Low Power Mapping of Video Processing Applications on VLIW Multimedia Processors

K. Masselos[1,2], F. Catthoor[2], C. E. Goutis[1], H. DeMan[2]
[1] VLSI Design Laboratory, Department of Electrical and Computer Engineering, University of Patras, Rio 26500, Greece. name@ee.upatras.gr
[2] IMEC, Kapeldreef 75, B 3001 Leuven, Belgium, name@imec.be

Abstract

In this paper the impact of power optimization code transformations in a software development context on VLIW multimedia processors is presented. The proposed transformations reduce heavily the power consumption by moving the main part of memory accesses from larger (off-chip) memories to much smaller (on-chip) ones. The effect of the power optimization transformations on system's performance (in terms of number of cycles) is also evaluated. The application of the proposed power optimization transformations most of the time leads to performance enhancement as well. The interaction of the proposed transformations for power optimization with the special performance oriented instructions present in VLIW processor architectures is the main focus of this paper. Experimental results demonstrate that the application of power optimization code transformations is orthogonal to the use of instructions, related to the arithmetic (sub-word) parallelisation.

1. Introduction

Multimedia recently became an integral part of information exchange. Three different approaches exist for the realization of multimedia applications: a) Use of dedicated custom hardware processors. This approach usually achieves the smallest area and power for given performance requirements. However it lacks flexibility since dedicated hardware can only realize a specific algorithm/task. b) Use of (parallel) programmable processors. This solution usually requires larger area and power but it offers increased flexibility allowing realization of more than one algorithms/tasks on the same hardware. c) Use of heterogeneous hardware-software systems to exploit the main advantages of the previous approaches.

Rapid advances in the area of embedded processors made them an attractive solution for the realization of real time multimedia applications. Embedded processors can be classified into two main categories: 1) domain specific processor cores (e.g. Philips TriMedia) 2) general purpose cores (e.g. ARM). The main difference between embedded and traditional processors is the fact that only a single application is running on embedded processors when the system is shipped. Longer analysis and compilation times are also allowed in embedded processors. Finally power consumption is an important design constraint in embedded systems due to portability as well as packaging and cooling issues [1].

Multimedia applications are very data-intensive and thus memory and global communication related power consumption forms the main part of the total power budget of such a system. Therefore, traditional compiler approaches focusing only on speed are not sufficient in our context. This is true both for custom hardware [2] and embedded processor architectures [3]. Typical multimedia algorithms consist of several subsystems. Power consumption is an extensive quantity i.e. it is related to the global data flow over conditional

and or parallel paths of the complete system. On the other hand performance is an intensive quantity i.e. depends only on the critical path of the system.

For uniprocessor custom hardware architectures, the ATOMIUM data transfer and storage exploration methodology [4] for data storage and transfer optimization has been proposed. However this methodology cannot be applied to an embedded target in a straightforward manner. This is because extra issues like performance (in terms of number of cycles) and code size must be taken into account in such systems. Furthermore in many embedded systems the architecture of the memory hierarchy is fixed offering less freedom for optimization. Memory power exploration for embedded architectures has been presented in [5, 6, 7]. In these approaches mapping on architectures consisting of an embedded core (ARM) and dedicated memory hierarchy is assumed. However no formal methodology to achieve the optimal power-performance balance in such architectures has been proposed yet. Moreover no research results and methodologies for this exploration in embedded systems with a fixed cache-based general-purpose memory hierarchy have been presented so far.

For the implementation of multimedia applications on programmable processors there is great need for design automation in the form of compilers, even larger than in the customized hardware case since designers in such case are more dependent on compilers than hardware designers. Traditional software compilers aim at finding the best array layout in memory for optimal caching and performance enhancement. However traditional compilers do not reduce either storage requirements or number of accesses to large background memories. This leads to a big loss in power consumption and to performance (number of cycles) overhead as well. It is clear that there is a big need for compilers (for complex multimedia processors) that can produce code of quality better than or comparable to that of manual code from C code, but these approaches cannot be directly ported to the new context discussed above. This is already the case for general purpose applications on RISC processors starting from C codes.

