Hardware/Software Architecture for Real-Time ECG Monitoring and Analysis Leveraging MPSoC Technology

Iyad Al Khatib[1], Davide Bertozzi[2], Francesco Poletti[3], Luca Benini[3], Axel Jantsch[1], Mohamed Bechara[4], Hasan Khalifeh[4], Mazen Hajjar[4], Rustam Nabiev[5], and Sven Jonsson[5]

[1] ECS, ICT, Royal Institute of Technology (KTH), Stockholm, Sweden
{iyad, axel}@imit.kth.se
[2] Engineering Department, University of Ferrara, Ferrara, Italy
dbertozzi@ing.unife.it
[3] DEIS, University of Bologna, Bologna, Italy
{fpoletti, lbenini}@deis.unibo.it
[4] ECE, FEA, American University of Beirut, Beirut, Lebanon
{mfb05, hik05, mah39}@aub.edu.lb
[5] Biomedical Engineering Department, Karolinska University Hospital
Huddinge, Stockholm, Sweden
{rustam.nabiev, sven.jonssoni}@karolinska.se

Abstract. The interest in high performance chip architectures for biomedical applications is gaining a lot of research and market interest. Heart diseases remain by far the main cause of death and a challenging problem for biomedical engineers to monitor and analyze. Electrocardiography (ECG) is an essential practice in heart medicine. However, ECG analysis still faces computational challenges, especially when 12 lead signals are to be analyzed in parallel, in real time, and under increasing sampling frequencies. Another challenge is the analysis of huge amounts of data that may grow to days of recordings. Nowadays, doctors use eyeball monitoring of the 12-lead ECG paper readout, which may seriously impair analysis accuracy. Our solution leverages the advance in multi-processor system-on-chip architectures, and it is centered on the parallelization of the ECG computation kernel. Our Hardware-Software (HW/SW) Multi-Processor System-on-Chip (MPSoC) design improves upon state-of-the-art mostly for its capability to perform real-time analysis of input data, leveraging the computation horsepower provided by many concurrent DSPs, more accurate diagnosis of cardiac diseases, and prompter reaction to abnormal heart alterations. The design methodology to go from the 12-lead ECG application specification to the final HW/SW architecture is the focus of this paper. We explore the design space by considering a number of hardware and software architectural variants, and deploy industrial components to build up the system.

Keywords: Multiprocessor System-on-Chip, embedded system design, HW/SW, electrocardiogram algorithms, real-time analysis, hardware space exploration.

P. Stenström (Ed.): Transactions on HiPEAC I, LNCS 4050, pp. 239–258, 2007.

1 Introduction

Despite the ongoing advances in heart treatment, in the United States [1] and Canada [2] as well as in many other countries, the various forms of cardiovascular disease (CVD) and stroke remain by far the number one cause of death for both men and women regardless of ethnic backgrounds. According to the World Health Organization (WHO) Report in 2003, 29.2% of total global deaths are due to CVD, many of which are preventable by action on the major primary risk factors and with proper monitoring [1]. It is estimated that by 2010, CVD will be the leading cause of death in developing countries. Since the rate of hospitalization increases with age for all cardiac diseases [3], a periodic cardiac examination is recommended. Hence, more efficient methods of cardiac diagnosis are desired to meet the great demand on heart examinations. However, state-of-the-art biomedical equipment for heartbeat sensing and monitoring lacks the ability of providing large-scale analysis and remote, real-time computation at the patient's location (point of need). The intention of this work is to use MPSoC microelectronic technology to meet the growing demand for telemedicine services, especially in the mobile environment. The project attempts to address the existing problem of reducing the costs for hospitals/medical-centers through using MPSoC-based designs that may replace biomedical machines and have higher quality, reduce the nurse's and doctor's work-load, and improve the quality of healthcare for patients suffering from heart diseases by exploring one potential solution. From the hospital side, deploying this solution will further reduce the costs of rehabilitating and following up on patients "primary care" since it allows better home-care. Home-care ensures continuity of care, reduces hospitalization costs, and enables patients to have a quicker return to their normal life styles. From a technical viewpoint, real-time processing of ECG data would allow a finer-granularity analysis with respect to the traditional eyeball monitoring of the paper ECG readout. Eventually, warning or alarm signals could be generated by the monitoring device and transmitted to the healthcare center via telemedicine links, thus allowing for a prompter reaction of the medical staff. In contrast, heartbeat monitoring and data processing are traditionally performed at the hospital, and for long monitoring periods a huge amount of collected data must be processed offline by networks of parallel computers. New models of healthcare delivery [2] are therefore required, improving productivity and access to care, controlling costs, and improving clinical outcomes. This poses new technical challenges to the design of biomedical ECG equipment, calling for the development of new integrated circuits featuring increased energy efficiency while providing higher computation capabilities.

The fast evolution of biomedical sensors and the trend in embedded computing are progressively making this new scenario technically feasible. Sensors today exhibit smaller size, increased energy efficiency and therefore prolonged lifetimes (up to 24 hours) [4], higher sampling frequencies (up to 10 kHz for ECG) and often provide for wireless connectivity. Unfortunately, a mismatch exists between advances in sensor technology and the capabilities of state-of-the-art heart analyzers [5][6][7]. They cannot usually keep up with the data acquisition rate, and are usually wall-plugged, thus preventing for mobile monitoring. On the contrary, the deployment of wearable devices such as SoC devices has to cope with the tight power budgets of such devices, potentially cutting down on the maximum achievable monitoring period. In this paper

we propose a wearable multi-processor biomedical-chip for electrocardiogram (MPSoC ECG biochip) paving the way for portable real-time electrocardiography applications targeting heart disorders. The biochip leverages the computation horsepower provided by many (up to twelve) concurrent DSPs and is able to operate in real-time while performing the finest granularity analysis as specified by the ECG application. Moreover, in case of heart failure emergency aid should arrive in a period of few minutes from the time when the heart failed, otherwise brain damage may occur. Hence, real time analysis must be done in few seconds to allow the alarm signal to reach the emergency aid team, which should act immediately. The biochip system builds upon some of the most advanced industrial components for MPSoC design (multi-issue VLIW DSPs, high-throughput system interconnect and commercial off-the-shelf biomedical sensors), which have been composed in a scalable and flexible platform. Therefore, we have ensured its reusability for future generations of ECG analysis algorithms and its suitability for porting of other biomedical applications, in particular those collecting input data from wired/wireless sensor networks [8].

This article builds upon the results of a paper from the ACM International Conference on Computing Frontiers 2006 [9]. We present our investigation that goes through all the steps of the design process, from application functional specification to hardware modeling and optimization. System performance has been validated through functional, timing accurate simulation on a virtual platform. We point out the need for simulation abstractions matching the application domain. A 0.13µm technology-homogeneous power estimation framework leveraging industrial power models is used for power management considerations [10][11]. The paper presents the process of software functional specification, optimization and parallelization, as well as the results of the hardware design space exploration, which leads to the final performance- and energy-optimized solution.

