electronics

MDPI

Article

A 13.3 Gbps 9/7M Discrete Wavelet Transform for CCSDS 122.0-B-1 Image Data Compression on a Space-Grade SRAM FPGA

Elias Machairas [1] and Nektarios Kranitis [1,2,*]

[1] Digital Systems and Computer Architecture Laboratory (DSCAL), National and Kapodistrian University of Athens (NKUA), 157 84 Athens, Greece; sdi1200107@di.uoa.gr
[2] Department of Digital Systems, University of the Peloponnese, 231 00 Sparta, Greece
* Correspondence: nkran@di.uoa.gr

Received: 18 June 2020; Accepted: 27 July 2020; Published: 31 July 2020

Abstract: Remote sensing is recognized as a cornerstone monitoring technology. The latest high-resolution and high-speed spaceborne imagers provide an explosive growth in data volume and instrument data rates in the range of several Gbps. This competes with the limited on-board storage resources and downlink bandwidth, making image data compression a mission-critical on-board processing task. The Consultative Committee for Space Data Systems (CCSDS) Image Data Compression (IDC) standard CCSDS-122.0-B-1 is a transform-based 2D image compression algorithm designed specifically for use on-board a space platform. In this paper, we introduce a high-performance architecture for a key-part of the CCSDS-IDC algorithm, the 9/7M Integer Discrete Wavelet Transform (DWT). The proposed parallel architecture achieves 2 samples/cycle while the very deep pipeline enables very high clock frequencies. Moreover, it exploits elastic pipeline principles to provide modularity, latency insensitivity and distributed control. The implementation of the proposed architecture on a Xilinx Kintex Ultrascale XQRKU060 space-grade SRAM FPGA achieves state-of-the-art throughput performance of 831 MSamples/s (13.3 Gbps @ 16bpp) allowing seamless integration with next-generation high-speed imagers and on-board data handling networking technology. To the best of our knowledge, this is the fastest implementation of the 9/7M Integer DWT on a space-grade FPGA, outperforming previous implementations.

Keywords: CCSDS; Image Data Compression; DWT 9/7M; FPGA

1. Introduction

The huge amounts of data generated from latest and future high-resolution, high-speed imagers in Earth Observation (EO) satellite missions, make image data compression one of the most challenging on-board payload data processing tasks. In 2005, the Consultative Committee for Space Data Systems (CCSDS) issued the Image Data Compression (IDC) standard CCSDS 122.0-B.1 [1]. CCSDS-IDC defines a particular transform-based image data compression algorithm applicable to many types of spaceborne instrument payloads. The recommended standard provides both lossless and lossy (both rate and quality limited) compression, suitable for monoband two-dimensional (2D) images. When compared to the JPEG2000 reference standard for 2D image compression for consumer applications, the CCSDS-IDC recommended standard has lower complexity since it was designed specifically targeting space applications and space systems design under Size, Weight, Power and Cost (SWaP-C) constraints. Both algorithms, rely on progressive Bit-Plane Encoding (BPE) of Discrete Wavelet Transformed (DWT) image data, both providing a choice of integer and a floating-point DWT so that effective lossless and lossy compression can be achieved, respectively. JPEG2000 offers better compression

effectiveness but has higher implementation complexity. Focusing on the DWT, the CCSDS-IDC recommended floating-point DWT is the same 9/7 float as the one included in the JPEG2000. However, the CCSDS-IDC integer DWT is a 9/7M DWT while the JPEG2000 is a 5/3 DWT. Although the 5/3 integer DWT has low complexity and provides excellent lossless compression effectiveness, in tests performed by the CCSDS Multispectral Hyperspectral Data Compression (SLS-MHDC) Working Group (WG) during the development of CCSDS-IDC, the 9/7M integer DWT showed better rate-distortion performance than the 5/3 integer DWT, when other parts of the compression algorithm are fixed. Moreover, in terms of lossless compression performance, the 9/7 integer DWT also performs better according to similar tests. This motivated the CCSDS SLS-MHDC WG to consider two 9/7 integer DWTs, referred to as "9/7F" and "9/7M" following the terminology of [2]. Comparing between the 9/7M and 9/7F filters, the 9/7M provides better compression performance, especially for lossless compression and at high bitrates. The analysis of the computational complexity between the 5/3, the 9/7M and the 9/7F filters [2] using the lifting scheme, shows that the 5/3 filter has the lowest complexity (5 additions, 2 shifts, 0 multiplications). The 9/7M filter has moderately higher complexity than the 5/3 filter (9 additions, 3 shifts, 0 multiplications) while the 9/7F filter has significantly higher complexity (26 additions, 18 shifts, 0 multiplications). For these reasons, the 9/7M filter was considered by the CCSDS SLS-MHDC WG as the best trade-off in terms of complexity and compression performance and was selected as the integer DWT for the CCSDS-IDC recommended standard. Recently, CCSDS updated the CCSDS-IDC recommended standard [3]. The Issue 2 adds modifications to support the Recommended Standard for Spectral Preprocessing Transform for Multispectral and Hyperspectral Image Compression [4]. It supports images with higher dynamic ranges and larger output word sizes (up to 64-bit words).

Application-Specific Integrated Circuits (ASICs) are designed to be optimum for a particular application. Therefore, ASICs for on-board image compression can achieve high performance and low power. There are two ASIC implementations of the 9/7M DWT. An ASIC was developed by Univ. of Idaho and NASA/GSFC in 0.25 um radiation hardened by design (RHDB) technology to implement the CCSDS-IDC DWT [5]. The ASIC is fully immune to Single Event Latch-up (SEL) up to 120 MeV-cm^2/mg and Single Event Upset (SEU) up to 55 MeV-cm^2/mg and achieves a maximum throughput of 20 MSamples/sec at a clock frequency of 100 MHz. The power consumption is 163 mW/MSample and thus requires about 3.3 Watts for the maximum throughput. The CCSDS Wavelet Image COMpression ASIC (CWICOM) [6] developed by Astrium in the framework of an ESA contract is implemented in a radiation tolerant technology (ATMEL ATC18RHA) with internal EDAC, can tolerate Total Ionizing Dose (TID) up to 100 Krad, implements CCSDS-122.0-B.1 and achieves a maximum throughput of 60 MSamples/sec at a clock frequency of 60 MHz. The power consumption is about 100 mW/MSample and thus requires about 6 Watts for the maximum throughput.

