2PARMA: Parallel Paradigms and Run-time Management Techniques for Many-Core Architectures

C. Silvano*, W. Fornaciari*, S. Crespi Reghizzi*, G. Agosta*, G. Palermo*, V. Zaccaria*, P. Bellasi*, F. Castro*, S. Corbetta*, A. Di Biagio*, E. Speziale*, M. Tartara*, D. Siorpaes†, H. Hübert‡, B. Stabernack‡, J. Brandenburg‡, M. Palkovic§, P. Raghavan§, C. Ykman-Couvreur§, A. Bartzas¶, S. Xydis¶, D. Soudris¶, T. Kempf‖, G. Ascheid‖, R. Leupers‖ H. Meyr‖, J. Ansari‖, P. Mähönen‖, and B. Vanthournout**

* Dipartimento di Elettronica e Informazione – Politecnico di Milano, Italy,
† STMicroelectronics, Italy, ‡ Fraunhofer HHI, Germany, § IMEC vzw, Belgium and IBBT, Belgium,
¶ Institute of Communication and Computer Systems – National Tech. University of Athens, Greece,
‖ RWTH – Aachen University, Germany, ** CoWare, Belgium

***Abstract*—The 2PARMA project focuses on the development of parallel programming models and run-time resource management techniques to exploit the features of many-core processor architectures.**

The main goals of the 2PARMA project are: the definition of a parallel programming model combining component-based and single-instruction multiple-thread approaches, instruction set virtualisation based on portable byte-code, run-time resource management policies and mechanisms as well as design space exploration methodologies for many-core computing architectures.

I. Introduction

The main trend in computing architectures consists of integrating small processing cores in a single chip where the cores are connected by an on-chip network. Given the technology trend, we would expect, in the coming years, to move from multi to many core architectures. Multi-core architectures are nowadays prevalent in general purpose computing and in high performance computing. In addition to dual- and quad-core general-purpose processors, more scalable multi-core architectures are widely adopted for high-end graphics and media processing, e.g. IBM Cell BE, NVIDIA Fermi, SUN Niagara and Tilera TILE64. To deal with this increasing number of processing cores integrated in a single chip, a global rethinking of software and hardaware desing approaches is necessary.

The 2PARMA project focuses on the design of a class of parallel and scalable computing processors, which we call Many-core Computing Fabric (MCCF) template. This template is composed of many homogeneous processing cores connected by an on-chip network. The class of Many-core Computing Fabric promises to increase performance, scalability and flexibility only if appropriate design and programming techniques will be defined to exploit the high degree of parallelism exposed by the architecture.

Benefits of Many-core Computing Fabric architectures include finer grained possibilities for energy efficiency paradigms, local process variations accounting, and improved silicon yield due to voltage/frequency island isolation possibilities. To exploit these potential benefits, effective run-time power and resource management techniques are needed.

Moreover the Many-core Computing Fabric offers customisation capabilities to extend and to configure at run-time the architectural template to address a variable workload.

The 2PARMA project aims at overcoming the lack of parallel programming models and run-time resource management techniques to exploit the features of many-core processor architectures focusing on the definition of a parallel programming model combining component-based and single-instruction multiple-thread approaches, instruction set virtualisation based on portable bytecode, run-time resource management policies and mechanisms as well as design space exploration methodologies for Many-core Computing Fabrics.

The research objectives of the project are intended to meet some of the main challenges in computing systems:

- To improve performance by providing software programmability techniques to exploit the hardware parallelism;
- To explore power/performance trade-offs and to provide runtime resource management and optimisation;
- To improve system reliability in terms of lifetime and yield of hardware components by providing transparent resource reconfiguration and instruction set virtualisation;
- To increase the productivity of the process of developing parallel software by using semi-automatic parallelism extraction techniques and extending the OpenCL programming paradigm for parallel computing systems.

The rest of this paper is organized as follows. Section II provides an introduction to the target architectural template. Section III describes the 2PARMA design flow and the design methodologies employed, while Section IV introduces the applications targeted in the project. Finally, Section V draws some conclusions and outlines the future work.

II. Many-Core Computing Fabric Architecture Template

The 2PARMA project focuses on the Many-core Computing Fabric (MCCF) template composed of many homogeneous processing cores connected by an on-chip network as shown in Figure 1.

