A SINGLE-PRECISION COMPRESSIVE SENSING SIGNAL RECONSTRUCTION ENGINE ON FPGAS

Fengbo Ren, Richard Dorrace, Wenyao Xu, Dejan Marković

Department of Electrical Engineering
University of California, Los Angeles
Los Angeles, CA 90095, U.S.A
E-mail: fren@ee.ucla.edu

ABSTRACT

Compressive sensing (CS) is a promising technology for the low-power and cost-effective data acquisition in wireless healthcare systems. However, its efficient real-time signal reconstruction is still challenging, and there is a clear demand for hardware acceleration. In this paper, we present the first single-precision floating-point CS reconstruction engine implemented a Kintex-7 FPGA using the orthogonal matching pursuit (OMP) algorithm. In order to achieve high performance with maximum hardware utilization, we propose a highly parallel architecture that shares the computing resources among different tasks of OMP by using configurable processing elements (PEs). By fully utilizing the FPGA recourses, our implementation has 128 PEs in parallel and operates at 53.7 MHz. In addition, it can support 2x larger problem size and 10x more sparse coefficients than prior work, which enables higher reconstruction accuracy by adding finer details to the recovered signal. Hardware results from the ECG reconstruction tests show the same level of accuracy as the double-precision C program. Compared to the execution time of a 2.27 GHz CPU, the FPGA reconstruction achieves an average speed-up of 41x.

1. INTRODUCTION

The compressive sensing (CS) framework offers a universal approach for compressive data sampling at sub-Nyquist rates [1], which promises to lower the cost and power consumption of data acquisition systems for a variety of signals [2]. However, the computational complexity of CS reconstruction algorithms is usually 1-2 orders of magnitude higher than orthogonal transforms [2]. This leads to inefficient real-time processing on general purpose processors, limiting the application of CS techniques on mobile platforms. Therefore, an efficient hardware solution is critically needed as it will benefit a variety of CS applications in terms of cost, portability and battery-life.

There have been a few related works on the VLSI design for CS signal reconstruction. The first FPGA implementation of the orthogonal matching pursuit (OMP) algorithm is presented in [3]. The application-specific integrated circuit (ASIC) implementations of several greedy algorithms are introduced in [4-7]. However, most of the existing work suffer from two major deficiencies that limit their practical usage: 1) they are all fixed-point implementations optimized for a particular input pattern and lack the flexibility to accommodate different applications. 2) The number of sparse coefficients supported by these implementations is so limited that the reconstruction accuracy cannot be guaranteed, especially when the signal is not ideally sparse on the selective basis.

Contributions: In this paper, we present the first single-precision floating-point CS reconstruction engine implemented a Kintex-7 XC7K325T FPGA using the OMP algorithm. In order to achieve high performance with maximum hardware utilization, we propose a highly parallel architecture and take into account resource sharing among different tasks of OMP by using configurable processing elements (PEs). The FPGA implementation has 128 PEs in parallel and operates at 53.7 MHz. It can support 2x larger problem size with 10x more sparse coefficients in signal reconstructions than prior VLSI designs. This allows more signal details to be recovered and leads to higher reconstruction accuracy. Emulation results from the ECG reconstruction tests show that the FPGA implementation can achieve the same level of accuracy as the double-precision C program with over 40x less reconstruction time.

2. PRELIMINARIES

The compressive sampling system is often modeled as a linear system given by

$$y = \boldsymbol{A}x + \beta, \tag{2.1}$$

where $x \in \mathbb{S}_k^n$ is a k-sparse signal given $\mathbb{S}_k^n = \{ x \in \mathbb{R}^n \mid \| x \|_0 \leq k \}$, $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is the sampling matrix, $\beta \in \mathbb{R}^m$ is a random noise with bounded energy $\| \beta \|_2 \leq \varepsilon$, and $y \in \mathbb{R}^m$ is the linear measurement or the sampled data. It has been proved that a $x \in \mathbb{S}_k^n$ can be exactly

978-1-4799-0004-6/13/$31.00 ©2013 IEEE

Table 1. Pseudo-Code of the OMP Algorithm.

