AN EFFICIENT GPGPU BASED CCSDS RECOMMENDED DWT DECOMPRESSOR AND OPTIMIZATION OF HYPERSPECTRAL COMPRESSION PARAMETERS

By

Maniya Shailjaben Mukeshbhai

Enrollment No: 160280723007

Guided by

Prof. Bakul B. Panchal

Assistant Professor in Information Technology Department
L.D. college of engineering, Ahmedabad, Gujarat

&

Hiren Rambhia

Scientist-SE
PCSVD/PCEG/SEDA
Space Application Centre
ISRO, Ahmedabad

Thesis Submitted to
Gujarat Technological University
In Partial Fulfilment of the Requirements for
The Degree of Master of Engineering
In Information Technology

May 2018

Information Technology Department

L.D. college of engineering - Ahmedabad- 380 015

भारत सरकार
अंतरिक्ष विभाग
अंतरिक्ष उपयोग केन्द्र
आंबावाडी विस्तार डाक घर,
अहमदाबाद-380 015. (भारत)
दूरभाष : +91-79-26913050, 26913060
वेबसाईट : www.sac.isro.gov.in/www.sac.gov.in

Government of India
Department of Space
SPACE APPLICATIONS CENTRE
Ambawadi Vistar P.O.
Ahmedabad - 380 015. (INDIA)
Telephone : +91-79-26913050, 26913060
website : www.sac.isro.gov.in/www.sac.gov.in

April 19, 2018

CERTIFICATE OF PRACTICAL TRAINING

Name of the student	:	MANIYA SHAILJABEN MUKESHBHAI
Name of the institution	:	L. D. COLLEGE OF ENGINEERING, AHMEDABAD
Period of training	:	19/06/2017 to 19/04/2018
Name of the division where training was taken	:	SEDA/PCEG/PCSVD

DETAILS OF TRAINING

"AN EFFICIENT GPGPU BASED CCSDS RECOMMENDED DWT DECOMPERESSOR AND OPTIMIZATION OF HYPERSPECTRAL COMPRESSION PARAMETERS"

PROFICIENCY SHOWN BY THE CANDIDATE

Attendance	:	REGULAR
Conduct	:	VERY GOOD
Practical ability	:	She has good analytical & programming skill. She is sincere and hardworking.

(J RAVISANKAR)
HEAD, HRDD

CERTIFICATE

This is to certify that research work embodied in this report entitled “AN EFFICIENT GPGPU BASED CCSDS RECOMMENDED DWT DECOMPRESSOR AND OPTIMIZATION OF HYPERSPECTRAL COMPRESSION PARAMETERS” was carried out by Ms. Shailjaben Mukeshbhai Maniya (Enrolment No: 160280723007) at L. D. College of Engineering, Ahmedabad for partial fulfilment of Master of Engineering degree in (Information Technology) to be awarded by Gujarat Technological University. This research work has been carried out under my supervision and is to the satisfaction of department.

Date:

Place:

Internal Guide
Prof. B. B. Panchal
Assistant Professor
L. D. College of Engineering

External Guide
Hiren Rambhia
Sci./Engr. –SE
SAC, ISRO

Head of Department
Prof. (Dr.) H. M. Diwanji
Head of the Dept., (I.T.)
L. D. College of Engineering

Principal
Prof. (Dr.) G.P. Vadodaria,
L.D. College of Engineering,
Ahmedabad

COMPLIANCE CERTIFICATE

This is to certify that research work embodied in this dissertation titled “AN EFFICIENT GPGPU BASED DWT DECOMPRESSOR AND OPTIMIZATION OF HYPERSPECTRAL PARAMETERS” was carried out by **Maniya Shailjaben Mukeshbhai (Enrollment No. 160280723007)** at L.D. College of Engineering (028) for partial fulfillment of Master of Engineering degree to be awarded by Gujarat Technological University. She has complied to the comments given by the Dissertation phase – I as well as Mid Semester Thesis Reviewer to my satisfaction.

Date:

Place:

Maniya Shailjaben Mukeshbhai

(Enrollment No. 160280723007)

Prof. Bakul Panchal

Assistant Professor
I.T. Department
L.D. College of Engineering

PAPER PUBLICATION CERTIFICATE

This is to certify that research work embodied in this dissertation titled “AN EFFICIENT GPGPU BASED DWT DECOMPRESSOR AND OPTIMIZATION OF HYPERSPECTRAL PARAMETERS” was carried out by **Maniya Shailjaben Mukeshbhai (Enrollment No. 160280723007)** at L.D. College of Engineering (028) for partial fulfilment of Master of Engineering degree to be awarded by Gujarat Technological University has published article “An Efficient GPGPU based DWT Decompressor And Optimization of Hyperspectral Parameters” for publication by the International Journal of Science and Research in Science, Engineering and Technology at Volume 4, Issue 4 on 10th April, 2018.

Date:

Place:

Candidate Name
Shailja M. Maniya,
Enrollment No. 160280723007
Information Technology Department
L. D. College of Engineering, Ahmedabad

Guide
Prof. Bakul B. Panchal,
Assistant Professor,
Information Technology Department
L. D. College of Engineering, Ahmedabad

THESIS APPROVAL CERTIFICATE

This is to certify that research work embodied in this dissertation titled "AN EFFICIENT GPGPU BASED DWT DECOMPRESSOR AND OPTIMIZATION OF HYPERSPECTRAL PARAMETERS" was carried out by **Maniya Shailjaben Mukeshbhai (Enrollment No. 160280723007)** at L.D. College of Engineering (028) is approved for award of the degree of Master of Engineering in I.T (Information Technology) by Gujarat Technological University.

Date:

Place:

Maniya Shailjaben Mukeshbhai

(Enrollment No. 160280723007)

Examiner(s):

___________________ ___________________

( ) ( )

UNDERTAKING ABOUT ORIGINALITY OF WORK

We hereby certify that we are the sole authors of this thesis and that neither any part of this thesis nor the whole of the thesis has been submitted for a degree to any other University or Institution.

We certify that, to the best of our knowledge, the current thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations or any other material from the work of other people included in our thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that we have included copyrighted material that surpasses the boundary of fair dealing within the meaning of the Indian Copyright (Amendment) Act 2012, we certify that we have obtained a written permission from the copyright owner(s) to include such material(s) in the current thesis and have included copies of such copyright clearances to our appendix.

We declare that this is a true copy of thesis, including any final revisions, as approved by thesis review committee.

We have checked write up of the present thesis using anti-plagiarism database and it is in allowable limit. Even though later on in case of any complaint pertaining of plagiarism, we are sole responsible for the same and we understand that as per UGC norms, University can even revoke Master of Engineering degree conferred to the student submitting this thesis.

Date:

Signature of Student :	**Signature of Guide:**
Name of Student : Shailjaben M. Maniya	**Name of Guide: Prof. Bakul Panchal**
Enrollment No: 160280723007	**Institute Code: 028**

DECLARATION CERTIFICATE

I hereby certify that I am the sole author of this thesis and that neither any part of this thesis nor the whole of the thesis has been submitted for a degree to any other University or Institution.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Indian Copyright Act, I Certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright.

Clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis review committee.

Date:

Place:

Signature:

Shailja M. Maniya
Enrollment no: 160280723007
M.E.(I.T.)
L.D. College of Engineering

ACKNOWLEDGEMENT

It is a great pleasure to acknowledge and thank everyone who have made this research work possible.

I offer my sincere gratitude and respect to Prof. (Dr.) G. P. Vadodaria, Principal, L. D. College of Engineering, Ahmedabad; Prof. (Dr.) H. M. Diwanji, Head and Associate Professor, Information Technology Department, L. D. College of Engineering, Ahmedabad for their invaluable guidance and suggestions to me during my work.

I wish to express my gratitude towards my guide Prof. Bakul B Panchal, Assistant Professor, Information Technology Department, L. D. College of Engineering, Ahmedabad for her guidance and encouragement for this research work

My heartfelt thanks to Mr. Ashish Misra, Scientist-G, and Head, Payload Checkout Software and Vision System Division (PCSVD), SAC-ISRO, Ahmedabad for his encouragement and guidance for this research work.

I wish to express my deep sense of appreciation and gratitude towards Mr. Amit Dave, Scientist-SG, PCSVD, SAC-ISRO, Ahmedabad for his valuable advice and supervision in this research work.

I extend my gratitude to My guide Mr. Hiren Rambhia, Scientist-SE, PCSVD, SAC-ISRO, Ahmedabad for his cordial help and guidance for completing this research work practically and successfully. Discussion with him has enriched my knowledge.

I wish to express my warm and sincere thanks to Mr. Ravi Shankar, head, HRDD, SAC-ISRO, Ahmedabad for his constant support and motivation.

I am especially thankful to all the scientist of PCSVD for giving me good ideas as and when required. I am grateful to SAC library officials and computer lab in-charge and technicians. I am grateful to all the officials of SAC-ISRO, Ahmedabad.

SHAILJA M. MANIYA

Enrollment No: 160280723007

M.E. I.T (Information Technology)

L.D.C.E Ahmedabad

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Satellite images are gaining more and more popularity in our daily life as they are helpful during situations like natural calamities or warfare. In order to save bandwidth as well as to speed up data transfer, compression can be used to download satellite images on the earth. The Consultative Committee for Space Data Systems (CCSDS) had proposed an image data compression standard (CCSDS-IDC) for satellite image compression. This standard provides good compression performance using Discrete Wavelet Transform (DWT) and Bit Plane Encoder. As Discrete Wavelet Transform (DWT) is time consuming, to meet real time requirement this data should be decompressed as soon as massive stream of bits downlinked on the earth. In this research work, efficient GPGPU based decompressor is proposed for time efficiency, Inverse DWT two-pixel computation gives best result among all versions of IDWT which is 20x times faster than CPU implementation. Overall Decompressor system takes 20ms. Efficient Compression of multispectral and hyperspectral image is also mandatory in current and future space missions in order to save bandwidth and storage space. The Consultative Committee for Space Data Systems (CCSDS), a consortium of the major space agencies in the world, has recently issued the CCSDS 123 standard for multispectral and hyperspectral image compression. This compression system consists of two functional parts, Predictor and Encoder. In this research work, specific compression parameters like number of prediction bands, register size, weight component resolution etc. which affects predictor and encoder of the compression system have been analysed to achieve best compression for AVIRIS sensor.

Chapter 1

Introduction

1.1 Overview

The remote sensing is the major field of research. The science of acquiring certain information from remote location (long distance) using optical sensors without physical contact with object is known as remote sensing. Data is acquired with help of satellites, aircrafts, robotic devices etc. Areas which use remote sensing are oceanography, agriculture, space, geology, forestry and weather forecasting. Firstly, this data is acquired and then Introduction processed to get desired information. Based on this information different applications are developed for predicting behavior of certain systems around us. A weather, volcano, sea, winds, planets, space etc. are under inspection at day and night. This system leads us in forecasting every process around us. Therefore, natural disasters can be predicted which helps humans to save their lives and properties.

The Acquired data can be in different forms. Particularly data is captured in signal format. Signal is converted into different format which is based on application. One of the formats is image. Nowadays we capture many images in our day to day life. Cameras are common thing in our life and we can use these images in many applications. Likewise, remote sensing acquires images of different regions of earth and other planets in space. This research work focuses on the colour image data decompression and compression of multispectral and hyperspectral image data.

General cameras that we use are RGB color cameras. We can see images of RGB camera in 2-Dimensional plane. Monochromatic image is a 2-Dimensional image. RGB image is a 3-Dimensional image as respect to storage. These can be understood by considering same scene as three respected bands. In case of RGB camera it is for red, green and blue color band. If we increase bands from 3 to tens of bands then that particular image will be called as multi spectral image. If image has hundreds of bands then image is called as hyperspectral image.

Now-a-days satellite images have been more and more popularly used during warfare or situations like natural calamities. In order to save the bandwidth usage as well as storage space, these satellite images should be compressed.

Compression is a technique which reduces number of bits needed to represent the data. As we need less number of bits to represent the data, we can save storage capacity, speed up file transfer, decreases cost for storage hardware as well as network bandwidth.

Basically There are two classes of data compression methods: lossless and lossy. Under lossless compression, the original data can be reproduced exactly, while under lossy compression, quantization or other approximations used in the compression process result in the inability to reproduce the original data set without some distortion. The increased information content of data subjected to lossless compression results in a larger volume of compressed data for a given source data set. For space data system, this compression standard for different types of images is provided by Consultative Committee for Space Data System (CCSDS.)

1.2 Organization of Research Work

Chapter 1 includes introduction of research. This chapter provides objective of research work and detail introduction of organization of thesis.

Chapter 2 is basically related to review of literature. Reasons behind research work and concepts related to research work are included in this chapter. It also briefly describes about remote sensing and its payloads. From monochromatic imaging to hyperspectral imaging all concepts are included in this chapter. Challenges of decompression of colour image data is also a contribution of this chapter. It gives a short description of references to relevant publications.

Chapter 3 is related to problem identification. After studying all the literature and current research which is already done, problem identification for dissertation is described with proposed methodology in this chapter.

Chapter 4 Part-1 provide detailed GPGPU based implementation of IDWT in four different versions each contributing to GPU code optimization. Part-2 of the chapter describes MATLAB implementation of CCSDS-123.0 Lossless Multi-spectral and Hyper-Spectral Compression.

Chapter 5 Part-1 includes the results and comparisons related to four different versions of Inverse Discrete Wavelet Transform of CCSDS-122.0 Decompressor. Part-2 of the chapter presents the results on effect of CCSDS 123.0 compressor settings on bits per sample.

Chapter 6 offers conclusion related to work.

Chapter 2

Literature Review

2.1 Remote Sensing

The idea of obtaining information about an object, area, or phenomenon without physical contact is old. The technology of obtaining information about physical objects or environment by the process of capturing imagery and digital representations of energy captured by sensors is remote sensing. In short, remote sensing is a tool which uses sensors to measure the amount of electromagnetic radiation (EMR) reflected from an object or geographic area from a distance and then after extracting information from the data using algorithms. It may function with geographic information systems (GIS).

Figure 2.1 shows remote sensing phenomenon. Remote sensing instrument or Sensor can be present at orbital platform of suborbital platform. Remote sensing instrument has Instantaneous Field-of-View (IFOV) which covers area under observation of instrument.

Motion of remote sensing instrument is controllable. Acquisition of information is done through motion of instrument in any one direction.

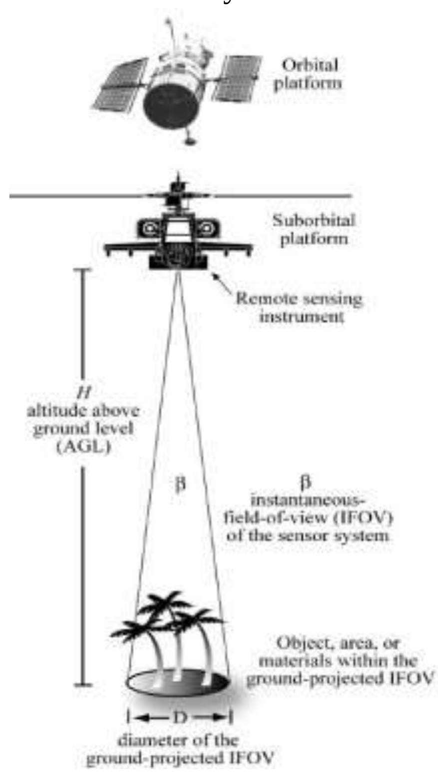

Fig 2.1 Remote Sensing System

2.2 Payload

Payload is the carrying capacity of an aircraft or launch vehicle, usually measured in terms of weight. Depending on the nature of the flight or mission, the payload of a vehicle may include cargo, passengers, flight crew, scientific instruments or experiments, or other equipment. Extra fuel, when optionally carried, is also considered part of the payload. In a commercial context (i.e., an airline or air freight carrier), payload may refer only to revenue-generating cargo or paying passengers.

For a rocket the payload can be a space probe, satellite or spacecraft which can carry humans, animals, or cargo.

2.3 Imaging System

One of these payloads is imaging system which includes optical camera system. Nowadays we all are very much used to with cameras. Many kinds of cameras are available with different types of resolutions and based on different kinds of applications.

2.3.1 Monochromatic Imaging

When image is captured in only one band then one scene can have only one image plane. An output of such image acquisition has two dimensional array of intensities respected to pixels. Such imaging has only one optical filter, band filter and detector for each pixel. This kind of imaging provide grey-scale image. Some applications need this kind of imaging system. Figure 2.2 shows graph of wavelength versus response which represents one band. This band can have wavelengths from visible band and infrared band also.

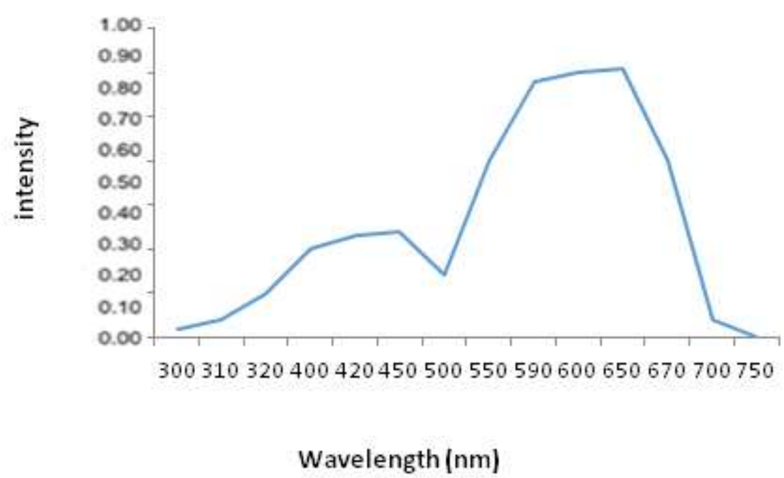

Fig. 2.2 Monochromatic spectral response

Image generated from this spectral response band will have intensities as shown in figure 2.3, which can also be used as visual output of gray scale image.

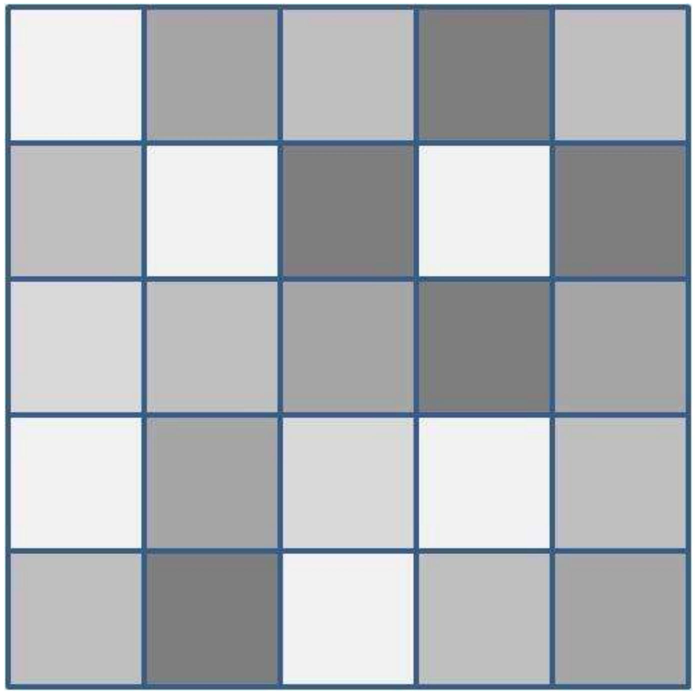

Fig. 2.3 Monochromatic intensities of image

2.3.2 Multispectral Imaging

Multispectral imaging is an extension of RGB imaging. As we know that RGB color camera has three bands, multispectral camera has tens of bands. Instead of three bands multispectral imaging camera has tens of bands. This is the reason why multispectral imaging has more spectral resolution than three band imaging.

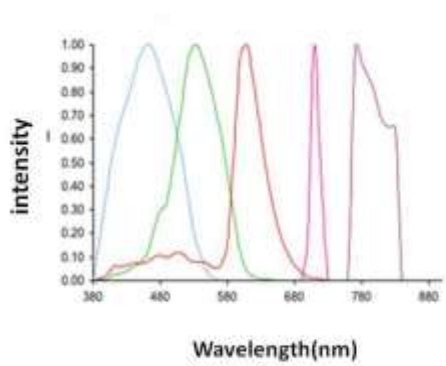

Fig. 2.4 multispectral image wavelength versus response

Bands can be from visible spectrum to infrared spectrum. Bands can be continuous or can be discontinuous. Figure 2.5 has five bands and at some places bands are overlapped and at some places they are discontinuous. Intensity response is normalised.

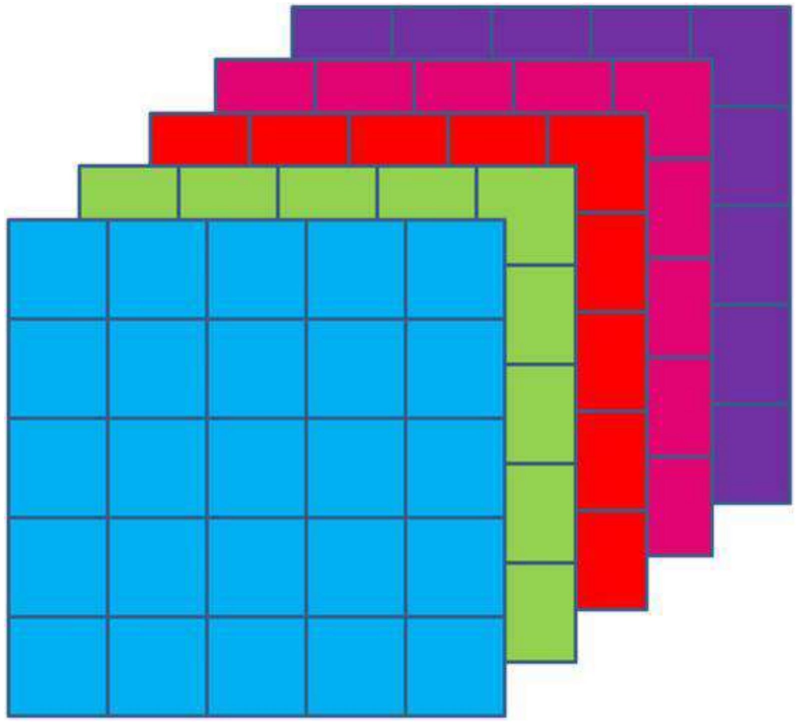

Fig.2.5 Multispectral image intensities

The reason behind overlapping bands in cameras is because of limitations present in instruments which are involved in image acquisition process. Filters are not so accurate that they can precisely filter the wavelengths. Therefore, bands are overlapped.

Here number of optical filters, band filters and detectors are depending on number of bands involved in imaging process. Figure 2.4 shows image cube structure of multispectral image.

2.3.3 Hyperspectral Imaging

Hyperspectral imaging system has hundreds of bands. Bands are overlapped, continuous and narrow. Figure 6 shows hyperspectral image cube which has hundreds of bands. Hundreds of bands create large size image. For one image of scene there will be hundreds of image planes with respect to bands. Bands can be from visible band to infrared band.

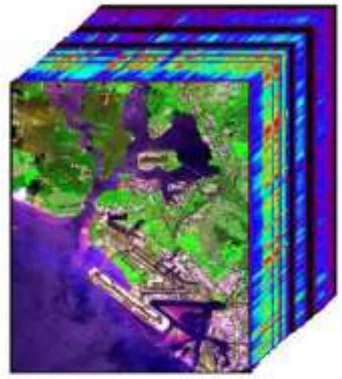

Fig.2.6 Hyperspectral image cube

Human eye is more sensitive to green band. That is the reason behind more use of green band pixel than other bands. When intensities are captured for this pattern Demosaicking will be started. In this process estimation of intensities of different bands for pixel is done using intensity values of neighbour pixels. Different techniques are available for this task. Output of such process will be 3-dimensional array which has third dimension equal to number of bands.

2.4 CCSDS Recommended Standard

For Space System, there is one committee called Consultative Committee for Space Data System (CCSDS) which was founded in 1982 by major space agencies of the world. Consultative Committee for Space Data System (CCSDS) is multi-national forum for the development of communication and data systems standards for spaceflight. Consultative Committee for Space Data System (CCSDS) approved documents contains Blue: Recommended Standards, Magenta: Recommended Practices, Green: Informational Reports, Orange: Experimental, Yellow: Records, Silver: Historical. From all of this, Blue Books: Recommended Standard defines specific interfaces, technical capabilities and protocols, provide prescriptive and/or normative definitions of interfaces, protocols, or other controlling standards such as encoding approaches.

Compression can be done for various types of data. Consultative Committee for Space Data System (CCSDS) has provided different compression methods for each type of data. i.e. CCSDS-122.0-B-1[1] gives details on how to compress Image Data. CCSDS-122 compression system contains two functional parts, Discrete Wavelet Transform (DWT) and Bit Plan Encoder. CCSDS-123.0-B-1[7] gives explanation on Lossless Multispectral and Hyperspectral Image compression.

