

# Fast and Adaptive Lossless On-board Hyperspectral Data Compression System for Space Applications

Nazeeh Aranki<sup>1</sup>, Alireza Bakhshi<sup>2</sup>, Didier Keymeulen<sup>1</sup>, Matthew Klimesh<sup>1</sup>

<sup>1</sup>Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Dr., Pasadena, CA 91109, USA

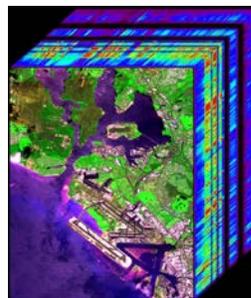
<sup>2</sup>B&A Engineering Inc., 440 South Cataract Ave #8, Sand Dimas, CA 91773

818-354-4285

[nazeeh.aranki@jpl.nasa.gov](mailto:nazeeh.aranki@jpl.nasa.gov)

*Abstract*— Efficient on-board lossless hyperspectral data compression reduces the data volume necessary to meet NASA and DoD limited downlink capabilities. The techniques also improves signature extraction, object recognition and feature classification capabilities by providing exact reconstructed data on constrained downlink resources. At JPL a novel, adaptive and predictive technique for lossless compression of hyperspectral data was recently developed. This technique uses an adaptive filtering method and achieves a combination of low complexity and compression effectiveness that far exceeds state-of-the-art techniques currently in use. The JPL-developed ‘Fast Lossless’ algorithm requires no training data or other specific information about the nature of the spectral bands for a fixed instrument dynamic range. It is of low computational complexity and thus well-suited for implementation in hardware, which makes it practical for flight implementations of pushbroom instruments. A prototype of the compressor (and decompressor) of the algorithm is available in software, but this implementation may not meet speed and real-time requirements of some space applications. Hardware acceleration provides performance improvements of 10x-100x vs. the software implementation (about 1M samples/sec on a Pentium IV machine). This paper describes a hardware implementation of the JPL-developed ‘Fast Lossless’ compression algorithm on a Field Programmable Gate Array (FPGA). The FPGA implementation targets the current state of the art FPGAs (Xilinx Virtex IV and V families) and compresses one sample every clock cycle to provide a fast and practical real-time solution for Space applications.<sup>1,2</sup>

where two of the dimensions are spatial and the third is spectral. A hyperspectral image can be regarded as a stack of individual images of the same spatial scene, with each such image representing the scene viewed in a narrow portion of the electromagnetic spectrum. These individual images are referred to as spectral bands. Hyperspectral images typically consist of hundreds of spectral bands; the voluminous amount of data comprising hyperspectral images makes them appealing candidates for data compression. An example of a hyperspectral data cube is shown in Figure 1. It was taken by the Airborne Visible and Infrared Imaging Spectrometer (AVIRIS), which uses diffraction gratings for band separation with two sets of CCD arrays, one with silicon chips to sense in the visible range and the other with Indium-Antimony (InSb) chips for wavelengths in the Near-IR to Short-Wave-IR range. AVIRIS has 224 detectors (channels) in the spectral dimension, extending over a range of 0.38 to 2.50  $\mu\text{m}$ . This arrangement leads to a spectral resolution for each chip of 0.01  $\mu\text{m}$ . The spatial resolution derived from this depends on the platform height. A typical mission, mounting AVIRIS on a NASA aircraft (ER-2), produces a spatial resolution of about 20 meters, but we can improve that to five meters by flying at lower altitudes, which, of course, narrows the width of the ground coverage [22].



**Figure 1:** An example of a hyperspectral data cube for Pearl Harbor, Hawaii taken by the AVIRIS instrument

Current NASA hyperspectral instruments either avoid compression or make use of only limited lossless image compression techniques during transmission. For example, the current state-of-the-practice is to use the Universal Source Encoder for Space (USES) chip [24]. USES implements the standard lossless compression proposed by the consultative committee for space data systems (CCSDS), which is based on the Rice algorithm [11], and has a multispectral mode, extending its operation to 3D data sets. The USES chip performance has low compression

## TABLE OF CONTENTS

1. INTRODUCTION.....	1
2. ADAPTIVE FILTERING.....	2
3. FPGA IMPLEMENTATION.....	5
4. PERFORMANCE AND BENCHMARKS.....	6
5. SUMMARY.....	7
ACKNOWLEDGEMENTS.....	7
REFERENCES.....	7
BIOGRAPHY.....	8

## 1. INTRODUCTION

Hyperspectral images are three-dimensional data sets,

<sup>1</sup>978-1-4244-2622-5/09/\$25.00 ©2009 IEEE.

<sup>2</sup> IEEEAC paper #1285, Version 3, Updated January 9, 2009

effectiveness as compared to other existing techniques and lacks the flexibility to be efficiently tailored to specific instrument needs, but has the advantage of being currently available in a radiation resistant form. The main reasons for utilization of such devices by NASA are: the limited downlink bandwidth, the need to reduce the risk of corrupting the data-stream needed for accurate science processing, and the lack of a viable on-board platform to perform significant image processing and compression. Future instruments with more sensors and much larger number of spectral bands will collect enormous volumes of data that will far outstrip the current ability to transmit it back to Earth (data rates for some instruments can go to several hundreds of Gbits/s). This gives rise to the need for efficient on-board hyperspectral data compression. Software solutions have limited throughput performance and are power hungry. Dedicated hardware solutions are highly desirable, taking load off the main processor while providing a power efficient solution at the same time. VLSI ASIC implementations are power and area efficient, but they lack flexibility for post-launch modifications and repair, they are not scalable and cannot be configured to efficiently match specific mission needs and requirements. FPGAs are programmable and offer a low cost and flexible solution compared to traditional Application-Specific Integrated Circuit (ASICs).

