# Integrated Optimization Methodology for Data Interleaving and Memory Mapping Exploration on VLIW architectures

IASON FILIPPOPOULOS, Norwegian University of Science and Technology
NAMITA SHARMA, Indian Institute of Technology Delhi
FRANCKY CATTHOOR, IMEC
PER GUNNAR KJELDSBERG, Norwegian University of Science and Technology
PREETI PANDA, Indian Institute of Technology Delhi

This work presents a methodology for efficient exploration of data interleaving and data-to-memory mapping options for SIMD (Single Instruction Multiple Data) platform architectures. The system architecture includes VLIW (Very Long Instruction Word) function units and a reconfigurable clustered memory. The scope is the reduction of the overall energy consumption by increasing the utilization of the function units and decreasing the number of memory accesses. The presented methodology is tested using a number of benchmark applications with irregularities on their access scheme. Potential gains are calculated based on the energy models both for the processing and the memory part of the system.

Categories and Subject Descriptors: D.2.2 [**Design Tools and Techniques**]: Design, Methodologies; B.7.1 [**Types and Design Styles**]: Memory Technologies, Design; B.8.2 [**Performance Analysis and Design Aids**]

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Data interleaving, single instruction multiple data (SIMD), system scenarios, design space exploration, memory reconfiguration

## 1. INTRODUCTION

The goal of this work is to improve both the performance and the energy consumption for data intensive applications. We focus on single instruction, multiple data (SIMD) architectures and deal with applications that have irregularities on their access scheme. SIMD architectures can potentially increase the performance of an application, providing that the utilization of them is high. However, applications with irregular access patterns do not provide compact sequences of data that are suitable for high utilization. Hence the performance is lower than expected. In order to improve the performance a systematic exploration of the interleaving options for application's data is needed.

The energy consumption can be divided into two parts, namely the processing and the memory subsystem. The energy needed for processing depends mainly on the utilization of the FUs and any potential stalls, if the memory cannot provide data on the needed rate. The interleaving exploration can increase the utilization of the processing subsystem and reduce time penalties for data loading. The energy consumption on the memory subsystem is affected by the number of memory accesses and the energy per access. Again, the memory architecture and the data-to-memory mapping decisions have a great impact on both the number of accesses and the energy per access.

## 2. MOTIVATIONAL EXAMPLE

A large number of papers have demonstrated the importance of the memory organization to the overall system energy consumption. As shown in [Gonzalez and Horowitz 1996] memory contributes around 40% to the overall power consumption in general purpose systems. Especially for embedded systems, the memory subsystem accounts for up to 50% of the overall energy consumption [Cheung et al. 2009] and the cycle-accurate simulator presented in [Simunic et al. 1999] estimates that the energy expenditures in the memory subsystem range from 35% up to 65% for different archi-

**ALGORITHM 1:** Motivational Example Algorithm

**for** $(i = 0; i < N; i + 4)$ **do**
    $result(i) = A(i) + B(i) + C(i) + D(i)$;
**end**

tectures. The breakdown of power consumption for a recently implemented embedded system presented in [Hulzink et al. 2011] shows that the memory subsystem consumes more than 40% of the leakage power on the platform. According to [Ma et al. 2007], conventional allocation and assignment of data done by regular compilers is suboptimal. Performance loss is caused by stalls for fetching data and data conflicts for different tasks, due to the limited size of memory and the competition between tasks.

To illustrate the sub-optimal utilization of SIMD architectures using conventional allocation and assignment of data, the simple example of Alg. 1 is used. In this example, we assume that the desired result is always the sum of 4 elements from arrays *A, B, C and D*. The access pattern shows an irregularity, as a result of the iteration index. For every group of four sequential array elements, there is only one used for the calculation of the result and the other three are skipped. An intuitive interleaving optimization is the interleaving of the arrays *A, B, C and D*, in order to generate sequences of elements that are all useful on the calculation of the *result* variable. A full interleaving exploration could reveal several options to produce larger sequences of array elements that are needed during the execution of Alg. 1. For example, the interleaving of every fourth line within the combined array ($A|B|C|D$) result in a sequence of 8 accessed elements.

The data-to-memory mapping is presented in Fig. 2. The memory architecture consists of four memory banks and the overall memory size is enough to fit the four arrays. The conventional approach maps each array in a separate bank. If the four arrays are interleaved, the mapping distributes the data through the banks. The first four elements are respectively from arrays A, B, C and D and so forth. As a result only one forth of the array A can be found on the first bank in contrast to the non-interleaving case, in which the whole array A is mapped on the first bank.