Current SRAM FPGA technology offers qualification for space applications, high density, and dynamic partial reconfiguration for in-flight adaptability and time–space partitioning of on-board data processing tasks. Such FPGA technology offers unique advantages over both one-time programmable FPGAs and ASICs and can be considered an excellent platform for implementation of on-board payload data processing due to its ability to support upgrades after launch, greatly enhancing mission profile and extending valuable system lifetime. The Xilinx 65 nm Virtex-5QV SRAM FPGA, currently in use in several space missions and U.S. classified programs, offers RHBD, high density and dynamic partial reconfiguration. Although Virtex-5QV (XQR5VFX130) is a mature product in stable production accumulating heritage since 2014, recently (May 2020), Xilinx announced the launch of the next generation of Radiation Tolerant (RT) space-grade FPGA technology, the 20 nm Kintex Ultrascale XQRKU060 SRAM FPGA. Having performed well in Single Event Effect (SEE) testing and with no identified need for additional RHBD modifications, with no SEL susceptibility seen up to 79.2 MeV-cm^2/mg and good TID test performance up to 100Krad, the Kintex UltraScale device is expected to achieve QML Class Y certification in September 2020.

There are a few existing space-grade FPGA implementations of the 9/7M DWT. In the CCSDS-IDC implementation of [7], the proposed DWT architecture achieves 1.33 cycles/sample. When targeting the Virtex-5QV FPGA, it achieves 136 MSamples/sec at a clock frequency of 191 MHz. When targeting a Kintex 7 (7k325t-ffg676) FPGA, it achieves 227 MSamples/sec at a clock frequency of 318 MHz. In the CCSDS-IDC implementation of [8], the proposed DWT architecture processes 2 samples/cycle and when targeting the Virtex-5QV FPGA, it achieves 200 MSamples/sec at a clock frequency of 100 MHz. In the CCSDS-IDC implementation of [9] for the Formosat-5 mission, the proposed DWT architecture when targeting the Virtex-5QV FPGA, achieves 40.4 MSamples/sec at a clock frequency of 45 MHz.

In this paper, we introduce a high-performance architecture for the 9/7M DWT of the CCSDS-IDC algorithm. The proposed parallel architecture achieves 2 samples/cycle while the very deep pipeline enables very high clock frequencies. Moreover, it exploits elastic pipeline principles to provide modularity, latency insensitivity and distributed control. The implementation of the proposed architecture on a Xilinx Kintex Ultrascale XQRKU060 space-grade SRAM FPGA achieves state-of-the-art throughput performance of 831 Msamples/s (13.3 Gbps @ 16bpp) and outperfoming previous implementations on the same reference FPGA technology (Virtex-5QV). Such a high-performance implementation allows seamless integration with next-generation high-speed imagers and on-board data handling networking technology (i.e., SpaceFibre high-speed serial link).

The paper is organized as follows: In Section 2, a brief description of the CCSDS-IDC algorithm is presented along with the focus of this paper, the 9/7M integer DWT. In Section 3, the proposed architecture for the 9/7M Integer DWT is introduced. The verification and validation strategy of the 9/7M Integer DWT design is described in Section 4. Experimental implementation results and comparisons are provided in Section 5, while Section 6 concludes the paper.

2. Background

2.1. CCSDS-122.0-B.2 Algorithm Overview

In this subsection a brief overview of the CCSDS-122.0-B.2 (CCSDS-IDC) recommended standard is provided. The CCSDS-IDC is a transform-based lossless and lossy compression algorithm and consists of two main functional parts depicted in Figure 1: (a) a Discrete Wavelet Transform (DWT) module that performs the decorrelation and (b) a Bit-Plane Encoder (BPE) which encodes the decorrelated data.

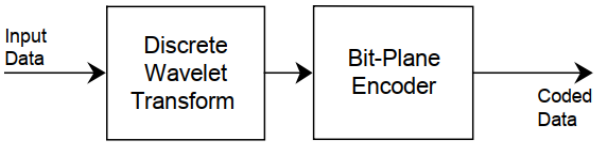

Figure 1. Block diagram of CCSDS-IDC.

The DWT part is a repeated application of a one-dimensional (1D) DWT. Two specific 1D wavelets are specified: the 9/7 biorthogonal DWT, referred to as '9/7 Float DWT' or simply 'Float DWT' and a non-linear, integer approximation to this transform, referred to as '9/7M Integer DWT' or simply 'Integer DWT'. The Float DWT, which is the same as in JPEG2000, can be used only for lossy compression exhibiting better rate-distortion performance. The 9/7M Integer DWT can be used for both lossy and lossless compression. In an implementation where only one option can be afforded, the use of only the 9/7M integer DWT is considered to be the best trade-off. The DWT performs three levels of two-dimensional (2D) wavelet decomposition, producing 10 sub-bands depicted in Figure 2. The subscripts added to LL, HL, HH, and LH (LH indicates horizontal Low-pass, vertical High-pass) denote the level of decomposition. At each level of decomposition, the LL sub-band from the previous level is decomposed, using a 2D DWT, and is replaced with four new sub-bands. Each new sub-band is half the width and half the height of the LL sub-band from which it was computed.

