IEEE P1904.3 Radio over Ethernet short introduction

Jouni Korhonen

IEEE 802.1TSN @Macau 2016

3/xx/2016

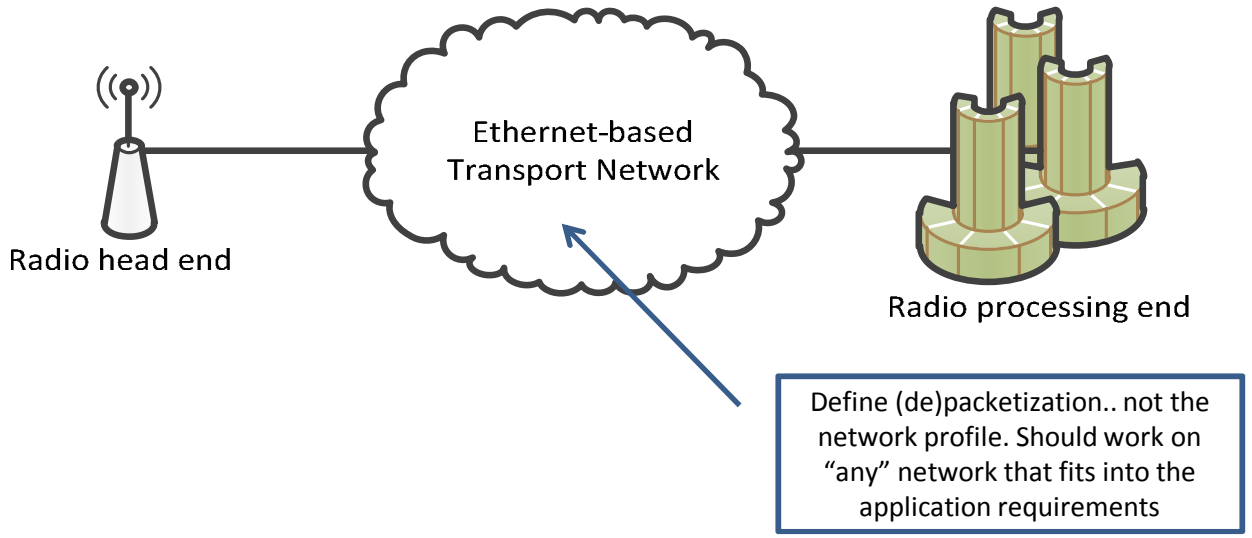
Disclaimer

- IEEE P1904.3 specification is currently a draft. Details _may_ change!

- This presentation does not represent the IEEE P1904.3 TF official views.

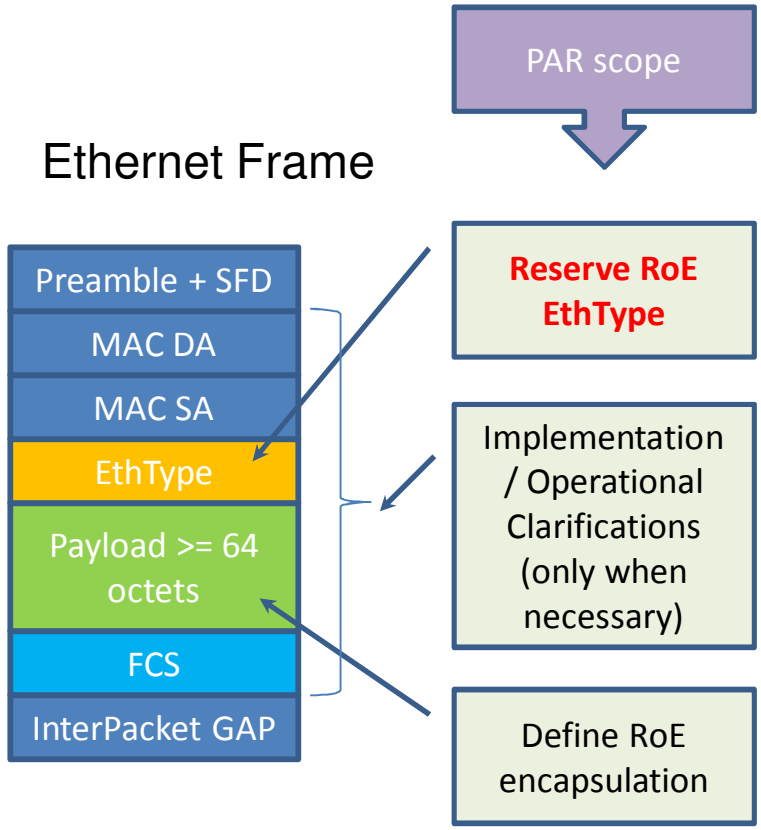
Background

- IEEE 1904.3 Radio over Ethernet (RoE) is an open & free standards effort to specify a transport protocol and an encapsulation format for transporting time-sensitive wireless (cellular) radio related application streams between two endpoints over Ethernet-based transport networks.



