10. AVTP Control Streams

10.1. Overview

The AVTP control stream PDU is designed to allow control messages to be streamed on an AVB network. Each AVTP control stream PDU may contain one or more control messages. AVTP control stream PDUs may include a timestamp that is designed to be used by devices to timestamp when the data this is included in the control message was sampled.

Editors Note: Do we need to add a use case discussion here that talks about tunneling CAN and flexray messages vs. the idea of a gateway that regenerates messages

Editors Note: Should we talk about topology with packet paths being CAN/FlexRay->Ethernet->CAN/FlexRay, CAN/FlexRay->Ethernet, Ethernet->CAN/FlexRay, Ethernet->Ethernet.

Editors Note: We may add a note here that clarifies that this format is only allowed for subtype x. This same text could be used in other section such as Simple Audio and Simple Video etc.

10.2. AVTP control stream data AVTPDU format

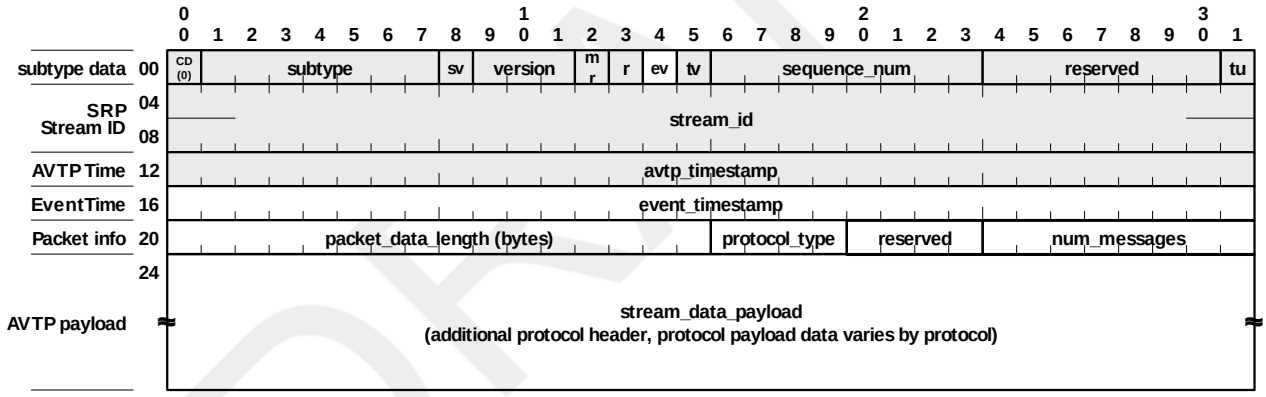

Figure 10.1. control stream PDU format

The control stream PDU consists of the fields contained in the AVTPDU common data header and the following fields:

—**ev** (event_timestamp valid): 1 bit
—**event_timestamp**: 32 bits
—**packet_data_length**: 16 bits
—**protocol_type**: 4 bits
—**reserved**: 4 bits
—**num_messages**: 8 bits

10.2.1. ev

The **ev** field is used to indicate whether the 32 bit **event_timestamp** field contains valid data.

The bit is set to one (1) if the **event_timestamp** field contains a valid timestamp.

The bit is set to zero (0) if the **event_timestamp** field is not valid.

10.2.2. event_timestamp

The **event_timestamp** field contains the acquisition time, if available, of the data contained in the first control message contained in the PDU. The **event_timestamp** field shall be in units of nanoseconds.

The **event_timestamp** field is formatted the same as the **avtp_timestamp** field as described in 5.4.8.

Editors Note: This needs a note or more description that identfies this as gPTP time, but does not contradict 5.4.8

10.2.3. packet_data_length

The **packet_data_length** field contains the length in octets of the **control_stream_data_payload**.

10.2.4. protocol_type

The **protocol_type** field contains the protocol of the messages contained in the PDU. The **protocol_type** field shall contain one of the protocol type values defined in Table 10.1, “protocol type” .

Table 10.1. protocol type

Value	Description
0	FlexRay
1	CAN Base
2	CAN Extended
3	LIN
4	TSCS
5-14	Reserved
15	Proprietary

10.2.5. num_messages

The **num_messages** field contains the number of messages of type **protocol_type** contained in the PDU. All messages contained in single AVTPDU shall be of the same protocol type.

10.3. Automotive Networks

Editors Note: Add a description of the use cases and how data flows across a backbone vs. a gateway. Also discuss that gateway and tunnel ends are not discovered in 1722a. Packet address detemination is done at some other layer not here.

