源程序级和算法级嵌入式软件功耗特性的分析与优化方法研究

罗 刚[1)] 郭 兵[1)] 沈 艳[2)] 廖海艳[1)] 任 磊[1)]

[1)](四川大学计算机学院 成都 610065)

[2)](电子科技大学机械电子工程学院 成都 610054)

摘 要 嵌入式系统的功耗优化可以在硬件和软件的多个层次进行,随着微电子技术的不断发展,各种底层先进硬件功耗优化技术的出现和应用,使得高层软件方面的功耗管理和优化技术逐步成为控制计算机系统功耗的重要手段.文中首先在完成嵌入式软件功耗度量的基础上,在硬件微结构级和电路级分析了软件功耗的产生原因和构成因素.然后,在软件源程序级和算法级两个层次上,采取相应措施改善影响嵌入式软件功耗特性的关联特征,以降低软件功耗.最后,针对"八皇后"典型算法问题进行了仿真实验,经过源程序级、算法级和源程序级与算法级混合 3 种优化,系统功耗最大降幅可以达到 93.2%,实验结果表明软件功耗优化方法对于降低系统功耗是可行的和有效的.

关键词 嵌入式系统功耗;软件功耗;功耗优化;源程序级;算法级

中图法分类号 TP302 **DOI 号**: 10.3724/SP.J.1016.2009.01869

Analysis and Optimization Method of Energy Consumption Characteristics in Embedded Software Based on Source-Code and Algorithm Level

LUO Gang[1)] GUO Bing[1)] SHEN Yan[2)] LIAO Hai-Yan[1)] REN Lei[1)]

[1)](*School of Computer Science & Engineering, Sichuan University, Chengdu* 610065)

[2)](*School of Mechatronics Engineering, University of Electronic Science & Technology of China, Chengdu* 610054)

Abstract The energy consumption optimization of embedded systems can be undertaken in multiple levels of hardware and software. Along with the rapid growth of micro-electronics technologies, various kinds of advanced low-level hardware energy consumption optimization technologies have been developed and applied, some high-level software energy consumption management and optimization technologies gradually become an important means to control the energy consumption of embedded systems. Firstly, after the energy consumption measurement of embedded software has been finished, this paper analyzes the generating reasons and constructing factors of embedded software energy consumption based on the micro-structure and circuit level of hardware. Then, in order to reduce the energy consumption of embedded software, some measures is adopted to improve the software-related factors of embedded software energy consumption characteristics based on the source-code and algorithm level of software. Finally, a C program of typical "Eight Queens" problem is optimized and compared by three methods of source-code level, algorithm level and mixing of source-code and algorithm level, the highest energy savings of embed-

收稿日期:2009-04-17;最终修改稿收到日期:2009-08-21.本课题得到国家"八六三"高技术研究发展计划项目基金(2008AA01Z105)资助.**罗 刚**,男,1984 年生,硕士研究生,主要研究方向为嵌入式实时系统.**郭 兵**,男,1970 年生,博士,教授,博士生导师,主要研究领域为嵌入式实时系统、SoC 和中间件.E-mail: guobing@sohu.com.**沈 艳**,女,1973 年生,博士,副教授,主要研究方向为分布式测量系统、嵌入式系统开发、无线传感器网络和机器人.**廖海艳**,女,1982 年生,硕士研究生,主要研究方向为嵌入式实时系统.**任 磊**,男,1985 年生,硕士研究生,主要研究方向为嵌入式实时系统.

ded systems can achieve up to 93.2%, and experimental results demonstrate that the energy consumption optimization methods of embedded software is feasible and effective to minimize the energy consumption of embedded systems.

