SSC10-VII-7

A Plug-and-play Approach Based on the I2C Standard

James Lyke, Jesse Mee
Air Force Research Laboratory
3550 Aberdeen Ave SE, Kirtland AFB, New Mexico, USA 87114-5776; (505) 846-4510
spaceelectronics@kirtland.af.mil

Fredrik Bruhn, Gael Chosson, Robert Lindegren, Henrik Lofgren, Jan Schulte
AAC Microtec AB
Dag Hammarskjölds väg 54 B,SE-751 83 Uppsala, Sweden; +46 707833215
{fredrik.bruhn,gael.chosson,robert.lindegren,henrik.lofgren,jan.schulte}@aacmicrotec.com

Scott Cannon, Jacob Christensen
Utah State University
Department of Computer Science, Logan Utah, USA 84322-4205; (435)797-2015
{scott.cannon,jacob.h.christensen}@cs.usu.edu

Bryan Hansen
Space Dynamics Laboratory
1695 North Research Park Way, North Logan, Utah, USA 84341; (435) 797-4600
bryan.hansen@sdl.usu.edu

Robert Vick
SAIC
2109 Air Park Road SE,Albuquerque, New Mexico, USA 87106; robert.w.vick@saic.com
robert.w.vick@saic.com

Alonzo Vera
Micro-RDC
8102 Menaul Boulevard NE,Suite B, Albuquerque, New Mexico, USA 87110; (505) 294-1962
alonzo.vera@micro-rdc.com

Josette Calixte-Rosengren
Swedish Defence Materiel Administration (FMV)
Banérgatan 62, SE-115 88 Stockholm, Sweden; +46 87826420
josette.calixte-rosengren@fmv.se

ABSTRACT

Plug-and-play architectures can reduce the timeline for constructing complex systems by automating the connections between components. While plug-and-play technologies have been successfully applied to aerospace systems, the overhead of the interface circuitry is a concern affecting its widespread use, particularly in smaller satellites. In this paper, we discuss a "minimalist" plug-and-play interface based on the popular inter-integrated circuit (I2C) standard, leading to dramatic simplifications of the interface circuitry necessary to be plug-and-play compliant. This concept, referred to as "mini-plug-and-play" (the space-qualified version is called "SPA-1"), has been created as a direct product of an international cooperative program between the United States and Sweden. At the simplest level, mini-PnP/SPA-1 is a protocol layer over I2C, readily implemented with existing devices that already support this ubiquitous standard. Using gateways, networks of mini-PnP/SPA-1 devices can connect to legacy forms of plug-and-play (e.g., SPA-U and SPA-S). Like these other legacy interfaces, SPA-1 devices support key features of plug-and-play, including electronic datasheets, automatic enumeration, and are readily integrated into plug-and-play software. This paper describes the development and demonstration of COTS and rad-tolerant versions of SPA-1 interface modules along with current status of the international program.

INTRODUCTION

The idea of a one-week spacecraft seems heretical. It is well-known that most satellite development programs have been plagued with cost growth and schedule overruns. In this case, costs are measured in billions of dollars and development times measured in years (some more than a decade). An aerospace analyst recently observed that all of the ten major military satellite systems under development by the US Department of Defense (DoD) were over budget and behind, raising the question "What are the things that these programs share in common that make it seem as though cost overruns are part of their nature? [1]".

Several years ago, the Air Force Research Laboratory (AFRL) began to study how technology could be used to reduce the complexity of systems using an architecture that implements a form of machine-negotiated interface. This approach, referred to as "Space Plug-and-Play Avionics" (SPA), was described previously in these proceedings [2]. Since then, the SPA technology concepts have been implemented in a variety of platforms, ranging from tactical satellites [3] to cubesats [4]. One implementation has been successfully operating in orbit (on the TacSat3 satellite [5-6]) for over a year at the time of this writing. In time trials, a SPA-based satellite has been assembled in less than two hours. Despite the successes of SPA, one of the primary criticisms is in the overhead associated with implementing the interface circuitry. Even though modules – called "appliqué sensor interface modules" (ASIMs) – have been created to simplify the creation of plug-and-play components, they are too bulky for the many simple components on a typical spacecraft. For tinier satellites, especially cubesats, the previous ASIMs consume too much power and displace too much volume to be effective.

To combat this problem, our team set out under an international agreement to create an extremely efficient version of a generic plug-and-play interface standard, far simpler than any we were previously aware of. Extending this technology to space applications would create a durable solution to the problems of size, weight, and power overhead in plug-and-play interface modules. This plug-and-play technology is referred to as "mini-plug-and-play (PnP)" (or SPA-1 for the space-qualified version). It preserves all of the key features of the previous SPA technologies, including self-description (through embedded electronic datasheets), automatic discovery when added to a network, and exposure of component services for use by a large system. Through simple bridging adaptors, entire networks of mini-PnP can be added to existing SPA networks, providing an upward/backward compatibility with previous USB-based (SPA-U) and spacewire-based (SPA-S) networks.

This paper is organized as follows. In the next discussion, we briefly describe the SPA philosophy for mixed-network systems, identifying the key technologies behind the SPA concept. Then we detail the rapid evolution and details of the mini-PnP/SPA-1 technology, followed by a section describing its application to simple systems in a number of subsystem domains. Finally, we provide a snapshot of the current status of the project and the expected extensions to flight programs.

BASIC PRINCIPLES OF THE SPACE PLUG-AND-PLAY AVIONICS (SPA) ARCHITECTURE

Terms like "plug-and-play" are often dulled through overuse. They can take on a range of definitions, and are subjectively interpreted. As such, it is important to clearly define the concept of any plug-and-play approach as concisely as practicable. In this section, we synopsize what it means to be plug-and-play in the sense of the SPA architecture.

SPA Components

In SPA hardware is referred to as "devices" or "components". Software is referred to as "applications" Components and applications are self describing using electronic datasheets referred to as extensible Transducer Electronic DataSheet (xTEDS). The xTEDS provide an interface description with sufficient detail for "most all" purposes. In SPA, no component features exist outside of those exposed by self-description (i.e., all features must exist in the xTEDS).

SPA Interfaces

SPA components employ some SPA-x interface. Standard interfaces are essential in a plug-and-play approach. In SPA, an interface has been designated as the combination of a physical layer (associated with a particular common interface standard) combined with other conventions and protocols governing the mechanisms used to expose the data within a device and the infrastructure needed to manage the device. For this reason, the SPA-U interface, while based on a compliant implementation of the USB physical layer and protocol, is *not* the same as a USB device, since USB devices do not have auxiliary power ports (for high amperage devices) or synchronization signals.