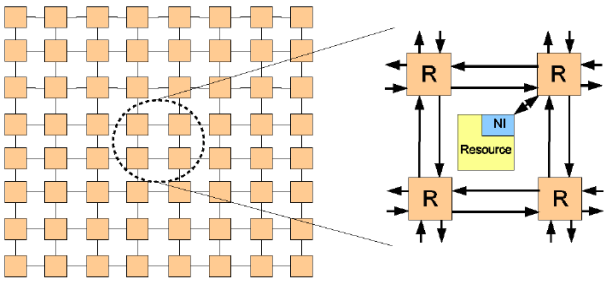

Fig. 1. 2PARMA Many-Core Computing Fabric Template

The project will demonstrate methodologies, techniques and tools by using innovative hardware platforms provided and developed by the partners, including the "Platform 2012" – an early implementation

978-0-7695-4076-4/10 $26.00 © 2010 IEEE
DOI 10.1109/ISVLSI.2010.93

of Many-core Computing Fabric provided by STMicroelectronics – and the many-core COBRA platform provided by IMEC.

A. STMicroelectronics Platform 2012

The P2012 program is a cooperation between STMicroelectronics and Commissariat a l'Energie Atomique (CEA) to design and prototype a regular computing fabric capable to improve manufacturing yield. Platform 2012 (P2012) is a high-performance programmable accelerator whose architecture meets requirements for next generation SoC products at 32nm and beyond. The goal of P2012 is twofold: from one side, it is to provide flexibility through massive programmable and scalable computing power; from the other side, to provide a solid way to deal with increasing manufacturability issues and energy constraints.

To achieve these two goals the P2012 program is planning to use the two following key enablers:

- Emerging 3D stacking techniques to revise the memory hierarchy organisation;
- STMicroelectronics deep know-how on applications to inject the correct specialisation level into the architecture.

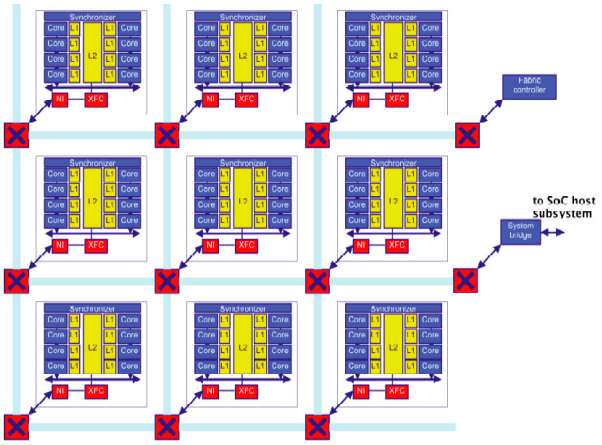

Fig. 2. Platform 2012 Template

Organised around an efficient Network-on-Chip communication infrastructure, P2012 enables connecting a large number of decoupled STxP70 processors SMP clusters, offering flexibility, scalability and high computation density. Figure 2 shows the Platform 2012 computing fabric composed of a variable number of 'tiles' that can be easily replicated to provide scalability. Each tile includes a computing cluster with its memory hierarchy and a communication engine. The computing fabric operation is coordinated by a fabric controller and is connected to the SoC host subsystem through a dedicated bridge, with DMA capabilities. Clusters of the fabric can be isolated to reduce power consumption (or to switch-off a faulty element) and frequency/voltage scaling can be applied in active mode.

The P2012 computing fabric is connected to a host processor such as the ARM Cortex A9, via a system bridge. The fabric is in this way exposed to legacy operating systems like the GNU/Linux OS.

Many P2012 platform design choices are still open to be explored, and the 2PARMA Consortium, which is one of the very early adopters of this technology, effectively contributes to the platform architecture specification and relevant optimisations.

B. IMEC ADRES-based COBRA Platform

The IMEC's COBRA platform is an advanced platform template targeting 4G giga-bit per second wireless communication. One instance of the platform is shown in Figure 3.

