HDL Implementation of OMP Based Compressed Sampled Reconstruction Algorithm

A thesis submitted in partial fulfilment of the requirement for

M.Tech Dual Degree

In

Electronics and Communication Engineering

(Specialization: VLSI Design and Embedded Systems)

SUBMITTED BY

Bibekananda Jena
710EC2045

Under the Guidance of:
Prof. S. Deshmukh
Assistant Professor
Department of Electronics and Communication
NIT Rourkela

Department of Electronics and Communication Engineering

National Institute of Technology, Rourkela

HDL Implementation of OMP Based Compressed Sampled Reconstruction Algorithm

A thesis submitted in partial fulfilment of the requirement for

M.Tech Dual Degree

In

Electronics and Communication Engineering

(Specialization: VLSI Design and Embedded Systems)

SUBMITTED BY

Bibekananda Jena
710EC2045

Under the Guidance of:
Prof. S. Deshmukh
Assistant Professor
Department of Electronics and Communication
NIT Rourkela

Department of Electronics and Communication Engineering

National Institute of Technology, Rourkela

DEPT. OF ELECRTONICS AND COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY

ROURKELA, ODISHA -769008

CERTIFICATE

This is to certify that the work presented in the thesis entitled "**HDL Implementation of OMP Based Compressed Sampled Reconstruction Algorithm"** by **Bibekananda Jena** is a bonafide record of the original research work carried out by him at National Institute of Technology, Rourkela under my supervision and guidance during 2014-2015 in partial fulfilment for the award of Dual Degree in Electronics and Instrumentation Engineering (Communication and Signal Processing), National Institute of Technology, Rourkela.

Place: NIT Rourkela

Date:

Prof. S. Deshmukh

Assistant Professor

Dept. of Electronics and Communication

NIT Rourkela

DEPT. OF ELECRTONICS AND COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY

ROURKELA, ODISHA -769008

DECLARATION

I hereby declare that the work presented in the thesis titled "**HDL Implementation of OMP Based Compressed sampled Reconstruction Algorithm**" being submitted in partial fulfilment for the degree of Master of Technology is a bonafide record of the research work done by me under the supervision of **Prof. Siddharth Deshmukh**, Dept. of Electronics and Communication Engineering, National Institute of Technology, Rourkela, India and that no part of this work has been presented for I also declare that due credit has been given to the information presented from other sources wherever used in this work through citations with details in the Reference section.

Bibekananda Jena

710EC2045

DEPT. OF ELECRTONICS AND COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY

ROURKELA, ODISHA -769008

ACKNOWLEDGEMENT

The research work has been partly made possible due to the continuous support motivation of lot of people from every aspect of my life. I sincerely extend my heartfelt gratitude to my project guide **Prof. Siddharth Deshmukh** for suggesting me the research topic and providing his guidance and supervision throughout the period of research work. I would also like to thank all the faculty members of the Department of Electronics and Communication Engineering, NIT Rourkela for their valuable help. I extend my gratitude and sincere thanks to the fellow students, research scholars and Lab Assistant at the Mobile Communication Lab, Dept. of ECE, NIT Rourkela. Finally, I would also like to thank my family and friends for their support and help.

Bibekananda Jena

710EC2045

INDEX

Abstract

Nearly all signal acquisition techniques follow the much celebrated Shannon's sampling theorem which specifies that the sampling rate of the signal must be at least two times the highest frequency present in the signal. The sampled data is then compressed to make it efficient for storage and transmission. Conventional approach to sampling is expensive in terms of data storage and transmission due to the large number of samples generated. Some cases increasing sampling rate is also very expensive like high speed ADCs, imaging systems, etc. It is also inefficient since lot of the data produced is redundant in nature since most naturally occurring signals are sparse in nature.

Compressive sensing addresses these inefficiencies by directly acquiring a compressed signal representation without going through the intermediate stage of acquiring all samples. The sampled data can be reconstructed using computationally intensive algorithm. CS is also superior to conventional approaches in the following regard that the CS performs the time consuming processes at the recovery end rather than the sensing end.

Reconstruction algorithms are complex and implementation of these algorithms in software is extremely slow and power consuming due to the reason that it is based on several layer of abstraction and shared resources between multiple processes. On the other hand hardware implementation takes advantage of hardware parallelism, custom datapath creation ability and dedicated hardware for each task. The hardware implementation in the project will be utilizing the OMP algorithm due to its less complexity and faster solution time. The algorithm will be implemented using VHDL.

The objective of the project will be to implement the OMP algorithm using optimal resources so as to reduce the reconstruction time without compromising with accuracy intended.

List of Figures

CHAPTER 1

INTRODUCTION

1. Introduction

Traditional sampling follows the celebrated Shannon's Sampling Theorem which states that the sampling rate must be at least twice the highest frequency content present in the signal for a faithful reconstruction of the signal. This approach works well for signal with low frequency and signals that are not sparse. When a signal with higher frequency content is encountered, the sampling produces a large amount of data, whose handling and storage becomes expensive and inefficient since higher frequency content requires higher sampling rate which in turn is expensive.

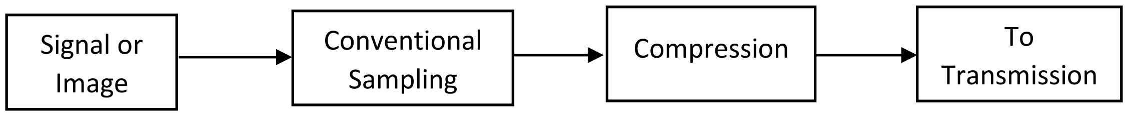

Figure 1 Conventional Sampling Approach

Another biggest drawback of the conventional sampling approach is that the compression stage follows the sampling stage. Even if the signal is heavily sparse sampling has to be done before any compression can occur. This method results in large amount of redundant data being generated after sampling process which is then discarded during the compression process. Compressive sensing eliminates this drawback by incorporating sensing and compression in one stage and thus representing the original signal with less elements than suggested by its bandwidth.

1.1. Compressive Sensing

Compressive Sensing or Compressed sampling is a novel method of sampling a signal below the stated Nyquist rate with the prior knowledge that the signal is sparse in the domain it is represented, if not then transformed to be represented in the domain in which it is sparse. The sampled signal is then reconstructed using optimization algorithm to produce an approximate replica of the original signal. The accuracy of the reproduced signal depends upon the accuracy requirement, computation time limit and the algorithm employed.

Compressed Sensing is essentially to project a signal linearly to a series of measurements that consists of fewer elements than original signal. It is a method to obtain a unique solution from an underdetermined linear system taking advantage of the prior knowledge that the true solution is sparse. [1]

Compressive sensing incorporates both compression and sensing in one stage thus eliminating the more time intensive compression stage which is usually present in the conventional sampling techniques to save data. The sensing and compression occur together in a single process as opposed to the conventional techniques in which compression process follows the sampling stage.

Let $x \in \mathbb{R}$ be a one dimensional signal and $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$ be an orthonormal transform matrix, where $\mathbb{R}$ is the set of real numbers. This matrix $\boldsymbol{\Psi}$ is called representation basis. If $x = \boldsymbol{\Psi} z$ and there are only $s \ll n$ nonzero entries in z, it is said that that x is s-sparse in $\boldsymbol{\Psi}$ domain. We sample x by $\boldsymbol{\Phi} \in \mathbb{R}^{m \times n}$, where $s < m < n$ to get [1]

$$y = \boldsymbol{\Phi} x = \boldsymbol{A} z \in \mathbb{R}^{m}$$

Where,

y = Sensed data

$\boldsymbol{A} = \boldsymbol{\Phi\Psi}$, also known as the Sensing Matrix

If $\boldsymbol{\Phi}$ obeys the order s-restricted isometry property (RIP) and has low coherence with $\boldsymbol{\Psi}$, then z (and in turn x) can be effectively reconstructed.

The crucial observation is that one can design efficient sensing or sampling protocols that capture the useful information content embedded in a sparse signal and condense it into a small amount of data. These protocols are non-adaptive and simply require correlating the signal with a small number of fixed waveforms that are incoherent with the sparsifying basis. The most remarkable fact about these sampling protocols is that they allow a sensor to very efficiently capture the information in a sparse signal without trying to comprehend that signal. The full-length signal is reconstructed from the small amount of collected data by using numerical optimization. In other words, CS is a very simple and efficient signal acquisition protocol which samples—in a signal independent fashion—at a low rate and later uses computational power for reconstruction from what appears to be an incomplete set of measurements. [2]

1.1.1. Sparsity

Sparsity refers to the sparseness of a signal. Sparseness is given by the number of nonzero components of a signal. A signal with s nonzero components is referred as s-sparse signal. It can also be said that the signal has a sparsity of s. Sparsity represents the idea that the information contained in the signal is far less than suggested by its bandwidth.

Sparsity is an essential requirement for compressive sensing. Even if the signal is not sparse in its original domain it can be transformed into a suitable domain where it is sparse in nature. This transformation occurs by projection of original signal onto the new domain represented by the representation basis. Mathematically speaking it occurs by multiplying the original signal vector with the representation basis.

1.1.2. Incoherence

Incoherence extends the duality between time and frequency and expresses the idea that objects having a sparse representation in the basis must be spread out in the domain in which they are acquired. This implies that the sensing basis and the representation basis must be incoherent.

Coherence measures the largest correlation between any two elements of the dictionary Φ and representation basis $\mathbf{\Psi}$. Coherence is denoted by $\mu(\mathbf{\Phi}, \mathbf{\Psi})$.A measure of coherence is given by [3]:

$$\mu(\boldsymbol{\phi}, \boldsymbol{\psi}) = \sqrt{n} \cdot \max_{1 \leq k,j \leq n} |< \boldsymbol{\phi}_k, \boldsymbol{\psi}_j >|$$

Where,

$\mu(\boldsymbol{\Phi}, \boldsymbol{\Psi})$ is the coherence between Φ and Ψ,

n is the original vector dimension

$\boldsymbol{\Phi}$ is the dictionary

$\boldsymbol{\Psi}$ is the representation basis.

The maximum and minimum bounds for the coherence $\mu(\boldsymbol{\Phi}, \boldsymbol{\Psi})$ is found to be $[1, \sqrt{n}]$ [3]. Compressive sensing mainly concerns itself with low coherence pairs, which means high incoherency. High incoherency between $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ means that the samples add new information that is not already represented by the known basis $\boldsymbol{\Psi}$.