A quick estimation for the difference in the energy consumption between the two approaches can be calculated using a simple energy model for the system. The system architecture has an ADD FU that performs operations over 4 words at a time and the memory to processor path has a width of 4 words. Each array element is assumed to have the size of one word. The register file at the processor can only store 5 words and the iteration variable $i$. The first approach

## 3. RELATED WORK

## 4. SYSTEM DESIGN EXPLORATION WORKFLOW

The general workflow of this work is presented in Fig.4.

### 4.1. Formal Model Representation of Access Pattern

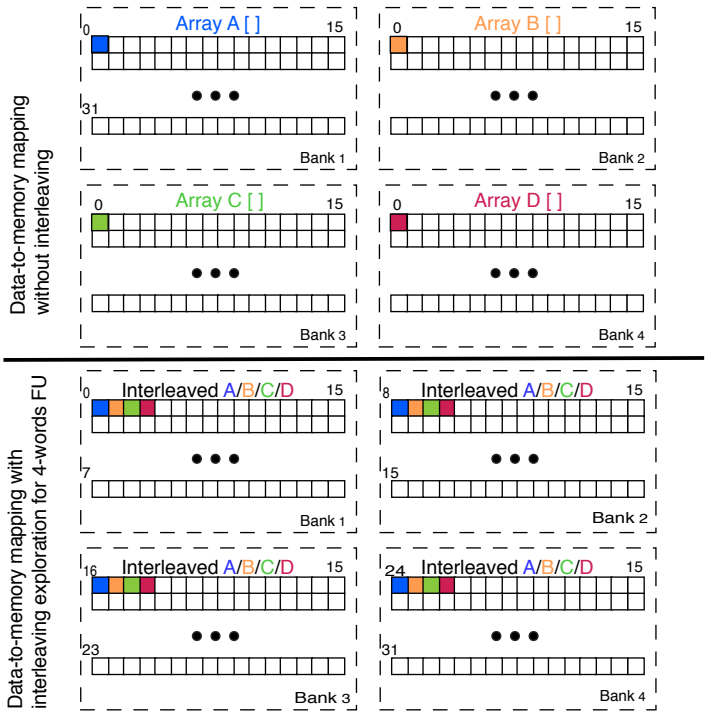
A representation model for the access pattern is employed in order to formally present each step of the methodology.

*Discussion about polyhedral and enumerative approaches*
*Analysis of A-H model based on Angeliki*
*Definition of algebra functions between access patterns*

### 4.2. Data Interleaving Exploration

*Algorithm for exploring data interleaving*

Fig. 1. Data-to-memory mapping for motivational example




### 4.3. Data-to-Memory Mapping Exploration

Given the input from the previous step, we explore the mapping of data-to-memory.

### 4.4. One way constraint propagation

The decisions taken on the interleaving step affect the mapping options. If the interleaving decisions lead to small compact sets of data, the mapping can be done on small energy efficient memory banks. On the other hand, if the mapping exploration is performed first, the freedom for interleaving is reduced. We split the decisions in two steps and the interleaving decisions are propagated as constraints on the mapping exploration phase.

## 5. TARGET ARCHITECTURE

*Analysis of architecture based on our previous papers*

General architecture is presented in Fig.5.

