SPACEWIRE PLUG-AND-PLAY: A ROADMAP

Session: Networks and Protocols

Long Paper

Peter Mendham, Albert Ferrer Florit, Steve Parkes

Space Technology Centre, University of Dundee, Dundee, DD1 4HN, UK

E-mail: [petermendham, aferrer, sparkes]@computing.dundee.ac.uk

ABSTRACT

SpaceWire Plug-and-Play has emerged as a topic of widespread interest, with organisations from across the world requiring some plug-and-play features in a wide variety of contexts. The plug-and-play sub-group of the SpaceWire working group developed the principles of a plug-and-play protocol for SpaceWire, with the aim of producing a concrete implementation. The authors of this paper further developed the concept by shifting the implementation to RMAP, providing the ability to leverage existing IP and experience, and simplifying the proposals

The paper begins by reviewing the history of SpaceWire plug-and-play, the current position of the fledgling standard, and its implementation both on new and existing hardware. A key development affecting the future of plug-and-play is the forthcoming SpaceWire-RT protocol. This paper presents a roadmap indicating how plug-and-play will interoperate with SpaceWire-RT, and the migration path that should be taken to ensure maximum functionality and flexibility, whilst retaining support for existing hardware.

1 INTRODUCTION

Since its official publication as an international standard in 2003 [1], SpaceWire has become the *de facto* standard for spacecraft onboard communication, at least within the payload. The rise in popularity of SpaceWire has resulted in a growth in the available instruments, onboard computers and other devices using the standard as the main method of data handling. By constructing equipment that uses SpaceWire for communication, a certain degree of interoperability is ensured (roughly corresponding to the data-link layer of the OSI model). Use of the Protocol ID standard [2] permits identification of standard protocols such as the Remote Memory Access Protocol (RMAP) [3] and a further level of interoperability. However, a SpaceWire network must still be constructed, and configured, carefully for a given application, usually requiring customised software and/or hardware. The lack of standardisation for simple tasks required on almost all SpaceWire networks limits the level of interoperability between devices and software, necessitating custom development for each application and resulting in higher development costs, longer development schedules and greater risk.

In the commercial, non-space, arena so-called 'Plug-and-Play' technologies have been developed to ease the integration of standard components and provide standardised mechanisms with which to carry out common tasks. The past three years have seen the development of a draft Plug-and-Play standard for SpaceWire. This paper discusses the current draft standard, presenting a service-oriented view of the document, highlighting its key features, and the use cases for their application. Given the existing popularity of SpaceWire, it would be difficult to immediately incorporate many of the features of SpaceWire Plug-and-Play (SpaceWire-PnP) into existing systems. The paper therefore presents a method for implementing a subset of services on existing hardware.

Perhaps the single largest imminent change to the way in which SpaceWire is used is the SpaceWire-RT (for Real Time, or Reliable and Timely) standard, which will provide quality of service for SpaceWire. SpaceWire-RT and SpaceWire-PnP are highly related, this paper discusses the connection between these two protocols, and the relationship they will have in systems of the future.

Finally, the paper discusses what the future might hold for SpaceWire systems, and for SpaceWire-PnP in particular, indicating the ways in which SpaceWire-PnP can be adapted and extended to cover new developments.

2 BACKGROUND

As described in a previous paper [4], from the perspective of a user of commercial equipment, application of the term 'plug-and-play' indicates that it should be possible to interface two or more arbitrary devices without the need for configuration. Plug-and-play generally involves two key aspects:

1. Automatic discovery and configuration of hardware and software systems in response to changes in hardware interfacing or availability including whilst the system is running.
2. Detection and configuration of the services that a plug-and-play enabled device provides.

SpaceWire does not offer a standard mechanism for detecting the topology of a network, or what devices are attached to it. Nor does it offer a standard mechanism for configuring the various aspects of a SpaceWire network, such as links and routers. SpaceWire also lacks and standard features to assist detection or configuration beyond the network, in the service domain. The absence of these features became increasingly important as the popularity of SpaceWire grew, and particularly in two application domains: the Operationally Responsive Space (ORS) programme [5] in the USA and the general use of laboratory test and development equipment and electrical ground support equipment (EGSE), especially within the European SpaceWire community.

The history and drivers behind the creation of a plug-and-play protocol for SpaceWire (SpaceWire-PnP), have been covered by the authors in previous papers [4,6] and will not be discussed here.