Keywords embedded system energy consumption; software energy consumption; energy consumption optimization; source-code level; algorithm level

1 引 言

根据美国 In-stat 公司的市场调查,截止 2007 年底,全世界各种计算机系统(包括通用的台式机系统、专用的嵌入式计算机系统和大型的服务器系统)的保有量超过了 30 亿台,且数量继续呈快速增长的势头,其中在用的 PC 机已经超过 10 亿台,预计到 2014 年时还将翻倍.在满负荷工作状态下,当前主流配置的 PC 机功率大致在 200 瓦左右,嵌入式计算机系统(简称嵌入式系统)的平均功率为 30 瓦左右,一天按 10 小时工作时间计算,那么 30 亿台计算机(按 10 亿台台式机和 20 亿台嵌入式系统计算)一天消耗的电力平均为 26 亿千瓦时(200 瓦×10 小时/天×10 亿+30 瓦×10 小时/天×20 亿=26 亿千瓦时/天),相当于三峡年发电量 500 亿千瓦时的 5.2%.如果按 1 度(千瓦时)电收费 0.5 元人民币计算的话,全球计算机用户一年要交纳的电费为 4745 亿元人民币(26 亿千瓦时/天×0.5 元/千瓦时×365 天=4745 亿元).目前在国家提倡"节能减排"的背景下,嵌入式系统的功耗是一个日益引起人们关注的热点问题,业已受到业内软/硬件开发商和政府的高度重视[1].

从嵌入式系统的组成分析,微处理器、总线、存储器、Cache 和 I/O 接口等硬件的电路活动"直接"导致了系统功耗的产生,但影响功耗的决不只是硬件本身,硬件依赖于运行其上的软件实现信息处理功能.而嵌入式软件(本文主要指嵌入式应用软件)本身不会产生功耗,功耗只是软件执行时的"副产品",软件的指令执行和数据存取等操作驱动了底层硬件的电路活动,"间接"导致了系统功耗的产生.因此,嵌入式软件是产生系统功耗的"主动"因素和"活跃"因素,这也是嵌入式软件功耗的本质含义.先前的许多研究表明,不同的汇编指令、源程序结构、软件算法和软件的高层体系结构(即软件体系结构)造成硬件不同的工作方式,从而对系统的功耗带来不同的影响.据统计,目前大约 80%以上的系统功耗是由软件驱动产生的.

嵌入式系统的功耗优化先期主要集中在硬件层,包括材料级、工艺级、电路级、门级、RTL 级、算法级以及微结构级等层次.功耗优化可以在各个层次上展开,越高的设计层次所提供的节能空间越大,设计效率也越高.随着微电子技术的不断发展,各种底层先进硬件功耗优化技术的出现和应用,使得高层软件方面的功耗优化技术逐步成为控制系统功耗的重要手段.目前,软件层的功耗优化技术可分为源程序级、算法级和软件体系结构级 3 个层次(如图 1 所示).

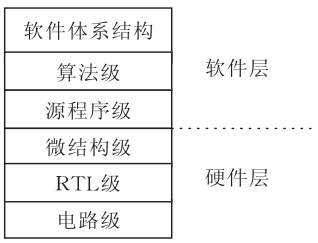

图 1 嵌入式系统功耗优化的主要层次

本文将在软件源程序级和算法级对"八皇后"典型算法问题进行分析与实验,从硬件微结构级和电路级分析程序优化前后功耗降低的原因,从而提出从软件层降低系统功耗的方法.

2 嵌入式软件功耗研究现状

自从 1946 年 2 月第一台电子计算机 ENIAC(Electronic Numerical Integrator and Calculator 电子数字积分器与计算器)在美国宾夕法尼亚大学诞生以来,旨在降低计算机系统功耗的研究就没有停止过.1999 年,DARPA(美国国防部高级研究计划署)启动 PAC/C(Power-Aware Computing and Communications 感功计算与通信)研究计划,研究内容非常广泛,涉及到计算机系统的所有方面,包括芯片、体系结构、编译器、操作系统、网络通信、实时系统、分布式系统到应用工程等多个技术层次,开发各种相关策略、算法、协议、工具和设备,为系统提供

一个完整的功耗解决方案.日本、英国、韩国、法国、俄罗斯等国家也对计算机系统的低功耗设计表现出了极大的兴趣,纷纷展开了该领域的研究工作.

