A Multi-Core Reconfigurable Architecture for Ultra-Low Power Bio-Signal Analysis

Loris Duch, Soumya Basu,
Rubén Braojos, David Atienza
Embedded Systems Laboratory
EPFL, Switzerland
Email: {loris.duch, soumya.basu,
ruben.braojoslopez, david.atienza}@epfl.ch

Giovanni Ansaloni,
Laura Pozzi
Università della Svizzera Italiana
Lugano, Switzerland
Email: {giovanni.ansaloni,
laura.pozzi}@usi.ch

***Abstract*—This paper introduces a novel computing architecture devoted to the ultra-low power analysis of multiple bio-signals. Its structure comprises several processors interfaced with a shared acceleration resource, implemented as a Coarse Grained Reconfigurable Array (CGRA). The CGRA supports the efficient execution of the computationally intensive kernels present in this application domain, while requiring a low reconfiguration overhead. The run-time behavior of the resulting heterogeneous system is orchestrated by a light-weight hardware mechanism, which concurrently synchronizes processors and regulates access to the reconfigurable accelerator. The architecture achieves speed-ups of up to 11x on different bio-signal processing kernels and system-level energy savings of up to 18.6%, with respect to a multi-core platform, which does not feature CGRA acceleration.**

I. Introduction

Chronic cardiac diseases require the long-term monitoring of affected patients, which impacts the quality of life of subjects and presents a high financial burden for healthcare providers [19]. In this context, Wireless Body Sensor Nodes (WBSNs) are an important technological aid, as they allow the continuous acquisition of bio-signals with little supervision from the medical staff, even outside of a hospital environment.

Today's Wireless Body Sensor Nodes (WBSNs) embed complex Digital Signal Processing (DSP) routines to extract high-level features from bio-signal acquisitions [5]. These "smart" WBSNs transmit only features (as opposed to samples) through the energy-hungry wireless link, resulting in large efficiency gains, thus enabling longer, less obtrusive and more clinically-relevant acquisitions. Nonetheless, these benefits can only be leveraged by performing the DSP stage within a tiny energy envelope.

As a consequence, herein we propose a domain-specific platform that operates at ultra-low power levels, harnessing the opportunities offered by the application characteristics typical of biomedical DSP. First, as processing is usually divided in well-defined phases, the (possibly parallel) workload is spread over different computing cores, similarly to [7]. Second, hardware acceleration is provided to efficiently execute the *computational kernels*, i.e.: compact and intensive code sections, which account for a vast portion of the overall DSP run-time.

To maintain a high degree of efficiency without sacrificing flexibility, the accelerator is programmable at the operation level, as a Coarse Grained Reconfigurable Array (CGRA) mesh [13]. In this way, a variety of kernels, possibly unknown at design time, can be supported, avoiding the high area, energy and configuration overheads, typical of the fine-grained reconfigurability provided by FPGAs. The CGRA mesh is interfaced as a shared resource, time- and space-multiplexed among the processors.

The run-time management of the different processing resources in such heterogeneous system is not a trivial task, especially when, as in the proposed platform, it must be supported with minimal area, energy and timing overheads. To this end, a dedicated synchronizer is employed to a) manage the computations on the multiple cores and b) manage the acceleration requests and their execution on the reconfigurable architecture.

The energy benefit deriving from our approach is two-fold. By separating the computation- and control-intensive parts of applications, each of them is efficiently mapped on dedicated resources. Moreover, the speed-ups ensuing from hardware acceleration decreases the ratio between active and idle times, which can then be leveraged by supporting aggressive deep-sleep modes.

The contributions of the paper are the following:

1) We introduce and evaluate a heterogeneous system devoted to bio-signal processing, which integrates multiple processors and a shared CGRA accelerator.
2) We propose a unified mechanism to jointly support synchronization among cores, acceleration of kernels, and power management at the system level with very low overhead.
3) We showcase the efficiency of the developed system while executing complex bio-signal DSP on ECG acquisitions, such as applications for filtering, classification and feature extraction.

II. State of the Art

To sustain bio-signal processing workloads [14] at ultra-low power levels, a number of domain-specific processors have been proposed, usually operating at Near-Threshold

Voltages (NTV) [2], [17]. To cope with the performance loss deriving from NTV supply levels, the authors of [10] and [7] adopted multiple processing cores, leveraging the application-level parallelism which characterizes bio-signal DSP. An orthogonal strategy focuses instead on dedicated hardware blocks (custom instructions [9] or accelerators [11]) to support computationally-intensive kernels. While efficient, this strategy is very inflexible, as each block can perform a single function.

