一种基于层次平台 SoC 设计中的软硬件划分方法

李仲宇[1],吴海波[2],夏新军[2]
LI Zhong-yu[1],WU Hai-bo[2],XIA Xin-jun[2]

1.湖南科技大学 信息学院,湖南 湘潭 411201
2.湖南科技大学 计算机学院,湖南 湘潭 411201
1.School of Imformation Science,Hunan University of Science and Technology,Xiangtan,Hunan 411201,China
2.School of Computer Science,Hunan University of Science and Technology,Xiangtan,Hunan 411201,China
E-mail:xjxia@hnust.edu.cn

LI Zhong-yu,WU Hai-bo,XIA Xin-jun.Method for HW/SW partitioning in hierarchical platform-based SoC design. Computer Engineering and Applications,2007,43(7):125-128.

Abstract: Hardware/software partitioning is a key problem in SoC design.The reasonable result of partitioning greatly affects the chip in cost,performance,flexibility and so on.This paper brings forward a method based on genetic algorithm for hardware/software partitioning in hierarchical platform-based design.An example is given to validate the method.
Key words: hardware/software partitioning;hierarchical platform-based design;constrained task-flow graph;genetic algorithm

摘 要:软硬件划分是 SoC 设计中的一个关键问题,合理的划分结果对最终生成的芯片在成本、性能、可扩展性等方面有重要影响。提出了在基于层次平台的 SoC 设计中,采用遗传算法进行软硬件划分的方法,并通过实验验证了其在 SoC 设计中的可行性。
关键词:软硬件划分;基于层次平台的设计;约束任务流图;遗传算法
文章编号:1002-8331(2007)07-0125-04 **文献标识码**:A **中图分类号**:TP391.72

微电子技术的快速发展,推动集成电路产品进入百万门级的规模,带有一个或者几个微处理器核的 SoC 产品由于同时具有软件的灵活性和硬件的高性能,已经逐渐成为集成电路产品发展的主流。但是随着规模的增大,设计复杂度和设计难度也增加了,同时由于市场竞争压力加大,要求设计周期缩短。如果继续按照传统的方法进行设计,设计效率和需求之间的差距将继续增加,这种情况对 SoC 设计方法学产生了巨大的影响,基于平台设计和软硬件协同设计方法应运而生。

1 SoC 系统设计方法学

传统设计方法的开始是系统工程师用 C 或者 C++描述系统模型,验证系统的概念和算法的正确性。算法验证后,通过手工方法将部分 C/C++模型转化为 VHDL 或者 Verilog 描述的硬件实现部分,经过仿真和综合得到最终的系统芯片,但是这个方法存在很多问题。

1.1 软硬件协同设计

如图 1 所示为 SoC 的软硬件协同设计流程。

这种设计思想需要解决许多新问题:

(1)系统的描述方法。以往的硬件描述语言无法描述系统的软件部分,需要一个完善的系统级设计语言。

(2)如何将这一设计理论与传统理论相结合。

(3)最优性原则的确定。除了以往的速度、面积等硬件优化指标外,与软件相关的如代码长度、资源利用率、稳定性等指标也必须予以考虑。

(4)如何统一验证的方法,对这样的一个包含软件和硬件的系统进行功能验证。

(5)系统的功耗问题。

SoC 的功耗不仅包括静态功耗,由于软件运行而引起的动态功耗也占有相当大的比重,这种动态功耗只能通过软硬件的

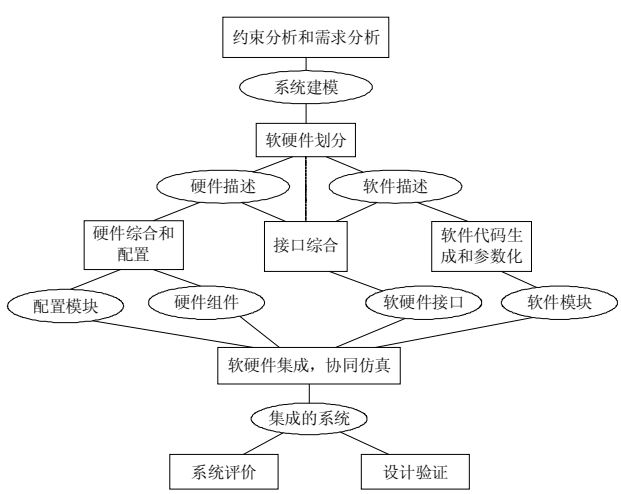


图 1 软硬件协同设计流程

基金项目:国家高技术研究发展计划(863)(the National High-Tech Research and Development Plan of China under Grant No.2002AA1Z1480);湖南省教育厅科技项目(No.05C183)。
作者简介:李仲宇(1972-),男,讲师;吴海波(1973-),男,讲师;夏新军(1974-),男,副教授,研究方向:VLSI 设计自动化。