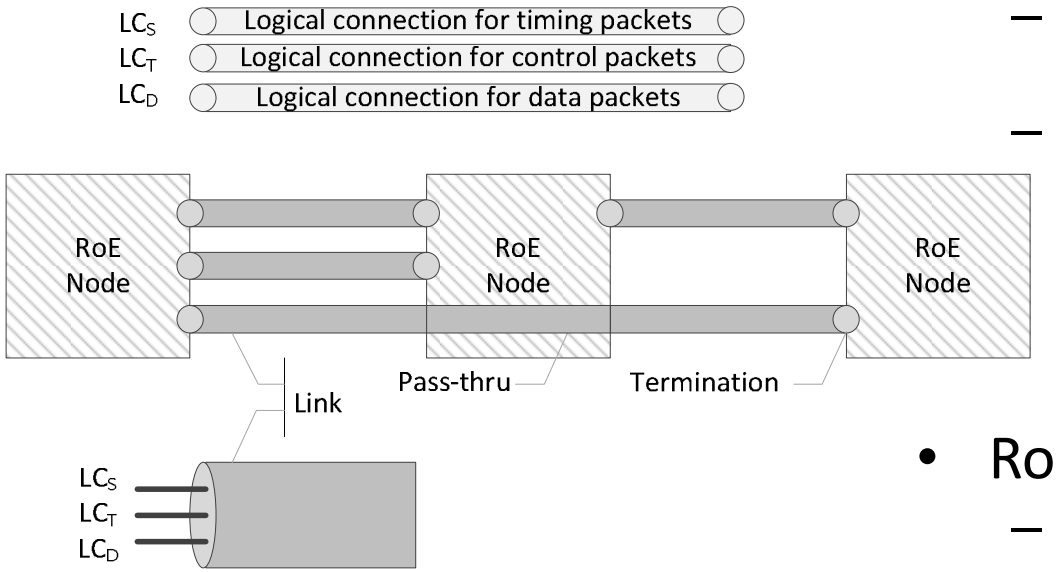
RoE standard scope

- RoE will define a native encapsulation header format for transporting time-sensitive “radio data” and “control data” streams.
- Definition of protocol primitives that allow:
  - Multiplexing of independent streams e.g. antenna carriers.
  - Timestamping or sequence numbering to enable time-synchronization of RoE packets and time alignment of streams.
- Control protocol for auxiliary non-data streams and for link & RoE endpoint management.
- Mapper(s) for existing CPRI framing standards to a native RoE encapsulation and transport.
- Encapsulation and transport of opaque data streams.



Ethernet packet remains unchanged.

RoE nodes and interfaces

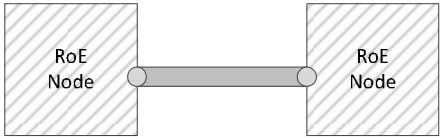


- RoE nodes:
  - Connection terminating node - “endpoint”.
  - Pass-through – RoE aware node that does not terminate a connection.

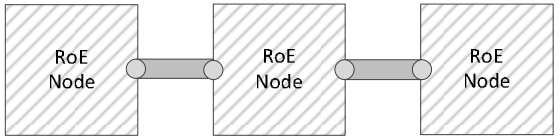
- RoE Interface for:
  - Timing packets over the LC_S logical connection.
  - Control packets over the LC_T logical connection.
  - Data packets over the LC_D logical connection.

Supported topologies

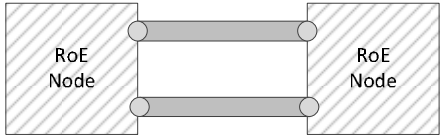
Single point to point:



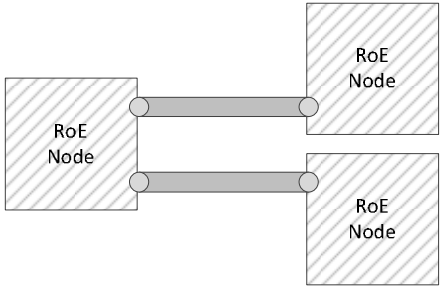
Chain:



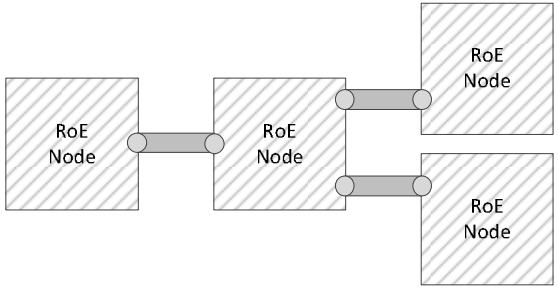
Multiple point to point:



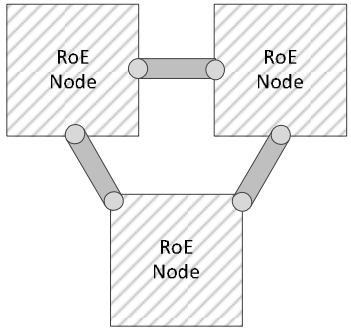
Tree:



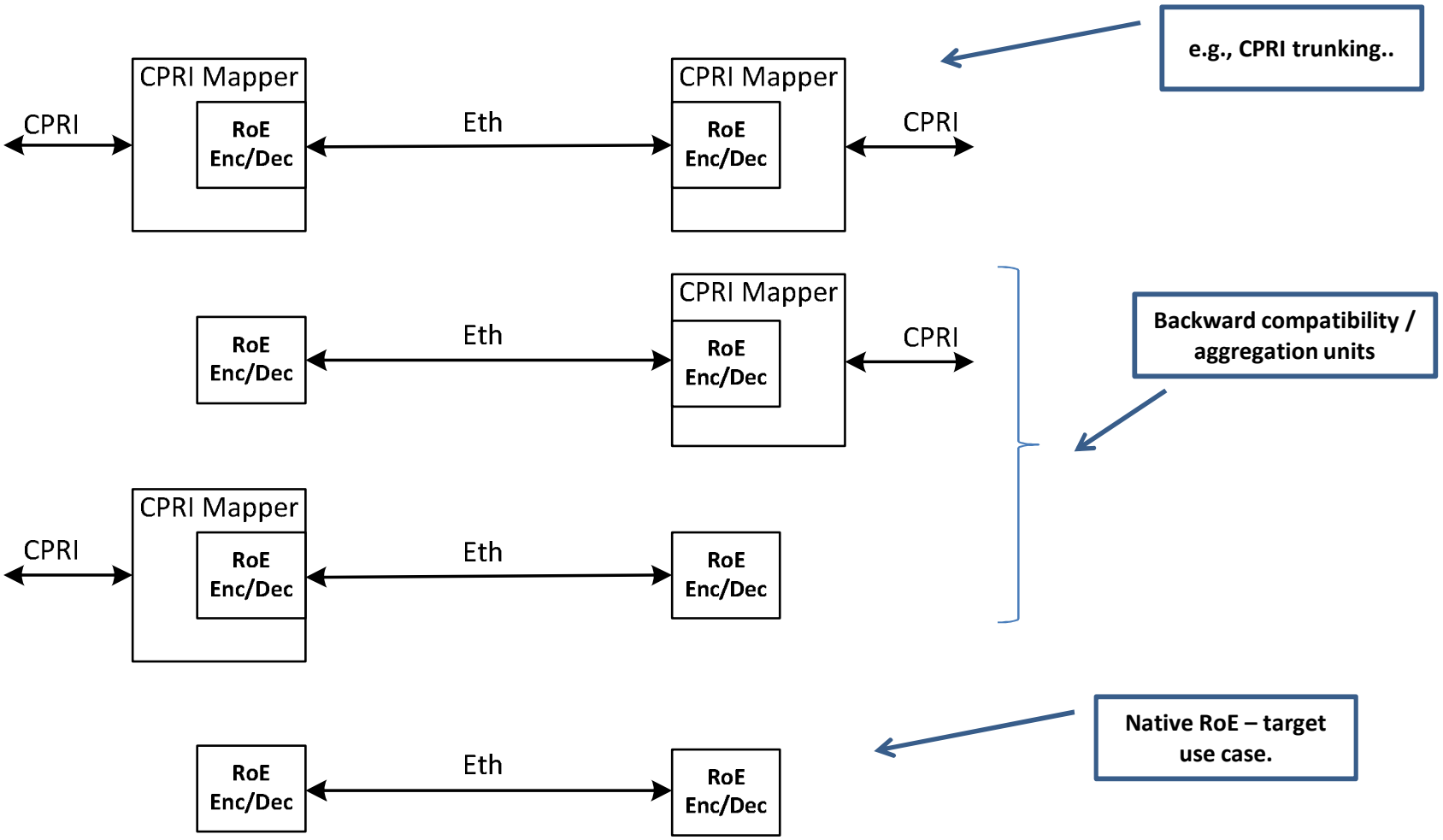
Point to multi-point:



Ring:

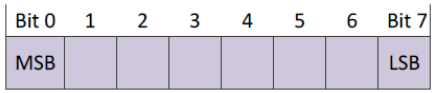


Supported use cases

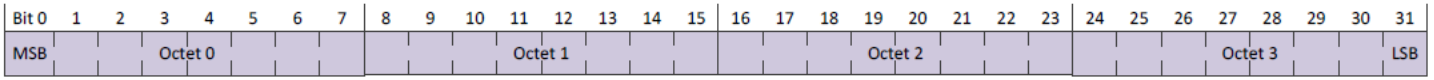


Byte and bit ordering

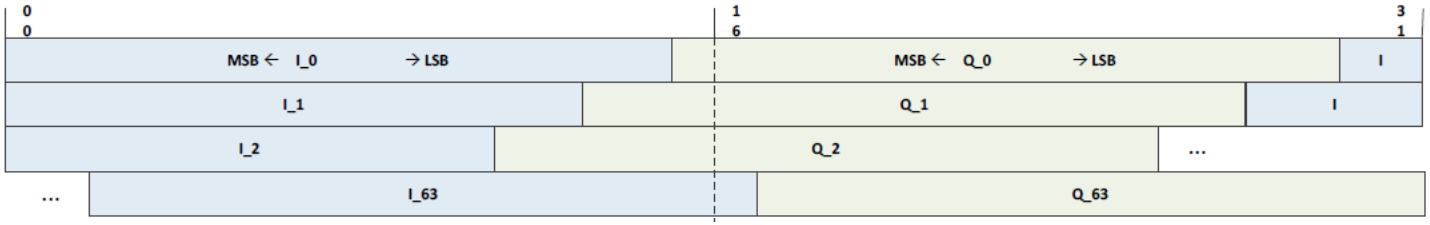
- Bits in an octet count from MSB (0) to LSB (7):



- Network (big endian) byte order in use:

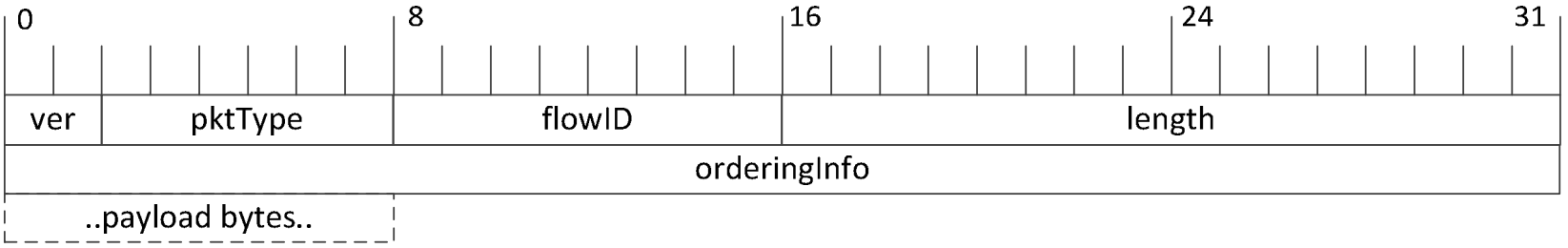


- In a payload samples are not interleaved (e.g., in a case of structure aware mappers):



RoE common header format

- The common header is shared by native RoE data, control and CPRI “mapper” packets.
- The header is placed in the transport protocol payload e.g., in a case of Ethernet into the Ethernet frame’s payload field.
- **orderingInfo** carries either a sequence number or a timestamp. *Decided per **flowID**/**pktType***. One flow cannot alternate between the two.
- **flowID**s are unidirectional and used for multiplexing multiple flows between two “RoE entities”.
  - Control packets can overload **flowID** for other purposes.
- **length** (in octets) covers everything starting after the 8 octets common header (including possible “extended header” or other future extensions).
  - Decouple from transport and allow concatenation of multiple packets.



RoE flows / connections

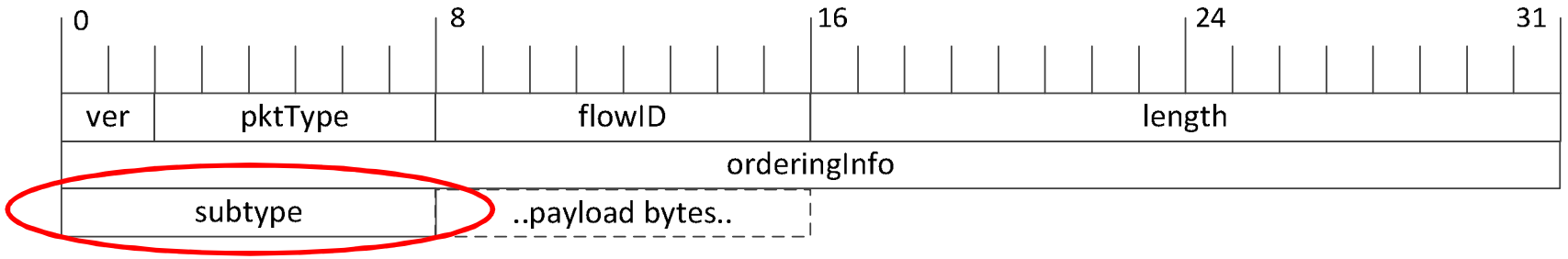
- A “RoE entity” that receives data RoE packets manages the **flowID** numbers space i.e., the RoE entity however it is addressed and identified cannot receive overlapping **flowID**s.

- A RoE node may have multiple “RoE entities” using different addressing and/or identity.

- A “RoE entity” consists of e.g., Ethernet ports, mappers and CPRI ports (more about these later).

RoE control packets

- Uses reserved **pktType**=0. Specific control packet types identified by the **subtype** (256 of those).
- The **flowID** is also set to reserved value 255 _unless_ otherwise specified by the control packet **subtype**.'
  - CPRI mappers use it to indicate the used "mapping rule".
- Control packets are used:
  - by CPRI mappers to carry e.g., control words.
  - by RoE for connection setup/management (in plans).



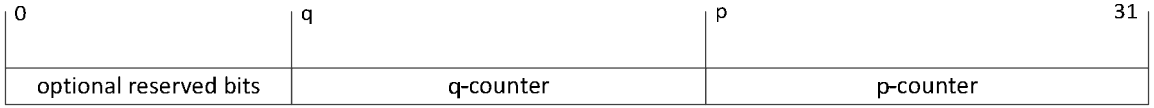
pktType allocation

- New **pktType**s added per need basis and required a standard effort.
- There *will* be few reserved **pktType**s for “experimental use”.
- A pktType is always associated with a complete packet handling and payload content description.