2 Biomedical Background

The electrocardiogram (ECG) is an electrical recording of the heart activity that is used as a diagnosis tool by physicians and doctors to check the status of the heart. The most commonly used way to detect the heart status is the 12-lead ECG technique. This technique uses nine sensors on the patient's body (Fig. 1). The three main sensors are distributed by: placing one sensor on the left arm (LA), a second sensor on the right arm (RA), and a third sensor on the left leg (LL). The right leg (RL) is connected by only a wire to be used as ground for the interconnected sensors. By only having these three sensors physicians can use a method known as the 3-lead ECG, which suffers from the lack of information about some parts of the heart but is useful for some emergency cases to have quick analysis. In this respect, medical doctors require more sensors (i.e., more leads). Hence, six more sensors (V1-V6) are added on the chest (Fig. 1). The voltages V1-V6 are measured with respect to *Ground* (G) on the right leg (RL). In some cases, physicians use these six chest-placed sensors to analyze the heart. Using all the nine sensors and interconnecting them for the 12-lead ECG gives twelve signals known in biomedical terms as: Lead I, Lead II, Lead III, aVR, aVL, aVF, V1, V2, V3, V4, V5, and V6 (Fig. 1). The 12-lead ECG produces

huge amounts of data especially when used for a long number of hours. Physicians use the 12-lead ECG method, because it allows them to view the heart in its three dimensional form; thus, enabling detection of any abnormality that may not be apparent in the 3-lead or 6-lead ECG technique. Fig. 2 shows an explanatory example of a typical ECG signal. The most important points on the ECG signal are the peaks: P, Q, R, S, T, and U. Each of these peaks is related to a heart action that is of importance to the medical analysis. Figure 3 shows real recorded signals from 12-leads, which are printed on the eyeballing paper.

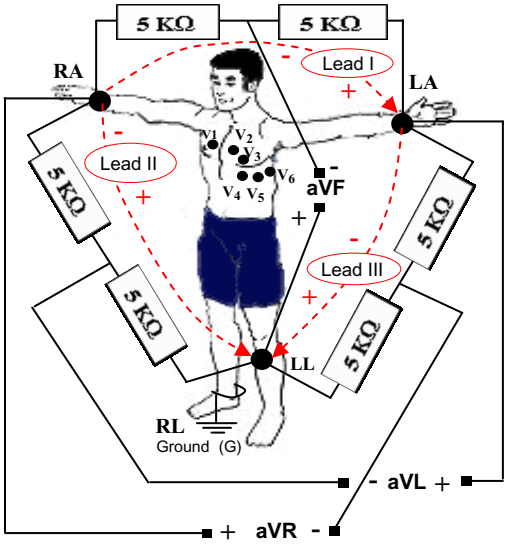


Fig. 1. 12-lead ECG: RA, LA, LL, & RL are the right arm, left arm, left leg, and right leg sensors; RL is grounded (G)

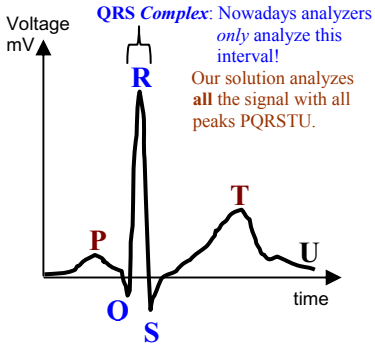


Fig. 2. Ideal ECG Signal for lead I

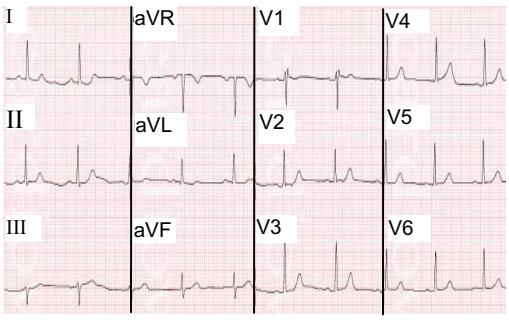


Fig. 3. Complete paper readout, which is not accurate to see peaks nor easy to read for long recordings

This paper printout is the classical medical technique used for looking at ECG signals, and it is still used. However, the eyeballing paper print makes the check of the different heart peaks and rhythms difficult and inaccurate due to its dependence on the physician's eyes. On the other hand, when using digital recording and filtering we can determine the peaks more accurately. Consequently, we can use digital computing to process the sensed data and analyze the heart beat. In addition, there are normal medical ranges for the inter-peak time intervals, and every combination of different inter-peak intervals proves a type of heart illness. The most important of the peaks is the R peak, which refers to the largest heart blood pump.

3 Previous Work

Electrocardiogram methods for heart analyses have been one of the most important medical practices, hence, the monitoring and analyses of ECG signals have not only gone through a lot of research work, but also many companies have investigated and worked on commercial solutions.

However, we are not aware of any solution in the research or the commercial markets that is composed of a single-chip real-time analysis solution for full 12-lead ECG, and that is able to estimate the heart period independent of the peak signals and, at the same time diagnose all the peaks: P, Q, R, S, T and U and their inter-peak intervals to result in disease diagnosis. Most of the work done involves only recording huge amounts of data in large storage media and then analyzing the stored data, but not allowing the ease of patient mobility. Most of the time, the patient has to be confined to a bed for a number of hours (could be for a whole day). Some commercial solutions are only capable of concluding if the heart beat is normal or abnormal but can not specify the period nor could they diagnose the disease. Other real time solutions available in the market, in healthcare institutes, and in research organizations, are *only* capable of sensing and transmitting ECG data [12] to: either a local machine [13] or to a distant healthcare center [14]. In both cases, the work that is executed involves checking if the heart beat is healthy or unhealthy without analyzing the disease and not in real-time. Moreover, the commercial solutions under study [15] do not look into the parallelization of the ECG analysis into multiple cores, so to speed up processing.

4 Sensing and Filtering Stage

ECG analysis requires three main phases: (i) acquiring the signals from the leads, (ii) filtering the lead-signals (each alone), and (iii) analysis (Fig. 4). Firstly, the sensing phase requires an A/D converter in order to be able to have digital data for our digital filter. We use 16 bit A/D converters, because our analysis algorithm and ECG biochip are designed based on having 16-bit filtered data as input. We briefly discuss the filtering method we use as an essential part of our proposed solution, and then we discuss the biochip design that depends on this filtering step.

The high investment in sensor technology and biomedical research in general gave the birth to biomedical sensors that have more advanced features than the commercial available ones just a few years ago. For instance, the nowadays sensors are characterized by prolonged lifetimes (up to 24 hours), and higher sampling frequencies (up to 10 kHz for ECG). Some sensor companies have produced wireless biomedical sensors in order to aid patient mobility [4]. This advance in biomedical sensors faces a mismatch with biomedical heartbeat analyzers that still lack behind to cope with the huge amounts of data, the high rates, and the wireless features that modern sensors can provide [6]. In our work, many sensors may be chosen, and for the moment we choose the sensors that can serve our real-time aim and that have reasonable prices for the market success of the solution, hence we choose the state of the art commercial sensor from Ambu Inc. silver/silver chloride "Blue Sensor R" [4] shown in Fig. 4. It is characterized by: 24 hour lifetime, superior adhesion, optimal signal measuring during stress tests. It is small to carry (57mm x 48mm), and it is easily wearable.

