UWS Academic Portal

An FPGA implementation of the matching pursuit algorithm for a compressed sensing enabled e-health monitoring platform

Kerdjadj , Oussama; Amira, Abbes; Ghanem, Khalida ; Ramzan, Naeem; Katsigiannis, Stamos; Chouireb, Fatima

Published in:
Microprocessors and Microsystems

DOI:
[10.1016/j.micpro.2019.03.007](https://doi.org/10.1016/j.micpro.2019.03.007)

Published: 30/06/2019

Document Version
Peer reviewed version

[Link to publication on the UWS Academic Portal](#)

Citation for published version (APA):

Kerdjadj , O., Amira, A., Ghanem, K., Ramzan, N., Katsigiannis, S., & Chouireb, F. (2019). An FPGA implementation of the matching pursuit algorithm for a compressed sensing enabled e-health monitoring platform. *Microprocessors and Microsystems*, 67, 131-139. <https://doi.org/10.1016/j.micpro.2019.03.007>

General rights

Copyright and moral rights for the publications made accessible in the UWS Academic Portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

If you believe that this document breaches copyright please contact pure@uws.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

UWS Academic Portal

An FPGA implementation of the Matching Pursuit algorithm for a Compressed Sensing enabled e-Health monitoring platform

Kerdjadj , Oussama; Amira, Abbas; Ghanem, Khalida ; Ramzan, Naeem; Katsigiannis, Stamos; Chouireb, Fatima

Published in:
Microprocessors and Microsystems

Accepted/In press: 26/03/2019

Document Version
Peer reviewed version

[Link to publication on the UWS Academic Portal](#)

Citation for published version (APA):

Kerdjadj , O., Amira, A., Ghanem, K., Ramzan, N., Katsigiannis, S., & Chouireb, F. (2019). An FPGA implementation of the Matching Pursuit algorithm for a Compressed Sensing enabled e-Health monitoring platform. *Microprocessors and Microsystems*.

General rights

Copyright and moral rights for the publications made accessible in the UWS Academic Portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

This is an Open Access item distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives License (<http://creativecommons.org/licenses/by-nc-nd/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited, and is not altered, transformed, or built upon in any way.

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

An FPGA implementation of the Matching Pursuit algorithm for a Compressed Sensing enabled e-Health monitoring platform

Oussama Kerdjidi^{a,c,*}, Abbas Amira^b, Khalida Ghanem^c, Naeem Ramzan^d, Stamos Katsigiannis^d, Fatima Chouireb^a

^aUniversity of Laghouat Amar Telidji, Telecommunications, Signals and Systems Laboratory, Laghouat, Algeria

^bDepartment of Computer Science and Engineering, College of Engineering, Qatar University, Qatar

^cCenter for Development of Advanced Technologies, Algiers, Algeria

^dUniversity of the West of Scotland, Paisley, Scotland, United Kingdom

Abstract

Wireless monitoring of physiological signals is an evolving direction in personalized medicine and home-based e-Health systems. There are several constraints in designing such systems, with two of the most important being energy consumption and data compression. Compressed Sensing (CS) is an emerging data compression technique that can be used to overcome those constraints. This work presents a low-complexity CS hardware implementation on a Field-Programmable Gate Array (FPGA) for the reconstruction of compressively sensed signals using the matching pursuit (MP) algorithm, targeting health-care applications. The proposed hardware design is based on pipeline optimization of the Programmable Logic (PL) implementation performed on the Zynq FPGA, which provides a significant performance enhancement, namely an increased processing speed and a reduced computational time since it is 115x faster than the Matlab implementation and 75x faster than the Processing System (PS) implementation carried out on the same Zynq FPGA device, while achieving alternative a high-quality signal recovery with a Peak Signal to Noise Ratio (PSNR) of 23.8 dB. Comparisons against other state-of-the-art methods showed that the low complexity of the MP algorithm can be exploited for providing almost similar results to more complex algorithms using 87 to 583 less Digital Signal Processor (DSP) cores, 28 to 540 less Block RAMs and 10300 to 84700 less Look-Up Table (LUT) slices.

Keywords: Compressed sensing, Matching pursuit, Hardware implementation, Low complexity, Zybo board, ECG

1. Introduction

Advances in technology are affecting multiple aspects of the information technology field and the related industry. Performance level enhancement, computational time reduction, ease of manufacturing, and portability of the final product are currently driving the industry. Apart from advances in hardware, newer and more efficient methods and algorithms are continuously proposed as well. One such technique with the potential to benefit multiple fields and applications is Compressed Sensing (CS) [1–3]. Compressed sensing is a signal processing method for an efficient acquisition and reconstruction of a signal. It takes advantage of the fact that most signals are sparse in nature. When used under specific conditions [4], this characteristic of sparse-

ness, provides a very powerful tool for CS against standard compression methods. CS has been used in many applications such as image and video processing [5–7], where the authors used CS to recover images or video from fewer data. It has also been applied in medical imaging, e.g. to reduce Magnetic Resonance Imaging (MRI) acquisition time [8, 9], as well as in wireless communications for sparse channel estimation [10, 11]. With the emergence and proliferation of health care applications using wearable sensors that require low energy consumption, CS is capable of providing enhanced performance in the field of physiological signals transmission [12]. Recently, owing to population aging and to the increase of life expectancy in worldwide, wireless body sensors that enable pervasive and remote health care monitoring have attracted considerably scholarly attention. Works on fall detection and human activity prediction for elderly people [13–15], as well as electrocardiography (ECG) and body temperature monitoring

*Corresponding author

Email address: kerdjidioussama@gmail.com (Oussama Kerdjidi)

[16–18], have been proposed. In sum, wireless body sensors promise to allow inexpensive, continuous, and remote health monitoring.