联合运行才能知道。基于以上原因,软硬件协同设计技术研究了多年仍然没有很成熟的产品。

1.2 基于平台的设计

基于平台的设计是目前面向 SoC 设计中一种比较流行的方法,它是由加州大学伯克利分校 UCB(University of California,Berkeley)的 Sangiovanni 教授首先提出。基于平台设计是一种“中间相遇”的设计方法(Meeting-in-the-middle)[1]。

图 2 显示了基于平台的设计流程,从图中可以看出,已有的 SoC 系统设计方法试图从系统描述通过软硬件划分、行为与系统结构建模、功能-结构映射、软硬件求精、软硬件集成等过程直接得到 SoC 系统的软件目标代码以及可综合的 RTL 硬件系统结构。在这些过程中,不断的进行性能分析与软硬件协同功能验证,试图保证系统功能与性能满足要求。

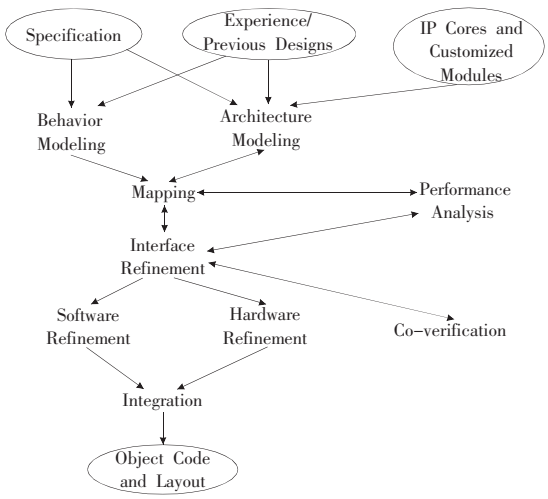


图 2 基于平台的设计流程

这些方法存在的问题主要有:(1)系统描述与 RTL 硬件系统结构之间的跨度很大,不能在真正意义上实现设计要素的正交,实际设计中两者之间难以直接映射。(2)基于平台的设计中,系统结构建模过程就确定了 SoC 系统的硬件结构,只能通过软件功能体现设计差异,损失了设计灵活性。(3)这些方法没有将系统性能约束嵌入到设计模型中,并使性能约束随设计过程向下传播,只是通过性能分析来被动地验证某设计步骤是否满足性能要求,在性能不满足时就可能出现设计步骤的较大反复。

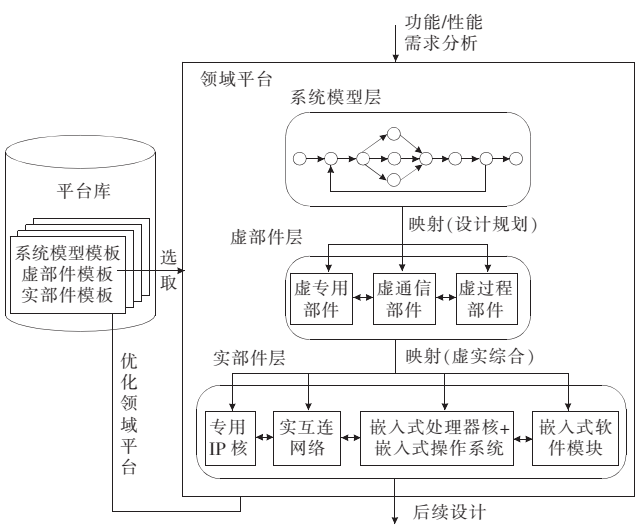


图 3 Hi-PBD 方法总体结构

1.3 基于层次平台的设计

图 3 显示了基于层次平台的 SoC 系统设计方法总体结构。平台库中有为不同 SoC 应用领域定义的通用平台,各领域通用平台不仅包含系统模型、虚部件和实部件 3 个层次的设计模板以及两次映射的结果,还包含领域虚部件库、实部件库、系统仿真测试库等可以为后来的设计提供局部定制与验证,从而在确保系统重用的同时又不失灵活性。

