システムソフトウェアと
オペレーティング・システム 91−1
（2002．8．21）

OSによる大容量外部メモリの省電力化の手法

小林 さとみ
京都大学
人文科学研究所
satomi@zinbun.kyoto-u.ac.jp

中西 恒夫
九州大学大学院
システム情報科学研究院
tun@f.csce.kyushu-u.ac.jp

福田 晃
九州大学大学院
システム情報科学研究院
fukuda@f.csce.kyushu-u.ac.jp

本稿ではOSのスケジューラと協調して行う省電力メモリ管理の実現方法について述べる．組み込みシステムでは，アクセス速度や電気的特性の異なる複数の外部メモリを搭載していることが多い．主にタスクを常駐するために用いられているDRAMは，複数の電力状態に遷移することが可能で，実メモリの容量より小さいバンクという単位で構成されている．また組み込みシステムでは，全てのタスクが主メモリ上に存在しているため，電力制御を単純に行うことができない．また従来ハードウエアで実装されている参照頻度の履歴から制御する方法では，実行予測を行っていないため十分でない．本稿ではスケジューラのTCBによる情報と，バンクとページの対応を与えるテーブルを利用してバンクをカテゴリに分類し，ソフトウエア的に省電力化を実現する方式を提案する．

A Power Aware Embedded Operating System with Controlling Each Off-chip Memory Bank

Satomi Kobayashi[1], Tsuneo Nakanishi[2] and Akira Fukuda[2]

[1] Institute for Research in Humanities, Kyoto University.
[2] Graduate School of Information Science and Electrical Engineering, Kyushu University.

This paper proposes a power aware memory management cooporated with the scheduler in a embedded operating system. Many embedded systems equip several memory chips with different access speed and different electrical characteristics. Among them, DRAM is used for task execution, and it has several power states and consists of banks of which size is smaller than total memory chip. In addtion, DRAM in embedded systems is used for executing tasks during the system is on, the change of power state can not be done with simple method. Some studies has been done as hardware implementation, although it is not proper solution. In this paper, we propose a methodology for implementation of memory management with controlling several banks which are devided into several categories. TCB information and paging mechanism are used to make partial memory into low power mode by software.

1 Introduction

Long battery life is one of the major issues for portable information devices such as PDAs and cellular phones. Such devices needs a large memory space and the memory power consumption have to be considered in developing process.

Studies on power saving have been done with both of HW and SW approaches. Most of the studies are compiler level or architecture design level approaches. There are not so many studies on an operating system level approach[4]–[7].

For embedded systems, the memory on a single board consumes much electric power.

DRAM is commonly used in portable devices, but needs a lot of power. As shown in Table 1 , [1] and [2], a large space of DRAM chips sometimes consumes higher power than CPUs.

In addtion, for memory architecture of embedded systems, some systems have internal memory in a chip containing a processor and external memory outside of the chip to provide large size of memory. Furthermore, the off-chip memory normally has several commands controlled power modes by

Table 1: The electrical characteristics of SRAM,DRAM and processor.

	Frequency	Voltage
Micron SRAM	200MHz	2.5 V
Micron SDRAM	133MHz	3.3 V
Mobile PentiumIII	600MHz	0.6 V
PentiumIII	600MHz	1.35 V

software.

Therefore we can propose a memory power saving scheme controlling high power off-chip memory banks into low mode by an operating system. When a electric device has large memory and memory consists of multi banks, if the memory can control memory power mode partially the high power saving performance can be expected.

For the above hardware architecture, we proposed an off-chip memory power saving scheme[8]. We showed that the proposed scheme saved the power compared with other existing schemes through simulation experiments.

The above target memory architecture was assumed that the off-chip memory consisted of a single memory bank. There are, however, some existing embedded systems that have multiple memory banks each of which has power saving mode to provide larger size of the off-chip memory. It is important to study power saving memory management for this target system.

We have organized this paper as follows. In the next section, we describe related work. Section 3 describe the target hardware architecture we consider in this paper from the viewpoint of memory architecture and software controllable power saving modes. Section 4 proposes and describes a power saving memory management scheme. Section 5 concludes the paper and describes future works.