1.1.3. Representation Basis

Compressive sensing relies on the fact that the signal to be sampled is sparse in nature. This doesn't hold true for all sorts of signals. For example signal such as sinc function is spread out all over the time domain. But the Fourier transform of $sinc$ function is $rect$ function which is highly sparse considering the fact that the signal's maximum energy component is present in the rectangular portion and rest of the portion is zero.

Signals such as image in spatial domain is not sparse in nature, thus not useful for doing compressive sensing in spatial domain. As stated in the Incoherency section, there exists a representation for every signal in which it is sparse in nature. Image, in particular are found to be spare in the wavelet domain.

1.1.4. Dictionary

Dictionary, Φ is the matrix onto which the transformed sparse signal is projected into. This matrix causes the dimension reduction in sampled data acquired. Dictionary needs to be largely incoherent with the sparsifying representation basis Ψ.

Fortunately it turns out that random matrices are largely incoherent with any fixed sparsifying basis Ψ. This has lead CS to strongly rely on random sensing matrices, since they provide universally incoherent sensing-sparsifying pairs and are well conditioned for reconstruction. However to obtain better results measurement matrix can be optimized to have lesser coherence.

1.1.5. Sensing Matrix

Sensing matrix or the measurement matrix is the matrix onto which the original data vector is projected. This matrix denoted by A is product of multiplication of dictionary Φ and representation basis Ψ. The measurement matrix doesn't depend upon the data to be measured rather it is dependent on the fact that it should satisfy the incoherence criteria and the restricted isometric property.

1.2. Reconstruction

The reconstruction of the compressed sampled signal involves solution of an underdetermined system of linear equations. The basic sensing equation is given by

$$\mathrm{y} = \mathbf{A}\mathrm{x}$$

Where,

y is the measured values

A is the sensing matrix

x is the original signal

Reconstruction requires the solution of equation for x. Direct solution of the equation never yields a unique solution even with the prior knowledge of the sparseness of the solution. The inverse equation is given by [4]

$$\hat{\mathrm{x}} = \{\mathbf{A}^{\mathrm{T}}\mathbf{A}\}^{-1}\mathbf{A}^{\mathrm{T}}\mathrm{y}$$

The signal reconstruction process is essentially to choose the best estimate of the original signal from all the possible solutions obtained from the above inverse equation. This is achieved by the convex optimization algorithm. The success and accuracy of the reconstructed signal is heavily dependent on the sparsity of original signal, sensing matrix and the optimization algorithm used.

The original signal is recovered by solving the following Convex Optimization problem:

$$\hat{x} = \arg\min\left|\left|x\right|\right|_1 \qquad \text{subject to } \mathrm{y} = \mathbf{A}\mathrm{x}$$

The minimum number of measurements, m required to faithfully reconstruct a compressed sampled signal is given by [2]

$$m \geq C.\mu(\mathbf{\Phi}, \mathbf{\Psi}).s.\log n$$

Where,

m is the number of measurements

C is some positive constant

$\mu(\Phi, \Psi)$ is the coherence measure between Φ and Ψ

s is the sparsity of the signal

n is the vector length of the original signal

Optimization algorithm which are used most widely are l_1 optimization algorithm and Orthogonal Matching Pursuit algorithm.

1.2.1. l_1-optimization Algorithm

l_1-optimization or l_1-norm optimization algorithm is an linear programming problem in the framework of convex optimization. The problem is stated as [2]

$$\min_x ||x||_1 \quad \text{subject to } \boldsymbol{A}x = y$$

The 1 subscript in $||x||_1$ denotes that it is norm 1 optimization algorithm. Replacing 1 by other whole number, n changes the problem into norm n optimization problem.

It is more complex in nature and time consuming. But the benefit of using this algorithm resides in the fact that the results from utilizing the said algorithm produces better result than any other algorithms present. Since it is time consuming, it becomes non useful in time critical reconstruction applications thus restricting its usage.

1.2.2. OMP

Orthogonal matching pursuit is a greedy iterative algorithm for approximatively solving the original l_0 pseudo-norm problem OMP constructs an approximation by going through an iteration process. At each iteration the locally optimum solution is calculated. OMP works by finding a basis vector in A that maximizes the correlation with the residual (initialized to y), and then recomputing the residual and coefficients by projecting the residual on all atoms in the dictionary using existing coefficients. OMP has the added advantage that the atom (each column vector of the sensing matrix is called atom) picked up once won’t be picked up in the next iteration. This is achieved by maintaining an atom index which is updated on each iteration. As, a result OMP produces an estimate of the sparse signal in m iterations.

OMP is the mathematical optimization problem of the form: [5]

$$\min_x ||x||_0 \text{ subject to } y = \boldsymbol{A}x$$

Where,

x is a $N \times 1$ solution vector (signal),
y is a $M \times 1$ vector of observations (measurements),
A is a $M \times N$ sensing matrix (or measurement matrix)
and $M < N$.

The biggest advantage of this algorithm is that it is much faster than the l_1 optimization algorithm but the accuracy of the algorithm is less than the optimization algorithm. Thus the project will utilize the OMP algorithm during the reconstruction phase. The algorithm also has an added benefit of being easier to implement than the l_1 optimization algorithm.

CHAPTER 2

COMPRESSIVE SENSING SYSTEM : ALGORITHMS

2. Compressive Sensing System : Algorithms

2.1. Sensing

Given a s-sparse signal of dimension N×1, a measurement matrix A of dimension M×N (M is the no of measurements). The sensed data Y (Samples) is given by

$$\mathrm{Y} = \mathbf{A} \times \mathrm{X}$$

The data Y is a projection of X on sensing matrix A. s, M and N follow the following inequality

$$s < M < N$$

The measurement matrix does not depend on the data which is to be sensed. For compressive sensing to work, the data must be sparse and the measurement matrix incoherent. Signal may not be sparse in their usual form but they in transform domain they can be represented as sparse. Incoherent matrix can be generated randomly. Alternatively measurement matrix can be optimised to decrease the coherence further. In this project we have used Gradient Descent Algorithm to increase incoherence between the representation basis and measurement matrix. The algorithm is explained below:

Optimisation of Measurement Matrix: To minimize the gradient using Gram Matrix we have used Gradient Descent Algorithm. Pseudo code for the same is given below

Algorithm 1

Input: Sparse representation basis $\boldsymbol{\Psi_{nn}}$, Step size 'step', Maximum number of iterations s, Identity matrix $\mathbf{I_{nn}}$

Output: Measurement matrix $\boldsymbol{\Phi}$

```
begin
    Initialize D to a M × N random matrix
    for k = 1 to s do
        for j = 1 to n do
            $dj = \frac{dj}{\|dj\|_2}$
        end
```

```
            $\mathbf{D} = \mathbf{D} - \text{step} \times \mathbf{D}(\mathbf{D}^{T}\mathbf{D} - \mathbf{I})$
        end
        $\boldsymbol{\phi} = \mathbf{D}\boldsymbol{\psi}^{-1}$
    end
```

The algorithm starts with initialisation of matrix D by an M×N random matrix which is iteratively updated. The columns of this matrix are normalized at each iteration. The value of step size determines how fast the values are reduced.

The main aim of this project is optimal reconstruction algorithm. Therefore sensing part is done using MATLAB. Reconstruction algorithm however is implemented using VHDL which take inputs from MATLAB.

2.2. Reconstruction

The algorithm utilized in this project for Compressed Sampling Reconstruction is given by [4]:

Algorithm 2:

a. Initialize the residual $\mathrm{R} = \mathrm{y}$, the index set $\tilde{\boldsymbol{\Phi}} = \emptyset$ and the iteration counter $t = 1$

b. Find the index λ_{t} which is most correlated to Φ by solving the optimization problem

$$\lambda_{\mathrm{t}} = \arg\max_{\mathrm{j}=1\ldots\mathrm{N}} \left| < \mathbf{R}_{\mathrm{t}-1}, \boldsymbol{\Phi}_{\mathrm{j}} > \right|$$

c. Update the index set Λ_t and column set $\tilde{\Phi}$

$$\Lambda_{\mathrm{t}} = \Lambda_{\mathrm{t}-1} \cup \{\lambda_{\mathrm{t}}\}$$

$$\tilde{\boldsymbol{\Phi}} = [\tilde{\boldsymbol{\Phi}} \ \ \Phi_{\lambda_t}]$$

d. Calculate the new residual according to

$$\mathrm{R}_{\mathrm{t}} = \mathrm{R}_{\mathrm{t}-1} - \left(\tilde{\boldsymbol{\Phi}}_{\mathrm{t}} \, \tilde{\boldsymbol{\Phi}}_{\mathrm{t}}'\right) . \mathrm{R}_{\mathrm{t}-1}$$

e. Increment t and return to step b if t is less than m

f. Solve the least square problem to find $\hat{x}$ for the indices in Λ

$$\hat{x} = \arg\min_{x} \left\| \tilde{\boldsymbol{\Phi}} \mathrm{x} - \mathrm{y} \right\|$$

2.2.1. Orthogonal matching Pursuit

Orthogonal matching pursuit (OMP) constructs an approximation by going through an iteration process. At each iteration the locally optimum solution is calculated. To begin with a residue is initialized with the vector which is required to be approximates i.e. r = Y. For each iteration a column vector in A is chosen which most closely resembles a residual vector r. The column number is stored in an index set and the entire column in a column set. This column set is solved using least square problem to find an approximated data x. The residue is than updated by subtracting the correlation from it for next iteration. It is repeated K times to find K non-zero data.

```
Algorithm 2

Input: Measurement matrix A, sensed data Y

Output: Approximation of sparse signal X ----- x

begin

    Initialize: residue r = y; index set i = ø; Column set C = ø;

    Iteration counter p = 0

    for (p = 0; p <= K; p = p + 1 )

        begin

            i=arg max |< r ,Aj >|

            Update column set C = [C] ∪ Aj

            Solve least square problem using QR decomposition

            $r_p = r_{p-1} - (C \cdot C')r_{p-1}$.

        end

end
```

