# Implementation of CCSDS Image Data Compression Standard on DSP Platform

Muhammad Ammar and Yasir Saleem

***Abstract*—The "Consultative Committee for Space Data Systems" (CCSDS) has recommended an image data compression standard: CCSDS 122.0-B-1, to be used onboard space data systems. This research presents the algorithm implementation of the CCSDS image data compression standard (CCSDS-IDC) on "Digital Signal Processor" platform (DSP). The algorithm is applied to two-dimensional digital grey scale image data from imaging payload devices and uses two-dimensional "Discrete Wavelet Transform" (DWT) followed by progressive "Bit Plane Encoder" (BPE) to generate the compressed encoded bit stream. This research aims to give an efficient implementation of the standard in terms of cost, memory utilization, customization flexibility, power and throughput, on the DSK6713 floating point DSP platform. DSP and "Synchronous Dynamic RAM" (SDRAM) based implementation scheme is adopted. SDRAM dynamic memory allocation scheme is used for program, data and working memory. DSP multiple ALUs and pipelining has been utilized for efficient throughput. The results evaluated for the lossy compression scheme validate the performance of the algorithm onboard DSP platform. The implementation also serves as reference practical system to be used in space-borne applications.**

***Index Terms*— CCSDS, image compression, DWT, bit plane encoder, DSP, DSK6713.**

## I. Introduction

IMAGE data compression is an important element in the onboard space data systems. It enables to reduce the amount of image data, resulting in the reduction of onboard memory and downlink transmission bandwidth requirements for space missions. Imaging payloads of space data systems generally belong to one of the two categories; CCD arrays or push-broom sensors. CCD arrays generate frames of images and push-broom based systems acquire a line or strip of an image at a time. Single channel panchromatic imaging sensors of both types effectively generate grayscale images having integer valued pixels. Acquisition of each frame or strip demands additional storage space and transmission bandwidth. Hence the task of image data compression comes in handy to compress the image data and reduce those requirements. To use an algorithm onboard space data systems, presents different challenges than those in general purpose applications. Onboard systems have limited computational resources, memory and power. Therefore the algorithm used must take all these constraints into consideration. "Consultative Committee for Space Data Systems" (CCSDS) [1] works on the issues faced by onboard space data systems. CCSDS has recommended many standards including the one which is the subject of this research work: (CCSDS 122.0-B-1) for image data compression [2].

An image compression scheme has generally two functional modules. The de-correlation of data is performed by some mathematical transform whereas the transformed data is processed by an encoder which performs the quantization and encoding to produce compressed image. Similarly the "CCSDS image data compression standard (CCSDS 122.0-B-1)" which will be abbreviated as "CCSDS-IDC" throughout this document, uses two functional modules i.e. the discrete wavelet transform (DWT) and the bit plane encoder (BPE) modules. It can perform both lossy and lossless compression and has very low complexity so that it can be implemented with minimum power and processing resources requirements [3].

Several research groups have been engaged in developing software and hardware for the implementation of the CCSDS-IDC algorithm, with majority focused to implement it on "Field Programmable Gate Arrays" (FPGA) and "Application Specific Integrated Circuits" (ASIC) for real time applications. A number of ASIC implementations of the algorithm and several architectures have been proposed but relatively not that much work has been done to implement the algorithm in its true form on "Digital Signal Processor" (DSP) platforms for buffered image data compression for off-line or non real-time applications.

ASICs have fixed operation while DSP can be reprogrammed. DSPs have the ability of processing digital data with greater efficiency than general purpose processors and have special architecture to support numerically intensive operations. Hence they are optimum for the task of image data compression. ASICs have real-time hardware processing advantage associated with them but implementation complexity and development cost are the major challenges. DSP can be used in non real-time image data compression applications where buffered image data onboard can be compressed for storage and downlink transmission.

Manuscript received April 15, 2015. This work was carried out by the author in partial fulfillment of Masters Degree at Department of Computer Science and Engineering, University of Engineering and Technology Lahore.

Muhammad Ammar has done his Masters in Computer Engineering from University of Engineering and Technology Lahore, (e-mail: i_ammar@hotmail.com).

Dr. Yasir Saleem has supervised this work. He is Associate Professor at the Department of Computer Science and Engineering, University of Engineering and Technology Lahore (e-mail: yasir@uet.edu.pk).

Implementation on DSP platform relatively costs much lower than that on ASIC. Total design costs for a typical 1-million-gate cell-based technology ASIC design, 0.13µm process is around $5.5 Million, and non-recurring engineering charges (NRE) for tools and setups add up to $1 Million or more. An improved methodology known as structured ASIC has total design cost around $500K and NRE costs ranging $100K to $200K [4]. On the other hand, commercially available DSP chips cost only a few dollars per unit and complete DSP based prototyping platforms cost within $500 to $1000.

"DSP Starter Kit": DSK6713 has been used as a primary DSP platform. It contains a floating point DSP TMS320C6713 onboard along with 16 Megabytes of "Synchronous Dynamic RAM" (SDRAM). A DSP platform like DSK6713 is a complete solution to many design projects and requires no additional development cost. The development can be readily started and re-programmability of the DSP presents a customizable and flexible solution to the problem.

DSP and SDRAM based implementation methodology is adapted. The general purpose implementation of the algorithm on PC platform is migrated to the DSP platform. DSP takes care of the algorithm processing while all the intermediate and compressed data is stored in SDRAM.

The implementation is then subjected to the performance evaluation for the test images. Compression results are obtained under lossy compression scheme by applying different test criteria. The results obtained validate the performance of the algorithm onboard DSP platform. Hence this work presents a low cost and flexible solution to onboard image compression problem which can be implemented on the existing space data systems by the space agencies.

The organization of the paper is as follows. Image compression background and related work is reviewed in section II. CCSDS-IDC specifications are briefly explained in section III. Section IV gives the DSP implementation scheme and explains the DSP-SDRAM implementation architecture. Section V discusses the implementation results and performance evaluations followed by the conclusion and future work suggestions in section VI.

## II. IMAGE DATA COMPRESSION BACKGROUND

Image data is being used in many fields of science & technology; ranging from medical imaging, agricultural applications and identification databases to the multimedia revolution and remote sensing applications etc. Some basic concepts of image data compression are reviewed here.

### *A. Image and Data Compression*

An image is a two-dimensional array of pixels i.e. intensity values. Data compression is the process of reducing the amount of data needed to represent a given quantity of information. There are some data redundancies in the image data which can be exploited in image data compression.

*1) Types of Redundancies*

There are three types of redundancies found in images.

a) Coding Redundancy
b) Inter-pixel Redundancy
c) Psycho-visual Redundancy

*a) Coding Redundancy:* The codes used to represent pixels values contain more bits than are needed. Arithmetic coding [5], Huffman coding [6] and Golomb coding [7] are some entropy coding techniques which exploit coding redundancy.

*b) Inter-pixel Redundancy:* The value of a pixel can be partially estimated by the value of its neighbor. De-correlation methods are used to express images in more suitable formats using transforms or prediction techniques. The most common transforms used are "Discrete Cosine Transform" (DCT) [8] and "Discrete Wavelet Transform" (DWT) [9].

*c) Psycho-visual Redundancy:* Human visual mechanism is more sensitive to lower frequencies than higher frequencies i.e. edges more than abundant patterns.

*2) Types of Image data Compression:* Image data compression techniques are divided into two basic categories:

a) Lossless compression
b) Lossy compression

*a) Lossless compression:* In Lossless image compression, the original image can be perfectly reproduced from its compressed representation. Lossless compression schemes exploit coding and statistical redundancies found in images. LZW [10] based coding, Huffman coding and Arithmetic coding are some examples.