Exploiting dependencies in all three dimensions of hyperspectral data sets promises substantially more effective compression than two-dimensional approaches such as applying conventional image compression to each spectral band independently. With that in mind, the JPL Fast Lossless hyperspectral compressor was developed. It is a predictive technique that uses an adaptive filtering method and achieves a combination of low complexity and compression effectiveness that far exceeds state-of-the-art techniques currently in use. It will be referred to in this paper as the “Fast Lossless” algorithm.

Fast Lossless algorithm uses an adaptive filtering method and achieves a combination of low complexity and compression effectiveness that is competitive with the best results from literature. Although we are primarily interested in application to hyperspectral imagery, the technique is also generally applicable to any sort of multispectral imagery. The algorithm described in this paper represents a particularly effective way of using adaptive filtering for predictive compression of hyperspectral images. It requires no training data or other specific information about the nature of the spectral bands for a fixed instrument dynamic range. It is of low computational complexity and well-suited for implementation in hardware. This makes it practical for flight implementations of pushbroom instruments.

Section 2 of this paper describes the compression algorithms and equivalent hardware implementation; Section 3 describes the hardware platform. Section 4 describes the results of our initial experiments, and Section 5 summarizes the project results.

## 2. ADAPTIVE FILTERING

### *Algorithm Background*

The JPL’s Fast Lossless encodes data samples one-at-a-time, typically in raster scan order. It uses a form of predictive compression, i.e. sample values are estimated by linear prediction, and the differences between the estimates and the actual sample values are encoded into the compressed bitstream. Only previously encoded samples are used to predict a given sample in order that the prediction operation can be duplicated by the decoder. Estimation of sample values by linear prediction is a natural strategy for lossless compression of hyperspectral images. This is a form of predictive compression, or, more specifically, a form of differential pulse code modulation (DPCM).

Fast Lossless compressor uses the sign algorithm [1], which is a variation of the Least Mean Square (LMS) algorithm[2], a well-known low-complexity adaptive filtering algorithm. The sign algorithm and the LMS algorithm are members of a family of low complexity adaptive linear filtering techniques, which are used extensively in signal processing applications such as audio data compression. However they have not been well studied for image or hyperspectral data compression. A straightforward extension of the LMS algorithm to two-dimensional (2-D) images is well documented in the literature with applications to image processing and application to filtering magnetic resonance imaging (MRI) data. In a few cases researchers have been directly interested in applying the LMS algorithm to image compression. An early example describes a fixed rate, lossy, predictive compression of (2-D) images. There has been a fair amount of work on lossless predictive compression of hyperspectral images that does not involve the LMS algorithm or its relatives. In particular, the methods used by Rizzo et al. [7] have low complexity and yield the compression effectiveness similar to that of our method. Good compression effectiveness results are also reported in the literature by Aiazzi et al. [8], but those results are obtained with methods of moderately high complexity.

### *Algorithm Description and Digital Implementation*

The essence of the Fast Lossless hyperspectral compression algorithm is adaptive linear predictive compression using the sign algorithm for filter adaptation, with local mean estimation and subtraction. We start with a brief description of the LMS algorithm and the sign algorithm. For both of these algorithms a desired signal  $d_t$  is to be estimated from an input (column) vector  $u_{t,k}$ , where  $t$  is an index which increases sequentially and represents the time index in the hardware implementation. The desired signal  $d_t$  is the sample value at spatial location  $(x, y)$  in spectral band  $z$  noted (referred to as  $Z_t$  in Figure 4 of the digital representation). The estimate  $\hat{d}_t$  is a linear function of  $u_{t,k}$ ; specifically,  $\hat{d}_t = w_{t,k}^T u_{t,k}$ , where  $w_{t,k}$  is the filter

weight vector at index  $t$ .  $u_{t,k}$  is represented by the sample values at spatial location  $(x, y)$  in spectral band  $z-k$ , (or  $Zt-k$ ) with  $z=1,2$  and  $3$ , as well as the sample values at neighborhood location  $(y-l,x)$ ,  $(y-l,x-l)$ ,  $(y,x-l)$  in spectral band  $z$ .

After an estimate  $\hat{d}_t$  is computed (referred to as *EC* in Figure 4), the error between the estimate  $\hat{d}_t$  and the desired signal  $d_t$  is computed, specifically,  $e_t = \hat{d}_t - d_t$  (referred to as  $\Delta$  in Figure 3).

This error value is used to update the filter weights. For the LMS algorithm,

$$w_{t,k+1} = w_{t,k} - \mu u_{t,k} e_t$$

For the sign algorithm illustrated in Figure 3,

$$w_{t,k+1} = w_{t,k} - \mu u_{t,k} \text{sgn}(e_t)$$

In each case  $\mu$  is a positive, scalar parameter (the step size parameter) that controls the trade-off between convergence speed and average steady-state error. A small  $\mu$  results in better steady state performance but slower convergence. In some variants of these algorithms the value of  $\mu$  changes over time. The sign algorithm has the property that under certain general assumptions the weight vectors it produces become clustered around the optimum weight vector in terms of minimizing the mean absolute estimation error. For a sufficiently small adaptation step size parameter, the asymptotic mean absolute estimation error can be made to be as close as desired to the minimum possible [6].