While a single standard is desirable, one size does not fit all, as suggested in the "pyramid diagram" (Fig. 1). This diagram attempts to illustrate the distribution of bandwidth needs in a complex system is dominated by many simple devices. A large spacecraft or launch vehicle may, for example, have many simple health and status monitoring devices, such as thermometers. Thermometers have very low data rates, usually sampled at rates of less than 1Hz. Of course, there are components that are more complex, such as guidance components, having data rates thousands of times higher, but there are not as many of these more complex components. Payload components can have even higher data rates, but there are even smaller numbers of these high performance components.

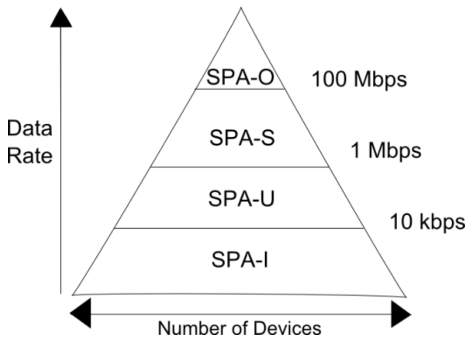


Figure 1. The "pyramid" of SPA interfaces.

SPA interfaces are typically viewed as "single-point" connections to devices. They provide power for the component, a command and data network connection, and a means for the distribution of time synchronization.

The original SPA standard was based on USB, the so-called "SPA-U" technology [1], based on the notion that it was an 80% solution, meaning that it was a one-size-fits-all solution "80% of the time". Soon after SPA-U, a plug-and-play system was devised for the higher-performance spacewire standard, giving rise to the SPA-S standard [7].

While the original two SPA standards (SPA-U and SPA-S) appeared to cover most needs in spacecraft, it was soon clear that both higher and lower performance SPA standards would be needed for practical systems. In the case of very high performance components, such as multi-gigabit cameras, it would be necessary to either use a higher speed non-SPA interface or bind together a large number of SPA-S interfaces to accommodate such components. For simpler devices, the power consequences and bulk of even the simpler SPA-U technology began to seem excessive. As such, we believe that four tiers for SPA are more optimal. Higher performance interfaces, based on the use of optical interconnect, called "SPA optical" or "SPA-O", can address the needs of very high performance components, but we shall not detail this concept further here. The lowest tier of SPA, the central focus of this paper and detailed in the next section, appears to provide a means to interface the many simple components in complex systems with minimum size, weight, and power penalties.

Applique Sensor Interface Modules. To simplify the creation of SPA devices, we followed the practice of other plug-and-play systems, which simplify the insertion of USB in peripherals through the use of interface chips. These chips translate the USB standard into a simpler, generic interface that is easily integrated into typical devices (like mice and keyboards), averting the need to master the subtleties of a complex specification (i.e., the USB standard). Similarly, for SPA the concept of the appliqué sensor interface module (ASIM) was introduced to simplify mating spacecraft components (especially legacy devices) to the SPA standard. An ASIM (Figure 2) contains circuitry and memory storage necessary for a non-SPA device to be converted into SPA-compatible form (in Sweden, the equivalent concept is referred to as the remote transceiver unit or "RTU"). ASIMs are not essential for SPA compatibility, but they can reduce the time needed to convert a typical component into a "SPA-ready" form and can insure a greater consistency in the conversion. In practice, an ASIM is connected to a user's device, intercepting all of the electrical signals that normally connect that device to a spacecraft. The ASIM in this case generates the power, command, data, and synchronization signals for the user's device. Since the ASIM may not always support all of this "care and feeding" in a direct interface, it may still be necessary to further create a customized interface to complete the "legacy conversion", especially if the device was not designed to be SPA compatible. By convention, the combination of a user's device with the ASIM (and any required customized interfaces) should be considered an integral unit, i.e., the SPA device.

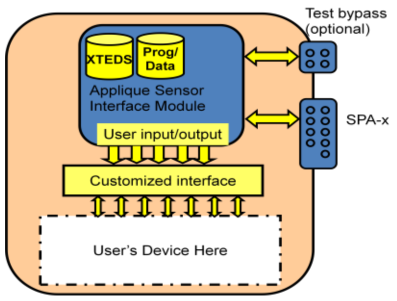


Figure 2. How appliqué sensor interface module (ASIM) is generically used to form a SPA Device

Test bypass. Another convenience made possible with ASIMs is the possibility of integrating support for a standard approach to hardware-in-the-loop simulation (HWILS). The concept for this integration in Figure 2 is shown as a separate *test bypass* connector. Inspired by the test access port (TAP) used in the JTAG standard [8] made popular for testing and configuring integrated circuits, test bypass permits an elegant approach for isolating elements described in a component xTEDS for direct substitution. With test bypass, the temperature normally read by a SPA thermometer can be overridden with a controlled value. Applying the test bypass concept across the spacecraft provides an unprecedented level of testability which has been shown to be very useful in the creation of SPA-based systems.

SPA Networks

SPA networks consist of *endpoint* components interconnected by hubs or routers (Figure 3). The concept of "hub" or "router" (or for that matter "switch" or "backplane") is closely tied to the definitions of the underlying physical layer of a particular SPA-*x* standard. In general, SPA-*x* extends the features of switching fabrics associated with interface *x*. In the case of SPA-U, a hub provides auxiliary power and synchronization and has in some cases been engineered to have greater fault tolerance than the traditional USB hub (leading to the idea of "robust hub" discussed in [1]). In SPA-S, the router, while spacewire compliant, implements a newer plug-and-play protocol, which supports address resolution protocols not part of the original spacewire standard.

While connections to SPA components are typically single-ported (i.e., one interface connection per endpoint), it is possible to have *redundant* endpoint connections. In Figure 3, endpoint component A has a redundant connection to two different routers.

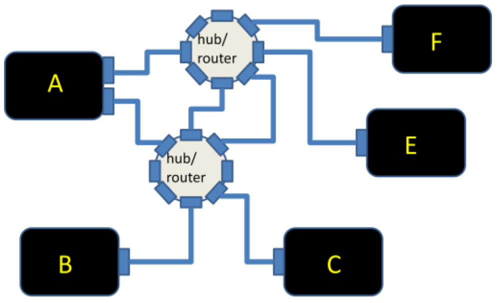


Figure 3. SPA network

SPA networks are scalable and size-agnostic. A SPA-based system supports scalability of processing, throughput, data storage, and power through the addition of modular devices or infrastructure hardware In principle, they can be unlimited in size, since hub/router components can be added to increase the overall bisection bandwidth of the system. However, specific SPA interface types may have limits imposed by the corresponding physical layer interface. For example, SPA-U networks are limited to 128 devices by the USB physical layer standard, though SPA systems could easily have many more then 128 SPA-U devices by having multiple SPA-U host ports (it is also possible to use more sophisticated hub concepts in which each port can act as host or endpoint, in effect regenerated the USB physical layer in multiple points of an overall SPA-U network).