Step	Operation
1	$r_0 = y, x_0 = \vec{0}, d = \vec{0}, \Lambda_0 = \emptyset, t = 1.$
2*	$c = \boldsymbol{A}^T r_{t-1}$, $\varphi = \arg\max_i \lvert c(i) \rvert$, $\Lambda_t = \Lambda_{t-1} \cup \varphi.$
3	$d(\Lambda_t) = \text{solve}_\alpha\{ \boldsymbol{A}_{\Lambda_t}{}^T \boldsymbol{A}_{\Lambda_t} \alpha = c(\Lambda_t)\},$
4	$r_t = r_{t-1} - \boldsymbol{A}_{\Lambda_t} d(\Lambda_t)$, $x_t = x_{t-1} + d$, $t = t + 1.$
5	If $\parallel r_t \parallel_2 \leq \varepsilon$, stop, otherwise, go to step 2

* Assuming that all atoms in $\boldsymbol{A}$ are normalized.

recovered by solving the ℓ_0 pseudo-norm minimization given as

$$\min \parallel x \parallel_0, \text{ subject to } \parallel y - \boldsymbol{A}x \parallel_2 \leq \varepsilon, \quad (2.2)$$

as long as $\boldsymbol{A}$ satisfies the *Null Space Property* (NSP) and the *Restricted Isometry Property* (RIP) of order $2k$ [1]. The minimization problem in (2.2) is NP-hard. Another approach to attack this problem is to relax the objective function in (2.2) to ℓ_1 norm minimization [1] as

$$\min \parallel x \parallel_1, \text{ subject to } \parallel y - \boldsymbol{A}x \parallel_2 \leq \varepsilon. \quad (2.3)$$

The problem in (2.3) is convex and can be solved through general convex optimization methods.

3. RECONSTRUCTION ALGORITHM

The concept of OMP [8] is built upon the following observation. When $x \in \mathbb{S}_k^n$ has only k non-zeros, the linear measurement $y = \boldsymbol{A}x$ can also be represented as $y = \boldsymbol{A}_\Lambda x(\Lambda)$, where Λ is the index set of the non-zero elements in x, called the active set. Note that given $\boldsymbol{A}_\Lambda \in \mathbb{R}^{m \times k}$ with $m > k$, $x(\Lambda)$ can be best approximated by solving the least square (LS) problem given as

$$\min \parallel y - \boldsymbol{A}_\Lambda x(\Lambda) \parallel_2, \quad (3.1)$$

as long as the active set Λ is identified.

The pseudo-code of the OMP algorithm is described in Table 1. There are three main tasks performed in each iteration: atom searching (AS) for identifying the active set (Step 2), least square (LS) solving for computing the updating direction (Step 3), and estimation update (EU) (Step 4). At iteration t, the number of floating point operations (FLOPs) required by each task is $2nm$, $2mt + 3t^2$, and $2mt$, respectively. Given $n > m \gg k$ [1] and $t \leq k$ [8], the AS task is found to be the performance bottleneck as it contributes the most computation in OMP. On the other hand, the LS task plays a pivotal role on the hardware utilization efficiency. This is because the matrix factorization in the LS task involves a variety of operations with complex data flows, introducing extra hardware complexity in terms of control and scheduling. However, due to the proper use of factorization techniques (see Section 4.3), the LS task only contributes a small part of the total computation. As a result, any hardware resource dedicated to performing the LS task will have a very low utilization rate.

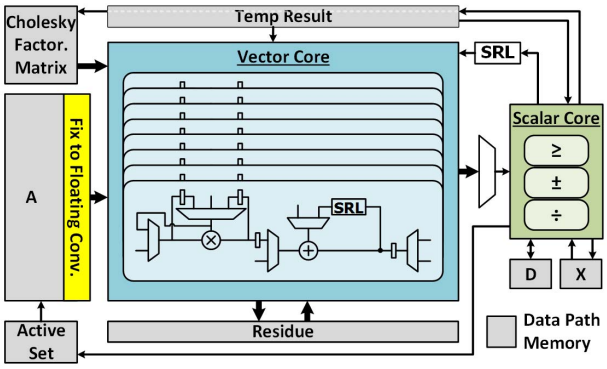

Fig. 1. VLSI architecture of the reconstruction engine.