2 Related Work

Over the past few years, power saving operating systems have been studied in verious fields and aspects.

Lu et al. [4] proposed HDD power saving optimization algorithm for PC environments using ACPI (Advanced Configuration and Power Interface). In their paper an operating system changed electric power states into three states of writing, spinning, and stopping by setting of time-out. The paper also optimize power consumption with using a prediction value of session time, and the length of actual session time.

Lebeck et al. [5] proposes a method that raises the hit rate of cache. They propose the memory management by deviding memory into paging unit of external memory. The external memory is Rambus RDRAM. A simple histogram is employed for its priority. The threshold for activating the paging mechanism is calculated before the execution, for getting paging overhead dynamically.

Qiu et al.[6] poropsed a stochastic model of a power-managed system. They make a model of demander and offer of power saving service, and they propose a solution to make a priority of that service model.

Marchal et al. [7] developed an application-specific power saving manager, a devise driver for MPEG applications. They devide 4 layer, and analyse control and data flow statically, to predict memory access.

3 Target Hardware Architecture

As we described before, the target hardware system is on-chip/off-chip memory architecture in this paper. This means the system of low power cost on-chip memory and high power cost off-chip memory. In addition, the off-chip memory consists of different architecture. Total memory including the on-chip memory and the off-chip one constitutes a linear physical address space. The second cache is not installed in this system. The hardware system contains MMU and supports virtual memory for paging. Among the off-chip memory, there exists three different types, ROM, SRAM and SDRAM in usual embedded systems.

A scheme proposed in this paper utilizes the virtual memory paging mechanism to replace virtual pages between the memory modules, to the off-chip memory from the on-chip memory or vice versa.

Each memory bank of the off-chip memory has multiple power modes provided by hardware, a high mode, a low one, and a sleep one. These modes can be controlled by software, memory management in an operating system. The high mode means normal mode that is provided so far. On the other hand, the low mode and the sleep mode are for saving the power.

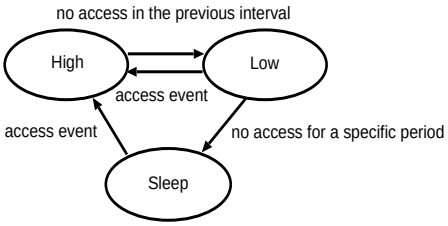

Figure 1: Power mode state transition in an off-chip memory bank.

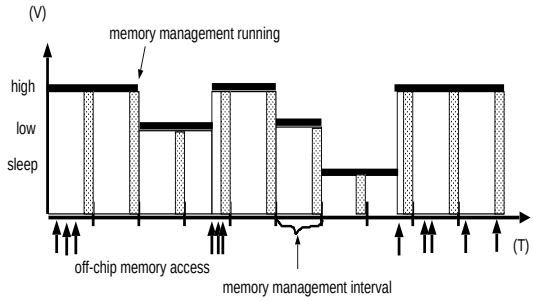

Figure 2: A time sequence example of changing the power mode.

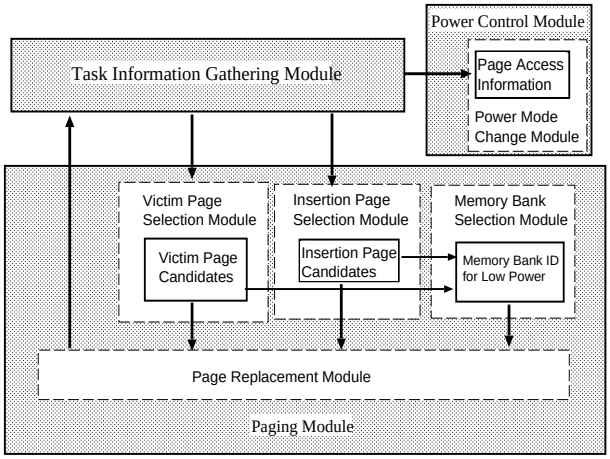

Figure 3: Organization of the power aware memory management.

