High Performance Space VPX Payload Computing Architecture Study

Richard Alena NASA Ames Research Center MS269-4, Rm 195 Moffett Field, CA 94035-1000 650-604-0262 richard.l.alena@nasa.gov	Patrick Collier Air Force Research Laboratory Albuquerque, NM 505-853-1209 Charles.collier10@us.af.mil	Mohammad Ahkter Barry Wood Integrated Device Technology Corp. Ottawa, Ontario Canada 613-592-0859 mohammad.akhter@idt.com	Soumik Sinharoy Orange Silicon Valley San Francisco CA soumik.sinharoy@orange.com	Deepak Shankar Mirabilis Design Sunnyvale, CA Deepak.shankar@mirabilisdesign.com
--	---	---	--	--

Abstract— This paper describes a functional reference design for a high-performance payload processor that captures images and spectra from multiple high-resolution instruments, processes and integrates multiple real-time data streams to perform feature recognition and spatial transformations providing autonomous navigation and rendezvous capability for future spacecraft and is equally applicable to Unmanned Aerial Systems (UAS). The proposed design uses two new standards: VITA 78 (SpaceVPX) for multi-processor architecture, and RapidIO (RIO) as the interconnect fabric. The SpaceVPX standard specifies physical form factor, logical, and physical interconnect technologies and architectures that can lead to high-performance fault tolerant computing for high-performance payloads. An overview of SpaceVPX and its relationship to OpenVPX is provided as a guide to practical implementations. The proposed design features a general-purpose host processor with GPU and FPGA-based image processing hardware. RIO is used for the instrument and processor interconnects, providing multiple gigabits per second of data communication capability. An overview of RIO features and operation is presented to complement the SpaceVPX architecture.

A notional Reference Architecture is proposed for analysis using multiple methods for estimating avionics performance. The study objectives are to characterize throughput, latency and sub-system utilization using conventional system analysis, hardware prototype measurements and modeling and simulation software. We conducted first-order performance studies to identify bottlenecks in memory speed, I/O capacity and processing power. Initial performance analysis was performed on memory throughput rates, producing first-order values used as a performance baseline. A model of the Reference Architecture using VisualSim Architect was created and simulations run, producing insight into the complex interactions occurring between subsystems. Furthermore, the results of a prototype hardware implementation focusing on RIO throughput are presented as additional metrics. The study predicts RIO throughput between key elements of the Reference Architecture and identify major bottlenecks, and improvements needed for meeting mission requirements. The objective of this paper is to provide guidance to avionics designers regarding the adoption of SpaceVPX today and its anticipated evolution in the next few years.

TABLE OF CONTENTS

1. INTRODUCTION	1
2. AUTONOMOUS EXPLORATION OPERATIONS	2
3. REFERENCE ARCHITECTURE.....	3
Performance Requirements	3
4. SPACEVPX OVERVIEW	4
OpenVPX Standards.....	4
Fault Tolerance	4
Profiles.....	5
Mechanical	5
Interoperability with OpenVPX	6
Specification Overview	6
5. RAPIDIO PROTOCOL OVERVIEW	6
6. AVIONICS PERFORMANCE ANALYSIS.....	8
Systems Performance Analysis	8
Performance modeling.....	9
Hardware Prototype Performance	11
7. REFERENCE ARCHITECTURE ANALYSIS.....	12
OpenVPX Implementation	14
8. CONCLUSIONS.....	14
APPENDIX.....	16
ACKNOWLEDGEMENTS	17
REFERENCES.....	17
BIOGRAPHY.....	17

1. INTRODUCTION

Future space missions will require next-generation computing hardware with significant performance increases to handle the anticipated data rates from earth observation and planetary science instruments. Moreover, as the target for such missions gets more distant, the need for autonomous operations (not coordinated directly in real-time from earth), also increases significantly. Such capability requires advances in host computer processing, instrument interface performance and payload data processing and communications. Of particular interest would be vision-based guidance and navigation capabilities, the foundation for remote autonomy.

The VITA 78 (SpaceVPX) technology standard complements the terrestrial OpenVPX standard, and products conforming to SpaceVPX will be available on the market in the next couple of years. The SpaceVPX standard offers advantages in terms of size, weight and power consumption as well as supporting fully redundant operation and broad support for co-processing functions. This paper provides an overview of the SpaceVPX standard and the RapidIO (RIO) standard, which provide the foundation and framework for the integration of the multiple functions needed for a complex spacecraft.

We define a Reference Architecture, based on OpenVPX, which represents a single chassis solution for an autonomous spacecraft able to determine its orbit, acquire high-resolution images and spectra from multiple instruments, process the instrument data to extract features providing navigation and observation information, to make independent decisions regarding minor orbital corrections and to identify and schedule observations of interest without ground control intervention. Much of this capability is anticipated to be performed by a vision processing system, either hosted in a GPU or in a dedicated FPGA. Of course, such a spacecraft would still be able to operate in the conventional manner, but this paper will focus on the functions needed for autonomy, which place the greatest requirements on real-time instrument data processing and data handling performance.

While the Reference Architecture is useful as a guide for designing and evaluating future Command and Data Handling (C&DH) and payload data processing systems, quantitative performance is key for ensuring the proposed design will meet spacecraft and mission performance requirements. After applying conventional system analysis to the memory array throughput, the quantitative performance of the Reference Architecture is predicted based on the computational performance modeling provided by Mirabilis Design's VisualSim Architect tool focusing on RIO throughput in a redundant configuration. The modeled performance is compared against a prototype hardware implementation using RIO in a single string configuration and the metrics compared. The RIO interconnect fabric provides scalable performance, which may or may not throttle the performance. We provide figures of merit for each of the major functions and quantify the predicted performance in each of these categories.

This paper is the result of an informal collaboration of engineers from a variety of organizations. NASA Ames Research Center is coordinating the paper in association with the Air Force Research Lab (AFRL) who focuses on VPX Standards. Contributions from Integrated Device Technology Corporation in the area of RIO performance and architecture, Orange Silicon Valley in the area of vision system performance and Mirabilis Design for performance modeling help complete the effort. The various contributors are interested in the area of vision-based autonomy, so have

current hardware and software prototypes demonstrating the technology.

The goal of this paper is to inform avionics designers of the advantages of building prototypes upon the OpenVPX standard toward full implementation aboard spacecraft and aircraft using the SpaceVPX standard. Furthermore, the performance modeling and estimates allow determining whether current products can be used for fulfilling autonomous mission requirements and where they might fall short. Finally, we try to overcome any shortfalls by determining what anticipate technological advances will result in fully meeting future requirements. The tools and methods are anticipated to be useful for a wide range of spacecraft design and mission planning activities.

2. AUTONOMOUS EXPLORATION OPERATIONS

As NASA missions go further from earth, light speed delays in communication necessitate greater on-board autonomy. In this case, autonomy is defined as independence from earth-based mission control. Given the high speed of planetary approach and landing, autonomy systems for Entry, Descent and Landing (EDL) applications must feature extremely high peak performance during critical maneuvers. Given the constraints of system mass, volume and power consumption; such a computational system must be implemented very effectively. Moreover, the computational system has to control critical functions while immersed in a highly challenging environment in terms of temperature, shock and vibration, vacuum tolerance and space radiation effects. Meeting the requirements for such a mission is a challenging engineering task requiring the best approach for computational architecture and packaging.