10.3.1. FlexRay

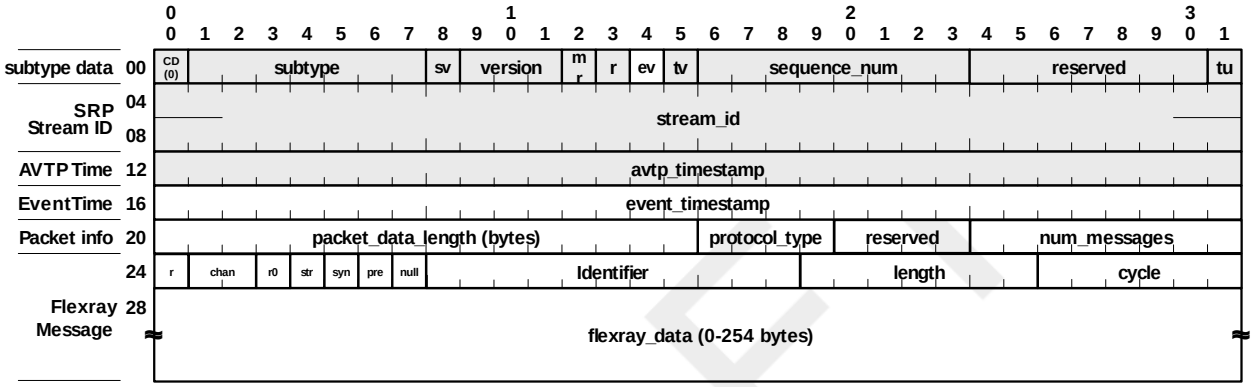

Figure 10.2. flexray PDU format

The flexray PDU consists of a control stream PDU and one or more flexray messages that contain the following fields:

—**r** (reserved): 1 bit
—**chan** (source channel): 2 bits
—**r0** (reserved 0): 1 bit
—**str** (startup): 1 bit
—**syn** (sync): 1 bit
—**pre** (payload preamble): 1 bit
—**null** (null): 1 bit
—**identifier**: 11 bits
—**length**: 8 bits
—**cycle**: 6 bits
—**flexray_data**: 0-254 bytes

10.3.1.1. r

The **r** bit is reserved.

10.3.1.2. chan

The **chan** bits indicate the channel that this flexray message was received on if the original source of the message was flexray. It is possible that flexray message was received on both the A and B channel. The **chan** bits are set according to Table 10.2, “Source Channel” .

Table 10.2. Source Channel

Value	Meaning
0	The original source of the message is not a FlexRay bus or is unknown.
1	The message was received on FlexRay channel A.
2	The message was received on FlexRay channel B.
3	The message was received on both FlexRay channel A and B.

NOTE—While it is possible that the message may be received on box FlexRay channels, since there is only one cycle field messages received on both channels but different cycles can not be represented

10.3.1.3. r0

The **r0** bit contains the value of the reserved 0 field in flexray message.

10.3.1.4. str

The **str** bit contains the flexray message startup frame indicator.

10.3.1.5. syn

The **syn** bit contains the flexray message sync frame indicator.

10.3.1.6. pre

The **pre** bit contains the flexray message payload preamble indicator.

10.3.1.7. null

The **null** bit contains the flexray message null frame indicator.

10.3.1.8. identifier

The **identifier** field contains the flexray message frame identifier if the message was sourced from a FlexRay bus. The **identifier** field has a valid range of 1 to 2047 if the message was sourced from a FlexRay bus. If the source of the message was not a FlexRay bus then the **identifier** field is set to zero (0);

10.3.1.9. length

The **length** field contains the number of bytes of **flexray_data** contained in this message (0-254).

10.3.1.10. cycle

The **cycle** field contains the flexray message cycle count if the message was sourced from a FlexRay bus. If the source of the message was not a FlexRay bus then the **cycle** field is set to zero (0).

10.3.1.11. flexray_data

The **flexray_data** field is a variable length field that contains from 0 to 254 bytes of message data.

