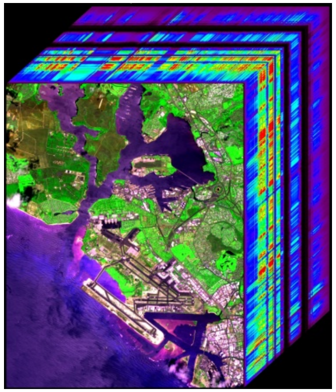
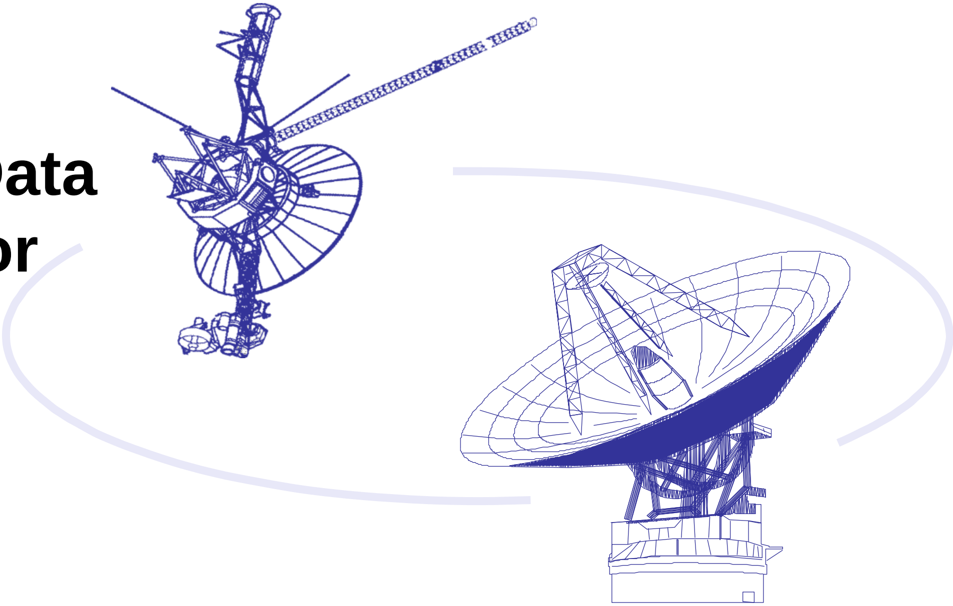
Real-time Airborne Demonstration of Fast Lossless Hyperspectral Data Compression System for AVIRIS-NG and PRISM





Didier Keymeulen, Huy Luong, Nazeeh Aranki, Charles Sarture, Michael Eastwood, Ian Mccubbin, Alan Mazer, Matt Klimesh, Robert Green, David Dolman (3), Alireza Bakhshi (2)

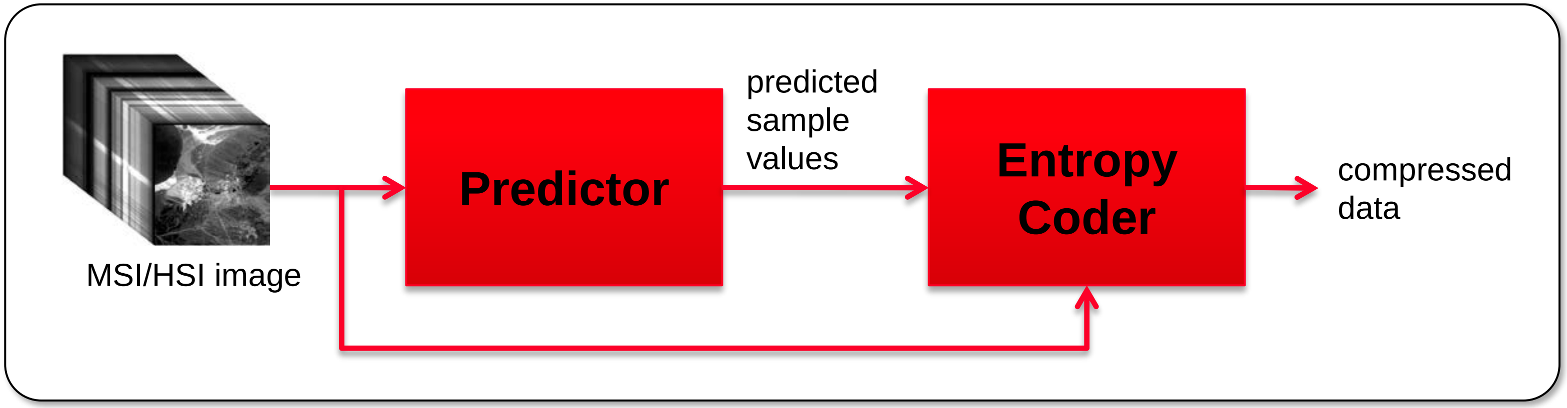
Jet Propulsion Laboratory
California Institute of Technology
(2) B&A Engineering Inc.
(3) Alpha Data Inc.

Outline

- Overview of Fast Lossless (FL) Hyperspectral Data Compression Algorithm
- Fast Lossless FPGA Implementation
- Airborne Demonstrations

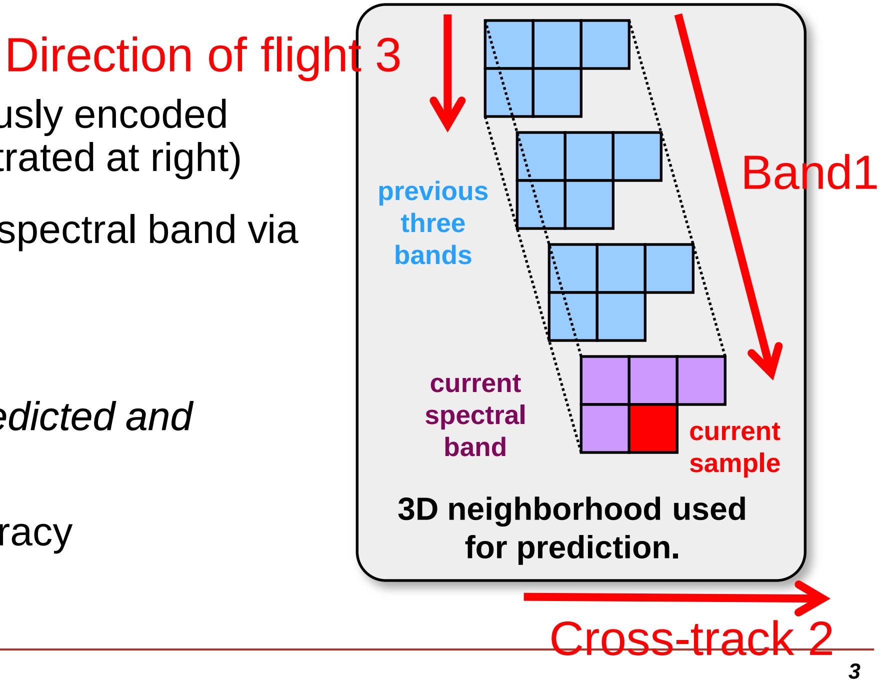
Fast Lossless (FL) MSI/HSI Compressor

FL Compressor Overview



Approach: Predictive compression, encoding samples one-at-a-time

- **Predictor**
  - Computes predicted sample value from previously encoded nearby samples (prediction neighborhood illustrated at right)
  - Adaptively adjusts prediction weights for each spectral band via adaptive linear prediction
- **Entropy Coder**
  - Losslessly encodes the *difference between predicted and actual sample values*
  - Adaptively adjusts to changing prediction accuracy



3D neighborhood used for prediction.