Obviously, a SPA network does not "care" about the physical size of a system within which it is embedded. Barring limits of physical miniaturization, a SPA system can be embedded in a thimble, or used in the largest physical platforms.

SPA networks are topology agnostic and self-organizing. Endpoints are created equal such that a modular SPA-compliant component is pluggable and usable at any location on the network. While the specific connections in Figure 4 differ from those in Figure 3, the two networks are logically equivalent.

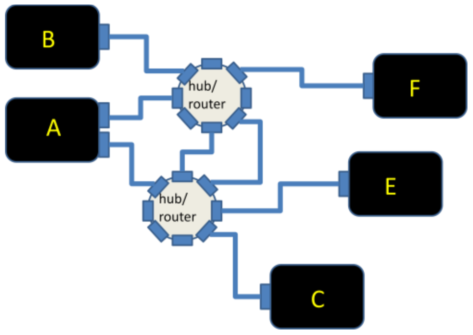
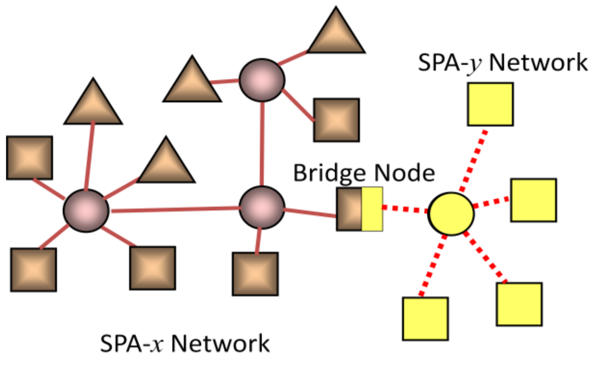


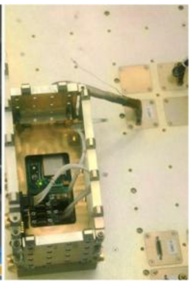
Figure 4. Re-arranged SPA network

SPA components communicate only through messages. In SPA networks, components operate through a sequence of transactions encapsulated with messaging protocols. The messaging protocols embed information useful to transport information throughout a SPA system.

Since "one size doesn't fit all", it is necessary to tackle the problem of mixed-network implementations in SPA systems. Bridges can be used to negotiate between SPA-x and SPA-y (Figure 5). Bridges provide an apparently simple adapter for users to connect any desired SPA-x component to a system, even if "*x*" is not native to that system. The hidden complexity involves the hardware and software infrastructure needed to launder signals between the dissimilar interface standard formats. Such "encapsulations of complexity" are consistent with theme of plug-and-play architectures.



(a)



(b) (c)

Figure 5. Connection of two SPA networks having different interfaces through a bridge. (a) Bridge schematic. (b) Laboratory demonstration of SPA-U components connecting to a SPA-S system. (c) Close-up of a SPA-U "container" capable of housing several small SPA-U modules, connecting to one of the uncommitted SPA-S ports on a plug-and-play modular panel.

SPA Systems

A SPA system is a network of SPA components. The system can be an enclave of SPA components within a larger conventional (non-SPA) system, such as the TacSat 3 spacecraft [6], which was a conventional satellite design containing a four-port SPA-U experiment having a traditional host (RS-422 plus 28VDC power) interface support the ad hoc protocol defined for the otherwise custom system. More exciting is the notion of platforms that fully embrace SPA architecture, such as the PnPSat series [3] in which even structural panels were SPA networks (each having eight SPA-S ports), coalescing into a larger SPA system when SPA components are added to panels and panels connected together.

SPA Middleware. At least one SPA component in a connected SPA network must capable of hosting software that can operate a mechanism referred to as "discovery and join". Simply put, this mechanism detects the existence of new components (and applications) on a SPA network and provides the ability to query the "services" provided by these components (as described in xTEDS). The xTEDS information is then registered within a lightweight embedded knowledge base, analogous to a searchable registry.

SPA systems do not require external data sources to operate. SPA components and applications find each other through a *look-up service*. The look-up service is an application that operates the aforementioned registry of the services of all SPA devices and applications. It emphasizes a *data-centric approach* utilizing queries for atomic data kinds, standard interfaces, and descriptive metadata exposed in the interfaces.

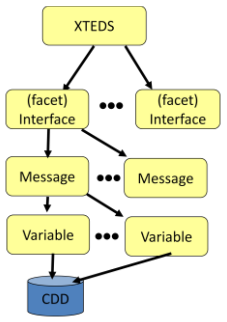
Most commonly, the open source software system known as the "Satellite Data Model (SDM)[1]" [9] has provided the plug-and-play mechanisms ("discovery and join" and "look-up service") for SPA networks. SDM, which has been compiled so far for Linux and VxWorks operating systems, not only provides these services and manages the messaging infrastructure, but also provides the framework for SPA-aware "apps" (software applications) that also contain xTEDS descriptions and automatically integrate into the embedded knowledge base. SDM matches producers of information with the consumers of information, analogous to concepts also referred to as "object resource brokering" and "service oriented architectures".

While SDM is often tightly coupled to the concept of SPA as a preferred embodiment, it is possible that alternative embodiments of middleware can be created. The idea of enhanced SDM and alternatives to SDM continues to be actively discussed in AFRL development programs, such as the Advanced Plug-and-play Technologies (APT) program, which has initiated six study contracts to explore refinements to the SPA concept.