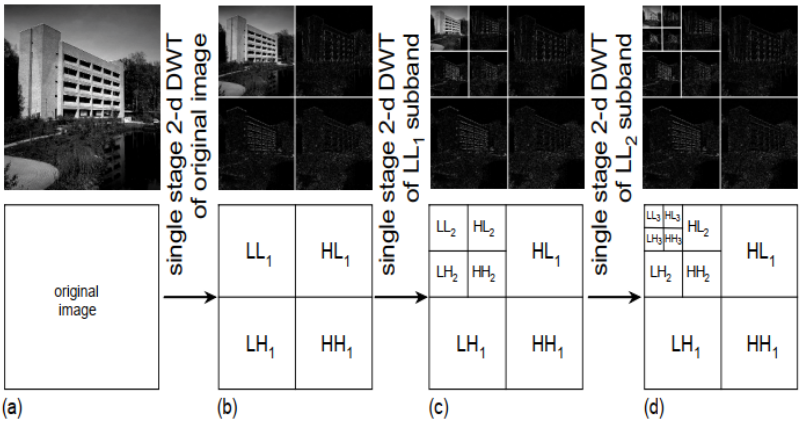

Figure 2. Three-level 2D DWT: (**a**) Original image. (**b**) 1st-level. (**c**) 2nd-level. (**d**) 3rd-level [10].

A single DWT coefficient depends only on a small cluster of pixels. Thus, it is generally not necessary to store a complete image frame of DWT data when full-frame compression is not being performed which is very important when trying to conserve memory resources for large frames or push-broom imagers. To limit the effects of data loss that may occur on the communications channel, the DWT data are partitioned into *segments*, each loosely corresponding to a different region of the image. Each segment is compressed independently, so that the effects of data loss or corruption are limited to the affected segment. Partitioning the DWT data into segments has also the benefit of limiting the memory required. The segment size can be adjusted to trade the degree of data protection and memory requirements for compression effectiveness and rate-distortion performance; smaller segments provide increased protection against data loss and low memory requirements, but tend to reduce the overall compression ratio and reconstructed image quality for lossy compression. Once the coefficients corresponding to a segment have been calculated and placed in the DWT coefficient buffer, the BPE can begin encoding that segment. The focus of this paper is the DWT part of the CCSDS-IDC. The interested reader can refer to the standard itself [1] for full details including the BPE.

2.2. 9/7M Discrete Wavelet Transform

In this subsection, we present the details of the CCSDS recommended 9/7M DWT. Every wavelet transform can be written using the lifting scheme and thus an integer version of every wavelet transform can be constructed. The recommended 9/7M wavelet filter requires one lifting step (i.e., there is only one prediction-update pair). Figure 3 illustrates the layout of the 9/7 lifting scheme showing the prediction, update and symmetrical copy operations while Figure 4 shows 9/7M filter coefficients for the forward lifting process.

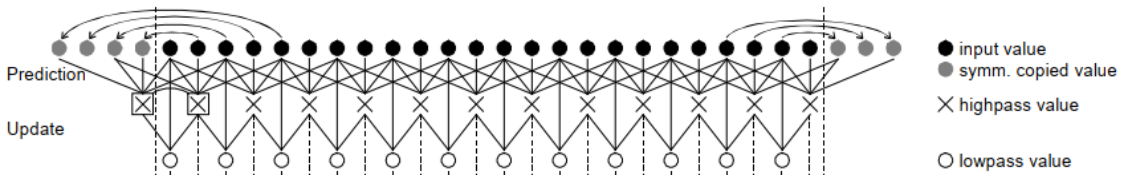

Figure 3. Schematic layout of the 9/7 lifting scheme [10].

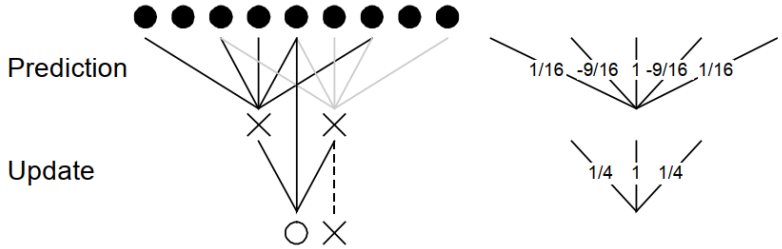

Figure 4. 9/7M filter coefficients for the forward lifting process [10].

The single-level 1D Integer DWT maps an input signal vector x_0, x_1,...., x_{2N-1} into two sets of wavelet coefficients, one high-pass set, D_j and one low-pass set, C_j, in accordance with Equations (1) and (2). Special boundary filters are required for the signal boundaries. Given the input signal x_i, the D_j values are computed first and used subsequently to compute C_j values.

$$\begin{aligned} D_0 &= x_1 - \left\lfloor \frac{9}{16}(x_0 + x_2) - \frac{1}{16}(x_2 + x_4) + \frac{1}{2} \right\rfloor \\ D_j &= x_{2j+1} - \left\lfloor \frac{9}{16}(x_{2j} + x_{2j+2}) - \frac{1}{16}(x_{2j-2} + x_{2j+4}) + \frac{1}{2} \right\rfloor \ , \ for \ j = 1, ..., N-3 \\ D_{N-2} &= x_{2N-3} - \left\lfloor \frac{9}{16}(x_{2N-4} + x_{2N-2}) - \frac{1}{16}(x_{2N-6} + x_{2N-2}) + \frac{1}{2} \right\rfloor \\ D_{N-1} &= x_{2N-1} - \left\lfloor \frac{9}{8}x_{2N-2} - \frac{1}{8}x_{2N-4} + \frac{1}{2} \right\rfloor \end{aligned} \quad (1)$$

$$\begin{aligned} C_0 &= x_0 - \left\lfloor -\frac{D_0}{2} + \frac{1}{2} \right\rfloor \\ C_j &= x_{2j} - \left\lfloor -\frac{D_{j-1} + D_j}{4} + \frac{1}{2} \right\rfloor \ , \ for \ j = 1, ..., N-1 \end{aligned} \quad (2)$$

The computational complexity of the recommended 9/7M DWT is 9 additions, 3 shifts, zero multiplications for a total of 12 operations.