Another application for the on-board autonomy processor would be determining primary targets for observation and detailed measurement. In general, instruments can provide data rates far in excess of what can be downlinked over the large distances typical of exploration missions. This requires judicious selection of instrument targets and the use of compression methods to fit the large data sets into the available downlink bandwidth. In general, only a portion of an image contains items of interest, while the rest should be ignored and its impact on resources minimized. Autonomously selecting areas with interesting features, identifying regions of interest, extracting key measurements and downlinking only the relevant data will allow the spacecraft to much more effectively use its limited time and resources for remote planetary exploration, a key precursor for human exploration. Additionally, the same instruments and computational resources needed for EDL can be applied to autonomous exploration.

Autonomous space or aero vehicle operation is dependent on the availability of real time data from onboard sensors. The most challenging phase of any space mission is precision landing on any solid or liquid surface with no human input. Such a level of autonomy is reliant on the ability of the onboard computing resources to acquire and

process the sensor data, extract the features and parameters relevant to the mission phase and to facilitate real time decision making involving mission success or even human safety. With ever increasing precision of all onboard sensors (e.g. cameras and multispectral imagers, higher bandwidth communication links) it will be increasingly challenging to accomplish real-time decision making with full autonomy and cognitive reasoning based on real time situation awareness. This ability will require onboard supercomputing capability, allowing autonomous spacecraft to process aggregated data throughput of near 100 Giga-bits per second (Gbps) from multiple sensor sources, which are then correlated using sensor fusion algorithms. The features identified are then used for complex decisions such as approach trajectory to the exact landing site and execution of the docking or landing operation. The future SpaceVPX architecture is required to be scalable and rapidly reconfigurable based on mission requirements. Such scalability can be achieved with a multiple processor configuration utilizing high-speed low-latency interconnects. Based on energy efficiency and fault tolerance features, RIO is a viable candidate for standard high-performance interconnects. Low latency can enable near linear scalability for a multiple processor computing system. This study will focus on the performance of the interconnection fabric in relation to the multiple heterogeneous processors of the Reference Architecture.

3. REFERENCE ARCHITECTURE

Figure 1 shows our proposed Reference Architecture based on the OpenVPX standard. Two instruments are proposed – a very high-speed, high-definition (HD) camera able to acquire images for EDL as well as scientific purposes and a multispectral imager featuring lower resolution and lower frame rate but with the capability of imaging in bands such as the Infra-Red (IR) providing information on the objects and shadows in the camera image. Such multispectral adjuncts to visible high-speed imaging could allow disambiguation of objects leading to faster decisions regarding the trajectory and hazard avoidance required for safe EDL operation.

Figure 1. Reference Architecture

Each instrument features a dedicated image processor cluster, optimized for each instrument. For example, the HD camera features the use of a Graphics Processor Unit (GPU) for image processing, since the GPU approach has proven to be very effective for this purpose and is particularly good in terms of computational efficiency per watt of power. By contrast, the multispectral imager would use a dedicated FPGA processor, burning a lot more power, but able to implement a wide variety of image processing algorithms to make best use of each type of image produced by this instrument. This approach is very conventional and is used to highlight how SpaceVPX enables such complex architectures. Other approaches are equally valid; our Reference Architecture for the payload processor is simply one of many approaches for solving this class of problems.

The payload processor is the main host processor for this entire computational system. It is assumed that the spacecraft itself hosts a capable Command and Data Handling (C&DH) processor for flight control functions, power control and other spacecraft bus functions. An interface between the C&DH processor and the payload processor allows communication between the two computational elements. For EDL, the payload processor may very well send a list of waypoints to the C&DH processor to execute the desired descent trajectory. For science operations, the payload processor may communicate with the spacecraft for instrument pointing, downlink scheduling or power management. Finally, the payload processor uses a dedicated high-rate downlink to send science data back to earth. We have purposely selected data rates requiring significant on-board processing to reduce the instrument data rates to those able to be downlinked, but this situation is actually typical for deep space missions.

Performance Requirements

The input numbers used for performance evaluation of our Reference Architecture are typical of instruments available today at the high-end of cost and performance. A 4K x 4K pixel (Ultra HD) camera featuring a high frame rate of 100 frames/second is used as the primary vision camera, producing a raw data rate of 51 Gbps assuming a 32-bit color depth. In contrast, the multispectral IR, mid-IR and near IR imager of 2K x 2K pixel resolution with 16 bit data fields will produce just 3.8 Gbps of data, assuming 20 frames per second. While such data rates can be supported in terms of I/O, extracting useful information at these data rates is challenging.

The downlink data rate is assumed to be 1 Gbps for the purposes of this study and is considered reasonable for a high-performance optical link. Such links have been demonstrated from the moon to earth during the LADEE mission.¹ The spacecraft data link is assumed to be a modest 1 Mbps rate link, supported by many interfaces such as SpaceWire.

The purpose of assuming such data rates is to drive the performance analysis of the Reference Architecture. The

basic input/output (I/O) functions are primary, in that the data needs to be acquired at the instrument rate into main memory. At this point, the data can be processed, stored in on-board non-volatile solid-state disks, and even downlinked back to earth. Given the large mismatch between the instrument output rates and the downlink and storage rates, some degree of processing will be required just to compress the data for downlink. To support autonomous flight, the data will be processed to extract features of interest, to combine the data provided by the imager and spectrometers using sensor fusion algorithms and finally feeding decision algorithms able to guide the spacecraft upon early approach to a planetary body or asteroid. This paper focuses on the resultant performance determined by both modeling and by benchmarking, yielding important information on the methods required for implementing such data rates.

4. SPACEVPX OVERVIEW

SpaceVPX or VITA 78 is a new standard being developed under the VITA Standards Organization. It builds on the VITA 65, OpenVPX™, and extends it for space applications.² SpaceVPX is a space-capable variation of the VITA 65 OpenVPX industry standard developed as an element of the Next Generation Space Interconnect Standard (NGSIS) working group.³ The SpaceVPX standard was ratified in April, 2015. An example switched topology use case for SpaceVPX is shown in Figure 2. Data from sensors are connected to Data In modules. Data is switched between inputs, mass storage, processing nodes and output modules. Downlinks from Data Out modules represent data being sent to the ground. Note the full redundancy present among the elements; this in part satisfies the need for additional fault tolerance in space systems. A second topology fully supported by SpaceVPX involves a peer-to-peer mesh between all elements instead of using a switch. Combinations of the two topologies are also easily assembled using the standard.

Figure 2. SpaceVPX Switch Topology

OpenVPX Standards

SpaceVPX builds upon several standards that are part of the ANSI/VITA OpenVPX family. These include the base

VITA 46 VPX standard and its ANSI/VITA 65 OpenVPX derivative. SpaceVPX also allows other compatible connectors to be used, including ANSI/VITA 60 and 63. ANSI/VITA 48.2 forms the base of the mechanical extensions in SpaceVPX.⁴ ANSI/VITA 62 defines a standardized power module. ANSI/VITA 66 and 67 also may be applied to replace electrical segments of the connector with RF or optical. ANSI/VITA 46.11, currently in trial usage, provides a base of the management protocol that SpaceVPX builds upon for fault tolerant management of the SpaceVPX system.⁵

To ease the transition from bussed protocols, OpenVPX created “bridge” profiles to enable legacy VME modules to interface with OpenVPX modules. Likewise, SpaceVPX includes profiles to enable cPCI modules to interface with SpaceVPX modules. Profiles also enable code reuse from legacy systems targeting PCI for new SpaceVPX systems.