国际上对于嵌入式软件功耗的研究历史并不长,1994 年 Tiwari 等人在文献[2-3]中首先提出了对软件功耗进行分析的基本概念,并建立了基本的指令级功耗模型,以 486DX 为例初步探讨了低功耗编译技术;Chung 等人从源程序代码变换角度,分析了在一个简单的硬件体系结构上编译器对于功耗和性能的影响,并利用能量方程和执行时间方程说明了对功耗地优化有时会导致性能的降低,对性能的优化有时会导致功耗增加[4];在文献[5]中,Dalal 等人利用循环展开、循环分块等优化技术对软件功耗进行了研究,并证实了利用源程序级代码优化技术能够降低软件功耗;Konstantakos 等在文献[6]中通过对基于微处理器的嵌入式系统能耗进行建模,证实了在特定嵌入式系统中,内存访问次数是决定嵌入式系统能耗的主要因素;Mukherjee 等针对数据中心的功耗管理问题,在计算能力、利用率等条件约束下,通过设计合理的软件体系结构,有效降低了数据中心的能源消耗[7];在文献[8]中,Xian 等人提出了一种可测量计算机各组成部件功耗的工具,并利用该工具成功地提高了短 CPU 时间片程序和 I/O 操作程序的功耗测量精度分别达 40% 和 90%;Aaraj 等在文献[9]中利用 ECC(椭圆曲线密码学)代替 SW-TPM(基于软件的可信赖平台模块)的 RSA 算法有效地降低了 SW-TPM 的执行时间和功耗;Brandolese 等将指令集仿真和静态源文件分析方法相结合,提出了基于统计学精度模型的估计 C 程序执行时间和功耗的全自动方法[10];Chandra 等在文献[11]中提出了一种处理器缓存能耗的线性模型,并在 ARM11 上利用 L1 数据缓存和 L2 统一缓存验证了该模型统计结果与实际结果的误差小于 4%;Petre 利用中间件语言 MIDAS 对网络能耗进行建模,成功地识别出两种不同类型的网络,并进一步区分出网络环境中的软件能耗和硬件能耗[12].

国内在嵌入式软件功耗分析与优化相关的研究工作起步晚于国外,但迄今已取得了一些可喜的成果.中国科学院计算技术研究所的赵荣彩等提出了在多线程体系结构中通过降低处理器执行频率减小系统功耗的理论模型和方法,有效解决了多线程低功耗的编译优化问题[13];电子科技大学吴琦等提出了当空闲时间长度服从 Pareto 分布时,基于截尾均值法小样本情况下 Pareto 分布形状参数的稳健有效估计算法和基于窗口大小自适应技术非平稳业务请求下的 DPM 控制算法,有效解决了计算机系统非平稳自相似业务条件下自适应动态功耗管理问题[14];中国科学技术大学雷霆等将性能无损的低能耗电压调度问题形式化为一个混合整数规划模型(MILP),提出了基于剖析结果的 PGS 算法和基于分析结果的 ADS 算法,可有效降低软件运行中所需的能耗[15];四川大学郭兵等提出了一种基于离散 Hopfield 神经网络的 RTOS 软/硬件划分方法,在一定运行时间约束下优化 RTOS 的功耗,明显地降低了可重构计算系统多任务 RTOS 的运行功耗[16].

3　嵌入式软件功耗分析与优化

目前一般的做法是通过对硬件层的微结构级和电路级进行系统功耗的分析和建模,为分析嵌入式软件功耗的产生原因和构成要素提供依据.

3.1　微结构级功耗模型

Tan 等通过微结构级的研究,建立了相应系统功耗统计模型[17],表示为

$$E_{sys}^{T}=E_{proc}^{T}+E_{idle}^{T}+E_{mem}^{T}+E_{uart}^{T}+E_{peri}^{T} \quad (1)$$

其中,E_{sys}^{T} 表示嵌入式系统总功耗,E_{proc}^{T} 表示处理器核工作时功耗,E_{idle}^{T} 表示处理器核空闲时功耗,E_{mem}^{T} 表示存储器功耗,E_{uart}^{T} 表示串口 UART 功耗,E_{peri}^{T} 表示其它外围设备功耗.在特定的应用(如计算密集应用)中,嵌入式系统处理器核空闲时功耗、串口 UART 功耗和外围设备功耗三者的值很小,一般可以忽略.因此,在本文中,将忽略 E_{idle}^{T}、E_{uart}^{T} 和 E_{peri}^{T} 3 个值,只考虑 E_{proc}^{T} 和 E_{mem}^{T} 对于嵌入式系统总功耗的影响.