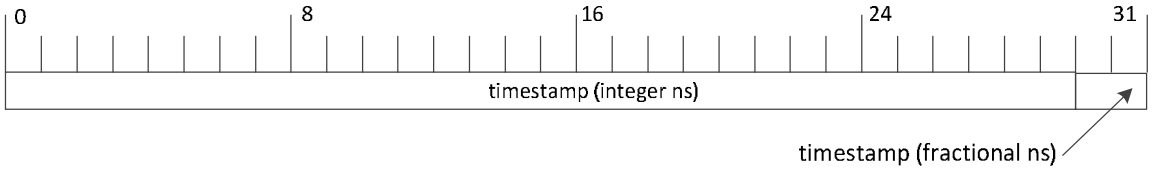
Binary value	Function	Description
00 0000b	Control Packet	Control packet between two RoE endpoints
00 0001b	Simple tunneling CPRI data packet	Data payload packet
00 0010b	Structure agnostic CPRI data packet	Data payload packet with RoE common frame header and structure agnostic CPRI payload.
00 0011b	Structure aware CPRI data packet	Data payload packet with RoE common frame header and structure aware CPRI payload.
00 0100b	Native RoE data flow packet	Data payload packet with RoE common frame header.
00 0101b	Slow C&M CPRI packet	C&M payload packet with 6 octet RoE frame header and structure aware CPRI Slow C&M payload.
00 0110b – 11 1111b	--	Reserved for future packet types

The orderingInfo field

- Two uses:
  - Sequence number has three “independent fields” with well defined semantics.



  - Timestamp, which is actually a _presentation_time_.



- Sequence number shall apply the following rule:
  $0 <= p < 32$ and $0 <= q <= p$ and $(32-p)+(p-q) <= 32$
- p-counter is mandatory, q-counter and rsvd bits may be omitted.
- Timestamp has 30 bits of 1ns resolution and 2 bits for 0.25ns resolution. Allows for ~1s window.

About sequence number

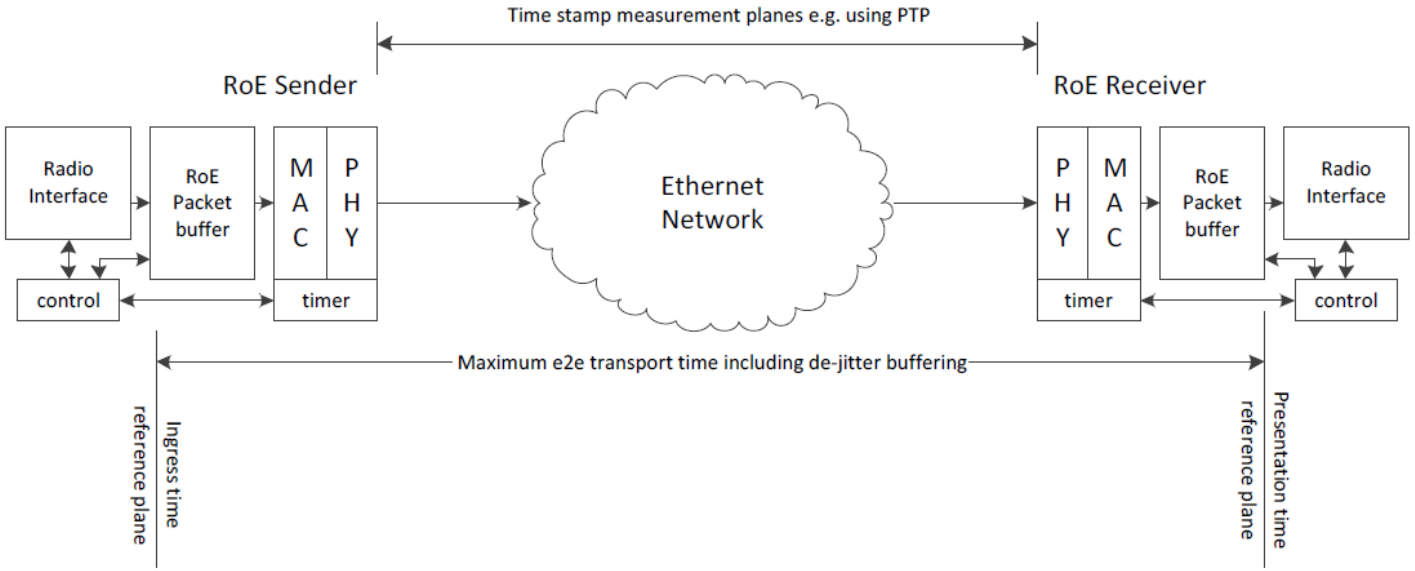
Variable	Bits	Default value	Description
seqNumPMax	32	0xFFFF FFFF	Maximum numerical value of the p-counter field so that seqNumPMax < 1 << 32-p.
seqNumPVal	32	0x0000 0000	The counter value of the p-counter field. When the seqNumPVal > seqNumPMax then the counter field wraps to a value seqNumPVal modulo (seqNumPMax+1). The counter wrap also causes the seqNumQVal to be incremented by seqNumQInc based on the seqNumQIncProp setting.
seqNumPInc	32	Undefined	The increment value that is dependent on the seqNumPIncProp setting.
seqNumPIncProp	3	0x001	• 0x0 no increment. • 0x1 increment by a constant on every sent packet. • 0x2 Increment by a packet payload size on every sent packet. • 0x3 to 0x7 reserved.
seqNumQMax	32	0x0000 0000	Maximum numerical value of the q-counter field so that seqNumQMax < 1 << p-q.
seqNumQVal	32	0x0000 0000	The counter value of the q-counter field. When the seqNumQVal > seqNumQMax then the counter field wraps to a value seqNumQVal modulo (seqNumQMax+1).
seqNumQInc	32	Undefined	The increment value that is dependent on the seqNumQIncProp setting.
seqNumQIncProp	3	0x001	• 0x0 no increment. • 0x1 increment by a constant on seqNumPVal wrap. • 0x2 to 0x7 reserved.
seqNumRsvd	32	0x0000 0000	Static value for reserved field, if available. Shall remain unchanged for the lifetime of the flow.

