Smart Payload Development for High Data Rate Instrument Systems

Paula J. Pingree
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive,
M/S 169-315
Pasadena, CA 91109
818-354-0587
Paula.J.Pingree@jpl.nasa.gov

Charles D. Norton
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive,
M/S 169-315
Pasadena, CA 91109
818-393-3920
Charles.D.Norton@jpl.nasa.gov

Abstract— Instruments proposed to satisfy measurement objectives of future NASA AOs, and the NRC Decadal Survey missions, will need on-board processing technologies to support autonomy, operations, and science data processing. Traditionally, payload design flows down from measurement objectives, to algorithm design, to realization on a computational platform with limited capabilities. We will describe a more integrated payload systems design approach applicable to the next generation of high data rate instruments for Earth and planetary systems, as well as the current challenges in implementing state-of-the-art on-board algorithms on the latest FPGA[1] hardware.

Two example systems will be discussed; the MATMOS[2] FTIR spectrometer design and a hyperspectral Sea, Water, Ice and Land (SWIL) classification algorithm. We will show, via demonstration of a sample FFT[3] algorithm implementation that is part of our conceptual FPGA-based computational architecture, how smart payload development techniques for the MATMOS spectrometer, proposed to Mars Scout, can reduce on-board instrument data volume by a factor of 80. The originally proposed MATMOS design used 2 RAD750 processors requiring more power, mass and volume than the FPGA solution. We also discuss optimization challenges in synthesizing to FPGA fabric a legacy C-code SWIL algorithm (for the Hyperion instrument deployed on EO-1). Our initial results show that a non-optimized algorithm uses 34% of the fabric's DSPs[4] for floating point computation, but the more desirable optimized approach uses 170%, inhibiting synthesis for the selected single FPGA.

Such technology challenges associated with designing on-board data processing payload systems for advanced instruments will be described. We assert that designing an FPGA-based computational instrument platform would serve as a sharable component to meet the on-board processing requirements of such instruments.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Smart Payload Motivation

On board computation has become a bottleneck for advanced science instrument and engineering capabilities. Currently available spacecraft processors have high power consumption, are expensive, require additional interface boards, and are limited in their computational capabilities. Recently developed hybrid field-programmable gate arrays (FPGAs), such as the Xilinx Virtex-4, offer the versatility of running diverse software applications on embedded processors while at the same time taking advantage of reconfigurable hardware resources all on the same chip package. These tightly coupled hardware/software co-designed systems are lower power and lower cost than

[1] Field Programmable Gate Array (FPGA)
[2] Mars Atmospheric Trace Molecule Spectroscopy (MATMOS)
[3] Fast Fourier Transform (FFT)
[4] Digital Signal Processor (DSP)

general-purpose single-board computers (SBCs), and promise breakthrough performance over radiation-hardened SBCs, leading to a new architecture for Smart Payload development (Table 1).

Computational Platform	Performance (DMIPS)	Power (W)
BAE RAD750 SBC	240	5
Xilinx Virtex-II Pro	450	0.5
Xilinx Virtex-4	680	*TBD*

Table 1. SBC vs. Embedded FPGAs

Designs based on embedded FPGA processors also benefit from the following advantages over SBCs:

- Higher level of reuse
- Reduced risk of obsolescence
- Simplified modification and update
- Increased implementation options through modularization

In 2006 we explored a Smart Payload concept for an instrument proposed for a 2011 Mars Scout mission that was facing on-board computational challenges. A paper titled, “An FPGA/SoC[5] Approach to On-Board Data Processing Enabling New Mars Science with Smart Payloads” was presented at the 2007 IEEE Aerospace Conference [1]. This paper summarizes this work and additionally presents new smart payload development in the area of on-board classification algorithms.

1.2 The MATMOS Instrument On-Board Data Processing Challenge

The most recent Mars Scout competition, for the 2011 launch opportunity, included the MARVEL[6] proposal mission concept (PI: Dr. Mark Allen, JPL). The goal of the MARVEL Scout mission is to find evidence of active Martian volcanism and life [2]. The primary science instrument on the MARVEL payload is a solar occultation Fourier Transform Spectrometer (FTS) called MATMOS for ultra-sensitive detection of trace gases such as CH_4, N_2O that might be produced by life or volcanism.

