SHYLOC-E: IMPROVING CCSDS STANDARD COMPLIANT IP CORES FOR ON-BOARD LOSSLESS COMPRESSION OF HYPERSPECTRAL IMAGES

Antonio J. Sanchez[1], Yubal Barrios[1], Lucana Santos[2], and Roberto Sarmiento[1]

[1]*Institute for Applied Microelectronics, Universidad de Las Palmas de Gran Canaria, Las Palmas de Gran Canaria, Spain*

[2]*European Space Research and Technology Centre, European Space Agency, Noordwijk, The Netherlands*

ABSTRACT

There is a growing interest in developing hardware-efficient implementations of compression algorithms for multispectral and hyperspectral sensors on-board flight systems. SHyLoC, composed by a pair of configurable and synthesizable IP cores, implements the CCSDS 121 and 123 lossless compression standards, and has been recently added to the set of the ESA's IP portfolio. At the moment, this implementation does not cover the full functionality described in the CCSDS standards. This work presents the extensions proposed over the SHyLoC IP cores in order to be fully compliant with the full CCSDS standards, while at the same time achieving higher performance in terms of compression efficiency and throughput. On the one hand, the unit-delay predictor defined in the CCSDS-121 standard has been included in the hardware implementation. On the other hand, the extension of the CCSDS-123 IP includes some optimizations of the AHB interface in order to improve the throughput, enabling burst transfers, as well as the implementation of the custom initialization of weights.

Key words: Hyperspectral imaging; Compression algorithms.

1. INTRODUCTION

Hyperspectral imaging has multiple applications for identification, surveillance and navigation purposes. For that reason, the use of hyperspectral sensors in flight systems such as drones, planes and satellites is increasing [1]. Hyperspectral sensors produce large amounts of data that need to be stored and either processed or transmitted. Hyperspectral data processing requires a high computational capacity that is not usually available in on-board systems due to the limitations in terms of power consumption. On the other hand, the limited data transmission bandwidths with the ground stations in comparison with the size of hyperspectral images constitute a bottleneck in this kind of applications, which will be aggravated with the progressive increase in the resolution of hyperspectral sensors. Because of this, on-board compression of hyperspectral data becomes mandatory.

The Consultative Committee for Space Data Systems (CCSDS) has developed several lossless data compression standards specifically designed for space applications [2, 3]. These standards provide efficient compression together with a reduced complexity, which fits well with the limited computational resources available in space systems. Among these standards, the CCSDS-121 constitutes a universal compressor applicable to any kind of digital data, while the CCSDS-123 specifically targets multispectral and hyperspectral images. Both compression standards make use of predictive pre-processing stages, well suited for low complexity implementations.

The European Space Agency (ESA) provides a portfolio of IP cores, which can be used by project partners for future space missions. This IP core portfolio has recently included two IP cores, which consist in hardware implementations of the CCSDS-121 and 123 compression standards respectively, which are known together as SHyLoC [4]. These IP cores are provided as technology independent, configurable and synthesizable VHDL designs and they are capable of working separately as well as jointly. However, the SHyLoC IP cores do not implement the full functionality of the respective CCSDS standards. Namely, the CCSDS-121 IP does not implement the pre-processing stage defined in the CCSDS-121 standard, while the CCSDS-123 IP lacks the custom weight initialization option [5].

This work presents the modifications proposed over the SHyLoC IP cores in order to implement the full functionality defined in the CCSDS standards, while at the same time achieving higher performance in terms of compression efficiency and throughput. On the one hand, the main feature of the CCSDS-121 IP extension is the inclusion of the unit-delay predictor defined in the CCSDS-121 standard, which additionally requires inserting reference samples in the compressed data. On the other hand, the extension of the CCSDS-123 IP implements the custom weight initialization. In addition, the throughput the CCSDS-123 IP can reach is improved in the Band-Interleaved (BI) architectures when an external memory is used to store intermediate results by optimizing the

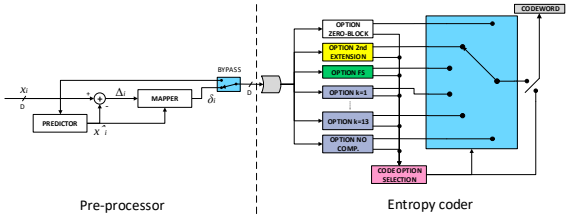

Figure 1. Block diagram of the CCSDS-121 standard

communications between the IP and the external memory. With that goal, the use of the AHB bus is optimized by implementing burst transfers.

