PhD in Information Technology and Electrical Engineering

Università degli Studi di Napoli Federico II

PhD Student: Innocenzo Mungiello

XXXI Cycle

Training and Research Activities Report – Third Year

Tutor: Alessandro Cilardo

1. Information

a. Innocenzo Mungiello, master's degree in Computer Engineering, magna cum laude, in 2015 from the University of Naples Federico II.
b. XXXI Cycle – ITEE
c. No Grant
d. Tutor: Alessandro Cilardo
e. From 04/09/2017 works in the R&D Department of Rete Ferroviaria Italiana

2. Study and Training activities

a. Ad Hoc Modules:
    i. *Guidelines for software development in safety-critical domains: the examples of railway signalling and avionics domains*, Alessandro Fantechi, 11/01/2018 – 12/01/2018, 2 CFU;

Student: Innocenzo Mungiello									Tutor: Alessandro Cilardo								Cycle XXXI									
innocenzo.mungiello@unina.it									acilardo@unina.it																	
	Credits year 1								Credits year 2								Credits year 3									
		1	2	3	4	5	6			1	2	3	4	5	6			1	2	3	4	5	6			
	Estimated	bimonth	bimonth	bimonth	bimonth	bimonth	bimonth	Summary	Estimated	bimonth	bimonth	bimonth	bimonth	bimonth	bimonth	Summary	Estimated	bimonth	bimonth	bimonth	bimonth	bimonth	bimonth	Summary	Total	Check
Modules	20	0	6	3	5	0	9	23	10	9	3	0	0	0	6	18		0	2	0	0	0	0	2	43	30-70
Seminars	5	1,8	0,4	0,7	1	0	1,3	5,2	5	2,9	1,3	1,4	0	0	0	5,6		0	0	0	0	0	0	0	11	10-30
Research	35	7	4	6	4	7	4	32	45	1	5	7	10	10	3	36	60	10	8	10	10	10	10	58	126	80-140
	60	8,8	10,4	9,7	10,0	7,0	14,3	60,2	60	13	9,3	8,4	10	10	9	60	60	10	10	10	10	10	10	60	180	180

3. Research Activities

a. Title: *Improving Multibank Memory Access Parallelism on SIMT Architectures.*
b. Nowadays, the computer industry, in order to cope with the various changes caused by technological and architectural advances, must take into account several key compromises. For decades, microprocessor architect designers have focused on increasing the density of transistors within the single chip in order to increase their computational performance. The turning point was in 2005 when the limits of the law proposed by Robert H. Dennard came to light as shown in Figure 1. Dennard's law is very related to that of Moore which is the number of transistors inside a chip, doubled almost every 18 months. Dennard claims that even the voltage necessary to power the chip could be properly scaled, in such a way as to make the power dissipated by the chip constant. Therefore, if every 18 months the number of transistors inside the chip doubled, it also doubled its characteristics of energy efficiency. However, Dennard did not take several factors into account:
    i. You cannot set the voltage under a minimum threshold below which the chip does not work properly. The blocking of this scaling meant that the

power dissipated by the chip was no longer constant and this led to the generation of a new problem called *"dark silicon"*, an under-utilization of the transistors present inside the chip. Since the maximum power dissipated by a chip with constant dimension is fixed, if the active transistors increase within it, this threshold will be overcome sooner or later, leading to chip breakage. The manufacturers then were forced to keep "off" most of the transistors (sometimes even 60%) to avoid this problem;

ii. The phenomenon of *leakage currents*, which is the current that were generated when the electrons, by tunnel effect, were able to overcome the insulating layer of CMOS transistors that, between the various production processes, became increasingly thinner. This phenomenon not only led to the increase of chip energy consumption, but also to the increase of its temperature and therefore, additional energy to dissipate excessive heat must be spent.

