计算机科学 1998 Vol. 25 №. 4

可重构计算和可重构计算机技术

Reconfigurable Computing and Reconfigurable Computer Technologies

佟 冬 胡铭曾 方滨兴

(哈尔滨工业大学计算机系 哈尔滨150001)

摘 要 In this paper, we present a definition and a classification of reconfiguration. Then we give a survey of reconfigurable technologies with some detail of implementing each level. Besides, the development of dynamic programmable devices, such as FPGA and FPID, make implementation of reconfiguration possible. In future, FPGA and FPID must play significant roles in this research area.

关键词 Reconfiguration, Reconfigurable computing, Reconfigurable computer, FPGA, FPID

现代的国防、科研及社会的很多方面都需要大量的计算，尤其是国防工业对计算的要求更为严格，为国防工业设计的计算机必须速度快、适应性强、可靠性高[1]。然而，现有的计算机不能高速地处理多种应用、达到高效率，究其主要原因是计算机与应用算法的不匹配[2]。解决的基本方法是使计算机可以改变自身来适应应用的要求。目前计算机的硬件是不可改变的，这些包括：基本电路不可变、处理器不可变、处理器互联结构(互联网络)不可变。另外，为不同的应用开发不同的计算机，同时也需开发不同的软件，包括操作系统及应用软件，这使软件的开发费用远远高于硬件。降低软件的开发费用也是设计大型计算机急需解决的问题。

采用可重构技术，可以解决上述问题。可重构的计算机通过对自身软硬件的改变，来适应不同应用的要求，可实现高速，高适应性和高可靠性。本文将对可重构技术进行探讨。

1 可重构的概念

所谓可重构是指在软件的控制下，计算系统如果可以利用可重用的资源，重构或重组成另一不同的计算系统，以适应不同应用的要求，则称这种计算系统是可重构的。

重构与重组是可重构计算系统改变其功能的两种方式。如果新计算系统的功能部件，在旧的计算系统中不存在，是用可重用的资源重新生成的，则称这种方式是重构。如果新计算系统的功能部件，在旧的计算系统中存在，只是通过重新组合形成新的计算结构，则称这种方式是重组。

可重用的资源是可重构的物质基础。在可编程器件出现之前，可重构计算机大多采用重组的方式，即可重用的功能模块在原系统中存在。八十年代中期出现的大规模可编程器件 FPGA 和 FPID，使重构方式成为可能。FPGA 和 FPID 的重用资源是基本的门和线。可以通过配置文件，配置每一个门的性质和线的连接，从基础上改变硬件的功能。

可重构的目的有两点，一是为了提高计算机的性能，使之能适应多种不同应用的要求；二是为了节省软硬件的开发费用，尽可能使用已有的资源来构造新的系统。

可重构技术是为了解决计算机与应用要求不匹配的问题，所以可重构可以按解决不同问题的层次分成四类：电路级可重构；指令级可重构；结构级可重构；软件级可重构。

可重用的资源决定可重构计算系统的性质。如

[14]S. Rosenschein, and L. Kaelbling. The systhesis of digital machines with provable epistemic properties. In Halpern, J. Y., editor, Proc. of the 1986 Conf. on Theoretical Aspects of Reasoning about Knowledge, Morgan Kaufmann Publishers, San Mateo, CA.

[15]A. S. Rao, and M. P. Georgeff, A Model-theoretic approach to the verification of situated reasoning systems, In Proc. of the 13th Intl. Joint Conf. on Artificial Intelligence(IJCAI-93), Cambery, France

[16]M. Wooldridge, The Logical Modelling of Computational Multi-Agent Syatems, PhD thesis, Department of Computation, UMIST, Manchester, UK, 1992

果可重用的资源是硬件，则计算机系统是硬件可重构的。如，可重用硬件资源可以是门级的，也可以是处理器级的。如果可重用的资源是软件，则计算系统是软件可重构的。

可重构技术又可按重构发生的时间分为静态可重构和动态可重构[4]。如果重构发生在系统运行前，即系统运行时不能重构，则称为静态可重构。如果在系统运行时可以重构，即系统本身可以根据不同的条件改变自身的功能，则称为动态可重构。

2 电路级可重构

目前一般的计算设备由多个固定功能的 VLSI 芯片组成，所以不能改变其硬件功能来适应应用算法的要求。如果将这些功能部件的逻辑功能用 FPGA 实现，当应用算法改变时，可以通过改变每个 FPGA 的配置来改变其功能，从而改变整个计算设备的功能。这种从基本的门级入手重构计算系统的方式，称为电路级可重构。目前的电路级可重构的计算设备一般由多个 FPGA 和 FPID 组成，其实际应用有两类：ASIC 仿真器（ASIC Emulator）和用户定制计算机（Custom Computer）。