On the other hand, even the state of the art sensors suffer from the usual problems that most biomedical sensors suffer from. For instance, data provided by biomedical sensors suffers from several types of noise: physiological variability of *QRS complexes* (The *QRS Complex* is shown in Fig. 2), baseline wander, muscle noise, artifacts due to electrode motion, power-line interference [16]. The presence of several noise sources might impair ECG analysis accuracy, as showed in the R-Peak detection marked by circled areas in Fig. 5. Two peaks may be detected where there should be only one. In order to deal with noisy input signals, we designed an IIR filter with order 3 that outputs its results in 16-bit binary format (Fig. 4).

However, we need to be aware of the fact that we want to look in our solution at high sampling frequencies (250Hz, 1000Hz and above), because we want to: (a) make use of the available accuracy of the state of the art sensors, (b) have finer granularity of data, and (c) get more accurate analysis since in some cases more data samples are needed to discover a disease; like, for instance, the medical case known as the R on T phenomena [17], where the R and the T peaks are very near in time so we need a very high number of samples and an intelligent algorithm to discover them. Moreover, it is extremely important to choose a sampling frequency that minimizes the risk of aliasing. The highest frequency needed for the ECG signal is 90Hz (due to the medical frequencies of the heart), which implies that the lowest sampling frequency that can be used is equal to the Nyquist rate (180Hz). However, in order to sample at such a frequency, the analogue signal has to be band limited to 90Hz, which can be achieved by the use of a complex analogue bandpass filter with a very sharp frequency response. This solution, although advantageous on limiting the amount of data to be stored, has a disadvantage on the analogue side, since the bandpass filter, being complex in order to meet the sharpness requirement, will probably have a considerable power consumption. An alternative solution would be to sample at a frequency much higher than the Nyquist rate, such that the analogue bandpass filter can have a relaxed frequency response, while still effectively filtering out the frequencies that would cause aliasing during sampling. For instance, by choosing a sampling frequency of 5kHz, all frequencies beyond 2.5kHz would have to be filtered out before sampling, but that task is simpler than before, since all frequencies between

90Hz and 2.5kHz can be attenuated without affecting the data needed for analysis. After sampling, band limitation to 90Hz can be implemented using a digital filter. This approach has the advantage of using a lower-complexity bandpass filter, and reducing considerably the risk of aliasing and folding. Moreover, increasing the number of samples increases the accuracy of the sample, and makes the overall filtered signal smoother when used for analysis.

Our IIR filter is built to deal with these problems. Another main advantage of using the IIR filter is to eliminate the noise that is directly proportional to the DC offset of the sensed ECG [16], which is around 0.1mv. The two plots in Fig. 5 clearly show how the filtering algorithm remedies this problem. In our implementation, the filter is implemented in hardware on a dedicated chip feeding the external SDRAM memory of our biochip. Our filter is the convolution of the noisy signal with the filter impulse response given in (1):

$$y[n] = \sum_{k} h[k] \times x[n-k]. \tag{1}$$

where, $x[n]$ is the noisy signal, $h[n]$ is the filter impulse response, and n is the sample index. This filter in (1) is also an infinite impulse response (IIR, Chebyschev filter), so it can be written as (2):

$$y[n] = \sum_{k} x[n-l] \times b[l] - \sum_{m=1} y[n-m] \times a[m]. \tag{2}$$

where, y is the output of the filter and x is the input, b is the vector that contains the filter coefficients for signal x, and a is the vector that contains the filter coefficients for output y.

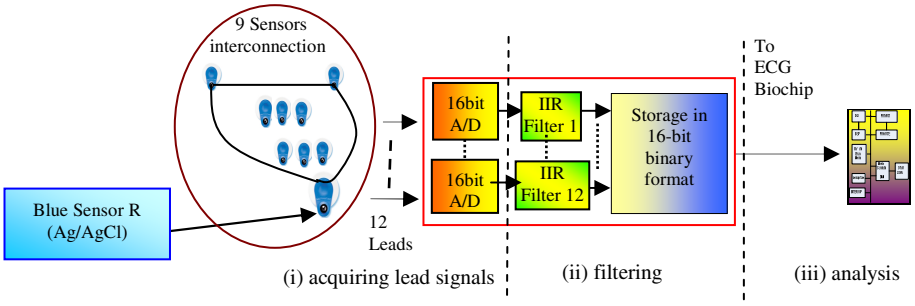


Fig. 4. The System for sensing and filtering of ECG lead signals before sending data to the ECG Biochip for analysis. Blue Sensor R is from Ambu Inc. [4].

The upper limits of the coefficients are dependent on the order of the filter being used. Our IIR filter is of order 3, because our ECG data does not require higher orders. We

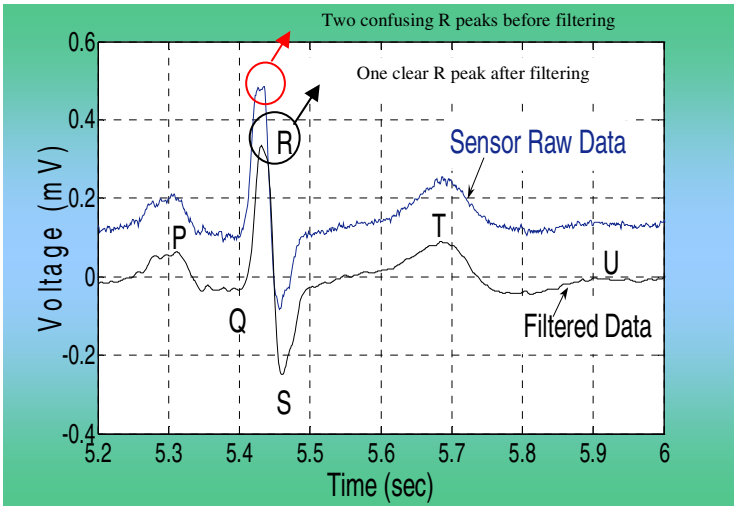


Fig. 5. ECG raw and filtered data (lead I)

can improve our filter (when needed) by simply knowing the needed values of the coefficients in vectors *a*[.] and *b*[.].

5 ECG Algorithm

Most ECG systems make use of the Pan-Tompkins analysis algorithm [18], which targets QRS complexes (Fig. 2) detection and consists of the cascade of four filters: (i) band pass, (ii) differentiator, (iii) squaring operation, and (iv) a moving window integrator. In principle, traditional ECG analysis starts from a reference point in the heart cycle (the R-peak is commonly used as the reference point). As a consequence, accurate detection of the R-peak of the QRS complex is a prerequisite for the reliable functionality of ECG analyzers [18]. However, as an effect of ECG signal high variability, R-peak detection might be inaccurate. For instance, in the R on T phenomena, a T peak may be wrongly taken for an R peak, and then the R-T interval will be considered as an R-R interval, and the period will be wrong. Hence, other QRS parameters will be consequently inaccurate. As a result, traditional techniques may fail in detecting some serious heart disorders such as the R-on-T phenomenon (associated with premature ventricular complexes) [17].