4 The Methodology for the Implementation of Power Aware Memory Management

4.1 Main Ideas

In this section, we propose the way of memory management decrease off-chip memory power consumption.

Memory management transfers the frequent access pages on off-chip memory into the off-chip memory controlling memory bank power mode. If the memory management controls the off-chip memory uniformly at once [8], the low-mode timing depends on the application program's data access, the probability of changing power mode is not so high enough. Therefore we use the method of making low power mode partially, when the memory dosen't have active tasks.

Since our target system has memory bank power control to change its states into one of three modes, and makes sleeping and low power mode time as long as possible. In addition, all tasks transit their executing state, executing, sleeping and stopping. Our memory management gathers executing information of application programs by accessing between fixed intervals.

The on-chip memory has limited paging area between on-chip pages and off-chip pages other than operating system area, in order to put high cost energy pages. Memory power consumption is calculated by summation of each memory bank of standby current and accessed current.

```
Standby + Access
```

The standby power can be devided into normal power mode and low power mode.

```
Normal mode power + Low mode power
   + Sleep mode power
```

When active tasks data area are allocated to the on-chip memory, memory management can change the off-chip memory power mode to the low mode, the sleep one as long as possible, as wide as possible. On the consequence, the power consumed by the off-chip memory can be decreased. Figure 2 is a time sequence and voltage of using three power modes.

4.2 Load and Execution of the Memory Management

Our memory management modules are executed as tasks, after the loading procedure finishes. All of operating system modules are located on the on-chip memory area.

A scheme proposed in this paper is realized with three modules, a task information gathering module, a power control module, and a paging module as shown in Figure 3.

Since all processes are treated as tasks in real-time systems, tasks execution is generated by interruption. In our memory management, task information gathering module starts its execution by timer interruption. It calls Paging module and Power control module when it decides to make off-chip memory's status into low power mode.

The task information gathering module is activated at every τ period time. The page transferring between insertion page candidates and victim page candidates is not always activated when task information gathering module runs. It runs when paging module decide the transferring will work in next paging interval.

4.3 The way to generate a Memory Bank Table

The task information gathering module is responsible for gathering information of tasks on the system, and generate a Mememory Bank Table. The memory management attaches the scheduler's TCB and MMU output, in order to pass the selected candidate memory banks to the next paging module for low power. As shown in Figure 4, the TCB has task's executing state parameters, priority parameters, and the logic address parameters. The executing state parameters are expressed as the task status parameters that the scheduler determines, executing, waiting, stopping flags. The priority parameters means priority number of the executable tasks. The logic address parameters have information of tasks virtual memory address to physical memory address.

In order to select a low power memory bank, we define a MBT (Memory Bank Table) which gives task and bank mapping. MBT is a data structure which consists of memory bank, the page information and priority information.

This memory bank table is filled when the page replacement occurs. Paging module access this area and decide the low power transit memory bank in next interval and insertion pages for paging.

Figure 4 shows the TCB physical address translation flow and the memory bank map table with which the TCB has the physical memory task page. The memory bank map table has a table when the memory access occurred.

The memory bank table consists of these parameters below.

Bank Property: This is an poinger of bank identification data structure which is assigned to each memory bank. In addtion, this includes memory electrical specification about ROM,SRAM,SDRAM.. and so on.

Page Property: This consists of pointers to a page data structure which includes a task data structure. Page's physical address differs from time to time when page transfer occures. MMU creates mapping information from logical address to physical address.

Priority: This data comes from the insertion page selection module and victim page selection module output. Decision result of low power bank priority which is made in the memory bank selection module. There are three dicision strategies for selecting low power banks.

This map table helps the Memory Bank Selection Module to decede which bank will work for low power performance. In addition, Task Information Gathering Module collect the memory access history which is reffered by selection of victim page candidates and insertion page candidates.

4.4 Timing for changing the power mode

When the power control module is activated, the module decreases a power mode level of the memory banks that are not accessed during the previous interval τ. That is, when the power mode level of a memory bank is the high mode,(the low level) the level is decreased to the low level(the sleep level). These are shown in Figures 1 and 2.

