

A review of optimisation and least-square problem methods on field programmable gate array-based orthogonal matching pursuit implementations

Muhammad Muzakkir Mohd Nadzri, Afandi Ahmad

Reconfigurable Computing for Analytics Acceleration (ReCAA) Research Laboratory,
Microelectronics and Nanotechnology Shamsuddin Research Centre (MiNTSRC), Universiti Tun Hussein Onn Malaysia (UTHM),
Malaysia

Article Info

Article history:

Received Jul 31, 2021
Revised Nov 29, 2021
Accepted Dec 17, 2021

Keywords:

Compressive sensing
FPGA
Least-square problem
Optimisation
Orthogonal matching pursuit

ABSTRACT

Orthogonal matching pursuit (OMP) is the most efficient algorithm used for the reconstruction of compressively sampled data signals in the implementation of compressive sensing. OMP operates in an iteration-based nature, which involves optimisation and least-square problem (LSP) as the main processes. However, optimisation and LSP processes comprise complex mathematical operations that are computationally demanding, and software-based implementations are slow, power-consuming, and unfit for real-time applications. To fill the research gap, we reviewed the optimisation and LSP techniques implemented on the FPGA platform as the hardware accelerator. Aspects that contributed to the performance, algorithm, and methods involved in the implemented works were discussed and compared. The methods were found to be improved when modified or combined. However, the best approach still depends on the requirement of the system to be developed, and this review is significant as a reference.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Afandi Ahmad
Reconfigurable Computing for Analytics Acceleration (ReCAA) Research Laboratory
Microelectronics and Nanotechnology-Shamsuddin Research Centre (MiNTSRC)
Universiti Tun Hussein Onn Malaysia (UTHM), Malaysia
Email: afandia@uthm.edu.my

1. INTRODUCTION

Signal compression conventionally depends on the Shannon sampling theorem [1], [2] to ensure the accomplishment of signal recovery. In recent years, compressive sensing (CS), as proposed in [3]-[5] has appeared as a new technique that can sample a signal at a rate lower than the Nyquist rate but still promises a reliable reconstruction signal. Despite the advantages offered by CS, it currently faces challenges in the reconstruction part of the signal, as the computation process is complex. Software-based CS is usually implemented in the graphic processor units (GPUs) and general central processing units (CPUs) of computers, with the GPU implementation being more efficient than the CPU implementation. However, a GPU implementation does not allow for the regular flow of data to the host, since it suffers from the significant issue of inconsistent memory bandwidth between the GPU and the main memory [6], rendering it incapable of attaining adequate real-time results on the recovered signal [7]. Thus, it is critical to provide a hardware-based enhancement, for example, using a field-programmable gate array (FPGA), to accelerate the computation as a solution that links to the analogue sensors to compute signals in real time manner [8].

Among various algorithms, orthogonal matching pursuit (OMP), introduced in [9], is a famous method implemented on the hardware platform and is an efficient reconstruction algorithm with an excellent

trade-off between computational complexity and accuracy. OMP, also known as the greedy method, works by finding the location of the non-zero component of the dictionary and selects the columns of the measurement matrix which mostly correlate with the measurement vector, a process called optimisation, and solves the least-square problem (LSP) to find the estimated signal. Optimisation and LSP are two (2) main problems in the computation process of OMP as the implementation of optimisation is time-consuming, since it usually requires a large number of matrix multiplications, and solving the LSP involves a complex matrix inversion. Both play a significant role in OMP iterative processes but significantly affect the cycle period.

The CS equation is $y = \Theta x$, where the output compressed signal y is obtained by multiplying k -sparse (non-zero element) from input signal x and measurement matrix Θ . Input signal x is known as k -sparse, as it will be sparse in the time domain or other appropriate transform domains after transformation, such as discrete cosine transform basis, Fourier basis and wavelet basis [10]. To ensure that signal x can be decompressed or recovered successfully, measurement matrix Θ must obey two (2) criteria, which are restricted isometry property (RIP) and incoherence aspect [11], [12].

To reconstruct signal x from compressed signal y , an underdetermined linear equation must be resolved, and one (1) of the famous and reliable methods is using OMP, where its full procedure is given in Algorithm 1 and its flow is demonstrated in Figure 1(a). As shown in Algorithm 1, the procedure begins with the initialisation of iteration counter i , active set Λ_0 , and residual R_0 initialise as y , where the provided input values are measurement matrix Θ , compressed signal y and k iterations. The second step is the optimisation operation, which is finding the most correlated column by computing the inner product of the current residual value from each iteration with the measurement matrix to be added to the active set. Next, in each iteration, the active set will be updated with the current index set of non-zero elements. After that, based on the current active set, the LSP method is used to find \hat{x} as the approximation of the original signal. In the last step, the residual vector is updated and the process is repeated for k number of iterations. After a few cycles, by including the processes of optimisation and LSP, the estimation of \hat{x} as the approximated value of the original signal x will be recovered.

Algorithm 1. OMP Algorithm

```

1- Initialise  $i = 0$ ; index set  $\Lambda_0 = \{\emptyset\}$ ;
   residual  $R_0 = y$ 
2- Find index  $\lambda_i = \arg \max_{j=1..N} |R_{i-1} \theta_j|$ 
3- Update  $\Lambda_i = \Lambda_{i-1} \cup \{\lambda_i\}$  and  $\theta_{\Lambda_i} = \theta_{\Lambda_{i-1}} \theta_{\lambda_i}$ 
4- Solve LSP:  $\hat{x}_i = \arg \min_x \|y - \theta_{\Lambda_i} x\|$ 
5- Update residual  $R_i = y - \theta_{\Lambda_i} \hat{x}_i$ 
6- Counter  $i = i+1$  and repeat step 2 if  $i < k$ 
7- Result of the final estimation of  $\hat{x}$ 

```

There are various OMP methods on the FPGA platform used in previous works in the last five (5) years [6], [13]-[21]. However, there are limited reviews on OMP methods, especially in terms of techniques and the performance of the FPGA with emphasis discussion on the implementation of optimisation and LSP. Most reviews broadly discussed the CS reconstruction approach [22], [23] and did not focus on OMP and FPGA implementations. To fill this research gap, the present review focused on the methods of optimisation and least-square problem used in FPGA-based OMP reconstruction algorithm implementations. This paper would benefit researchers by comparing each method, analysing and assessing the most suitable methods to be implemented and improved upon.

