IEEE P1904.3™/D0.2
Draft Standard for Radio over Ethernet Encapsulations and Mappings

Sponsor

**Standards Development Board**of the **IEEE Communications Society**

Approved <XX MONTH 20XX>

**IEEE-SA Standards Board**

Copyright © 2014 by the Institute of Electrical and Electronics Engineers, Inc.

Three Park Avenue

New York, New York 10016-5997, USA

All rights reserved.

This document is an unapproved draft of a proposed IEEE Standard. As such, this document is subject to change. USE AT YOUR OWN RISK! Because this is an unapproved draft, this document must not be utilized for any conformance/compliance purposes. Permission is hereby granted for IEEE Standards Committee participants to reproduce this document for purposes of international standardization consideration. Prior to adoption of this document, in whole or in part, by another standards development organization, permission must first be obtained from the IEEE Standards Activities Department (stds.ipr@ieee.org). Other entities seeking permission to reproduce this document, in whole or in part, must also obtain permission from the IEEE Standards Activities Department.

IEEE Standards Activities Department

445 Hoes Lane

Piscataway, NJ 08854, USA

Abstract: [///Editor’s note: TEXT HERE]

Keywords: [///Editor’s note: TEXT HERE]

**IEEE Standards** documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied “**AS IS**.”

The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation, or every ten years for stabilization. When a document is more than five years old and has not been reaffirmed, or more than ten years old and has not been stabilized, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon his or her independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration. A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal interpretation of the IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position, explanation, or interpretation of the IEEE.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Recommendations to change the status of a stabilized standard should include a rationale as to why a revision or withdrawal is required. Comments and recommendations on standards, and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board

445 Hoes Lane

Piscataway, NJ 08854

USA

Authorization to photocopy portions of any individual standard for internal or personal use is granted by The Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

This introduction is not part of IEEE P1904.3/D0.x

[///Editor’s note: This standard TBD …]

Notice to users

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

This document is copyrighted by the IEEE. It is made available for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making this document available for use and adoption by public authorities and private users, the IEEE does not waive any rights in copyright to this document.

Updating of IEEE documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect. In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE-SA Website at <http://standards.ieee.org/index.html> or contact the IEEE at the address listed previously. For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEE-SA Website at <http://standards.ieee.org/index.html>.

Errata

Errata, if any, for this and all other standards can be accessed at the following URL:
<http://standards.ieee.org/findstds/errata/index.html>. Users are encouraged to check this URL for errata periodically.

Interpretations

Current interpretations can be accessed at the following URL: <http://standards.ieee.org/findstds/interps/index.html>.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA website <http://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or nondiscriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this draft standard was submitted to the IEEE-SA Standards Board for approval, the following is a place holder:

, Working Group Chair

, Editor

The following individuals submitted technical contributions or commented on the draft standard at various stages of the project development.

Name

The following members of the <individual/entity> balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

(to be supplied by IEEE)

Balloter1

Balloter2

Balloter3

Balloter4

Balloter5

Balloter6

Balloter7

Balloter8

Balloter9

When the IEEE-SA Standards Board approved this standard on <XX MONTH 20XX>, it had the following membership:

***(to be supplied by IEEE)***

**<Name>,** *Chair*

**<Name>,** *Vice Chair*

**<Name>,** *Past President*

**<Name>,** *Secretary*

SBMember1

SBMember2

SBMember3

SBMember4

SBMember5

SBMember6

SBMember7

SBMember8

SBMember9

\*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

<Name>, *NRC Representative*

<Name>, *DOE Representative*

<Name>, *NIST Representative*

<Name>

*IEEE Standards Program Manager, Document Development*

<Name>

*IEEE Standards Program Manager, Technical Program Development*

**Contents**

[1 Overview 12](#_Toc434907504)

[1.1 Scope 13](#_Toc434907505)

[1.2 Purpose 13](#_Toc434907506)

[1.3 Coverage 13](#_Toc434907507)

[2 Normative references 14](#_Toc434907508)

[3 Definitions, acronyms, and abbreviations 15](#_Toc434907509)

[3.1 Definitions 15](#_Toc434907510)

[3.2 Acronyms and abbreviations 15](#_Toc434907511)

[3.3 Special Terms 16](#_Toc434907512)

[4 Radio over Ethernet (RoE) base protocol 17](#_Toc434907513)

[4.1 Overview 17](#_Toc434907514)

[4.1.1 Undelying network requirements 17](#_Toc434907515)

[4.1.2 RoE endpoints 17](#_Toc434907516)

[4.1.3 Encapsulation and decapsulation functions 17](#_Toc434907517)

[4.1.4 Mapper function 18](#_Toc434907518)

[4.2 RoE Ethernet Type 18](#_Toc434907519)

[4.3 Bit and octet ordering, and numerical presentation 18](#_Toc434907520)

[4.4 RoE common frame format 19](#_Toc434907521)

[4.4.1 ver (version) field 19](#_Toc434907522)

[4.4.2 pkt\_type (packet type) field 19](#_Toc434907523)

[4.4.3 flow\_id (flow identifier) field 20](#_Toc434907530)

[4.4.4 Timestamp 20](#_Toc434907535)

[4.4.5 Sequence number 21](#_Toc434907536)

[4.4.6 Payload field 21](#_Toc434907540)

[4.5 RoE control packet frame format (000000b) 21](#_Toc434907551)

[4.5.1 ver (version) field 21](#_Toc434907552)

[4.5.2 pkt\_type (packet type) field 21](#_Toc434907553)

[4.5.3 flow\_id (flow identifier) field 21](#_Toc434907556)

[4.5.4 T (timestamp select) field 22](#_Toc434907557)

[4.5.5 timestamp/sequence number field 22](#_Toc434907558)

[4.5.6 subtype field 22](#_Toc434907561)

[4.5.7 Payload field 22](#_Toc434907562)

[4.6 RoE control packet subtype 00000001b format 22](#_Toc434907563)

[4.7 RoE control packet subtype 00000010b format 22](#_Toc434907564)

[4.8 RoE control packet subtype 00000011b format 22](#_Toc434907565)

[4.9 RoE pkt\_type 000000b subtype 00000100b (CPRI control words) 22](#_Toc434907566)

 [*[Editors note : TBD packet format for carrying control words (excuding Fast C&M channel).]* 22](#_Toc434907567)

[4.10 Simple tunneling CPRI data packet (000001b) 22](#_Toc434907569)

[4.10.1 ver (version) field 22](#_Toc434907570)

[4.10.2 pkt\_type (packet type) field 23](#_Toc434907571)

[4.10.3 flow\_id (flow identifier) field 23](#_Toc434907572)

[4.10.4 T (timestamp select) field 23](#_Toc434907573)

[4.10.5 timestamp/sequence number field 23](#_Toc434907574)

[4.10.6 Payload field 23](#_Toc434907575)

[4.11 Structure agnostic CPRI data packet (000010b) 23](#_Toc434907576)

[4.11.1 ver (version) field 23](#_Toc434907577)

[4.11.2 pkt\_type (packet type) field 23](#_Toc434907578)

[4.11.3 flow\_id (flow identifier) field 23](#_Toc434907579)

[4.11.4 T (timestamp select) field 23](#_Toc434907580)

[4.11.5 timestamp/sequence number field 23](#_Toc434907581)

[4.11.6 Payload field 23](#_Toc434907582)

