A Methodology for Power Efficient Partitioning of Data-dominated Algorithm Specifications within Performance Constraints

K. Masselos¹,², K. Danckaert², F. Catthoor²,³, C. E. Goutis¹, H. DeMan²,³
¹ VLSI Design Laboratory, Department of Electrical and Computer Engineering, University of Patras, Rio 26500, Greece
² IMEC, Kapeldreef 75, B 3001 Leuven, Belgium
³ Also professor at Kath. Univ. Leuven, Belgium
¹name@ee.upatras.gr, ²name@imec.be

Abstract

A methodology for power efficient partitioning of real-time data-dominated system specifications is presented. The proposed methodology aims at reducing the memory requirements in realizations of such applications by applying extensive code transformations in the initial system specification before partitioning over processors. This reorganization basically aligns the data production and consumption between the different procedures of the initial specification thus reducing the memory size requirements (and the resulting power) of the system’s realizations especially those in the interfaces between different processors. The main novel contribution is that performance issues are explicitly taken into account during power oriented system-level transformations. The proposed methodology can be applied both in a parallel (programmable) processor context and also in heterogeneous hardware-software architectures.

1 Introduction

Portability as well as packaging and cooling issues made power consumption an important design consideration [5]. Realizations of data dominated multimedia applications require a large amount of memory for the storage of the large multi-dimensional array-type data structures present in such applications. Advances in memory technology keep decreasing the memory power cost, however the ever-increasing storage requirements of data dominated applications retain memory related power consumption as the dominant contribution to the total (board-level) system power cost [1]. This is true for embedded systems [3] and especially those based on parallel processor [2] or heterogeneous H/W-S/W architectures where memories are usually present in the interfaces between the different processors.

Most of the existing approaches in the parallel processing focus on parallelization, load balancing and communication. Existing hardware-software codesign approaches target issues such as modeling, specification and simulation, interface design and synthesis and mainly partitioning. Background memory related issues and power consumption are not taken into account in both contexts leading to power sub-optimal results. The outline of a formal strategy for power minimization and global memory optimization in a parallel processor context is described in [2]. However it does not incorporate timing issues.

Data dominated applications such as multimedia usually consist of a number of different complex sub-modules. In a high level description (e.g. C or C++) of the system the different sub-modules correspond to procedures that communicate to each other by exchanging several large array signals. Conventionally, partitioning both in a parallel processor and in a heterogeneous H/W-S/W context is performed based on this initial description taking into consideration load balancing only. This leads to a significant number of large buffers mainly between the processors in the final implementation, which is very inefficient in terms of power. The aim of this paper is to describe a methodology for power efficient partitioning of data dominated multimedia applications. The main idea is the application of extensive memory reorganization (through transformations) of the initial description of the target system before the partitioning. This leads to decreased memory requirements in the final system’s implementation.

The rest of the paper is organized as follows: In section 2 the proposed methodology is explained. The target architecture is described in section 3. In section 4 the application of the proposed methodology on an important demonstrator is described in detail. Finally conclusions are offered in section 5.

2 Proposed Methodology

The strategy proposed in this paper removes the memory bottlenecks caused by traditional partitioning approaches by performing extensive memory reorganization of the initial specification of the target application before the partitioning. The initial specification is reorganized by applying a number of memory optimizing code transformations that are included in the ATOMIUM methodology [1] for data storage and transfer optimization. The memory optimized system specification is finally used as input to the partitioning procedure.
In this paper, a detailed order is proposed in which several data transfer and storage optimizing transformations included in [1] are applied to the initial specification. The new methodology extensions are meant in particular for the context of programmable predefined processors when both power reduction and performance issues are crucial. The main categories of transformations included in the proposed methodology are data flow, loop/control flow, data reuse and in-place mapping transformations. The methodology is described in figure 1.

The transformations can be grouped in three categories depending on the way they operate on the array signals of the initial system specification.

a) **Access removing transformations:** Reduce the number of accesses to the different array signals present in the algorithm's description.

b) **Size reduction transformations:** Reduce the size of array signals that are present in the initial specification allowing storage in smaller levels of the memory hierarchy that lie closer to the processing units and thus reducing power consumption.

c) **Access moving transformations:** Introduce extra array signals where accesses from larger array signals existing in the initial specification are moved to. The new signals are smaller than the existing signals and thus they can be stored in smaller levels of the memory hierarchy.