*b) Lossy compression:* In Lossy compression, there is some controlled loss of data which results in higher compression ratios at the cost of some loss of quality. JPEG [11] is an example of lossy image compression scheme.

*3) Image Compression Model:* Image compression system consists of the following two main blocks: Encoder and Decoder.

*a) Encoder:* Fig. 1 shows the basic building blocks of a source encoder. Mapper module maps the input image pixels performing the de-correlation using transform or prediction, which reduces the inter-pixel redundancies. The quantizer block limits the accuracy of the mapper output values to pre-determined fidelity criteria. This is the step where major compression takes place. The function of symbol encoder is to present the quantized data using a shorter set of codes.

*b) Decoder:* The decoder performs the reverse function of that of the encoder. However quantization is generally irreversible hence the quantization block is excluded from the decoder as shown in the Fig. 1.

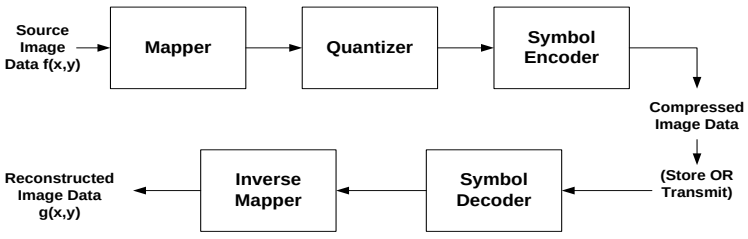


Figure 1. Image Compression Model (a) Encoder (b) Decoder

### B. Literature Review

A research conducted by Faria et al. [12] compares several data compression schemes applied to satellite imagery. The image compression schemes are divided into two broad categories based on the method of de-correlation. These are:

- Prediction based compression techniques.
- Transform based compression techniques.

Differential Pulse Code Modulation (DPCM) [13], JPEG-LS [14] and lossless JPEG [15] are some prediction based techniques and two of the most commonly used transform techniques for image compression systems are:

- Discrete Cosine Transform (DCT)
- Discrete Wavelet transform (DWT)

DCT based compression techniques have been used in space-borne applications. They constitute the JPEG [11] series. DCT based schemes tend to introduce the block artifacts. An advantage which the DWT has over DCT is that it avoids the block artifacts [12]. CCSDS-IDC chosen for this research work is a DWT based algorithm.

An excellent research work carried out by G. Yu et al. [16] surveys about 40 space missions and investigates the image compression techniques used onboard. The research reveals that more than half (55%) of the onboard compression systems are transform based, and 45% of the transform based compression systems are DCT based while 10% are DWT based. According to the research [16], there is a rapid increase of interest in DWT based systems after year 2005. JPEG2000 [17] and CCSDS-IDC are based on DWT. Therefore DWT based compression techniques like CCSDS-IDC and their implementations are modern trends in research of onboard image data compression.

There are different implementation approaches adopted by the researchers. They can be divided into three broad categories [16] i.e. software, hardware and software-hardware.

Hardware implementations are based on ASICs and FPGAs. While software implementation techniques are based on general purpose processors, DSPs etc. FPGA-DSP based implementations combine software and hardware. This DSP implementation of the CCSDS-IDC is based on the software implementation algorithm for the general purpose PC platform by the HongQiang Wong at University of Nebraska Lincoln [18].

According to the review paper [16], around 40% of the onboard image compression system implementations are based on ASICs. This is because of their superior throughput and hardware implementation. But ASICs initial development and implementation cost is a major challenge. This research work proposes the low cost alternative in the form of DSP implementation where existing systems can be adapted to implement the algorithm without major investment in hardware.

A number of researches have been carried out for the implementation design of the CCSDS-IDC on FPGA [19, 20, 21]. But keeping in view the fact that larger the algorithm, the FPGA implementation design complexity increases drastically.

Multiple DSP processors based implementation scheme for onboard image compression carried out by W. Kun and W. Qingyuwan [22] presents a compression scheme similar to the CCSDS-IDC. The multiple DSP platforms are used to provide parallelism and pipelining. The focus is on the improvement of coding efficiency and algorithm speed at the cost of multiple processing units i.e. DSPs. Using multiple DSP platform increases the overall cost and design complexity of the system. Our work is targeted to achieve the DSP implementation of image compression scheme which is cost-effective by using single DSP platform as well as efficient by using a floating point DSP architecture supporting instruction pipelining. The implementation scheme can be migrated to the existing DSP onboard electronics.

The research papers reviewed above suggest that DWT based schemes like CCSDS-IDC, are more efficient at compression and are subject of interest for modern designs. FPGA and ASICs have advantage over DSPs in terms of speed, but their implementations, development costs and their qualification in space are really challenging tasks. This research work is targeted to achieve the DSP based low-cost and flexible implementation of the CCSDS image data compression algorithm.

## III. CCSDS Image Data Compression Algorithm

The algorithm consists of following two functional modules as shown in Fig. 2.

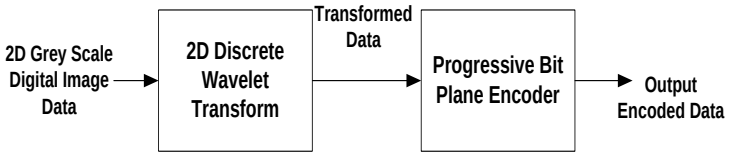


Figure 2. CCSDS-IDC Functional Block Diagram

The algorithm works on the two dimensional digital image data i.e. grey scale images from panchromatic single channel image sensors in space imaging systems. The de-correlation module consists of discrete wavelet transform which is followed by the progressive "Bit Plane Encoder" (BPE) module.

The bit plane encoder produces encoded output bit-stream in the form of a single segment or a series of segments. Each segment has a segment header which is followed by the encoded data.

### A. Specifications of the Discrete Wavelet Transform

The recommended standard specifies the two-dimensional, three-level discrete wavelet transform (DWT) of the input digital image data. Two types of DWT may be used.

- Integer DWT.
- Float DWT.

Both types of transforms use 9 filter taps to calculate low-pass output and 7 filter taps to calculate high-pass output

referred to as 9/7 DWT. Float DWT requires floating point arithmetic calculations and provides lossy compression while Integer DWT is a non-linear approximation to float DWT and supports only lossless compression.

Input image data may be of 4,8,12 or 16-bit pixels. Two-dimensional, single-level DWT is obtained by the repeated application of one-dimensional DWT on the rows and columns of the image frame and results into four sub-bands as shown in Fig.3.

The four sub-bands produced after single-level DWT decomposition are called horizontal low pass-vertical low pass (LL), horizontal high pass-vertical low pass (HL), horizontal high pass-vertical high pass (HH), and horizontal low pass-vertical high pass (LH). The remaining correlation in the LL sub-band of the transformed image is further reduced by applying further levels of two-dimensional DWT to the LL sub-band of previous transformation level decomposition, thus producing a multi-level DWT as shown in Fig. 3.

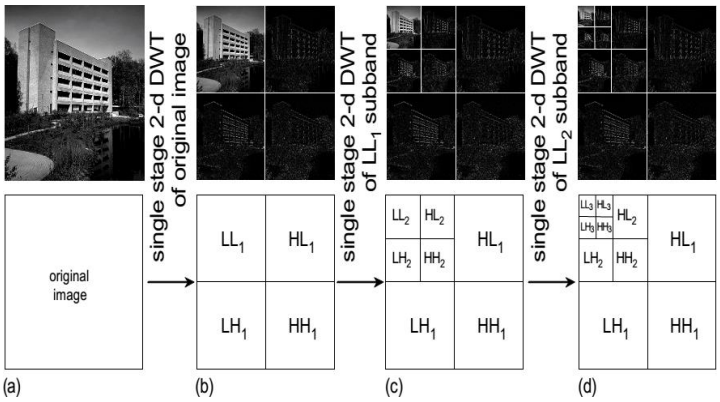


Figure 3. 3-level, 2-dimensional DWT decomposition of an input image [2]

### B. *Specifications of the Bit Plane Encoder*

After the DWT process, the coefficients produced are to be encoded by the bit plane encoder. Before encoding these coefficients are rounded to the closet integer values (float DWT) or scaled by multiplying with the scaling factors (integer DWT).