The aim of our research is the development of a methodology for the application of code transformations in order to minimize power consumption, while meeting real time performance constraints, and this targeted on data-dominated applications realized on programmable processors. This methodology will be included in our ACROPOLIS source-to-source multimedia pre-compiler. A first step towards this direction is described in this paper. Specifically the interaction of the code transformations with the special subword parallelisation instructions present in most multimedia oriented processors (including also Intel MMX technology) is explored.

The rest of the paper is organized as follows. In section 2 the target architecture is presented. The model used for the evaluation of power consumption is described in section 3. In section 4 the proposed methodology is presented. The interaction of the proposed code transformations with the custom performance oriented instructions of multimedia processors is explored using a small but realistic algorithmic kernel typical in multimedia applications in section 5. In section 6 the low power mapping of a complex video processing application on a specific VLIW multimedia processor is described. Conclusions are offered in section 7.

2. Target Architecture

The target architecture is based on (parallel) programmable video or multimedia processors (Philips TriMedia, TMS320C6x, etc.) as this is the current state-of-the-art in embedded real time multi-dimensional signal processing applications [8]. The target architecture is described in figure 1. The main points of this architecture are:

a) **Processing unit:** Multiple functional units are present and constitute either a single threaded superscalar or VLIW processor or a multithreaded multiprocessor. These functional units can be programmed. The programming is mainly performed through the program memory (directed by a compiler or assembler). In the VLIW case (Philips TriMedia, TI C60 etc.) almost everything is compile time controlled and the functional units can be programmed separately. This part of the system is not affected significantly by the proposed storage and transfer methodology. However a large register file is usually present foreground and close to the functional units. Because of its large size this register file plays an important role and heavily affects the result of the application of the proposed transformations.

b) **Memory organization:** The memory hierarchy consists of a number of caches. At least one cache level resides on-chip. The transfer and storage organization is left almost always to hardware (hardware controlled caches) however software controlled caches can be handled also by our global methodology. The main memory resides off-chip. The internal data organization is usually determined at link time. The memory hierarchy is general purpose and fully fixed i.e. it cannot be adapted to a specific application as in [5, 6, 7].

c) **Bus organization:** On-chip a centralized bus organization is usually adopted with one or two busses. This architecture is easier to control or program but forms a potential data transfer bottle-neck and is very power inefficient especially for data dominated multimedia applications [9]. The main memory is typically accessed over a single wide bus. The data transfer is controlled by the hardware (Memory Management Unit). Compile time optimizations of the external data transfer are possible.

