POWER EXPLORATION OF PARALLEL EMBEDDED ARCHITECTURES IMPLEMENTING DATA-REUSE TRANSFORMATIONS

N. Kavvadias[1], A. Zanikopoulos[1], Ch. Voliotidis[1], S. Kougia[1], A. Chatzigeorgiou[1], N. Zervas[2], S. Nikolaidis[1]

[1]Electronics and Computers Div., Department of Physics
Aristotle University of Thessaloniki, 54006 Thessaloniki, Greece
[2]VLSI Design Laboratory, Department of Electrical Engineering
University of Patras, Patras 26500, Greece
Email address: snikolaid@physics.auth.gr

ABSTRACT

Efficient use of data-reuse transformations combined with a custom memory hierarchy that exploits the temporal locality of data related memory accesses can have a significant impact on system power consumption, especially in data dominated applications e.g. multimedia processing. In this paper the effect of data-reuse decisions on power consumption, area and performance of multimedia applications implemented on uni- and dual- processor embedded cores is explored. By this work it is clarified that conclusions for the transformations effect on multi-processor architectures can be extracted by the corresponding effect on the uni-processor architecture. In this way the exploration space can be significantly reduced. A motion estimation algorithm, namely the two-dimensional logarithmic search, and a discrete cosine transform (DCT) algorithm are used as demonstrator applications.

1. INTRODUCTION

With the emergence of portable multimedia applications energy consumption has been promoted to a major design consideration [1] due to the requirements for long battery life, large integration scale and the related cooling and reliability issues [2]. Consequently, significant research effort is devoted to the development of design strategies, especially in higher design levels, where the largest savings can be achieved.

A formalized methodology for data-reuse exploration, in order to reduce the system power consumption of data-dominated applications has been proposed [2][3]. The exploitation of data-reuse transformations points to a specific memory hierarchy where copies of data signals from larger memories that exhibit high data-reuse are stored to additional layers of smaller and less power consuming memories [2]. By exploiting the temporal locality of data memory references [3], the largest part of the data memory accesses that are conducted on power-hungry off-chip memories is moved to smaller on-chip memories and significant power savings can be obtained [2].

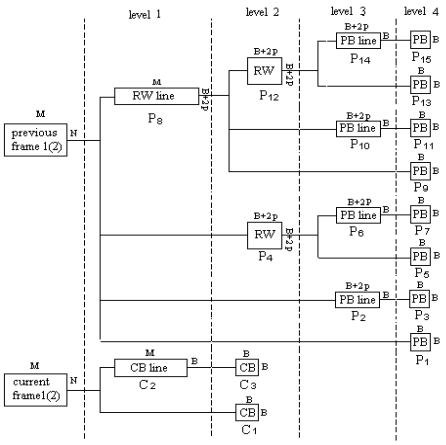
Related work for partitioned multimedia algorithms has been proposed in [2][4]. Some experimental results on power, area and performance for the case of multiprocessor embedded architectures have been given in [6].

In this paper we explore the effect on power, performance and area of data-reuse transformations for the case of uni- and dual-processor embedded architectures. As demonstrators, a fast motion-estimation algorithm, namely the two-dimensional logarithmic search motion estimation algorithm [5], and a typical row-column decomposition DCT algorithm have been used. The experimental results show that each data-reuse transformation has similar effect on power, performance and area whether it is applied to single-processor or dual-processor architectures. Finally it is proved that the data-reuse decision should be carried out at an early stage of the design hierarchy, i.e. prior to the partitioning step.

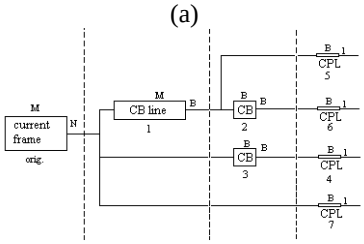
2. DATA REUSE TRANSFORMATIONS

For motion-estimation like algorithms, the possible data-reuse transformations with the introduced levels in the memory hierarchy [3] are shown in Fig. 1. The parameters for these algorithms are: the size of the current and previous frame ($N \times M$), block size ($B \times B$) and the search region size $[-p,p]$ around the location of the specific block in the current frame. These transformations involve memories for a line of reference windows (RW line), a reference window (RW), a line of candidate blocks (PB line), a candidate

This work was supported by the ED 501 PENED'99 project funded by G.S.R.T. of the Greek Ministry of Development an the European Union.



