# SPA CAN Bus Subnet Adaptation*

Christoffer Holmstedt
christoffer.holmstedt@gmail.com

## ABSTRACT

Abstract. What is the state of the art? Maximum 250 words.

## 1. INTRODUCTION

*The following introduction is the same as the one for SPA Local Subnet Standard [2] with a slight modification in the last paragraph. SPA Local has been changed to SPA CAN Bus.*

SPA embraces and implements a collection of standards designed to facilitate rapid constitution of spacecraft systems using modular components.

The data portion of the standard is based on a data-centric model in which components self-describe their interfaces when they register with the system. This self-description is facilitated by an XML byte string called an extensible Transducer Electronic Data Sheet (xTEDS). The xTEDS defines the identity of the component, and the organization of its data interfaces.

SPA is a networked data exchange model. One of the premises of SPA is that there is no distinction between a hardware device that supports a data interface and a software application that does the same - thus all endpoints in a SPA network, physical or virtual, are referred to as "components".

Other components may use queries to express their data consumption needs. These queries are based on the specification of data "kind" with "qualifiers" that provide additional context to focus the search. When matches are found, the consumer may subscribe to messages of providers that meet their search criteria. As such, SPA systems dynamically bind their data at runtime.

*This report was written in October 2013 in the advanced level project course DVA425 at Mälardalen University, Sweden.

The SPA CAN Bus Subnet Adaptation specifies the means by which SPA components interoperate on a CAN Bus Subnet. It specifies the inter SPA-C component communication method used in order to establish a SPA CAN Bus Subnet and the messages, protocols, and interactions required in order to facilitate PnP functionality.

## 2. TERMS AND DEFINITIONS

Terms and definitions are copied from documents related to these topics. These are the SPA Standard Drafts [3, 4, 6, 5, 2] and the CAN Bus 2.0 Specification [1].

### 2.1 General

- **UUID:** Universally Unique Identifier
- **URN:** Uniform Resource Name
- **ASIM:** Appliqué Sensor Interface Module

### 2.2 Space Plug-and-Play Avionics

- **SM-x:** SPA Subnet Manager, where x represents a given technology protocol.
- **CUUID:** Component Universally Unique Identifier.
- **xTEDS:** Extensible Transducer Electronic Data Sheet.
- **XUUID:** xTEDS Universally Unique Identifier.
- **CAS:** The Central Addressing Service (CAS) is responsible for providing logical address blocks to be assigned to each hardware or software component. The CAS stores the logical address block and logical address for each SPA Manager in the SPA Network.
- **Plug-and-Play (PnP):** Ability to connect a device to the larger system and have the two work together with little or no set-up required (e.g., automated system recognition and data exchange).
- **SPA Lookup Service:** The SPA Lookup Service is responsible for accepting component registration and providing data source route information for components requesting a particular type of service (or returning a nack if the service is not available).
- **SPA Manager:** The SPA manager is responsible for performing discovery for a particular subnet. It maps incoming packets to the correct SPA endpoint on the subnet, encapsulating the SPA packet with the correct protocol header. In the reverse direction it removes

the protocol header and possibly adds a new header conforming to the subnet the packet is about to enter. It is also responsible for topology discovery and reporting within the subnet.

- **SPA-L / SPA-Local:** The SPA Local Subnet Adaptation specifies the means by which SPA components interoperate on a local processing node. It specifies the inter-process communication method used in order to establish a SPA local network and the messages, protocols, and interactions required in order to facilitate PnP functionality.
- **SM-c:** SPA CAN Bus Subnet Manager.
- **SPA-c:** The SPA CAN Bus Subnet Adaptation.

## 2.3 CAN Bus

- **CAN FD:** CAN with flexible data-rate. Supported from Linux kernel 3.6, can be tested through virtual CAN interfaces.

# 3. THE SPA CAN BUS SUBNET

The CAN Bus is specified in different ISO Standards and Bosch has shared their initial specification available for free on internet [1]. The main obstacle to solve to be able to set up a SPA-C Subnet is the difference between SPA and CAN Bus when it comes to addressing components.

In SPA each component must have an address this is the opposite of how a CAN Bus works. Within a CAN Bus all componenents reads the same message at the same time, depending on the message id it's dropped or read by specific components. For the continuation of this CAN Bus adaptation the assumption is made that it's a viable solution to implement a sort of address resolution protocol on top of the CAN Bus without loosing too much bandwidth/efficiency. The rationale behind this is presented in appendix A.