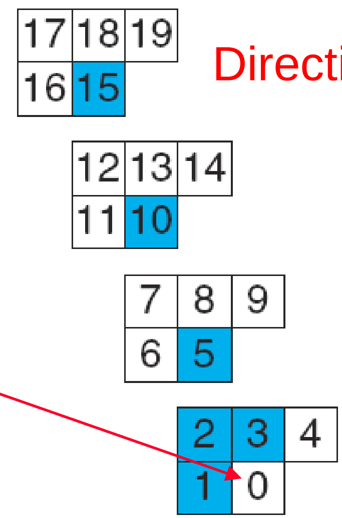
Compression Algorithm: Estimation

- **Purpose:** Estimate a desired signal d_t from an input vector $\vec{u}_t$ using a linear estimator that is adaptively updated from previous results
- **Compression of Estimate Error :**
  - Form estimate: $\hat{d}_t = \vec{w}_t^T \vec{u}_t$
  - Calculate estimation error: $e_t = \hat{d}_t - d_t$

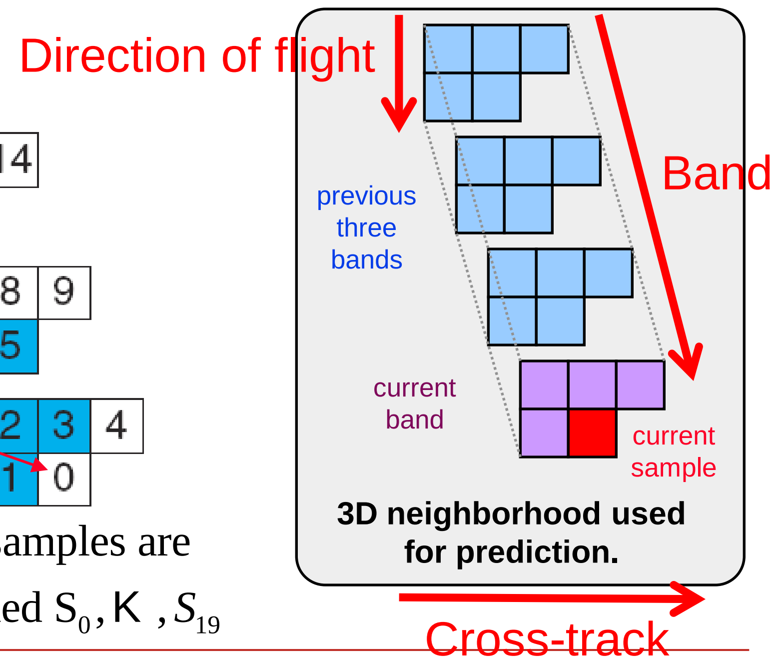
    e_t is encoded in the compressed bitstream
  - Update filter weights using the sign algorithm: $\vec{w}_{t+1} = \vec{w}_t - \mu \vec{u}_t \operatorname{sgn}(e_t)$

    where μ is the "adaptation step size" parameter
- **Naive approach:** use local neighborhood to construct $\vec{u}_t$ around $d_t = s_0$

with $d_t = s_0$ and $\vec{u}_t = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_5 \\ s_{10} \\ s_{15} \end{bmatrix}$



The samples are labelled $S_0, \ldots, S_{19}$



But performs poorly ….

Compression Algorithm: Local Mean Subtraction

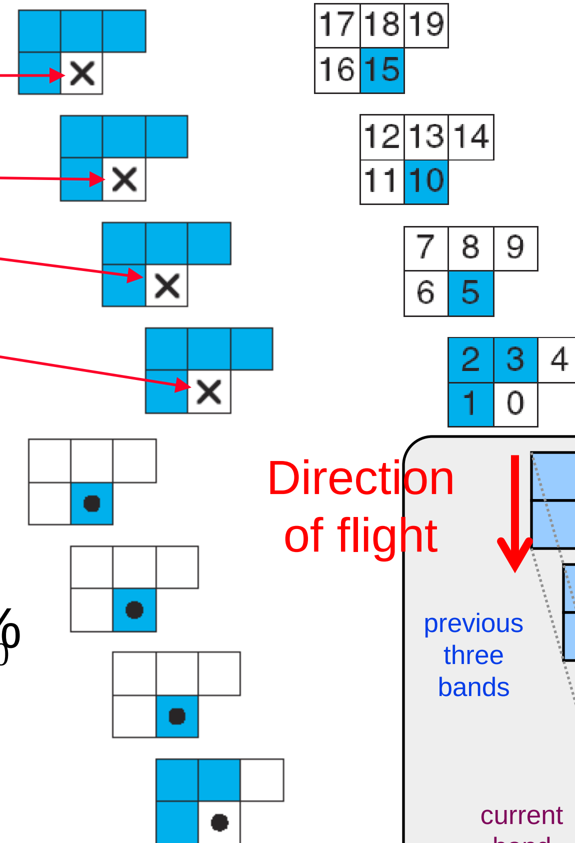
- **Our solution**: compute simple preliminary estimates $\tilde{s}_i$ in each band at the spatial location of the sample being predicted, and subtract from the input samples.

$$\tilde{s}_{15} = (s_{16} + s_{17} + s_{18} + s_{19})/4$$

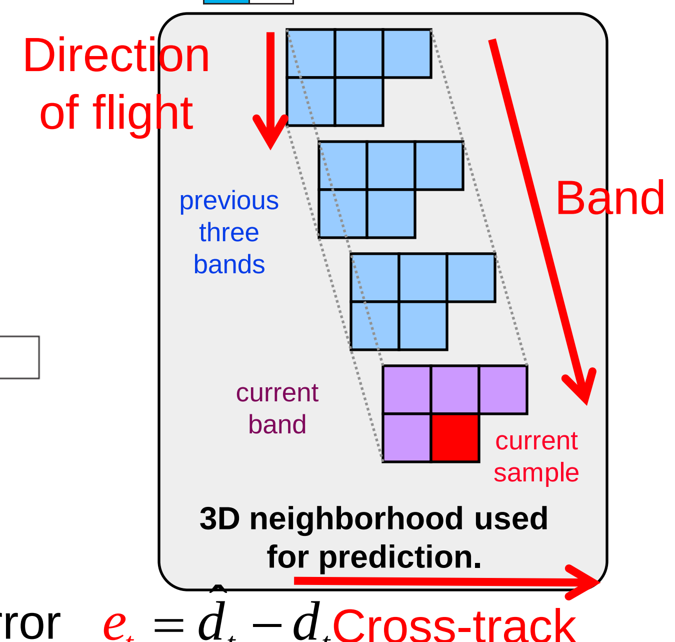
$$\tilde{s}_{10} = (s_{11} + s_{12} + s_{13} + s_{14})/4$$

$$\tilde{s}_{5} = (s_{6} + s_{7} + s_{8} + s_{9})/4$$

$$\tilde{s}_{0} = (s_{1} + s_{2} + s_{3} + s_{4})/4$$



Use $\vec{u}_t = \begin{bmatrix} s_1 - \tilde{s}_0 \\ s_2 - \tilde{s}_0 \\ s_3 - \tilde{s}_0 \\ s_5 - \tilde{s}_5 \\ s_{10} - \tilde{s}_{10} \\ s_{15} - \tilde{s}_{15} \end{bmatrix}$ and $d_t = s_0 - \tilde{s}_0$