2.2.2. Least Square Problem

The linear least square problem is given by

Minimize x : $\left\|y - \mathbf{\Phi}x\right\|_2$

Since we find K maximum correlated column of the original measurement matrix, A is therefore a K×N matrix. x is the n element solution vector. From the theory of CS we know that m < n. Thus the no of equations is less than the no of variables and the system is said to be under-determined. There exists infinite solution for this system. The solution which minimizes the equation is called the least square solution. Since the minimum value of this equation is zero, the equation can be rewritten as

$$x = A^{-1}y$$

Thus we need to find inverse of matrix A. We used QR decomposition for the same which's algorithm is explained in the next topic. The measurement matrix A can be factorized as

$$A = QR$$

Where Q is an orthogonal matrix i.e. $Q^TQ = I$ and R is a right triangular matrix. The problem can now be solved as: [4]

$$A\hat{x} = y$$

$$\{A^TA\}\hat{x} = A^Ty$$

Let $C = \{A^TA\}$

$$C\hat{x} = A^Ty$$

$$\hat{x} = C^{-1}A^Ty$$

$$\hat{x} = \{QR\}^{-1}A^Ty$$

$$\hat{x} = R^{-1}Q^{-1}A^Ty$$

$$y = R^{-1}Q^TA^Tx$$

QR Decomposition Algorithm: We used modified Gram–Schmidt algorithm for QR decomposition. Pseudo code is given below:

Algorithm 3

Input: Measurement matrix A of dimension $S \times S$

Output: Matrix Q and R

```
begin
    for k = 1:S
```

```
                    $Q_k = \frac{A_k}{||A_k||_2}$
          for j = 1:k
                    $R_{kj} = Q_k{}' \times A_j$
                    $A_j = A_j - R_{kj} \cdot A_j$
                    end
          end
end
```

The algorithm starts by storing the elements of input matrix into the Q matrix. In every iteration we select one column of Q, calculate the diagonal element of R. This element is the Euclidean norm of column of matrix Q which has been selected. The Q matrix column is divided by the diagonal element of R. The other elements of the row of R is calculated by taking product of columns of matrix A and the column of matrix Q. The Q matrix is updated in the next step and the iteration runs again till we reach the last element of matrix R.

Inversion of upper triangular matrix R: This is the second step in solving the least square problem. The inverse of a triangular matrix requires less calculation compared to full matrix because of zero entries. The algorithm for the same is given below:

Algorithm 4

Input: Right triangular matrix, R

Output: Right Triangular matrix $V = R^{-1}$

```
Begin
for i=1:n
$V_{ii} = 1/R_{ii}$
end
for j = 1:n
          begin
                    for i = 1:j-1
```

```
              begin
                  for k = 1:j-1 begin
                  $V(i,j) = V(i,j) + V(i,k) * R(k,j)$
                  end
              end
          end
      end
  end
end
```

CHAPTER 3

IMPLEMENTATION

3. Implementation

Implementation has been done on MATLAB for the sensing stage and on VHDL for the reconstruction stage. Target FPGA is Artix 7 100T. The communication protocol used for communication between the two stages is UART. UART is implemented on the FPGA to enable the FPGA to communicate with the PC where the UART is implemented as virtual COM port through the USB.

Data format utilised for the project is of fixed point type. The data format of the measured values will be unsigned 0.16 (0 integer bits and 15 fraction bits) fixed point format with range of values from -0.999969482421875 to +0.999969482421875. The data format of the sensing matrix will be signed 0.15 (0 integer bits and 15 fraction bits) fixed point format as the measured values but with a restricted range of values from -0.999969482421875 to +0.999969482421875. The minimum resolution of the data format used will be 0.000030517578125.

3.1. Block Diagram

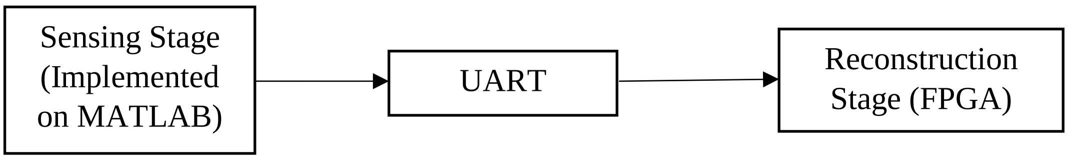
Sensing Stage (Implemented on MATLAB)
UART
Reconstruction Stage (FPGA)

Figure 2 Block Diagram of CS system

3.2. Sensing Stage

The sensing stage is implemented in MALAB 2014. The sensing stage implemented is designed to sample signals with max sparsity of 8 on a 256 length data vector with 64 measurements taken. This requires for a dictionary of dimension 64×256. The dictionary is generated by using the random matrix generator and then it is transformed into a orthogonal matrix so as to satisfy the necessary requirements of sensing matrix.

The original signal is unsigned or treated as having only positive signed values. The sensing is done using the measurement matrix and the sampled data is converted to binary fixed point representation mentioned above. The sensed data is then sent through the communication module to the target hardware which in this case is a FPGA.

3.3. Communication Module

Communication between the sensing stage and the reconstruction stage is achieved through the use of UART. UART on the sensing stage, which happens to be a PC, is implemented by use of Virtual COM port over physical USB port. The data from the sensing stage is transmitted though the USB port. The reconstruction stage utilizes a USB – UART Bridge to convert USB data packets into serial communication data.

The UART module is implemented using the FPGA resources. The UART is configured with 921600 bps (900 Kbps) Baud Rate, 8 Data bits, no Parity bits, 1 Stop Bits with Hardware Flow Control. The UART has provision to default to 9600 bps baud rate if the UART module for some reason doesn't initialise to 921600 bps. The UART module on reconstruction stage has 2 wires (TX, RX) to connect to the sensing stage. Presence of USB – UART Bridge eliminates the need for implementing the whole protocol. UART module in the FPGA has internal signals (DIN, DOUT) connected to the reconstruction module. The signals are 24 bit wide.

DIN signal, which is obviously an input to the UART module, has the format as specified below:

Index	Estimated Signal Data
23 16	15 0

The index represents the index of the estimated signal vector element sent through the bits 15 to 0. The data is sent MSB first and on top of that MSb first too.

DOUT signal, which the output from the UART module, has the format as specified below:

State	Bank Select	Data Bits
23 22	21 16	15 0

The state represents the role of the data received in the Data Bits. Bank Select data field selects the bank from 64 banks of memory available for writing.

RTL Design of the UART:

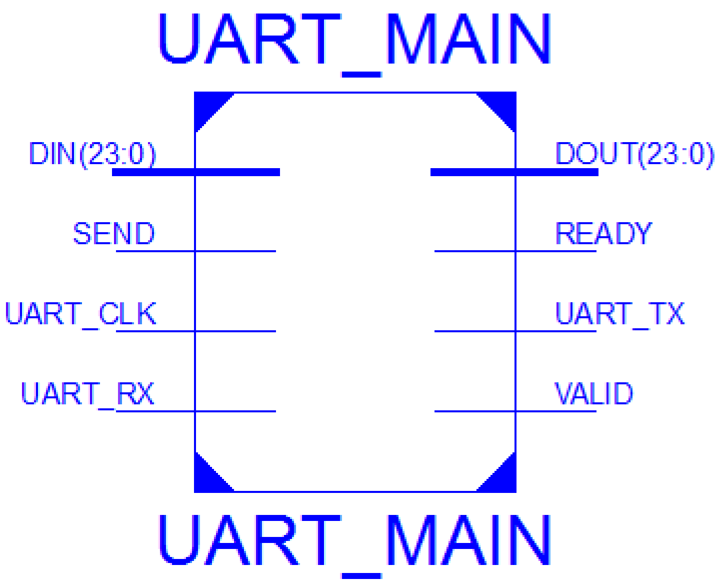

Figure 3 RTL of UART Module

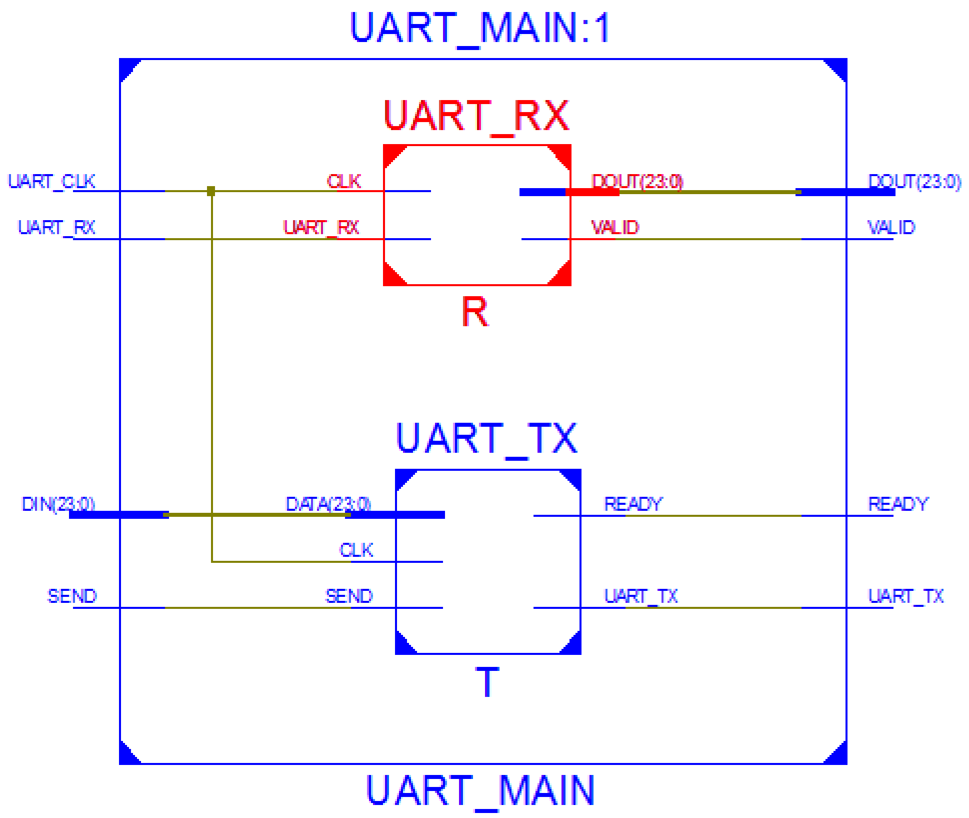

Figure 4 RTL of the Submodules of the UART Module

3.4. Reconstruction Stage

Reconstruction stage is implemented in the FPGA. The HDL program used for implementation is VHDL and the toolchain Vivado ISE WebPACK version.