The rest of the paper is structured as follows. Section 2 summarizes the CCSDS-121 and 123 compression standards. Then, section 3 introduces the SHyLoC IP cores. Later, section 4 addresses the proposed modifications of the SHyLoC IP cores. Next, section 5 shows the implementation results. Finally, section 6 concludes the paper.

2. CCSDS LOSSLESS COMPRESSION STANDARDS

2.1. CCSDS-121 standard

The CCSDS 121.0-B-2 standard [2] constitutes a universal lossless data compressor. It consists of a preprocessing stage, and an entropy coder. The preprocessor is in charge of predicting the value of each input sample, computing the prediction residuals, and mapping them into values which are then coded by the entropy coder. The block scheme of the full CCSDS-121 standard is shown in Fig. 1. The maximum allowed width (i.e. dynamic range) for the input samples is 32 bits.

The CCSDS-121 standard defines a simple unit-delay predictor to be used as preprocessor. This kind of predictor uses just the previous sample as an estimator of the current one. Therefore, in the case of hyperspectral images, the order in which input samples are processed will affect the prediction residuals, and hence the compression efficiency. Reference samples must be periodically inserted in order to be able to regenerate the input image. In any case, this preprocessor is optional: it can be omitted or replaced with another preprocessor.

The entropy coder of the CCSDS-121 is basically an adaptive Rice coder. The incoming preprocessed samples are grouped into blocks of size J, defined by the user. Each block is compressed with the option which produces the shortest output, among the available ones:

- Fundamental sequence (FS). Each input sample δ_i is encoded as δ_i zeroes followed by a one.

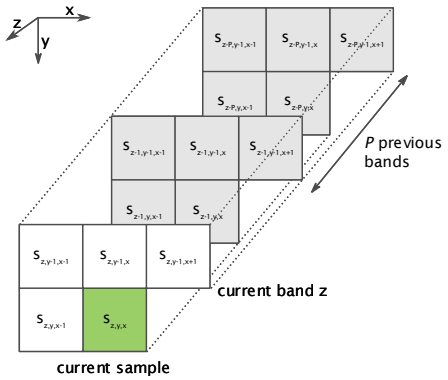

Figure 2. Set of samples used for prediction

- Sample splitting. First, each input sample is split by removing the k least significant bits. The high part of δ_i is coded with the FS, while the low part is left uncompressed.
- Second-extension. Each pair of input samples δ_i and δ_{i+1} is transformed into a new symbol γ, which is coded using FS.
- Zero-block. This option denotes one or more consecutive blocks of all-zeroes. It is the only case where a single codeword may represent more than one compressed block.
- No compression. The input samples are left unaltered.

In any case, a unique identifier is attached to each compressed block in order to know which compression option has been used.

2.2. CCSDS-123 standard

On the other hand, the CCSDS 123.0-B-1 standard [3] is a lossless data compressor specifically devised for multispectral and hyperspactral images. Similarly to the CCSDS-121 standard, the CCSDS-123 is based on prediction techniques.

The preprocessor of the CCSDS-123 predicts the value of each input sample using a neighbourhood of samples around the current sample in the same band as well as in the previous bands, as it is shown in Fig. 2. In this way, the compression efficiency is not affected by the preprocessing order, as opposed to the CCSDS-121 standard. The number of bands P used in the prediction can be configured between 0 and 15.

The predictor processes the input image in a single pass, independently of the chosen order. The compression algorithm is represented in Fig. 3, and summarized next. For each input sample, first a *local sum* for the current