3 PRINCIPLES OF SPACEWIRE-PNP

3.1 BASIC PRINCIPLES

The central goal of the SpaceWire-PnP standard is interoperability at the network level. As such SpaceWire-PnP provides services to discover, identify and configure the features of a SpaceWire network, as covered by the SpaceWire standard, plus a few more corresponding to only the most common use cases. SpaceWire-PnP does not require devices to support more of the SpaceWire standard than is required to achieve their objectives: if something is optional in the SpaceWire standard, SpaceWire-PnP does not require that it be implemented.

SpaceWire-PnP takes the same view of a network as the SpaceWire standard, being entirely composed of links, nodes and routers. Nodes and routers are both referred to as *devices*. Nodes fall into one of two categories: active nodes, which may discover portions of the network and may attempt to manage other resources; passive nodes, which do not do either and expect to be managed by an active node. All devices have a configuration port (port zero) to which SpaceWire-PnP operations are addressed. Each network device is managed by one active node which is referred to as that device's *owner*. A device's owner may change, but at any given time a device has only one owner. Any node may read the settings of a device, but only the owner may modify them. If a node wishes to modify the settings of a device it does not own, it must do this through the device owner. This is discussed in more detail below.

3.2 NETWORK DISCOVERY AND DEVICE OWNERSHIP

The discovery of a SpaceWire network is driven by an active node. The network is discovered through an iterative exploration of links, routers and nodes, starting with those directly connected to the driving node and working outwards. Simulation work carried out by the authors [4] has determined that in most practical cases, a breadth-first search is fastest and uses the fewest resources. In this type of discovery a node should discover all of the resources attached to a router before attempting to discover what devices are, in turn, attached to those. As an active node discovers the network, it can claim ownership of each of the devices it finds.

A network may be discovered, and owned, by multiple active nodes. The possibility for concurrent discovery and attempts at device ownership requires that claiming ownership is an atomic operation. Claiming a device must identify the owner in such a way that they may be contacted, both to ensure their continued presence on the network and to permit the device to be configured by non-owners. In claiming a node, the claiming operation must atomically set the address (either logical or path) of the owner. If a logical address is used, routing resources may need to be configured to ensure that the owner can be reached. If a logical address is used in claiming a router, the router must automatically configure the routing tables when a device is claimed such that the logical address refers to the port on which the claiming operation took place.

A network with more that one active node presents the potential for competition in device ownership. To resolve a competition situation, each device is assigned a priority. Typically, this priority would be set *a priori* by the system designer such that devices are owned and managed in a deterministic manner. However, SpaceWire-PnP is also designed to deal with the case of a truly open system, in which nothing is known or assigned before devices are added to the network. In this case,

there is the potential for competition between devices with identical priorities. The conflict is resolved by using the physical port number with which the competing actives nodes are using to access the device.

3.3 Device Configuration and Proxy Identifiers

A device may only be configured by its owner. However, many network devices, especially routers, represent a shared resource, and many nodes may have an interest in their configuration. This problem is solved through the use of *owner-proxies*. When an owner claims a device, it atomically sets its address to identify itself. It must also, in the same atomic operation, assign the device a *proxy identifier* which may be used by any other node wishing to configure the device.

When carrying out any discovery or configuration operation the proxy ID is specified. A proxy ID of zero indicates no proxy, or that the discovery or configuration operation is addressed to the same device as the target of the transaction. A non-zero proxy ID allows a node to proxy the discovery or configuration for another device, one that it owns. This permits an owner node to vet requested configuration operations, disallowing ones that may cause issues with the correct operation of the network.

3.4 Principles of Implementation and RMAP

SpaceWire-PnP has a unique protocol ID which clearly identifies the packet as plug-and-play. With the exception of the protocol ID, SpaceWire-PnP uses a 100% compatible implementation of the RMAP protocol [3] for its basic semantics. The RMAP standard gives a great deal of flexibility to implementations, and SpaceWire-PnP standardises many areas that RMAP leaves open. For example, SpaceWire-PnP defines:

1. The RMAP operations that must be implemented: read, verified acknowledged write and RMW.
2. The RMW operation is defined as being a conditional write: the write only takes place if the read returns a value which matches the mask operation (see [8] for a discussion of this).
3. The addressing size and mode: 32-bit wide and incrementing only; all operations must take place on a multiple of 4-bytes. Non-incrementing addressing must be implemented if electronic datasheets, data sources or data sinks are implemented (see below).
4. The device must support a minimum of a 4-byte write and a 16-byte read.