Our approach takes a different perspective: instead of looking for the R-peaks and then detecting the period, we detect the period first (via autocorrelation) and then look for the peaks. We use an autocorrelation function (ACF) to calculate the heartbeat period without looking for peaks. Then, we can restrict our analysis to a time window equal to the period and detect all peaks. Although potentially more accurate, our algorithm incurs a higher computational complexity: 3.5 million multiplications, which have been reduced to 1.75 million through a number of code

(SW) optimizations. The single-chip multiprocessor architecture that will be selected for the practical implementation of the algorithm will provide the scalable computation horsepower needed for the highly accurate ECG analysis that we are targeting. The autocorrelation we use, as shown in (3), has a certain number of Lags (L) to minimize the computation for our specific application as discussed below. We validated our algorithm over several medical traces [19][20].

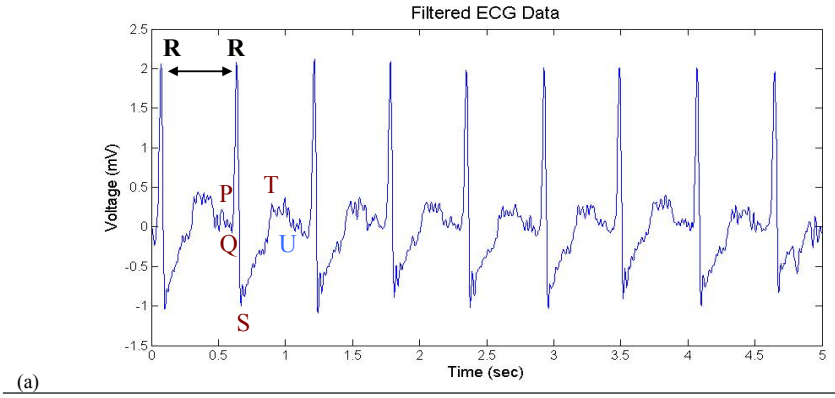
$$R_y[k] = \sum_{n=-\infty}^{n=\infty} y[n] \times y[n-k] \tag{3}$$

where *Ry* is the autocorrelation function, *y* is the filtered signal under study, *n* is the index of the signal *y*, and *k* is the number of lags of the autocorrelation (*L* has an effect on the performance due to the high number of multiplications).

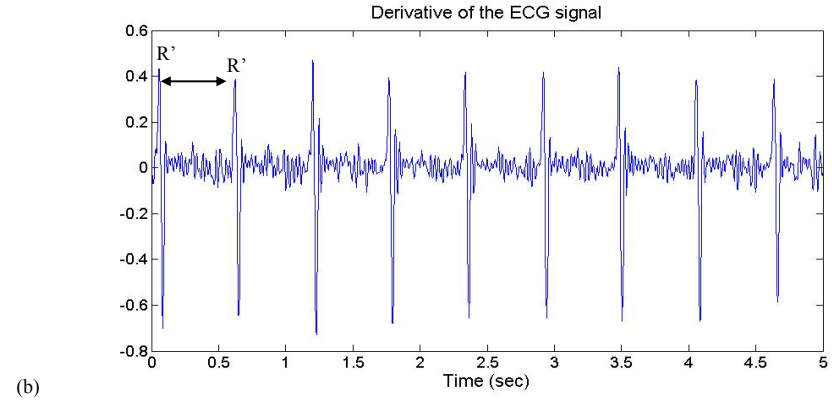
We run the experiments for n = 1250, 5000 and 50,000 relative to the sampling frequencies of 250, 1000, and 10,000Hz, respectively. In order to minimize errors and execution time we use the derivative of the ECG filtered signal since if a function is periodic then its derivative is periodic. Hence the autocorrelation function of the derivative can give the period as shown in Fig. 6. In order to be able to analyze ECG data in real-time and to be reactive in transmitting alarm signals to healthcare centers (in less than 1 minute), a minimum amount of acquired data has to be processed at a time without losing the validity of the results. For the heart beat period, we need at least 4 seconds of ECG data in order for the ACF to give correct results.

The autocorrelation function is deployed within the algorithm shown in Fig. 7, which computes the required medical parameters: heart period, peaks P, Q, R, S, T, and U, and inter-peak time spans. Peak heights and inter-peak time ranging outside normal values, which indicates different kinds of diseases, are detected with our algorithm. From a functional viewpoint, the algorithm consists of two separate execution flows: one that finds the period using the autocorrelation function (process 1 in Fig. 7), and another one that finds the number, amplitude and time interval of the peaks in the given 4-second ECG data (process 2 in Fig. 7). In process 1, we firstly find the discrete derivative of the ECG signal.

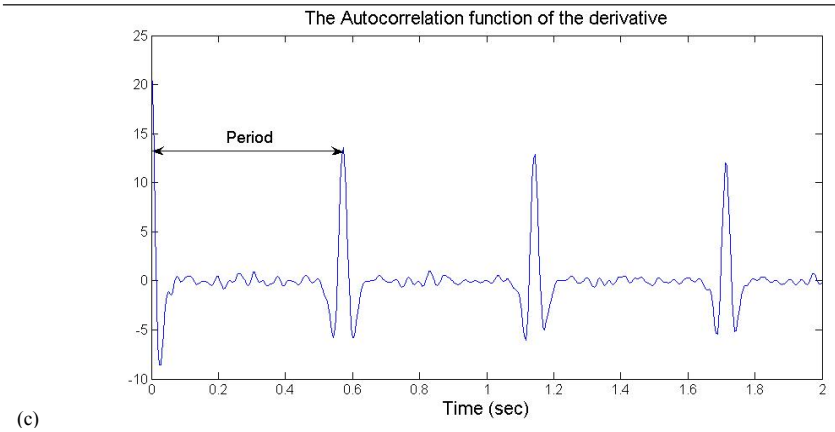
This will not affect the analysis since the derivative of a periodic signal is periodic with the same period. The advantage of taking the derivative, and thus adding some overhead to the code, is that the fluctuations taking place in the signal and especially those around the peaks would be reduced to a near-zero-value. Moreover, performance overhead associated with derivative calculation of the ECG signal is negligible compared to the rest of the algorithm, especially the autocorrelation part. Finally, if the original signal is periodic, then the autocorrelation of the derivative of the signal is periodic by definition, with the same period as that of the original signal under test. In process 2, a threshold is used to find the peaks. This threshold was experimentally set to 60% of the highest peak in the given search interval.



