Network Discovery Protocols

USER REQUIREMENTS

SPACEWIRE PLUG-AND-PLAY PROTOCOL

Reference:	SSL/08717/DOC/001
Issue:	1.2
Date:	26/02/2012

SciSys UK Ltd.
Methuen Park
Chippenham
Wiltshire
SN14 0GB
United Kingdom

Tel: +44 (0)1249 466466
Fax: +44 (0)1249 466661

www.scisys.co.uk

Intentionally Blank

DOCUMENT CONTROL

This document has no controlled or maintained paper copies. The master document is held as the Microsoft Word file "DOC 001 SpaceWire Plug-and-Play Protocol Requirements v1.2.doc" in the Network Discovery Protocols project directory on the SciSys in-house computer network. All files have widespread access for reading only.

Changes to the document will be notified to its audience.

	Name	Function	Date
Prepared by:	Peter Mendham, Philippe Fourtier, Steve Parkes	Authors	26/02/2012
Checked by:	Stuart Fowell	Project Manager	29/02/2012
Approved by:	Stuart Fowell	Project Manager	29/02/2012
Authorised by:	Allan Pascoe	Operations Manager	29/02/2012

ISSUE RECORD

Issue	Issue Date	Sections Affected	Relevant Information
1.0	28/10/2011	All	Initial issue
1.1	16/01/2012	All	Added requirements to skeleton and correct minor errors
1.2	29/02/2012	All	Review comments incorporated, all stakeholders requirements added and collation carried out

Intentionally Blank

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

None

1. INTRODUCTION

1.1 Purpose and Scope

The SpaceWire standard defines the aspects of a highly flexible and capable communication system; however, SpaceWire does not offer a standard mechanism for discovering 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 standard features to assist detection or configuration beyond the network, in the service domain.

This lack of standardisation for simple tasks that are required on almost all SpaceWire networks limits the level of interoperability which may exist between devices and software, and the extent to which both hardware and software can be re-used between different applications.

1.1.1 Scope of the Project

The main objective of the Network Discovery Protocols activity is to develop, validate and demonstrate a protocol to permit SpaceWire network discovery and management in a standard and interoperable manner, offering 'plug-and-play' features for SpaceWire.

The goals of the project are:

- to gather requirements for a SpaceWire plug-and play protocol;
- to design and specify a SpaceWire plug-and play protocol;
- to develop an implementation of the protocol encompassing hardware and software;
- to provide a test bench, for validation of the protocol;
- to provide a demonstrator to permit demonstration of protocol features.

1.1.2 Scope of the Document

This document gathers requirements for the SpaceWire plug-and-play protocol. There are four different sources for these requirements:

- Thales Alenia Space, taking the role of a system integrator;
- SciSys, taking the role of a software supplier;

- STAR-Dundee, taking the role of an equipment supplier;
- ESA requirements from the Statement of Work [AD7] plus input from the SpaceWire working group representing end customer requirements.

Once requirements have been gathered from these sources they can then be collated into a single requirements list.

When complete, this document corresponds to deliverable D1 of the Network Discovery Protocols activity: the Protocol User Requirements.

1.2 Executive Summary

This document gathers requirements on a SpaceWire plug-and-play protocol for the Network Discovery Protocols activity. When complete, this document corresponds to deliverable D1 of the Network Discovery Protocols activity: the Protocol User Requirements.

After this introduction the document has six further sections:

- Section 2 outlines the terms of reference for the requirements, describing the intended scope of the activity;
- Section 3 gathers requirements from the perspective of a system integrator, provided by Thales Alenia Space;
- Section4 gathers requirements from the perspective of a software supplier, provided by SciSys;
- Section 5 gathers requirements from the perspective of an equipment supplier, provided by STAR-Dundee;
- Section 6 gathers requirements from the perspective of a customer, in this case requirements are taken from the ESA statement of work for this activity [AD7] together with inputs from the SpaceWire community via the SpaceWire Working Group meetings archive;
- Section 7 collates all requirements into a single list.

1.3 Applicable and Reference Documents

1.3.1 Applicable Documents

AD1 ECSS-E-ST-50-12C, "SpaceWire – Links, nodes, routers", 31 July 2008.

AD2 ECSS-E-ST-50-51C, "SpaceWire protocol identification", 5 February 2010.

AD3 ECSS-E-ST-50-52C, "SpaceWire - Remote memory access protocol", 5 February 2010.

AD4 ECSS-E-ST-50-53C, "SpaceWire - CCSDS packet transfer protocol", 5 February 2010.

AD5 ECSS Drafting Template with Explanations and Examples, June 2009.

AD6 ECSS Standard Template Version 5.6, August 2010.

AD7 ESA, Statement of Work, Network Discovery Protocols (SpaceWire Plug-and-Play), TEC-EDP/DJ/2010/SoW/02, Issue 1, 4 November 2010.

1.3.2 Reference Documents

RD1 SpaceWire-PnP Protocol Definition, Draft A Issue 2.1, 16th September 2009, Peter Mendham, Space Technology Centre, School of Computing, University of Dundee

RD2 "SpaceWire Plug and Play: A Roadmap", International SpaceWire Conference, November 2008, Peter Mendham, Albert Ferrer Florit, Steve Parkes

RD3 "SpaceWire Plug-and-Play: Fault-Tolerant Network Management for Arbitrary Network Topologies", International SpaceWire Conference, September 2007, Albert Ferrer Florit, Martin Süss

RD4 "SpaceWire Plug-and-Play: An Early Implementation and Lessons Learned", AIAA Infotech@Aerospace 2007 Conference and Exhibit, May 2007, Barry M Cook and C Paul H Walker, 4Links Ltd

RD5 "ESA and NASA requirements on SpaceWire PnP", March 2007, ESA & NASA

RD6 "PnP aspects, ESA contribution", 8th SpaceWire Working Group, January 2007, Albert Ferrer Florit, ESA/ESTEC

RD7 "PnP aspects, 4Links contribution", 8th SpaceWire Working Group, January 2007, Barry Cook, Paul Walker, 4Links Ltd.

RD8 "SpaceWire-D", Deterministic Control and Data Delivery Over SpaceWire Networks Draft B, April 2010, Steve Parkes

RD9 "The Operation and Uses of the SpaceWire Time-Code", International SpaceWire Seminar, 2003, Steve Parkes, Space Systems Research Group, University of Dundee

RD10 UoD_SpW_10X_DataSheet_Issue-2.0, 13th October 2006, Chris McClements & Steve Parkes, University of Dundee

RD11 SpW-10X SpaceWire Router User Manual, Issue 3.1, 20th January 2008

RD12 SpaceWire-RTC Datasheet 1.8, Dec 2008, Karl Engström & Sandi Habinc (Aeroflex Gaisler), Peter Sinander (RUAG Space)

RD13 SpaceWire Remote Terminal Controller User's Manual RTC-100-0012 Version 2.4, December 2009, Aeroflex Gaisler & RUAG Space

RD14 SMCS116SpW Data Sheet rev.A, July 2007, ATMEL

RD15 SMCS116SpW User Manual rev1.0, July 2007, P. Rastetter

RD16 SMCS332SpW Data Sheet rev.B, May 2008, ATMEL

RD17 SMCS332SpW User Manual 1.5, 10 July 2007, U. Liebstückel

RD18 http://www.eclipse.org/

1.3.3 Subordinate Documents

None.

1.4 Terms, Definitions and Abbreviated Terms

CRC	Cyclic redundancy check
EGSE	Electrical Ground Support Equipment
ETE	end-to-end
FDIR	Fault Detection, Identification/Isolation and Recovery
GAR	Group Adaptive Routing
GDIR	General Design and Interface Requirements
GPS	Global Positioning System
ICU	Instrument Control Unit
PDHT	Payload Data Handling Terminal
PnP	Plug-and-Play
PPS	Pulse-per-second
PTP	(CCSDS) Packet Transfer Protocol
PUS	Packet Utilization Standard

QoS	Quality-of-Service
RMAP	Remote Memory Access Protocol
RW	Reaction Wheel
SbP	SpaceWire-based Protocol
SOIS	Spacecraft Onboard Interface Services
SpW	SpaceWire
SpW-D	SpaceWire-Deterministic
STR	Star Tracker
TM	Telemetry

1.5 Open Issues and Assumptions

1.5.1 Assumptions

None.

1.5.2 Open Issues

The following items are TBD:

- Req PnP-SYS-FN-304 (page 25)
- Req PnP-SYS-FN-403 (page 26)

End of Section

2. TERMS OF REFERENCE

This section describes the terms of reference for the generation of requirements on a SpaceWire plug-and-play protocol by all three stakeholders. The discussion in this section is intended to act as guidance for requirement elicitation, not so as to limit the scope of the requirements, but to ensure that the requirements have sufficient coverage to permit design of a plug-and-play protocol.

The baseline for requirements is taken as those presented in the Statement of Work [AD7]. The following subsections address groups of requirements, presenting key areas for consideration. Where a number of different options are discussed, these are not intended to be exhaustive or complete. Stakeholders should feel at liberty to describe requirements outside the scope of the discussion here, which is intended as a starting point only.

2.1 Functional Requirements

The core part of this activity is the design of a plug-and-play protocol which permits the discovery of devices on a network and the determination of any changes to the network over time. This device and topology discovery might be carried out using one of a number of techniques. Indeed, it might be appropriate for the protocol to support more than one.

On other communication technologies, such as buses where broadcast is possible, the presence of a new device may be detected through an announcement from the device itself. Whilst announcement permits the identification of device additions, clearly the technique cannot be further applied to device removal. In order to detect both addition and removal an unaffected part of the network can be responsible for detecting and announcing changes. In the case of a technology which utilises point-to-point links through routers, the router can be responsible for detecting the addition or removal of a device and making the announcement on behalf of the device. This is sometimes referred to as notification. Where neither announcement nor notification is feasible, it may be possible for an interested device to determine the topology of the network through discovery. In this case a single device, acting as a master, progressively discovers attached devices through probing the network using, for example, a tree algorithm. Discovery is most applicable where the technology does not have provision for announcement or notification.

The ability to refer to and communicate with discovered devices often requires the use of a device identifier which is guaranteed to be unique for the complete network whatever changes may take place. A device identifier can arise from one of two sources: either it is pre-assigned before the device is discovered (i.e. static); or it is assigned as part of the device discovery process (i.e. dynamic). There are various advantages and disadvantages to each option. Many network technologies choose the static option (*c.f.*

Ethernet MAC addresses) but the appropriateness of this option in the context of SpaceWire must be determined by the stakeholders. Again, it might be necessary to support multiple options in the protocol.

The functionality of the SpaceWire plug-and-play protocol must be driven by its expected use cases. As such, what information can and should be provided by a discovered device, as well as what device functions can be configured, should be carefully considered for both nodes and routers.

The requirements in the Statement of Work include the concept of generic data sources and data sinks which can be used to introduce a standard mechanism for sourcing and sinking data. Again, such features should be driven from expected use cases.

2.2 Coverage Requirements

SpaceWire offers the ability to construct arbitrary topology networks, with careful use of devices and links to provide redundancy. The expected use cases of network topologies and redundancy should drive the requirements on the SpaceWire plug-and-play protocol. Redundancy does not only apply to 'passive' devices such as sensors and actuators, but also to those 'active' devices which wish to discover network topology. To what extent 'active' device redundancy is required must be determined by the stakeholders.

2.3 Design, Performance and Resource Requirements

Each stakeholder must also consider the necessary performance of the SpaceWire plug-and-play protocol on the basis of expected use cases. It may be useful to impose resource requirements on the expected implementations of the protocol in order to limit the complexity of the design.

Depending on the expected use cases for the protocol, it may also be necessary to consider compatibility with existing protocols and/or devices in the design of the SpaceWire plug-and-play protocol.

End of Section

3. SYSTEM INTEGRATOR REQUIREMENTS

3.1 Context

Avionics architectures are composed of physical units, all interconnected through different kinds of links, using various protocols. SpaceWire is one of these embedded technologies for board-to-board communication. Characterized by high data rates, it allows building actual networks, by implementing routing capabilities.

This opens the door for embedding ground technologies as is the PnP, which greatly enhance the re-use of designs from one mission to another, enriching each time the number of known units.

Spacecraft are usually composed of a platform (service module) and instrument(s) (payload module). Up to now, those are clearly separated and have their own communication networks. The service module includes the Command and Control interface with ground, who can control the payload module through that indirect way.