The Bit plane encoder works on the blocks of wavelet coefficients. Each block consists of 64 wavelet coefficients and maps to a localized region of the image. LL3 sub-band is the most compact and contains the lowest frequency image transformed coefficients also known as DC coefficients. Each level of other sub-bands contains relatively high frequency coefficients also known as AC coefficients.

A block comprises of a single DC coefficient from the LL3 sub-band and 63 AC coefficients which are structured into three families F0, F1, F2, with respect to their sub-bands of origin. A family Fi in the block comprises of 1 parent coefficient, pi, 4 children coefficients Ci, and 16 grand-children coefficients Gi as shown in Fig. 4.

Blocks are arranged together in the form of segments. A segment contains S consecutive blocks. Similarly segments are divided further into gaggles. A gaggle contains 16 blocks.

#### 1) *Bit Planes*

A bit plane consists of the bth bit of each DC coefficient and the bth bit of each AC coefficient magnitude. The Bit plane encoder encodes the bit planes starting from most significant bit plane to the least significant bit plane.

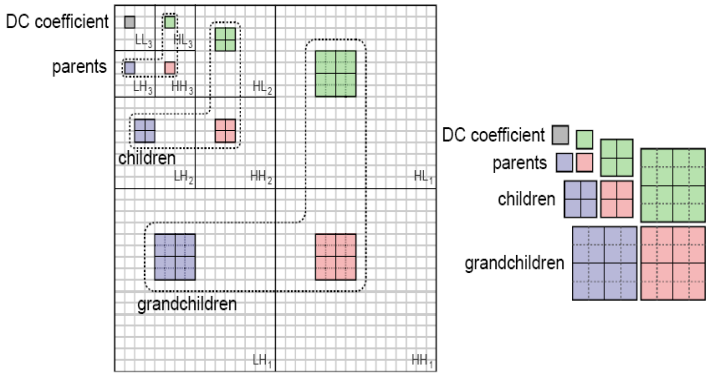


Figure 4. Block and family structure of DWT transformed data [3]

#### 2) *Structure of Single Coded Segment*

Within a single coded segment, first the segment header is coded. After that the quantized DC coefficients are coded, then the AC coefficient bit depths are coded and after that the bit planes of the DWT coefficient blocks are coded as depicted in the Fig.5.

| Segment Header |
|---|
| Initial coding of DC coefficients |
| AC coefficients bit depths |
| Encoded bit plane b = bitDepthAC − 1 |
| Encoded bit plane b = bitDepthAC − 2 |
| …… |
| Encoded bit plane b = 0 |

Figure 5. Structure of a single coded segment

#### 3) *Segment Header*

Segment header consists of four parts as listed below:

- Part I: three or four bytes (compulsory).
- Part II: Five bytes (optional).
- Part III: Three bytes (optional).
- Part IV: Eight bytes (optional).

#### 4) *Initial coding of DC coefficient*

The initial coding of the DC coefficients is performed in two stages:

- Coding quantized DC coefficients.
- Coding additional bit planes of DC coefficients.

#### 5) *Stages of Bit Plane Encoding*

Each bit plane is encoded in multiple stages i.e. Stage (0 to 4).Stage 0 coding for each block is the *bth* most significant bit of each DC coefficient. Stage 1 encodes the bit planes containing magnitudes of parent coefficients in a segment. Stage 2 encodes children coefficients and stage 3 encodes the bit planes containing magnitudes of grand-children coefficients in a segment. Stage 4 encodes the remaining bits of each AC coefficient.

To perform entropy coding on the AC coefficients bit planes, there are two steps involved: For stages 1 to 3, a sequence of binary words having variable lengths are defined which indicate the bits which are going to be encoded in these stages. The second step involves the entropy coding of these variable length binary words adaptively to generate the encoded bit stream. SegByteLimit parameter restricts the number of bytes in the compressed segment. The quality limit dictates the amount of DWT coefficients to be coded. Compression is done until a quality limit or the byte limit is reached. BPE detailed flow chart for DC and AC coefficients encoding is shown in Fig. 6.

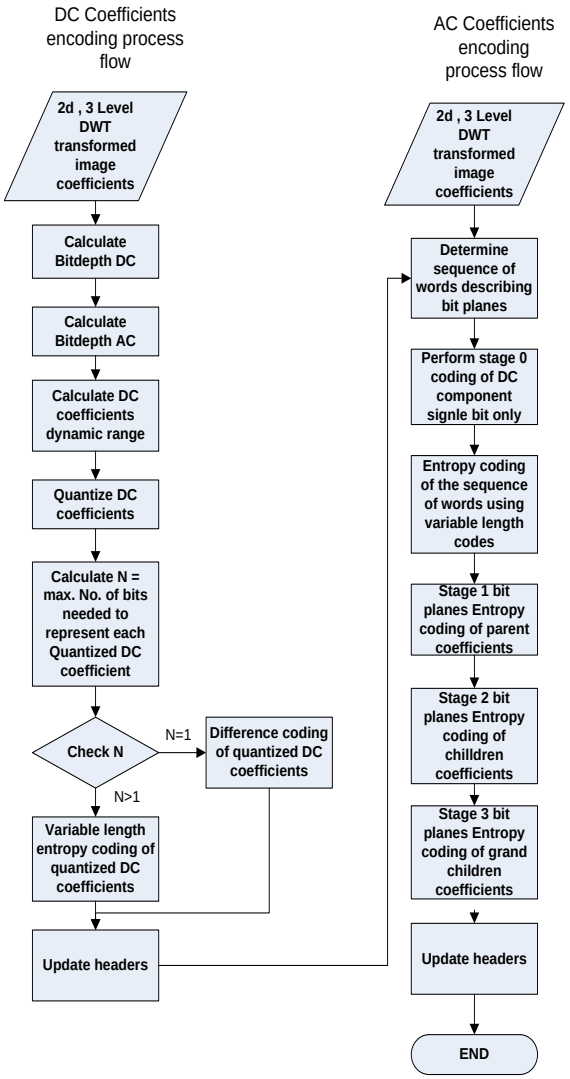


Figure 6 . Bit Plane Encoder flow chart

## IV. Implementation Design

The digital signal processor platform chosen for this project is the DSP Starter Kit (DSK6713). It is built around the floating point digital signal processor; DSP TMS320C6713 by Texas Instruments™. It has onboard 16 Megabytes SDRAM along with audio codec and other onboard resources [23].

DSP TMS320C6713 salient features include operation up to 225 MHz, Very Long Instruction Word Architecture (VLIW), 2 Fixed Point and 4 fixed & floating point ALUs. 2 fixed & floating point multipliers. It has 32-bit wide External Memory Interface (EMIF) to access off-chip memory and a set of control registers for EMIF configuration to select memory type and timing requirements [24]. Multiple ALUs, floating point hardware multipliers capabilities and pipelining architecture make DSP C6713 a suitable choice for image compression task.