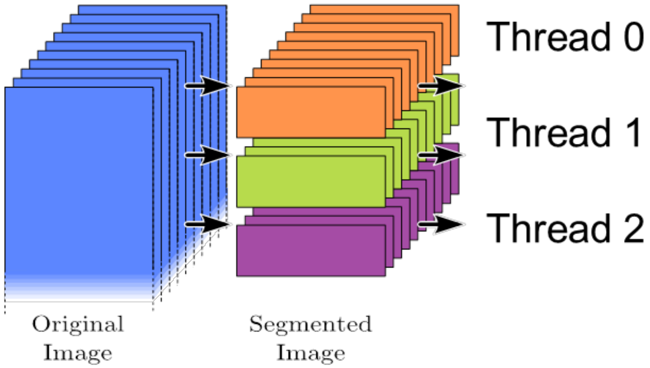


3D neighborhood used for prediction.

to compute the estimate $\hat{d}_t = \vec{w}_t^T \vec{u}_t$ and the estimate error $e_t = \hat{d}_t - d_t$

Compression Algorithm: Implementation

- **Sign algorithm** is used for weight adaptation
- **Estimation error** is encoded using Golomb power-of-2 codes
- **Dataset is divided into parts (32 lines each)**, which are compressed independently. This provides some error containment.
- **Each spectral band has its own prediction weights**, maintained independently of the prediction weights for other spectral bands



Compression Algorithm: Other Methods

Compare our **"Fast lossless"** compression algorithm with:

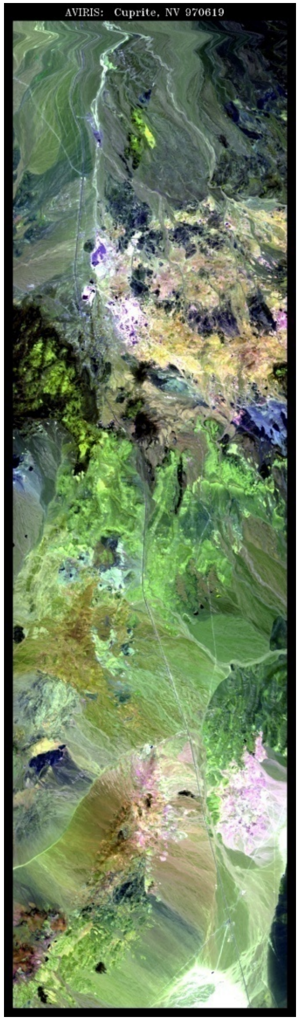
- ***ICER-3D***: *a 3-D-wavelet-based compressor which is the state-of-the-art (ICER-2D is used on both spirit and opportunity MER rovers)*
- ***Rice/USES (GSFC)****: algorithm used in USES chip, with the multispectral predictor option.*
- ***JPEG-LS****: is most efficient for 2D and is applied to the spectral bands independently*

Other Methods:

- **Differential JPEG-LS**: JPEG-LS applied to the differences between the successive spectral bands
- **SLSQ and SLSQ-OPT**: two versions of Spectral-oriented Least Squares (SLSQ) [Rizzo et al., 2005]. Algorithms with complexity roughly similar to that of ours.
- **3-D CALIC**: a nontrivial extension of the basic (2-D) CALIC algorithm to multispectral imagery. More complex.
- **M-CALIC**: multiband CALIC, another extension of CALIC to multispectral imagery. More complex.
- **ASAP**: Adaptive Selection of Adaptive Predictors [Aiazzi et al., 2001]; more computationally intensive than any of the other compressors in the tables

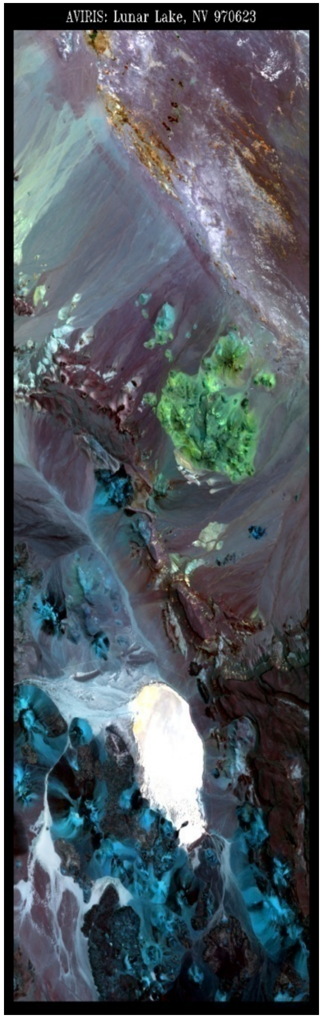
Comparison using Aviris Data Sets Test Bed

Moffett Field (vegetation, urban, water)

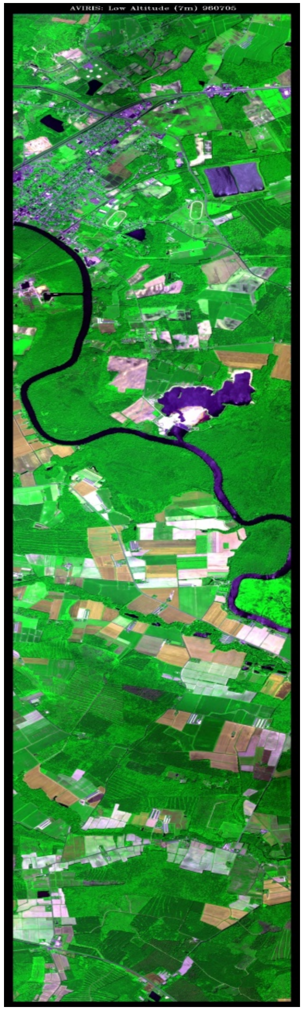


Cuprite (geological features)

Jasper Ridge (vegetation)