3. Proposed Architecture

The proposed architecture is a line-based DWT architecture which enables a single FPGA chip solution without any external memory requirements. The top-level architecture of the proposed 3-level 2D-DWT is shown in Figure 5. The DWT unit receives two consecutive pixels (odd and even) in parallel and performs three levels of 2D-DWT producing 10 sub-bands. In the proposed streaming architecture, all three cascaded levels of 2D-DWT are operating in parallel, independently of each other, in an elastic pipeline with distributed control to handle back-pressure, with the LL sub-band of each level driving the next level. The independent 2D-DWT computational modules in the elastic pipeline are decoupled with full throughput using small FIFOs. Internally to the 2D-DWT, elasticity is provided using Elastic Buffers (ELBs) between sub-units, also at full throughput. To increase the reusability, testing and ease of maintenance of sub-units, a modular architecture is considered. Towards this direction, the same 2D-DWT VHDL design was reused in all three levels using VHDL generics to configure the different memory sizes required. The 2D-DWT unit is designed to receive two pixels per cycle. This is essential for the first level; however, asymmetric ("gearbox") decoupling FIFOs with ratio 1:2 are required between the two subsequent levels.

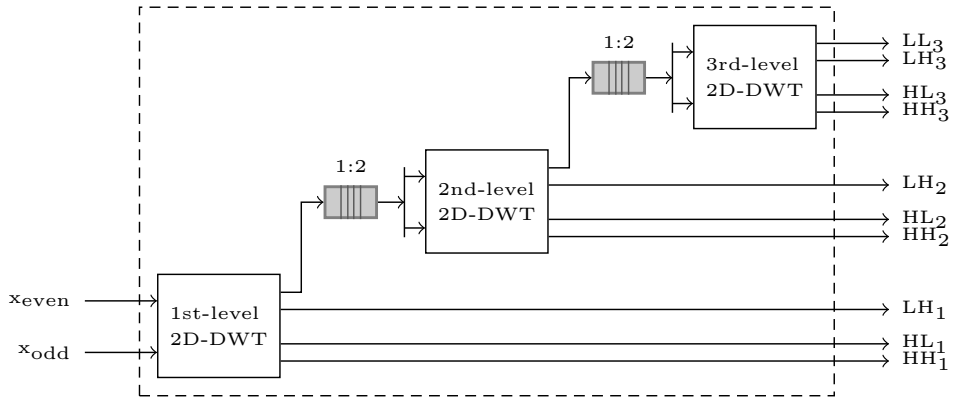

Figure 5. 3-level 2D-DWT Unit.

The input pixels are expected to enter the 3-level 2D-DWT unit in a raster-scan pixel order using an AXI4-Stream compatible interface. The proposed architecture provides flexibility supporting both frame images (produced by CCD array type sensors) and strip-based images (produced by push-broom type sensors). The maximum image width (number of columns) is statically (at compile time) configurable using VHDL generics. The input interface towards the sensor ReadOut Electronics requires end-of-line (EOL) and end-of-stream (EOS) signals accompanying the input data stream. Note that it is easy to provide an external wrapper to generate these 'EOL' and 'EOS' signals based on statically (at compile time) or dynamically (at run-time) configured image dimensions (width and height). An 'EOS' signal is also propagated to the ten sub-band output streams of the 3-level 2D-DWT unit to ease the integration of the proposed DWT design at system-level for a CCSDS-IDC implementation. The ten sub-band outputs are also using an AXI4-Stream interface. In the following subsections, a detailed description of each of the sub-units is provided.

3.1. Architecture of the 2D-DWT Unit

Each 2D-DWT unit consists of one Horizontal and two Vertical DWT units, as seen in Figure 6.

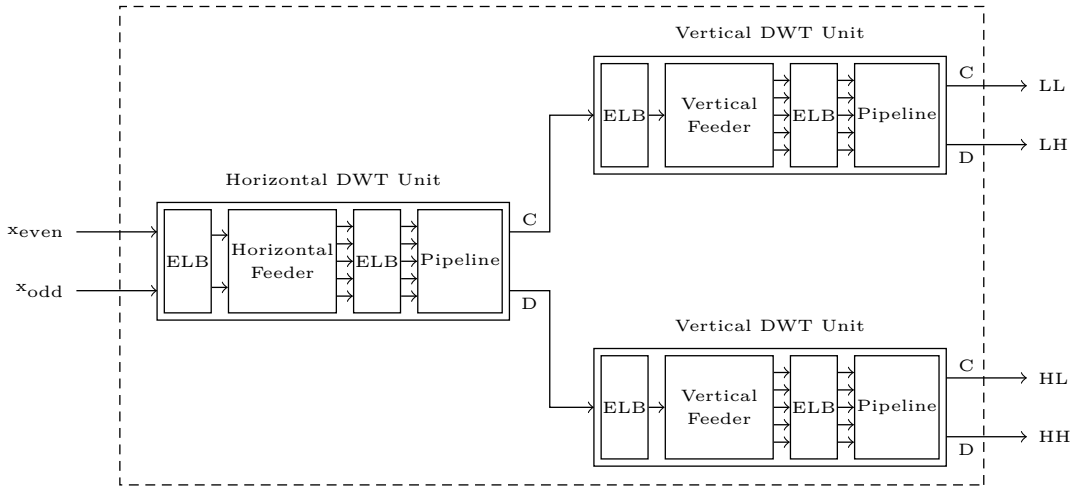

Figure 6. 2D-DWT Unit.

These units also have modular architecture using a common arithmetic pipeline design, since they perform the same 9/7M DWT. However, Horizontal and Vertical DWT units differ in the way they

rearrange the incoming data so that the proper 5-tuples of pixels/coefficients (x_{2j-2}, x_{2j}, x_{2j+1}, x_{2j+2}, x_{2j+4}) enter the arithmetic pipeline. In particular, the Horizontal DWT unit performs the DWT operation on neighbor pixels/coefficients of the same input line, while the Vertical DWT performs the DWT operation on pixels/coefficients residing in the same column relating to the five previous input rows. The unit that generates the proper 5-tuples out of the input stream is called a 'Feeder' (Horizontal or Vertical), as it feeds the right data into the arithmetic Pipeline unit.