Mandatory set..

Static for the lifetime of the flow.. can be used to carry “data”

About timestamp

- RoE uses Midnight, 1 January 1970 as its epoch. It is assumed (but not mandated) that both RoE endpoints have an access to a reference time source.
- The time source shall be synchronized to international atomic time (TAI).
- A RoE sender calculates the presentation time at the RoE receiver.
  - Has to take the entire end to end delay between the RoE sender and receiver reference planes into account - not just the network delay.
- Current specification does not specify any specific use case with timestamps.
  - Possible future pktType carrying both sequence number and timestamp needed for CPRI truncking use cases (where each CPRI connection may be in their own time domain).



Mappers

- Currently three mappers (to be) specified:
  - Structure aware CPRI mapper.
  - Structure agnostic (CPRI) mapper.
  - Tunneling mapper.

Mappers – tunneling mapper

- Carries a “bit stream” including e.g., K-characters and preserves line coding like 8B/10B.
- Details are still tbd.

Mappers – structure agnostic (CPRI)

- Understands CPRI framing and radio frame boundaries.
- A RoE packet carries N * CPRI Basic Frames.
- Removes 8B/10B or 64B/66B line coding.
- There is not further splitting or looking into CPRI frames that is needed for synchronization and packetizing CPRI Basic Frames into RoE.

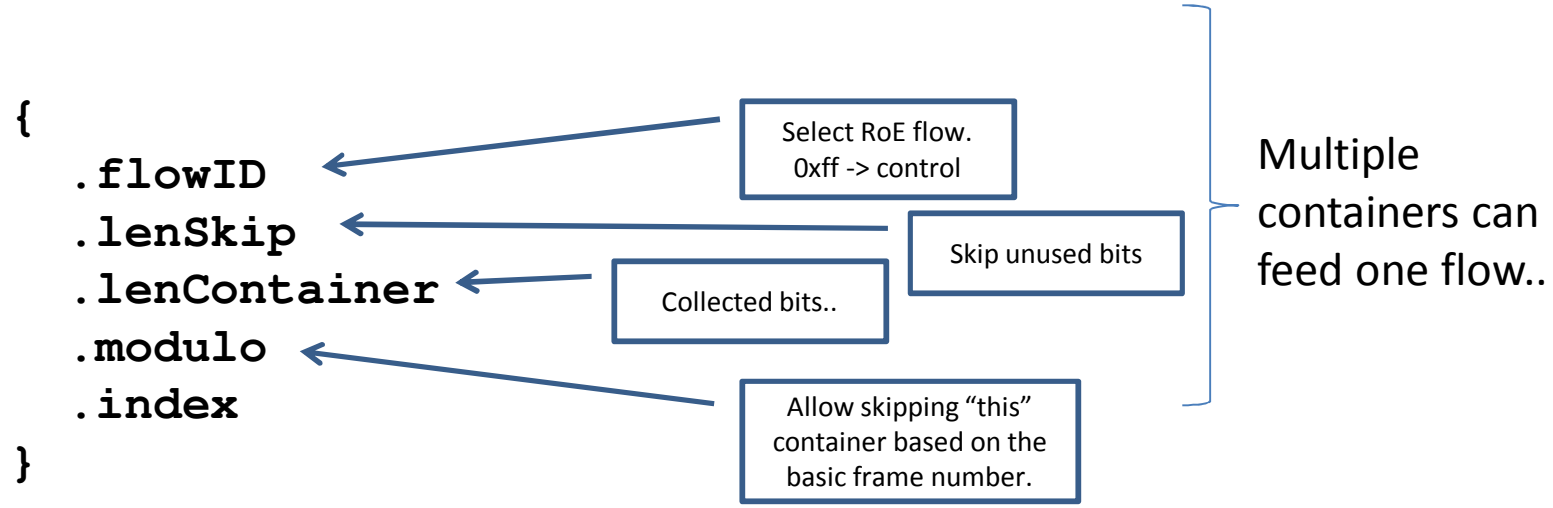
Mappers – CPRI structure aware

- Two sets of mapping rules:
  - Data part rules (IQ samples).
  - Control word rules.

- Basically supports CPRI Mapping Method #3 without stuffing samples.