Transformations that enable the application of all the above-described power optimizing transformations may also be applied. Another important issue in our approach is that the array signals in an algorithm description are classified based on their functionality to the following categories:

a) **Array signals with long lifetimes:** These signals are “alive” either throughout or for long periods in algorithm's code. Typical examples are the input and output signals of an algorithm, coefficient arrays and look-up tables.

b) **Intermediate array signals:** These signals are produced as intermediate results of the algorithm and usually have shorter lifetimes in comparison to the signals of the first category.

The first step in the code transformation of the initial specification must be the reduction of the size of the arrays in the initial specification since these arrays are candidates to appear in the boundaries between system's submodules. This is achieved by applying size reduction transformations. This is the step of the initial specification reorganization that has the main impact on the partitioning and is crucial for the quality of its result. After the application of this step, the array signals that are exchanged between procedures in the original specification normally become quite small, leading to significantly reduced buffers in possible realizations. Access moving transformations and the rest of the data storage and transfer optimization steps included in [1] can be applied in a second stage to further improve the power efficiency of the system's realization. They do not as significantly affect the result of the partitioning procedure but are also best done beforehand still. Since the application of the access moving transformations usually introduces new arrays (increasing the memory requirements) a power-area trade-off must be explored in this point. The power optimizing transformations described above may affect performance in a programmable processor context. Especially when applied manually, the effect of the power optimizing transformations on performance must be evaluated before making final decisions on the code transformations to ensure that performance constraints are met. The last step is the final mapping to the physical memory architecture is performed. The completely memory optimized description is passed to a more conventional processor partitioning step.

3 **Target Architecture**

The target architecture of the proposed partitioning methodology is shown in figure 2.

![Figure 2. General target architecture](image-url)
mainly by performance requirements. Local memories are also present in each one of the processors. Both levels of memories are connected to the communication network with a number of busses (determined by performance requirements). Direct connection of the two levels of memories is also possible.

4 Application of the Proposed Methodology

Realization of the QSDPCM video compression algorithm [6] on the general architecture described in section 3 using a number of identical instruction set (software) digital signal processors is assumed. The number of processors has been determined taking into consideration the computational complexity and the real time performance requirements of the application. It is assumed that the long lifetime signals are stored in one memory block while the intermediate signals are stored in another memory block. The memory blocks may lie either off-chip or on-chip depending on their size and may possibly be multiport (the number of ports depends on possible conflicts between signals and speed requirements). In a first step a number of different partitionings of the QSDPCM application have been performed based on the original QSDPCM description. More details on the different partitioning solutions can be found in [4]. In the next step the manually applicable methodology described in section 2 have been applied for the partitioning of the QSDPCM application. The power figures for the different partitioning solutions after the application of the proposed methodology are presented in figure 3. For the evaluation of the data transfer and storage related power consumption the power models of [1] have been used.

The best power results are obtained for the hybrid partitioning 2 although the results for the other hybrid and the task level solutions are very close. Data level partitioning is the worst solution. The power consumption is greatly improved in comparison to the cases where partitioning is performed in the initial specification (by 5 times for hybrid and task level and by 15 times for data level). It can be said that the impact of the size reduction step on power consumption is larger than that of the access moving step. An important observation is that the best solution for power (hybrid 2) when partitioning is performed after the memory reorganization is not the same as the best solution for power when partitioning is performed based on the initial specification (hybrid 1).

During all the transformation steps applied above, the impact on performance has been continuously incorporated and those transformations that would decrease performance have been discarded from the search space. A performance-oriented transformation step including a number of techniques widely proposed in the compilation literature such as loop invariant code motion and common sub-expression elimination to reduce the complexity of address and control expressions (increased after the application of the power optimizing transformations) has been applied as well. Since these transformations do not affect the memory power consumption they can be applied as a last step in the sequence of transformations to remove performance penalties possibly introduced by the application of the power oriented transformations. To evaluate the effect of the proposed power optimizing methodology in the case of the QSDPCM, the fully transformed QSDPCM code has been mapped on ARM. The power oriented transformations most of the times improve performance. After applying the power oriented transformations performance is improved by 17%. The application of the performance oriented transformation further improves performance (by 16%) to achieve a total improvement of 33% without sacrificing any of the power savings.

5 Conclusions

A methodology for power efficient partitioning of data dominated applications has been proposed. The main idea is the application of a preprocessing step that reorganizes the initial specification in terms of memory before partitioning. In this way no large buffers appear in the final system's realization especially in the boundaries between different processors. By reducing the memory requirements the power consumption due to data storage and transfers, which forms the major part of the total power of data dominated applications, is significantly reduced. The methodology allows reducing this power, while not violating the real-time constraint. The proposed methodology can be applied both in a parallel processor and in a heterogeneous hardware-software context.

References