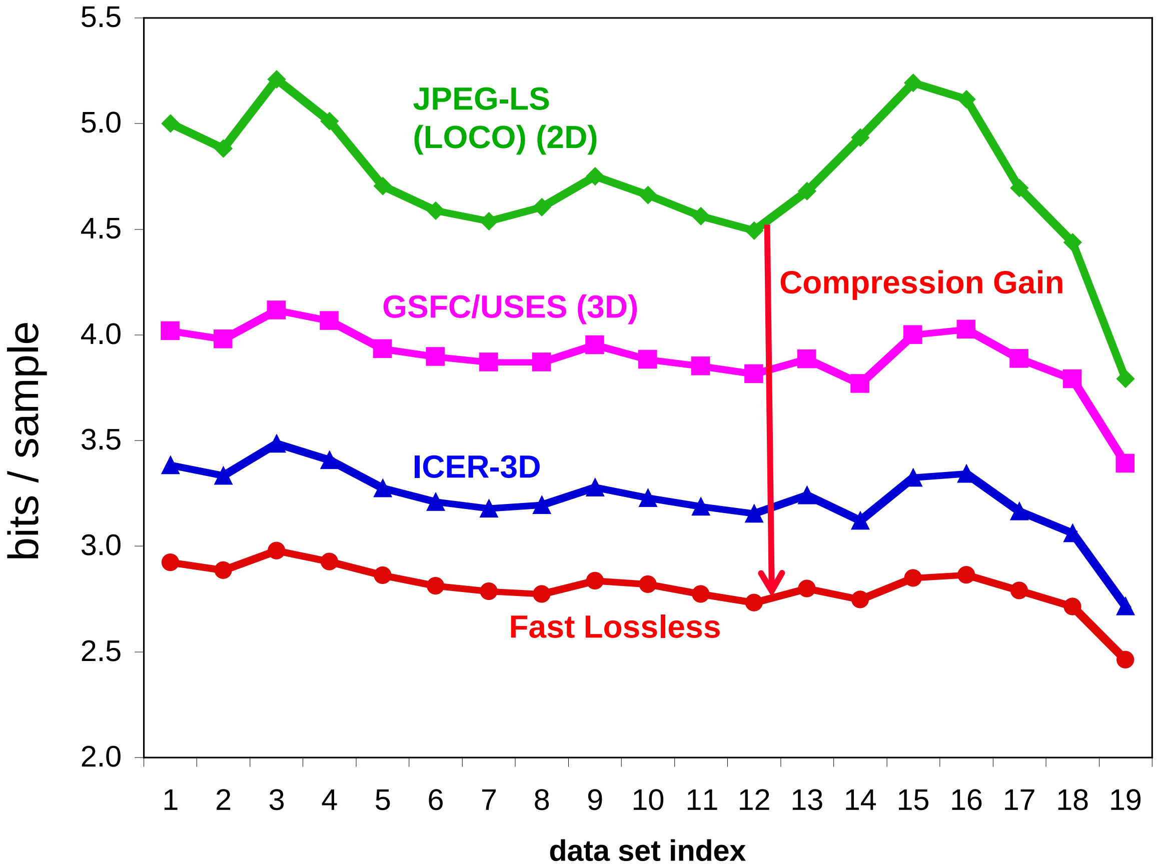
Lunar Lake (calibration)



Low Altitude (high spatial resolution)

AVIRIS data sets represent different scenes

Comparison for raw AVIRIS Data



Compressor	rate (bits/sample)
JPEG-LS (2D)	4.73
GSFC/USES Multispectral	3.89
ICER-3D	3.23
Fast Lossless	**2.81**

Compression performance averaged over 19 uncalibrated AVIRIS hyperspectral test data sets.

About 40% lower bit rate than state-of-the-art 2D approach (GSFC/USES).

Tests using 19 uncalibrated AVIRIS data sets:

• original sample size: 12 bits/sample

• data size: (614 × 512) pixels × 224 bands

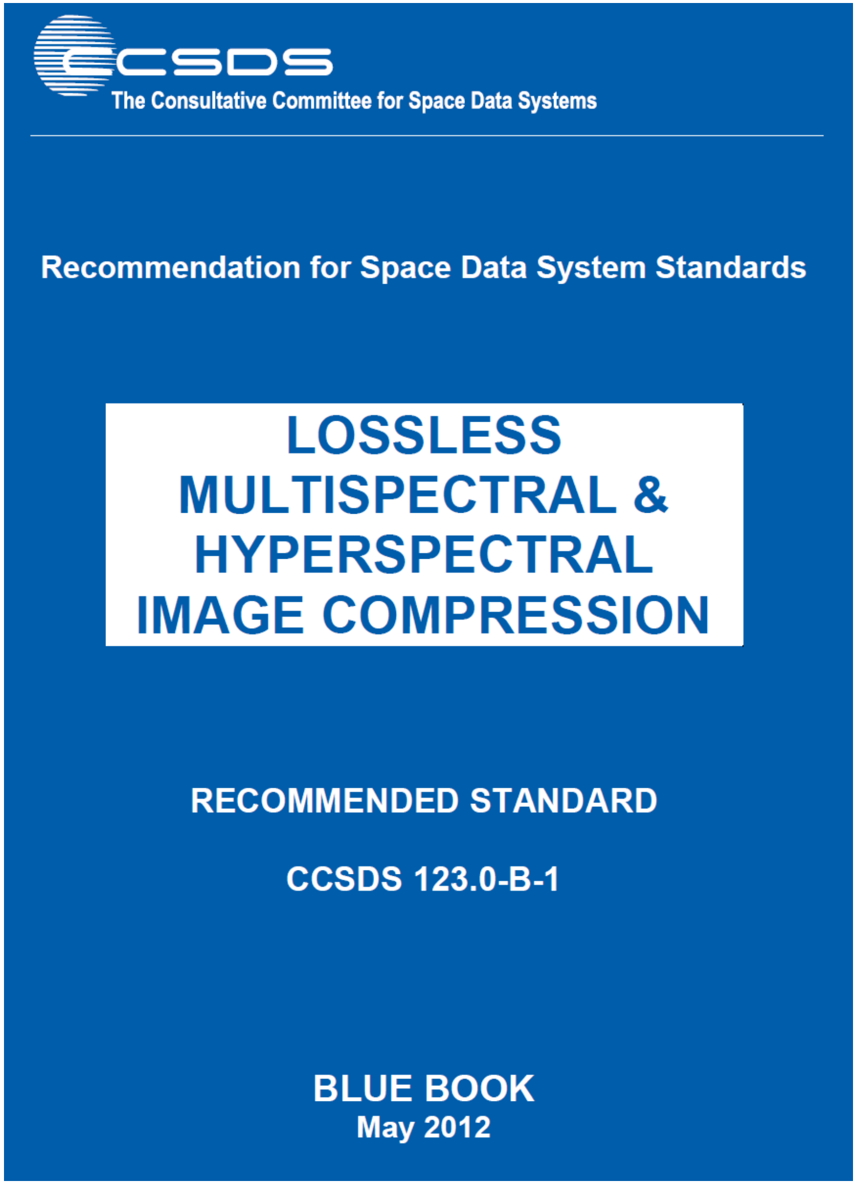
Methods:

JPEG-LS: is most efficient for 2D**; GSFC/USES** use chip; **ICER-3D** SOA (ICER-2D MER rovers)

Compression Algorithm Features

- **Performance:** outstanding compression effectiveness
- **Robust**; requires no training data or other specific information about the nature of the spectral bands for a fixed instrument dynamic range
- **Simple**: well-suited for implementation on FPGA hardware and easily parallelizable
- **Low computational complexity**. required operations per sample are:
  - 6 integer multiplications
  - 25 integer addition, subtraction, or bit shift operations
  - Golomb coding operations
- **Modest memory requirement**: enough to hold one spatial-spectral slice of the data (e.g., ≤650 Kbytes for AVIRISng data with 481 bands and 640 samples/line)
- **Instrument**: well-suited to push broom instruments