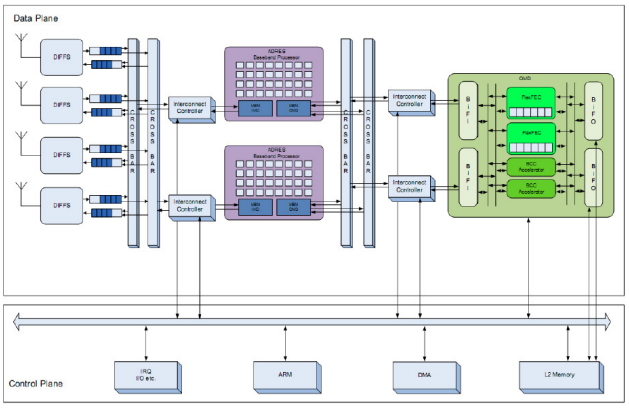

Fig. 3. IMEC COBRA platform

This platform can be customised to handle very high data rates as well as low throughputs in a scalable way. This platform largely consists of 4 types of cores. *DIFFS*, an ASIP processor tuned towards sensing and synchronisation, and optimised for very low power. It is tuned towards average duty cycle. *ADRES* [1], a coarse-grained reconfigurable core template [2] consisting of a number of functional units connected in a given interconnect network. The core has been tuned to be capable of doing inner modem processing of various standards efficiently. *FlexFEC* [3], a flexible forward error correction ASIP that is capable of doing different outer modem processing. It is a SIMD engine template where the instruction set, bit width of the data-path and the number of SIMD slots can be chosen based on the set of requirements of the standard to be run. A *ARM* host processor for controlling the tasks on the platform (e.g. the run-time manager task).

The first three cores (DIFFS, ADRES, FlexFEC) cores can be instantiated for a mix of targeted standards that need to be supported. Also all parts of the platform are programmable in C (ADRES, ARM) or assembly (FlexFEC, DIFFS). The communication is ensured by customised InterConnect Controller (ICC) cores that are programmable at assembly level as well.

In this platform, besides the type and the size of each core, the number of each type of core can be selected based on the different standards that need to be supported on the platform. For example for the highest throughput modes for Wireless LAN 802.11n 4x4 MIMO, the number of DIFFS cores may be 4 and two (multi-threaded) ADRES cores and two FlexFEC cores. In case the platform has to support only a low end Wireless LAN SISO standard or a basic LTE SISO reception, one DIFFS core, ADRES core and one FlexFEC would be sufficient to meet the requirements of this mode.

III. Design Flow and Tools

The main goals of the 2PARMA project related to the analysis and development of the complete software layer able to exploit the features of future many-core processor architectures presented in the previous section. This goal has been tackled from several standpoints as presented in the following subsections.

The tool environment and design flow of the 2PARMA project is shown in Figure 4. The basic idea behind the 2PARMA project is to combine the automatic extraction of parallelism to dynamic compilation to exploit the management of system resources at runtime.

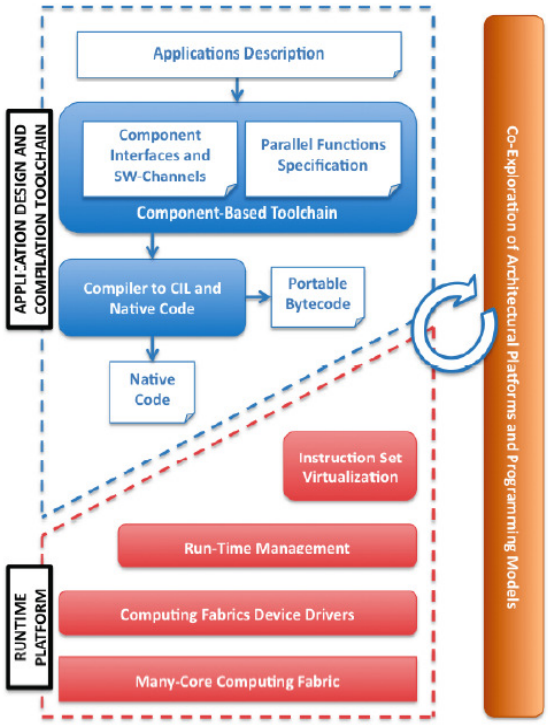

Fig. 4. 2PARMA Design Flow and Tools

Starting from the specifications of the industrial applications and architecture to be used for the integration and validation of the design flow, the main goal is to define a parallel compilation tool-chain and Operating System support. The compilation tool chain starts with the component-based application source code (C-based) to be assembled and compiled to byte-code and further dynamically translated to machine code. Then the machine code execution and deployment will be supported by an OS layer to provide isolated logical devices efficiently communicating (device-to-device and host-to-device). The GNU/Linux operating system will be used as the software reference common ground for what the host processor is concerned.

Another main goal consists in developing methodologies and tools to support the application/architecture co-exploration. More in detail, the project focuses on profiling the parallel applications aimed at finding the bottleneck of the target platform and on the robust design space co-exploration of static and dynamic parameters by considering dynamic workloads, while identifying hints/guidelines for dynamic resource management.

