The 2004 IEEE Asia-Pacific Conference on
Circuits and Systems, December 6-9, 2004

AN ENERGY-DELAY EFFICIENT POWER MANAGEMENT SCHEME FOR EMBEDDED SYSTEM IN MULTIMEDIA APPLICATIONS

Wei-Cheng Lin and Chung-Ho Chen
Department of Electrical Engineering, National Cheng-Kung University
No.1 University Road, Tainan, Taiwan 70101, R.O.C
Tel: 886-6-2757575-62400-722   Tel: 886-6-2757575-62394
Kevin@casmail.ee.ncku.edu.tw   chchen@mail.ncku.edu.tw

ABSTRACT

High-performance embedded systems tend to use caches and memory hierarchy to speed up program execution. This increases DRAM idle time (inter-access time) and provides opportunity for reducing memory energy usage by performing memory state transition to a low-power mode. However, additional delay due to resynchronization may greatly increase the system response time. This work focuses on exploiting state transition techniques to reduce DRAM energy usage and also mitigating the penalty of resynchronization time. We propose a state transition technique based on address prediction that predicts the inter-access time between memory accesses. According to predicted inter-access time, the DRAM is directly put into a low-energy mode and transits back to the operation mode as the idle timer expires to avoid resynchronization overhead. Experiments using multimedia application show that the proposed scheme has achieved the best energy-delay performance than other previous policies.

1. INTRODUCTION

Memory energy reduction strategies have become an important concern in the design of embedded systems. Pervious works on this issue [1-4] have focused mainly on the memory power saving aspect while ignoring the impact of resynchronization time.

To reduce memory power consumption, modern DRAMs (e.g., RDRAM [6]) provide various low-power modes. A resynchronization time is required when the memory transits back to the active mode. If a DRAM memory is put into a low-power mode frequently, resynchronization overhead may degrade the performance in terms of the execution time and offset the benefit of memory energy reduction.

In this paper, we present state transition scheme based on an inter-access time predictor that decides whether to put the memory into a low-power mode and when to transit back to the active mode to avoid resynchronization overhead. Our scheme uses the access characteristics to predict the inter-access time for multimedia applications. Observing the behavior of memory accesses, we found that certain address requested by the processor accompanies quite a long inter-access time. These addresses and inter-access time are recorded in a table to predict the next inter-access time.

Fig. 1 shows the idea of this simple yet efficient state transition policy. When the access address is found in the table, the DRAM is put into a low-power mode immediately after the transaction and the memory is wakened up to the operation mode as the predicted timer expires. In our experiment, this scheme can cover around 90% of long inter-access times (over 20 cycles) and avoid around 80% the resynchronization overhead.

The rest of this paper is organized as followers. Section 2 presents the background and related work. The proposed architecture and simulation system are discussed in Section 3 and Section 4, respectively. Section 5 shows the results of simulations. Finally, Section 6 summarizes the conclusions of this paper.

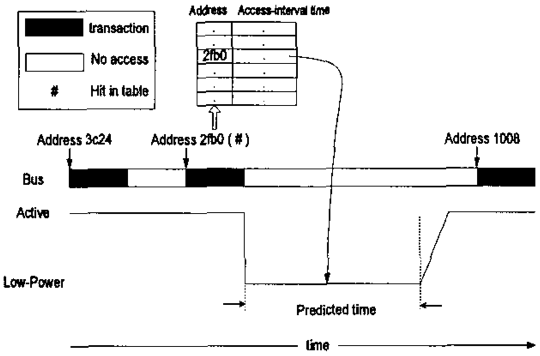

Fig. 1 The basic idea of the proposed scheme

2. BACKGROUNDS AND RELATED WORK

Table 1 shows the operation current and resynchronization time (based on cycle time of 2.5ns) in the different operation mode of a RDRAM memory. A RDRAM stays in the attention state waiting for access request to come. When an access request arrives, the RDRAM transits from the attention state to the active state. Typically, a lower power state requires longer resynchronization time. RDRAM can be put into any of

0-7803-8660-4/04/$20.00 ©2004 IEEE

low-power modes (standby, napping or power-down) when it is not servicing a memory request.

Each RDRAM bank can be put into a low-power operating mode independently to reduce power consumption. Various power management policies have been proposed for saving energy usage [1, 2, 4]. Embedded systems incorporate RDRAM for multimedia applications have been proposed in [5]. Multimedia applications such as image and audio processing strongly exhibit the principles of spatial and temporal locality. The next access is likely close to the current location (spatial locality) or is likely to be needed in the near future (temporal locality).

Table 1. Example of operating currents and resynchronization time in different operation modes for a RDRAM memory.

Operating mode	Operating current (mA)	resynchronization time (cycles)
ACT	550	0
ATTN	148	0
STBY	101	2
NAP	4.2	30
PDN	2.8	9000

Consequently, the accesses often locate in the same bank for a long duration of time. The bank that currently serves the access is called the active bank and the others are called idle banks. Usually the idle banks are put into a low-power mode. In this work, we focus on reducing the energy wasted during the inter-access time of the active bank.