2.4.1 CCSDS-122 .0-B-1 (Lossless Image Data Compression)

This Recommended Standard provides the Compression method for Image data. It has explained Lossy Compression as well as Lossless Compression. 'Integer 9/7 DWT is Lossless Compression whereas 'Float 9/7 DWT' is Lossy Compression. The Compressor consists of two functional parts as shown in Fig. 2.7, Discrete Wavelet Transform and Bit Plane Encoder.

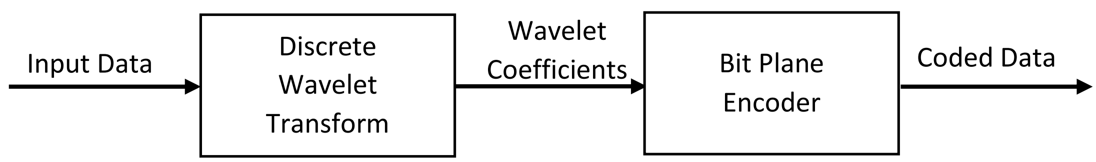

Fig. 2.7 General Schematic of coder

This Recommended Standard for the decorrelation module makes use of a three-level, two dimensional (2-d), separable Discrete Wavelet Transform (DWT) with nine and seven taps for low- and high-pass filters, respectively. Such a transform is produced by repeated application of a one-dimensional (1-d) DWT.

2.4.2 CCSDS-123 .0-B-1 (Lossless Multispectral and Hyperspectral Image Compression)

This recommended paper explains the coding system for multispectral and hyperspectral image data as shown in Fig. 2.8.

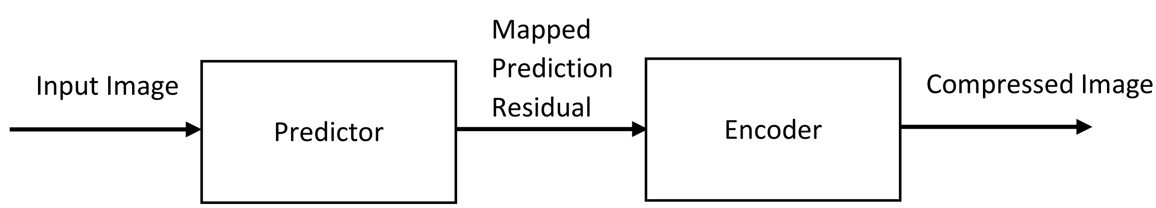

Fig.2.8 Compressor Schematic

In this compressor system, Predictor as well as Encoder consists various parameters. Different Compression results can be achieved by varying these parameters. These Compression parameters consists of number of Prediction bands, Local Sum type, Prediction modes, local diff vector types, register sizes, weight component resolution etc.

2.5 GPGPU

General-purpose computing on graphics processing units (GPGPU) is using Graphics Processing Unit (GPU), which can handle the computation on graphics, traditionally handled by CPU (Central Processing Unit). Using CPU-GPU one can achieve very good results that multiple CPUs cannot even offer with specialization in each chip. GPGPU provides parallelization of computation on image or any other graphic. GPU operates at lower frequencies and it has much more number of cores than general CPU. GPUs were originally created for high-performance workstation and they were also very expensive.

Thus, Using GPU one can achieve massive parallelism and efficient results on image or any other graphics data. There are so many generations of GPUs. As the years grow, GPUs are not developed with many features and can get excellent performance than traditional CPU.

2.6 CUDA

CUDA stands for Compute Unified Device Architecture. NVIDIA introduced CUDA at November, 2006.CUDA is a hardware or software Architecture for issuing and managing computations on the GPU. It acts as data-parallel Computing Device which does not need mapping to graphic API. CUDA software stack contains several layers as shown in below Fig.2.9 [6],

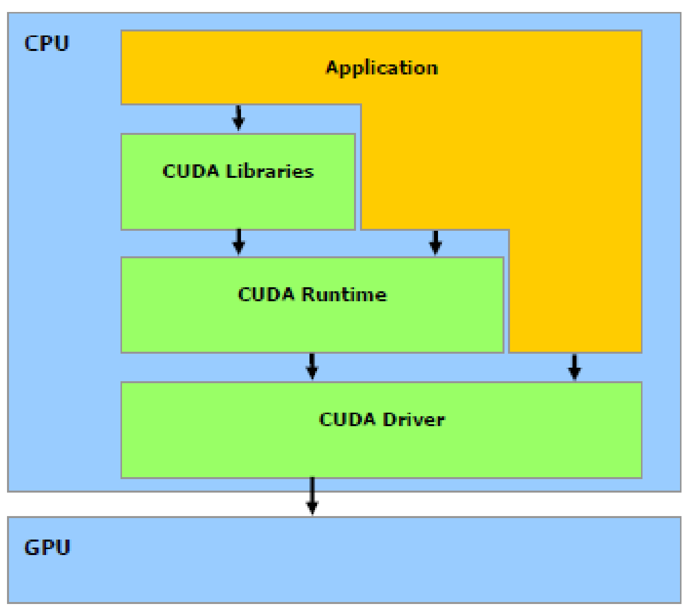

Fig.2.9 Compute Unified Device Architecture Software Stack [6]

2.6.1 Streaming Multiprocessor

In CUDA Architecture, there are some predefined number of Streaming Multiprocessors (SMs) which may vary from one to other generation of CUDA GPUs. i.e. TeslaK40C accelerator (Kepler GK110) [7] has 15 SMs as shown in Fig. 2.10.

Fig. 2.10 Kepler GK110 Architecture [7]

Streaming Multiprocessor Architecture of Kepler GK110 [7] is shown in the Fig. 2.11. Each SM in the Tesla K40C contains 192 single-precision CUDA Cores, 64 double precision units, 32 special function units (SFU) and 32 load/store units (LD/ST). Each SM has 65,536 Registers with it and also 64KB Shared Memory + L1 Cache. One SM contains 4 Warp Schedulers as shown in Fig. 2.12 [7] which executes 32 threads parallel in one warp, thus each SM allows 4 warps to be executed concurrently. Two independent instructions per warp can be dispatched each cycle.

Each Streaming Multiprocessors can run blocks simultaneously on the GPU. These blocks in turns have threads. Each thread has some unique ID through which we can compute on the data. A grid of thread blocks is executed on the device by executing one or more blocks simultaneously on each SM using time slicing. Each block is split into SIMD groups of threads called warps.

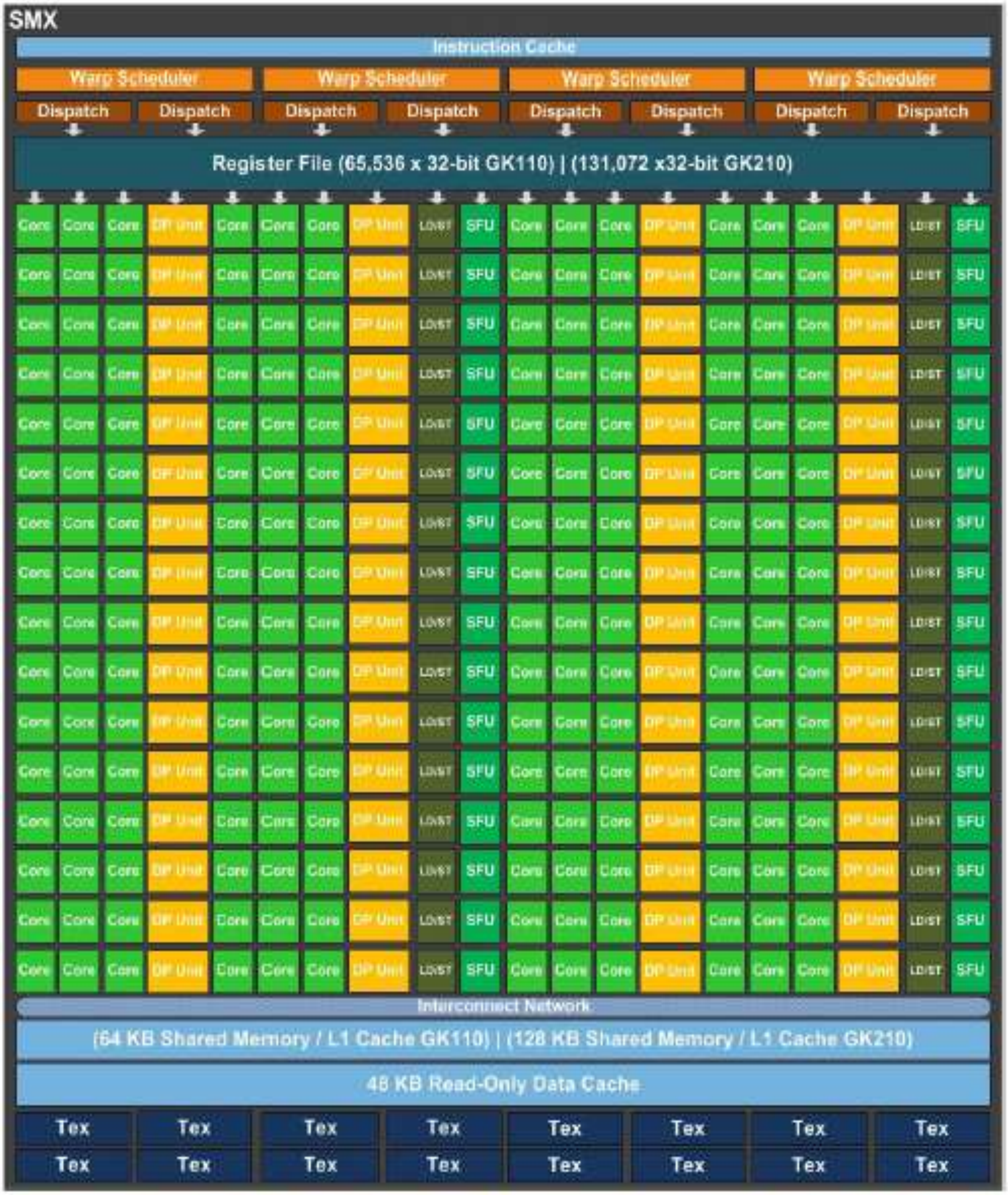

Fig. 2.11 Streaming Multiprocessor Architecture [7]

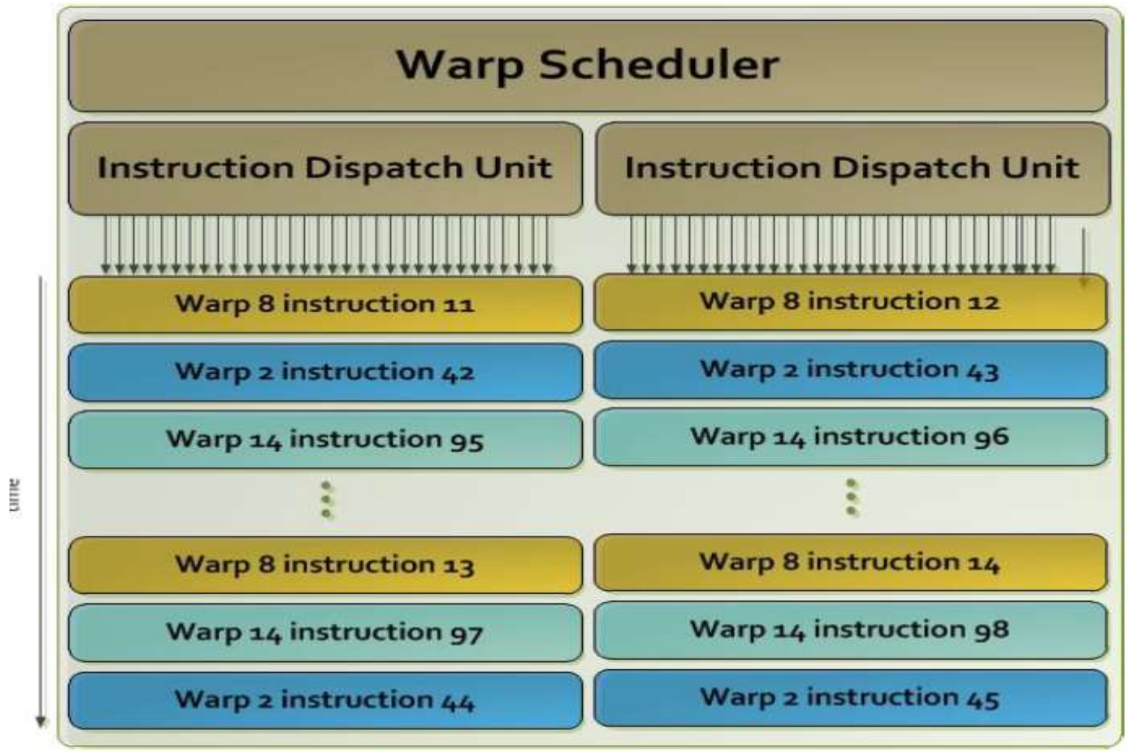

Fig. 2.12 Warp Scheduler Unit [7]

2.6.2 CUDA Memory

CUDA Memory plays very important role for performance point of view as the load and store time matters a lot for any computation. As CUDA has different types of memories having different characteristics, one can use them efficiently and optimization can be achieved. CUDA memory model is as shown in Fig.2.13 from, it contains different types of memories like Global Memory, Register Memory, Shared Memory, Local Memory, Texture Memory and Constant Memory. Each memory has its own benefits for speed execution and load-store time. These memory types and characteristics are shown in Table 1.

Table 2.1 CUDA Memory types with characteristics

Types of Memory	Position	Caches	Accessibility	Area
Register	On chip	No	W/R	One Thread
Local	On chip	Yes	W/R	One Thread
Shared	On chip	N/A	W/R	All threads in Block
Texture	Off chip	Yes	W/R	All host + threads
Constant	Off chip	Yes	R	All host + threads
Global	Off chip	Yes	W/R	All host + threads

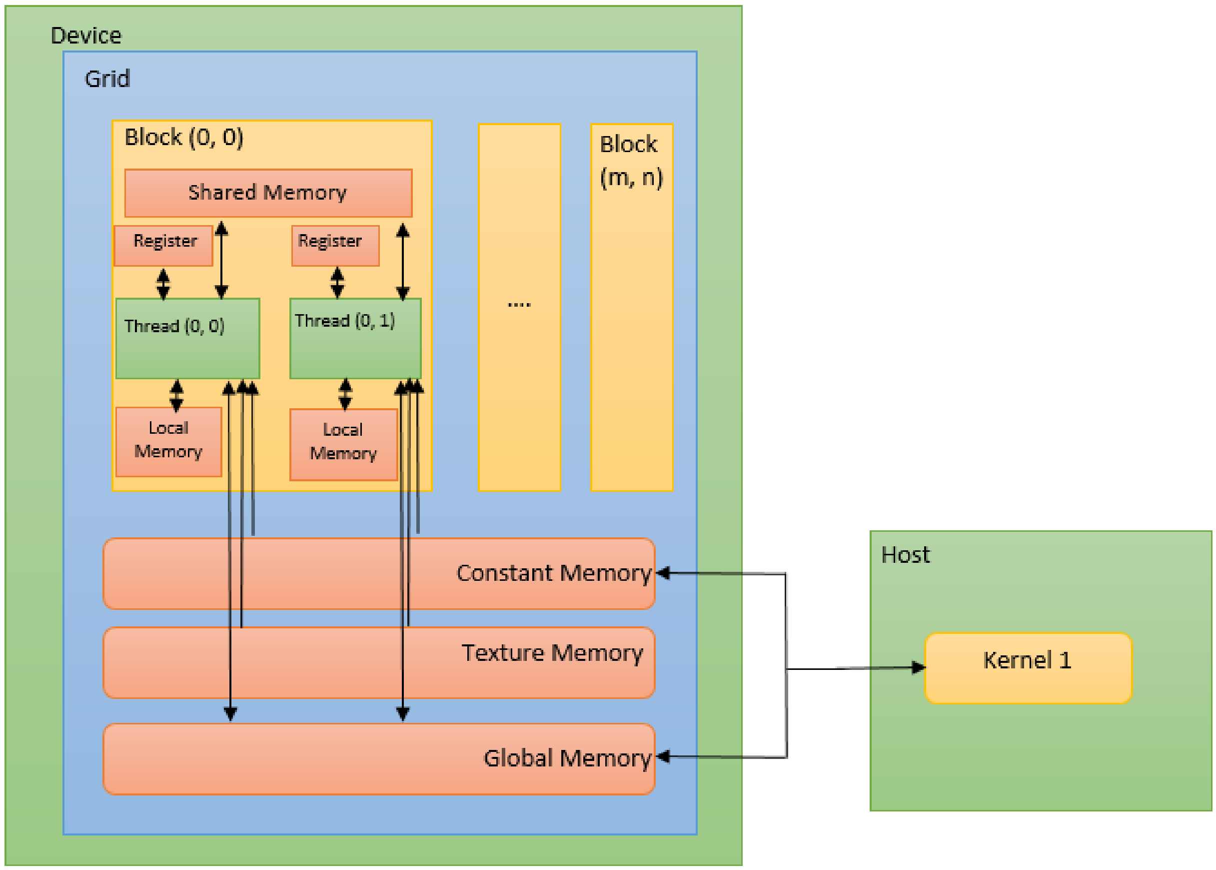

Fig. 2.13 CUDA Memory Model

2.6.2.1 Register Memory

Register Memory is the fastest memory on the GPU. It is on-Chip memory and having enough bandwidth to deliver peak performance. Each GK110 supports 65536 32 bit registers. The maximum number of registers which can be used by kernel is 63.

2.6.2.2 Local Memory

Local Memory is used when automatic variable is used. These automatic variable can be declared using **__shared__, __device__** or **__constant__** qualifier.Generally these automatic variable resides in registers except some cases like

I. Array size cannot be determined at the compile time.
II. Large structure that would consume too much space.

III. When kernel using too much register memory than available on SM which leads to register spill and data is saved in the local memory.

2.6.2.3 Shared Memory

Shared Memory is also known as **smem**, having size of 16KB to 48KB as per our requirement. It shared for only threads of one block. As L1 cache and Shared memory share 64KB memory, one can set the size of shared memory of 16KB, 32KB or 48KB. Shared Memory can be declared in three different ways.

I. In static number within the kernel or globally in the file
__shared__ signed short int sh_var[256];
II. Dynamically within the kernel by driver API function calling.
III. Dynamically via the execution configuration.

2.6.2.4 Constant Memory

Constant Memory is used to store and broadcast read-only data to all threads on GPU. Constant memory size is 64KB. When every thread in the warp size tries to access same address then this Constant Memory can be useful. It can be declared using **__constant__** qualifier. It resides inside the global memory. It is read only and does not depend on thread IDs.

2.6.2.5 Texture Memory

Texture Memory is off-Chip memory, resides in global memory and cached in Texture Cache. The texture cache is optimized for 2D spatial locality, so threads of the same warp that read texture or surface addresses that are close together in 2D will achieve best performance. Also, it is designed for streaming fetches with a constant latency, a cache hit reduces DRAM bandwidth demand but not fetch latency. It only occupies 8KB per SM.

Texture memory can be used for following scenarios:

I. If memory reads does not follow any access patterns
II. Addressing calculation is not done by the kernel
III. Packed data needs to be broadcasted to separate variables in single operations

2.6.2.6 Global Memory

Global Memory plays an important role in CUDA program as load-store from Global Memory affects too much on the performance of the Kernel in terms of time. Global Memory is the slowest memory of all type of memories. As global memory provides higher bandwidth, it should be used in such a way that bandwidth utilization should be done properly.

2.7 Literature Survey

2.7.1 Research Paper 1: Recommendation for space Data System [Image Data Compression] [1]

This Recommended Standard provides the Compression method for Image data. It has explained Lossy Compression as well as Lossless Compression. ’Integer 9/7 DWT is Lossless Compression whereas ‘Float 9/7 DWT’ is Lossy Compression. The Compressor consists of two functional parts as shown in Fig. 2.14.

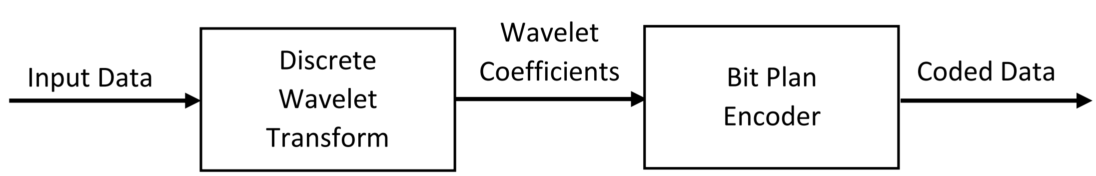

Fig. 2.14 General Schematic of coder

This Recommended Standard for the decorrelation module makes use of a three-level, two dimensional (2-d), separable Discrete Wavelet Transform (DWT) with nine and seven taps for low- and high-pass filters, respectively. Such a transform is produced by repeated application of a one-dimensional (1-d) DWT.

- **9/7 Integer Discrete Wavelet Transform**

Given input values of x_i , the D_j values in equation 2 shall be computed first and used subsequently to compute C_j values in equation 6. Where D is high pass values and C is low pass values.

$$D_0 = x_1 - \left\lfloor \frac{9}{16}(x_0+x_2) - \frac{1}{16}(x_2+x_4) + \frac{1}{2} \right\rfloor \qquad (1)$$

$$D_j = x_{2j+1} - \lfloor \tfrac{9}{16}(x_{2j}+x_{2j+2}) - \tfrac{1}{16}(x_{2j+2}+x_{2j+4}) + \tfrac{1}{2} \rfloor \qquad \text{for } j=1,\ldots,N-3 \tag{2}$$

$$D_{N-2} = x_{2N-3} - \lfloor \tfrac{9}{16}(x_{2N-4}+x_{2N-2}) - \tfrac{1}{16}(x_{2N-6}+x_{2N-2}) + \tfrac{1}{2} \rfloor \tag{3}$$

$$D_{N-1} = x_{2N-1} - \lfloor \tfrac{9}{8} x_{2N-2} - \tfrac{1}{8} x_{2N-4} + \tfrac{1}{2} \rfloor \tag{4}$$

$$C_0 = x_0 - \lfloor - \tfrac{D0}{2} + \tfrac{1}{2} \rfloor \tag{5}$$

$$C_j = x_{2j} - \lfloor - \tfrac{D_{j-1} + Dj}{4} + \tfrac{1}{2} \rfloor \qquad \text{for } j=1,\ldots, N-1 \tag{6}$$

Image decorrelation is accomplished using a two-dimensional DWT, which is performed by iterated application of the one-dimensional DWT. Viewing the image as a data matrix consisting of rows and columns of signal vectors, a single-level 2-d DWT shall be performed on the image in the following two steps in the following order:

a) the 1-d DWT shall be performed on each image row, producing a horizontally lowpass and a horizontally high-pass filtered intermediate data array, each half as wide as the original image array, as illustrated in figure 2.15(b)

b) the 1-d DWT shall be applied to each column of both intermediate data arrays to produce four subbands as shown in figure 2.15(c).

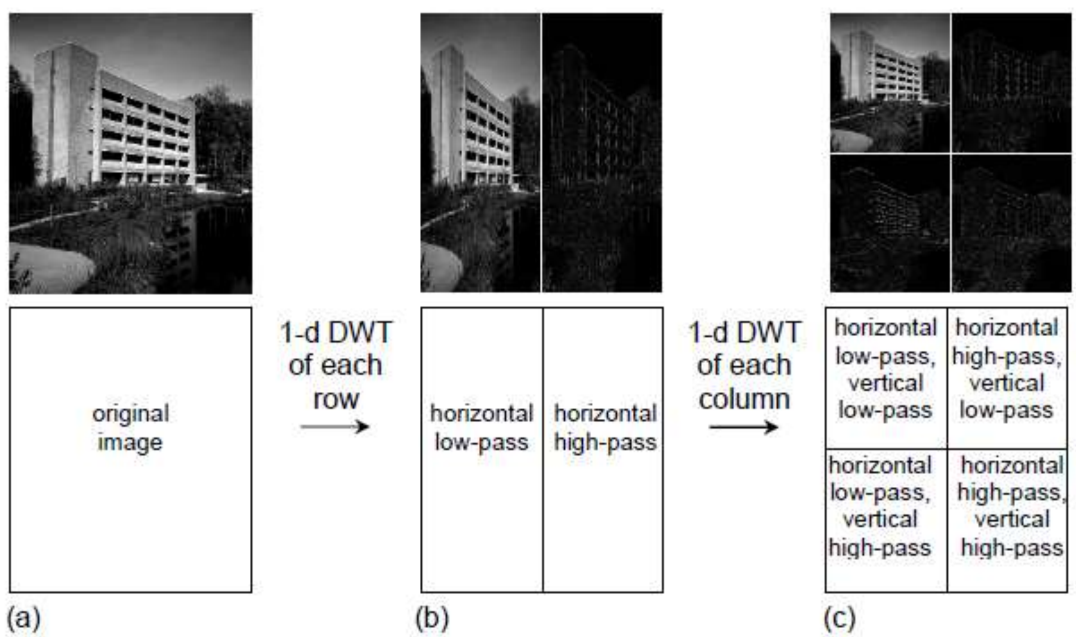

Fig. 2.15 One level 2D DWT [1]

To increase compression effectiveness, correlation remaining in the LL subband after the 2-d DWT decomposition is exploited by applying further levels of DWT decomposition to produce a multi-level 2-d DWT.

