Reducing Access Energy of On-Chip Data Memory Considering Active Data Bitwidth

Takanori Okuma   Yun Cao   Masanori Muroyama   Hiroto Yasuura
Department of Computer Science and Communication Engineering
Kyushu University
6–1 Kasuga Koen, Kasuga, Fukuoka, 816-8580 Japan
{okuma,cao,muroyama,yasuura}@c.csce.kyushu-u.ac.jp

ABSTRACT

This paper presents a new concept called active data bitwidth, which is the effective data length of data bus. By means of profiling the active data bitwidth dynamically, we present a novel low-energy memory access technique for on-chip data memory design. By reducing the redundant access energy of data memory, our experimental results of two real applications, show that we can achieve significant energy reduction. Compared to the monolithic memory, for JPEG, 52.2%; for MPEG-2 84.2%, the energy reduction is reported. Compared to the memory banking technique, 12.3% energy reduction for JPEG and 65.9% for MPEG-2 is reported.

Categories and Subject Descriptors

C.3 [**Computer Systems Organization**]: Special-Purpose and Application-Based Systems

General Terms

Design

1. INTRODUCTION

The advent of new VLSI technologies as well as the advent of state-of-the-art design techniques such as System-on-Chips (SoCs) design methodologies has made multi-million gate chips a reality. Recent embedded SoCs employ core processors as basic computational units in order to have highly flexibility for increasing amount of system functionality. However, the highly flexibility causes a waste of energy consumption which is important design parameter for SoCs design, especially for battery-powered applications such as PDAs (Personal Digital Assistants), cellular phones, and digital cameras. In a word, processors are energy-inefficient with respect to dedicated architectures[12]. One of the key issues in the design of energy-efficient processor-based architectures for embedded system is the energy consumed by memory access. Low-power memory organizations can greatly reduce the overall energy consumption of the system especially for data-dominant applications. Several researchers have pointed out that the energy consumption in memories can take a dominant fraction on the energy budget of a whole embedded system for data-dominated applications[3]. Embedded processor-based systems allow for customization of memory configuration based on application-specific requirements[9]. Memory size and organization can be tailored to application requirements, and application-specific memory architectures can be developed to minimize memory access energy for a given embedded application.

Many energy optimization approaches have specifically wrestled with the reducing memory energy in SoCs. Farrahi et al.[5] studied memory segmentation/portioning problem to exploit sleep mode operation for minimization of the average power consumption. Su and Despain[14], Ko et al.[8], and Shiue and Chakrabatry[13] studied power-efficient cache organizations. They identified cache sub-banking as an effective technique to reduce cache power consumption. Panda and Dutt[10] proposed low-power memory mapping based on access patterns to reduce the transitions on the address bus. This work focuses on off-chip memory accesses in which a significant amount of power is consumed in the address bus. Coumeri and Thomas[4] presented an environment for exploration of low-power on-chip SRAM organization. Ishihara and Yasuura[7] presented power reduction technique for instruction memories in application specific systems. Benini et al.[1] presented design methodology based on the idea of mapping the most frequently accessed data onto a small memory which can be placed very close to the processor. In [2], they also proposed an algorithm for the automatic partitioning of on-chip SRAM in multiple banks that can be independently accessed.

In this paper, we present a novel low energy memory access technique based on dynamically changing effective width of data, which called **Active Data Bitwidth**. Our technique can reduce the redundant access energy of on-chip data memory by exploring the active data bitwidth of data which is accessed. We focus on vertical partitioning of on-chip SRAM bank for low energy consumption. In embedded systems, the on-chip SRAM is a valid alternative to caches which well known an architectural optimization technique for memory design. Caches exploit the principle of locality in memory access patterns[6], and provide a flexible way to store most frequently accessed memory locations, where they can be efficiently read and written. However, storing data into a cache and retrieving data from it is much

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ISLPED'02, August 12-14, 2002, Monterey, California, USA.
Copyright 2002 ACM 1-58113-475-4/02/0008 ...$5.00.