This review is also significant, since CS has been implemented in various research areas and applications, such as compressive imaging [24]-[26], biomedical applications [27], [28], pattern recognition [29], [30], sound processing [31]-[34], video processing [35], [36], and microelectronics applications [37], [38], and its most well-known implementation is in the field of communication systems, such as wireless networks and antenna [39]-[47]. It has been shown that this field is growing and has a wide opportunity for exploration using this technique of compression. The enhancement in terms of computation process, especially in terms of hardware, is really significant to accommodate the demand of real-time applications for reduced latency, time consumption, memory storage, and power consumption.

The remaining sections of this paper are organised as follows. A review of FPGA-based OMP implementations is discussed and explained in section 2. Several flow charts focusing on optimisation and LSP processes in current methods of OMP implementation are depicted and important parameters are also compared. OMP's optimisation and LSP strategies are described in sections 3 and 4, respectively. Lastly, the conclusion is outlined in section 5.

2. FPGA-BASED OMP IMPLEMENTATIONS

In this section, a discussion on previous works on the implementation of FPGA-based OMP is presented. The main objective in this section is to review implementation strategies that emphasise optimisation and LSP. In [6], the architecture proposed was divided into four (4) computing blocks: K -point inner product and comparator unit (K -IPCU), Cholesky inversion unit (CIU), residual computation unit (XCU), and reconstructed signal computation unit (RCU). In these four (4) blocks, all input data, as well as intermediate matrices and vectors, were stored in the memory elements. The optimisation process was enhanced by the K -IPCU block with the ability of parallel pipeline inner product computation and comparison in each cycle. Modified Cholesky decomposition combined with the Newton-Raphson method was used to solve the complexity of matrix inversion for the LSP. The proposed architecture design was able to optimise both area and time executions by reusing the hardware of matrix-vector multiplication for other components of the algorithm and exploiting the parallelism inside each dependent operation.

The improved OMP method to solve the crucial optimisation process using partial Fourier dictionary, which is commonly used in radar imaging, was proposed in [13]. Fast Fourier transform (FFT) was used to optimise correlation, and the conjugate gradient (CG) method was used to solve the large-scale least-square issue. The suggested method was based on the partial Fourier basis, also known as the modulated Fourier basis, which is less complex and has been reported as efficient when implemented on the hardware platform. Most importantly, by utilising fast transformation, it was feasible to compute the highest correlation in the optimisation process in real time. Moreover, the matrix may be preconstructed, and so only one (1) vector from the dictionary matrix had to be stored. In the normal implementation of OMP, the LSP process is executed in each iteration. The Gram-Schmidt orthogonalisation method was utilised, and hence the process became efficient, as the LSP process was only executed once. The LSP implementation also utilised the CG iteration method to figure out the sparse solution.

The idea of the algorithmic transformation method, known as matrix inversion bypass (MIB), was discussed in [14]. MIB enhances the implementation process of optimisation and LSP by parallelising the process. By manipulating the results from the previous iteration, it decouples the computation of intermediate signal estimates and is able to bypass the matrix inversion operation. Both optimisation and LSP are indirectly improved from the implementation of MIB.

The optimisation process of OMP is improved by the implementation of the particle-in-cell (PIC) method, as explained in [15]. PIC is a method that can select two (2) relevant indices from the measurement matrix in each cycle to improve the OMP algorithm, hence resulting in decreased number of iterations by nearly half the number of cycles. It also reduces the chance of missing the true index and of choosing an incorrect one, as compared with traditional OMP methods. In terms of LSP implementation, common Moore pseudoinverse was used by solving using the modified Cholesky decomposition method. Instead of utilising the Newton-Raphson method to solve the division operation, as in [6], the Goldschmidt algorithm was used. The Goldschmidt approach may be employed because of the parallel action in the matrix inversion unit, which varies from the Newton iterative methods in the serial procedure.

The partitioned inversion method based on the incremental computation technique for a better OMP was proposed in [16]. While solving the LSP in the OMP algorithm, the suggested partitioned inversion, which used the inversion result from the previous iteration, reduced the complexity of the matrix inversion operation at every loop cycle. By re-utilising the computation results obtained in the previous OMP iteration, three (3) properties of the input matrix, known as conjugate symmetry, positive definiteness, and overlapped regions, for the inversion in each OMP iteration were utilised, hence reducing computation complexity. In addition, multiple measurement vectors (MMVs) were applied to the OMP algorithm to improve the sparse signal recovery compared with the single measurement vector, and this is called simultaneous OMP (SOMP).

Improvements were made for optimisation by focusing on hardware implementation [17]. The optimiser unit was primarily composed of three (3) computational components: fixed-point multiplier unit, adder tree, and maximum index selection. These components were organised in parallel and utilised at several stages of the OMP process. An alternative Cholesky decomposition, or modified Cholesky, was chosen to solve the LSP with the Goldschmidt method to solve the division operation, instead of using Newton-Raphson.

Quick response (QR) factorisation was preferred over other methods [18]. QR factorisation is recommended when the measurement matrix is unstructured and dense, despite the fact that it has a higher computational complexity than other approaches. In the QR and matrix inversion blocks, a dedicated divider was utilised to conduct the division operation and to discover the reciprocal. To eliminate LSP recurrence in each iteration, the modified Gram-Schmidt algorithm was used. A single matrix multiplication unit (MMU) was proposed to execute all operations in order to break the nature of the OMP, which is dependent on the preceding stage. For correlation optimisation, the suggested MMU supported two (2) types of multiplications: vector-vector multiplication and scalar-vector multiplication.

Square-root-free QR decomposition was introduced in [19]. The proposed algorithm adopted an incremental quick response decomposition (QRD), and hence it was further optimised by eliminating the square-root operation to ease hardware implementation. The suggested design, which avoided the complicated square root unit, consisted mostly of some basic computing units, in which the computing process was broken down into numerous simple operations that may be mapped to the appropriate hardware for pipelining to optimise OMP implementation.