Then, the Run-Time Resource Manager (RTRM) provides adaptive task and data allocation as well as scheduling of the different tasks and the accesses to the data for many-core architectures. Furthermore, the adequate power management techniques as well as the integration to the Linux OS will be provided.

To conclude, the concrete results of the project are related to the analysis and development of the complete software layer able to exploit the features of future many-core processor architectures, and can be summarised as follows:

- An integrated compiler toolchain and OS Layer, supporting Component-Based Software Engineering (CBSE) and expression of data-level parallelism as well as logical isolation provided by OS abstractions.
- A design toolset for supporting the HW/SW co-exploration considering the robustness with respect to the run-time system workload.
- A set of techniques to manage at run-time the system resources.

Based on the set of operating points given by the DSE tool at design time and the info collected at run-time on system workload and resource utilisation, the run-time management techniques will optimise data allocation and data access scheduling, task mapping and scheduling and power consumption.

The rest of this Section provides more detail on the techniques employed in the design flow.

A. Programmability of Many-core Computing Fabrics

2PARMA project tackles the issue of programmability of Multi-core Computing Fabrics at both the programming language and Operating System level. On one hand, it leverages the increasingly popular Component-Based Software Engineering (CBSE) and devel-ops parallelism extraction techniques to identify opportunities for parallelisation at a high level in the design phase; 2PARMA then employs extensions of existing standards for parallel programming, such as OpenCL, to express data parallelism for Many-core Computing Fabrics.

The 2PARMA compiler toolchain will benefit from techniques that automatically handle memories local to processor clusters usually available in GPGPU architectures, as well as automated lowering of higher-level code to OpenCL [4]. This will allow the programmer to first design the application under a shared memory paradigm, and then perform fine-tuning on a view of the application where the shared memory abstraction is removed.

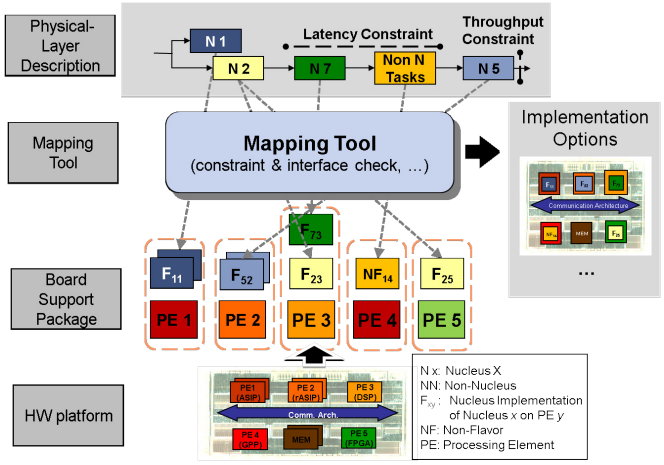

Fig. 5. Nucleus-based CBSE Toolchain

For CBSE methodologies, the applicability of the concepts and tools developed in the cross-disciplinary "Nucleus" flagship project of the UMIC Research Center [5] at RWTH Aachen University will be investigated. The key idea pursued by the Nucleus project [6] is to define critical algorithmic kernels that capture common functionalities among different communication standards. In a later stage these Nuclei are assembled to construct the complete application, as shown

in Figure 5. In contrast to existing CBSE tool-chains, requirements such as latency and throughput are integrated into the application description directly. Furthermore, the Nuclei are mapped to Flavors - efficient and optimized implementations for one Nucleus on a particular Processing Element (PE) - that are kept within the Board Support Package of a given HW platform. This allows the mapping tools to identify possible implementation options by performing interface and constraint checks. Among these different options designers can select the final implementation that achieves the best performance.

B. Dynamic Compilation for Instruction Set Virtualisation

A critical issue in the adoption of a new platform is the ability to provide effective (in terms of performance) support of legacy and third party code. In the context of 2PARMA, dynamic compilation is used to solve this issue, by virtualising the instruction set exposed by the Many-Core Computing Fabric and exposing a bytecode intermediate language to the application developer.

Given a bytecode program, the dynamic compiler must load the input bytecode, decode it to an intermediate language, optimize it for the target platform, translate it to machine code and finally execute it, thus introducing a latency between method invocation and execution. In the 2PARMA project, we adopt ILDJIT [7], a dynamic compiler for ECMA-335 bytecode language that breaks up this sequence in a software pipeline, where each of pipeline step (a compiler thread) performs one step of the compilation process on a different method.