### *A. Algorithm Implementation*

The algorithm implementation for this work is based on and adapted from the software implementation by the HongQiang Wong at University of Nebraska Lincoln [18]. The aforementioned implementation code is written in C language for general purpose PC platform and is available as open source for custom implementations. The algorithm is migrated to the DSP platform. As the general purpose PC has extensive memory and computational resources at hand, but the onboard electronics has limited memory onboard and limited computational resources. Therefore the real challenge is to make the code functional efficiently with the available resources. The implementation supports the 8-bit test images of size 256 x 256 pixels, due to limited amount of available SDRAM. And it can be extended to any size up to 16 bit images by increasing the amount of working memory i.e. SDRAM.

The DSP programming environment is called Code Composer Studio™ (CCS). It is an integrated development environment by Texas Instruments™. Code is written in the standard C language i.e. ISO C 1989 [25]. Implementation details are explained in the following sections:

### *B. Implementation Setup*

DSK6713 is configured in a test setup in debug mode. Powered up from a DC power supply, which gives 5 volts DC to the DSK. A USB debug interface connects the DSK to the host computer. The project is compiled in CCS™ software, which generates the output executable (.out) program file. The program file is loaded through USB debug interface into the DSP and SDRAM as determined by the linker command file. The image data should be stored either in the onboard memory or host computer. The implementation setup consisting of host computer, DSK6713, power and USB interface is shown in Fig. 7.

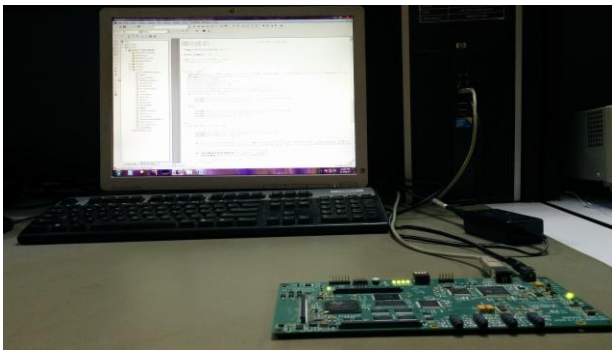

Figure 7. DSK6713 connected to power and USB interface with host computer in lab implementation setup

### C. Input Image Requirements

This implementation supports the input image frame of size 256 x 256 pixels, each pixel having 8-bit integer value i.e. grey scale. The larger image frames can be divided into smaller frames of 256 x 256 pixels each. Compression options are specified at the start of the program which include DWT type, No. of blocks per segment, image dimensions, input and output image bits per pixel etc.

### D. Selection of Compression Options

The algorithm input parameters and compression options' selection dictates the implementation results and compression efficiency explained as follows:

*1) Optional Segment Headers*

Segment header part 1 has the flag bits which indicate the presence or absence of the optional segment header parts 2, 3 & 4. Usually the parameters in optional headers are fixed for an entire image. Therefore only the first segment contains all four header parts but the next segments only contain part 1 of the segment header. This implementation follows the same scheme.

*2) DWT Type Selection*

The selection of DWT type is made through configuration of bit 0 in Segment Header part 4. A value of 0 will select Float DWT and a value of 1 will select Integer DWT.

*3) Control of Image Fidelity and Compressed Data Volume*

There are two limits which control the compressed data volume; byte limit and quality limit:

*a) Byte Limit:* Byte limit constraints the max. number of bytes in each coded segment. The parameter SegByteLimit in header part 2 is used to control the byte limit. Another approach which is used in this implementation is the fixed rate control. The rate control is achieved by specifying a rate in bits per pixel which is then used in calculating the max. number of bytes required for each segment i.e. SegByteLimit.

*b) Quality Limit:* Quality limit constraints the extent of DWT coefficients to be coded in each segment. For this purpose header part 2 contains three parameters to stop coding at an earlier stage i.e. after DC coefficients encoding or after specified bit plane or stage:

*4) No. of Blocks per segment*

The value of S = No. of blocks per segment affects the memory requirements and compression effectiveness. Too small a segment and compression efficiency can be significantly low but small segments help to constraint an error in transmission to that small segment only.

### E. DSP – SDRAM Implementation Architecture

Image processing involves large amount of image data and intermediate arrays, therefore to handle these arrays, sufficient amount of data memory must be available. DSK6713 has onboard SDRAM MT48LC4M32B2 whose configuration is 1 Meg x 32 x 4 Banks [26]. This implementation presents the DSP-SDRAM implementation architecture.

Each memory interfaced to DSP EMIF has its own address range and chip selection signals (CEn). SDRAM is interfaced to CE0 as shown in the Fig. 8(a). SDRAM address, data, clock and control signals are interfaced to the DSP EMIF as shown in Fig. 8(b). The DSP TMS320C6713 has a built-in SDRAM controller, which handles all the read and write timings as well as refresh cycles of SDRAM. SDRAM control register sets up the memory row, column and bank size while SDRAM timing register sets up the read and write cycle timing as well as auto refresh cycle timing. In this case 300 EMIF clock cycles or 6µs is the refresh cycle timing after which a refresh cycle is carried out by the SDRAM controller. In this implementation scheme, SDRAM is partitioned into sections and each section is dedicated for some program or data memory.

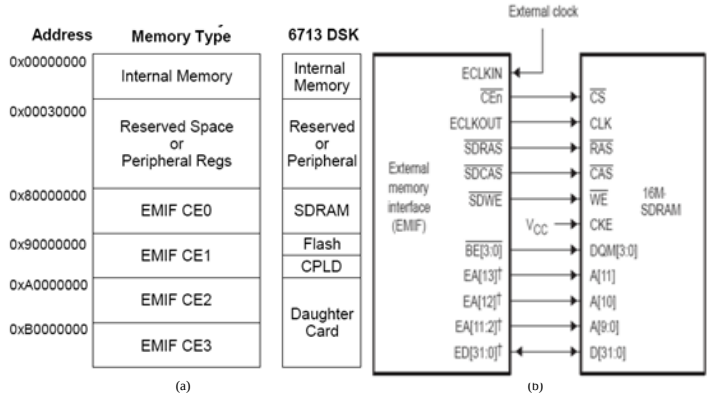


Figure 8 . (a) Memory mapping of DSK6713, (b) DSP - SDRAM EMIF signals connections

*1) Allocation of Sections in Memory*

Linker command file for the implementation of CCSDS-IDC on DSK6713 allocates different sections of code and data to different memory spaces on the DSK6713 as shown in Table 1.

Table 1. Allocation of memory spaces through linker command file

| Memory Space | Description | Origin Address (Hex) | Memory Space Length (Hex) |
|---|---|---|---|
| IVECS | Interrupt Vectors | 0x0 | 0x220 |
| IRAM | Internal memory | 0x00000220 | 0x0002FDE0 |
| SDRAM | External RAM for program and data | 0x80000000 | 0x01000000 |

Table 2. Allocation of code and data sections to SDRAM

| Memory Section | Description | Allocated Memory Space |
|---|---|---|
| .vectors | Interrupt Vector Table | IVECS |
| .text | Executable Code | SDRAM |
| .cinit | Initialized global and static Variables | SDRAM |
| .stack | Stack Memory | SDRAM |
| .sysmem | Heap memory for dynamic memory allocation | SDRAM |

Table 2 shows the important memory sections and their allocation to SDRAM. SDRAM is used as the major working memory. The executable code is allocated to the SDRAM. All initialized global and static variables are stored in the SDRAM allocated section. Two important sections; stack and heap both

have been allocated 3 MB space each. The stack sections stores all the local variables and temporary program data. Heap section used for global data structures for which memory is allocated dynamically. Hence this implementation is solely based on the SDRAM in terms of operating, working and storage memory.

*2) DSP – SDRAM Dynamic Memory Allocation*

The implementation program allocates the memory in run-time. i.e. dynamic memory allocation scheme is used which allocates the required memory in the SDRAM and upon completing the task frees the memory. C language malloc( ), calloc( ) functions are used for allocation of memory and free( ) function is used to free up the allocated memory so that user can assign it again for a different task.