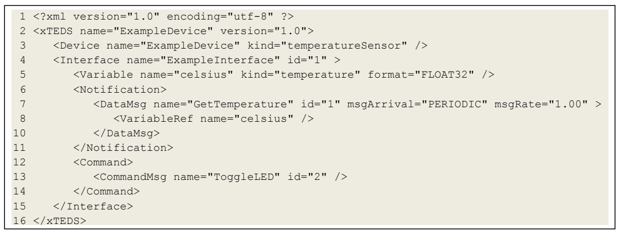
Ontology and System Conventions. Ontology can be thought of as a machine-accessible structuring of knowledge for a particular domain. In SPA, the atomic bits of knowledge are captured in a common data dictionary (CDD). The electronic datasheets (xTEDS)

[1] SDM is currently an open-source project maintained by Space Dynamics Laboratory (SDL) / Utah State University under contract to the Air Force Research Laboratory.

represent an arrangement of terms from the CDD to form messages (command or data), a number of which comprise an *interface*, one or more of which define an electronic datasheet. A very simple but representative example of an xTEDS (which is in XML format) is shown Figure 6. A hierarchical representation is shown in Figure 6a, and the corresponding XML structure for a simple example thermometer is shown in Figure 6b.



(a)



```
<?xml version="1.0" encoding="utf-8" ?>
<xTEDS name="ExampleDevice" version="1.0">
   <Device name="ExampleDevice" kind="temperatureSensor" />
   <Interface name="ExampleInterface" id="1" >
      <Variable name="celsius" kind="temperature" format="FLOAT32" />
      <Notification>
         <DataMsg name="GetTemperature" id="1" msgArrival="PERIODIC" msgRate="1.00" >
            <VariableRef name="celsius" />
         </DataMsg>
      </Notification>
      <Command>
         <CommandMsg name="ToggleLED" id="2" />
      </Command>
   </Interface>
</xTEDS>
```

(b)

Figure 6. Representative example of a very simple electronic datasheet.

The key differentiation of space applications from non-space, including automotive, medical, or perhaps children's toys, is the structure of knowledge domains as represented in the CDD. Possibly the most important work in building a self-consistent "universe" of plug-and-play systems hinges on the correct and consistent capture of the units of measure, the styles, the labels, and the capture of best practices. Failing this, the CDD, xTEDS, and everything in SPA that builds on this foundation are at some level compromised. For these reasons, we consider SPA a data-driven architecture, and a system is ultimately only as good as the practices used in representing its data.

CREATION OF A MINIMALIST GENERIC PLUG-AND-PLAY TECHNOLOGY

The goals for a minimalist plug-and-play technology were simple: achieve a plug-and-play interface standard in the minimum size, weight, and power footprint possible, with the minimum number of wires. The generic form of this technology came to be known as "mini-PnP", which could be universally applied in many ground applications at low cost. The space qualified version is referred to as SPA-1. SPA-1 devices are capable of being integrated into more sophisticated SPA systems through bridges. Through the principles discussed, the simplest SPA thermometer is logically on equal footing with the most sophisticated payload, differing only in the lengths of their descriptions.

In this section, we discuss the interface trade study, the design of the protocol, the interface module developments, and the considerations for more

Interconnect Trade Study

The requirements for minimal interconnect begin with the consideration of a physical layer approach. We reviewed a number of obvious candidates before arriving at the decision to pursue I2C, including SPI, RS-422, RS-485, SMbus, microwire, 1-wire, and wireless protocols.

To be viable, the physical layer requirements are summarized as follows: (1) the candidate must support protocols for self-description, discovery, and scalability; (2) the candidate must have the minimal wires necessary; (3) the protocol burden must be light enough to implement with simple processors or state machines.

Most of the candidates fall into a narrower set of classes: asynchronous buses, SPI-class buses, I2C-class buses, and wireless buses.

Asynchronous serial buses include the RS-232, RS-422, and RS-485. Of these, only the RS-485 supports multidrop connections (the others are point-to-point, requiring the creation of a routing infrastructure, similar to that done in spacewire, to permit scaling). RS-485 supports only rudimentary features to resolve ones from zeros, resolve binary words, but not to perform any higher level protocol functions. As such, in order to implement SPA, it would be necessary to do much from scratch, which is possible, but would include reinventing features that existed in other physical layer standards. This possibly violates the third requirement.

The SPI bus (and derivatives, such as microwire) is a popular choice for some memory devices due to speed (tens of megahertz), but requires four wires (not counting power delivery), which violate the second requirement. Scaling involves adding addition lines, one per device, or daisy chaining which reduces fault tolerance and can complicate the overall protocol. Microwire is a restricted implementation of SPI, which is also proprietary.

A variety of wireless protocols exist, including 802.11, Zigbee, and Bluetooth. They remain attractive options for the future, and appear to meet the basic requirements for a candidate physical layer. The drawback is that they do not work unless devices have self-contained power sources or have physical conductors for power delivery. Since wireless protocols also create other potential issues, such as electromagnetic interference, we did not consider them as primary candidates, except for the intriguing cases of devices that can energy scavenge (eliminating the need for *any* wire connections), but we shall not discuss such possibilities further here.

I2C actually covers a number of interface standards (which include SMBus and 1-wire), and is attractive since it only requires two pins (other than power and ground), and supports a very simple addressing and data transfer scheme. While it would be necessary to add address resolution protocols, I2C appeared to offer the most benefit from features offered, ubiquity, and the possibility of having the smallest implementation footprint.

To complete the physical layer specification, it is necessary to specify the signaling levels for the I2C signals, as well as how physical power is delivered. In SPA, synchronization must be supported and provisions for test bypass ideally should be supported. The signaling levels for the I2C pins has been set for 3.3VDC, and the physical power is provided by two separate pins (5VDC), bringing the total number to four. To eliminate the need for additional pins, synchronization and test bypass are handled through provisions in protocol design.

It turns out that we can improve even further, however, and eliminate two more pins by modulating the I2C signals (SDA, SCL) directly onto the power lines, creating a true two-pin plug-and-play technology. We can refer to this interesting variant, which has been demonstrated in the laboratory as mini-plug-and-play, two pin or simply “MP2”. The four-pin version is referred to as “MP4” or simply “MP”.

Connectors

A number of connector concepts have been considered for SPA-1. One candidate is the low-cost commercial Pico-EZMate (Molex) (Figure 7), which has a 1.45mm height and 6mm width (for a four-pin configuration). The recommended pinout at the time of this writing is (1-VDD, 2-SCL, 3-SDA, 4-GND), with 5VDC for power and 3.3V for signal.

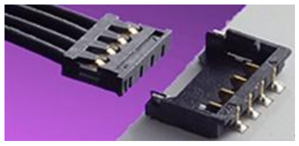


Figure 7. Pico-EZMate connector by Molex (photo from www.molex.com)

Protocol Design

The presumed structure of a MP/SPA-1 network includes a MP master and a number of MP devices, connected in a multi-drop fashion (unless bus extensions are employed). The protocol requirements for MP/SPA-1 devices include support for common functions, device-specific functions, discovery, and (optionally) test bypass.