Four major interconnect planes organize the connections in OpenVPX. The data plane provides high-speed multi-gigabit fabric connections between modules and typically carries payload and mission data. The control plane, also a fabric, typically has less capacity and is used for configuration, setup, diagnostics and other operational control functions within the payload as well as lower speed data transfers. The utility plane’s function is providing setup and control of the basic modules functions typically having to do with power sequencing and low level diagnostics, as well as the power, clocks and other base signals needed for system operation. The expansion plane may be used as a separate connection between modules utilizing similar or bridging heritage interfaces in a more limited topology such as a bus or ring. Pins not defined as part of any of these planes are typically user defined and are available for pass through from daughter or mezzanine cards or to rear transition modules (RTM). For maximum module reuse, the user-defined pins should be configurable to not interfere with modules that use these pins in a different way. ANSI/VITA 65 should be consulted for more detail on these structures.

Fault Tolerance

In evaluating the use of OpenVPX for potential space usage, several shortcomings were noticed. The biggest one was the lack of features that could support a full single-fault tolerant and highly reliable configuration. Utility signals were bussed and in most cases only supported one set of signals, via signal pins to a module. A pure OpenVPX system has opportunities for multiple failures as a result. A full management control mechanism was also not fully defined with VITA 46.11. The fact that the typical OpenVPX control planes are PCI Express or Ethernet, was another shortcoming of concern since their usage in space applications was minimal and SpaceWire is the dominant medium speed data and control plane interface for most spacecraft. A third area was the desire to reuse the

infrastructure of OpenVPX for prototyping and testing SpaceVPX on the ground.

The goal of SpaceVPX is to achieve an acceptable level of fault tolerance, while maintaining a reasonable level of compatibility with existing OpenVPX components – this includes connector pin assignments for the board and the backplane. For the purposes of fault tolerance, a module (defined as a printed wire assembly which conforms to defined mechanical and electrical specifications) is considered the minimum redundancy element. The utility plane and control plane within SpaceVPX are all distributed redundantly and in star topologies to provide fault tolerance to the entire system.

To provide the required level of fault tolerance, the utility plane signals needed to be dual-redundant and then switched to each SpaceVPX card function. A trade study was performed early to compare between various implementations including adding the switching to each card in various ways as well as creating a unique switching card. The latter approach was chosen so that SpaceVPX cards could each receive the same Utility plane signals that an OpenVPX card receives with minor adjustments for any changes in topology. This became known as the Space Utility Management (SpaceUM) module and is a major contribution of the standard for 6U board size systems. (The VITA 78.1 (SpaceVPX Lite) specification under development provides system solutions that replace the SpaceUM module with a DirectUtility Control module. These new modules allow smaller size, weight, and power constrained systems a measure of signal distribution and control without the inclusion of a SpaceUM.) The SpaceUM module contains up to eight sets of power and signal switches to support SpaceVPX modules. It receives one power bus from each of two Power Supplies and one set of utility plane signals from each of two System Controller functions required in the SpaceVPX backplane. The various parts of the SpaceUM module are considered extensions of the Power Supply, System Controller and other SpaceVPX modules for reliability calculation and thus do not require their own redundancy. Two Management protocol options are provided for control of the system over the utility plane's system management interface; one is a subset of the IPMC defined in VITA 46.11; the other is a simpler direct access protocol developed specifically for SpaceVPX. Both utilize the utility plane for access to the managed modules.

Profiles

Since each Slot, Module, and Backplane profile in OpenVPX is fully defined and interlinked, the changes made for SpaceVPX require a SpaceVPX version of each of these profiles to be specified. Specifically, the slot profiles provide a physical mapping of data ports onto the slots backplane connector; which is agnostic to the type of protocol used to convey data from the slot to the backplane. The backplane profile is, obviously, a physical mapping of the backplane data transfer capabilities. It is slot dependent

and specifically defines a topology for channel number and buses for data interconnect. The module profiles are extensions of their accompanying slot profiles that allow for a mapping of protocols to each module port. Each module profile includes information on thermal, power, and mechanical requirements. Many are very close to OpenVPX and thus should enable use of OpenVPX modules and backplanes for prototyping or testing, but are different enough to require full specification. The section of the SpaceVPX standard that defines profiles was a significant effort and forms a majority of the completed standard.

A total of eleven 6U and ten 3U backplane profiles were defined to cover the spectrum of potential payload topologies expected in SpaceVPX usage. The 6U backplane profiles are summarized in the Appendix, Table I, while the 3U backplane profiles are shown in Table II. The first six cover switched Backplane profiles. The first three use a single fat pipe (4 lanes of data plane fabric) as the connection to the switch, and the next three use a double fat pipe (8 lanes of data plane fabric). These are followed by 5 mesh topologies, two each with 1 or 2 fat pipes between each peer and the last one with a special integrated power supply. Each of the groups of three or two are then separated by the location and grouping of the controller and data plane switches. The first of each switch topology group has an integrated data plane switch on the same card as the controller. The second has a separate controller but with connections to the data plane and the third of each group has a separate control plane without a data plane connection. The mesh topologies have a mesh profile with an integrated controller on a mesh peer and a separate profile with a controller without a mesh connection. The profiles are defined for the maximum number of slots possible for the data switch or controller implementation within the topology. Lower numbers of slots are possible by not implementing the additional nodes.

Since CompactPCI (cPCI) has been the backplane interface of choice for many spacecraft payloads in the past decade, special slot definitions were created for bridging SpaceVPX modules to heritage cPCI modules. The expansion plane may be used to bridge from a payload or controller module to other modules with cPCI or to a heritage CompactPCI module. This is also marked in Appendix Tables I and II.

Mechanical

SpaceVPX builds on the OpenVPX infrastructure but expands it for space modules, which are typically larger than commercial modules. 3U and 6U modules are defined just as in OpenVPX; however SpaceVPX defines 220, 280, and 340 heights in addition to the 160 mm one. Card pitches of 0.8, and 1.0 inches are kept from OpenVPX and 1.2 inch pitch is added. Modules are defined that use either levers or extraction tools for removal from backplanes and chassis. To keep all dimensions in one place, SpaceVPX includes all dimensions on mechanical drawings within the standard.

Keeping interoperability with OpenVPX has been a driver in how SpaceVPX has evolved. Defining needed changes to signals so they still operate with an OpenVPX board was a challenge in defining and refining the standard. Initial looks at existing commercial OpenVPX boards show it should be possible to mix OpenVPX boards with SpaceVPX board profiles. It is expected this will be one of the first areas proven out as SpaceVPX modules begin to be developed. The intent was to produce a specification, and a standard that allows for the use of OpenVPX boards for development work until such a time that any and all issues and concerns of function and capability are answered. With the addition of a number of SpaceVPX slot and module profiles that mimic OpenVPX slots and modules; it allows for an easier transition to those latter board types when migrating to an operational system.