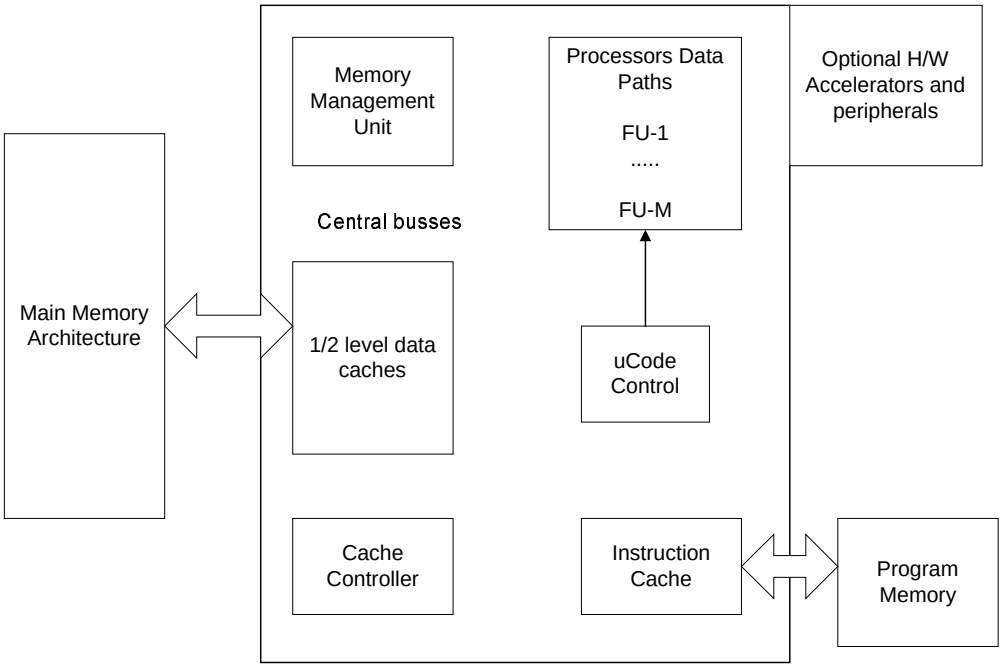

fig.1: General architecture model.

The proposed methodology is mainly related to points b, c of the target architecture and for this reason in the rest of the paper a simpler architecture model will be used. This model includes the points in the architecture that are mainly affected by the proposed methodology and is equally representative of the target architecture. The simplified model is shown in figure 2. Only one level of cache is present however but the proposed methodology can be easily extended also in cases where more cache levels are present.

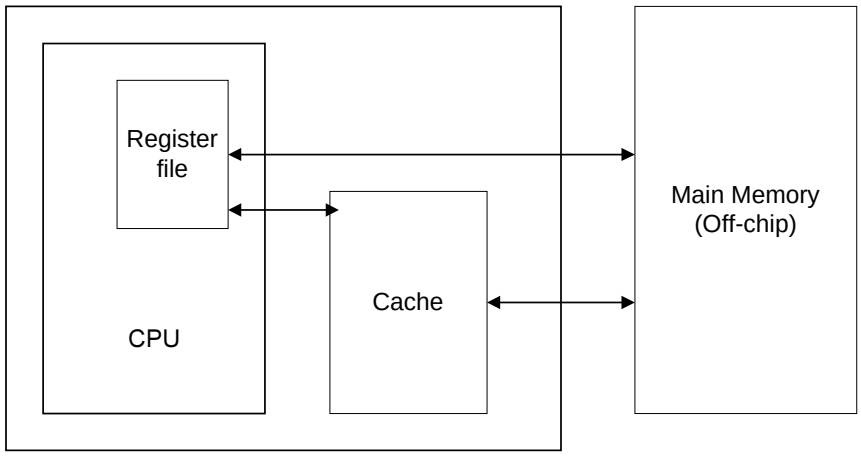

fig. 2: Simplified architecture model.

3. Power Model

In our work evaluates only the power due to memory accesses is evaluated since this power forms the dominant part of the total power budget. Specifically the power due to accesses to the background memories (caches, main off-chip memory) is only evaluated since the power due to the accesses to the register file is much smaller.

For the on-chip memories (caches) the power model used depends upon access frequency, memory size, number of read/write ports and the number of bits that can be accessed in every access. The model is valid for any type of memory (SRAM, DRAM etc.). The model is linear with respect to the access frequency while the dependence on the memory size is determined by a polynomial function. This function is completely dependent on technology. Thus the power consumed by an on-chip memory is given by the following equation:

$$P = (\#\, accesses / \sec ond) \quad \times \quad f(size) \qquad (1)$$

For the off-chip memories the power consumption is determined by the access frequency only [10]. In our work a low power 1 Mb SRAM [11] is assumed. The power consumed in an off-chip memory is given by the following equation:

$$P = (\#\, accesses / \sec ond) \times Energy / transfer \qquad (2)$$

Assuming a memory hierarchy with only one level of cache (on-chip) the total power consumption of the memory hierarchy is given by the following equation:

$$P_{total} = P_{cache} + P_{main} = (\#\, cache\ accesses / \sec ond) \times f(size) + (\#\, main\ memory\ accesses / \sec ond) \times Energy / off\ \ chip\ memory\ access \quad (3)$$

In this paper, the number of accesses for the cache and the main memory are approximated as:

$$\#\, cache\ accesses = \#\, cache\ hits + \#\, cache\ misses + \#\ writebacks \quad (4)$$
$$\#\, main\ memory\ accesses = \#\, cache\ misses + \#\ writebacks$$

In this way the accesses due to the traffic between the main memory and the cache as well as the accesses of the main memory through the cache by-pass (if present) are not taken into consideration. However the above-described model is accurate enough to allow relative comparisons. A more detailed model should be used to make decisions for the final set of most promising alternatives.