All operations are addressed to the configuration port of a device and the return address should not include the port on the device that is being used, as this is added to the return path automatically, essential for network discovery.

4 SpaceWire-PnP Services

4.1 Service Summary

The facilities that SpaceWire-PnP provides can be grouped together into the following services:

1. **Device Identification and Status.** Provides core information to allow the identification of the device, its type (node/router, and device type field), and vendor information. Additionally, the device status, ownership details and port activity parameters are accessible. Error reporting is also part of this service: SpaceWire-PnP, Protocol ID and SpaceWire-RT error recording can be accessed, if implemented.
2. **Capability Discovery.** Provides a summary of the capabilities of this device. As SpaceWire-PnP is concerned only with interoperability up to the network level, *capabilities* are defined as protocols that this device supports, and the ways in which they can be transported (e.g. using SpaceWire-RT). A device may also use the capability service to provide one or more electronic datasheets, the format of which is identified by a type field but is not standardised by SpaceWire-PnP. This permits ORS xTEDS files [7] to be provided.
3. **Device Ownership.** Provides an atomic mechanism for claiming ownership of a device, identifying and contacting an owner and resolving conflicts.
4. **Owner-Proxy.** Device owners must use the proxy service to provide access to the devices they own. All claimed devices must identify their proxy ID.
5. **Link Status Configuration.** Provides the ability to query the status of any device link and configure its state and speed.
6. **Router Status and Configuration.** This service is applicable to routing devices only. Provides support for the configuration of routing tables and routing mechanisms, as well as time-code propagation.
7. **Time-Code Source.** This service, if implemented, provides a standard method for exposing a time-code source, allowing time-code generation to be enabled/disabled, the frequency of time-codes to be configured, and ticks to be generated manually, if appropriate.
8. **Generic Data Source.** Provides a standard way for a device, such as an instrument, to source data.
9. **Generic Data Sink.** Provides a standard way for a device (such as an actuator) to sink data.

Note that all devices should implement services 1-5, service 6 should be implemented by all routers. Services 7-9 are optional, to be implemented if appropriate.

4.2 Generic Data Sources and Sinks

A device, such as an instrument, may serve the function of sourcing data. SpaceWire-PnP provides a standard mechanism to enable this. A device may implement zero or more data sources which provide data in packets of a bounded size, with the maximum specified by the device. The device also identifies a type, indicating the content of the data. Three different mechanisms are provided by which data can be sourced:

1. Basic reads, where the device provides a data-ready status indicator in addition to a field from which the data can be read. Reads when the source is not ready return an error.

2. Delayed response reads, where the device does not respond to a read until data is ready. A timeout can be specified to ensure that the source responds, even if data is not ready.
3. Initiated RMAP writes, where the device transfers the data as an RMAP verified or un-verified, un-acknowledged write operation, targeting a specified SpaceWire address with a specified local RMAP address.

SpaceWire-PnP also provides a standard mechanism for the implementation of data sinks, such as actuators. Again, a device may implement zero or more data sinks which accept packets up to a specified maximum size. Two different mechanisms are provided by which data can be sunk:

1. Basic (unacknowledged) writes, where the device provides a sink-ready status indicator, in addition to a field to which the data can be written. Writes when the sink is not ready are discarded.
2. Queued acknowledged writes, where the device will accept one or more acknowledged writes. If more than one write is accepted concurrently, the writes are placed in a FIFO queue. If the queue is full, the sink returns an error.

Data sources and sinks most obviously apply to nodes; however, router-driven network discovery, in which a router notifies interested active nodes of changes to link status, can be implemented as a generic data source attached to a routing device.

5 LEGACY SUPPORT IN SPACEWIRE-PNP

SpaceWire-PnP is a new protocol which is transported by a compliant implementation of an existing standard (RMAP). As such, hardware and software for handling SpaceWire-PnP (RMAP) packets already exists and is well proven. However, as the encoding of the various parameters (the RMAP address space) is new, this is not implemented by current hardware, nor is support for the SpaceWire-PnP protocol ID.