MATMOS will produce vast volumes of raw data in short (3 minute) bursts during the Martian sunrise and sunset (Figure 1). Getting all of this data back to Earth is prohibitively expensive. Fortunately, the remainder of the time (112 minutes per orbit) is available for science data processing and compression. Such on-board data processing is highly cost-effective for planetary missions because it reduces downlinked data volume by a factor ~80 as well as DSN access time which also impacts cost.

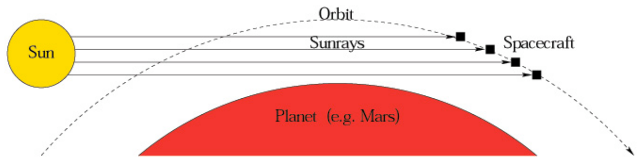


Fig. 1. The solar occultation observational geometry; two opportunities per orbit.

Raw interferogram data are read directly into MATMOS instrument memory (2 GByte) during the 3-minute occultation. On-board processing converts raw interferogram data into averaged and compressed spectra during time remaining in each orbit. Processing cannot afford to fall behind, as there is only sufficient memory for 2 orbits of raw interferogram data. Once per orbit, compressed spectra are sent from the MATMOS instrument to the spacecraft computer for subsequent downlink to Earth. The summarized steps of on-board data processing are listed below. *Note: a full description of the MATMOS instrument and it’s on-board processing is provided in [1].*

1. An observation every 2.5 sec. leads to 192 kHz * 2.5 s = **480,000** raw samples/channel.
2. After re-sampling, 2^{18} = **262,144** HgCdTe points and 2^{19} = **524,288** InSb points[7]
3. Time-domain data stream re-sampling to path difference domain reduces data volume by a factor of **1.5**
4. **(2*480000) / (262144 + 524288) * 1.5 = 1.83**, combined re-sampling data reduction
5. FFT phase correction by convolution reduces data volume by a factor of **2**
6. FFT spectrum computation reduces data volume by a factor of **6.1**
7. Further processing of the spectra (includes averaging scans taken above atmosphere) reduces data volume by a factor of **2**
8. Standard image compression techniques applied before transmission to S/C computer reduces data volume by a factor of **1.8**
9. Total combined data reduction = **1.83 * 2 * 6.1 * 2 *1.8 = 80**

[5] System-on-a-chip (SoC)

[6] Mars Volcanic Emission and Life (MARVEL)

[7] HgCdTe = Mercury Cadmium Telluride
InSb = Indium Antimonide; the two MATMOS detectors

1.3 *MATMOS Baseline Design Trade*

In late 2006, MARVEL examined the capability of the Xilinx Virtex-II Pro (V2P) FPGA and the BAE RAD750 to handle the extensive on-board data processing required for MATMOS. As Table I shows, the V2P requires significantly more time to perform the same data processing steps as the RAD750. This is primarily due to two reasons: smaller cache size and lack of a hardware floating-point unit in the V2P.

TABLE I
RESULTS FROM VIRTEX-II PRO RESEARCH TASK [11]

Processor:	RAD750	Xilinx	Xilinx
Operating System:	VxWorks	Linux	Linux
Clock Speed:	133 MHz	300 MHz	300 MHz
Memory:	128 MB	128 MB	128 MB
Software Component			w/Perflib
Reject Dark Interferograms	<1	<1	<1
Interferogram Re-sampling	69	3404	780
Non-Linearity Correction	1	14	4
Phase Correction	42	488	142
Fast Fourier Transformation	15	272	90
Spectra Averaging	2	(10)	(3)
Lossless Compression	1	1	1
Total (112 min available)	**130 min**	**4200 min**	**1020 min**

MATMOS data processing relies heavily on FFT calculations. The Xilinx Virtex-II Pro FPGA (V2P) does not have a built-in floating-point unit (FPU) thereby requiring that all floating-point operations be emulated in the software. This rendered it incapable of meeting MATMOS's data processing requirements. Even with the integration of a soft-core FPU, the V2P still could not finish the necessary calculation in the time allotted. The small (16 KB) V2P L1 data and instruction cache was also a contributing factor [3]. In an application such as this, the tremendous data processing requires frequent accesses to external memory. A larger L1 cache means that more data can fit in this high-speed memory resulting in fewer cache misses and fewer accesses to main memory, which carries with it a high latency.