对于 E_{proc}^{T},可表示为

$$E_{proc}^{T}=\sum_{i=1}^{N_{instr}^{T}}\{E_{proc}[instr\_type(i)]\times N_{cyc}(i)\} \quad (2)$$

其中,N_{instr}^{T} 表示程序运行总指令数,$instr\_type(i)$ 表示第 i 条指令的指令类型,$E_{proc}[instr\_type(i)]$ 表示当执行 $instr\_type(i)$ 类指令时,每个时钟周期内处理器功耗,$N_{cyc}(i)$ 表示第 i 条指令执行所需要的时钟周期数.

对于 E_{proc},可表示为

$$E_{proc}=\frac{V_{dd}\times I_{instr}}{f_{clk}} \quad (3)$$

其中,V_{dd} 表示处理器核的工作电压,I_{instr} 表示指令执行时的电流,f_{clk} 表示时钟频率.

对于 E_{mem}^{T},可表示为

$$E_{\text{mem}}^{\text{T}}=E_{\text{mem}}\times N_{\text{mem\_cyc}} \tag{4}$$

其中，$N_{\text{mem_cyc}}$表示存储器处于工作状态时存储器占用的总线周期数，E_{mem}表示每一个总线周期存储器功耗，在特定的硬件环境下一般是恒定不变的.

综上，嵌入式系统的总功耗可表示为

$$\begin{aligned}E_{\text{sys}}^{\text{T}}&=E_{\text{proc}}^{\text{T}}+E_{\text{mem}}^{\text{T}}\\&=\sum_{i=1}^{N_{\text{instr}}^{\text{T}}}\{E_{\text{proc}}[instr\_type(i)]\times N_{\text{cyc}}(i)\}+E_{\text{mem}}\times N_{\text{mem\_cyc}}\\&=\sum_{i=1}^{N_{\text{instr}}^{\text{T}}}\left\{\frac{V_{\text{dd}}\times I_{\text{instr}}}{f_{\text{clk}}}[instr\_type(i)]\times N_{\text{cyc}}(i)\right\}+E_{\text{mem}}\times N_{\text{mem\_cyc}}\end{aligned} \tag{5}$$

从微结构级考虑，嵌入式软件功耗主要与软件自身的 3 个特征因素相关，包括

(1) 软件运行时执行的二进制总指令数量.

(2) 软件运行时访问内存的二进制指令数量.

(3) 或者软件运行时的 CPU 周期数.

因此，通过减少软件运行时执行的二进制指令数量和访问内存的二进制指令数量都可以达到降低软件功耗的目的. 其中，软件运行时二进制指令数量的减少，实际是减少式(5)中的 $N_{\text{instr}}^{\text{T}}$参数；当软件访问内存的二进制指令数量减少后，内存占用总线的周期数将减少，式(5)中的 $N_{\text{mem_cyc}}$参数也相应会减少.

3.2 电路级功耗模型

在电路级功耗模型中，CMOS 集成电路信号翻转所造成的动态功耗占电路总功耗的 80% 以上，可表示为[18]

$$P=0.5C_LV_{DD}^2Af \tag{6}$$

其中，P 表示程序运行总功耗，C_L 表示负载电容，V_{DD}表示供电电压，A 表示电路翻转的平均频度，f 表示电路工作频率. 在特定的硬件环境下 C_L，V_{DD} 和 f 恒定不变，因此程序运行总功耗(P)和电路翻转的平均频度(A)之间存在线性关系，可表示为

$$P=KA \tag{7}$$

其中，K 表示式(6)中 C_L，V_{DD}和 f 共同影响因素总和.