Basic Implementation Details are:

- Fixed point number representation used.
- Data width – 16 bits
- Fixed point format (0.15Q), 1 bit for sign and rest 15 for fractional bits.
- Data range is -1 to 1.
- Implemented Vector length

$$x = 256$$

$$y = 64$$

$$\hat{x} = 8$$

- Sensing Matrix Dimension, ϕ : 64×256

3.4.1. Block Diagram

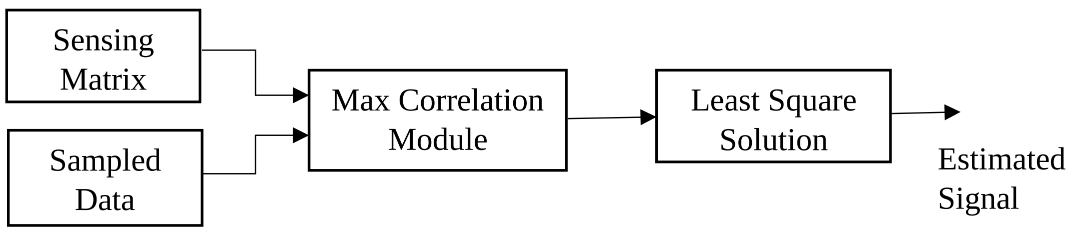

Figure 5 Block Diagram of Implementation Stage

3.4.2. Target Hardware

The target hardware for implementation of the reconstruction stage is Xilinx Artix 7 100T FPGA on the Nexys 4 development board.

3.4.3. Top Module – CS_Reco

The CS_Reco module is implemented through the use of six submodules which are as below:

- MUL_RAM_Array Module
- Adder Tree Module
- Max Module
- FISQ Module

MUL_RAM_Array Module and Adder Tree Module are shared between the Max Correlated module and Least Square Solution module to minimize the resource usage. The Max Correlated Module finds the maximum correlated vector and the Least Square module solves the Least Square problem by using QR decomposition ,inverting the Right Triangular matrix and FISQ module.

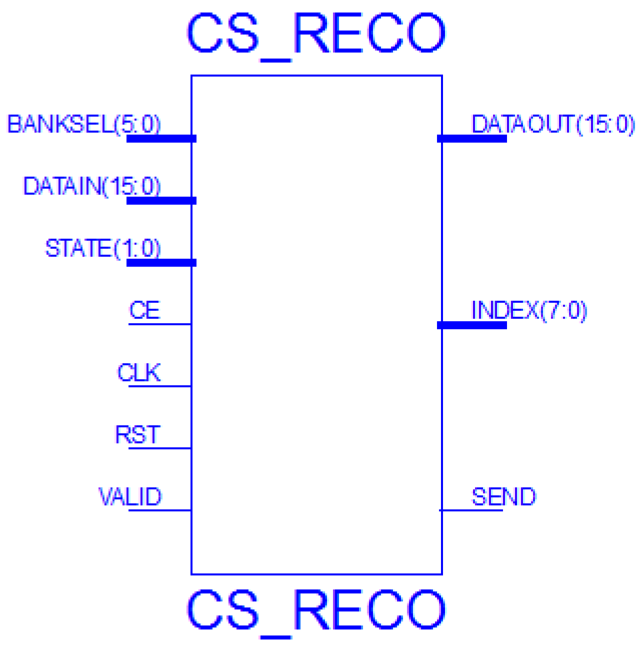

Figure 6 RTL of the CS_RECO Module

MUL RAM Array Module

The MUL RAM Array Module implements an array of RAM and a 16 bit register followed by Multiplier particularly 64 in number for the parallel access to the all the elements of columns of the sensing matrix, Φ. The RAM is implemented on the Block RAM

Resources in the Single Port Configuration and the Multiplier on the DSP48e resources available on the FPGA.

The RAM has data width of 15 bits and depth of 290 for the first 8 RAMs of the array and 256 for the rest of the RAMs in the array, to store 1 row of the sensing matrix, C Matrix, Q Matrix, R Matrix, V Matrix and the Residue register. The output of the Multiplier is recast into the signed 0.15 fixed point format to conserve resources.

The MUL_RAM_Array module is shared between the Max Correlation Module and the Least Square Module for storing data, multiplication and parallel subtraction operations.

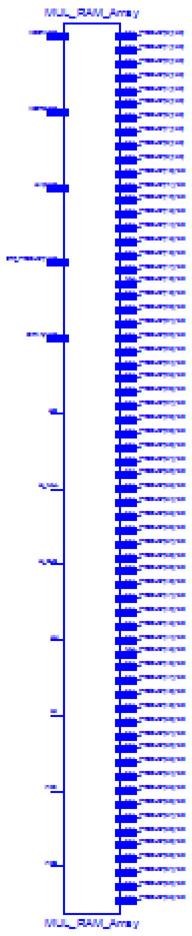

Figure 7 RTL of the MUL RAM Array Module

Adder Tree Module

The Adder Tree Module is implemented using 8:1 Adder Trees. The 64 inputs from the MUL RAM Array are input to the module and module produces the sum in two clock cycles. Single Adder Tree produces output in single clock cycle. For Least Square Solutions, a single module is implemented for performing the addition work.

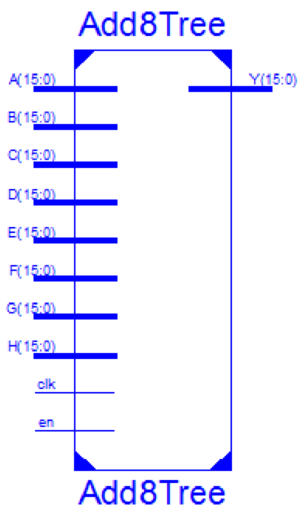

Figure 8 RTL of the Adder Tree Module

Max Block

The Max Module calculates the maximum correlated vector and outputs the max correlated sum and the index of the max correlated vector. It is essentially an comparator circuit.

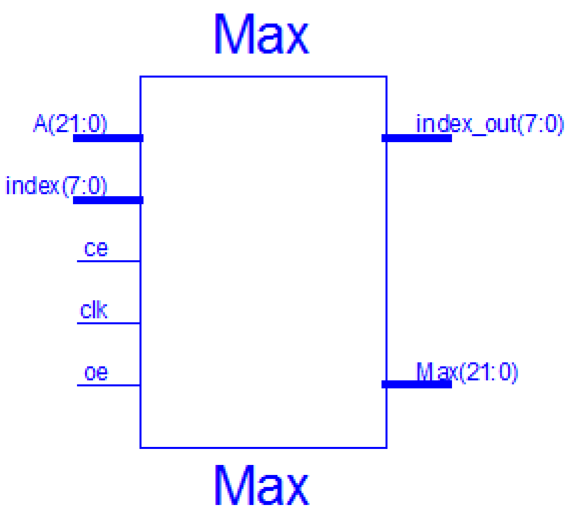

Figure 9 RTL of Max Module

FISQ Module

The FISQ Module implements the Fast Inverse Square Root algorithm along with the Newton's Iteration method for increasing accuracy. The C algorithm for implementing the FISQ algorithm [11] is as below:

```
float invSqrt(float x)
{
     float xhalf = 0.5f*x;
     union
     {
            float x;
          int i;
     } u;
     u.x = x;
     u.i = 0x5f3759df - (u.i >> 1);
     x = u.x * (1.5f - xhalf * u.x * u.x);
     return x;
}
```

Synthesized RTL of the FISQ Module is as below:

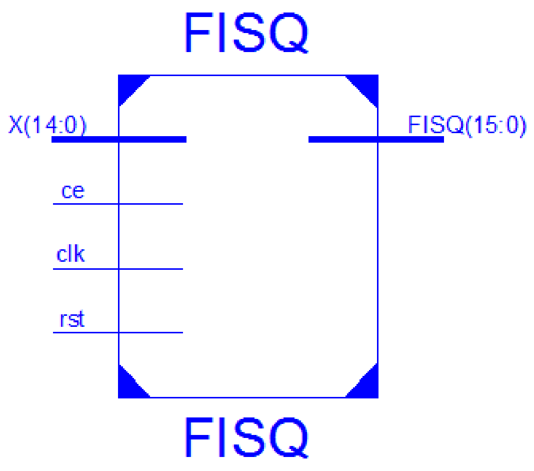

Figure 10 RTL of the FISQ Module

CHAPTER 4

RESULTS

4. Results

4.1. FISQ Error Analysis

FISQ or Fast Inverse Square Root has been implemented using an approximate algorithm to circumvent the usage of Divider and Square root operations. This algorithm as described in implementation introduces error into the calculation. Since the inverse square value is utilized in the Graham Schmidt QRD process, it remains essential to minimize the error from using the algorithm, which is done by applying the Newton's Iteration Method.