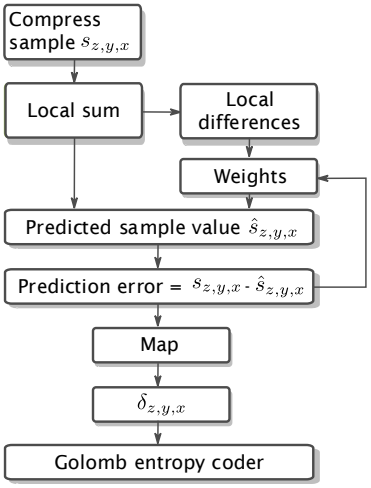

Figure 3. Scheme of the CCSDS 123 compression algorithm

band as well as the previous P bands is computed, which is a linear combination of the neighbour sample values. The set of samples which are used to compute the local sums is determined by the selected local sum type: in the neighbour-oriented mode, all the previously processed neighbour samples are used, while in the column-oriented mode just the sample right on top is used. Then, with the local sums and the neighbour sample values, the *local differences* are computed. These are then grouped in the *local differences vector* $U_{z,y,x}$, whose composition depends on the chosen prediction mode. Under the reduced mode, just the central differences of the P previous bands are used, while in the full mode the directional differences of the current band are also included. The inner product of the local differences vector and a *weight vector* is computed, which is in turn used to compute the *predicted sample* $\hat{s}_{z,y,x}$. The components of the weight vector are updated with each new sample based on the prediction residual, the local differences, and some user-defined parameters. The predictor uses independent weight vectors for each band. The initialization of these weight vectors can be done either by default (the same values for all the bands) or custom (user-specified, each band can be initialized to specific values). Finally, the prediction residual is mapped into an unsigned integer $\delta_{z,x,y}$, which is passed to the entropy coder.

On the other hand, the CCSDS-123 standard allows the use of the entropy coder defined in the CCSDS-121 standard, here known as block-adaptive coder. Alternatively, a sample-adaptive coder is proposed, which is a more sophisticated version of an adaptable Rice coder. With the sample-adaptive encoder, the samples are compressed one by one rather than in groups. The code used for each individual sample depends on some image statistics, namely a counter and an accumulator, which are updated with every new sample according to some user-defined parameters.

As it can be seen, the standard has several options and configurable parameters. These parameters affect the compression efficiency, and they should be set depending on the characteristics of the image and the sensor. The standard also defines a header, which is appended at the beginning of each compressed image, where the selected values for all these options are specified.

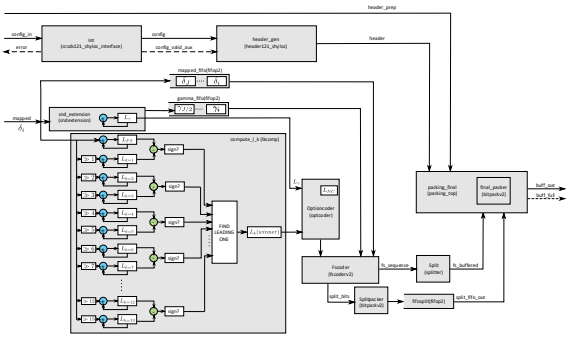

Figure 4. CCSDS-121 IP - Block diagram of the compression core [5]

3. SHYLOC

SHyLoC is a hardware implementation of the CCSDS-121 and 123 compression algorithms. It consists of two IPs, implementing each one of the CCSDS standards respectively. These IPs can work as independent compressors, or jointly. In order to do so, the IPs have been designed with compatible interfaces. In addition, both IP cores include an AHB interface [6] for configuration purposes. SHyLoC is provided as a pair of technology independent, configurable and synthesizable VHDL designs, and it has recently joined the portfolio of ESA IP cores [4].

3.1. CCSDS-121 IP

The CCSDS-121 IP implements the entropy coder of the 121 standard without any preprocessor. It includes all the necessary logic to perform the reception of the runtime configuration values (through AHB) and the input samples (through a dedicated interface), the compression and the flow control of the output bitstream [5].

The compression core of the CCSDS-121 IP is represented in Fig. 4. First, the *int* module reads the runtime configuration values, after being adapted to the IP's clock frequency. It also validates the configuration, arising an error if the configuration values are not correct. The configuration values are then transmitted to the *header_gen* module, which generates the header values according to the configuration and send them to the *packing_final* module. This module splits the output bitstream in words of the size of the output *buff_out* signal. Every time the output buffer is full, a valid flag is generated so that the output value can be captured. The rest of the modules con-

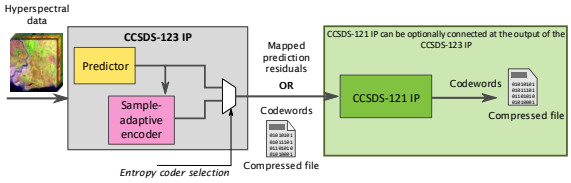

Figure 5. Overview of the SHyLoC IP cores [5]

stitute the compression engine, implementing the entropy coder described in section 2.1.