In each case, messages sent by either the MP master or MP device conform to a simple format, a common binary sequence with a structure having a prefix (header) and suffix (payload). The header contains a byte representing a binary representation of a mnemonic token and a two-byte length field. The payload is 0 or more bytes.

Common Functions. MP/SPA-1 devices must support “common function” commands given by the MP master to a particular device (expect to be universal across all MP devices), including: self-test, reset, initialization, version test, xTEDS download, timekeeping. All common functions are mandatory, with the possible exception of the timekeeping function. A summary of these common functions and the expected responses are summarized in Table 1.

Table 1: Common Functions for mini-PnP(MP)/SPA-1 Devices.

Command	Mnemonic	Response
Reset	R	Status Message
Initialization	I	Status Message
Self-test	T	Status Message
Version	U	Version Message
Time-at-tone	O	Status Message
xTEDS	X	xTEDS Message

Device-specific Functions. Device-specific functions correspond to those functions codified in the electronic datasheet (xTEDS) for a specific device. In general, devices have one or more “interfaces” (discussed in the previous section), and each interface has one or more messages. The messages sent to the device by the MP master can be thought of as having the logical form

MPmaster →Device: IFID.MID(arguments)

where *IFID* is the interface identification (single byte), and *MID* is the specific message identification (also single byte).

Devices must respond to these device-specific commands and requests for data. The requests for data are either "one-time" or recurring. They do not have arguments. Recurring requests are periodically supplied by the device at a time interval specified by the device itself in its xTEDS. Recurring messages can be cancelled as well, through a separate command for that purpose.

Discovery. The hallmark of any plug-and-play approach hinges on the support for discovery or "enumeration". In MP/SPA-1, this support involves: an address resolution protocol (ARP), which determines a unique address for a MP/SPA-1 device; enumeration messages, which test for the existence of a discovered device, and operations pertaining to the self-contained electronic datasheet (xTEDS) (mandatory). Since I2C does not employ a native ARP, one was defined for MP/SPA-1. To facilitate the ARP, a global unique identifier (GUID) is defined. Since the basic I2C standard supports only 127 addresses, there would usually be contention in any pre-determined static assignment of addresses. By using the GUID, overloaded (redundant) assignments can be resolved by testing the GUIDs and detecting mismatches (the I2C method of arbitration allows "0" to win in bus conflicts) and exploiting the GUID to permit device re-assignment. In the end, all devices take on a unique I2C address within the 127 assignments allowed in the standard, even if all devices were initially assigned the same I2C address in the beginning.

Enumerated devices can be tested using an enumerate command, for which devices return a "hello" message. Non-enumerated devices do not say "hello".

Support of test bypass (optional). The test bypass feature is very useful, but has traditionally required four additional pins to support. In larger SPA networks, test bypass connections are separated combined and routed with a distinct physical network. In keeping with a minimalist philosophy, if MP/SPA-1 support test bypass, they must do it without additional pins. In other words, MP/SPA-1 must employ *in-band* test bypass, meaning that the test traffic is superimposed on the same channel used in routine operation. The idea of test bypass is a departure from previous work in test bypass. In the previous work, test bypass interconnections, like JTAG, were physically separated from the signals used in routine operation. It was felt that such separation would lead to more realistic testing, since communications pertaining to test would not load the operational network. In simpler networks, however, the loading may be less of a concern, and easily managed.

MP/SPA-1 Network Operation

MP/SPA-1 networks operate in three phases. The first phase executes address resolution using the previously described ARP. The second phase enumerates devices, initializing them, testing the version identification for consistency, self-testing (if necessary), and reading the xTEDS. The final phase is routine operation, which involves a round robin alternation, cycling through the known (enumerated) devices with a sub-phase consisting of MP master-requested commands or message subscription (as well as cancellation) requests for each enumerated device, followed by a sub-phase involving responses from each enumerated device. A final sub-phase performs new device detection. This sub-phase searches for new devices in the currently unused address space. The alternation between sub-phases and cycling within repeats indefinitely.

Difference Between MP and SPA-1

MP and SPA-1 are very closely related. MP is in fact a generic approach, defined mostly by the I2C overlays discussed in this section. MP need not be space-qualified, need not use SPA middleware, nor (strictly speaking) be used in plug-and-play networks. For use in non-plug-and-play networks, it is only necessary to capture the xTEDS information manually and program the corresponding sequences in a host system. Doing this, of course, obviates the benefits of a plug-and-play approach.

SPA-1, which is otherwise protocol-identical to MP, refers to a space-qualified implementation of MP, to include the domains of knowledge, the use of SDM middleware, the radiation-hardening of components, and qualification of the parts, materials, and processes making it suitable for use in space.

SYSTEM IMPLEMENTATION

Implementation of MP/SPA networks and platforms based on them can be simple if planned carefully. As in the case of the components in other plug-and-play systems (e.g., personal computers), much of the work should have done long before a platform design is commissioned. **One must not confuse the idea of building systems fast with the idea of creating systems that *can* be built fast.** When one needs a keyboard for a personal computer, they do not typically commission a research program to build a keyboard, they simply buy one and plug it in. The act of buying and plugging can unfold in minutes. Month or years before, some company that sold that keyboard implemented an arbitrarily complex research and development program resulting in the keyboard that anyone could buy on demand in the moment it was needed. Given that perspective, MP/SPA systems are only as good as their catalogs, just as operating systems in computers are ultimately only as good as their base of pre-developed applications.

In this section we explore the building blocks needed for effective SPA-based systems. Since we now restrict the discussion to space systems, we may drop mention of "MP" (without loss of generality).

Building good SPA-1 components

In order to build good SPA components, we require good starting materials, namely the spacecraft devices themselves. These may be gyros, reaction wheels, thermometers, batteries, or scientific measurement instruments. We call these "raw devices". There are no predetermined limits for what "raw device" can be made into a SPA device. First, we describe the generic procedure for forming a SPA compliant component. We then describe a methodology that dramatically simplifies the effort otherwise required.

SPA-1 Devices from Scratch. The basic strategy to create a SPA-1 device involves reprogramming some raw device to implement the protocol described in the previous section and rewiring it to conform to the SPA-1 standard[2]. This is often not possible directly, since many devices do not have a built-in I2C interface or (if they do) cannot reprogram the raw device native I2C to be SPA compliant. In this case, it is necessary to introduce an interface board to translate the signals expected by the raw device into a form expected by the SPA-1 protocol. Since the SPA-1 connector expects the device to use 5VDC for its power source, it may also be necessary to convert 5VDC into the voltage(s) expected by the raw device.