There are several predictors proposed previously for that we use to compare with our approach. A constant threshold predictor (CTP) transfers states based on a constant number of idle cycles measured by statistics or calculations [1}. An adaptive threshold predictor (ATP) can dynamically adapt the thresholds to avoid miss prediction. A history-based predictor (HBP) estimates the inter-access time based on the previous inter-access time.

3. ADDRESS-AWARE PREDICTOR ARCHITECTURE

The predictor we use is an address-aware predictor (AAP). We record a chosen address and the inter-access time associated with this address for memory state transition and inter-access time prediction. The selection criterion is based on if the inter-access time is long enough.

Fig 2 shows the architecture of the AAP scheme that is simply a small cache. Each of the entries contains a 32-bit address field, and a 10-bit field for inter-access cycles. The rest includes comparators, a counter, a 32-bit address register, and a subtractor. The operation follows the steps labeled in Fig 2 (The six steps are shown as a circled number.) The counter starts counting right after each bus transaction is completed.

Step 1: When a new access request arrives on the address bus, the counter is read. This value is the inter-access time between the previous access and this incoming access. Note that at this moment, the address register still holds the address of the previous access.

Step 2: The value of the counter is compared to a threshold value (M) to determine whether the inter-access time is long enough to be stored in the table. If the inter-access time is smaller than the threshold value, updating table in Step 3 is ignored.

Step 3: The previous address and the inter-access time are written into the table.

Step 4: Then the current address is written into the address register and the counter starts counting right after the bus transaction is completed. Note that the counter holds the maximum value when the inter-access time exceeds the maximum counter value.

Step 5; The content of the address register is compared with the addresses stored in the table. Assume that a match occurs, which signals the memory controller to put the active bank into a low power mode.

Step 6: The inter-access time stored in hit entry is subtracted by a value N in order to wake up the idle bank in advance to avoid resynchronization overhead.

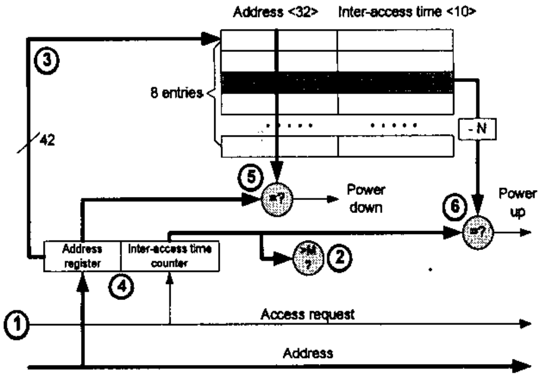

Fig. 2 The architecture of the address-aware predictor scheme

4. SIMULATION SYSTEM

We have designed an RDRAM controller that operates in close page policy with the proposed scheme (AAP). Fig 3 shows the simulation system.that contains an ARM9 RISC CPU as the host processor, AHB bus, RCU (RDRAM Control Unit), RAC (Rambus ASIC Cell) and, RDRAM devices. The processor, AHB bus, RCU and AAP are all written in Verilog RTL code while the RAC and RDRAM devices are written in Verilog behavior-level.

The proposed scheme was simulated by running the benchmark programs on a RISC processor. We use three multimedia benchmarks (JPEG encoder, MPEG layer-2 audio decoder, and MP3 decoder).

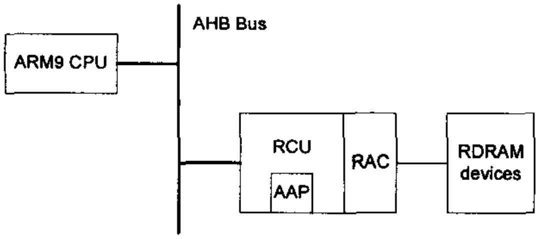

Fig. 3 The simulation system

The specification of the system used in the simulation is listed as follows:

- The RISC processor operates in 200MHz clock rate.
- The CPU includes an 8KB instruction cache arranged as a 64-way set-associative cache.
- 'The CPU includes an 8KB data cache arranged as a 64-way set-associative cache and a write buffer that can hold up to16 words of data and four separate addresses.
- 32MB RDRAM organized as 4 banks.
- The value M is set to 20 and the value N is also set to 20.
- The table size is 8 entries.

5. RESULT OF SIMULATION

The schemes evaluated include close page policy (CP), constant threshold predictor (CTP), adaptive threshold predictor (ATP), address-aware predictor (AAP), and aggressive power-down policy (APP). The CP always closes the active row after an access and holds in the attention mode. APP also uses the close page policy and immediately transits to the napping mode after completing the access for maximum power saving.

Fig. 4 shows the comparisons of energy consumption among the schemes investigated. The energy consumed by the CP is used as a base. Not surprisingly, the APP scheme consumes the less energy. Fig. 5 compares the program execution time among the standard scheme (CP) and the improved schemes (CTP, ATP, HBP, AAP and APP).

Compared with the baseline memory model (CP), the AAP scheme reduces energy consumption by 15% and increases execution time by only 3.3%. The CTP and the ATP predictor achieve the same degree of energy consumption. The HPB is too simple to predict the inter-access time correctly, that results in poor performance in energy saving.