4. HARDWARE IMPLEMENTATION

In order to achieve high performance with maximum hardware utilization, we design a highly parallel VLSI architecture with configurable PEs, where all the computing resources for performing the AS task are also reused by other tasks. Such resource sharing strategy improves the hardware utilization, allowing for more PEs to be mapped onto the FPGA for further speed-up.

4.1. Architecture Design

The top-level block diagram of the proposed VLSI architecture is shown in Fig. 1. The vector core integrates multiple PEs in parallel with a flexible data-path. Both the PE and the data-path connection can be dynamically configured through different microcode. This enables the vector core to support a selected set of vector operations. A shift register logic (SRL) unit is used in the feedback path of each PE for providing the folding capability to process long vectors.

The scalar core supports scalar comparison, addition, accumulation, and division. Depending on the top-level data-path configuration, the scalar core can either post-process a selected result from the vector core (e.g. inner product) or process independent data separately. When the two cores are connected as a processing group, more advanced operations, such as correlation sorting, forward substitution (FS), and backward substitution (BS), can be executed. Note that the global feedback path with SRL unit is dedicated for performing FS and BS with iterative vector operations.

The computation cores are linked by seven data-path memories. Note that in our design, the sampling matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is explicitly stored in order to accommodate the CS reconstructions on different basis. The complex data flow of OMP is enforced by customized local memory controllers, which are synchronized by a top-

level finite-state machine (FSM). Using data-path memories helps improve the power efficiency especially when the supported problem size is large, as most of the data movement can be replaced by pointer manipulations.

4.2. Data Fromat

In order to figure out the word-length required to preserve the software-level accuracy for different CS problems, we investigate the dynamic range (DR) requirement of each computing unit in our design using double-precision MATLAB simulations. The worst-case DR requirement of the vector core is found to be over 70 bits. Such a large DR is mainly due to the solution searching characteristic of OMP that the scales of most variables are gradually scaled down as the residue approaches zero. Besides, sharing the computing resources also tends to enlarge the DR as it has to cover the worst case of all different tasks. As a result, a fixed-point implementation would incur large area overhead for preserving the software-level accuracy in different CS problems. Therefore, we adopt the single-precision floating-point data format in our design. It can provide the required DR with a 32-bit word-length, leading to significant area reduction especially in the data-path memories.

Nonetheless, $\boldsymbol{A}$ is still stored in a fixed-point data format. Since it needs not to be updated during the operations, thereby having a limited DR. A parallelized fixed-point to floating-point conversion interface is inserted at the memory output (Fig. 1).

4.3. Square-Root-Free OMP

In our design, the infrequent square root operations are avoided through algorithm reformulation. For solving the LS problem in (3.1), we adopt an alternative Cholesky factorization method [9] given by

$$\boldsymbol{\Phi} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\boldsymbol{T}} \tag{4.1}$$

where $\boldsymbol{\Phi} = {\boldsymbol{A}_{\Lambda_t}}^T \boldsymbol{A}_{\Lambda_t}$ is a positive definite matrix, $\boldsymbol{L}$ is a lower triangular matrix with normalized diagonal elements, and $\boldsymbol{D}$ is a diagonal matrix that absorbs the square-rooted factors from both $\boldsymbol{L}$ and $\boldsymbol{L}^{\boldsymbol{T}}$ and cancels the square root operations. The stopping criterion $\| r_t \|_2 \leq \varepsilon$ can be reformulated as ${r_t}^T r_t \leq \varepsilon^2$ and computed with doubled dynamic range.

5. EXPERIMENT RESULTS

The single-precision CS reconstruction engine is implemented on a Xilinx Kintex-7 XC7K325T-FBG900 FPGA (speed grade -2). By efficiently utilizing the resources on the FPGA, the reconstruction engine has 128 PEs in parallel and can support a flexible problem size of $n \leq 1740$ and $m \leq 640$. To the best of our knowledge, it is the largest problem size supported by VLSI designs to date. In addition, our implementation can support up to $k \leq 320$ sparse coefficients in signal reconstructions, which is almost 10x more than previous work [3-7]. Reconstructing more coefficients adds finer details to the recovered signal. This is critical to the reconstruction accuracy, especially when the signal is not ideally sparse on the selective basis.