In the present study, we propose a module for wearable sensor-based remote health care application that exploits CS techniques and programmable hardware architectures to achieve enhanced performance, Fig 1a illustrates the overview of the system. One aspect of the proposed work is to exploit compressed sensing in order to create a system capable of reducing the number of samples used for signal construction, which would thus minimize the consumption of energy during transmission. The second aspect of our study focuses on the hardware implementation in a portable embedded system and the acceleration of computations. To this end, we used the Shimmer platform [19] which can provide various signals: accelerometer, gyroscope, ECG, and electromyogram (EMG). The ECG is a physiological signal that reflects the cardiac electrical activity and contains substantial information about the human body. Furthermore, it has sparse signal characteristics, which can be represented as several significant atoms in a suitable basis. The CS theory ensures that, in this case, the signal can be recovered from a small number of random projections using linear programming algorithms under certain conditions. In addition, an implementation on System-on-Chip (SoC) [20] can be used to accelerate the real-time performance of the system by adapting the algorithm in order to exploit the pipeline, the parallelism, and co-design (software/hardware). For the target application, we opted to use the Zynq SoC [21], as, alongside the characteristics and advantages of traditional FPGA platforms, it additionally provides the flexibility and the choice between different interfaces, optimization, programmable logic (PL), and processing system (PS). The biggest advantage of the Zynq SoC is that it combines the software programmability of an ARM-based CPU with the hardware programmability of an FPGA. The signal reconstruction algorithm used when employing CS techniques has a major impact on the performance of a system. Currently, several reconstruction algorithms which are defined either in the context of convex optimization, or greedy approaches, are available [22]. Due to their low implementation cost and high recover speed, the most commonly used greedy algorithms are the Matching Pursuit [2] algorithm and its derivative Orthogonal Matching Pursuit (OMP) [23]. In this study, aiming to accelerate the computations in the proposed system by exploiting the pipeline while achieving high quality signal recovery, we present a hardware implementation of the basic Matching Pursuit (MP) algorithm. In response to the

current demand of lower complexity and lower energy consumption, the proposed approach can be effectively employed in the developed health care applications. The proposed schemes are implemented and tested out on the Xilinx Zybo board [21]. The implementation is simulated and tested using synthetic and real ECG signals collected using the Shimmer platform. The following two implementations were developed and tested: i) a software running on the Processing System (PS) part of the Zybo board and ii) an Intellectual Property (IP) core that executed in the Programmable Logic (PL) part of the board. Multiple optimizations were used in order to reduce complexity and computational time. The remainder of this paper is organized as follows. Section 2 provides background information in the field of Compressed Sensing, while the architecture and the methods used in the proposed health monitoring systems are described in section 3. Furthermore, in section 4, the performance of the proposed systems is evaluated, whereas conclusions are drawn and future research directions are outlined in section 5.

2. Background

Compressed sensing as a signal processing method to acquire and reconstruct a signal using fewer samples than the original signal was first introduced in 2006 by Donoho [1] and Candes et al. [24]. Capitalizing on the largely sparse nature of most physiological signals, it is possible to apply the CS theory to sense, compress, and recover various types of such signals. The main objective of CS is to solve the following equation (see Eq. 1):

$$\mathbf{Y} = \mathbf{\Theta}\mathbf{x} \quad (1)$$

Where $\mathbf{x}$ is the non-sparse input signal vector with length N , the M is the length signal $\mathbf{Y}$. $\mathbf{\Theta} = \mathbf{\Phi}\mathbf{\Psi}$, with $\mathbf{\Theta}$ being the $M \times N$ sensing matrix, $\mathbf{\Phi}$ is the sensing basis, and $\mathbf{\Psi}$ is the $N \times N$ sparsifying basis. $M \ll N$ observations are extracted from the N samples through the processing shown in Eq. 1 which could be rewritten as follows (see Eq. 2):

$$\mathbf{Y} = \mathbf{\Phi}\mathbf{f} \quad (2)$$

Where $\mathbf{f} = \mathbf{\Psi}\mathbf{x}$ is the projection of the original signal on the space generated by the sparsifying matrix. For a CS algorithm to be efficient, three key conditions are to be satisfied. Firstly, the signal $\mathbf{f}$ needs to be sparse in a given domain; secondly, the matrix $\mathbf{\Phi}$ should satisfy the Restricted Isometric Property (RIP) [25]; and, thirdly, the matrices $\mathbf{\Psi}$ and $\mathbf{\Phi}$ are to be incoherent [4].

To solve these equations, many algorithms have been proposed, such as convex, combinatorial, and greedy algorithms [26], with each type of technique having its respective advantages and disadvantages. The computational complexity of each of these three types of techniques is shown in Table 1, where c is a positive constant and K the sparsity of the signal.

Numerous studies have proposed the use of CS techniques in the field of health care monitoring. For instance, there has been a proposal to use CS methods in remote monitoring of patients through ECG and EEG [31]; in that study, the authors exploited a learning dictionary to increase the compression ratio. Furthermore, Kerdjidj et al. [32] investigated the effect of the sparsifying dictionaries on raw ECG signals for high-quality recovery and reduction of dictionary size. In addition, Vega et al. [33] also proposed a system for monitoring heart activity through ECG signals, transmitted via Bluetooth. In [34], Darren et al. investigated the performance of CS among other standard compression algorithms for applications using bioelectric signals, while Zhang et al. [35] examined the reduction of power consumption in monitoring systems using CS methods on an arrhythmia database for several algorithms. Finally, Craven et al. [36] exposed the impact of CS in the monitoring of clinical and ambulatory ECG for QRS beat-detection.

While practical applications of CS in hardware platforms remain rather limited, various hardware implementations of CS techniques have recently been proposed. For example, the Orthogonal Matching Pursuit (OMP)-based greedy algorithms [37–39] are frequently preferred in hardware implementations [40–42] for real-time signal recovery. One problem of iterative algorithms like the OMP is the latency in execution. Consequently, many studies have focused on accelerating the computation of the algorithm by optimizing implementations for CPUs, GPUs, and FPGAs [43].