[4.12 Structure aware CPRI data packet (000011b) 23](#_Toc434907583)

[4.12.1 ver (version) field 23](#_Toc434907584)

[4.12.2 pkt\_type (packet type) field 24](#_Toc434907585)

[4.12.3 flow\_id (flow identifier) field 24](#_Toc434907586)

[4.12.4 T (timestamp select) field 24](#_Toc434907587)

[4.12.5 timestamp/sequence number field 24](#_Toc434907588)

[4.12.6 Payload field 24](#_Toc434907589)

[4.13 Native RoE data flow packet (000100b) 24](#_Toc434907590)

[4.13.1 ver (version) field 24](#_Toc434907591)

[4.13.2 pkt\_type (packet type) field 24](#_Toc434907592)

[4.13.3 flow\_id (flow identifier) field 24](#_Toc434907593)

[4.13.4 T (timestamp select) field 24](#_Toc434907594)

[4.13.5 timestamp/sequence number field 24](#_Toc434907595)

[4.13.6 payload data 24](#_Toc434907597)

[4.13.7 Control data 25](#_Toc434907598)

[4.14 Native RoE data flow packet (100100b) with extended header 25](#_Toc434907599)

[4.14.1 extended\_header\_space 25](#_Toc434907600)

[5 Timing and synchronization considerations 26](#_Toc434907601)

[5.1.1 General assumptions 26](#_Toc434907602)

[5.1.2 RoE Presentation time 26](#_Toc434907603)

[5.1.3 Presentation time measurement points 26](#_Toc434907604)

[6 RoE link setup 27](#_Toc434907605)

[6.1 Variables 27](#_Toc434907606)

[6.2 Synchronizing endpoints 27](#_Toc434907608)

[7 RoE mappers 28](#_Toc434907610)

[7.1 Overview 28](#_Toc434907611)

[7.2 Mapper definition for constructing RoE data packet payload 28](#_Toc434907612)

[7.2.1 Container definition 28](#_Toc434907613)

[7.2.2 Segment definition 29](#_Toc434907614)

[7.2.3 Payload example 29](#_Toc434907615)

[7.3 Simple tunneling mapper 30](#_Toc434907616)

[7.3.1 Use of sequence number 30](#_Toc434907617)

[7.3.2 Use of control packets 30](#_Toc434907618)

[7.4 Structure agnostic data packet (000010b) mapper 30](#_Toc434907619)

[7.4.1 Use of sequence number 31](#_Toc434907622)

[7.4.2 Use of RoE control packets 31](#_Toc434907623)

[7.5 Structure aware data Packet (000011b) mapper 31](#_Toc434907624)

[7.5.1 Use of sequence numbers for RoE pkt\_type 000011b subtype 00000100b 33](#_Toc434907625)

[7.5.2 Handling of control words 33](#_Toc434907626)

[7.5.3 Fast C&M channel packets 33](#_Toc434907627)

[Annex A. Header examples 34](#_Toc434907628)

[Annex B. Timestamp calculation example algorithm 35](#_Toc434907629)

[Annex C. Sequence number psudo code 36](#_Toc434907630)

[Bibliography (informative) 37](#_Toc434907631)

# Overview

In order to lower the cost and improve the functionality in a 4G/5G wireless network a centralization of many of the most expensive components of the wireless network is being proposed by the wireless community. This centralization is referred to as a Cloud (or Centralized) Radio Access Network or more commonly as a C-RAN architecture.

The C-RAN architecture lowers the cost because it allows the expensive components of the wireless network to be shared by many antenna sites rather than having dedicated components at/for each antenna site. When an antenna site is busy a greater proportion of expensive resources may be dedicated to that site by reducing the proportion of expensive resources currently dedicated to less busy antenna sites. This can be done dynamically by adaptting to the normal daily changes in traffic patterns within a metro area (or within a building if a pico-cell wireless architecture is being deployed). Likewise, improved functionality becomes possible since a more central location can co-ordinate the Radio Frequency (RF) behavior of multiple antennas, not just those at a single site. For example, this behavior permits multiple antenna sites to co-ordinate transmissions towards a distant user device for the purposes of either improved throughput and/or improved distance.

In order to achieve the goals of a C-RAN architecture, a protocol must exist to separate centrally located expensive parts of the wireless network from the cheaper distributed antenna sites. The nature of this protocol depends in large part on how the functional split is done between the cheaper and the more expensive wireless components. At one extreme, almost all the expensive parts are located next to the antenna site (the current situation) while at the other extreme only very dumb/cheap components are left at the antenna site.

The protocol used to effect this functional split must be carried between the C-RAN central control location and the sites of the antennas. This can be done over dedicated fibers with Wave Division Multiplexing (WDM), it can be done over a Time Domain Multiplexed (TDM) network, or it can be done over a packet switched network. It would be beneficial to carry this protocol over a packet switched Ethernet network due to the ubiquity and cost advantages of Ethernet in the Metro and as a distribution network within buildings. Regardless of the transport mechanism, the network that separates the antenna sites and the C-RAN central control sites is referred to as the Fronthaul network when most of the expensive components are centrally located while the Backhaul network is the case when most of the expensive components are located near the antenna sites. Various intermediate options are also possible and expected with the next generation C-RAN architectures.

Several protocols exist to enable this fronthaul split and the most important one is the Common Public Radio Interface (CPRI). CPRI is an example of an extreme functional split which essentially carries a very low level encoded format that is predominantly just a stream of In-phase/Quadrature (I/Q) samples of the RF signal that the antenna should transmit or receive. CPRI is designed to be transported over an optical fiber pair at a variety of speeds and encoding formats. There are many strict requirements on the protocol such as an end to end delay of no more than 100us and a differential delay of no more than +/- 8ns.

In order to successfully use Ethernet network as a transport network for CPRI or other types of functional split radio protocols for 4G/5G a method of encapsulation of these Radio protocols over Ethernet is required.

Radio Over Ethernet is therefore the use of Ethernet to encapsulate and fronthaul a variety of radio protocols as packets from between the C-RAN and the antenna sites.

There are broadly two types of encapsulation techniques. The simplest encapsulation mechanisms are oblivious to the structure of the protocol they are encapsulating. This we refer to as the structure–agnostic mechanism. The alternative, is an encapsulation mechanism that is partially aware of the structure of the protocol it encapsulates and we refer to this as structure-aware mechanism. This mechanism allows various optimizations.

## Scope

1. This document defines the encapsulation and mapping of radio protocols to be fronthauled over Ethernet networks. Furthermore, both structure-agnostic and structure-aware definitions are provided for the most common and current radio protocol – Common Public Radio Interface (CPRI).
2. This standard does not specify whether or how the Ethernet packets are guaranteed the strict QoS required by the encapsulated radio protocols. It does however recommend the support of 802.1CM profile or equivalent to help ensure such QoS guarantees. Alternative transparent mechanisms are also permitted in conjunction with or in lieu of 802.1CM profile.
3. We expect that a full implementation of Radio over Ethernet would comprise;
4. the above mentioned encapsulations,
5. a networking tehnology that minimizes delay and PDV,
6. a clock distribution mechanism, and
7. ingress/egress mapping functions that encapsulate/decapsulate while dejittering and retiming the recovered signal.

This specification is concerned with encapsulation and mapping only.