### *F. CCSDS IDC Algorithm Implementation Process Flow*

The implementation of the algorithm can be divided into three main processes:

1) DSP and peripherals initializations process.
2) DWT process.
3) BPE process.

The implementation program starts with initialization of the DSP registers for DSP core and peripherals. Once DSP and its peripherals are initialized, it proceeds to the algorithm.

The algorithm starts with the DWT process where input image frame goes through 3 levels of DWT. The filter coefficients, the transform operations, intermediate working arrays and transformed image frame data are allocated to the SDRAM.

Similarly the BPE process involves coding of the DWT transformed coefficients to produce the coded bit stream. After each step of BPE process as shown in the Fig.9, the output is concatenated to the generated encoded bit stream that is being stored in the SDRAM allocated buffer. The following sections explain the implementation and flow of each of the three processes.

*1) DSP and Peripherals Initialization Process*

Following are the initialization and configuration steps required to setup the DSP in operational mode:

a) Initialization of chip support library for DSP programming.
b) Configuration of DSP, EMIF and peripheral clocks.
c) SDRAM timing and memory interface Configuration.
d) Timer initialization.
e) DSK peripherals initialization.

*2) DWT Process*

As shown in the Fig.9. The input image frame consisting of 256×256 input image pixels is subjected to the 3-level, 2-dimensional DWT process. As discussed in the Section III-A, DWT uses filter banks to generate low pass and high pass output at each level. Hence transforming a signal is equivalent to passing it through a filter bank resulting into transform coefficients. In multi-level DWT it is an iterative process which in turn decomposes the input signal into DWT coefficients. For iterative filtering operation, the Multiply-Accumulate (MAC) feature of the C6713 DSP floating point multiplier unit is exploited and results in efficient calculation of filtered DWT coefficients. Low pass and high pass filtering operations are carried out in parallel by two hardware multiplier units of DSP C6713 in a single instruction using the VLIW pipelined architecture of DSP.

DWT coefficients are buffered in SDRAM. Each row of the input image is taken input at a time instead of whole image frame into the DWT process for efficient memory implementation and for line based camera image inputs. After 1-dimensional DWT 9/7 filter operations, two sets of coefficients are produced, High-pass and Low-pass, each half in size of the original row. Float size (4 bytes each) values are also stored by dynamic memory allocation in SDRAM and memory is freed at each iteration, hence efficiently utilizing the available memory. Intermediate buffer size for DWT operation on a single row is (256+5)*4=1044 Bytes. Transformed coefficients are stored in the allocated buffer in SDRAM. Second dimension DWT is performed on the columns of the 1-d DWT transformed coefficients and the buffer size for transformed image frame is (256*256)*4 = 262144 bytes.

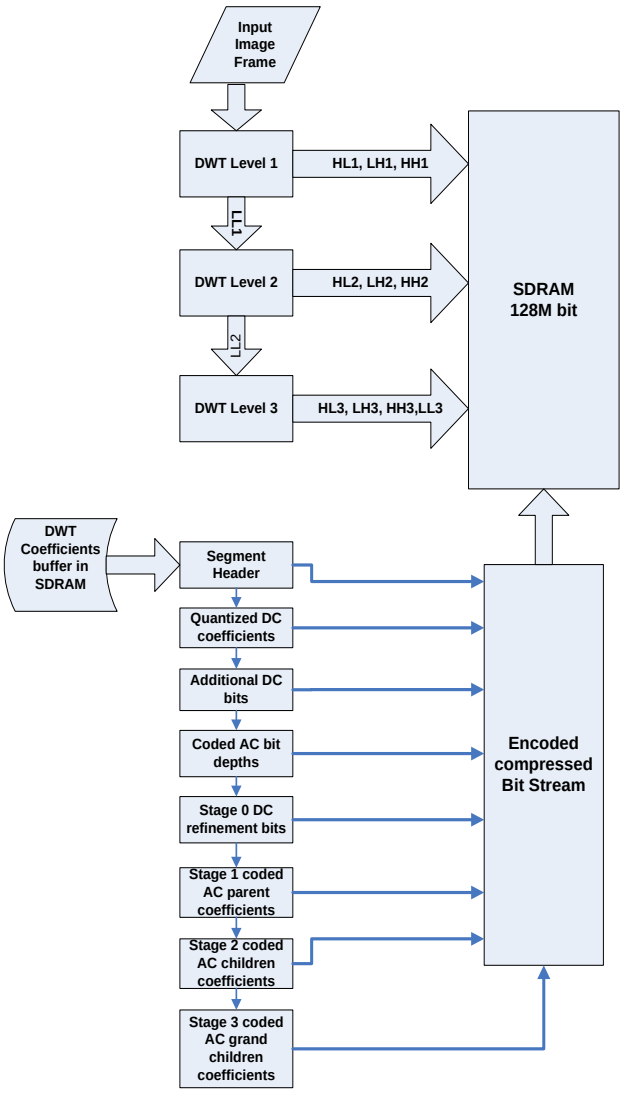


Figure 9. SDRAM memory allocation and buffering scheme for DWT and BPE processes

*3) BPE Process*

The transformed DWT coefficients are then input to the bit plane encoder. The implementation sets aside the block structure by dynamic memory allocation to each block of DWT transformed image frame. BPE process works on the segments of blocks. The bit plane encoder first calculates some basic parameters for the segment to be coded i.e. BitdepthDC and BitdepthAC i.e. maximum no. of bits to represent DC and AC coefficients respectively. Then the DC coefficients are first quantized according to their dynamic range using differential coding. After that the coding of DC coefficient takes place in two steps.

The first step encodes the quantized DC coefficients using the predictive scheme and the remaining q-bitdepthAC bits are encoded in the second step. The further remaining bits of each DC coefficients are encoded in the stage 0 of the Bit plane encoding scheme.

After the DC coefficients encoding, the AC coefficients encoding scheme starts. The parent coefficients bit planes are encoded in the stage 1, after that the children coefficients bit planes are encoded followed by the grand-children coefficients bit planes encoding using the variable length code and entropy coding. Any remaining coefficients are concatenated uncompressed to the encoded bit stream during stage 4. At each and every step of BPE process, the efficient memory management scheme is adopted and dynamic memory buffers are allocated in SDRAM for each coding stage. Memory buffers are freed after the completion of that stage thus available for the next step of the process. The coded data is appended to the bit stream in progressive way. Fig. 9 shows the stages and of BPE process and progressive appending of data into SDRAM buffers.

*G. Source Code Flow and Flow Diagrams*

Flow diagram of CCS-IDC implementation is shown in Fig. 10. The program first checks and validates the compression parameters and image dimensions and in case of successful validation proceeds to the algorithm.

Now the dynamic memory allocation comes into play and assigns memory pointer for the input image data in the SDRAM. Input image is then read from the host computer or memory buffer and stored in the allocated block in SDRAM. Image dimensions are checked for the need of padding rows and columns. If dimensions are not a multiple of 8, then padding rows are added at the end of the rows / columns. Now the program assigns the memory pointer for the transformed image data and calculates the No. of blocks for the image to be transformed.