系统模型层在算法级描述系统的功能与性能。已有的系统描述模型(如离散事件、有限状态机、数据流/控制流、Petri 网等)等都只能描述系统的功能行为而不具备性能描述能力,因此采用约束任务流图(Constrained Task-flow Graph,CTG)来进行系统建模[2],图 4 给出了一个完整的 CTG 模型例子。图中用灰色标识出来的任务为辅助任务,其它的任务都是普通任务。辅助任务主要用来定义任务的执行控制关系,反映出任务的并行、分支以及循环(反馈)等许多控制结构。CTG 模型是一个可变粒度的算法级描述模型,该模型通过定义约束属性描述任务的性能约束,通过算法映射描述任务的行为。

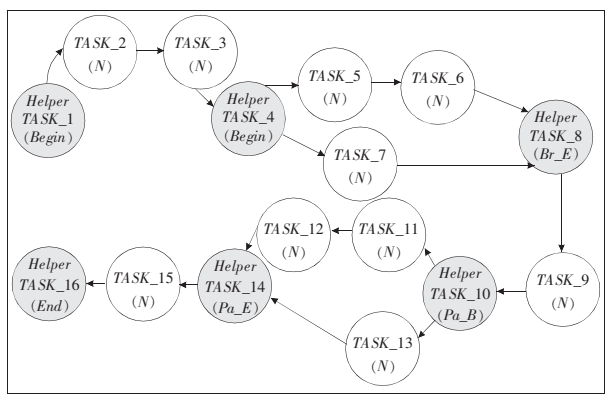


图 4 约束任务流图示例

虚部件层是 SoC 系统设计的中间层,包括 3 种虚部件:虚专用部件、虚过程部件和虚通信部件。其中,虚专用部件是一类硬 IP 核的抽象(微处理器核与 DSP 除外),因此不需包含过多的硬件实现细节;虚过程部件是拟映射到嵌入式处理器核(MPU、DSP 等)上运行的软件算法过程;虚通信部件定义了虚专用部件、虚过程部件之间的互连通信行为。在系统任务粒度规划合理的情况下,约束任务流图模型中的叶任务可直接映射到虚部件。

实部件层是实际的 SoC 软硬件系统层,包括专用 IP 核的结构与功能描述、嵌入式处理器核的指令集结构和资源配置描述、专用 IP 核与嵌入式处理器核之间的互连通信网络结构和时序描述、在嵌入式处理器核上运行的嵌入式软件模块等。

2 软硬件划分

2.1 软硬件划分的相关研究

软硬件划分有两种方法,一种是面向软件的方法,在开始将所有的实现映射到软件实现,然后通过评价,将性能关键任务节点移动到硬件实现,直到满足时间约束;一种是面向硬件的方法,在开始将所有的实现映射到硬件,然后逐渐将节点移动到软件实现。面向软件的代表是 Henkel 在 COSYMA 系统中使用的方法[3],该方法采用了模拟退火算法实现。面向硬件的代表方法是 Gupta 等研究者在 VULCAN 所使用的方法[4],该算法是一个贪婪(Greedy)算法,因此只能求得局部最优解。

Berkeley 大学 Kalavade 等提出的 GCLP 方法是启发式算法中的一个典型[5],采用了类似列表调度(List Scheduling)的方法进行性能的评价。Kalavade 又以 GCLP 算法为基础,扩展了简单情况的软硬件划分,提出任务的实现有算法级、转换级和资源级等 3 个级别的选择层次,提出了 MIBS 算法[6],解决了扩展的划分问题。

迭代式算法的研究也比较广泛,Washington 大学的 Ta-Cheng Lin 等人提出了一个 PSGA 的算法[7],这是一个基于遗传算法的划分算法。算法的输入是 VHDL 描述,通过高级综合工具转化为 CDFG,将 CDFG 映射到软件和硬件上。

Karam S.Chatha 等人的 MAGELLAN 算法是一个可以针对多处理器划分的算法[8],这个算法使用了一个层次化的方法,将层次化的控制数据任务流图映射到层次化的体系结构模板上。