(a)



(b)



(c)

Fig. 6. Heart period analysis: (a) ECG signal peaks P, Q, R, S, T, and U; (b) derivative amplifying R peaks; (c) autocorrelation of the derivative characterized by significant periodic peaks having the same value as the period of the ECG signal in *(b)* and thus *(a)*

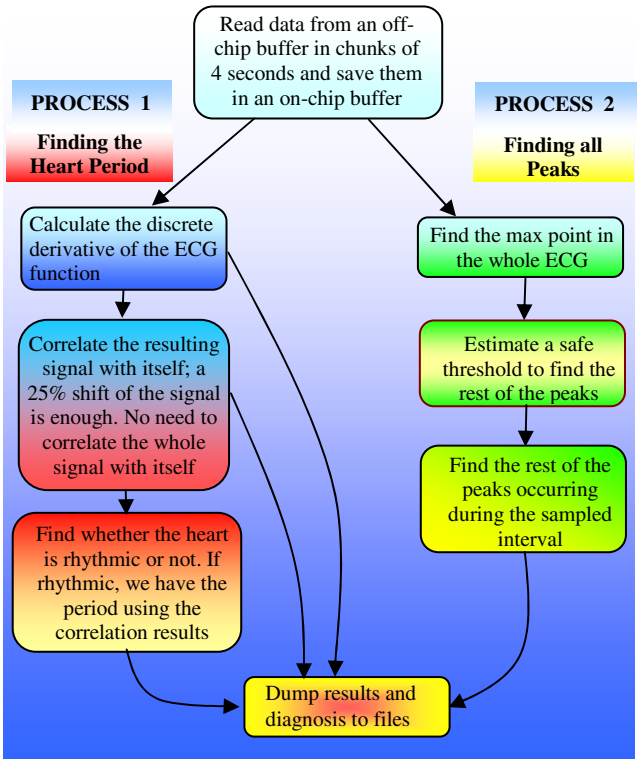


Fig. 7. The Autocorrelation function-based methodology for ECG analysis

Our proposed ECG-analysis algorithm was conceived to be parallel and hence scalable from the ground up. Since each lead senses and analyzes data independently, each lead can then be assigned to a different processor. So, to extend ECG analysis to 15-lead ECG or more, then what is required is to change the number of processing elements in the system. Alternatively, more leads can be processed by the same processor core provided the real-time requirements are achieved.

6 MPSoC Architecture

In order to process filtered ECG data in real-time, we chose to deploy a parallel Multi-Processor System-on-Chip architecture. The key point of these systems is to break up functions into parallel operations, thus speeding up execution and allowing individual cores to run at a lower frequency with respect to traditional monolithic processor cores.

Technology today allows the integration of tens of cores onto the same silicon die, and we therefore designed a parallel system with up to 13 masters and 16 slaves (see Fig. 8). Since we are targeting a platform of practical interest, we chose advanced industrial components [21]. The processing elements are multi-issue VLIW DSP cores from STMicroelectronics, featuring 32KB instruction and data caches.

Processor speed can achieve 400 MHz, although 200 MHz can be preferred in more power-aware solutions. These cores leverage the flexibility of programmable cores and the computation efficiency of DSP cores. Each processor core has its own private memory (512KB each), which is accessible through the bus, and can access an on-chip shared memory (8KB are enough for this application) for storing computation results. Other relevant slave components are a semaphore slave, implementing the test-and-set operation in hardware and used for synchronization purposes by the processors or for accessing critical sections, and an interrupt slave, which distributes interrupt signals to the processors. Interrupts to a certain processor are generated by writing to a specific location mapped to this slave core. The STBus interconnect from STMicroelectronics was instantiated as the system communication backbone. STBus can be instantiated both: as a shared bus or as a partial or full crossbar, thus allowing efficient interconnect design and providing flexible support for design space exploration. Bus frequency is 200 MHz.

In our first implementation, we target a shared bus to reduce system complexity (see Fig. 8) and assess whether application requirements can already be met or not with this configuration. We then explore also a crossbar-based system, which is sketched in Fig. 9. The inherent increased parallelism exposed by a crossbar topology allows decreasing the contention on shared communication resources, thus reducing overall execution time. In our implementation, only the instantiation of a 3x6 crossbar was interesting for the experiments. We put a private memory on each branch of the crossbar, which can be accessed by the associated processor core or by a DMA engine for off-chip to on-chip data transfers. Finally, we have a critical component for system performance which is the memory controller. It allows efficient access to the external 64MB SDRAM off-chip memory. A DMA engine is embedded in the memory controller tile, featuring multiple programming channels. The controller tile has two ports on the system interconnect: one slave port for control and one master port for data transfers. The overall controller is optimized to perform long DMA-driven data transfers. Embedding the DMA engine in the controller has the additional benefit of minimizing overall bus traffic with respect to traditional standalone solutions. Our implementation is particularly suitable for I/O intensive applications such as the one we are targeting in this work.

In the above description, we have reported the worst case system configurations. In fact, fewer cores can be easily instantiated if needed. In contrast, this architectural template is very scalable and allows for further future increase in the number of processors. This will allow to run in real time even more accurate ECG analyses for the highest sampling frequency available in sensors (10,000Hz, and 15 leads, for instance), since this platform is able to provide scalable computational power. The entire system has been simulated by means of the MPSIM simulation environment [21], which provides for cycle-accurate functional simulation of complete MPSoCs at a maximum simulation speed of about 200Kcycles/second (running on a P4 at 3.5GHz). The simulator provides also a power characterization framework leveraging 0.13μm technology-homogeneous industrial power models from STMicroelectronics [10][11]. We believe that for life-critical applications such as ECG real-time analysis, it is important to conduct low-level accurate simulations in order to perfectly understand system level behavior and have a predictable system with minimum degrees of uncertainty.

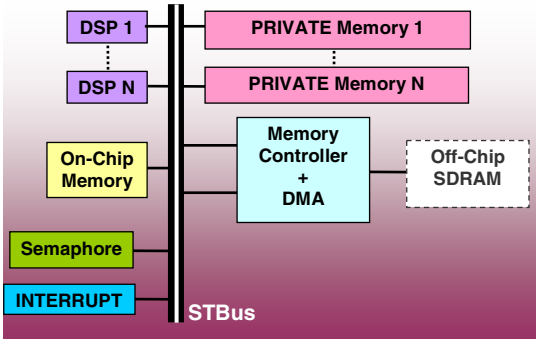


Fig. 8. Single bus architecture with STBus interconnect

Each processor core programs the DMA engine to periodically transfer input data chunks onto their private on-chip memories. Moved data typically corresponds to 4 seconds of data acquisition at the sensors: 10KB at 1000Hz sampling frequency, transferred on average in 319279 clock cycles (DMA programming plus actual data transfer) on a shared bus with 12 processors. The consumed bus bandwidth is about 6MBytes/sec, which is negligible for an STBus interconnect, whose maximum theoretical bandwidth with 1 wait state memories exceeds 400Mbyte/sec. Then each processor performs computation independently, and accesses its own private memory for cache line refills. Different solutions can be explored, such as processing more leads onto the same processor, thus impacting the final execution time. Output data, amounting to 64 bytes, are written to the on-chip shared memory, but their contribution to the consumed bus bandwidth is negligible. In principle, when the shared memory is filled beyond a certain level, its content can be swapped by the DMA engine to the off-chip SDRAM, where the history of 8 hours of computation can be stored. Data can also be remotely transmitted via a telemedicine link.

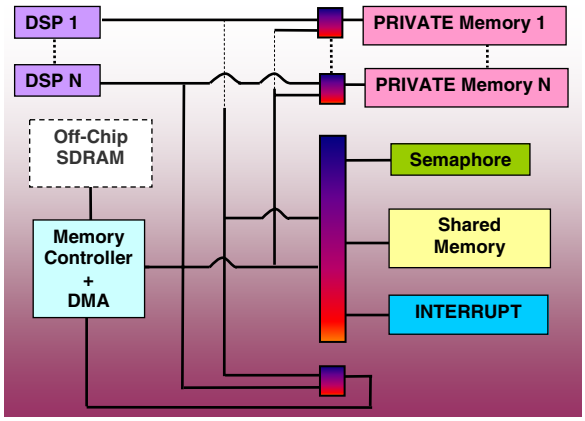


Fig. 9. Crossbar architecture with STBus interconnect. Low-bandwidth slaves have been grouped to the same crossbar branch (partial crossbar concept).