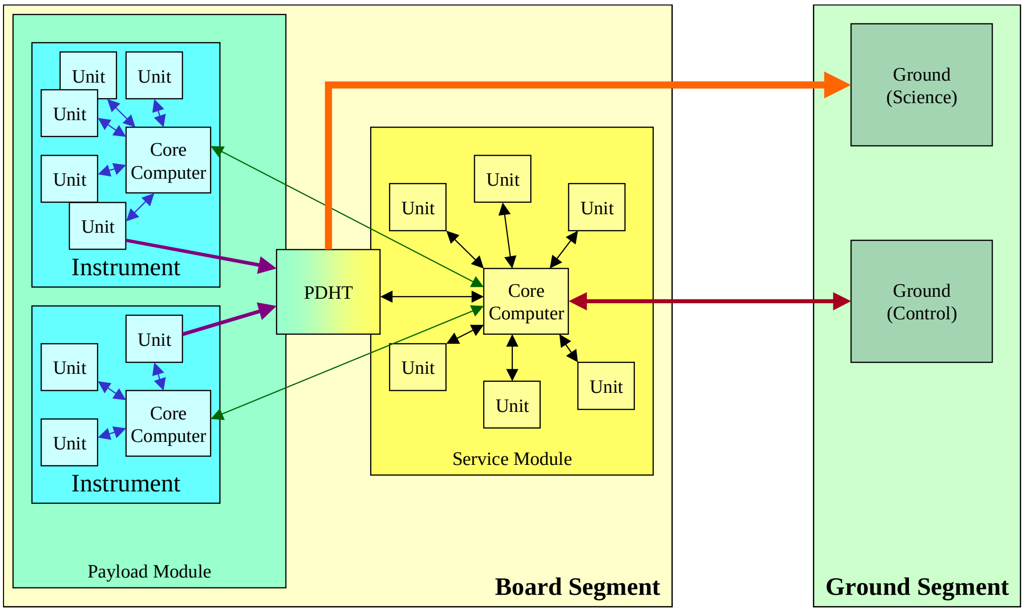

Figure 3-1: Usual architecture of a space mission (science/ observation)

Three kinds of networks appear, centralized on different physical units.

The "*service*" network: it involves all the platform units, with the platform core computer as data flow concentrator. The herein units have always similar roles and characteristics from one mission to another. They are sensors and actuators dedicated to service applications as AOCS, active thermal control, power control. They are often duplicated, to create redundancy, and cross-strapped, which makes several operational configurations of units for that network. Data are exchanged in all directions and are critical for most of them (i.e. high level of QoS, ETE delays to be strictly respected). This network is working all along the mission's lifetime.

The "*payload*" network: it involves all the instruments (instrument Data Handling computers), with the platform core computer as data flow concentrator. This network is aimed at supporting the execution of the mission plan and controlling the payload module. It can also cover science data transfers when these flows remain in a reasonable volume. One can consider families of payloads, thanks to the kind of measurements done (observation, interferometry …), but instruments are most of the time very different in term of Command/Control from one mission to another, except in case of series. They are more often cross-strapped and can be redundant, which makes several operational configurations of units for that network. Data are exchanged in all directions and have a lower criticality than for the service network. This network is working all along the operational lifetime (which excludes some contingengy cases).

The "*storage*" network: it involves all the instruments (instrument Data Handling computers or data processing computers), with the PDHT (mass storage unit) as data flow concentrator. This network is aimed at routing the science data to the mass memory, waiting for restitution to ground. Data streams are basically unidirectional and have a low criticality thanks to control networks. This network is working all along the operational lifetime (which excludes some contingency cases).

Instruments local network(s): there is one per instrument, involving all instrument's units, with the instrument Data Handling computer as data flow concentrator. These networks are aimed at executing the instruments' activities. The units involved are mostly specific, even if common functional bricks can be considered (Cryo cooler, scan mechanisms, detectors and electronics, calibration subsystems etc…). They can be duplicated, for redundancy, but generally not cross-strapped. Data are exchanged in all directions and can have high criticality, especially in term of time constraints (synchronized activities). This network is working all along the operational lifetime (which excludes contingency cases).

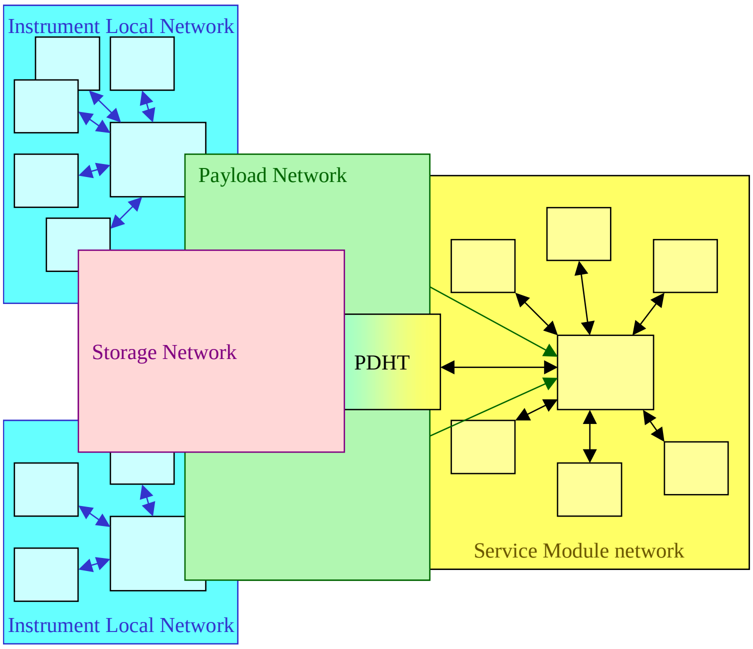

Figure 3-2: The different embedded networks (illustration)

3.2 SpaceWire

SpaceWire is more and more often used, especially in the "storage" network for now. But the trend is to extend its use to other parts of embedded communications (i.e. the other networks identified in previous paragraph). The question is raised of network sharing critical and non-critical data.

The following assumes that SpaceWire use will spread out to all onboard communications and all units (which is not obvious today). In that frame, a short analysis can be led on units involved in each network, to evaluate their opportunity to be re-used. This re-usability level is one of the criteria which raises the interest of implementing a PnP protocol on board, such to also re-use qualified software and units from one mission to another. Device recognition supposes that units have to implement PnP too, which will require development efforts on mean or long term to adapt off-the-shelf units. Hence, networks should be able to include PnP and non-PnP units.

3.3 Network Characteristics, Thanks to PnP

The "*service*" network looks to be the best candidate for PnP implementation, as avionics is composed of units which are generally off-the-shelf actuators and sensors, re-used in many missions. Furthermore, several models are often used in a platform, for redundancy or functional reasons (RW, MAG, STR, GPS …).

The "*instrument local*" network involves various units, often specific. Nevertheless, units having similar functionalities appear in several missions, as cryo-coolers, scan mechanisms, detectors, imagers, data processing units etc.

The other networks (*payload* and *storage*) include nodes exchanging the same kind of messages, supported by a common mission protocol, imposed by a GDIR document.

3.4 Definitions

The definitions listed here apply to the System Integrator Requirements only.

A **Physical Unit** is any box which is physically connected to the network through one or several SpW links. A physical unit can include one or several functional unit(s).

A **Functional Unit** is any function connected to the network.

A **Router** is a unit, whose purpose is to route packets to other nodes

A **node** is any unit, identified or not, which is an end-terminal (i.e. packets source and/or consumer).

A **Peripheral Device** is any Node which has been successfully identified via the PnP protocol.

A **Control Device** is any Node which gets capability to identify devices over the network, via the PnP protocol.

Note: A **physical unit** can include several **functional units** as **router**(s) and **node**(s) / **device**(s). SMCS332SpW is an example where a single component includes 3 SpW ports, which can be configured as router or node.

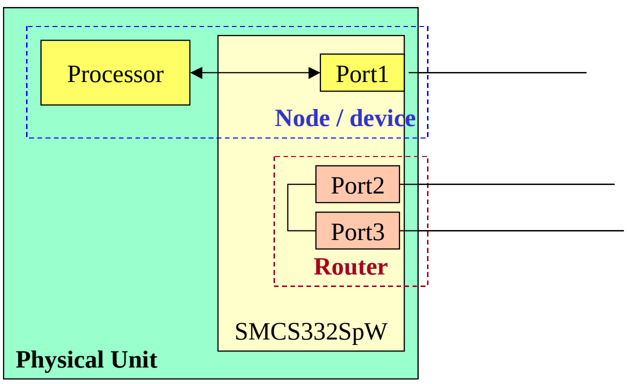

Figure 3-3: Example of units, node, router ...

3.5 Requirements

3.5.1 Functional Requirements

3.5.1.1 Compliance with Standard

Req PnP-SYS-FN-001

The PnP protocol shall be compatible with the SpW standard:

- Link connection,
- Cargo structure:
    - addressing (logical or physical),
    - Protocol Id.

#

Req PnP-SYS-FN-002

The PnP protocol shall not preclude usage of other protocols for functional exchanges between devices (RMAP, SpW deterministic, others …), in particular it shall be compliant with further PUS communications between devices.

#

Req PnP-SYS-FN-003

The PnP protocol shall work whatever the way used to reach a node (GAR allowed).

#

Req PnP-SYS-FN-004

The PnP protocol shall still work, even after modifying routers configuration.

#

Req PnP-SYS-FN-005

The PnP protocol shall not depend upon nodes' address values.

#

Req PnP-SYS-FN-006

The PnP protocol shall not be disturbed by the time-code processing (emit or receive).

#

3.5.1.2 Configuration

Req PnP-SYS-FN-101

The PnP protocol shall allow to configure a node as a **control device** or as a **peripheral device**.

Note: An ICU (instrument control unit) can behave as a **peripheral device** within the **"payload" network**, and as a **control device** in the **instrument local network**.

#

3.5.1.3 Control Devices

Req PnP-SYS-FN-201

The PnP protocol shall provide services, allowing a **control device** to detect any **node** joining the network and to try to identify this **node** as a known **device**.

#

Req PnP-SYS-FN-202

It shall be possible to add new drivers in the library of **peripheral devices** embedded in a **control device**.

#

Req PnP-SYS-FN-203

It shall be possible to configure the library of **peripheral device** drivers embedded in **control devices** at SW generation time.

#

Req PnP-SYS-FN-204

The PnP protocol shall not preclude use of off-the-shelves equipment (**peripheral device**) which does not implement PnP.

Note: The PnP must not result in stopping activity of a control device if a node is not identified as a known device.

#

Req PnP-SYS-FN-205

The PnP protocol shall inform local application, any time another node joins the network (i.e. connects), providing at least its address in the network.

Note: Any detection of node/device can then be notified in TM (event report).

#

Req PnP-SYS-FN-206

The PnP protocol shall automatically try to identify any node joining the network (i.e. at connection time).

#

Req PnP-SYS-FN-207

PnP communications with connecting node shall stop after successful or failed identification of that node.

> Note: From that time, the communication with the node is opened, using SpaceWire standard, then through the identified driver or through mission specific processing.

#

Req PnP-SYS-FN-208

Successful or failed node identification shall be notified in TM.

#

Req PnP-SYS-FN-209

The PnP protocol shall provide a way for the **control device** to bypass the identification step (when the node keeps memory of addresses of already identified devices, once disconnected).

> Note: In flight, there is no reason why devices should change. Hence, they could be identified at initialization time, never after, so saving time when setting them in use for a second time or in case of reconfiguration. Identification is done each time there is no such data saved.

#

3.5.1.4 Detection of Devices

Req PnP-SYS-FN-301

The PnP protocol shall provide a service, allowing any **peripheral device** to provide its identification over the network, to **control device**.

> Note: More and more devices (new developments) are required to be PUS terminals (GPS, instruments, mass memory), except for off-the-shelves units or recurrent models (STR, GPS, instruments).

#

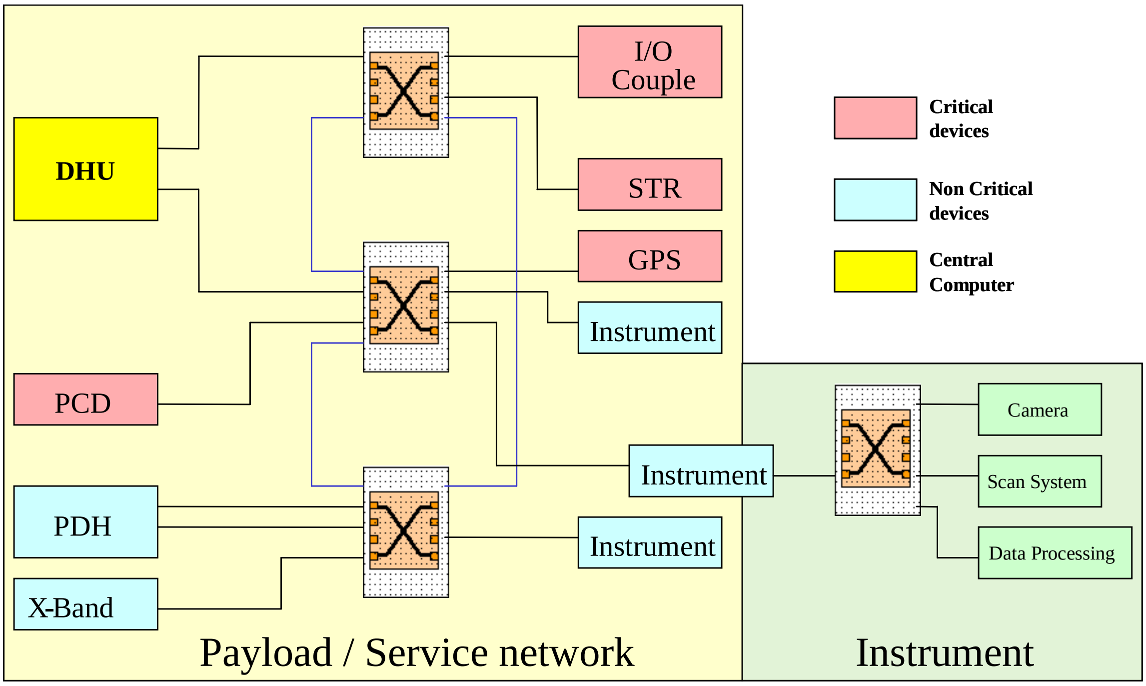

Figure 3-4: Example of common "Payload" and "Service" networks

Req PnP-SYS-FN-302

When identification fails, PnP protocol services shall declare the node as UNKNOWN DEVICE.