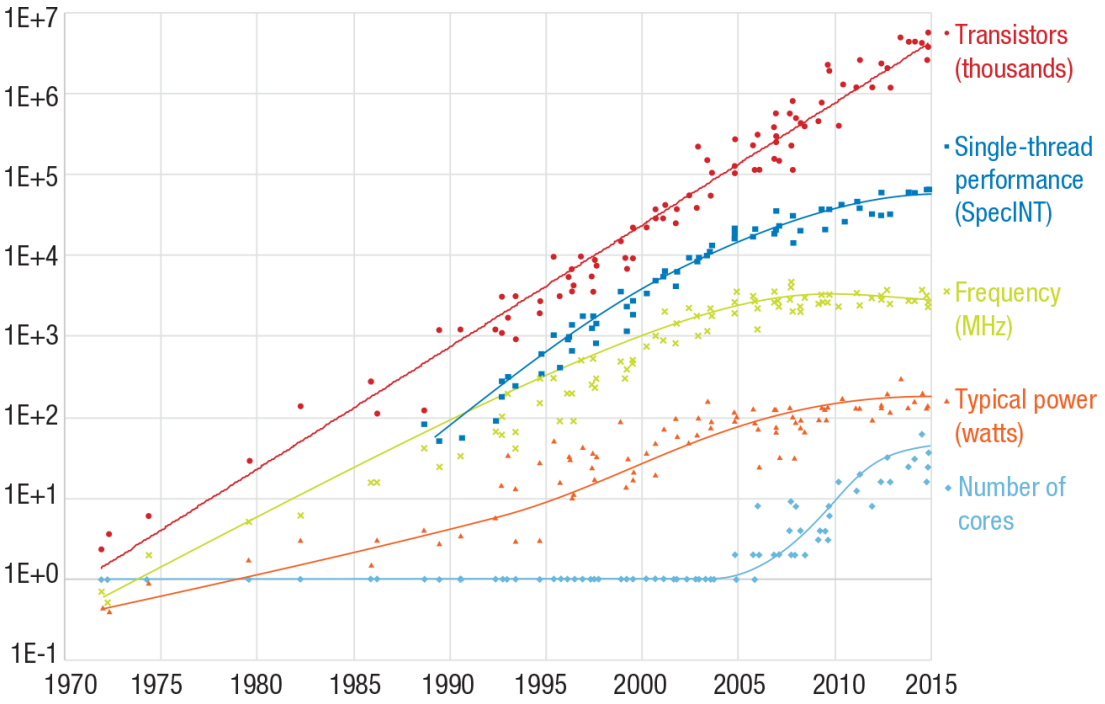

Figure 1 Dennard Scaling

Thus, while Moore's law continues to apply today, Dennard's law came to a halt in 2005 when it faced an increase in performance per watt by a factor of only 1.2 rather than 2.8 expected. Hence the need to have to create new architectures, focus on parallelism and start thinking about energy efficiency as the *true* performance metric. This explains the shift to the multiple-core and the subsequent many-core ideas. The new architectures like FPGA, DSP and GPUs, introduced more challenging issues, then as now: interconnections, shared memory, cooperation, load balancing, dependency, synchronization and last but not least, ways for

programmers to write applications that exploit the increasing number of processors without loss in needed time or quality. Therefore, in contrast with the past, the reduction of the power consumption is currently a fundamental challenge and it is becoming critical across all segments of computing, from the end-users who want ever longer battery life and lower weight and size for their laptops, tablets and smartphones, to the data centers, whose power demands and cooling costs continue to rise. In the just described context, the advance towards an always existed wall has passed almost noiselessly. The advent of heterogeneous and many-core computing exacerbates the gap between the processor and memory performance, the so called *memory wall* shown in Figure 2. Therefore, finding solutions to the memory wall is a crucial step to achieve the HPC target of human brain computing, otherwise known as exascale computing. The memory performance does not affect only the overall performance of the systems, but also impact on its energy performance. In particular, looking at today's GPUs, the power contribution of data movement compared to processing can be as high as 85%.
This scenario presents new challenges for the memory infrastructure from the memory controller to the on-chip and off-chip design, the interconnection, caching, coherency etc. It is meaningful to underline that talking about memory performance can be misleading if not explicitly related to one of the two dimensions along which it extends: *bandwidth* and *latency*. The two concepts are not always directly related and a correct performance evaluation must be described along them to well understand *pros* and *cons* of new technologies. For instance, the newest technological innovation of 3D-stacked DRAM, benefits bandwidth-hungry HPC applications that show an high level of memory parallelism, but it is not expected to break the memory wall as claimed.

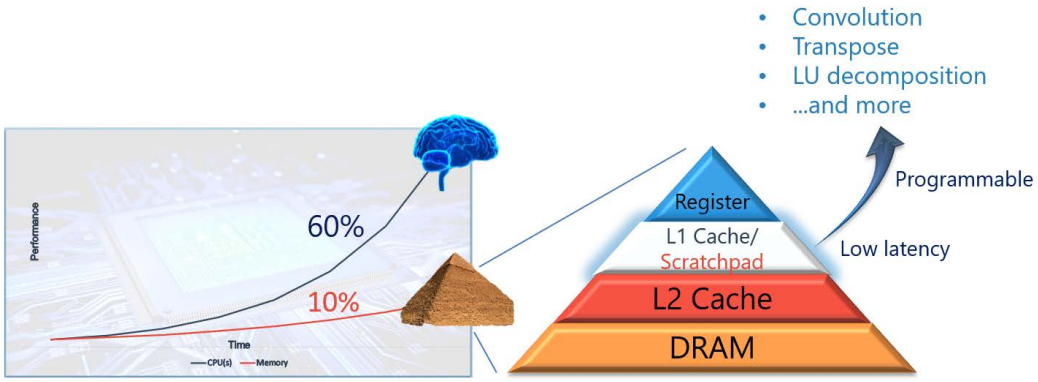

Figure 2 The Unbreakable Memory Wall

In this context, the on-chip memory has increased in importance and complexity along with the advances in processor performance, to offload the larger but slower memories and to allow processing units to fastly communicate. This means that an efficient use of this precious resource would lead to lowered elapsed time for more complex algorithms employing them. These include scratch-pad memory in GPUs (e.g. *shared memory* in NVIDIA devices), as well as dedicated on-chip memory banks

in FPGAs, which can be possibly customized based on the application needs. Such facilities are critical both for performance and energy consumption and are normally organized in a multi-banked structure, potentially enabling parallel data accesses to some regions of the address space. For these reasons, this work focuses its experimental phases on the on-chip scratch-pad memory pointing out some bottlenecks. In fact, when a resource is shared by multiple cores some problems could arise: the contention could generate conflicts and, a memory designed to give a high bandwidth serving multiple requests in parallel, could be accessed inefficiently, causing performance decreasing for the application. In addition, in some architecture like GPUs, the shared memory may be a limiting factor for the number of threads that can run concurrently, because of the inability to completely exploit the available resources.

