶段或相邻阶段的任务分配在同一组。根据每一组任务计算复杂性和每一节点的中断服务线程运行时间满足定时要求，选择完成每组任务所需的 DSP 进程数。同一组的任务再分配到一个或多个进程的任务队列，进程又映射到 DSP 上。为了提高运算速度，可将计算密集的部分代码封装成静态库，这些代码在编译生成时，可选择优化选项。在代码实现中，要充分利用面向 TigerSHARC DSP 体系结构的优化信号处理库。

另一方面要根据节点上可用的数据存储量，分配 DSP 存储空间。ADSP-TS101S 有三个内存大小为 64K 字的片内存储器块，任务的局部数据分布在片内，全局数据分布在共享外部存储器。为了提高软件的重配置能力，任务中的部分对象要动态地创建，即要动态分配内存。因此必须根据静态数据与动态数据存储量划分 ADSP-TS101S 的内存块，定义静态数据段、堆段和栈段的大小。

软件运行时，每个 DSP 节点定义一个 DSP 进程。节点主线程读取节点任务配置，并定义与配置

DSP 进程的任务队列中的任务对象，完成任务对象映射到 DSP 的操作。这样，就实现了在一个硬件平台上完成不同类型声纳的全阵元域信号计算仿真。

3.2.4 时延隐藏的数据通信设计

TS101 DSP 可通过 Link 口和簇总线进行通信，这种处理器间的互联能力为构成不同的多 DSP 拓扑结构提供了基础。根据数据流，可灵活地进行拓扑设置。数据通信设计的目的是保证流水线的数据吞吐量，减少传输时延。由于 Link 口和簇总线数据传输速率的限制，内核运算量与数据传输量只能在一定程度上达到平衡。因此，为了提高 DSP 的运行效率，达到隐藏数据通信时延的目的，可将数据包在后台提前一拍传送。因此需采取计算与通信重叠的时延包容方式开发硬件资源间的重叠。

计算与通信重叠的实现依赖于 DSP 软件的并行模式和硬件 I/O 机制。结合流水线并行模式和 ADSP-TS101S DSP DMA 控制器与计算内核独立的特点，运用乒乓操作和非等待 DMA 控制方式进行节点间数据交换，实现 DSP 上的内核计算任务与数据传输任务重叠执行。

乒乓操作是避免 DSP 任务重叠执行造成存储资源冲突的一种方法，所付出的代价是存储资源。要实现乒乓操作，在 DSP 内存空间中，需要定义结构相同的两片数据区和一个存储标志。根据标志，当一片数据区用于接收数据，另一片则用于处理数据，处理完数据后，标志反转。由于 DMA 数据传送不需内核干预，因此在 DMA 控制下，用于节点间的数据通信的时间开销主要是 DMA 通道的初始化与启动。图 6 是计算与通信任务并发活动设计。在 DSP 节点上，信号仿真任务与数据传输任务分别在内核与 DMA 控制器控制下可以重叠执行。

DMA 传送还必须采用非等待方式，否则将不能实现内核处理与数据通信的完全并发。所谓非等待方式，即节点间是异步通信，这就要求在一个定时中断服务中，每个 DMA 通道只能启动一次 DMA 传送。由此，流水线的数据吞吐量将取决于 DSP 上的任务处理时间，数据的通信时间可以忽略。

4 仿真系统软件实现

仿真系统软件可分为两部分：实时并行 DSP 程序和主机显控程序。

实时并行 DSP 程序是基于上述方法设计，在 AD 公司提供的集成开发环境 VisualDSP++3.5 下，

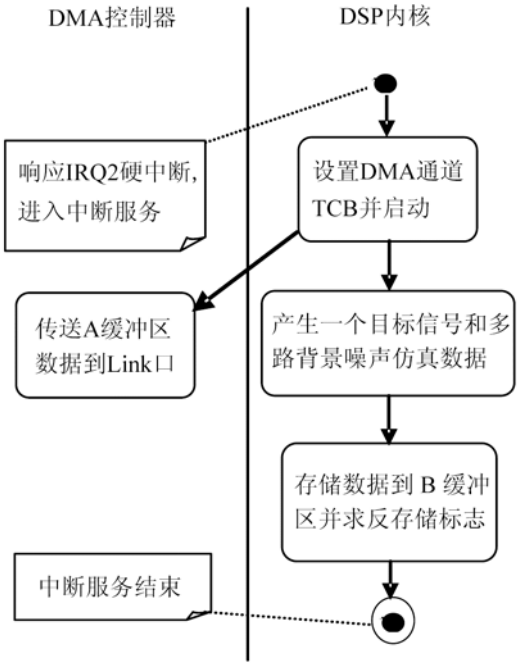

图 6 计算与通信任务活动图

Fig.6 Active diagram for computation and communication tasks

用标准 C++/C 编程语言和 EZ-DSP 信号处理库开发完成。

主机的操作系统采用 WindowsXP，主机显控软件开发环境是 VC++6.0。主机软件主要完成四方面的工作。(1) 人机交互。海洋环境、目标、声纳平台参数的设置以及任务对象初始化配置并传送给 DSP；(2) 根据用户的选择，进行目标态势仿真，并将仿真结果传送给 DSP；(3) DSP 复位和 DSP 软件下载。(4) 收集所有 DSP 运行结果并显示。