2.2 软硬件划分的目标

软硬件划分就是为系统的每个功能确定它的实现方式,实际是一个多元优化的问题,自动的软硬件划分是一个 NP-hard 问题。系统功能划分到软件或者硬件影响全局的开销和性能,系统功能在硬件实现可以通过硬件加速和并发的执行提高系统的性能,但是增加了制造 ASIC 的成本,灵活性也比较小;系统功能在软件实现可以有比较好的灵活性,但是对系统性能有影响。

2.3 软硬件划分的形式化定义

定义 H 是系统功能中实现在硬件上的功能的集合,S 是系统功能中实现在软件上功能的集合,O 是全部系统功能的集合,且 $H\subset O$,$S\subset O$,$H\cup S=O$,$H\cap S=\phi$。

$Hsize(H)$是功能实现在硬件上所需要的面积,就是晶体管的数量。

$Perfomance(G)$:对于一个给定的划分$\{H,S\}$,全部的执行时间。

$Cons=(C1\cdots Cm)$为系统的性能约束集合,这里 $Cj=\{G, timecon\}$表示在功能组合 G 中所有功能允许的最大执行时间为 $timecon$,其中,$G\subset O$。

一个划分满足性能约束就是它的 $Performance(Cj.G)\leqslant Cj.timecon$,对于所有的 $j=1\cdots m$。因此给定功能集合 O 和性能约束集合 $Cons$,软硬件划分问题就是找到一个性能满足约束而面积 $Hsize(H)$最小的划分$\{H,S\}$。

3 基于层次平台的软硬件划分

遗传算法具有搜索解空间大,收敛速度快,全局搜索,不容易陷入局部最小等优点,在 NP 问题中应用比较广泛。由于遗传算法的这些优点,适合用于解决软硬件划分问题。基于层次平台 SoC 设计中采用遗传算法进行软硬件划分的流程主要包括如下步骤:

(1)基于约束规划结果生成初始的种群;

(2)根据初始划分结果确定最大进化代数;

(3)种群变异和交叉生成中间代;

(4)中间代和旧一代个体择优进入下一代;

(5)根据新一代调整适应值;

(6)重复过程(3)、(4)、(5),直到进化达到最大进化代数。

3.1 任务编码

遗传算法求解不能直接作用在问题的解空间上,首先需要对问题的解进行编码。系统在虚部件库搜索得到的适合某个任务的专用虚部件可能不止一个,因此需要设计者对每个任务实现选择具体的专用部件,因此本文的算法采用整数向量编码,每个编码是固定长度的整数向量。整数向量 $\boldsymbol{T}=(t_1,t_2\ \cdots,t_N)$,编码的长度 N 等于任务流图中的任务数量,编码向量的元素与任务流图的任务一一对应,每个元素的取值范围为[0, MAX_*vc_num*],值为 0 说明使用软件实现方式,不是 0 说明是采用对应的一个专用部件实现,MAX_*vc_num* 是任务中有最大对应专用部件的数量。比如编码(0,1,0,2,0,0,4),可以确定任务 1、3、5、6 是采用软件方式实现,任务 1 采用它的实现方式 1,任务 4 采用它的实现方式 2,任务 7 采用它的实现方式 4。

3.2 初始种群的生成

在基于层次平台 SoC 设计中,经过约束任务规划以后,生成初始种群的工作就有了大量的先验信息,不需要采用随机生成初始种群的方法,而是可以根据约束任务规划的结果生成初始种群,这将加快算法的收敛。计算量比较大、对处理器性能要求比较高的任务属于性能关键任务,软硬件划分时优先将性能关键任务划分到硬件。任务规划根据性能仿真的结果确定性能关键任务,因此生成初始种群时,对性能关键任务选择硬件实现方式。如果任务满足约束,说明初步的规划是足够的,探索将硬件实现的任务移动到软件实现;如果不满足系统约束,就主要考虑将划分到软件运行的任务移动到硬件实现。移动任务实现方式时主要考虑两个方面的影响:遗传算法搜索方式的影响和通信的影响。

$$P_{si}=1-f_{gti}(t_x,t_{constrain})-f_{comm}(t_i,t_{is})$$

$$P_{ij}=\frac{f_{gti}(t_x,t_{constrain})+f_{comm}(t_i,t_{is})}{k}$$

$$g(t_x,t_{constrain})=\left|\frac{t_x-t_{constrain}}{t_x+t_{constrain}}\right|$$