The software pipeline allows even sequential programs compiled in CIL to benefit from multiple hardware cores: while one core executes the current method, the other ones can be used to pre-compile and optimize methods that will have to be used in the future [8]. Moreover, it provides the opportunity of further reducing overheads by pre-compiling bytecode into a lower-level, but still machine independent, intermediate format (*IR*). The compilation of IR to machine code only represents a minimal fraction of the total compilation time, thus allowing a lighter compiler framework to be used in embedded devices [7].

C. Runtime Management

2PARMA project aims at improving energy efficiency w.r.t. conventional power management strategies, by supporting efficient and optimal task, data and devices managements able to dynamically adapting to the changing context, taking into account the Quality-of-Service (QoS) requirements imposed by the user to each application.

In 2PARMA we push the boundary of this trade-off to reduce the design time effort and move more responsibility to the runtime resource manager, at different abstraction levels. We accomplish this task by providing a runtime manager (RTM) with metadata information covering both runtime and design time knowledge of both hardware and software. The runtime management performs adaptive task mapping and scheduling [9], dynamic data management [10] and system-wide power management [11].

The first role of the RTM is to monitor the dynamic applications behaviour to control the available and required platform resources at run time while meeting user requirements. To handle multiple tasks competing at run time for limited resources, a scenario-based mapping technique supporting inter-task scenarios will be developed. Also for many of these applications, obtaining efficient results (performance and power) is impracticable if the quality is not lowered. Hence a run-time monitoring technique will be developed to tune well-chosen parameters based on input data to meet the requirements while maximizing the output quality.

The second role of the runtime manager is to handle the dynamism in the control flow and data usage (dynamic (de)allocation of data), by determining a suitable allocation strategies that meet the application needs. To that end, new fit, coalescing, and split policies, taking application characteristics into account, may be needed [10].

Finally, the runtime manager is responsible for the adaptive power management of the many-core computing fabric architecture. This is achieved by identifying a suitable top-level modelling of the entities composing the overall systems, in terms of exchanged data, exposition of control settings and status information of the components/devices. The manager is responsible to combine the adaptive runtime task and data management schemes (component/device specific optimisations) with the adaptive power management policies, being aware of the presence of local optimisation strategies exposed by the rest of the system components (e.g., device drivers and in general any other resource manager).

The end result is a distributed runtime QoS Constrained Power Manager (CPM) working at the OS-level [11], based on the following concepts. *System-Wide Metrics* (SWMs) which are parameters describing behaviours of a running system and represent QoS requirements. They could be either ”abstract” (ASMs) or platform dependent (PSMs). The firsts are exposed to user-space and can be used by application to assert QoS requirements. The seconds instead are defined in the platform code and are used to keep track of hardware inter-dependencies. *Device Working Regions* (DWRs) defining the mapping between the operating modes of a devices and the SWMs that define the QoS level supported by each operating mode. *Feasible System Configurations* (FSCs) which are the n-dimensional intersections of at least a DWR for each device (where n is the number of SWMs defined). They identify the system-wide working points of the target platform where certain QoS levels are granted. *Constraints* on SWMs defined at run-time according to the QoS requirements of applications or drivers on these parameters. All the QoS requirements on the same SWM are translated on a constraint using an aggregation function which depends on the type of the parameter. *Multi-Objective optimization*, which could consider different performance parameters, by assigning a weight to each SWM, and energy consumptions, by assigning a power consumption measure to each FSC.

In practice, CPM-related activities inside the OS, can be grouped in three main phases:

- FSC Identification: at boot time all the device drivers registers to CPM by exposing their DWRs. All FSCs can be automatically identified by performing the intersection of DWRs.
- FSC ordering: every time the optimization goals change, the FSC are sorted according to the global optimization policy. This happens usually when the device usage scenarios change.
- FSC selection: at run-time applications can assert QoS requirements on a specific SWM. These requirements are aggregated to produce a new constraint for each SWM. These constraints could invalidate some FSC. If the current FSC is also invalidated then a new candidate is selected according to the ordering defined in the ordering phase.

Finally all drivers are notified about the new FSC and required to update accordingly their operating mode.

The CPM model has been preliminary implemented as a Linux kernel framework (version 2.6.30) and tested under some use-cases to evaluate its overhead, which is negligible (always less than 0.01%) [11].