基于 FPGA 和 FPID 的 ASIC 仿真器由仿真软件和硬件组成，可以实现大规模的门级逻辑设计，自动地用硬件模拟 ASIC。仿真软件将设计映射到多个 FPGA 中，并分配 FPGA 之间的连接。仿真时，主机将激励信号传送给仿真器，并得到输出结果，以验证设计的正确性。与常用的逻辑设计验证方法（软件模拟，原型试验等）相比，这种仿真器具有快速经济的特点，是目前大型 ASIC 设计验证的主要方式。比较成功的商业化仿真器是 Qickturn 公司的 RPM[5]和 Realizer[6]，RPM 的结构是多 FPGA 组成二维 MESH 网，Realizer 是由 FPGA 和 FPID 组成的 partial crossbar 结构，MESH 和 Partial crossbar 结构是目前多 FPGA 中使用最多的互连结构，其结构如图1所示，其中 F 代表 FPGA，L 代表 FPID。

用户定制计算机是通用专用化计算设备，同样由多个 FPGA 按一定结构构成，可实现用户定义的大计算量计算任务。设计者将算法直接用硬件实现，并映射到多 FPGA 系统中。比起算法在通用机上的软件实现，用户定制计算的速度更快，而且它可以象通用机一样重新编程以适应另一应用，区别在于所编的是硬件。著名的用户制定计算系统有 SPLASH[7]等。SPLASH 是多 FPGA 的线性 systolic 阵列，擅长完成串比较算法，在某些特定应用下，其性能可超过 Cray-2。

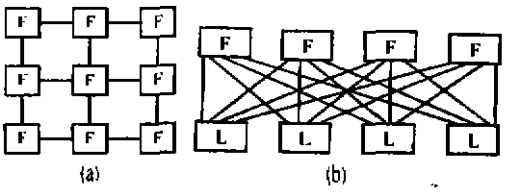

图1 多 FPGA 系统常用结构：
(a)MESH 结构
(b)partial crossbar 结构

3 指令级可重构

一般的通用计算机的设计目标是为尽可能多的任务提供满意的性能，所以其指令集都是功能简单的指令。对于那些对性能有特殊要求的计算任务，如数字信号处理等，往往不能胜任。解决的方法是设计专用的计算设备。为通用计算机提供特殊的计算支持，以实现大计算量指令和子程序的执行，这种提高通用计算机性能的重构方法称为指令级可重构。

指令级可重构的研究最早开始于六、七十年代，当时的很多机器都利用了微码技术，即用几个微操作来完成一条指令的执行。这些微代码存在可重写的 ROM 中，设计者可以通过改写指令的微操作来修改指令的行为，以达到重新设计指令集的目的。同时，处理器内部的各功能模块可以在微码的控制下改变连接方式，来改变处理器的结构。将这一过程自动化，用编译来完成从设计者的要求到微码改变的过程，是后期的研究热点。

可重配置器件 FPGA 的出现为指令级可重构提供了新的器件和新的方法。设计者不用改变处理器的任何设计，只需将 FPGA 或 FPGA 阵列连接在处理器或通用计算机上，正常的指令或子程序在通用处理器上执行，特殊指令或子程序在 FPGA 上执行，即可完成增强指令集的目的。

在现有的系统中，Harvard 大学的 PRISC[8]是处理特殊指令的指令级重构，法国 DEC 实验室的 PAM[9]和美国 Brown 大学的 PRISM[10]是处理大计算量子程序的指令级重构。PRISC 由通用 RISC 处理器及可编程的 FPGA 功能部件组成，FPGA 功能部件完成 RISC 指令集之外的特殊指令。PRISM 和 PAM 由通用计算机和可重构的多 FPGA 辅助计算设备组成，辅助设备提供一些特殊子程序的硬件实现。通用计算机处理控制及一般的简单计算，辅助设备处理特殊的大计算量计算。

电路级和指令级可重构系统的基本硬件结构如图2所示。电路级和指令级可重构的共同特点是利用

可编程的器件使基本电路和处理器的功能改变，FPGA 和 FPID 是这两种可重构技术的基本实现手段。其区别在于是否有指令或子程序的概念。指令级可重构计算机的使用是程序级的，编程人员在程序中使用特殊指令或特殊子程序，再由编译连接可重构硬件执行，也可为自定义的特殊应用直接编写硬件和由编译自动生成硬件。电路级可重构计算设备是门级的。设计者必须直接设计算法的硬件实现，没有指令或子程序的概念。