Since the actual literature does not provide efficient solutions to efficiently reorganize conflictual access patterns, this work aims to mathematically describe the mapping problem and the related implications. Moreover, it presents some optimization techniques that, in some cases, do not involve extra memory and can decrease or eliminate multi-banked memory conflicts, in order to overcame the aforementioned problems and make the most of hardware performance.

Methodology

As mentioned before, this work aims to mathematically describe the memory mapping problem in order to determine some source code optimization that increase the system performance also in terms of *performance per watt*. This amounts to identify data layout transformations in order to:

- speed up the loading and storing of the data in the various memories,
- decrease the communication and synchronization time between the various cores, and
- make the assignment of the different tasks to the various architectures of the system more efficient.

Contextualizing the research problem in the field of source code optimizations for GPUs, the first promising analytical model is certainly the *polyhedral* one which allows a very smooth and streamlined transformation of the data layout. The polyhedral model is a mathematical model that provides a powerful mathematical abstraction to describe the possible transformations on grafted cycles, seeing each iteration as a whole point in a well-defined space called *polyhedron*. Thanks to this it is possible to use the linear algebra and linear programming tools to optimize the grafted cycles and to obtain improvements both on the location of the data and on the parallelization of the latter (see Figure 3).

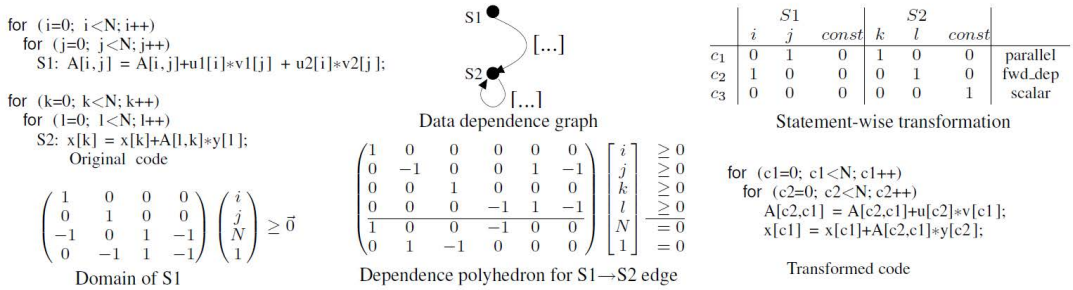

Figure 3 The Polyhedral Approach

Unfortunately, this model can only be applied to a certain category of data that represent a small percentage of those treated by the scientific community. In addition, some of the solutions obtained with this approach resulted in worsening system performance as they wasted shared memory and thence limiting the number of threads that can run concurrently on the GPUs.

So bearing that in mind, I started researching on which were the memory access patterns most used by the HPC applications. The result of this research is a pattern that I called *Transpose Like*. In this pattern, store operations are performed row-wise while load operations are performed column-wise, or vice versa.

Because of the finite number of banks in the local memory, different store/load operations can incur conflicts. The approach used on this pattern aims at gaining a deeper understanding of conflict-avoiding techniques, resulting in a formulation of the problem that allows zero conflicts and zero memory overheads under most circumstances. In particular, the proposed methodology relies on an Integer Linear Programming (ILP) model to describe the problem in terms of linear conditions ensuring optimal bank mapping strategies. I also propose a method for enumerating the solution space exhaustively and evaluating each solution based on the code complexity induced by the scheme.

4. Products

a. Publications:
   i. A. Cilardo and I. Mungiello, *Zero-conflict Memory Mapping for Transpose-like Kernels in SIMT Architectures*, Journal of Parallel and Distributed Computing, 2018, submitted.

b. Conferences:
   i. T. Zoppi, I. Mungiello (RFI), L. Sarti, A. Bondavalli, *Safe Visualization of Critical Information on OTS Devices*, 23rd IEEE Pacific Rim International Symposium on Dependable Computing (PRDC 2018), Fast Abstract, accepted.
   ii. I. Mungiello (RFI) and F. De Rosa (RFI), *RFI R&D Department Projects*, Italian Workshop of Embedded Systems (IWES 2018).

5. Tutorship

- Co-Advisor for a master thesis about railways wireless communications entitled, "*Implementation of a Multipath communication protocol for railway application, based on GSM-R and LTE*". Corso di Laurea in Telecommunications Engineering, Università di Bologna, 16/03/2018 under the supervision of Rete Ferroviaria Italiana (RFI).