A new architecture of a low-power fast OMP (LPF-OMP) algorithm was presented in [20]. The LPF-OMP can avoid the computation of pseudoinverse to save time and storage requirement to store the pseudoinverse matrix. The proposed algorithm used a partial evaluation of incremental QR decomposition using the modified Gram-Schmidt algorithm. To minimise reconstruction time, the correlation phase of the traditional OMP method was changed to search over a varied number of columns in subsequent iterations.

The OMP implementation used Gram-Schmidt orthogonalisation to enhance the signal residuals' update process so that the signal recovery only had to execute the least-square solution once, resulting in a significant reduction in the number of matrix operations in hardware implementation [21]. At the final step, the LSP was solved using Cholesky factorisation. The research work also focused on FPGA designs using hardware description language (HDL) and high-level synthesis (HLS), as HLS is a new design that is based on the C/C++ trend, which can reduce the time-to-market of design implementation.

As the summarisation of this section, the flow charts of the methods implemented are depicted in Figure 1, and important parameters comprising FPGA performance, signal recovery, and the optimisation and LSP methods used are tabulated in Table 1 (see Appendix). Figure 1 shows the flow charts of the OMP implementation processes in the previous works. From the methods implemented, six (6) flow charts have been summarised. In the flow charts, the dark-coloured boxes represent the methods approached to enhance the computation in the optimisation and LSP processes. Figure 1(a) shows the flow chart of the normal OMP implementation, while the other five (5) show the flow charts of the improved OMP processes. In Figure 1(b), the flow chart shows that the optimisation process was enhanced by using K-IPCU, PIC, and parallel hardware optimisation, while the LSP process used the combination of modified Cholesky + Newton Raphson, Cholesky + Goldschmidt, and Cholesky decomposition + Goldschmidt. In Figure 1(c), the optimisation process used FFT, Single MMU, and parallel hardware optimisation, but the flow in the LSP process was different, as the Gram-Schmidt method was applied to execute the LSP process just once after the iterations were completed.

Next, in Figure 1(d) and Figure 1(e), both the optimisation and LSP processes were replaced with the MIB and partition inversion methods, as these methods can bypass the inversion operation. Lastly, the flow chart in Figure 1(f) is almost similar to that in Figure 1(c) but the difference is the implementation of the Gram-Schmidt method. The Gram-Schmidt method in Figure 1(f) is a modified version that can avoid the pseudoinverse implementation. All flow charts show that a lot of methods were used to enhance the implementation processes of optimisation and LSP. Each method aimed to reduce the complexity of computation by avoiding some operations in the repetitive OMP iterations. The flow charts will benefit researchers in comparing and analysing each method.

In Table 1 (see Appendix), several FPGA performances are tabulated. The FPGA as the hardware accelerator using several main components of the hardware unit, such as the multiplexer, adder tree, reciprocal, memory, and registers, was explored to increase the parallelism capabilities in the implementation of optimisation and LSP. Parallelism capabilities were proposed, which utilised, reused, or shared the computing hardware components' resources between the various tasks and stages in the OMP reconstruction algorithm [48]-[50]. The process of parallelising various tasks in the OMP implementation will minimise implementation time and the performance for signal reconstruction can be better.

Nevertheless, the performance of parallel architectures designs still requires intensive exploration regarding output quality versus output performance of the reconstructed signal in the implementation of the OMP. There are a lot more of recent OMP implementation strategies, such as those presented in [51], which were only simulated using the MATLAB software but not on the hardware platform. Implementation in terms of hardware to accelerate the process of the new strategies will benefit the overall implementation in order to have an efficient real-time system.

There was also a discussion in [21] concerning the design implementation of the FPGA either using hardware description languages (HDLs), such as Verilog and very-high-speed integrated circuit (VHSIC) hardware description language (VHDL), or using high-level synthesis (HLS). HLS uses the behaviour of the C/C++ description of a system to automatically generate the HDL design description instead of using the original HDL implementation to design the architecture. HSL eases the process and increases design productivity but it may affect in terms of time and usage of area.