This power controle scheme make banks of stopping tasks(low priority tasks) into low power mode automatically. Accodingly we have to consider the banks of high priority tasks.

Accesses to the on-chip memory are always permitted. On the other hand, it depends on the power mode of the off-chip memory whether accesses to the off-chip memory are permitted or not. When an off-chip memory bank is in the high mode, accesses to the memory bank are always permitted. On the other hand, when an off-chip memory bank is in the low mode, the power of the memory bank is saved. In this case, accesses to the memory bank are not permitted to cause access exception to software, memory management in an operating system.

When the off-chip memory is in the sleep mode, waking up for normal access takes longer time than when it is in the low power mode. When the system needs its use in sleep mode, it sends signals to make the memory wake up. This behavior is the same as the case of the low mode except for more time until the access is permitted.

4.5 Low Power Memory Bank Selection Algorithms

The decision for low power memory banks is caliculated from the insertion page candidate data and the victim page candidate data and the MBT.

For implementation,three main strategies can be proposed for low power.

A: the banks have the longest low power mode in the next interval.

B: the banks change their power state as soon as possible.

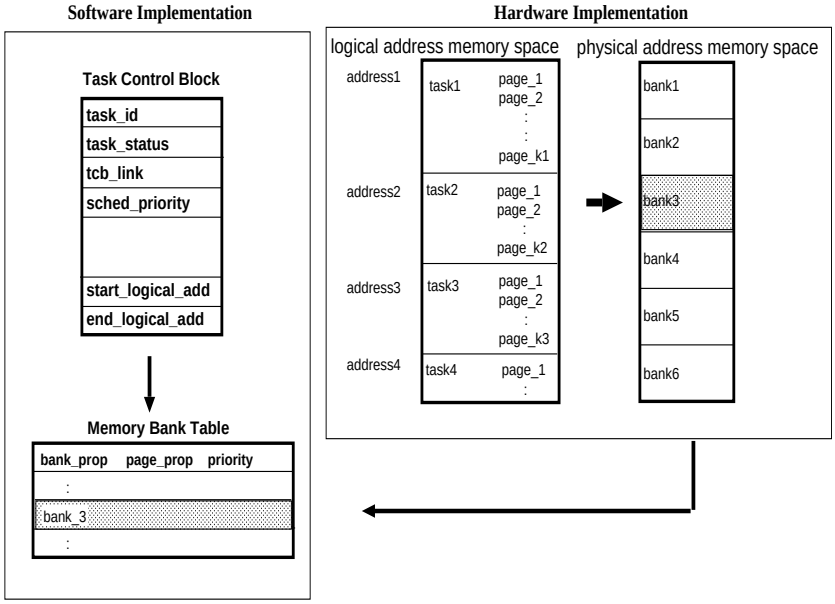

Figure 4: Information gathering flow from TCB and MMU output to Memory Bank Table.

C: the banks become in the low or sleep mode with lightest overhead.

In the bank selection module, the three strategies procedure function is different when the interpretation of the output of the insertion page and the victim page selection and the MBT. We will show an example of strategy A.

(1)The insertion page selection

The insertion page selection module is responsible for choosing the candidate pages in the off-chip memory to be inserted into the on-chip memory. The real candidates are selected when the low power memory bank finishes the procedure of getting memory bank candidates. The processing of selecting insertion page candidates is shown in the following processes.

1)The module chooses the off-chip memory banks where the highest priority task is loaded. The highest priority task means the task scheduled by the processor scheduler at the next scheduling time point.

2)Among the off-chip memory banks chosen by the above processing, the module chooses the off-chip memory bank to which the most frequently accessed page in the past interval. Some prediction methods is employed.

3)The module chooses the pages that are in the highest priority task and in the memory bank chosen by the above processing 2) as the insertion pages.