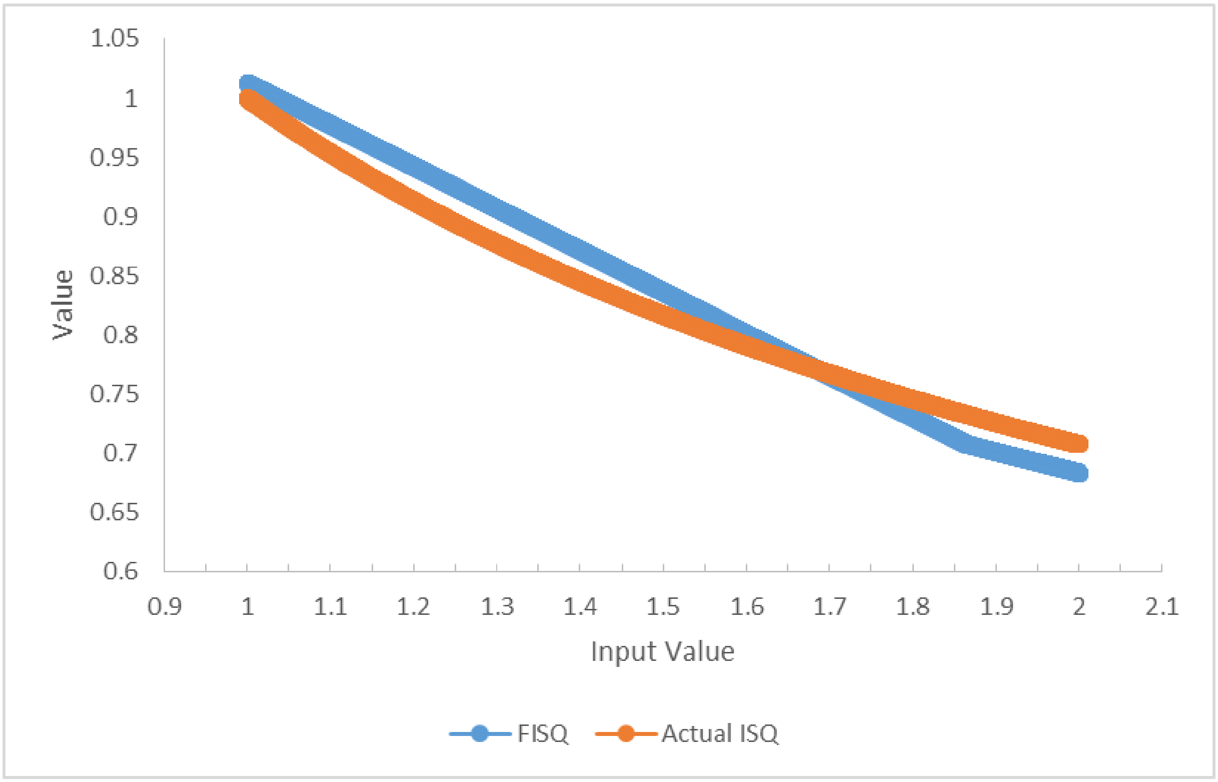

Figure 11 Comparison of the values obtained from the algorithm with the actual values

As evident from the graph above using only the approximate algorithm gives out output with notable differences from the original value. Using the FISQ module in this state will increase the error in the whole system.

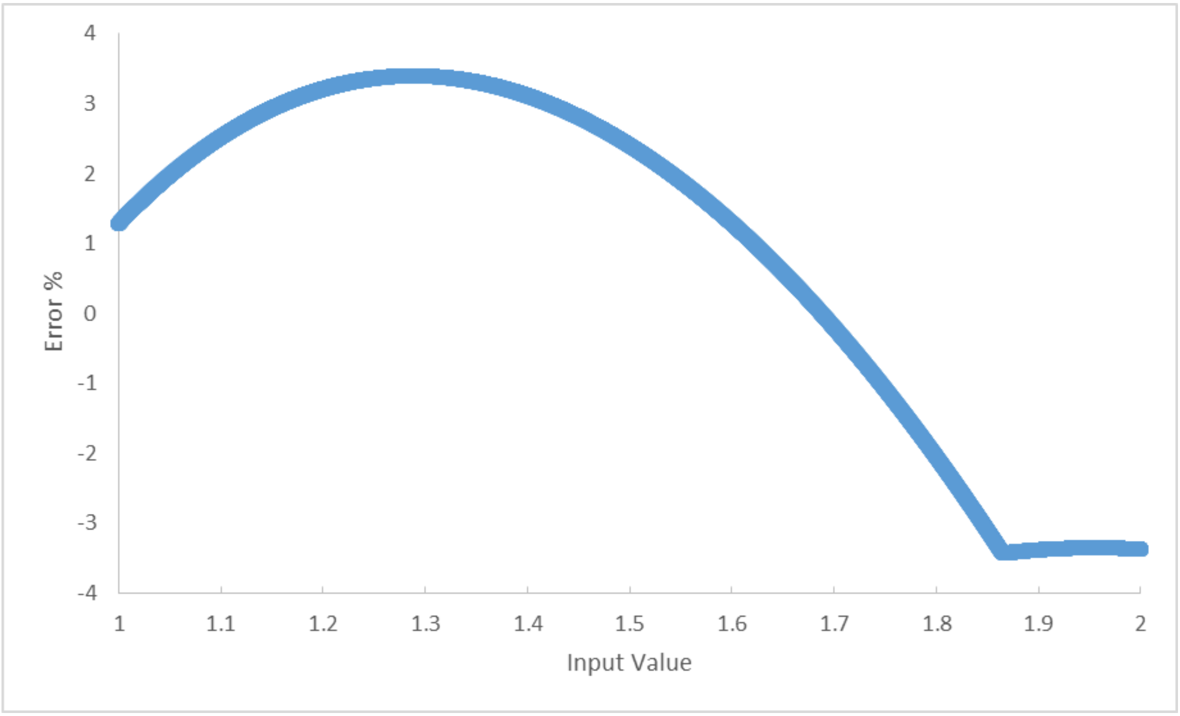

Figure 12 Error % between the original value and approximate values

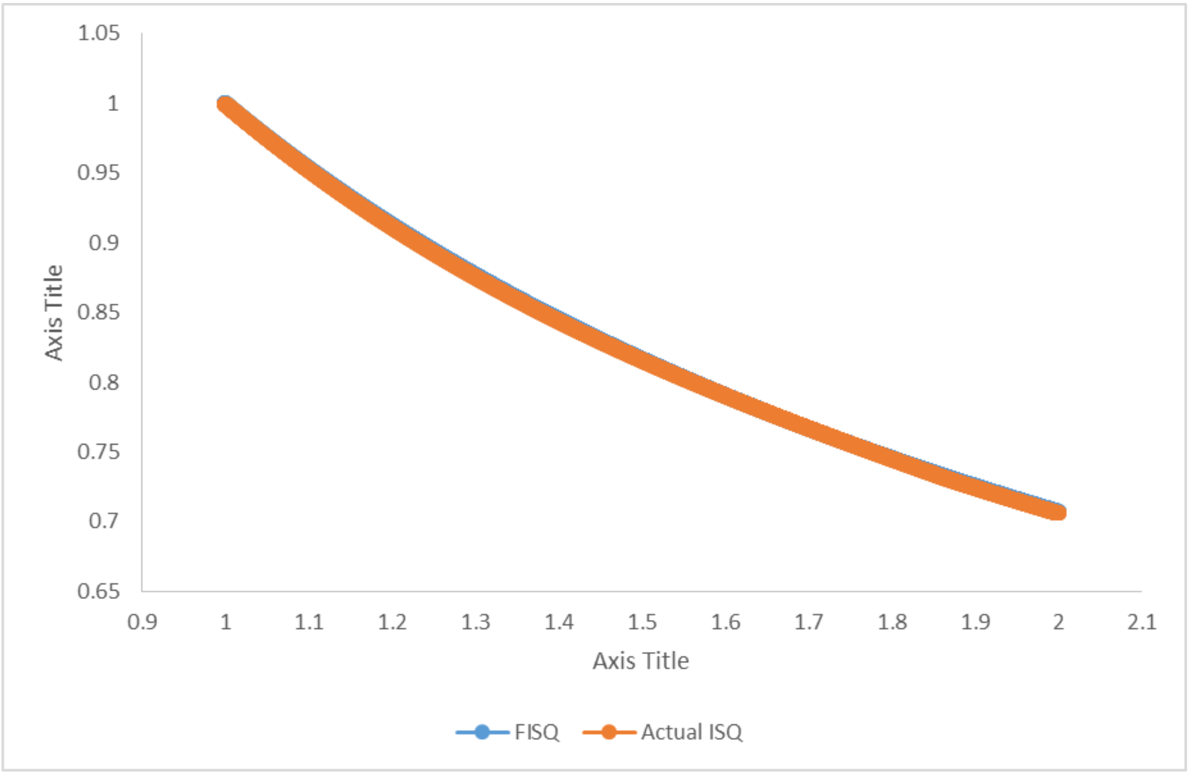

Figure 13 Comparison of the values from FISQ Module after application of Newton's Iteration Method and Original Values

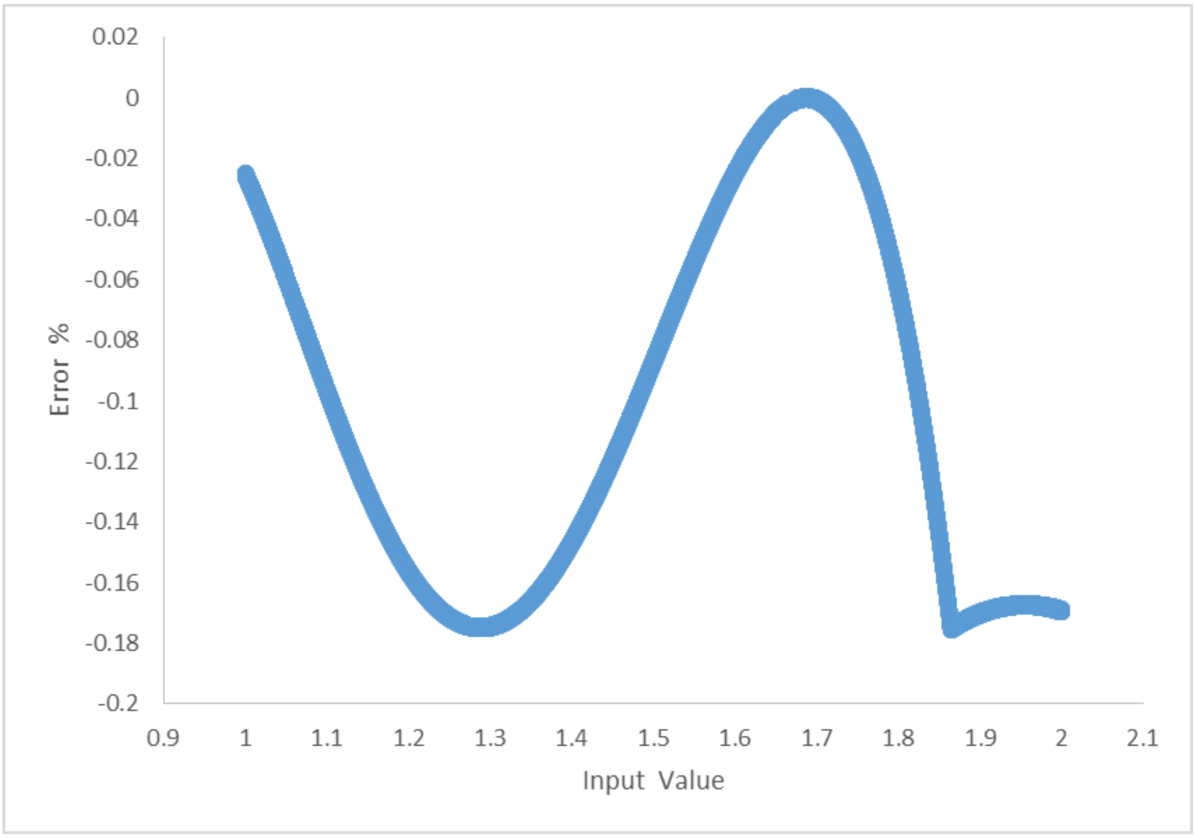

Figure 14 Error % between Original values and FISQ value after application of Newton's Iteration Method

As evident from the above graph utilization of the Newton's Iteration Method reduces the error in approximation by considerable amount. Theoretical calculations suggest that utilization of another iteration of Newton's method will exhaust the precision bits of single precision floating point number. In this project, only one iteration is used to save resource and reconstruction time.