In contributing to SpaceWire-PnP, the UoD was able to draw from its extensive experience in designing SpaceWire routers, culminating in the design for the SpW-10X (Atmel device AT7910E). The SpW-10X implements many features that correspond to the core SpaceWire-PnP services. These are implemented using RMAP with a device-specific address space. The SpW-10X supports SpaceWire-PnP services as summarised below:

1. **Device Identification and Status.** Vendor and device ID are provided.
2. **Capability Discovery.** Capability information can be inferred.
3. **Device Ownership.** Limited atomicity is provided by the SpW-10X implementation of RMAP RMW. By combining this with suitable timeouts, the device ownership mechanisms required for SpaceWire-PnP can be implemented.
4. **Owner-Proxy.** The SpW-10X can support the specification of a proxy ID.
5. **Link Status Configuration.** Full link status and configuration is provided.
6. **Router Status and Configuration.** All standard features are provided.

The SpW-10X does not implement generic data source or sink services as these are not relevant. Although the SpW-10X can source time-codes, there is no method for accessing the time-code source via RMAP.

With a different low-level implementation, a SpW-10X-aware version of SpaceWire-PnP can implement all of the core, required services in a standard way, allowing full interoperability with later SpaceWire-PnP devices. Providing the mechanisms are publicly available, any legacy device which supports the core features of SpaceWire-PnP, especially ownership, can be incorporated into a network in this manner.

The Remote Terminal Controller (RTC) (Atmel device AT7913E) is intended as a highly capable processing unit with SpaceWire interfaces (in addition to CAN and MIL-STD-1553 interfaces). The RTC also has a fully-featured RMAP target core; however, this cannot be used for SpaceWire-PnP as it maps the RMAP address space directly onto the host address space without any provision for address translation. The most appropriate way to implement SpaceWire-PnP on the RTC is therefore is software, where a full implementation can be achieved.

6 IMPLEMENTATION OF SPACEWIRE-PNP

By using a compliant implementation of RMAP as the transport for SpaceWire-PnP, the protocol aims the leverage existing hardware and software as much as possible. A device must support two potential differences from a standard RMAP implementation in order to comply with SpaceWire-PnP:

1. SpaceWire-PnP packets are addressed to the configuration port (port zero) of every device, therefore the device must recognise a zero byte at the beginning of the packet. For routers, this is likely to be handled by the routing fabric.
2. Although SpaceWire-PnP uses an implementation of RMAP, it does not use the RMAP protocol ID as this would fail to specify the standard nature of the address space being used. The device must therefore recognise the SpaceWire-PnP protocol ID.

A standard RMAP core (either hardware or software) must therefore be adapted to recognise both the leading zero (if applicable) and the protocol ID. A simple adapter layer can be added before the core to handle these features and provide a flag to users of the RMAP core to indicate that a transaction is for SpaceWire-PnP rather than RMAP. In this manner, an RMAP implementation can support both SpaceWire-PnP and standard RMAP transactions with the same implementation, this means that devices that currently implement RMAP require very little additional logic (again, either hardware or software) in order to support SpaceWire-PnP. For devices that don't currently implement RMAP, the set of RMAP commands that must be supported for a basic implementation of SpaceWire-PnP is sufficiently small to limit the overhead required for SpaceWire-PnP support.

7 SPACEWIRE-PNP AND SPACEWIRE-RT

SpaceWire, as an asynchronous network, does not offer any quality of service (QoS), either in respect to reliability or in respect to timeliness. The forthcoming SpaceWire-RT protocol [8] adds a, largely transparent, layer on top of SpaceWire to provide a full range of qualities of service. These features, along with those provided by SpaceWire-PnP, will form an essential part of the CCSDS SOIS (spacecraft onboard

interface services) SpaceWire sub-network mapping [9]. The relationship between SpaceWire-RT and SpaceWire-PnP is threefold:

1. SpaceWire-PnP may be used on a SpaceWire-RT network which means that active nodes must obey certain rules in order to preserve the QoS provided by SpaceWire-RT.
2. SpaceWire-RT uses SpaceWire-PnP mechanisms to initiate, configure, manage and close *channels*, the virtual end-to-end links over which SpaceWire-RT transports data.
3. SpaceWire-RT can be used to provide reliable and/or timely channels over which SpaceWire-PnP transactions can take place. Whilst RMAP offers very basic reliability in the form of CRCs and replies, these alone are unlikely to be sufficient. SpaceWire-RT offers the *assured* and *guaranteed* services which both provide reliable transport (on an asynchronous or scheduled network respectively)

SpaceWire-RT support can be viewed as a further (tenth) service provided by SpaceWire-PnP.