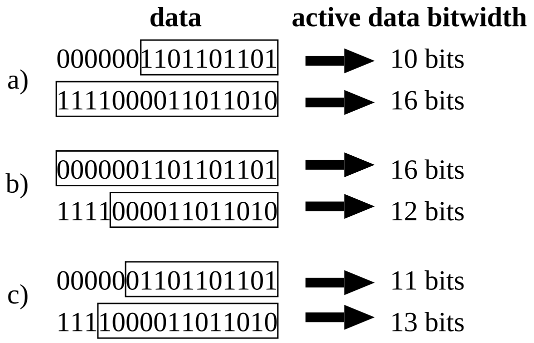

Figure 1: Example of Active Data Bitwidth

more energy consuming than accessing a memory containing the same amount of data because of cache misses. The on-chip SRAM is often called **Scratch-Pad Memory**[11]. In this architecture, the most frequently accessed addresses are statically mapped onto scratch-pad SRAM to guarantee power and performance efficiency. The main difference between the scratch-pad SRAM and data cache is that the SRAM guarantees a single-cycle access time, whereas access to the cache is subject to cache misses. In addition, scratch-pad SRAMs are particularly useful in real time embedded systems for data-intensive applications, where access patterns can be profiled and studied at design time, and where caches are known to perform suboptimally and to reduce predictability in real-time performance[2].

The rest of the paper is organized in the following way. Section 2 describes our motivation. Section 3 presents our low-energy memory access technique. Experimental results are shown in Section 4. At last Section 5 concludes our work and gives the future work.

2. MOTIVATION

2.1 Active Data Bitwidth

A 32-bit processor usually makes an operation using 32-bit data width. However, for an application, the effective data width called *active data bitwidth*, which is exactly used, sometimes is far less than 32bits. In run time, the data stored in memory change dynamically, therefore the active data width of the data also change dynamically.

We determine the active data bitwidth by the following three cases,

a) remains after deleting the continuous "0" in upper bits

b) remains after deleting the continuous "1" in upper bits

c) remains after changing the continuous "0" to only one "0", or remains after changing the continuous "1" to only one "1"

Figure 1 shows an example of active data bitwidth for data with 16bits data width. For one system, to define the active data bitwidth, only one case can be considered. In our technique, we use c) as the principle to determine the active data bitwidth. But we can select other case according to an application.

2.2 Memory Vertical Partitioning for Low Energy

Allocating more or fewer memories has an effect on the chip area and on the energy consumption of the memory architecture[9]. Large memories consume more energy per access than small memories, due to the longer word- and bit-lines. So the energy consumed by a single large memory containing all the data is much larger than when the data is distributed over several smaller memories.

When a single SRAM memory array is partitioned into several segments for low energy, the most frequently accessed data block is allocated a small memory in on-chip SRAM bank partitioning such as [2]. Then, the average power in accessing the memory hierarchy is decreased, because a large fraction of accesses is concentrated on a small, power-efficient memory. These techniques optimized memory energy by partitioning bit line or called horizontal partitioning. Different with those techniques, we optimize memory energy by partitioning word line (vertical partitioning), in which the data are stored in several segments separately while may be stored in a single memory if also using above techniques.

Figure 2 shows an example stored in a memory, of which word line is partitioned into four parts. When the processor reads the memory, it reads data from the memory by suited addresses of different segments. Then the read data are put on data bus after packed (shown in Junction block in Figure 2).

The innovation of this paper is that we present a memory access technique based on active data bitwidth, using which the access count of memory is reduced by accessing only necessary memory segment.

In the example of Figure 2, when a read access is performed, if the data have 10-bit active data bitwidth, only the two memory segments of lower bits are accessed, and the data are put on data bus after encoded, which result in energy reduction of redundant memory access dynamically.

