Appliqué Sensor Interface Module: An Enabling Technology for Space Plug-and-Play Systems

Jeffrey Scott
Space Electronics Branch, Air Force Research Laboratory
3550 Aberdeen Ave. SE., Kirtland AFB, NM, 87117; (505) 846-6280
jeffrey.scott@kirtland.af.mil

James Lyke
Space Electronics Branch, Air Force Research Laboratory
3550 Aberdeen Ave. SE., Kirtland AFB, NM, 87117; (505) 846-5812
james.lyke@kirtland.af.mil

Patrick McGuirk
Micro-RDC
8102 Menaul Blvd NE. Albuquerque, NM, 87110; (505) 294-1962
patrick.mcguirk@micro-rdc.com

Mark Shaw
Data Design Corporation
7851-A Beechcraft Ave., Gaithersburg, MD, 20879; (301) 670-1157
mshaw@datadesigncorp.net

Don Fronterhouse
Scientific Simulations Incorporated
2951 Marina Bay Dr. #130-306., League City, TX, 77573; (505) 846-0224
don@ssi-sw.com

ABSTRACT

One approach to addressing the aggressive demands of the Operationally Responsive Space mission has been the development of the Space Plug-and-Play Avionics (SPA) architecture. The SPA architecture enables the rapid development of space systems through the assembly of self-describing components. The automation inherent in the SPA concept makes it possible to build systems in a fraction of the time by reducing human-induced interface errors, one of the key factors resulting in costly developmental delays. One of the key enablers of this paradigm shift is the Appliqué Sensor Interface Module (ASIM). Just as a USB interface chip makes it possible to add modular "plug-and-play" (PnP) components to a personal computer, the ASIM makes it possible to add modular PnP components (from thermometers to cameras and payload elements) to a responsive spacecraft. Two generations of ASIMs have been developed and demonstrated in the Responsive Space Testbed. The commercially available version supports hardware and software features such as a self-contained microprocessor, embedded electronic datasheet, a simple application programming interface, and a novel test bypass facility to simplify the testing of SPA-enabled components. The ASIMs are planned for use in several upcoming sounding rocket and space experiments. This paper will explain the role of the ASIM in the SPA architecture, the current status of the ASIM design, and the roadmap of future ASIM developments.

INTRODUCTION

ACHEIVING Operationally Responsive Space (ORS) is a necessary step in mitigating the risks inherent in the traditional space model. The United States is one of the nations most dependent on space and yet has little capability to responsively augment existing assets or reconstitute lost or damaged assets. ORS attempts to address these problems by adding the capabilities currently lacking in the nation's space portfolio. The objectives of ORS include the capability to responsively reconstitute lost capabilities, augment or surge existing capabilities, fill existing gaps in capabilities, exploit new technical or operational innovations, respond to unforeseen or episodic events, and to enhance survivability and deterrence[1]. These capabilities are becoming increasingly important but the

demands of ORS are not trivial. Meeting the objectives of ORS requires a new way of thinking and addressing the issues faced by the space community.

The Space Plug-and-Play Avionics (SPA) architecture is one approach currently being developed to meet the aggressive demands of ORS. It involves four key concepts which will be described in detail: encapsulation of complexity, self-describing networks, machine-negotiated interfaces, and the test bypass interface. Every one of these concepts benefits from the Appliqué Sensor Interface Module (ASIM). The ASIM is an embedded microcontroller device that can either be designed into a new SPA component or "stuck" on the end of a legacy component to make it SPA compatible.

BACKGROUND

Before delving too deeply into a discussion on the ASIM, it would be helpful to gain some background knowledge on the ORS initiative in general and SPA in particular. Following is a discussion of the origins and evolution of ORS as well as a general overview of SPA.

Emergence of ORS

The origins of the Operational Responsive Space initiative can be traced back to early 2001 when the National Security Space Commission identified critical weaknesses in America's space capabilities. The Commission reported that the nation's current space systems are vulnerable to a range of attacks that could seriously degrade its space capability[2]. Although space supremacy has been an asymmetric advantage for the US in the past, the nation's dependence on space could also be leveraged by an enemy with catastrophic results. The US is more dependent than any other nation on its commercial, military, and intelligence space assets. The political, economic, and military value of these assets make them inviting targets for those hostile to the US[2].

Later in 2001 the Air Force Space Command (AFSPC) drafted a Mission Need Statement for ORS outlining the requirements for a more responsive spacelift capability. These requirements were stated as follows:

> (1) On-demand satellite deployment to augment and quickly replenish constellations to support crises and combat operations; (2) Launch to sustain required constellations for peacetime operations; (3) Recoverable, rapid-response transport to, through, and from space; and (4) Integrated space operations mission planning to provide near real-time automated planning to enable on-demand execution of space operations[3].

The 2001 Space commission findings and the definition of requirements by AFSPC generated discussions about how to meet the challenges of ORS. The Office of Force Transformation (OFT) developed and adopted a new business model based on ORS and the Air Force executed at least two studies on the topic.

The business model adopted by OFT was meant to address current trends in the acquisition, development, fielding and utilization of America's space capabilities and to ensure the nation's space superiority for the future. Some of these "ominous trends" as described by the late Vice Admiral Arthur K. Cebrowski, director of the Office of Force Transformation included: "falling barriers to competitive entry into the 'commons' of space, an increasing dependency on space capabilities, and emerging vulnerabilities in current space systems[4]." Cebrowski was also concerned about the issue of operationalizing national space utilities and advocated a system where demand was driven by operational- and tactical-level commanders and military capabilities were designed directly for the commander. He envisioned a small, low-cost, sub-optimized satellite designed for a single tactical or operational mission. The TacSat series of experimental satellites was instituted to make this vision a reality.

One of the first studies performed to examine the feasibility and benefits of ORS was an AFSPC Analysis of Alternatives (AoA) for Operationally Responsive Spacelift. The study concluded that ORS could provide significant military utility at the campaign level through the use of responsive space-asset delivery[5]. It also suggested that modularity may be a factor in achieving ORS[6].

The Air Force Research Lab (AFRL) performed its own Responsive Space Advanced Technology Study (RSATS) in 2004 to determine what type of technologies might help achieve ORS. The study revealed that Plug-and-Play technologies similar to those used in the commercial electronics industry could be used to help achieve ORS capabilities such as the rapid reconstitution and augmentation of existing space assets[7]. The study featured recently completed work based on an AFRL proposal to develop the Adaptive Avionics Experiment (AAE), which embraced many of the key principles behind a modern plug-and-play (PnP) approach for aerospace. The AAE focused on avionics as the area most readily transformed into a PnP system, with the following four elements as crucial: appliqué sensor network, adaptive wiring manifold, high-performance computing on-orbit, and software definable radio[8]. The appliqué sensor network, proving to be the most useful of the four was expanded and refined to become the current SPA architecture[8].

SPA overview

The collection of concepts developed by AFRL to realize PnP space systems is collectively termed Space Plug-and-Play Avionics (SPA). These concepts include self-forming networks, machine-negotiated interfaces, encapsulation of complexity, and test bypass.

Encapsulation

The most fundamental concept in the SPA paradigm is that of encapsulation—hiding complexity within modular building blocks in order to simplify design. In SPA, this concept manifests itself both in the design of hardware and software. In hardware, the complex inner workings of the device are hidden from the rest of the system. Only single-point electrical connections consisting of data, power, and time synchronization are used to connect the device to the SPA network. Software encapsulation occurs at many levels, but the greatest example is in the use of XML-based or eXtended Transducer Electronic DataSheets (xTEDS) to precisely define the interfaces between components and even "pieces of software." The goal of this architecture is the achievement of "pure" or "glueless" hardware and software modularity. "Gluelessness" is a very constrained form of modularity that allows rapid integration to occur[9]. Instead of requiring custom electronics or software (the glue) to interface one modular block with another, each block contains everything it needs to maintain compatibility with other blocks in the system.

Self-forming Networks

The second important SPA concept is that of self-forming networks. In SPA, every device is considered an endpoint on the network, including both traditional bus components, such as reaction wheels or torque rods, and payload components, such as imaging devices. In fact, even structures are endpoints and can be treated in the same manner as other SPA devices on the network. For example, a spacecraft structural panel may contain its own harnessing and internal routers and hubs—essentially an entire SPA sub-network in itself, but the panel is also an endpoint and can be treated as such in the larger SPA network that is the PnP spacecraft. The result is a collection of endpoints separated by hubs or routers and arranged in any order or configuration. One could take a number of the panels just described and connect them in a box or arbitrary shape to form a spacecraft bus. The SPA network is created dynamically as devices are introduced. Figure 1 shows how any SPA device (triangular block in Figure 1-a) can become an endpoint on the network in any available location (Figure 1-b).