Concerning the DWT operation, the input to the $k+1$ level 2D-DWT unit are data rows from the kth LL sub-band LL_k. A row from this sub-band has width equal to $width/2^k$ pixels where $width$ is the maximum number of columns of the input image. The output of the 2D-DWT are data rows of four sub-bands at level $k+1$: LL_{k+1}, HL_{k+1}, LH_{k+1}, HH_{k+1}, where the index $k = 0, 1, 2$ denotes the level of the dyadic decomposition and LL_0 denotes the input image. The Horizontal DWT unit splits each LL_k input into a low-pass and a high-pass stream. Each of these two streams is then passed to the Vertical DWT unit, producing the final LL_{k+1}, HL_{k+1}, LH_{k+1} and HH_{k+1} sub-band streams. Both Horizontal and Vertical DWT units produce coefficients as soon as the corresponding input data become available in a streaming pipeline.

The proposed architecture results in low memory requirements, as the DWT units only have to store and update input data needed for calculating the current output high-pass D_j and low-pass C_j coefficients. It should be noted that the memory requirements are dominated by the Vertical DWT unit where seven rows are buffered on-chip. The memory requirements depend solely on the maximum image width and not on the total image size (independent of the image height) making the architecture suitable for push-broom imagers.

Finally, Elastic Buffers (ELBs) have been used between sub-units, as seen in Figure 6, to improve system timing closure and provide decoupling of independent computational pipelines of the proposed streaming architecture.

3.2. Horizontal DWT Unit

The Horizontal DWT unit is depicted in Figure 6. It has also a modular architecture, and it is composed of two sub-units, each with its own controller: a Horizontal Feeder and an arithmetic Pipeline. The two sub-units are communicating using ELBs. Each of these two sub-units is described in this section.

The Horizontal Feeder is depicted in Figure 7. It receives 2 input pixels (in the 1st DWT level) or coefficients (for subsequent DWT levels) in parallel (x_{even}, x_{odd}) and produces a valid 5-tuple to be used by the arithmetic pipeline for calculating the high-pass D_j and low-pass C_j coefficients. Since the Horizontal DWT processes the inputs sequentially, the filter taps are implemented on FPGA slice registers. Additionally, this unit takes care of the special mirroring conditions, specified in the recommended 9/7M DWT (see Equations (1)), by multiplexing specific slice-register-based filter taps.

The Pipeline unit implements the arithmetic pipeline of the recommended 9/7M DWT. It receives the proper 5-tuples by the Feeder unit and calculates the high-pass D_j and low-pass C_j coefficients. Eight pipeline stages were required to achieve timing closure for the targeted frequency. The datapath of the Pipeline unit is shown in Figure 8.

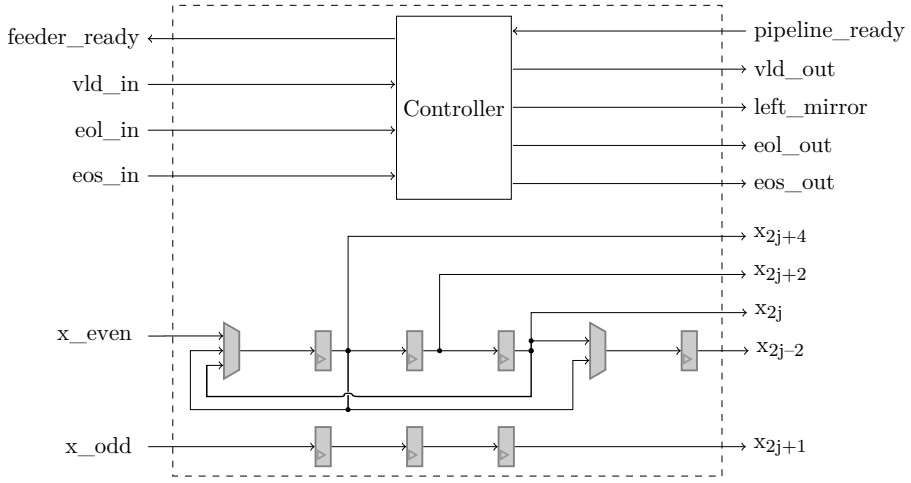

Figure 7. Horizontal Feeder Unit.

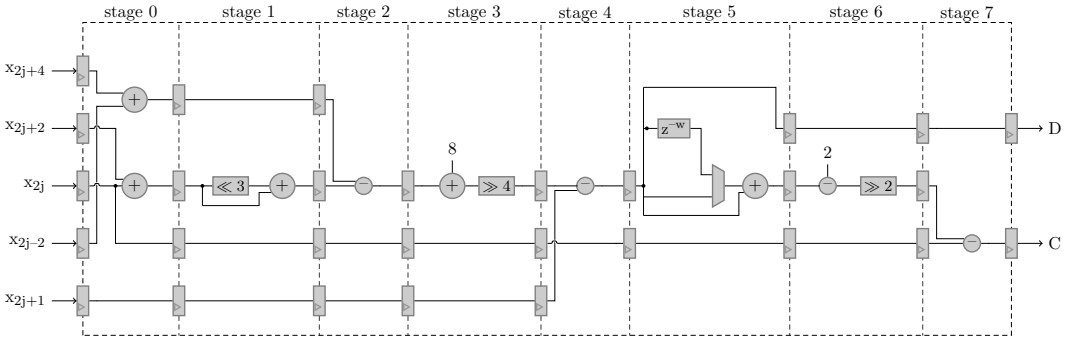

Figure 8. Datapath of the arithmetic Pipeline Unit.