Specification Overview

The SpaceVPX standard is made up of 21 sections. Section 1 includes the introduction, use cases, terminology, structure, naming and references. Section 2 defines standard compliance. Section 3 focuses on the utility planes including requirements for the dual redundant planes, system management protocols and SpaceUM module definition. Section 4 provides the mechanical definition. Section 5 defines the protocols to be used on the different planes. Sections 6, 10 and 14 define the slot profiles for 6U and 3U modules. Sections 7, 11, 15 and 18 define the backplane profiles. Sections 8, 12 and 16 define the module profiles and sections 9, 13, 17 and 19 define the chassis profiles. Section 20 defines the power reference and Section 21 defines 32-bit and 64 bit cPCI pin mappings.

After six months of studies and trades and four months of focused standard writing and development, an initial draft of the SpaceVPX standard was created in April 2013. At that point the VITA 78 study group became a working group and seven months later, the VITA 78 working group held a first working group ballot in late 2013. Two additional working group ballots have occurred and the standard passed a VITA-wide public review ballot in December 2014. SpaceVPX modules are expected to emerge from development starting in 2015.

5. RAPIDIO PROTOCOL OVERVIEW

The RapidIO (RIO) protocol was introduced as a high performance packet-based fabric technology standard in 2002 and is being evaluated by NASA for next generation avionics. Since its inception, millions of RIO-based devices have been shipped around the world from a large number of OEMs and silicon suppliers to meet the interconnect requirements in 3G/4G wireless base stations, video servers, real-time sensor/image processing and wafer inspection, military communications, embedded processing and high-performance computing and data analytics.⁶ Some of these

applications have benefited from the ultra-low-latency scalable fabric architecture, while others have taken advantage of the deterministic guarantee of delivery, low power implementation, fault tolerance and reliability features to cluster a large numbers of heterogeneous computing elements. RapidIO key features are summarized in Table 1 below.

Figure 3. RapidIO Protocol Layers

The RapidIO (RIO) protocol and packet formats are specified in a three-layer architectural hierarchy - Physical, Transport, and Logical, as shown in Figure 3. As opposed to other serial protocols, RIO is natively available in various system-on-a-chip (SoC) components and is implemented in hardware supporting simplified messaging and remote memory semantics. The protocol supports short, medium and long-reach links on or across boards over the backplane. The standard also supports both copper and optical links.

The Physical layer describes mechanisms for reliable packet exchange on a RIO link. The Transport layer defines a standard programming model and routing mechanisms for sending RIO packets through a RIO network. The Logical layer defines transaction types which hardware/software entities can exchange using the Physical and Transport layers. Each layer can evolve independently, in a backwards and forwards compatible manner.

The Physical layer uses standard XAUI 8B/10B encoding for Gen1 and Gen2 links, and Interlaken-like 64/67B encoding for 10 Gbps and higher speed links. A link consists of 1, 2, 4, 8 or 16 lanes, where each lane consists of two unidirectional transmit/receive pairs. The Physical layer defines three kinds of information to be exchanged on a RIO link: ordered sequences, control symbols, and packets. Ordered sequences exchange lane specific information. Control symbols are short (8 byte) quantities that are used for flow control, error management, time distribution, and reliable packet exchange. Packets consist of a 2-byte physical layer header, a transport layer header, logical layer header, and up to 256 bytes of data. Control symbols may be embedded within packets, giving RIO the lowest latency in-band control path of any protocol.

Table 1: Key RapidIO Protocol Features

Attributes	RapidIO Protocol Features
Latency	<ul style="list-style-type: none"> • Typical switching latency around 100 ns • Guaranteed lowest and predictable application-to-application latency with deterministic packet delivery • Control symbol provides lowest-latency system control and synchronization.
Power	<ul style="list-style-type: none"> • Typical power consumption of PCIe to RapidIO NIC ~2W • Typical power consumption of RapidIO Switch ~8W • Typical power consumption of SoC with RapidIO ~10W
Topology/Multi-Root System	<ul style="list-style-type: none"> • Supports any topology and multi-root system
Scalability	<ul style="list-style-type: none"> • With up to 32-bit device ID, the protocol has capability to support billions of devices
Priority and Reliability	<ul style="list-style-type: none"> • Maintains deterministic latency for high priority packet • Supports reliable and unreliable data transfer
QoS Flow Control	<ul style="list-style-type: none"> • Comprehensive QoS through flow control and congestion management: virtual channels, lowest latency control-symbol-based congestion management using VOQ-BP, tx- and rx-based flow control, data streaming extended header traffic management. • Maintains superior QoS through efficient packet header formats and short packets to avoid Head-of-line blocking impact of large packets.
Fault-tolerance/Reliability	<ul style="list-style-type: none"> • Rapid detection and isolation of fault conditions based on control symbols. • In-order reliable delivery in hardware – lowest software overhead.
Bandwidth	<ul style="list-style-type: none"> • Supports a range of bandwidth with low granularity to meet the target system performance and power requirements. • x1, x2, x4, x8, x16 lanes • 1.25G, 2.5G, 3.125G, 5G, 6.25G (Gen2), 10.3125G (Gen3), 25G (Gen4 under development) • Supports asymmetric links for power saving (x4 in one direction, x1 in reverse)
Mixed Traffic	<ul style="list-style-type: none"> • Supports various transaction sizes and types, allowing mixed traffic in the network. Possible to interrupt large data transfer with short control-symbol-based communication. Supports both lossless and lossy traffic using VCs.
Data Replication	<ul style="list-style-type: none"> • Supports uni-cast, multi-cast and broadcast through RapidIO switch fabric.
Security	<ul style="list-style-type: none"> • Possible to support security at the port as well as at the protocol level. • Possible to support application-to-application secured queue pairs.
Virtualization	<ul style="list-style-type: none"> • Hardware-based lightweight protocol stacks simplify I/O virtualization by allowing direct sharing of I/O resources from the application. Low-latency high-throughput network allows easier VM migration.

Each device in a RIO system is identified using a device ID. The Transport header of each packet consists of two device IDs: a destination ID and a source ID. The destination ID is used by RIO switch devices to route received packets to the correct output port(s). A standard routing programming model is used by network management software to determine how packets are routed through a RIO network. Response packets are formulated by swapping the destination and source IDs of received request packets.

The Logical layer supports a wide variety of transaction semantics, including:

- Maintenance: Register reads and writes, as well as fault event notification
- Read/Write/Atomic : 34, 50, and 66 bit address spaces for each RapidIO device
- Doorbells: 16 bit event
- Messaging: 4 KB message transfer
- Data Streaming: 64 KB message transfer with optional XON/XOFF, rate, and credit based flow control

While RIO is a strongly layered specification, there are two functions that are unified across all three layers. The first is Error Management, which defines an interoperable programming model and functionality for fault detection, reporting, and isolation mechanisms for the Physical, Transport, and Logical layers. The second is System Initialization, which defines interoperable mechanisms and fault tolerant algorithms for initializing a RIO system of unknown configuration.