The higher power, higher cost, much larger BAE RAD750 (radiation hardened) processor nearly met the requirements (130 min. vs. 112 min.) and therefore, not one but two RAD750s are currently baselined just for MATMOS at a significant cost in critical resources to the MARVEL mission.

2 VIRTEX-4FX DESIGN TRADE

2.1 *The new Virtex-4 FX with APU*

The newer (2006 release) Xilinx Virtex-4 FPGA could potentially be flight qualified in time to serve a 2011 Mars Scout mission, and our recent research explored its capabilities. The Xilinx Virtex-4 FPGA is a relatively new device that comes with an impressive set of features. The Virtex-4 is designed for lower power consumption with higher logic density and extensive DSP and embedded processing capabilities. The Virtex-4 is built in three families: LX (logic), SX (DSP applications), and FX (embedded processing). Our design is targeted at the FX family (V4FX). Virtex-4 FPGAs in this family feature one or two embedded PowerPC405 cores. These processor cores can be clocked at up to 450MHz, thus providing extreme performance in a mixed HW/SW environment. Arguably the most notable feature of the V4FX is the auxiliary processor unit (APU) that provides a flexible high bandwidth interface for direct connection of fabric co-processor modules (FCM) to the integrated PowerPC405 core. This interface integrates directly into the processor pipeline and allows for the development of custom hardware accelerators. It is the APU that sets the V4FX apart from the V2P and makes it possible to meet MATMOS's data processing requirements [4]. An FPU is an example of an FCM that when coupled with the APU, we found, could provide enough computational power on the V4FX to meet MATMOS's data processing requirements. Other soft-core coprocessors may also be integrated to further improve performance.

2.1.1 *Our APU-FPU System*

We built a simple embedded processing system with an APU-based floating-point unit (APU-FPU) on Xilinx's ML403 development board. The ML403 board is based on the V4FX12 FPGA that includes a single PowerPC405 processor core [5]. The resource utilization of this system is shown in Table II. The ML410 column is given for comparison (it was not implemented).

TABLE II
RESOURCE UTILIZATION OF THE APU-FPU SYSTEM

Resource	Usage	ML403 (V4FX12)	ML410 (V4FX60)
Slices	3,455	63%	14%
BRAM	34	94%	15%
XtremeDSP	4	12%	3%

The performance of this system was evaluated with a standalone C-program that calculated a 1,048,576-point FFT used to benchmark computational efficiency. The same program was run in three separate instances on the FPGA and compared to the performance measured on the Motorola MCP750 single board computer. The MCP750 is a commercially available single board computer with a PowerPC750 processor and is nearly identical to BAE's RAD750 currently targeted for MATMOS [6]. The MCP750 runs at 233 MHz while the RAD750 is clocked at 133 MHz.

To get a strong indication of the performance improvement seen with the APU-FPU, the FPGA embedded processing system was evaluated in three configurations (Table III):

- No FPU, software FP emulation through GCC
- No FPU, software FP emulation through IBM Perflib
- Hardware acceleration via APU-FPU

In all three configurations the CPU frequency was 200 MHz and the FPU frequency was 100 MHz.

TABLE III
CURRENT VIRTEX-4FX WORK RESULTS (APU-FPU)

Xilinx Virtex-4FX12 PowerPC405 No OS (GCC 3.4.1, patched) CPU Freq: 200 MHz FPU Freq: 100 MHz	FFT Time / Iteration (sec)	Speedup (vs. base case)
No FPU	40	base case
No FPU, Perflib	16	2.50
APU-FPU	3.56	11.24

(a) Virtex-4FX12 Results

Motorola MCP750 PowerPC750 VxWorks OS / Compiler Freq: 233 MHz	FFT Time / Iteration (sec)	Speedup (vs. base case)
Hardware FPU	0.637	62.79

(b) MCP750 Results

The most significant speedup with the V4FX, as expected, was seen with the floating-point unit integrated in hardware and connected to the PowerPC405 core via the APU. This APU-FPU implementation executed each FFT iteration at 3.56 seconds. This is a tremendous speedup of 11.24x over GCC software emulation and more than four times faster than emulation via Perflib. Even with such a notable speedup over plain software emulation, the APU-FPU system is still nearly six times slower at executing the same FFT as the Motorola MCP750 single board computer.