从电路级考虑，在一个 CPU 时钟周期内，一个单位电路(即电容)要么进行翻转，要么不翻转，即翻转的次数只能是 1 或者 0. 因此，对于程序总体而言，式(7)可以表示为

$$P=KA=K\sum_{i=1}^{M}\left(\sum_{j=1}^{N}(C_{ij}\times F)\right) \tag{8}$$

其中，M 表示特定应用程序的 CPU 周期数，N 表示特定应用程序运行时硬件电路中被使用的电容总数量，C_{ij}表示电路中的第 j 个电容，$C_{ij}\times F$ 表示在第 i 个 CPU 周期内，电路中第 j 个电容是否翻转，$F\in\{0,1\}$，F 为 0 表示不翻转，F 为 1 表示翻转.

根据嵌入式系统硬件电路的特性，某种具体二进制指令的运行总是使用特定的硬件电路，并且特定硬件电容的翻转次数是固定的. 对于一个具体的二进制应用程序(如计算密集应用)，式(8)中 N 参数不变. 因此，可以通过减少式(8)中的 M 参数，即减少程序运行的 CPU 周期数达到降低嵌入式软件功耗的目的.

3.3 嵌入式软件功耗优化方法

目前有关嵌入式软件的功耗优化方法主要基于软件的某一方面特征开展工作，主要包括

(1) 编译器优化，即指令变换、指令重排、循环结构优化、存储器和 Cache 分配等.

(2) 源程序优化，即表达式变化、优化数据表示、程序结构重排等.

(3) 算法优化，即消除重复计算、改善数据结构、压缩数据存储空间、算法选择等.

(4) 软件体系结构优化，即体系结构的选择和变换等.

实际上，编译器许多与机器无关的前端优化工作和源程序优化是等价的，编译器只是包含了这些优化方法的工具实现，同时，编译器的优化能力也存在一定的局限性，因此，可将此部分优化工作归入源程序级优化中.

我们主要在软件的源程序级和算法级，通过改善嵌入式软件自身的相关特征，来降低软件功耗. 一些典型的方法如：

(1) 在源程序级，循环结构中采用减计数循环代替增计数循环、循环不变量代码外提和循环展开 3 种方法. 在固定次数的循环中，当采用增计数循环时，经过编译后的 ARM CPU 二进制程序判断是否进行下一次循环的步骤是：首先经过一次加法(add)操作，再经过显式地比较(cmp)操作，最后根据比较结果判断是否进入循环(BCC)操作，一次循环该过程需要三条二进制指令. 当采用减计数循环时，判断是否进入循环的步骤是：首先经过一次减法(SUBS)操作，然后根据减法设置的寄存器位(以 ARM CPU 为例，该位是 R15 中的条件码标志位 Z)判断是否进入循环(BNE)操作，一次循环该过程需要两条二进制指令. 因此，通过循环结构中减计数循环代替增计数循环的方法，循环的多次执行能够减少软件运行时执行的二进制总指令数量，从而有效

降低嵌入式软件功耗.在循环结构中,将循环不变量代码外提,使得程序在每次循环中减少了该类变量初始化并为其重复计算值的步骤;循环展开通常利用 CPU 内部寄存器存储临时变量,在完成相同逻辑运算时,被展开的循环能够降低循环内前后变量的相关性,尽量避免打断 CPU 流水线,从而减少软件运行的二进制总指令数和访问内存的二进制指令数量,最终达到降低软件功耗的目的.

(2)在算法级,采用改善数据结构的方法.如在许多数字信号处理程序中,采用 C 语言数组存储处理数据,每个数据值只能为 0 或 1,可利用 C 语言 long 型变量的各个二进制位代替相应 C 语言数组值.在 long 型变量能够表示的位数范围内,使用二进制位能够充分利用位进行操作,节约了内存空间,减少了利用数组表示时数据在 CPU 和内存之间的传递次数,也减少了软件访问内存的二进制指令数量,从而降低嵌入式软件功耗.

4　仿真实验与分析

4.1　实验环境及方法

本文采用典型"八皇后"问题验证源程序级、算法级和源程序级与算法级混合优化对嵌入式系统功耗的影响."八皇后"问题描述为:在一个 8×8 的国际象棋盘上,有 8 个皇后,每个皇后占一格,要求皇后间不会出现相互"攻击"的现象,即不能有两个皇后处在同一行、同一列或同一对角线上.