4.2. Simulation Results

Simulation of the whole sensing and reconstruction stages of the Compressed Sampling problem has been done on the MATLAB.

4.2.1. Simulation Results with Number of Measurements Varied

Simulation is carried out on a 4096 length random signal with sampled vector length of 256 for sparsity of 8. Signal was generated at random using sparse random generator and low amplitude Gaussian noise was added to the signal to simulate real world scenario where the low amplitude values below certain threshold are considered zero so as to consider the signal sparse.

Simulation waveforms as attached below:

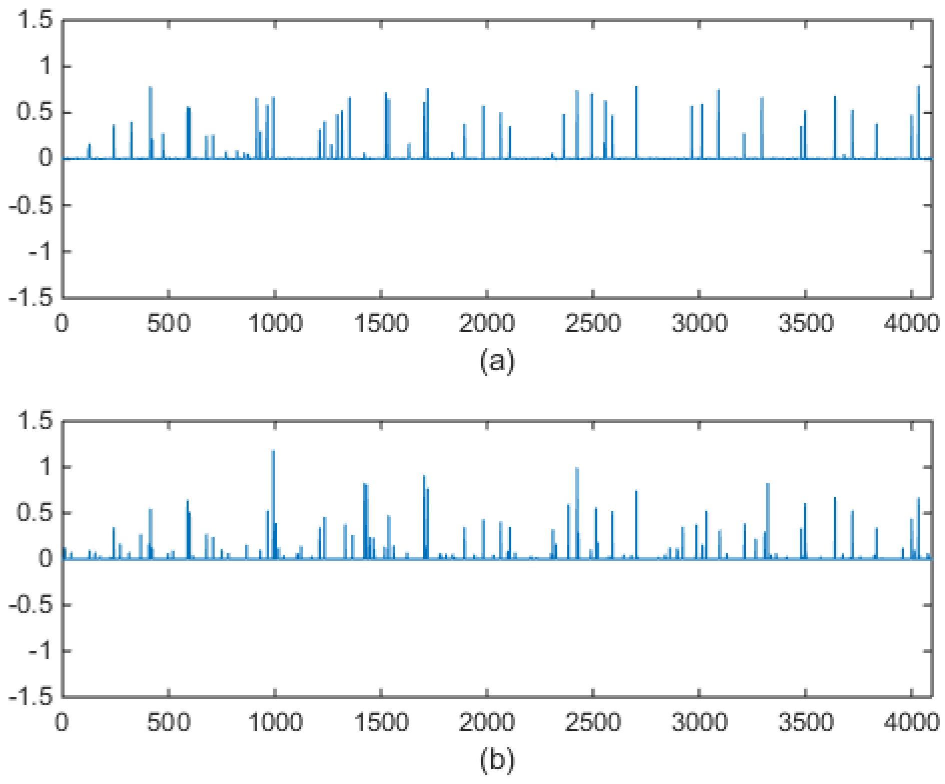

Figure 15 CS Reconstruction with Number of Measurements, m=16. MSE of reconstructed signal is 0.00290763897881587

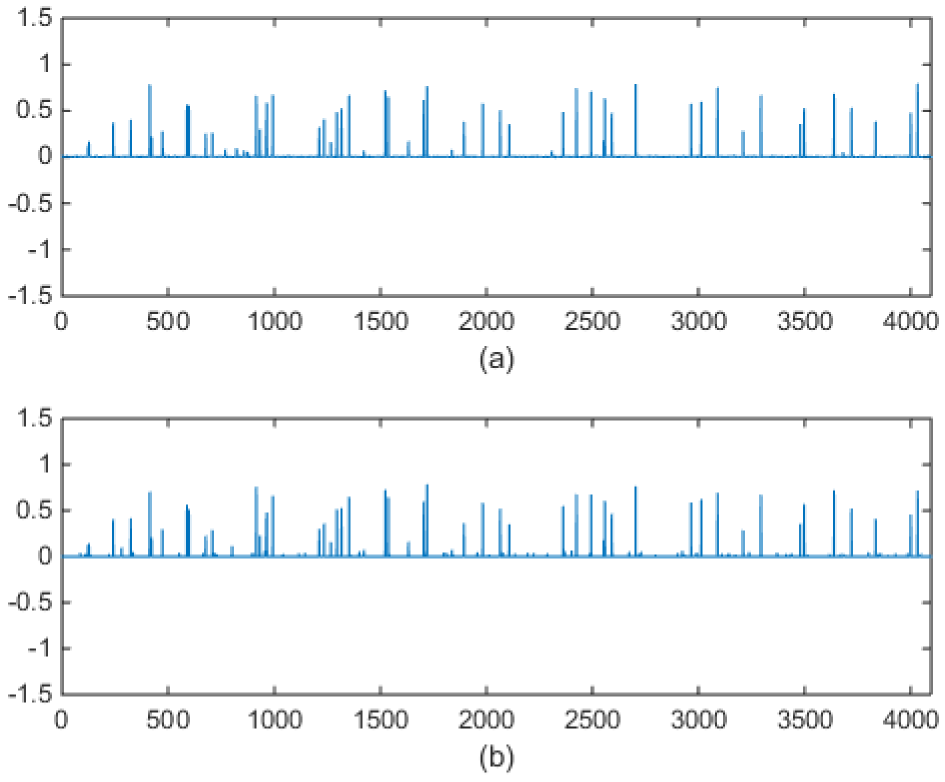

Figure 16 CS Reconstruction with Number of Measurements, m=32. MSE of reconstructed signal is 9.49620736657815e-05

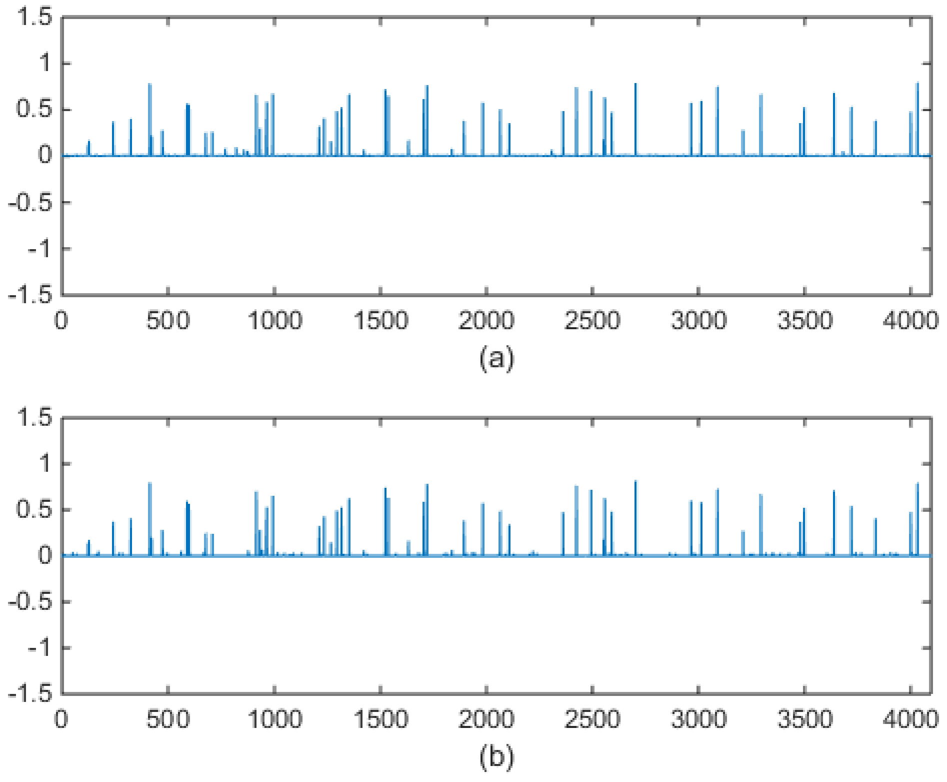

Figure 17 CS Reconstruction with Number of Measurements, m=64. MSE of reconstructed signal is 5.31479595498806e-05

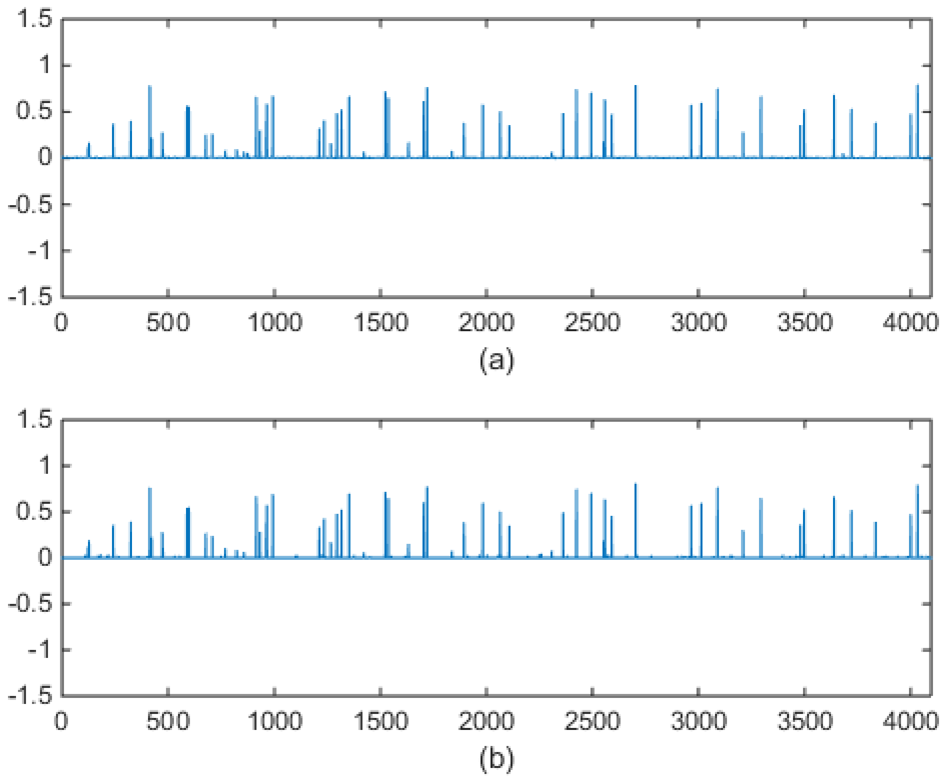

Figure 18 CS Reconstruction with Number of Measurements, m=128. MSE of reconstructed signal is 3.95939928824365e-05