7 Experimental Results

The first analysis was done to profile the execution of the code and to determine the best coding solution in terms of energy, execution time, and precision. Furthermore, we have explored the design space searching for the best platform configuration for the 12-lead ECG data analysis. Alternative system configurations have been devised for different levels of residual battery lifetime, trading off power with accuracy.

7.1 Floating Point vs. Fixed Point Code

We ran two different code implementations: (a) one using *floating point* variables and (b) one using *fixed point integers* [22] with an exponent of 22. Fig. 10 shows the results for the two different code implementations from *time* (*execution time*) and *energy* (*relative*) points of view. The ST220 processor core runs at 200MHz. We have performed the analysis for 3, 6 and 12 leads; furthermore we process each lead on a separate core.

We found that the precision of the results obtained with fixed point code, by using 64 bit integer data types representation, almost matches the results obtained with floating point code for a large number of input data traces. On the contrary, the time needed to process data, and also the energy required, decreases up to 5 times. This is mainly due to the fact that, like many commercial DSPs, our processor cores do not have a dedicated floating point unit. Therefore, floating point computations are emulated by means of a C software library linked at compile time. Fig. 10 also shows that even with 12 concurrent processors, the bus is not saturated, since we observe negligible effects on the stretching of task execution times. In contrast, adding more processors determines a linear increase in energy dissipation.

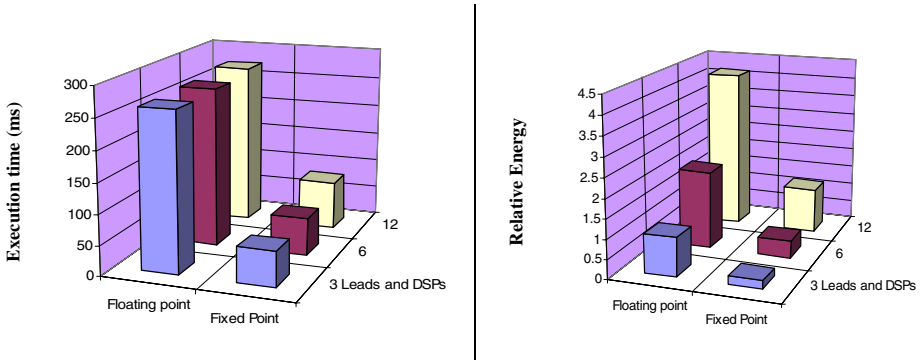


Fig. 10. Comparison between different code implementations for the analysis of the 3-lead, 6-lead and 12-lead ECG. Data analysis for each lead is computed on a separate processor core. Sampling frequency of input data was 250Hz. System operating frequency was 200 MHz.

7.2 Comparison Between Processor Cores

We then compared the performance of an ARM7TDMI with the ST220 DSP core, in order to assess the relative performance of the chosen VLIW DSP core with respect to a reference and popular architecture for general purpose computing, when put at work to process the computation kernel of our specific application. In order to have a safe comparison, we set similar dimensions of the cache memory (32KB) for the two solutions, and we run two simulations for the processing of one ECG-Lead at 250Hz sampling frequency. We count execution cycles to make up for the different clock frequencies.

We adopt this single-core solution, since our first aim is to investigate the computation efficiency of the two cores for our specific biomedical application, and de-emphasize system level interaction effects such as synchronization mismatches or contention latency for bus access. In Fig. 11, we can observe that the ST220 DSP proves more effective both in execution time and energy consumption, as expected. In detail, the ARM core is 9 times slower than the ST220 in terms of execution time, and it consumes more than twice the energy incurred by the DSP. These results can be explained based on three considerations:

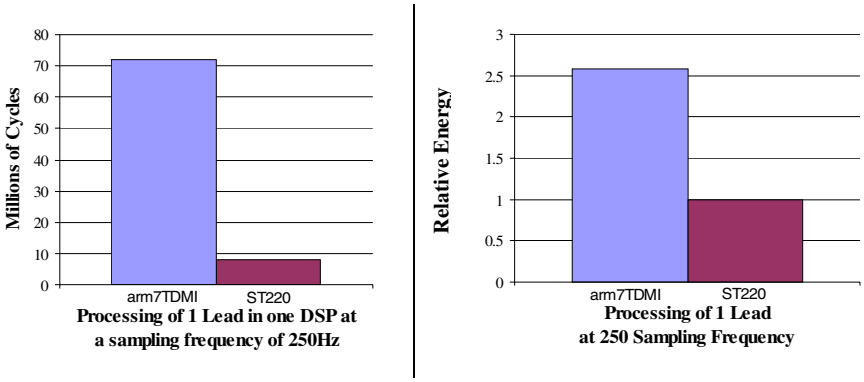


Fig. 11. Comparing ARM7TDMI with ST200 DSP performances, when processing 1 Lead at 250Hz sampling frequency

- The ST220 has better software development tools, which result in a smaller executable code. The size of the executable code for the ARM is 1.7 times larger than that of the ST220.
- The ST220 is a VLIW DSP core, therefore it is able to theoretically achieve the maximum performance of 4 instructions per cycle (i.e., 1 bundle).
- A metric which is related to both previous considerations is the static instructions per-cycle, which depends on the compiler efficiency and on the multi-pipeline execution path of the ST220. For our application, this metric turns out to be 2.9 instructions-per-bundle for ST220.

7.3 Allocation of Computation Resources

Based on previous findings (Sect. 7.1 and Sect. 7.2), we will adopt a HW/SW architecture consisting of the ST220 DSP core and a fixed point coding implementation of the algorithm for the experiments that follow. The ST220 will be operated at its typical frequency of 400MHz, while the rest of the system will run at 200 MHz. We now want to optimally configure the system to satisfy the application requirements at the minimum hardware cost. We therefore measure the execution time and the energy dissipation for an increasing number of DSP cores in order to find the optimal configuration of the system. Since commercially available ECG solutions target sampling frequencies ranging from 250 to 1000Hz, we performed the exploration for these two extreme cases for the 12-lead ECG signal. We analyze a chunk of 4secs of input data, which provides a reasonable margin for safe detection of heartbeat disorders.

Figure 12 shows that if we increase the number of processors, the execution time scales almost linearly, at least up to 6 processors. After that, we observe diminishing returns in increasing system parallelism. Since the real-time requirement of 4 seconds for the overall computation is largely met, we conclude that in the range of interest (up to 6 processors) second order effects typical of multi-processor systems (e.g., bus contention reducing the offered bandwidth to the processor cores with respect to the requested one) are negligible. A single shared bus and even a single processor core are well suited for this case.

However, this does not mean that the amount of data moved across the bus is negligible. This data is, however, read by the processor cores throughout the entire execution time, thus absorbing only a small portion of the bus bandwidth. In this regime, bus performance is still additive, i.e. the bus delivers a bandwidth which equals the sum of the bandwidth requirements of the processor cores.

