
Peter Mendham

25 February 2010
Agenda

- Requirements and aims
- RMAP usage
- SpaceWire-PnP services
  - Device Identification
  - Network Management
  - Link Configuration
  - Router Configuration
- Extensibility and capabilities
- Applying SpaceWire-PnP
- Known issues and discussion points
- How to use the UoD document
SpaceWire-PnP Aims

› Protocol aims
  › Interoperability and reuse
  › Standard mechanisms for standard features
  › Support device/network discovery as required by SOIS

› Document aims
  › A complete solution
  › A starting point for discussion
Perspective

› PnP views the network like the SpaceWire standard
  › Links
  › Nodes
  › Routers

{ Devices

› No topology restrictions

› Both nodes and routers have links
  › Nodes have 1 or more links
  › Routers have 2 or more links

› Every device on the network has a port zero
  › This is the target for PnP transactions
Levels of Support

- Managed Networks
  - Important role for system designer
  - Competition during discovery process removed by design
  - Competition for configuration of devices removed by design
  - Simplest case

- Open Networks
  - Network handles all competition issues
  - Deals with networks where design is not known *a priori*
  - More flexible but more complicated
What is Standardised?

- A set of parameters on the target
  - This is a standardised RMAP address space
- An interface of primitives at the initiator
  - Satisfying the requirements for SOIS
- A description of how the initiator and target will both behave
RMAP Utilisation

› Semantics required for plug-and-play closely match RMAP

› Use a well-defined implementation of RMAP
  › 32-bit wide addressing and alignment
  › Big endian
  › Incrementing addressing
  › Acknowledged, verified writes
  › Pre-defined key
  › RMW implementation (optional) is a conditional write

› Use a different protocol ID
  › To distinguish from generic RMAP traffic
  › E.g. Mass memory device
So is SpaceWire-PnP a Protocol?

› Probably not…
  › A specific implementation of RMAP
  › Standardised address space
  › Standardised primitives
  › Standard semantics of use
  › Identified with a protocol ID
› Does that make it a protocol?
Target Parameters

- Follow a regular form
- Parameters are made up of 32-bit fields
- Optionally, a parameter may have multiple entries
  - This is to permit tables, such as routing tables
  - The root entry has one set of fields
  - Every other non-root entry has a different but identical set of fields
- For example, the link configuration parameter
  - Root entry has one field giving the number of links
  - Has a non-root entry for each link, each of which has the same fields
Core Services

- Four core services defined
  - Device Identification
    - Read-only, constant fields
    - A few, mirrored, read-only dynamic fields
  - Network Management
  - Link Configuration
    - All devices
  - Router Configuration
    - Routers only
  - Optionally, there is also a time-code source

- Basic discovery
- Satisfies SOIS
- Necessary for SpaceWire-specific configuration
Device Identification Service

➢ Permits the gathering of device information
  ➢ Including type of device

➢ Parameters:
  ➢ Device Information
  ➢ Vendor String (Optional)
  ➢ Product String (Optional)
  ➢ Device Status
  ➢ Capability List
Device Information and Status

- Identifies the device
  - Vendor ID and Product ID (like PCI, USB etc.)
  - Type (node/router)
  - Number of ports
  - Optional static device ID
  - Vendor and Product string lengths

- Provides current status
  - Active ports
  - Device ID (non-static)
  - Return port

Read-Only and Constant (PROM)

Read-Only and Dynamic, Mirrored
## Example Parameter Fields

### Table 5-3: Device Information Parameter Fields

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Vendor ID/ Product ID</td>
<td>Contains 16-bit vendor and product IDs</td>
</tr>
<tr>
<td>1</td>
<td>Region/Number of Ports</td>
<td>Indicates preferred device region gives port count</td>
</tr>
<tr>
<td>2</td>
<td>Static Device ID High</td>
<td>High 32 bits of the 64-bit static device ID (if present)</td>
</tr>
<tr>
<td>3</td>
<td>Static Device ID Low</td>
<td>Low 32 bits of the 64-bit static device ID (if present)</td>
</tr>
<tr>
<td>4</td>
<td>Version/Instance ID</td>
<td>Version and System instance of this device type</td>
</tr>
<tr>
<td>5</td>
<td>Operation/String Lengths</td>
<td>Length of the vendor and product strings (can be zero)</td>
</tr>
<tr>
<td>6-31</td>
<td>Reserved</td>
<td>Reserved for future use</td>
</tr>
</tbody>
</table>
DI DS Primitives