D. Design Space Exploration

Design space exploration plays a crucial role in designing many-core computing platforms [12], [13], [14]. Design alternatives may consist of the tuning of processor micro-architectural components, different mappings of software tasks to resources, different scheduling policies implemented on shared resources as well as lower level design parameters. In this context, the 2PARMA project provides to the designer design space exploration methodologies to trade-off the system-level metrics (such as energy and delay) by considering the dynamic evolution of the system. To this end, the MULTICUBE Explorer framework will be extended to support run time DSE.

Focusing on the combined optimisation of parallel programming models and architectural paramters for many-core platforms, it is expected that conventional or state-of-the-art profiling techniques cannot be used for the task of analysing and profiling. Profiling memory accesses on a cycle accurate basis [15] is not sufficiently supported by available profiling tools due to the fact that only shared memory architectures were modelled at the time. Moreover, many core platforms will be built upon completely different interconnection networks requiring new profiling techniques taking into account connection topologies. The influence on the overall system performance of the implemented connection topology and the resulting fragmentation of memory accesses abroad the distributed memory will be evaluated by a set of tools. Based on the profiling methodologies developed in the project, it will be possible to get an in depth view of how parallel programming models behave on many core platforms. The results will be used to co-optimise the programming model and the architecture of the target platform.

IV. Applications

The Many-Core Computing Fabric template is designed as a coprocessor for computationally intensive applications in high-end embedded scenarios. To prove its effectiveness, and the effectiveness of the design flow and tools produced in the 2PARMA project, it is necessary to employ real world applications of considerable industrial impact. These applications will be engineered, optimised and specialised using the methodologies described in Section III, and tested on the two target implementations of the Many-Core Computing Fabric template. In this Section, we introduce the three applications chosen for the 2PARMA project: *Scalable Video Coding* (SVC), *Cognitive Radio*, and *Multi View Video* (MVV).

A. Scalable Video Coding

SVC [16] also known as layered video coding has already been included in different video coding standards in the past. Scalability has always been a desirable feature of a media bit stream for different services and especially for best-effort networks that are not provisioned to provide suitable QoS and especially suffer from significantly varying throughput. Thus a service needs to dynamically adapt to the varying transmission conditions. E.g., a video encoder shall be capable of adapting the media rate of the video stream to the transmission conditions to provide at least acceptable quality at the clients, but shall also be able to explore the full benefits of available higher system resources. Within a typical multimedia session the video consumes the major part of the total available transmission rate compared to control and audio data. Therefore, an adaptation capability for the video bit rate is of primary interest in a multimedia session. Strong advantages of a video bit rate adaptation method relying on a scalable representation are drastically reduced processing requirements in network elements compared to approaches that require video re-encoding or transcoding. Thus, H.264/AVC-based SVC is of major practical interest and it is therefore highly important to investigate implementation aspects of SVC.

SVC is an ideal application for demonstrating runtime resource management, including power management techniques. An SVC implementation will be provided by Fraunhofer HHI within the context of the 2PARMA project.

B. Cognitive radio

From the domain of wireless communications, a cognitive radio application will be provided by RWTH Aachen University. This application includes both physical and MAC-layer processing. Especially, the low latency as well as high throughput and reconfiguration requirements of state-of-the-art wireless communication standards makes the cognitive radio application as a highly appropriate use case for the 2PARMA project and its parallel programming models.

Following the Nucleus CBSE approach we identify the commonalities among different wireless MAC protocols. These functional commonalities among MAC protocols are identified as the fundamental building blocks so that a particular protocol can be realized by simply combining the required set of functionalities together. These unit blocks for MACs are expressed through well defined interfaces so that these can generically be re-used in different MAC implementations. We have developed a tool (called the *wiring engine*) that combines the different components of the MAC together by coordinating the control and data flow among the blocks. The wiring engine will also be able to exploit the parallelism in a particular MAC realization to achieve execution efficiency.

Our approach of composing MAC protocols based on the same set of functional components using the wiring engine, leads to the realization of a wide range of protocols and allows run-time adaptation [17]. We are also investigating the design of a MAC description language and correspondingly a MAC interpreter. In the future, a host meta-compiler can be used for realizing MAC protocols in a highly efficient manner. By implementing the MAC modules demanding high degree of computations and communication in the silicon as kernel functionalities, our approach allows to meet the strict timing deadlines thereby giving high degree of performance gains and flexibility. Furthermore, our methodology also facilitates much deeper cross-layer designs between MAC and physical layer kernels, which are demanded by cognitive and spectrum agile MACs [18].

