

一种变电压多核处理器上的有效节能方法

王颖锋 刘志镜

(西安电子科技大学计算机学院 西安 710071)

摘要 在考虑转换开销和核间通信开销的情况下,针对变电压多核处理器上存在时间约束的含依赖任务的应用,提出了一种用于实时多核嵌入式系统开销感知的综合节能方法。该方法用RDAG算法将任务独立化后,将动态电源管理、自适应衬底偏置和动态电压调节有效地结合起来。分别用几个随机任务图和代表实际应用的任務图做2,3处理器核上的模拟实验,结果表明提出的方法优于原方法。

关键词 能量优化,动态功耗,泄漏功耗,多核

中图分类号 TP316 **文献标识码** A

Efficient Method of Energy Saving on Variable Voltage Multi-core Processor

WANG Ying-feng LIU Zhi-jing

(School of Computer Science and Technology, Xidian University, Xi'an 710071, China)

Abstract For applications including dependent tasks with timing constraint in the case of considering transition overhead and inter-core communication overhead on variable voltage multi-core processors, an overhead-conscious synthesis method of energy optimization was proposed for real-time multi-core embedded systems. The method effectively combined dynamic power management with adaptive body bias and dynamic voltage scaling based on independent tasks produced by the RDAG algorithm. Simulation experiments were conducted with several random task graphs and task graphs representing real applications on 2,3 cores, respectively. The experimental results show that the proposed method gets advantage over the original method.

Keywords Energy optimization, Dynamic power consumption, Leakage power consumption, Multi-core

1 引言

随着嵌入式系统的应用发展,能量消耗问题已经成为嵌入式系统设计的关注重点。功耗问题不是一个单一问题,它影响到系统的安全性、电池供电系统的电池寿命、系统的散热代价等。高功耗严重制约着嵌入式系统的应用发展。直到最近几年,动态功耗一直是功耗的主要来源^[1]。然而,当集成工艺进入到纳米时代,特别是亚纳米时代,动态功耗和泄漏功耗的比例发生了变化。杨等指出,对于70nm的处理器,泄漏功耗比动态功耗小;对于50nm的处理器,预测泄漏功耗将与动态功耗相当;而对于35nm的处理器,泄漏功耗比动态功耗大^[2]。由此可见,随着集成工艺的发展,泄漏功耗已经不容忽视。

为了降低系统级能量,许多研究者提出了各种任务调度算法。文献[3,4]分别指出了如何降低在单处理机环境下调度依赖任务和独立任务的系统能量。文献[5,6]提出了如何节约在多处理机环境下调度独立任务所消耗的系统能量。文献[7,8]提出了在多处理机环境下如何降低有依赖关系的任务调度所导致的系统能量消耗。事实上,将依赖任务转化为独立任务,更能节约系统能量。文献[9]提出了一种在多核系

统上将依赖任务转换为独立任务后,在硬实时约束下,以能量最小化为目的的任务调度算法。

然而,上述研究没有考虑到如何降低泄漏能量。本文以纳米级处理器为研究对象,在考虑转换开销和核间通信开销的情况下,针对变电压多核处理器上存在时间约束的含有依赖任务的应用的能量优化问题,建立了包含动态能量和泄漏能量的能量模型。然后提出了一个两阶段方法来解决多核嵌入式系统能量最小化的问题。首先,利用RDAG算法将任务独立化,然后提出一个改进的弹簧算法将动态电源管理、自适应衬底偏置和动态电压调节有效地结合起来。

2 能量模型和性能模型

本文以MOSFET为对象建立能量模型。在MOSFET电路中,动态功耗和泄漏功耗是总功耗的主要组成部分。本文的能量公式是在总结文献[10,11]中动态功耗和静态功耗基础上得出的,动态功耗可表示为

$$P_{AC} = C_{eff} V_{dd}^2 f \quad (1)$$

式中, C_{eff} 是每周平均开关电容, V_{dd} 是电源电压, f 是时钟频率。而静态功耗可表示为

$$P_{DC} = V_{dd} K_3 e^{K_4 V_{dd}} e^{K_5 V_{bs}} + |V_{bs}| I_j \quad (2)$$

到稿日期:2009-10-20 返修日期:2010-01-20 本文受陕西省“13115”科技创新工程重大专项(批准号:2007ZDKG-57)资助。

王颖锋(1976—),女,博士生,主要研究方向为调度算法、实时系统的功耗压缩及容错技术、高性能计算、并行计算等,E-mail: wyf96126@126.com;刘志镜(1957—),男,教授,主要研究方向网络多媒体技术、视觉计算、虚拟现实技术和数据挖掘等。

