# NETWORK MANAGEMENT AND CONFIGURATION USING RMAP

**Session: SpaceWire Standardisation**

**Long Paper**

Peter Mendham, Stuart Mills, Steve Parkes

*Space Systems Research Group*

*School of Computing, University of Dundee, Dundee, DD1 4HN, Scotland*

*E-mail: petermendham@computing.dundee.ac.uk, smills@computing.dundee.ac.uk, sparkes@computing.dundee.ac.uk*

## ABSTRACT

The success of the SpaceWire standard has resulted in the availability of a wide variety of SpaceWire devices. Allowing these devices to inter-operate easily is an open-problem with growing importance. This problem has two different manifestations: interoperability in ground and test equipment, where ease of use is the main driver; and interoperability of flight hardware where improvements could ease both hardware and software reuse and lower costs.

This paper proposes a standard mechanism for network management and configuration of network devices building on the remote memory access protocol (RMAP). The paper argues that RMAP is an appropriate starting point as it layers well in a protocol stack, and many vendors have existing RMAP capabilities. Adding network management and configuration features in this way could be a fairly easy task for both existing and future SpaceWire equipment.

## 1 INTRODUCTION

Since the publication of the SpaceWire standard by the European Cooperation on Space Standardization (ECSS) in January 2003 [1], SpaceWire has emerged as one of the main data-handling networks for use onboard spacecraft. It is now being used on many ESA, NASA and JAXA spacecraft and by research organisations and space industry across the world. This widespread interest has resulted in the availability of a large number of SpaceWire devices, however, there is no standard way to interrogate or configure these devices, limiting the extent to which existing hardware can interoperate and both hardware and software can be reused between missions. This paper examines what would be required to ensure a basic level of interoperability, and proposes a solution using the Remote Memory Access Protocol (RMAP) [2].

The next section discusses interoperability, and its implications, for both ground and flight. The outcomes of this discussion serve as a set of requirements for the rest of the paper. The following section gives a brief overview of the RMAP protocol and presents the argument for using it here. The paper then presents the technical details of the proposal examining the two different parts of the interoperability problem separately: network management and network configuration.

This work has a close relationship with the proposed Plug-And-Play Standard for SpaceWire; the penultimate section of the paper clarifies the relationship and presents a small amount of historical background. The final section summarises and concludes the discussions.

## 2 SPACEWIRE INTEROPERABILITY

One of the greatest assets of SpaceWire, and perhaps the most important reason for its popularity, is its inherent simplicity. SpaceWire CODECs and routers require relatively few gates, saving cost, mass and power. SpaceWire packets are simply the routing addresses, followed by the data cargo and an end of packet marker. SpaceWire links are relatively simple; at all times they must be in one of six states, furthermore, the standard gives clear indications of the ways these states should be controlled. As part of the standardisation of SpaceWire addressing, the SpaceWire standard gives a clear discussion of a many features that routers must, or can, support. Despite this, there is no standard way to interrogate, or configure these features. A vendor-agnostic method for managing the standard features of SpaceWire Devices would bring a wide range of benefits.

### 2.1 GROUND AND TEST EQUIPMENT

Interoperability standards would provide benefits for ground-based systems in two main areas: development and test. In the first case, a developer may wish to set up a SpaceWire network for prototyping and simulation. It is likely that the network will use equipment supplied by a number of different vendors. Currently, this means that vendor specific software, or other mechanisms, must be used to configure each of the devices. Should any network management functions be required, such as detecting the status of individual links, or the topology of the network, vendor specific functions are again required, in some cases a single operation that involves multiple devices may be forced to use multiple software libraries from different vendors, each with their own conventions.

By providing standard methods for the interrogation of SpaceWire networks, test equipment can discover network topologies and provide diagnoses on network problems in real time. Such features could be especially useful during final spacecraft integration. With knowledge of vendor-agnostic methods of interrogating routers and devices, electrical ground support equipment (EGSE) could identify and pinpoint problems with flight devices, and network configuration.

### 2.2 FLIGHT EQUIPMENT

Although a spacecraft is generally a closed system, similar inter-operability problems are faced. If flight devices from multiple vendors are used, vendor specific mechanisms must be employed to, for example, configure routing tables. This limits the degree to which software and other spacecraft components can be re-used. Standard methods for interrogating a network could permit flight software to easily determine the current status of SpaceWire links to diagnose problems or confirm the availability of a device, perhaps in response to a mode change. Utilising a vendor-agnostic standard for these purposes reduces the amount of testing that must be done, lowers risk and increases the amount of reuse possible within, and between, missions.