6. AVIONICS PERFORMANCE ANALYSIS

Systems that support complex avionics and autonomous spacecraft require significant processing, storage and communication bandwidth. With the advent of Integrated Modular Architecture and resource sharing for multiple concurrent tasks, the introduction of multi-core processors and OpenVPX based systems interconnected by high-speed serial interfaces creates system complexity which makes the computation of data throughput and latency a challenge.

There are multiple ways to evaluate the performance of target hardware architecture, including analytical methods, physical tests and discrete simulation. The analytical method can provide a view of the worst case execution time and identify the key bottlenecks in a given architecture. Based on a test prototype, the system can be measured during operation using representative finite use cases and workloads. Discrete-event simulation can create a realistic model of the end-to-end system including the workload, use cases, Real-Time Operating System (RTOS) interactions, processing sub-system, storage, communication, FPGA clusters, and the interconnect network. The user can run

hundreds of scenarios to quickly identify the bottlenecks and the best performing areas.

Systems Performance Analysis

The conventional method for estimating avionics performance is derived from systems engineering and analysis. Using this approach, the basic performance of elements such as memory, storage, processing and interconnects is estimated using the known characteristics for that type of device and its implementation technology. In the case of avionics, these characteristics have been improving by orders of magnitude with each generation of computing technology.

For the purposes of this limited analysis, only memory, storage and interconnection performance are evaluated using first-order characteristics obtained from component literature. The system's analyst must identify the relationships between the components to correctly identify the limiting factors for performance. Errors in understanding the interactions in a complex system will lead directly to large errors in performance estimates; hence this method should be used for initial evaluation of a candidate architecture, rather than formal characterization. Nonetheless, this approach is useful, particularly for early trade studies.

The main memory data transfer rates are around 1600 Mega-Bytes per second (MBps) or 12.8 Gbps based on the use of a JEDEC DDR3-1600 memory array of 64-bit width.⁷ These DDR3 memory speeds range from 6.4 Gbps to 17 Gbps depending upon clock speed. For just about any processor or I/O operation, main memory is generally the major limiting factor for performance. Conventional processors are using 64-bit data paths, so these numbers are valid for a large number of diverse processing modules. Graphical Processing Units (GPUs) are the exception. In order to capture or render very high-resolution images at high frame rates, GPU designers typically increase the memory width to 128 and even 256 bits. This increases the memory throughput linearly to 25.6 Gbps and 51.2 Gbps respectively.

Using main memory data rates, the size and speed of the I/O interconnects can be estimated. The corresponding RIO Gen2 implementation for matching CPU memory data transfer rates would use four lanes at 5 Gbps, producing 20 Gbps of raw bandwidth, nicely meeting the 12.8 Gbps requirement once overhead is included. To handle the UHD camera input for the GPU-based processor, 16 lanes at 5 Gbps would be required, delivering raw performance of 80 Gbps, which is able to handle the 52 Gbps camera rate. Note that for Gen3 RIO at 10 Gbps, one would only require 8 lanes, which indicates that Gen3 would be useful today, if only to reduce the number of lanes needed.

Inter-processor data rate requirements between the various functional modules can only be determined by making assumptions regarding the type of processing performed by

the modules. We assume that the GPU-based image processing modules require the highest data rates; acquire instrument data at the full rate; and process the images in real-time to produce “objects” representing features in the images. These objects are passed to the host CPU from each image-processing module for incorporation into the decision-making autonomy algorithms. These objects can be targets on the planetary surface for landing or observation and we estimate the number of concurrent objects supported using the RIO-based inter-processor data paths. For the purposes of this analysis, we assume the use of four lanes at 5 Gbps for a total data rate of 20 Gbps for the inter-processor RIO bus. If one assumes an “object” can be described using 1KB, this results in an I/O limited capacity of around 20 million features per second for the Reference Architecture. Therefore, inter-processor data rates should not be a limiting factor, as the computation required to extract that many features is the dominant factor.

The other data rates involved in the Reference Architecture are the Solid-State Disk (SSD) drive Storage array and the Communications downlink. Communications rates at 1 Gbps are supported on optical links, so this rate represents the current performance limit. High-speed flash memory is able to store data at 10 MBps for a Class 10 high-speed device, which translates to 80 Mbps memory write speeds for a byte-wide array. In fact, just a casual look at the data rate numbers indicates that the SSD array should be much wider to increase its data rate. For example, if we assume that the downlink is driven from the Storage array, we would like the Storage subsystem to be able to provide close to 1 Gbps data rate to the downlink. This would require an increase of a factor of twelve in speed. Moving to a 64-bit flash memory array would increase the data rate by a factor of 8 to 640 Mbps, which starts to match the downlink data rate of 1 Gbps. Flash array read operations tend to peak at 250 Mbps.⁸ This is a key insight provided by conventional analysis – that the flash array is the major limiting factor for overall performance, and moving to high-speed flash memory arrays will be required.

Performance modeling

We have modeled this system using a commercial system-level simulation package called VisualSim Architect from Mirabilis Design Inc.⁹ The simulation method used is discrete-event simulation with Monte Carlo variations. We modeled the system using the standard libraries in VisualSim Architect for the processors, DMA, memory, communication, OpenVPX and RapidIO functions. The libraries are derived from manufacturer’s datasheets. The system is setup with parameters for the common attributes including the speeds, traffic rates, processing times, packet sizes and the RapidIO network attributes including the switch speed and number of lanes. We executed the simulation for a variety of rates, RIO speeds and lane configurations: speeds of 2 and 5 Gbps were used at four lanes and 1 Gbps was used with four lanes and six lanes. The simulation time selection is an important part of the modeling and is a function of the number of transactions

that are required to reach steady state. Steady state for this model would be the time beyond which buffer occupancy does not increase and based on our experiments, we determined that 50 μ s is a suitable duration for the simulation.

The model created using VisualSim Architect is shown in Figure 4. The system model consists of a FPGA-based processor, GPU-based processor, solid-state flash array for storage and a communication sub-system for uplink and downlink. The flash array modeled is a high-speed parallel type, representing a Storage subsystem suitable for VPX systems. These sub-systems have a OpenVPX interface to the RIO network. The OpenVPX systems are interconnected using RIO in a double-redundant configuration with each sub-system connected to two switches in parallel. The RIO fabric is 5 Gbps with 4 lanes providing a 20 Gbps link for each of the processing systems. The FPGA processor generates read and write requests every 249 ns. The spectrometer image frames are full HD at 20 frames per second; the GPU cluster is processing Ultra HD at 4K MPEG4 at 100 frames/second, as per the Reference Architecture. The simulated use case is for the FPGA processor, GPU processor and the Communications (Comm) subsystems to send read/write requests to the Storage module. This would represent sending the processed image data and objects to the Storage array for subsequent downlink via the Comm subsystem. The Storage array modeled is a high-performance design with a RIO interface based on a real product design. Our goal is to get maximum efficiency out of the Storage subsystem and to size the RIO network for this system design.