我们的测试环境是,在 RedHat Linux 9.0 操作系统上,使用交叉编译器 arm-linux-gcc 编译 UCOS II 和解"八皇后"问题源程序,并将编译后可执行文件在功耗模拟平台 EMSIM 上运行得到最终结果.

EMSIM 是建立在 GNU GDB 之上,并进行了全面的改造和扩展,可模仿多种完整目标嵌入式计算机系统的指令模拟器.它可分为 4 个层次:用户接口模块、符号处理模块、目标控制模块、目标模拟模块,其中的目标模拟模块是分析软件功耗组成的重要基础[17].我们的实验使用 EMSIM 模仿 Intel SA1100 CPU 指令.

为了方便对功耗效果进行比较,我们首先设计了可读性最好的 C 语言程序(简称原始程序)求解"八皇后"问题,其中,算法采用数组表示在 8×8 的格子上是否有皇后存在,具有清晰的逻辑步骤,程序代码由 main 函数、主功能函数及其它辅助函数组成,具有简洁的语法结构;然后,分别对该原始程序进行源程序级优化、算法级优化和源程序级与算法级混合优化 3 种处理,成为功耗更加节省的程序,采用交叉编译器 arm-linux-gcc 2.95.3 对 3 类程序分别编译(为避免干扰,编译时不使用优化选项 O1 和 O2);最后,采用交叉编译器 arm-linux-gcc 2.95.3 对原始程序进行编译和 O1 级、O2 级优化,成为运行更加高效的程序.

在源程序级优化程序中主要采用循环次数递减、循环不变量代码外提和循环展开三种方法对程序进行改进,在算法级优化程序中主要采用 C 语言中 long 型变量的二进制位表示在 8×8 的格子上皇后是否存在.

4.2　实验结果分析

"八皇后"问题的原始程序(图中编号为 L1)、源程序级优化程序(图中编号为 L2)、编译器 O1 级优化程序(图中编号为 L3)、编译器 O2 级优化程序(图中编号为 L4)、算法级优化程序(图中编号为 L5)以及源程序级与算法级混合优化程序(图中编号为 6)的实际运行结果如表 1 所示.

表 1　多种优化程序运行结果

程序类别	执行次数	CPU 周期数/Tick	二进制总指令条数	内存访问二进制指令条数	能耗/nJ
原始程序	1965	9351911	1938707	930864	24096802.7184
源程序级优化程序	1965	2834802	574562	277103	7316399.0533
编译器 O1 级优化程序	1965	2471752	574323	71206	6192166.2378
编译器 O2 级优化程序	1965	2511121	586709	69242	6303322.1359
算法级优化程序	2057	433217	83957	45202	2337489.8636
源程序级与算法级混合优化程序	2057	427775	78239	45202	1651354.2576

在表 1 中,执行次数表示为求解"八皇后"问题各程序主功能函数的递归调用次数,在完成相同功能的条件下,不同方法需要的递归调用次数不同,CPU 周期数表示主功能函数实际运行的 CPU 周期数总和,二进制总指令数表示主功能函数执行的二进制指令总条数(包括内存访问二进制指令数和其他非访存二进制指令数),内存访问二进制指令数表示主功能函数执行内存访问的二进制指令条数,能

耗表示主功能函数的能耗总和.

表 2 表示对原始程序进行多种优化后所取得的二进制总指令数、内存访问二进制指令数和功耗减少比例,可以看出,随着主功能函数二进制总指令数或内存访问二进制指令数的减少,程序能耗也随之降低,这符合式(5)中减少 $N_{\text{instr}}^{\text{T}}$,$N_{\text{mem\_cyc}}$ 可以降低功耗的预测.

表 2 多种优化程序各构成因素和能耗变化比例

程序类别	二进制总指令数减少比例/%	内存访问二进制指令数减少比例/%	能耗降低比例/%
原始程序	—	—	—
源程序级优化程序	70.3	70.2	69.6
编译器 O1 级优化程序	70.3	92.3	74.3
编译器 O2 级优化程序	69.8	92.6	73.8
算法级优化程序	95.7	95.1	90.3
源程序级与算法级混合优化程序	96.0	95.1	93.2

图 2 表示求解“八皇后”问题的多种优化程序能耗和 CPU 周期数的关系,可以看出,两者的变化几乎呈线性关系,符合式(8)的预计. 因此,通过减少 CPU 周期数,能够有效地降低嵌入式软件功耗.