## 3 THE REMOTE MEMORY ACCESS PROTOCOL (RMAP)

The Remote Memory Access Protocol is a simple yet flexible method for querying and configuring a SpaceWire device. RMAP provides commands for:

- Write
- Read
- Read/Modify/Write

Additionally, RMAP provides support for an 8-bit CRC which may optionally be used to check the data contents of a write operation (a verified write). The source of a write command may also request an acknowledgement (an acknowledged write) which will return a standard status code. The two may be combined to form a verified, acknowledged write. A read operation is straightforward, with the reply packet containing the requested data.

Read/modify/write (RMW) operations are slightly more complicated. An RMW packet contains two fields: a data value and a mask value. The operation first performs a read, and responds with the data. It then uses the mask and data fields to perform a write in an implementation specific way.

All RMAP commands use an address field to specify the data on which to perform the operation. As there are no restrictions on the values in this field, the protocol imposes no semantics above the three command types.

Additionally, RMAP provides:

- A transaction identifier to permit the tracking of transactions by the host requesting the operation. This requires no extra logic at the device.
- A destination key, essentially a ‘magic cookie’, included in every command must be matched by the device before the operation is allowed to continue.

Either path or logical addressing may be used with RMAP.

### 3.1 BENEFITS OF RMAP

Read and write operations form the basis for many protocols and, combined with the verification and acknowledgement facilities, RMAP represents a versatile lowest layer in a protocol stack. The lack of complex semantics ensures the protocol’s flexibility. If facilities, such as transaction identifiers and destination keys, are not required the corresponding logic may not be implemented and the overhead in packet size is minimal.

A number of manufacturers and organisations have found RMAP useful and have implemented hardware and software solutions for devices and hosts. Any protocol using RMAP as a lower layer would be able to leverage existing intellectual property, limiting the risk for adopters of new technology.

### 3.2 Interaction with the Protocol Identifier

RMAP works within the proposed Protocol Identification standard [3], which places a centrally assigned numerical identifier after the address. If a logical address is not used, a byte of padding (essentially a fake logical address, usually with the value 254) must be inserted so that the packet arriving at the device is consistent.

To configure or manage routers, the standard specifies that port zero is reserved for this purpose. To permit the interrogation of all devices, without *a priori* knowledge of their type, nodes must also respect a packet arriving as if it is destined for port zero. In this case a node should recognise the packet as a configuration packet, if it supports that feature, or it should discard the packet. This behaviour has been proposed as an alteration to the Protocol Identification standard.

RMAP has a dedicated protocol identifier of 01h. Unfortunately, whilst this identifies the packet contents as RMAP, it does not describe any semantics imposed on the RMAP commands. For this reason, this proposal uses an alternative protocol identifier, which specifies network management and configuration using RMAP.

### 3.3 Specifying an RMAP Return Address

If a host is sending an RMAP read command to an unknown device in order to, for example, determine its type, the host may not know the port number to which it is connected. In this case it is impossible for the host to specify a return address. To solve this problem, we propose that any packet requiring a response which specifies a Source Logical Address of zero will have its return address modified to include the path out of the port through which the request was received.

### 3.4 Use of RMAP Address Space

RMAP has a 40-bit address field (including the extended address byte) of which this proposal uses only a fraction. To ease both packet formation and decoding, the address parameter is split into three fields, each a byte long. These form the lowest three bytes of the address field; the upper two bytes are not used. The fields are:

- A command field, which identifies the structure or table being addressed.
- An index field, which specifies an entry in a table; this is not used for structures.
- A byte offset field which permits a read or write command to offset into a structure or table entry.

Addressing is therefore byte-wide in this proposal.

## 4 Network Management Tasks

Network management, in this context, involves a network manager detecting or verifying the topology of the network, the devices that are connected, and the status of the SpaceWire links that connect them. This section first reviews the parameters that would be useful for network management and then proposes data structures for storing the parameters. In this proposal, all network management parameters are read-only.

## 4.1 DEVICE INFORMATION PARAMETERS

Device information permits network managers to identify SpaceWire devices, and their capabilities. The device would need to specify the following:

- Whether the device is a node or a router.
- The number of ports the device has.