The *snd_extension* module is in charge of computing the length of a block encoded with the second-extension option, and transforming each pair of input samples δ_i and δ_{i+1} into a new value γ, which is stored in an intermediate FIFO. On the other hand, the *compute_l_k module* computes the length of a block encoded with the FS as well as all the sample splitting options. The number of options to be evaluated depends on the dynamic range of the input samples and the user-selected configuration parameters. The option with the minimum length is stored in a register L_k(winner), and it is compared with the length of the second-extension and no compression by the *optioncoder* module, which selects the best coding option. The *compute_l_k module* also identifies if a block contains all zeroes, and in that case the zero-block option is chosen. Once the encoding option is selected, the *fscoder* module encodes the mapped or gamma values, or the number of zero-blocks, according to the FS sequence, along with the option identifier. Finally, the sequence encoded by the *fscoder*, as well as the uncompressed sample splits, are processed and sent to the final packer to be outputted.

3.2. CCSDS-123 IP

On the other hand, the CCSDS-123 IP implements the predictor as well as the sample-adaptive encoder defined in the CCSDS-123 standard [7]. In addition, it is possible to connect the CCSDS-121 IP core as the block-adaptive entropy coder defined in the standard (see Fig. 5) [4]. In such case, the CCSDS-123 IP is configured to perform just the prediction stage, and both IPs need to be configured independently.

The CCSDS-123 IP implements the necessary components to perform the compression. It includes a configuration core (which is in charge of receiving the configuration from the AHB interface, adapting clock frequencies, generating the header, validating the configuration and disseminating it to the rest of modules), the predictor, the sample-adaptive encoder, a control module and a dispatcher (which receives the output from the compression modules, packets it and sends it to the output) [5]. The CCSDS-123 IP includes two AHB interfaces: one for configuration purposes, and the other to connect an external memory to store intermediate values. Only

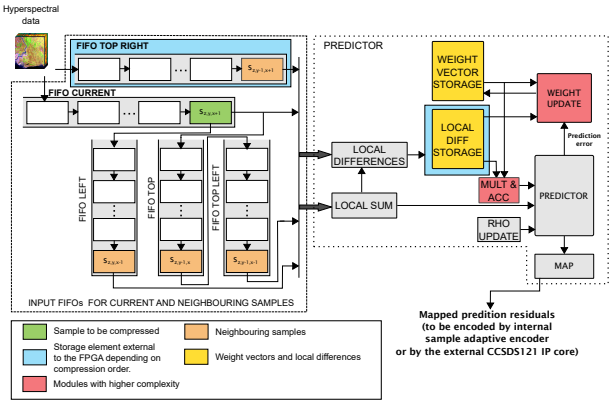

Figure 6. CCSDS-123 IP - Predictor block diagram [5]

single transfers are allowed, which limits the compressor performance when the external memory is used.

The predictor of the CCSDS-123 IP core is able to compress samples in either Band-Interleaved-by-Pixel (BIP), Band-Interleaved-by-Line (BIL) or Band-Sequential (BSQ) orderings. In order to do so, different predictor architectures have been designed with the goal of optimizing the computations for each ordering. A total of 4 architectures have been implemented: BIP, BIP-MEM, BIL and BSQ. The ordering must be therefore selected at implementation time, determining which predictor architecture is loaded. A basic block diagram of the predictor is shown in Fig. 6, which is common to all the architectures. The input samples are arranged in a set of FIFOs, in such a way that the output of each FIFO corresponds to each one of the neighbour samples in the current band. Thus, the amount of samples stored in each FIFO depends on the input ordering, the maximum image dimensions and the number of bands P used for prediction. The predictor performs the computation of the local sums and local differences, the computation of the predicted sample, the mapping and the weights update. In order to save hardware resources, the local differences are stored to be reused in subsequent bands. In addition, in some architectures it is necessary to store the weight vectors when changing the current band. The storage elements which demand the largest memories are highlighted in blue, and can be placed on external memories depending on the selected architecture. Finally, regarding the weights initialization, the CCSDS-123 IP implements just the default initialization defined in the CCSDS-123 standard, not the custom-weight initialization.

The BIP and BIP-MEM architectures take the best advantage on the parallelization possibilities of the prediction algorithm. Provided that there are enough samples in the spectral dimension, these architectures are able to process one sample per clock cycle. Therefore, this is the ordering with the highest possible throughput. In order to do so, the weights update module must be replicated, and the updated weight vectors are stored in a FIFO structure. The particularity of the BIP-MEM architecture is that it stores the contents of the top right FIFO in an external

memory using an AHB master interface.