This Recommended Standard specifies three levels of decomposition. At each level, the 2-d DWT described in Fig.2.15 shall be applied to the LL subband produced by the previous level of decomposition.

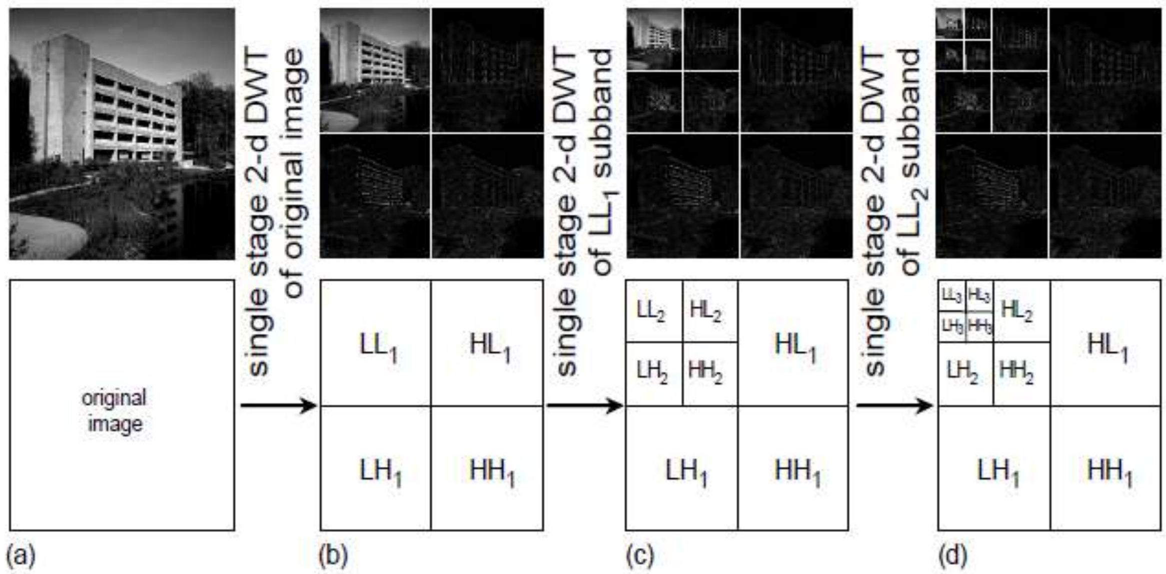

Fig. 2.16 Three level 2D DWT [1]

Output from the DWT will look like as shown in Fig. 2.17

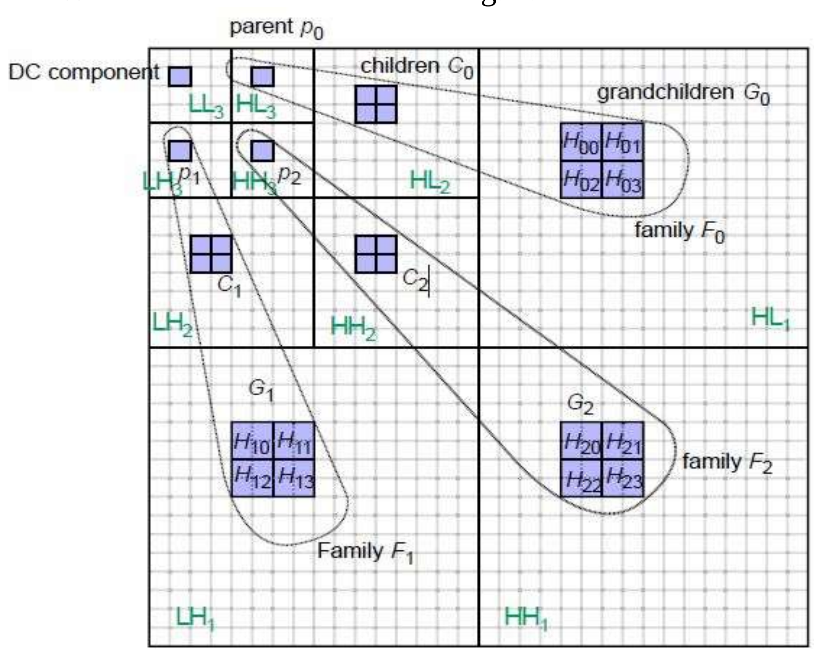

Fig. 2.17 Schematic Wavelet Transformed Image [1]

- **Bit Plan Encoder**

Schematic Wavelet Transformed Image will be given to bit plan encoder as input and it processes wavelet coefficients in groups of 64 coefficients as shown in Fig. 12 referred to as blocks. This block will contain one DC Coefficient and 63 AC Coefficients. Each family F_i in the block has one parent coefficient, p_i, a set C_i of four children coefficients, and a set G_i of sixteen grandchildren coefficients.

- **Summary**

In this Recommended Standard, compression system for image data is provided. In the compressor system consists of two modules, Discrete Wavelet Transform (DWT) and Bit Plan Encoder.

Discrete Wavelet Transform (DWT) process is such that we can implement it in parallel whereas Bit Plan Encoder can only be done in sequential manner as previous stage's result is going to be used in next stage's computation.

2.7.2 Research Paper 2: A GPU-Accelerated Wavelet Decompression System with SPIHT and Reed-Solomon Decoding for satellite Images [2]

This paper discusses the decoding system for satellite images with DWT, SPIHT and Reed-Solomon. The decoding system overview [2] is as shown in below Fig. 2.18.

Fig.2.18 Overview of decoding System [2]

In this paper 9/7 Float Inverse Discrete Wavelet Transform is implemented on GPU because CPU time for DWT is the most time consuming part as shown in Table 2.

Table 2.2 [2]

CPU TIME OF EACH COMPONENT OF THE DECODING SYSTEM

RS decoding	18 ms
SPIHT decoding	62 ms
IDWT	841 ms

Here the approach for Inverse DWT contains horizontal filter with shared memory which is doing block by block computation and transpose is also done using shared memory block by block. In the improvement of this vertical filter, they are not again reading overlap but they are performing coalesced access so that memory transfer time is concealed as they are reading other large amount of data in parallel. They got better result in their proposed method as shown in Fig. 2.19.

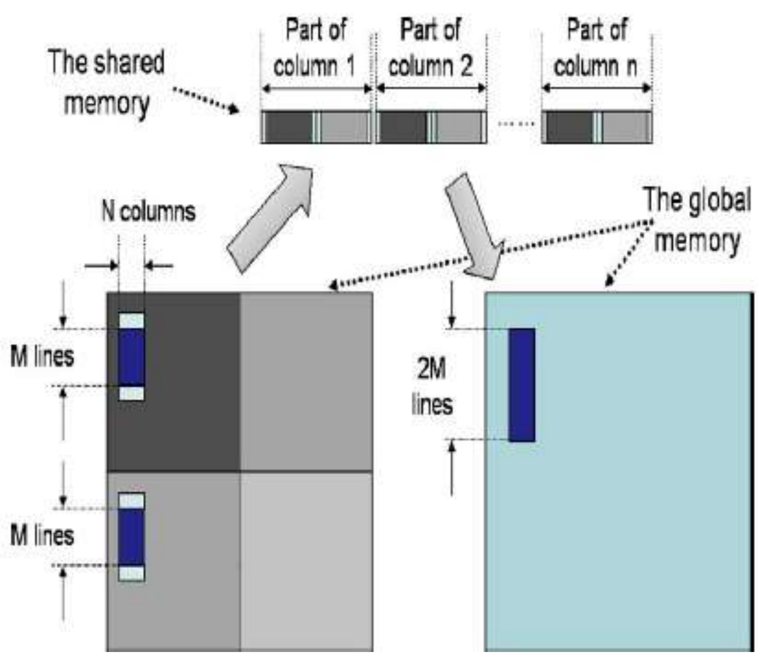

Fig. 2.19 Block Segmentation approach [2]

Finally, they have used CPU-GPU pipeline for whole decoding system as shown in Fig. 2.20

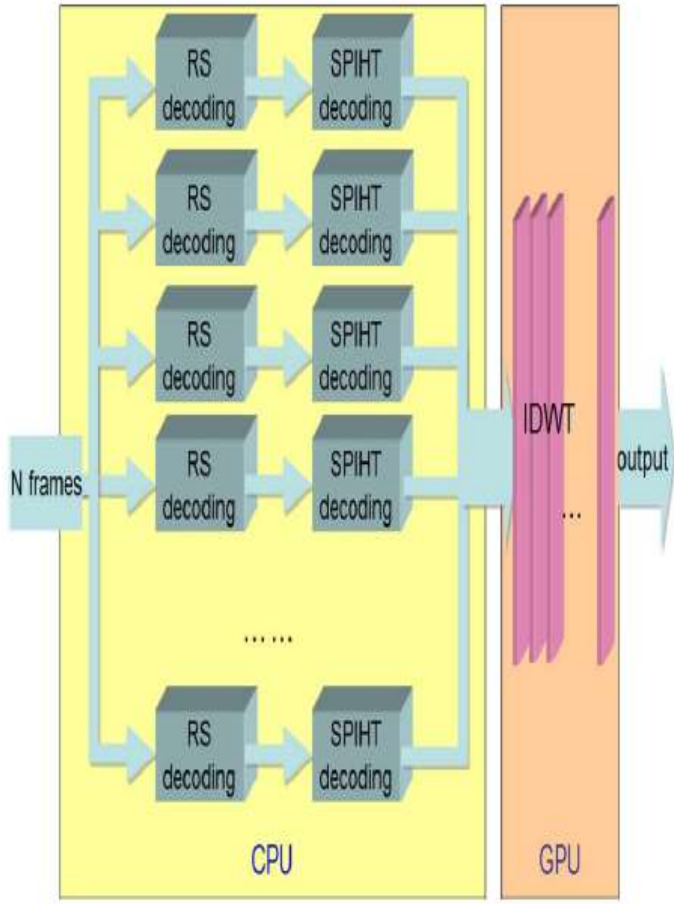

Fig. 2.20 Overview of RS+SPIHT+IDWT Decoding System [2]

- **Summary**

This paper has implemented decompressor for 9/7 Float Discrete Wavelet Transform with SPIHT and Reed Solomon technique. For the IDWT, they have used horizontal filter, vertical filter and transpose of the matrix using shared memory. Further improvement has also been made on the vertical filter. Decoding system is implemented using CPU-GPU pipeline fashion to accelerate time consumption part (9/7 Float IDWT).

2.7.3 Research Paper 3: Time efficiency comparison of wavelet and inverse Wavelet transform on different platforms [3]

This paper compares time efficiency of integer 9/7 DWT as well as Integer 9/7 Inverse DWT on three different platforms. These platforms include MATLAB, Python-PIL and Python – OpenCV. Fig. 2.21 shows the time comparison chart for all of these platforms. As shown in the Fig. 17 Python – PIL requires longer time to process the same data which is processed much faster on other two platforms.

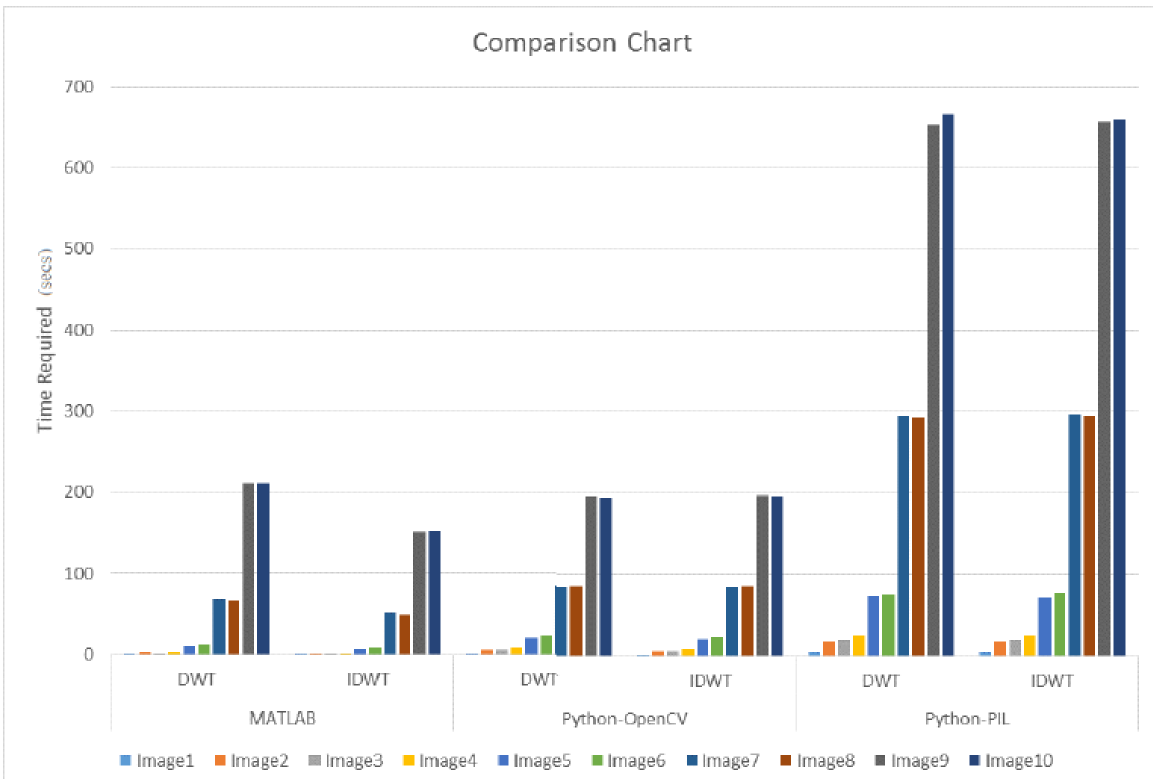

Fig.2.21 Comparison chart [3]

- **Summary**

In this paper, Python-OpenCV gives best results in terms of time for 9/7 Integer DWT. Python is open source tool and coding Complexity of Python and MATLAB is nearly same so use of Python-OpenCV is recommended here.

Further, this implementation can be done using NVIDIA, CUDA.

2.7.4 Research Paper 4: Efficient parallelization of the Discrete Wavelet Transform algorithm using memory - oblivious Optimization [4]

This paper discusses 9/7 Integer Discrete Wavelet Transform on two different systems. They have implemented 9/7 Integer DWT on two different CPUs, AMD Phenom II and Intel Xeon E5-2680 respectively. Also these implementation is done on the Accelerators, NVIDIA Tesla C2070 (Fermi) and Intel Xeon Phi 5110P. Table 2.3 shows the timing results for different Image Sizes for each system. As shown in Fig. 2.22 [4] NVIDIA Tesla gives the best performance from all of the CPUs and GPUs.

Table 2.3 Execution time for DWT on different Platforms [4]

	256x256	512x512	1024x1024	2048x2048	4096x4096	8192x8192
Phenom ser.	5.7	11.7	66.4	303.4	1348.6	7809.8
Phenom opt.	1.9	4.5	20.4	55.4	197.6	740.6
Tesla	0.2	0.5	1.2	3.9	14.6	58.4
Xeon E5 ser.	1.0	3.6	24.6	108.2	546.3	2589.6
Xeon E5 opt.	1.2	2.3	6.4	16.3	47.2	144.8
Xeon Phi	7.8	28.8	60.1	130.2	263.7	590.1

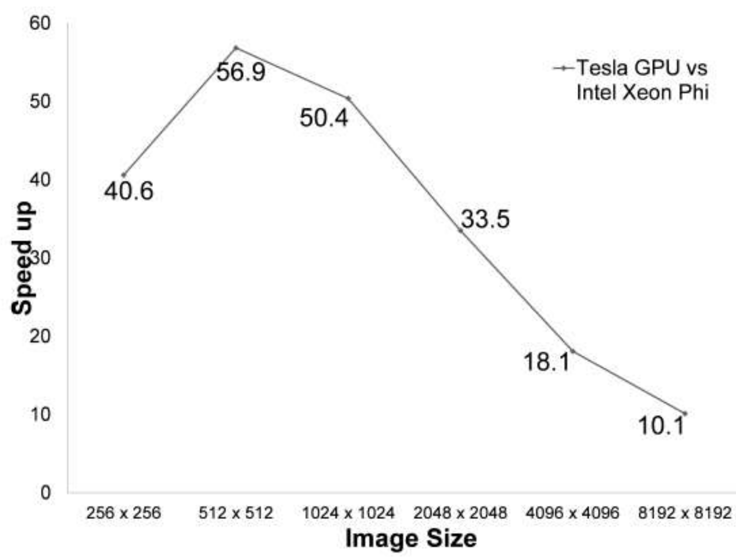

Fig.2.22 Tesla GPU vs Intel Xeon Phi overall speedup diagram [4]

- **Summary**

For time consuming process like Discrete Wavelet Transform, we can use GPU for parallelization and can achieve speedup. From all of the four systems, AMD Phenom II, Intel Xeon E5-2680, NVIDIA Tesla C2070 (Fermi) and Intel Xeon Phi 5110P, NVIDIA Tesla gives best results.

Table 2.4 Literature Review Summary

Paper No.	Title	Merits	Demerits
1	Image Data Compression	Saves bandwidth and also saves storage space	IDWT Process is time consuming so Decompressor takes long time
2	A GPU-Accelerated Wavelet Decompression System With SPIHT and Reed-Solomon Decoding for Satellite Images	Efficient decompressor is made using GPU and shared memory using transpose and Block Segmentation Approach	-
3	Time Efficiency Comparison of Wavelet and Inverse Wavelet Transform on Different Platforms	DWT and IDWT is performed on different Open Source platforms like MATLAB, Python -PIL, Python –OpenCV.	This time consumption can be still reduced using NVIDIA, CUDA
4	Efficient parallelization of the Discrete Wavelet Transform algorithm using memory-oblivious optimizations	9/7 Integer DWT has been implemented and compared efficiently on different CPU and GPU platforms	-

In case of CCSDS-122.0-B-1 [1], its decoding system consists of 9/7 Integer Discrete Wavelet Transform (DWT) and Bit Plan Encoder. As described by [2], [3], [4] we can say that 9/7 Integer Discrete Wavelet Transform (DWT) is the most time consuming process in the decompressor. To make efficient Decompressor we have to implement 9/7 Integer Discrete Wavelet Transform (DWT) computation in parallel. 9/7 Integer DWT has been implemented in [3] and [4] on GPU and MATLAB or Python respectively. 9/7 Float IDWT has been implemented on GPU as described in [2].

Abhishek S. Shetty at el. [3] proposed that 9/7 Integer Inverse DWT can be implemented on NVIDIA GPU for better results. Whole decoding system can be implemented using CPU-GPU pipeline fashion just like done in [2]. Thus, efficient Decompressor for CCSDS-122.0-B-1 can be implemented through CPU-GPU pipeline. In proposed work, Tesla K40C Accelerator is going to be used. Its architecture, memory details can be found in [7].

2.7.5 Research Paper 5: Recommended standard for space data system Lossless Multispectral and Hyperspectral Image Compression] [9]

This recommended paper explains the coding system for multispectral and hyperspectral image data as shown in Fig. 2.23.

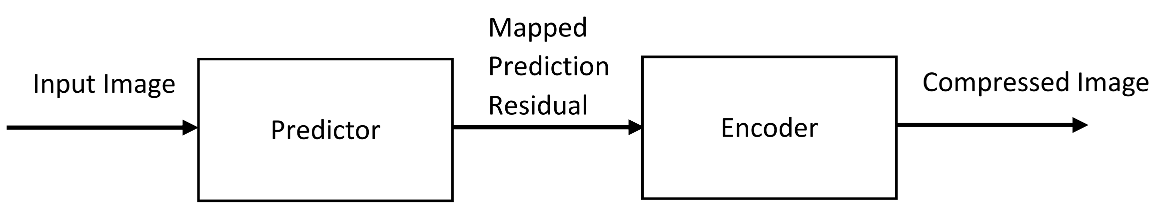

Fig. 2.23 Compressor Schematic [9]

In this compressor system, Predictor as well as Encoder consists various parameters. Different Compression results can be achieved by varying these parameters. These Compression parameters consists of number of Prediction bands, Local Sum type, Prediction modes, local diff vector types, register sizes, weight component resolution etc.

- **Summary**

CCSDS-123.0-B-1 compression system consists predictor and encoder which is controlled by specific parameters like number of prediction bands, register size, weight component resolution etc. By tuning the compression parameters best compression can be achieved for different type of specific sensors for multispectral and hyperspectral image data.

2.7.6 Research Paper 6: Multispectral and Hyperspectral Lossless Compressor for Space Application (HyLoc): A Low – Complexity FPGA Implementation of the CCSDS 123 Standard [10]

This paper has built one hardware architecture with aim of achieving low hardware occupancy and high performance on space-qualified FPGA from Microsemi RTAX family. Hardware is implemented in such a way that effect of CCSDS-123 configuration parameters on the compression efficiency and hardware complexity provides flexibility so it can be adopted for different application scenarios. Table 2.5 shows one of the results of parameters on the compression ratio.

Table 2.5 Effects of compressor parameters on compression ratio [10]

Parameter	Default in empordá	Conclusions
Number of bands for prediction (P)	15	Higher P yields better compression, however setting $P > 3$ does not show improvement.
Prediction mode	Full	Raw images $\rightarrow$ Reduced+Column; Calibrated images $\rightarrow$ Full+Neighbor
Local sum mode	Neighbor	
Register size	32	Does not have a significant impact. It has to be large enough to prevent overflow.
Weight component resolution (Ω)	13	Has a noticeable impact. A large Ω yields more compression.
Weight update scaling exponent change interval (t_{inc})	6	Does not have a significant impact.
Weight update scaling exponent initial parameter (ν_{min})	-1	Does not have a significant impact.
Weight update scaling exponent final parameter (ν_{max})	3	Has a moderate impact. In general, better compression is achieved for higher ν_{max}.

- **Summary**

The compression of multispectral and hyperspectral image depends on various parameters like number of prediction bands, weight component resolution, register size, prediction mode. By tuning the compression parameters best compression can be achieved for different type of sensors for multispectral and hyperspectral image data.

Thus, in case of CCSDS-123.0-B-1 [9], to achieve good compression for different types of sensors we have to tune the compression parameters. These compression parameters contain Prediction bands, Local Sum type, Prediction modes, local difference vector types, register sizes, weight component resolution etc.

Chapter 3

Proposed Methodology

3.1 Problem Statement

CCSDS-122.0-B-1 based decompressor is real time need which can decompress the data as soon as massive streams of bits downlinked on the earth. This decompressor consists of Inverse DWT and Bit plane decoder, but here as the IDWT is the most time consuming process, to make efficient decompressor, IDWT process should be speeded up.

CCSDS-123.0-B-1 Recommended Standard defines the compression method for multispectral and hyperspectral image data. The compressor consists of two functional parts, Predictor and Encoder. These Predictor and Encoder is dependent on specific parameters. By tuning these parameters, one can get best compression for multispectral and hyperspectral images for different types of sensors.

3.2 Proposed Algorithm Work Flow Diagram for CCSDS-122.0 (Part -1)

As explained in above Problem Statement, to accelerate the IDWT process in the Decompressor, it should be implemented on GPU. In our proposed work, we are going to use NVIDIA Tesla K40c Accelerator. Work Flow of the Proposed Algorithm for CCSDS-122.0-B-1 is shown in Fig. 3.1.

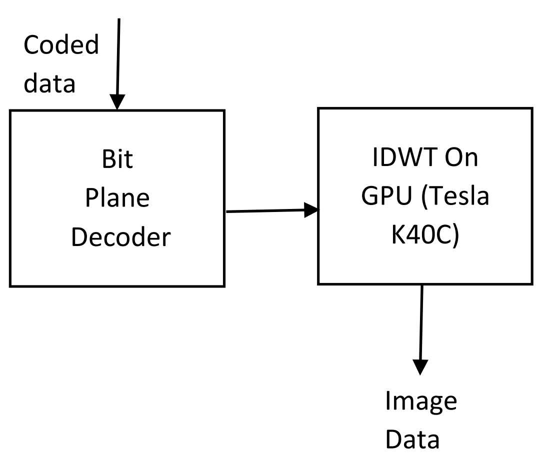

Fig. 3.1 CCSDS-122.0-B-1 Work Flow

3.3 Highest Level IDWT Work Flow Diagram

Steps to be followed for computing one level Inverse Discrete Wavelet Transform is written as below:

Level 3 Decomposition to determine LL2 Subbands from highest level subbands LL3, HL3, LH3, HH3

- Combine subbands LL3 and HL3 using row wise inverse filter and store the output as L3
- Combine subbands HL3 and HH3 using row wise inverse filter and store the output as H3
- Combine intermediate outputs L3 & h3 using column wise inverse filter and store the Output in Image_level3.