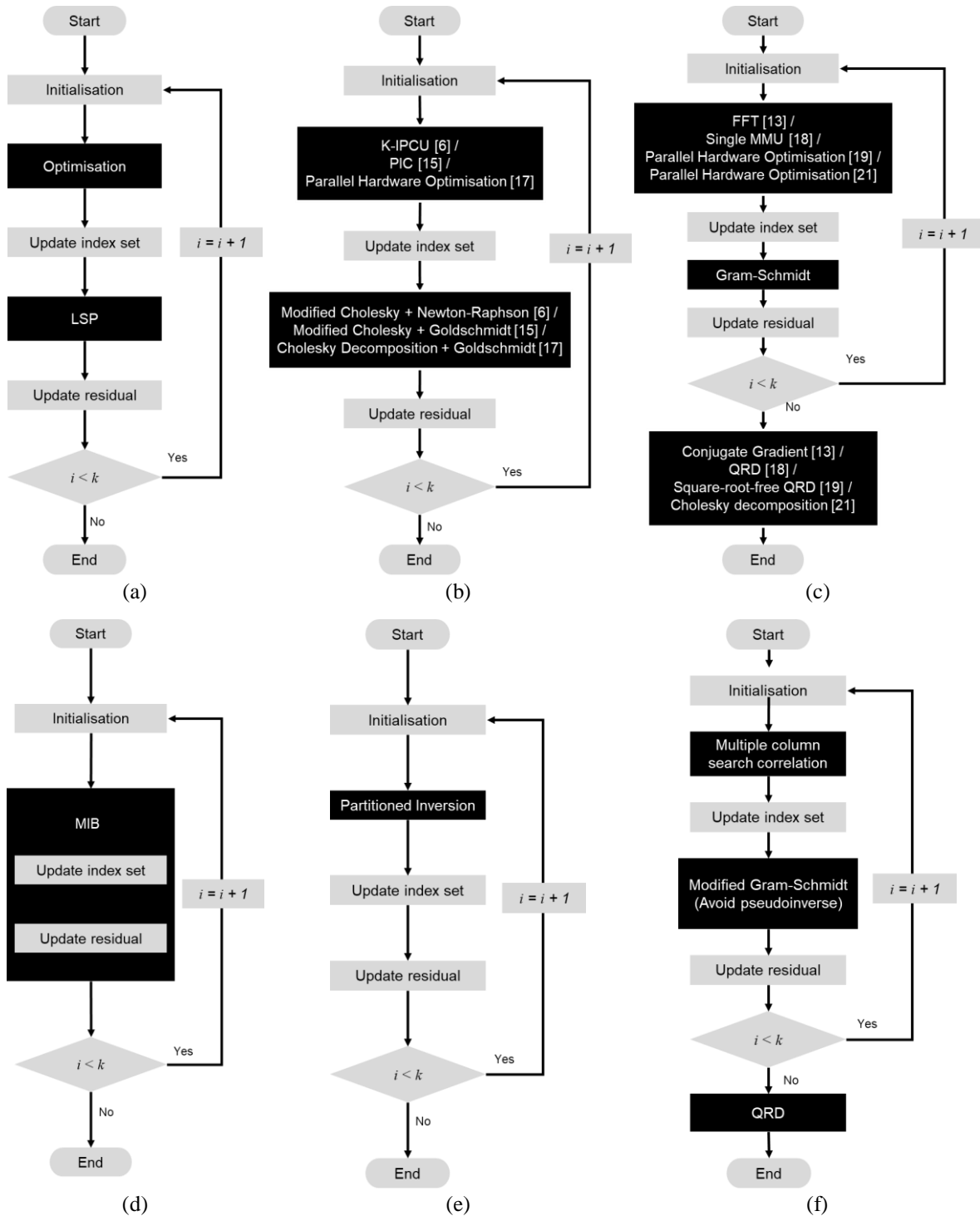


Figure 1. Flow charts of methods for (a) Normal OMP, (b) Improved OMP in [6], [15], [17], (c) Improved OMP in [13], [18], [19], [21], (d) Improved OMP in [14], (e) Improved OMP in [16], and (f) Improved OMP in [20]

3. OPTIMISATION STRATEGIES

As stated in Algorithm 1, the optimisation equation of $arg\ max_{j=1..N} |\langle R_{i-1}, \theta_j \rangle|$ is an important operation in the OMP algorithm to find the most correlated column of measurement matrix θ with residual vector R . This process is called inner product computation (IPC) or matrix-vector multiplication. For a large-scale measurement matrix, the correlation process is time-consuming because it involves a large number of matrix multiplications.

There are several approaches to solve the optimisation problem in terms of computation, such as using Fourier basis, parallel correlation indices, and MIB, but most available studies in the literature usually solved the optimisation problem by utilising FPGA hardware parallelism capabilities using mainly three (3) computing parts: multiplier unit, addition mechanism, and index selection. A multiplier unit is a component used as the multiplier computation in the optimisation process. Finding the most correlated column usually involves a massive multiplication operation, depending on the size of the sampling matrix (row and column), and hence is reflected by the number of multiplier units. The multiplier unit is commonly designed to suit the various stages of the OMP algorithm, which also involves vector-vector multiplication. Parallelism capabilities of the FPGA include pipelining each of the multiplier method performed to keep timing and latency low. This approach indirectly enhances the efficiency of OMP implementation.

The result from the multiplier operation next must be added to obtain the matrix-vector multiplication result. An FPGA addition mechanism, such as the efficient architecture of the adder tree, is usually proposed. Because adders are also needed in several stages of OMP implementation, various architecture designs of the adder tree, depending on the number of adders, are pipelined to increase the performance. The final goal of the optimisation process is to find the most correlated column index with the residual. In each cycle, after the multiplier and adder operations, two (2) unit of register and multiplier are used to store and compare the values using the comparator, and hence the maximum value is obtained by repeating this process. Instead of enhancing optimisation computation using hardware, approaches in terms of improving algorithm computation will also boost the optimisation process, as depicted in Table 2. Both software and hardware enhancements can be utilised to enhance the performance of OMP.

Table 2. Optimisation solutions based on algorithm computation

Reference	Algorithm	Features
[13]	Partial Fourier Basis (Fast Fourier Transform)	<ul style="list-style-type: none"> • Less complexity in hardware • Fast-transform
[14]	Matrix Inversion Bypass (MIB)	<ul style="list-style-type: none"> • Only one (1) vector to store from dictionary matrix • Decouples the computation of intermediate signal estimates • Bypasses matrix inversion
[15]	Parallel correlation indices	<ul style="list-style-type: none"> • Can select the two (2) relevant indices from the measurement matrix in each iteration • Reduce almost half of the number of iterations

4. LSP STRATEGIES

The LSP is one (1) of the important operations to reconstruct the original signal from the compressed signal. In the implementation of the OMP algorithm, the LSP is solved in each iteration with a complex mathematical operation, which increases the latency of the OMP implementation. Figure 2 shows the block diagram of the common LSP operation.

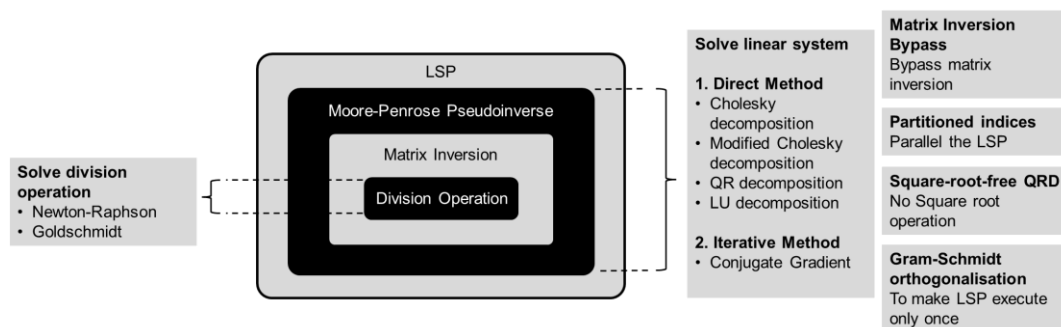


Figure 2. Methods to solve LSP

The LSP block diagram, as shown in Figure 2, involves several arithmetic operations and is commonly solved using the Moore-Penrose pseudoinverse method. Moore-Penrose pseudoinverse [52], expressed by $A^+ = (A^T A)^{-1} A^T$, where A^+ is the pseudoinverse of Moore-Penrose, involves the operation of matrix inversion and requires the division operation. The naive implementation of matrix inversion requires high computational effort. To make the implementation more efficient, various methods, such as coordinate

rotation digital computer (CORDIC), Newton-Raphson, Cholesky factorisation, and QR decomposition, have been presented and the common methods are summarised in Figure 2.