One of such studies, by Bai et al. [40], presented a Field-Programmable Gate Array (FPGA) implementation of the OMP greedy algorithm, while Ren et al. [41] proposed a single-precision FPGA implementation for high-speed processing. Furthermore, as showcased in [42] exploiting the parallelism of the algorithm and reducing the area overhead, an FPGA implementation of OMP reduces the execution time by reusing the hardware for matrix-vector multiplications.

Rabah et al. [42] proposed a model designed using Simulink for an FPGA implementation of the OMP algorithm. The execution time was shown to be reduced by exploiting the parallelism on some complex functions of the algorithm, by the reduction of area

overhead, and by the reuse of the hardware matrix-vector multiplication in other components of the algorithm. The evaluation of the quality of recovery is assured by the PSNR. Kulkarni et al. [44] proposed another solution to reduce the execution time and hardware complexity of the OMP algorithm through its implementation on an Application Specific Integrated Circuit (ASIC). To reduce hardware costs in terms of operation latency and area requirements, in that work a thresholding technique was applied to the OMP algorithm such that to decrease the number of cycle counts in the selection phase. Moreover, a weight vector based on the gradient descent method was investigated with the aim of optimizing the least square kernel in the updating part. Results showed that their proposed modifications induce up to 2.1 times less area-delay product and achieve 40% less energy consumption with a recovery quality guaranteed by the two metrics PSNR and Structural Similarity Index Measures (SSIM).

Finally, in [45], the authors proposed an FPGA architecture based on Matrix Inverse Bypass (MIB) where hardware resources are dynamically allocated for high-speed computation, especially the dominant matrix-vector multiplications with varying sizes at the different stages of each iteration and at different iterations in signal recovery. While CS reduces the hardware requirements of signal acquisition, the recovery of the original signal still remains an arduous task. Despite recent advances in the development of fast recovery algorithms, computational complexity remains high. Due to this high computational complexity, signal reconstruction is still a challenging task, especially in the case of autonomous real-time signal recovery. To address this problem, in the present study, we propose and evaluate an FPGA implementation of the matching pursuit (MP) algorithm for compressed sensing reconstruction capable of reconstructing signals for the proposed health care application. To the best of our knowledge, few previous studies have attempted to address the problem of hardware implementation of the MP algorithm on hardware, e.g. [46, 47], targeting real time video reconstruction. Among these few studies is the research by Acciarito et al. [48] who reported improving the performance of the OMP algorithm by using a mixed hardware and software implementation on the Xilinx Zynq FPGA. Another relevant study is Pinto et al. [49] where the OMP technique was implemented on FPGA and microcontroller for a multi-signal (ECG and EMG) system exploiting Fourier and wavelet analysis. Finally, Liu et al. [50] evaluated the quality, speed, and power energy of a consumption of hardware FPGA implementation of the Block Sparse Bayesian learning (BSBL) CS algo-

rithm for ECG and electroencephalography (EEG) signals.

3. Proposed system and implementation design

3.1. Overview of the MP algorithm and its complexity

In the present study, we opted to use the less complex CS reconstruction algorithm in order to employ less hardware resources and to increase performance. As shown in Table 1, greedy algorithms satisfy this requirement. A greedy algorithm attempts to find feasible solutions by solving the problem step by step. The Matching Pursuit (MP) and its derivative Orthogonal Matching Pursuit (OMP) algorithms [2, 23] are the most widely used greedy CS algorithms. The MP algorithm was selected for this work, since it has lower complexity than the other greedy algorithms and its proposed software and hardware implementation is based on Algorithm 1, which is used to solve Eq. 1. The MP algorithm encompasses two phases: selection and update. Phase 1 (selection) is used to find the best-correlated atom from the dictionary with the residual current error. Phase 2 (update) ensures the update by subtracting the previously calculated correlated part from the residue. The algorithm terminates when some termination criteria have been satisfied or the maximum number of iterations has been exceeded. The MP algorithm is outlined below in Algorithm 1.

Algorithm 1 Matching Pursuit

Input: Φ : Sensing matrix ; Y : Observation vector; K : Sparsity of signal X .

Output: $\hat{X}$: Original signal estimation

Procedure:

- 1: Initialize residual $r_0 = Y$ and set iteration counter $i = 1$.
- 2: Find index $\lambda_i = \arg\max |\langle r_{i-1}, \phi_j \rangle|$ for $j = 1 \dots N$.
- 3: Calculate the new approximation $\hat{X}_i = \hat{X}_{i-1} + \langle \phi_{\lambda_i}, r_{i-1} \rangle \phi_{\lambda_i}$ and the new residual $r_i = Y - \hat{X}_i = r_{i-1} - \langle \phi_{\lambda_i}, r_{i-1} \rangle \phi_{\lambda_i}$
- 4: Increment iteration counter $i = i + 1$, return to step 2 if $i < K$
- 5: Output estimation $\hat{X}$

A MATLAB reference implementation [51] of the OMP algorithm was used in order to evaluate its recovery quality, against that of the MP algorithm, in terms of PSNR in relation to the number of iterations used. As is evident from the result for an input signal with the size of 128 samples (see Fig. 2), the MP algorithm requires more iterations to achieve recovery quality similar to that of the OMP algorithm. For example, for 40 iterations, the OMP algorithm achieved the PSNR of ca. 43 dB, whereas, for the same number of iterations, the MP algorithm achieved the PSNR of 39 dB. Overall, 25% more iterations (i.e. 50 iterations)

were needed in order for the MP algorithm to achieve a similar PSNR value than that of the OMP algorithm. Previous studies also support the finding that OMP outperforms MP in terms of recovery quality. Nevertheless, due to its lower computational complexity, an optimized hardware implementation of the MP algorithm can offer lower computational time and similar quality as compared to OMP.