#

Req PnP-SYS-FN-303

The PnP protocol services shall allow to report any device identification in TM.

#

Req PnP-SYS-FN-304

The device identification service of the PnP protocol shall not last more than TBD msec (nodes contribution excluded).

#

Req PnP-SYS-FN-305

It shall be possible to configure a delay before first identification attempt (such to cover initialization phases of nodes, where those are not yet listening to communication ports).

#

Req PnP-SYS-FN-306

It shall be possible to configure the number of identification attempts (such to cover initialization phases of nodes, where those are not yet listening to communication ports).

#

3.5.1.5 Disconnection of Devices

Req PnP-SYS-FN-401

The PnP protocol shall detect nodes/devices disconnection.

#

Req PnP-SYS-FN-402

The PnP protocol shall allow to report any node/device disconnection in TM.

#

Req PnP-SYS-FN-403

The disconnection detection at PnP protocol level shall not last more than TBD msec (SpaceWire contribution excluded).

#

3.5.1.6 Redundancy

Req PnP-SYS-FN-501

The device detection function of the PnP protocol shall provide a unique identification of peripheral devices, even though if those are identical units (same family, same type etc.)

#

3.5.2 Coverage Requirements

Req PnP-SYS-CV-001

The use of PnP protocol shall not constrain the network topology.

#

Req PnP-SYS-CV-002

The PnP protocol shall support several identical peripheral devices within a network (i.e. devices sharing the same driver).

#

Req PnP-SYS-CV-002

The control device shall be able to execute the PnP in parallel with nominal communications with other devices (using any RMAP, PTP, PUS or others).

#

3.5.3 Performance Requirements

Req PnP-SYS-PF-001

The PnP protocol shall not lead to endless waiting delays (i.e. shall always complete or abort).

#

3.5.4 Design Requirements

N/A

3.5.5 Interface Requirements

N/A

3.5.6 Environmental Requirements

N/A

3.5.7 Physical and Resource Requirements

N/A

3.5.8 Operational Requirements

N/A

3.5.9 Human Factors Requirements

N/A

3.5.10 Logistics Support Requirements

N/A

3.5.11 Product Assurance Requirements

N/A

3.5.12 Configuration and Implementation Requirements

N/A

3.5.13 Verification and Testing Requirements

N/A

End of Section

4. SOFTWARE SUPPLIER REQUIREMENTS

This chapter presents requirements on a SpaceWire plug-and-play protocol from the perspective of a software supplier. These requirements arise principally from experience of developing onboard software; however, requirements arising from the development of EGSE, test and support software are also considered.

4.1 Use Cases

Requirements are derived from a set of six use cases each of which corresponds to a common software task. The use cases are listed below in priority order, with the first use case being the highest priority, as it is the most common. In each case, the requirements arise from the need to perform a particular use case in a standard manner. With current technology each use case either requires bespoke development for a given mission/application, or is not possible at all. A standard mechanism for each use case would save considerable time, effort and cost in software development and validation.

4.1.1 Device Presence and Network Topology Confirmation

When initialising the I/O tasks onboard a spacecraft it is necessary to confirm the presence of the required devices. Typically a list of expected devices is available, together with their position in the overall network topology. The presence and position of each device must be confirmed at least once, but it may be appropriate to repeat the process after a significant reconfiguration, such as a mode change. Where a mode change results in the powering down of a device, it may also be important to confirm that a device is *not* present.

Additionally, some devices provide a summary status value which may be used to determine the correct operation of a device. Usually the access to this status value involves a device-specific mechanism.

4.1.2 Network Configuration

SpaceWire networks are extremely flexible, with many devices supporting a number of configurable features. Many of these features are identified in the SpaceWire standard including:

- link transmission speeds (after start up);
- link state control (e.g. start/autostart);
- routing table configuration;

- time-code generation and handling.

The configuration mechanisms for each of these features, if present in a given device, are device-specific.

4.1.3 Mission/Network Policy Implementation

The use of a SpaceWire network onboard a spacecraft presents a number of design decisions in how the network is used. These decisions effectively become part of the network policy for a given mission. In each case, the application of policy involves the configuration of one or more network devices. If the presence of network devices is being determined then network accesses and configuration will already be being performed. In a number of cases some configuration, such as link speed, may be necessary to support the process of device presence confirmation. By enabling network policy to be specified through a suitable interface the user of the plug-and-play protocol can ensure that all configuration is done according to policy and that network configuration operations are carried out efficiently.

Elements of network configuration which form part of network policy are the following:

- the choice of path or logical addressing;
- logical address assignment (if in use) and the subsequent configuration of routing tables;
- link mode, such as the use of start and autostart modes as well as features such as 'start on request' and 'disable on silence' for power management;
- link speeds, including for power management.

Whereas the mechanisms for discovery, configuration and management can be generic for any SpaceWire network the policy will necessarily change from mission to mission. It is therefore important that a SpaceWire plug-and-play protocol support the clear separation of policy from mechanisms and, ideally, provides a clear interface to support the implementation of network policy.

4.1.4 Acquiring from Sensors

The most common activities for an onboard SpaceWire network are acquiring data from data source devices, such as sensors or instruments, and supplying data to data sink devices, such as actuators.

Mechanisms for acquiring data are, in general, device-specific but follow one of a number of predefined patterns, as follows.

- The simplest pattern is a query, or 'Get' pattern in which an initiator sends a query packet to the device which then returns with the data.

- This pattern can be extended by permitting the device to delay its response to the query until data is ready. This is referred to as a 'Delayed-Response Get'. In this pattern, the device response not only conveys data, but the timing of the response also signals when the data became available. This pattern can be further extended by permitting multiple queries to be queued at the device.

- An alternative to the Delayed-Response Get pattern is available if interrupt signalling is supported. In this case the device may signal the availability of data via an interrupt. Another node may then query the data using a straightforward Get.

- The final pattern is a straightforward 'Issue' where the device is (pre-)configured to send data packets to a suitable node whenever data becomes available. This may, of course, be periodic. In this pattern, the device itself is the initiator and the issued packets may be directed at the network node which was responsible for configuring the device, or another node, such as a mass memory.

To permit these patterns to be used in a standard manner a standard protocol must be used. The only contender for SpaceWire is the RMAP protocol, as no standard packet protocol exists. The CCSDS Packet Transfer Protocol (CPTP) could also be used, especially for the Issue pattern, and may be appropriate for devices which already utilise CCSDS packets. CCSDS packet-based protocols such as the European Packet Utilisation Standard (PUS) are application layer protocols and it may not be appropriate to configure their use with a SpaceWire plug-and-play protocol.

4.1.5 Commanding Actuators

The actuator, or data sink, case mirrors the patterns presented for the data source case in Section . Four patterns exist for transferring data from some node to the data sink device, each corresponds to a data source case.

- The simplest pattern is a command, or 'Set' pattern in which an initiator sends a command packet to the device. The device may acknowledge the command, but this is optional.

- This pattern can be extended by permitting the device to delay its response to the command until the device is available for commanding. This is referred to as a 'Delayed-Response Set'. In this pattern the device may have reasons for delaying its use of the command, for example it may be synchronised to an external source such as a PPS signal. The device may provide an acknowledgement, in which case the timing of the response signals when the command has been acted on. This pattern can be further extended by permitting multiple commands to be queued at the device.

- An alternative to the Delayed-Response Set pattern is available if interrupt signalling is supported. In this case the device may signal its availability for commanding via an interrupt. Another node may then command the device using a straightforward Set.

- The final pattern is a 'Query' where the device is (pre-)configured to send data request packets to a suitable node whenever the device is available for commanding. This may, of course, be periodic. This pattern is the counterpart of the 'Issue' pattern in that the device itself is the initiator. Query packets may be directed at the network node which was responsible for configuring the device, or another node, such as a mass memory or even a sensor.

Again, the use of standard patterns requires the use of a standard protocol. In this case the only appropriate protocol is the RMAP protocol.

4.1.6 Network Change Detection and FDIR

Changes in the availability of devices give rise to changes in network topology. These changes may have been commanded, or they may be unexpected, for example due to the failure of a device. In either case, the detection of a change is essential to help determine if the network has entered an anomalous state. If a change has been commanded, then the change is expected and any different change is anomalous. An unexpected change is, by definition, anomalous. Network change detection therefore forms an important part of FDIR.

Two main mechanisms exist for detecting network topology changes:

- network discovery, in which an interested node polls network devices in order to incrementally discover the topology, usually using a tree algorithm;

- notification, in which a router which is already part of the network detects a change to activity on its ports and reports this asynchronously to an interested node.

The latter case is more complicated but results in significantly less network traffic. In either case it may be necessary to discover only part of the network and/or to seed the discovery process with current knowledge of the network. It is then possible to detect deviations from a 'last-known' or 'known-good' state.

A typical FDIR period, particularly for payload applications of SpaceWire, is in the order of 1Hz. In extreme cases, for example on the platform side of the spacecraft, the FDIR period may approach the control period of the spacecraft, typically in the order of 10Hz.

Fault mitigation onboard a spacecraft typically involves the use of one or more redundant units which may exist in hot, cold or warm redundancy. The discovery and management of devices must permit the potential for duplicate, redundant, devices active on the network including the potential for duplicate nodes carrying out discovery and management activities.

4.2 Alignment with CCSDS SOIS

The CCSDS SOIS standards have been designed to specify a standard communication architecture and a standard set of services to support onboard communication. The subnetwork services are the lowest level services and provide access to the services of an onboard bus or network such as SpaceWire. There are five subnetwork services: Packet, Memory Access, Device Discovery, Test and Synchronisation. The first four are relevant to a SpaceWire plug-and-play protocol with Device Discovery and Test being directly in scope. As the acceptance of SOIS grows in the industry so it becomes increasingly important that a SpaceWire plug-and-play protocol provides the facilities necessary to satisfy the Device Discovery and Test services. Additionally, it would be helpful if a plug-and-play protocol provided the functions necessary to permit the detection and management of network features which support Packet, Memory Access and Synchronisation services.

4.3 Requirements

This section elicits a set of requirements from the use cases and concerns in the previous sections. Requirements are presented on the two interfaces to the protocol:

- the service interface presented to software;
- the SpaceWire protocol encompassing packet encoding and exchange semantics. This is effectively the interface between devices.

For each interface, requirements are related to the use cases presented above.

4.3.1 Functional Requirements

4.3.1.1 Service Interface

Req PnP-SW-FN-001

The PnP protocol service interface shall permit the presence and location of a known device to be determined.

#

Req PnP-SW-FN-002

The PnP protocol service interface shall permit each located device to be uniquely identified.

#

Req PnP-SW-FN-003

The PnP protocol service interface shall permit the operational status of each located device to be determined.

#

Req PnP-SW-FN-011

The PnP protocol service interface shall permit all aspects of a SpaceWire link to be configured, providing such features are supported by the device, including:

- link state;
- transmit speed;
- time-code forwarding.

#

Req PnP-SW-FN-012

The PnP protocol service interface shall permit link distributed interrupt forwarding to be configured, if supported by the device.

#

Req PnP-SW-FN-013

The PnP protocol service interface shall permit all aspects of a SpaceWire router to be configured, providing such features are supported by the device, including:

- routing table configuration;
- routing arbitration;
- routing time out watchdog;
- time-code counter reset.

#

Req PnP-SW-FN-014

The PnP protocol service interface shall permit interrupt forwarding time outs to be configured, if supported by the routing device.

#

Req PnP-SW-FN-015

The PnP protocol service interface shall permit the status of a SpaceWire link to be determined, providing status features are supported by the device, including:

- link state (running/not running);
- whether a disconnect error has occurred;
- whether a parity error has occurred;
- whether an escape error has occurred;
- whether a credit error has occurred;
- whether a character sequence error has occurred.

#

Req PnP-SW-FN-016

The PnP protocol service interface shall permit the status of a SpaceWire router to be determined, providing status features are supported by the device, including:

- whether a watchdog time out error has occurred;
- whether a packet address error has occurred.