Additionally, a device may wish to identify itself by specifying:

- A vendor identifier; centrally assigned to be unique for each vendor.
- A product identifier, assigned by the vendor.
- A device class identifier; centrally assigned to describe the function of this device.
- Version information.

It may also be important to uniquely identify a device on a network. This could be achieved by specifying:

- A globally unique device identifier.

For maximum flexibility, a vendor may decide not to assign this parameter, and to allow it to be written to. For this reason, a device identifier is included in the data structures for network configuration (see below).

An RMAP verified write operation requires the device to store the contents of the whole packet so that it can verify the contents against the CRC. Any practical device will have limited buffer space. A device may therefore wish to specify:

- The maximum allowed write packet size.

## 4.2 LINK STATUS MONITORING PARAMETERS

For network management, a host may wish to determine the following:

- The status of all links attached to this device (whether or not they are in the 'run' state).
- Whether any errors have been detected on the links.
- The maximum speed the links may be run at.

The SpaceWire standard specifically mentions the following errors:

- Disconnect errors.
- Parity errors.
- Escape errors.

- Transmit credit errors.
- Receive credit errors.
- Character sequence errors.

The maximum speed of a link must be greater than or equal to 10 Mbit/s, the start-up speed of SpaceWire links.

## 4.3 Network Discovery Parameters

Using the device information presented so far, a host may begin to explore the network and determine what devices are present. To identify the route that the host is using to interrogate the device, the device should make available:

- The port number through which the current request was received.

## 4.4 Router Information Parameters

During packet routing, routers may use a variety of arbitration mechanisms to determine access to an output port between competing packets. It may choose to use priorities for physical ports, logical addresses, or both. A router should therefore specify:

- The maximum priority that may be used for physical port arbitration.
- The maximum priority that may be used for logical address arbitration.

## 4.5 Device Information Structures

The parameters introduced above are organised into two structures:

- A device information structure for parameters that are common to both nodes and routers.
- A router identification structure for routers only.

These structures are shown in Figure 1 and Figure 2 respectively. The “Active Port Bitmask” in Figure 1 identifies which ports are in the ‘run’ state. Bit 1 set to a ‘1’ indicates that port 1 is running. Bit 0 is not used and is always set to a ‘0’.

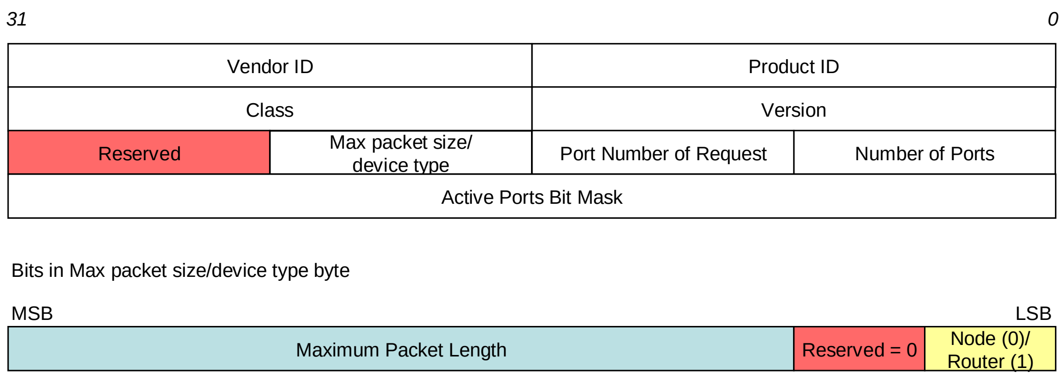

**Figure 1: Device Information Structure**

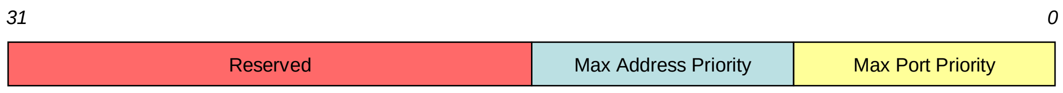

**Figure 2: Router Information Structure**

### 4.6 Port Status Table

The port status table contains 32 entries, one for each physical port from 0-31. The first entry, referencing the configuration port, is not used. A port status table entry is shown in Figure 3.