## Purpose

1. The purpose of this standard is to describe the exact header formats and packet encapsulations required to;

transport any newly defined fronthaul protocol over Ethernet (i.e. the “native RoE” encapsulation that can be used as-is or used to transport other existing fronthaul protocols in a ways described in the next two bullet points

Transport a C-RAN Radio Fronthaul protocol such as CPRI in a bit transparent manner (structure agnostic) over Ethernet.

Transport the C-RAN Radio Fronthaul protocol CPRI over Ethernet where knowledge of the frame format is used to optimize the choice of packet sizes/headers/alighnment etc.

## Coverage

1. [///Editor’s note: This specification provides TBD ...]

# Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

# Definitions, acronyms, and abbreviations

## Definitions

For the purpose of this document, the following terms and definitions apply. The IEEE Standards Dictionary Online should be consulted for terms not defined in this clause.[[1]](#footnote-1)

[///Editor’s note: TBD]

## Acronyms and abbreviations

4G/5G – Fourth Generation and Fifth Generation wireless networking technologies.

BF – Basic Frame

BFN – Node B Frame Number

BER – Bit Error Rate

CPRI – Common Public Radio Interface

CRAN /C-RAN – Cloud / Centralized Radio Access Network

Endpoint – The original sender or the final receiver of a RoE communication

FDD – Frequency Division DuplexI/Q - Inphase and Quadrature

Jitter – Deviation in clock frequency from true periodicity

LAN – Local Access Network

LSB – Least Significant Bit

MSB – Most Significant Bit

PDV – Packet Delay Variation

RF – Radio Frequency

RoE – Radio over Ethernet

TAI – International Atomic Time

TLV – Type Length Value

ToD – Time of DayUTC – Universal Coordinated Time

UTRA – Universal Terrestrial Radio Access (3GPP)

VLAN – Virtual LAN

WDM –Wave Division Multiplexing

## Special Terms

///Term: Definition

# Radio over Ethernet (RoE) base protocol

[///Editor’s Note: this clause will describe the native RoE encapsulation transport format. The following subclauses will also describe the overall RoE architecture, showing encapsulation and decapsulation function locations, and the mapper function locations. This clause also lists the underlying assumptions a RoE enabled architecture has.]

## Overview

### Undelying network requirements

[///Editor’s note: Tbd].

1. A mesh network comprised of bridges and point to point Ethernet links
2. The number of actual links and nodes are not in scope as long as the delay and the PDV are within the required timing.[///Comment ‘what is the required timing’]
3. The network will need management for delay and packet delay variation
4. Highly managed network [///Comment: Define ‘highly’]
5. Support for ToD distribution if there is no other means for end points for clock sync
6. No retransmission (minimum affective 1012 BER) for RoE traffic.
7. Network is required to have sufficient bandwidth to carry RoE traffic.
8. The maximum one way delay has to be less than half of the available roundtrip delay.
9. Ethernet network that preserves the frame source and destination addresses.

### RoE endpoints

This document uses terms **endpoint** and **RoE endpoint** meaning a RoE capable networking node that is either the originator or the final receiver of a RoE communication. There may be zero or more intermediate networking nodes between RoE endpoints.

Only RoE end-points are required to be RoE aware.

This document also further details the roles of the RoE endpoints in places where it is important to know whether the endpoint has a role of a **RoE master** or a role of a **RoE slave** during the communication

[///Editor’s note: TBD: port mode? ]

[///Editor’s note: Can Ethernet port handle multiple mappers at the same time or just one? Proposal – one type.]

### Encapsulation and decapsulation functions

[///Editor’s note: Tbd]



Figure 1 - RoE endpoints and supported functions

### Mapper function

A mapper is a function/process that is capable of converting other transport framing formats to a native RoE framing format and vise versa. This specification describes two flavors of mappers. The first concerns how to separate control data and sample data parts, and packetize the sample data. The second concerns how to process and packetize the control data. The latter is called as “control process” in this specification.

### Control data and the control process

If segments also contain control data, those are handled by a “control process” whose responsibility is to collect a reasonable amount of control data (based on the segment and container rules) before constructing a separate RoE control packet or other Ethernet packet (e.g. in a case of CPRI Fast C&M channel). The control process is responsible for meeting possible timing constraints on delivering control data within the required time frame. The behavior of the “control process” shall be defined by each mapper that makes use of the control data.

## RoE Ethernet Type

All RoE packets shall use the EtherType value shown in Table 1.

Table 1 - RoE EtherType

|  |  |
| --- | --- |
| **Purpose** | **EtherType** |
| RoE packet | XX-XX16 (\*) |

[///Editor’s note: (\*) The value will be assigned at Sponsor Ballot time.]

## Bit and octet ordering, and numerical presentation

This document assumes network byte ordering (i.e. big endian). Figure 3 illustrates the bit ordering and numbering within an octet. Similarly Figure 2 illustrates the bit and octet ordering, and corresponding numbering within a 32 bit word.

 

Figure - bit ordering and numbering within an octet

 

Figure - bit and octet ordering and numbering within a 32 bit word

The following numerical notations are used in this document:

* Integer value has no specific notation, for example: 69
* Hexadecimal value has a prepended “0x”, for example: 0xdeadbeef
* Binary value has a trailing “b”, for example: 11001010b

## RoE common frame format

This subclause documents the first 6 or 10 octets of the frame (i.e. the RoE header) that are common to all RoE flow data and control packets. Figure 2 illustrates the frame format and its fields. The offset zero (0) is the first octet of the RoE frame. The common RoE frame has the following header fields:

1. **ver** (version) field: 2 bits
2. **pkt\_type** (packet type) field: 6 bits
3. **flow\_id** (flow identifier) field: 8 bits
4. **timestamp** (timestamp/seqnum) field: 32 bit



Figure 2: RoE encapsulation common frame format – the RoE header

There is no dedicated field for the RoE packet **payload** size. The lower layers transporting (e.g. Ethernet MAC in this specification) has to provide a means for RoE application to determine the size of its payload.

[///Editor’s note: Add a note here that RoE packets may have variable size even within one flow..]

The RoE header is placed into the transport protocol payload field, which in this document context is the Ethernet frame payload field.

### ver (version) field

The **ver** field indicates the version of the RoE protocol and RoE common format for data, control and future packet types.

The version defined in this document is zero (00b). The **ver** field shall be set to 00b.

### pkt\_type (packet type) field

The 6 bit **pkt\_type** field is used to define the RoE packet type and the type of flow carried by the RoE packets. This document reserves packet types listed in Table 2.

Table 2 – RoE pkt\_type values

|  |  |  |
| --- | --- | --- |
| **Binary value** | **Function** | **Description** |
| 000000b | Control Packet | Control packet between two RoE endpoints |
| 000001b | Simple tunneling CPRI data packet | Data payload packet  |
| 000010b | Structure agnostic CPRI data packet | Data payload packet with 6 octet RoE frame header and structure agnostic CPRI payload. |
| 000011b | Structure aware CPRI data packet | Data payload packet with 6 octet RoE frame header and structure aware CPRI payload. |
| 000100b | Native RoE data flow packet | Data payload packet with 6 octet RoE frame header. |
| 100100b | Native RoE data flow packet  | Data payload packet with 10 octet RoE frame header. |
| 000101b | Slow C&M CPRI packet | Data payload packet with 6 octet RoE frame header and structure aware CPRI Slow C&M payload. |
| 000100b – 011111b | -- | Reserved for future packet types |
| 100010b – 111111b | -- | Reserved for future packet types |

### flow\_id (flow identifier) field

The **flow\_id** identifies a specific flow between two endpoints. The endpoints are defined as Ethernet packet Source Address (SA) and Destination Address (DA) pair in the context of this specification. The **flow\_id** allows multiplexing up to 256 flows between two endpoints.

