

一种动态可重构协处理器参数化系统级模型研究

王 鹏, 林水生, 黄乐天

(电子科技大学 通信与信息工程学院, 四川 成都 611731)

摘 要: 提出了参数化系统级模型. 该模型不依赖于具体结构, 以任务布局与重构处理分离的两级结构处理任务调用, 通过参数方式实现不同设计方案的硬件结构和布局算法的配置. 采用 SystemC 语言对模型进行了建模验证, 仿真结果表明, 通过指定上下文的下载、配置和执行等时间开销参数, 在系统级设计空间探索中, 能很好地模拟动态重构协处理器.

关键词: 动态可重构处理器; 动态重构; 设计空间探索; 系统级模型; SystemC 建模

中图分类号: TP302

文献标识码: A

文章编号: 1000-7180(2011)02-0086-05

Research of Parameterized System-level Model for Dynamically Reconfigurable Co-processors

WANG Peng, LIN Shui-sheng, HUANG Le-tian

(School of Communication and Information Engineering,
University of Electronic Science and Technology of China, Chengdu 611731, China)

Abstract: This paper presents a parameterized system-level model, which is independent of any concrete reconfigurable architecture and deals with the hardware tasks with the formation of placement and reconfiguration separated. The hardware architecture and placement algorithm of different design can be configured in the model by parameters. The system-level model is constructed with SystemC, and the results show that it can simulate dynamic reconfiguration for the system-level design space exploration by specifying the parameters of context download time, configuration time and execution time.

Key words: dynamically reconfigurable processor; dynamic reconfiguration; design space exploration; system-level model; SystemC modeling

1 引言

使用可重构技术的 SoC 通过对有限的可重构资源分时复用获得了较高的灵活性, 同时也增加了系统设计空间的自由度. 这使得设计者评估和优化系统性能的工作变得十分复杂, 因为系统性能不仅仅和面积等约束条件有关, 还和软件/硬件划分、任务调度算法、布局算法有关. 动态可重构设计空间探索的相关研究有可重构系统结构模型^[1]、性能分析模型^[2]、搜索方法^[3]及重构系统建模和仿真构架的研究等. 设计空间探索构架试图将系统结构模型和

设计算法集成在一起, 与单一的结构模型和算法评估模型相比, 能够构造一个更加灵活而全面的评估平台, 如 Rsesawe^[4], COSMOS^[5] 和 Perfecto^[6] 等框架.

在当前的动态可重构硬件(或动态可重构处理器)重构处理建模中, 或者是针对具体结构, 如 SyC-ERS, 或者是缺乏对不同算法的支持, 如 DRCF^[7], DRC^[8], 而一个优化的方案是基于对结构和算法的综合考虑, 这使得它们对系统结构供选方案的设计空间探索支持比较有限. 本文提出不依赖于具体结构的动态可重构处理器抽象模型.

收稿日期: 2010-02-04; 收稿日期: 2010-05-08

基金项目: 国家自然科学基金项目(60676014)

2 可重构 SoC 结构及动态可重构协处理器

在可重构系统中,整个系统可以分为 CPU 部分和可重构部分两大块,CPU 部分执行软件任务和整个系统的控制;可重构部分负责一些耗时较多的主要计算部分.作为协处理器与主处理器耦合的动态可重构 SoC 典型结构如图 1 所示.

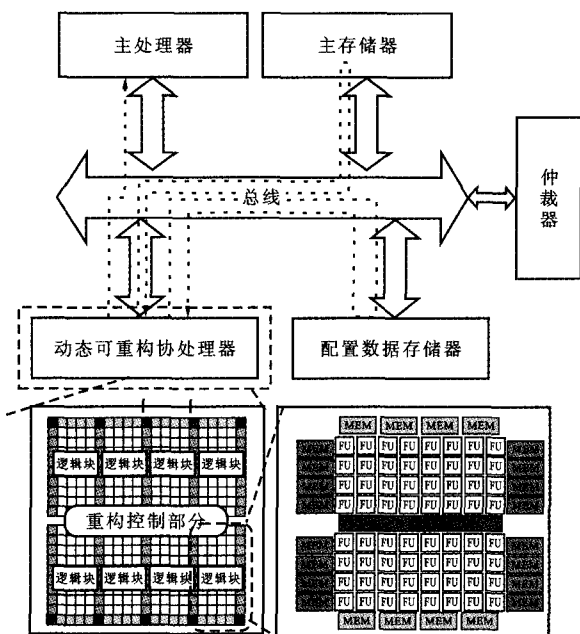


图1 动态可重构 SoC 典型结构

图1结构中的动态可重构处理器(Dynamically Reconfigurable Processor, DRP)能够独立执行较大粒度的计算任务,主处理器不需要对其连续监管,而只需要初始化可重构硬件并发送任务指令和必要数据或数据存储地址的信息.动态可重构协处理器在收到任务信息后,除完成重构和相关计算外,对外主要表现为图1虚线箭头所示的三种行为^[6]:从配置数据存储器获取硬件任务配置比特流;据主处理器给出的地址参数从主存储器获取数据;任务处理结束后向主处理器返回响应信号.