Figure 2 shows two (2) types of linear system solution methods. The first is direct methods, where the solution is computed through lower/upper triangular decomposition and by solving the triangular system. The second is the iterative method, where the solution is approximated by performing iterations from an initial vector. Direct methods are more suitable for small systems compared with iterative methods, which are suited for solving larger-scale problems. The conjugate gradient (CG) method is a well-studied iterative method that has been shown to be highly efficient in software simulations and resilient at solving large sparse linear systems.

From the review, in the last five (5) years, the common approach implemented in the FPGA to solve the Moore-Penrose pseudoinverse is by using direct methods, as these promise better results. The only iterative method, which is the conjugate gradient method, was implemented in [13], as the proposed work focused on applications with larger data. Some methods were modified, such as modified Cholesky, as the modification makes the system not require the square-root operation. Also, some implementations utilised the method of Gram-Schmidt orthogonalisation, as it can make the process of high computation of the LSP be executed only once instead of in each iteration. Some other methods have also been implemented, such as MIB, which can bypass the matrix inversion, partitioned indices, which can divide the input matrix, and square-root-free QR decomposition, which can evade the square-root process.

In terms of the division operation, instead of using the Newton-Raphson method, as in [6], the Goldschmidt algorithm has also been implemented. The Goldschmidt algorithm was employed due to the parallel operation in the matrix inversion unit, which was different from the Newton iterative method with its serial process. One of the main characteristics of the Goldschmidt algorithm, unlike the Newton-Raphson algorithm, is that the multiplications are independent. This property makes the Goldschmidt algorithm suitable for hardware implementation. Parallel computing can double the computational speed of division and iterative operations.

The overall comparison of the methods is summarised in Table 3, and the important operations involved, which are division, square root, LSP executed once, and avoid pseudoinverse, are tabulated. It shows that several methods have been introduced to reduce the computations of the LSP operation by avoiding several mathematical operations. Such as the division, square root, and the pseudoinverse operations, hence executing the LSP just executed once instead of in each iteration.

Table 3. Comparison of operations involved in each method

LSP Method	Division operation	Square root	LSP executed once	Avoid pseudoinverse
Modified Cholesky decomposition + Newton-Raphson	Yes	No	No	No
Gram-Schmidt orthogonalisation + Conjugate Gradient (CG)	Yes	Yes	Yes	No
Matrix Inversion Bypass (MIB)	No	No	No	No
Modified Cholesky decomposition + Goldschmidt	Yes	No	No	No
Partitioned inversion	Yes	Yes	No	No
Cholesky decomposition + Goldschmidt	Yes	Yes	No	No
Gram-Schmidt orthogonalisation + QR decomposition	Yes	Yes	Yes	No
Gram-Schmidt orthogonalisation + Square-root-free QR decomposition	Yes	No	Yes	No
Modified Gram-Schmidt (Avoid pseudoinverse) + QR decomposition	Yes	Yes	No	Yes
Gram-Schmidt orthogonalisation + Cholesky decomposition	Yes	Yes	Yes	No

5. CONCLUSION

In the implementation of the OMP algorithm, optimisation and LSP have been given much attention as major problems and as a research gap to be improved, especially for real-time implementations. In this paper, FPGA-based implementations of OMP were reviewed and various approaches were discussed. Several comparisons in terms of FPGA performance, reconstruction time, and features for each method were done to assist readers. The combination of methods, such as Gram-Schmidt orthogonalisation combined with Cholesky decomposition, and the modification of methods to avoid certain operations, such as to avoid the square-root operation, will enhance OMP performance. However, in certain cases, the method selected still depends on the application, for example, one that requires large-scale input data. Last but not least, this paper will benefit researchers in assessing the most suitable method to be selected and improved upon.

APPENDIX

Table 1. FPGA-based OMP implementations

Reference Year	[6] 2015	[13] 2016	[14] 2017	[15] 2018	[16] 2018	[17] 2018	
Setup							
FPGA platform	Xilinx Virtex-6 XC6VLX240 T-1FF1156	Xilinx Virtex-7 XC7VX690T	Xilinx Virtex-7 XC7VX690T	Xilinx Kintex-7 XC7K325T-FBG900 HDL	Xilinx Virtex6 XC6VVSX475T-1ff1156	Xilinx Kintex UltraScale	Xilinx Virtex-5 XC5VVSX50T
Software	HLS, MATLAB Simulink + Xilinx system generator (XSG)	Verilog HDL, Xilinx ISE 14.3	Verilog HDL, Xilinx ISE 14.3	HLS, SynthesisedHDL	HLS, SynthesisedHDL	HLS, SynthesisedHDL	Verilog HDL, Xilinx ISE 14.1
Size (N, M, k)	1,024, 256, 36	32, 128, 5	512, 2,048, 12	512, configurable, configurable	1,024, 256, 36	128, 32, 5	128, 32, 5
Data format (bit)	18-bit data precision	Single precision & fixed-point (complex Data)	Single precision & fixed-point (complex Data)	Floating point: 32 bits	32-bit data precision	16 and 32-bit fixed-point (complex data)	18, 24 and 32-bit data precision
FPGA Performance							
Max frequency (MHz)	119.96	165	165	87	135.4	250	118
Registers	N/A	193,053	193,053	N/A	N/A	N/A	N/A
Occupied slices	6208	282,332	282,332	92k	7,860	N/A	3,693
DSP cores	589	1,745	1,745	93	1,544	2,032	43
Block RAM	576	573	573	128	342	307	42
Dynamic power	3,233 mW	N/A	N/A	N/A	4,370 mW	N/A	319 mW
Recovery Performance							
Reconstruction time (μ s)	340	18.3	391.8	250 for 36 iterations	170	27	7.75
Recovery signal-to-noise-ratio (RSNR)	N/A	N/A	N/A	N/A	31.04 dB	N/A	N/A
Methods							
Optimisation	K -point inner product comparator unit	Partial Fourier Basis (Fast Fourier Transform)	Partial Fourier Basis (Fast Fourier Transform)	Matrix Inversion Bypass (MIB)	Parallel correlation indices (PIC)	Multiple measurement vectors (MMV)	Parallel Hardware Optimisation
LSP	Modified Cholesky decomposition + Newton-Raphson	Gram-Schmidt orthogonalisation + Conjugate Gradient (CG)	Gram-Schmidt orthogonalisation + Conjugate Gradient (CG)	Matrix Inversion Bypass (MIB)	Modified Cholesky decomposition + Goldschmidt	Partitioned inversion	Cholesky decomposition + Goldschmidt

Table 1. FPGA-based OMP implementations (cont.)