In order to use active data bitwidth, we should store it. We use a memory called active size recorder, which is set in select block. Figure 2 shows the example, which needs an active size recorder with 2-bit $\times N$ size. We need to account for the energy consumed in the entire partitioned memory system. i.e., the address, data buses, the active size recorder and the control signals. These components introduce a no-negligible overhead on energy consumption that must be offset by the savings given by our technique. A dynamic access profile obtained through instruction-level simulation for an embedded multimedia application shows that the active data bitwidth of some 32-bit data are less than 16bits in most time. This means that the energy overhead caused by accessing active size recorder is far less than the energy saving by only accessing the memory segments determined by active data bitwidth.

When a address of memory is read/written, each memory segment is controlled as the following,

Read Operation

(R-1) Read an active segment size for the target address *addr* from the active size recorder.

(R-2) Read an effective part of data for the target address *addr* from memory segments corresponding to the active segment size.

(R-3) Lengthen the effective part of data by sign extension

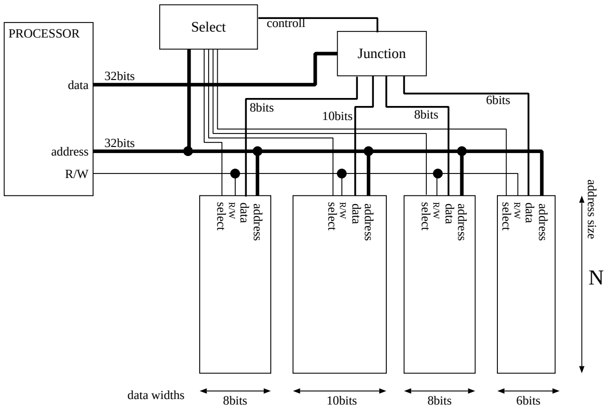

Figure 2: Memory Partitioning Example

until data bus width.

Write Operation

(W-1) Compute minimum active segment size including active data bitwidth of written data to the target address *addr*.

(W-2) Write the effective part of data to the target address *addr* for memory segments corresponding to the active segment size.

(W-3) Write the active segment size to the target address *addr* of the active size recorder.

For write operation, (W-2), (W-3) can be performed by parallel process.

Our technique has some overhead of performance (R-1), but it is less than those techniques by partitioning memory word line. In fact the overhead of performance induced by our technique is little. Also, our technique looks like Asanovic's dynamic zero compression[15]. Their technique adds an additional *zero indicator bit* to each cache byte that indicates whether the byte contains all zero bits. Our technique considers not only all zero but also all one. Additionally, we can make data width of a segment into any size.

3. LOW ENERGY MEMORY ACCESS

3.1 Memory Model

Because a static random access memory (SRAM) does not require additional fabrication steps and dedicated technology, it can be easily integrated onto the same chip with the processor and other logic circuits. Therefore, embedded SRAMs are much more common in SoC design than no-volatile memories and DRAMs.

For memories, we assume that the power dissipated for charging the global bit line is in proportional to the number of partitioned segments, and the power dissipated in a single segment is in proportional to the size of the segment. Under these assumptions, the memory energy consumption can be approximated by formula (1), where N_{seg}, N_{word}, α, β, and γ denote the number of segments, the number of words, and coefficients for each term, respectively.

$$E_{mem} = \alpha \cdot N_{seg} + \beta \cdot \frac{N_{word}}{N_{seg}} + \gamma \tag{1}$$

In formula (1), the first and second terms represent the energy dissipated for charging the global bit line and the energy dissipated in a single memory segment respectively. The last term represents a constant factor in memory energy consumption. From formula (1), it is easy to derive that the number of memory segments which minimizes the memory power consumption is $\sqrt{(\beta/\alpha) \cdot N_{word}}$. We generated some SRAM models by Alliance CAD System $Ver.4.0$ with $0.5\mu m$ double metal CMOS technology, and using the SPICE simulation of these memories with the different configurations, we obtained the estimation models of SRAM as follows:

$$\begin{aligned} e_r(x) &= 24.9 \cdot x \cdot \sqrt{N_{word}} + 56 \; [pJ/cycle] && (2) \\ e_w(x) &= 197 \cdot x \cdot \sqrt{N_{word}} + 369 \; [pJ/cycle] && (3) \end{aligned}$$

where e_r and e_w express the access energy of memories for read and write operations respectively. In addition, x is the bit width of memory and N_{word} is the number of words.