(a)



(b)

Fig. 1: Memory hierarchy levels and corresponding transformations for (a) the three-step search algorithm, (b) the DCT algorithm

block (PB), a line of current blocks (CB line), a current block (CB) and a current block pixel line (CP line).

3. TARGET ARCHITECTURE

Concerning the data memory organization an application specific data memory architecture (ASDMA) is assumed [2]. In the uni-processor architecture each memory layer communicates with the processor through a common bus. Since the main focus is on parallel processing systems, the flexibility of using distributed or shared memory layers imposes the mapping of the transformed algorithm onto three different memory architectures [4]: distributed memory architecture (DMA), shared memory architecture (SMA), and shared-distributed memory architecture (SDMA). Every memory layer in these partitioning architectures is of the same size as the corresponding layer of the single processor architecture.

In the DMA architecture, each processor core has its own memory hierarchy, which depends on the applied transformation. With shared memory architecture (SMA) all memory levels are common for the two processors. In the SDMA scheme, the higher levels of memory hierarchy are common, while the lower levels are distributed.

4. DATA-REUSE TRANSFORMATIONS EXPLORATION

To illustrate the effect of data-reuse transformations (Fig. 1) on power consumption, a single-processor and a dual-processor platform have been simulated [8]. Typical values for the algorithmic parameters have been used [5]: $N \times M$=144×176, B=16, p=7 for the three-step logarithmic search and $N \times M$=144×176, B=8 for the DCT algorithm.

4.1. Results for energy consumption

The total data-memory energy consumption for a given memory hierarchy is calculated by the sum of the energy consumption of every memory layer included in that hierarchy:

$$E_{d\_total} = \sum_i f_i F(S_i, Nr\_ports_i) \quad (1)$$

where the power consumed due to accesses in *i-th* memory layer, is proportional to the number of accesses, f_i, and depends on the size, S_i, the number of ports, Nr_ports_i, of the memory, the power supply and the technology.

For the case of the single processor, (1) becomes:

$$E_{d\_single} = \sum_i f_i F(S_i, 1) \quad (2)$$

since Nr_ports_i=1 for every memory layer. For the distributed architecture the energy consumption is:

$$E_{d\_distr} = \sum_i [f_{1i} F(S_i,1) + f_{2i} F(S_i,1)] = \sum_i [f_{1i} + f_{2i}] \cdot F(S_i,1) \quad (3)$$

where indexes 1 and 2 denote the processors. According to (3), (f_{1i}+f_{2i}) is the number of total accesses for the two processors in i-*th* memory layer. For the DMA architecture, for obvious reasons, it holds:

$$E_{d\_distr} = \sum_i (f_{1i} + f_{2i}) F(S_i,1) = \sum_i f_i F(S_i,1) = E_{d\_single} \quad (4)$$

In the case of the SMA, the sum of the accesses of the two processors to each memory is equal to the number of accesses of the single processor to that memory. The energy consumption for SMA is given below:

$$E_{d\_shared} = \sum_i [f_{1i} F(S_i,2) + f_{2i} F(S_i,2)] = \sum_i (f_{1i} + f_{2i}) F(S_i,2) = \sum_i f_i F(S_i,2) \xrightarrow{F(S_i,2) > F(S_i,1)} E_{d\_shared} > E_{d\_distr} \quad (5)$$

In case of the SDMA, the same as before holds for the accesses, while the energy consumption lies between the two other cases:

$$E_{d\_single} = E_{d\_distr} < E_{d\_shar-distr} < E_{d\_shared} \quad (6)$$

which can be clearly observed from the results in Fig.2.

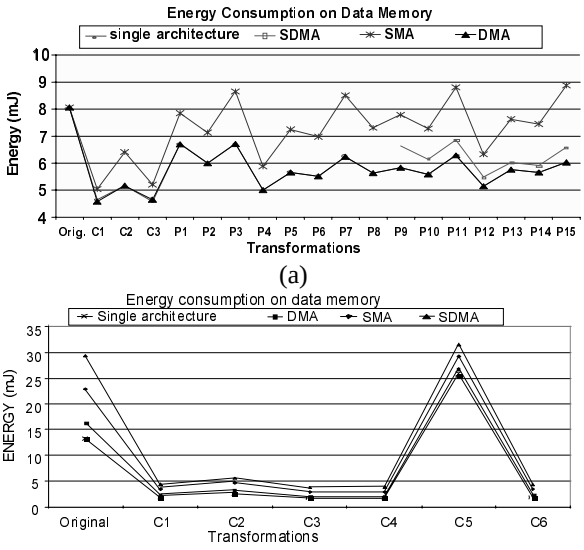


Fig. 2: Energy Consumption on Data Memory, (a) the motion estimation algorithm, (b) the DCT algorithm

When only data memory power consumption is considered, transformation C_3 and P_4 provide the optimal solution for the current and previous frames respectively, for the motion estimation algorithm.