The performance results obtained from the APU-FPU system suggest that it is not possible to meet MATMOS's data processing requirements with one single-core processor FPGA. There are many optimization techniques, however, that have yet to be tried. Rewriting the various algorithms may allow for a more efficient use of the hardware FPU, for example loop unrolling and instruction re-ordering could minimize FPU pipeline stalls. Breaking up the large FFT into smaller FFTs that could execute in sequence is also a promising optimization technique. Nevertheless, even with the results as they are at this moment, it seems possible to meet MATMOS's data processing requirements in a dual processor core, multi-FPGA system.

2.1.2 Applying Results to MATMOS

Without investigating additional optimization techniques, the following calculations show how a dual-core, dual-FPGA system can potentially meet MATMOS's data processing requirements. This is only a rough estimate, assuming full data parallelism between cores, and still needs to be evaluated.

- MATMOS needs **<u>1.2 x RAD750</u>**, 133 MHz
    - o 112 minutes available for science data processing
    - o RAD750 needs 130 minutes = 1.16*112 min
- Scaled down MCP750 is **<u>3.2x faster</u> than V4FX**
    - o MCP750 (233 MHz) is 5.59x faster than V4FX12 (200 MHz)
    - o MCP750 is 233 MHz, scale down to 133 MHz to compare (0.57x)
    - o Real speed factor: 3.19 = 5.59*0.57
- Use dual core FPGA: MCP750 = **<u>1.6 x dual core V4FX</u>** (3.2 / 2)
- For flight: 1.2*1.6 = **<u>1.9 x dual core V4FX</u>**

Thus, it seems possible that a dual-core, dual-V4FX system can meet MATMOS's data processing requirements. This will be a low power, low cost, low mass, customizable, and reconfigurable solution. The calculations above suggest a small margin that will probably not be acceptable for a flight system. However, with additional optimization techniques yet to be tried, the margin is expected to increase significantly.

3 SYSTEM-LEVEL CONCEPTS

3.1.1 An FPGA Instrument Computer for Space

With the results above in mind, a future smart payload computer for space is likely to have multi-processor FPGAs. As an example, such a computer might contain two dual-core V4FX FPGAs running at 200-400 MHz. The two FPGAs, fully configured, would consume less then 10 W of total power (a worst-case estimate). The FPGA processors could run Linux, VxWorks, the Xilinx MicroKernel (XMK), or have a dedicated standalone application managing all processes and resources. A good candidate for development is Xilinx's ML410 evaluation board (Figure 2).

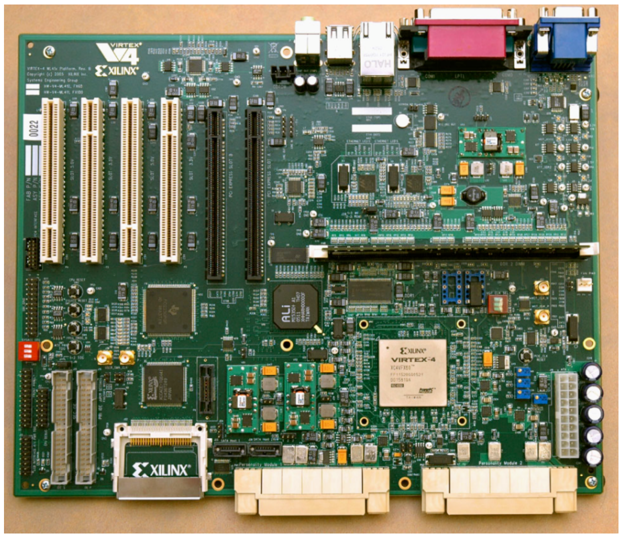


Fig. 2. A good candidate for the development of a future instrument computer for space. The Xilinx ML410 evaluation board comes with the V4FX60 FPGA that features two embedded PowerPC405 processors [5].

This smart payload computer will require a large amount of memory to hold all of the unprocessed raw data coming directly from the instrument. The memory will consist of a large amount of block ram (on chip – up to 1 MB), and the system board itself will need to have a lot of off chip RAM (GBytes). An average size PROM will be included to support saving and restoring hardware configuration images. The L1 cache would be configured to support all onboard memories (except PROM). The I/O subsystem on this instrument computer may include interfaces such as 1553B, Ethernet, RocketIO (MGTs), Firewire, and serial (RS232). The hardware/software system will need to have triple modular redundancy (TMR) fault tolerance to address single event upsets (SEU) due to higher levels of radiation in space.