In the BSQ architecture, it is required to store a complete vector of local differences per sample. All these vectors are stored in an external memory through the AHB interface, and read when required. In addition, in this architecture the data dependencies impose a noticeable throughput limitation. Therefore, the multiply and accumulate operations as well as the weight updating are performed in a serial manner in order to reduce resources utilization.

Finally, the BIL architecture is an hybrid of the BIP and BSQ ones. It inherits most of the components of the BIP architecture, but uses a specific chain of FIFOs in order to store the local differences vectors. In addition, a specific scheduling is devised in order to achieve the highest possible throughput, taking into account that the data dependencies when compressing a line are not the same that when changing between lines.

4. EXTENSIONS OF THE SHYLOC IP CORES

The SHyLoC implementation of the CCSDS compression algorithms lacks some of the features defined in the standards. Namely, the CCSDS-121 IP does not implement the unit-delay predictor, while the CCSDS-123 IP does not implement the custom initialization of weight vectors. In addition, the implementation of the CCSDS-123 standard does not take full advantage of the AMBA AHB bus capabilities when external memories are used to store intermediate results, because burst transactions are not supported [5].

In order to address these issues, an extension of the SHyLoC implementation has been devised. This new version is denoted as SHyLoC-e and was made by means of modifications of the SHyLoC VHDL source code.

The SHyLoC-e IP cores have been validated in simulation against software golden references.

4.1. Extension of the CCSDS-121 IP

On the one hand, the main improvement of the CCSDS-121 IP is the implementation of the unit-delay predictor of the standard, described in section 2.1. It simply consists in a register (holding the value of the previous sample), a subtraction and a mapper, as it is shown in the left side of Fig. 1. The predictor can be bypassed in order to insert reference samples, which must be periodically performed according to the user-defined parameters. The predictor module receives the configuration parameters from the entropy coder.

However, the use of the unit-delay predictor forces the inclusion of periodic reference samples, which must be properly handled by the entropy coder. This imposes some modifications of the entropy coder itself. First of all, the identification of which blocks of samples include a reference sample is implemented in the finite state machine of the entropy coder. Then, the *snd_extension* and *compute_l_k* modules have been modified to correctly compute the length of the compressed block with each compression option if the reference sample is present according to the following rules [2]:

- Fundamental sequence. The reference sample is left uncompressed, so its contribution to the length of the compressed block is related to the dynamic range of the samples rather than its value. The rest of the samples in the block are compressed as usual.
- Sample splitting. Similar to the FS, the reference sample is not compressed nor split.
- Second-extension. The reference sample is independently coded without compression. In order to compute the γ values, the first sample of the block (which corresponds to the reference sample) is replaced by a zero.
- Zero-block. The reference sample is not taken into account when checking the all-zeroes condition.
- No compression. There are no changes.

In any case, the reference sample is coded right after the identifier of the coding option. In order to do so, the *fscoder* module was properly modified. Finally, the *header_gen* module generates the proper preprocessor header field if the unit-delay predictor is used.

On the other hand, the characteristics of the input data have been modified. The maximum dynamic range of the input samples has been extended from 16 to 32 bits, which mainly affects the way the endianness is handled. In addition, due to the inclusion of the predictor, the input samples can be signed, which is properly handled by the predictor itself.

4.2. Extension of the CCSDS-123 IP

In the case of the CCSDS-123 IP, the modifications are focused on improving the communications through the AHB bus. In particular, the BIP-MEM architecture uses an external memory to store intermediate results during the compression, which is accessed through AHB following the scheme of Fig. 7. The input samples are passed to the AHB master, which writes them in memory by properly setting the address and control signals, and later it reads back the samples when they are required as the top right neighbours of subsequent samples. A pair of coupling FIFOs are used to adapt the throughputs of the AHB master and the predictor. There is a gap between the write and read operations in the external memory: the read operations do not begin until the first row of the image with all its bands is loaded in memory, and then the write and read addresses always maintain a gap of one spectral row.

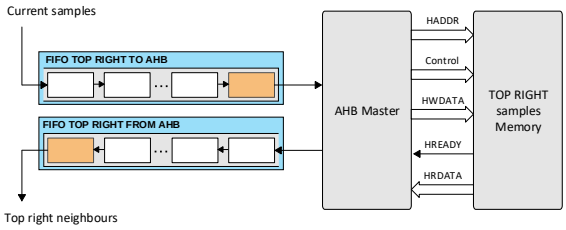

Figure 7. CCSDS-123 IP - External memory interface

The AHB master is in charge of setting the proper read and write addresses on each transfer, and ensuring that there are no data losses.