The throughput analysis is from simultaneous GPU, FPGA and Comms systems interactions with the Storage array. The throughput graphs in Figure 5. shows the simulation time on the x-axis (10E-5 seconds) and the RIO transaction rate per second on the y-axis. The left graph shows that the GPU board is able to conduct 12 million transactions per second. Now, the simulation varies the data packet size between 16 to 64 bytes, so that converting this rate into actual data throughput rate requires estimation. Using an average of 40 bytes per transaction, the data throughput is 480 MBps, or 3.84 Gbps. The FPGA transaction rate shown on the right graph is 6 million transactions per second, which results in an estimated throughput of 240 MBps or 1.920 Gbps. Since the simulation is driving all the modules simultaneously, the net throughput rate into Storage is the addition of the two, or 5.76 Gbps. Therefore, these interactions appear to be limited by the speed of the flash memory Storage array, which has roughly half the performance of a DDR3 memory array.

Figure 4. VisualSim Architect Model

Figure 5. Model Throughput Graphs

The latency plot in Figure 6 has the simulation time on the x-axis and the end-to-end response time or latency on the y-axis. End-to-end response time or latency is measured from the completion of the processing request to the storage device and the response, either the data for a read or an acknowledgement for the write operation. The Storage latency shown is the latency for the Read request at the Storage device. This is the latency per transaction, and each transaction transfers multiple bytes. Therefore latency per byte is about a factor of 40 lower than the transaction latency or about 0.5 ns. It is very difficult to obtain such multiple interaction analysis using any method other than discrete-event simulation, and the graph shows the complexity of the resulting transactions to the Storage array.

Figure 6. Model Latency Graph

Figure 7. Hardware Throughput Measurement Diagram

Hardware Prototype Performance

The main objective of our study is to validate the performance, scalability and reliability for a fault-tolerant space interconnect fabric that may be used in future space vehicles. This third method involves measurements made on prototype hardware. The prototype hardware is shown in Figure 7 consisting of 4 lanes of 5 Gbps RIO using the IDT TS721 PCIe to RIO NIC connected using the IDT CPS-1848 RIO switch. The test set was powered using two Xeon processors running the Goodput utility for determining net throughput and latency statistics. Different packet sizes for data transfer were used to determine throughput at each level. The concurrent CPU utilization for DMA transfers was measured. The graph shows that the peak payload rate available to the user and occurred using 64 KB data set size at 1,686 MBps or 13.49 Gbps. The corresponding link rate at that packet size is 14.46 Gbps without encoding overhead (~18.07 Gbps with encoding on the wire) using an open-source software driver without any optimization. Additional tests suggest the CPU overhead ranged from 20% to under 1%.

Figure 8. RapidIO Throughput

There are two additional modes in RIO: Direct Mode and Message Passing Mode. Direct Mode allows different RIO endpoints to see each other's memory and allows remote access to the memory. Typical usage of Direct I/O mode will be when a CPU core wants to access a remote node's memory. This is achieved by having each node provide an Inbound Window and allowing remote nodes to read or write to it. Therefore, the access size is restricted to 1/2/4 and 8 bytes for a 64-bit wide array. In this study, we determine how long it takes to write data from a Master Node to a Slave Node's memory, and read data from a Slave Node's memory. The latency measurements cover various CPU write/read access sizes (from 1 byte to 8 bytes) to insure that the latency is pretty much constant and very low. These direct I/O mode operations can be used to exchange small amounts of data between RIO nodes without having to use DMA operations, which are meant to be more efficient at high throughput to move larger amounts of data. The

numbers for latency measured are 1.8 μ sec for read operations and 1.1 μ sec for write operations and these numbers remain constant for transfer sizes of 1, 2, 4 and 8 bytes.

Figure 9. RapidIO CPU Utilization

As mentioned above, another mode is Message Passing Mode. This mode can be used when multiple nodes are not aware of each other's memory. In this case, each node can communicate and pass data to each other by using the Message Passing Mode interface, which consists of transmitting data messages through the mailbox mechanism defined by the RIO specification. A message can consist of up to 16 packets of up to 256 bytes each, for a maximum of 4KB per message. This Message Passing Mode is not meant to be used for the main high performance data path, but can be used for synchronization/control between RIO nodes.

7. REFERENCE ARCHITECTURE ANALYSIS

Our Reference Architecture can be characterized using the numbers from each of these different methods of estimating subsystem I/O performance. This analysis shows throughput issues in several areas requiring proper design of the RIO interconnect to support the processing throughput requirements. For this analysis, we assume that the RIO-based inter-processor switched data bus is composed of four lanes at 5 Gbps delivering a raw data rate of 20 Gbps to each slot. By standardizing the data rate to each co-processor, we can perform a simple comparison of the utilization required by each module in the Reference Architecture.

The UHD camera represents the module driving Reference Architecture I/O throughput requirements. For example, a total of 51 Gbps would be needed to support the camera input on a separate camera interface, not part of the inter-processor fabric. If implemented as RIO Gen2 5 Gbps interfaces, 16 lanes would be required, the maximum number specified in the standard. However, memory data transfer rates are around 12.8 Gbps to a 64-bit wide DDR3 memory array, which helps determine the breakout of high-speed camera RIO data paths into the multiple GPU-based processors. Since the camera output rate exceeds the rate at

which a given GPU processor can receive such data into its main memory, the camera data has to be broken out on the RIO camera interface into multiple GPU processor modules. Using this approach, it will take at least four GPU processors to handle the UHD camera data rates using a 64-bit wide DDR3 memory array on the GPU processor. The UHD camera would use 16 lanes (5 Gbps) of RIO, broken out into four sets of 4 lanes, each driving a separate GPU processor card to acquire the camera data. The use of four parallel GPUs for handling the UHD camera is shown in Figure 10.

The output side of the GPU modules connects to the inter-processor RIO switched bus. The GPU processors take a portion of the UHD camera image stream and reduce the data rate, either using image compression or by extracting features. We assume that this reduces the camera data rate by a factor of 6 to an output rate of 2 Gbps from each GPU module to the RIO inter-processors bus, a reasonable number for GPU-based processing.

The spectrometer interface requires about 3.84 Gbps, so could be implemented as a single lane 5 Gbps interface between the instrument and the FPGA processor. Therefore, the spectrometer can be implemented in a more conventional manner, using a single FPGA-based processor card and RIO bus interface. If we assume that the full input rate is reflected to the output of the FPGA processor, then up to 3.8 Gbps of data is provided to the inter-processor RIO bus. These throughput numbers are based on instrument definitions and processing assumptions.

The other data rates involved in the Reference Architecture are the SSD Storage array and the Comms downlink. Paralleling and interleaving flash memory chips using custom controllers and interfaces can greatly increase their performance. The performance modeling shows that such a high-performance SSD Storage array could support up to 5.7 Gbps rates. The data rate required for the communications downlink is 1 Gbps based on similarity to demonstrated technology. For this function, we are assuming that the data for downlink is cached within the Storage array.

The host processor is the main data transfer engine on the RIO inter-processor bus, using either DMA or CPU-driven transfer modes. Note that our prototype hardware measurements characterize 64-bit processors using 64-bit DDR3 main memory, so that the measured 13.5 Gbps maximum transfer rate can be used as the figure of merit for the integrated analysis.