JPL Lossless Data Compression is a CCSDS Standard


The Consultative Committee for Space Data Systems

Recommendation for Space Data System Standards

LOSSLESS MULTISPECTRAL & HYPERSPECTRAL IMAGE COMPRESSION

RECOMMENDED STANDARD

CCSDS 123.0-B-1

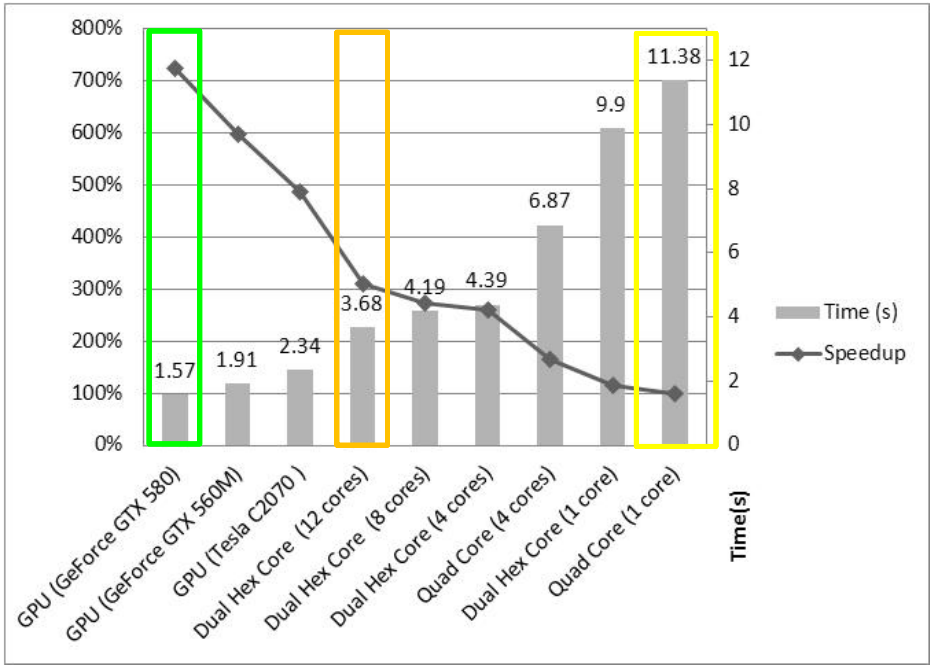
BLUE BOOK
May 2012

The Consultative Committee for Space Data Systems (CCSDS) Multispectral & Hyperspectral Data Compression working group has adopted the FL compressor as international standard CCSDS-123.0-B-1

FL verification software has demonstrated outstanding performance on all of the myriad airborne and spaceborne imagers represented in the CCSDS test data set:

- Hyperspectral imagers:
  AVIRIS, Hyperion, SFSI, CASI, M3, CRISM
- Ultraspectral sounders:
  AIRS, IASI
- Multispectral imagers:
  MODIS, MSG, PLEIADES, VEGETATION, SPOT5

High Speed FL Implementations: CPU/GPU



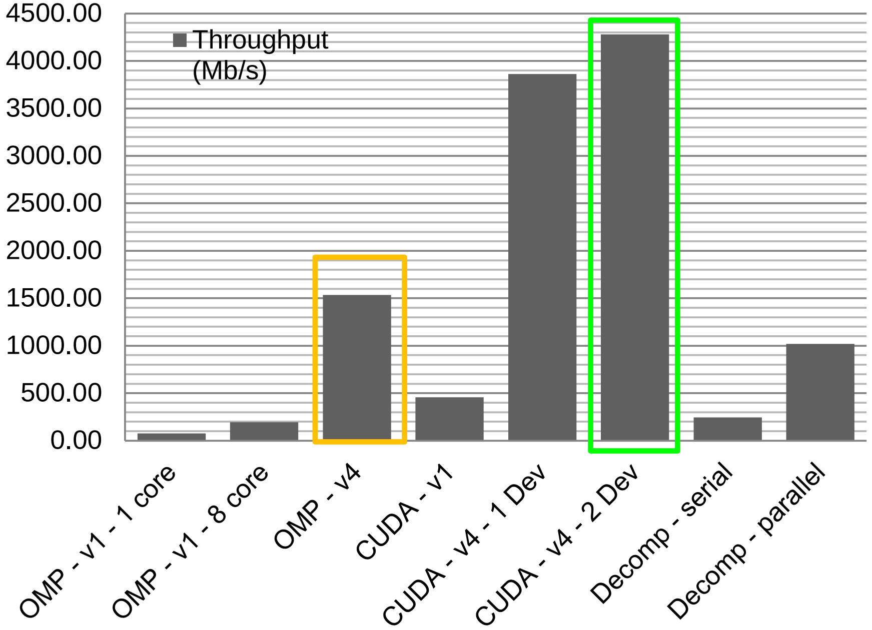
	Speedup	Time (s)	Speed (Mbit/s)	Speed (MSamp/s)
GPU GeForce GTX 580	725%	1.57	583.08	44.85
GPU GeForce GTX 560M	596%	1.91	479.29	36.87
GPU Tesla C2070	486%	2.34	391.21	30.09
Dual Hex Core (12 cores)	309%	3.68	248.76	19.14
Dual Hex Core (8 cores)	272%	4.19	218.48	16.81
Dual Hex Core (4 cores)	259%	4.39	208.53	16.04
Quad Core (4 cores)	196%	6.87	133.25	10.25
Dual Hex Core (1 core)	115%	9.9	92.47	7.11
Quad Core (1 core)	100%	11.38	80.44	6.19

Data Rate:
AVRISng (481*640 pixels per frames @100 frames/sec): 500Mbit/s
Future (481*1600 pixels per frames @100 frames/sec): 1300 Mbit/s
FPGA FL: 640 Mbit/s

- FL is well-suited for high-speed parallel implementations:
  - **GPU: 7× speed-up** – A GPU hardware implementation targeting the current state-of-the-art GPUs from NVIDIA®: mobile version GTX560M and desktop version GTX580
  - **OpenMP: 3× speed-up** – A 12-core implementation targeting the mobile Intel® quad-core i7™ processor and the desktop Intel® hexa-core Xeon™ processor
- Example: uncalibrated AVIRIS hyperspectral image (137MBytes)
  - Compression time: 11.38 sec on single-core CPU, 3.68 sec on 12-core CPU, and 1.57 sec on GPU

High Speed FL Implementations: CPU/GPU

Version 2: Even faster with re-designed data path



Version	Time (ms)	Throughput (Mb/s)	Throughput (MSamp/s)	Speedup vs. V1
OMP - v1 - 8 core	4488	194.53	14.96	1.00
OMP - v4 – 12 core	**569**	**1534.68**	**118.05**	**7.89**
CUDA - v1	1910	457.08	35.16	1.00
CUDA - v4 - 1 GPU	**226**	**3862.97**	**297.15**	**8.45**
CUDA - v4 - 2 GPU	**204**	**4279.56**	**329.20**	**9.36**
Decompress (serial)	3585	243.53	18.73	1.00
Decompress (parallel)	**857**	**1018.16**	**78.32**	**4.18**