In the SHyLoC implementation of the CCSDS-123 standard, the AHB master supports only single transfers. This is easy to implement, but it forces the AHB master to request the control of the bus on every transfer, which may impact the overall performance. In the extension, the AHB master has been modified to support incremental burst transactions with a maximum of 16 beats per burst. The read and write bursts are interleaved. The size of the burst transfer is configured at compile time, although the AHB master is able to configure shorter bursts if the execution requires it (for example, when the end of the image is reached).

In addition, a new architecture BIL-MEM has been devised. It processes input samples in BIL order as in the BIL architecture, but uses the AHB master to store the contents of the top right FIFO in an external memory, exactly as in the BIP-MEM architecture. In this architecture, burst transfers are also supported, like in the BIP-MEM architecture of the CCSDS-123 extension.

Regarding the custom weight initialization, a preliminary study has been performed about how it could be implemented. The best alternative consists in reusing the memory which stores the weight vectors for loading the custom initial values, as shown in Figure 8. During the IP

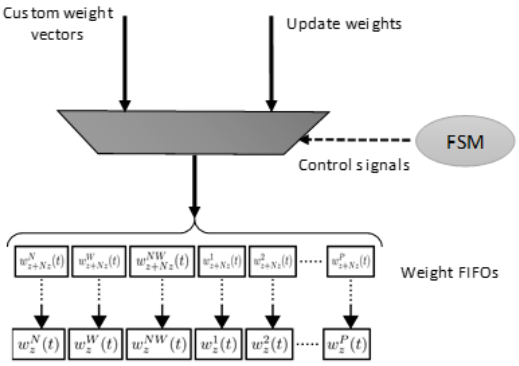

Figure 8. Weight management for custom initialization

Table 1. Implementation results of the CCSDS-121 IP

Device	Resources				Clk. freq. (MHz)
	BRAM	DSP48	LUT	FF	
Virtex5	3	1	3657	1501	118
RTG4	11	3	5419	1347	51.4
NG-MEDIUM	11	5	9371	1639	33.1

configuration, the weight FIFOs are loaded with the custom initial values received by the configuration port. During compression, the weight vectors are cyclically read from the FIFOs, updated and stored again. Alternatively, the weight vectors of the previous compression could be reused as custom initialization values, which can be interesting in the case of image partitioning. In this case, the weight FIFOs would not be reset between executions. However, not all the architectures need to store the weight vectors along the compression, in particular BSQ. For this architecture, the custom weight initialization imposes an extra memory overhead.

5. RESULTS

This section shows some implementation results of the SHyLoC IP cores in different FPGA technologies. These will serve as a reference point for comparison once the SHyLoC-e is implemented in the same FPGA technologies.

First, the CCSDS-121 IP has been implemented in different devices. A specific configuration of the IP has been implemented, with block size J equals to 32, dynamic range D 16, and a width of the output buffer of 32 bits. The target devices were a Virtex5 XC5VFX123T from Xilinx, a RTG4 150 from Microsemi, and the NanoXplore NG-MEDIUM. The implementation results of the CCSDS-121 IP are shown in Table 1. The table shows for each target device the resources utilization in terms of RAM blocks, DSP units, LUTs and flip-flops, as well as the estimated maximum clock frequency. The implementation results for the Virtex5 and RTG4 FPGAs were obtained by means of Synplify software, while for the NG-MEDIUM the NanoXmap tool version 2.8.5 was used. It can be noticed that the Virtex5 implementation achieves the highest clock frequency and consumes less resources than the other two implementations. It must be taken into account that the Virtex5 architecture uses larger logic elements. For example, the Virtex5 architecture uses 6-input LUTs while the other two use 4-input LUTs.

With respect to the CCSDS-123 IP, several configurations have been implemented, adapting the IP to different hyperspectral and multispectral sensors. The configurations differ in the dimensions of the processed images and the dynamic range of the input samples, whose values appear in the Table 2. The image dimensions (Nx, Ny and Nz) are indicated in terms of pixels, while the dynamic range is expressed as bits per pixel (bpp). The rest of the configuration options are shared by all the configurations, where the most relevant are:

Table 2. Set of images used in the implementation

Image	Nx	Ny	Nz	bpp
Landsat	1024	1024	6	8
Aviris	512	680	224	16
Airs	90	135	1501	14
Runtime cfg.	512	1024	256	16

- Number of bands used for prediction, $P = 3$. This value was selected because it has been observed that the compression efficiency does not improve for values of $P > 3$ [7].
- Weight resolution, $W = 13$.
- Neighbour-oriented local sums.
- Full prediction mode.
- Sample-adaptive encoder.
- Width of the output buffer of 32 bits.