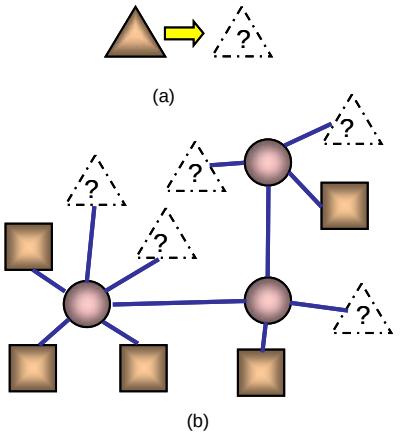

Figure 1. SPA network structure.

Machine-negotiated Interfaces

Glueless modularity and self-describing networks are achieved in the SPA architecture through the use of the third SPA concept—machine-negotiated interfaces. SPA interfaces are defined by components in their resident xTEDS and managed by the Satellite Data Model (SDM)[10]. The xTEDS contains descriptions of all commands accepted, variables produced, and data messages that can be delivered by the device[9]. It fully describes the services and data provided by the device and represents the protocol for accessing these services and data. SDM is a type of "middleware" that manages the SPA distributed network and makes it possible for applications and components to share data and services without needing to know addresses or specific messaging structures. It consists of five functional managers to accomplish this:

- *Processor Manager (PM)*. Resident on each processor, this manager is charged with the task of keeping its processor busy by executing (and terminating) requested tasks.
- *Data Manager (DM)*. This manager keeps track of all available resources (data, commands, and services) using a Data List and Message List. The lists are updated as processes (either applications or devices) are added or terminated.
- *Task Manager (TM)*. The TM manages all active and pending tasks.
- *Sensor Manager (SM)*. Each SM is responsible for interfacing between a specific data network and the SDM processing network. There can be as many Sensor Managers as is necessary for a system.

- *Network Manager (NM)*. This Manager keeps track of the locations of all devices on the network. It manages a routing table making it possible for any device to send data to any other device without having to know its address or physical location.

When a SPA device is connected to a SPA network, the device is automatically detected and enumerated via the SPA-*x* interconnection protocol and the SM requests the device's xTEDS. The SM interprets the xTEDS and registers all device capabilities with the DM.

Test Bypass

While not necessary for a functioning PnP system, the Test Bypass Interface (TBI) adds rapid test capability to SPA, and is a crucial piece in helping to achieve ORS. The TBI allows the introduction of a Hardware in-the-loop Simulator (HWILS) into the system. A HWILS reproduces the particular phenomenology of actuators and sensors and feeds it in a controlled manner into the system. The simulated data are injected directly at the location of the component via the TBI. The signals and data produced by the SPA device are overridden by the test bypass engine. The simulated data appear to the rest of the system as *in situ* data. Test Bypass makes it possible to perform real-time, day-in-the-life tests of the entire system in as unintrusive a manner as possible.

ROLE OF ASIM IN THE SPA ARCHITECTURE

One of the challenges faced by the commercial computer electronics industry in designing PnP devices was the sheer complexity of the interfaces. The USB 2.0 specification, for example is a 650 page document[12] with numerous supplements. Implementing from scratch the necessary electronics and software to support this interface for every new device or device type would be a daunting task. In the commercial marketplace of the PC industry, this problem was solved through the emergence of third-party interface chips and intellectual property (IP) blocks that implemented the USB standard. The complexity of the PnP interface is then encapsulated in a simple logic block that is combined with the rest of the device design be it a USB mouse or keyboard, etc. to make a distinct PnP component. In this manner, the component manufacturer can focus on his/her specific area of expertise (i.e. designing a mouse) rather than laboriously (and unnecessarily) reengineering the interface.

In the SPA architecture, a similar approach is taken to encapsulate the concepts particular to SPA into a single component that can be combined with another non-SPA device to make it Space Plug-and-Play compatible. This component, called an Appliqué Sensor Interface Module (ASIM), not only acts as a SPA-*x* interface chip, but also includes other SPA-enabling features such as xTEDS, power management, time synchronization and Test Bypass. Figure 2 shows the block diagram of a generic ASIM. The following sections describe the role an ASIM plays in the SPA architecture.

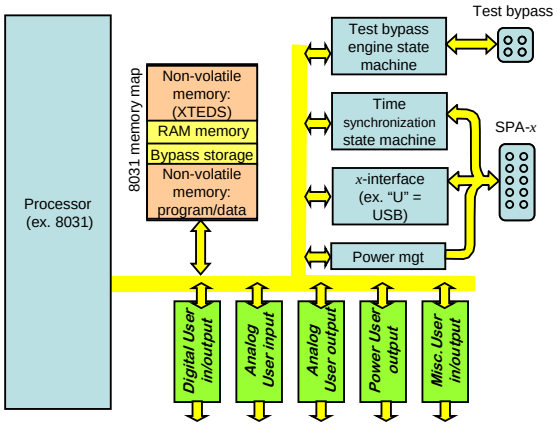

Figure 2. Generic Appliqué Sensor Interface Module (ASIM).

A bridge between legacy component standards and SPA standards

The aerospace industry is plagued with a vast array of incompatible interface standards. The integration of numerous components and payloads utilizing many different connection standards into a spacecraft bus is one of the more time-consuming aspects to spacecraft design, often leading to time delays and cost overruns. However, if non-PnP components are affixed with a SPA interface, the actions of integrating components into a SPA-compliant bus are reduced to simple plugging functions, thereby vastly reducing satellite build time. One of the primary functions of the ASIM is to serve as a bridge between legacy components and a SPA network. On one side, which we will refer to as the host side, the ASIM functions as a SPA device, communicating with the SPA network via the SPA-*x* protocol. On the other side, which we will refer to as the target side, the ASIM communicates with the legacy device according to its native communications protocol.

A certain amount of time is still required to program the ASIM to communicate with the legacy device, but this overhead is a small price to pay for the time and reduction of human-induced errors saved during integration. This action could even be incorporated into the component design itself, encapsulating both the

SPA nature of the ASIM and functionality of the component into one single truly SPA-compatible device. All SPA devices and structural panels could be designed in this way and stored until required for a new spacecraft. Spacecraft construction would then consist of connecting panels together to form a bus, selecting whatever components are required for the specific mission, pulling them off the shelf and plugging them into the panels.

Encapsulation of complexity

There are a number of ways in which the ASIM achieves encapsulation of complexity, a desirable element of the SPA paradigm. The previously described goal of glueless modularity is accomplished by constraining each modular building block to do three things: 1) to perform its natural function as a sensor, processor or actuator, 2) to provide as simple a physical interface as possible, and 3) to be able to negotiate that interface without outside influence or custom "glue" by the spacecraft designers. In previously explaining how encapsulation is accomplished in the SPA architecture, we stated that this was accomplished both in hiding the complexity of the hardware (including various communications interfaces or signal connections) behind a simple, single-point interface and by encapsulation of software including self-description of a device via its xTEDS. The ASIM plays a key role in both areas as described below.

In Hardware

There are essentially three models for complexity hiding in hardware interfaces, as shown in Figure 3. The first approach (Figure 3a) is simply to not be concerned with it, leading to a polyglot of wires from components that are intermingled with similar polyglots. This is typical practice in many complex systems, including aerospace systems. A second approach (Figure 3b) involves the creation of an intermediate interface, which is used to launder a number of disparate interfaces into a standard middle or meta-interface. In this case, a standard interface is involved, and the approach can be successful in reducing interface complexity, but the encapsulation is still limited and much of the interfaces between components are exposed. The most aggressive modular approach to encapsulation involves the use of a single point connection (Figure 3c). While this also may involve the use of a meta-interface, the meta interface and the component are encapsulated, almost literally as a black box.

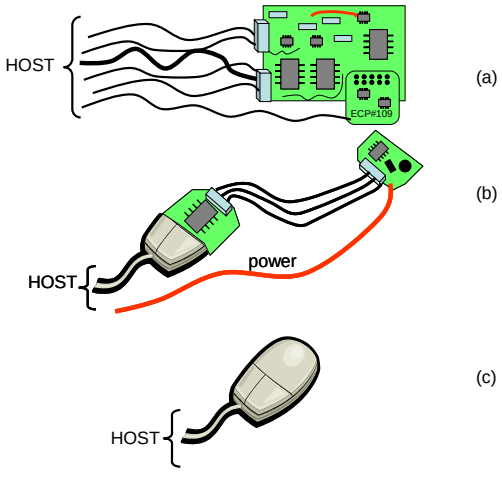

Figure 3. Three encapsulation approaches. (a) Non-encapsulated. (b) Meta-interface. (c) Encapsulation.