3.2. Proposed system and implementation architecture

The proposed system is a wearable sensor-based remote health care application that targets monitoring elderly peoples movements. An overview of the system is provided on Fig. 1a. The system will be deployed in different environments such as houses, hospitals, or ambulances. A ShimmerTM [52] wireless wearable sensor is used for signal acquisition, collecting ECG, accelerometer, and gyroscope data. The collected data are sent via Bluetooth to a Zynq SoC for further processing. The proposed system architecture aims to transmit the acquired signals in a compressed form that would lead to less energy consumption and high recovery rates. The global aim of the system is to identify different daily activities, as well as to detect falls, by exploiting the accelerometer and gyroscope signals. Furthermore, the proposed system can be used to monitor the ECG signal while the user is in different places (home, hospital, etc.), engaged in various daily activities. To this end, since power consumption is of utmost importance, data compression, transmission, and recovery have to be achieved using low complexity algorithms that do not require considerable hardware resources. CS techniques were selected for compressing the signals in order to exploit their sparse nature and the low computational complexity of the compression approach. The following three implementations of the proposed CS approach were developed and tested: (1) a software one on MATLAB, used as a reference implementation; (2) a software one running on the Processing System (PS) part of the Zybo board, and (3) an FPGA one (IP core) that executed in the Programmable Logic (PL) part of the board. Figure 1b illustrates the hardware architecture of the processing unit of the proposed module for the system.

3.2.1. Implementation on PS

The implementation of the MP algorithm consists of the compression and the recovery modules a data-flow graph is illustrated to show the different functions of MP algorithm (see Fig. 3a 3b and 3c). The compression process starts by reading the signal S_i and the dictionary $D_{i,j}$ (matrix) from the Zybo boards storage unit

(SD card). For each iteration K , the correlation between the columns of matrix D and the signal S is computed in order to find the index ind of the most correlated column. Then, the new coefficient $coef$ is computed by multiplying the signal S with the column D_{ind} . The resulting $coef$ and its corresponding dictionary index ind are then stored. In the next step, a temporary vector tmp is computed by multiplying $coef$ with D_{ind} and, subsequently, the computed tmp is subtracted from the initial signal S_i in order to compute the input signal S_{i+1} for the next iteration of the algorithm. The coefficients $coef$ and the index ind , computed during the compression procedure, are used for the recovery of the signal. The dictionary's column D_{ind} is loaded and multiplied by $coef$. The result is accumulated in order to recover the signal. An example of signal recovery using the MP algorithm on the PS part of the Zybo board compared to running the algorithm on MATLAB is shown in Fig. 4, exhibiting minimal difference in the recovered signals between the two implementations, which can be attributed to accuracy differences between MATLAB and Zybo.

3.2.2. Implementation on PL

In order to achieve lower computational time, a hardware implementation of the MP algorithm was also developed, designed, and implemented on the PL part of the Zynq platform. Fig. 5 illustrates the proposed hardware block in the Vivado Design Suite.

The proposed IP is built using Vivado HLS [53, 54] to create, synthesize the function, and translate the code on the register-transfer level (RTL) logic, where this logic can provide an IP using the Verilog / VHDL languages. To control and transfer data between the microprocessor and the IP, an AXI-lite Slave peripheral interface is used between the two principal blocks. In Vivado HLS, the *Pipeline* (P) and the *Unroll* (U) directives for code optimization are used to accelerate the algorithms computation and to reduce computational time (see Table 2). Algorithms 2 and 3 provide a pseudo code for the proposed IP implementation of the MP algorithm for the compression and the recovery modules, respectively. Details on the use of directives for code optimization are also included in the pseudo code.

In addition to the IPs, some necessary IP catalogs have to be included in the implementation, such as the Zynq7 Processing System that is used to run the software in Zynq with a direct physical connection to the *DDR memory*. The memory mapped register can be accessed by processors via the *AXI4 Interconnect*, which controls and monitors a set of blocks of *custom logic* in the PL. The *Concat IP* is employed to concatenate

Algorithm 2 Compression

- 1: Compute the correlation ($Corr$)
 $Corr = A \times B$ (**Unroll** or **Pipeline**)
 - 2: Find the index (Ind) of the best correlation
 $Index = \max(Corr)$
 - 3: Compute the Coefficients ($Coef$)
 $Coef = A_{Ind} \times B$ (**Unroll** or **Pipeline**)
 - 4: Update the signal B
 $B_{new} = B_{old} - [A_{index} \times Coef]$ (**Unroll** or **Pipeline**)
-

Algorithm 3 Recovery

- 1: Downland coefficients ($Coef$) and index (Ind)
 $Ind_1, \dots, iter = index_1, index_2, \dots, index_{iter}$
 $Coef_1, \dots, iter = coef_1, coef_2, \dots, coef_{iter}$
 - 2: Extract the data from dictionary A
 $NewMat_{1:Ind} = A_1, A_2, \dots, A_{Ind}$
 - 3: Reconstruct the signal B_{rec}
 $B_{Rec} = NewMat_{1:Ind} \times Coef_1, \dots, iter$
-

the interrupt signals provided from the different blocks. Finally, the *Processing System Reset* provides a mechanism to handle the reset conditions for the system. Apart from the IP core, the necessary driver and control code were implemented using the Software Development Kit of Xilinx (SDK) and ran on the ARM cortex CPU of the Zynq7 board.

4. Experimental setup and results

In order to validate the Processing System (PS) and the Programmable Logical (PL) implementations, the MATLAB implementation of the MP algorithm was used as a reference, and the acquired results were compared. To this end, three different datasets were created consisting of ECG, accelerometer, and gyroscope signals collected in the lab from 17 volunteers, using the Shimmer [19] wireless wearable sensor for signal acquisition.

4.1. Vivado HLS simulation

The dictionary matrix D was generated using MATLAB and was stored along the signal S (input data) in the storage unit (SD card) of the board. Thereafter, the Xilinx Vivado HLS tool was used in order to create and build the hardware architecture, coded in C, and tested on the C level simulation to evaluate the performance and the results of the implementation against the MATLAB reference implementation. After verification by the RTL simulation, the C code was translated into HDL

by the C/C++ synthesis. The clock period was set to 10 ns, 15 ns and 30 ns. Table 3 provides detail on the hardware resources and computational time.