Reference Year	[18] 2019	[19] 2019	[20] 2020	[21] 2021
Setup				
FPGA platform	Xilinx Virtex-6 xc6vlx240t	Xilinx Artix-7 XC&A100t	Xilinx Kintex-7 FPGA	Xilinx Virtex-6 xc6vsx240t
Software	HDL	HDL	HDL	HDL
Size (N, M, k)	1024, 256, 36	1024, 80, 16	1024, 256, 36	1024, 256, 64
Data format (bit)	18-bit data precision	18-bit data precision	18-bit data precision	10-bit data precision
FPGA Performance				
Max frequency (MHz)	133.33	210	210	131.5
Registers	N/A	N/A	N/A	N/A
Occupied slices	10,902	4,828	28,443	8,525
DSP cores	386	152	523	410
Block RAM	430	166	386	431
Dynamic power (mW)	2357	609	3,233	1,819
Recovery Performance				
Reconstruction time (μ s)	327	139.32	238	76.43
Recovery signal-to-noise-ratio (RSNR)	N/A	N/A	28.55 dB	N/A

Table 1. FPGA-based OMP implementations (*cont.*)

Reference	[18]	[19]	[20]	[20]	[21]	[21]	
Year	2019	2019	2019	2020	2020	2021	
Methods							
Optimisation	Single matrix multiplication unit	Single matrix multiplication unit	Parallel Hardware Optimisation	Multiple column search correlation	Multiple column search correlation	Parallel Hardware Optimisation	Parallel Hardware Optimisation
LSP	Gram-Schmidt orthogonalisation + QRD	Gram-Schmidt orthogonalisation + QRD	Gram-Schmidt orthogonalisation + Square-root-free QRD	Modified Gram-Schmidt (Avoid pseudo-inverse) + QRD	Modified Gram-Schmidt (Avoid pseudo-inverse) + QRD	Gram-Schmidt orthogonalisation + Cholesky decomposition	Gram-Schmidt orthogonalisation + Cholesky decomposition

ACKNOWLEDGEMENTS

This research was supported by Universiti Tun Hussein Onn Malaysia (UTHM) through GPPS (Vot H534). Communication of this research is made possible through monetary assistance by Universiti Tun Hussein Onn Malaysia and the UTHM Publisher's Office via Publication Fund E15216.

REFERENCES




- [1] M. Fardad, S. M. Sayedi, and E. Yazdian, "A low-complexity hardware for deterministic compressive sensing reconstruction," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 65, no. 10, pp. 3349-3361, Oct. 2018, doi: 10.1109/TCSI.2018.2803627.
- [2] H. Mamaghanian, N. Khaled, D. Atienza, and P. Vanderghenst, "Design and exploration of low-power analog to information conversion based on compressed sensing," *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 2, no. 3, pp. 493-501, Sep. 2012, doi: 10.1109/JETCAS.2012.2220253.
- [3] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489-509, Feb. 2006, doi: 10.1109/TIT.2005.862083.
- [4] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289-1306, Apr. 2006, doi: 10.1109/TIT.2006.871582.
- [5] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5406-5425, Dec. 2006, doi: 10.1109/TIT.2006.885507.
- [6] H. Rabah, A. Amira, B. K. Mohanty, S. Almaadeed, and P. K. Meher, "FPGA implementation of orthogonal matching pursuit for compressive sensing reconstruction," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 23, no. 10, pp. 2209-2220, Oct. 2015, doi: 10.1109/TVLSI.2014.2358716.
- [7] T. R. Braun, "An evaluation of GPU acceleration for sparse reconstruction," *Signal Process. Sens. Fusion, Target Recognit. XIX*, vol. 7697, 2010, doi: 10.1117/12.849536.
- [8] S. Liu, N. Lyu, and H. Wang, "The implementation of the improved OMP for AIC reconstruction based on parallel index selection," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 26, no. 2, pp. 319-328, Feb. 2018, doi: 10.1109/TVLSI.2017.2765677.
- [9] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655-4666, Dec. 2007, doi: 10.1109/TIT.2007.909108.
- [10] V. Jagannatha, G. Jyothi, and M. Z. Kurian, "FPGA implementation of IEEE 802.15.3c Transceiver," *Int. J. Adv. Res. Electr. Electron. Instrum. Eng.*, vol. 2, no. 6, pp. 2448-2451, Jun. 2013.
- [11] E. Candès and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse Probl.*, vol. 23, no. 3, pp. 969-985, Apr. 2007, doi: 10.1088/0266-5611/23/3/008.
- [12] E. J. Candès, "The restricted isometry property and its implications for compressed sensing," *Comptes Rendus Math.*, vol. 346, no. 9-10, pp. 589-592, Apr. 2008, doi: 10.1016/j.crma.2008.03.014.
- [13] Y. Quan, Y. Li, X. Gao, and M. Xing, "FPGA implementation of real-time compressive sensing with partial fourier dictionary," *Int. J. Antennas Propag.*, vol. 2016, pp. 1-12, 2016, doi: 10.1155/2016/1671687.
- [14] G. Huang and L. Wang, "An FPGA-based architecture for high-speed compressed signal reconstruction," *ACM Trans. Embed. Comput. Syst.*, vol. 16, no. 3, 2017, doi: 10.1145/3056481.
- [15] S. Liu, N. Lyu, and H. Wang, "The implementation of the improved OMP for AIC reconstruction based on parallel index selection," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 26, no. 2, pp. 319-328, Feb. 2018, doi: 10.1109/TVLSI.2017.2765677.
- [16] S. Kim *et al.*, "Reduced computational complexity orthogonal matching pursuit using a novel partitioned inversion technique for compressive sensing," *Electronics*, vol. 7, no. 9, pp. 1-10, Sep. 2018, doi: 10.3390/electronics7090206.
- [17] Ö. Polat and S. K. Kayhan, "High-speed FPGA implementation of orthogonal matching pursuit for compressive sensing signal reconstruction," *Comput. Electr. Eng.*, vol. 71, no. July, pp. 173-190, Oct. 2018, doi: 10.1016/j.compeleceng.2018.07.017.
- [18] S. Roy, D. P. Acharya, and A. K. Sahoo, "Low-Complexity Architecture of Orthogonal Matching Pursuit Based on QR Decomposition," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 27, no. 7, pp. 1623-1632, Jul. 2019, doi: 10.1109/TVLSI.2019.2909754.
- [19] X. Ge, F. Yang, H. Zhu, X. Zeng, and D. Zhou, "An efficient FPGA implementation of orthogonal matching pursuit with square-root-free QR decomposition," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 27, no. 3, pp. 611-623, Mar. 2019, doi: 10.1109/TVLSI.2018.2879884.
- [20] S. Roy, D. P. Acharya, and A. K. Sahoo, "Fast OMP algorithm and its FPGA implementation for compressed sensing-based sparse signal acquisition systems," *IET Circuits, Devices Syst.*, vol. 15, no. 6, pp. 511-521, Sep. 2021, doi: 10.1049/cds2.12047.
- [21] J. Li, P. Chow, Y. Peng, and T. Jiang, "FPGA implementation of an improved OMP for compressive sensing reconstruction," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 29, no. 2, pp. 259-272, Feb. 2021, doi: 10.1109/TVLSI.2020.3030906.