10.3.2. CAN Base

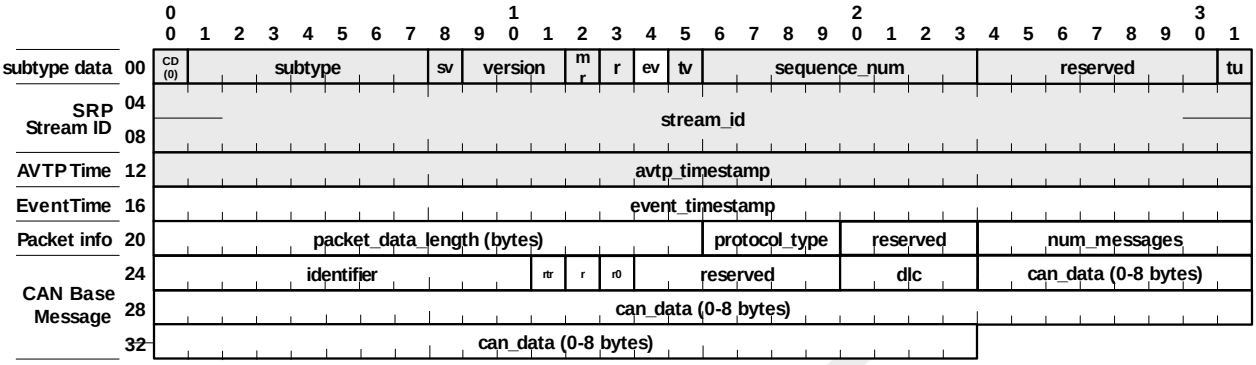

Figure 10.3. CAN PDU format

The CAN base PDU consists of a control stream PDU and one or more CAN base messages that contain the following fields:

—**identifier**: 11 bits
—**rtr**: 1 bit
—**r**: 1 bit
—**r0**: 1 bit
—**dlc**: 4 bits
—**can_data**: 0-8 bytes

10.3.2.1. identifier

The 11 bit **identifier** field contains the CAN message identifier.

> **NOTE—**The CAN message identifier must be unique per bus. For example, A-Can and B-Can could use ID 100 but 100 can be used just a single time in each CAN.

10.3.2.2. rtr

The **rtr** bit contains the CAN message remote transmission request bit.

10.3.2.3. r

The **r** bit is reserved.

10.3.2.4. r0

The **r0** bit contains the CAN message reserved bit 0 (r0).

10.3.2.5. dlc

The 4-bit **dlc** field contains the number of bytes of **can_data** contained in this CAN message (0-8)

10.3.2.6. can_data

The **can_data** field is a variable length field that contains from 0 to 8 bytes of CAN message data. The length of the **can_data** field is determined by the **dlc** field.

10.3.3. CAN Extended

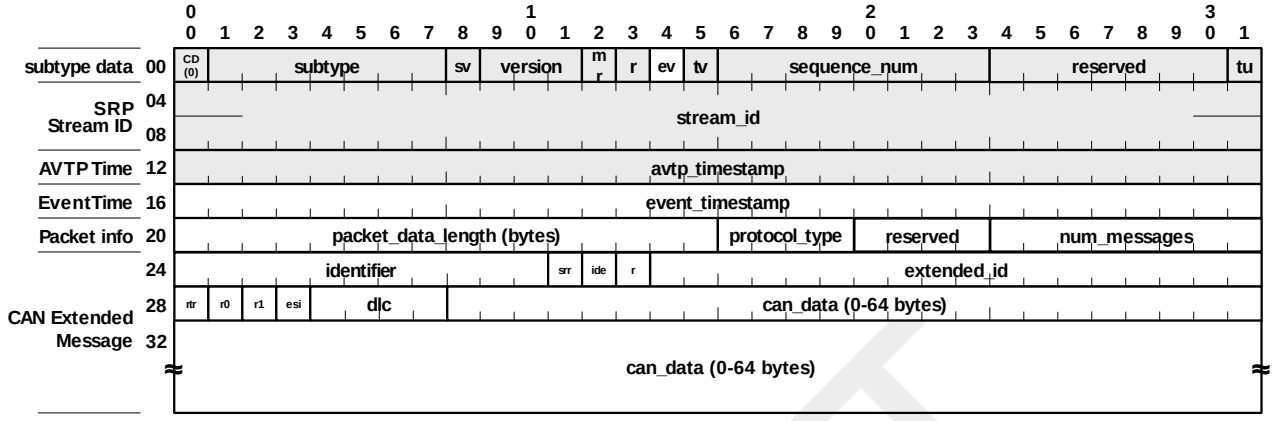

Figure 10.4. CAN PDU format

The CAN extended PDU consistes of a control stream PDU and one or more CAN extended messages that contain the following fields:

—**identifier**: 11 bits
—**srr**: 1 bit
—**ide**: 1 bit
—**r**: 1 bit
—**extended_id**: 18 bits
—**rtr**: 1 bit
—**r0**: 1 bit
—**r1**: 1 bit
—**esi**: 1 bit
—**dlc**: 4 bits
—**can_data**: 0-64 bytes

10.3.3.1. identifier

The 11 bit **identifier** field contains the first part of the CAN message identifier.