The **flow\_id** identifier has no routing function and is solely interpreted by the endpoints. The **flow\_id** identified flow may consist of multiple subflows (i.e. a group flow). The interpretation of flow content and possible subflows is solely controlled by the endpoints.

This document reserves flow identifier values listed in Table 3.

Table 3 – RoE flow\_id values

|  |  |  |
| --- | --- | --- |
| **Binary value** | **Function** | **Description** |
| 0000000b | NIL flow\_id | Reserved flow\_id indicating that the field shall not be interpreted as any specific flow.  |
| 0000001b – 1111111b | flow\_id number | Flow identifiers available for use to identify specific flows between two endpoints. |

### Timestamp

The **timestamp** field is 32 bits in size and expresses the absolute time for presentation, relative to a defined reference plane, of the information within the packet at the receiving endpoint of the RoE packet. Both the transmitting and receiving endpoints must share the same understanding of the Time of Day (ToD) in order for the information to be presented at the desired time.

The format of the timestamp field is shown in Figure 2.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | … | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| timestamp (integer ns) | timestamp (fractional ns) |

Figure 2: Format of the timestamp field

The 30 most significant bits of the timestamp field count in units of nanoseconds and the value ranges from 0ns to 999,999,999ns (0x0 to 0x3B9AC9FF, respectively). The two least significant bits of the timestamp field count in units of 0.25ns and the value ranges from 0 to 0.75ns (0x0 to 0x3, respectively). If sub-nanosecond timestamping is not used, these two bits shall be set to 0 at the sender and shall be ignored at the receiver.

The timestamp value is capable of expressing a presentation time of up to one second in the future. Both endpoints must account for the rollover condition of the timestamp field after 999,999,999.75ns. See Annex B for example algorithms that truncate the presentation time into the timestamp field at the transmitting endpoint and that recover the presentation time from the timestamp field at the receiving endpoint. Refer to subclauses 4.10.4, 4.11.4, 4.12.4, and 4.13.4 for more details on the usage of the timestamp and the presentation time.

### Sequence number

[///Editor’s note: NEW TEXT HERE]

~~The sequence number field is 31 bits in size and wraps to seqNumMinimum after exceeding its maximum value seqNumMaximum-1. The highest value for the seqNumMaximum is 2^31-1. The following shall hold: 0≤seqNumMinimum<seqNumMaximum-1. The sequence number is increased by a constant value seqNumIncrement known by both RoE packet sending and receiving endpoint. The seqNumIncrement shall comply with: seqNumIncrement<(seqNumMaximum- seqNumMinimum-1).~~

~~The sequence number is initialized to an implementation specific value seqNumStart between seqNumMinimum and seqNumMaximum-1 at the endpoint reset. The internal structure of the sequence number is known and interpreted by RoE endpoints~~.

### Payload field

The content, structure and size of the payload field is specific to a RoE packet type and its definition. The payload may contain a flow of In-phase and Quadrature (I/Q) samples for a single antenna carrier or a group of antenna carriers. Both single and group content is identified by a **flow\_id** between two RoE endpoints. Furthermore, when specific mappers are applied the payload field can contain, for example, an individual antenna carrier component flow of a decomposed CPRI basic frame. In a case of RoE control packets the payload may contain appropriate control and management information, for example, in a form of TLVs or other encoding scheme.

The total RoE payload field size shall always be full octets. If payload size modulo 8 is not 0 then the last octet of the payload is added trailing padding 0-bits until the payload size modulo 8 is 0.

## RoE control packet frame format (000000b)

This subclause documents the first 6 or 10 octets of the frame that is common to RoE control packets. Figure 5 illustrates the frame format and its fields. The RoE contral packet frame format follows the generic RoE frame format defined in subclause 0 unless stated otherwise.



Figure 5: RoE Control Packet common frame format

### ver (version) field

See subclause 4.2.1.

### pkt\_type (packet type) field

The **pkt\_type** field for a RoE Control Packet shall be set to value 000000b (see Table 2).

### flow\_id (flow identifier) field

The **flow\_id** field shall be set to value 0000000b (see Table 3) unless otherwise specified by a RoE Control Packet **subtype** definition. See subclause 4.4.8 for further details regarding the RoE Control Packet subtypes.

### timestamp/sequence number field

See subclause 1.1.1.1. Note that sequence numbers may behave differently between RoE control packets and their associated RoE data packet flows. For example the **seqNumMaximum** and **seqNumIncrement** can be different for RoE control packets and data packets. The RoE control packet subtype specification shall describe the exact sequence number handling.

### subtype field

The **subtype** field is size of 8 bits and defines additional control packet types. This document reserves Control Packet **subtype** values listed in Table 4.

Table 4 – RoE Control Packet subtype values

|  |  |  |
| --- | --- | --- |
| **Binary value** | **Function** | **Description** |
| 0000 0000b | -- | Reserved for future use |
| 0000 0001b | Tbd | Reserved for RoE |
| 0000 0010b | Tbd | Reserved for RoE |
| 0000 0011b | Tbd | Reserved for RoE |
| 0000 0100b | CPRI11 mapper control words | Ctrl\_AxC and VSD data. |
| 0000 0101b – 1111 1111b | Reserved | Control packet subtypes available for use between two RoE endpoints. |

### Payload field

See subclause 4.2.7. for generic definition. The content depends on the control packet subtype.

## RoE control packet subtype 00000001b format

Tbd for RoE endpoint dynamic discovery and configuration purposes.

## RoE control packet subtype 00000010b format

Tbd for RoE endpoint dynamic discovery and configuration purposes.

## RoE control packet subtype 00000011b format

Tbd for RoE endpoint dynamic discovery and configuration purposes.

## RoE pkt\_type 000000b subtype 00000100b (CPRI control words)

[///Editor’s note : TBD packet format for carrying control words (excuding Fast C&M channel).]

## Simple tunneling CPRI data packet (000001b)

### ver (version) field

See subclause 4.2.1.

### pkt\_type (packet type) field

The **pkt\_type** field for a simple tunneling data packet shall be set to value 000001b (see Table 2).

### flow\_id (flow identifier) field

[///Editor’s note: TEXT HERE]

### timestamp/sequence number field

[///Editor’s note: TEXT HERE]

### Payload field

[///Editor’s note: TEXT HERE]

## Structure agnostic CPRI data packet (000010b)

### ver (version) field

See subclause 4.2.1.

### pkt\_type (packet type) field

The **pkt\_type** field for a structure agnostic data packet shall be set to value 000010b (see Table 2).

### flow\_id (flow identifier) field

[///Editor’s note: TEXT HERE]

### timestamp/sequence number field

[///Editor’s note: TEXT HERE]

### Payload field

[///Editor’s note: TEXT HERE]

## Structure aware CPRI data packet (000011b)

### ver (version) field

See subclause 4.2.1.

### pkt\_type (packet type) field

The **pkt\_type** field for a structure aware data packet shall be set to value 000011b (see Table 2).