The ASIM supports full encapsulation and helps achieve pure modularity by providing a single point interface between components and the SPA network. Various separate power, ground, signal and data connections of legacy components are connected to the ASIM on the target side and combined into one simple host-side SPA connection which includes power, data, time synch and a single-point ground. To be more specific, the SPA interface uses the SPA-*x* interconnection protocol signals (such as *VBUS*, *D-*, *D+*, and *GND* in the case of USB) intact as the "data" portion and simply adds to that *28V* power and return conductors, *PPS_P* and *PPS_N* for 1 Hz (pulse-per-second) time synchronization, and a single-point ground line for chassis grounding. Some implementations of SPA (such as the Plug-and-Play Satellite[11] in its current form) include SPA USB (SPA-U) and SPA SpaceWire (SPA-S) interfaces as well as a test bypass interface together in one 25-pin, single-point SPA connector. This provision allows the exploration of both low- and high-speed components in the same interface, along with test bypass. For missions like PnPSat, whose purpose *is* to explore system-wide utilization of PnP, the dual-SPA-plus-bypass connector provides flexibility to study the impacts of network utilization. The co-integration of test bypass allows the connection of hardware-in-the-loop simulation (HWILS) through an interface to a single spacecraft panel, providing access to every component on the entire spacecraft bus through test bypass routers that are located in each panel. In other missions, SPA-U is allocated through 9-pin connectors, and test bypass is accommodated on secondary connectors.

The best way to incorporate test bypass in a developing SPA system is still an open issue. The use of separately articulated connections for test bypass reflected an initial intent to use test bypass as a temporary connection to parts of a SPA network. In practice, test

bypass has been shown to be valuable, but a large bundle of individual connections is cumbersome. Combining the test bypass with the primary SPA interface simplifies the cabling problem, at the expense of introducing a bit more complexity in the use of test bypass routers. However, it is not always necessary or desirable to constrain test bypass to follow the topology of the overall SPA network.

In Software

As in all embedded systems, software provides intelligence to the device. In the case of SPA, this provision includes commanding/controlling a component, wrapping up the complexity of the interfaces into one simple defining document for each device or application (the xTEDS), and properly interpreting and utilizing those interfaces both on the side of the SPA device and on the side of the greater SPA network. The ASIM plays an important role in all three aspects: First, the ASIM attends to the "care and feeding" of the device. The ASIM not only collects data from and/or commands the component according to its native communications protocol, but also takes care of component safety (maintaining certain power and temperature thresholds) and provides data and commands to the device that may be required for its correct operation.

Second, an ASIM performs the simple but important function of storing the xTEDS for its device and sending this document to the SDM when requested by the SM. Besides simply housing this document, though, the ASIM contains the application code that performs all the functionality described therein. For example, if the xTEDS for a temperature sensor includes a data message that sends the device temperature once every second, then the ASIM must contain code to read the temperature sensor, perform any necessary conversions, time stamp the data, arrange it in the proper format and send it once a second.

Finally, an ASIM cooperates with its SM to negotiate the SPA interface. In the SPA-U interface, the ASIM and SM communicate using a special messaging protocol called the appliqué sensor messaging interface or Appliqué Sensor Interface (ASI). The ASI includes a number of commands called by the SM that must be supported by the ASIM, as listed in Table 1. Depending on the attached component, the ASIM may not be required to perform every command listed in the table, but at the very least should send a response for each. ASIM responses are listed in Table 2.

Table 1: ASI Command Messages.

Name	Cmd	Length	Data							
Initialize	I	0								
Reset	R	0								
*Self test	T	0								
Request Data	M	8	interface ID	message ID	IP address					port
Cancel message	C	2	interface ID	message ID						
*Request counted strean	N	6	interface ID	message ID	count					
Power on	P	0								
Power down	F	0								
Request Version	U	0								
Command	V	length	interface ID	message ID	data					
Read xTEDS	X	0								
Time at the Tone	O	8	seconds							μseconds
Status	S	1	status							
Data	D	length	interface ID	message ID	data					
xTEDS	X	length	xTEDS							
xTEDS & PID	Y	length	PID (4 bytes)	xTEDS						
Version	V	1	version							

Table 2: ASIM Response to ASI Command Messages.

Name	Response
Initialize	Status
Reset	Status
*Self test	Status
Request Data	Data
Cancel message	
*Request counted stream	Data
Power on	Status
Power down	Status
Command	Status
Read xTEDS	xTEDS or xTEDS and PID
Time at the Tone	Status

Unified approach to building PnP networks

SPA networks consist of "endpoints" interconnected by hubs and routers. The order and location of endpoints is usually unimportant because the location of each endpoint and the capability provided by it to the system are maintained by SDM. However, some mechanism must exist that describes this capability to SDM and negotiates interfaces in order to effectively provide this capability. As we have already seen, these two functions (self-description of components and machine-negotiated interfaces) are provided by the ASIM. If every endpoint (including structural components) is a SPA device with its own ASIM, the self-forming network paradigm of the SPA architecture can be better maintained and implemented in a consistent fashion.

Simplified component and system testing

The SPA paradigm cuts a significant amount of time out of the traditional space vehicle development schedule by simplifying testing. Rather than designing or acquiring complex test structures to physically exercise components or having to, for example, heat components to provide realistic thermal data to thermometers throughout the system, SPA utilizes a novel test bypass interface to inject simulated data into the system at the component level. Here, too, the ASIM provides a consistent, unifying mechanism for

implementing the concepts of SPA. While not required strictly to be a SPA device, the test bypass concept must be considered so useful as to compel its incorporation within the SPA framework as an adjunct concept. Much in the way that software development tools provide useful test and debug features to support programmers, test bypass provides useful test and debug support for complex systems based on SPA.

When used in an ASIM implementation, the "test bypass engine" is tightly coupled to the ASIM internal processor. As the ASIM collects data from an attached component, it has the option to write these data into a dual-ported register file (as shown in Figure 4) where they are time stamped before being provided to the rest of the system (beyond the device using the ASIM) through the SPA interface. The time stamp is in itself useful, but the variables in this register file can also be overridden when *bypassed* by controlled data provided from an outside source, such as a HWILS facility. The ASIM supports a 255-word dual-port register file (register 0 is reserved for the current time) which is normally accessible by the ASIM internal processor. When test bypass is activated or engaged, however, any write operations of component data to the register file by the processor are masked and instead synthetic data is written from an external source through the test bypass interface. The masking is selective, as determined by a mask bit for each variable. If the ASIM then receives a request for data, it reads the register file

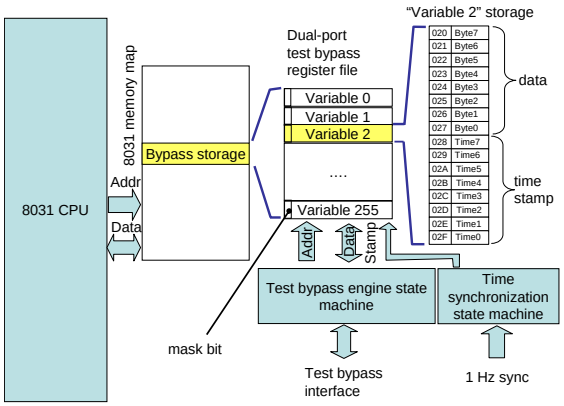

Figure 4. Test Bypass.

and sends out the data it finds there (now simulated data rather than local data collected from the component). The requesting application or device does not know it has been "fooled." Hence, test bypass is non-intrusive.

Besides enabling system-level testing through the TBI, the ASIM also enables simplified component-level tests. Referring back to the ASI commands listed in Table 1, we can see that one of these commands is a "self test." The implementation of this command is device dependent. It is the responsibility of the ASIM developer to write a test routine tailored to the attached device. An example of such a test routine could be the spinning up of a reaction wheel (and measuring its speed), or executing a component's own custom functional tests as designed by the component manufacturer. The self test may be simple or extremely rigorous, depending on the desire of the SPA device developer. It is expected, in any event, that more thorough factory tests will be performed on each device well before it is provided to a plug-and-play spacecraft. The detachment of detailed factory tests of components from integrated system testing is either a hallmark or critical limitation of the SPA philosophy, depending on one's perspective. The philosophy follows that of USB components, the idea being that users do not rip open keyboards to inspect solder joints or re-verify the quality of the embedded microprocessor software. Users use devices and are not auditors, but trust the quality of devices or vote with their checkbook. While this is a dramatic oversimplification for a warfighting platform, elements of this approach make sense, particular if one wishes to build a system 100X faster.