The modularity of the proposed architecture enables sharing the arithmetic pipeline design by both horizontal and vertical units since they perform the same 9/7M DWT. The calculation of each low-pass coefficient C_j of the DWT, requires both current D_j and its previous D_{j-1} high-pass coefficients. However, D_{j-1} is different for horizontal and vertical units, and in particular, in the Horizontal DWT unit is the previous one computed, while in the Vertical DWT unit it is the one computed one line (of width w) earlier. This D_{j-1} coefficient is depicted in Figure 8 in pipeline stage 5 using the z^{-w} configurable delay symbol. The delay is configured statically at compile time using a VHDL generic which dictates whether a simple register (w=*1*) or a FIFO with depth equal to the line width w is instantiated, depending on whether the arithmetic pipeline is meant to be used in a horizontal unit or in a vertical one respectively. Furthermore, in order to calculate the first low-pass coefficient C_0, along with the D_{j-1} coefficient, a specific 'left_mirror' control signal is required that accompanies the input 5-tuples which is set to '1' by the corresponding Horizontal or Vertical Feeder for the 5-tuples corresponding to the first low-pass coefficient C_0 calculation, for 1 cycle or for line-width cycles, respectively. This 'left_mirror' control signal informs the arithmetic pipeline's controller how to produce the select signal of the multiplexer after the z^{-w} element as seen in Figure 8 in pipeline stage 5. The computational complexity of the proposed implementation of the datapath of the arithmetic pipeline unit is 9 additions, 3 shifts and 0 multiplications which is in accordance with the 9/7M algorithm specifications.

3.3. Vertical DWT Unit

The Vertical DWT unit, also depicted in Figure 6, is instantiated two times in the 2D-DWT unit. It uses the same arithmetic Pipeline unit as the Horizontal DWT unit; however, a different Feeder unit is required.

The Vertical Feeder is more complicated than the Horizontal Feeder since the Vertical DWT processes the input rows in parallel. The five input rows required for the Vertical DWT calculation are buffered in BlockRAM-based FIFOs. The Vertical Feeder is depicted in Figure 9. The proposed architecture uses a total of seven FIFOs with depth equal to $width/2^k$ pixels where *width* is the maximum number of columns of the input image. As with the Horizontal Feeder, the vertical one takes care of the special mirroring conditions, specified in the recommended 9/7M DWT (see Equation (1)), by multiplexing specific FIFO-based filter taps.

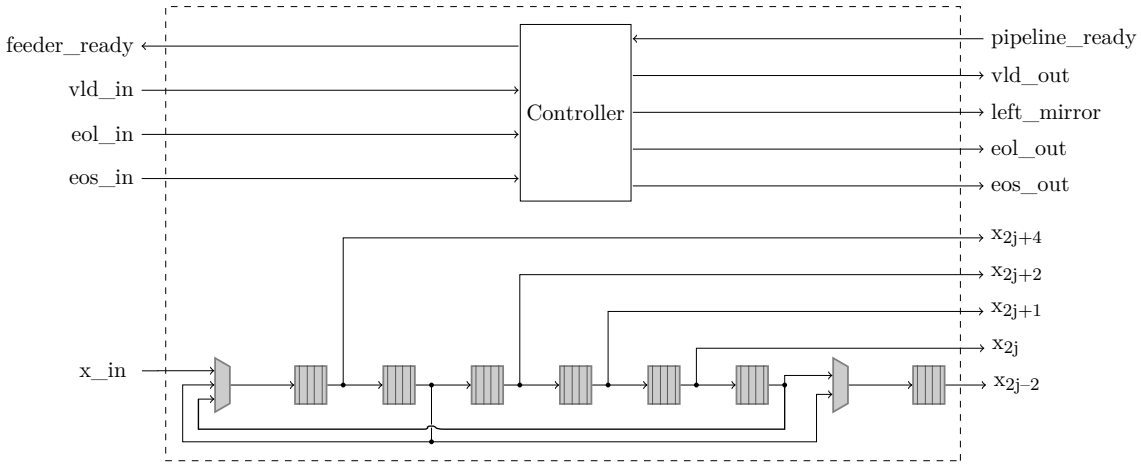

Figure 9. Vertical Feeder Unit.

3.4. Throughput of the Proposed 9/7M DWT

The throughput performance of the proposed 3-level 2-D 9/7M DWT architecture is calculated using Equation (3).

$$Throughput = \frac{width * height}{total\_cycles} * F \quad (MSamples/s) \tag{3}$$

In Equation (3), *width* and *height* refer to the width and height of the image in pixels and F is the achievable clock frequency in MHz. The variable *total_cycles* in the denominator of Equation (3) is the total number of cycles required to perform the DWT operation while considering no stalls from the 9/7M DWT source (considered always valid) and sink (considered always ready) and it can be calculated using the following equation:

$$total\_cycles = \frac{width * height}{2} + 2 * height + 4 + 4 * \left(\frac{width}{2^1}\right) + 4 * \left(\frac{width}{2^2}\right) + 4 * \left(\frac{width}{2^3}\right) + 79 \tag{4}$$

In Equation (4), the first term, $\frac{width*height}{2}$, denotes the number of cycles required for the total number of image pixels to enter the DWT considering receiving two pixels per cycle. The second term, $2 * height$, denotes the additional two cycles per row for the mirroring operation of the 1st level horizontal DWT. The third term, +4, denotes the additional 4 cycles needed for the mirroring operation of the 2nd and 3rd level horizontal DWT units (2 cycles each). The fourth term $4 * \left(\frac{width}{2^1}\right)$ denotes the additional 4 cycles per column for the mirroring operation of the $\frac{width}{2^1}$ columns of the 1st level vertical DWT. The fifth term $4 * \left(\frac{width}{2^2}\right)$ denotes the additional 4 cycles per column for the mirroring operation of the $\frac{width}{2^2}$ columns of the 2nd level vertical DWT. The sixth term $4 * \left(\frac{width}{2^3}\right)$ denotes the additional 4 cycles per column for the mirroring operation of $\frac{width}{2^3}$ columns of the 3rd level vertical DWT. All these stall cycles due to mirroring must be added, since for one unit to start its mirroring

operation, the previous one in the cascade must have finished its own mirroring operation. The last term in Equation (4) denotes the total latency of the six levels of arithmetic pipelines (two levels per 2D-DWT), the latency of the two asymmetric ("gearbox") decoupling FIFOs between the levels of 2D-DWT and the latency of the ELBs between the sub-units of each 2D-DWT for a total of 79 cycles. The total number of cycles can be written as in Equation (5).

$$total\_cycles = \frac{width * height}{2} + 2 * height + 3.5 * width + 83 \tag{5}$$

Thus, replacing the *total_cycles* in the denominator of Equation (3), the throughput can be calculated as in Equation (6).

$$Throughput = \frac{width * height}{\frac{width*height}{2} + 2 * height + 3.5 * width + 83} * F \quad (MSamples/s) \tag{6}$$

For large images (large values of *width* and *height*) or for images produced by push-broom sensors, the fraction term in Equation (6) which denotes the samples/cycle is close to 2 samples/cycle.