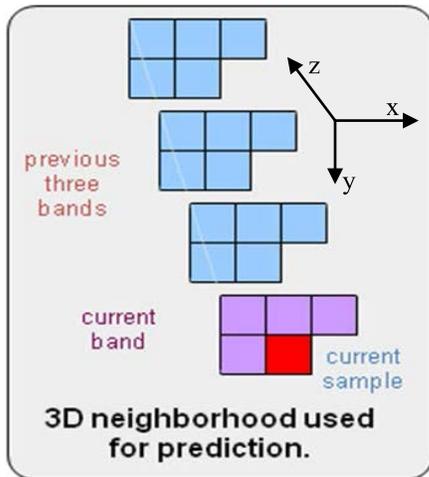


Figure 2: 3D neighborhood prediction

To overcome problems of poor combinations of convergence speed and steady-state performance, a local mean subtraction method was used, motivated by [14]. In our local mean subtraction method, for each sample we compute a preliminary estimate using a fixed, causal, linear predictor involving only samples from the same band (purple cells in Figure 2). The preliminary estimate of sample  $s(x, y, z)$  is denoted by  $\tilde{s}(x, y, z)$  which is the local mean, (LMt in Figure 4), between sample values at  $(y-l,x-$

$l,z)$ ,  $(y-l,x,z)$ ,  $(y,x-l,z)$  and  $(y-l,x+l,z)$ , noted respectively as  $B3$ ,  $B2$ ,  $B1$  and  $BZ$  in Figure 4. For our implementation we use a six-sample prediction neighborhood with three samples from the same band as the sample to be predicted, and one sample each from the three preceding bands (blue cells in Figure 2), all samples are corrected using the local mean subtraction method so that:

$$u_{t,k} = \begin{bmatrix} s(x-1, y-1, z) - \mathcal{R}(x, y, z) \\ s(x, y-1, z) - \mathcal{R}(x, y, z) \\ s(x-1, y, z) - \mathcal{R}(x, y, z) \\ s(x, y, z-1) - \mathcal{R}(x, y, z-1) \\ s(x, y, z-2) - \mathcal{R}(x, y, z-2) \\ s(x, y, z-3) - \mathcal{R}(x, y, z-3) \end{bmatrix} = \begin{bmatrix} Diff3 \\ Diff2 \\ Diff1 \\ Diff4 \\ Diff5 \\ Diff6 \end{bmatrix}$$

is the corresponding input vector. The general rule is to adjust each sample in the prediction neighborhood by the preliminary estimate in the same band as the sample but at the spatial location of the sample being predicted. Since this is done as part of a predictive compression algorithm, the difference  $e_t = \hat{d}_t - d_t$  is encoded in the compressed bit stream using the Golomb-Rice codes [23] with parameters that are powers of 2. The decompressor decodes this difference  $e_t$  from the bitstream, and can compute  $\hat{d}_t$  and  $\mathcal{R}(x, y, z)$  from previously decoded samples, and therefore can reconstruct the value  $s(x, y, z)$ . Further details of the algorithm can be found in [3]