One of the unique benefits of the SPA paradigm is that components of all types could be developed, tested and stored for an indefinite period of time. It is in fact expected that components already exist when a spacecraft is implemented in response to an emergent mission need. Actual spacecraft system design would be accomplished primarily through the use of automated software that would take (as inputs) mission requirements and determine (as outputs) which of the available components would be required to form a buildable spacecraft and in what configuration the components would need to be set. Selected components could then be pulled off the shelf and plugged into the bus without painstaking tests involving racks of specialized equipment. The self tests would be performed on components as needed to ensure they still function properly as they are activated and integrated into a system. The test bypass interface then plays a role in supporting platform-level "day-in-the-life" tests through a HWILS facility, which could in a minimal case be a single laptop computer.

CURRENT STATUS

AFRL has created two prototype versions of an ASIM, referred to as "Generation 0" (Gen 0) and "Generation 1" (Gen 1). Gen 0 was a "house" version of the ASIM (developed by SAIC, Albuquerque, NM) used predominately in the Responsive Space Testbed (RST) for initial exploration of SPA devices and networks. The most current version of the ASIM (Gen 1)[13] is based on the SPA-U interface and has been made

commercially available (Data Design Corp., Gaithersburg, MD). The ASIM is currently a small printed wiring board (PWB), employing an FPGA-based design serving as a "soft testbed" for further refinement. The following sections describe the hardware and software features of the Gen 1 ASIM in greater detail.

Hardware

The Gen 1 ASIM (Figure 6) is manufactured on a PWB measuring 2x2 inches, with 100 pins in two double-row pin headers. One header is labeled the "host-side" connector and includes the SPA-U, time synchronization, test bypass, JTAG interfaces as well as power and an isolated serial port. The other ("target-

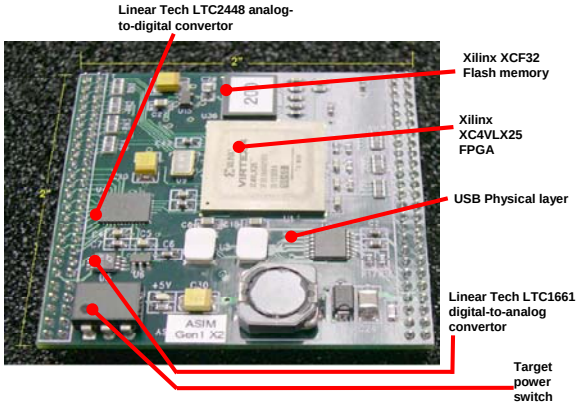

Figure 6. Gen 1 ASIM.

side") header includes most of the connections necessary for interfacing to a component: digital and analog I/O ports, 3.3V power for sensor electronics, and a switched 28V power connection that can be commanded by SDM through the ASIM to turn SPA components on or off. The Gen 1 ASIM architecture is similar to the generic ASIM depicted in Figure 2.

The core ASIM component is a Xilinx Virtex 4 high density FPGA (XC4VLX25), surrounded by various low-density supporting circuitries. An effort has been made to express as much of the ASIM design in the FPGA as possible in order to facilitate movement of the design to a flight-qualified Structured ASIC in the future. Figure 5 shows the basic intellectual property (IP) blocks in the FPGA. The main processor block is a variant of the Intel 8031 architecture (similar to Dallas Semiconductor's 80C320) implemented as a 48 MHz softcore processor. Also resident on the FPGA are peripheral device interfaces for digital and analog I/O and a USB serial interface, memory mapped to the processor. Other IP blocks used by the ASIM include a test bypass engine and a time

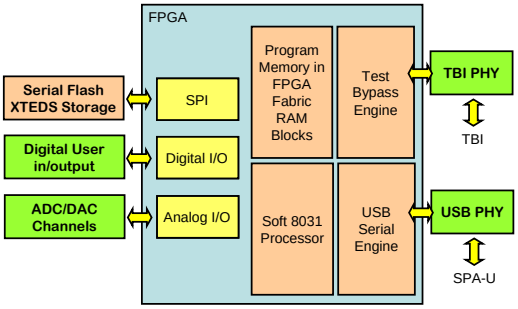

Figure 5. Gen 1 ASIM FPGA IP blocks.

synchronization/timestamping mechanism. A flash device (Xilinx XCF32PFS4BC) provides non-volatile configuration memory for the FPGA and the code/data configurations used in the softcore 8031. The user can program this memory via JTAG to allow the device boot from flash at power on. A second smaller EEPROM (Atmel 8Kx8) is connected only to the FPGA and provides persistent storage of the device xTEDS.

The ASIM supports power switching to a load (a user's device) through a MOSFET at 700mA. Power consumption of the Gen 1 ASIM is about 1.3W, more or less depending on usage.

SPA-U Connections

SPA-U connections are located on the host side pin header and include all standard USB signals (*VBUS*, *D+*, *D-*, *GND*), 28V power and return conductors (*28V_P*, *28V_RET*), time synchronization (*PPS_P* and *PPS_N*) and a single point ground (*CHGND*). SPA-U signaling is limited to full speed (12MHz) only. Additionally, the +5V USB power source is used only for signaling functions, not to provide power to a peripheral device.

The one pulse-per-second (1PPS) signals are implemented with an RS-422 differential pair and can be either a receiver (as in most SPA devices) or a source (if, for example, the attached device is a GPS receiver it can feed the 1PPS signal through the ASIM to the rest of the spacecraft).

Serial Connections

The host side header also includes a set of configurable isolated serial ports, test bypass connections, and pins reserved for SpaceWire. The isolated serial port pins are electrically insulated from the FPGA using galvanic isolators. On the Gen 1 Development Breadboard (also manufactured by Data Design) these isolated serial pins are connected to an RS-232 converter that can be used for debugging ASIM application code. The Test Bypass

pins are connected to the serial port of the test bypass engine in the FPGA. Although not originally part of the Gen 1 ASIM FPGA baseline hardware, the RST has been undergoing research to develop a SpaceWire version of SPA which will be discussed later in this paper.

Target Side I/O

The Gen 1 ASIM target side I/O includes 16 digital I/O pins, 16 analog inputs, and 2 analog outputs. The digital I/O can be configured in a bitwise fashion as either input or output. The analog inputs feed into an LTC2448 sigma-delta 24-bit analog to digital converter. The analog outputs are connected directly to an LTC1661 dual 10-bit digital to analog converter.

Software

In order to facilitate the development of coherent application software, the abstract software model shown in Figure 8 was developed for the Gen 1 ASIM. The main executive dispatches tasks associated with the SPA-U communications. The USB process knows how to use the USB serial engine to communicate over the SPA-U network and exports functions for applications to use in sending data to the host. The application code collects, calculates, and modifies data from a component in order to prepare the data to be sent to the host. It also commands the device when a command is received from the host. Hardware drivers are written

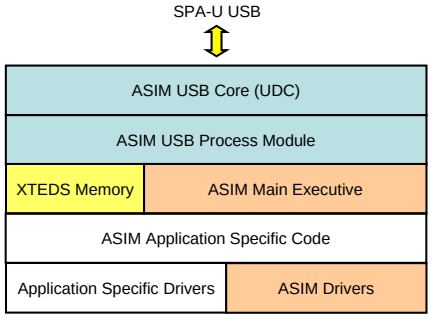

Figure 8. ASIM Abstract Software Stack.

and functions exported to manage I/O with any piece of hardware. The ASIM drivers have already been written for all ASIM hardware. Application specific drivers include those written by an ASIM developer for any hardware associated with the attached device. The xTEDS is written to describe the application software including the commands, data and services it provides. It is stored in flash and accessed by the host independent of the application software.

Baseline source code has been written in C to implement much of the communications and management of ASIM hardware and to provide a framework for the ASIM developer to write application specific code for his/her device. The baseline source code is a modular design with all modules tied together in a Keil Microvision compiler project. Two types of modules exist, driver modules, which abstract the use of hardware, and task modules which are written for every major software piece. An outline of the baseline source code is shown in Figure 7.

The application task module includes a few key functions that are worth mentioning. The process function (ASIMAPP_Process) covers the care and feeding of the device. It performs a number of tasks in a round-robin fashion to monitor the device electronics and write data to the registers.