It is necessary to prepare an xTEDS specification representing the electronic datasheet to be embedded with the raw device. This can actually be done in simple text editors (like Microsoft *Windows* notepad), if done carefully. Interfaces, commands, messages, must be captured concisely, ideally using a common data dictionary. Mistyping the label is semantic equivalent of falling off a cliff, and will render part of the device invisible to applications looking for it.

Following these steps, one must implement software to run on either the microcontroller resident within the raw device (or failing that, the one in the interface board between the raw device and the SPA interface) that implements each of the messages specified in the xTEDS. Here, as in creating the xTEDS itself, great care must be taken to transcribe the details of the interface correctly. Otherwise, one runs the risk of having a SPA device that operates inconsistently with its own data sheet.

Ideally, these items (the raw device together with any interface circuitry) should be packaged neatly into a compact enclosure exposing the two or four pin connector (corresponding to MP2 and MP4, respectively). If done correctly, a SPA device is formed that can be instantly recognized and operated by a SPA network, platform, or system.

SPA-1 Devices Built with Tools. Another method that can eliminate much of the tedium in creating SPA devices from raw devices involves the use of tools, namely pre-designed interface modules as well as automated code development tools. SPA-1 ASIMs (US) and RTUs (Sweden) take much of the tedium out of building SPA-1 devices. Each contain a microcontroller, non-volatile memory storage and several user input/output terminals, as well as auxiliary functions that are convenient for interface to a wide variety of raw devices.

To assist in the creation of xTEDS and ASIM/RTU code, webtools have been created [10] to allow the formation of xTEDS (linked to a common data dictionary) that are guaranteed correct by construction. Furthermore, the xTEDS description can (at the press of a button) be used to automatically generate the shell of a code system useful for implementing a working SPA device with minimal risk of incorrectly forming the function calls corresponding to the xTEDS associated with the raw device.

Building good SPA networks

The flexibility of I2C offers many prospects for constructing "legal" SPA-1 networks, but there are also constraints. These cases are illustrated in Figure 8. The basic requirement is that a network have a SPA-1 master and one or more SPA-1 devices. Other devices can be added to the same four pins of the SPA-1 bus. This *multidrop* style is shown in Figure 8a. A single cable, punctuated with periodic connectors, can be used to effect the desired connections.

Up to 127 devices can be added to such an arrangement. Chances are that long before that number is reached, the 100pF capacitance limit of the I2C standard would be exceeded. For that reason, it may be desired to use bus repeaters, which employ I2C bus

[2] Consult https://pnpsoftware.sdl.usu.edu/ for the latest information.

repeaters, as suggested in Figure 8b. In their simplest form repeaters can regenerate the bus, but not all commercial repeaters can be daisy-chained, so care must be used (as in any I2C) network. In the SPA version of repeaters, the power lines may be simply passed through or fused.

The next level of sophistication involves creating more intelligent SPA-1 bus devices involving active intelligence. These could include bus isolators, formed for example, by including a SPA device *within* the isolator to accept a command to separate the bus and depower it, which could be a useful strategy in managing problematic devices in a network. The logical extension of the concept is to form a full-fledged SPA-1 router or hub, as suggested in Figure 8c.

It also makes sense, especially in large SPA systems with spatially distributed sub-networks of SPA-1 devices, to form several SPA-1 masters throughout a system, each handling a smaller local network of SPA devices, as suggested in Figure 8d.

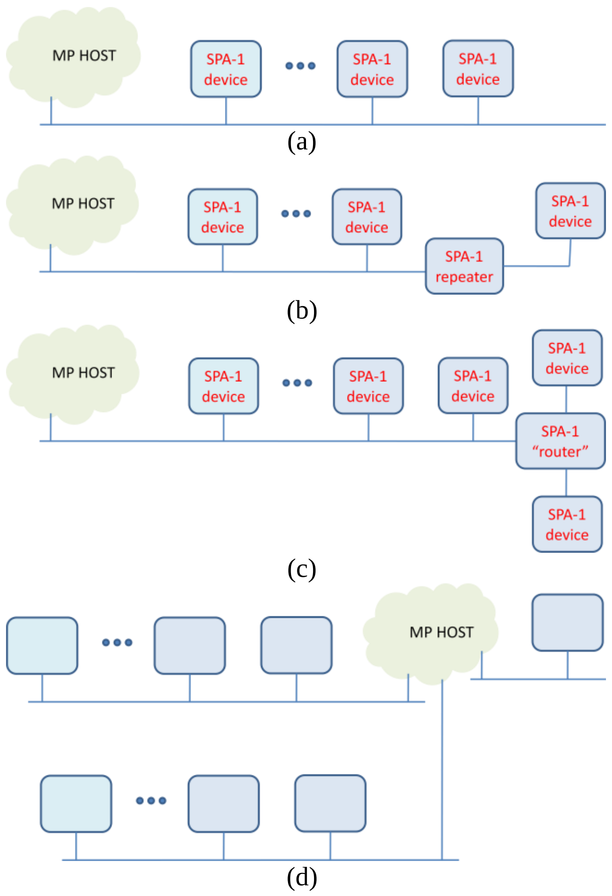


Figure 8. SPA-1 network concepts. (a) Basic multidrop. (b) With bus repeater. (c) With a SPA-1 router/hub. (d) Multiple independent SPA-1 networks.

STATUS AND APPLICATIONS

Practically all significant advancement in the MP/SPA-1 concept occurred after an international agreement between the US and Sweden was signed in August 2009. This marked the beginning of a frenetic and productive collaboration that led to the progress we describe in this section.

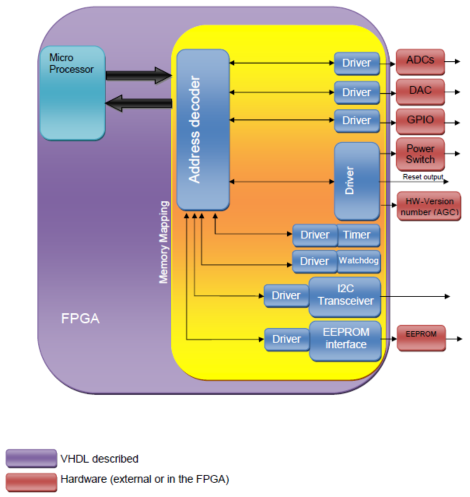
ASIMs and RTUs

No fewer than three interface modules were created since January 2010, when the details of the MP/SPA-1 protocol were worked out and hastily demonstrated on a AT90-based SPA-U "teaching ASIM" from a CubeFlow kit [11]. Each was based on a PIC architecture, either using an actual PIC or a PIC clone. The PIC clones were rendered in FPGA form for validation, with the goal of transferring them into a radiation-hardened structured ASIC technology (90nm).