式中, V_{bs} 是体偏置电压, I_j 是源漏区和体区间的 $p-n$ 结反向偏置电流之和。 K_3-K_5 是常设备参数。由于调节电源电压和体偏置电压对操作频率有影响, 因此操作频率

$$f = \frac{((1+K_1)V_{dd} + K_2V_{bs} - V_{th1})^\alpha}{L_d K_6} \quad (3)$$

式中, K_1, K_2, K_6 和 V_{th1} 是与电路相关的常数, α 反映速度饱和度。对于给定的频率, 优化的体偏置电压和供应电压可以由式(4)和式(5)得出^[11]

$$\frac{\partial E_{cv}}{\partial V_{bs}} = \{L_g K_3 f^{-1} (k_1 V_{bs} + k_2) e^{k_3 V_{bs} + k_4} - I_j L_g f^{-1} + 2C_{eff} (k_5 V_{bs} + k_6)\} \quad (4)$$

$$V_{dd} = (L_g K_6 f - K_2 V_{bs} + V_{th1}) / (1 + K_1) \quad (5)$$

式中, 变量 $k_1 \sim k_6$ 是从其它处理器常数中得出的常数, L_g 是电路中逻辑门电路个数。由于电源电压和体偏置电压转换会导致一定的电压转换时间和能量, 当电压从 V_{ddk} 转换到 V_{ddj} , 同时体偏置电压从 V_{bsk} 转换到 V_{bsj} 所对应的电压转换能量 $\epsilon_{k,j}$ 和时间延迟 $\delta_{k,j}$ 分别为

$$\epsilon_{k,j} = C_r |V_{ddk} - V_{ddj}|^2 + C_s |V_{bsk} - V_{bsj}|^2 \quad (6)$$

$$\delta_{k,j} = \max(p_{Vdd} |V_{ddk} - V_{ddj}|, p_{Vbs} |V_{bsk} - V_{bsj}|) \quad (7)$$

式中, C_r 是电源导轨电容, C_s 是基底和阱电容之和。 p_{Vdd} 和 p_{Vbs} 分别为时间斜率和电压斜率。

有 N_u 个周期数的任务 u 以频率 f 执行, 其执行时间可以表示为

$$t_u = N_u / f \quad (8)$$

除了动态能量开销、泄漏能量开销和电压转换能量外, 本文还考虑了通信能量消耗、休眠能量消耗、从活动状态到休眠状态所引起的状态转换能量开销, 因此系统的总能量可以表示为

$$E_{total} = E_{AC} + E_{DC} + E_{sch} + E_{com} + E_{sleep} + E_{trans} \quad (9)$$

式中, E_{AC} 代表动态能量, E_{DC} 代表泄漏能量, E_{sch} 代表电压转换能量, E_{com} 代表通讯能量, E_{sleep} 代表处理核处于休眠状态所消耗的能量, E_{trans} 代表从运行状态转换到休眠状态所消耗的能量。

3 任务映射和电压调节算法

3.1 RDAG 算法

依据重定时原理^[12], 文献[9]提出了一种将依赖任务转换为独立任务的 RDAG 算法。该算法的思想是首先给每个节点的初始重定时值赋以零。然后将所有的叶节点放入队列, 在放入的过程中每次尾指针均指向新进入的叶节点。在队列不为空的情况下, 将队列中的每个节点依次出队, 对于每个出队节点, 判断该节点的重定时值加 1 与其父节点的重定时值哪个最大, 取其中最大值作为该出队节点的重定时值。如果其父节点不是尾节点, 则将其父节点入队, 并将尾指针指向其父节点。如此反复, 就可以获得任务图中所有节点的重定时值。本文利用了 RDAG 算法, 将依赖任务转换为独立任务, 然后应用提出的多任务映射和电压调节 (MTMVS) 算法进行任务映射和电压调节。

3.2 MTMVS 算法

在介绍 MTMVS 算法之前, 首先介绍文中用到的一个概念: 模式。其具体定义如下。

定义 1 电源电压、体偏置电压和工作频率是一一对应的关系, 将每组电源电压、体偏置电压和工作频率称为一个电

压/频率模式。

MTMVS 算法是将任务映射和电压调节联合起来减少能量消耗的一种方法, 该算法具体描述如下。

输入: 独立化后的任务图 G , 其中包含 n 个独立任务, 任务集截止时间 D , 处理器芯片上核数 K , 电源电压、体偏置电压、工作频率等信息

输出: 一个有效的电压分配和任务映射以及总能量

1) 将电源电压、体偏置电压、工作频率按降序排列;
2) 安排每个任务最低电源电压、体偏置电压和频率, 即最低模式, 获得任务的初始执行时间;