Main Executive (ASIMFW.C / GLOBAL.H)		
USB Communication Task (USB.C) Performs all SPA-U communication. Requests data from application task.	**Application Task** (ASIMAPP.C) Performs sensor specific tasks. Uses USB task exports for communication.	**Debug Task** (DEBUG.C) Uses the serial port for printf debugging. Can be removed when not needed.
ASIM Driver Modules USB Device Definitions (UDC.H) Analog Input (ADC.C) Analog Output (DAC.C) Digital I/O and Power Relay (DIO.C)	**Application Driver Modules** Test Bypass Register File (REGFILE.C) Sample Application (LCD.C)	Driver For Serial Port (SERIAL.C)

Figure 7. Baseline Source Code.

ASIMAPP_DataMessage responds to requests for data from SDM (via USB drivers) by reading data from the registers (note this could be real sensor data or a HWIL simulation) constructing a data message and sending it back to the host via the USB module. Depending on how the data message is defined in the xTEDS, this function may be coded to send the data periodically. The last function worth mentioning is ASIMAPP_CommandMessage. When the USB module receives a command message from the host it calls this function so any code for commanding the device should be located here.

Movement to SPA-S

USB

One of the originally baselined communications protocols for the Responsive Space Testbed (RST) was USB 1.1, which has served as the primary communication mechanism to interconnect various spacecraft component modules. One of the disadvantages of using USB 1.1 is the 12 Mbps communications speed. Because USB is a bus type architecture, this bandwidth must be shared among all devices on the USB network, and USB's high overhead must be recognized.

Another disadvantage of USB is that it imposes a tree network topology, which is usually not conducive to the

redundancy inherent in space systems. That is, the tree structure requires a root node, a USB host, to direct all transactions on the bus. No communications can occur directly between endpoints. All communications are initiated by the USB host. The topology of USB required that special USB hubs, termed Robust Hubs, be engineered for the project. The Robust Hubs are capable of electrically controlling USB connections to ensure that only one USB host is activated within the space system, even though there may be more than one present. The Robust Hubs also force the given spacecraft topology to conform to the USB specification, through steering, activating and deactivating USB links.

While USB does have disadvantages when considering its use in a space system, it also has several advantages. The primary advantage of USB is the inherent plug-and-play features which are incorporated into the USB standard. These features include the automatic detection and enumeration of newly attached USB devices, as well as the ability to recognize their detachment from the network. These built-in PnP features of USB worked well in supporting the development of modular PnP components. That is, the ability to immediately recognize the attachment or detachment of a device is critical in PnPSat, where a rapid integration of components is required.

SpaceWire

The motivation to utilize SpaceWire (SpW) began as a mechanism to increase the data bandwidth for certain components. Since the popularity of SpW has flourished, and since it was specifically designed for satellite network communications, it seemed like a logical choice. During the process of incorporating SpW into the RST, it became apparent that supporting two communications protocols (USB and SpW) was not required. The utilization of one network communications protocol for all tasks would result in a simpler system. Thus, the migration from USB to SpW has begun.

SpW is a network interconnection communications protocol standardized by the European Space Agency. It was designed and developed specifically for space applications. The implementation of SpW into a programmable logic device is relatively straightforward, and it occupies a small silicon footprint. LVDS (Low Voltage Differential Signaling) is utilized by SpW at the physical layer. LVDS I/O is common in all major FPGA vendors, as well as standard cell ASIC design libraries. Because SpW uses a data-strobe encoding mechanism, the inclusion of a phase lock loop is not required. A simple XOR gate is all that is required to extract the bits from the data stream.

SpW has been successfully implemented into the Gen 1 ASIM in RST. The SpW link core was tied into the 8031 peripheral space, in much the same manner as the USB endpoint core was. There are challenges at the network level when transitioning from USB to SpW. One such challenge is the requirement to specify a destination in SpW. With USB, all transactions are originated and destined for the USB Host. Thus, no network routing information needs to be specified. With SpW, any node is a valid destination for a data packet. Thus, changes must be implemented to direct packets to the appropriate destination. This routing must be incorporated at the originator of the packet, the ASIM for data packets. Because of this, additional driver development and corresponding changes are required to ensure the ASIM adequately specifies the appropriate destination for its outgoing messages.

SpW is a switched fabric architecture. It consists of a series of point-to-point links, interconnected with routers, supporting intelligent routing through a header-consumption mechanism inherent in the SpW protocol. With this, bandwidth scales as the number of devices and routers are added to the network. This is in contrast to USB, which must share its bandwidth with additional devices. Thus, a large increase in the data throughput will be realized, which can be important for devices generating large volumes of data. The targeted upper link speed is 100 Mbps, with a realizable path to 200 Mbps. AFRL has demonstrated SpW on previous programs at link speeds of 625 Mbps. It is important to note that individual links can be operated at different speeds, depending on the capabilities of the two devices attached to that link. One link could operate at 10 Mbps, and interoperate with a high speed processor connection at 625 Mbps. The router is responsible for maintaining the various link speeds and throttling data flow.

SpaceWire PnP

SpW is only a bulk transport network protocol. SpW merely delivers data from point A to point B, which could be anywhere on the SpW network. There is no inherent PnP support in SpW. If a device is attached to a SpW network, there is no process whereby it will be recognized by a SpW network manager.

To address these deficiencies in SpW, relative to RST, a working group was formed to examine how PnP features could be incorporated into SpW. The working group consists of members from NASA Goddard, NRL (Naval Research Laboratory), AFRL, ESA (European Space Agency) and various industry representatives.

The working group has generated a protocol for including PnP into SpW. The main means of accomplishing this are for SpW routers to send messages to network managers when a device has been attached or detached from a router port. Network managers must write their return path into SpW routers. An interlock type mechanism has been developed to guarantee that any event will be recognized by the network managers. Additionally, the protocol readily supports multiple network managers, or hosts, to allow redundancy, as may occur in a space-based system.

SpW does include the possibility of layering changes onto the bare protocol, made feasible through the use of the Protocol ID, as defined in a new standard, ECSS-E-50-11. With the Protocol ID, various protocols may be layered on SpW and advanced features can be introduced. This is the mechanism in which the PnP Protocol has been added to SpW. The newly developed SpW PnP Protocol will be submitted to ESA for ratification in the near future.

DEVELOPING A SPA DEVICE

Conceptually, a SPA device is an encapsulated object, as suggested in the Figure 3c concept of modularity. Within this "black box" are a user's raw device and an ASIM, as depicted in the simplified architecture shown in Figure 9. In general, the ASIM eliminates the need for a SPA developer to understand the detailed

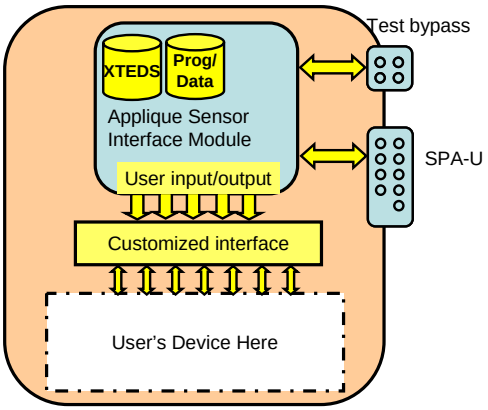

Figure 9. Simplified architecture of a SPA device.

transactional protocols of SPA-U (the primary emphasis of this paper), but instead only a simplified software-hardware interface. Ideally, native features available from the ASIM reduce the need for custom interfaces, but probably cannot altogether eliminate them.

When developing a SPA device, the most important thing for the developer to do is to fully understand the ASIM and SDM. The two most important resources are: (1) an ASIM development kit and (2) access to a version of the SDM documentation and source code. At the time of this writing, an ASIM development kit is available (Data Design Corp, Gaithersburg MD. URL: http://www.datadesigncorp.net) that includes an ASIM development breadboard, a User's Guide, and other useful documentation including hardware datasheets and schematics. The original SDM was developed in cooperation with Utah State University (Utah State University Space Software Lab, Logan UT. URL: http://gonzales.cs.usu.edu), and an extensive base of information is available on this implementation. While these versions of ASIM and SDM are currently well-maintained, they are not the only implementations. Other private implementations of the ASIM are under development, and at least one other version of SDM is under development[14].

After digesting the available ASIM and SDM documentation, the next step is to design interface electronics between the legacy component and ASIM or to design the SPA device electronics using the ASIM as a component. Many resources have been included on the ASIM for interfacing with a sensor, so very little external circuitry is required.

After the hardware has been designed and a prototype is available, the ASIM developer can begin writing the software. It is important early on to prepare an adequate and compliant xTEDS to describe the component application and interfaces. We envision the emergence of a well-defined common data dictionary (CDD), from which many common classes of components can be universally represented. Besides being necessary for device self-description, the xTEDS will also act as a framework for writing the ASIM application code. Currently, the xTEDS schema is available through USU, and Data Design supports a web-based xTEDS generator. A commercial XML validator can be used to write and validate the xTEDS against the schema. Once the xTEDS is written it must be transferred to the non-volatile memory of the ASIM using software tools provided by Data Design with their development kit.