#

Req PnP-SW-FN-021

The PnP protocol service interface shall permit the link and router configuration policy to be applied during the discovery process, where appropriate.

#

Req PnP-SW-FN-031

The PnP protocol service interface shall permit the acquisition of data from a device according to one of the following patterns:

- Get;
- Delayed-Response Get;
- Issue.

#

Req PnP-SW-FN-032

The PnP protocol service interface shall permit the interrupt-triggered acquisition of data from a device if interrupts are supported by the device and the necessary portion of the network.

#

Req PnP-SW-FN-041

The PnP protocol service interface shall permit the commanding of a device according to one of the following patterns:

- Acknowledged Set;
- Unacknowledged Set;
- Acknowledged Delayed-Response Set;
- Unacknowledged Delayed-Response Set
- Query.

#

Req PnP-SW-FN-042

The PnP protocol service interface shall permit the interrupt-triggered commanding of a device if interrupts are supported by the device and the necessary portion of the network.

#

Req PnP-SW-FN-051

The PnP protocol service interface shall permit changes to network topology to be detected via explicit invocation.

#

Req PnP-SW-FN-052

The PnP protocol service interface shall permit changes to network topology to be detected automatically.

Note: This may be carried out using polled discovery or notification.

#

Req PnP-SW-FN-053

The PnP protocol service interface shall permit changes to device operational status to be detected via explicit invocation.

#

Req PnP-SW-FN-054

The PnP protocol service interface shall permit changes to device operational status to be detected automatically.

Note: This may be carried out using polled discovery or notification.

#

Req PnP-SW-FN-061

The PnP protocol service interface shall permit multiple devices of identical type.

#

Req PnP-SW-FN-062

The PnP protocol service interface shall permit multiple devices engaged in network/device discovery simultaneously.

#

4.3.1.2 SpaceWire Protocol

Req PnP-SW-FN-101

The PnP protocol shall permit the presence and location of a known device to be determined.

#

Req PnP-SW-FN-102

The PnP protocol shall permit each located device to be uniquely identified.

#

Req PnP-SW-FN-103

The PnP protocol service interface shall permit the operational status of each located device to be determined.

#

Req PnP-SW-FN-111

The PnP protocol shall permit all aspects of a SpaceWire link to be configured, providing such features are supported by the device, including:

- link state;
- transmit speed;
- time-code forwarding.

#

Req PnP-SW-FN-112

The PnP protocol shall permit link distributed interrupt forwarding to be configured, if supported by the device.

#

Req PnP-SW-FN-113

The PnP protocol shall permit all aspects of a SpaceWire router to be configured, providing such features are supported by the device, including:

- routing table configuration;
- routing arbitration;
- routing time out watchdog;
- time-code counter reset.

#

Req PnP-SW-FN-114

The PnP protocol shall permit interrupt forwarding time outs to be configured, if supported by the routing device.

#

Req PnP-SW-FN-115

The PnP protocol shall permit the status of a SpaceWire link to be determined, providing status features are supported by the device, including:

- link state (running/not running);
- whether a disconnect error has occurred;
- whether a parity error has occurred;
- whether an escape error has occurred;
- whether a credit error has occurred;
- whether a character sequence error has occurred.

#

Req PnP-SW-FN-116

The PnP protocol shall permit the status of a SpaceWire router to be determined, providing status features are supported by the device, including:

- whether a watchdog time out error has occurred;
- whether a packet address error has occurred.

#

Req PnP-SW-FN-121

The PnP protocol shall permit the link and router configuration policy to be applied during the discovery process, where appropriate.

#

Req PnP-SW-FN-131

The PnP protocol shall permit the acquisition of data from a device according to one of the following patterns:

- Get;
- Delayed-Response Get;
- Issue.

#

Req PnP-SW-FN-132

The PnP protocol shall permit the interrupt-triggered acquisition of data from a device if interrupts are supported by the device and the necessary portion of the network.

#

Req PnP-SW-FN-141

The PnP protocol shall permit the commanding of a device according to one of the following patterns:

- Acknowledged Set;
- Unacknowledged Set;
- Acknowledged Delayed-Response Set;
- Unacknowledged Delayed-Response Set
- Query.

#

Req PnP-SW-FN-142

The PnP protocol shall permit the interrupt-triggered commanding of a device if interrupts are supported by the device and the necessary portion of the network.

#

Req PnP-SW-FN-151

The PnP protocol shall permit changes to network topology to be detected.

#

Req PnP-SW-FN-152

The PnP protocol shall permit changes to network topology to be detected automatically via notification if the necessary routers support this feature.

#

Req PnP-SW-FN-153

The PnP protocol service interface shall permit changes to device operational status to be detected.

#

Req PnP-SW-FN-154

The PnP protocol shall permit changes to device operational status to be detected automatically via notification if the necessary routers support this feature.

#

Req PnP-SW-FN-161

The PnP protocol shall permit multiple devices of identical type.

#

Req PnP-SW-FN-162

The PnP protocol shall permit multiple devices engaged in network/device discovery simultaneously.

#

4.3.2 Coverage Requirements

4.3.2.1 Service Interface

Req PnP-SW-CV-001

The PnP protocol service interface shall not place restrictions on network topology.

#

Req PnP-SW-CV-002

The PnP protocol service interface shall not place restrictions on the number of devices in a network.

#

Req PnP-SW-CV-003

The PnP protocol service interface shall not place restrictions on the number of devices simultaneously engaged in network discovery and management.

#

4.3.2.2 SpaceWire Protocol

Req PnP-SW-CV-101

The PnP protocol shall not place restrictions on network topology.

#

Req PnP-SW-CV-102

The PnP protocol shall not place restrictions on the number of devices in a network.

#

Req PnP-SW-CV-103

The PnP protocol shall not place restrictions on the number of devices simultaneously engaged in network discovery and management.

#

4.3.3 Performance Requirements

4.3.3.1 Service Interface

Req PnP-SW-PF-001

The PnP protocol service interface shall permit network/device changes to be detected within a period of at least 1 second per network change.

Note: i.e. a network change involving two devices becoming connected simultaneously must be detectable within 2 seconds.

#

4.3.3.2 SpaceWire Protocol

Req PnP-SW-PF-101

The PnP protocol shall permit network/device changes to be detected within a period of at least 1 second per network change.

Note: i.e. a network change involving two devices becoming connected simultaneously must be detectable within 2 seconds.

#

4.3.4 Design Requirements

4.3.4.1 Service Interface

Req PnP-SW-DS-001

The PnP protocol service interface shall be designed to be as simple as possible.

Note: i.e. the number of primitives should be minimised and redundancy between primitives should be minimised. Additionally, coupling between primitives should be minimised.

#

4.3.4.2 SpaceWire Protocol

Req PnP-SW-DS-101

The PnP protocol shall be designed to be as simple as possible.

Note: i.e. the number of packet format variations should be minimised and redundancy between packet format variations should be minimised. Additionally, the protocol semantics should be kept as simple as possible.

#

Req PnP-SW-DS-102

The PnP protocol shall utilise the RMAP packet format and underlying protocol semantics.

#

4.3.5 Interface Requirements

4.3.5.1 Service Interface

Req PnP-SW-IF-001

The PnP protocol service interface shall be designed to align with the CCSDS SOIS subnetwork services.

#

4.3.6 Environmental Requirements

N/A

4.3.7 Physical and Resource Requirements

Req PnP-SW-PR-001

It shall be feasible to implement the PnP protocol entirely in software for execution on a LEON2 processor running at 50MHz with no more than 4Mbytes of memory.

#

4.3.8 Operational Requirements

N/A

4.3.9 Human Factors Requirements

N/A

4.3.10 Logistics Support Requirements

N/A

4.3.11 Product Assurance Requirements

N/A

4.3.12 Configuration and Implementation Requirements

N/A

4.3.13 Verification and Testing Requirements

N/A

End of Section

5. EQUIPMENT SUPPLIER REQUIREMENTS

This chapter presents requirements on a SpaceWire plug-and-play protocol from the perspective of an equipment supplier. These requirements come from experience developing equipment for two very different purposes: for flight use and for use in test, development and electronic ground support.

1.1 Requirements

This section contains a set of requirements obtained through consideration of the features relating to PnP currently provided in SpaceWire equipment, and features often requested by users. The areas considered include:

- network configuration and status;
- interface and router device configuration and status;
- link configuration and status;
- debug information.

The requirements have been separated in to one of these categories where appropriate.

1.1.1 Functional Requirements

1.1.1.1 Network Configuration and Status

Req PnP-EQ-FN-001

The PnP protocol shall permit the presence and location of a known device to be determined.

#

Req PnP-EQ-FN-002

The PnP protocol shall permit each located device to be uniquely identified.

#

Req PnP-EQ-FN-003

The PnP protocol shall permit changes to network topology to be detected.

#

Req PnP-EQ-FN-004

The PnP protocol shall permit multiple devices of identical type.

#

Req PnP-EQ-FN-005

The PnP protocol shall permit multiple devices engaged in network/device discovery simultaneously.

#

1.1.1.2 Interface and Router Device Configuration and Status

Req PnP-EQ-FN-101

The PnP protocol shall permit the operational status of each located device to be determined.

#

Req PnP-EQ-FN-102

The PnP protocol shall permit changes to device operational status to be detected.

#

Req PnP-EQ-FN-103

The PnP protocol shall permit all aspects of a SpaceWire router to be configured, providing such features are supported by the device, including:

- routing table configuration;
- routing arbitration;
- routing time out watchdog;
- maximum packet size watchdog;
- disable ports on silence;
- start ports on request;
- time-code counter reset.

Note: disabling ports on silence involves disabling a port when it is not in use for a period of time, in order to save power. Starting ports on request involves the router starting the link when a packet is to be sent on that link.

#

Req PnP-EQ-FN-104

The PnP protocol shall permit the status of a SpaceWire router to be determined, providing status features are supported by the device, including:

- whether a watchdog time out error has occurred;
- whether a maximum packet size watchdog error has occurred;

- whether a packet address error has occurred.

#

Req PnP-EQ-FN-105

The PnP protocol shall permit a device to be enabled as a time-code master, providing this feature is supported by the device.

#

Req PnP-EQ-FN-106

The PnP protocol shall permit the time-code generation frequency of a device to be configured, providing this feature is supported by the device.

#

1.1.1.3 Link Configuration and Status

Req PnP-EQ-FN-201

The PnP protocol shall permit the type of a SpaceWire port to be determined, from one of the following:

- configuration;
- SpaceWire;
- external.

#

Req PnP-EQ-FN-202

The PnP protocol shall permit all aspects of a SpaceWire link to be configured, providing such features are supported by the device, including:

- link state;
- transmit speed;
- time-code forwarding.

#

Req PnP-EQ-FN-203

The PnP protocol shall permit the status of a configuration port to be determined, providing status features are supported by the device, including whether each possible error specific to the PnP protocol has occurred.

#

Req PnP-EQ-FN-204

The PnP protocol shall permit the status of a SpaceWire link to be determined, providing status features are supported by the device, including:

- link state (running/not running);
- whether a disconnect error has occurred;
- whether a parity error has occurred;
- whether an escape error has occurred;
- whether a credit error has occurred;
- whether a character sequence error has occurred.

#

Req PnP-EQ-FN-205

The PnP protocol shall permit the status of an external port to be determined, providing status features are supported by the device.

#

1.1.1.4 Debug Information

Req PnP-EQ-FN-301

The PnP protocol may permit debug aspects of a SpaceWire router to be configured, providing such features are supported by the device, including:

- enable self-addressing.

Note: self-addressing refers to the ability of a router to route a packet out of the same port on which it is received. This is not a feature that it likely to be required onboard a spacecraft, but can be very useful during development and testing.

#

Req PnP-EQ-FN-302

The PnP protocol may permit the debug status of a SpaceWire link to be determined, providing debug status features are supported by the device, including:

- current transmit flow control credit;
- current receive flow control credit.

#

Req PnP-EQ-FN-303

The PnP protocol may permit the debug status of a SpaceWire device to be determined, providing debug status features are supported by the device, including:

- device identity;

- manufacturer identity;
- device version;
- device version build date.

Note: the device and manufacturer identity may be represented numerically.

#

Req PnP-EQ-FN-304

The PnP protocol may permit a device to visually identify itself when requested for debugging purposes.

Note: this is a very useful feature when test and development involves a large rack of equipment where each device may look identical. The visual identification can be achieved by flashing the device's LEDs, for example. This can also be very useful in a demonstration system, for example to indicate the device currently being configured.

#

1.1.2 Coverage Requirements

1.1.2.1 Network Configuration and Status

Req PnP-EQ-CV-001

The PnP protocol may restrict the number of devices in a network to a maximum number of devices.

Note: unique identification of a device is likely to involve the use of a 32-bit register, so this may limit the maximum number of devices to e.g. $2^{32} – 1$).

#

Req PnP-EQ-CV-002

The PnP protocol shall not place restrictions on network topology, other than the maximum number of devices permitted in the network.