Each one of these configurations has been implemented for each one of the four architectures of the IP (BIP, BIP-MEM, BIL and BSQ), which makes a total of 16 implementations.

The different configurations of the CCSDS-123 IP have been implemented in a Xilinx Virtex5 XC5VFX123T as well as a Microsemi RTG4 150. The Synplify software has been used to perform the logic synthesis and to obtain the implementation results for every configuration.

Synthesis results on the Virtex5 FPGA are shown in Table 3. For each configuration-architecture pair, the resources utilization in terms of RAM blocks, DSP units and LUTs is indicated, as well as the estimated clock frequency and throughput. In terms of resources utilization, the main difference between implementations is the memory usage. On the one hand, the memory requirements depend on the dimensions of the target image. On the other hand, the architecture determines if the memory is allocated internally (BIP and BIL architectures) or externally (BSQ and BIP-MEM) to the FPGA. In terms of performance, the BIP architecture is clearly the winner, achieving the highest throughput. This is because the BIP ordering is the one which takes the most advantage of parallelization in the compression algorithm. It can be noticed that the use of an external memory with the BIP processing order (the BIP-MEM architecture) reduces the throughput to approximately the half, in part due to a suboptimal use of the communication bus which is addressed in the extension of the IP. The other two architectures have even lower performance than the BIP-MEM, being the BIL architecture around twice faster than BSQ.

Similarly, the implementation results on the RTG4 FPGA are shown in Table 4. The resources utilization here is expressed in terms of multiply-and-accumulate blocks (MACC), micro SRAM blocks (RAM64), large SRAM blocks (RAM1K) and LUTs. The same tendencies about the resources utilization and performance are observed here. The main difference with respect to the Virtex5 results is that the use of the external memory in the BIP-MEM architecture has a lower impact in the performance with respect to BIP. In addition, it can be observed than the estimated clock frequency and throughput are lower than in the Virtex5 implementations.

6. CONCLUSIONS

This work presents the SHyLoC-e IP cores, an extension of the SHyLoC modules intended to implement some of the CCSDS standard features which the original SHyLoC lacks, as well as improving the throughput and compression efficiency. On the one hand, the unit-delay predictor of the CCSDS-121 standard has been implemented, and the entropy coder of the CCSDS-121 IP has been modified in order to process periodic reference samples. On the other hand, the use of the AHB master interface in the CCSDS-123 IP has been optimized by enabling burst transactions, which improves the throughput of the BIP-MEM architecture. In addition, a new architecture BIL-MEM has been devised, which is based in BIL but uses an external memory to store intermediate results. The implementation of the custom weight initialization has been analysed, and it will be developed in the future.

The next step is the mapping of the SHyLoC-e IP cores in different FPGA technologies in order to obtain implementation and performance results, similar to those obtained for the SHyLoC implementation.

ACKNOWLEDGEMENTS

This project is supported by ESA through the TRP-AO8032 contract.

REFERENCES

[1] Julie Transon, Raphal d'Andrimont, Alexandre Maugnard, and Pierre Defourny, "Survey of hyperspectral earth observation applications from space in the sentinel-2 context," *Remote Sensing*, vol. 10, no. 2, 2018.

[2] The Consultative Committee for Space Data Systems, *Lossless Data Compression, Recommended Standard CCSDS 121.0-B-2*, CCSDS, May 2012, Blue Book.

[3] The Consultative Committee for Space Data Systems, *Lossless Multispectral and Hyperspectral Image Compression, Recommended Standard CCSDS 123.0-B-1*, CCSDS, May 2012, Blue Book.

[4] European Space Agency (ESA), "Shyloc ip core," https://www.esa.int/Our_Activities/Space_Engineering_Technology/Microelectronics/SHyLoC_IP_Core, Accessed: 2018-09-14.