3.2 Application-Specific Exploration

Using our technique, we can customize data memory by the profile data of a given application from instruction level simulator. The customizable parameters are listed as follows,

- N_{seg}: the number of memory segments
- b_i: the bit width of memory segment i $\quad (b_i \geq 1)$

The customization must satisfy the formula (4).

$$\sum_{i=1}^{N_{seg}} b_i = D_{width} \tag{4}$$

where D_{width} is the data path width of processor (equal to data bus width). In this case, the size of active size recorder is $\lceil log_2 N_{seg} \rceil \times N_{word}$. N_{word} is the number of words for each segment.

The following items are determined based on profile data.

- N_{access}: the number of total memory access counts.
- $\delta_r(i)$: the function, which return "1" when it is a read access, return "0" when it is a write access, for the ith memory access;
- $data(i)$: the value read or written for the ith memory access

We perform the application-specific exploration to optimize memory access energy by using the profiled data. The formula (9) is used as the evaluation model.

$$\begin{aligned} E_r(i) &= \delta_r(i) \cdot \left\{ \sum_{j=1}^{act(i)} e_r(b_j) + e_r(b_{rec}) \right\} && (5) \\ E_w(i) &= (1 - \delta_r(i)) \cdot \left\{ \sum_{j=1}^{act(i)} e_w(b_j) + e_w(b_{rec}) \right\} && (6) \\ act(i) &= \min\{x | \sum_{j=1}^{x} b_j \geq data(i).abw\} && (7) \\ b_{rec} &= \lceil \log_2 N_{seg} \rceil && (8) \\ E_{total} &= \sum_{i=1}^{N_{access}} \{E_r(i) + E_w(i)\} && (9) \end{aligned}$$

Table 1: Experimental Results

application	Our Technique				Monolithic			Memory Banking		
	N_{word}	E_r E_w	E_r^{rec} E_r^{rec}	total	E_r E_w	total	Our Saving	E_r E_w	total	Our Saving
mpeg2play	302834	0.75J 0.59J	0.14J 0.74J	2.22J	2.3J 11.8J	14.1J	84.2%	1.08J 5.43J	6.51J	65.9%
ijpeg	917987	3.16J 4.36J	0.37J 0.76J	8.65J	5.94J 12.17J	18.11J	52.2%	4.58J 5.28J	9.86J	12.3%

where $E_r(i), E_w(i)$ are the energy for the ith memory read access and write access respectively. $act(i)$ is the active segment size of the ith memory access. $data(i).abw$ is the active data bitwidth of $data(i)$, b_{rec} is bit width of active size recorder.

4. EXPERIMENTAL RESULTS

We present some exploration results on several embedded application programs. Our target is to customize memory suitable to a given application.

Table 1 shows the results of the experiments employed our low-energy memory design technique based on active data bitwidth. To illustrate the effectiveness of our technique, we compare the experimental results to not only monolithic memory, but also memory designed by banking technique, which is usually used by most of memory designers for real embedded system design. We use two real embedded applications as benchmarks. The profile data are got by using Simplescalar Simulator, and we used energy model in section 3.1. In Table 1, the first 4 columns *our technique* show the results using our technique. N_{word} is the number of words in memory and E_r, E_w are the energy consumption for read/write access respectively. E_r^{rec}, E_w^{rec} are the read/write energy consumption for the active size recorder respectively. *total* shows the total energy consumption of memory. The next 3 columns show the results for monolithic memory, in which our saving means the results using our technique compared to monolithic one achieved. The last 3 columns are the results for memory banking technique, in which our saving means the results using our technique compared to memory banking technique achieved. Experimental results shows the dramatic energy reduction up to 84.2% compared to the monolithic memory and 65.9% compared to memory banking technique.