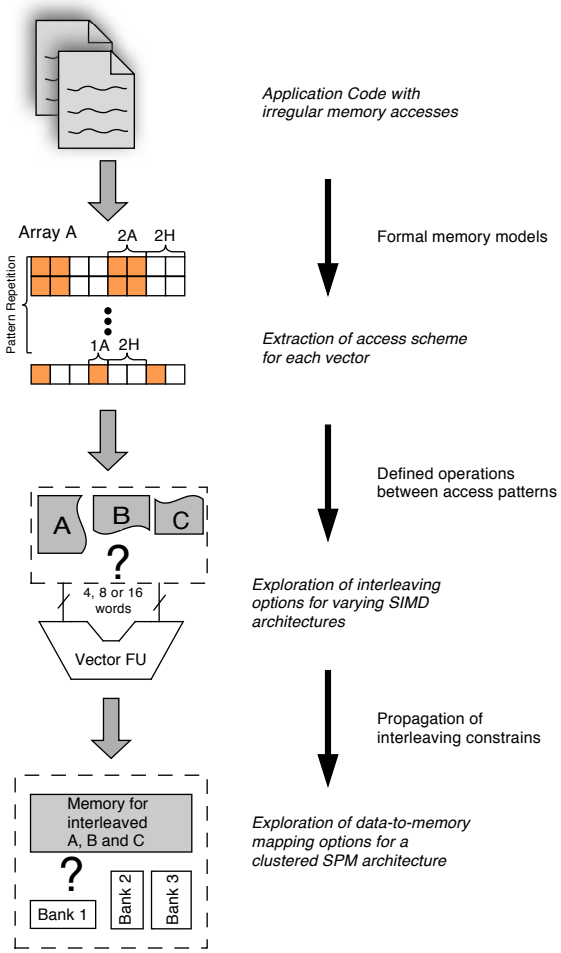
## 6. APPLICATIONS

An example of the channel estimation kernel is presented in Fig. .

Suitable applications:

— Channel estimation kernel
— SOR benchmark
— Motion estimation kernel
— PGM-armor
— Maybe image processing...

Fig. 2. Methodology steps




## 7. RESULTS

The design exploration is applied to the chosen application benchmark and energy numbers are derived based on the described target platform. Results are presented for the four cases:

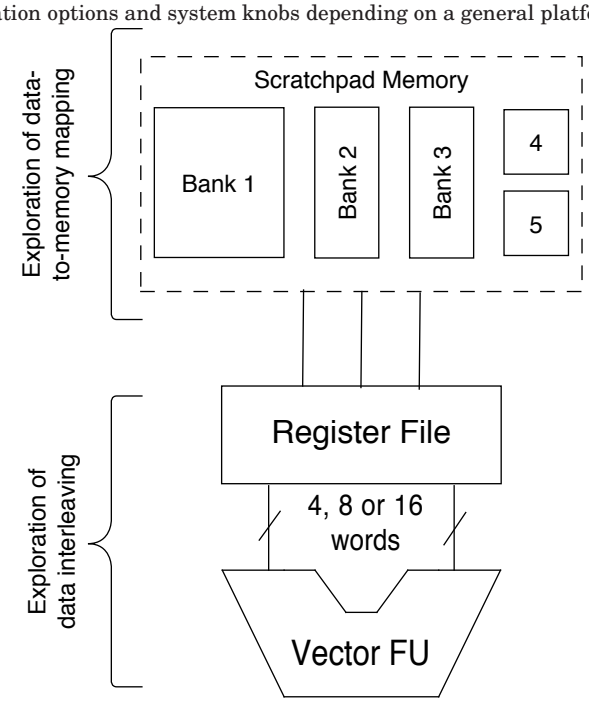
— No optimization
— Interleaving exploration with a static memory platform (gain A)
— Data mapping exploration on a reconfigurable platform without optimized interleaving (gain B)
— Co-exploration of interleaving and mapping options (gain C, C>A+B )

## REFERENCES

E. Cheung, H. Hsieh, and F. Balarin. 2009. Memory subsystem simulation in software TLM/T models. In *Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific*. 811 –816. DOI:http://dx.doi.org/10.1109/ASPDAC.2009.4796580

R. Gonzalez and M. Horowitz. 1996. Energy dissipation in general purpose microprocessors. *Solid-State Circuits, IEEE Journal of* 31, 9 (sep 1996), 1277 –1284. DOI:http://dx.doi.org/10.1109/4.535411

Fig. 3. Exploration options and system knobs depending on a general platform architecture




J. Hulzink, M. Konijnenburg, M. Ashouei, A. Breeschoten, T. Berset, J. Huisken, J. Stuyt, H. de Groot, F. Barat, J. David, and others. 2011. An Ultra Low Energy Biomedical Signal Processing System Operating at Near-Threshold. *Biomedical Circuits and Systems, IEEE Transactions on* 5, 6 (2011), 546–554.

Zhe Ma and others. 2007. *Systematic Methodology for Real-Time Cost-Effective Mapping of Dynamic Concurrent Task-Based Systems on Heterogenous Platforms* (1st ed.). Springer Publishing Company, Incorporated.

T. Simunic, L. Benini, and G. De Micheli. 1999. Cycle-accurate simulation of energy consumption in embedded systems. In *Design Automation Conference, 1999. Proceedings. 36th*. 867 –872. DOI:http://dx.doi.org/10.1109/DAC.1999.782199

Table I. Normalized energy consumption