- [22] Z. Wang, S. Huang, S. Wang, S. Zhuang, Q. Wang, and W. Zhao, "Compressed sensing method for health monitoring of pipelines based on guided wave inspection," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 7, pp. 4722-4731, Jul. 2020, doi: 10.1109/TIM.2019.2951891.
- [23] M. Rani, S. B. Dhok, and R. B. Deshmukh, "A systematic review of compressive sensing: concepts, implementations and applications," *IEEE Access*, vol. 6, pp. 4875-4894, Jan. 2018, doi: 10.1109/ACCESS.2018.2793851.
- [24] T. Arildsen, C. S. Oxvig, P. S. Pedersen, J. Ostergaard, and T. Larsen, "Reconstruction algorithms in undersampled AFM imaging," *IEEE J. Sel. Top. Signal Process.*, vol. 10, no. 1, pp. 31-46, Feb. 2016, doi: 10.1109/JSTSP.2015.2500363.
- [25] S. Vijay Kartik, R. E. Carrillo, J.-P. Thiran, and Y. Wiaux, "A Fourier dimensionality reduction model for big data interferometric imaging," *Mon. Not. R. Astron. Soc.*, vol. 468, no. 2, pp. 2382-2400, Mar. 2017, doi: 10.1093/mnras/stx531.
- [26] R. Palmeri, M. T. Bevacqua, L. Di Donato, L. Crocco, and T. Isernia, "Microwave imaging of non-weak targets in stratified media via virtual experiments and compressive sensing," in *2017 11th European Conference on Antennas and Propagation (EUCAP)*, Mar. 2015, vol. 14, no. 1, pp. 1035-1038, doi: 10.23919/EuCAP.2017.7928490.
- [27] A. Amir and O. Zuk, "Bacterial community reconstruction using compressed sensing," *J. Comput. Biol.*, vol. 18, no. 11, pp. 1723-1741, Nov. 2011, doi: 10.1089/cmb.2011.0189.
- [28] K.-K. Poh and P. Marziliano, "Compressive sampling of EEG signals with finite rate of innovation," *EURASIP J. Adv. Signal Process.*, vol. 183105, pp. 1-12, Mar. 2010, doi: 10.1155/2010/183105.
- [29] A. K. Bhateja, S. Sharma, S. Chaudhury, and N. Agrawal, "Iris recognition based on sparse representation and k-nearest subspace with genetic algorithm," *Pattern Recognit. Lett.*, vol. 73, pp. 13-18, Apr. 2016, doi: 10.1016/j.patrec.2015.12.009.
- [30] S. Sivapalan, R. K. Rana, D. Chen, S. Sridharan, S. Denmon, and C. Fookes, "Compressive sensing for gait recognition," in *2011 International Conference on Digital Image Computing: Techniques and Applications*, Dec. 2011, pp. 567-571, doi: 10.1109/DICTA.2011.101.
- [31] Z. Lei, K. Yang, R. Duan, and P. Xiao, "Localization of low-frequency coherent sound sources with compressive beamforming-based passive synthetic aperture," *J. Acoust. Soc. Am.*, vol. 137, no. 4, pp. EL255-EL260, Apr. 2015, doi: 10.1121/1.4915003.
- [32] P. Harris, R. Philip, S. Robinson, and L. Wang, "Monitoring anthropogenic ocean sound from shipping using an acoustic sensor network and a compressive sensing approach," *Sensors*, vol. 16, no. 3, p. 415, Mar. 2016, doi: 10.3390/s16030415.
- [33] H. You, Z. Ma, W. Li, and J. Zhu, "A speech enhancement method based on multi-task bayesian compressive sensing," *IEICE Trans. Inf. Syst.*, vol. E100.D, no. 3, pp. 556-563, Mar. 2017, doi: 10.1587/transinf.2016EDP7350.
- [34] A. Omara, A. Hefnawy, and A. Zekry, "On sparse compression complexity of speech signals," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 1, no. 2, pp. 329-341, Feb. 2016, doi: 10.11591/ijeecs.v1.i2.pp329-340.
- [35] M. P. Edgar, M.-J. Sun, G. M. Gibson, G. C. Spalding, D. B. Phillips, and M. J. Padgett, "Real-time 3D video utilizing a compressed sensing time-of-flight single-pixel camera," *Opt. Trapp. Opt. Micromanipulation XIII*, vol. 9922, pp. 1-8, Sep. 2016, doi: 10.1117/12.2239113.
- [36] A. Veeraraghavan, D. Reddy, and R. Raskar, "Coded strobing photography: compressive sensing of high speed periodic videos," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 4, pp. 671-686, Apr. 2011, doi: 10.1109/TPAMI.2010.87.
- [37] C. Liao, J. Tao, X. Zeng, Y. Su, D. Zhou, and X. Li, "Efficient spatial variation modeling of nanoscale integrated circuits via hidden markov tree," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 35, no. 6, pp. 971-984, Jun. 2016, doi: 10.1109/TCAD.2015.2481868.
- [38] H. Huang, H. Yu, C. Zhuo, and F. Ren, "A compressive-sensing based testing vehicle for 3D TSV pre-bond and post-bond testing data," in *Proceedings of the 2016 on International Symposium on Physical Design*, Apr. 2016, pp. 19-25, doi: 10.1145/2872334.2872351.
- [39] S. K. Sharma, E. Lagunas, S. Chatzinotas, and B. Ottersten, "Application of compressive sensing in cognitive radio communications: a survey," *IEEE Commun. Surv. Tutorials*, vol. 18, no. 3, pp. 1838-1860, 2016, doi: 10.1109/COMST.2016.2524443.
- [40] Y. Liu, H. Ruan, L. Wang, and A. Nehorai, "The random frequency diverse array: a new antenna structure for uncoupled direction-range indication in active sensing," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 2, pp. 295-308, Mar. 2017, doi: 10.1109/JSTSP.2016.2627183.
- [41] F. Aderohunmu, D. Brunelli, J. Deng, and M. Purvis, "A data acquisition protocol for a reactive wireless sensor network monitoring application," *Sensors*, vol. 15, no. 5, pp. 10221-10254, Apr. 2015, doi: 10.3390/s150510221.
- [42] W. Wang, S. Wang, J. Yang, and H. Liu, "Under-sampling of PPM-UWB communication signals based on CS and AIC," *Circuits, Syst. Signal Process.*, vol. 34, no. 11, pp. 3595-3609, Nov. 2015, doi: 10.1007/s00034-015-0026-4.
- [43] E. Ashraf, A. A. M. Khalaf, and S. M. Hassan, "Real time FPGA implementation of SAR radar reconstruction system based on adaptive OMP compressive sensing," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 20, no. 1, p. 185, Oct. 2020, doi: 10.11591/ijeecs.v20.i1.pp185-196.
- [44] A. Wang, F. Lin, Z. Jin, and W. Xu, "A configurable energy-efficient compressed sensing architecture with its application on body sensor networks," *IEEE Trans. Ind. Informatics*, vol. 12, no. 1, pp. 15-27, Feb. 2016, doi: 10.1109/TII.2015.2482946.
- [45] R. W. Heath, N. Gonzalez-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave MIMO systems," *IEEE J. Sel. Top. Signal Process.*, vol. 10, no. 3, pp. 436-453, Apr. 2016, doi: 10.1109/JSTSP.2016.2523924.
- [46] J.-C. Shen, J. Zhang, E. Alsusa, and K. B. Letaief, "Compressed CSI acquisition in FDD massive MIMO: how much training is needed?," *IEEE Trans. Wirel. Commun.*, vol. 15, no. 6, pp. 4145-4156, Jun. 2016, doi: 10.1109/TWC.2016.2535310.
- [47] M. E. Eltayeb, T. Y. Al-Naffouri, and H. R. Bahrami, "Compressive sensing for feedback reduction in MIMO broadcast channels," *IEEE Trans. Commun.*, vol. 62, no. 9, pp. 3209-3222, Sep. 2014, doi: 10.1109/TCOMM.2014.2347964.
- [48] A. Septimus and R. Steinberg, "Compressive sampling hardware reconstruction," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, May 2010, pp. 3316-3319, doi: 10.1109/ISCAS.2010.5537976.
- [49] L. Bai, P. Maechler, M. Muehlberghuber, and 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)*, Dec. 2012, no. 1, pp. 53-56, doi: 10.1109/ICECS.2012.6463559.
- [50] P. Blache, H. Rabah, and A. Amira, "High level prototyping and FPGA implementation of the orthogonal matching pursuit algorithm," in *2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, Jul. 2012, pp. 1336-1340, doi: 10.1109/ISSPA.2012.6310501.
- [51] H. Zhu, W. Chen, and Y. Wu, "Efficient implementations for orthogonal matching pursuit," *Electron.*, vol. 9, no. 9, pp. 1-23, 2020, doi: 10.3390/electronics9091507.
- [52] G. H. Golub and L. C. Van, *Matrix Computation*, 4th editio. The Johns Hopkins University Press Baltimore, 2013.