3) 用 4/3 近似算法^[13]取得初始任务映射 M 和调度长度 L ;

4) if ($L \leq D$)

5) 这是一个可行映射, goto 27;

6) else

7) goto 9;

8) end if

9) While(true)

10) if ($L > D$)

11) 选择调度长度最长的处理核, 在考虑转换和通信开销的情况下, 用二分查找算法查找剩余模式, 将该核上所有任务升高到其查找获得的模式, 寻找能量增加最少的任务;

12) 将有最小 ΔE 的任务的处理时间作为比对象, 将处理时间较之小的任务以及该任务用 4/3 近似算法进行部分重映射;

13) 将各个核上的任务按照从高到低的模式排序, 刷新 L ;

14) else

15) 选择调度长度最短的处理核, 将其调度长度记为 $L_{shortest}$, 在考虑电压转换和通信开销的情况下, 将该核上所有任务降低一个模式, 寻找处理时间增加最小的任务, 将其降低一个模式后增加的时间记为 Δt ;

16) if ($L_{shortest} + \Delta t \leq D$)

17) $L_{shortest} = L_{shortest} + \Delta t$;

18) if ($L_{shortest} > L$)

19) $L \leftarrow L_{shortest}$;

20) end if

21) 接受这次变化, 将该处理核上的任务按照从高到低的模式排序, 刷新该处理核调度长度;

22) else

23) 不接受这次变化, 此次变换前的任务映射和电压调节为可行任务映射和电压调节, break;

24) end if

25) end if

26) end while

27) 检查每个处理核是否存在空闲时间;

28) 将存在空闲时间并且空闲时间大于从运行状态转换到休眠状态所消耗的时间的处理核, 转入休眠状态;

29) 计算系统的能量消耗。

MTMVS 是 SpringS 的改进算法, 较 SpringS 更能降低能量消耗。首先将电源电压、体偏置电压和频率按照从高到低的顺序排序, 以便在电压调节进行过程中二分查找参考模式。然后安排每个任务以最低电源电压、最低体偏置电压和频率, 用 4/3 近似算法获得最初的调度和调度长度, 这不同于 SpringS 算法, SpringS 算法是安排给每个任务以最低电源电压, 没考虑体偏置电压。MTMVS 算法在每次模式变换后, 都使各个核上的任务保持从高到低的模式排列, 这样减少了模式转换带来的能量开销。而 SpringS 算法则可能会引进较多的

模式变换能量。例如,一个核上有一个任务,运行该任务时处理核的模式低于前后两个任务运行时处理核所处的模式,并且运行前后两个任务,处理核所处的模式相同,在这种情况下, SpringS 算法就要进行两次模式转换,计算两次模式转换的能量开销。而 MTMVS 算法是将任务按照模式从高到低顺序进行排序,因此只进行一次模式转换,计算一次模式转换的能量开销。

4 实验以及结果讨论

4.1 实验建立

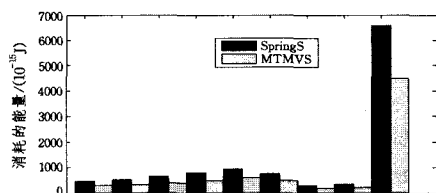
本文在 matlab2007 运行环境下进行仿真实验,对算法进行验证。实验所用的能量模型为 70nm 的能量模型,具体能量计算公式为第 2 节的式(9)。实验中使用真实应用任务集和用随机任务生成器产生的任务集进行测试。共用了 7 组随机任务集,编号为 G1-G7,节点个数范围为 17-65,执行周期数满足区间[2,6]的均匀分布。真实应用任务集为离散余弦变换(DCT)和稀疏矩阵求解器(SMS)。实验中将每组任务集以 7 个等间距时间限制进行测试。每个任务集的初始时间限制为用 4/3 近似算法获得的调度长度的近似数。测试任务集的特征见表 1。所模拟的处理器有个频率,分别为 7.8 GHz, 10.4GHz, 13GHz 和 15.6GHz。每个频率所对应的优化供应电压和体偏置电压可由式(4)和式(5)得出。在只应用动态电压调节的情况下,处理核的体偏置电压 $V_{bs}=0$ 。

在给定时间限制内将 MTMVS 算法与 SpringS 算法进行比较。图 1(a)和(b)给出了 MTMVS 算法与 SpringS 算法在 2,3 核的情况下消耗的平均能量比较。