动态可重构处理器结构主要包括重构控制部分和重构资源部分.重构资源由可重构逻辑块组成,每个逻辑块包含多个基本计算单元——功能单元 FU.动态可重构处理器内部可重构硬件资源结构可以在较大范围内变化,在不同可重构硬件技术中,逻辑块可以是大小确定的 slice、column 或 tile.重构资源的规模、互连结构、计算粒度、局部上下文存储器容量等因可重构硬件技术而异,没有统一的结构.为了在统一的抽象模型中得到硬件结构的精确模拟,在系统级设计空间探索中将动态可重构处理器结构

参数化,以一组参数 $Arc = \langle N_{tile}, A_{tile}, R_{tile}, M_{cfg}, N_{mem}, W_{mem} \rangle$ 表征动态可重构处理器的结构,各参数含义如下:

N_{tile} , 逻辑块规模.表示逻辑块的数量.

A_{tile} , 逻辑阵列矩阵.描述各逻辑块位置.

R_{tile} , 路由资源.描述逻辑块间互连结构.

M_{cfg} , 配置模式.一般可分为单一逻辑块单独配置和一行/列共同配置两种模式.

N_{mem} , 局部上下文存储器的容量.表示能存储上下文的个数.

W_{mem} , 局部上下文存储器与总线的接口带宽,该参数会影响上下文传输时读写存储器的次数.

通过将结构参数化,很容易通过参数配置硬件结构.在做设计空间探索时,这些参数将作为执行布局选择的约束条件,影响重构处理过程及结果.

3 DRP 系统级模型

在 DRP 的系统级模型中,关心的是动态可重构处理器的模块功能和模块间的通信,而不考虑具体硬件结构.在接收到硬件任务后,动态可重构处理器的重构处理可以分为两步:

- 在动态可重构逻辑资源内为该硬件任务找到一个合适的位置.
- 配置被选中的逻辑块并执行任务.

为了使 DRP 模型对不同可重构逻辑资源结构和不同布局算法具有较好的适应性,将 DRP 分为两级:一级执行布局决策,另一级完成任务配置和执行.因为各逻辑块上的任务并行执行,不同逻辑块上的配置和执行可以相互交叠,若将重构控制和布局控制在一级中实现,那么随着可重构逻辑块数量的增加和阵列的变化,重构控制会变得复杂化,很难适应不同的硬件结构.将动态可重构处理器抽象为如图2所示,布局决策和任务配置执行分为上下两级,两级之间相对独立,便于各自实现和管理.

模型中 Tile 是可重构单元的抽象,代表图1中动态可重构处理器重构资源部分的一个逻辑块. Tile 内局部配置数据存储器容量 M 、配置模式 M_{cfg} 以参数方式配置于 MC. Context 表示 Tile 上的配置数据,STC 实时监视 Tile 执行状态,完成配置数据下载、切换、配置、执行等控制,并将任务执行状态返回给 MC 模块.重构处理进程如图3所示.