Data Rate:
AVRISng (481*640 pixels per frames @100 frames/sec): 500Mbit/s
Future (481*1600 pixels per frames @100 frames/sec): 1300 Mbit/s
FPGA FL: 640 Mbit/s

- Redesigned data path implementation: Parallel computation across multiple 32 frames of the full image
- Total speed-up for Version 2
  - **GPU: 56× speed-up**– 137MB AVIRIS image compression time: 204 ms (vs. 11.38 sec)
  - **12-core CPU: 20× speed-up**– 137MB AVIRIS image compression time: 569 ms (vs. 11.38 sec)
- True real-time performance (2×-5× real-time target of 800Mb/s or 50MSamples/sec) BUT require 100 Watt

FL FPGA: **ARTEMIS & AVIRIS-NG**

FL FPGA Compression IPs for whiskbroom and pushbroom imagers

- **Xilinx Virtex-4 Lab Demonstration for ARTEMIS**
  - Implemented on Xilinx Virtex4 ML401 prototype board.
  - 17 MB image data (32 frames) uploaded serially to 256 DDR SDRAM prior to compression

- **Xilinx Virtex-5 Real-Time Airborne Onboard Compression**
  - Implemented pushbroom compressor on COTS Virtex 5 (equivalent to V5 Rad-hard device). Compresses one sample every clock cycle, a speed of 40 MSample/sec
  - Implementation tested in National Instruments PXI environment which includes a PXIe-7962R board with Xilinx Virtex-5 SX50T and two 256MBytes DRAMs. The system is connected to the airborne AVIRIS-NG HSI instrument and provides real-time onboard compression

ML401 Board

NI PXIe-7962R

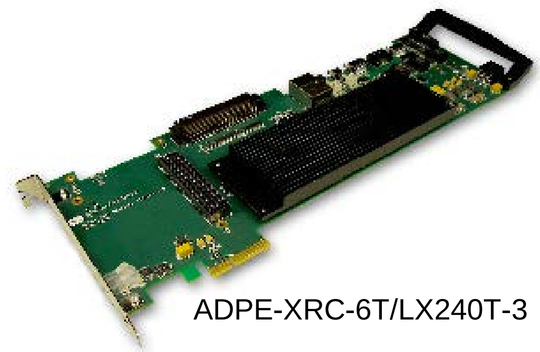
Twin Otter hosting AVIRIS-NG

14

FL FPGA: PRISM & AVIRISng

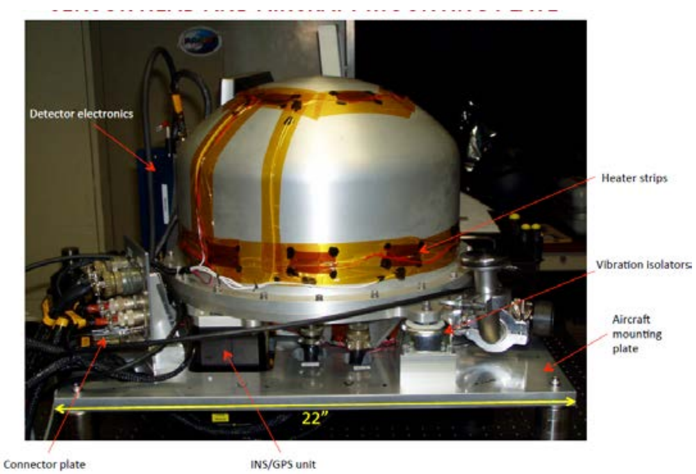
Real-time aircraft onboard compression

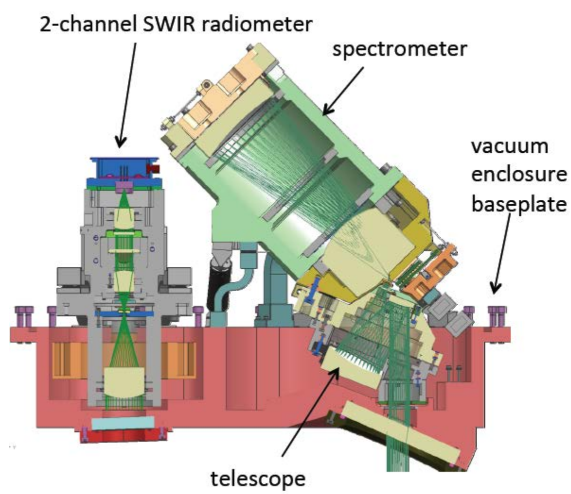
- Implemented pushbroom FL compressor on a COTS Virtex 6. Compresses one sample every clock cycle, a speed of 40 MSample/sec.
- Implementation tested via Alpha-Data ADPE-XRC-6T which includes
  - Xilinx Virtex-6 LX240T
  - two 256MBytes DRAMs (32bits data word, 3.2GBytes/sec per bank)
  - PCIe x4 Gen2 (500MBytes/sec per lane).
- PRISM and AVIRISng HSI image data transferred in real-time (60MBytes/sec) to the Virtex-6 via Alpha-Data FMC-CLINK-MINI camera link board, compressed on the Virtex-6 and transferred through PCIe to a 1GBytes SSD drive configured as RAID0 (500MBytes/sec)