Where, LUT: Look-Up Table, FF: Flip-Flops, DSP: Digital Signal Processor, BRAM: Block Random Access Memory.

4.2. Implementations performance

After validating and generating the IP cores using Vivado HLS, the proposed architecture was implemented on the PL of the Xilinx Zybo Board, while the necessary driver and control code were deployed on the ARM cortex CPU of the Zynq7. The performance of the proposed PL implementation in terms of computational time for 5 iterations of the compression algorithm is shown in Table 5, alongside the size of the input data and the directives used for optimization.

In order to further decrease computational time and increase performance, the proposed architecture was improved by adapting the previous IP core. The main modification consisted of executing the main loop (iterations) of Algorithm 2 outside the IP. To this end, this loop will be executed on the PS part, while the rest of the algorithm will be executed as an IP core on the PS part. The results of the experiments showed that the inclusion of the main loop in the IP core resulted in the consumption of more resources and led to higher computational time. Table 4 shows the performance of the improved approach against the PS and MATLAB implementations for the dictionary of size 100×100 , the original signal of 100 samples, the clock period of 30 ns, and various iteration numbers. Furthermore, signal recovery quality was evaluated in terms of the PSNR (see Table 4).

The computations for the PS implementation were performed on the Zybo board (650 MHz dual-core CortexTM-A9 processor with Xilinx 7-series FPGA), while the MATLAB implementation was tested on a computer equipped with an Intel Core i5-520M CPU at 2.67 GHz and 4 GB of DDR3 memory. The results of the comparison between the original and the recovered signal acquired using the MATLAB and the PL implementations for various numbers of iterations are shown in Fig. 6.

4.3. Comparison to other implementations

To the best of our knowledge, the issue of the hardware implementation of the MP algorithm has not been extensively studied yet. The OMP algorithm is the preferred choice in hardware implementations. Consequently, in the present study, we opted to compare the

proposed FPGA design of the MP algorithm with the existing FPGA implementations of the OMP algorithm [40–42, 45]. Since the MP algorithm is less complex than OMP (see Table 1), the comparisons focused on the scenarios where similar recovery quality was achieved by both algorithms.

A comparison between the proposed FPGA implementation of the MP algorithm against available OMP FPGA implementations is provided in Table 6. In this comparison, the following attributes were considered: board type, precision, maximum frequency, signal size, recovery time, area resource for ASIC and material resources (DSP cores, LUT slices). The proposed implementation uses a low cost Zynq Soc (Zybo board) with the maximum frequency of 33 MHz, which is one of the lowest among the examined works. Nevertheless, the proposed implementation achieved the lowest computational time: 150μs for compression and 25μs for recovery, for 40 iterations of the algorithm, due to some serial computation in the architecture proposed. While the proposed design exploited the pipeline and unrolls optimizations, and used 32 bits precision which could have affected material resources requirements (LUT, DSP core, and Block RAMs), it still achieved the lowest material resources requirements among the examined works. When compared with other hardware solutions, such as ASIC, the recovery time of our proposed implementation is shown to be lower mainly due to the very high frequency of the used system. This could have a critical impingement on real-time applications, such as e-health field, in which the signal reconstruction step should not penalize further machine learning-based analysis and subsequent decision making. In summary, our solution achieves less algorithm complexity compared to the literature; thanks to use of enroll optimization and pipeline strategy. Further, the signal length is considerably reduced resulting into gain in memory and computation resources. To reduce the energy consumption, we employed a reasonably low frequency as done by [38, 41]. Contrary to the other approaches, we utilize more bits to keep good recovery precision.

4.4. Power consumption

Apart from exposing the number of LUT, flip-flops, DSP, and BRAM (Table 3), the Xilinx Vivado tool also offered the possibility to depict the energy consumption and thermal properties of the design after synthesizing it on the board. Two types of power consumption are reported by the Vivado tool; static and dynamic. Dynamic power consisted of the power consumption of the clock, Bram power, signal power, logic power, the DSP, and

and the Zynq processing system (PS7). Static power referred to the energy consumption due to transistor leakage. The energy consumption of the proposed IP core is summarized in Table 7. It is evident that the highest power consumption is caused by the Zynq processing system (PS7) due to the execution of the required driver and control programs on the ARM dual core Cortex CPU.

Conclusion

In this paper, we proposed an FPGA hardware implementation of the Matching Pursuit (MP) compressed sensing algorithm for use in an e-Health platform. We have created, modeled and simulated the architecture of the MP algorithm using the Vivado tools, and a MATLAB implementation was used for reference. The MP algorithm was implemented both in the Processing system (PS) and the Programmer Logical (PL) part of a low cost Xilinx Zynq-7000 XC7Z010T-1CLG4000 board, and the design of the PL implementation was further optimized to reduce computational times. All these implementations aimed at low computational times, low energy consumption, and high-quality signal recovery, and were compared to the reference MATLAB implementation for validation. Similar recovery quality to the more complex Orthogonal Matching Pursuit (OMP) algorithm was achieved by our proposed implementation of the MP algorithm by increasing the number of iterations, while still achieving lower computational times compared to other proposed FPGA implementations of the OMP algorithm. Future work will focus on optimizing the proposed hardware design.