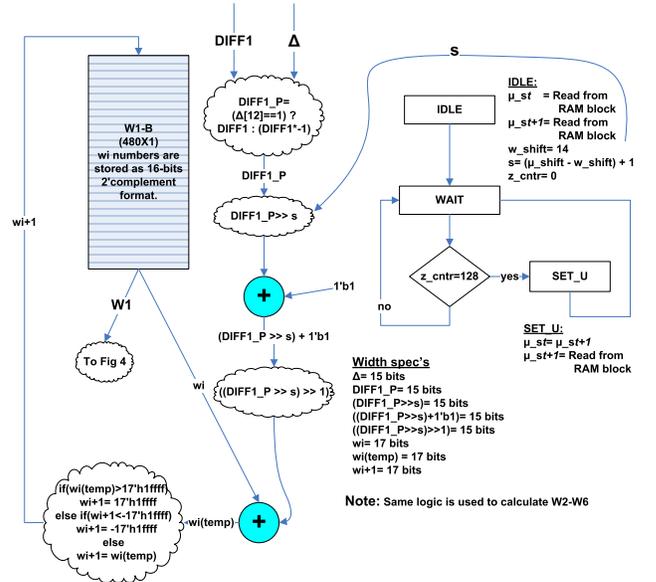
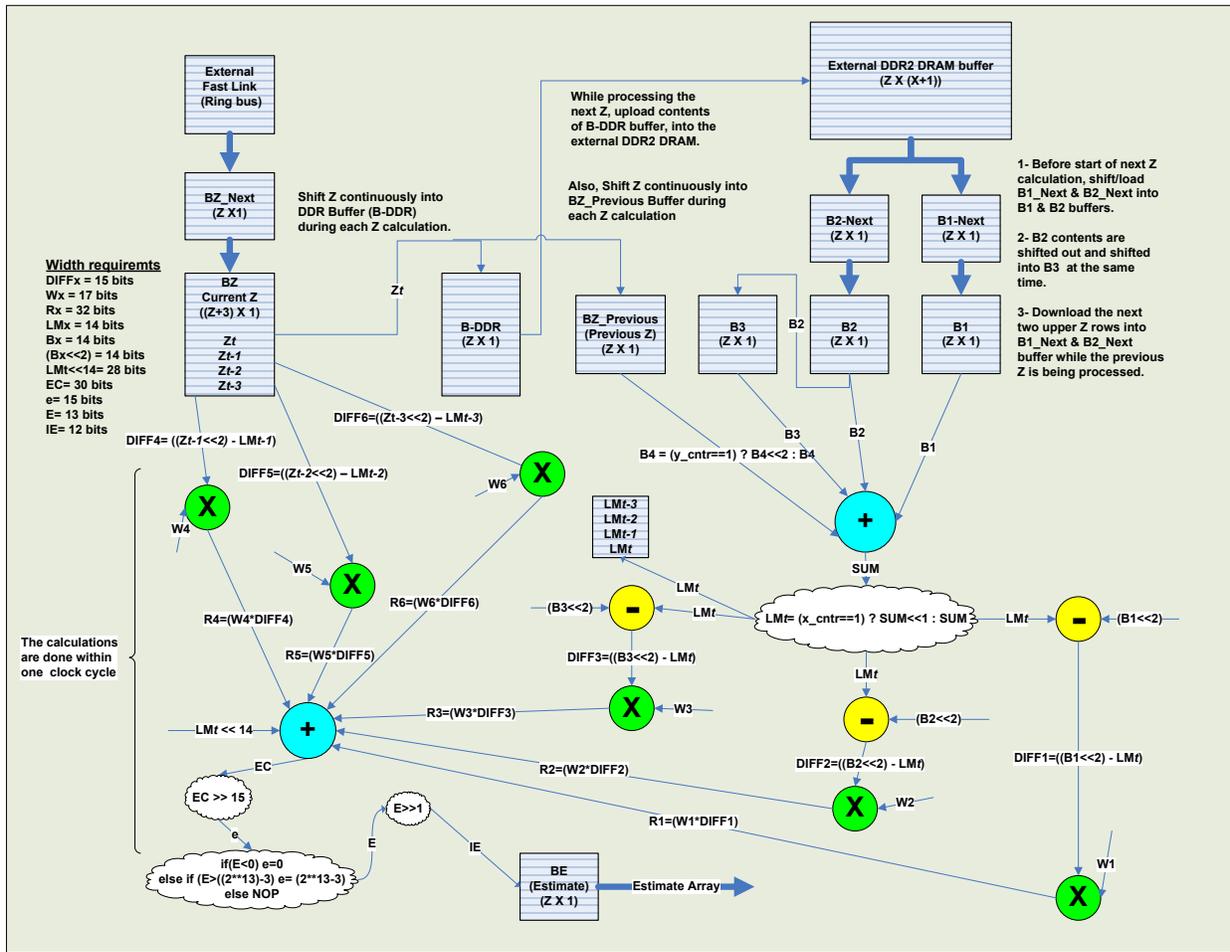
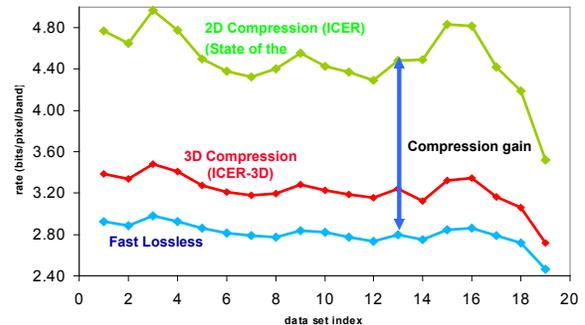


Figure 3: Block diagram of the digital implementation that illustrates the filter weights updates,  $w_{t,k+1} = w_{t,k} - \mu u_{t,k} \text{sgn}(e_t)$  as a function of  $u_{t,k}$ , referred to as *DIFF1*, the sign of the error between the estimate and the desired signal  $e_t$ , referred to as  $\Delta$  and the step size parameter  $\mu$ , referred to as *s*



**Figure 4.** Block diagram of the digital implementation for the computation of the estimate  $\hat{d}_t$  (referred to as  $EC$ ) using a six sample neighborhood with four samples from the same band as the sample to be predicted ( $B1, B2, B3, B4$ ) and one sample each from the three preceding bands ( $Zt-1, Zt-2, Zt-3$ ). The computation of the estimate uses a local mean subtraction method: for each sample an estimate is computed using fixed, causal, linear predictor involving only samples from the same band ( $LMt, LMt-1, LMt-2, LMt-3$ ). The estimate  $\hat{d}_t$  is computed by multiplying and adding respectively the weights ( $W1$  to  $W6$ ) by the inputs ( $DIFF1$  to  $DIFF6$ ) and is calculated within one clock cycle (33MHz for the Virtex IV LX160 device). The external Ring Bus provides the sensor data at a speed up to 800 Mbits/sec.

The Fast Lossless algorithm provides outstanding compression effectiveness. JPL's tests with uncalibrated AVIRIS data sets demonstrate compression results of about 40% lower bit rate than state-of-the-art 2D approaches (approximately 4:1 compression ratio) as shown in Figure 5. In addition to making use of correlations in all three dimensions, the algorithm also performs well compared to more complicated 3-D algorithms recently developed by other researchers, such as the ICER-3D [5][6][7].