Finally, C code must be written for the ASIM. The baseline source code is well-commented and acts as both a template and tutorial. Most of the code will be written in the application module (ASIMAPP.C) although the developer may want to write application drivers depending on the hardware that must be interfaced. The baseline includes a Keil Microvision project file and a batch file to convert the compiler output into an image suitable for input into the Xilinx tools. After compiling the code and flashing it to the ASIM, the device and attached ASIM should be tested with the SDM software.

DEMONSTRATIONS

Several flight demonstrations are planned that incorporate SPA as an experiment or as the entire avionics suite on the spacecraft. These include the following:

RESE

One of the first space experiments designed by AFRL to include SPA elements is the Re-Entry Structures Experiment (RESE). RESE is a sounding rocket that will be tested in the skies over White Sands Missile Range later this year. The rocket is a modular design with several flight decks, allowing six new technologies to be tested simultaneously. The AFRL Responsive Space Testbed has developed an entire SPA deck for RESE with four SPA components, three of which were provided by SAIC. Each of these was designed as a SPA device from inception with Gen 0 ASIM technology incorporated directly into the design and include a Magnetometer, a Therocouple and a Strain Gauge. The fourth experiment, an IMU designed by Montana State University, will be interfaced with a Gen 1 ASIM. The SPA deck uses a Parvus single-board computer running SDM version 1.4 on a linux mini-system.

SAE

The second SPA flight experiment is the Spacecraft Avionics Experiment (SAE). This is another entirely self-contained SPA flight deck that will be incorporated into TacSat3. This will be the first SPA technology to fly on an ORS-supported tactical satellite. Two "flavors" of SPA will be incorporated into the SAE: one is the "Smart Deck" which utilizes the method developed by AFRL as described in this paper (includes a SPA-U host and ASIMs interfaced with legacy components). The second is a proprietary version of SPA developed by MicroSat Systems, Inc. called the Intelligent Power and Data Ring (IPDR).

SAE involves four experiments including a Medium Sun Sensor, a Rate Sensor, an array of temperature sensors, and an AC Coupled Interconnect experiment. Both the Temperature Sensor Array (provided by Data Design Corporation), and the AC Coupled Interconnect (designed by North Carolina State University) feature Gen 1 ASIMs. However, the Sun Sensor and Rate Sensor will be interfaced to the SPA network using MSI's IPDR. The flight computer selected for this mission is the XScale Processor running a version of SDM ported to this computer. Depending on the TacSat3 launch schedule, SAE may be the first SPA experiment to earn flight heritage.

PnPSat

The final SPA flight experiment currently in the works is an entirely SPA-compliant satellite incorporating all the ideas listed thus far called the Plug-and-Play Satellite (PnPSat)[11]. This satellite is currently being designed and built in-house at AFRL's Responsive Space Testbed (RST) with the cooperation of a number of government contractors—mostly under the auspices of the Small Business Innovative Research (SBIR) program. The PnPSat will test two SPA interfaces, SPA-S and SPA-U, utilizing both a Robust Hub design for SPA-U devices and a SpaceWire Router for the SPA-S.

ROADMAP FOR FUTURE ASIM WORK

The future of the ASIM is tightly connected to the future of SPA. It is not enough to create a viable SPA, even to make it available to an interested community. Indeed, just as Moore's Law gives us continuously better commercial electronics for terrestrial applications, we must plan and execute a future for better ASIM and SPA components. In this case, better means both of improving the functionality and performance of ASIMs as well as reducing size, weight, and power, while improving robustness. Not all objectives will be met with the same ASIM, but a manageably small family could emerge to address the wide diversity of possible PnP devices in future spacecraft. This section will expose our current thought processes on the future of the SPA hardware infrastructure.

Current FPGA design as a "soft testbed" for development

The Gen 1 ASIM platform represents not only a near term evaluation and implementation platform, but also a developmental platform. As previously described, it is already being used to explore a SpaceWire implementation. The same platform can be used to study other types of SPA-*x* concepts or to examine new approaches for test, non-intrusive monitoring, synchronization, or any other features that might be considered useful extensions. It is also possible to replace the softcore with a different one or to explore tighter coupling of raw devices to the processor to enhance performance. Obviously, if the changes are significant departures from the current SPA-U design, they may not be directly compatible with the SPA infrastructure.

Toward a Family of ASIMs

It is not sensible to attempt with one interconnect standard to address all possible uses for that interconnect. For example, while ten gigabit Ethernet

(10GE) is capable of supporting almost any component bandwidth need, it would be imbalanced to use it for a single thermometer device. However, simpler interfaces are incapable of supporting the needs of high bandwidth devices. As such, it makes sense to consider a multi-tiered strategy in SPA for meeting a large range of needs in SPA devices, from the very simplest to the most demanding. Figure 10 is proposed as a pyramid principle, in which the height of the pyramid is performance and the width is component quantity. The notion suggested is that while at least some devices in a system will need very high performance interconnections, most parts of a system do not need the highest levels of performance. The diagram suggests four tiers:

- Very low data rate devices (< 10kbps)
- Low data rate devices (< 1Mbps)
- High data rate devices (< 1 Gbps)
- Very high data rate devices (>1 Gbps)

In the current SPA development, two of the four tiers are addressed: tier 2 is addressed by either USB (SPA-U) or the light form of SpaceWire (SPA-S), and tier 3 is addressed by SpaceWire (SPA-S). The top tier (tier 4), which shall be referred to as "SPA-10", is not currently addressed, meaning that the choice of physical layer protocol has not been established. SPA-10 is intended for the highest-performance payload and processing networks within a spacecraft. SPA-S does not naturally extend far beyond its present state-of-the-art (about 625 Mbps) due to the lack of an embedded clock recovery mechanism. Similarly, the bottom tier (tier 1) implementation, which shall be referred to as "SPA-1", which is intended for extremely simple devices such as

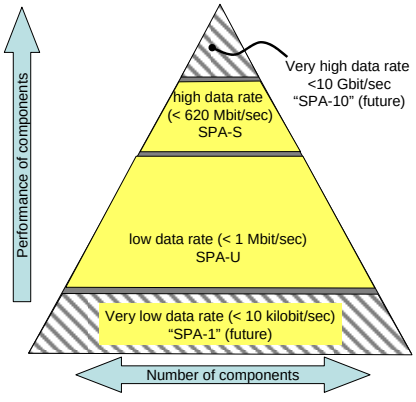

Figure 10. A pyramid diagram of interconnections.

a single switch, thermometer, or bolt (perhaps), is also undefined at present.

For planning purposes, the Figure 11 roadmap illustrates a possible set of development "trajectories"

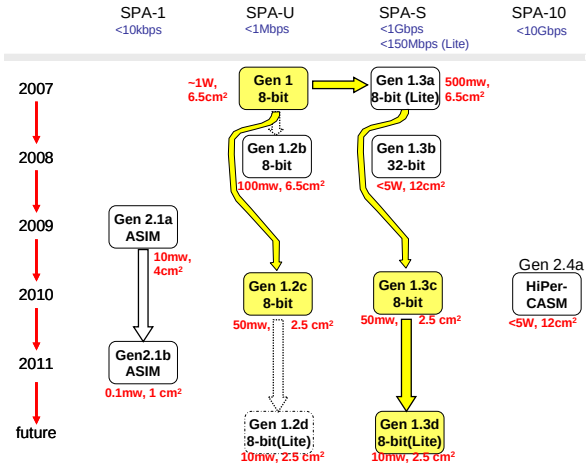

Figure 11. ASIM family roadmap.

for ASIMs that would accompany the four different tiers of interconnection performance. To simplify their differentiation, a nomenclature is employed of the form g.tr, where g is the "generation" (or spiral) of SPA, t is the tier or level, and r is the revision of ASIM for a particular generation and tier. For example, "Gen1.2c" refers to the third major version ("c") of a 1st generation ASIM, targeted to tier 2 in Figure 10. The nomenclature refers to ASIMs developed in association with AFRL research, and is not intended to extend to independent implementations of ASIMs by others. The potential ASIM family members are next discussed based on interconnection type.