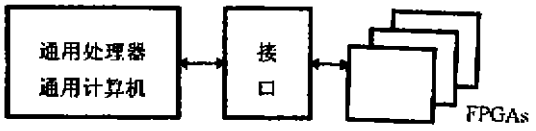

图2 电路级和指令级可重构计算设备的一般结构

目前，电路级可重构和指令级可重构的研究重点有两点：一是设计多 FPGA 系统的通用互连结构，提高 FPGA 和 FPID 的利用率；二是研究高级描述（程序或硬件描述语言）到 FPGA 实现的自动映射软件，包括综合、电路划分及配置文件的生成等。

4 结构级可重构

由多个处理器按一定互连结构组成的并行计算机是实现大型计算的基本手段。但由于其结构不可变，使其潜在能力（加速比）不能达到最高。正如前面所说的，主要因为处理器不可变、处理机个数不可变、处理器间互连网络不可变。这些因素都属于体系结构的范畴，我们称上述因素可改变的计算机为结构级可重构的。

4.1 处理器可变

一个处理器可以用两个参数表示：指令集和基本字长。指令集的可重构在上一节已经详细地讨论过，这一节说明基本字长的可变性。基本字长可变，可以通过处理器级连实现，来满足应用对不同精度的要求。早期的一些并行机如早期 MPP 使用的是位处理器，MPP 由128×128个位处理器组成，可以灵活地实现可变字长的标量和向量运算。一般地，如果处理器的字长为 m（m<n），则进行 n 位计算需要 n/m 个级连处理器同时计算。

4.2 处理器个数可变

这里所说的处理器个数是相对于应用而言的，即应用程序需要 n 个处理器进行并行计算，可重构计算机应该有为此应用提供 n 个处理器的能力。目前为止，大多数处理器个数可变的计算机，都是通过大量的处理器冗余实现可重构的。早期的 PASM[11] 就是这样的可重构计算机。PASM 是 Purdue 大学80年代初设计制造的可重划分的 SIMD/MIMD 计算机，它由几个中央控制器统一控制的 SIMD 计算机组成。中央控制器根据应用计算的性质，决定由几个 SIMD 计算机进行计算，并决定采取 SIMD 或 MIMD 形式计算。目前有许多通用巨型计算机也是处理器个数可变的，如 Intel Paragon。同时，当处理器字长变化时，处理器个数也相应地变化。

4.3 互连网络可变

并行计算机的并行性很大程度上由处理器之间的互连网络决定。改变互连网络可以使计算机适应不同的算法，达到计算可重构的目的。互连网络可重构分三种情况：一是互连网络的基本带宽可变；二是互连网络的划分；三是互连网络的拓扑结构可变。

处理器的基本字长可变要求互连网络的基本带宽也必须可变，带宽的变化可以采用几个网络通路合并的方式实现。处理器数可变要求互连网络必须是可划分的。同时，计算机基本字长的可变也可以用网络划分实现。

不同的算法往往需要不同的互连网络拓扑结构，因此互连网络的可重构对可重构计算有着十分重要的意义。互连网络重构有两种实现方法，一种是模拟，即用一种互连网络模拟另一种；另一种是冗余通路直接重构网络。互连网络的可重构是目前的研究热点。其中，Reconfigurable Mesh[12] 通过在 Mesh 网的每个结点上添加多功能开关，使非近邻的结点也能直接交换数据。Reconfigurable Ring[13] 由多个环形网络组成，处理器对每个环都有通信结点，通过控制通信结点的开关，可实现多种网络拓扑结构。

结构级可重构的一个重要应用是容错。大量容错的研究都是关于如何通过计算机部件的冗余，使当某个部件出错时，计算机仍能继续正常工作。可以说，容错计算机就是可重构的计算机。

到目前为止，对结构级可重构的研究大都是利用重组的方式。FPGA 和 FPID 技术的出现为结构级可重构提供了新的器件，使重构成为可能。分析这两种器件可发现，FPGA 的重构重点是逻辑门，FPID 的重构重点是连线。而任意计算设备正是由门和连线组成。因此，如果处理器和其他逻辑部件用 FPGA 参与实现，而互连网络用 FPID 参与实现，在软件的控制下，是有可能达到任意重构的。