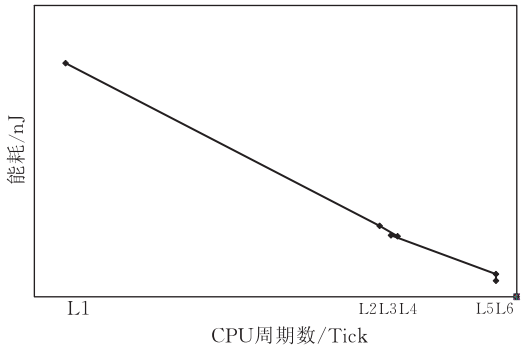

图 2 功耗与 CPU 周期数的关系

从实验结果可以看出,对嵌入式软件进行源程序级和算法级混合优化后,功耗的降低效果最明显,超过了目前交叉编译器优化的效果,这也进一步说明,关于嵌入式软件功耗分析与优化的研究是一项很有意义的工作.

5 结束语

本文利用硬件层微结构级和电路级功耗模型,分析嵌入式软件功耗的产生原因和构成要素;然后,在软件源程序级和算法级提出了功耗优化方法,对嵌入式软件功耗优化有一定的指导作用;最后,在仿真实验平台上利用“八皇后”典型算法问题验证了优化方法的有效性. 下一步需要研究的主要问题是软件体系结构级嵌入式软件功耗的分析与优化方法.

参 考 文 献

[1] Guo Bing, Shen Yan, Shao Zi-Li. The redefinition and some discussion of green computing//Proceedings of the 2008 China National Computer Conference. Xi′an, China, 2008: 44-50(in Chinese)
(郭兵, 沈艳, 邵子立. 绿色计算的重定义与若干探讨//中国计算机大会(CNCC2008). 中国, 西安, 2008: 44-50)

[2] Tiwari Vivek, Malik Sharad, Wolfe Andrew. Power analysis of embedded software: A first step towards software power minimization. IEEE Transactions on VLSI Systems, 1994, 2(4): 437-445

[3] Tiwari Vivek, Malik Sharad, Wolfe Andrew. Compilation techniques for low energy: An overview//Proceedings of the IEEE Symposiumon Low Power Electronics. San Diego, CA, USA, 1994: 38-39

[4] Chung Eui-Young, Benini Luca, de Micheli Giovanni. Source code transformation based on software cost analysis//Proceeding of ISSS'01. Montréal, Québec, Canada, 2001: 153-158

[5] Dalal Vishal, Ravikumar C P. Software power optimization in an embedded system//Proceedings of the 14th International Conference on VLSI Design(VLSID'01). Bangalore, India, 2000: 254-259

[6] Konstantakos V, Chatzigeorgiou A, Nikolaidis S, Laopoulos T. Energy consumption estimation in embedded systems. IEEE Transactions on Instrumentation and Measurement, 2008, 57(4): 797-804

[7] Mukherjee T, Tang Qinghui, Ziesman C, Gupta S K S, Cayton P. Software architecture for dynamic thermal management in datacenters//Proceedings of the 2nd International Conference on Communication Systems Software and Middleware(COMSWARE'07). Bangalore, India, 2007: 1-11

[8] Xian Changjiu, Cai Le, Lu Yung-Hsiang. Power measurement of software programs on computers with multiple I/O components. IEEE Transactions on Instrumentation and Measurement, 2007, 56(5): 2079-2086

[9] Aaraj Najwa, Raghunathan Anand, Ravi Srivaths, Jha Niraj K. Energy and execution time analysis of a software-based trusted platform module//Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'07). Nice, France, 2007: 1-6

[10] Brandolese Carlo. Source-level estimation of energy consumption and execution time of embedded software//Proceedings of the 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools(DSD'08). Parama, Italy, 2008: 115-123