- ### Mapping of Highest level IDWT Flow Diagram on GPU

```
    INITIALIZATION:

-   Allocation of Memory on the Host (CPU) Side

-   Allocation of Memory on the Device (GPU) Side


             for i = 1:3

              copy of subbands from CPU to GPU
              Compute Kernel
              storing of intermediate on global memory

             end

-   copy final result to CPU
```

After this process, Bit Plane encoder can be implemented serially as one level encoding uses previous level's output for computation. Thus, whole decoding System can be implemented in CPU-GPU pipeline fashion which is shown in Fig.3.2.

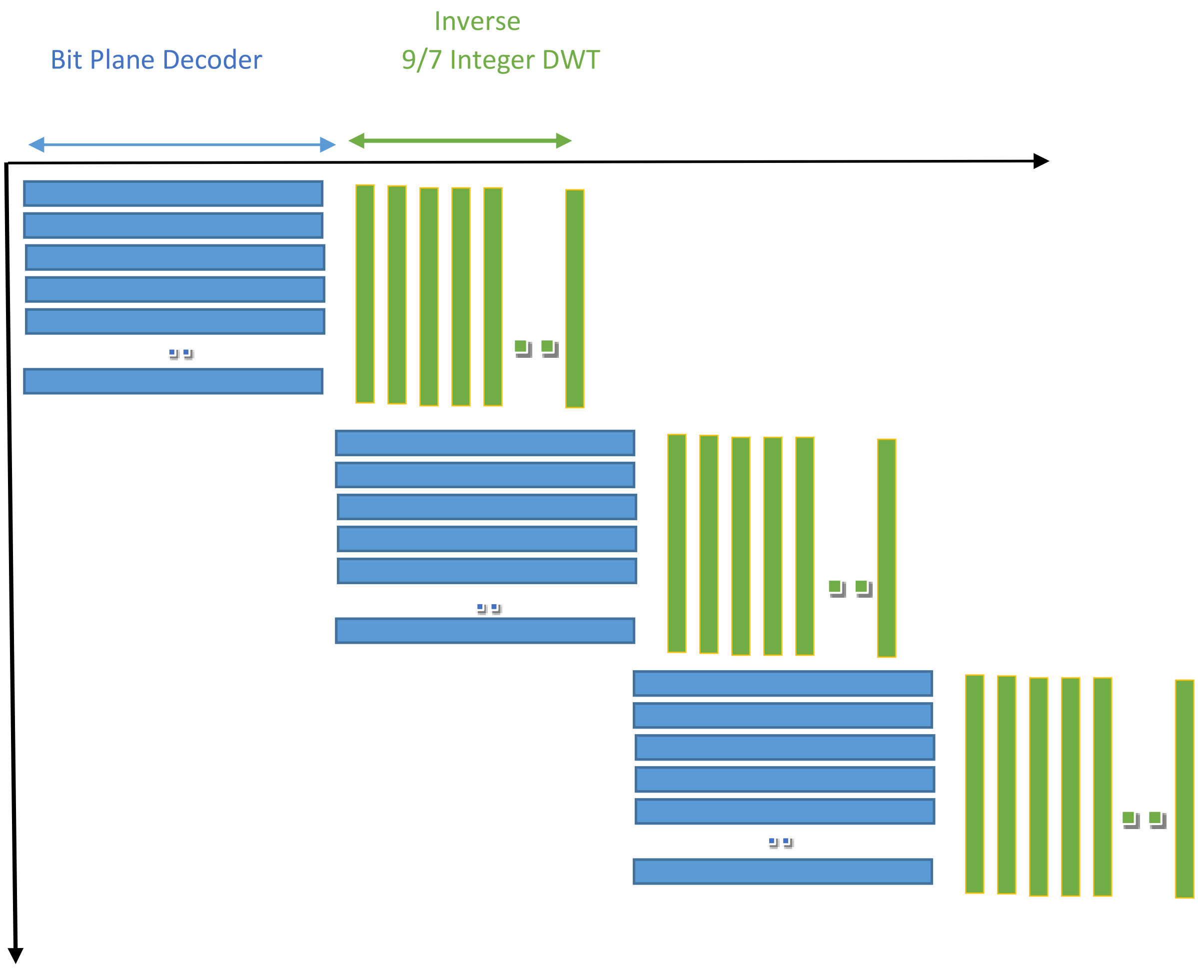

Fig. 3.2 CPU-GPU Pipeline Decompression System

3.4 CCSDS-123.0 Compression Algorithm Work Flow Diagram (Part-2)

The workflow of algorithm for the CCSDS-123.0 Compression is shown in Fig. 3.3.

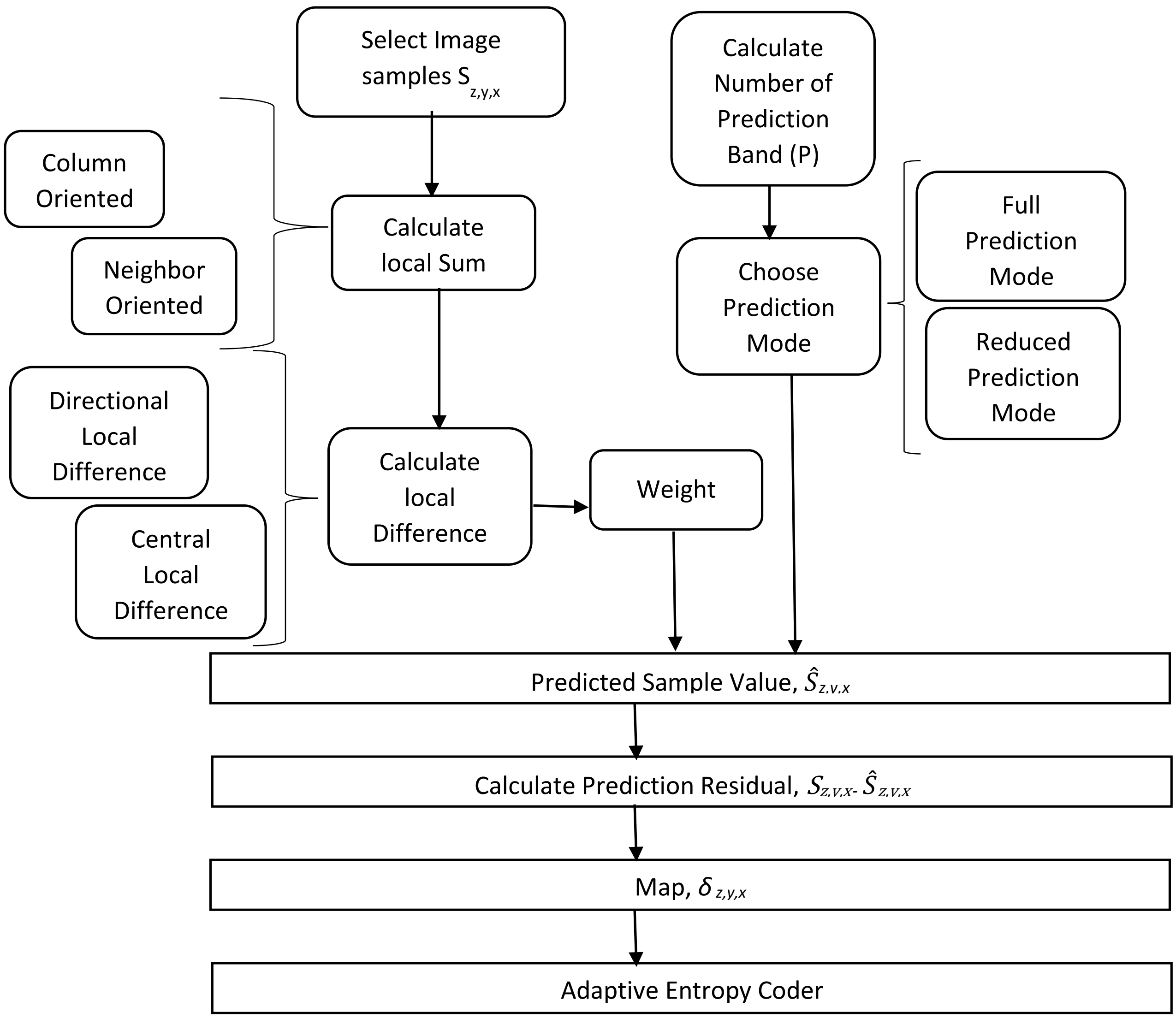

Fig. 3.3 Workflow of CCSDS 123 algorithm

- **Methodology of Proposed Work:**

  Step 1: Select the sample image value, $s_{z,y,x}$ and choose number of prediction bands

  Step 2: calculate local sum and choose prediction mode.

  Step 3: Calculate local difference.

  Step 4: Calculate weight

  Step 5: Calculate predicted sample value.

  Step 6: Calculate prediction Residual

  Step 7: Map the value.

  Step 8: Give mapped value to Adaptive Entropy coder.

Chapter 4

Implementation Details

4.1 Software and Tools used for CCSDS-122.0 Decompressor (Part -1)

Implementation of proposed system is done using the coding and their results are compared with different versions of the system itself. This experiment is carried out at the following experimental set up.

Platform Specification

- **Platform** : Microsoft Windows
- **Edition** : Windows Server 2012 R2 Standard
- **System type** : 64 – bit Operating System
- **RAM** : 32.00 GB
- **Processor** : Intel(R) Xeon(R) CPU E5-2650 v3 @2.30 GHz

C Language

C is a general-purpose, imperative computer programming language. It supports structured programming, lexical variable scope and recursion, while a static type system prevents many unintended operations. By design, C provides constructs that map efficiently to typical machine instructions, and therefore it has found lasting use in applications that had formerly been coded in assembly language, including operating systems, as well as various application software for computers ranging from supercomputers to embedded systems.

C was designed to be compiled using a relatively straightforward compiler, to provide low-level access to memory, to provide language constructs that map efficiently to machine instructions and to require minimal run-time support. C was therefore useful for many applications that had formerly been coded in assembly language, such as in system programming.

CUDA Platform

CUDA is a general-purpose parallel computing platform and programming model that leverages the parallel compute engine in NVIDIA GPUs to solve many complex computational problems in a more efficient way. Using CUDA, you can access the GPU for computation, as has been traditionally done on the CPU. The CUDA platform is accessible through CUDA-accelerated libraries, compiler directives, application programming interfaces, and extensions to industry-standard programming languages, including C, C++, Fortran, and Python.

Microsoft Visual Studio 2012

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, web sites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce native code as well as managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source control systems and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle.

NVIDIA Nsight 5.4

NVIDIA Nsight is the ultimate development platform for heterogeneous computing. This powerful debugging and profiling tool enables you to fully optimize the performance of the CPU and GPU. It does not only do optimize performance but also they help you to gain a better understanding of your code - identify and analyse bottlenecks and observe the behaviour of all system activities.

NVIDIA Nsight visual studio edition includes Graphics Debugger, CUDA Profiler, Direct3D (including Direct Compute dispatches) and OpenGL Frame Profiler. Direct 3D, OpenGL, and Vulkan frame debugger with render state and draw call inspection is a part of Graphics Debugger. CUDA Profiler consists of Visual and command line interfaces to collect counters, statistics, and derived values for specified CUDA kernel launches. Customizable reports provide results, source and disassembly views, memory throughput diagrams, and execution flow charts as well as Unlimited experiments on live kernels. It has powerful visualization capabilities which shows the pre-transformed geometry from the state of Direct3D or OpenGL machine.

- **Bit Plan Decoder**

  Bit Plane Decoder decodes coded data & generates the wavelet coefficient. Different decoding techniques are applied to decode the AC coefficient and DC coefficient. The coded data contains segment headers followed by DC & AC coefficients. Compressed segment contains following header sequence:

  1. Part 1 (three or four bytes)
  2. Part 2 (five bytes)
  3. Part 3 (three bytes)
  4. Part 4 (eight bytes)

Following figure 4.1 shows overview of the header structure when all parts are included.

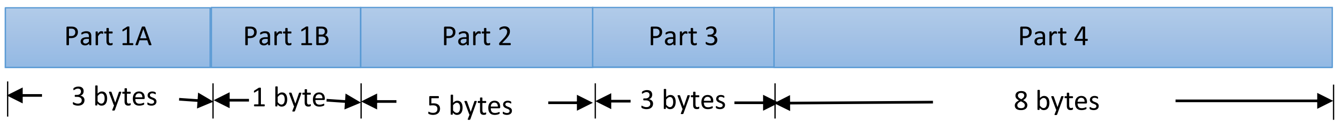

Fig. 4.1 Overview of the header structure when all headers parts are included

There is different coded data format for Uncoded data option & coded data option. This Uncoded and Coded data format is as shown in fig 4.2 & 4.3 respectively. By observing coded data format as well as uncoded data format, it is clear that length of gaggle may vary as it depends on the value of code option, k. It is bit operation with variable length reading which cannot be parallelized. It decodes block by block. Block contains 16 gaggles. If it is 1st block of segment, then 1st gaggle of block contains one reference sample and 15 mapped samples. Rice Decoding is being used for the decoding of DC coefficient. The flow of the Rice decoding is as shown in fig. 4.4.

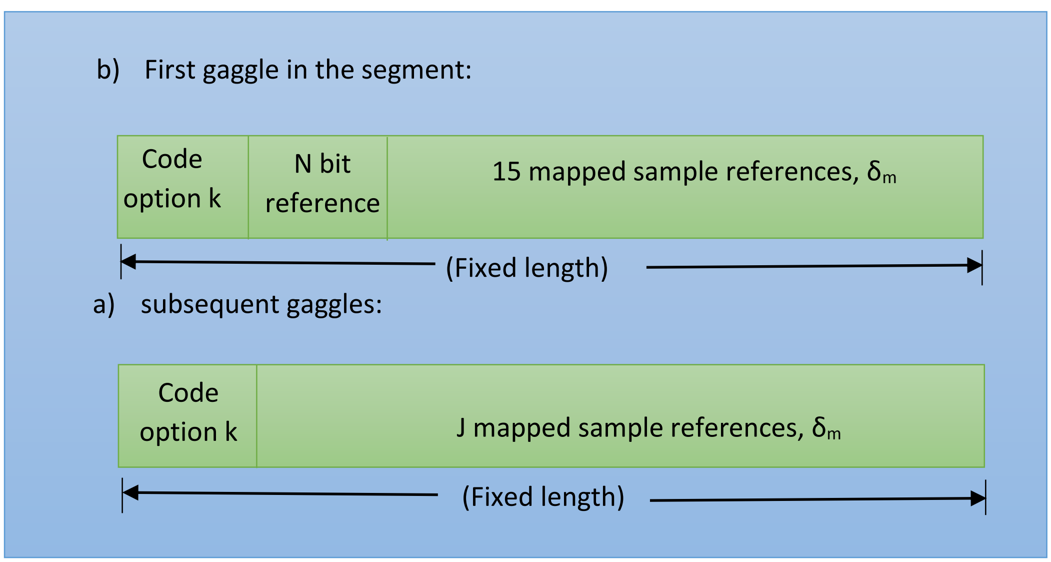

Fig. 4.2 Coded data format for a gaggle when uncoded data option is used

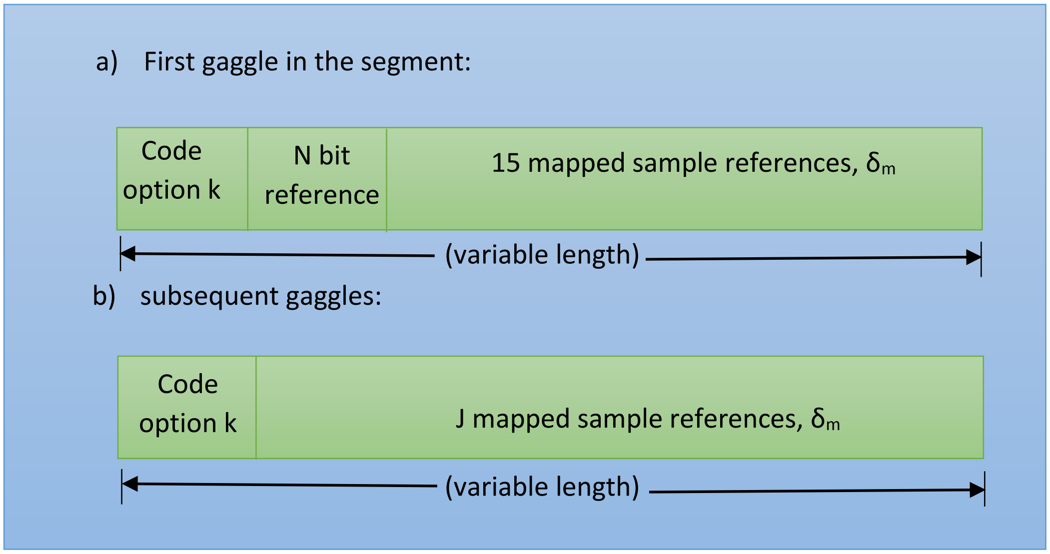

Fig. 4.3 Coded data format for a gaggle when a coding option is selected

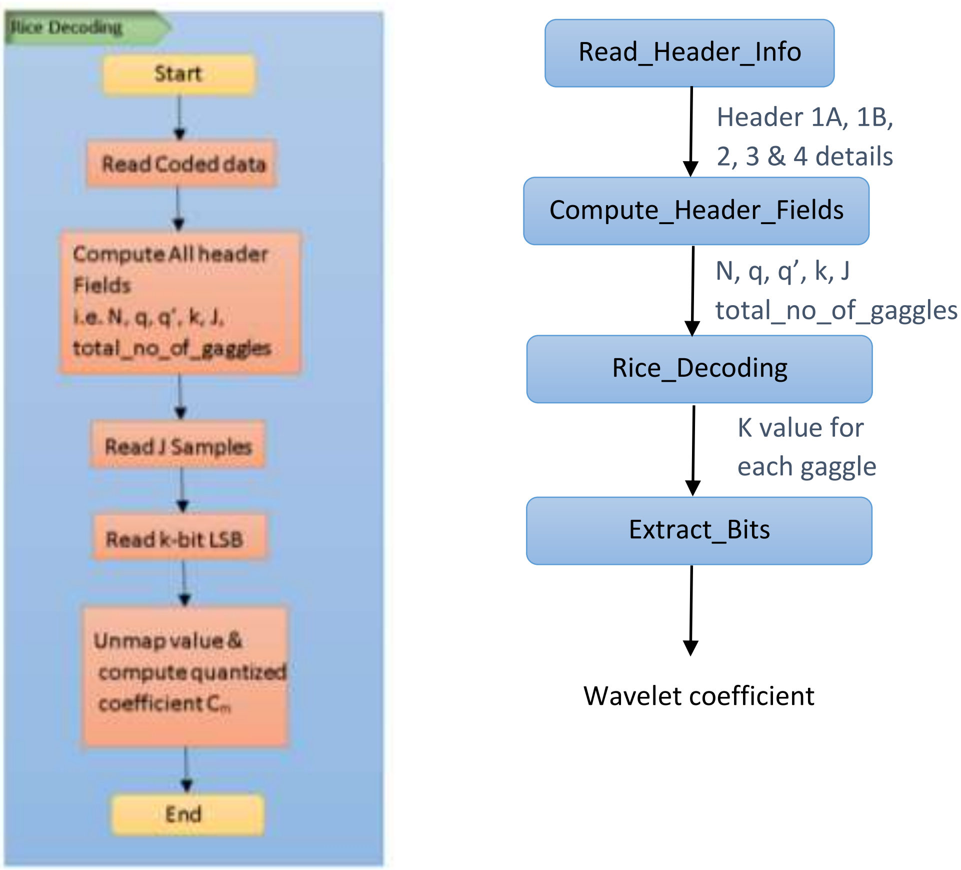

Fig. 4.4 Rice Decoding Flow Chart & functional hierarchy

- **Inverse Discrete Wavelet Transform**

The Inverse Discrete Wavelet Transform (IDWT) can be considered as correlation module, as with every level of transform data gets correlated. This standard module uses of a three-level two dimensional (2-d) IDWT which passes through nine low pass and seven high pass filters. Basically two Inverse DWTs are specified in the CCSDS Recommended Standard [1], ‘Inverse 9/7 Integer DWT’ and ‘Inverse 9/7 Float DWT’. ‘Inverse 9/7 Integer DWT’ is Lossless Decompression whereas ‘Inverse 9/7 Float DWT’ is Lossy Decompression as it uses quantization. We are going to focus on Lossless Decompression which is Inverse 9/7 Integer DWT.

The inverse Integer DWT maps two sets of wavelet coefficients [1], a low-pass set, C_j, and a high-pass set, D_j, back to a signal vector x_j. Special boundary filters are required at either end of the data sets, for $j=0$, $j=1$, $j=2N-3$, and $j=2N-1$.

$$x_1 = D_0 + \left\lfloor \frac{9}{16}(x_0+x_2) - \frac{1}{16}(x_2+x_4) + \frac{1}{2} \right\rfloor \tag{7}$$

$$x_{2j+1} = D_j + \left\lfloor \frac{9}{16}(x_{2j}+x_{2j+2}) - \frac{1}{16}(x_{2j-2}+x_{2j+4}) + \frac{1}{2} \right\rfloor \quad \text{for } j=1,\dots,N-3 \tag{8}$$

$$x_{2N-3} = D_{N-2} + \left\lfloor \frac{9}{16}(x_{2N-4}+x_{2N-2}) - \frac{1}{16}(x_{2N-6}+x_{2N-2}) + \frac{1}{2} \right\rfloor \tag{9}$$

$$x_{2N-1} = D_{N-1} + \left\lfloor \frac{9}{8}x_{2N-2} - \frac{1}{8}x_{2N-4} + \frac{1}{2} \right\rfloor \tag{10}$$

$$x_0 = C_0 + \left\lfloor -\frac{D0}{2} + \frac{1}{2} \right\rfloor \tag{11}$$

$$x_{2j} = C_j + \left\lfloor -\frac{D_{j-1}+Dj}{4} + \frac{1}{2} \right\rfloor \quad \text{for } j=1,\dots, N-1 \tag{12}$$

The single-level 2-d DWT transform [1] shall be inverted by repeated application of the 1-d inverse to columns and rows of the transformed data array in the reverse order to that in which the 1-d transforms were applied:

a) each column shall be inverted to produce the intermediate transformed data arrays:
1) the 1-d DWT inverse shall be applied to columns of the LL and LH subbands to obtain the intermediate horizontal low-pass array,
2) the 1-d DWT inverse shall be applied to columns of the HL and HH subbands to obtain the intermediate horizontal high-pass array;
b) the 1-d DWT inverse shall be applied to rows of the intermediate horizontal low-pass and horizontal high-pass arrays to recover the original image array.

The inversion process of a multi-level DWT shall be as follows [1]:
a) The four subbands of highest level, LL3, LH3, HL3, HH3, shall be inverted using an inverse single-level 2-d DWT to yield the single subband LL2, which then replaces the higher-level subbands in the transform data matrix;

b) The four subbands LL2, LH2, HL2, HH2 shall be inverted to yield the single subband LL1, which again replaces the higher-level subbands in the transform data matrix;
c) A final single-level 2-d inverse DWT shall be applied to subbands LL1, LH1, HL1, HH1 to reproduce the original image.