5 软件级可重构

可重构的软件分三类[14]，可重构的软件；可重定向的软件（Retargetable Software）；可重用的软件。

可重构的软件是管理上述可重构计算机的操作系统和编译程序。可重构计算机对不同应用的不同配置可以看成是计算机的不同状态，可重构软件的功能就是控制可重构计算机如何在不同状态之间切换。其另一个功能是当计算机需要一个新的状态时，如何生成这种状态。可重构的软件可分为三级：显式(explicit)重构，计算机状态的改变由程序的指令显式控制；隐式(implicit)重构，状态的改变由操作系统自动判断和实现；重构的方法论，指只当没有需要的状态时，如何利用已有资源生成这种状态。

随着计算机的功能越来越强，为每一个计算机编写软件的开销也越来越高，如何减少软件的开发费用，已成为必须解决的问题。可重定向的软件技术和可重用的软件技术就是为此目的发展起来的。

可重定向的软件技术是指为一台计算机设计的软件，也可在另一台结构不同的计算机上使用。共有三种解决方法：一是转换(Transport)，设计指令集的转换程序。自动转换软件；二是仿真(Emulation)，用计算机模拟另一台计算机，则可以直接使用被模拟计算机的全部软件；三是定义统一的硬件无关语言，软件可以在不同机器上分别编译生成，早期的 Ada，ANSIC 及近期的 Java 语言都是为这种目的设计的。可重定向软件最成功的例子是 UNIX 操作系统，目前几乎所有的大型机的操作系统都是 UNIX 或类 UNIX。

可重用的软件技术解决如何利用已设计完成的软件，来减少开发费用的问题。早期的软件库和面向对象技术正是为这一目的开发的。目前使用广泛的 VC++和 Java 都是可重用软件技术的杰出范例。

结束语 FPGA 和 FPID 为硬件可重构提出了新的挑战。首先应该是设计思想的转变，以前的设计思想是化简，使硬件资源最少。可编程部件为设计提供了另一设计的思想方式：重配置，即不用合并而只需根据应用的需求重构即可。这种重配置的思想既能节省硬件的开销，又能使硬件达到最高的效率。目前 FPGA 和 FPID 技术的主要问题是，硬件速度不快，器件的利用率不高，自动设计软件还不够完善。但是根据 Noyce 理论，在不久的将来关于可编程器件各种技术一定会有显著的提高。如果，可编程器件的性能能达到目前 ASIC 的水平，那么可以将本文讨论的可重构计算技术有机地结合，完全可设计出全可重构功能的计算机。

可重构技术可以解决目前计算机研究中的许多不足，使计算机的功能更强，适应性更高，开发的费用更少。可重构计算必然在今后的计算机研究中得到广泛的重视。

参 考 文 献

[1] S. I. Kartashev et al., Design for Adaptability, Guest Editor's Introduction, IEEE Computer, 19(2)1986

[2] S. I. Kartashev et al., Supercomputing Systems: Reconfigurable Architectures, Series: Designing and Programming Modern Computer Systems, Vol. Ⅰ, Prentice-Hall Inc. 1989

[3] I-Cube Inc., IQ Family Data Sheet, March 1996. Http://www.icube.com.

[4] Brad L. Hutchings et al., Implementation Approaches for Reconfigurable Logic Application, 5th Intl. Workshop on Field Programmable Logic and Applications, FPGA'95, Aug. 1995

[5] Stephen Walters, Computer-Aided Prototyping for ASIC-Based Systems, IEEE D&T., June 1991

[6] Michael Butts et al., An Efficient Logic Emulation System, Proc. of ICCD'92, 1992

[7] Maya Gokhale, et al., Building and Using a Highly Parallel Programmable Logic Array, IEEE Computer, 24(1)1991

[8] Rahul Razdan et al., A High-Performance Microarchitecture with Hardware-Programmale Fuctional Units, Proc. 27th Annual IEEE/ACM Intl. Symp. on Microarchitecture, 1994

[9] J. Vuillemin et al., Programmable Active Memories: Reconfigurable Systems Come of Age, IEEE trans. VLSI, 4(1)1996

[10] Peter M. Athanas et al., Processor Reconfiguration Through Instruction-Set Metamorphosis, IEEE Computer, 26(3)1993

[11] Howard Jay Siegel et al., PASM: A Partitionable SIMD/MIMD System for Image Processing and Pattern Recognition, IEEE trans. Computers, C-30(12) 1981

[12] R. Miller et al., Parallel Computations on Reconfigurable Meshes, IEEE trans. Computers, Vol. 42, 1993

[13] Arnold L. Rosenberg et al., The Reconfigurable Ring of Processors: Fine-Grain Tree-Structured Computations, IEEE trans. Computers, 46(10)1997

[14] Nikitas A. Alexandridis, Adaptable Software and Hardware: Problems and Solutions, IEEE Computer, Feb. 1986