C. Multi-View Video

With the current development of electronic, network, and computing technology, Multi-View Video (MVV) becomes a reality and allows countering the limitations of conventional single video. MVV refers to a set of N temporal synchronised video streams coming from cameras that capture the same scene from different viewpoints.

In particular, within the context of the 2PARMA project, we consider a cross-based stereo matching algorithm [19], assuming two aligned left and right cameras. The algorithm compute stereo pixel depth by means of their disparity (difference on the x coordinate), which can then be visualised in grayscale encoding, as shown in Figure 6. The cross-based stereo matching algorithm is an area-based local method, where the disparity computation at a given pixel only depends on intensity values within a finite window. The challenge of this method is that the support window should be large enough to include enough intensity variation for reliable matching, while it should be small enough to avoid disparity variation inside the window. Therefore, to obtain accurate disparity results at reasonable costs, an appropriate support window should be selected adaptively.

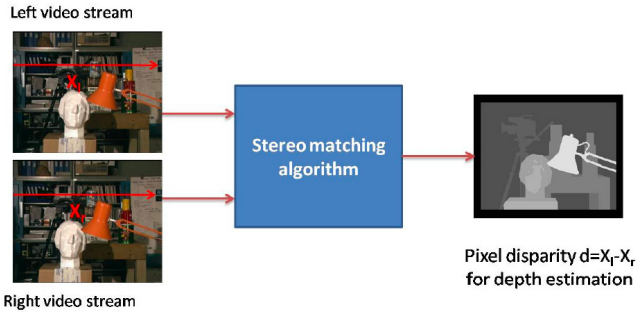

Fig. 6. Stereo matching algorithm

Figure 7 depicts the overall flow of the stereo matching algorithm. The prefiltering step reduces the image noise. The local cross construction step identifies for each pixel the support window likely belonging to the same depth, as follows. It constructs a local cross around each pixel. Local crosses are obtained by aggregating neighbouring consecutive pixels, whose colour difference is less than a given threshold and whose distance is less than a maximum arm length. The cost aggregation step measures a matching cost for each stereo pair of pixels. This cost is used in the disparity selection and refinement step. Pixels contained in the same local support window mostly originate from the same scene patch, and hence they share similar disparities. Disparity values can range within a given interval.

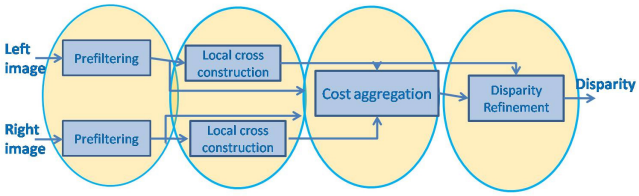

Fig. 7. Overall flow of the stereo matching algorithm

V. Summary and Project Outcomes

The 2PARMA project tackles the issue of programming and managing a Many-Core Computing Fabric – a novel architectural template represented within the project by STM Platform 2012 and IMEC Cobra architectures.

A design flow has been defined, starting with the high-level implementation of the application and leading to runtime management of the application execution, in a highly-variable context where multiple applications compete for resources. Design space exploration and profiling techniques close the feedback loop, helping the designer in refining the application for each target platform.

Finally, a set of high-impact applications has been selected to demonstrate the effectiveness of the proposed methodologies.

References

[1] V. Derudder, B. Bougard, A. Couvreur, A. Dewilde, S. Dupont, L. Folens, L. Hollevoet, F. Naessens, D. Novo, P. Raghavan, T. Schuster, K. Stinkens, J.-W. Weijers, and L. V. der Perre, "A 200mbps+ 2.14nj/b digital baseband multi processor system-on-chip for sdrs," in *VLSI Circuits, 2009 Symposium on*, june 2009, pp. 292 –293.

[2] B. Mei, S. Vernalde, D. Verkest, H. D. Man, and R. Lauwereins, "Adres: An architecture with tightly coupled vliw processor and coarse-grained reconfigurable matrix," in *FPL*, ser. Lecture Notes in Computer Science, P. Y. K. Cheung, G. A. Constantinides, and J. T. de Sousa, Eds., vol. 2778. Springer, 2003, pp. 61–70.