由于主机通过 PCI 总线可以直接访问任何一个 DSP 处理器的内部存储器及外部共享全局存储器，主机与 DSP 之间采取共享存储的方式交换数据。主机将在线实时变化的参数（如目标运动参数）以广播方式定时地写入各 DSP 的片内，而将大量不必实时更新的参数和初始化数据写入片外 SDRAM。主机通过 PCI 设备驱动函数库访问 DSP 内部存储器及片外 SDRAM。设备驱动程序基于 WinDriver 开发实现。

该系统已实现了多种声纳型号的阵元域波形仿真功能重载。每个型号声纳均可关联多个不同的目标，且目标的数量可变。声纳的工作频段、采样率、通道数和处理功能均可改变。这些选项的改变均涉及到仿真算法、运算规模以及 DSP 间数据流向的根本改变。目前已实现了基于同一硬件结构的 DSP 软件静态重配置。

5　结束语

为了满足声纳阵元域信号实时仿真的需求，采用由多片 TigerSHARC DSP 和工控机为核心的并行处理系统硬件。针对仿真系统开放性需求，结合构件化的软件设计方法，根据声纳阵元域信号仿真算法特点，提出了 DSP 并行计算软件的设计中的基本问题与解决方法。重点讨论了并行处理系统中有关任务构件的封装、任务映射和时延、隐藏的数据通信等设计与实现方法。由于这些方法的采用，解决了并行多任务实时系统通常难以进行适应性修改的难题。对于进一步定量分析与设计可复用的、动态重配置的高性能实时仿真系统具有指导意义和应用价值。

参　考　文　献

[1] 王希敏, 蔡志明. 水声信息系统仿真软件构架模型[J]. 兵工学报, 2007, **28**(4): 471-476.
WANG Ximin, CAI Zhiming. Software Architecture on the Modeling and Simulation for Underwater Acoustics Information Systems [J].Acta Armamentarii, 2007, **28**(4): 471-476. (in Chinese)

[2] WANG Ximin, WANG Zhe. Design and implementation of memory pools for embedded DSP[A]. Proceedings –International Conference on Computer Science and Software Engineering[C]. Wuhan, 2008, 160-164.

[3] 杨芙清, 梅宏. 构件化软件设计与实现[M]. 北京: 清华大学出版社, 2008: 154-184.
YANG Fuqing, Mei Hong. Design and Implementation of Component-Based Software[M]. Beijing: Tsinghua University Press, 2008: 154-184. (in Chinese)

[4] Madhura Purnaprajna. Genetic algorithms for hardware-software partitioning and optimal resource allocation[J]. Journal of System Architecture, 2007, **53**: 339-354.

[5] Silvia M. Figueira. Optimal partitioning of nodes to space-sharing parallel tasks[J]. Parallel Computing, 2006, **32**: 313-324.

[6] Anu Purhonen, Eila Niemela, Mari Matinlassi. Viewpoints of DSP software and service architectures[J]. The Journal of System and Software, 2004, **69**: 57-73.

[7] 陈国良. 并行计算-结构.算法.编程[M]. 北京: 高等教育出版社, 2003: 161-179.
CHEN Guoliang. Parallel Computing: architecture, algorithm, programming[M]. Beijing: Higher Education Press, 2003: 161-179. (in Chinese)

[8] Timothy G. Mattson, Beverly A.Sanders, and Berna L.Massingill. 并行编程模式[M]. 敖富江译, 北京:清华大学出版社，2005:89-113.
Timothy G. Mattson, Beverly A.Sanders, and Berna L.Massingill. Patterns for Parallel Programming[M]. AO Fu-Jiang, ttanslated. Beijing: Tsinghua University Press, 2005: 89-113. (in Chinese)

[9] Steve MacDonald, Duane Szafron,and Jonathan Schaeffer. Rethinking the pipeline as object-oriented states with transformation[A]. Proceedings of the 9the International Workshop on High-Level Parallel Programming Models and Supportive Environments (HIPS 2004)[C]. 2004: 12-21.