Reconfigurable solutions are good candidates to couple the efficiency typical of dedicated hardware with a degree of flexibility. However, bit-level reconfigurable arrays (such as FPGAs) present huge overheads in terms of area, reconfiguration time, and power consumption. CGRAs dramatically reduce these overheads by being programmable only at the operation level, allowing efficient mapping of computational kernels [13], [4].

We exploit the parallel nature of coarse-grained reconfiguration by interconnecting a CGRA instance as a shared accelerator in a multi-core system. Our approach has some similarities with the one adopted in [8]. Nonetheless, the authors of this work adopt the limiting assumption that the reconfigurable fabric can be accessed only by one core at a time. Conversely, our platform concurrently supports the execution of multiple kernels, arbitrating acceleration requests at run-time.

III. Shared CGRA accelerator

CGRA architectures are structured as two-dimensional meshes of tightly interconnected Reconfigurable Cells (RCs). RCs embed a dedicated ALU coupled with a small local register file. This arrangement allows CGRAs to efficiently execute intensive innermost loops. By modulo-scheduling loops on the mesh, their execution can be effectively parallelized across subsequent iterations [3].

The configuration overhead of CGRAs, as well as the area devoted to the configuration logic, is orders-of-magnitude smaller than that of fine-grained FPGAs, as only the desired ALU operations and the routing of operands must be specified for each cell. Multiple operations can be cyclically performed providing a set configuration words for each RC, and activating the proper one during execution [1].

Figure 1 provides a high-level view of the envisaged CGRA mesh. Each of its cells features a datapath (DP), which is composed of an ALU, a 4-word register file, and multiplexers able to select the input operands (either from the register file, from the ALU output or from the outputs of neighboring cells). The ALU can execute arithmetic and bit-wise operations (AND, OR, XOR, etc.). The CGRA is interfaced to the system data memory by means of a multi-channel DMA block. This unit uses the memory ports of the processors that requested a given acceleration, and therefore do not require dedicated read and write ports toward the memory subsystem.

At run-time, each kernel being mapped in the CGRA undergoes a configuration and an execution phase. During configuration, the parameters of the kernel invocation (such as the addresses of inputs and outputs in data memory and the number of iterations) are retrieved from the issuing processors and used to configure the program counters of the employed columns and the required DMA channels. While a single kernel can be configured at a time, execution of different kernels can instead proceed concurrently on separate CGRA columns, effectively employing the available computing resources.

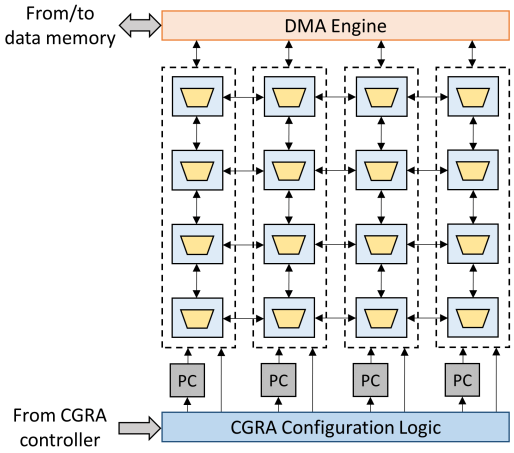

Fig. 1: Block scheme of the shared coarse-grained reconfigurable accelerator.

During execution, the functionality of RCs is dictated by their active configuration word, selected on a cycle-by-cycle basis by column-wise Program Counters (PCs). After the RCs have finished the computations, the desired outputs are stored by the DMA engine in the system data memory.