#

Req PnP-EQ-CV-003

The PnP protocol shall not place restrictions on the number of devices simultaneously engaged in network discovery and management, other than the maximum number of devices permitted in the network.

#

1.1.3 Performance Requirements

1.1.3.1 Network Configuration and Status

Req PnP-EQ-PF-001

The PnP protocol shall permit network/device changes to be detected within a period of 1 second per network change.

Note: i.e. a network change involving two devices becoming connected simultaneously must be detectable within 2 seconds.

#

Req PnP-EQ-PF-002

The PnP protocol may permit the time period within which a network change should be detected to be increased beyond the limit of 1 second when used in a test environment.

Note: increasing this time period may be necessary when virtual spacecraft integration is used to integrate components virtually using the Internet, where latency and bandwidth is greatly reduced.

#

1.1.4 Design Requirements

Req PnP-EQ-DS-001

The PnP protocol shall be designed to be as simple as possible.

#

Req PnP-EQ-DS-002

The PnP protocol shall utilise the RMAP packet format and underlying protocol semantics.

#

1.1.4.1 Debug Information

Req PnP-EQ-DS-101

PnP protocol packets shall be easily identifiable from other SpaceWire packets when analysing the packets crossing a SpaceWire link.

Note: the PnP Protocol must use its own Protocol ID to distinguish it from other RMAP packets.

#

Req PnP-EQ-DS-102

The source of PnP protocol packets shall be easily identifiable when analysing the packets crossing a SpaceWire link.

Note: if using the RMAP packet format, PnP packets will include a source address.

#

1.1.5 Interface Requirements

1.1.5.1 Interface and Router Device Configuration and Status

Req PnP-EQ-IF-001

The PnP protocol shall operate with existing SpaceWire routers.

#

1.1.6 Environmental Requirements

N/A

1.1.7 Physical and Resource Requirements

1.1.7.1 Interface and Router Device Configuration and Status

Req PnP-EQ-PR-001

It shall be possible to implement the PnP protocol on a device which is only capable of responding to PnP commands, not initiating them.

> Note: a device which does not operate as the initiator of commands should only require RMAP target-like functionality.

#

Req PnP-EQ-PR-002

It shall be possible to implement the PnP protocol in an ASIC without the need for an accompanying memory device.

> Note: these devices will not be capable of implementing a unique MAC address.

#

1.1.8 Operational Requirements

N/A

1.1.9 Human Factors Requirements

N/A

1.1.10 Logistics Support Requirements

N/A

1.1.11 Product Assurance Requirements

N/A

1.1.12 Configuration and Implementation Requirements

N/A

1.1.13 Verification and Testing Requirements

N/A

End of Section

2. CUSTOMER REQUIREMENTS

The requirements in this chapter have been compiled from the Statement of Work for this activity [AD7] plus input from the SpaceWire Working Group. All Working Group input is taken from archives of Working Group meeting presentations which are generally available via the working group website. Each requirement is labelled with its original source for traceability.

2.1 Definitions

These definitions are taken from the Statement of Work [AD7] and are relevant to the customer requirements only.

Connection of a SpaceWire device to a SpaceWire network is defined by the activation of the SpaceWire link to which the device is connected. This might result from physical connection of an already active device (SpaceWire link interface in Started state) or from activation of an already physically connected passive device (switching the SpaceWire link interface to Enabled).

Disconnection of a SpaceWire device from a SpaceWire network is defined by the de-activation of the SpaceWire link from which the device is disconnected. This might result from physical disconnection of a connected device (SpaceWire link interface in Run state) or from de-activation of a connected device without physical disconnection (switching the SpaceWire link interface to Disabled). The case where a transient error on the link leads to Link Disconnect and immediate link re-establishment is not considered as disconnection in the sense of SpW-PnP.

A **SpaceWire-based Protocol (SbP)** is a protocol that is expected to be carried on a SpaceWire cargo because its semantics relates to SpaceWire's semantics.

The **SbP capabilities** (of a SpaceWire device) are the SpaceWire-based protocols (SbPs) that this device supports, and the ways in which they can be transported (e.g. using raw SpaceWire or SpW-D).

The **SpW-PnP** SbP is a SpaceWire-based protocol that is described in section A.1.1.2 and fulfils the requirements expressed in section A.1.2.

A **SpW-PnP enabled device** is a SpaceWire device that can execute and respond to SpW-PnP requests and messages.

A **SpW-PnP master** is a device that can

1. Automatically discover and configure hardware and software systems in response to changes in hardware interfacing or availability, including whilst the system is running,

2. Detect and configure the services that connected SpW-PnP enabled devices provide, by establishing and controlling some SpW-PnP based communication with all other devices of a SpaceWire network.

2.2 Requirements

2.2.1 Functional Requirements

2.2.1.1 Device Detection and Identification

Req PnP-EC-FN-001

Upon connection of a Plug-and-Play enabled SpaceWire device to a SpaceWire network, the SpaceWire Plug-and-Play protocol shall provide a mechanism for any SpW-PnP master to detect and locate this connection.

> Note: The connection may be physical or SpaceWire protocol-wise (see definitions in section). The Plug-and-Play enabled SpaceWire device being connected to the SpaceWire network may be itself a SpW-PnP master or not. It might even be the only SpW-PnP master of the network. The effect of connecting of some non Plug-and-Play enabled SpaceWire device to a Plug-and-Play enabled SpaceWire network is not specified and will depend on the behaviour of the non Plug-and-Play enabled SpaceWire device.

Source: [AD7] [SpWPnP-FR-DDI-REQ-10]

#

Req PnP-EC-FN-002

Upon disconnection of a Plug-and-Play enabled SpaceWire device from a SpaceWire network, the SpaceWire Plug-and-Play protocol shall provide a mechanism for any SpW-PnP master still connected to the network to detect and locate this disconnection.

> Note: The disconnection may be physical or SpaceWire protocol-wise (see definitions in section). The effect of disconnecting of some non Plug-and-Play enabled SpaceWire device to a Plug-and-Play enabled SpaceWire network is not specified and will depend on the behaviour of the non Plug-and-Play enabled SpaceWire device.

Source: [AD7] [SpWPnP-FR-DDI-REQ-20]

#

Req PnP-EC-FN-003

The SpaceWire Plug-and-Play protocol shall allow uniquely identifying all Plug-and-Play enabled devices connected on the network.

Source: [AD7] [SpWPnP-FR-DDI-REQ-30]

#

2.2.1.2 Device Ownership

Req PnP-EC-FN-101

Upon detection and localisation of connection of a Plug-and-Play enabled SpaceWire device to a SpaceWire network, the SpaceWire Plug-and-Play protocol shall provide an atomic mechanism for a SpW-PnP master connected to the network to claim ownership of this device, identifying and contacting other ownership claimers and resolving conflicts.

> Note: The Plug and Play enabled SpaceWire device being connected to the SpaceWire network may be itself a SpW-PnP master or not.

Source: [AD7] [SpWPnP-FR-DO-REQ-40]

#

Req PnP-EC-FN-102

Upon detection and localisation of disconnection of a Plug-and-Play enabled SpaceWire device from a SpaceWire network, the SpaceWire Plug-and-Play protocol shall provide a mechanism for a SpW-PnP master still connected to the network to renounce ownership of this device, informing all knowledgeable devices and resolving conflicts.

Source: [AD7] [SpWPnP-FR-DO-REQ-50]

#

2.2.1.3 Device Identification and Status

Req PnP-EC-FN-201

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide core information to allow the device owner identifying the device, its type (node or router, and device type field), and vendor information.

Source: [AD7] [SpWPnP-FR-DIS-REQ-60]

#

Req PnP-EC-FN-202

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the device status, ownership details and port activity parameters.

Source: [AD7] [SpWPnP-FR-DIS-REQ-70]

#

Req PnP-EC-FN-203

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide error reporting: SpW-PnP, Protocol ID and other SbP (e.g. SpaceWire-D) error recording shall be accessible, if implemented.

Source: [AD7] [SpWPnP-FR-DIS-REQ-80]

#

2.2.1.4 Capability Discovery

Req PnP-EC-FN-301

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide a summary of the capabilities of this device.

> Note: As SpW-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 SpW-D). 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 SpW-PnP.

Source: [AD7] [SpWPnP-FR-CD-REQ-90]

#

2.2.1.5 Owner-Proxy

Req PnP-EC-FN-401

Device owners shall use a proxy service to provide access to the devices they own.

Source: [AD7] [SpWPnP-FR-OP-REQ-100]

#

2.2.1.6 Link Status and Configuration

Req PnP-EC-FN-501

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability to query the status of any device link and configure its state and speed.

Source: [AD7] [SpWPnP-FR-LSC-REQ-110]

#

2.2.1.7 Router Status and Configuration

Req PnP-EC-FN-601

Upon ownership acquisition over a SpaceWire router, or at any time that this router is connected, the SpaceWire Plug-and-Play protocol shall provide support to configure the routing tables and routing mechanisms of this router, as well as Time-Code propagation settings.

Source: [AD7] [SpWPnP-FR-RSC-REQ-120]

#

2.2.1.8 Time-Code Source

Req PnP-EC-FN-701

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, the SpaceWire Plug-and-Play protocol shall provide the ability to query if this device is Time-Code enabled, i.e. if it has the capability to introduce Time-Codes into the SpaceWire network.

Source: [AD7] [SpWPnP-FR-TCS-REQ-130]

#

Req PnP-EC-FN-702

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that the device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability to allow Time-Code generation to be enabled and disabled.

Source: [AD7] [SpWPnP-FR-TCS-REQ-140]

#

Req PnP-EC-FN-703

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire Time-Code enabled device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability to configure the frequency of Time-Codes.

Source: [AD7] [SpWPnP-FR-TCS-REQ-150]

#

Req PnP-EC-FN-704

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire Time-Code enabled device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability to configure the device for on-demand tick generation.

Source: [AD7] [SpWPnP-FR-TCS-REQ-160]

#

2.2.1.9 Generic Data Source

Req PnP-EC-FN-801

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability for this device to register to its owner for RMAP Instant Reads.

Note: A SpaceWire device A registers to its owner for RMAP Instant Reads when it provides to its owner a data-ready status indicator in addition to the necessary information (e.g. Target SpaceWire

Address, Target Logical Address, Key, Extended Address, Address, and Data Length) for some other device B to later issue regular RMAP Read commands to device A for a fixed maximum amount of data sitting at the same location within device A. To be able to issue such RMAP Read commands, device B needs to get the necessary information from device A's owner, possibly via a proxy. Device B may be granted exclusive access to the specified data within device A. Reads when the source is not ready will return an error.

Source: [AD7] [SpWPnP-FR-GDS-REQ-170]

#

Req PnP-EC-FN-802

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability for this device to register to another one for RMAP Delayed Reads.

Note: A SpaceWire device A registers to its owner for RMAP Delayed Reads when it provides to its owner a data-ready status indicator in addition to the necessary information (e.g. Target SpaceWire Address, Target Logical Address, Key, Extended Address, Address, and Data Length) for some other device B to later issue regular RMAP Read commands to device A for a fixed maximum amount of data sitting at the same location within device A. To be able to issue such RMAP Read commands, device B needs to get the necessary information from device A's owner, possibly via a proxy. Device B may be granted exclusive access to the specified data within device A. Reads when the source is not ready will result in delaying the response until data is ready. A timeout must be specified to ensure that the source responds eventually, even if data is never ready.

Source: [AD7] [SpWPnP-FR-GDS-REQ-180]

#

Req PnP-EC-FN-803

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability for this device to register to another one for Initiated RMAP Writes.

Note: A SpaceWire device A registers to its owner for Initiated RMAP Writes when it provides to its owner a data-ready status indicator in addition to the necessary information (e.g. Data Length) for some other device B to accept that device A later issues regular RMAP Write commands to device B for a fixed maximum amount of data to be written at the same location within device B. To enable device A to issue such RMAP Writes commands, device B needs to provide the necessary information (e.g. Target SpaceWire Address, Target Logical Address, Key, Extended Address, and Address) to device A via device A's owner or a proxy. Device B may be granted exclusive access to the specified data within device A. Writes when the destination is not ready will be discarded.

Source: [AD7] [SpWPnP-FR-GDS-REQ-190]

#

2.2.1.10 Generic Data Sinks

Req PnP-EC-FN-901

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability for this device to register to its owner for Non-Queued Unacknowledged RMAP Writes.

Note: A SpaceWire device A registers to its owner for Non-Queued Unacknowledged RMAP Writes when it provides to its owner a sink-ready status indicator in addition to the necessary information (e.g. Target SpaceWire Address, Target Logical Address, Key, Extended Address, Address, and Data Length) for some other device B to later issue regular RMAP non-acknowledged, verified or non-verified Write commands to device A for a fixed maximum amount of data to be written at the same location within device A. To be able to issue such RMAP Write commands, device B needs to get the necessary information from device A's owner, possibly via a proxy. Device B may be granted exclusive access to the specified sink area within device A. Writes when the sink is not ready will be discarded.