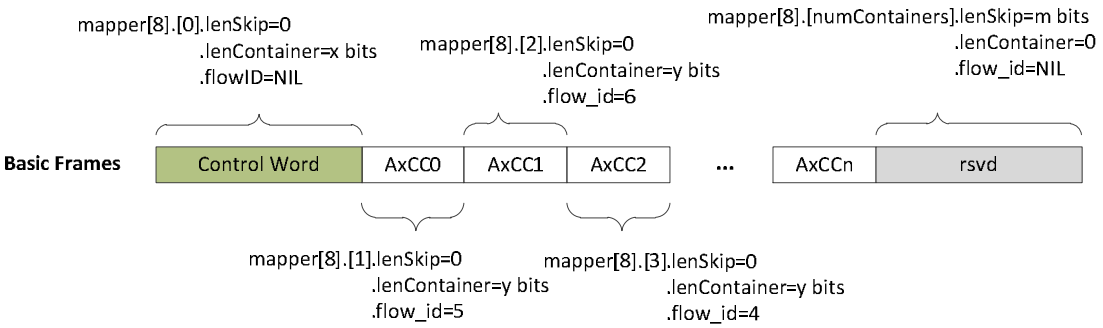
Mappers – rules for (de)packing data

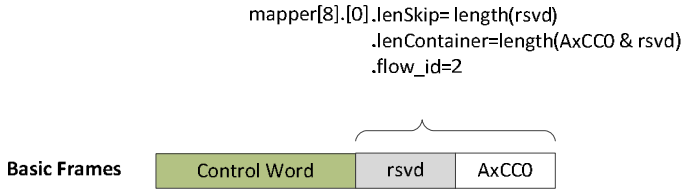
- A rule always describes a full CPRI Basic Frame. A single **(de)mapper[(de)mapperID].[containerID]** parameter set is described as below:



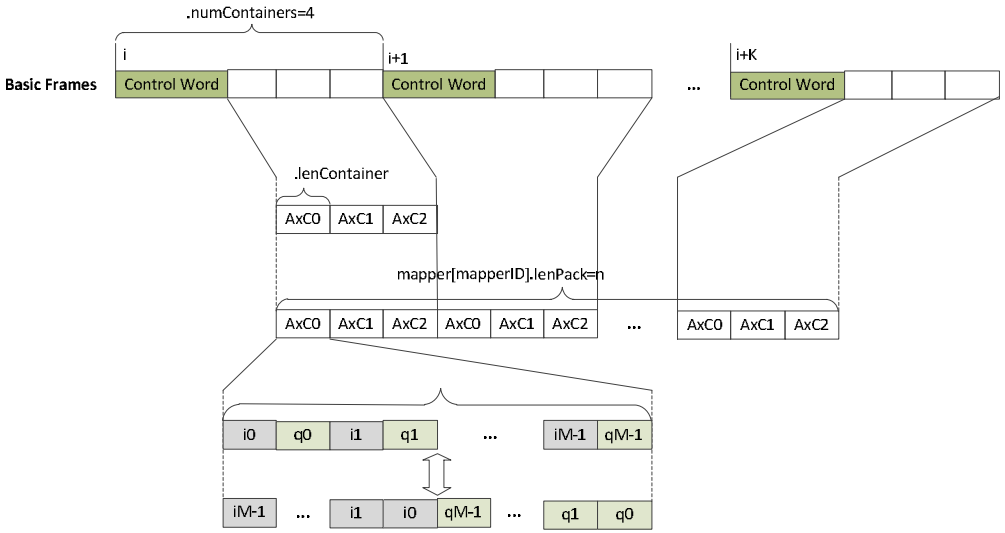
- Capable of describing CPRI mapping method #3.
- Current specification allows up to 8 rules.

Mappers – rule examples



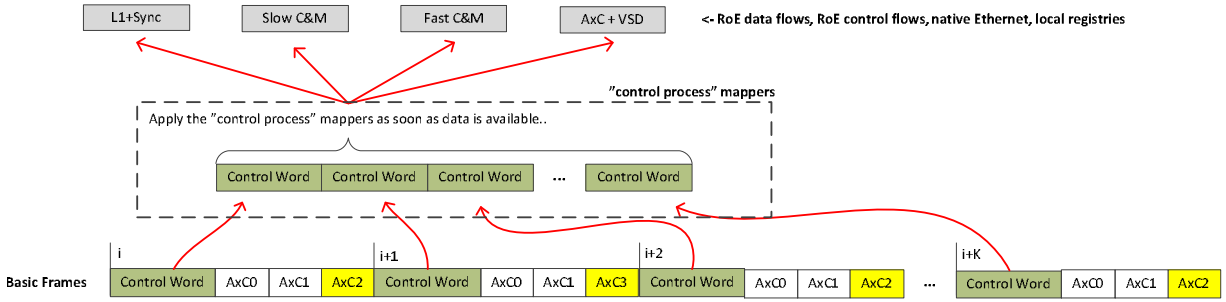


- Several implicit assumptions:
  - container describes on sample.
  - Samples are (de)interleaved.



Mappers – CPRI structure aware and control word handling

- A “control process” handles control words.
  - Collected by the mapper parser (flowID=255).



- Synchronization and L1 protocol fields are only for local consumption and not packetized.
- Slow C&M channel -> packetized as a data flow.
- Fast C&M channel -> packetized as “native” Ethernet packets (i.e., may involve changing the PCS to match the used link).
- VSD and Ctrl_AxC channels -> packetized as RoE control packets.