The flow of Decompressor system is as shown in the fig. 4.5

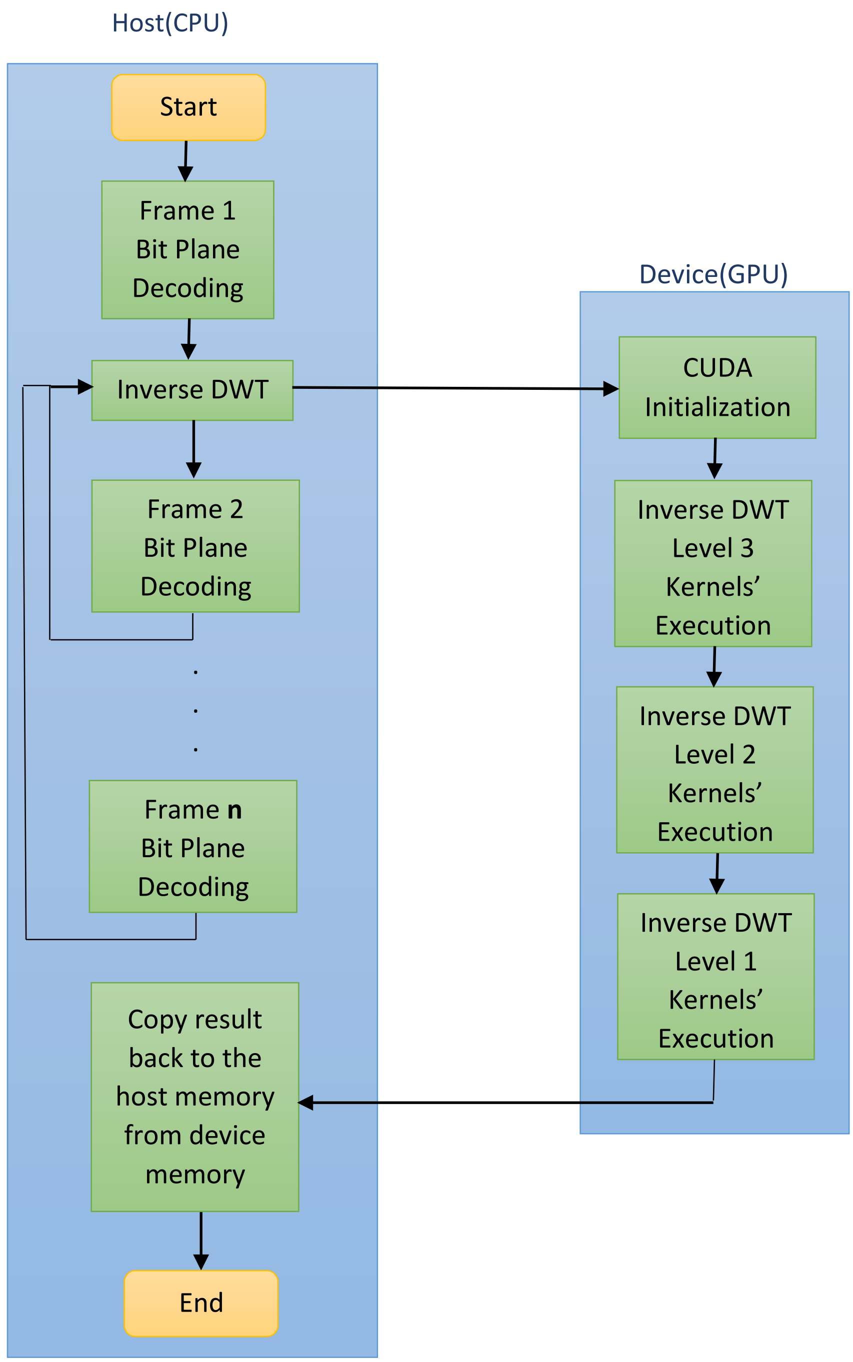

Fig. 4.5 Work Flow of Decompressor system

4.2 Different Versions of Inverse DWT (Part -1)

This sub section contains implementation details of different versions of Inverse Discrete Wavelet Transform which is implemented on GPU.

4.2.1 Decompressor_Unoptimized

This version of decompressor takes the input image from binary file and then transfers this data from CPU to GPU. CPU launches number of threads same as number of pixels of an image for computation on GPU. There are total 4 kernels for computation of the Inverse 9/7 Integer Discrete Wavelet Transform. In this version, L2 cache is only used whereas L1 cache is disabled.

All compressed image data is copied in the global memory & result is also stored back into the global memory. IDWT_Kernel_1 and IDWT_Kernel2 is for computation of horizontal filter. IDWT_Kernel_3 and IDWT_Kernel_4 is for vertical filter computation which generates one level 2D inverse DWT image.

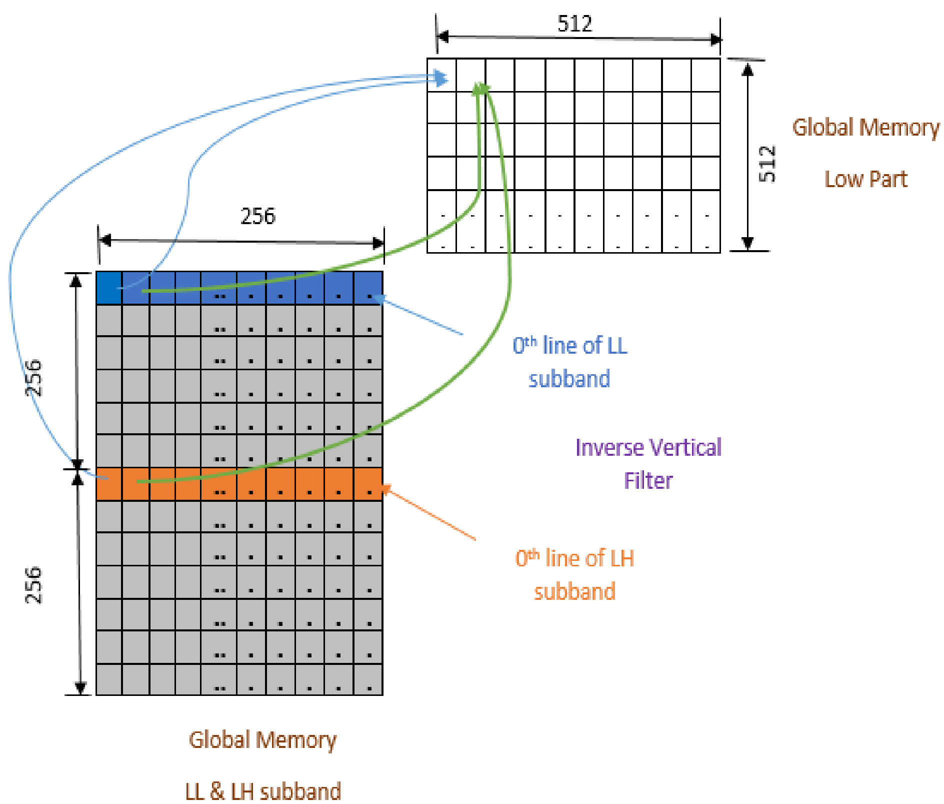

Fig. 4.6 The Computation in Horizontal Filter

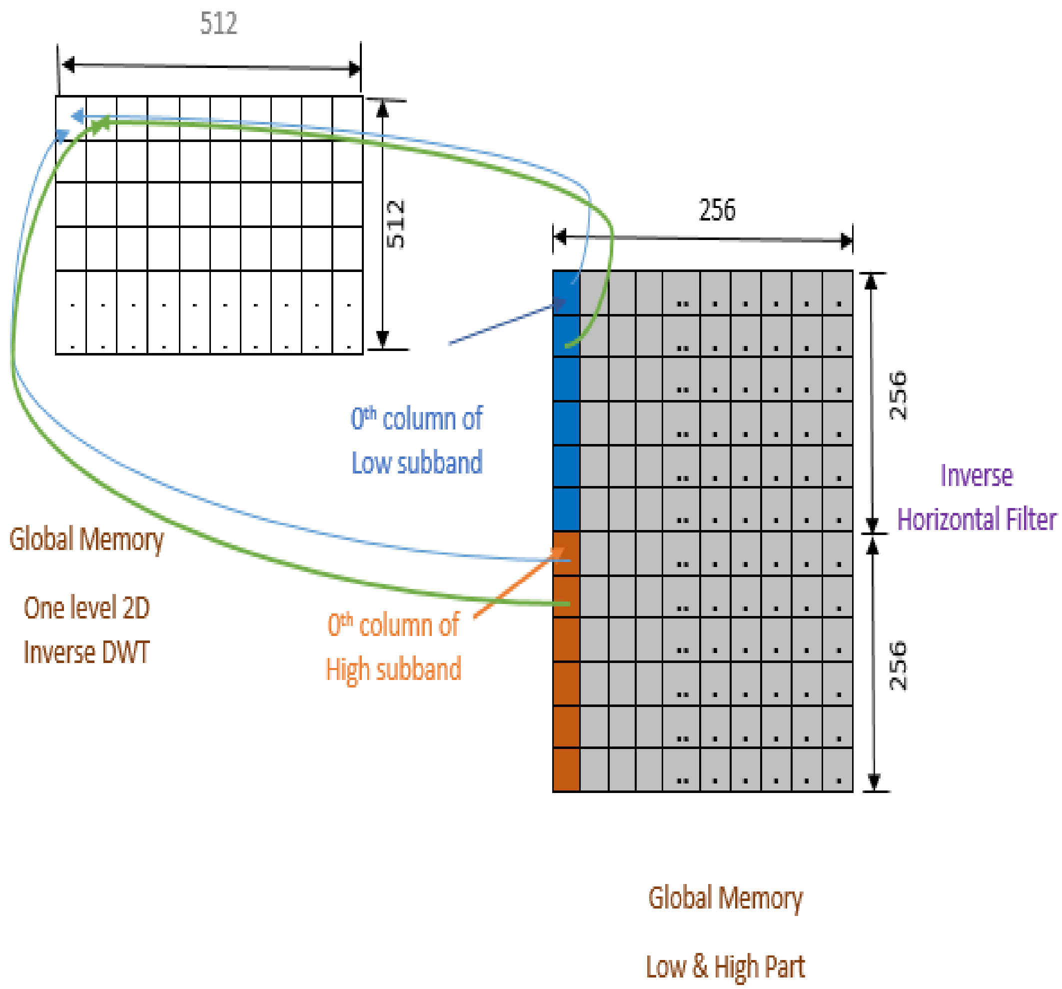

Fig. 4.7 The Computation in Vertical filter

Pseudo Code

Input : Wavelet Coefficients

Output : Original Image

Inverse_DWT_1

{

Allocate memory for wavelet coefficients at host side

Allocate memory for wavelet coefficients at device side

Read Wavelet Coefficients from the binary file

Copy wavelet coefficients from host to device side

For i=1 to 3

launch InverseDWT_Kernel_1 for LL and LH subband to compute Low part even rows elements

launch InverseDWT_Kernel_2 for LL and LH subband to compute Low part odd rows elements

launch InverseDWT_Kernel_1 for HL and HH subband to compute High part even rows elements

launch InverseDWT_Kernel_2 for HL and HH subband to compute High part odd rows elements

launch InverseDWT_Kernel_3 for Low and High part to compute image pixels even column elements

launch InverseDWT_Kernel_4 for Low and High part to compute image pixels odd column elements

end For

copy decompressed result from device to host

}

InverseDWT_Kernel_1

{

compute general **index**

compute equation (11) for each row

compute equation (12) for each row

}

InverseDWT_Kernel_2

{

compute general **index**

compute equation (7) for row

compute equation (8) for row

compute equation (9) for row

compute equation (10) for row

}

InverseDWT_Kernel_3

{

compute general **index**

compute equation (11) for column

compute equation (12) for column

}

InverseDWT_Kernel_4

{

compute general **index**

compute equation (7) for column

compute equation (8) for column

compute equation (9) for column

compute equation (10) for column

}

4.2.2 Decompressor_Math_Optimization

This version of decompressor takes the input image from binary file and then transfers this data from CPU to GPU. CPU launches number of threads same as number of pixels of an image for computation on GPU. There are total 4 kernels for computation of the Inverse 9/7 Integer Discrete Wavelet Transform. As we know, multiplication and division operations are time consuming operations. In order to get better performance in terms of time, multiplication and division operations are replaced by shift operations. Division operation is more time consuming than multiplication, due to this reason division operation is replaced by multiplication in this version.

i.e. for LL3 & LH3 subband, equation $x_{2j+1} = D_j/8 + \lfloor \frac{9}{16}(x_{2j}+x_{2j+2}) - \frac{1}{16}(x_{2j-2}+x_{2j+4}) + \frac{1}{2} \rfloor$ can be rewritten as $x_{2j+1} = D_j >> 3 + \lfloor (9 * (x_{2j}+x_{2j+2}) - (x_{2j-2}+x_{2j+4}) + 8) * 0.0625 \rfloor$

In above equation, division is replaced by shift operation and same equation is rewritten using only multiplication operation.

Pseudo Code

Input : Wavelet Coefficients

Output : Original Image

Inverse_DWT_2

{

Allocate memory for wavelet coefficients at host side

Allocate memory for wavelet coefficients at device side

Read Wavelet Coefficients from the binary file

Copy wavelet coefficients from host to device side

For i=1 to 3

launch InverseDWT_Kernel_1 for LL and LH subband to compute Low part even rows elements

launch InverseDWT_Kernel_2 for LL and LH subband to compute Low part odd rows elements

launch InverseDWT_Kernel_1 for HL and HH subband to compute High part even rows elements

launch InverseDWT_Kernel_2 for HL and HH subband to compute High part odd rows elements

launch InverseDWT_Kernel_3 for Low and High part to compute image pixels even column elements

launch InverseDWT_Kernel_4 for Low and High part to compute image pixels odd column elements

end For

copy decompressed result from device to host

}

InverseDWT_Kernel_1

{

compute general **index**

compute equation (11) for each row

compute equation (12) for each row

}

InverseDWT_Kernel_2

```
{
        compute general index
        compute equation (7) for row
        compute equation (8) for row
        compute equation (9) for row
        compute equation (10) for row
}
InverseDWT_Kernel_3
{
        compute general index
        compute equation (11) for column
        compute equation (12) for column
}
InverseDWT_Kernel_4
{
        compute general index
        compute equation (7) for column
        compute equation (8) for column
        compute equation (9) for column
        compute equation (10) for column
}
```

4.2.3 Decompressor_Two_Pixels_Computation

This version of decompressor takes the input image from binary file and then transfers this data from CPU to GPU. CPU launches number of threads same as half of number of pixels of an image for computation on GPU. There are total 4 kernels for computation of the Inverse 9/7 Integer Discrete Wavelet Transform which computes 2 pixel values using one thread with enabling L1 cache.

In the implementation, we have used Tesla K40c Accelerator for the parallel computation of Inverse Discrete wavelet transform. In this GPU, when we read any data from the global memory it takes 256 bytes (128 pixel values) at a time. When we request again same values then it is fetched from the L1 cache which reads 128 bytes (64 pixel values) at a time.

In previous case, we were using only 32 pixel values out of 64 pixel values at a time for the computation of IDWT. Consider the Eq. (6), when j=1, we require 0^{th} row/column elements, D_0 and 1^{st} row/column elements, D_1 and for j=2, we require 1^{st} row/column elements, D_1 and 2^{nd} row/column elements, D_2 for the computation of all even row/column elements of next level subband i.e. LL2 subband. Here we are reading 1^{st} row/column elements, D_1 for two times and using only 32 pixel values out of the 64 values in L1 cache. In the optimized version, one thread is going to compute two pixel values at a time which in turns uses all 64 pixel values which are being fetched in one clock cycle from L1 cache. Thus, we get higher hit ratio for L1 cache and speedup is achieved. This memory fetch and computation step is as shown in fig 4.8.

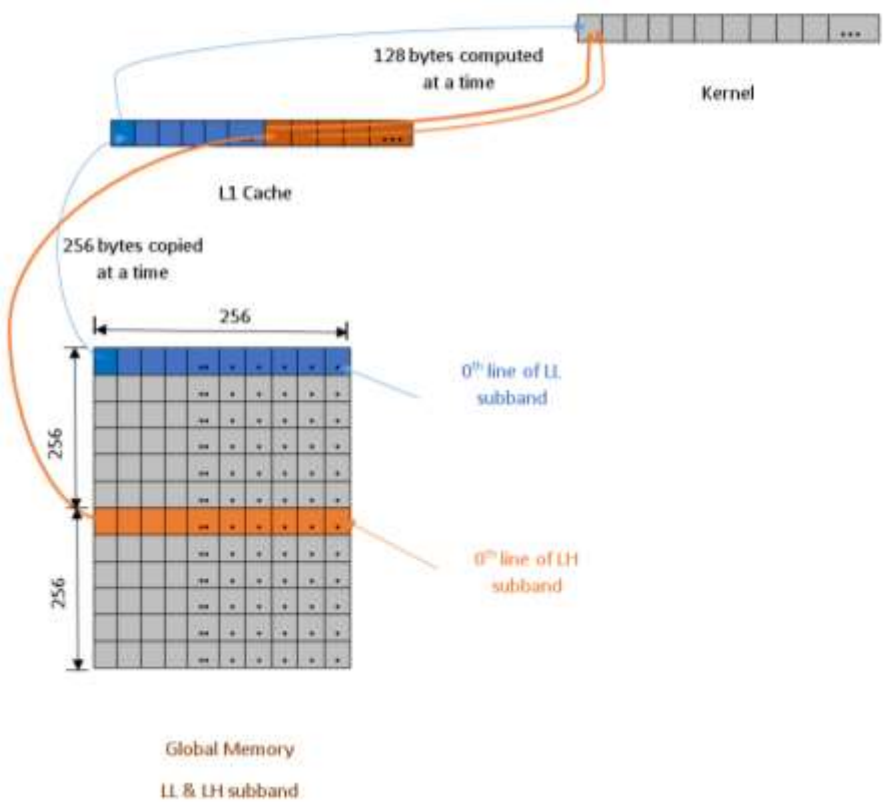

Fig. 4.8 Fetch step for horizontal filter

Pseudo Code

Input : Wavelet Coefficients

Output : Original Image

Inverse_DWT_3

{

Allocate memory for wavelet coefficients at host side

Allocate memory for wavelet coefficients at device side

Read Wavelet Coefficients from the binary file

Copy wavelet coefficients from host to device side

For i=1 to 3

launch InverseDWT_Kernel_1 for LL(32-bit) and LH(32-bit) subband to compute Low part even rows elements

launch InverseDWT_Kernel_2 for LL(32-bit) and LH(32-bit) subband to compute Low part odd rows elements

launch InverseDWT_Kernel_1 for HL(32-bit) and HH(32-bit) subband to compute High part even rows elements

launch InverseDWT_Kernel_2 for HL(32-bit) and HH(32-bit) subband to compute High part odd rows elements

launch InverseDWT_Kernel_3 for Low and High part to compute image pixels even column elements

launch InverseDWT_Kernel_4 for Low and High part to compute image pixels odd column elements

end For

copy decompressed result from device to host

}

InverseDWT_Kernel_1

{

compute general **index**

compute equation (11) for each row

compute equation (12) for each row

}

InverseDWT_Kernel_2

{

compute general **index**

compute equation (7) for row

compute equation (8) for row

compute equation (9) for row

compute equation (10) for row

}

InverseDWT_Kernel_3

{

compute general **index**

compute equation (11) for column

compute equation (12) for column

}

InverseDWT_Kernel_4

{

compute general **index**

compute equation (7) for column

compute equation (8) for column

compute equation (9) for column

compute equation (10) for column

}

4.2.4 Decompressor_Shared_Memory

This version of decompressor takes the input image from binary file and then transfers this data from CPU to GPU. CPU launches number of threads same as half of number of pixels of an image for computation on GPU. There is only one kernel for computation of the Inverse 9/7 Integer Discrete Wavelet Transform which computes pixel values using tiling approach and shared memory.

In this version, all four subbands LL, LH, HL and HH are copied in the shared memory block by block. Low & High part of size 32x36 block size, results of LL, LH subbands and HL, HH subbands respectively computed using horizontal filter in the shared memory. Here, shared memory bank conflict is zero. Then, one level inverse 2D Discrete Wavelet Transform is computed using Vertical Filter in the shared memory and result is copied back to Global Memory in the 32x32 block size.

In the parallel implementation, the time consuming part is the memory load store operation. Considering previous version of decompressor, result of all four kernels are stored in the global memory. Thus, in one level inverse 2-D DWT is accessing global memory six times. AS Time required for memory load store operation in the global memory has significant impact on the performance, access global memory for one time during all computation of one level 2-D Inverse DWT to optimize the performance. It can be done by using tiling approach and shared memory.

In this optimized version, we have done the computation on block size of 36x36 by launching only 32x32 threads in the one block. Here we have hallo region of 2 columns and 2 rows. Considering the Eq. (2) for inverse vertical filter, i.e. j=1, the elements of row number 0,2,4 & 6 are required to compute 3nd row element. Now, if we consider block of 32x32 as output for one block then for last 31st row element, we require 30,32,34 & 36 row element for the computation. So, here we have hallo region of 2 rows at upper side as well as for lower side of block to compute rows 0,1, …,31(32 rows). Considering the Inverse Horizontal Filter, to compute the column elements we require hallo region of 2 columns at left side as well as for right side as explained in case of inverse vertical filter. Thus, we require total 36x36 load operation for computation of 32x32 block size. We also have overlapping in loading in tiling approach. This Loading operation from global to shared memory is shown in below fig. 4.9.