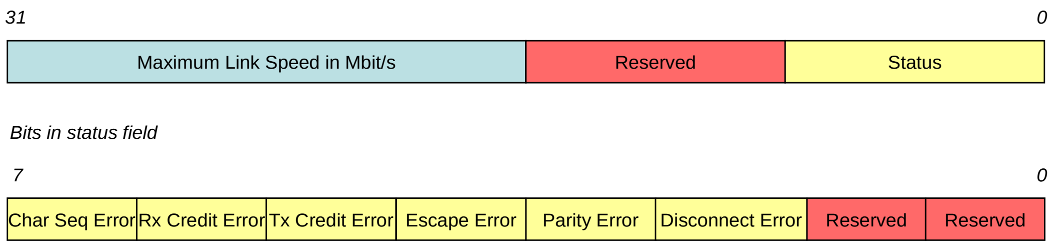

**Figure 3: Port Status Table Entry**

## 5 Network Configuration Tasks

During network configuration a host configures the device for operation. For all devices, a host must be able to configure each of the SpaceWire links. For routers only, the host must be able to configure the way in which arbitration is carried out, and to set-up the routing table.

### 5.1 Link Configuration

To configure a link, a host must be able to:

- Reset each of the errors in the port status table.
- Set the state of the link.
- Set the speed of the link.

The state of a link may be:

- Idle – if the link is not running, it won't start; if it is running it will continue to run.
- Start – start the link.
- Auto-start – listen for NULLs and start the link if they are received.
- Disable – if the link is not running, it won't start; if it is running it will be stopped immediately.

The link speed may be set to any value between 10 Mbit/s and the supported maximum. A device may choose to alter the value to the nearest supported speed below the one specified.

For arbitration, routers may need to associate a priority with each link (see below).

## 5.2 Router Configuration

A routing device may be configured to set the arbitration mode that it uses to arbitrate between multiple packets competing for the same output port.

Using information provided in the SpaceWire standard, the arbitration of packets competing for an output port can be decided using a combination of up to three techniques, applied in the order specified below:

1. Address Priority uses the destination address of the received packet. Arbitration is carried out by comparing the priorities assigned to the destination addresses in the routing table (see below). The highest arbitration priority wins.
2. Port Priority uses the port at which the packet arrives. Arbitration is carried out by comparing the priorities assigned to the arrival port in the port configuration table (see below). The highest arbitration priority wins.
3. If competition still exists, an arbitration mode can be chosen by selecting a arbitration method such as random, round robin, fixed, etc.

A host should be able to enable and disable these features individually. If a router chooses not to support one or more of these methods the host will not be able to enable that feature. A router must support fixed arbitration as a minimum.

To enable logical address-based routing, a router must allow a host to associate the following with each logical address:

- One or more physical ports, more to permit group adaptive routing or packet distribution.
- The option to remove the logical address from the packet.
- The option to enable packet distribution where the packet is forwarded to every port in the group rather that only one. If the arrival port is included on the list the packet is only forwarded to other ports.
- A routing priority.

## 5.3 PORT CONFIGURATION TABLE

All devices have a port configuration table with as many entries as the device has physical ports. There is no entry for the configuration port. Entries are as shown in Figure 4.

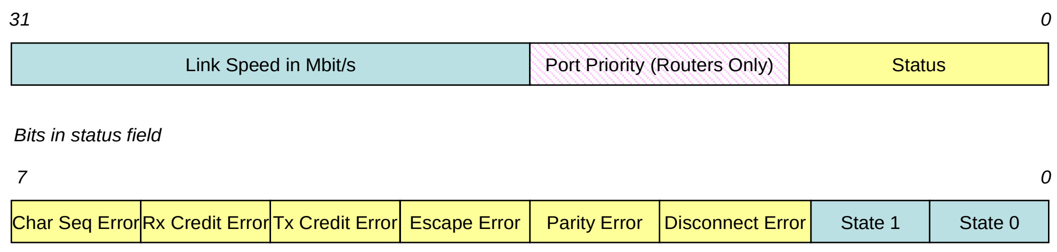

**Figure 4: Port Configuration Table Entry**

The status field is a copy of that given in the equivalent port status stable entry, except that writing a '1' to an error bit resets that error. The two state bits are encoded as follows:

0. Idle

1. Start

2. Auto-start

3. Disable

The port priority field is reserved for nodes.

## 5.4 ROUTER CONFIGURATION STRUCTURE AND ROUTING TABLE

To configure the arbitration mode on routers, a router configuration structure is provided; this is shown in Figure 5.