Breakthrough performance may be achieved with such a smart payload computer if custom co-processors are implemented in the FPGA fabric. These co-processors will address various computationally intensive algorithms that can benefit significantly from a parallelized, hardware implementation. Such co-processors may include the APU-FPU (as demonstrated in this paper), a dedicated FFT core, filters (FIR), and many others specifically targeted at signal and image processing. The possibilities are endless and they are just becoming available today.

3.2 *A MATMOS Computer Concept*

Applying the features discussed in the previous section and taking into account the results obtained from the APU-FPU work, a MATMOS computer may look similar to the concept in Figure 3.

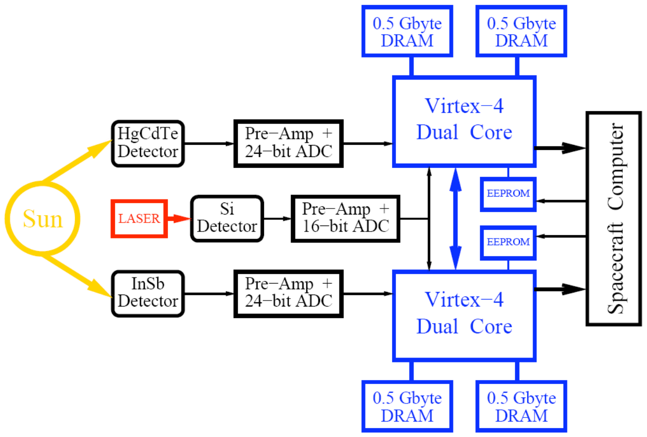


Fig. 3. A MATMOS computer concept that will meet the data processing requirements.

As discussed previously, using two dual-core V4FX FPGAs should be sufficient to meet MATMOS's data processing requirements. Additionally, all external off chip interfaces (e.g., the memory and ADC interfaces) will be implemented in the FPGA fabric, thus eliminating the need for various logic boards to provide the necessary interfaces.

3.3 *Further Software Optimizations*

There is still a lot of work to be done that could lead to further optimization of the resources available on the FPGA. For example, smaller FFTs may be used in sequence in order to have the data fit in the L1 cache. The problem with using a large FFT is that the data does not fit in the cache resulting in cache misses and the need to fetch the data from external memory, which brings with it a high latency. This optimization in particular has the potential for a significant performance increase.

Other optimizations to be evaluated include:

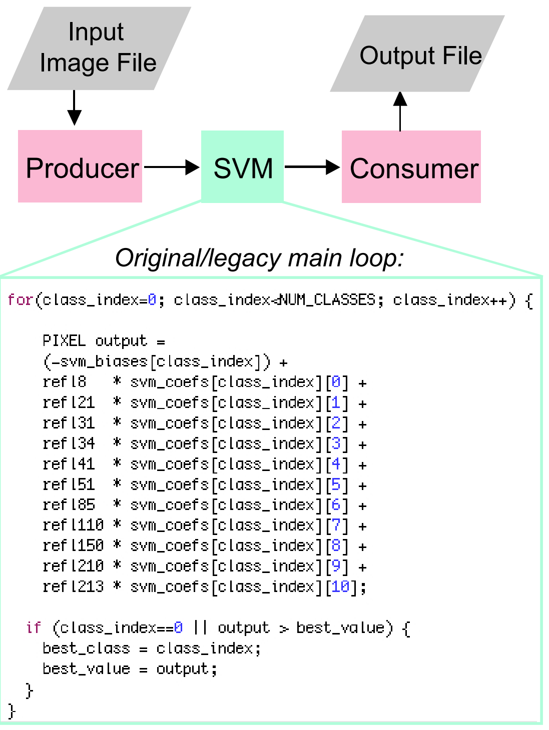
- Support for vector math operations with hardware acceleration
- OS (Linux, VxWorks, XMK) with access to APU co-processors
- Additional co-processors targeted at specific algorithms (example: 5.25 ms 1,000,000 point FFT core, on the market today)
- Optimize APU-FPU utilization
    - Parallelize code to take advantage of pipelined multiply and add/sub
    - Remove data dependencies
    - Evaluate the use of optimized fused multiply-add/sub operation
- Further optimizations in memory
    - Look into different memory configurations
    - Look into different cache options

Besides optimizing the performance of the FPGA embedded processing system, it is also important to look into space flight qualification. SEU mitigation is also an important issue that must not be overlooked.