Figure 10. Reference Architecture RIO Configuration

Resultant inter-processor RIO bus data rates for the integrated Reference Architecture can be determined by making assumptions regarding data flow in the system. We defined instrument data rates based on high-performance hardware; we defined the inter-processor interconnect as consisting of four 5 Gbps RIO interfaces based on current hardware prototypes; we made reasonable assumptions regarding the reduction of data rates achieved in the GPU and FPGA co-processors; we used the data rates from modeling to characterize a high-speed SSD Storage array. The purpose of characterizing these subsystems is to produce an integrated analysis of RIO inter-processor throughput rates and limitations.

Table 2 shows the quantitative results of the Reference Architecture Inter-processor RIO Bus throughput analysis. The key metric is the percentage utilization of the inter-processor RIO bus by each subsystem. The four GPUs provide 40 percent, the FPGA provides 20 percent and the Storage and Comms subsystems provide 28 and 5 percent of the data rate and bus utilization. This results in a total of 93 percent maximum utilization of the inter-processor RIO bus as shown on the Total Source sum line. The Host computer is assumed to be the destination for all the source data transfers. Since the host processor can only handle a maximum of 67 percent utilization, (13.5 Gbps) there is oversubscription of the Host by about 25%, as shown in the Throughput Deficit line. This initial analysis shows that the system will not be able to run at real-time rates if all data paths are active at the same time, resulting in data pipeline congestion, which will affect the capability of the payload computer to handle the full instrument data rates.

Table 2: RIO Bus Utilization

Subsystem	Data Rate Gbps	Capacity Gbps	Utilization Percent
GPU processor 1	2.0	20.0	10.0
GPU processor 2	2.0	20.0	10.0
GPU processor 3	2.0	20.0	10.0
GPU processor 4	2.0	20.0	10.0
FPGA processor	3.8	20.0	19.0
Comms	1.0	20.0	5.0
Storage - 64-bit	5.7	20.0	28.5
Total Source	18.5	140.0	92.5
Host computer	13.5	20.0	67.5
Throughput Deficit	5.0	120.0	25.0
Interprocessor RIO	18.5	20.0	92.5

However, the Reference Architecture use case can be modified to allow it to meet the mission requirements. We define two different modes of operation: a real-time instrument mode where all the subsystems except communications and storage are active, for example during an EDL phase of flight. After this critical phase, the data cached in main memory can be transferred to Storage and

downlinked with the Comms function. Using this assumption, the Reference Architecture can reasonably be expected to deliver the required performance based on the RIO throughput analysis, with the numbers derived from the various methods of performance estimation presented.

This analysis shows the reason why our Reference Architecture needs to keep all real-time data in its multiple main memory arrays – the 12.8 Gbps rate of these memory arrays is the highest data rate supported by the processing hardware. That is, all high-speed transfers are generally from DDR3 memory to other DDR3 memory. Going to other subsystems can extract a high price in terms of throughput or latency. Note that for the purposes of this paper, memory rates used are typical of commercial terrestrial computers. For radiation tolerance, memory speeds are often de-rated, particularly when error correction is used. This may be true for the RIO and the processor speeds as well. Therefore, certain scaling factors will have to be applied for space computers with radiation tolerance. Processor contention for main memory access is another factor that has been ignored to simplify throughput analysis.

OpenVPX Implementation

An example of the Reference Architecture implemented as an OpenVPX system is shown in Figure 11. This is a single-string configuration using RIO as its primary Data Plane interconnects. While SpaceWire is shown in the diagram as the Control Plane, it is used for subsystem control and spacecraft interface functions and was not considered in the analysis. A single RIO data switch slot and one Utility Module Controller provide the switch and management infrastructure. Power is switched by the Power Switch module from the power source and then individually routed to each slot. Similarly the management functions are controlled by the Utility Control Module and routed to all other modules.

As a result of this analysis, the number of RIO lanes required for supporting each functional block has been determined and diagramed. Four GPU-based image-processing modules are required for the high-speed camera and only one FPGA-based processor is required for the spectrometer data. The high-speed portion of the data transfer and processing involves only the use of the RIO Bus interconnects and the high-speed memory arrays. Storage is used for communication and spacecraft interface functions, but is too slow for the high-speed real-time processing required for real-time approach and landing functions. Instead these functions are active only during the portions of the mission where the real-time autonomous navigation functions are idle.

8. CONCLUSIONS

The overview of SpaceVPX standard development shows how currently available components can be configured into an OpenVPX implementation that is reasonably similar to that anticipated using SpaceVPX compliant products.

Products using RapidIO can achieve throughput in excess of 80 Gbps using current generation components. Overall system throughput is currently limited by memory data rates – to either main DDR3 memory or to flash-based non-volatile storage. Both simulations and measurements of prototype hardware support these performance numbers using commercial components.

The performance analysis applied to the Reference Architecture produced one key insight. At least four image processing cards would be required to handle the camera. Inter-processor data rates can be supported provided each processing card handles its part of the image field independently. If not, additional traffic on the Inter-processor fabric will occur. Another insight is the need to increase the speed of the Storage array by a factor of 12 or more over a conventional SSD implementation. Finally, storage and communication rates are significant enough, so as to preclude using these subsystems during active autonomous navigation using the high-resolution instruments. This is due to the lack of sufficient margin in the RIO-based inter-processor interconnect of the Reference Architecture.

Major improvements can be made even using current technology. The GPU-based processors can be upgraded to using 128 and even 256 bit wide main memory arrays, with a resultant increase in data rates of a factor of two or four. Note that such an improvement is not available using general-purpose host processors, which only support a 64-bit wide path to main memory. Improvements in host processing can only be achieved by using multiple host processing modules, each with its own main memory array and may be required. Major improvements in SSD Storage array response time and throughput are needed, to at least match the downlink rate. Finally, moving to next-generation RIO and more lanes will be required to support the scale-up of co-processing needed for deep space missions.

The actual requirements for such autonomous missions remain to be defined, but this architecture trade study shows what can be supported using current technology and helps lead the way to increased performance from future space avionics systems.

Figure 11. Reference Architecture OpenVPX Implementation

APPENDIX

Table I. SpaceVPX 6U Profiles

Topology Configuration	Fat Pipes per Connection	Payload Fat Pipes	Data Plane Switch or Mesh Size	Control Plane Switch Size	Control & Mgmt Plane Hubs Integrated with Data Plane Switch	Control & Mgmt Plane Hubs with Data Plane Connection	Control & Mgmt Plane Hubs wo/ Data Plane Connection	Payload Slots	Data Switch Slots	Controller Slots	SpaceUM Slots (Supports 8 slots)	Payload Supports J2 Expansion Plane	PCI Positions on J5 (P=Payload C= Controller)	Maximum Total Slots
Switch 1	1	2	12S	16	x			12	2		2	X	P	16
Switch 2	1	2	22S	26		x		20	2	2	3	X	P/C	27
Switch 3	1	2	22S	28			x	22	2	2	4	X	P/C	30
Switch 4	2	4	6D	10	x			6	2		1	X	P	9
Switch 5	2	4	11D	15		x		9	2	2	2	X	P/C	15
Switch 6	2	4	11D	17			x	11	2	2	2	X	P/C	17
Mesh 1	1	12	13S	15		x		11		2	2	X		15
Mesh 2	1	19	20S	24			x	20		2	3		C	25
Mesh 3	2	12	7D	9		x		5		2	1	X		8
Mesh 4	2	18	10D	14			x	10		2	2	X	C	14
Mesh 5	4	20	6Q	10			x	6		2	1		C	9

