An Application-Specific Memory Partitioning Method for Low Power

Songping Mai[1*], Chun Zhang[2], Yixin Zhao[2], Jun Chao[2], Zhihua Wang[2]
[1] Department of Electronic Engineering, Tsinghua University, Beijing 100084, P. R. China
[2] Institute of Microelectronics, Tsinghua University, Beijing 100084, P. R. China
*Email: msp03@mails.tsinghua.edu.cn

Abstract
Memory consumes large portion of power in today's digital signal processors (DSPs). This paper proposes a memory partitioning method for power reduction. Based on dynamic execution profile of application-specific programs running on a given DSP, a multi-banked on-chip memory architecture well fitted to the application is developed. With a practical and maneuverable process, this method can achieve a near-optimum solution for low power. Experimental result shows that it can save about 50% of the power at best with little area overhead.

1. Introduction

Digital signal processors (DSPs) are very demanding in terms of memory as they require memory for both fetching instructions and manipulating data. DSP with modified Harvard architecture is a typical case: it usually contains three memories, one for storing instruction and the other two for storing data. These memories are usually large volume of on-chip SRAMs and greatly speed up the data processing especially in those data-dominant applications. However, this decisive advantage of intensive memories is counter-balanced by their tremendous power dissipation. How to minimize the power dissipation of memories thus becomes a key issue in today's DSP design.

Memory partitioning is an effective method among various power optimization approaches specifically tackling the memory power challenge[1]. To reduce power, a single large-volume memory can be properly divided into several small-volume sub-banks that can be independently accessed according to application requirements. For each memory access, only the addressed sub-bank is activated and the others keep inactive. The overall switched capacitance per access is decreased and therefore the power reduction can be achieved. Power dissipation decreases with the size of sub-banks. Unfortunately, there are some drawbacks to this method: (1) Silicon area overhead caused by duplication of addressing and control logic of sub-banks; (2) Place and route trouble in layout design because of too many segmented micro-blocks. These drawbacks prevent arbitrary partitioning and advance the problem of how to get an optimum partition for low power.

Several authors have discussed this problem in depth and focused to develop automatic techniques[2-5]. The main idea of the optimal partitioning is to profile the memory access patterns to find out the most frequently accessed addresses and try best to map these addresses onto a small-volume scratch-pad RAM. However, the optimal partitioning algorithms are usually very complicated and require lots of parameters. In fact, if we consider the actual constraints in practical designs and ignore some trivial factors, the partitioning process will become much simpler and can be handled by hand. With regard to realistic conditions, a wieldy partitioning method is proposed in this paper based on a practical design.

This paper is organized as follows. In Section 2, the design specifications are given. Section 3 describes the memory partitioning method in detail. The experimental result is presented in Section 4. And a summary is finally given in Section 5.

2. Design specifications

2.1 A DSP with typical core-RAM structure

The design used in this paper is a 24-bit fixed point DSP with typical core-RAM structure. The block diagram of the DSP is shown in figure 1. Its memory is composed of three independent SRAMs: an instruction RAM (PRAM) and two data RAMs (XRAM and YRAM). This DSP is intended for cochlear implant application[6]. All the RAMs are dual-port since data sampling and processing with memory access requirement are performed simultaneously. A coarse analysis of gate-level power consumption shows that the memory consumes about 60% of the total power. Therefore, to reduce the power of the memory becomes a primary goal of the DSP design.

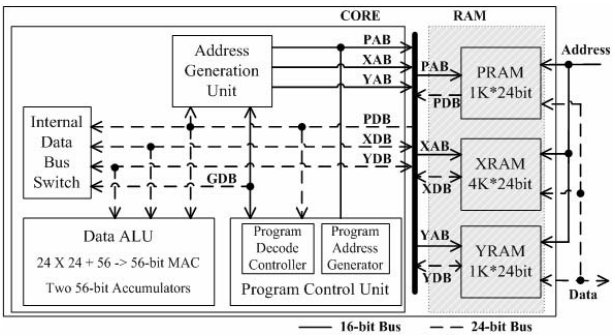

Figure 1. Block digram of the DSP

2.2 Application definition

The memory partitioning method discussed in this

1-4244-1132-7/07/$25.00 © 2007 IEEE

paper is application-specific, so a definite application is essential. As mentioned above, application programs running on the DSP core are speech processing algorithms dedicated to cochlear stimulation, such as CIS, ACE and so on[7]. Filter bank is the key component of this kind of programs, which consumes most of the computing time and energy.