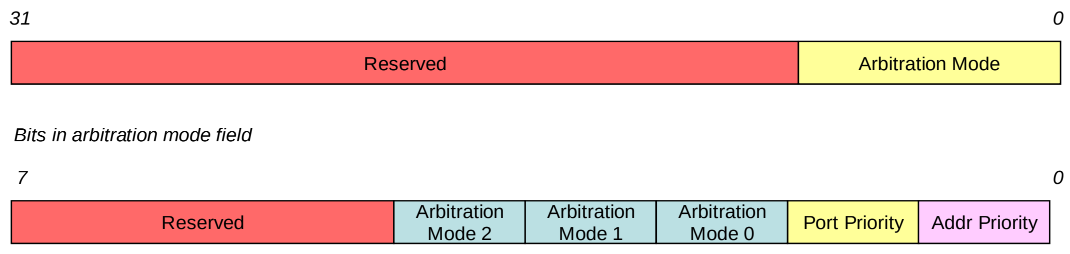

**Figure 5: Router Configuration Structure**

The arbitration mode field decodes as:

0. Fixed arbitration

1. Round-robin arbitration

2. Random arbitration

All other values are vendor specific.

The routing table has entries which control the routing of packets for logical addresses (see Figure 6). The most important field is the port association bitmask, which indicates the ports that are to be used for routing the selected logical address. Bit 1 indicates port 1 and so on. As routing is not permitted to the configuration port, the lowest bit is used to specify address deletion. A routing table entry is shown in Figure 6. An entry is provided for all addresses from 1 to 254.

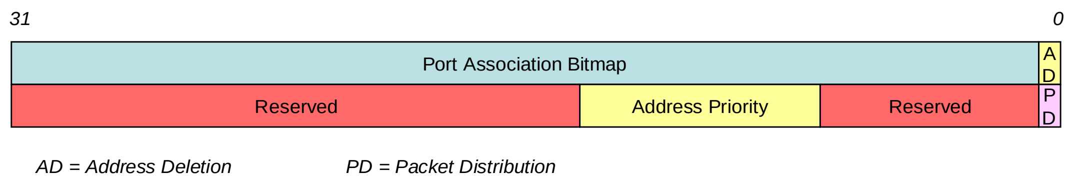

**Figure 6: Routing Table Entry**

## 6 RELATIONSHIP WITH THE PROPOSED PLUG AND PLAY STANDARD

This work has two starting points: the configuration space of the University of Dundee router, which is accessed using RMAP; and the proposals produced by the Plug-and-Play working party, a sub-group of the SpaceWire Working Group. The current proposal is accessible from the working party's discussion forum [4]. Although heavily indebted to the work that the plug-and-play group has done, this proposal has a slightly different focus, which is why the term plug-and-play has not been used here.

## 7 CONCLUSION

This paper argued the need for a standard method for network management and configuration in order to promote interoperability between SpaceWire devices from different vendors. Such interoperability would make equipment easier to use, permit operations such as network discovery in a consistent manner and enable higher levels of software and hardware reuse. As a network management and configuration protocol is largely based on the semantics of read/write (or set/get), this paper argues that RMAP, a protocol with increasing levels of vendor support, would be an appropriate starting point. To permit identification of the semantics imposed by the network management and configuration protocol, a distinct protocol identifier should be used.

The paper then identified the key features that could be considered as standard, derived from the SpaceWire standard itself, such as link status and speed, and router configuration. Facilities are also provided to permit network discovery.

This work has derived partially from the University of Dundee router configuration space, and partly from the draft specification produced by the SpaceWire plug-and-play working party. The next steps for this work are to: consider existing devices, and whether any special support is needed; include support for sharing the configuration of SpaceWire devices between multiple hosts without introducing conflicts; aligning this proposal closer with the work being done by the plug-and-play working party.

## 8 REFERENCES

1. European Cooperation for Space Standardization, Standard ECSS-E-50-12A, “SpaceWire – Link, Nodes, Routers and Networks”, Issue 1, European Cooperation for Space Standardization, January 2003.

2. European Cooperation for Space Standardization, Standard ECSS-E-50-11, “Remote Memory Access Protocol”, Draft F, European Cooperation for Space Standardization, December 2006.

3. European Cooperation for Space Standardization, Standard ECSS-E-50-11, “Protocol Identification”, Draft B, European Cooperation for Space Standardization, December 2006.

4. SpaceWire Plug-and-Play Working Party, “SpaceWire Plug-and-Play Specification”, http://tech.groups.yahoo.com/group/SpaceWirePnP/files/Draft%20Specification/, 2007