[11] Chandra L, Roy S. Estimation of energy consumed by software in processor caches//Proceedings of the IEEE International Symposium on VLSI Design, Automation and Test (VLSI-DAT'08). Hsinchu, Taiwan, 2008: 21-24

[12] Petre L. Energy-aware middleware//Proceedings of the 15th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS'08). Belfast, England, 2008: 326-334

[13] Zhao Rong-Cai, Tang Zhi-Min, Zhang Zhao-Qing, Gao Guang R. A multithreaded compiler optimization technology with low power. Journal of Software, 2002, 13(6): 1123-1129(in Chinese)
(赵荣彩, 唐志敏, 张兆庆, Gao Guang R. 低功耗多线程编译优化技术. 软件学报, 2002, 13(6): 1123-1129)

[14] Wu Qi, Xiong Guang-Ze. Adaptive dynamic power management for non-stationary self-similar requests. Journal of Software, 2005, 16(8): 1499-1505(in Chinese)
(吴琦, 熊光泽. 非平稳自相似业务下动态功耗管理自适应算法. 软件学报, 2005, 16(8): 1499-1505)

[15] Lei Ting, Li Xi, Zhou Xue-Hai. Performance lossless voltage scheduling for low energy software. Journal of Computer Research and Development, 2006, 43(6): 1090-1096 (in Chinese)
(雷霆, 李曦, 周学海. 低能耗软件设计中的性能无损电压调度技术研究. 计算机研究与发展, 2006, 43(6): 1090-1096)

[16] Guo Bing, Shen Yan, Wang Dian-Hui, Li Zhi-Shu, Chen Xiang-Dong. A power optimization approach to real-time operating systems based on discrete Hopfield neural networks. Chinese Journal of Computers, 2007, 30(9): 1573-1579(in Chinese)
(郭兵, 沈艳, 王殿辉, 李志蜀, 陈向东. 一种基于离散Hopfield神经网络的RTOS功耗优化方法. 计算机学报, 2007, 30(9): 1573-1579)

[17] Tan T K, Raghunathan A, Jha N K. A simulation framework for energy-consumption analysis of OS-driven embedded applications. IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, 2003, 22(9): 1284-1294

[18] Najm Farid N. Transition density: A new measure of activity in digital circuits. IEEE Transactions on Computer Aided Design of Integrated Circuits and System, 1993, 12(2): 310-323

LUO Gang, born in 1984, M. S. candidate. His research interests focus on embedded real-time system.

GUO Bing, born in 1970, Ph. D., professor, Ph. D. supervisor. His current research interests include embedded real-time system, SoC and middleware.

SHEN Yan, born in 1973, Ph. D., associate professor. Her current research interests include distributed measurement systems, embedded system development, wireless sensor networks, and robotics.

LIAO Hai-Yan, born in 1982, M. S. candidate. Her research interests focus on embedded real-time system.

REN Lei, born in 1985, M. S. candidate. His research interests focus on embedded real-time system.

Background

The work was supported by the National High Technology Development Program (863 Program) of China under grant No. 2008AA01Z105. In this project, the power consumption optimization of embedded systems is based on a Software/Hardware co-design method, and cooperated with Department of Computer Science Laboratory of Embedded systems of the Hong Kong Polytechnic University. In fact, since 2002, the authors have been focusing on the power consumption of embedded systems.

A large number of computer systems have been widely used, and play an very important role in the progress of human society development, but "Every coin has two sides", it also has some clear negative impact on the environment, and power consumption is one of the main aspects.

The power consumption optimization can be carried out at various levels, the higher design level provides a larger space to optimize power consumption and higher design efficiency. At the early stage, power consumption optimization of embedded systems focused on hardware level, including material level, technological level, circuit level, gate level, RTL level, algorithm level and micro-structure level, etc. With the continuous development of microelectronic technologies the application of lots of underlying hardware power consumption optimization technologies make high-level software power consumption optimization technologies gradually become an important way to control system power consumption.

At present, the software level power consumption optimization techniques mainly depend on the characteristics of software, and can be divided into source code level, algorithm level and software architecture level. Software is higher than hardware on structural level, and the power consumption optimization methods of software level provide greater power saving space, which is the significance of research presented in this paper.