## 3.1 SPA-C Message Composition

text

### *3.1.1 SPA Message*

text

### *3.1.2 SPA-C Message*

text

### *3.1.3 SPA-C Message Header*

text

# 4. DISCOVERY

text

# 5. SPA PACKET ROUTING OVER SPA-L

text

## 5.1 Routing a SPA packet on the SPA-C subnet

text

### *5.1.1 Direct routing between SPA-C components*

text

# 6. DIFFERENTATION OF TRAFFIC

text

# 7. RECONFIGURATION

text

# 8. SPA CAN BUS MESSAGE FORMAT

text

# 9. REQUIREMENTS

text

# 10. REFERENCES


[1] Can bus 2.0 specification. `http://www.bosch-semiconductors.de/media/pdf_1/canliteratur/can2spec.pdf`. Accessed 2013-10-03.

[2] Spa local subnet adaptation standard draft. Not available for public review.

[3] Spa logical interface standard draft. `http://aiaa.kavi.com/public/pub_rev/SPA_S-133-3-201X_PR_Final.pdf`. Accessed 2013-10-03.

[4] Spa networking standard draft. `http://aiaa.kavi.com/public/pub_rev/SPA_S-133-2-201X_PR_Final.pdf`. Accessed 2013-10-03.

[5] Spa physical interface standard draft. `public/pub_rev/SPA_S-133-2-201X_PR_Final.pdf`. Accessed 2013-10-03.

[6] Spa spacewire subnet adaptation standard draft. `http://aiaa.kavi.com/public/pub_rev/SPA_S-133-9-201X_PR.pdf`. Accessed 2013-10-03.


# APPENDIX

# A. BOOTSTRAPING

## A.1 Problem Statement

When starting a new system the first thing that occurs is the "Network Topology Discovery" [4]. This process assigns a logical address to each component in the network. This contradicts the CAN Bus specification where no component is identified by a specific address.

The assumption is made that it's more important to implement SPA over CAN Bus than it is to follow the intentions of the CAN Bus specification. The first thing that needs to be defined is how should different components be addressed in a local CAN Bus subnet?

## A.2 Rationale

According to SPA each subnet should have its own subnet manager, for the CAN Bus subnet that component is called SM-c (SPA Manager CAN Bus). When the SM-c has been assigned a logical address block from the Central Addressing Service (CAS) component the SM-c must assign a logical address to each component on its subnet. It's the SM-c responsibility to keep a routing table between SPA logical addresses and CAN Bus subnet message IDs.

A straightforward solution is to assign one CAN Bus Message ID for each component that it should listen on. The

problem with this is that if more than one component tries to contact another component, at the same time, both will start their respective message with the same CAN Message ID. This will cause a bit error on one of the components that may cause it switch to ”bus off” state. To reactivate the components that have gone into ”bus off” state, special messages must be sent over the CAN Bus. To prevent the message id collision, each pair of components should be assigned a unique pair of CAN Bus Message IDs that they can communicate over.

With the mapping of SPA logical addresses to local CAN Bus subnet addresses out of the way, it's time to look at the actual assignment. During ”Network Topology Discovery” within a SPA-Local interconnect (within one processing unit) all components can contact the SM-l component through a Well-known port of 3500 [2]. Applying a similiar approach on a CAN Bus where all connected components contact the SM-c with a well-known CAN Bus Message ID would cause similiar problems as described earlier, message ID collisions.

For a SpaceWire subnet the process is turned around, it's the ”SPA Manager SpaceWire” (SM-s) that is responsible to initiate contact. It first try to identify all others SM-s components on its subnet and then other SPA Components. An example on how this is done is given in the SPA SpaceWire Adaptation Standard Annex A [6]. The difference with CAN Bus is the already available addresses. The SM-s sends a specific message to all possible addresses in the network, which is not possible on a CAN Bus. A SM-c could instead send a specific message with a well-known message id that all other SM-c:s on the network could listen to. The new problem that arise concerns the response, which message ID should the response have?

If a well-known message ID is defined for the response, message ID collision could occur if multiple SM-c:s exist on the network. This is not a viable solution. To make SPA work over CAN Bus minor configuration must be made before adding new components and during initial assembly of a CAN Bus subnet.

Earlier in appendix A it says that each pair of components must be assigned a unique pair of CAN Bus Message IDs. With only 10 components in a CAN Subnet that would mean 90 message IDs have to be defined manually. This clearly shows that defining all Message IDs during the configuration phase is not a desired solution. To simplify this process the SM-c is given the responsiblitiy to map all message IDs between different SPA components. The assignment that is done manually is the message IDs used between the SM-c and each component on the network. With 10 components on the network only 18 message IDs would have to be assigned instead of earlier mentioned 90 (with increase amount of components the difference will be even higher).

To recap one possible solution for Network Topology Discovery within a CAN Bus is as follows:

1. Configure each component with its CUUID
2. Configure each component with its xTEDS file and XUUID
3. Assign a pair of CAN Bus Message IDs for each component. One to listen on and one to transmit on.
4. Configure the responsible SPA Manager CAN Bus to listen and send on the earlier defined CAN Bus Message IDs for each component.