4 CLASSIFICATION APPLICATION

JPL has developed classification algorithms that can be used onboard spacecraft to identify high priority data for downlink to Earth and to provide onboard data analysis to enable rapid reaction to dynamic events. To meet NASA's science objectives these classifiers detect flooding, volcanic eruptions and sea ice break-up. Current pixel-based machine learning and instrument autonomy algorithms that have successfully detected and identified various natural phenomena are flying on computational technologies such as the RAD6K and Mongoose V processors that have limited computing power, extremely limited active storage capabilities and are no longer considered state-of-the-art.

With limited seed funding we have implemented, on the Virtex-4FX60, a linear Support Vector Machine (SVM) classification algorithm. This migration to a low power, high-speed FPGA computing platform adds flexibility and scalability to the system. The development phase of this FPGA co-design effort focused on partitioning the previously software-only legacy algorithm into portions to be parallelized and implemented in the FPGA hardware fabric to take advantage of high-speed parallel processing capabilities. Figure 4 illustrates the partitioned system.



```
for(class_index=0; class_index<NUM_CLASSES; class_index++) {

    PIXEL output =
    (-svm_biases[class_index]) +
    refl8   * svm_coefs[class_index][0] +
    refl21  * svm_coefs[class_index][1] +
    refl31  * svm_coefs[class_index][2] +
    refl34  * svm_coefs[class_index][3] +
    refl41  * svm_coefs[class_index][4] +
    refl51  * svm_coefs[class_index][5] +
    refl85  * svm_coefs[class_index][6] +
    refl110 * svm_coefs[class_index][7] +
    refl150 * svm_coefs[class_index][8] +
    refl210 * svm_coefs[class_index][9] +
    refl213 * svm_coefs[class_index][10];

  if (class_index==0 || output > best_value) {
    best_class = class_index;
    best_value = output;
  }
}
```

Figure 4. FPGA co-design for the SVM algorithm

The Producer is coded in a file called sw.c. It reads an input image file containing 857,856 pixels and streams data to the SVM. The Consumer, also coded in sw.c, streams data from the SVM and writes pixel classifications (e.g., snow, water, ice, land, cloud, or unclassified) to an output file. The original legacy SVM code is put in a file called hw.c. The SVM algorithm in hw.c was transformed from C-to-HDL using the CoDeveloper^{TM} tool set by *ImpulseC [7]* and simulated to validate execution of the co-designed system. The sw.c program will execute on the V4FX embedded PowerPC processor and communicate with the hardware-accelerated algorithm in the FPGA fabric.

Next the converted HDL code was synthesized for the Virtex-4FX FPGA using the Xilinx ISE & EDK development environment. Synthesis determined the V4FX resource output for the SVM algorithm to be:

- 10 Adders/Subtractors (6 bit)
- 1 Adder/Subtractor (32 bit)
- 11 Multipliers (3 bit)
- 1 Comparator (2 bit)
- 2 Comparators (32 bit)
- 12 Floating Point Adders/Subtractors (32 bit)
- 11 Floating Point Multipliers (32 bit)

This translates to the following resource utilization for the V4FX60 device on the ML410 development platform. Note the ML410 is the same platform targeted for the MATMOS concept.

FPGA RESOURCES	**V4FX60 (on ML410)**
Number of Slices:	8495 out of 25280 **(33%)**
Number of Slice Flip Flops:	9640 out of 50,560 **(19%)**
Number of 4 input LUTs:	10495 out of 50560 **(20%)**
Number of DSP48s:	44 out of 128 **(34%)**
Maximum frequency:	129.95 MHz

We tried to synthesize an optimized version of the SVM algorithm that unrolled the main loop and redefined matrices to allow parallel access, however this design did not fit on the V4FX60. Next steps for this task would be to download the successfully generated hardware/software image to the ML410 development platform for implementation and evaluation. Other Xilinx development platforms with higher capability FPGAs could also be investigated for the optimized design [5].