IV. Reconfigurable Multi-Core System

Similarly to [7], our target platform (Figure 2) embeds 8 RISC processors interconnected to multi-banked instruction (8 banks) and data (16 banks) memories through combinational crossbars. Each processor adopts a Harvard architecture, featuring a three-stage pipeline. Processors can be individually clock-gated by a light-weight synchronizer unit while waiting

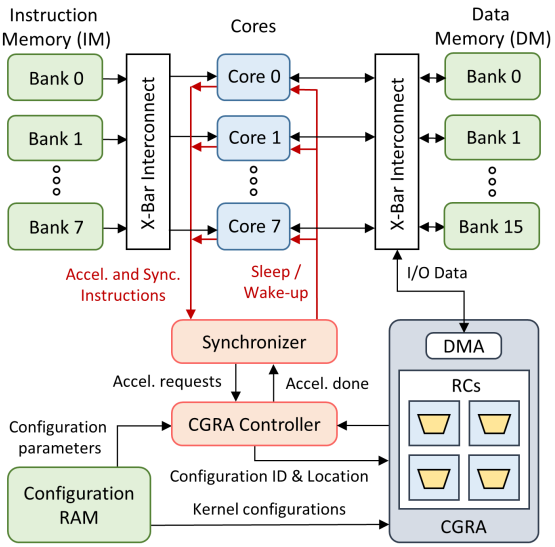

Fig. 2: High-level view of the heterogeneous platform.

for another core to finish its task or when a kernel acceleration is performed on the CGRA.

The CGRA Controller arbitrates acceleration requests through a request queue, mapping kernels (whose configuration words are stored in a dedicated Configuration RAM) when enough resources are available.

At the software (instruction set) level, the architecture features the synchronization instructions introduced in [7], which support SIMD execution modes and the management of producer-consumer relationships between threads. A further instruction set extension allows processors to request the execution of a kernel on the CGRA mesh. It is defined as ACCEL *#literal*, where the literal specifies the kernel ID. The following kernel parameters are communicated to the CGRA controller by setting memory mapped registers:

- The address and the length of the input data to be processed by the kernel running on the CGRA.
- The address and length of the destination buffer where to store the values computed by the kernel.
- The number of required loop iterations.

A processor issuing an ACCEL instruction is clock-gated by the synchronizer. CGRA execution is then initiated by the controller: as soon as resources (i.e.: RCs) are available on the mesh, an acceleration request signal is sent to the CGRA unit, along with the acceleration ID and other configuration parameters. Afterwards, the CGRA itself fetches the remaining configuration words from the Configuration RAM and programs the RCs of the assigned columns. When the kernel is mapped, the execution starts. Upon its completion, the requesting processor exits the clock-gated state and software execution can proceed.

V. Experimental Setup and Simulation Results

In order to evaluate the energy and performance benefits of the heterogeneous multi-core system, we developed a hybrid framework, comprising an HDL implementation of the CGRA mesh, which is used to accurately characterize its architecture from a timing and energy perspective. Retrieved data, in conjunction with the post-synthesis evaluation of the processing and memory elements described in [7], was then employed in a cycle accurate SystemC simulator of the whole platform, which allowed evaluation of the system across the execution of entire applications. We considered an implementation on a 65 nm UMC low-leakage cell library and an operating frequency of 1 MHz.

We assessed the system performance employing three electrocardiogram processing benchmarks, which present different workloads and computational characteristics. The applications perform multi-lead morphological ECG filtering (3L-MF [18]), multi-lead ECG delineation (3L-MMD [16]) and selective processing based on heartbeat classification (RP-CLASS [6]). The employed ECG records, consisting of excerpts of 5000 samples acquired at 500Hz, are extracted from the MIT-BIH Arrhythmia database [15].

In order to identify the application kernels and to inspect their structures in terms of control and data flow, we used a profiling pass built on top of the LLVM toolchain [12]. While automated strategies for mapping kernels on CGRAs have been proposed [3], this step was manually performed for this work. Seven different computationally intensive kernels where considered, as reported in Figure 3. Several kernels are common for the three benchmarks, while others are only present in a subset of the target applications.

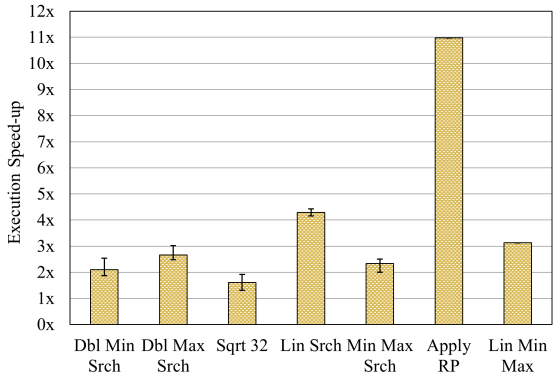

Fig. 3: Speed-ups of kernels running on the CGRA mesh with respect to their software execution.