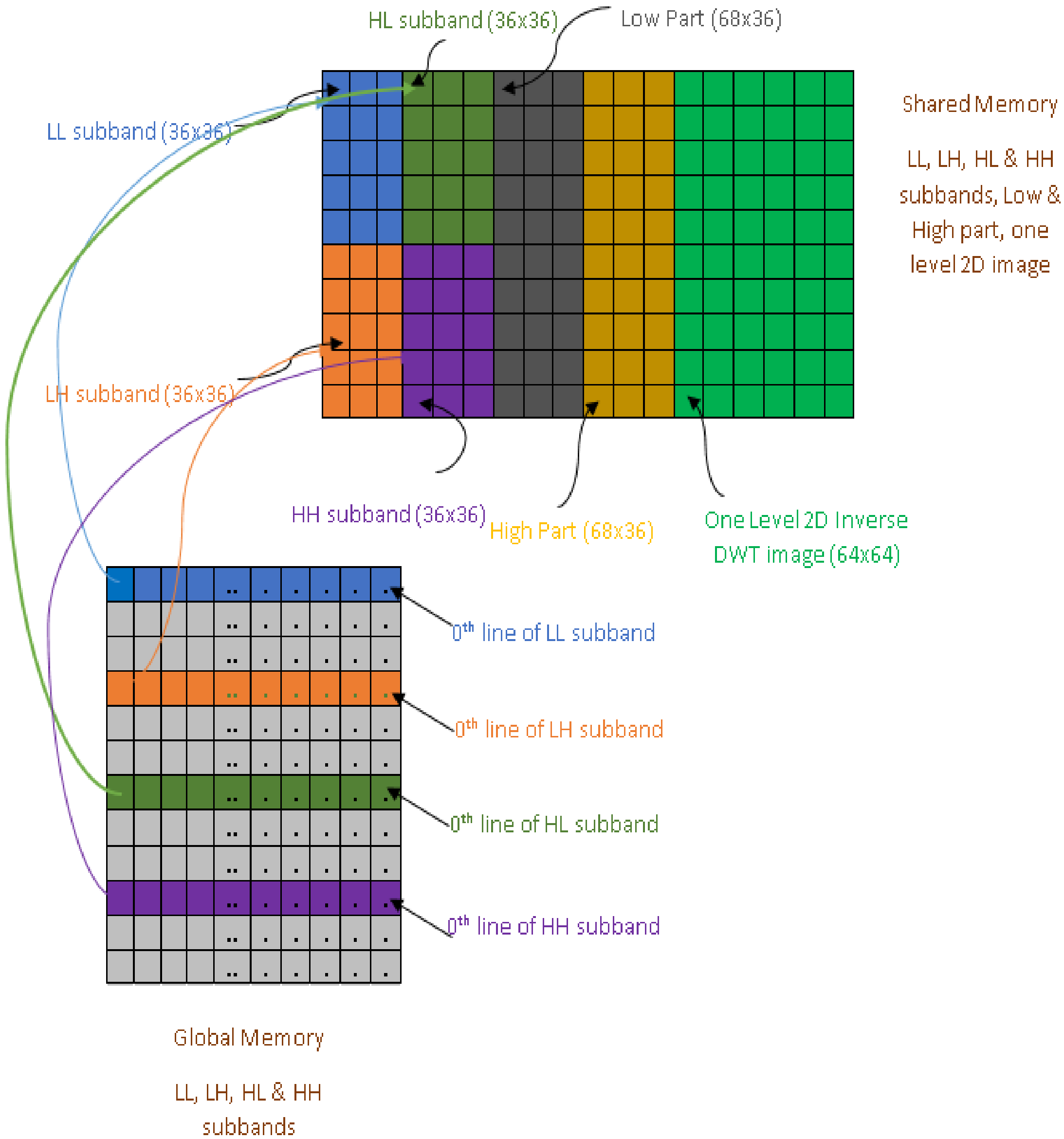

Fig. 4.9 Fetch and Copy step of kernel in shared memory

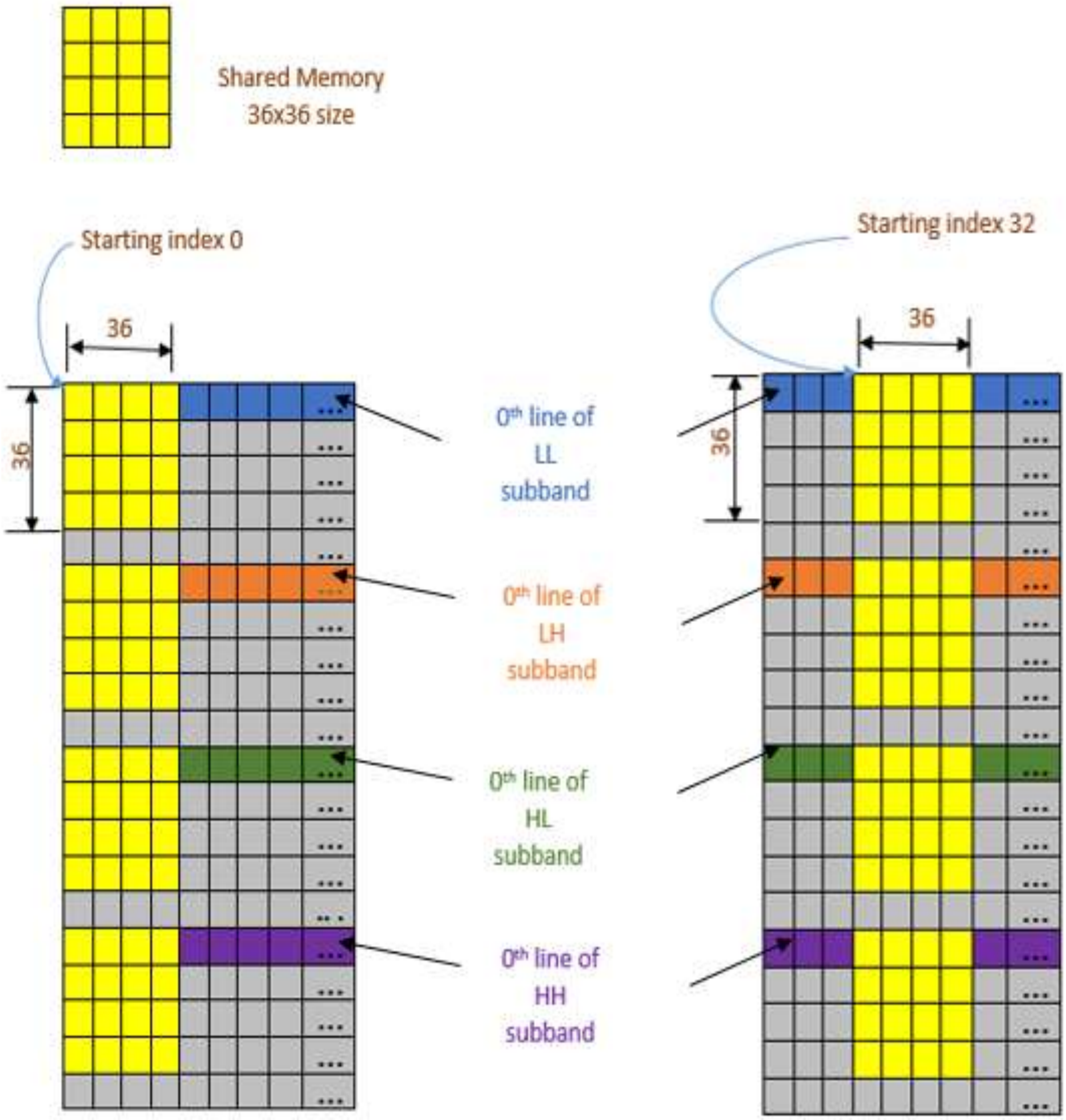

Fig. 4.10 Block movement in global memory

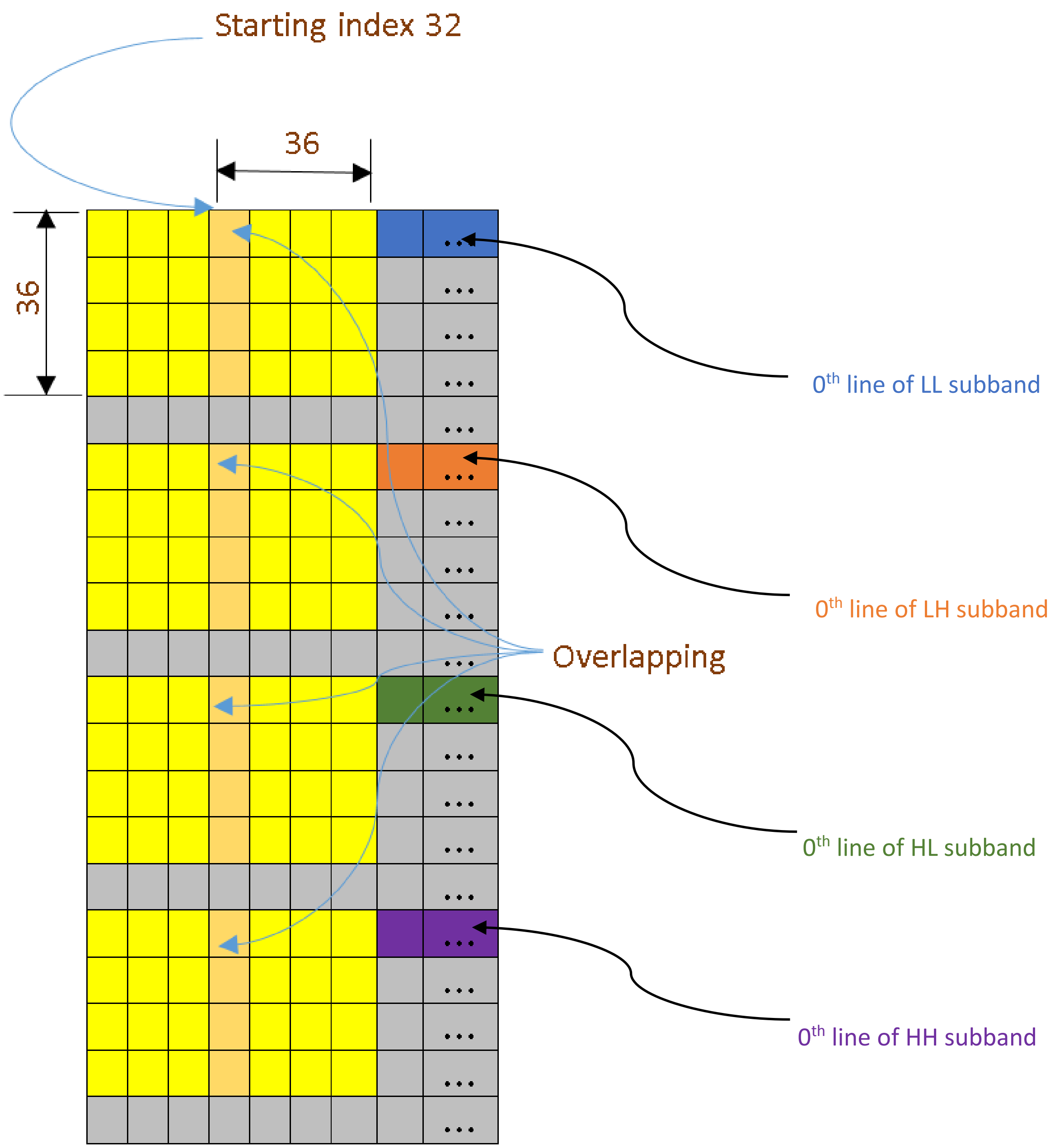

Fig. 4.11 overlapping in block movement

Pseudo Code

Input : Wavelet Coefficients

Output : Original Image

Inverse_DWT_4

{

Allocate memory for wavelet coefficients at host side

Allocate memory for wavelet coefficients at device side

Read Wavelet Coefficients from the binary file

Copy wavelet coefficients from host to device side

For i=1 to 3

launch InverseDWT_Kernel to compute image pixel values using LL, LH, HL and HH subbands.

end For

copy decompressed result from device to host

}

InverseDWT_Kernel

{

compute **block_column_index** and **block_row_index** to iterate through block

compute **block_column_index_h** and **block_row_index_h** for hallo region

compute general **index**

// computation of LL & LH components

copy LL and LH components' 32x32 values in the **shared variable**

copy **halo region values** of LL and LH components in the **shared variable**

compute **Low Part** and **Low Part halo region** even rows elements and store it to **shared variable**

compute **Low Part** and **Low Part halo region** odd rows elements and store it to **shared variable**

// computation of HL & HH components

copy HL and HH components' 32x32 values in the **shared variable** copy **halo region values** of HL and HH components in the **shared variable**

compute **High Part** and **High Part halo region** even rows elements and store it to **shared variable**

compute **High Part** and **High Part halo region** odd rows elements and store it to **shared variable**

// Computation of **LL subband**

Compute **LL subband** even columns elements and store it to **result**

Compute **LL subband** odd columns elements and store it to **result**

}

4.3 Software and Tools for CCSDS-123.0 Compressor (Part-2)

Matlab

MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment. It is proprietary programming language developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, C#, Java, Fortran and Python. Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the MuPAD symbolic engine, allowing access to symbolic computing abilities. An additional package, Simulink, adds graphical multi-domain simulation and model-based design for dynamic and embedded systems.

The MATLAB application is built around the MATLAB scripting language. Common usage of the MATLAB application involves using the Command Window as an interactive mathematical shell or executing text files containing MATLAB code. MATLAB also supports object-oriented programming including classes, inheritance, virtual dispatch, packages, pass-by-value semantics, and pass-by-reference semantics.

4.4 Functions of CCSDS-123.0 Compressor

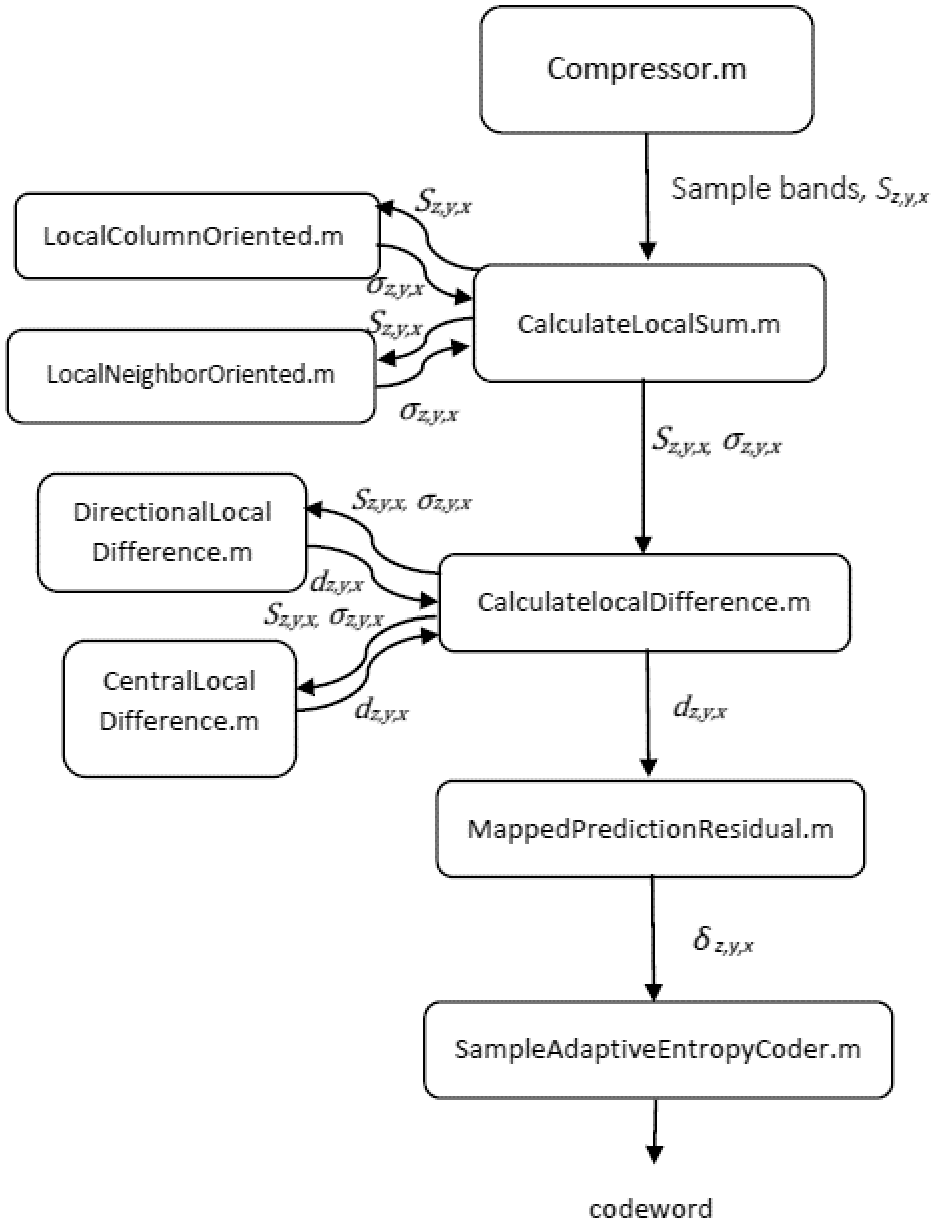

Fig. 4.9 Functions hierarchy of CCSDS-123.0 Compressor

Chapter 5

Results

5.1 Memory Statistics and Timing Details for CCSDS-122.0 Decompressor (Part-1)

5.1.1 Bit_Plane_Decoder_Unoptimized

- **Timing details:**

Table 5.1 Time taken by Bit Plane Decoder Unoptimized version
(Image size 256x256)

Function Name	Time (ms)
Rice_Decoding	28

5.1.2 Bit_Plane_Decoder_Optimized

- **Timing details:**

Table 5.2 Time taken by Bit Plane Decoder Optimized version
(Image size 256x256)

Function Name	Time (ms)
Rice_Decoding	5
Extract_Bit	3
Dequantization	3
Total	11

5.1.3 Decompressor_Unoptimized

- **Memory statistics:**

Table 5.3 Load-store operations for Decompressor_Unoptimized
(Image size 256x256)

Kernel	Load/Kernel launch	Store/Kernel launch
InverseDWT_Kernel_1	6136	2048
InverseDWT_Kernel_2	10224	2048
InverseDWT_Kernel_3	13312	4608

InverseDWT_Kernel_4	27136	5632
Total	73168	18432

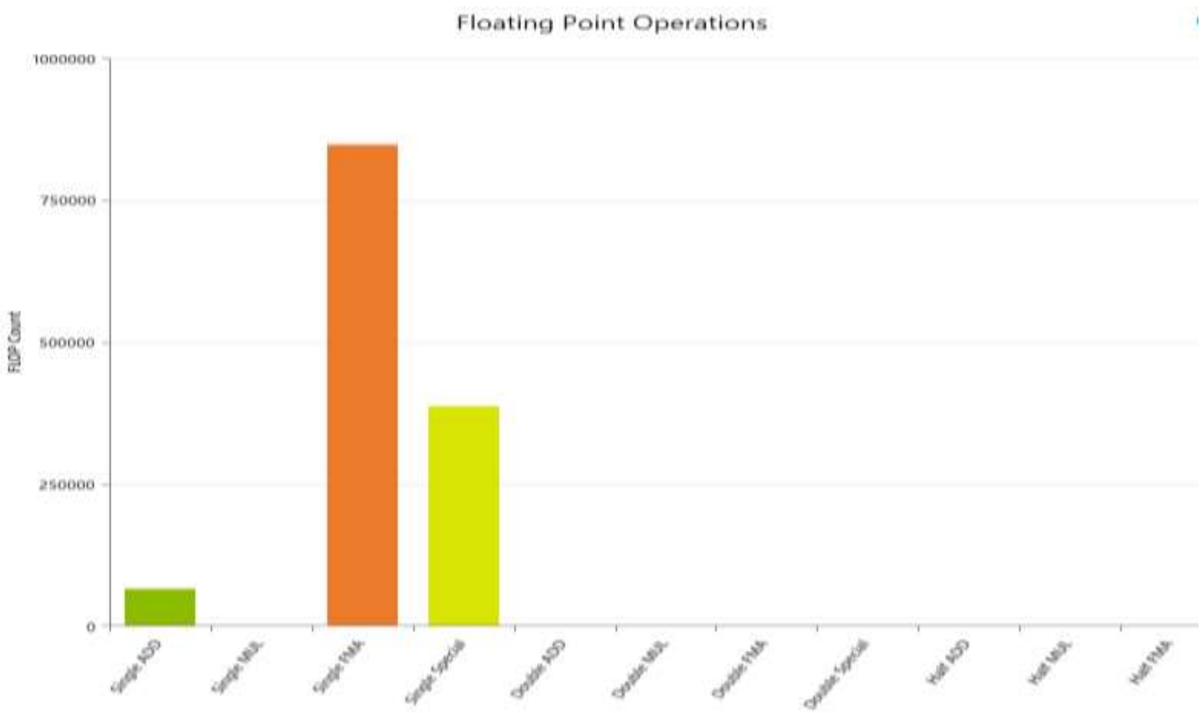

Fig. 5.1 Floating point operation for Decompressor_Unoptimized

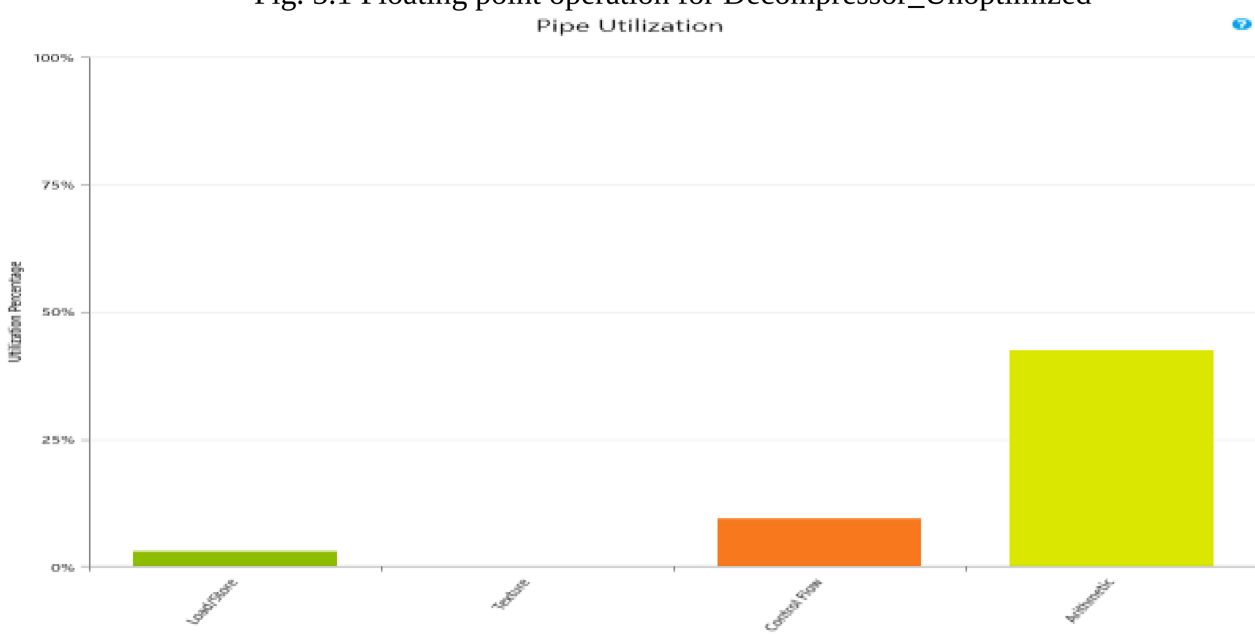

Fig. 5.2 Pipeline Utilization for Decompressor_Unoptimized

- **Timing details:**

Time taken by each kernel execution is as shown in table 5.4. Total time taken by this approach is 1.617 ms.

Table 5.4 Time taken by Decompressor_Unoptimized

Level	Kernel name	Image size	Time (μ sec)
3	InverseDWT_Kernel_1	256x256	12
3	InverseDWT_Kernel_2	256x256	11
3	InverseDWT_Kernel_3	256x256	23
3	InverseDWT_Kernel_4	256x256	25
2	InverseDWT_Kernel_1	512x512	40
2	InverseDWT_Kernel_2	512x512	34
2	InverseDWT_Kernel_3	512x512	80
2	InverseDWT_Kernel_4	512x512	88
1	InverseDWT_Kernel_1	1024x1024	149
1	InverseDWT_Kernel_2	1024x1024	131
1	InverseDWT_Kernel_3	1024x1024	308
1	InverseDWT_Kernel_4	1024x1024	337

5.1.4 Decompressor_Math_Optimization

- **Memory statistics:**

Table 5.5 Load-Store operations for Math_Optimized
(Image size 256x256)

Kernel	Load/Kernel Launch	Store/Kernel Launch
InverseDWT_Kernel_1	6136	2048
InverseDWT_Kernel_2	10224	2048
InverseDWT_Kernel_3	13312	4608
InverseDWT_Kernel_4	27136	5632
Total	73168	18432

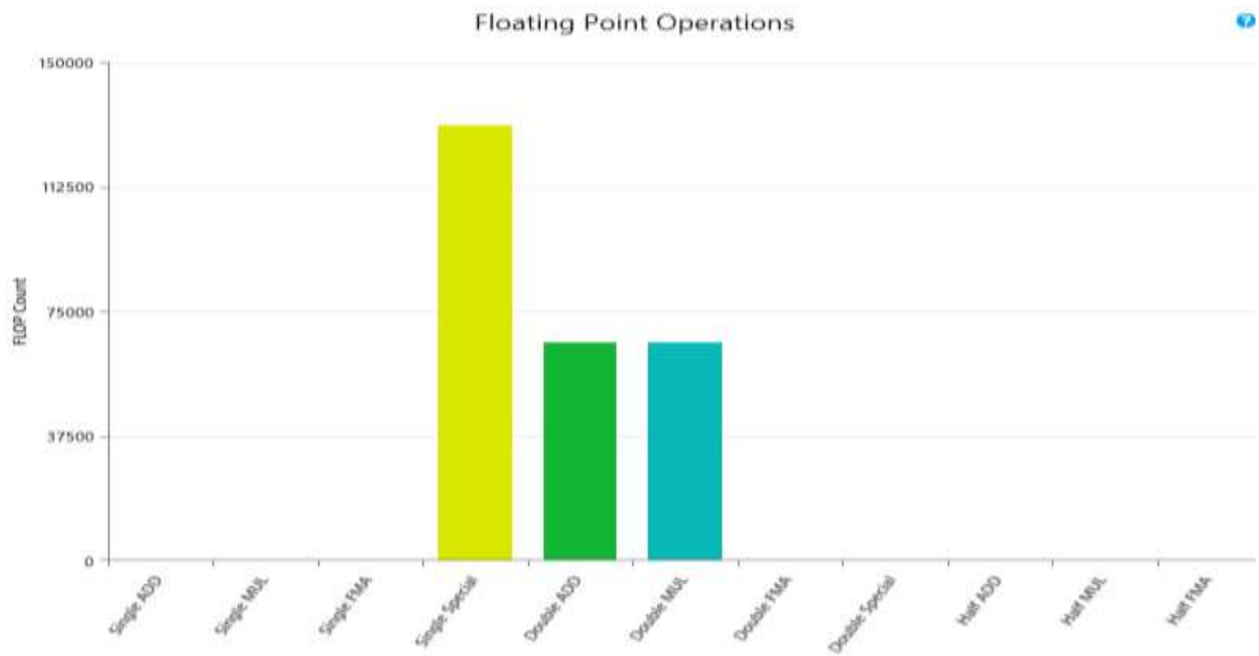

Fig. 5.3 Floating point operation for Decompressor_Math_Optimized

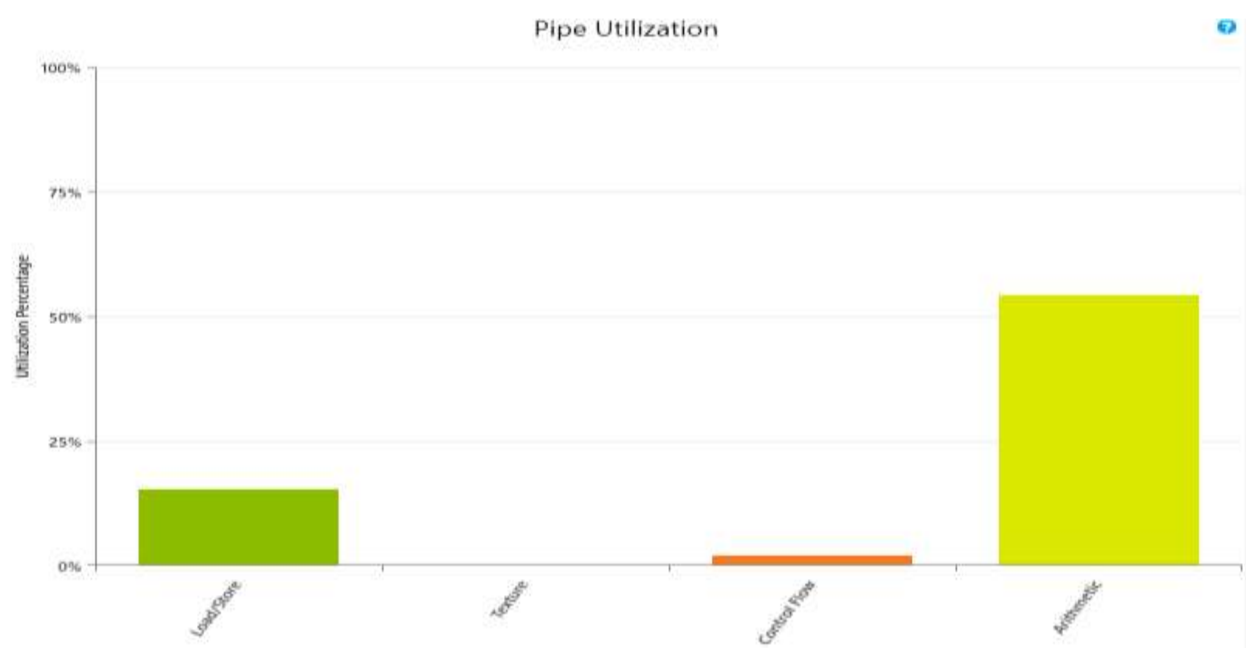

Fig. 5.4 Pipeline Utilization for Decompressor_Math_Optimized

- **Timing details:**

Time taken by each kernel execution is as shown in table 5.6. Total time taken by this approach is 1.464 ms.