Four application programs are chosen to represent the cochlear speech algorithms: a 64-tap FIR filter (*fir*), a 64-point FFT (*fft*), a 64-tap adaptive least mean square filter (*lms*) and a 128-point CIS (*cis*). The former three are common algorithms wildly used in signal processing and the last is a typical cochlear stimulating algorithm.

2.3 SRAM model and power analysis flow

The SRAM used in the design is Artisan's IP model based on UMC 0.18um standard CMOS process. SRAM IP with behavior model, gate-level model and layout view can be generated by Artisan's RAM compiler according to user's specification[8]. There are some constraints on the RAM size, among which the most important is that RAM size should be larger than 32-word. This is the practical foundation of our simplified memory partitioning method.

Power analysis on the SRAM is performed by Synopsys's PrimePower[9]. The dynamic power is derived from the switching activity of the test bench and the static power is extracted from the SRAM model. The analysis accuracy depends on the SRAM model. For commercial IP library, the error compared with practical measurement result is generally within 10%.

3. Memory partitioning for low power

3.1 Premise and practical observation

Although the RAM compiler can provide various sizes of RAM model, the premise below is used: all sub-banks are with the size of 2^i words. There are several benefits with this premise: (1) The addressing and control logic of sub-banks is simplified, making its area and power overhead neglectable; (2) Arbitrary sizes of sub-bank segments are avoided, facilitating the layout design; (3) Search space of the optimal algorithm is greatly shrunk, making the hand-craft partitioning feasible. Although the third point might prevent designers from obtaining the globally optimum solution, measures can be taken to compensate it. Anyway, this premise enables designers to hand-craft the partitioning process and to get the near-optimum result with little time and effort, which is validated by the experimental result later on.

Since the largest size of the memory block (XRAM) is 4K words and the smallest size that the RAM compiler can provide is 32 words, the sizes of the sub-banks are thus limited to 32, 64, 128, 256, 512, 1024, 2048 and 4096. Simulation data about access power and physical area of RAMs of these sizes are shown in table 1. Due to the energy consumed by those deselected sub-banks, energy per access of the partitioned memory is a little higher than that of the activated sub-bank.

An obvious fact in the table 1 is that energy consumed per access monotonically increases with sub-bank size while total area of partitioned memory of a certain volume changes inversely with sub-bank size. For example, if a 4K-word RAM is divided into 128 pieces of 32-word RAMs, the energy consumed by a write access decreases by 50% from 282.9 μJ to 142.4 μJ, while the physical area increases by 7.87 times.

For memory of a certain volume, the optimum partitioning method for the lowest power is to divide it into pieces of the smallest sub-banks, while the optimum partitioning method for the smallest area is to keep it as a monolithic bank. However, neither is a good choice for a practical design because attention should be paid to both power and area. There are two ways to constrain the partitioning: (1) Give weight factors to power and area respectively and work out the minimum value of the cost function with weighed power and area; (2) Give a maximum area and find out the minimum value of the

Table 1. Access power and physical area of RAMs with volume of 2^i words

Size of memory sub-bank (bit)	Monolithic Memory			Partitioned Memories of 4K-word				
	Energy per Read (μJ)	Energy per Write (μJ)	Area (μm^2)	Number of sub-banks	Energy per Read (μJ)	Energy per Write (μJ)	Total Area (μm^2)	Area Ratio
4096 x 24	235.2	282.9	1435740	1	235.2	282.9	1435740	1
2048 x 24	174.9	219.9	825940	2	175.7	222.1	1651881	1.15
1024 x 24	149.4	193.2	531337	4	150.5	194.1	2125350	1.48
512 x24	135.3	167.4	302511	8	136.5	168.6	2420088	1.68
256 x 24	126.6	153.1	189702	16	127.9	154.7	3035237	2.11
128 x 24	122.3	145.8	130692	32	123.6	147.2	4182149	2.91
64 x 24	120.0	142.2	103024	64	123.1	143.8	6593550	4.59
32 x 24	119.0	140.5	88323	128	121.2	142.4	11305399	7.87

power function under the given area. The second method is adopted in our partitioning process because it is more practical as a maximum area constraint is usually much easier to obtain than weight factors.

3.2 Locality of instruction and data reference

General DSP programs abide by the locality of reference[10]: (1) Temporal locality — Recently accessed data are likely to be accessed in the near future; (2) Spatial locality — Data whose addresses are near one another tend to be referenced close together in time. This locality property is wildly exploited in cache design. It can also be used to optimize the memory organization.