SPA-1 ASIMs

The "tier 1" ASIM is envisioned as supporting the deep infiltration of SPA into minor components, such as brackets, distributed scalar sensors, and very simple devices. As such, an extremely low power and compact module would be required. It is expected that the first such ASIM (Gen2.1a) would operate with a 10mW average power budget in a 4cm^2 footprint. Goals for a second version are much more aggressive: 100µW in a 1cm^2 footprint. The physical layer interconnect standard for the tier 1 ASIM, called "SPA-1" is as yet undefined but logical candidates include the Maxim 1-wire[15] or a form of data-on-power interface. Yamar (http://www.yamar.com), for example, offers a number of data-on-power interface components for automotive use. Extremely Spartan microcontrollers would likely also be required, possibilities include some form of Microdot architecture[16] or asynchronous

implementations of commercial processors, such as Caltech's Lutonium, which could theoretically implement a 8031 class architecture with 25,000 MIPS/W performance[17]. Implementing self-description is expected to follow the same model as in other SPA tiers, through the use of xTEDS.

SPA-U ASIMs

Tier 2 ASIMs are predominately based on the SPA-U standard. The current Gen1 ASIM represents the first "near production" ASIM at the tier 2 level. As a developmental platform, it is not optimized for use in a flight system. The SRAM-based FPGA implementation provides maximal flexibility as we continue to refine the design, but this flexibility results in less power efficiency and increased susceptibility to single (and multiple) bit upset events in a flight environment. To combat these issues, AFRL is developing a 90nm rad-hard structured ASIC techology. Presumably, the IP contents of the ASIM would be transferred to a structured ASIC, leading to a more power-efficient (and compact) version. This version is most desirable, but not achievable in the near term due to the developmental status of the structured ASIC fabric. As such, we must consider the introduction of an interim variant of the ASIM, which could be based in an antifuse FPGA technology. This version, designated Gen1.2b, would be more quickly mobilized (as early as 2008) and presumably would be more power efficient, compact, and resilient to radiation effects (since configuration memory is eliminated) compared to the ASIM described in this paper. The ASIM based on structured ASIC is then designated Gen1.2c, projected for availability by 2010. It is likely that additional improvements in the ASIM design and the availability of a 65nm structured ASIC technology will lead to even more efficient implementations, and a placeholder Gen1.2d is indicated as a possible future ASIM, targeting a 10mW power consumption in a 2.5cm^2 footprint.

SPA-S ASIMs

SpaceWire, when combined with power distribution, synchronization support, and network discovery protocols, becomes "SPA-S". Two ASIM developments are currently in progress. The first of these (Gen1.3a), is based on the use of a softcore SpaceWire "lite" interface integrated into the Gen1 ASIM, which replaces the USB interface. Though SpaceWire is cast in the role of a "tier 3" interface, the Gen1.3a ASIM is actually more balanced as a "tier 2" solution, due to the lower speed (160-200 Mbps) of the "lite" SpaceWire core and the use of the 8031 as the central ASIM processor. To accommodate higher-performance SPA-S device designs, it is necessary to use a "full" SpaceWire core (capable of supporting >600 Mbps) and a more powerful (e.g. 32-bit) processor as the ASIM central processing unit. One promising development, supported through AFRL and BAE Systems (Manassas, VA), employs a system-on-a-chip architecture centered around the 32-bit Rad6000. The design (Figure 12), which includes a four-port SpaceWire router, employs a rich variety of breakout user interfaces, including JTAG, PCI, MIL-STD-1553, and UARTs. The radiation-hardened technology this component is based upon and higher processor throughput makes this design particularly attractive for legacy conversions and payload devices. At the time of this writing, the BAE component is under design. A Gen1.3b ASIM based upon it could be available by 2008. Beyond this, work

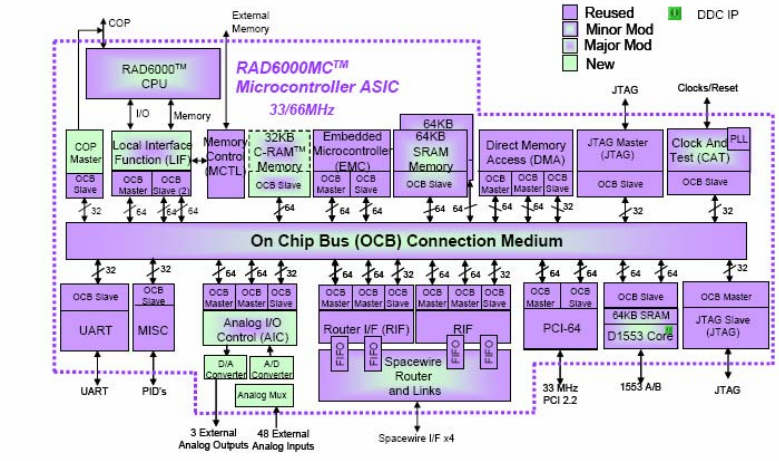

Figure 12. System-on-a-chip implementation, suitable for a SPA-S ASIM design.

on creating SPA-S based processing solutions in the laboratory is being explored based on the AFRL-developed Wafer Scale Signal Processor (WSSP)[19]. These developments are targeting a grid-like computing fabric, but it is not difficult to conceptualize the creation of a very high-performance SPA-S ASIM based on WSSP (although no such ASIM is represented in Figure 11). Furthermore, it is expected that the same advances leading to improved versions of the SPA-U ASIM could be transferred to the Gen1.3a SpaceWire lite implementation, leading to considerably improved versions (Gen1.3c and Gen1.3d, as depicted in Figure 11).

High-Performance ASIM

It is clear that a number of higher-performance sensors and communication elements in aerospace systems will drive bandwidths that are in excess of the bandwidths available in SpaceWire (i.e. < 1 Gbps). Many interconnection choices for a 10Gbps version of SPA (called "SPA-10") are available, but none have been chosen at the time of this writing. We have already taken steps to develop high-performance processing platforms suitable for implementing a "SPA-10" ASIM. This work, referred to as the "massively parallel processor" (MPP)[20], is based on extensions of previous work on the Malleable Signal Processor (MSP)[21-23], which focused on creating front-end processing blocks for high-performance sensors.

The target MPP architecture is shown in Figure 13. MPP generalizes the previous MSP work, providing a scalable supercomputing building block system that employs reconfigurable processing nodes and a switch fabric with non-blocking crossbars. The MPP is based on the creation of a powerful but fundamental node (Figure 13a) based on a single high-performance FPGA (Xilinx Virtex IV, XC4VFX100) with two dedicated (hardcore IP) PowerPC 405 processors, a generous amount of on-board memory (i.e., dual banks of DDR2 memory, totaling two gigabytes), self-contained configuration management infrastructure, and very high off-board connectivity. Many other FPGA-based processor designs, including the previous MSP prototypes, employed an ad hoc arrangement of several FPGAs in a tightly-coupled arrangement on a single board. Scalability is possible, but problematic, due to the need to divide complex problems into a partition of multi-FPGA clusters. By focusing on a single FPGA, the need to create ad hoc custom intra-board, inter-FPGA interfaces is eliminated. To compensate for the tight-coupling of a multi-FPGA design, the MPP implements 20 off-node MGT links, supporting up to 200 gbps off-node bandwidth.

In order to implement the MPP effectively, it will be necessary to commit a fraction of the available gate resources of the FPGA to implement the bus structures, state machines, and a non-blocking crossbar. The use of non-blocking crossbars permits the effective scaling of MPP nodes to form very large parallel architectures of configurable processing nodes, as suggested in Figure 13b. Each node is autonomously configurable, permitting the flexible implementation of fault tolerance strategies. Configuration of the current MPP prototype is managed using a specialized component developed by Xilinx for this purpose (referred to the "System Ace"). As it is the objective of SPA to support space programs, it is necessary to develop a radiation-hardened version of the system ace, which is simply referred to in Figure 13a as "space system ace". This "space system ace" will support a number of functions commonly required in space implementations of SRAM-based FPGAs including the distribution of bitstreams, configuration memory scrubbing, and possibly the support of hardened clocks, guarded user input/output signals, a small scratch-pad memory, and other concepts identified in research from another AFRL program, referred to as the "Virtual FPGA"[24]. At least one BAE Systems program is being pursued presently by AFRL that examines a "universal FPGA support device"[18] that implements FPGA configurations

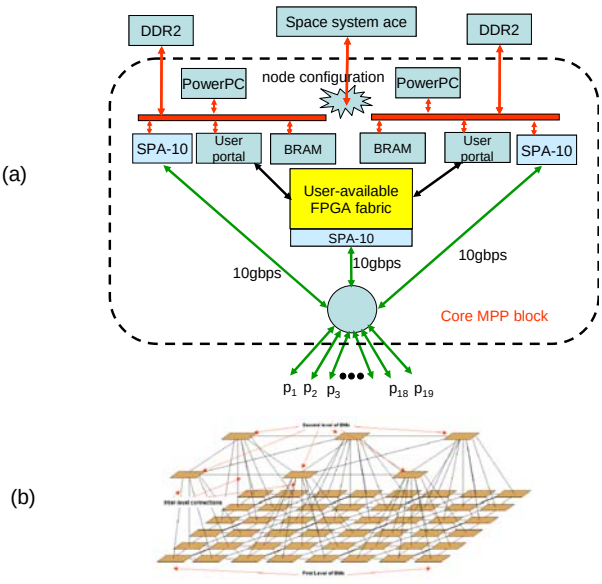

Figure 13. The massively-parallel processor (MPP). (a) Node architecture. (b) Example scaled system of nodes, depicting a hierarchical arrangement.

using rad-hard chalcogenide memory (instead of flash as used in commercial systems) for non-volatile storage.