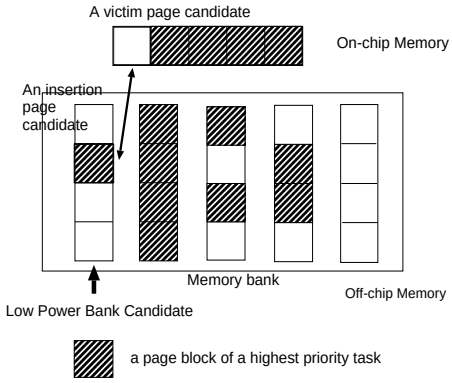

Figure 5: A page replacement example between the on-chip memory and the off-chip memory banks.

(2)The victim page selection

This module is responsible for choosing victim pages in the on-chip memory if there are enough spaces to allocate the insertion pages. There are some policy alternatives to choose the victim pages such as conventional policies, LRU(Least Recently Used), LFU(Least Frequently Used), and so on. In addition, there are some alternatives whether these policies are globally performed or not.

5 Conclusions and Future Work

In this paper, we proposed a methodology for implement a power saving operating system by controlling memory bank power mode. The proposed

```
int Tcb[MAXTASK];
/* pointers to the TCB */
int Mbt[MAXTASK];
/* pointers to the memory bank table */
int CandidateBank[MAXBANK];
int InsPage[MAXINSERTION];
int LowPowerBank;

SelectLowPowerBank(htaskid)
{

  get the highest priority task id from Tcb
  htaskid = the highest priority task id
  set Ins_page empty

  if(the htaskid is not empty){
      set the bank ids of the MBT
         to CandidateBank
      get bank ids which have accessed pages
         in previous memory management interval
      delete bank ids of unaccessed pages
         from CandidateBank
      change CandidateBank order in the small
         pages number order
  else{
      set htaskid empty
  }

  InsPage = page id of the first CandidateBank
  set the the first CandidateBank's bank id to the
    LowPowerBank

}
```

Figure 6: Low power bank selection algorithm.

scheme utilizes the schedulers task information and combined paging between the off-chip memory and the off-chip memory banks with changing the power mode of the off-chip memory banks.

In future work, we evaluate and verificate the proposed scheme from the viewpoint of speed and power performance by using the simulator developed in [9]. In addition, this implementation of a power aware operation system is also under construction with Hitachi SH3.

Acknowlegements

This research is partially supported by Grant in Aid for Scientific Research (No.12480099) from The Ministry of Education Science, Sports and Culture of Japan.

We would like to appreciate Prof. Katsumasa Watanabe, Dr. Masaki Nakanishi at Nara Institute of Science and Technology and Dr. Takashi Horiyama at Kyoto University for their precious advice.

References

[1] "Micron memory data sheet", *http://www.micron.com/* .

[2] "Samsung memory data sheet", *http://samsungelectronics.com/* .

[3] F.Catthoor, S.Wuytack, E.D.Greef, F.Balasa, L.Nachtergaele and A.Vandecappele: "Custom Memory Management Methodology", *Kluwer Academic Publishers*.

[4] Y.H.Lu, T.Simunic and G.Micheli: "Software Controlled Power Management," *Proc. of the 7th Int'l Workshop on Hardware/Software Codesign*, pp.157-161, 1999.

[5] A.R.Lebeck, X.Fan, H.Zeng and C.Ellis: "Power Aware Page Allocation," *Proc. of the Architectural Support for Programming Languages and Operating Systems*, 2000.

[6] Q. Qiu, Q. Wu and M. Pedram: "Stochastic Modeling of a Power-Managed System: Construction and Optimization," *Proc. of ISLPED*, pp.194-199, 1999.

[7] P.Marchal, C.Wong, A.Prayati, N.Cossement, F.Catthoor, R.Lauwereins, D.Verkest and H.De Man: "Dynamic Memory Oriented Transformations in the MPEG4 IM1-Player on a Low Power Platform," *Proc. of PACS 2000*, pp.40-50, 2001.

[8] S.Kobayashi, T.Nakanishi and A.Fukuda: "Power Aware Memory Management for Embedded Systems," *Proc. of SCI 2002*, 2002(to appear).

[9] S.Kobayashi, T.Nakanishi and A.Fukuda: "A Simulator of Memory Management for Low Power," *Proc. of PDPTA 2001*,Vol.1, pp.431-436, 2001.