### flow\_id (flow identifier) field

[///Editor’s note: TEXT HERE]

### timestamp/sequence number field

[///Editor’s note: TEXT HERE]

### Payload field

[///Editor’s note: TEXT HERE]

## Native RoE data flow packet (000100b)

This subclause describes the native RoE data packet format. The packet payload carries a single flow or a group flow of radio sample data between two RoE endpoints. The common RoE frame header content is described in subclause 0.

### ver (version) field

See subclause 4.2.1.

### pkt\_type (packet type) field

The **pkt\_type** field for a simple tunneling data packet shall be set to value 000001b (see Table 2).

### flow\_id (flow identifier) field

[///Editor’s note: TEXT HERE]

### timestamp/sequence number field

[///Editor’s note: TEXT HERE]

### payload data

See subclause 4.2.7 for the generic definition.



Figure 7 – I/Q sample data container and bit ordering

### Control data

If segments also contain control data, those are handled by a “control process” whose responsibility is to collect a reasonable amount of control data (based on the segment and container rules) before constructing a separate RoE control packet or other Ethernet packet (e.g. in a case of CPRI Fast C&M channel). The control process is responsible for meeting possible timing constraints on delivering control data within the required time frame. The behavior of the “control process” shall be defined by each mapper that makes use of the control data.

## Native RoE data flow packet (100100b) with extended header

This subclause describes the native RoE data packet format with extended\_header\_space added to the common RoE frame header. The packet payload carries a single flow of radio sample data between two RoE endpoints. The RoE packet except for the extended\_header\_space is described in subclause 4.5.

#  [///Editor’s note: TEXT HERE]Timing and synchronization considerations

[///Editor’s note: This clause lists for example reference time assumptions, and how the synchronization is realized in general.]



Figure 8: Presentation time measurement points

### General assumptions

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, when available, shall provide Time of Day (ToD) in nanoseconds and synchronized to international atomic time (TAI).

### RoE Presentation time

The RoE presentation time is used to achieve time synchronization between the RoE endpoints. The presentation time is calculated by the RoE sender and represents the time when the RoE packet payload has to be played out from the RoE receiver packet buffer to the consumer of the payload data.

### Presentation time measurement points

Figure 8 illustrates the measurements planes for the RoE presentation time. When a RoE sender calculates the presentation time at the RoE receiver, it has to take the entire end to end delay between the RoE sender and receiver reference planes into account. The end to end delay consists of the networking delay (i.e. the transit time), processing and buffering delays at both RoE endpoints. The buffer at the RoE receiver side has to be big enough to compensate packet delay variation introduced by the network and internal processing at both endpoints.

The method for measuring the end to end delay is implementation and deployment specific.

# RoE link setup

[///Editor’s note: TEXT HERE. How much scope are we putting here?]

## Variables

Table List of variables

|  |  |
| --- | --- |
| RoE.numContainers |  |
| RoE.numSegments |  |
| RoE.container[0..n].lenContainer |  |
| RoE.segment.lenSkip |  |
| RoE.segment.lenSegment |  |
| RoE.segment.flow\_id |  |
| RoE.container[0..n].lenContainer |  |
| RoE.container[0..n].flow\_id |  |
| RoE.container[0..n].modulo |  |
| RoE.container[0..n].index |  |
| RoE.container[0..n].ctrl |  |
| seqNumIncrement |  |
| seqNumMinimum |  |
| seqNumMaximum |  |
| seqNumStart |  |
| CPRI10.lenBasicFrame |  |
| CPRI10.numBasicFramesPerPacket |  |
| tstampWindowSize |  |
| tstampWindowMask |  |
| tstampTstampMask |  |

[///Editor’s note: TEXT HERE]

## Synchronizing endpoints

[///Editor’s note: TEXT HERE]

# RoE mappers

[///Editor’s note: This clause defines one or more mappers to/from existing radio framing formats to/from RoE native transport encapsulation format.]

## Overview

[///Editor’s note: This subclause defines a mapper to/from CPRI v6.1 framing to/from RoE transport. It captures both structure agnostic and structure aware cases. Proposal to handle 8B/10B and 64B/66B CPRI PHYs as separate mappers. ]

## Mapper definition for constructing RoE data packet payload

[///Editor’s note: Need define scope here. I believe this is for both struture aware and agnostic. but not for tunnelling or native?]

The content of the payload field is divided into **RoE.numContainers** bit fields (i.e. containers) that again can be repeated **RoE.numSegment** times. A **RoE.Container[0..RoE.numContainers-1]** array is described as below:

**{**

 **.flow\_id**

 **.ctrl**

 **.lenSkip**

 **.lenContainer**

 **.modulo**

 **.index**

**}**

Each segment is described using **RoE.segment** that has a similar content as a container:

**{**

**.flow\_ids**

 **.lenSkip**

 **.lenSegment**

**}**

[///JK Comment: Actually, the RoE.Container[] has exactly the same expression power as RoE.segment when setting .ctrl accordingly and .modulo to 0. We can consider merging the two but for now they are here to make separation a bit "clearer" what is per packet non-radio sample data and what is "control".]

The container and the segment descriptions work for both extracting data from some source and describing the construction of the RoE payload field as well.

The RoE.numSegments also implicitly defines the amount of data collected before starting to construct one or more RoE packets

### Container definition

The “**flow\_id**” identifies to which RoE flow or group of flows this container belongs to. Typically the “**flow\_id**” equals to an antenna carrier that is placed into separate RoE packets/flows. The “**ctrl**” defines whether this contrainer is about control (1) or data payload (0). This selection can be used, for example, output data selectively to control or data RoE packets flows.

The “**lenSkip**” describes the number of unused bits and “the **lenContainer**” the number of actual payload bits per each container. Note that the “**lenSkip**” bit are only affective when extracting/storing data from/to some other source than RoE payload field. When containers are stored into or read from the RoE payload field “skip bits” are not written or read.

The “**modulo**” allows skipping containers and skipped containers are handled in a same way as “**lenSkip**” bits. The modulo operation is applied to a sequence of input data that is counted from 0 to **RoE.numSegments**-1. The segment to select is matched comparing the “**index**” to the output of the modulo operation. The “**modulo**” value 0 means container skipping is not used. For example, to skip every second input container set the modulo to 2 to keep every container set the modulo to 1 (and index to 0) and to turn off modulo logic set the modulo to 0..

The above scheme allows constructing rather compex payload fields as well as very simple ones. The container definitions are per direction i.e. there may be different values for transmit and receive directions.

### Segment definition

The “**flow\_ids**” identifies to which RoE flows or group of flows this segment belongs to. The “**flow\_ids**” may equal to a single antenna carrier that is placed into separate RoE data packets/flows or may equal to a list antenna carriers.

[///Editor’s note: Need to describe uni/bi directionality]

The “**lenSkip**” and the “**lenSegment**” for the segment describe a bit field that precedes all containers within a segment. The bit field described by the “**lenSkip**” and the “**lenSegment**” are not meant for the RoE data packets/flows in a typical case and are likely to require additional control processing before being packetized into any RoE packets.

### Payload example

Figure 6 illustrates how containers and segments relate to each other. The figure is just an example of many possible configurations.



Figure - relation between segments and containers

In the case when a container carries sample data in a form of I/Q components the samples shall be arranged and stored as shown in Figure 7. Effectively bits are stored in a network order (the most significant bit comes first) into the payload field, first the whole I component followed by the whole Q component of the antenna I/Q sample data stream.