A channel may be opened by specifying all channel parameters in an acknowledged, verified SpaceWire-PnP write. The channel is successfully opened if a non-error return code is given in the reply. If a channel number is not known *a priori*, status information for all channels can be queried and used to find potentially free channel numbers. A channel may be opened by a node which is neither the source nor the destination. A channel can be closed using a simple SpaceWire-PnP operation which clears all channel information and releases the associated resources.

SpaceWire-PnP provides a simple table which lists the status of all open SpaceWire-RT channels. This permits an active node to check all open channels for errors, and to verify the presence and correct operation of channels. Additionally, a status query can include requests from the device being queried for SpaceWire-RT channels to be opened on its behalf. This permits both passive nodes (and routers), and active nodes not assigned management bandwidth on a scheduled network, to request channels with appropriate QoS. Once the request has been satisfied, a network node manager can write back to the requesting node with the new channel number.

On joining a SpaceWire network, an active node must account for the fact that the network may be using SpaceWire-RT, and it may be scheduled. To ensure scheduling rules are not broken, a new node must wait before commencing network discovery, or any other SpaceWire-PnP operation. If, during that time, a time-code is received, the node must wait until timeslot zero, which is reserved for management operations. At that point the node may proceed to discover the network. On an asynchronous (un-scheduled) SpaceWire-RT network, SpaceWire-PnP may proceed as normal.

8 THE FUTURE

Through its various drafts, SpaceWire-PnP has evolved to cover the most important features of a SpaceWire network, including some of the key use cases, such as data sources and data sinks. A guiding principle has been efficiency, which would be improved by including a standard mechanism for the exchange of topology information between those nodes carrying out network discovery. SpaceWire-PnP is

well positioned to take account of changes including any additions. One possible such addition is the real-time signalling mechanism proposed by Sheynin *et al* [10].

9 CONCLUSIONS

The SpaceWire-PnP draft standard has come a long way since it was created as part of the work of the SpaceWire Working Group plug-and-play sub-group. The proposals in this paper seek to provide a highly flexible and extensible standard which leverages existing technology and provides appropriate opportunities for the support of legacy systems such as the sophisticated SpW-10X router and the RTC device. This is accomplished using an implementation of RMAP as a transport and providing standard mechanisms to permit the discovery and management of network resources by multiple nodes simultaneously in an interoperable manner.

The standardisation and wide-spread implementation of SpaceWire-PnP will provide the basis for device and vendor interoperability, lowering spacecraft development time, costs and risk, and expanding the market for SpaceWire devices.

10 REFERENCES

1. ECSS, "SpaceWire - Links, nodes, routers and networks", ECSS-E-50-12A, 24th January 2003.
2. ECSS, "Protocol Identification", ECSS-E-50-11 Draft F, 4th December 2006.
3. ECSS, "Remote Memory Access Protocol", ECSS-E-50-11 Draft F, 4th December 2006.
4. Mendham *et al*, "Plug-and-Play Technology for SpaceWire: Drivers and Alternatives", IAC-07-B4.7.06, Proceedings of the 58th International Astronautical Congress, Hyderabad, India, 2007.
5. US DOD, "Plan for Operationally Responsive Space", A Report to Congressional Defense Committees, National Security Space Office, USA, 17th April 2007.
6. Mendham *et al*, "Interoperability And Rapid Spacecraft Development using SpaceWire Plug-And-Play", IAC-08-B4.7.06, Proceedings of the 59th International Astronautical Congress, Glasgow, UK, 2008.
7. Lyke *et al*, "Space Plug-and-Play Avionics", Proceedings of the 3rd Responsive Space Conference, Los Angeles, CA, USA, 2005.
8. Parkes and Ferrer-Florit, "SpaceWire-RT Initial Protocol Definition", Draft 2.1, SpaceNet Report No. SpW-RT WP3-200.1, ESA Contract No. 220774-07-NL/LvH, October 2008.
9. Fowell and Taylor, "The SOIS Plug-and-Play Architecture and its Proposed Mapping onto SpaceWire", Proceedings of Data Systems in Aerospace, DASIA 2008, Palma de Mallorca, Spain, 2008.
10. Sheynin *et al*, "Real-Time Signalling in SpaceWire Network", Proceedings of the 1st International SpaceWire Conference, Dundee, UK, 2007.