Regarding the instruction memory energy consumption, the relation between the number of executed instructions in each architecture is given by:

$$f_{1,2}^{SMA} > f_{1,2}^{SDMA} > f_{1,2}^{DMA} > (1/2)\, f_{total}^{Single} \qquad (7)$$

Results obtained by simulation, for the energy consumed on the instruction memory, are given in Fig. 3. For the particular class of algorithms, this energy component is significantly greater than the data related.

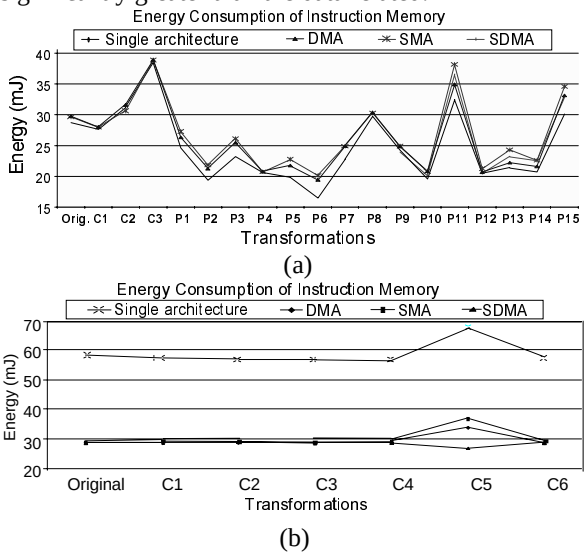


Fig. 3: Energy Consumption of Instruction Memory

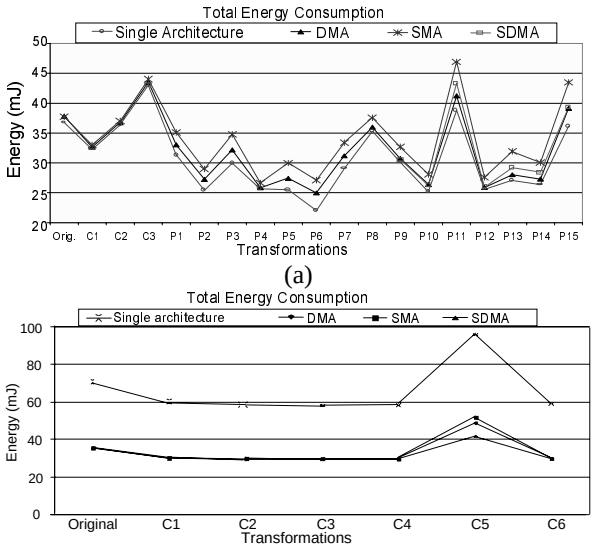


Fig. 4: Total Energy Consumption

In Fig. 4 the total power consumption is shown. The DMA seems to be the most energy efficient of the parallel ones according to (6), (7) and the experimental results. The SMA is the most power costly since it consists of dual-port memories resulting in higher energy cost per access, while the SDM stands between the two extreme cases. Note that for the previous frame transformation P_4 is no longer the most power efficient and the best solution is achieved by transformation P_6, proving the necessity to consider the power consumption due to instruction memory accesses. Finally, from Fig. 4 it is observed that the relative effect of each transformation on the total energy remains unaffected by the number of processors and the memory architectures.

4.2. Results on area

The area occupied by data memory elements is shown in Fig. 5 for both algorithms. Only on-chip memory elements are considered. We should first note that the introduction of additional data memory layers comes with an inevitable area penalty. It is also obvious that the distributed memory hierarchy is the most inefficient in terms of data memory area, since the on-chip memory modules occupy twice as much area than the single processor case. Moreover, it is less area efficient than the shared architecture, since two single port memory blocks occupy more area than a single dual-port memory. Shared-distributed architecture lies in between since it employs separate single-port memory blocks for the higher levels and dual-port memory blocks for the lower levels.