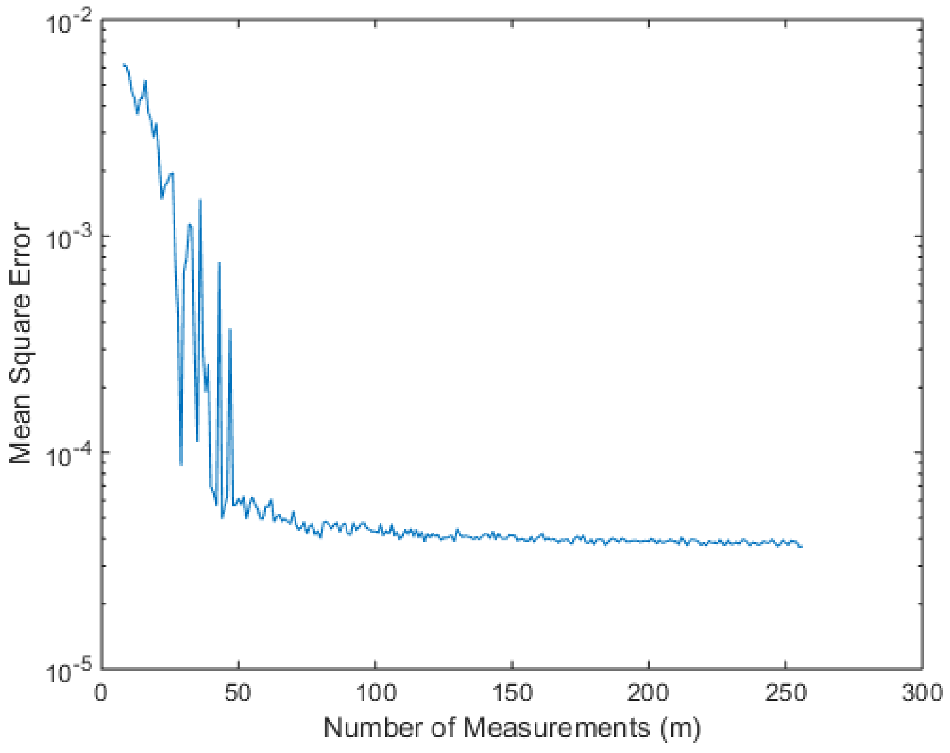

Figure 19 MSE of Reconstructed signal with variation in number of measurements

As, seen from the graph, it is clear that the larger the number of measurements, larger is the probability of reconstruction with less error. We see that the reconstructed signal MSE or Mean Square Error decreases with increase of number of measurements for number of measurements low, but remains almost constant in higher number of measurements area. This shows that, we can reconstruct the success with high probability for number of measurements greater than or equal to 3 times the sparsity for this type of dictionary.

4.2.2. Simulation Results with Sparsity varied

Another parameter that can be varied is the sparsity level of the signal. For this purpose we use the same dictionary but with number of measurements fixed at 128. The sparsity is varied from 16 to128. It is found that lower the sparsity, higher the chance of recovery of signal.

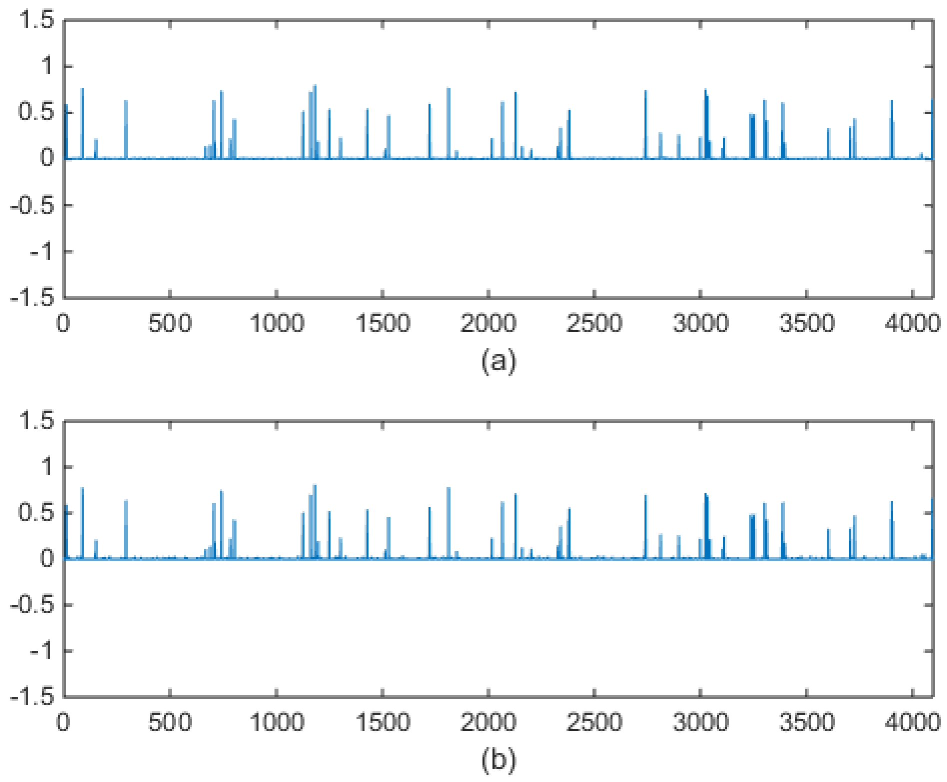

Figure 20 CS Reconstruction with Sparsity, s=16. MSE of reconstructed signal is 4.80328602702109e-05

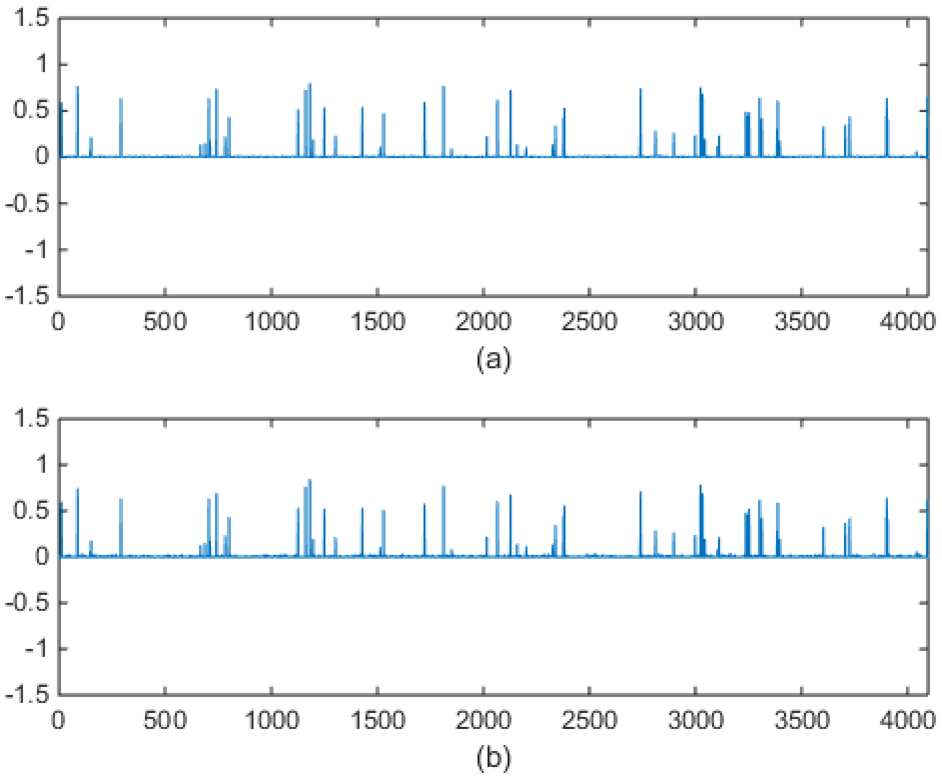

Figure 21 CS Reconstruction with Sparsity, s=32. MSE of reconstructed signal is 5.34504725853364e-05

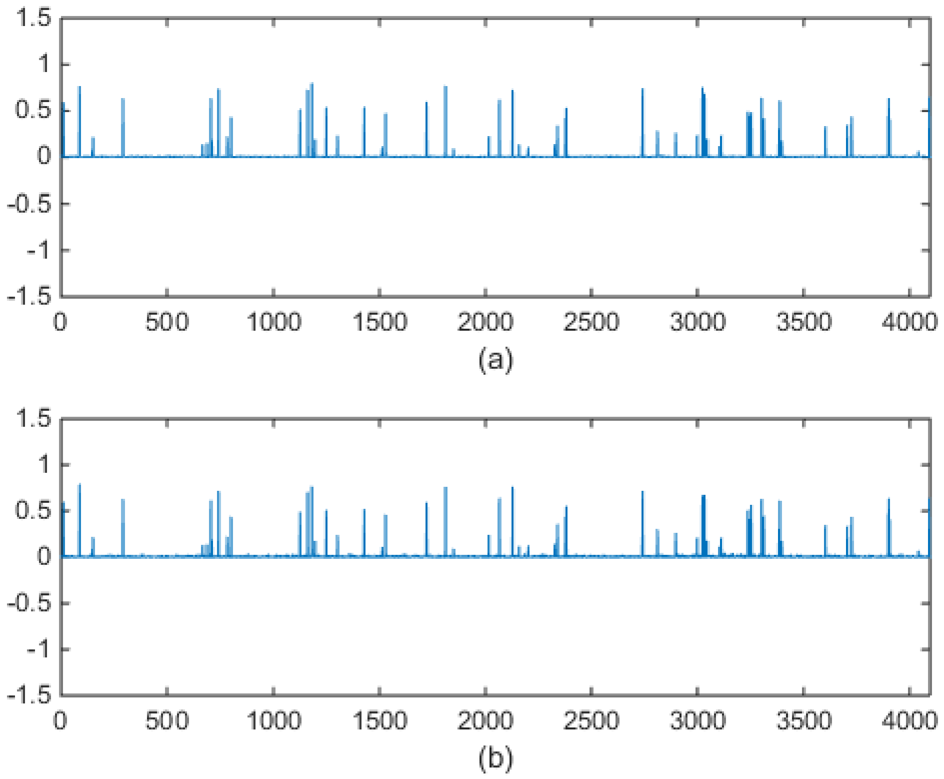

Figure 22 CS Reconstruction with Sparsity, s=64. MSE of reconstructed signal is 5.25423356642963e-05.