In this example one possible way to express 64 time 15 bits I/Q sample pairs as a one antenna carrier flow could be:

1. **RoE.numSegments**=64
2. **RoE.segment.lenSkip**=0
3. **RoE.segment.lenSegment**=0
4. **RoE.numContainers**=1
5. **RoE.container[0].ctrl**=0
6. **RoE.container[0].lenSkip**=0
7. **RoE.container[0].lenContainer**=30 (i.e. 2\*15 bit sample components)
8. **RoE.container[0].modulo**=0

Note that the example assumes I/Q samples are not interleaved. No padding is required.

If the payload is not I/Q sample data the same bit ordering, continuous storing and padding of bits shall still apply. CPRI structure agnostic mapper

This subclause defines a structure agnostic CPRI to RoE mapper. This mapper does not interpret the CPRI frame content in any way. The mapper packetizes a number of CPRI Basic Frames into a RoE packet payload.

This mapper shall remove the 8B/10B line coding used by CPRI for line rate options 1 to 7. The mapper shall be aware of the start of the radio frame. In the context of this mapper and CPRI v6.1 specification the radio frame is the 10ms frame number, which for UTRA-FDD would be aligned with NodeB Frame Number (BFN).

## Simple tunneling mapper

This document names simple tunneling mapper as “**CPRI00**”.

The simple tunneling mapper is very simple. The entire serial CPRI data stream is simply encapsulated in the payload.

### Use of sequence number

Since all frame timing, including K28.5, HFN and BFN are preserved within the fully encapsulated CPRI stream in the payload, the sequence number is only useful to detect dropped packets.

### Use of control packets

## The simple tunneling mapper “CPRI00” does not use any RoE control packets. Structure agnostic data packet (000010b) mapper

This document names structure agnostic CPRI mapper as “**CPRI10**”.

The mapper extracts/stores **CPRI10.lenBasicFrame** octets from/to the CPRI stream i.e. an individual CPRI Basic Frame (BF). **CPRI10.numBasicFramesPerPacket** are stored/extracted to/from RoE packets. If **CPRI10.numBasicFramesPerPacket**>1 then the mapper shall ensure the BF that starts the 10ms radio frame is the first BF in the RoE packet payload. For each RoE packet that starts the 10ms radio frame the RoE header **S**=1. Otherwise the **S**=0.

Other RoE configuration parameters shall be set as follows:

1. **RoE.numSegments**=**CPRI10.numBasicFramesPerPacket**
2. **RoE.segment.lenSkip=0**
3. **RoE.segment.lenSegment=0**
4. **RoE.numContainer**=1
5. **RoE.container[0].lenSkip**=0
6. **RoE.container[0].lenContainer**=**CPRI10.lenBasicFrame**\*8
7. **RoE.container[0].flow\_id**=?
8. **RoE.container[0].ctrl**=0
9. **RoE.container[0].modulo**=0
10. **seqNumMinimum**=0
11. **seqNumMaximum**=256\*150/**CPRI10.numBasicFramesPerPacket**
12. **seqNumIncrement**=1

[///Editor’s note: Draw example figure here.]

### Use of sequence number

The sequence number is incremented by one (1) for each sent RoE data packet and the sequence number wraps around every 256\*150/**CPRI10.numBasicFramesPerPacket** sent packets (e.g. if there are 8 BFs per RoE packet the **seqNumMaximum** is 4800). Note that sequence number may be non-zero when RoE header S-flag is set.

### Use of RoE control packets

## There are no associated control packets for the “CPRI10” mapper. Structure aware data Packet (000011b) mapper

This document names structure-aware CPRI mapper as “**CPRI11**”.

[///Editor’s note: This is what we called the “better” mapper.]

[///Editor’s note: Proposal to require that GSM and other “non-UMTS Chip” antenna carriers are already resampled to some integer divisible UMTS Chip rate within CPRI traffic before the mapper is applied.]

This subclause defines a structure-aware CPRI to RoE mapper that looks into the CPRI frame and is able to further divide its content into different components. The mapper packetizes a number of CPRI Basic Frames worth of I/Q samples/AxC Containers for one AxC into a RoE data packet payload.

This mapper shall remove the 8B/10B line coding used by CPRI for line rate options 1 to 7. The mapper shall be aware of the start of the radio frame. In the context of this mapper and CPRI v6.1 specification the radio frame is the 10ms frame number, which for UTRA-FDD would be aligned with NodeB Frame Number (BFN).

The RoE header sequence numbers are used i.e. T-flag shall be set to zero (0) for both RoE data and control packets.

[///Editor’s note: This is rather under specified and assumes that everything complies to some integer fraction of UMTS Chip rate.]

The mapper extracts/stores **CPRI11.lenBasicFrame** octets from/to the CPRI stream i.e. an individual CPRI Basic Frame (BF). The mapper buffers **CPRI11.numBasicFramesForRoEPacket** worth of CPRI BFs and then stored/extracted individual AxC containers to/from one or more RoE packets. If **CPRI11.numBasicFramesForRoEPacket**>1 then the mapper shall ensure the BF that starts the 10ms radio frame is the first BF in the RoE packet payload. For each RoE packet that starts the 10ms radio frame the RoE header **S**=1. Otherwise the **S**=0.

The below RoE configuration parameter example is for CPRI line rate option 3 (assuming 20MHz LTE and 2x2 MIMO) and CPRI mapping method 1 without any stuffing bits. The AxC0 has flow\_is 1 and the AxC1 has flow\_is 2 i.e., there will be two RoE data packet flows.

The I/Q sample size is 15 bits per component.The AxC Container Block contains 256 BFs. One RoE data packet will contain 64 I/Q samples i.e. 8 BFs worth of samples.

The RoE configuration parameters shall be set as follows:

1. **RoE.numSegments**=**CPRI11.numBasicFramesForRoEPacket=8**
2. **RoE.segment.lenSkip=0**
3. **RoE.segment.lenSegment=32**
4. **RoE.segment.flow\_ids=1,2**
5. **RoE.numContainer**=16
6. **RoE.container[0,2,4,6,8,10,12,14].lenSkip**=0
7. **RoE.container[0,2,4,6,8,10,12,14].lenContainer**=**30**
8. **RoE.container[0,2,4,6,8,10,12,14].flow\_id**=1
9. **RoE.container[0,2,4,6,8,10,12,14].ctrl**=0
10. **RoE.container[0,2,4,6,8,10,12,14].modulo**=0
11. **RoE.container[1,3,5,7,9,11,13,15].lenSkip**=0
12. **RoE.container[1,3,5,7,9,11,13,15].lenContainer**=**30**
13. **RoE.container[1,3,5,7,9,11,13,15].flow\_id**=2
14. **RoE.container[1,3,5,7,9,11,13,15].ctrl**=0
15. **RoE.container[1,3,5,7,9,11,13,15].modulo**=0
16. **seqNumMinimum**=0
17. **seqNumMaximum**=256\*150/**CPRI11.numBasicFramesForRoEPacket**
18. **seqNumIncrement**=1

The RoE.container definition describes 16 container fields, 8 for each AxC. This creates two RoE data packet flows with different flow\_id and each RoE data packet then contains 8\*8\*30 bits worth of CPRI AxC Containers.