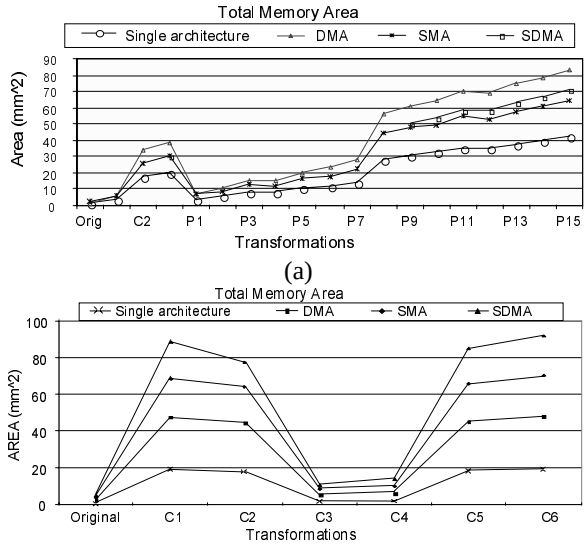


Fig. 5: Total Data Memory Area

4.3. Results on performance

In Fig. 6 performance is defined as the total number of required clock cycles for processing a frame. Since all partitioned architectures have somewhat similar performance, the selection of the most appropriate code transformation and memory architecture should be based mainly on energy and area criteria.

5. CONCLUSIONS

In this paper, the effect of data-reuse transformations on multimedia algorithms implemented on single- and multi-processor architectures, is explored. A number of code transformations that aim at moving background data memory accesses to smaller foreground memories, which are less energy costly has been applied to two popular multimedia algorithms. The effect of these transformations on energy consumption, performance and data memory area has been investigated for three data memory architectures: distributed, shared, and shared-distributed. Experimental results prove that significant energy reduction and performance boost can be achieved. It is also concluded that the relative effect of each transformation on energy and performance remains unaffected by the number of processors and the memory architecture. Consequently, full exploration can be performed on a uni-processor architecture, minimizing the required exploration space.

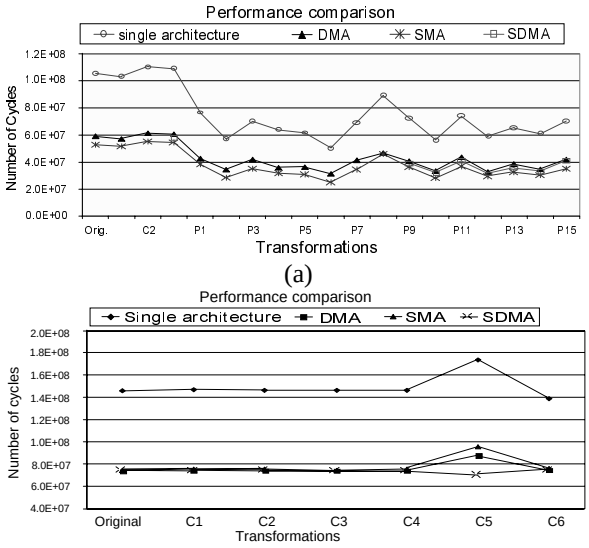


Fig. 6: Performance comparison

6. REFERENCES

[1] A. Chandrakasan and R. Brodersen, "*Low Power Digital CMOS Design*", Kluwer Academic Publishers, Boston, 1995

[2] F. Cathoor, S. Wuytack et al., *Custom Memory Management Methodology*, Kluwer Academic Publishers, Boston 1998.

[3] S. Wuytack, J.Ph. Diguet, F. Catthoor, and De Man. "Formalized methodology for data reuse exploration for low-power hierarchy memory mappings*" IEEE Trans. on VLSI systems*, vol. 6, No 4, pp. 529-537, Dec. 1998.

[4] K. Masselos, F. Catthoor, H. De Man, and C.E. Goutis, "Strategy for Power Efficient Design of Parallel Systems" in *IEEE Trans. on VLSI systems*, vol. 7, No 2, pp. 258-265, June 1999.

[5] V. Bhaskaran, K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*, 2nd ed., Kluwer Academic Publishers, Boston 1999.

[6] D. Soudris et al, "Data-Reuse and Parallel Embedded Architectures for Low-Power, Real-Time Multimedia Applications", *Proc. of 10 Int. Workshop on Power and Timing Modeling, Optimization and Simulation* (PATMOS'00), September 2000.

[7] ARM software development toolkit, v2.11, Copyright 1996-7, Advanced RISC Machines.

[8] S. Kougia, A. Chatzigeorgiou, S. Nikolaidis, "Analytical Exploration of Power Efficient Data-Reuse Transformations on Multimedia Applications", *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP'01), May 2001.