ADPE-XRC-6T/LX240T-3

FMC-CLINK-MINI

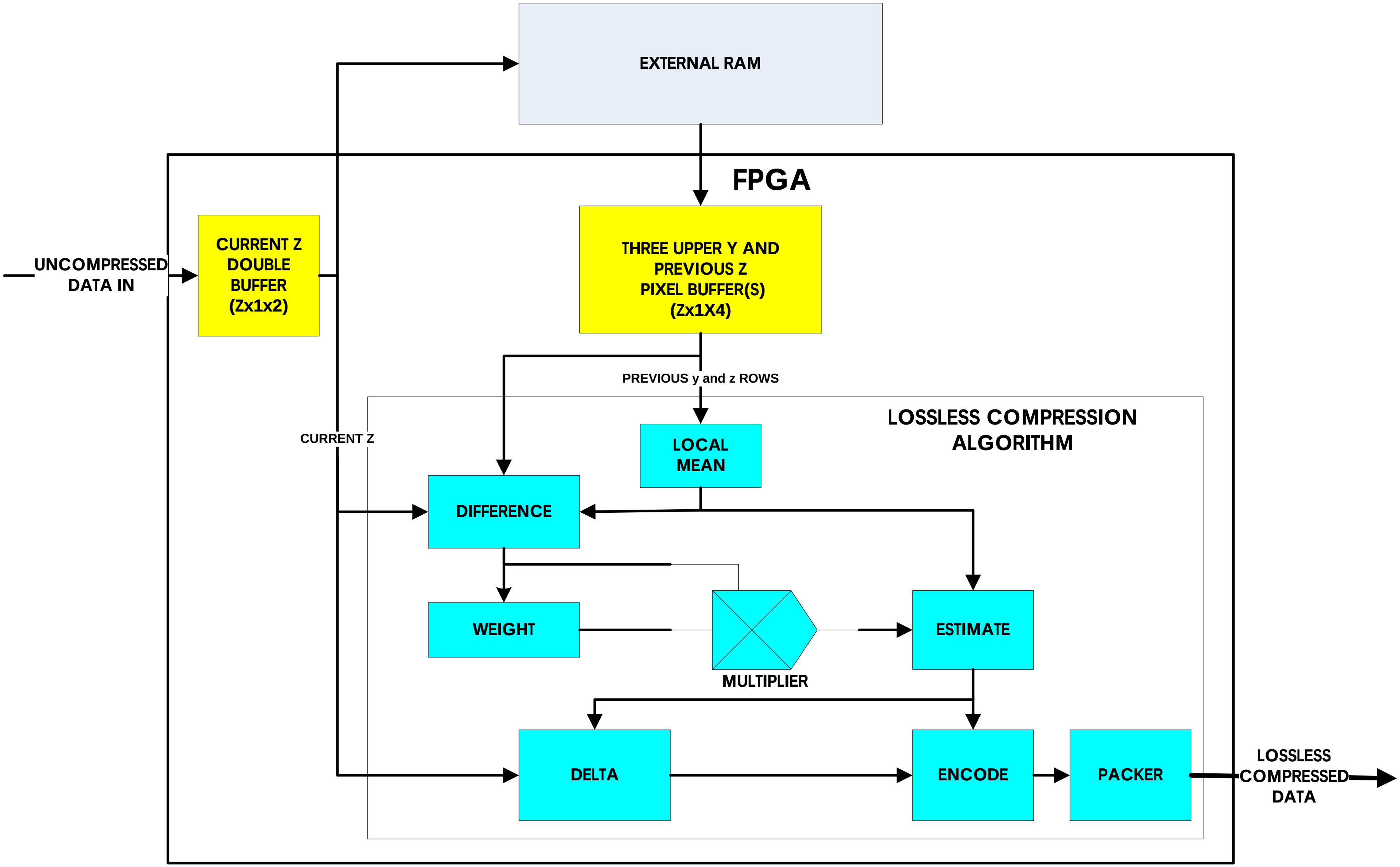




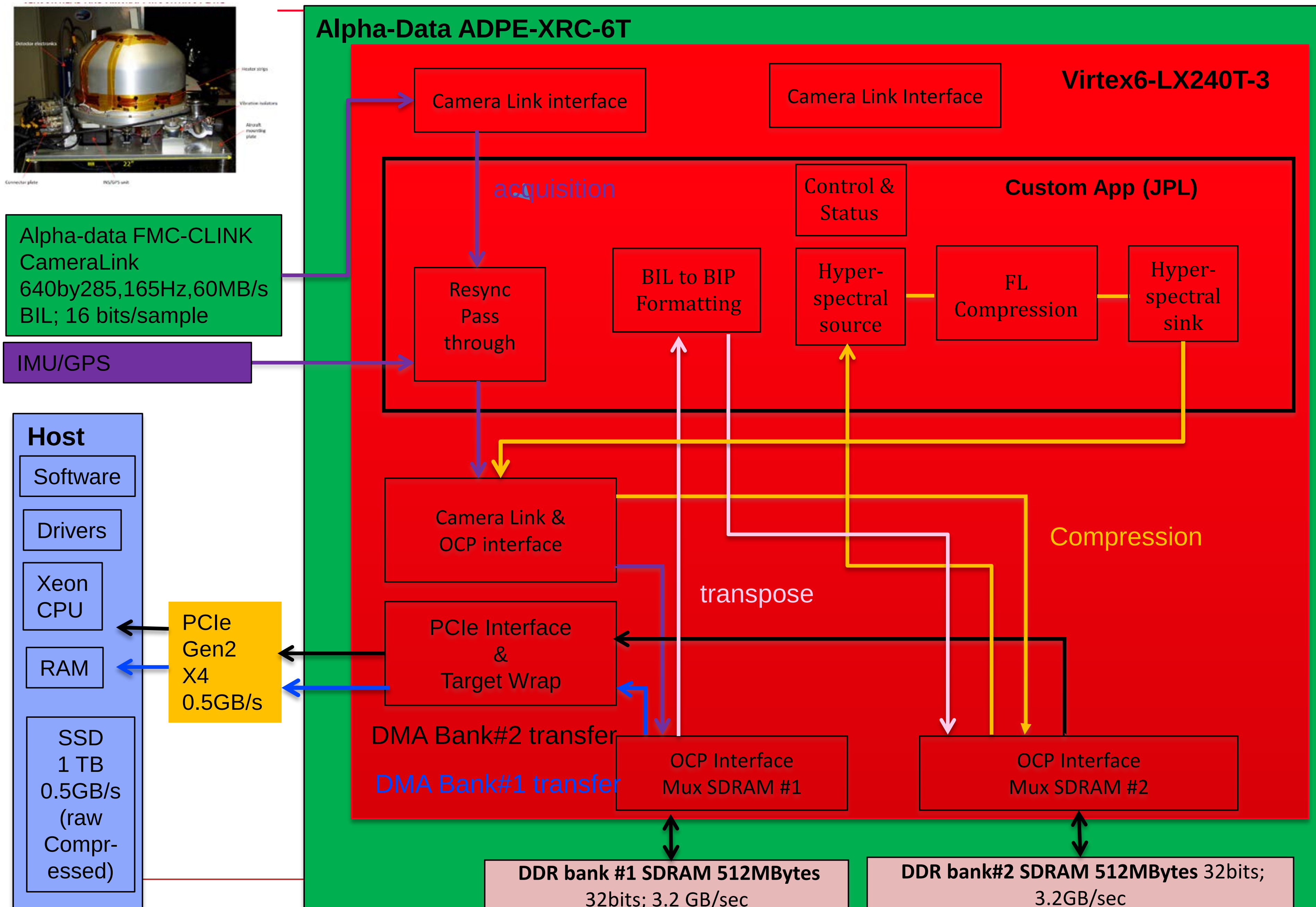
PRISM HSI

PRISM HSI Support Equipment

FL FPGA IP Main Block Diagram



FL FPGA Architecture



FL FPGA Resource Utilization – Virtex6

Device Utilization Virtex6-LX240T-3 (Compressor and Interface)

	Available	Used	Utilization All	Utilization Compressor	Utilization Virtex5 Compressor (estimate)
Slice Register (Flip-Flop)	301,440	37,284	12%	4%	8%
Slice Look-up-table (LUTs)	150,720	37,374	24%	8%	8%
Fully used LUT-Flip Flop pairs	50,693	19,105	38%	13%	26%
Block RAM/FIFO	416	108	25%	12%	12%
DSP 48eS	768	6	1%	1%	1%

Device Utilization SDRAM (AVIRISng)