Source: [AD7] [SpWPnP-FR-GDS-REQ-200]

#

Req PnP-EC-FN-902

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability for this device to register to its owner for Non-Queued Acknowledged RMAP Writes.

Note: A SpaceWire device A registers to its owner for Non-Queued Acknowledged RMAP Writes when it provides to its owner a sink-ready status indicator in addition to the necessary information (e.g. Target SpaceWire Address, Target Logical Address, Key, Extended Address, Address, and Data Length) for some other device B to later issue regular RMAP non-acknowledged, verified or non-verified Write commands to device A for a fixed maximum amount of data to be written at the same location within device A. To be able to issue such RMAP Write commands, device B needs to get the necessary information from device A's owner, possibly via a proxy. Device B may be granted exclusive access to the specified sink area within device A. Successful Writes will return positive RMAP acknowledgment. Writes when the sink is not ready will be discarded and return a negative RMAP acknowledgment.

Source: [AD7] [SpWPnP-FR-GDS-REQ-210]

#

Req PnP-EC-FN-903

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability for this device to register to its owner for Queued Unacknowledged RMAP Writes.

Note: A SpaceWire device A registers to its owner for Queued Unacknowledged RMAP Writes when it provides to its owner a sink-ready status indicator in addition to the necessary information (e.g. Target SpaceWire Address, Target Logical Address, Key, Extended Address, Address, and Data Length) for some other device B to later issue regular RMAP non-acknowledged, verified or non-verified Write commands to device A for a fixed maximum amount of data to be written at the same location within device A. To be able to issue such RMAP Write commands, device B needs to get the necessary information from device A's owner, possibly via a proxy. Device B may be granted exclusive access to the specified sink area within device A. Device A will accept one or more unacknowledged writes. If more than one Write is accepted subsequently, the Writes are placed in a FIFO queue. If the queue is full, the incoming Writes will be discarded.

Source: [AD7] [SpWPnP-FR-GDS-REQ-220]

#

Req PnP-EC-FN-904

Upon ownership acquisition over a Plug-and-Play enabled SpaceWire device, or at any time that this device is connected, the SpaceWire Plug-and-Play protocol shall provide the ability for this device to register to its owner for Queued Acknowledged RMAP Writes.

> Note: A SpaceWire device A registers to its owner for Queued Acknowledged RMAP Writes when it provides to its owner a sink-ready status indicator in addition to the necessary information (e.g. Target SpaceWire Address, Target Logical Address, Key, Extended Address, Address, and Data Length) for some other device B to later issue regular RMAP non-acknowledged, verified or non-verified Write commands to device A for a fixed maximum amount of data to be written at the same location within device A. To be able to issue such RMAP Write commands, device B needs to get the necessary information from device A's owner, possibly via a proxy. Device B may be granted exclusive access to the specified sink area within device A. Device A will accept one or more acknowledged writes. If more than one Write is accepted subsequently, the Writes are placed in a FIFO queue. If the queue is full, the sink returns an error.

Source: [AD7] [SpWPnP-FR-GDS-REQ-230]

#

2.2.2 Coverage Requirements

2.2.2.1 Network Topology

Req PnP-EC-CV-001

The SpaceWire Plug-and-Play protocol shall allow discovering and configuring any Plug-and-Play enabled SpaceWire network with unknown and arbitrary topology.

> Note: There might be some limitation on the number of SpaceWire devices in the network.

Source: [AD7] [SpWPnP-CR-TOP-REQ-240]

#

Req PnP-EC-CV-002

The SpaceWire Plug-and-Play protocol shall allow (re)discovering and (re)configuring any Plug-and-Play enabled SpaceWire network whilst arbitrary network topology changes can occur at any time due to failures or user intervention.

> Note: There might be some limitation on the number of SpaceWire devices in the network.

Source: [AD7] [SpWPnP-CR-TOP-REQ-250]

#

Req PnP-EC-CV-003

The SpaceWire Plug-and-Play protocol shall allow (re)discovering and (re)configuring any Plug-and-Play enabled SpaceWire network whilst devices or subnets can be connected and disconnected to/from any element of an existing network at any time and devices newly connected may not be in reset status.

Note: There might be some limitation on the number of SpaceWire devices in the network.

Source: [AD7] [SpWPnP-CR-TOP-REQ-260]

#

Req PnP-EC-CV-004

The SpaceWire Plug-and-Play protocol shall allow (re)discovering and (re)configuring any Plug-and-Play enabled SpaceWire network whilst multiple devices with the same hardware configuration may be present in the same network.

Note: There might be some limitation on the number of SpaceWire devices in the network.

Source: [AD7] [SpWPnP-CR-TOP-REQ-270]

#

2.2.2.2 Redundancy

Req PnP-EC-CV-101

The SpaceWire Plug-and-Play protocol shall allow discovering and configuring any Plug-and-Play enabled SpaceWire network with unknown and arbitrary topology.

Note: In particular, SpW-PnP shall support the handling of redundant units.

Source: [AD7] [SpWPnP-CR-RED-REQ-280]

#

Req PnP-EC-CV-102

The SpaceWire Plug-and-Play protocol shall allow discovering and configuring any Plug-and-Play enabled SpaceWire network by using multiple SpW PnP masters. Design shall avoid race conditions.

Note: In particular, SpW-PnP shall support the handling of redundant SpW PnP masters.

Source: [AD7] [SpWPnP-CR-RED-REQ-290]

#

Req PnP-EC-CV-103

In case of redundant SpW PnP masters, the SpaceWire Plug-and-Play protocol shall avoid race conditions.

Note: This is part of the support to the handling of redundant SpW PnP masters.

Source: [AD7] [SpWPnP-CR-RED-REQ-300]

#

2.2.3 Performance Requirements

2.2.3.1 Network Discovery and Configuration

Req PnP-EC-PF-001

Based on the computational performance of devices and units currently used on ESA spacecraft, the SpaceWire Plug-and-Play protocol shall allow first discovery and configuration of any Plug-and-Play enabled SpaceWire network within as many seconds as the network contains devices.

> Note: SpW-PnP is expected to allow discovering a ten-device SpaceWire network from scratch in 10 seconds, a 15-device network in 15 seconds, etc.

Source: [AD7] [SpWPnP-PR-NDC-REQ-310]

#

2.2.3.2 Network Change Notification

Req PnP-EC-PF-101

Once the SpaceWire Plug-and-Play protocol has discovered and configured a given Plug-and-Play enabled SpaceWire network, it shall allow detecting one change in the network topology and perform the related reconfiguration of the Plug-and-Play enabled SpaceWire devices within one second, based on the computational performance of devices and units currently used on ESA spacecraft.

> Note: There might be exceptions to this requirement when a single change in the network leads to a very complex reconfiguration of all devices of a large network.

Source: [AD7] [SpWPnP-PR-NCN-REQ-320]

#

2.2.4 Design Requirements

2.2.4.1 ECSS writing rules

Req PnP-EC-DS-001

The specification of the SpaceWire Plug-and-Play protocol shall follow the rules as described in [AD5] and [AD6].

> Note: This will allow quicker path towards ECSS standardisation of SpW-PnP once the SpaceWire Working Group has approved it.

Source: [AD7] [SpWPnP-DR-EWR-REQ-330]

#

2.2.4.2 RMAP Heritage

Req PnP-EC-DS-101

To achieve Plug-and-Play capabilities, SpW-PnP shall implement messages ("requests" or "commands") that allow reading and setting parameters in the SpaceWire devices connected on the network.

> Note: It is highly recommended that the syntax and synchronisation of RMAP are fully reused and possibly tailored. Of course, the semantics will differ and will have to be described.

Source: [AD7] [SpWPnP-DR-RH-REQ-340]

#

Req PnP-EC-DS-102

The SpaceWire Plug-and-Play protocol shall be designed so as to allow or not verification of the integrity of the messages.

> Note: It is highly recommended that the syntax and synchronisation of RMAP are fully reused and possibly tailored. Of course, the semantics will differ and will have to be described.

Source: [AD7] [SpWPnP-DR-RH-REQ-350]

#

Req PnP-EC-DS-103

The SpaceWire Plug-and-Play protocol shall be designed so as to allow or not acknowledgment of messages.

> Note: It is highly recommended that the syntax and synchronisation of RMAP are fully reused and possibly tailored. Of course, the semantics will differ and will have to be described.

Source: [AD7] [SpWPnP-DR-RH-REQ-360]

#

2.2.5 Interface Requirements

2.2.5.1 Protocol Stack

Req PnP-EC-IF-001

The SpaceWire Plug-and-Play protocol shall be designed so as to operate "on top of" SpaceWire. i.e. SpW-PnP messages shall fit in the cargo of SpaceWire packets as defined in [AD1] and [AD2].

> Note: In other word, SpW-PnP shall be a SbP. As much as possible, provision should also be made so that SpW-PnP can operate alongside other SbPs such as SpW-D as defined in [RD8]. Indeed, the initial network discovery and configuration can be assumed to operate on raw SpaceWire but network change notification and reconfiguration must be allowed when another SbP such as SpW-D is operating.

Source: [AD7] [SpWPnP-IR-PS-REQ-370]

#

2.2.5.2 SpaceWire routers

Req PnP-EC-IF-101

The SpaceWire Plug-and-Play protocol shall operate with SpaceWire routers as defined in [AD1].

Note: In other word, SpW-PnP shall not require any modification of the existing SpaceWire routers.

Source: [AD7] [SpWPnP-IR-ROU-REQ-380]

#

2.2.6 Environmental Requirements

N/A

2.2.7 Physical and Resource Requirements

Req PnP-EC-PR-001

The SpaceWire Plug-and-Play protocol shall be designed so that a reasonably optimised implementation (in terms of hardware and/or VHDL and/or software) fits into devices and units currently used on ESA spacecraft units (including “small” units such as minor payload instruments).

Source: [AD7] [SpWPnP-PRR-SWN-REQ-390]

#

2.2.8 Operational Requirements

N/A

2.2.9 Human Factors Requirements

N/A

2.2.10 Logistics Support Requirements

N/A

2.2.11 Product Assurance Requirements

N/A

2.2.12 Configuration and Implementation Requirements

N/A

2.2.13 Verification and Testing Requirements

N/A

End of Section

3. COLLATED REQUIREMENTS

This chapter collates requirements from the four stakeholders represented by the chapters above. Full traceability from each of the collated requirements to the source requirements is listed.

Furthermore, each requirement is assigned a *mandatory* or *extended* rating depending on how it will be approached in this activity. A rating of *mandatory* indicates that the requirement must be addressed; a rating of *extended* indicates that the requirement may be addressed if time/resources permit, but that it should be considered during design of the protocol.

3.1 Requirements

These collated requirements attempt to capture the meaning of all requirements listed so far, combining requirements where possible.

3.1.1 Functional Requirements

3.1.1.1 Network Configuration and Status

Req PnP-REQ-FN-001 *[Mandatory]*

The PnP protocol and associated service interface shall permit the presence and location of a known device to be determined.

#

Req PnP-REQ-FN-002 *[Mandatory]*

The PnP protocol and associated service interface shall permit an unknown network topology to be determined.

#

Req PnP-REQ-FN-003 *[Mandatory]*

The PnP protocol and associated service interface shall permit an known network topology to be confirmed.

#

Req PnP-REQ-FN-004 *[Mandatory]*

The PnP protocol and associated service interface shall permit each located device to be uniquely identified.

#

Req PnP-REQ-FN-005 *[Mandatory]*

The PnP protocol and associated service interface shall permit the type of each located device to be identified.

Note: this includes whether the device is a node or router.

#

Req PnP-REQ-FN-006 *[Mandatory]*

The PnP protocol and associated service interface shall permit the vendor of each located device to be identified.

#

Req PnP-REQ-FN-007 *[Mandatory]*

The PnP protocol and associated service interface shall permit the capabilities of each located device to be identified.

Note: as SpW-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 SpW-D). 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 SpW-PnP.

#

Req PnP-REQ-FN-008 *[Mandatory]*

The PnP protocol and associated service interface shall permit changes to network topology to be detected.

#

Req PnP-REQ-FN-009 *[Extended]*

The PnP protocol and associated service interface shall permit changes to network topology to be detected via notification, if this is supported by the necessary device(s).

#

Req PnP-REQ-FN-010 *[Mandatory]*

The PnP protocol shall permit the link and router configuration policy to be applied during the discovery process, where appropriate.

#

3.1.1.2 Device Configuration and Status

Req PnP-REQ-FN-101 *[Mandatory]*

The PnP protocol and associated service interface shall permit all aspects of a SpaceWire device to be configured, providing such features are supported by the device, including:

- time-code counter reset.

#

Req PnP-REQ-FN-102 *[Mandatory]*

The PnP protocol and associated service interface shall permit all aspects of a SpaceWire router to be configured, providing such features are supported by the device, including:

- routing table configuration;
- routing arbitration;
- routing time out watchdog;
- maximum packet size watchdog;
- disable ports on silence;
- start ports on request.