10.3.3.2. srr

The **srr** field contains the CAN message substitute remote request bit.

10.3.3.3. ide

The **ide** field contains the CAN message identifier extension bit.

10.3.3.4. r

The **r** bit is reserved.

10.3.3.5. extended_id

The 18-bit **extended_id** field contains the second part of the CAN message identifier.

10.3.3.6. rtr

The **rtr** bit contains the CAN message remote transmission request bit.

10.3.3.7. r0

The **r0** bit contains the CAN message reserved bit 0 (r0).

10.3.3.8. r1

The **r1** bit contains the CAN message reserved bit 1 (r1).

10.3.3.9. esi

10.3.3.10. dlc

The 4-bit **dlc** field indicates the number of bytes of **can_data** contained in the CAN message (0-64 bytes). The **dlc** field is set to indicate the data length as defined in Table 10.3, “Data Length Code” .

Table 10.3. Data Length Code

Data Length Code	Number of Data Bytes
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	12
10	16
11	20
12	24
13	32
14	48

Data Length Code	Number of Data Bytes
15	64

10.3.3.11. can_data

The **can_data** field is a variable length field that contains from 0 to 64 bytes of message data. The length of the **can_data** field is determined by the **dlc** field.

10.3.4. LIN

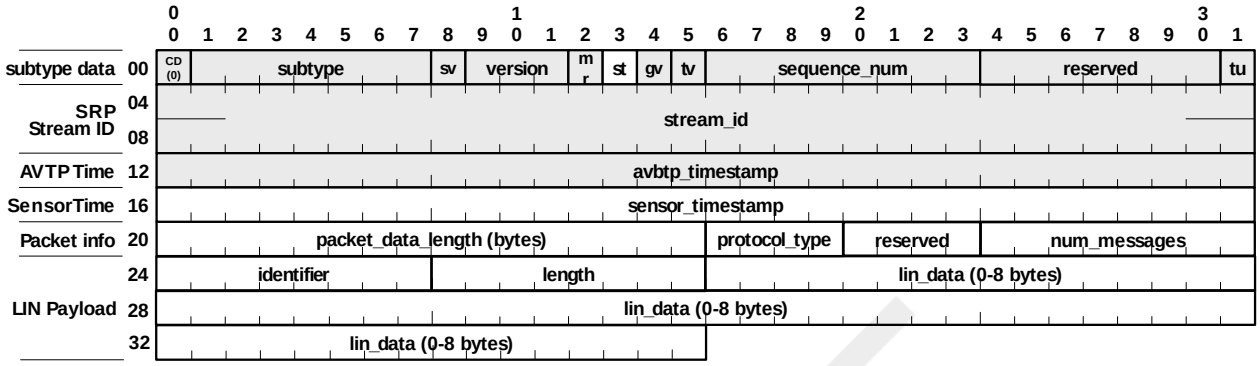

Figure 10.5. LIN PDU format

The LIN PDU consists of a control stream PDU and one or more LIN messages that contain the following fields:

—**identifier**: 8 bits
—**length**: 8 bits
—**lin_data**: 0-8 bytes

10.3.4.1. identifier

The 8 bit **identifier** contains the LIN message identifier.

10.3.4.2. length

The **length** field contains the number of bytes of **lin_data** contained in this message (0-8).

10.3.4.3. lin_data

The **lin_data** field is a variable length field that contains from 0 to 8 bytes of message data.

10.4. TSCS

Time Sensitive Control Streams (TSCS) use the header format as specified by Figure 10.6, “TSCS PDU format” .