The projected ASIM based on the MPP is referred to as HiPer-CASIM. To extend the concept of the MPP node as an ASIM, it is necessary to add a breakout user interface, power management, and additional state machines to support synchronization and test bypass as shown in Figure 14. This ASIM would contain a SPA-10 routing infrastructure, with the likely form of SPA-10 exploiting the MGT infrastructure available in the MPP design.

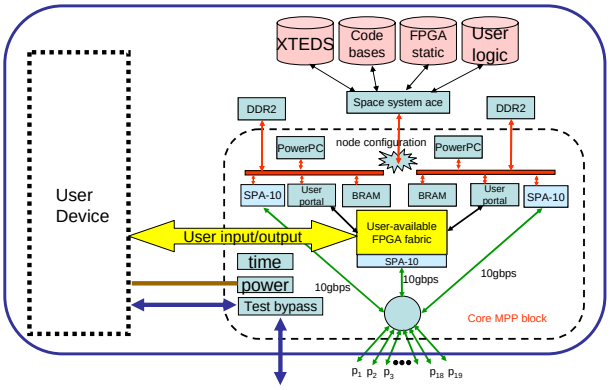

Figure 14. High-performance computing ASIM (HiPer-CASIM).

CONCLUSION

This paper has served to explain the ASIM both as a device and as an enabling technology for SPA systems. Following the model of commercial PnP technology, the ASIM acts as an interface control chip, hiding the complexity of device interfaces and negotiating these interfaces through xTEDS and application code. Additionally, the ASIM provides a method of simplified component-level testing and non-intrusive system-level testing through the TBI. The current Gen 1 ASIM supports both SPA-U and SPA-S protocols and serves as a soft testbed for further development and refinement. Eventually, an entire family of ASIMs is envisioned to meet anticipated power, bandwidth, and size requirements of future SPA components and systems. A number of flight experiments are planned that will incorporate ASIMs as key components in the SPA architecture. Gen 1 ASIMs will fly both on RESE and SAE and a more robust version will fly on the PnPSat. These experiments will provide crucial data and flight heritage necessary to prove SPA as a viable concept in achieving ORS.

REFERENCES

1. Wegner, P. "Operationally Responsive Space," presentation, AFRL/VS, April 2007.
2. Rumsfeld, D., "Report of the Commission to Assess United States National Security Space Management and Organization," 2001. URL: http://www.fas.org/spp/military/commission/executive_summary.pdf [cited 9 May 2007]
3. HQ AFSPC/DRS, "Mission Need Statement for Operationally Responsive Space," AFSPC001-01, 2001. URL: http://www.smad.com/ORS%20MNS%20_Final%20Dec01_.pdf [cited 9 May 2007]
4. Cebrowski, A.K., and J.W. Raymond, "Operationally Responsive Space: A New Defense Business Model" Parameters, US Army War College Quarterly, Vol. 35, No. 2, 2005, pp. 67-77.
5. Brown, K.K., "Is Operationally Responsive Space the Future of Access to Space for the US Air Force?" Air and Space Power Journal, Vol. 20, No. 2, Summer 2006, pp. 11-18.
6. Noel, J., R. Escorpizo, and E. Jones, "Transforming the National Spacelift Architecture," Proceedings of the 2nd AIAA Responsive Space Conference, Los Angeles, CA, April 19-22, 2004.
7. Wegner, P.M., and R.R. Kiziah, "Pulling the Pieces Together at AFRL," Proceedings of the 4th AIAA Responsive Space Conference, Los Angeles, CA, April 24-27, 2006.
8. Lyke, J., D. Fronterhouse, S. Cannon, D. Lanza, and W. Byers, "Space Plug-and-Play Avionics," Proceedings of the 3rd AIAA Responsive Space Conference, Los Angeles, CA, April 25-28, 2005.
9. Lyke, J., S. Cannon, D. Fronterhouse, D. Lanza, and W. Byers, "A Plug-and-play System for Spacecraft Components Based on the USB Standard," Proceedings of the 19th Annual AIAA/USU Conference on Small Satellites, Logan, UT, August 8-11, 2005.
10. Sundberg, K., S. Cannon, T. Hospodarsky, and D. Fronterhouse, "The Satellite Data Model," International Conference on Embedded Systems and Applications (ESA '06), Las Vegas, NV, June 2006 (ISBN 1-60132-017-5/CSREA, Editor H. R. Arabnia).

11. Fronterhouse, D., J. Lyke, and S. Achramowicz, “Plug-and-play Satellite (PnPSat)”, Proceedings of the AIAA Infotech Conference, 7-9 May 2007, Rohnert Park, CA.

12. Garney, J., J. Lueker, et al, Universal Serial Bus Specification, Revision 2.0, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips, Apr 2000. URL: http://www.usb.org/developers/docs [cited 5 May 5, 2007].

13. Shaw, M. “Appliqué Sensor Interface Module (ASIM) Application Development Kit For Generation One Experimental Modules,” Version 07.03.06, AFRL, 2007. URL: http://www.datadesigncorp.net/ASIM/downloads /ASIM_Development_Kit_Users_Guide.pdf [cited 9 May 2007]

14. Bruaene, J.V. “Standards-based Plug-and-play Distribution,” Proceedings of the AIAA Infotech Conference, 7-9 May 2007, Rohnert Park, CA.

15. “Overview of 1-Wire Technology and Its Use”, Application Note 1796, publication of Dallas Semiconductor/Maxim Corporation, 3 December 2002. URL: http://www.maxim-ic.com/appnotes.cfm /an_pk/1796 [cited 18 May 2007]

16. Donohoe, G.W., J.C. Lyke, and S. Cannon, “Microdot: a Tiny Microcontroller for Distributed Sensor Interfacing,” Proceedings of the 2nd International Conference on Integrated MicroNano Technology, Pasadena, CA, 11-13 April, 1999.

17. Martin, A.J., et al, “The Lutonium: A Sub-Nanojoule Asynchronous 8051 Microcontroller,” Proceedings of the 9th International Symposium on Asynchronous Circuits and Systems, 2003.

18. Marshall, J.R., R.W. Berger, J.E. Robertson, and S. Miller, “Reconfigurable and Processing Building Blocks in Responsive Space,” Proceedings of the AIAA Infotech Conference, 7-9 May 2007, Rohnert Park, CA.

19. Linderman, R.W. et al “A dependable high performance wafer scale architecture for embedded signal processing,” IEEE Transactions on Computers, January 1998, 47(1):125-128.

20. Parra, J., H. Pollard, and J.C. Lyke, “Towards a Reconfigurable Multiprocessor Architecture for Space missions: The AFRL-UNM High-end Reconfigurable System,” Proceedings of the Military Applications of Programmable Logic Devices (MAPLD) 2006 Conference, September 2006.

21. Kinashi, Y., R. Linderman, and J. Lyke, “DITP: A Flexible miniaturized Sensor Fusion Processor for next Generation Interceptor Seekers and Surveillance Sensors,” Proceedings of the 7th Annual AIAA/BMDO Technology Readiness Conference and Exhibit, Colorado Springs, CO. August 3-6, 1998

22. McGuirk, P., J.C. Lyke, G.W. Donohoe, “Malleable Signal Processor: A General-purpose Module for Sensor Integration,” Proceedings of the MAPLD 2000 Conference, Sept. 26-28, 2000.

23. Coxe, R.L., G.H. Romero, A. Pakyari, M. Leary, J. Lyke, and D. Fronterhouse, “Development of the Malleable Signal Processor (MSP) for the Roadrunner On-Board Processing Experiment (ROPE) on the Tacsat-2 Spacecraft,” Proceedings of the 2005 MAPLD Conference.

24. Marty, B. and J.C. Lyke, “Virtual Field Programmable Gate Array Triple Modular Redundant Cell Design,” Air Force Research Laboratory Technical Report AFRL-VS-PSTR-2004-1093, 28 April 2004