- DI_DS_READ_INFO.request
- DI_DS_READ_INFO.indication
- DI_DS_READ_VENDOR_STRING.request
- DI_DS_READ_VENDOR_STRING.indication
- DI_DS_READ_PRODUCT_STRING.request
- DI_DS_READ_PRODUCT_STRING.indication
- DI_DS_READ_STATUS.request
- DI_DS_READ_STATUS.indication
- DI_DS_READ_CAPABILITY_LIST.request
- DI_DS_READ_CAPABILITY_LIST.indication
DIDS Example Initiator Primitive

- DIDS_READ_INFO.request
  - RMAP_Parameters
- DIDS_READ_INFO.indication
  - Result
  - Vendor_ID
  - Product_ID
  - PreferredRegion
  - Router_Node
  - Support_Level
  - Port_Count
  - Device_ID
  - Version
  - Instance_ID
Network Management Service

› Permits the unique identification of devices
› Enables network discovery
› Parameters:
  › Read-write network ID (just a 32-bit register)
  › Logical address (for nodes only, and optional)
NMS Primitives

- NMS_READ_NETWORK_ID.request
- NMS_READ_NETWORK_ID.indication
- NMS_WRITE_NETWORK_ID.request
- NMS_WRITE_NETWORK_ID.indication
- NMS_READ_DEVICE_LA.request
- NMS_READ_DEVICE_LA.indication
- NMS_WRITE_DEVICE_LA.request
- NMS_WRITEDEVICE_LA.indication
- NMS_DISCOVER_NETWORK.request
- NMS_DISCOVER_NETWORK.indication

Optional

- NMS_DISCOVER_NETWORK.request
- NMS_DISCOVER_NETWORK.indication
Link Configuration Service

- Determine number and status of links
- Configure links
- Parameters:
  - Link activity, as a bit field
  - Reference transmit rate
  - Configuration for each link
    - Link type and status/errors (read-only)
    - Transmit rate
    - Link state
Transmit Rate Abstraction