The locality is true for application programs running on our DSP: (1) For PRAM which stores the instruction code, application programs are usually full of short loops and ranges of continuous addresses containing loop code are repeatedly accessed; (2) For XRAM which stores sampling points and delay lines, addresses of delay lines are also repeatedly accessed; (3) For YRAM which stores filter coefficients, addresses of coefficients are constantly referenced in order.

An implication of the locality is that, if the memory access patterns are profiled, locality of the frequently accessed addresses can be observed. In another word, if the memory addresses are grouped according to their accessed frequency, we can find that addresses in one group are adjacent to each other. This kind of data access regularity helps to improve the memory partitioning.

3.3 Partitioning method in generality

Taking the area requirement and the data locality into account, a partitioning method can be developed as follows:

(1) Profile the memory access patterns and divide the logical memory addresses into N groups according to their accessed frequency, where the group number changes inversely with the accessed frequency, i.e. the *1*st group is the most frequently accessed and the *N*th group is the least frequently accessed. Addresses in one group will turn out to be continuous because of the data locality.

(2) Partition the physical memory into N sub-banks and guarantee $A_1 \leqslant A_2 \leqslant \cdots \leqslant A_N$ and $\sum_i A_i \leqslant A_{max}$, where A_i is the area of *i*th sub-bank and A_{max} is the given maximum area.

(3) Revise the application programs and map the logical address groups into physical memory sub-banks, making addresses in group *i* a subset of sub-bank *i*.

The last step seems quite troublesome. In fact, it doesn't take much effort thanks to the data locality. What designers needs to do is just to match the initial addresses of the groups with the beginning addresses of the corresponding sub-banks. Once these two kinds of addresses are aligned, the mapping is automatically finished. In this process, only some minor changes on branch addresses and data initial addresses are required.

3.4 Memory organization in practical DSP design

While the above partitioning method is adopted for all memories, there are still some special considerations on the sub-bank number and the address sequence for different types of memories. Below is the detailed partitioning result for the three RAMs in our DSP:

(1) PMEM: Instruction memory is full of loops that are usually nested. For example, the core of the *fft* program is a triple nested loop and the *cis* program contains loops hexad-nested. This observation tells us that the PMEM should be divided into as many small sub-banks as possible to accommodate various nested loops. With the maximum area limited, the partitioning result of PMEM is shown in figure 2(a). A 128-word bank is placed foremost to hold the initial code and interrupt vectors that are not frequently accessed. The following are four 32-word banks for loops, the smallest sub-banks for the most frequently accessed addresses. Then is a 256-word bank for less frequently accessed addresses. And aftermost is a 512-word bank for extension, which is seldom accessed for most of application programs.

(2) XMEM: There are three kinds of data — Raw sampling data, processed data and delay lines. Delay lines are the most frequently accessed items. The raw data and processed data are usually accessed only once. Besides, all the processing algorithms we used are performed on series of 64 points or 128 points. With all these considerations, the partitioning result of XMEM is shown in figure 2(b).

(3) YMEM: There are two kinds of data — Coefficients and temporary variables. Coefficients are accessed much more frequently than temporary variables. A simple partition of YMEM is shown in figure 2(c).

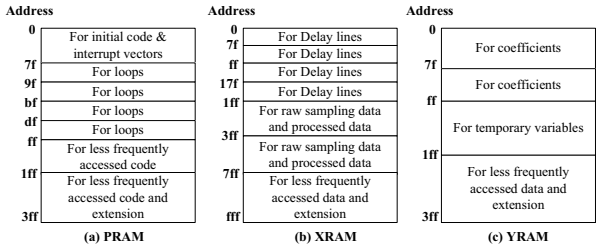

Figure 2. Partitioning result of the three RAMs

4. Experimental result

In our experiment, the proposed memory partitioning method (PMP) is compared with two other methods of memory organization. The first is the smallest-area organization method (SAO) that keeps the memory as a monolithic bank. The second is the lowest-power partitioning method (LPP) which divides the memory into smallest sub-banks. The contrasted power result with test