References

- [1] D. L. Donoho, Compressed sensing, *IEEE Trans. Inf. Theor.* 52 (4) (2006) 1289–1306. doi:10.1109/TIT.2006.871582.
- [2] S. Mallat, Z. Zhang, Matching pursuits with time-frequency dictionaries, *Signal Processing, IEEE Trans* 41 (12) (1993) 3397–3415. doi:10.1109/78.258082.
- [3] H. Djelouat, A. Amira, F. Bensaali, Compressive sensing-based iot applications: A review, *Journal of Sensor and Actuator Networks* 7 (4). doi:10.3390/jsan7040045.
- [4] E. J. Cands, The restricted isometry property and its implications for compressed sensing, *Comptes Rendus Mathematique* 346 (9) (2008) 589 – 592. doi:https://doi.org/10.1016/j.crma.2008.03.014.
- [5] Z. J. Xu, J. J. Zhang, Y. Zhang, Rgb image processing based on compressed sensing, in: *IET International Conference on Information Science and Control Engineering 2012 (ICISCE 2012)*, 2012, pp. 1–4. doi:10.1049/cp.2012.2300.
- [6] V. J. Barranca, G. Kovai, D. Zhou, D. Cai, Efficient image processing via compressive sensing of integrate-and-fire neuronal network dynamics, *Neurocomputing* 171 (Supplement C) (2016) 1313 – 1322. doi:https://doi.org/10.1016/j.neucom.2015.07.067.
- [7] R. G. Baraniuk, T. Goldstein, A. C. Sankaranarayanan, C. Studer, A. Veeraraghavan, M. B. Wakin, Compressive video sensing: Algorithms, architectures, and applications, *IEEE Signal Processing Magazine* 34 (1) (2017) 52–66. doi:10.1109/MSP.2016.2602099.
- [8] M. Sandilya, S. Nirmala, Compressed sensing trends in magnetic resonance imaging, *Engineering Science and Technology, an International Journal* 20 (4) (2017) 1342 – 1352. doi:https://doi.org/10.1016/j.jestch.2017.07.001.
- [9] Z. Lai, X. Qu, Y. Liu, D. Guo, J. Ye, Z. Zhan, Z. Chen, Image reconstruction of compressed sensing mri using graph-based redundant wavelet transform, *Medical Image Analysis* 27 (Supplement C) (2016) 93 – 104, discrete Graphical Models in Biomedical Image Analysis. doi:https://doi.org/10.1016/j.media.2015.05.012.
- [10] W. U. Bajwa, J. Haupt, A. M. Sayeed, R. Nowak, Compressed channel sensing: A new approach to estimating sparse multipath channels, *Proceedings of the IEEE* 98 (6) (2010) 1058–1076. doi:10.1109/JPROC.2010.2042415.
- [11] C. R. Berger, Z. Wang, J. Huang, S. Zhou, Application of compressive sensing to sparse channel estimation, *IEEE Communications Magazine* 48 (11) (2010) 164–174. doi:10.1109/MCOM.2010.5621984.
- [12] B. Liu, Z. Zhang, Quantized compressive sensing for low-power data compression and wireless telemonitoring, *IEEE Sensors Journal* 16 (23) (2016) 8206–8213. doi:10.1109/JSEN.2016.2550602.
- [13] R. M. Gibson, A. Amira, N. Ramzan, P. C. de-la Higuera, Z. Pervez, Matching pursuit-based compressive sensing in a wearable biomedical accelerometer fall diagnosis device, *Biomedical Signal Processing and Control* 33 (Supplement C) (2017) 96 – 108. doi:https://doi.org/10.1016/j.bspc.2016.10.016.
- [14] O. Kerdjidi, N. Ramzan, K. Ghanem, A. Amira, F. Chouireb, Fall detection and human activity classification using wearable sensors and compressed sensing, *Journal of Ambient Intelligence and Humanized Computing* doi:10.1007/s12652-019-01214-4.
- [15] T. N. Gia, V. K. Sarker, I. Tcareno, A. M. Rahmani, T. Westerland, P. Liljeberg, H. Tenhunen, Energy efficient wearable sensor node for iot-based fall detection systems, *Microprocessors and Microsystems* 56 (2018) 34 – 46. doi:https://doi.org/10.1016/j.micpro.2017.10.014.
- [16] D. Kofja, R. Stojanovi, A. KoloÅvari, A. Åkraba, Designing a low-cost real-time group heart rate monitoring system, *Microprocessors and Microsystems* 63 (2018) 75 – 84. doi:https://doi.org/10.1016/j.micpro.2018.08.010.
- [17] A. Ghosh, S. Aich, A. K. Ghosh, P. Das, S. Mahato, A. Dey, Patient health monitoring system, in: *2016 International Conference on Intelligent Control Power and Instrumentation (ICICPI)*, 2016, pp. 114–117. doi:10.1109/ICICPI.2016.7859685.
- [18] H. Djelouat, M. Al Disi, A. Amira, F. Bensaali, X. Zhai, Compressive sensing based ecg biometric system, in: K. Arai, S. Kapoor, R. Bhatia (Eds.), *Intelligent Systems and Applications*, Springer International Publishing, Cham, 2019, pp. 126–137.
- [19] A. Burns, E. Doheny, B. Greene, T. Foran, D. Leahy, K. O'Donovan, M. McGrath, An extensible platform for physiological signal capture, in: *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, 2010, pp. 3759–3762. doi:10.1109/IEMBS.2010.5627535.
- [20] L. E. M. Brackenbury, L. A. Plana, J. Pepper, System-on-chip design and implementation, *IEEE Transactions on Education*