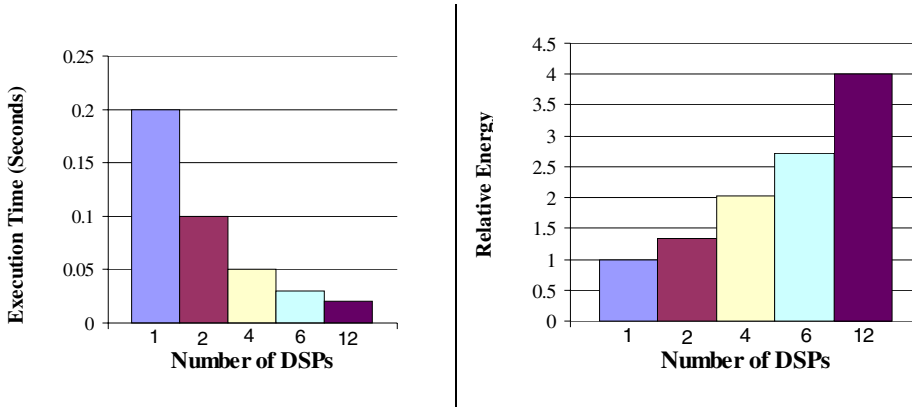


Fig. 12. Execution Time and relative energy of the system with an increasing number of DSPs and input data sampled at 250Hz sampling frequency. System interconnect is a shared bus.

Moreover, the good scalability of the application is also due to memory controller performance. In fact, at the beginning of the computation each processor loads

processing data from the off-chip to the on-chip memory, hence, requiring peak memory controller bandwidth. The architecture of the memory controller proves capable of providing the required bandwidth in an additive fashion.

By looking at the 1000Hz plot (Fig. 13), we observe that for the single processor case, the time it takes for a DSP to process 12 leads increases by more than 15 times with respect to the 250Hz case. Energy has increased as well by 90%. We still have about 1 second margin before the deadline (4 seconds), which is enough to perform additional analysis of the results of the individual lead-computations and converge to a diagnosis based on computed heartbeat parameters.

In case a larger margin is needed, the increased workload can be effectively tackled by activating a larger number of processor cores. This comes at smoother energy degradation than the 250 Hz case, as showed in Fig. 13 (for the 1kHz sampling frequency). The larger number of energy consuming cores is better amortized by the savings on application execution.

Although even for the 1kHz case, 1 DSP already meets the real-time requirements, the inherent parallelism of our architecture is useful in many senses. Firstly, when the margin to the deadline is too tight to run a complex diagnosis algorithm, the execution time can be reduced by using more processors. Secondly, working with a large number of processors allows sustaining higher sampling frequencies than 1kHz and more complex algorithms for high accuracy analysis. Thirdly, more processors can help save power, since instead of running one processor at full-speed, we may want to run more processors at reduced speeds thus cutting down on overall system energy.

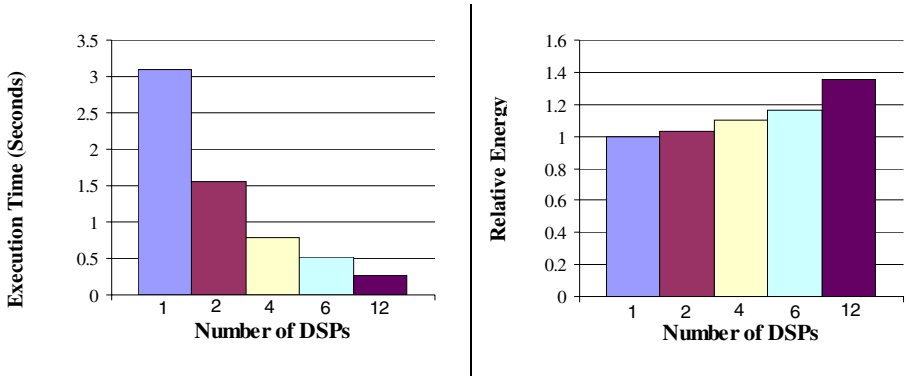


Fig. 13. Execution Time and relative energy of the system with an increasing number of DSPs and input data sampled at 1000Hz sampling frequency. System interconnect is a shared bus.

An overview of the performance and energy overhead that is incurred when moving from 250Hz to 1000kHz sampling frequencies of input data is reported in Fig. 14. Interestingly, the performance plot shows a constant 15x increase in computation time up to 4 processors. In the 6 processor case, the larger amount of data which needs to be transferred on the bus by each processor (due to data over-sampling) determines an increase of bus access times and therefore a longer execution time. As we push system parallelism to the limit, we observe (see the 12 DSPs case) that the computation workload is fully parallelized, and a huge but unique peak bandwidth is

requested to the bus. Moving from 1 DSP to 12 DSPs, we move from 12 null contention bandwidth peaks to a single, heavy contention peak. This traffic profile shapes the execution time ratio curve as showed in Fig. 14.

The energy-ratios plot confirms that the overhead for introducing more processors is worth in the 1000Hz case, while is not fully justified for the 250Hz case due to the different computation complexities to be tackled.

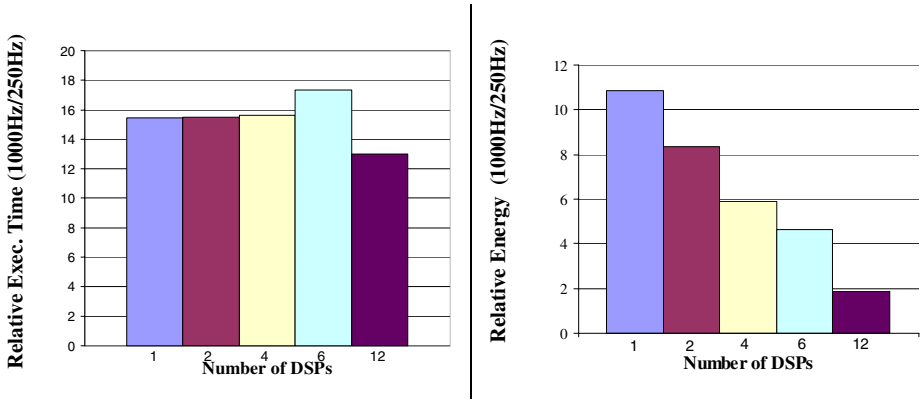


Fig. 14. Relative Execution Time and Energy Ratios between the 1000Hz and the 250Hz sampling frequency experiments

7.4 HW/SW Optimization for Aggressive Scalability

We are interested in assessing the achievable upper bound in system performance. This paves the way for further improvements of the biomedical algorithm, and it supports the use of the high data acquisition capabilities of the state-of-the-art biomedical sensors (i.e. higher sampling frequencies).