- Designed to be simple and flexible
  - And reflect current practice
- Control of **reference rate** for all links
- Control of **link rates** individually
- Each rate can be controlled either as a numeric rate, or as a divider

```
Reference Rate Source ➔ Reference Rate Control ➔ Link Rate Control

...```

---

SpaceWire-PnP
LCS Primitives

- LCS_READ_PORT_ACTIVITY.request
- LCS_READ_PORT_ACTIVITY.indication
- LCS_READ_REFERENCE_RATE.request
- LCS_READ_REFERENCE_RATE.indication
- LCS_WRITE_REFERENCE_RATE.request
- LCS_WRITE_REFERENCE_RATE.indication
- LCS_READ_LINK_CONTROL.request
- LCS_READ_LINK_CONTROL.indication
- LCS_WRITE_LINK_RATE.request
- LCS_WRITE_LINK_RATE.indication
- LCS_WRITE_LINK_PRIORITY.request
- LCS_WRITE_LINK_PRIORITY.indication
- LCS_WRITE_LINK_STATE.request
- LCS_WRITE_LINK_STATE.indication
Router Configuration Service

› Only for routers (obviously)
› Router configuration and status
› Parameters
   › Router configuration
      › Watchdog timeout (optional)
      › Arbitration mode
      › Time-code counter control
   › Routing table
      › Port association
      › Mechanism and arbitration control
      › Partial implementations permissible
RCS Primitives

› RCS_READ_WATCHDOG_TIMEOUT.request
› RCS_READ_WATCHDOG_TIMEOUT.indication
› RCS_WRITE_WATCHDOG_TIMEOUT.request
› RCS_WRITE_WATCHDOG_TIMEOUT.indication
› RCS_READ_ARBITRATION_MODE.request
› RCS_READ_ARBITRATION_MODE.request
› RCS_WRITE_ARBITRATION_MODE.request
› RCS_WRITE_ARBITRATION_MODE.request
› RCS_READ_TIME_COUTER.request
› RCS_READ_TIME_COUTER.indication
› RCS_RESET_TIME_COUTER.request
› RCS_RESET_TIME_COUTER.indication
› RCS_ENABLE_TIME_COUNTER.request
› RCS_ENABLE_TIME_COUNTER.indication
› RCS_READ_LA_COUNT.request
› RCS_READ_LA_COUNT.indication
› RCS_READ_ROUTING_TABLE_ENTRY.request
› RCS_READ_ROUTING_TABLE_ENTRY.indication
› RCS_WRITE_ROUTING_TABLE_ENTRY.request
› RCS_WRITE_ROUTING_TABLE_ENTRY.indication
Summary So Far

- Have presented
  - Principles of SpaceWire-PnP
  - Which bits are standardised
    - RMAP usage
    - RMAP address space (parameters)
    - Primitives
  - Functions logically grouped into services
    - Device Identification Service
    - Network Management Service
    - Link Configuration Service
    - Router Configuration Service
SpaceWire-PnP Extensibility

- SpaceWire-PnP is a convenient mechanism for detecting and configuring
- Can it be used as a “gateway” to more functionality?
- Devices can define their **capabilities**
  - Identifiable feature set
  - Supported by a SpaceWire-PnP service
    - Parameters
    - Primitives
    - Permits identification and configuration of the capability
Capabilities

- Device can provide a list of *capabilities*
- Capabilities based on protocol ID
  - A protocol which is supported
  - Optionally “transported” over another protocol
  - Supports nesting of “transports”
- Examples
  - CPTP over SpaceWire-(R)T
  - A standardised address space “transported” over RMAP
Describing RMAP Address Spaces

- SpaceWire-PnP document proposes a method for describing RMAP address spaces
- Capability services allow the description of:
  - Memory regions which exist to receive data: **data sinks** (e.g. actuators)
  - Memory regions which permit access to generated data: **data sources** (e.g. sensors)
- Also permits non-trivial access mechanisms
  - Delayed response reads and writes
  - Initiated reads and writes
Using SpaceWire-PnP (1): SOIS

- Supports services necessary for SOIS
- Device information, network ID and link activity together permit network and device discovery
- Minimal implementation requirements:
  - 12 words of read-only constant registers
  - 1 read-only dynamic register
  - 1 read-write register
- Minimal set of primitives
  - 5 pairs (request/indication)
Using SpW-PnP (2): Datasheets

- Can use data source capability service to describe an RMAP region to read a datasheet from
  - E.g. direct interface to a PROM
- Data source type identifies format of datasheet
  - E.g. xTEDS
- Minimal implementation (in addition to previous)
  - 8 read-only words
  - 2 primitive pairs
- Uses the same RMAP core as for SpaceWire-PnP
Using SpW-PnP (3): RMAP Spaces

› Can use data source/sink capability services to describe an existing RMAP address space
  › E.g. JAXA standardised memory map
› Same resource requirements as datasheet example for read-only
  › Add 8 read-only words and 4 primitive pairs for read-write
  › This adds a data sink
Using SpW-PnP (4): Notification

› Ability for routers (or any device) to automatically inform a network manager when status changes
  › E.g. link connect/disconnect
› Uses a simple data source
› Additional requirements (from datasheet case):
  › 1 read-write field for a target source
  › 12 read-write fields for an initiator source
› Features to support multiple, uncoordinated network managers are documented

- Capability services could easily be added to support the configuration of mechanisms such as SpaceWire-(R)T and SpaceWire-D
  - No changes to SpaceWire-PnP necessary
- Standard SpaceWire-PnP device configuration easily fits within time slots
  - Works well with SpaceWire-D
  - Could be transported over SpaceWire-(R)T
- Level 2 support needs documenting further
Using SpW-PnP (6): GenFAS

- The MARC hardware, built by SEA, has simplified SpW-10X compatible address spaces on each node and router
- SpaceWire-PnP defines 10X compatibility
- SciSys has implemented the full set of core SpaceWire-PnP primitives in the GenFAS software (executing on MARC)
  - Was a valuable learning experience
  - Fairly trivial (~2k LOC, heavily commented)
  - Works well!
Known Issues/ Discussion Points (1)

› Possibly confusing terminology: link and port used almost interchangeably
  › Haven’t got around to fixing this
› Couple of minor changes necessary for full SOIS support
  › Haven’t got around to updating document
› Deliberate mirroring of fields to support consolidated reads
  › Might not want this
Known Issues/ Discussion Points (2)

› Time-code handling is just one possible way
› Interrupts not in current document version
› Capabilities support full range of (extended) PIDs
  › Probably unnecessary: simplifications possible?
  › There may be a better way to identify capabilities than by protocol ID
  › However, the concept of capabilities is useful

› And more…
How to Use the SpW-PnP Document

› This is a **discussion document**

› It is:
  › A complete proposal
  › The product of experience and research
  › The result of inputs from many people

› It is **not:**
  › Expected to become a standard as it is!
A Guide to the Document

› The document is long, but don’t be scared
› There is a detailed introduction
› Level 2 support is documented
  › In a self-contained section
  › Can safely be ignored unless you are interested
› Compatibility with the 10X is documented
› Document is repetitive in structure
  › Each parameter, entry, field, primitive and parameter is documented in detail
Page Breakdown: Level 1 Services

Total: 65 Pages

- Primitives, 35
- Parameters, 27
- Discussion, 3
Summarising SpaceWire-PnP

- Protocol utilising RMAP
- UoD document available: SpaceWire-PnP v2.1
- Defines
  - Target parameters
  - Initiator primitives (service interface)
  - Behaviours (algorithms) where necessary
- Simple
- Does not require extra feature support
- Flexible and extensible
  - Can use capability services to extend support
Questions?
Discussion?