**Figure 5:** Compression performance average over 19 uncalibrated AVIRIS hyperspectral test data sets. ICER and ICER-3D are state-of-the-art 2D image and 3D hyperspectral compressors developed at JPL.

### 3. FPGA IMPLEMENTATION

The Fast Lossless algorithm illustrated above was implemented and integrated into a reconfigurable system for a spacecraft payload requiring high communication throughput. The reconfigurable system takes advantage of high-density SRAM-based FPGAs to accommodate the on-board computer resulting in an efficient hardware architecture in terms of power, area, and speed.

#### *Background*

FPGA and ASIC hardware implementations for lossless hyperspectral data compression have been proposed by other researchers. At JPL, Scalable and Embedded FPGA implementation of the ICER-3D hyperspectral data compressor, a lossless and lossy wavelet based compressor, was developed. The implementation targets the Xilinx Virtex-II Pro architecture and it takes advantage of the FPGA embedded PowerPC core and the on-chip bus architecture. Such platforms allow efficient partitioning of the algorithm into software and hardware modules to take full advantage of the available hardware resources and provide a system on a chip (SoC) solution for the hyperspectral data compression problem. The implementation was prototyped on a Virtex II pro platform, and tested with a clock of 50Mhz resulting in a throughput of 8 Msample/sec .

Surrey Space Center developed a reconfigurable Intellectual Property (IP) cores using the Xilinx AccelDSP tool [10]. Their IP core implemented a design based on an extended Rice algorithm [11] proposed by the CCSDS with a combination of 2-D prediction and independency coding and utilizing a pre-scanning scheme. This approach achieved better compression performance than JPEG-LS. Their implementation was tested on a ZestSC2 FPGA prototyping board with a clock at 48 MHz and demonstrated a power consumption of 625mW.

The team lead by Bristol University proposed a universal algorithm and hardware architecture for context-based statistical lossless compression of multiple types of data using FPGA devices that support partial and dynamic reconfiguration [12][13]. Their proposed compression system uses a dynamically reconfigurable modeling stage followed by statically configured probability estimation and arithmetic coding stages. Dynamic modeling is specialized to each data type and uses a combination of context modeling, predictive coding and motion estimation depending on the data type being processed: 1-D general data, 2-D image data or 3-D multispectral images or video. The throughput performance, of the proposed system is 100Mbits/sec on a Xilinx Virtex-4 SX35 FPGA.

The team lead by GSFC developed an ASIC implementation of a new CCSDS 2D Image Compression Recommendation [14] [15]. The algorithm adopted in the recommendation consists of a two-dimensional discrete wavelet transform of the image, followed by progressive bit-plane coding of the transformed data. The algorithm provides lossless compression and is suitable for both frame-based image data and scan-based sensor data, and has

applications for near-Earth and deep-space missions. This hardware implementation separates the Discrete Wavelet Transform (DWT) and Bit-Plane-Encoder (BPE) into two ASICs. The chips are expected to process over 20 Msamples/sec at lower than 0.15 watts/Msamples/sec. The throughput rate is limited by currently available rad-hard RAM chip that would serve as the external RAM for the BPE processing [16]

Other current hardware developments of lossless image compression algorithms are based on several lossless compression hardware devices for universal data such as files (tapes, hard disk drives, file servers) and communication data (LAN, WAN, wireless) that are currently commercially available. Their performance has been compared for throughputs up to 1.6Gbit/s compression [17]. These ASIC compressors are the ALDC1-40S [18] (IBM) and the AHA3521 [19] that implements the adaptive lossless data compression (ALDC) (LZ1) algorithm, the AHA3211 [20] that implements the DCLZ (LZ2) algorithm and the Hi/fn 9600 [21] that implements the Lempel-Ziv Stac (LZS) (LZ1) algorithm. Due to the limitations of the above devices, we have decided to investigate a new approach using FPGA for our particular lossless hyperspectral data compression.

#### *FPGA implementation – Architecture and Data Flow*

The architecture of Fast Lossless compression algorithm is shown in Figure 6. The implementation works on 32 frames of hyperspectral data at a time. Raw imagery data is stored as three dimensional cube (for example  $X=640$ ,  $Z=480$ ,  $Y=32$ ). Each pixel of the hyperspectral cube can accommodate up to 14 bits depending on the resolution of the sensory data.