4. Summary of Proposed Methodology

The aim of our methodology is the minimization of power consumption of multimedia applications realized on programmable multimedia processors while meeting performance constraints. For this reason a number of code transformations are applied to the original code of the target application. These are not the real focus of this paper but they are listed here as background information to follow the rest of the paper. In this way the number of memory accesses to large background memories (main memory, cache) is reduced. The main categories of code transformations applied are the following:

1) Global data and loop/control flow transformations: Data flow transformations [16] reduce the number of accesses from the CPU to the memory hierarchy. Loop transformations [12] increase the locality of accesses and reduce the number of transfers between the main memory and the cache (or the various levels of the memory hierarchy in the general case). The most frequently used loop transformations are the loop merging, loop interchange, strip mining and loop splitting.

2) Data reuse transformations: These transformations arrange the array signals of the original description exhibiting data reuse in such a way as to better match the target memory hierarchy. This is achieved by introducing temporary signals which satisfy the size limits of smaller memories in the memory hierarchy. A methodology for the application of data reuse transformations in custom hardware architectures is described in [13] but it requires extensions in our context.

3) In-place mapping: This transformation consists of two steps: a) Intra signal in-place. This step optimizes the storage order inside a single signal. b) Inter signal in-place. This step optimizes the storage order between different signals. More details on in-place transformations can be found in [14, 17].

5. Illustration of Interaction with Special Multimedia Processor Instructions on Small DCT Test-Vehicle.

5.1 Mapping without exploiting special instructions

A formalized methodology for the application of the above transformations in our context, is under development. In this paper, the effect of the application of power optimization code transformations on both power and performance is demonstrated using two small but relative algorithmic kernels, namely the one and the two dimensional DCT, which are present in many multimedia applications. Both the one and the two dimensional DCT were selected to be 8 point (a typical size in image and video processing applications). Since both kernels are relatively small there are not many opportunities for optimization. The main transformation applied to both algorithms was the loop merging. The optimized algorithms have been mapped on Philips TriMedia multimedia processor. For the simulation a gray scale image of 256*256 pixels was used. The 8 point one dimensional DCT was applied to the rows of the image while the 8 point two dimensional DCT was implemented using the row-column decomposition. The results of this mapping can be found in tables 1, 2.

Description	Power (Rel.)	Delay (Rel.)
Original	1	1
Transformed	0.95	0.88
Original using custom instr.	0.90	0.73
Transformed using custom instr.	0.79	0.56

Table 1: Results of mapping the 1-D DCT on TriMedia.

Description	Power (Rel.)	Delay (Rel.)
Original	1	1
Transformed	0.95	0.89
Original using custom instr.	0.93	0.84
Transformed using custom instr.	0.74	0.59

Table 2: Results of mapping the 2-D DCT on TriMedia.

The effect of the application of the power optimization methodology is relatively small since only one transformation is applied. An important observation is that the performance is also improved after the application of the power oriented transformation.

5.2 Interaction with the customized multimedia processor instructions

The processors in our target domain include special instructions that exploit special architectural features to enhance performance. Specifically these customized instructions exploit two basic features of the target architecture as described in section 2: a) The presence of multiple functional units b) The wide busses connecting the memories with the CPU. The main idea is the simultaneous access and transfer of more than one words from the memories to the functional units of the CPU. These words can then be processed in parallel by different functional units.