Mappers – CPRI control word parser

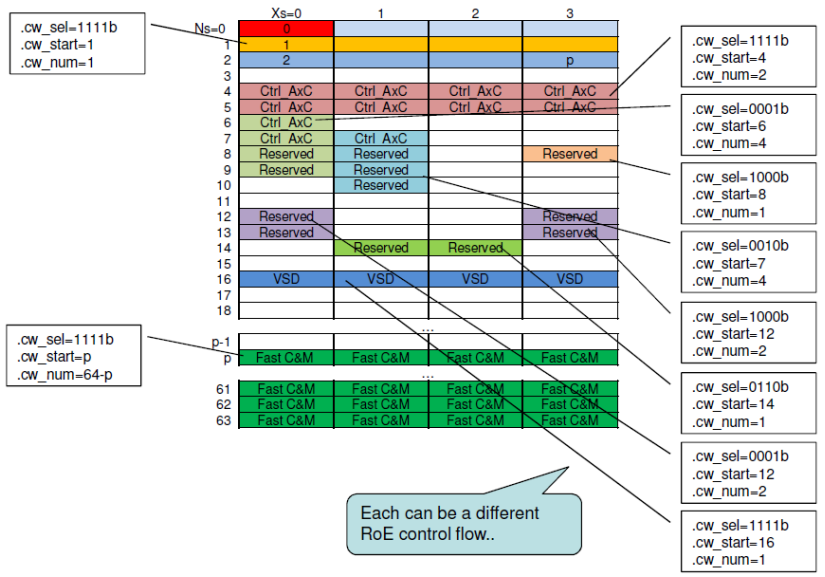
- Fixed “handling” of Slow/Fast C&M and Synchronization and L1 protocol fields.
- VSD and Ctrl_AxC use the following common container construction:

```
{
    .cwSel
    .cwStart
    .cwNum
    .flowID
    .filterMode
    .hfnModulo
    .hfnIndex
    .offset
    .value
    .mask
}
```

Subchannel word bitmask (Xs)

Subchannel #

How many Subchannels

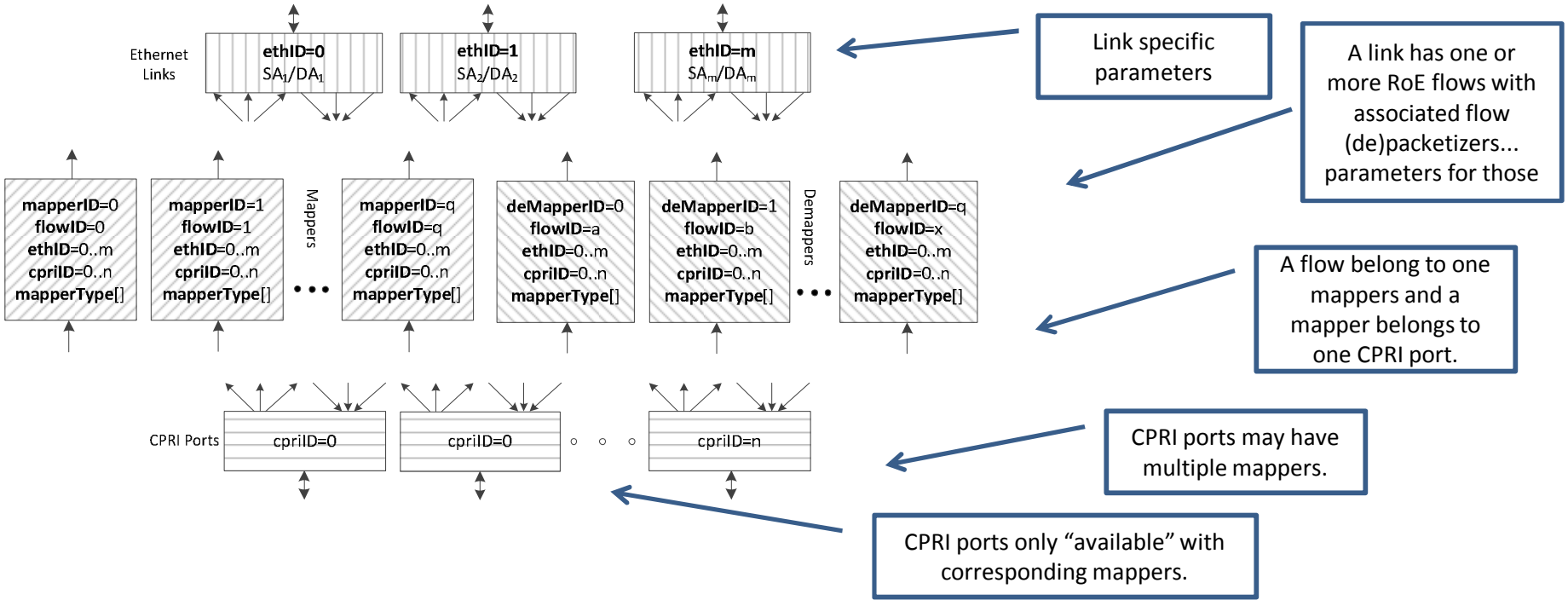


- Up to 8 rules are possible.
- The rule is indicated using the RoE header **flowID**.

.filterMode	Description
0	Filtering is disabled. RoE packet is generated on every Hyper Frame.
1	Non-zero content i.e., the extracted content has non-zero values.
2	Periodic generation according to modulo logic.
3	Pattern match.

Configuration and parameters

- The RoE specification has a lot of configuration variables. They describe how the RoE link/connection and mappers are supposed to behave.
- Some RoE link/connection related parameters (such as compression and ciphering) will have only default – “not used” value defined in this specification.
- Parameters are structured into a hierarchy rather than a register file. This is needed for management and possible control protocol purposes:



Questions? Comments?

Source: Flame warriors