Once the memory has been set aside for the transformed image, 2-dimensional, 3-level DWT process is started and transformed DWT coefficients are stored in the allocated memory in the SDRAM. After the DWT process the BPE process is started which involves coding of the DC coefficients first as shown in Fig. 9. DC and AC bit depths are calculated for each segment of blocks. After that the quantization of DC coefficients is performed. The coding option for quantized DC coefficients is selected based on the

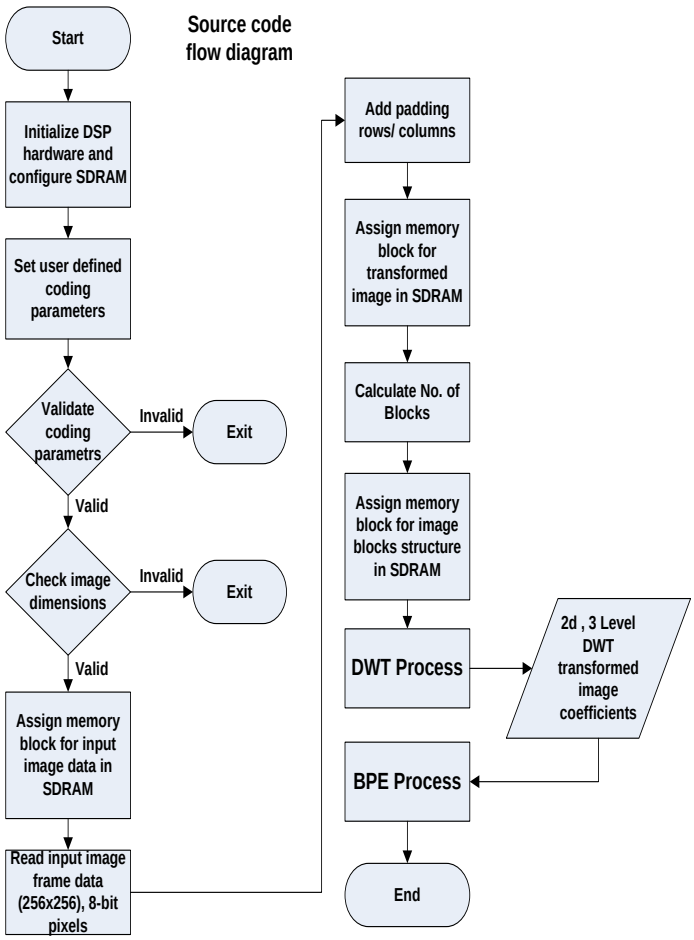


Figure 10 . Source code flow diagram

value of N which results in either difference coding or entropy coding of quantized DC coefficients.

The segment headers are updated after each iteration. Finally the AC coefficients are coded as shown in the flow diagram i.e. Fig 9. To encode the bit planes of AC coefficients, the sequence of words are defined which describe the bit plane values. The sequence of words is entropy coded stage by stage after performing stage 0 coding of DC component single bits. The output coded bit stream is stored in the form of consecutive coded segments in the SDRAM buffer ready for transmission.

## V. Performance Evaluation and Results

The main objectives of CCSDS-IDC algorithm implementation on the DSP platform DSK6713 are: to prototype a practical working system, efficiently utilizing the onboard processing and memory resources, and validate the performance of the algorithm. The test images are subjected to the lossy compression scheme on DSP platform DSK6713 and the results are recorded under different criteria. This section discusses the compression experiments, results and evaluates performance of the DSP implementation scheme.

### A. Test Images

CCSDS data compression working group has compiled a set of test images which contain solar, planetary, radar, and a variety of space images. A subset of these images has been used to test the performance of DSP implementation scheme. The size of test images is customized where required to fit in the available memory onboard DSP platform. The implementation has been evaluated for the 8-bit test images of size 256 x 256, due to limited amount of available SDRAM.

The list of test images is shown in Table 3. These images are copy right of their respective owners and are available free for use at http://cwe.ccsds.org/sls/docs/sls-dc/ .

### B. Performance Evaluation Parameters

The performance evaluation of the image compression algorithm is done through measurement and analysis of following parameters:

*1) Bit Rate*

The total number of bits of compressed image divided by the total number of pixels in that image. i.e. bits per pixel in compressed image.

Table 3. Test Images and criteria used for performance evaluation

| **Customized Test images** | **Lunar** | **Europa** | **marstest** | **b7** | **Spot-la** |
|---|---|---|---|---|---|
| **image dimensions (width × height)** | 256 × 256 | 256 × 256 | 256 × 256 | 256 × 256 | 256 × 256 |
| **Bits per pixel** | 8 | 8 | 8 | 8 | 8 |
| **Total No. of Blocks** | 1024 | 1024 | 1024 | 1024 | 1024 |
| **Blocks per segment** | 256 | 64 | 256 | 128 | 64 |

*2) Maximum Absolute Error (MAE)*

The maximum difference between the original pixel values and the reconstructed pixel values.

*3) Mean Squared Error (MSE)*

It is defined as the mean of the squares of errors. MSE gives the variance and bias of the process.

*4) Peak Signal to Noise Ratio (PSNR)*

It is the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation [27]. It is measured in dB.

### C. Performance Evaluation as a function of Bit Rate

PSNR is the measure of image quality and the MAE and MSE are the measures of distortion. The term rate-distortion performance is used to evaluate the image distortion as a function of rate.

The test images listed in Table 3 are subjected to the performance evaluation under lossy compression scheme. The PSNR, MAE and MSE values are recorded after compression for each of the images at different bit rates.

### D. Selection of Compression Options

To achieve lossy compression following parameters are selected as recommended by the CCSDS [3]:

*1) DWT Type*

Float DWT

*2) Optional Segment Headers*

First segment contains all optional segment headers but the remaining segments do not contain optional headers.

*3) No. of Blocks per segment*

S is varied between 64 blocks per segment to 256 blocks per segment to evaluate compression performance at different bit rates.

*4) Compression Control*

Fixed rate compression is achieved by calculating *SegByteLimit* parameter to fit the desired compression rate.

All other parameters are taken to their default values as defined by the CCSDS-IDC standard. Images are subjected to the lossy compression at the bit rates of 0.25, 0.5, 1.0 and 2.0. The results are plotted to verify the compression effectiveness. Rate distortion performance is analyzed by plotting MAE parameter as a function of bit rate. PSNR parameter is also plotted as a function of bit rate achieved.

### E. Compression Results

Table 4 lists the results obtained after compression at different bit rates for test images. The values are calculated after reconstruction of the compressed image on host computer and then solving for the PSNR, MAE and MSE parameters.

Table 4. Performance Evaluation results

| | | **DSP Implementation results** | | | **CCSDS Published results** | |
|---|---|---|---|---|---|---|
| **Test Image** | **Compressed image Bits per pixel** | ***PSNR*** | ***MAE*** | ***MSE*** | ***PSNR*** | ***MAE*** |
| lunar | 0.25 | 27.66 | 79 | 111.41 | 27.71 | 82 |
| | 0.5 | 30.60 | 48 | 56.58 | 30.92 | 49 |
| | 1.0 | 34.90 | 27 | 21.00 | 35.41 | 26 |
| | 2.0 | 40.89 | 14 | 5.29 | 41.78 | 14 |
| marstest | 0.25 | 27.55 | 59 | 115.17 | 27.35 | 80 |
| | 0.5 | 30.84 | 43 | 53.96 | 30.45 | 46 |
| | 1.0 | 35.08 | 26 | 20.33 | 34.67 | 26 |
| | 2.0 | 41.29 | 12 | 4.86 | 41.19 | 16 |
| Spot-la | 0.25 | 30.37 | 78 | 59.67 | 30.54 | 80 |
| | 0.5 | 32.80 | 45 | 34.07 | 32.82 | 48 |
| | 1.0 | 36.17 | 36 | 15.70 | 35.91 | 27 |
| | 2.0 | 41.72 | 12 | 4.36 | 41.12 | 13 |
| europa | 0.25 | 19.72 | 141 | 693.13 | 18.98 | 149 |
| | 0.5 | 22.15 | 98 | 395.78 | 21.53 | 112 |
| | 1.0 | 25.72 | 73 | 173.99 | 24.95 | 84 |
| | 2.0 | 31.49 | 31 | 46.13 | 31.05 | 39 |
| b7 | 0.25 | 36.88 | 34 | 13.31 | 39.32 | 37 |
| | 0.5 | 39.60 | 22 | 7.12 | 42.43 | 24 |
| | 1.0 | 42.68 | 16 | 3.50 | 46.00 | 11 |
| | 2.0 | 47.46 | 6 | 1.16 | 50.43 | 5 |

The results show increasing trend in PSNR values with the increase in compressed bit rate. On the other hand the MAE and MSE decrease gradually with the improvement in compressed image bit rate. Hence greater the compressed image bit rate, better the quality of the image as well as errors on lower side. These results give intuitive verification of the

CCSDS-IDC algorithm implementation and endorse the results published by the CCSDS [3].