In order to push our HW/SW design to suit more accurate analysis while respecting the real-time constraint, we look at how we can push both: the specific-application algorithm (SW) and the HW architecture while considering the high medical demands of *correctness* and *accuracy* of results at the service level (medical service). To have higher accuracy and be able to diagnose arrhythmias like the R-on-T phenomena [17] and other medical cases, we found that the biomedical analyses necessitate higher sampling frequencies as input. The need for analysis at higher frequencies delivers the reality that: not only do we need to look at HW issues, but we also have to look at the algorithm parameters. In previous experiments, we used a 4-second input chunk to leave a safety margin for the input signals, and we used the number of Lags (L) variable to compensate for the data chunk size. We found that in the case of higher frequencies we can change some parameters so that the input data chunk can be optimized while still keeping good service (medical) level results. The solution is that we restrict the analysis chunk-size of our biomedical algorithm to 3.5 seconds (instead of 4 seconds), which also effects the number of multiplications that are needed. From the HW viewpoint, we simulated a 12 processor system performing the 12-lead ECG analysis with increasing sampling frequencies to determine the threshold value beyond which the system does not converge to a solution in real-time. We found that

the limit for the input sampling frequency to be 2200Hz (maximum). We verified that in this operating condition, system performance is communication-limited, i.e. the shared bus architecture is not able to keep up with the increase in communication bandwidth requirements any more. Therefore, we face the need to push the hardware as the algorithm was pushed to the maximum. By further performing hardware optimization, we were able to replace the shared bus with a full crossbar, and observed that 12 leads could be processed then in slightly more than 1 second, i.e. well below the 3.5 seconds deadline. Such an optimized HW/SW architecture was proved to work in real-time up to a sampling frequency of 4000Hz. In this condition, the system turns out to be computation-dominated, hence the communication architecture is not the bottleneck.

The flexibility of our system interconnect allows to achieve the same performance with less hardware resources. In fact, a partial crossbar design was experimented, consisting of grouping low bandwidth cores on the same crossbar branch. We observed that performance with the partial crossbar closely matches that of a full-crossbar (less than 2% average difference) but with almost 3 times less hardware resources. We found the optimal crossbar configuration (5x5 instead of 13x13) by accurate characterization of shared bus performance. On a shared bus, we increased the number of processors and observed when the execution time started deviating as an effect of bus contention. With up to 4 cores connected to the same communication resource, this latter is still able to work in an additive regime. Hence, it is not necessary to use full crossbars, but partial crossbars can be equally effective with less hardware resources.

8 Conclusion and Future Work

We present an application-specific MPSoC architecture for real-time ECG analysis, which paves the way for novel healthcare delivery scenarios (e.g., mobility) and for accurate diagnosis of heart-related diseases in real-time. Although a single DSP architecture proves capable of meeting the real-time requirements of our biomedical applications for lower than the maximum (10kHz) that state-of-the-art biomedical-sensors can deliver, the inherent parallelism we provide prevents the architecture from being the bottleneck for further advances in the field of ECG analysis. Our biochip solution can support the increasing sampling frequencies of biomedical sensors and the increased computation efficiency of analysis algorithms optimized for accuracy. We propose a case of such algorithms, leveraging auto-correlation function as a better performing alternative to the traditional and commonly-used Pan-Tompkins algorithm. An in-depth comparison of these algorithms goes beyond the scope of this paper, and is left for future work. The hardware architecture was built based on industrial components, and its performance upper bounds were clearly identified. The optimized HW/SW platform proves capable of dealing with up to 4000Hz sampling frequencies, when system performance becomes computation-limited.

Acknowledgments. We would like to thank Mr. Rabih Salibah, Mr. Fadi Bitar, Ms. Sandra Al Hattab, and Mohammad AbdelHak from the American University of Beirut (AUB), Beirut, Lebanon, for their discussions and work on the filters and validations.

References

1. Fuster V.: Epidemic of Cardiovascular Disease and Stroke: The Three Main Challenges, Circulation, Vol. 99, Issue 9, (March 1999) 1132-1137
2. Heart and Stroke Foundation of Canada: The Changing Face of Heart Disease and Stroke in Canada 2000, Annal report (1999)
3. Chan, C., Han, J., Ramjeet, D.: LabVIEWTM Design of a Vectorcardiograph and 12-Lead ECG Monitor: Final Year Project for the Bachelor of Science Degree in the University of Manitoba (March 2003)
4. Ambu, Inc. biomedical devices company: www.ambuusa.com
5. Harland, C., Clark, T., Prance, R.: Electric Potential Probes– New Directions in the remote sensing of the human body, Measurement Science and Technology, Vol. 13, (2002) 163-169
6. Harland, C., Clark, T., Prance, R.: High resolution ambulatory electrocardiographic monitoring using wrist-mounted electric potential sensors, Measurement Science and Technology, Vol. 14 (2003) 923-928
7. Malmivuo, J., Plonsey, R.: Bioelectromagnetism: Principles and Applications of Bioelectric and Biomagnetic Fields, Oxford University Press (1995)
8. Chevrollier, N., Golmie, N.: On the Use of Wireless Network Technologies in Healthcare Environments, Proceedings of the fifth IEEE workshop on Applications and Services in Wireless Networks, ASWN2005 (June 2005) 147-152
9. Khatib, I. A., Bertozzi, D., Poletti, F., Benini, L., Jantsch, A., Bechara, M., Khalifeh, H., Hajjar, M., Nabiev, R., Jonsson, S.: MPSoC ECG Biochip: A Multiprocessor System-on-Chip for Real-Time Human Heart Monitoring and Analysis, ACM SIGMICRO International Conference on Computing Frontiers (May 2006)
10. Loghi, M., Poncino, M., Benini, L.: Cycle-Accurate Power Analysis for Multiprocessor Systems-on-a-Chip,' GLSVLSI04, Great Lake Symposium on VLSI (April 2004) 401-406
11. Bona, A., Zaccaria, V., Zafalon, R.: System level power modeling and simulation of high-end industrial network-on-chip, Design and Test in Europe Conference-DATE (February 2004) 318-323
12. Lo, B., Thiemjarus, S., King, R., Yang, G.: Body Sensor Network–A Wireless Sensor Platform for Pervasive Healthcare Monitoring, Adjunct Proceedings of the 3rd International Conference on Pervasive Computing-PERVASIVE'05 (May 2005) 77-80
13. Association of Cardiac Technology in Victoria-ACTIV: http://www.activinc.org.au/
14. Code Blue- Wireless Sensor Networks for Medical Care: http://www.eecs.harvard.edu/~mdw/proj/codeblue/
15. BIOPAC Systems Inc.: http://biopac.com/
16. Company-Bosch, E., Hartmann, E.: ECG Front-End Design is Simplified with MicroConverter, Journal of Analog Dialogue, Vol. 37 (November 2003)
17. Aaron Segal: EKG tutorial, EMT-P (1997): http://www.drsegal.com/medstud/ecg/
18. Pan, J. and Tompkins, W.: A Real-Time QRS Detection Algorithm, IEEE Transactions on Biomedical Engineering, Vol. BME-32, No. 3 (March 1985)
19. PhysioBank, physiologic signal archives, for biomedical research: http://www.physionet.org/physiobank/database/ptbdb/
20. MIT-BIH arrhythmia database- Tape directory and format specification: Document BMEC TR00, Mass. Inst. Tech. Cambridge (1980)
21. Loghi, M., Angiolini, F., Bertozzi, D., Benini, L., Zafalon, R.: Analyzing On-Chip Communication in an {MPSoC} Environment, Design and Test in Europe Conference-DATE (February 2004) 752-757
22. ARM DAI 0033A Note 33: Fixed Point Arithmetic on the ARM (September 1996)