布局决策模块 MC 实现对 Tile 的管理,根据各 Tile 执行状态和布局算法为任务选择一个执行位置,将该任务交予对应 Tile 执行.在图2模型中,

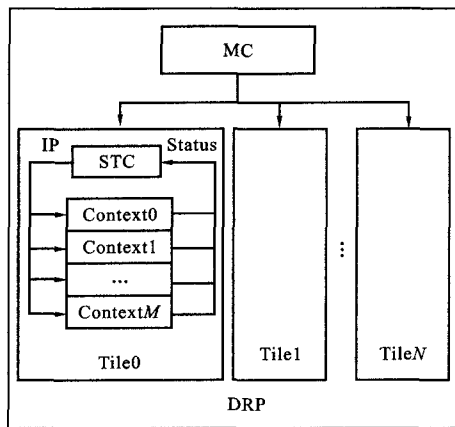


图2 DRP抽象模型

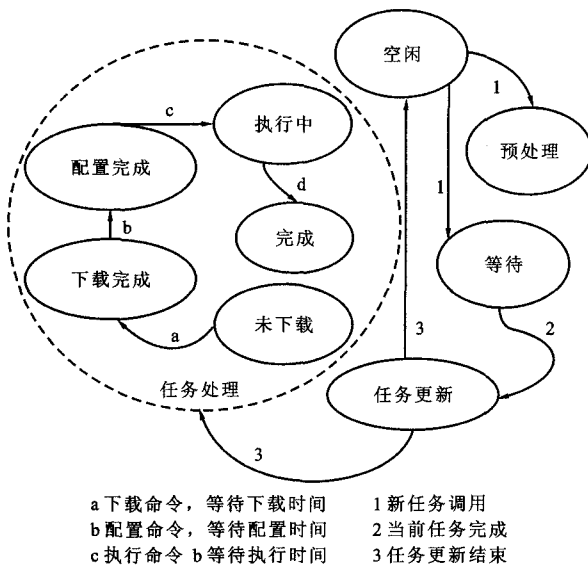


图3 Tile重构状态图

MC可以执行不同的布局算法, 将不同的布局算法抽象为参数 $A_{\text{placement}}$, 通过配置参数选择布局算法。为了在系统级设计空间探索中精确模拟硬件结构, 将可重构处理器的结构以第2节中定义的参数配置于MC中, 这些参数将作为布局选择的约束条件。

4 DRP的SystemC建模描述

用SystemC的事务级建模来仿真该系统模型。将模型划分为三个模块: 模块MC (Manager and Controller), 模块STC (State Transition Controller) 及模块Contexts。其中STC和Contexts模块实例化连接构成图2中的基本重构单元Tile。MC与STC间定义接口 $\text{DRP_MC2STC_TaskAllocation_if}$, 而STC通过 $\text{DRP_STC2Contexts_if}$ 接口与Contexts通信。

MC模块: 实现布局算法。为了能将DRP模型嵌入设计空间探索构架中, MC模块预留有和总线

互连的端口。MC模块管理DRP内部资源, 实时监视各Tile中功能执行情况, 根据布局算法将任务交给指定Tile执行。

STC模块: 作为重构处理控制单元, 处理MC分配的任务。以IP信号选择context的使用。STC模块内部包括三个并行执行的SC_THREAD进程:

$\text{pre_process}()$: 预处理进程。将新任务的配置数据下载到Tile内部, 为当前任务完成后配置新任务做准备。

$\text{task_update}()$: 任务更新进程。等当前任务结束后将新任务更新为当前任务, 等待下一个任务调用。

$\text{task_processing}()$: 任务处理进程。根据当前context的状态向Contexts模块发送context下载、配置、执行命令。

Contexts模块: 据STC模块的控制信号模拟动态重构。本模块包含多个context, 每个context表示一个硬件任务。context定义为类, 其数据成员TaskNUM、Dltime、Cnftime、Runtime、Status分别代表任务号、下载时间、配置时间、运行时间及状态, 以参数形式予以配置; 成员函数 $\text{context_action}()$ 模拟功能执行。context状态表征硬件任务的执行进度, 定义如下:

- NONEXISTENT: 未下载到Tile;
- EXISTENT: 已下载到Tile;
- CNFDONE: 配置完成;
- EXECUTING: 已激活, 执行中;
- DONE: 执行结束, 任务终止。

$\text{DRP_MC2STC_TaskAllocation_if}$ 接口: 接口内定义了 $\text{taskcall}(\dots)$ 和 $\text{get_status}(\dots)$ 两个方法, 它们在STC内部实现。MC通过调用 $\text{taskcall}()$ 向独立编址的Tile发送任务调用信号, 通过调用 $\text{get_status}()$ 方法获取Tile状态。Tile状态为IDLE和BUSY。

$\text{DRP_STC2Contexts_if}$ 接口: 接口内定义三个方法, 在Contexts模块内实现, 接口方法定义如下:

- $\text{contextcall}(\text{int ip}, \text{STC2Cnt_cmd command})$;
- $\text{statusmodify}(\text{int ip}, \text{context_status status})$;
- $\text{get_context_status}(\text{int ip})$ 。

STC通过调用 $\text{get_context_status}()$ 监视context状态。 $\text{statusmodify}()$ 接口方法用于状态修改,

主要在 STC 模块的预处理进程中被调用,保证存储于 tile 内部的 context 数量恒定,以模拟 tile 内部存储器中 context 的下载更新. contextcall() 执行重构等操作,操作命令由任务执行状态决定. STC 调用 context 的操作命令为: DOWNLOAD, CONFIGURATION 和 EXECUTE.