Table 5.6 Time taken by Decompressor_Math_Optimized

Level	Kernel name	Image size	Time (μ sec)
3	InverseDWT_Kernel_1	256x256	11
3	InverseDWT_Kernel_2	256x256	10
3	InverseDWT_Kernel_3	256x256	19
3	InverseDWT_Kernel_4	256x256	26
2	InverseDWT_Kernel_1	512x512	32
2	InverseDWT_Kernel_2	512x512	30
2	InverseDWT_Kernel_3	512x512	66
2	InverseDWT_Kernel_4	512x512	94
1	InverseDWT_Kernel_1	1024x1024	120
1	InverseDWT_Kernel_2	1024x1024	120
1	InverseDWT_Kernel_3	1024x1024	260
1	InverseDWT_Kernel_4	1024x1024	363

5.1.5 Decompressor_Two_Pixel_Computation

- **Memory statistics:**

Table 5.7 Load-Store operation for Two_Pixel_Computation

(Image size 256x256)

Kernel	Load/Kernel Launch	Store/Kernel Launch
InverseDWT_Kernel_1	3068	1024
InverseDWT_Kernel_2	5104	1024
InverseDWT_Kernel_3	13312	4608
InverseDWT_Kernel_4	26112	5632
Total	55768	14336

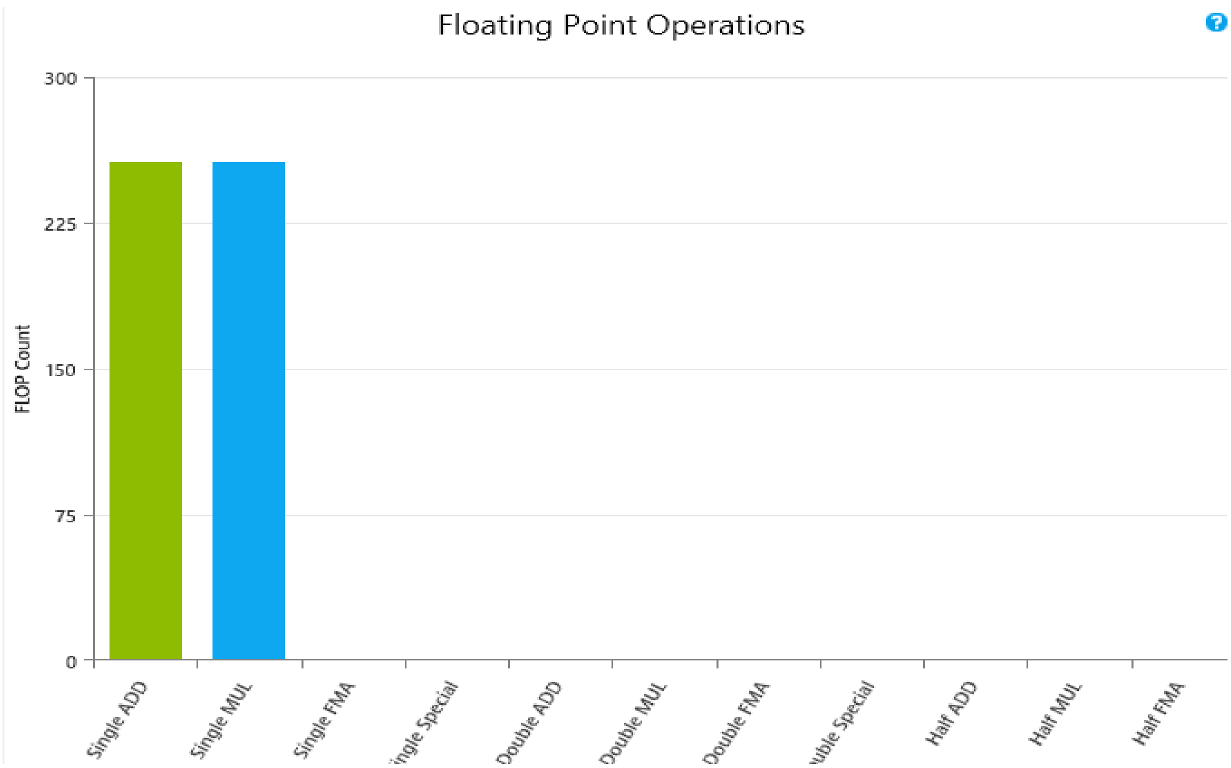

Fig. 5.5 Floating point operation for Two_Pixel_Computation

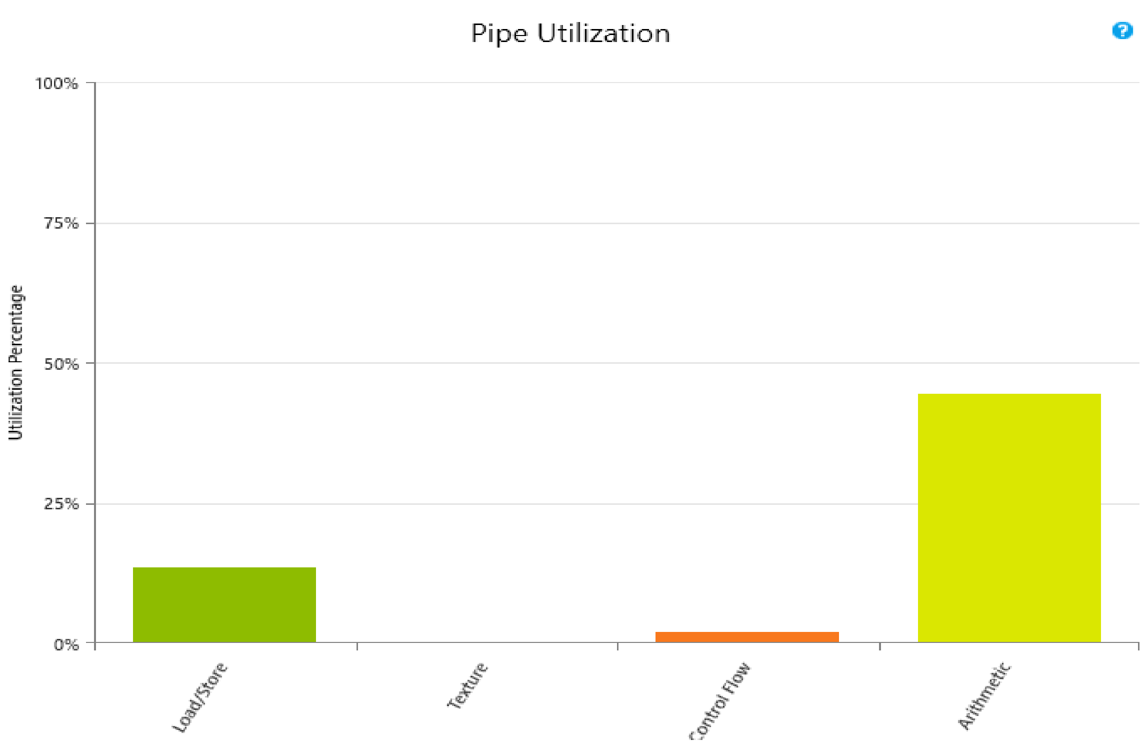

Fig. 5.6 Pipeline Utilization for Two_Pixel_Computation

- **Timing details:**

Time taken by each kernel execution is as shown in table 5.8. Total time taken by this approach is 0.827 ms.

Table 5.8 Time taken by Decompressor _Two_Pixel_Computation

Level	Kernel name	Image size	Time(μ sec)
3	InverseDWT_Kernel_1	256x256	5
3	InverseDWT_Kernel_2	256x256	6
3	InverseDWT_Kernel_3	256x256	10
3	InverseDWT_Kernel_4	256x256	19
2	InverseDWT_Kernel_1	512x512	12
2	InverseDWT_Kernel_2	512x512	15
2	InverseDWT_Kernel_3	512x512	43
2	InverseDWT_Kernel_4	512x512	67
1	InverseDWT_Kernel_1	1024x1024	48
1	InverseDWT_Kernel_2	1024x1024	55
1	InverseDWT_Kernel_3	1024x1024	162
1	InverseDWT_Kernel_4	1024x1024	247

5.1.6 Decompressor_Shared_Memory

- **Memory statistics:**

Table 5.9 Load-store operation for Decompressor_shared_memory

kernel	Image size	Load/Kernel Launch	Store/Kernel Launch
IDWT_Kernel	256x256	19456	13932

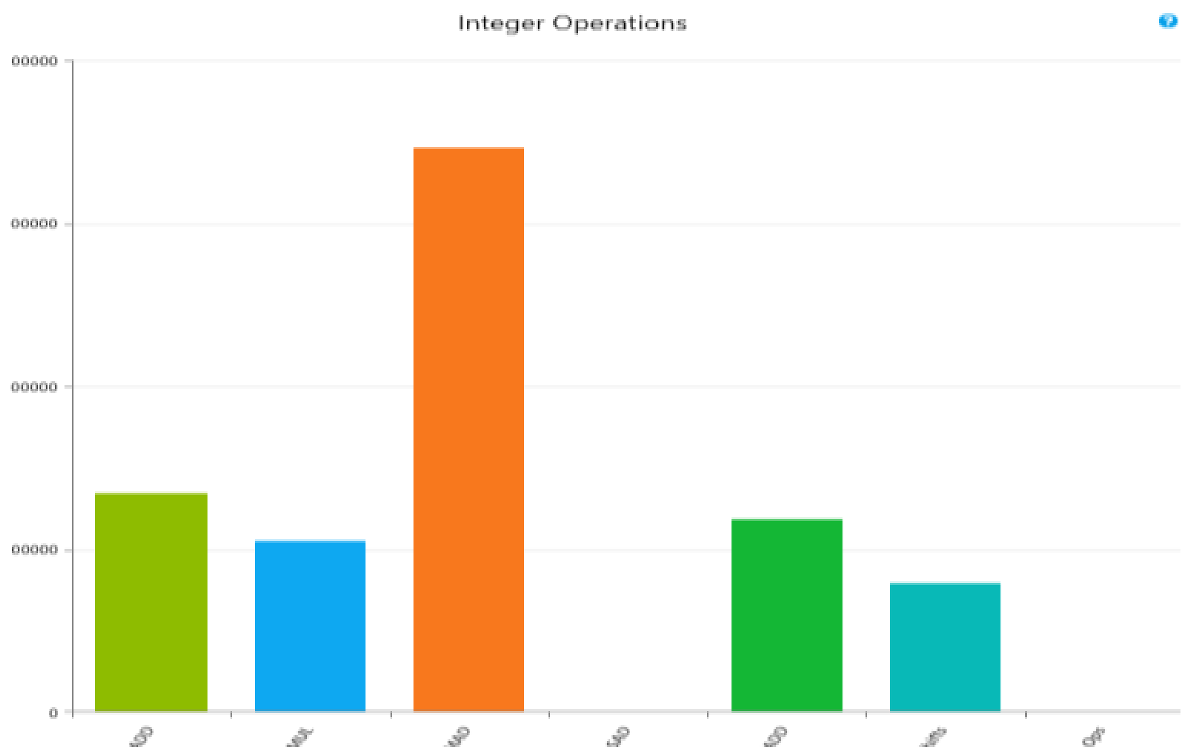

Fig. 5.7 Integer Operation for Decompressor_Shared_Memory

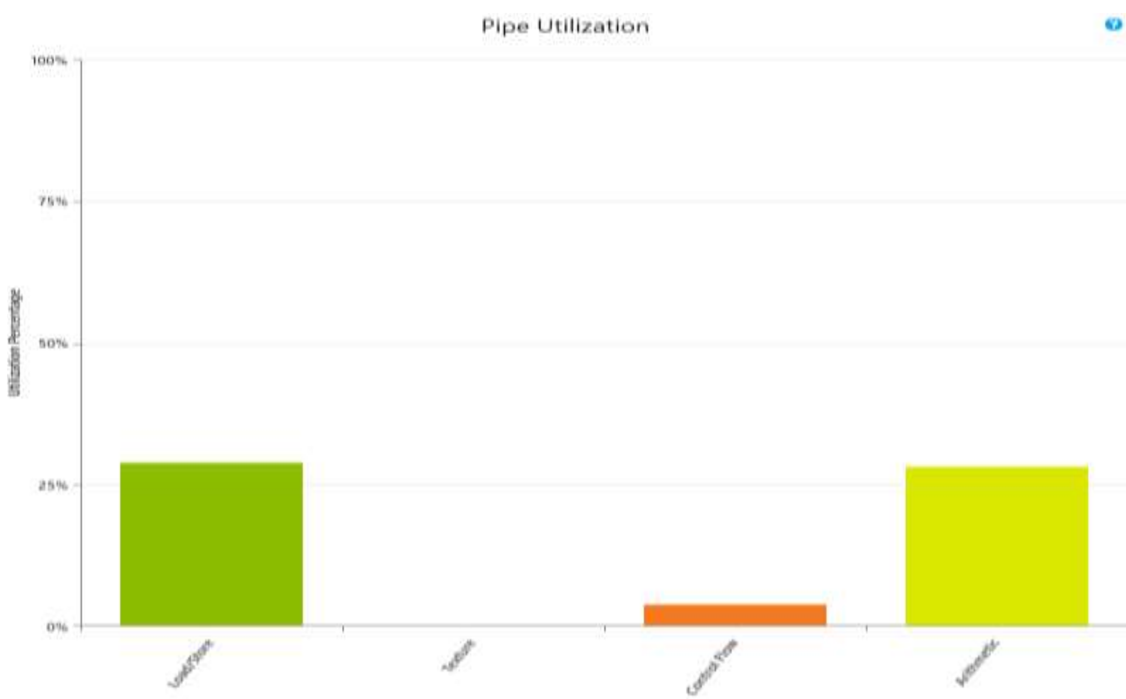

Fig. 5.8 Pipe Utilization for Decompressor_Shared_Memory

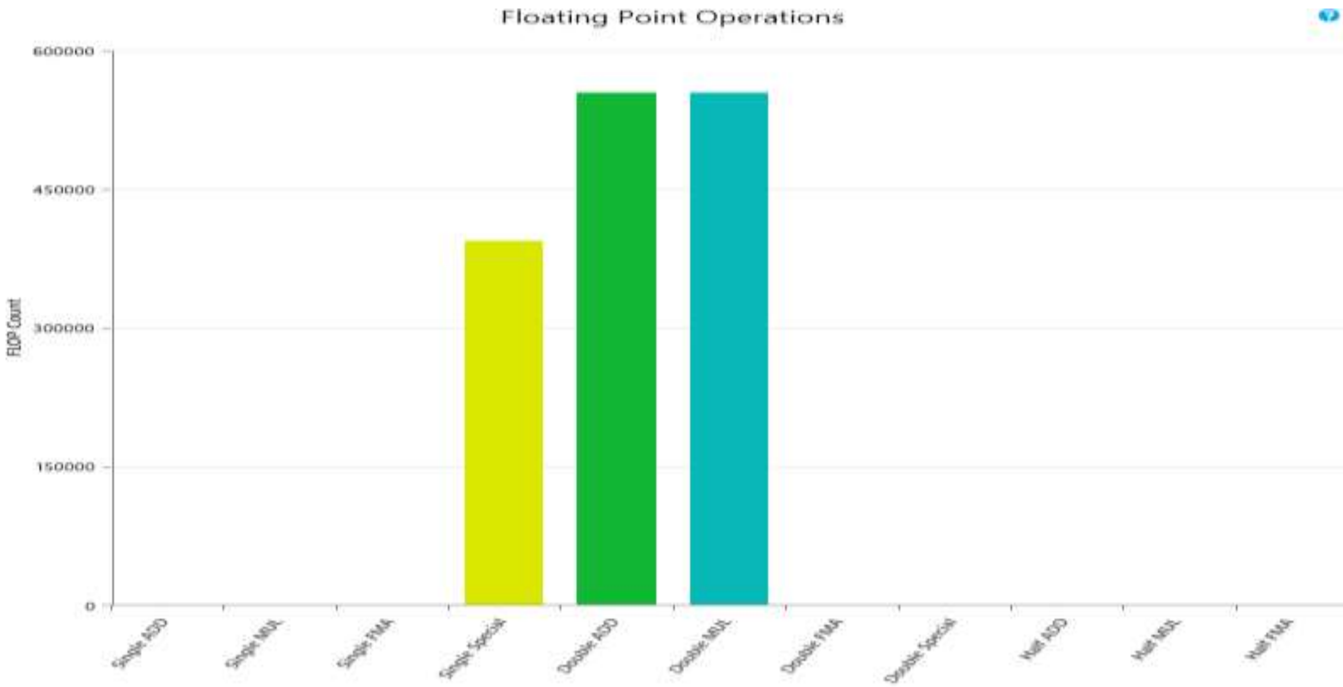

Fig. 5.9 Floating point operation for Decompressor_Shared_Memory

- **Timing details:**

Time taken by each kernel execution is as shown in table 5.10. Total time taken by this approach is 1.679 ms.

Table 5.10 Time taken by shared memory kernel for different sizes of images

Level	**Image size**	**Time(μ sec)**
3	256x256	99
2	512x512	330
1	1024x1024	1250

Table 5.11 Time taken by Decompressor System

Bit Plane Decoding		Inverse DWT		Decompressor System
Version name	***Time(ms)***	***Version name***	***Time(ms)***	***Total Time(ms)***
Unoptimized	28	Unoptimized	15	43
Unoptimized	28	Math_Optimized	10	38
Unoptimized	28	Two_Pixel_Computation	9	37
Unoptimized	28	Shared_Memory	50	78
Optimized	11	Unoptimized	15	26
Optimized	11	Math_Optimized	10	21
Optimized	11	Two_Pixel_Computation	9	20
Optimized	11	Shared_Memory	50	61

5.2 Performance Comparison for CCSDS-122.0 Decompressor (Part-1)

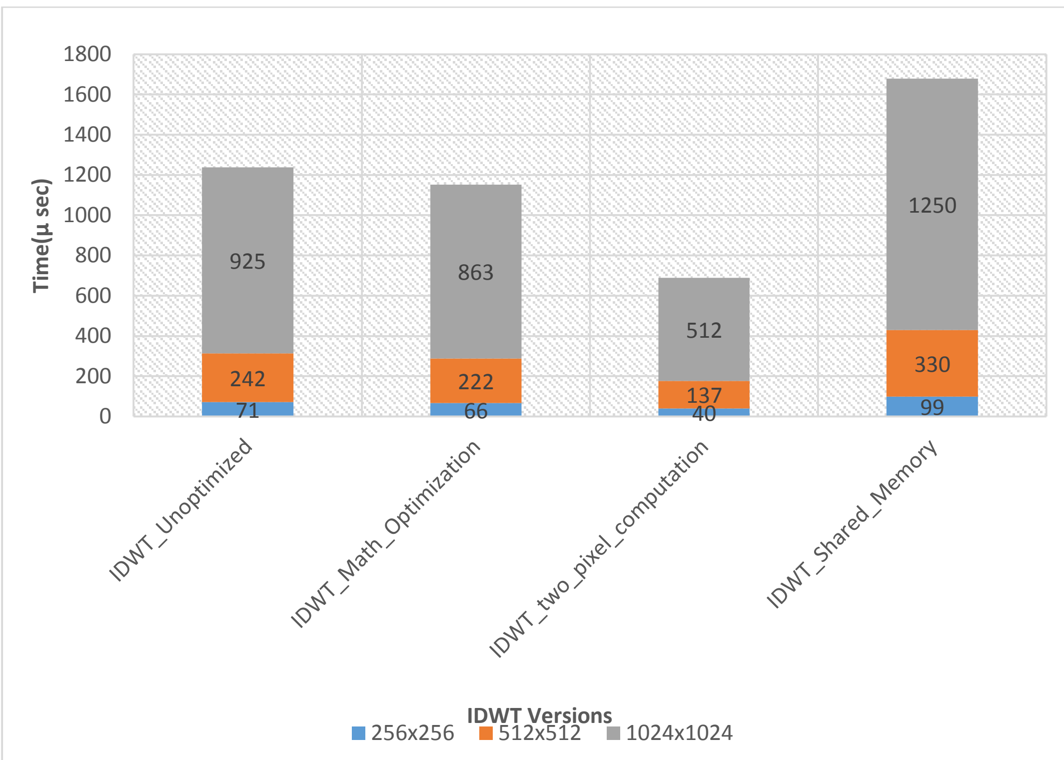

Fig. 5.10 Time Performance Comparison chart for IDWT Versions

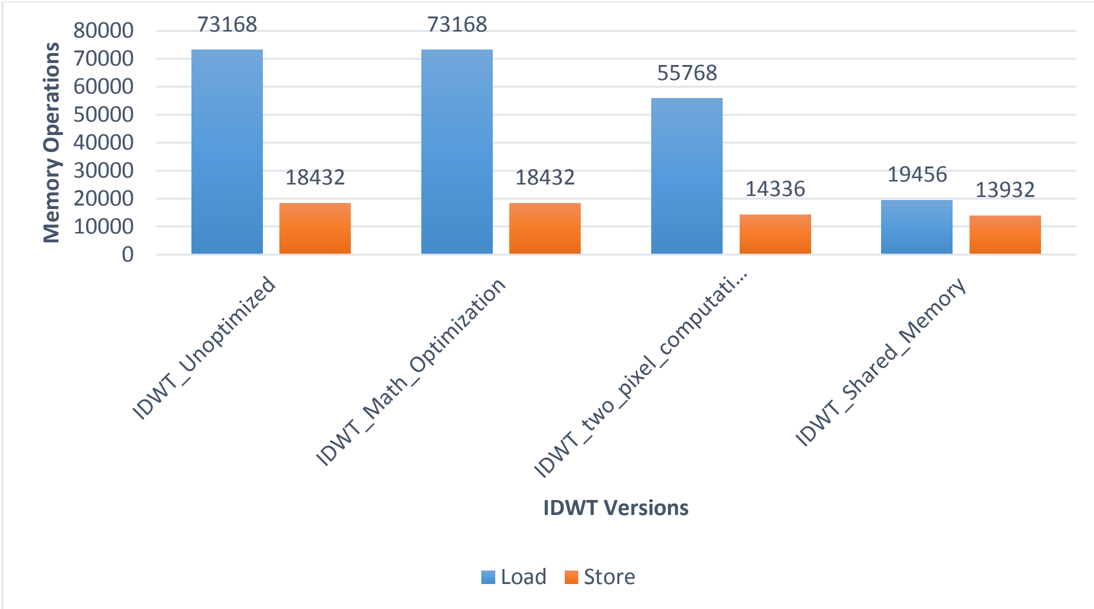

Fig. 5.11 Memory load-store Comparison chart for IDWT Versions

(Image size 256x256)