$$f_{gti}(t_x,t_{constrain})=\begin{cases}0.05 & g(t_x,t_{constrain})\in[0,0.05)\\ \alpha g(t_x,t_{constrain}) & g(t_x,t_{constrain})\in[0.05,0.33)\\ 0.33 & g(t_x,t_{constrain})\in[0.33,\infty)\end{cases}$$

$$f_{comm}(t_i,t_{is})=\begin{cases}0.05 & g_{comm}(t_i,t_{is})\in(0,0.05)\\ \beta g_{comm}(t_i,t_{is}) & g_{comm}(t_i,t_{is})\in[0.05,0.33)\\ 0.33\beta & g_{comm}(t_i,t_{is})\in[0.33,\infty)\end{cases}$$

$$g_{comm}(t_i,t_{is})=\left|\frac{t_i-t_{is}}{t_i+t_{is}}\right|$$

$$\beta=\begin{cases}1.5 & f_{gti}\in[0.05,0.1)\\ 1.5+15(f_{gti}-0.05) & f_{gti}\in[0.1,0.25)\\ 3 & f_{gti}\in[0.25,0.33)\end{cases}$$

P_{si} 表示任务选择在软件上实现的概率;假设任务 T_i 有 k_i 个对应的专用硬件实现方式,选择在硬件 j 上实现的概率为 P_{ij};$f_{gti}(t_x,t_{constrain})$表示遗传算法搜索方式影响下的概率,体现了任务规划的准确程度;$g(t_x,t_{constrain})$表示初始划分个体 X 的性能和性能约束的差距,t_x 表示个体 X 的执行时间,$t_{constrain}$ 表示性能的约束时间,α 是针对相对差距 $g(t_x,t_{constrain})$提出的参数,确保 $\alpha g(t_x,t_{constrain})$在 0.05 到 0.33 之间变化;$f_{comm}(t_i,t_{is})$表示通信影响下选择硬件的概率,体现了每个任务初始约束分配的准确程度;对于任务 T_i,定义通信对于约束任务规划结果影响因素 $g_{comm}(t_i,t_s)$,其中 t_i 是任务 T_i 在计算了通信时间之后的软件实际执行时间,t_{is} 是任务 T_i 经过约束分配之后分配的时间。

3.3 算法终止条件的设计

$$N_{MAX\_evolu}=150\times\left(\log\left(\frac{N_{numtask}}{10}\right)+1\right)\times\left(\delta\frac{1}{g(t_x,t_{constrain})+\lambda}\right)$$

其中 N_{MAX_evalu} 表示进化代数;δ 和 λ 是两个调整参数,根据工程实际需要调整。

3.4 适应值的设计

$$fitness=\alpha_f\times e^{-\left|\frac{t_Y-t_{constrain}}{t_Y+t_{constrain}}\right|}$$

$$f(N_Y,N_N)=\left|\frac{N_Y-N_N}{N_Y+N_N}\right|$$

$$\alpha_f=\begin{cases}1 & t_Y>t_{constrain} \quad N_Y>N_N\\ 1+\beta_f\times(e^{f(N_Y,N_N)}-1) & t_Y>t_{constrain}\end{cases}$$

$$\alpha_f=\begin{cases}1+\beta_f\times(e^{f(N_Y,N_N)}-1) & t_Y>t_{constrain} \quad N_Y<N_N\\ 1 & t_Y>t_{constrain}\end{cases}$$

t_y 是个体 Y 的完成时间,N_Y 为种群中满足约束的个体的数量,N_N 为种群中不满足约束的个体的数量;α_f 是一个参数,调整适应值在一定范围内变化;β_f 需要根据工程实际需要确定。

3.5 交叉和变异的设计

本文的算法采用整数编码,算法中需要根据每个任务对应的实现方式的数量进行变异。例如,一个任务有两种专用虚部件的实现方式,再加上软件实现,这个任务的映射选择有 3 种。种群个体变异时,染色体中代表这个任务的基因对每种实现方式都有相同的选择概率。算法交叉时,采用多点交叉,交叉点的个数与任务数成正比。如果两个交叉点的实现数量不同,一个个体的交叉点的值超过相应个体交叉点的最大值,就采用随机的方式交叉。