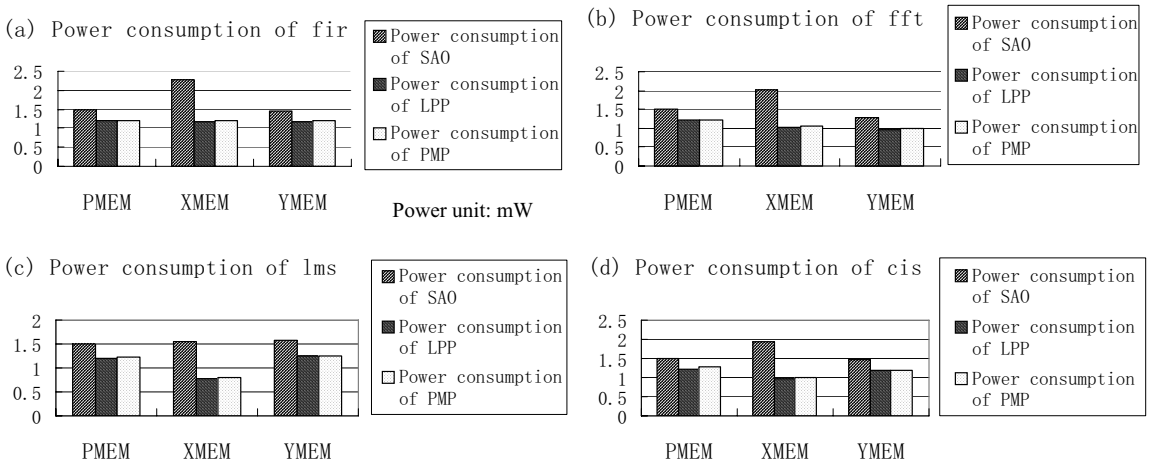

Figure 3. Power consumption of different application programs

bench *fir*, *fft*, *lms* and *cis* running on the DSP at 10MHz are given in figure 3. And the contrasted area result is shown in figure 4. It can be concluded from the results that (1) the proposed method achieves low power result very close to the lowest value of LPP; (2) the area overhead of the proposed method is less than 2 times of that of the SAO, which is quite economical compared with area overhead of the LPP that is about 6 times of the SAO; (3) the proposed solution works smoothly with various types of application programs. In a word, the proposed method achieves very low power with little area overhead. Another outcome of the experiment is that the XRAM saves much more power than the other two RAMs. It is just the potential of the partitioning method: the larger the volume of memory, the more power can be reduced.

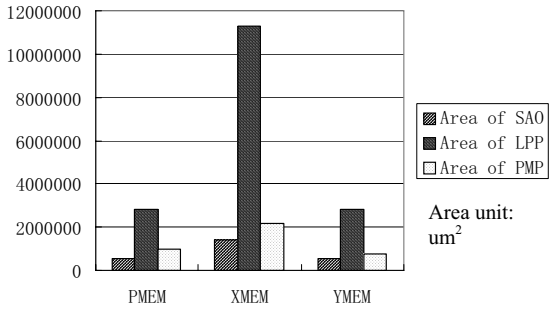

Figure 4. Area overhead of three methods

5. Summary

An application-specific memory partitioning method is introduced with some realistic considerations. Based on this method, a on-chip memory structure for a DSP dedicated to cochlear implant application is developed. The partitioning process is maneuverable and can be carried out by hand. Experimental result shows that this method can achieve near-optimum result for low power at the little cost of additional silicon area.

Acknowledgment

This work is partly supported by National Natural Science Foundation of China (No. 60475018 and No. 60506007).

References

[1] J. M. Rabaey, A. Chandrakasan and B. Nikolic, Digital integrated circuits: a design perspective, 2nd Ed., Prentice-Hall international Inc., p.701-707 (2004).

[2] F. Angiolini, L. Benini, and A. Caprara, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 24, p. 1660-1676 (2005).

[3] Qingfeng Zhuge, E.H.-M. Sha, Bin Xiao; C. Chantrapornchai, IEEE Transactions on Signal Processing, Volume 52, p. 1090-1099 (2004).

[4] S. Krishnamoorthy, U. Catalyurek, J. Nieplocha, A.Rountev, P. Sadayappan, ACM/IEEE Supercomputing Conference, p.34-36 (2006).

[5] L. Benini, A. Macii and M. Poncino, International Symposium on Low Power Electronics and Design, p.78-83(2000).

[6] Songping Mai, Chun Zhang, Mian Dong and Zhihua Wang, Acoustics, IEEE International Conference on Speech and Signal Processing, p. V125-V128 (2006).

[7] P. C. Loizou, IEEE Signal Process Magazine, Vol. 15, p. 101-130(1998).

[8] Artisan Components, Inc., Artisan standard library 0.13um-0.25um SRAM generator user manual(2004).

[9] Synopsys Inc., PrimePower Manual(2006).

[10] J. L. Hennessy and D. A. Patterson, Computer Architecture: A Quantitative Approach, 3rd Ed., Beijing: China Machine Press(2002).