Table 3. Implementation results of the CCSDS-123 IP in the Virtex5

Configuration	Architecture	Resources						Clk. freq. (MHz)	Throughput (Msamples/s)
		BRAM		DSP48		LUT			
Landsat	BIP	2	(0.67%)	8	(2.50%)	3633	(4.43%)	154	154
	BIP-MEM	0	(0%)	7	(2.19%)	4546	(5.55%)	130	81
	BSQ	1	(0.34%)	5	(1.56%)	4398	(5.37%)	113	9
	BIL	5	(1.68%)	7	(2.19%)	4327	(5.28%)	153	28
Aviris	BIP	74	(24.83%)	8	(2.50%)	4675	(5.71%)	131	130
	BIP-MEM	10	(3.36%)	8	(2.5%)	5529	(6.75%)	129	80
	BSQ	1	(0.34%)	5	(1.56%)	5633	(6.88%)	132	10
	BIL	74	(24.83%)	9	(2.81%)	5321	(6.50%)	131	24
Airs	BIP	123	(41.28%)	8	(2.50%)	4564	(5.57%)	132	131
	BIP-MEM	11	(3.69%)	8	(2.50%)	5272	(6.44%)	130	68
	BSQ	1	(0.34%)	6	(1.88%)	5264	(6.43%)	110	10
	BIL	123	(41.28%)	9	(2.81%)	5042	(6.15%)	132	20
Runtime conf.	BIP	74	(24.83%)	10	(3.13%)	5750	(7.02%)	143	142
	BIP-MEM	10	(3.36%)	15	(4.69%)	6486	(7.92%)	124	68
	BSQ	1	(0.34%)	9	(2.81%)	6574	(8.02%)	110	11
	BIL	74	(24.83%)	12	(3.75%)	6211	(7.58%)	121	20

Table 4. Implementation results of the CCSDS-123 IP in the RTG4

Configuration	Architecture	Resources								Clk. freq. (MHz)	Throughput (Msamples/s)
		MACC		RAM64		RAM1K		LUT			
Landsat	BIP	7	(1.52%)	31	(14.76%)	4	(1.91%)	4934	(3.25%)	81	80
	BIP-MEM	7	(1.52%)	33	(15.71%)	0	(0%)	6104	(4.02%)	73	51
	BSQ	7	(1.52%)	25	(11.90%)	1	(0.48%)	5918	(3.90%)	81	6
	BIL	7	(1.52%)	38	(18.10%)	10	(4.78%)	5378	(3.54%)	73	14
Aviris	BIP	13	(2.81%)	62	(29.52%)	129	(61.72%)	7263	(4.78%)	62	62
	BIP-MEM	13	(2.81%)	64	(30.48%)	1	(0.48%)	7658	(5.04%)	74	46
	BSQ	11	(2.38%)	36	(17.14%)	1	(0.48%)	6987	(4.60%)	69	5
	BIL	13	(2.81%)	65	(30.95%)	135	(64.59%)	7552	(4.97%)	62	11
Airs	BIP	7	(1.95%)	20	(9.52%)	278	(133.01%)	7488	(12.27%)	59	59
	BIP-MEM	7	(2.81%)	22	(10.48%)	22	(10.53%)	7331	(6.01%)	59	48
	BSQ	6	(1.95%)	30	(14.29%)	0	(0%)	6967	(5.31%)	74	6
	BIL	7	(1.95%)	30	(14.29%)	275	(131.58%)	7805	(12.77%)	60	11
Runtime conf.	BIP	16	(3.46%)	64	(30.48%)	129	(61.72%)	9398	(10.99%)	61	60
	BIP-MEM	16	(4.55%)	66	(31.43%)	1	(0.48%)	9792	(8.33%)	74	46
	BSQ	14	(3.25%)	38	(18.10%)	1	(0.48%)	9386	(7.03%)	58	4
	BIL	16	(3.46%)	67	(31.90%)	135	(64.59%)	9762	(11.65%)	62	11

[5] University of Las Palmas de Gran Canaria, *SHyLoC Product Datasheet*, Oct 2017, https://amstel.estec.esa.int/tecedm/ipcores/SHyLoC_Datasheet_v1.0.pdf.

[6] ARM, *AMBA Specification*, ARM Limited, May 1999, Rev 2.0.

[7] L. Santos, L. Berrojo, J. Moreno, J. F. Lpez, and R. Sarmiento, "Multispectral and hyperspectral lossless compressor for space applications (hyloc): A low-complexity fpga implementation of the ccsds 123 standard," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 2, pp. 757–770, Feb 2016.