4 实验结果

为了验证本文的算法,使用了一个有 40 个任务的例子进行实验,相应的参数确定如表 1。

表 1 采用遗传算法进行软硬件划分的参数

参数	名称	值
任务数量	N_{num_task}	40
种群规模	N	80
变异概率	P_m	0.000 2
交叉概率	P_c	0.9
初始种群参数	α	1
最大进化代数参数	λ	0.6
	δ	2
适应值调整参数	β_f	0.115

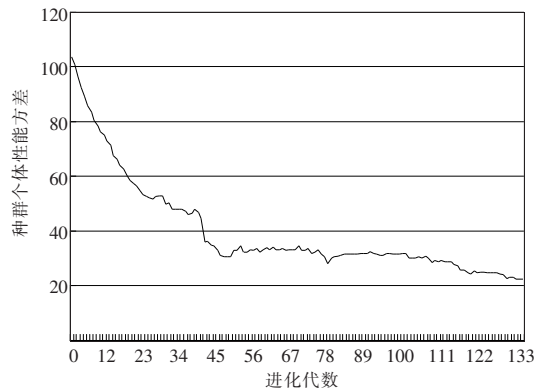
图 5(a)显示了种群个体性能方差随进化代数变化的曲线图,随着进化代数的增大,种群个体性能方差逐渐稳定在 20 左右。

图 5(b)显示了满足约束个体和不满足约束个体所占比例随着进化代数变化的曲线,随着进化代数的增大,满足约束个体和不满足约束个体所占比例都趋近于 50%。

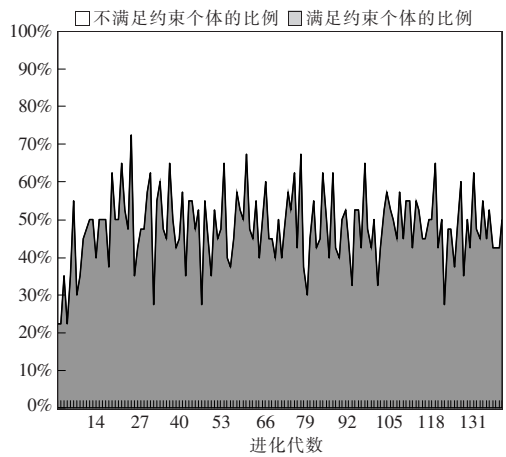
5 结束语

基于层次平台设计方法,课题组设计实现了面向数字流媒体应用的软硬件协同设计环境 SoC-CDE,系统设计中采用了本文所述的基于遗传算法的软硬件划分方法。应用 SoC-CDE 设计环境,进行了 MP3 播放器和 CDMA 无线通信芯片的设计,经测试和验证,最终生成的芯片基本达到了应用需求分析中所确定的面积、功耗、延迟等性能约束指标。

(收稿日期:2006 年 5 月)



(a)种群个体性能方差随进化代数变化的曲线图



(b)满足约束个体和不满足约束个体所占比例随进化代数变化的曲线

图 5 实验结果

参考文献:

[1] Sangiovanni V A.Defining platform-based design [EB/OL].(2002-03-05).http://www.eedesign.com/features/exclusive/OEG20020204-S0062.

[2] Xiong Zhi-hui,Li Si-kun.A new SoC system modeling method[C]//The 8th Inter Conf on CAD/Graphics,Macau,2003:157-161.

[3] Ernst R.The COSYMA environment for hardware-software cosynthesis of small embedded systems[J].Microprocessors and Microsystems,1996,20(3).

[4] Gupta R K.Special issue on partitioning methods for embedded systems [J].Design Automation for Embedded Systems,1997,2(2):123-261.

[5] Kalavade,Lee E A.A global criticality/local phase driven algorithm for the constrained hardware/software partitioning problem[C]//Proceedings of the 3rd International Workshop on Hardware/Software Co-design,Grenoble,France,1994-09:42-48.

[6] Kalavade,Lee E A.The extended partitioning problem:hardware/software mapping,scheduling,and implementation-bin selection[J].Journal of Design Automation for Embedded Systems,1997,2(2):126 -163.

[7] Lin Ta-cheng,Sait S M,Cyre W R.Buffer size driven partitioning for HW/SW co-design[C]//IEEE International Conference on Computer Design,ICCD'98,Austin,USA,1998:596-601.

[8] Chatha K S,Vemuri R,MAGELLAN:multiway hardware-software partitioning and scheduling for latency minimization of hierarchical control dataflow task graphs[C]//Proceedings of CODES 2001,Copenhagen,Denmark,2001:42-47.