Table 2. Implementation Results on FPGA*

	System	Usage Breakdown†		
		VC	SC	DMs
Frequency	53.7 MHz			
LUTs	186,799 (91%)	90%	3%	7%
DSP48s	258 (31%)	99%	0%	1%
Block RAMs	382 (86%)	0%	0%	100%

* Xilinx Kintex-7 XC7K325T-FBG900

† VC: vector core, SC: scalar core, DMs: data-path memories

Table 2 summarizes the system performance and the resource utilization profile of the FPGA implementation. After performing global retiming, the critical path is found to pass through the floating-point multiplier. A system frequency of 53.7 MHz is realized. The implementation uses 91% of the LUTs, 31% of the DSP48 slices, and 98% of the BRAMs on the FPGA device. Note that all the BRAMs are used to construct data-path memories, where 65% is used for storing $\boldsymbol{A}$ in a 10-bit fixed-point data format. Resulting from the resource sharing efforts, the vector core has a constant 100% temporal utilization rate during the operations. Moreover, 90% of the LUT usage is dedicated to the parallelization of the vector core for achieving higher performance.

In order to evaluate the reconstruction accuracy, we use the ECG data from MIT-BIH database [10] to perform CS sampling and reconstruction. In our experiment, the ECG signals are segmented with a fixed window size of $n = 1024$ and sampled by Bernoulli random matrices with different compression (m/n) ratios. For comparison purposes, the signals are reconstructed on the DWT basis

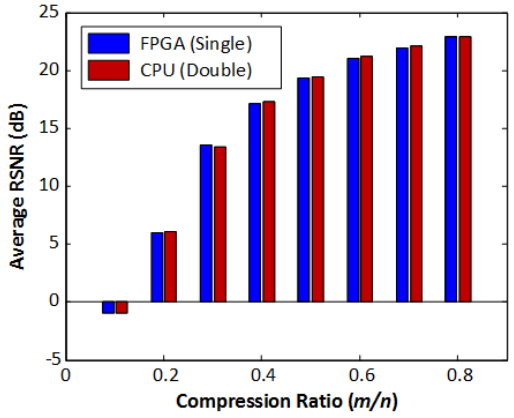

Fig. 2. Average RSNR performance measured from the ECG reconstruction in C program and on FPGA at different compression ratio (m/n).

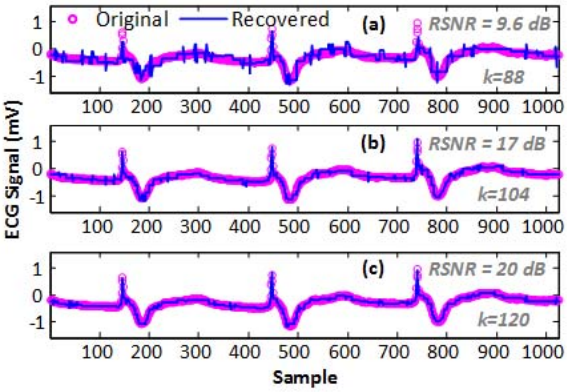

Fig. 3 An ECG signal recovered by FPGA with a compression ratio of (a) $m/n = 0.2$, (b) $m/n = 0.3$, and (c) $m/n = 0.4$.

using both C program and the FPGA implementation. The reconstruction accuracy is measured by the reconstruction SNR (RSNR) defined as

$$\text{RSNR}=20 \cdot \log_{10}\left(\frac{\|x\|_2}{\|x-\hat{x}\|_2}\right). \quad (5.1)$$

where $x \in \mathbb{R}^n$ is the original signal, and $\hat{x} \in \mathbb{R}^n$ is the reconstructed signal. Figure 2 shows the average RSNR performance measured from the C program (double-precision) and the FPGA reconstructions (single-precision). As the floating-point data format is used, the reconstruction engine on FPGA is able to achieve the same level of accuracy as the C program in this test. Figure 3 illustrates an ECG signal recover by the FPGA implementation versus the original signal at different compression ratio. Note that as the RSNR performance is improved with more measurements, the number of sparse coefficients (k) that needs to be reconstructed also increases. This clearly shows the importance of supporting a large k for achieving a good reconstruction quality. In this sense, our implementation that offers 10x larger such capability than prior work is a more viable computing solution for the accuracy-driven CS problems.