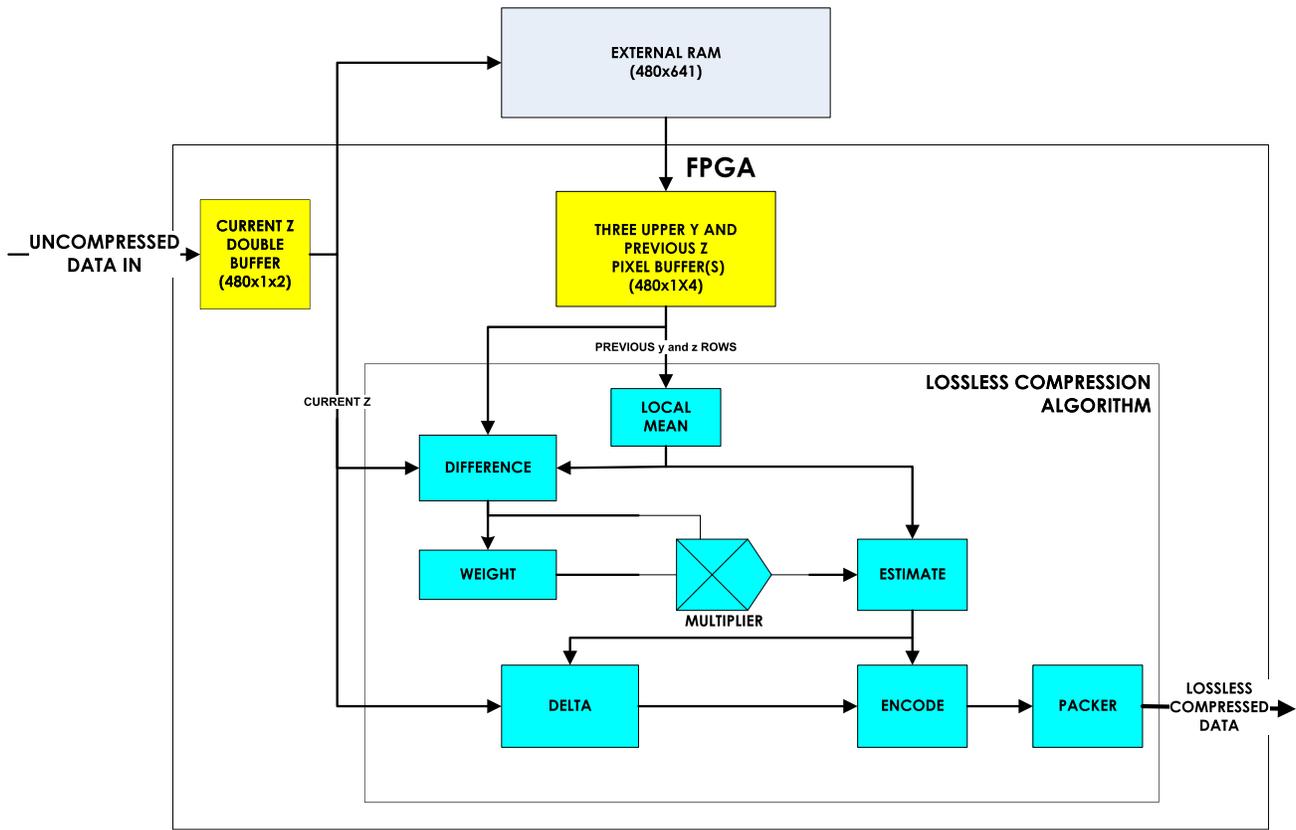
The current implementation targets the Xilinx Virtex IV, LX160 FPGA, and assumes a BIP (Byte interleaved by Pixel) format. The basic blocks of the implementation are:

LOCAL MEAN is an Accumulator and 4x16 bits Shift Register that is used to store the sum of three upper spatial pixels from the previous line and the previous spectral pixel from the previous band Shift Register is used to store and shift the last four (4) accumulator results.

DIFFERENCE block consists of six (6) identical subtract modules that are used to subtract the local mean values from the previous spatial and spectral pixels

WEIGHT block is made up of six (6), of length Z (number of spectral bands) by up to 14 bits FIFOs that is used to adjust the input to the multiplier. Values of weight are re-calculated for every new Z row. On power up all the weight values are initialized.

MULTIPLIER block includes six (6) Virtex IV/LX160, 18x18 multiplier primitives. Multipliers are used to multiply the output of Difference block with their adjusted Weight values.



**Figure 6** Compression algorithm block diagram with a critical path delay of 29.51ns. The critical path delay is measured between the uncompressed data and the lossless compressed data which includes the computation of the error  $e_t = \hat{d}_t - d_t$  and its encoding in the compressed bit stream using the Golomb-Rice codes.

ESTIMATE block consists of an Accumulator and a Comparator. The Accumulator is used to sum the multiplier values. The Estimate is adjusted and clipped depending on the accumulated result.

DELTA block subtracts the value of current pixel data from the Estimate to adjust the WEIGHT and also used as an input to the ENCODER.

ENCODER is made up of Comparator, of length Z (number of spectral bands) by up to 14 bits FIFO, Look Up Table and miscellaneous circuitry. Output of the Encoder is used to determine the width of the packed data.

PACKER includes Virtex IV distributed RAM and multiplexers. Distributed RAM is used for Look Up Table to adjust the compressed data into the final packed data word. Lossless compressed data is packed and output 32-bits at a time.

The basic concept of the data flow for the FPGA implementation (Figure 6) is as follows. Initially raw imagery data of each spectral (Z) row is input one row at a time into the “CURRENT Z DOUBLE BUFFER”. This buffer is configured as 2Z by up to 14 bits wide shift register. Two rows of length Z are stored in “CURRENT Z DOUBLE BUFFER”. New data is shifted in as pixel data is compressed.

Three upper spatial pixels (Y-1, Y and Y+1) and previous spectral pixel (Z) are needed to compress the current pixel data. For the first pixel (y=0) and first row (y=0), initial values are used for compression.

Current pixel data being compressed is also shifted out in parallel to the external RAM to produce an efficient pipeline. This data is inputted back to the “THREE UPPER Y AND PREVIOUS Z” internal FPGA pixel buffers for subsequent pixel processing.