Note: disabling ports on silence involves disabling a port when it is not in use for a period of time, in order to save power. Starting ports on request involves the router starting the link when a packet is to be sent on that link.

#

Req PnP-REQ-FN-103 *[Extended]*

The PnP protocol and associated service interface shall permit interrupt forwarding time outs to be configured, if supported by the device.

#

Req PnP-REQ-FN-104 *[Mandatory]*

The PnP protocol and associated service interface shall permit the a device to be queried to determine if the device has the ability to act as a time-code master.

#

Req PnP-REQ-FN-105 *[Mandatory]*

The PnP protocol and associated service interface shall permit a device to be enabled as a time-code master, providing this feature is supported by the device.

#

Req PnP-REQ-FN-106 *[Mandatory]*

The PnP protocol and associated service interface shall permit the time-code generation frequency of a time-code master device to be configured, providing this feature is supported by the device.

#

Req PnP-REQ-FN-107 *[Mandatory]*

The PnP protocol and associated service interface shall permit a time-code master device to be commanded to generate a specified time-code tick on demand, providing this feature is supported by the device.

#

Req PnP-REQ-FN-108 *[Mandatory]*

The PnP protocol and associated service interface shall permit the operational status of each located device to be determined.

#

Req PnP-REQ-FN-109 *[Mandatory]*

The PnP protocol and associated service interface shall permit changes to device operational status to be detected.

#

Req PnP-REQ-FN-110 *[Extended]*

The PnP protocol and associated service interface shall permit changes to device operational status to be detected via notification, if this is supported by the device.

#

Req PnP-REQ-FN-111 *[Mandatory]*

The PnP protocol and associated service interface shall permit the status of a SpaceWire router to be determined, providing such features are supported by the device, including:

- whether a watchdog time out error has occurred;
- whether a maximum packet size watchdog error has occurred;
- whether a packet address error has occurred.

#

3.1.1.3 Link Configuration and Status

Req PnP-REQ-FN-201 *[Mandatory]*

The PnP protocol and associated service interface shall permit the type of a SpaceWire port to be determined, including:

- configuration;
- SpaceWire;
- external.

#

Req PnP-REQ-FN-202 *[Mandatory]*

The PnP protocol and associated service interface shall permit all aspects of a link to be configured, providing such features are supported by the device, including:

- link state;
- transmit speed;
- time-code forwarding.

#

Req PnP-REQ-FN-203 *[Extended]*

The PnP protocol and associated service interface shall permit link distributed interrupt forwarding to be configured, if supported by the device.

#

Req PnP-REQ-FN-204 *[Mandatory]*

The PnP protocol and associated service interface shall permit the status of a link to be determined, providing status features are supported by the device, including:

- link state (running/not running);
- whether a disconnect error has occurred;
- whether a parity error has occurred;
- whether an escape error has occurred;
- whether a credit error has occurred;
- whether a character sequence error has occurred.

#

3.1.1.4 Protocol Status

Req PnP-REQ-FN-301 *[Mandatory]*

The PnP protocol and associated service interface shall permit the status of the PnP protocol handler on any device to be determined, including whether each possible error specific to the PnP protocol has occurred.

#

3.1.1.5 Data Acquisition

Req PnP-REQ-FN-401 *[Mandatory]*

The PnP protocol and associated service interface shall permit the acquisition of data from a device according to one of the following patterns:

- Get;

- Delayed-Response Get;
- Issue.

#

Req PnP-REQ-FN-402 *[Extended]*

The PnP protocol and associated service interface shall permit the interrupt-triggered acquisition of data from a device if interrupts are supported by the device and the necessary portion of the network.

#

3.1.1.6 Commanding

Req PnP-REQ-FN-501 *[Mandatory]*

The PnP protocol and associated service interface shall permit the commanding of a device according to one of the following patterns:

- Acknowledged Set;
- Unacknowledged Set;
- Acknowledged Delayed-Response Set;
- Unacknowledged Delayed-Response Set;
- Query.

#

Req PnP-REQ-FN-502 *[Extended]*

The PnP protocol and associated service interface shall permit the interrupt-triggered commanding of a device if interrupts are supported by the device and the necessary portion of the network.

#

3.1.1.7 Debugging

Req PnP-REQ-FN-601 *[Mandatory]*

The PnP protocol and associated service interface may permit debug aspects of a SpaceWire router to be configured, providing such features are supported by the device, including:

- enable self-addressing.

  Note: self-addressing refers to the ability of a router to route a packet out of the same port on which it is received. This is not a feature that it likely to be required onboard a spacecraft, but can be very useful during development and testing.

#

Req PnP-REQ-FN-602 *[Mandatory]*

The PnP protocol and associated service interface may permit the debug status of a SpaceWire link to be determined, providing debug status features are supported by the device, including:

- current transmit flow control credit;
- current receive flow control credit.

#

Req PnP-REQ-FN-603 *[Mandatory]*

The PnP protocol and associated service interface may permit the debug status of a SpaceWire device to be determined, providing debug status features are supported by the device, including:

- device version;
- device version build date.

#

Req PnP-REQ-FN-604 *[Mandatory]*

The PnP protocol and associated service interface may permit a device to visually identify itself when requested for debugging purposes.

Note: this is a very useful feature when test and development involves a large rack of equipment where each device may look identical. The visual identification can be achieved by flashing the device's LEDs, for example. This can also be very useful in a demonstration system, for example to indicate the device currently being configured.

#

3.1.1.8 Device Ownership

Req PnP-REQ-FN-701 *[Mandatory]*

The PnP protocol and associated service interface shall ensure that each device to be detected, monitored and configured by the protocol is owned by a single device on the network.

#

Req PnP-REQ-FN-702 *[Mandatory]*

The PnP protocol and associated service interface shall provide a mechanism by which devices may attempt to claim ownership of devices.

#

Req PnP-REQ-FN-703 *[Mandatory]*

The PnP protocol and associated service interface shall ensure that in conflicts of device ownership all competing owners are aware of the outcome.

#

Req PnP-REQ-FN-704 *[Extended]*

The PnP protocol and associated service interface shall provide a mechanism by which devices may relinquish ownership of devices.

#

Req PnP-REQ-FN-705 *[Mandatory]*

Where a device is an owner the PnP protocol and associated service interface shall provide an owner-proxy mechanism by which other devices may access any owned device in a controlled manner.

#

3.1.1.9 Service Interface

Req PnP-REQ-FN-801 *[Mandatory]*

The PnP protocol service interface shall permit the network topology and any topology changes to be determined automatically.

Note: this may be carried out via polling or via explicit notification

#

Req PnP-REQ-FN-802 *[Mandatory]*

The PnP protocol service interface shall permit the network topology and any topology changes to be determined only when explicitly invoked.

#

Req PnP-REQ-FN-803 *[Mandatory]*

The PnP protocol service interface shall permit changes to device operational status to be detected via explicit invocation.

#

Req PnP-REQ-FN-804 *[Mandatory]*

The PnP protocol service interface shall permit changes to device operational status to be detected automatically.

Note: This may be carried out using polled discovery or notification.

#

Req PnP-REQ-FN-805

The PnP protocol service interface shall indicate topology changes to the service user when enabled to detect changes automatically.

Note: the manner of this notification is not specified, but it may be in a TM.

#

Req PnP-REQ-FN-806 *[Mandatory]*

The PnP protocol service interface shall indicate any errors in network discovery or device management to the service user immediately.

Note: the manner of this notification is not specified, but it may be in a TM.

#

Req PnP-REQ-FN-807 *[Mandatory]*

The PnP protocol service interface shall permit a device start-up time to be configured during which a device is not expected to respond to protocol operations.

Note: this is to permit the slow start-up of devices.

#

Req PnP-REQ-FN-808 *[Mandatory]*

The PnP protocol service interface shall permit the number of detection attempts to be made on a device before reporting an error to be configured.

#

3.1.2 Coverage Requirements

3.1.2.1 Service Interface

Req PnP-REQ-CV-001 *[Extended]*

The PnP protocol service interface shall not place restrictions on network topology.

Note: the service interface is abstract and should therefore not place any restrictions on topology. However, see Protocol Coverage Requirements (Section 3.1.2.2) for restrictions which may be imposed by the protocol.

#

Req PnP-REQ-CV-002 *[Extended]*

The PnP protocol service interface shall not place restrictions on the number of devices in a network.

Note: the service interface is abstract and should therefore not place any restrictions on topology. However, see Protocol Coverage Requirements (Section 3.1.2.2) for restrictions which may be imposed by the protocol.

#

Req PnP-REQ-CV-003 *[Extended]*

The PnP protocol service interface shall not place restrictions on the number of devices simultaneously engaged in network discovery and management.

> Note: the service interface is abstract and should therefore not place any restrictions on topology. However, see Protocol Coverage Requirements (Section 3.1.2.2) for restrictions which may be imposed by the protocol.

#

Req PnP-REQ-CV-004 *[Extended]*

The PnP protocol service interface shall not place restrictions on the types of devices in a network.

> Note: the service interface must permit several identical devices within a network

#

Req PnP-REQ-CV-005 *[Extended]*

The PnP protocol service interface shall not place restrictions on the network topology changes which can be detected and managed.

> Note: the service interface is abstract and should therefore not place any restrictions on topology. However, see Protocol Coverage Requirements (Section 3.1.2.2) for restrictions which may be imposed by the protocol.

#

3.1.2.2 SpaceWire Protocol

Req PnP-REQ-CV-101 *[Extended]*

The PnP protocol may restrict the number of devices in a network to a maximum number of devices.

> Note: unique identification of a device is likely to involve the use of a 32-bit register, so this may limit the maximum number of devices to e.g. $2^{32} - 1$).

#

Req PnP-REQ-CV-102 *[Extended]*

The PnP protocol shall not place restrictions on network topology, other than the maximum number of devices permitted in the network.

#

Req PnP-REQ-CV-103 *[Extended]*

The PnP protocol shall not place restrictions on the number of devices simultaneously engaged in network discovery and management, other than the maximum number of devices permitted in the network.

#

Req PnP-REQ-CV-104 *[Extended]*

The PnP protocol shall not place restrictions on the types of devices in a network.

Note: the protocol must permit several identical devices within a network

#

Req PnP-REQ-CV-105 *[Extended]*

The PnP protocol shall not place restrictions on the network topology changes which can be detected and managed, other than the maximum number of devices permitted in the network.

#

Req PnP-REQ-CV-106 *[Extended]*

The PnP protocol shall not place restrictions on the state of a device which is added to a network.

Note: a device added to a network managed by the PnP protocol may be in an arbitrary state.

#

Req PnP-REQ-CV-107 *[Extended]*

The PnP protocol shall not restrict the simultaneous use of other nominal communications on the network.

Note: this is especially the use of RMAP, PTP, PUS or others protocols and assumes that these other protocols are not consuming 100% of the available bandwidth on any link (this is the case for present nominal operations).

#

Req PnP-REQ-CV-108 *[Mandatory]*

The PnP protocol shall not place restrictions on the type of addressing (path or logical) used for nominal communications.

Note: this includes the possibility of group adaptive routing and packet distribution.

#

Req PnP-REQ-CV-109 *[Extended]*

The PnP protocol shall not place restrictions on the assignment of logical addresses.

#

Req PnP-REQ-CV-110 *[Extended]*

The PnP protocol shall not place restrictions on the configuration of routers in the network.

#

Req PnP-REQ-CV-111 *[Extended]*

The use of the PnP protocol shall not preclude the use of non-PnP enabled devices in the network.

#

3.1.3 Performance Requirements

3.1.3.1 Service Interface

Req PnP-REQ-PF-001 [Mandatory]

The PnP protocol service interface shall permit network/device changes to be detected within a period of at least 1 second per network change.

> Note: i.e. a network change involving two devices becoming connected simultaneously must be detectable within 2 seconds.

#

Req PnP-REQ-PF-002 *[Extended]*

All operations offered by the PnP protocol service interface shall complete under all network conditions.

> Note: no matter what the state of the network the protocol shall never dead-lock or live-lock, i.e. operations shall always complete or abort.

#

3.1.3.2 SpaceWire Protocol

Req PnP-REQ-PF-101 *[Mandatory]*

The PnP protocol shall permit network/device changes to be detected within a period of at 1 second per network change.

> Note: i.e. a network change involving two devices becoming connected simultaneously must be detectable within 2 seconds.

#

Req PnP-REQ-PF-102 *[Mandatory]*

The PnP protocol may permit the time period within which a network change should be detected to be increased beyond the limit of 1 second when used in a test environment.

> Note: increasing this time period may be necessary when virtual spacecraft integration is used to integrate components virtually using the Internet, where latency and bandwidth is greatly reduced.

#

3.1.4 Design Requirements

3.1.4.1 Writing Rules

Req PnP-REQ-DS-001 *[Mandatory]*

The specification of the PnP protocol shall follow the rules as described in [AD5] and [AD6].

> Note: This will allow quicker path towards ECSS standardisation of SpW-PnP once the SpaceWire Working Group has approved it.