The same above example with 14 bits per I/Q component i.e. there would be total 16 bits of stuffing in each BF after the control word:

1. **RoE.container[0].lenSkip**=16
2. **RoE.container[2,4,6,8,10,12,14].lenSkip**=0
3. **RoE.container[0,2,4,6,8,10,12,14].lenContainer**=28
4. **RoE.container[0,2,4,6,8,10,12,14].flow\_id**=1
5. **RoE.container[0,2,4,6,8,10,12,14].ctrl**=0
6. **RoE.container[0,2,4,6,8,10,12,14].modulo**=0
7. **RoE.container[1,3,5,7,9,11,13,15].lenSkip**=0
8. **RoE.container[1,3,5,7,9,11,13,15].lenContainer**=28
9. **RoE.container[1,3,5,7,9,11,13,15].flow\_id**=2
10. **RoE.container[1,3,5,7,9,11,13,15].ctrl**=0
11. **RoE.container[1,3,5,7,9,11,13,15].modulo**=0

///[Editor’s note: Draw example figure here.]

Use of sequence numbers for RoE pkt\_type 000011b

The sequence number is incremented by one (1) for each sent RoE data packet and the sequence number wraps around every 256\*150/**CPRI11.numBasicFramesForRoEPacket** sent packets (e.g. if there are 8 BFs worth of I/Q samples for one AxC per RoE packet the **seqNumMaximum** is 4800). Note that sequence number may be non-zero when RoE header S-flag is set.

### Use of sequence numbers for RoE pkt\_type 000011b subtype 00000100b

[///Editor’s note: TEXT HERE]

[///Editor’s note: would contain the BFH and the HFN the control words belong to.]

### Handling of control words

The structure aware, “CPRI11” mapper has multiple “control process” mappers to process CPRI Hyper Frame control words. Table 5 lists the control process mappers, their respective naming/variable prefixes and which sub-channels they (typically) concern. The mapper definitions and variables have the common prefix **CPRI11.ctrl**. Unless otherwise stated all variables are assumed to be prefixed with **CPRI11**.

The **CPRI11.ctrl.cw\_size** defines the size of the CPRI control word (Tcw) in octets i.e. Tcw/8 for all control process mappers. The value shall be between 1 and 16. Note that in cases where the CPRI control word size (Tcw) is less that the CPRI word size (T) the mapper shall handle required padding or skipping unused bits implicitly. Unused bits are intrepreted as zero (0) bits.

Table Control Process mappers for CPRI control words

|  |  |  |
| --- | --- | --- |
| **Mapper description** | **CPRI control words** | **Mapper prefix** |
| Synchronization and L1 protocol fields | Sub-channels 0 & 2 | CPRI11.ctrl.sync\_l1 |
| Slow C&M channel | Sub-channel 1 | CPRI11.ctrl.slow |
| Fast C&M channel | Sub-channels p->63 | CPRI11.ctrl.fast |

[///Editor’s note: mechanism required for AxC\_ctrl and VSD data.]

The control words the “control process” mappers operate upon is structured in the same way as CPRI structures it. Areas not extracted or transported to/from the CPRI stream are assumed to be all zero (0). Figure 9 illustrates the known CPRI Hyper Frame control words construction. From the processing point of view control words are always processed in the order they arrive i.e., the first element is the control word 0 (Ns=Xs=0), the second element is the control word 1 (Ns=1, Xs=0), etc. The control process mappers are applied to the control words as they arrive and then stored into the memory as a dense array for possible second stage processing (such as placing data into RoE data packet payload field).

[///Editor’s note: picture required.]

Figure CPRI Hyper Frame worth of Control Words

The control process mappers for Slow C&M and Fast C&M channelsblocks use the following common **container** construction:

**{**

 **.cw\_sel**

 **.cw\_start**

 **.cw\_num**

**.flow\_id**

**.filter\_mode**

**.hfn\_modulo**

**.hfn\_index**

**.offset**

**.value**

**.mask**

**}**

There can be, depending on the case, one or more containers per control process mapper. The control process mapper for Slow C&M and Fast C&M have the following common construction:

**{**

 **.container[0..n]**

**}**

[///Editor’s note: Need to agree on mask scheme]

The .cw\_sel is a four bit mask for selecting sub-channel words (Xs) to extract. The selected sub-channel words equal to “logical or” of corresponding sub-channel masks. See Table 6 for the mask values.

Table 6 Sub-channel word bit masks

|  |  |
| --- | --- |
| Binary mask | Sub-channel word (Xs index) |
| 0000b | sub-channel processing disabled – container not in use. |
| 0001b | Xs = 0 |
| 0010b | Xs = 1 |
| 0100b | Xs = 2 |
| 1000b | Xs = 3 |

The number of containers depend on the used control word mapper. Both the **CPRI11.ctrl.slow** and **CPRI11.ctrl.fast** have only one container e.g., **CPRI11.ctrl.fast.container.** The combination of **.cw\_sel** and multiple containers allow handling of arbitrary areas within CPRI control words as illustrated in Figure 9.

[///Editor’s note: Range is a function of line rate. 64 is valid for 4.9GHz]

The .cw\_start defines the start of sub-channel (Ns) and has the valid range from 0 to 63. The .cw\_size defines the number of extracted sub-challes and the valid range is from 1 to 64. The mask defined by .cw\_sel applies to the “area” defined by the .cw\_start and .cw\_num, The specific mapper definitions may have more specific restriction to the ranges. If there are multiple containers the areas they define shall not overlap each other.

The **.flow\_id** maps to RoE header **flow\_id** field.

The .filter\_mode specifically concerns the control process mapper when it has to generate a RoE packet. See Table 7 for further details.

Table Hyper Frame filtering options

|  |  |
| --- | --- |
| .filter\_mode | 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 | Extracted content has changed since the previously generated RoE packet. |
| 4 | Pattern match. |

The .hfn\_modulo operates on the entire extracted (Hyper Frame size) CPRI control words area and combined with .hfn\_index allows selecting specific Hyper Frames for further processing. The .hfn\_modulo has the valid range from 1 to 150. The .hfn\_index has the valid range from 0 to 149. The modulo logic is synchronized with the current Hyper Frame Number (HFN). For example current\_HFN%.hfn\_modulo would select Hyper Frame control words for control process mapper processing when the reminder of the modulo operation equals to .hfn\_index.

The .offset, .value and .mask concern the pattern match .filter\_mode. The .offset has valid range from 0 to 4095. Both .value and .mask are 32 bit values. The pattern match is applied to the Hyper Frame after applying other parser options such as .cw\_sel, .cw\_start and .cw\_size. The offset is relative to the extracted (dense array of) control words. The match is true when the 32 bit value extracted from the memory equals to the .value after applying a “logical and” to it using the .mask.

Whenever parameter configurations refer to a value “p” that refers to the pointer in CPRI control words (see Figure 9) indicating the start of Fast C&M channel sub-channels.

#### Synchronization and L1 protocol fields

CPRI Synchronization and L1 protocol fields are not transported over the RoE. They are only provided for the local use by the “control process”. The following information is supported (using CPRI control word notation):

1. HFN (Hyperframe number) at location Z.64.0 i.e., control word 64.
2. BFN (CPRI 10ms frame number) at locations Z.128.0 and Z.192.0 i.e., control words 128 and 192.
3. Protocol version at location Z.2.0 i.e., control word 2.
4. HDLC bit rate at location Z.66.0 i.e., control word 66.
5. L1 signaling at location Z.130.0 i.e., control word 130.
6. Ethernet pointer at location Z.194.0 i.e., control word 194.