Compression of one pixel of the data happens once every FPGA clock cycle. Compressed data is input to the PACKER module which packs the compressed data into 32 bits data words. Each 32 bits data word may contain several pixels of data. Also, a compressed pixel datum may fall into two 32-bit data words boundaries, which in turn will be decoded by the decompression algorithm accordingly. The implementation assumes an external fast link to bring the raw data to the FPGA such as a ring bus (up to 800 Mbits/sec).

#### 4. PERFORMANCE AND BENCHMARKS

Our FPGA implementation was benchmarked on the Xilinx Virtex IV LX160 device and ported to a Xilinx prototype board (Figure 7). The current implementation has a critical path of 28.5 nsec which dictated a clock speed of

33MHz. It compresses one sample every clock cycle, which results in a speed of 33MSample/sec or 33 times faster than the software implementation running on a Pentium IV machine.

The implementation has a rather low device utilization of the Xilinx Virtex IV LX160 as shown in the table 1 making the total power consumption of the implementation about 1.27 watts out of which 0.576 watts is consumed by 384mA of internal quiescent current at 1.5V internal voltage on the LX160.



**Figure 7: FPGA Development Board**

**Table 1: LX160 Device Utilization**

	Available[#]	Used[#]	Used[%]
BUFGs	32	2	6%
DSP48s	96	6	6%
FIFO16s	288	1	1%
External IOBs	768	79	10%
OLOGICs	960	21	2%
RAMB16s	288	8	2%
Slices	67584	3577	5%

Our FPGA implementation is easily portable to other FPGA platforms and to an ASIC implementation. It can also be scaled for faster processing.

## 5. SUMMARY

We presented in this paper an FPGA implementation of a novel hyperspectral data compression algorithm, the JPL adaptive Fast Lossless compressor. The implementation targets the Xilinx Virtex IV FPGAs and provides an acceleration of at least 33 times the software implementation, making the use of this compressor practical for satellites and planet orbiting missions with hyperspectral instruments. Future development will provide multiple implementations and options to deploy various versions of the algorithm to accommodate data from different instrument types.

## ACKNOWLEDGEMENTS

The work described in this publication was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. This work was funded by Air Force Research Laboratory through the grant, entitled “Fast Lossless On-Board Hyperspectral Data Compression”. Special thanks to Randy Odle, program manager, who also supported this research at JPL, and to Ian Ferguson for his technical support to this research

## REFERENCES

- [1] A. Gersho. “Adaptive filtering with binary reinforcement”. *IEEE Transactions on Information Theory*, IT-30(2):191–199, March 1984.
- [2] B. Widrow, J. R. Glover, J. M. McCool, J. Kaunitz, C. S. Williams, R. C. Goodlin, J. R. Zeidler, R. H. Hearn, and E. Dong. “Adaptive Noise Cancelling: Principles and Applications”. *The Proceedings of the IEEE*, 63(12):1692–1716, December 1975.
- [3] M. Klimesh, “Low-Complexity Lossless Compression of Hyperspectral Imagery via Adaptive Filtering,” *The Interplanetary Network Progress Report*, vol. 42-163, Jet Propulsion Laboratory, Pasadena, California, pp.1–10, November 15, 2005.
- [4] N. Aranki, C. Villalpando, J. Namkung, A. Kiely, M. Klimesh, H. Xie, “Hyperspectral Data Compression on Reconfigurable Platforms”, NASA New Technology Report NPO-42834
- [5] A. Kiely, N. Aranki, M. Klimesh, H. Xie, “Hyperspectral Data Compression based on the Three Dimensional Wavelet Transform,” NASA New Technology Report NPO-42835
- [6] A. Kiely, M. Klimesh, H. Xie, N. Aranki, “Context Modeler for Compression of Wavelet-Transformed Hyperspectral Data,” NASA New Technology Report NPO-43239
- [7] A. Kiely, M. Klimesh, H. Xie, N. Aranki, “ICER-3D Hyperspectral Data Compression Software,” NASA New Technology Report NPO-43238.
- [8] F. Rizzo, B. Carpentieri, G. Motta, and J. A. Storer. “Low-complexity lossless compression of hyperspectral imagery via linear prediction”. *IEEE Signal Processing Letters*, 12(2):138–141, February 2005.
- [9] B. Aiuzzi, L. Alparone, and S. Baronti. “Near-lossless compression of 3-D optical data”. *IEEE Transactions on Geoscience and Remote Sensing*, 39(11):2547–2557, November 2001.
- [10] Guoxia Yu, Tanya Vladimirova, Xiaofeng Wu, Martin N. Sweeting, “A New High-Level Reconfigurable Lossless Image Compression System for Space Applications”, In NASA/ESA Conference on Adaptive Hardware and Systems, 2008. AHS '08. 22-25 June 2008, pp 183-190. IEEE Computer Society
- [11] CCSDS, Lossless Data Compression, Recommendation for space data system standards vol. 121.0-B-1: CCSDS, 1997.
- [12] Nunez-Yanez, J.L.; Xiaolin Chen; Canagarajah, N.; Vitulli, R., “Statistical Lossless Compression of Space Imagery and General Data in a Reconfigurable Architecture” In NASA/ESA Conference on Adaptive Hardware and Systems, 2008. AHS '08. 22-25 June 2008, pp 172-177. IEEE Computer Society.
- [13] Nunez-Yanez, J.L.; Chouliaras, V.A., “A configurable statistical lossless compression core based on variable order Markov modeling and arithmetic coding”, In IEEE

Transactions on Computers. Volume 54, Issue 11, Nov. 2005 Page(s):1345 – 1359.