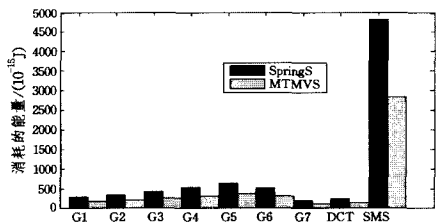
4.2 实验结果讨论

表 1 任务集特征

测试集	节点个数	边数	时间范围(ns)
G1	26	47	4.8~6
G2	34	57	5~6
G3	43	71	5.78~6.78
G4	54	83	6.67~8.47
G5	65	94	8.34~10.74
G6	51	67	6.42~7.62
G7	17	23	2.50~3.10
DCT	17	52	3.21~5.01
SMS	96	96	63.70~75.70



(a) 2 个处理核情况下的平均消耗能量比较



(b) 3 个处理核情况下的平均消耗能量比较

图 1 处理核平均消耗能量比较

从图 1(a)和(b)可以看出, MTMVS 算法下系统所消耗的能量明显低于 SpringS 算法下系统所消耗的能量。产生这

样的结果,原因有两个:一是因为 SpringS 只应用了 DVS 技术,导致泄露电流增长,在压缩动态能量的同时,导致泄漏能量占主导地位。而 MTMVS 算法同时使用了 DVS 和 ABB 技术,因此可以同时降低动态能量和泄漏能量;二是因为 MTMVS 算法采用了重排序策略,降低了模式转换的能量,而 SpringS 算法并没有采用这种策略。

结束语 本文针对降低多核处理器能量消耗的问题,提出一个两阶段解决办法。首先利用 RDAG 算法将依赖任务转换为独立任务,然后应用 MTMVS 算法进行任务映射和电压调节。仿真实验结果证明, MTMVS 算法较 SpringS 算法有效地降低了系统的能量。

参考文献

- [1] 张骏,樊晓姬,刘松鹤. 多核多线程处理器的低功耗设计技术研究[J]. 计算机科学, 2007, 34(10): 301-305
- [2] Yan L, Luo J, Jha N K. Joint dynamic voltage scaling and adaptive body biasing for heterogeneous distributed real-time embedded systems[J]. IEEE Trans. on CAD of Integrated Circuits and Systems, 2005, 24(7): 1030-1041
- [3] 刘惠,陈平,杜军朝. 可变电电压处理器的最优动态电压选择算法[J]. 西安电子科技大学学报, 2009, 36(3): 486-490, 540
- [4] 韩建军,刘同涛,李庆华,等. 基于任务同步及节能的单机系统实时动态调度算法[J]. 高技术通讯, 2008, 18(11): 1180-1186
- [5] Aydin H, Yang Q. Energy-aware partitioning for multiprocessor real-time systems[C]// IEEE Proceedings of the 17th International Parallel and Distributed Processing Symposium, Nice, France, 2003: 113-121
- [6] Luo J, Jha N K. Power-efficient Scheduling for Heterogeneous Distributed Real-time Embedded Systems[J]. IEEE Transaction on Computer-Aided Design of Integrated Circuits and System, 2007, 26(6): 1161-1170
- [7] Zhang Y, Hu X S, Chen D Z. Task scheduling and voltage selection for energy minimization[C]// Proceedings of the 39th Annual Design Automation Conference, New Orleans, Louisiana, 2002: 183-188
- [8] Zhu D, Melhem R, Childers B R. Scheduling with Dynamic Voltage/Speed Adjustment Using Slack Reclamation in Multiprocessor Real-time Systems[J]. IEEE Trans. Parallel Distrib. Syst, 2003, 14(7): 686-700
- [9] Liu H, Shao Z L, Wang M, et al. Overhead-aware System-level Joint Energy and Performance Optimization for Streaming Applications on Multiprocessor Systems-on-Chip[C]// Proceedings of the 2008 Euromicro Conference on Real-Time Systems, Washington, DC, 2008: 92-101
- [10] Andrei A, Schmitz M, Elesand P, et al. Simultaneous communication and processor voltage scaling for dynamic and leakage energy reduction in time constrained systems[C]// Proceedings of the IEEE/ACM 2004 International Conference on Computer-Aided Design, Washington, DC, 2004: 362-369
- [11] Martin S, Flautner K, Mudge T, et al. Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads[C]// Proceedings of the 2002 IEEE/ACM International Conference on Computer-aided Design, San Jose, California, 2002: 721-725
- [12] Leiserson C E, Saxe J B. Retiming synchronous circuitry[J]. Algorithmica, 1991, 6(1): 5-35
- [13] Vazirani V. Approximation Algorithms(1st ed.)[M]. Springer, 2001