Table II. SpaceVPX 3U Profiles

Topology Configuration	Fat Pipes per Connection	Payload Fat Pipes	Data Plane Switch or Mesh Size	Control Plane Switch Size	Control & Mgmt Plane Hubs Integrated with Data Plane Switch	Control & Mgmt Plane Hubs with Data Plane Connection	Control & Mgmt Plane Hubs wo/ Data Plane Connection	Payload Slots	Data Switch Slots	Controller Slots	SpaceUM Slots (Supports 2 slots)	Payload Supports J2 Expansion Plane on J2	PCI Bridge Positions on J2 (P=Payload C= Controller)	Maximum Total Slots
Switch 1	1	2	3	5	x			3	2		3	16/32	P	8
Switch 2	1	2	6	8		x		4	2	2	4	16/32	P	12
Switch 3	1	2	7	9			x	7	2	2	6	16/32	P	17
Switch 4	2	4	2	4	x			2	2		2	8/16		6
Switch 5	2	4	3	5		x		1	2	2	3	8/16		8
Switch 6	2	4	3	7			x	3	2	2	4	8/16		11
Mesh 1	1	4	5	5		x		3		2	3	8/16		8
Mesh 2	1	7	8	10			x	8		2	5			15
Mesh 3	2	4	3	3		x		1		2	2	8/16		5
Mesh 4	2	6	4	6			x	4		2	3			9

ACKNOWLEDGEMENTS

The authors thank Scott Goedeke and Bryan Silbermann for their comments and review of this paper.

REFERENCES

- [1] *NASA Fact Sheet: Lunar Laser Communication Demonstration - NASA's First Space Laser Communication System Demonstration*, (PDF). NASA.gov. August 2013.
- [2] *SpaceVPX Systems ANSI/VITA 78.00*, www.vita.org, 2015
- [3] *OpenVPX Architectural Framework for VPX*, ANSI/VITA 46.00, www.vita.org, 2012
- [4] *VPX REDI: Mechanical Specifications for Microcomputers Using Conduction Cooling Applied to VPX*, ANSI/VITA 48.2, 2010.
- [5] *VPX: System Management*, ANSI/VITA 46.11, 2015
- [6] *RapidIO Interconnect Specification 3.1*, www.rapidio.org, September 2014
- [7] *JEDEC JESD21-C, JEDEC Configurations for Solid State Memories*, www.jedec.org, 2015.
- [8] *Long-term performance analysis of Intel Mainstream SSDs, PC Perspective*. 2009-02-13.
- [9] *Mirabilis Design web site*, <http://mirabilisdesign.com/>

BIOGRAPHY

Richard L. Alena is a Computer Engineer in the Intelligent Systems Division at NASA Ames. He was a Co-investigator for Wireless Sensor Networks aboard the SOAREX-8 mission in 2015. Mr. Alena worked on the Ground Data System and performed Communications Analysis during operations for the LCROSS Lunar Mission and on avionics and software architectures for Lunar Surface Systems for human missions. He was the chief architect of a flight experiment conducted aboard Shuttle and Mir using laptop computers, personal digital assistants and servers in a wireless network for the ISS. He was group lead for Intelligent Mobile Technologies, developing planetary exploration systems for field simulations. Mr. Alena holds an M.S. in Electrical Engineering and Computer Science from the University of California, Berkeley. He is the winner of the Ames Honor Award for Engineering in 2010, NASA Silver Snoopy Award, numerous NASA Group Achievement Awards and a Space Flight Awareness Award.

Charles Patrick Collier is a Senior Electrical Research Engineer with the Air Force Research Laboratory Space Vehicles Directorate. He was the Fiber Optic Tech Lead for a Multi-Access Laser Communication within TSAT. He is the Technical Advisor for the Space Communications Program with a primary focus on satellite based networks, photonics, and free space laser communications. Patrick is currently the Program Manager/Chair for the VITA 78 (SpaceVPX), VITA 78.1 (SpaceVPX Lite), and the RapidIO Space Device Class efforts. He is also the current Technical Lead for the System Open System Architecture.

Mohammad Akhter is Director of Architecture with the Interface and Connectivity Division of Integrated Device Technology (IDT) Corp. He has more than 17 years of experience in mobile and satellite communications, as a research engineer, air-interface and modem developer, and system architect covering satellite systems (Intelsat, Inmarsat, ICO, DVB-S/S2), computing architectures, and wireless access networks (LTE/LTE-A/C-RAN). He is participating in the ETSI edge computing and RapidIO coherent scale-out working group to realize the vision of next-generation low-latency access network through the convergence of high performance computing and data-path architecture. While at IDT, Mohammad has played a key role in developing the RapidIO fabric and compression architectures and the cloud based heterogeneous system for highly scalable deep learning analytics framework as part of the IDT open HPAC Lab. He is currently a member of the IEEE.

Barry Wood is a Principal Product Applications Engineer with the Interface and Connectivity Division of IDT. He worked in Nortel Networks Digital Multiplex Switch (DMS) product line for 10 years, developing firmware and software for fault tolerant compute platforms. While at IDT, he led the Systems Engineering software team, held the post of Architect for Tundra's Gen1 RapidIO switches, and eventually moved to the Applications Engineering team. Barry has also contributed to many areas of the RapidIO specification, including the 3.1 revision, which contains features developed by the RapidIO NGSIS Task Group for space and terrestrial applications. Barry is currently the Chair of the RapidIO Technical Working Group, and Chair of the Software Task Group.

Soumik Sinharoy is a Senior Product Manager at Orange Silicon Valley, an innovation subsidiary of telecommunications giant Orange. He has over 15 years of IT experience and more than a decade of experience in the telecom industry. He was the co-lead for real field trial of world's first 40 Gbps global range InfiniBand with U.S. Department of Energy. He was instrumental in evangelizing that GPU accelerated analytics as the next evolution towards enterprise databases. He currently pursues applied research in the field of High Performance Computing, High Performance Edge Analytics and applied Artificial Intelligence for Unmanned Systems and future Space Vehicles. He also evangelizes the vision of future of supercomputing via various real life demonstrations at conferences in

collaboration with industry partners. Mr. Sinharoy holds a degree in Computer Science and Engineering from University of South Carolina.

Deepak Shankar is the Founder and CEO of Mirabilis Design Inc, a systems engineering software provider. Mr. Shankar has over 20 years of experience in advanced system architecture analysis of electronics systems, ranging from semiconductors to avionics systems. Deepak Shankar has presented at many international conferences, IEEE events and has been a member of the AUTOSAR, RapidIO and Docsis consortium. Prior to Mirabilis Design, Mr. Shankar was VP of Business Development at MemCall, a fabless semiconductor company and SpinCircuit, a supply chain joint venture of HP, Cadence and Flextronics. Prior to that, Deepak spent many years in product marketing at Cadence Design Systems. Mr. Shankar has an MBA from UC Berkeley, MS from Clemson University and BS from Coimbatore Institute of Technology, both in Electronics and Communication.

1
2
3
4
5
6
7
8
9