5 仿真

系统结构和布局算法是以参数化方式配置于本模型结构中,本文专注于该模型本身的重构模拟,不考虑具体布局算法和硬件结构. 在该系统模型中, MC 模块通过布局算法将来自 CPU 的任务调用分配给 Tile, Tile 执行任务处理. Tile 个数、阵列外形和配置模式是可以配置的,但是每个 Tile 处理任务调用的机制一样. 为了说明模型功能,以 MC 模块向 Tile0 分配任务为例进行仿真,验证 Tile 任务调用处理. 假定某个应用总共有 4 个硬件执行任务,配

置数据分别映射为 context,且单个任务配置执行所需可重构逻辑块为一个 Tile, Tile 内部存储器可存储两个 context. 初始化完成后预存 context0 和 context1 于 Tile 内部的存储器,并完成 context0 配置,当前任务指针指向 context0,其他 context 存储于配置数据存储器. 各 context 参数及初始状态如表 1 所示.

表 1 任务参数

上下文	任务号	下载时间/	配置时间/	运行时间/	初始状态
		ns	ns	ns	
context0	0	90	150	600	CNFDONE
context1	1	60	100	500	EXISTENT
context2	2	40	150	600	NONEXISTENT
context3	3	90	200	750	NONEXISTENT

用 IEEE Std 1666™ – 2005 SystemC 在 Microsoft Visual Studio 2008 平台仿真,设定时钟周期为 10ns,跟踪到图 4 所示的波形.

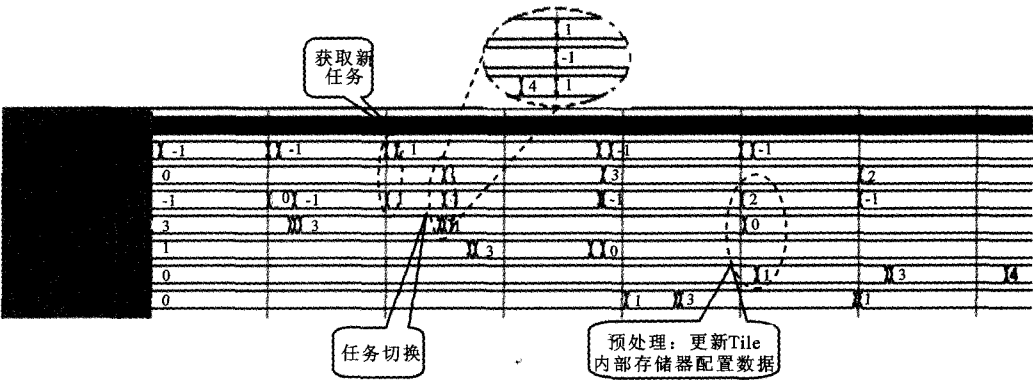


图 4 仿真波形

图中 task 为 MC 模块分配给 Tile0 的任务, current_ip 为当前正在处理的任务, next_ip 为 STC 新接收到的来自 MC 的任务. 在仿真启动后 $t=0$ 时刻 MC 向 Tile0 发送任务 0, 由于当前任务预先指向 context0 且已经配置完成, 直接开始执行.

在 $t=500$ ns 时刻 MC 再次调用任务 0, 延迟一个时钟周期(SystemC 的 Δ 延迟赋值机制)后 next_ip 获得该任务信息. 此时当前任务尚在执行中, 不能立即执行. 等上次调用结束, 即 context0 状态为 DONE 后, 将新的任务赋予当前任务指针 current_ip 并修改状态为 CNFDONE, 延迟一个时钟周期直接开始新一轮执行.

$t=1\ 000$ ns 时 MC 调用任务 1, 该情况为前一任务尚未结束收到非重复任务调用的情况. 任务 0 执行结束后切换任务, context0 状态修改为 EXISTENT. 由于初始化时任务 1 预存到 Tile 内, 直接开

始对 t_1 的配置、执行.
 $t=1\ 900$ ns 时 MC 发送任务 3 调用, 此时上一任务调用结束. 由于任务 3 的配置数据 context3 存储于外部配置数据存储器, 在获得该任务调用后将 context3 下载到 Tile 内, context1 被换出. 随后进行配置、执行.
 $t=2\ 500$ ns 时刻 Tile 收到 MC 任务 2 调用, 当前任务 3 尚未执行结束, 而任务 2 配置数据未下载, 执行预处理. 将未执行的 context0 换出, 下载 context2 到 Tile, 待任务 3 执行结束后切换任务, 配置、执行任务 2. 在 $t=3\ 630$ ns 时刻完成所有调用任务的执行.