To investigate the benefits deriving from our approach, we compared the proposed architecture with an equivalent multi-core platform that does not feature the CGRA accelerator. We first analyzed the resulting performance at the kernel level, evaluating savings from both run-time and energy perspectives.

As shown in Figure 3, execution on the CGRA mesh achieves speed-ups ranging from 1.6x to 11.0x while executing the selected kernels, compared to a software-only alternative. The reported results account for resource conflicts, which arise when several concurrent requests cannot be allocated at the same time on the limited CGRA resources.

The considerable time reductions achieved are coupled with a superior energy efficiency of the CGRA unit, which is represented in Figure 4. This figure compares the energy consumed by executing the selected kernels on the multi-core system and on the CGRA. It shows that by accelerating the kernels on the CGRA it is possible to achieve energy savings of up to 94.9% when compared to a software-only execution, with an average reduction of 73.3%.

At the system level, the CGRA acceleration of just few kernels per application results in a sizeable reduction of the active times of processors (as highlighted in Figure 5), leading to an increase in overall energy efficiency of the platform. In fact, by kernels on the CGRA, not only the dynamic energy of the cores is decreased, but also fewer accesses to the instruction and data memories are required. In the case of the 3L-MF benchmark, this effect is particularly noticeable, as the active time of cores is reduced from 50.4% to 31.3%.

The resulting energy savings are detailed in Figure 6, which provides the consumption breakdown of the multi-core system with and without the CGRA accelerator for the three investigated benchmarks. The comparison showcases that the energy consumed by the cores and the instruction memory is reduced by a large margin when the CGRA is employed, as a

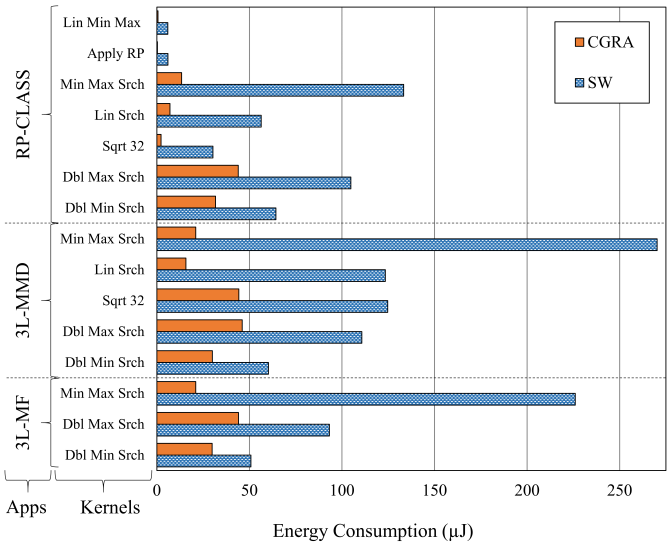

Fig. 4: Energy consumed by the different kernels employed in the considered benchmarks, when executed on the accelerator (CGRA) and on the processing cores (SW).

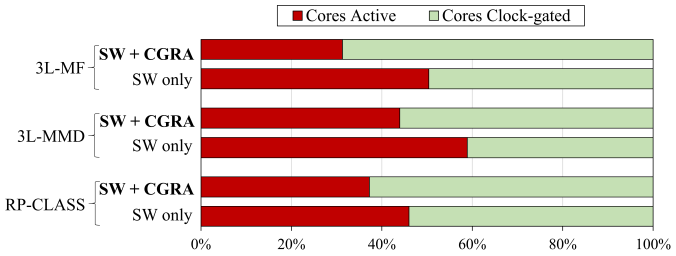

Fig. 5: Multi-core utilization time with and without CGRA acceleration (% of total run-time).

considerable part of the applications workloads are outsourced to the CGRA and a much smaller amount of instructions are fetched at runtime. For all applications, these reductions in energy consumption more than compensate the overhead associated with the inclusion of the CGRA accelerator, resulting in a decrease of the overall system energy budget, including both dynamic and static consumption, of up to 18.6% (for the 3L-MF case).