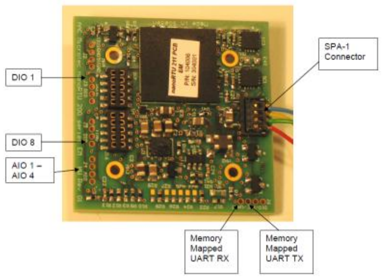
Nano-RTU. The nanoRTU was developed by AAC Microtec to serve as a workhorse platform for evaluation and use of the MP/SPA-1 standard. The heart of the nanoRTU (Figure 9) is implemented in an Actel ProASIC FPGA using a PIC16F84 architecture. It supports the MP/SPA-1 interface, along with analog-to-digital, digital-to-analog convertors and general purpose input/output pins (Figure 9a).

The evaluation board version of the nanoRTU is shown in Figure 9b. This board is 34mm x 34mm and displaces only 25% of the surface of a CubeFlow nanomodular format facet [4]. The evaluation format is very useful for quick demonstration development, concept evaluation, as well as incorporation in actual products. At the time of this writing the nanoRTU had not been evaluated for radiation performance, but data should be obtained for total dose performance in Summer 2010.

The contents of the ProASIC FPGA have also been translated into a format conducive for implementation in radiation hardened structured ASIC technology (discussed in [12-13]). This low-cost structured ASIC technology is via programmable, therefore allowing entire programmations to be defined with a single mask layer. The nanoRTU targets a 3mm x 3mm die size, the smallest in the current family of reticle designs. Using the radiation-hardened die, it should be possible to create a small multichip module version of the nanoRTU approximately 10mm x 10mm, substantially more compact than any radiation-hardened processor that has so far been proposed.



(a)



(b)

Figure 9 nanoRTU. (a) Block diagram. (b) Evaluation version.

SPA-1 ASIM. The second implementation of the MP protocol was carried out in the US by Micro-RDC (Albuquerque, NM) following the scheme shown in Figure 10. This ASIM employed the Wishbone architecture to simplify the hardware realization (at the expense of size and performance). A second I2C interface was implemented (for optional peripherals), along with a SPI interface (for external memory), along with several general purpose input/output lines.

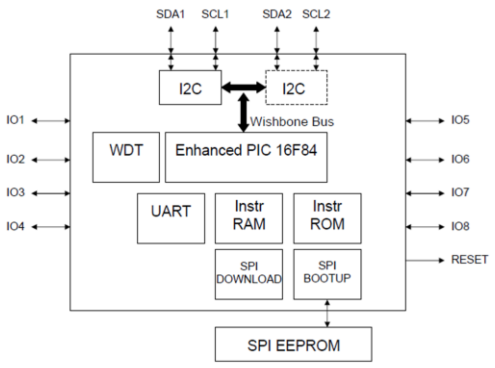


Figure 10. SPA-1 ASIM architecture

As in the case of the nanoRTU, the SPA-1 ASIM is designed to be ported to the 3mm x 3mm rad-hard structured ASIC technology. It has at the time of this writing completed initial implementation on larger evaluation boards where the design is implemented in a Xilinx FPGA.

Commercial MP Design. A third implementation of the MP protocol was developed in-house at AFRL to create a low-cost prototyping aid. The initial design (Figure 10) was quite compact (8mm x 8mm body) in a quad flat package configuration. Work is on-going to improve the manufacturability of this design. It will likely be used in future editions of the Cubeflow training kits used to teach SPA technology.

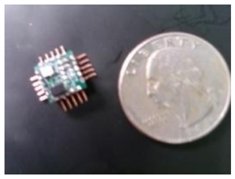


Figure 10. Mini-PnP ASIM based on commercial PIC processor.

RAMPART

The SPA-1 technology is being targeted to several flight opportunities, the first being a small test network of ASIM/RTU modules. To test radiation performance of representative of SPA-1 in space, a small module referred to as "Cricket" is under development for inclusion in the RAMPART cubesat mission [14]. The mission architecture is a very compact enclosure (34mm x 70mm x 10mm) containing an Atmel AT90 process as a SPA-1 master for a network of up to six ASIM/RTUs, specifically the previously discussed design. Since RAMPART is a propulsion demonstration for Cubesats, it will if successful raise its own Apogee from 450km to 1200km, providing a harsher radiation exposure to study the performance of this tiny SPA network.

Trailblazer Series

The Trailblazer (TB) series is a set of Cubeflow-based plug-and-play systems to demonstrate the ability of SPA to be used to quickly assemble fieldable Cubesats. The work is being pursued as a Summer 2010 study, focusing on two 1U Cubesats (TB1 and TB1.5), a 2U Cubesat (TB2) and a 3U Cubesat (TB3). While no firm manifests have been identified for the trailblazers, we hope to exploit the "kit of parts" formed in the exercise for incorporation into future flight projects.

The TB series explores issues pertaining to SPA as a first priority (over missions that might actually be useful to somebody). For example, the TrailBlazer 1.5 concept shown in Figure 11 is a SPA interpretation for a “modern Sputnik”. Completely SPA-1 based the design consists of three modules: a command and data handling system (CDH), a radio module, and a power module. Since these modules are all SPA-1 components, the satellite can be formed with a single four wire harness. Much of the satellite is empty space, and it does not do anything useful except to transmit a beacon signal, but serves to illustrate just how simple a satellite can be (if not simplistic) using a SPA methodology.

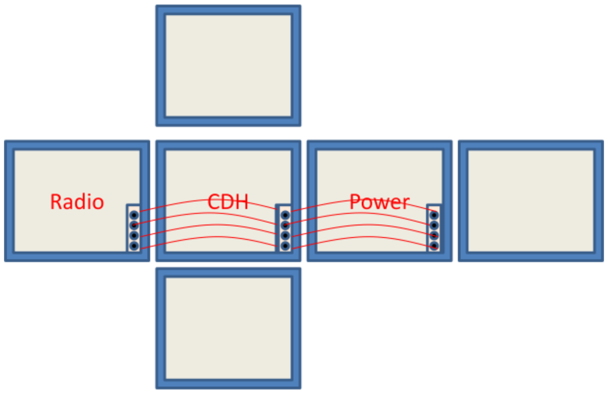


Figure 11. Trailblazer 1.5

QuadSat/PnP

By far the most ambitious SPA-based nanosatellite platform proposed containing SPA is the Swedish QuadSat-PnP 1. The primary impetus for the QuadSat-PnP 1 system architecture is the pervasive use of Space Plug-and-Play Avionics (SPA) standard based on integrating a number of “SPA-ready” avionics building blocks. The components will contain a SPA interfaces, and a number of research circuits relating to power management and distribution concepts will be tested, including latch-up current limiters (LCL). Figure 12 illustrates the QuadSat-PnP 1 architecture, which is based on a combination of SPA-1 and SPA-U components. The architecture as currently envisioned is planned to be single string, without redundancy except for the main power distribution unit.