通过仿真分析可知, 通过配置任务的参数, 模型可以根据当前任务状态和前后任务的关系, 模拟动态可重构处理器处理硬件任务调用——更新局部上下文存储器的配置数据, 即下载新任务配置数据到

闲置的上下文存储位置,为新任务的配置做准备;前一任务完成后自动切换到新的任务.由于重构控制和布局策略模块的分离实现,通过结构参数和布局算法参数配置不同的结构布局算法不会影响重构控制,可以快速建立不同应用的仿真模型.

6 结束语

针对动态可重构 SoC 设计空间探索构架中的动态可重构协处理器建模,本文提出布局决策和重构控制相分离的系统级模型,并用 SystemC 语言实现模型描述,模拟了对任务调用的重构处理.该模型中结构和布局算法采用参数化配置,通过调整参数可以模拟不同结构和布局算法的重构处理,具有较好的通用性和灵活性.可将该模型嵌入动态可重构 SoC 的设计空间探索构架,实现动态可重构处理器快速建模,有助于系统不同方案的快速分析和优化.

参考文献:

- [1] 孙康,沈海斌,王继民,等.参数化的可重构系统抽象模型[J].浙江大学学报:工学版,2008,42(8):1323—1329.
- [2] William W C Chu. Reconfigurable computing systems cost/benefit analysis model [D]. Waterloo: University of Waterloo, 2005.
- [3] 季爱明,沈海斌,严晓浪.可重构结构设计空间快速搜索方法[J].电子与信息学报,2006,28(9):1744—1747.
- [4] Kamana Sigdel, Mark Thompson, Carlo Galuzzi, et al. rSesame — A generic system-level runtime simulation framework for reconfigurable architectures [C]// Proceedings of The 2009 International Conference on Field-Programmable Technology. Sydney: IEEE, 2009: 460—464.
- [5] Wu Kehuai, Jan Madsen. COSMOS: a system-level modelling and simulation framework for coprocessor-coupled reconfigurable systems [C]// Proceedings of 2007 International Conference on Embedded Computer Systems; Architectures, Modeling and Simulation. Samos: IEEE, 2007: 128—136.
- [6] HSIUNG Pao-ann, LIN Chaosheng, LIAO Chih-feng. Perfecto: A systemc-based design-space exploration framework for dynamically reconfigurable architectures [J]. ACM Trans Reconfigurable Technol Syst, 2008, 1(3): 1—30.
- [7] Antti Pelkonen, Kostas Masselos, Miroslav Cupak. Systemk-level modeling of dynamically reconfigurable hardware with SystemC [C]// Proceedings of the 10th Reconfigurable Architectures Workshop, International Parallel and Distributed Processing Symposium. Nice: RAW, 2003: 174—181.
- [8] Yang Qu, Kari Tiensyrja, Kostas Masselos. System-level modeling of dynamically reconfigurable co-processors [J]. Lecture notes in computer science, 2004: 881—885.

作者简介:

王 鹏 男,(1985—),硕士研究生.研究方向为嵌入式系统设计、可重构计算等.

(上接第 85 页)

- [4] 肖飞,何书专,李丽,等.一种 0.18 μm 的高速 USB 2.0 收发器[J].微电子学与计算机,2009,26(12):96—99.
- [5] Lopez Delgadillo E, Garcia Andrade M A, Diaz Mendez J A. Automatic impedance control for chip-to-chip interconnections on Electronics [C]// 15th IEEE International Conference Circuits and Systems, 2008. Malta: IEEE, 2008: 332—335.
- [6] Compaq, Hewlett-Packard, Intel, et al. Universal serial bus specification, Revision 2.0 [M]. 2000: 119—194.
- [7] Koo Kyoung Hoi, Kyungkido, Seo Jinho. Impedance Control Circuit; United States, 2003/0030579 A1 [P]. 2003—02—13.
- [8] Rabaey Jan M, Anantha Chandrakasan, Borivoje Nikoli. 数字集成电路——电路、系统与设计 [M]. 北京:电子工业出版社, 2004.

作者简介:

潘 科 男,(1982—),硕士研究生.研究方向为超大规模集成电路设计.

何书专 男,(1974—),工程师.研究方向为集成电路设计技术.

李 丽 女,(1975—),副教授,硕士生导师.研究方向为超大规模集成电路设计.