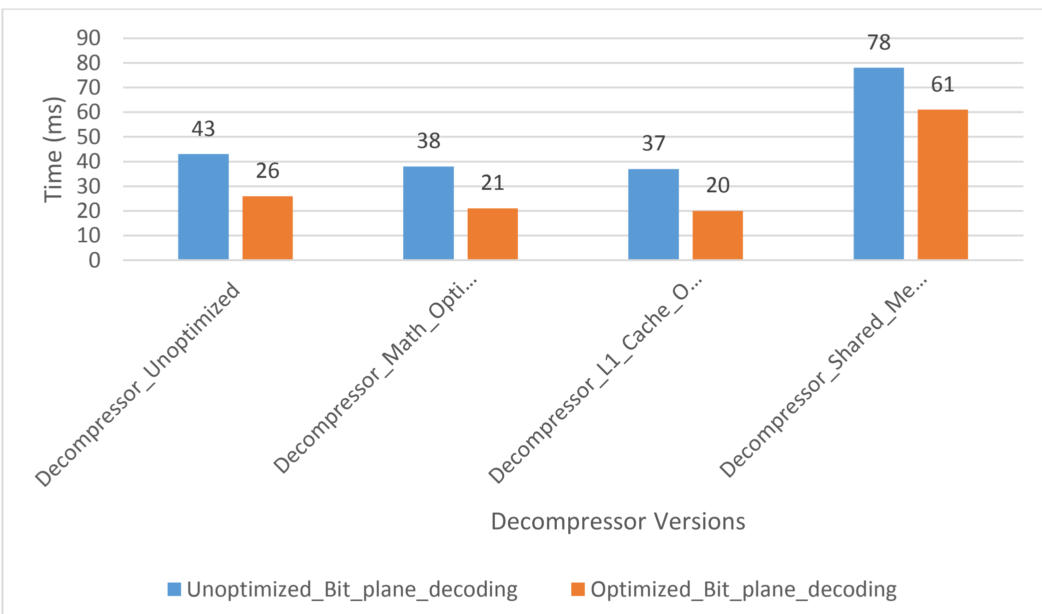

Fig. 5.12 Time performance comparison for decompressor system

5.3 Performance Comparison for CCSDS-123.0 Compressor (Part-2)

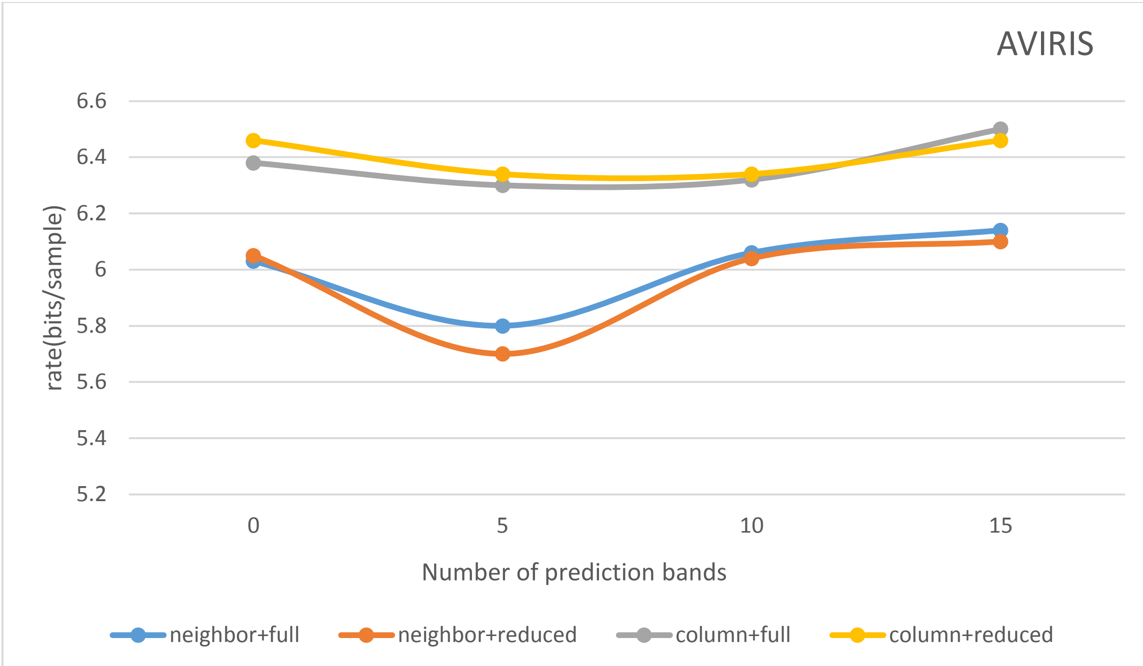

Fig. 5.13 Performance comparison chart for number of prediction bands for AVIRIS

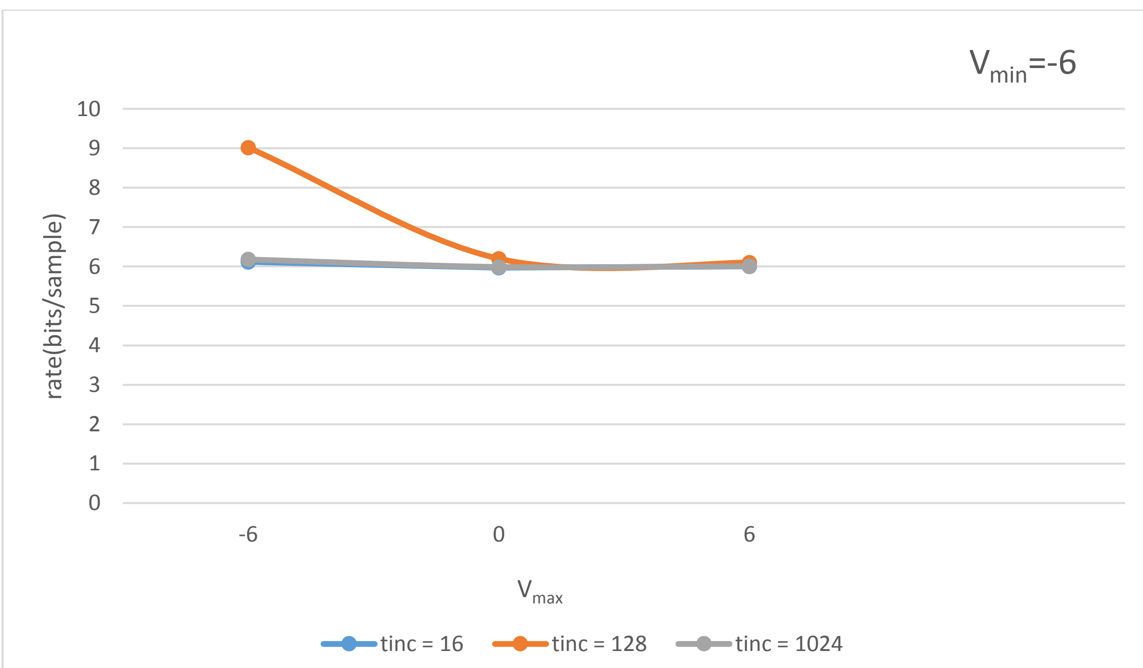

Fig. 5.14 Performance comparison chart for V_{max} Vs rate when V_{min}=-6

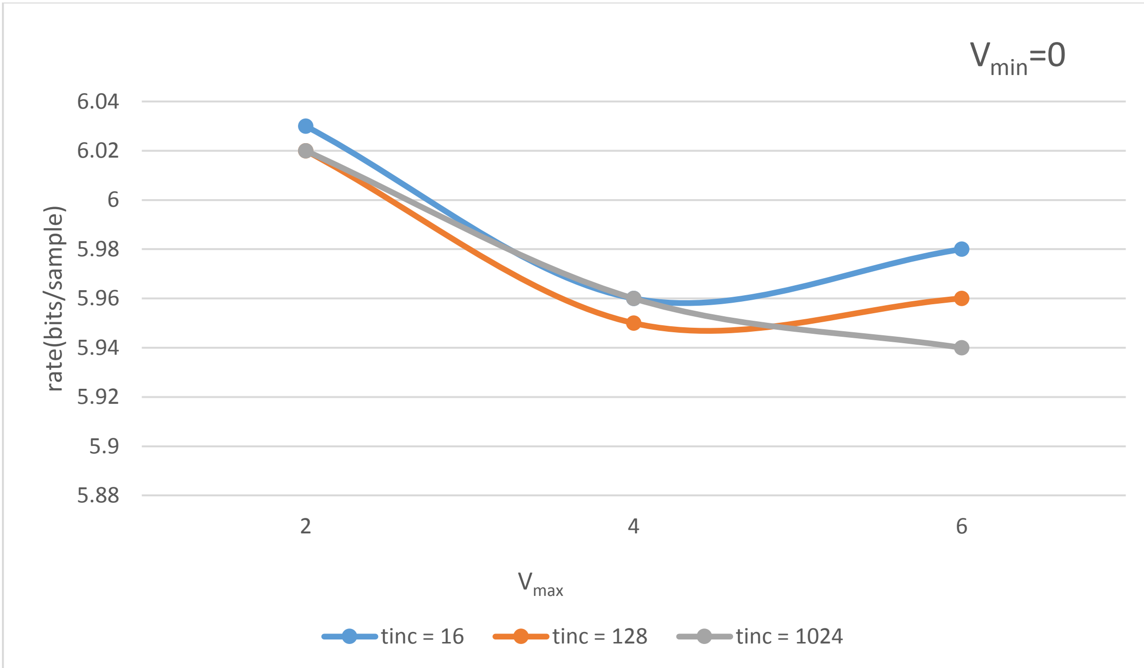

Fig. 5.15 Performance comparison chart for V_{max} Vs rate when $V_{min}=0$

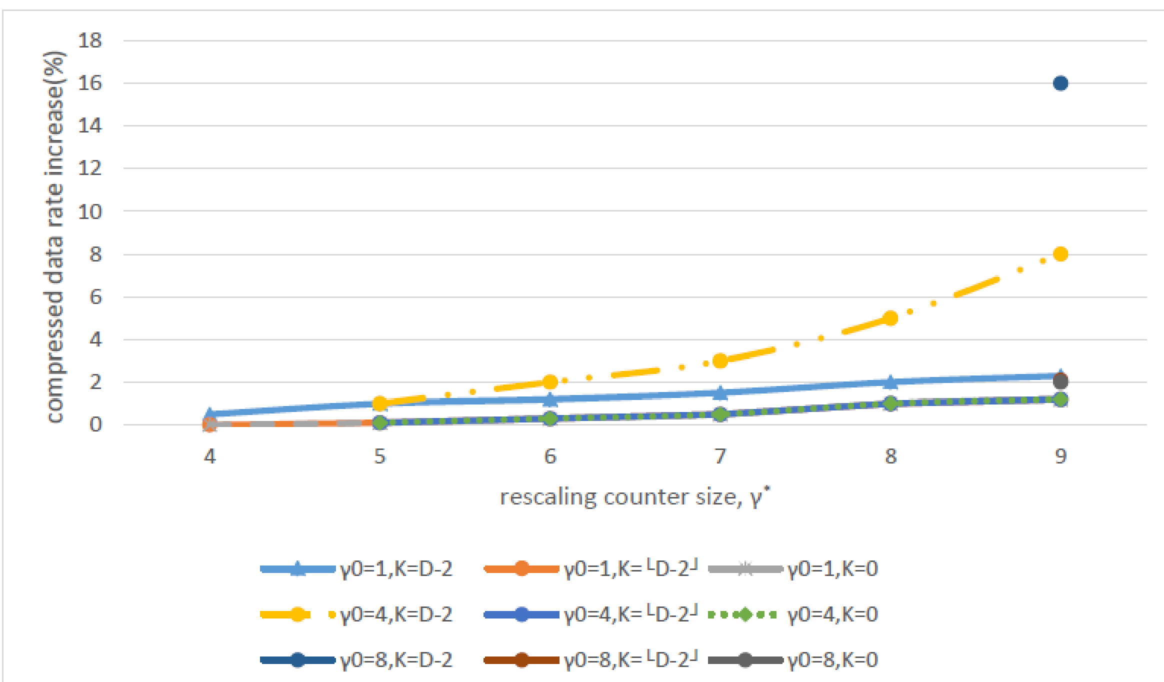

Fig. 5.16 Change in Compressed Data Rate as a Function of Rescaling Counter Size γ*

Chapter 6

Conclusion

The CCSDS-122.0-B-1 based decompressor contains two phases, Bit Plane decoder and Inverse 9/7 Integer Discrete Wavelet Transform. As 9/7 Integer Discrete Wavelet Transform is most time consuming part, it has been implemented on GPU for Speedup. There are total four versions of Inverse DWT. Decompressor_Two_Pixels_Computation is best in terms of time and Decompressor_Shared_Memory is best in terms of memory load store operation among these versions. Decompressor System of Version 3, Two Pixels Computation with Optimized Bit Plane Decoder is efficient. Optimized_Bit _Plane_Decoder & Inverse_DWT_Two_pixels take 11 ms and 9 ms respectively. Thus, total time for Decompressor System is 20 ms.

The CCSDS-123.0-B-1 Compression system contains two modules, Predictor and Encoder. Compression analysis of this system is complex as it depends on various parameters. Comparison of compression quality is done through rate (bits/sample) with specific parameters. Increment in the number of prediction bands has small impact on performance. For AVIRIS sensor, neighbour oriented with reduced mode gives best results in terms of rate. It is also observed that when Weight update scaling Exponent Parameter, V_{max} is small, compression quality is not good but as V_{max} increases it gives better results up to a point, but then it remains almost constant for higher values of V_{max}. The use of smaller value of γ^* causes the improved performance when poor value of K is used.

Chapter 7

References

1. "Image Data Compression",Recommendation for space data system standards, CCSDS 122.0-B-1. Blue Book,November 2005.
2. Changhe Song, Yunsong Li, and Bormin Huang,"A GPU-Accelerated Wavelet Decompression System With SPIHT and Reed-Solomon Decoding for Satellite Images",IEEE journal of selected topics in applied earth observations and remote sensing, vol. 4, no. 3, september 2011.
3. Abhishek S. Shetty, Abhijit V. Chitre and Yogesh H. Dandawate, ”Time Efficiency Comparison of Wavelet and Inverse Wavelet Transform on Different Platforms”,International Conference on Computing Communication Control and automation (ICCUBEA) IEEE-2016.
4. Anastasis Keliris, Vasilis Dimitsasy, Olympia Kremmyday, Dimitris Gizopoulosy and Michail Maniatakosz, ” Efficient parallelization of the Discrete Wavelet Transform algorithm using memory-oblivious optimizations”, 25th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS),pp.25-32, 2015
5. John Nickolls,” GPU Parallel Computing Architecture and CUDA Programming Model “, Hot chips 19 Symposium (HCS) IEEE, pp.1-12, 2007
6. Khoirudin and Jiang Shun-Liang, “Gpu application in cuda memory”, Advanced Computing: An International Journal (ACIJ), Vol.6, No.2, pp.1-10, March 2015
7. NVIDIA, “NVIDIA’s Next Generation CUDATM Compute Architecture: Kepler TM GK110/210”, United States, 2014.
8. NVIDIA, “Cuda C Programming Guide”, United States, September 2017.
9. "Lossless Multispectral & Hyperspectral Image Compression", Recommendation for space data system standards, CCSDS 123.0-B-1. Blue Book,May 2012.
10. Lucana Santos, Luis Berrojo, Javier Moreno, José Fco. López, and Roberto Sarmiento,” Multispectral and Hyperspectral Lossless Compressor for Space Applications (HyLoC): A Low-Complexity FPGA Implementation of the CCSDS 123 Standard”, IEEE journal of selected topics in applied earth observations and remote sensing, vol. 9, no. 2, pp.757-770, February 2016.

WEBSITES:

1. https://public.ccsds.org/
2. https://www.wikipedia.org/
3. https://developer.nvidia.com/

APPENDIX A

ABBREVIATION NOTATION

CCSDS	Consultative Committee for Space Data System
GPGPU	General Purpose Graphics Processing Unit
GIS	Graphical Information System
RGB	Red Green Blue
IDC	Image Data Compression
DWT	Discrete Wavelet Transform
CUDA	Computed Unified Device Architecture
SM	Streaming Multiprocessor
SFU	Special Function Unit
LD	Load
ST	Store
SIMD	Single Instruction Multiple Data
IDWT	Inverse Discrete Wavelet Transform
IDE	Integrated Development Environment
AVIRIS	Airborne Visible/Infrared Imaging Spectrometer

APPENDIX B

REVIEW CARD

GUJARAT TECHNOLOGICAL UNIVERSITY

(Established Under Gujarat Act No.: 20 of 2007)

ગુજરાત ટેક્નોલોજીકલ યુનિવર્સિટી

(ગુજરાત અધિનિયમ ક્રમાંક : ૨૦/૨૦૦૭ દ્વારા સ્થાપિત)

ESTD - 2007

Master of Engineering

(Dissertation Review Card)

Name of Student : MANIYA SHAILJABEN MUKESHBHAI

Enrollment No. : 1 6 0 2 8 0 7 2 3 0 0 7

Student's Mail ID:- shailjamaniya1994@gmail.com

Student's Contact No. : 9426408185

College Name : L.D. COLLEGE OF ENGINEERING

College Code : 0 2 8

Branch Code : 2 3 **Branch Name :** INFORMATION TECHNOLOGY

Theme of Title : IMAGE PROCESSING

Title of Thesis : AN EFFICIENT GPGPU BASED CCSDS RECOMMENDED DWT DECOMPRESSOR AND OPTIMIZATION OF HYPERSPECTRAL COMPRESSION PARAMETERS

Supervisor's Detail		***Co-supervisor's Detail***	
Name :	B. B. Panchal	**Name :**	HIREN RAMBHIA
Institute :	LDCE	**Institute :**	SAC, ISRO
Institute Code :	028	**Institute Code :**	-
Mail Id :	bakul@ldce.ac.in	**Mail Id :**	rambhia hiren@sac.isro.gov.in
Mobile No. :	9426476891	**Mobile No. :**	9714132117

❖ **Comments For Internal Review (2730002)** (Semester 3)

Exam Date : 15/ 9/ 19

Sr. No.	Comments given by Internal review panel (Please write specific comments)	Modification done based on Comments
1	Good work	
	Try to implement proposed work	done
2.	Title must be changed to only reflect the work.	done
3.	papers must be included on subtopics that cover the research area	done

(Guide Sign.)

Particulars	Internal Review Panel	
	Expert 1	Expert 2
Name :	A. R. Patel	H. M. Diwanji
Institute :	L.D.C.E	L.D.C.E.
Institute Code :	028	028
Mobile No. :	9510225586	9924073670
Sign :		h. Diwanji

Particulars	Internal Guide Details	
	Expert 1	Expert 2
Name :	D. D. Panchal	
Institute :	LDCE	
Institute Code :	028	
Mobile No. :	9426413584	
Sign :		

Enrollment No. of Student : 1 6 0 2 8 0 7 2 3 0 0 7

❖ **Comments of Dissertation Phase-1 (2730003)** **(Semester 3)**

Exam Date : 04/12/2017

Hall No : 8

Title : AN EFFICIENT GPGPU BASED CCSDS RECOMMENDED DWT DECOMPRESSOR AND OPTIMIZATION OF HYPERSPECTRAL COMPRESSION PARAMETERS

1. Appropriateness of title with proposal. (Yes/ No) Yes
2. Whether the selected theme is appropriate according to the title ? (Yes / No) Yes
3. Justify rational of proposed research. (Yes/ No) Yes
4. Clarity of objectives. (Yes/ No) Yes

Enrollment No. of Student: 1 6 0 2 8 0 7 2 3 0 0 7

Hall No.: 8

Exam Date: 04/12/2017

Sr. No.	Comments given by External Examiners (DP-I) : (Please write specific comments)	Modification done based on Comments
1.	Need to specify the evalution parameters.	
2.	Good work.	
3.	flow of work is good.	
		(Internal Guide Sign.)

- *Approved* ✓
- *Approved with suggested recommended changes* ☐
- *Not Approved* ☐

Please tick on any on.
If approved/approved with suggession then put marks ≥ 50 %.

> ## *Details of External Examiners :*

Particulars	*Full Name*	*University / College Name & Code*	*Mobile No.*	*Sign.*
Expert 1	Dr. S. D. Panchal	017 - VGEC		
Expert 2	Dr. V. K. Thakar	BVM - 018		

Enrollment No. of Student: | 1 | 6 | 0 | 2 | 8 | 0 | 7 | 2 | 3 | 0 | 0 | 7 |

❖ **Comments of Mid Sem Review (2740001)** **(Semester 4)**

Hall No: 8

Exam Date: 03/03/2018

Sr. No.	**Comments given by External Examiners:** i) The appropriateness of the major highlights of work done; State here itself if work can be approved with some additional changes. ii) Main reasons for approving the work. iii) Main reasons if work is not approved.	**Modification done based on Comments**
–	Good Presentation Skills	
–	Try to complete remaining	done
	implementation work.	
		Internal Guide Sign.

- *Approved* ☑
- *Approved with suggested recommended changes* ☐
- *Not Approved* ☐

Please tick on any on.
If approved/approved with suggession then put marks ≥ 50 %.

➢ *Details of External Examiners:*

Particulars	*Full Name*	*University / College Name & Code*	*Mobile No.*	*Sign.*
Expert 1	Dr. Viral Borisagar	G.T.U	—	
Expert 2	Prof. Falguni Patel	SVIT	—	

Enrollment No. of Student : | | | | | | | | | | | | |

❖ **Comments of DP-II Review (2740002)** **(Semester 4)**

Hall No :

Exam Date : ___/___/___

Sr. No.	**Comments given by External Examiners :** i) Main reasons for approving the work. ii) Main reasons if work is not approved.	Modification done based on Comments

- *Approved* ☐
- *Not Approved* ☐

Please tick on any one.

If approved then put marks ≥ 50 %.

➢ *Details of External Examiners :*

Particulars	*Full Name*	*University / College Name & Code*	*Mobile No.*	*Sign.*
Expert 1				
Expert 2				

APPENDIX C

PLAGIARISM REPORT

turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	**ld engineering college**
Assignment title:	**Final Report_139**
Submission title:	**Shailja Report**
File name:	**Shailja_Report.docx**
File size:	**5.22M**
Page count:	**82**
Word count:	**11,194**
Character count:	**66,080**
Submission date:	**06-Apr-2018 11:13AM (UTC+0530)**
Submission ID:	**543184258**

Shailja Report

ORIGINALITY REPORT

16%	13%	6%	14%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	public.ccsds.org Internet Source	4%
2	Submitted to Sultan Agung Islamic University Student Paper	2%
3	Submitted to Visvesvaraya Technological University Student Paper	2%
4	gtu.ac.in Internet Source	1%
5	Abhishek S. Shetty, Abhijit V. Chitre, Yogesh H. Dandawate. "Time efficiency comparison of wavelet and inverse wavelet transform on different platforms", 2016 International Conference on Computing Communication Control and automation (ICCUBEA), 2016 Publication	1%
6	Submitted to Ibra College of Technology Student Paper	1%
7	Submitted to Sreenidhi International School	

Student Paper

1%

8	en.wikipedia.org Internet Source	1%
9	refbase-es.iuma.ulpgc.es Internet Source	1%
10	developer.nvidia.com Internet Source	1%
11	Submitted to Texas A&M University, College Station Student Paper	1%

Exclude quotes On
Exclude bibliography On
Exclude matches < 1%

APPENDIX D

PAPER PUBICATION CERTIFICATES

IJSRSET

Scientific Journal Impact Factor = 4.916

International Journal of Scientific Research in Science, Engineering and Technology

Print ISSN : 2395-1990, Online ISSN : 2394-4099

Ref : IJSRSET/Certificate/Volume 4/Issue 4/4446 10-Apr-2018

CERTIFICATE OF PUBLICATION

This is to certify that **Shailja Maniya** has published a research paper entitled *'Efficient parallelization of Inverse DWT using GPGPU'* in the International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Volume 4, Issue 4, March-April-2018.

This Paper can be downloaded from the following IJSRSET website link
http://ijsrset.com/IJSRSET1844367
ijsrset Team wishes all the best for bright future

Editor in Chief
International Journal of Scientific Research in Science, Engineering and Technology
Website : http://www.ijsrset.com

IJSRSET

UGC Approved Journal [Journal No : 47147]

LAA

IJSRSET

L. D. COLLEGE OF ENGINEERING

INFORMATION TECHNOLOGY DEPARTMENT

NATIONAL CONFERENCE

ON

ADVANCE RESEARCH TRENDS IN INFORMATION AND COMPUTING TECHNOLOGY

20TH JANUARY 2018.

CERTIFICATE

This is to certify that MANIYA SHAILJABEN MUKESHBHAI has presented/published the paper titled A SURVEY OF EFFICIENT CCSDS RECOMMENDED DWT DECOMPRESSOR at "National Conference on Advance Research Trends in Information and Computing Technology" held on 20th January 2018, organized by information technology department in association with LAA and Sponsored by GUJCOST.

Dr. G. P. VADODARIYA
CHIEF PATRON
PRINCIPAL

Prof. H. M. DIWANJI
CO-ORDINATOR
HOD-IT

Dr. M. C. PARIKH
CO-COORDINATOR
ASSOCIATE PROFESSOR-IT