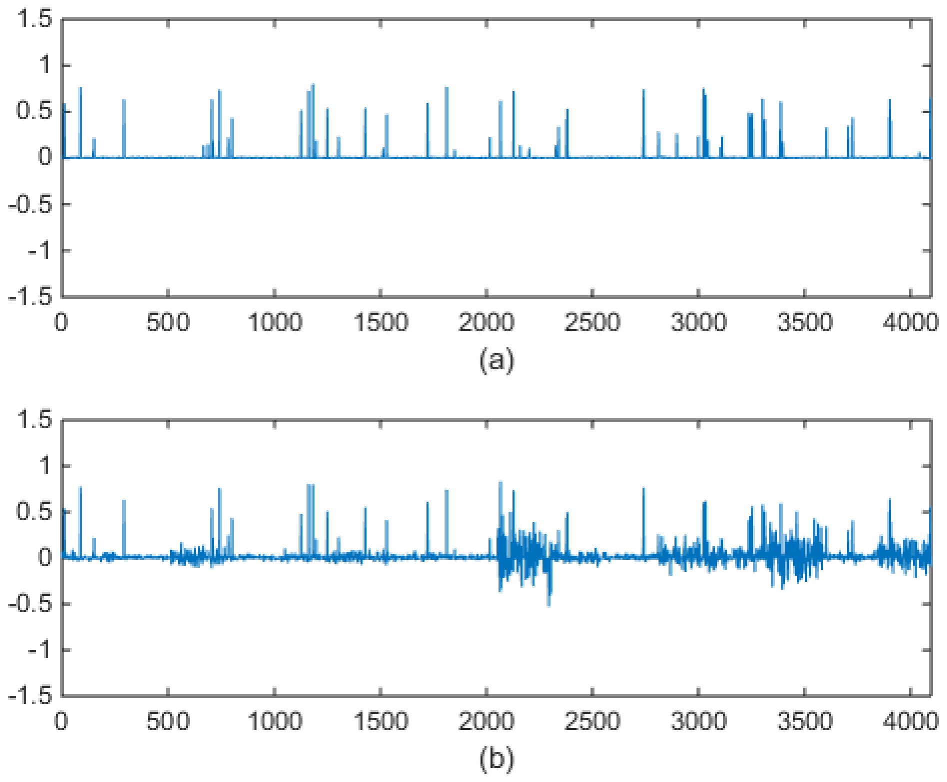

Figure 23 CS Reconstruction with Sparsity, s=128. MSE of reconstructed signal is 0.0210434520458666.

4.3. Hardware Implementation Results

The whole reconstruction stage design was implemented on the target hardware Artix 7 100T FPGA. System clock of the FPGA ran at 100MHz and was used throughout the design. Implementation result show that the whole reconstruction for a vector of length 256 and sparsity 8 with number of measurements 64 takes about 25 μs, ignoring the overhead for UART communication whereas the MATLAB simulation of the code took about 0.31 s.

CHAPTER 5

CONCLUSION

5. Conclusion

This work implements the CS reconstruction algorithm on FPGA. The signals sampled here are 1D signals which are sparse in their natural domain. We found that the minimum bound for number of measurements for faithful recovery of the compressed sampled signal to be more than or equal to the 4 times the sparsity of the signal sampled. This is in conformance with the theory as stated before. We also found that the lesser the sparsity, less number of measurements required. We also found that the randomly generated dictionary used here works as an excellent sensing matrix for sampling purposes. As expected the reconstruction time on the hardware beat the time to reconstruct on the software environment.

For future work, the reconstructed time can be reduced by running the individual modules at different clock frequency, such that the overall system runs at a higher speed than the currently use single clock throughout the system. The sensing stage can be implemented on the hardware. The communication used here is UART which can be replace by much faster networking protocol such as the Wi-Fi and Ethernet, etc. The work can further be extended to compress sample images, audio signals and video signals.

References

[1] Yong Fang. "2D sparse signal recovery via 2D orthogonal matching pursuit ", *Science China Information Sciences*, 04/2012

[2] http://dsp.rice.edu/cs

[3] Emmanuel J. Candes, Michael B. Watkin, “An Introduction To Compressed Sampling” *IEEE Signal Processing Magazine*, vol.25, no. 2, pp. 21-30, Mar 2008

[4] J. Stanislaus and T. Mohsenin, “Low-complexity FPGA implementation of compressive sensing reconstruction,” *SPIE Conference on Defense, Security, and Sensing*, April 2012.

[5] Chen, Junjie, Qilian Liang, John Paden, and Prasad Gogineni. "Compressive sensing analysis of Synthetic Aperture Radar raw data", *2012 IEEE International Conference on Communications (ICC)*, 2012.

[6] http://www.cs.ubc.ca/labs/scl/spot

[7] A. Septimus and R. Steinberg, “Compressive sampling hardware reconstruction,” in Circuits and Systems (ISCAS), *Proceedings of 2010 IEEE International Symposium* on, 2010, pp. 3316–3319.

[8] K. Hayashi,M. Nagahar, T. Tanaka, “A User’s Guide To Compressed Sensing For Communications Systems”, *Ieee Trans. Commun.*, Vol.E96-B,No.3 March 2013

[9] Joel A. Tropp, Member, Ieee, And Anna C. Gilbert “Signal Recovery From Random Measurements Via Orthogonal Matching Pursuit.” *Ieee Transactions On Information Theory*, Vol. 53, No. 12, December 2007.

[10] Vahid Abolghasemi, Saideh Ferdowsi, Bahador Makkiabadi, And Saeid Sanei, “On Optimization Of The Measurement Matrix For Compressive Sensing.” *18th European Signal Processing Conference* (Eusipco-2010).

[11] “Fast inverse square root,” Aug. 3 2011. [Online]. Available: http://en.wikipedia.org/wiki/Fast inverse square root

[12] Rushton, Andrew. *VHDL for logic synthesis*. John Wiley & Sons, 2011.

[13] Woods, Roger, et al. *FPGA Based Implementation of Signal Processing Systems*. John Wiley & Sons, Ltd, 2008. By Roger Woods

[14] Bhasker, Jayaram, and Jayaram Bhasker. *A Vhdl primer*. Prentice Hall PTR, 1999.

[15] J.-L. Starck, F. Murtagh, and J. Fadili, Sparse Image and Signal Processing. Cambridge University, 2010.

[16] M. Andrecut,"Fast GPU implementation of sparse signal recovery from random projections," 2008. [Online]. Available: http://www. arxiv. org/PS cache/arxiv/pdf/0809/0809. 1833v1. pdf

[17] O. Maslennikow, P. Ratuszniak, and A. Sergyienko,"Implementation of Cholesky LLT-Decomposition Algorithm in FPGA-Based Rational Fraction Parallel Processor," Mixed Design of Integrated Circuits and Systems, 2007. MIXDES '07. 14th International Conference on, pp. 287-292, 2007.

[18] E. Candès,"Compressive sampling," in Proceedings of the International Congress of the Mathematicians, 2006, pp. 1433-1452.

[19] M. Karkooti, J. Cavallaro, and C. Dick,"FPGA Implementation of Matrix Inversion Using QRD-RLS Algorithm," Signals, Systems and Computers, 2005. Conference Record of the Thirty-Ninth Asilomar Conference on, pp. 1625-1629, 2006.

[20] Y. Chen and X. Zhang,"High-speed architecture for image reconstruction based on compressive sensing," Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on, pp. 1574-1577, 2010.

[21] Candès, E. J., Romberg, J., Tao, T., Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inform. Theory 52 (2006), 489–509.

[22] Boyd, S.,Vandenberghe, L.,Convex Optimization. Cambridge University Press, Cambridge 2004.

[23] D. L. Donoho, “Compressed sensing,” IEEE Trans. on Information Theory, vol. 52, pp. 1289 – 1306, Apr. 2006.

[24] D. L. Donoho, Y. Tsaig, and Jean-Luc Starck, “Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit,” Technical Report, Mar. 2006.

[25] T. Do, T. D. Tran, and L. Gan, “Fast compressive sampling with structurally random matrices,” Proceedings of Acoustics, Speech and Signal Processing, 2008. ICASSP 2008.

[26] Y. Pati, R. Rezaifar, and P. Krishnaprasad, “Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition,” presented at the 27th Asilomar Conf. Signals, Systems and Comput., Nov. 1993.

[27] E. Candès, “The restricted isometry property and its implications for compressed sensing,” Compte Rendus de l’Academie des Sciences. Paris, France, 2008, vol. 346, pp. 589–592, ser. I

[28] D. Needell and R. Vershynin, “Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit,” Found. Comput. Math., vol. 9, no. 3, pp. 317–334, 2009.

[29] D. Needell and R. Vershynin, “Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit,” IEEE J. Sel. Topics Signal Process., vol. 4, no. 2, pp. 310–316, Apr. 2010.

[30] Bechler, Paweł, and Przemysław Wojtaszczyk. "Error estimates for orthogonal matching pursuit and random dictionaries." *Constructive Approximation* 33.2 (2011): 273-288.