With respect to the execution time, the increased delay of the AAP is smaller than the other power-aware state transition policies since it can avoid around 80% of resynchronization time. The APP reduces the memory energy consumption by 25%. It performs much better than the others in energy saving because the APP enters the power saving mode immediately. However, this reduction in memory energy is at the expense of increasing the execution time by 22.6%.

Fig. 6 shows the reduced energy divided by the additional delay. Because the additional delay of AAP is very small, it has the best energy-delay performance than the other policies evaluated.

6. CONCLUSION

In this paper, we propose an effective DRAM power management scheme (AAP) that predicts inter-access time according to the access characteristics in cache-based embedded systems for multimedia applications. The scheme reduces the DRAM energy usage and mitigates the penalty of resynchronization time. We show that the proposed scheme has achieved the best energy-delay performance than other previous policies.

ACKNOWLEDGMENTS

The work in this paper is in part supported by the National Science Council, Taiwan ROC, under NSC 92-2220-E-006-006.

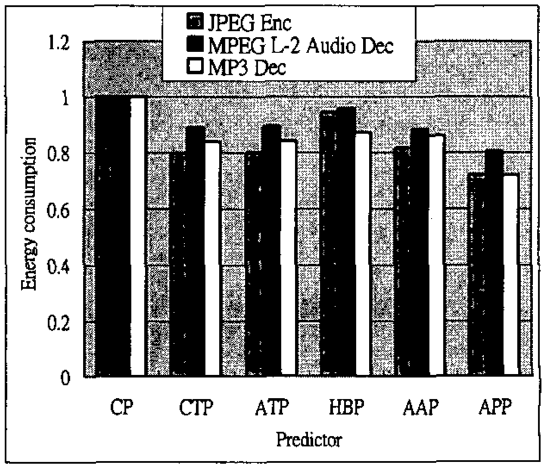

Fig. 4 Comparison of energy consumption using different predictors

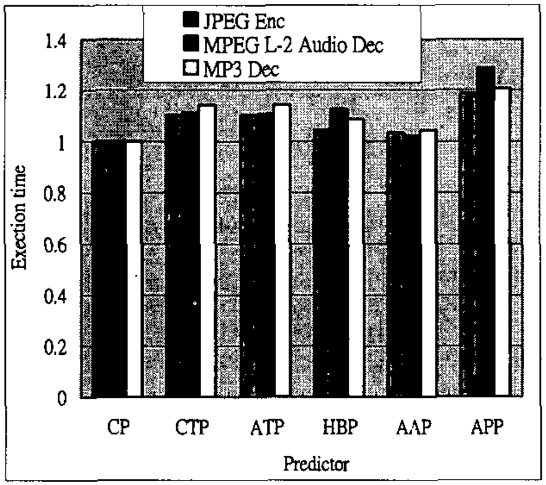

Fig. 5 Comparison of program execution time using different predictors

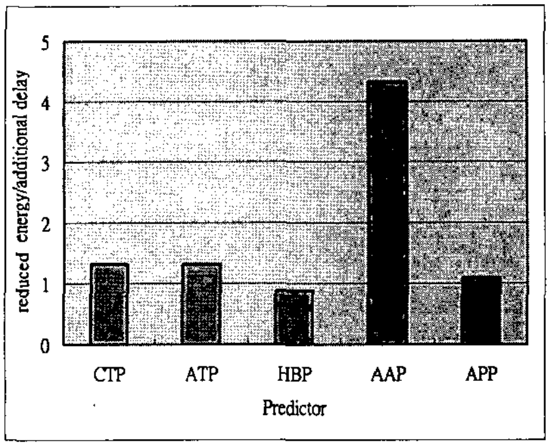

Fig. 6 Reduced energy divided by additional delay

7. REFERENCES

[1] Delaluz V., Kandemir M., Vijaykrishnan N., Sivasubramaniam A., and Irwin. M. J., "Hardware and software techniques for controlling DRAM power modes," IEEE Transactions on Computers, Vol.50, No11, pp. 1154 - 1173, November 2001.

[2] Victor De La Luz, Mahmut T. Kandemir, and Ibrahim Kolev. "Automatic data migration for reducing energy consumption in multi-bank memory systems," in the proceedings of the 39th Design Automation Conference, pp. 213-218, June 2002.

[3] Gries M., "The impact of recent DRAM architectures on embedded systems performance," in the proceedings of the 26th Euromicro Conference, pp. 282 - 289, Sept., 2000.

[4] Xiaobo Fan, Ellis C.S. and Lebeck A.R., "Memory controller policies for DRAM power management," International Symposium on Low Power Electronics and Design (ISLPED), pp. 129 - 134, August 2001.

[5] Suzuki K., Daito M., Inoue T., Nadehara K., Nomura M., Mizuno M., Iima T., Sato S., Fukuda T., Arai T., Kuroda I., and Yamashina M., "A 2000-MOPS embedded RISC processor with a Rambus DRAM controller," IEEE Journal of Solid-State Circuits, Volume: 34 , Issue: 7 , pp. 1010 - 1021, July 1999.

[6] Rambus. RDRAM, 1999. http://www.rambus.com.