On the other hand, reconstructing more coefficients requires more iterations of the OMP algorithm thereby more reconstruction time. However, this overhead can be disregarded as long as the real-time requirement can be met by the FPGA acceleration in most applications. With 128 PEs operating at 53.7 MHz in parallel, the ECG reconstruction (Fig. 3) on FPGA takes 39.9 μs per iteration. Compared to the execution time of the C program powered by a 2.27 GHz CPU, over 40x speed-up is achieved on average.

6. CONCLUSION

In this paper, we present the first single-precision floating-point CS reconstruction engine implemented on a Kintex-7 XC7K325T FPGA. In order to achieve high performance with maximum hardware utilization, we design a highly parallel VLSI architecture with configurable PEs for enabling the resource sharing between different tasks of OMP. By fully utilizing the recourses on FPGA, our implementation has 128 PEs in parallel and operates at 53.7 MHz. In addition, it can support twice the problem size and nearly 10x more sparse coefficients in signal reconstructions than prior VLSI designs. Emulation results from the ECG reconstruction tests show that the proposed hardware solution can achieve the same level of accuracy as the software counterpart. Compared to the execution time of a 2.27 GHz CPU, an average speed-up of 41x is achieved by the FPGA implementation.

7. REFERENCES

[1] E. Candès, et al., "An Introduction to Compressive Sampling", *IEEE Signal Process. Mag.,* vol. 25, no. 2, pp. 21-30, Mar. 2008.

[2] M. Duarte, et al., "Single Pixel Imaging via Compressive Sampling", *IEEE Signal Process. Mag.,* vol. 25, no. 2, pp. 83-91, Mar. 2008.

[3] A. Septimus, et al., "Compressive Sampling Hardware Reconstruction", in *Proc. Int. Symp. Circuits and Systems (ISCAS),* Paris, Fance, May 2010, pp. 3316-3319.

[4] P. Maechler, et al., "Matching Pursuit: Evaluation and Implementation for LTE Channel Estimation", in *Proc. 44th Asilomar Conf. Signals, Systems and Computers,* Pacific Grove, CA, Nov. 2010, pp. 400-405.

[5] J. Stanislaus, et al., "High Performance Compressive Sensing Reconstruction Hardware with QRD Process", in *Proc. Int. Symp. Circuits and Systems (ISCAS),* Seoul, Korea, May 2012, pp. 29-32.

[6] M. Patrick, et al., "VLSI Design of Approximate Message Passing for Signal Restoration and Compressive Sensing", *IEEE J. Emerg. Sel. Topic Circuits Syst.,* vol. 2, no. 3, pp. 579-590, Sep. 2012.

[7] L. Bai, et al., "High-Speed Compressed Sensing Reconstruction on FPGA Using OMP and AMP", in *Proc. 19th Int. Conf. Electronics, Circuits and Systems (ICECS),* Seville, Spain, Dec. 2012, pp. 53-56.

[8] J. Tropp, et al., "Signal Recovery from Random Measurements via Orthogonal Matching Pursuit", *IEEE Trans. Inform. Theory,* vol. 53, no. 12, pp 4655-4666, Dec. 2007.

[9] D. Yang "Turbo Bayesian Compressed Sensing", Ph.D. dissertation, Dept. Elect. Eng., Univ. of Tennessee, Knoxville, Knoxville, TN, 2011.

[10] G. Moody, et al., "The impact of the MIT-BIH Arrhythmia Database," *IEEE Eng. in Med. and Biol.,* vol. 20, no. 3, pp. 45-50, May 2001.