VI. Conclusions

In this paper we have introduced a heterogeneous and ultra-low power architecture devoted to bio-signal processing applications. In the medical domain, the workload of applications is often divided between control-dominated phases and computationally-intensive phases within compact loops (kernels). The illustrated platform can efficiently support both: the former on multiple ultra-low power processing cores, the latter by employing a coarse-grained reconfigurable array, interfaced to the cores as a shared acceleration resource. The above-mentioned features allow the developed system to achieve tangible overall energy savings of up to 18.6%, when executing complex bio-signal processing applications, in comparison to an equivalent multi-core solution without CGRA acceleration of kernels.

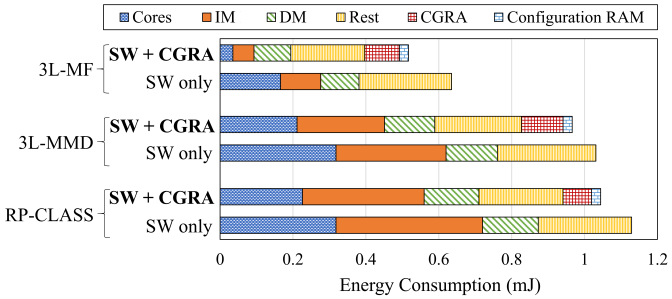

Fig. 6: System energy consumption for the different applications, while executing on the multi-core platform with and without CGRA acceleration.

Acknowledgment

This work has been partially supported by the E4Bio (no. 200021_159853) RTD project evaluated by the Swiss NSF.

References

[1] G. Ansaloni et al. EGRA: A coarse grained reconfigurable architectural template. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 19(6):1062–1074, 2011.

[2] M. Ashouei et al. A voltage-scalable biomedical signal processor running ecg using 13pJ/cycle at 1MHz and 0.4V. In *ISSCC*, pages 332–334, Feb 2011.

[3] M. Bingfeng et al. Exploiting loop-level parallelism on coarse-grained reconfigurable architectures using modulo scheduling. In *Proc. DATE*, pages 296–301, 2003.

[4] F. Bouwens et al. Architectural exploration of the adres coarse-grained reconfigurable array. In *Reconfigurable Computing: Architectures, Tools and Applications*, pages 1–13. Springer, 2007.

[5] R. Braojos et al. Embedded real-time ECG delineation methods: A comparative evaluation. In *IEEE-BIBE*, pages 99–104. IEEE, 2012.

[6] R. Braojos et al. A methodology for embedded classification of heartbeats using random projections. In *Proc. DATE*, pages 899–904, March 2013.

[7] R. Braojos et al. Hardware/software approach for code synchronization in low-power multi-core sensor nodes. In *Proc. DATE*, pages 1–6. IEEE, 2014.

[8] L. Chen et al. Shared reconfigurable fabric for multi-core customization. In *Proc. DAC*, pages 830–835. ACM, 2011.

[9] J. Constantin et al. TamaRISC-CS: An ultra-low-power application-specific processor for compressed sensing. In *VLSI-SoC*, pages 159–164. IEEE, 2012.

[10] Y. He et al. Xetal-Pro: An ultra-low energy and high throughput SIMD processor. In *Proc. DAC*, pages 543–548. ACM, 2010.

[11] J. Kwong et al. An energy-efficient biomedical signal processing platform. *JSSC*, 46(7):1742–1753, 2011.

[12] C. Lattner et al. Llvm: A compilation framework for lifelong program analysis & transformation. In *Proc. CGO04*, Palo Alto, California, Mar 2004.

[13] M. Lee et al. Design and implementation of the MorphoSys reconfigurable computing processor. *VLSI signal processing systems for signal, image and video technology*, 24(2-3):147–164, 2000.

[14] A. Pantelopoulos et al. A survey on wearable sensor-based systems for health monitoring and prognosis. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(1):1–12, 2010.

[15] PhysioBank. http://www.physionet.org/physiobank/.

[16] F. Rincon et al. Development and evaluation of multilead wavelet-based ECG delineation algorithms for embedded wireless sensor nodes. *IEEE T-ITB*, 15(6):854–863, Nov 2011.

[17] S. R. Sridhara et al. Microwatt embedded processor platform for medical system-on-chip applications. *JSSC*, 46(4):721–730, 2011.

[18] Y. Sun et al. ECG signal conditioning by morphological filtering. *CBM*, 32(6):465–479, 2002.

[19] World Health Organization. Cardiovascular diseases. http://www.who.int/topics/cardiovascular_diseases/en, 2016.