5. CONCLUSION

This paper presents a novel low-energy memory access technique based on active data bitwidth. By means of profiling the active data bitwidth dynamically, we present a novel low-energy memory access technique for on-chip data memory design. By reducing the redundant access energy of data memory using our technique, the total energy consumption of memory is decreased drastically. Our experimental results show that we can achieve significant energy reduction. Compared to the monolithic memory, for JPEG, 52.2%; for MPEG-2 84.2%, the energy reduction is reported. Compared to the memory banking technique (horizontal partitioning), 12.3% energy reduction for JPEG and 65.9% for MPEG-2 is achieved. Future work is to present a new algorithm for customize the memory automatically.

6. REFERENCES

[1] L. Benini, A. Macii, E. Macii, and M. Poncino. “Synthesis of Application-Specific Memories for Power Optimization in Embedded Systems”. In *Proc. of 37th DAC*, pages 300–303, June 2000.

[2] L. Benini, A. Macii, and M. Poncino. “A Recursive Algorithm for Low-Power Memory Partitioning”. In *Proc. of International Symposium on Low Power Electronics and Design*, pages 78–83, August 2000.

[3] F. Catthoor, S. Wuytack, E. D. Greef, and F. Balasa. *Custom Memory Management Methodology: Exploration of Memory Organization for Embedded Multimedia System Design.* Kluwer, 1998.

[4] S. L. Coumeri and D. E. Thomas. “An Environment for Exploring Low Power Memory Configurations in System Level Design”. In *Proc. of ICCDesign*, pages 348–353, September 1999.

[5] A. H. Farrahi, G. E. Tellez, and M. Sarrafzadeh. “Memory Segmentation to Exploit Sleep Mode Operation”. In *Proc. of 32th Design Automation Conference*, pages 36–41, June 1995.

[6] J. Hennessy and D. Patterson. *Computer Archtecture A Quantitive Approach.* Morgan Kaufman, 1996.

[7] T. Ishihara and H. Yasuura. “A Power Reduction Technique with Object Code Merging for Application Specific Embedded Processors”. In *Proc. of DATE*, pages 617–622, March 2000.

[8] U. Ko and P. Balsara. “Energy Optimization of Multilevel Cache Architectures for RISC and CISC Processors”. *IEEE Transanctions on VLSI Systems*, 6(2):pages 299–308, Jun 1998.

[9] P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, and A. Vandercappelle. “Data and Memory Optimization Techniques for Embedded Systems”. *ACM Transactions of Design Automation of Electronics Systems*, 6(2):pages 149–206, April 2001.

[10] P. R. Panda and N. D. Dutt. “Low-Power Memory Mapping Through Reducing Address Bus Activity”. *IEEE Transanctions on VLSI Systems*, 7(3):pages 309–320, September 1999.

[11] P. R. Panda, N. D. Dutt, and A. Nicolau. “On-Chip vs. Off-Chip Memory: The Data Partitioning Problem in Embedded Processor-Based Systems”. *ACM Transactions of Design Automation of Electronics Systems*, 5(3):pages 682–704, July 2000.

[12] J. Rabaey and M. Pedram. *Low Power Design Methodologies.* Kluwer, 1996.

[13] W.-T. Shiue and C. Chakrabarti. “Memory Exploration for Low Power Embedded Systems”. In *Proc. of 36th Design Automation Conference*, pages 140–145, June 1999.

[14] C. Su and A. Despain. “Cache Design Tradeoffs for Power and Performance Optimization: A Case Study”. In *Proc. of International Symposium on Low Power Electronics and Design*, pages 63–68, April 1995.

[15] L. Villa, M. Zhang, and K. Asanovic. “Dynamic Zero Compression for Cache Energy Reduction”. In *Proc. of MICRO-33*, 2000.