The performance evaluation of the algorithm on test images and analysis of results show that the algorithm is successfully implemented on DSP platform. Implementation results are plotted against the compressed image bit rate. The PSNR gets better with the increase in compressed image bit rate as shown in the PSNR curves for test images in Fig. 11. The rate distortion curves in Fig. 12 depict that the errors or distortion decreases with the increase in compressed image bit rate. The visual quality of the images at low bit rates is quite good. Bit rates of even as low as 0.25 and 0.5 bits per pixel exhibit very good visual quality which is why the CCSDS image data compression algorithm has been chosen. DSP implementation just opens up room for more customization and improvements.

The snapshots of original and compressed images at different bit rates are shown in Fig.13.

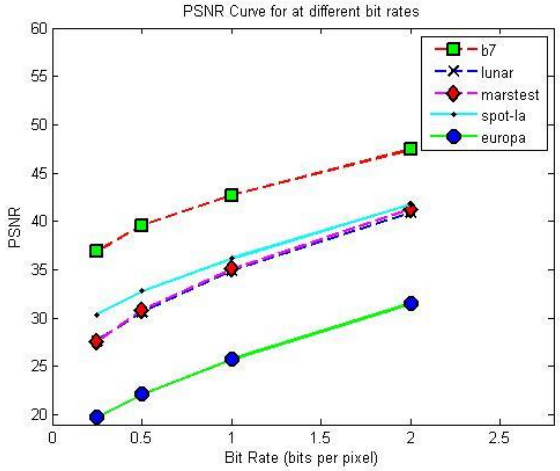


Figure 11 . DSP implementation performance evaluation: PSNR Curves for test images as a function of bit rate

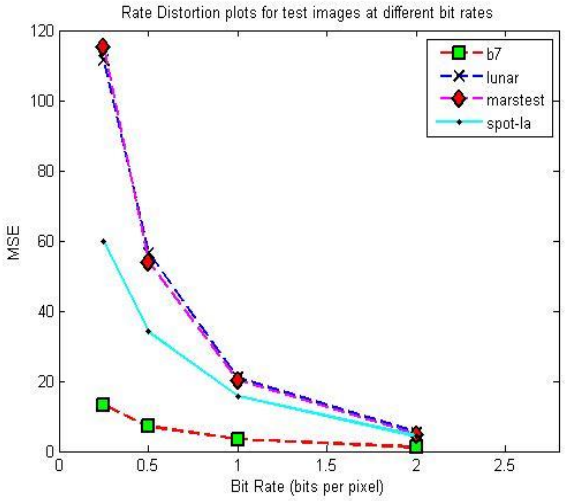


Figure 12. DSP implementation performance evaluation: Rate distortion performance for test images

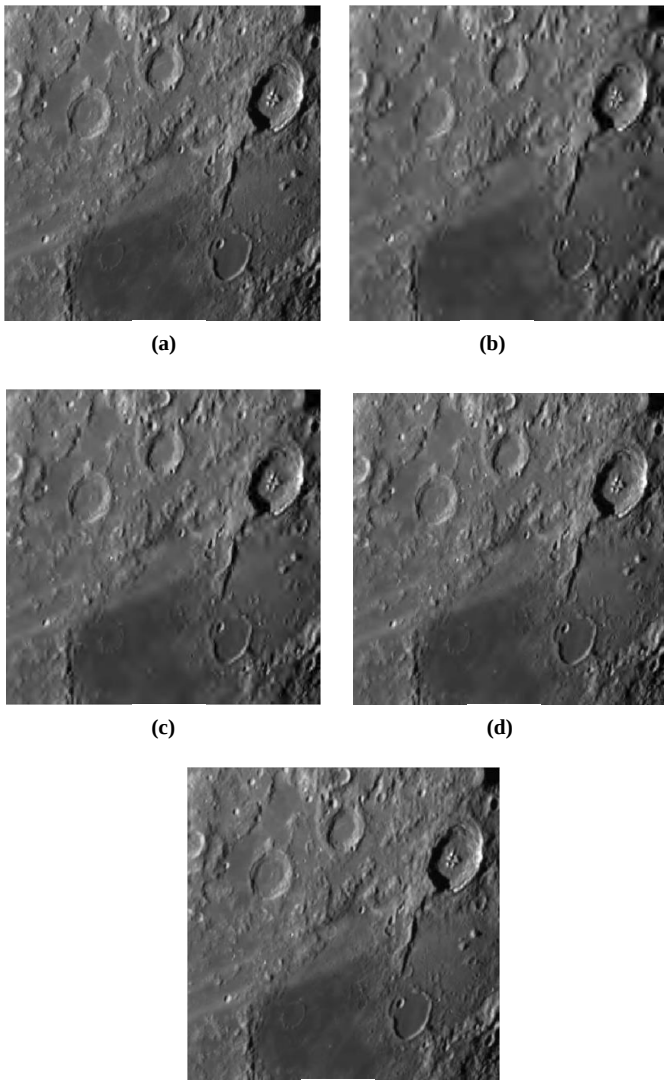


Figure 13. Snapshots of test image "lunar" compressed at different bit rates (a) original test image at 8 bits per pixel, (b) 0.25 bits per pixel , (c) 0.5 bits per pixel, (d) 1.0 bits per pixel, (e) 2.0 bits per pixel

### *F. Throughput Performance Benchmarking*

This implementation scheme is targeted towards buffered image data compression onboard using the readily available hardware resources by embedding the software onboard DSP processor at a certain frame rate or throughput. Compression Throughput performance for 8-bit image frame of 256x256 at the given test conditions for 2 bits per pixel is calculated and given in the Table 5.

CCS™ software offers four levels of code optimizations: level-0 optimization assigns variables to registers. Level-1 performs optimization at local expression and variables. Loop optimizations are performed with parallel operations and pipelining in level 2. Uncalled functions are removed in level-3 optimization. The optimizations are performed up to level-2 in this implementation, and results show an increase in throughput with the better optimization level. The software pipelining and loop level optimization at level 2 improve the throughput significantly by 35%.

Table 5. Throughput performance bench marking results

| **Process Module** | **Function** | **Execution Time (seconds)** | | | |
|---|---|---|---|---|---|
| | | ***Without optimization*** | ***Optim. Level 0*** | ***Optim. Level 1*** | ***Optim. Level 2*** |
| INIT | Initialize DSP hardware and peripherals | 0.01 | 0.01 | 0.01 | 0.01 |
| DWT | 2-dimensional Discrete Wavelet transform | 14.6 | 13.9 | 12.8 | 9.5 |
| BPE (DC Encoding) | DC coefficients encoding | 3.4 | 3.2 | 2.9 | 1.8 |
| BPE (AC Encoding) | AC coefficients encoding | 9.4 | 8.9 | 8.1 | 6.3 |
| Total | | 27.5 | 26.1 | 23.9 | 17.7 |

## VI. CONCLUSION

This research work successfully demonstrates the practical implementation of CCSDS-IDC algorithm on the DSP platform DSK6713. The compression results obtained after the DSP implementation validate the performance of the algorithm on DSP platform. The DSP based implementation is quite flexible and software can be customized to fulfill the requirements of particular application in contrast to the ASIC implementation which gives fixed operation. Combined with the low-cost of DSP hardware, the scheme can be readily implemented on the already existing DSP based onboard hardware in space data systems.