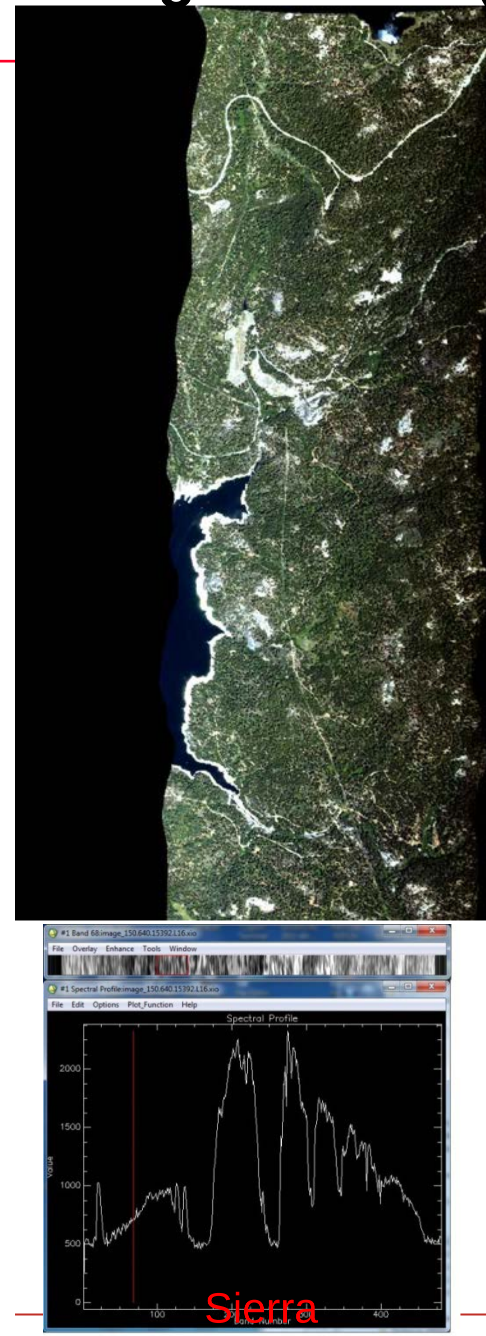
	Available	Used	Utilization
SDRAM Bank#1 (2 segments)	256 MBytes	40 MBytes	20 %
SDRAM Bank#2 (3 segments)	256 MBytes	60 MBytes	24 %

Timing: Critical Path

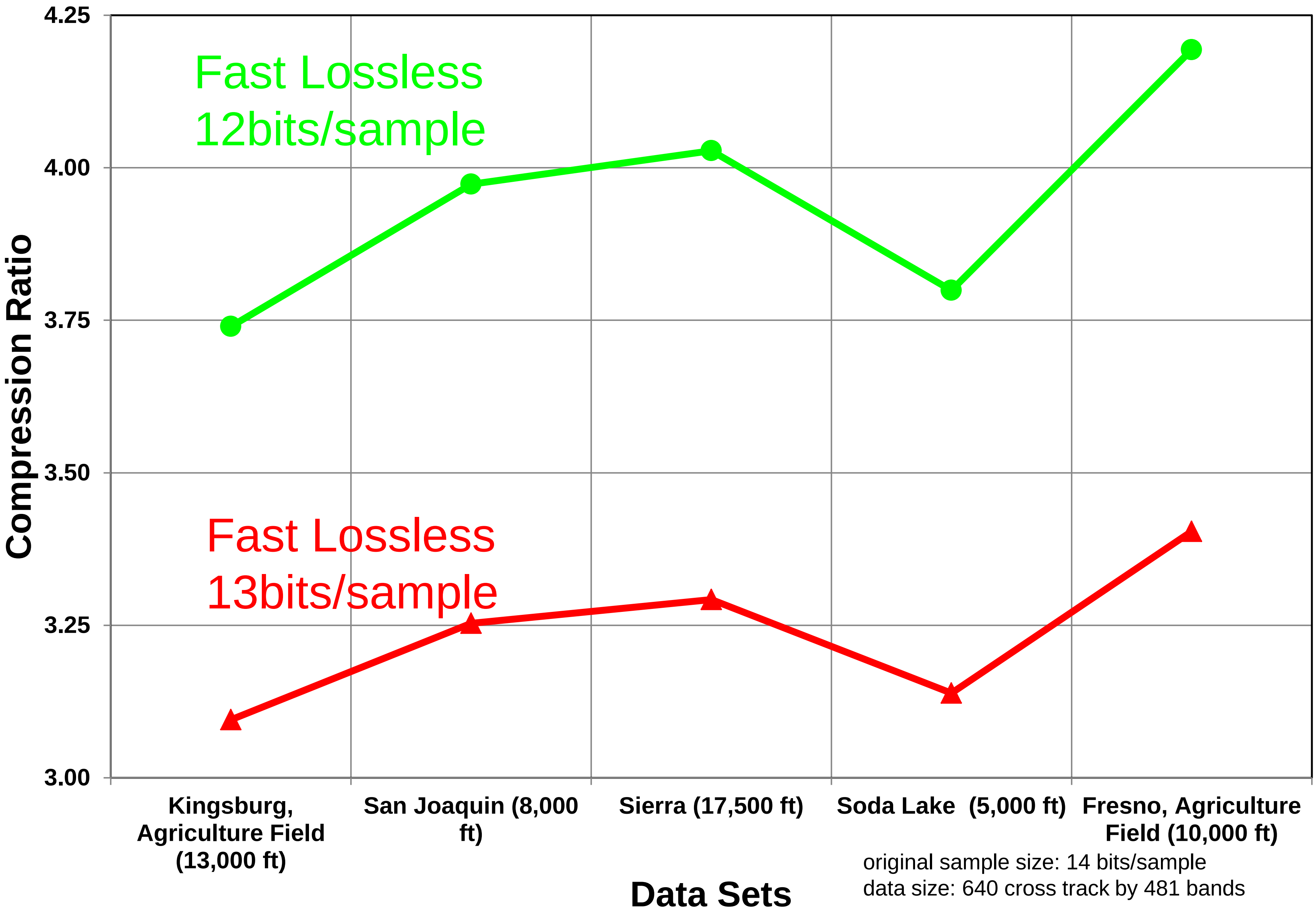
Block	Critical Path Timing
Synchronization frames with IMU/GPS	<25ns
Transpose BIP to BIL	<10ns
Predictor	12.070 ns
Entropy Encoder	10.029 ns
Packer	7.377 ns

The implementation compresses one sample every clock cycle, which results in a speed of 40 MSample/sec

Comparison during airbone AVIRISng mission (June 2014)



Comparison during airborne AVIRISng mission (June 2014)



Summary

We presented an FPGA implementation of a novel hyperspectral data compression algorithm and its flight demonstation: JPL adaptive Fast Lossless compressor.

The implementation targets the Xilinx Virtex FPGAs and provides an acceleration of at least 7 times the software implementation on a single core of the Intel® Hex Core™ i7, making the use of this compressor practical for satellites and planet orbiting missions with hyperspectral instruments.

Future development will provide multiple implementations and near lossless data compression for accommodating large Focal Plane Array (FPA). We will also develop options to deploy various versions of the algorithm to accommodate data from different instrument types as well as radiance and reflectance data. And finally explore new hardware technologies such as System-on-the-Chip (SoC) to embed the compression next to the FPA ROI and fast I/O interface to the instrument (e.g. optical).