CONCLUSIONS

While the idea for a minimalist SPA was not new, it was only after our collaboration began that real progress could be made in realizing the vision. In this paper, we discussed the rapid evolution of a new SPA technology, one based on the I2C standard. We believe this format for SPA, derived from the generic mini-PnP technology, represents an easily integrated approach allowing a great variety of simple components to be made “SPA-ready”.

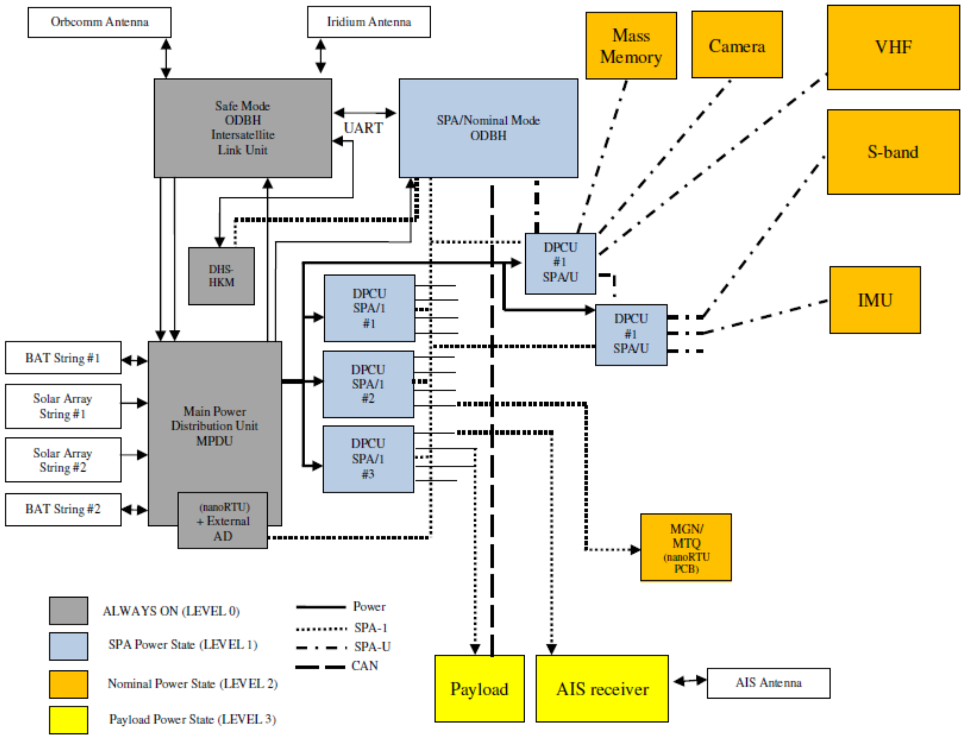


Figure 15. Quadsat-PnP 1

References

1. Marco Cáceres , “Cost overruns plague military satellite programs”, *Aerospace America* (publication of AIAA), January 2006.

2. Lyke, J., Cannon, S., Fronterhouse, D., Lanza, D., and Byers, T. "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, 8-11 August, 2005.

3. Fronterhouse,Don, Martin, Maurice. "Building SPA PnP Satellites", Proceedings of the 7th Responsive Space Conference, April 27-29, 2009, Los Angeles, CA.

4. McNutt, Christopher *et al*. CubeFlow: A Modular Open Systems Architecture for CubeSats, Proceedings of the 7th Responsive Space Conference, April 27-29, 2009, Los Angeles, CA.

5. Bob Brewin (2008-01-22). "Air Force working on cheaper plug-and-play satellites". GovernmentExecutive.com.

6. T. Cooley, T. M. Davis, S. Straight, “ARTEMIS - Advanced Responsive Tactically-Effective Military Imaging Spectrometer: Tactical Satellite 3 for Responsive Space Missions,” Proceedings of the 2006 International Symposium on Spectral Sensing Research (ISSSR), May 31-June 2, 2006, Bar Harbor, MA, USA, URL: http://www.isssr2006.com/TALK_UPLOADS/Thomas-Cooley.pdf

7. P. McGuirk, G. Rakow, C. Kimmery, P. Jaffe, "SpaceWire Plug-and-Play (PnP)", Proceedings of the AIAA Infotech Conference, 7-9 May 2007, Rohnert Park, CA. (paper)

8. Joint Test Action Group (JTAG), “IEEE standard test access port and boundary-scan architecture”, IEEE Standard 1149.1, 1990 and supplement 1149.1-2001, 2001

9. S. Cannon, "Responsive Space Plug & Play with the Satellite Data Model", Proceedings of the AIAA Infotech Conference, 7-9 May 2007, Rohnert Park, CA.

10. Christensen, J., Cannon, S., and B. Hansen, "Automatic Software Generation of ASIM Program Code from an xTEDS", Proceedings of the 2010 AIAA Infotech Conference, 20-22 Apr 2010 Atlanta, GA.

11. Kief, C., Hansen, B., Mee, J., and J. Christensen, "CubeFlow: Training for a New Space Community", Proceedings of the 2010 AIAA Infotech Conference, 20-22 Apr 2010 Atlanta, GA

12. Robert Pugh, David Alexander, James Lyke, Marc Owens,"Abolishing the Oxymoron “Affordable Rad-Hard Electronics”, AIAA SPACE 2009 Conference and Exposition, Pasadena, California, Sep. 14-17, 2009

13. David Alexander, Ken Hunt, Marc Owens, James Lyke, "Affordable Rad-hard-An Impossible Dream?", proceedings of the 22nd Annual Conference on Small Satellites, 11-14 August 2008.

14. Moore *et al.*, “3D Printing and MEMS Propulsion for the RAMPART 2U CUBESAT”, Proceedings of the 24th Annual Conference on Small Satellites (this conference), August 2010