SDRAM based dynamic memory utilization scheme, combined with parallel operations and pipelining architecture of DSP, is both memory efficient and adaptive. Throughput performance of the implementation gives an intuitive data compression solution for buffered image data. Hence DSP based customizable and cost-effective implementation of the CCSDS-IDC is achieved.

There is always room of improvement in embedded computing. The performance enhancement of the DSP implementation may be the next step in improving this work. This implementation tests the 8-bit images of size 256 × 256 pixels. Future projects can implement this algorithm on an extended memory bank DSP platform to remove limitation on the size of the images.

DSP and FPGA combination hardware can be used to achieve the optimum goals of resource utilization and performance efficiency. FPGA can be used to provide buffers and DSP pipelining can improve the efficiency of the algorithm.

## ACKNOWLEDGMENT

The author would like to extend his gratitude to Mr. Muhammad Imran (G.M), and higher management at SUPARCO, Satellite Research and Development Center Lahore (SRDC-L) for their support, valuable inputs and feedback. The author also thanks the Chairman and teachers at the Dept. of Computer Science and Engineering, University of Engineering and Technology Lahore, for their support in the completion of this work.

## REFERENCES

[1] Consultative Committee for Space Data Systems. [Online]. Available: http://www.ccsds.org

[2] Consultative Committee for Space Data Systems, "CCSDS 122.0-B-1: Image Data Compression, Blue Book," Nov. 2005. [Online]. Available: http://public.ccsds.org/publications/archive/122x0b1c2.pdf

[3] Consultative Committee for Space Data Systems, "Image Data Compression CCSDS 120.1-G-1," Jun. 2007, Washington, DC:CCSDS. ser. Green Book. [Online].Available:http://public.ccsds.org /publications/archive/120x1g1e1.pdf

[4] K. C. Wu and Y. W. Tsai, "Structured ASIC, evolution or revolution?" ISPD, pp.103-106, 2004.

[5] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," Communications of the ACM, vol. 30, no. 6, pp. 520–540, 1987.

[6] D. A. Huffman, "A method for the construction of minimum redundancy codes," Proceedings of the Institute of Radio Engineers, vol. 40, no. 9, pp. 1098–1101, 1952.

[7] S. Golomb, "Run-length encoding," IEEE Transactions on Information Theory, vol. 12, no. 3, pp. 399–401, 1966.

[8] K. Cabeen, and P. Gent (July 2014), "Image Compression and Discrete Cosine Transform",[online].Available:http://online.redwoods.cc.ca.us/instruct/darnold/LAPRJ/Fall98/PKen/dct.pdf

[9] C.E. Heil and D.F. Walnut, "Continuous and Discrete Wavelet Transforms," SIAM Review, vol. 31, no. 4, pp. 628-666, 1989.

[10] M . Nelson, "LZW Data Compression," in Dr. Dobb's Journal, Vol. 14, No. 10, October 1989, pp. 29-37.

[11] Gregory K. Wallace, "The Jpeg Still Picture Compression Standard", IEEE Transactions on Consumer Electronics, vol. 38, no. 1, Feb. 1992 pp xvii-xxxiv.

[12] L. Faria, L. Fonseca and M. Costa, "Performance Evaluation of Data Compression Systems Applied to Satellite Imagery," Journal of Electrical and Computer Engineering, Vol. 2012, No. 2012, 2012, 15 p.

[13] N. Moayeri, "A low-complexity, fixed-rate compression scheme for color images and documents," The Hewlett-Packard Journal, vol. 50, no. 1, pp. 46–52, 1998.

[14] ISO/IEC FCD 14495-1, "Lossless and near-lossless coding of continuous tone still images (JPEG-LS), ISO international standard," July 1997.

[15] W. B. Pennebaker and J. L. Mitchell, JPEG: Still Image Data Compression Standard, Springer, New York, NY, USA, 1st edition, 1992.

[16] G. Yu, T. Vladimirova, and M. N. Sweeting, "Image compression systems on board satellites,"Acta Astronautica, vol. 64, no.9-10, pp. 988–1005, 2009.

[17] Information Technology—JPEG2000 Image Coding System, Final Committee Draft Version 1.0, ISO/IEC JTC l/SC 29/WG1 N1646R, March, 2000.

[18] Hongqiang Wang, "CCSDS Image Data Compression C source codes", http://hyperspectral.unl.edu/, University of Nebraska-Lincoln, Sept 2008.

[19] A. A. Khan and Muhammad Arsalan. "Implementation of CCSDS based imaging satellite packet processing unit on FPGA hardware." In Emerging Technologies (ICET), 2012 International Conference on, pp. 1-6. IEEE, 2012.

[20] Albert Lin, "Hardware Implementation of a Real-Time Image Data Compression for Satellite Remote Sensing", Remote Sensing – Advanced Techniques and Platforms, InTech, 2012.

[21] Alberto G. Villafranca, "New Data Compression Algorithms for Space Applications: Software and Hardware Implementations", PhD's thesis, Universitat Politecnica de Catalunya (UPC), 2010.

[22] W. Kun and W. Qingyuan, " Wavelet image compression algorithm based on bitplane coding and optimization implementation on multiple dsp platform," in IEEE International Workshop on Imaging Systems and Techniques, 2009. IST '09.

[23] Spectrum Digital Incorporated, "DSK6713 Technical Reference", Rev. C, 2003.

[24] Texas Instruments, "TMS320C6713 datasheet and manual," 2002. Available: http://www.ti.com

[25] Programming Language − C , ANSI X3.159−1989.

[26] Micron Technology Incorporated, "MT48LC4M32B2 datasheet and manual", Rev. U, April 2013. Available: http://www.micron.com

[27] Wikipedia contributors. (2014, March 4). Peak Signal to Noise Ratio. [Online].

## BIOGRAPHY

**Muhammad Ammar** Received the Bachelors degree in Industrial Electronics from Institute of Industrial Electronics Engineering (IIEE), N.E.D University of Engineering & Technology Karachi, in 2008. He received Masters degree in Computer Engineering from University of Engineering & Technology Lahore in 2014, with the thesis titled: "Implementation of CCSDS Image Data Compression Standard on DSP Platform". His interests include image and signal processing, machine vision, embedded systems, software and hardware implementation and programming in C, C++, JAVA and GUI development. He is a member of Pakistan Engineering Council.

**Dr. Yasir Saleem** is currently serving as an Associate Professor in University of Engineering and Technology (UET), Lahore, Pakistan. His research interests include computer networks, power electronics, digital signal processing and control system. He completed his secondary education (O-level and A-level) form England. He had achieved his Bachelor, Master and PhD degree from Electrical Engineering Department of UET in 2002, 2004 and 2011 respectively. During his PhD he has worked in Energy Conversion Lab, Universiti Technologi Malaysia (UTM) for one semester under supervision of Prof. Dr. Zainal Salam who is Professor in Power Electronics and Renewable Energy, Faculty of Electrical Engineering, Universiti Teknologi Malaysia, Malaysia. He has supervised 10+ Msc theses and 05 PhD scholars are enrolled under his supervision. Currently he has authored / co-authored 20+ journal publications and 10+ conference papers at national and international level in field of Computer / IT and Electrical Engineering.