5 CONCLUSION

FPGAs with embedded processing capabilities will in the near future demonstrate breakthrough performance previously impossible with traditional processors. This paper presented results from work done to evaluate the performance of the V4FX12 FPGA and the recently released APU floating-point unit against the data processing requirements of the MATMOS instrument, proposed for a 2011 mission to Mars. Together, the V4FX12 and the APU FPU achieve performance previously unattainable with FPGA embedded processing. The current results suggest that a dual-core, dual-V4FX60 system can meet MATMOS's data processing requirements with lower power and cost as well as smaller size and weight. Furthermore, the optimizations yet to be tried have tremendous potential to reach a single-FPGA and, perhaps, a single embedded processor core solution in the near future.

Additionally, this paper presented preliminary results from the synthesis of a legacy software SVM classification algorithm to the V4FX60 FPGA platform. Using C-to-HDL translation tools this work was made possible with very limited funding. Hardware acceleration of such algorithms is promising to provide capabilities for more advanced on-board classification algorithms in future missions.

6 ACKNOWLEDGEMENTS

Dmitriy Bekker (Rochester Institute of Technology graduate student), Jean-Francois Blavier (JPL), and Geoff Toon (JPL) for their contributions to the MATMOS FPGA development effort.

Lucas Scharenbroich (JPL) for his contribution to the SVM Classification Algorithm FPGA development effort.

Abdullah Aljabri (JPL) for his vision and support of Smart Payload technology development.

7 REFERENCES

[1] P. Pingree, Jean-Francois Blavier, Geoff Toon, Dmitriy Bekker, *"An FPGA/SoC Approach to On-Board Data Processing Enabling New Mars Science with Smart Payloads"*, Proceedings of 2007 IEEE Aerospace Conference.

[2] G. Toon, P. Wennberg, M. Allen, M. Richardson, and J. Eiler, *"Solar Occultation FTIR at Mars: A Space Odyssey for 2011,"* Presentation. Available: http://www-sosst.larc.nasa.gov/meetings/2004/0615/presentations/22-geofftoon.pdf

[3] *Virtex-II Pro and Virtex-II Pro X Platform FPGAs: Complete Data Sheet (DS083 v4.5)*, Xilinx Inc., San Jose, CA, 2005. Available: http://direct.xilinx.com/bvdocs/publications/ds083.pdf

[4] *Virtex-4 Family Overview (DS112 v1.5)*, Xilinx Inc., San Jose, CA, 2006. Available: http://direct.xilinx.com/bvdocs/publications/ds112.pdf

[5] *Xilinx Development Boards*, Xilinx Inc., San Jose, CA, 2006. Available: http://www.xilinx.com/publications/matrix/matrix_XDev_boards.pdf

[6] *MCP750 CompactPCI Host Slot Processor*, Motorola Computer Group, Temple, AZ, 2001. Available: http://www.acttechnico.com/mcp750.pdf

[7] David Pellerin and Scott Thibault, *"Practical FPGA Programming in C"*, Prentice Hall, 2005. http://www.impulsec.com/

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

8 BIOGRAPHY

Paula Pingree is a Senior Engineer in the Instruments and Science Data Systems Division at JPL. She has been involved in the design, integration, test and operation of several JPL flight projects, most recently Deep Impact (DI). She has worked on the Tunable Laser Spectrometer development for the 2009 Mars Rover and is presently the Electronics CogE for the Juno Mission's Microwave Radiometer. She also enjoys research and technology development for Smart Payloads "in her spare time." Paula has a Bachelor of Engineering degree in Electrical Engineering from Stevens Institute of Technology in Hoboken, NJ, and an MSEE degree from California State University Northridge. She is a member of IEEE.

Charles D. Norton is Principal Engineer in the Instruments and Science Data Systems Division and Supervisor of the Model-Based Systems Engineering Group at JPL. His work involves Earth and space science modeling with an emphasis on high performance computing applications. He also leads efforts in model-based validation of precision deployable structures and advocates for development of smart payload instrument concepts. Charles has a BSE in Electrical Engineering and Computer Science from Princeton University, and an MS and PhD in Computer Science from Rensselaer Polytechnic Institute. He is a Senior Member of IEEE, recipient of the 2004 JPL Lew Allen Award for Excellence, and a 2006 NASA Exceptional Service Medal.