- [14] Pen-Shu Yeh; Armbruster, P.; Kiely, A.; Masschelein, B.; Moury, G.; Schaefer, C.; Thiebaut, C., “The New CCSDS Image Compression Recommendation”, In IEEE Aerospace Conference 2005, 5-12 March 2005 Page(s):4138 – 4145. IEEE
- [15] Donohoe, G.W.; Buehler, D.M.; Hass, K.J.; Walker, W.; Pen-Shu Yeh, “Field Programmable Processor Array: Reconfigurable Computing for Space”, IEEE Aerospace Conference 2007, 3-10 March 2007 Page(s):1 – 6. IEEE
- [16] PS Yeh, J Venbrux, “A High Performance Image Data Compression Technique for Space Applications”, In NASA Earth Science Technology Conference, 2003.
- [17] Nunez, J.L. Jones, S. “Gbit/s lossless data compression hardware”, In IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Volume: 11, Issue: 3, On page(s): 499- 510, June 2003.
- [18] ALDC1-40S-M, IBM Corporation, IBM Microelectronics Division, New York, 1994.
- [19] AHA3521 “40 Mbytes/s ALDC Data Compression Coprocessor IC”, Advanced Hardware Architectures Inc, Pullman, WA, 1997.
- [20] AHA3211 “40 Mbytes/s DCLZ Data Compression Coprocessor IC”, Advanced Hardware Architectures Inc., Pullman, WA, 1997.
- [21] 9600 Data Compression Processor, Hi/fn Inc, Los Gatos, CA, 1999.
- [22] William Campbell, Nicholas M. Short, “Remote Sensing Tutorial”, 2004  
[http://www.fas.org/irp/imint/docs/rst/Sect13/Sect13\\_9.html](http://www.fas.org/irp/imint/docs/rst/Sect13/Sect13_9.html)
- [23] R.G. Gallager and D.C. Van Voorhis. “Optimal source codes for geometrically distributed integer alphabets”. IEEE Transactions on Information Theory, IT-21 (2): 228-230, March 1975.
- [24] “Image Data Compression”, Recommendation for Space Data Systems Standard, The Consultative Committee for Space Data Systems (CCSDS), Blue Book, Issue 1. Nov. 2005 (CCSDS 122.0-B-1).

## BIOGRAPHY



*Nazeeh Aranki received the BSEE, MSEE and Ph.D. in Electrical Engineering from the Caltech and USC. Nazeeh has 25 years of experience in design and implementation of digital and FPGA based systems. Since he joined JPL in 1994, his research interests have included reconfigurable hardware, digital signal and image processing, data compression, parallel processing, evolvable hardware, and neural networks. Nazeeh developed algorithms for compression of hyperspectral data and their implementations on reconfigurable platforms. He was awarded a patent and the NASA Space Act award for his contribution in the development of an FPGA-based neuroprocessor for automotive applications in control and diagnostics. He also served as the principal investigator and task Manager on a number of NASA, DARPA and*

*AFRL projects related to data compression and power aware computing and communications. Nazeeh is currently a senior member of the engineering and research team of the Avionics Equipment Section at JPL,*



*Alireza Bakhshi is technical expert in FPGA/ASIC and high speed Digital circuit design for real-time DSP application on FPGA. In addition Alireza has twenty years of experience in embedded applications, Digital circuits and Systems design in Space Radiation Hardened, Airport systems and Commercial avionics environments. He has contributed to multiple NASA missions as lead of the digital electronics such as the Microwave Limb Sounder (MLS), Mars Exploration Rover (MER), Ocean Surface Topography Mission (OSTM), Mars Science Laboratory (MSL) flight projects and technology projects such as Ultra Long Life (ULL) and, Operation of FPGAs in extreme low temperatures. He is president and CEO of B&A Engineering Inc.*



*Didier Keymeulen received the BSEE, MSEE and Ph.D. in Electrical Engineering and Computer Science from the Free University of Brussels, Belgium in 1994. In 1996 he joined the computer science division of the Japanese National Electrotechnical Laboratory as senior researcher. Currently he is principal member of the technical staff of JPL in the Bio-Inspired Technologies Group. At JPL, he is responsible for DoD and NASA applications on evolvable hardware for adaptive computing that leads to the development of fault-tolerant electronics and autonomous and adaptive sensor technology. He participated also as test electronics lead, to Tunable Laser Spectrum instrument on Mars Science Laboratory. He served as the chair, co-chair, and program-chair of the NASA/ESA Conference on Adaptive Hardware. Didier is a member of the IEEE.*

*Matthew Klimesh received B.S.E., M.S.E., and Ph.D. degrees, all in electrical engineering from the University of Michigan in Ann Arbor, in 1989, 1990, and 1995, respectively. He spent one year as a research fellow (postdoc) at Michigan. Since 1996 he has been with the Information Processing Group at Caltech's Jet Propulsion Laboratory, working primarily on research and development of data compression algorithms for space applications. His research interests include source coding, data compression, network coding, rate-distortion theory, channel coding, probability, and discrete mathematics.*