[3] F. Naessens, V. Derudder, H. Cappelle, L. Hollevoet, P. Raghavan, M. Desmet, A. AbdelHamid, I. Vos, L. Folens, S. O'Loughlin, S. Singirikonda, S. Dupont, J.-W. Weijers, A. Dejonghe, and L. V. der Perre, "A 10.37 mm2 675 mw reconfigurable ldpc and turbo encoder and decoder for 802.11n, 802.16e and 3gpp-lte," in *VLSI Circuits, 2010 Symposium on*, june 2010, pp. 292–293.

[4] Andrea Di Biagio and Giovanni Agosta, "Improved Programming of GPU Architectures through Automated Data Allocation and Loop Restructuring," in *Proceedings of the 2PARMA Workshop (ARCS2010 Workshop)*, Feb. 2010.

[5] "Ultra high-speed Mobile Information and Communication systems (UMIC) research center, RWTH Aachen University," http://www.umic.rwth-aachen.de/.

[6] V. Ramakrishnan, E. M. Witte, T. Kempf, D. Kammler, G. Ascheid, H. Meyr, M. Adrat, and M. Antweiler, "Efficient And Portable SDR Waveform Development: The Nucleus Concept," in *IEEE Military Communications Conference (MILCOM 2009)*, Oct. 2009.

[7] S. Campanoni, G. Agosta, S. Crespi-Reghizzi, and A. D. Biagio, "A highly flexible, parallel virtual machine: design and experience of ildjit," *Softw., Pract. Exper.*, vol. 40, no. 2, pp. 177–207, 2010.

[8] Michele Tartara, Simone Campanoni, Giovanni Agosta and Stefano Crespi Reghizzi, "Parallelism and Retargetability in the ILDJIT Dynamic Compiler," in *Proceedings of the 2PARMA Workshop (ARCS2010 Workshop)*, Feb. 2010.

[9] Z. Ma, P. Marchal, D. P. Scarpazza, P. Yang, C. Wong, J. I. Gmez, S. Himpe, C. Ykman-Couvreur, and F. Catthoor, *Systematic Methodology for Real-Time Cost-Effective Mapping of Dynamic Concurrent Task-Based Systems on Heterogenous Platforms*. Springer Publishing Company, Incorporated, 2007.

[10] A. Bartzas, M. Peon-Quiros, C. Poucet, C. Baloukas, S. Mamagkakis, F. Catthoor, D. Soudris, and J. M. Mendias, "Software metadata: Systematic characterization of the memory behaviour of dynamic applications," *Journal of Systems and Software*, vol. In Press, Corrected Proof, pp. –, 2010.

[11] P. Bellasi, W. Fornaciari, and D. Siorpaes, "A hierarchical distributed control for power and performances optimization of embedded systems," in *ARCS*, ser. Lecture Notes in Computer Science, C. Müller-Schloer, W. Karl, and S. Yehia, Eds., vol. 5974. Springer, 2010, pp. 37–48.

[12] H. Chang, L. Cooke, M. Hunt, G. Martin, A. J. McNelly, and L. Todd, *Surviving the SOC revolution: a guide to platform-based design*. Norwell, MA, USA: Kluwer Academic Publishers, 1999.

[13] ARTEMIS Strategic Research Agenda Working Group, "Strategic Research Agenda: Design Methods and Tools," ARTEMIS, Tech. Rep., 2006. [Online]. Available: https://www.artemisia-association.org/downloads/RAPPORT_DMT.pdf

[14] M. Duranton, S. Yehia, B. D. Sutter, K. D. Bosschere, A. Cohen, B. Falsafi, G. Gaydadjiev, M. Katevenis, J. Maebe, H. Munk, N. Navarro, A. Ramirez, O. Temam, and M. Valero, "The HiPEAC 2012-2020 vision," HiPEAC, Tech. Rep. [Online]. Available: http://www.hipeac.net/roadmap

[15] H. Hübert and B. Stabernack, "Profiling-based hardware/software co-exploration for the design of video coding architectures," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 19, no. 11, pp. 1680–1691, 2009.

[16] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the h.264/avc standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 17, no. 9, pp. 1103 –1120, Sep. 2007.

[17] J. Ansari, X. Zhang, A. Achtzehn, M. Petrova, and P. Mhnen, "Decomposable MAC Framework for Highly Flexible and Adaptable MAC Realizations," Demonstration abstract. Proc. of DySPAN (Singapore), 2010.

[18] Claudia Cormio and Kaushik R. Chowdhury, "A survey on MAC protocols for cognitive radio networks," *Ad Hoc Networks*, vol. 7, no. 7, pp. 1315–1329, 2009.

[19] K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 19, no. 7, pp. 1073–1079, 2009.