3.5. Memory Storage Requirements

The memory storage requirements of the proposed architecture are dominated by the Vertical DWT unit where seven rows must be buffered in seven BlockRAM-based FIFOs hosted in the Vertical Feeder sub-unit. Furthermore, one BlockRAM-based FIFO is also required in the arithmetic pipeline of the Vertical DWT unit to store the row of D_{j-1} coefficients. Thus, eight BlockRAM-based FIFOs are required in each Vertical DWT unit and 16 BRAMs-based FIFOs are required in each of the three 2D-DWT units in cascade. The data width of these FIFOs is 20-bits considering the dynamic range expansion of the integer DWT operation from the 16-bit dynamic range of input data [10]. The depth of the FIFOs is configured to accommodate a row. The width of the row depends on the level k of the dyadic decomposition being $width/2^k$ pixels where *width* is the number of columns of the input image. Thus, the FIFO depth is $width/2^1$ for the two vertical units of the 1st level, $width/2^2$ for the two vertical units of the 2nd level and $width/2^3$ for the two vertical units of the 3rd level.

The memory storage requirements in terms of BlockRAM use is given in Equation (7).

$$\begin{aligned} memory\_storage\_requirements &= \left(16 * \frac{width}{2} + 16 * \frac{width}{4} + 16 * \frac{width}{8}\right) * 20 \\ &= (14 * width) * 20 \\ &= 280 * width \quad (bits) \end{aligned} \tag{7}$$

For example, considering a maximum width of 4096 pixels (this is statically configured at compile time using a VHDL generic) the memory storage requirements are 1146880 bits.

4. 9/7M DWT Verification and Validation Strategy

The 9/7M DWT VHDL design has been extensively verified by RTL simulation using Mentor Graphics Questa. The tests performed include a significant amount of test images from the corpus of images [10] available in [11], the artificial test pattern image of 288 × 248 pixels that was explicitly designed for testing DWT implementations producing maximum dynamic range expansion in all the sub-bands after DWT processing and a significant amount series of random test images. The UNL software implementation in C [12] was used as a golden reference model of 9/7M DWT.

The testing framework is based on VUnit [13] and a set of python scripts, as well as the C golden model of the compressor [12]. Each test campaign is described as a .CSV file with a test on each line and all the static (at compile time) parameters (VHDL generics). Python scripts read the .CSV file and interpret the parameters to invoke the golden compressor binary to produce the verification data.

Then a testbench instrumented with VUnit is invoked with the Questa simulator comparing with golden data.

The 9/7M DWT design has been also validated on-chip. The validation platform was a Zynq-7020 SoC FPGA hosted in a ZedBoard development board by Digilent. The XC7Z020 SoC consists of an integrated processing system (PS) and programmable logic (PL), on a single die. The test architecture was based on the principles of pseudorandom testing where test data images were applied by linear-feedback shift registers (LFSRs) to the 9/7M DWT Design Under Test (DUT) and test responses (ten DWT sub-bands) were compacted by multiple-input-signature-registers (MISRs). The ARM-based PS running bare-metal software was the test controller, responsible for the configuration of the LFSR seed and the monitoring and checking of the ten MISR signatures using AXI4-Lite interface. Moreover, the ARM-based PS was communicating with a host-PC via a UART interface. The user of the host-PC provided several test configurations including different image dimensions and LFSR seeds to the ARM processor test controller. A custom hardware test wrapper was designed to provide the 9/7M DWT design with the LFSR pseudorandom data using the AXI4-Stream interface along with the necessary side-channel control EOL and EOS signals. An extensive on-chip test campaign applied several thousands of successful pseudorandom tests.

5. Experimental Results

The proposed architecture was implemented targeting the XCKU060-1FFVA1617l pin compatible commercial equivalent of the upcoming XQRKU060 space-grade FPGA. The throughput performance goal for the XCKU060 device was higher than 800 MSamples/sec for a single instance of the DWT being a direct requirement from a major European space industry prime for next-generation on-board compression modules. Moreover, experimental results are also provided for comparisons with other work targeting the space-grade Virtex-5QV (XQR5VFX130) FPGA. The Xilinx Vivado Design Suite and the ISE Design Suite tools were used for the implementation targeting the XCKU060 and the V5FX130 devices, respectively.

The frequency of 416.7 MHz (period 2.4 ns) was achieved for the maximum image width configuration considered (4096 pixels) although higher clock frequencies were possible for smaller image widths than 4096. The proposed architecture processes 2 samples/cycle and thus the guaranteed throughput performance is higher than 815 MSamples/sec for all image dimensions considered. The throughput performance for the 4096 × 4096 image is 831 MSamples/sec (13.3 Gbps @ 16 bpp). The detailed implementation statistics including FPGA resources and Static Timing Analysis after Place and Route, for different image dimension configurations are shown in Table 1.

Table 1. Implementation statistics targeting XCKU060-1 FPGA.

Image Dimensions	**512 × 512**	**1024 × 1024**	**2048 × 2048**	**4096 × 4096**
LUTs	7931	8136	8331	8484
BRAMs	28	28	36	52
Registers	11379	11527	11663	11,807
DSP48E	0	0	0	0
Power (W)	0.759	0.761	0.772	0.793
F_{max} (MHz)	416.7	416.7	416.7	416.7
Clock cycles	133,971	529,924	2,108,420	8,411,140
Samples/cycle	1.957	1.979	1.989	1.995
Throughput (MSamples/s)	815	824	829	831

The available LUTs and BRAMs in the XCKU060 are 331680 and 1080, respectively. Thus, the LUT and BRAM use is less than 2.56% and 4.81%, respectively, for maximum image line-width equal to 4096 pixels. The negligible differences in samples/cycle is due to the Horizontal and Vertical DWT mirroring stalls and the deep pipeline latency. The power consumption statistics were evaluated using the Xilinx Vivado power estimator on the post Place and Route design using default environmental

settings. Comparison with other work targeting the same space-grade V5QV (XQR5VFX130) FPGA is shown in Table 2.