The TriMedia processor which is used as target for our exploration, includes these sub-word parallelism instructions as well. Using these instructions up to four 8-bit words (usually present in image and video applications - pixels) can be accessed and transferred simultaneously to the CPU through the 32 bit busses connecting the memories with the CPU. The customized instructions were applied to both the original and the transformed descriptions of the one and two dimensional DCTs mainly to speed up the transposition required. The results of this mapping can again be found in tables 1, 2 respectively. The use of these instructions clearly improves performance. The most important observation though is the fact that the effect of the customized instructions is largely independent from the gains due to the application of the power optimization transformations. This means that our power optimization methodology is to a first approximation orthogonal to the customized performance oriented instructions present in multimedia processors. Actually, the results in tables 1, 2 even imply that the effect of the customized instructions is enhanced after the application of the power optimization transformations (especially for the 2D DCT case).

6. Mapping of a Real Life Video Processing Application on a Multimedia Processor

In this section the mapping of a complex image/video compression application, namely a Quad tree Structured DPCM algorithm, on TriMedia is described.

6.1 QSDPCM algorithm

The Quadtree Structured Difference Pulse Code Modulation (QSDPCM) [15] technique is an interframe adaptive coding technique for video signals. The algorithm consists of two basic parts: The first one is a three-level hierarchical motion estimation and the second is a quadtree mean decomposition of the motion compensated frame-to-frame difference signal.

6.2 Application of power optimization transformations

The QSDPCM algorithm in its initial form operates on a frame basis i.e. each sub-module pass information for a complete frame to the relevant sub-modules. Most of the signals produced by the sub-modules must be stored in the background memories because of their large size. This leads to increased power consumption because of the large number of accesses to the background memories.

To remove this power bottleneck a global, complex loop merging based transformation has been applied in the original description. Since in QSDPCM algorithm the blocks of each frame are processed independently i.e. there are no dependencies between two blocks of the same frame. Thus this loop merging transformation is a valid transformation. Two outer loops are introduced iterating over the block indices. The procedures are also modified to operate on a block basis. In this way the algorithm operates on a block basis and the signals exchanged between the different sub-modules are small enough allowing storage even in the large register file. This global transformation has the largest impact on the memory related power consumption.

In a second step data reuse transformations were applied in the motion estimation routines of the algorithm. New temporary signals were added to store the candidate matching blocks and reduce the number of accesses to the previous frame array signal.

The effect of the application of the above-described transformations on power and performance are shown in table 3. Power consumption is significantly reduced although many

possibilities for local power optimization still exist. An important point is again that performance is also improved after the application of the power optimization transformations.

6.3 Interaction with the customized multi-media processor instructions

Customized instructions to exploit sub-word parallelism were applied to both the original and the transformed description of the QSDPCM code. Specifically the custom instruction of TriMedia for motion estimation was applied in the motion estimation routines of the QSDPCM algorithm. This instruction accesses and transfers 4 pixels of the candidate matching block simultaneously and calculates their contribution to the matching criterion. The results of the use of the customized instructions in QSDPCM are shown in table 3. The impact of these instructions is smaller this time since there are not so many opportunities for their application. The fact that the customized operations are orthogonal to the proposed power optimization transformations is clear once more.

Description	Power (Rel.)	Delay (Rel.)
Original	1	1
Transformed	0.52	0.89
Original using custom instr.	0.99	0.99
Transformed using custom instr.	0.51	0.53

Table 3: Results of mapping QSDPCM on TriMedia.

7. Conclusions

In this paper, of a detailed power-performance exploration study of video processing applications on a multimedia VLIW processor was presented. The impact of the proposed code transformations aiming at memory power reduction was explored. Power consumption is significantly reduced after the application of the proposed transformations. A formalized methodology for the application of the power optimization transformations is under development. The effect of these transformations on performance is positive as well. The main contribution of this paper, is the demonstration that the application of our power optimizing code transformations is largely orthogonal to the use of customized arithmetic performance oriented instructions exploiting sub-word parallelisation (as present in multimedia processors). Furthermore, we have shown that our code transformations have a larger impact in realistic data-dominated multi-media applications, than the customized instructions.