#

3.1.4.2 Service Interface

Req PnP-REQ-DS-101 *[Mandatory]*

The PnP protocol service interface shall be designed to be as simple as possible.

> Note: i.e. the number of primitives should be minimised and redundancy between primitives should be minimised. Additionally, coupling between primitives should be minimised.

#

3.1.4.3 SpaceWire Protocol

Req PnP-REQ-DS-201 *[Mandatory]*

The PnP protocol shall be designed to be as simple as possible.

> Note: i.e. the number of packet format variations should be minimised and redundancy between packet format variations should be minimised. Additionally, the protocol semantics should be kept as simple as possible.

#

Req PnP-REQ-DS-202 *[Mandatory]*

The PnP protocol shall utilise the RMAP packet format and underlying protocol semantics.

#

Req PnP-REQ-DS-203 *[Mandatory]*

PnP protocol packets shall be easily identifiable from other SpaceWire packets.

> Note: the PnP Protocol must use its own Protocol ID to distinguish it from other RMAP packets.

#

Req PnP-REQ-DS-204 *[Mandatory]*

The use of the CRC in verifying data for a PnP protocol set (or write) command shall be optional.

Note: it is assumed that a set, or write, is based on an RMAP write command which provides 'verify' as an option; this should be retained as an option.

#

Req PnP-REQ-DS-205 *[Mandatory]*

The generation of a reply is response to a PnP protocol set (or write) command shall be optional.

Note: it is assumed that a set, or write, is based on an RMAP write command which provides 'acknowledge' as an option; this should be retained as an option.

#

3.1.5 Interface Requirements

3.1.5.1 Service Interface

Req PnP-REQ-IF-001 *[Mandatory]*

The PnP protocol service interface shall be designed to align with the CCSDS SOIS subnetwork services.

#

3.1.5.2 Device Compatibility

Req PnP-REQ-IF-101 *[Mandatory]*

The PnP protocol shall operate with SpaceWire routers as defined in [AD1].

#

Req PnP-REQ-IF-102 *[Mandatory]*

The PnP protocol shall operate with existing SpaceWire routers as defined in [].

#

3.1.6 Environmental Requirements

N/A

3.1.7 Physical and Resource Requirements

3.1.7.1 SpaceWire Protocol

Req PnP-REQ-PR-001 *[Mandatory]*

The PnP protocol shall be designed so that a reasonably optimised implementation (in terms of hardware and/or VHDL and/or software) fits into devices and units currently used on ESA spacecraft units (including “small” units such as minor payload instruments).

Note: by implication this includes the implementation of the protocol on a LEON2 device such as the RTC [].

#

Req PnP-REQ-PR-002 *[Mandatory]*

It shall be possible to implement the PnP protocol on a device which is only capable of responding to PnP commands, not initiating them.

Note: a device which does not operate as the initiator of commands should only require RMAP target-like functionality.

#

Req PnP-REQ-PR-003 *[Mandatory]*

It shall be possible to implement the PnP protocol in an ASIC without the need for an accompanying memory device.

Note: these devices will not be capable of implementing a unique MAC address.

#

3.1.8 Operational Requirements

N/A

3.1.9 Human Factors Requirements

N/A

3.1.10 Logistics Support Requirements

N/A

3.1.11 Product Assurance Requirements

N/A

3.1.12 Configuration and Implementation Requirements

N/A

3.1.13 Verification and Testing Requirements

N/A

3.2 Traceability

3.2.1 Functional Requirements

Requirement	Source Requirements
PnP-REQ-FN-001	PnP-SYS-FN-201, PnP-SW-FN-001, PnP-SW-FN-101, PnP-EQ-FN-001
PnP-REQ-FN-002	PnP-SYS-FN-201, PnP-EC-FN-001, PnP-EC-CV-001, PnP-EC-CV-101
PnP-REQ-FN-003	PnP-SYS-FN-201, PnP-SYS-FN-209
PnP-REQ-FN-004	PnP-SYS-FN-206, PnP-SYS-FN-301, PnP-SYS-FN-501, PnP-SW-FN-002, PnP-SW-FN-102, PnP-EQ-FN-002, PnP-EC-FN-003
PnP-REQ-FN-005	PnP-SYS-FN-206, PnP-SYS-FN-301, PnP-EQ-FN-303, PnP-EC-FN-201
PnP-REQ-FN-006	PnP-EQ-FN-303, PnP-EC-FN-201
PnP-REQ-FN-007	PnP-EC-FN-301
PnP-REQ-FN-008	PnP-SYS-FN-401, PnP-SW-FN-051, PnP-SW-FN-151, PnP-EQ-FN-003, PnP-EC-FN-001, PnP-EC-FN-002
PnP-REQ-FN-009	PnP-SW-FN-052, PnP-SW-FN-152
PnP-REQ-FN-010	PnP-SW-FN-021, PnP-SW-FN-121
PnP-REQ-FN-101	PnP-SW-FN-013, PnP-SW-FN-113, PnP-EQ-FN-103
PnP-REQ-FN-102	PnP-SW-FN-013, PnP-SW-FN-113, PnP-EQ-FN-103, PnP-EC-FN-601
PnP-REQ-FN-103	PnP-SW-FN-014, PnP-SW-FN-114
PnP-REQ-FN-104	PnP-EC-FN-701
PnP-REQ-FN-105	PnP-EQ-FN-105, PnP-EC-FN-702
PnP-REQ-FN-106	PnP-EQ-FN-106, PnP-EC-FN-703
PnP-REQ-FN-107	PnP-EC-FN-704
PnP-REQ-FN-108	PnP-SW-FN-003, PnP-SW-FN-103, PnP-EQ-FN-101, PnP-EC-FN-202
PnP-REQ-FN-109	PnP-EQ-FN-102
PnP-REQ-FN-110	PnP-SW-FN-154
PnP-REQ-FN-111	PnP-SW-FN-016, PnP-SW-FN-116, PnP-EQ-FN-104, PnP-EC-FN-601

PnP-REQ-FN-201	PnP-EQ-FN-201
PnP-REQ-FN-202	PnP-SW-FN-011, PnP-SW-FN-111, PnP-EQ-FN-202, PnP-EC-FN-501, PnP-EC-FN-601
PnP-REQ-FN-203	PnP-SW-FN-012, PnP-SW-FN-112
PnP-REQ-FN-204	PnP-SW-FN-015, PnP-SW-FN-115, PnP-EQ-FN-203, PnP-EQ-FN-204, PnP-EQ-FN-205, PnP-EC-FN-202, PnP-EC-FN-501
PnP-REQ-FN-301	PnP-EC-FN-203
PnP-REQ-FN-401	PnP-SW-FN-031, PnP-SW-FN-131, PnP-EC-FN-801, PnP-EC-FN-802, PnP-EC-FN-803
PnP-REQ-FN-402	PnP-SW-FN-032, PnP-SW-FN-132
PnP-REQ-FN-501	PnP-SW-FN-041, PnP-SW-FN-141, PnP-EC-FN-901, PnP-EC-FN-902, PnP-EC-FN-903, PnP-EC-FN-904
PnP-REQ-FN-502	PnP-SW-FN-042, PnP-SW-FN-142
PnP-REQ-FN-601	PnP-EQ-FN-301
PnP-REQ-FN-602	PnP-EQ-FN-302
PnP-REQ-FN-603	PnP-EQ-FN-303
PnP-REQ-FN-604	PnP-EQ-FN-304
PnP-REQ-FN-701	PnP-EC-FN-101
PnP-REQ-FN-702	PnP-EC-FN-101
PnP-REQ-FN-703	PnP-EC-FN-101, PnP-EC-FN-202
PnP-REQ-FN-704	PnP-EC-FN-102
PnP-REQ-FN-705	PnP-EC-FN-401
PnP-REQ-FN-801	PnP-SW-FN-052, PnP-SW-FN-152, PnP-EC-FN-001, PnP-EC-FN-002
PnP-REQ-FN-802	PnP-SYS-FN-207, PnP-SYS-FN-209, PnP-SW-FN-051, PnP-SW-FN-151
PnP-REQ-FN-803	PnP-SW-FN-053, PnP-SW-FN-153
PnP-REQ-FN-804	PnP-SW-FN-054, PnP-SW-FN-153
PnP-REQ-FN-805	PnP-SYS-FN-205, PnP-SYS-FN-208, PnP-SYS-FN-303, PnP-SYS-FN-402

PnP-REQ-FN-806	PnP-SYS-FN-208, PnP-SYS-FN-302, PnP-SYS-FN-303, PnP-SYS-FN-402
PnP-REQ-FN-807	PnP-SYS-FN-305
PnP-REQ-FN-808	PnP-SYS-FN-306

3.2.2 Coverage Requirements

Requirement	**Source Requirements**
PnP-REQ-CV-001	PnP-SYS-CV-001, PnP-SW-CV-001, PnP-EQ-CV-002, PnP-EC-CV-001, PnP-EC-CV-101
PnP-REQ-CV-002	PnP-SYS-CV-001, PnP-SW-CV-002, PnP-EQ-CV-001
PnP-REQ-CV-003	PnP-SYS-FN-101, PnP-SW-FN-062, PnP-SW-CV-003, PnP-EQ-FN-005, PnP-EQ-CV-003, PnP-EC-CV-102
PnP-REQ-CV-004	PnP-SYS-FN-501, PnP-SYS-CV-002, PnP-SW-FN-061, PnP-EQ-FN-004, PnP-EC-CV-004
PnP-REQ-CV-005	PnP-EC-CV-002
PnP-REQ-CV-101	PnP-SW-CV-102, PnP-EQ-CV-001
PnP-REQ-CV-102	PnP-SYS-CV-001, PnP-SW-CV-101, PnP-EC-CV-001, PnP-EC-CV-101
PnP-REQ-CV-103	PnP-SYS-CV-001, PnP-SW-FN-162, PnP-SW-CV-103, PnP-EQ-FN-005, PnP-EQ-CV-003, PnP-EC-CV-102
PnP-REQ-CV-104	PnP-SYS-FN-501, PnP-SYS-CV-002, PnP-SW-FN-161, PnP-EQ-FN-004, PnP-EC-CV-004
PnP-REQ-CV-105	PnP-EC-CV-002
PnP-REQ-CV-106	PnP-SYS-FN-005, PnP-EC-CV-003
PnP-REQ-CV-107	PnP-SYS-FN-002, PnP-SYS-CV-002
PnP-REQ-CV-108	PnP-SYS-FN-003
PnP-REQ-CV-109	PnP-SYS-FN-005
PnP-REQ-CV-110	PnP-SYS-FN-004, PnP-EC-CV-003
PnP-REQ-CV-111	PnP-SYS-FN-204

3.2.3 Performance Requirements

Requirement	Source Requirements
PnP-REQ-PF-001	PnP-SYS-FN-304, PnP-SYS-FN-403, PnP-SW-PF-001, PnP-EQ-CV-001, PnP-EC-PF-001, PnP-EC-PF-101
PnP-REQ-PF-002	PnP-SYS-PF-001, PnP-EC-FN-101, PnP-EC-CV-102, PnP-EC-CV-103
PnP-REQ-PF-101	PnP-SYS-FN-304, PnP-SYS-FN-403, PnP-SW-PF-101, PnP-EQ-CV-001, PnP-EC-PF-001, PnP-EC-PF-101
PnP-REQ-PF-102	PnP-EQ-CV-002

3.2.4 Design Requirements

Requirement	Source Requirements
PnP-REQ-DS-001	PnP-EC-DS-001
PnP-REQ-DS-101	PnP-SW-DS-001, PnP-EQ-DS-001
PnP-REQ-DS-201	PnP-SW-DS-101, PnP-EQ-DS-001
PnP-REQ-DS-202	PnP-SYS-FN-001, PnP-SW-DS-102, PnP-EQ-DS-002, PnP-EQ-DS-102, PnP-EC-DS-101, PnP-EC-IF-001
PnP-REQ-DS-203	PnP-SYS-FN-001, PnP-EQ-DS-101
PnP-REQ-DS-204	PnP-EC-DS-102
PnP-REQ-DS-205	PnP-EC-DS-103

3.2.5 Interface Requirements

Requirement	Source Requirements
PnP-REQ-IF-001	PnP-SW-IF-001
PnP-REQ-IF-101	PnP-EQ-IF-001, PnP-EC-IF-101
PnP-REQ-IF-102	PnP-EQ-IF-001

3.2.6 Environmental Requirements

N/A

3.2.7 Physical and Resource Requirements

Requirement	Source Requirements
PnP-REQ-PR-001	PnP-SW-PR-001, PnP-EC-PR-001
PnP-REQ-PR-002	PnP-EQ-PR-001
PnP-REQ-PR-003	PnP-EQ-PR-002

3.2.8 Operational Requirements

N/A

3.2.9 Human Factors Requirements

N/A

3.2.10 Logistics Support Requirements

N/A

3.2.11 Product Assurance Requirements

N/A

3.2.12 Configuration and Implementation Requirements

N/A

3.2.13 Verification and Testing Requirements

N/A

End of Document