Table 2. Comparisons with other work targeting the same XQR5VFX130 space-grade FPGA.

	Kefalas et al. [7]	**Manthey et al. [8]**	**Lin et al. [9]**	**This Paper**
Image width (pixels)	1920	4096	12,000	4096
LUTs	N.A.	9036	N.A.	9965
BRAMs	N.A.	45	N.A.	60
F_{max}(MHz)	196	100	45	256
Throughput (MSamples/s)	131	200	40.4	511

The proposed 9/7M DWT architecture when implemented targeting the V5QV FPGA achieves a maximum frequency of 256 MHz (period 3.9 ns). Considering the 1.995 samples/cycle achieved for an image width of 4096 pixels, it achieves a throughput of 511 MSamples/s (8.17 Gbps @ 16 bpp). This throughput compares favorably with the implementation of [8], which is the current state-of-the-art also providing with 2 samples/cycle. The proposed architecture sets a new state-of-the-art in throughput performance (speedup of 2.55 targeting the same V5QV FPGA technology), while the FPGA resources are kept as low as possible.

6. Conclusions

In this work, we have introduced a high-performance architecture for the 9/7M Integer Discrete Wavelet Transform of the CCSDS 122.0-B-1 Image Data Compression (IDC) algorithm. The proposed parallel architecture achieves 2 samples/cycle while the very deep pipeline enables very high clock frequencies. Moreover, it exploits elastic pipeline principles to provide modularity, latency insensitivity and distributed control. The implementation of the proposed architecture on a Xilinx Kintex Ultrascale XQRKU060 space-grade SRAM FPGA achieves state-of-the-art throughput performance of 831 MSamples/s (13.3 Gbps @16bpp) allowing seamless integration with next-generation high-speed imagers and on-board data handling networking technology. To the best of our knowledge, this is the fastest implementation of the 9/7M Integer DWT on a space-grade FPGA, also outperforming previous implementations. In future work, we will evaluate the effect of Single Event Upsets (SEUs) on the proposed space-grade FPGA implementation and possible mitigation techniques using fault injections experiments.

Author Contributions: Conceptualization, N.K.; methodology, E.M. and N.K.; software, E.M. and N.K.; validation, E.M.; formal analysis, E.M. and N.K.; investigation, E.M. and N.K.; resources, E.M. and N.K.; data curation, E.M. and N.K.; writing—original draft preparation, E.M. and N.K.; writing—review and editing, N.K.; visualization, E.M.; supervision, N.K.; project administration, N.K.; funding acquisition, N.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH–CREATE–INNOVATE (project code: T1EDK-04298).

Acknowledgments: The authors would like to thank the assigned editor, the guest editors and the anonymous reviewers for their constructive comments and suggestions that significantly improved this paper.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Consultative Committee for Space Data Systems (CCSDS). *Image Data Compression CCSDS 122.0-B-1*; Blue Book; CCSDS: Washington, DC, USA, November 2005.
2. Adams, M.D.; Kossentni, F. Reversible integer-to-integer wavelet transforms for image compression: performance evaluation and analysis. *IEEE Trans. Image Process.* **2000**, *9*, 1010–1024. [CrossRef] [PubMed]

3. Consultative Committee for Space Data Systems (CCSDS). *Image Data Compression CCSDS 122.0-B-2*; Blue Book; CCSDS: Washington, DC, USA, September 2017.
4. Consultative Committee for Space Data Systems (CCSDS). *Spectral Preprocessing Transform for Multispectral and Hyperspectral Image Compression CCSDS 122.1-B-1*; Blue Book; CCSDS: Washington, DC, USA, September 2017.
5. Winterrowd, P.; Orbe, C.; Whitaker, S.; Cameron, E.; Nelson, R.; Maki, G.; Fisher, D.; Yeh, P.S. A 320 Mbps flexible discrete wavelet transform processor for extreme environments. In Proceedings of the 2010 IEEE Aerospace Conference, Big Sky, MT, USA, 15 April 2010; pp. 1–8. [CrossRef]
6. Vitulli, R.; Poupat, J.-L. CWICOM; The new CCSDS image compression ASIC. In Proceedings of the 2012 ESA Workshop on Onboard Payload Data Compression (OBPDC), Barcelona, Spain, 29–31 October 2012.
7. Kefalas, N.; Theodoridis, G. High-throughput FPGA implementation of the CCSDS 122.0-B-1 compression standard. In Proceedings of the 2017 27th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS), Thessaloniki, Greece, 25–27 September 2017; pp. 1–8. [CrossRef]
8. Manthey, K.; Krutz, D.; Juurlink, B. Reconfigurable architecture for real-time image compression on-board satellites. *J. Appl. Remote Sens.* **2015**, *9*, 097497. [CrossRef]
9. Lin, A.; Chang, C.-F.; Lin, M.-C.; Jan, L.-J. Field-programmable gate array implementation of Consultative Committee for Space Data Systems image data compression. *J. Appl. Remote Sens.* **2012**, *6*, 063551. [CrossRef]
10. Consultative Committee for Space Data Systems (CCSDS). *Image Data Compression CCSDS 120.1-G-2*; Green Book; CCSDS: Washington, DC, USA, February 2015.
11. Consultative Committee for Space Data Systems (CCSDS), CCSDS 122 Test Image Set. Available online: http://cwe.ccsds.org/sls/docs/sls-dc/ (accessed on 27 July 2020).
12. University of Nebraska-Lincoln (UNL), Software implementation of CCSDS-122 (IDC) in C. Available online: http://hyperspectral.unl.edu (accessed on 27 July 2020).
13. VUnit testing for VHDL. Available online: http://vunit.github.io (accessed on 27 July 2020).