- 53 (2) (2010) 272–281. doi:10.1109/TE.2009.2014858.
- [21] L. H. Crockett, R. A. Elliot, M. A. Enderwitz, R. W. Stewart, The Zynq Book: Embedded Processing with the Arm Cortex-A9 on the Xilinx Zynq-7000 All Programmable Soc, Strathclyde Academic Media, UK, 2014.
- [22] S. Boyd, L. Vandenberghe, *Convex Optimization*, Cambridge University Press, New York, NY, USA, 2004.
- [23] J. A. Tropp, A. C. Gilbert, Signal recovery from random measurements via orthogonal matching pursuit, *IEEE TRANS. INFORMATION THEORY* 53 (2007) 4655–4666.
- [24] E. Candes, J. Romberg, T. Tao, Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information, *Information Theory, IEEE Trans* 52 (2) (2006) 489–509. doi:10.1109/TIT.2005.862083.
- [25] A. M. Tillmann, M. E. Pfetsch, The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing, *IEEE Transactions on Information Theory* 60 (2) (2014) 1248–1259. doi:10.1109/TIT.2013.2290112.
- [26] G. Pfander, *Sampling Theory, a Renaissance: Compressive Sensing and Other Developments*, Applied and Numerical Harmonic Analysis, Springer International Publishing, 2015. URL <https://books.google.dz/books?id=pREpCwAAQBAJ>
- [27] H. Huang, S. Misra, W. Tang, H. Barani, H. Al-Azzawi, Applications of compressed sensing in communications networks, *CoRR abs/1305.3002*.
- [28] D. Needell, R. Vershynin, Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit, *IEEE Journal of Selected Topics in Signal Processing* 4 (2) (2010) 310–316. doi:10.1109/JSTSP.2010.2042412.
- [29] M. A. T. Figueiredo, R. D. Nowak, S. J. Wright, Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems, *IEEE Journal of Selected Topics in Signal Processing* 1 (4) (2007) 586–597. doi:10.1109/JSTSP.2007.910281.
- [30] M. A. Iwen, Combinatorial sublinear-time fourier algorithms, *Found. Comput. Math.* 10 (3) (2010) 303–338. doi:10.1007/s10208-009-9057-1.
- [31] D. Craven, B. McGinley, L. Kilmartin, M. Glavin, E. Jones, Adaptive dictionary reconstruction for compressed sensing of ecg signals, *IEEE Journal of Biomedical and Health Informatics* 21 (3) (2017) 645–654. doi:10.1109/JBHI.2016.2531182.
- [32] O. Kerdjadj, K. Ghanem, A. Amira, F. Harizi, F. Chouireb, Concatenation of dictionaries for recovery of ecg signals using compressed sensing techniques, in: 2014 26th International Conference on Microelectronics (ICM), 2014, pp. 112–115. doi:10.1109/ICM.2014.7071819.
- [33] V. P. Rachim, W. Y. Chung, Wearable noncontact armband for mobile ecg monitoring system, *IEEE Transactions on Biomedical Circuits and Systems* 10 (6) (2016) 1112–1118. doi:10.1109/TBCAS.2016.2519523.
- [34] D. Craven, B. McGinley, L. Kilmartin, M. Glavin, E. Jones, Compressed sensing for bioelectric signals: A review, *IEEE Journal of Biomedical and Health Informatics* 19 (2) (2015) 529–540. doi:10.1109/JBHI.2014.2327194.
- [35] J. Zhang, Z. Gu, Z. L. Yu, Y. Li, Energy-efficient ecg compression on wireless biosensors via minimal coherence sensing and weighted ℓ_1 minimization reconstruction, *IEEE Journal of Biomedical and Health Informatics* 19 (2) (2015) 520–528. doi:10.1109/JBHI.2014.2312374.
- [36] D. Craven, B. McGinley, L. Kilmartin, M. Glavin, E. Jones, Energy-efficient compressed sensing for ambulatory ecg monitoring, *Computers in Biology and Medicine* 71 (Supplement C) (2016) 1 – 13. doi:https://doi.org/10.1016/j.combiomed.2016.01.013.
- [37] J. L. V. M. Stanislaus, T. Mohsenin, High performance compressive sensing reconstruction hardware with qrd process, in: 2012 IEEE International Symposium on Circuits and Systems, 2012, pp. 29–32. doi:10.1109/ISCAS.2012.6271921.
- [38] Y. C. Cheng, P. Y. Tsai, M. H. Huang, Matrix-inversion-free compressed sensing with variable orthogonal multi-matching pursuit based on prior information for ecg signals, *IEEE Transactions on Biomedical Circuits and Systems* 10 (4) (2016) 864–873. doi:10.1109/TBCAS.2016.2539244.
- [39] A. Septimus, R. Steinberg, Compressive sampling hardware reconstruction, in: Proceedings of 2010 IEEE International Symposium on Circuits and Systems, 2010, pp. 3316–3319. doi:10.1109/ISCAS.2010.5537976.
- [40] L. Bai, P. Maechler, M. Muehlberghuber, H. Kaeslin, High-speed compressed sensing reconstruction on fpga using omp and amp, in: 2012 19th IEEE International Conference on Electronics, Circuits, and Systems (ICECS 2012), 2012, pp. 53–56. doi:10.1109/ICECS.2012.6463559.
- [41] F. Ren, R. Dorrace, W. Xu, D. Markovi, A single-precision compressive sensing signal reconstruction engine on fpgas, in: 2013 23rd International Conference on Field programmable Logic and Applications, 2013, pp. 1–4. doi:10.1109/FPL.2013.6645574.
- [42] H. Rabah, A. Amira, B. K. Mohanty, S. Almaadeed, P. K. Meher, Fpga implementation of orthogonal matching pursuit for compressive sensing reconstruction, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 23 (10) (2015) 2209–2220. doi:10.1109/TVLSI.2014.2358716.
- [43] A. Kulkarni, T. Mohsenin, Accelerating compressive sensing reconstruction omp algorithm with cpu, gpu, fpga and domain specific many-core, in: 2015 IEEE International Symposium on Circuits and Systems (ISCAS), 2015, pp. 970–973. doi:10.1109/ISCAS.2015.7168797.
- [44] A. Kulkarni, T. Mohsenin, Low overhead architectures for omp compressive sensing reconstruction algorithm, *IEEE Transactions on Circuits and Systems I: Regular Papers* 64 (6) (2017) 1468–1480. doi:10.1109/TCSI.2017.2648854.
- [45] G. Huang, L. Wang, An fpga-based architecture for high-speed compressed signal reconstruction, *ACM Trans. Embed. Comput. Syst.* 16 (3) (2017) 89:1–89:23. doi:10.1145/3056481.
- [46] A. Jarrah, M. M. Jamali, Reconfigurable fpga/gpu-based architecture of block compressive sampling matching pursuit algorithm, *Journal of Circuits, Systems and Computers* 24 (04) (2015) 1550055. arXiv: <http://www.worldscientific.com/doi/pdf/10.1142/S0218126615500553>, doi:10.1142/S0218126615500553.
- [47] A. P. Brown, Y. Meng, R. A. Iltis, Matching pursuit channel estimation: Efficient algorithm design and reconfigurable hardware implementation, 2007.
- [48] S. Acciarito, G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, G. M. Khanal, M. Re, *Compressive Sensing Reconstruction for Complex System: A Hardware/Software Approach*, Springer International Publishing, Cham, 2018, pp. 192–200. doi:10.1007/978-3-319-55071-8_25.
- [49] S. S. Pinto, L. E. Mendoza, H. J. Velandia, V. Molina, L. J. Cervelon, *Compressive Sensing Hardware in 1-D Signals*, *Tecnicencia* 10 (2015) 5 – 10. doi:10.18180/tecnicencia.2015.19.2.
- [50] B. Liu, Z. Zhang, G. Xu, H. Fan, Q. Fu, Energy efficient telemonitoring of physiological signals via compressed sensing: A fast algorithm and power consumption evaluation, *Biomedical Signal Processing and Control* 11 (Supplement C) (2014) 80 – 88. doi:https://doi.org/10.1016/j.bspc.2014.02.010.