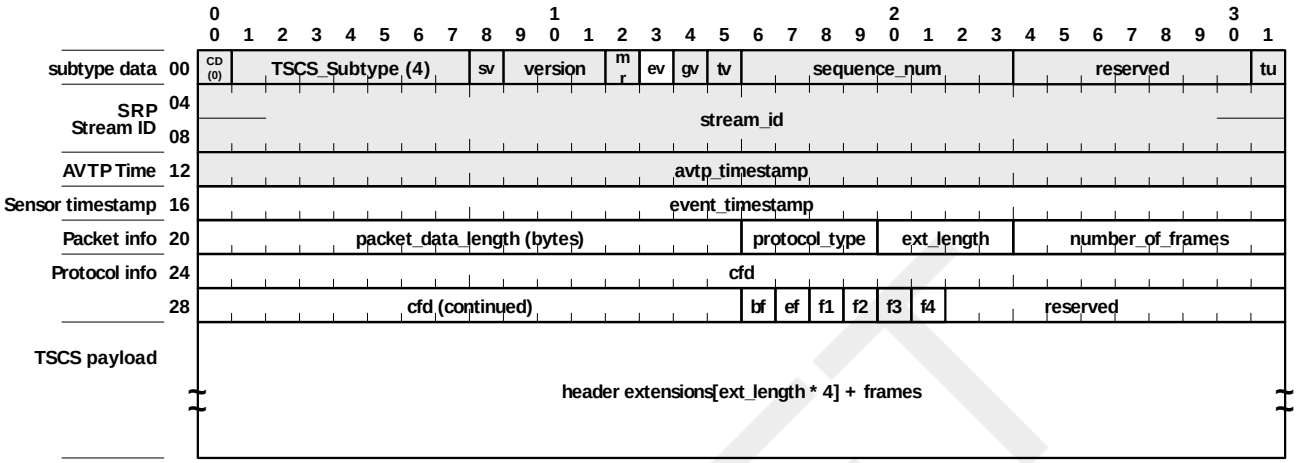

Figure 10.6. TSCS PDU format

—**cfd** (control format descriptor): 48 bits
—**bf** (begin frame flag): 1 bit
—**ef** (end frame flag): 1 bit
—**f1** (flag 1): 1 bit
—**f2** (flag 2): 1 bit
—**f3** (flag 3): 1 bit
—**f4** (flag 4): 1 bit
—**tscs_payload**: 0-1468 octets

10.4.1. The CFD field

The **cfd** field is a 48 bit field containing an EUI-48 value.

If the three most significant octets of the **cfd** field is the IEEE Std. 1722 assigned OUI, `90-e0-f0` or `91-e0-f0`, then the **cfd** field specifies a standard protocol, otherwise the **cfd** field specifies a vendor specific control protocol and the **cfd** field shall be prefixed by the vendor's assigned OUI24 or OUI36.

The least significant bit of the first octet of the **cfd** field is normally reserved in MAC-48 address for the unicast/ multicast bit. However, in the **cfd** field it is used as the **packetized** field, regardless if the CFD is a standard or vendor specific CFD.

10.4.1.1. Packetized CFD field

The **packetized** field in the **cfd** field represents the transport style of the time sensitive control data. The **packetized** field may be:

—zero (0): control stream
—one (1): control packets

10.4.1.1.1. CFD for control stream

When the **packetized** bit of the **cfd** field is zero (0), the **tscs_payload** is an undelimited stream of data, analogous to a unidirectional TCP/IP stream without acknowledgement or automatic resends or a unidirectional serial data port. The **bf** and **ef** bits are to be unused and set to 0.

10.4.1.1.2. CFD for control packets

When the **packetized** bit is one (1), the **tscs_payload** is a delimited stream of data, analogous to a UDP packet transport mechanism. The **bf** and **ef** bits are used for delimiting packets. The delimited packets may span multiple TSCS packets and may be larger than one MTU.

10.4.1.2. Standard CFD's

When the **cfd** field is in the form `90-e0-f0-XX-YY-ZZ` or `91-e0-f0-XX-YY-ZZ`, then the value of `XX` can be one of the values definned in Table 10.4, “Standard Control Formats” .

Table 10.4. Standard Control Formats

XX code	Description
`00` $_{16}$	`YY` represents IEEE Std. 1722-2011 **subtype** field. `ZZ` represents protocol revision.
`01` $_{16}$	`YY-ZZ` doublet represents protocol defined by IANA 'DCCP Well Known Ports' or 'DCCP Registered Ports.'
`02 - FF` $_{16}$	Reserved

10.4.2. bf field

When the **bf** is set to one (1) the first octet in this payload represents the first octet of the control packet.

The **bf** field is only used when the **cfd** field represents a **packetized** cfd form.

10.4.3. ef field

When the **ef** is set to one (1) the last octet in this payload represents the last octet of the control packet.

The **ef** field is only used when the **cfd** field represents a **packetized** cfd form.

10.4.4. f1,f2,f3,f4 fields

The **f1**, **f2**, **f3**, and **f4** fields are bit flags that may be used by vendor specific CFD's. These flags are unused for standard CFD's.

10.4.5. tscs_payload

The **tscs_payload** field's length is specified by the **packet_data_length** field minus 8 bytes. The entire frame is limited to the network's MTU.