BIOGRAPHIES OF AUTHORS

Muhammad Muzakkir Mohd Nadzri    received his B.Eng. and M.Eng. in Electronic and Electrical Engineering from Universiti Tun Hussein Onn Malaysia (UTHM) in 2015 and 2018, respectively. He has worked as a research assistant at Reconfigurable Computing for Analytics Acceleration (ReCAA) Research Laboratory, Microelectronics and Nanotechnology-Shamsuddin Research Centre (MiNTSRC), UTHM. Currently, he is a PhD candidate in the field Electrical Engineering at UTHM with his dissertation titled “Efficient Reconfigurable Architectures of Compressive Sensing for Wireless Guided Wave Pipelines Inspection”. His research interests include pipeline inspection, compressive sensing, and reconfigurable computing. He can be contacted at email: muzakkirnadzri@gmail.com.



Afandi Ahmad    is an associate professor under the Faculty of Electrical and Electronic Engineering as well as the head of Reconfigurable Computing for Analytics Acceleration (ReCAA) Research Laboratory, Microelectronics and Nanotechnology-Shamsuddin Research Centre (MiNTSRC), UTHM. He received his Ph.D. in Electronic and Computer Engineering at Brunel University, London, UK, in 2010. He has been awarded a number of grants and has published papers in various journals during his career to date. He is a member of IEEE and IET. His research interests include embedded systems, reconfigurable computing, image processing, and medical applications. He can be contacted at email: afandia@uthm.edu.my.