- [51] M. documentation, Matching pursuit algorithms (2017).
 750 URL <https://www.mathworks.com/help/wavelet/ug/matching-pursuit-algorithms.html?requestedDomain=true#>
- [52] A. Burns, B. R. Greene, M. J. McGrath, T. J. O'Shea, B. Kuris,
 755 S. M. Ayer, F. Strojescu, V. Cionca, ShimmerTM - a wireless sensor platform for noninvasive biomedical research, *IEEE Sensors Journal* 10 (9) (2010) 1527–1534. doi:<https://doi.org/10.1109/JSEN.2010.2045498>.
- [53] D. O'Loughlin, A. Coffey, F. Callaly, D. Lyons, F. Morgan, Xilinx vivado high level synthesis: Case studies, in: 25th IET Irish Signals Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communications Technologies (ISSC 2014/CIICT 2014), 2014, pp. 352–356. doi:[10.1049/cp.2014.0713](https://doi.org/10.1049/cp.2014.0713).
- [54] S. Ravi, M. Joseph, Tthls: an hls tool for testable hardware
 765 generation, *Design Automation for Embedded Systems* 21 (3) (2017) 231–246. doi:[10.1007/s10617-017-9192-x](https://doi.org/10.1007/s10617-017-9192-x).

Figure 1: Overview; (a) Remote health care system for elderly people (b) Diagram of the proposed implementation on the Zynq for PS and PL

Table 1: Complexity of CS Algorithms [27]

Algorithm type	Complexity
Greedy algorithms [28]	$O(MN \log K)$
Convex optimization algorithms [29]	$O(M^2 N^{3/2})$
Combinatorial algorithms [30]	$O(K(\log N)^c)$

Figure 2: Comparison of recovery quality of MP and OMP algorithm using PSNR metric

Figure 3: (a) Flow graph MP algorithm, (b) Compress block, (c) Recovery block

Figure 4: Comparison between Matlab and Zybo PS implementation

Figure 5: Diagram of the implementation in the Vivado Design Suite

Table 2: Used Directives for the MP algorithm

Instance	Directives	Function
Interface	S-AXI	Create AXI-lite slave interface
Loop	Unroll	Create multiple copy of the loop body
Loop	Pipeline	Creates a parallel process architecture from sequential operation

Table 3: Hardware resources and computational time

Block	Cycles	LUT (%)	FF	DSP(%)	BRAM(%)
Compress	10513	42	20	7	30
Recovery	4028	29	5	7	35

Table 4: Computational times and recovery quality of the MP algorithm

Input size	Iter	Matlab (ms)		PS (ms)		PL (ms)	PSNR (dB)	
		Compression	Recovery	Compression	Recovery	Compression	Matlab	Zybo
100 × 100	10	14.69	0.77	4.79	0.11	0.128	13.82	13.82
	20	14.73	0.84	9.58	0.24	0.129	21.31	21.31
	30	15.07	1.48	14.37	0.35	0.139	23.15	23.15
	40	17.30	2.26	19.16	0.48	0.150	23.69	23.67
	50	17.79	2.56	23.95	5.95	0.160	23.88	23.87

Figure 6: Recovered signal comparison between MATLAB and ZYBO for various numbers of iterations. (a) 10, (b) 30, and (c) 40 iterations.

Table 5: Computational time and optimizations used for the PL implementation of Algorithm 2

Matrix Size	Time (ms)	Directives
10×10	0.0178430	2x U
	0.1095100	5x U
	0.0087880	3U, 2x P
50×50	0.0162950	3x U
	0.8202950	1x U, 2x P
	0.1158370	4x P
100×100	1.6478150	n/a
	1.6140015	1x U
	0.8477940	2x P

U: Unroll directive, P: Pipeline directive

Table 6: Performance comparison implementation results against implementations of the OMP

Implementation	[38]	[40]	[41]	[42]	[44]	[45]	Proposed
Technology	90nm	Virtex-6	Kintex-7	Virtex-6	65nm	Kintex-7	Zybo
Precision (bits)	-	18	32	18	8.6	32	32
Max Freq (MHz)	29	100	53.7	120	272.5	87	33.3
Signal length	128	256	256	Up to 640	256	512	100
Reconstruction (<i>ms</i>)	0.23	0.63	1.43	0.34	0.0116	0.25	0.15
Block RAMs	-	258	382	576	-	64 /128	36
DSP cores	-	261	258	589	-	93	6
LUT Slices	-	32k	18k	6k	-	92k	7.3k
Area (μm^2)	-	-	-	-	384,886	-	-

Table 7: Estimation of power consumption

Power Consumption	Utilization details	Power (W)	Utilization (%)
Static	Device static	0.137	8
Dynamic	PS7	1.556	91
	Bram	0.065	4
	Signals	0.037	2
	Clocks	0.014	1
	Logic	0.009	1
	DSP	0.004	1