References

[1] J. M. Rabaey, M. Pedram, “Low Power Design Methodologies”, Kluwer Academic Publishers 1995.

[2] T. H. Meng, B. Gordon, E. Tsern, A. Hung, "Portable video-on-demand in wireless communication", special issue on "Low Power Design" of the proceedings of the IEEE, Vol. 83, No. 4, pp. 659-680, April 1995.

[3] V. Tiwari, S. Malik, A. Wolfe, "Power analysis of embedded software: a first step towards software power minimization", IEEE Trans. on VLSI Systems, Vol. 2, No. 4, pp. 437-445, Dec. 1994.

[4] L. Nachtergaele, F. Catthoor, F. Balasa, F. Franssen, E. De Greef, H. Samsom and H. DeMan, "Optimization of memory organization and hierarchy for decreased size and power in video and image processing systems" in records of the 1995 IEEE intl. workshop on Memory Technology, design and Testing, pp. 82-87, San Jose, California, August 1995.

[5] L. Nachtergaele, D. Moolenaar, B. Vanhoof, F. Catthoor,"System-level power optimization of video codecs on embedded cores: a systematic approach", Journal of VLSI Signal Processing, Kluwer, Boston 1998.
[6] D. Moolenaar, L. Nachtergaele, F. Catthoor, H. DeMan, "System-level power exploration for MPEG-2 decoder on embedded cores : a systematic approach",in. proc. of SIPS'97.
[7] N. D. Zervas, K. Masselos, C. E. Goutis, "Code transformations for embedded multimedia applications : impact on power and performance", in proc. of Power Driven Microarchitecture workshop of ISCA'98.
[8] T. Halfhill and J. Montgomery, "Chip fashion: multi-media chips", Byte Magazine, pp.171-178, Nov. 1995.
[9] F. Catthoor, "Energy-delay efficient data storage and transfer architectures: circuit technology versus design methodology solutions", in. proc. of DATE 98, pp. 709-714.
[10] F. Catthoor, S. Wuytack, E. DeGreef, F. Franssen, L. Nachtergaele, H. DeMan, "System-level transformations for low power data transfer and storage", paper collection on "Low power CMOS design", (eds. A. Chandrakasan, R. Brodersen), IEEE Press, pp.609-618, 1998.
[11] T. Seki, E. Itoh, C. Furukawa, I. Maeno, T. Ozawa, H. Sano, N. Suzuki, "A 6-ns 1-Mb CMOS SRAM with Latched Sense Amplifier", IEEE Journal of Solid State Circuits, Vol.28, No.4, pp.478-483, Apr. 1993.
[12] D. Kulkarni, M. Stumm, “Loop and Data Transformations: A tutorial”, Technical Report CSRI-337, Computer Systems Research Institute, University of Toronto, June 1993.
[13] J. P. Diguet, S. Wuytack, F. Catthoor, H. DeMan, “Hierarchy Explorartion in High Level Memory Management”, in proc. of the 1997 International Symposium on Low Power Electronics and Design, Monterey CA, August 18-20.
[14] E. De Greef, "Storage size reduction for multimedia applications", Doctoral Dissertation, Dept. of EE, K.U.Leuven, January 1998.
[15] P. Strobach, “A New Technique in Scene Adaptive Coding”, Proc. 4th Eur. Signal Processing Conf., EUSIPCO-88, Grenoble, France, Elsevier Publ., Amsterdam, pp. 1141-1144, Sep. 1988.
[16] F. Catthoor, M. Janssen, L. Nachtergaele, H. DeMan, "System-Level Data-Flow Transformations for Power Reduction in Image and Video Processing", in proc. of ICECS'96, pp. 1025-1028.
[17] E. De Greef, F. Catthoor, H. DeMan, "Memory Size Reduction through Storage Order Optimization for Embedded Parallel Multimedia Applications", Intl. Parallel Processing Symposium (IPPS) in Proc. Workshop on Parallel Processing and Multimedia", April 1997, pp.84-98.