#### Slow C&M Packet (pkt\_type 000100b)

The Slow C&M channel shall have the following parameterization:

1. **CPRI11.ctrl.slow.container.cw\_start**=1
2. **CPRI11.ctrl.slow.container.cw\_num**=1
3. **CPRI11.ctrl.slow.container.flow\_id**=0

The Slow C&M channel should have the following parameterization:

1. **CPRI11.ctrl.slow.container.cw\_sel**=1111b
2. **CPRI11.ctrl.slow.container.filter\_mode**=0

The rest of the parameters depend on the deployment.

The Slow C&M channel content is transported over RoE data packets. The payload is described as follows and the parameters apply to control words as available for the “control process”:

1. **RoE.numSegments**=1
2. **RoE.segment.lenSkip=**0
3. **RoE.segment.lenSegment=**0
4. **RoE.numContainer**=1
5. **RoE.container[0].lenSkip**=0
6. **RoE.container[0].lenContainer**=**CPRI11.ctrl\_cw\_size**\* \*num\_of\_1s\_in\_**CPRI11.ctrl.slow.container.cw\_sel**
7. **RoE.container[0].flow\_id**=**CPRI11.ctrl.slow.container.flow\_id**
8. **RoE.container[0].ctrl**=0
9. **RoE.container[0].modulo**=0
10. **seqNumMinimum**=0
11. **seqNumMaximum**=150
12. **seqNumIncrement**=1

[///Editor’s note: if there is a need to count also radio frames then the size of the seqNumMaximum has to be increased. For example: seqNumManimum = 15000 would wrap after 100 BFNs i.e. have 1 sec worth of sequence number window.]

### Fast C&M channel packets

The “control process” shall extract the control words for the Fast C&M channel and create an appropriate Ethernet packet out of it. The Fast C&M channel is sent/received as native Ethernet traffic. The used Physical Coding Sublayer (PCS) shall be according to the underlying link.

The Fast C&M channel shall have the following parameterization:

1. **CPRI11.ctrl.fast.container.cw\_start**=**p** (see Figure 9)
2. **CPRI11.ctrl.fast.container.cw\_num**=64-**p**

The Fast C&M channel should have the following parameterization:

1. **CPRI11.ctrl.fast.container.cw\_sel**=1111b
2. **CPRI11.ctrl.fast.container.filter\_mode**=0

The rest of parameters are not needed.

FFS: handling of Ethernet packets with payload less than 64 octets. Proposal to zero pad the packet to minimum required 64 octets.

#### Use of sequence numbers with pkt\_type 000100b

The following shall apply:

1. **seqNumMinimum**=0
2. **seqNumMaximum**=150
3. **seqNumIncrement**=1

[///Editor’s note: JK: The outcome of sequence number discussion needs to be reflected. see sub-section 6.3.2.2 editor’s note for larger sequence number.]

#### Use of sequence numbers with RoE pkt\_type 000000b subtype 00000100b

The following shall apply:

1. **seqNumMinimum**=0
2. **seqNumMaximum**=150
3. **seqNumIncrement**=1

[///Editor’s note: see sub-section 6.3.2.2 editor’s note for larger sequence number.]

1. Header examples

[///Editor’s note: TEXT HERE]

1. Example algorithm for timestamp conversion

The following C-like pseudocode algorithm illustrates how to:

* convert the IEEE 1588v2-based presentation time into the 32 bit RoE timestamp value (presentation\_to\_timestamp)
* convert the 32 bit RoE timestamp value into an IEEE 1588v2-based (PTP-based) presentation time (timestamp\_to\_presentation)

The correct usage of the recovered presentation time from the 32 bit RoE timestamp value requires common agreement on the local time. The clocks of the RoE packet transmitter and receiver must be synchronized to a common reference.

In this example algorithm, the 2 bit fractional nanoseconds field is not used and is set to 00b by the RoE transmitter and is ignored by the RoE receiver and the presentation time is limited to a time that is less than 500,000,000ns after the timestamp was generated at the RoE packet transmitter.

typedef uint64\_t uint48\_t;

#define TSTAMPSZE 1000000000

#define WINDOWSZE  500000000

struct Timestamp {

    uint48\_t secondsField;

    uint32\_t nanosecondsField;

};

// Convert a PTP timestamp (the presentation time) to a 32 bit timestamp.

// The 2 fractional nanoseconds bits are forced to 00b.

uint32\_t presentation\_to\_timestamp( const struct Timestamp\* presPTP ) {

    return presPTP->nanosecondsField << 2;

}

// Convert a 32 bit timestamp to a PTP timestamp (the presentation time).

// The 2 fractional nanoseconds bits are ignored.

void timestamp\_to\_presentation( const struct Timestamp\* localPTP,

                                struct Timestamp\* presPTP,

                                uint32\_t ts ) {

    int32\_t diff;

    uint32\_t wrap;

    ts >>= 2;       // remove fractional nanoseconds

    diff = ts - localPTP->nanosecondsField;

    wrap = abs(diff) > WINDOWSZE ? 1 : 0;

    presPTP->secondsField = localPTP->secondsField;

    presPTP->nanosecondsField = ts;

    if (diff < 0) {

        presPTP->secondsField += wrap;

    } else {

        presPTP->secondsField -= wrap;

    }

}

1. Sequence number psudo code

 typedef struct {

    uint32\_t pval;

    uint32\_t qval;

    uint32\_t pmax;

    uint32\_t qmax;

    uint32\_t rsvd;

    uint8\_t q, p;

} seqnum\_t;

int initSeqNum( seqnum\_t\* sn,

                uint32\_t pmax, uint32\_t qmax,

                uint32\_t pval, uint32\_t qval,

                uint32\_t rsvd ) {

    // sanity checks

    if (pval >= pmax) return -1;

    if (qmax == 0 && qval != 0) return -1;

    // initialize sequence number..

    sn->pmax = pmax;

    sn->qmax = qmax;

    sn->pval = pval;

    sn->qval = qval;

    sn->p = pmax > 0 ? 1 : 0;

    sn->q = qmax > 0 ? 1 : 0;

    // calculate q and p

    while (pmax >>= 1) sn->p++;

    while (qmax >>= 1) sn->q++;

    // more sanity checks

    if (sn->p == 0) return -1;

    if (sn->p + sn->q > 32) return -1;

    // remaining initialization

    sn->rsvd = sn->q+sn->p >= 32 ? 0 : rsvd << sn->q+sn->p;

    return 0;

}

uint32\_t getSeqNum( const seqnum\_t\* sn ) {

    return  sn->rsvd | sn->qval << sn->p | sn->pval;

}

uint32\_t incSeqNum( seqnum\_t\* sn, uint32\_t pinc, uint32\_t qinc ) {

    int overflow = 0;

    if ((sn->pval += pinc) > sn->pmax) {

        sn->pval -= (sn->pmax+1);

        overflow = 1;

    }

    if (overflow && ((sn->qval += qinc) > sn->qmax)) {

        sn->qval -= (sn->qmax+1);

    }

    return getSeqNum( sn );

}

# Bibliography (informative)

[///Editor’s note: TEXT HERE]

1. IEEE Standards Dictionary Online subscription is available at <http://www.ieee.org/portal/innovate/products/standard/standards_dictionary.html>. [↑](#footnote-ref-1)